aws

개발자 가이드

AWS Data Pipeline

API 버전 2012-10-29

AWS Data Pipeline: 개발자 가이드

Table of Contents

AWS Data Pipeline 는 더 이상 신규 고객이 사용할 수 없습니다. 의 기존 고객은 평소와 같이 서비스를 계속 사용할 AWS Data Pipeline 수 있습니다. 자세히 알아보기

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.

AWS Data Pipeline란 무엇인가요?

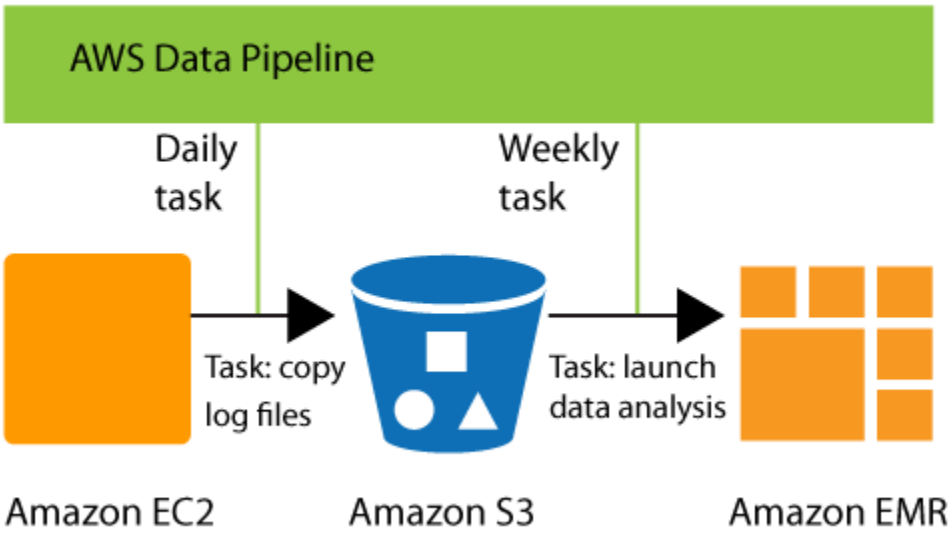
> **Note**
>
> AWS Data Pipeline 서비스가 유지 관리 모드에 있으며 새로운 기능이나 리전 확장은 계획되어 있지 않습니다. 기존 워크로드를 마이그레이션하는 방법에 대해 알아보고 발견하려면 에서 워크로드 마이그레이션 AWS Data Pipeline을(를) 참조하세요.

AWS Data Pipeline 는 데이터의 이동 및 변환을 자동화하는 데 사용할 수 있는 웹 서비스입니다. AWS Data Pipeline를 사용하면 이전 작업의 성공적인 완료에 따라 작업이 달라질 수 있도록 데이터 기반 워크플로를 정의할 수 있습니다. 데이터 변환의 파라미터를 정의하고 설정한 로직을 AWS Data Pipeline 적용합니다.

의 다음 구성 요소가 함께 AWS Data Pipeline 작동하여 데이터를 관리합니다.

- 파이프라인 정의에서는 데이터 관리의 비즈니스 로직을 지정합니다. 자세한 내용은 파이프라인 정의 파일 구문 단원을 참조하십시오.
- 파이프라인은 정의된 작업 활동을 수행할 Amazon EC2 인스턴스를 생성함으로써 작업을 예약하고 실행합니다. 파이프라인 정의를 파이프라인에 업로드한 다음 파이프라인을 활성화합니다. 파이프라인 실행을 위한 정의를 편집할 수 있는데, 효력을 발휘하려면 파이프라인을 다시 활성화해야 합니다. 파이프라인을 비활성화하고 데이터 원본을 수정한 다음 파이프라인을 다시 활성화할 수 있습니다. 파이프라인을 사용한 작업을 완료하면 이를 삭제할 수 있습니다.
- Task Runner는 작업에 대한 폴링을 수행한 다음 작업을 수행합니다. 예를 들어, Task Runner는 로그 파일을 Amazon S3에 복사하고 Amazon EMR 클러스터를 시작할 수 있습니다. Task Runner는 파이프라인 정의로 생성된 리소스에 자동으로 설치되고 실행됩니다. 사용자 지정 작업 실행기 애플리케이션을 작성하거나에서 제공하는 작업 실행기 애플리케이션을 사용할 수 있습니다 AWS Data Pipeline. 자세한 내용은 작업 실행기 단원을 참조하십시오.

예를 들어 AWS Data Pipeline 를 사용하여 매일 웹 서버의 로그를 Amazon Simple Storage Service(Amazon S3)에 아카이브한 다음 해당 로그에 대해 주간 Amazon EMR(Amazon EMR) 클러스터를 실행하여 트래픽 보고서를 생성할 수 있습니다.는 데이터를 복사하는 일일 작업과 Amazon EMR 클러스터를 시작하는 주간 작업을 AWS Data Pipeline 예약합니다. AWS Data Pipeline 또한 로그 업로드가 예기치 않게 지연되더라도 Amazon EMR이 분석을 시작하기 전에 최종 날짜의 데이터가 Amazon S3에 업로드될 때까지 대기하도록 합니다.



내용

에서 워크로드 마이그레이션 AWS Data Pipeline

AWS 는 2012년에 AWS Data Pipeline 서비스를 시작했습니다. 당시 고객은 다양한 컴퓨팅 옵션을 사용하여 서로 다른 데이터 소스 간에 데이터를 안정적으로 이동할 수 있는 서비스를 찾고 있었습니다. 이제 고객에게 더 나은 경험을 제공하는 다른 서비스도 있습니다. 예를 들어 AWS Glue 를 사용하여 Apache Spark 애플리케이션을 실행하고 오케스트레이션하거나, AWS Step Functions를 사용하여 AWS 서비스 구성 요소를 오케스트레이션하거나, Amazon Managed Workflows for Apache Airflow(Amazon MWAA)를 사용하여 Apache Airflow의 워크플로 오케스트레이션을 관리할 수 있습니다.

이 주제에서는에서 AWS Data Pipeline 대체 옵션으로 마이그레이션하는 방법을 설명합니다. 선택한 옵션은 현재 AWS Data Pipeline의 워크로드에 따라 다릅니다. 의 일반적인 사용 사례를 AWS Glue AWS Step Functions 또는 Amazon MWAA로 마이그레이션 AWS Data Pipeline 할 수 있습니다.

워크로드를 로 마이그레이션 AWS Glue

AWS Glue는 분석 사용자가 여러 소스의 데이터를 쉽게 검색, 준비, 이동, 통합할 수 있도록 하는 서버리스 데이터 통합 서비스입니다. 작성, 작업 실행, 워크플로 오케스트레이션을 위한 도구가 포함됩니다. 를 사용하면 70개 이상의 다양한 데이터 소스를 검색 및 연결하고 중앙 집중식 데이터 카탈로그에서 데이터를 관리할 AWS Glue수 있습니다. 추출, 변환, 로드(ETL) 파이프라인을 시각적으로 생성, 실행, 모니터링하여 데이터 레이크에 데이터를 로드할 수 있습니다. 또한 Amazon Athena, Amazon EMR, Amazon Redshift Spectrum을 사용하여 카탈로그화된 데이터를 즉시 검색하고 쿼리할 수 있습니다.

다음과 같은 AWS Glue 경우 AWS Data Pipeline 워크로드를 로 마이그레이션하는 것이 좋습니다.

- 다양한 데이터 소스, 비주얼 에디터 및 노트북을 포함한 저작 인터페이스, 데이터 품질 및 민감한 데이터 감지와 같은 고급 데이터 관리 기능을 지원하는 서버리스 데이터 통합 서비스를 찾고 있습니다.
- 워크로드를 AWS Glue 워크플로, 작업(Python 또는 Apache Spark) 및 크롤러(예: 기존 파이프라인은 Apache Spark를 기반으로 구축됨)로 마이그레이션할 수 있습니다.
- 수집, 처리, 전송, 무결성 테스트, 품질 검사 등 데이터 파이프라인의 모든 측면을 처리할 수 있는 단일 플랫폼이 필요합니다.
- 기존 파이프라인은 DynamoDB 테이블을 Amazon S3로 내보내는 등 AWS Data Pipeline 콘솔의 사전 정의된 템플릿에서 생성되었으며 동일한 목적 템플릿을 찾고 있습니다.
- 워크로드는 Apache Hive와 같은 특정 Hadoop 생태계 애플리케이션에 의존하지 않습니다.
- 워크로드에 온프레미스 서버를 오케스트레이션할 필요가 없습니다.

AWS 는 크롤러(데이터 검색) 및 ETL 작업(데이터 처리 및 로드)에 대해 초 단위로 청구되는 시간당 요금을 청구합니다. AWS Glue Studio는 AWS Glue 리소스를 위한 내장 오케스트레이션 엔진이며 추가 비용 없이 제공됩니다. 요금 책정에 대한 자세한 내용은 AWS Glue 요금 책정을 참조하세요.

워크로드를 AWS Step Functions로 마이그레이션

AWS Step Functions는 비즈니스 크리티컬 애플리케이션을 위한 워크플로를 구축할 수 있는 서버리스 오케스트레이션 서비스입니다. Step Functions를 사용하면 시각적 편집기를 사용하여 워크플로를 구축하고 AWS Lambda, Amazon EMR, DynamoDB 등과 같은 250개 이상의 AWS 서비스에 대한 11,000개 이상의 작업과 직접 통합할 수 있습니다. Step Functions를 사용하여 데이터 처리 파이프라인을 오케스트레이션하고, 오류를 처리하고, 기본 AWS 서비스에 대한 제한 한도 작업을 수행할 수 있습니다. 기계 학습 모델을 처리 및 게시하는 워크플로를 생성하고, 마이크로 서비스를 오케스트레이션할 수 있을 뿐만 아니라 추출, 변환 및 로드(ETL) 워크플로 AWS Glue를 생성하는 등의 제어 AWS 서비

스도 생성할 수 있습니다. 또한 사람의 상호 작용이 필요한 애플리케이션을 위해 오래 실행되는 자동화된 워크플로를 만들 수 있습니다.

마찬가지로 AWS Data Pipeline AWS Step Functions는에서 제공하는 완전 관리형 서비스입니다 AWS. 인프라 관리, 작업자 패치, OS 버전 업데이트 관리 등을 수행할 필요가 없습니다.

다음과 같은 경우 AWS Data Pipeline 워크로드를 AWS Step Functions로 마이그레이션하는 것이 좋습니다.

- 가용성이 뛰어난 서버리스 워크플로 오케스트레이션 서비스를 찾고 있습니다.
- 단일 작업 실행의 세분화로 비용을 청구할 수 있는 비용 효율적인 솔루션을 찾고 있습니다.
- 워크로드는 Amazon EMR, Lambda AWS Glue또는 DynamoDB와 같은 다른 여러 AWS 서비스에 대한 작업을 오케스트레이션합니다.
- 워크플로 생성을 위한 드래그 앤 드롭 비주얼 디자이너와 함께 제공되고 새로운 프로그래밍 개념을 배울 필요가 없는 로우 코드 솔루션을 찾고 계실 것입니다.
- 11,000개 이상의 작업을 기본적으로 처리하는 250개 이상의 다른 AWS 서비스와의 통합 out-of-the-box을 제공하고 사용자 지정 비AWS 서비스 및 활동과의 통합을 허용하는 서비스를 찾고 있습니다.

AWS Data Pipeline 및 Step Functions 모두 JSON 형식을 사용하여 워크플로를 정의합니다. 이를 통해 워크플로를 소스 제어에 저장하고, 버전을 관리하고, 액세스를 제어하고, CI/CD로 자동화할 수 있습니다. Step Functions는 완전히 JSON을 기반으로 하는 Amazon States Language라는 구문을 사용하며, 워크플로의 텍스트 표현과 시각적 표현 사이를 원활하게 전환할 수 있습니다.

Step Functions를 사용하면 현재 AWS Data Pipeline에서 사용 중인 것과 동일한 버전의 Amazon EMR을 선택할 수 있습니다.

AWS Data Pipeline 관리형 리소스에 대한 활동을 마이그레이션하려면 Step Functions의 AWS SDK 서비스 통합을 사용하여 리소스 프로비저닝 및 정리를 자동화할 수 있습니다.

온프레미스 서버, 사용자 관리형 EC2 인스턴스 또는 사용자 관리형 EMR 클러스터에서 활동을 마이그레이션하려면 인스턴스에 SSM 에이전트를 설치할 수 있습니다. Step Functions에서 AWS Systems Manager의 명령 실행 명령을 통해 명령을 시작할 수 있습니다. Amazon EventBridge에 정의된 스케줄에서 상태 머신을 시작할 수도 있습니다.

AWS Step Functions에는 표준 워크플로와 Express 워크플로라는 두 가지 유형의 워크플로가 있습니다. 표준 워크플로의 경우 애플리케이션을 실행하는 데 필요한 상태 전환 횟수를 기준으로 요금이 부과됩니다. 익스프레스 워크플로의 경우 워크플로에 대한 요청 수와 기간을 기준으로 요금이 부과됩니다. AWS Step Functions 가격 책정에서 가격 책정에 대해 자세히 알아보십시오.

워크로드를 Amazon MWAA로 마이그레이션하기

Amazon MWAA(Apache Airflow용 관리형 워크플로)는 Apache Airflow에 대한 관리형 오케스트레이션 서비스로 클라우드에서 종단 간 데이터 파이프라인을 대규모로 쉽게 설정하고 운영할 수 있습니다. Apache Airflow는 “워크플로”라고 하는 일련의 프로세스와 작업을 프로그래밍 방식으로 작성, 예약 및 모니터링하는 데 사용되는 오픈 소스 도구입니다. Amazon MWAA를 사용하면 확장성, 가용성 및 보안을 위해 기본 인프라를 관리할 필요 없이 Airflow 및 Python 프로그래밍 언어를 사용하여 워크플로를 생성할 수 있습니다. Amazon MWAA는 필요에 맞게 워크플로 실행 용량을 자동으로 확장하고 보안 AWS 서비스와 통합되어 데이터에 대한 빠르고 안전한 액세스를 제공합니다.

마찬가지로 AWS Data Pipeline Amazon MWAA는에서 제공하는 완전 관리형 서비스입니다 AWS. 이러한 서비스와 관련된 몇 가지 새로운 개념을 배워야 하지만, 인프라 관리, 작업자 패치, OS 버전 업데이트 관리 등은 필요하지 않습니다.

다음과 같은 경우 AWS Data Pipeline 워크로드를 Amazon MWAA로 마이그레이션하는 것이 좋습니다.

- Python에서 작성된 워크플로를 오케스트레이션할 수 있는 고가용성 관리형 서비스를 찾고 있습니다.
- 휴대성을 극대화하기 위해 완전히 관리되고 널리 채택되는 오픈 소스 기술인 Apache Airflow로 전환하고자 합니다.
- 수집, 처리, 전송, 무결성 테스트, 품질 검사 등 데이터 파이프라인의 모든 측면을 처리할 수 있는 단일 플랫폼이 필요합니다.
- 관찰성을 위한 풍부한 UI, 실패한 워크플로에 대한 재시작, 백필, 작업 재시도 등의 기능을 갖춘 데이터 파이프라인 오케스트레이션을 위해 설계된 서비스를 찾고 있습니다.
- 비서비스 AWS 뿐만 아니라 800개 이상의 사전 구축된 운영자 및 센서와 함께 제공되는AWS 서비스를 찾고 있습니다.

Amazon MWAA 워크플로는 Python을 사용하는 방향성 비순환 그래프(DAG)로 정의되므로, 이를 소스 코드로 취급할 수도 있습니다. Airflow의 확장 가능한 Python 프레임워크를 사용하면 거의 모든 기술과 연결되는 워크플로를 구축할 수 있습니다. 워크플로를 보고 모니터링할 수 있는 풍부한 사용자 인터페이스가 제공되며 버전 제어 시스템과 쉽게 통합되어 CI/CD 프로세스를 자동화할 수 있습니다.

Amazon MWAA를 사용하면 현재 AWS Data Pipeline에서 사용 중인 것과 동일한 버전의 Amazon EMR을 선택할 수 있습니다.

AWS 는 Airflow 환경이 실행되는 시간과 작업자 또는 웹 서버 용량을 늘리기 위한 추가 Auto Scaling에 대한 요금을 부과합니다. Apache Airflow에 대한 Amazon Managed Workflows 요금 정책에서 요금 정책에 대해 자세히 알아보십시오.

개념 매핑하기

다음 표에는 서비스에서 사용하는 주요 개념의 매핑이 나와 있습니다. Data Pipeline에 익숙한 사람들이 Step Functions와 MWAA 용어를 이해하는 데 도움이 될 것입니다.

Data Pipeline	글루	단계 함수	Amazon MWAA
파이프라인	워크플로	워크플로	다이렉트 아크릴 그래프
Pipeline 정의 JSON	워크플로 정의 또는 Python 기반 블루프린트	Amazon State Language JSON	Python 기반
활동	작업	상태 및 태스크	작업(운영자 및 센서)
인스턴스	작업 실행	실행	DAG 실행
Attempts	시도 재시도	캐처 및 리트라이어	재시도
파이프라인 일정	예약 트리거	EventBridge 스케줄러 작업	Cron, 시간표, 데이터 인식
파이프라인 표현식 및 함수	블루프린트 라이브러리	Step Functions의 내장 함수 및 AWS Lambda	확장 가능한 Python 프레임워크

샘플

다음 섹션에는에서 개별 서비스로 마이그레이션할 때 참조할 수 있는 공개 예제 AWS Data Pipeline 가 나열되어 있습니다. 이를 예제로 참조하고 사용 사례에 따라 업데이트하고 테스트하여 개별 서비스에 자체 파이프라인을 구축할 수 있습니다.

AWS Glue 샘플

다음 목록에는 가장 일반적인 AWS Data Pipeline 사용 사례에 대한 샘플 구현이 포함되어 있습니다 AWS Glue.

- Spark 작업 실행
- JDBC에서 Amazon S3로 데이터 복사(Amazon Redshift 포함)
- Amazon S3에서 JDBC로 데이터 복사(Amazon Redshift 포함)
- Amazon S3에서 DynamoDB로 데이터 복사
- Amazon Redshift 간 데이터 이동
- DynamoDB 테이블에 대한 교차 계정 교차 리전 액세스

AWS Step Functions 샘플

다음 목록에는 AWS Step Functions의 가장 일반적인 AWS Data Pipeline 사용 사례에 대한 샘플 구현이 포함되어 있습니다.

- Amazon EMR 작업 관리
- Amazon EMR Serverless에서 데이터 처리 작업 실행
- Hive/Pig/Hadoop 작업 실행
- 대규모 데이터 세트 쿼리(Amazon Athena, Amazon S3, AWS Glue)
- Amazon Redshift를 사용하는 ETL 워크플로 실행
- AWS Glue 크롤러 오케스트레이션

AWS Step Functions 사용에 대한 추가 자습서 및 샘플 프로젝트를 참조하세요.

Amazon MWAA 샘플

다음 목록에는 Amazon MWAA의 가장 일반적인 AWS Data Pipeline 사용 사례에 대한 샘플 구현이 포함되어 있습니다.

- Amazon EMR 작업 실행
- Apache Hive 및 Hadoop용 사용자 지정 플러그인 생성
- Amazon S3에서 Redshift로 데이터 복사
- 원격 EC2 인스턴스에서 쉘 스크립트 실행

- 하이브리드(온프레미스) 워크플로 오케스트레이션

Amazon MWAA를 사용하기 위한 추가 자습서 및 샘플 프로젝트를 참조하세요.

관련 서비스

AWS Data Pipeline 는 다음 서비스와 함께 작동하여 데이터를 저장합니다.

- Amazon DynamoDB — 저비용에 빠른 성능으로 완벽하게 관리되는 NoSQL 데이터베이스를 제공합니다. 자세한 내용은 Amazon DynamoDB 개발자 안내서를 참조하세요.
- Amazon RDS — 대량의 데이터 세트로 확장되며 완벽하게 관리되는 관계형 데이터베이스를 제공합니다. 자세한 내용은 Amazon Relational Database Service 개발자 안내서를 참조하세요.
- Amazon Redshift —빠르고 완벽하게 관리되는 페타바이트 규모의 데이터 웨어하우스로 경제적이고 쉽게 방대한 양의 데이터를 분석할 수 있습니다. 자세한 내용은 Amazon Redshift Database 개발자 안내서를 참조하세요.
- Amazon S3 — 안전하고, 안정적이며, 확장성이 뛰어난 객체 스토리지를 제공합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서를 참조하세요.

AWS Data Pipeline 는 다음 컴퓨팅 서비스와 함께 작동하여 데이터를 변환합니다.

- Amazon EC2 —소프트웨어 시스템의 구축 및 호스팅에 사용할 수 있도록 규모를 변경할 수 있는 컴퓨팅 용량, 즉 Amazon 데이터 센터의 서버를 제공합니다. 자세한 내용은 Amazon EC2 사용 설명서를 참조하세요.
- Amazon EMR — Apache Hadoop 또는 Apache Spark와 같은 프레임워크를 사용하여 Amazon EC2 서버에 있는 방대한 양의 데이터를 용이하고 신속하고 비용 효율적으로 배포하고 처리할 수 있도록 해 줍니다. 자세한 내용은 Amazon EMR 개발자 가이드를 참조하세요.

액세스 AWS Data Pipeline

다음 인터페이스 중 하나를 사용하여 파이프라인을 생성하고, 액세스하고, 관리할 수 있습니다.

- AWS Management Console— AWS Data Pipeline에 액세스할 때 사용할 수 있는 웹 인터페이스를 제공합니다.
- AWS Command Line Interface (AWS CLI) - Windows, macOS 및 Linux AWS Data Pipeline를 포함하여 다양한 AWS 서비스에 대한 명령을 제공합니다. 설치에 대한 자세한 내용은 단원을 AWS CLI참

조하십시오AWS Command Line Interface. 에 대한 명령 목록은 datapipeline을 AWS Data Pipeline 참조하세요.

- AWS SDK — 언어별 API를 제공하고, 서명 계산, 요청 재시도 처리 및 오류 처리와 같은 많은 연결 세부 정보를 관리합니다. 자세한 정보는 AWS SDK를 참조하세요.
- 쿼리 API—HTTPS 요청을 사용하여 호출하는 하위 수준의 API를 제공합니다. 쿼리 API 사용은 AWS Data Pipeline에 액세스하는 가장 직접적인 방법이지만, 애플리케이션에서 요청에 서명할 해시 생성 및 오류 처리와 같은 하위 수준의 세부 정보를 처리해야 합니다. 자세한 내용은 AWS Data Pipeline API 참조를 참조하세요.

가격 책정

Amazon Web Services에서는 사용한 만큼만 비용을 지불하며, 의 경우 활동 및 사전 조건이 실행되도록 예약된 빈도와 실행 위치에 따라 파이프라인에 대한 AWS Data Pipeline비용을 지불합니다. 자세한 내용은 AWS Data Pipeline 요금을 참조하세요.

AWS 계정을 만든 지가 12개월이 안 된 경우에는 프리 티어를 이용할 수 있습니다. 프리 티어에는 월간 3건의 저빈도 사전 조건 및 5건의 저빈도 활동 무상 이용권이 포함됩니다. 자세한 내용은 AWS 프리 티어 단원을 참조하세요.

파이프라인 작업 활동에 대해 지원되는 인스턴스 유형

가 파이프라인을 AWS Data Pipeline 실행하면 파이프라인 구성 요소를 컴파일하여 실행 가능한 Amazon EC2 인스턴스 세트를 생성합니다. 각 인스턴스는 특정 작업 수행에 필요한 모든 정보를 포함합니다. 인스턴스의 전체 세트는 파이프라인의 할 일 목록입니다. AWS Data Pipeline 은 인스턴스를 작업 실행기에 분배하여 처리하도록 합니다.

EC2 인스턴스는 인스턴스 유형이라고 하는 다양한 구성으로 제공됩니다. 각 인스턴스 유형마다 CPU, 입/출력 및 스토리지 용량이 다릅니다. 활동을 위한 인스턴스 유형을 지정하는 것 외에도, 다양한 구매 옵션을 선택할 수 있습니다. AWS 리전에 따라 일부 인스턴스 유형은 사용할 수 없습니다. 사용할 수 없는 인스턴스 유형을 사용하여 클러스터를 생성하는 경우, 파이프라인이 프로비저닝에 실패하거나 프로비저닝이 멈출 수 있습니다. 인스턴스 가용성에 대한 자세한 내용은 Amazon EC2 가격 책정 페이지를 참조하세요. 해당 리전에서 인스턴스 유형을 사용할 수 있는 알아보기 위해 인스턴스 구매 옵션 링크를 열고 리전별로 필터링합니다. 이러한 인스턴스 유형, 패밀리 및 가상화 유형에 대한 자세한 내용에 대해서는 Amazon EC2 인스턴스 및 Amazon Linux AMI 인스턴스 유형 매트릭스를 참조하세요.

다음 표에서는가 AWS Data Pipeline 지원하는 인스턴스 유형을 설명합니다. AWS Data Pipeline 를 사용하여 AWS Data Pipeline 가 지원되지 않는 리전을 포함하여 모든 리전에서 Amazon EC2 인스턴스

를 시작할 수 있습니다. AWS Data Pipeline 가 지원되는 리전에 대한 자세한 내용은 AWS 리전 및 엔드포인트를 참조하세요.

내용

AWS 리전에 의한 기본 Amazon EC2 인스턴스

파이프라인 정의에서 인스턴스 유형을 지정하지 않는 경우 AWS Data Pipeline 에서 기본적으로 인스턴스를 실행합니다.

다음 표에는가 지원되는 리전에서가 기본적으로 AWS Data Pipeline 사용하는 Amazon EC2 인스턴스가 나열되어 AWS Data Pipeline 있습니다.

리전 이름	리전	인스턴스 유형
미국 동부(버지니아 북부)	us-east-1	m1.small
미국 서부(오레곤)	us-west-2	m1.small
아시아 태평양(시드니)	ap-southeast-2	m1.small
아시아 태평양(도쿄)	ap-northeast-1	m1.small
EU(아일랜드)	eu-west-1	m1.small

다음 표에는가 지원되지 않는 리전에서가 기본적으로 AWS Data Pipeline 시작하는 Amazon EC2 인스턴스 AWS Data Pipeline 가 나열되어 있습니다.

리전 이름	리전	인스턴스 유형
미국 동부(오하이오)	us-east-2	t2.small
미국 서부(캘리포니아 북부 지역)	us-west-1	m1.small

리전 이름	리전	인스턴스 유형
아시아 태평양(뭄바이)	ap-south-1	t2.small
아시아 태평양(싱가포르)	ap-southeast-1	m1.small
아시아 태평양(서울)	ap-northeast-2	t2.small
캐나다(중부)	ca-central-1	t2.small
EU(프랑크푸르트)	eu-central-1	t2.small
EU(런던)	eu-west-2	t2.small
EU(파리)	eu-west-3	t2.small
남아메리카(상파울루)	sa-east-1	m1.small

추가적으로 지원된 Amazon EC2 인스턴스

파이프라인 정의에서 인스턴스 유형을 지정하지 않는 경우 생성되는 기본 인스턴스 외에도, 다음 인스턴스가 지원됩니다.

다음 표에는가 AWS Data Pipeline 지원하고 지정된 경우 생성할 수 있는 Amazon EC2 인스턴스가 나열되어 있습니다.

인스턴스 클래스	인스턴스 유형
범용	t2.nano | t2.micro | t2.small | t2.medium | t2.large
컴퓨팅 최적화	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
메모리 최적화	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 | m5.24xlar

인스턴스 클래스	인스턴스 유형
	ge | 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
스토리지 최적화	i2.xlarge | i2.2xlarge | i2.4xlarge | i2.8xlarge | hs1.8xlarge | g2.2xlarge | g2.8xlarge | d2.xlarge | d2.2xlarge | d2.4xlarge | d2.8xlarge

지원된 Amazon EMR 클러스터에 대한 Amazon EC2 인스턴스

이 표에는 지정된 경우가 Amazon EMR 클러스터에 대해를 지원하고 생성할 수 있는 Amazon EC2 인스턴스가 나열되어 있습니다. AWS Data Pipeline 자세한 내용을 알아보려면 Amazon EMR 관리 안내서의 지원되는 인스턴스 유형을 참조하세요.

인스턴스 클래스	인스턴스 유형
범용	m1.small | m1.medium | m1.large | m1.xlarge | m3.xlarge | m3.2xlarge
컴퓨팅 최적화	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
메모리 최적화	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 | r4.large | r4.xlarge | r4.2xlarge | r4.4xlarge | r4.8xlarge | r4.16xlarge

인스턴스 클래스	인스턴스 유형
스토리지 최적화	h1.4xlarge | hs1.2xlarge | hs1.4xlarge| hs1.8xlarge | i2.xlarge | i2.2xlarge | i2.4large | i2.8xlarge | d2.xlarge | d2.2xlarge| d2.4xlarge | d2.8xlarge
액셀러레이티드 컴퓨팅	g2.2xlarge | cg1.4xlarge

AWS Data Pipeline 개념

시작하기 전에의 주요 개념과 구성 요소에 대해 읽어보세요 AWS Data Pipeline.

내용

Pipeline Definition

파이프라인 정의는 비즈니스 로직을에 전달하는 방법입니다 AWS Data Pipeline. 여기에는 다음 정보가 포함됩니다.

- 데이터 원본의 이름, 위치 및 형식
- 데이터를 변환하는 활동
- 변환 작업의 일정
- 활동 및 사전 조건을 실행하는 리소스
- 활동 예약 전에 충족해야 하는 사전 조건
- 파이프라인 실행이 진행되는 동안 상태 업데이트를 알리는 방법

파이프라인 정의에서는 작업을 AWS Data Pipeline 결정하고, 예약하고, 작업 실행기에 할당합니다. 작업이 성공적으로 완료되지 않으면는 지침에 따라 작업을 AWS Data Pipeline 재시도하고 필요한 경우 다른 작업 실행기에 다시 할당합니다. 작업이 반복적으로 실패하면 파이프라인이 알림 메시지를 보내도록 구성할 수 있습니다.

예를 들어, 파이프라인 정의에서 애플리케이션에 의해 생성된 로그 파일이 2013년도에 매월 Amazon S3 버킷에 보관되도록 지정할 수 있습니다. 그러면 AWS Data Pipeline 가 해당 월의 일수가 30일이든, 31일이든, 28일이든, 29일이든 상관없이 각각 한 달 치의 데이터에 대해 복사를 수행하는 12개 작업을 생성합니다.

파이프라인 정의를 생성하는 방법은 다음과 같습니다.

- AWS Data Pipeline 콘솔을 사용하여 그래픽으로 표시
- 명령줄 인터페이스에 의해 사용되는 형식의 JSON 파일을 작성하여 텍스트 방식으로 정의 생성
- AWS SDK 또는 AWS Data Pipeline API 중 하나를 사용하여 웹 서비스를 호출하는 프로그래밍 방식으로 정의 생성

파이프라인 정의는 다음과 같은 유형의 구성요소를 포함할 수 있습니다.

파이프라인 구성요소

데이터 노드

작업에 사용되는 입력 데이터의 위치 또는 출력 데이터를 저장할 위치

활동

컴퓨팅 리소스와 일반적인 입력 및 출력 데이터 노드를 사용하여 일정대로 수행할 작업의 정의

사전 조건

작업 실행 전에 반드시 충족되어야 할 조건문.

리소스

파이프라인에서 정의한 작업을 수행하는 컴퓨팅 리소스입니다.

작업

지정된 조건(예: 활동 실패)이 충족될 때 트리거되는 작업입니다.

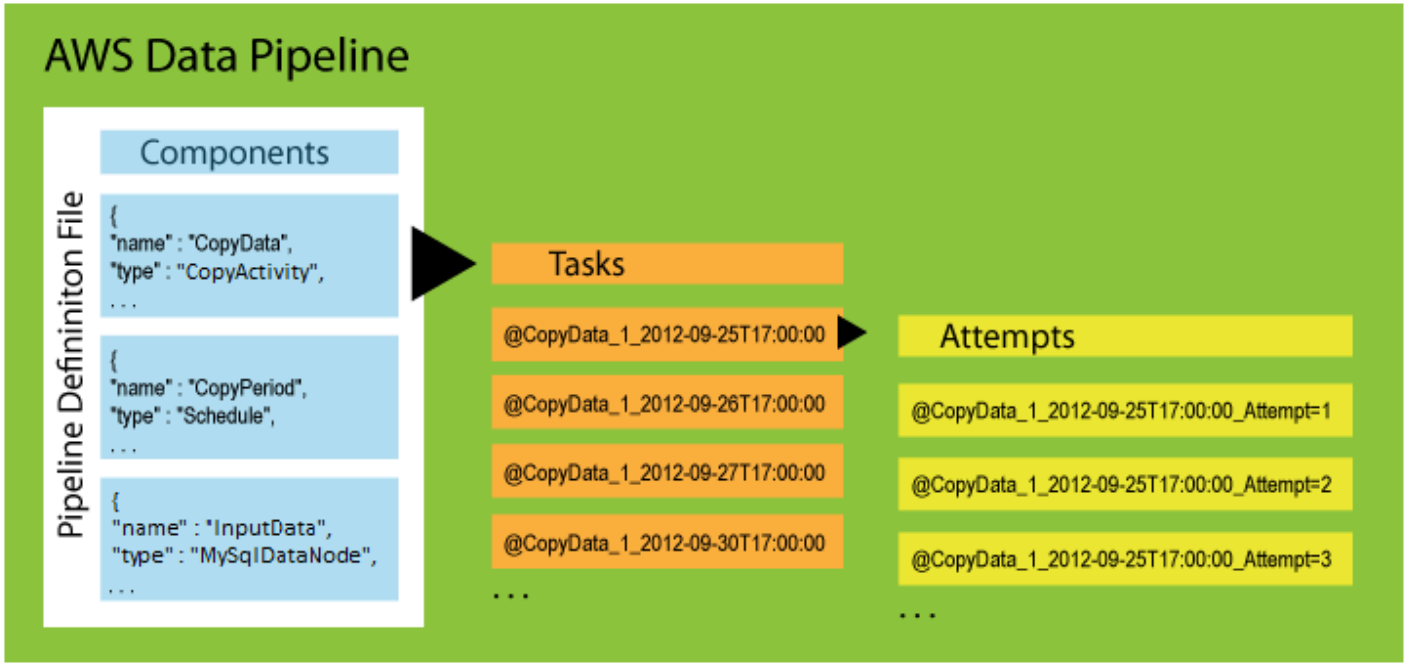
자세한 내용은 파이프라인 정의 파일 구문 단원을 참조하십시오.

파이프라인 구성요소, 인스턴스 및 시도

다음과 같은 세 가지 유형의 항목이 예약된 파이프라인과 연결됩니다.

- 파이프라인 구성요소 — 파이프라인 구성요소는 파이프라인의 비즈니스 로직을 나타내며 파이프라인의 다양한 섹션에 의해 표현됩니다. 파이프라인 구성요소는 워크플로우의 데이터 원본, 활동, 일정 및 사전 조건을 지정합니다. 상위 구성요소로부터 속성을 상속받을 수 있습니다. 구성요소 간의 관계는 참조에 의해 정의됩니다. 파이프라인 구성 요소는 데이터 관리의 규칙을 정의합니다.
- 인스턴스 -가 파이프라인을 AWS Data Pipeline 실행하면 파이프라인 구성 요소를 컴파일하여 실행 가능한 인스턴스 세트를 생성합니다. 각 인스턴스는 특정 작업 수행에 필요한 모든 정보를 포함합니다. 전체 인스턴스 세트는 파이프라인의 할 일 목록입니다.는 처리할 작업 실행기에 인스턴스를 AWS Data Pipeline 전달합니다.
- 시도— 견실한 데이터 관리를 위해 AWS Data Pipeline 는 실패한 작업을 재시도합니다. 작업이 허용 최대 재시도 횟수에 도달할 때까지 재시도를 계속하여 반복합니다. 시도 객체는 다양한 시도, 결과 및 실패 사유(해당하는 경우)를 추적합니다. 기본적으로 카운터가 있는 인스턴스입니다.는 Amazon EMR 클러스터 및 EC2 인스턴스와 같은 이전 시도의 동일한 리소스를 사용하여 재시도를 AWS Data Pipeline 수행합니다.

> **Note**
>
> 재시도 실패 작업은 내결함성 전략의 중요한 부분으로 AWS Data Pipeline 정의는 재시도 제어에 관한 조건 및 임계값을 제공합니다. 그러나 재시도 횟수가 너무 많을 경우 AWS Data Pipeline 이 지정된 재시도 횟수를 모두 소진할 때까지는 장애를 보고하지 않기 때문에 복구 불가능한 장애의 감지가 지연될 수 있습니다. 허용 횟수를 초과한 재시도가 AWS 리소스 상에서 실행되는 경우에는 추가 요금이 발생할 수 있습니다. 따라서 재시도 및 관련 설정을 제어하는 데 사용하는 AWS Data Pipeline 기본 설정을 초과하는 것이 적절한 시기를 신중하게 고려하세요.

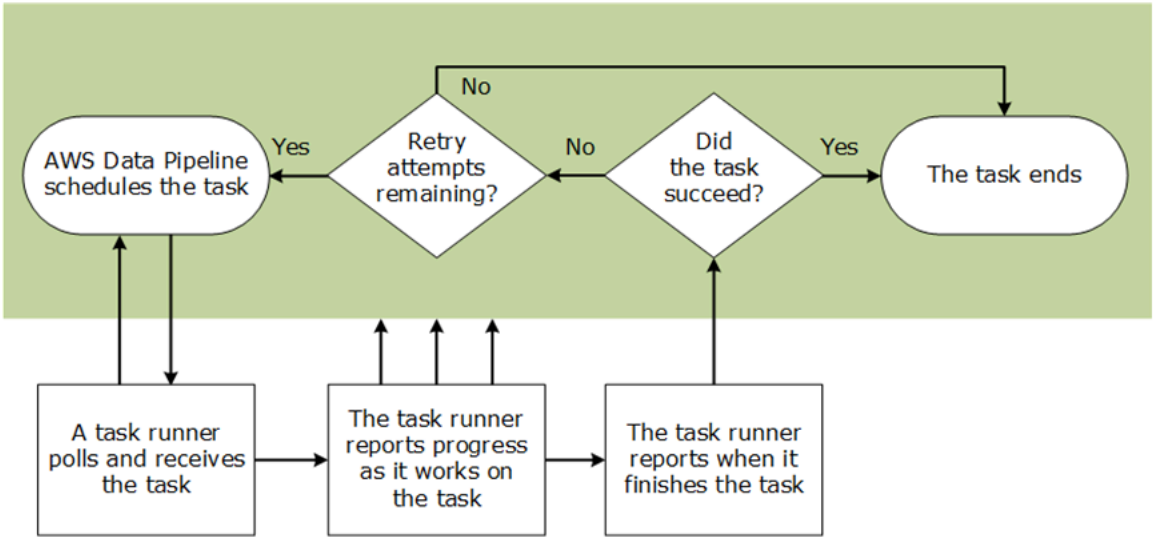


작업 실행기

작업 실행기는 작업을 AWS Data Pipeline 폴링한 다음 해당 작업을 수행하는 애플리케이션입니다.

AWS Data Pipeline가 Task Runner에서 제공하는 기본적인 작업 실행기입니다. Task Runner가 설치 및 구성되면 활성화한 파이프라인과 연결된 작업을 AWS Data Pipeline 폴링합니다. 작업이 Task Runner에 할당되어 있을 경우, 해당 작업을 수행하고 상태를 다시 AWS Data Pipeline로 전송합니다.

다음 다이어그램은 AWS Data Pipeline 및 작업 실행기가 상호 작용하여 예약된 작업을 처리하는 방법을 보여줍니다. 작업은 AWS Data Pipeline 서비스가 작업 실행기와 공유하는 개별 작업 단위입니다. 작업은 보통 여러 작업을 산출하는 활동 및 리소스의 일반적인 정의인 파이프라인과 다릅니다.



Task Runner를 사용하여 파이프라인을 처리할 수 있는 두 가지 방법이 있습니다.

- AWS Data Pipeline 는 AWS Data Pipeline 웹 서비스에서 시작하고 관리하는 리소스에 Task Runner를 설치합니다.
- 오랫동안 실행되는 EC2 인스턴스나 온프레미스 서버와 같이 사용자가 관리하는 컴퓨팅 리소스에는 Task Runner를 직접 설치합니다.

작업 실행기 작업에 대한 자세한 내용은 Task Runner로 작업하기 단원을 참조하세요.

데이터 노드

에서 데이터 노드 AWS Data Pipeline는 파이프라인 활동이 입력 또는 출력으로 사용하는 데이터의 위치와 유형을 정의합니다.는 다음 유형의 데이터 노드를 AWS Data Pipeline 지원합니다.

DynamoDBDataNode

HiveActivity 또는 EmrActivity이(가) 사용할 데이터를 포함하는 DynamoDB 테이블.

SqlDataNode

파이프라인 활동이 사용할 데이터를 나타내는 SQL 테이블 및 데이터베이스 쿼리.

> **Note**
>
> 이전에는 MySqlDataNode가 사용되었습니다. 대신 SqlDataNode를 사용합니다.

RedshiftDataNode

RedshiftCopyActivity이(가) 사용할 데이터를 포함하는 Amazon Redshift 테이블.

S3DataNode

파이프라인 활동이 사용할 하나 이상의 파일을 포함하는 An Amazon S3 위치.

데이터베이스 수

AWS Data Pipeline 는 다음과 같은 유형의 데이터베이스를 지원합니다.

JdbcDatabase

JDBC 데이터베이스.

RdsDatabase

Amazon RDS 데이터베이스.

RedshiftDatabase

Amazon Redshift 데이터베이스.

활동

에서 AWS Data Pipeline활동은 수행할 작업을 정의하는 파이프라인 구성 요소입니다.는 한 위치에서 다른 위치로 데이터 이동, Hive 쿼리 실행 등과 같은 일반적인 시나리오를 수용하는 몇 가지 사전 패키징된 활동을 AWS Data Pipeline 제공합니다. 활동은 확장 가능하므로, 자체의 사용자 지정 스크립트를 실행하여 무한한 조합을 지원할 수 있습니다.

AWS Data Pipeline 는 다음과 같은 유형의 활동을 지원합니다.

CopyActivity

한 위치에서 다른 위치로의 데이터 복사.

EmrActivity

Amazon EMR 클러스터 실행.

HiveActivity

Amazon EMR 클러스터 상에서의 Hive 쿼리를 실행합니다.

HiveCopyActivity

고급 데이터 필터링과 S3DataNode 및 DynamoDBDataNode이(가) 지원되는 Amazon EMR 클러스터에서 Hive 쿼리를 실행합니다.

PigActivity

Amazon EMR 클러스터에서 Pig 스크립트를 실행합니다.

RedshiftCopyActivity

Amazon Redshift에/로부터 복사.

ShellCommandActivity

사용자 지정 UNIX/Linux 셸 명령을 활동으로 실행합니다.

SqlActivity

데이터베이스에서 SQL 쿼리를 실행합니다.

일부 활동은 데이터 및 데이터베이스 스테이징을 특별히 지원합니다. 자세한 내용은 파이프라인 활동으로 데이터 및 테이블 준비 단원을 참조하십시오.

사전 조건

에서 AWS Data Pipeline사전 조건은 활동을 실행하기 전에 true여야 하는 조건문이 포함된 파이프라인 구성 요소입니다. 예를 들어 사전 조건은 파이프라인 활동이 복사를 시도하기 전에 소스 데이터가 있는지 확인할 수 있습니다.는 데이터베이스 테이블 존재 여부, Amazon S3 키 존재 여부 등 일반적인 시나리오를 수용하는 여러 사전 패키징된 사전 조건을 AWS Data Pipeline 제공합니다. 그러나 사전 조건은 확장 가능하므로, 자체의 사용자 지정 스크립트를 실행하여 무한한 조합을 지원할 수 있습니다.

사전 조건에는 두 가지 유형의 사전 조건(시스템 관리형 사전 조건과 사용자 관리형 사전 조건)이 있습니다. 시스템 관리형 사전 조건은 사용자를 대신하여 AWS Data Pipeline 웹 서비스에서 실행되며 컴퓨팅 리소스가 필요하지 않습니다. 사용자 관리형 사전 조건은 `runsOn` 및 `workerGroup` 필드를 사용하여 사용자가 지정한 컴퓨팅 리소스에서만 실행됩니다. `workerGroup` 리소스는 사전 조건을 사용하는 활동으로부터 추출됩니다.

시스템 관리형 사전 조건

DynamoDBDataExists

특정 DynamoDB 테이블에 데이터가 존재하는지 확인합니다.

DynamoDBTableExists

DynamoDB 테이블이 존재하는지 확인합니다.

S3KeyExists

Amazon S3 키가 존재하는지 확인합니다.

S3PrefixNotEmpty

Amazon S3 접두사가 비어 있는지 확인합니다.

사용자 관리형 사전 조건

존재함

데이터 노드가 존재하는지 확인합니다.

ShellCommandPrecondition

사용자 지정 UNIX/Linux 셸 명령을 사전 조건으로 실행합니다.

리소스

에서 AWS Data Pipeline리소스는 파이프라인 활동이 지정하는 작업을 수행하는 컴퓨팅 리소스입니다. 는 다음 유형의 리소스를 AWS Data Pipeline 지원합니다.

Ec2Resource

파이프라인 활동에 의해 정의된 작업을 수행하는 EC2 인스턴스입니다.

EmrCluster

EmrActivity과(와) 같은 파이프라인 활동에 의해 정의된 작업을 수행하는 Amazon EMR 클러스터입니다.

리소스는 작업 대상 데이터 세트와 동일한 리전에서 실행 가능합니다(AWS Data Pipeline과 다른 리전이라도 가능). 자세한 내용은 여러 리전의 리소스와 파이프라인 사용 단원을 참조하십시오.

리소스 제한

AWS Data Pipeline 는 많은 수의 동시 작업을 수용하도록 확장되며 대규모 워크로드를 처리하는 데 필요한 리소스를 자동으로 생성하도록 구성할 수 있습니다. 이렇게 자동으로 생성된 리소스는 사용자가

제어할 수 있으며, AWS 계정 리소스 제한을 기준으로 계수할 수 있습니다. 예를 들어 데이터를 처리하기 AWS Data Pipeline 위해 20노드 Amazon EMR 클러스터를 자동으로 생성하도록를 구성하고 AWS 계정에 EC2 인스턴스 제한이 20으로 설정된 경우 사용 가능한 채우기 리소스가 실수로 소진될 수 있습니다. 따라서 디자인할 때 이러한 리소스 제한을 고려하거나 계정 한계를 알맞게 늘리는 것이 좋습니다. 서비스 할당량에 대한 자세한 내용은 AWS 일반 참조의 AWS 서비스 할당량을 참조하세요.

> **Note**
>
> 한도는 `Ec2Resource` 구성요소 객체당 인스턴스 하나입니다.

지원되는 플랫폼

파이프라인은 다음 플랫폼에서 리소스를 시작할 수 있습니다.

EC2-Classic

다른 고객과 공유하는 단일 일반 네트워크에서 리소스가 실행됩니다.

EC2-VPC

AWS 계정에 속하도록 논리적으로 독립된 Virtual Private Cloud(VPC)에서 리소스가 실행됩니다.

AWS 계정에서는 리전 별로 두 가지 플랫폼 모두 또는 EC2-VPC에서만 리소스를 시작할 수 있습니다. 자세한 내용은 Amazon EC2 사용 설명서의 지원되는 플랫폼을 참조하세요.

AWS 계정에서 EC2-VPC만 지원하는 경우 각 AWS 리전에서 기본 VPC가 자동으로 생성됩니다. 기본적으로 리소스는 기본 VPC의 기본 서브넷에서 시작됩니다. 아니면 기본이 아닌 VPC를 생성하고 리소스를 구성할 때 VPC의 서브넷 중 하나를 지정한 다음, 기본이 아닌 VPC의 지정된 서브넷에서 리소스를 시작합니다.

VPC에서 인스턴스를 시작할 경우 해당 VPC 전용으로 생성되는 보안 그룹을 지정해야 합니다. VPC에서 인스턴스를 시작하는 경우 EC2-Classic용으로 생성된 보안 그룹은 지정할 수 없습니다. 또한 보안 그룹 이름이 아닌 보안 그룹 ID를 사용하여 VPC의 보안 그룹을 식별해야 합니다.

Amazon EMR 클러스터 및를 사용하는 Amazon EC2 스팟 인스턴스 AWS Data Pipeline

파이프라인은 해당 Amazon EMR 클러스터 리소스에서 Amazon EC2 스팟 인스턴스를 태스크 노드로 사용할 수 있습니다. 기본적으로 파이프라인은 온디맨드 인스턴스를 사용합니다. 스팟 인스턴스를 통

해 예비 EC2 인스턴스를 사용하고 실행할 수 있습니다. 스팟 인스턴스 요금 모델은 온디맨드 및 예약 인스턴스 요금 모델을 보완한 것으로서, 애플리케이션에 따라서는 가장 경제적으로 컴퓨팅 용량을 확보할 수 있는 방법입니다. 자세한 내용은 Amazon EC2 스팟 인스턴스 제품 페이지를 참조하세요.

스팟 인스턴스를 사용하는 경우는 클러스터가 시작될 때 Amazon EMR에 스팟 인스턴스 최고 가격을 AWS Data Pipeline 제출합니다. 클러스터의 작업을 `taskInstanceCount` 필드를 사용하여 정의한 스팟 인스턴스 태스크 노드 수에 자동으로 할당합니다. 작업 노드의 스팟 인스턴스를 AWS Data Pipeline 제한하여 온디맨드 코어 노드를 파이프라인을 실행할 수 있도록 합니다.

실패 혹은 완료한 파이프라인 리소스 인스턴스를 편집하여 스팟 인스턴스를 추가할 수 있습니다. 그러면 파이프라인은 클러스터를 다시 시작할 때 작업 노드에 스팟 인스턴스를 사용합니다.

스팟 인스턴스 고려 사항

에서 스팟 인스턴스를 사용하는 경우 다음 AWS Data Pipeline고려 사항이 적용됩니다.

- 스팟 인스턴스 가격이 해당 인스턴스의 최고 가격을 초과하거나 Amazon EC2 용량 문제가 있으면 스팟 인스턴스가 종료될 수 있습니다. 그러나는 항상 온디맨드 인스턴스이고 종료되지 않는 코어 노드가 있는 클러스터를 AWS Data Pipeline 사용하기 때문에 데이터가 손실되지 않습니다.
- 스팟 인스턴스는 용량을 비동기식으로 채워 가기 때문에 시작하는 데 시간이 더 오래 걸릴 수 있습니다. 그러므로 스팟 인스턴스 파이프라인은 비슷한 온디맨드 인스턴스 파이프라인에 비해 실행 시간이 더 길어지기도 합니다.
- 스팟 인스턴스를 받지 못하면(예: 최고 가격이 너무 낮은 경우) 클러스터가 실행되지 않을 수 있습니다.

작업

AWS Data Pipeline 작업은 성공, 실패 또는 지연 활동과 같은 특정 이벤트가 발생할 때 파이프라인 구성 요소가 수행하는 단계입니다. 활동의 이벤트 필드는 특정 작업(`EmrActivity`의 `onLateAction` 필드에 있는 `snsalarm` 참조)을 가리킵니다.

AWS Data Pipeline 는 파이프라인 및 해당 구성 요소의 상태를 무인 방식으로 나타내는 기본 방법으로 Amazon SNS 알림을 사용합니다. 자세한 내용은 Amazon SNS를 참조하세요. SNS 알림 외에도 AWS Data Pipeline 콘솔 및 CLI를 사용하여 파이프라인 상태 정보를 얻을 수 있습니다.

AWS Data Pipeline 는 다음 작업을 지원합니다.

SnsAlarm

onSuccess, OnFail 및 onLateAction 이벤트를 바탕으로 주제에 SNS 알림을 전송하는 작업.

Terminate

대기 중이거나 완료되지 않은 활동, 리소스 또는 데이터 노드의 취소를 트리거하는 작업. onSuccess, OnFail 또는 onLateAction을 포함하는 작업은 종료할 수 없습니다.

파이프라인 사전 모니터링

문제를 탐지하는 최선의 방법은 처음부터 파이프라인을 미리 모니터링하는 것입니다. 파이프라인 구성 요소가 실패하거나 예약된 시작 시간에 시작되지 않는 경우와 같은 특정 상황 또는 이벤트를 알리도록 파이프라인 구성 요소를 구성할 수 있습니다. AWS Data Pipeline 는 onSuccess, OnFail및와 같은 Amazon SNS 알림과 연결할 수 있는 파이프라인 구성 요소에 이벤트 필드를 제공하여 알림을 쉽게 구성할 수 있도록 합니다onLateAction.

에 대한 설정 AWS Data Pipeline

AWS Data Pipeline 를 처음 사용하기 전에 다음 작업을 완료합니다.

작업

- 에 가입 AWS
- AWS Data Pipeline 및 파이프라인 리소스에 대한 IAM 역할 생성
- IAM 주체(사용자 및 그룹)가 필요한 작업을 수행하도록 허용
- 프로그래밍 방식 액세스 권한 부여

이러한 작업을 완료한 후 사용을 시작할 수 있습니다 AWS Data Pipeline. 기본 자습서는 시작하기 AWS Data Pipeline을 참조하십시오.

에 가입 AWS

Amazon Web Services(AWS)에 가입하면를 포함한 AWS의 모든 서비스에 AWS 계정이 자동으로 등록됩니다 AWS Data Pipeline. 사용자에게는 사용한 서비스에 대해서만 요금이 청구됩니다. AWS Data Pipeline 사용률에 대한 자세한 내용은 AWS Data Pipeline을 참조하세요.

에 가입 AWS 계정

이 없는 경우 다음 단계를 AWS 계정완료하여 생성합니다.

에 가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 엽니다.
2. 온라인 지시 사항을 따르세요.

   등록 절차 중 전화 또는 텍스트 메시지를 받고 전화 키패드로 확인 코드를 입력하는 과정이 있습니다.

   에 가입하면 AWS 계정AWS 계정 루트 사용자이 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스에 액세스할 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

AWS 는 가입 프로세스가 완료된 후 확인 이메일을 보냅니다. 언제든지 https://aws.amazon.com/으로 이동하고 내 계정을 선택하여 현재 계정 활동을 확인하고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

에 가입한 후 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 AWS 계정보호 AWS IAM Identity Center, AWS 계정 루트 사용자활성화 및 생성합니다.

보안 AWS 계정 루트 사용자

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 계정 소유자AWS Management Console로에 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면 AWS Sign-In 사용 설명서의 루트 사용자로 로그인을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

   지침은 IAM 사용 설명서의 AWS 계정 루트 사용자(콘솔)에 대한 가상 MFA 디바이스 활성화를 참조하세요.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 AWS IAM Identity Center설정을 참조하세요.
2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

   를 자격 증명 소스 IAM Identity Center 디렉터리 로 사용하는 방법에 대한 자습서는 사용 AWS IAM Identity Center 설명서의 기본값으로 사용자 액세스 구성을 IAM Identity Center 디렉터리 참조하세요.

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM Identity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자를 사용하여 로그인하는 데 도움이 필요하면 AWS Sign-In 사용 설명서의 AWS 액세스 포털에 로그인을 참조하세요.

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 Create a permission set를 참조하세요.
2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 그룹 추가를 참조하세요.

AWS Data Pipeline 및 파이프라인 리소스에 대한 IAM 역할 생성

AWS Data Pipeline 에는 작업을 수행하고 AWS 리소스에 액세스할 수 있는 권한을 결정하는 IAM 역할이 필요합니다. 파이프라인 역할은가 AWS Data Pipeline 보유한 권한을 결정하고, 리소스 역할은 EC2 인스턴스와 같은 파이프라인 리소스에서 실행되는 애플리케이션이 보유한 권한을 결정합니다. 파이프라인을 생성할 때 이 정보를 지정합니다. 사용자 지정 역할을 지정하지 않고 기본 역할 `DataPipelineDefaultRole` 및 `DataPipelineDefaultResourceRole`을(를) 사용하는 경우에도 먼저 역할을 생성하고 권한 정책을 연결해야 합니다. 자세한 내용은 에 대한 IAM 역할 AWS Data Pipeline 단원을 참조하십시오.

IAM 주체(사용자 및 그룹)가 필요한 작업을 수행하도록 허용

파이프라인을 사용하려면 계정의 IAM 보안 주체(사용자 또는 그룹)가 파이프라인에서 정의한 다른 서비스에 필요한 AWS Data Pipeline 작업과 작업을 수행할 수 있도록 허용해야 합니다.

권한을 단순화하기 위해 AWSDataPipeline_FullAccess 관리형 정책을 사용자가 IAM 보안 주체에 연결할 수 있습니다. 이 관리형 정책을 사용하면 보안 주체가 사용자에게 필요한 모든 작업과 사용자 지정 역할이 지정되지 않은 AWS Data Pipeline 경우에 사용되는 기본 역할에 대한 `iam:PassRole` 작업을 수행할 수 있습니다.

이 관리형 정책을 신중하게 평가하고 사용자에게 필요한 권한으로만 권한을 제한하는 것이 좋습니다. 필요한 경우, 이 정책을 출발점으로 사용하고 권한을 제거하여 IAM 보안 주체에 연결할 수 있는 보다 제한적인 인라인 권한 정책을 생성하십시오. 자세한 내용과 예제 정책은 에 대한 정책 예제 AWS Data Pipeline섹션을 참조하십시오.

파이프라인을 사용하는 모든 IAM 보안 주체에 연결된 정책에는 다음 예와 유사한 정책 설명이 포함되어야 합니다. 이 명령문을 사용하면 IAM 보안 주체가 파이프라인이 사용하는 역할에 대해 `PassRole`작업을 수행할 수 있습니다. 기본 역할을 사용하지 않는 경우, *`MyPipelineRole`* 및 *`MyResourceRole`*을(를) 직접 생성한 사용자 지정 역할로 바꾸십시오.

JSON

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

다음 절차는 IAM 그룹 생성 방법을 시연하고, AWSDataPipeline_FullAccess 관리형 정책을 그룹에 붙이고, 그 다음 사용자를 그룹에 추가하는 방법을 보여줍니다. 이 절차를 모든 인라인 정책에 사용할 수 있습니다.

사용자 그룹 **DataPipelineDevelopers**를 생성하고 AWSDataPipeline_FullAccess 정책을 연결하려면

1. IAM 콘솔(https://console.aws.amazon.com/iam/)을 엽니다.
2. 탐색 창에서 [Groups]를 선택하고 [Create New Group]을 선택합니다.
3. 그룹 이름을 입력하고(예: **DataPipelineDevelopers**) 다음 단계를 선택합니다.
4. **AWSDataPipeline_FullAccess**을(를) 필터에 입력한 다음 목록에서 그것을 선택합니다.
5. 다음 단계를 선택한 다음, 그룹 생성을 선택합니다.
6. 그룹에 사용자를 추가하려면:
    a. 그룹 목록에서 생성한 그룹을 선택합니다.
    b. 그룹 작업을 선택하고, 그룹에 사용자 추가를 선택합니다.
    c. 목록에서 추가하려는 사용자를 선택한 다음에 사용자를 그룹에 추가를 선택합니다.

프로그래밍 방식 액세스 권한 부여

사용자는 AWS 외부에서와 상호 작용하려는 경우 프로그래밍 방식으로 액세스해야 합니다 AWS Management Console. 프로그래밍 방식 액세스를 부여하는 방법은에 액세스하는 사용자 유형에 따라 다릅니다 AWS.

사용자에게 프로그래밍 방식 액세스 권한을 부여하려면 다음 옵션 중 하나를 선택합니다.

프로그래밍 방식 액세스가 필요한 사용자	목적	방법
IAM	(권장) 콘솔 자격 증명을 임시 자격 증명으로 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • 자세한 AWS CLI내용은 AWS Command Line Interface 사용 설명서의 AWS 로컬 개발을 위한 로그인을 참조하세요. • AWS SDKs 경우 SDK 및 도구 참조 안내서의 AWS 로컬 개발을 위한 로그인을 참조하세요. AWS SDKs
작업 인력 ID (IAM Identity Center에서 관리되는 사용자)	임시 자격 증명을 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • 자세한 AWS CLI내용은 AWS Command Line Interface 사용 설명서의 AWS CLI 를 사용하도록 구성을 AWS IAM Identity Center 참조하세요. • AWS SDKs, 도구 및 AWS APIs의 경우 SDK 및 도구 참조 안내서의 IAM Identity

프로그래밍 방식 액세스가 필요한 사용자	목적	방법
		Center 인증을 참조하세요. AWS SDKs
IAM	임시 자격 증명을 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	IAM 사용 설명서의 AWS 리소스에서 임시 자격 증명 사용의 지침을 따릅니다.
IAM	(권장되지 않음) 장기 자격 증명을 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • 자세한 AWS CLI내용은 AWS Command Line Interface 사용 설명서의 IAM 사용자 자격 증명을 사용하여 인증을 참조하세요. • AWS SDKs 및 도구의 경우 SDK 및 도구 참조 안내서의 장기 자격 증명을 사용하여 인증을 참조하세요. AWS SDKs • AWS APIs 경우 IAM 사용 설명서의 IAM 사용자의 액세스 키 관리를 참조하세요.

시작하기 AWS Data Pipeline

AWS Data Pipeline 를 사용하면 반복 데이터 처리 워크로드를 안정적이고 비용 효율적으로 시퀀싱, 예약, 실행 및 관리할 수 있습니다. 이 서비스는 비즈니스 논리에 기반하는 온프레미스와 클라우드 모두에서 정형 및 비정형 데이터를 사용하여 ETL(extract-transform-load: 추출-변환-로드) 활동을 쉽게 설계할 수 있게 해줍니다.

사용하려면 데이터 처리를 위한 비즈니스 로직을 지정하는 파이프라인 정의를 AWS Data Pipeline생성합니다. 일반적인 파이프라인 정의는 실행할 작업을 정의하는 활동, 입력 및 출력 데이터의 위치와 유형을 정의하는 데이터 노드, 활동 실행 시점을 정하는 일정으로 구성됩니다.

이 자습서에서는 Apache 웹 서버 로그에서 GET 요청 수를 계산하는 셸 명령 스크립트를 실행합니다. 이 파이프라인은 1시간 동안 15분마다 실행되며, 반복될 때마다 Amazon S3로 출력을 기록합니다.

사전 조건

시작하기 전에 에 대한 설정 AWS Data Pipeline의 작업을 완료해야 합니다.

파이프라인 객체

파이프라인은 다음 객체를 사용합니다.

ShellCommandActivity

입력 로그 파일을 읽고 오류 수를 계산합니다.

S3DataNode (입력)

입력 로그 파일이 있는 S3 버킷입니다.

S3DataNode (출력)

출력용 &S3 버킷입니다.

Ec2Resource

가 활동을 수행하는 데 AWS Data Pipeline 사용하는 컴퓨팅 리소스입니다.

대량의 로그 파일 데이터가 있는 경우에는 EC2 인스턴스 대신 EMR 클러스터를 사용하여 파일을 처리하도록 파이프라인을 구성할 수 있습니다.

일정

1시간 동안 15분마다 활동을 실행하는 것으로 정의합니다.

작업

- 파이프라인 생성
- 실행 중인 파이프라인 모니터링
- 출력 검토
- 파이프라인 삭제

파이프라인 생성

를 시작하는 가장 빠른 방법은 템플릿이라는 파이프라인 정의를 AWS Data Pipeline 사용하는 것입니다.

파이프라인을 생성하려면

1. https://console.aws.amazon.com/datapipeline/ AWS Data Pipeline 콘솔을 엽니다.
2. 탐색 모음에서 리전을 선택합니다. 현재 위치와 관계없이 사용자가 고를 수 있는 리전을 임의로 선택합니다. 많은 AWS 리소스는 리전에 고유하지만 파이프라인과 다른 리전에 있는 리소스를 사용할 수 AWS Data Pipeline 있습니다.
3. 처음 표시되는 화면은 현재 리전에서 파이프라인을 생성했는지 여부에 따라 달라집니다.

   a. 이 리전에서 파이프라인을 생성하지 않았다면 콘솔에 소개 화면이 표시됩니다. 지금 시작을 선택합니다.

   b. 이 리전에서 이미 파이프라인을 생성했다면 해당 리전의 파이프라인이 나열된 페이지가 콘솔에 표시됩니다. 새 파이프라인 생성을 선택합니다.
4. 이름에 파이프라인 이름을 입력합니다.
5. (선택 사항) 설명에 파이프라인에 대한 설명을 입력합니다.
6. 소스의 경우는 [Build using a template]을 선택한 후 다음 템플릿 [Getting Started using ShellCommandActivity]를 선택합니다.
7. 템플릿을 선택할 때 열린 [Parameters] 부분 아래의 [S3 input foler] 및 [Shell command to run]은 기본값을 유지합니다. [S3 output folder] 옆의 폴더 아이콘을 클릭하고, 버킷 또는 폴더 중 하나를 선택한 다음 [Select]를 클릭합니다.
8. [Schedule] 아래의 기본값을 그대로 둡니다. 파이프라인을 활성화하면 파이프라인 실행이 시작된 후 1시간 동안 15분마다 실행합니다.

   원할 경우 [Run once on pipeline activation]을 선택할 수도 있습니다.

9. Pipeline Configuration(파이프라인 구성)에서 로깅을 활성화된 상태로 두십시오. S3 location for logs(로그의 S3 위치) 아래의 폴더 아이콘을 선택하고 버킷이나 폴더 중 하나를 선택한 후 선택을 선택합니다.

   원하는 경우, 로깅을 대신 비활성화할 수 있습니다.
10. Security/Access(보안/액세스)에서 IAM 역할 설정을 기본값으로 유지합니다.
11. Activate를 클릭합니다.

    필요하면 Edit in Architect(아키텍트에서 편집)를 선택하여 이 파이프라인을 수정합니다. 예를 들어 사전 조건을 추가할 수 있습니다.

실행 중인 파이프라인 모니터링

파이프라인을 활성화하고 나면 파이프라인 진행률을 모니터링할 수 있는 실행 세부 정보 페이지로 이동하게 됩니다.

파이프라인 진행률을 모니터링하려면

1. [Update]를 클릭하거나 F5를 눌러 표시된 상태를 업데이트합니다.

   > **Tip**
   >
   > 열거된 실행이 없는 경우, [Start (in UTC)]와 [End (in UTC)]에 파이프라인의 예약된 시작 및 종료가 포함되는지 확인한 다음 [Update]를 클릭합니다.

2. 파이프라인에 있는 모든 객체의 상태가 `FINISHED`가 되면 파이프라인이 예약된 작업을 성공적으로 완료한 것입니다.
3. 파이프라인이 성공적으로 완료되지 않으면 파이프라인 설정에서 문제를 확인하십시오. 실패하거나 완료되지 않은 파이프라인 인스턴스 실행 문제 해결에 대한 자세한 내용은 공통 문제 해결 단원을 참조하세요.

출력 검토

Amazon S3 콘솔을 열고 버킷으로 이동합니다. 파이프라인을 한 시간 동안 15분마다 실행한 경우 타임스탬프가 지정된 하위 폴더 4개가 표시됩니다. 각 하위 폴더에는 이름이 `output.txt`인 파일의 출력이 포함되어 있습니다. 매번 동일한 입력 파일에서 스크립트를 실행했기 때문에 출력 파일이 동일합니다.

파이프라인 삭제

요금이 발생하는 것을 중지하려면 파이프라인을 삭제하십시오. 파이프라인을 삭제하면 파이프라인 정의 및 연결된 모든 객체가 삭제됩니다.

파이프라인을 삭제하려면

1. List Pipelines(파이프라인 나열) 페이지에서 파이프라인을 선택합니다.
2. 작업을 클릭한 후 삭제를 선택합니다.
3. 확인 메시지가 나타나면 삭제를 선택합니다.

이 자습서의 출력으로 끝난 경우에는 Amazon S3 버킷의 출력 폴더를 삭제하십시오.

파이프라인 작업

명령줄 인터페이스(CLI) 또는 AWS SDK를 사용하여 파이프라인을 관리, 생성 및 수정할 수 있습니다. 이후 단원들에서 기본 AWS Data Pipeline 개념을 소개하고 파이프라인 작업 방법을 설명합니다.

> **Important**
> 시작하기 전에 에 대한 설정 AWS Data Pipeline 단원을 참조하세요.

내용

파이프라인 생성

AWS Data Pipeline 는 파이프라인을 생성할 수 있는 몇 가지 방법을 제공합니다.

- 편의를 위해 제공된 템플릿과 함께 AWS Command Line Interface (CLI)를 사용합니다. 자세한 내용은 CLI를 사용하여 Data Pipeline 템플릿에서 파이프라인을 생성합니다. 단원을 참조하십시오.
- JSON 형식의 파이프라인 정의 파일과 함께 AWS Command Line Interface (CLI)를 사용합니다.
- 언어별 API가 있는 AWS SDK를 사용합니다. 자세한 내용은 API 작업 단원을 참조하십시오.

CLI를 사용하여 Data Pipeline 템플릿에서 파이프라인을 생성합니다.

데이터 파이프라인은 템플릿이라고 하는 여러 개의 사전 구성 파이프라인 정의를 제공합니다. 템플릿을 사용하여를 AWS Data Pipeline 빠르게 시작할 수 있습니다. 이러한 템플릿은 Amazon S3 위치(`s3://datapipeline-us-east-1/templates/`)의 공용 버킷에서 다음과 같이 사용할 수 있습니다. 이러한 사전 정의된 템플릿은 특정 사용 사례를 달성하기 위해 생성되며 파이프라인을 생성하는 데 사용할 수 있습니다. `aws s3 ls --recursive "s3://datapipeline-us-east-1/templates/"`을(를) 사용하여 사용 가능한 모든 템플릿을 나열할 수 있습니다.

CLI를 사용하여 템플릿에서 파이프라인 생성

DynamoDB 테이블을 Amazon S3로 내보내는 파이프라인을 생성하고 싶다고 가정합니다. 이 경우 사용할 템플릿은 `s3://datapipeline-us-east-1/templates/DynamoDB Templates/Export DynamoDB table to S3.json`에서 찾을 수 있습니다.

템플릿 JSON을 다운로드하고 CLI를 사용하여 파이프라인을 생성하려면

1. `aws s3 cp` CLI 또는 curl을 사용하여 템플릿을 다운로드합니다. 예제:

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

2. 필요에 따라 다운로드한 템플릿을 수정합니다. 예를 들어 최신 EMR 릴리스 버전을 사용하려면 `EmrClusterForBackup`객체의 `releaseLabel`필드를 변경하고, 마스터 및 코어 인스턴스 유형을 변경하고, 템플릿에서 파라미터의 기본값을 변경합니다.
3. `create-pipeline` CLI를 사용하여 파이프라인을 생성합니다. 예제:

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

4. 생성된 파이프라인 ID를 기록해 둡니다.
5. `put-pipeline-definition`을(를) 사용해서 정의를 업로드합니다. `--parameter-values` 옵션을 사용하여 기본값을 재정의하려는 파라미터 값을 입력합니다.

템플릿에 대한 자세한 내용은 템플릿 선택 단원을 참조하십시오.

템플릿 선택

`s3://datapipeline-us-east-1/templates/` 템플릿은 Amazon S3 버킷에서 다운로드하는 데 사용 가능합니다.

템플릿

- ShellCommandActivity 사용 시작하기
- AWS CLI 명령 실행
- S3로 DynamoDB 내보내기
- S3에서 DynamoDB 백업 데이터 가져오기
- Amazon EMR 클러스터에서 작업 실행
- Amazon RDS MySQL 테이블을 Amazon S3에 전체 복사
- Amazon RDS MySQL 테이블을 Amazon S3로 증분 복사
- S3 데이터를 Amazon RDS MySQL 테이블로 로드
- Amazon Redshift에 Amazon RDS MySQL 테이블 전체 복사
- Amazon RDS MySQL 테이블을 Amazon Redshift로 증분 복사
- Amazon S3에서 Amazon Redshift로 데이터 로드

ShellCommandActivity 사용 시작하기

[Getting Started using ShellCommandActivity] 템플릿은 셸 명령 스크립트를 실행하여 로그 파일의 GET 요청 수를 셉니다. 파이프라인의 예약 실행 시마다 타임스탬프가 찍힌 Amazon S3 위치에 출력이 기록됩니다.

템플릿은 다음 파이프라인 객체를 사용합니다.

- ShellCommandActivity
- S3InputNode
- S3OutputNode
- Ec2Resource

AWS CLI 명령 실행

이 템플릿은 예약된 간격으로 사용자 지정 AWS CLI 명령을 실행합니다.

S3로 DynamoDB 내보내기

DynamoDB 테이블을 S3로 내보내기 템플릿은 DynamoDB 테이블에서 데이터를 Amazon S3 버킷으로 내보내도록 Amazon EMR 클러스터 일정을 정합니다. 이 템플릿은 Amazon EMR 클러스터를 사용하는데, 그 크기는 DynamoDB 테이블에서 사용할 수 있는 처리량 값에 비례합니다. 테이블에서 IOP를 늘릴 수 있지만 이 경우 가져오기 및 내보내기 과정에서 비용이 추가로 발생할 수 있습니다. 전에는 내보내기에서 HiveActivity를 사용했지만 현재는 네이티브 MapReduce를 사용합니다.

템플릿은 다음 파이프라인 객체를 사용합니다.

- EmrActivity
- EmrCluster
- DynamoDBDataNode
- S3DataNode

S3에서 DynamoDB 백업 데이터 가져오기

S3에서 DynamoDB 백업 데이터 가져오기 템플릿은 Amazon S3에서 이전에 생성된 DynamoDB 백업을 DynamoDB 테이블로 로드하도록 Amazon EMR 클러스터 일정을 정합니다. DynamoDB 테이블의 기존 항목이 백업 데이터 항목으로 업데이트되고, 새 항목이 테이블에 추가됩니다. 이 템플릿은 Amazon EMR 클러스터를 사용하는데, 그 크기는 DynamoDB 테이블에서 사용할 수 있는 처리량 값에 비례합니다. 테이블에서 IOP를 늘릴 수 있지만 이 경우 가져오기 및 내보내기 과정에서 비용이 추가로 발생할 수 있습니다. 전에는 가져오기에서 HiveActivity를 사용했지만 현재는 네이티브 MapReduce를 사용합니다.

템플릿은 다음 파이프라인 객체를 사용합니다.

- EmrActivity
- EmrCluster
- DynamoDBDataNode
- S3DataNode
- S3PrefixNotEmpty

Amazon EMR 클러스터에서 작업 실행

Elastic MapReduce 클러스터에서 작업 실행 템플릿은 제공된 파라미터에 따라 Amazon EMR 클러스터를 시작하고 지정된 일정에 따라 실행 단계를 시작합니다. 작업이 완료되면 EMR 클러스터가 종료됩

니다. 선택사항인 부트스트랩 작업을 지정하여 클러스터에서 추가 소프트웨어를 설치하거나 애플리케이션 구성을 변경할 수 있습니다.

템플릿은 다음 파이프라인 객체를 사용합니다.

- EmrActivity
- EmrCluster

Amazon RDS MySQL 테이블을 Amazon S3에 전체 복사

Amazon RDS MySQL 테이블을 S3에 전체 복사 템플릿은 전체 Amazon RDS MySQL 테이블을 복사하여 그 출력을 Amazon S3 위치에 저장합니다. 출력은 지정된 Amazon S3 위치에 있는 타임스탬프가 지정된 하위 폴더에 CSV 파일로 저장됩니다.

템플릿은 다음 파이프라인 객체를 사용합니다.

- CopyActivity
- Ec2Resource
- SqlDataNode
- S3DataNode

Amazon RDS MySQL 테이블을 Amazon S3로 증분 복사

Amazon RDS MySQL 테이블을 S3로 증분 복사템플릿은 Amazon RDS MySQL 테이블의 데이터를 증분 복사하여 그 출력을 Amazon S3 위치에 저장합니다. Amazon RDS MySQL 테이블에 마지막 수정된 열이 있어야 합니다.

이 템플릿은 예약된 시작 시간부터 시작하여 예약 간격마다 발생한 테이블 변경 사항을 복사합니다. 일정 유형은 시계열이므로 복사가 특정 시간에 예약된 경우는 시간 내에 속하는 마지막으로 수정된 타임스탬프가 있는 테이블 행을 AWS Data Pipeline 복사합니다. 테이블에 대한 물리적 삭제는 복사되지 않습니다. 예약 실행을 할 때마다 Amazon S3 위치의 타임스탬프가 있는 하위 폴더에 결과가 기록됩니다.

템플릿은 다음 파이프라인 객체를 사용합니다.

- CopyActivity
- Ec2Resource

- SqlDataNode
- S3DataNode

S3 데이터를 Amazon RDS MySQL 테이블로 로드

S3 데이터를 RDS MySQL 테이블로 로드 템플릿은 Amazon EC2 인스턴스가 CSV 파일을 아래에 지정된 Amazon S3 파일 경로로부터 Amazon RDS MySQL 테이블로 복사하도록 예약합니다. CSV 파일에 헤더 행이 있으면 안 됩니다. 이 템플릿은 Amazon RDS MySQL 테이블의 기존 항목을 Amazon S3 데이터 항목으로 업데이트하고, 새 항목을 Amazon S3 데이터에서 Amazon RDS MySQL 테이블에 추가합니다. 데이터를 기존 테이블로 로드하거나 SQL 쿼리를 제공하여 새 테이블을 생성할 수 있습니다.

템플릿은 다음 파이프라인 객체를 사용합니다.

- CopyActivity
- Ec2Resource
- SqlDataNode
- S3DataNode

Amazon RDS에서 Amazon Redshift로의 템플릿

다음 두 템플릿은 변환 스크립트를 사용하여 Amazon RDS MySQL에서 Amazon Redshift로 테이블을 복사합니다. 이 스크립트는 다음 조항이 있는 소스 테이블 스키마를 사용하여 Amazon Redshift 테이블을 생성합니다.

- 배포 키가 지정되지 않은 경우에는 Amazon RDS 테이블의 첫 번째 프라이머리 키가 배포 키로 설정됩니다.
- Amazon Redshift로 복사할 때는 Amazon RDS MySQL 테이블에 있는 열을 건너뛰지 못합니다.
- (선택 사항) Amazon RDS MySQL을 Amazon Redshift 열 데이터 형식 매핑에게 템플릿의 파라미터 중 하나로 제공할 수도 있습니다. 이렇게 지정하는 경우, 스크립트는 이 매핑을 사용하여 Amazon Redshift 테이블을 생성합니다.

`Overwrite_Existing` Amazon Redshift 삽입 모드를 사용하는 경우:

- 배포 키가 제공되지 않은 경우에는 Amazon RDS MySQL 테이블의 프라이머리 키를 사용합니다.
- 배포 키가 제공되지 않았는데 테이블에 복합 기본 키가 있으면 첫 번째 키를 배포 키로 사용합니다. 첫 번째 복합 키만이 Amazon Redshift 테이블의 프라이머리 키로 설정됩니다.

- 배포 키가 제공되지 않고 Amazon RDS MySQL 테이블에 프라이머리 키가 없으면 복사 작업이 실패합니다.

Amazon Redshift에 대한 자세한 내용은 다음 항목을 참조하세요.

- Amazon Redshift 클러스터
- Amazon Redshift 복사
- 분산 스타일 및 DISTKEY 예제
- 정렬 키

아래 표에서 이 스크립트가 데이터 형식을 변환하는 방법을 설명합니다.

MySQL과 Amazon Redshift 사이의 데이터 형식 변환

MySQL 데이터 형식	Amazon Redshift 데이터 형식	참고
TINYINT, TINYINT (크기)	SMALLINT	MySQL: -128 ~ 127. 괄호 안에 최대 자릿수를 명시할 수 있습니다. Amazon Redshift: INT2. 2바이트 부호화 정수
TINYINT UNSIGNED, TINYINT (크기) UNSIGNED	SMALLINT	MySQL: 0 ~ 255 UNSIGNED. 괄호 안에 최대 자릿수를 명시할 수 있습니다. Amazon Redshift: INT2. 2바이트 부호화 정수
SMALLINT, SMALLINT(크기)	SMALLINT	MySQL: -32768 ~ 32767 일반. 괄호 안에 최대 자릿수를 명시할 수 있습니다. Amazon Redshift: INT2. 2바이트 부호화 정수

MySQL 데이터 형식	Amazon Redshift 데이터 형식	참고
SMALLINT UNSIGNED, SMALLINT(크기) UNSIGNED,	INTEGER	MySQL: 0 ~ 65535 UNSIGNED*. 괄호 안에 최대 자릿수를 명시할 수 있습니다. Amazon Redshift: INT4. 4바이트 부호화 정수
MEDIUMINT, MEDIUMINT(크기)	INTEGER	MySQL: 388608 ~ 8388607. 괄호 안에 최대 자릿수를 명시할 수 있습니다. Amazon Redshift: INT4. 4바이트 부호화 정수
MEDIUMINT UNSIGNED, MEDIUMINT(크기) UNSIGNED	INTEGER	MySQL: 0 ~ 16777215. 괄호 안에 최대 자릿수를 명시할 수 있습니다. Amazon Redshift: INT4. 4바이트 부호화 정수
INT, INT(크기)	INTEGER	MySQL: 147483648 ~ 2147483647 Amazon Redshift: INT4. 4바이트 부호화 정수
INT UNSIGNED, INT(크기) UNSIGNED	BIGINT	MySQL: 0 ~ 4294967295 Amazon Redshift: INT8. 8바이트 부호화 정수
BIGINT BIGINT(크기)	BIGINT	Amazon Redshift: INT8. 8바이트 부호화 정수

MySQL 데이터 형식	Amazon Redshift 데이터 형식	참고
BIGINT UNSIGNED BIGINT(크기) UNSIGNED	VARCHAR(20*4)	MySQL: 0 ~ 184467440 73709551615 Amazon Redshift: 네이티브에 상당하는 것이 없어 문자 배열을 사용합니다.
FLOAT FLOAT(크기,d) FLOAT(크기, d) UNSIGNED	REAL	최대 자릿수는 크기 파라미터에 지정될 수 있습니다. 소수점 오른쪽의 최대 자릿수는 d 파라미터에 지정됩니다. Amazon Redshift: FLOAT4
DOUBLE(크기, d)	DOUBLE PRECISION	최대 자릿수는 크기 파라미터에 지정될 수 있습니다. 소수점 오른쪽의 최대 자릿수는 d 파라미터에 지정됩니다. Amazon Redshift: FLOAT8
DECIMAL(크기, d)	DECIMAL(크기, d)	DOUBLE이 문자열로 저장되어, 고정 소수점에 허용됨. 최대 자릿수는 크기 파라미터에 지정될 수 있습니다. 소수점 오른쪽의 최대 자릿수는 d 파라미터에 지정됩니다. Amazon Redshift: 네이티브에 상당한 것이 없습니다.

MySQL 데이터 형식	Amazon Redshift 데이터 형식	참고
CHAR(크기)	VARCHAR(크기*4)	고정된 길이의 문자열을 유지합니다. 여기에는 글자, 숫자 및 특수 문자가 포함될 수 있습니다. 정해진 크기가 괄호 안에 파라미터로 명시됩니다. 최대 255자까지 저장할 수 있습니다. 오른쪽에 스페이스가 있습니다. Amazon Redshift: CHAR 데이터 형식은 멀티바이트 문자를 지원하지 않아 VARCHAR을 사용합니다. 문자 테이블을 U+10FFFF로 제한하는 RFC3629에 따라 문자당 최대 바이트 수가 4입니다.
VARCHAR(크기)	VARCHAR(크기*4)	최대 255자까지 저장할 수 있습니다. VARCHAR은 무효 UTF-8 코드 포인트, 즉 0xD800 - 0xDFFF, (바이트 시퀀스: ED A0 80 - ED BF BF), 0xFDD0 - 0xFDEF, 0xFFFE, 0xFFFF, (바이트 시퀀스: EF B7 90 - EF B7 AF, EF BF BE, EF BF BF)를 지원하지 않습니다.
TINYTEXT	VARCHAR(255*4)	최대 255자 길이의 문자열을 포함합니다.

MySQL 데이터 형식	Amazon Redshift 데이터 형식	참고
TEXT	VARCHAR(최대)	최대 65,535자 길이의 문자열을 포함합니다.
MEDIUMTEXT	VARCHAR(최대)	0 ~ 16,777,215 Char
LONGTEXT	VARCHAR(최대)	0 ~ 4,294,967,295 Char
BOOLEAN BOOL TINYINT(1)	BOOLEAN	MySQL: 이 유형은 TINYINT(1)의 동의어입니다. 제로 값은 false로 간주됩니다. 비-제로 값은 true로 간주됩니다.
BINARY[(M)]	varchar(255)	M은 0 ~ 255 바이트임, FIXED
VARBINARY(M)	VARCHAR(최대)	0 ~ 65,535바이트
TINYBLOB	VARCHAR(255)	0 ~ 255바이트
BLOB	VARCHAR(최대)	0 ~ 65,535바이트
MEDIUMBLOB	VARCHAR(최대)	0 ~ 16,777,215바이트
LONGBLOB	VARCHAR(최대)	0 ~ 4,294,967,295바이트
ENUM	VARCHAR(255*2)	문자 열거형 문자열의 길이가 아니라 테이블 정의의 열거형 값이 제한됩니다.
SET	VARCHAR(255*2)	열거형과 같음.
DATE	DATE	(YYYY-MM-DD) "1000-01-01" ~ "9999-12-31"
TIME	VARCHAR(10*4)	(hh:mm:ss) "-838:59:59" ~ "838:59:59"

MySQL 데이터 형식	Amazon Redshift 데이터 형식	참고
DATETIME	TIMESTAMP	(YYYY-MM-DD hh:mm:ss) 1000-01-01 00:00:00" ~"9999-12-31 23:59:59"
TIMESTAMP	TIMESTAMP	(YYYYMMDDhhmmss) 19700101000000 ~ 2037+
YEAR	VARCHAR(4*4)	(YYYY) 1900 ~ 2155
열 SERIAL	ID 생성 / 이 열은 복사되므로 OLAP 데이터 웨어하우스는 이 속성이 필요하지 않습니다. 변환 시에 SERIAL 키워드가 추가되지 않습니다.	사실 SERIAL은 이름이 SEQUENCE인 엔터티입니다. 이것은 나머지 테이블에 별도로 존재합니다. column GENERATED BY DEFAULT 다음과 동일: CREATE SEQUENCE name; CREATE TABLE table (column INTEGER NOT NULL DEFAULT nextval(name));

MySQL 데이터 형식	Amazon Redshift 데이터 형식	참고
column BIGINT UNSIGNED NOT NULL AUTO_INCR EMENT UNIQUE	ID 생성 / 이 열은 복사되므로 OLAP 데이터 웨어하우스는 이 속성이 필요하지 않습니다. 따라서 변환 시에 SERIAL 키워드가 추가되지 않습니다.	사실 SERIAL은 이름이 SEQUENCE인 엔터티입니다. 이것은 나머지 테이블에 별도로 존재합니다. column GENERATED BY DEFAULT 다음과 동일: CREATE SEQUENCE name; CREATE TABLE table (column INTEGER NOT NULL DEFAULT nextval(n ame));
ZEROFILL	변환 시에 ZEROFILL 키워드가 추가되지 않습니다.	INT UNSIGNED ZEROFILL NOT NULL ZEROFILL은 표시된 필드 값을 열 정의에 지정된 표시 너비까지 0으로 채웁니다. 표시 너비보다 긴 값은 잘리지 않습니다. ZEROFILL 사용은 UNSIGNED도 의미합니다.

Amazon Redshift에 Amazon RDS MySQL 테이블 전체 복사

Amazon RDS MySQL 테이블을 Amazon Redshift로 전체 복사 템플릿은 Amazon S3 폴더에서 데이터를 스테이징하여 Amazon RDS MySQL 테이블 전체를 Amazon Redshift 테이블로 복사합니다. Amazon S3 스테이징 폴더는 Amazon Redshift 클러스터와 동일한 리전에 있어야 합니다. 테이블이 아직 없는 경우, 소스 Amazon RDS MySQL 테이블과 동일한 스키마로 Amazon Redshift 테이블이 생성됩니다. 테이블을 생성하는 중에, 모든 Amazon RDS MySQL을 적용하고 싶은 Amazon Redshift 열 데이터 형식 재지정에 제공하세요.

템플릿은 다음 파이프라인 객체를 사용합니다.

- CopyActivity
- RedshiftCopyActivity
- S3DataNode
- SqlDataNode
- RedshiftDataNode
- RedshiftDatabase

Amazon RDS MySQL 테이블을 Amazon Redshift로 증분 복사

Amazon RDS MySQL 테이블을 Amazon Redshift로 증분 복사 템플릿은 Amazon S3 폴더에서 데이터를 스테이징하여 Amazon RDS MySQL 테이블의 데이터를 Amazon Redshift 테이블로 복사합니다.

Amazon S3 스테이징 폴더는 Amazon Redshift 클러스터와 동일한 리전에 있어야 합니다.

AWS Data Pipeline 는 번역 스크립트를 사용하여 소스 Amazon RDS MySQL 테이블과 스키마가 동일한 Amazon Redshift 테이블이 아직 없는 경우 해당 테이블을 생성합니다. Amazon Redshift 테이블을 생성하는 중에, 모든 Amazon RDS MySQL을 적용하고 싶은 Amazon Redshift 열 데이터 형식 재지정에 제공해야 합니다.

이 템플릿은 예약된 시작 시간부터 시작하여 예약한 간격 사이에 발생한 Amazon RDS MySQL 테이블 변경 사항을 복사합니다. Amazon RDS MySQL 테이블에 대한 물리적 삭제는 복사되지 않습니다. 마지막 수정 시간 값을 저장하는 열 이름을 입력해야 합니다.

기본 템플릿을 사용하여 증분 Amazon RDS 복사를 위해 파이프라인을 생성하는 경우, 기본 이름이 `RDSToS3CopyActivity`인 활동이 생성됩니다. 이 이름은 바꿀 수 있습니다.

템플릿은 다음 파이프라인 객체를 사용합니다.

- CopyActivity
- RedshiftCopyActivity
- S3DataNode
- SqlDataNode
- RedshiftDataNode
- RedshiftDatabase

Amazon S3에서 Amazon Redshift로 데이터 로드

S3에서 Redshift로 데이터 로드 템플릿은 Amazon S3 폴더의 데이터를 Amazon Redshift 테이블로 복사합니다. 데이터를 기존 테이블로 로드하거나 SQL 쿼리를 제공하여 테이블을 생성할 수 있습니다.

이 데이터는 Amazon Redshift COPY옵션에 따라 복사됩니다. Amazon Redshift 테이블에 Amazon S3의 데이터와 동일한 스키마가 있어야 합니다. COPY 옵션을 알아보려면, Amazon Redshift 데이터 개발자 안내서의 COPY를 참조하세요.

템플릿은 다음 파이프라인 객체를 사용합니다.

- CopyActivity
- RedshiftCopyActivity
- S3DataNode
- RedshiftDataNode
- RedshiftDatabase
- Ec2Resource

파라미터화된 템플릿을 사용하여 파이프라인 생성

파라미터화된 템플릿을 사용하여 파이프라인 정의를 사용자 지정할 수 있습니다. 따라서 공통 파이프라인 정의를 생성할 수 있지만 파이프라인 정의를 새 파이프라인에 추가할 때는 다른 파라미터를 제공해야 합니다.

내용

- 파이프라인 정의에 myVariables 추가
- 파라미터 객체 정의
- 파라미터 값 정의
- 파이프라인 정의 제출

파이프라인 정의에 myVariables 추가

파이프라인 정의 파일을 생성할 때 #{my*Variable*} 구문을 사용하여 변수를 지정하세요. 변수 앞에 my를 붙여야 합니다. 예를 들어, 파이프라인 정의 파일 pipeline-definition.json에는 변수 *myShellCmd*, *myS3InputLoc*, *myS3OutputLoc*이 포함됩니다.

> **Note**
>
> 파이프라인 정의의 파라미터 최대수는 50개입니다.

```
{
  "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"
    }
  ]
}
```

파라미터 객체 정의

파이프라인 정의에서 변수를 정의하는 파라미터 객체가 있는 별도의 파일을 생성할 수 있습니다. 예를 들어, JSON 파일 `parameters.json`에는 위의 예제 파이프라인 정의에 있는 *myShellCmd*, *myS3InputLoc*, *myS3OutputLoc* 변수의 파라미터 객체가 포함됩니다.

```
{
  "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**
>
> 별도의 파일을 사용하지 않고 이러한 객체를 파이프라인 정의 파일에 직접 추가할 수 있습니다.

다음 표는 파라미터 객체의 속성을 설명합니다.

파라미터 속성

속성	Type	설명
id	문자열	파라미터의 고유 식별자입니다. 입력 또는 표시될 때 값을 마스킹하려면 별표('*')를 접두사로 추가합니다. 예: *myVariable —. 이렇게 하면 AWS Data Pipeline이 저장하기 전에 값이 암호화됩니다.
description	문자열	파라미터 설명입니다.
type	String, Integer, Double 또는 AWS::S3::ObjectKey	입력 값과 유효성 검사 규칙의 허용 범위를 정의하는 파라미터 유형입니다. 기본값은 String입니다.
선택 사항	부울	파라미터가 선택인지 필수인지 여부를 나타냅니다. 기본값은 false입니다.

속성	Type	설명
allowedValues	문자열 목록	파라미터에 허용되는 모든 값을 열거합니다.
기본값	문자열	파라미터의 기본값입니다. 파라미터 값을 사용하여 이 파라미터의 값을 지정하면 이 값이 기본값을 다시 정의합니다.
isArray	부울	파라미터가 어레이인지를 나타냅니다.

파라미터 값 정의

파라미터 값을 사용하여 변수를 정의할 별도의 파일을 생성할 수 있습니다. 예를 들어, JSON 파일 `file://values.json`에는 위의 예제 파이프라인 정의에 있는 변수 *myS3OutputLoc* 값이 포함됩니다.

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

파이프라인 정의 제출

파이프라인 정의를 제출하면 파라미터, 파라미터 객체 및 파라미터 값을 지정할 수 있습니다. 예를 들어 다음과 같이 put-pipeline-definition AWS CLI 명령을 사용할 수 있습니다.

```
$ 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**
>
> 파이프라인 정의의 파라미터 최대수는 50개입니다. `parameter-values-uri`의 파일 크기 상한은 15kB입니다.

파이프라인 보기

명령줄 인터페이스(CLI)를 사용하여 파이프라인을 볼 수 있습니다.

를 사용하여 파이프라인을 보려면 AWS CLI

- 다음 list-pipelines 명령을 사용하여 파이프라인을 나열합니다.

```
aws datapipeline list-pipelines
```

파이프라인 상태 코드 해석

AWS Data Pipeline 콘솔 및 CLI에 표시되는 상태 수준은 파이프라인 및 해당 구성 요소의 상태를 나타냅니다. 파이프라인 상태는 파이프라인 개요입니다. 자세한 정보를 확인하려면 개별 파이프라인 구성 요소의 상태를 봐야 합니다.

파이프라인이 준비 상태이거나(파이프라인 정의가 유효성 검사 통과), 현재 작업을 진행 중이거나, 작업을 완료한 경우에는 파이프라인이 `SCHEDULED` 상태입니다. 파이프라인이 활성화되지 않았거나 작업을 수행할 수 없는 경우(예: 파이프라인 정의가 유효성 검사 실패) 파이프라인이 `PENDING` 상태입니다.

파이프라인 상태가 `PENDING`, `INACTIVE` 또는 `FINISHED`이면 비활성 상태로 간주됩니다. 비활성 파이프라인은 요금이 발생합니다(자세한 내용은 요금 참조).

상태 코드

`ACTIVATING`

EC2 인스턴스와 같은 구성 요소나 리소스가 시작 중입니다.

`CANCELED`

사용자가 실행하기 AWS Data Pipeline 전에 구성 요소를 취소했습니다. 이것은 이 구성 요소가 의존하는 다른 구성 요소나 리소스에서 오류가 발생할 경우, 자동으로 일어날 수 있습니다.

CASCADE_FAILED

이 구성 요소나 리소스는 하나 이상의 종속성에서 비롯된 캐스케이드 오류로 인해 취소되었지만 해당 구성 요소가 오류의 근원이 아닐 수 있습니다.

DEACTIVATING

파이프라인이 비활성화되는 중입니다.

FAILED

구성 요소 또는 리소스에 오류가 발생했고 작동이 멈췄습니다. 구성 요소 또는 리소스에 오류가 발생하면 이 구성 요소나 리소스에 의존하는 다른 구성 요소로 취소와 오류가 캐스케이딩될 수 있습니다.

FINISHED

구성 요소가 할당된 작업을 완료했습니다.

INACTIVE

파이프라인이 비활성화되었습니다.

PAUSED

구성 요소가 일시 중지되었고 현재 작업을 수행하고 있지 않습니다.

PENDING

파이프라인이 처음 활성화될 준비가 되었습니다.

RUNNING

리소스가 실행 중이고 작업을 수신할 준비가 되었습니다.

SCHEDULED

이 리소스는 실행되도록 예약되어 있습니다.

SHUTTING_DOWN

리소스가 작업을 성공적으로 완료한 후 종료 중입니다.

SKIPPED

구성 요소가 현재 일정 이후의 타임스탬프를 사용하여 파이프라인이 활성화된 후 실행 간격을 건너뛰었습니다.

TIMEDOUT

리소스가 terminateAfter 임계값을 초과하여 중지되었습니다 AWS Data Pipeline. 리소스가 이 상태에 도달하면 AWS Data Pipeline 이 리소스의 actionOnResourceFailure, retryDelay 및 retryTimeout 값을 무시합니다. 이 상태는 리소스에만 적용됩니다.

VALIDATING

파이프라인 정의를 검증하는 중입니다 AWS Data Pipeline.

WAITING_FOR_RUNNER

구성 요소가 작업자 클라이언트가 작업 항목을 검색하기를 기다리고 있습니다. 구성 요소와 작업자 클라이언트 관계는 해당 구성 요소가 정의한 runsOn 또는 workerGroup 필드에 의해 제어됩니다.

WAITING_ON_DEPENDENCIES

구성 요소가 작업 수행 전에 기본 및 사용자 구성 사전 조건이 충족되었는지 확인하는 중입니다.

파이프라인 및 구성요소 상태 해석

각 파이프라인과 해당 파이프라인 안의 구성요소에서 HEALTHY, ERROR, "-", No Completed Executions 또는 No Health Information Available 상태가 반환됩니다. 파이프라인 구성요소가 1차 실행을 완료한 이후 또는 구성요소 사전 조건이 충족되지 않은 경우에만 파이프라인이 상태를 갖게 됩니다. 구성요소의 상태가 파이프라인 상태로 집계되며, 파이프라인 실행 세부 정보를 볼 때 오류 상태가 제일 먼저 표시됩니다.

파이프라인 상태

HEALTHY

모든 구성요소의 집계 상태가 HEALTHY입니다. 이것은 하나 이상의 구성요소가 성공적으로 완료되었음을 의미합니다. HEALTHY 상태를 클릭하면 가장 최근에 성공적으로 완료된 파이프라인 구성요소 인스턴스를 실행 세부 정보 페이지에서 볼 수 있습니다.

ERROR

파이프라인에서 구성요소 하나 이상의 상태가 ERROR입니다. ERROR 상태를 클릭하면 실행 세부 정보 페이지에서 가장 최근에 실패한 파이프라인 구성요소 인스턴스를 볼 수 있습니다.

No Completed Executions 또는 No Health Information Available

이 파이프라인에 보고된 상태가 없습니다.

> **Note**
>
> 구성요소들이 그 상태를 바로 업데이트하면 파이프라인 상태가 업데이트되는 데 최대 5분까지 걸릴 수 있습니다.

구성요소 상태

HEALTHY

성공을 실행적으로 완료하여 FINISHED 또는 MARK_FINISHED 상태가 표시된 구성요소(Activity 또는 DataNode)는 상태가 HEALTHY입니다. 구성요소 이름 또는 HEALTHY 상태를 클릭하면 실행 세부 정보 페이지에서 가장 최근에 성공적으로 완료된 파이프라인 구성요소 인스턴스를 볼 수 있습니다.

ERROR

구성요소 레벨에서 오류가 발생했거나 그 사전 조건 하나가 충족되지 않았습니다. FAILED, TIMEOUT 또는 CANCELED로 이 오류가 발생합니다. 구성요소 이름 또는 ERROR 상태를 클릭하면 실행 세부 정보 페이지에서 가장 최근에 실패한 파이프라인 구성요소 인스턴스를 볼 수 있습니다.

No Completed Executions 또는 No Health Information Available

이 구성요소에 보고된 상태가 없습니다.

파이프라인 정의를 보려면

명령줄 인터페이스(CLI)를 사용하여 파이프라인 정의를 봅니다. CLI는 파이프라인 정의 파일을 JSON 형식으로 프린트합니다. 파이프라인 정의 파일의 구문과 사용에 대한 자세한 내용은 파이프라인 정의 파일 구문 단원을 참조하세요.

CLI를 사용할 경우에는 수정을 제출하기 전에 파이프라인 정의를 검색하는 것이 좋습니다. 왜냐하면 사용자가 작업을 마친 후에 다른 사용자 또는 프로세스에 의해 파이프라인 정의가 변경되었을 가능성이 있기 때문입니다. 현재 정의의 사본을 다운로드하고 이것을 토대로 수정하면 가장 최근의 파이프라인 정의로 작업할 수 있습니다. 수정 후에 파이프라인 정의를 다시 검색하여 업데이트에 성공했는지를 확인하는 것도 좋습니다.

CLI를 사용할 경우 두 가지 각기 다른 버전의 파이프라인을 가져올 수 있습니다. active 버전은 현재 실행 중인 파이프라인입니다. latest 버전은 실행 중인 파이프라인을 편집할 때 생성되는 사본입니

다. 편집된 파이프라인을 업로드하면 active 버전이 되고 이전 active 버전은 사용할 수 없게 됩니다.

를 사용하여 파이프라인 정의를 가져오려면 AWS CLI

전체 파이프라인 정의를 가져오려면 get-pipeline-definition 명령을 사용하세요. 파이프라인 정의는 표준 출력(stdout)으로 인쇄됩니다.

다음 예제에서는 지정된 파이프라인의 파이프라인 정의를 가져옵니다.

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

특정 파이프라인 버전을 검색하려면 --version 옵션을 사용합니다. 다음 예제에서는 지정된 파이프라인의 active 버전을 검색합니다.

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

파이프라인 인스턴스 세부 정보 보기

파이프라인 진행 상태를 모니터링할 수 있습니다. 인스턴스 상태에 대한 자세한 내용은 파이프라인 상태 세부 정보 해석 단원을 참조하세요. 실패하거나 완료되지 않은 파이프라인 인스턴스 실행 문제 해결에 대한 자세한 내용은 공통 문제 해결 단원을 참조하세요.

를 사용하여 파이프라인의 진행 상황을 모니터링하려면 AWS CLI

파이프라인이 실행한 횟수 내역 같은 파이프라인 인스턴스 세부 정보를 검색하려면 list-runs 명령을 사용합니다. 이 명령을 사용하여 현재 상태 또는 시작 날짜-범위에 따라 반환된 실행 목록을 필터링할 수 있습니다. 파이프라인 기간과 일정에 따라 실행 내역이 클 수 있으므로 결과를 필터링하는 것이 좋습니다.

다음 예제에서는 모든 실행의 정보를 검색합니다.

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

다음 예제에서는 완료된 모든 실행의 정보를 검색합니다.

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

다음 예제에서는 지정된 시간 프레임에서 시작된 모든 실행의 정보를 검색합니다.

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

파이프라인 로그 보기

파이프라인을 생성할 때 콘솔에서 Amazon S3 위치를 지정하거나 SDK/CLI의 기본 객체에 있는 pipelineLogUri을(를) 사용하여 파이프라인 수준의 로깅을 지원합니다. 해당 URI 내 각 파이프라인의 디렉터리 구조는 다음과 같습니다.

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

파이프라인 df-00123456ABC7DEF8HIJK의 디렉토리 구조는 다음과 같습니다.

```
df-00123456ABC7DEF8HIJK
    -ActivityId_fXNzc
        -@ActivityId_fXNzc_2014-05-01T00:00:00
            -@ActivityId_fXNzc_2014-05-01T00:00:00_Attempt=1
```

ShellCommandActivity는 이러한 활동과 연결된 stderr 및 stdout의 로그가 각 시도의 디렉터리에 저장됩니다.

emrLogUri가 설정된 EmrCluster 같은 리소스는 이 값이 우선합니다. 그 외에는 리소스(이러한 리소스의 TaskRunner 로그 포함)가 위의 파이프라인 로깅 구조를 따릅니다.

주어진 파이프라인 실행에 대한 로그를 보려면

1. query-objects을(를) 호출하여 ObjectId을(를) 검색하여 정확한 객체 ID를 가져옵니다. 예제:

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

   query-objects은(는) 페이지가 매겨진 CLI이고 주어진 pipeline-id 항목에 대해 더 많은 실행이 있는 경우 페이지 매김 토큰을 반환할 수 있습니다. 토큰을 사용하여 원하는 객체를 찾을 때까지 모든 시도를 진행할 수 있습니다. 예를 들어, 반환된 ObjectId는 다음과 같습니다. @TableBackupActivity_2023-05-020T18:05:18_Attempt=1
2. Objectid를 사용하기, 다음을 사용하여 로그 위치를 검색합니다.

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

실패한 활동의 오류 메시지

오류 메시지를 받으려면 먼저 `query-objects`을(를) 사용하는 ObjectID를 가져오십시오.

실패한 ObjectID를 검색한 후 `describe-objects`CLI를 사용하여 실제 오류 메시지를 가져옵니다.

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

객체를 취소하거나 재실행하거나 완료된 것으로 표시합니다.

`set-status` CLI를 사용하여 실행 중인 객체를 취소하거나, 장애가 발생한 객체를 재실행하거나, 실행 중인 객체를 Finished로 표시합니다.

먼저 `query-objects`CLI를 사용하여 객체 ID를 가져옵니다. 예제:

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

`set-status` CLI를 사용하여 원하는 객체의 상태를 변경합니다. 예제:

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

파이프라인 편집

파이프라인 하나의 일부 측면을 변경하려면 그 파이프라인 정의를 업데이트하면 됩니다. 실행 중인 파이프라인을 변경한 후에는 파이프라인을 다시 활성화해야 변경이 적용됩니다. 그리고 하나 이상의 파이프라인 구성요소를 다시 실행할 수 있습니다.

내용

제한 사항

파이프라인이 PENDING상태이고 활성화되지 않은 동안에는 파이프라인을 변경할 수 없습니다. 파이프라인을 활성화한 후에 파이프라인을 편집할 수 있으며 이때 다음과 같은 제한이 있습니다. 변경을 저장하고 파이프라인을 다시 실행한 후에 파이프라인 객체를 새로 실행할 때 변경이 적용됩니다.

- 객체는 제거할 수 없습니다.
- 기존 객체의 예약 기간을 변경할 수 없습니다.
- 기존 객체의 참조 필드를 추가, 삭제 또는 수정할 수 없습니다.
- 새 객체의 출력 필드에서 기존 객체를 참조할 수 없습니다.
- 객체의 예약된 시작 날짜를 변경할 수 없습니다(그 대신, 특정 날짜와 시간의 파이프라인을 활성화합니다).

를 사용하여 파이프라인 편집 AWS CLI

명령줄 도구를 사용하여 파이프라인을 편집할 수 있습니다.

먼저 get-pipeline-definition 명령을 사용하여 현재 파이프라인 정의의 사본을 다운로드합니다. 이렇게 하면 가장 최근 파이프라인 정의를 수정할 수 있습니다. 다음 예제에서는 파이프라인 정의를 표준 출력(stdout)으로 인쇄합니다.

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

파이프라인 정의를 파일로 저장하고 필요에 따라 편집합니다. put-pipeline-definition 명령을 사용하여 파이프라인 정의를 업데이트합니다. 다음 예제에서는 업데이트된 파이프라인 정의 파일을 업로드합니다.

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

get-pipeline-definition 명령을 사용하여 파이프라인 정의를 다시 검색하고 업데이트가 성공했는지 확인할 수 있습니다. 파이프라인을 활성화하려면 다음 activate-pipeline 명령을 사용하세요.

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

필요할 경우 다음과 같이 --start-timestamp 옵션을 사용하여 특정 날짜와 시간의 파이프라인을 활성화할 수 있습니다.

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

하나 이상의 파이프라인 구성요소를 다시 실행하려면 set-status 명령을 사용합니다.

파이프라인 복제

복제하면 파이프라인 사본이 생겨 이것을 사용하여 새 파이프라인의 이름을 지정할 수 있습니다. 오류가 발생해도 임의 상태의 파이프라인을 복제할 수 있으나 새 파이프라인은 수동으로 활성화하기 전까지 PENDING 상태를 유지합니다. 새 파이프라인은 복제 작업에서 활성 버전이 아닌 원래 파이프라인 정의의 최신 버전이 사용됩니다. 복제 작업에서 원래 파이프라인의 전체 일정이 새 파이프라인에 복사되지 않고 기간 설정만 복사됩니다.

AWS CLI를 사용하여 파이프라인을 복제하려면:

1. 새 이름과 고유 ID를 가진 새 파이프라인을 생성합니다. 반환된 파이프라인 ID를 기록해 둡니다.
2. get-pipeline-definition CLI를 사용하여 복제할 기존 파이프라인의 파이프라인 정의를 가져와서 복제하거나 임시 파일에 쓸 수 있습니다. 파일 경로를 지정합니다.
3. put-pipeline-definition CLI를 사용하여 기존 파이프라인에서 새 파이프라인으로 파이프라인 정의를 복사합니다.
4. get-pipeline-definition CLI를 사용하여 새 파이프라인의 정의를 가져와 파이프라인 정의를 확인합니다.

```
# 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
```

파이프라인 태그 지정

태그는 사용자가 정의하는 키와 선택 값으로 이루어진 대소문자를 구분하는 키-값 쌍입니다. 각 파이프라인에 최대 10개의 태그를 적용할 수 있습니다. 태그 키는 각 파이프라인마다 고유해야 합니다. 파이프라인에 이미 연결된 키를 통해 태그를 추가하면 해당 태그의 값이 업데이트됩니다.

또한 파이프라인에 태그를 적용하면 그 기본 리소스(예: Amazon EMR 클러스터와 Amazon EC2 인스턴스)에도 태그가 전파됩니다. 그러나 이러한 태그가 `FINISHED` 또는 종료 상태의 리소스에는 적용되지 않습니다. 필요할 경우 CLI를 사용하여 이러한 리소스에 태그를 적용할 수 있습니다.

태그 사용을 마치면 파이프라인에서 이를 제거할 수 있습니다.

AWS CLI를 사용하여 파이프라인에 태그를 지정하려면

새 파이프라인에 태그를 추가하려면 create-pipeline 명령에 `--tags` 옵션을 추가하세요. 예를 들어, 다음 옵션은 `production` 값이 있는 `environment` 태그와 `sales` 값이 있는 `owner` 태그가 있는 파이프라인을 생성합니다.

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

기존 파이프라인에 태그를 추가하려면 다음과 같이 add-tags 명령을 사용합니다.

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

기존 파이프라인에서 태그를 제거하려면 다음과 같이 remove-tags 명령을 사용합니다.

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

파이프라인 비활성화

실행 중인 파이프라인을 비활성화하면 파이프라인 실행이 일시 중지됩니다. 파이프라인 실행을 재개하려면 파이프라인을 활성화하면 됩니다. 그 후에 변경할 수 있습니다. 예를 들어, 유지관리가 예약된 데이터베이스로 데이터를 기록할 경우 파이프라인을 비활성화하고, 유지관리가 완료될 때까지 기다렸다가, 파이프라인을 활성화할 수 있습니다.

파이프라인을 비활성화할 때 실행 중인 활동에 관한 사항을 지정할 수 있습니다. 기본적으로 이러한 활동은 바로 취소됩니다. 또는 파이프라인 비활성화 전에 활동이 완료될 때까지 AWS Data Pipeline 을 대기시킬 수 있습니다.

비활성화된 파이프라인을 활성화할 때 재개 시점을 지정할 수 있습니다. AWS CLI 또는 API를 사용하면 파이프라인이 기본적으로 마지막으로 완료된 실행부터 재개되거나 파이프라인을 재개할 날짜와 시간을 지정할 수 있습니다.

를 사용하여 파이프라인 비활성화 AWS CLI

다음 deactivate-pipeline 명령을 사용하여 파이프라인을 비활성화합니다.

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

실행 중인 모든 활동이 끝난 후에만 파이프라인을 비활성화하려면 다음과 같이 `--no-cancel-active` 옵션을 추가합니다.

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

준비가 완료되면 다음 activate-pipeline 명령을 사용하여 비활성 상태였던 파이프라인 실행을 재개할 수 있습니다.

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

특정 날짜와 시간부터 파이프라인을 시작하려면 다음과 같이 `--start-timestamp` 옵션을 추가합니다.

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

파이프라인 삭제

애플리케이션 테스트 때 생성한 파이프라인처럼 더 이상 필요 없는 파이프라인은 삭제하여 활성 사용에서 제거해야 합니다. 파이프라인을 삭제하면 삭제 상태가 됩니다. 파이프라인이 삭제 상태가 되면 그 파이프라인 정의와 실행 내역이 사라집니다. 따라서 파이프라인 설명을 포함해 파이프라인 작업을 수행하지 못합니다.

> **Important**
>
> 삭제 후에는 파이프라인을 복원할 수 없으므로 앞으로 파이프라인이 필요하지 않은지 확인한 후에 삭제하세요.

를 사용하여 파이프라인을 삭제하려면 AWS CLI

파이프라인을 삭제하려면 delete-pipeline 명령을 사용하세요. 다음 명령은 지정된 파이프라인을 삭제합니다.

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

파이프라인 활동으로 데이터 및 테이블 준비

AWS Data Pipeline 는 파이프라인의 입력 및 출력 데이터를 스테이징하여 ShellCommandActivity 및와 같은 특정 활동을 더 쉽게 사용할 수 있습니다HiveActivity.

데이터 스테이징을 사용하면 입력 데이터 노드의 데이터를 활동 실행 중인 리소스로 복사할 수 있으며, 마찬가지로 리소스에서 출력 데이터 노드로도 복사할 수 있습니다.

Amazon EMR 또는 Amazon EC2 리소스에서 스테이징된 데이터는 활동의 쉘 명령 또는 Hive 스크립트의 특수 변수를 이용해 사용할 수 있습니다.

테이블 스테이징도 데이터 스테이징과 유사하지만, 스테이징된 데이터가 데이터베이스 테이블의 양식을 띤다는 점이 다릅니다.

AWS Data Pipeline 는 다음 스테이징 시나리오를 지원합니다.

- ShellCommandActivity로 데이터 스테이징
- Hive 및 스테이징 지원 데이터 노드로 테이블 스테이징
- Hive 및 스테이징 비-지원 데이터 노드로 테이블 스테이징

> **Note**
>
> 스테이징은 stage 필드가 ShellCommandActivity 등의 활동에서 true로 설정될 때만 작동합니다. 자세한 내용은 ShellCommandActivity 단원을 참조하십시오.

그리고 데이터 노드와 활동의 관계를 다음 네 방식으로 설정할 수 있습니다.

리소스에서 로컬로 데이터 스테이징

입력 데이터가 자동으로 리소스 로컬 파일 시스템에 복사됩니다. 출력 데이터가 자동으로 리소스 로컬 파일 시스템에서 출력 데이터 노드로 복사됩니다. 예를 들어, ShellCommandActivity 입력

및 출력을 스테이징 = true로 구성하면 입력 데이터는 INPUTx_STAGING_DIR로 사용할 수 있고, 출력 데이터는 OUTPUTx_STAGING_DIR로 사용할 수 있습니다. 여기서 x는 입력 또는 출력 번호입니다.

활동의 입력 및 출력 정의 스테이징

입력 데이터 형식(열 이름과 테이블 이름)이 자동으로 활동의 리소스에 복사됩니다. 예를 들어, `HiveActivity`를 스테이징 = true로 구성할 때가 이에 해당합니다. 입력 `S3DataNode`에 지정된 데이터 형식을 사용하여 Hive 테이블의 테이블 정의를 스테이징합니다.

활성화되지 않은 스테이징

입력 및 출력 객체와 그 필드를 활동에 사용할 수 있지만 데이터는 사용하지 못합니다. 예를 들어, `EmrActivity`(기본값) 또는 기타 활동을 스테이징 = false로 구성할 때가 이에 해당합니다. 이 구성에서 데이터 필드는 활동에서 AWS Data Pipeline 표현식 구문을 사용하여 참조할 수 있으며 이는 종속성이 충족될 때만 발생합니다. 이것은 종속성 확인 기능만 합니다. 활동의 코드는 입력값의 데이터를 해당 활동을 실행하는 리소스로 복사하는 역할을 담당합니다.

객체 사이의 종속성 관계

객체 사이에 종속 관계가 있는데, 이것 때문에 스테이징이 활성화되지 않을 때와 유사한 상황이 발생합니다. 이로 인해 데이터 노드나 활동이 다른 활동을 실행하는 사전 조건 역할을 하게 됩니다.

ShellCommandActivity로 데이터 스테이징

`S3DataNode` 객체`ShellCommandActivity`를 데이터 입력 및 출력으로 사용하는 시나리오를 생각해 보세요. 다음 예제`${INPUT1_STAGING_DIR}`와 `${OUTPUT1_STAGING_DIR}` 같이 데이터 노드를 AWS Data Pipeline 자동으로 스테이징하여 환경 변수를 사용하는 로컬 파일 폴더인 것처럼 셸 명령에 액세스할 수 있도록 합니다. 이름이 `INPUT1_STAGING_DIR` 및 `OUTPUT1_STAGING_DIR`인 변수의 숫자 부분은 활동이 참조하는 데이터 노드의 수에 따라 증분합니다.

> **Note**
>
> 이 시나리오는 데이터 입력과 출력이 `S3DataNode` 객체일 경우에만 설명대로 작동합니다. 그리고 출력 데이터 스테이징은 `directoryPath`가 출력 `S3DataNode` 객체에 설정된 경우에만 허용됩니다.

```
{
```

```
  "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')}"
  }
},
...
```

Hive 및 스테이징 지원 데이터 노드로 테이블 스테이징

S3DataNode 객체를 데이터 입력 및 출력으로 HiveActivity 사용하여를 사용하는 시나리오를 생각해 보세요.에 대한 다음 예제${input1}와 ${output1} 같이 Hive 테이블인 것처럼 Hive 스크립트에 액세스할 수 있도록 데이터 노드를 AWS Data Pipeline 자동으로 스테이징합니다HiveActivity. 이름이 input 및 output인 변수의 숫자 부분은 활동이 참조하는 데이터 노드의 수에 따라 증분합니다.

> **Note**
>
> 이 시나리오는 데이터 입력 및 출력이 S3DataNode 또는 MySqlDataNode 객체인 경우에만 설명대로 작동합니다. DynamoDBDataNode는 테이블 스테이징이 지원되지 않습니다.

```
{
  "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"
  }
},
...
```

Hive 및 스테이징-비지원 데이터 노드로 테이블 스테이징

DynamoDBDataNode가 데이터 입력이고 S3DataNode 객체가 출력인 HiveActivity를 사용하는 시나리오를 생각해봅니다. DynamoDBDataNode에 대해 데이터 스테이징을 사용할 수 없으므로, 먼저 Hive 스크립트 안에 수동으로 테이블을 생성하고, 이름이 #{input.tableName}인 변수를 사용하여 DynamoDB 테이블을 참조해야 합니다. DynamoDB 테이블이 출력일 경우에도 비슷한 명칭이 적용됩니다. 단, 변수 #{output.tableName}을(를) 사용하는 것이 다릅니다. 이 예제에서는 출력 S3DataNode 객체에 스테이징을 사용할 수 있어 ${output1} 같은 출력 데이터 노드를 참조할 수 있습니다.

> **Note**
>
> 이 예제에서는가 표현식을 AWS Data Pipeline 사용하여 tableName 또는에 액세스하기 때문에 테이블 이름 변수에 #(해시) 문자 접두사가 있습니다directoryPath. 표현식 평가의 작동 방식에 대한 자세한 내용은 섹션을 AWS Data Pipeline참조하세요표현식 평가.

```
{
  "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"
  }
},
...
```

여러 리전의 리소스와 파이프라인 사용

기본적으로 Ec2Resource 및 EmrCluster 리소스는와 동일한 리전에서 실행되지만 AWS Data Pipeline는 다른 리전의 입력 데이터를 통합하는 한 리전에서 리소스를 실행하는 등 여러 리전에서 데이터 흐름을 오케스트레이션하는 기능을 AWS Data Pipeline 지원합니다. 리소스가 지정 리전에서 실행되게 하면 리소스를 그 종속 데이터 세트와 같은 위치에 배치할 수 있으며, 지연 시간을 줄이고 리전 간 데이터 전송 요금의 발생을 방지하여 성능을 극대화할 수 있습니다. Ec2Resource 및의 region 필드를 AWS Data Pipeline 사용하는 것과 다른 리전에서 실행되도록 리소스를 구성할 수 있습니다EmrCluster.

다음 예제 파이프라인 JSON 파일은 유럽(아일랜드) 리전에서 EmrCluster리소스를 실행하는 방법을 보여줍니다. 단, 작업할 클러스터의 대용량 데이터가 동일 리전에 있다는 것을 전제로 합니다. 이 예제에서 일반 파이프라인과 유일하게 다른 점은 EmrCluster의 region 필드 값이 eu-west-1로 설정되어 있다는 것입니다.

```
{
  "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"
    }
  ]
}
```

다음 표에는 선택할 수 있는 리전과 `region` 필드에서 사용하는 연결된 리전 코드가 나열되어 있습니다.

> **Note**
>
> 다음 목록에는가 워크플로를 오케스트레이션하고 Amazon EMR 또는 Amazon EC2 리소스를 시작할 AWS Data Pipeline 수 있는 리전이 포함되어 AWS Data Pipeline 있습니다.는 이러한 리전에서 지원되지 않을 수 있습니다. AWS Data Pipeline 가 지원되는 리전에 대한 자세한 내용은 AWS 리전 및 엔드포인트를 참조하세요.

리전 이름	리전 코드
미국 동부(버지니아 북부)	us-east-1

리전 이름	리전 코드
미국 동부(오하이오)	us-east-2
미국 서부(캘리포니아 북부)	us-west-1
미국 서부(오리건)	us-west-2
캐나다(중부)	ca-central-1
유럽(아일랜드)	eu-west-1
유럽(런던)	eu-west-2
유럽(프랑크푸르트)	eu-central-1
아시아 태평양(싱가포르)	ap-southeast-1
아시아 태평양(시드니)	ap-southeast-2
아시아 태평양(뭄바이)	ap-south-1
아시아 태평양(도쿄)	ap-northeast-1
아시아 태평양(서울)	ap-northeast-2
남아메리카(상파울루)	sa-east-1

캐스케이드 실패 및 재실행

AWS Data Pipeline 를 사용하면 종속성이 실패하거나 사용자가 취소할 때 파이프라인 객체가 작동하는 방식을 구성할 수 있습니다. 실패가 다른 파이프라인 객체(소비자)에게까지 캐스케이딩되는지 확인하여 무한 대기를 방지할 수 있습니다. 모든 활동, 데이터 노드 및 사전 조건에는 이름이 `failureAndRerunMode`이고 기본값이 none인 파일이 있습니다. 캐스케이드 실패를 활성화하려면 `failureAndRerunMode` 필드를 `cascade`로 설정합니다.

이 필드가 활성화되면 파이프라인 객체가 `WAITING_ON_DEPENDENCIES` 상태에서 차단되고, 대기 중인 명령 없이 종속 요소가 실패한 경우 캐스케이드 실패가 발생합니다. 캐스케이드 실패 시 다음 이벤트가 발생합니다.

- 객체가 실패하면 그 소비자가 CASCADE_FAILED로 설정되고, 원래 객체와 그 소비자의 사전 조건이 CANCELED로 설정됩니다.
- FINISHED, FAILED 또는 CANCELED 상태의 객체는 모두 무시됩니다.

원래 실패한 객체와 연결된 사전 조건은 제외하고 캐스케이드 실패는 실패한 객체의 종속 요소(업스트림)에서 작동하지 않습니다. 캐스케이드 실패의 영향을 받는 파이프라인 객체는 onFail 같은 사후 조치나 재시도를 트리거할 수 있습니다.

캐스케이드 실패의 세부 영향은 객체 유형에 따라 다릅니다.

활동

종속 요소 중 하나가 실패하면 활동이 CASCADE_FAILED로 변하며, 이후 활동 소비자에서 캐스케이드 실패를 트리거합니다. 활동을 좌우하는 리소스가 실패할 경우 해당 활동은 CANCELED 상태가 되며, 그 모든 소비자는 CASCADE_FAILED로 변합니다.

데이터 노드 및 사전 조건

실패한 활동의 출력으로 구성되는 데이터 노드는 CASCADE_FAILED 상태로 변합니다. 데이터 노드 실패는 연결된 모든 사전 조건으로 전파되어, 사전 조건이 CANCELED 상태로 변합니다.

리소스

리소스에 좌우되는 객체가 FAILED 상태이고, 리소스 자체는 WAITING_ON_DEPENDENCIES 상태이면 리소스가 FINISHED 상태로 변합니다.

캐스케이드 실패 객체 재실행

기본적으로 활동 또는 데이터 노드만 재실행하면 연결된 리소스가 재실행됩니다. 그러나 파이프라인 객체에서 failureAndRerunMode 필드를 cascade로 설정하면 대상 객체에 대한 재실행 명령이 다음 조건에서 모든 소비자에게 전파됩니다.

- 대상 객체의 소비자가 CASCADE_FAILED 상태입니다.
- 대상 객체의 종속 요소에 대기 중인 재실행 명령이 없습니다.
- 대상 객체의 종속 요소가 FAILED, CASCADE_FAILED 또는 CANCELED 상태가 아닙니다.

CASCADE_FAILED 객체를 재실행할 때 그 종속 요소 중 하나가 FAILED, CASCADE_FAILED 또는 CANCELED 상태이면 재실행에 실패하고 객체를 CASCADE_FAILED 상태로 반환합니다. 실패한 객체를

성공적으로 재실행하려면 종속성 체인의 실패를 추적하여 실패의 원래 원인을 찾아 해당 객체를 재실행해야 합니다. 리소스에서 재실행 명령을 다시 내릴 때 이에 좌우되는 객체의 재실행도 시도합니다.

캐스케이드 실패 및 채우기

캐스케이드 실패를 활성화하고 많은 채우기를 생성하는 파이프라인이 있는 경우 파이프라인 런타임 오류로 인해 유용한 작업을 수행하지 않고 리소스가 빠르게 생성 및 삭제될 수 있습니다. 파이프라인을 저장할 때 다음 경고 메시지로이 상황에 대해 알리려고 AWS Data Pipeline 시도합니다. *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. 캐스케이드 실패는 다운스트림 활동을 로 빠르게 설정하고 더 이상 필요하지 않은 EMR 클러스터 및 EC2 리소스를 CASCADE_FAILED 종료할 수 있기 때문입니다. 짧은 시간 범위에서 파이프라인을 테스트하여 이 상황의 영향을 제한하는 것이 좋습니다.

파이프라인 정의 파일 구문

이 섹션의 지침은 AWS Data Pipeline 명령줄 인터페이스(CLI)를 사용하여 파이프라인 정의 파일을 수동으로 작업하기 위한 것입니다. 이는 AWS Data Pipeline 콘솔을 사용하여 대화형으로 파이프라인을 설계하는 대신 사용할 수 있습니다.

UTF-8 파일 형식을 사용하여 파일 저장을 지원하는 텍스트 편집기를 사용하여 파이프라인 정의 파일을 수동으로 생성하고 AWS Data Pipeline 명령줄 인터페이스를 사용하여 파일을 제출할 수 있습니다.

AWS Data Pipeline 는 파이프라인 정의 내에서 다양한 복잡한 표현식과 함수도 지원합니다. 자세한 내용은 파이프라인 표현식 및 함수 단원을 참조하십시오.

파일 구조

파이프라인 생성의 첫 단계는 파이프라인 정의 파일에서 파이프라인 정의 객체를 구성하는 일입니다. 다음 예제는 파이프라인 정의 파일의 일반 구조를 보여줍니다. 이 파일은 두 객체를 정의하는데, 이 객체는 '{' 및 '}'로 구분되고, 콤마로 분리됩니다.

다음 예제에서는 첫 번째 객체가 필드라고 하는 이름-값 쌍 2개를 정의합니다. 두 번째 객체는 3개 필드를 정의합니다.

```
{
  "objects" : [
    {
       "name1" : "value1",
```

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

파이프라인 정의 파일을 생성할 때는 필요하게 될 파이프라인 객체 유형을 선택하고, 이것을 파이프라인 정의 파일에 추가한 다음 해당 필드를 추가해야 합니다. 파이프라인 객체에 대한 자세한 내용은 파이프라인 객체 참조 섹션을 참조하세요.

예를 들어, 입력 데이터 노드용 파이프라인 정의 객체와 출력 데이터 노드용 파이프라인 정의 객체를 생성할 수 있을 것입니다. 그런 다음 Amazon EMR을 사용하여 입력 데이터를 처리하는 것과 같은 활동의 파이프라인 정의 객체도 생성합니다.

파이프라인 필드

파이프라인 정의 파일에 포함시킬 객체 유형을 확인한 후에 각 파이프라인 객체의 정의에 필드를 추가합니다. 필드 이름은 인용 부호 안에 들어가며, 다음 예제에서처럼 스페이스, 콜론, 스페이스로 필드 값과 분리됩니다.

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

텍스트 문자열, 다른 객체의 참조, 함수 호출, 표현식 또는 이 모든 유형의 정렬 목록이 필드 값이 될 수 있습니다. 필드 값에 사용할 수 있는 데이터 형식에 대한 자세한 내용은 단순한 데이터 유형 단원을 참조하세요. 필드 값을 평가할 때 사용할 수 있는 함수에 대한 자세한 내용은 표현식 평가 단원을 참조하세요.

필드는 2048자로 제한됩니다. 객체 크기는 20KB이므로 객체에 큰 필드 여러 개를 추가할 수 없습니다.

각 파이프라인 객체에는 다음 예제와 같이 `id` 및 `type` 필드가 포함되어야 합니다. 객체 유형에 따라 이 외의 필드가 필요할 수도 있습니다. 자신에게 의미가 있고, 파이프라인 정의 안에서 고유한 `id` 값을 선택하세요. `type` 값은 객체 유형을 지정합니다. 지원되는 파이프라인 정의 객체 유형(파이프라인 객체 참조 주제에 나열되어 있는) 중 하나를 지정하세요.

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

```
}
```

각 객체의 필수 필드와 선택 필드에 관한 자세한 내용은 객체 문서를 참조하세요.

다른 객체에 있는 한 객체의 필드를 포함시키려면 객체 참조와 함께 parent 필드를 사용합니다. 예를 들어, 객체 "B"에는 그 필드 "B1"과 "B2" 그리고 객체 "A", "A1", "A2"의 필드가 포함됩니다.

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

ID "Default"로 객체에서 공통 필드를 정의할 수 있습니다. 이러한 필드는 그 parent 필드를 다른 객체의 참조로 명시적으로 설정하지 않는 파이프라인 정의 파일 내 각 객체에 자동으로 포함됩니다.

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

사용자 정의 필드

파이프라인 구성요소에서 사용자 정의 또는 사용자 지정 필드를 생성하고, 표현식으로 참조할 수 있습니다. 다음 예제는 이름이 myCustomField 및 my_customFieldReference이고, S3DataNode 객체에 추가된 사용자 지정 필드를 보여줍니다.

```
{
  "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"}
```

```
  },
```

사용자 정의 필드의 이름은 모두 소문자인 "my" 단어가 앞에 붙고, 그 뒤에 대문자 또는 밑줄 문자가 있어야 합니다. 그리고 사용자 정의 필드는 앞의 `myCustomField` 예제와 같이 문자열 값 또는 앞의 `my_customFieldReference` 예제와 같은 파이프라인 구성요소의 참조일 수 있습니다.

> **Note**
>
> 사용자 정의 필드에서는 사용자가 추가하는 사용자 지정 필드 문자열 값이 아닌 다른 파이프라인 구성 요소에 대한 유효한 참조 AWS Data Pipeline 만 확인합니다.

API 작업

> **Note**
>
> 와 상호 작용하는 프로그램을 작성하지 않는 경우 AWS SDKs를 설치할 필요가 AWS Data Pipeline없습니다. 콘솔 또는 명령줄 인터페이스를 사용하여 파이프라인을 생성하고 실행할 수 있습니다. 자세한 내용은 에 대한 설정 AWS Data Pipeline 섹션을 참조하세요.

와 상호 작용하는 애플리케이션을 작성 AWS Data Pipeline 하거나 사용자 지정 Task Runner를 구현하는 가장 쉬운 방법은 AWS SDKs. AWS SDK에는 원하는 프로그래밍 환경에서 웹 서비스 API를 간단하게 호출하는 기능이 있습니다. 자세한 내용은 AWS SDK 설치 단원을 참조하십시오.

AWS SDK 설치

AWS SDKs는 API를 래핑하고 서명 계산, 요청 재시도 처리, 오류 처리와 같은 많은 연결 세부 정보를 처리하는 함수를 제공합니다. SDKs에는를 호출하는 애플리케이션 작성을 시작하는 데 도움이 되는 샘플 코드, 자습서 및 기타 리소스도 포함되어 있습니다 AWS. SDK에서 래퍼 함수를 호출하면 AWS 애플리케이션 작성 프로세스를 크게 간소화할 수 있습니다. AWS SDKs를 다운로드하고 사용하는 방법에 대한 자세한 내용은 샘플 코드 및 라이브러리를 참조하십시오.

AWS Data Pipeline 지원은 다음 플랫폼SDKs에서 사용할 수 있습니다.

- Java용 AWS SDK
- Node.js용 AWS SDK
- PHP용 AWS SDK

- Python용 AWS SDK(Boto)
- Ruby 용 AWS SDK
- AWS SDK for .NET

에 HTTP 요청 AWS Data Pipeline

의 프로그래밍 객체에 대한 전체 설명은 AWS Data Pipeline API 참조를 AWS Data Pipeline참조하세요.

AWS SDKs 중 하나를 사용하지 않는 경우 POST 요청 방법을 사용하여 HTTP를 통해 AWS Data Pipeline 작업을 수행할 수 있습니다. POST 메서드를 사용하는 경우, 요청의 헤더에 작업을 지정하고 요청 본문에 JSON 형식으로 작업 데이터를 입력해야 합니다.

HTTP 헤더 콘텐츠

AWS Data Pipeline HTTP 요청의 헤더에 다음 정보가 필요합니다.

- `host` AWS Data Pipeline 엔드포인트입니다.

  엔드포인트에 대한 자세한 내용은 리전 및 엔드포인트를 참조하십시오.
- `x-amz-date` HTTP 날짜 헤더 또는 AWS x-amz-date 헤더에 타임스탬프를 제공해야 합니다. (일부 HTTP 클라이언트 라이브러리에서는 날짜 헤더를 설정할 수 없습니다.) x-amz-date 헤더가 있으면 요청 인증 시 모든 날짜 헤더가 무시됩니다.

  HTTP/1.1 RFC에 지정된 다음 3개 형식 중 하나로 날짜를 지정해야 합니다.
  - Sun, 06 Nov 1994 08:49:37 GMT (RFC 822, RFC 1123 이후)
  - Sunday, 06-Nov-94 08:49:37 GMT(RFC 1036에 의해 폐기된 RFC 850)
  - Sun Nov 6 08:49:37 1994(ANSI C asctime() 형식)
- `Authorization` AWS가 요청의 유효성 및 진위를 확인하기 위해 사용하는 인증 파라미터의 집합입니다. 이 헤더를 생성하는 자세한 방법은 Signature Version 4 Signing Process 단원을 참조하십시오.
- `x-amz-target` 요청 및 데이터 작업의 대상 서비스로 다음과 같은 형식을 띕니다. `<<serviceName>>_<<API version>>.<<operationName>>`

  예: `DataPipeline_20121129.ActivatePipeline`
- `content-type` JSON 및 버전을 지정하는 부분입니다. 예: `Content-Type: application/x-amz-json-1.0`

다음은 파이프라인을 활성화하기 위한 HTTP 요청의 헤더 예제입니다.

```
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
```

HTTP 본문

HTTP 요청의 본문에는 HTTP 요청의 헤더에 지정한 작업의 데이터가 포함되며, 데이터는 각 AWS Data Pipeline API의 JSON 데이터 스키마에 따라 형식이 지정되어야 합니다. AWS Data Pipeline JSON 데이터 스키마는 각 작업에 사용할 수 있는 데이터 및 파라미터 유형(예: 비교 연산자 및 열거 상수)을 정의합니다.

HTTP 요청의 본문 형식 지정

JSON 데이터 형식을 사용하여 데이터 값과 데이터 구조를 동시에 표현합니다. 괄호 표기를 사용하여 요소를 다른 요소 안에 중첩할 수 있습니다. 다음 예제는 객체 3개와 해당 슬롯으로 구성되는 파이프라인 정의의 입력 요청을 보여줍니다.

```
{
 "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"}

      ]
    }
  ]
}
```

HTTP 응답 처리

다음은 HTTP 응답의 몇 가지 중요 헤더와 애플리케이션에서 이를 처리하는 방법입니다.

- HTTP/1.1—이 헤더 다음에는 상태 코드가 이어집니다. 코드 값 200은 작업 성공을 나타냅니다. 그 외의 값은 오류를 나타냅니다.
- x-amzn-RequestId -이 헤더에는 요청 문제를 해결해야 하는 경우 사용할 수 있는 요청 ID가 포함되어 있습니다 AWS Data Pipeline. 요청 ID의 예로는 K2QH8DNOU907N97FNA2GDLL8OBVV4KQNSO5AEMVJF66Q9ASUAAJG가 있습니다.
- x-amz-crc32 - HTTP 페이로드의 CRC32 체크섬을AWS Data Pipeline 계산하고 x-amz-crc32 헤더에이 체크섬을 반환합니다. 클라이언트 측에서 자체 CRC32 체크섬을 계산하여 x-amz-crc32 헤더와 비교해 보는 것이 좋습니다. 체크섬이 일치하지 않을 경우 데이터가 전송 중 손상되었을 수 있습니다. 이 경우 요청을 다시 시도해야 합니다.

AWS SDK 사용자의 경우, SDK가 Amazon DynamoDB로부터 오는 각 회신의 체크섬을 계산하고 불일치가 감지될 경우 자동으로 재시도하기 때문에 수동으로 확인을 수행하지 않아도 됩니다.

샘플 AWS Data Pipeline JSON 요청 및 응답

다음 예제는 새 파이프라인 생성 요청을 보여줍니다. 그런 다음 새로 생성된 파이프라인의 파이프라인 식별자를 포함하여 AWS Data Pipeline 응답을 표시합니다.

HTTP POST 요청

```
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 Pipeline 응답

```
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"}
```

의 보안 AWS Data Pipeline

의 클라우드 보안 AWS 이 최우선 순위입니다. AWS 고객은 보안에 가장 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 이점을 누릴 수 있습니다.

보안은 AWS 와 사용자 간의 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 는 AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호할 책임이 있습니다. AWS 또한는 안전하게 사용할 수 있는 서비스를 제공합니다. 타사 감사자는 AWS 규정 준수 프로그램 일환으로 보안의 효과를 정기적으로 테스트하고 확인합니다. 에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 프로그램 제공 범위 내 AWS 서비스규정 준수 프로그램 AWS Data Pipeline참조하세요.
- 클라우드의 보안 - 사용자의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 귀하는 귀사의 데이터 민감도, 귀사의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다 AWS Data Pipeline. 다음 주제에서는 보안 및 규정 준수 목표를 충족하도록 AWS Data Pipeline 를 구성하는 방법을 보여줍니다. 또한 AWS Data Pipeline 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법을 알아봅니다.

주제

의 데이터 보호 AWS Data Pipeline

AWS 공동 책임 모델의 데이터 보호에 적용됩니다 AWS Data Pipeline. 이 모델에 설명된 대로 AWS 는 모든를 실행하는 글로벌 인프라를 보호할 책임이 있습니다 AWS 클라우드. 사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 이 콘텐츠에는 사용하는 AWS 서비스 서비스의 보안 구성과 관리 작업이 포함되어 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 자격 증명을 보호하고 AWS 계정 AWS IAM Identity Center 또는 AWS Identity and Access Management (IAM)를 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용합니다.
- SSL/TLS를 사용하여 AWS 리소스와 통신합니다. TLS 1.2 이상을 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다 AWS CloudTrail.
- 내부의 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용합니다 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용합니다.
- 명령줄 인터페이스 또는 API를 AWS 통해에 액세스할 때 FIPS 140-2 검증 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 Federal Information Processing Standard(FIPS) 140-2를 참조하세요.
- AWS Data Pipeline 는 Amazon EMR 및 Amazon EC2 리소스용 IMDSv2를 지원합니다. Amazon EC2 Amazon EMR과 함께 IMDSv2를 사용하려면 버전 5.23.1, 5.27.1 또는 5.32 이상 또는 버전 6.2 이상을 사용하십시오. 자세한 내용은 Amazon EC2 인스턴스에 대한 메타데이터 서비스 요청 구성 및 IMDSv2 사용을 참조하십시오.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 형식 텍스트 필드에 입력하지 않는 것이 좋습니다. 여기에는 AWS Data Pipeline 또는 기타 AWS 서비스 에서 콘솔, API AWS CLI또는 AWS SDKs를 사용하여 작업하는 경우가 포함됩니다. 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버로 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 자격 증명 정보를 URL에 포함해서는 안 됩니다.

에 대한 자격 증명 및 액세스 관리 AWS Data Pipeline

보안 자격 증명은 AWS의 서비스에서 사용자를 식별하고 파이프라인과 같은 AWS 리소스의 사용 권한을 부여합니다. AWS Data Pipeline 및 AWS Identity and Access Management (IAM)의 기능을 사용하여 AWS Data Pipeline 및 다른 사용자가 보안 자격 증명을 공유하지 않고도 AWS Data Pipeline 리소스에 액세스할 수 있도록 허용할 수 있습니다.

조직들이 파이프라인에 대한 액세스 권한을 공유하여 해당 조직의 개인이 파이프라인을 공동으로 개발하고 유지할 수 있습니다. 그러나 다음을 수행해야 합니다.

- 사용자가 특정 파이프라인에 액세스할 수 있는 제어
- 생산 파이프라인이 실수로 편집되는 것 방지
- 감사자에게 파이프라인에 대한 읽기 전용 액세스 권한을 부여하되, 변경은 못하게 함

AWS Data Pipeline 는 다음과 같은 다양한 기능을 제공하는 AWS Identity and Access Management (IAM)과 통합됩니다.

- 에서 사용자 및 그룹을 생성합니다 AWS 계정.
- 의 사용자 간에 AWS 리소스를 쉽게 공유할 수 있습니다 AWS 계정.
- 각 사용자에게 고유한 보안 자격 증명을 할당합니다.
- 서비스 및 리소스에 대한 각 사용자의 액세스 권한을 제어합니다.
- AWS 계정에서 모든 사용자에 대해 단일 청구서를 받습니다.

에서 IAM을 사용하면 조직의 사용자가 특정 API 작업을 사용하여 작업을 수행할 수 있는지 여부와 특정 AWS 리소스를 사용할 수 있는지 여부를 AWS Data Pipeline제어할 수 있습니다. 파이프라인 태그 및 작업자 그룹에 기반한 IAM 정책을 사용하여 다른 사용자와 파이프라인을 공유하고 다른 사용자의 액세스 권한 레벨을 제어할 수 있습니다.

내용

에 대한 IAM 정책 AWS Data Pipeline

기본적으로 IAM 객체는 AWS 리소스를 생성 또는 수정할 수 있는 권한이 없습니다. IAM 객체에게 리소스 생성 또는 수정 및 작업 수행을 허용하려면, IAM 사용자에게 필요한 특정 리소스 및 API 작업을 사용할 권한을 부여하는 IAM 정책을 생성하고, 해당 권한을 필요로 하는 IAM 사용자 또는 그룹에게 정책을 연결해야 합니다.

사용자 또는 사용자 그룹에 정책을 연결하면 지정된 리소스에 대해 지정된 작업을 수행할 권한이 허용되거나 거부됩니다. IAM 정책에 대한 일반적 내용은 IAM 사용자 설명서에서 권한 및 정책을 참조하세요. 사용자 지정 IAM 정책 관리 및 생성에 대한 자세한 내용은 IAM 정책 관리 섹션을 참조하세요.

내용

- 정책 구문
- 태그를 이용해 파이프라인에 대한 액세스 제어
- 작업자 그룹을 이용해 파이프라인에 대한 액세스 제어

정책 구문

IAM 정책은 하나 이상의 문으로 구성된 JSON 문서입니다. 각 명령문의 구조는 다음과 같습니다.

```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"*",
    "Condition":{
      "condition":{
        "key":"value"
        }
      }
    }
  ]
}
```

정책 명령문은 다음 요소로 구성됩니다.

- 효과(Effect): 효과(effect)는 `Allow` 또는 `Deny`일 수 있습니다. 기본적으로 IAM 객체는 리소스 및 API 작업을 사용할 권한이 없으므로 모든 요청이 거부됩니다. 명시적 허용은 기본 설정을 무시합니다. 명시적 거부는 모든 허용을 무시합니다.

- 작업: 작업은 권한을 부여하거나 거부할 특정 API 작업입니다. 에 대한 작업 목록은 AWS Data Pipeline API AWS Data Pipeline참조의 작업을 참조하세요.
- 리소스: 작업의 영향을 받는 리소스입니다. 여기서 유일한 유효 값은 "*"입니다.
- 조건(Condition): 조건(Condition)은 선택 사항입니다. 정책이 적용되는 시점을 제어하는 데 사용할 수 있습니다.

  AWS Data Pipeline 는 AWS 전체 컨텍스트 키(조건에 사용 가능한 키 참조)와 다음 서비스별 키를 구현합니다.

  - `datapipeline:PipelineCreator` — 파이프라인을 생성하는 사용자에 대한 액세스 권한을 부여합니다. Grant the pipeline owner full access에 나오는 예제를 참조하세요.
  - `datapipeline:Tag` — 파이프라인 태그 지정에 따라 액세스 권한을 부여합니다. 자세한 내용은 태그를 이용해 파이프라인에 대한 액세스 제어 단원을 참조하십시오.
  - `datapipeline:workerGroup` — 작업자 그룹의 이름에 따라 액세스 권한을 부여합니다. 자세한 내용은 작업자 그룹을 이용해 파이프라인에 대한 액세스 제어 단원을 참조하십시오.

태그를 이용해 파이프라인에 대한 액세스 제어

파이프라인의 태그를 참조하는 IAM 정책을 생성할 수 있습니다. 이것을 사용하여 파이프라인 태그 지정으로 다음 작업을 수행할 수 있습니다.

- 파이프라인에 대한 읽기 전용 액세스 권한 부여
- 파이프라인에 대한 읽기/쓰기 액세스 권한 부여
- 파이프라인에 대한 액세스 차단

예를 들어, 관리자에게 생산과 개발의 두 가지 파이프라인 환경이 있고 각 환경마다 IAM 그룹이 있다고 가정해봅니다. 생산 환경의 파이프라인의 경우, 관리자가 생산 IAM 그룹의 사용자에게는 읽기/쓰기 액세스 권한을 부여하고, 개발자 IAM 그룹 사용자에게는 읽기 전용 액세스 권한만 부여합니다. 개발 환경의 파이프라인의 경우, 관리자가 생산 및 개발자 IAM 그룹 사용자 모두에게는 읽기/쓰기 액세스 권한을 부여합니다.

이 시나리오를 실현하기 위해 관리자는 생산 파이프라인에 "environment=production" 태그를 지정하고, 개발자 IAM 그룹에게 다음 정책을 연결합니다. 첫 번째 명령문은 모든 파이프라인에 대한 읽기 전용 액세스 권한을 부여합니다. 두 번째 명령문은 "environment=production" 태그가 없는 파이프라인에 대한 읽기/쓰기 액세스 권한을 부여합니다.

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"}
      }
    }
  ]
}
```

또한, 관리자는 생산 IAM 그룹에 다음 정책을 연결합니다. 이 명령문은 모든 파이프라인에 대한 모든 액세스 권한을 부여합니다.

JSON

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

```
}
```

자세한 예제는 Grant users read-only access based on a tag 및 Grant users full access based on a tag를 참조하세요.

작업자 그룹을 이용해 파이프라인에 대한 액세스 제어

작업자 그룹 이름을 참조하는 IAM 정책을 생성할 수 있습니다.

예를 들어, 관리자에게 생산과 개발의 두 가지 파이프라인 환경이 있고 각 환경마다 IAM 그룹이 있다고 가정해봅니다. 관리자는 각각 생산, 생산 전 및 개발자 환경용으로 구성된 작업 실행기가 있는 데이터베이스 서버 3개를 갖고 있습니다. 관리자는 생산 IAM 그룹 사용자가 작업을 생산 리소스로 추진하는 파이프라인을 생성할 수 있는지 그리고 개발 IAM 그룹 사용자가 작업을 생산 전 및 개발자 리소스 모두로 추진할 수 있는지 확인해야 합니다.

이 시나리오를 실현하기 위해 관리자가 생산 리소스에 생산 자격 증명을 사용하여 작업 실행기를 설치하고 `workerGroup`를 "prodresource"로 설정합니다. 그리고 관리자가 개발 자격 증명을 사용하여 개발 리소스에 작업 실행기를 설치하고 `workerGroup`을 "pre-production" 및 "development"로 설정합니다. 관리자가 개발자 IAM 그룹에 다음 정책을 연결하여 "prodresource" 리소스에 대한 액세스를 차단합니다. 첫 번째 명령문은 모든 파이프라인에 대한 읽기 전용 액세스 권한을 부여합니다. 두 번째 명령문은 작업자 그룹 이름에 접두사 "dev" 또는 "pre-prod"가 있을 때 파이프라인에 대한 읽기/쓰기 액세스 권한을 부여합니다.

그리고 관리자가 생산 IAM 그룹에 다음 정책을 연결하여 "prodresource" 리소스에 대한 액세스 권한을 부여합니다. 첫 번째 명령문은 모든 파이프라인에 대한 읽기 전용 액세스 권한을 부여합니다. 두 번째 명령문은 작업자 그룹 이름에 접두사 "prod"가 있을 때 읽기/쓰기 액세스 권한을 부여합니다.

에 대한 정책 예제 AWS Data Pipeline

다음 예제는 사용자에게 파이프라인에 대한 모든 또는 제한된 액세스 권한을 부여하는 방법을 보여줍니다.

내용

- 예제 1: 태그에 따라 사용자에게 읽기 전용 액세스 권한 부여
- 예제 2: 태그에 따라 사용자에게 모든 액세스 권한 부여
- 예제 3: 파이프라인 소유자에게 모든 액세스 권한 부여
- 예제 4: 사용자에게 AWS Data Pipeline 콘솔에 대한 액세스 권한 부여

예제 1: 태그에 따라 사용자에게 읽기 전용 액세스 권한 부여

다음 정책은 사용자가 읽기 전용 AWS Data Pipeline API 작업을 사용하도록 허용하지만 "environment=production" 태그가 있는 파이프라인에서만 사용할 수 있습니다.

ListPipelines API 작업은 태그 기반 권한 부여는 지원하지 않습니다.

JSON

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

예제 2: 태그에 따라 사용자에게 모든 액세스 권한 부여

다음 정책은 사용자가 ListPipelines을 제외한 모든 AWS Data Pipeline API 작업을 사용할 수 있도록 허용하지만 "environment=test" 태그가 있는 파이프라인에만 사용할 수 있습니다.

JSON

```
{
```

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

예제 3: 파이프라인 소유자에게 모든 액세스 권한 부여

다음 정책은 사용자가 자신의 파이프라인에서만 모든 AWS Data Pipeline API 작업을 사용할 수 있도록 허용합니다.

예제 4: 사용자에게 AWS Data Pipeline 콘솔에 대한 액세스 권한 부여

다음 정책에 따라 사용자는 AWS Data Pipeline 콘솔을 사용하여 파이프라인을 생성하고 관리할 수 있습니다.

이 정책에는 `roleARN` AWS Data Pipeline 에 연결된 특정 리소스에 대한 `PassRole` 권한에 대한 작업이 포함됩니다. 자격 증명 기반(IAM) `PassRole` 권한에 대한 자세한 내용은 블로그 게시물 Granting Permission to Launch EC2 Instances with IAM Roles (PassRole Permission)을 참조하세요.

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"
    ]
  }
 ]
}
```

에 대한 IAM 역할 AWS Data Pipeline

AWS Data Pipeline 는 AWS Identity and Access Management 역할을 사용합니다. IAM 역할에 연결된 권한 정책에 따라 수행할 수 있는 작업 AWS Data Pipeline 및 애플리케이션과 AWS 액세스할 수 있는 리소스가 결정됩니다. 자세한 내용은 IAM 사용 설명서에서 IAM 역할을 참조하세요.

AWS Data Pipeline 에는 두 가지 IAM 역할이 필요합니다.

- 파이프라인 역할은 AWS 리소스에 대한 AWS Data Pipeline 액세스를 제어합니다. 파이프라인 객체 정의에서, `role`필드가 이 역할을 지정합니다.
- EC2 인스턴스 역할은 Amazon EMR 클러스터의 EC2 인스턴스를 포함하여 EC2 인스턴스에서 실행되는 애플리케이션이 AWS 리소스에 대해 갖는 액세스를 제어합니다. 파이프라인 객체 정의에서, `resourceRole`필드가 이 역할을 지정합니다.

> **Important**
>
> 기본 역할이 있는 AWS Data Pipeline 콘솔을 사용하여 2022년 10월 3일 이전에 파이프라인을 생성한 경우에서 사용자를 `DataPipelineDefaultRole` 위해를 AWS Data Pipeline 생성하고 `AWSDataPipelineRole` 관리형 정책을 역할에 연결했습니다. 2022년 10월 3일부터 `AWSDataPipelineRole`관리형 정책은 더 이상 사용되지 않으며 콘솔을 사용할 때 파이프라인을 위해 파이프라인 역할을 지정해야 합니다.
> 기존 파이프라인을 검토하여 `DataPipelineDefaultRole`이(가) 파이프라인과 연결되어 있는지, `AWSDataPipelineRole`이(가) 해당 역할에 연결되어 있는지 결정하는 것이 좋습니다. 그런 경우, 이 정책이 허용하는 액세스를 검토하여 보안 요구 사항에 적합한지 확인하세요. 필요에 따라 이 역할에 연결된 정책 및 정책 설명을 추가, 업데이트 또는 교체하세요. 대안으로, 다른 권한 정책으로 생성한 역할을 사용하도록 파이프라인을 업데이트할 수도 있습니다.

AWS Data Pipeline 역할에 대한 권한 정책 예제

각 역할에는 역할이 액세스할 수 있는 AWS 리소스와 역할이 수행할 수 있는 작업을 결정하는 하나 이상의 권한 정책이 연결되어 있습니다. 이 항목은 파이프라인 역할에 대한 권한 정책의 예를 제공합니다. 또한 기본 EC2 인스턴스 역할 `DataPipelineDefaultResourceRole`에 대한 관리형 정책인 `AmazonEC2RoleforDataPipelineRole`의 내용도 제공합니다.

파이프라인 역할 권한 정책 예

다음 예제 정책은가 Amazon EC2 및 Amazon EMR 리소스로 파이프라인을 실행 AWS Data Pipeline 해야 하는 필수 함수를 허용하도록 범위가 지정됩니다. 또한 많은 파이프라인에 필요한 Amazon Simple Storage Service 및 Amazon Simple Notification Service와 같은 다른 AWS 리소스에 액세스할 수 있는 권한을 제공합니다. 파이프라인에 정의된 객체에 AWS 서비스의 리소스가 필요하지 않은 경우 해당 서비스에 액세스할 수 있는 권한을 제거하는 것이 좋습니다. 예를 들어 파이프라인이 DynamoDBDataNode을(를) 정의하지 않거나 SnsAlarm 작업을 사용하지 않는 경우, 해당 작업에 대한 allow 문을 제거하는 것이 좋습니다.

- 를 AWS 계정 ID*111122223333*로 바꿉니다.

- *NameOfDataPipelineRole*을(를) 파이프라인 역할(이 정책이 연결된 역할)의 이름으로 바꾸십시오.
- *NameOfDataPipelineResourceRole*을(를) EC2 인스턴스 역할 이름으로 대체합니다.
- *us-west-1*을 애플리케이션에 적합한 리전으로 대체합니다.

EC2 인스턴스 역할의 기본 관리형 정책

AmazonEC2RoleforDataPipelineRole 스크립트의 콘텐츠는 아래와 보여집니다. 이는 AWS Data Pipeline의 기본 리소스 역할에 연결된 관리형 정책입니다DataPipelineDefaultResourceRole. 파이프라인에 대한 리소스 역할을 정의할 때는이 권한 정책으로 시작한 다음 필요하지 않은 AWS 서비스 작업에 대한 권한을 제거하는 것이 좋습니다.

이 글을 쓰여진 시점의 최신 버전인 정책 버전 3이 표시됩니다. IAM 콘솔을 사용하여 최신 정책 버전을 확인합니다.

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": ["*"]
    }]
```

```
}
```

에 대한 IAM 역할 생성 AWS Data Pipeline 및 역할 권한 편집

다음 절차에 따라 IAM 콘솔을 AWS Data Pipeline 사용하여에 대한 역할을 생성합니다. 프로세스는 다음 두 단계로 구성됩니다. 첫째, 생성한 권한 정책을 생성하여 역할에 연결합니다. 그런 다음 역할을 생성하고 정책을 연결합니다. 역할을 생성한 후, 권한 정책을 연결 및 분리하여 역할의 권한을 변경할 수 있습니다.

> **Note**
>
> 아래 설명과 같이 콘솔을 AWS Data Pipeline 사용하기 위한 역할을 생성하면 IAM은 역할에 필요한 적절한 신뢰 정책을 생성하고 연결합니다.

에 대한 역할과 함께 사용할 권한 정책을 생성하려면 AWS Data Pipeline

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택한 후 정책 생성을 선택합니다.
3. JSON 탭을 선택합니다.
4. 파이프라인 역할을 생성하는 경우, 보안 요구 사항에 맞게 편집하여 파이프라인 역할 권한 정책 예의 정책 예제의 내용을 복사하여 붙여넣으십시오. 대안으로, 사용자 지정 EC2 인스턴스 역할을 생성하는 경우, EC2 인스턴스 역할의 기본 관리형 정책의 예제에 대해 동일한 작업을 수행하세요.
5. 정책 검토를 선택합니다.
6. 정책 이름(예: `MyDataPipelineRolePolicy`)과 선택 사항인 설명을 입력한 다음에 정책 생성을 선택합니다.
7. 정책의 이름을 적습니다. 역할을 생성할 때 그것이 필요합니다.

에 대한 IAM 역할을 생성하려면 AWS Data Pipeline

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
3. 사용 사례 선택에서 데이터 파이프라인을 선택합니다.
4. 사용 사례 선택에서 다음 중 하나를 수행합니다.

- `Data Pipeline`을 선택하여 파이프라인 역할을 생성합니다.
- `EC2 Role for Data Pipeline`을 선택하여 리소스 역할을 생성합니다.

5. 다음: 권한을 선택합니다.
6. 에 대한 기본 정책 AWS Data Pipeline 이 나열된 경우 다음 단계를 수행하여 역할을 생성한 다음 다음 다음 절차의 지침에 따라 편집합니다. 그렇지 않으면, 위 절차에서 생성한 정책의 이름을 입력한 다음 그것을 목록에서 선택합니다.
7. 다음: 태그를 선택하고, 역할에 추가할 태그를 입력한 후, 다음: 검토를 선택합니다.
8. 역할 이름(예: `MyDataPipelineRole`) 및 선택 사항인 설명을 입력한 다음 역할 생성을 선택합니다.

에 대한 IAM 역할에 대한 권한 정책을 연결하거나 분리하려면 AWS Data Pipeline

1. IAM 콘솔(https://console.aws.amazon.com/iam/)을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 검색 상자에 편집하려는 역할 이름(예: DataPipelineDefaultRole 또는 MyDataPipelineRole)을 입력한 다음 목록에서 역할 이름을 선택합니다.
4. 권한 탭에서 다음을 수행합니다.
   - 권한 정책을 분리하려면 권한 정책에서 정책 항목의 맨 오른쪽에 있는 제거 버튼을 선택합니다. 확인하라는 메시지가 나타나면 분리를 선택합니다.
   - 이전에 만든 정책을 연결하려면 정책 연결을 선택합니다. 검색 상자에 편집하려는 정책의 이름을 입력하고 목록에서 정책을 선택한 다음 정책 연결을 선택합니다.

기존 파이프라인의 역할 변경

파이프라인에 다른 파이프라인 역할 또는 리소스 역할을 할당하려는 경우 AWS Data Pipeline 콘솔에서 아키텍트 편집기를 사용할 수 있습니다.

콘솔을 사용하여 파이프라인에 할당된 역할을 편집하려면

1. https://console.aws.amazon.com/datapipeline/ AWS Data Pipeline 콘솔을 엽니다.
2. 목록에서 파이프라인을 선택한 다음 작업, 편집을 선택합니다.
3. 아키텍트 에디터의 오른쪽 창에서 기타를 선택합니다.

4. 리소스 역할 및 역할 목록에서 할당 AWS Data Pipeline 할 역할을 선택한 다음 저장을 선택합니다.

에서 로깅 및 모니터링 AWS Data Pipeline

AWS Data Pipeline 는 사용자 AWS CloudTrail, 역할 또는 서비스가 수행한 작업에 대한 레코드를 제공하는 AWS 서비스와 통합됩니다 AWS Data Pipeline. CloudTrail은에 대한 모든 API 호출을 이벤트 AWS Data Pipeline 로 캡처합니다. 캡처되는 호출에는 AWS Data Pipeline 콘솔의 호출과 API 작업에 대한 코드 호출이 AWS Data Pipeline 포함됩니다. 추적을 생성하면 이벤트를 포함한 CloudTrail 이벤트를 지속적으로 Amazon S3 버킷에 배포할 수 있습니다 AWS Data Pipeline. 추적을 구성하지 않은 경우에도 이벤트 기록에서 CloudTrail 콘솔의 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 수행된 요청, 요청이 수행된 AWS Data Pipeline IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

CloudTrail에 대한 자세한 설명은 AWS CloudTrail 사용자 가이드를 참조하십시오.

AWS Data Pipeline CloudTrail의 정보

AWS 계정을 생성할 때 계정에서 CloudTrail이 활성화됩니다. 에서 활동이 발생하면 AWS Data Pipeline해당 활동이 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. AWS 계정에서 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 설명은 CloudTrail 이벤트 기록으로 이벤트 보기를 참조하세요.

에 대한 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려면 추적을 AWS Data Pipeline 생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 콘솔에서 추적을 생성하면 기본적으로 모든 리전에 추적이 적용됩니다. 추적은 AWS 파티션의 모든 리전에서 이벤트를 로깅하고 지정한 Amazon S3 버킷으로 로그 파일을 전송합니다. 추가적으로, CloudTrail 로그에서 수집된 이벤트 데이터를 추가 분석 및 처리하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음 자료를 참조하세요.

- 추적 생성 개요
- CloudTrail 지원 서비스 및 통합
- CloudTrail에서 Amazon SNS 알림 구성
- 여러 리전으로부터 CloudTrail 로그 파일 받기 및 여러 계정으로부터 CloudTrail 로그 파일 받기

모든 AWS Data Pipeline 작업은 CloudTrail에서 로깅되며 AWS Data Pipeline API 참조 작업 장에 설명되어 있습니다. 예를 들어, CreatePipeline 작업에 대한 호출은 로그 파일의 항목을 생성합니다.

모든 이벤트 또는 로그 항목에는 요청을 생성했던 사용자에 관한 정보가 포함됩니다. ID 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 IAM 역할 보안 인증 정보로 했는지 여부.
- 역할 또는 페더레이션 사용자의 임시 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 요청이 다른 AWS 서비스에서 이루어졌는지 여부입니다.

자세한 설명은 CloudTrail userIdentity 요소를 참조하세요.

AWS Data Pipeline 로그 파일 항목 이해

트레일이란 지정한 S3 버킷에 이벤트를 로그 파일로 입력할 수 있게 하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 이벤트는 모든 소스로부터의 단일 요청을 나타내며 요청 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 들어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 직접 호출의 주문 스택 트레이스가 아니므로 특정 순서로 표시되지 않습니다.

다음 예제는 `CreatePipeline` 작업을 보여주는 CloudTrail 로그 항목입니다.

```
    {
"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
  ]
}
```

의 인시던트 대응 AWS Data Pipeline

에 대한 인시던트 대응 AWS Data Pipeline 은 AWS responsibility. AWS 에는 인시던트 대응을 관리하는 문서화된 공식 정책 및 프로그램이 있습니다.

널리 영향을 미치는 AWS 운영 문제는 AWS Service Health Dashboard에 게시됩니다. Personal Health Dashboard를 통해 개별 계정에도 운영 문제가 게시됩니다.

에 대한 규정 준수 검증 AWS Data Pipeline

AWS Data Pipeline 는 AWS 규정 준수 프로그램의 범위에 속하지 않습니다. 특정 규정 준수 프로그램 범위에 속하는 AWS 서비스의 목록은 규정 준수 프로그램 제공 범위 내AWS 서비스를 참조하세요. 일반적인 내용은 AWS 규정 준수 프로그램을 참조하세요.

의 복원력 AWS Data Pipeline

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다. AWS 리전은 물리적으로 분리되고 격리된 여러 가용 영역을 제공하며,이 가용 영역은 지연 시간이 짧고 처리량이 높으며 중복성이 높은 네트워킹과 연결됩니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 내용은 AWS 글로벌 인프라를 참조하세요.

의 인프라 보안 AWS Data Pipeline

관리형 서비스인는 Amazon Web Services: 보안 프로세스 개요 백서에 설명된 AWS 글로벌 네트워크 보안 절차로 AWS Data Pipeline 보호됩니다.

AWS 에서 게시한 API 호출을 사용하여 네트워크를 AWS Data Pipeline 통해에 액세스합니다. 클라이언트가 전송 계층 보안(TLS) 1.0 이상을 지원해야 합니다. TLS 1.2 이상을 권장합니다. 클라이언트는 Ephemeral Diffie-Hellman(DHE) 또는 Elliptic Curve Ephemeral Diffie-Hellman(ECDHE)과 같은 PFS(전달 완전 보안, Perfect Forward Secrecy)가 포함된 암호 제품군도 지원해야 합니다. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 위탁자와 관련된 시크릿 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 보안 자격 증명을 생성하여 요청에 서명할 수 있습니다.

의 구성 및 취약성 분석 AWS Data Pipeline

구성 및 IT 제어는 AWS 와 고객 간의 공동 책임입니다. 자세한 내용은 AWS 공동 책임 모델을 참조하세요.

자습서

다음 자습서는 AWS Data Pipeline으로 파이프라인을 생성하고 사용하는 과정을 단계별로 설명합니다.

자습서

- Hadoop 스트리밍과 함께 Amazon EMR을 사용하여 데이터 처리
- 를 사용하여 Amazon S3 버킷 간에 CSV 데이터 복사 AWS Data Pipeline
- 를 사용하여 MySQL 데이터를 Amazon S3로 내보내기 AWS Data Pipeline
- 를 사용하여 Amazon Redshift에 데이터 복사 AWS Data Pipeline

Hadoop 스트리밍과 함께 Amazon EMR을 사용하여 데이터 처리

AWS Data Pipeline 를 사용하여 Amazon EMR 클러스터를 관리할 수 있습니다. 를 AWS Data Pipeline 사용하여 클러스터가 시작되기 전에 충족해야 하는 사전 조건(예: 오늘의 데이터가 Amazon S3에 업로드되었는지 확인), 클러스터를 반복적으로 실행하기 위한 일정 및 사용할 클러스터 구성을 지정할 수 있습니다. 다음 자습서는 단순한 클러스터를 시작하는 방법을 설명합니다.

이 자습서에서 단순한 Amazon EMR 클러스터용 파이프라인을 생성하여 Amazon EMR이 제공한 기존 Hadoop Streaming 작업을 실행하고, 작업이 성공적으로 완료된 후에 Amazon SNS 알림을 전송합니다. 이 작업에 AWS Data Pipeline 에서 제공하는 Amazon EMR 클러스터 리소스를 사용합니다. 샘플 애플리케이션은 이름이 WordCount이며, 콘솔에서 수동으로 실행할 수도 있습니다. 사용자를 대신하여 AWS Data Pipeline 에서 생성된 클러스터는 Amazon EMR 콘솔에 표시되며 AWS 계정으로 청구됩니다.

파이프라인 객체

파이프라인은 다음 객체를 사용합니다.

EmrActivity

파이프라인에서 수행할 작업을 정의합니다(Amazon EMR이 제공한 기존 Hadoop Streaming 작업 실행).

EmrCluster

리소스 AWS Data Pipeline 는를 사용하여이 활동을 수행합니다.

클러스터는 Amazon EC2 인스턴스 세트입니다.는 클러스터를 AWS Data Pipeline 시작한 다음 작업이 완료된 후 클러스터를 종료합니다.

일정

이 활동의 시작 날짜, 시간 및 기간입니다. 종료 날짜와 시간을 지정할 수도 있습니다.

SnsAlarm

작업이 성공적으로 끝나면 사용자가 지정한 항목으로 Amazon SNS 알림을 전송합니다.

내용

시작하기 전

다음 단계가 완료되어야 합니다.

- 에 대한 설정 AWS Data Pipeline의 작업을 완료합니다.
- (선택사항) 클러스터의 VPC와 VPC의 보안 그룹을 설정합니다.
- 이메일 알림을 전송할 주제를 생성하고 주제의 Amazon Resource Name(ARN)을 메모합니다. 자세한 내용은 Amazon Simple Notification Service 시작 안내서의 항목 생성하기 단원을 참조하십시오.

명령줄을 사용하여 클러스터 시작

Amazon EMR 클러스터를 정기적으로 실행하여 웹 로그를 분석하거나 과학 데이터 분석을 수행하는 경우 AWS Data Pipeline 를 사용하여 Amazon EMR 클러스터를 관리할 수 있습니다. 를 사용하면 클러스터가 시작되기 전에 충족해야 하는 사전 조건을 지정할 AWS Data Pipeline수 있습니다(예: 오늘의 데이터가 Amazon S3에 업로드되었는지 확인). 이 자습서는 단순한 Amazon EMR 기반 파이프라인의 모델이 될 수 있는 클러스터 또는 좀 더 관련된 파이프라인의 일부로 클러스터를 시작하는 방법을 설명합니다.

사전 조건

CLI를 사용하려면 다음 단계를 완료해야 합니다.

1. 명령줄 인터페이스(CLI)를 설치하고 구성합니다. 자세한 내용은 액세스 AWS Data Pipeline 단원을 참조하십시오.
2. DataPipelineDefaultRole 및 DataPipelineDefaultResourceRole이라는 이름의 IAM 역할이 존재하는지 확인하십시오. AWS Data Pipeline 콘솔은 이러한 역할을 자동으로 생성합니다. AWS Data

Pipeline 콘솔을 한 번 이상 사용하지 않은 경우 이러한 역할을 수동으로 생성해야 합니다. 자세한 내용은 에 대한 IAM 역할 AWS Data Pipeline 단원을 참조하십시오.

작업

- 파이프라인 정의 파일 생성
- 파이프라인 정의 업로드 및 활성화
- 파이프라인 실행 모니터링

파이프라인 정의 파일 생성

다음 코드는 Amazon EMR이 제공한 기존 Hadoop 스트리밍 작업을 실행하는 단순한 Amazon EMR 클러스터의 파이프라인 정의 파일입니다. 이 샘플 애플리케이션은 이름이 WordCount이며, Amazon EMR 콘솔을 사용하여 실행할 수도 있습니다.

이 코드를 텍스트 파일로 복사하고 `MyEmrPipelineDefinition.json`으로 저장합니다. Amazon S3 버킷 위치를 사용자가 소유한 Amazon S3 버킷 이름으로 바꿔야 합니다. 시작 및 종료 날짜도 바꿔야 합니다. 클러스터`startDateTime`를 즉시 시작하려면를 과거 1일, 미래 `endDateTime` 1일로 설정합니다. AWS Data Pipeline 그런 다음는 작업 백로그로 인식되는 것을 해결하기 위해 "지난 기한" 클러스터를 즉시 시작하기 시작합니다. 이 채우기는 첫 번째 클러스터가 AWS Data Pipeline 시작되기까지 한 시간 정도 기다릴 필요가 없음을 의미합니다.

```
{
  "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"
    }
  ]
}
```

이 파이프라인은 다음 3개의 객체가 있습니다.

- `Hourly`: 작업 일정을 나타냅니다. 활동에서 필드 하나로 일정을 설정할 수 있습니다. 그러면 활동이 해당 일정(이 경우는 매시간)에 따라 실행됩니다.
- `MyCluster`, 이는 클러스터를 실행할 때 사용한 Amazon EC2 인스턴스의 세트를 나타냅니다. 클러스터로 실행할 EC2 인스턴스의 크기와 수를 지정할 수 있습니다. 인스턴스 수를 지정하지 않으면 클러스터가 마스터 노드와 작업 노드 2개로 시작됩니다. 클러스터를 시작할 서브넷을 지정할 수 있습니다. Amazon EMR이 제공하는 AMI에 추가 소프트웨어를 로드할 부트스트랩 작업 같이 클러스터에 구성을 더 추가할 수 있습니다.
- `MyEmrActivity`, 이는 클러스터에서 처리할 계산을 나타냅니다. Amazon EMR은 스트리밍, 캐스케이딩 및 Scripted Hive를 비롯한 여러 유형의 클러스터를 지원합니다. `runsOn` 필드는 클러스터의 기본 사양으로 사용하는 MyCluster를 다시 참조합니다.

파이프라인 정의 업로드 및 활성화

파이프라인 정의를 업로드하고 파이프라인을 활성화해야 합니다. 다음 예제 명령에서 *`pipeline_name`*을 파이프라인의 레이블로 대체하고, *`pipeline_file`*을 파이프라인 정의 `.json` 파일의 정규화된 경로로 대체하십시오.

AWS CLI

파이프라인 정의를 생성하고 파이프라인을 활성화하려면 다음 create-pipeline 명령을 사용하십시오. 대부분의 CLI 명령에서 이 값을 사용하므로 파이프라인 ID를 기록하십시오.

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

파이프라인 정의를 업로드하려면 다음 put-pipeline-definition 명령을 사용하십시오.

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

파이프라인 유효성 검사에 성공하면 validationErrors 필드가 비게 됩니다. 경고가 있으면 검토해야 합니다.

파이프라인을 활성화하려면 다음 activate-pipeline 명령을 사용하십시오.

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

다음 list-pipelines 명령을 사용하면 파이프라인 목록에 파이프라인이 표시되는지 확인할 수 있습니다.

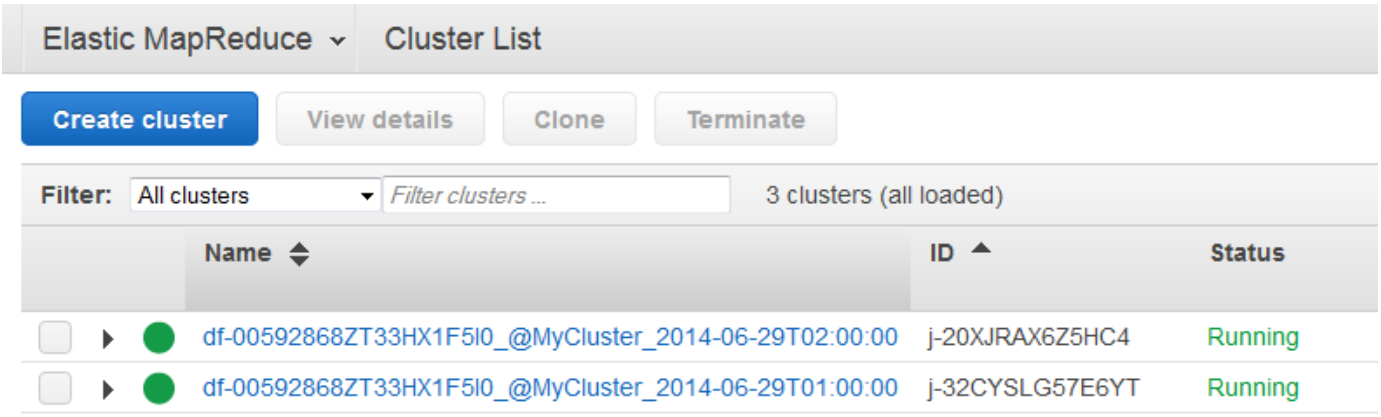
```
aws datapipeline list-pipelines
```

파이프라인 실행 모니터링

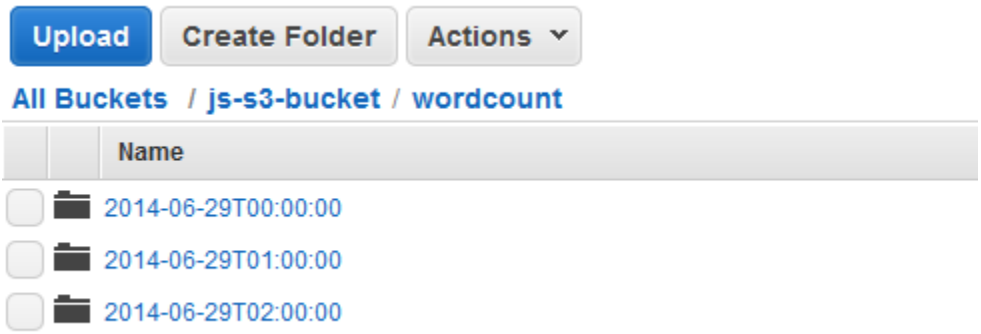
Amazon EMR 콘솔을 AWS Data Pipeline 사용하여 시작된 클러스터를 볼 수 있으며 Amazon S3 콘솔을 사용하여 출력 폴더를 볼 수 있습니다.

에서 시작한 클러스터의 진행 상황을 확인하려면 AWS Data Pipeline

1. Amazon EMR 콘솔을 엽니다.
2. 에서 생성된 클러스터의 이름은 *<pipeline-identifier>*_@*<emr-cluster-name>*_*<launch-time>* 형식 AWS Data Pipeline 입니다.



3. 실행 중 하나가 완료되면, Amazon S3 콘솔을 열고 타임스탬프가 지정된 출력 폴더가 존재하는지 그리고 그 안에 클러스터의 예상 결과가 포함되는지 확인합니다.



를 사용하여 Amazon S3 버킷 간에 CSV 데이터 복사 AWS Data Pipeline

를 AWS Data Pipeline란 무엇인가요? 읽고 AWS Data Pipeline 사용하여 데이터의 이동 및 변환을 자동화하기로 결정한 후에는 이제 데이터 파이프라인 생성을 시작해야 합니다. AWS Data Pipeline 작동 방식을 쉽게 이해할 수 있도록 간단한 작업을 살펴보겠습니다.

이 자습서는 데이터 파이프라인을 생성하여 한 Amazon S3 버킷에서 다른 버킷으로 데이터를 복사하고, 복사 작업이 성공적으로 완료된 후에 Amazon SNS 알림을 전송하는 과정을 설명합니다. 이 복사 활동에에서 관리하는 EC2 인스턴스 AWS Data Pipeline 를 사용합니다.

파이프라인 객체

파이프라인은 다음 객체를 사용합니다.

CopyActivity

이 파이프라인에 대해가 AWS Data Pipeline 수행하는 활동(한 Amazon S3 버킷에서 다른 버킷으로 CSV 데이터 복사).

> **Important**
>
> `CopyActivity` 및 `S3DataNode`에서 CSV 파일 형식을 사용할 때 제한이 있습니다. 자세한 내용은 CopyActivity 단원을 참조하십시오.

일정

이 활동의 시작 날짜, 시간 및 반복입니다. 종료 날짜와 시간을 지정할 수도 있습니다.

Ec2Resource

가이 활동을 수행하는 데 AWS Data Pipeline 사용하는 리소스(EC2 인스턴스)입니다.

S3DataNode

이 파이프라인의 입력 및 출력 노드(Amazon S3 버킷)입니다.

SnsAlarm

지정된 조건이 충족되면 작업을 수행해야 AWS Data Pipeline 합니다(작업이 성공적으로 완료된 후 주제에 Amazon SNS 알림 전송).

내용

시작하기 전

다음 단계가 완료되어야 합니다.

- 에 대한 설정 AWS Data Pipeline의 작업을 완료합니다.
- (선택사항) 인스턴스의 VPC와 VPC의 보안 그룹을 설정합니다.
- Amazon S3 버킷을 데이터 소스로 생성.

자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 생성을 참조하세요.

- Amazon S3 버킷에 데이터를 업로드합니다.

  자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷에 객체 추가를 참조하세요.
- 다른 Amazon S3 버킷을 데이터 대상으로 생성합니다.
- 이메일 알림을 전송할 주제를 생성하고 주제의 Amazon Resource Name(ARN)을 메모합니다. 자세한 내용은 Amazon Simple Notification Service 시작 안내서의 항목 생성하기 단원을 참조하십시오.
- (선택 사항) 이 자습서는 AWS Data Pipeline이 생성한 기본 IAM 역할 정책을 사용합니다. 자신의 IAM 역할 정책 및 신뢰 관계를 생성하고 구성하려면 에 대한 IAM 역할 AWS Data Pipeline에서 설명하는 지침에 따릅니다.

명령줄을 사용하여 CSV 데이터 복사

한 Amazon S3 버킷에서 다른 버킷으로 데이터를 복사할 파이프라인을 생성하고 사용할 수 있습니다.

사전 조건

시작하기 전에 다음 단계를 완료해야 합니다.

1. 명령줄 인터페이스(CLI)를 설치하고 구성합니다. 자세한 내용은 액세스 AWS Data Pipeline 단원을 참조하십시오.
2. DataPipelineDefaultRole 및 DataPipelineDefaultResourceRole이라는 이름의 IAM 역할이 존재하는지 확인하십시오. AWS Data Pipeline 콘솔은 이러한 역할을 자동으로 생성합니다. AWS Data Pipeline 콘솔을 한 번 이상 사용하지 않은 경우 이러한 역할을 수동으로 생성해야 합니다. 자세한 내용은 에 대한 IAM 역할 AWS Data Pipeline 단원을 참조하십시오.

작업

- JSON 형식으로 파이프라인 정의
- 파이프라인 정의 업로드 및 활성화

JSON 형식으로 파이프라인 정의

이 예제 시나리오는 JSON 파이프라인 정의와 AWS Data Pipeline CLI를 사용하여 두 Amazon S3 버킷 사이의 데이터 복사 일정을 특정 시간 간격으로 정하는 방법을 보여줍니다. 전체 파이프라인 정의 JSON 파일마다 각 부분의 설명이 뒤따릅니다.

> **Note**
>
> JSON 형식 파일의 구문을 확인하고 .json 파일 확장자로 파일 이름을 지정할 수 있는 텍스트 편집기를 사용할 것을 권장합니다.

쉽게 이해할 수 있도록 이 예제에서는 선택 필드는 건너뛰고 필수 필드만 설명합니다. 이 예제의 전체 파이프라인 JSON 파일은 다음과 같습니다.

```
{
  "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"
      }
    }
  ]
}
```

일정

파이프라인은 시작 및 종료 일자와 기간으로 일정을 정의하여 이 파이프라인의 활동 실행 주기를 결정합니다.

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

Amazon S3 데이터 노드

그 다음에 입력 S3DataNode 파이프라인 구성요소는 입력 파일의 위치, 이 경우는 Amazon S3 버킷 위치를 정의합니다. 입력 S3DataNode 구성요소는 다음 필드로 정의됩니다.

```
{
  "id": "S3Input",
  "type": "S3DataNode",
```

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

Id

입력 위치용 사용자 정의 이름입니다(참조용 라벨).

Type

파이프라인 구성요소 유형입니다. 즉, Amazon S3 버킷에서 데이터가 상주하는 위치에 일치할 "S3DataNode"입니다.

일정

“MySchedule”로 표시된 JSON 파일의 이전 라인에서 생성한 일정 구성요소에 대한 참조입니다.

경로

데이터 노드와 연결된 데이터 경로입니다. 데이터 노드 구문은 이 유형으로 결정됩니다. 예를 들어, Amazon S3 경로의 구문은 데이터베이스 테이블에 맞는 다른 구문을 따릅니다.

그 다음에 출력 S3DataNode 구성요소는 데이터의 출력 대상 위치를 정의합니다. 이것은 입력 S3DataNode 구성요소와 동일한 형식을 따릅니다. 단, 구성요소 이름과 대상 파일을 나타내는 경로는 다릅니다.

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

Resource

이것은 복사 작업을 수행할 전산 리소스의 정의입니다. 이 예제에서 AWS Data Pipeline 는 EC2 인스턴스를 자동으로 생성하여 복사 작업을 수행하고 작업이 완료된 후 리소스를 종료해야 합니다. 여기에 정의된 필드는 작업할 EC2 인스턴스의 생성과 기능을 제어합니다. EC2Resource는 다음 필드로 정의됩니다.

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

Id

참조용 레이블인 파이프라인 일정용 사용자 정의 이름입니다.

Type

작업을 수행할 전산 리소스 유형입니다. 이 경우는 EC2 인스턴스입니다. 다른 리소스 유형도 있습니다(예: EmrCluster 유형).

일정

이 전산 리소스를 생성할 일정입니다.

instanceType

생성할 EC2 인스턴스의 크기입니다. 수행하려는 작업의 로드와 가장 일치하는 적절한 크기의 EC2 인스턴스를 설정해야 합니다 AWS Data Pipeline. 이 경우는 m1.medium EC2 인스턴스를 설정합니다. 다른 인스턴스 유형과 각 인스턴스를 언제 사용하는지에 관한 자세한 내용은 http://aws.amazon.com/ec2/instance-types/의 Amazon EC2 인스턴스 유형 항목을 참조하십시오.

Role

리소스에 액세스할 계정의 IAM 역할입니다(예: Amazon S3 버킷에 액세스하여 데이터 검색).

resourceRole

리소스를 생성할 계정의 IAM 역할입니다(예: 사용자 대신 EC2 인스턴스 생성 및 구성). Role과 ResourceRole은 동일한 역할일 수 있지만 보안 구성에서 각각 더 세분화됩니다.

활동

수행할 작업을 나타내는 활동의 정의인 JSON 파일에 있는 마지막 선택입니다. 이 예제에서는 CopyActivity를 사용하여 http:///ec2/instance-types/ 버킷에 있는 CSV 파일의 데이터를 또 다른 파일로 복사합니다. CopyActivity 구성요소는 다음 필드로 정의됩니다.

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

Id

참조용 레이블인 활동용 사용자 정의 이름입니다.

Type

수행할 활동의 유형입니다(예: MyCopyActivity).

runsOn

이 활동이 정의하는 작업을 수행할 전산 리소스입니다. 이 예제에서는 이전에 정의한 EC2 인스턴스의 참조를 제공합니다. runs0n 필드를 사용하면 AWS Data Pipeline 에서 EC2 인스턴스를 생성합니다. [runs0n] 필드는 AWS 인프라에 리소스가 있음을 나타내며, workerGroup 값은 자체 온프레미스 리소스를 사용하여 작업을 수행해야 한다는 것을 나타냅니다.

Input

복사할 데이터의 위치입니다.

출력

대상 위치 데이터입니다.

일정

이 활동을 실행할 일정입니다.

파이프라인 정의 업로드 및 활성화

파이프라인 정의를 업로드하고 파이프라인을 활성화해야 합니다. 다음 예제 명령에서 *pipeline_name*을 파이프라인의 레이블로 대체하고, *pipeline_file*을 파이프라인 정의 .json 파일의 정규화된 경로로 대체하십시오.

AWS CLI

파이프라인 정의를 생성하고 파이프라인을 활성화하려면 다음 create-pipeline 명령을 사용하십시오. 대부분의 CLI 명령에서 이 값을 사용하므로 파이프라인 ID를 기록하십시오.

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

파이프라인 정의를 업로드하려면 다음 put-pipeline-definition 명령을 사용하십시오.

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

파이프라인 유효성 검사에 성공하면 validationErrors 필드가 비게 됩니다. 경고가 있으면 검토해야 합니다.

파이프라인을 활성화하려면 다음 activate-pipeline 명령을 사용하십시오.

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

다음 list-pipelines 명령을 사용하면 파이프라인 목록에 파이프라인이 표시되는지 확인할 수 있습니다.

```
aws datapipeline list-pipelines
```

를 사용하여 MySQL 데이터를 Amazon S3로 내보내기 AWS Data Pipeline

이 자습서는 MySQL 데이터베이스에 있는 테이블의 데이터(행)를 Amazon S3 버킷에 있는 CSV(쉼표로 구분된 값) 파일로 복사한 다음 복사 작업이 성공적으로 완료된 후 Amazon SNS 알림을 전송할 데이터 파이프라인을 생성하는 과정을 설명합니다. 이 복사 활동에는에서 제공하는 EC2 인스턴스 AWS Data Pipeline 를 사용합니다.

파이프라인 객체

파이프라인은 다음 객체를 사용합니다.

- CopyActivity
- Ec2Resource
- MySqlDataNode
- S3DataNode
- SnsAlarm

내용

시작하기 전

다음 단계가 완료되어야 합니다.

- 에 대한 설정 AWS Data Pipeline의 작업을 완료합니다.
- (선택사항) 인스턴스의 VPC와 VPC의 보안 그룹을 설정합니다.
- Amazon S3 버킷을 데이터 출력으로 생성합니다.

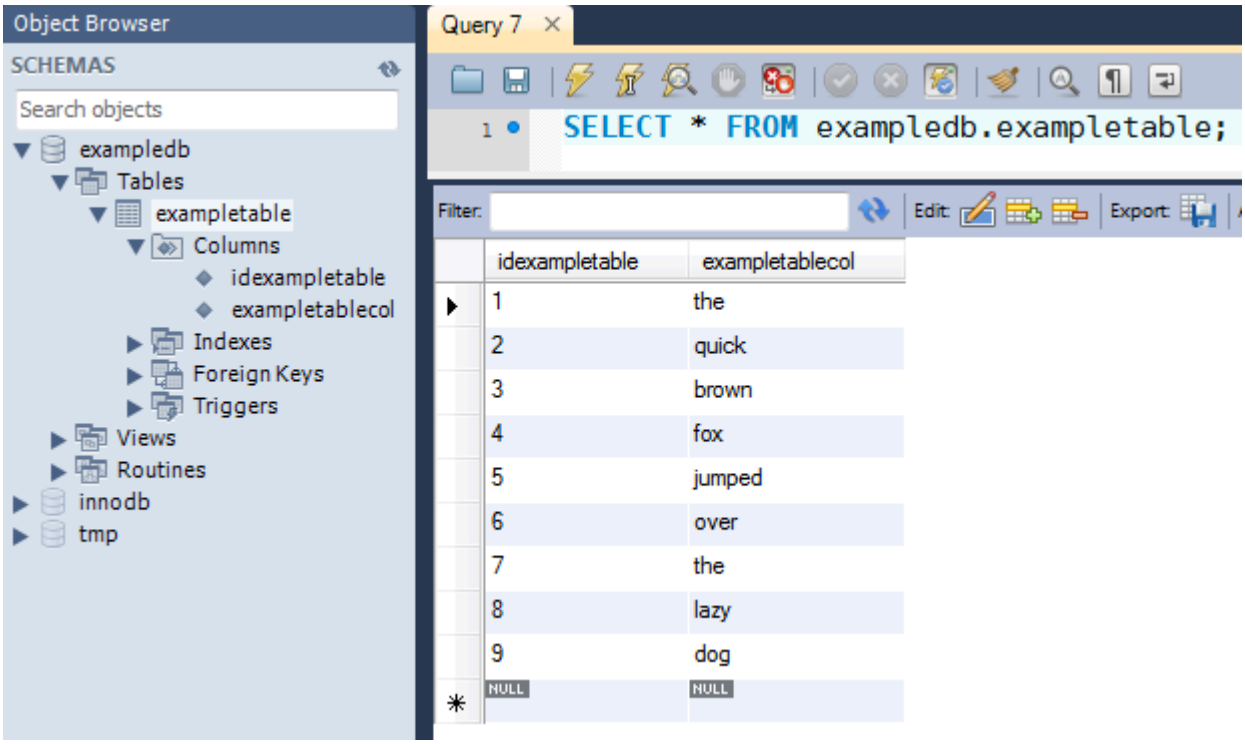
  자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 생성을 참조하십시오.
- MySQL 데이터베이스 인스턴스를 데이터 소스로 생성하여 시작합니다.

  자세한 내용을 알아보려면 Amazon RDS 시작 안내서의 DB 인스턴스 실행을 참조하십시오. Amazon RDS 인스턴스가 생성된 후에는 MySQL 설명서의 테이블 생성을 참조하십시오.

> **Note**
>
> MySQL 인스턴스를 생성할 때 사용한 사용자 이름과 암호를 메모합니다. MySQL 데이터베이스 인스턴스를 시작한 후에 인스턴스의 엔드포인트를 메모합니다. 나중에 이 정보가 필요합니다.

- MySQL 데이터베이스 인스턴스를 연결하고, 테이블을 생성한 다음, 테스트 데이터 값을 새로 생성된 테이블에 추가합니다.

  설명을 위해 다음 구성과 샘플 데이터를 갖는 MySQL 테이블로 이 자습서를 만들었습니다. 다음은 MySQL Workbench 5.2 CE의 스크린샷입니다.



  자세한 내용은 MySQL 설명서의 테이블 생성과 MySQL Workbench 제품 페이지를 참조하십시오.

- 이메일 알림을 전송할 주제를 생성하고 주제의 Amazon Resource Name(ARN)을 메모합니다. 자세한 내용은 Amazon Simple Notification Service 시작 안내서의 항목 생성하기 섹션을 참조하십시오.
- (선택 사항)이 자습서에서는에서 생성한 기본 IAM 역할 정책을 사용합니다 AWS Data Pipeline. 사용자의 IAM 역할 정책 및 신뢰 관계를 생성하고 구성하려면 에 대한 IAM 역할 AWS Data Pipeline에서 설명하는 지침에 따릅니다.

명령줄을 사용하여 MySQL 데이터 복사

MySQL 테이블의 데이터를 Amazon S3 버킷의 파일로 복사할 파이프라인을 생성할 수 있습니다.

사전 조건

시작하기 전에 다음 단계를 완료해야 합니다.

1. 명령줄 인터페이스(CLI)를 설치하고 구성합니다. 자세한 내용은 액세스 AWS Data Pipeline 단원을 참조하십시오.
2. DataPipelineDefaultRole 및 DataPipelineDefaultResourceRole이라는 이름의 IAM 역할이 존재하는지 확인하십시오. AWS Data Pipeline 콘솔은 이러한 역할을 자동으로 생성합니다. AWS Data Pipeline 콘솔을 한 번 이상 사용하지 않은 경우 이러한 역할을 수동으로 생성해야 합니다. 자세한 내용은 에 대한 IAM 역할 AWS Data Pipeline 단원을 참조하십시오.
3. Amazon S3 버킷과 Amazon RDS 인스턴스를 설정합니다. 자세한 내용은 시작하기 전 단원을 참조하십시오.

작업

- JSON 형식으로 파이프라인 정의
- 파이프라인 정의 업로드 및 활성화

JSON 형식으로 파이프라인 정의

이 예제 시나리오는 JSON 파이프라인 정의와 AWS Data Pipeline CLI를 사용하여 MySQL 데이터베이스에 있는 테이블의 데이터(행)를 Amazon S3 버킷에 있는 CSV(쉼표로 구분된 값) 파일로 지정된 시간 간격에 복사하는 방법을 설명합니다.

전체 파이프라인 정의 JSON 파일마다 각 부분의 설명이 뒤따릅니다.

> **Note**
>
> JSON 형식 파일의 구문을 확인하고 .json 파일 확장자로 파일 이름을 지정할 수 있는 텍스트 편집기를 사용할 것을 권장합니다.

```
{
  "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"
```

```
    }
  ]
}
```

MySQL 데이터 노드

입력 MySqlDataNode 파이프라인 구성요소는 입력 데이터의 위치를 정의하며, 이 경우 위치는 Amazon RDS 인스턴스입니다. 입력 MySqlDataNode 구성요소는 다음 필드로 정의됩니다.

```
{
  "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

참조 전용 라벨인 사용자 정의 이름입니다.

사용자 이름

데이터베이스 테이블에서 데이터를 검색할 권한이 있는 데이터베이스 계정의 사용자 이름입니다. *my-username*을 사용자 이름으로 바꿉니다.

일정

JSON 파일의 이전 라인에서 생성한 일정 구성요소에 대한 참조입니다.

이름

참조 전용 라벨인 사용자 정의 이름입니다.

*비밀번호

가 암호 값을 암호화해야 함을 나타내는 별표 접두사가 있는 데이터베이스 계정의 암호 AWS Data Pipeline 입니다. *my-password*를 사용자 계정의 정확한 암호로 바꿉니다. 암호 필드 앞에 별표 특수 문자가 있습니다. 자세한 내용은 특수 문자 단원을 참조하십시오.

표

복사할 데이터가 있는 데이터베이스 테이블의 이름입니다. *table-name*을 데이터베이스 테이블 이름으로 바꿉니다.

connectionString

데이터베이스에 연결할 CopyActivity 객체의 JDBC 연결 문자열입니다.

selectQuery

데이터베이스 테이블에서 복사할 데이터를 지정하는 유효 SQL SELECT 쿼리입니다. `#{table}`은 JSON 파일에서 앞 줄의 "table" 변수로 제공되는 테이블 이름을 다시 사용하는 표현식입니다.

Type

SqlDataNode 유형은 이 예제에서는 MySQL을 사용하는 Amazon RDS 인스턴스입니다.

> **Note**
>
> MySqlDataNode 유형은 사용되지 않습니다. MySqlDataNode는 계속 사용할 수 있지만 SqlDataNode 사용을 권장합니다.

Amazon S3 데이터 노드

그 다음에 S3Output 파이프라인 구성요소는 출력 파일의 위치를 정의하며, 이 경우는 Amazon S3 버킷 위치에 있는 CSV 파일입니다. 출력 S3DataNode 구성요소는 다음 필드로 정의됩니다.

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

```
},
```

Id

참조 전용 레이블인 사용자 정의 ID입니다.

일정

JSON 파일의 이전 라인에서 생성한 일정 구성요소에 대한 참조입니다.

filePath

이 예제에서 CSV 출력 파일인 데이터 노드와 연결된 데이터로 가는 경로입니다.

이름

참조 전용 라벨인 사용자 정의 이름입니다.

Type

파이프라인 객체 유형은 Amazon S3 버킷에서 데이터가 상주하는 위치와 일치해야 하는 S3DataNode입니다.

Resource

이것은 복사 작업을 수행할 전산 리소스의 정의입니다. 이 예제에서는 복사 작업을 수행하고 작업이 완료된 후 리소스를 종료하기 위해 EC2 인스턴스를 AWS Data Pipeline 자동으로 생성해야 합니다. 여기에 정의된 필드는 작업할 EC2 인스턴스의 생성과 기능을 제어합니다. EC2Resource는 다음 필드로 정의됩니다.

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

Id

참조 전용 레이블인 사용자 정의 ID입니다.

일정

이 전산 리소스를 생성할 일정입니다.

이름

참조 전용 라벨인 사용자 정의 이름입니다.

Role

리소스에 액세스할 계정의 IAM 역할입니다(예: Amazon S3 버킷에 액세스하여 데이터 검색).

Type

작업을 수행할 전산 리소스 유형입니다. 이 경우는 EC2 인스턴스입니다. 다른 리소스 유형도 있습니다(예: EmrCluster 유형).

resourceRole

리소스를 생성할 계정의 IAM 역할입니다(예: 사용자 대신 EC2 인스턴스 생성 및 구성). Role과 ResourceRole은 동일한 역할일 수 있지만 보안 구성에서 각각 더 세분화됩니다.

활동

수행할 작업을 나타내는 활동의 정의인 JSON 파일에 있는 마지막 선택입니다. 이 경우. CopyActivity 구성요소를 사용하여 Amazon S3 버킷에 있는 파일의 데이터를 다른 파일로 복사합니다. CopyActivity 구성요소는 다음 필드로 정의됩니다.

```
{
  "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

참조용 레이블인 사용자 정의 ID입니다.

Input

복사할 MySQL 데이터의 위치입니다.

일정

이 활동을 실행할 일정입니다.

이름

참조용 라벨인 사용자 정의 이름입니다.

runsOn

이 활동이 정의하는 작업을 수행할 전산 리소스입니다. 이 예제에서는 이전에 정의한 EC2 인스턴스의 참조를 제공합니다. `runsOn` 필드를 사용하면 AWS Data Pipeline 에서 EC2 인스턴스를 생성합니다. [`runsOn`] 필드는 AWS 인프라에 리소스가 있음을 나타내며, workerGroup 값은 자체 온프레미스 리소스를 사용하여 작업을 수행해야 한다는 것을 나타냅니다.

onSuccess

활동이 성공적으로 완료될 경우에 전송할 SnsAlarm

onFail

활동이 실패할 경우에 전송할 SnsAlarm

출력

CSV 출력 파일의 Amazon S3 위치

Type

실행할 활동의 유형입니다.

파이프라인 정의 업로드 및 활성화

파이프라인 정의를 업로드하고 파이프라인을 활성화해야 합니다. 다음 예제 명령에서 *pipeline_name*을 파이프라인의 레이블로 대체하고, *pipeline_file*을 파이프라인 정의 .json 파일의 정규화된 경로로 대체하십시오.

AWS CLI

파이프라인 정의를 생성하고 파이프라인을 활성화하려면 다음 create-pipeline 명령을 사용하십시오. 대부분의 CLI 명령에서 이 값을 사용하므로 파이프라인 ID를 기록하십시오.

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

파이프라인 정의를 업로드하려면 다음 put-pipeline-definition 명령을 사용하십시오.

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

파이프라인 유효성 검사에 성공하면 validationErrors 필드가 비게 됩니다. 경고가 있으면 검토해야 합니다.

파이프라인을 활성화하려면 다음 activate-pipeline 명령을 사용하십시오.

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

다음 list-pipelines 명령을 사용하면 파이프라인 목록에 파이프라인이 표시되는지 확인할 수 있습니다.

```
aws datapipeline list-pipelines
```

를 사용하여 Amazon Redshift에 데이터 복사 AWS Data Pipeline

이 자습서에서는 AWS Data Pipeline 콘솔의 Redshift로 복사 템플릿 또는 AWS Data Pipeline CLI를 사용하는 파이프라인 정의 파일을 사용하여 Amazon S3에서 Amazon Redshift로 데이터를 주기적으로 이동하는 파이프라인을 생성하는 프로세스를 안내합니다.

Amazon S3는 클라우드에 데이터를 저장하는 웹 서비스입니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서를 참조하세요.

Amazon Redshift는 클라우드의 데이터 웨어하우스 서비스입니다. 자세한 내용은 Amazon Redshift 관리 가이드의 섹션을 참조하십시오.

이 자습서는 몇 가지 사전 요구사항이 있습니다. 다음 단계를 완료한 후에만 콘솔 또는 CLI를 사용하여 자습서를 계속 진행할 수 있습니다.

내용

시작하기 전: COPY 옵션 구성 및 데이터 로드

내에서 Amazon Redshift에 데이터를 복사하기 전에 다음을 AWS Data Pipeline확인해야 합니다.

- Amazon S3에서 데이터를 로드합니다.
- Amazon Redshift에서 COPY활동을 설정합니다.

이러한 옵션이 작동하고 데이터 로드를 성공적으로 완료하면 복사를 수행하기 위해 이러한 옵션을 AWS Data Pipeline으로 전송합니다.

COPY 옵션을 알아보려면, Amazon Redshift 데이터 개발자 안내서의 COPY를 참조하십시오.

Amazon S3에서 데이터를 로드하는 단계는 Amazon Redshift 데이터베이스 개발자 안내서의 Amazon S3에서 데이터 로드를 참조하십시오.

예를 들어 Amazon Redshift에서 다음 SQL 명령은(LISTING이)라는 이름이 붙은 새 테이블을 생성하고 Amazon S3에 있는 공개적으로 사용 가능한 버킷에서 샘플 데이터를 복사합니다.

<iam-role-arn>과 리전은 사용자의 값으로 바꾸십시오.

이 예제에 대한 세부 내용은 Amazon Redshift 시작 안내서의 Amazon S3에서 샘플 데이터 로드를 참조하십시오.

```
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';
```

파이프라인 설정, 보안 그룹 생성, Amazon Redshift 클러스터 생성

자습서에 맞게 설정하려면

1. 에 대한 설정 AWS Data Pipeline의 작업을 완료합니다.
2. 보안 그룹을 생성합니다.
   a. Amazon EC2 콘솔을 엽니다.
   b. 탐색 창에서 보안 그룹을 클릭합니다.
   c. 보안 그룹 생성을 클릭합니다.
   d. 보안 그룹의 이름과 설명을 지정합니다.
   e. [EC2-클래식] VPC의 경우 No VPC를 선택합니다.
   f. [EC2-VPC] VPC에 대한 사용자 VPC의 ID를 선택합니다.
   g. 생성을 클릭합니다.
3. [EC2-Classic] Amazon Redshift 클러스터 보안 그룹을 생성하고 Amazon EC2 보안 그룹을 지정합니다.
   a. Amazon Redshift 콘솔을 엽니다.
   b. 탐색 창에서 보안 그룹을 클릭합니다.
   c. 클러스터 보안 그룹 생성을 클릭합니다.
   d. 클러스터 보안 그룹 생성 대화 상자에서 클러스터 보안 그룹의 이름과 설명을 지정합니다.
   e. 새 클러스터 보안 그룹 이름을 클릭합니다.
   f. 연결 유형 추가을 클릭합니다.
   g. 연결 유형 추가 대화 상자에서, 연결 유형에서 EC2 보안 그룹을 선택하고, EC2 보안 그룹 이름에서 생성한 보안 그룹을 선택하고, 그 다음 승인을 클릭합니다.
4. [EC2-VPC] Amazon Redshift 클러스터 보안 그룹을 생성하고 VPC 보안 그룹을 지정합니다.
   a. Amazon EC2 콘솔을 엽니다.

b. 탐색 창에서 보안 그룹을 클릭합니다.

c. 보안 그룹 생성을 클릭합니다.

d. 보안 그룹 생성 대화상자에서 보안 그룹의 이름과 설명을 지정하고 VPC의 VPC ID를 선택합니다.

e. 규칙 추가을 클릭합니다. 유형, 프로토콜 및 포트 범위를 지정하고 [Source.]에서 보안 그룹의 ID를 입력합니다. 2단계에서 생성한 보안 그룹을 선택합니다.

f. 생성을 클릭합니다.

5. 다음은 단계의 요약입니다.

   기존 Amazon Redshift 클러스터가 있다면 클러스터 ID를 적어 둡니다.

   새 클러스터를 생성하고 샘플 데이터를 로드하려면 Amazon Redshift로 시작하기의 단계에 따르십시오. 클러스터 생성에 대한 일반적인 정보는 Amazon Redshift 관리 안내서의 Amazon Redshift 클러스터 생성을 참조하십시오.

   a. Amazon Redshift 콘솔을 엽니다.

   b. [Launch Cluster]를 클릭합니다.

   c. 클러스터에 필요한 세부정보를 입력한 다음 계속를 클릭합니다.

   d. 노드 구성을 제공한 다음 계속를 클릭합니다.

   e. 추가 구성 정보 페이지에서 생성한 클러스터 보안 그룹을 선택한 다음 계속를 클릭합니다.

   f. 클러스터 사양을 검토한 다음 클러스터 시작를 클릭합니다.

명령줄을 사용하여 Amazon Redshift로 데이터 복사

이 자습서는 Amazon S3에서 Amazon Redshift로 데이터를 복사하는 방법을 설명합니다. Amazon Redshift에서 새 테이블을 생성한 다음 AWS Data Pipeline 를 사용하여 CSV 형식의 샘플 입력 데이터가 포함된 퍼블릭 Amazon S3 버킷에서이 테이블로 데이터를 전송합니다. 소유하고 있는 Amazon S3 버킷에 로그가 저장됩니다.

Amazon S3는 클라우드에 데이터를 저장하는 웹 서비스입니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서를 참조하세요. Amazon Redshift는 클라우드의 데이터 웨어하우스 서비스입니다. 자세한 내용은 Amazon Redshift 관리 가이드의 섹션을 참조하십시오.

사전 조건

시작하기 전에 다음 단계를 완료해야 합니다.

1. 명령줄 인터페이스(CLI)를 설치하고 구성합니다. 자세한 내용은 액세스 AWS Data Pipeline 단원을 참조하십시오.
2. DataPipelineDefaultRole 및 DataPipelineDefaultResourceRole이라는 이름의 IAM 역할이 존재하는지 확인하십시오. AWS Data Pipeline 콘솔은 이러한 역할을 자동으로 생성합니다. AWS Data Pipeline 콘솔을 한 번 이상 사용하지 않은 경우 이러한 역할을 수동으로 생성해야 합니다. 자세한 내용은 에 대한 IAM 역할 AWS Data Pipeline 단원을 참조하십시오.
3. Amazon Redshift에서 COPY명령을 설정하며, 그 이유는 AWS Data Pipeline내에서 복사를 수행할 때 이와 동일한 옵션이 작동해야 하기 때문입니다. 자세한 내용은 시작하기 전: COPY 옵션 구성 및 데이터 로드 단원을 참조하세요.
4. Amazon Redshift 데이터베이스를 설정하십시오. 자세한 내용은 파이프라인 설정, 보안 그룹 생성, Amazon Redshift 클러스터 생성 단원을 참조하십시오.

작업

- JSON 형식으로 파이프라인 정의
- 파이프라인 정의 업로드 및 활성화

JSON 형식으로 파이프라인 정의

이 예제 시나리오는 Amazon S3 버킷에서 Amazon Redshift로 데이터를 복사하는 방법을 보여줍니다.

전체 파이프라인 정의 JSON 파일마다 각 부분의 설명이 뒤따릅니다. JSON 형식 파일의 구문을 확인하고 `.json` 파일 확장자로 파일 이름을 지정할 수 있는 텍스트 편집기를 사용할 것을 권장합니다.

```
{
  "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"
      }
    }
  ]
}
```

이러한 객체에 대한 자세한 내용은 다음 설명서를 참조하십시오.

Objects

- 데이터 노드
- Resource
- 활동

데이터 노드

이 예제에서는 입력 데이터 노드, 출력 데이터 노드 및 데이터베이스를 사용합니다.

입력 데이터 노드

입력 S3DataNode파이프라인 구성요소는 Amazon S3의 입력 데이터 위치와 입력 데이터의 데이터 형식을 정의합니다. 자세한 내용은 S3DataNode 단원을 참조하십시오.

이 입력 구성요소는 다음 필드로 정의됩니다.

```
{
  "id": "S3DataNodeId1",
  "schedule": {
    "ref": "ScheduleId1"
  },
  "filePath": "s3://datapipeline-us-east-1/samples/hive-ads-samples.csv",
  "name": "DefaultS3DataNode1",
  "dataFormat": {
    "ref": "CSVId1"
  },
  "type": "S3DataNode"
},
```

`id`

참조 전용 레이블인 사용자 정의 ID입니다.

`schedule`

일정 구성요소 참조입니다.

`filePath`

이 예제에서 CSV 입력 파일인 데이터 노드와 연결된 데이터로 가는 경로입니다.

`name`

참조 전용 라벨인 사용자 정의 이름입니다.

`dataFormat`

이 활동에서 처리할 데이터 형식의 참조입니다.

출력 데이터 노드

출력 RedshiftDataNode파이프라인 구성요소는 출력 데이터의 위치를 정의합니다. 이 경우에는 Amazon Redshift 데이터베이스에 있는 테이블입니다. 자세한 내용은 RedshiftDataNode 단원을 참조하십시오. 이 출력 구성요소는 다음 필드로 정의됩니다.

```
{
  "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`

참조 전용 레이블인 사용자 정의 ID입니다.

`schedule`

일정 구성요소 참조입니다.

`tableName`

Amazon Redshift 테이블의 이름입니다.

`name`

참조 전용 라벨인 사용자 정의 이름입니다.

`createTableSql`

데이터베이스에서 테이블을 생성하는 SQL 표현식입니다.

`database`

Amazon Redshift 데이터베이스에 대한 참조입니다.

Database

RedshiftDatabase 구성요소는 다음 필드로 정의됩니다. 자세한 내용은 RedshiftDatabase 단원을 참조하십시오.

```
{
  "id": "RedshiftDatabaseId1",
  "databaseName": "dbname",
  "username": "user",
  "name": "DefaultRedshiftDatabase1",
  "*password": "password",
  "type": "RedshiftDatabase",
  "clusterId": "redshiftclusterId"
},
```

id

참조 전용 레이블인 사용자 정의 ID입니다.

databaseName

논리 데이터베이스의 이름입니다.

username

데이터베이스에 연결할 사용자 이름입니다.

name

참조 전용 라벨인 사용자 정의 이름입니다.

password

데이터베이스에 연결할 비밀번호입니다.

clusterId

Redshift 클러스터의 ID입니다.

Resource

이것은 복사 작업을 수행할 전산 리소스의 정의입니다. 이 예제에서 AWS Data Pipeline 는 EC2 인스턴스를 자동으로 생성하여 복사 작업을 수행하고 작업이 완료된 후 인스턴스를 종료해야 합니다. 여기에 정의된 필드는 작업할 인스턴스의 생성과 기능을 제어합니다. 자세한 내용은 Ec2Resource 단원을 참조하십시오.

Ec2Resource는 다음 필드로 정의됩니다.

```
{
  "id": "Ec2ResourceId1",
  "schedule": {
    "ref": "ScheduleId1"
  },
  "securityGroups": "MySecurityGroup",
  "name": "DefaultEc2Resource1",
  "role": "DataPipelineDefaultRole",
  "logUri": "s3://myLogs",
  "resourceRole": "DataPipelineDefaultResourceRole",
  "type": "Ec2Resource"
},
```

id

참조 전용 레이블인 사용자 정의 ID입니다.

schedule

이 전산 리소스를 생성할 일정입니다.

securityGroups

리소스 풀에서 인스턴스에 사용할 보안 그룹입니다.

name

참조 전용 라벨인 사용자 정의 이름입니다.

role

리소스에 액세스할 계정의 IAM 역할입니다(예: Amazon S3 버킷에 액세스하여 데이터 검색).

logUri

Ec2Resource에서 Task Runner 로그를 백업할 Amazon S3 대상 경로입니다.

resourceRole

리소스를 생성할 계정의 IAM 역할입니다(예: 사용자 대신 EC2 인스턴스 생성 및 구성). Role과 ResourceRole은 동일한 역할일 수 있지만 보안 구성에서 각각 더 세분화됩니다.

활동

수행할 작업을 나타내는 활동의 정의인 JSON 파일에 있는 마지막 선택입니다. 이 경우 RedshiftCopyActivity구성요소를 사용하여 Amazon S3에서 Amazon Redshift로 데이터를 복사합니다. 자세한 내용은 RedshiftCopyActivity 단원을 참조하십시오.

RedshiftCopyActivity 구성요소는 다음 필드로 정의됩니다.

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

id

참조 전용 레이블인 사용자 정의 ID입니다.

input

Amazon S3 소스 파일의 참조입니다.

schedule

이 활동을 실행할 일정입니다.

insertMode

인서트 유형(KEEP_EXISTING, OVERWRITE_EXISTING 또는 TRUNCATE)입니다.

name

참조 전용 라벨인 사용자 정의 이름입니다.

`runsOn`

이 활동이 정의하는 작업을 수행할 전산 리소스입니다.

`output`

Amazon Redshift 대상 테이블의 참조입니다.

파이프라인 정의 업로드 및 활성화

파이프라인 정의를 업로드하고 파이프라인을 활성화해야 합니다. 다음 예제 명령에서 *`pipeline_name`*을 파이프라인의 레이블로 대체하고, *`pipeline_file`*을 파이프라인 정의 `.json` 파일의 정규화된 경로로 대체하십시오.

AWS CLI

파이프라인 정의를 생성하고 파이프라인을 활성화하려면 다음 create-pipeline 명령을 사용하십시오. 대부분의 CLI 명령에서 이 값을 사용하므로 파이프라인 ID를 기록하십시오.

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

파이프라인 정의를 업로드하려면 다음 put-pipeline-definition 명령을 사용하십시오.

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

파이프라인 유효성 검사에 성공하면 `validationErrors` 필드가 비게 됩니다. 경고가 있으면 검토해야 합니다.

파이프라인을 활성화하려면 다음 activate-pipeline 명령을 사용하십시오.

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

다음 list-pipelines 명령을 사용하면 파이프라인 목록에 파이프라인이 표시되는지 확인할 수 있습니다.

```
aws datapipeline list-pipelines
```

파이프라인 표현식 및 함수

이 단원은 연결된 데이터 형식을 포함해 파이프라인에서 표현식과 함수를 사용하는 구문을 설명합니다.

단순한 데이터 유형

다음 데이터 유형을 필드 값으로 설정할 수 있습니다.

유형

- DateTime
- Numeric
- 객체 참조
- Period
- 문자열

DateTime

AWS Data Pipeline 는 "YYYY-MM-DDTHH:MM:SS" 형식으로 표현된 날짜 및 시간만 UTC/GMT로 지원합니다. 다음은 UTC/GMT 시간대에서 `Schedule` 객체의 `startDateTime` 필드를 `1/15/2012, 11:59 p.m.`으로 설정한 예제입니다.

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

Numeric

AWS Data Pipeline 는 정수와 부동 소수점 값을 모두 지원합니다.

객체 참조

파이프라인 정의에 있는 객체입니다. 이것은 현재 객체, 파이프라인 내 다른 곳에 정의된 객체의 이름 또는 키워드 node로 참조되는 현재 객체를 필드에 표시하는 객체일 수 있습니다. node에 대한 자세한 정보는 필드 및 객체 참조 섹션을 참조하세요. 파이프라인 객체 유형에 대한 자세한 내용은 파이프라인 객체 참조 단원을 참조하세요.

Period

예약된 이벤트의 실행 빈도를 나타냅니다. "N [years|months|weeks|days|hours|minutes]" 형식으로 표현되며, 여기서 N은 양의 정수 값입니다.

최소 기간은 15분이며, 최대 기간은 3년입니다.

다음 예제에서는 Schedule 객체의 period 필드를 3시간으로 설정합니다. 따라서 3시간마다 실행되는 일정이 생성됩니다.

```
"period" : "3 hours"
```

문자열

표준 문자열 값입니다. 문자열은 큰따옴표(")로 묶어야 합니다. 문자열의 이스케이프 문자에 백슬래시 문자(\)를 포함할 수 있습니다. 여러 줄의 문자열은 지원되지 않습니다.

다음 예제는 id 필드에 유효한 문자열 값 예입니다.

```
"id" : "My Data Object"

"id" : "My \"Data\" Object"
```

문자열 값을 평가하는 표현식도 문자열에 담을 수 있습니다. 이러한 표현식은 문자열에 삽입되며, "#{" 및 "}"로 구분합니다. 다음 예제에서는 표현식을 사용하여 현재 객체 이름을 경로에 삽입합니다.

```
"filePath" : "s3://amzn-s3-demo-bucket/#{name}.csv"
```

표현식 사용에 관한 자세한 내용은 필드 및 객체 참조 및 표현식 평가 단원을 참조하세요.

Expressions

표현식을 사용하여 관련 객체에서 값을 공유할 수 있습니다. 표현식은 런타임에 AWS Data Pipeline 웹 서비스에서 처리되므로 모든 표현식이 표현식 값으로 대체됩니다.

표현식은 "#{" 및 "}"로 구분됩니다. 법적 고지 문자열이 있는 파이프라인 정의 객체에서 표현식을 사용할 수 있습니다. 슬롯이 참조이거나 ID, NAME, TYPE, SPHERE 유형 중 하나일 경우에는 그 값이 축약되어 평가 및 사용되지 않습니다.

다음 표현식은 AWS Data Pipeline 함수 중 하나를 호출합니다. 자세한 내용은 표현식 평가 단원을 참조하십시오.

```
#{format(myDateTime,'YYYY-MM-dd hh:mm:ss')}
```

필드 및 객체 참조

표현식이 존재하는 현재 객체의 필드 또는 참조로 연결되는 다른 객체의 필드를 표현식에서 사용할 수 있습니다.

슬롯 형식은 생성 시간과 객체 생성 시간 순으로(예: `@S3BackupLocation_2018-01-31T11:05:33`) 구성됩니다.

또한 Amazon S3 백업 위치의 슬롯 ID처럼 파이프라인 정의에 지정된 해당 슬롯 ID를 참조할 수도 있습니다. 슬롯 ID를 참조하려면 `#{parent.@id}`를 사용합니다.

다음 예제에서는 `filePath` 필드가 동일 객체 내 `id` 필드를 참조하여 파일 이름을 형성합니다. `filePath`의 값은 `"s3://amzn-s3-demo-bucket/ExampleDataNode.csv"`로 평가됩니다.

```
{
  "id" : "ExampleDataNode",
  "type" : "S3DataNode",
  "schedule" : {"ref" : "ExampleSchedule"},
  "filePath" : "s3://amzn-s3-demo-bucket/#{parent.@id}.csv",
  "precondition" : {"ref" : "ExampleCondition"},
  "onFail" : {"ref" : "FailureNotify"}
}
```

참조로 연결된 다른 객체에 있는 필드를 사용하려면 node 키워드를 사용합니다. 이 키워드는 경보 및 사전 조건 객체와 함께 사용해야 합니다.

이전 예제에서 `SnsAlarm`의 표현식은 `Schedule`의 날짜와 시간을 나타낼 수 있는데, 그 이유는 `S3DataNode`가 둘 다 참조하기 때문입니다.

구체적으로 `FailureNotify`의 `message` 필드는 `ExampleSchedule`의 `@scheduledStartTime` 및 `@scheduledEndTime` 실행 시간 필드를 사용할 수 있는데 그 이유는 `ExampleDataNode`의 `onFail` 필드가 `FailureNotify`를 참조하고 그 `schedule` 필드가 `ExampleSchedule`을 참조하기 때문입니다.

```
{
```

```
    "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**
>
> 다른 시스템이나 작업에 의존하는 파이프라인 내 작업 같이 종속 요소가 있는 파이프라인을 생성할 수 있습니다. 파이프라인에 특정 리소스가 필요할 경우에는 데이터 노드와 작업에 연결한 사전 조건을 사용하여 파이프라인에 종속 요소를 추가합니다. 그러면 파이프라인의 복원력이 강화되고 디버그도 용이해집니다. 그리고 교차 파이프라인 문제는 해결이 어려우므로 가급적이면 단일 파이프라인 안에 종속 요소를 유지하는 것이 좋습니다.

중첩 표현식

AWS Data Pipeline 를 사용하면 값을 중첩하여 더 복잡한 표현식을 생성할 수 있습니다. 예를 들어, 시간을 계산하고(scheduledStartTime에서 30분 빼기) 결과를 포맷하여 파이프라인 정의에 사용하려면 활동에서 다음 표현식을 사용할 수 있을 것입니다.

```
#{format(minusMinutes(@scheduledStartTime,30),'YYYY-MM-dd hh:mm:ss')}
```

표현식이 SnsAlarm 또는 Precondition의 일부일 경우 node 접두사를 사용합니다.

```
#{format(minusMinutes(node.@scheduledStartTime,30),'YYYY-MM-dd hh:mm:ss')}
```

Lists

목록과 목록의 함수로 표현식을 평가할 수 있습니다. 예를 들어, `"myList":["one","two"]` 같이 목록이 정의되어 있다고 가정해보겠습니다. 이 목록이 표현식 `#{'this is ' + myList}`에 사용되면 `["this is one", "this is two"]`로 평가됩니다. 목록이 2개일 경우 이 둘을 평가할 때 결국 데이터 파이프라인에 의해 결합됩니다. 예를 들어, `myList1`이 `[1,2]`로 정의되고, `myList2`가 `[3,4]`으로 정의되면 표현식 `[#{myList1}, #{myList2}]`는 `[1,2,3,4]`로 평가됩니다.

노드 표현식

AWS Data Pipeline 는 파이프라인 구성 요소의 상위 객체에 PreCondition 대한 역참조를 위해 SnsAlarm 또는의 #{node.*} 표현식을 사용합니다. SnsAlarm와 PreCondition는 역참조가 없는 리소스나 활동에서 참조하므로 node가 참조 페이지를 참조하는 방법입니다. 예를 들어, 다음 파이프라인 정의는 장애 알림이 node를 사용하여 부모, 이 경우 ShellCommandActivity를 참조하고, 부모의 예약된 시작 및 종료 시간을 SnsAlarm 메시지에 포함시키는 방법을 보여 줍니다. ShellCommandActivity의 scheduledStartTime 참조는 node 접두사가 필요 없습니다. scheduledStartTime이 자체를 나타내기 때문입니다.

> **Note**
>
> 앞에 AT (@) 부호가 있는 필드는 실행 시간 필드입니다.

```
{
  "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 는 사용자 정의 필드에 대한 전이적 참조를 지원하지만 런타임 필드는 지원하지 않습니다. 전이 참조는 다른 파이프라인 구성요소에 의존하는 두 파이프라인 구성요소 사이에서 매개 역할을 하는 참조입니다. 다음 예제는 전이 사용자 정의 필드의 참조와 비-전이 실행 시간 필드의 참조를 보여주며, 둘 다 유효합니다. 자세한 내용은 사용자 정의 필드 단원을 참조하십시오.

```
{
  "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"
}
```

표현식 평가

AWS Data Pipeline 는 필드 값을 계산하는 데 사용할 수 있는 함수 집합을 제공합니다. 다음은 `makeDate` 함수를 사용하여 `Schedule` 객체의 `startDateTime` 필드를 `"2011-05-24T0:00:00"` GMT/UTC로 설정하는 예제입니다.

```
"startDateTime" : "makeDate(2011,5,24)"
```

수학 함수

다음 함수는 숫자 값에 사용할 수 있습니다.

함수	설명
+	덧셈. 예시: `#{1 + 2}` Result: 3
-	뺄셈.

함수	설명
	예시: `#{1 - 2}` Result: `-1`
*	곱셈. 예시: `#{1 * 2}` Result: 2
/	나눗셈. 정수 2개를 나누면 결과가 중간에 잘립니다. 예제: `#{1 / 2}`, 결과: `0` 예제: `#{1.0 / 2}`, 결과: `.5`
^	지수. 예시: `#{2 ^ 2}` Result: `4.0`

문자열 함수

다음 함수는 문자열 값에 사용할 수 있습니다.

함수	설명
+	연결. 비-문자열 값이 먼저 문자열로 변환됩니다. 예시: `#{"hel" + "lo"}` Result: `"hello"`

날짜 및 시간 함수

다음 함수는 DateTime 값에 사용할 수 있습니다. 예제에서 myDateTime 값은 May 24, 2011 @ 5:10 pm GMT입니다.

> **Note**
>
> 의 날짜/시간 형식 AWS Data Pipeline 은 Java 날짜 및 시간 클래스를 대체하는 Joda Time입니다. 자세한 내용은 Joda Time - Class DateTimeFormat을 참조하세요.

함수	설명
`int day(DateTime myDateTime)`	DateTime 값의 날짜를 정수로 가져옵니다. 예시: `#{day(myDateTime)}` Result: 24
`int dayOfYear(DateTime myDateTime)`	DateTime 값의 연중 날짜를 정수로 가져옵니다. 예시: `#{dayOfYear(myDateTime)}` Result: 144
`DateTime firstOfMonth(DateTime myDateTime)`	지정된 DateTime에서 월 시작일에 해당하는 DateTime 객체를 생성합니다. 예시: `#{firstOfMonth(myDateTime)}` Result: `"2011-05-01T17:10:00z"`

함수	설명
`String format(DateTime myDateTime,String format)`	지정된 형식의 문자열을 사용하여 지정된 DateTime의 변환 결과인 문자열 객체를 생성합니다. 예시: `#{format(myDateTime,'YYYY-MM-dd HH:mm:ss z')}` Result: `"2011-05-24T17:10:00 UTC"`
`int hour(DateTime myDateTime)`	DateTime 값의 시를 정수로 가져옵니다. 예시: `#{hour(myDateTime)}` Result: 17
`DateTime makeDate(int year,int month,int day)`	지정된 연, 월, 일의 자정에 해당하는 DateTime 객체를 UTC에 생성합니다. 예시: `#{makeDate(2011,5,24)}` Result: `"2011-05-24T0:00:00z"`

함수	설명
`DateTime makeDateTime(int year,int month,int day,int hour,int minute)`	지정된 연, 월, 일, 분에 해당하는 DateTime 객체를 UTC에 생성합니다. 예시: `#{makeDateTime(2011,5,24,14,21)}` Result: `"2011-05-24T14:21:00z"`
`DateTime midnight(DateTime myDateTime)`	지정된 DateTime을 기준으로 현재 자정에 해당하는 DateTime 객체를 생성합니다. `MyDateTime` 이 `2011-05-25T17:10:00z` 인 예제의 결과는 다음과 같습니다. 예시: `#{midnight(myDateTime)}` Result: `"2011-05-25T0:00:00z"`
`DateTime minusDays(DateTime myDateTime,int daysToSub)`	지정된 DateTime에서 지정된 일수를 뺀 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{minusDays(myDateTime,1)}` Result: `"2011-05-23T17:10:00z"`

함수	설명
`DateTime minusHours(DateTime myDateTime,int hoursToSub)`	지정된 DateTime에서 지정된 시 수를 뺀 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{minusHours(myDateTime,1)}` Result: `"2011-05-24T16:10:00z"`
`DateTime minusMinutes(DateTime myDateTime,int minutesToSub)`	지정된 DateTime에서 지정된 분 수를 뺀 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{minusMinutes(myDateTime,1)}` Result: `"2011-05-24T17:09:00z"`
`DateTime minusMonths(DateTime myDateTime,int monthsToSub)`	지정된 DateTime에서 지정된 월 수를 뺀 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{minusMonths(myDateTime,1)}` Result: `"2011-04-24T17:10:00z"`

함수	설명
`DateTime minusWeeks(DateTime myDateTime,int weeksToSub)`	지정된 DateTime에서 지정된 주 수를 뺀 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{minusWeeks(myDateTime,1)}` Result: `"2011-05-17T17:10:00z"`
`DateTime minusYears(DateTime myDateTime,int yearsToSub)`	지정된 DateTime에서 지정된 연 수를 뺀 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{minusYears(myDateTime,1)}` Result: `"2010-05-24T17:10:00z"`
`int minute(DateTime myDateTime)`	DateTime 값의 분을 정수로 가져옵니다. 예시: `#{minute(myDateTime)}` Result: `10`
`int month(DateTime myDateTime)`	DateTime 값의 월을 정수로 가져옵니다. 예시: `#{month(myDateTime)}` Result: 5

함수	설명
`DateTime plusDays(DateTime myDateTime,int daysToAdd)`	지정된 DateTime에 지정된 일수를 더한 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{plusDays(myDateTime,1)}` Result: `"2011-05-25T17:10:00z"`
`DateTime plusHours(DateTime myDateTime,int hoursToAdd)`	지정된 DateTime에 지정된 시 수를 더한 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{plusHours(myDateTime,1)}` Result: `"2011-05-24T18:10:00z"`
`DateTime plusMinutes(DateTime myDateTime,int minutesToAdd)`	지정된 DateTime에 지정된 분 수를 더한 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{plusMinutes(myDateTime,1)}` Result: `"2011-05-24 17:11:00z"`

함수	설명
`DateTime plusMonths(DateTime myDateTime,int monthsToAdd)`	지정된 DateTime에 지정된 월 수를 더한 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{plusMonths(myDateTime,1)}` Result: `"2011-06-24T17:10:00z"`
`DateTime plusWeeks(DateTime myDateTime,int weeksToAdd)`	지정된 DateTime에 지정된 주 수를 더한 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{plusWeeks(myDateTime,1)}` Result: `"2011-05-31T17:10:00z"`
`DateTime plusYears(DateTime myDateTime,int yearsToAdd)`	지정된 DateTime에 지정된 연 수를 더한 결과가 되는 DateTime 객체를 생성합니다. 예시: `#{plusYears(myDateTime,1)}` Result: `"2012-05-24T17:10:00z"`

함수	설명
`DateTime sunday(DateTime myDateTime)`	지정된 DateTime을 기준으로 이전 일요일에 해당하는 DateTime 객체를 생성합니다. 지정된 DateTime이 일요일인 경우에는 지정된 DateTime이 결과가 됩니다. 예시: `#{sunday(myDateTime)}` Result: `"2011-05-22 17:10:00 UTC"`
`int year(DateTime myDateTime)`	DateTime 값의 연을 정수로 가져옵니다. 예시: `#{year(myDateTime)}` Result: `2011`
`DateTime yesterday(DateTime myDateTime)`	지정된 DateTime을 기준으로 이전 일에 해당하는 DateTime 객체를 생성합니다. 결과는 minusDays(1)와 같습니다. 예시: `#{yesterday(myDateTime)}` Result: `"2011-05-23T17:10:00z"`

특수 문자

AWS Data Pipeline 는 다음 표와 같이 파이프라인 정의에 특별한 의미가 있는 특정 문자를 사용합니다.

특수 문자	설명	예제
@	실행 시간 필드입니다. 이 문자는 파이프라인이 실행될 때만 사용할 수 있는 필드의 필드 이름 접두사입니다.	@actualStartTime @failureReason @resourceStatus
#	표현식. 표현식은 "#{" 및 "}"로 구분되며 중괄호의 내용은 로 평가됩니다 AWS Data Pipeline. 자세한 내용은 Expressions 단원을 참조하십시오.	#{format(myDateTime,'YYYY-MM-dd hh:mm:ss')} s3://amzn-s3-demo-bucket/#{id}.csv
*	암호화 필드입니다. 이 문자는 AWS Data Pipeline 가 콘솔 또는 CLI와 AWS Data Pipeline 서비스 간에 전송 중인이 필드의 내용을 암호화해야 함을 나타내는 필드 이름 접두사입니다.	*암호

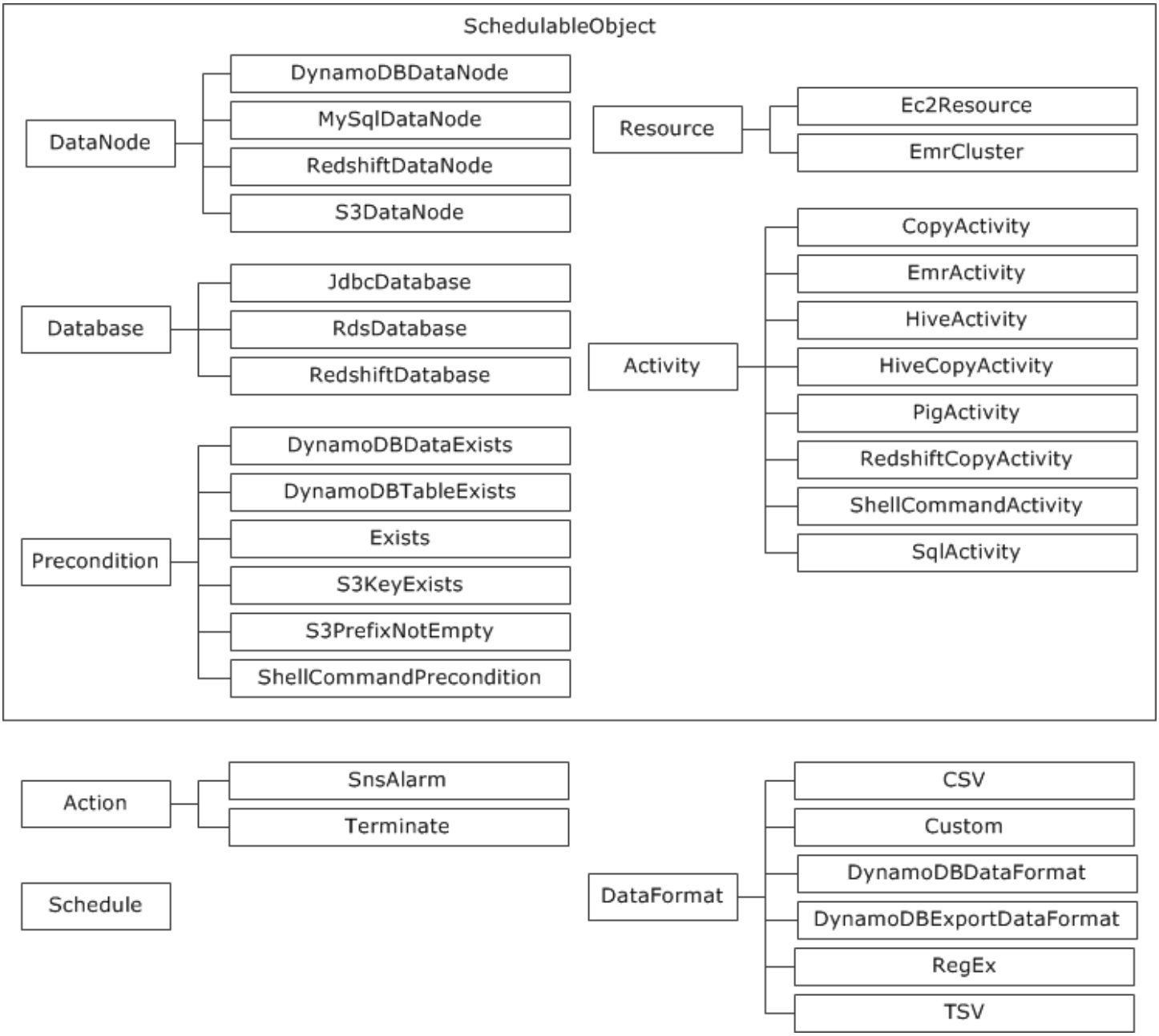
파이프라인 객체 참조

파이프라인 정의에 다음 파이프라인 객체 및 구성요소를 사용할 수 있습니다.

내용

> **Note**
>
> AWS Data Pipeline Java SDK를 사용하는 예제 애플리케이션은 GitHub의 Data Pipeline DynamoDB Export Java Sample을 참조하세요.

다음은의 객체 계층 구조입니다 AWS Data Pipeline.



데이터 노드

다음은 AWS Data Pipeline 데이터 노드 객체입니다.

Objects

- DynamoDBDataNode
- MySqlDataNode
- RedshiftDataNode

- S3DataNode
- SqlDataNode

DynamoDBDataNode

HiveActivity 또는 EMRActivity객체에 대한 입력으로서 지정된 DynamoDB를 사용하여 데이터 노드를 정의합니다.

> **Note**
>
> DynamoDBDataNode 객체는 Exists 사전 조건을 지원하지 않습니다.

예제

다음은 이 객체 유형의 예제입니다. 이 객체는 동일한 파이프라인 정의 파일에서 정의하려는 다른 두 객체를 참조합니다. CopyPeriod는 Schedule 객체이고 Ready는 사전 조건 객체입니다.

```
{
  "id" : "MyDynamoDBTable",
  "type" : "DynamoDBDataNode",
  "schedule" : { "ref" : "CopyPeriod" },
  "tableName" : "adEvents",
  "precondition" : { "ref" : "Ready" }
}
```

구문

필수 필드	설명	슬롯 유형
tableName	DynamoDB 테이블.	문자열

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 사용자가 다른 객체로 일정 참조를 지정해야 합니다.	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

액체 호출 필드	설명	슬롯 유형
	사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 일정을 참조하세요.	

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정한 경우 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
dataFormat	이 데이터 노드로 기술된` 데이터의 DataFormat입니다. 현재 HiveActivity와 HiveCopyActivity가 지원됩니다.	참조 객체, "dataFormat":{"ref":"myDynamoDBDataFormatId"}
dependsOn	실행 가능한 다른 객체의 종속성을 지정	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period

선택 필드	설명	슬롯 유형
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
readThroughputPercent	DynamoDB 프로비저닝 처리 속도를 테이블에 할당된 범위로 유지하도록 읽기 작업 속도를 설정합니다. 값은 0.1~1.0 사이의 배정밀도입니다.	배정밀도 실수

선택 필드	설명	슬롯 유형
리전	DynamoDB 테이블이 있는 리전의 코드입니다. 예를 들어 us-east-1입니다. Hive에서 DynamoDB 테이블 스테이징을 실행할 때 HiveActivity가 이것을 사용합니다.	열거
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
runsOn	활동 또는 명령을 실행할 전산 리소스입니다. Amazon EC2 인스턴스 또는 Amazon EMR 클러스터가 이에 해당합니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거

선택 필드	설명	슬롯 유형
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. workerGroup이 있을 때 runsOn 값을 제공하면 workerGroup이 무시됩니다.	문자열
writeThroughputPercent	DynamoDB 프로비저닝 처리 속도를 테이블에 할당된 범위로 유지하도록 쓰기 작업 속도를 설정합니다. 그 값은 0.1과 1.0 사이의 두 배입니다.	배정밀도 실수

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열

실행 시간 필드	설명	슬롯 유형
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

MySqlDataNode

MySQL를 사용하여 데이터 노드를 정의합니다.

> **Note**
>
> `MySqlDataNode` 유형은 사용되지 않습니다. 그 대신 SqlDataNode를 사용할 것을 권장합니다.

예제

다음은 이 객체 유형의 예제입니다. 이 객체는 동일한 파이프라인 정의 파일에서 정의하려는 다른 두 객체를 참조합니다. `CopyPeriod`는 `Schedule` 객체이고 `Ready`는 사전 조건 객체입니다.

```
{
  "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" }
}
```

구문

필수 필드	설명	슬롯 유형
테이블	MySQL 데이터베이스에 있는 테이블의 이름입니다.	문자열

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 사용자가 다른 객체로 일정 참조를 지정해야 합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
createTableSql	SQL은 테이블을 생성할 테이블 표현식을 생성합니다.	문자열
데이터베이스	데이터베이스의 이름입니다.	참조 객체. 예: "database":{"ref": "myDatabaseId"}
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref" :"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
insertQuery	테이블에 데이터를 삽입할 SQL 문입니다.	문자열
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}

선택 필드	설명	슬롯 유형
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
runsOn	활동 또는 명령을 실행할 전산 리소스입니다. Amazon EC2 인스턴스 또는 Amazon EMR 클러스터가 이에 해당합니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}

선택 필드	설명	슬롯 유형
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거
schemaName	테이블을 보유하는 스키마의 이름	문자열
selectQuery	테이블에서 데이터를 가져올 SQL 문장입니다.	문자열
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. workerGroup이 있을 때 runsOn 값을 제공하면 workerGroup이 무시됩니다.	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- S3DataNode

RedshiftDataNode

Amazon Redshift을 사용하여 데이터 노드를 정의합니다. `RedshiftDataNode`은(는) 파이프라인에서 사용하는 데이터베이스 내의 데이터 테이블과 같은 데이터 속성을 나타냅니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "id" : "MyRedshiftDataNode",
  "type" : "RedshiftDataNode",
  "database": { "ref": "MyRedshiftDatabase" },
  "tableName": "adEvents",
  "schedule": { "ref": "Hour" }
}
```

구문

필수 필드	설명	슬롯 유형
데이터베이스	테이블이 상주하는 데이터베이스.	참조 객체. 예: "database":{"ref": "myRedshiftDatabaseId"}
tableName	Amazon Redshift 테이블의 이름입니다. 이 테이블은 존재하지 않은 상태에서 createTableSql을 제공한 경우에 생성됩니다.	문자열

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 사용자가 다른 객체로 일정 참조를 지정해야 합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
createTableSql	데이터베이스에서 테이블을 생성하는 SQL 표현식입니다. 테이블을 생성해야 하는 스키마를 지정하는 것이 좋습니다. 예를 들어 CREATE TABLE mySchema.myTable(bestColumn varchar(25) primary key distkey, numberOfWins integer sortKey)은 tableName으로 지정된 테이블이 schemaName 필드로 지정된 스키마에 없는 경우 createTableSql 필드에서 스크립트를 AWS Data Pipeline 실행합니다. 예를 들어, schemaName을 mySchema로 지정했는데	문자열

선택 필드	설명	슬롯 유형
	createTableSql 필드에 mySchema가 없는 경우 테이블이 잘못된 스키마에 생성됩니다(기본적으로 PUBLIC에 생성됨). 이것은 AWS Data Pipeline이 CREATE TABLE 구문을 분석하지 않기 때문에 발생합니다.	
dependsOn	실행 가능한 다른 객체의 종속성을 지정	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수.	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}

선택 필드	설명	슬롯 유형
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
primaryKeys	`RedShiftCopyActivity` 의 대상 테이블로 primaryKey를 지정하지 않을 경우 mergeKey 역할을 하게 될 primaryKey를 사용하여 칼럼 목록을 지정할 수 있습니다. 그러나 Amazon Redshift 테이블에 기존 primaryKey가 정의되어 있는 경우, 이 설정이 기존 키를 다시 정의합니다.	문자열
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
runsOn	활동 또는 명령을 실행할 전산 리소스입니다. Amazon EC2 인스턴스 또는 Amazon EMR 클러스터가 이에 해당합니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}

선택 필드	설명	슬롯 유형
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거
schemaName	이 선택 필드는 Amazon Redshift 테이블용 스키마의 이름을 지정합니다. 지정되지 않으면 스키마 이름이 PUBLIC이 되며, 이것은 Amazon Redshift의 기본 스키마입니다. 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서를 참조하세요.	문자열
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. workerGroup이 있을 때 runsOn 값을 제공하면 workerGroup이 무시됩니다.	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances":

실행 시간 필드	설명	슬롯 유형
		{"ref":"myRunnable ObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnable ObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열

시스템 필드	설명	슬롯 유형
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

S3DataNode

Amazon S3를 사용하여 데이터 노드를 정의합니다. 기본적으로 S3DataNode는 서버 측 암호화를 사용합니다. 이를 비활성화하려면 s3EncryptionType을 NONE으로 설정합니다.

> **Note**
>
> S3DataNode를 CopyActivity에 입력으로 사용하는 경우 CSV 및 TSV 데이터 형식만 지원됩니다.

예제

다음은 이 객체 유형의 예제입니다. 이 객체는 동일한 파이프라인 정의 파일에서 정의하려는 다른 객체를 참조합니다. CopyPeriod는 Schedule 객체입니다.

```
{
  "id" : "OutputData",
  "type" : "S3DataNode",
  "schedule" : { "ref" : "CopyPeriod" },
  "filePath" : "s3://amzn-s3-demo-bucket/#{@scheduledStartTime}.csv"
}
```

구문

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 사용자가 다른 객체로 일정 참조를 지정해야 합니다. 사용자가 객체에서 일정을 명확히 설정하여(예:	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

액체 호출 필드	설명	슬롯 유형
	"schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
압축	S3DataNode가 기술하는 데이터의 압축 유형입니다. "none"은 압축되지 않은 것이며, "gzip"은 gzip 알고리즘으로 압축됩니다. 이 필드는 Amazon Redshift를 사용할 때와 S3DataNode를 CopyActivity와 함께 사용할 때만 지원됩니다.	열거
dataFormat	이 S3DataNode로 기술된` 데이터의 DataFormat입니다.	참조 객체. 예: "dataFormat":{"ref":"myDataFormatId"}
dependsOn	실행 가능한 다른 객체의 종속성을 지정	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}

선택 필드	설명	슬롯 유형
directoryPath	Amazon S3 디렉터리 경로, URI: s3://my-bucket/my-key-for-directory입니다. filePath 또는 directoryPath 값을 제공해야 합니다.	문자열
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
filePath	Amazon S3의 객체 경로 URI입니다(예: s3://my-bucket/my-key-for-file). filePath 또는 directoryPath 값을 제공해야 합니다. 이는 폴더와 파일 이름을 나타냅니다. 디렉토리에 여러 파일을 담을 수 있도록 directoryPath 값을 사용합니다.	문자열
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
manifestFilePath	Amazon Redshift에서 지원하는 형식의 매니페스트 파일에 대한 Amazon S3 경로입니다. 매니페스트 파일을 AWS Data Pipeline 사용하여 지정된 Amazon S3 파일을 테이블에 복사합니다. 이 필드는 RedShiftCopyActivity가 S3DataNode를 참조할 때만 유효합니다.	문자열
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}

선택 필드	설명	슬롯 유형
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
runsOn	활동 또는 명령을 실행할 전산 리소스입니다. Amazon EC2 인스턴스 또는 Amazon EMR 클러스터가 이에 해당합니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}
s3EncryptionType	Amazon S3 암호화 유형을 다시 정의합니다. 그 값은 SERVER_SIDE_ENCRYPTION 또는 NONE입니다. 기본적으로 서버 측 암호화가 활성화되어 있습니다.	열거

선택 필드	설명	슬롯 유형
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. workerGroup이 있을 때 runsOn 값을 제공하면 workerGroup이 무시됩니다.	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- MySqlDataNode

SqlDataNode

SQL을 사용하여 데이터 노드를 정의합니다.

예제

다음은 이 객체 유형의 예제입니다. 이 객체는 동일한 파이프라인 정의 파일에서 정의하려는 다른 두 객체를 참조합니다. CopyPeriod는 Schedule 객체이고 Ready는 사전 조건 객체입니다.

```
{
  "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" }
}
```

구문

필수 필드	설명	슬롯 유형
테이블	SQL 데이터베이스에 있는 테이블의 이름입니다.	문자열

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 사용자가 다른 객체로 일정 참조를 지정해야 합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

액체 호출 필드	설명	슬롯 유형
	기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
createTableSql	SQL은 테이블을 생성할 테이블 표현식을 생성합니다.	문자열
데이터베이스	데이터베이스의 이름입니다.	참조 객체. 예: "database":{"ref": "myDatabaseId"}
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref" :"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
insertQuery	테이블에 데이터를 삽입할 SQL 문입니다.	문자열
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period

선택 필드	설명	슬롯 유형
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period

선택 필드	설명	슬롯 유형
runsOn	활동 또는 명령을 실행할 전산 리소스입니다. Amazon EC2 인스턴스 또는 Amazon EMR 클러스터가 이에 해당합니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거
schemaName	테이블을 보유하는 스키마의 이름	문자열
selectQuery	테이블에서 데이터를 가져올 SQL 문장입니다.	문자열
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. workerGroup이 있을 때 runsOn 값을 제공하면 workerGroup이 무시됩니다.	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances":

실행 시간 필드	설명	슬롯 유형
		{"ref":"myRunnable ObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnable ObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열

시스템 필드	설명	슬롯 유형
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- S3DataNode

활동

다음은 AWS Data Pipeline 활동 객체입니다.

Objects

- CopyActivity
- EmrActivity
- HadoopActivity
- HiveActivity
- HiveCopyActivity
- PigActivity
- RedshiftCopyActivity
- ShellCommandActivity
- SqlActivity

CopyActivity

한 위치에서 다른 위치로의 데이터를 복사합니다. `CopyActivity`는 S3DataNode와 SqlDataNode를 입력 및 출력으로 지원하며 일반적으로 복사 작업은 레코드별로 수행됩니다. 그러나 `CopyActivity`은(는) 다음 조건이 모두 충족되면 고성능 Amazon S3에서 Amazon S3로의 복사를 제공합니다.

- 입력 및 출력은 S3DataNodes입니다.

- dataFormat 필드는 입력 및 출력이 동일합니다.

압축된 데이터 파일을 입력으로 제공하고 S3 데이터 노드에서 compression 필드를 사용하여 이를 나타내지 않으면 CopyActivity가 실패할 수도 있습니다. 이 경우 CopyActivity가 레코드 끝 문자를 제대로 감지하지 못하여 작업이 실패합니다. 또한 CopyActivity는 디렉터리에서 다른 디렉터리로의 복사와 디렉터리에 파일 복사를 지원하지만, 파일을 디렉터리에 복사하는 경우에는 레코드별로 복사합니다. 마지막으로 CopyActivity은(는) 멀티파트 Amazon S3 파일 복사를 지원하지 않습니다.

CopyActivity에는 해당 CSV 지원에 대한 특정 제한이 있습니다. CopyActivity의 입력으로 S3DataNode를 사용하는 경우에는 Amazon S3 입력 및 출력 필드에서 CSV 데이터 파일 형식의 Unix/Linux 변형만 사용할 수 있습니다. Unix/Linux 변형에는 다음 사항이 필요합니다.

- 구분 기호는 ","(쉼표) 문자여야 합니다.
- 레코드가 따옴표로 묶여있지 않습니다.
- 기본 이스케이프 문자가 ASCII 값 92(백슬래시)입니다.
- 레코드의 끝 식별자가 ASCII 값 10(또는 "\n")입니다.

일반적으로 Windows 기반 시스템에서는 다른 레코드 끝 문자 시퀀스를 사용하며 캐리지 리턴 및 줄 바꿈을 함께 사용합니다(ASCII 값 13, ASCII 값 10). CopyActivity가 레코드 끝을 제대로 감지할 수 있도록 추가 메커니즘(예: 입력 데이터를 수정하기 위한 사전 복사 스크립트)을 사용하여 이 차이를 수용해야 합니다. 그렇지 않으면 CopyActivity가 반복적으로 실패합니다.

CopyActivity를 사용하여 PostgreSQL RDS 객체에서 TSV 데이터 형식으로 내보낼 때 기본 NULL 문자는 \n입니다.

예제

다음은 이 객체 유형의 예제입니다. 이 객체는 동일한 파이프라인 정의 파일에서 정의하려는 다른 세 객체를 참조합니다. CopyPeriod는 Schedule 객체이고 InputData 및 OutputData는 데이터 노드 객체입니다.

```
{
  "id" : "S3ToS3Copy",
  "type" : "CopyActivity",
  "schedule" : { "ref" : "CopyPeriod" },
  "input" : { "ref" : "InputData" },
  "output" : { "ref" : "OutputData" },
  "runsOn" : { "ref" : "MyEc2Resource" }
```

```
}
```

구문

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 사용자가 다른 객체로 일정 참조를 지정해야 합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
runsOn	활동 또는 명령을 실행할 전산 리소스입니다. Amazon EC2 인스턴스 또는 Amazon EMR 클러스터가 이에 해당합니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. workerGroup이 있을 때 runsOn 값을 제공하면 workerGroup이 무시됩니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
입력	입력 데이터 소스입니다.	참조 객체, 예: "input":{"ref":"myDataNodeId"}
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}

선택 필드	설명	슬롯 유형
output	출력 데이터 소스입니다.	참조 객체, 예: "output":{"ref":"myDataNodeId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period

선택 필드	설명	슬롯 유형
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn":

실행 시간 필드	설명	슬롯 유형
		{"ref":"myRunnable ObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandActivity
- EmrActivity
- 를 사용하여 MySQL 데이터를 Amazon S3로 내보내기 AWS Data Pipeline

EmrActivity

EMR 클러스터를 실행합니다.

AWS Data Pipeline 는 Amazon EMR과 다른 형식의 단계를 사용합니다. 예를 들어는 EmrActivity 단계 필드의 JAR 이름 뒤에 쉼표로 구분된 인수를 AWS Data Pipeline 사용합니다. 다음 예제에서는 해당 AWS Data Pipeline 형식의 단계에 이어 Amazon EMR 형식의 단계를 보여줍니다.

```
s3://amzn-s3-demo-bucket/MyWork.jar arg1 arg2 arg3
```

```
"s3://amzn-s3-demo-bucket/MyWork.jar,arg1,arg2,arg3"
```

예제

다음은 이 객체 유형의 예제입니다. 이 예제에서는 이전 버전의 Amazon EMR을 사용합니다. 사용 중인 Amazon EMR 클러스터 버전과 비교하여 이 예제의 정확성을 확인합니다.

이 객체는 동일한 파이프라인 정의 파일에서 정의하려는 다른 세 객체를 참조합니다. MyEmrCluster는 EmrCluster 객체이고 MyS3Input 및 MyS3Output는 S3DataNode 객체입니다.

> **Note**
>
> 이 예제에서는 step 필드를 Pig 스크립트, Hadoop 스트리밍 클러스터, 파라미터를 포함하는 사용자 지정 JAR 등 원하는 클러스터 문자열로 바꿉니다.

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**
>
> 한 단계에서 애플리케이션에 인수를 전달하려면 다음 예제와 같이 스크립트의 경로에 리전을 지정해야 합니다. 또한 전달하는 인수를 이스케이프해야 합니다. 예를 들어, `script-runner.jar`를 사용하여 셸 스크립트를 실행하고 인수를 스크립트에 전달하려면 이를 구분하는 쉼표를 이스케이프해야 합니다. 다음은 이를 수행하는 방법을 설명하는 단계 슬롯입니다.
>
> ```
> "step" : "s3://eu-west-1.elasticmapreduce/libs/script-runner/script-
> runner.jar,s3://datapipeline/echo.sh,a\\\\,b\\\\,c"
> ```
>
> 이 단계에서는 `script-runner.jar`를 사용하여 `echo.sh` 셸 스크립트를 실행하고 a, b 및 c를 단일 인수로 스크립트에 전달합니다. 첫 번째 이스케이프 문자는 결과 인수에서 제거되므로 다시 이스케이프해야 할 수도 있습니다. 예를 들어 JSON에서 `File\.gz`를 인수로 가졌다면 `File\\\\.gz`를 사용하여 이스케이프할 수 있습니다. 그러나 첫 번째 이스케이프가 삭제되었기 때문에 `File\\\\\\\\.gz` 를 사용해야 합니다.

구문

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 다른 객체로 일정 참조를 지정합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: `"schedule": {"ref": "DefaultSchedule"}` 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

액체 호출 필드	설명	슬롯 유형
	datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
runsOn	이 작업이 실행될 Amazon EMR 클러스터입니다.	참조 객체. 예: "runsOn":{"ref":"myEmrClusterId"}
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. `workerGroup` 이 있을 때 `runsOn` 값을 제공하면 `workerGroup` 이 무시됩니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
입력	입력 데이터의 위치입니다.	참조 객체. 예: "input":{"ref":"myDataNodeId"}

선택 필드	설명	슬롯 유형
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수.	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체가 성공하면 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
output	출력 데이터의 위치입니다.	참조 객체. 예: "output":{"ref":"myDataNodeId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 Amazon S3 URI(예: 's3://BucketName/Prefix/')입니다.	문자열
postStepCommand	모든 단계가 끝난 후에 실행될 셸 스크립트입니다. 스크립트를 여러 개(최대 255개) 지정하려면 `postStepCommand` 필드를 여러 개 추가합니다.	문자열

선택 필드	설명	슬롯 유형
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
preStepCommand	임의 단계가 실행되기 전에 실행될 셸 스크립트입니다. 스크립트를 여러 개(최대 255개) 지정하려면 `preStepCommand` 필드를 여러 개 추가합니다.	문자열
reportProgressTimeout	원격 작업에서 `reportProgress` 를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
resizeClusterBeforeRunning	이 활동을 수행하기 전에 입력 또는 출력으로 지정된 DynamoDB 테이블이 포함되도록 클러스터 크기를 조정합니다. **Note** 가를 입력 또는 출력 데이터 노드`DynamoDBDataNode` 로 `EmrActivity` 사용하고 를 `resizeClusterBeforeRunning` 로 설정하면가 `m3.xlarge` 인스턴스 유형을 사용하여 TRUE AWS Data Pipeline 시작합니다. 그러면 해당 인스턴스 유형 선택을 `m3.xlarge` 가 덮어써서 월 요금이 증가할 수 있습니다.	부울
resizeClusterMaxInstances	크기 조정 알고리즘으로 요청할 수 있는 인스턴스의 최대 수에 대한 제한입니다.	Integer

선택 필드	설명	슬롯 유형
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 값은 cron, ondemand 및 timeseries 입니다. timeseries 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정됩니다. cron 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. ondemand 일정을 사용하면 활성화될 때마다 한 번씩 파이프라인을 실행할 수 있습니다. 그러므로 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. ondemand 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 scheduleType 이어야 합니다. ondemand 파이프라인을 사용하려면 후속 실행마다 ActivatePipeline 작업을 호출합니다.	열거
단계	클러스터가 실행할 하나 이상의 단계입니다. 최대 255개까지 여러 단계를 지정하려면 여러 단계 필드를 추가합니다. JAR 이름 뒤에 쉼표로 구분된 인수를 사용합니다. 예: "s3://amzn-s3-demo-bucket/MyWork.jar,arg1,arg2,arg3 "	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}

실행 시간 필드	설명	슬롯 유형
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 Amazon EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 `errorId`입니다.	문자열
errorMessage	이 객체가 실패한 경우의 `errorMessage` 입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열

시스템 필드	설명	슬롯 유형
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandActivity
- CopyActivity
- EmrCluster

HadoopActivity

클러스터에서 MapReduce 작업을 실행합니다. 클러스터는 AWS Data Pipeline 으로 관리되는 EMR 클러스터이거나 TaskRunner를 사용하는 경우에는 다른 리소스일 수 있습니다. 병렬 방식으로 작업을 실행하는 경우 HadoopActivity를 사용하세요. 이를 통해 Hadoop 1에서 MapReduce 리소스 매니저 또는 YARN 프레임워크의 예약 리소스를 사용할 수 있습니다. Amazon EMR 단계 작업을 사용하여 순차적으로 작업을 진행하려는 경우에도 EmrActivity을(를) 여전히 사용할 수 있습니다.

예제

에서 관리하는 EMR 클러스터를 사용한 HadoopActivity AWS Data Pipeline

다음 HadoopActivity 객체는 EmrCluster 리소스를 사용하여 프로그램을 실행합니다.

```
{
  "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”
 }
```

다음은 Hadoop 2 기반 AMI용으로 YARN에서 FairScheduler 및 대기열을 구성하는 해당 *MyEmrCluster*입니다.

```
{
  "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”]
}
```

Hadoop 1에서 FairScheduler를 구성하는 데 사용하는 EmrCluster입니다.

```
{
      "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"
          }
```

다음 EmrCluster는 Hadoop 2 기반 AMI의 CapacityScheduler를 구성합니다.

```
{
```

```
      "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"
    }
```

기존 EMR 클러스터를 사용한 HadoopActivity

이 예제에서는 workergroups 및 TaskRunner를 사용하여 기존 EMR 클러스터에서 프로그램을 실행합니다. 파이프라인 정의는 HadoopActivity를 사용하여 다음을 수행합니다.

- *myWorkerGroup* 리소스에서만 MapReduce 프로그램을 실행합니다. 작업자 그룹에 대한 정보는 Task Runner를 사용하여 기존 리소스에서 작업 실행를 참조하세요.
- preActivityTaskConfig 및 postActivityTaskConfig를 실행합니다.

```
{
  "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"
      }
    }
```

```
  ]
}
```

구문

필수 필드	설명	슬롯 유형
jarUri	Amazon S3 또는 클러스터의 로컬 파일 시스템에서 HadoopActivity와 함께 실행될 JAR의 위치입니다.	문자열

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 사용자가 다른 객체로 일정 참조를 지정해야 합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
runsOn	이 작업이 실행될 EMR 클러스터입니다.	참조 객체. 예: "runsOn":{"ref":"myEmrClusterId"}
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. workerGroup이 있을 때 runsOn 값을 제공하면 workerGroup이 무시됩니다.	문자열

선택 필드	설명	슬롯 유형
인수	JAR에 전달할 인수입니다.	문자열
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
hadoopQueue	활동이 제출될 Hadoop 스케줄러 대기열 이름입니다.	문자열
입력	입력 데이터의 위치입니다.	참조 객체, 예: "input":{"ref":"myDataNodeId"}
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period

선택 필드	설명	슬롯 유형
mainClass	HadoopActivity과 함께 실행하는 JAR의 주 클래스입니다.	문자열
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
output	출력 데이터의 위치입니다.	참조 객체, 예: "output":{"ref":"myDataNodeId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
postActivityTaskConfig	실행할 사후 활동 구성 스크립트입니다. 이것은 Amazon S3의 셸 스크립트 URI와 인수 목록으로 구성됩니다.	참조 객체. 예: "postActivityTaskConfig":{"ref":"myShellScriptConfigId"}

선택 필드	설명	슬롯 유형
preActivityTaskConfig	실행할 사전 활동 구성 스크립트입니다. 이것은 Amazon S3의 셸 스크립트 URI와 인수 목록으로 구성됩니다.	참조 객체. 예: "preActivityTaskConfig":{"ref":"myShellScriptConfigId"}
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열

시스템 필드	설명	슬롯 유형
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandActivity
- CopyActivity
- EmrCluster

HiveActivity

EMR 클러스터에서 Hive 쿼리를 실행합니다. `HiveActivity`을(를) 사용하면 Amazon EMR 활동을 보다 쉽게 설정하고 Amazon S3 또는 Amazon RDS에서 제공되는 입력 데이터를 기반으로 Hive 테이블을 자동으로 생성할 수 있습니다. 소스 데이터에서 실행할 HiveQL만 지정하면 됩니다.는 `HiveActivity` 객체의 입력 필드를 기반으로 `${input1}`, `${input2}`등으로 Hive 테이블을 AWS Data Pipeline 자동으로 생성합니다.

Amazon S3 입력의 경우 `dataFormat`필드를 사용하여 Hive 열 이름을 생성합니다.

MySQL(Amazon RDS) 입력의 경우 SQL 쿼리의 열 이름을 사용하여 Hive 열 이름을 생성합니다.

> **Note**
>
> 이 활동은 Hive CSV Serde를 사용합니다.

예제

다음은 이 객체 유형의 예제입니다. 이 객체는 동일한 파이프라인 정의 파일에서 정의하는 다른 세 객체를 참조합니다. `MySchedule`는 `Schedule` 객체이고 `MyS3Input` 및 `MyS3Output`는 데이터 노드 객체입니다.

```
{
```

```
  "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" }
}
```

구문

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 다른 객체로 일정 참조를 지정합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
hiveScript	실행할 Hive 스크립트입니다.	문자열
scriptUri	실행할 Hive 스크립트의 위치입니다(예: s3://scriptLocation).	문자열

필수 그룹	설명	슬롯 유형
runsOn	이 `HiveActivity` 가 실행될 EMR 클러스터입니다.	참조 객체. 예: "runsOn":{"ref":"myEmrClusterId"}
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. `workerGroup` 이 있을 때 `runsOn` 값을 제공하면 `workerGroup` 이 무시됩니다.	문자열
입력	입력 데이터 소스입니다.	참조 객체. 예: "input":{"ref":"myDataNodeId"}
output	출력 데이터 소스입니다.	참조 객체. 예: "output":{"ref":"myDataNodeId"}

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
hadoopQueue	작업이 제출될 Hadoop 스케줄러 대기열 이름입니다.	문자열

선택 필드	설명	슬롯 유형
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수.	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
postActivityTaskConfig	실행할 사후 활동 구성 스크립트입니다. 이것은 Amazon S3의 셸 스크립트 URI와 인수 목록으로 구성됩니다.	참조 객체. 예: "postActivityTaskConfig":{"ref":"myShellScriptConfigId"}

선택 필드	설명	슬롯 유형
preActivityTaskConfig	실행할 사전 활동 구성 스크립트입니다. 이것은 Amazon S3의 셸 스크립트 URI와 인수 목록으로 구성됩니다.	참조 객체. 예: "preActivityTaskConfig":{"ref":"myShellScriptConfigId"}
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
reportProgressTimeout	원격 작업에서 `reportProgress` 를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
resizeClusterBeforeRunning	이 활동을 수행하기 전에 입력 또는 출력으로 지정된 DynamoDB 데이터 노드가 포함되도록 클러스터 크기를 조정합니다. **Note** 활동이를 입력 또는 출력 데이터 노드`DynamoDBDataNode` 로 사용하고를 `resizeClusterBeforeRunning` 로 설정하면가 `m3.xlarge` 인스턴스 유형을 사용하여 TRUE AWS Data Pipeline 시작합니다. 그러면 해당 인스턴스 유형 선택을 `m3.xlarge` 가 덮어써서 월 요금이 증가할 수 있습니다.	부울
resizeClusterMaxInstances	크기 조정 알고리즘으로 요청할 수 있는 인스턴스의 최대 수에 대한 제한입니다.	Integer
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period

선택 필드	설명	슬롯 유형
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거
scriptVariable	스크립트 실행 중에 Hive로 전달할 Amazon EMR의 스크립트 변수를 지정합니다. 예를 들어 `SAMPLE=s3://elasticmapreduce/samples/hive-ads` 및 `FILTER_DATE=#{format(@scheduledStartTime,'YYYY-MM-dd')}%` 예제 스크립트 변수는 SAMPLE 및 FILTER_DATE 변수를 Hive에 전달합니다. 이 필드는 여러 개의 값을 허용하며 `script` 필드 및 `scriptUri` 필드와 연동합니다. 또한 `scriptVariable` 은 스테이지가 `true`로 설정되든 `false`로 설정되든 상관없이 작동합니다. 이 필드는 특히 AWS Data Pipeline 표현식과 함수를 사용하여 Hive로 동적 값을 전송할 때 유용합니다.	문자열

선택 필드	설명	슬롯 유형
stage	스크립트 실행 이전 또는 이후의 스테이징 활성화 여부를 결정합니다. Hive 11에는 허용되지 않으므로 Amazon EMR AMI 버전 3.2.0 이상을 사용해야 합니다.	부울

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 Amazon EMR 단계 로그.	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandActivity
- EmrActivity

HiveCopyActivity

EMR 클러스터에서 Hive 쿼리를 실행합니다. `HiveCopyActivity`을(를) 사용하면 DynamoDB 테이블 사이에서 데이터를 보다 쉽게 복사할 수 있습니다. `HiveCopyActivity`은(는) HiveQL 문을 사용하여 열 및 행 수준에서 DynamoDB의 입력 데이터를 필터링합니다.

예제

다음 예제는 데이터를 필터링하는 동안 타임스탬프를 기반으로 `HiveCopyActivity` 및 `DynamoDBExportDataFormat`을 사용하여 한 `DynamoDBDataNode`에서 다른 로 데이터를 복사하는 방법을 보여줍니다.

```
{
  "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"
    }
  ]
}
```

구문

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 사용자가 다른 객체로 일정 참조를 지정해야 합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
runsOn	실행할 클러스터를 지정합니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. `workerGroup` 이 있을 때 `runsOn` 값을 제공하면 `workerGroup` 이 무시됩니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고한 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
filterSql	복사할 DynamoDB 또는 Amazon S3 데이터의 하위 집합을 필터링하는 Hive SQL 문장의 일부입니다. 가 자동으로 AWS Data Pipeline 추가하기 때문에 필터에는 조건자만 포함되어야 하며 `WHERE` 절로 시작해서는 안 됩니다.	문자열
입력	입력 데이터 소스입니다. 이 값은 `S3DataNode` 또는 `DynamoDBDataNode` 여야 합니다. `DynamoDBNode` 를 사용하는 경우 `DynamoDBExportDataFormat` 을 지정합니다.	참조 객체, 예: "input":{"ref":"myDataNodeId"}

선택 필드	설명	슬롯 유형
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수.	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
output	출력 데이터 소스입니다. 입력이 S3DataNode 인 경우 이 값은 DynamoDBDataNode 여야 합니다. 그 외의 경우에는 S3DataNode 또는 DynamoDBDataNode 가 될 수 있습니다. DynamoDBNode 를 사용하는 경우 DynamoDBExportDataFormat 을 지정합니다.	참조 객체, 예: "output":{"ref":"myDataNodeId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 Amazon S3 URI(예: 's3://BucketName/Key/')입니다.	문자열

선택 필드	설명	슬롯 유형
postActivityTaskConfig	실행할 사후 활동 구성 스크립트입니다. 이것은 Amazon S3의 셸 스크립트 URI와 인수 목록으로 구성됩니다.	참조 객체. 예: "postActivityTaskConfig":{"ref":"myShellScriptConfigId"}
preActivityTaskConfig	실행할 사전 활동 구성 스크립트입니다. 이것은 Amazon S3의 셸 스크립트 URI와 인수 목록으로 구성됩니다.	참조 객체. 예: "preActivityTaskConfig":{"ref":"myShellScriptConfigId"}
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
reportProgressTimeout	원격 작업에서 `reportProgress` 를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
resizeClusterBeforeRunning	이 활동을 수행하기 전에 입력 또는 출력으로 지정된 DynamoDB 데이터 노드가 포함되도록 클러스터 크기를 조정합니다. **Note** 활동이를 입력 또는 출력 데이터 노드`DynamoDBDataNode` 로 사용하고를 `resizeClusterBeforeRunning` 로 설정하면가 `m3.xlarge` 인스턴스 유형을 사용하여 TRUE AWS Data Pipeline 시작합니다. 그러면 해당 인스턴스 유형 선택을 `m3.xlarge` 가 덮어써서 월 요금이 증가할 수 있습니다.	부울

선택 필드	설명	슬롯 유형
resizeClusterMaxInstances	크기 조정 알고리즘으로 요청할 수 있는 인스턴스의 최대 수에 대한 제한입니다.	Integer
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 Amazon EMR 단계 로그.	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Object를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandActivity
- EmrActivity

PigActivity

PigActivity는 ShellCommandActivity 또는를 사용할 필요 AWS Data Pipeline 없이의 Pig 스크립트에 대한 기본 지원을 제공합니다EmrActivity. 그 밖에도 PigActivity는 데이터 스테이징을 지원합니다. 스테이지 필드가 true로 설정되면 AWS Data Pipeline 에서는 사용자의 추가 코드 없이 입력 데이터가 Pig의 스키마로 스테이징됩니다.

예제

다음 예제 파이프라인은 PigActivity를 사용하는 방법을 보여줍니다. 이 예제 파이프라인은 다음 단계를 수행합니다.

- MyPigActivity1은 Amazon S3에서 데이터를 로드하고 몇 개의 데이터 열을 선택하여 Amazon S3에 업로드하는 Pig 스크립트를 실행합니다.
- MyPigActivity2는 첫 번째 출력을 로드하고 데이터 열 몇 개와 데이터 행 3개를 선택하여 이를 두 번째 출력으로 Amazon S3에 업로드합니다.
- MyPigActivity3은 두 번째 출력 데이터를 로드하고 두 개의 데이터 행과 "다섯 번째"라는 이름의 열만을 Amazon RDS에 삽입합니다.
- MyPigActivity4는 Amazon RDS 데이터를 로드하고 데이터의 첫 번째 행을 선택하여 Amazon S3에 업로드합니다.

```
{
  "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"
    }
  ]
}
```

pigTestScript.q의 내용은 다음과 같습니다.

```
B = LIMIT ${input1} $three; ${output1} = FOREACH B GENERATE $column1, $column2, Third,
 Fourth, Fifth, Sixth, Seventh, Eighth;
```

구문

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 사용자가 다른 객체로 일정 참조를 지정해야 합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
스크립트	실행할 Pig 스크립트입니다.	문자열
scriptUri	실행할 Pig 스크립트의 위치입니다(예: s3://scriptLocation).	문자열

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
runsOn	이 PigActivity가 실행될 EMR 클러스터입니다.	참조 객체. 예: "runsOn":{"ref":"myEmrClusterId"}
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. `workerGroup` 이 있을 때 `runsOn` 값을 제공하면 `workerGroup` 이 무시됩니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고한 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거

선택 필드	설명	슬롯 유형
입력	입력 데이터 소스입니다.	참조 객체. 예: "input":{"ref":"myDataNodeId"}
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수.	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
output	출력 데이터 소스입니다.	참조 객체. 예: "output":{"ref":"myDataNodeId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 Amazon S3 URI(예: 's3://BucketName/Key/')입니다.	문자열

선택 필드	설명	슬롯 유형
postActivityTaskConfig	실행할 사후 활동 구성 스크립트입니다. 이것은 Amazon S33의 쉘 스크립트 URI와 인수 목록으로 구성됩니다.	참조 객체. 예: "postActivityTaskConfig":{"ref":"myShellScriptConfigId"}
preActivityTaskConfig	실행할 사전 활동 구성 스크립트입니다. 이것은 Amazon S3의 셸 스크립트 URI와 인수 목록으로 구성됩니다.	참조 객체. 예: "preActivityTaskConfig":{"ref":"myShellScriptConfigId"}
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
reportProgressTimeout	원격 작업에서 `reportProgress` 를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
resizeClusterBeforeRunning	이 활동을 수행하기 전에 입력 또는 출력으로 지정된 DynamoDB 데이터 노드가 포함되도록 클러스터 크기를 조정합니다. **Note** 활동이를 입력 또는 출력 데이터 노드`DynamoDBDataNode` 로 사용하고를 `resizeClusterBeforeRunning` 로 설정하면가 `m3.xlarge` 인스턴스 유형을 사용하여 TRUE AWS Data Pipeline 시작합니다. 그러면 해당 인스턴스 유형 선택을 `m3.xlarge` 가 덮어써서 월 요금이 증가할 수 있습니다.	부울

선택 필드	설명	슬롯 유형
resizeClusterMaxInstances	크기 조정 알고리즘으로 요청할 수 있는 인스턴스의 최대 수에 대한 제한입니다.	Integer
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 시계열 스타일 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정되고 Cron 스타일 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. 온디맨드 일정을 사용하면 파이프라인을 활성화될 때 한 번씩 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 schuleType이어야 합니다. 온디맨드 파이프라인을 사용하려면 이후 실행할 때마다 ActivatePipeline 작업을 호출하면 됩니다. 값은 cron, ondemand 및 timeseries입니다.	열거
scriptVariable	Pig 스크립트에 전달할 인수입니다. scriptVariable을 script 또는 scriptUri와 함께 사용할 수 있습니다.	문자열
stage	스테이징 활성화 여부를 결정하며, Pig 스크립트를 사용하여 ${INPUT1} 및 ${OUTPUT1} 같은 스테이징 데이터 테이블에 액세스할 수 있습니다.	부울

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 Amazon EMR 단계 로그.	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열

시스템 필드	설명	슬롯 유형
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandActivity
- EmrActivity

RedshiftCopyActivity

DynamoDB 또는 Amazon S3에서 Amazon Redshift로 데이터를 복사합니다. 데이터를 기존 테이블로 로드하거나 기존 테이블에 데이터를 쉽게 병합할 수 있습니다.

다음은 `RedshiftCopyActivity`를 사용할 사용 사례의 개요입니다.

1. 먼저 AWS Data Pipeline 를 사용하여 Amazon S3에서 데이터를 스테이징합니다.
2. `RedshiftCopyActivity`을(를) 사용하여 Amazon RDS 및 Amazon EMR에서 Amazon Redshift로 데이터를 이동합니다.

   이렇게 하면 Amazon Redshift에 데이터를 로드하여 데이터를 분석할 수 있습니다.
3. SqlActivity을(를) 사용하여 Amazon Redshift에 로드한 데이터에 대해 SQL 쿼리를 수행합니다.

또한 `RedshiftCopyActivity`는 매니페스트 파일을 지원하므로 `S3DataNode` 작업을 할 수 있습니다. 자세한 내용은 S3DataNode 단원을 참조하십시오.

예제

다음은 이 객체 유형의 예제입니다.

형식 변환을 보장하기 위해, 이 예제에서는 EMPTYASNULL 및 `commandOptions`의 IGNOREBLANKLINES 특수 변환 파라미터를 사용합니다. 자세한 내용은 Amazon Redshift 데이터베이스 개발자 가이드의 데이터 변환 파라미터를 참조하세요.

```
{
  "id" : "S3ToRedshiftCopyActivity",
  "type" : "RedshiftCopyActivity",
  "input" : { "ref": "MyS3DataNode" },
  "output" : { "ref": "MyRedshiftDataNode" },
  "insertMode" : "KEEP_EXISTING",
  "schedule" : { "ref": "Hour" },
  "runsOn" : { "ref": "MyEc2Resource" },
  "commandOptions": ["EMPTYASNULL", "IGNOREBLANKLINES"]
}
```

다음 예제 파이프라인 정의는 APPEND 삽입 모드를 사용하는 활동을 보여줍니다.

```
{
  "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"
      }
    }
  ]
}
```

APPEND 작업은 기본 키 또는 정렬 키에 관계 없이 테이블에 항목을 추가합니다. 예를 들어 다음 테이블이 있는 경우 동일한 ID와 사용자 값으로 레코드를 추가할 수 있습니다.

```
ID(PK)     USER
1          aaa
2          bbb
```

동일한 ID와 사용자 값으로 레코드를 추가할 수 있습니다.

```
ID(PK)     USER
1          aaa
2          bbb
1          aaa
```

> **Note**
>
> APPEND 작업이 중단되었다가 재시도되는 경우, 결과 재실행 파이프라인이 처음부터 잠재적으로 추가됩니다. 이로 인해 더 많은 중복이 발생할 수 있으므로 특히 행 수를 계산하는 논리가 있는 경우에는 이 동작을 숙지하고 있어야 합니다.

자습서는 를 사용하여 Amazon Redshift에 데이터 복사 AWS Data Pipeline을 참조하세요.

구문

필수 필드	설명	슬롯 유형
insertMode	로드할 데이터의 행과 겹치는 대상 테이블의 기존 데이터로 AWS Data Pipeline 수행하는 작업을 결정합니다. 유효 값은 KEEP_EXISTING , OVERWRITE_EXISTING , TRUNCATE 및 APPEND입니다. KEEP_EXISTING 은 테이블에 새 행을 추가하고, 기존 행은 그대로 둡니다. KEEP_EXISTING 과 OVERWRITE_EXISTING 은 기본 키, 정렬 키 및 배포 키를 사용하여 어느 수신 행이 기존 행과 일치하는지 식별합니다. Amazon Redshift 데이터베이스 개발자 안내서의 새 데이터 업데이트 및 삽입을 참조하세요. TRUNCATE는 새 데이터를 기록하기 전에 대상 테이블의 데이터를 모두 삭제합니다. APPEND는 Redshift 테이블 끝에 모든 레코드를 추가합니다. APPEND는 기본, 배포 키 또는 정렬 키를 사용할 필요가 없어 중복 가능성이 있는 항목이 추가될 수 있습니다.	열거

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 다른 객체로 일정 참조를 지정합니다.	"schedule":{"ref":"mySchedu leId"} 와 같은 참조 객체

액체 호출 필드	설명	슬롯 유형
	대부분의 경우, 모든 객체가 일정을 상속할 수 있도록 해당 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 예를 들어 `"schedule": {"ref": "DefaultSchedule"}` 을 지정하여 객체에서 일정을 명시적으로 설정할 수 있습니다. 파이프라인의 마스터 일정에 중첩된 일정이 포함된 경우, 일정 참조가 있는 부모 객체를 생성합니다. 선택형 일정 구성 예제에 대한 자세한 내용은 일정을 참조하세요.	

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
runsOn	활동 또는 명령을 실행할 전산 리소스입니다. Amazon EC2 인스턴스 또는 Amazon EMR 클러스터가 이에 해당합니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. `workerGroup` 이 있을 때 `runsOn`값을 제공하면 workerGroup이 무시됩니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period

선택 필드	설명	슬롯 유형
commandOptions	COPY 작업 중에 Amazon Redshift 데이터 노드로 전달할 파라미터를 가져옵니다. 파라미터에 대한 내용을 알아보시려면 Amazon Redshift 데이터베이스 개발자 안내서의 COPY를 참조하세요. COPY는 테이블을 로드할 때 문자열을 대상 열의 데이터 형식으로 묵시적으로 변환하려 시도합니다. 자동으로 이루어지는 기본 데이터 변환 외에도 오류가 수신되거나 다른 변환이 필요한 경우, 추가 변환 파라미터를 지정할 수 있습니다. 자세한 내용은 Amazon Redshift 데이터베이스 개발자 가이드의 데이터 변환 파라미터를 참조하세요. 데이터 형식이 입력 또는 출력 데이터 노드와 연결된 경우에는 제공된 파라미터는 무시됩니다. 복사 작업에서는 먼저 COPY를 사용하여 스테이징 테이블에 데이터를 삽입한 다음 INSERT 명령을 사용하여 스테이징 테이블의 데이터를 대상 테이블로 복사하기 때문에 테이블 자동 압축을 활성화할 수 있는 COPY 명령과 같은 일부 COPY 파라미터는 적용되지 않습니다. 압축이 필요할 경우에는 열 인코딩 세부 정보를 CREATE TABLE 문에 추가합니다. 또한 Amazon Redshift 클러스터에서 데이터를 언로드하고 Amazon S3 에서 파일을 생성해야 하는 경우, RedshiftCopyActivity 은(는) Amazon Redshift의 UNLOAD 작업에 의존합니다. 복사 및 언로드 중에 성능을 개선하려면 UNLOAD 명령에서 PARALLEL OFF 파라미터를 지정합니다. 자세한 내용은 Amazon Redshift 데	문자열

선택 필드	설명	슬롯 유형
	이터베이스 개발자 안내서에서 UNLOAD를 참조하세요.	
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체: `"dependsOn": {"ref":"myActivityId"}`
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
입력	입력 데이터 노드입니다. Amazon S3, DynamoDB 또는 Amazon Redshift가 데이터 소스가 될 수 있습니다.	참조 객체: `"input":{"ref":"myDataNodeId"}`
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체: `"onFail": {"ref":"myActionId"}`
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체: `"onLateAction": {"ref":"myActionId"}`

선택 필드	설명	슬롯 유형
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체: `"onSuccess": {"ref":"myActionId"}`
output	출력 데이터 노드입니다. Amazon S3 또는 Amazon Redshift가 출력 위치가 될 수 있습니다.	참조 객체: `"output": {"ref":"myDataNodeId"}`
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체: `"parent": {"ref":"myBaseObjectId"}`
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체: `"precondition": {"ref":"myPreconditionId"}`
대기열	Amazon Redshift에서 `query_group` 설정에 해당하며, 이것을 사용하여 대기열 내 위치에 따라 동시 실행 활동을 할당하고 우선 순위를 정할 수 있습니다. Amazon Redshift는 동시 연결 수를 15로 제한합니다. 자세한 내용은 Amazon RDS 데이터베이스 개발자 안내서의 대기열에 쿼리 할당을 참조하세요.	문자열

선택 필드	설명	슬롯 유형
reportProgressTimeout	원격 작업에서 `reportProgress` 를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
scheduleType	파이프라인에서 객체에 대한 일정 조정을 지정할 수 있습니다. 값은 `cron`, `ondemand` 및 `timeseries` 입니다. `timeseries` 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정됩니다. `Cron` 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. `ondemand` 일정을 사용하면 활성화될 때마다 한 번씩 파이프라인을 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. `ondemand` 파이프라인을 사용하려면 후속 실행마다 `ActivatePipeline` 작업을 호출합니다. `ondemand` 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 `scheduleType` 이어야 합니다.	열거

선택 필드	설명	슬롯 유형
transformSql	입력 데이터를 변환할 때 사용되는 SQL SELECT 표현식입니다. staging이라는 테이블에서 transformSql 표현식을 실행합니다. DynamoDB 또는 Amazon S3의 데이터를 복사하면, AWS Data Pipeline 이 "스테이징"이라고 하는 테이블을 생성하고, 처음에는 여기에 데이터를 로드합니다. 이 테이블의 데이터는 대상 테이블을 업데이트할 때 사용됩니다. transformSql 의 출력 스키마는 최종 대상 테이블의 스키마와 일치해야 합니다. transformSql 옵션을 지정하는 경우, 지정된 SQL 문에서 두 번째 스테이징 테이블이 생성됩니다. 이후 이 두 번째 스테이징 테이블은 최종 대상 테이블에서 업데이트됩니다.	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체: "activeInstances":{"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체: `"cascadeFailedOn": {"ref":"myRunnableObjectId"}`
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체: `"waitingOn": {"ref":"myRunnableObjectId"}`

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 구입니다. 수명 주기상 객체의 위치를 나타냅니다. 예를 들어 구성 요소 객체가 인스턴스 객체를 트리거하고, 인스턴스 객체는 시도 객체를 실행합니다.	문자열

ShellCommandActivity

명령 또는 스크립트를 실행합니다. ShellCommandActivity를 사용하여 시계열 또는 Cron 같은 예약된 작업을 실행할 수 있습니다.

stage 필드가 참으로 설정되고 S3DataNode과(와) 함께 사용되는 경우 ShellCommandActivity은(는) 데이터 스테이징 개념을 지원합니다. 즉, Amazon S3에서 스테이지 위치(예: Amazon EC2 또는 로컬 환경)로 데이터를 이동하고 스크립트 및 ShellCommandActivity을(를) 사용하여 데이터에 대한 작업을 수행한 후 이를 다시 Amazon S3로 이동할 수 있습니다.

이 경우 셸 명령이 입력 S3DataNode에 연결되면 셸 스크립트는 ShellCommandActivity 입력 필드를 참조해 ${INPUT1_STAGING_DIR}, ${INPUT2_STAGING_DIR} 및 기타 필드를 사용하는 데이터에서 직접 작동할 수 있습니다.

마찬가지로 쉘 명령의 출력을 출력 디렉터리에서 스테이징하여 ${OUTPUT1_STAGING_DIR}, ${OUTPUT2_STAGING_DIR} 등에 의해 참조되는 Amazon S3에 자동으로 푸시할 수 있습니다.

이러한 표현식을 통해 데이터 변환 논리에 사용할 셸 명령에 명령줄 인수로 전달할 수 있습니다.

ShellCommandActivity는 Linux 스타일의 오류 코드 및 문자열을 반환합니다. ShellCommandActivity에서 오류가 발생하면 반환된 error는 0이 아닌 값이 됩니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "id" : "CreateDirectory",
  "type" : "ShellCommandActivity",
  "command" : "mkdir new-directory"
}
```

구문

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 schedule 간격을 실행할 때 호출됩니다.	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

액체 호출 필드	설명	슬롯 유형
	이 객체의 종속 실행 순서를 설정하려면 다른 객체로 schedule 참조를 지정합니다. 이 요구 사항을 충족하려면, 예컨대 "schedule": {"ref": "DefaultSchedule"} 을 지정하여 객체에서 schedule을 명시적으로 설정합니다. 대부분의 경우에는 모든 객체가 상속할 수 있도록 schedule 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 파이프라인이 일정 트리로 구성된 경우(마스터 일정 안의 일정) 일정 참조가 있는 부모 객체를 생성합니다. 로드를 분산하기 위해는 물리적 객체를 일정보다 약간 앞서 AWS Data Pipeline 생성하지만 일정에 따라 실행합니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
명령	실행할 명령입니다. $을 사용하여 명령의 파라미터를 지정할 위치 파라미터와 scriptArgument 를 참조합니다. 이 값 및 연결된 파라미터는 Task Runner를 실행 중인 환경에서 작동해야 합니다.	문자열
scriptUri	파일을 다운로드하여 셸 명령으로 실행할 Amazon S3 URI 경로입니다. 단 하나의 scriptUri 또는 command 필드만 지정합니	문자열

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
	다. `scriptUri` 는 파라미터를 사용할 수 없으며, `command`를 사용합니다.	

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
runsOn	Amazon EC2 인스턴스 또는 Amazon EMR 클러스터 등, 활동이나 명령을 실행할 전산 리소스입니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}
workerGroup	작업 라우팅에 사용됩니다. `workerGroup` 이 있을 때 `runsOn` 값을 제공하면 `workerGroup` 이 무시됩니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고한 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 지정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
입력	입력 데이터의 위치입니다.	참조 객체, 예: "input":{"ref":"myDataNodeId"}

선택 필드	설명	슬롯 유형
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수.	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
output	출력 데이터의 위치입니다.	참조 객체, 예: "output":{"ref":"myDataNodeId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 Amazon S3 URI(예:'s3://BucketName/Key/')입니다.	문자열

선택 필드	설명	슬롯 유형
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}
reportProgressTimeout	원격 작업에서 `reportProgress` 를 연속으로 호출하는 제한 시간입니다. 이것이 설정되면 지정 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주되어 재시도될 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
scheduleType	파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 값은 cron, ondemand 및 `timeseries` 입니다. `timeseries` 로 설정하면 각 간격이 종료될 때 인스턴스 일정이 지정됩니다. Cron로 설정하면 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. ondemand로 설정하면 활성화될 때마다 한 번씩 파이프라인을 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. ondemand 일정을 사용하려면 기본 객체에서 이것을 파이프라인의 객체에 유일한 `scheduleType` 으로 지정합니다. ondemand 파이프라인을 사용하려면 후속 실행마다 `ActivatePipeline` 작업을 호출합니다.	열거

선택 필드	설명	슬롯 유형
scriptArgument	명령으로 지정된 명령에 전달할 문자열의 JSON 형식 어레이입니다. 예를 들어, 명령이 `echo $1 $2`인 경우 `scriptArgument` 를 `"param1", "param2"`로 지정합니다. 여러 인수와 파라미터의 경우, `"scriptArgument":"arg1","scriptArgument":"param1","scriptArgument":"arg2","scriptArgument":"param2"` 와 같이 `scriptArgument` 를 전달합니다. `scriptArgument` 는 command와 함께 사용해야 하며 `scriptUri` 절과 함께 사용하면 오류가 발생합니다.	문자열
stage	스테이징 활성화 여부를 결정하며, 셸 명령을 사용하여 `${INPUT1_STAGING_DIR}` 및 `${OUTPUT1_STAGING_DIR}` 같은 스테이징 데이터 변수에 액세스할 수 있습니다.	부울
stderr	명령에서 리디렉션된 시스템 오류 메시지를 수신하는 경로입니다. `runsOn` 필드를 사용할 경우에는 활동을 실행할 리소스가 임시적이므로 Amazon S3 경로가 되어야 합니다. 그러나 `workerGroup` 필드를 지정할 경우에는 로컬 파일 경로가 허용됩니다.	문자열
stdout	명령에서 리디렉션된 출력을 수신하는 Amazon S3 경로입니다. `runsOn` 필드를 사용할 경우에는 활동을 실행할 리소스가 임시적이므로 Amazon S3 경로가 되어야 합니다. 그러나 `workerGroup` 필드를 지정할 경우에는 로컬 파일 경로가 허용됩니다.	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 `cancellationReason` 입니다.	문자열
@cascadeFailedOn	객체 실패를 유발한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	Amazon EMR 활동 시도 시에만 사용할 수 있는 Amazon EMR 단계 로그.	문자열
errorId	이 객체가 실패한 경우의 `errorId`입니다.	문자열
errorMessage	이 객체가 실패한 경우의 `errorMessage` 입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	Amazon EMR 기반 활동 시도 시에만 사용할 수 있는 Hadoop 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간입니다.	DateTime
@scheduledStartTime	객체의 일정 시작 시간입니다.	DateTime
@상태	객체의 상태입니다.	문자열
@version	객체를 생성하는 데 사용되는 AWS Data Pipeline 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류입니다.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID입니다.	문자열
@sphere	수명 주기상 객체의 위치. 구성요소 객체가 인스턴스 객체를 트리거하고, 인스턴스 객체는 시도 객체를 실행합니다.	문자열

참고

- CopyActivity
- EmrActivity

SqlActivity

데이터베이스에서 SQL 쿼리(스크립트)를 실행합니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "id" : "MySqlActivity",
  "type" : "SqlActivity",
  "database" : { "ref": "MyDatabaseID" },
  "script" : "SQLQuery" | "scriptUri" : s3://scriptBucket/query.sql,
  "schedule" : { "ref": "MyScheduleID" },
}
```

구문

필수 필드	설명	슬롯 유형
데이터베이스	제공된 SQL 스크립트를 실행할 데이터베이스.	참조 객체. 예: "database":{"ref": "myDatabaseId"}

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 다른 객체로 일정 참조를 지정해야 합니다. 예컨대 `"schedule": {"ref": "DefaultSchedule"}` 을 지정하여 객체에서 일정을 명시적으로 설정할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 파이프라인에 마스터 일정 내에서 중첩된 일정 트리가 있는 경우, 일정 참조가 있는 부모 객체를 생성합니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	참조 객체. 예: "schedule":{"ref": "myScheduleId"}

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
스크립트	실행할 SQL 스크립트. script 또는 scriptUri를 지정해야 합니다. 스크립트가 Amazon S3에 저장되면, 해당 스크립트는 표현식으로 평가되지 않습니다. 스크립트가 Amazon S3에 저장되는 경우에는 scriptArgument에 대해 복수 값을 지정하는 것이 도움이 됩니다.	문자열
scriptUri	이 활동에서 실행할 SQL 스크립트의 위치를 지정하는 URI.	문자열

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
runsOn	활동 또는 명령을 실행할 전산 리소스입니다. Amazon EC2 인스턴스 또는 Amazon EMR 클러스터가 이에 해당합니다.	참조 객체. 예: "runsOn":{"ref":"myResourceId"}
workerGroup	작업자 그룹입니다. 이것은 작업 라우팅에 사용됩니다. `workerGroup` 이 있을 때 `runsOn` 값을 제공하면 `workerGroup` 이 무시됩니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
dependsOn	실행 가능한 다른 객체의 종속성을 지정합니다.	참조 객체. 예: "dependsOn":{"ref":"myActivityId"}
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
입력	입력 데이터의 위치입니다.	참조 객체, 예: "input":{"ref":"myDataNodeId"}
lateAfterTimeout	객체 실행이 시작되어야 하는 예약된 파이프라인 시작 후 기간입니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수	Integer

선택 필드	설명	슬롯 유형
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	'lateAfterTimeout'에 의해 지정된 파이프라인의 예약된 시작 이후 기간 이내에 객체가 아직 예약되지 않았거나 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
output	출력 데이터의 위치입니다. 이는 스크립트 내에서의 참조(에: `#{output.tablename}`) 및 출력 데이터 노드에서의 'createTableSql' 설정에 의한 출력 테이블 생성 목적에만 유효합니다. SQL 쿼리의 출력은 출력 데이터 노드에 기록되지 않습니다.	참조 객체, 예: "output":{"ref":"myDataNodeId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
pipelineLogUri	파이프라인의 로그를 업로드할 S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
precondition	또는 사전 조건을 정의합니다. 모든 사전 조건이 충족되기 전까지 데이터 노드에 "READY"가 표시되지 않습니다.	참조 객체. 예: "precondition":{"ref":"myPreconditionId"}

선택 필드	설명	슬롯 유형
대기열	[Amazon Redshift만 해당] Amazon Redshift에서 쿼리 그룹 설정에 해당하며, 이것을 사용하여 쿼리 내 위치에 따라 동시 실행 활동을 할당하고 우선 순위를 정할 수 있습니다. Amazon Redshift는 동시 연결 수를 15로 제한합니다. 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서의 대기열에 쿼리 할당을 참조하세요.	문자열
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period

선택 필드	설명	슬롯 유형
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 값은 cron, ondemand 및 timeseries 입니다. timeseries 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정됩니다. cron 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. ondemand 일정을 사용하면 활성화될 때마다 한 번씩 파이프라인을 실행할 수 있습니다. 이 경우 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. ondemand 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 scheduleType 이어야 합니다. ondemand 파이프라인을 사용하려면 후속 실행마다 ActivatePipeline 작업을 호출합니다.	열거
scriptArgument	스크립트에 사용되는 변수의 목록입니다. 아니면 표현식을 스크립트 영역에 직접 입력할 수도 있습니다. 스크립트가 Amazon S3에 저장되는 경우에는 scriptArgument에 대한 복수 값 지정이 도움이 됩니다. 예: #{format(@scheduledStartTime, "YY-MM-DD HH:MM:SS"}\n#{format(plusPeriod(@scheduledStartTime, "1 day"), "YY-MM-DD HH:MM:SS"}	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 반영하는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열

시스템 필드	설명	슬롯 유형
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

리소스

다음은 AWS Data Pipeline 리소스 객체입니다.

Objects

- Ec2Resource
- EmrCluster
- HttpProxy

Ec2Resource

파이프라인 활동에 의해 정의된 작업을 수행하는 Amazon EC2 인스턴스입니다.

AWS Data Pipeline 는 이제 세션 지향 방법을 사용하여 인스턴스에서 메타데이터 정보를 검색할 때 인증을 더 잘 처리하는 Amazon EC2 인스턴스용 IMDSv2를 지원합니다. Amazon EC2 세션은 Amazon EC2 인스턴스에서 실행되는 소프트웨어가 로컬에 저장된 Amazon EC2 인스턴스 메타데이터 및 자격 증명에 액세스하는 데 사용하는 일련의 요청을 시작하고 종료합니다. 소프트웨어는 IMDSv2에 대한 간단한 HTTP PUT 요청으로 세션을 시작합니다. IMDSv2는 Amazon EC2 인스턴스에서 실행되는 소프트웨어에 비밀 토큰을 반환하며, 이 소프트웨어는 토큰을 암호로 사용하여 IMDSv2에 메타데이터 및 자격 증명을 요청합니다.

> **Note**
>
> 기본 AMI는 IMDSv2와 호환되지 않으므로 Amazon EC2 인스턴스에 IMDSv2를 사용하려면 설정을 수정해야 합니다. 다음 SSM 파라미터 `/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-ebs`을(를) 통해 검색할 수 있는 새 AMI 버전을 지정할 수 있습니다.

인스턴스를 지정하지 않은 경우에서 AWS Data Pipeline 생성하는 기본 Amazon EC2 인스턴스에 대한 자세한 내용은 섹션을 참조하세요AWS 리전에 의한 기본 Amazon EC2 인스턴스.

예제

EC2-Classic

> **Important**
>
> 2013년 12월 4일 이전에 생성된 AWS 계정만 EC2-Classic 플랫폼을 지원합니다. 이러한 계정 중 하나가 있는 경우 VPC보다 오히려 EC2-Classic 네트워크의 파이프라인에 대해 EC2Resource 객체를 생성할 수 있는 옵션이 있을 수 있습니다. VPC에서 모든 파이프라인의 리소스를 생성하는 것을 강력히 권장합니다. 또한 EC2-Classic에 기존 리소스가 있는 경우 이를 VPC로 마이그레이션하는 것이 좋습니다.

다음 예제 객체는 일부 옵션 필드가 설정된 EC2-Classic에서 EC2 인스턴스를 시작합니다.

```
{
  "id" : "MyEC2Resource",
  "type" : "Ec2Resource",
  "actionOnTaskFailure" : "terminate",
  "actionOnResourceFailure" : "retryAll",
  "maximumRetries" : "1",
  "instanceType" : "m5.large",
  "securityGroups" : [
    "test-group",
    "default"
  ],
  "keyPair" : "my-key-pair"
}
```

EC2-VPC

다음 예제 객체는 일부 옵션 필드가 설정된 기본이 아닌 VPC에서 EC2 인스턴스를 시작합니다.

```
{
  "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"
}
```

구문

필수 필드	설명	슬롯 유형
resourceRole	Amazon EC2 인스턴스가 액세스할 수 있는 리소스를 제어하는 IAM 역할입니다.	문자열
역할	가 EC2 인스턴스를 생성하는 데 AWS Data Pipeline 사용하는 IAM 역할입니다.	문자열

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 다른 객체로 일정 참조를 지정합니다. 이 작업을 다음 중 한 가지 방법으로 수행할 수 있습니다. • 파이프라인의 모든 객체가 일정을 상속받도록 하려면 명시 적으로 객체에 일정을 설정합니다. `"schedule": {"ref": "DefaultSchedule"}` . 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 유용합니다. • 파이프라인에 마스터 일정 내에서 중첩된 일정이 있는 경우, 일정 참조가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성	참조 객체, 예: `"schedule":{"ref":"myScheduleId"}`

액체 호출 필드	설명	슬롯 유형
	예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

선택 필드	설명	슬롯 유형
actionOnResourceFailure	이 리소스의 리소스 실패 후 취한 조치입니다. 유효 값은 "retryall" 및 "retrynone" 입니다.	문자열
actionOnTaskFailure	이 리소스의 작업 실패 후 취한 조치입니다. 유효한 값은 "continue" 또는 "terminate" 입니다.	문자열
associatePublicIpAddress	인터페이스로 퍼블릭 IP 주소를 할당할지 여부를 나타냅니다. 인스턴스가 Amazon EC2 또는 Amazon VPC에 포함된 경우 기본값은 true입니다. 그렇지 않은 경우 기본값은 false입니다.	부울
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 지정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
availabilityZone	Amazon EC2 인스턴스를 시작할 가용 영역입니다.	문자열
disableIMDSv1	기본값은 false이며 IMDSv1 및 IMDSv2를 모두 활성화합니다. 이 값을 true로 설정하면 IMDSv1이 비활성화되고 IMDSv2만 제공됩니다.	부울
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거

선택 필드	설명	슬롯 유형
httpProxy	클라이언트가 AWS 서비스에 연결하는 데 사용하는 프록시 호스트입니다.	참조 객체, 예: `"httpProxy": {"ref":"myHttpProxyId"}`
imageId	인스턴스에 사용할 AMI의 ID입니다. 기본적으로 HVM AMI 가상화 유형을 AWS Data Pipeline 사용합니다. 사용되는 특정 AMI ID는 리전을 기반으로 합니다. HVM AMI를 선택 지정하여 기본 AMI를 겹쳐쓸 수 있습니다. AMI 유형에 대해 자세한 내용은, Amazon EC2 사용 가이드Linux AMI 가상화 유형 및 Linux AMI 찾기를 참조하세요.	문자열
initTimeout	리소스가 시작되기 전까지의 대기 시간입니다.	Period
instanceCount	사용되지 않음.	Integer
instanceType	시작할 Amazon EC2 인스턴스의 유형입니다.	문자열
keyPair	키 페어 이름. 키 쌍을 지정하지 않고 Amazon EC2 인스턴스를 시작하면 인스턴스에 로그온되지 않습니다.	문자열
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수.	Integer
minInstanceCount	사용되지 않음.	Integer

선택 필드	설명	슬롯 유형
onFail	현재 객체에 장애가 있을 때 실행할 작업입니다.	참조 객체, 예: `"onFail": {"ref":"myActionId"}`
onLateAction	객체가 아직 예약되지 않았거나 계속 실행 중인 경우에 트리거되어야 하는 작업입니다.	참조 객체, 예: `"onLateAction": {"ref":"myActionId"}`
onSuccess	현재 객체가 성공하면 실행할 작업입니다.	참조 객체, 예: `"onSuccess": {"ref":"myActionId"}`
parent	슬롯을 상속해 올 현재 객체의 상위 객체입니다.	참조 객체, 예: `"parent": {"ref":"myBaseObjectId"}`
pipelineLogUri	파이프라인의 로그를 업로드할 Amazon S3 URI(예: `'s3://BucketName/Key/'`)입니다.	문자열
리전	Amazon EC2 인스턴스가 실행되어야 할 리전의 코드입니다. 기본적으로 인스턴스는 파이프라인과 동일한 리전에서 실행됩니다. 종속 데이터 세트와 동일한 리전에서 인스턴스를 실행할 수 있습니다.	열거
reportProgressTimeout	원격 작업에서 `reportProgress` 를 연속으로 호출하는 제한 시간입니다. 이것이 설정되면 지정 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주되어 재시도됩니다.	Period

선택 필드	설명	슬롯 유형
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
runAsUser	TaskRunner를 실행할 사용자.	문자열
runsOn	이 객체에서는 이 필드가 허용되지 않습니다.	참조 객체, 예:"runsOn": {"ref":"myResourceId"}
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로, 종료 시점으로 또는 필요할 때 지정할 수 있습니다. 유효한 값: • timeseries . 각 간격이 종료될 때 인스턴스 일정이 지정됩니다. • cron. 각 간격이 시작할 때 인스턴스 일정이 지정됩니다. • ondemand. 활성화될 때마다 한 번씩 파이프라인을 실행할 수 있습니다. 그러므로 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. 온디맨드 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 scheduleType 이어야 합니다. 온디맨드 파이프라인을 사용하려면 후속 실행마다 ActivatePipeline 작업을 호출하세요.	열거
securityGroupIds	리소스 풀의 인스턴스에 대해 사용할 하나 이상의 Amazon EC2 보안 그룹의 ID입니다.	문자열
securityGroups	리소스 풀의 인스턴스에 대해 사용할 하나 이상의 Amazon EC2 보안 그룹입니다.	문자열

선택 필드	설명	슬롯 유형
spotBidPrice	스팟 인스턴스의 시간당 최대 금액(달러)이며, 0초과 20.00 미만의 십진수 값입니다.	문자열
subnetId	인스턴스를 시작할 Amazon EC2 서브넷의 ID입니다.	문자열
terminateAfter	이 숫자만큼의 시간이 지난 뒤 리소스를 종료합니다.	Period
useOnDemandOnLastAttempt	스팟 인스턴스를 마지막으로 요청할 때 스팟 인스턴스가 아니라 온디맨드 인스턴스를 요청합니다. 이렇게 하면 이전의 시도가 모두 실패했어도 마지막 시도가 중단되지 않습니다.	부울
workerGroup	이 객체에서는 이 필드가 허용되지 않습니다.	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체, 예:"activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason 입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체, 예:"cascadeFailedOn":

실행 시간 필드	설명	슬롯 유형
		`{"ref":"myRunnableObjectId"}`
emrStepLog	Amazon EMR 활동을 시도할 때만 사용할 수 있는 단계 로그.	문자열
errorId	이 객체가 실패한 경우의 오류 ID입니다.	문자열
errorMessage	이 객체가 실패한 경우의 오류 메시지입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@failureReason	리소스 실패 이유입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	Amazon EMR 활동을 시도할 때 사용할 수 있는 Hadoop 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 나타내는 객체의 상태입니다.	문자열
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime

실행 시간 필드	설명	슬롯 유형
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체를 생성할 당시의 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체, 예: `"waitingOn": {"ref":"myRunnableObjectId"}`

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	수명 주기상 객체의 위치. 구성요소 객체가 인스턴스 객체를 트리거하고, 인스턴스 객체는 시도 객체를 실행합니다.	문자열

EmrCluster

Amazon EMR 클러스터의 구성을 나타냅니다. 이 객체는 EmrActivity 및 HadoopActivity에 의해 클러스터를 시작합니다.

내용

스케줄러

스케줄러는 Hadoop 클러스터에서 리소스 할당 및 작업 우선 순위를 지정하는 방법을 제공합니다. 관리자 또는 사용자는 사용자 및 애플리케이션의 다양한 클래스에 대한 스케줄러를 선택할 수 있습니다. 스케줄러는 대기열을 사용하여 사용자 및 애플리케이션에 리소스를 할당할 수 있습니다. 클러스터를 생성할 때 이러한 대기열을 설정합니다. 그런 다음 특정 작업 유형 및 사용자에 우선순위를 설정할 수 있습니다. 이렇게 하면 클러스터 리소스를 효과적으로 사용할 수 있으며 둘 이상의 사용자가 클러스터에 작업을 제출할 수 있습니다. 세 가지 유형의 스케줄러를 사용할 수 있습니다.

- FairScheduler — 상당한 기간 동안 리소스를 일정하게 예약을 시도하는 스케줄러입니다.
- CapacityScheduler — 대기열을 사용하여 클러스터 관리자가 다양한 우선 순위 및 리소스 할당 대기열에 사용자를 할당할 수 있는 스케줄러입니다.
- Default — 기본 스케줄러 사이트를 구성할 수 있는 클러스터에 사용되는 스케줄러입니다.

Amazon EMR 릴리스 버전

Amazon EMR 릴리스는 빅 데이터 에코시스템의 오픈 소스 애플리케이션입니다. 각 릴리스는 클러스터를 생성할 때 Amazon EMR을 설치하고 구성하도록 선택한 여러 빅 데이터 애플리케이션, 구성 요소 및 기능으로 구성됩니다. 릴리스 레이블을 사용하여 릴리스 버전을 지정합니다. 릴리스 레이블은 `emr-x.x.x` 형식입니다. 예를 들어 `emr-5.30.0`입니다. 릴리스 레이블 `emr-4.0.0`을 기반으로 한 Amazon EMR 클러스터는 나중에 `releaseLabel` 속성을 사용하여 `EmrCluster` 객체의 릴리스 레이블을 지정합니다. 이전 버전에서는 `amiVersion` 속성을 사용합니다.

> **Important**
>
> 릴리스 버전 5.22.0 이상을 사용하여 생성된 모든 Amazon EMR 클러스터는 나중에 서명 버전 4를 사용하여 Amazon S3에 대한 요청을 인증합니다. 일부 이전 릴리스 버전에서는 서명 버전 2를 사용합니다. 서명 버전 2 지원이 중단되고 있습니다. 자세한 내용은 Amazon S3 업데이트

— SigV2 사용 중단 기간 연장 및 수정을 참조하세요. 서명 버전 4를 지원하는 Amazon EMR 릴리스 버전을 사용하는 것이 좋습니다. EMR 4.7.x부터 이전 릴리스 버전의 경우 시리즈의 최신 릴리스가 서명 버전 4를 지원하도록 업데이트되었습니다. 이전 버전의 EMR 릴리스를 사용하는 경우 시리즈의 최신 릴리스를 사용하는 것이 좋습니다. 또한 EMR 4.7.0 이전 릴리스는 사용하지 마십시오.

고려 사항 및 제한

최신 버전의 Task Runner 사용

릴리스 레이블을 가지고 자체 관리형 `EmrCluster` 객체를 사용 중인 경우에는 최신 Task Runner를 사용합니다. 작업 실행기에 대한 정보는 Task Runner로 작업하기를 참조하세요. 모든 Amazon EMR 구성 분류에 대한 속성 값을 구성할 수 있습니다. 자세한 내용을 알아보려면 Amazon EMR 릴리스 안내서, the section called “EmrConfiguration” 및 the section called “속성” 객체 참조의 구성 애플리케이션을 참조하세요.

IMDSv2에 대한 지원

이전에는 IMDSv1만 AWS Data Pipeline 지원되었습니다. 이제는 Amazon EMR 5.23.1, 5.27.1, 5.32 이상 및 Amazon EMR 6.2 이상에서 IMDSv2를 AWS Data Pipeline 지원합니다. IMDSv2는 세션 지향 방법을 사용하여 인스턴스에서 메타데이터 정보를 검색할 때 인증을 더 잘 처리합니다. TaskRunner-2.0을 사용하여 사용자 관리 리소스를 생성하여 IMDSv2 호출을 수행하도록 인스턴스를 구성해야 합니다.

Amazon EMR 5.32 이상 및 Amazon EMR 6.x

Amazon EMR 5.32 이상 및 6.x 릴리스 시리즈는 Hadoop 버전 3.x를 사용합니다. 이 버전에서는 Hadoop 버전 2.x와 비교하여 Hadoop의 클래스 경로를 평가하는 방식이 크게 변경되었습니다. Joda-Time과 같은 일반 라이브러리는 클래스 경로에서 제거되었습니다.

EmrActivity 또는 HadoopActivity이(가) Hadoop 3.x에서 제거된 라이브러리에 대한 종속성이 있는 Jar 파일을 실행하는 경우, 단계는 오류 `java.lang.NoClassDefFoundError` 또는 `java.lang.ClassNotFoundException`이(가) 발생하면서 실패합니다. Amazon EMR 5.x 릴리스 버전을 사용하여 문제 없이 실행한 Jar 파일에서 이 문제가 발생할 수 있습니다.

문제를 해결하려면 `EmrActivity` 또는 `HadoopActivity`을(를) 시작하기 전에 Jar 파일 종속성을 `EmrCluster` 상의 객체의 Hadoop 클래스 경로에 복사해야 합니다. 이를 수행할 bash 스크립트를 제공합니다. bash 스크립트는와 같이 *`MyRegion`*이 `EmrCluster` 객체가 실행되는 AWS 리전인 다음 위치에서 사용할 수 있습니다`us-west-2`.

```
s3://datapipeline-MyRegion/MyRegion/bootstrap-actions/latest/TaskRunner/copy-jars-to-
hadoop-classpath.sh
```

스크립트를 실행하는 방법은 EmrActivity 또는가에서 관리하는 리소스에서 HadoopActivity 실행되는지 AWS Data Pipeline 또는 자체 관리형 리소스에서 실행되는지에 따라 달라집니다.

에서 관리하는 리소스를 사용하는 경우 EmrCluster 객체bootstrapAction에를 AWS Data Pipeline추가합니다. bootstrapAction은(는) 복사할 스크립트와 Jar 파일을 인수로 지정합니다. EmrCluster 객체당 최대 255개의 bootstrapAction 필드를 추가할 수 있으며 bootstrapAction 필드를 이미 부트스트랩 작업이 있는 EmrCluster 객체에 추가할 수 있습니다.

이 스크립트를 부트스트랩 작업으로 지정하려면 다음 구문을 사용하세요. 여기서 JarFileRegion은(는) Jar 파일이 저장되는 리전이고, 각 *MyJarFile n*은 Amazon S3에서 Hadoop 클래스 경로로 복사할 Jar 파일의 절대 경로입니다. 기본적으로 Hadoop 클래스 경로에 있는 Jar 파일은 지정하지 마십시오.

```
s3://datapipeline-MyRegion/MyRegion/bootstrap-actions/latest/TaskRunner/copy-jars-to-
hadoop-classpath.sh,JarFileRegion,MyJarFile1,MyJarFile2[, ...]
```

다음 예제는 Amazon S3에 있는 Jar 파일 두 개(my-jar-file.jar 및 emr-dynamodb-tool-4.14.0-jar-with-dependencies.jar)를 복사하는 부트스트랩 작업을 지정합니다. 이 예에서 사용되는 리전은 us-west-2입니다.

```
{
  "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"]
}
```

파이프라인을 저장하고 활성화해야 새 bootstrapAction에 대한 변경 사항이 적용됩니다.

자체 관리형 리소스를 사용하는 경우, 스크립트를 클러스터 인스턴스로 다운로드하고 SSH를 사용하여 명령줄에서 실행할 수 있습니다. 스크립트는 /etc/hadoop/conf/shellprofile.d(이)라는 이름이 지정된 디렉터리와 해당 디렉터리에 datapipeline-jars.sh(이)라는 이름의 파일을 생성합니다. 명령줄 인수로 제공된 jar 파일은 스크립트에서 만든 /home/hadoop/datapipeline_jars(이)라는 디렉터리에 복사됩니다. 클러스터가 다르게 설정된 경우 스크립트를 다운로드한 후 적절하게 수정하세요.

명령줄에서 스크립트를 실행하는 구문은 이전 예제에 표시된 bootstrapAction을(를) 사용하는 것과 약간 다릅니다. 다음 예제에서 볼 수 있듯이 같이 인수 간에 쉼표 대신 스페이스를 사용합니다.

```
./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
```

Amazon EMR 권한

사용자 지정 IAM 역할을 생성하는 경우, 작업을 수행하기 위해 클러스터에 필요한 최소 권한을 신중하게 고려하세요. Amazon S3에 있는 파일, Amazon RDS Amazon Redshift 또는 DynamoDB에 있는 데이터와 같은 필수 리소스에 권한을 부여해야 합니다. visibleToAllUsers를 false로 설정하려면 역할에 적절한 해당 권한이 있어야 합니다. DataPipelineDefaultRole에는 해당 권한이 없음을 참조하세요. EmrCluster 객체 역할로서 DataPipelineDefaultRole 역할 및 DefaultDataPipelineResourceRole 객체 역할의 조합을 제공하거나 이런 목적을 위해 직접 역할을 생성해야 합니다.

구문

액체 호출 필드	설명	슬롯 유형
schedule	이 객체는 예약 간격을 실행할 때 호출됩니다. 이 객체의 종속 실행 순서를 설정하려면 다른 객체로 일정 참조를 지정합니다. 사용자가 객체에서 일정을 명확히 설정하여(예: "schedule": {"ref": "DefaultSchedule"} 지정)하여 이 요건을 충족할 수 있습니다. 대부분의 경우에서는 모든 객체가 상속할 수 있도록 일정 참조를 기본 파이프라인 객체에 두는 것이 좋습니다. 또는 파이프라인에 일정 트리가 있는 경우(마스터 일정 안의 일정) 사용자가 일정 참조	참조 객체, 예: "schedule":{"ref": "myScheduleId"}

액체 호출 필드	설명	슬롯 유형
	가 있는 부모 객체를 생성할 수 있습니다. 선택형 일정 구성 예제에 대한 자세한 내용은 단원을 참조하세요https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

선택 필드	설명	슬롯 유형
actionOnResourceFailure	이 리소스의 리소스 실패 후 취한 조치입니다. 유효 값은 지정 시간 동안 클러스터의 모든 작업을 재시도하는 "retryall"과 "retrynone "입니다.	문자열
actionOnTaskFailure	이 리소스의 작업 실패 후 취한 조치입니다. 유효 값은 클러스터를 종료하지 않는 "continue"와 "terminate"입니다.	문자열
additionalMasterSecurityGroupIds	EMR 클러스터의 추가 마스터 보안 그룹의 식별자로, sg-01XXXX6a 형식을 따릅니다. 자세한 내용은 Amazon EMR 관리 안내서의 Amazon EMR 추가 보안 그룹을 참조하세요.	문자열
additionalSlaveSecurityGroupIds	EMR 클러스터의 추가 슬레이브 보안 그룹의 식별자로, sg-01XXXX6a 형식을 따릅니다.	문자열
amiVersion	클러스터 노드를 설치할 때 Amazon EMR이 사용하는 Amazon Machine Image(AMI) 버전입니다. 자세한 내용은 Amazon EMR 관리 안내서를 참조하세요.	문자열
애플리케이션	쉼표로 구분된 인수가 있는 클러스터에 설치하는 애플리케이션입니다. 기본적으로 Hive 및 Pig가 설치됩니다. 이 파라미터는 Amazon EMR 버전 4.0 이상에만 적용됩니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고한 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
availabilityZone	클러스터를 실행할 가용 영역입니다.	문자열
bootstrapAction	클러스터가 시작될 때 실행할 작업입니다. 쉼표로 구분된 인수를 지정할 수 있습니다. 최대 255개까지 여러 작업을 지정하려면 여러 `bootstrapAction` 필드를 추가합니다. 기본 동작은 부트스트랩 작업이 없는 클러스터를 시작하는 것입니다.	문자열
구성	Amazon EMR에 대한 구성. 이 파라미터는 Amazon EMR 버전 4.0 이상에만 적용됩니다.	참조 객체, 예:`"configuration":{"ref":"myEmrConfigurationId"}`
coreInstanceBidPrice	Amazon EC2 인스턴스에 대해 지불하고자 하는 최고 스팟 가격입니다. 입찰 가격이 정해지면 Amazon EMR은 인스턴스 그룹에 대해 스팟 인스턴스를 활성화합니다. 달러 단위로 지정됩니다.	문자열
coreInstanceCount	클러스터에 사용할 코어 노드의 수입니다.	Integer
coreInstanceType	코어 노드에 사용할 Amazon EC2 인스턴스의 유형입니다. 지원된 Amazon EMR 클러스터에 대한 Amazon EC2 인스턴스을(를) 참조하세요.	문자열

선택 필드	설명	슬롯 유형
coreGroupConfiguration	Amazon EMR 클러스터 코어 인스턴스 그룹의 구성. 이 파라미터는 Amazon EMR 버전 4.0 이상에만 적용됩니다.	참조 객체, 예: `"configuration": {"ref": "myEmrConfigurationId"}`
coreEbsConfiguration	Amazon EMR 클러스터의 코어 그룹에서 각 코어 노드에 연결될 Amazon EBS 볼륨의 구성입니다. 자세한 내용을 알아보려면 Amazon EC2 사용 설명서의 EBS 최적화를 지원하는 인스턴스 유형을 참조하세요.	참조 객체, 예: `"coreEbsConfiguration": {"ref": "myEbsConfiguration"}`
customAmiId	Amazon EMR 릴리스 5.7.0 이상에만 적용됩니다. Amazon EMR이 Amazon EC2 인스턴스를 프로비저닝할 때 사용할 사용자 지정 AMI의 AMI ID를 지정합니다. 부트스트랩 작업 대신 클러스터 노드 구성을 사용자 지정하는 데 사용할 수도 있습니다. 자세한 내용은 Amazon EMR 관리 안내서에서 다음 항목을 참조하세요. 사용자 지정 AMI 사용	문자열

선택 필드	설명	슬롯 유형
EbsBlockDeviceConfig	인스턴스 그룹과 연결된 요청한 Amazon EBS 블록 디바이스의 구성입니다. 인스턴스 그룹의 각 인스턴스와 연결될 지정된 개수의 볼륨을 포함합니다. `volumesPerInstance` 및 `volumeSpecification` 을 포함합니다. 여기서, • `volumesPerInstance` 는 인스턴스 그룹의 각 인스턴스와 연결될 특정 볼륨 구성이 포함된 EBS 볼륨 수입니다. • `volumeSpecification` 은(는) Amazon EMR 클러스터의 EC2 인스턴스에 연결된 EBS 볼륨에 대해 요청될 기비바이트(GiB)의 볼륨 유형, IOPS 및 크기 같은 Amazon EBS 볼륨 사양입니다.	참조 객체, 예: `“EbsBlockDeviceConfig”: {“ref”: “myEbsBlockDeviceConfig”}`
emrManagedMasterSecurityGroupId	Amazon EMR 클러스터의 마스터 보안 그룹 식별자로, `sg-01XXXX6a` 형식을 따릅니다. 자세한 내용을 알아보려면 Amazon EMR 관리 안내서의 인스턴스 플릿 구성을 참조하세요.	문자열
emrManagedSlaveSecurityGroupId	Amazon EMR 클러스터의 슬레이브 보안 그룹 식별자로, `sg-01XXXX6a` 형식을 따릅니다.	문자열
enableDebugging	Amazon EMR 클러스터에서 디버깅을 활성화합니다.	문자열
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
hadoopSchedulerType	클러스터의 스케줄러 유형입니다. 유효한 형식은 `PARALLEL_FAIR_SCHEDULING` , `PARALLEL_CAPACITY_SCHEDULING` , 및 `DEFAULT_SCHEDULER` 입니다.	열거

선택 필드	설명	슬롯 유형
httpProxy	클라이언트가 AWS 서비스에 연결할 때 사용할 프록시 호스트입니다.	참조 객체. 예: "httpProxy":{"ref" :"myHttpProxyId"}
initTimeout	리소스가 시작되기 전까지의 대기 시간입니다.	Period
keyPair	Amazon EMR 클러스터의 마스터 노드에 로그인할 때 사용하는 Amazon EC2 키 페어입니다.	문자열
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
masterInstanceBidPrice	Amazon EC2 인스턴스에 대해 지불하고자 하는 최고 스팟 가격입니다. 0~20.00 사이의 십진수 값을 지정합니다. 달러 단위로 지정됩니다. 이 값을 설정하면 Amazon EMR 클러스터 마스터 노드의 스팟 인스턴스가 활성화됩니다. 입찰 가격이 정해지면 Amazon EMR은 인스턴스 그룹에 대해 스팟 인스턴스를 활성화합니다.	문자열
masterInstanceType	마스터 노드에 사용할 Amazon EC2 인스턴스의 유형입니다. 지원된 Amazon EMR 클러스터에 대한 Amazon EC2 인스턴스을(를) 참조하세요.	문자열
masterGroupConfiguration	Amazon EMR 클러스터 마스터 인스턴스 그룹의 구성입니다. 이 파라미터는 Amazon EMR 버전 4.0 이상에만 적용됩니다.	참조 객체, 예: `“configuration”: {“ref”: “myEmrConfigurationId”}`

선택 필드	설명	슬롯 유형
masterEbsConfiguration	Amazon EMR 클러스터의 마스터 그룹에서 각 마스터 노드에 연결될 Amazon EBS 볼륨의 구성입니다. 자세한 내용을 알아보려면 Amazon EC2 사용 설명서의 EBS 최적화를 지원하는 인스턴스 유형을 참조하세요.	참조 객체, 예: `“masterEbsConfiguration”: {“ref”: “myEbsConfiguration”}`
maxActiveInstances	구성요소의 동시 활성 인스턴스 최대수입니다. 재실행은 활성 인스턴스의 수에 포함되지 않습니다.	Integer
maximumRetries	장애 시 재시도 최대 횟수.	Integer
onFail	현재 객체에 장애가 있을 때 실행할 작업입니다.	참조 객체, 예:`"onFail": {"ref":"myActionId"}`
onLateAction	객체가 아직 예약되지 않았거나 여전히 완료되지 않은 경우에 트리거해야 하는 작업입니다.	참조 객체, 예:`"onLateAction":{"ref":"myActionId"}`
onSuccess	현재 객체가 성공하면 실행할 작업입니다.	참조 객체, 예:`"onSuccess": {"ref":"myActionId"}`
parent	슬롯을 상속해 올 현재 객체의 상위 객체입니다.	참조 객체, 예:`"parent": {"ref":"myBaseObjectId"}`

선택 필드	설명	슬롯 유형
pipelineLogUri	파이프라인의 로그를 업로드할 Amazon S3 URI(예: 's3://BucketName/Key/')입니다.	문자열
리전	Amazon EMR 클러스터가 실행되어야 할 리전의 코드입니다. 기본적으로 이 클러스터는 파이프라인과 동일한 리전에서 실행됩니다. 종속 데이터 세트와 동일한 리전에서 클러스터를 실행할 수 있습니다.	열거
releaseLabel	EMR 클러스터용 릴리스 레이블.	문자열
reportProgressTimeout	원격 작업에서 `reportProgress` 를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
resourceRole	가 Amazon EMR 클러스터를 생성하는 데 AWS Data Pipeline 사용하는 IAM 역할입니다. 기본 역할은 `DataPipelineDefaultRole` 입니다.	문자열
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
역할	EC2 노드를 생성하기 위해 Amazon EMR로 전달되는 IAM 역할입니다.	문자열
runsOn	이 객체에서는 이 필드가 허용되지 않습니다.	참조 객체, 예:`"runsOn": {"ref":"myResourceId"}`
securityConfiguration	클러스터에 적용된 EMR 보안 구성의 식별자 이름입니다. 이 파라미터는 Amazon EMR 버전 4.8.0. 이상에만 적용됩니다.	문자열

선택 필드	설명	슬롯 유형
serviceAccessSecurityGroupId	Amazon EMR 클러스터의 서비스 액세스 보안 그룹의 식별자입니다.	문자열. `sg-01XXXX6a` 형식을 따릅니다. 예: `sg-1234abcd` .
scheduleType	일정 유형을 사용하여 파이프라인 정의에 있는 객체의 일정을 간격 시작 시점으로 또는 종료 시점으로 지정할 수 있습니다. 값은 `cron`, `ondemand` 및 `timeseries` 입니다. `timeseries` 일정 조정은 각 간격이 종료될 때 인스턴스 일정이 지정됩니다. `cron` 일정 조정은 각 간격이 시작될 때 인스턴스 일정이 지정됩니다. `ondemand` 일정을 사용하면 활성화될 때마다 한 번씩 파이프라인을 실행할 수 있습니다. 그러므로 다시 실행하기 위해 파이프라인을 복제하거나 다시 생성할 필요가 없습니다. `ondemand` 일정을 사용하려면 기본 객체에서 지정해야 하며, 이것이 파이프라인의 객체에 지정된 유일한 `scheduleType` 이어야 합니다. `ondemand` 파이프라인을 사용하려면 후속 실행마다 `ActivatePipeline` 작업을 호출합니다.	열거
subnetId	Amazon EMR 클러스터를 시작할 서브넷의 식별자입니다.	문자열
supportedProducts	Amazon EMR 클러스터에 타사 소프트웨어를 설치하는 파라미터입니다(예: 타사 Hadoop 배포 버전 설치).	문자열
taskInstanceBidPrice	EC2 인스턴스에 대해 지불하고자 하는 최고 스팟 가격입니다. 0~20.00 사이의 십진수 값을 지정합니다. 달러 단위로 지정됩니다. 입찰 가격이 정해지면 Amazon EMR은 인스턴스 그룹에 대해 스팟 인스턴스를 활성화합니다.	문자열

선택 필드	설명	슬롯 유형
taskInstanceCount	Amazon EMR 클러스터에 사용할 작업 노드의 수입니다.	Integer
taskInstanceType	작업 노드에 사용할 Amazon EC2 인스턴스의 유형입니다.	문자열
taskGroupConfiguration	Amazon EMR 클러스터 작업 인스턴스 그룹의 구성입니다. 이 파라미터는 Amazon EMR 버전 4.0 이상에만 적용됩니다.	참조 객체, 예: `"configuration": {"ref": "myEmrConfigurationId"}`
taskEbsConfiguration	Amazon EMR 클러스터의 작업 그룹에서 각 작업 노드에 연결될 Amazon EBS 볼륨의 구성입니다. 자세한 내용을 알아보려면 Amazon EC2 사용 설명서의 EBS 최적화를 지원하는 인스턴스 유형을 참조하세요.	참조 객체, 예: `"taskEbsConfiguration": {"ref": "myEbsConfiguration"}`
terminateAfter	이 여러 시간 이후에 리소스를 종료합니다.	Integer

선택 필드	설명	슬롯 유형
VolumeSpecification	Amazon EMR 클러스터의 Amazon EC2 인스턴스에 연결된 Amazon EBS 볼륨에 대해 요청될 기비바이트(GiB)의 볼륨 유형, IOPS 및 크기 같은 Amazon EBS 볼륨 사양입니다. 이 노드는 코어, 마스터 또는 작업 노드일 수 있습니다. `VolumeSpecification` 에는 다음이 포함됩니다. • `iops()` 정수. Amazon EBS 볼륨이 지원하는 초당 I/O 작업 수(IOPS)입니다(예: 1000). 자세한 내용은 Amazon EC2 사용 설명서의 EBS I/O Characteristics를 참조하세요. • `sizeinGB()` . 정수. 기비바이트(GiB) 단위의 Amazon EBS 볼륨 크기입니다(예: 500). 볼륨 유형과 하드 드라이브 크기의 유효한 조합에 대한 자세한 내용은 Amazon EC2 사용 설명서의 EBS 볼륨 유형을 참조하세요. • `volumetType` . 문자열. Amazon EBS 볼륨 유형. 예: gp2. 지원되는 볼륨 유형에는 gp2, io1, st1, sc1 등이 있습니다. 자세한 내용을 알아보려면 Amazon EC2 사용 설명서의 EBS 볼륨 유형을 참조하세요.	참조 객체, 예: `“VolumeSpecification”: {“ref”: “myVolumeSpecification”}`
useOnDemandOnLastAttempt	리소스를 마지막으로 요청할 때 스팟 인스턴스가 아니라 온디맨드 인스턴스를 요청합니다. 이렇게 하면 이전의 시도가 모두 실패했어도 마지막 시도가 중단되지 않습니다.	부울
workerGroup	이 객체에서 허용되지 않는 필드.	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	Amazon EMR 활동을 시도할 때만 사용할 수 있는 단계 로그.	문자열
errorId	이 객체가 실패한 경우의 오류 ID입니다.	문자열
errorMessage	이 객체가 실패한 경우의 오류 메시지입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
@failureReason	리소스 실패 이유입니다.	문자열
@finishedTime	이 객체의 실행이 완료된 시간입니다.	DateTime
hadoopJobLog	Amazon EMR 활동을 시도할 때 사용할 수 있는 Hadoop 작업 로그.	문자열
@healthStatus	종료 상태에 도달한 마지막 객체 인스턴스의 성공 또는 실패를 나타내는 객체의 상태입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@healthStatusFromInstanceId	종료 상태에 도달한 마지막 인스턴스 객체의 ID입니다.	문자열
@healthStatusUpdatedTime	상태가 마지막으로 업데이트된 시간입니다.	DateTime
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
@lastDeactivatedTime	이 객체가 마지막으로 비활성화된 시간입니다.	DateTime
@latestCompletedRunTime	실행이 완료된 최근 실행 시간입니다.	DateTime
@latestRunTime	실행이 예약된 최근 실행 시간입니다.	DateTime
@nextRunTime	다음으로 예약된 실행 시간입니다.	DateTime
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체를 생성할 당시의 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	수명 주기상 객체의 위치. 구성요소 객체가 인스턴스 객체를 트리거하고, 인스턴스 객체는 시도 객체를 실행합니다.	문자열

예제

다음은 이 객체 유형의 예제입니다.

내용

- hadoopVersion을 사용하여 Amazon EMR 클러스터 시작
- 릴리스 레이블이 emr-4.x 이상인 Amazon EMR 클러스터를 시작합니다.
- Amazon EMR 클러스터에 추가 소프트웨어 설치
- 3.x 릴리스에서 서버 측 암호화를 비활성화
- 4.x 릴리스에서 서버 측 암호화를 비활성화
- Hadoop KMS ACL을 구성하고 HDFS에서 암호화 영역 생성
- 사용자 지정 IAM 역할을 지정
- AWS SDK for Java에서 EmrCluster 리소스 사용
- 프라이빗 서브넷에서 Amazon EMR 클러스터 구성
- 클러스터 노드에 EBS 볼륨 연결

hadoopVersion을 사용하여 Amazon EMR 클러스터 시작

Example

다음 예제는 AMI 버전 1.0 및 Hadoop 0.20.을 사용하여 Amazon EMR 클러스터를 시작합니다.

```
{
  "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"]
}
```

릴리스 레이블이 emr-4.x 이상인 Amazon EMR 클러스터를 시작합니다.

Example

다음 예제는 최신 `releaseLabel` 필드를 사용하여 Amazon EMR 클러스터를 시작합니다.

```
{
  "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"}
}
```

Amazon EMR 클러스터에 추가 소프트웨어 설치

Example

`EmrCluster`은(는) Amazon EMR 클러스터에 타사 소프트웨어를 설치하는 `supportedProducts` 필드를 제공합니다(예: MapR과 같은 Hadoop의 사용자 지정 배포 설치). 타사 소프트웨어의 쉼표로 분리된 인수 목록에 대한 읽기 및 실행을 허용합니다. 다음 예제는 `EmrCluster`의 `supportedProducts` 필드를 사용하여 설치된 Karmasphere Analytics로 사용자 지정 MapR M3 에디션 클러스터를 생성하고 이 클러스터에서 `EmrActivity` 객체를 실행하는 방법을 보여줍니다.

```
{
    "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"
}
```

3.x 릴리스에서 서버 측 암호화를 비활성화

Example

에서 생성한 하둡 버전 2.x를 사용하는 EmrCluster 활동은 기본적으로 서버 측 암호화를 AWS Data Pipeline 활성화합니다. 서버 측 암호화를 비활성화하려면 클러스터 객체 정의에서 부트스트랩 작업을 지정해야 합니다.

다음 예제에서는 서버 측 암호화가 비활성화된 EmrCluster 활동을 만듭니다.

```
{
   "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"]
}
```

4.x 릴리스에서 서버 측 암호화를 비활성화

Example

EmrConfiguration 객체를 사용하여 서버 측 암호화를 비활성화해야 합니다.

다음 예제에서는 서버 측 암호화가 비활성화된 EmrCluster 활동을 만듭니다.

```
{
   "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"
 }
```

Hadoop KMS ACL을 구성하고 HDFS에서 암호화 영역 생성

Example

다음 객체는 Hadoop KMS에 대한 ACL을 생성하고 HDFS에서 암호화 영역과 해당 암호화 키를 생성합니다.

```
{
```

```
      "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"
    }
```

사용자 지정 IAM 역할을 지정

Example

기본적으로는 DataPipelineDefaultRole 사용자를 대신하여 리소스를 생성하기 위해 Amazon EMR 서비스 역할 및 Amazon EC2 인스턴스 프로파일DataPipelineDefaultResourceRole로 AWS Data Pipeline 전달합니다. 그러나 사용자 지정 Amazon EMR 서비스 역할과 사용자 지정 인스턴스 프로파일을 생성하고 대신 사용할 수 있습니다. AWS Data Pipeline 에는 사용자 지정 역할을 사용하여 클러스터를 생성할 수 있는 충분한 권한이 있어야 하며를 신뢰할 수 있는 엔 AWS Data Pipeline 터티로 추가해야 합니다.

다음 예시 객체는 Amazon EMR 클러스터에 대한 사용자 지정 역할을 지정합니다.

```
{
   "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"
}
```

AWS SDK for Java에서 EmrCluster 리소스 사용

Example

다음 예제에서는 EmrCluster 및 EmrActivity을(를) 사용하여 Amazon EMR 4.x 클러스터를 생성해 Java SDK를 사용하는 Spark 단계를 실행하는 방법을 알아봅니다.

```
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);

     }

}
```

프라이빗 서브넷에서 Amazon EMR 클러스터 구성

Example

이 예제는 클러스터를 VPC의 프라이빗 서브넷에서 실행하는 구성을 포함합니다. 자세한 내용은 Amazon EMR 관리 가이드의 Amazon EMR 클러스터를 VPC에서 시작을 참조하세요. 이 구성은 선택 사항입니다. `EmrCluster` 객체를 사용하는 모든 파이프라인에서 이 구성을 사용할 수 있습니다.

프라이빗 서브넷에서 Amazon EMR 클러스터를 실행하려면 `SubnetId`, `emrManagedMasterSecurityGroupId`, `emrManagedSlaveSecurityGroupId` 및 `serviceAccessSecurityGroupId`을(를) `EmrCluster` 구성에서 지정합니다.

```
{
  "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"
  }
}
```

클러스터 노드에 EBS 볼륨 연결

Example

EBS 볼륨을 파이프라인 내에서 EMR 클러스터의 노드 유형 하나에 연결할 수 있습니다. EBS 볼륨을 노드에 연결하려면 `EmrCluster` 구성에서 `coreEbsConfiguration`, `masterEbsConfiguration` 및 `TaskEbsConfiguration`을 사용합니다.

이 Amazon EMR 클러스터 예제에는 마스터, 작업 및 코어 노드에 대해 Amazon EBS 볼륨이 사용됩니다. 자세한 내용을 알아보려면 Amazon EMR 관리 안내서의 Using Automatic Scaling in Amazon EMR의 Amazon EBS 볼륨을 참조하세요.

이러한 구성은 선택 사항입니다. `EmrCluster` 객체를 사용하는 모든 파이프라인에서 이러한 구성을 사용할 수 있습니다.

파이프라인에서 EmrCluster 객체 구성을 클릭하고 Master EBS Configuration(마스터 EBS 구성), Core EBS Configuration(코어 EBS 구성) 또는 Task EBS Configuration(작업 EBS 구성) 중 하나를 선택한 후, 다음 예제와 비슷한 구성 세부 정보를 입력합니다.

```
{
  "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"
  }
}
```

참고

- EmrActivity

HttpProxy

HttpProxy로 자체 프록시를 구성하고 이를 통해 Task Runner에서 AWS Data Pipeline 서비스에 액세스할 수 있습니다. 이 정보를 사용하여 실행 중인 Task Runner를 구성할 필요가 없습니다.

TaskRunner의 HttpProxy 예제

다음 파이프라인 정의는 `HttpProxy` 객체를 보여줍니다.

```
{
  "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": []
}
```

구문

필수 필드	설명	슬롯 유형
hostname	클라이언트가 AWS 서비스에 연결할 때 사용할 프록시의 호스트.	문자열

필수 필드	설명	슬롯 유형
포트	클라이언트가 AWS 서비스에 연결할 때 사용할 프록시 호스트의 포트.	문자열

선택 필드	설명	슬롯 유형
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
*암호	프록시 비밀번호.	문자열
s3NoProxy	Amazon S3에 연결할 때 HTTP 프록시 비활성화	부울
사용자 이름	프록시의 사용자 이름입니다.	문자열
windowsDomain	NTLM 프록시의 Windows 도메인 이름입니다.	문자열
windowsWorkgroup	NTLM 프록시의 Windows 작업 그룹 이름입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열

시스템 필드	설명	슬롯 유형
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

사전 조건

다음은 AWS Data Pipeline 사전 조건 객체입니다.

Objects

- DynamoDBDataExists
- DynamoDBTableExists
- 존재함
- S3KeyExists
- S3PrefixNotEmpty
- ShellCommandPrecondition

DynamoDBDataExists

DynamoDB 테이블에 데이터가 존재하는지 확인하는 사전 조건입니다.

구문

필수 필드	설명	슬롯 유형
역할	사전 조건을 실행할 때 사용할 역할을 지정합니다.	문자열
tableName	점검할 DynamoDB 테이블입니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열

선택 필드	설명	슬롯 유형
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
preconditionTimeout	아직 충족되지 않은 사전 조건이 '실패'로 표시되기 시작하는 기간	Period
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period

선택 필드	설명	슬롯 유형
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
currentRetryCount	이번 시도에서 사전 조건을 시도한 횟수	문자열
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열

실행 시간 필드	설명	슬롯 유형
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
lastRetryTime	이번 시도 안에서 사전 조건을 시도한 마지막 횟수	문자열
노드	이 사전 조건이 실행 중인 노드	참조 객체. 예: "node": {"ref":"myRunnableObjectId"}
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

DynamoDBTableExists

DynamoDB 테이블이 존재하는지 확인하는 사전 조건입니다.

구문

필수 필드	설명	슬롯 유형
역할	사전 조건을 실행할 때 사용할 역할을 지정합니다.	문자열
tableName	점검할 DynamoDB 테이블입니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}

선택 필드	설명	슬롯 유형
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
preconditionTimeout	아직 충족되지 않은 사전 조건이 '실패'로 표시되기 시작하는 기간	Period
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn":

실행 시간 필드	설명	슬롯 유형
		{"ref":"myRunnable ObjectId"}
currentRetryCount	이번 시도에서 사전 조건을 시도한 횟수	문자열
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
lastRetryTime	이번 시도 안에서 사전 조건을 시도한 마지막 횟수	문자열
노드	이 사전 조건이 실행 중인 노드	참조 객체. 예: "node": {"ref":"myRunnableObjectId"}
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

존재함

데이터 노드 객체가 존재하는지 확인합니다.

> **Note**
>
> 대신에 시스템이 관리하는 사전 조건을 사용할 것을 권장합니다. 자세한 내용은 사전 조건 단원을 참조하십시오.

예제

다음은 이 객체 유형의 예제입니다. `InputData` 객체는 이 `Ready` 객체뿐 아니라 동일한 파이프라인 정의 파일에서 정의하려는 다른 객체를 참조합니다. `CopyPeriod`는 `Schedule` 객체입니다.

```
{
  "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"
}
```

구문

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}

선택 필드	설명	슬롯 유형
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
preconditionTimeout	아직 충족되지 않은 사전 조건이 '실패'로 표시되기 시작하는 기간	Period
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}

실행 시간 필드	설명	슬롯 유형
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
노드	이 사전 조건이 실행 중인 노드.	참조 객체. 예: "node": {"ref":"myRunnableObjectId"}
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandPrecondition

S3KeyExists

Amazon S3 데이터 노드에 키가 존재하는지 확인합니다.

예제

다음은 이 객체 유형의 예제입니다. `s3Key` 파라미터가 참조하는 키인 `s3://amzn-s3-demo-bucket/mykey`가 존재하면 사전 조건이 트리거됩니다.

```
{
"id" : "InputReady",
"type" : "S3KeyExists",
"role" : "test-role",
"s3Key" : "s3://amzn-s3-demo-bucket/mykey"
}
```

첫 번째 파이프라인이 종료되기를 기다리는 두 번째 파이프라인에서 `S3KeyExists`를 사전 조건으로 사용할 수도 있습니다. 그렇게 하려면 다음을 수행하세요.

1. 첫 번째 파이프라인의 완료가 끝날 때 Amazon S3에 파일을 씁니다.
2. 두 번째 파이프라인에서 `S3KeyExists` 사전 조건을 생성합니다.

구문

필수 필드	설명	슬롯 유형
역할	사전 조건을 실행할 때 사용할 역할을 지정합니다.	문자열
s3Key	Amazon S3 키.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	한 번 더 원격 작업을 완료하기 전의 제한 시간입니다. 이 시간을 설정하면 시작 후 설정된 시간 내에 완료되지 않는 원격 활동이 다시 시도됩니다.	Period
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maximumRetries	장애 시 시작되는 최대 시도 횟수입니다.	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}

선택 필드	설명	슬롯 유형
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
preconditionTimeout	아직 충족되지 않은 사전 조건이 '실패'로 표시되기 시작하는 기간.	Period
reportProgressTimeout	원격 작업에서 `reportProgress` 를 연속으로 호출하는 제한 시간입니다. 이것이 설정되면 지정 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주되어 재시도될 수 있습니다.	Period
retryDelay	두 번의 연속 시도 사이의 제한 시간 간격입니다.	Period

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
currentRetryCount	이번 시도에서 사전 조건을 시도한 횟수	문자열
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
lastRetryTime	이번 시도 안에서 사전 조건을 시도한 마지막 횟수	문자열
노드	이 사전 조건이 실행 중인 노드	참조 객체. 예: "node": {"ref":"myRunnableObjectId"}
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandPrecondition

S3PrefixNotEmpty

지정된 접두사(URI로 표시됨)가 있는 Amazon S3 객체가 존재하는지 확인하는 사전 조건입니다.

예제

다음은 필수, 선택 및 표현식 필드를 사용하는 이 객체 유형의 예입니다.

```
{
  "id" : "InputReady",
  "type" : "S3PrefixNotEmpty",
  "role" : "test-role",
  "s3Prefix" : "#{node.filePath}"
```

```
}
```

구문

필수 필드	설명	슬롯 유형
역할	사전 조건을 실행할 때 사용할 역할을 지정합니다.	문자열
s3Prefix	객체의 존재 여부를 확인하는 Amazon S3 접두사입니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}

선택 필드	설명	슬롯 유형
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
preconditionTimeout	아직 충족되지 않은 사전 조건이 '실패'로 표시되기 시작하는 기간	Period
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn":

실행 시간 필드	설명	슬롯 유형
		{"ref":"myRunnable ObjectId"}
currentRetryCount	이번 시도에서 사전 조건을 시도한 횟수	문자열
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
lastRetryTime	이번 시도 안에서 사전 조건을 시도한 마지막 횟수	문자열
노드	이 사전 조건이 실행 중인 노드.	참조 객체. 예: "node": {"ref":"myRunnableObjectId"}
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간.	DateTime
@scheduledStartTime	객체의 일정 시작 시간.	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandPrecondition

ShellCommandPrecondition

사전 조건으로 실행할 수 있는 Unix/Linux 셸 명령입니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "id" : "VerifyDataReadiness",
  "type" : "ShellCommandPrecondition",
  "command" : "perl check-data-ready.pl"
}
```

구문

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
명령	실행할 명령입니다. 이 값 및 연결된 파라미터는 Task Runner를 실행 중인 환경에서 작동해야 합니다.	문자열
scriptUri	파일을 다운로드하여 셸 명령으로 실행할 Amazon S3 URI 경로입니다. 단 하나의 scriptUri 또는 command 필드만 지정합니다. scriptUri는 파라미터를 사용할 수 없으며, command를 사용합니다.	문자열

선택 필드	설명	슬롯 유형
attemptStatus	원격 활동에서 가장 최근에 보고된 상태입니다.	문자열
attemptTimeout	원격 작업 완료의 제한 시간입니다. 이 필드를 설정하면 설정된 시작 시간 이내에 완료되지 않는 원격 활동을 재시도할 수 있습니다.	Period
failureAndRerunMode	종속 요소에 장애가 있거나 재시도될 때의 소비자 노드 거동을 설명합니다.	열거
lateAfterTimeout	파이프라인 시작 후 객체가 완료되어야 하는 경과 시간입니다. 스케줄 유형이 ondemand(으)로 설정되지 않은 경우에만 트리거됩니다.	Period
maximumRetries	장애 시 재시도 최대 횟수	Integer
onFail	현재 객체 장애 시 실행할 작업입니다.	참조 객체. 예: "onFail":{"ref":"myActionId"}

선택 필드	설명	슬롯 유형
onLateAction	객체가 아직 예약되지 않았거나 아직 완료되지 않은 경우에 트리거되어야 하는 작업입니다.	참조 객체. 예: "onLateAction":{"ref":"myActionId"}
onSuccess	현재 객체 성공 시 실행할 작업입니다.	참조 객체. 예: "onSuccess":{"ref":"myActionId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
preconditionTimeout	아직 충족되지 않은 사전 조건이 '실패'로 표시되기 시작하는 기간	Period
reportProgressTimeout	원격 작업에서 reportProgress를 연속으로 호출하는 제한 시간입니다. 이 필드를 설정하면 지정된 기간 동안 진행 상황을 보고하지 않는 원격 활동은 중단된 것으로 간주하고 재시도할 수 있습니다.	Period
retryDelay	두 번의 재시도 사이의 제한 시간 간격입니다.	Period
scriptArgument	셸 스크립트로 전달될 인수	문자열
stderr	명령에서 리디렉션된 시스템 오류 메시지를 수신하는 Amazon S3 경로입니다. `runsOn` 필드를 사용할 경우에는 활동을 실행할 리소스가 임시적이므로 Amazon S3 경로가 되어야 합니다. 그러나 `workerGroup` 필드를 지정할 경우에는 로컬 파일 경로가 허용됩니다.	문자열

선택 필드	설명	슬롯 유형
stdout	명령에서 리디렉션된 출력을 수신하는 Amazon S3 경로입니다. `runsOn` 필드를 사용할 경우에는 활동을 실행할 리소스가 임시적이므로 Amazon S3 경로가 되어야 합니다. 그러나 `workerGroup` 필드를 지정할 경우에는 로컬 파일 경로가 허용됩니다.	문자열

실행 시간 필드	설명	슬롯 유형
@activeInstances	현재 예약되어 있는 활성 인스턴스 객체의 목록입니다.	참조 객체. 예: "activeInstances": {"ref":"myRunnableObjectId"}
@actualEndTime	이 객체의 실행이 완료된 시간입니다.	DateTime
@actualStartTime	이 객체의 실행이 시작된 시간입니다.	DateTime
cancellationReason	이 객체가 취소된 경우의 cancellationReason입니다.	문자열
@cascadeFailedOn	객체 실패가 발생한 종속 체인에 대한 설명입니다.	참조 객체. 예: "cascadeFailedOn": {"ref":"myRunnableObjectId"}
emrStepLog	EMR 활동 시도 시에만 사용할 수 있는 EMR 단계 로그	문자열
errorId	이 객체가 실패한 경우의 errorId입니다.	문자열
errorMessage	이 객체가 실패한 경우의 errorMessage입니다.	문자열
errorStackTrace	이 객체가 실패한 경우의 오류 스택 트레이스입니다.	문자열

실행 시간 필드	설명	슬롯 유형
hadoopJobLog	EMR 기반 활동 시도 시에만 사용할 수 있는 하둡 작업 로그.	문자열
hostname	작업 시도를 선택한 클라이언트의 호스트 이름입니다.	문자열
노드	이 사전 조건이 실행 중인 노드	참조 객체. 예: "node": {"ref":"myRunnableObjectId"}
reportProgressTime	원격 활동에서 진행 상황을 보고한 가장 최근 시간입니다.	DateTime
@scheduledEndTime	객체의 일정 종료 시간	DateTime
@scheduledStartTime	객체의 일정 시작 시간	DateTime
@상태	이 객체의 상태입니다.	문자열
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열
@waitingOn	이 객체가 대기 중인 종속 요소 목록에 대한 설명입니다.	참조 객체. 예: "waitingOn":{"ref":"myRunnableObjectId"}

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- ShellCommandActivity
- 존재함

데이터베이스 수

다음은 AWS Data Pipeline 데이터베이스 객체입니다.

Objects

- JdbcDatabase
- RdsDatabase
- RedshiftDatabase

JdbcDatabase

JDBC 데이터베이스를 정의합니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "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"
}
```

구문

필수 필드	설명	슬롯 유형
connectionString	데이터베이스에 액세스할 JDBC 연결 문자열입니다.	문자열

필수 필드	설명	슬롯 유형
jdbcDriverClass	JDBC 연결 전에 로드할 드라이버 클래스입니다.	문자열
*암호	제공할 비밀번호.	문자열
사용자 이름	데이터베이스에 연결할 때 제공하는 사용자 이름.	문자열

선택 필드	설명	슬롯 유형
databaseName	연결할 논리 데이터베이스의 이름	문자열
jdbcDriverJarUri	데이터베이스 연결에 사용되는 JDBC 드라이버 JAR 파일의 Amazon S3 내 위치입니다. AWS Data Pipeline은 이 JAR 파일을 읽을 수 있는 권한이 있어야 합니다.	문자열
jdbcProperties	이 데이터베이스의 JDBC 연결에서 속성으로 설정될 A=B 형식 쌍.	문자열
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열

시스템 필드	설명	슬롯 유형
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

RdsDatabase

Amazon RDS 데이터베이스를 정의합니다.

> **Note**
>
> RdsDatabse는 Aurora를 지원하지 않습니다. Aurora 대신에 the section called "JdbcDatabase"을(를) 사용하세요.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "id" : "MyRdsDatabase",
  "type" : "RdsDatabase",
  "region" : "us-east-1",
  "username" : "user_name",
  "*password" : "my_password",
  "rdsInstanceId" : "my_db_instance_identifier"
}
```

Oracle 엔진의 경우 `jdbcDriverJarUri` 필드가 필수적이며 다음 드라이버를 지정할 수 있습니다. `http://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html` SQL Server 엔진의 경우 `jdbcDriverJarUri` 필드가 필수적이며 다음 드라이버를 지정할 수 있습니다. `https://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=11774` MySQL 및 PostgreSQL 엔진의 경우 `jdbcDriverJarUri` 필드는 선택 사항입니다.

구문

필수 필드	설명	슬롯 유형
*암호	제공할 비밀번호.	문자열
rdsInstanceId	DB 인스턴스의 `DBInstanceIdentifier` 속성.	문자열
사용자 이름	데이터베이스에 연결할 때 제공하는 사용자 이름.	문자열

선택 필드	설명	슬롯 유형
databaseName	연결할 논리 데이터베이스의 이름	문자열
jdbcDriverJarUri	데이터베이스 연결에 사용되는 JDBC 드라이버 JAR 파일의 Amazon S3 내 위치입니다. AWS Data Pipeline은 이 JAR 파일을 읽을 수 있는 권한이 있어야 합니다. MySQL 및 PostgreSQL 엔진은 이 필드가 지정되지 않은 경우 기본 드라이버로 사용되지만 이 필드를 사용하여 기본값을 다시 지정할 수 있습니다. Oracle 및 SQL Server 엔진은 이 필드가 필수 필드입니다.	문자열
jdbcProperties	이 데이터베이스의 JDBC 연결에서 속성으로 설정될 A=B 형식 쌍.	문자열
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
리전	데이터베이스가 존재하는 리전의 코드입니다. 예를 들어 us-east-1입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

RedshiftDatabase

Amazon Redshift 데이터베이스를 정의합니다. `RedshiftDatabase`은(는) 파이프라인에서 사용되는 데이터베이스의 속성을 나타냅니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "id" : "MyRedshiftDatabase",
  "type" : "RedshiftDatabase",
  "clusterId" : "myRedshiftClusterId",
  "username" : "user_name",
  "*password" : "my_password",
  "databaseName" : "database_name"
}
```

기본적으로 객체는 `clusterId` 필드가 필요한 Postgres 드라이버를 사용합니다. Amazon Redshift 드라이버를 사용하려면 `connectionString` 필드에 Amazon Redshift 콘솔에서 "jdbc:redshift:"로 시작하는 Amazon Redshift 데이터베이스 연결 문자열을 대신 지정하세요.

구문

필수 필드	설명	슬롯 유형
*암호	제공할 비밀번호.	문자열
사용자 이름	데이터베이스에 연결할 때 제공하는 사용자 이름.	문자열

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
clusterId	Amazon Redshift 클러스터가 생성될 때 사용자가 제공한 식별자입니다. 예를 들어, Amazon Redshift 클러스터의 엔드포인트가 mydb.example.us-east-1.redshift.amazonaws.com일 경우에 정확한 식별자는 `mydb`입니다. Amazon Redshift 콘솔에서는 클러스터 식별자 또는 클러스터 이름에서 이 값을 가져올 수 있습니다.	문자열
connectionString	파이프라인과 다른 계정이 소유한 Amazon Redshift 인스턴스에 연결할 JDBC 엔드포인트입니다. `connectionString` 및 `clusterId`를 둘 다 지정할 수 없습니다.	문자열

선택 필드	설명	슬롯 유형
databaseName	연결할 논리 데이터베이스의 이름.	문자열
jdbcProperties	이 데이터베이스의 JDBC 연결에서 속성으로 설정될 A=B 형식 페어.	문자열

선택 필드	설명	슬롯 유형
parent	슬롯을 상속해 올 현재 객체의 상위 객체입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
리전	데이터베이스가 존재하는 리전의 코드입니다. 예를 들어 us-east-1입니다.	열거

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

데이터 형식

다음은 AWS Data Pipeline 데이터 형식 객체입니다.

Objects

- CSV 데이터 형식
- 사용자 지정 데이터 형식
- DynamoDBDataFormat
- DynamoDBExportDataFormat
- RegEx 데이터 형식

- TSV 데이터 형식

CSV 데이터 형식

열 구분 기호가 쉼표이고 레코드 구분 기호가 줄 바꿈 문자인 위치의 쉼표로 구분된 데이터 형식입니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "id" : "MyOutputDataType",
  "type" : "CSV",
  "column" : [
    "Name STRING",
    "Score INT",
    "DateOfBirth TIMESTAMP"
  ]
}
```

구문

선택 필드	설명	슬롯 유형
열	이 데이터 노드가 설명하는 데이터의 필드별로 지정된 데이터 형식이 있는 열 이름입니다. 예: 호스트 이름 STRING 값이 여러 개인 경우에는 스페이스로 분리된 열 이름과 데이터 형식을 사용합니다.	문자열
escapeChar	파서에게 다음 문자를 무시하라고 지시하는 문자(예: "\")입니다.	문자열
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

사용자 지정 데이터 형식

특정 열 구분 기호, 레코드 구분 기호 및 이스케이프 문자의 조합으로 정의되는 사용자 정의 데이터 형식입니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "id" : "MyOutputDataType",
  "type" : "Custom",
  "columnSeparator" : ",",
  "recordSeparator" : "\n",
  "column" : [
    "Name STRING",
    "Score INT",
    "DateOfBirth TIMESTAMP"
  ]
}
```

구문

필수 필드	설명	슬롯 유형
columnSeparator	데이터 파일에서 열 끝을 나타내는 문자입니다.	문자열

선택 필드	설명	슬롯 유형
열	이 데이터 노드가 설명하는 데이터의 필드별로 지정된 데이터 형식이 있는 열 이름입니다. 예: 호스트 이름 STRING 값이 여러 개인 경우에는 스페이스로 분리된 열 이름과 데이터 형식을 사용합니다.	문자열
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
recordSeparator	데이터 파일에서 행의 끝을 나타내는 문자입니다(예: "\n"). 단일 문자만 지원됩니다.	문자열

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열

시스템 필드	설명	슬롯 유형
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

DynamoDBDataFormat

DynamoDB 테이블에 스키마를 적용하여 Hive 쿼리를 통해 액세스할 수 있습니다. `DynamoDBDataFormat`은(는) `HiveActivity` 객체와 `DynamoDBDataNode` 입력 및 출력과 함께 사용됩니다. `DynamoDBDataFormat`에서는 Hive 쿼리의 모든 열을 지정해야 합니다. Hive 쿼리에서 특정 열을 얼마나 유연하게 지정하는지에 대한 내용이나 Amazon S3 지원에 대한 내용은 DynamoDBExportDataFormat을(를) 참조하세요.

> **Note**
>
> DynamoDB Boolean 유형은 Hive Boolean 유형에 매핑하지 않습니다. 그러나 0 또는 1의 DynamoDB 정수 값을 Hive Boolean 유형에 매핑할 수 있습니다.

예제

다음 예제에서는 `DynamoDBDataFormat`을 사용하여 스키마를 `DynamoDBDataNode` 입력에 할당하는 방법을 보여주고 명명된 열을 사용해 `HiveActivity` 객체가 데이터를 액세스하고 `DynamoDBDataNode` 출력에 데이터를 복사합니다.

```
{
  "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"
```

```
    }
  ]
}
```

구문

선택 필드	설명	슬롯 유형
열	이 데이터 노드가 설명하는 데이터의 필드별로 지정된 데이터 형식이 있는 열 이름입니다. 예를 들어 `hostname STRING`입니다. 예: 여러 값의 경우, 스페이스로 분리된 열 이름과 데이터 형식을 사용합니다.	문자열
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}

실행 시간 필드	설명	슬롯 유형
@version	파이프라인 버전에서 객체를 생성하는 데 사용합니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류입니다.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID입니다.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

DynamoDBExportDataFormat

Amazon DynamoDB 테이블에 스키마를 적용하여 Hive 쿼리를 통해 액세스할 수 있습니다. `HiveCopyActivity` 객체와 `DynamoDBDataNode` 또는 `S3DataNode` 입력 및 출력에 `DynamoDBExportDataFormat`을 사용합니다. `DynamoDBExportDataFormat`에는 다음 장점이 있습니다.

- DynamoDB와 Amazon S3 지원을 모두 제공합니다.
- Hive 쿼리의 특정 열로 데이터를 필터링할 수 있음
- 스파스 스키마가 있는 경우에도 DynamoDB 의 모든 속성을 내보냄

> **Note**
>
> DynamoDB Boolean 유형은 Hive Boolean 유형에 매핑하지 않습니다. 그러나 0 또는 1의 DynamoDB 정수 값을 Hive Boolean 유형에 매핑할 수 있습니다.

예제

다음 예제에서는 타임스탬프에 기반하여 필터링을 하는 동안 `HiveCopyActivity`와 `DynamoDBExportDataFormat`을 사용하여 하나의 `DynamoDBDataNode`에서 다른 로 데이터를 복사하는 방법을 보여줍니다.

```
{
  "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"
    }
  ]
}
```

구문

선택 필드	설명	슬롯 유형
열	이 데이터 노드가 설명하는 데이터의 필드별로 지정된 데이터 형식이 있는 열 이름입니다. 예: 호스트이름 STRING	문자열
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

RegEx 데이터 형식

정규식에 의해 정의된 사용자 지정 데이터 형식입니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
```

```
  "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"
  ]
}
```

구문

선택 필드	설명	슬롯 유형
열	이 데이터 노드가 설명하는 데이터의 필드별로 지정된 데이터 형식이 있는 열 이름입니다. 예: 호스트 이름 STRING 값이 여러 개인 경우에는 스페이스로 분리된 열 이름과 데이터 형식을 사용합니다.	문자열
inputRegEx	S3 입력 파일의 구문을 분석할 정규식입니다. inputRegEx를 사용하여 파일에서 상대적으로 비정형 데이터의 열을 검색합니다.	문자열
outputFormat	inputRegEx로 검색하고, Java 포맷터 구문을 사용하여 %1$s %2$s로 참조되는 열 필드입니다.	문자열
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

TSV 데이터 형식

열 구분 기호가 탭 문자이고 레코드 구분 기호가 줄 바꿈 문자인 위치의 쉼표로 구분된 데이터 형식입니다.

예제

다음은 이 객체 유형의 예제입니다.

```
{
  "id" : "MyOutputDataType",
  "type" : "TSV",
  "column" : [
    "Name STRING",
    "Score INT",
    "DateOfBirth TIMESTAMP"
  ]
}
```

구문

선택 필드	설명	슬롯 유형
열	이 데이터 노드에 의해 설명된 데이터의 열 이름과 데이터 형식입니다. 예를 들어, "Name STRING"은 데이터 형식 STRING 필드에서 Name로 명명된 열을 가리킵니다. 쉼표로 구분되는 별도의 복수 열 이름 및 데이터 형식 페어(예 참조)입니다.	문자열
columnSeparator	한 열의 필드를 다음 열의 필드와 분리하는 문자입니다. 기본값은 '\t'입니다.	문자열
escapeChar	파서에게 다음 문자를 무시하라고 지시하는 문자(예: "\")입니다.	문자열
parent	슬롯을 상속해 올 현재 객체의 상위 객체입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
recordSeparator	레코드를 분리하는 문자입니다. 기본값은 '\n'입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열

시스템 필드	설명	슬롯 유형
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

작업

다음은 AWS Data Pipeline 작업 객체입니다.

Objects

- SnsAlarm
- Terminate

SnsAlarm

활동이 실패하거나 성공적으로 끝나면 Amazon SNS 알림 메시지를 전송합니다.

예제

다음은 이 객체 유형의 예제입니다. `node.input` 및 `node.output` 값은 해당 `onSuccess` 필드에서 이 객체를 참조하는 데이터 노드 또는 활동에서 나옵니다.

```
{
  "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}."
}
```

구문

필수 필드	설명	슬롯 유형
message	Amazon SNS 알림 본문입니다.	문자열

필수 필드	설명	슬롯 유형
역할	Amazon SNS 알람을 생성할 때 사용하는 IAM 역할입니다.	문자열
subject	Amazon SNS 알림 메시지의 제목 줄입니다.	문자열
topicArn	메시지의 대상 Amazon SNS 항목 ARN입니다.	문자열

선택 필드	설명	슬롯 유형
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}

실행 시간 필드	설명	슬롯 유형
노드	이 작업이 실행 중인 노드.	참조 객체. 예: "node":{"ref":"myRunnableObjectId"}
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

Terminate

보류 중이거나 완료되지 않은 활동, 리소스 또는 데이터 노드의 취소를 트리거하는 작업입니다. `lateAfterTimeout` 값으로 시작하지 않으면는 활동, 리소스 또는 데이터 노드를 CANCELLED 상태로 전환하려고 AWS Data Pipeline 시도합니다.

`onSuccess`, `OnFail` 또는 `onLateAction` 리소스를 포함하는 작업은 종료할 수 없습니다.

예제

다음은 이 객체 유형의 예제입니다. 이 예제에서 `MyActivity`의 `onLateAction` 필드는 작업 `DefaultAction1`에 대한 참조를 포함합니다. `onLateAction`에 대한 작업을 제공하는 경우 `lateAfterTimeout` 값을 제공하여 활동이 만료되었다고 간주된 파이프라인의 예약된 시작 이후 경과한 기간도 표시해야 합니다.

```
{
  "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"
}
```

구문

선택 필드	설명	슬롯 유형
parent	슬롯을 상속해 올 현재 객체의 상위 객체입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}

실행 시간 필드	설명	슬롯 유형
노드	이 작업이 실행 중인 노드.	참조 객체. 예: "node": {"ref":"myRunnableObjectId"}
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

일정

예약된 이벤트의 타이밍을 정의합니다(예: 활동 실행 시점).

> **Note**
>
> 일정의 시작 시간이 과거인 경우는 파이프라인을 AWS Data Pipeline 채우고 지정된 시작 시간부터 즉시 실행 예약을 시작합니다. 테스트/개발의 경우 상대적으로 짧은 간격을 사용합니

다. 그렇지 않으면는 해당 간격 동안 파이프라인의 모든 실행을 대기열에 추가하고 예약하려고 AWS Data Pipeline 시도합니다. scheduledStartTime는 파이프라인 활성화를 차단하여 파이프라인 구성 요소가 1일 이전인 경우 실수로 채우지 않도록 AWS Data Pipeline 합니다.

예제

다음은 이 객체 유형의 예제입니다. 이 예제에서는 매 시간 2012-09-01 00:00:00시에 시작되고 2012-10-01 00:00:00시에 종료되는 일정을 정의합니다. 처음 기간은 2012-09-01 01:00:00시에 종료됩니다.

```
{
  "id" : "Hourly",
  "type" : "Schedule",
  "period" : "1 hours",
  "startDateTime" : "2012-09-01T00:00:00",
  "endDateTime" : "2012-10-01T00:00:00"
}
```

다음 파이프라인은 FIRST_ACTIVATION_DATE_TIME에 시작하고 2014-04-25 22:00:00시까지 매 시간 실행됩니다.

```
{
     "id": "SchedulePeriod",
     "name": "SchedulePeriod",
     "startAt": "FIRST_ACTIVATION_DATE_TIME",
     "period": "1 hours",
     "type": "Schedule",
     "endDateTime": "2014-04-25T22:00:00"
   }
```

다음 파이프라인은 FIRST_ACTIVATION_DATE_TIME에 시작하고 매 시간 실행되어 세 번 작동한 후에 완료됩니다.

```
{
     "id": "SchedulePeriod",
     "name": "SchedulePeriod",
     "startAt": "FIRST_ACTIVATION_DATE_TIME",
     "period": "1 hours",
     "type": "Schedule",
     "occurrences": "3"
```

```
   }
```

다음 파이프라인은 2014-04-25 22:00:00에 시작하고 매 시간 실행되어 세 번 작동한 후에 종료됩니다.

```
{
     "id": "SchedulePeriod",
     "name": "SchedulePeriod",
     "startDateTime": "2014-04-25T22:00:00",
     "period": "1 hours",
     "type": "Schedule",
     "occurrences": "3"
   }
```

기본 객체를 사용하는 온디맨드

```
{
  "name": "Default",
  "resourceRole": "DataPipelineDefaultResourceRole",
  "role": "DataPipelineDefaultRole",
  "scheduleType": "ondemand"
}
```

명시적 일정 객체를 사용하는 온디맨드

```
{
  "name": "Default",
  "resourceRole": "DataPipelineDefaultResourceRole",
  "role": "DataPipelineDefaultRole",
  "scheduleType": "ondemand"
},
{
  "name": "DefaultSchedule",
  "type": "Schedule",
  "id": "DefaultSchedule",
  "period": "ONDEMAND_PERIOD",
  "startAt": "ONDEMAND_ACTIVATION_TIME"
},
```

다음 예제에서는 기본 객체에서 일정을 상속하는 방법, 해당 객체에 대해 명시적으로 일정을 설정하는 방법, 상위 참조에서 일정을 지정하는 방법을 보여줍니다.

기본 객체에서 상속된 일정

```
{
  "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!'"
    }
  ]
}
```

객체에서의 명시적 일정

```
{
  "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!'"
    }
  ]
}
```

상위 참조의 일정

```
{
  "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!'"
     }
  ]
}
```

구문

필수 필드	설명	슬롯 유형
기간	파이프라인을 실행해야 하는 빈도입니다. 형식은 "N [분|시|일|주|월]"이며, 여기서 N은 시간 지정자 중 하나가 뒤에 붙는 숫자입니다. 예를 들	Period

필수 필드	설명	슬롯 유형
	어, "15분"은 15분마다 파이프라인을 실행합니다. 최소 기간은 15분이며, 최대 기간은 3년입니다.	

필수 그룹(다음 중 하나를 제공해야 함)	설명	슬롯 유형
startAt	예약된 파이프라인 실행을 시작하는 날짜와 시간입니다. 유효 값은 FIRST_ACTIVATION_DATE_TIME이며, 이것은 온디맨드 파이프라인 생성을 위해 사용되지 않습니다.	열거
startDateTime	예약 실행이 시작되는 날짜 및 시간입니다. startDateTime 또는 startAt 중 하나만 사용해야 합니다.	DateTime

선택 필드	설명	슬롯 유형
endDateTime	예약 실행이 종료되는 날짜 및 시간입니다. startDateTime 또는 startAt 값 이후의 날짜와 시간이어야 합니다. 기본 행동은 파이프라인이 종료될 때까지 실행을 예약하는 것입니다.	DateTime
occurrences	활성화된 파이프라인을 실행하는 횟수입니다. occurrences를 endDateTime과 함께 사용할 수 없습니다.	Integer
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@firstActivationTime	객체 생성 시간입니다.	DateTime
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

유틸리티

다음 유틸리티 객체는 다른 파이프라인 객체를 구성합니다.

주제

- ShellScriptConfig
- EmrConfiguration
- 속성

ShellScriptConfig

활동을 사용하여 preActivityTaskConfig 및 postActivityTaskConfig에 대한 셸 스크립트를 실행합니다. 이 객체는 HadoopActivity, HiveActivity, HiveCopyActivity 및 PigActivity에 사용할 수 있습니다. 스크립트에 대해 인수 목록과 S3 URI를 지정합니다.

예제

인수를 사용하는 ShellScriptConfig:

```
{
   "id" : "ShellScriptConfig_1”,
   "name" : “prescript”,
   "type" : "ShellScriptConfig",
   "scriptUri": “s3://my-bucket/shell-cleanup.sh”,
   "scriptArgument" : ["arg1","arg2"]
 }
```

구문

이 객체에는 다음 필드가 포함됩니다.

선택 필드	설명	슬롯 유형
parent	슬롯을 상속해 올 현재 객체의 상위 객체입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
scriptArgument	shell 스크립트와 함께 사용할 인수 목록입니다.	문자열
scriptUri	다운로드하여 실행해야 하는 Amazon S3 내 스크립트 URI입니다.	문자열

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열

시스템 필드	설명	슬롯 유형
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

EmrConfiguration

EmrConfiguration 객체는 릴리스 4.0.0 또는 이상 버전을 사용하는 EMR 클러스터에 사용되는 구성입니다. 구성(목록)은 RunJobFlow API를 호출하는 파라미터입니다. Amazon EMR용 구성 API는 분류 및 속성을 사용합니다.는 해당하는 속성 객체와 함께 EmrConfiguration을 AWS Data Pipeline 사용하여 파이프라인 실행에서 시작된 EMR 클러스터의 하둡, Hive, Spark 또는 Pig와 같은 EmrCluster 애플리케이션을 구성합니다. 구성은 새로운 클러스터로 변경될 수 있기 때문에 기존 리소스에 EmrConfiguration 객체를 제공할 수 없습니다. 자세한 내용은 https://docs.aws.amazon.com/ElasticMapReduce/latest/ReleaseGuide/ 단원을 참조하십시오.

예제

다음 구성 객체는 `core-site.xml`으로 `io.file.buffer.size` 및 `fs.s3.block.size` 속성을 설정합니다.

```
[
   {
      "classification":"core-site",
      "properties":
      {
         "io.file.buffer.size": "4096",
         "fs.s3.block.size": "67108864"
      }
   }
]
```

해당 파이프라인 객체 정의는 `property` 필드의 속성 객체 목록 및 EmrConfiguration 객체를 사용합니다.

```
{
  "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"
    }
  ]
}
```

다음 예제는 hadoop-env 분류를 사용하여 Hadoop 환경을 설정하도록 사용된 중첩 구성입니다.

```
[
  {
    "classification": "hadoop-env",
```

```
    "properties": {},
    "configurations": [
      {
        "classification": "export",
        "properties": {
          "YARN_PROXYSERVER_HEAPSIZE": "2396"
        }
      }
    ]
  }
]
```

다음은 이 구성을 사용하는 해당 파이프라인 정의 객체입니다.

```
{
  "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"
    },
  ]
}
```

다음 예에서는 EMR 클러스터의 Hive 지정 속성을 수정합니다.

```
{
    "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"
        }
    ]
}
```

구문

이 객체에는 다음 필드가 포함됩니다.

필수 필드	설명	슬롯 유형
분류	구성에 대한 분류입니다.	문자열

선택 필드	설명	슬롯 유형
구성	이 구성의 하위 구성입니다.	참조 객체. 예: "configuration":{"ref":"myEmrConfigurationId"}
parent	슬롯을 상속할 현재 객체의 부모입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}
속성	구성 속성입니다.	참조 객체. 예: "property":{"ref":"myPropertyId"}

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- EmrCluster
- 속성
- Amazon EMR 릴리즈 가이드

속성

EmrConfiguration 객체에서 사용할 단일 키값 속성입니다.

예제

다음 파이프라인 정의는 EmrConfiguration 객체 및 launch an EmrCluster를 시작하는 해당 속성 객체를 보여줍니다.

```
{
  "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"
    }
  ]
}
```

구문

이 객체에는 다음 필드가 포함됩니다.

필수 필드	설명	슬롯 유형
키	키	문자열
값	값	문자열

선택 필드	설명	슬롯 유형
parent	슬롯을 상속해 올 현재 객체의 상위 객체입니다.	참조 객체. 예: "parent":{"ref":"myBaseObjectId"}

실행 시간 필드	설명	슬롯 유형
@version	객체와 함께 생성된 파이프라인 버전입니다.	문자열

시스템 필드	설명	슬롯 유형
@오류	잘못 형성된 객체를 설명하는 오류.	문자열
@pipelineId	이 객체가 속하는 파이프라인의 ID.	문자열
@sphere	객체의 타원 무늬는 수명 주기 내 위치를 나타냅니다. Component Objects는 Attempt Objects를 실행하는 Instance Objects를 야기합니다.	문자열

참고

- EmrCluster
- EmrConfiguration
- Amazon EMR 릴리즈 가이드

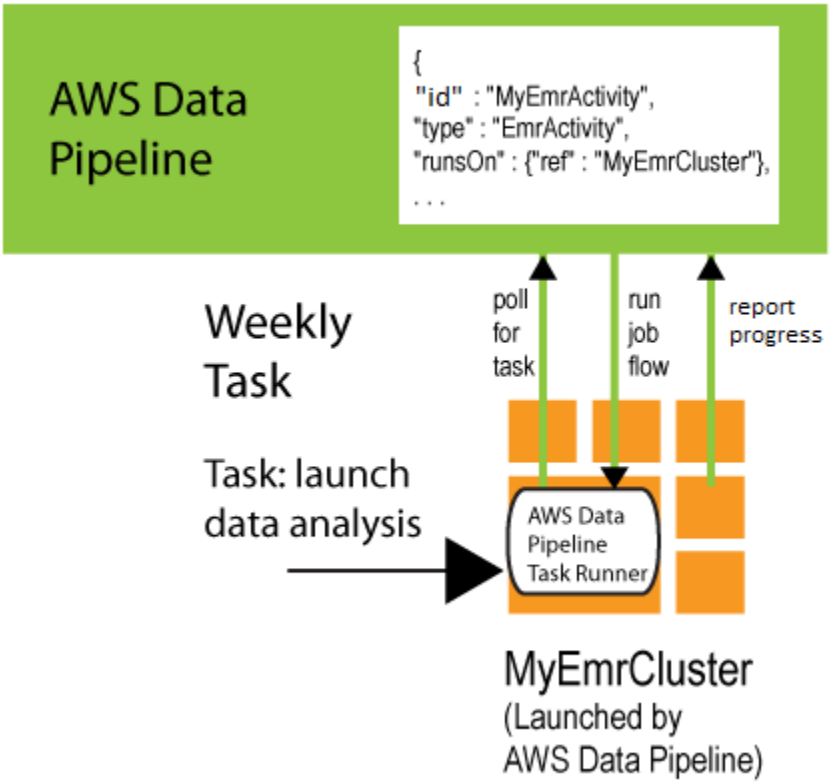
Task Runner로 작업하기

Task Runner는 예약된 작업을 AWS Data Pipeline 폴링하고 Amazon EC2 인스턴스, Amazon EMR 클러스터 또는 기타 컴퓨팅 리소스에서 실행하여 상태를 보고하는 작업 에이전트 애플리케이션입니다. 애플리케이션에 따라 다음을 선택할 수 있습니다.

- AWS Data Pipeline 가 하나 이상의 Task Runner 애플리케이션을 설치 및 관리할 수 있도록 허용합니다. 파이프라인이 활성화되면 활동 runsOn 필드에서 참조하는 기본 `Ec2Instance` 또는 `EmrCluster` 객체가 자동으로 생성됩니다.는 EC2 인스턴스 또는 EMR 클러스터의 마스터 노드에 Task Runner를 AWS Data Pipeline 설치합니다. 이 패턴에서는 대부분의 인스턴스 또는 클러스터 관리를 자동으로 수행할 AWS Data Pipeline 수 있습니다.
- 사용자가 관리하는 리소스에서 파이프라인 전부 또는 일부를 실행합니다. 잠재적 리소스에는 장시간 실행되는 Amazon EC2 인스턴스, Amazon EMR 클러스터 또는 물리적 서버가 포함됩니다. AWS Data Pipeline 웹 서비스와 통신할 수 있는 경우 거의 모든 곳에 작업 실행기(작업 실행기 또는 자체 디바이스의 사용자 지정 작업 에이전트일 수 있음)를 설치할 수 있습니다. 이 패턴에서, 사용되는 리소스와 그 관리 방식을 거의 완벽하게 제어한다고 가정할 때, Task Runner를 수동으로 설치하여 구성해야 합니다. 그럴려면 Task Runner를 사용하여 기존 리소스에서 작업 실행의 설명대로 이 단원에 나오는 절차를 사용하세요.

Task Runner AWS Data Pipeline관리형 리소스

에서 리소스를 시작하고 관리하면 AWS Data Pipeline웹 서비스는 파이프라인에서 작업을 처리하기 위해 해당 리소스에 Task Runner를 자동으로 설치합니다. 활동 객체의 `runs0n`필드용 전산 리소스(Amazon EC2 인스턴스 또는 Amazon EMR 클러스터)를 지정합니다. AWS Data Pipeline 가 이 리소스를 시작할 때 해당 리소스에 Task Runner를 설치하고 `runs0n` 필드가 이 리소스로 설정된 모든 활동 객체를 처리하도록 구성합니다. 가 리소스를 AWS Data Pipeline 종료하면 Task Runner 로그가 종료되기 전에 Amazon S3 위치에 게시됩니다.



예를 들어, 파이프라인에서 EmrActivity를 사용할 경우 runsOn 필드에서 EmrCluster 리소스를 지정합니다. 는 해당 활동을 AWS Data Pipeline 처리할 때 Amazon EMR 클러스터를 시작하고 마스터 노드에 Task Runner를 설치합니다. 그러면 이 Task Runner는 runsOn 필드가 EmrCluster 객체로 설정된 활동의 작업을 처리합니다. 다음 파이프라인 정의 발췌 부분은 두 객체 사이의 이 관계를 설명합니다.

```
{
  "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"
}
```

이 활동 실행에 대한 정보와 예제는 EmrActivity 단원을 참조하세요.

파이프라인에 여러 AWS Data Pipeline관리형 리소스가 있는 경우 Task Runner가 각 리소스에 설치되고 모두 처리할 작업에 AWS Data Pipeline 대해 폴링됩니다.

Task Runner를 사용하여 기존 리소스에서 작업 실행

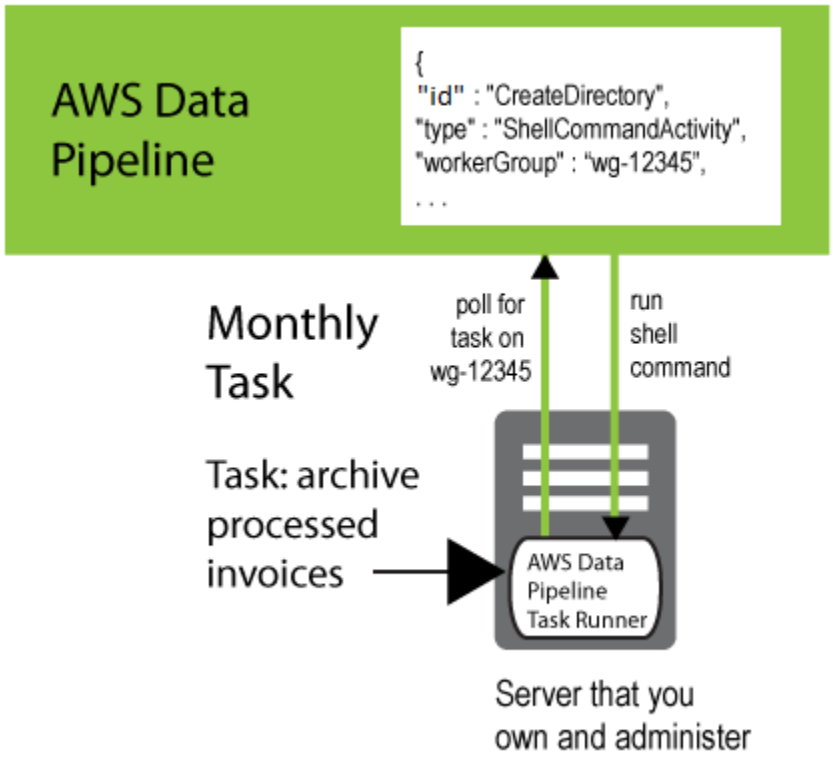
Amazon EC2 인스턴스, 물리적 서버 또는 워크스테이션과 같이 관리하는 컴퓨팅 리소스에 Task Runner를 설치할 수 있습니다. Task Runner는 AWS Data Pipeline 웹 서비스와 통신할 수 있는 경우 호환되는 모든 하드웨어 또는 운영 체제에 어디에나 설치할 수 있습니다.

이 접근 방식은 예를 들어를 사용하여 조직의 방화벽 내에 저장된 데이터를 처리하려는 경우에 유용 AWS Data Pipeline 할 수 있습니다. 로컬 네트워크의 서버에 Task Runner를 설치하면 로컬 데이터베이스에 안전하게 액세스한 다음 실행할 다음 작업을 AWS Data Pipeline 폴링할 수 있습니다. 가 파이프라인 처리를 AWS Data Pipeline 종료하거나 삭제하면 수동으로 종료할 때까지 Task Runner 인스턴스가 컴퓨팅 리소스에서 계속 실행됩니다. 파이프라인 실행이 완료된 후에도 Task Runner 로그는 유지됩니다.

사용자가 관리하는 리소스에서 Task Runner를 사용하려면 먼저 Task Runner를 다운로드한 후에 이 단원의 절차에 따라 그것을 전산 리소스에 설치해야 합니다.

> **Note**
>
> Linux, UNIX 또는 macOS에서만 Task Runner를 설치할 수 있습니다. Task Runner는 Windows 운영 체제에서 지원됩니다.
> Task Runner 2.0을 사용하는 데 필요한 최소 Java 버전은 1.7입니다.

처리해야 하는 파이프라인 활동에 설치한 Task Runner를 연결하려면 `workerGroup`필드를 객체에 추가하고, 해당 작업자 그룹 값에 폴링하도록 Task Runner를 구성합니다. Task Runner JAR 파일을 실행할 때 작업자 그룹 문자열을 파라미터(예: `--workerGroup=wg-12345`)로 전달하여 구성합니다.




```
{
  "id" : "CreateDirectory",
  "type" : "ShellCommandActivity",
  "workerGroup" : "wg-12345",
  "command" : "mkdir new-directory"
}
```

Task Runner 설치

이 섹션에서는 Task Runner와 그 필수 구성 요소를 설치하고 구성하는 방법을 설명합니다. 간단한 수동 과정으로 설치할 수 있습니다.

Task Runner를 설치하려면

1. Task Runner는 Java 버전 1.6 또는 1.8이 필요합니다. Java가 설치되었는지 그리고 실행 버전을 확인하려면 다음 명령을 사용합니다.

   ```
   java -version
   ```

   컴퓨터에 Java 1.6 또는 1.8이 설치되지 않은 경우, 이 버전 중 하나를 http://www.oracle.com/technetwork/java/index.html에서 다운로드하세요. Java를 다운로드하여 설치한 후 다음 단계를 진행합니다.

2. `TaskRunner-1.0.jar`을 https://s3.amazonaws.com/datapipeline-us-east-1/us-east-1/software/latest/TaskRunner/TaskRunner-1.0.jar에서 다운로드한 다음 대상 컴퓨팅 리소스의 폴더에 복사합니다. `EmrActivity` 작업을 실행하는 Amazon EMR 클러스터의 경우는 클러스터의 프라이머리 노드에 Task Runner를 설치합니다.

3. Task Runner를 사용하여 AWS Data Pipeline 웹 서비스에 연결하여 명령을 처리할 때 사용자는 데이터 파이프라인을 생성하거나 관리할 권한이 있는 역할에 프로그래밍 방식으로 액세스해야 합니다. 자세한 내용은 프로그래밍 방식 액세스 권한 부여 단원을 참조하십시오.

4. Task Runner는 HTTPS를 사용하여 AWS Data Pipeline 웹 서비스에 연결합니다. AWS 리소스를 사용하는 경우 적절한 라우팅 테이블 및 서브넷 ACL에서 HTTPS가 활성화되어 있는지 확인합니다. 방화벽 프록시를 사용하는 경우에는 포트 443이 열려 있어야 합니다.

(선택 사항) Amazon RDS에 대한 Task Runner 액세스 권한 부여

Amazon RDS를 통해 데이터베이스 보안 그룹(DB 보안 그룹)을 사용하여 DB 인스턴스에 대한 액세스를 제어할 수 있습니다. DB 보안 그룹은 DB 인스턴스에 대한 네트워크 액세스를 제어하는 방화벽처럼 작동합니다. 기본적으로 DB 인스턴스에 대한 네트워크 액세스는 해제되어 있습니다. Task Runner가 Amazon RDS 인스턴스에 액세스할 수 있도록 DB 보안 그룹을 수정해야 합니다. Task Runner는 실행 중인 인스턴스로부터 Amazon RDS 액세스 권한을 얻으므로 Amazon RDS 인스턴스에 추가하는 계정 및 보안 그룹은 Task Runner를 설치한 위치에 따라 달라집니다.

EC2-Classic에서 Task Runner에 대한 액세스를 부여하려면

1. Amazon RDS 콘솔을 엽니다.
2. 탐색 창에서 Instances를 선택한 다음 DB 인스턴스를 선택합니다.
3. 보안 및 네트워크에서 보안 그룹을 선택하면 이 DB 보안 그룹이 선택된 보안 그룹 페이지가 열립니다. DB 보안 그룹의 세부 정보 아이콘을 선택합니다.

4. [Security Group Details]에서 해당하는 연결 유형 및 세부 정보로 규칙을 생성합니다. 이러한 필드는 아래의 설명처럼 Task Runner가 실행 중인 위치에 따라 다릅니다.

   - `Ec2Resource`
     - 연결 유형: `EC2 Security Group`

       세부 정보: *`my-security-group-name`* (EC2 인스턴스에 생성한 보안 그룹 이름)
   - `EmrResource`
     - 연결 유형: `EC2 Security Group`

       세부 정보: `ElasticMapReduce-master`
     - 연결 유형: `EC2 Security Group`

       세부 정보: `ElasticMapReduce-slave`
   - 로컬 환경(온프레미스)
     - 연결 유형: `CIDR/IP`:

       세부 정보: *`my-ip-address`* (컴퓨터의 IP 주소 또는 컴퓨터에서 방화벽이 사용되는 경우는 네트워크의 IP 주소 범위)
5. 추가를 클릭합니다.

EC2-VPC에서 Task Runner에 대한 액세스를 부여하려면

1. Amazon RDS 콘솔을 엽니다.
2. 탐색 창에서 인스턴스를 선택합니다.
3. DB 인스턴스의 세부 정보 아이콘을 선택합니다. 보안 및 네트워크에서 보안 그룹 링크를 열면 Amazon EC2 콘솔로 이동합니다. 보안 그룹에 기존 콘솔 디자인을 사용할 경우에는 콘솔 페이지 상단에 표시된 아이콘을 선택하여 새 콘솔 디자인으로 전환합니다.
4. [Inbound] 탭에서 [Edit]와 [Add Rule]을 선택합니다. DB 인스턴스를 시작할 때 사용한 데이터베이스 포트를 지정합니다. 소스는 아래의 설명처럼 Task Runner가 실행 중인 위치에 따라 다릅니다.

   - `Ec2Resource`
     - *`my-security-group-id`* (EC2 인스턴스에 생성한 보안 그룹 ID)
   - `EmrResource`
     - *`master-security-group-id`* (`ElasticMapReduce-master` 보안 그룹의 ID)
     - *`slave-security-group-id`* (`ElasticMapReduce-slave` 보안 그룹의 ID)

- 로컬 환경(온프레미스)
  - *ip-address* (컴퓨터의 IP 주소 또는 컴퓨터에서 방화벽이 사용되는 경우는 네트워크의 IP 주소 범위)

5. 저장을 클릭합니다.

Task Runner 시작하기

Task Runner를 설치한 디렉터리로 설정된 새 명령 프롬프트 창에서 다음 명령으로 Task Runner를 시작합니다.

```
java -jar TaskRunner-1.0.jar --config ~/credentials.json --workerGroup=myWorkerGroup --
region=MyRegion --logUri=s3://amzn-s3-demo-bucket/foldername
```

`--config` 옵션은 사용자의 자격 증명 파일을 가리킵니다.

`--workerGroup` 옵션은 작업자 그룹 이름을 지정합니다. 이 이름은 처리할 작업의 파이프라인에 지정된 값과 같아야 합니다.

`--region` 옵션은 실행할 작업을 가져올 서비스 리전을 지정합니다.

`--logUri` 옵션은 Amazon S3 내 위치로 압축 로그를 보낼 때 사용됩니다.

Task Runner가 활성화되면 로그 파일이 터미널 창에 기록된 경로를 인쇄합니다. 다음은 예입니다.

```
Logging to /Computer_Name/.../output/logs
```

작업 실행기는 로그인 셸과 분리된 상태로 실행되어야 합니다. 터미널 애플리케이션을 사용하여 컴퓨터에 연결할 경우 nohup 또는 screen 같은 유틸리티를 사용하여 로그아웃 시 작업 실행기 애플리케이션이 남지 않도록 해야 합니다. 명령줄 옵션에 대한 자세한 내용은 Task Runner 구성 옵션을 참조하세요.

Task Runner 로깅 확인

Task Runner가 작동하는지 확인하는 가장 쉬운 방법은 로그 파일을 쓰고 있는지 확인하는 것입니다. Task Runner는 Task Runner가 설치된 디렉터리 `output/logs` 아래의 디렉터리에 시간별 로그 파일을 기록합니다. 파일 이름은 `Task Runner.log.YYYY-MM-DD-HH`이며, 여기서 HH는 00시부터 23시까지(UDT 기준) 실행됩니다. 스토리지 공간을 절약하기 위해 8시간 이상된 로그 파일은 GZip으로 압축됩니다.

Task Runner 스레드 및 사전 요구 사항

Task Runner는 각 작업, 활동 및 사전 조건에 스레드 풀을 사용합니다. 의 기본 설정은 2`--tasks`입니다. 즉, 작업 풀에서 두 개의 스레드가 할당되고 각 스레드가 새 작업에 대해 AWS Data Pipeline 서비스를 폴링합니다. 따라서 `--tasks`는 파이프라인 처리량 최적화에 도움을 주기 위해 사용할 수 있는 성능 튜닝 속성입니다.

사전 조건에 대한 파이프라인 재시도 논리는 Task Runner에서 발생합니다. 사전 조건 객체를 폴링하기 AWS Data Pipeline 위해 두 개의 사전 조건 스레드가 할당됩니다. Task Runner는 사전 요구 사항에 정의한 사전 조건 객체 retryDelay 및 preconditionTimeout 필드를 준수합니다.

사전 조건 폴링 타임아웃과 재시도 횟수가 감소하면 애플리케이션 성능 향상에 도움이 되는 경우가 많습니다. 그리고 장시간 실행 사전 조건이 있는 애플리케이션은 타임아웃 및 재시도 값은 증가해야 합니다. 사전 조건 객체에 관한 자세한 내용은 사전 조건 단원을 참조하세요.

Task Runner 구성 옵션

이것은 Task Runner를 시작할 때 명령줄에서 사용 가능한 구성 옵션입니다.

명령줄 파라미터	설명
`--help`	명령줄 도움말입니다. 예시: `Java -jar TaskRunner-1.0.jar --help`
`--config`	`credentials.json` 파일의 경로와 파일 이름입니다.
`--accessId`	요청 시 사용할 Task Runner의 AWS 액세스 키 ID입니다. `--accessID` 및 `--secretKey` 옵션은 파일 사용의 대안을 제공합니다. `credentials.json` 파일도 있는 경우에는 `--accessID` 및 `--secretKey` 옵션이 우선합니다.
`--secretKey`	요청 시 Task Runner가 사용할 AWS 보안 키입니다. 자세한 내용은 `--accessID` 단원을 참조하십시오.

명령줄 파라미터	설명
`--endpoint`	엔드포인트는 웹 서비스의 진입점인 URL입니다. 요청을 하는 리전의 AWS Data Pipeline 서비스 엔드포인트입니다. 선택 사항. 일반적으로 리전을 지정하는 것으로 충분하므로 엔드포인트를 설정할 필요가 없습니다. AWS Data Pipeline 리전 및 엔드포인트 목록은의 AWS Data Pipeline 리전 및 엔드포인트를 참조하세요AWS 일반 참조.
`--workerGroup`	Task Runner가 작업을 검색할 작업자 그룹의 이름입니다. 필수 사항입니다. Task Runner가 웹 서비스를 폴링할 때 사용자가 제공한 자격 증명과 `workerGroup` 값을 사용하여 검색할 작업(있는 경우)을 선택합니다. 사용자에게 의미가 있는 어떤 이름이든 사용할 수 있지만, 요구 사항은 문자열이 Task Runner와 이에 해당하는 파이프라인 활동 사이에 일치해야 합니다. 작업자 그룹 이름은 리전으로 제한됩니다. 다른 리전에 동일한 작업자 그룹 이름이 있어도 Task Runner는 항상 `--region`에 지정된 리전에서 작업을 가져옵니다.
`--taskrunnerId`	진행 상황을 보고할 때 사용할 작업 실행기 ID입니다. 선택 사항.
`--output`	로그 출력 파일의 Task Runner 디렉터리입니다. 선택 사항. 로그 파일은 Amazon S3로 밀려나갈 때까지 로컬 디렉터리에 저장됩니다. 이 옵션은 기본 디렉터리를 재정의합니다.

명령줄 파라미터	설명
`--region`	사용할 리전입니다. 선택사항이지만 항상 리전을 설정하는 것이 좋습니다. 리전을 지정하지 않으면 Task Runner가 기본 서비스 리전 `us-east-1` 에서 작업을 검색합니다. 그 밖에 지원되는 리전은 `eu-west-1` , `ap-northeast-1` , `ap-southeast-2` , `us-west-2` 입니다.
`--logUri`	Task Runner가 매시간 로그 파일을 백업할 Amazon S3 대상 경로입니다. Task Runner가 종료하면 로컬 디렉터리의 활성 로그가 Amazon S3 대상 폴더로 밀려나갑니다.
`--proxyHost`	Task Runner 클라이언트에 의해 사용되어 AWS 서비스에 연결하는 프록시의 호스트입니다.
`--proxyPort`	Task Runner 클라이언트에 의해 사용되어 AWS 서비스에 연결하는 프록시 호스트의 포트입니다.
`--proxyUsername`	프록시의 사용자 이름입니다.
`--proxyPassword`	프록시의 암호입니다.
`--proxyDomain`	NTLM 프록시의 Windows 도메인 이름입니다.
`--proxyWorkstation`	NTLM 프록시의 Windows 워크스테이션 이름입니다.

프록시와 함께 작업 실행기 사용

프록시 호스트를 사용할 경우에는 작업 실행기를 호출할 때 해당 configuration을 지정하거나 환경 변수 HTTPS_PROXY를 설정할 수 있습니다. 작업 실행기와 함께 사용되는 환경 변수는 AWS 명령줄 인터페이스에 사용되는 것과 동일한 구성을 수용합니다.

Task Runner 및 사용자 지정 AMI

파이프라인에 `Ec2Resource` 객체를 지정하면는 Task Runner를 설치하고 구성하는 AMI를 사용하여 EC2 인스턴스를 AWS Data Pipeline 생성합니다. 이 경우 PV 호환 인스턴스 유형이 필요합니다. 또는 Task Runner와 함께 사용자 지정 AMI를 생성한 후 `Ec2Resource`객체의 `imageId`필드를 사용하여 이 AMI의 ID를 지정할 수 있습니다. 자세한 내용은 Ec2Resource 단원을 참조하십시오.

사용자 지정 AMI가 Task Runner에 성공적으로 사용하려면 다음 요구 사항을 충족해야 합니다 AWS Data Pipeline .

- 인스턴스가 실행될 리전에 AMI를 생성합니다. 자세한 내용을 알아보려면 Amazon EC2 사용 설명서의 고유 AMI 생성을 참조하세요.
- 사용할 인스턴스 유형에서 AMI 가상화 유형이 지원되어야 합니다. 예를 들어, I2 및 G2 인스턴스 유형은 HVM AMI가 필요하며, T1, C1, M1 및 M2 인스턴스는 PV AMI가 필요합니다. 자세한 내용은 Amazon EC2 사용 설명서의 Linux AMI 가상화 유형을 참조하세요.
- 다음 소프트웨어를 설치합니다.
  - Linux
  - Bash
  - wget
  - unzip
  - Java 1.6 또는 1.8
  - cloud-init
- 이름이 `ec2-user`인 사용자 계정을 만들고 구성합니다.

문제 해결

문제가 있는 경우 AWS Data Pipeline가장 일반적인 증상은 파이프라인이 실행되지 않는 것입니다. 콘솔 및 CLI가 제공하는 데이터를 사용하여 문제를 확인하고 해결책을 찾을 수 있습니다.

내용

파이프라인 오류 찾기

AWS Data Pipeline 콘솔은 파이프라인의 상태를 시각적으로 모니터링하고 실패하거나 불완전한 파이프라인 실행과 관련된 오류를 쉽게 찾을 수 있는 편리한 도구입니다.

콘솔을 사용하여 실패했거나 완료되지 않은 실행에 관한 오류를 찾으려면

1. 파이프라인 목록 조회 페이지에서, 파이프라인 인스턴스 중 하나라도 상태 열이 완료가 아닐 경우 파이프라인에 충족해야 할 사전 조건이 있거나 실패한 것이므로 파이프라인 문제를 해결해야 합니다.
2. 파이프라인 목록 조회 페이지에서 인스턴스 파이프라인을 찾고 왼쪽에 있는 삼각형을 선택하여 세부 정보를 확장합니다.
3. 이 패널 하단의 예외 세부 정보 보기를 선택하면 선택한 인스턴스의 세부 정보가 표시된 인스턴스 요약 패널이 열립니다.
4. 인스턴스 요약 패널에서 인스턴스 옆에 있는 삼각형을 선택하여 인스턴스의 추가 세부 정보를 표시한 다음, 세부 정보를 선택하고 더 보기를 선택합니다. 선택한 인스턴스의 상태가 실패이면 세부 정보 상자에 오류 메시지 항목과 `errorStackTrace` 및 기타 정보가 표시됩니다. 이 정보를 파일에 저장할 수 있습니다. 확인을 선택합니다.
5. 지금 인스턴스 요약 창에서 시도 횟수를 선택하여 각 시도에 대한 세부 정보를 표시합니다.
6. 완료되지 않았거나 실패한 인스턴스에 대해 조치를 취하려면 인스턴스 옆의 확인란을 선택합니다. 그러면 작업이 활성화됩니다. 이제 작업(`Rerun|Cancel|Mark Finished`)을 선택합니다.

파이프라인에 서비스를 제공하는 Amazon EMR 클러스터를 식별합니다.

EMRCluster 또는가 EMRActivity 실패하고 AWS Data Pipeline 콘솔에서 제공하는 오류 정보가 명확하지 않은 경우 Amazon EMR 콘솔을 사용하여 파이프라인을 제공하는 Amazon EMR 클러스터를 식별할 수 있습니다. 이렇게 하면 Amazon EMR이 제공하는 로그를 쉽게 찾아 발생한 오류에 관해 자세히 볼 수 있습니다.

자세한 Amazon EMR 오류 정보를 보려면

1. AWS Data Pipeline 콘솔에서 파이프라인 인스턴스 옆의 삼각형을 선택하여 인스턴스 세부 정보를 확장합니다.

2. 실행 세부 정보 보기를 선택한 다음, 해당 구성요소 옆에 있는 삼각형을 선택합니다.
3. 세부 정보 열에서 더 보기...를 선택합니다. 해당 구성요소의 세부 정보가 나열된 정보 화면이 열립니다. 다음과 같이 화면에서 instanceParent 값을 찾아 복사합니다. `@EmrActivityId_xiFDD_2017-09-30T21:40:13`
4. Amazon EMR 콘솔로 이동하여 이름의 instanceParent 값과 일치하는 Amazon EMR 클러스터를 검색한 다음 디버그를 선택합니다.

   > **Note**
   >
   > 디버그 버튼이 작동하려면 EmrActivity `enableDebugging` 옵션을 `true`로 설정하고 `EmrLogUri` 옵션을 유효 경로로 설정해야 합니다.

5. 이제 어떤 Amazon EMR 클러스터에 파이프라인 장애를 일으키는 오류가 있는지 알았으니,Amazon EMR 개발자 안내서의 문제 해결 팁을 따르십시오.

파이프라인 상태 세부 정보 해석

AWS Data Pipeline 콘솔 및 CLI에 표시되는 다양한 상태 수준은 파이프라인 및 해당 구성 요소의 상태를 나타냅니다. 파이프라인 상태는 파이프라인 개요입니다. 자세한 정보를 확인하려면 개별 파이프라인 구성요소의 상태를 봐야 합니다. 콘솔에서 파이프라인을 클릭하거나 CLI를 사용하여 파이프라인 구성요소 세부 정보를 검색하면 됩니다.

상태 코드

ACTIVATING

EC2 인스턴스와 같은 구성 요소나 리소스가 시작 중입니다.

CANCELED

사용자가 실행하기 AWS Data Pipeline 전에 구성 요소를 취소했습니다. 이것은 이 구성 요소가 의존하는 다른 구성 요소나 리소스에서 오류가 발생할 경우, 자동으로 일어날 수 있습니다.

CASCADE_FAILED

이 구성 요소나 리소스는 하나 이상의 종속성에서 비롯된 캐스케이드 오류로 인해 취소되었지만 해당 구성 요소가 오류의 근원이 아닐 수 있습니다.

DEACTIVATING

파이프라인이 비활성화되는 중입니다.

FAILED

구성 요소 또는 리소스에 오류가 발생했고 작동이 멈췄습니다. 구성 요소 또는 리소스에 오류가 발생하면 이 구성 요소나 리소스에 의존하는 다른 구성 요소로 취소와 오류가 캐스케이딩될 수 있습니다.

FINISHED

구성 요소가 할당된 작업을 완료했습니다.

INACTIVE

파이프라인이 비활성화되었습니다.

PAUSED

구성 요소가 일시 중지되었고 현재 작업을 수행하고 있지 않습니다.

PENDING

파이프라인이 처음 활성화될 준비가 되었습니다.

RUNNING

리소스가 실행 중이고 작업을 수신할 준비가 되었습니다.

SCHEDULED

이 리소스는 실행되도록 예약되어 있습니다.

SHUTTING_DOWN

리소스가 작업을 성공적으로 완료한 후 종료 중입니다.

SKIPPED

구성 요소가 현재 일정 이후의 타임스탬프를 사용하여 파이프라인이 활성화된 후 실행 간격을 건너뛰었습니다.

TIMEDOUT

리소스가 terminateAfter 임계값을 초과하여 중지되었습니다 AWS Data Pipeline. 리소스가 이 상태에 도달하면 AWS Data Pipeline 이 리소스의 actionOnResourceFailure, retryDelay 및 retryTimeout 값을 무시합니다. 이 상태는 리소스에만 적용됩니다.

VALIDATING

파이프라인 정의를 검증하는 중입니다 AWS Data Pipeline.

WAITING_FOR_RUNNER

구성 요소가 작업자 클라이언트가 작업 항목을 검색하기를 기다리고 있습니다. 구성 요소와 작업자 클라이언트 관계는 해당 구성 요소가 정의한 runsOn 또는 workerGroup 필드에 의해 제어됩니다.

WAITING_ON_DEPENDENCIES

구성 요소가 작업 수행 전에 기본 및 사용자 구성 사전 조건이 충족되었는지 확인하는 중입니다.

오류 로그 찾기

이 섹션에서는가 AWS Data Pipeline 작성하는 다양한 로그를 찾는 방법을 설명합니다.이 로그를 사용하여 특정 실패 및 오류의 원인을 확인할 수 있습니다.

파이프라인 로그

영구적인 위치에 로그 파일을 생성하도록 파이프라인을 구성할 것을 권장하며, 이에 대한 다음 예제에서 pipelineLogUri필드를 파이프라인의 Default객체에서 사용하여 모든 파이프라인 구성요소가 기본적으로 Amazon S3 로그 위치를 사용하게 합니다(특정 파이프라인 구성요소에 로그 위치를 구성하여 이것을 다시 정의할 수 있음).

> **Note**
>
> Task Runner는 기본적으로 다른 위치에 그 로그를 저장하는데, 이것은 파이프라인이 끝나고 Task Runner를 실행하는 인스턴스가 종료될 때는 사용할 수 없습니다. 자세한 내용은 Task Runner 로깅 확인 단원을 참조하십시오.

파이프라인 JSON 파일의 AWS Data Pipeline CLI를 사용하여 로그 위치를 구성하려면 다음 텍스트로 파이프라인 파일을 시작합니다.

```
{ "objects": [
{
  "id":"Default",
  "pipelineLogUri":"s3://amzn-s3-demo-bucket/error_logs"
},
...
```

파이프라인 로그 디렉터리가 구성되면 Task Runner가 디렉터리에 로그 사본을 생성하는데, Task Runner 로그에 관한 앞 단원에서 설명한 것과 동일한 형식과 파일 이름을 사용합니다.

Hadoop 작업 및 Amazon EMR 단계 로그

HadoopActivity, HiveActivity, PigActivity등의 Hadoop 기반 활동으로 실행 시간 슬롯, hadoopJobLog에서 반환되는 위치에서 Hadoop 작업 로그를 볼 수 있습니다. EmrActivity에는 자체 로깅 기능이 있으며, 이러한 로그는 Amazon EMR이 선택하고 실행 시간 슬롯, emrStepLog가 반환하는 위치를 사용하여 저장됩니다. 자세한 내용은 Amazon EMR 개발자 안내서의 로그 파일 보기를 참조하십시오.

공통 문제 해결

이 주제에서는 문제의 다양한 증상과 AWS Data Pipeline 이를 해결하기 위한 권장 단계를 제공합니다.

내용

- EMR 클러스터 실패와 오류: 요청에 포함된 보안 토큰이 잘못되었음
- 리소스에 대한 액세스 권한 부족
- 상태 코드: 400 오류 코드: PipelineNotFoundException
- 파이프라인 생성으로 보안 토큰 오류 발생
- 콘솔에서 파이프라인 세부 정보를 볼 수 없음
- 원격 실행기 오류 상태 코드: 404, AWS Service: Amazon S3
- 액세스 거부 - 기능을 실행할 권한이 없음 datapipeline
- 이전 버전의 Amazon EMR AMI가 대용량 CSV 파일의 거짓 데이터를 생성할 수도 있음
- AWS Data Pipeline 제한 증가

보류 상태에서 멈춘 파이프라인

PENDING 상태에 멈춰있는 것처럼 보이는 파이프라인은 파이프라인이 아직 활성화되지 않았거나 파이프라인 정의의 오류로 인해 활성화에 실패했음을 나타냅니다. AWS Data Pipeline CLI를 사용하여 파이프라인을 제출하거나 AWS Data Pipeline 콘솔을 사용하여 파이프라인을 저장하거나 활성화하려고 할 때 오류가 발생하지 않았는지 확인합니다. 그리고 파이프라인에 유효한 정의가 있는지 확인합니다.

CLI를 사용하여 화면에서 파이프라인 정의를 보려면

```
aws datapipeline --get-pipeline-definition --pipeline-id df-EXAMPLE_PIPELINE_ID
```

파이프라인 정의가 완료되었는지 확인하고, 닫는 괄호를 확인하고, 필요한 쉼표를 확인하고, 누락된 참조가 있는지 확인하고, 기타 구문 오류를 확인합니다. JSON 파일의 구문을 눈으로 확인할 수 있는 텍스트 편집기를 사용하는 것이 가장 좋습니다.

실행기 대기 상태에서 멈춘 파이프라인 구성요소

파이프라인이 SCHEDULED 상태이고 하나 이상의 작업이 WAITING_FOR_RUNNER 상태에 멈춰있는 것으로 보일 경우에는 해당 작업의 runsOn 또는 workerGroup 필드 설정한 값이 유효한지 확인합니다. 두 값이 모두 비었거나 없는 경우에는 작업과 작업을 실행할 작업자 사이가 연결되지 않았기 때문에 작업을 시작할 수 없습니다. 이 상황은 작업은 정의했지만 이 작업을 실행할 컴퓨터는 정의하지 않은 경우입니다. 가능하다면 파이프라인 구성요소에 할당된 workerGroup 값이 Task Runner로 구성한 workerGroup 값과 이름과 대소문자가 정확히 동일한지 확인합니다.

> **Note**
>
> workerGroup이 있을 때 runsOn 값을 제공하면 workerGroup이 무시됩니다.

이 문제의 또 다른 잠재적 원인은 Task Runner에 제공된 엔드포인트 및 액세스 키가 AWS Data Pipeline CLI 도구가 설치된 AWS Data Pipeline 콘솔 또는 컴퓨터와 동일하지 않기 때문입니다. 눈에 보이는 오류가 없는 새 파이프라인을 생성했어도 보안 인증의 차이 때문에 Task Runner가 잘못된 위치를 폴링하거나, 위치는 맞지만 파이프라인 정의로 지정된 작업을 찾아 실행할 권한이 부족한 위치로 폴링합니다.

WAITING_ON_DEPENDENCIES 상태에서 멈춘 파이프라인 구성요소

파이프라인이 SCHEDULED 상태이고 하나 이상의 작업이 WAITING_ON_DEPENDENCIES 상태에 멈춰있는 것으로 보일 경우에는 파이프라인의 초기 사전 조건이 충족되었는지 확인합니다. 논리 체인에서 첫 번째 객체의 사전 조건이 충족되지 않으면 이 첫 번째 객체에 좌우되는 객체는 WAITING_ON_DEPENDENCIES 상태 밖으로 이동할 수 없습니다.

예를 들어, 다음 파이프라인 정의 발췌 부분을 생각해보겠습니다. 이 경우에 InputData 객체의 사전 조건 'Ready'는 InputData 객체가 완료되기 전에 데이터가 존재해야 한다는 것을 지정합니다. 데이터가 존재하지 않을 경우 InputData 객체는 WAITING_ON_DEPENDENCIES 상태를 유지하며 경로 필드로 지정한 데이터가 제공될 때까지 기다립니다. InputData에 의존하는 객체도 WAITING_ON_DEPENDENCIES 상태를 유지하며 InputData 객체가 FINISHED 상태가 될 때까지 기다립니다.

```
{
    "id": "InputData",
    "type": "S3DataNode",
    "filePath": "s3://elasticmapreduce/samples/wordcount/wordSplitter.py",
    "schedule":{"ref":"MySchedule"},
    "precondition": "Ready"
},
{
    "id": "Ready",
    "type": "Exists"
...
```

그리고 객체가 데이터에 액세스할 수 있는 권한이 있는지도 확인합니다. 앞 예제에서 자격 증명 필드의 정보에 경로 필드에 지정된 데이터에 액세스할 수 있는 권한이 없다면 데이터가 존재해도 경로 필드로

지정된 데이터에 액세스하지 못하기 때문에 InputData 객체가 WAITING_ON_DEPENDENCIES 상태에 멈추게 됩니다.

Amazon S3와 통신하는 리소스에 이와 연결된 공용 IP 주소가 없을 수도 있습니다. 예를 들어, 퍼블릭 서브넷의 Ec2Resource는 이와 연결된 퍼블릭 IP 주소가 있어야 합니다.

끝으로, 특정 상황에서는 연결된 활동의 예정 시작 시간보다 훨씬 더 먼저 리소스 인스턴스가 WAITING_ON_DEPENDENCIES 상태에 도달할 수 있는데, 이 경우 리소스 또는 활동이 실패하는 것처럼 보일 수 있습니다.

예약한 시간에 실행이 시작되지 않음

일정 간격이 시작할 때 작업이 시작할지(Cron 스타일 일정 유형) 또는 일정 간격이 끝날 때 작업이 시작할지(시계열 일정 유형)를 결정하는 정확한 일정 유형을 선택했는지 확인합니다.

그리고 일정 객체에서 날짜를 정확하게 지정했는지 그리고 startDateTime 및 endDateTime 값이 다음 예제처럼 UTC 형식인지 확인합니다.

```
{
    "id": "MySchedule",
    "startDateTime": "2012-11-12T19:30:00",
    "endDateTime":"2012-11-12T20:30:00",
    "period": "1 Hour",
    "type": "Schedule"
},
```

잘못된 순서로 실행되는 파이프라인 구성요소

파이프라인 구성요소의 시작 및 종료 시간이 잘못된 순서로 실행되거나 예상과 다른 시퀀스로 실행된다고 느껴질 수 있습니다. 시작 시각에 사전 조건이 충족되면 파이프라인 구성요소가 동시에 실행될 수 있다는 것을 알아야 합니다. 다시 말해, 파이프라인 구성요소는 기본적으로 순서대로 실행되지 않습니다. 특정 실행 순서가 필요할 경우에는 사전 조건과 dependsOn 필드로 실행 순서를 제어해야 합니다.

정확한 파이프라인 구성요소의 참조로 채워진 dependsOn 필드를 사용해야 하며 구성요소 사이에 필요한 모든 포인터가 있어야만 원하는 순서를 구현할 수 있습니다.

EMR 클러스터 실패와 오류: 요청에 포함된 보안 토큰이 잘못되었음

IAM 역할, 정책 및 신뢰 관계가 에 대한 IAM 역할 AWS Data Pipeline의 설명과 같은지 확인합니다.

리소스에 대한 액세스 권한 부족

IAM 역할에 설정한 권한에 따라가 EMR 클러스터 및 EC2 인스턴스에 AWS Data Pipeline 액세스하여 파이프라인을 실행할 수 있는지 여부가 결정됩니다. 그리고 IAM은 더 나아가 사용자를 대신하여 리소스를 생성할 수 있는 신뢰 관계 개념을 제공합니다. 예를 들어 EC2 인스턴스를 사용하여 명령을 실행하여 데이터를 이동하는 파이프라인을 생성할 때이 EC2 인스턴스를 프로비저닝할 AWS Data Pipeline 수 있습니다. 특히 수동으로 액세스할 수 있지만 액세스할 수 AWS Data Pipeline 없는 리소스와 관련된 문제가 발생하는 경우에 설명된 대로 IAM 역할, 정책 및 신뢰 관계를 확인합니다에 대한 IAM 역할 AWS Data Pipeline.

상태 코드: 400 오류 코드: PipelineNotFoundException

이 오류는 IAM 기본 역할에가 올바르게 작동하는 AWS Data Pipeline 데 필요한 권한이 없을 수 있음을 의미합니다. 자세한 내용은 에 대한 IAM 역할 AWS Data Pipeline 단원을 참조하십시오.

파이프라인 생성으로 보안 토큰 오류 발생

파이프라인 생성을 시도할 때 다음 오류가 수신됩니다.

'pipeline_name'의 파이프라인을 생성하지 못했습니다. 오류: UnrecognizedClientException - 요청에 포함된 보안 토큰이 잘못되었습니다.

콘솔에서 파이프라인 세부 정보를 볼 수 없음

AWS Data Pipeline 콘솔 파이프라인 필터는 파이프라인이 제출된 시기와 관계없이 파이프라인의 예약된 시작일에 적용됩니다. 과거에 발생하는 예약된 시작일을 사용하여 새 파이프라인을 제출할 수 있는데, 이 경우 기본 날짜 필터가 표시되지 않을 수 있습니다. 파이프라인 세부 정보를 보려면 날짜 필터를 변경하여 예약된 파이프라인 시작일이 날짜 범위 필터 안에 포함시켜야 합니다.

원격 실행기 오류 상태 코드: 404, AWS Service: Amazon S3

이 오류는 Task Runner가 Amazon S3에 있는 사용자 파일에 액세스하지 못했음을 의미합니다. 다음을 확인합니다.

- 자격 증명을 정확하게 설정했는지
- 액세스하려는 Amazon S3 버킷이 존재합니다.
- Amazon S3 버킷에 액세스할 수 있는 권한이 있습니다.

액세스 거부 - 기능을 실행할 권한이 없음 datapipeline

Task Runner 로그에 다음과 같이 유사한 오류가 있을 수 있습니다.

- ERROR 상태 코드: 403
- AWS 서비스: DataPipeline
- AWS 오류 코드: AccessDenied
- AWS 오류 메시지: 사용자: arn:aws:sts::XXXXXXXXXXXX:federated-user/i-XXXXXXXX는 datapipeline:PollForTask를 실행할 권한이 없습니다.

> **Note**
>
> 이 오류 메시지에서는 PollForTask가 다른 AWS Data Pipeline 권한의 이름으로 바뀔 것입니다.

이 오류 메시지는 지정한 IAM 역할에 상호 작용하는 데 필요한 추가 권한이 필요함을 나타냅니다 AWS Data Pipeline. 사용자의 IAM 역할 정책에 다음 라인이 포함되는지 확인합니다. 여기서 PollForTask는 사용자가 추가해야 할 권한 이름으로 바뀝니다(모든 권한을 부여하려면 * 사용). 새 IAM 역할을 생성하고 정책을 적용하는 방법에 대한 자세한 내용은 IAM 사용 가이드의 IAM 정책 관리를 참조하십시오.

```
{
"Action": [ "datapipeline:PollForTask" ],
"Effect": "Allow",
"Resource": ["*"]
}
```

이전 버전의 Amazon EMR AMI가 대용량 CSV 파일의 거짓 데이터를 생성할 수도 있음

Amazon EMR AMIs 3.9 이전(3.8 이하)에서는 사용자 지정 InputFormat을 AWS Data Pipeline 사용하여 MapReduce 작업에 사용할 CSV 파일을 읽고 씁니다. 이것은 서비스 Amazon S3에서 테이블을 스테이징할 때 사용됩니다. 이 InputFormat에서 대용량 CSV 파일의 기록을 읽을 때 정확하게 복사되지 않는 테이블이 생성될 수 있다는 문제가 발견되었습니다. 이 문제는 이후의 Amazon EMR 릴리스에서 해결되었습니다. Amazon EMR AMI 3.9 또는 Amazon EMR 릴리스 4.0.0 이상을 사용하시기 바랍니다.

AWS Data Pipeline 제한 증가

경우에 따라 특정 AWS Data Pipeline 시스템 제한을 초과할 수 있습니다. 예를 들어, 기본 파이프라인 한계는 각각 50개의 객체가 있는 파이프라인 20개입니다. 이 한계보다 많은 파이프라인이 필요하다고 생각될 경우에는 여러 파이프라인을 병합하여 파이프라인 수는 줄이고 각 객체는 늘려보십시오. AWS Data Pipeline 한도에 대한 자세한 내용은 AWS Data Pipeline 제한 단원을 참조하십시오. 단, 파이프라인 병합 기술을 사용하여 한계 부근에서 작업할 수 없는 경우에는 이 데이터 파이프라인 제한 증가 양식을 사용하여 용량 증가를 요청합니다.

AWS Data Pipeline 제한

모든 사용자에게 용량이 있는지 확인하기 위해는 할당할 수 있는 리소스와 리소스를 할당할 수 있는 속도에 대한 제한을 AWS Data Pipeline 부과합니다.

내용

계정 제한

단일 AWS 계정에는 다음 제한이 적용됩니다. 추가 용량이 필요할 경우 Amazon Web Services Support Center 요청 양식을 사용하여 용량을 늘릴 수 있습니다.

속성	Limit	조정 가능
파이프라인 수	100	예
파이프라인당 객체 수	100	예
객체당 활성 인스턴스 수	5	예
객체당 필드 수	50	아니요
필드 이름 또는 식별자당 UTF8 바이트 수	256	아니요
필드당 UTF8 바이트 수	10,240	아니요
객체당 UTF8 바이트 수	15,360(필드 이름 포함)	아니요
객체에서 인스턴스의 생성 속도	5분당 1	아니요

속성	Limit	조정 가능
파이프라인 활동의 재시도	작업당 5	아니요
재시도 간 최소 지연	2 minutes	아니요
최소 일정 간격	15분	아니요
단일 객체에 롤업할 수 있는 최대 수	32	아니요
Ec2Resource 객체당 최대 EC2 인스턴스 수	1	아니요

웹 서비스 호출 제한

AWS Data Pipeline 는 웹 서비스 API를 호출할 수 있는 속도를 제한합니다. 이러한 제한은 콘솔, CLI 및 Task Runner와 같이 사용자를 대신하여 웹 서비스 API를 호출하는 AWS Data Pipeline 에이전트에도 적용됩니다.

단일 AWS 계정에는 다음 제한이 적용됩니다. 따라서 사용자를 포함해 계정 총 사용량이 이 한계를 초과하지 못합니다.

버스트 속도를 사용하면 비활성 기간에 웹 서비스 호출을 저장했다가 단시간에 모두 소비할 수 있습니다. 예를 들어, CreatePipeline의 일반 호출 속도는 5초당 1회입니다. 30초 동안 서비스를 호출하지 않으면 6개 호출이 저장됩니다. 이후 웹 서비스를 1초에 6회 호출할 수 있을 것입니다. 이것은 버스트 제한 미만이고, 평균 호출이 일반 속도 제한으로 유지되기 때문에 호출이 정체되지 않습니다.

속도 제한과 버스트 제한을 초과하면 웹 서비스 호출이 안 되고 조절 예외가 반환됩니다. 작업자의 기본 구현인 Task Runner는 전송률 조절 예외로 인해 실패한 API 호출을 자동으로 재시도합니다. Task Runner에는 백오프 기능이 있어서 이후에 API를 호출하려는 시도가 점점 더 긴 간격으로 발생합니다. 작업자에게 알릴 경우 유사한 재시도 논리를 실행하는 것이 좋습니다.

이러한 제한은 개별 AWS 계정에 적용됩니다.

API	일반 속도 제한	버스트 제한
ActivatePipeline	초당 호출 1회	100회 호출

API	일반 속도 제한	버스트 제한
CreatePipeline	초당 호출 1회	100회 호출
DeletePipeline	초당 호출 1회	100회 호출
DescribeObjects	초당 호출 2회	100회 호출
DescribePipelines	초당 호출 1회	100회 호출
GetPipelineDefinition	초당 호출 1회	100회 호출
PollForTask	초당 호출 2회	100회 호출
ListPipelines	초당 호출 1회	100회 호출
PutPipelineDefinition	초당 호출 1회	100회 호출
QueryObjects	초당 호출 2회	100회 호출
ReportTaskProgress	초당 호출 10회	100회 호출
SetTaskStatus	초당 호출 10회	100회 호출
SetStatus	초당 호출 1회	100회 호출
ReportTaskRunnerHeartbeat	초당 호출 1회	100회 호출
ValidatePipelineDefinition	초당 호출 1회	100회 호출

조정 고려 사항

AWS Data Pipeline 는 많은 수의 동시 작업을 수용하도록 확장되며 대규모 워크로드를 처리하는 데 필요한 리소스를 자동으로 생성하도록 구성할 수 있습니다. 이렇게 자동으로 생성된 리소스는 사용자가 제어할 수 있으며, AWS 계정 리소스 제한을 기준으로 계수할 수 있습니다. 예를 들어 데이터를 처리하기 위해 20노드 Amazon EMR 클러스터를 자동으로 생성 AWS Data Pipeline 하도록를 구성하고 AWS 계정에 EC2 인스턴스 제한이 20으로 설정된 경우 사용 가능한 채우기 리소스가 실수로 소진될 수 있

습니다. 따라서 디자인할 때 이러한 리소스 제한을 고려하거나 계정 한계를 알맞게 늘리는 것이 좋습니다.

추가 용량이 필요할 경우 Amazon Web Services Support Center 요청 양식을 사용하여 용량을 늘릴 수 있습니다.

AWS Data Pipeline 리소스

다음은 AWS Data Pipeline을 사용할 때 도움이 되는 리소스입니다.

- AWS Data Pipeline 제품 정보 – AWS Data Pipeline에 대한 정보를 얻을 수 있는 기본 웹 페이지입니다.
- AWS Data Pipeline 기술 FAQ - 개발자가이 제품에 대해 묻는 상위 20개 질문을 다룹니다.
- 출시 정보 – 최신 릴리스에 대한 수준 높은 개요를 제공합니다. 특히 새로운 기능, 상관 관계 및 알려진 문제점에 대해 다룹니다.
- AWS Data Pipeline 토론 포럼 –Amazon Web Services 관련 기술적 질문을 논의할 수 있는 개발자를 위한 커뮤니티 기반 포럼입니다.

- 클래스 및 워크숍 - AWS 기술을 연마하고 실용적인 경험을 얻는 데 도움이 되는 자기 주도형 실습 외에도 역할 기반 및 특수 과정으로 연결되는 링크입니다.
- AWS 개발자 센터 - 자습서를 살펴보고, 도구를 다운로드하고, AWS 개발자 이벤트에 대해 알아봅니다.
- AWS 개발자 도구 - AWS 애플리케이션 개발 및 관리를 위한 개발자 도구, SDKs, IDE 툴킷 및 명령줄 도구에 대한 링크입니다.
- 리소스 센터 시작하기 -를 설정하고 AWS 계정, AWS 커뮤니티에 가입하고, 첫 번째 애플리케이션을 시작하는 방법을 알아봅니다.
- 실습 튜토리얼 – 단계별 튜토리얼에 따라 AWS에서 첫 번째 애플리케이션을 시작합니다.
- AWS 백서 - 아키텍처, 보안 및 경제와 같은 주제를 다루고 Solutions Architects 또는 기타 기술 전문가가 작성한 AWS 포괄적인 기술 AWS 백서 목록 링크입니다.
- AWS Support 센터 - AWS Support 사례를 생성하고 관리하기 위한 허브입니다. 포럼, 기술 FAQs, 서비스 상태 및 같은 기타 유용한 리소스에 대한 링크도 포함되어 있습니다 AWS Trusted Advisor.
- 지원 - 클라우드에서 애플리케이션을 구축하고 실행하는 데 도움이 지원되는 one-on-one 빠른 응답 지원 채널에 대한 정보를 제공하는 기본 웹 페이지입니다.
- Contact Us(문의처) - AWS 결제, 계정, 이벤트, 침해 및 기타 문제에 대해 문의할 수 있는 중앙 연락 창구입니다.
- AWS 사이트 약관 - 저작권 및 상표, 계정, 라이선스, 사이트 액세스 및 기타 주제에 대한 자세한 정보입니다.

문서 기록

이 설명서는의 2012-10-29 버전과 연결되어 있습니다 AWS Data Pipeline.

변경	설명	릴리스 날짜
AWS Data Pipeline 신규 고객은를 더 이상 사용할 수 없습니다.	AWS Data Pipeline 는 더 이상 신규 고객이 사용할 수 없습니다. 의 기존 고객은 평소와 같이 서비스를 계속 사용할 AWS Data Pipeline 수 있습니다. 자세히 알아보기	2025년 7월 25일
AWS CLI를 사용하여 특정 절차를 수행하기 위한 설명서가 추가되었습니다. AWS Data Pipeline 콘솔 관련 절차를 제거했습니다.	자세한 내용은 파이프라인 복제, 파이프라인 로그 보기, CLI를 사용하여 Data Pipeline 템플릿에서 파이프라인을 생성합니다. 섹션을 참조하세요.	2023년 5월 26일
에서 다른 대체 서비스로 마이그레이션 AWS Data Pipeline 하기 위한 콘텐츠와 샘플을 더 추가했습니다.	각 대안, 서비스 간 개념 매핑 및 샘플에 대한 자세한 정보가 포함된 AWS Glue AWS Step Functions 또는 Amazon MWAA로 마이그레이션 AWS Data Pipeline 하기 위한 주제가 업데이트되었습니다. 자세한 내용은 에서 워크로드 마이그레이션 AWS Data Pipeline 단원을 참조하십시오.	2023년 3월 31일
IMDSv2 AWS Data Pipeline 지원에 대한 정보가 추가되었습니다.	AWS Data Pipeline 는 Amazon EMR 및 Amazon EC2 리소스용 IMDSv2를 지원합니다. Amazon EC2 자세한 내용은 의 데이터 보호 AWS Data Pipeline, EmrCluster, Ec2Resource 섹션을 참조하세요.	2022년 12월 16일
에서 다른 대체 서비스로 마이그레이션 AWS Data Pipeline 하기 위한 주제가 추가되었습니다.	이제 고객에게 더 나은 데이터 통합 경험을 제공하는 다른 AWS 서비스가 있습니다. 의 일반적인 사용 사례를 AWS Glue AWS Step Functions 또는 Amazon MWAA로 마이그레이션 AWS Data Pipeline 할 수 있습니다. 자세한 내용은 에서 워크로드 마이그레이션 AWS Data Pipeline 단원을 참조하십시오.	2022년 12월 16일

변경	설명	릴리스 날짜
지원되는 Amazon EC2 및 Amazon EMR 인스턴스 목록이 업데이트되었습니다. 인스턴스에 사용된 HVM(Hardware Virtual Machine) AMI의 ID 목록이 업데이트되었습니다.	지원되는 Amazon EC2 및 Amazon EMR 인스턴스 목록이 업데이트되었습니다. 자세한 내용은 파이프라인 작업 활동에 대해 지원되는 인스턴스 유형 단원을 참조하십시오. 인스턴스에 사용된 HVM(Hardware Virtual Machine) AMI의 ID 목록이 업데이트되었습니다. 자세한 내용은 구문 및 `imageId` 검색을 참조하세요.	2018년 11월 9일
Amazon EBS 볼륨을 클러스터 노드에 붙이기 위한 구성과 Amazon EMR 클러스터를 프라이빗 서브넷에서 실행하기 위한 구성이 추가되었습니다.	구성 옵션이 `EMRcluster` 객체에 추가되었습니다. Amazon EMR 클러스터를 사용하는 파이프라인에서 이러한 옵션을 사용할 수 있습니다. `coreEbsConfiguration` , `masterEbsConfiguration` 및 `TaskEbsConfiguration` 필드를 사용하여 Amazon EBS 볼륨을 Amazon EMR 클러스터의 코어, 마스터 및 태스크 노드에 첨부하는 작업을 구성합니다. 자세한 내용은 클러스터 노드에 EBS 볼륨 연결 단원을 참조하십시오. `emrManagedMasterSecurityGroupId` , `emrManagedSlaveSecurityGroupId` 및 `ServiceAccessSecurityGroupId` 필드를 사용하여 프라이빗 서브넷에서 Amazon EMR 클러스터를 구성합니다. 자세한 내용은 프라이빗 서브넷에서 Amazon EMR 클러스터 구성 단원을 참조하십시오. `EMRcluster` 구문에 대한 자세한 내용은 EmrCluster 항목을 참조하세요.	2018년 4월 19일

변경	설명	릴리스 날짜
지원되는 Amazon EC2 및 Amazon EMR 인스턴스 목록이 추가되었습니다.	파이프라인 정의에서 인스턴스 유형을 지정하지 않으면에서 기본적으로 AWS Data Pipeline 생성하는 인스턴스 목록이 추가되었습니다. 지원되는 Amazon EC2 및 Amazon EMR 인스턴스 목록이 추가되었습니다. 자세한 내용은 파이프라인 작업 활동에 대해 지원되는 인스턴스 유형 단원을 참조하십시오.	2018년 3월 22일
온디맨드 파이프라인의 지원이 추가되었습니다.	• 온디맨드 파이프라인의 지원이 추가되어, 다시 활성화하여 파이프라인을 다시 실행할 수 있습니다.	2016년 2월 22일
RDS 데이터베이스의 추가 지원	• `rdsInstanceId` , `region` 및 `jdbcDriverJarUri` 가 RdsDatabase에 추가되었습니다. • SqlActivity의 `database`가 업데이트되어 `RdsDatabase` 도 지원됩니다.	2015년 8월 17일
추가 JDBC 지원	• SqlActivity의 `database`가 업데이트되어 `JdbcDatabase` 도 지원됩니다. • JdbcDatabase에 `jdbcDriverJarUri` 가 추가되었습니다. • Ec2Resource 및 EmrCluster에 `initTimeout` 를 추가했습니다. • Ec2Resource에 `runAsUser` 이(가) 추가되었습니다.	2015년 7월 7일
HadoopActivity, 가용 영역 및 스팟 지원	• Hadoop 클러스터로 병렬 작업을 제출하는 지원이 추가되었습니다. 자세한 내용은 HadoopActivity 단원을 참조하십시오. • Ec2Resource 및 EmrCluster로 스팟 인스턴스를 요청하는 기능이 추가되었습니다. • 지정된 가용 영역에서 `EmrCluster` 리소스를 시작하는 기능이 추가되었습니다.	2015년 6월 1일

변경	설명	릴리스 날짜
파이프라인 비활성화	활성 파이프라인을 비활성화하는 지원이 추가되었습니다. 자세한 내용은 파이프라인 비활성화 단원을 참조하십시오.	2015년 4월 7일
템플릿 및 콘솔 업데이트	새 템플릿이 추가되었습니다. [Getting Started with ShellCommandActivity] 템플릿을 사용하는 '시작하기' 장이 업데이트되었습니다. 자세한 내용은 CLI를 사용하여 Data Pipeline 템플릿에서 파이프라인을 생성합니다. 단원을 참조하십시오.	2014년 11월 25일
VPC 지원	가상 프라이빗 클라우드(VPC)에서 리소스를 시작하는 지원이 추가되었습니다.	2014년 3월 12일
리전 지원	여러 서비스 리전의 지원이 추가되었습니다. 외에도 `us-east-1` , `eu-west-1` , 및에서 `ap-northeast-1` `ap-southeast-2` 가 지원 AWS Data Pipeline 됩니다`us-west-2` .	2014년 2월 20일
Amazon Redshift 지원	새 콘솔 템플릿(Redshift로 복사)과 템플릿을 시연하는 자습서를 AWS Data Pipeline포함하여의 Amazon Redshift에 대한 지원이 추가되었습니다. 자세한 내용은 를 사용하여 Amazon Redshift에 데이터 복사 AWS Data Pipeline, RedshiftDataNode, RedshiftDatabase 및 RedshiftCopyActivity 부분을 참조하세요.	2013년 11월 6일
PigActivity	Pig 네이티브 지원을 제공하는 PigActivity가 추가되었습니다. 자세한 내용은 PigActivity 단원을 참조하십시오.	2013년 10월 15일
새로운 콘솔 템플릿, 활동 및 데이터 형식	새로운 HiveCopyActivity 및 DynamoDBExportDataFormat을 포함하는 새로운 CrossRegion DynamoDB Copy 콘솔 템플릿이 추가되었습니다.	2013년 8월 21일
캐스케이드 실패 및 재실행	AWS Data Pipeline 계단식 실패 및 재실행 동작에 대한 정보가 추가되었습니다. 자세한 내용은 캐스케이드 실패 및 재실행 단원을 참조하십시오.	2013년 8월 8일

변경	설명	릴리스 날짜
문제해결 동영상	AWS Data Pipeline 기본 문제 해결 비디오가 추가되었습니다. 자세한 내용은 문제 해결 단원을 참조하십시오.	2013년 7월 17일
활성 파이프라인 편집	활성 파이프라인 편집 및 파이프라인 구성요소 재실행에 관한 자세한 내용이 추가되었습니다. 자세한 내용은 파이프라인 편집 단원을 참조하십시오.	2013년 7월 17일
서로 다른 리전에서 리소스 사용	서로 다른 리전에서 리소스를 사용하는 것에 관한 자세한 내용이 추가되었습니다. 자세한 내용은 여러 리전의 리소스와 파이프라인 사용 단원을 참조하십시오.	2013년 6월 17일
WAITING_ON_DEPENDENCIES 상태	CHECKING_PRECONDITIONS 상태가 WAITING_ON_DEPENDENCIES로 변경되었고, 파이프라인 객체의 @waitingOn 실행 시간 필드가 추가되었습니다.	2013년 5월 20일
DynamoDBDataFormat	DynamoDBDataFormat 템플릿이 추가되었습니다.	2013년 4월 23일
웹 로그 처리 동영상 및 스팟 인스턴스 지원	"AWS Data Pipeline, Amazon EMR 및 Hive로 웹 로그 처리" 동영상과 Amazon EC2 스팟 인스턴스 지원이 소개되었습니다.	2013년 2월 21일
	AWS Data Pipeline 개발자 안내서의 최초 릴리스입니다.	2012년 12월 20일