206 Final report: Fiery Latinas

Marcela Passos
Carolina Janicke

1. Our goal

   We want to examine how the response time for fires between the cities of Los Angeles and New York differ during similar times of the day. Additionally, we also want to determine if certain neighborhoods in NYC have more fires on average. Our database contains 4 tables, two of which have fire ID, response time from the fire department, and time of day that the fires were reported as columns.

2. What we achieved

   We were able to use two API's to gather our data into tables with 100 entries
   Created 4 tables- fire ID, response time from the fire department, and time of day that the fires were reported as columns.
   We converted time variables to only have 4 digits (hr and minutes) to be more easily read and understood
   Using the fire_data.db we were able to calculate the average number of fires in different NYC neighborhoods, and the average response time for fires during two hour periods during the day
   We used MatPlotLib to create a bar graph and line graphs to visualize and compare the data

3. Problems we faced

   Finding two API's with data that was relevant to one another
   It was hard for us to ensure that each table in our database had exactly 100 data points with a separation of 4 batches of 25
   We struggled with successfully parsing through the NYC_data.json because of the way it was formatted after getting data from the NYC API. Once we figured out that it was a list of

lists of dictionaries than we were able to successfully parse through the Json file

Figuring out how to calculate the average fire response for non-overlapping 2 hour periods was also a challenge

Fixing our function calculations to ensure the averages were mathematically correct

4. Our calculations

   LA- The average response time of the firefighters to get on scene for every 2 hour period

   NYC- The average number of fires per neighborhood

   Both- Comparing the average response time of firefighters between LA and NYC

5. How to run our code

   Set Up Database: Connect to the "fire_data.db" database and create the first table.

   Insert Data: Load data from "LA_data.json" and insert it into the database table.

   Calculate Averages: Calculate the average response time per period.

   Close Database Connection: Close the database connection.

   Handle Errors: Catch and print any errors that occur during execution.

6. Breakdown of our functions

   LA

   1. get_fire_dep_data(classification_group)
      - Fetches fire department data from LA city government's dataset.
      - Appends received data to "LA_data.json" if available.
      - Handles HTTP and JSON decoding errors.

   2. set_up_database(db_name)
      - Sets up an SQLite database
      - Returns cursor (`cur`) and connection (`conn`) objects.

3. create_first_table(cur, conn)
   - Creates the "LA_Fires" table in the database.
   - Table has columns for fire ID, time, and response time.

4. insert_data_to_fires_table(cur, conn)
   - Inserts fire data from the LA dataset into the database.
   - Limits entries to 100 and 8 per hour.
   - Calculates response time and adds to the table.

5. calculate_avg_response_time_per_period(cur, conn)
   - Calculates average response time per 2-hour period.
   - Writes results to "calculations.txt".

NYC

1. get_fire_data(classification_group)
   - Retrieves structural fire data from the NYC government API.
   - Appends the received data to "NYC_data.json".
   - Handles HTTP errors and JSON decoding errors.

2. set_up_database(db_name)
   - Sets up an SQLite database connection using the provided name.
   - Drops existing tables if they exist to ensure a clean start.

3. delete_database(db_name)
   - Deletes the specified database file if it exists.

4. create_first_nyc_table(cur, conn)
   - Creates the "NYC_Fires" table in the database.
   - Table includes columns for fire ID, date, time, and response time.

5. create_neighborhood_table(cur, conn)
   - Creates the "neighborhood_ID" table in the database.
   - Table includes columns for neighborhood ID and neighborhood name.

6. create_fire_neighborhood_relationship_table(cur, conn)
   - Creates the "Fire_Neighborhood_Relationship" table in the database.
   - Table establishes a relationship between fires and neighborhoods.

7. insert_data_to_fires_table(cur, conn, json_data)
   - Inserts fire data into the "NYC_Fires" table.
   - Processes the data in batches to handle large datasets.

8. insert_data_to_neighborhood_table(cur, conn, json_data)
   - Inserts neighborhood data into the "neighborhood_ID" table.
   - Establishes relationships between fires and neighborhoods.

9. calculate_avg_fires_per_neighborhood(cur, conn)
   - Calculates the average number of fires per neighborhood.
   - Write results to "calculations.txt".

10. calculate_avg_response_time_per_period(cur, conn)
   - Calculates the average response time per 2-hour period in NYC.
   - Write results to "calculations.txt".

7. Our comparison
   a.
8. Resources
   ChatGPT- we used chat gpt to debug our code, to help structure our functions, and make them more efficient. It helped us a lot in

figuring out how to limit our database to 100 rows for each table and how to use the datetime import to convert the string data into a DateTime object to make calculations.

https://dev.socrata.com/docs/filtering.html -
https://www.geeksforgeeks.org/saving-api-result-into-json-file-in-python/ -

https://data.cityofnewyork.us/Public-Safety/Fire-Incident-Dispatch-Data/8m42-w767/about_data

https://data.lacity.org/Public-Safety/LAFD-Response-Metrics-Raw-Data/n44u-wxe4/data_preview

https://data.lacity.org/Public-Safety/Station-Response-Metrics-Updated/4hqz-rxff/data_preview
https://www.dallasopendata.com/widgets/skju-bfp8?mobile_redirect=true

These resources helped us get the APis, understanding Filtering & Querying, Saving API Result Into JSON File in Python.

https://github.com/marce113/SI206FinalProj