Group Members: Julian Rojas, Christian Anderson, Jordan Cline, Alyssa George

* **E**xtract: your original data sources and how the data was formatted (CSV, JSON, pgAdmin 4, etc).

In order to create our databases of New York City hotels, restaurants, and boroughs we used several sources of information. For the New York City hotels, we used a dataset from Data World that had hotels by borough (Data.world/city-of-ny/tjus-cn27), a csv from Kaggle with just zip codes and boroughs (https://www.kaggle.com/kimjinyoung/nyc-borough-zip), and the Yelp Master Fusion API. The Yelp Master Fusion API returned information from JSON which we used a Jupyter Notebook to then turn that into a csv.

* **T**ransform: what data cleaning or transformation was required.

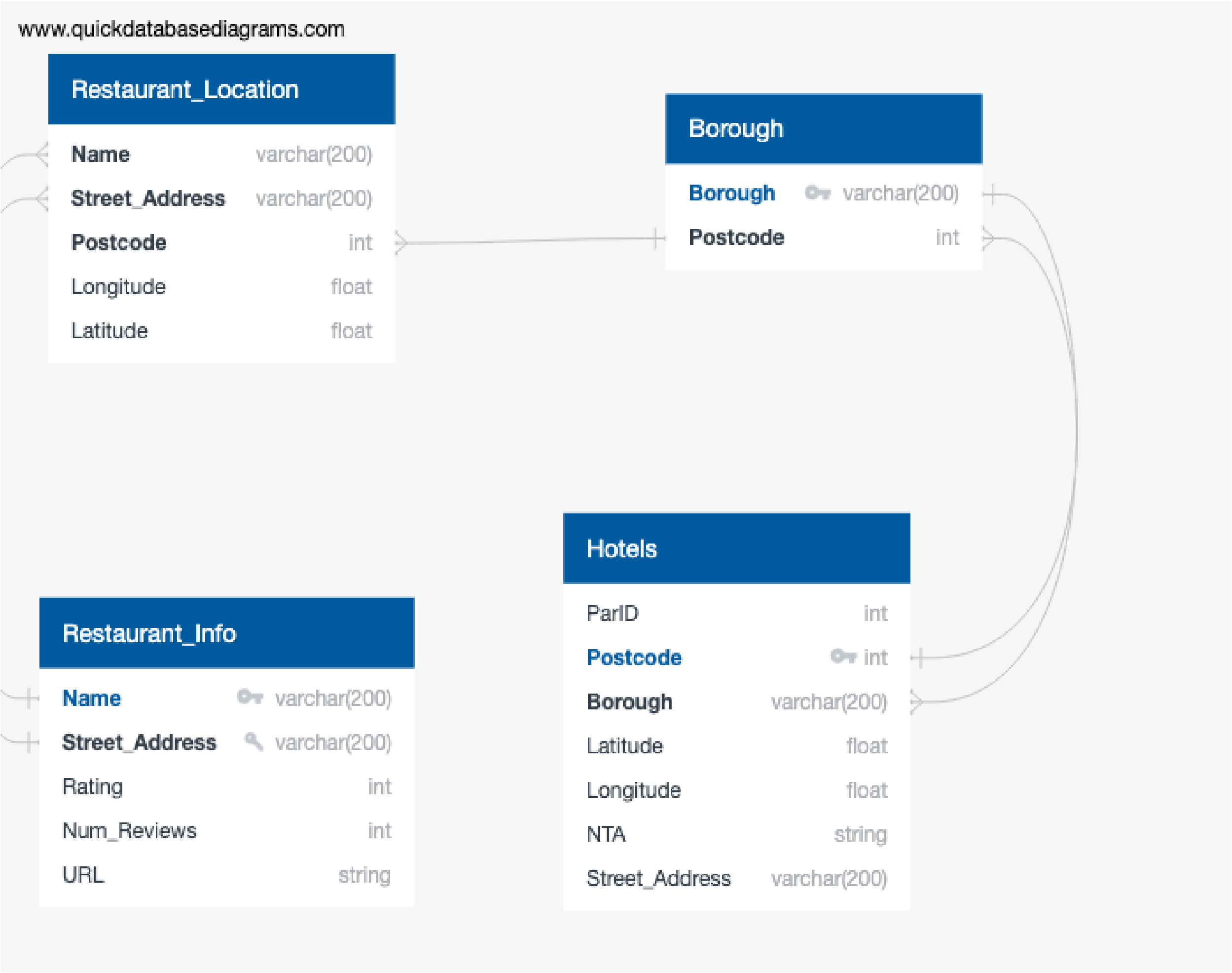
In order to clean the data up we followed several steps. The first step was to use the Yelp Master Fusion API – we were not able to loop the API in the time allotted so instead we manually looped the table 20 times to get the allowed 1000 restaurants. We chose to sort by rating in order to get the best 1000 restaurants in New York City according to the Yelp algorithm. Then all 20 csvs were merged in order to create one master CSV of data of all restaurants.

To clean/transform the restaurant and the hotel data, different extraneous columns were dropped and the columns were then renamed inside of pandas to create new csvs. For restaurants we then split the information into two different csvs, one with just location information and the other with general restaurant information like the rating, number of reviews, and url.

* **L**oad: the final database, tables/collections, and why this was chosen.

We built the final database in SQL using PostgreSQL. We chose to do a SQL database because all of the data is relational. The boroughs and zip codes connect to hotels, the zip codes connect to restaurants, hotels, and boroughs, the restaurant name and street name connect in both the restaurant info and restaurant location.

This database could be useful to someone in a number of ways. It could help them plan a food tour based on a zip code or based on a borough. It could help someone decide what borough to stay in depending on the restaurants or hotels. It could be improved by getting the count of hotels and restaurants in each borough. It could also be improved by writing a function that has the locations of every hotel and restaurant and helps identify how close they are to each other.



Restaurant_Location	
Name	varchar(200)
Street_Address	varchar(200)
Postcode	int
Longitude	float
Latitude	float

Borough		
Borough	🔑	varchar(200)
Postcode		int

Restaurant_Info		
Name	🔑	varchar(200)
Street_Address	🔧	varchar(200)
Rating		int
Num_Reviews		int
URL		string

Hotels		
ParID		int
Postcode	🔑	int
Borough		varchar(200)
Latitude		float
Longitude		float
NTA		string
Street_Address		varchar(200)

Fig. 1 : Example of ERD from SQL Database