Leveraging Unstructured Data - Lab 5 : Cluster automation using CLI commands v1.3

## Overview

In this lab, you will create a cluster using CLI commands and learn about the Dataproc-GCP workflow and workflow automation.

## Objectives

In this lab, you will perform the following tasks:

* Create a customized Dataproc cluster using Cloud Shell

## Task 1: Preparation

You will be creating the Dataproc cluster from the training VM. There are several values that will be required by the commands. You will create environment variables for these values.

### **Open the SSH terminal and connect to the training VM**

1. In the Console, on the **Navigation menu** ( 7a91d354499ac9f1.png) click **Compute Engine** > **VM instances**.
2. Locate the line with the instance called **training\_vm**.
3. On the far right, under 'connect', click on **SSH** to open a terminal window.
4. In this lab, you will enter CLI commands on the **training\_vm**.

### **Verify the bucket to be used for Dataproc staging**

Dataproc can use a Cloud Storage bucket to stage its files during initialization. You can use this bucket to stage application programs or data for use by Dataproc. The bucket can also host Dataproc initialization scripts and output. The bucket name must be globally unique. Qwiklabs has already created a bucket for you that has the same name as the Project ID, which is already globally unique.

1. In the Console, on the **Navigation menu** ( 7a91d354499ac9f1.png) click **Storage** > **Browser**. Verify that the bucket exists. Notice the default storage class and the location (region) of this bucket. You will be using this region information next.

On the **training\_vm** SSH terminal, set the BUCKET.

BUCKET=<bucket-name>

You can use $BUCKET in CLI commands. And if you need to enter the bucket name <your-bucket> in a text field in Console, you can quickly retrieve the name with **echo $BUCKET**.

### **Identify a region and zone**

You will be creating a Dataproc cluster in a specific region. The Dataproc cluster and the bucket it will use for staging must be in the same region. Since the bucket you are using already exists, you will need to match the environment variable **$MYREGION** to the bucket region.

1. You can use find the region used by Qwiklabs on the Qwiklabs tab under Connection Details, labeled **QL Region**.

The zone must be in the same region **$MYZONE** will contain this value.

1. You can find the zone used by Qwiklabs on the Qwiklabs tab under Connection Details, labeled **QL Zone**.

On the **training\_vm** SSH terminal, set the REGION and ZONE.

MYREGION=<region>

MYZONE=<zone>

### **Identify a project**

One environment variable that you will set is **$PROJECT\_ID** that contains the Google Cloud project ID required to access billable resources.

1. In the Console, on the **Navigation menu** ( 7a91d354499ac9f1.png) click **Home**. In the panel with Project Info, the **Project ID** is listed. You can also find this information in the Qwiklabs tab under Connection Details, where it is labeled **GCP Project ID**.

On the **training\_vm** SSH terminal, set the PROJECT\_ID.

PROJECT\_ID=<project ID>

### **Identify the browser IP address**

You will use the browser IP address to enable your local browser to reach the Dataproc cluster.

1. Find your computer's browser IP address by opening a browser window and viewing <http://ip4.me/> Copy the IP address.
2. Create an environment variable named **BROWSER\_IP**.

BROWSER\_IP=<your-browser-ip>

### **Copy sample files to the training\_vm home directory**

The sample files you need are have already been archived on training\_vm. You will need to copy them into your user directory with the following command.

1. In the **training\_vm** SSH terminal window.

cd

cp -r /training/training-data-analyst .

ls

1. You should now have the **/training** directory in your home directory. And it should have files within it.

Click Check my progress to verify the objective.

Copy sample files to the training\_vm home directory

Check my progress

## Task 2: Customize the Dataproc initialization action

### **Review the cluster initialization script**

1. Review the cluster customization script.

cd ~/training-data-analyst/courses/unstructured/

cat init-script.sh

This script installs the Google Python Client API on all the machines in the cluster.

1. Customize the initialization to add instructions to clone the course repository to the Master Node.

Use **nano** or **vi** to edit the **init-script.sh** file.

#!/bin/bash

# install Google Python client on all nodes

apt-get update

apt-get install -y python-pip

pip install --upgrade google-api-python-client

ROLE=$(/usr/share/google/get\_metadata\_value attributes/dataproc-role)

if [[ "${ROLE}" == 'Master' ]]; then

git clone https://github.com/GoogleCloudPlatform/training-data-analyst

fi

You can create your own cluster initialization actions using this script as an example. Initialization actions are commonly used to install software on the nodes in the cluster. And this script shows how to distinguish the Master node from the Worker nodes.

## Task 3: Create the Dataproc cluster

1. Verify that the Cloud Storage bucket exists and that the $BUCKET and $MYZONE environment variables are still set. The bucket will be used by the Dataproc cluster to stage files as the cluster initializes.

echo $BUCKET $MYREGION $MYZONE

echo $PROJECT\_ID

Allow the command to default to your account project and default location.

1. Copy the cluster customization script to the bucket.

gsutil cp init-script.sh gs://$BUCKET

Cloud Storage is a very sophisticated distributed and resilient data service that supports Spark RDDs. It is connected to Dataproc by a petabit bisection bandwidth network enabling the data to be processed from where it is located rather than needing to be copied. So you can use Cloud Storage instead of HDFS.

Because data in Cloud Storage survives cluster shutdown, if you used it instead of HDFS, you can terminate clusters when they are not being used to reduce the expense. You can schedule the cluster to terminate after it is idle for a period (when the jobs are done).

<https://cloud.google.com/dataproc/docs/concepts/configuring-clusters/scheduled-deletion>

You can also use Cloud Storage to stage applications and to hold output files.

A cluster can use a single Global Endpoint or Regional Endpoints. You can select the zone where the cluster will be created or allow the zone to be chosen for you.

[https://cloud.google.com/dataproc/docs/concepts/configuring-clusters/auto-zone](https://cloud.google.com/dataproc/docs/concepts/configuring-clusters/auto-zone" \t "\\_blank)

1. In addition to the custom initialization script, you can use initialization scripts that have been predefined. The script located at: **gs://cloud-training-demos/dataproc/datalab.sh** installs Datalab on the Master Node. Datalab is a notebook-based development environment based on Jupyter notebooks.
2. Notice that this cluster includes two preemptible worker nodes.
3. Create the custom cluster

gcloud dataproc clusters create cluster-custom \

--bucket $BUCKET \

--subnet default \

--zone $MYZONE \

--region $MYREGION \

--master-machine-type n1-standard-2 \

--master-boot-disk-size 100 \

--num-workers 2 \

--worker-machine-type n1-standard-1 \

--worker-boot-disk-size 50 \

--num-preemptible-workers 2 \

--image-version 1.2 \

--scopes 'https://www.googleapis.com/auth/cloud-platform' \

--tags customaccess \

--project $PROJECT\_ID \

--initialization-actions 'gs://'$BUCKET'/init-script.sh','gs://cloud-training-demos/dataproc/datalab.sh'

Wait until the command is complete in the terminal before continuing. It may take several minutes for all the initialization actions to finish.

Options used in this command include security, cost-savings, and flexibility features.

--tags: Applies a network tag so you can automate the creation of firewall rules.

--scopes: Applies Cloud IAM restrictions and permissions to the cluster.

--num-preemptible-workers: Controls the number of low cost worker nodes present.

--initialization-actions: Customizes the software on the cluster.

Options for further study:

--no-address, --network, --subnet:

VMs only have internal IPs for added security. Requires enabling GCP API private access on the network, establishing specific firewall rules, and passing the subnet.

<https://cloud.google.com/dataproc/docs/concepts/configuring-clusters/network>

Cluster creation can be scripted based on this CLI command. Therefore cluster creation can be automated. You can create clusters periodically using a Cron job. You could write an application detect an PySpark file being added to a bucket in Cloud Storage and use that to trigger cluster creation. You could also integrate Dataproc cluster creation and job submission into a Continuous Integration environment like Travis or Jenkins. In this way you only start and pay for clusters when they are needed.

## Task 4: Verify Cluster Customization

In this task, you will verify that the customization initialization action installed Cloud Datalab on the Master Node.

1. Verify that your Browser's IP address is set in an environment variable for use in the firewall rule.

echo $BROWSER\_IP

1. Create a firewall rule.

gcloud compute \

--project=$PROJECT\_ID \

firewall-rules create allow-custom \

--direction=INGRESS \

--priority=1000 \

--network=default \

--action=ALLOW \

--rules=tcp:9870,tcp:8088,tcp:8080 \

--source-ranges=$BROWSER\_IP/32 \

--target-tags=customaccess

1. Locate the Master Node External IP Address. In the Console, on the **Navigation menu** ( 7a91d354499ac9f1.png) click **Dataproc** > **Clusters**. Click on **cluster-custom**.
2. Click on **VM Instances**.
3. Click on **cluster-custom-m**.
4. In the Network interfaces section, find the External IP. Highlight and copy it.
5. Open a new browser tab or window. Enter **<external\_IP>:8080** and press return.

You should see the Google Cloud Datalab.

Creating the custom cluster is the objective of this lab. If this was your production environment, your next steps might be:

* Turn the create commands into a script so that you can start up a cluster on demand.
* Add an option to the command to terminate the cluster after a quiet period.
* Turn the firewall rule into a script so that you can enable/disable external (browser) access only when it is required for administration activities.
* Develop and test your application in Datalab notebooks.
* Host the production application in a Cloud Storage bucket.
* Host and access your data in either Cloud Storage, BigQuery, or Bigtable.
* For capacity, Edit the number of preemptible worker nodes using Console, and the running cluster will adapt.
* Shut down the cluster when not in use, or schedule auto termination.