create table parking2007

(

summons\_number int,

plate\_id string,

registration\_state string,

plate\_type string,

issue\_date string,

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 int,

violation\_location string,

violation\_precinct int,

issuer\_precinct int,

issuer\_code int,

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 int,

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 serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde'

with serdeproperties ("separatorChar" = ",", "quoteChar" = "\"", "escapeChar" = "\\")

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

------------------------------------------------------------------------------------------

Note- I am taking applying serde properties because the dataset has some strings with quote Characters

------------------------------------------------------------------------------------------

load data local inpath '/home/cloudera/BIGDATA/Parking.csv'

into table parking2007;

--------------------------------------------------------

CREATE TABLE `parking2007\_orc`(

`summons\_number` int,

`plate\_id` string,

`registration\_state` string,

`plate\_type` string,

`issue\_date` date,

`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` int,

`issuer\_command` string,

`issuer\_squad` string,

`violation\_time` timestamp,

`time\_first\_observed` timestamp,

`violation\_county` string,

`violation\_in\_front\_of\_or\_opposite` string,

`house\_number` string,

`street\_name` string,

`intersecting\_street` string,

`date\_first\_observed` date,

`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` int,

`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\_code` int,

`vehicle\_body\_type` string)

clustered by (issue\_date) into 5 buckets;

insert into parking2007\_orc partition(violation\_code, vehicle\_body\_type )

select

summons\_number,

plate\_id,

registration\_state,

plate\_type,

from\_unixtime(unix\_timestamp(issue\_date,'MM/dd/yyyy')) issue\_date ,

vehicle\_make,

issuing\_agency,

street\_code1,

street\_code2,

street\_code3,

from\_unixtime(unix\_timestamp(cast(vehicle\_expiration\_date as string),'yyyyMMdd')) vehicle\_expiration\_date,

violation\_location,

violation\_precinct,

issuer\_precinct,

issuer\_code,

issuer\_command,

issuer\_squad,

from\_unixtime(unix\_timestamp(

concat(issue\_date,' ',concat(violation\_time,'M')),'MM/dd/yyyy hhmmaa')) violation\_time,

from\_unixtime(unix\_timestamp(

concat(cast(date\_first\_observed as string),' ',concat(time\_first\_observed,'M')),'yyyyMMdd hhmmaa')) time\_first\_observed,

violation\_county,

violation\_in\_front\_of\_or\_opposite,

house\_number,

street\_name,

intersecting\_street,

from\_unixtime(unix\_timestamp(

cast(date\_first\_observed as string),'yyyyMMdd')) 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\_code,

vehicle\_body\_type

from parking2007

limit 100000

PART I:

**1.) Find the total number of tickets for the year.**

Ans:

Select count(\*) total\_tickets from parking2007\_orc;

**2.) Find out how many unique states the cars which got parking tickets came from.**

Ans:

SELECT distinct registration\_state FROM parking2007\_orc;

**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 )**

Ans:

SELECT count(\*) FROM parking2007\_orc

where street\_code1 = ‘0’ or street\_code2 = ‘0’ or street\_code3 = ‘0’

PART II:

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

Ans:

SELECT violation\_code, count(\*) cnt

FROM default.parking2007\_orc

GROUP BY violation\_code

order by cnt 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(\*) cnt

FROM parking2007\_orc

GROUP BY vehicle\_body\_type

order by cnt desc limit 5;

SELECT vehicle\_make, count(\*) cnt

FROM parking2007\_orc

GROUP BY vehicle\_make

order by cnt desc limit 5;

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

**a.) Violating Precincts (this is the precinct of the zone where the violation occurred)**

**Ans:**

select violation\_precinct, count(\*) from parking2007\_orc

group by violation\_precinct;

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

Ans:

select issuer\_precinct, count(\*) from parking2007\_orc

group by issuer\_precinct;

**4.) 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?**

Ans:

with cte as --for getting count of violations in each precinct

(

SELECT issuer\_precinct,

violation\_code,

count(violation\_code) over (partition by issuer\_precinct) cnt

FROM parking2007\_orc

),

cte1 as-- for getting violation count wrt each precinct and violation code.

(

select issuer\_precinct, cnt precinct\_viol\_count, violation\_code, count(\*) Cnt\_violations

from cte

group by issuer\_precinct, cnt, violation\_code

),

cte2 as-- ranking each precinct based on the no. of violations occured descending

(

select

issuer\_precinct,violation\_code,Cnt\_violations,

dense\_rank() over(order by precinct\_viol\_count desc) rnk

from cte1

)

select issuer\_precinct,violation\_code,Cnt\_violations from cte2

where rnk<=3

order by issuer\_precinct, Cnt\_violations desc

**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.**

**Ans:** Already handled the time field while loading into ORC table. See Insert Query above.

**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**

**Ans:**

with cte as-- for dividing time range into 6 bins

(

select

case

when cast(date\_format(violation\_time, 'H') as int) between 0 and 3 then '1st (0-3)'

when cast(date\_format(violation\_time, 'H') as int) between 4 and 7 then '2nd (4-7)'

when cast(date\_format(violation\_time, 'H') as int) between 8 and 11 then '3rd (8-11)'

when cast(date\_format(violation\_time, 'H') as int) between 12 and 15 then '4th (12-15)'

when cast(date\_format(violation\_time, 'H') as int) between 16 and 19 then '5th (16-19)'

when cast(date\_format(violation\_time, 'H') as int) between 20 and 23 then '6th (20-23)'

end time\_group,

violation\_code

from

parking2007\_orc

),

cte1 as

(

select

time\_group,

violation\_code,

count(\*) violations

from cte

group by time\_group,

violation\_code),

cte2 as

(

select time\_group, violation\_code, violations,

dense\_rank() over (partition by time\_group order by violations desc) rnk

from cte1

)

select time\_group, violation\_code, violations from cte2

where rnk <=3

with cte as

(

select

case

when cast(date\_format(violation\_time, 'H') as int) between 0 and 3 then '1st'

when cast(date\_format(violation\_time, 'H') as int) between 4 and 7 then '2nd'

when cast(date\_format(violation\_time, 'H') as int) between 8 and 11 then '3rd'

when cast(date\_format(violation\_time, 'H') as int) between 12 and 15 then '4th'

when cast(date\_format(violation\_time, 'H') as int) between 16 and 19 then '5th'

when cast(date\_format(violation\_time, 'H') as int) between 20 and 23 then '6th'

end time\_group,

violation\_code

from

parking2007\_orc

),

cte1 as (

select

time\_group,

violation\_code,

count(\*) violations

from cte

group by time\_group,

violation\_code)

select time\_group, violation\_code, violations,

dense\_rank() over (partition by time\_group order by violations desc) rnk

from cte1

having rnk<=3

**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)**

**Ans:**

with cte as-- for dividing time range into 6 bins

(

select

case

when cast(date\_format(violation\_time, 'H') as int) between 0 and 3 then '1st (0-3)'

when cast(date\_format(violation\_time, 'H') as int) between 4 and 7 then '2nd (4-7)'

when cast(date\_format(violation\_time, 'H') as int) between 8 and 11 then '3rd (8-11)'

when cast(date\_format(violation\_time, 'H') as int) between 12 and 15 then '4th (12-15)'

when cast(date\_format(violation\_time, 'H') as int) between 16 and 19 then '5th (16-19)'

when cast(date\_format(violation\_time, 'H') as int) between 20 and 23 then '6th (20-23)'

end time\_group,

violation\_code,

count(violation\_code) over (partition by violation\_code) violations

from

parking2007\_orc

),

cte1 as -- for getting 3 violation\_codes with most violations

(

select

\*,

dense\_rank() over (order by violations desc) most\_viol\_rnk

from cte

)

select violation\_code, time\_group, violations Total\_Viol\_under\_code,

count(time\_group) Total\_Viol\_under\_timegroup

from cte1

where most\_viol\_rnk<=3

group by violation\_code, time\_group, violations

order by violation\_code, Total\_Viol\_under\_timegroup desc

**8.) 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, May); Summer(June, July, August); Fall(September, October, November); Winter(December, January, February))**

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

**Ans:**

with cte as

(

select

case

when date\_format(violation\_time, 'MMM') in ('Mar','Apr','May') then 'Spring'

when date\_format(violation\_time, 'MMM') in ('Jun','Jul','Aug') then 'Summer'

when date\_format(violation\_time, 'MMM') in ('Sep','Oct','Nov') then 'Fall'

when date\_format(violation\_time, 'MMM') in ('Dec','Jan','Feb') then 'Winter'

end season,

violation\_code

from

parking2007\_orc

),

cte2 as

(

select season, violation\_code,

count(\*) violatons\_per\_season\_per\_code

from cte

group by season, violation\_code

),

cte3 as (

select season, violation\_code,

dense\_rank() over(partition by season order by violatons\_per\_season\_per\_code desc) rnk

from cte2

)

select season, violation\_code from cte3

where rnk<=3;