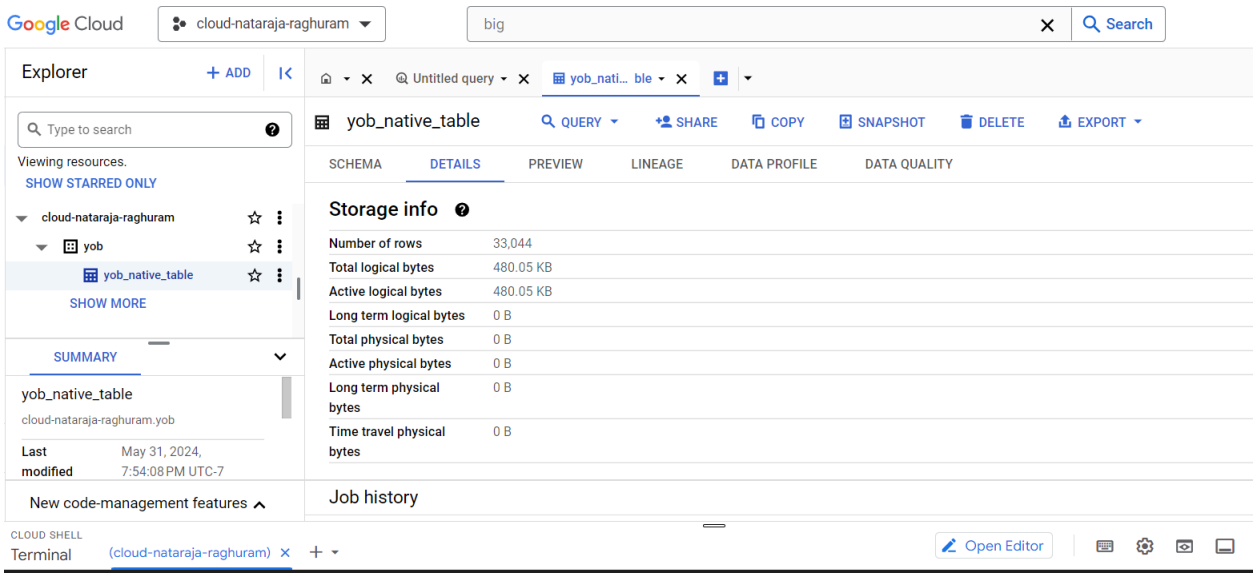
09.1g: BigQuery, BigLake

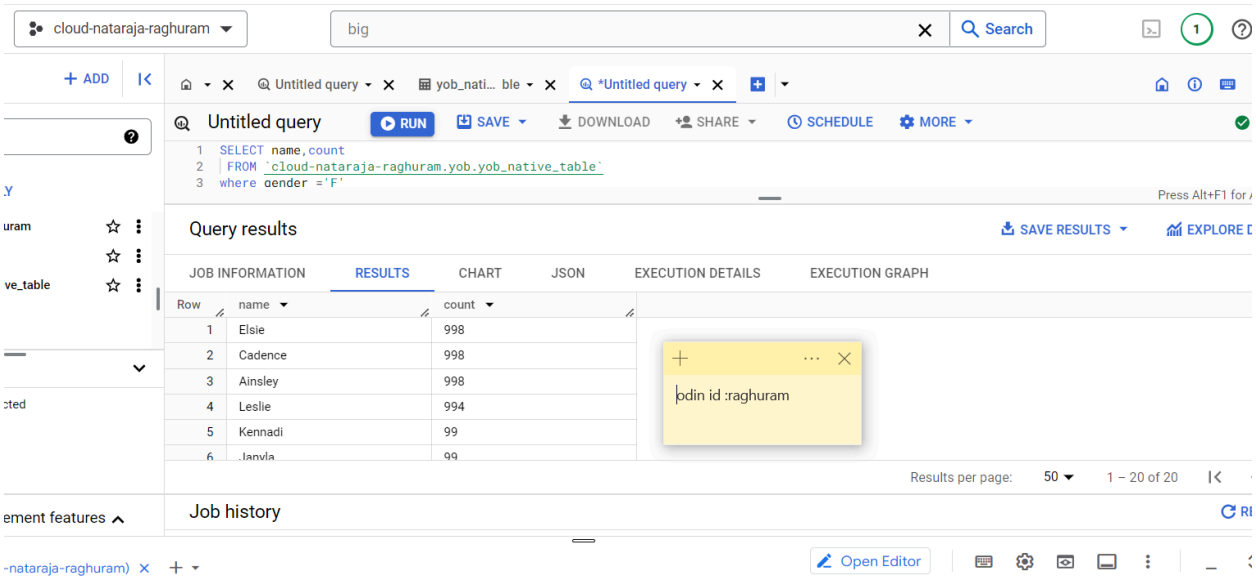
# 3. Create dataset

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




# 4. Query data

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




- Screenshot your results and include it in your lab notebook

```
raghuram@cloudshell:~ (cloud-nataraja-raghuram)$ bq query "SELECT name,count
 FROM [cloud-nataraja-raghuram.yob.yob_native_table]
where gender ='M'
ORDER BY count ASC
LIMIT 20"
+-----------+-------+
|   name    | count |
+-----------+-------+
| Alexx     | 10    |
| Airen     | 10    |
| Aasir     | 10    |
| Alyjah    | 10    |
| Aldric    | 10    |
| Abdulahad | 10    |
| Abubacarr | 10    |
| Alika     | 10    |
| Aarron    | 10    |
| Alontae   | 10    |
| Amarian   | 10    |
| Agrim     | 10    |
| Amara     | 10    |
| Aison     | 10    |
| Airam     | 10    |
| Adlai     | 10    |
| Alter     | 10    |
| Aizik     | 10    |
| Albaraa   | 10    |
| Aedin     | 10    |
+-----------+-------+
```

- Screenshot your results and include it in your lab notebook

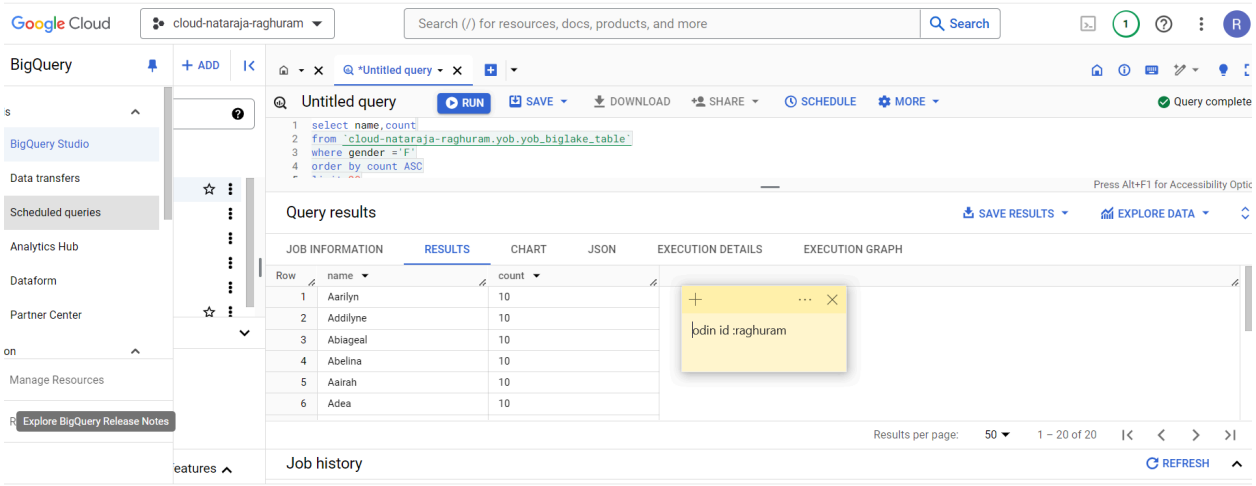
```
cloud-nataraja-raghuram> SELECT name,count  FROM [cloud-nataraja-raghuram.yob.yob_native_table] where gender ='M' ORDER BY count ASC LIMIT 10
+-----------+-------+
|   name    | count |
+-----------+-------+
| Aedin     | 10    |
| Abubacarr | 10    |
| Aasir     | 10    |
| Airen     | 10    |
| Agrim     | 10    |
| Abdulahad | 10    |
| Aarron    | 10    |
| Aison     | 10    |
| Airam     | 10    |
| Adlai     | 10    |
+-----------+-------+
```

- Screenshot your results and include it in your lab notebook

```
cloud-nataraja-raghuram> SELECT name,count  FROM [cloud-nataraja-raghuram.yob.yob_native_table] where name='Raghuram'
cloud-nataraja-raghuram>
```

# 9. Query data

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



09.2g: Jupyter Notebooks

# 3. BigQuery query

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

  **This query will process 3.05 GB when run. That is almost 18 gb of less data**

- How many twins were born during this time range?

  **375362**

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

  **On average, single babies (plurality 1) weigh approximately 2.17116 units more than twins (plurality 2)**

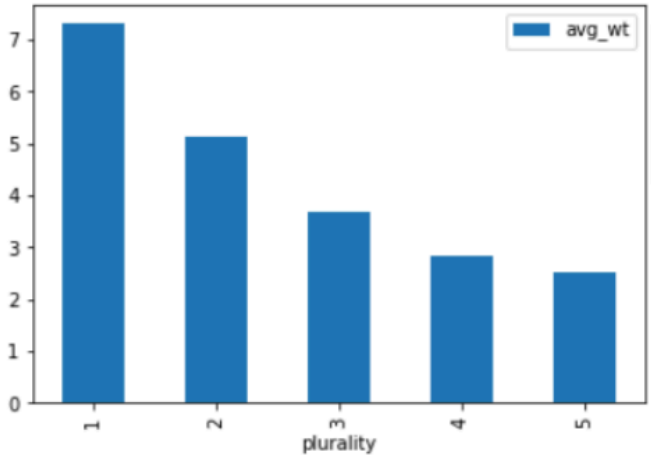
# 6. Run queries

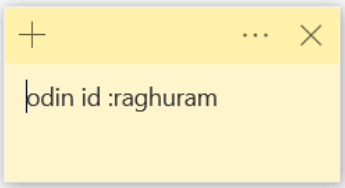
- Show the plots generated for the two most important features for your lab notebook

  **Plurality and gestation weeks**

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

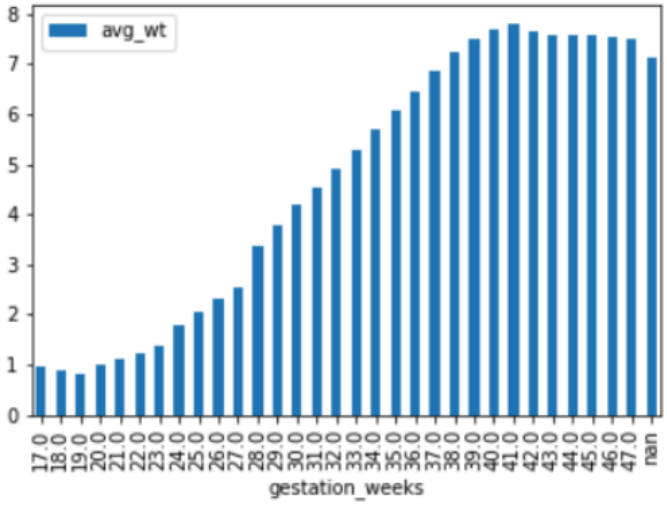
```
[9]: <matplotlib.axes._subplots.AxesSubplot at 0x7f492262e690>
```

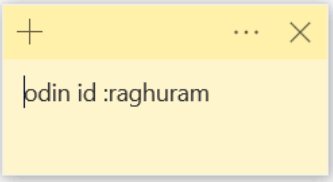



```
[11]: df = get_distinct_values('gestation_weeks')
      df.plot(x='gestation_weeks', y='avg_wt', kind='bar')
```

```
[11]: <matplotlib.axes._subplots.AxesSubplot at 0x7f4921971350>
```




## 8. Mobility

- What day saw the largest spike in trips to grocery and pharmacy stores?

**2020-03-13**

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

  **-49**

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

| | | |
|---|---|---|
| 1 | Detroit Metropolitan Wayne County | 45.416666666664 |
| 2 | McCarran International | 45.600000000009 |
| 3 | San Francisco International | 47.2666666666 |

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

| | | |
|---|---|---|
| 1 | McCarran International | 40.933333333333337 |
| 2 | Detroit Metropolitan Wayne County | 46.133333333333334 |
| 3 | San Francisco International | 51.333333333333 |

## 10. Mortality

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

  **Table:excess_deaths**

  **Columns:placename, start_date, excess_deaths**

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

  **Tables:us_counties**

  **Columns: date, county, deaths**

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

  **Tables: us_states**

  **Columns: date, state_name, confirmed_cases**

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

  **Tables:mask_use_by_county**

  **Columns: county_fips_code, always**

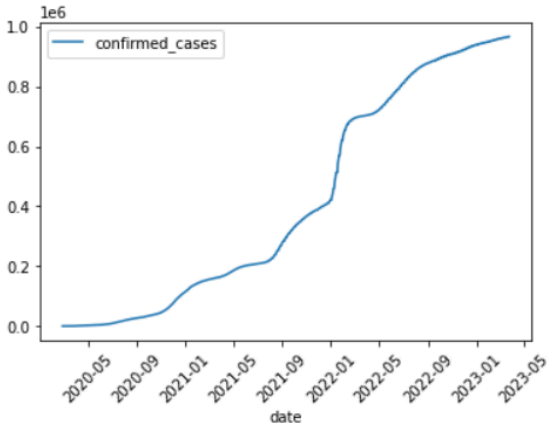
## 11. Run example queries

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

```
[35]: def get_distinct_values(column_name):
        query_string = """
      SELECT date, confirmed_cases
      FROM `bigquery-public-data.covid19_nyt.us_states`
      WHERE state_name = 'Oregon'
      ORDER BY date ASC
        """
        return bigquery.Client().query(query_string).to_dataframe().sort_values(column_name)
      df = get_distinct_values('confirmed_cases')

[36]: df.plot(x='date', y='confirmed_cases', kind='line', rot=45)

[36]: <matplotlib.axes._subplots.AxesSubplot at 0x7f4920704250>
```



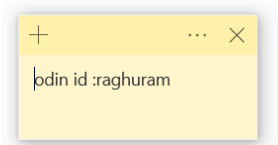

odin id :raghuram

- 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.

[39]:
```
query_string = """
SELECT state_name, MIN(date) as date_of_1000
FROM `bigquery-public-data.covid19_nyt.us_states`
WHERE deaths > 1000
GROUP BY state_name
ORDER BY date_of_1000 ASC
"""
df = bigquery.Client().query(query_string).to_dataframe()
df.head(10)
```

[39]:

| | state_name | date_of_1000 |
|---|---|---|
| 0 | New York | 2020-03-29 |
| 1 | New Jersey | 2020-04-06 |
| 2 | Michigan | 2020-04-09 |
| 3 | Louisiana | 2020-04-14 |
| 4 | Massachusetts | 2020-04-15 |
| 5 | Illinois | 2020-04-16 |
| 6 | California | 2020-04-17 |
| 7 | Connecticut | 2020-04-17 |
| 8 | Pennsylvania | 2020-04-17 |
| 9 | Florida | 2020-04-24 |

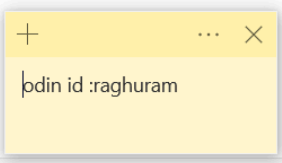



- Take a screenshot for your lab notebook of the Top 5 counties and the states they are located in.

[41]:
```
query_string = """
SELECT DISTINCT mu.county_fips_code, mu.always, ct.county
FROM `bigquery-public-data.covid19_nyt.mask_use_by_county` as mu
LEFT JOIN `bigquery-public-data.covid19_nyt.us_counties` as ct
ON mu.county_fips_code = ct.county_fips_code
ORDER BY mu.always DESC
"""
df = bigquery.Client().query(query_string).to_dataframe()
df.head(5)
```

[41]:

| | county_fips_code | always | county |
|---|---|---|---|
| 0 | 06027 | 0.889 | Inyo |
| 1 | 36123 | 0.884 | Yates |
| 2 | 06051 | 0.880 | Mono |
| 3 | 48229 | 0.880 | Hudspeth |
| 4 | 48141 | 0.877 | El Paso |



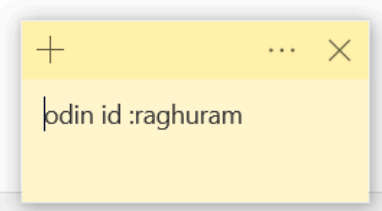

## 12. Write queries

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

```
query_string = """
SELECT date, deaths, county
FROM `bigquery-public-data.covid19_nyt.us_counties`
WHERE county = 'Multnomah'
ORDER BY date ASC

"""
df = bigquery.Client().query(query_string).to_dataframe()
df.head(5)
```

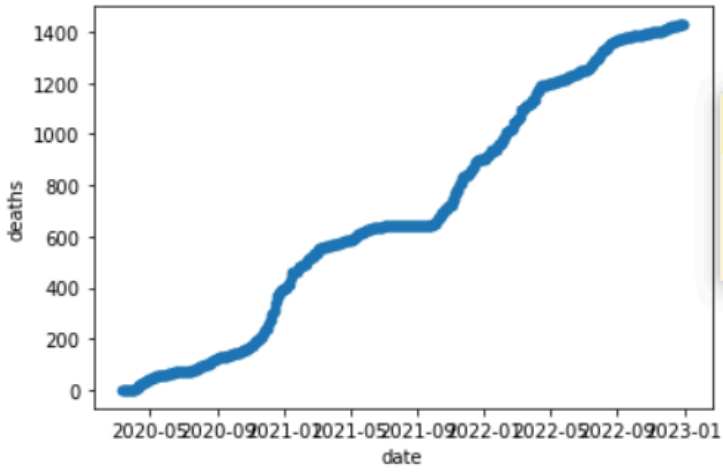
| | date | deaths | county |
|---|---|---|---|
| **0** | 2020-03-10 | 0 | Multnomah |
| **1** | 2020-03-11 | 0 | Multnomah |
| **2** | 2020-03-12 | 0 | Multnomah |
| **3** | 2020-03-13 | 0 | Multnomah |
| **4** | 2020-03-14 | 1 | Multnomah |

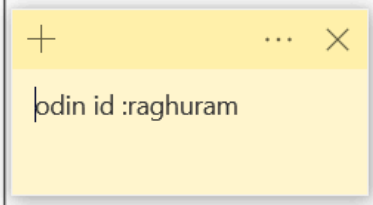



```
df.plot(x='date', y='deaths' , kind='scatter')
```

```
[44]: df.plot(x='date', y='deaths' , kind='scatter')
```

```
[44]: <matplotlib.axes._subplots.AxesSubplot at 0x7f4920865c10>
```



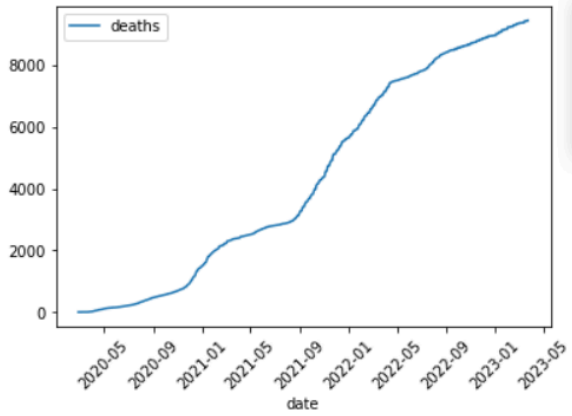

- Plot the results and take a screenshot for your lab notebook.

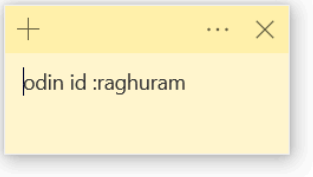
[49]:
```
query_string = """
SELECT date, deaths
FROM `bigquery-public-data.covid19_nyt.us_states`
WHERE state_name = 'Oregon'
ORDER BY date ASC

"""
df = bigquery.Client().query(query_string).to_dataframe()

df.plot(x='date',y='deaths',kind='line',rot=45)
```

[49]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f490af9b8d0>
```




### 09.3g: Dataproc

## 6. Run computation

- How long did the job take to execute?

  **Approximately 25 seconds**

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

  **Pi is roughly 3.1415509514155096**

## 8. Run computation again

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

  **17 seconds , 8 seconds approximately faster**

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

  **Pi is roughly 3.1416372314163725**

# 09.4g: Dataflow

## 3. Beam code

- Where is the input taken from by default?

  **parser.add_argument('--input',**
  **default='../javahelp/src/main/java/com/google/cloud/training/dataanalyst/javahelp/',**
  **help='Input directory')**

- Where does the output go by default?

  **/tmp/output**

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

  The packageUse() transform extracts package names from lines containing the specified keyword (import), splits them into components, and yields each component with a count of 1 for aggregation.

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

Beam's CombinePerKey is used to combine the values for each key in a collection of key-value pairs. The TotalUse operation implements a summation of counts for each package name, effectively aggregating the total occurrences of each package across all input lines.

- Which operations correspond to a "Map"?

  **GetImports , PackageUse**

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

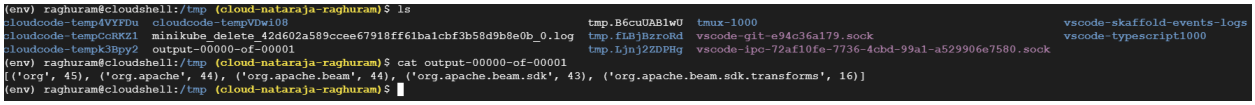
  **TotalUse' > beam.CombinePerKey(sum)**

- Which operation corresponds to a "Reduce"?

  **'TotalUse' >> beam.CombinePerKey(sum)**

# 4. Run pipeline locally

- Take a screenshot of its contents




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

  **The data in the output file corresponds to the top 5 most frequently used Java package prefixes within the analyzed Java files. Each tuple in the output consists of a package prefix and its corresponding usage count.**

# 5. Dataflow Lab #2 (Word count)

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

  **Read Stage, Split Stage, PairWithOne Stage, GroupAndSum Stage, Format Stage, Write Stage**

- Describe what each stage does.

  **Read Stage: Reads the input text file into a PCollection.**

  **Split Stage: Splits each line into individual words using a regular expression.**

  **PairWithOne Stage: Maps each word to a key-value pair, where the key is the word, and the value is 1.**

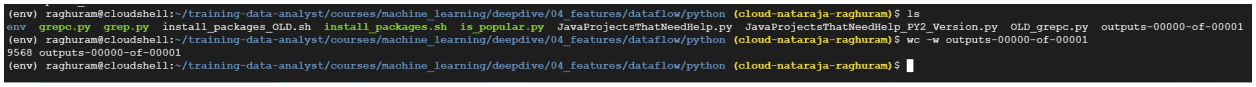
**GroupAndSum Stage: Groups the key-value pairs by the word and adds the values for each key.**

**Format Stage: Formats the word count results into strings.**

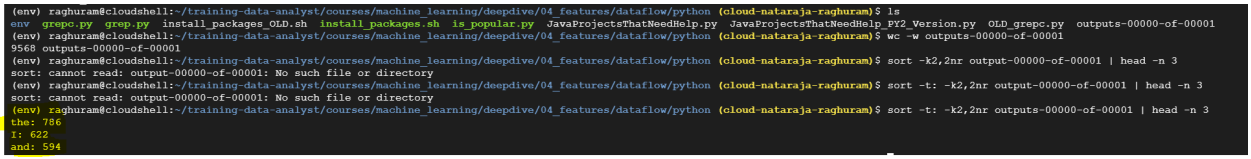
**Write Stage: Writes the formatted word count to an output text file.**

# 6. Run code locally

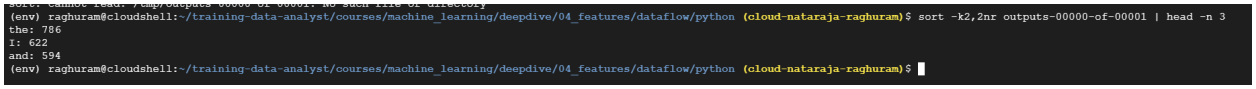
- Use wc with an appropriate flag to determine the number of different words in King Lear.




- 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




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

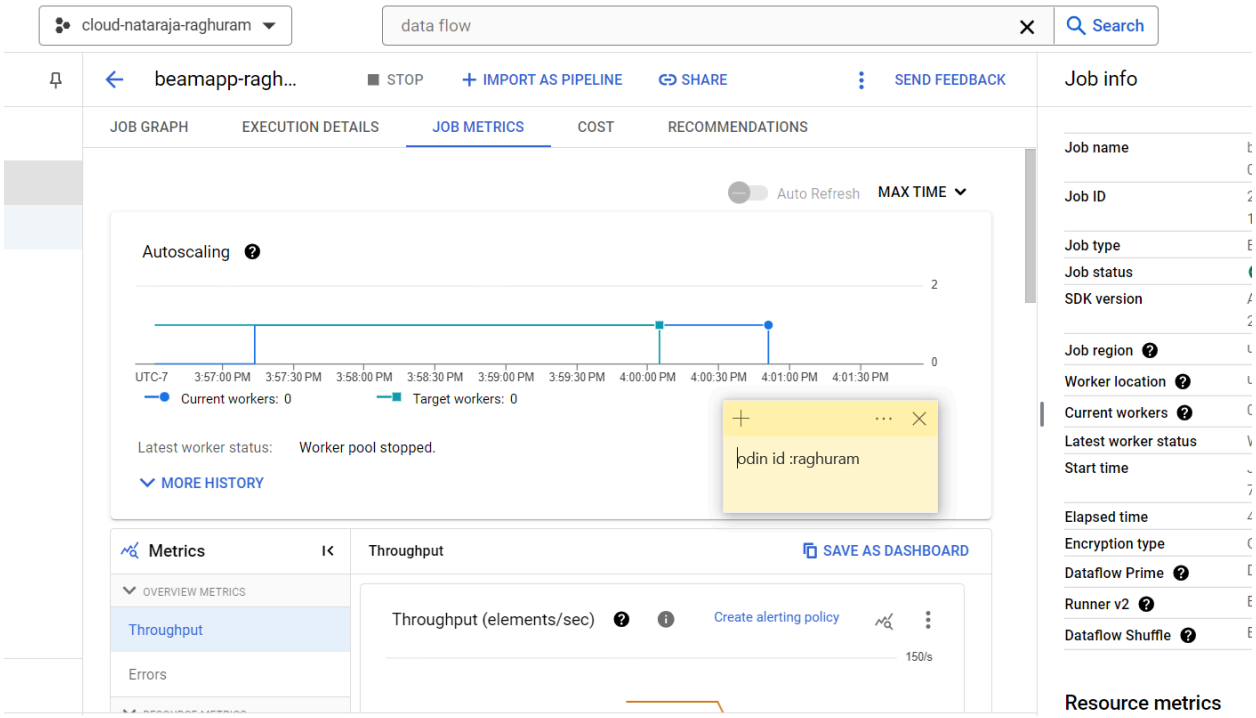



# 9. Run code using Dataflow runner

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

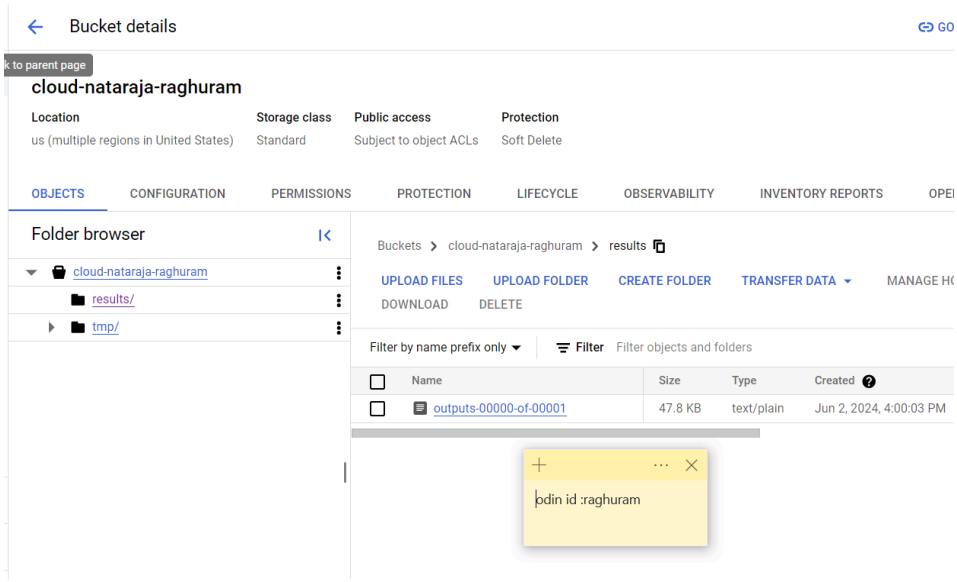
**Write succeeded it took 2 seconds**

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



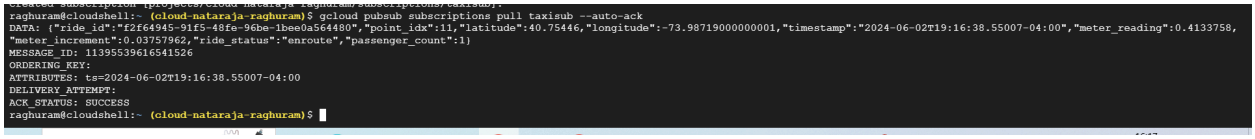

- Examine the output directory in Cloud Storage. How many files has the final write stage in the pipeline created?

  One file



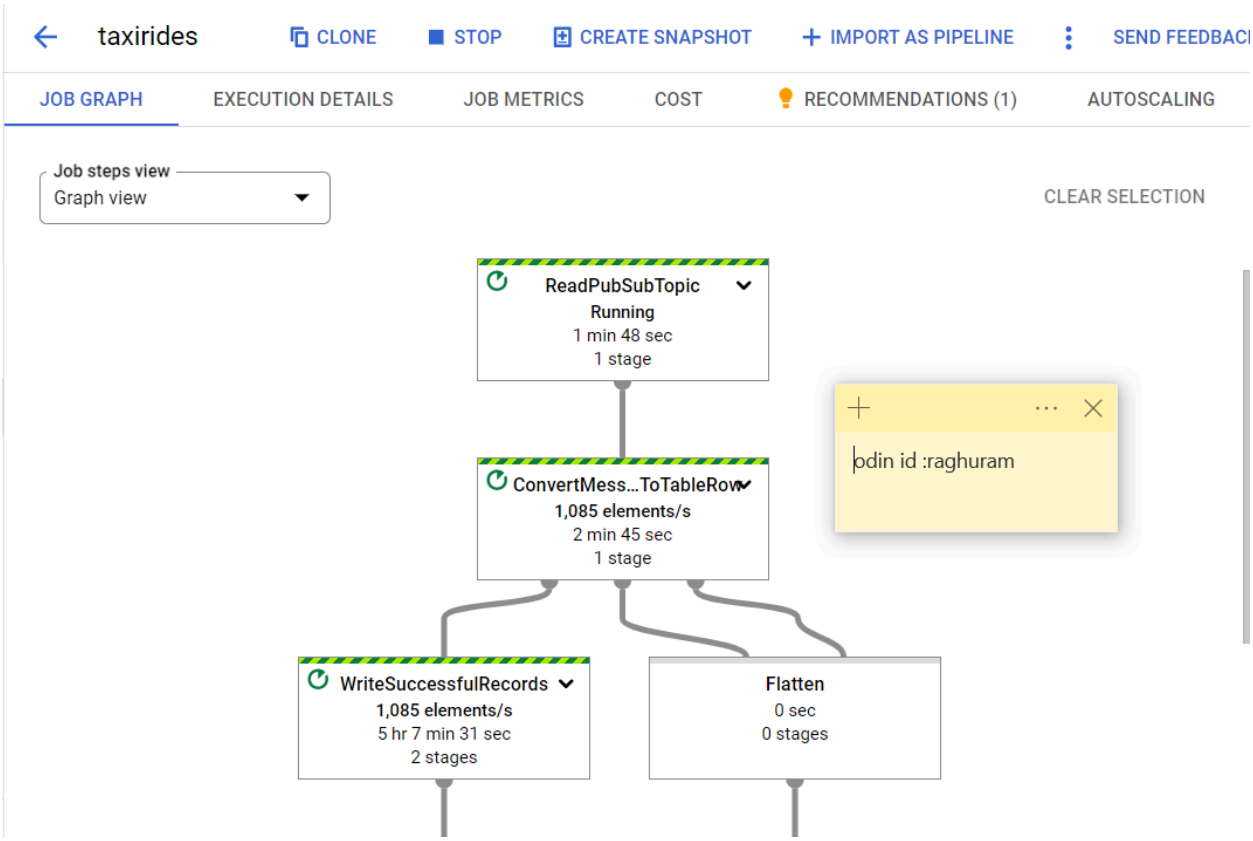
## 12. View raw data from PubSub

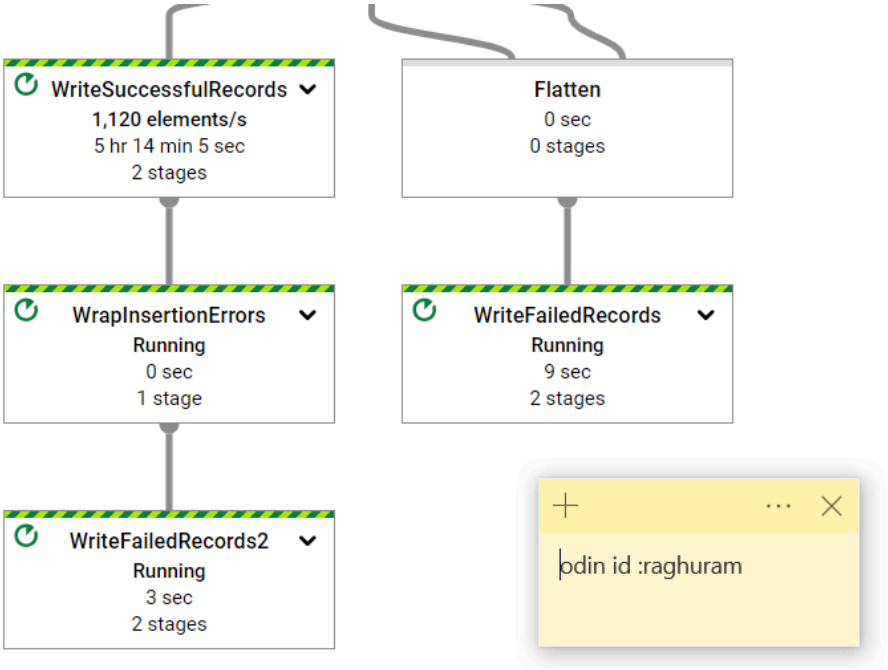
- Take a screenshot listing the different fields of this object.




## 14. Run Dataflow job from template

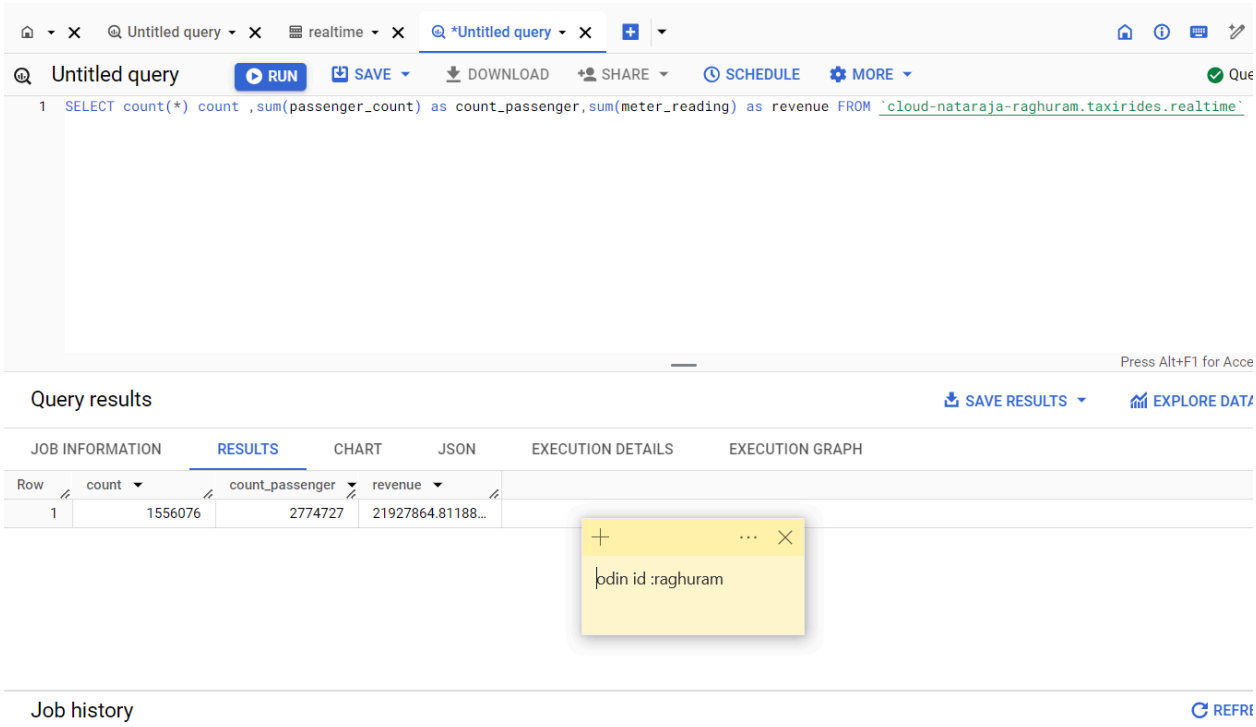
- Take a screenshot of the pipeline that includes its stages and the number of elements per second being handled by individual stages.



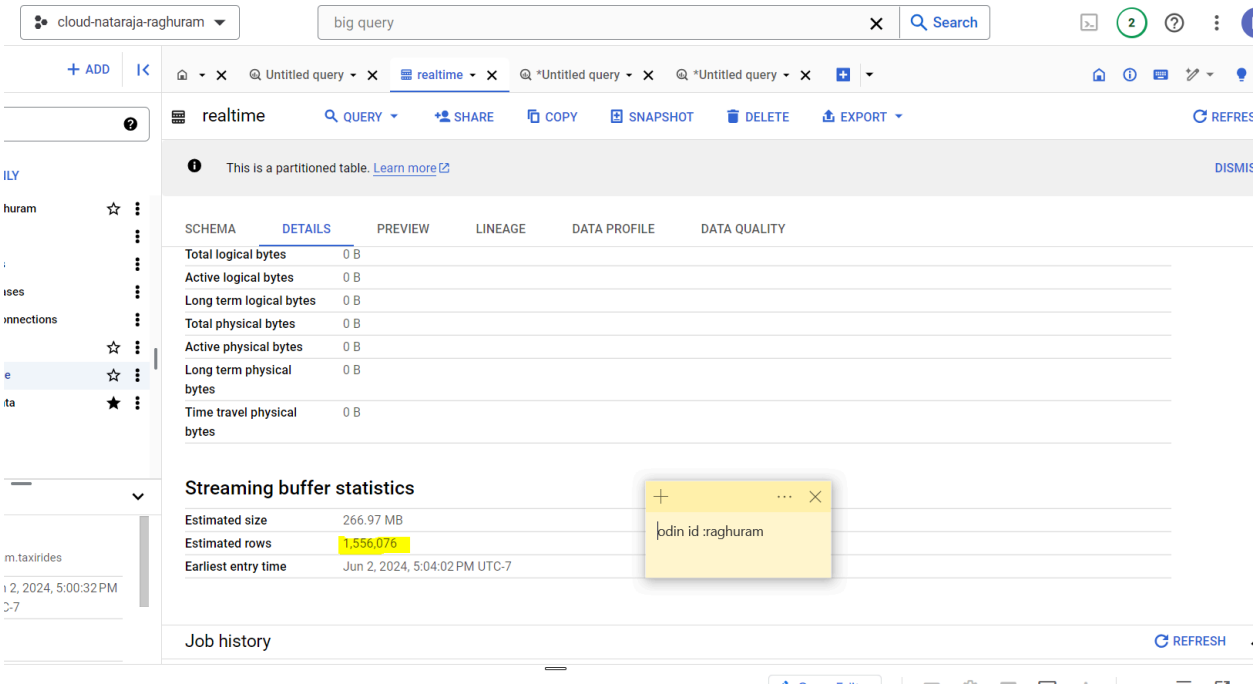



## 15. Query data in BigQuery

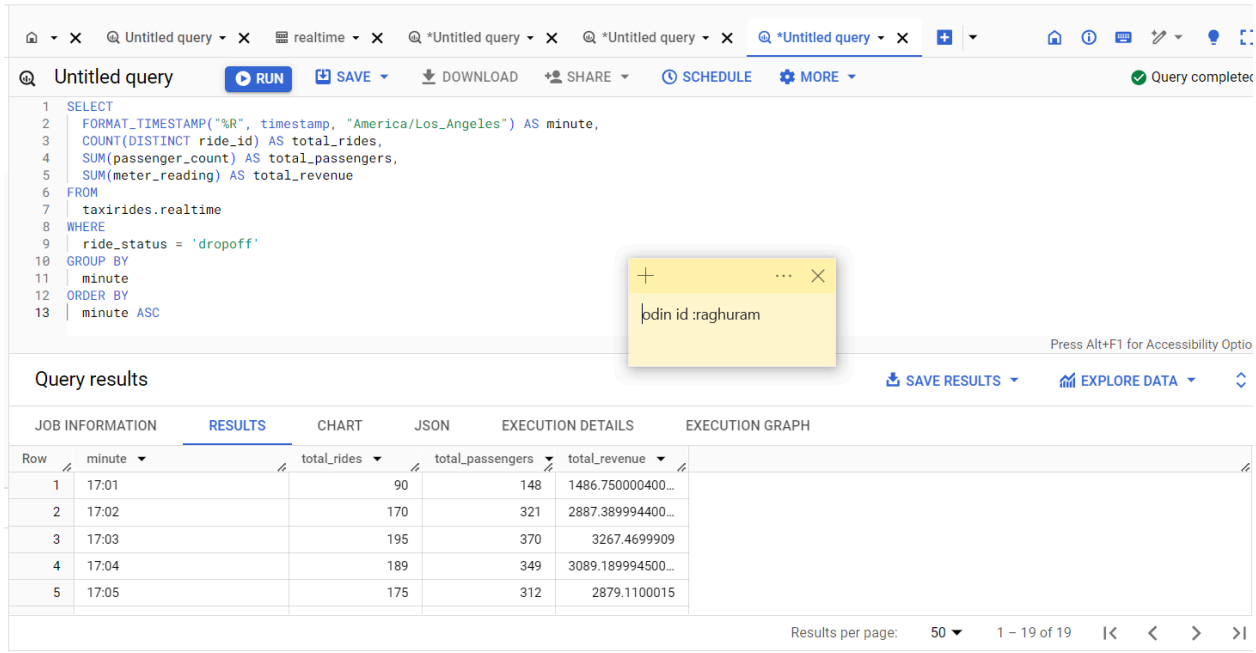
- Take a screenshot showing the number of passengers and the amount paid for the first ride




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




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



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