

Overview (Qwiklabs)

Set up your environments

Task 1. Create a Cloud SQL instance

Task 2. Create tables

Task 3. Stage data in Cloud Storage

Task 4. Load data from Cloud Storage into Cloud SQL tables

Task 5. Explore Cloud SQL data

Task 6. Launch Dataproc

Task 7. Run the ML model

Task 8. Run your ML job on Dataproc


Task 9. Explore inserted rows with SQL

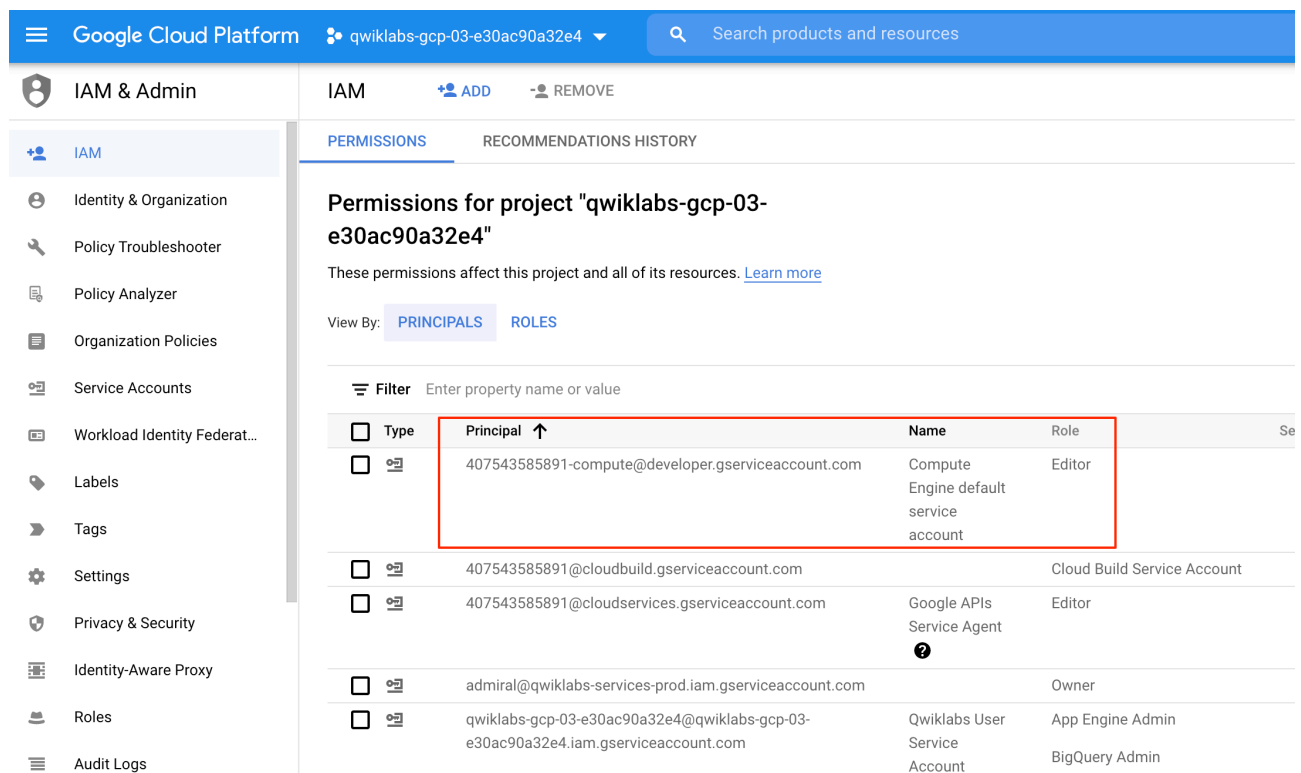
In this lab, you populate rentals data in Cloud SQL for the rentals recommendation engine to use. The recommendations engine itself will run on Dataproc using Spark ML.

Set up your environments

1. Check project permissions

Before you begin your work on Google Cloud, you need to ensure that your project has the correct permissions within Identity and Access Management (IAM).

1. In the Google Cloud console, on the **Navigation menu** () , click **IAM & Admin** > **IAM**.
2. Confirm that the default compute Service Account `{project-number}-compute@developer.gserviceaccount.com` is present and has the `editor` role assigned. The account prefix is the project number, which you can find on **Navigation menu** > **Home**.



The screenshot shows the Google Cloud Platform console with the IAM & Admin section selected. The left sidebar lists various IAM-related tools. The main content area displays the 'Permissions for project "qwiklabs-gcp-03-e30ac90a32e4"' page. The 'PRINCIPALS' tab is active, showing a table of service accounts and their roles. A red box highlights the first entry, which is the default compute service account with the 'Editor' role.

Type	Principal ↑	Name	Role	Se
<input type="checkbox"/>	407543585891-compute@developer.gserviceaccount.com	Compute Engine default service account	Editor	
<input type="checkbox"/>	407543585891@cloudbuild.gserviceaccount.com		Cloud Build Service Account	
<input type="checkbox"/>	407543585891@cloudservices.gserviceaccount.com	Google APIs Service Agent	Editor	
<input type="checkbox"/>	admiral@qwiklabs-services-prod.iam.gserviceaccount.com		Owner	
<input type="checkbox"/>	qwiklabs-gcp-03-e30ac90a32e4@qwiklabs-gcp-03-e30ac90a32e4.iam.gserviceaccount.com	Qwiklabs User Service Account	App Engine Admin BigQuery Admin	

If the account is not present in IAM or does not have the `editor` role, follow the steps below to assign the required role.

- In the Google Cloud console, on the **Navigation menu**, click **Home**.
- Copy the project number (e.g. 729328892908).

- On the **Navigation menu**, click **IAM & Admin > IAM**.
- At the top of the **IAM** page, click **Add**.
- For **New principals**, type:

{project-number}-compute@developer.gserviceaccount.com

Replace {project-number} with your project number.

- For **Role**, select **Project** (or **Basic**) > **Editor**. Click **Save**.

Task 1. Create a Cloud SQL instance

1. In the Google Cloud Console, Select **Navigation menu > SQL** (in the Databases section).
2. Click **Create instance**.
3. Click **Choose MySQL**.
4. For **Instance ID**, type **rentals**.

Instance ID
ID is permanent. Use lowercase letters and numbers.

5. Scroll down and specify a **Root password**. Before you forget, note down the root password.
6. For **Region** select **us-central1**.
7. Click **Create instance** to create the instance. It will take a minute or so for your Cloud SQL instance to be provisioned.

Task 2. Create tables

1. Creation of Three Tables: Recommendation, Rating and Accommodation

```

CREATE DATABASE IF NOT EXISTS recommendation_spark;
USE recommendation_spark;
DROP TABLE IF EXISTS Recommendation;
DROP TABLE IF EXISTS Rating;
DROP TABLE IF EXISTS Accommodation;
CREATE TABLE IF NOT EXISTS Accommodation
(
  id varchar(255),
  title varchar(255),
  location varchar(255),
  price int,
  rooms int,
  rating float,
  type varchar(255),
  PRIMARY KEY (ID)
);
CREATE TABLE IF NOT EXISTS Rating
(
  userId varchar(255),
  accoId varchar(255),
  rating int,
  PRIMARY KEY(accoId, userId),
  FOREIGN KEY (accoId)
    REFERENCES Accommodation(id)
);
CREATE TABLE IF NOT EXISTS Recommendation
(
  userId varchar(255),
  accoId varchar(255),
  prediction float,
  PRIMARY KEY(userId, accoId),
  FOREIGN KEY (accoId)
    REFERENCES Accommodation(id)
);
SHOW DATABASES;

```

Be careful of every function of these three tables:

- 1) When a user rates a house (giving it four stars for example), an entry is added to the Rating table.
- 2) General information about houses, such as the number of rooms they have and their average rating is stored in the Accommodation table.
- 3) The job of the recommendation engine is to fill out the Recommendation table for each user and house: this is the predicted rating of that house by that user.

2. In **Cloud SQL**, click **rentals** to view instance information.

Connect to the database

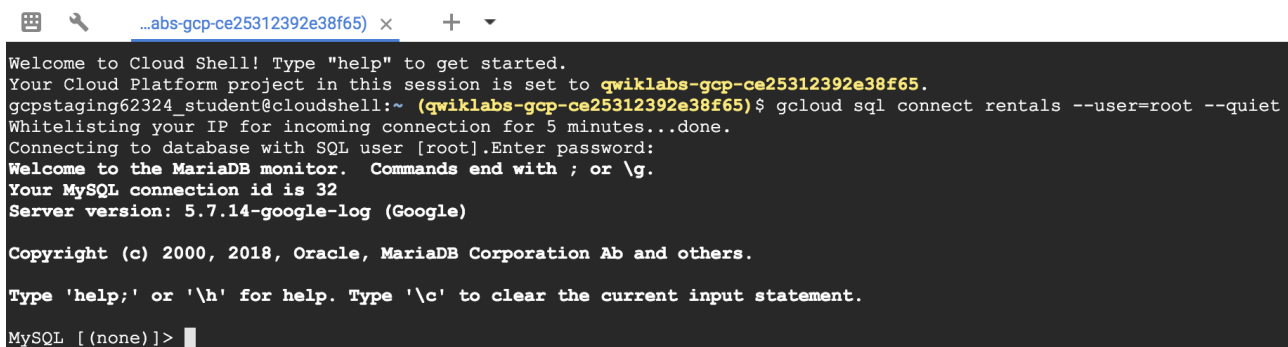
1. Find the **Connect to this instance** box on the page and click on **Open Cloud Shell**.

2. If required, click **Continue**. Wait for Cloud Shell to load.
3. Once Cloud Shell loads, you will see the below command already typed:
 - `gcloud sql connect rentals --user=root --quiet`
4. Press **ENTER**.
5. Wait for your IP Address to be whitelisted.

Allowlisting your IP for incoming connection for 5 minutes...:

6. When prompted, enter your password and press **ENTER** (note: you will not see your password typed in or even ****).

You can now run commands against your database!



```
...abs-gcp-ce25312392e38f65) x +  
Welcome to Cloud Shell! Type "help" to get started.  
Your Cloud Platform project in this session is set to qwiklabs-gcp-ce25312392e38f65.  
gcpstaging62324_student@cloudshell:~ (qwiklabs-gcp-ce25312392e38f65)$ gcloud sql connect rentals --user=root --quiet  
Whitelisting your IP for incoming connection for 5 minutes...done.  
Connecting to database with SQL user [root].Enter password:  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MySQL connection id is 32  
Server version: 5.7.14-google-log (Google)  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MySQL [(none)]> █
```

7. Run the following command:

SHOW DATABASES;

You should see the default system databases:

```
+-----+
| Database           |
+-----+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+-----+
```

Note: You must always end your mySQL commands with a semi-colon
;

8. Copy and paste the below SQL statement you analyzed earlier into the command line.

```
CREATE DATABASE IF NOT EXISTS recommendation_spark;
USE recommendation_spark;
DROP TABLE IF EXISTS Recommendation;
DROP TABLE IF EXISTS Rating;
DROP TABLE IF EXISTS Accommodation;
CREATE TABLE IF NOT EXISTS Accommodation
(
  id varchar(255),
  title varchar(255),
  location varchar(255),
  price int,
  rooms int,
  rating float,
  type varchar(255),
  PRIMARY KEY (ID)
);
CREATE TABLE IF NOT EXISTS Rating
(
  userId varchar(255),
  accoId varchar(255),
  rating int,
  PRIMARY KEY(accoId, userId),
  FOREIGN KEY (accoId)
    REFERENCES Accommodation(id)
);
CREATE TABLE IF NOT EXISTS Recommendation
(
  userId varchar(255),
  accoId varchar(255),
  prediction float,
  PRIMARY KEY(userId, accoId),
  FOREIGN KEY (accoId)
```

```
REFERENCES Accommodation(id)
);
SHOW DATABASES;
```

9. Press **ENTER**.

10. Confirm that you now see `recommendation_spark` as a database:

```
+-----+
| Database          |
+-----+
| information_schema |
| mysql             |
| performance_schema |
| recommendation_spark |
| sys               |
+-----+
```

11. Run the following command to show the tables:

```
USE recommendation_spark;
SHOW TABLES;
```

12. Press **ENTER**.

13. Confirm that you see the three tables:

```
+-----+
| Tables_in_recommendation_spark |
+-----+
| Accommodation                  |
| Rating                        |
| Recommendation                  |
+-----+
```

13. Run the following query:

```
SELECT * FROM Accommodation;
```

You'll get Empty set (0) in the Accommodation table?

Task 3. Stage data in Cloud Storage

Option 1: Use the command line

1. Open a new Cloud Shell tab (**do not use your existing mySQL Cloud Shell tab**).
2. Copy and paste the following command:

```
echo "Creating bucket: gs://$DEVSHHELL_PROJECT_ID"
gsutil mb gs://$DEVSHHELL_PROJECT_ID
echo "Copying data to our storage from public dataset"
gsutil cp gs://cloud-training/bdml/v2.0/data/accommodation.csv gs://$DEVSHHELL_PROJECT_ID
gsutil cp gs://cloud-training/bdml/v2.0/data/rating.csv gs://$DEVSHHELL_PROJECT_ID
echo "Show the files in our bucket"
gsutil ls gs://$DEVSHHELL_PROJECT_ID
echo "View some sample data"
gsutil cat gs://$DEVSHHELL_PROJECT_ID/accommodation.csv
```

3. Press **ENTER**.

Option 2: Use the Cloud Console UI

Skip these steps if you have already loaded your data using the command line.

1. Navigate to **Storage** and select **Cloud Storage > Browser**.
2. Click **Create Bucket** (if one does not already exist).

3. Specify your project name as the bucket name.
4. Click **Create**.
5. Download the below files locally and then upload them inside of your new bucket:

<https://storage.googleapis.com/cloud-training/bdml/v2.0/data/accommodation.csv>

<https://storage.googleapis.com/cloud-training/bdml/v2.0/data/rating.csv>

Task 4. Load data from Cloud Storage into Cloud SQL tables

1. Navigate back to **SQL**.
2. Click on **rentals**.

Import accommodation data

1. Click **Import** (top menu).
2. Specify the following:
 - Source: Click **Browse** > [Your-Bucket-Name] > **accommodation.csv**
Click **Select**.
 - Format of import: **CSV**
 - Database: select `recommendation_spark` from the dropdown list
 - Table: copy and paste: `Accommodation`
3. Click **Import**.

Google Cloud Platform qwiklabs-gcp-01-1d19b3f6a0e4 Search products and services

SQL Import data from Cloud Storage

MASTER INSTANCE

- Overview
- Connections
- Users
- Databases
- Backups
- Replicas
- Operations

Source

Choose the file you'd like to import data from
Browse for a file, or enter the path for one (bucket/folder/file). Make sure you have read access first. [Learn more](#)

☒ qwiklabs-gcp-01-1d19b3f6a0e4/accommodation.csv

Indicate the format of the file you're importing

☐ SQL
A plain text file with a sequence of SQL commands, like the output of mysqldump

☒ CSV
If your Cloud Storage file is a CSV file, select CSV. The CSV file should be a plain text file with one line per row and comma-separated fields.

Destination

Choose the database and table in your Cloud SQL instance that you'd like to import your file into

Database [?](#)
recommendation_spark

Table [?](#)
Accommodation

When you import, a Cloud SQL service account will be granted read access to your Cloud Storage file and the bucket that contains it. This will be reflected in your permissions.

- You will be redirected back to the Overview page. Wait one minute for the data to load.

Import user rating data

- Click **Import** (top menu).
- Specify the following:
 - Source: Click **Browse** > [Your-Bucket-Name] > **rating.csv**
Click **Select**.
 - Format of import: **CSV**
 - Database: select `recommendation_spark` from the dropdown list
 - Table: copy and paste: `Rating`
- Click **Import**.

4. You will be redirected back to the Overview page. Wait one minute for the data to load.

Task 5. Explore Cloud SQL data

1. If you closed your Cloud Shell connection to mySQL, open it again by finding **Connect to this instance** and clicking **Open Cloud Shell**.
2. Press **ENTER** when prompted to log in.
3. Provide your password and press **ENTER**.
4. Query the ratings data:

```
USE recommendation_spark;
```

```
SELECT * FROM Rating
```

```
LIMIT 15;
```

5. Use a SQL aggregation function to count the number of rows in the table.

```
SELECT COUNT(*) AS num_ratings  
FROM Rating;
```

```
(1186 ratings)
```

6. What is the average review rating of accommodations?

```
SELECT  
  COUNT(userId) AS num_ratings,  
  COUNT(DISTINCT userId) AS distinct_user_ratings,  
  MIN(rating) AS worst_rating,  
  MAX(rating) AS best_rating,  
  AVG(rating) AS avg_rating  
FROM Rating;
```

```
(2.46)
```

In machine learning, you will need a rich history of user preferences for the model to learn from. Run the below query to see which users have provided the most ratings.

```
SELECT
  userId,
  COUNT(rating) AS num_ratings
FROM Rating
GROUP BY userId
ORDER BY num_ratings DESC;
```

(the top user leave 75 reviews)



7. Exit the mysql prompt by typing **exit**.

Task 6. Launch Dataproc


You use Dataproc to train the 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

To 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 region of your Cloud SQL instance:

<input type="checkbox"/> Instance ID 	Type	IP address	Instance connection name	High availability	Location
<input type="checkbox"/>  rentals	MySQL 2nd Gen 5.7	35.192.37.112	qwiklabs-gcp-3cab94e41b50482f:us-central1:rentals	Add	us-central1-c

In the snapshot above, the region is **us-central1** and zone is **us-central1-c**.

2. In the Cloud Console, on the **Navigation menu** () , click **Dataproc** and click **Enable API** if prompted.
3. Once enabled, click **Create cluster** and name your cluster **rentals**.
4. Leave the **Region** as it is i.e. **us-central1** and change the **Zone** to **us-central1-c** (in the same zone as your Cloud SQL instance). This will minimize network latency between the cluster and the database.
5. Click on **Configure nodes**.

6. For **Master node**, for **Machine type**, select **n1-standard-2 (2 vCPUs, 7.5 GB memory)**.
7. For **Worker nodes**, for **Machine type**, select **n1-standard-2 (2 vCPUs, 7.5 GB memory)**.
8. Leave all other values with their default and click **Create**. It will take 1-3 minutes to provision your cluster.
9. Note the **Name**, **Zone** and **Total worker nodes** in your cluster.
10. Copy and paste the below bash script into your Cloud Shell (optionally change **CLUSTER**, **ZONE**, **NWORKERS** if necessary before running)

```

echo "Authorizing Cloud Dataproc to connect with Cloud SQL"

CLUSTER=rentals

CLOUDSQL=rentals

ZONE=us-central1-c

NWORKERS=2

machines="$CLUSTER-m"

for w in `seq 0 $((NWORKERS - 1))`; do

    machines="$machines $CLUSTER-w-$w"

done

echo "Machines to authorize: $machines in $ZONE ... finding their IP addresses"

ips=""

for machine in $machines; do

    IP_ADDRESS=$(gcloud compute instances describe $machine --zone=$ZONE --
format='value(networkInterfaces.accessConfigs[].natIP)' | sed "s/[/]/g" | sed "s/\\[/g" )/32

    echo "IP address of $machine is $IP_ADDRESS"

    if [ -z $ips ]; then

        ips=$IP_ADDRESS

    else

        ips="$ips,$IP_ADDRESS"

    fi
done

```

```
fi
```

```
done
```

```
echo "Authorizing [$ips] to access cloudsql=$CLOUDSQL"
```

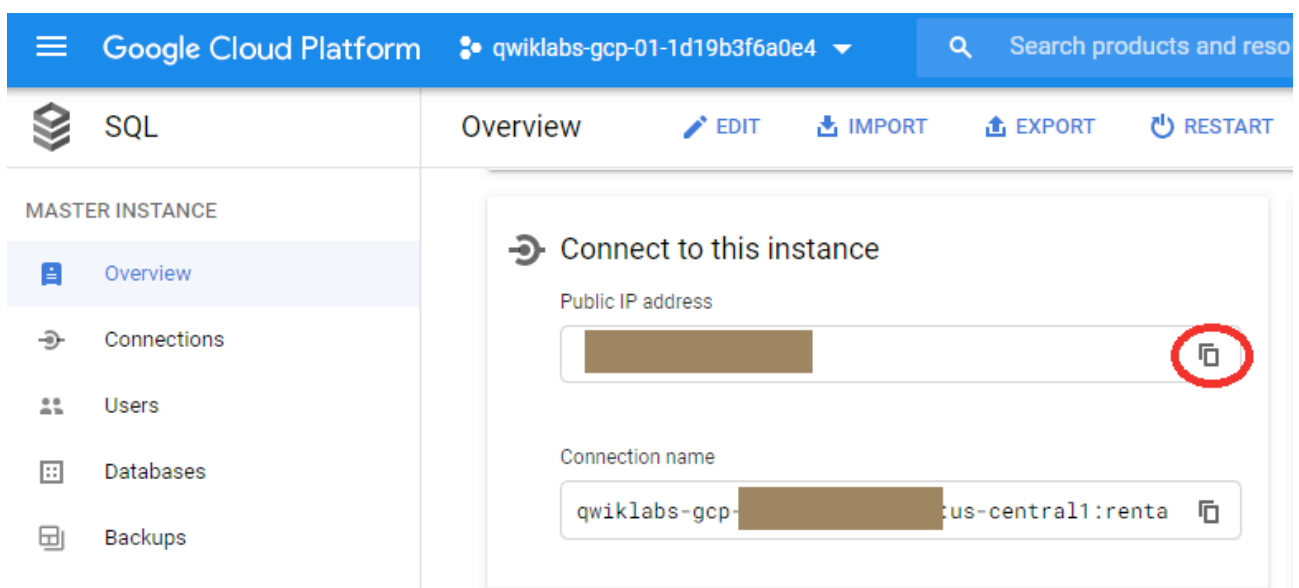
```
gcloud sql instances patch $CLOUDSQL --authorized-networks $ips
```

11. Press **ENTER**. When prompted, type **Y**, then press **ENTER** again to continue.

12. Wait for the patching to complete. You will see the following:

```
Patching Cloud SQL instance...done.
```

13. On the main Cloud SQL page, under **Connect to this instance**, copy your **Public IP Address** to your clipboard. (Alternatively, write it down because you're using it next.)



Task 7. Run the ML model

Next, you create a trained model and apply it to all the users in the system. Your data science team has created a recommendation model using Apache Spark and is written in Python. Copy it over into your staging bucket.

1. Copy over the model code by executing the below commands in Cloud Shell:

```
gsutil cp gs://cloud-training/bdml/v2.0/model/train_and_apply.py train_and_apply.py
cloudshell edit train_and_apply.py
```

2. When prompted, select **Open in New Window**.
3. Wait for the Editor UI to load.
4. Open the `train_and_apply.py` file, find line 30: **CLOUDSQL_INSTANCE_IP**, and paste the Cloud SQL IP address you copied earlier.

```
# MAKE EDITS HERE
CLOUDSQL_INSTANCE_IP = '<paste-your-cloud-sql-ip-here>' #
<---- CHANGE (database server IP)
CLOUDSQL_DB_NAME = 'recommendation_spark' # <--- leave as-is
CLOUDSQL_USER = 'root' # <--- leave as-is
CLOUDSQL_PWD = '<type-your-cloud-sql-password-here>' # <----
CHANGE
```

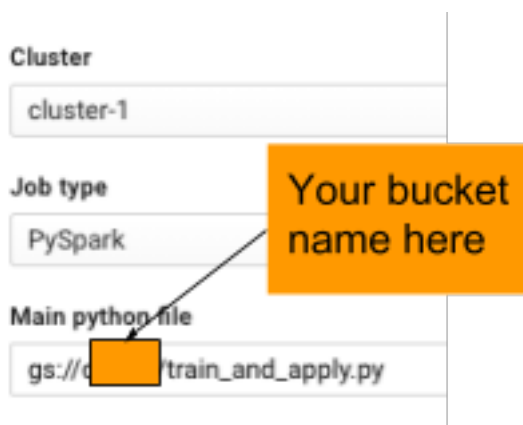
5. Find line 33: **CLOUDSQL_PWD** and type in your Cloud SQL password,

6. The editor will autosave but to be sure, select **File > Save**.
7. From the Cloud Shell ribbon, click on the **Open Terminal** icon and copy this file to your Cloud Storage bucket using this Cloud Shell command:

```
gsutil cp train_and_apply.py gs://$DEVSHHELL_PROJECT_ID
```

Task 8. Run your ML job on Dataproc

1. In the **Dataproc** console, click **rentals** cluster.
2. Click **Submit job**.
3. For **Job type**, select **PySpark** and for **Main python file**, specify the location of the Python file you uploaded to your bucket. Your `<bucket-name>` is likely to be your Project ID, which you can find by clicking on the Project ID dropdown in the top navigation menu.



The screenshot shows the Dataproc console job configuration form. It has three main sections: 'Cluster', 'Job type', and 'Main python file'. The 'Cluster' section has a dropdown menu with 'cluster-1' selected. The 'Job type' section has a dropdown menu with 'PySpark' selected. The 'Main python file' section has a text input field containing 'gs://<bucket-name>/train_and_apply.py'. An orange callout box with the text 'Your bucket name here' points to the '<bucket-name>' placeholder in the 'Main python file' field.

```
gs://<bucket-name>/train_and_apply.py
```

4. For **Max restarts per hour**, enter **1**.
5. Click **Submit**.
6. Select **Navigation menu > Dataproc > Job** tab to see the Job status.

Note: It will take up to 5 minutes for the job to change from **Running** to **Succeeded**. You can continue to the next section on querying the results while the job runs. 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 9. Explore inserted rows with SQL

1. In a new browser tab, open **SQL** (in the Databases section).
2. Click **rentals** to view details related to your Cloud SQL instance.
3. Under **Connect to this instance** section, click **Open Cloud Shell**. This will start a new Cloud Shell tab. In the Cloud Shell tab press **ENTER**.
It will take a few minutes to allow your IP for the incoming connection.
4. When prompted, type the root password you configured, then press **ENTER**.
5. At the mysql prompt, type:

```
USE recommendation_spark;
```

```
SELECT COUNT(*) AS count FROM Recommendation;
```

If you are getting an Empty Set (0) - wait for your Dataproc job to complete. If it's been more than 5 minutes, your job has likely failed and will require troubleshooting.

Tip: You can use the up arrow in Cloud Shell to return your previous command (or query in this case)

(You'll get the number of recommendation provided by the model)

6. Find the recommendations for a user:

```
SELECT
    r.userid,
    r.accoid,
    r.prediction,
    a.title,
    a.location,
    a.price,
    a.rooms,
    a.rating,
    a.type
FROM Recommendation as r
JOIN Accommodation as a
ON r.accoid = a.id
WHERE r.userid = 10;
```

Your result should be similar to the below result:

```
+-----+-----+-----+-----+.
| userid | accoid | prediction | title
|...
+-----+-----+-----+-----+.
| 10     | 41     | 1.7748766 | Big Calm Manor
|...
| 10     | 21     | 1.7174504 | Big Peaceful Cabin
|...
| 10     | 46     | 1.7159091 | Colossal Private Castle
|...
| 10     | 31     | 1.5783813 | Colossal Private Castle
|...
| 10     | 32     | 1.5584077 | Immense Private Hall
|...
+-----+-----+-----+-----+.

```

These are the five accommodations that you would recommend. Note that the quality of the recommendations is not great because the 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.