TEAM:

MELINDA EUDY

KELLY ROSE

DARRIUS SUMMERS

PROJECT IDEA:

* Covid-19 has created a society with a bit of claustrophobia
* What will people want to do now that states are opening back up?
* Day trips? Weekend Trips?
* Distance? North and South Carolina
* Something new!
* Result:
  + Bars in NC and SC
  + What are they rated?
  + How expensive?
  + Reviews?

DATA SETS:

* Kaggle – Breweries and Beer Pubs in the us

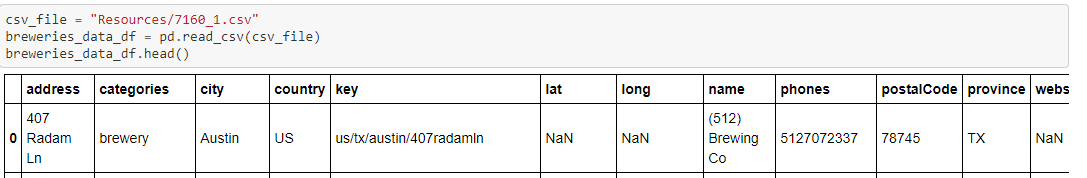


* ‘https://api.yelp.com/v3/businesses/search’
  + params = {'term':'bars','location':'NC'}
  + response=requests.get(url, params=params, headers=headers).json()
* ‘https://api.yelp.com/v3/businesses/search’
  + params = {'term':'bars','location':'SC'

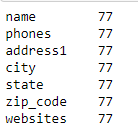
response=requests.get(url, params=params, headers=headers).json()

EXTRACTION – KAGGLE .CSV FILE

* Inspected File

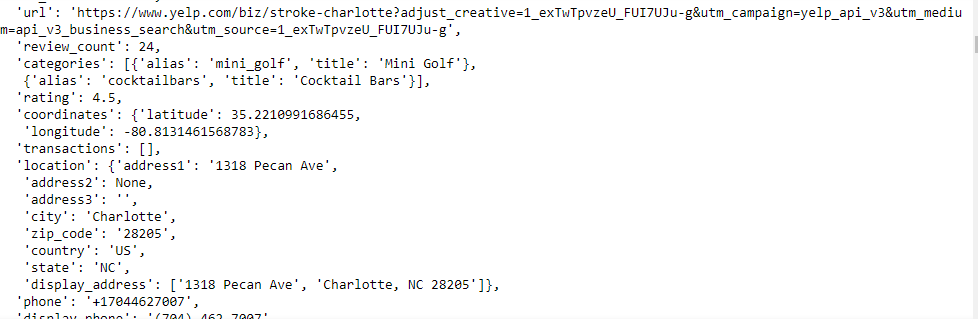


* Removed rows with NaN data
  + breweries\_data\_df.dropna(inplace=**True**)
* Removed columns not required for final data table
* Sorted the table into a readable order
  + **del** breweries\_data\_df["key"]
  + **del** breweries\_data\_df["long"]
  + **del** breweries\_data\_df["lat"]
  + **del** breweries\_data\_df["categories"]
  + breweries\_data\_df = breweries\_data\_df[["name", "phones", "address", "city", "province", "postalCode", "websites"]]
* Data set reduced to North and South Carolina only
  + options = ['NC', 'SC']
  + breweries\_data\_df = breweries\_data\_df[breweries\_data\_df['province'].isin(options)]
* Renamed columns
  + breweries\_data\_df=breweries\_data\_df.rename(columns ={"province": "state", "postalCode": "zip\_code", "address": "address1"})
* Final Count: 77 Bars in breweries\_data\_df



EXTRACTION – Yelp – North Carlina

* + headers = {'Authorization': 'Bearer **%s**' % api\_key}
  + url='https://api.yelp.com/v3/businesses/search''https://api.yelp.com/v3/businesses/search'
  + params = {'term':'bars','location':'NC'}
  + response=requests.get(url, params=params, headers=headers).json()
  + bars\_nc = json.dumps(response, indent = 4, sort\_keys=**True**)
  + response['businesses']
  + Inspect the Data



TRANSFORM – Yelp – North Carlina

* + Created a list of data needed for final DataFrame
    - bar\_id = []
    - address1 = []
    - price = []
    - rating = []
    - review\_count = []
    - **for** row **in** response['businesses']:

**try**:

bar\_id.append(row['id']),

address1.append(row['location']['address1']),

price.append(row['price']),

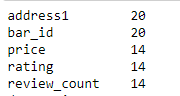
rating.append(row['rating']),

review\_count.append(row['review\_count'])

**except**:

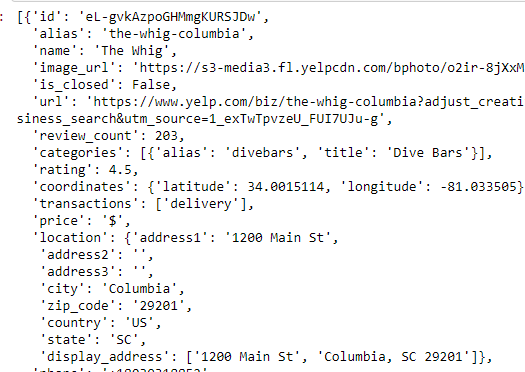
**pass**

* + Created a Pandas DataBase that does NOT remove rows with NaN (thank you Geronimo)
    - data = {'address1':address1, 'bar\_id': bar\_id, 'price': price, 'rating': rating, 'review\_count': review\_count}
    - bars\_nc\_df = pd.DataFrame(dict([ (k,pd.Series(v)) **for** k,v **in** data.items() ]))
  + Final Count: 20 Bars in North Carolina



EXTRACTION – Yelp – South Carlina

* + headers = {'Authorization': 'Bearer **%s**' % api\_key}
  + url='https://api.yelp.com/v3/businesses/search''https://api.yelp.com/v3/businesses/search'
  + params = {'term':'bars','location':'SC'}
  + response=requests.get(url, params=params, headers=headers).json()
  + bars\_sc = json.dumps(response, indent = 4, sort\_keys=**True**)
  + response['businesses']
  + Inspect the Data



TRANSFORM – Yelp – South Carlina

* + Created a list of data needed for final DataFrame
    - bar\_id = []
    - address1 = []
    - price = []
    - rating = []
    - review\_count = []
    - **for** row **in** response['businesses']:

**try**:

bar\_id.append(row['id']),

address1.append(row['location']['address1']),

price.append(row['price']),

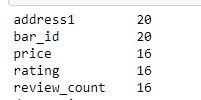
rating.append(row['rating']),

review\_count.append(row['review\_count'])

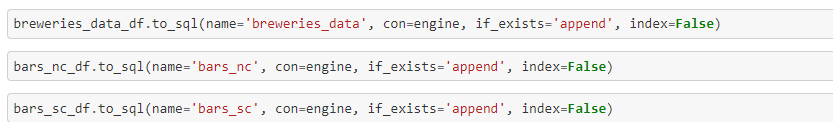
**except**:

**pass**

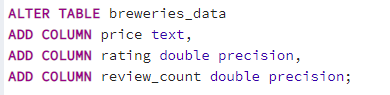
* + Created a Pandas DataBase that does NOT remove rows with NaN (thank you Geronimo)
    - data = {'address1':address1, 'bar\_id': bar\_id, 'price': price, 'rating': rating, 'review\_count': review\_count}
    - bars\_sc\_df = pd.DataFrame(dict([ (k,pd.Series(v)) **for** k,v **in** data.items() ]))
  + Final Count: 20 Bars in South Carolina



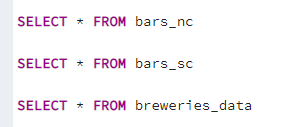
SEND TABLES TO POSTGRES (pgAdmin4):



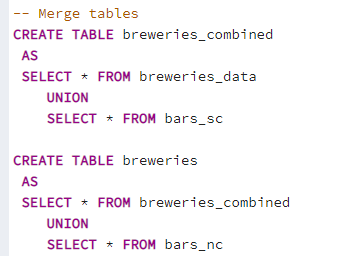
* Altered breweries\_data by adding three columns



* Confirmed tables were present in pgAdmin4

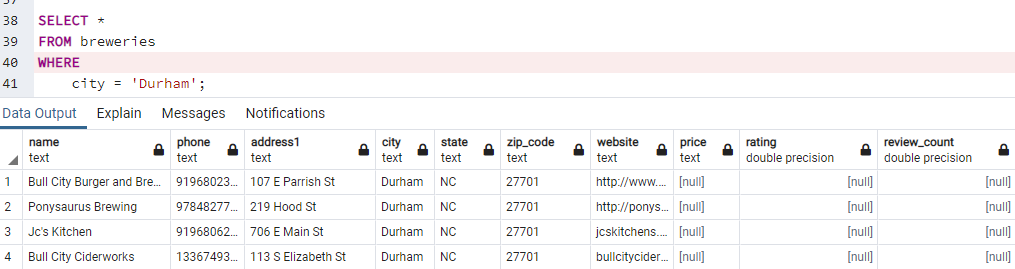


* Merged to final table: breweries



LOAD:

* Confirmed merged table
* Confirmed size of table : 117 rows



Conclusion:

* Confirmed size of table : 117 rows