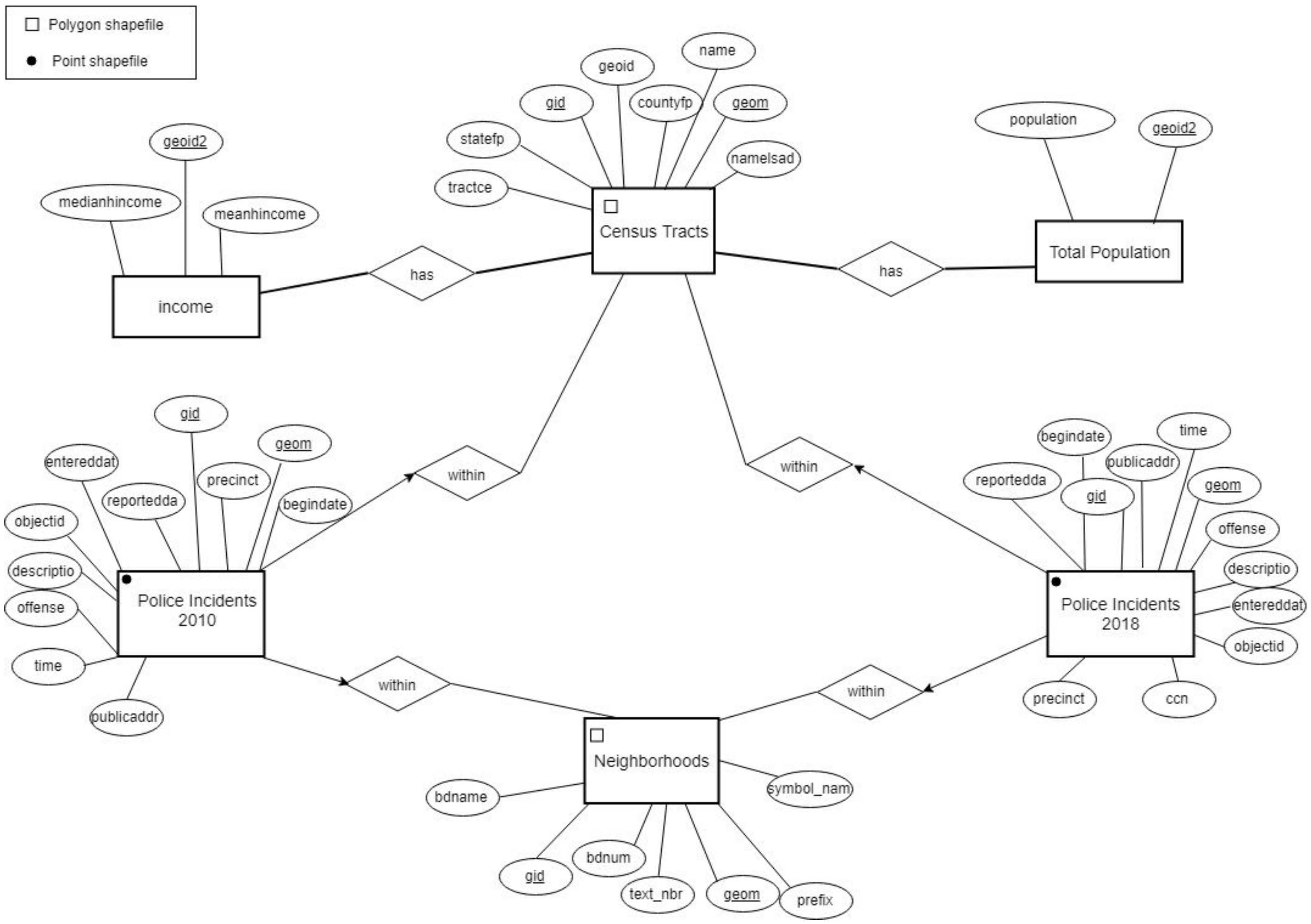
Meric Birol - Birol001
CSCI 4707

Minneapolis Crime Data Analysis

Using PostGIS, I analyzed the Police Incident 2010 and 2018 data of Minneapolis. I used 6 tables within PostGIS software called pgAdmin. (For some analysis I used Jupyter notebook along with some python libraries to connect to PostGIS database, to execute queries, and to plot data. I wrote more about this process after the Analysis section.)

(Minnesota) **Census Tracts** polygon shapefile has *gid* and *geom(geometric field)* as primary keys, and several attributes including *geoid, statefp, countyfp, tractce, namelslad,* and *name*. (Minneapolis) **Police incident 2010** and **2018** tables have *geom* and *gid* as primary keys, and several attributes including *precinct, objectid, offense (code), descriptio (offense description), time, public address,* and *begindate*. The Police incident point shapefiles are within census tracts and/or neighborhoods polygon shapefile. A (Minneapolis) **neighborhood** polygon shapefile has *gid* and *geom* as primary keys, and *bdname (neighborhood name), bdnum(neighborhood number), prefix, text_nbr*, and *symbol_nam* fields. **Income** by (Minnesota) census tracts table has *geoid2* as a primary key, and *medianhincome* (median household income) and *meanhincome* (mean household income) as attribute fields. **Population** by (Minnesota) census tracts table has *geoid2* as primary key and *population* field as an attribute. Please see the ER diagram below.



Uploading tables to the PostGIS database

I used shp2pgsql function to upload Census tracts, Neighborhoods, Police incidents 2010 and 2018 shapefiles to the PostGIS database on command prompt window. (The PostGIS automatically created indexes on the geometry field which is called “geom”.) For example:

```
shp2pgsql -s 4326 -I -W LATIN1 "C:\Users\Meric
Birol\Documents\tl_2018_27_tract\tl_2018_27_tract.shp" tracts | psql -h localhost -d MericBirol -U
postgres
```

I downloaded total population and income csv tables from ACS website. After modifying the tables, I used “create table” statements and created tables on the PostGIS database. Then, I used “\Copy” function on the SQL shell and copied those csv tables to the database. For create table statements and uploading the tables:

```
CREATE TABLE Income  (
                     GEOid2 varchar primary key,
                     MedianHIncome int,
                     MeanHIncome int);

CREATE TABLE PopulationbyTracts    (
                                    GEOid2 varchar primary key,
                                    label varchar(255),
                                    Population int);


\copy Income from 'C:\Users\Meric Birol\Documents\selectedEco\IncomebyCensus.csv' WITH CSV HEADER;
```

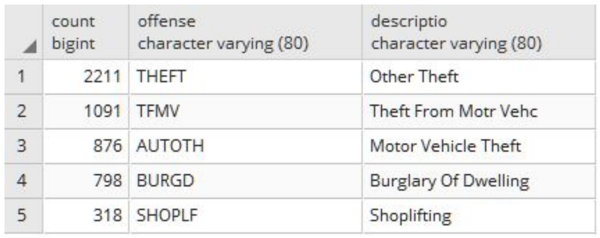
Analysis

Question 1: What are the top 5 offense types in Police incident 2018 table?

The query:

```
SELECT COUNT(offense) count, offense, descriptio
FROM pinc2018
GROUP BY offense, descriptio
ORDER BY count desc
LIMIT 5
```

The output:



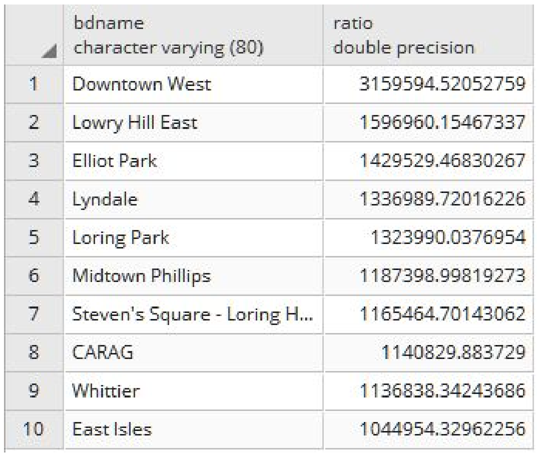
	count bigint	offense character varying (80)	descriptio character varying (80)
1	2211	THEFT	Other Theft
2	1091	TFMV	Theft From Motr Vehc
3	876	AUTOTH	Motor Vehicle Theft
4	798	BURGD	Burglary Of Dwelling
5	318	SHOPLF	Shoplifting

Question 2: Which are the top 10 neighborhoods that have the most police incidents by area?

For the query below, I used ST_Area() which calculates the area of given geom fields, and ST_Contains() which returns boolean field of the polygon geom fields whether they contains point geom fields. See the query below:

```
WITH table1 as(
SELECT COUNT(offense), bdname, ST_Area(n.geom) area
FROM neighborhoods n, pinc2018 p
WHERE ST_Contains(n.geom,p.geom)
GROUP BY n.bdname, ST_Area(n.geom))

SELECT bdname, count/area ratio
FROM table1
ORDER BY ratio desc
LIMIT 10
```

The output:



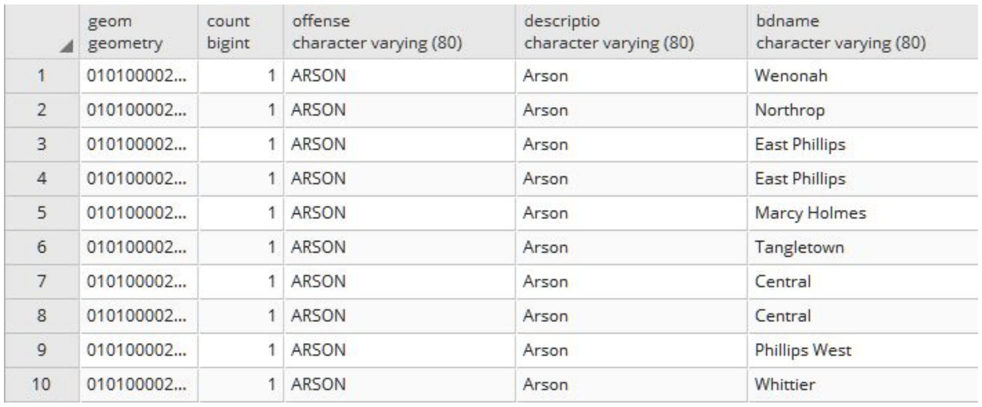
	bdname character varying (80)	ratio double precision
1	Downtown West	3159594.52052759
2	Lowry Hill East	1596960.15467337
3	Elliot Park	1429529.46830267
4	Lyndale	1336989.72016226
5	Loring Park	1323990.0376954
6	Midtown Phillips	1187398.99819273
7	Steven's Square - Loring H...	1165464.70143062
8	CARAG	1140829.883729
9	Whittier	1136838.34243686
10	East Isles	1044954.32962256

Question 3: What is the number of occurrences of each offense type for 2018 data per neighborhood?

This view below counts offense types by neighborhood:

```
CREATE VIEW incTypes as(
WITH table1 AS(
SELECT n.bdname, p.geom,p.offense,p.descriptio
FROM neighborhoods n, pinc2018 p
WHERE ST_Contains(n.geom,p.geom) )

SELECT geom,COUNT(geom),offense,descriptio,bdname
FROM table1
GROUP BY offense,geom,descriptio,bdname)
```

The first 10 rows of the view incTypes:

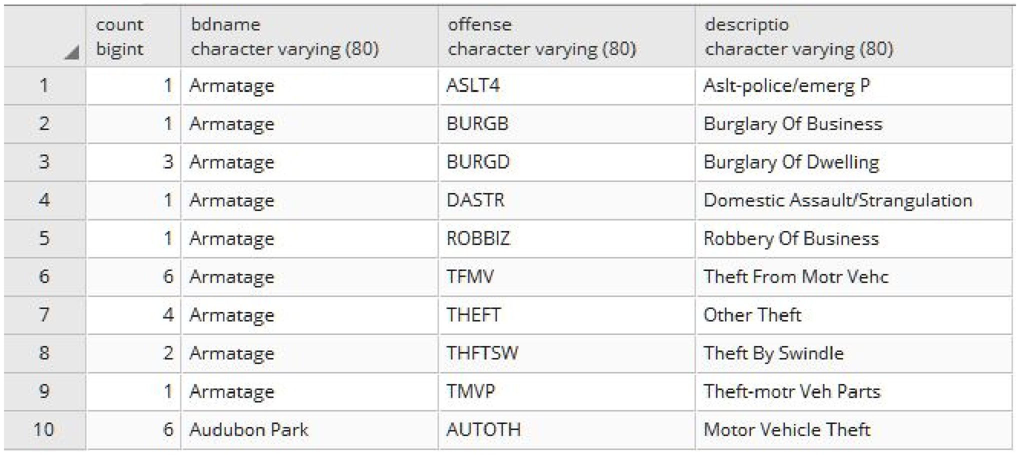


	geom	count	offense	descriptio	bdname
	geometry	bigint	character varying (80)	character varying (80)	character varying (80)
1	010100002...	1	ARSON	Arson	Wenonah
2	010100002...	1	ARSON	Arson	Northrop
3	010100002...	1	ARSON	Arson	East Phillips
4	010100002...	1	ARSON	Arson	East Phillips
5	010100002...	1	ARSON	Arson	Marcy Holmes
6	010100002...	1	ARSON	Arson	Tangletown
7	010100002...	1	ARSON	Arson	Central
8	010100002...	1	ARSON	Arson	Central
9	010100002...	1	ARSON	Arson	Phillips West
10	010100002...	1	ARSON	Arson	Whittier

In order to find number of the offense type, I run this query:

```
SELECT COUNT(offense), bdname, offense, descriptio
FROM incTypes
GROUP BY bdname,offense, descriptio
ORDER BY bdname
```

The first 10 rows of the output is:



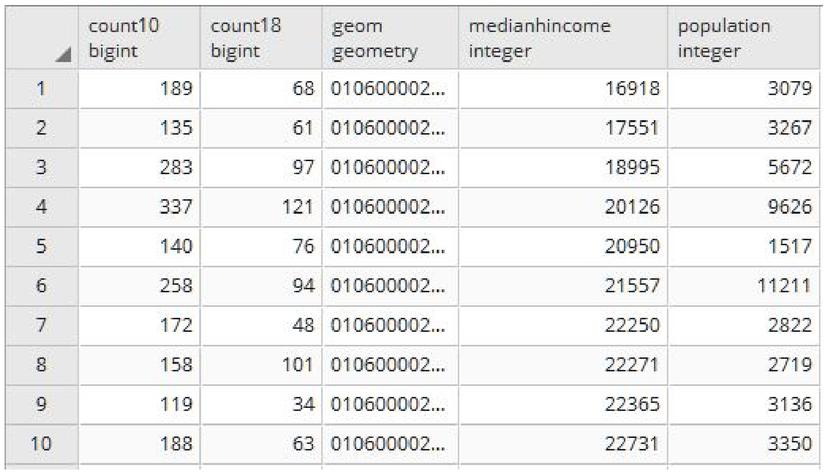
	count	bdname	offense	descriptio
	bigint	character varying (80)	character varying (80)	character varying (80)
1	1	Armatage	ASLT4	Aslt-police/emerg P
2	1	Armatage	BURGB	Burglary Of Business
3	3	Armatage	BURGD	Burglary Of Dwelling
4	1	Armatage	DASTR	Domestic Assault/Strangulation
5	1	Armatage	ROBBIZ	Robbery Of Business
6	6	Armatage	TFMV	Theft From Motr Vehc
7	4	Armatage	THEFT	Other Theft
8	2	Armatage	THFTSW	Theft By Swindle
9	1	Armatage	TMVP	Theft-motr Veh Parts
10	6	Audubon Park	AUTOTH	Motor Vehicle Theft

Question 4: What is the police incident density within Census tracts?

The query below creates views that show the number of incidents within a census tract and its population and median household income. In this query I used/joined income, tracts, population, police incidents 2010 and 2018 tables.

```
CREATE VIEW popinccount18 AS(
WITH tractIncome AS(
SELECT t.geoid,t.geom, i.medianhincome, p.population
FROM tracts t, income i,populationbytracts p
WHERE t.geoid=i.geoid2 AND t.geoid=p.geoid2)
SELECT COUNT(p18.geom) as count18,t.geom,t.medianhincome,t.population
FROM tractIncome t, pinc2018 p18
WHERE ST_Contains(t.geom,p18.geom)
GROUP BY t.geom,t.medianhincome,t.population
ORDER BY t.medianhincome);


CREATE VIEW popinccount AS(
SELECT COUNT(p10.geom) as count10,t.count18,t.geom,t.medianhincome,t.population
FROM popinccount18 t, pinc2010 p10
WHERE ST_Contains(t.geom,p10.geom)
GROUP BY t.count18,t.geom,t.medianhincome,t.population
ORDER BY t.medianhincome)
```

Here is the first 10 rows of the popinccount view:

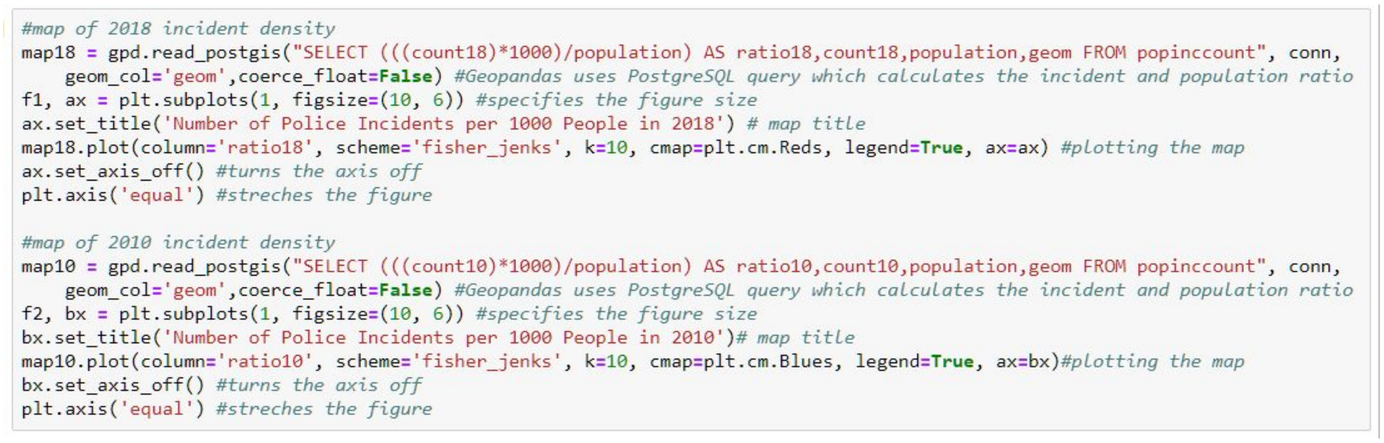


	count10 bigint	count18 bigint	geom geometry	medianhincome integer	population integer
1	189	68	010600002...	16918	3079
2	135	61	010600002...	17551	3267
3	283	97	010600002...	18995	5672
4	337	121	010600002...	20126	9626
5	140	76	010600002...	20950	1517
6	258	94	010600002...	21557	11211
7	172	48	010600002...	22250	2822
8	158	101	010600002...	22271	2719
9	119	34	010600002...	22365	3136
10	188	63	010600002...	22731	3350

In this query below shows number of police incidents per 1000 people in census tracts.

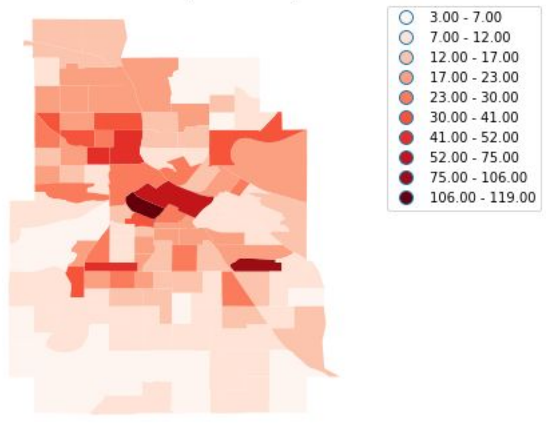
```
SELECT (((count18)*1000)/population) AS ratio18,count18,population,geom
FROM popinccount
```

I run this query in a python notebook called Jupyter, and plotted two choropleth maps for 2010 and 2018 police incident tables: (I used geopandas, matplotlib and psy2copg libraries of python for this and the next question)

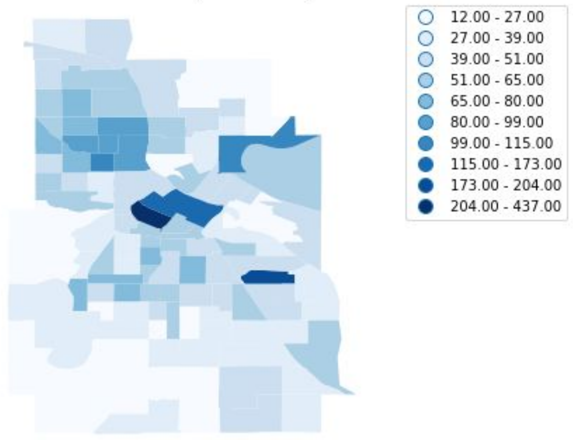


```
#map of 2018 incident density
map18 = gpd.read_postgis("SELECT (((count18)*1000)/population) AS ratio18,count18,population,geom FROM popinccount", conn,
    geom_col='geom',coerce_float=False) #Geopandas uses PostgreSQL query which calculates the incident and population ratio
f1, ax = plt.subplots(1, figsize=(10, 6)) #specifies the figure size
ax.set_title('Number of Police Incidents per 1000 People in 2018') # map title
map18.plot(column='ratio18', scheme='fisher_jenks', k=10, cmap=plt.cm.Reds, legend=True, ax=ax) #plotting the map
ax.set_axis_off() #turns the axis off
plt.axis('equal') #streches the figure

#map of 2010 incident density
map10 = gpd.read_postgis("SELECT (((count10)*1000)/population) AS ratio10,count10,population,geom FROM popinccount", conn,
    geom_col='geom',coerce_float=False) #Geopandas uses PostgreSQL query which calculates the incident and population ratio
f2, bx = plt.subplots(1, figsize=(10, 6)) #specifies the figure size
bx.set_title('Number of Police Incidents per 1000 People in 2010')# map title
map10.plot(column='ratio10', scheme='fisher_jenks', k=10, cmap=plt.cm.Blues, legend=True, ax=bx)#plotting the map
bx.set_axis_off() #turns the axis off
plt.axis('equal') #streches the figure
```

Number of Police Incidents per 1000 People in 2018

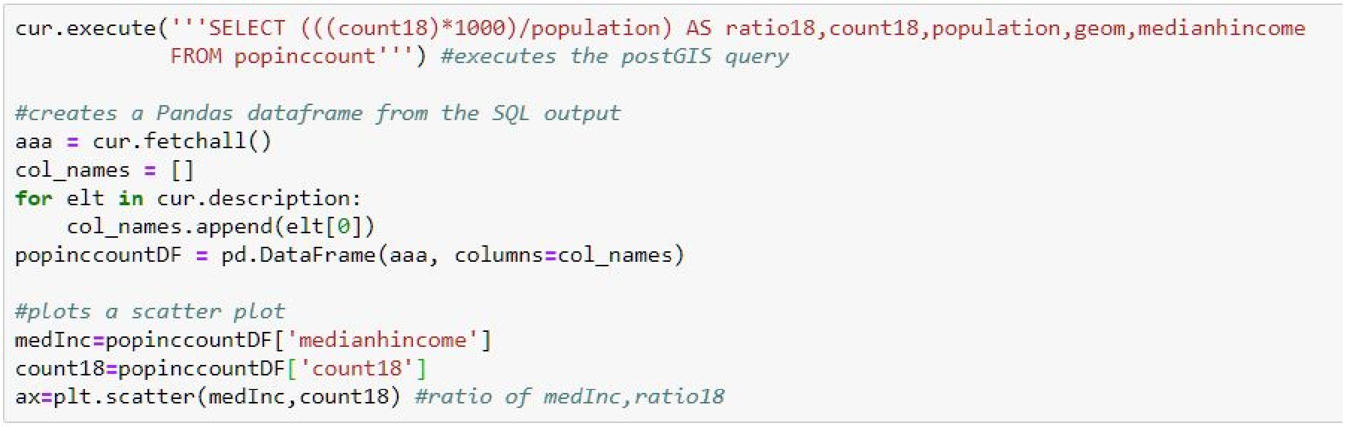


Number of Police Incidents per 1000 People in 2010

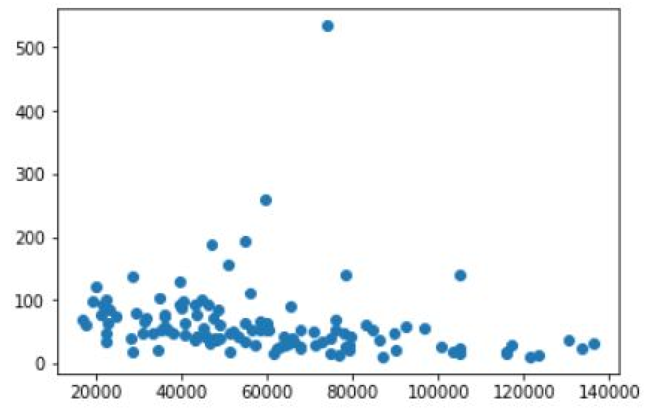


Question 5: What is the correlation between median household income and police incidents in 2018?

I analyzed this question in Jupyter. The scatter plot below shows 2018 police incidents and median household ratio. Even though the scatter plot does not show the difference much enough, the lower the median income gets, the more incident happens.

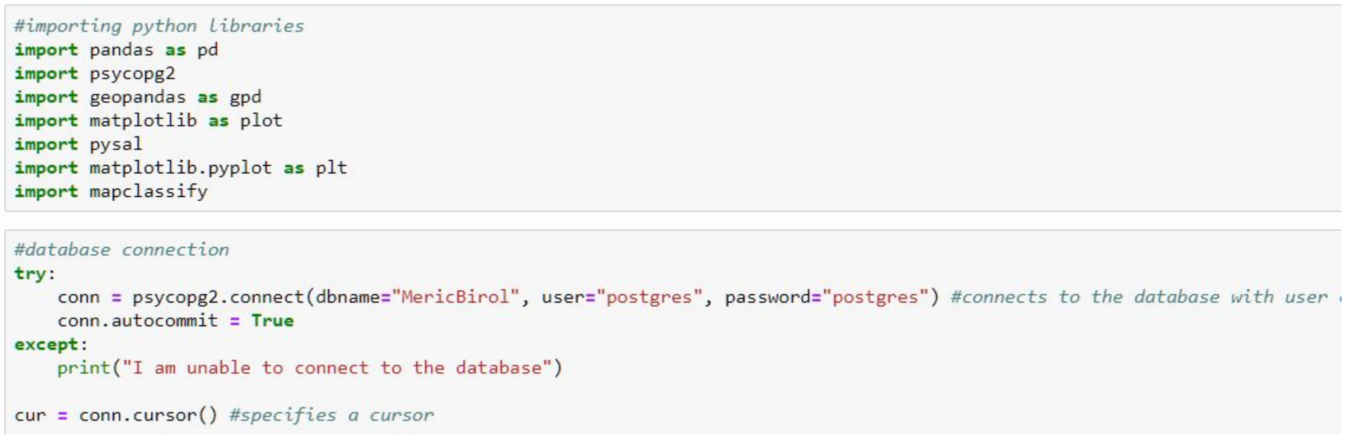


```
cur.execute('''SELECT (((count18)*1000)/population) AS ratio18,count18,population,geom,medianhincome
            FROM popinccount''') #executes the postGIS query

#creates a Pandas dataframe from the SQL output
aaa = cur.fetchall()
col_names = []
for elt in cur.description:
    col_names.append(elt[0])
popinccountDF = pd.DataFrame(aaa, columns=col_names)

#plots a scatter plot
medInc=popinccountDF['medianhincome']
count18=popinccountDF['count18']
ax=plt.scatter(medInc,count18) #ratio of medInc,ratio18
```



Extra

When I used Jupyter, I connected to the PostGIS database with psy2copg library. I attached a screenshot below:

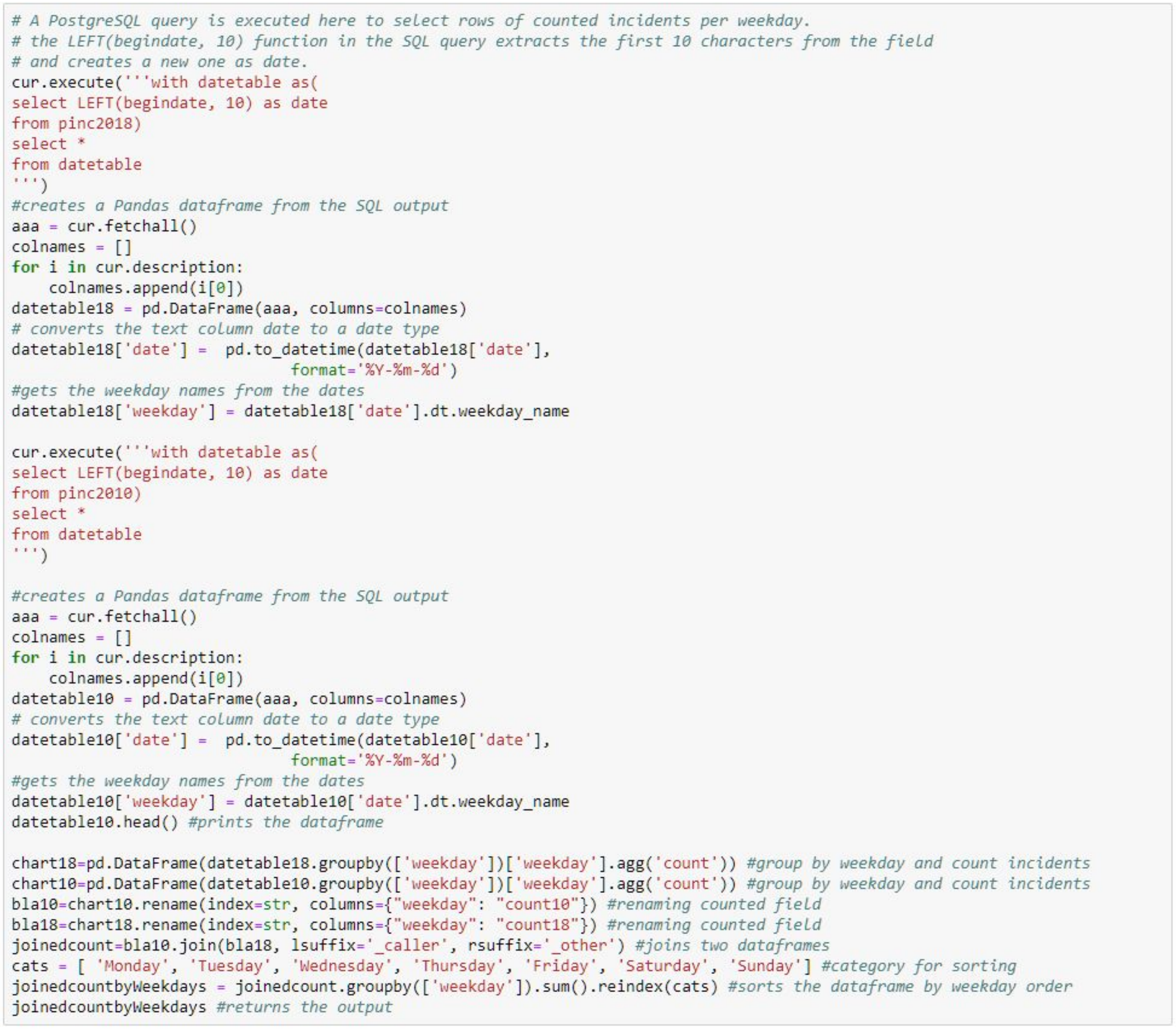


```
#importing python libraries
import pandas as pd
import psycopg2
import geopandas as gpd
import matplotlib as plot
import pysal
import matplotlib.pyplot as plt
import mapclassify
```

```
#database connection
try:
    conn = psycopg2.connect(dbname="MericBirol", user="postgres", password="postgres") #connects to the database with user
    conn.autocommit = True
except:
    print("I am unable to connect to the database")

cur = conn.cursor() #specifies a cursor
```

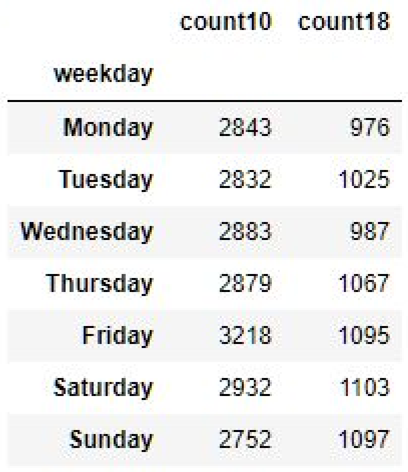
After setting the connection and cursor, I executed queries with cur.execute command. For an example please see the image in **Question 5**. After I executed the queries, I converted the output to Pandas DataFrame. Then I used matplotlib for plotting.

I also calculated how many incidents occurred per weekday names. In order to do that, I used LEFT() function of SQL. The date field type of the Police incident 2018 data was text, and it had date information followed by time information for each incident (For example: 2018-05-24T08:00:00.000Z). I needed first 10 characters of that field, and after I extracted it, I converted the output to Pandas dataframe, and used to_datetime() function of Pandas to convert text to date field. I attached a screenshot of the code below:



```python
# A PostgreSQL query is executed here to select rows of counted incidents per weekday.
# the LEFT(begindate, 10) function in the SQL query extracts the first 10 characters from the field
# and creates a new one as date.
cur.execute('''with datetable as(
select LEFT(begindate, 10) as date
from pinc2018)
select *
from datetable
''')
#creates a Pandas dataframe from the SQL output
aaa = cur.fetchall()
colnames = []
for i in cur.description:
    colnames.append(i[0])
datetable18 = pd.DataFrame(aaa, columns=colnames)
# converts the text column date to a date type
datetable18['date'] =  pd.to_datetime(datetable18['date'],
                              format='%Y-%m-%d')
#gets the weekday names from the dates
datetable18['weekday'] = datetable18['date'].dt.weekday_name

cur.execute('''with datetable as(
select LEFT(begindate, 10) as date
from pinc2010)
select *
from datetable
''')

#creates a Pandas dataframe from the SQL output
aaa = cur.fetchall()
colnames = []
for i in cur.description:
    colnames.append(i[0])
datetable10 = pd.DataFrame(aaa, columns=colnames)
# converts the text column date to a date type
datetable10['date'] =  pd.to_datetime(datetable10['date'],
                              format='%Y-%m-%d')
#gets the weekday names from the dates
datetable10['weekday'] = datetable10['date'].dt.weekday_name
datetable10.head() #prints the dataframe

chart18=pd.DataFrame(datetable18.groupby(['weekday'])['weekday'].agg('count')) #group by weekday and count incidents
chart10=pd.DataFrame(datetable10.groupby(['weekday'])['weekday'].agg('count')) #group by weekday and count incidents
bla10=chart10.rename(index=str, columns={"weekday": "count10"}) #renaming counted field
bla18=chart18.rename(index=str, columns={"weekday": "count18"}) #renaming counted field
joinedcount=bla10.join(bla18, lsuffix='_caller', rsuffix='_other') #joins two dataframes
cats = [ 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday'] #category for sorting
joinedcountbyWeekdays = joinedcount.groupby(['weekday']).sum().reindex(cats) #sorts the dataframe by weekday order
joinedcountbyWeekdays #returns the output
```

Output:



	count10	count18
weekday		
Monday	2843	976
Tuesday	2832	1025
Wednesday	2883	987
Thursday	2879	1067
Friday	3218	1095
Saturday	2932	1103
Sunday	2752	1097

Data Sources

Neighborhood - Minneapolis Open Data
Census Tracts: Census Bureau
Police Incidents 2010: Minneapolis Open Data
Police Incidents 2018: Minneapolis Open Data
Total Population: ACS table from Factfinder website (modified on Excel)
Income: ACS table from Factfinder website (modified on Excel)