Recommendations ML with Dataproc

# Overview

In this lab you use Dataproc to train a recommendations machine learning model based on users' previous ratings. You then apply that model to create a list of recommendations for every user in the database.

# Objectives

In this lab you learn how to perform the following tasks:

* Launch Dataproc
* Train and apply ML model written in PySpark to create product recommendations
* Explore inserted rows in Cloud SQL

# Task 1: Create Assets

In this task you will download assets to your LOD machine and stage some files onto Cloud Storage.

1. Extract the downloaded zip bundle in-situ
2. Navigate to the downloads\day3\spark\_ml folder to verify the content is as show…

A screenshot of a computer

Description automatically generated

1. Log into Google Cloud
2. Navigate to Cloud Storage and delete any buckets remaining from previous labs
3. Open Cloud Shell
4. Upload the extracted SparkML folder to cloud shell. Move into the SparkML directory
5. Run the following commands to create a storage bucket and upload files..

**export BUCKET=$(gcloud config get-value project)**

**gsutil mb "gs://$BUCKET"**

**gsutil cp \*.csv gs://$BUCKET/sql/**

**gsutil cp \*.sql gs://$BUCKET/sql/**

1. Confirm that you have the following….

A screenshot of a computer

Description automatically generated

# Task 2: Create Cloud SQL for MySQL instance

In this task you create a Cloud SQL for MySQL instance to host the recommendation\_spark database that you will create in Task 3.

1. In the Cloud Console, on the Navigation menu ( ), click SQL (in the Databases section).
2. Click **Create Instance**.
3. Click Choose **MySQL**.
4. On the Create a MySQL instance page, specify the following, and leave the remaining settings as their defaults:

**Property Value**

Instance ID **rentals**

Password **easyPassword1@**

Region **us-central1 (Iowa)**

1. Select **Enterprise** as the Cloud SQL Edition
2. Click into the ‘Choose a preset for this edition…. And select **Sandbox**….

A screenshot of a cloud version

Description automatically generated

1. Expand ‘SHOW CONFIGURATION OPTIONS’
2. Under **Data Protection**, disable **Automate daily backups** and uncheck **Enable deletion protection**
3. Click on **CREATE INSTANCE**. The instance creation will take 3-4 minutes.
4. Whilst waiting, we need to identify the IP address that your cloud shell currently using as we must allow this address to communicate to the Cloud SQL instance.
5. In cloud shell, run the following command and note the output…
6. Switch back to the Cloud Console. Once the SQL Instance is created, In the Configuration section, select **Edit Configuration**…

A screenshot of a computer

Description automatically generated

1. Scroll down to the **‘Connections’** drop down and expand
2. Click on ‘**ADD A NETWORK’**
3. Name the New network ‘**shell**’, enter your IP address followed by ‘/32’ and click on ‘**DONE**’…
4. Scroll down and click on ‘**SAVE**’

# Task 3: Create and populate tables

In this task you authorize access to the Cloud Storage bucket for the Cloud SQL service account, and then import the table definitions from your Cloud Storage bucket.

1. Scroll down and copy the service account credentials. This will have the format of **xxxxxxxxxxx-xxxxx@gcp-sa-cloud-sql.iam.gserviceaccount.com**.
2. In Cloud Shell, define a variable to hold the service account name. Replace the value COPIED\_SERVICE\_ACCOUNT in the example below with your service account name:

**set MEMBER\_NAME=COPIED\_SERVICE\_ACCOUNT**

1. Grant the Cloud SQL service account the role storage.objectViewer to allow Cloud SQL to import the data from the Cloud Storage bucket:

**gsutil iam ch serviceAccount:%MEMBER\_NAME%:roles/storage.objectViewer gs://YOUR\_BUCKET\_NAME\_HERE**

1. Click AUTHORISE if prompted
2. In the Cloud Console, on the Cloud SQL Overview page, click **Import**.
3. Click Browse. This will bring up a list of buckets. Click on the bucket you created, then navigate into **sql**, click **table\_creation.sql**, then click Select.
4. Click Import.

**Note**: If .sql or .csv files import fails in first attempt, try again. You can scroll down to the Operations and logs section which shows the status of Create finished if the import completes successfully.

1. Click Import.
2. Click Browse, navigate into sql, click **accommodation.csv**, and then click Select.
3. In Import data from Cloud Storage, specify the following values and leave the remaining as defaults:

**Property Value**

File format **CSV**

Database **recommendation\_spark**

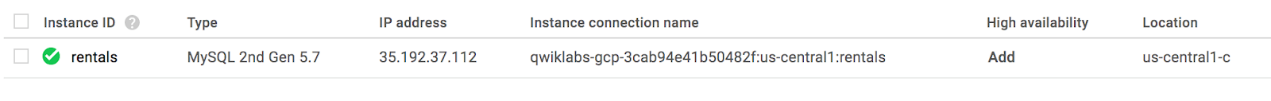
Table **Accommodation**

1. Click Import.
2. Repeat the Import process (steps 9 - 12) for **rating.csv**, but for Table, type **Rating**.

# Task 4: Launch Dataproc

In this task you launch Dataproc and configure it so that each of the machines in the cluster can access Cloud SQL.

1. In the Cloud Console, on the Navigation menu ( ), click SQL and note the location of your Cloud SQL instance:



In the above image, the location is us-central1-c

1. In the Cloud Console, on the Navigation menu ( ), click Dataproc and click Enable API if prompted. Once enabled, click **Create cluster on Compute Engine**.
2. In Set up cluster, specify the following values and leave the remaining as defaults:

**Property Value**

Name **cluster-1**

Region **us-central1**

Zone The same zone as your Cloud SQL instance

1. For **Cluster type**, select **Standard (1 master, N workers)**
2. Scroll down to Components
3. In Create a cluster, click **Configure nodes (optional)**.
4. For **Manager node**, select **N1** Series, and n1**-standard-2** as the Machine type.
5. Repeat step 7 for **Worker nodes**
6. Leave all other values with their default and click **Create**.

It will take 1-2 minutes to provision your cluster.

1. Navigate to Compute Engine and you will see the cluster node being created.
2. Identify and authorize the nodes to access SQL by running the following in cloud shell replacing placeholders <> with appropriate values…

**bash authorize\_dataproc.sh <Cluster-Name> <Zone> <Total-Worker-Nodes>**

# Task 5: Run ML model

In this task you copy the Cloud SQL public IP address to the train\_and\_apply.py file and then create a trained model to apply it to all the users in the system.

1. In the Cloud Shell, define an environment variable named SQLIP and obtain the Cloud SQL Public IP address:

**export SQLIP=$(gcloud sql instances describe rentals --format="value (ipAddresses[0].ipAddress)")**

1. If prompted… enter ‘Y’ to enable the sqlapi and then re-enter the above command
2. Verify the Cloud SQL IP address was written to the environment variable:

**echo $SQLIP**

You should see the public IP address of the Cloud SQL server.

1. Replace the placeholder IP address in the **train\_and\_apply.py** file:

**sed -i "s/104.155.188.32/$SQLIP/" ./train\_and\_apply.py**

1. Verify the Cloud SQL public IP address was written to the train\_and\_apply.py file:

**grep -i 'CLOUDSQL\_INSTANCE\_IP =' ./train\_and\_apply.py**

You should see CLOUDSQL\_INSTANCE\_IP = followed by the public IP address of the Cloud SQL server.

1. Copy the train\_and\_apply.py file to your Cloud Storage bucket:

**gsutil cp ~/training-data-analyst/CPB100/lab3b/sparkml/tr\*.py gs://$BUCKET/**

1. Display the location of your main python file:

**echo "The location of your main python file is: gs://$BUCKET/train\_and\_apply.py"**

Record this as you will need it in the following steps.

1. In the Dataproc console, click Jobs.

Click Submit job.

1. In Job type, specify the following values and leave the remaining as defaults:

**Property Value**

Cluster select your cluster from the drop down menu

Job type **PySpark**

Main python file as you recorded in the previous step

Max restarts per hour **1**

1. Click Submit and wait for the job Status to change from Running (this will take up to 5 minutes) to Succeeded.

If the job Failed, please troubleshoot using the logs and fix the errors. You may need to re-upload the changed Python file to Cloud Storage and clone the failed job to resubmit.

# Task 6: Explore inserted rows

In this task you explore the explore the list of recommendations for a user.

1. In the Cloud Console, on the Navigation menu ( ), click SQL (in the Storage section).
2. Click rentals to view details related to your Cloud SQL instance.
3. Under Connect to this instance section, click Open Cloud Shell to connect using gcloud. In Cloudshell tab press Enter.

It will take a few minutes to Allowlist your IP for incoming connection.

1. When prompted, type the root password, **easyPassword1@** and then press Enter.
2. At the mysql prompt, type:

**use recommendation\_spark;**

This sets the database in the mysql session.

1. Find the recommendations for some user:

**select r.userid, r.accoid, r.prediction, a.title, a.location, a.price, a.rooms, a.rating, a.type from Recommendation as r, Accommodation as a where r.accoid = a.id and r.userid = 10;**

These are the five accommodations that we would recommend to her. Note that the quality of the recommendations are not great because our dataset was so small (note that the predicted ratings are not very high). Still, this lab illustrates the process you'd go through to create product recommendations. Experiment by changing the **userid** number and re-running the script.

# End of lab clean-up

To conserve costs, delete the following resources…

* Dataproc cluster
* SQL Instance
* Cloud Storage buckets: {Project-ID} and all buckets named ‘dataproc-\*’