3/17/2025

CS 512
Final Project

Kanavikar, Disha Basavaraja
OREGON STATE UNIVERSITY
KANAVIKD@OREGONSTATE.EDU

Title

Final Project : Yelp database

Dataset description

Yelp data was sourced from the assignment page on Canvas, although it is publicly open. The original data was in JSON format and was derived from a subset of data from Yelp that includes businesses, reviews, and user data from 11 metropolitan areas. Each file is composed of a single object type, one JSON-object per-line.
Examples include business names, ratings, and location. All the data was obtained through six JSON files: business.json, checkin.json, review.json, tip.json, and user.json.
https://business.yelp.com/data/resources/open-dataset/

Points System –

- Data is not provided in a standardized format: 3pts
  - Data was provided in Json format but it required a lot of formatting : 2 pts
- Data is split up across multiple files to begin with: 1pt
  - Yes data was split across multiple files: 1pt
- Data is in a format other than one of the following: 2pts
  - Data was provided in JSON format but was irregular: 1 pt
- Data contains strings with punctuation (quotes or commas): 1pt
  - Yes, data contained strings with punctuations: 1 pt
- Data set is larger than 1GB in size: 1pt
  - Business.json file was 120 MB and reviews.json was 5.4 GB provided: 1 pt
- Data set is composed of more than one type of related data: 2pts
  - Business.json file contains business data which includes location data, and categories and reviews.json contains full review text data including the user_id that wrote the review and the business_id the review is written for : 2 pts
- Data set needs to be accessed in a way other than connecting to a database or downloading a file: 1pt
  - Data was accessed by downloading a zip file : 0pts

Total data complexity points: 9pts

I chose this dataset because it includes data about businesses and their reviews, which gives me a clear picture of both the businesses and how their customers feel. I found the dataset to be very versatile, as it covers different types of businesses in various locations, making it useful for solving many modern data problems. It can help with customer sentiment analysis, where I can better understand how customers feel about a business. Additionally, this dataset can provide insights into business performance, customer preferences, and trends in different areas. By analyzing it, businesses can improve customer satisfaction, adjust their services, and make smarter decisions. It's also great for exploring topics like trends, customer feedback, and market needs, which are all important for businesses to stay competitive and meet customer expectations

Problem Statement

The objective of this project is to wrangle, preprocess data and integrating it with Google cloud platform services for further analysis. Through a detailed examination of the dataset, I have identified several business questions that can provide actionable insights for decision-making and strategy development.

1. **Profitability Analysis by Business Category:**
    - Which business category is the most profitable in different states across the U.S.?
    - Which business category is the least profitable in these states?
2. **Review Patterns by Time of Day:**
    - At what times of the day do businesses receive the highest volume of reviews?
    - What is the distribution of positive and negative reviews across different times of the day?
3. **Business Category Distribution:**
    - Which business category has the highest number of businesses?
    - Which category has the fewest businesses? (This will be visualized through a bar graph for clarity.)

These business questions are designed to provide insights that can guide operational improvements, strategic planning, and performance optimizations. The answers will be derived from data analysis and supported by visualizations, enabling stakeholders to make data-driven decisions.

OSEMN Process

1. Obtain

The data was retrieved from the Yelp database website in the form of a zip file. There are 5 files in this dataset – businesses, reviews, users, checkins and tips. The file yelp_academic_dataset _business.json has data that contains information about business which includes business_id, location, attributes and categories of businesses that business serves. The file yelp_academic_dataset _reviews.json has json data which contains reviews contains full review text data including the user_id that wrote the review and the business_id the review is written for. The file yelp_academic_dataset _users.json has user data and all the metadata associated with the user. The file yelp_academic_dataset _checkin,json includes records of checkins on a business. The file yelp_academic_dataset _tips.json contains tips written by a user on a business. Tips are shorter than reviews and tend to convey quick suggestions.

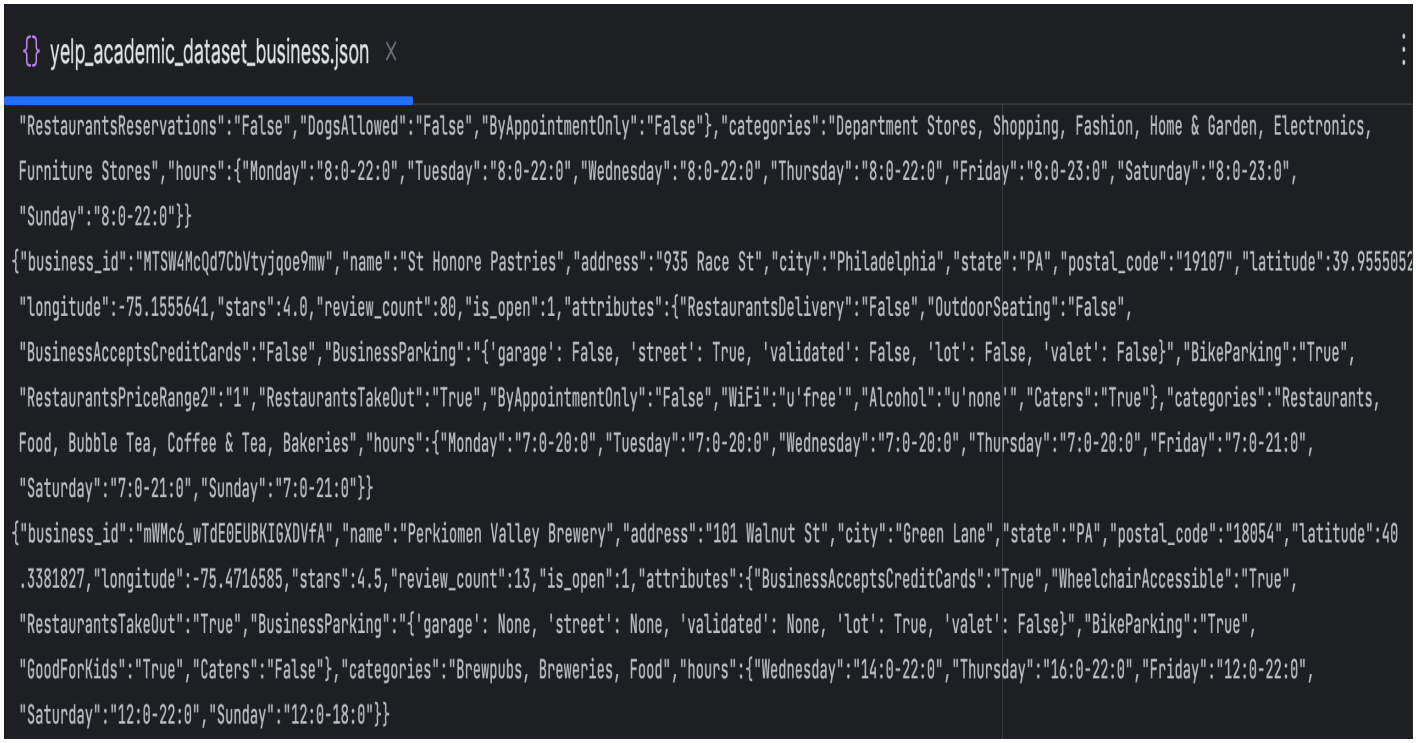
2. Scrub

After retrieving the data, the yelp_academic_dataset _business.json and yelp_academic_dataset _review.json files were reformatted into properly structured JSON files(initial_business.json and initial_reviews.json). The original files had missing commas between JSON objects in each line. To address this, regular expressions (regex) were used to match the pattern of the closing brace } and the opening brace {, followed by either the business_id or review_id string, and then replaced them with the same pattern, but with a comma inserted between the opening and closing braces.

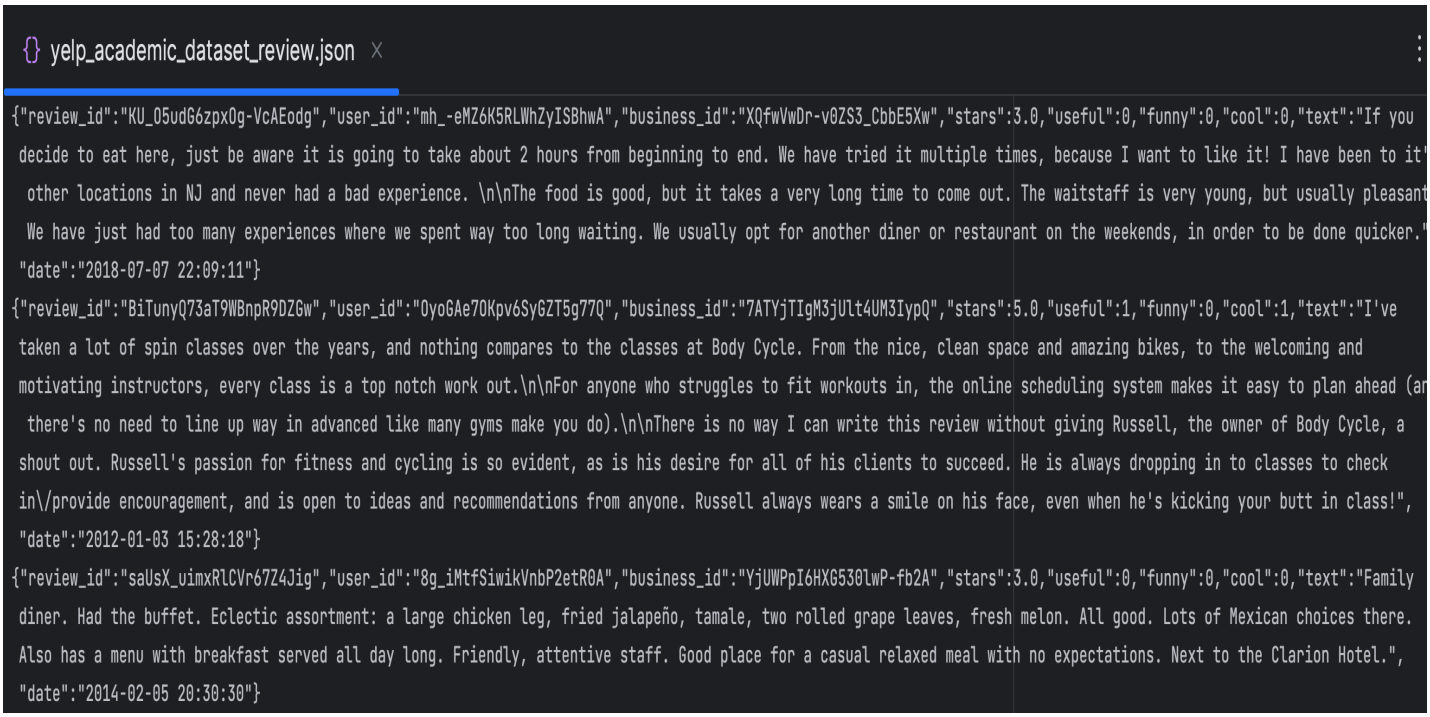
Additionally, unnecessary columns from both files were removed. The cleaned and correctly formatted JSON files were then saved to new files for further analysis.

Initial sample of uncleaned and unformatted json files – The data I started with -

1. yelp_academic_dataset_business.json –



2. yelp_academic_dataset_review..json



Data wrangling

1. Business csv file

I wrote a data wrangling function to clean and transform the raw JSON dataset of businesses from the Yelp Academic Dataset for easier analysis. It begins by reading the raw data and the raw data is stored as a string for further processing. The function corrects the formatting of the JSON data to ensure it can be properly parsed. I replaced occurrences of `}\n*{"business_id":` with `},\n{"business_id":`. This adjustment adds commas between business entries, making the JSON valid for parsing. It then wraps the corrected data with a root structure of JSON which was missing in the raw data. The corrected JSON is loaded into a Python dictionary. This Json file is then converted to csv Once the data is loaded, it removes irrelevant fields such as "attributes" and "hours," which are unnecessary for analysis.

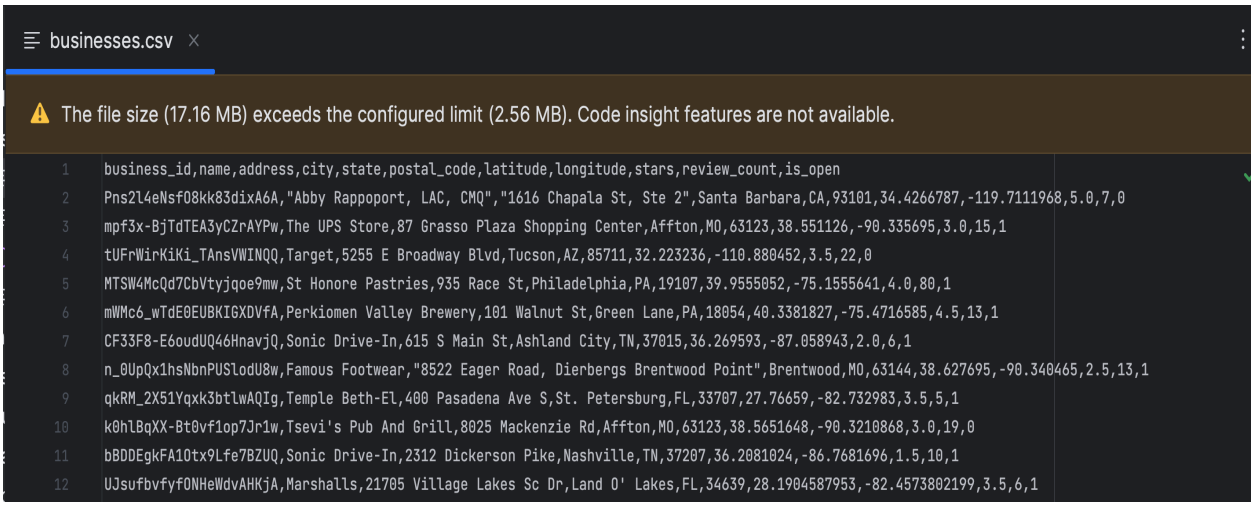
I converted the "categories" field from a comma-separated string to a list format, simplifying the process of analyzing individual categories. Finally, the cleaned and corrected data is saved to a .csv file in a readable format for ease of loading and storage efficiency. I used scripts from previous assignments for the same

2. Reviews json file

The function I wrote for data wrangling of reviews.json file is similar to the data wrangling function of business.json file. The `extracting_reviews_json` function begins by reading the raw data from the raw file and stores it in a string format for formatting and corrections. This The function corrects the formatting of the raw data by replacing occurrences of `}\n*{"review_id":"` with `},\n{"review_id":"` making the data valid JSON. The function then wraps the reviews in a root structure for easier parsing. After loading the corrected JSON, I iterated through each review and removed the `"text"` field to reduce data size and focus on other features. Finally, the cleaned and corrected data is saved to a .csv file in a readable format for ease of loading and storage efficiency. I used scripts from previous assignments for the same

Cleaned and reformatted csv files –

1. businesses.csv



2. reviews.csv

reviews.csv

```
review_id,user_id,business_id,stars,useful,funny,cool,date
KU_O5udG6zpxOg-VcAEodg,mh_-eMZ6K5RLWhZyISBhwA,XQfwVwDr-v0ZS3_CbbE5Xw,3.0,0,0,0,2018-07-07 22:09:11
BiTunyQ73aT9WBnpR9DZGw,OyoGAe7OKpv6SyGZT5g77Q,7ATYjTIgM3jUlt4UM3IypQ,5.0,1,0,1,2012-01-03 15:28:18
saUsX_uimxRlCVr67Z4Jig,8g_iMtfSiwikVnbP2etR0A,YjUWPpI6HXG530lwP-fb2A,3.0,0,0,0,2014-02-05 20:30:30
AqPFMleE6RsU23_auESxiA,_7bHUi9Uuf5__HHc_Q8guQ,kxX2SOes4o-D3ZQBkiMRfA,5.0,1,0,1,2015-01-04 00:01:03
Sx8TMOWLNuJBWer-0pcmoA,bcjbaE6dDog4jkNY91ncLQ,e4Vwtrqf-wpJfwesgvdgxQ,4.0,1,0,1,2017-01-14 20:54:15
JrIxlS1TzJ-iCu79ul40cQ,eUta8W_HdHMXPzLBBZhL1A,04UD14gamNjLY0IDYVhHJg,1.0,1,2,1,2015-09-23 23:10:31
6AxgBCNX_PNTOxmbRSwcKQ,r3zeYsv1XFBRA4dJpL78cw,gmjsEdUsKpj9Xxu6pdjH0g,5.0,0,2,0,2015-01-03 23:21:18
_ZeMknuYdlQcUqng_Im3yg,yfFzsLmaWF2d4Sr0UNbBgg,LHSTtnW3YHCeUkRDGyJOyw,5.0,2,0,0,2015-08-07 02:29:16
ZKvDG2sBvHVdF5oBNUOpAQ,wSTuiTk-sKNdcFyprzZAjg,B5XSoSG3SfvQGtKEGQ1tSQ,3.0,1,1,0,2016-03-30 22:46:33
pUycOfUwM8vqX7KjRRhUEA,59MxRhNVhU9MYndMkz0wtw,gebiRewfieSdtt17PTW6Zg,3.0,0,0,0,2016-07-25 07:31:06
rGQRf8UafX7OTlMNN19I8A,1WHRWwQmZOZDAhp2Qyny4g,uMvVYRgGNXf5boolA9HXTw,5.0,2,0,0,2015-06-21 14:48:06
l3Wk_mvAog6XANIuGQ9C7Q,ZbqSHbgCjzVAqaa7NKWn5A,EQ-TZ2eeD_E0BHuvoaeG5Q,4.0,0,0,0,2015-08-19 14:31:45
XW_LfMv0fV21l9c6xQd_lw,9OAtfnWag-ajVxRbUTGIyg,lj-E32x9_FA7GmUrBGBEWg,4.0,0,0,0,2014-06-27 22:44:01
8JFGBuHMoiNDyfcxuWNtrA,smOvOajNG0lS4Pq7d8g4JQ,RZtGWDLCAtuipwaZ-UfjmQ,4.0,0,0,0,2009-10-14 19:57:14
UBp0zWyH60Hmw6Fsasei7w,4Uh27DgGzsp6PqrH913giQ,otQS34_MymijPTdNBoBdCw,4.0,0,2,0,2011-10-27 17:12:05
```

3. categories.csv

categories.csv

```
business_id,category
Pns2l4eNsfO8kk83dixA6A,Doctors
Pns2l4eNsfO8kk83dixA6A,Traditional Chinese Medicine
Pns2l4eNsfO8kk83dixA6A,Naturopathic/Holistic
Pns2l4eNsfO8kk83dixA6A,Acupuncture
Pns2l4eNsfO8kk83dixA6A,Health & Medical
Pns2l4eNsfO8kk83dixA6A,Nutritionists
mpf3x-BjTdTEA3yCZrAYPw,Shipping Centers
mpf3x-BjTdTEA3yCZrAYPw,Local Services
mpf3x-BjTdTEA3yCZrAYPw,Notaries
mpf3x-BjTdTEA3yCZrAYPw,Mailbox Centers
mpf3x-BjTdTEA3yCZrAYPw,Printing Services
tUFrWirKiKi_TAnsVWINQQ,Department Stores
tUFrWirKiKi_TAnsVWINQQ,Shopping
tUFrWirKiKi_TAnsVWINQQ,Fashion
tUFrWirKiKi_TAnsVWINQQ,Home & Garden
tUFrWirKiKi_TAnsVWINQQ,Electronics
tUFrWirKiKi_TAnsVWINQQ,Furniture Stores
MTSW4McQd7CbVtyjqoe9mw,Restaurants
MTSW4McQd7CbVtyjqoe9mw,Food
MTSW4McQd7CbVtyjqoe9mw,Bubble Tea
MTSW4McQd7CbVtyjqoe9mw,Coffee & Tea
MTSW4McQd7CbVtyjqoe9mw,Bakeries
mWMc6_wTdE0EUBKIGXDVfA,Brewpubs
mWMc6_wTdE0EUBKIGXDVfA,Breweries
mWMc6_wTdE0EUBKIGXDVfA,Food
```

Snippet of code wrote to scrub the files –

1. business.py

business.py

```python
import json
import re
import csv

1 usage
def correcting_business_json(raw_file):
    with open(raw_file,'r') as rfile:
        raw_data = rfile.read()
    fixed_json = re.sub( pattern: '}\n*{"business_id":', repl: '},\n{"business_id":', raw_data.strip())
    fixed_json = '{\n"businesses": [' + fixed_json + ']\n}'
    json_data = json.loads(fixed_json)

    for business in json_data["businesses"]:
        business.pop("attributes", None)
        business.pop("hours", None)
        if business["categories"]:
            business["categories"] = list(business["categories"].split(", "))
        else: []

    with open("initial_business.json", "w") as file:
        json.dump(json_data, file, indent=4)

correcting_business_json('yelp_academic_dataset_business.json')
```

business.py

```python
correcting_business_json('yelp_academic_dataset_business.json')

1 usage
def json_to_csv(json_file):
    with open(json_file, 'r') as jfile:
        content = json.load(jfile)
    list_of_businesses = list(content.values())

    for item in list_of_businesses[0]:
        fieldnames = item.keys()

    with open('intermediate_business.csv', 'w', newline='') as fp:
        writer = csv.DictWriter(fp, fieldnames)
        writer.writeheader()

        for item in list_of_businesses[0]:
            writer.writerow(item)
json_to_csv('initial_business.json')
```

2. reviews.py

reviews.py

```python
import re
import csv


1 usage
def extracting_reviews_json(raw_file):
    with open(raw_file,'r') as rfile:
        raw_data = rfile.read()
    fixed_json = re.sub( pattern: '}\n*{"review_id":"', repl: '},\n{"review_id":"', raw_data.strip())
    fixed_json = '{\n"reviews": [' + fixed_json + ']\n}'

    json_data = json.loads(fixed_json)
    for review in json_data["reviews"]:
        review.pop("text", None)

    with open('initial_reviews.json', 'w') as jfile:
        json.dump(json_data, jfile, indent=4)

    print("Initial json file created")
extracting_reviews_json('yelp_academic_dataset_review.json')
```

reviews.py

```python
    print("Initial json file created")
extracting_reviews_json('yelp_academic_dataset_review.json')

1 usage
def json_to_csv(json_file):
    with open(json_file, 'r') as jfile:
        content = json.load(jfile)
    list_of_reviews = list(content.values())

    for item in list_of_reviews[0]:
        fieldnames = item.keys()

    with open('intermediate_reviews.csv', 'w', newline='') as fp:
        writer = csv.DictWriter(fp, fieldnames)
        writer.writeheader()

        for item in list_of_reviews[0]:
            writer.writerow(item)
json_to_csv('initial_reviews.json')
```

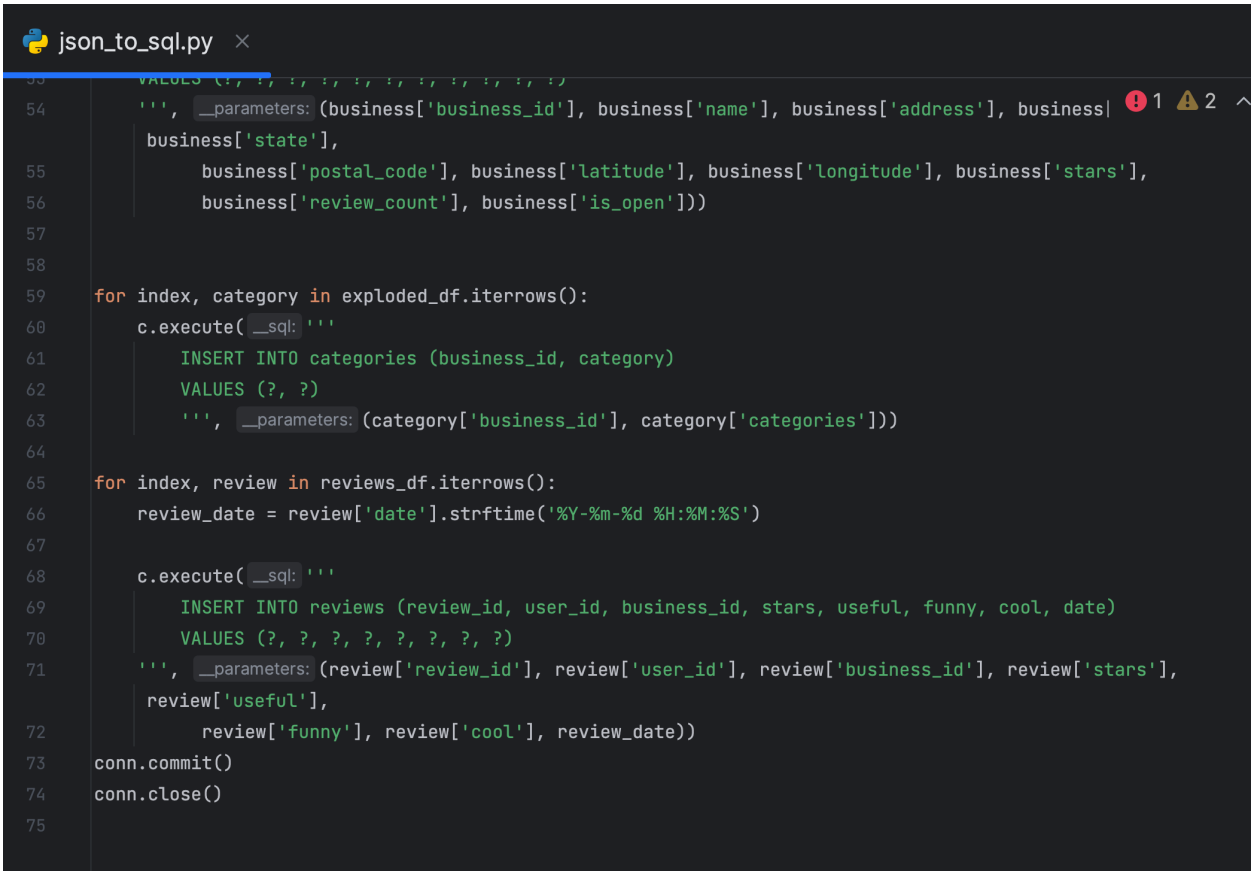
I converted business.csv to business and categories table in this script and extracted the categories.csv from the sqllite –

json_to_sql.py

```python
import ...

business_df = pd.read_json('initial_business.json')
reviews_df = pd.read_json('initial_reviews.json')

conn = sqlite3.connect('yelp2.db')
c = conn.cursor()

c.execute('''
CREATE TABLE IF NOT EXISTS businesses (
    business_id TEXT PRIMARY KEY,
    name TEXT,
    address TEXT,
    city TEXT,
    state TEXT,
    postal_code TEXT,
    latitude REAL,
    longitude REAL,
    stars REAL,
    review_count INTEGER,
    is_open INTEGER
);
''')

c.execute('''
CREATE TABLE IF NOT EXISTS categories (
    business_id TEXT,
    category TEXT
);
''')
```

json_to_sql.py

```python
)

c.execute('''
CREATE TABLE IF NOT EXISTS reviews (
    review_id TEXT PRIMARY KEY,
    user_id TEXT,
    business_id TEXT,
    stars REAL,
    useful INT,
    funny INT,
    cool INT,
    date DATETIME,
    FOREIGN KEY(business_id) REFERENCES businesses(business_id)
);
''')

category_data = business_df[['business_id','categories']]
exploded_df = category_data.explode('categories', ignore_index=True)

for index, business in business_df.iterrows():
    c.execute( _sql: '''
    INSERT INTO businesses (business_id, name, address, city, state, postal_code, latitude, longitude,
     stars, review_count, is_open)
    VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
    ''', _parameters: (business['business_id'], business['name'], business['address'], business['city'],
     business['state'],
          business['postal_code'], business['latitude'], business['longitude'], business['stars'],
          business['review_count'], business['is_open']))
```



3. Explore – Data Analysis

The cleaned and formatted files are ready to be explored and analyzed for business insights.

Question 1 - Review Trends by Time of Day:

Description – Understanding when businesses receive the most reviews, what are the peak months and how the trend has changed over the years is critical for optimizing customer engagement and service delivery for the yelp database.

I focused on review trends over time, with particular emphasis on understanding peak months, peak time of the day and how the trend has evolved year-over-year. By analyzing when businesses receive the most reviews, I aim to uncover patterns of customer engagement, identify seasonal variations, and track any significant shifts over time

Key components of the question include:

1. **Peak months**: Determining which months see the highest volume of reviews in the months of all years.
2. **Year-over-year changes**: Identifying trends over multiple years to see if customer engagement is increasing or fluctuating.
3. **Time-of-day patterns**: Analyzing if there is a time-of-day effect in review submissions

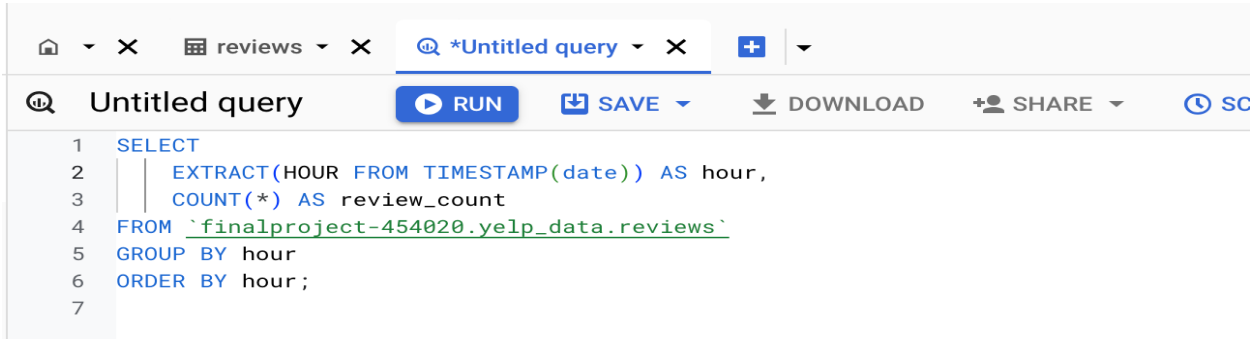
Process to answer the question – After loading the reviews table into the Google BigQuery, we will write queries for each cases. As we want to know the volume of the reviews for each case, we will extract the required time from the date column in the table. We will then select the count of number of reviews for each time that we extracted like hour of the day, month of the year and overall year analysis

Data Analysis –

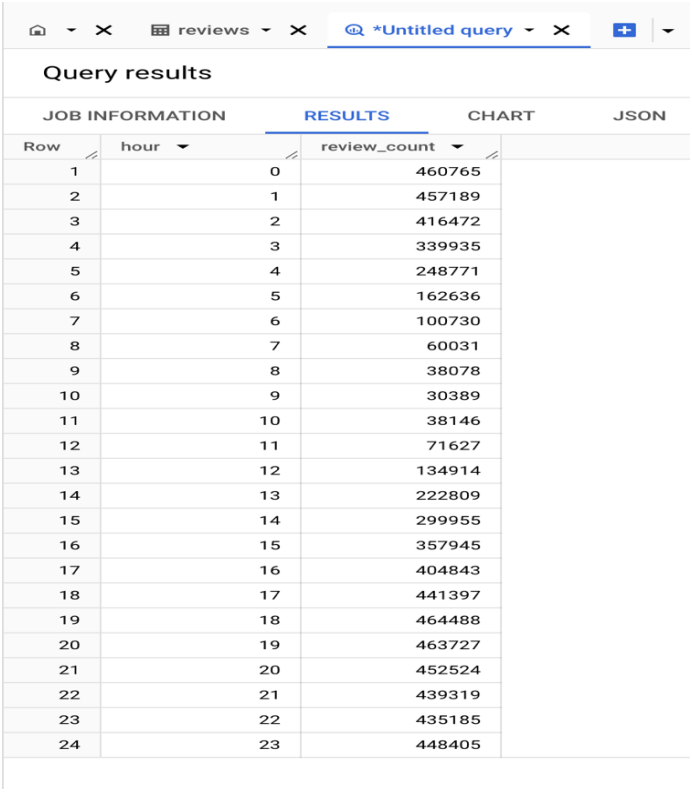
I queried for all the 3 cases in Google BigQuery –

Data analysis on time-of-day pattern

Query -

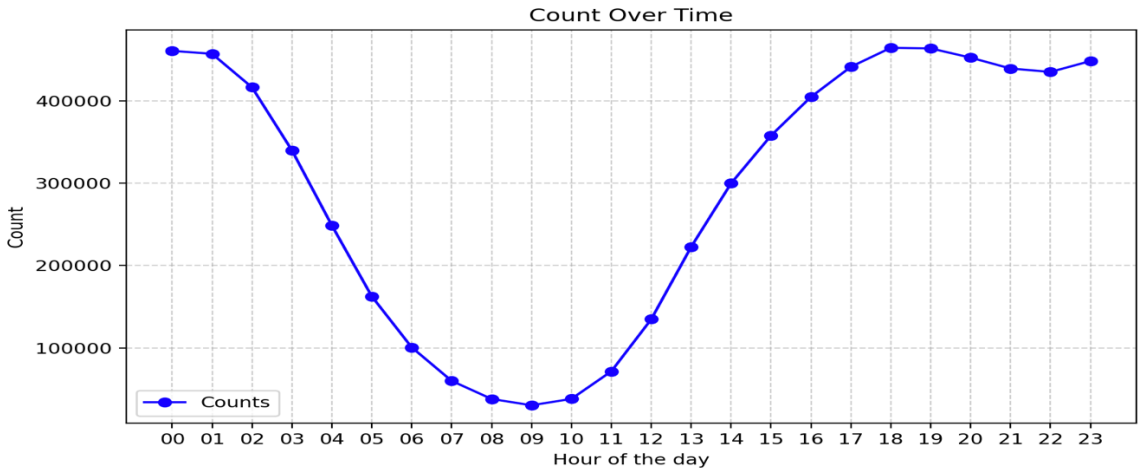


The query results were as follows -



Row	hour	review_count
1	0	460765
2	1	457189
3	2	416472
4	3	339935
5	4	248771
6	5	162636
7	6	100730
8	7	60031
9	8	38078
10	9	30389
11	10	38146
12	11	71627
13	12	134914
14	13	222809
15	14	299955
16	15	357945
17	16	404843
18	17	441397
19	18	464488
20	19	463727
21	20	452524
22	21	439319
23	22	435185
24	23	448405

Data visualizations - Time of day pattern



Data analysis on monthly pattern

reviews | *Untitled query

Untitled query | RUN | SAVE | DOWNLOAD

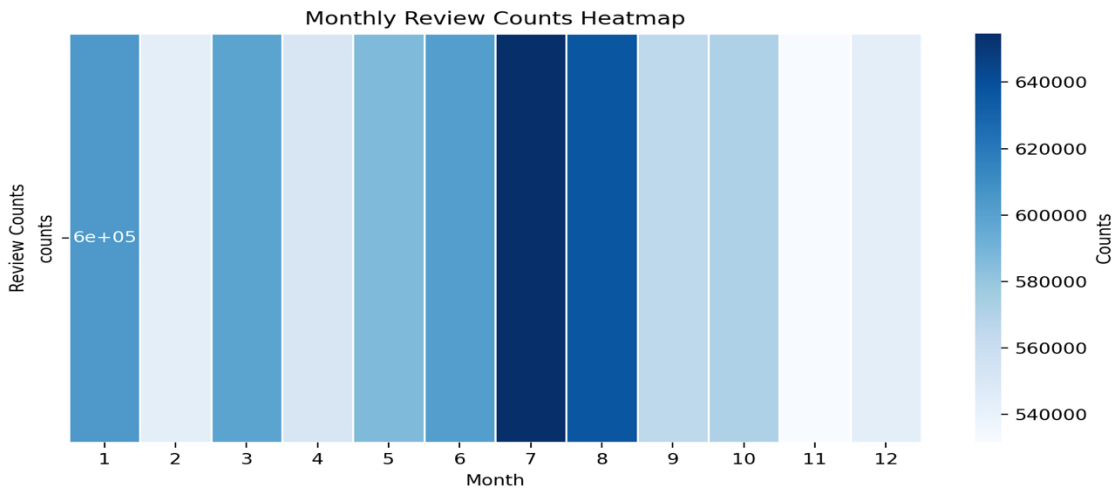
```
1  SELECT
2      EXTRACT(month FROM TIMESTAMP(date)) AS month,
3      COUNT(*) AS review_count
4  FROM `finalproject-454020.yelp_data.reviews`
5  GROUP BY month
6  ORDER BY month;
7
```

Query results

JOB INFORMATION | RESULTS | CHA

Row	month	review_count
1	1	604532
2	2	544125
3	3	598555
4	4	551471
5	5	586575
6	6	601737
7	7	654627
8	8	636384
9	9	565374
10	10	571809
11	11	531518
12	12	543573

Data visualization - Monthly trends –



Data analysis on yearly pattern

Query –

reviews *Untitled query

Untitled query RUN SAVE DOWN

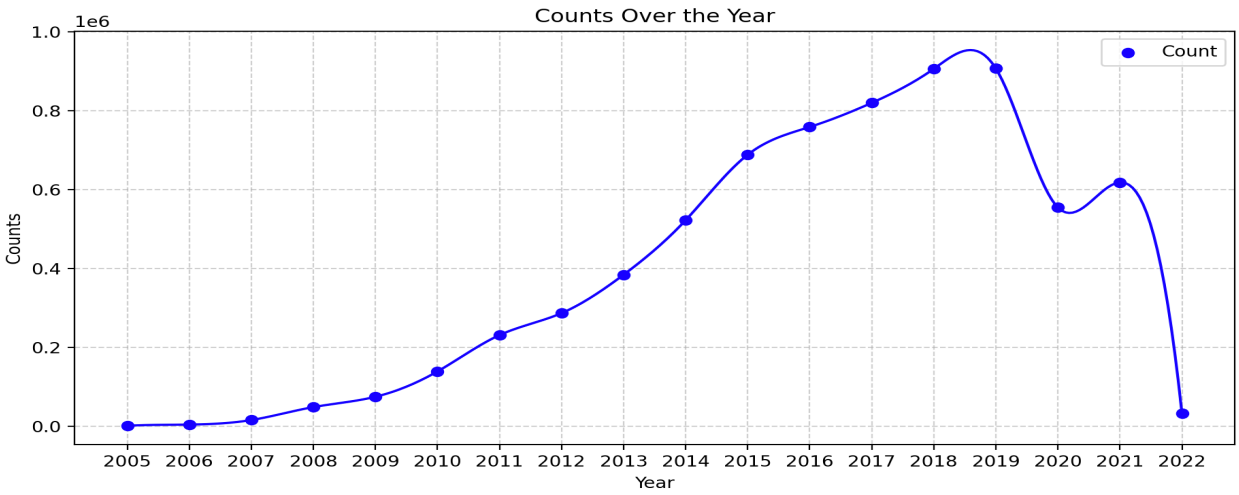
```
1  SELECT
2      EXTRACT(year FROM TIMESTAMP(date)) AS year,
3      COUNT(*) AS review_count
4  FROM `finalproject-454020.yelp_data.reviews`
5  GROUP BY year
6  ORDER BY year;
7
```

Query results

JOB INFORMATION | RESULTS | CHART

Row	year	review_count
1	2005	854
2	2006	3853
3	2007	15363
4	2008	48226
5	2009	74387
6	2010	138587
7	2011	230813
8	2012	286570
9	2013	383950
10	2014	522275
11	2015	688415
12	2016	758882
13	2017	820048
14	2018	906362
15	2019	907284
16	2020	554557
17	2021	618189
18	2022	31665

Yearly trends –



Analyses/Answer –

- Time of day – The first visualization illustrates a clear daily cycle. It displays that peak review activity occurs around midnight (00:00 - 01:00) and evening hours (17:00 - 22:00). The lowest activity occurs between 06:00 and 10:00, indicating that fewer reviews are submitted in the early morning. There is a sharp increase in review activity after midday, likely due to users engaging more in the evening.
- Monthly trends –The heatmap shows the monthly review counts, suggests that review volumes peak around July and August (darker shades indicating higher counts). There are fluctuations throughout the year, but mid-year (summer) appears to have the highest activity, while months like February and November show relatively lower counts.
- Year-to-year changes – The counts over the year, shows a significant increase in review counts from 2005 to around 2019, with a peak around 2019. However, there is a steep decline after 2019, likely due to external factors such as the pandemic in 2020, which might have impacted user engagement. A slight recovery is visible around 2021, but 2022 shows another sharp drop, indicating fluctuating customer engagement trends in recent years.

Question 2 – Does business with more categories mean business with more rating?

Description - This analysis aims to identify how having many different business categories under a single business (business_id) affects the rating of that business. Understanding this will lead us to answer the question that if a business operates different categories, does it attract more customers and hence higher rating or business that operate under just one category can focus solely and gain a higher rating than the former.

By analyzing this, I aim to uncover patterns of correlation between a rating a business has and its subcategories.

Process – I extracted the number of categories associated with each business (business_id) from the categories table by counting the occurrences of each business in the dataset. To analyze the relationship between the number of categories and business ratings, I joined this data with the

businesses table using an inner join on business_id to retrieve the corresponding stars (ratings) for each business.

After obtaining the dataset with business_id, category count, and stars, I grouped the data by business_id to ensure each business appears only once in the results. Finally, I sorted the businesses in ascending order based on the number of categories to observe trends between category diversity and business ratings.

I have plotted a box plot which groups ratings (stars) by category count to see how ratings are distributed for businesses with different categories counts.

Data Analysis –

Query –

*Untitled query

Untitled query RUN SAVE DOWNLOAD SHARE SCHEDULE OPEN IN MORE

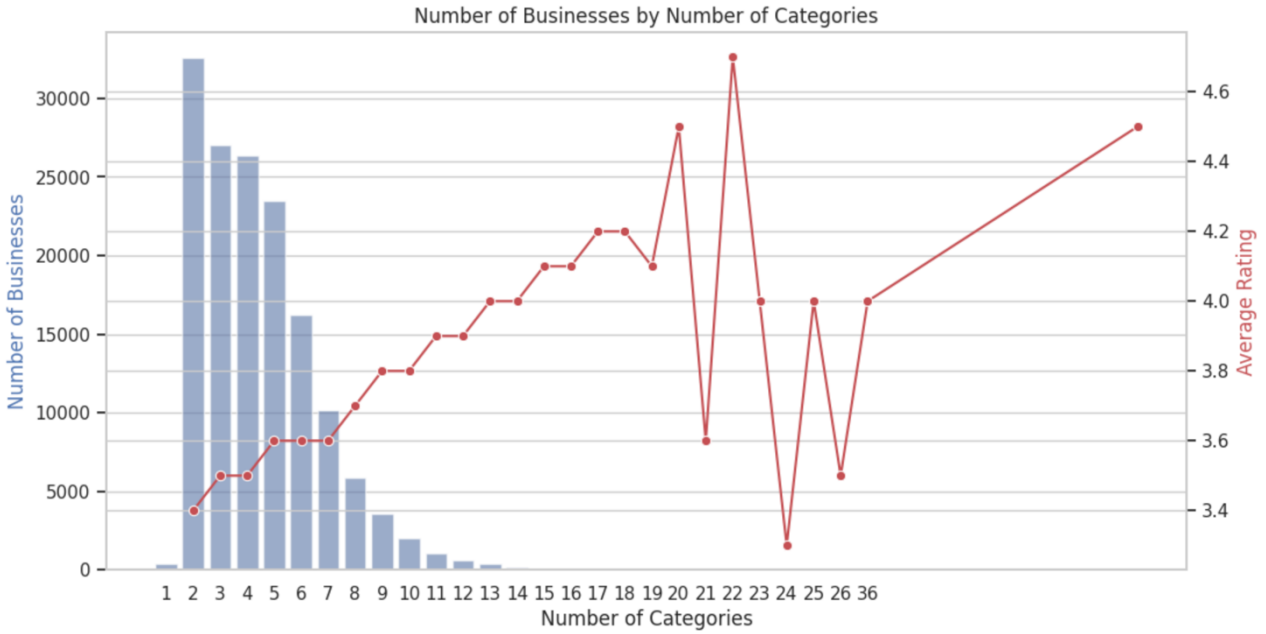
```
1  WITH CategoryCounts AS (
2      SELECT business_id, COUNT(category) AS category_count
3      FROM `firstproject-436502.yelp.categories`  WHERE category IS NOT NULL
4      GROUP BY business_id
5  )
6  SELECT
7      cc.category_count AS num_categories,
8      COUNT(b.business_id) AS num_businesses,
9      MIN(b.stars) AS min_rating,
10     MAX(b.stars) AS max_rating,
11     ROUND(AVG(b.stars),1) AS avg_rating
12 FROM CategoryCounts cc
13 JOIN `firstproject-436502.yelp.businesses`  b ON cc.business_id = b.business_id
14 GROUP BY cc.category_count
15 ORDER BY cc.category_count;
16
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	num_categories	num_businesses	min_rating	max_rating	avg_rating
1	1	381	1.0	5.0	3.4
2	2	32563	1.0	5.0	3.5
3	3	27065	1.0	5.0	3.5
4	4	26354	1.0	5.0	3.6
5	5	23450	1.0	5.0	3.6
6	6	16233	1.0	5.0	3.6
7	7	10189	1.0	5.0	3.7
8	8	5869	1.0	5.0	3.8
9	9	3598	1.0	5.0	3.8
10	10	2024	1.0	5.0	3.9
11	11	1083	1.0	5.0	3.9
12	12	647	1.5	5.0	4.0
13	13	373	1.0	5.0	4.0
14	14	178	2.0	5.0	4.1
15	15	88	1.5	5.0	4.1
16	16	60	2.0	5.0	4.2
17	17	41	3.0	5.0	4.2
18	18	21	2.5	5.0	4.1
19	19	10	4.0	5.0	4.5
20	20	4	2.5	4.0	3.6
21	21	3	4.0	5.0	4.7
22	22	3	3.5	4.5	4.0
23	23	2	3.0	3.5	3.3
24	24	1	4.0	4.0	4.0
25	25	1	3.5	3.5	3.5
26	26	1	4.0	4.0	4.0
27	36	1	4.5	4.5	4.5

Data visualization –



Answer/ Analysis –

The majority of businesses operate within a limited range of categories, specifically between 2 and 7. Businesses with only one category display the lowest average rating of 3.4, standing out from all others. There's a gentle upward trend in average ratings as the number of categories increases from 2 to 7. The average ratings is higher when businesses have 8 to 17 categories. Businesses with an extremely high number of categories (18+) are very rare, making conclusive generalizations difficult. The average ratings tends to increase the more number of categories exists among the businesses. This suggests diversification may enhance customer satisfaction and increase ratings for businesses. Businesses with more categories may be able to offer variety of service and cater diverse audience. Businesses operating in niches may experience high cutomer satisfaction. The trend of increasing average ratings may be disrupted by external factors.

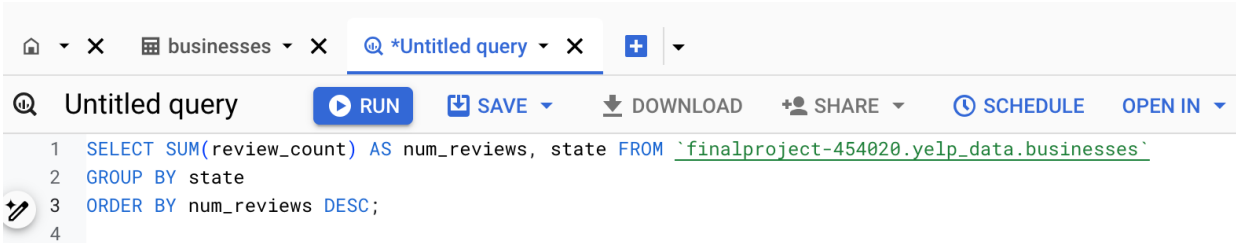
Question 3 – Number of reviews and states of US.

Description - This analysis aims to identify the number of users in each state using the Yelp website by analyzing the total number of reviews across all businesses within each state in the U.S. The primary objective is to explore how the number of reviews varies from state to state. Some states may have more businesses or higher levels of customer activity, leading to a greater number of reviews. Additionally, patterns or correlations may emerge, such as whether states with higher population densities tend to have more reviews or if specific states show significantly more business activity and reviews than others. By examining these variations, this analysis seeks to provide insights into geographic trends in user engagement on the Yelp platform.

Process – I extracted the total review count per state by summing the reviews column and grouping the results by state from the businesses table. This aggregated the review data by state, allowing me to analyze how review volumes vary across the U.S. I then visualized the data on a bubble map to identify geographic patterns and trends in user activity, such as which states have the highest or lowest review counts. The visualization is a bubble map showing review count across different U.S. states, with numeric labels indicating values associated with each location. I used Tableau to create this visualization and also implemented hover which you can see by clicking the link.

Data Analysis

Query -



Query results -

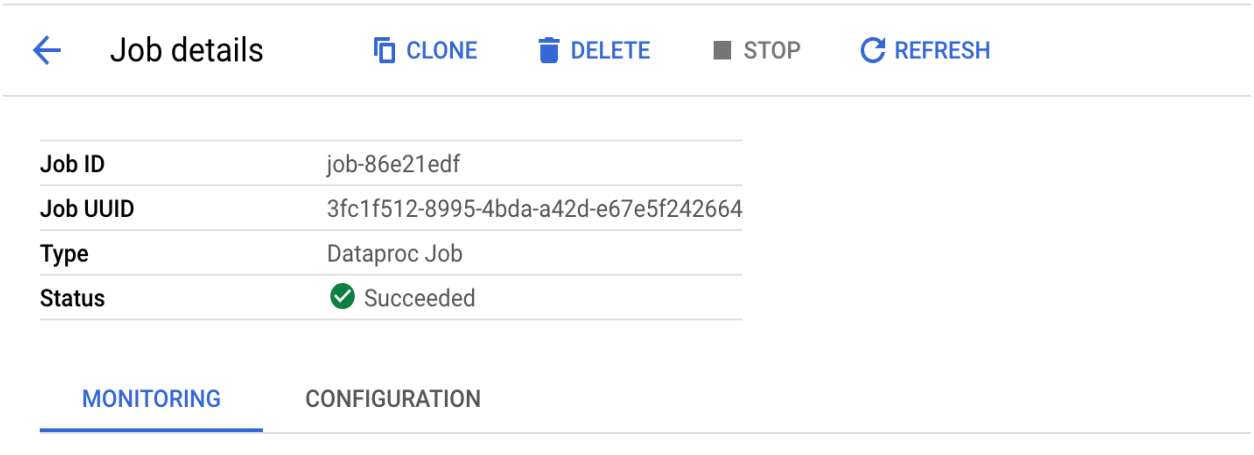
Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS

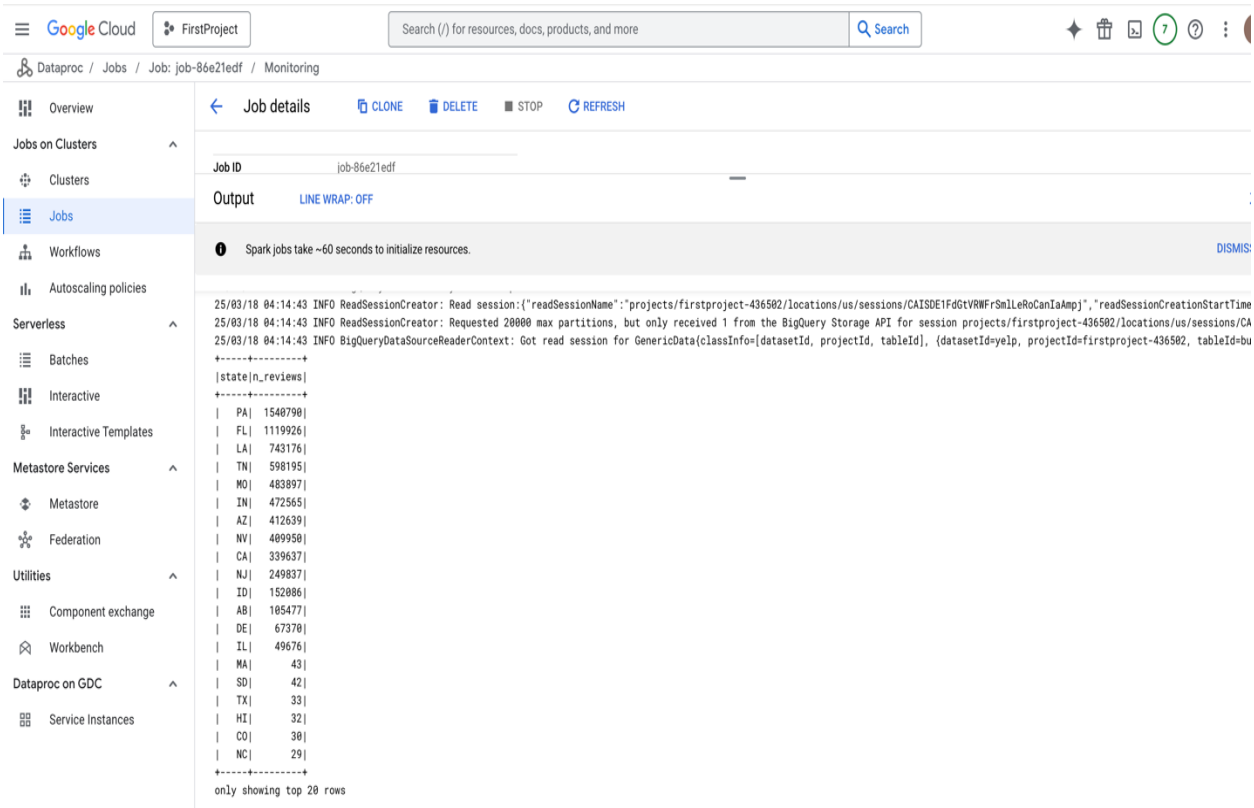
Row	num_reviews	state
1	1540790	PA
2	1119926	FL
3	743176	LA
4	598195	TN
5	483897	MO
6	472565	IN
7	412639	AZ
8	409950	NV
9	339637	CA
10	249837	NJ
11	152086	ID
12	105477	AB
13	67370	DE
14	49676	IL
15	43	MA
16	42	SD
17	33	TX
18	32	HI
19	30	CO
20	29	NC
21	19	WA
22	19	UT
23	10	VI
24	10	VT
25	9	MI
26	6	MT

I also ran the same query through a dataproc job and got similar results –

Job details -



Results -



Python script -

Window_spark_yelp.py

```python
#!/usr/bin/python
from pyspark.sql import SparkSession

spark = SparkSession \
            .builder \
            .master('yarn') \
            .appName('Yelp') \
            .getOrCreate()

# Use the Cloud Storage bucket for temporary BigQuery export data used
# by the connector.
bucket = "final_project3033"
spark.conf.set('temporaryGcsBucket', bucket)

# Load data from BigQuery.
businesses = spark.read.format('bigquery') \
        .option('table', 'firstproject-436502.yelp.businesses') \
        .load()
businesses.createOrReplaceTempView('businesses')

# Perform word count.
businesses_query = spark.sql(
    'SELECT state AS state, SUM(review_count) as n_reviews FROM businesses GROUP BY state
     ORDER BY n_reviews DESC'
)
businesses_query.show()
businesses_query.printSchema()

# Saving the data to BigQuery
businesses_query.write.format('bigquery') \
    .option('table', 'firstproject-436502.yelp.business_state_reviews') \
```

Data visualization –

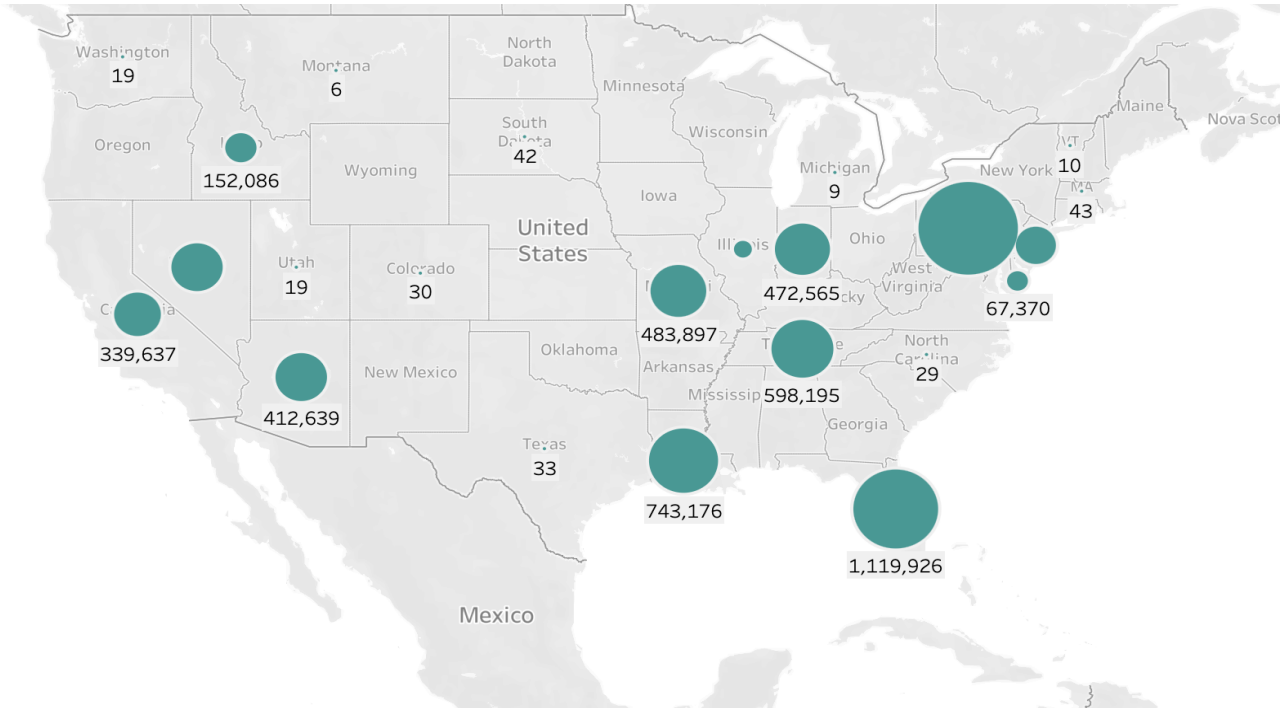


Tableau link -
https://public.tableau.com/shared/3295BWPW2?:display_count=n&:origin=viz_share_link

Answer/Analysis –

The review counts are spread across the entire US region, but some states have no data ot have verry small count.

There are some high value clusters in the map. Pennsylvania has the largest bubble with a value of 1,540,790 indicating it leads in the reviews obtained by businesses over the years. This is followed by Florida with a value of 1,119,926. Texas (743,176), a southern state, also has a significant concentration. The western states California (339,637) and Arizona (412,639) have mid-range counts.

Some Northern states and those in the Pacific Northwest (e.g., Washington with 19 and Montana with 6) have lower review count.

We can clearly say there is a noticeable concentration of high user engagement in the southern and midwestern regions. Northern and sparsely populated states tend to have less reviews, suggesting differences in either population or activity related to the measured metric.

Table Schema for the Relational Database:

The relational database used in the provided code consists of three tables:

1. **businesses**:
    - **business_id (TEXT PRIMARY KEY)**: Unique identifier for each business.
    - **name (TEXT)**: The name of the business.
    - **address (TEXT)**: The physical address of the business.
    - **city (TEXT)**: The city where the business is located.
    - **state (TEXT)**: The state where the business is located.
    - **postal_code (TEXT)**: The postal code for the business address.
    - **latitude (REAL)**: Latitude coordinate of the business.
    - **longitude (REAL)**: Longitude coordinate of the business.
    - **stars (REAL)**: Average rating of the business.
    - **review_count (INTEGER)**: Number of reviews for the business.
    - **is_open (INTEGER)**: Whether the business is currently open (1 = open, 0 = closed).
2. **categories**:
    - **business_id (TEXT)**: Foreign key referencing the `businesses` table to associate categories with businesses.
    - **category (TEXT)**: The category assigned to the business (e.g., "Restaurant", "Hotel").
3. **reviews**:
    - **review_id (TEXT PRIMARY KEY)**: Unique identifier for each review.
    - **user_id (TEXT)**: Identifier for the user who posted the review.
    - **business_id (TEXT)**: Foreign key referencing the `businesses` table to link the review with a specific business.

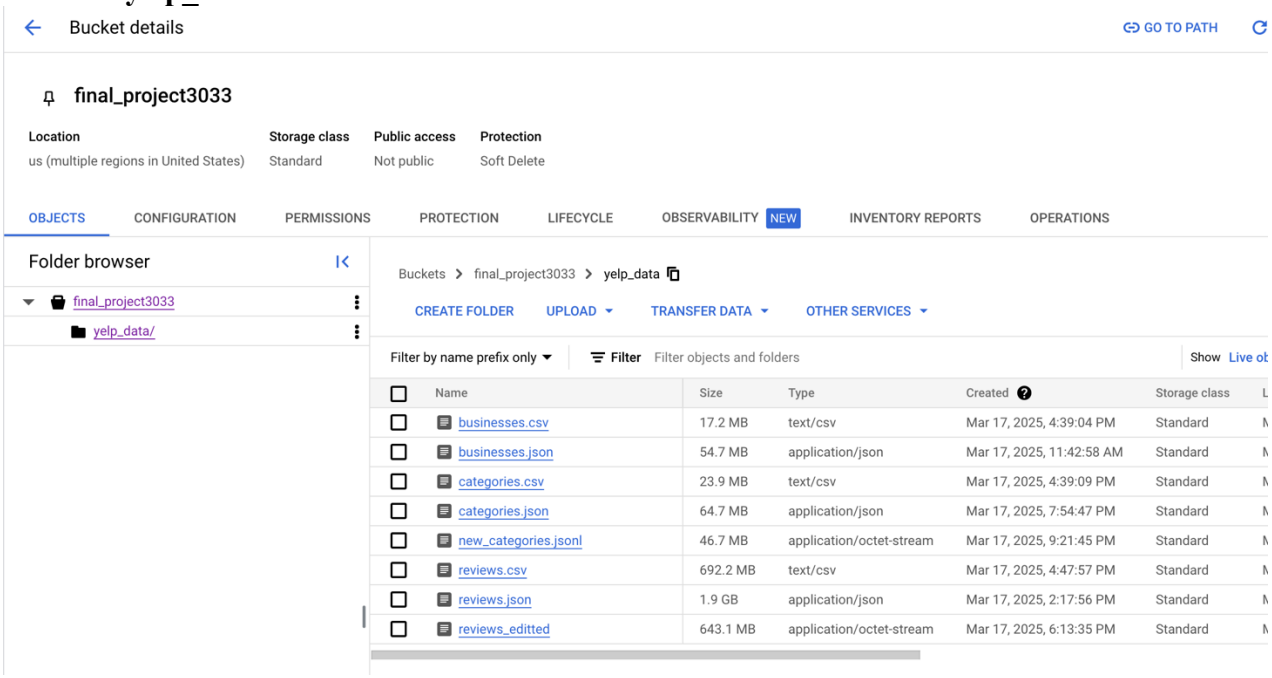
- **stars (REAL)**: Rating given by the user in the review.
- **useful (INTEGER)**: Number of useful votes the review received.
- **funny (INTEGER)**: Number of funny votes the review received.
- **cool (INTEGER)**: Number of cool votes the review received.
- **date (DATETIME)**: Date and time when the review was posted.

Process Used to Load Data in Cloud for Querying

Step 1 – Initial data loading –

I wrote python scripts to convert the initial data of businesses and reviews to 3 tables of businesses, categories and reviews each(snippets above). Then I converted them to csv formats and uploaded the files from local them to the google storage bucket using the upload files option.

Bucket name – final_project3033
Folder – yelp_data

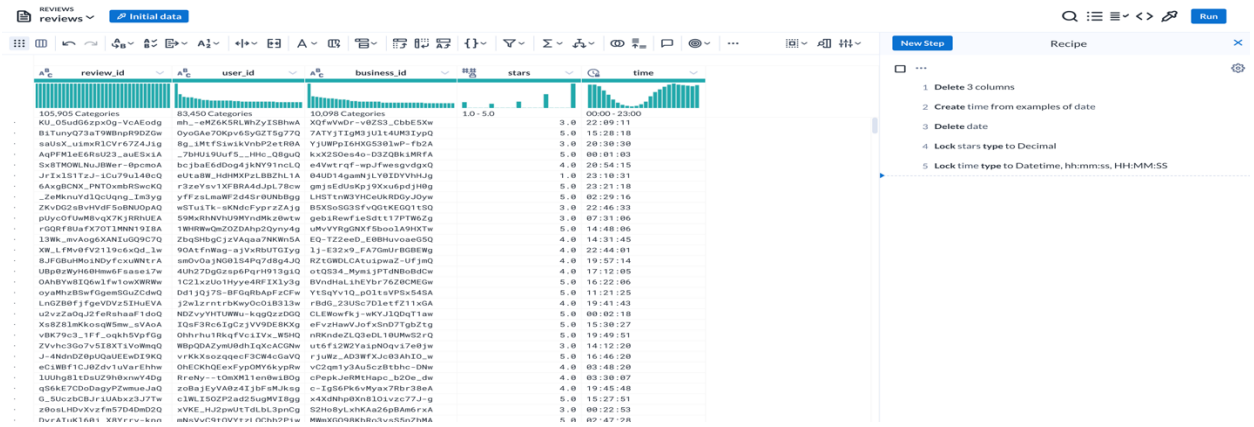


2. Modifying table through dataproc –

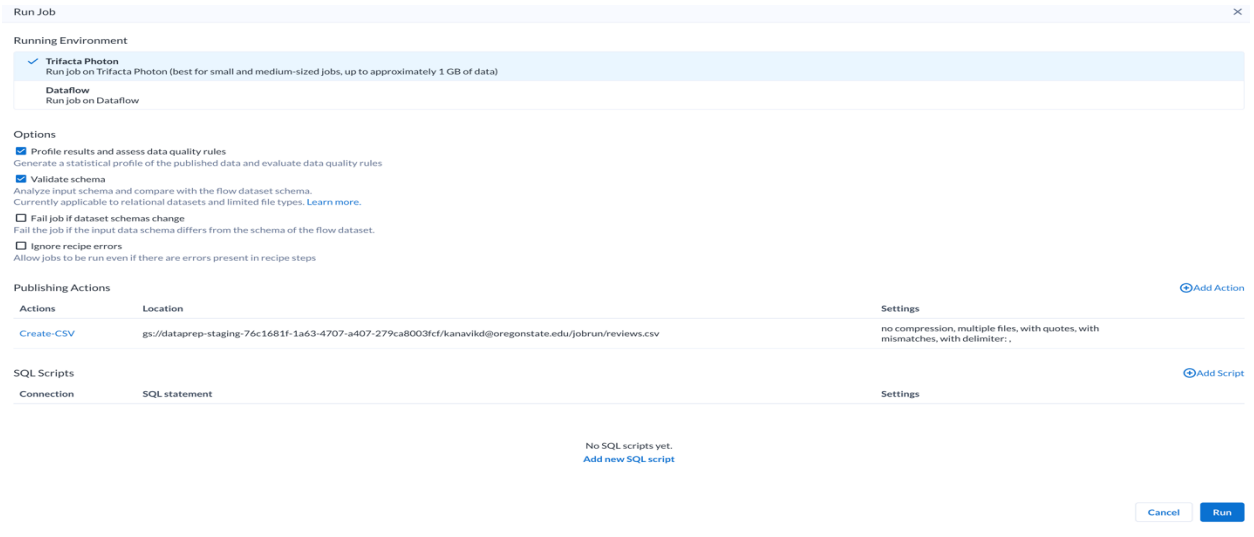
I edited one table through dataproc initially to check results. But eventually I changed it back to original format. However my dataproc flow job ran successfully and I was able to get the edited file to my local.

Initial file – (reviews)

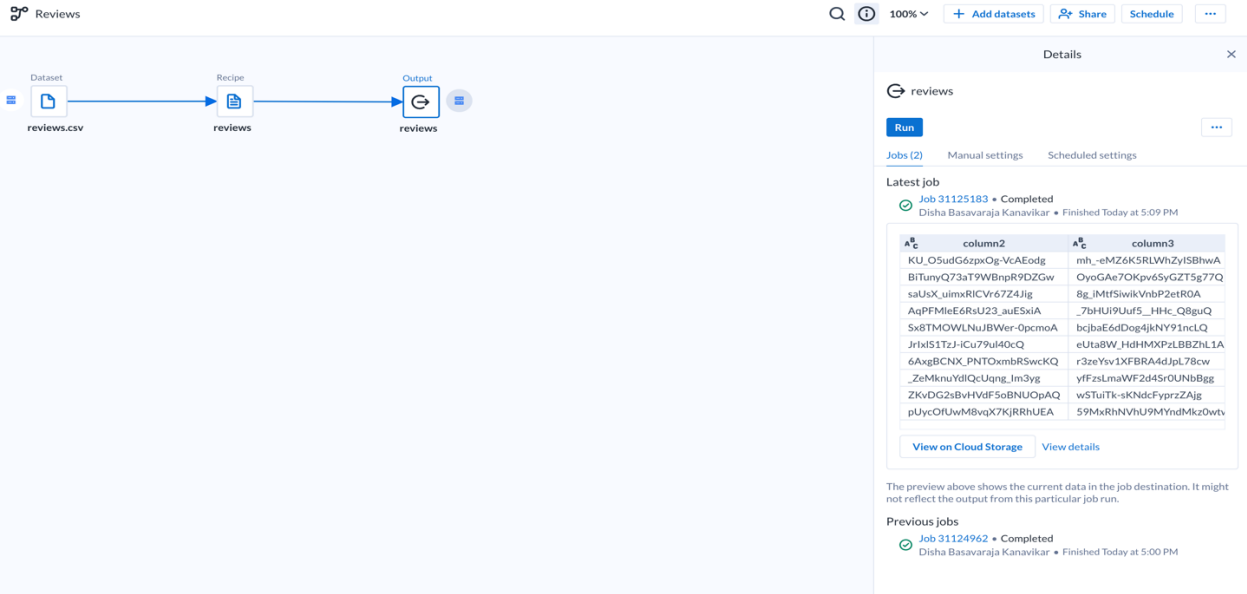
Reviews table modified and its recipe –



Job execution –



Results that the job ran successfully -

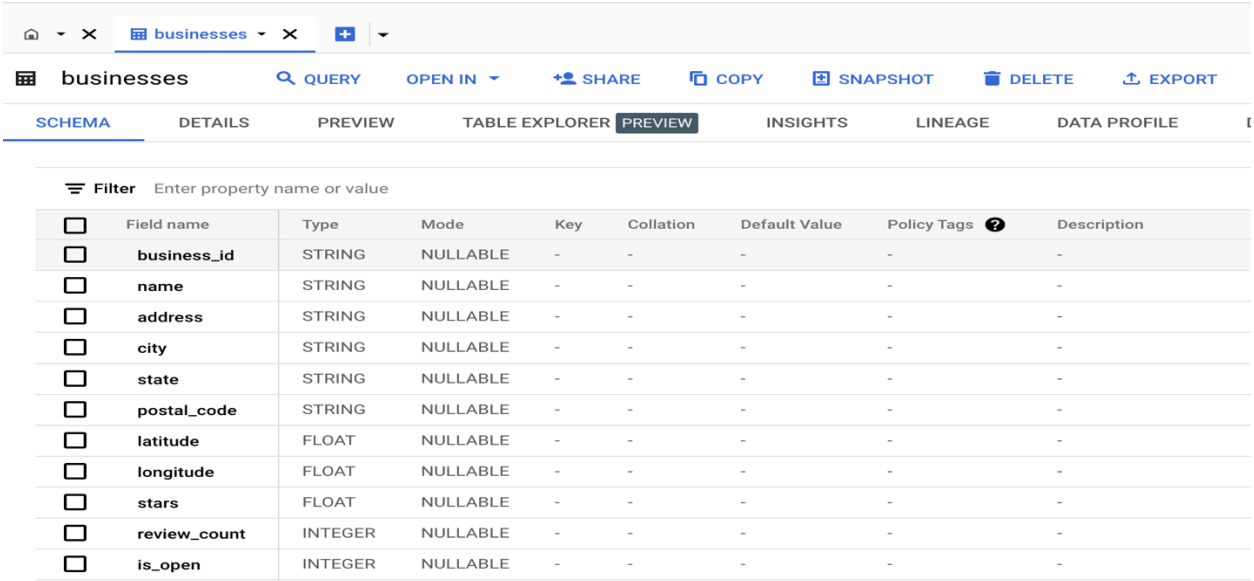


3. Loading the data into BigQuery tables –

I created a new dataset and uploaded the 3 csv files from my google storage bucket with schema autodetection on. This resulted in having perfect tables in bigquery –

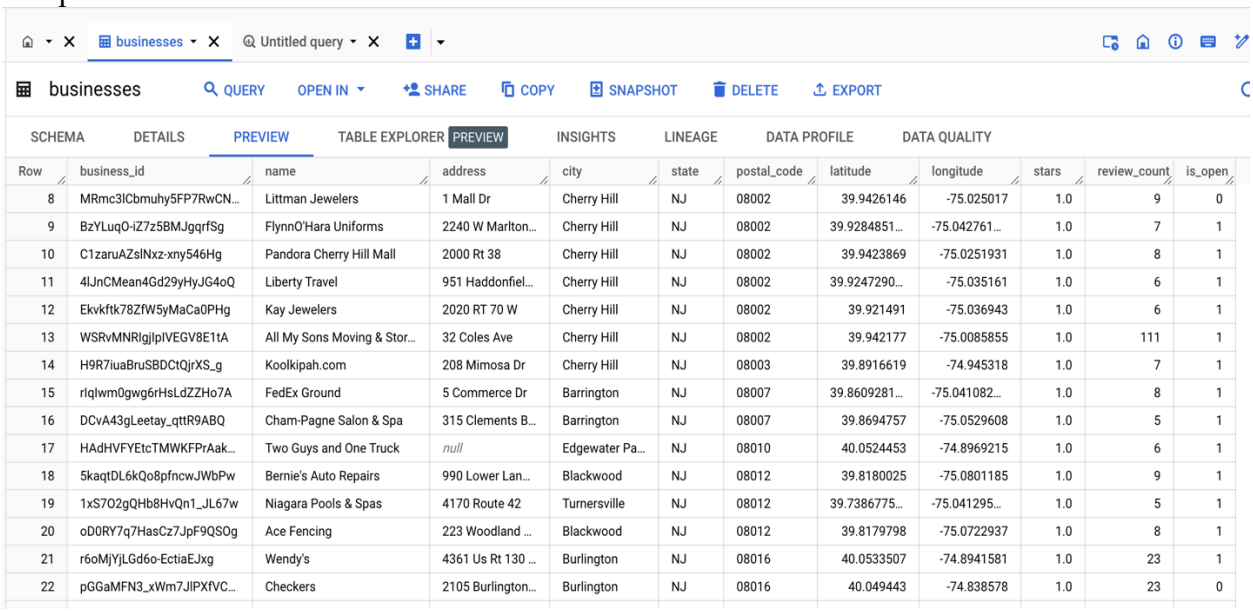
Businesses table

Schema –



Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
business_id	STRING	NULLABLE	-	-	-	-	-
name	STRING	NULLABLE	-	-	-	-	-
address	STRING	NULLABLE	-	-	-	-	-
city	STRING	NULLABLE	-	-	-	-	-
state	STRING	NULLABLE	-	-	-	-	-
postal_code	STRING	NULLABLE	-	-	-	-	-
latitude	FLOAT	NULLABLE	-	-	-	-	-
longitude	FLOAT	NULLABLE	-	-	-	-	-
stars	FLOAT	NULLABLE	-	-	-	-	-
review_count	INTEGER	NULLABLE	-	-	-	-	-
is_open	INTEGER	NULLABLE	-	-	-	-	-

Sample rows –


businesses | Untitled query

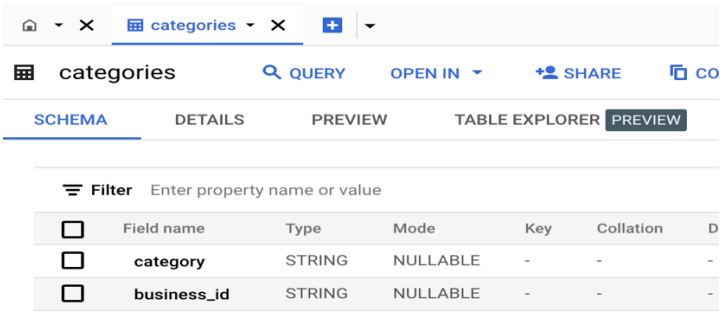
businesses QUERY OPEN IN SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS PREVIEW TABLE EXPLORER PREVIEW INSIGHTS LINEAGE DATA PROFILE DATA QUALITY

Row	business_id	name	address	city	state	postal_code	latitude	longitude	stars	review_count	is_open
8	MRmc3lCbmuhy5FP7RwCN...	Littman Jewelers	1 Mall Dr	Cherry Hill	NJ	08002	39.9426146	-75.025017	1.0	9	0
9	BzYLuqO-iZ7z5BMJgqrfSg	FlynnO'Hara Uniforms	2240 W Marlton...	Cherry Hill	NJ	08002	39.9284851...	-75.042761...	1.0	7	1
10	C1zaruAZslNxz-xny546Hg	Pandora Cherry Hill Mall	2000 Rt 38	Cherry Hill	NJ	08002	39.9423869	-75.0251931	1.0	8	1
11	4lJnCMean4Gd29yHyJG4oQ	Liberty Travel	951 Haddonfiel...	Cherry Hill	NJ	08002	39.9247290...	-75.035161	1.0	6	1
12	Ekvkftk78ZfW5yMaCa0PHg	Kay Jewelers	2020 RT 70 W	Cherry Hill	NJ	08002	39.921491	-75.036943	1.0	6	1
13	WSRvMNRlgjlpIVEGV8E1tA	All My Sons Moving & Stor...	32 Coles Ave	Cherry Hill	NJ	08002	39.942177	-75.0085855	1.0	111	1
14	H9R7iuaBruSBDCtQjrXS_g	Koolkipah.com	208 Mimosa Dr	Cherry Hill	NJ	08003	39.8916619	-74.945318	1.0	7	1
15	rlqIwm0gwg6rHsLdZZHo7A	FedEx Ground	5 Commerce Dr	Barrington	NJ	08007	39.8609281...	-75.041082...	1.0	8	1
16	DCvA43gLeetay_qttR9ABQ	Cham-Pagne Salon & Spa	315 Clements B...	Barrington	NJ	08007	39.8694757	-75.0529608	1.0	5	1
17	HAdHVFYEtcTMWKFPrAak...	Two Guys and One Truck	*null*	Edgewater Pa...	NJ	08010	40.0524453	-74.8969215	1.0	6	1
18	5kaqtDL6kQo8pfncwJWbPw	Bernie's Auto Repairs	990 Lower Lan...	Blackwood	NJ	08012	39.8180025	-75.0801185	1.0	9	1
19	1xS7O2gQHb8HvQn1_JL67w	Niagara Pools & Spas	4170 Route 42	Turnersville	NJ	08012	39.7386775...	-75.041295...	1.0	5	1
20	oD0RY7q7HasCz7JpF9QSOg	Ace Fencing	223 Woodland ...	Blackwood	NJ	08012	39.8179798	-75.0722937	1.0	8	1
21	r6oMjYjLGd6o-EctiaEJxg	Wendy's	4361 Us Rt 130 ...	Burlington	NJ	08016	40.0533507	-74.8941581	1.0	23	1
22	pGGaMFN3_xWm7JlPXfVC...	Checkers	2105 Burlington...	Burlington	NJ	08016	40.049443	-74.838578	1.0	23	0

Categories table

Schema -


categories

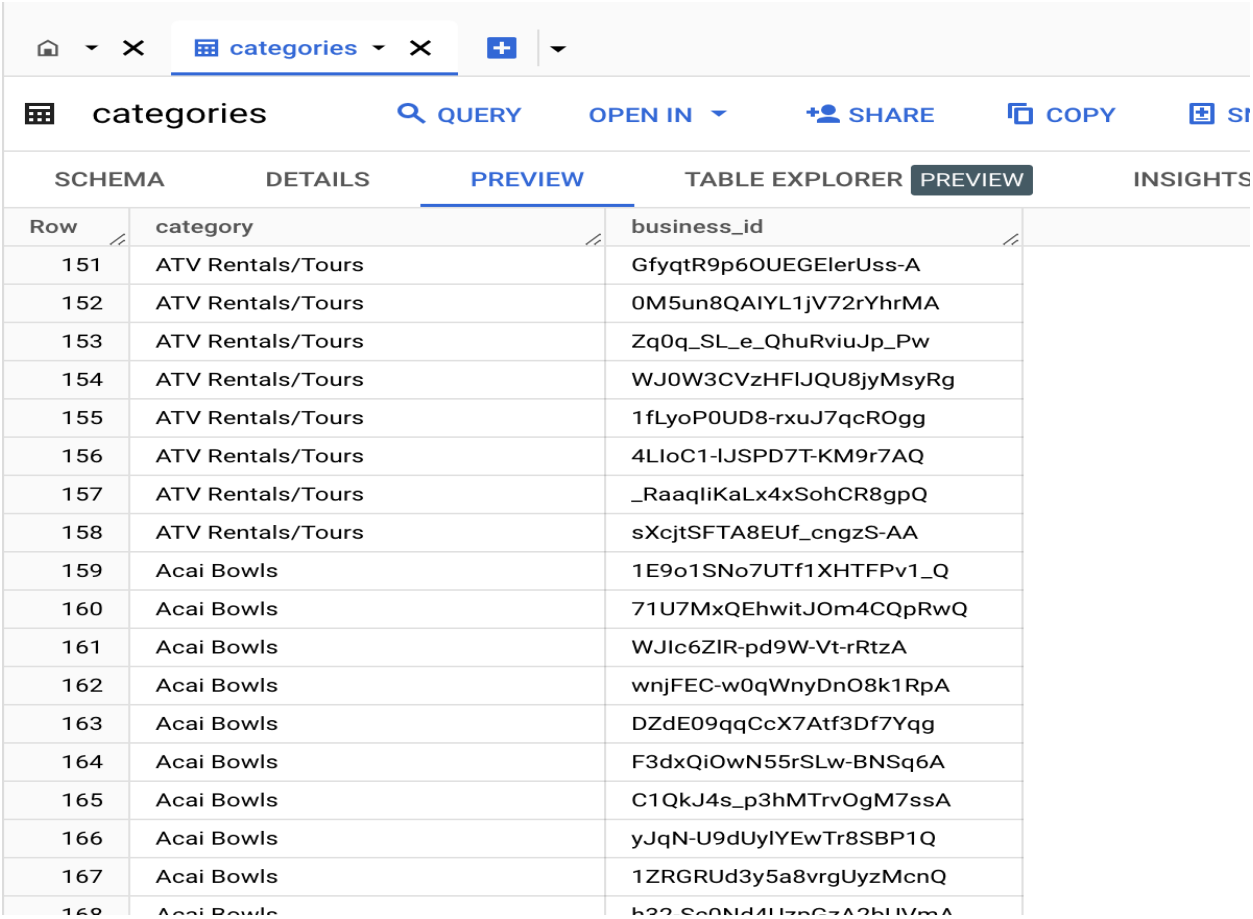
categories QUERY OPEN IN SHARE CO

SCHEMA DETAILS PREVIEW TABLE EXPLORER PREVIEW

Filter Enter property name or value

Field name	Type	Mode	Key	Collation	D
category	STRING	NULLABLE	-	-	-
business_id	STRING	NULLABLE	-	-	-

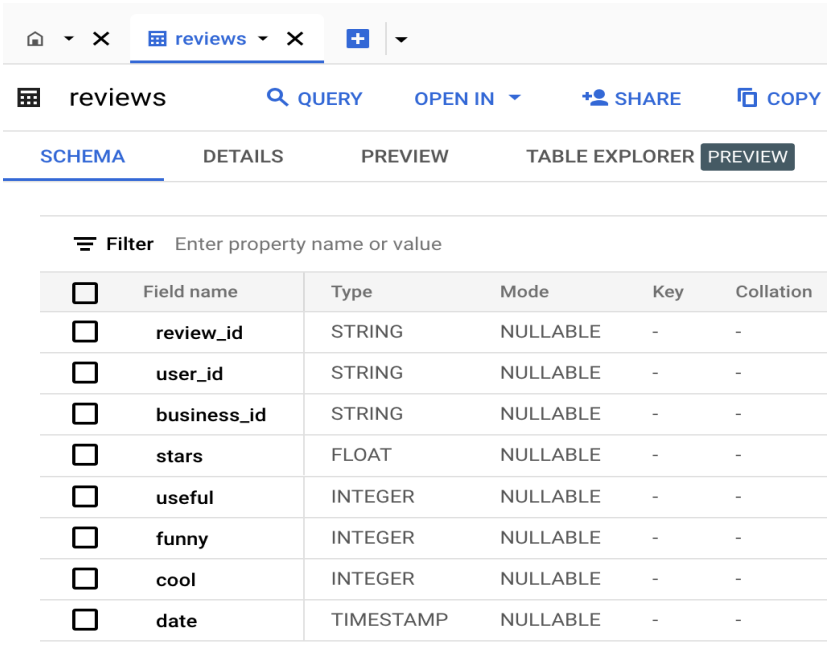
Sample rows –



Row	category	business_id
151	ATV Rentals/Tours	GfyqtR9p6OUEGElerUss-A
152	ATV Rentals/Tours	0M5un8QAIYL1jV72rYhrMA
153	ATV Rentals/Tours	Zq0q_SL_e_QhuRviuJp_Pw
154	ATV Rentals/Tours	WJ0W3CVzHFlJQU8jyMsyRg
155	ATV Rentals/Tours	1fLyoP0UD8-rxuJ7qcROgg
156	ATV Rentals/Tours	4LIoC1-lJSPD7T-KM9r7AQ
157	ATV Rentals/Tours	_RaaqliKaLx4xSohCR8gpQ
158	ATV Rentals/Tours	sXcjtSFTA8EUf_cngzS-AA
159	Acai Bowls	1E9o1SNo7UTf1XHTFPv1_Q
160	Acai Bowls	71U7MxQEhwitJOm4CQpRwQ
161	Acai Bowls	WJIc6ZIR-pd9W-Vt-rRtzA
162	Acai Bowls	wnjFEC-w0qWnyDnO8k1RpA
163	Acai Bowls	DZdE09qqCcX7Atf3Df7Yqg
164	Acai Bowls	F3dxQiOwN55rSLw-BNSq6A
165	Acai Bowls	C1QkJ4s_p3hMTrvOgM7ssA
166	Acai Bowls	yJqN-U9dUylYEwTr8SBP1Q
167	Acai Bowls	1ZRGRUd3y5a8vrgUyzMcnQ

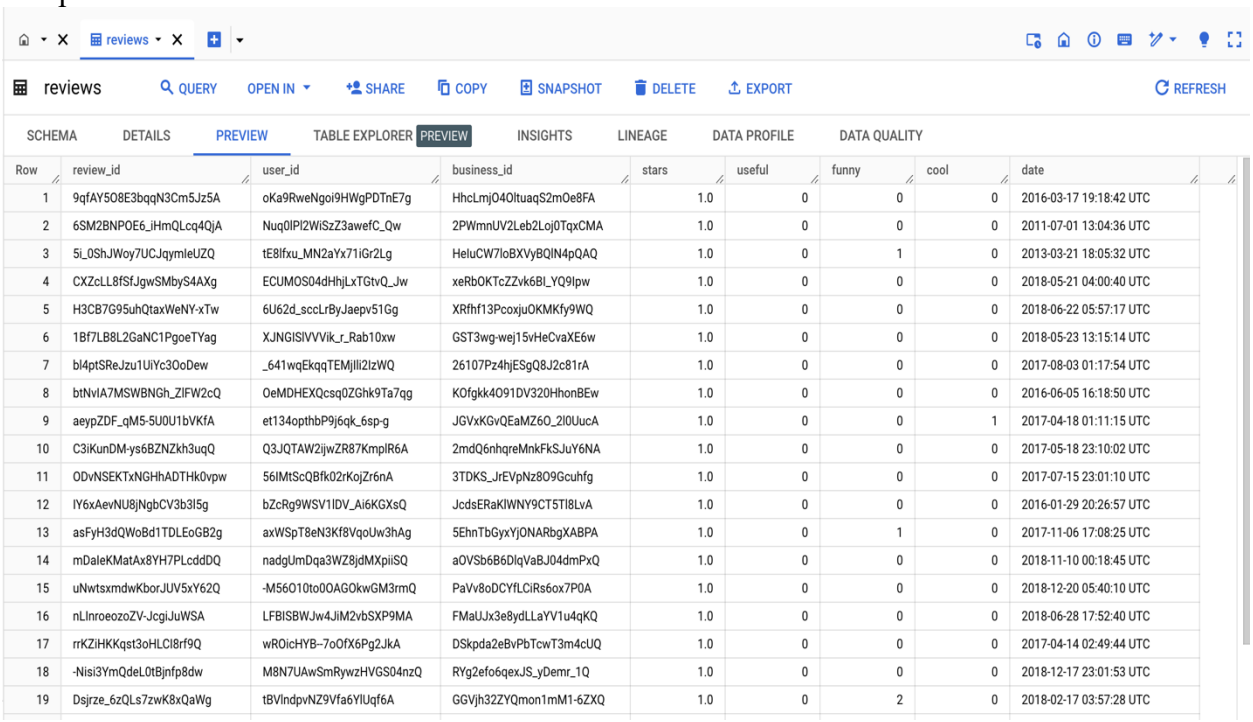
Reviews table –

Schema –



Field name	Type	Mode	Key	Collation
review_id	STRING	NULLABLE	-	-
user_id	STRING	NULLABLE	-	-
business_id	STRING	NULLABLE	-	-
stars	FLOAT	NULLABLE	-	-
useful	INTEGER	NULLABLE	-	-
funny	INTEGER	NULLABLE	-	-
cool	INTEGER	NULLABLE	-	-
date	TIMESTAMP	NULLABLE	-	-

Sample rows -



reviews | QUERY | OPEN IN | SHARE | COPY | SNAPSHOT | DELETE | EXPORT | REFRESH

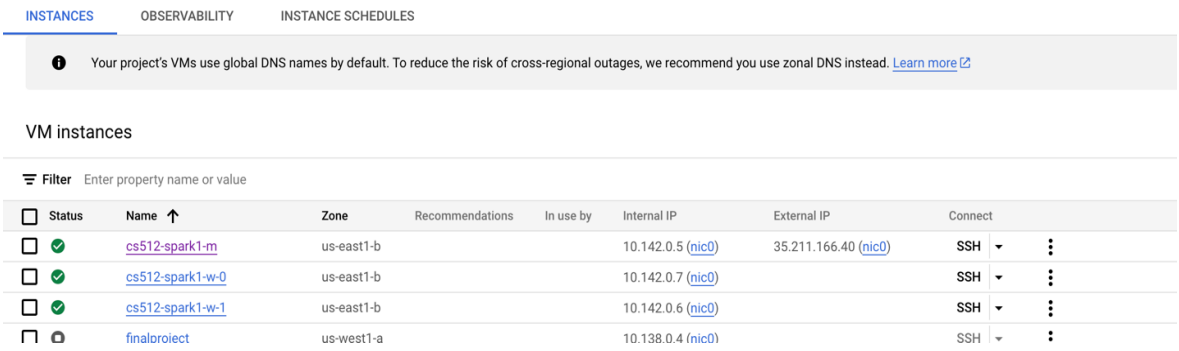
SCHEMA | DETAILS | PREVIEW | TABLE EXPLORER PREVIEW | INSIGHTS | LINEAGE | DATA PROFILE | DATA QUALITY

Row	review_id	user_id	business_id	stars	useful	funny	cool	date
1	9qfAY5O8E3bqqN3Cm5Jz5A	oKa9RweNgoi9HWgPDTnE7g	HhcLmjO40ltuaqS2mOe8FA	1.0	0	0	0	2016-03-17 19:18:42 UTC
2	6SM2BNPOE6_iHmQLcq4QjA	Nuq0lPl2WiSzZ3awefC_Qw	2PWmnUV2Leb2Loj0TqxCMA	1.0	0	0	0	2011-07-01 13:04:36 UTC
3	5i_0ShJWoy7UCJqymleUZQ	tE8lfxu_MN2aYx71iGr2Lg	HeluCW7loBXVyBQIN4pQAQ	1.0	0	1	0	2013-03-21 18:05:32 UTC
4	CXZcLL8fSfJgwSMbyS4AXg	ECUMOS04dHhjLxTGtvQ_Jw	xeRbOKTcZZvk6BI_YQ9Ipw	1.0	0	0	0	2018-05-21 04:00:40 UTC
5	H3CB7G95uhQtaxWeNY-xTw	6U62d_sccLrByJaepv51Gg	XRfhf13PcoxjuOKMKfy9WQ	1.0	0	0	0	2018-06-22 05:57:17 UTC
6	1Bf7LB8L2GaNC1PgoeTYag	XJNGISlVVVik_r_Rab10xw	GST3wg-wej15vHeCvaXE6w	1.0	0	0	0	2018-05-23 13:15:14 UTC
7	bl4ptSReJzu1UiYc3OoDew	_641wqEkqqTEMjlli2lzWQ	26107Pz4hjESgQ8J2c81rA	1.0	0	0	0	2017-08-03 01:17:54 UTC
8	btNvIA7MSWBNGh_ZlFW2cQ	OeMDHEXQcsq0ZGhk9Ta7qg	KOfgkk4O91DV320HhonBEw	1.0	0	0	0	2016-06-05 16:18:50 UTC
9	aeypZDF_qM5-5U0U1bVKfA	et134opthbP9j6qk_6sp-g	JGVxKGvQEaMZ6O_2l0UucA	1.0	0	0	1	2017-04-18 01:11:15 UTC
10	C3iKunDM-ys6BZNZkh3uqQ	Q3JQTAW2ijwZR87KmplR6A	2mdQ6nhqreMnkFkSJuY6NA	1.0	0	0	0	2017-05-18 23:10:02 UTC
11	ODvNSEKTxNGHhADTHk0vpw	56IMtScQBfk02rKojZr6nA	3TDKS_JrEVpNz8O9Gcuhfg	1.0	0	0	0	2017-07-15 23:01:10 UTC
12	IY6xAevNU8jNgbCV3b3l5g	bZcRg9WSV1IDV_Ai6KGXsQ	JcdsERaKlWNY9CT5TI8LvA	1.0	0	0	0	2016-01-29 20:26:57 UTC
13	asFyH3dQWoBd1TDLEoGB2g	axWSpT8eN3Kf8VqoUw3hAg	5EhnTbGyxYjONARbgXABPA	1.0	0	1	0	2017-11-06 17:08:25 UTC
14	mDaleKMatAx8YH7PLcddDQ	nadgUmDqa3WZ8jdMXpiiSQ	aOVSb6B6DlqVaBJ04dmPxQ	1.0	0	0	0	2018-11-10 00:18:45 UTC
15	uNwtsxmdwKborJUV5xY62Q	-M56O10to0OAGOkwGM3rmQ	PaVv8oDCYfLCiRs6ox7P0A	1.0	0	0	0	2018-12-20 05:40:10 UTC
16	nLlnroeozoZV-JcgiJuWSA	LFBISBWJw4JiM2vbSXP9MA	FMaUJx3e8ydLLaYV1u4qKQ	1.0	0	0	0	2018-06-28 17:52:40 UTC
17	rrKZiHKKqst3oHLCl8rf9Q	wROicHYB--7oOfX6Pg2JkA	DSkpda2eBvPbTcwT3m4cUQ	1.0	0	0	0	2017-04-14 02:49:44 UTC
18	-Nisi3YmQdeL0tBjnfp8dw	M8N7UAwSmRywzHVGS04nzQ	RYg2efo6qexJS_yDemr_1Q	1.0	0	0	0	2018-12-17 23:01:53 UTC
19	Dsjrze_6zQLs7zwK8xQaWg	tBVlndpvNZ9Vfa6VlUqf6A	GGVjh32ZYQmon1mM1-6ZXQ	1.0	0	2	0	2018-02-17 03:57:28 UTC

4. Creating a spark job –

I used the Window_spark_planes_0.py from our previous assignments and modified it to match my requirements. I uploaded it to the Google bucket and created a spark job in Dataproc to execute it. Here are the snippet. The spark job also creates a table in bigquery named – “business_state_reviews”

I used the Spark VM instances we created earlier –



INSTANCES | OBSERVABILITY | INSTANCE SCHEDULES

Your project's VMs use global DNS names by default. To reduce the risk of cross-regional outages, we recommend you use zonal DNS instead. Learn more

VM instances

Filter Enter property name or value

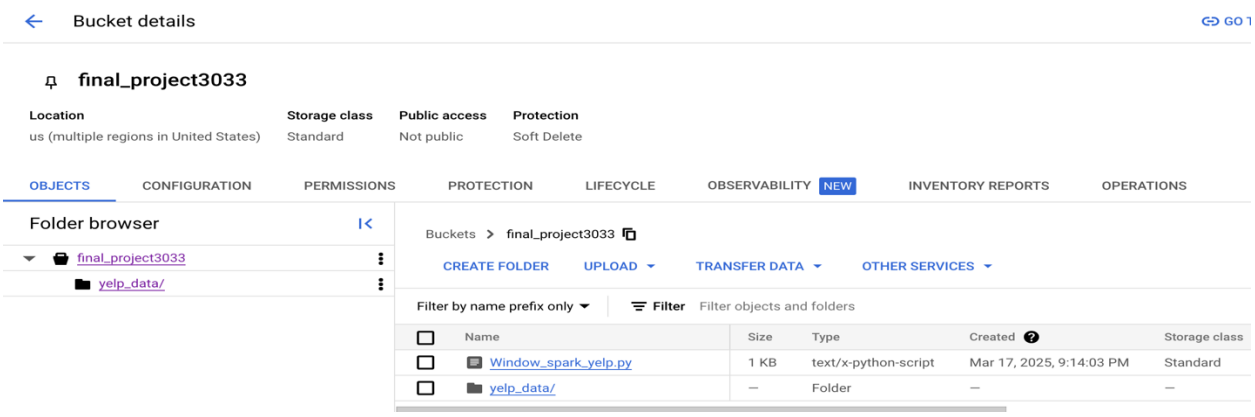
Status	Name	Zone	Recommendations	In use by	Internal IP	External IP	Connect
✓	cs512-spark1-m	us-east1-b			10.142.0.5 (nic0)	35.211.166.40 (nic0)	SSH
✓	cs512-spark1-w-0	us-east1-b			10.142.0.7 (nic0)		SSH
✓	cs512-spark1-w-1	us-east1-b			10.142.0.6 (nic0)		SSH
	finalproject	us-west1-a			10.138.0.4 (nic0)		SSH

1. **Python script –**

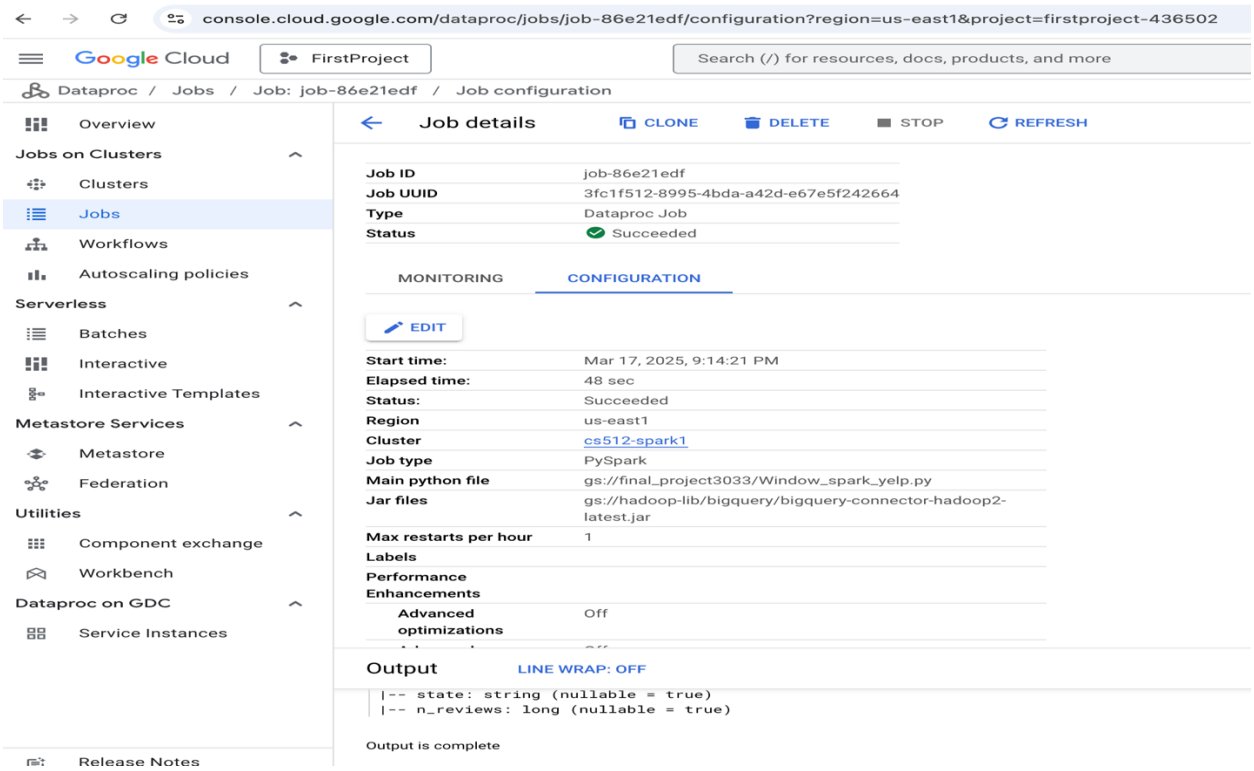
Window_spark_yelp.py

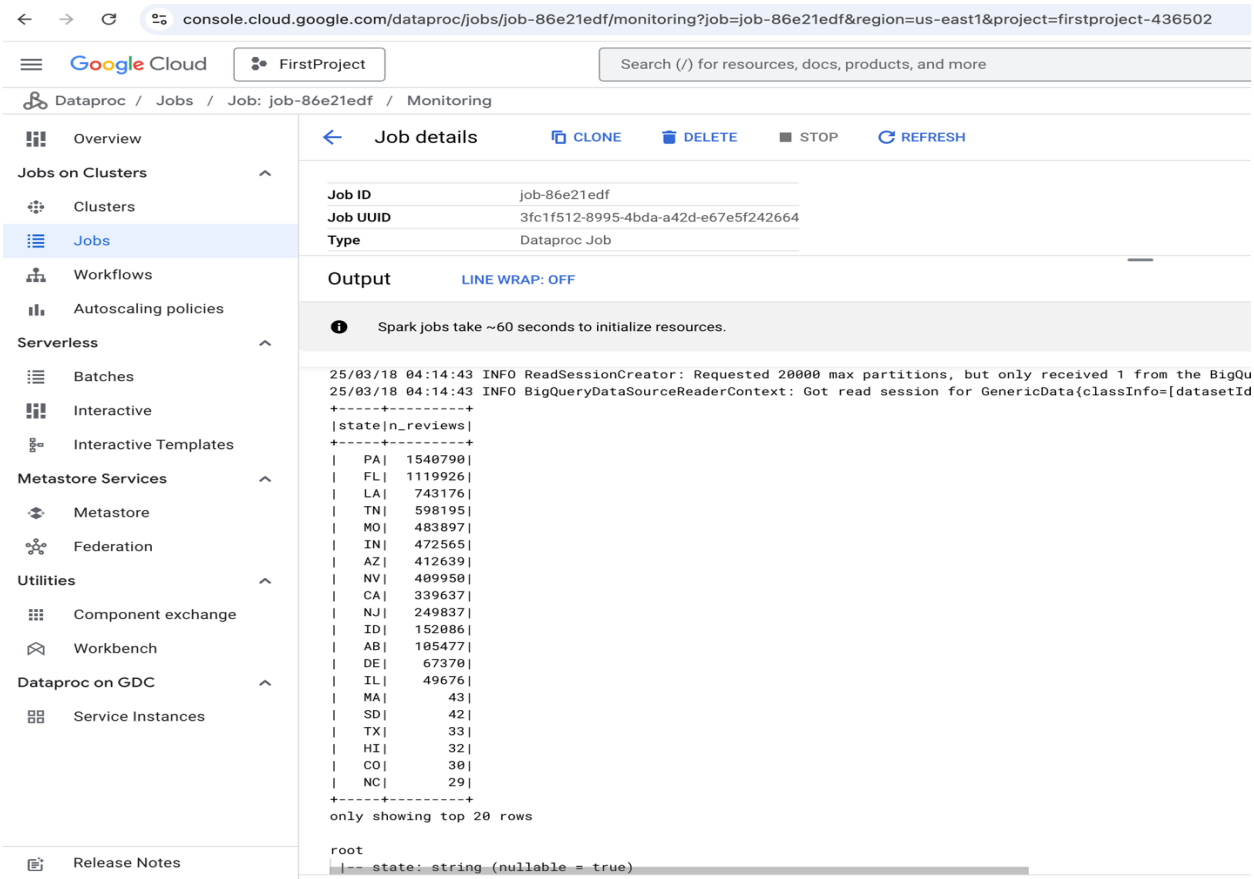
```python
#!/usr/bin/python
from pyspark.sql import SparkSession

spark = SparkSession \
            .builder \
            .master('yarn') \
            .appName('Yelp') \
            .getOrCreate()

# Use the Cloud Storage bucket for temporary BigQuery export data used
# by the connector.
bucket = "final_project3033"
spark.conf.set('temporaryGcsBucket', bucket)

# Load data from BigQuery.
businesses = spark.read.format('bigquery') \
        .option('table', 'firstproject-436502.yelp.businesses') \
        .load()
businesses.createOrReplaceTempView('businesses')

# Perform word count.
businesses_query = spark.sql(
    'SELECT state AS state, SUM(review_count) as n_reviews FROM businesses GROUP BY state ORDER BY n_reviews DESC'
)
businesses_query.show()
businesses_query.printSchema()

# Saving the data to BigQuery
businesses_query.write.format('bigquery') \
    .option('table', 'firstproject-436502.yelp.business_state_reviews') \
    .mode('overwrite') \
    .save()

spark.stop()

```

2. **Python file in bucket –**

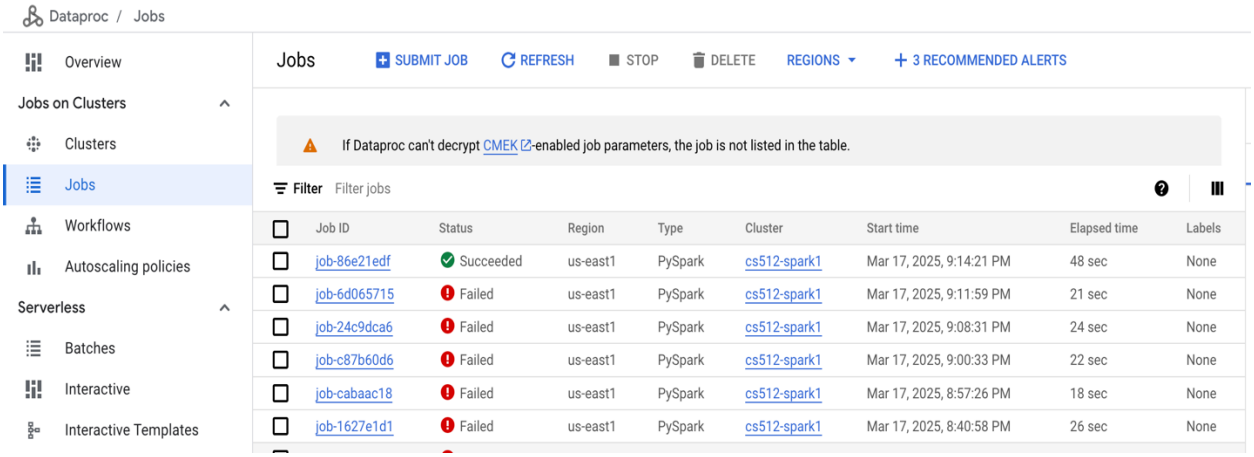


3. Job created and its output –

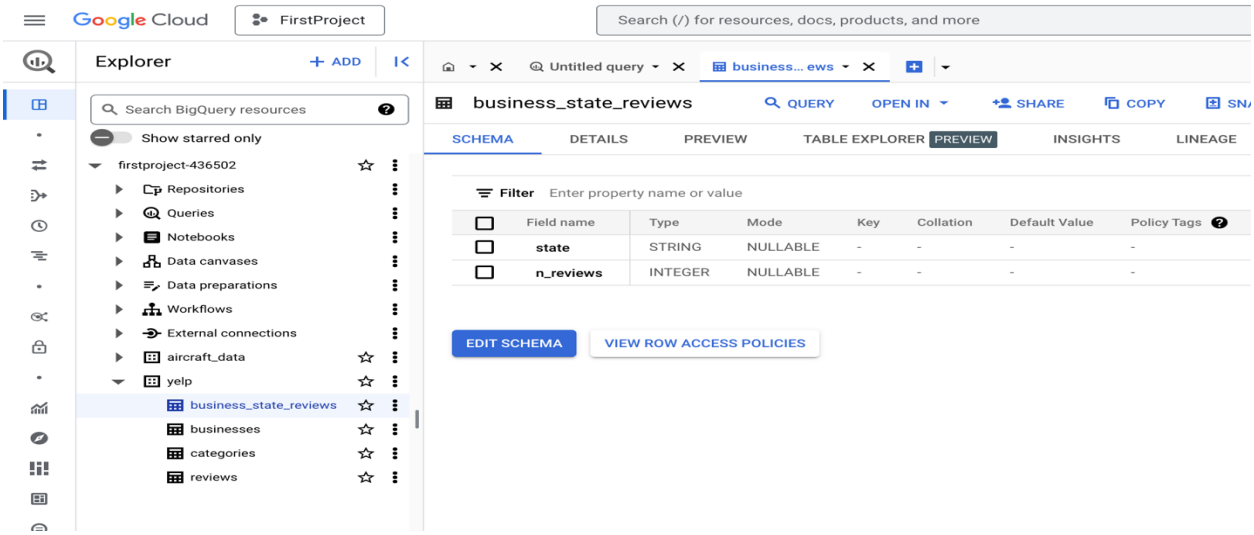




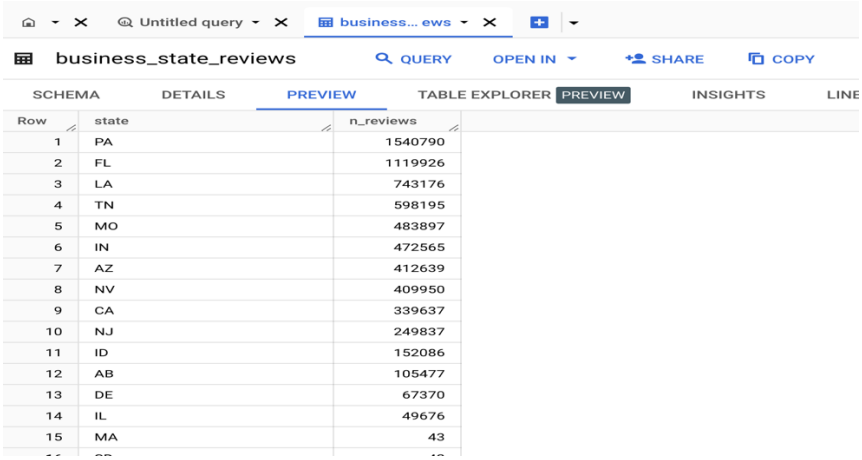
As shown below, this was a successful job and it took 48 secs and many others before that failed.



4. Its output in BigQuery –



Sample rows –



Row	state	n_reviews
1	PA	1540790
2	FL	1119926
3	LA	743176
4	TN	598195
5	MO	483897
6	IN	472565
7	AZ	412639
8	NV	409950
9	CA	339637
10	NJ	249837
11	ID	152086
12	AB	105477
13	DE	67370
14	IL	49676
15	MA	43

Group contribution –

I confirm that I completed this assignment independently. I would like to extend my sincere gratitude to the GTA for their meticulous feedback, which significantly improved my understanding. I am also deeply appreciative of the Professor for creating such an insightful and relevant course. It has been one of the most valuable learning experiences in my graduate program, providing me with practical knowledge directly applicable to the industry.