databricksProject#4: Notebook Exploratory Data Analysis using Databricks

(https://databricks.com)

Project#4: Exploratory Data Analysis using Databricks

(Total Points: 10)

Instruction:

Perform exploratory data analysis (EDA) to gain insights using Community Edition of Databricks.

Use the Notebook "Project#3: Exploratory Data Analysis using Databricks" DBC file for the assigment

“Databricks Overview Labs CS 5165 | 6065 v1_1.dbc” file has been provided for review. It is in Canvas.

Homework Tasks:

1. Calculate the Total count of Crimes for each of the 6 United States cities listed below using 2016 crime data in dbfs:/mnt/training/crime-data-2016

2. Provide Total count of different Types of Crimes for each of 6 United States cities listed below using crime-data-2016

3. Calculate the total Robbery count for each of the 3 United States cities listed below using crime-data-2016

4. Find the months with the Highest and Lowest Robbery counts for each of the 3 United States cities listed below using crime-data-2016

5. Combine all three cities robberies-per-month views into one and Find the month with the Highest and Lowest combined Robbery counts using crime-data-2016

6. Plot the robberies per month for each of the three cities, producing one Graph using the contents of combinedRobberiesByMonthDF

7. Find the "per capita robbery rates" using ther Hint below, and plot graph as above for the per capita robbery rates per month for each of the three cities, producing one Graph using the contents of robberyRatesByCityDF

8. Find the monthly HOMICIDE counts for each of the 2 United States cities listed below using crime-data-2016, and Combine both cities HOMICIDE-per-month views into one and Find the month with the Highest and Lowest combined HOMICIDE-per-month counts. See Question 6.

9. Find the "per capita HOMICIDE rates" using the Hint in Qn 7, and plot graph as above for the per capita HOMICIDE rates per month for each of the two cities, producing one Graph using the contents of HOMICIDERatesByCityDF

10. A stretch goal to address a Data Science question on predicting crimes for a city as a time series model. How would you predict future values for monthly Robbery count rate for Log Angeles using the historical values from crime-data-2016. Data Science is a multi-year discipline. I am not expecting anything fancy. There is no wrong answer. I do expect students to explore and give their best shot.

Submission: 1. Export the completed "Project#4: Exploratory Data Analysis using Databricks.dbc" as DBC Archive file with all the information in Blackboard 2. Your spark code/command and results included

Exploratory Data Analysis using Databricks

Perform exploratory data analysis (EDA) to gain insights from a data lake.

Instructions

In `dbfs:/mnt/training/crime-data-2016`, there are a number of Parquet files containing 2016 crime data from seven United States cities:

- New York

- Los Angeles
- Chicago
- Philadelphia
- Dallas
- Boston

The data is cleaned up a little, but has not been normalized. Each city reports crime data slightly differently, so examine the data for each city to determine how to query it properly.

Your job is to use some of this data to gain insights about certain kinds of crimes.

Getting Started

Run the following cell to configure our "classroom."

```
%run ./Includes/Classroom-Setup
```

Initialized classroom variables & functions...

Datasets are already mounted to **/mnt/training** from **s3a://databricks-corp-training/common**

Imported Test Library...

Created user-specific database

Using the database **changjb_mail_uc_edu_db**.

All done!

```
print("username: " + username)
print("userhome: " + userhome)
```

```
username: changjb@mail.uc.edu
userhome: dbfs:/user/changjb@mail.uc.edu
```

Question 1: Calculate the Total count of Crimes for each of the 6 United States cities listed below using 2016 crime data in dbfs:/mnt/training/crime-data-2016:

- New York
- Los Angeles
- Chicago
- Philadelphia
- Dallas
- Boston

Hint:

Start by creating DataFrames for Los Angeles, Philadelphia, and Dallas data.

Use `spark.read.parquet` to create named DataFrames for the files you choose.

To read in the parquet file, use
`crimeDataNewYorkDF = spark.read.parquet("/mnt/training/crime-data-2016/Crime-Data-New-York-2016.parquet")`

Use the following view names:

City	DataFrame Name	Path to DBFS file
Los Angeles	`crimeDataLosAngelesDF`	`dbfs:/mnt/training/crime-data-2016/Crime-Data-Los-Angeles-2016.parqu`

City	DataFrame Name	Path to DBFS file
Philadelphia	`crimeDataPhiladelphiaDF`	`dbfs:/mnt/training/crime-data-2016/Crime-Data-Philadelphia-2016.parc`

Hint: Los Angeles

```
crimeDataNewYorkDF = spark.read.parquet("dbfs:/mnt/training/crime-data-2016/Crime-Data-New-York-2016.parquet")
print('New York:', crimeDataNewYorkDF.count())

crimeDataLosAngelesDF = spark.read.parquet("dbfs:/mnt/training/crime-data-2016/Crime-Data-Los-Angeles-2016.parquet")
print('Los Angeles:', crimeDataLosAngelesDF.count())

crimeDataChicagoDF = spark.read.parquet("dbfs:/mnt/training/crime-data-2016/Crime-Data-Chicago-2016.parquet")
print('Chicago:', crimeDataChicagoDF.count())

crimeDataPhiladelphiaDF = spark.read.parquet("dbfs:/mnt/training/crime-data-2016/Crime-Data-Philadelphia-
2016.parquet")
print('Philadelphia:', crimeDataPhiladelphiaDF.count())

crimeDataDallasDF = spark.read.parquet("dbfs:/mnt/training/crime-data-2016/Crime-Data-Dallas-2016.parquet")
print('Dallas:', crimeDataDallasDF.count())

crimeDataBostonDF = spark.read.parquet("dbfs:/mnt/training/crime-data-2016/Crime-Data-Boston-2016.parquet")
print('Boston:', crimeDataBostonDF.count())
```

```
New York: 468241
Los Angeles: 217945
Chicago: 267872
Philadelphia: 168664
Dallas: 99642
Boston: 218610
```

Question 2: Provide Total count of different Types of Crimes for each of 6 United States cities listed below using crime-data-2016:

- New York
- Los Angeles
- Chicago
- Philadelphia
- Dallas
- Boston

Notice in the Dataframes:

- The `crimeDataNewYorkDF` and `crimeDataBostonDF` DataFrames use different names for the columns.
- The data itself is formatted differently and different names are used for similar concepts.

This is common in a Data Lake. Often files are added to a Data Lake by different groups at different times. The advantage of this strategy is that anyone can contribute information to the Data Lake and that Data Lakes scale to store arbitrarily large and diverse data. The tradeoff for this ease in storing data is that it doesn't have the rigid structure of a traditional relational data model, so the person querying the Data Lake will need to normalize data before extracting useful insights.

Same Type of Data, Different Structure

Examine crime data to determine how to extract homicide statistics.

Because the data sets are pooled together in a Data Lake, each city may use different field names and values to indicate homicides, dates, etc.

For example:

- Some cities use the value "HOMICIDE", "CRIMINAL HOMICIDE" or "MURDER".
- In the New York data, the column is named `offense_description` while in the Boston data, the column is named

Hint: Los Angeles

```
# TODO
from pyspark.sql.functions import countDistinct, lower, col
#crimeDataNewYorkDF.columns
print('New York: ', crimeDataNewYorkDF.select('offenseDescription').distinct().count())

#crimeDataLosAngelesDF.columns
print('Los Angeles: ', crimeDataLosAngelesDF.select('crimeCodeDescription').distinct().count())

#crimeDataChicagoDF.columns
print('Chicago: ', crimeDataChicagoDF.select('primaryType').distinct().count())

#crimeDataPhiladelphiaDF.columns
print('Philadelphia: ', crimeDataPhiladelphiaDF.select('ucr_general_description').distinct().count())
#crimeDataPhiladelphiaDF.select('text_general_code').show()

print('Dallas: ', crimeDataDallasDF.select('typeOfIncident').distinct().count())

#crimeDataBostonDF.columns
print('Boston: ', crimeDataBostonDF.select('OFFENSE_DESCRIPTION').distinct().count())
```

```
New York:  62
Los Angeles:  125
Chicago:  33
Philadelphia:  26
Dallas:  370
Boston:  243
```

Question 3: Calculate the total Robbery count for each of the 3 United States cities listed below using crime-data-2016:

- Los Angeles
- Philadelphia
- Dallas

Hint: For each table, examine the data to figure out how to extract *robbery* statistics.

Each city uses different values to indicate robbery. Commonly used terminology is "larceny", "burglary" or "robbery." These challenges are common in data lakes. To simplify things, restrict yourself to only the word "robbery" (and not attempted-roberty, larceny, or burglary).

Explore the data for the three cities until you understand how each city records robbery information. If you don't want to worry about upper- or lower-case, remember to use the DataFrame `lower()` method to converts column values to lowercase.

Create a DataFrame containing only the robbery-related rows, as shown in the table below.

Hint: For each table, focus your efforts on the column listed below.

Focus on the following columns for each table:

DataFrame Name	Robbery DataFrame Name	Column
`crimeDataLosAngelesDF`	`robberyLosAngelesDF`	`crimeCodeDescription`

Hint: Los Angeles

```
crimeDataLosAngelesDF.select('crimeCodeDescription').where(lower(col('crimeCodeDescription')).contains('robbery')).dis
tinct().collect()
crimeDataPhiladelphiaDF.select('ucr_general_description').where(lower(col('ucr_general_description')).contains('robber
y')).distinct().collect()
crimeDataDallasDF.select('typeOfIncident').where(lower(col('typeOfIncident')).contains('robbery')).distinct().collect(
)

robberyLosAngelesDF = crimeDataLosAngelesDF.where("lower(crimeCodeDescription) == 'robbery'")
robberyPhiladelphiaDF = crimeDataPhiladelphiaDF.where("lower(ucr_general_description) == 'robbery'")
robberyDallasDF = crimeDataDallasDF.where(lower(col('typeOfIncident')).startswith('robbery'))
print('Los Angeles:', robberyLosAngelesDF.count())
print('Philadelphia:', robberyPhiladelphiaDF.count())
print('Dallas:', robberyDallasDF.count())
```

```
Los Angeles: 9048
Philadelphia: 6149
Dallas: 6824
```

Question 4: Find the months with the Highest and Lowest Robbery counts for each of the 3 United States cities listed below using crime-data-2016:

- Los Angeles
- Philadelphia
- Dallas

Hint: Now that you have DataFrames of only the robberies in each city, create DataFrames for each city summarizing the number of robberies in each month.

Your DataFrames must contain two columns:

- `month`: The month number (e.g., 1 for January, 2 for February, etc.).
- `robberies`: The total number of robberies in the month.

Use the following DataFrame names and date columns:

City	DataFrame Name	Date Column
Los Angeles	`robberiesByMonthLosAngelesDF`	`timeOccurred`
Philadelphia	`robberiesByMonthPhiladelphiaDF`	`dispatch_date_time`
Dallas	`robberiesByMonthDallasDF`	`startingDateTime`

Hint: Los Angeles

```
# TODO
from pyspark.sql.functions import month

robberiesByMonthLosAngelesDF = robberyLosAngelesDF.withColumn('month',
month('timeOccurred')).groupBy('month').count().orderBy(col('count').desc())
print('Los Angeles:')
print(f'Month: {robberiesByMonthLosAngelesDF.first()[0]}, Count: {robberiesByMonthLosAngelesDF.first()[1]}')
robberiesByMonthLosAngelesDF_min = robberiesByMonthLosAngelesDF.orderBy(col('count').asc()).first()
print(f'Month: {robberiesByMonthLosAngelesDF_min[0]}, Count: {robberiesByMonthLosAngelesDF_min[1]}')
# spark.createDataFrame([robberiesByMonthLosAngelesDF.first(),
robberiesByMonthLosAngelesDF.orderBy(col('count').asc()).first()]).show()
robberiesByMonthPhiladelphiaDF = robberyPhiladelphiaDF.withColumn('month',
month('dispatch_date_time')).groupBy('month').count().orderBy(col('count').desc())
print('Philadelphia:')
print(f'Month: {robberiesByMonthPhiladelphiaDF.first()[0]}, Count: {robberiesByMonthPhiladelphiaDF.first()[1]}')
robberiesByMonthPhiladelphiaDF_min = robberiesByMonthLosAngelesDF.orderBy(col('count').asc()).first()
print(f'Month: {robberiesByMonthPhiladelphiaDF_min[0]}, Count: {robberiesByMonthPhiladelphiaDF_min[1]}')
# spark.createDataFrame([robberiesByMonthPhiladelphiaDF.first(),
robberiesByMonthPhiladelphiaDF.orderBy(col('count').asc()).first()]).show()
robberiesByMonthDallasDF = robberyDallasDF.withColumn('month',
month('startingDateTime')).groupBy('month').count().orderBy(col('count').desc())
print('Dallas:')
print(f'Month: {robberiesByMonthDallasDF.first()[0]}, Count: {robberiesByMonthDallasDF.first()[1]}')
robberiesByMonthDallasDF_min = robberiesByMonthDallasDF.orderBy(col('count').asc()).first()
print(f'Month: {robberiesByMonthDallasDF_min[0]}, Count: {robberiesByMonthDallasDF_min[1]}')
# spark.createDataFrame([robberiesByMonthDallasDF.first(),
robberiesByMonthDallasDF.orderBy(col('count').asc()).first()]).show()
```

```
Los Angeles:
Month: 12, Count: 853
Month: 2, Count: 675
Philadelphia:
Month: 10, Count: 572
Month: 2, Count: 675
Dallas:
Month: 1, Count: 743
Month: 3, Count: 412
```

Question 5: Combine all three cities robberies-per-month views into one and Find the month with the Highest and Lowest combined Robbery counts using crime-data-2016:

- Los Angeles
- Philadelphia
- Dallas

Create another DataFrame called `combinedRobberiesByMonthDF`, that combines all three robberies-per-month views into one. In creating this view, add a new column called `city`, that identifies the city associated with each row. The final view will have the following columns:

- `city`: The name of the city associated with the row. (Use the strings "Los Angeles", "Philadelphia", and "Dallas".)
- `month`: The month number associated with the row.

Hint: Combined 3 Cities

```
# TODO
from pyspark.sql.functions import lit
from pyspark.sql.functions import sum as spark_sum
la = robberiesByMonthLosAngelesDF.withColumn('city', lit('Los Angeles')).withColumnRenamed('count',
'robbery').orderBy('month')
ph = robberiesByMonthPhiladelphiaDF.withColumn('city', lit('Philadelphia')).withColumnRenamed('count',
'robbery').orderBy('month')
da = robberiesByMonthDallasDF.withColumn('city', lit('Dallas')).withColumnRenamed('count', 'robbery').orderBy('month')
combinedRobberiesByMonthDF = la.union(ph).union(da).select('city', 'month', 'robbery')
totalDF = combinedRobberiesByMonthDF.groupBy('month').agg(spark_sum('robbery').alias('total
robbery')).orderBy(col('total robbery').desc())
spark.createDataFrame([totalDF.first(), totalDF.orderBy(col('total robbery').asc()).first()]).show()
```

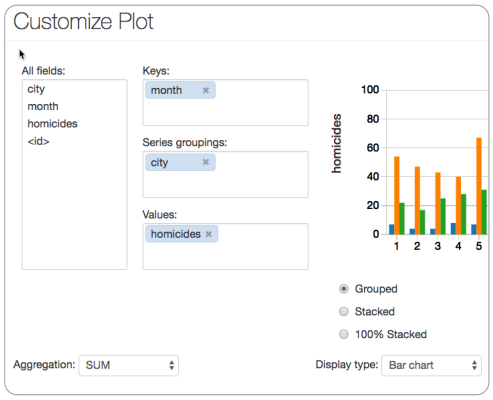
```
+-----+-------------+
|month|total robbery|
```

```
+-----+-------------+
|   12|         2061|
|    2|         1526|
+-----+-------------+
```

Question 6: Plot the robberies per month for each of the three cities, producing one Graph using the contents of `combinedRobberiesByMonthDF`

Adjust the plot options to configure the plot properly, as shown below:

When you first run the cell, you'll get an HTML table as the result. To configure the plot:

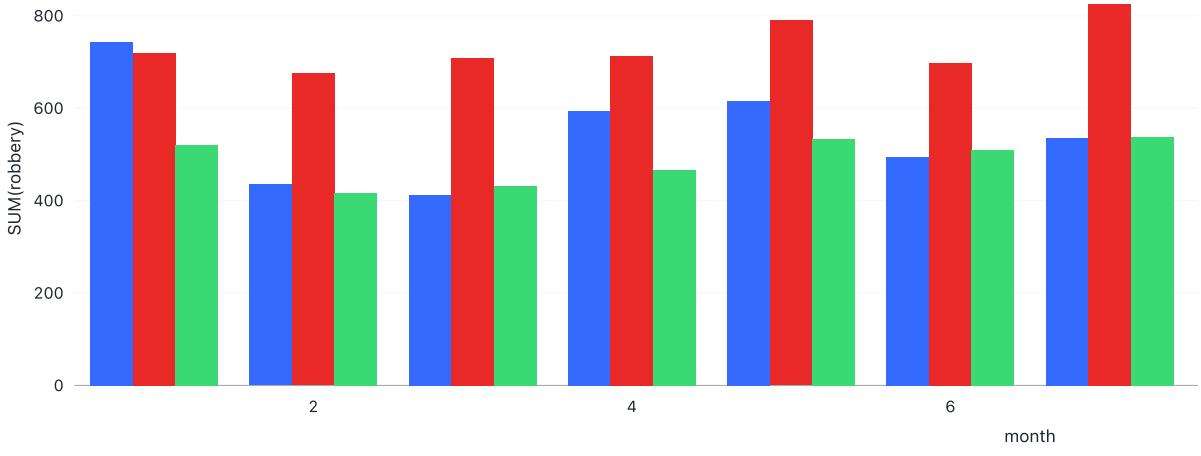


1. Click the graph button.
2. If the plot doesn't look correct, click the **Plot Options** button.
3. Configure the plot similar to the following example.

Hint: Order your results by `month`, then `city`.


```
display(combinedRobberiesByMonthDF)
```

Table Visualization 1



36 rows

Question 7: Find the "per capita robbery rates" using ther Hint below, and plot graph as above for the per capita robbery rates per month for each of the three cities, producing one Graph using the contents of `robberyRatesByCityDF`:

- Los Angeles
- Philadelphia
- Dallas

While the above graph is interesting, it's flawed: it's comparing the raw numbers of robberies, not the per capita robbery rates.

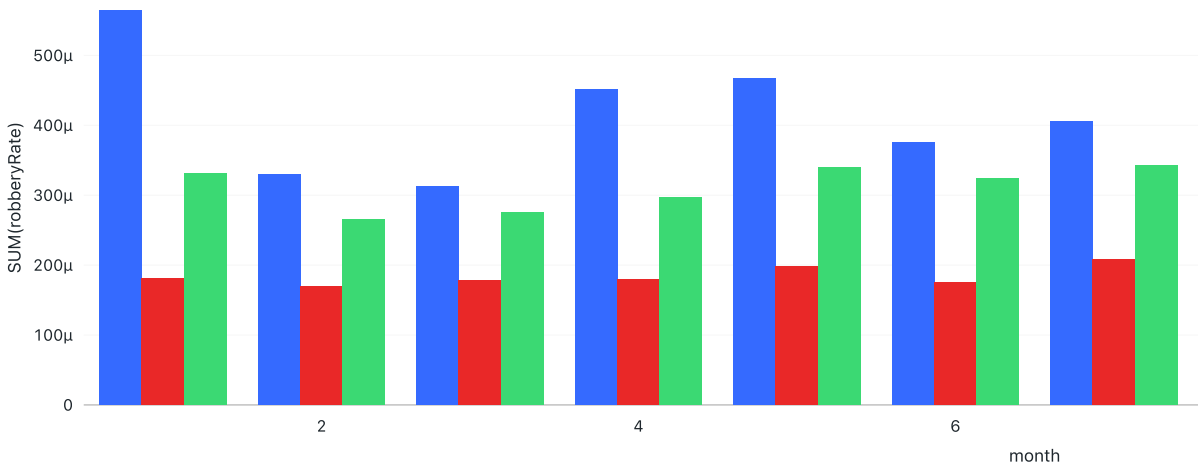
The DataFrame (already created) called `cityDataDF` (dbfs:/mnt/training/City-Data.parquet) contains, among other data, estimated 2016 population values for all United States cities with populations of at least 100,000. (The data is from Wikipedia (https://en.wikipedia.org/wiki/List_of_United_States_cities_by_population).)

- Use the population values in that table to normalize the robberies so they represent per-capita values (total robberies divided by population).
- Save your results in a DataFrame called `robberyRatesByCityDF`.
- The robbery rate value must be stored in a new column, `robberyRate`.

Next, graph the results, as above.

```
cityDataDF = spark.read.parquet("dbfs:/mnt/training/City-Data.parquet")
robberyRatesByCityDF = combinedRobberiesByMonthDF.join(cityDataDF.select('city', 'estPopulation2016'), on='city',
how='left').withColumn('robberyRate', col('robbery') / col('estPopulation2016'))
display(robberyRatesByCityDF)
```

Table Visualization 1



36 rows

Question 8: Find the monthly HOMICIDE counts for each of the 2 United States cities listed below using crime-data-2016, and Combine both cities HOMICIDE-per-month views into one and Find the month with the Highest and Lowest combined HOMICIDE-per-month counts. See Question 6.

- New York
- Boston

Hint: Same Type of Data, Different Structure In this section, we examine crime data to determine how to extract homicide statistics. Each city may use different field names and values to indicate homicides, dates, etc. For example:

- Some cities use the value "HOMICIDE", "CRIMINAL HOMICIDE" or "MURDER".
- In the New York data, the column is named `offenseDescription` while in the Boston data, the column is named `OFFENSE_CODE_GROUP`.
- In the New York data, the date of the event is in the `reportDate`, while in the Boston data, there is a single column named `MONTH`.

To get started, create a temporary view containing only the homicide-related rows. At the same time, normalize the data structure of each table so all the columns (and their values) line up with each other. In the case of New York and Boston, here are the unique characteristics of each data set:

| | Offense-Column | Offense-Value | Reported-Column | Reported-Data Type | |-|----------------------|----------------------|----------------------------------| | New York | `offenseDescription` | starts with "murder" or "homicide" | `reportDate` | `timestamp` | | Boston | `OFFENSE_CODE_GROUP` | "Homicide" | `MONTH` | `integer` | For the upcoming aggregation, you need to alter the New York data set to include a `month` column which can be computed from the `reportDate` column using the

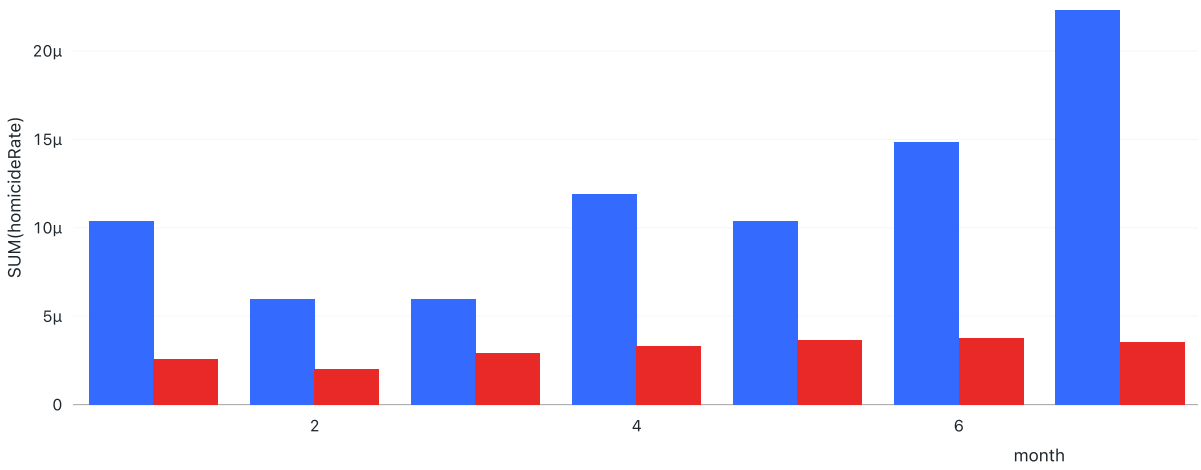
```
homicideNewYorkDF = crimeDataNewYorkDF.select('offenseDescription',
'reportDate').where(lower(col('offenseDescription')).startswith('murder') |
lower(col('offenseDescription')).startswith('homicide'))
homicideNewYorkDF = homicideNewYorkDF.withColumn('month', month('reportDate')).withColumnRenamed('offenseDescription',
'offense').select('offense', 'month')
homicideByMonthNewYorkDF = homicideNewYorkDF.groupBy('month').count().orderBy('month').withColumnRenamed('count',
'homicide')
#print('New York:')
#homicideByMonthNewYorkDF.show()
#spark.createDataFrame([homicideByMonthNewYorkDF.first(),
homicideByMonthNewYorkDF.orderBy(col('count').asc()).first()]).show()
homicideBostonDF = crimeDataBostonDF.select('OFFENSE_CODE_GROUP', 'MONTH').where(lower(col('OFFENSE_CODE_GROUP')) ==
'homicide')
homicideBostonDF = homicideBostonDF.withColumnRenamed('OFFENSE_CODE_GROUP', 'offense').withColumnRenamed('MONTH',
'month').select('offense', 'month')
homicideByMonthBostonDF = homicideBostonDF.groupBy('month').count().orderBy('month').withColumnRenamed('count',
'homicide')
#print('Boston:')
#homicideByMonthBostonDF.show()
#spark.createDataFrame([homicideByMonthBostonDF.first(),
homicideByMonthBostonDF.orderBy(col('count').asc()).first()]).show()
nk = homicideByMonthNewYorkDF.withColumn('city', lit('New York'))
bo = homicideByMonthBostonDF.withColumn('city', lit('Boston'))
combinedHomicideByMonthDF = nk.union(bo).select('city', 'month', 'homicide')
totalDF =
combinedHomicideByMonthDF.groupBy('month').agg(spark_sum('homicide').alias('homicide')).orderBy(col('homicide').desc()
)
spark.createDataFrame([totalDF.first(), totalDF.orderBy(col('homicide').asc()).first()]).show()
```

```
+-----+--------+
|month|homicide|
+-----+--------+
|    8|      50|
|    2|      21|
+-----+--------+
```

Question 9: Find the "per capita HOMICIDE rates" using the Hint in Qn 7, and plot graph as above for the per capita HOMICIDE rates per

```
homicideRatesByCityDF = combinedHomicideByMonthDF.join(cityDataDF.select('city', 'estPopulation2016'), on='city',
how='left').withColumn('homicideRate', col('homicide') / col('estPopulation2016'))
display(homicideRatesByCityDF)
```

Table Visualization 1



24 rows

Question 10: A stretch goal to address a Data Science question on predicting crimes for a city as a time series model. How would you predict future values for monthly Robbery Count rate for Log Angeles using the historical values from crime-data-2016. Data Science is a

Data Preparation

I think it is a good way to start from cleaning up the data we have from the dataset, including removing missing values, duplicates, outliers, etc. Then the next step will be aggregating the data by month and get the monthly Robbery count.

Time Series Model Selection

I think time series model such as simple RNN, LSTM, or GRU model may work well on this task. By lokking at the data in the past, the model learns and can therefore predict the crimes for the city.

Potential problems

1. Many factors of crime are not included in the dataset, and it might lead to low accuracy of prediction
2. The critical crime factor changes all the time, and the model might no learn this part

References

The crime data used in this notebook comes from the following locations:

City	Original Data
Boston	https://data.boston.gov/group/public-safety (https://data.boston.gov/group/public-safety)
Chicago	https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2 (https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2)

City	Original Data
Dallas	https://www.dallasopendata.com/Public-Safety/Police-Incidents/tbnj-w5hb/data (https://www.dallasopendata.com/Public-Safety/Police-Incidents/tbnj-w5hb/data)
Los Angeles	https://data.lacity.org/A-Safe-City/Crime-Data-From-2010-to-Present/y8tr-7khq (https://data.lacity.org/A-Safe-City/Crime-Data-From-2010-to-Present/y8tr-7khq)
New Orleans	https://data.nola.gov/Public-Safety-and-Preparedness/Electronic-Police-Report-2016/4gc2-25he/data (https://data.nola.gov/Public-Safety-and-Preparedness/Electronic-Police-Report-2016/4gc2-25he/data)
New York	https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Historic/qgea-i56i (https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Historic/qgea-i56i)