As we work to finalize work on our Course Progress system, some variations may be seen. Please know that your progress is being tracked properly, even if it is not shown properly on the display. If you have an immediate need, please contact Support so we can investigate your account individually. Thank you for your continued patience!

Create Streaming Data Pipeline on GCP with Cloud Pub/Sub, Dataflow, and BigQuery

Cancel Lab | Complete Lab

107 Min. Remaining | Advanced

How was this lab?

Credentials

How do I connect?

Google Labs Account

Username

cloud_user_p_4112dc@linuxacademygclabs.com

Password

PU8Em//w

Open Google Console

Additional Information and Resources

Many data engineer scenarios on GCP involve a multi-step streaming data pipeline from ingestion, to processing, to storage/analysis. In this lab, we will create a simulated end to end streaming pipeline of all steps, which will finish in analyzing captured streaming data for insights.

Be sure to launch the lab in your browser's incognito (or other private browsing) mode to avoid cached login issue.

Learning Objectives

Prepare Your Environment

Enable pub/sub and dataflow APIs:

```
gcloud services enable dataflow.googleapis.com
gcloud services enable pubsub.googleapis.com
```

Create a Cloud Storage bucket for Dataflow staging:

```
gsutil mb gs://$DEVSHELL_PROJECT_ID
```

Download the GitHub repository used for lab resources:

```
cd ~
git clone
```

Tools

Instant Terminal | Diagram

Video | Guide

Create Streaming Data Pipeline on GCP with Cloud Pub/Sub, Dataflow, and BigQuery

Introduction

This lab will simulate live highway sensor data which will be published to a Cloud Pub/Sub topic. Then, a Cloud Dataflow streaming pipeline will subscribe to it. The pipeline will take the streaming sensor data, transform it, and insert it into a BigQuery table. We will then view the streaming inserts in BigQuery while they are in progress, and attempt to gain some useful insights from the streaming data.

Solution

Many data engineer scenarios on GCP involve a multi-step streaming data pipeline from ingestion, to processing, to storage/analysis. In this lab, we will create a simulated end to end streaming pipeline of all steps, which will finish in analyzing captured streaming data for insights.

How to Log in to Google Lab Accounts

On the lab page, right-click **Open GCP Console** and select the option to open it in a new private browser window. This option will read differently depending on your browser. In Chrome it says "Open Link in Incognito Window". In Firefox it says "Open link in new private window." In Microsoft Edge, the message will be "Open in InPrivate window." And in Safari, press **Alt** or **Option**, then right click to get a menu where you will choose "Open link in new private window."

This will avoid any cached login issues. Once you're at the login screen, sign into Google Cloud Platform using the credentials provided on the lab page.

On the **Welcome to your new account** screen, review the text

```
https://github.com/linuxacademy/googledataengineer
```

Create Pub/Sub Topic

```
gcloud pubsub topics create sandiego
```

Create a BigQuery Dataset to Stream Data Into

Create a BigQuery dataset to stream data into:

```
bq mk --dataset $DEVSHELL_PROJECT_ID:demos
```

The table will be named `average_speeds`. We do not create the table, but Dataflow will create it within the dataset for us.

View the Dataflow Template

We will not be interacting with the template directly. We will be using a script that will install the Java environment and execute the template as a Dataflow job:

```
vim
googledataengineer/courses/streaming/process/sandiego/src/main/java/com/google/cloud/training/dataanalyst/sandiego/AverageSpeeds.java
```

Create the Dataflow Streaming Job

Go to the Dataflow job script directory:

```
cd
~/googledataengineer/courses/streaming/process/sandiego
```

Execute the script that creates the Dataflow streaming job, and subscribe to the Pub/Sub topic.

This script passes along the Project ID, staging bucket (also the Project ID), and the name of the Java template to use:

```
./run_oncloud.sh $DEVSHELL_PROJECT_ID
$DEVSHELL_PROJECT_ID AverageSpeeds
```

When complete, the streaming job will be subscribed to our Pub/Sub topic, and waiting for streaming input from our simulated sensor data.

Publish Simulated Traffic Sensor Data to Pub/Sub via a Python Script and Pre-Created Dataset

Browse to the Python script directory:

```
cd ~/googledataengineer/courses/streaming/publish
```

Install any requirements for the Python script:

```
sudo pip install -U google-cloud-pubsub
```

Download the simulated sensor data:

```
gsutil cp gs://la-gcloud-course-
```

```
resources/sandiego/sensor_obs2008.csv.gz .
```

Execute the Python script to publish simulated streaming data to Pub/Sub:

```
./send_sensor_data.py --speedFactor=60 --
project=$DEVSHELL_PROJECT_ID
```

View the Streamed Data in BigQuery

In BigQuery, execute the following query to view the current streamed data, both in the table and in the streaming buffer:

```
SELECT *
FROM `demos.average_speeds`
```

Notice the total count of records at the bottom. Wait about a minute and run the same query again (be sure to uncheck **use cached results** in query options) and notice that the number has increased.

Use Aggregated Queries to Gain Insights

Let's get some use out of this data. If we wanted to forecast some necessary road maintenance, we would need to know which lanes have the most traffic, to know which ones will require resurfacing first.

Enter the following query to view which lanes have the most sensor counts:

```
SELECT lane, sum(lane) as total
FROM `demos.average_speeds`
GROUP BY lane
ORDER BY total DESC
```

We can also view which lanes have the highest average speeds:

```
SELECT lane, avg(speed) as average_speed
FROM `demos.average_speeds`
GROUP BY lane
ORDER BY average_speed DESC
```