Homework 1

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 complete hw1.py, which performs SQL queries on the PostgreSQL database using psycopg. Psycopg is a Python package that connects Postgres with Python, while you can use both SQL and Python to complete your application.
Using user, host, and dbname (let's assume there is no password required), you can 1) create a connection and then 2) a cursor to execute and fetch/commit your query. Please see this basic example. Please note that a commit operation saves the updates done through your query permanently on your database.

Required:

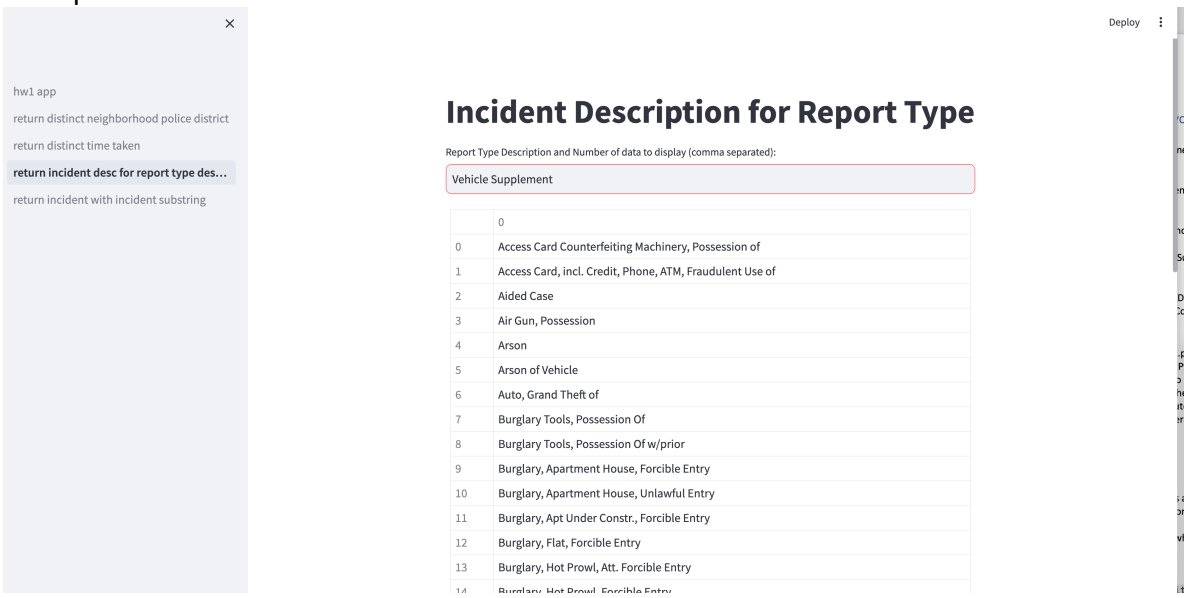
```
conda install -c conda-forge psycopg
```

Furthermore, we will extend hw1_app.py, which is a Streamlit application. Streamlit is a package that you can easily create a data science application, and you can simply execute it by "`streamlit run hw1_app.py`".
Please see this basic example to extend the app, where you will complete .py files in the pages folder.

Required:

```
conda install -c conda-forge streamlit
```

Example:



Questions

1. Complete drop_tables().
   This function connects to a database using user, host, and dbname, and drops all the tables, including report_type, incident_type, location, and incident.
   This function should work regardless of the existence of the tables without any errors.

2. Complete create_tables().
   For the given user, host, and dbname, this function connects to a database and creates four different tables, including report_type, incident_type, location, and incident.
   The schema of each table is the following.
   report_type
   - report_type_code - varying character (length of 2), not null
   - report_type_description - varying character (length of 100), not null
   - primary key - report_type_code

   incident_type
   - incident_code - integer, not null
   - incident_category - varying character (length of 100), null
   - incident_subcategory - varying character (length of 100), null
   - incident_description - varying character (length of 200), null
   - primary key - incident_code

   location
   - longitude - real, not null
   - latitude - real, not null
   - supervisor_district - real, null

- police_district - varying character (length of 100), not null
- neighborhood - varying character (length of 100), null
- primary key – (longitude, latitude) pair

incident

- id - integer, not null
- incident_datetime - timestamp, not null
- report_datetime - timestamp, not null
- longitude - real, null
- latitude - real, null
- report_type_code - varying character (length of 2), not null
- incident_code - integer, not null
- primary key - id
- foreign key - report_type_code, incident_code, and (longitude, latitude) pair. Those values should be updated when referenced values are updated.

3. Complete copy_data().
   Using user, host, dbname, and dir, this function connects to the database and loads data to report_type, incident_type, location, and incident tables from report_type.csv, incident_type.csv, location.csv and incident.csv located in dir.
   Note: Each file includes a header.

4. Complete return_distinct_neighborhood_police_district().
   Using user, host, dbname, dir, and n, this function connects to the database and returns n unique rows of neighborhood and police_district in the location table, where the neighborhood is not null.
   The returned output should be ordered by neighborhood and police_district in ascending order.
   If n is not given, it returns all the rows.

5. Complete return_distinct_time_taken().
   Using user, host, dbname, dir, and n, this function connects to the database and returns n unique differences between report_datetime and incident_datetime in days in descending order.
   If n is not given, it returns all the rows.

6. Complete return_incident_with_incident_substring().
   Using user, host, dbname, dir, substr, and n, this function connects to the database and returns n unique id and incident_datetime in the incident table where its incident_code corresponds to the incident_description that includes substr, a substring ordered by id in ascending order.
   The search for the existence of the given substring, substr should be case-insensitive.
   If n is not given, it returns all the rows.

7. Complete return_incident_desc_for_report_type_desc().
   This function connects to the database using user, host, dbname, dir, substr, and n. It returns n unique incident_description in the incident_type table where report_type_code of any incidents in the incident table corresponds to desc, a report_type_description ordered by incident_description in ascending order.
   The search of the report_type_description should be case-insensitive.
   If n is not given, it returns all the rows.

8. Complete update_report_type().
   Using user, host, dbname, dir, and n, this function connects to the database and updates report_type_code from from_str to to_str on the report_type table.

9. Complete return_distinct_time_taken.py and return_incident_with_incident_substring.py.
   - return_distinct_time_taken.py has a text entry to accent a number to limit the number of records to display. If the number is not given, it should display all the data in a table format.
   - return_incident_with_incident_substring.py has a text entry to accept substring and the number of data to display separated by a comma. If the number is not given, it should display all the data in a table format.
   - Please see the other two files in the pages folder for the format of expected outputs.
   - For this example, user, host, and dbname are stored in user_definition.py.

10. Make sure all the code follows PEP8 standards.
    If you run pycodestyle yourcode.py, it should not return any output.