HW8: Database Joins & Matplotlib

In this homework, you will select data from a database, process it, and create a visualization using Matplotlib. This is similar to the final steps of your pipeline for the final project.

We have provided:

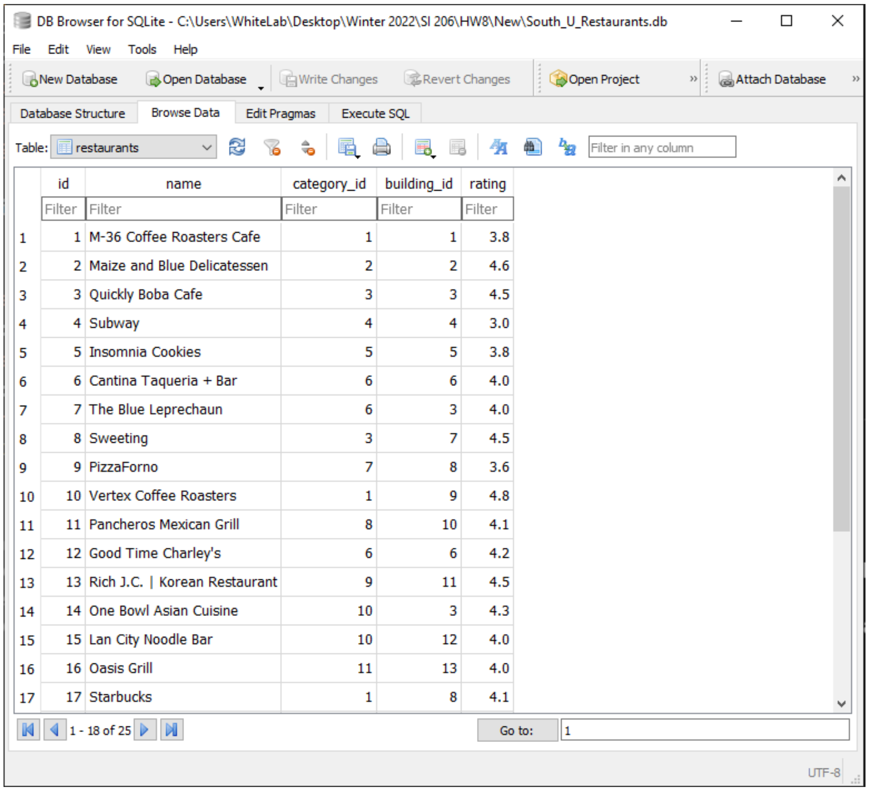
- *South_U_Restaurants.db* - a database with local restaurant data collected from Google.
- HW8.py - starter code for the functions below.

Make sure you are using Anaconda python for this assignment (preferred), or have installed Matplotlib on your own (using `pip install matplotlib` or another installation method). We have also provided test cases that will pass if the functions are written correctly. <u>You should not edit these test cases.</u> **Note:** It is okay for the extra credit test case to fail if you do not attempt the extra credit; you can also comment out those specific test cases.

Before you start: Look at the database

Check out *South_U_restaurants.db* in your DB Browser for SQLite program.

1. Open DB Browser for SQLite
2. Click on "Open Database" and choose South_U_*Restaurants.db*.
3. Click on Browse Data
4. Take some time to familiarize yourself with the table and column names



DB Browser for SQLite - C:\Users\WhiteLab\Desktop\Winter 2022\SI 206\HW8\New\South_U_Restaurants.db

Table: restaurants

	id	name	category_id	building_id	rating
	Filter	Filter	Filter	Filter	Filter
1	1	M-36 Coffee Roasters Cafe	1	1	3.8
2	2	Maize and Blue Delicatessen	2	2	4.6
3	3	Quickly Boba Cafe	3	3	4.5
4	4	Subway	4	4	3.0
5	5	Insomnia Cookies	5	5	3.8
6	6	Cantina Taqueria + Bar	6	6	4.0
7	7	The Blue Leprechaun	6	3	4.0
8	8	Sweeting	3	7	4.5
9	9	PizzaForno	7	8	3.6
10	10	Vertex Coffee Roasters	1	9	4.8
11	11	Pancheros Mexican Grill	8	10	4.1
12	12	Good Time Charley's	6	6	4.2
13	13	Rich J.C. | Korean Restaurant	9	11	4.5
14	14	One Bowl Asian Cuisine	10	3	4.3
15	15	Lan City Noodle Bar	10	12	4.0
16	16	Oasis Grill	11	13	4.0
17	17	Starbucks	1	8	4.1

1 - 18 of 25

Go to: 1

Part 1: Process the data

Complete the ***load_rest_data(db)*** function that accepts the filename of the database as a parameter, and returns a nested dictionary. Each outer key of the dictionary is the name of each restaurant in the database, and each inner key is a dictionary, where the key:value pairs should be the category, building, and rating for the restaurant. The dictionary should look like:

Expected return value:
{'M-36 Coffee Roasters Cafe': {'category': 'Cafe', 'building': 1101, 'rating': 3.8}, . . . }

Your function must pass all the unit tests to get full credit.

Note: Because all of the restaurants are on the same street (in this case, South University Ave), the addresses only contain the building numbers.

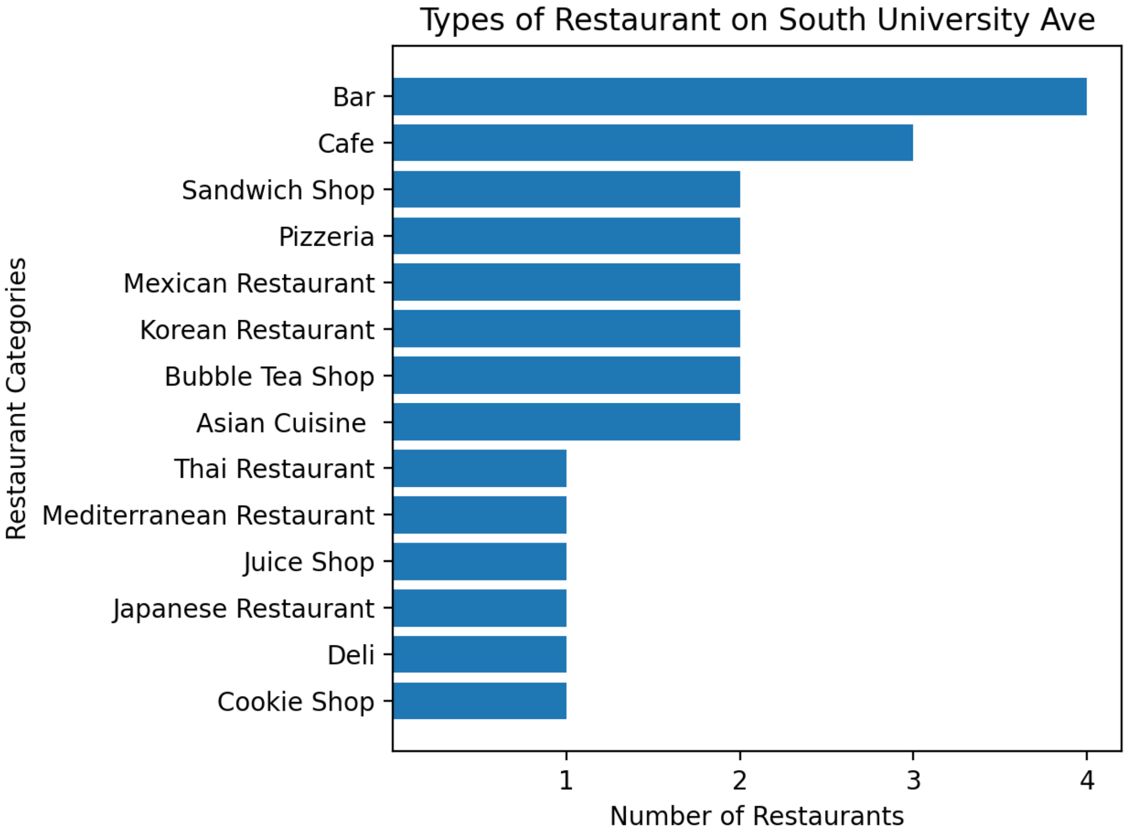
Part 2: Visualize the category data

Complete the function ***plot_rest_categories(db****)*, which accepts the filename of the database as a parameter and returns a dictionary. The keys should be the restaurant categories and the values should be the number of restaurants in each category (**hint:** use the SQL COUNT keyword).

Expected return value:
{'Asian Cuisine ': 2, 'Bar': 4, 'Bubble Tea Shop': 2, 'Cafe': 3, 'Cookie Shop': 1, 'Deli': 1, 'Japanese Restaurant': 1, 'Juice Shop': 1, 'Korean Restaurant': 2, 'Mediterranean Restaurant': 1, 'Mexican Restaurant': 2, 'Pizzeria': 2, 'Sandwich Shop': 2, 'Thai Restaurant': 1}

The function should also create a bar chart (horizontal or vertical – figure out which one gives a better visualization) with restaurant categories along one axis and the counts along the other axis. In the chart, the counts should be descending from one side to the other.

Example chart:



Submit an image file of your bar chart to Canvas, along with your repository link.

Part 3: Find restaurants in a specified building

Complete the function ***find_rest_in_building(building_num, db)***, which accepts the building number and the filename of the database as parameters and returns a list of restaurant names. You need to find all the restaurant names which are in the specific building. The restaurants should be sorted by their **rating** from highest to lowest (**hint:** Use SQL WHERE keyword).

For example, for building number 1140, the expected return value is:

['BTB Burrito', "Good Time Charley's", 'Cantina Taqueria + Bar']

Extra Credit: Visualize more data

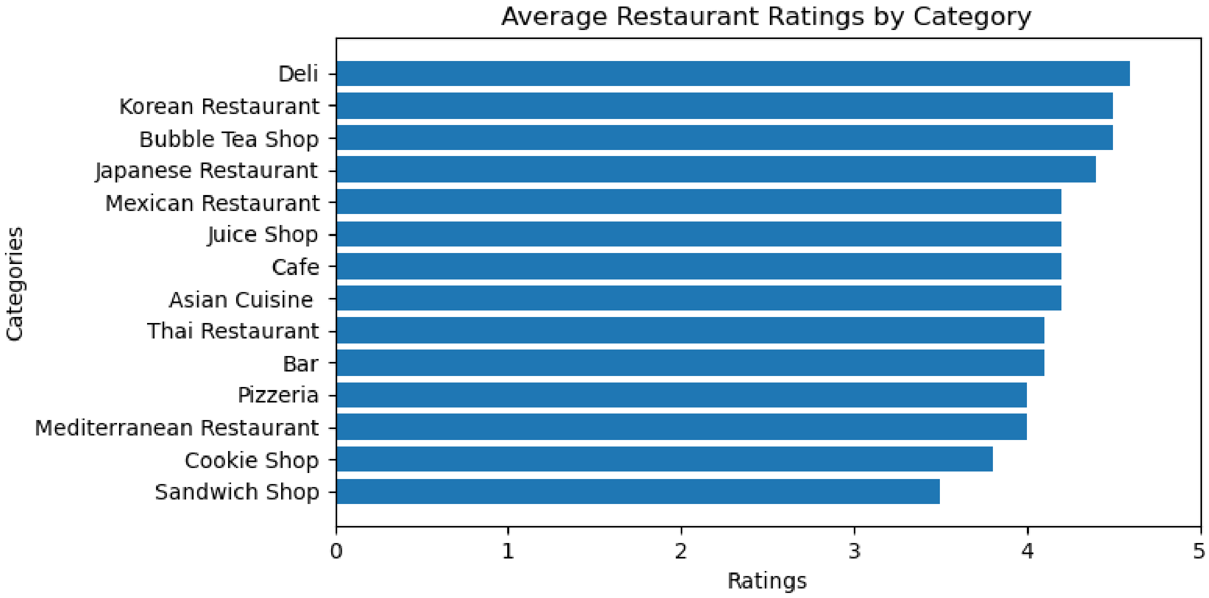
Let’s write a function to determine which category and building have (on average) the highest-rated restaurants.

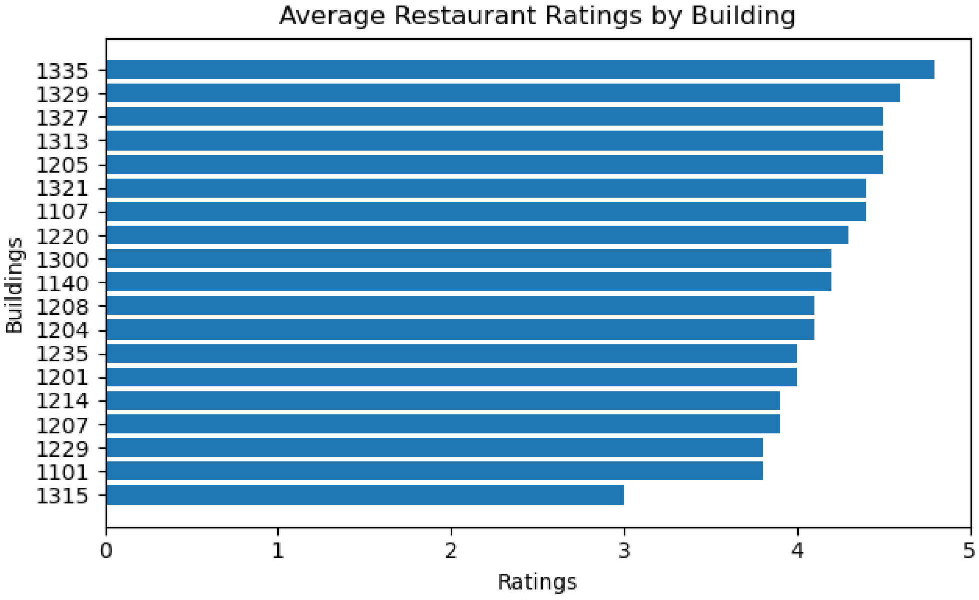
Complete function ***get_highest_rating(db)*** to plot two bar charts in one figure using `plt.subplot()`.

For the first bar chart, the y-axis will be different categories of restaurants. The x-axis will be the average rating for the restaurants in each type (**hint**: use the AVG keyword when writing your query). The average values should be rounded to one decimal place. Sort the y-axis in **descending order** from top-to-bottom by rating.

For the second bar chart, the y-axis will be different building numbers. The x-axis will be the average rating for the restaurants in each building. The average values should also be rounded to one decimal place, and the y-axis should be sorted in **descending order** by rating.

The chart must have appropriate axis labels and a title. The limit of the x-axis should be **0 - 5** for both charts. You can use `plt.figure(figsize=(8,8))` to adjust the size of the figure. Your chart should look like this:





Finally, this function should return a list of two tuples. The first tuple contains the highest-rated restaurant category and the average rating of the restaurants in that category, and the second tuple contains the building number which has the highest rating of restaurants and its average rating.

Expected Output:
[('Deli', 4.6), (1335, 4.8)]

Grading

Code passes all unit tests	18 pts (2 pts per unit test with 9 tests)
Submission of bar chart image file	5 pts
Created a bar chart from the data	22 pts
Title on bar chart	5 pts
Informative X-axis label on bar chart	5 pts
Informative Y-axis label on bar chart	5 pts
Correct code and image file for extra credit	*10 pts extra credit*
Total	**60 pts + 10 pts extra credit**