Springboard Data Engineering Final Presentation

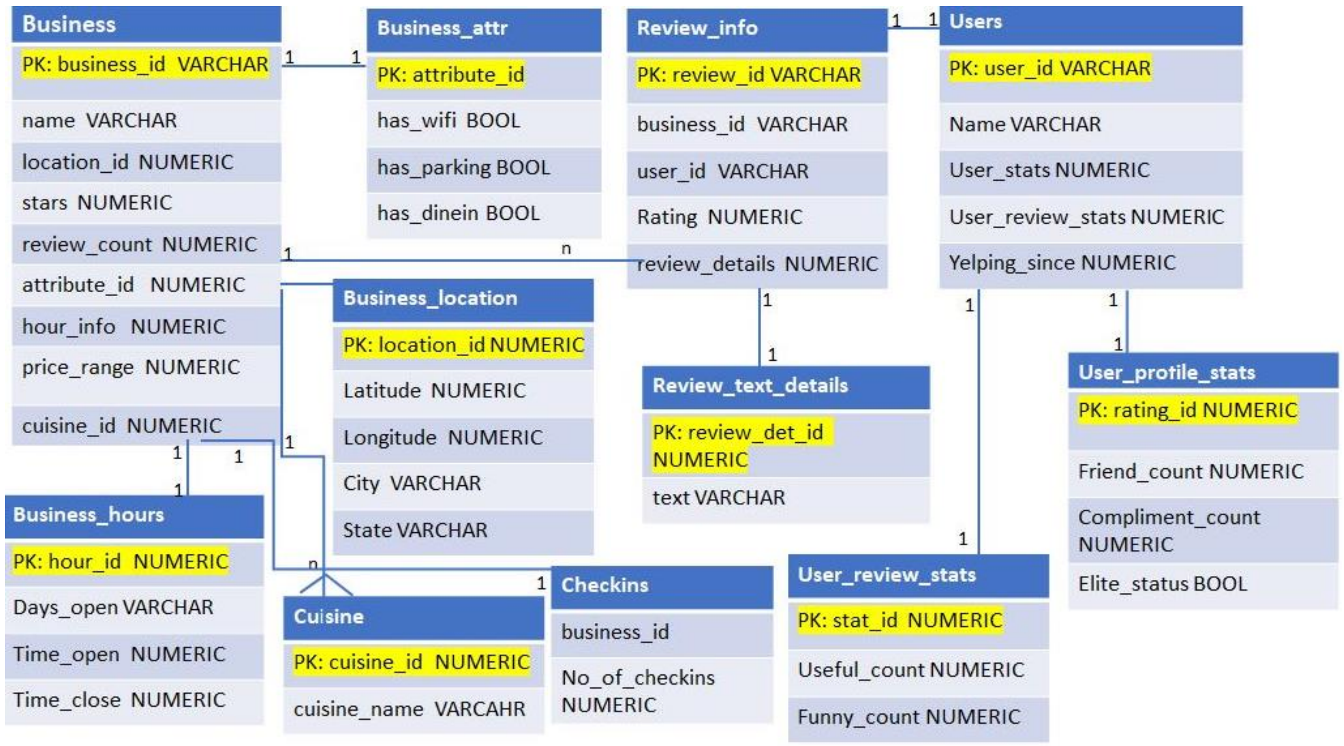
Rachael Cardoso

Introduction

- The goal of this project was to explore key concepts to handle large dataset, create a pipeline to process and finally use tools to query and visualize the findings
- For my study I chose to focus on a Yelp Dataset – available at: https://www.kaggle.com/yelp-dataset/yelp-dataset
- Yelp Dataset is comprised of 4 smaller datasets focused on business, user, tips, reviews and checkins- each providing different layers of information about each business
- The research question I aimed to explore is: **What impact does scarcity have on the rating and overall review of a restaurant?**

Data Exploration

- Before creating the data pipeline it's important to explore each dataset to understand schema and structure. This was done using pandas to read a small chunk of each dataset to get an overview of how to process an clean.



Data Pipeline Prototype

- Due to the size of the datasets (50 MB) the full dataset couldn’t be loaded in pandas as a dataframe. Instead the dataset was loaded in chunks and then transformations were applied.

```
if filename.endswith("business.json"):
    for df in pd.read_json(self.unzip_folder_path + r'\/' + filename, lines=True, chunksize=1000):
```

- Additionally some of the main functions to clean data were filtering non restaurants, delete Nan values, and handling unique characters that would obstruct querying
- This initial data pipeline took about 2-3 hours to fully process all datasets and store them as csv files locally

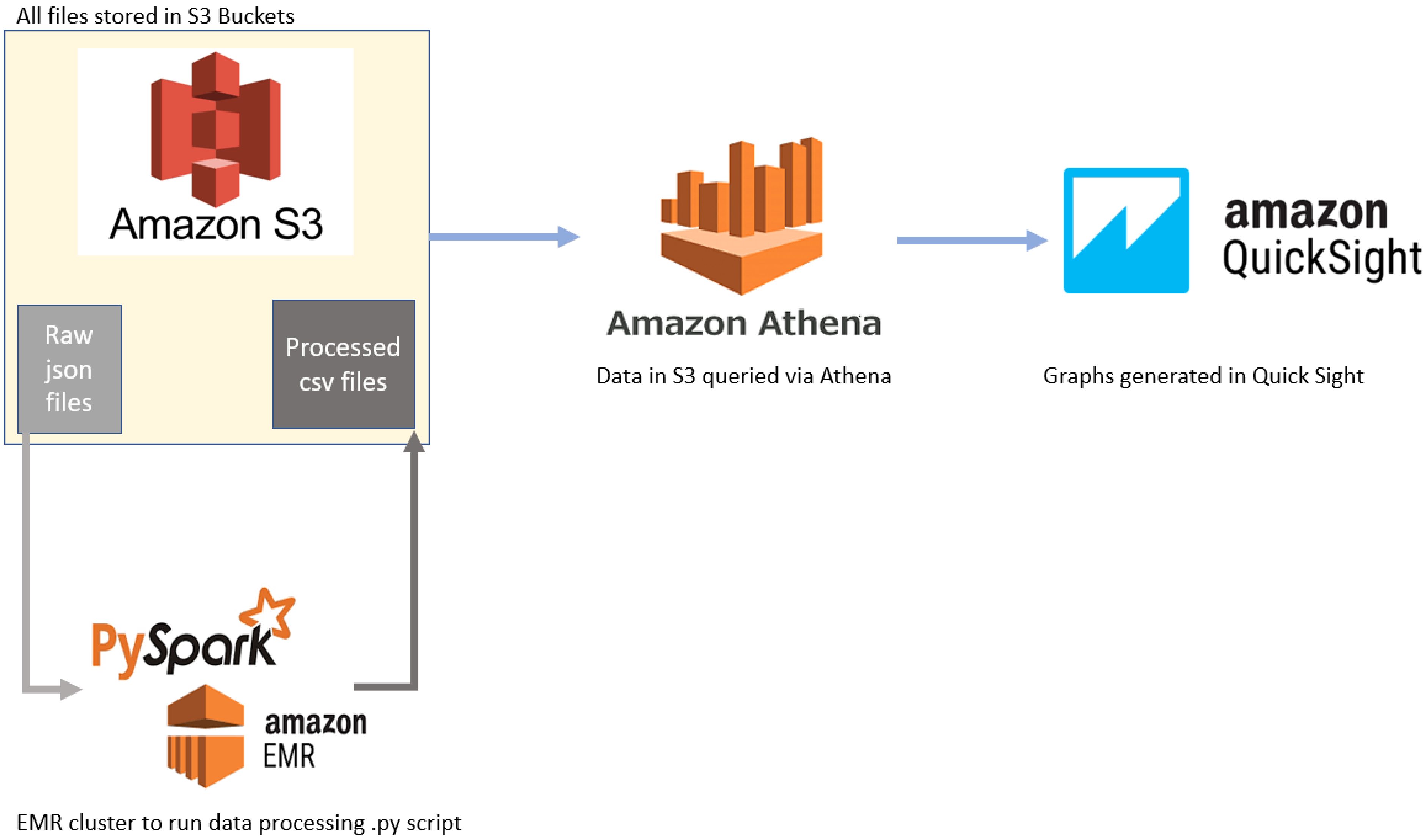
Migrating to Spark

- As mentioned in the previous slide, loading and processing datasets locally took 2-3 hours + consumed a lot of storage and memory. Thus, the pipeline was migrated to Spark and executed on an EMR cluster (more details in upcoming slides)
- Besides execution time, one of the biggest gains in Spark was code length and simplicity. Since Spark is optimized for handling big data and parallelizing datasets the multiple for loops I needed to process data could be replaced with a single line of code:

```
df_b = spark.read.option("header",True).csv(r"dataset.csv")
```

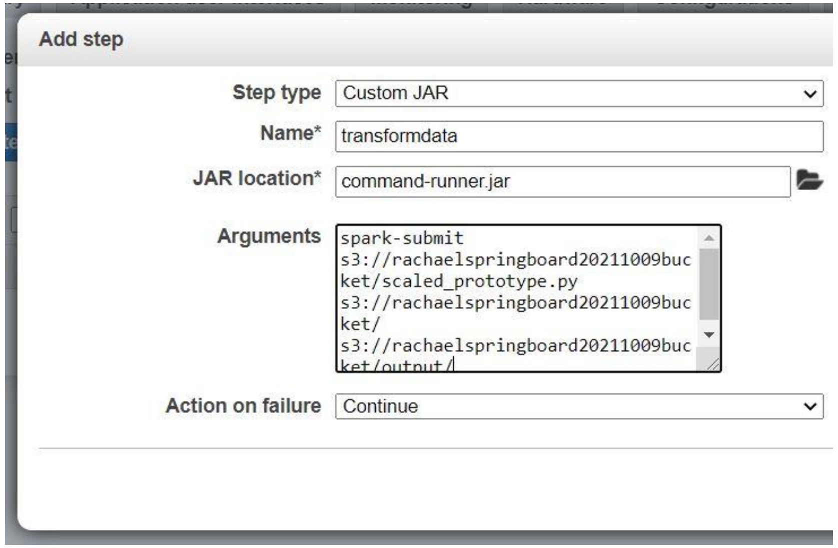
```
for filename in os.listdir(self.unzip_folder_path):
    if filename.endswith('business.json'):
        count = 0
        for dataframe in pd.read_json(self.unzip_folder_path + r'\/' + filename, lines=True, chunksize=1000):
            b = Business(dataframe=dataframe)
            df = b.remove_non_restaurants()
            df = b.aggregate_hours_open(df=df)
            df = b.clean_attributes(df=df)
            df.to_csv(self.output_path + r"\business_" + str(count) + '.csv', index=False)
            count += 1
        pass
```

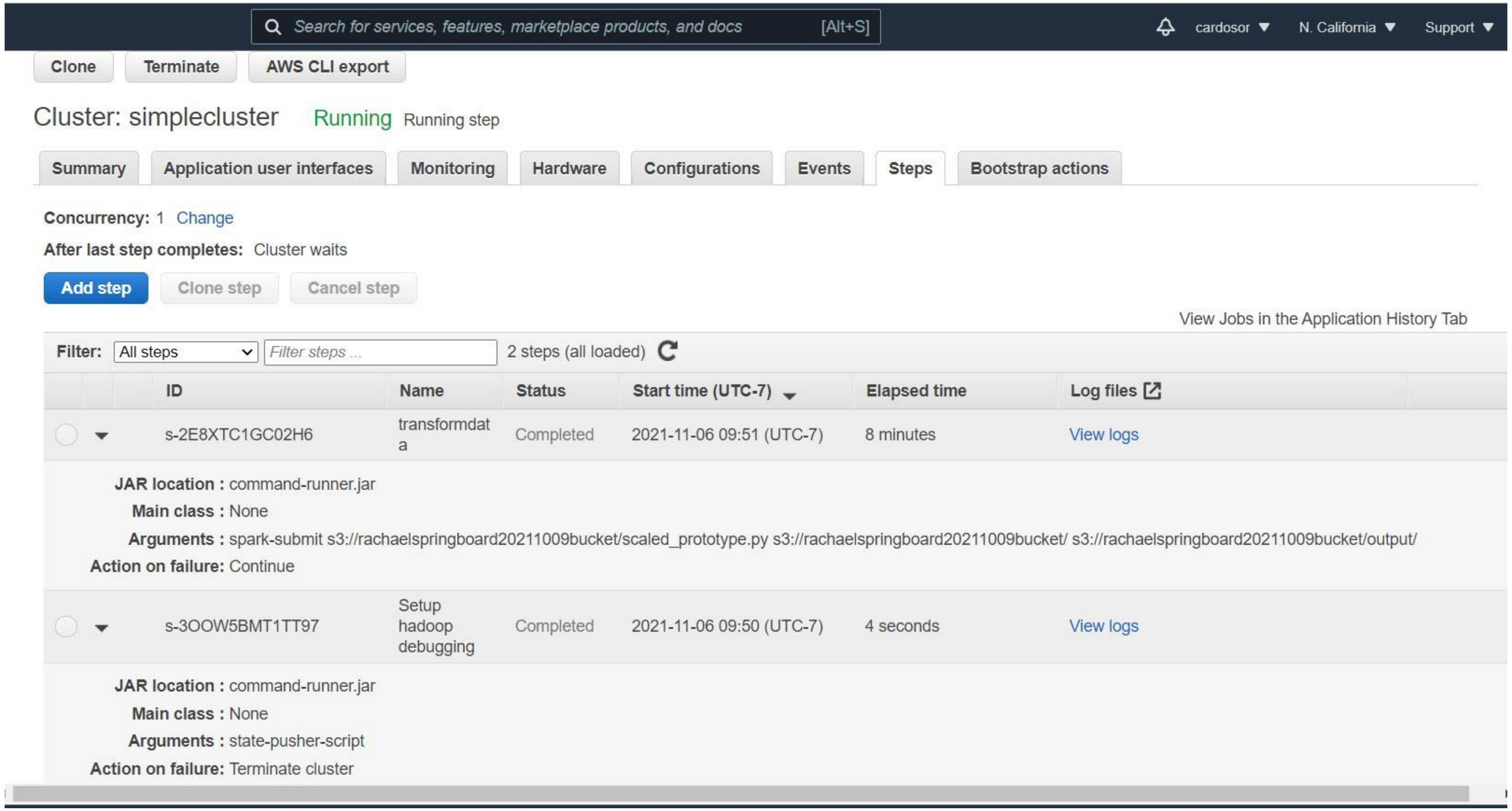
AWS Infrastructure



AWS Infrastructure | EMR

- Below are the screenshots used to setup and execute the cluster





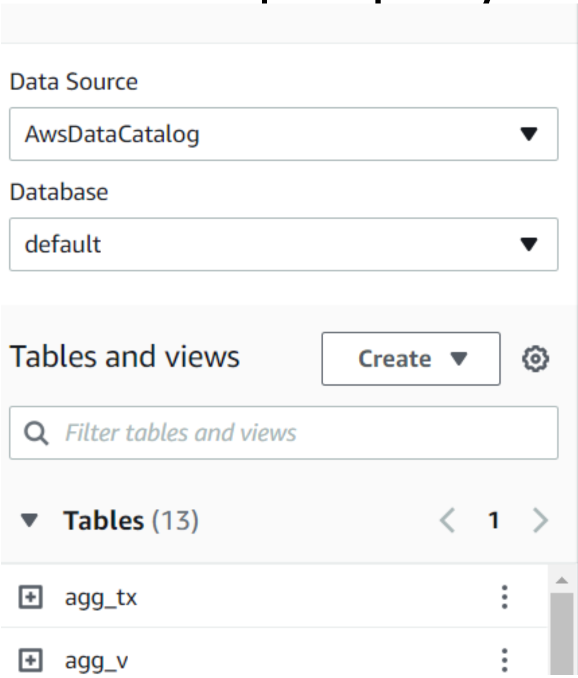
Deploying Final Code

- After confirming the cluster executed successfully after manually setting it up in the console, the cluster creation and deployment was moved to a shell script so as to automate the process
- The benefit of this approach is better repeatability for future cluster creation and auto-terminating cluster when computations complete

```
aws emr create-cluster --release-label emr-5.33.1  --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m5.xlarge InstanceGroupType=CORE,InstanceCour

aws emr describe-cluster --cluster-id j-107L34TIBRK2

aws emr add-steps --cluster-id j-9ZDMCFI38DE2 --steps Type=CUSTOM_JAR,Name="Spark Program",Jar="command-runner.jar",ActionOnFailure=CONTINUE,Args=["spark-submit",s3://r

aws emr put-auto-termination-policy --cluster-id j-9ZDMCFI38DE2 --auto-termination-policy IdleTimeout=60
```

Querying data with Athena

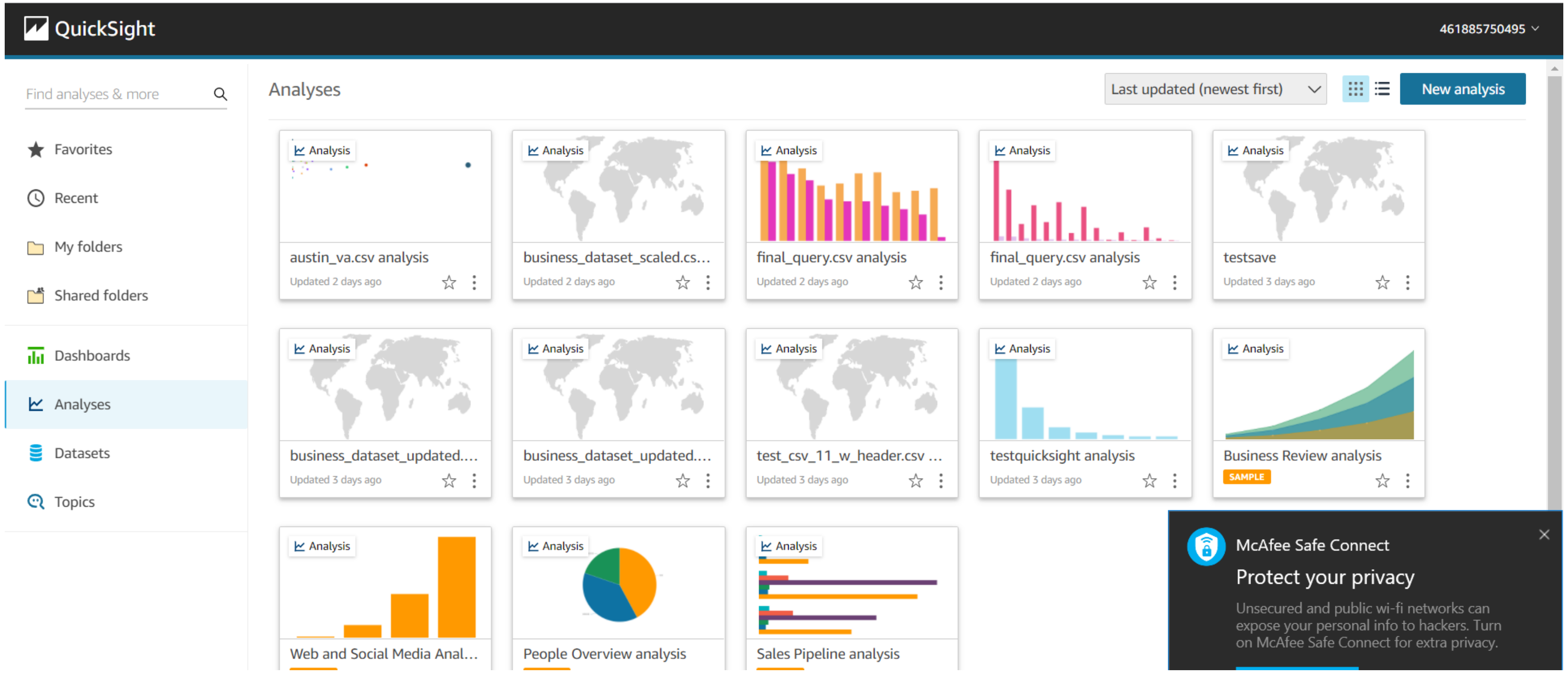
- With all the processed csv files in the s3 bucket, Athena was used to query the data and make sub-tables and views to showcase correlations between cuisine and rating in states, cities and a geospatial area given lat/long coordinates.
- The following queries demonstrate how the data was first imported into Athena and then a sample query used for later visualization.



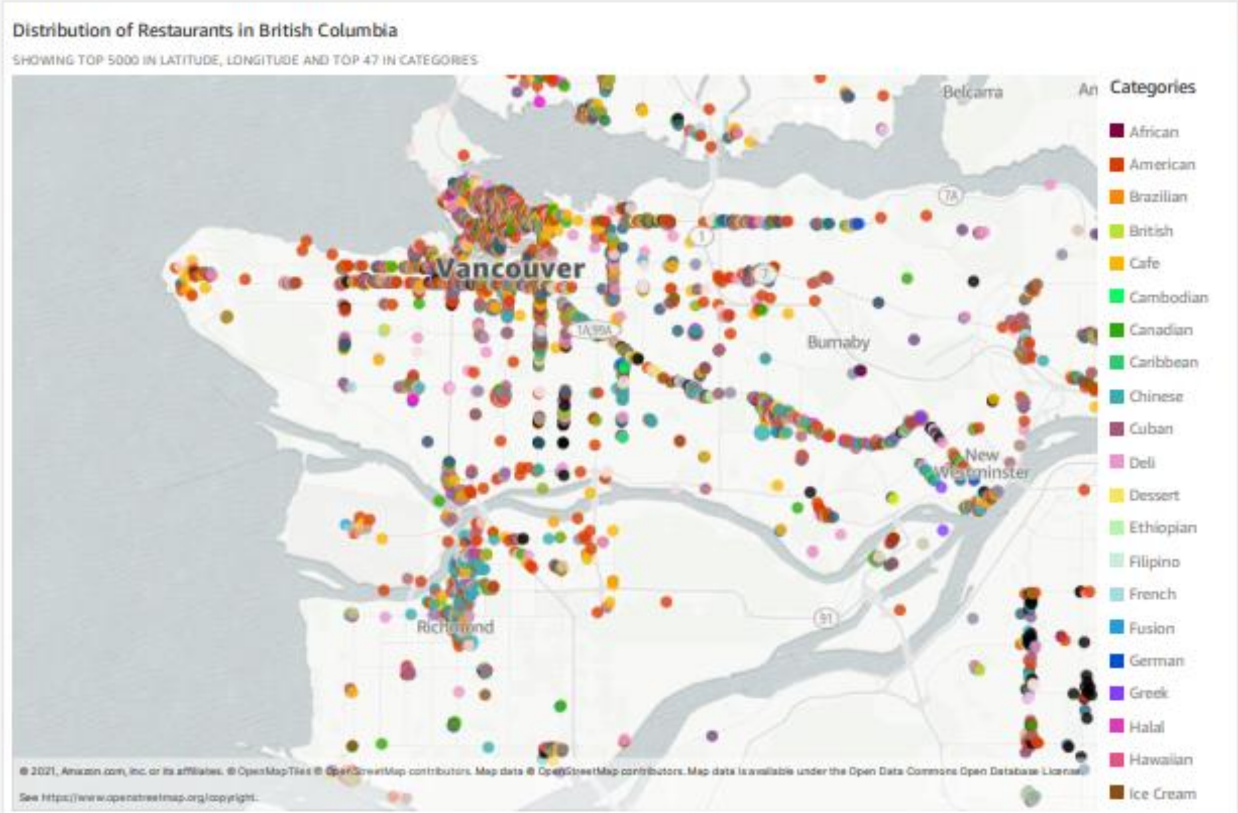

```
CREATE EXTERNAL TABLE IF NOT EXISTS business (
              address VARCHAR (255),
              business_id VARCHAR(255),
              categories VARCHAR (255),
              city VARCHAR (255),
              is_open INT ,
              latitude DECIMAL(10, 7),
              longitude DECIMAL(11, 7) ,
              name VARCHAR (255),
              zipcode INT,
              review_count INT,
              stars DECIMAL(2,1) ,
              state VARCHAR (255),
              price INT
               )
               ROW FORMAT DELIMITED
               FIELDS TERMINATED BY ","
               LINES TERMINATED BY "\n"
               LOCATION 's3://rachaelspringboard20211009bucket/testathena/';
```

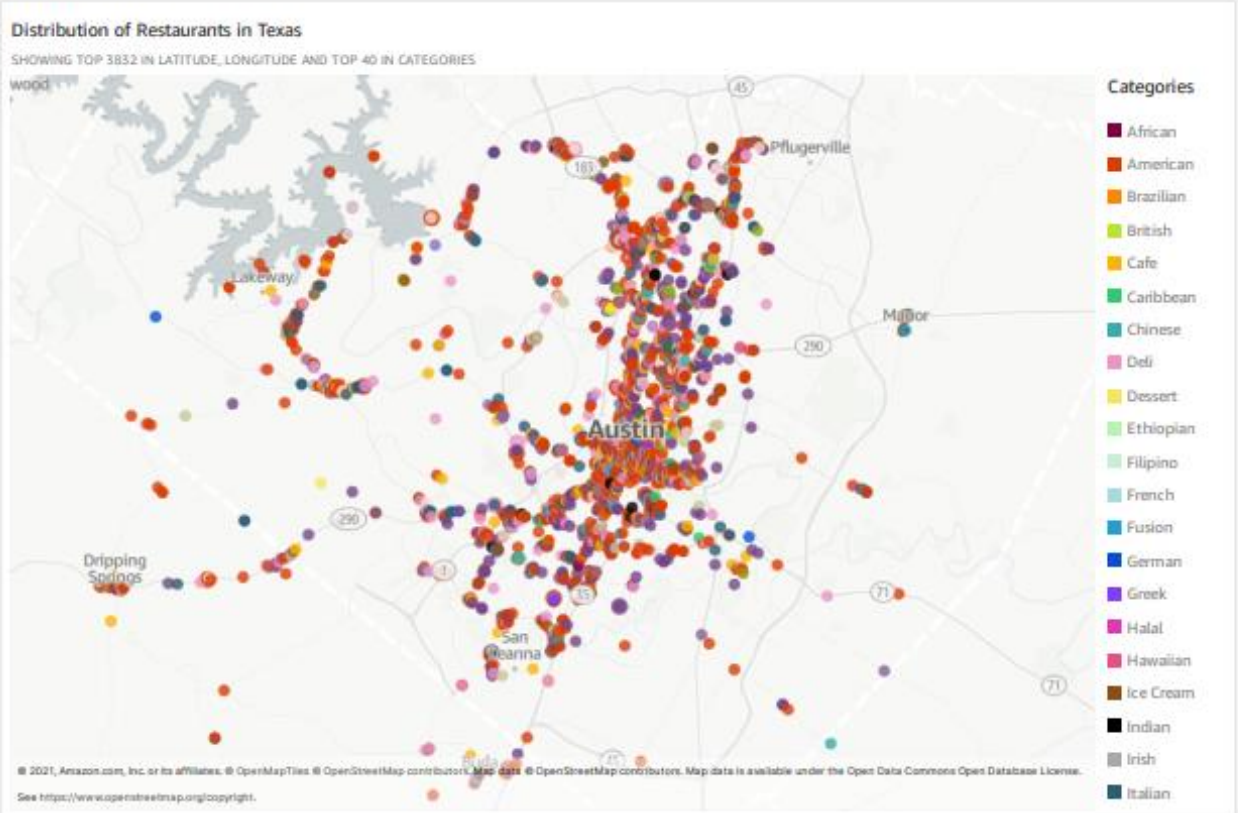
Data Visualization

- After the queries were executed in Athena, Quicksight was used for the data visualizations
- Since it's a part of the AWS ecosystem and was easy to integrate directly with the query outputs in s3



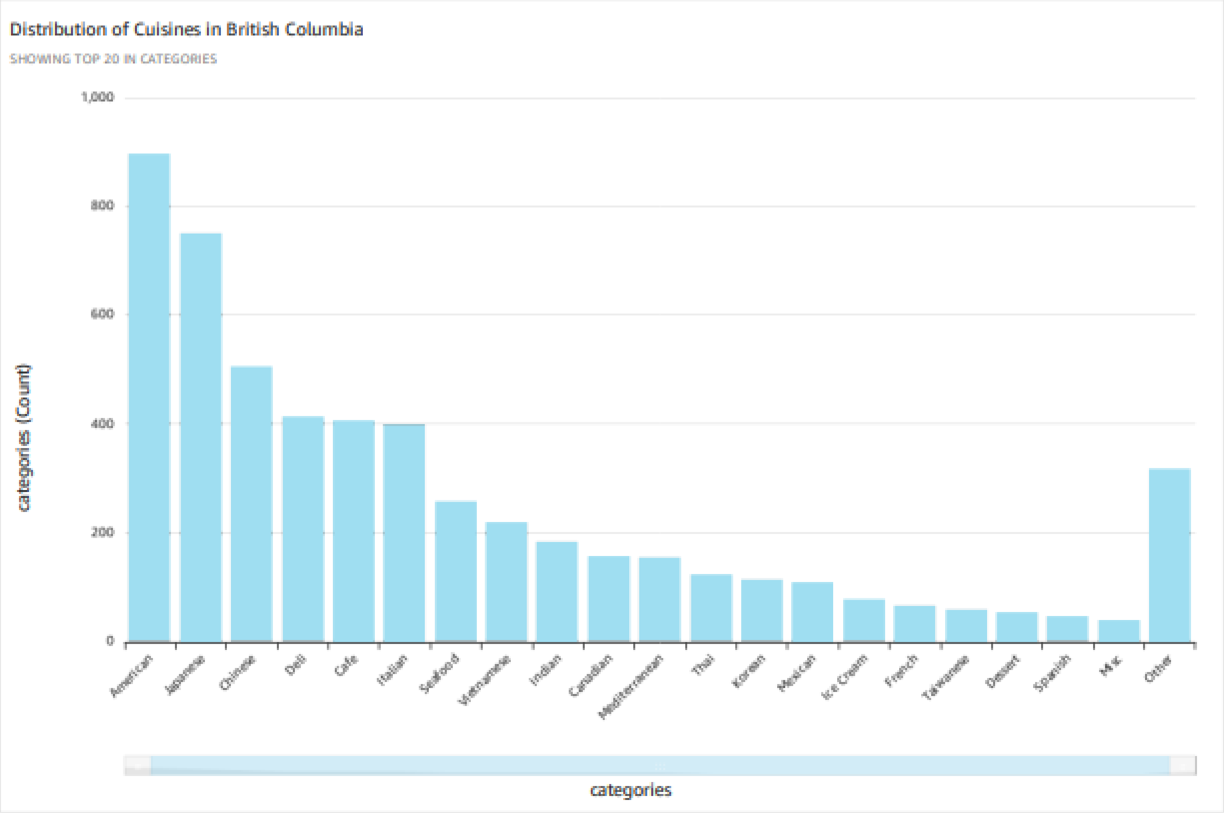
Final Plots

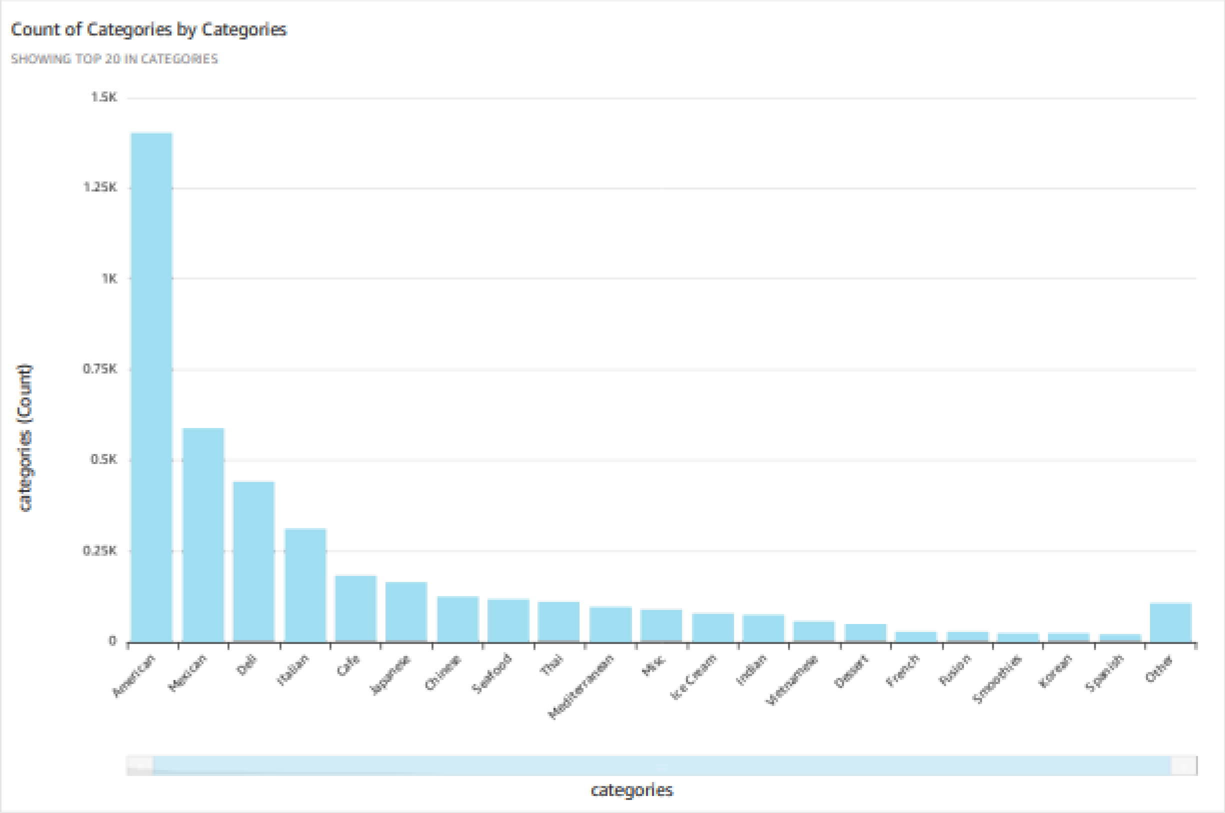




To study the impact of cuisine scarcity, 2 states with the largest distribution of restaurants were chosen for this case study. Plots above depict the distribution of restaurants in British Columbia and Texas, colored according to cuisine. The purpose of these plots is to portray a geographical overview for context

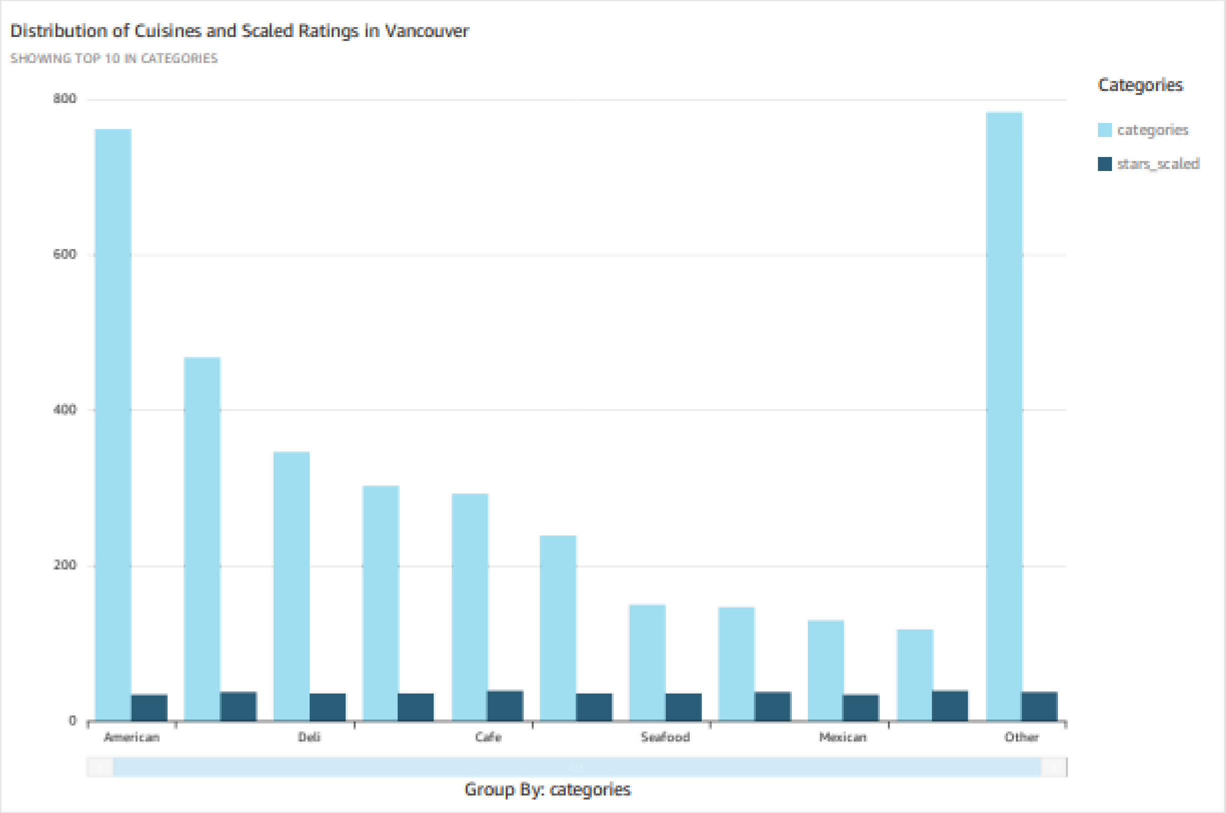
```
SELECT distinct categories, count(categories) as total from business
WHERE state LIKE 'BC'
GROUP BY categories
ORDER BY total DESC
```

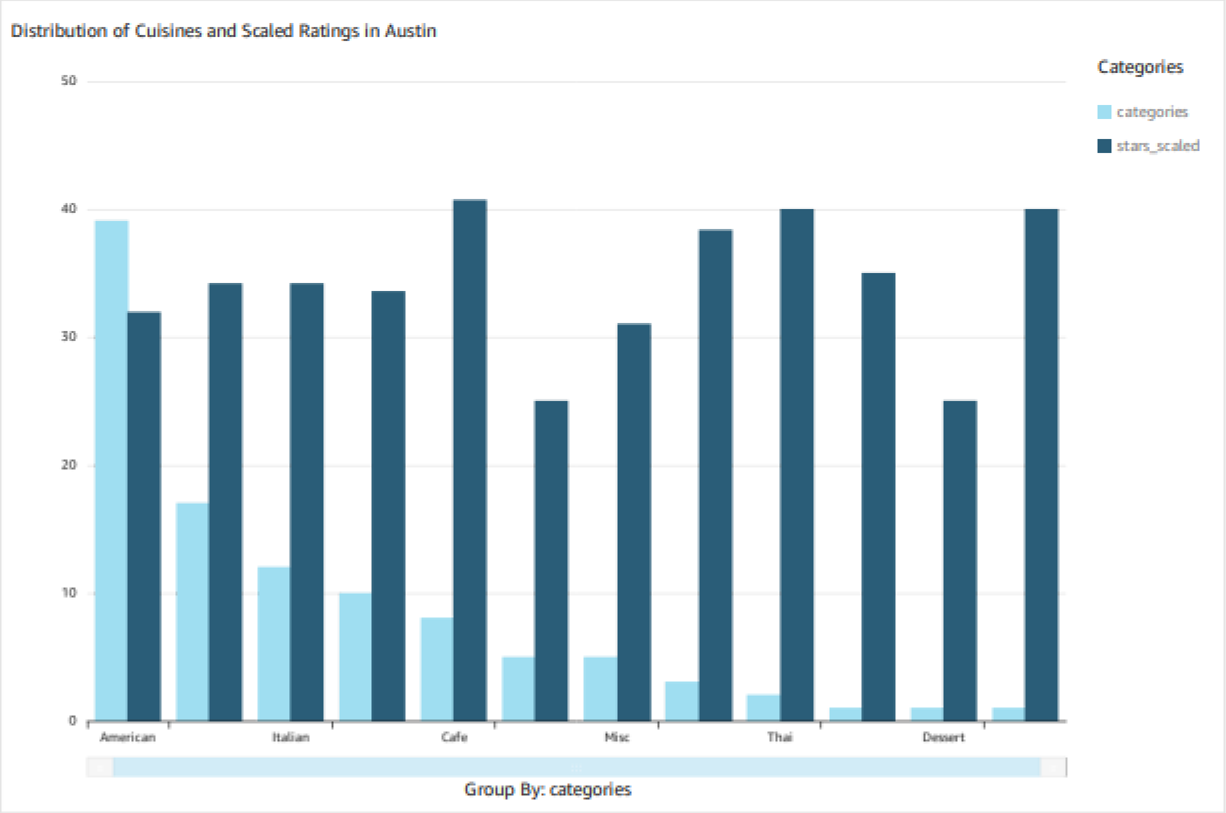




These charts depict the distribution of each cuisine in British Columbia (L) and Texas (R). These two regions states have diverse demographics and ethnic populations which is reflected in the population of restaurants. Top cuisines in BC are Japanese, Chinese, Delis and Cafes whereas in Texas the top cuisines are Mexican, Delis, Italian and Cafes.

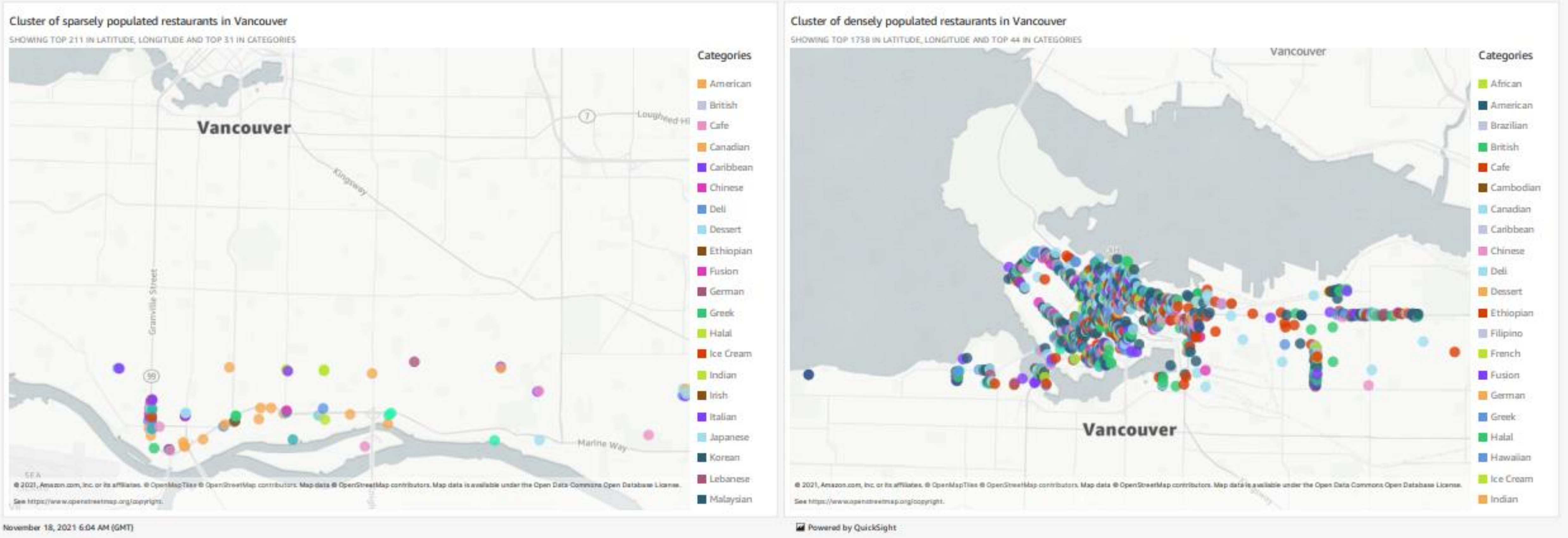
```
SELECT categories as cuisine, count(categories) as sum_cuisine, avg(stars) as avg_rating FROM business
WHERE city LIKE 'Vancouver'
GROUP BY categories
ORDER BY categories
```





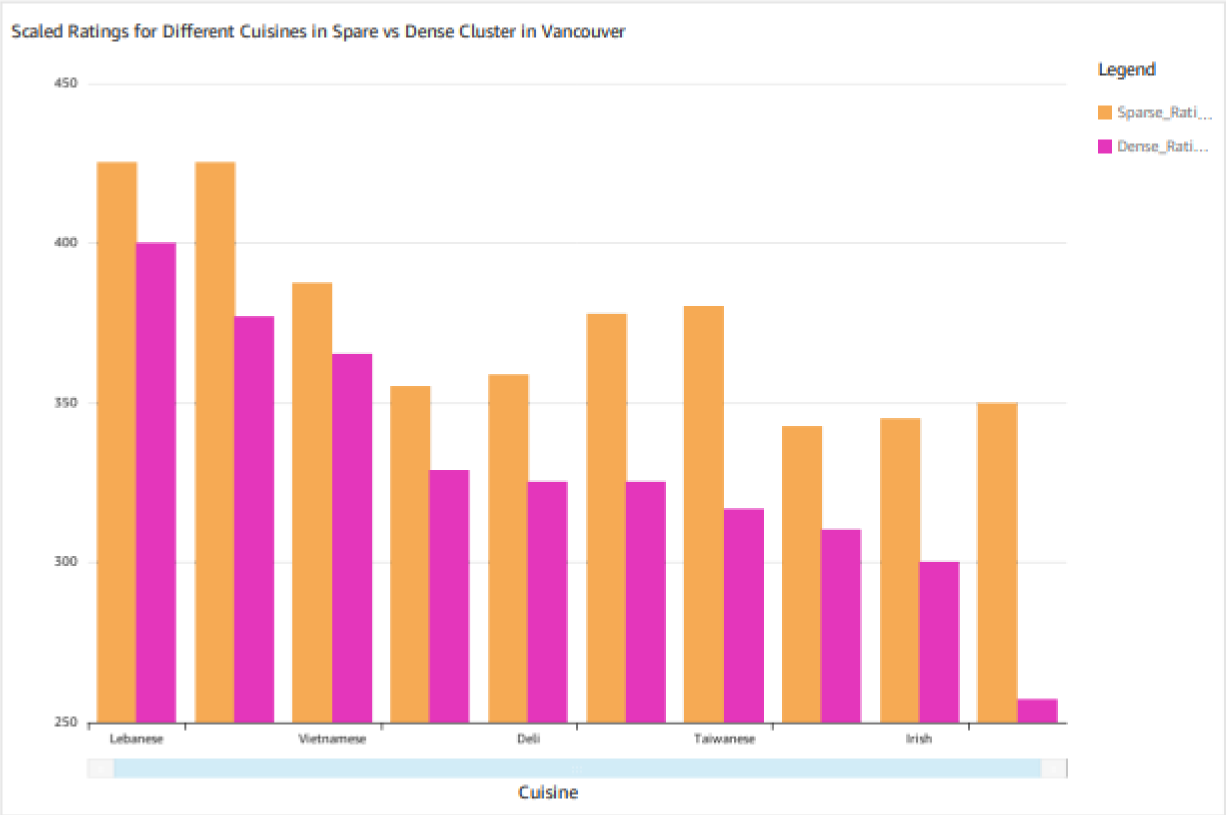
These plots depict the population of each cuisine with their average star rating for restaurants in Vancouver and Austin. For better representation on the combined y axis, the ratings were scaled 10x (3.5 -> 35 etc). Some interesting observations are the rating of Mexican restaurants in Austin [590 total] are on average 1 star lower than in Vancouver [72 total] whereas Japanese restaurants in Vancouver [432 total] are on average 0.4 stars lower than in Austin [162 total]. This is a good indication that scarcity/lower sample size ca inflate the rating of a particular cuisine.

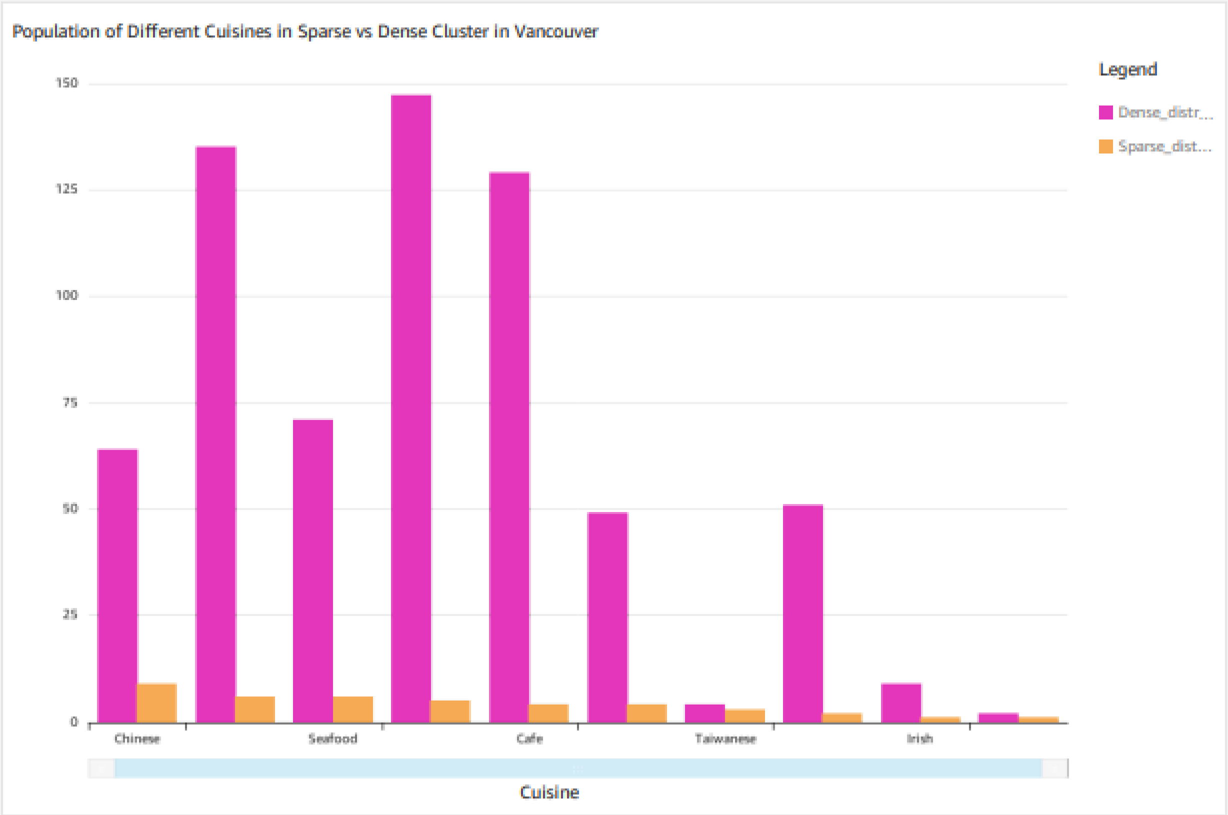
```sql
SELECT categories as cuisine, count(categories) as sum_cuisine, avg(stars) as avg_rating FROM query
WHERE latitude BETWEEN 40.001 AND 40.04
AND longitude BETWEEN -105.29 AND -105.20
GROUP BY categories
ORDER BY categories
```



Finally, zooming in on Vancouver, 2 different ‘clusters’ were compared to see if the rating of a cuisine was impacted if it had a smaller populated in a 2 km latitude spread.

```
SELECT s.cuisine, s.avg_rating as dense_cluster_rating, s.sum_cuisine as sparse_count, d.avg_rating as
    sparse_cluster_rating, d.sum_cuisine as dense_count from "dense_cluster_vancouver" as d
INNER JOIN sparse_cuisines_rating_count as s ON s.cuisine=d.cuisine;
```





The 2 bar graphs compare the star rating and population of restaurants in the sparely populated and densely populated areas in Vancouver. The number of restaurants of almost every cuisine is much higher (R) which inversely affect the star rating (L). Almost every cuisine in the sparse cluster has a much higher rating – for instance Italian restaurants are rated 1 star higher on average in the spare cluster with only 6 Italian restaurants compared to the dense cluster with 135.

Learnings/Future Additions

- For this case study I chose to focus on two states, but it would be interesting to also look at trends across clusters in an expanded data set to see how the population distribution and ratings compare
- Adding another dimension to data analysis – weight rating with regards to other amenities restaurant offers like wifi, drinks, open 24 hours to see how it impacts rating
- Automating the querying + data visualization process as well to streamline the process even more