Lab BigQuery in JupyterLab on Vertex AI

setting

Open BigQuery Console

1.In the Google Cloud Console, on the **Navigation menu** , click **BigQuery**. The **Welcome to BigQuery in the Cloud Console** dialog opens. This dialog provides a link to the quickstart guide and lists UI updates.

2.Click **Done** to close the dialog.

Start a JupyterLab Notebook Instance

1.Click on the **Navigation Menu**.

2.Navigate to **Artificial Intelligence**, **Vertex AI**, then to **Workbench**.

3.You'll be redirected to **User-Managed Notebooks** tab on the main page for **Notebooks** on Vertex AI.

When the tab loads if you notice a link entitled **Enable Notebooks API**, click that link to allow the background Notebooks API to be upgraded. The upgrade will occur promptly.

Click on the **New Instance** icon on the top of the page.

4.In the menu that pops down, select the **Python 3** option.

5.A screen entitled **New notebook** will be shown. Leave the default options and click on **Create**.

6.After a few minutes, the Vertex AI console will have your instance name followed by **Open Jupyterlab**. Click **Open Jupyterlab**.

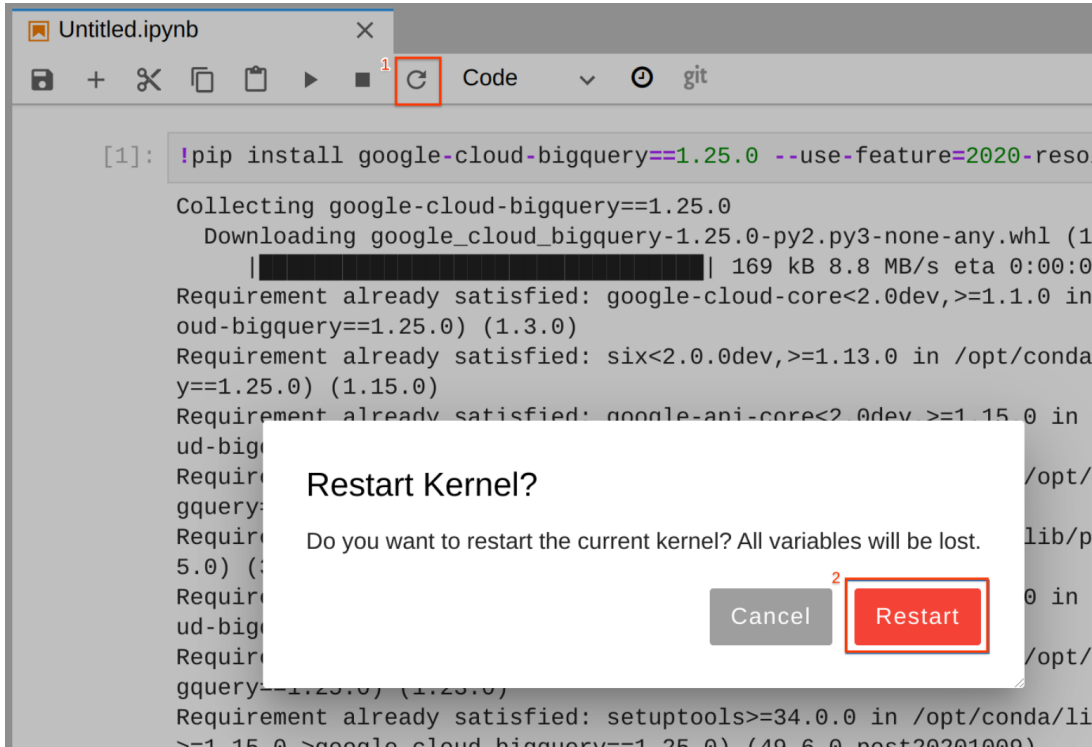
7.A new tab will open in your browser with the JupyterLab environment. Select **Python 3** under **Notebook**.

Execute a BigQuery query

1.Execute the following Python install command by hitting **Shift + Enter** in the first cell of the notebook to install the google-cloud-bigquery library at version 1.25.0.

```
!pip install google-cloud-bigquery==1.25.0 --use-feature=2020-resolver
```

Note: You may safely ignore the following notifications: **WARNING: --use-feature=2020-resolver...** and **ERROR: pip's dependency resolver...**.Restart the kernel by clicking **Restart kernel** icon > **Restart**.



Execute a BigQuery query

2.Enter the following query in the second cell of the notebook.

```
%%bigquery df
SELECT
  departure_delay,
  COUNT(1) AS num_flights,
  APPROX_QUANTILES(arrival_delay, 10) AS arrival_delay_deciles
FROM
  `bigquery-samples.airline_ontime_data.flights`
GROUP BY
  departure_delay
HAVING
  num_flights > 100
ORDER BY
  departure_delay ASC
```

Python (pandas)

The command makes use of the magic function %%bigquery. Magic functions in notebooks provide an alias for a system command. In this case, %%bigquery runs the query in the cell in BigQuery and stores the output in a Pandas DataFrame object named df.

3.Run the cell by hitting **Shift + Enter**, when the cursor is in the cell. Alternatively, if you navigate to the **Run** tab you can click on **Run Selected Cells**. Note the keyboard shortcut for this action in case it is not Shift + Enter. There should be no output when executing the command.

4.View the first five rows of the query's output by executing the following code in a new cell:

```
df.head()
```

Make a Plot with Pandas

We're going to use the Pandas DataFrame containing our query output to build a plot that depicts how arrival delays correspond to departure delays. Before continuing, if you are unfamiliar with Pandas the Ten Minute Getting Started Guide is recommended reading.

1.To get a DataFrame containing the data we need we first have to wrangle the raw query output. Enter the following code in a new cell to convert the list of arrival_delay_deciles into a Pandas Series object. The code also renames the resulting columns.

```
import pandas as pd
percentiles = df['arrival_delay_deciles'].apply(pd.Series)
percentiles.rename(columns = lambda x : '{0}%'.format(x*10), inplace=True)
percentiles.head()
```

2.Since we want to relate departure delay times to arrival delay times we have to concatenate our percentiles table to the departure_delay field in our original DataFrame. Execute the following code in a new cell:

df = pd.concat([df['departure_delay'], percentiles], axis=1)
df.head()

3.Before plotting the contents of our DataFrame, we'll want to drop extreme values stored in the 0% and 100% fields. Execute the following code in a new cell:

```
df.drop(labels=['0%', '100%'], axis=1, inplace=True)
df.plot(x='departure_delay', xlim=(-30,50), ylim=(-50,50));
```

[23]:
```
df.drop(labels=['0%', '100%'], axis=1, inplace=True)
df.plot(x='departure_delay', xlim=(-30,50), ylim=(-50,50));
```