Week 9 Lab
Submitted by: Shrikrishna Bhat

Table of Contents

9.1g BigQuery, BigLake *2*

Card 3 Create dataset **2**

Card 4. Query data **3**

Card 9 – Query Data **7**

9.2g Jupyter Notebooks *8*

Card 3 BigQuery query **8**

Card 6 Run queries **8**

Card 8 Mobility **9**

Card 9 Airport Traffic **10**

Card 10 Mortality **11**

Card 11 – Run Queries **12**

Card 12 Write queries **14**

9.3g Dataproc Setup *16*

Card 6 – Run Computation **16**

9.4g Dataflow *18*

Card 3 Beam Code **18**

Card 4 - Run pipe locally **19**

Card 5 – Dataflow Lab #2 (Word Count) **20**

Card 6 Run Code locally **21**

Card 9 Run Code using Dataflow runner **22**

Card 12 View raw data from PubSub **23**

Card 14 Run Dataflow job from template **24**

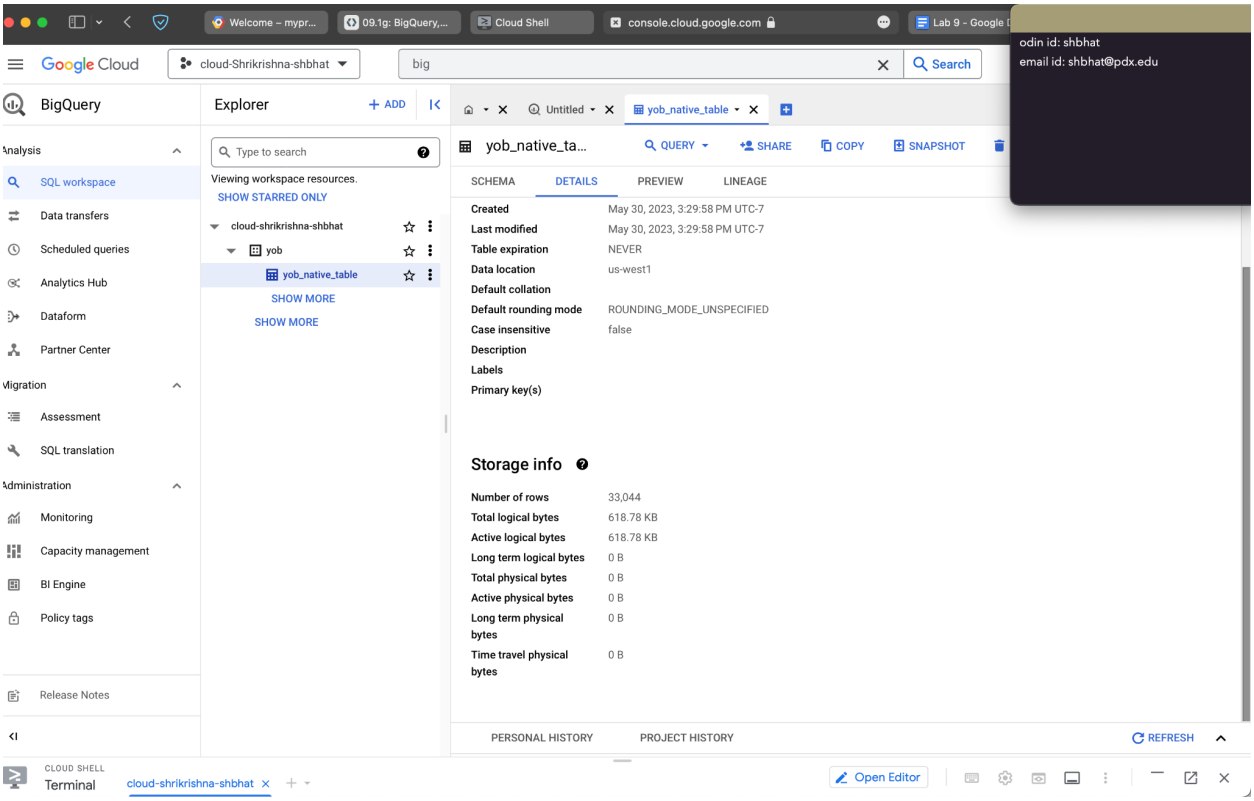
Card 15 Query data in BigQuery **25**

Card 16 – Data visualization **28**

9.1g BigQuery, BigLake

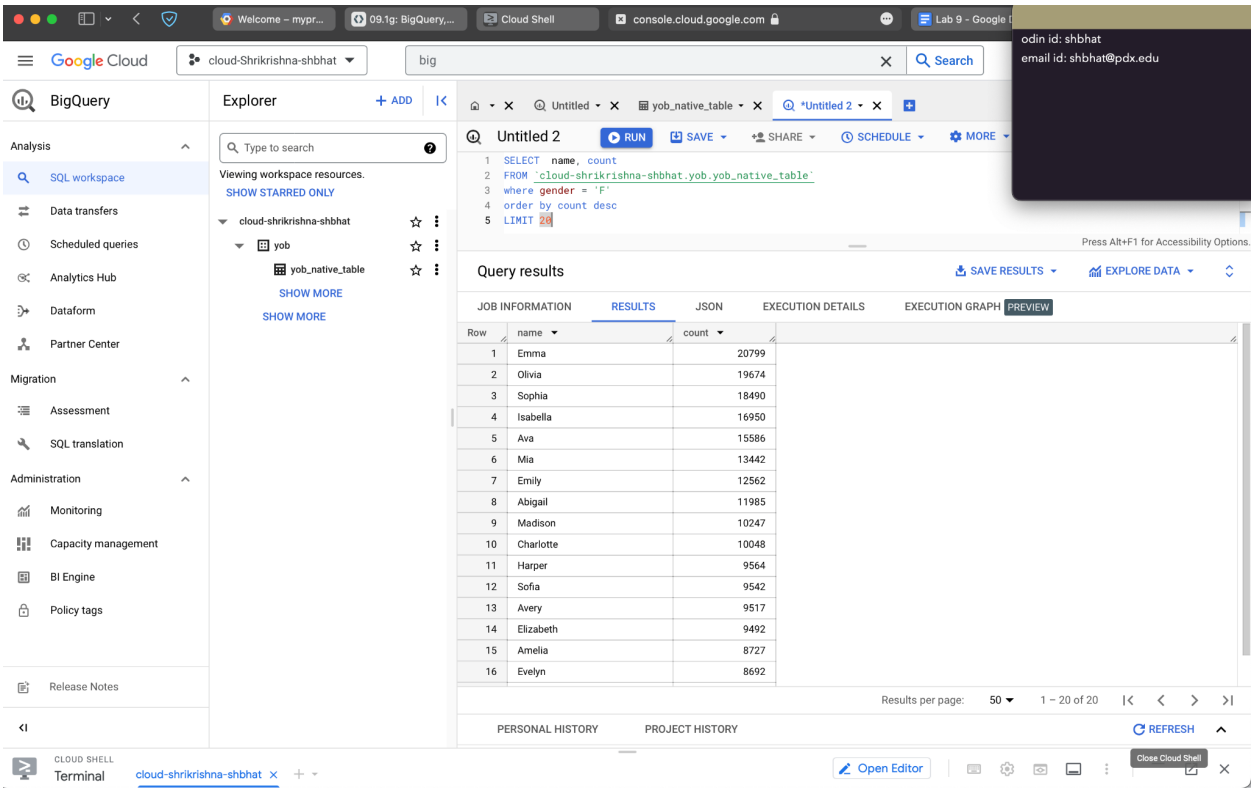
Card 3 Create dataset

9.1.1Take a screenshot of the table's details that includes the number of rows in the table.

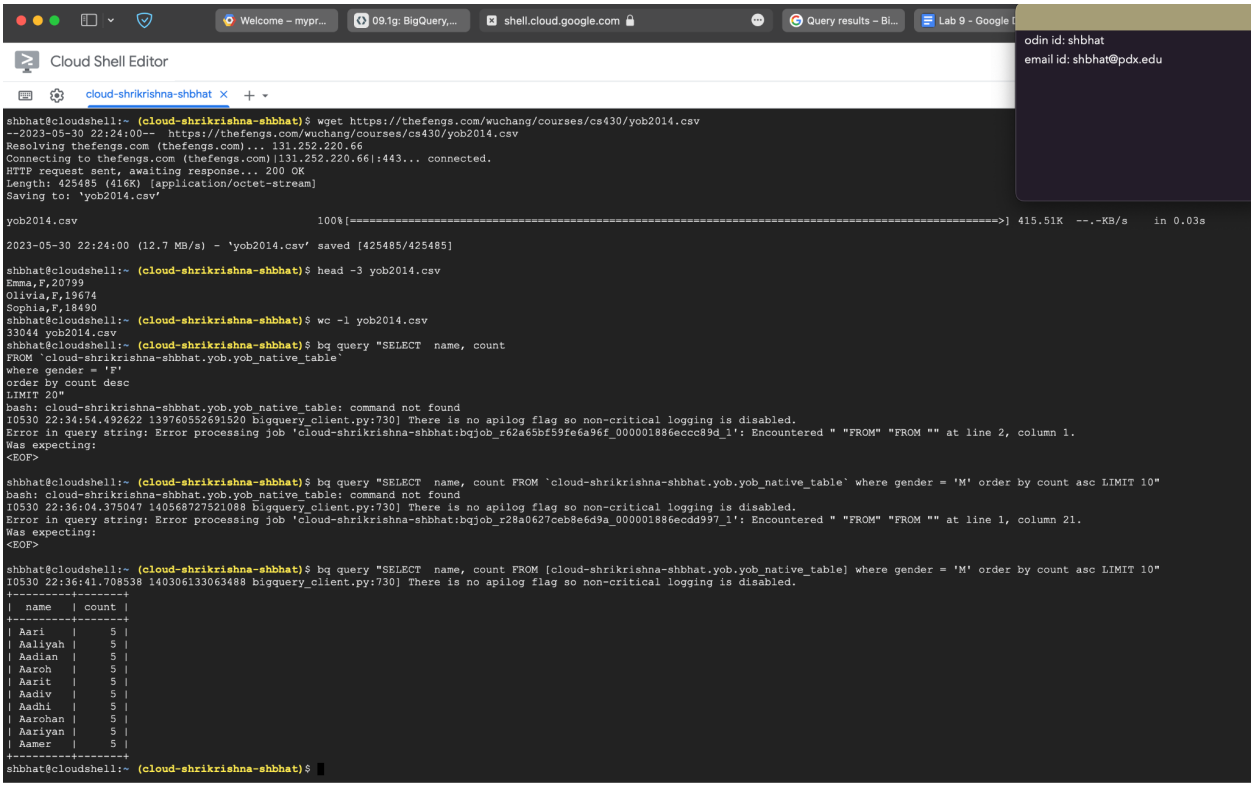


Card 4. Query data

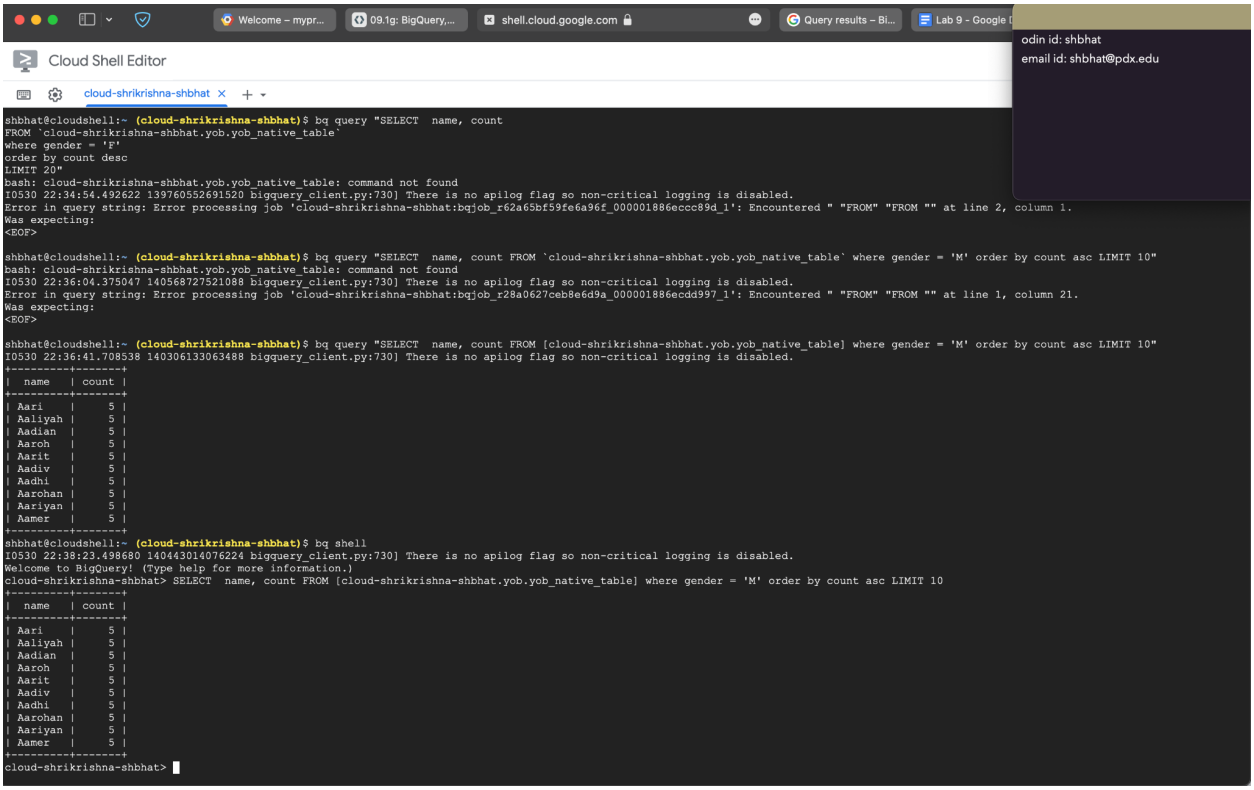
9.1.2 Screenshot the query results and include it in your lab notebook



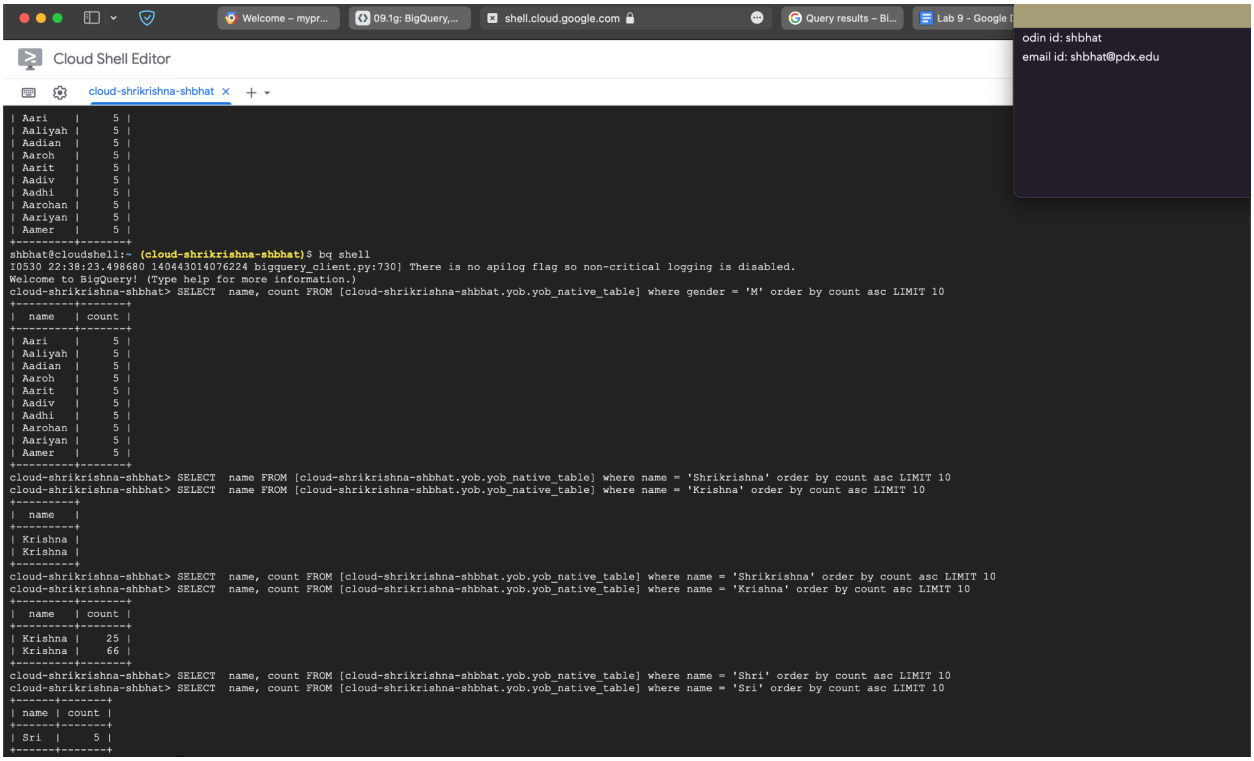
9.1.3 Screenshot your results and include it in your lab notebook



9.1.4 Screenshot your results and include it in your lab notebook

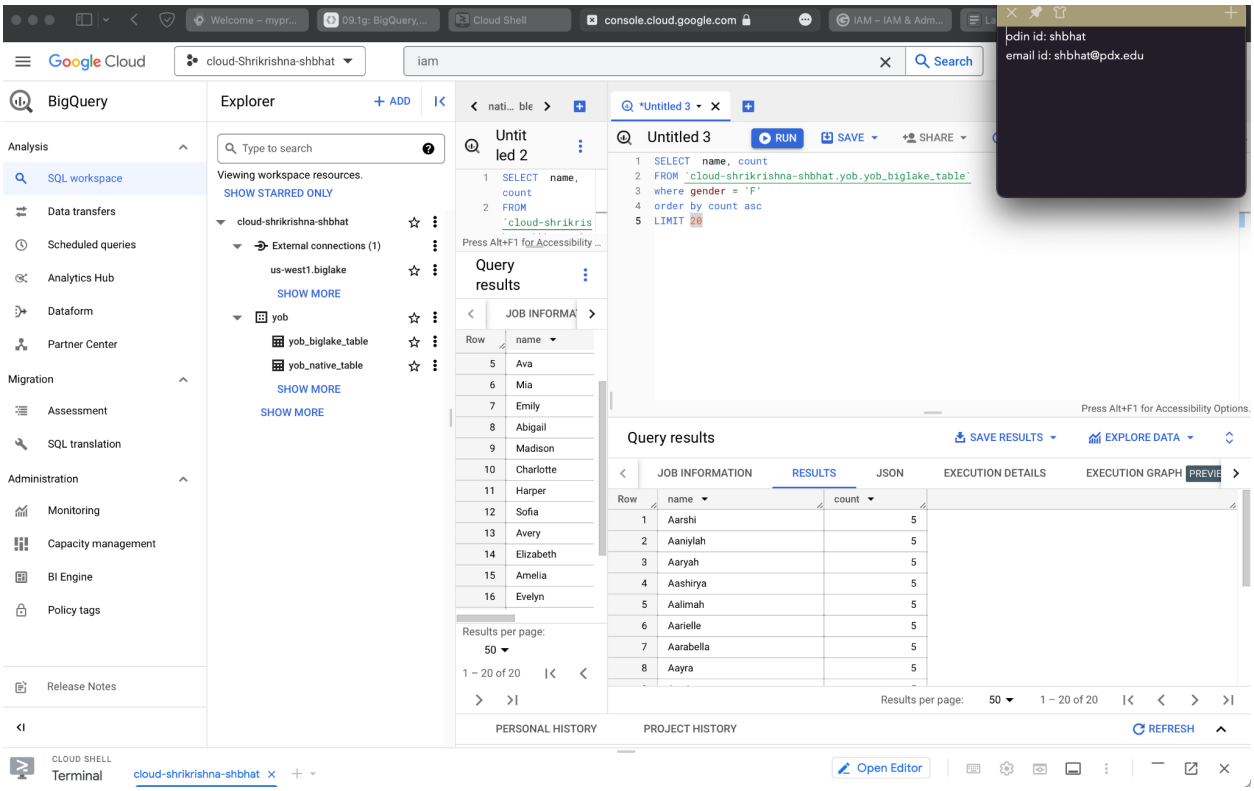


9.1.5 Screenshot your results and include it in your lab notebook



Card 9 – Query Data

9.1.6 Screenshot the query results and include it in your lab notebook



9.2g Jupyter Notebooks

Card 3 BigQuery query

9.2.1 How much less data does this query process compared to the size of the table?

This query will run with 18 gb less data

9.2.2 How many twins were born during this time range?

375362

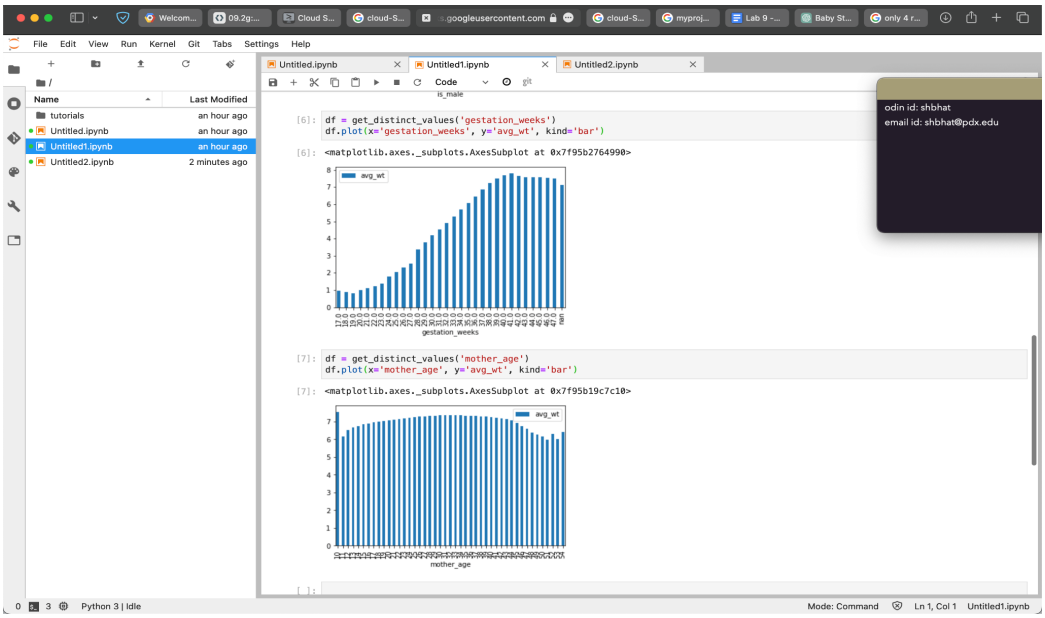
9.2.3 How much lighter on average are they compared to single babies?

2.17

Card 6 Run queries

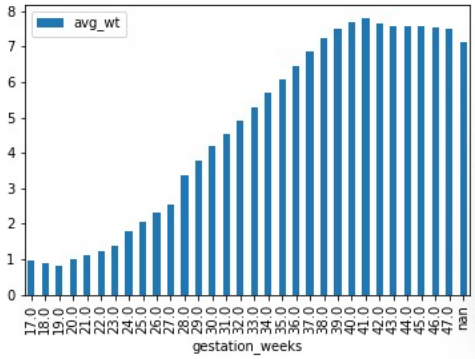
9.2.4 Show the plots generated for the two most important features for your lab notebook

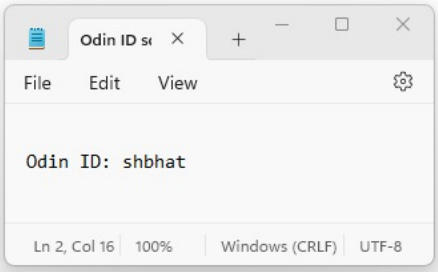
Gestation_week and plurality



Initially had taken wrong screenshot hence had to run again and take screenshot so the screenshot differs from the rest.

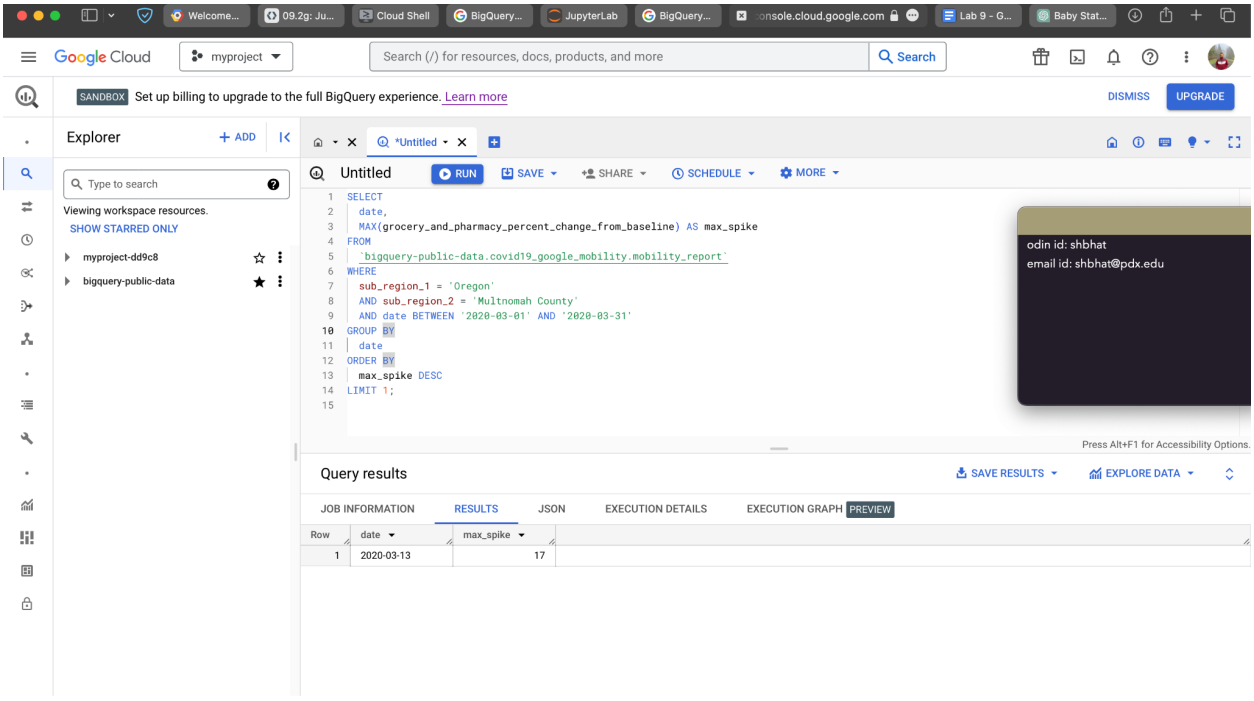
```
[9]: df = get_distinct_values('gestation_weeks')
     df.plot(x='gestation_weeks', y='avg_wt', kind='bar')

[9]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9436d53810>
```



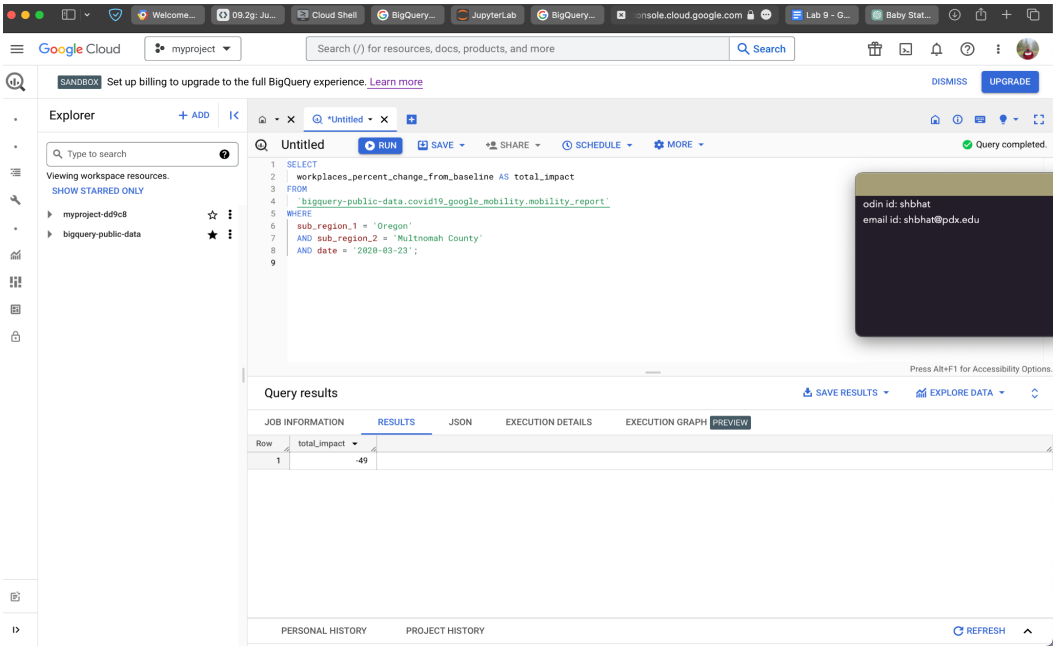


Card 8 Mobility

9.2.5 What day saw the largest spike in trips to grocery and pharmacy stores? Answer – 2020-03-13



9.2.6 On the day the stay-at-home order took effect (3/23/2020), what was the total impact on workplace trips? Answer: -49



Card 9 Airport Traffic

9.2.7 Which three airports were impacted the most in April 2020 (the month when lockdowns became widespread)?

Newark Liberty International

Daniel K. Inouye International

Chicago O'Hare International

9.2.8 Run the query again using the month of August 2020. Which three airports were impacted the most?

Newark Liberty International

Charlotte Douglas International

Dallas/Fort Worth International

Card 10 Mortality

9.2.10 What table and columns identify the place name, the starting date, and the number of excess deaths from COVID-19?

The table that identifies the place name, starting date, and number of excess deaths from COVID-19 is the "excess_deaths" table. The relevant columns are:

placename: The place in the country reported

start_date: The first date included in the period

excess_deaths: The number of deaths minus the expected deaths

9.2.11 What table and columns identify the date, county, and deaths from COVID-19?

The table that identifies the date, county, and deaths from COVID-19 is the "us_counties" table. The relevant columns are:

date: Date reported

county: County in the specified state

deaths: The total number of confirmed deaths of COVID-19

9.2.12 What table and columns identify the date, state, and confirmed cases of COVID-19?

The table that identifies the date, state, and confirmed cases of COVID-19 is the "us_states" table. The relevant columns are:

date: Date reported

state_name: State reported

confirmed_cases: The total number of confirmed cases of COVID-19

9.2.13 What table and columns identify a county code and the percentage of its residents that report they always wear masks?

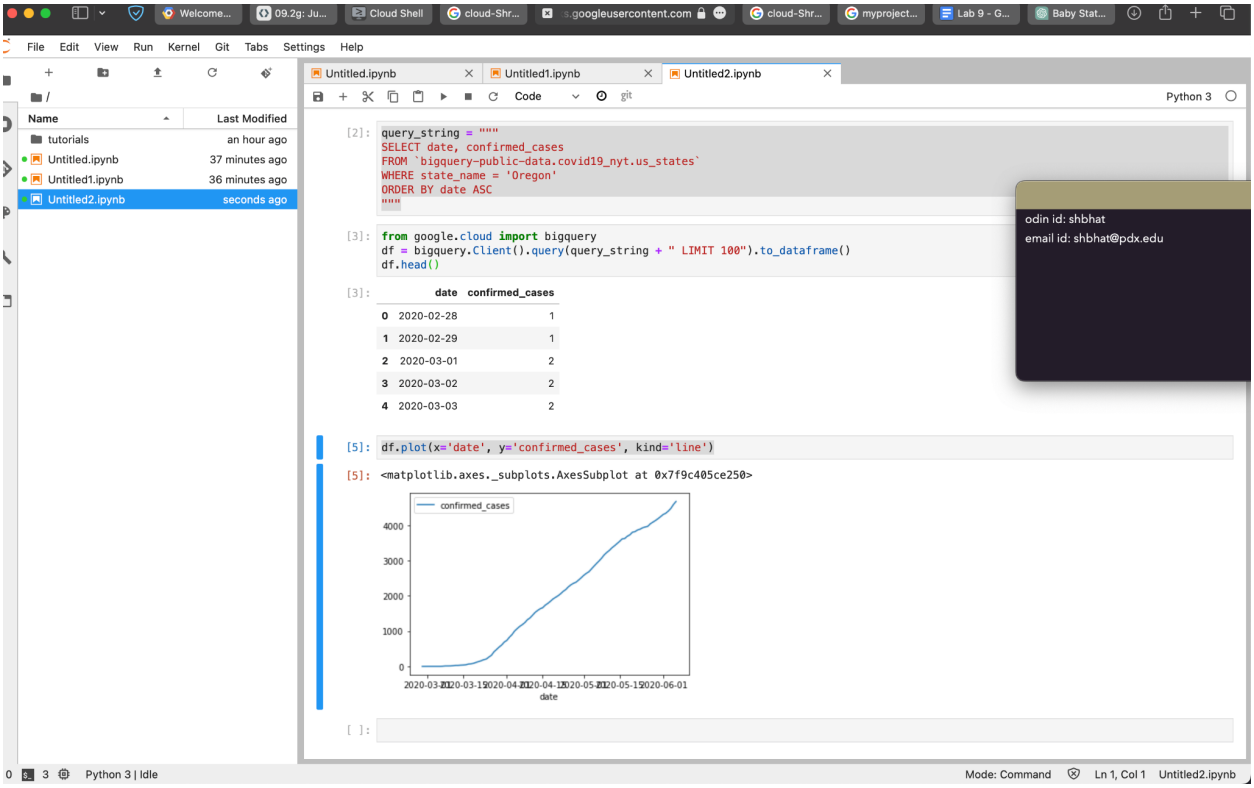
The table that identifies a county code and the percentage of its residents that report they always wear masks is the "mask_use_by_county" table. The relevant columns are:

county_fips_code: Standard geographic identifier for the county

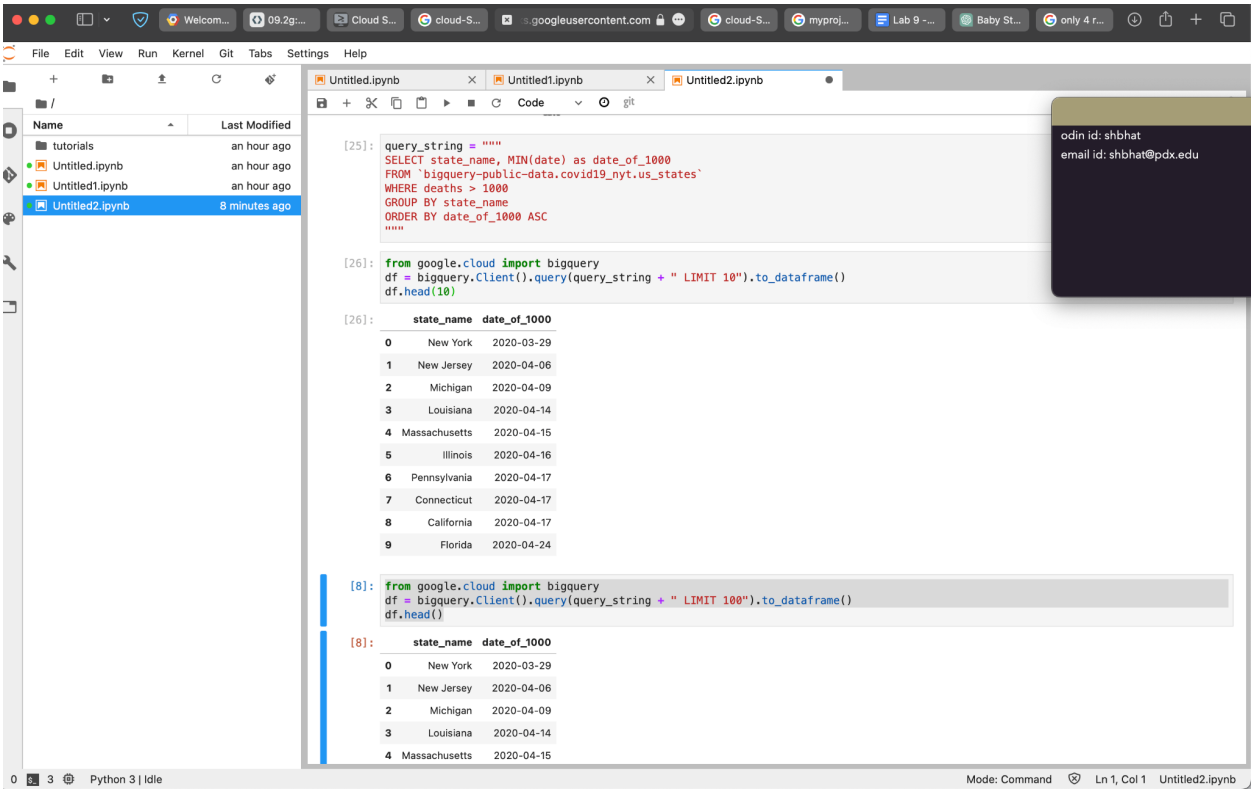
always: The estimated share of people in this county who would say always in response to the mask-wearing question

Card 11 – Run Queries

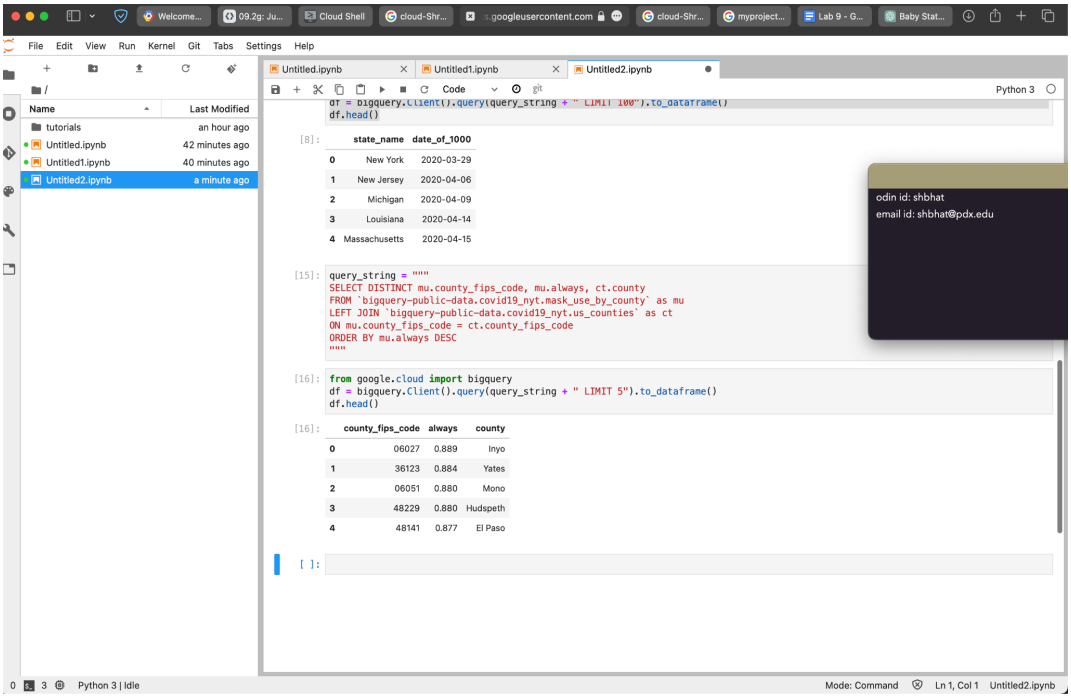
9.2.14 Show a screenshot of the plot and the code used to generate it for your lab notebook



9.2.15 From within your Jupyter notebook, run the query and write code that shows the first 10 states that reached 1000 deaths from COVID-19. Take a screenshot for your lab notebook.

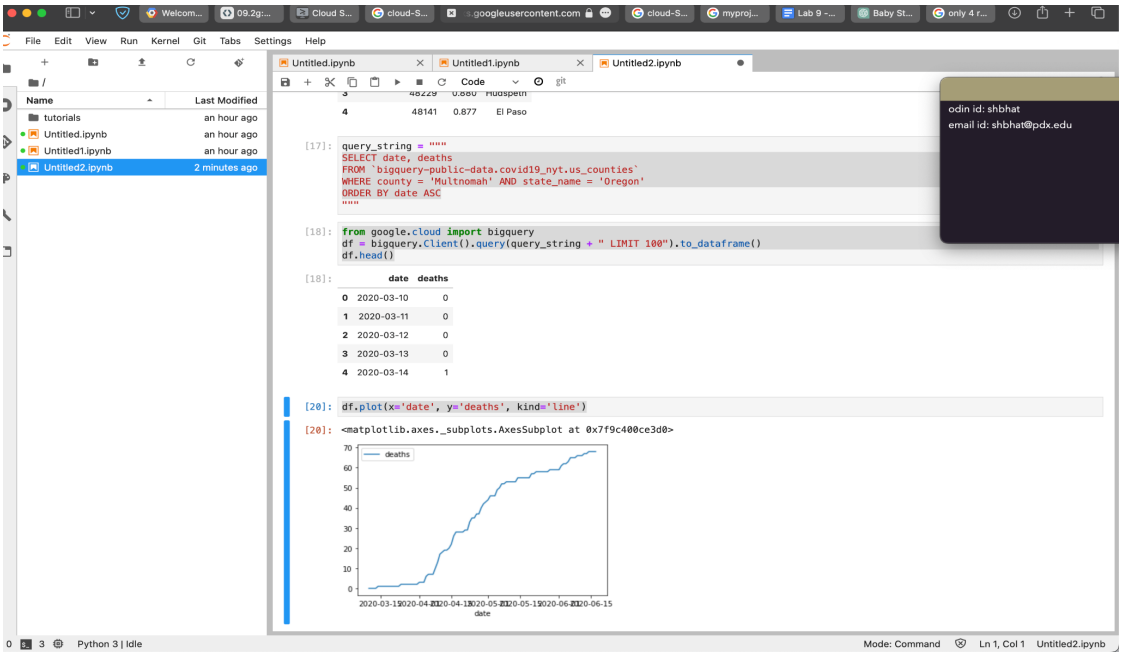


9.2.16 Take a screenshot for your lab notebook of the Top 5 counties and the states they are located in.

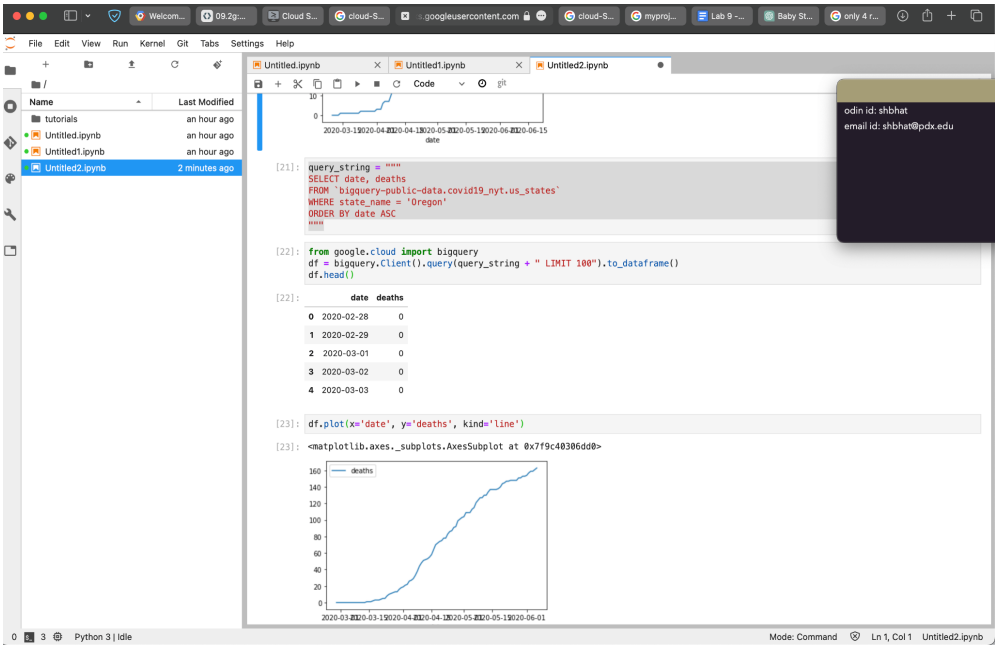


Card 12 Write queries

9.2.17 Plot the results and take a screenshot for your lab notebook.



9.2.18 Plot the results and take a screenshot for your lab notebook.



9.3g Dataproc Setup

Card 6 – Run Computation

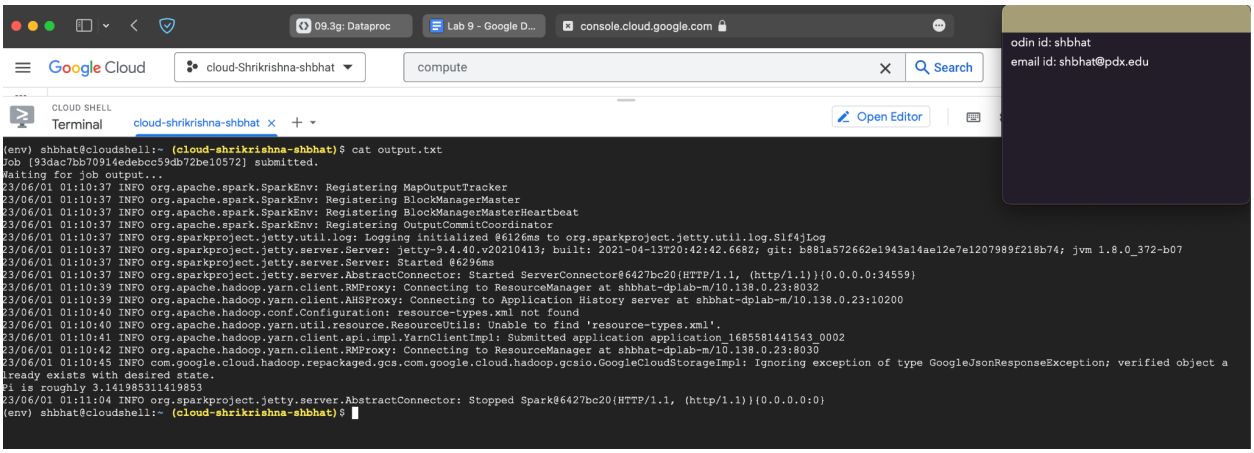
9.3.1 For your lab notebook:

- **How long did the job take to execute?**

  About 30s

- **Examine output.txt and show the estimate of π calculated**

  Pi is roughly 3.141985311419853



Card 8 – Run Computation again

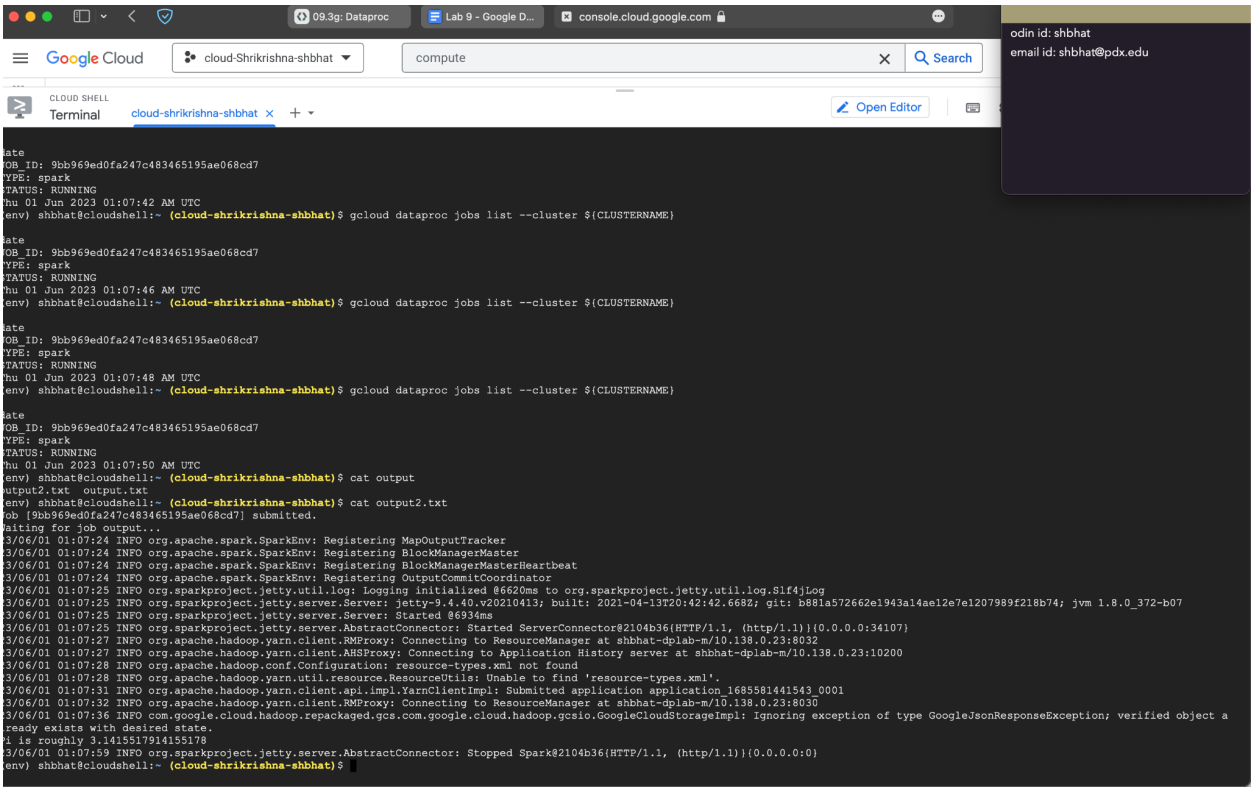
9.3.2 Answer the questions in the lab.

- **How long did the job take to execute? How much faster did it take?**

  It took 11s, it was 19s faster

- **Examine output2.txt and show the estimate of π calculated.**

  Pi is roughly 3.14155179141551 78


09.3g: Dataproc
Lab 9 - Google D...
console.cloud.google.com
odin id: shbhat
email id: shbhat@pdx.edu
Google Cloud
cloud-Shrikrishna-shbhat
compute
Search
CLOUD SHELL
Terminal
cloud-shrikrishna-shbhat
Open Editor
ate
OB_ID: 9bb969ed0fa247c483465195ae068cd7
YPE: spark
TATUS: RUNNING
hu 01 Jun 2023 01:07:42 AM UTC
env) shbhat@cloudshell:~ (cloud-shrikrishna-shbhat)$ gcloud dataproc jobs list --cluster ${CLUSTERNAME}
ate
OB_ID: 9bb969ed0fa247c483465195ae068cd7
YPE: spark
TATUS: RUNNING
hu 01 Jun 2023 01:07:46 AM UTC
env) shbhat@cloudshell:~ (cloud-shrikrishna-shbhat)$ gcloud dataproc jobs list --cluster ${CLUSTERNAME}
ate
OB_ID: 9bb969ed0fa247c483465195ae068cd7
YPE: spark
TATUS: RUNNING
hu 01 Jun 2023 01:07:48 AM UTC
env) shbhat@cloudshell:~ (cloud-shrikrishna-shbhat)$ gcloud dataproc jobs list --cluster ${CLUSTERNAME}
ate
OB_ID: 9bb969ed0fa247c483465195ae068cd7
YPE: spark
TATUS: RUNNING
hu 01 Jun 2023 01:07:50 AM UTC
env) shbhat@cloudshell:~ (cloud-shrikrishna-shbhat)$ cat output
utput2.txt output.txt
env) shbhat@cloudshell:~ (cloud-shrikrishna-shbhat)$ cat output2.txt
ob [9bb969ed0fa247c483465195ae068cd7] submitted.
aiting for job output...
3/06/01 01:07:24 INFO org.apache.spark.SparkEnv: Registering MapOutputTracker
3/06/01 01:07:24 INFO org.apache.spark.SparkEnv: Registering BlockManagerMaster
3/06/01 01:07:24 INFO org.apache.spark.SparkEnv: Registering BlockManagerMasterHeartbeat
3/06/01 01:07:24 INFO org.apache.spark.SparkEnv: Registering OutputCommitCoordinator
3/06/01 01:07:25 INFO org.sparkproject.jetty.util.log: Logging initialized @6620ms to org.sparkproject.jetty.util.log.Slf4jLog
3/06/01 01:07:25 INFO org.sparkproject.jetty.server.Server: jetty-9.4.40.v20210413; built: 2021-04-13T20:42:42.668Z; git: b881a572662e1943a14ae12e7e1207989f218b74; jvm 1.8.0_372-b07
3/06/01 01:07:25 INFO org.sparkproject.jetty.server.Server: Started @6934ms
3/06/01 01:07:25 INFO org.sparkproject.jetty.server.AbstractConnector: Started ServerConnector@2104b36{HTTP/1.1, (http/1.1)}{0.0.0.0:34107}
3/06/01 01:07:27 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at shbhat-dplab-m/10.138.0.23:8032
3/06/01 01:07:27 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at shbhat-dplab-m/10.138.0.23:10200
3/06/01 01:07:28 INFO org.apache.hadoop.conf.Configuration: resource-types.xml not found
3/06/01 01:07:28 INFO org.apache.hadoop.yarn.util.resource.ResourceUtils: Unable to find 'resource-types.xml'.
3/06/01 01:07:31 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1685581441543_0001
3/06/01 01:07:32 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at shbhat-dplab-m/10.138.0.23:8030
3/06/01 01:07:36 INFO com.google.cloud.hadoop.repackaged.gcs.com.google.cloud.hadoop.gcsio.GoogleCloudStorageImpl: Ignoring exception of type GoogleJsonResponseException; verified object a
ready exists with desired state.
i is roughly 3.1415517914155178
3/06/01 01:07:59 INFO org.sparkproject.jetty.server.AbstractConnector: Stopped Spark2104b36{HTTP/1.1, (http/1.1)}{0.0.0.0:0}
env) shbhat@cloudshell:~ (cloud-shrikrishna-shbhat)$

9.4g Dataflow

Card 3 Beam Code

9.4.1 Answer the questions

- **Where is the input taken from by default?**

  By default, the input for the script is taken from the '../javahelp/src/main/java/com/google/cloud/training/dataanalyst/javahelp/' directory. The input files should have a '.java' extension.

- **Where does the output go by default?**

  The output, by default, will be written to the '/tmp/output' directory with a filename generated by Beam.

- **Examine both the `getPackages()` function and the `splitPackageName()` function. What operation does the '`PackageUse()`' transform implement?**

  The 'getPackages()' function takes a line of code and a keyword ('import' in this case) and extracts the Java package names from that line. It uses the 'splitPackageName()' function to split a package name into its individual components. The 'PackageUse()' transform takes a line of code and a keyword and yields a tuple of each package used and a count of 1.

- **Look up Beam's `CombinePerKey`. What operation does the `TotalUse` operation implement?**

  The 'TotalUse' operation, using the 'CombinePerKey' transform, implements the "Reduce" operation. It combines the counts for each package and sums them up based on the package key.

9.4.2 Answer the questions

- **Which operations correspond to a "Map"?**

  Operations that correspond to a "Map" in the pipeline are:

  'GetImports': Extracts lines starting with the 'import' keyword.

  'PackageUse': Extracts the packages from each line and yields them with a count of 1.

- **Which operation corresponds to a "Shuffle-Reduce"?**

  The operation that corresponds to a "Shuffle-Reduce" is:

  'TotalUse': Performs the shuffling of key-value pairs based on the package key and reduces the values by summing them up.

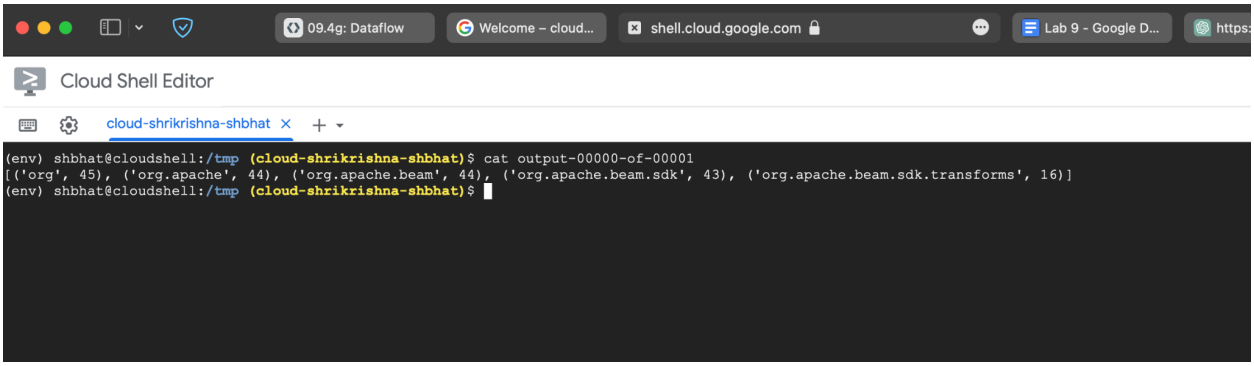
- **Which operation corresponds to a "Reduce"?**

  The operation that corresponds to a "Reduce" is:

  'TotalUse': It combines the counts for each package and sums them up based on the package key, reducing the values to a single count for each package.

Card 4 - Run pipe locally

9.4.3 Take a screenshot of its contents



9.4.4 Explain what the data in this output file corresponds to based on your understanding of the program.

Based on the program's logic, the output file will contain the top 5 most used Java packages in the input Java code files. Each line in the output file will consist of a package name and its corresponding count, separated by a tab or space.

The program scans the input Java code files, identifies the lines that start with the keyword 'import', and extracts the package names from those lines. It then counts the occurrences of each package and combines the counts using the 'CombinePerKey' transform.

The 'Top_5' transform selects the top 5 packages with the highest counts, based on the count values. Finally, the selected packages and their counts are written to the output file.

Therefore, the output file will provide insights into the most commonly used Java packages in the analyzed codebase, allowing developers to identify frequently utilized dependencies and potentially optimize their code or manage dependencies more effectively.

Card 5 – Dataflow Lab #2 (Word Count)

9.4.5 What are the names of the stages in the pipeline?

Answer is in the below question.

9.4.6 Describe what each stage does.

Both answers are written below where stages and explaination is given

In the given code, the stages in the pipeline are as follows:

Read: Reads the text file specified in the input argument using ReadFromText. Produces a PCollection named lines containing the lines of the input text file.

Split: Applies the WordExtractingDoFn function using ParDo. Splits each line of text into individual words using regular expression matching.

Produces a PCollection of words. PairWithOne: Maps each word to a key-value pair where the word is the key and the value is 1. Assigns an initial count of 1 to each word.

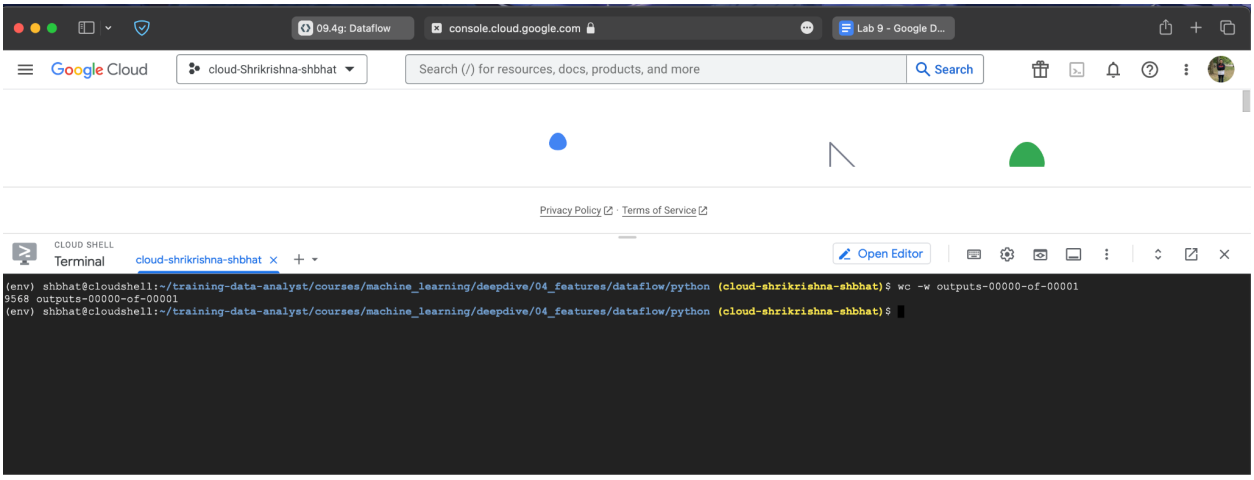
GroupAndSum: Groups the key-value pairs by key (word). Combines the values (counts) for each key (word) using the CombinePerKey transform with the sum function. Produces a PCollection of key-value pairs where the key is a unique word and the value is the sum of counts for that word.

Format: Maps each key-value pair to a formatted string using the format_result function. Formats the word and its count as "%s: %d" where %s is the word and %d is the count. Produces a PCollection of formatted strings.

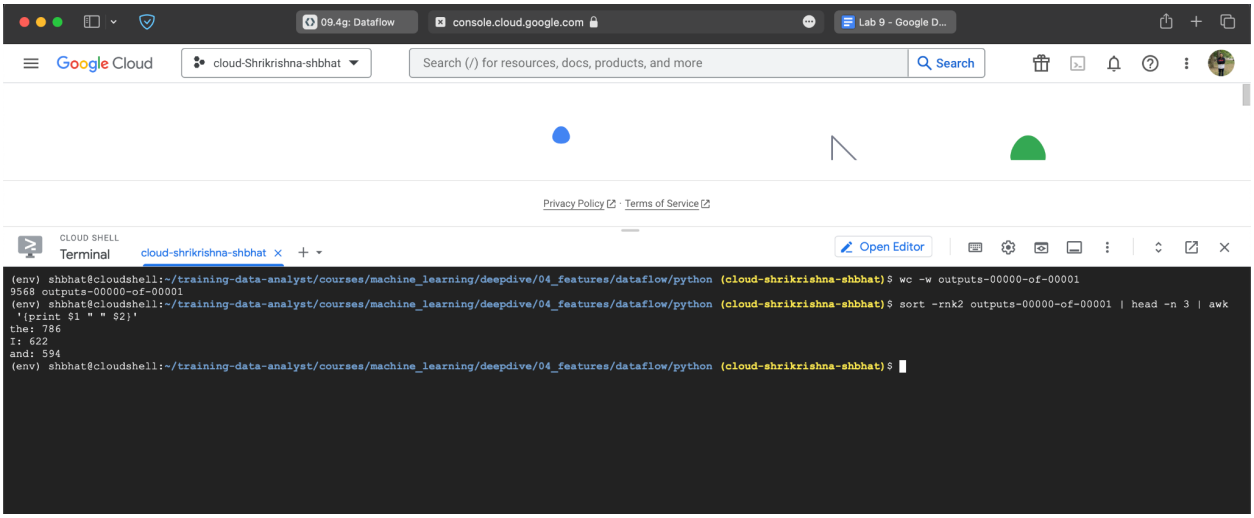
Write: Writes the output PCollection to the output file specified in the --output argument using the WriteToText transform. Each stage in the pipeline performs a specific transformation or action on the data, such as reading from a text file, splitting lines into words, counting the occurrences of each word, formatting the results, and writing the output to a file

Card 6 Run Code locally

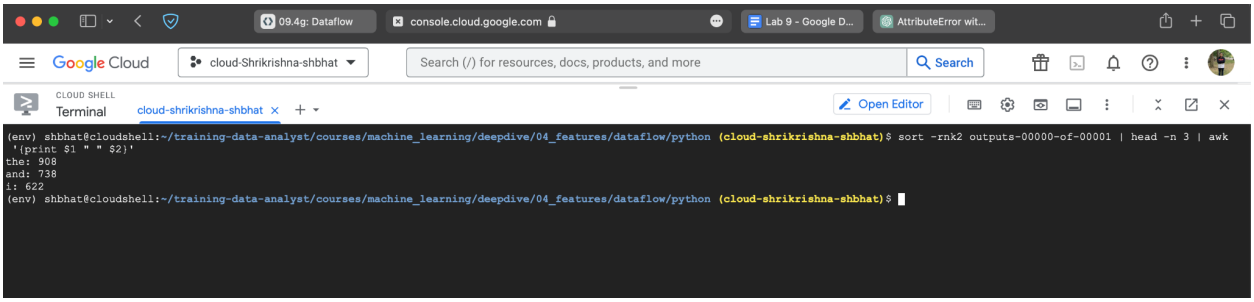
9.4.7 Use `wc` with an appropriate flag to determine the number of unique words in King Lear.



9.4.8 Use sort with appropriate flags to perform a *numeric* sort on the *key field* containing the count for each word in *descending* order. Pipe the output into `head` to show the top 3 words in King Lear and the number of times they appear

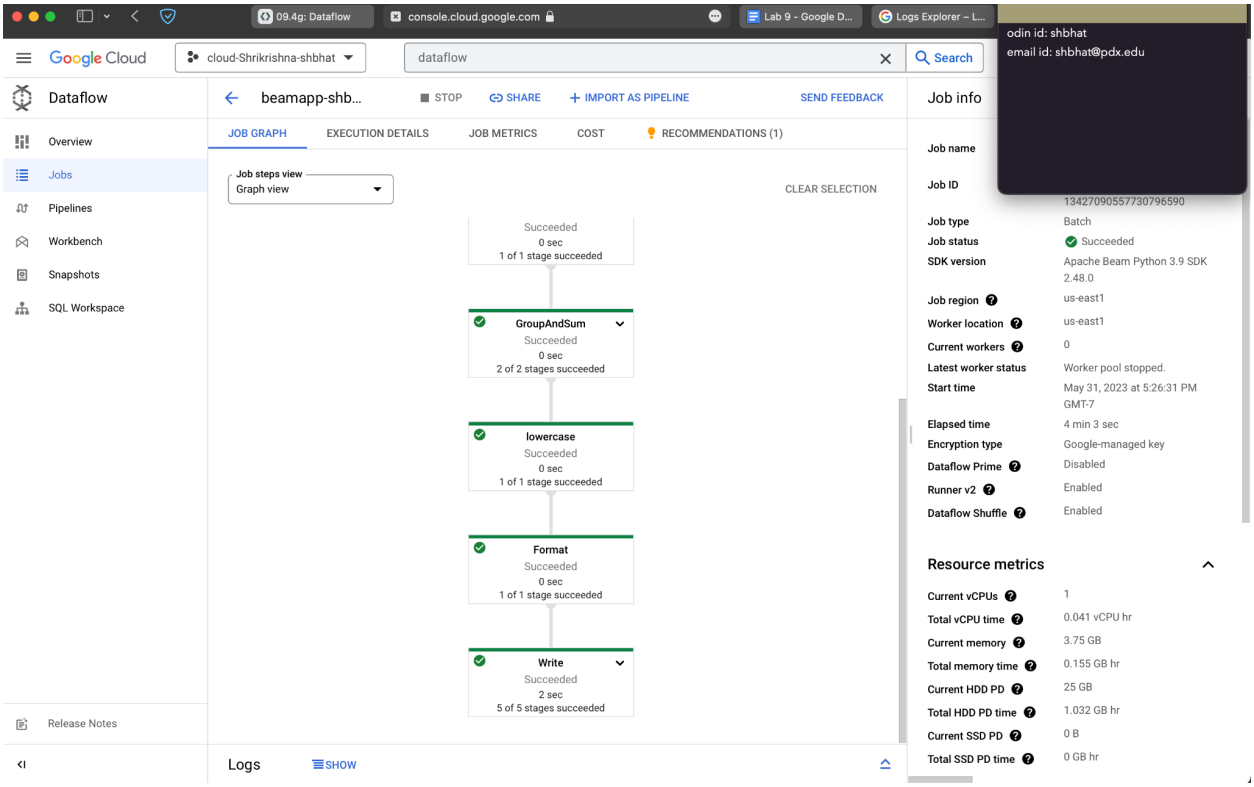


9.4.9 Use the previous method to show the top 3 words in King Lear, case-insensitive, and the number of times they appear.

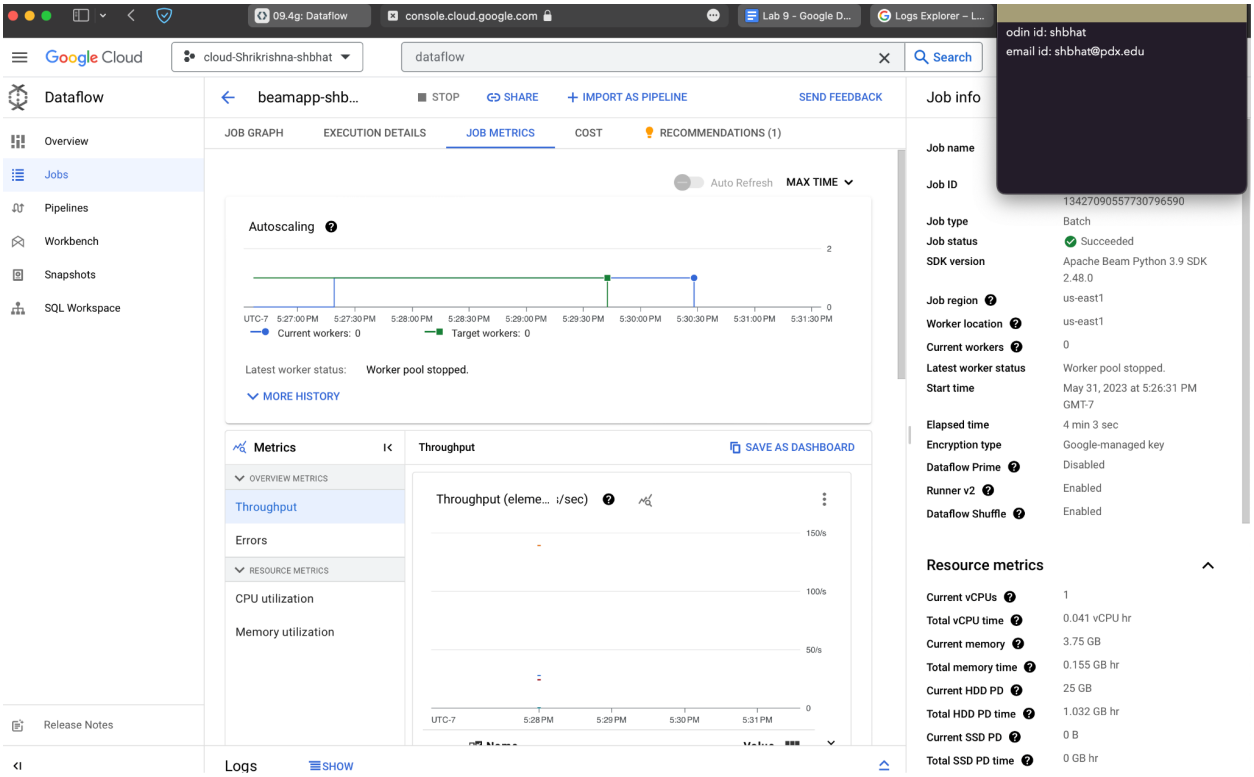


Card 9 Run Code using Dataflow runner

9.4.10 The part of the job graph that has taken the longest time to complete.



9.4.11 The autoscaling graph showing when the worker was created and stopped.

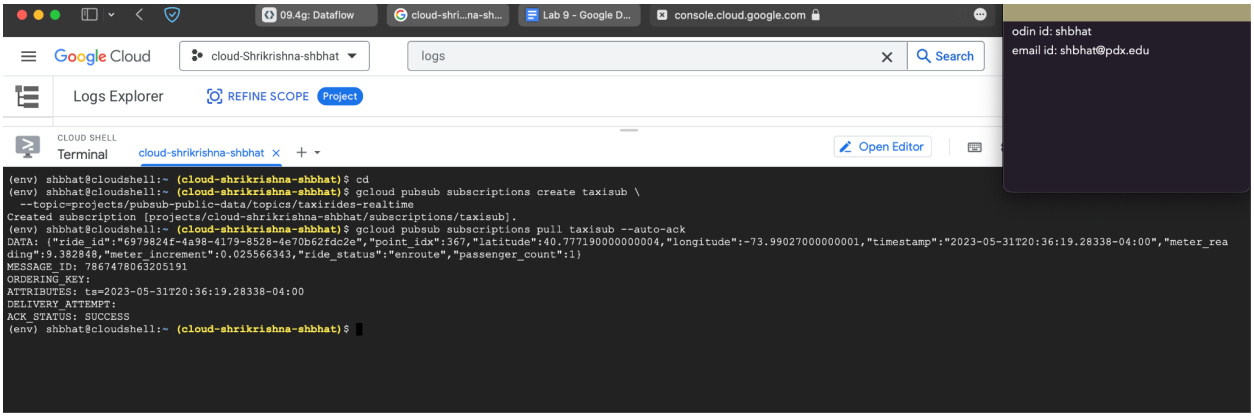


9.4.12 Examine the output directory in Cloud Storage. How many files has the final write stage in the pipeline created?

Final write stage has created 6 files.

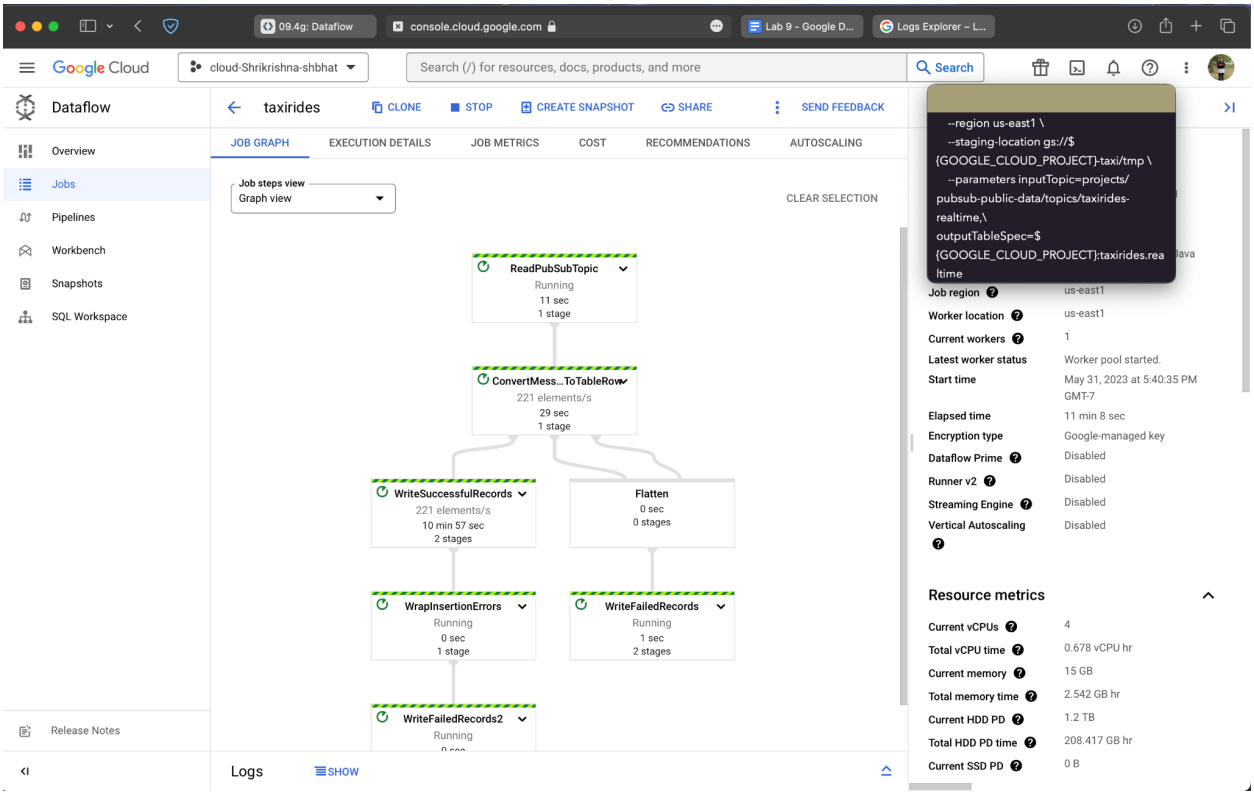
Card 12 View raw data from PubSub

9.4.13 Take a screenshot listing the different fields of this object.



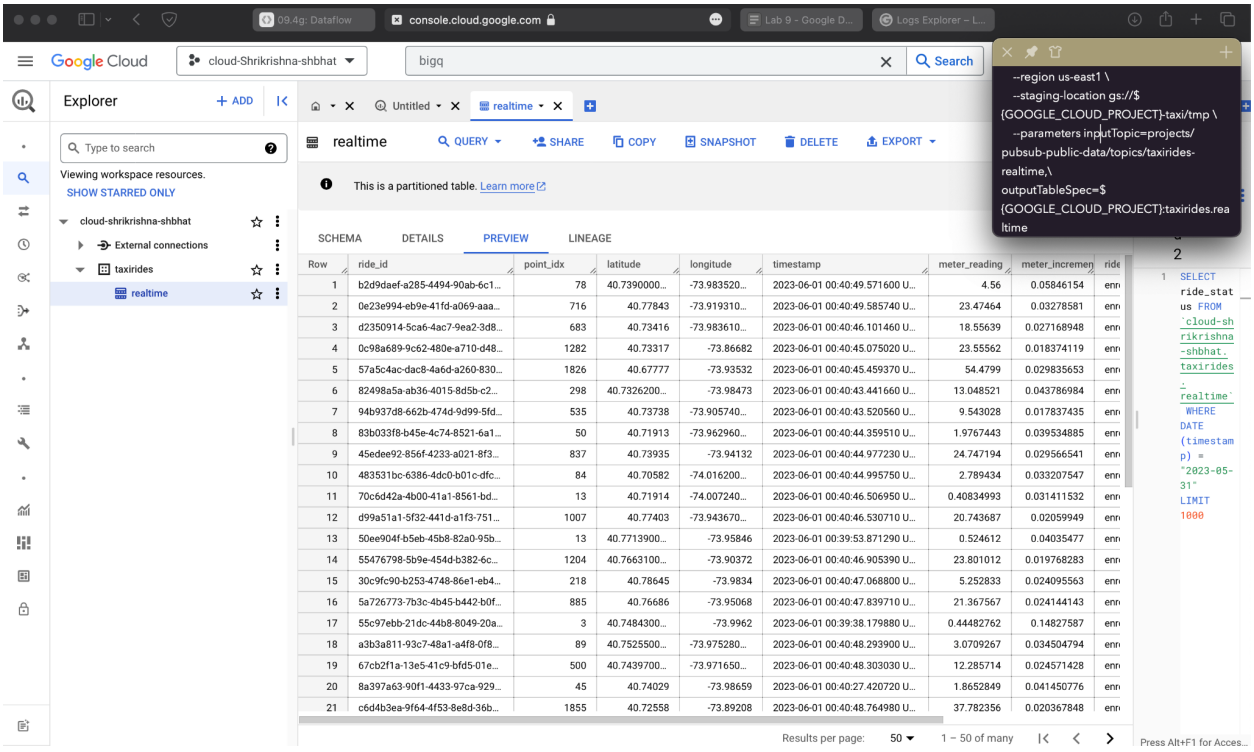
Card 14 Run Dataflow job from template

9.4.14 Take a screenshot of the pipeline that includes its stages and the number of elements per second being handled by individual stages.

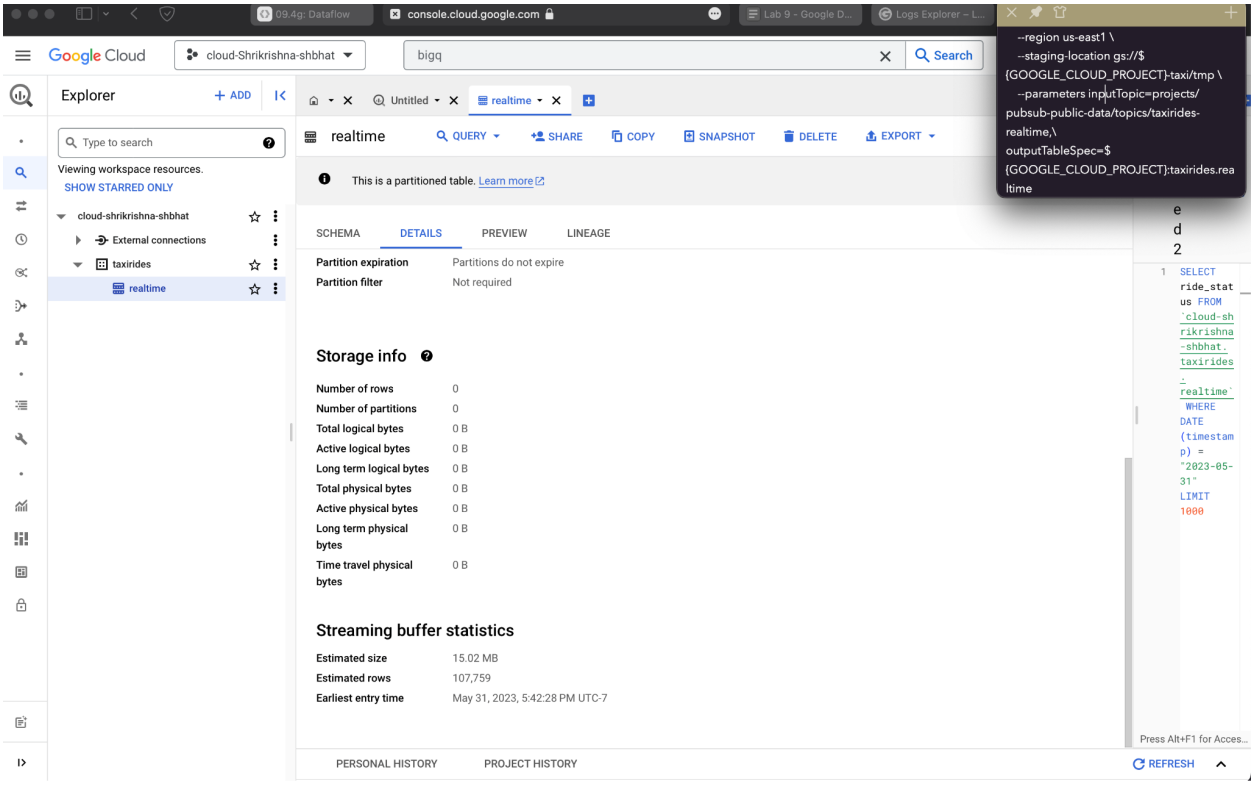


Card 15 Query data in BigQuery

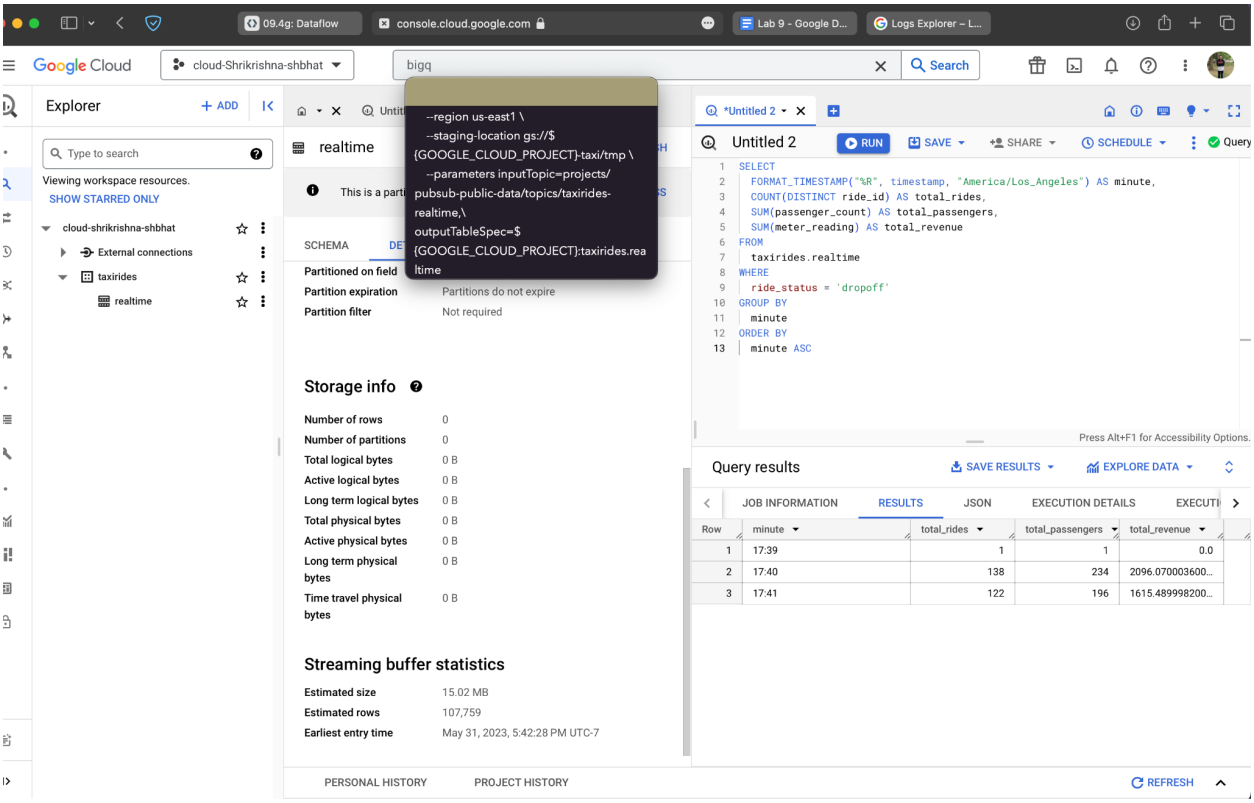
9.4.15 Take a screenshot showing the number of passengers and the amount paid for the first ride



9.4.16Take a screenshot showing the estimated number of rows in the table.



9.4.17 Take a screenshot showing the per-minute number of rides, passengers, and revenue for the data collected.



Card 16 – Data visualization

9.4.18Take a screenshot showing the plot for your data for your lab notebook