

EECS E6893 Big Data Analytics - Fall 2019

Homework Assignment 0: Intro to Big Data Analytics on GCP

Due Friday, September 20th, 2019 by 5:00pm

TL;DR

1. Setup GCP and Cloud SDK
2. Learn Dataproc (Spark), Cloud Storage, and BigQuery
3. HW: write a report containing your answers to the 3 questions and submit to Canvas

Abstract

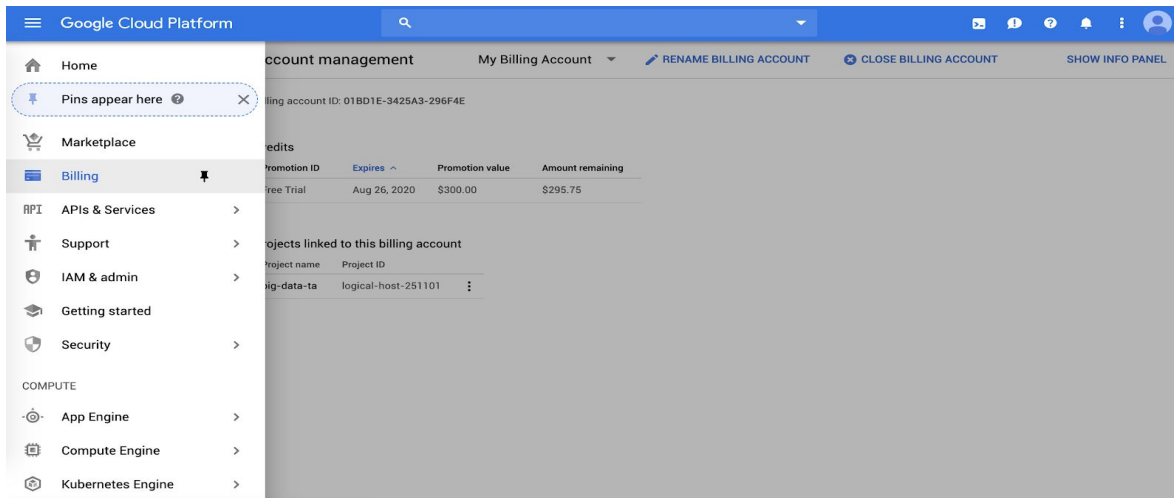
The goals of this assignment are to (1) become familiar with one of the most popular cloud computing platforms: [Google Cloud Platform](https://cloud.google.com/) (GCP), (2) have hands-on exposure to several [big data products](#) offered by GCP: Cloud Storage, BigQuery, and Dataproc (Hadoop and Spark), (3) lay the foundations for the rest of the semester.

In this assignment, you will setup your GCP account and environment, create a Spark cluster, load data from Cloud Storage and BigQuery, and process data with Spark. Note that this assignment is “documentation heavy”, which is inevitable since you have to learn how to use tools on GCP. Later assignments should be more focused on data analysis and algorithm implementation.

Warm-up exercises

1. GCP account setup

- a. Head over to GCP and make sure you have a google account.
<https://cloud.google.com>
- b. Apply for \$300 credit for a year.
<https://cloud.google.com/free/>
- c. Go to Billing -> Account management to check your credit. You should expect \$300 in the “Promotional value”.



2. Install Cloud SDK

Google Cloud SDK is a set of tools that you can use to manage resources and applications hosted on Google Cloud Platform. It should be handy if you have this installed on your local computer.

- Choose a tutorial that is suitable for your local environment.
<https://cloud.google.com/sdk/docs/quickstarts>
- Follow the tutorial thoroughly and install SDK on your computer. Remember to create a GCP project first as noted in the tutorials.
- You should expect similar info pop up when you enter the command:

```
[MacBook-Pro-4:~]$ gcloud info
Google Cloud SDK [299.0.0]

Platform: macOS [x86_64] ('Darwin', 'dyn-160-39-134-97.dyn.columbia.edu', '18.7.0', 'Darwin Kernel Version 18.7.0: Tue Aug 20 16:57:14 PDT 2019; root:xnu-4903.202.2/RELEASE_ARM_T8020')
Python version: [2.7.14 |Anaconda, Inc.| (default, Oct 5 2017, 02:28:52) [GCC 4.2.1 Compatible Clang 4.0.1 (tags/RELEASE_401/final)]]
Python Location: [/Users/frank/miniconda2/bin/python2]
Site Packages: [Disabled]
```

3. Dataproc

Dataproc is an on-demand, fully managed cloud service for running Apache Hadoop and Spark on GCP. By using Dataproc, we don't need to maintain the underlying infrastructure, and we could easily increase / decrease the number of resources we want. It also offers built-in integration with other services on GCP like Cloud Storage and BigQuery. In this exercise, you'll create a Spark cluster, and submit a job that runs a Spark example program - "Pi calculation".

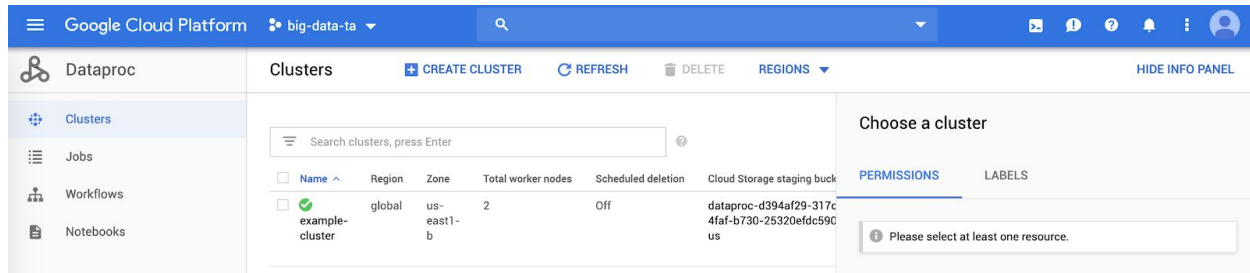
- Learn how to create a cluster and run a Spark job by following this tutorial.

<https://cloud.google.com/dataproc/docs/quickstarts/quickstart-gcloud>

Note that GCP provides different ways to achieve the same goal, like

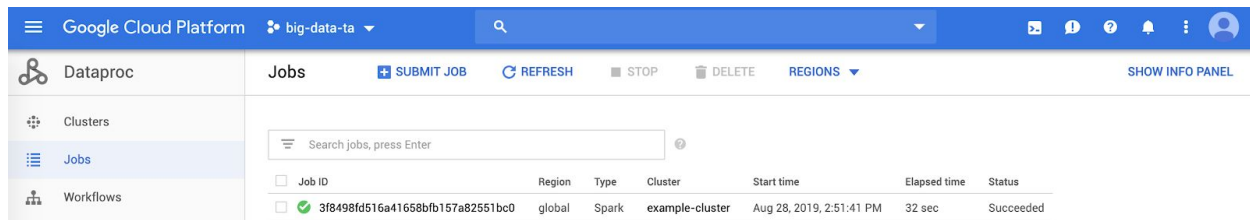
using the API explorer or console, but we believe it is beneficial to use the command-line tool.

- b. At the end of the tutorial, you should expect the following things:
 - i. You've created a computing cluster with 3 nodes (1 master + 2 workers):



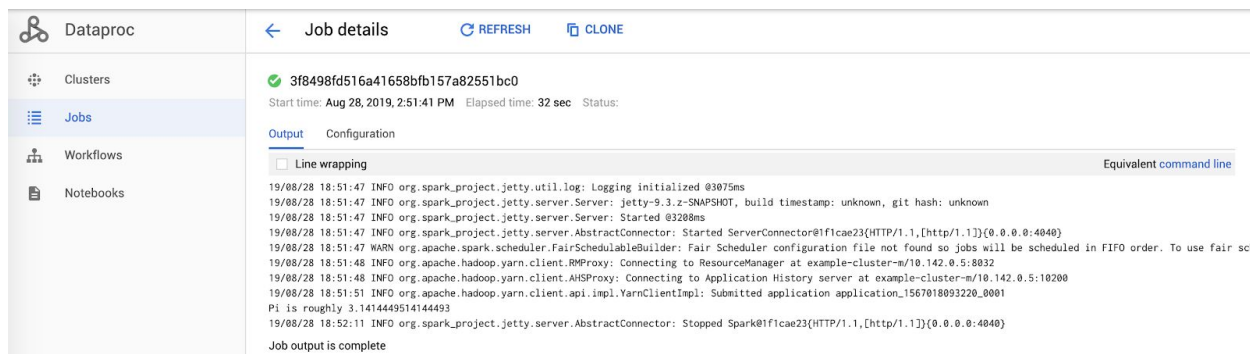
The screenshot shows the Google Cloud Platform interface for Dataproc Clusters. On the left, a sidebar lists 'Clusters', 'Jobs', 'Workflows', and 'Notebooks'. The main panel shows a table of clusters. One cluster, 'example-cluster', is highlighted. To the right, a 'Choose a cluster' dialog is open, displaying a message: 'Please select at least one resource.'

- ii. You've submitted a Spark job:



The screenshot shows the Google Cloud Platform interface for Dataproc Jobs. The main panel displays a table of jobs. One job is listed with ID '3f8498fd516a41658bfb157a82551bc0', type 'Spark', and status 'Succeeded'.

- iii. You've successfully calculated the value of Pi:



The screenshot shows the Google Cloud Platform interface for Dataproc Job details. The 'Output' tab is selected, showing a log of the job execution. The log includes the following text:

```

19/08/28 18:51:47 INFO org.spark_project.jetty.util.log: Logging initialized @3075ms
19/08/28 18:51:47 INFO org.spark_project.jetty.server.Server: jetty-9.3.z-SNAPSHOT, build timestamp: unknown, git hash: unknown
19/08/28 18:51:47 INFO org.spark_project.jetty.server.Server: Started @3208ms
19/08/28 18:51:47 INFO org.spark_project.jetty.server.AbstractConnector: Started ServerConnector@1f1cae23(HTTP/1.1,[http/1.1])(0.0.0.0:4040)
19/08/28 18:51:47 WARN org.apache.spark.scheduler.FairSchedulableBuilder: Fair Scheduler configuration file not found so jobs will be scheduled in FIFO order. To use fair sch
19/08/28 18:51:48 INFO org.apache.hadoop.yarn.client.RMPProxy: Connecting to ResourceManager at example-cluster-m/10.142.0.5:8032
19/08/28 18:51:48 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at example-cluster-m/10.142.0.5:10200
19/08/28 18:51:51 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1567018093220_0001
Pi is roughly 3.1414449514144493
19/08/28 18:52:11 INFO org.spark_project.jetty.server.AbstractConnector: Stopped Spark@1f1cae23(HTTP/1.1,[http/1.1])(0.0.0.0:4040)
Job output is complete
  
```

Optional reading:

Create a cluster with Jupyter Notebook access:

<https://cloud.google.com/dataproc/docs/tutorials/jupyter-notebook>

Create a cluster with single node to save money:

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

In short, you could create a single node cluster with Jupyter Notebook access by using a command similar to this:

```

MacBook-Pro-4:hw1 frank$ gcloud beta dataproc clusters create example-cluster --optional-components=ANACONDA,JUPYTER --image-version=1.3 --enable-component-gateway --bucket big_data_ta --project logical-host-251101 --single-node --metadata 'PIP_PACKAGES=graphframes=0.6' --initialization-actions gs://dataproc-initialization-actions/python/pip-install.sh
  
```

Take a look at the source code and examples on the official Spark site:

<https://spark.apache.org/examples.html>

4. “Word Count” using Google Cloud Storage and Spark

Cloud Storage is a file storage system built by Google for GCP. It’s an Infrastructure as a Service (IaaS), comparable to AWS S3 service. It has [pros and cons](#) compared to the standard HDFS on Hadoop. It is handy to perform operations with Cloud Storage and Spark on GCP.

- a. Study the [Spark programming guide](#).
- b. Learn how to write and execute a word-count spark program with Cloud Storage and Spark.
<https://cloud.google.com/dataproc/docs/tutorials/gcs-connector-spark-tutorial>
- c. The result should be similar to:

```
wordCounts.take(5)
```

```
[(u'a', 2), (u'we', 1), (u'would', 1), (u'What's", 1), (u'sweet.', 1)]
```

Optional reading:

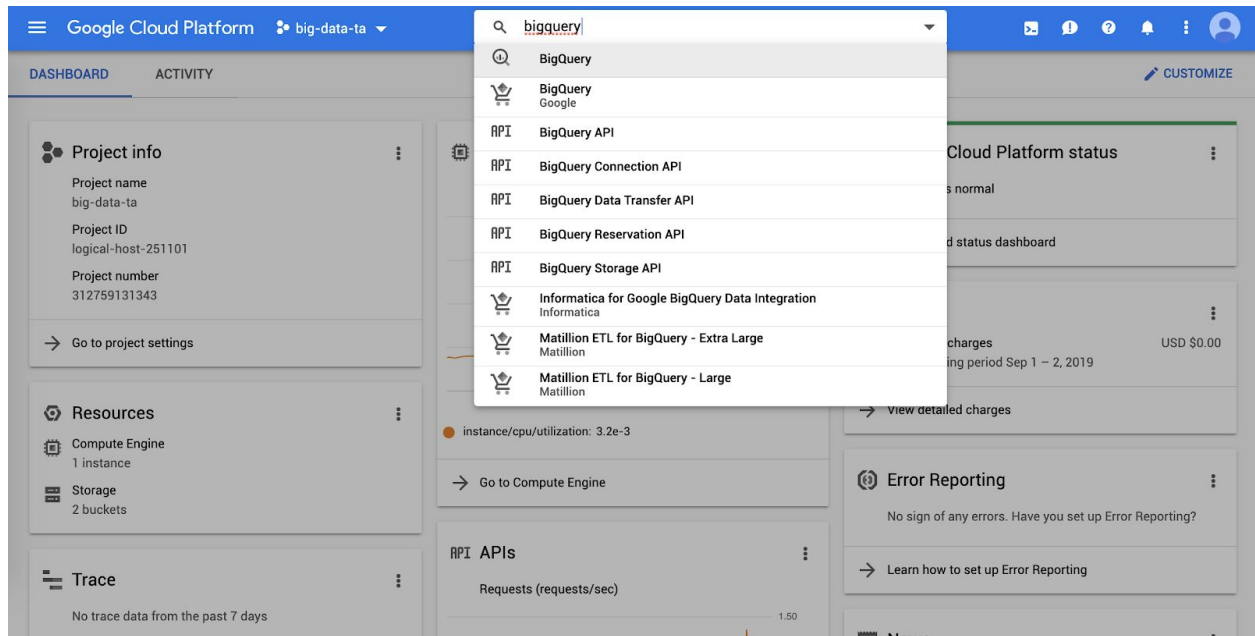
Learn gsutil tool to interact with Cloud Storage:

<https://cloud.google.com/storage/docs/gsutil>

5. BigQuery

BigQuery is a data warehouse solution developed by Google for GCP. Like Apache Hive, they both are data warehouse software. Learn more about how they compare in [this article](#). In BigQuery, you could write SQL-like queries to interact with data from different sources.

- a. Explore BigQuery and learn how to use front-end UI to perform query.
<https://cloud.google.com/bigquery/docs/bigquery-web-ui#overview>



- b. Learn how to load data into BigQuery.
<https://cloud.google.com/bigquery/docs/loading-data>
- c. Check out the BigQuery SQL documentation as a reference.
<https://cloud.google.com/bigquery/docs/reference/standard-sql/query-syntax>

Optional reading:

Take a look at how to query BigQuery data:

<https://cloud.google.com/bigquery/docs/query-overview>

Trace the code to know how to use BigQuery with Spark:

<https://cloud.google.com/dataproc/docs/tutorials/bigquery-connector-spark-example>

Remember to delete your dataproc clusters when you finish executions to save money.

Homework Submissions

1. Warm-up exercises:

For the “Pi calculation” in exercise 3 and “word count” example in exercise 4:

- (1) Provide screenshots to prove you’ve completed the exercises. (10%)
- (2) List the Spark [transformations and actions](#) involved in each exercise. Identify the RDD operation that triggers the program to execute. (20%)

2. NYC Bike expert (30%)

Download the NYC Citi bike data [here](#), upload the csv file to a Cloud Storage bucket, then load the data into BigQuery (check the [documentation](#)). You could choose “auto detect” schema when creating the table. Answer the following questions and provide screenshots of your queries:

- (1) How many unique *station_ids* are there in the dataset?
- (2) What’s the largest *capacity* for a station? List all the *station_id* of the stations that have the largest capacity?
- (3) What’s the total number of bikes available in *region_id* 70?

3. Understanding William Shakespeare (40%)

Write a Spark program to find the 5 most frequent words of the [provided shakes.txt file](#) extracted from Shakespeare’s works. Provide your results (words and its frequency) and take a screenshot of your code / jupyter notebook:

- (1) Find top 5 frequent words without any text preprocessing.
- (2) Find top 5 frequent words by first filtering out stop words provided by NLTK package. The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs to conduct natural language processing in Python.