Linux Academy

Hands-On Training

Using Data Pipeline to Copy DynamoDB Data to S3

Contents

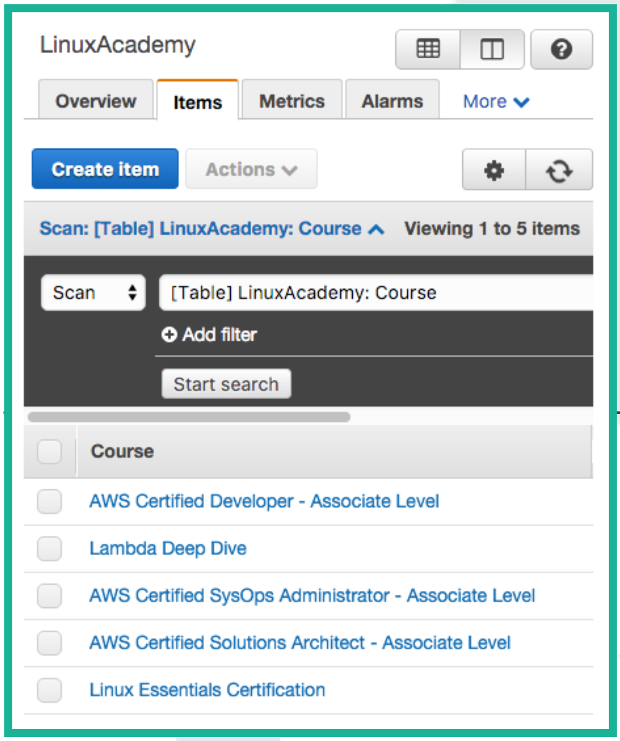
Introduction

AWS Data Pipeline allows you to move data between AWS compute and storage services with ease. In this lab, we learn how to manage this first hand by taking our sample DynamoDB data and, through the use of Data Pipeline, copying it to an S3 bucket to work as a backup.

Getting Started

Log into the AWS Console using the given credentials. A DynamoDB table has already been created for you, as have IAM roles.

Open the **DynamoDB Dashboard**, and navigate to **Tables**. A *LinuxAcademy* table is available. Upon clicking the table, and viewing the **Items**, a list of courses should appear. This is the data with which we are working.



□	Course
□	AWS Certified Developer - Associate Level
□	Lambda Deep Dive
□	AWS Certified SysOps Administrator - Associate Level
□	AWS Certified Solutions Architect - Associate Level
□	Linux Essentials Certification

Creating a Data Pipeline

Using this data, we need to create a Data Pipeline that will back it up to an S3 bucket, in a JSON-based format. The JSON data can also be used for other analytics.

The Data Pipeline needs to launch inside of a subnet. Open your **VPC Dashboard** to view your subnets. You can use either subnet, as long as you have ensured there is an Internet gateway attached. Note the *Subnet ID*.

Open the **Data Pipeline Dashboard**, and select **Get started now**.

We are giving our Pipeline the **Name** of *Backup DynamoDB Table*, and provided a **Description** of *This pipeline will create a backup of our DynamoDB table*. You can change these as desired.

We are **building using a template**, specifically the template *Export DynamoDB table to S3*, which is the exact process we wish to perform. You can also build sources using AWS Architect, although this is outside the scope of this lab.

For our **Parameters** we need to select a **Source DynamoDB table name**, which is *LinuxAcademy*, the name of our table. The **Output S3 folder** only has a single S3 bucket available, so select this.

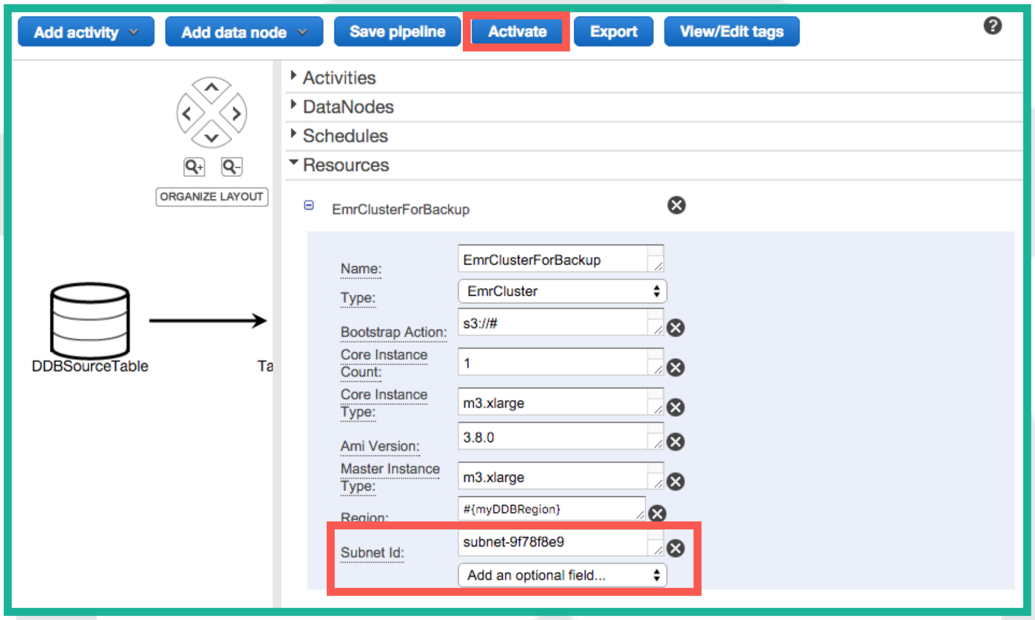
Although we can set schedules and manage batch processing for our pipeline, we are setting **Run** (under **Schedule**) to *once on the pipeline activations*, since this is a lab. In production, you may need to take these backups on a schedule or suplement it with other neccesary actions.

Select the only available bucket for **S3 location for logs**.

Our **IAM rule** should be set to *custom*, with both the **Pipeline role** and **EC2 instance role** set to the longest option that includes the name of this lab.

Editing in Architect

Before we can activate our pipeline, we need to edit it in the AWS Architect. At the bottom of the page, select **Edit in Architect**.



Name: EmrClusterForBackup
Type: EmrCluster
Bootstrap Action: s3://#
Core Instance Count: 1
Core Instance Type: m3.xlarge
Ami Version: 3.8.0
Master Instance Type: m3.xlarge
Region: #{myDDBRegion}
Subnet Id: subnet-9f78f8e9
Add an optional field...

Open the **Resources** menu, and select **Add an optional field**, *Subnet ID*. Add your subnet ID copied above to this option. This is necessary because we do not have a default VPC.

Select **Activate**.

If you refresh the page, you can see your pipeline listed as *WAITING_ON_DEPENDENCIES*. Should you navigate to **Elastic MapReduce**, you can even see the cluster that was created for this pipeline. It will be terminated when pipeline has finished.

Viewing the Results

Navigate to the **S3 Dashboard**, and open your S3 bucket. One folder contains logs while the other contains your DynamoDB backups. The folder named after the current date should be the one containing the database information.

From here, you can download the data from the S3 bucket as desired.

Using the Backup Data

The JSON data taken from the DynamoDB table can also be used to repopulate the database, as needed. If you return to your **Pipeline Dashboard** and create another pipeline, you will see, under templates, an option to Import *DynamoDB backup data from S3*. For extra practice, see if you can configure a pipeline for restoring DynamoDB data!