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:

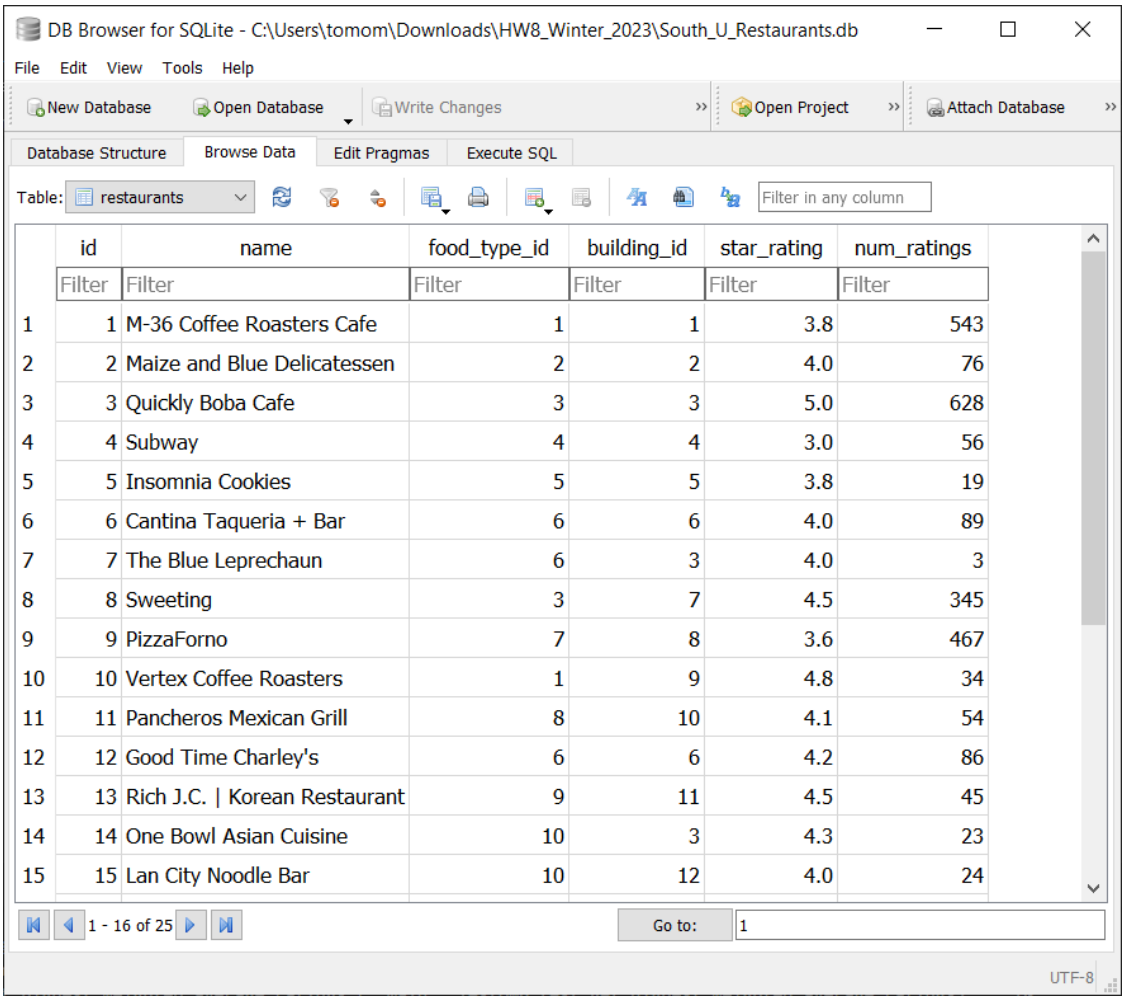
- *South_U_Restaurants.db* - a database with local restaurant altered data collected from Google.
- HW8.py - starter code for the functions below.

Make sure you are using Anaconda ("base":conda) 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. You should not edit these test cases**.** **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.



	id	name	food_type_id	building_id	star_rating	num_ratings
	Filter	Filter	Filter	Filter	Filter	Filter
1	1	M-36 Coffee Roasters Cafe	1	1	3.8	543
2	2	Maize and Blue Delicatessen	2	2	4.0	76
3	3	Quickly Boba Cafe	3	3	5.0	628
4	4	Subway	4	4	3.0	56
5	5	Insomnia Cookies	5	5	3.8	19
6	6	Cantina Taqueria + Bar	6	6	4.0	89
7	7	The Blue Leprechaun	6	3	4.0	3
8	8	Sweeting	3	7	4.5	345
9	9	PizzaForno	7	8	3.6	467
10	10	Vertex Coffee Roasters	1	9	4.8	34
11	11	Pancheros Mexican Grill	8	10	4.1	54
12	12	Good Time Charley's	6	6	4.2	86
13	13	Rich J.C. | Korean Restaurant	9	11	4.5	45
14	14	One Bowl Asian Cuisine	10	3	4.3	23
15	15	Lan City Noodle Bar	10	12	4.0	24

Part 1: Process the data

Complete the ***load_restaurant_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 *food_type*, *building_number*, *star_rating*, and *num_reviews* for the restaurant. The dictionary should look like:

Expected return value:
{‘M-36 Coffee Roasters Cafe’: {‘food_type: ‘Cafe’, ‘building_number’: 1101, ‘star_rating’: 3.8, 'num_ratings': 543}, . . . }

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

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

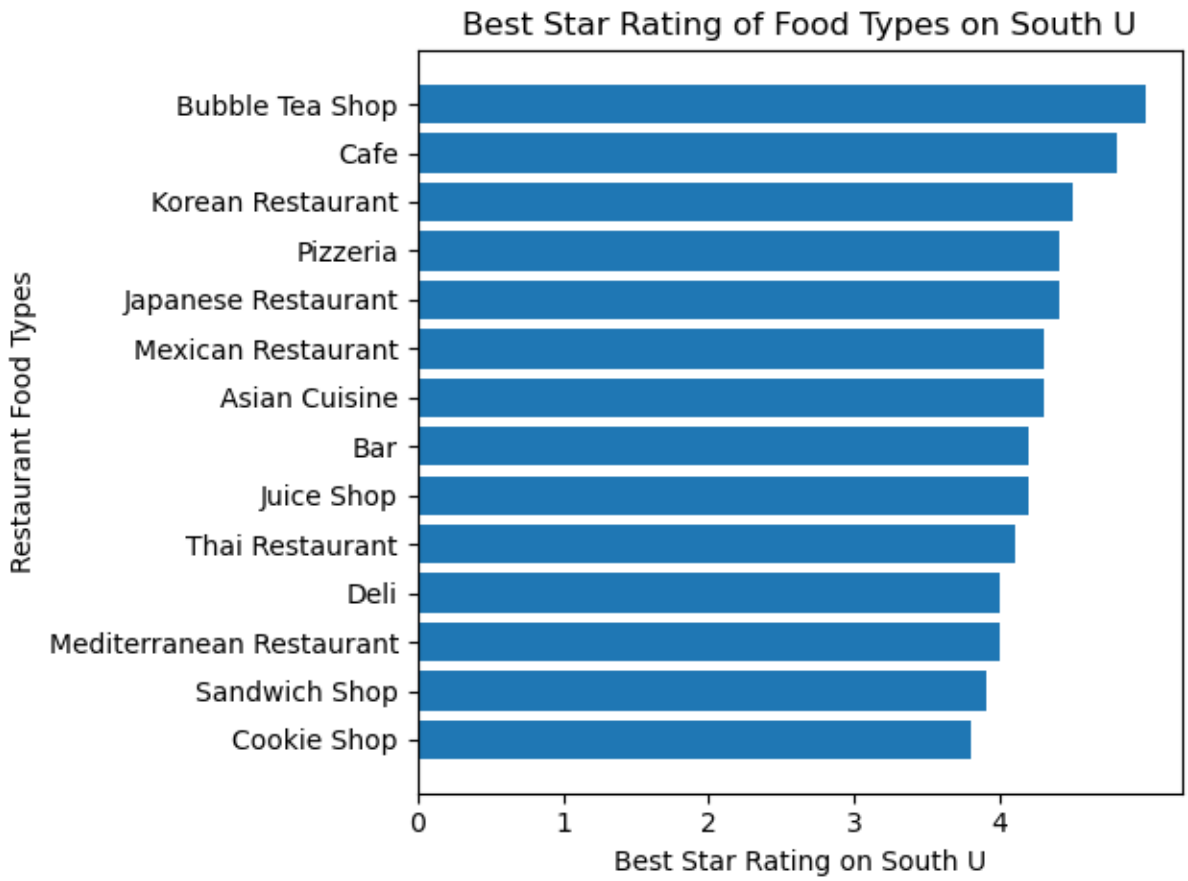
Part 2: Visualize the Food_ Type data

Complete the function ***plot_best_star_ratings_by_food_type(db)***, which accepts the filename of the database as a parameter and returns a dictionary. The keys should be the restaurant *food_types* and the values should be the corresponding highest *star_rating* of the restaurants of that *food_type* (**hint:** use the SQL MAX keyword).

Expected return value:
{'Bubble Tea Shop': 5.0, 'Cafe': 4.8, 'Korean Restaurant': 4.5, 'Pizzeria': 4.4, 'Japanese Restaurant': 4.4, 'Mexican Restaurant': 4.3, 'Asian Cuisine': 4.3, 'Bar': 4.2, 'Juice Shop': 4.2, 'Thai Restaurant': 4.1, 'Deli': 4.0, 'Mediterranean Restaurant': 4.0, 'Sandwich Shop': 3.9, 'Cookie Shop': 3.8}

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

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_restaurants_in_building(building_number, 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 *star_rating* from highest to lowest (**hint:** Use the 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 *food_type* and *building_number* have the highest weighted average *star_rating* for restaurants. For our calculations, the weight (w_i) of each restaurant's rating will be the number of ratings (*num_ratings*) that restaurant received. **Hint: the weighted average can be calculated entirely in your SQL query, although this is not required. Some SQL functions you may find useful are SUM() and ROUND().**

Formula

$$W = \frac{\sum_{i=1}^{n} w_i X_i}{\sum_{i=1}^{n} w_i}$$

W = weighted average

n = number of terms to be averaged

w_i = weights applied to x values

X_i = data values to be averaged

For an example of the weighted average, take the *food_type* "Bubble Tea Shop". There are two restaurants of this *food_type*:

- Quickly Boba Cafe with a *star_rating* of 5.0 and a *num_ratings* of 628
- Sweeting with a *star_rating* of 4.5 and a *num_ratings* of 345

This means that the weighted average rating for the *food_type* "Bubble Tea Shop" is:

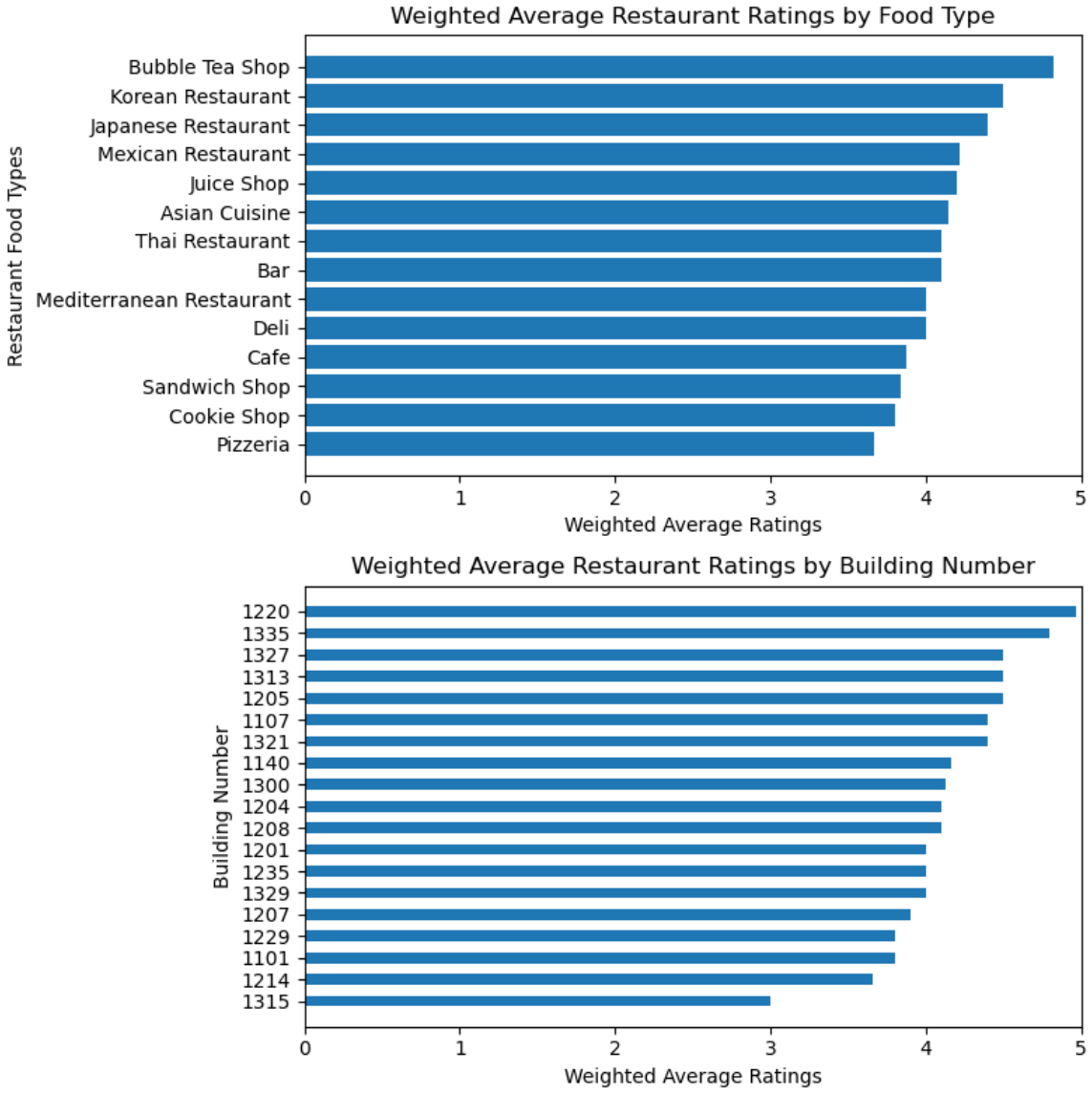
$$\frac{(5.0\times628)+(4.5\times345)}{(628)+(345)} = 4.82$$

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

For the first bar chart, the y-axis will be different *food_type* of each restaurant. The x-axis will be the weighted average *star_rating* for the restaurants of each *food_type*. The average values should be rounded to two decimal places. 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 weighted average *star_rating* for the restaurants in each building. The average values should also be rounded to two decimal places, 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 *food_type* and its weighted average of restaurants, and the second tuple contains the highest rated *building_number* and its weighted average of restaurants.

Expected Output:
[('Bubble Tea Shop', 4.82), (“1220”, 4.97)]

Grading

load_restaurant_data(db)	10 pts
plot_best_star_ratings_by_food_type(db)	10 pts
find_restaurants_in_building(building_number, db)	10 pts
Submission of bar chart image file	5 pts
Created a bar chart from the data	10 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
get_highest_weighted_average_ratings(db) *Correct code and image file for extra credit*	*6 pts extra credit*
Total	**60 pts + 6 pts extra credit**