NYC Parking Tickets: An Exploratory Analysis

BACKGROUND and OBJECTIVE

New York City is a thriving metropolis. Just like most other metros that size, one of the biggest problems its citizens face is parking. The classic combination of a huge number of cars and a cramped geography is the exact recipe that leads to a huge number of parking tickets.

The main aim of the case study is to analyse the parking ticket data based on years 2015,2016 and 2017 and derive insights based on the same.

PROCEDURE

The following steps are involved in this case study:

- Load and initialise SparkR.
- Load the required data files.
- Clean the data.
- Perform Exploratory Data Analysis (EDA).
- Analyse and derive relevant insights and conclusions.

ASSUMPTIONS USED

The following assumptions are used in this case study.

- 'Violation Location' is considered for analysis and examining the data.
- Violation and Issue Precinct numbers 0 are not considered for analysis.
- NA and Null values are ignored during analysis.

NOTE
Relevant insights are mentioned in bold.

Loading SparkR and setting up the environment for analysis

```
#loading 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 the spark session

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

#reading data for years 2015, 2016 and 2017 by loading the three required csv files

nyc_15 <- read.df("hdfs:///common_folder/nyc_parking/Parking_Violations_Issued_-_Fiscal_Year_2015.csv", source = "csv", header = TRUE, inferSchema = TRUE)

nyc_16 <- read.df("hdfs:///common_folder/nyc_parking/Parking_Violations_Issued_-_Fiscal_Year_2016.csv", source = "csv", header = TRUE, inferSchema = TRUE)

nyc_17 <- read.df("hdfs:///common_folder/nyc_parking/Parking_Violations_Issued_-_Fiscal_Year_2017.csv", source = "csv", header = TRUE, inferSchema = TRUE)

Analysing and deriving insights for the year 2015

#------ Year 2015 --------

#Check the spark dataframe

head(nyc_15)

str(nyc_15)

#Examine the size

nrow(nyc_15)

#11809233

ncol(nyc_15)

#51

#Examine the dataframe schema

printSchema(nyc_15)

#Examining the Data

#1. Find the total number of tickets

collect(select(nyc_15,count(nyc_15$'Summons Number')))

#11809233

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

```
collect(select(nyc_15,countDistinct(nyc_15$'Registration State')))

#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")

#Creating the temporary view for SQL query analysis

createOrReplaceTempView(nyc_15, "nyc_15_view")

#replacing numeric value with state having maximum records

State <- SparkR::sql("select 'Registration State' from nyc_15_view group by 'Registration State' order by count(*) desc limit 1")

nyc_15$'Registration State' <- ifelse(nyc_15$'Registration State'=="99","CA",nyc_15$'Registration State')

#run the collect function again

collect(select(nyc_15,countDistinct(nyc_15$'Registration State')))

#68

#3. Check the number of tickets that don’t have the address for violation location on them

NullLocation <- filter(nyc_15,isNull(nyc_15$'Violation Location'))

count(NullLocation)

#1799170

#Aggregation tasks and deriving insights

#1. Check the top 5 violation code freq

v_count <- summarize(groupBy(nyc_15, nyc_15$`Violation Code`), count = n(nyc_15$`Violation Code`))

View(head(arrange(v_count, desc(v_count$count))))

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

#Violation Code   count
```

#21 1630912

#38 1418627

#14 988469

#36 839197

#37 795918

#7 719753

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

#The top 5 violation codes are 21,38,14,36 and 37

#2. Check the top 5 vehicle body type get parking tickets

v_body <- summarize(groupBy(nyc_15, nyc_15$`Vehicle Body Type`), count = n(nyc_15$`Summons Number`))

View(head(arrange(v_body, desc(v_body$count))))

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

#Vehicle Body Type count

#SUBN 3729346

#4DSD 3340014

#VAN 1709091

#DELV 892781

#SDN 524596

#2DSD 319046

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

 # the top 5 vehicle body type that got parking tickets are SUBN,4DSD,VAN,DELV and SDN

#3. Find the top 5 vehicle body make can get parking tickets

v_make <- summarize(groupBy(nyc_15, nyc_15$`Vehicle Make`), count = n(nyc_15$`Summons Number`))

View(head(arrange(v_make, desc(v_make$count))))

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

#Vehicle Make count

#FORD 1521874

#TOYOT 1217087

#HONDA 1102614

#NISSA 908783

#CHEVR 897845

#FRUEH 432073

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

#the top 5 vehicle body make that got parking tickets are FORD,TOYOT,HONDA,NISSA andCHEVR

#3(a). Violation precinct

```
non_zero_preinct <- filter(nyc_15, nyc_15$`Violation Precinct` > 0)

Vio_preinct <- summarize(groupBy(non_zero_preinct, non_zero_preinct$`Violation Precinct`),

count = n(non_zero_preinct$`Violation Precinct`))

View(head(arrange(Vio_preinct, desc(Vio_preinct$count))))
```

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

#Violation Precinct count

#19 598351

#18 427510

#14 409064

#1 329009

#114 320963

#13 305250

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

#3(b). Issuer Precinct

```
non_zero_issuer <- filter(nyc_15, nyc_15$`Issuer Precinct` > 0)
```

```
iss_preinct <- summarize(groupBy(non_zero_issuer, non_zero_issuer$`Issuer Precinct`),
count = n(non_zero_issuer$`Issuer Precinct`))

View(head(arrange(iss_preinct, desc(iss_preinct$count))))
```

```
########################
#Issuer Precinct count
#19                     579998
#18                     417329
#14                     392922
#1                      318778
#114                    314437
#13                     296403
########################
```

#The top 5 violation and issuer precincts are 19,18,14,1 and 114

```
#4. Violation & Issuer Precinct
non_zero_iv_preinct <- filter(nyc_15, nyc_15$`Violation Precinct` > 0 | nyc_15$`Issuer Precinct` > 0)
iss_preinct <- summarize(groupBy(non_zero_issuer, non_zero_issuer$`Issuer Precinct`), count =
n(non_zero_issuer$`Issuer Precinct`))
View(head(arrange(iss_preinct, desc(iss_preinct$count))))
```

```
########################
#Violation Precinct     Issuer Precinct  count
#19                     19               579491
#18                     18               416832
#14                     14               392081
#1                      1                314582
#114                    114              314163
#13                     13               296100
########################
```

#The top 5 violation and issuer precincts are 19,18,14,1 and 114

```
#5. drop all na values from the dataset

nyc_15 <- dropna(nyc_15, how="all")

#5. drop all na values from the dataset

nyc_15 <- dropna(nyc_15, how="all")


#remove erroneous date formats

createOrReplaceTempView(nyc_15, "nyc_15_view")

nyc15 <- SparkR::sql("select `Violation Time`, Violation Code` from nyc_15_view")


nyc2015 <- mutate(nyc16, Hour = substr(nyc15$Time,1, 2), Min = substr(nyc15$Time, 3,4), Half = substr(nyc15$Time, 5,5))


nyc20151 <- transform(nyc2015, Hour = ifelse(nyc2015$Half == 'P' & nyc2015$Hour < 12, nyc2015$Hour+12, nyc2015$Hour))

nyc15 <- transform(nyc20151, Hour = cast(nyc20151$Hour, "integer"), Min = cast(nyc20151$Min, "integer"))


nrow(filter(nyc15, nyc15$Hour > 23))



Hour2015 <- summarize(groupBy(nyc15, nyc15$Hour), count = n(nyc15$Hour))


#Divide 24 hours into six equal discrete bins of time.

createOrReplaceTempView(nyc_15, "nyc_15_view")


bin2015 <- SparkR::sql("select nyc_15_view.Time, nyc_15_view.Code, nyc_15_view.Hour, nyc_15_view.Min, nyc_15_view.Half,

                       CASE WHEN Hour > 23 THEN 'Invalid'

                       WHEN Hour < 4 THEN '0-3'

                       WHEN Hour >=4 AND Hour <8 THEN '4-7'

                       WHEN Hour >=8 and Hour <12 THEN '8-11'
```

```
                WHEN Hour >=12 and Hour <16 THEN '12-15'
                WHEN Hour >=16 and Hour <20 THEN '16-19'
                ELSE '20-23' END as Bin from nyc_15_view")


head(bin2015)
Binned2015 <- summarize(groupBy(bin2015, bin2015$Bin, bin2015$Code), count=n(bin2015$Code))

#find out the top 3 violation codes
top3ViolationCodes <- filter(bin2015, bin2015$Code %in% c(21,36,38))

ViolationCodeSummary <- summarize(groupBy(top3ViolationCodes, top3ViolationCodes$Code,
top3ViolationCodes$Bin), count=n(top3ViolationCodes$Bin))

createOrReplaceTempView(Summary, "SummaryView")

top3VioCount <- SparkR::sql("select bin, code, count from (select bin, code, count, dense_rank() over
                (partition by code order by count desc) as rank from "SummaryView") tmp where
rank <=1 order by code")

head(top3VioCount)
#most common violation codes are 14,21,38
##########################
##bin code   count
#1  8-11   14  317009
#2  8-11   21 1291540
#3 12-15   38  609616
##########################
#6. Finding seasonality
createOrReplaceTempView(nyc_15, "nyc_15_view")
nyc2015 <- SparkR::sql("select `Violation Code`, `Issue Date` from "nyc_15_view")
```

```
#convert data and time into appropriate format

nyc2015 <- mutate(nyc2015 , Date = to_date(nyc2015$`Issue Date`, 'MM/dd/yyyy'))

nyc2015 <- mutate(nyc2015 , Week = weekofyear(nyc2015$Date))

#removing irrelevant dates

nrow(filter(nyc2015, "Date < '2015-01-01'"))

createOrReplaceTempView(nyc2015, "nyc_15_view")


Season <- SparkR::sql("select nyc_15_view.code, nyc_15_view.NewDate, CASE WHEN Week <=13
THEN 'Spring'

                WHEN Week > 13 AND Week < 27 THEN 'Summer'

                WHEN Week >=27 AND Week < 40 THEN 'Autumn'

                ELSE 'Winter' END as Seasons from nyc_15_view")



Tickets <- summarize(groupBy(Season, Season$Seasons), count=n(Season$code))

head(Tickets)

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

# Quarters   count

# Summer    3236607

# Spring    3135405

# Autumn    2935987

# Winter    2501234

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

#summer season has the most occuring violations


ViolationsPerSeason <- summarize(groupBy(Season, Season$Seasons, Season$code),
count=n(Season$code))

head(ViolationsPerSeason)


createOrReplaceTempView(ViolationsPerSeason, "ViolationPerSeasonView")
```

```
top3ViolationPerSeason <- SparkR::sql("select Quarters, code, count from (select Quarters, code,
count, dense_rank() over
                    (partition by Quarters order by count desc) as rank from
ViolationPerSeasonView) tmp where rank <=3")


#4 quarters/seasons , top3 in each


head(top3ViolationPerSeason, 12)
```

#the three most common violation codes are 21, 38, 14

```
#7. Find total occurrences of the three most common violation codes


parking_2015_viol_codes_counts <- summarize(groupBy(nyc_15, parking_2015$Vio1ation_Code),
count = n(parking_2015$Violation_Code))


head(arrange(parking_2015_viol_codes_counts, desc(parking_2015_viol_codes_counts$count)))
########################
#Violation_Code count
#1 21 1630912
#2 38 1418627
#3 14 988469
#4 36 839197
#5 37 795918
#6 7 719753
########################
```

#the three most common violation codes are 21, 38, 14

```
#According to https://www1.nyc.gov/site/finance/vehicles/services-violation-codes.page
```

#Fine amount for violation code 21 is (65+45)/2 = 55

#Fine amount for violation code 38 is (65+35)/2 = 50

#Fine amount for violation code 14 is (115+115)/2 = 115

#find the total amount collected for the three violation codes with maximum tickets.

#State the code which has the highest total collection.

createOrReplaceTempView(parking_2015_viol_codes_counts, "parking_2015_violation_code“)

pr_2015_viol_codes_counts <- sq1("SELECT Violation_Code, \

CASE WHEN Violation Code = 21 THEN count * 55 \

WHEN Violation_Code 38 THEN count * 50 \

WHEN Violation_Code 14 THEN count * 115 \

ELSE 0 END as revenue FROM parking_2915_violation_code \

WHERE Violation_Code IN (21, 38, 14)")

head(arrange(pr_2015_vi01_codes_counts, desc(pr_2015_viol_codes_counts$revenue)))

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

Violation_Code revenue

1 14 113673935

2 21 89700160

3 38 70931350

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

#Hence violation 14 has highest total collection.

Analysing and deriving insights for the year 2016

#------ Year 16 --------

#Check the spark dataframe

head(nyc_16)

str(nyc_16)

```
#Examine the size
nrow(nyc_16)
#10626899
ncol(nyc_16)
#51

#Examine the dataframe schema
printSchema(nyc_16)

#Examining the Data
#1. Find the total number of tickets
collect(select(nyc_16,count(nyc_16$'Summons Number')))
#10626899

#2. Find out the number of unique states from where the cars that got parking tickets came from.
collect(select(nyc_16,countDistinct(nyc_16$'Registration State')))
#68

#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")

#Creating the temporary view for SQL query analysis
createOrReplaceTempView(nyc_16, "nyc_16_view")

#replacing numeric value with state having maximum records
State <- SparkR::sql("select 'Registration State' from nyc_15_view group by 'Registration State' order by count(*) desc limit 1")
nyc_16$'Registration State' <- ifelse(nyc_16$'Registration State'=="99","CA",nyc_16$'Registration State')

#run the collect function again
```

```
collect(select(nyc_16,countDistinct(nyc_16$'Registration State')))

#67


#3. Check the number of tickets that don't have the address for violation location on them

NullLocation <- filter(nyc_16,isNull(nyc_16$'Violation Location'))

count(NullLocation)

#1868656


#1. Check the top 5 violation code freq

v_count <- summarize(groupBy(nyc_16, nyc_16$`Violation Code`), count = n(nyc_16$`Violation Code`))

View(head(arrange(v_count, desc(v_count$count))))


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

#Violation Code count

#21          1531587

#36          1253512

#38          1143696

#14          875614

#37          686610

#20          611013

########################
```

#The top 5 violation codes are 21,36,38,14 and 37

```
#2.Check the top 5 vehicle body type get parking tickets


v_body <- summarize(groupBy(nyc_16, nyc_16$`Vehicle Body Type`), count = n(nyc_16$`Summons Number`))

View(head(arrange(v_body, desc(v_body$count))))

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

#Vehicle Body Type    count

#SUBN                 3466037
```

#4DSD 2992107

#VAN 1518303

#DELV 755282

#SDN 424043

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

 # the top 5 vehicle body type that got parking tickets are SUBN,4DSD,VAN,DELV and SDN

#Find the top 5 vehicle body make can get parking tickets

v_make <- summarize(groupBy(nyc_16, nyc_16$`Vehicle Make`), count = n(nyc_16$`Summons Number`))

View(head(arrange(v_make, desc(v_make$count))))

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

Vehicle Make count

FORD 1324774

TOYOT 1154790

HONDA 1014074

NISSA 834833

CHEVR 759663

FRUEH 423590

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

#the top 5 vehicle body make that got parking tickets are FORD,TOYOT,HONDA,NISSA and CHEVR

#3(a). Violation precinct

non_zero_preinct <- filter(nyc_16, nyc_16$`Violation Precinct` > 0)

Vio_preinct <- summarize(groupBy(non_zero_preinct, non_zero_preinct$`Violation Precinct`),

count = n(non_zero_preinct$`Violation Precinct`))

View(head(arrange(Vio_preinct, desc(Vio_preinct$count)))

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

#Violation Precinct count

#19 554465

#18 331704

#14 324467

#1 303850

#114 291336

#13 288370

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

#3(b). Issuer Precinct

```
non_zero_issuer <- filter(nyc_16, nyc_16$`Issuer Precinct` > 0)

iss_preinct <- summarize(groupBy(non_zero_issuer, non_zero_issuer$`Issuer Precinct`),
count = n(non_zero_issuer$`Issuer Precinct`))

View(head(arrange(iss_preinct, desc(iss_preinct$count))))
```

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

#Issuer Precinct count

#19 540569

#18 323132

#14 315311

#1 295013

#114 286924

#13 282635

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

#The top 5 violation and issuer precincts are 19,18,14,1 and 114.

#4. Violation & Issuer Preinct

```
non_zero_iv_preinct <- filter(nyc_16, nyc_16$`Violation Precinct` > 0 | nyc_16$`Issuer Precinct` > 0)

iv_preinct_16 <- summarize(groupBy(non_zero_iv_preinct, non_zero_iv_preinct$`Issuer Precinct`,
non_zero_iv_preinct$`Violation Precinct`), count = n(non_zero_iv_preinct$`Violation Code`))

View(head(arrange(iv_preinct_16, desc(iv_preinct_16$count))))
```

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

#Violation Precinct	Issuer Precinct	count
#19	19	540189
#18	18	322827
#14	14	314559
#1	1	290311
#114	114	286730
#13	13	282477

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

#The top 5 violation and issuer precincts are 19,18,14,1 and 114.

#5. drop all na values from the dataset

```
nyc_16 <- dropna(nyc_16, how="all")
```

#remove erroneous date formats

```
createOrReplaceTempView(nyc_16, "nyc_16_view")

nyc16 <- SparkR::sql("select `Violation Time`, Violation Code` from nyc_16_view")

nyc2016 <- mutate(nyc16, Hour = substr(nyc16$Time,1, 2), Min = substr(nyc16$Time, 3,4), Half = substr(nyc16$Time, 5,5))

nyc20161 <- transform(nyc2016, Hour = ifelse(nyc2016$Half == 'P' & nyc2016$Hour < 12, nyc2016$Hour+12, nyc2016$Hour))

nyc16 <- transform(nyc20161, Hour = cast(nyc20161$Hour, "integer"), Min = cast(nyc20161$Min, "integer"))

nrow(filter(nyc16, nyc16$Hour > 23))

Hour2016 <- summarize(groupBy(nyc16, nyc16$Hour), count = n(nyc16$Hour))
```

#Divide 24 hours into six equal discrete bins of time.

```
createOrReplaceTempView(nyc_16, "nyc_16_view")


bin2016 <- SparkR::sql("select nyc_16_view.Time, nyc_16_view.Code, nyc_16_view.Hour,
nyc_16_view.Min, nyc_16_view.Half,
                       CASE WHEN Hour > 23 THEN 'Invalid'
                       WHEN Hour < 4 THEN '0-3'
                       WHEN Hour >=4 AND Hour <8 THEN '4-7'
                       WHEN Hour >=8 and Hour <12 THEN '8-11'
                       WHEN Hour >=12 and Hour <16 THEN '12-15'
                       WHEN Hour >=16 and Hour <20 THEN '16-19'
                       ELSE '20-23' END as Bin from nyc_16_view")



head(bin2016)
Binned2016 <- summarize(groupBy(bin2016, bin2016$Bin, bin2016$Code), count=n(bin2016$Code))


#find out the top 3 violation codes
top3ViolationCodes <- filter(bin2016, bin2016$Code %in% c(21,36,38))


ViolationCodeSummary <- summarize(groupBy(top3ViolationCodes, top3ViolationCodes$Code,
top3ViolationCodes$Bin), count=n(top3ViolationCodes$Bin))


createOrReplaceTempView(Summary, "SummaryView")


top3VioCount <- SparkR::sql("select bin, code, count from (select bin, code, count, dense_rank() over
                        (partition by code order by count desc) as rank from "SummaryView") tmp where
rank <=1 order by code")


head(top3VioCount)
#most common violation codes are 21,36,38
##########################
#bin code   count
```

```
#1  8-11   21 1209244
#2  8-11   36  586791
#3 12-15   38  488363
##########################
#6. Finding seasonality
createOrReplaceTempView(nyc_16, "nyc_16_view")
nyc2016 <- SparkR::sql("select `Violation Code`, `Issue Date` from "nyc_16_view")

#convert data and time into appropriate format
nyc2016 <- mutate(nyc2016 , Date = to_date(nyc2016$`Issue Date`, 'MM/dd/yyyy'))
nyc2016 <- mutate(nyc2016 , Week = weekofyear(nyc2016$Date))
#removing irrelevant dates
nrow(filter(nyc2016, "Date < '2016-01-01'"))
createOrReplaceTempView(nyc2016, "nyc_16_view")

Season <- SparkR::sql("select nyc_16_view.code, nyc_16_view.NewDate, CASE WHEN Week <=13
THEN 'Spring'
                    WHEN Week > 13 AND Week < 27 THEN 'Summer'
                    WHEN Week >=27 AND Week < 40 THEN 'Autumn'
                    ELSE 'Winter' END as Seasons from nyc_17_view")


Tickets <- summarize(groupBy(Season, Season$Seasons), count=n(Season$code))
head(Tickets)
##########################
# Quarters   count
# Summer  2286025
# Spring 2710934
# Autumn 2681882
# Winter 2948058
##########################
```

#summer season has the most occuring violations

```
ViolationsPerSeason <- summarize(groupBy(Season, Season$Seasons, Season$code),
count=n(Season$code))

head(ViolationsPerSeason)


createOrReplaceTempView(ViolationsPerSeason, "ViolationPerSeasonView")


top3ViolationPerSeason <- SparkR::sql("select Quarters, code, count from (select Quarters, code,
count, dense_rank() over

                        (partition by Quarters order by count desc) as rank from
ViolationPerSeasonView) tmp where rank <=3")
```

#most common violation codes are 21,36,38

```
#4 quarters/seasons , top3 in each


head(top3ViolationPerSeason, 12)

#7. Find total occurrences of the three most common violation codes


parking_2016_viol_codes_counts <- summarize(groupBy(nyc_16, parking_2016$Vio1ation_Code),

count = n(parking_2016$Violation_§ode))


head(arrange(parking_2016_viol_codes_counts, desc(parking_2016_viol_codes_counts$count)))

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

# Violation_Code count


# 1 21 1531587

# 2 36 1253512

# 3 38 1143696

# 4 14 875614

# 5 37 686610

# 6 20 611013
```

```
#######################
```

#the three most common violation codes are 21, 36,38

#Fine amount for violation code 21 is (65+45)/2 = 55

#Fine amount for violation code 36 is (50+50)/2 = 50

#Fine amount for violation code 38 is (65+35)/2 = 50

```
#find the total amount collected for the three violation codes with maximum tickets.
#State the code which has the highest total collection.
createOrReplaceTempView(parking_2016_viol_codes_counts, "parking_2016_violation_code“)

pr_2016_viol_codes_counts <- sq1("SELECT Violation_flode, \
CASE WHEN Violation Code = 21 THEN count * 55 \
WHEN Violation_Code 38 THEN count * 50 \
WHEN Violation_Code 36 THEN count * 50 \
ELSE 0 END as revenue FROM parking_2016_violation_code \
WHERE Violation_Code IN (21, 38, 36)")
head(arrange(pr_2016_viol_codes_counts, desc(pr_2016_viol_codes_counts$revenue)))
#######################
# Violation_Code revenue

# 1 21 84237285
# 2 36 62675600
# 3 38 57184800
#######################
```

#Hence violation 21 has highest total collection.

Analysing and deriving insights for the year 2017

```
#------ Year 2017 --------
#Check the spark dataframe
head(nyc2017)
```

```
str(nyc2017)

#Examine the size
nrow(nyc2017)
#10803028

ncol(nyc2017)
#43

#Examine the dataframe schema
printSchema(nyc2017)

#Examining the Data

#1. Find the total number of tickets
collect(select(nyc2017,count(nyc2017$'Summons Number')))
#10803028

collect(select(nyc2017,countDistinct(nyc2017$'Summons Number')))
#10803028

#this indicates that no duplicate Summons Number fields are present

#2. Find out the number of unique states from where the cars that got parking tickets came from.
collect(select(nyc2017,countDistinct(nyc2017$'Registration State')))

#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")

#Creating the temporary view for SQL query analysis
createOrReplaceTempView(nyc2017, "nyc2017_view")
```

```
State <- SparkR::sql("select 'Registration State' from nyc_15_view group by 'Registration State' order by count(*) desc limit 1")

#replacing numeric value with state having maximum records

nyc2017$'Registration State' <- ifelse(nyc2017$'Registration State'=="99","NY",nyc2017$'Registration State')


#run the collect function again

collect(select(nyc2017,countDistinct(nyc2017$'Registration State')))

#66


#3. Check the number of tickets that don’t have the address for violation location on them

NullLocation <- filter(nyc2017,isNull(nyc2017$'Violation Location'))

count(NullLocation)

#2072400


#------ Year 2017 --------

#Check the spark dataframe

head(nyc_17)

str(nyc_17)


#Examine the size

nrow(nyc_17)

#10803028


ncol(nyc_17)

#43


#Examine the dataframe schema

printSchema(nyc_17)


#Examining the Data
```

```
#1. Find the total number of tickets

collect(select(nyc_17,count(nyc_17$'Summons Number')))

#10803028


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

collect(select(nyc_17,countDistinct(nyc_17$'Registration State')))


#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")


#Creating the temporary view for SQL query analysis

createOrReplaceTempView(nyc_17, "nyc_17_view")


#replacing numeric value with state having maximum records

States <- SparkR::sql("select nyc_17_view.'Registration State' from nyc_17_view group by
nyc_17_view.'Registration State' order by desc limit 1")

nyc_17$'Registration State' <- ifelse(nyc_17$'Registration State'=="99","CA",nyc_17$'Registration
State')


#run the collect function again

collect(select(nyc_17,countDistinct(nyc_17$'Registration State')))

#66


#3. Check the number of tickets that don’t have the address for violation location on them

NullLocation <- filter(nyc_17,isNull(nyc_17$'Violation Location'))

count(NullLocation)

#2072400


#1. Check the top 5 violation code freq

v_count <- summarize(groupBy(nyc_17, nyc_17$`Violation Code`), count = n(nyc_17$`Violation
Code`))
```

View(head(arrange(v_count, desc(v_count$count))))

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

#Violation Code count

#21 1528588

#36 1400614

#38 1062304

#14 893498

#20 618593

#46 600012

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

#The top 5 violation codes are 21,36,38,14 and 20

#2.Check the top 5 vehicle body type get parking tickets

v_body <- summarize(groupBy(nyc_17, nyc_17$`Vehicle Body Type`), count = n(nyc_17$`Summons Number`))

View(head(arrange(v_body, desc(v_body$count))))

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

#Vehicle Body Type count

#SUBN 3719802

#4DSD 3082020

#VAN 1411970

#DELV 687330

#SDN 438191

#2DSD 274380

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

 # the top 5 vehicle body type that got parking tickets are SUBN,4DSD,VAN,DELV and SDN

#Find the top 5 vehicle body make can get parking tickets

v_make <- summarize(groupBy(nyc_17, nyc_17$`Vehicle Make`), count = n(nyc_17$`Summons Number`))

```
View(head(arrange(v_make, desc(v_make$count))))
########################
#Vehicle Make  count
#FORD          1280958
#TOYOT         1211451
#HONDA         1079238
#NISSA         918590
#CHEVR         714655
#FRUEH         429158
########################
```

#the top 5 vehicle body make that got parking tickets are FORD,TOYOT,HONDA,NISSA and CHEVR

```
#3(a). Violation precinct
non_zero_preinct <- filter(nyc_17, nyc_17$`Violation Precinct` > 0)

Vio_preinct <- summarize(groupBy(non_zero_preinct, non_zero_preinct$`Violation Precinct`),
count = n(non_zero_preinct$`Violation Precinct`))

View(head(arrange(Vio_preinct, desc(Vio_preinct$count)))
########################
#Violation Precinct      count
#19                      535671
#14                      352450
#1                       331810
#18                      306920
#114                     296514
#13                      246595
########################

#3(b). Issuer Precinct
non_zero_issuer <- filter(nyc_17, nyc_17$`Issuer Precinct` > 0)
```

```
iss_preinct <- summarize(groupBy(non_zero_issuer, non_zero_issuer$`Issuer Precinct`),
count = n(non_zero_issuer$`Issuer Precinct`))

View(head(arrange(iss_preinct, desc(iss_preinct$count))))
```

```
#######################
#Issuer Precinct count
#19                  521513
#14                  344977
#1                   321170
#18                  296553
#114                 289950
#13                  240833
#######################
```

#The top 5 violation and issuer precincts are 19,14,1, 18 and 114

```
#4. Violation & Issuer Precinct
non_zero_iv_preinct <- filter(nyc_17, nyc_17$`Violation Precinct` > 0 | nyc_17$`Issuer Precinct` > 0)
iv_preinct_17 <- summarize(groupBy(non_zero_iv_preinct, non_zero_iv_preinct$`Issuer Precinct`,
non_zero_iv_preinct$`Violation Precinct`), count = n(non_zero_iv_preinct$`Violation Code`))
View(head(arrange(iv_preinct_17, desc(iv_preinct_16$count))))
```

```
#######################
#Violation Precinct   Issuer Precinct   count
#19                  19                521182
#14                  14                344347
#1                   1                 316544
#18                  18                296369
#114                 114               289723
#13                  13                240684
#######################
```

#The top 5 violation and issuer precincts are 19,14,1, 18 and 114

```
#5. drop all na values from the dataset

nyc_17 <- dropna(nyc_17, how="all")

#remove erroneous date formats

createOrReplaceTempView(nyc_17, "nyc_17_view")

nyc17 <- SparkR::sql("select `Violation Time`, ‘Violation Code` from nyc_15_view")


nyc2017 <- mutate(nyc17, Hour = substr(nyc17$Time,1, 2), Min = substr(nyc17$Time, 3,4), Half =
substr(nyc17$Time, 5,5))


nyc20171 <- transform(nyc2017, Hour = ifelse(nyc2017$Half == 'P' & nyc2017$Hour < 12,
nyc2017$Hour+12, nyc2017$Hour))

nyc17 <- transform(nyc20171, Hour = cast(nyc20171$Hour, "integer"), Min = cast(nyc20171$Min,
"integer"))


nrow(filter(nyc17, nyc17$Hour > 23))



Hour2017 <- summarize(groupBy(nyc17, nyc17$Hour), count = n(nyc17$Hour))


#Divide 24 hours into six equal discrete bins of time.

createOrReplaceTempView(nyc_17, "nyc_17_view")


bin2017 <- SparkR::sql("select nyc_17_view.Time, nyc_17_view.Code, nyc_17_view.Hour,
nyc_17_view.Min, nyc_17_view.Half,
                       CASE WHEN Hour > 23 THEN 'Invalid'

                       WHEN Hour < 4 THEN '0-3'

                       WHEN Hour >=4 AND Hour <8 THEN '4-7'

                       WHEN Hour >=8 and Hour <12 THEN '8-11'

                       WHEN Hour >=12 and Hour <16 THEN '12-15'

                       WHEN Hour >=16 and Hour <20 THEN '16-19'

                       ELSE '20-23' END as Bin from nyc_17_view")
```

```
head(bin2017)

Binned2017 <- summarize(groupBy(bin2017, bin2017$Bin, bin2017$Code), count=n(bin2017$Code))


#find out the top 3 violation codes

top3ViolationCodes <- filter(bin2017, bin2017$Code %in% c(21,36,38))


ViolationCodeSummary <- summarize(groupBy(top3ViolationCodes, top3ViolationCodes$Code,
top3ViolationCodes$Bin), count=n(top3ViolationCodes$Bin))


createOrReplaceTempView(Summary, "SummaryView")


top3VioCount <- SparkR::sql("select bin, code, count from (select bin, code, count, dense_rank() over

                    (partition by code order by count desc) as rank from "SummaryView") tmp where
rank <=1 order by code")


head(top3VioCount)

#most common violation codes are 21,36,38

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

#### bin code   count

#1  8-11   21 1182691

#2  8-11   36  751422

#3 12-15   38  462859

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


#6. Finding seasonality

createOrReplaceTempView(nyc_17, "nyc_17_view")

nyc2017 <- SparkR::sql("select `Violation Code`, `Issue Date` from "nyc_17_view")


#convert data and time into appropriate format

nyc2017 <- mutate(nyc2017 , Date = to_date(nyc2017$`Issue Date`, 'MM/dd/yyyy'))
```

```
nyc2017 <- mutate(nyc2017 , Week = weekofyear(nyc2017$Date))

#removing irrelevant dates

nrow(filter(nyc2017, "Date < '2017-01-01'"))

createOrReplaceTempView(nyc2017, "nyc_17_view")


Season <- SparkR::sql("select nyc_17_view.code, nyc_17_view.NewDate, CASE WHEN Week <=13
THEN 'Spring'

                WHEN Week > 13 AND Week < 27 THEN 'Summer'

                WHEN Week >=27 AND Week < 40 THEN 'Autumn'

                ELSE 'Winter' END as Seasons from nyc_17_view")



Tickets <- summarize(groupBy(Season, Season$Seasons), count=n(Season$code))

head(Tickets)

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

# Quarters   count

# Summer 3236607

# Spring 3135405

# Autumn 2935987

# Winter 2501234

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

#summer season has the most occuring violations


ViolationsPerSeason <- summarize(groupBy(Season, Season$Seasons, Season$code),
count=n(Season$code))

head(ViolationsPerSeason)


createOrReplaceTempView(ViolationsPerSeason, "ViolationPerSeasonView")


top3ViolationPerSeason <- SparkR::sql("select Quarters, code, count from (select Quarters, code,
count, dense_rank() over
```

(partition by Quarters order by count desc) as rank from
ViolationPerSeasonView) tmp where rank <=3")

#4 quarters/seasons , top3 in each

head(top3ViolationPerSeason, 12)

#most common violation codes are 21,36,38

#7. Find total occurrences of the three most common violation codes

parking_2017_viol_codes_counts <- summarize(groupBy(nyc_17, parking_2017$Vio1ation_Code),

count = n(parking_2017$Violation_£ode))

head(arrange(parking_2017_viol_codes_counts, desc(parking_2017_viol_codes_counts$count)))

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

Violation_Code count

1 21 1528588

2 36 1400614

3 38 1062304

4 14 893498

5 20 618593

6 46 600012

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

#the three most common violation codes are 21, 36,38

#Fine amount for violation code 21 is (65+45)/2 = 55

#Fine amount for violation code 36 is (50+50)/2 = 50

#Fine amount for violation code 38 is (65+35)/2 = 50

#find the total amount collected for the three violation codes with maximum tickets.

#State the code which has the highest total collection.

createOrReplaceTempView(parking_2017_viol_codes_counts, "parking_2017_vio1ation_code“)

pr_2017_viol_codes_counts <- sq1("SELECT Violation_pode, \

CASE WHEN Violation Code = 21 THEN count * 55 \

WHEN Violation_Code 38 THEN count * 50 \

WHEN Violation_Code 36 THEN count * 50 \

ELSE 0 END as revenue FROM parking_2017_violation_code \

WHERE Violation_Code IN (21, 38, 36)")

head(arrange(pr_2017_viol_codes_counts, desc(pr_2017_viol_codes_counts$revenue)))

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

Violation_Code revenue

1 21 84072340

2 36 70030700

3 38 53115200

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

#Hence violation 21 has highest total collection.

Stopping the sparkR session

sparkR.stop()

CONCLUSION

After analysing the data from the years 2015,2016 and 2017, the following results were obtained

- The summer season has the most number of violations in each year.
- The highest number of violations were recorded in the year 2015.

Thus, all the analysis is done and the insights and findings are mentioned in bold.