AWS DATA PIPELINE

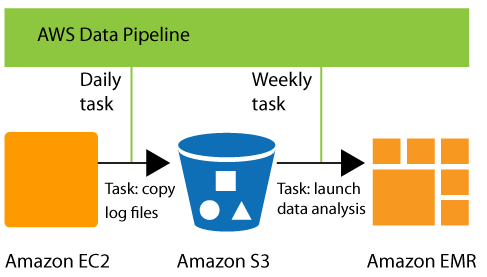
What is AWS Data Pipeline?

AWS Data Pipeline is a web service that you can use to automate the movement and transformation of data. With AWS Data Pipeline, you can define data-driven workflows, so that tasks can be dependent on the successful completion of previous tasks. You define the parameters of your data transformations and AWS Data Pipeline enforces the logic that you've set up.

The following components of AWS Data Pipeline work together to manage your data:

* A pipeline definition specifies the business logic of your data management. For more information, see Pipeline Definition File Syntax.
* A pipeline schedules and runs tasks by creating Amazon EC2 instances to perform the defined work activities. You upload your pipeline definition to the pipeline, and then activate the pipeline. You can edit the pipeline definition for a running pipeline and activate the pipeline again for it to take effect. You can deactivate the pipeline, modify a data source, and then activate the pipeline again. When you are finished with your pipeline, you can delete it.
* Task Runner polls for tasks and then performs those tasks. For example, Task Runner could copy log files to Amazon S3 and launch Amazon EMR clusters. Task Runner is installed and runs automatically on resources created by your pipeline definitions. You can write a custom task runner application, or you can use the Task Runner application that is provided by AWS Data Pipeline. For more information, see Task Runners.

For example, you can use AWS Data Pipeline to archive your web server's logs to Amazon Simple Storage Service (Amazon S3) each day and then run a weekly Amazon EMR (Amazon EMR) cluster over those logs to generate traffic reports. AWS Data Pipeline schedules the daily tasks to copy data and the weekly task to launch the Amazon EMR cluster. AWS Data Pipeline also ensures that Amazon EMR waits for the final day's data to be uploaded to Amazon S3 before it begins its analysis, even if there is an unforeseen delay in uploading the logs.



AWS Data Pipeline Concepts

Before you begin, read about the key concepts and components for AWS Data Pipeline.

Contents

* Pipeline Definition
* Pipeline Components, Instances, and Attempts
* Task Runners
* Data Nodes
* Databases
* Activities
* Preconditions
* Resources
* Actions

Pipeline Definition

A pipeline definition is how you communicate your business logic to AWS Data Pipeline. It contains the following information:

* Names, locations, and formats of your data sources.
* Activities that transform the data.
* The schedule for those activities.
* Resources that run your activities and preconditions.
* Preconditions that must be satisfied before the activities can be scheduled.
* Ways to alert you with status updates as pipeline execution proceeds.

**Pipeline Components, Instances, and Attempts**

There are three types of items associated with a scheduled pipeline:

* **Pipeline Components** — Pipeline components represent the business logic of the pipeline and are represented by the different sections of a pipeline definition. Pipeline components specify the data sources, activities, schedule, and preconditions of the workflow. They can inherit properties from parent components. Relationships among components are defined by reference. Pipeline components define the rules of data management.
* **Instances** — When AWS Data Pipeline runs a pipeline, it compiles the pipeline components to create a set of actionable instances. Each instance contains all the information for performing a specific task. The complete set of instances is the to-do list of the pipeline. AWS Data Pipeline hands the instances out to task runners to process.
* **Attempts** — To provide robust data management, AWS Data Pipeline retries a failed operation. It continues to do so until the task reaches the maximum number of allowed retry attempts. Attempt objects track the various attempts, results, and failure reasons if applicable. Essentially, it is the instance with a counter. AWS Data Pipeline performs retries using the same resources from the previous attempts, such as Amazon EMR clusters and EC2 instances.

**Note**

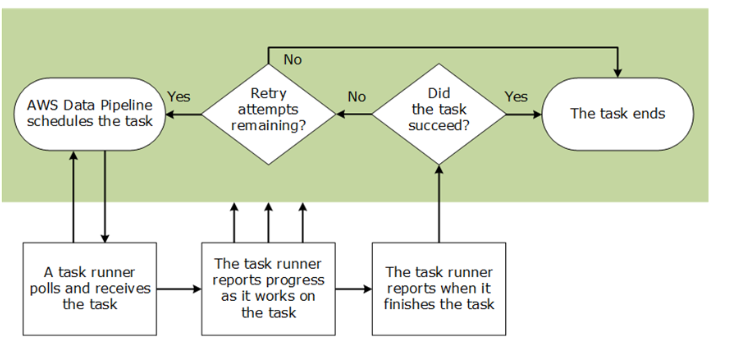
Retrying failed tasks is an important part of a fault tolerance strategy, and AWS Data Pipeline definitions provide conditions and thresholds to control retries. However, too many retries can delay detection of an unrecoverable failure because AWS Data Pipeline does not report failure until it has exhausted all the retries that you specify. The extra retries may accrue additional charges if they are running on AWS resources. As a result, carefully consider when it is appropriate to exceed the AWS Data Pipeline default settings that you use to control re-tries and related settings.

Task Runners

A task runner is an application that polls AWS Data Pipeline for tasks and then performs those tasks.

Task Runner is a default implementation of a task runner that is provided by AWS Data Pipeline. When Task Runner is installed and configured, it polls AWS Data Pipeline for tasks associated with pipelines that you have activated. When a task is assigned to Task Runner, it performs that task and reports its status back to AWS Data Pipeline.

The following diagram illustrates how AWS Data Pipeline and a task runner interact to process a scheduled task. A task is a discrete unit of work that the AWS Data Pipeline service shares with a task runner. It differs from a pipeline, which is a general definition of activities and resources that usually yields several tasks.



There are two ways you can use Task Runner to process your pipeline:

* AWS Data Pipeline Installs Task Runner for you on resources that are launched and managed by the AWS Data Pipeline web service.
* You install Task Runner on a computational resource that you manage, such as a long-running EC2 instance, or an on-premises server.

Data Nodes

In AWS Data Pipeline, a data node defines the location and type of data that a pipeline activity uses as input or output. AWS Data Pipeline supports the following types of data nodes:

**DynamoDBDataNode**

A DynamoDB table that contains data for HiveActivity or EmrActivity to use.

**SqlDataNode**

An SQL table and database query that represent data for a pipeline activity to use.

***Note***

Previously, MySqlDataNode was used. Use SqlDataNode instead.

**RedshiftDataNode**

An Amazon Redshift table that contains data for RedshiftCopyActivity to use.

**S3DataNode**

An Amazon S3 location that contains one or more files for a pipeline activity to use.

**Databases**

AWS Data Pipeline supports the following types of databases:

**JdbcDatabase**

A JDBC database.

**RdsDatabase**

An Amazon RDS database.

**RedshiftDatabase**

An Amazon Redshift database.

Activities

In AWS Data Pipeline, an activity is a pipeline component that defines the work to perform. AWS Data Pipeline provides several pre-packaged activities that accommodate common scenarios, such as moving data from one location to another, running Hive queries, and so on. Activities are extensible, so you can run your own custom scripts to support endless combinations.

AWS Data Pipeline supports the following types of activities:

**CopyActivity**

Copies data from one location to another.

**EmrActivity**

Runs an Amazon EMR cluster.

**HiveActivity**

Runs a Hive query on an Amazon EMR cluster.

**HiveCopyActivity**

Runs a Hive query on an Amazon EMR cluster with support for advanced data filtering and support for S3DataNode and DynamoDBDataNode.

**PigActivity**

Runs a Pig script on an Amazon EMR cluster.

**RedshiftCopyActivity**

Copies data to and from Amazon Redshift tables.

**Preconditions**

In AWS Data Pipeline, a precondition is a pipeline component containing conditional statements that must be true before an activity can run. For example, a precondition can check whether source data is present before a pipeline activity attempts to copy it. AWS Data Pipeline provides several pre-packaged preconditions that accommodate common scenarios, such as whether a database table exists, whether an Amazon S3 key is present, and so on. However, preconditions are extensible and allow you to run your own custom scripts to support endless combinations.

There are two types of preconditions: system-managed preconditions and user-managed preconditions. System-managed preconditions are run by the AWS Data Pipeline web service on your behalf and do not require a computational resource. User-managed preconditions only run on the computational resource that you specify using the runsOn or workerGroup fields. The workerGroup resource is derived from the activity that uses the precondition.

Resources

In AWS Data Pipeline, a resource is the computational resource that performs the work that a pipeline activity specifies. AWS Data Pipeline supports the following types of resources:

**Ec2Resource**

An EC2 instance that performs the work defined by a pipeline activity.

**EmrCluster**

An Amazon EMR cluster that performs the work defined by a pipeline activity, such as EmrActivity.

Resources can run in the same region with their working dataset, even a region different than AWS Data Pipeline. For more information, see Using a Pipeline with Resources in Multiple Regions.

Actions

AWS Data Pipeline actions are steps that a pipeline component takes when certain events occur, such as success, failure, or late activities. The event field of an activity refers to an action, such as a reference to snsalarm in the onLateAction field of EmrActivity.

AWS Data Pipeline relies on Amazon SNS notifications as the primary way to indicate the status of pipelines and their components in an unattended manner. For more information, see Amazon SNS. In addition to SNS notifications, you can use the AWS Data Pipeline console and CLI to obtain pipeline status information.

AWS Data Pipeline supports the following actions:

**SnsAlarm**

An action that sends an SNS notification to a topic based on onSuccess, OnFail, and onLateAction events.

**Terminate**

An action that triggers the cancellation of a pending or unfinished activity, resource, or data node. You cannot terminate actions that include onSuccess, OnFail, or onLateAction.

**Proactively Monitoring Pipelines**

The best way to detect problems is to monitor your pipelines proactively from the start. You can configure pipeline components to inform you of certain situations or events, such as when a pipeline component fails or doesn't begin by its scheduled start time. AWS Data Pipeline makes it easy to configure notifications by providing event fields on pipeline components that you can associate with Amazon SNS notifications, such as onSuccess, OnFail, and onLateAction.

Working with Pipelines

You can administer, create, and modify pipelines using the AWS Data Pipeline console, an AWS SDK, or the command line interface (CLI). The following sections introduce fundamental AWS Data Pipeline concepts and show you how to work with pipelines.

**Contents**

* Scheduling Pipelines
* Creating a Pipeline
* Viewing Your Pipelines
* Editing Your Pipeline
* Cloning Your Pipeline
* Tagging Your Pipeline
* Deactivating Your Pipeline
* Deleting Your Pipeline
* Staging Data and Tables with Pipeline Activities
* Launching Resources for Your Pipeline into a VPC
* Using Amazon EC2 Spot Instances in a Pipeline
* Using a Pipeline with Resources in Multiple Regions
* Cascading Failures and Reruns
* Pipeline Definition File Syntax
* Working with the API

Creating a Pipeline

AWS Data Pipeline provides several ways for you to create pipelines:

* Use the console with a template provided for your convenience. For more information, see Creating Pipelines Using Console Templates.
* Use the console to manually add individual pipeline objects. For more information, see Creating Pipelines Using the Console Manually.
* Use the AWS Command Line Interface (CLI) with a pipeline definition file in JSON format.
* Use an AWS SDK with a language-specific API. For more information, see Working with the API.

Export MySQL Data to Amazon S3 Using AWS Data Pipeline

This tutorial walks you through the process of creating a data pipeline to copy data (rows) from a table in MySQL database to a CSV (comma-separated values) file in an Amazon S3 bucket and then sending an Amazon SNS notification after the copy activity completes successfully. You will use an EC2 instance provided by AWS Data Pipeline for this copy activity.

Pipeline Objects

The pipeline uses the following objects:

* CopyActivity
* Ec2Resource
* MySqlDataNode
* S3DataNode
* SnsAlarm

**Contents**

Before You Begin

Copy MySQL Data Using the AWS Data Pipeline Console

Copy MySQL Data Using the Command Line

Copy MySQL Data Using the AWS Data Pipeline Console

You can create a pipeline to copy data from a MySQL table to a file in an Amazon S3 bucket.

**Tasks**

* Create the Pipeline
* Save and Validate Your Pipeline
* Verify Your Pipeline Definition
* Activate Your Pipeline
* Monitor the Pipeline Runs
* (Optional) Delete Your Pipeline

Create the Pipeline

First, create the pipeline. The pipeline must be created in the same region as your target RDS instance.

**To create your pipeline**

Open the AWS Data Pipeline console at https://console.aws.amazon.com/datapipeline/.

The first screen that you see depends on whether you've created a pipeline in the current region.

If you haven't created a pipeline in this region, the console displays an introductory screen. Choose Get started now.

If you've already created a pipeline in this region, the console displays a page that lists your pipelines for the region. Choose Create new pipeline.

In Name, enter a name for your pipeline.

(Optional) In Description, enter a description for your pipeline.

For Source, select Build using a template, and then select the following template: Full copy of RDS MySQL table to S3.

Under the Parameters section, which opened when you selected the template, do the following:

For DBInstance ID, enter the DB instance name of the Aurora DB instance you want to use to copy data from the Aurora cluster.

To locate the endpoint details for your DB instance, see Connecting to a DB Instance Running the MySQL Database Engine in the Amazon RDS User Guide.

For RDS MySQL username, enter the user name you used when you created your MySQL database instance.

In the RDS MySQL password field, enter the password you used when you created your DB instance.

In the EC2 instance type field, enter the instance type for your EC2 instance.

Click the folder icon next to Output S3 folder, select one of your buckets or folders, and then click Select.

Under Schedule, choose on pipeline activation.

Under Pipeline Configuration, leave logging enabled. Choose the folder icon under S3 location for logs, select one of your buckets or folders, and then choose Select.

If you prefer, you can disable logging instead.

Under Security/Access, leave IAM roles set to Default.

Click Edit in Architect.

In the left pane, separate the icons by dragging them apart. This is a graphical representation of your pipeline. The arrows indicate the connections between the objects.

Next, configure the database name setting, which currently is not present on the available template.

In the left pane, click RDSDatabase.

In the right pane, under the rds\_mysql section, for Add an optional field... choose Database Name.

Type the Database Name of your target database and add optional fields.

You can configure the Amazon SNS notification action AWS Data Pipeline performs after the copy activity finishes successfully.

To configure the Amazon SNS notification action

In the right pane, click Activities.

From Add an optional field, select On Success.

From the newly added On Success, select Create new: Action.

In the right pane, click Others.

Under DefaultAction1, do the following:

Enter a name for your notification (for example, CopyDataNotice).

From Type, select SnsAlarm.

In the Message field, enter the message content.

In the Subject field, enter the subject line for your notification.

In the Topic Arn field, enter the ARN of your topic.

Leave Role field set to the default value.

Save and Validate Your Pipeline

You can save your pipeline definition at any point during the creation process. As soon as you save your pipeline definition, AWS Data Pipeline looks for syntax errors and missing values in your pipeline definition. If your pipeline is incomplete or incorrect, AWS Data Pipeline generates validation errors and warnings. Warning messages are informational only, but you must fix any error messages before you can activate your pipeline.

**To save and validate your pipeline**

Choose Save pipeline.

AWS Data Pipeline validates your pipeline definition and returns either success or error or warning messages. If you get an error message, choose Close and then, in the right pane, choose Errors/Warnings.

The Errors/Warnings pane lists the objects that failed validation. Choose the plus (+) sign next to the object names and look for an error message in red.

When you see an error message, go to the specific object pane where you see the error and fix it. For example, if you see an error message in the DataNodes object, go to the DataNodes pane to fix the error.

After you fix the errors listed in the Errors/Warnings pane, choose Save Pipeline.

Repeat the process until your pipeline validates successfully.

Verify Your Pipeline Definition

It is important that you verify that your pipeline was correctly initialized from your definitions before you activate it.

**To verify your pipeline definition**

On the List Pipelines page, look for your newly-created pipeline.

AWS Data Pipeline has created a unique Pipeline ID for your pipeline definition.

The Schedule State column in the row listing your pipeline should show PENDING.

Choose the triangle icon next to your pipeline. A pipeline summary pane below shows the details of your pipeline runs. Because your pipeline is not yet activated, you are not likely to see any execution details. However, you will see the configuration of the pipeline definition.

Activate Your Pipeline

Activate your pipeline to start creating and processing runs. The pipeline starts based on the schedule and period in your pipeline definition.

Important

If activation succeeds, your pipeline is running and might incur usage charges. For more information, see AWS Data Pipeline pricing. To stop incurring usage charges for AWS Data Pipeline, delete your pipeline.

**To activate your pipeline**

* Choose Activate.
* In the confirmation dialog box, choose Close.

Monitor the Pipeline Runs

After you activate your pipeline, you are taken to the Execution details page where you can monitor the progress of your pipeline.

**To monitor the progress of your pipeline runs**

Choose Update or press F5 to update the status displayed.

Tip

If there are no runs listed, ensure that Start (in UTC) and End (in UTC) cover the scheduled start and end of your pipeline, and then choose Update.

When the status of every object in your pipeline is FINISHED, your pipeline has successfully completed the scheduled tasks. If you created an SNS notification, you should receive email about the successful completion of this task.

If your pipeline doesn't complete successfully, check your pipeline settings for issues. For more information about troubleshooting failed or incomplete instance runs of your pipeline, see Resolving Common Problems.

(Optional) Delete Your Pipeline

To stop incurring charges, delete your pipeline. Deleting your pipeline deletes the pipeline definition and all associated objects.

**To delete your pipeline**

On the List Pipelines page, select your pipeline.

Click Actions, and then choose Delete.

When prompted for confirmation, choose Delete.