**Running a Pyspark Job on Cloud Dataproc Using Google Cloud Storage**

Introduction

In hands-on lab, we will cover how to use Google Cloud Storage as the primary input and output location for Dataproc cluster jobs.

Solution

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 the browser being used:

* 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."
* In Safari, press **Alt** or **Option**, then right click to get a menu where we will choose "Open link in new private window."

This will avoid any cached login issues. Once we're at the login screen, sign into Google Cloud Platform using the login info provided on the *Credenitals* section of the hands-on lab page.

On the *Welcome to your new account* screen, review the text, and click **Accept**. In the "Welcome L.A.!" window that pops up once we're signed in, check to agree to the terms of service, choose country of residence, and then click **Agree and Continue**.

Prepare Our Environment

1. Click the "square command icon" in the top right of the screen.
2. Click **START CLOUD SHELL**.
3. First, we need to enable the Dataproc API, with:

gcloud services enable dataproc.googleapis.com

1. Next, we will create a Cloud Storage bucket:

gsutil mb -l us-central1 gs://$DEVSHELL\_PROJECT\_ID-data

1. Verify in the web console that the bucket was created by accessing the top-left menu, and then click **Storage**.

**Note:** We should see that our bucket name is identical to the project ID, followed by -data.

1. Now create the ephemeral dataproc cluster:

gcloud dataproc clusters create wordcount --zone=us-central1-f --single-node --master-machine-type=n1-standard-2

1. Verify that the dataproccluster was created by accessing the top-left menu, and then click **Dataproc** underneath the *BIG DATA* section.

**Note**: It can take a few minutes for the creation process to complete.

1. Finally, download the wordcount.py file that will be used for the pyspark job:

gsutil cp -r gs://la-gcp-labs-resources/data-engineer/dataproc/\* .

1. We can view the directory contents with the ls command.
2. Look at the wordcount.py file directly with the following command:

vim wordcount.py

1. Use the following command to view the file that we will be copying shortly:

vim romeoandjuliet.txt

Submit the Pyspark Job to the Dataproc Cluster

1. In Cloud Shell, type the following, and then hit enter:
2. gcloud dataproc jobs submit pyspark wordcount.py --cluster=wordcount -- \
3. gs://la-gcp-labs-resources/data-engineer/dataproc/romeoandjuliet.txt \

gs://$DEVSHELL\_PROJECT\_ID-data/output/

1. View the progress by going back to the *Dataproc* page's *Cluster Details* section, and click **Jobs** on the top-left menu to access the job.

**Note**: This job may take approximately 30-45 seconds to complete, and we should see a confirmation message in the *Job Details* section when clicking on the job.

1. Navigate to the top-left menu, and then click **Storage**.
2. Click on the **data location** bucket.

**Note**: Do *not* click on the **staging bucket** that has dataproc its name.

1. Click the output\ folder.

Review the Pyspark Output

1. In Cloud Shell, download output files from the GCS output location:

gsutil cp -r gs://$DEVSHELL\_PROJECT\_ID-data/output/\* .

**Note**: Alternatively, we could download them to our local machine via the web console.

1. We can view the contents again, with the ls command.
2. Use the following to see an output file:

vim part-00001

1. Use the following to see the output file:

vim part-00000

Delete the Dataproc Cluster

1. We don't need our cluster any longer, so let's delete it. In the web console, go to the top-left menu and into **BIGDATA** > **Dataproc**.
2. Select the **wordcount** cluster, then click **DELETE** > **OK** to confirm.

Our job output still remains in Cloud Storage, allowing us to delete Dataproc clusters when no longer in use to save costs, while preserving input and output resources.

Conclusion

Congratulations - you've completed this hands-on lab!