Table of Contents

[Objective 3](#_Toc118923003)

[Problem Statement 3](#_Toc118923004)

[Download Dataset 3](#_Toc118923005)

[Create database 3](#_Toc118923006)

[Create table schema 3](#_Toc118923007)

[Load data in the table 5](#_Toc118923008)

[Create partitioned table for ny\_parking\_violation\_data 5](#_Toc118923009)

[Set properties for dynamic partitioning 7](#_Toc118923010)

[Load the data in the partitioned table FROM ny\_parking\_violation\_data\_partitioned 7](#_Toc118923011)

[The analysis can be divided into two parts: 8](#_Toc118923012)

[Part-I: Examine the data 8](#_Toc118923013)

[1.) Find the total number of tickets for the year. 8](#_Toc118923014)

[2.) Find out how many unique states the cars which got parking tickets came from. 8](#_Toc118923015)

[3.) Some parking tickets don’t have addresses on them, which is cause for concern. Find out how many such tickets there are (i.e. tickets WHERE either "Street Code 1" or "Street Code 2" or "Street Code 3" is empty) 8](#_Toc118923016)

[Part-II: Aggregation tasks 10](#_Toc118923017)

[1.) How often does each violation code occur? (frequency of violation codes - find the top 5) 10](#_Toc118923018)

[2.) How often does each vehicle body type get a parking ticket? How about the vehicle make? (find the top 5 for both) 10](#_Toc118923019)

[4.) A precinct is a police station that has a certain zone of the city under its command. Find the (5 highest) frequencies of: 10](#_Toc118923020)

[a.) Violating Precincts (this is the precinct of the zone WHERE the violation occurred) 10](#_Toc118923021)

[b.) Issuer Precincts (this is the precinct that issued the ticket) 10](#_Toc118923022)

[5.) Find the violation code frequency across 3 precincts which have issued the most number of tickets - do these precinct zones have an exceptionally high frequency of certain violation codes? 11](#_Toc118923023)

[For top 3 precinct with highest no. of tickets issued – 11](#_Toc118923024)

[Here is the breakdown of highest numbers of violations by violation\_code for each precincts mentioned above. 11](#_Toc118923025)

[ Precinct – 19 11](#_Toc118923026)

[ Precinct – 14 11](#_Toc118923027)

[ Precinct – 18 11](#_Toc118923028)

[Alternatively, below code could be used to get results for all 3 precincts together 12](#_Toc118923029)

[5.) Find out the properties of parking violations across different times of the day: The Violation Time field is specified in a strange format. Find a way to make this into a time attribute that you can use to divide into groups. 12](#_Toc118923030)

[6.) Divide 24 hours into 6 equal discrete bins of time. The intervals you choose are at your discretion. For each of these groups, find the 3 most commonly occurring violations 12](#_Toc118923031)

[Queries to break down of total tickets issued divided in each view 13](#_Toc118923032)

[ Bin 1 13](#_Toc118923033)

[ Bin 2 13](#_Toc118923034)

[ Bin 3 13](#_Toc118923035)

[ Bin 4 13](#_Toc118923036)

[ Bin 5 13](#_Toc118923037)

[ Bin 6 14](#_Toc118923038)

[7.) Now, try another direction. For the 3 most commonly occurring violation codes, find the most common times of day (in terms of the bins from the previous part) 14](#_Toc118923039)

[8.) Let’s try and find some seasonality in this data 14](#_Toc118923040)

[Consider Seasons as below and divide 14](#_Toc118923041)

[By Create View on table - 14](#_Toc118923042)

[b.) Then, find the 3 most common violations for each of these seasons. 15](#_Toc118923043)

[ Most common violation for Spring 15](#_Toc118923044)

[ Most common violation for Summer 15](#_Toc118923045)

[ Most common violation for Fall 15](#_Toc118923046)

[ Most common violation for Winter 15](#_Toc118923047)

Objective - The assignment is meant for you to apply learnings of the module on Hive on a real-life dataset. One of the major objectives of this assignment is gaining familiarity with how an analysis works in Hive and how you can gain insights from large datasets.

Problem Statement - New York City is a thriving metropolis and just like most other cities of similar size, one of the biggest problems its residents face is parking. The classic combination of a huge number of cars and a cramped geography is the exact recipe that leads to a large number of parking tickets.

In an attempt to scientifically analyse this phenomenon, the NYC Police Department regularly collects data related to parking tickets. This data is made available by NYC Open Data portal. We will try and perform some analysis on this data.

Download Dataset - https://data.cityofnewyork.us/browse?q=parking+tickets

Note: Consider only the year 2017 for analysis and not the Fiscal year.

# Create database

CREATE database mini\_project\_2;

# Create table schema

CREATE table ny\_parking\_violation\_data

(

Summons\_Number int,

Plate\_ID string,

Registration\_State string,

Plate\_Type string,

Issue\_Date date,

Violation\_Code int,

Vehicle\_Body\_Type string,

Vehicle\_Make string,

Issuing\_Agency string,

Street\_Code1 int,

Street\_Code2 int,

Street\_Code3 int,

Vehicle\_Expiration\_Date date,

Violation\_Location string,

Violation\_Precinct int,

Issuer\_Precinct int,

Issuer\_Code string,

Issuer\_Command string,

Issuer\_Squad string,

Violation\_Time string,

Time\_First\_Observed string,

Violation\_County string,

Violation\_In\_Front\_Of\_Or\_Opposite string,

House\_Number string,

Street\_Name string,

Intersecting\_Street string,

Date\_First\_Observed int,

Law\_Section int,

Sub\_Division string,

Violation\_Legal\_Code string,

Days\_Parking\_In\_Effect string,

From\_Hours\_In\_Effect string,

To\_Hours\_In\_Effect string,

Vehicle\_Color string,

Unregistered\_Vehicle string,

Vehicle\_Year string,

Meter\_Number string,

Feet\_From\_Curb int,

Violation\_Post\_Code string,

Violation\_Description string,

No\_Standing\_or\_Stopping\_Violation string,

Hydrant\_Violation string,

Double\_Parking\_Violation string

)

row format delimited

fields terminated by ","

tblproperties ("skip.header.line.count=1");

# Load data in the table

load data local inpath 'file:///home/cloudera/mini\_project\_2/Parking\_Violations\_Issued\_Fiscal\_Year\_2017.csv' into table ny\_parking\_violation\_data;

# Create partitioned table for ny\_parking\_violation\_data

CREATE table ny\_parking\_violation\_data\_PARTITIONED

(

Summons\_Number int,

Plate\_ID string,

Registration\_State string,

Plate\_Type string,

Issue\_Date date,

Violation\_Code int,

Vehicle\_Body\_Type string,

Vehicle\_Make string,

Issuing\_Agency string,

Street\_Code1 int,

Street\_Code2 int,

Street\_Code3 int,

Vehicle\_Expiration\_Date date,

Violation\_Location string,

Violation\_Precinct int,

Issuer\_Precinct int,

Issuer\_Code string,

Issuer\_Command string,

Issuer\_Squad string,

Violation\_Time string,

Time\_First\_Observed string,

Violation\_In\_Front\_Of\_Or\_Opposite string,

House\_Number string,

Street\_Name string,

Intersecting\_Street string,

Date\_First\_Observed int,

Law\_Section int,

Sub\_Division string,

Violation\_Legal\_Code string,

Days\_Parking\_In\_Effect string,

From\_Hours\_In\_Effect string,

To\_Hours\_In\_Effect string,

Vehicle\_Color string,

Unregistered\_Vehicle string,

Vehicle\_Year string,

Meter\_Number string,

Feet\_From\_Curb int,

Violation\_Post\_Code string,

Violation\_Description string,

No\_Standing\_or\_Stopping\_Violation string,

Hydrant\_Violation string,

Double\_Parking\_Violation string

)

PARTITIONED by (violation\_county string)

clustered by (violation\_code) sorted by (violation\_code) into 8 buckets

tblproperties ("skip.header.line.count=1");

# Set properties for dynamic partitioning

Set hive.exec.dynamic.partition=true;

Set hive.exec.dynamic.partition.mode=nonstrict;

Set hive.enforce.bucketing=true;

Set hive.vectorization.enable=true;

# Load the data in the partitioned table FROM ny\_parking\_violation\_data\_partitioned

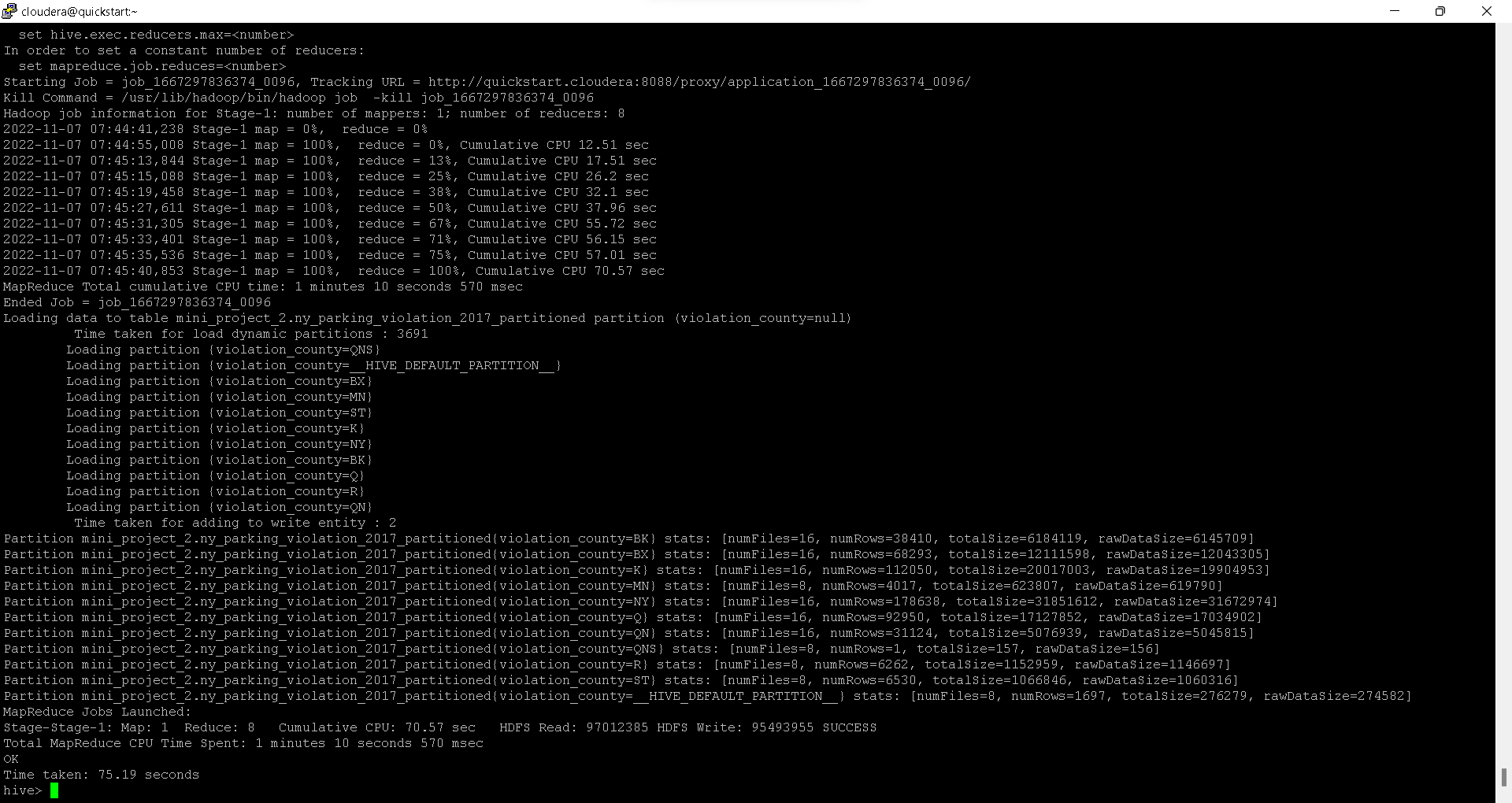
Insert into ny\_parking\_violation\_data\_PARTITIONED partition(violation\_county)

SELECT Summons\_Number, Plate\_ID, Registration\_State, Plate\_Type, Issue\_Date, Violation\_Code, Vehicle\_Body\_Type, Vehicle\_Make, Issuing\_Agency, Street\_Code1, Street\_Code2, Street\_Code3, Vehicle\_Expiration\_Date, Violation\_Location, Violation\_Precinct, Issuer\_Precinct, Issuer\_Code, Issuer\_Command, Issuer\_Squad, Violation\_Time, Time\_First\_Observed, Violation\_In\_Front\_Of\_Or\_Opposite,House\_Number, Street\_Name, Intersecting\_Street, Date\_First\_Observed, Law\_Section, Sub\_Division, Violation\_Legal\_Code, Days\_Parking\_In\_Effect, FROM\_Hours\_In\_Effect, To\_Hours\_In\_Effect, Vehicle\_Color,

Unregistered\_Vehicle, Vehicle\_Year, Meter\_Number, Feet\_From\_Curb, Violation\_Post\_Code,

Violation\_Description, No\_Standing\_or\_Stopping\_Violation, Hydrant\_Violation, Double\_Parking\_Violation , Violation\_County

FROM ny\_parking\_violation\_data WHERE year(issue\_date) = ‘2017’



# The analysis can be divided into two parts:

# Part-I: Examine the data

# Find the total number of tickets for the year.

SELECT count(distinct summons\_number) as Total\_violations

FROM ny\_parking\_violation\_data\_partitioned;

# Find out how many unique states the cars which got parking tickets came from.

SELECT registration\_state, count(distinct plate\_id) as Total\_cars\_by\_state\_violation

FROM ny\_parking\_violation\_data\_partitioned

GROUP BY registration\_state order by Total\_cars\_by\_state\_violation desc;

# Some parking tickets don’t have addresses on them, which is cause for concern. Find out how many such tickets there are (i.e. tickets WHERE either "Street Code 1" or "Street Code 2" or "Street Code 3" is empty)

SELECT registration\_state, count(summons\_number) as Tickets\_w\_o\_add

FROM ny\_parking\_violation\_data

WHERE street\_code1 = 0 or street\_code2 = 0

OR street\_code3 = 0 GROUP BY registration\_state order by Tickets\_w\_o\_add desc;

# Part-II: Aggregation tasks

# 1.) How often does each violation code occur? (frequency of violation codes - find the top 5)

SELECT violation\_code, count(violation\_code) as frequency\_of\_violation\_code

FROM ny\_parking\_violation\_data\_partitioned

GROUP BY violation\_code order\_by frequency\_of\_violation\_code desc limit 5;

# 2.) How often does each vehicle body type get a parking ticket? How about the vehicle make? (find the top 5 for both)

SELECT vehicle\_body\_type, count(distinct summons\_number) total\_tickets

FROM ny\_parking\_violation\_data\_partitioned

GROUP BY vehicle\_body\_type order by total\_tickets desc limit 5;

SELECT vehicle\_make, count(distinct summons\_number) total\_tickets

FROM ny\_parking\_violation\_data\_ partitioned

GROUP BY vehicle\_make order by total\_tickets desc limit 5;

# A precinct is a police station that has a certain zone of the city under its command. Find the (5 highest) frequencies of:

# Violating Precincts (this is the precinct of the zone WHERE the violation occurred)

SELECT violation\_precinct, count(summons\_number) as total\_tickets

FROM ny\_parking\_violation\_data\_ partitioned

GROUP BY violation\_precinct order by total\_tickets desc limit 5;

# b.) Issuer Precincts (this is the precinct that issued the ticket)

SELECT issuer\_precinct, count(summons\_number) as total\_tickets

FROM ny\_parking\_violation\_data\_ partitioned

GROUP BY issuer\_precinct order by total\_tickets desc limit 5;

# Find the violation code frequency across 3 precincts which have issued the most number of tickets - do these precinct zones have an exceptionally high frequency of certain violation codes?

# For top 3 precinct with highest no. of tickets issued –

SELECT issuer\_precinct, count(violation\_code) as total\_tickets

FROM ny\_parking\_violation\_data\_partitioned

GROUP BY issuer\_precinct , order by total\_tickets desc limit 7;

(\*\*limit by 7 because, to exclude records with missing precinct data\*\*)

* Precincts with highest tickets are 19, 14 & 18 in respective order

# Here is the breakdown of highest numbers of violations by violation\_code for each precincts mentioned above.

# Precinct – 19

SELECT violation\_code, count(violation\_code) as total\_tickets

FROM ny\_parking\_violation\_data\_partitioned

WHERE issuer\_precinct = 19

GROUP BY violation\_code order by total\_tickets desc limit 3;

# Precinct – 14

SELECT violation\_code, count(violation\_code) as total\_tickets

FROM ny\_parking\_violation\_data\_partitioned

WHERE issuer\_precinct = 19

GROUP BY violation\_code order by total\_tickets desc limit 3;

# Precinct – 18

SELECT violation\_code, count(violation\_code) as total\_tickets

FROM ny\_parking\_violation\_data\_partitioned

WHERE issuer\_precinct = 19

GROUP BY violation\_code order by total\_tickets desc limit 3;

# Alternatively, below code could be used to get results for all 3 precincts together

SELECT Issuer\_Precinct,Violation\_Code, count(\*) as Total\_tickets

FROM ny\_parking\_violation\_data

WHERE Issuer\_Precinct in (18,19,14)

GROUP BY Issuer\_Precinct,Violation\_Code order by Total\_tickets desc limit 10;

# 5.) Find out the properties of parking violations across different times of the day: The Violation Time field is specified in a strange format. Find a way to make this into a time attribute that you can use to divide into groups.

SELECT FROM\_UNIXTIME(UNIX\_TIMESTAMP(REGEXP\_EXTRACT(violation\_time,'(.\*)[A-Z]',1),'HHmm'),"HH:mm") as time FROM ny\_parking\_violation\_data limit 5;

# Divide 24 hours into 6 equal discrete bins of time. The intervals you choose are at your discretion. For each of these groups, find the 3 most commonly occurring violations

CREATE VIEW nyparking\_violation PARTITIONED on (Violation\_Code) AS

SELECT Summons\_Number, Violation\_Time, Issuer\_Precinct,

CASE

WHEN SUBSTRING(Violation\_Time,1,2) IN ('00','01','02','03','12')

AND UPPER(SUBSTRING (Violation\_Time,-1))='A' THEN 1

WHEN SUBSTRING(Violation\_Time,1,2) IN ('04','05','06','07')

AND UPPER(SUBSTRING (Violation\_Time,-1))='A' THEN 2

WHEN SUBSTRING(Violation\_Time,1,2) IN ('08','09','10','11')

AND UPPER(SUBSTRING (Violation\_Time,-1))='A' THEN 3

WHEN substring(Violation\_Time,1,2) IN ('12','00','01','02','03') AND

UPPER(SUBSTRING (Violation\_Time,-1))='P' THEN 4

WHEN SUBSTRING(Violation\_Time,1,2) IN ('04','05','06','07') AND

UPPER(SUBSTRING(Violation\_Time,-1))='P' THEN 5

WHEN SUBSTRING(Violation\_Time,1,2) IN ('08','09','10','11') AND

UPPER(SUBSTRING(Violation\_Time,-1))='P' THEN 6

ELSE NULL

END AS Violation\_Time\_bin, Violation\_Code

FROM ny\_parking\_violation\_data

WHERE Violation\_Time is NOT NULL OR (length(Violation\_Time)=5

AND UPPER(SUBSTRING(Violation\_Time,-1))in ('A','P')

AND SUBSTRING(Violation\_Time,1,2) in ('00','01','02','03','04','05','06','07', '08','09','10','11','12'));

# Queries to break down of total tickets issued divided in each view

# Bin 1

SELECT violation\_code, count(\*) tickets\_issued

FROM nyparking\_violation

WHERE violation\_Time\_bin == 1 GROUP BY violation\_code ORDER BY tickets\_issued desc LIMIT 3;

# Bin 2

SELECT violation\_code, count(\*) tickets\_issued

FROM nyparking\_violation

WHERE violation\_Time\_bin == 2 GROUP BY violation\_code ORDER BY tickets\_issued desc LIMIT 3;

# Bin 3

SELECT violation\_code, count(\*) tickets\_issued

FROM nyparking\_violation

WHERE violation\_Time\_bin == 3 GROUP BY violation\_code ORDER BY tickets\_issued desc LIMIT 3;

# Bin 4

SELECT violation\_code, count(\*) tickets\_issued

FROM nyparking\_violation

WHERE violation\_Time\_bin == 4 GROUP BY violation\_code ORDER BY tickets\_issued desc LIMIT 3;

# Bin 5

SELECT violation\_code, count(\*) tickets\_issued

FROM nyparking\_violation

WHERE violation\_Time\_bin == 5 GROUP BY violation\_code ORDER BY tickets\_issued desc LIMIT 3;

# Bin 6

SELECT violation\_code, count(\*) tickets\_issued

FROM nyparking\_violation

WHERE violation\_Time\_bin == 6 GROUP BY violation\_code ORDER BY tickets\_issued desc LIMIT 3;

# Now, try another direction. For the 3 most commonly occurring violation codes, find the most common times of day (in terms of the bins from the previous part)

SELECT violation\_code Violations, Violation\_Time\_bin Time\_Bin, count(\*) AS Tickets\_Issued

FROM nyparking\_violation

GROUP BY violation\_code, Violation\_Time\_bin ORDER BY Tickets\_Issued desc LIMIT 10;

# Let’s try and find some seasonality in this data

a.) First, divide the year into some number of seasons, and find frequencies of tickets for each season. (Hint: A quick Google search reveals the following seasons in NYC: Spring (March, April, March); Summer (June, July, August); Fall (September, October, November); Winter (December, January, February))

Consider Seasons as below and divide these into Views

Spring: March, April, May

Summer: June, July, August

Fall: September, October, November

Winter: December, January, February

# By Create View on table with bins for each season-

CREATE VIEW parking\_violation\_by\_seasons\_bin AS

SELECT Summons\_number,

CASE

WHEN SUBSTRING(issue\_date,4,2) IN ('03', '04', '05') THEN 'Spring'

WHEN SUBSTRING(issue\_date,4,2) IN ('06', '07', '08') THEN 'Summer'

WHEN SUBSTRING(issue\_date,4,2) IN ('09', '10', '11') THEN 'Fall'

WHEN SUBSTRING(issue\_date,4,2) IN ('12', '01', '02') THEN 'Winter'

ELSE 'Unknown' END AS Season, violation\_code

FROM ny\_parking\_violation\_2017;

# b.) Then, find the 3 most common violations for each of these seasons.

# Most common violation for Spring

SELECT violation\_code, COUNT(\*) AS Tickets\_issued

FROM parking\_violation\_by\_seasons\_bin WHERE Season = 'Spring'

GROUP BY violation\_code ORDER BY Tickets\_issued desc LIMIT 3;

# Most common violation for Summer

SELECT violation\_code, COUNT(\*) AS Tickets\_issued

FROM parking\_violation\_by\_seasons\_bin WHERE Season = 'Summer'

GROUP BY violation\_code ORDER BY Tickets\_issued desc LIMIT 3;

# Most common violation for Fall

SELECT violation\_code, COUNT(\*) AS Tickets\_issued

FROM parking\_violation\_by\_seasons\_bin WHERE Season = 'Fall'

GROUP BY violation\_code ORDER BY Tickets\_issued desc LIMIT 3;

# Most common violation for Winter

SELECT violation\_code, COUNT(\*) AS Tickets\_issued

FROM parking\_violation\_by\_seasons\_bin WHERE Season = 'Winter '

GROUP BY violation\_code ORDER BY Tickets\_issued desc LIMIT 3;