Leveraging Unstructured Data - Lab 2 : Work with structured and semi-structured data v1.3

## Overview

Structured data has a useful organization or schema. Unstructured data includes not only data that is without a schema, but also data that has some structure, but that structure is not useful for the intended analysis or query.

In this lab, you will explore the native Hadoop ecosystem tools used for working with structured and semi-structured data, Hive and Pig.

## Objectives

In this lab, you will perform the following tasks:

* Use the Hive CLI and run a Pig job
* Hive is used for structured data, similar to SQL
* Pig is used for semi-structured data, similar to SQL + scripting

## Task 1: Preparation

A Dataproc cluster has been prepared for you. If you login to GCP before the progress bar reports that the "Lab is Running", you may have to wait several minutes for the cluster to transition from "Provisioning" to "Running" before the cluster completes setup.

You will be performing most of the lab steps from the Master Node of the cluster in an SSH terminal window.

1. In the Console, on the **Navigation menu** (7a91d354499ac9f1.png), click **Dataproc** > **Clusters**.
2. Locate the cluster named **dataproc-cluster**. Which region and zone is it located in? The region and zone have been selected automatically for you by Qwiklabs.
3. Notice the Cloud Storage staging bucket defined for this cluster. This bucket has the same name as the project ID, which is a convenient way to make the name globally unique.
4. Click on the name **dataproc-cluster** to go to the Cluster details page.
5. The Cluster details page opens to the **Monitoring** tab. Click on the tab labeled **VM Instances**.

### **Open the Master Node terminal**

1. On the line for the VM named **dataproc-cluster-m** you will see that it has the Role of Master and there is an SSH link next to it. Click on **SSH** to open a terminal window to the Master Node.

## Task 2: Enable secure web access to the Dataproc cluster

### **Create a restrictive firewall rule using Target tags, IP address, and protocol**

Create a firewall rule that allows access only to the Master Node from your computer's IP address. Only ports 8088 (Hadoop Job Interface) and 9870 (Hadoop Admin interface) will be permitted.

### **Verify that the network tag is set on the Master Node**

Verify that the network tag **hadoopaccess** is set on the Master Node. That will apply the firewall rule to the Master Node, giving your laptop access to it.

1. In the Console, on the **Navigation menu** (7a91d354499ac9f1.png), click **Compute Engine** > **VM Instances**.
2. Click on the Master Node, **dataproc-cluster-m**.
3. Verify that under Network Tags it lists **hadoopaccess**.
4. If the tag is not there, click **EDIT**.
5. Under Network Tags add the tag: **hadoopaccess**
6. Click **Save**.

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

You will use the browser IP address to allow your local browser to connect to the Dataproc cluster.

1. Find your computer's browser IP address by opening a browser window and viewing <https://whatismyip.com/>. Copy the Public IPv4 address.

### **Create the firewall rule**

1. In the Console, on the **Navigation menu** (7a91d354499ac9f1.png) click **VPC Network** > **Firewall rules**.
2. Click **Create Firewall Rule**.
3. Specify the following, and leave the remaining settings as their defaults:

|  |  |
| --- | --- |
| **Property** | **Value(type value or select option as specified)** |
| **Name** | allow-hadoop |
| **Network** | default |
| **Priority** | 1000 |
| **Direction of traffic** | Ingress |
| **Action on match** | Allow |
| **Targets** | Specified target tags |
| **Target tags** | hadoopaccess |
| **Source IP ranges** | <your-IP>/32 |
| **Specified protocols and ports** | Check tcp and enter port number 9870,8088 |

1. Click **Create**.

It will take a few minutes for the firewall rule to become active.

Click Check my progress to verify the objective.

Enable secure web access to the Dataproc cluster

Check my progress

## Task 3: Prepare the data for Hive

### **Copy sample files to the Master node home directory**

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

1. In the Master Node SSH terminal window.

cd

cp -r /training .

ls

1. In the Master Node SSH terminal window. You should now have the /training directory in your home directory. And it should have files within it.

ls

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

ls pet\*.\*

### **Copy the data file to HDFS**

1. View the structured data in the text file.

cat pet-details.txt

The data is comma delimited. It is organized into fields and rows.

1. Stage the data in HDFS.

hadoop fs -mkdir /pet-details

hadoop fs -put pet-details.txt /pet-details

### **View the data files using the Hadoop Administration interface**

1. In the Console, on the **Navigation menu** (7a91d354499ac9f1.png) click **Compute Engine** > **VM Instances**.
2. In the list of VM instances, in the row for **dataproc-cluster-m**, highlight the **External IP** and copy it.
3. Open a new browser tab or window and paste the External IP. Add ":9870" after the IP and press enter. Example: <External IP>:9870
4. You should now see the Hadoop Administration interface. Under **Utilities**, click on **Browse the file system**. Click on the folder **/pet-details**.
5. Notice that the file **pet-details.txt** is inside **/pet-details**.
6. Leave the Hadoop Administration interface open. You will return to it in later steps.

Click Check my progress to verify the objective.

Prepare the data for Hive

Check my progress

## Task 4: Explore Hive using the Hive interactive CLI

### **Use HIVE to access the data in HDFS as if it were in a database**

Hive provides a subset of SQL. The way it does this is by maintaining metadata to define a schema on top of the data. This is one way to work with a large amount of distributed data in HDFS using familiar SQL syntax.

1. In the master node SSH window, make sure you are in the right directory and start the Hive CLI interpreter.

hive

1. In the hive interpreter, create a database. This creates the metadata schema for a database.

CREATE DATABASE pets;

USE pets;

1. Create a table. Notice that the comma delimiter is specified in table creation, not at data ingest.

CREATE TABLE details (Type String, Name String, Breed String, Color String, Weight Int) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

SHOW TABLES;

DESCRIBE pets.details;

1. Establish the relationship between the metadata schema and the data in HDFS.

load data INPATH '/pet-details/pet-details.txt' OVERWRITE INTO TABLE details;

1. Verify that everything is working.

SELECT \* FROM pets.details;

1. Quit the hive interpreter.

quit;

### **Use the Hadoop Administration interface to see how hive works**

Hive ingested the **pet-details.txt** file into a data warehouse format requiring a schema. You will use the Hadoop Administration interface to see this transformation.

1. Return to the Hadoop Administration interface in the browser.
2. Under **Utilities**, click on **Browse the file system**. Click on the folder **/pet-details**. The file **pet-details.txt** is gone.
3. Under **Utilities**, click on **Browse the file system**. Then click on **user > hive > warehouse > pets.db** > **details**. The file **pet-details.txt** has been moved to this location.

Hive is designed for batch jobs and not for transactions. It ingests data into a data warehouse format requiring a schema. It does not support real-time queries, row-level updates, or unstructured data. Some queries may run much slower than others due to the underlying transformations Hive has to implement to simulate SQL.

## Task 5: Run a Pig job

### **Run a Pig job**

1. In the master node's SSH window, view the Pig application.

cat pet-details.pig

In line 'x1', the load statement in the application creates a schema on top of the HDFS data file. Lines 'x2' through 'x5' perform transformations on the data. And the last line stores the result in a folder called **/GroupedByType** in HDFS.

1. The application expects to find the ingest file in HDFS in the directory **/pet-details**. Make another copy of the data at that location.

hadoop fs -put pet-details.txt /pet-details

1. Run the application:

pig < pet-details.pig

1. Return to the browser tab containing the Hadoop Applications interface and refresh it, or reopen it with <External-IP>:8088. Notice that Pig generated a Java MapReduce job which is running on the cluster. Click the browser refresh button to watch for job completion.
2. Return to the browser tab containing the Hadoop Administration interface and refresh it, or reopen it with <External-IP>:9870. Under **Utilities**, click on **Browse the file system**. In the resulting list, click on **GroupedByType**. This is the output directory specified in the Pig application. The file named **part-r-00000** is the HDFS file containing the output. You cannot view the contents from here. First, you must download that part to the local file system
3. Return to the SSH terminal on the Master node, cloud-dataproc-m and make a local output directory and retrieve the results from HDFS.

cd

mkdir output

cd output

hadoop fs -get /GroupedByType/part\* .

1. Now you can view the results of the Pig job.

cat part-r-00000

Pig provides SQL primitives similar to Hive, but in a more flexible scripting language format. Pig can also deal with semi-structured data, such as data having partial schemas, or for which the schema is not yet known. For this reason it is sometimes used for Extract Transform Load (ETL). It generates Java MapReduce jobs. Pig is not designed to deal with unstructured data.