Homework 2

About the Data

The data is originally from DataSF : Police Department Incident Reports: 2018 to Present.

The Data directory includes four csv files.

incident_type.csv

- It includes records of Incident Code, Incident Category, Incident Subcategory, and Incident Description

report_type.csv

- It includes records of Report Type Code, and Report Type Description

location.csv

- It includes records of Longitude, Latitude, Supervisor District, Police District, and Analysis Neighborhood

incident.csv

- It includes records of Incident ID, Incident Datetime, Report Datetime, Longitude, Latitude, Report Type Code, and Incident Code

About the Tasks

In this assignment, we will extend Homework 1, including more advanced queries and create indexes.

Questions

1. Complete the `select_all()` decorator, which 1) retrieve keyword arguments including `user`, `host`, and `dbname`, 2) executes a SQL query string returned from a function and 3) returns the output. (1 pt)

Example:

```
@select_all
def return_incident_category_count(**kargs):
    # complete here

return_incident_category_count(user='postgres', host='127.0.0.1', dbname='msds691_HW', n=
5)
```

This should return:

```
[('Other Miscellaneous', 101), ('Larceny Theft', 90), ('Robbery', 72), ('Drug Offense', 6
0), ('Burglary', 52)]
```

2. Complete the `return_incident_category_count()` function. This function returns the registered incident category and the number of incident types that belong to that category. This function connects to a database using the parameters `user`, `host`, `dbname`, and `n`, and retrieves `n` records of `incident_category` along with their corresponding count from the `incident_type` table. The function only retrieves records where `incident_category` is not null, and orders them by count in descending order. If there are rows with the same count, the function sorts them alphabetically by `incident_category` in ascending order. If the parameter `n` is not provided, the function returns all rows. (0.5 pt)

Example:

```
return_incident_category_count(user=user, host=host, dbname=dbname, n=10)
```

This should return:

```
[('Other Miscellaneous', 101), ('Larceny Theft', 90),
 ('Robbery', 72), ('Drug Offense', 60),
 ('Burglary', 52), ('Assault', 42),
 ('Fraud', 41), ('Disorderly Conduct', 38),
 ('Other Offenses', 35), ('Malicious Mischief', 29)]
```

3. Complete the `return_incident_count_by_category_subcategory()` function.The purpose of this function is to calculate the occurrence of incidents belonging to a `incident_category` and `incident_subcategory`. This function connects to the database using the provided `user`, `host`, `dbname`, `count_limit`, and `n` parameters. It returns `n` records of `incident_category`, `incident_subcategory`, and their count (occurrence) in

the `incident` table where the count is greater than `count_limit`. The output is ordered by occurrence in descending order. If there are records with the same count value, they are ordered by `incident_category` alphabetically (ascending). (0.5 pt)

Example:

```
return_incident_count_by_category_subcategory(user=user,
                                              host=host,
                                              dbname=dbname,
                                              count_limit=20000)
```

This should return:

```
[('Larceny Theft', 'Larceny - From Vehicle', 129025),
 ('Larceny Theft', 'Larceny Theft - Other', 50955),
 ('Malicious Mischief', 'Vandalism', 42180),
 ('Motor Vehicle Theft', 'Motor Vehicle Theft', 38329),
 ('Recovered Vehicle', 'Recovered Vehicle', 29587),
 ('Other Miscellaneous', 'Other', 23526),
 ('Non-Criminal', 'Non-Criminal', 22619),
 ('Assault', 'Simple Assault', 22155),
 ('Lost Property', 'Lost Property', 21441)]
```

4. Complete the `return_count_by_location_report_type_incident_description()` function. This function aims to find the monthly count of incidents in a registered location.This function connects to the database using the given `user`, `host`, `dbname`, `year`, and `n` parameters, and returns an output of `n` rows (if `n` is given) or all rows of the following columns: `year` (extracted from `incident_datetime`), `month` (also extracted from `incident_datetime`), `longitude`, `latitude`, `neighborhood`, `report_type_description`, `incident_description`, and the corresponding count, which is ordered by count in descending order, and then by `year`, `month`, `longitude`, `latitude`, `report_type_description`, and `incident_description` in ascending order. (1.5 pt)

Example:

```
return_count_by_location_report_type_incident_description(user=user,
                                                          host=host,
                                                          dbname=dbname,
                                                          year=2022,
                                                          n=10)
```

This should return:

```
[(2022, 7, -122.41349, 37.77999, 'Tenderloin', 'Initial', 'Narcotics Paraphernalia, Posses
sion of', 84),
 (2022, 4, -122.40371, 37.784046, 'Financial District/South Beach', 'Initial', 'Theft, Sho
plifting, $50-$200', 70),
 (2022, 8, -122.41349, 37.77999, 'Tenderloin', 'Initial', 'Narcotics Paraphernalia, Posses
sion of', 70),
 (2022, 2, -122.40371, 37.784046, 'Financial District/South Beach', 'Initial', 'Theft, Sho
plifting, <$50', 58),
 (2022, 3, -122.40371, 37.784046, 'Financial District/South Beach', 'Initial', 'Theft, Sho
plifting, <$50', 55),
 (2022, 2, -122.40371, 37.784046, 'Financial District/South Beach', 'Initial', 'Theft, Sho
plifting, $50-$200', 53),
 (2022, 3, -122.40371, 37.784046, 'Financial District/South Beach', 'Initial', 'Theft, Sho
plifting, $50-$200', 52),
 (2022, 11, -122.422005, 37.805496, 'Russian Hill', 'Initial', 'Theft, From Locked Vehicl
e, >$950', 45),
 (2022, 12, -122.422005, 37.805496, 'Russian Hill', 'Initial', 'Theft, From Locked Vehicl
e, >$950', 44),
 (2022, 1, -122.40371, 37.784046, 'Financial District/South Beach', 'Initial', 'Theft, Sho
plifting, <$50', 40)]
```

5. Complete the `return_avg_interval_days_per_incident_code()` function. This function calculates the average number of days taken between `incident_datetime` and `report_datetime` for each `incident_code`. Using `user`, `host`, `dbname`, and `n`, this function connects to the database and returns `n` rows of `incident_code`, `incident_description`, and `avg_interval_days`, where `avg_interval_days` is the average difference between `report_datetime` and `incident_datetime` extracted as days.
   The output should be ordered by `avg_interval_days` in descending order. If there are multiple rows with the same `avg_interval_days`, order by `incident_code` in ascending order.
   If `n` is not given, it returns all the rows. (1 pt)

Example:

```
return_avg_interval_days_per_incident_code(user=user,
                                           host=host,
                                           dbname=dbname,
                                           n=10)
```

This should return:

```
[(26160, 'Perjury', 535),
 (9161, 'Crimes Against Revenue & Property of State', 521),
 (10035, 'Embezzlement, Grand Theft By Collector', 438),
 (6141, 'Theft, By Prostitute, <$50', 378),
 (13080, 'Human Trafficking (B)-Involuntary Servitude (not sex-related)', 315),
 (10025, 'Embezzlement, Grand Theft By Brokers/Agents', 252),
 (7203, 'VIN Switch', 178),
 (10075, 'Embezzlement, Grand Theft Private Property', 176),
 (74022, 'SFMTA Employee-Non Operator/Station Agent-Other Employee', 169),
 (9060, 'Real Estate Fraud', 167)]
```

6. Complete the `return_monthly_count()` function. This function returns the number of incidents in each month of each year.

   Using `user`, `host`, `dbname`, and `n`, this function connects to the database and returns `n` rows of `year`, `jan`, `feb`, `mar`, `apr`, `may`, `jun`, `jul`, `aug`, `sep`, `oct`, `nov` and `dec`, where each column includes the number of incidents for the corresponding year and month, ordered by year in ascending order. If `n` is not given, it returns all the rows. (1.5 pt)

Example:

```
return_monthly_count(user=user, host=host, dbname=dbname, n=2)
```

This should return:

```
[(2018, 11032, 9599, 10355, 10283, 10614, 10212, 11361,
  11384, 10441, 10860, 9776, 10146),
 (2019, 9875, 8920, 9616, 9594, 9916, 9978, 10792, 11299,
  10662, 11226, 10002, 10344)]
```

7. Assuming that the query `return_count_by_location_report_type_incident_description()` (Q4) is the most frequently used query in your database, complete `create_index()` which creates indexes to improve its performance by at least 10%.
   For this question, you can assume that there will be no insertions or updates made to the database afterwards.
   Using streamlit, the create_index will display the query improvement after you enter the absolute path of the data directory. (1.5 pt)

8. Make sure all the code follows PEP8 standards. If you run pycodestyle hw2.py, it should not return any output. (0.5 pt)