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:

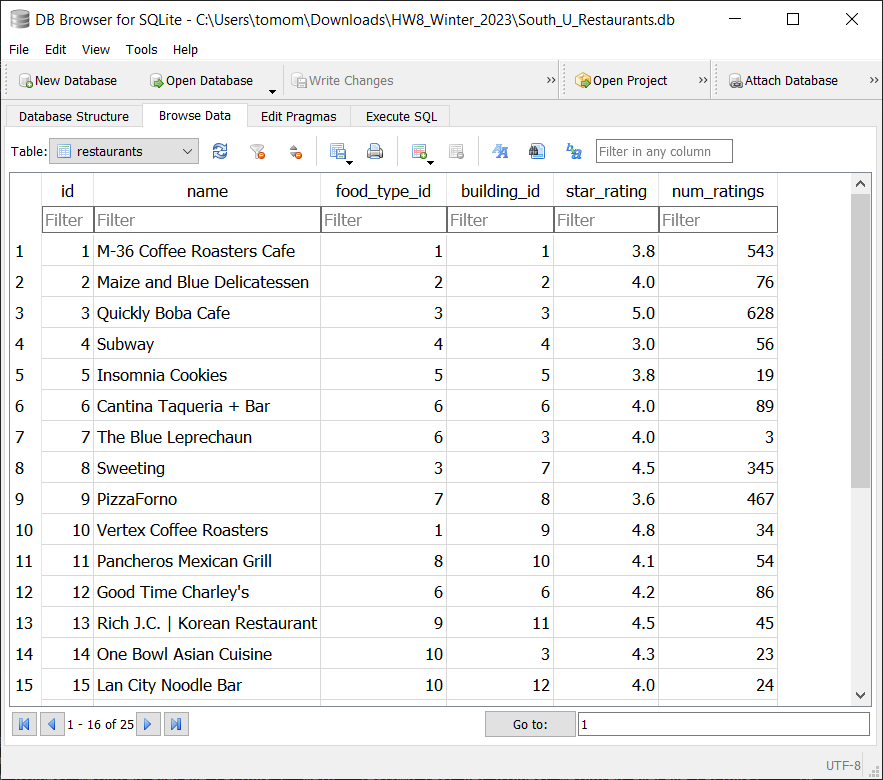
* *South\_U\_Restaurants.db* - a database with local restaurant altered data collected from Google.
* HW8.py - starter code for the functions below.

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.



# 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

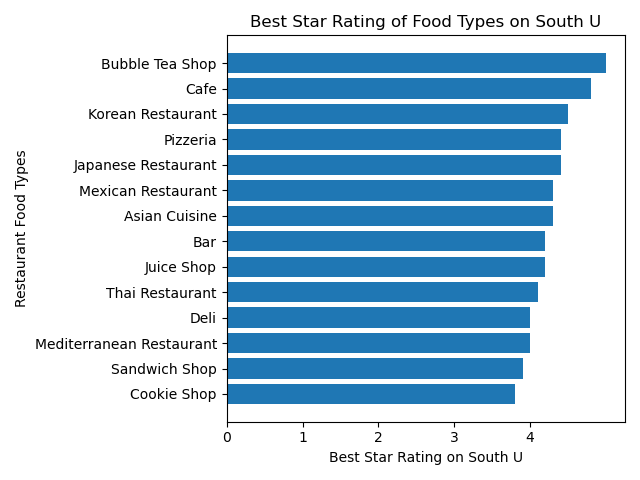
Complete the function ***plot\_best\_star\_ratings\_by\_food\_type(db****)*, which accepts the filename of the database as a parameter and returns a dictionary. The keys should be the restaurant *food\_types* and the values should be the corresponding highest *star\_rating* of the restaurants of that *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](https://en.wikipedia.org/wiki/Weighted_arithmetic_mean) *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()**](https://www.w3resource.com/sqlite/aggregate-functions-and-grouping-sum.php) **and** [**ROUND()**](https://www.w3resource.com/sqlite/core-functions-round.php)**.**

A math formula with numbers and symbols

Description automatically generated

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

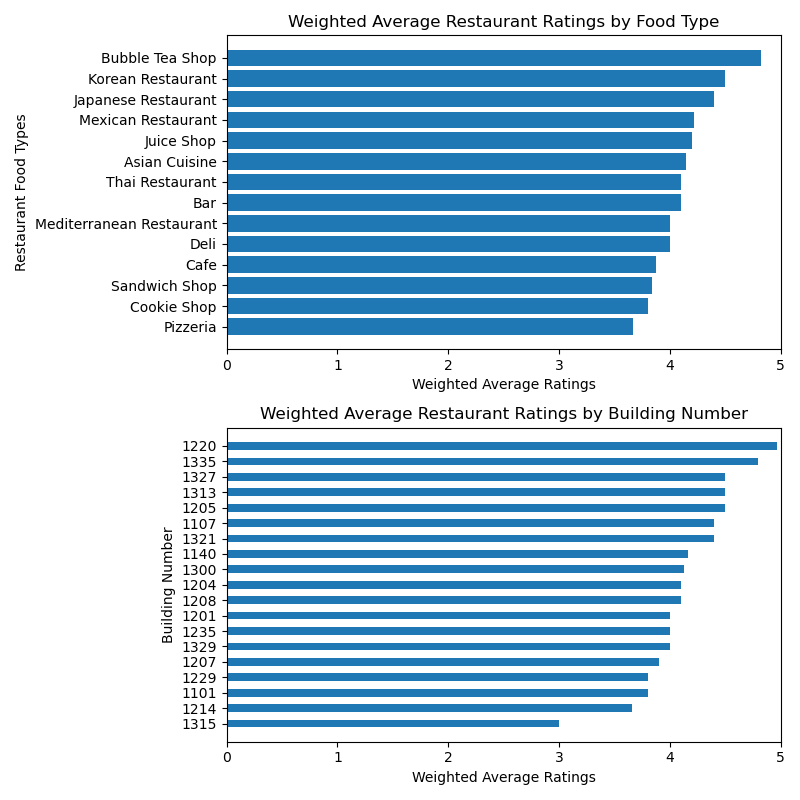
This means that the weighted average rating for the *food\_type* “Bubble Tea Shop” is:

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** |