Hive -mini project-2

Abhijit Barik

```
#Table creation

create table parking_violations

(

Summons_Number bigint,

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 string,

Violation_Location int,

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 int,
Vehicle_Year string,
Meter_Number string,
Feet_From_Curb int,
Violation_Post_Code string,
Violation_Description string,
No_Standing_or_Stopping_Violation string,
Hydrant_Violation string,
Double_Parking_Violation string)
row format delimited
fields terminated by ','
tblproperties ("skip.header.line.count" = "1");
```

as the above table uses string datatype for dates, so in order to correct that a new table is created with the required datatype

```
create table violations_parking
(
Summons_Number bigint,
Plate_ID string,
Registration_State string,
Plate_Type string,

Issue_Date date,

Violation_Code int,

Vehicle_Body_Type string,

Vehicle_Make string,

Issuing_Agency string,

Street_Code1 int,

Street_Code2 int,

Street_Code3 int,

Vehicle_Expiration date,

Violation_Location int,

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 int,

Vehicle_Year string,

Meter_Number string,

Feet_From_Curb int,

Violation_Post_Code string,

Violation_Description string,

No_Standing_or_Stopping_Violation string,

Hydrant_Violation string,

Double_Parking_Violation string)

row format delimited

fields terminated by ',';

Insert the data from the parking_violations table into violations_parking table

insert overwrite table violations_parking select

Summons_Number bigint,

Plate_ID string,

Registration_State string,

Plate_Type string,

from_unixtime(unix_timestamp(Issue_Date,'MM/dd/YYYY'),'yyyy-MM-dd'),

Violation_Code int,

Vehicle_Body_Type string,

Vehicle_Make string,

Issuing_Agency string,

Street_Code1 int,

Street_Code2 int,

Street_Code3 int,

from_unixtime(unix_timestamp(Vehicle_Expiration,'YYYYMMdd'),'yyyy-MM-dd'),

Violation_Location int,

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 int,

Vehicle_Year string,

Meter_Number string,

Feet_From_Curb int,

Violation_Post_Code string,

Violation_Description string,

No_Standing_or_Stopping_Violation string,

Hydrant_Violation string,

Double_Parking_Violation string

from parking_violations;
```

Now create a partition and bucketing of a table as this improves the execution speed of queries as the data size is big..

```
create table park_viol_part_buck

(

Summons_Number bigint,

Plate_ID string,

Registration_State string,

Plate_Type string,

Issue_Date date,

Violation_Code int,

Vehicle_Body_Type string,

Vehicle_Make string,

Issuing_Agency string,

Street_Code1 int,

Street_Code2 int,

Street_Code3 int,

Vehicle_Expiration date,

Violation_Location int,

Violation_Precinct int,

Issuer_Precinct int,

Issuer_Code int,

Issuer_Command string,

Issuer_Squad string,

Violation_Time string,

Time_First_Observed string,

Violation_In_Front_Of_Or_Opposite string,

House_Number string,

Street_Name string,

Intersecting_Street string,

Date_First_Observed int,

Law_Section int,
Sub_Division string,
Violation_Legal_Code string,
Days_Parking_In_Effect string,
From_Hours_In_Effect string,
To_Hours_In_Effect string,
Vehicle_Color string,
Unregistered_Vehicle int,
Vehicle_Year string,
Meter_Number string,
Feet_From_Curb int,
Violation_Post_Code string,
Violation_Description string,
No_Standing_or_Stopping_Violation string,
Hydrant_Violation string,
Double_Parking_Violation string)
partitioned by (Violation_County string)
clustered by (Violation_Code)
sorted by(Violation_Code) into 8 buckets
row format delimited
fields terminated by ','
tblproperties ("skip.header.line.count" = "1");
```

Before loading the data into the above table some properties are needed to be set...

```
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.enforce.bucketing = true;
```

Now insert the data into the parted and bucketed table from the violations parking table insert into park_viol_part_buck partition(Violation_County) select

Summons_Number,Plate_ID,Registration_State,Plate_Type,Issue_Date,Violation_Code,

Vehicle_Body_Type,Vehicle_Make,Issuing_Agency,Street_Code1,Street_Code2,

Street_Code3,Vehicle_Expiration,Violation_Location,Violation_Precinct,

Issuer_Precinct,Issuer_Code,Issuer_Command,Issuer_Squad,Violation_Time,

Time_First_Observed,Violation_In_Front_Of_Or_Opposite,House_Number,Street_Name,

Intersecting_Street,Date_First_Observed,Law_Section,Sub_Division,Violation_Legal_Code,

Days_Parking_In_Effect,From_Hours_In_Effect,To_Hours_In_Effect,Vehicle_Color,

Unregistered_Vehicle,Vehicle_Year,Meter_Number,Feet_From_Curb,Violation_Post_Code,

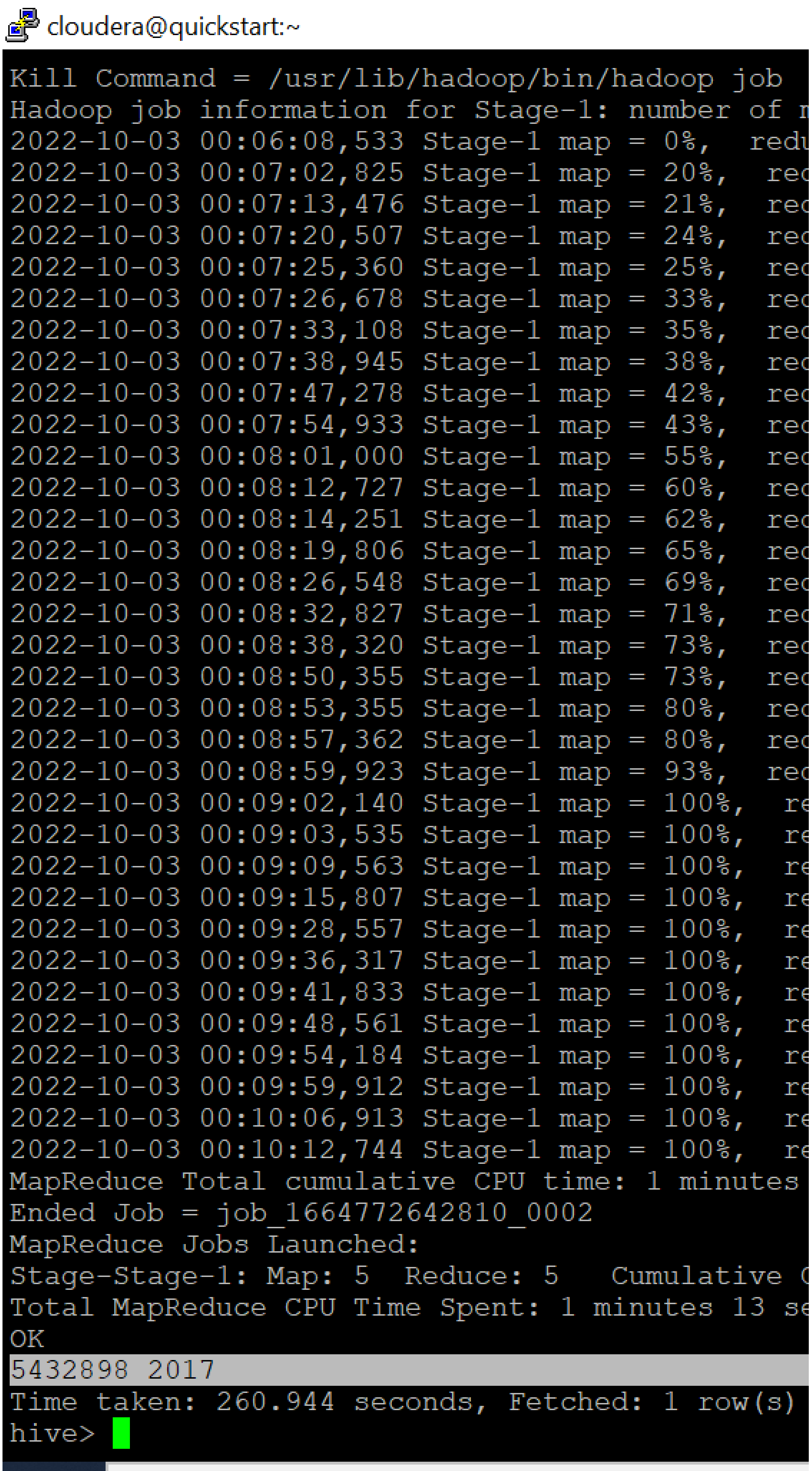
Violation_Description,No_Standing_or_Stopping_Violation,Hydrant_Violation,

Double_Parking_Violation,Violation_County from violations_parking

where year(Issue_Date) = '2017';

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

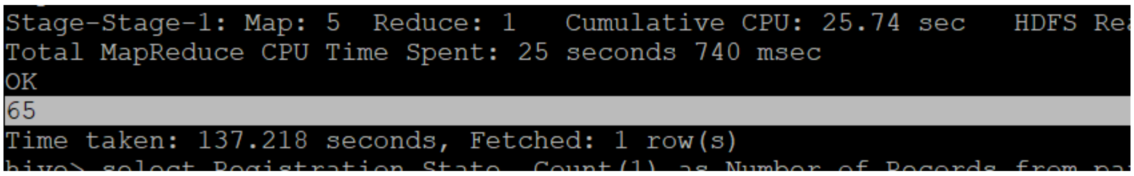
ANS) select count(distinct Summons_Number) as Tickets_Total ,year(Issue_Date) as year from park_viol_part_buck group by year(Issue_Date);



```
cloudera@quickstart:~
Kill Command = /usr/lib/hadoop/bin/hadoop job
Hadoop job information for Stage-1: number of r
2022-10-03 00:06:08,533 Stage-1 map = 0%,  redu
2022-10-03 00:07:02,825 Stage-1 map = 20%,  red
2022-10-03 00:07:13,476 Stage-1 map = 21%,  red
2022-10-03 00:07:20,507 Stage-1 map = 24%,  red
2022-10-03 00:07:25,360 Stage-1 map = 25%,  red
2022-10-03 00:07:26,678 Stage-1 map = 33%,  red
2022-10-03 00:07:33,108 Stage-1 map = 35%,  red
2022-10-03 00:07:38,945 Stage-1 map = 38%,  red
2022-10-03 00:07:47,278 Stage-1 map = 42%,  red
2022-10-03 00:07:54,933 Stage-1 map = 43%,  red
2022-10-03 00:08:01,000 Stage-1 map = 55%,  red
2022-10-03 00:08:12,727 Stage-1 map = 60%,  red
2022-10-03 00:08:14,251 Stage-1 map = 62%,  red
2022-10-03 00:08:19,806 Stage-1 map = 65%,  red
2022-10-03 00:08:26,548 Stage-1 map = 69%,  red
2022-10-03 00:08:32,827 Stage-1 map = 71%,  red
2022-10-03 00:08:38,320 Stage-1 map = 73%,  red
2022-10-03 00:08:50,355 Stage-1 map = 73%,  red
2022-10-03 00:08:53,355 Stage-1 map = 80%,  red
2022-10-03 00:08:57,362 Stage-1 map = 80%,  red
2022-10-03 00:08:59,923 Stage-1 map = 93%,  red
2022-10-03 00:09:02,140 Stage-1 map = 100%,  re
2022-10-03 00:09:03,535 Stage-1 map = 100%,  re
2022-10-03 00:09:09,563 Stage-1 map = 100%,  re
2022-10-03 00:09:15,807 Stage-1 map = 100%,  re
2022-10-03 00:09:28,557 Stage-1 map = 100%,  re
2022-10-03 00:09:36,317 Stage-1 map = 100%,  re
2022-10-03 00:09:41,833 Stage-1 map = 100%,  re
2022-10-03 00:09:48,561 Stage-1 map = 100%,  re
2022-10-03 00:09:54,184 Stage-1 map = 100%,  re
2022-10-03 00:09:59,912 Stage-1 map = 100%,  re
2022-10-03 00:10:06,913 Stage-1 map = 100%,  re
2022-10-03 00:10:12,744 Stage-1 map = 100%,  re
MapReduce Total cumulative CPU time: 1 minutes
Ended Job = job_1664772642810_0002
MapReduce Jobs Launched:
Stage-Stage-1: Map: 5  Reduce: 5   Cumulative C
Total MapReduce CPU Time Spent: 1 minutes 13 se
OK
5432898 2017
Time taken: 260.944 seconds, Fetched: 1 row(s)
hive>
```

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

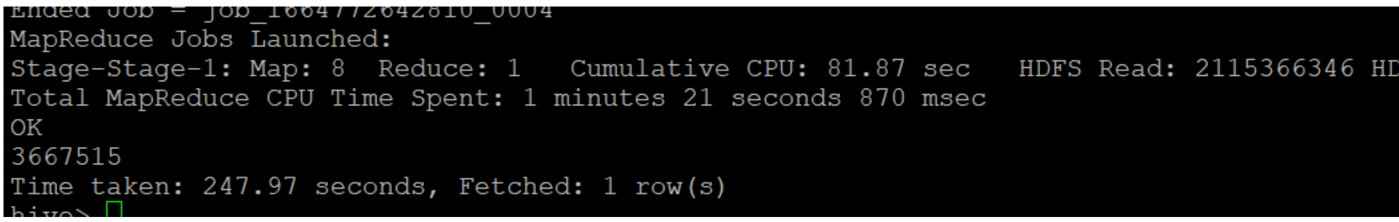
Ans) select count(distinct Registration_State) as No_of_States from park_viol_part_buck;



```
Stage-Stage-1: Map: 5  Reduce: 1   Cumulative CPU: 25.74 sec   HDFS Re
Total MapReduce CPU Time Spent: 25 seconds 740 msec
OK
65
Time taken: 137.218 seconds, Fetched: 1 row(s)
hive> select Registration_State, Count(1) as Number_of_Records from pa
```

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(distinct summons_number) as No_Tickets_without_address from violations_parking where Street_code1 = 0 or Street_code2 = 0 or Street_code3 = 0;

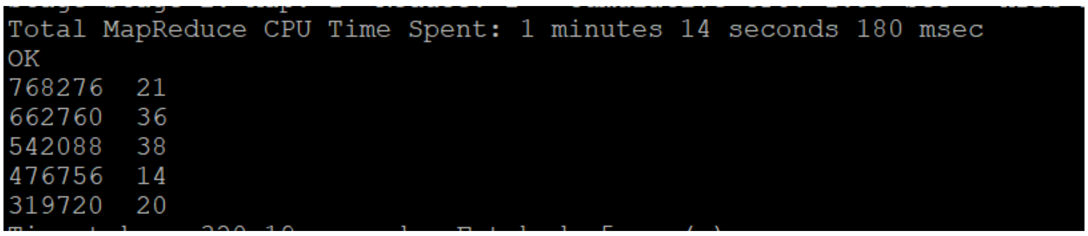


```
Ended Job = job_1664772642810_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 8  Reduce: 1   Cumulative CPU: 81.87 sec   HDFS Read: 2115366346 HD
Total MapReduce CPU Time Spent: 1 minutes 21 seconds 870 msec
OK
3667515
Time taken: 247.97 seconds, Fetched: 1 row(s)
hive> 
```

Part -II: Aggregation tasks

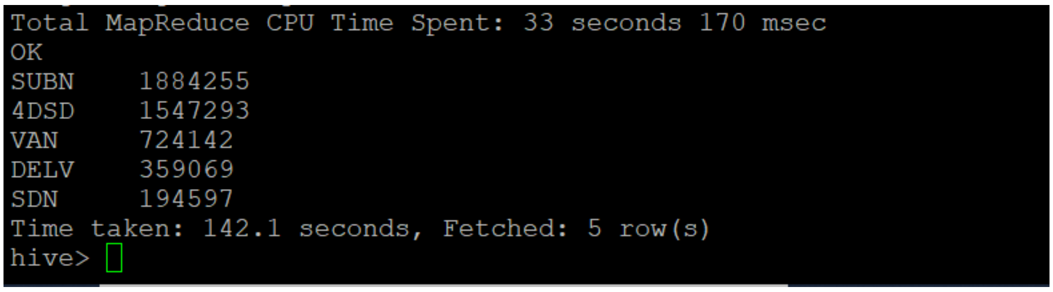
1.) How often does each violation code occur? (frequency of violation codes - find the top 5)

Ans) select count(Violation_Code) as frequency_of_violation,Violation_Code from park_viol_part_buck group by Violation_Code order by frequency_of_violation desc limit 5;


```
Total MapReduce CPU Time Spent: 1 minutes 14 seconds 180 msec
OK
768276  21
662760  36
542088  38
476756  14
319720  20
```

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

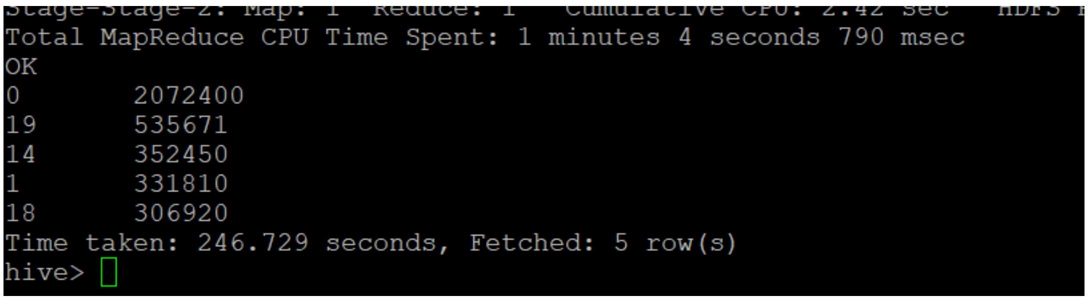
Ans) select Vehicle_Body_Type,count(summons_number)as frequency_of_getting_parking_ticket from park_viol_part_buck group by Vehicle_Body_Type order by frequency_of_getting_parking_ticket desc limit 5;


```
Total MapReduce CPU Time Spent: 33 seconds 170 msec
OK
SUBN    1884255
4DSD    1547293
VAN     724142
DELV    359069
SDN     194597
Time taken: 142.1 seconds, Fetched: 5 row(s)
hive>
```

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)

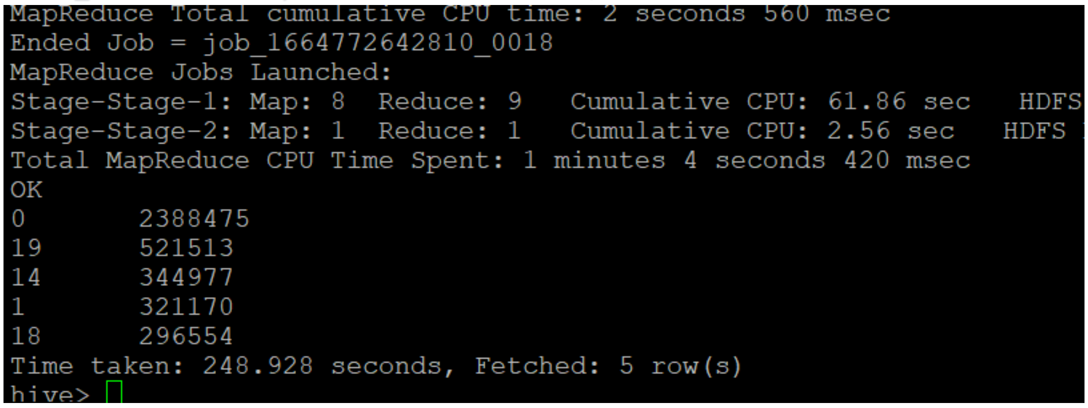
Ans) select Violation_Precinct,count(*) as IssuedTicket from violations_parking group by Violation_Precinct order by IssuedTicket desc limit 5;



```
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.42 sec   HDFS
Total MapReduce CPU Time Spent: 1 minutes 4 seconds 790 msec
OK
0       2072400
19      535671
14      352450
1       331810
18      306920
Time taken: 246.729 seconds, Fetched: 5 row(s)
hive>
```

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

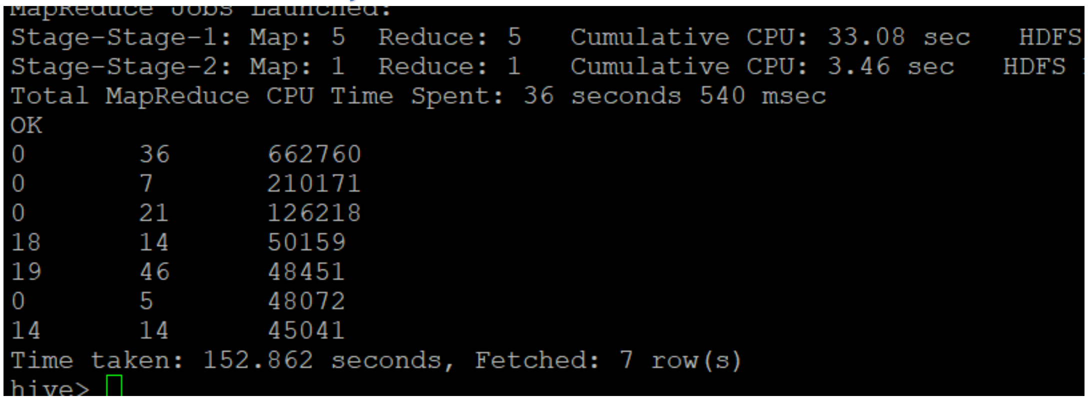
Ans) select Issuer_Precinct,count(*) as IssuedTicket from violations_parking group by Issuer_Precinct order by IssuedTicket desc limit 5;



```
MapReduce Total cumulative CPU time: 2 seconds 560 msec
Ended Job = job_1664772642810_0018
MapReduce Jobs Launched:
Stage-Stage-1: Map: 8  Reduce: 9   Cumulative CPU: 61.86 sec   HDFS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.56 sec   HDFS
Total MapReduce CPU Time Spent: 1 minutes 4 seconds 420 msec
OK
0       2388475
19      521513
14      344977
1       321170
18      296554
Time taken: 248.928 seconds, Fetched: 5 row(s)
hive>
```

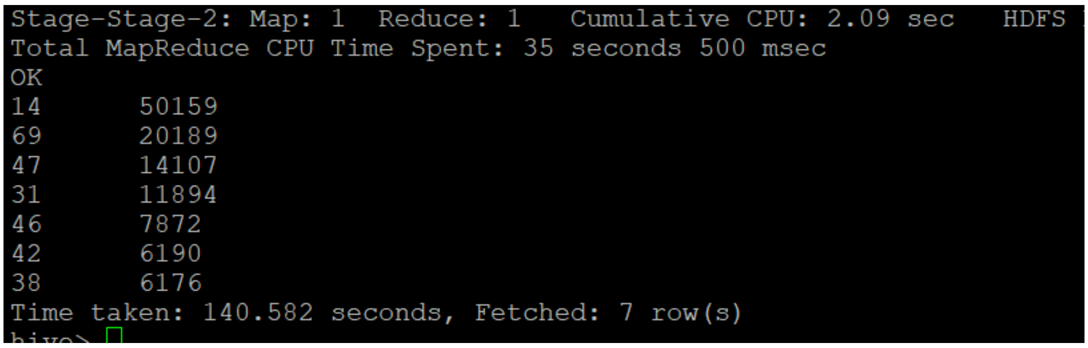
4.) Find the violation code frequency across 3 precincts which have issued the most number of tickets - do these precinct zones have an exceptionally high frequency of certain violation codes?

Ans) select Issuer_Precinct,Violation_Code, count(*) as TicketsIssued from park_viol_part_buck group by Issuer_Precinct, Violation_Code order by TicketsIssued desc limit 7;



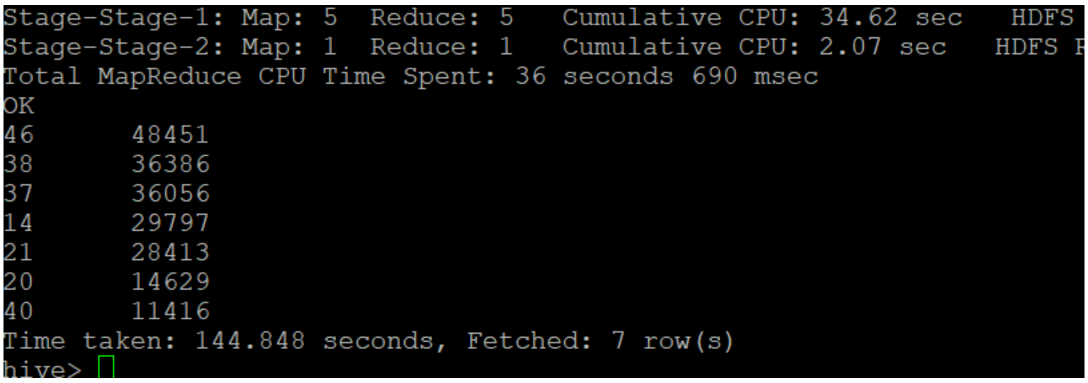
```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 5  Reduce: 5   Cumulative CPU: 33.08 sec   HDFS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 3.46 sec   HDFS
Total MapReduce CPU Time Spent: 36 seconds 540 msec
OK
0       36      662760
0       7       210171
0       21      126218
18      14      50159
19      46      48451
0       5       48072
14      14      45041
Time taken: 152.862 seconds, Fetched: 7 row(s)
hive>
```

```
select Violation_Code, count(*) as TicketsIssued from park_viol_part_buck where
Issuer_Precinct=18 group by Violation_Code order by TicketsIssued desc limit 7;
```



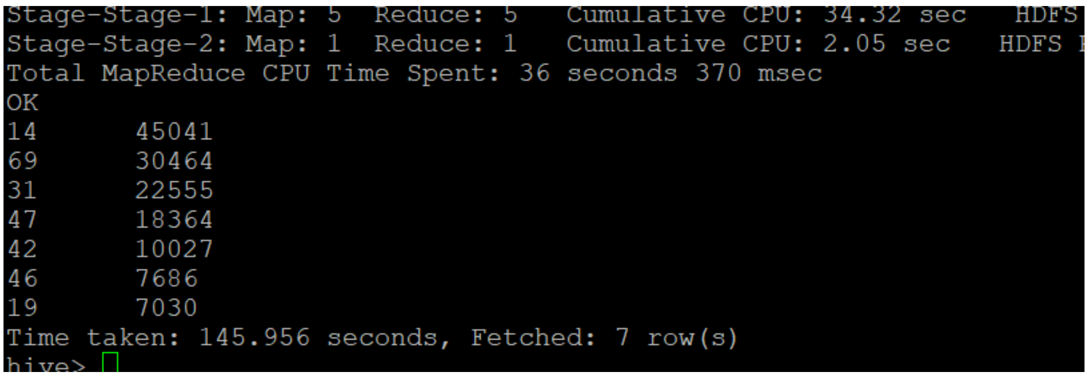
```
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.09 sec   HDFS
Total MapReduce CPU Time Spent: 35 seconds 500 msec
OK
14      50159
69      20189
47      14107
31      11894
46      7872
42      6190
38      6176
Time taken: 140.582 seconds, Fetched: 7 row(s)
hive>
```

```
select Violation_Code, count(*) as TicketsIssued from park_viol_part_buck where
Issuer_Precinct=19 group by Violation_Code order by TicketsIssued desc limit 7;
```



```
Stage-Stage-1: Map: 5  Reduce: 5   Cumulative CPU: 34.62 sec   HDFS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.07 sec   HDFS
Total MapReduce CPU Time Spent: 36 seconds 690 msec
OK
46      48451
38      36386
37      36056
14      29797
21      28413
20      14629
40      11416
Time taken: 144.848 seconds, Fetched: 7 row(s)
hive>
```

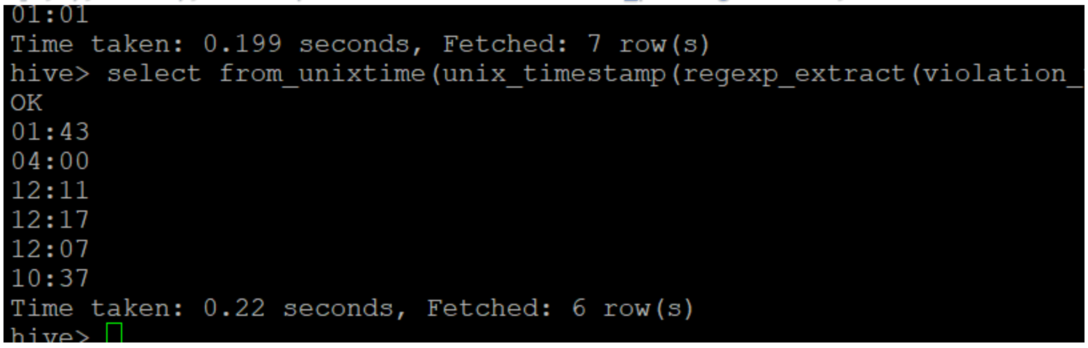
```
select Violation_Code, count(*) as TicketsIssued from park_viol_part_buck where
Issuer_Precinct=14 group by Violation_Code order by TicketsIssued desc limit 7;
```



```
Stage-Stage-1: Map: 5  Reduce: 5   Cumulative CPU: 34.32 sec   HDFS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.05 sec   HDFS
Total MapReduce CPU Time Spent: 36 seconds 370 msec
OK
14      45041
69      30464
31      22555
47      18364
42      10027
46      7686
19      7030
Time taken: 145.956 seconds, Fetched: 7 row(s)
hive>
```

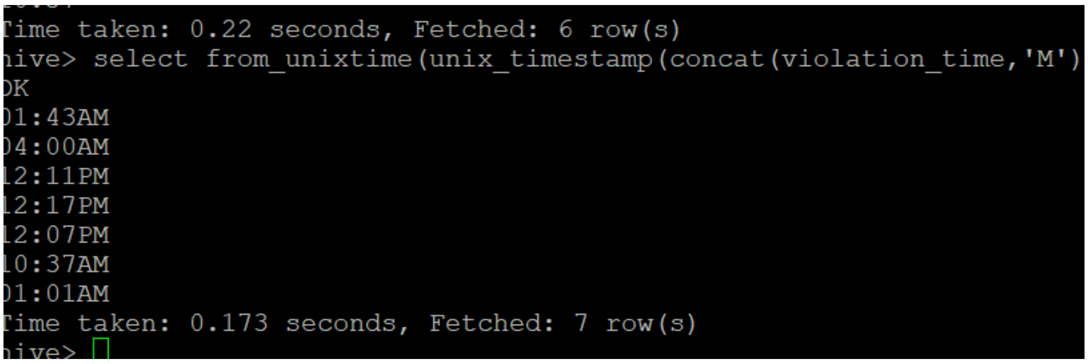
5.) Find out the properties of parking violations across different times of the day: The Violation Time field is specified in a strange format. Find a way to make this into a time attribute that you can use to divide into groups.

```
Ans) select from_unixtime(unix_timestamp(regexp_extract(violation_time,'(.*)[A-
Z]',1),'HHmm'),"HH:mm") as data from violations_parking limit 7;
```



```
01:01
Time taken: 0.199 seconds, Fetched: 7 row(s)
hive> select from_unixtime(unix_timestamp(regexp_extract(violation_
OK
01:43
04:00
12:11
12:17
12:07
10:37
Time taken: 0.22 seconds, Fetched: 6 row(s)
hive>
```

```
select from_unixtime(unix_timestamp(concat(violation_time,'M'),
'HHmmaaa'),"HH:mmaaa") as data from violations_parking limit 7;
```



```
Time taken: 0.22 seconds, Fetched: 6 row(s)
hive> select from_unixtime(unix_timestamp(concat(violation_time,'M')
OK
01:43AM
04:00AM
12:11PM
12:17PM
12:07PM
10:37AM
01:01AM
Time taken: 0.173 seconds, Fetched: 7 row(s)
hive>
```

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

```
create view park_viol_part_view partitioned on (Violation_Code) as
select Summons_Number, Violation_Time, Issuer_Precinct,
case
when substring(Violation_Time,1,2) in ('00','01','02','03','12') and
upper(substring(Violation_Time,-1))='A' then 1
when substring(Violation_Time,1,2) in ('04','05','06','07') and
upper(substring(Violation_Time,-1))='A' then 2
when substring(Violation_Time,1,2) in ('08','09','10','11') and
upper(substring(Violation_Time,-1))='A' then 3
when substring(Violation_Time,1,2) in ('12','00','01','02','03') and
upper(substring(Violation_Time,-1))='P' then 4
when substring(Violation_Time,1,2) in ('04','05','06','07') and
upper(substring(Violation_Time,-1))='P' then 5
when substring(Violation_Time,1,2) in ('08','09','10','11') and
upper(substring(Violation_Time,-1))='P'then 6
else null end as Violation_Time_bin,Violation_Code
from park_viol_part_buck
where Violation_Time is not null or (length(Violation_Time)=5 and
upper(substring(Violation_Time,-1))in ('A','P')
```

and substring(Violation_Time,1,2) in ('00','01','02','03','04','05','06','07','08','09','10','11','12'));

BIN=1

```
select Violation_Code,count(*) TicketsIssued from park_viol_part_view where
Violation_Time_bin == 1 group by Violation_Code order by TicketsIssued desc
limit 2;
```

BIN2

```
select Violation_Code,count(*) TicketsIssued from park_viol_part_view where
Violation_Time_bin == 2 group by Violation_Code order by TicketsIssued desc
limit 2;
```

BIN3

```
select Violation_Code,count(*) TicketsIssued from park_viol_part_view where
Violation_Time_bin == 3 group by Violation_Code order by TicketsIssued desc
limit 2;
```

BIN4

```
select Violation_Code,count(*) TicketsIssued from park_viol_part_view where
Violation_Time_bin == 4 group by Violation_Code order by TicketsIssued desc
limit 2;
```

BIN5

```
select Violation_Code,count(*) TicketsIssued from park_viol_part_view where
Violation_Time_bin == 5 group by Violation_Code order by TicketsIssued desc
limit 3;
```

BIN6

```
select Violation_Code,count(*) TicketsIssued from park_viol_part_view where
Violation_Time_bin == 6 group by Violation_Code order by TicketsIssued desc
limit 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)

```
select Violation_Time_bin, count(*) TicketsIssued from park_viol_part_view where
Violation_Code in (21,36,37,38) group by Violation_Time_bin order by
TicketsIssued desc limit 3;
```

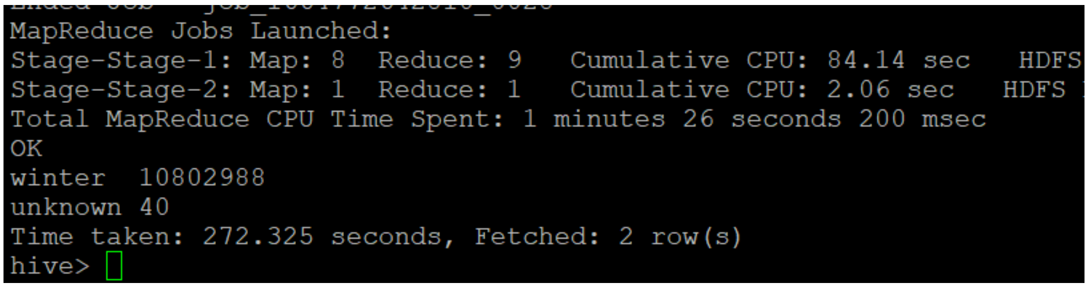
8.) Let's try and find some seasonality in this data

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, March);

```
Summer(June, July, August); Fall(September, October, November);
Winter(December, January, February))
```

```
create view tickets_issued_view_part partitioned on (Violation_Code) as
select Issuer_Precinct,
case
when MONTH(Issue_Date) between 03 and 05 then 'spring'
when MONTH(Issue_Date) between 06 and 08 then 'summer'
when MONTH(Issue_Date) between 09 and 11 then 'autumn'
when MONTH(Issue_Date) in (1,2,12) then 'winter'
else 'unknown' end  as season,Violation_Code from
violations_parking;
```

```
select season, count(*) as TicketsIssued from tickets_issued_view_part group by
season order by TicketsIssued desc;
```



```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 8  Reduce: 9   Cumulative CPU: 84.14 sec   HDFS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.06 sec   HDFS
Total MapReduce CPU Time Spent: 1 minutes 26 seconds 200 msec
OK
winter  10802988
unknown 40
Time taken: 272.325 seconds, Fetched: 2 row(s)
hive>
```

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

```
select Violation_Code, count(*) as TicketsIssued from tickets_issued_view_part
where season = 'spring' group by Violation_Code order by TicketsIssued desc
limit 3;
```

select Violation_Code, count(*) as TicketsIssued from tickets_issued_view_part where season = 'spring' group by Violation_Code order by TicketsIssued desc limit 3;

select Violation_Code, count(*) as TicketsIssued from tickets_issued_view_part where season = 'autumn' group by Violation_Code order by TicketsIssued desc limit 3;

select Violation_Code, count(*) as TicketsIssued from tickets_issued_view_part where season = 'winter' group by Violation_Code order by TicketsIssued desc limit 3;