#/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Load SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

spark\_path <- 'usr\local\spark'

if(nchar(Sys.getenv("SPARK\_HOME")) < 1) {

Sys.setenv(SPARK\_HOME=spark\_path)

}

library(SparkR, lib.loc= c(file.path(Sys.getenv("SPARK\_HOME"), "R", "lib")))

#/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Initialising a Spark session \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

sparkR.session(master = "yarn-client", sparkConfig = list(spark.driver.memory = "1g"))

#/\*\*\*\*\*\*\*\* Before executing any hive-sql query from RStudio, you need to add a jar file in RStudio \*\*\*\*\*\*\*\*\*\*/

sql("ADD JAR /opt/cloudera/parcels/CDH/lib/hive/lib/hive-hcatalog-core-1.1.0-cdh5.11.2.jar")

#/\*\*\*\*\*\*\*\*\*\*\* Loading the Dataset from the HDFS to Sparkdata frame \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_SPK\_DF <- SparkR::read.df("/common\_folder/nyc\_parking/Parking\_Violations\_Issued\_-\_Fiscal\_Year\_2017.csv", "CSV", header="true", inferSchema = "true")

# /\*\*\*\*\*\*\*\*\*\*\*\* 1. The Column names has spaces in the dataset and we have to remove the spaces inorder to run through the SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*/

colnames(Parking\_Tickets\_SPK\_DF)

Parking\_Tickets\_SPK\_DF <- rename(Parking\_Tickets\_SPK\_DF, Summons\_Number=Parking\_Tickets\_SPK\_DF$`Summons Number`,

Plate\_ID=Parking\_Tickets\_SPK\_DF$`Plate ID`,Registration\_State=Parking\_Tickets\_SPK\_DF$`Registration State`,

Issue\_Date=Parking\_Tickets\_SPK\_DF$`Issue Date`,Violation\_Code=Parking\_Tickets\_SPK\_DF$`Violation Code`,

Vehicle\_Body\_Type=Parking\_Tickets\_SPK\_DF$`Vehicle Body Type`,Vehicle\_Make=Parking\_Tickets\_SPK\_DF$`Vehicle Make`,

Violation\_Precinct=Parking\_Tickets\_SPK\_DF$`Violation Precinct`,Issuer\_Precinct=Parking\_Tickets\_SPK\_DF$`Issuer Precinct`,

Violation\_Time=Parking\_Tickets\_SPK\_DF$`Violation Time`)

#/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Examine the data, find summary statistics \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

printSchema(Parking\_Tickets\_SPK\_DF)

# root

# |-- Summons Number: long (nullable = true)

# |-- Plate ID: string (nullable = true)

# |-- Registration State: string (nullable = true)

# |-- Issue Date: timestamp (nullable = true)

# |-- Violation Code: integer (nullable = true)

# |-- Vehicle Body Type: string (nullable = true)

# |-- Vehicle Make: string (nullable = true)

# |-- Violation Precinct: integer (nullable = true)

# |-- Issuer Precinct: integer (nullable = true)

# |-- Violation Time: string (nullable = true)

head(Parking\_Tickets\_SPK\_DF,5)

# Summons Number Plate ID Registration State Issue Date Violation Code Vehicle Body Type Vehicle Make Violation Precinct

# 1 5092469481 GZH7067 NY 2016-07-10 7 SUBN TOYOT 0

# 2 5092451658 GZH7067 NY 2016-07-08 7 SUBN TOYOT 0

# 3 4006265037 FZX9232 NY 2016-08-23 5 SUBN FORD 0

# 4 8478629828 66623ME NY 2017-06-14 47 REFG MITSU 14

# 5 7868300310 37033JV NY 2016-11-21 69 DELV INTER 13

# Issuer Precinct Violation Time

# 1 0 0143A

# 2 0 0400P

# 3 0 0233P

# 4 14 1120A

# 5 13 0555P

nrow(Parking\_Tickets\_SPK\_DF)

# 10803028

ncol(Parking\_Tickets\_SPK\_DF)

# 10

#/\*\*\*\*\*\*\*\*\*\*\*\* For using SQL, you need to create a temporary view \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

createOrReplaceTempView(Parking\_Tickets\_SPK\_DF, "Parking\_Tickets\_SQL\_tbl")

#/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* EDA \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# 1. For the scope of this analysis, we wish to analyse the parking tickets over the year 2017.

#/\*\*\*\*\*\*\*\*\*\*\*\*\*\* filtering the Issue date for the 2017 Year alone Through SPARK Syntaxes \*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# Summons Number Plate ID Registration State Issue Date Violation Code Vehicle Body Type Vehicle Make Violation Precinct

# 1 8478629828 66623ME NY 2017-06-14 47 REFG MITSU 14

# 2 5096917368 FZD8593 NY 2017-06-13 7 SUBN ME/BE 0

# 3 1407740258 2513JMG NY 2017-01-11 78 DELV FRUEH 106

# 4 1413656420 T672371C NY 2017-02-04 40 TAXI TOYOT 73

# 5 8480309064 51771JW NY 2017-01-26 64 VAN INTER 17

# 6 1416638830 GLP367 NY 2017-04-30 20 SUBN DODGE 17

# Issuer Precinct Violation Time

# 1 14 1120A

# 2 0 0852P

# 3 106 0015A

# 4 73 0525A

# 5 17 0256P

# 6 17 1232A

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# 5431909

#/\*\*\*\*\*\*\*\*\*\*\*\*\*\* filtering the Issue date for the 2017 Year(01-01-2017 to 31-12-2017) alone Through SQL queries \*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_SQL\_2017 <- SparkR::sql("select \* from Parking\_Tickets\_SQL\_tbl where Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'")

head(Parking\_Tickets\_SQL\_2017)

# Summons\_Number Plate\_ID Registration\_State Issue\_Date Violation\_Code Vehicle\_Body\_Type Vehicle\_Make Violation\_Precinct

# 1 8478629828 66623ME NY 2017-06-14 47 REFG MITSU 14

# 2 5096917368 FZD8593 NY 2017-06-13 7 SUBN ME/BE 0

# 3 1407740258 2513JMG NY 2017-01-11 78 DELV FRUEH 106

# 4 1413656420 T672371C NY 2017-02-04 40 TAXI TOYOT 73

# 5 8480309064 51771JW NY 2017-01-26 64 VAN INTER 17

# 6 1416638830 GLP367 NY 2017-04-30 20 SUBN DODGE 17

# Issuer\_Precinct Violation\_Time

# 1 14 1120A

# 2 0 0852P

# 3 106 0015A

# 4 73 0525A

# 5 17 0256P

# 6 17 1232A

nrow(Parking\_Tickets\_SQL\_2017)

# 5431909

# 2. Checking whether the Registration State Column for the 2017 Year has the Null Values

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Registration\_State =="" &

Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# 0 --> no rows with Null Registration\_State

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl where Registration\_State is null

and Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'"))

# 0--> no rows with Null Registration\_State

# 3. Checking whether the Plate\_ID Column for the 2017 Year has the Null Values

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Plate\_ID =="" &

Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# 0 --> no rows with Null Plate\_ID

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl where Plate\_ID is null

and Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'"))

# 0--> no rows with Null Plate\_ID

# 3. Checking whether the Issue\_Date Column has the Null Values

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Issue\_Date==""))

# 0 --> no rows with Null Issue\_Date

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl where Issue\_Date is null"))

# 0--> no rows with Null Issue\_Date

# [1] Summons\_Number Plate\_ID Registration\_State Issue\_Date Violation\_Code Vehicle\_Body\_Type

# [7] Vehicle\_Make Violation\_Precinct Issuer\_Precinct Violation\_Time

# 4. Checking whether the Violation\_Code Column has the Null Values

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Violation\_Code=="" &

Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# 0 --> no rows with Null Violation\_Code

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl where Violation\_Code is null

and Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'"))

# 0 --> no rows with Null Violation\_Code

# 5. Checking whether the Vehicle\_Body\_Type Column has the Null Values

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Vehicle\_Body\_Type=="" &

Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# 0 --> no rows with Null Vehicle\_Body\_Type

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl where Vehicle\_Body\_Type is null

and Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'"))

# 0 --> no rows with Null Vehicle\_Body\_Type

# 6. Checking whether the Vehicle\_Make Column has the Null Values

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Vehicle\_Make=="" &

Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# 0 --> no rows with Null Vehicle\_Make

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl where Vehicle\_Make is null

and Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'"))

# 0 --> no rows with Null Vehicle\_Make

# 7. Checking whether the Violation\_Precinct Column has the Null Values

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Violation\_Precinct=="" &

Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# 0 --> no rows with Null Violation\_Precinct

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl where Violation\_Precinct is null

and Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'"))

# 0 --> no rows with Null Violation\_Precinct

# 8. Checking whether the Issuer\_Precinct Column has the Null Values

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Issuer\_Precinct=="" &

Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# 0 --> no rows with Null Issuer\_Precinct

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl where Issuer\_Precinct is null

and Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'"))

# 0 --> no rows with Null Issuer\_Precinct

# 9. Checking whether the Violation\_Time Column has the Null Values

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Violation\_Time=="" &

Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# 0 --> no rows with Null Violation\_Time

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl where Violation\_Time is null

and Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'"))

# 0 --> no rows with Null Violation\_Time

# 10. Checking duplicate Summons number

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# library(sparklyr)

#

# nrow(summarize(group\_by(Parking\_Tickets\_SPK\_DF,Parking\_Tickets\_SPK\_DF$Summons\_Number),count=n(Parking\_Tickets\_SPK\_DF$Summons\_Number)))

#

# # filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31"))

# # 0 --> no rows with Null Violation\_Time

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(DISTINCT Summons\_Number) from Parking\_Tickets\_SQL\_tbl

where Issue\_Date >= '2017-01-01' and Issue\_Date <= '2017-12-31'"))

# 5431909 --> no duplicates available with Summons\_Number

# There are no duplicates and Null values available in the dataset but we will create the view based by filtering the issue\_date for 2017 year alone

Parking\_Tickets\_SPK\_DF\_2017 <- filter(Parking\_Tickets\_SPK\_DF, Parking\_Tickets\_SPK\_DF$Issue\_Date >= "2017-01-01" & Parking\_Tickets\_SPK\_DF$Issue\_Date <= "2017-12-31")

createOrReplaceTempView(Parking\_Tickets\_SPK\_DF\_2017 , "Parking\_Tickets\_SQL\_tbl\_2017")

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* END OF EDA \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Examine the data \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

################## 1. Find the total number of tickets for the year. ###########################

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(summarize(groupBy(Parking\_Tickets\_SPK\_DF\_2017),

count=n(Parking\_Tickets\_SPK\_DF\_2017$Summons\_Number)))

# count

# 5431909

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(Summons\_Number) as no\_of\_tickets from Parking\_Tickets\_SQL\_tbl\_2017"))

# count(Summons\_Number)

# 5431909 --> the total number of tickets for the year 2017

################### 2. Find out the number unique states from where the cars that got parking tickets came from. #############

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(distinct Registration\_State) as unique\_states\_count from Parking\_Tickets\_SQL\_tbl\_2017 where Summons\_Number is not null and length(Summons\_Number) >1"))

# unique\_states\_count

# 65

################### 2.1 Find out the number of tickets from each unique states from where the cars that got parking tickets came from. #############

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_SPK\_DF\_2017\_unique\_state <- summarize(groupBy(Parking\_Tickets\_SPK\_DF\_2017,Parking\_Tickets\_SPK\_DF\_2017$Registration\_State),

Ticket\_count=n(Parking\_Tickets\_SPK\_DF\_2017$Summons\_Number))

Parking\_Tickets\_SPK\_DF\_2017\_unique\_state\_COUNT <- arrange(Parking\_Tickets\_SPK\_DF\_2017\_unique\_state, desc(Parking\_Tickets\_SPK\_DF\_2017\_unique\_state$Ticket\_count))

head(Parking\_Tickets\_SPK\_DF\_2017\_unique\_state\_COUNT)

# Registration\_State Ticket\_count

# 1 NY 4273944

# 2 NJ 475824

# 3 PA 140285

# 4 CT 70403

# 5 FL 69468

# 6 IN 45525

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select Registration\_State,count(1) AS Ticket\_count from Parking\_Tickets\_SQL\_tbl\_2017

group by Registration\_State order by 2 desc"))

# Registration\_State Ticket\_count

# 1 NY 4273944

# 2 NJ 475824

# 3 PA 140285

# 4 CT 70403

# 5 FL 69468

# 6 IN 45525

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Registration\_State vs. Number of Tickets) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

library(ggplot2)

printSchema(Parking\_Tickets\_SPK\_DF\_2017\_unique\_state\_COUNT)

Parking\_Tickets\_R\_DF\_2017\_unique\_state\_COUNT <- as.data.frame(head(select(Parking\_Tickets\_SPK\_DF\_2017\_unique\_state\_COUNT,

list(Parking\_Tickets\_SPK\_DF\_2017\_unique\_state\_COUNT$Registration\_State,

Parking\_Tickets\_SPK\_DF\_2017\_unique\_state\_COUNT$Ticket\_count)),5))

ggplot(Parking\_Tickets\_R\_DF\_2017\_unique\_state\_COUNT,aes(x=Registration\_State, y=Ticket\_count)) +geom\_col() + xlab("Registration\_State") + ylab("Number of Tickets") +

ggtitle("Plot1. Registration\_State vs. Number of Tickets") + geom\_text(aes(label=Ticket\_count),vjust=-0.3)

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* END OF Examine the data \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Aggregation tasks \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

################## 1. How often does each violation code occur? Display the frequency of the top five violation codes. #############

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_SPK\_DF\_2017\_VC <- summarize(groupBy(Parking\_Tickets\_SPK\_DF\_2017,Parking\_Tickets\_SPK\_DF\_2017$Violation\_Code),

Number\_of\_Tickets=n(Parking\_Tickets\_SPK\_DF\_2017$Violation\_Code))

Parking\_Tickets\_SPK\_DF\_2017\_VC\_count <- arrange(Parking\_Tickets\_SPK\_DF\_2017\_VC, desc(Parking\_Tickets\_SPK\_DF\_2017\_VC$Number\_of\_Tickets))

head(Parking\_Tickets\_SPK\_DF\_2017\_VC\_count,5)

# Violation\_Code count

# 1 21 768085

# 2 36 662765

# 3 38 542079

# 4 14 476664

# 5 20 319644

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select COUNT(1) AS Freq\_violation\_code,Violation\_Code from Parking\_Tickets\_SQL\_tbl\_2017

group by Violation\_Code order by 1 desc limit 5"))

# Freq\_violation\_code Violation\_Code

# 1 768085 21

# 2 662765 36

# 3 542079 38

# 4 476664 14

# 5 319644 20

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Violation\_Code vs. Number of Tickets) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

printSchema(Parking\_Tickets\_SPK\_DF\_2017\_VC\_count)

Parking\_Tickets\_SPK\_DF\_2017\_VC\_count$Violation\_Code <- cast(Parking\_Tickets\_SPK\_DF\_2017\_VC\_count$Violation\_Code,'string')

Parking\_Tickets\_R\_DF\_2017\_VC\_count <- as.data.frame(head(select(Parking\_Tickets\_SPK\_DF\_2017\_VC\_count,list(Parking\_Tickets\_SPK\_DF\_2017\_VC\_count$Violation\_Code,Parking\_Tickets\_SPK\_DF\_2017\_VC\_count$Number\_of\_Tickets)),5))

ggplot(Parking\_Tickets\_R\_DF\_2017\_VC\_count,aes(x=Violation\_Code, y=Number\_of\_Tickets)) +geom\_col() + xlab("Violation\_Code") + ylab("Number of Tickets") +

ggtitle("Plot2. Violation\_Code vs. Number of Tickets") + geom\_text(aes(label=Number\_of\_Tickets),vjust=-0.3)

######################### 2. How often does each 'vehicle body type' get a parking ticket? How about the 'vehicle make'? #############

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# /\*\*\*\*\*\*\*\*\*\*\*\* Vehicle\_Body\_Type \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_SPK\_DF\_2017\_VBT <- summarize(groupBy(Parking\_Tickets\_SPK\_DF\_2017,Parking\_Tickets\_SPK\_DF\_2017$Vehicle\_Body\_Type),

Number\_of\_Tickets=n(Parking\_Tickets\_SPK\_DF\_2017$Vehicle\_Body\_Type))

Parking\_Tickets\_SPK\_DF\_2017\_VBT\_count <- arrange(Parking\_Tickets\_SPK\_DF\_2017\_VBT, desc(Parking\_Tickets\_SPK\_DF\_2017\_VBT$Number\_of\_Tickets))

head(Parking\_Tickets\_SPK\_DF\_2017\_VBT\_count,5)

# Vehicle\_Body\_Type Number\_of\_Tickets

# 1 SUBN 1883953

# 2 4DSD 1547312

# 3 VAN 724027

# 4 DELV 358984

# 5 SDN 194191

# /\*\*\*\*\*\*\*\*\*\*\*\* Vehicle\_Make \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_SPK\_DF\_2017\_VM <- summarize(groupBy(Parking\_Tickets\_SPK\_DF\_2017,Parking\_Tickets\_SPK\_DF\_2017$Vehicle\_Make),

Number\_of\_Tickets=n(Parking\_Tickets\_SPK\_DF\_2017$Vehicle\_Make))

Parking\_Tickets\_SPK\_DF\_2017\_VM\_count <- arrange(Parking\_Tickets\_SPK\_DF\_2017\_VM, desc(Parking\_Tickets\_SPK\_DF\_2017\_VM$Number\_of\_Tickets))

head(Parking\_Tickets\_SPK\_DF\_2017\_VM\_count,5)

# Vehicle\_Make Number\_of\_Tickets

# 1 FORD 636843

# 2 TOYOT 605288

# 3 HONDA 538884

# 4 NISSA 462017

# 5 CHEVR 356031

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# /\*\*\*\*\*\*\*\*\*\*\*\* Vehicle\_Body\_Type \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select COUNT(1) AS Parking\_Ticket,Vehicle\_Body\_Type from Parking\_Tickets\_SQL\_tbl\_2017

group by Vehicle\_Body\_Type order by 1 desc limit 5"))

#

# Parking\_Ticket Vehicle\_Body\_Type

# 1 1883953 SUBN

# 2 1547312 4DSD

# 3 724027 VAN

# 4 358984 DELV

# 5 194191 SDN

# /\*\*\*\*\*\*\*\*\*\*\*\* Vehicle\_Make \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select COUNT(1) AS Parking\_Ticket,Vehicle\_Make from Parking\_Tickets\_SQL\_tbl\_2017

group by Vehicle\_Make order by 1 desc limit 5"))

# Parking\_Ticket Vehicle\_Make

# 1 636843 FORD

# 2 605288 TOYOT

# 3 538884 HONDA

# 4 462017 NISSA

# 5 356031 CHEVR

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Vehicle\_Make vs. Number of Tickets) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

printSchema(Parking\_Tickets\_SPK\_DF\_2017\_VM\_count)

Parking\_Tickets\_SPK\_R\_2017\_VM\_count <- as.data.frame(head(select(Parking\_Tickets\_SPK\_DF\_2017\_VM\_count,

list(Parking\_Tickets\_SPK\_DF\_2017\_VM\_count$Vehicle\_Make,

Parking\_Tickets\_SPK\_DF\_2017\_VM\_count$Number\_of\_Tickets)),5))

ggplot(Parking\_Tickets\_SPK\_R\_2017\_VM\_count,aes(x=Vehicle\_Make, y=Number\_of\_Tickets)) +geom\_col() + xlab("Vehicle\_Make") + ylab("Number of Tickets") +

ggtitle("Plot3. Vehicle\_Make vs. Number of Tickets") + geom\_text(aes(label=Number\_of\_Tickets),vjust=-0.3)

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Vehicle\_Body\_Type vs. Number of Tickets) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

printSchema(Parking\_Tickets\_SPK\_DF\_2017\_VBT\_count)

Parking\_Tickets\_SPK\_R\_2017\_VBT\_count <- as.data.frame(head(select(Parking\_Tickets\_SPK\_DF\_2017\_VBT\_count,

list(Parking\_Tickets\_SPK\_DF\_2017\_VBT\_count$Vehicle\_Body\_Type,

Parking\_Tickets\_SPK\_DF\_2017\_VBT\_count$Number\_of\_Tickets)),5))

ggplot(Parking\_Tickets\_SPK\_R\_2017\_VBT\_count,aes(x=Vehicle\_Body\_Type, y=Number\_of\_Tickets)) +geom\_col() + xlab("Vehicle\_Body\_Type") + ylab("Number of Tickets") +

ggtitle("Plot4. Parking\_Tickets\_SPK\_R\_2017\_VBT\_count vs. Number of Tickets") + geom\_text(aes(label=Number\_of\_Tickets),vjust=-0.3)

########### 3. A precinct is a police station that has a certain zone of the city under its command. Find the (5 highest) frequency of tickets for each of the following:

########### 'Violation Precinct' (this is the precinct of the zone where the violation occurred). Using this, can you make any insights for parking violations in any specific areas of the city?

########### 'Issuer Precinct' (this is the precinct that issued the ticket)

########### Here you would have noticed that the dataframe has 'Violating Precinct' or 'Issuing Precinct' as '0'. These are the erroneous entries.

########### Hence, provide the record for five correct precincts. (Hint: Print top six entries after sorting)

############ 3.1 'Violation Precinct' (this is the precinct of the zone where the violation occurred). ###################

############ Using this, can you make any insights for parking violations in any specific areas of the city? ############

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_SPK\_DF\_2017\_VP\_NOT\_ZERO <- filter(Parking\_Tickets\_SPK\_DF\_2017, Parking\_Tickets\_SPK\_DF\_2017$Violation\_Precinct != 0)

Parking\_Tickets\_SPK\_DF\_2017\_VP <- summarize(groupBy(Parking\_Tickets\_SPK\_DF\_2017\_VP\_NOT\_ZERO,Parking\_Tickets\_SPK\_DF\_2017\_VP\_NOT\_ZERO$Violation\_Precinct),

Number\_of\_Tickets=n(Parking\_Tickets\_SPK\_DF\_2017\_VP\_NOT\_ZERO$Violation\_Precinct))

Parking\_Tickets\_SPK\_DF\_2017\_VP\_count <- arrange(Parking\_Tickets\_SPK\_DF\_2017\_VP, desc(Parking\_Tickets\_SPK\_DF\_2017\_VP$Number\_of\_Tickets))

head(Parking\_Tickets\_SPK\_DF\_2017\_VP\_count,5)

# Violation\_Precinct Number\_of\_Tickets

# 1 19 274444

# 2 14 203553

# 3 1 174702

# 4 18 169131

# 5 114 147444

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# checking whether the Violation Precinct has zero..if it has those records are not to be considred

head(SparkR::sql("select Count(1) from Parking\_Tickets\_SQL\_tbl\_2017 where Violation\_Precinct=0"))

# 925595--> There are 925595 records with Violation\_Precinct=0 and hence will filter out this and then lets take the parking tickets count

head(SparkR::sql("Select Count(1) as no\_of\_parking\_violations,Violation\_Precinct from Parking\_Tickets\_SQL\_tbl\_2017 where Violation\_Precinct<>0

group by Violation\_Precinct order by 1 desc limit 5"))

# no\_of\_parking\_violations Violation\_Precinct

# 1 274444 19

# 2 203553 14

# 3 174702 1

# 4 169131 18

# 5 147444 114

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Violation\_Precinct vs. Number of Tickets) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

printSchema(Parking\_Tickets\_SPK\_DF\_2017\_VP\_count)

Parking\_Tickets\_SPK\_DF\_2017\_VP\_count$Violation\_Precinct <- cast(Parking\_Tickets\_SPK\_DF\_2017\_VP\_count$Violation\_Precinct,'string')

Parking\_Tickets\_SPK\_R\_2017\_VP\_count <- as.data.frame(head(select(Parking\_Tickets\_SPK\_DF\_2017\_VP\_count,

list(Parking\_Tickets\_SPK\_DF\_2017\_VP\_count$Violation\_Precinct,

Parking\_Tickets\_SPK\_DF\_2017\_VP\_count$Number\_of\_Tickets)),5))

ggplot(Parking\_Tickets\_SPK\_R\_2017\_VP\_count,aes(x=Violation\_Precinct, y=Number\_of\_Tickets)) +geom\_col() + xlab("Violation\_Precinct") + ylab("Number of Tickets") +

ggtitle("Plot5. Violation\_Precinct vs. Number of Tickets") + geom\_text(aes(label=Number\_of\_Tickets),vjust=-0.3)

# 3.2 'Issuer Precinct' (this is the precinct that issued the ticket)

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_SPK\_DF\_2017\_IP\_NOT\_ZERO <- filter(Parking\_Tickets\_SPK\_DF\_2017, Parking\_Tickets\_SPK\_DF\_2017$Issuer\_Precinct != 0)

Parking\_Tickets\_SPK\_DF\_2017\_IP <- summarize(groupBy(Parking\_Tickets\_SPK\_DF\_2017\_IP\_NOT\_ZERO,Parking\_Tickets\_SPK\_DF\_2017\_IP\_NOT\_ZERO$Issuer\_Precinct),

Number\_of\_Tickets=n(Parking\_Tickets\_SPK\_DF\_2017\_IP\_NOT\_ZERO$Issuer\_Precinct))

Parking\_Tickets\_SPK\_DF\_2017\_IP\_count <- arrange(Parking\_Tickets\_SPK\_DF\_2017\_IP, desc(Parking\_Tickets\_SPK\_DF\_2017\_IP$Number\_of\_Tickets))

head(Parking\_Tickets\_SPK\_DF\_2017\_IP\_count,5)

# Issuer\_Precinct Number\_of\_Tickets

# 1 19 266961

# 2 14 200495

# 3 1 168740

# 4 18 162994

# 5 114 144054

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# checking whether the Issuer Precinct has zero..if it has those records are not to be considred

head(SparkR::sql("select Count(1) from Parking\_Tickets\_SQL\_tbl\_2017 where Issuer\_Precinct=0"))

# 1078404--> There are 1078404 records with Issuer\_Precinct=0 and hence will filter out this and then lets take the parking tickets count

head(SparkR::sql("Select Count(1) as no\_of\_parking\_violations,Issuer\_Precinct from Parking\_Tickets\_SQL\_tbl\_2017 where Issuer\_Precinct<>0

group by Issuer\_Precinct order by 1 desc limit 5"))

# no\_of\_parking\_violations Issuer\_Precinct

# 1 266961 19

# 2 200495 14

# 3 168740 1

# 4 162994 18

# 5 144054 114

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Violation\_Precinct vs. Number of Tickets) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

printSchema(Parking\_Tickets\_SPK\_DF\_2017\_IP\_count)

Parking\_Tickets\_SPK\_DF\_2017\_IP\_count$Issuer\_Precinct <- cast(Parking\_Tickets\_SPK\_DF\_2017\_IP\_count$Issuer\_Precinct,'string')

Parking\_Tickets\_SPK\_R\_2017\_IP\_count <- as.data.frame(head(select(Parking\_Tickets\_SPK\_DF\_2017\_IP\_count,

list(Parking\_Tickets\_SPK\_DF\_2017\_IP\_count$Issuer\_Precinct,

Parking\_Tickets\_SPK\_DF\_2017\_IP\_count$Number\_of\_Tickets)),5))

ggplot(Parking\_Tickets\_SPK\_R\_2017\_IP\_count,aes(x=Issuer\_Precinct, y=Number\_of\_Tickets)) +geom\_col() + xlab("Issuer\_Precinct") + ylab("Number of Tickets") +

ggtitle("Plot6. Issuer\_Precinct vs. Number of Tickets") + geom\_text(aes(label=Number\_of\_Tickets),vjust=-0.3)

############## 4. Find the violation code frequency across three precincts which have issued the most number of tickets - do these precinct zones

############## have an exceptionally high frequency of certain violation codes? Are these codes common across precincts?

############## Hint: In the SQL view, use the 'where' attribute to filter among three precincts.

############## 4.1 Find the violation code frequency across three precincts With Respect to Issuer\_Precinct

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("SELECT count(1) as check\_wrt\_Issuer\_Precinct,a.Violation\_Code from Parking\_Tickets\_SQL\_tbl\_2017 as a

join (select count(1) as Parking\_violation\_count,Issuer\_Precinct from Parking\_Tickets\_SQL\_tbl\_2017

where Issuer\_Precinct<>0 group by Issuer\_Precinct order by 1 desc limit 3)b on a.Issuer\_Precinct=b.Issuer\_Precinct

group by a.Violation\_Code order by 1 desc "))

# check\_wrt\_Issuer\_Precinct Violation\_Code

# 1 113187 14

# 2 68869 46

# 3 48190 38

# 4 43782 37

# 5 39046 69

# 6 33499 21

############## 4.2 Find the violation code frequency across three precincts With Respect to Violation\_Precinct

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("SELECT count(1) as check\_wrt\_Violation\_Precinct,a.Violation\_Code from Parking\_Tickets\_SQL\_tbl\_2017 as a

join (select count(1) as Parking\_violation\_count,Violation\_Precinct from Parking\_Tickets\_SQL\_tbl\_2017

where Violation\_Precinct<>0 group by Violation\_Precinct order by 1 desc limit 3)b on a.Violation\_Precinct=b.Violation\_Precinct

group by a.Violation\_Code order by 1 desc "))

# check\_wrt\_Violation\_Precinct Violation\_Code

# 1 116487 14

# 2 72730 46

# 3 49364 38

# 4 44219 37

# 5 39057 69

# 6 35472 21

############## 4.3 Find the violation code frequency across three precincts which have issued the most number of tickets

# 'Issuer Precinct' (this is the precinct that issued the ticket), so calculating the no\_of tickets based on this

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("SELECT count(1) as check\_wrt\_Issuer\_Precinct,a.Violation\_Code,b.Issuer\_Precinct from Parking\_Tickets\_SQL\_tbl\_2017 as a

join (select count(1) as Parking\_violation\_count,Issuer\_Precinct from Parking\_Tickets\_SQL\_tbl\_2017

where Issuer\_Precinct<>0 group by Issuer\_Precinct order by 1 desc limit 3)b on a.Issuer\_Precinct=b.Issuer\_Precinct

group by a.Violation\_Code,b.Issuer\_Precinct order by 1 desc "),20)

# check\_wrt\_Issuer\_Precinct Violation\_Code Issuer\_Precinct

# 1 48445 46 19

# 2 45036 14 14

# 3 38354 14 1

# 4 36386 38 19

# 5 36056 37 19

# 6 30464 69 147

# 7 29797 14 19

# 8 28415 21 19

# 9 22555 31 14

# 10 19081 16 1

# 11 18364 47 14

# 12 15408 20 1

# 13 14629 20 19

# 14 12745 46 1

# 15 11416 40 19

# 16 10027 42 14

# 17 9926 16 19

# 18 8535 38 1

# 19 7679 46 14

# 20 7526 17 1

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Issuer\_Precinct vs. Number of Tickets [With respect to top 3 violation code frequency]) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_R\_DF\_IP\_top3\_precincts <- as.data.frame(SparkR::sql("SELECT count(1) as check\_wrt\_Issuer\_Precinct,a.Violation\_Code,b.Issuer\_Precinct from Parking\_Tickets\_SQL\_tbl\_2017 as a

join (select count(1) as Parking\_violation\_count,Issuer\_Precinct from Parking\_Tickets\_SQL\_tbl\_2017

where Issuer\_Precinct<>0 group by Issuer\_Precinct order by 1 desc limit 3)b on a.Issuer\_Precinct=b.Issuer\_Precinct

group by a.Violation\_Code,b.Issuer\_Precinct order by 1 desc "))

Parking\_Tickets\_R\_DF\_IP\_top3\_precincts$Issuer\_Precinct <- as.character(Parking\_Tickets\_R\_DF\_IP\_top3\_precincts$Issuer\_Precinct)

Parking\_Tickets\_R\_DF\_IP\_top3\_precincts$Violation\_Code <- as.character(Parking\_Tickets\_R\_DF\_IP\_top3\_precincts$Violation\_Code)

str(Parking\_Tickets\_R\_DF\_IP\_top3\_precincts)

ggplot(Parking\_Tickets\_R\_DF\_IP\_top3\_precincts[1:20, ])+aes(x=Issuer\_Precinct,y=check\_wrt\_Issuer\_Precinct, fill=factor(Violation\_Code)) +geom\_col()+

ggtitle("Plot7. Issuer\_Precinct vs. Number of Tickets\_WRT\_TOP3\_precincts\_with\_Violation\_Code\_Frequency") + geom\_text(aes(label=check\_wrt\_Issuer\_Precinct),vjust=-0.3)

############### 5. You’d want to find out the properties of parking violations across different times of the day:

############### Find a way to deal with missing values, if any.

############### Hint: Check for the null values using 'isNull' under the SQL. Also, to remove the null values, check the 'dropna' command in the API documentation.

###############

############### 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.

###############

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

############### Hint: Use the CASE-WHEN in SQL view to segregate into bins. For finding the most commonly occurring violations, a similar approach can be used as mention in the hint for question 4.

###############

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

############### 5.1 Find a way to deal with missing values, if any

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SparkR \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

nrow(filter(Parking\_Tickets\_SPK\_DF\_2017, Parking\_Tickets\_SPK\_DF\_2017$Violation\_Time=="" ))

# 0 --> no rows with Null Violation\_Time

# From the above, concludes that no Records exists with Null values but however testing with dropna

nrow(dropna(Parking\_Tickets\_SPK\_DF\_2017, how=c("any","all"), minNonNulls = NULL, cols = "Violation\_Time"))

# 5431909 --> RETURNED ALL ROWS, WHICH INDICATES THAT NO ROWS HAS BEEN DROPPED AS IT DOESN'T HAVE ANY NULL VALUES

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select count(1) from Parking\_Tickets\_SQL\_tbl\_2017 where Violation\_Time is null"))

# 0 --> no rows with Null Violation\_Time

############### 5.2 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.

############### Divide 24 hours into 6 equal discrete bins of time. The intervals you choose are at your discretion.

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins <- SparkR::sql("select \*,

(case when substr(Violation\_Time,1,2) in('00','01','02','03','12') and substr(Violation\_Time,5,1)='A' Then '0-3'

when substr(Violation\_Time,1,2) in('04','05','06','07') and substr(Violation\_Time,5,1)='A' Then '4-7'

when substr(Violation\_Time,1,2) in('08','09','10','11') and substr(Violation\_Time,5,1)='A' Then '8-11'

when substr(Violation\_Time,1,2) in('12','01','02','03') and substr(Violation\_Time,5,1)='P' Then '12-15'

when substr(Violation\_Time,1,2) in('04','05','06','07') and substr(Violation\_Time,5,1)='P' Then '16-19'

when substr(Violation\_Time,1,2) in('08','09','10','11') and substr(Violation\_Time,5,1)='P' Then '20-23' end) as Time\_Bin

from Parking\_Tickets\_SQL\_tbl\_2017")

############### 5.3 For each of these groups, find the three most commonly occurring violations

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

createOrReplaceTempView(Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins,"Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL")

head(SparkR::sql("select Violation\_Code,Time\_Bin,count(1)as ticket\_freq from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL group by Violation\_Code,Time\_Bin "))

head(SparkR::sql("select \*, dense\_rank() over(partition by Time\_Bin order by ticket\_freq desc) as rnk from(select Violation\_Code,Time\_Bin,count(1)as ticket\_freq from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL group by Violation\_Code,Time\_Bin) "))

head(SparkR::sql("select Violation\_Code,Time\_Bin,ticket\_freq from (select \*, ROW\_NUMBER() over(partition by Time\_Bin order by ticket\_freq desc) as rnk from(select Violation\_Code,Time\_Bin,count(1)as ticket\_freq from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL group by Violation\_Code,Time\_Bin))

where rnk < 4"),24)

# Violation\_Code Time\_Bin ticket\_freq

# 1 36 12-15 286284

# 2 38 12-15 240721

# 3 37 12-15 167025

# 4 21 8-11 598067

# 5 36 8-11 348165

# 6 38 8-11 176570

# 7 46 <NA> 39

# 8 21 <NA> 28

# 9 40 <NA> 22

# 10 38 16-19 102855

# 11 14 16-19 75902

# 12 37 16-19 70345

# 13 21 0-3 36957

# 14 40 0-3 25866

# 15 78 0-3 15528

# 16 14 4-7 74114

# 17 40 4-7 60652

# 18 21 4-7 57897

# 19 7 20-23 26293

# 20 40 20-23 22337

# 21 14 20-23 21045

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Violation\_Code vs. Number of Tickets [With respect to top 3 Time\_Bin]) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_R\_DF\_VC\_top3\_TB <- as.data.frame((SparkR::sql("select Violation\_Code,Time\_Bin,ticket\_freq from (select \*, ROW\_NUMBER() over(partition by Time\_Bin order by ticket\_freq desc) as rnk from(select Violation\_Code,Time\_Bin,count(1)as ticket\_freq from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL group by Violation\_Code,Time\_Bin))

where rnk < 4")))

Parking\_Tickets\_R\_DF\_VC\_top3\_TB$Violation\_Code <- as.character(Parking\_Tickets\_R\_DF\_VC\_top3\_TB$Violation\_Code)

str(Parking\_Tickets\_R\_DF\_TB\_top3\_VC)

ggplot(Parking\_Tickets\_R\_DF\_VC\_top3\_TB)+aes(x=Violation\_Code,y=ticket\_freq, fill=factor(Time\_Bin)) +geom\_col()+

ggtitle("Plot8. Violation\_Code vs. Number of Tickets [With respect to Time\_Bin]") + geom\_text(aes(label=ticket\_freq),vjust=-0.3)

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

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select Violation\_Code,count(1)as ticket\_freq\_wrt\_VC from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL group by Violation\_Code order by 2 desc LIMIT 3"))

# Violation\_Code ticket\_freq\_wrt\_VC

# 1 21 768085

# 2 36 662765

# 3 38 542079

# head(SparkR::sql("select inner\_query.Violation\_Code as Violation\_Code,Time\_Bin,count(1) as tkt\_count\_wrt\_top\_vc from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL outer\_query

# join (select Violation\_Code,count(1) as count from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL

# group by Violation\_Code order by count desc limit 3)inner\_query on outer\_query.Violation\_Code=inner\_query.Violation\_Code

# group by outer\_query.Violation\_Code,outer\_query.Time\_Bin

# order by tkt\_count\_wrt\_top\_vc desc"),10)

head(SparkR::sql(" SELECT sub.Violation\_Code as Violation\_Code, Time\_Bin, count(1) as total\_tickets from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL main

join (SELECT Violation\_Code, count(1) as count from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL

group by Violation\_Code order by count desc limit 3)sub on main.Violation\_Code = sub.Violation\_Code

group by sub.Violation\_Code, main.Time\_Bin

order by total\_tickets DESC,Violation\_Code ASC"),4)

# Violation\_Code Time\_Bin total\_tickets

# 1 21 8-11 598067

# 2 36 8-11 348165

# 3 36 12-15 286284

# 4 38 12-15 240721

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Time\_Bin vs. Number of Tickets [With respect to top 3 Violation\_Code]) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_R\_DF\_TB\_top3\_VC <- as.data.frame((SparkR::sql("SELECT sub.Violation\_Code as Violation\_Code, Time\_Bin, count(1) as total\_tickets from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL main

join (SELECT Violation\_Code, count(1) as count from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL

group by Violation\_Code order by count desc limit 3)sub on main.Violation\_Code = sub.Violation\_Code

group by sub.Violation\_Code, main.Time\_Bin

order by total\_tickets DESC,Violation\_Code ASC")))

Parking\_Tickets\_R\_DF\_TB\_top3\_VC$Violation\_Code <- as.character(Parking\_Tickets\_R\_DF\_TB\_top3\_VC$Violation\_Code)

str(Parking\_Tickets\_R\_DF\_TB\_top3\_VC)

ggplot(Parking\_Tickets\_R\_DF\_TB\_top3\_VC[1:4, ])+aes(x=Violation\_Code,y=total\_tickets, fill=factor(Time\_Bin)) +geom\_col()+

ggtitle("Plot9. Time\_Bin vs. Number of Tickets [With respect to top 3 Violation\_Code]") + geom\_text(aes(label=total\_tickets),vjust=-0.3)

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

############### First, divide the year into some number of seasons, and find frequencies of tickets for each season. (Hint: Use Issue Date to segregate into seasons)

############### Then, find the three most common violations for each of these seasons.

############### (Hint: A similar approach can be used as mention in the hint for question 4.)

############### 6.1 divide the year into some number of seasons, and find frequencies of tickets for each season. (Hint: Use Issue Date to segregate into seasons)

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Seasons <- SparkR::sql("select \*,(case when substr(Issue\_Date,6,2) in ('06','07','08') Then 'Summer'

when substr(Issue\_Date,6,2) in ('09','10','11') Then 'Autumn'

when substr(Issue\_Date,6,2) in ('12','01','02') Then 'Winter'

when substr(Issue\_Date,6,2) in ('03','04','05') Then 'Spring' end) season from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL")

createOrReplaceTempView(Seasons,"Seasons\_sql")

head(SparkR::sql("select season,count(1) no\_of\_tickets from Seasons\_sql group by season order by 2 desc"))

# season no\_of\_tickets

# 1 Spring 2873383

# 2 Winter 1704681

# 3 Summer 852866

# 4 Autumn 979

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Seasons vs. Number of Tickets) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_R\_DF\_SEASONS <- as.data.frame((head(SparkR::sql("select season,count(1) no\_of\_tickets from Seasons\_sql group by season order by 2 desc"))))

ggplot(Parking\_Tickets\_R\_DF\_SEASONS)+aes(x=season,y=no\_of\_tickets) +geom\_col()+

ggtitle("Plot10.Seasons vs. Number of Tickets") + geom\_text(aes(label=no\_of\_tickets),vjust=-0.3)

############### 6.2 find the three most common violations for each of these seasons

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select Violation\_Code,count(1)as ticket\_freq\_wrt\_VC from Seasons\_sql group by Violation\_Code order by 2 desc LIMIT 3"))

# Violation\_Code ticket\_freq\_wrt\_VC

# 1 21 768085

# 2 36 662765

# 3 38 542079

# head(SparkR::sql(" SELECT sub.Violation\_Code as Violation\_Code, Time\_Bin, count(1) as count from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL main

# join (SELECT Violation\_Code, count(1) as count from Parking\_Tickets\_tbl\_2017\_wth\_hr\_bins\_SQL

# group by Violation\_Code order by count desc limit 3)sub on main.Violation\_Code = sub.Violation\_Code

# group by sub.Violation\_Code, main.Time\_Bin

# order by count desc"),15)

head(SparkR::sql(" SELECT sub.Violation\_Code as Violation\_Code, season, count(1) as count from Seasons\_sql main

join (SELECT Violation\_Code, count(1) as count from Seasons\_sql

group by Violation\_Code order by count desc limit 3)sub on main.Violation\_Code = sub.Violation\_Code

group by sub.Violation\_Code, main.season

order by count desc"),15)

# Violation\_Code season count

# 1 21 Spring 402424

# 2 36 Spring 344834

# 3 38 Spring 271167

# 4 21 Winter 238181

# 5 36 Winter 221268

# 6 38 Winter 187386

# 7 21 Summer 127352

# 8 36 Summer 96663

# 9 38 Summer 83518

# 10 21 Autumn 128

# 11 38 Autumn 8

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GRAPH(Seasons vs. Number of Tickets[With respect to most common Violation codes]) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Parking\_Tickets\_R\_DF\_SEASONS\_top3\_vc <- as.data.frame((SparkR::sql("SELECT sub.Violation\_Code as Violation\_Code, season, count(1) as no\_of\_tickets from Seasons\_sql main

join (SELECT Violation\_Code, count(1) as count from Seasons\_sql

group by Violation\_Code order by count desc limit 3)sub on main.Violation\_Code = sub.Violation\_Code

group by sub.Violation\_Code, main.season

order by no\_of\_tickets desc")))

ggplot(Parking\_Tickets\_R\_DF\_SEASONS\_top3\_vc)+aes(x=season,y=no\_of\_tickets, fill=factor(Violation\_Code)) +geom\_col()+

ggtitle("Plot11. Seasons vs. Number of Tickets[With respect to most common Violation codes]") + geom\_text(aes(label=no\_of\_tickets),vjust=-0.3)

############### 7. The fines collected from all the parking violation constitute a revenue source for the NYC police department.

############### Let’s take an example of estimating that for the three most commonly occurring codes

############### 7.1 Find total occurrences of the three most common violation codes

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql("select Violation\_Code,count(1)as ticket\_freq\_wrt\_VC from Seasons\_sql group by Violation\_Code order by 2 desc LIMIT 3"))

# Violation\_Code ticket\_freq\_wrt\_VC

# 1 21 768085

# 2 36 662765

# 3 38 542079

############### 7.2 lists the fines associated with different violation codes. They’re divided into two categories,

############### one for the highest-density locations of the city, the other for the rest of the city.

############### For simplicity, take an average of the two.

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Avg\_fines\_SDF <- SparkR::sql("select \*, (case when Violation\_Code=21 then ((65+45)/2)

when Violation\_Code=36 then ((50+50)/2)

when Violation\_Code=38 then ((65+35)/2) else null end) as Avg\_fines from Seasons\_sql")

createOrReplaceTempView(Avg\_fines\_SDF,"Avg\_fines\_SQL")

head(SparkR::sql("select distinct Violation\_Code,Avg\_fines from Avg\_fines\_SQL where Violation\_Code in(21,36,38)"))

# Violation\_Code Avg\_fines

# 1 21 55

# 2 38 50

# 3 36 50

############### 7.3 Using this information, find the total amount collected for the three violation codes with maximum tickets.

############### State the code which has the highest total collection

# /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Through SQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

head(SparkR::sql(" SELECT distinct innerquery.Violation\_Code as Violation\_Code,

case when innerquery.Violation\_Code=21 then Avg\_fines \* innerquery.count

when innerquery.Violation\_Code=36 then Avg\_fines \* innerquery.count

when innerquery.Violation\_Code=38 then Avg\_fines \* innerquery.count

else null end total\_amt\_collected

from Avg\_fines\_SQL outerquery

join (SELECT Violation\_Code, count(1) as count from Avg\_fines\_SQL

group by Violation\_Code order by count desc limit 3)innerquery on outerquery.Violation\_Code = innerquery.Violation\_Code order by 2 desc"))

# Violation\_Code total\_amt\_collected

# 1 21 42244675

# 2 36 33138250

# 3 38 27103950

sparkR.stop()