Assignment: Notebook for Graded Assessment

Introduction

Using this Python notebook you will:

1. Understand three Chicago datasets
2. Load the three datasets into three tables in a SQLIte database
3. Execute SQL queries to answer assignment questions

Understand the datasets

To complete the assignment problems in this notebook you will be using three datasets that are available on the city of Chicago's Data Portal:

1. Socioeconomic Indicators in Chicago
2. Chicago Public Schools
3. Chicago Crime Data

1. Socioeconomic Indicators in Chicago

This dataset contains a selection of six socioeconomic indicators of public health significance and a “hardship index,” for each Chicago community area, for the years 2008 – 2012.

A detailed description of this dataset and the original dataset can be obtained from the Chicago Data Portal at:

https://data.cityofchicago.org/Health-Human-Services/Census-Data-Selected-socioeconomic-indicators-in-C/kn9c-c2s2

2. Chicago Public Schools

This dataset shows all school level performance data used to create CPS School Report Cards for the 2011-2012 school year. This dataset is provided by the city of Chicago's Data Portal.

A detailed description of this dataset and the original dataset can be obtained from the Chicago Data Portal at:

https://data.cityofchicago.org/Education/Chicago-Public-Schools-Progress-Report-Cards-2011-/9xs2-f89t

3. Chicago Crime Data

This dataset reflects reported incidents of crime (with the exception of murders where data exists for each victim) that occurred in the City of Chicago from 2001 to present, minus the most recent seven days.

A detailed description of this dataset and the original dataset can be obtained from the Chicago Data Portal at:

https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2

Download the datasets

This assignment requires you to have these three tables populated with a subset of the whole datasets.

In many cases the dataset to be analyzed is available as a .CSV (comma separated values) file, perhaps on the internet.

Use the links below to read the data files using the Pandas library.

- Chicago Census Data

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DB0201EN-SkillsNetwork/labs/FinalModule_Coursera_V5/data/ChicagoCensusData.csv?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_te
SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDB0201ENSkillsNetwork20127838-2021-01-01

- Chicago Public Schools

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DB0201EN-SkillsNetwork/labs/FinalModule_Coursera_V5/data/ChicagoPublicSchools.csv?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_te
SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDB0201ENSkillsNetwork20127838-2021-01-01

- Chicago Crime Data

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DB0201EN-SkillsNetwork/labs/FinalModule_Coursera_V5/data/ChicagoCrimeData.csv?

utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_te SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDB0201ENSkillsNetwork20127838-2021-01-01

NOTE: Ensure you use the datasets available on the links above instead of directly from the Chicago Data Portal. The versions linked here are subsets of the original datasets and have some of the column names modified to be more database friendly which will make it easier to complete this assignment.

Store the datasets in database tables

To analyze the data using SQL, it first needs to be loaded into SQLite DB. We will create three tables in as under:

1. **CENSUS_DATA**
2. **CHICAGO_PUBLIC_SCHOOLS**
3. **CHICAGO_CRIME_DATA**

Load the `pandas` and `sqlite3` libraries and establish a connection to `FinalDB.db`

```
In [1]: !pip install pandas
```

```
Requirement already satisfied: pandas in /home/jupyterlab/conda/envs/pytho
n/lib/python3.7/site-packages (1.3.5)
Requirement already satisfied: python-dateutil>=2.7.3 in /home/jupyterlab/
conda/envs/python/lib/python3.7/site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /home/jupyterlab/conda/env
s/python/lib/python3.7/site-packages (from pandas) (2023.3)
Requirement already satisfied: numpy>=1.17.3 in /home/jupyterlab/conda/env
s/python/lib/python3.7/site-packages (from pandas) (1.21.6)
Requirement already satisfied: six>=1.5 in /home/jupyterlab/conda/envs/pyt
hon/lib/python3.7/site-packages (from python-dateutil>=2.7.3->pandas) (1.1
6.0)
```

Load the SQL magic module

```
In [14]: %load_ext sql
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
```

Use `Pandas` to load the data available in the links above to dataframes. Use these dataframes to load data on to the database `FinalDB.db` as required tables.

```
In [ ]:
```

Establish a connection between SQL magic module and the database `FinalDB.db`

```
In [15]: %sql sqlite:///FinalDB
```

```
Out[15]: 'Connected: @FinalDB'
```

In [39]:
```
import sqlite3
import pandas as pd
import csv
conn = sqlite3.connect('FinalDB.db')
cur = conn.cursor()
```

You can now proceed to the the following questions. Please note that a graded assignment will follow this lab and there will be a question on each of the problems stated below. It can be from the answer you received or the code you write for this problem. Therefore, please keep a note of both your codes as well as the response you generate.

Problems

Now write and execute SQL queries to solve assignment problems

Problem 1

Find the total number of crimes recorded in the CRIME table.

In [40]:
```
import sqlite3
import pandas as pd

# Load your data into Pandas DataFrames (replace these paths with your ac
census_data = pd.read_csv('./ChicagoCensusData.csv')
public_schools_data = pd.read_csv('./ChicagoPublicSchools.csv')
crime_data = pd.read_csv('./ChicagoCrimeData.csv')

# Establish a connection to the SQLite database (replace 'FinalDB.db' wit
conn = sqlite3.connect('FinalDB.db')

# Write DataFrames to SQLite tables
census_data.to_sql('CENSUS_DATA', conn, index=False, if_exists='replace')
public_schools_data.to_sql('CHICAGO_PUBLIC_SCHOOLS', conn, index=False, i
crime_data.to_sql('CHICAGO_CRIME_DATA', conn, index=False, if_exists='rep

# Close the connection
conn.close()
```

In [37]:
```
type(data)
```

Out[37]:
```
pandas.core.frame.DataFrame
```

In [42]:
```
%sql sqlite:///FinalDB
```

Out[42]:
```
'Connected: @FinalDB'
```

In [46]:
```
import sqlite3
import pandas as pd

# Connect to the SQLite database
conn = sqlite3.connect('FinalDB.db')

# Query the first few rows of a table (replace TABLE_NAME with the actual
query = "SELECT * FROM CENSUS_DATA LIMIT 5;"
result = pd.read_sql_query(query, conn)

# Display the result
print(result)

# Close the connection
conn.close()
```

```
   COMMUNITY_AREA_NUMBER COMMUNITY_AREA_NAME  PERCENT_OF_HOUSING_CROWDED
\
0                    1.0         Rogers Park                         7.7
1                    2.0          West Ridge                         7.8
2                    3.0              Uptown                         3.8
3                    4.0      Lincoln Square                         3.4
4                    5.0        North Center                         0.3

   PERCENT_HOUSEHOLDS_BELOW_POVERTY  PERCENT_AGED_16__UNEMPLOYED  \
0                              23.6                          8.7
1                              17.2                          8.8
2                              24.0                          8.9
3                              10.9                          8.2
4                               7.5                          5.2

   PERCENT_AGED_25__WITHOUT_HIGH_SCHOOL_DIPLOMA  \
0                                          18.2
1                                          20.8
2                                          11.8
3                                          13.4
4                                           4.5

   PERCENT_AGED_UNDER_18_OR_OVER_64  PER_CAPITA_INCOME  HARDSHIP_INDEX
0                              27.5              23939            39.0
1                              38.5              23040            46.0
2                              22.2              35787            20.0
3                              25.5              37524            17.0
4                              26.2              57123             6.0
```

In [78]:

```
%load_ext sql

# Connect to the SQLite database
%sql sqlite:///FinalDB.db

# SQL query to select the first few rows from the CENSUS_DATA table
result = %sql SELECT * FROM CENSUS_DATA LIMIT 5;

# Display the result
result
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
```

Out[78]:

COMMUNITY_AREA_NUMBER	COMMUNITY_AREA_NAME	PERCENT_OF_HOUSING_CR
1.0	Rogers Park	
2.0	West Ridge	
3.0	Uptown	
4.0	Lincoln Square	
5.0	North Center	

In [48]:

```
import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('FinalDB.db')

# Specify the table name you want to read
table_name = 'CENSUS_DATA'  # Replace with the actual table name

# Query and fetch all rows from the table
query = f"SELECT * FROM {table_name};"
cursor = conn.cursor()
cursor.execute(query)

# Fetch all rows as a list of tuples
rows = cursor.fetchall()

# Display the result
for row in rows:
    print(row)

# Close the connection
conn.close()
```

```
(1.0, 'Rogers Park', 7.7, 23.6, 8.7, 18.2, 27.5, 23939, 39.0)
(2.0, 'West Ridge', 7.8, 17.2, 8.8, 20.8, 38.5, 23040, 46.0)
(3.0, 'Uptown', 3.8, 24.0, 8.9, 11.8, 22.2, 35787, 20.0)
(4.0, 'Lincoln Square', 3.4, 10.9, 8.2, 13.4, 25.5, 37524, 17.0)
(5.0, 'North Center', 0.3, 7.5, 5.2, 4.5, 26.2, 57123, 6.0)
(6.0, 'Lake View', 1.1, 11.4, 4.7, 2.6, 17.0, 60058, 5.0)
(7.0, 'Lincoln Park', 0.8, 12.3, 5.1, 3.6, 21.5, 71551, 2.0)
(8.0, 'Near North Side', 1.9, 12.9, 7.0, 2.5, 22.6, 88669, 1.0)
(9.0, 'Edison Park', 1.1, 3.3, 6.5, 7.4, 35.3, 40959, 8.0)
(10.0, 'Norwood Park', 2.0, 5.4, 9.0, 11.5, 39.5, 32875, 21.0)
(11.0, 'Jefferson Park', 2.7, 8.6, 12.4, 13.4, 35.5, 27751, 25.0)
(12.0, 'Forest Glen', 1.1, 7.5, 6.8, 4.9, 40.5, 44164, 11.0)
(13.0, 'North Park', 3.9, 13.2, 9.9, 14.4, 39.0, 26576, 33.0)
(14.0, 'Albany Park', 11.3, 19.2, 10.0, 32.9, 32.0, 21323, 53.0)
(15.0, 'Portage Park', 4.1, 11.6, 12.6, 19.3, 34.0, 24336, 35.0)
(16.0, 'Irving Park', 6.3, 13.1, 10.0, 22.4, 31.6, 27249, 34.0)
(17.0, 'Dunning', 5.2, 10.6, 10.0, 16.2, 33.6, 26282, 28.0)
(18.0, 'Montclaire', 8.1, 15.3, 13.8, 23.5, 38.6, 22014, 50.0)
(19.0, 'Belmont Cragin', 10.8, 18.7, 14.6, 37.3, 37.3, 15461, 70.0)
(20.0, 'Hermosa', 6.9, 20.5, 13.1, 41.6, 36.4, 15089, 71.0)
(21.0, 'Avondale', 6.0, 15.3, 9.2, 24.7, 31.0, 20039, 42.0)
(22.0, 'Logan Square', 3.2, 16.8, 8.2, 14.8, 26.2, 31908, 23.0)
(23.0, 'Humboldt park', 14.8, 33.9, 17.3, 35.4, 38.0, 13781, 85.0)
(24.0, 'West Town', 2.3, 14.7, 6.6, 12.9, 21.7, 43198, 10.0)
(25.0, 'Austin', 6.3, 28.6, 22.6, 24.4, 37.9, 15957, 73.0)
(26.0, 'West Garfield Park', 9.4, 41.7, 25.8, 24.5, 43.6, 10934, 92.0)
(27.0, 'East Garfield Park', 8.2, 42.4, 19.6, 21.3, 43.2, 12961, 83.0)
(28.0, 'Near West Side', 3.8, 20.6, 10.7, 9.6, 22.2, 44689, 15.0)
(29.0, 'North Lawndale', 7.4, 43.1, 21.2, 27.6, 42.7, 12034, 87.0)
(30.0, 'South Lawndale', 15.2, 30.7, 15.8, 54.8, 33.8, 10402, 96.0)
(31.0, 'Lower West Side', 9.6, 25.8, 15.8, 40.7, 32.6, 16444, 76.0)
(32.0, 'Loop', 1.5, 14.7, 5.7, 3.1, 13.5, 65526, 3.0)
(33.0, 'Near South Side', 1.3, 13.8, 4.9, 7.4, 21.8, 59077, 7.0)
(34.0, 'Armour Square', 5.7, 40.1, 16.7, 34.5, 38.3, 16148, 82.0)
(35.0, 'Douglas', 1.8, 29.6, 18.2, 14.3, 30.7, 23791, 47.0)
(36.0, 'Oakland', 1.3, 39.7, 28.7, 18.4, 40.4, 19252, 78.0)
(37.0, 'Fuller Park', 3.2, 51.2, 33.9, 26.6, 44.9, 10432, 97.0)
(38.0, 'Grand Boulevard', 3.3, 29.3, 24.3, 15.9, 39.5, 23472, 57.0)
(39.0, 'Kenwood', 2.4, 21.7, 15.7, 11.3, 35.4, 35911, 26.0)
(40.0, 'Washington Park', 5.6, 42.1, 28.6, 25.4, 42.8, 13785, 88.0)
(41.0, 'Hyde Park', 1.5, 18.4, 8.4, 4.3, 26.2, 39056, 14.0)
(42.0, 'Woodlawn', 2.9, 30.7, 23.4, 16.5, 36.1, 18672, 58.0)
(43.0, 'South Shore', 2.8, 31.1, 20.0, 14.0, 35.7, 19398, 55.0)
(44.0, 'Chatham', 3.3, 27.8, 24.0, 14.5, 40.3, 18881, 60.0)
(45.0, 'Avalon Park', 1.4, 17.2, 21.1, 10.6, 39.3, 24454, 41.0)
(46.0, 'South Chicago', 4.7, 29.8, 19.7, 26.6, 41.1, 16579, 75.0)
(47.0, 'Burnside', 6.8, 33.0, 18.6, 19.3, 42.7, 12515, 79.0)
(48.0, 'Calumet Heights', 2.1, 11.5, 20.0, 11.0, 44.0, 28887, 38.0)
(49.0, 'Roseland', 2.5, 19.8, 20.3, 16.9, 41.2, 17949, 52.0)
(50.0, 'Pullman', 1.5, 21.6, 22.8, 13.1, 38.6, 20588, 51.0)
(51.0, 'South Deering', 4.0, 29.2, 16.3, 21.0, 39.5, 14685, 65.0)
(52.0, 'East Side', 6.8, 19.2, 12.1, 31.9, 42.8, 17104, 64.0)
(53.0, 'West Pullman', 3.3, 25.9, 19.4, 20.5, 42.1, 16563, 62.0)
(54.0, 'Riverdale', 5.8, 56.5, 34.6, 27.5, 51.5, 8201, 98.0)
(55.0, 'Hegewisch', 3.3, 17.1, 9.6, 19.2, 42.9, 22677, 44.0)
(56.0, 'Garfield Ridge', 2.6, 8.8, 11.3, 19.3, 38.1, 26353, 32.0)
(57.0, 'Archer Heights', 8.5, 14.1, 16.5, 35.9, 39.2, 16134, 67.0)
(58.0, 'Brighton Park', 14.4, 23.6, 13.9, 45.1, 39.3, 13089, 84.0)
(59.0, 'McKinley Park', 7.2, 18.7, 13.4, 32.9, 35.6, 16954, 61.0)
(60.0, 'Bridgeport', 4.5, 18.9, 13.7, 22.2, 31.3, 22694, 43.0)
```

```
(61.0, 'New City', 11.9, 29.0, 23.0, 41.5, 38.9, 12765, 91.0)
(62.0, 'West Elsdon', 11.1, 15.6, 16.7, 37.0, 37.7, 15754, 69.0)
(63.0, 'Gage Park', 15.8, 23.4, 18.2, 51.5, 38.8, 12171, 93.0)
(64.0, 'Clearing', 2.7, 8.9, 9.5, 18.8, 37.6, 25113, 29.0)
(65.0, 'West Lawn', 5.8, 14.9, 9.6, 33.6, 39.6, 16907, 56.0)
(66.0, 'Chicago Lawn', 7.6, 27.9, 17.1, 31.2, 40.6, 13231, 80.0)
(67.0, 'West Englewood', 4.8, 34.4, 35.9, 26.3, 40.7, 11317, 89.0)
(68.0, 'Englewood', 3.8, 46.6, 28.0, 28.5, 42.5, 11888, 94.0)
(69.0, 'Greater Grand Crossing', 3.6, 29.6, 23.0, 16.5, 41.0, 17285, 66.0)
(70.0, 'Ashburn', 4.0, 10.4, 11.7, 17.7, 36.9, 23482, 37.0)
(71.0, 'Auburn Gresham', 4.0, 27.6, 28.3, 18.5, 41.9, 15528, 74.0)
(72.0, 'Beverly', 0.9, 5.1, 8.0, 3.7, 40.5, 39523, 12.0)
(73.0, 'Washington Height', 1.1, 16.9, 20.8, 13.7, 42.6, 19713, 48.0)
(74.0, 'Mount Greenwood', 1.0, 3.4, 8.7, 4.3, 36.8, 34381, 16.0)
(75.0, 'Morgan Park', 0.8, 13.2, 15.0, 10.8, 40.3, 27149, 30.0)
(76.0, "O'Hare", 3.6, 15.4, 7.1, 10.9, 30.3, 25828, 24.0)
(77.0, 'Edgewater', 4.1, 18.2, 9.2, 9.7, 23.8, 33385, 19.0)
(None, 'CHICAGO', 4.7, 19.7, 12.9, 19.5, 33.5, 28202, None)
```

In [49]:

```
import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('FinalDB.db')

# Specify the table name for which you want to count the rows
table_name = 'CHICAGO_CRIME_DATA'  # Replace with the actual table name

# Query to count the number of rows in the table
query = f"SELECT COUNT(*) FROM {table_name};"
cursor = conn.cursor()
cursor.execute(query)

# Fetch the result
count = cursor.fetchone()[0]

# Display the result
print(f"Number of rows in '{table_name}': {count}")

# Close the connection
conn.close()
```

```
Number of rows in 'CHICAGO_CRIME_DATA': 533
```

In [80]:

```
%load_ext sql

# Connect to the SQLite database
%sql sqlite:///FinalDB.db

# Specify the table name for which you want to count the rows
table_name = 'CHICAGO_CRIME_DATA'  # Replace with the actual table name

# Query to count the number of rows in the table
result = %sql SELECT COUNT(*) FROM $table_name;

# Display the result
count = result
print(f"Number of rows in '{table_name}': {count}")
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
Number of rows in 'CHICAGO_CRIME_DATA': +----------+
| COUNT(*) |
+----------+
|   533    |
+----------+
```

In [81]:

```
%load_ext sql

# Connect to the SQLite database
%sql sqlite:///FinalDB.db
%sql SELECT COUNT(*) FROM CHICAGO_CRIME_DATA;
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
```

Out[81]:

COUNT(*)
533

Problem 2

List community area names and numbers with per capita income less than 11000.

In [93]:

```
%load_ext sql

# Connect to the SQLite database
%sql sqlite:///FinalDB.db

#%sql  SELECT Community_AREA FROM CENSUS_DATA;
%sql SELECT Community_AREA_NAME, COMMUNITY_AREA_NUMBER FROM CENSUS_DATA W
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
```

Out[93]:

COMMUNITY_AREA_NAME	COMMUNITY_AREA_NUMBER
West Garfield Park	26.0
South Lawndale	30.0
Fuller Park	37.0
Riverdale	54.0

In [ ]:

In [56]:

```
import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('FinalDB.db')
```

```
# Execute the SQL query
query = """
SELECT Community_AREA_NAME,COMMUNITY_AREA_NUMBER
FROM CENSUS_DATA
WHERE Per_Capita_Income < 11000;
"""
cursor = conn.cursor()
cursor.execute(query)

# Fetch and print the result
result = cursor.fetchall()
for row in result:
    print(row)

# Close the connection
conn.close()
```

```
('West Garfield Park', 26.0)
('South Lawndale', 30.0)
('Fuller Park', 37.0)
('Riverdale', 54.0)
```

In [ ]:

Problem 3

List all case numbers for crimes involving minors?(children are not considered minors for the purposes of crime analysis)

In [72]:

```
import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('FinalDB.db')  # Replace with your actual database

# Create a cursor
cursor = conn.cursor()

# SQL query to list case numbers for crimes involving minors
query = """
SELECT CASE_NUMBER
FROM CHICAGO_CRIME_DATA
WHERE DESCRIPTION LIKE ?;
"""

# Execute the query
description_keyword = '%MINOR%'
cursor.execute(query, (description_keyword,))

# Fetch and print the results
result = cursor.fetchall()
for row in result:
    print(row[0])

# Close the connection
conn.close()
```

```
HL266884
HK238408
```

```
In [ ]:
```

```
In [97]: %sql SELECT CASE_NUMBER FROM CHICAGO_CRIME_DATA WHERE DESCRIPTION LIKE '%
```

```
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
```

Out[97]:

CASE_NUMBER
HL266884
HK238408

Problem 4

List all kidnapping crimes involving a child?

```
In [76]: import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('FinalDB.db')  # Replace with your actual database

# Create a cursor
cursor = conn.cursor()

# SQL query to list all kidnapping crimes involving a child
query = """
SELECT CASE_NUMBER, PRIMARY_TYPE, DESCRIPTION
FROM CHICAGO_CRIME_DATA
WHERE PRIMARY_TYPE = 'KIDNAPPING' AND DESCRIPTION LIKE '%CHILD%';
"""

# Execute the query
cursor.execute(query)

# Fetch and print the results
result = cursor.fetchall()
for row in result:
    print(row)

# Close the connection
conn.close()
```

```
('HN144152', 'KIDNAPPING', 'CHILD ABDUCTION/STRANGER')
```

```
In [99]: %sql SELECT CASE_NUMBER, PRIMARY_TYPE, DESCRIPTION FROM CHICAGO_CRIME_DAT
```

```
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
```

Out[99]:

CASE_NUMBER	PRIMARY_TYPE	DESCRIPTION
HN144152	KIDNAPPING	CHILD ABDUCTION/STRANGER

Problem 5

List the kind of crimes that were recorded at schools. (No repetitions)

In [100…

```
%sql SELECT DISTINCT PRIMARY_TYPE FROM CHICAGO_CRIME_DATA WHERE LOCATION_
```

```
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
```

Out[100]:

PRIMARY_TYPE
BATTERY
CRIMINAL DAMAGE
NARCOTICS
ASSAULT
CRIMINAL TRESPASS
PUBLIC PEACE VIOLATION

Problem 6

List the type of schools along with the average safety score for each type.

In [101…

```
%sql SELECT "Elementary, Middle, or High School" as School_Type, AVG(SAFE
```

```
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
```

Out[101]:

School_Type	Average_Safety_Score
ES	49.52038369304557
HS	49.62352941176471
MS	48.0

Problem 7

List 5 community areas with highest % of households below poverty line

In [102…

```
%sql SELECT Community_Area_Name, PERCENT_HOUSEHOLDS_BELOW_POVERTY FROM CE
```

```
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
```

Out[102]:

COMMUNITY_AREA_NAME	PERCENT_HOUSEHOLDS_BELOW_POVERTY
Riverdale	56.5
Fuller Park	51.2
Englewood	46.6
North Lawndale	43.1
East Garfield Park	42.4

Problem 8

Which community area is most crime prone? Display the coumminty area number only.

In [103…

```
%sql SELECT COMMUNITY_AREA_NUMBER FROM CHICAGO_CRIME_DATA GROUP BY COMMUN
```

```
  sqlite:///FinalDB
* sqlite:///FinalDB.db
Done.
```

Out[103]:

COMMUNITY_AREA_NUMBER
25.0

Double-click **here** for a hint

Problem 9

Use a sub-query to find the name of the community area with highest hardship index

In [104…

```
%sql SELECT COMMUNITY_AREA_NAME FROM CENSUS_DATA WHERE HARDSHIP_INDEX = (
```

```
  sqlite:///FinalDB
* sqlite:///FinalDB.db
Done.
```

Out[104]:

COMMUNITY_AREA_NAME
Riverdale

In [112…

```
%load_ext sql

# Connect to the SQLite database
%sql sqlite:///FinalDB.db
%sql SELECT  *FROM CHICAGO_CRIME_DATA LIMIT 2;
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
  sqlite:///FinalDB
* sqlite:///FinalDB.db
Done.
```

Out[112]:

ID	CASE_NUMBER	DATE	BLOCK	IUCR	PRIMARY_TYPE	DESCRIPTION
3512276	HK587712	2004-08-28	047XX S KEDZIE AVE	890	THEFT	FROM BUILDING
3406613	HK456306	2004-06-26	009XX N CENTRAL PARK AVE	820	THEFT	$500 AND UNDER

Problem 10

Use a sub-query to determine the Community Area Name with most number of crimes?

In [113…

```
%sql SELECT BLOCK FROM CHICAGO_CRIME_DATA WHERE COMMUNITY_AREA_NUMBER = (
   sqlite:///FinalDB
 * sqlite:///FinalDB.db
Done.
```

Out[113]:

BLOCK
055XX W GLADYS AVE
058XX W ARTHINGTON ST
017XX N AUSTIN AVE
0000X N LATROBE AVE
017XX N LUNA AVE
008XX N CICERO AVE
048XX W NORTH AVE
017XX N NATCHEZ AVE
056XX W ROOSEVELT RD
051XX W MADISON ST
047XX W SUPERIOR ST
014XX N MENARD AVE
048XX W HURON ST
050XX W DIVISION ST
005XX N WALLER AVE
010XX N WALLER AVE
052XX W CONGRESS PKWY
051XX W KINZIE ST
053XX W CONGRESS PKWY
005XX N LAVERGNE AVE
004XX N LARAMIE AVE
047XX W FULTON ST
004XX N PINE AVE
051XX W MADISON ST
009XX N LOREL AVE
049XX W MAYPOLE AVE
012XX N LOCKWOOD AVE
016XX N MENARD AVE
0000X N WALLER AVE
011XX N LOREL AVE
050XX W NORTH AVE
050XX W JACKSON BLVD
049XX W WASHINGTON BLVD
017XX N MASON AVE

BLOCK
068XX W NORTH AVE
051XX W WASHINGTON BLVD
054XX W POTOMAC AVE
0000X S MAYFIELD AVE
009XX N MASSASOIT AVE
0000X N KENTON AVE
002XX N LARAMIE AVE
050XX W VAN BUREN ST
059XX W CHICAGO AVE

Author(s)

Hima Vasudevan

Rav Ahuja

Ramesh Sannreddy

Contribtuor(s)

Malika Singla

Abhishek Gagneja

Change log

Date	Version	Changed by	Change Description
2023-10-18	2.6	Abhishek Gagneja	Modified instruction set
2022-03-04	2.5	Lakshmi Holla	Changed markdown.
2021-05-19	2.4	Lakshmi Holla	Updated the question
2021-04-30	2.3	Malika Singla	Updated the libraries
2021-01-15	2.2	Rav Ahuja	Removed problem 11 and fixed changelog
2020-11-25	2.1	Ramesh Sannareddy	Updated the problem statements, and datasets
2020-09-05	2.0	Malika Singla	Moved lab to course repo in GitLab
2018-07-18	1.0	Rav Ahuja	Several updates including loading

Date	Version	Changed by	Change Description
			instructions
2018-05-04	0.1	Hima Vasudevan	Created initial version