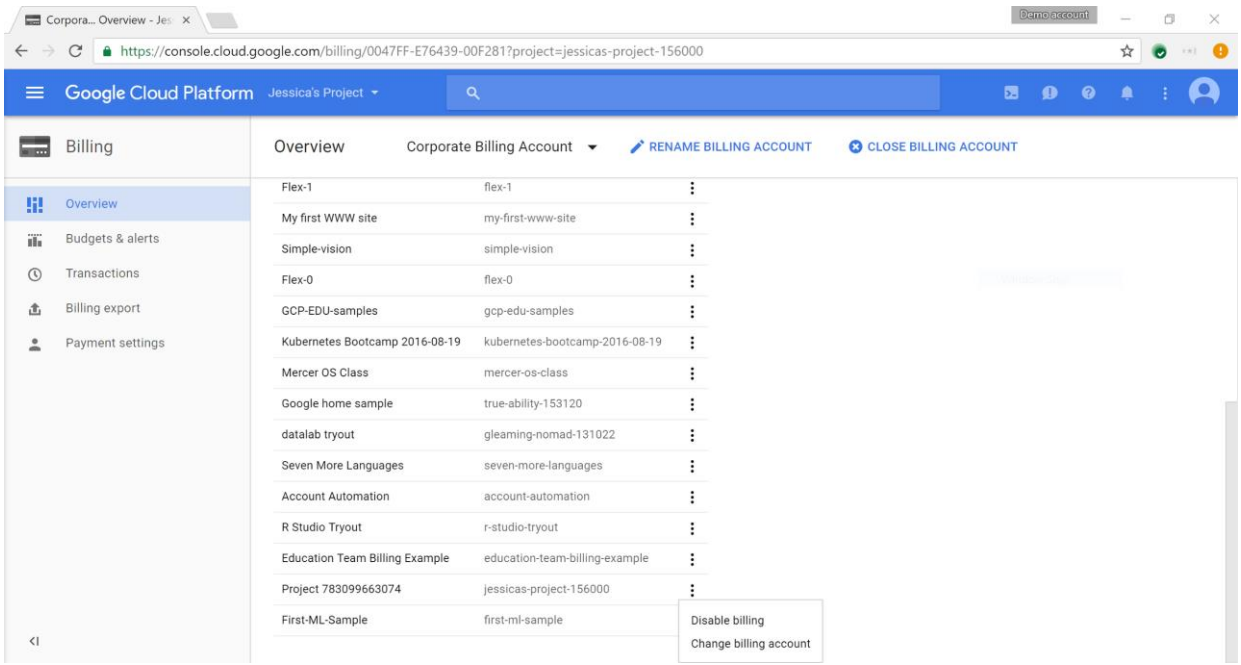
Instructions for Monitoring Google Cloud Platform Billing

There are numerous tools to help you and your students manage your accounts. You can find tools in the “Billing” menu from the “Main” menu.

Changing Billing Accounts

If you want to change billing accounts for a project or disable billing altogether, in the “Billing” section of the left-hand menu, you can adjust the billing account for your projects by selecting the “Dots” icon to the right hand side of the project name:

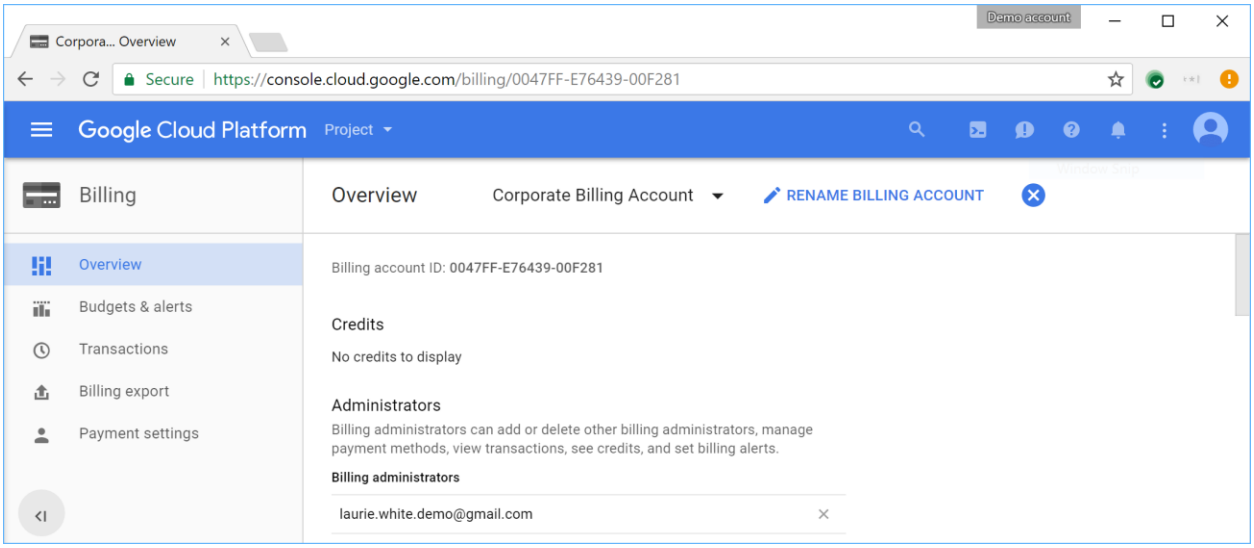


Disabling billing is a quick way to stop a project.

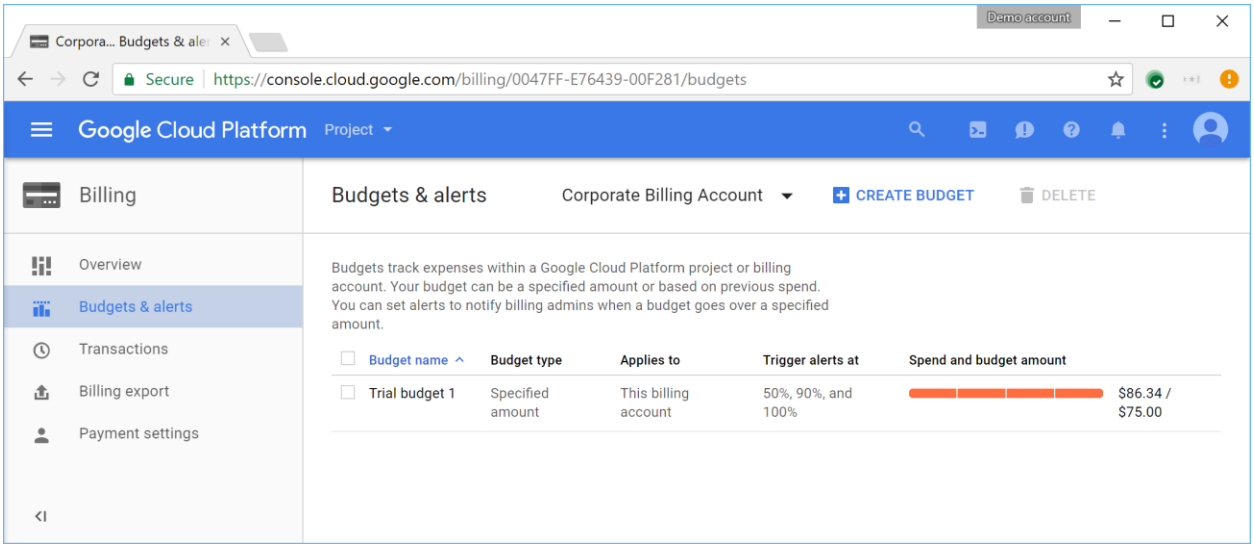
Creating Budgets

You can create budgets to monitor your billing account and individual projects. These budgets will create alerts but **they will not turn off billing for projects.** A project will continue to run, even if it is over budget. You will have to manually go into the billing for the project to disable billing if a project is overrunning. The budget will not update in real time, so it may be a day or two from the time a project begins to overspend until you discover it.

To create a budget, from the “Billing” menu, select “Budgets and Alerts”:

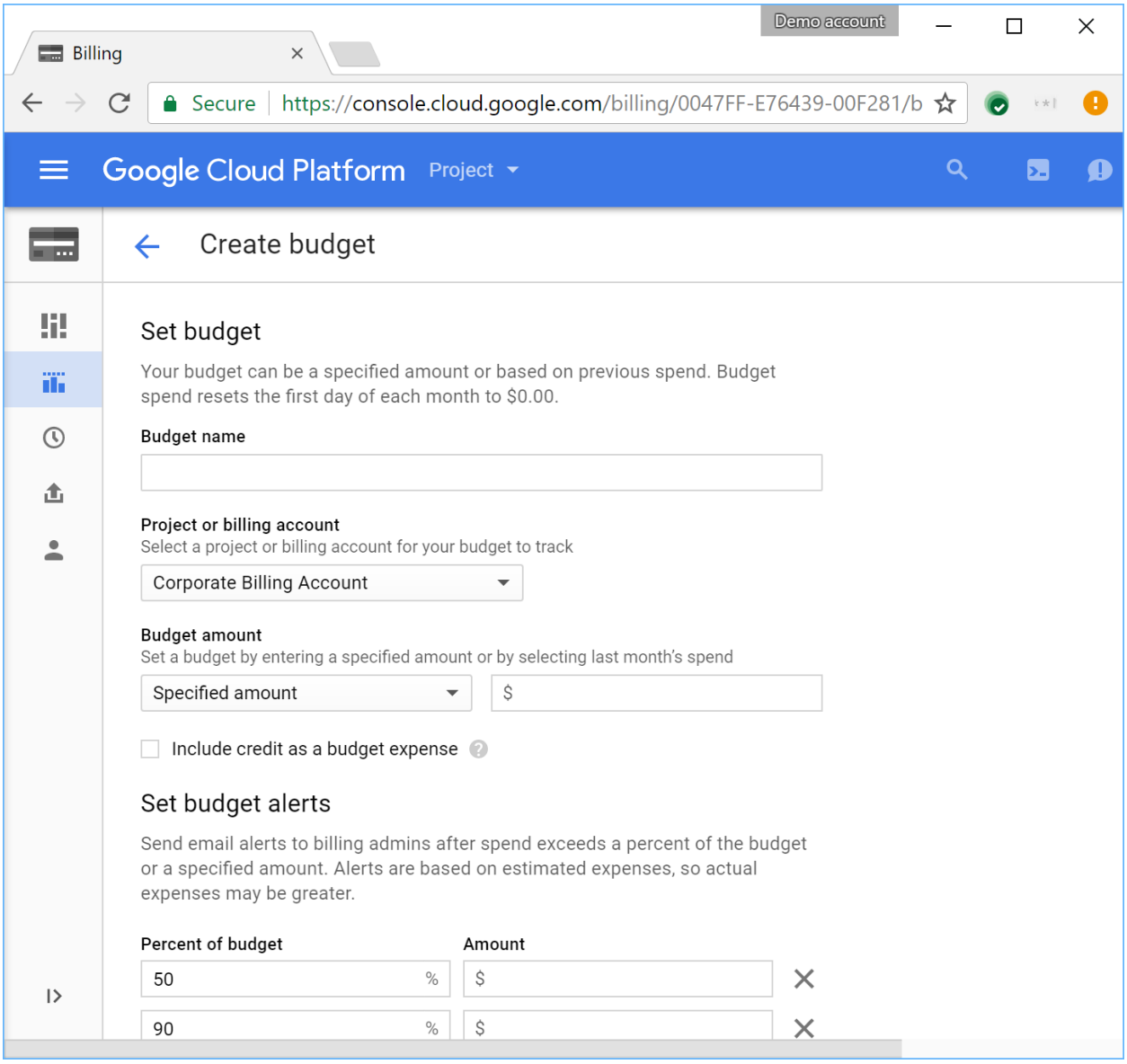


From there, select “Create Budget”:



(Notice that this account is already over budget for the month. Recall, a budget does not actually turn off any services; instead it just alerts the billing administrator when the budget is exceeded.)

Once you've selected "Create Budget," enter the details of the budget you want and how often you receive alerts:



You can select to watch your overall budget or individual projects. "Budgets" work on a calendar month period. You can determine how much you want the budget to be for a given month. You can also customize how often and at what percentage usage you wish to be informed.

Billing Export

If you wish to have a detailed report of all of the services used by your billing account you can set up "Billing Export". Access "Billing Export" from the "Billing" menu in Cloud Console. The details can be stored in either Cloud Storage or BigQuery. If you opt to use Cloud Storage, you can get the data stored in either JSON or CSV format.

Exporting billing data to BigQuery makes it easy to quickly find all projects that are spending over a limit you can set and to see what services you are being charged for. For example, the query below will list all projects that have spent over $0.10 in the current month.

```
SELECT
  project.name,
  cost
FROM
  [your BigQuery table]
WHERE
  cost > 0.1
ORDER BY
   cost DESC
```

At the end of this course, you may close your billing account.

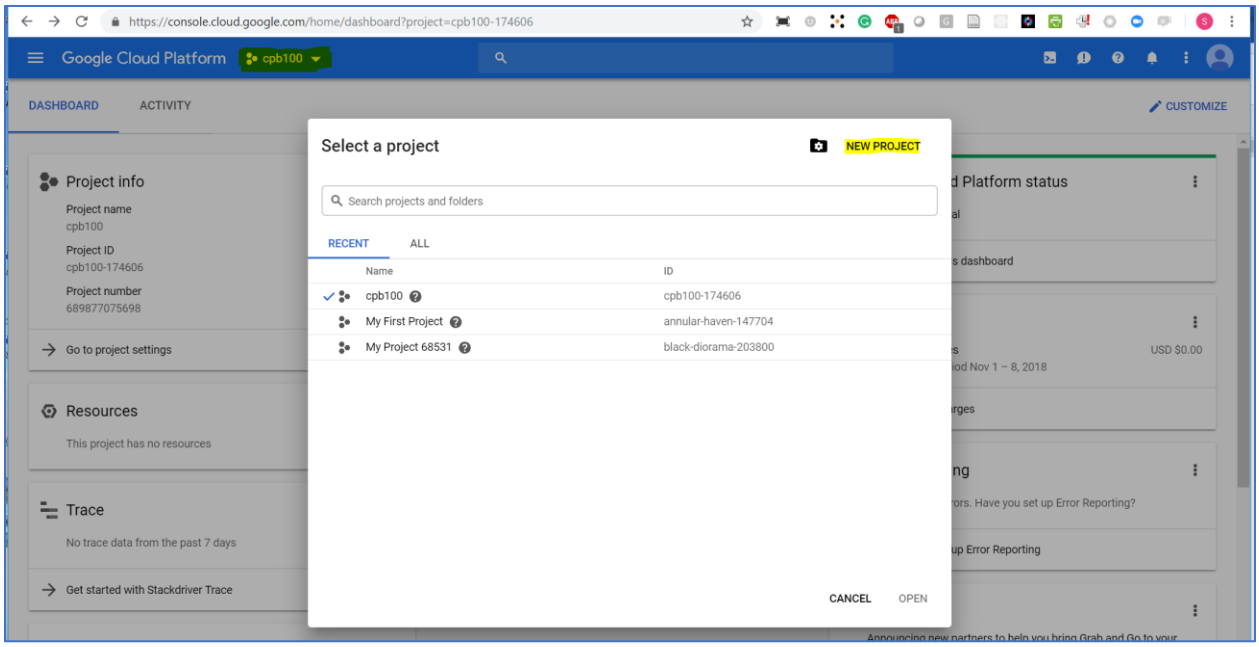
Create a new project

You create your first project using the Google Cloud Platform Console. The project is used to complete all the exercises -

To create a project:

Step 1

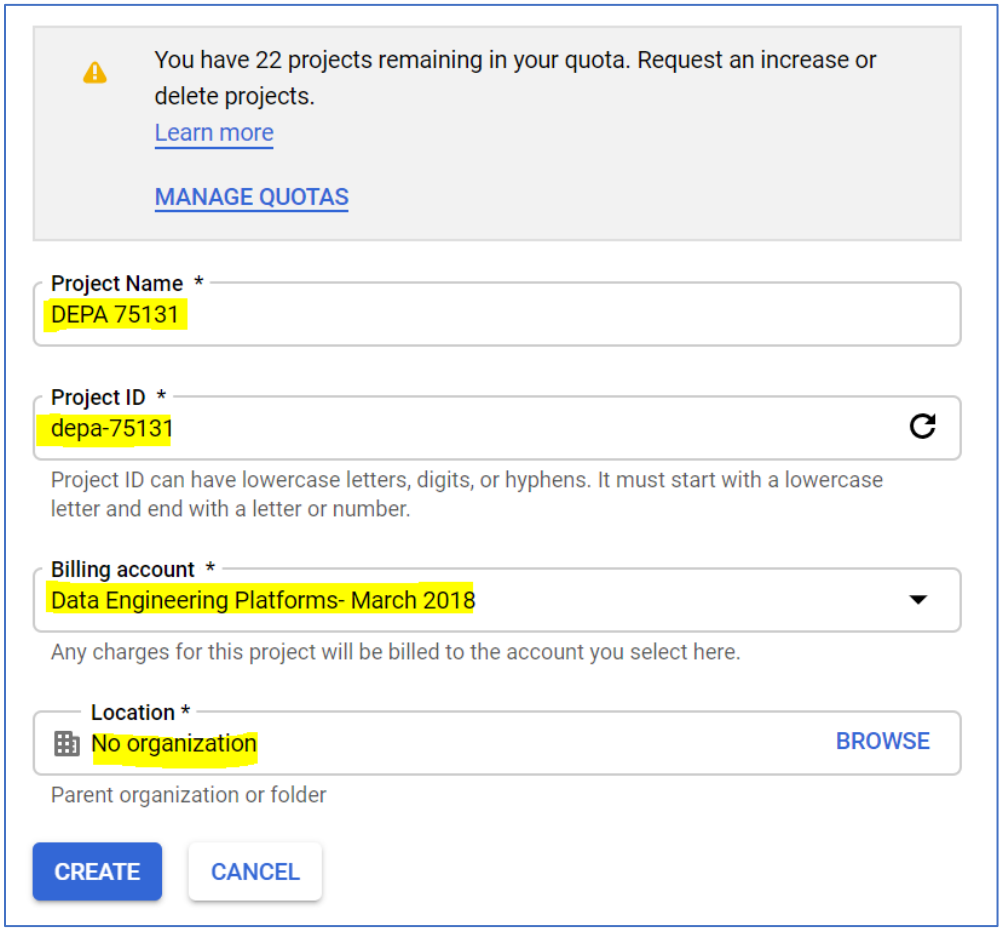
In the Google Cloud Platform Console, click Select a project > Create a new project.



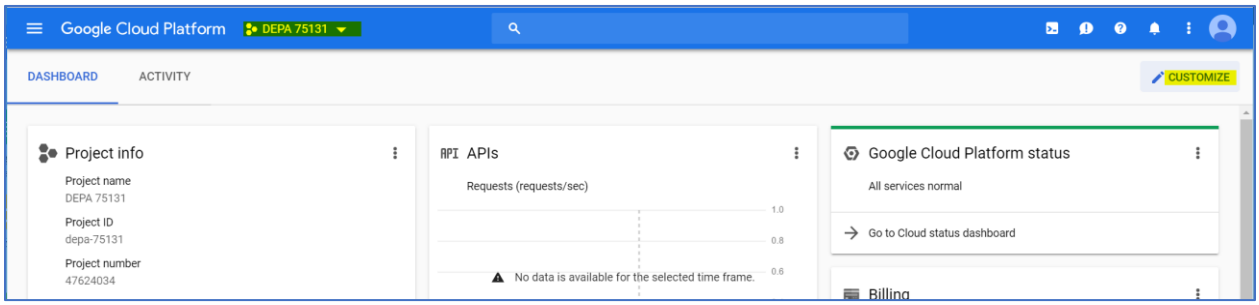
Step 2

In the ‘New Project' dialog:

- For Project name type DEPA.
- Make a note of the Project ID (example: depa-75131) in the text below the project name box; you need it later.



- Click Create. Choose the correct project and play around with customize.

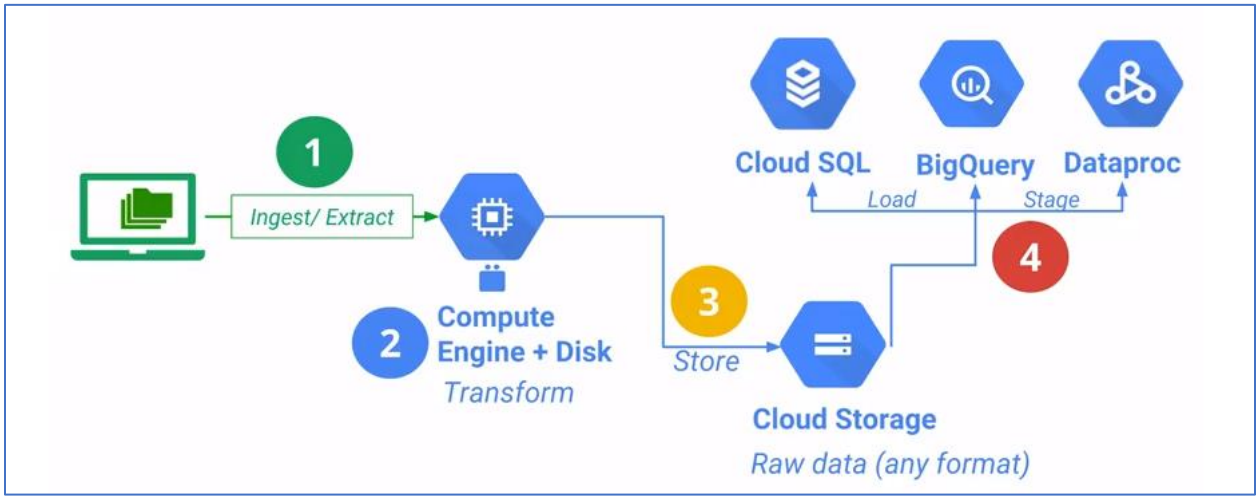


Step 3

In the upper right corner of the console, click Upgrade.

If the Upgrade button does not appear, you may skip this step.

When you upgrade your account, you immediately have access to standard service quotas, which are higher than those available on the free trial. Many of the labs in this course require standard service quotas.



Create a Compute Engine instance:

Step 1

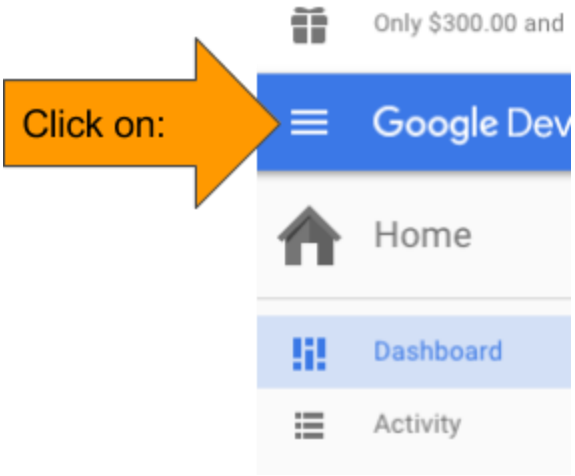
Browse to https://cloud.google.com/

Step 2

Click on **Console**.

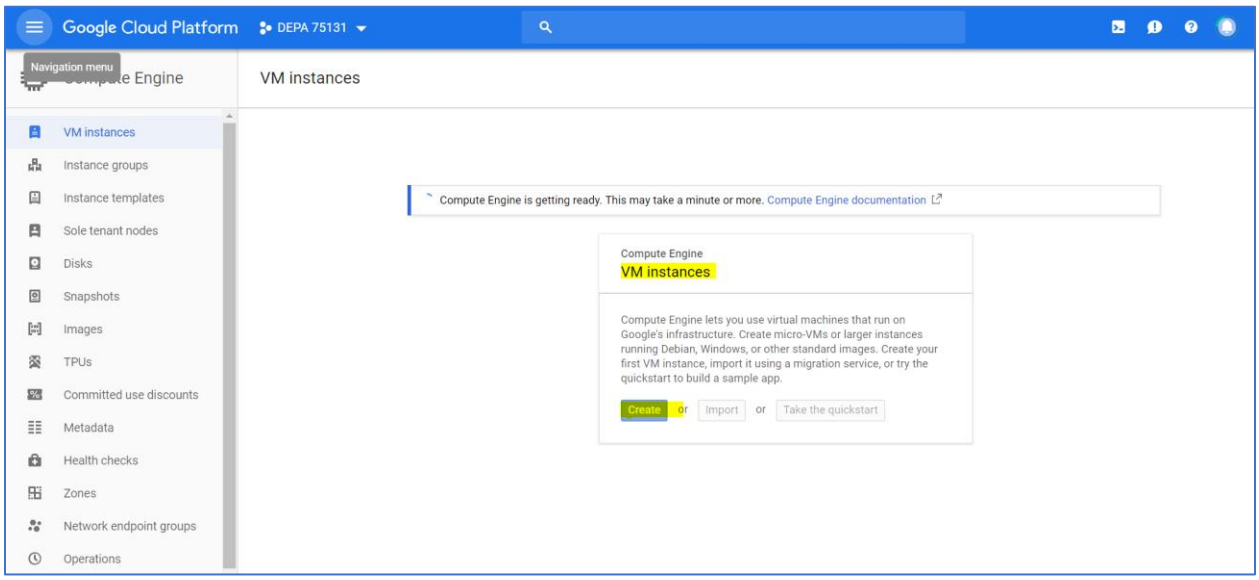
Step 3

Click on the Menu (three horizontal lines):



Step 4

Select **Compute Engine**.

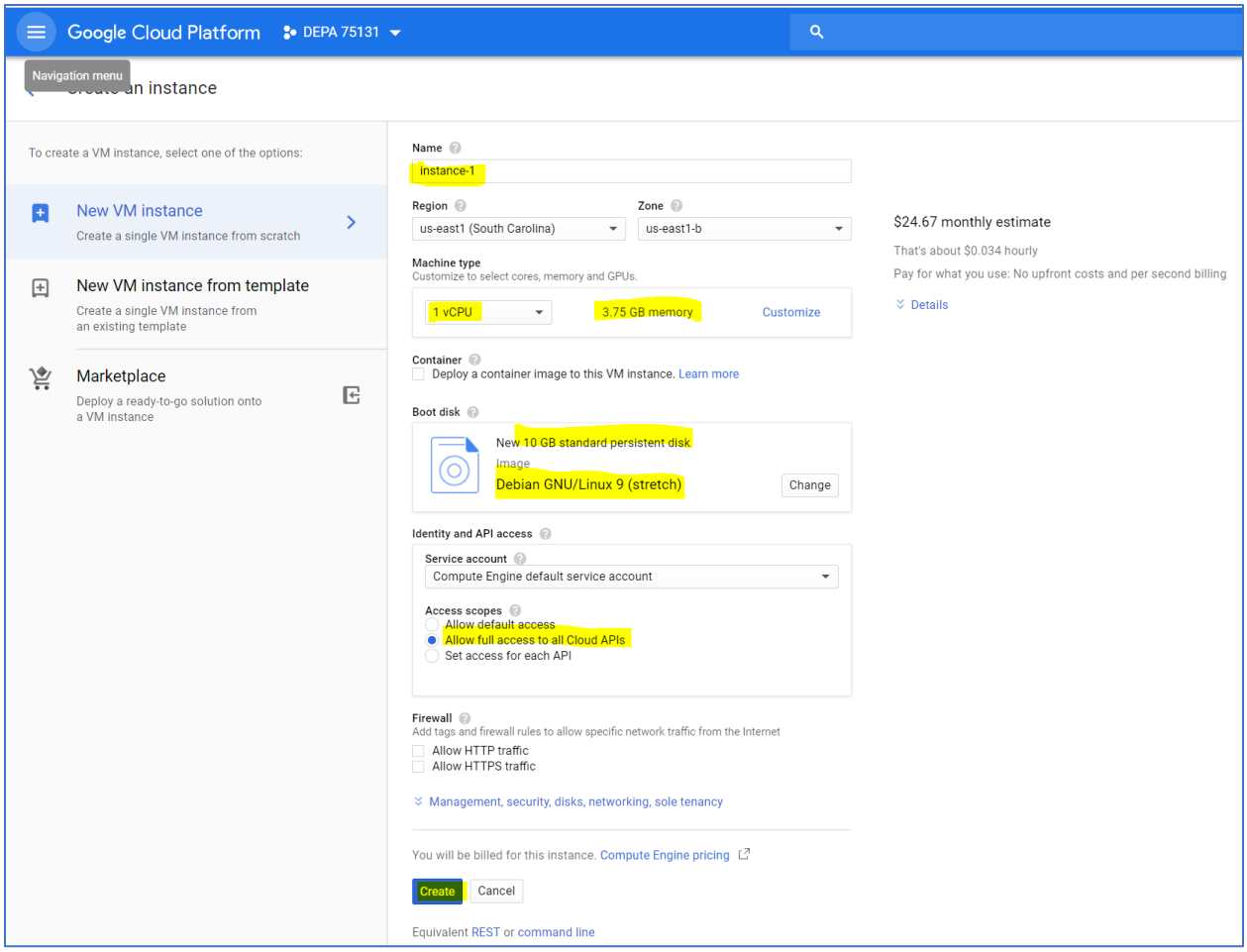


Step 5

Click **Create Instance** and wait for a form to load. You will need to change some options on the form that comes up.

Step 6

Change Identify and API access for the Compute Engine default service account to **Allow full access to all Cloud APIs**:



Step 7

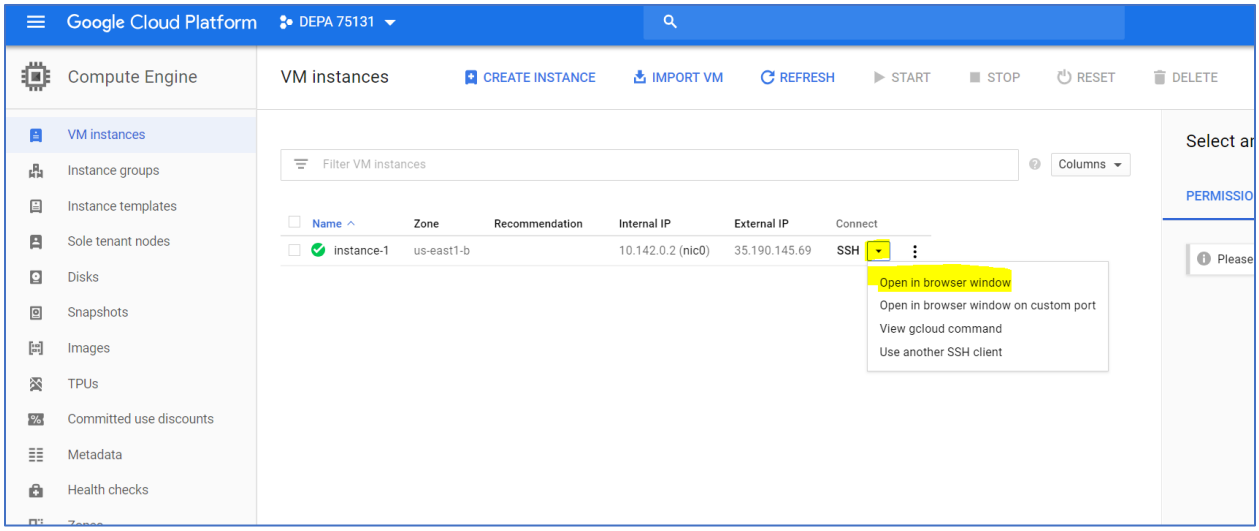
Now, click **Create**

SSH into the instance

You can remotely access your Compute Engine instance using Secure Shell (SSH):

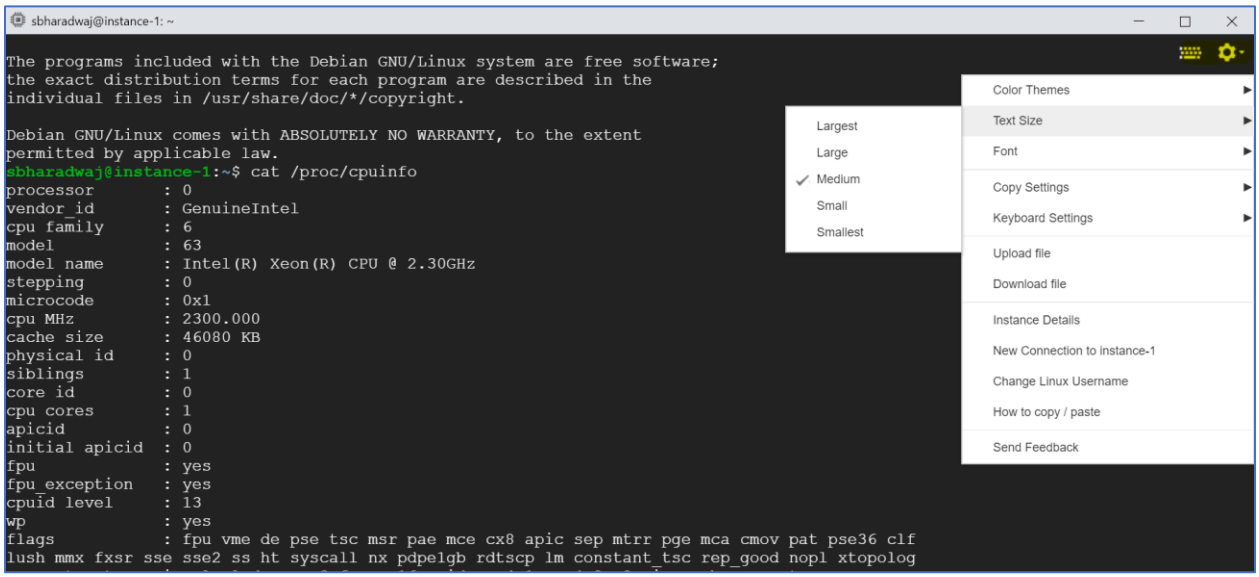
Step 1

Click on **SSH**:



Note

SSH keys are automatically transferred, and that you can ssh directly from the browser, with no extra software needed.



Step 2

To find some information about the Compute Engine instance, type the following into the command-line:

```
cat /proc/cpuinfo
```

5. Install software

Step 1

Type the following into command-line:

```
sudo apt-get update
sudo apt-get -y -qq install git
```

Step 2

Verify that git is now installed

```
git --version
```

---

git is a share point to give a better control

Cloud Storage

Ingest USGS data

To ingest the data into the Compute Engine instance:

Step 1

SSH into your Compute Engine instance

Step 2

On the command-line, type:

```
git clone https://github.com/GoogleCloudPlatform/training-data-analyst
```

This downloads the code from github.

Note

If you get a git authorization error, it is likely that the github URL has a typo in it. Please copy and paste the above code.

Step 3

Navigate to the folder corresponding to this lab:

```
cd training-data-analyst/CPB100/lab2b
```

Step 4

Examine the ingest code using less:

```
less ingest.sh
```

The less command allows you to view the file (Press the **spacebar** to scroll down; the letter **b** to **b**ack up a page; the letter **q** to **q**uit).

The program ingest.sh downloads a dataset of earthquakes in the past 7 days from the US Geological Survey.

Step 5

Run the ingest code:

```
bash ingest.sh
```

Step 6

Verify that some data has been downloaded:

```
head earthquakes.csv
```

The head command shows you the first few lines of the file.

Transform the data

You will use a Python program to transform the raw data into a map of earthquake activity:

Step 1

The transformation code is explained in detail in this notebook:

https://github.com/GoogleCloudPlatform/datalab-samples/blob/master/basemap/earthquakes.ipynb

Feel free to read the narrative to understand what the transformation code does. The notebook itself was written in Datalab, a GCP product. { https://cloud.google.com/datalab/docs/quickstart }

Step 2

over-write

First, install the necessary Python packages on the Compute Engine instance:

```
bash install_missing.sh
```

Step 3

Then, run the transformation code:

```
python3 transform.py
```

Step 4

You will notice a new image file if you list the contents of the directory:

```
ls -l
```

Create bucket

Create a bucket using the GCP console:

Step 1

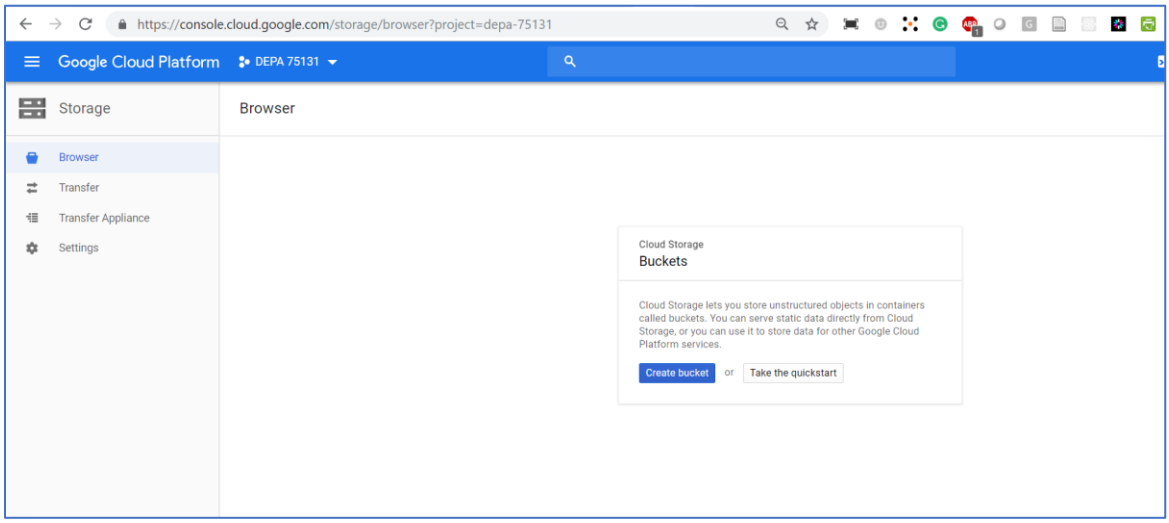
Browse to the GCP Console by visiting http://cloud.google.com

Step 2

Click on the Menu (3 bars) at the top-left and select **Storage**

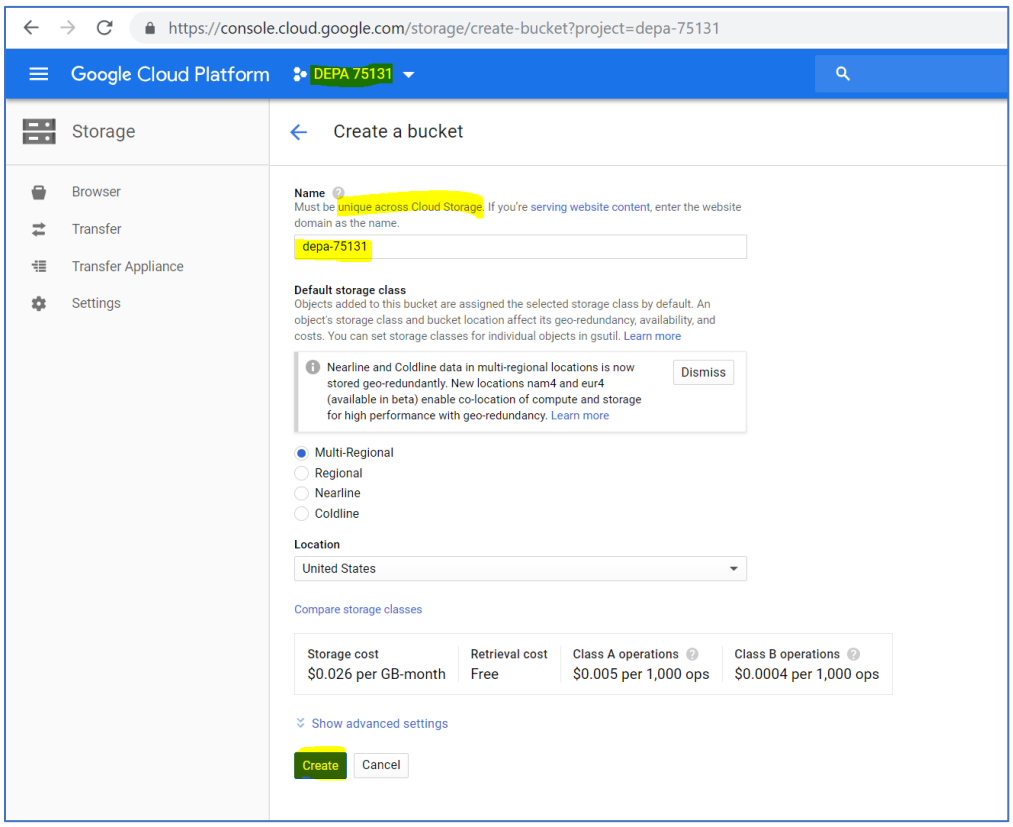
Step 3

Click on **Create Bucket**



Step 4

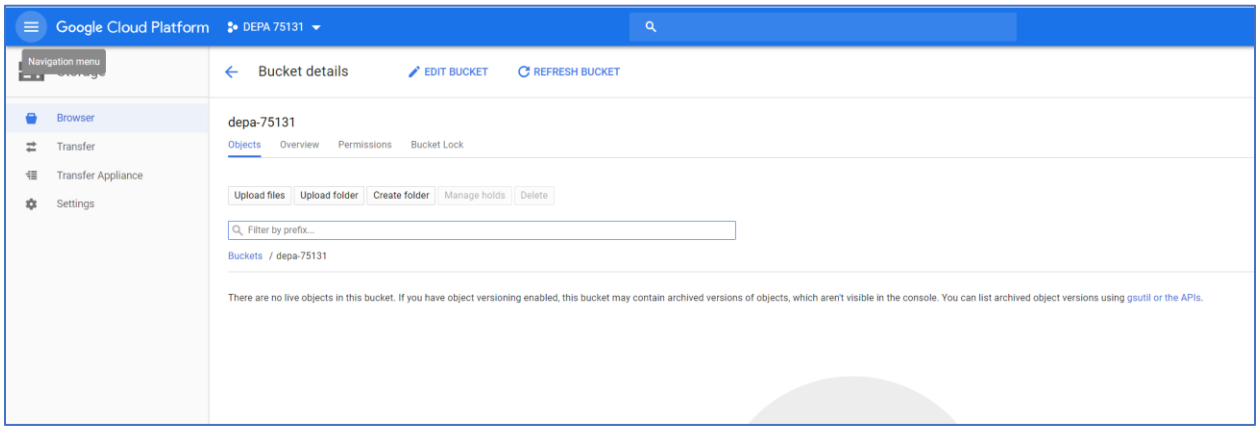
Choose a globally unique bucket name (your project name is unique, so you could use that). Then, click **Create**. (Example: depa-75131)



Step 5

Note down the name of your bucket: depa-75131

In this and future labs, you will insert this whenever the directions ask for <YOUR-BUCKET>.



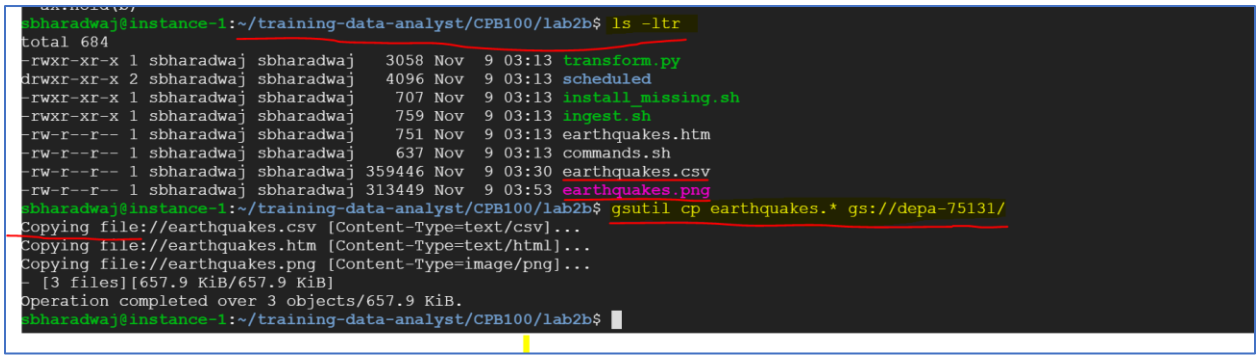
Store data

To store the original and transformed data in Cloud Storage

Step 1

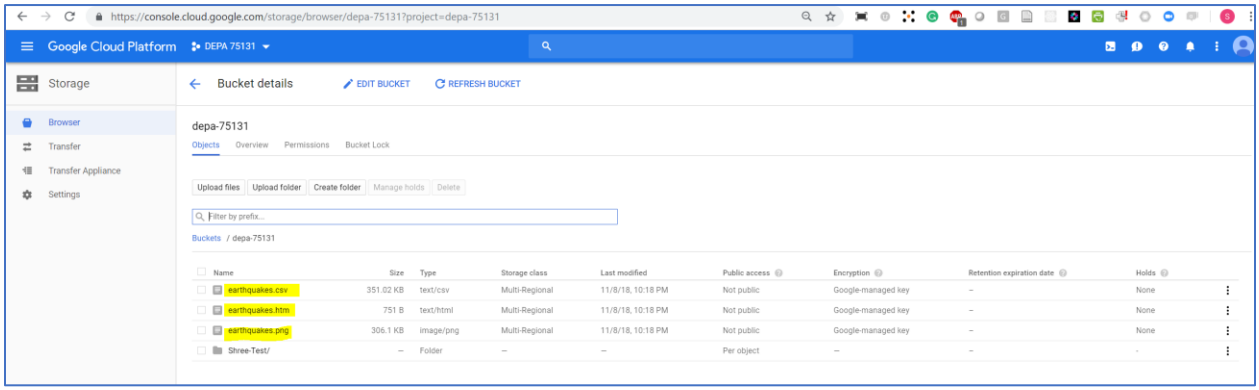
In the SSH window of the Compute Engine instance, type the below to copy files to cloud storage

```
gsutil cp earthquakes.* gs://<YOUR-BUCKET>/
```



Step 2

On the GCP console, click on your bucket name, and notice there are three new files present. Play around with the create folder and uploading files option as well.



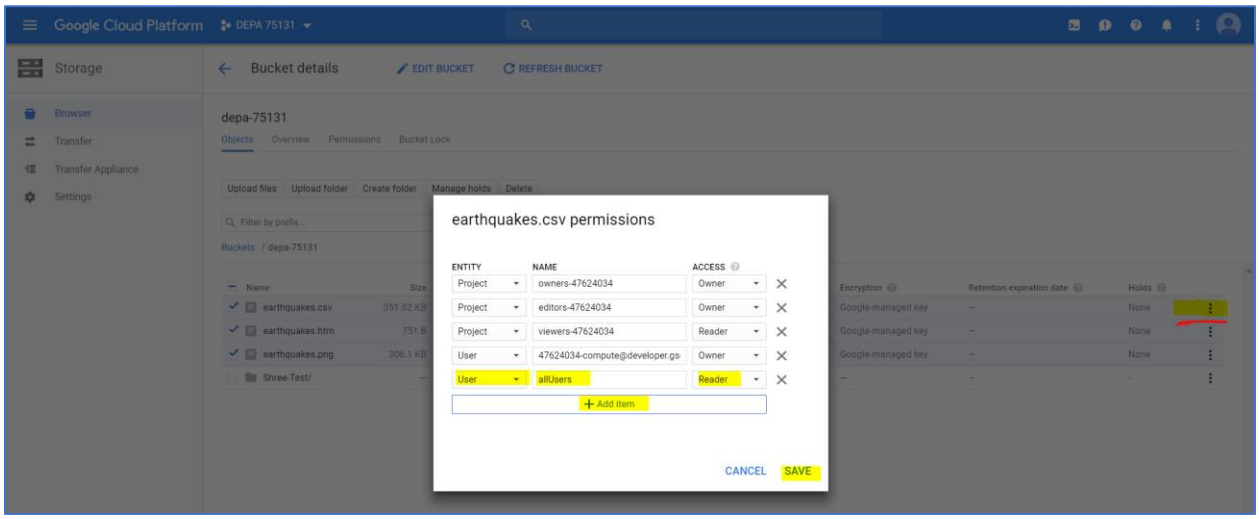
Publish Cloud Storage files to web

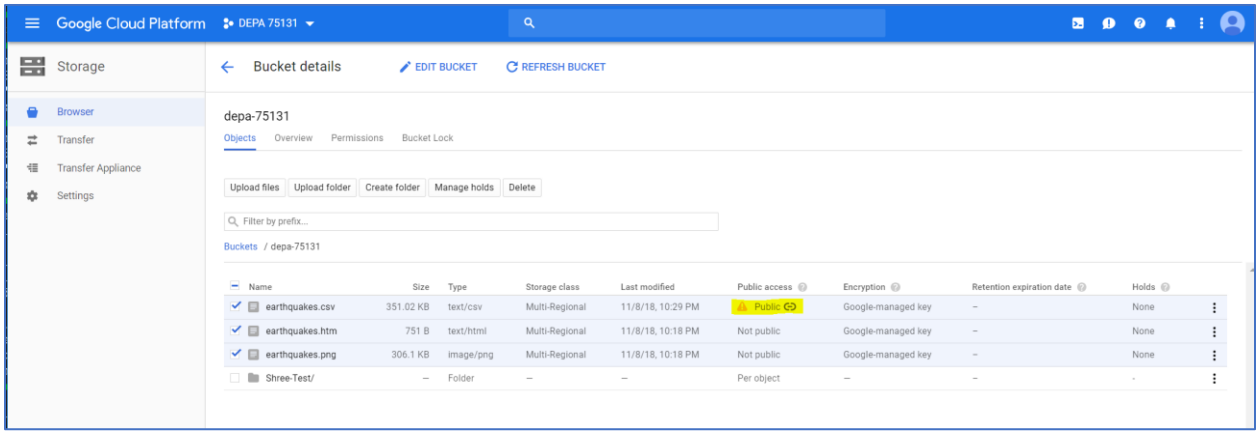
To publish Cloud Storage files to the web:

Step 1

On the GCP console, select *all three* earthquakes files that you uploaded to the bucket and edit permissions to publish cloud storage files to the web (https://cloud.google.com/storage/docs/access-control/making-data-public)

Step 2





Clean up

To delete the Compute Engine instance (since we won't need it any more):

Step 1

On the GCP console, click the Menu (three horizontal bars) and select **Compute Engine**

Step 2

Click on the checkbox corresponding to the instance that you created (the default name was instance-1)

Step 3

Click on the **Delete** button in the top-right corner. Deleting the instance has no impact on the files that you stored on Cloud Storage

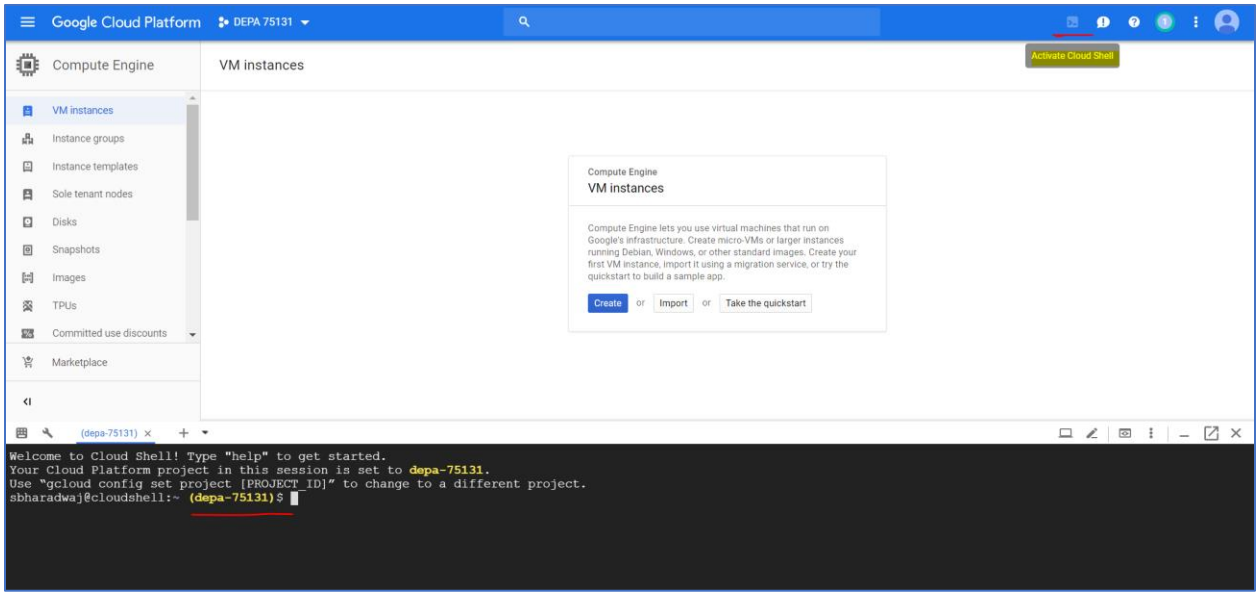
---

Explore data in Cloud Shell

To explore the labs code in Cloud Shell:

Step 1

Start Cloud Shell if you have not already done so. On GCP console, click on the **Activate Google Cloud Shell** button: (https://cloud.google.com/shell/)



Step 2

On the command-line, type:

```
git clone https://github.com/GoogleCloudPlatform/training-data-analyst
```

This downloads the code from github

Step 3

Navigate to the folder corresponding to this lab:

```
cd training-data-analyst/CPB100/lab3a
```

Step 4

Examine the table creation file using less:

```
less cloudsql/table_creation.sql
```

The less command allows you to view the file (Press the **spacebar** to scroll down; the letter **b** to **b**ack up a page; the letter **q** to **q**uit).

Step 5

Examine the data files using head:

```
head cloudsql/*.csv
```

The head command shows you the first few lines of each file.

Stage .sql and .csv files into Cloud Storage

Stage the table definition and data files into Cloud Storage, so that you can later import them into Cloud SQL:

Step 1

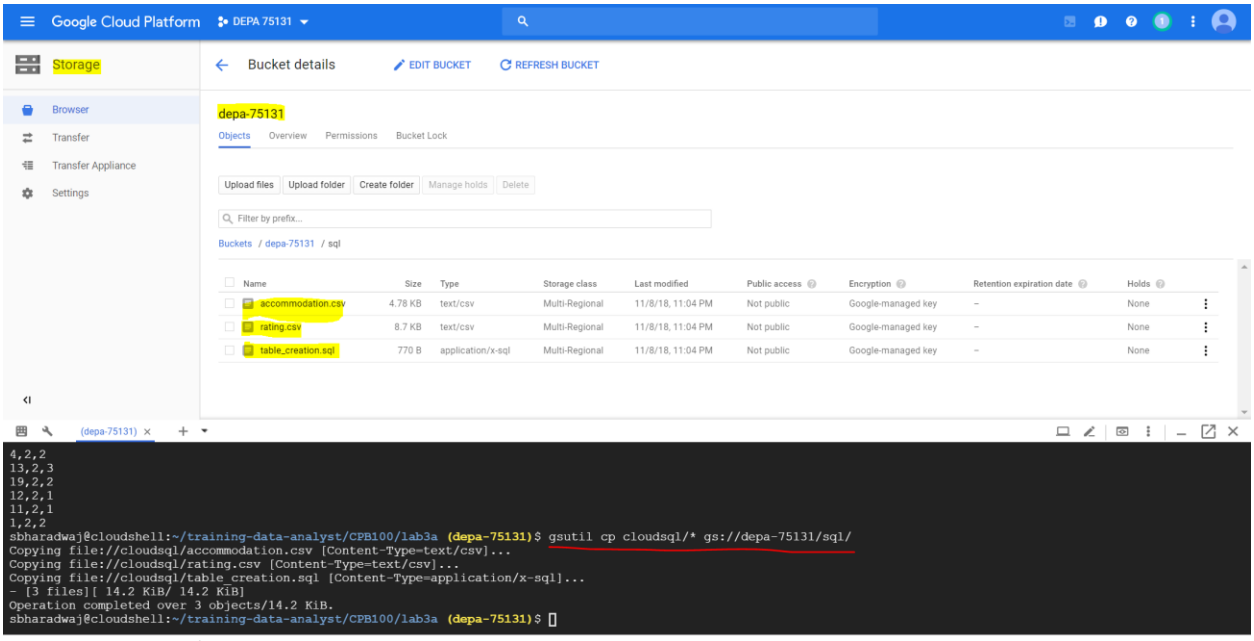
From Cloud Shell within the lab3a directory, type:

```
gsutil cp cloudsql/* gs://<BUCKET-NAME>/sql/
```

substituting the name of the bucket that you created in Lab 2b.

Step 2

From the GCP console, go to Storage, navigate to your bucket and verify that the .sql and .csv files now exist on Cloud Storage.

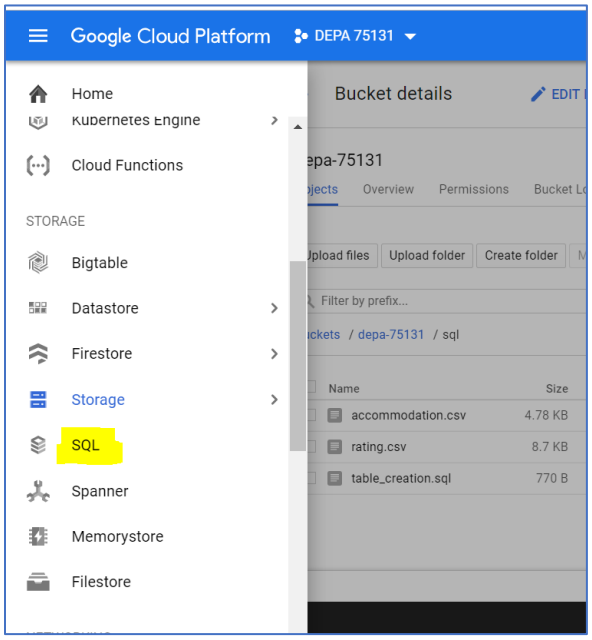


Create Cloud SQL instance

To create Cloud SQL instance:

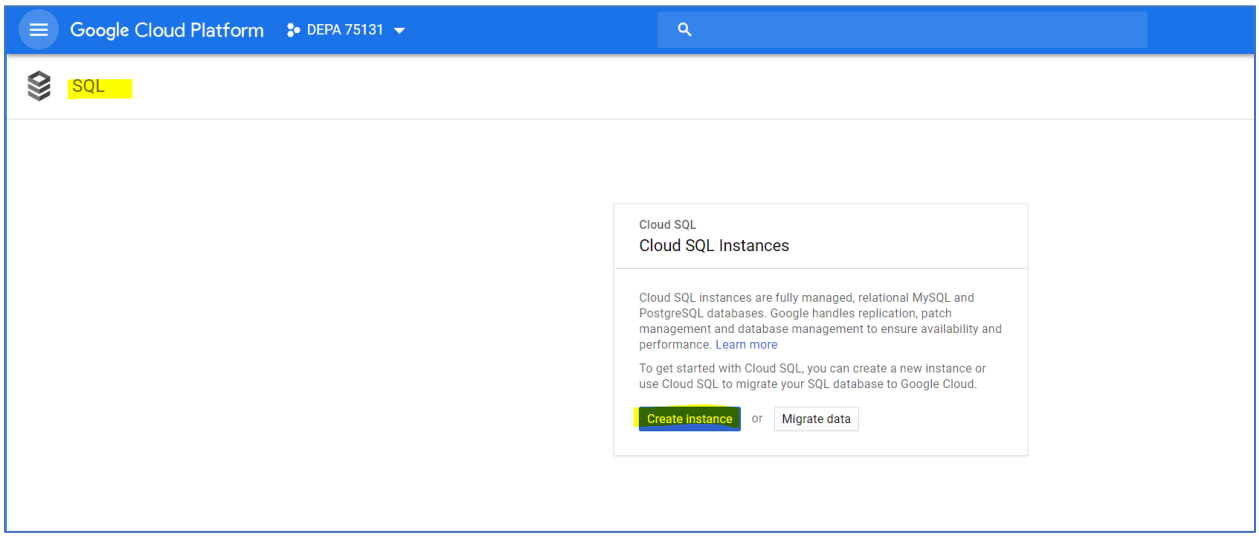
Step 1

From the GCP console main menu (horizontal bars), select **SQL** (in the Storage section).



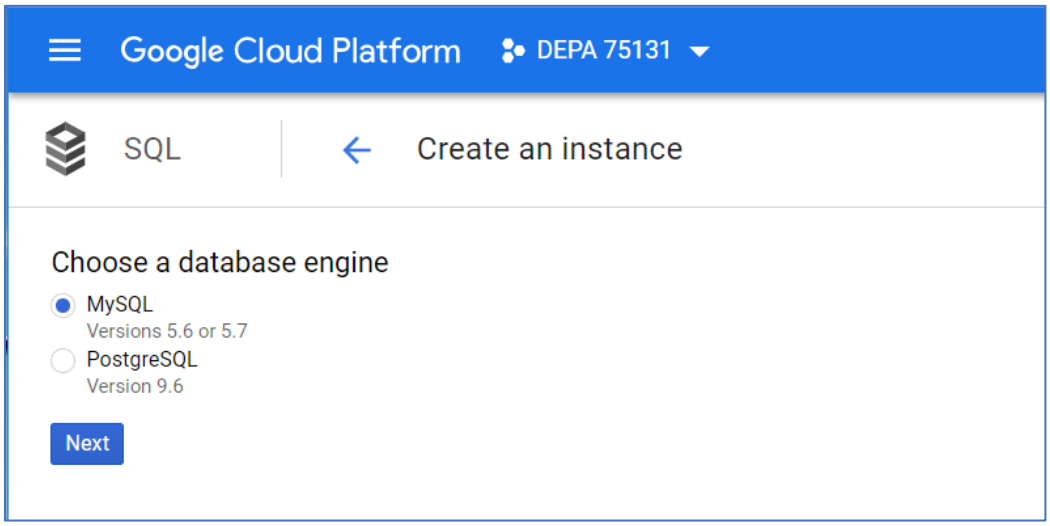
Step 2

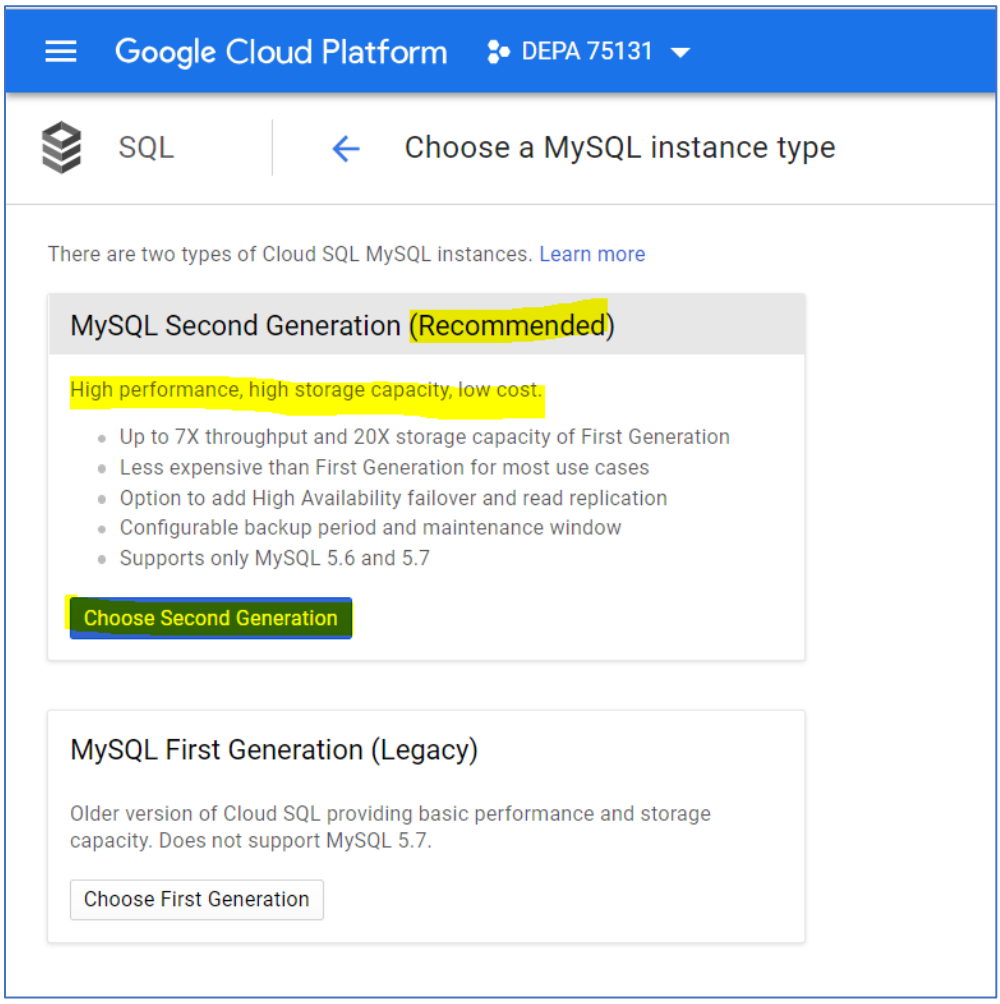
Click on **Create Instance**



Step 3

Choose **MySQL**, then click Next. On the next dialog, choose Cloud SQL **Second Generation**



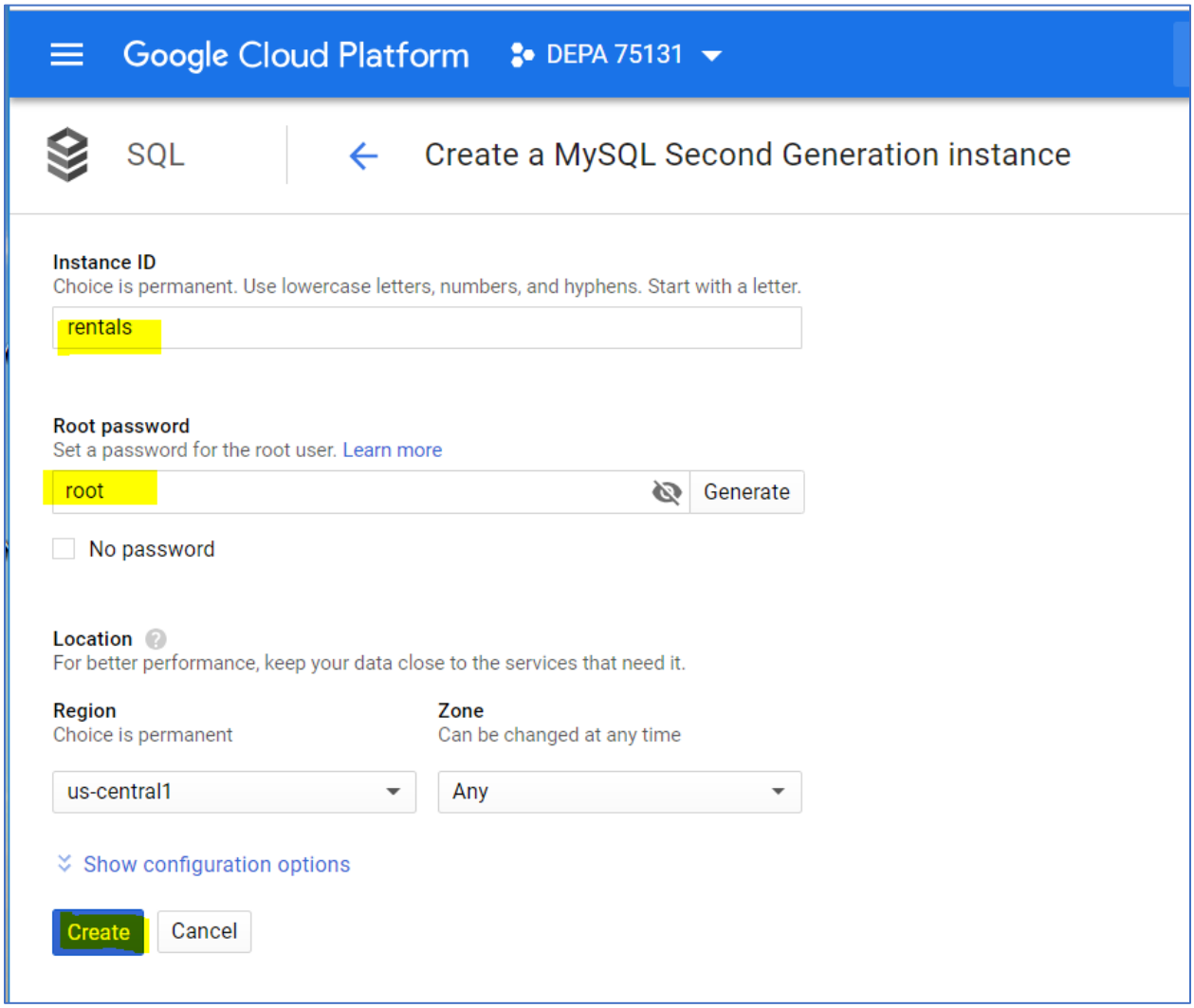


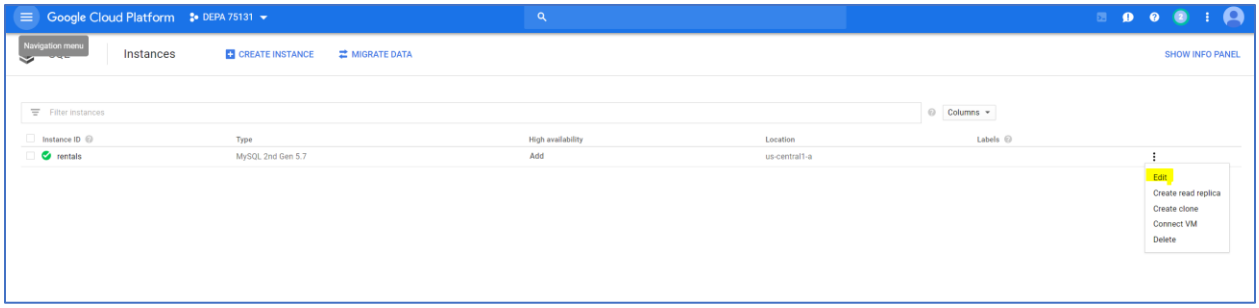
Step 4

Give the instance the ID **rentals**

Step 5

Scroll down and specify a root password. Before you forget, note down the root password (please don't do this in real-life!) in a table that looks like this.

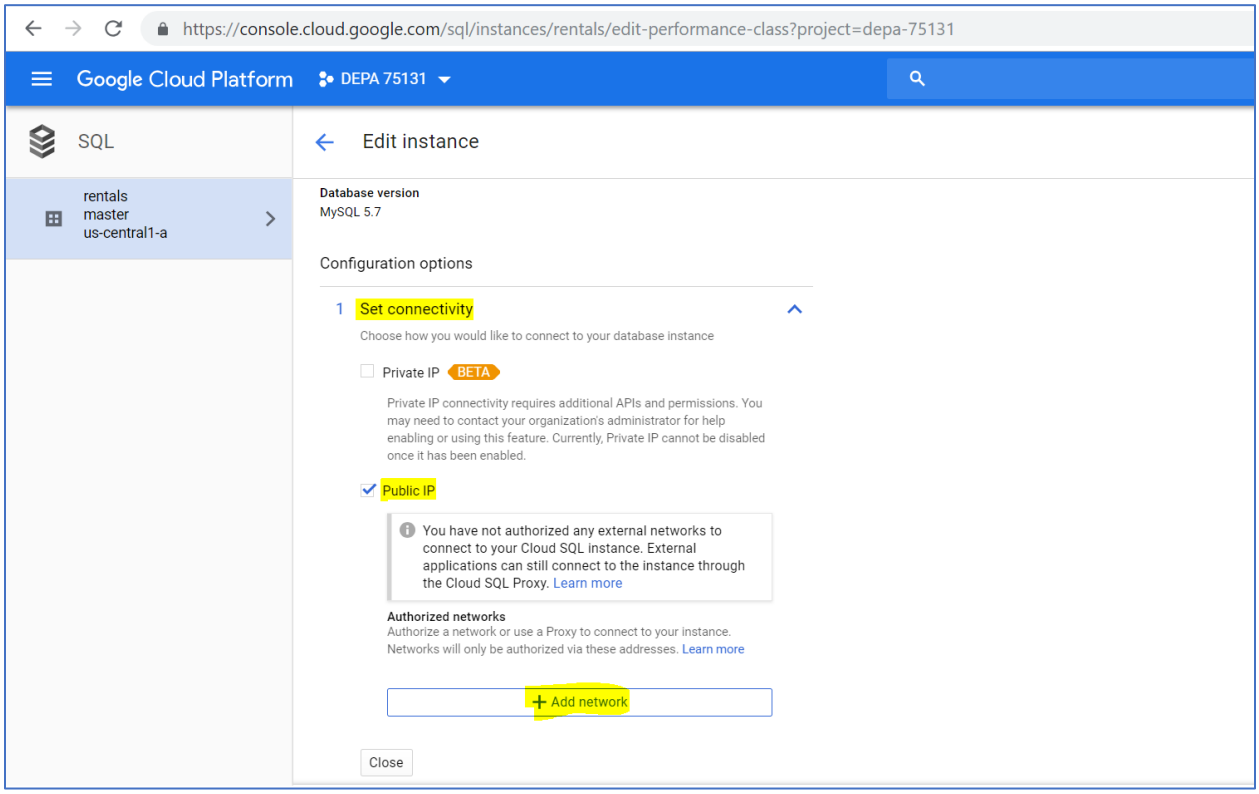




Machine	Password/IP Address
Root password	_____?

Step 6

Scroll down (Configuration Options > Set Connectivity > Authorized Networks) and click on the **Add network** button



Step 7

From Cloud Shell within the lab3a directory, find your IP address by typing:

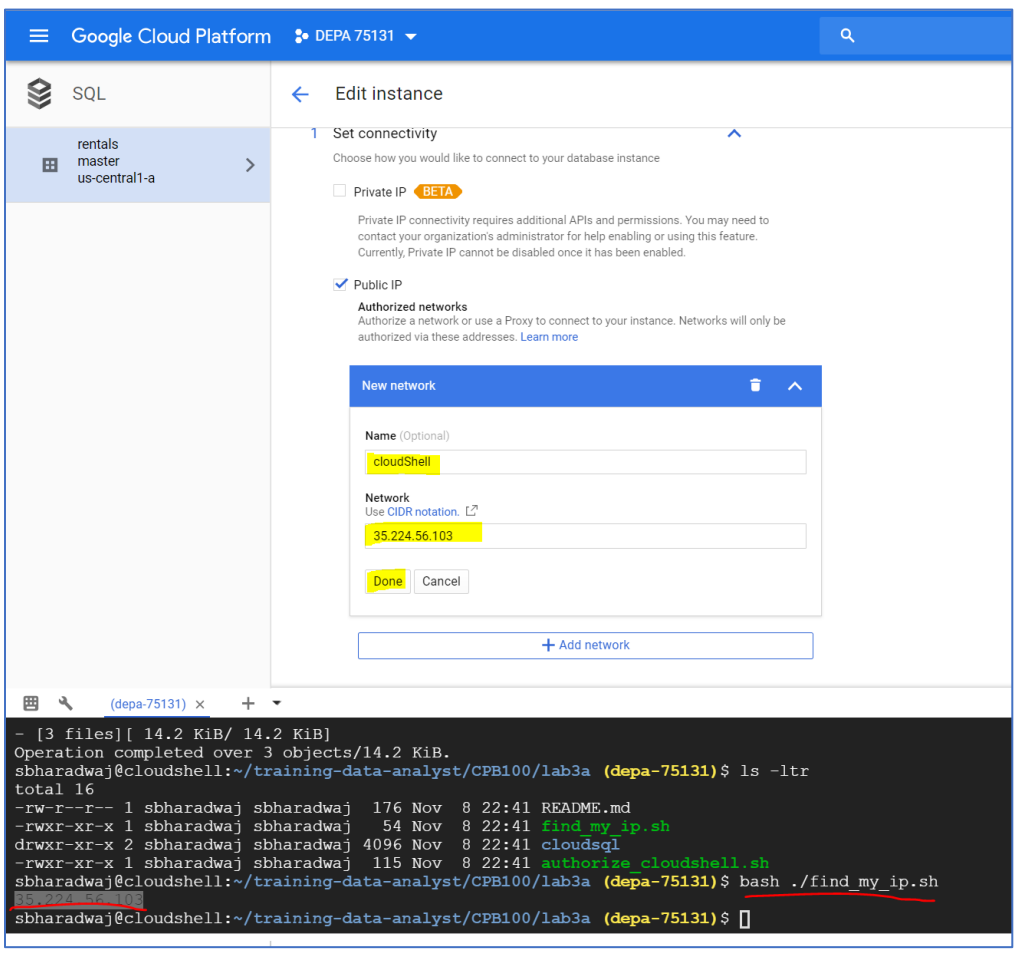
```
bash ./find_my_ip.sh
```

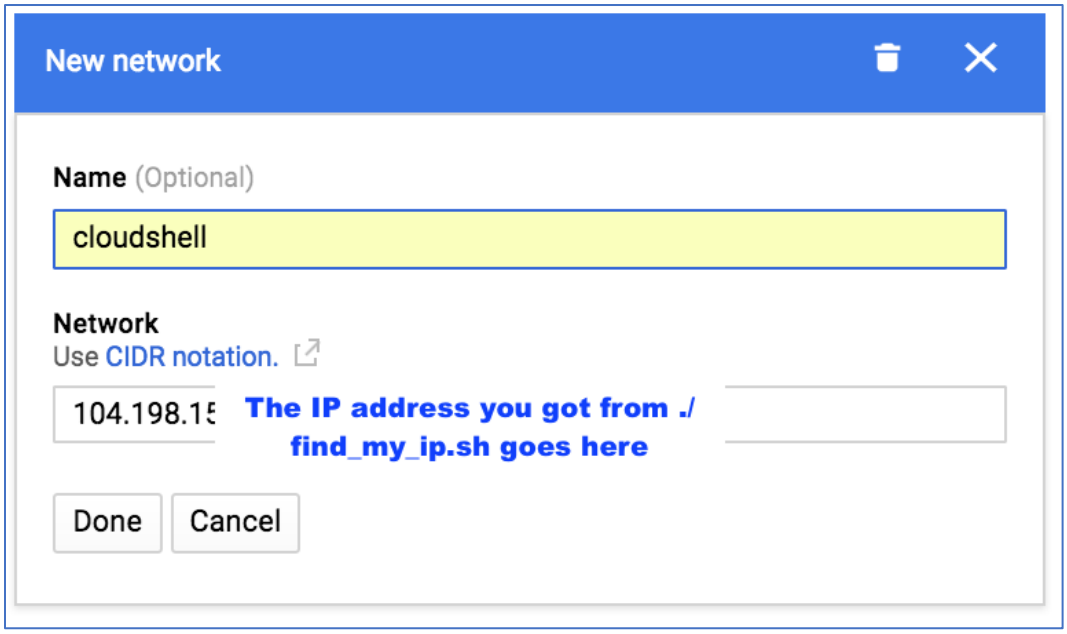
Note this down in the second row of a table like this:

Machine	Password/IP Address
Root password	(Step 5)
Cloud Shell (client)	_____ ?
Cloud SQL (MySQL server)	(Step 10)

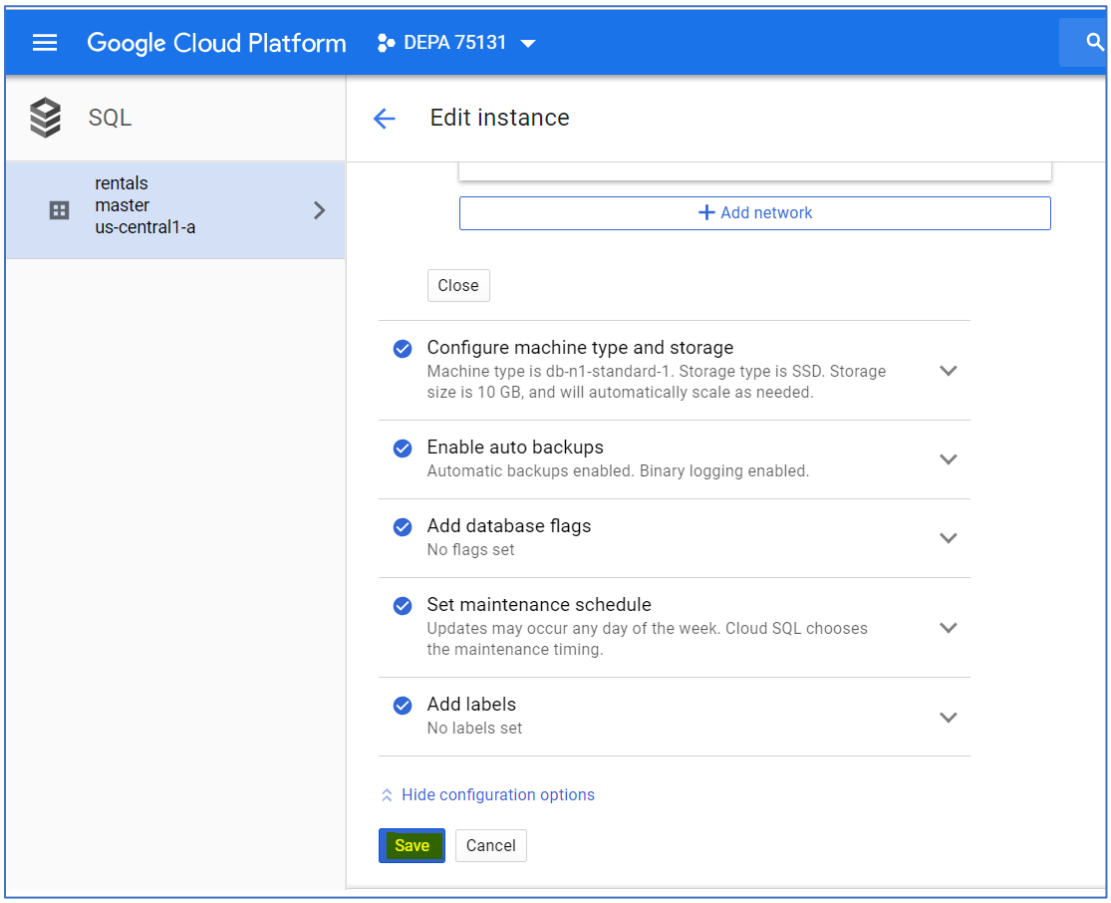
Step 8

In the Add Network box, plug your IP address (Name can be anything) and click **Done**





Note: If you lose your Cloud Shell VM due to inactivity, you will have to reauthorize your new Cloud Shell VM with Cloud SQL. For your convenience, lab3a includes a script called **authorize_cloudshell.sh** that you can run.

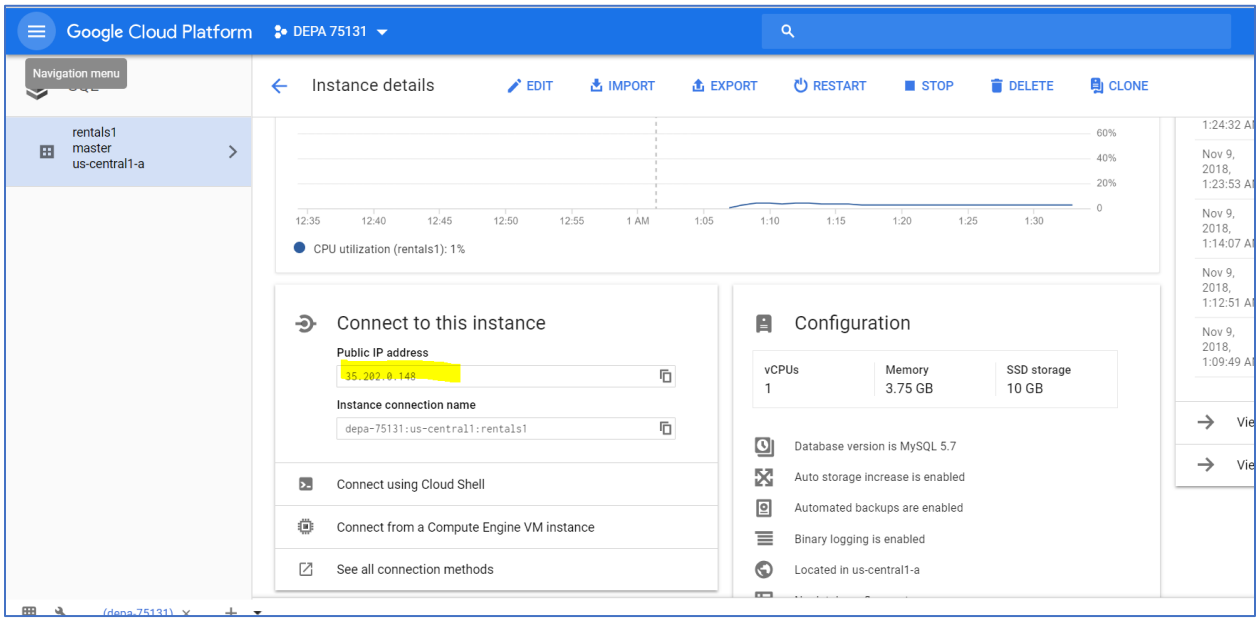


Step 9

Click **Create** to create the instance. It will take a minute or so for your Cloud SQL instance to be provisioned...

Step 10

Note down the IP address of your Cloud SQL instance (from the browser window) in the *third* row of the table you started (Overview > connect to this instance):



Note this down in the second row of a table like this:

Machine	Password/IP Address
Root password	(Step 5)
Cloud Shell (client)	(Step 7)
Cloud SQL (MySQL server)	_____ ?

Create tables

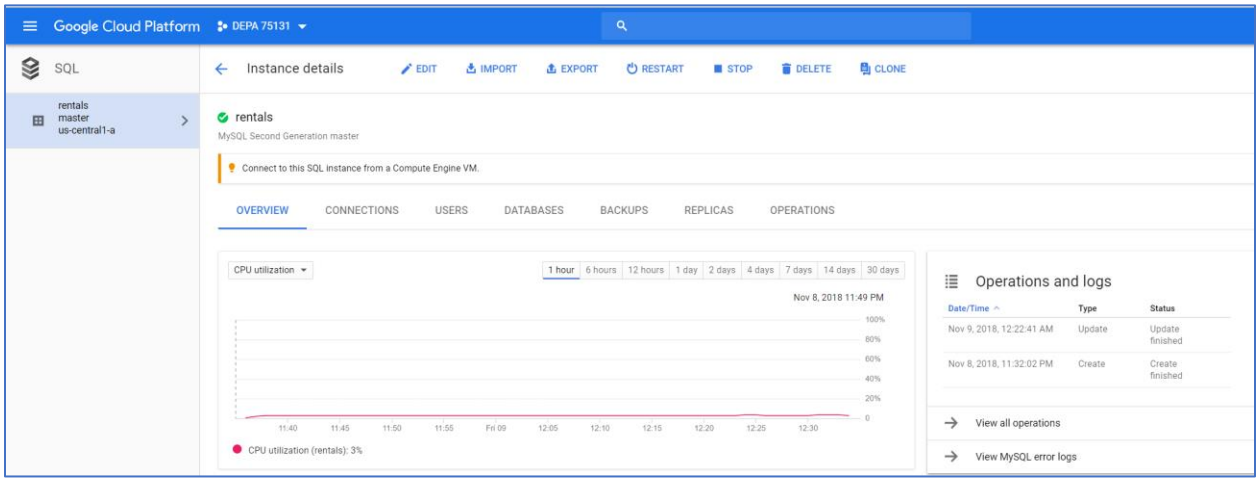
To import table definitions from Cloud Storage:

Step 1

Click on the hyperlink named **rentals** i.e. your Cloud SQL instance name

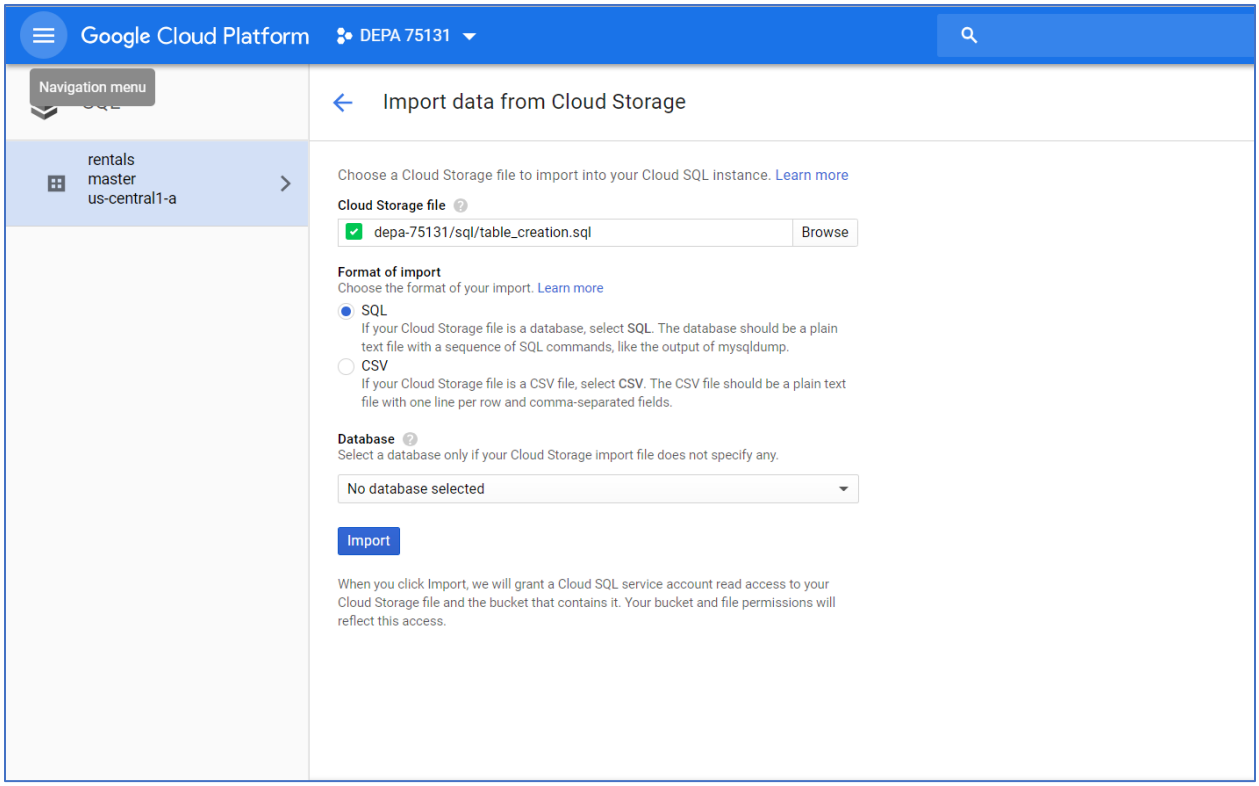
Step 2

Click on **Import** (on the top menu bar)



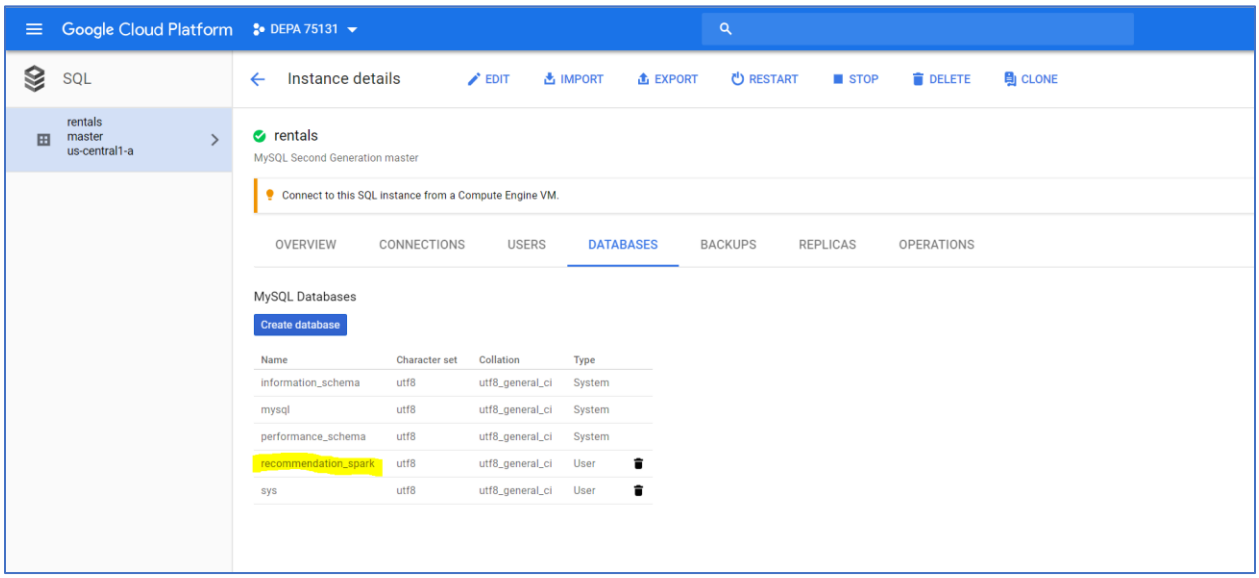
Step 3

Click on the **Browse** button and browse to table_creation.sql (DDL) and **Select** it



Step 4

Click **Import**.



Populate tables

To import CSV (data) files from Cloud Storage:

Step 1

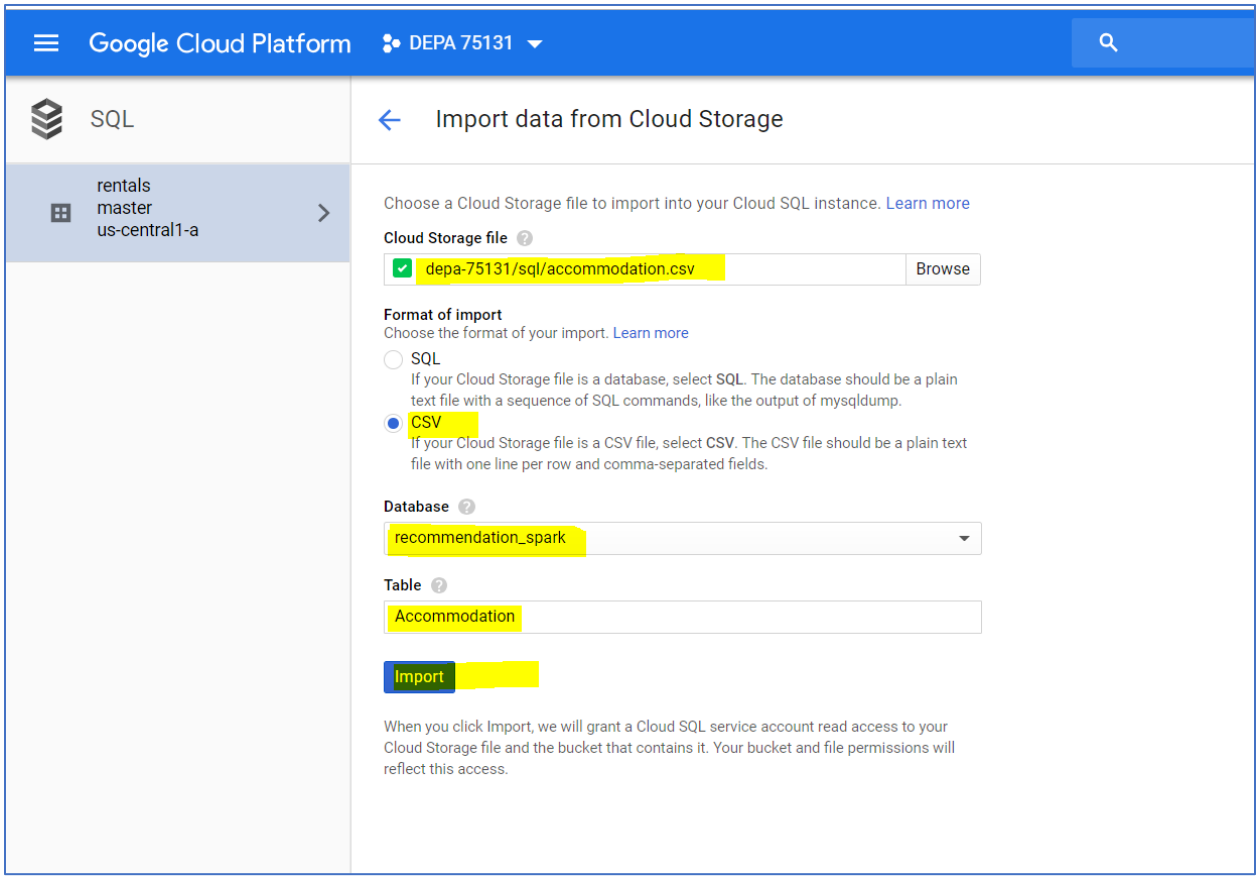
From the GCP console page with the Cloud SQL instance details, click on **Import** (top menu)

Step 2

Click on the **Browse** button, browse to accommodation.csv and **Select** it. Fill out the rest of the dialog as follows:

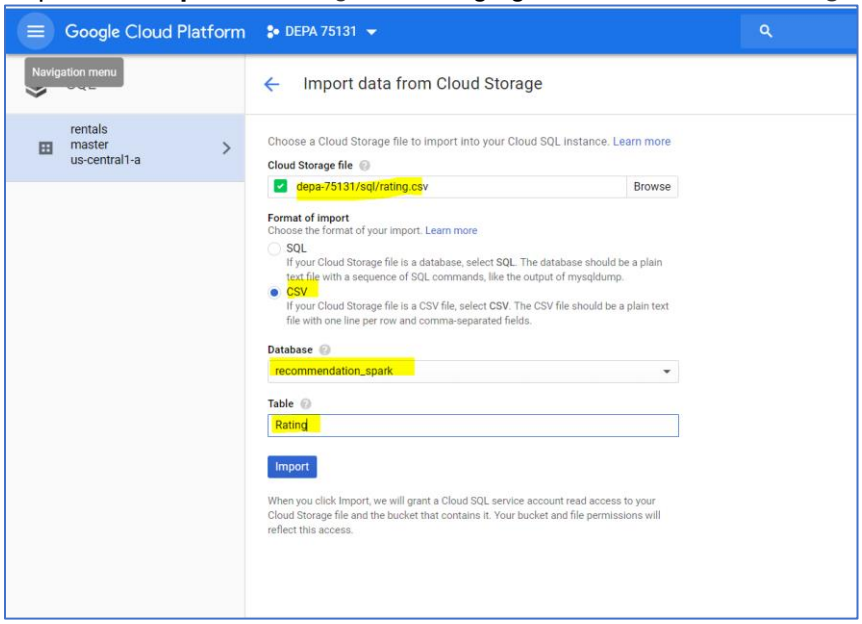
The Database is recommendation_spark

The Table is **Accommodation**



Step 3

Repeat the **Import** for rating.csv, changing the table name to **Rating**.



Explore Cloud SQL

To explore Cloud SQL, you can use the tool mysql from Cloud Shell:

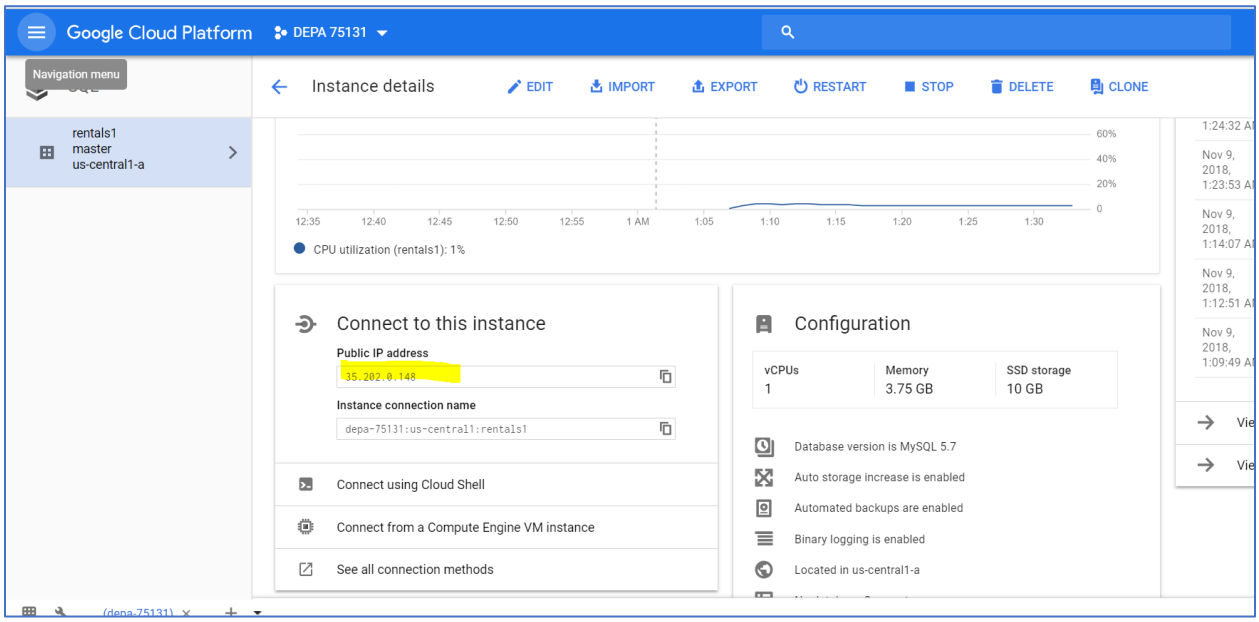
Step 1

In the Cloud Shell window, type:

```
mysql --host=<MySQLIP> --user=root --password
```

The IP address is the one for the database server (i.e. the third row in the notes). You can also find it from the instance details on the cloud console.

MySQL will prompt you for the root password. Type that into the prompt when prompted.



Step 2

In the Cloud Shell, at the mysql prompt, type:

```
use recommendation_spark;
```

This sets the database in the mysql session.

Step 3

View the list of tables you created. This will be helpful to prevent any typos in your query in step 4.

```
show tables;
```

Step 4

Let's see if there is a great deal out there somewhere:

```
select * from Accommodation where type = 'castle' and price < 1500;
```

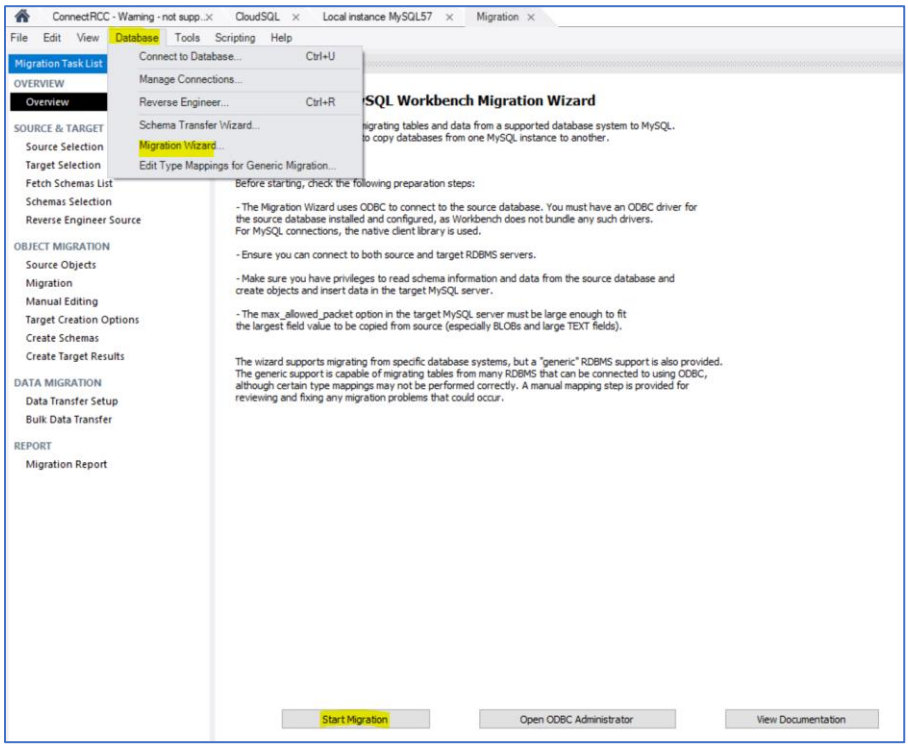
All the cheap castles are rated poorly. Oh well.

Migrate Data

To migrate data from local instance to Cloud SQL.

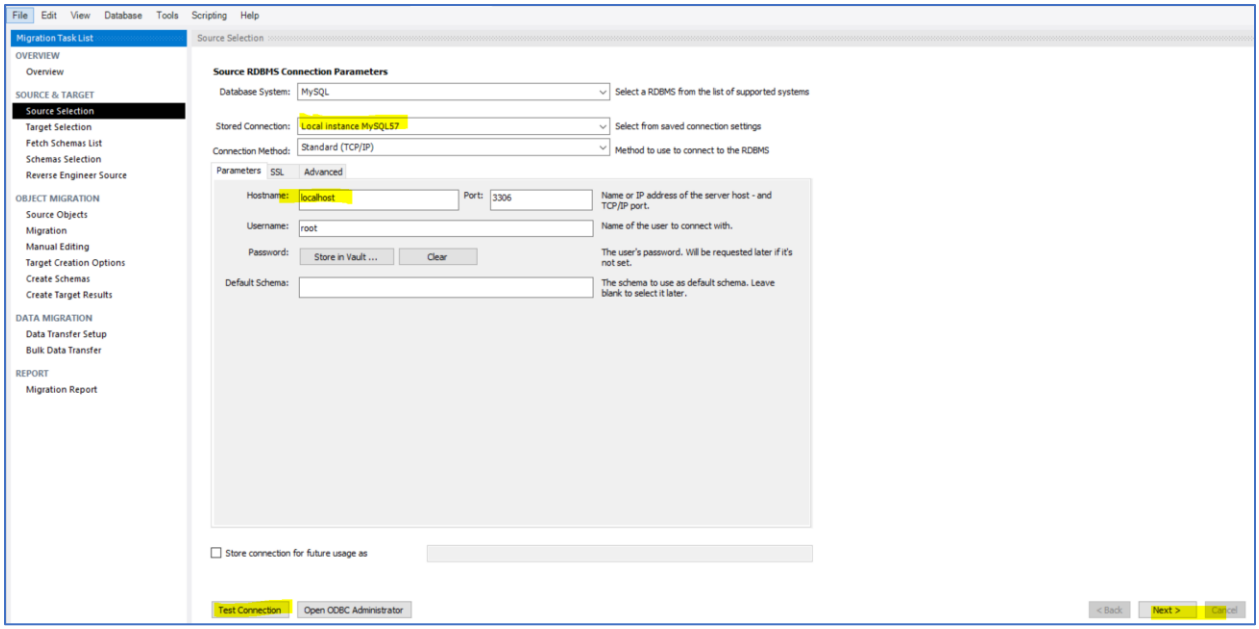
Step 1

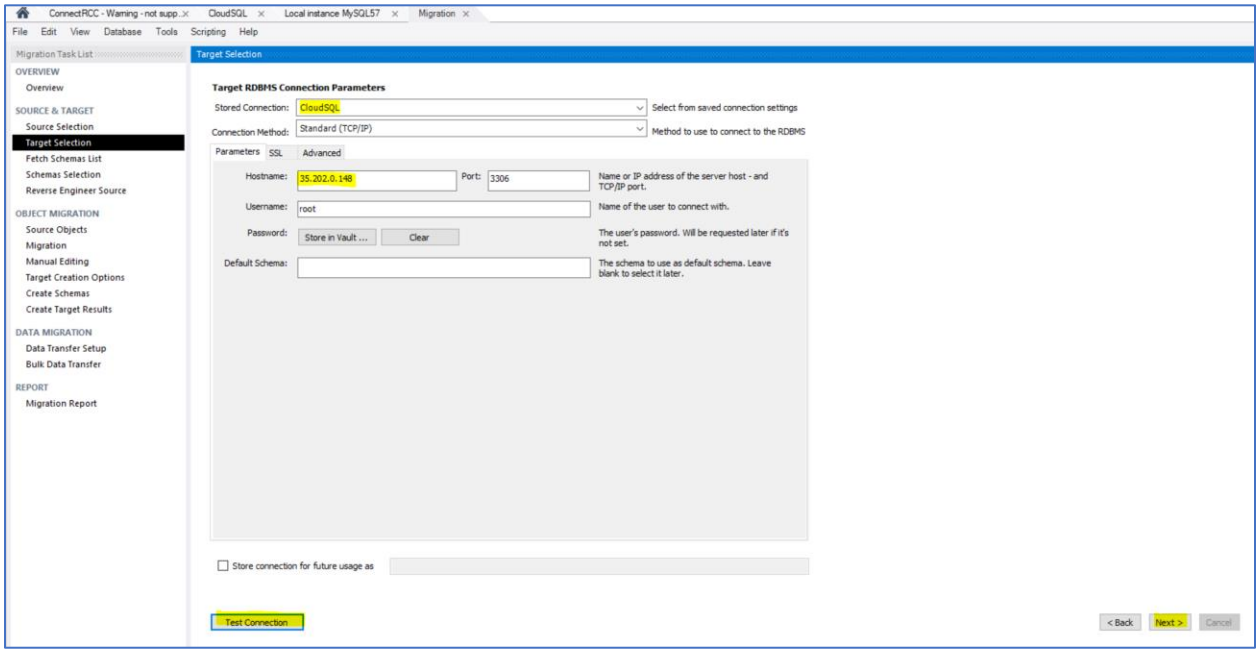
From MySQL workbench go to Database > Migration Wizard



Step 2

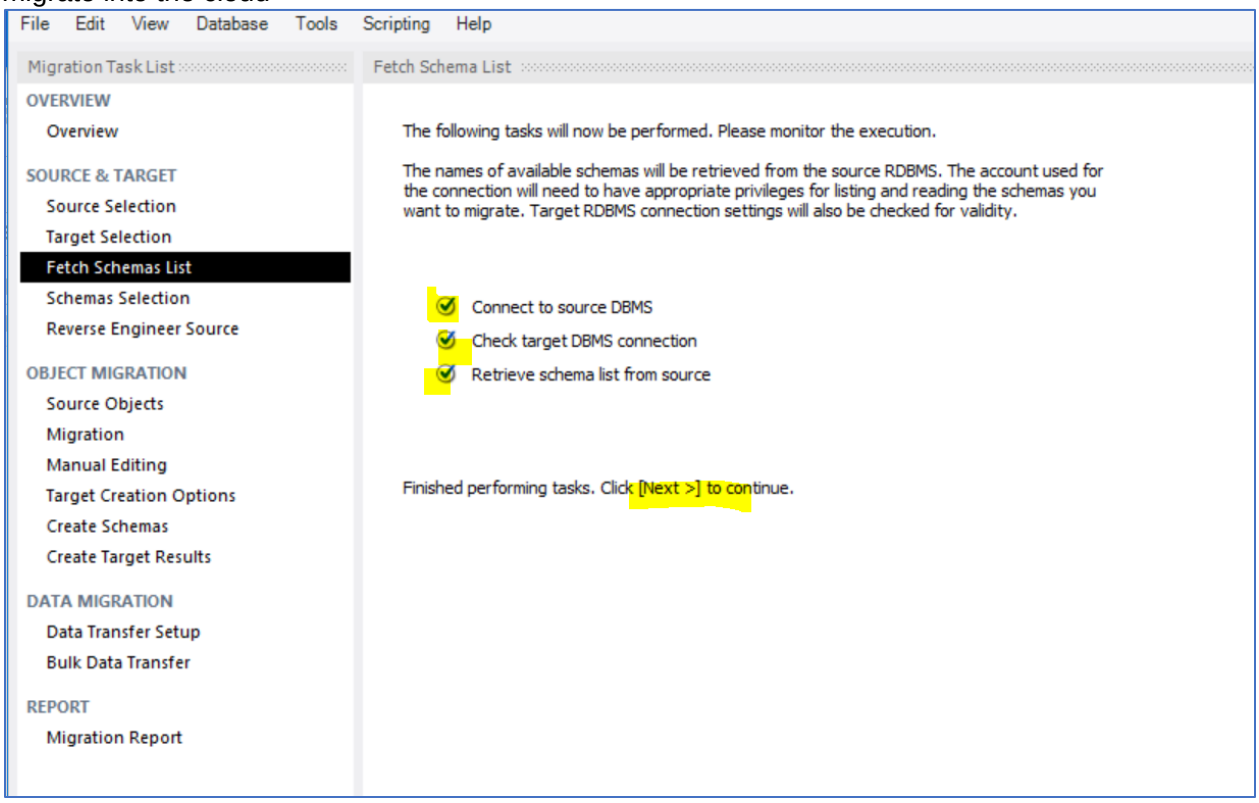
Select Source & Target

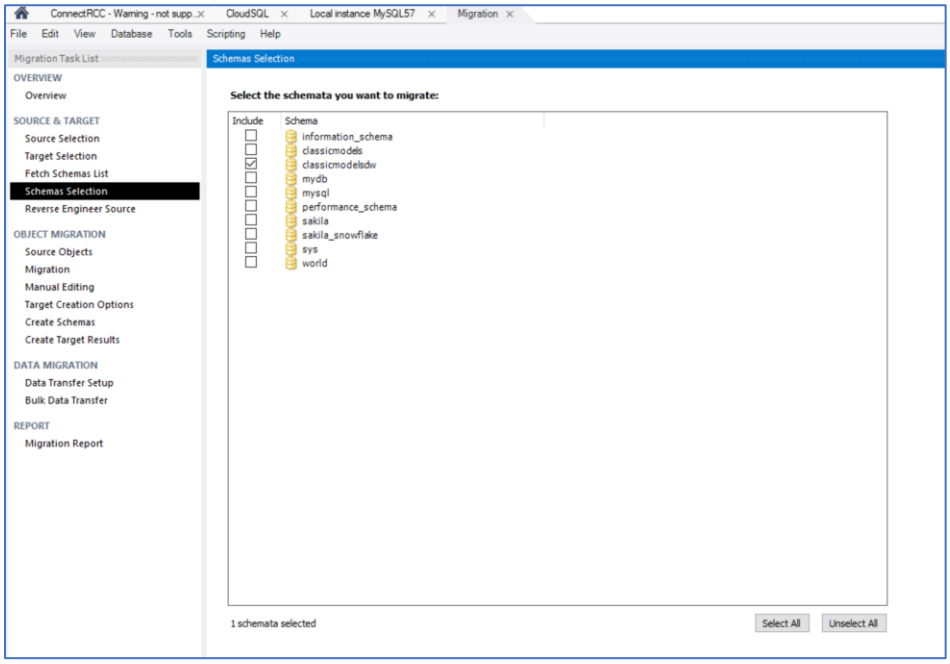

File Edit View Database Tools Scripting Help
Migration Task List
Source Selection
OVERVIEW
Overview
SOURCE & TARGET
Source Selection
Target Selection
Fetch Schemas List
Schemas Selection
Reverse Engineer Source
OBJECT MIGRATION
Source Objects
Migration
Manual Editing
Target Creation Options
Create Schemas
Create Target Results
DATA MIGRATION
Data Transfer Setup
Bulk Data Transfer
REPORT
Migration Report
Source RDBMS Connection Parameters
Database System: MySQL
Select a RDBMS from the list of supported systems
Stored Connection: Local instance MySQL57
Select from saved connection settings
Connection Method: Standard (TCP/IP)
Method to use to connect to the RDBMS
Parameters SSL Advanced
Hostname: localhost
Port: 3306
Name or IP address of the server host - and TCP/IP port.
Username: root
Name of the user to connect with.
Password: Store in Vault ... Clear
The user's password. Will be requested later if it's not set.
Default Schema:
The schema to use as default schema. Leave blank to select it later.
Store connection for future usage as
Test Connection
Open ODBC Administrator
< Back
Next >
Cancel

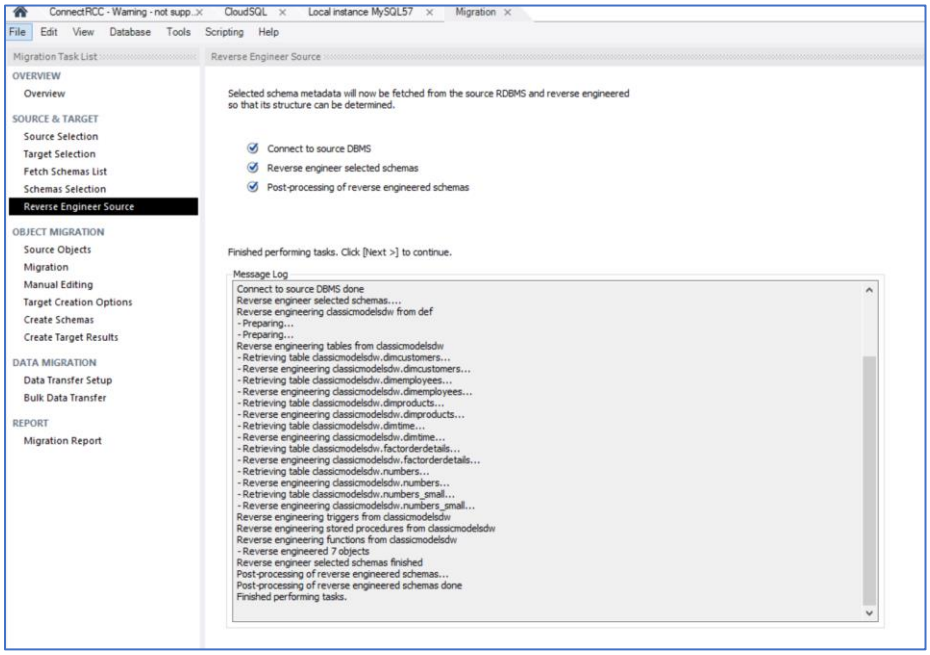

ConnectRCC - Warning - not supp...
CloudSQL
Local instance MySQL57
Migration
File Edit View Database Tools Scripting Help
Migration Task List
Target Selection
OVERVIEW
Overview
SOURCE & TARGET
Source Selection
Target Selection
Fetch Schemas List
Schemas Selection
Reverse Engineer Source
OBJECT MIGRATION
Source Objects
Migration
Manual Editing
Target Creation Options
Create Schemas
Create Target Results
DATA MIGRATION
Data Transfer Setup
Bulk Data Transfer
REPORT
Migration Report
Target RDBMS Connection Parameters
Stored Connection: CloudSQL
Select from saved connection settings
Connection Method: Standard (TCP/IP)
Method to use to connect to the RDBMS
Parameters SSL Advanced
Hostname: 35.202.0.148
Port: 3306
Name or IP address of the server host - and TCP/IP port.
Username: root
Name of the user to connect with.
Password: Store in Vault ... Clear
The user's password. Will be requested later if it's not set.
Default Schema:
The schema to use as default schema. Leave blank to select it later.
Store connection for future usage as
Test Connection
< Back
Next >
Cancel

Step 3

Make sure the connections are successful & select to Classic Models Datawarehouse schema to migrate into the cloud

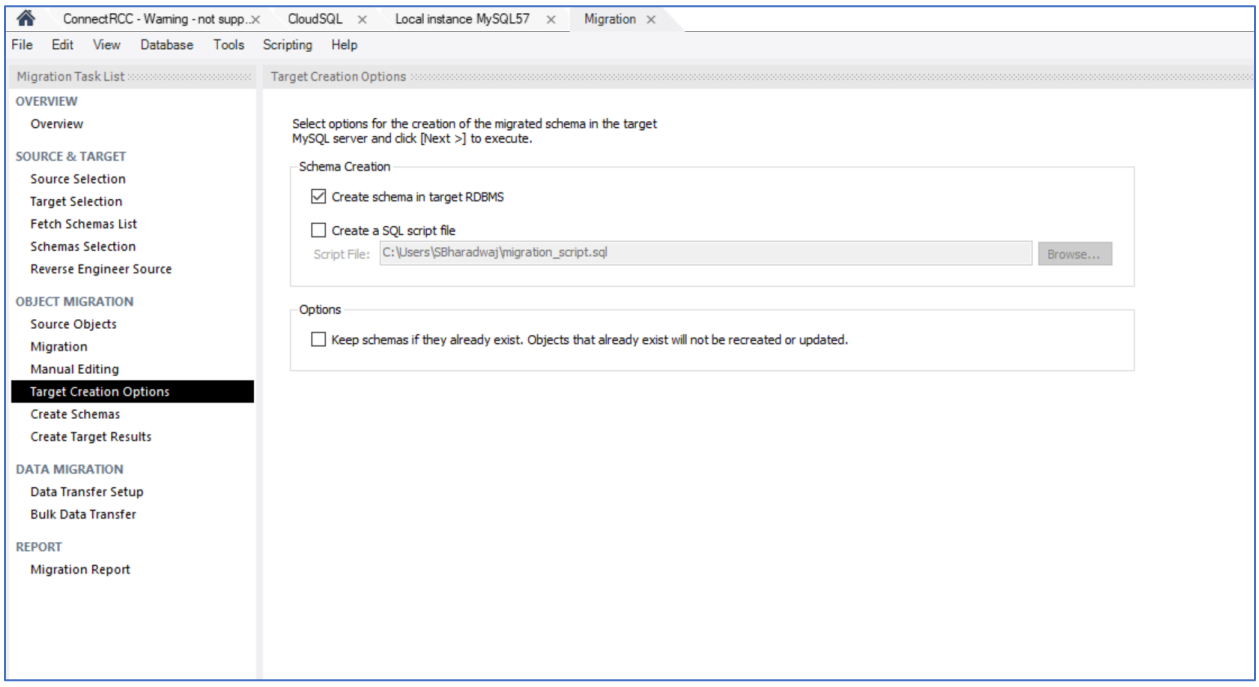


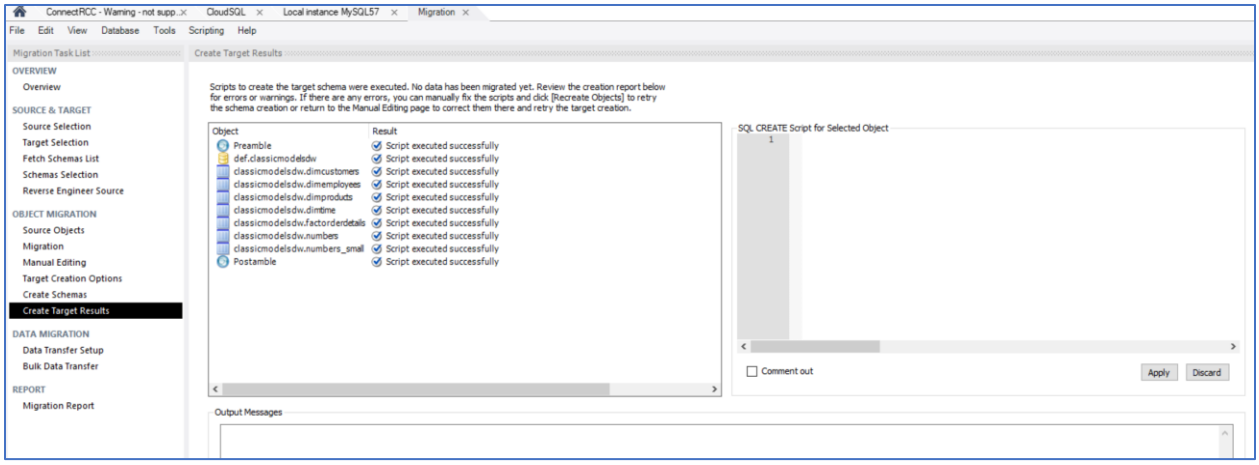


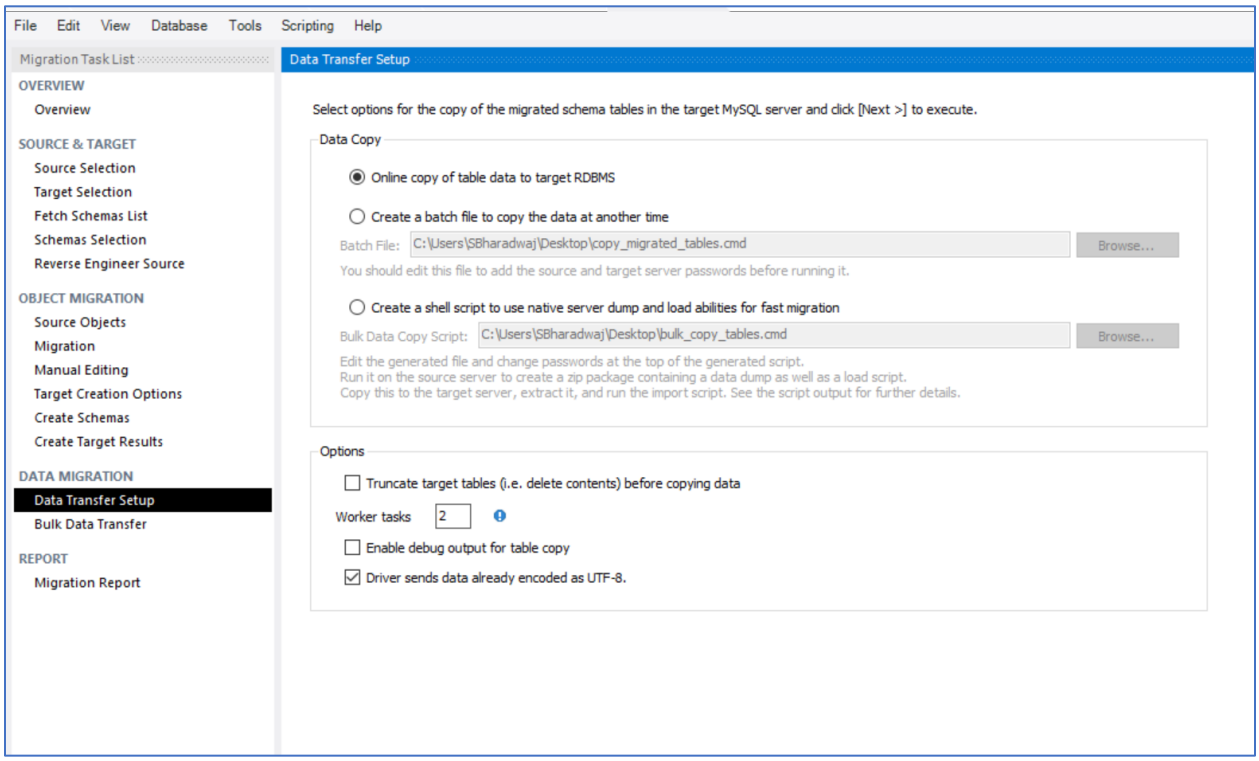


Step 4

Cleck Next > all the way through







Step 5

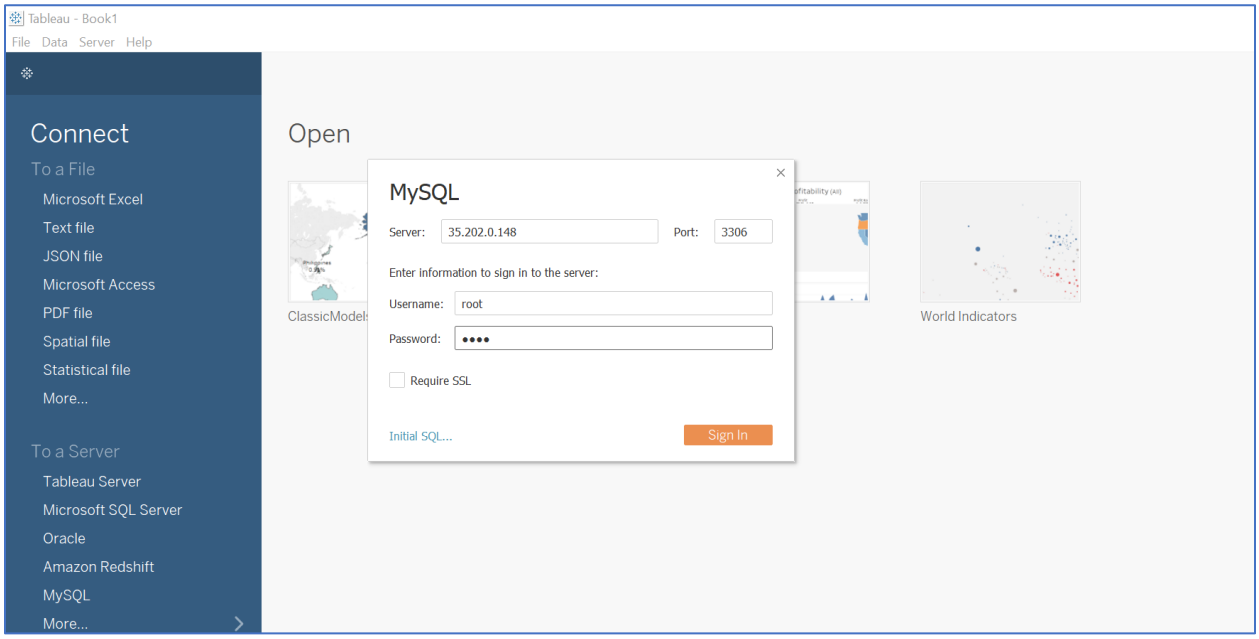
Login to cloudSQL to to make sure the tables/schema is migrated.

Migrate Data

To migrate data from local instance to Cloud SQL.

Connect to Tableau

To migrate data from local instance to Cloud SQL.



Pub/Sub

Install Google Cloud SDK
https://cloud.google.com/sdk/docs/quickstart-windows/
https://cloud.google.com/sdk/docs/quickstart-macos/

Step 1

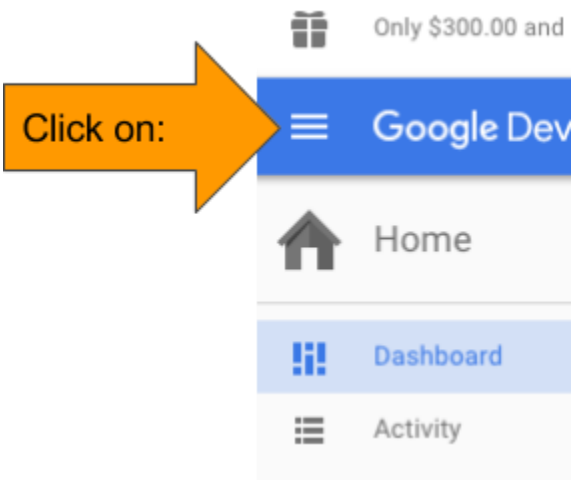
Browse to https://cloud.google.com/

Step 2

Click on **Console**.

Step 3

Click on the Menu (three horizontal lines):

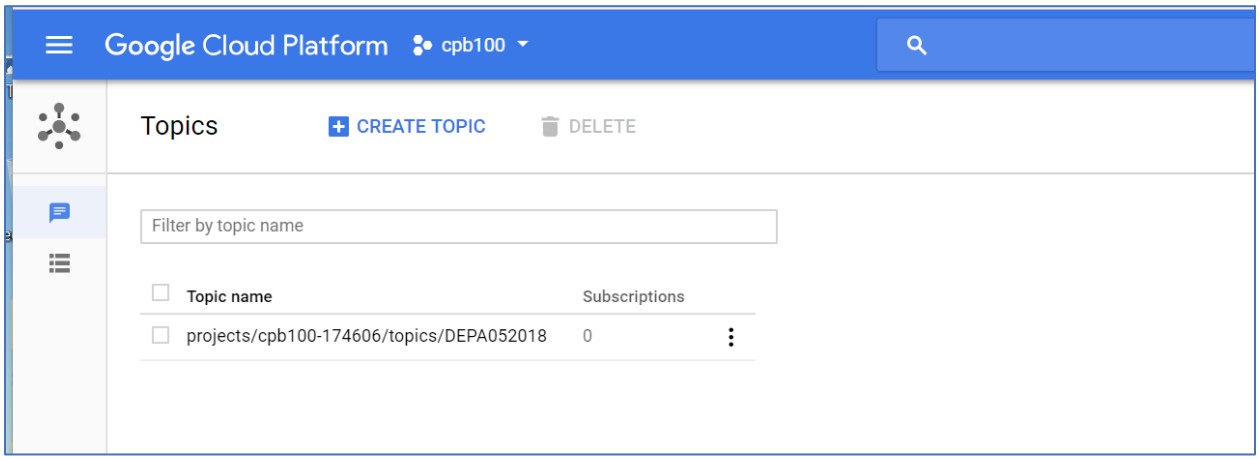


Step 4

Select **BIG DATA** > Pub/Sub

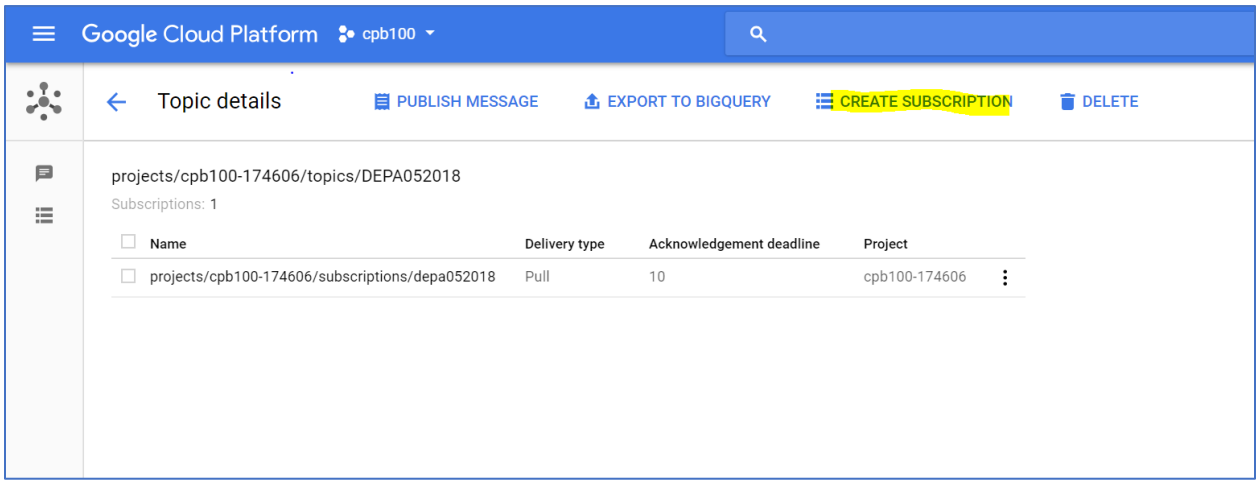
Step 5

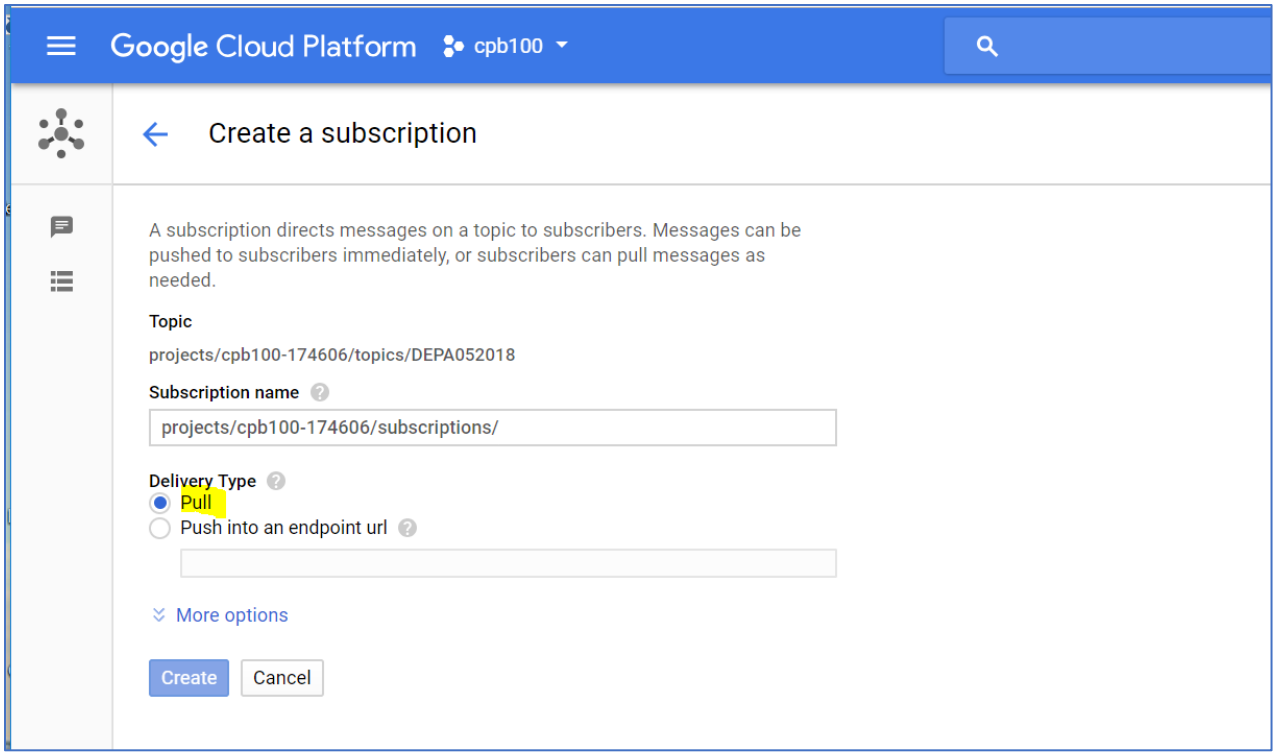
Click on CREATE TOPIC



Step 6

Click on CREATE SUBSCRIPTION





Step 6

Publish event
gcloud pubsub topics publish <Topic-name> --message <text msg>

Subscriber data pull
gcloud pubsub subscriptions pull --auto-ack <subscription-name>

Dataflow

Step 1

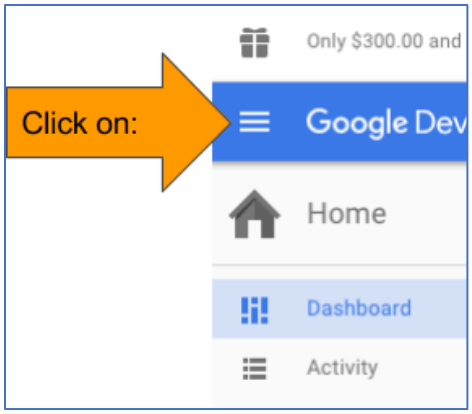
Browse to https://cloud.google.com/

Step 2

Click on **Console**.

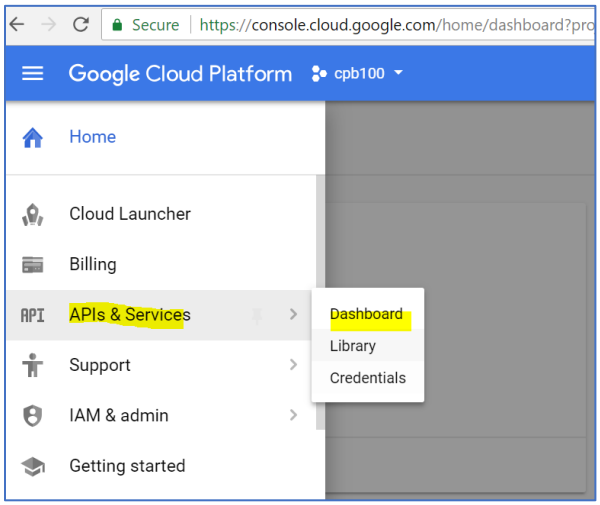
Step 3

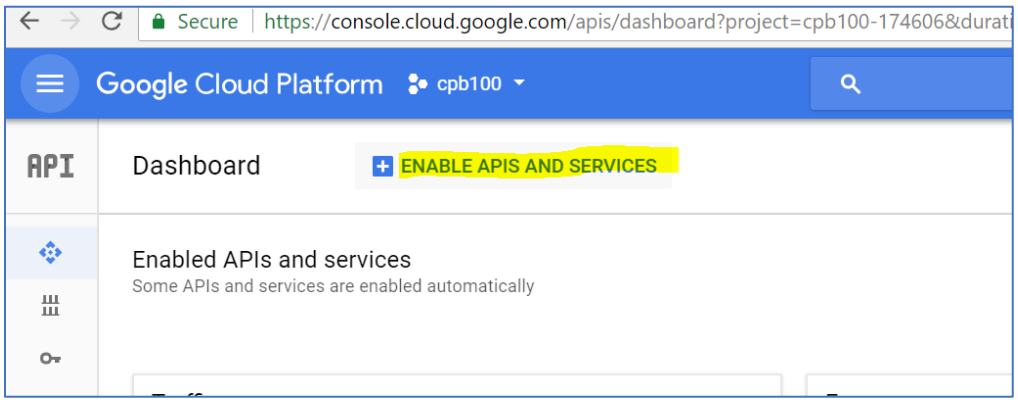
Click on the Menu (three horizontal lines):



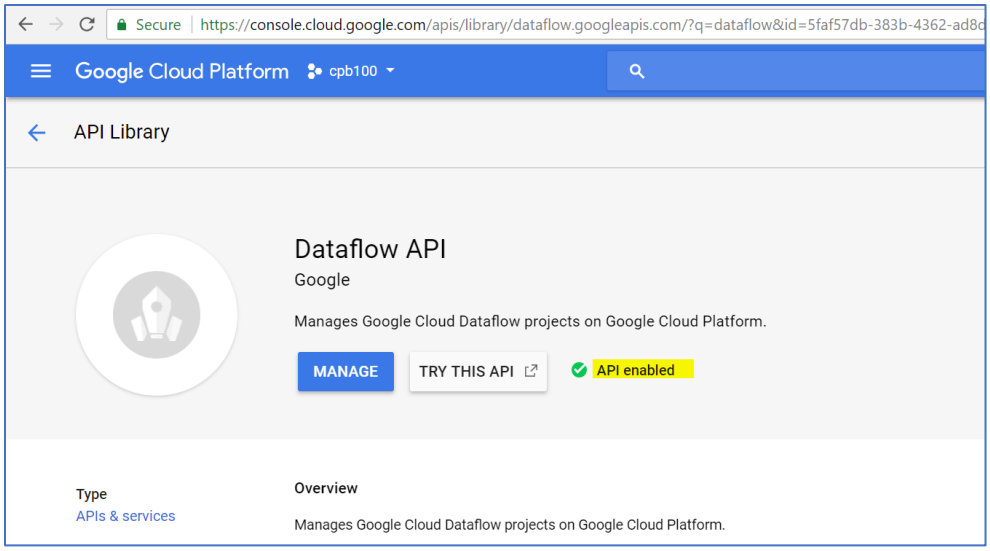
Step 4

Enable Data flow API





Enable Data flow API and once enabled we should see the below.



Step 5

Setup data flow jobs

Secure | https://console.cloud.google.com/dataflow/createjob?project=cpb100-174606

Google Cloud Platform cpb100

Dataflow

Create job from template

Job name
Specify a job name for the Dataflow job.

my-job-name

Cloud Dataflow Regional Endpoint
The regional endpoint where metadata will be stored for the job. Learn more

us-central1

Cloud Dataflow template
A pipeline that reads from a Pub/Sub topic and writes messages text files stored in GCS. Note that this pipeline assumes no newlines in the body of the Pub/Sub message and thus each message becomes a single line in the output file.

Cloud PubSub to GCS Text

Parameters

Input Cloud Pub/Sub topic
Cloud Pub/Sub topic to read the input from. The topic name should be in the format of projects/<project-id>/topics/<topic-name>.

Output Cloud Storage directory
Path and filename prefix for writing output files (ex: gs://bucket-name/path/). This value must end in a slash.

Output file prefix
The prefix to place on each windowed file (ex: output-).

Output file suffix (Optional)
[Optional] The suffix to place on each windowed file. Typically a file extension (.txt, .csv, etc.).

The shard template (Optional)
[Optional] The shard template defines the unique/dynamic portion of each windowed file. Recommended to use the default (W-P-SS-of-NN). At runtime, 'W' is replaced with the window date range and 'P' is replaced with the pane info. Repeating sequences of the letters 'S' or 'N' (example: SSS-NNN) are replaced with the shard number and number of shards respectively. The pipeline assumes a single file output and thus will produce the

Secure | https://console.cloud.google.com/dataflow/jobsDetail/locations/us-central1/jobs/2018-05-07_22_14_29-12220673706090425688?project=cpb100-174606

Google Cloud Platform cpb100

shree123