

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 –

```
{} yelp_academic_dataset_business.json X :  
  
"RestaurantsReservations": "False", "DogsAllowed": "False", "ByAppointmentOnly": "False"}, "categories": "Department Stores, Shopping, Fashion, Home & Garden, Electronics, Furniture Stores", "hours": {"Monday": "8:0-22:0", "Tuesday": "8:0-22:0", "Wednesday": "8:0-22:0", "Thursday": "8:0-22:0", "Friday": "8:0-23:0", "Saturday": "8:0-23:0", "Sunday": "8:0-22:0"}}  
  
{ "business_id": "MTSW4McQd7CbVtyjqoe9mw", "name": "St Honore Pastries", "address": "935 Race St", "city": "Philadelphia", "state": "PA", "postal_code": "19107", "latitude": 39.9555052, "longitude": -75.1555641, "stars": 4.0, "review_count": 80, "is_open": 1, "attributes": {"RestaurantsDelivery": "False", "OutdoorSeating": "False", "BusinessAcceptsCreditCards": "False", "BusinessParking": {"garage": False, "street": True, "validated": False, "lot": False, "valet": False}, "BikeParking": "True", "RestaurantsPriceRange2": "1", "RestaurantsTakeOut": "True", "ByAppointmentOnly": "False", "WiFi": "u'free'", "Alcohol": "u'none'", "Caters": "True"}, "categories": "Restaurants, Food, Bubble Tea, Coffee & Tea, Bakeries", "hours": {"Monday": "7:0-20:0", "Tuesday": "7:0-20:0", "Wednesday": "7:0-20:0", "Thursday": "7:0-20:0", "Friday": "7:0-21:0", "Saturday": "7:0-21:0", "Sunday": "7:0-21:0"}}  
  
{ "business_id": "mWMc6_wTdE0EUBKIGXDvfa", "name": "Perkiomen Valley Brewery", "address": "101 Walnut St", "city": "Green Lane", "state": "PA", "postal_code": "18054", "latitude": 40.3381827, "longitude": -75.4716585, "stars": 4.5, "review_count": 13, "is_open": 1, "attributes": {"BusinessAcceptsCreditCards": "True", "WheelchairAccessible": "True", "RestaurantsTakeOut": "True", "BusinessParking": {"garage": None, "street": None, "validated": None, "lot": True, "valet": False}, "BikeParking": "True", "GoodForKids": "True", "Caters": "False"}, "categories": "Brewpubs, Breweries, Food", "hours": {"Wednesday": "14:0-22:0", "Thursday": "16:0-22:0", "Friday": "12:0-22:0", "Saturday": "12:0-22:0", "Sunday": "12:0-18:0"}}
```

#### 2. yelp\_academic\_dataset\_review.json

```
{} yelp_academic_dataset_review.json X :  
  
{"review_id": "KU_05udG6zpx0g-VcAEodg", "user_id": "mh_-eMz6K5RLWhZyISBhwA", "business_id": "XQfwVwDr-v0ZS3_CbbE5Xw", "stars": 3.0, "useful": 0, "funny": 0, "cool": 0, "text": "If you decide to eat here, just be aware it is going to take about 2 hours from beginning to end. We have tried it multiple times, because I want to like it! I have been to it other locations in NJ and never had a bad experience. \n\nThe food is good, but it takes a very long time to come out. The waitstaff is very young, but usually pleasant. We have just had too many experiences where we spent way too long waiting. We usually opt for another diner or restaurant on the weekends, in order to be done quicker.", "date": "2018-07-07 22:09:11"}  
  
{"review_id": "BiTunyQ73aT9WBnpR9DZGw", "user_id": "OyoGAe70Kpv6SyGZT5g77Q", "business_id": "7ATYjTiG3jUlt4UM3IypQ", "stars": 5.0, "useful": 1, "funny": 0, "cool": 1, "text": "I've taken a lot of spin classes over the years, and nothing compares to the classes at Body Cycle. From the nice, clean space and amazing bikes, to the welcoming and motivating instructors, every class is a top notch work out.\n\nFor anyone who struggles to fit workouts in, the online scheduling system makes it easy to plan ahead (and there's no need to line up way in advanced like many gyms make you do).\n\nThere is no way I can write this review without giving Russell, the owner of Body Cycle, a shout out. Russell's passion for fitness and cycling is so evident, as is his desire for all of his clients to succeed. He is always dropping in to classes to check in\\provide encouragement, and is open to ideas and recommendations from anyone. Russell always wears a smile on his face, even when he's kicking your butt in class!", "date": "2012-01-03 15:28:18"}  
  
{"review_id": "saUsX_vimxRLCVr67Z4Jig", "user_id": "8g_iMtfSiwikVnbP2etR0A", "business_id": "YjUWPpI6HXG530lwp-fb2A", "stars": 3.0, "useful": 0, "funny": 0, "cool": 0, "text": "Family diner. Had the buffet. Eclectic assortment: a large chicken leg, fried jalape\u00f1o, tamale, two rolled grape leaves, fresh melon. All good. Lots of Mexican choices there. Also has a menu with breakfast served all day long. Friendly, attentive staff. Good place for a casual relaxed meal with no expectations. Next to the Clarion Hotel.", "date": "2014-02-05 20:30:30"}
```

# 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 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

```
businesses.csv
⚠ The file size (17.16 MB) exceeds the configured limit (2.56 MB). Code insight features are not available.

1 business_id,name,address,city,state,postal_code,latitude,longitude,stars,review_count,is_open
2 Pns2l4eNsf08kk83dixA6A,"Abby Rappoport, LAC, CMQ","1616 Chapala St, Ste 2",Santa Barbara,CA,93101,34.4266787,-119.7111968,5.0,7,0
3 mpf3x-BjTdTEA3yCzrAYPw,The UPS Store,87 Grasso Plaza Shopping Center,Affton,MO,63123,38.551126,-90.335695,3.0,15,1
4 tUFrWirkKiKi_TAnsVWINQQ,Target,5255 E Broadway Blvd,Tucson,AZ,85711,32.223236,-110.880452,3.5,22,0
5 MTSW4McQd7CbVtyjqoe9mw,St Honore Pastries,935 Race St,Philadelphia,PA,19107,39.9555052,-75.1555641,4.0,80,1
6 mWMc6_wTde0EUBKIGXDVFa,Perkiomen Valley Brewery,101 Walnut St,Green Lane,PA,18054,40.3381827,-75.4716585,4.5,13,1
7 CF33F8-E6oudUQ46HnavjQ,Sonic Drive-In,615 S Main St,Ashland City,TN,37015,36.269593,-87.058943,2.0,6,1
8 n_8UpQx1hsNbnPUSlodU8w,Famous Footwear,"8522 Eager Road, Dierbergs Brentwood Point",Brentwood,MO,63144,38.627695,-90.340465,2.5,13,1
9 qkRM_2X51Yqxk3btlwAQIg,Temple Beth-El,400 Pasadena Ave S,St. Petersburg,FL,33707,27.76659,-82.732983,3.5,5,1
10 k0hlBqXX-Bt0vf1op7Jr1w,Tsevi's Pub And Grill,8025 Mackenzie Rd,Affton,MO,63123,38.5651648,-90.5210868,3.0,19,0
11 bBDD EgkFA10tx9Lfe7BZUQ,Sonic Drive-In,2312 Dickerson Pike,Nashville,TN,37207,36.2081024,-86.7681696,1.5,10,1
12 UJsufbvfyfONHeWdvAHkjA,Marshalls,21705 Village Lakes Sc Dr,Land O' Lakes,FL,34639,28.1904587953,-82.4573802199,3.5,6,1
```

## 2. reviews.csv

review_id	user_id	business_id	stars	useful	funny	cool	date
KU_05udG6zpx0g-VcAEodg, mh_-eMZ6K5RLWhZyISBhwA, XQfwVwDr-v0ZS3_CbbE5Xw, 3.0, 0, 0, 0, 2018-07-07 22:09:11	BiTunyQ73aT9WBnpR9DZGw, OyoGAe70Kpv6SyGZT5g77Q, 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, kxX2S0es4o-D3ZQBkiMRfa, 5.0, 1, 0, 1, 2015-01-04 00:01:03	Sx8TM0WLNUJBWer-0pcmoA, bcjbaE6dDog4jkNY91ncLQ, e4Vwtrqf-wpJfwsqvgdxQ, 4.0, 1, 0, 1, 2017-01-14 20:54:15	JrIxLS1TzJ-iCu79u140cQ, eUta8W_HdHMXPzLBBZhL1A, 04UD14gamNjLY0IDYVhHJg, 1.0, 1, 2, 1, 2015-09-23 23:10:31	6AxgBCNX_PNT0xmbRSwcKQ, r3zeYsv1XF BRA4dJpL78cw, gmjsEdUsKpj9Xxu6pdjh0g, 5.0, 0, 2, 0, 2015-01-03 23:21:18	_ZeMknuYdLQcUqng_Im3yg, yffFzsLmaWF2d4Sr0UNbBgg, LHSTtnW3YHCeUkRDGyJ0yw, 5.0, 2, 0, 0, 2015-08-07 02:29:16
ZKvDG2sBvHVdF5oBNU0pAQ, wSTuiTk-sKNdcFyprzAjq, B5XSoSG3SfvQGtKEGQ1tSQ, 3.0, 1, 1, 0, 2016-03-30 22:46:33	pUyc0fUwM8vqX7KjRRhUEA, 59MxRhNVhU9MYndMkz0wtw, gebiRewfieSdt17PTW6Zg, 3.0, 0, 0, 0, 2016-07-25 07:31:06	rGQRf8UafX70TLMN N19I8A, 1WHRWwQmZ0ZDAhp2Qyny4g, uMvVYRgGNXf5boolA9HXTw, 5.0, 2, 0, 0, 2015-06-21 14:48:06	l3Wk_mvAog6XANIuGQ9C7Q, ZbqSHbgCjzVAqaa7NKWn5A, EQ-TZ2eeD_E0BHuviaeG5Q, 4.0, 0, 0, 0, 2015-08-19 14:31:45	XW_LfMv0fV21l9c6xQd_lw, 90AtfnWag-ajVxRbUTGIyg, lj-E32x9_FA7GmUrBGBEWg, 4.0, 0, 0, 0, 2014-06-27 22:44:01	8JFGBuHMo iNDyfcxuWNtrA, sm0v0ajNG0ls4Pq7d8g4JQ, RZtGWDLCAtuipwaZ-UfjmQ, 4.0, 0, 0, 0, 2009-10-14 19:57:14	UBp0zWyH60Hmw6Fsasei7w, 4Uh27DgGzsp6PqrH913giQ, otQS34_MymijPTdNB0BdCw, 4.0, 0, 2, 0, 2011-10-27 17:12:05	

## 3. categories.csv

business_id	category
Pns2L4eNsf08kk83dixA6A	Doctors
Pns2L4eNsf08kk83dixA6A	Traditional Chinese Medicine
Pns2L4eNsf08kk83dixA6A	Naturopathic/Holistic
Pns2L4eNsf08kk83dixA6A	Acupuncture
Pns2L4eNsf08kk83dixA6A	Health & Medical
Pns2L4eNsf08kk83dixA6A	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_wTdE0EUBKIGXDVF A	Brewpubs
mWMc6_wTdE0EUBKIGXDVF A	Breweries
mWMc6_wTdE0EUBKIGXDVF A	Food

## Snippet of code wrote to scrub the files –

### 1. business.py

```
business.py ×

1 import json
2 import re
3 import csv
4
5 usage
6 def correcting_business_json(raw_file):
7     with open(raw_file, 'r') as rfile:
8         raw_data = rfile.read()
9         fixed_json = re.sub(pattern: '}\\n*{"business_id":', repl: '}', raw_data.strip())
10        fixed_json = '{\\n"businesses": [' + fixed_json + ']\\n}'
11        json_data = json.loads(fixed_json)
12
13        for business in json_data["businesses"]:
14            business.pop("attributes", None)
15            business.pop("hours", None)
16            if business["categories"]:
17                business["categories"] = list(business["categories"].split(", "))
18            else: []
19
20            with open("initial_business.json", "w") as file:
21                json.dump(json_data, file, indent=4)
22
23 correcting_business_json('yelp_academic_dataset_business.json')
```

```
business.py ×

22 correcting_business_json('yelp_academic_dataset_business.json')
23
24 usage
25 def json_to_csv(json_file):
26     with open(json_file, 'r') as jfile:
27         content = json.load(jfile)
28         list_of_businesses = list(content.values())
29
30         for item in list_of_businesses[0]:
31             fieldnames = item.keys()
32
33             with open('intermediate_business.csv', 'w', newline='') as fp:
34                 writer = csv.DictWriter(fp, fieldnames)
35                 writer.writeheader()
36
37                 for item in list_of_businesses[0]:
38                     writer.writerow(item)
39
40 json_to_csv('initial_business.json')
```

## 2. reviews.py

```
reviews.py ×

2 import re
3 import csv
4
5
6 1 usage
7 def extracting_reviews_json(raw_file):
8     with open(raw_file, 'r') as rfile:
9         raw_data = rfile.read()
10    fixed_json = re.sub(pattern: '}\n*{"review_id": "', repl: '},\n{"review_id": "', raw_data.strip())
11    fixed_json = '{\n"reviews": [' + fixed_json + ']\n}'
12
13    json_data = json.loads(fixed_json)
14    for review in json_data["reviews"]:
15        review.pop("text", None)
16
17    with open('initial_reviews.json', 'w') as jfile:
18        json.dump(json_data, jfile, indent=4)
19
20    print("Initial json file created")
extracting_reviews_json('yelp_academic_dataset_review.json')
```

```
reviews.py ×

18
19     print("Initial json file created")
extracting_reviews_json('yelp_academic_dataset_review.json')
20
21
22 1 usage
23 def json_to_csv(json_file):
24     with open(json_file, 'r') as jfile:
25         content = json.load(jfile)
26         list_of_reviews = list(content.values())
27
28         for item in list_of_reviews[0]:
29             fieldnames = item.keys()
30
31             with open('intermediate_reviews.csv', 'w', newline='') as fp:
32                 writer = csv.DictWriter(fp, fieldnames)
33                 writer.writeheader()
34
35                 for item in list_of_reviews[0]:
36                     writer.writerow(item)
37 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 sqlite –

```
json_to_sql.py >
1 > import ...
3
4 business_df = pd.read_json('initial_business.json')
5 reviews_df = pd.read_json('initial_reviews.json')
6
7 conn = sqlite3.connect('yelp2.db')
8 c = conn.cursor()
9
10 c.execute('''
11 CREATE TABLE IF NOT EXISTS businesses (
12     business_id TEXT PRIMARY KEY,
13     name TEXT,
14     address TEXT,
15     city TEXT,
16     state TEXT,
17     postal_code TEXT,
18     latitude REAL,
19     longitude REAL,
20     stars REAL,
21     review_count INTEGER,
22     is_open INTEGER
23 );
24 ''')
25
26 c.execute('''
27 CREATE TABLE IF NOT EXISTS categories (
28     business_id TEXT,
29     category TEXT
30 );
31 ...''')

json_to_sql.py >
32
33 c.execute('''
34 CREATE TABLE IF NOT EXISTS reviews (
35     review_id TEXT PRIMARY KEY,
36     user_id TEXT,
37     business_id TEXT,
38     stars REAL,
39     useful INT,
40     funny INT,
41     cool INT,
42     date DATETIME,
43     FOREIGN KEY(business_id) REFERENCES businesses(business_id)
44 );
45 ''')
46
47 category_data = business_df[['business_id','categories']]
48 exploded_df = category_data.explode('categories', ignore_index=True)
49
50 for index, business in business_df.iterrows():
51     c.execute(_sql'''
52     INSERT INTO businesses (business_id, name, address, city, state, postal_code, latitude, longitude,
53     stars, review_count, is_open)
54     VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
55     ''', _parameters_(business['business_id'], business['name'], business['address'], business['city'],
56     business['state'],
57     business['postal_code'], business['latitude'], business['longitude'], business['stars'],
58     business['review_count'], business['is_open']))
```

```
json_to_sql.py ×

53
54     ''', _parameters: (business['business_id'], business['name'], business['address'], business['phone'],
55     business['state'],
56     business['postal_code'], business['latitude'], business['longitude'], business['stars'],
57     business['review_count'], business['is_open']))
58
59     for index, category in exploded_df.iterrows():
60         c.execute(_sql: '''
61             INSERT INTO categories (business_id, category)
62             VALUES (?, ?)
63             ''', _parameters: (category['business_id'], category['categories']))
64
65     for index, review in reviews_df.iterrows():
66         review_date = review['date'].strftime('%Y-%m-%d %H:%M:%S')
67
68         c.execute(_sql: '''
69             INSERT INTO reviews (review_id, user_id, business_id, stars, useful, funny, cool, date)
70             VALUES (?, ?, ?, ?, ?, ?, ?, ?)
71             ''', _parameters: (review['review_id'], review['user_id'], review['business_id'], review['stars'],
72             review['useful'],
73             review['funny'], review['cool'], review_date))
74     conn.commit()
75     conn.close()
```

### *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

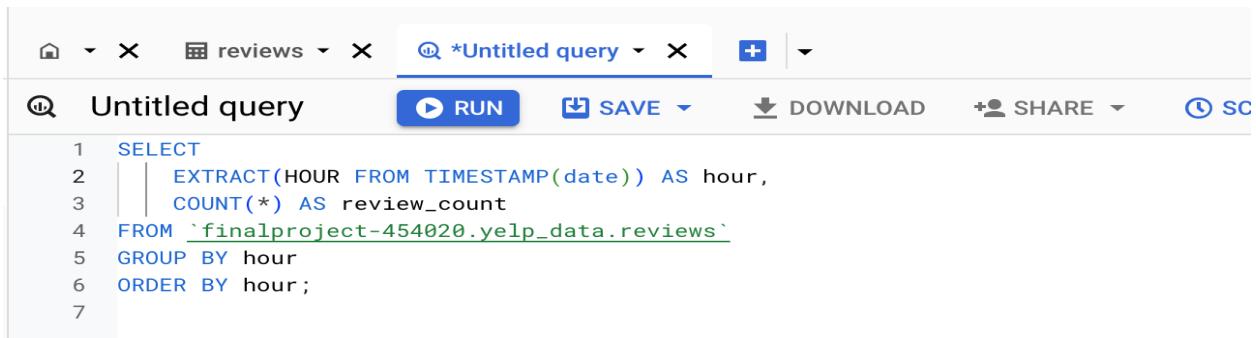
**Process to answer the question** – After loading the reviews table into the Google BigQuery, we will write queries for each cases. As we want to know the volume of the reviews for each case, we will extract the required time from the date column in the table. We will then select the count of number 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

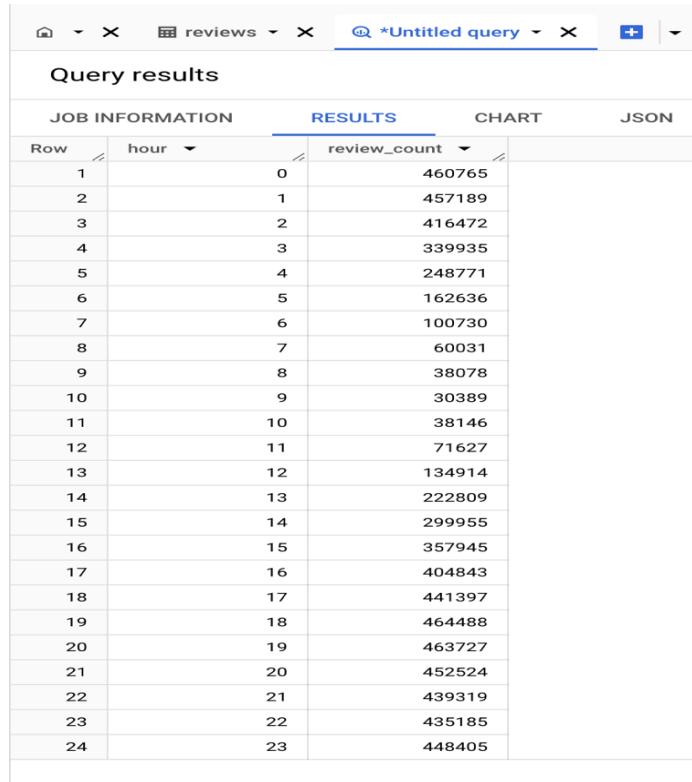
Query -



```
reviews *Untitled query
@ Untitled query
  RUN SAVE DOWNLOAD SHARE SC

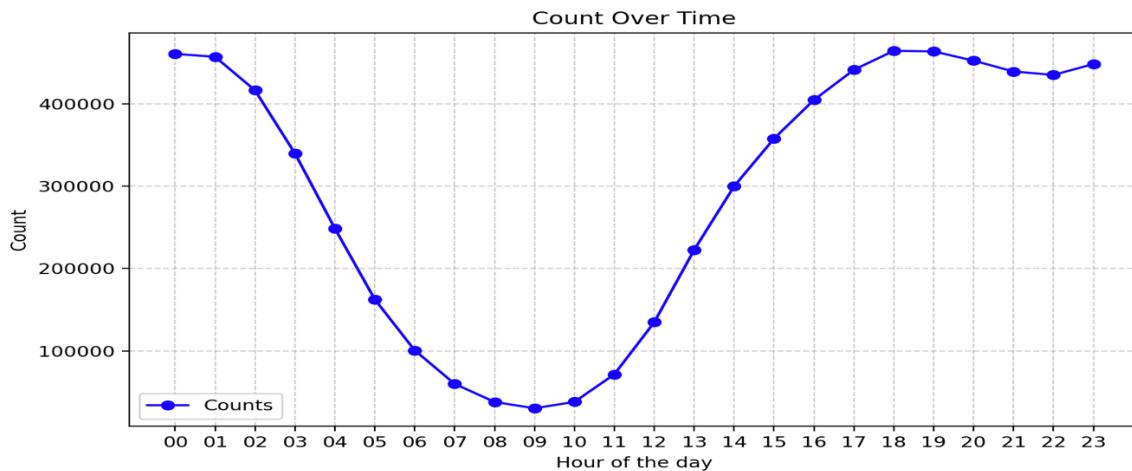
1 SELECT
2   EXTRACT(HOUR FROM TIMESTAMP(date)) AS hour,
3   COUNT(*) AS review_count
4 FROM `finalproject-454020.yelp_data.reviews`
5 GROUP BY hour
6 ORDER BY hour;
7
```

The query results were as follows -



Query results			
JOB INFORMATION		RESULTS	CHART
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



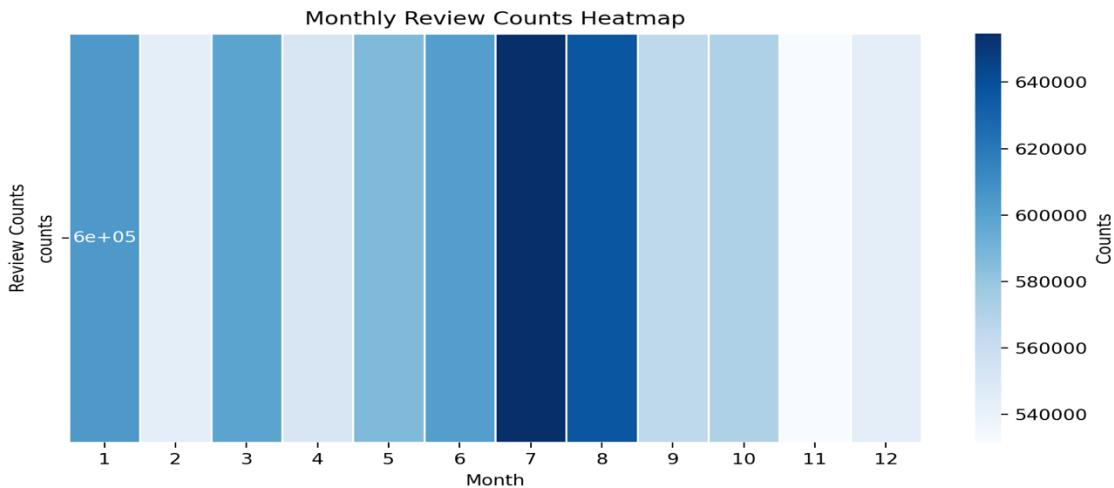
## Data analysis on monthly pattern

```
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 –

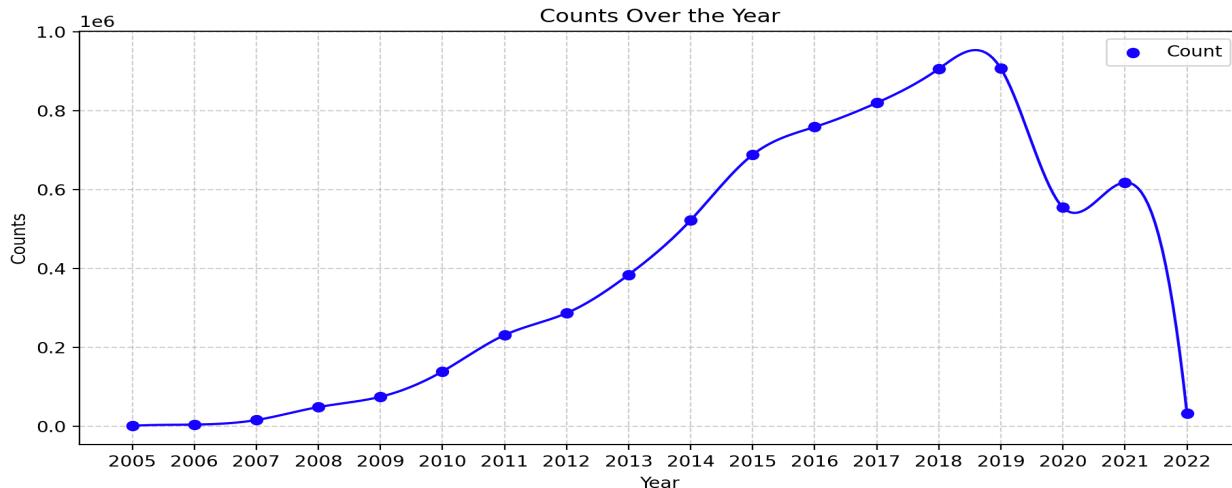
Screenshot of a database query interface showing an "Untitled query" with the following SQL code:

```
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 –

The screenshot shows a database query editor interface. At the top, there's a toolbar with icons for file operations (New, Open, Save, etc.), a search bar, and tabs for 'Untitled query'. Below the toolbar, the main area has a title 'Untitled query' and several buttons: RUN, SAVE, DOWNLOAD, SHARE, SCHEDULE, OPEN IN, and MORE. The code area contains a numbered query:

```
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;
```

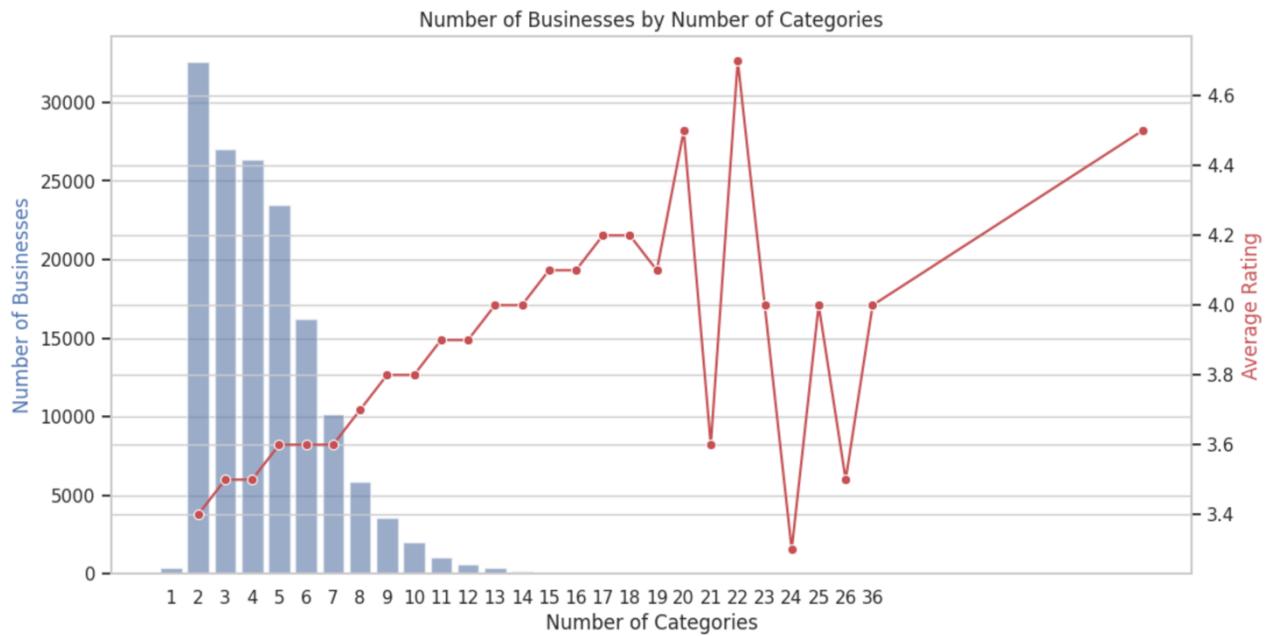
Below the code, there's a section titled 'Query results' with a table. The table has columns: Row, num\_categories, num\_businesses, min\_rating, max\_rating, and avg\_rating. The data is as follows:

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	26254	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

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	26254	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 customer satisfaction. The trend of increasing average ratings may be disrupted by external factors.

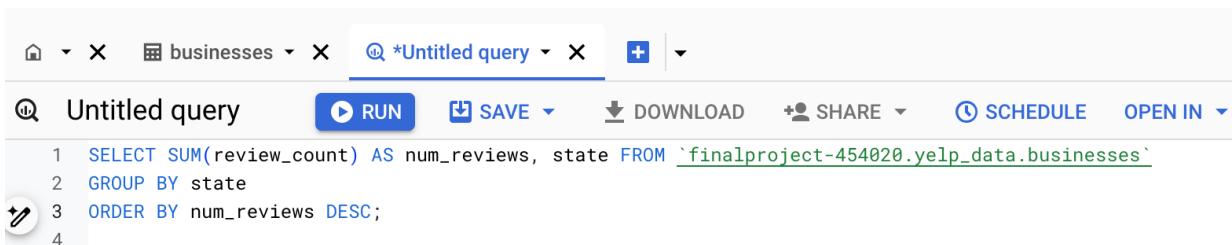
## Question 3 – Number of reviews and states of US.

**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.

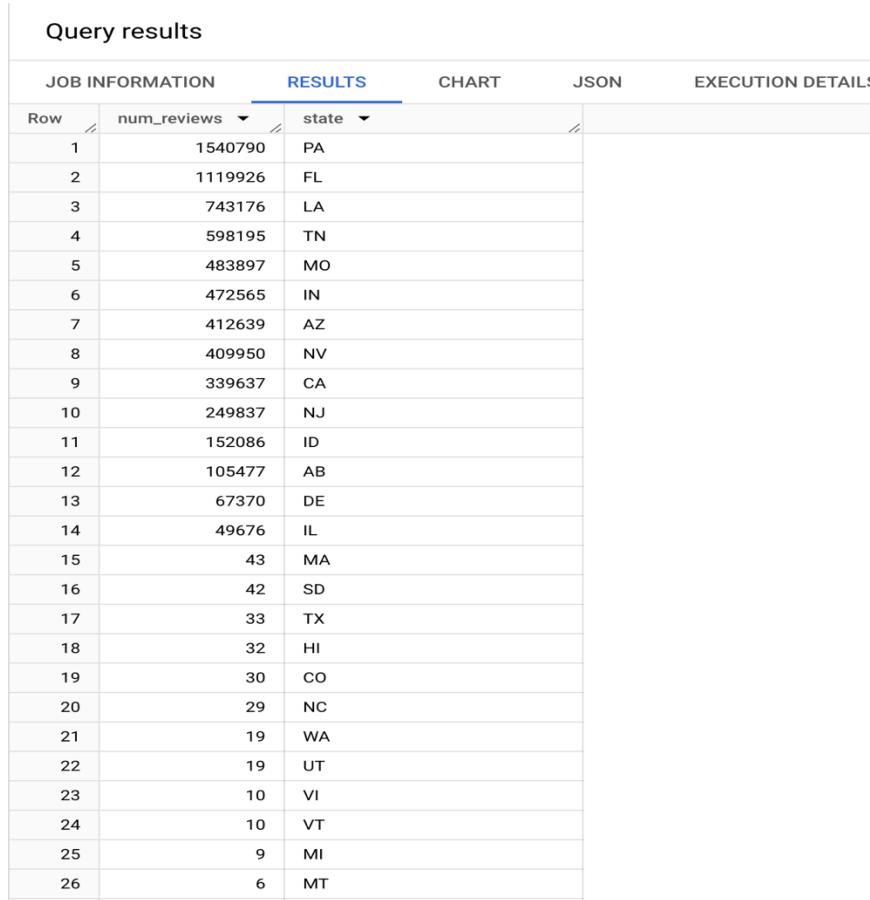
## Data Analysis

### Query -



```
④ Untitled query
RUN SAVE DOWNLOAD SHARE SCHEDULE OPEN IN
1 SELECT SUM(review_count) AS num_reviews, state FROM `finalproject-454020.yelp_data.businesses`
2 GROUP BY state
3 ORDER BY num_reviews DESC;
4
```

### Query results -



Query results		
JOB INFORMATION		RESULTS
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 -

Job details

CLONE DELETE STOP REFRESH

Job ID	job-86e21edf
Job UUID	3fc1f512-8995-4bda-a42d-e67e5f242664
Type	Dataproc Job
Status	Succeeded

MONITORING      CONFIGURATION

Results -

Google Cloud FirstProject

Dataproc / Jobs / Job: job-86e21edf / Monitoring

Search (/) for resources, docs, products, and more

Job details

CLONE DELETE STOP REFRESH

Output LINE WRAP: OFF

Spark jobs take ~60 seconds to initialize resources.

```
25/03/18 04:14:43 INFO ReadSessionCreator: Read session:{"readSessionName":"projects/firstproject-436502/locations/us/sessions/CAISDE1FdGtVRWFrSmleRoCanIaAmp"}, "readSessionCreationStartTime": "25/03/18 04:14:43", "readSessionCreationEndTime": "25/03/18 04:14:43", "readSessionCreationDuration": 0, "readSessionCreationError": null}
```

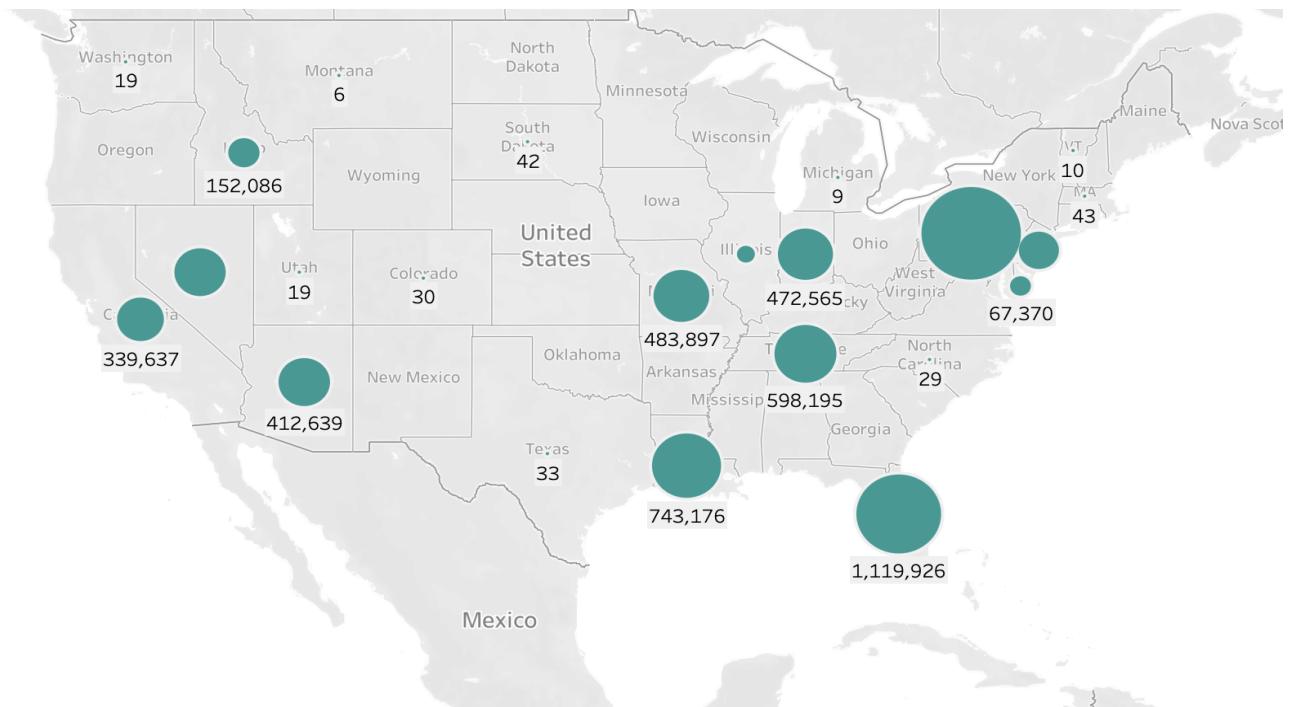
state	n_reviews
PA	1540790
FL	1119926
LA	743176
TN	598195
MO	483997
IN	472565
AZ	412639
NV	409958
CA	339637
NJ	249837
ID	152886
AB	105477
DE	67370
IL	49676
MA	43
SD	42
TX	33
HJ	32
CO	30
NC	29

only showing top 20 rows

## Python script -

```
1  #!/usr/bin/python
2  from pyspark.sql import SparkSession
3
4  spark = SparkSession \
5      .builder \
6      .master('yarn') \
7      .appName('Yelp') \
8      .getOrCreate()
9
10 # Use the Cloud Storage bucket for temporary BigQuery export data used
11 # by the connector.
12 bucket = "final_project3033"
13 spark.conf.set('temporaryGcsBucket', bucket)
14
15 # Load data from BigQuery.
16 businesses = spark.read.format('bigquery') \
17     .option('table', 'firstproject-436502.yelp.businesses') \
18     .load()
19 businesses.createOrReplaceTempView('businesses')
20
21 # Perform word count.
22 businesses_query = spark.sql(
23     'SELECT state AS state, SUM(review_count) as n_reviews FROM businesses GROUP BY state
24     ORDER BY n_reviews DESC'
25 )
26 businesses_query.show()
27 businesses_query.printSchema()
28
29 # Saving the data to BigQuery
30 businesses_query.write.format('bigquery') \
31     .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](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 or have very 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**

Name	Type	Size	Created	Storage class
businesses.csv	text/csv	17.2 MB	Mar 17, 2025, 4:39:04 PM	Standard
businesses.json	application/json	54.7 MB	Mar 17, 2025, 11:42:58 AM	Standard
categories.csv	text/csv	23.9 MB	Mar 17, 2025, 4:39:09 PM	Standard
categories.json	application/json	64.7 MB	Mar 17, 2025, 7:54:47 PM	Standard
new_categories.jsonl	application/octet-stream	46.7 MB	Mar 17, 2025, 9:21:45 PM	Standard
reviews.csv	text/csv	692.2 MB	Mar 17, 2025, 4:47:57 PM	Standard
reviews.json	application/json	1.9 GB	Mar 17, 2025, 2:17:56 PM	Standard
reviews_editted	application/octet-stream	643.1 MB	Mar 17, 2025, 6:13:35 PM	Standard

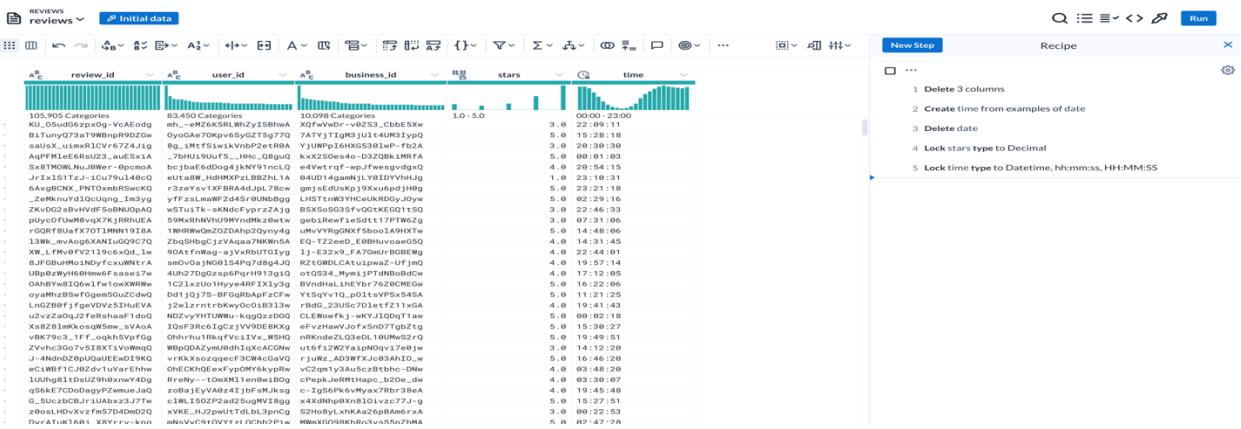
### 2. Modifying table through dataproc –

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.

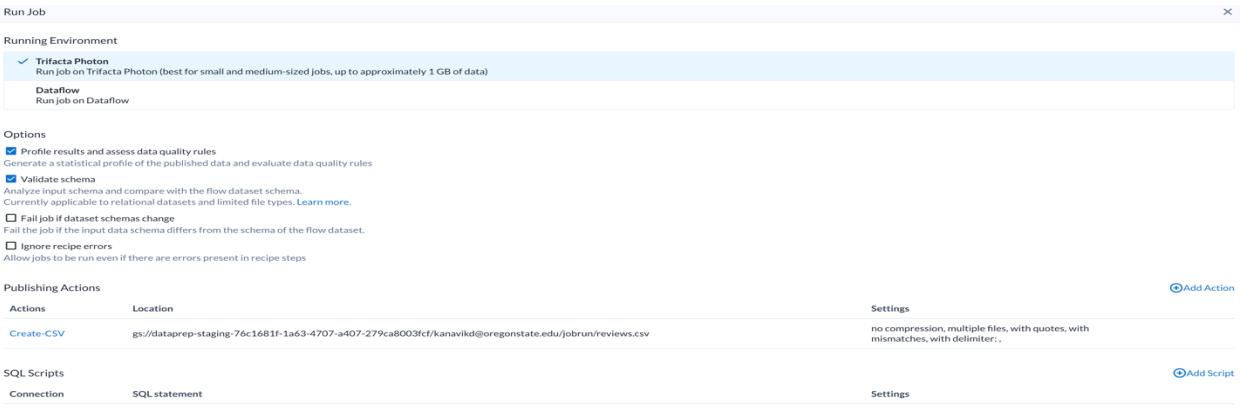
## Initial file – (reviews)



Reviews table modified and its recipe –



## Job execution –



## Results that the job ran successfully -

The screenshot shows the Google Cloud Data Studio interface. At the top, there's a navigation bar with a search icon, a 100% zoom level, and buttons for '+ Add datasets', 'Share', 'Schedule', and '...'. Below the navigation bar is a toolbar with icons for 'Dataset', 'Recipe', and 'Output'. A flow diagram shows a 'Dataset' (reviews.csv) connected to a 'Recipe' (reviews), which then leads to an 'Output' (reviews). To the right of the flow diagram is a 'Details' panel for a job named 'reviews'. The 'Jobs (2)' tab is selected, showing 'Job 31125183 • Completed' by Disha Basavaraja Kanavikar, finished today at 5:09 PM. The output table has three columns: 'column1', 'column2', and 'column3'. The data preview shows several rows of random-looking strings. Below the preview, there are buttons for 'View on Cloud Storage' and 'View details'. At the bottom of the panel, it says 'The preview above shows the current data in the job destination. It might not reflect the output from this particular job run.' There is also a link to 'Previous jobs'.

### 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 –

The screenshot shows the Google BigQuery UI for the 'businesses' table. At the top, there's a header with tabs for 'SCHEMA', 'DETAILS', 'PREVIEW', 'TABLE EXPLORER', 'INSIGHTS', 'LINEAGE', and 'DATA PROFILE'. The 'SCHEMA' tab is selected. Below the header is a table with columns for 'Field name', 'Type', 'Mode', 'Key', 'Collation', 'Default Value', 'Policy Tags', and 'Description'. The table contains 11 rows, each representing a field in the 'businesses' table:

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

## Sample rows –

Screenshot of a data preview interface showing sample rows for the 'businesses' table.

The interface includes a top navigation bar with tabs for 'businesses', 'QUERY', 'OPEN IN', 'SHARE', 'COPY', 'SNAPSHOT', 'DELETE', and 'EXPORT'. Below this is a header row with tabs for 'SCHEMA', 'DETAILS', 'PREVIEW' (which is selected), 'TABLE EXPLORER', 'INSIGHTS', 'LINEAGE', 'DATA PROFILE', and 'DATA QUALITY'.

The main area displays a table with 22 rows of data. The columns are: Row, business\_id, name, address, city, state, postal\_code, latitude, longitude, stars, review\_count, and is\_open.

Row	business_id	name	address	city	state	postal_code	latitude	longitude	stars	review_count	is_open
8	MRmc3ICbmuh5FP7RwCN...	Littman Jewelers	1 Mall Dr	Cherry Hill	NJ	08002	39.9426146	-75.025017	1.0	9	0
9	BzYLuqO-iZ7z5BMJgqfSg	Flynn O'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	4lJnCmean4Gd29yHyJ4oQ	Liberty Travel	951 Haddonfiel...	Cherry Hill	NJ	08002	39.9247290...	-75.035161	1.0	6	1
12	Ekvkftk78ZW5yMaCa0PHg	Kay Jewelers	2020 RT 70 W	Cherry Hill	NJ	08002	39.921491	-75.036943	1.0	6	1
13	WSRvMNRlgjlpIVEGV8E1A	All My Sons Moving & Stor...	32 Coles Ave	Cherry Hill	NJ	08002	39.942177	-75.0085855	1.0	111	1
14	H9R7iuauBruSBDCtQjrXS_g	Koolkipah.com	208 Mimosa Dr	Cherry Hill	NJ	08003	39.8916619	-74.945318	1.0	7	1
15	rIqlwm0gwg6rHsldZZh07A	FedEx Ground	5 Commerce Dr	Barrington	NJ	08007	39.8609281...	-75.041082...	1.0	8	1
16	DCvA43gLeejay_qttR9ABQ	Cham-Pagne Salon & Spa	315 Clements B...	Barrington	NJ	08007	39.8694757	-75.0529608	1.0	5	1
17	HAdHFVYEtcTMWKFPrAak...	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	1xS702gQHb8HvQn1_JL67w	Niagara Pools & Spas	4170 Route 42	Turnersville	NJ	08012	39.7386775...	-75.041295...	1.0	5	1
20	oD0RY7q7HasCz7JpF9QS0g	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_xWm7JIPXfVC...	Checkers	2105 Burlington...	Burlington	NJ	08016	40.049443	-74.838578	1.0	23	0

## Categories table

### Schema -

Screenshot of a data schema interface showing the 'categories' table.

The interface includes a top navigation bar with tabs for 'categories', 'QUERY', 'OPEN IN', 'SHARE', and 'COPY'. Below this is a header row with tabs for 'SCHEMA', 'DETAILS', 'PREVIEW' (which is selected), and 'TABLE EXPLORER'.

The main area displays a table with 3 rows of data. The columns are: Field name, Type, Mode, Key, Collation, and D.

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

## Sample rows –

categories

QUERY OPEN IN SHARE COPY

SCHEMA DETAILS PREVIEW TABLE EXPLORER PREVIEW INSIGHTS

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	WJ0W3CVzHFIJQU8jyMsyRg
155	ATV Rentals/Tours	1fLyoP0UD8-rxuJ7qcROgg
156	ATV Rentals/Tours	4LloC1-IJSPD7T-KM9r7AQ
157	ATV Rentals/Tours	_RaaqliKaLx4xSohCR8gpQ
158	ATV Rentals/Tours	sXcjSFTA8EUf_cngzS-AA
159	Acai Bowls	1E9o1SN07UTf1XHTFPv1_Q
160	Acai Bowls	71U7MxQEhwitJ0m4CQpRwQ
161	Acai Bowls	WJlc6ZIR-pd9W-Vt-rRtzA
162	Acai Bowls	wrnjFEC-w0qWnyDnO8k1RpA
163	Acai Bowls	DZdE09qqCcX7Atf3Df7Yqg
164	Acai Bowls	F3dxQiOwN55rSLw-BNSq6A
165	Acai Bowls	C1QkJ4s_p3hMTrvOgM7ssA
166	Acai Bowls	yJqN-U9dUyIYEwTr8SBP1Q
167	Acai Bowls	1ZRGRUd3y5a8vrgUyzMcnQ
168	Acai Bowls	1ZGRGRUd3y5a8vrgUyzMcnQ

Reviews table –

Schema –

reviews

QUERY OPEN IN SHARE COPY

SCHEMA DETAILS PREVIEW TABLE EXPLORER PREVIEW

Filter Enter property name or value

<input type="checkbox"/>	Field name	Type	Mode	Key	Collation
<input type="checkbox"/>	review_id	STRING	NULLABLE	-	-
<input type="checkbox"/>	user_id	STRING	NULLABLE	-	-
<input type="checkbox"/>	business_id	STRING	NULLABLE	-	-
<input type="checkbox"/>	stars	FLOAT	NULLABLE	-	-
<input type="checkbox"/>	useful	INTEGER	NULLABLE	-	-
<input type="checkbox"/>	funny	INTEGER	NULLABLE	-	-
<input type="checkbox"/>	cool	INTEGER	NULLABLE	-	-
<input type="checkbox"/>	date	TIMESTAMP	NULLABLE	-	-

## Sample rows -

The screenshot shows the BigQuery interface with the 'reviews' dataset selected. The 'PREVIEW' tab is active, displaying 19 rows of data from the 'reviews' table. The columns shown are: review\_id, user\_id, business\_id, stars, useful, funny, cool, and date. Each row contains a unique identifier and various numerical values representing user interactions and timestamp.

Row	review_id	user_id	business_id	stars	useful	funny	cool	date
1	9qfAY508E3bqqN3Cm5Jz5A	oKa9RweNg0i9HWgPDtNE7g	HhcLmj040ltuaqS2m0e8FA	1.0	0	0	0	2016-03-17 19:18:42 UTC
2	6SM2BNPOE6_iHmQLcq4QJA	Nuq0IP2WiSzZ3awefC_Qw	2PwMmUV2Leb2loj0TqxCMa	1.0	0	0	0	2011-07-01 13:04:36 UTC
3	5_lOShJWoy7UCJqymleUZQ	tElfxu_MN2aYx7iGr2Lg	HeLuCW7loBXVybQIn4pQAQ	1.0	0	1	0	2013-03-21 18:05:32 UTC
4	CXZcLL8sfJgwSMbySA4Xg	ECUMOS04dIhjlxTGvJ_w	xeRbOKTcZZvk6BLYQ9lpw	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	XJNGISiVVVik_r_Rab10xw	GST3wg-wej15HeCraXE6w	1.0	0	0	0	2018-05-23 13:15:14 UTC
7	bl4ptSReJzu1UiYc30oDew	_641wqEkqqTEMjll2l2zWQ	26107Pz4hjEsqQ8J2c81A	1.0	0	0	0	2017-08-03 01:17:54 UTC
8	bTnIA7MSWBNGh_ZIFW2cQ	OeMDHEXQcsq0ZGhk9Ta7qg	KOfgk4091DV320hbonBEw	1.0	0	0	0	2016-06-05 16:18:50 UTC
9	aeypZDF_qM5-5U0U1bVKfA	et134opthbP9j6qk_6sp-g	JGVxKGvQEaMZ60_2l0UucA	1.0	0	0	1	2017-04-18 01:11:15 UTC
10	C3iKunDM-ys6BZN2k3uqq	Q3JQTAW2jwZR87KmplR6A	2mdQ6nhqrqeMnkFksJuY6NA	1.0	0	0	0	2017-05-18 23:10:02 UTC
11	ODvNSEKTxNGHrADTHk0vpw	56IMtScQBfk02rKojZr6nA	3TDKS_JrEvPnZ809GcuHfg	1.0	0	0	0	2017-07-15 23:01:10 UTC
12	IY6xAevNU8jNgbCV3b15g	bzCrg9WSV1IDV_A_6KGXsQ	JcdsRaKIWNy9CT5Tl8LVA	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	aOVStb6B6DlqVaBJ04dmPxQ	1.0	0	0	0	2018-11-10 00:18:45 UTC
15	uNwtsxmwdwKborJUV5xY62Q	-M56010to00AOGOkwGM3rnQ	PaVv8oDCYfLCiRs6ox7P0A	1.0	0	0	0	2018-12-20 05:40:10 UTC
16	nLnrroeozV-ZcgjJuWSA	LFBISBWJw4jIM2vbSXP9MA	FMaUJx3e8ydLLaYY1u4qKQ	1.0	0	0	0	2018-06-28 17:52:40 UTC
17	rrZIHKKgst3oHLC18rf9Q	wROicHYB-7oFx6Pg2JKA	DSKpda2eBvPbTcwT3m4cUQ	1.0	0	0	0	2017-04-14 02:49:44 UTC
18	-Nisi3YmQdeLoBjnjfp8dw	M8N7UawSmRywzHVGs04nzQ	RYg2efo6qexJS_yDemr_1Q	1.0	0	0	0	2018-12-17 23:01:53 UTC
19	Dsjrze_6zQls7zwK8xQaWg	tBVldpvNZ9Vfa6YUqf6A	GGVjh3ZYQmon1mM1-6ZXQ	1.0	0	2	0	2018-02-17 03:57:28 UTC

## 4. Creating a spark job –

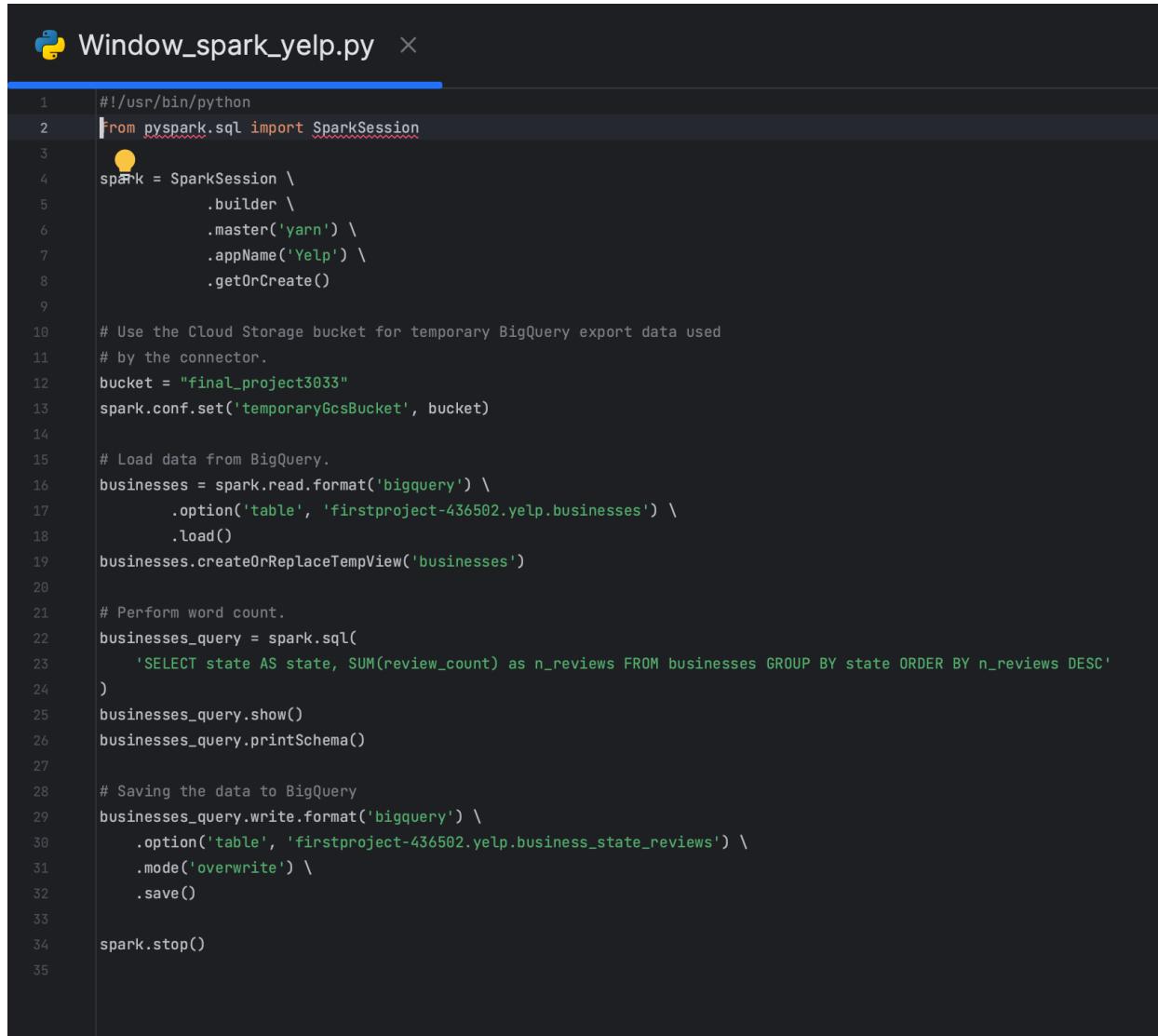
I used the Window\_spark\_planes\_0.py from our previous assignments and modified it to match my requirements. I uploaded it to the Google bucket and created a spark job in Dataproc to execute it. Here are the snippet. The spark job also creates a table in bigquery named – “business\_state\_reviews”

I used the Spark VM instances we created earlier –

The screenshot shows the GCP VM instances page. It lists four VM instances under the 'VM instances' section:

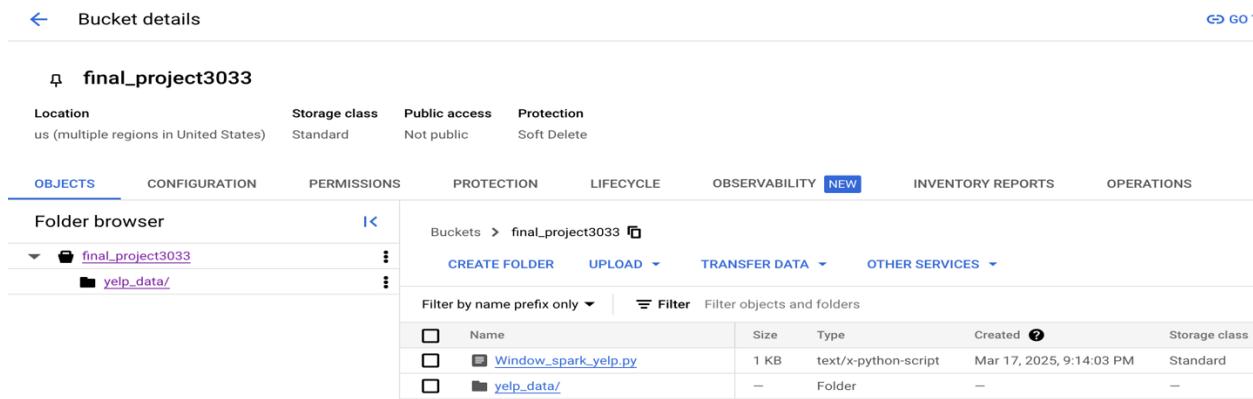
- cs512-spark1-m**: Status: green, Zone: us-east1-b, Internal IP: 10.142.0.5 (nic0), External IP: 35.211.166.40 (nic0), Connect via SSH.
- cs512-spark1-w-0**: Status: green, Zone: us-east1-b, Internal IP: 10.142.0.7 (nic0), External IP: 35.211.166.41 (nic0), Connect via SSH.
- cs512-spark1-w-1**: Status: green, Zone: us-east1-b, Internal IP: 10.142.0.6 (nic0), External IP: 35.211.166.42 (nic0), Connect via SSH.
- finalroject**: Status: yellow, Zone: us-west1-a, Internal IP: 10.138.0.4 (nic0), External IP: 35.211.166.43 (nic0), Connect via SSH.

## 1. Python script –



```
1  #!/usr/bin/python
2  from pyspark.sql import SparkSession
3
4  spark = SparkSession \
5      .builder \
6      .master('yarn') \
7      .appName('Yelp') \
8      .getOrCreate()
9
10 # Use the Cloud Storage bucket for temporary BigQuery export data used
11 # by the connector.
12 bucket = "final_project3033"
13 spark.conf.set('temporaryGcsBucket', bucket)
14
15 # Load data from BigQuery.
16 businesses = spark.read.format('bigquery') \
17     .option('table', 'firstproject-436502.yelp.businesses') \
18     .load()
19 businesses.createOrReplaceTempView('businesses')
20
21 # Perform word count.
22 businesses_query = spark.sql(
23     'SELECT state AS state, SUM(review_count) as n_reviews FROM businesses GROUP BY state ORDER BY n_reviews DESC'
24 )
25 businesses_query.show()
26 businesses_query.printSchema()
27
28 # Saving the data to BigQuery
29 businesses_query.write.format('bigquery') \
30     .option('table', 'firstproject-436502.yelp.business_state_reviews') \
31     .mode('overwrite') \
32     .save()
33
34 spark.stop()
35
```

## 2. Python file in bucket –



OBJECTS	CONFIGURATION	PERMISSIONS	PROTECTION	LIFECYCLE	OBSERVABILITY	INVENTORY REPORTS	OPERATIONS
Folder browser				Buckets > final_project3033			
final_project3033							
yelp_data/							

Filter by name prefix only ▾ | Filter Filter objects and folders

Name	Size	Type	Created	Storage class
Window_spark_yelp.py	1 KB	text/x-python-script	Mar 17, 2025, 9:14:03 PM	Standard
yelp_data/	—	Folder	—	—

### 3. Job created and its output –

<https://console.cloud.google.com/dataproc/jobs/job-86e21edf/configuration?region=us-east1&project=firstproject-436502>

Job ID: job-86e21edf  
Job UUID: 3fc1f512-8995-4bda-a42d-e67e5f242664  
Type: Dataproc Job  
Status: Succeeded

**MONITORING** **CONFIGURATION**

Start time: Mar 17, 2025, 9:14:21 PM  
Elapsed time: 48 sec  
Status: Succeeded  
Region: us-east1  
Cluster: cs512-spark1  
Job type: PySpark  
Main python file: gs://final\_project3033/Window\_spark\_yelp.py  
Jar files: gs://hadoop-lib/bigquery/bigquery-connector-hadoop2-latest.jar  
Max restarts per hour: 1  
Labels  
Performance  
Enhancements  
Advanced optimizations: Off

**Output** LINE WRAP: OFF

```
!-- state: string (nullable = true)
!-- n_reviews: long (nullable = true)
```

Output is complete

<https://console.cloud.google.com/dataproc/jobs/job-86e21edf/monitoring?job=job-86e21edf&region=us-east1&project=firstproject-436502>

Job ID: job-86e21edf  
Job UUID: 3fc1f512-8995-4bda-a42d-e67e5f242664  
Type: Dataproc Job

**Output** LINE WRAP: OFF

Spark jobs take ~60 seconds to initialize resources.

```
25/03/18 04:14:43 INFO ReadSessionCreator: Requested 20000 max partitions, but only received 1 from the BigQu
25/03/18 04:14:43 INFO BigQueryDataSourceReaderContext: Got read session for GenericData{classInfo=[datasetId
+-----+
|state|n_reviews|
+-----+
| PA| 1540790|
| FL| 1119926|
| LA| 743176|
| TN| 598195|
| MO| 483897|
| IN| 472565|
| AZ| 412639|
| NV| 409950|
| CA| 339637|
| NJ| 249837|
| ID| 152086|
| AB| 105477|
| DE| 67370|
| IL| 49676|
| MA| 43|
| SD| 42|
| TX| 33|
| HI| 32|
| CO| 30|
| NC| 29|
+-----+
only showing top 20 rows
```

root

```
!-- state: string (nullable = true)
```

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

Job ID	Status	Region	Type	Cluster	Start time	Elapsed time	Labels
job-86e21edf	Succeeded	us-east1	PySpark	cs512-spark1	Mar 17, 2025, 9:14:21 PM	48 sec	None
job-6d065715	Failed	us-east1	PySpark	cs512-spark1	Mar 17, 2025, 9:11:59 PM	21 sec	None
job-24c9dca6	Failed	us-east1	PySpark	cs512-spark1	Mar 17, 2025, 9:08:31 PM	24 sec	None
job-c87b60d6	Failed	us-east1	PySpark	cs512-spark1	Mar 17, 2025, 9:00:33 PM	22 sec	None
job-cabaac18	Failed	us-east1	PySpark	cs512-spark1	Mar 17, 2025, 8:57:26 PM	18 sec	None
job-1627e1d1	Failed	us-east1	PySpark	cs512-spark1	Mar 17, 2025, 8:40:58 PM	26 sec	None

#### 4. Its output in BigQuery –

Field name	Type	Mode	Key	Collation	Default Value	Policy Tags
state	STRING	NULLABLE	-	-	-	-
n_reviews	INTEGER	NULLABLE	-	-	-	-

#### 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.