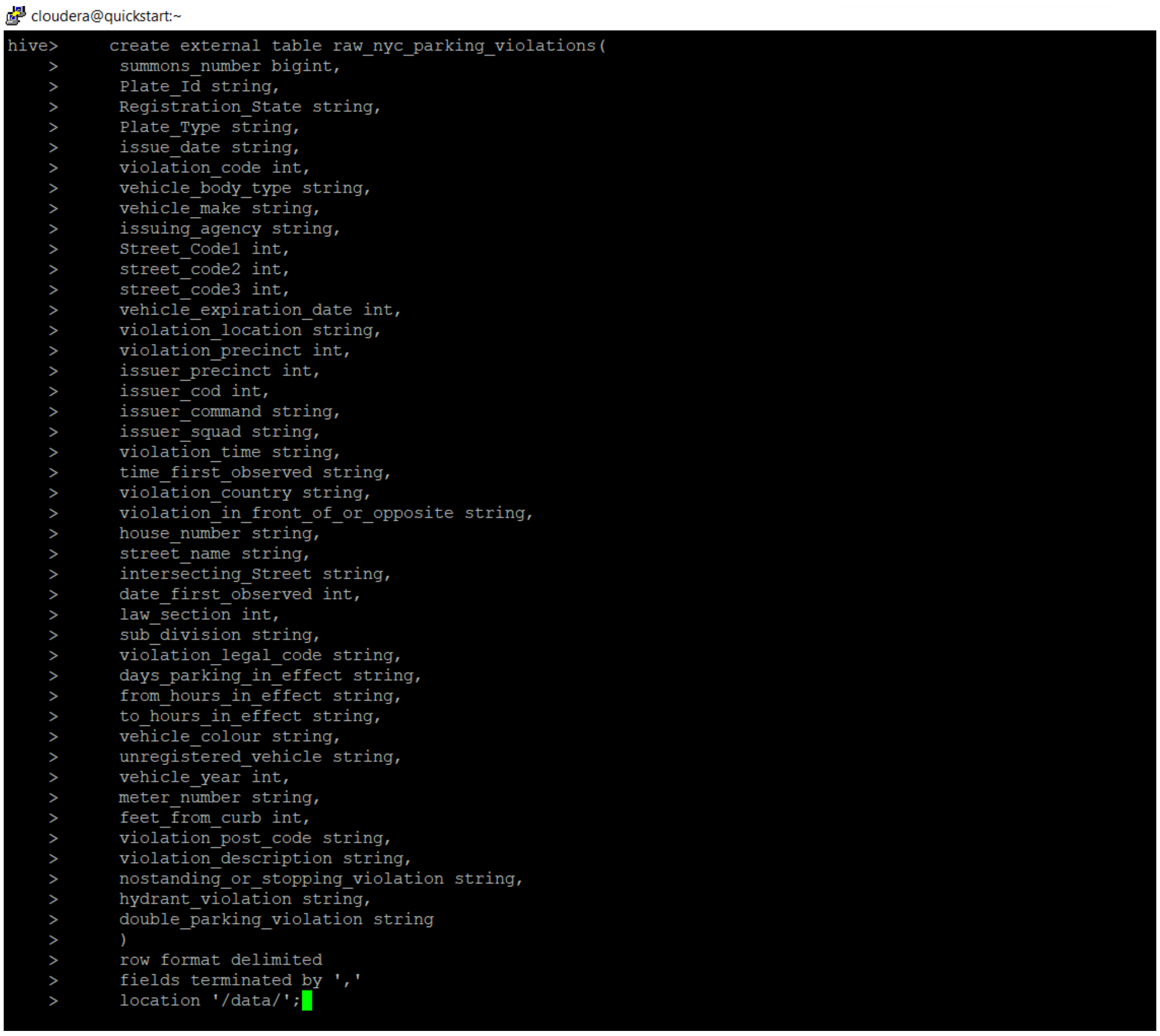
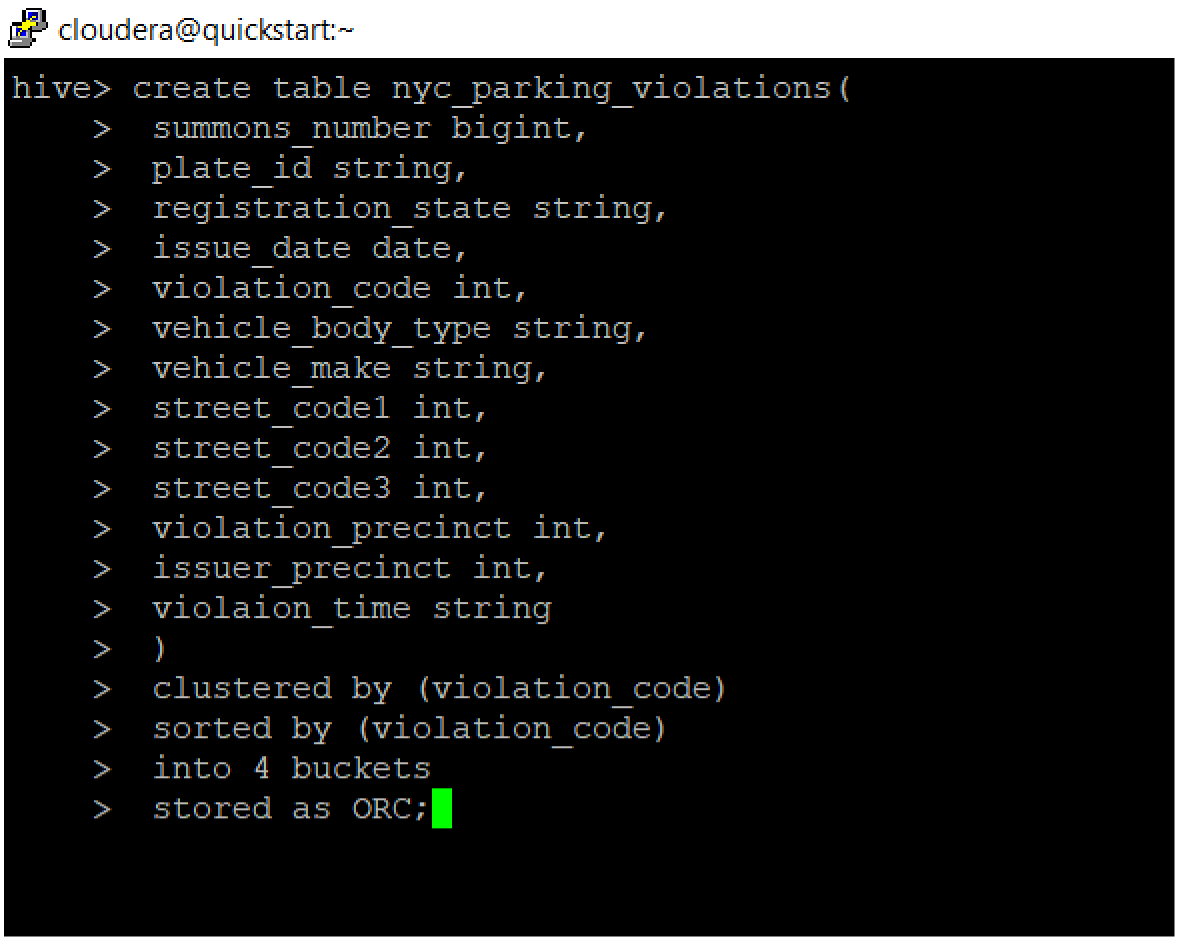


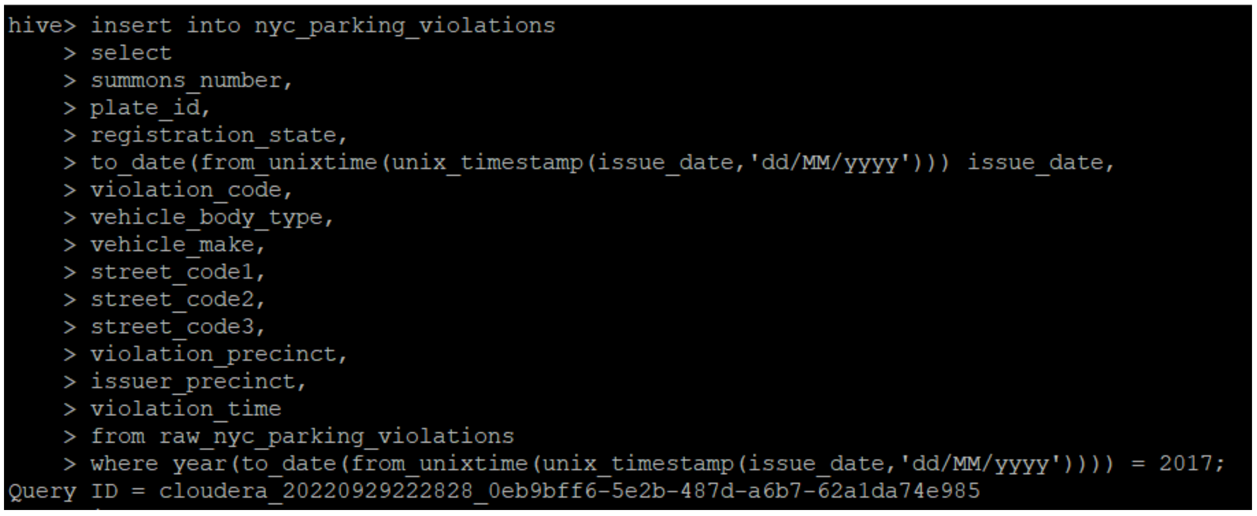
cloudera@quickstart:~

```
hive>     create external table raw_nyc_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_date int,
    >      violation_location string,
    >      violation_precinct int,
    >      issuer_precinct int,
    >      issuer_cod int,
    >      issuer_command string,
    >      issuer_squad string,
    >      violation_time string,
    >      time_first_observed string,
    >      violation_country 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_colour string,
    >      unregistered_vehicle string,
    >      vehicle_year int,
    >      meter_number string,
    >      feet_from_curb int,
    >      violation_post_code string,
    >      violation_description string,
    >      nostanding_or_stopping_violation string,
    >      hydrant_violation string,
    >      double_parking_violation string
    >      )
    >      row format delimited
    >      fields terminated by ','
    >      location '/data/';
```



cloudera@quickstart:~

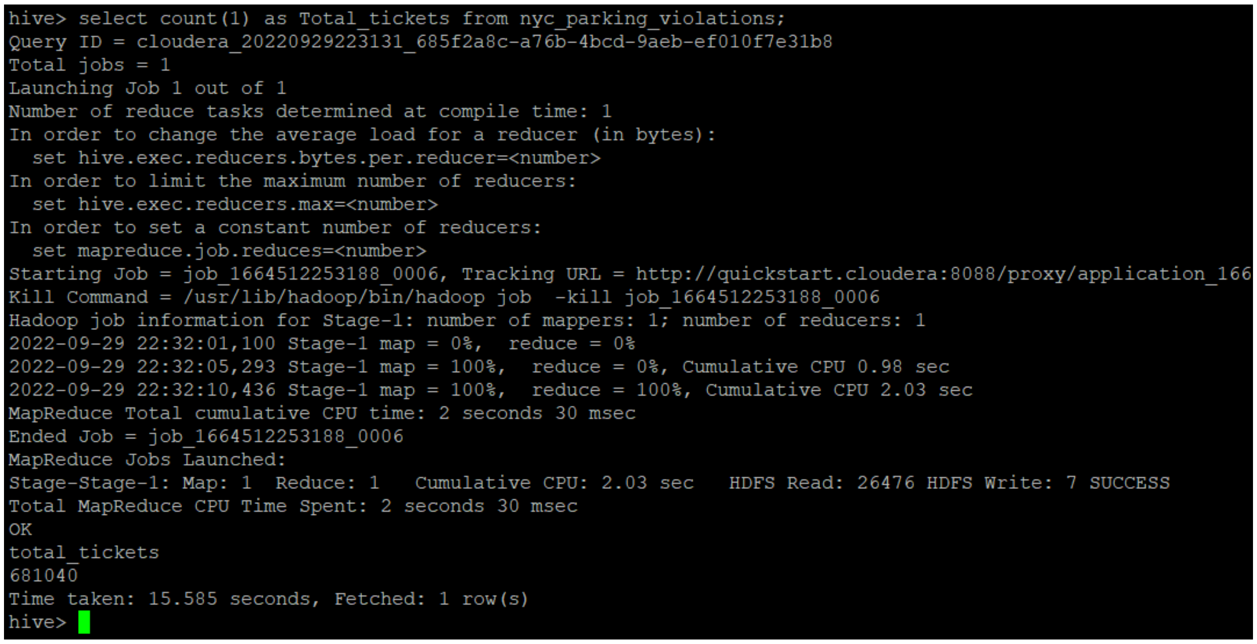
```
hive> create table nyc_parking_violations(
    >  summons_number bigint,
    >  plate_id string,
    >  registration_state string,
    >  issue_date date,
    >  violation_code int,
    >  vehicle_body_type string,
    >  vehicle_make string,
    >  street_code1 int,
    >  street_code2 int,
    >  street_code3 int,
    >  violation_precinct int,
    >  issuer_precinct int,
    >  violaion_time string
    >  )
    >  clustered by (violation_code)
    >  sorted by (violation_code)
    >  into 4 buckets
    >  stored as ORC;
```



```
hive> insert into nyc_parking_violations
    > select
    > summons_number,
    > plate_id,
    > registration_state,
    > to_date(from_unixtime(unix_timestamp(issue_date,'dd/MM/yyyy'))) issue_date,
    > violation_code,
    > vehicle_body_type,
    > vehicle_make,
    > street_code1,
    > street_code2,
    > street_code3,
    > violation_precinct,
    > issuer_precinct,
    > violation_time
    > from raw_nyc_parking_violations
    > where year(to_date(from_unixtime(unix_timestamp(issue_date,'dd/MM/yyyy')))) = 2017;
Query ID = cloudera_20220929222828_0eb9bff6-5e2b-487d-a6b7-62a1da74e985
```

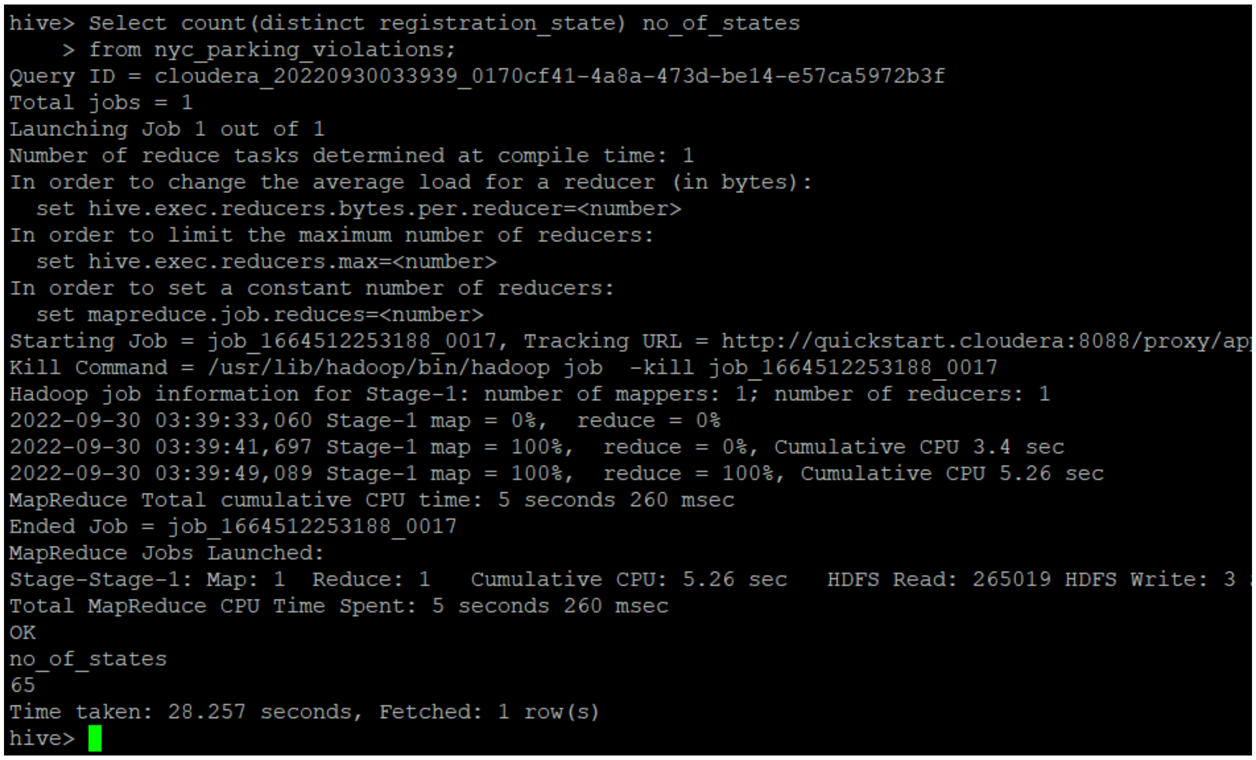
Part-I: Examine the data

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



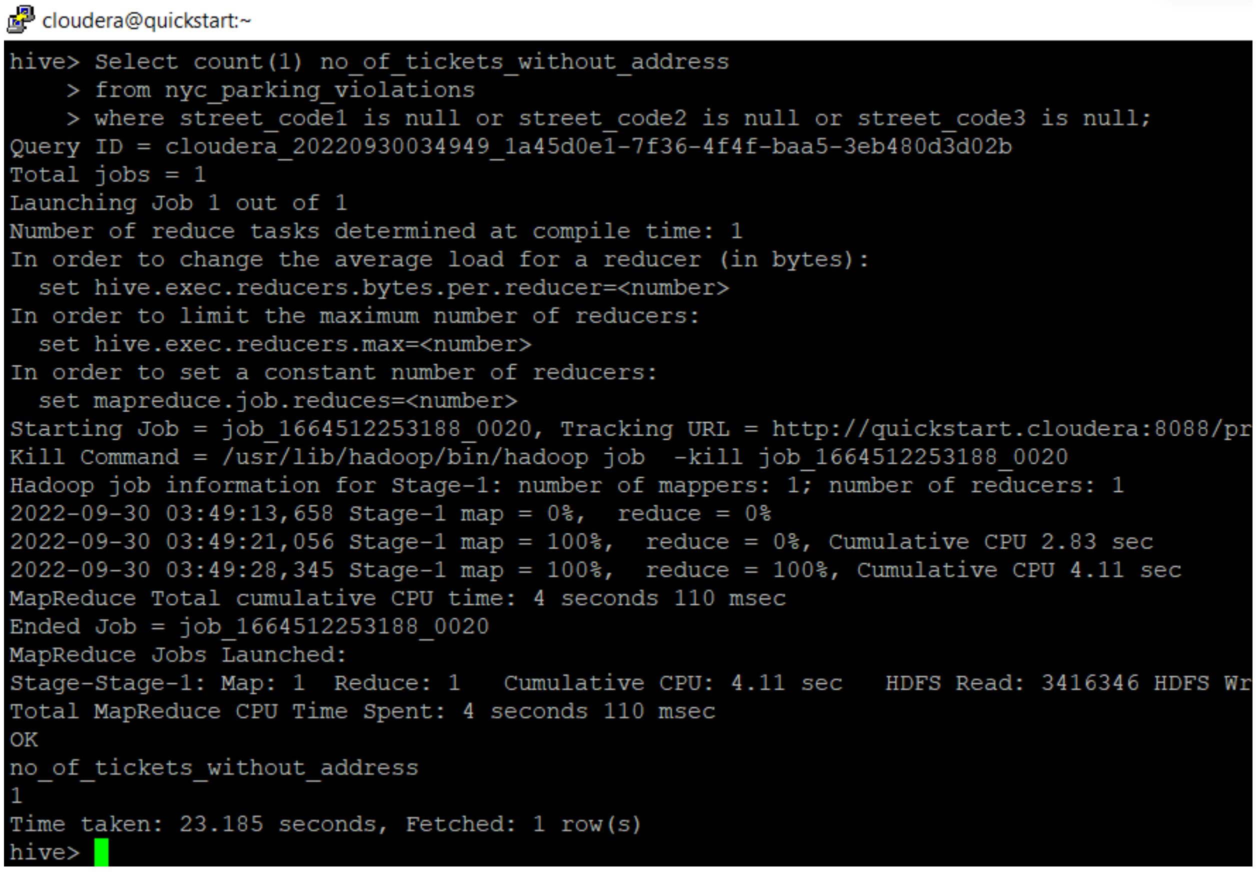
```
hive> select count(1) as Total_tickets from nyc_parking_violations;
Query ID = cloudera_20220929223131_685f2a8c-a76b-4bcd-9aeb-ef010f7e31b8
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1664512253188_0006, Tracking URL = http://quickstart.cloudera:8088/proxy/application_166
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1664512253188_0006
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-09-29 22:32:01,100 Stage-1 map = 0%,  reduce = 0%
2022-09-29 22:32:05,293 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.98 sec
2022-09-29 22:32:10,436 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.03 sec
MapReduce Total cumulative CPU time: 2 seconds 30 msec
Ended Job = job_1664512253188_0006
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.03 sec   HDFS Read: 26476 HDFS Write: 7 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 30 msec
OK
total_tickets
681040
Time taken: 15.585 seconds, Fetched: 1 row(s)
hive>
```

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



```
hive> Select count(distinct registration_state) no_of_states
    > from nyc_parking_violations;
Query ID = cloudera_20220930033939_0170cf41-4a8a-473d-be14-e57ca5972b3f
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1664512253188_0017, Tracking URL = http://quickstart.cloudera:8088/proxy/ap
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1664512253188_0017
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-09-30 03:39:33,060 Stage-1 map = 0%,  reduce = 0%
2022-09-30 03:39:41,697 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.4 sec
2022-09-30 03:39:49,089 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.26 sec
MapReduce Total cumulative CPU time: 5 seconds 260 msec
Ended Job = job_1664512253188_0017
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.26 sec   HDFS Read: 265019 HDFS Write: 3
Total MapReduce CPU Time Spent: 5 seconds 260 msec
OK
no_of_states
65
Time taken: 28.257 seconds, Fetched: 1 row(s)
hive>
```

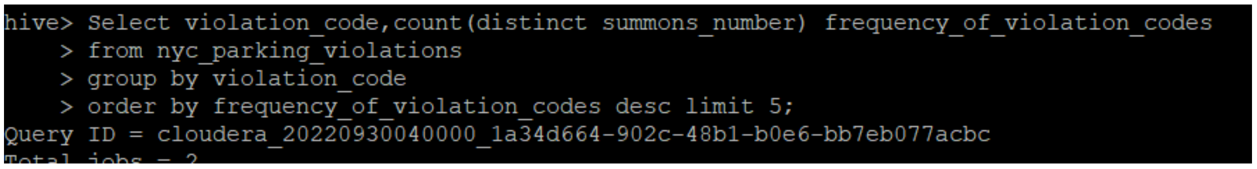
3.) Some parking tickets don't have addresses on them, which is cause for concern. Find out how many such tickets there are(i.e. tickets where either "Street Code 1" or "Street Code 2" or "Street Code 3" is empty)


cloudera@quickstart:~

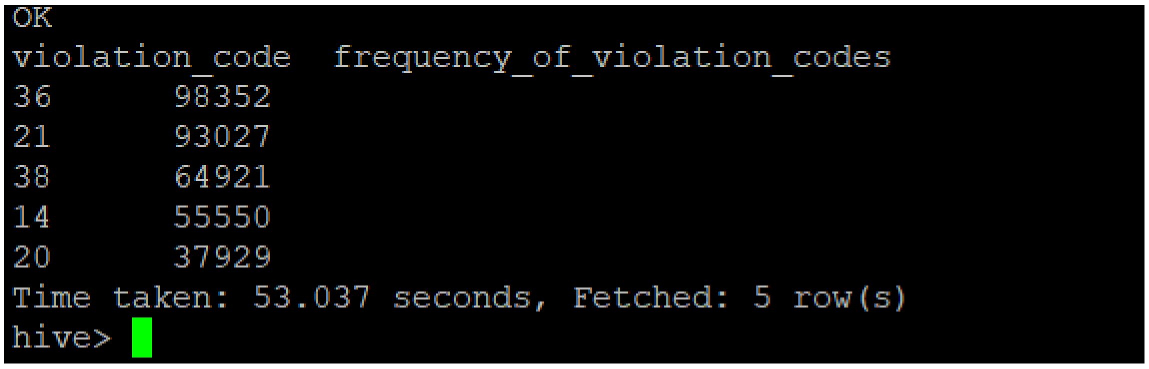
```
hive> Select count(1) no_of_tickets_without_address
    > from nyc_parking_violations
    > where street_code1 is null or street_code2 is null or street_code3 is null;
Query ID = cloudera_20220930034949_1a45d0e1-7f36-4f4f-baa5-3eb480d3d02b
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1664512253188_0020, Tracking URL = http://quickstart.cloudera:8088/pr
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1664512253188_0020
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-09-30 03:49:13,658 Stage-1 map = 0%,  reduce = 0%
2022-09-30 03:49:21,056 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.83 sec
2022-09-30 03:49:28,345 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.11 sec
MapReduce Total cumulative CPU time: 4 seconds 110 msec
Ended Job = job_1664512253188_0020
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 4.11 sec   HDFS Read: 3416346 HDFS Wr
Total MapReduce CPU Time Spent: 4 seconds 110 msec
OK
no_of_tickets_without_address
1
Time taken: 23.185 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)

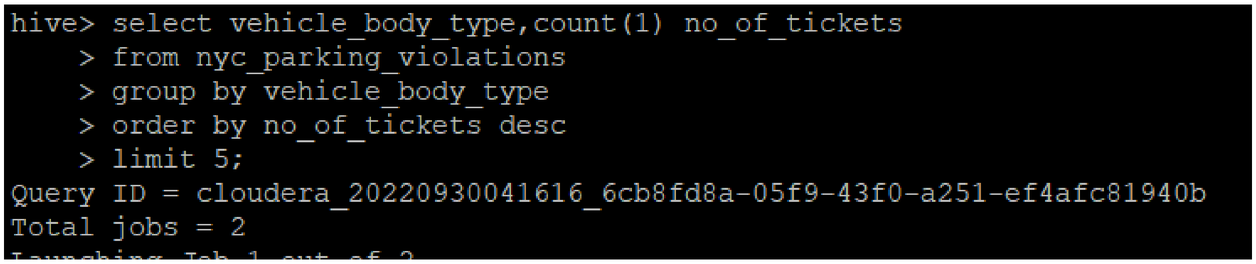


```
hive> Select violation_code,count(distinct summons_number) frequency_of_violation_codes
    > from nyc_parking_violations
    > group by violation_code
    > order by frequency_of_violation_codes desc limit 5;
Query ID = cloudera_20220930040000_1a34d664-902c-48b1-b0e6-bb7eb077acbc
```

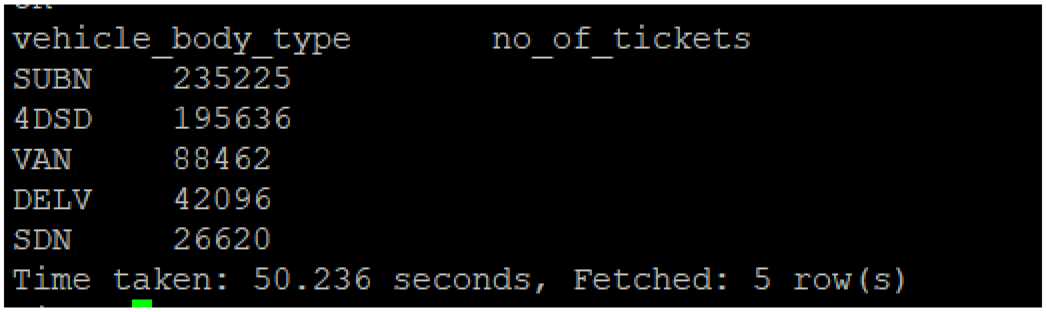


```
OK
violation_code  frequency_of_violation_codes
36      98352
21      93027
38      64921
14      55550
20      37929
Time taken: 53.037 seconds, Fetched: 5 row(s)
hive>
```

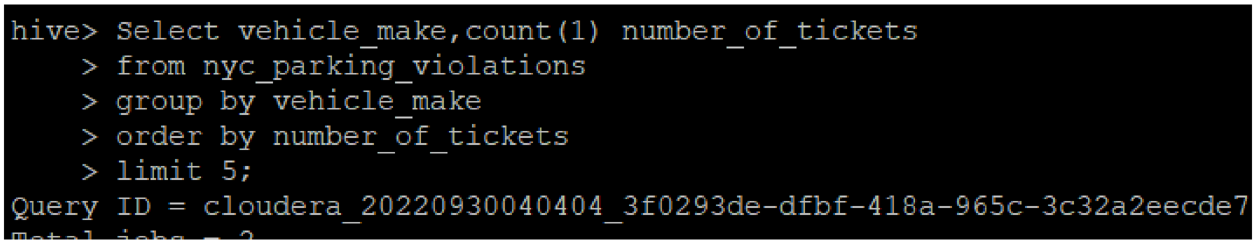
2.) How often does each vehicle body type get a parking ticket? How about the vehicle make? (find the top 5 for both)



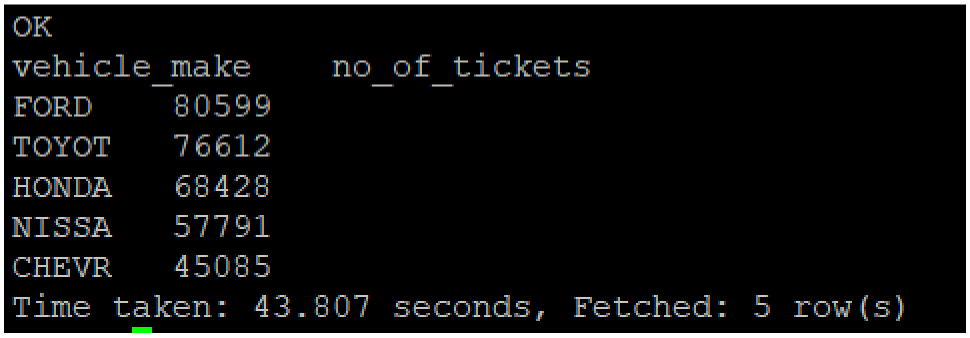
```
hive> select vehicle_body_type,count(1) no_of_tickets
    > from nyc_parking_violations
    > group by vehicle_body_type
    > order by no_of_tickets desc
    > limit 5;
Query ID = cloudera_20220930041616_6cb8fd8a-05f9-43f0-a251-ef4afc81940b
Total jobs = 2
```



```
vehicle_body_type       no_of_tickets
SUBN    235225
4DSD    195636
VAN     88462
DELV    42096
SDN     26620
Time taken: 50.236 seconds, Fetched: 5 row(s)
```



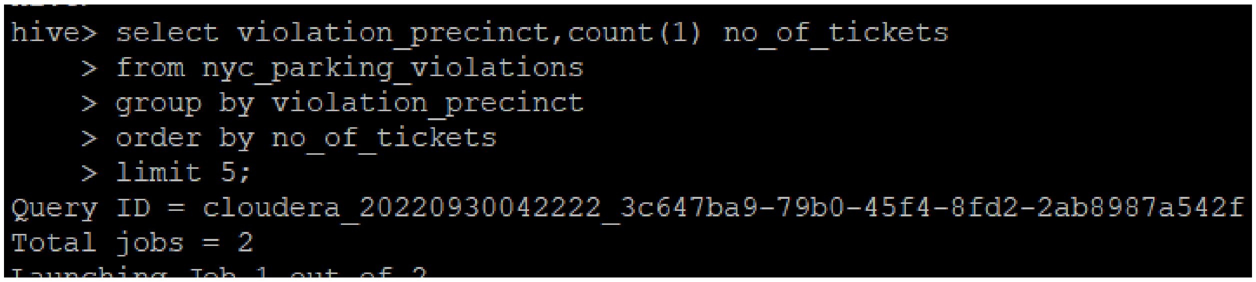
```
hive> Select vehicle_make,count(1) number_of_tickets
    > from nyc_parking_violations
    > group by vehicle_make
    > order by number_of_tickets
    > limit 5;
Query ID = cloudera_20220930040404_3f0293de-dfbf-418a-965c-3c32a2eecde7
```



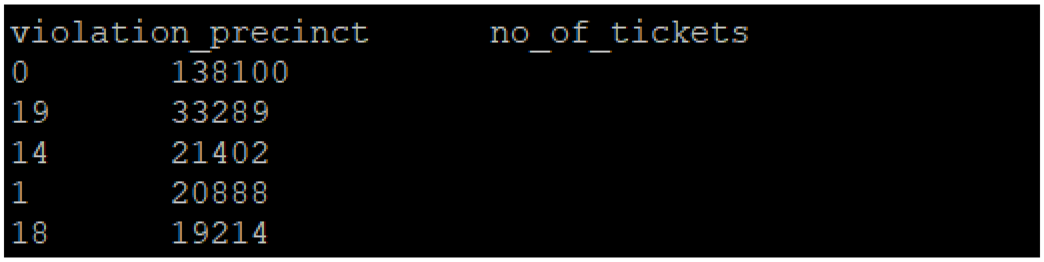
```
OK
vehicle_make    no_of_tickets
FORD    80599
TOYOT   76612
HONDA   68428
NISSA   57791
CHEVR   45085
Time taken: 43.807 seconds, Fetched: 5 row(s)
```

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)

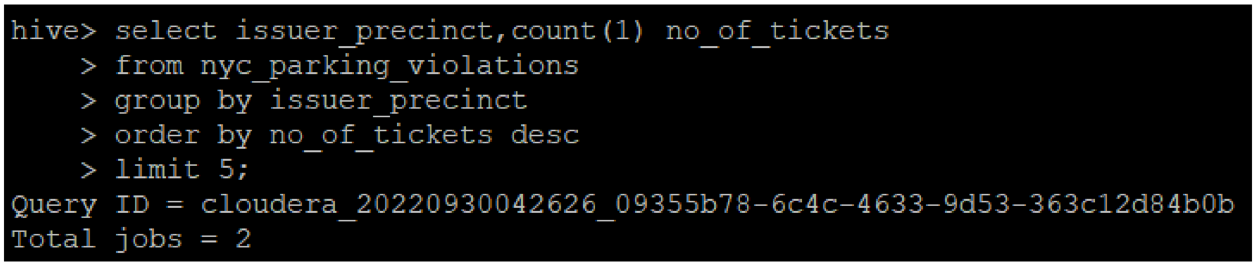


```
hive> select violation_precinct,count(1) no_of_tickets
    > from nyc_parking_violations
    > group by violation_precinct
    > order by no_of_tickets
    > limit 5;
Query ID = cloudera_20220930042222_3c647ba9-79b0-45f4-8fd2-2ab8987a542f
Total jobs = 2
```

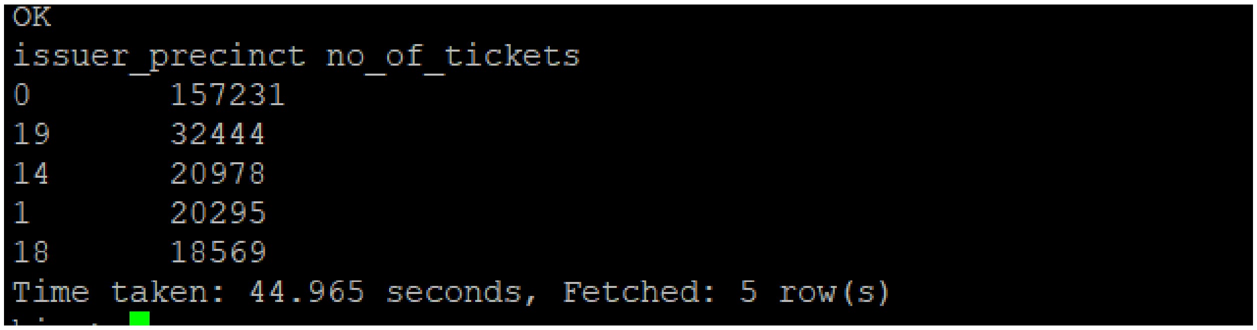


```
violation_precinct      no_of_tickets
0       138100
19      33289
14      21402
1       20888
18      19214
```

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

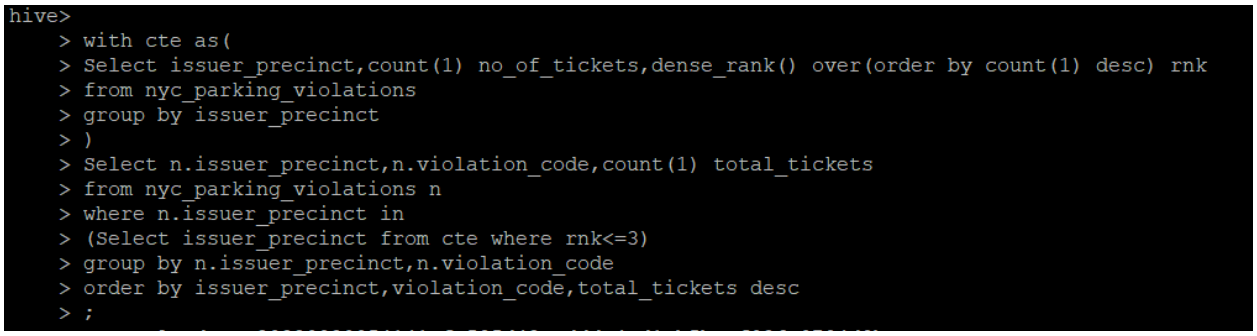


```
hive> select issuer_precinct,count(1) no_of_tickets
    > from nyc_parking_violations
    > group by issuer_precinct
    > order by no_of_tickets desc
    > limit 5;
Query ID = cloudera_20220930042626_09355b78-6c4c-4633-9d53-363c12d84b0b
Total jobs = 2
```

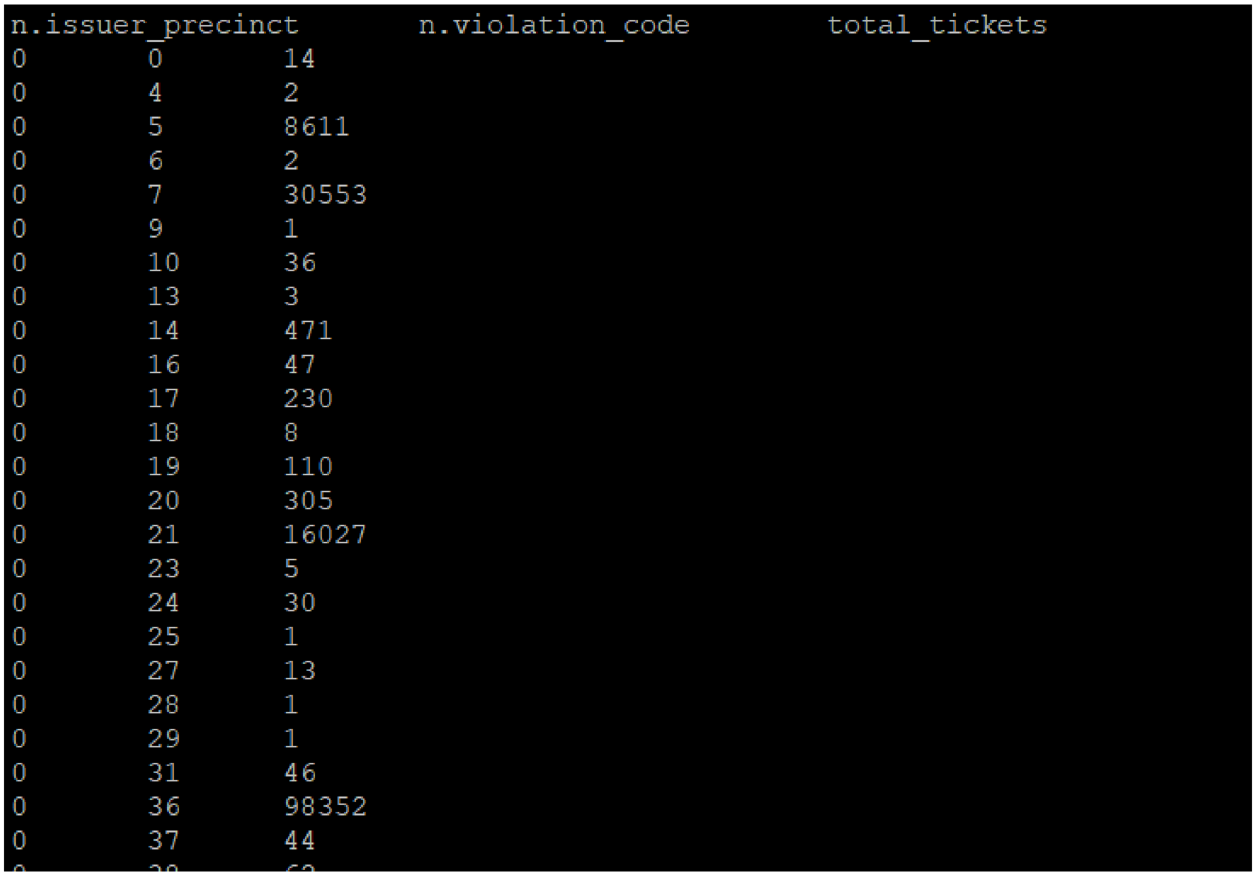


```
OK
issuer_precinct no_of_tickets
0       157231
19      32444
14      20978
1       20295
18      18569
Time taken: 44.965 seconds, Fetched: 5 row(s)
```

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?



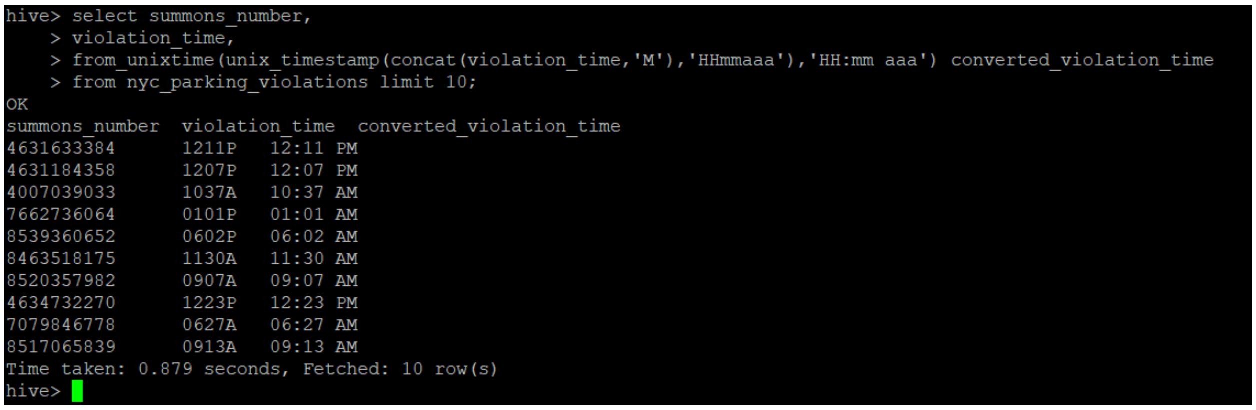
```
hive>
    > with cte as(
    > Select issuer_precinct,count(1) no_of_tickets,dense_rank() over(order by count(1) desc) rnk
    > from nyc_parking_violations
    > group by issuer_precinct
    > )
    > Select n.issuer_precinct,n.violation_code,count(1) total_tickets
    > from nyc_parking_violations n
    > where n.issuer_precinct in
    > (Select issuer_precinct from cte where rnk<=3)
    > group by n.issuer_precinct,n.violation_code
    > order by issuer_precinct,violation_code,total_tickets desc
    > ;
```



```
n.issuer_precinct       n.violation_code        total_tickets
0       0       14
0       4       2
0       5       8611
0       6       2
0       7       30553
0       9       1
0       10      36
0       13      3
0       14      471
0       16      47
0       17      230
0       18      8
0       19      110
0       20      305
0       21      16027
0       23      5
0       24      30
0       25      1
0       27      13
0       28      1
0       29      1
0       31      46
0       36      98352
0       37      44
```

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

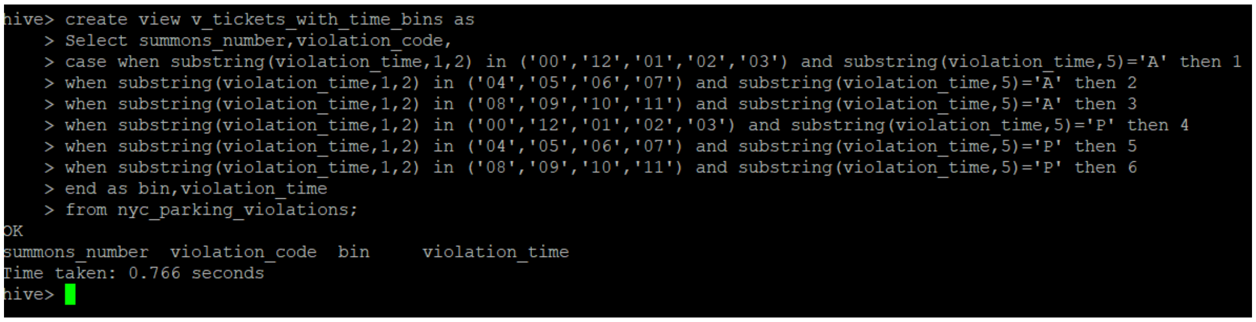
select summons_number,
violation_time,
from_unixtime(unix_timestamp(concat(violation_time,'M'),'HHmmaaa'),'HH:mm aaa')
converted_violation_time
from nyc_parking_violations limit 10;



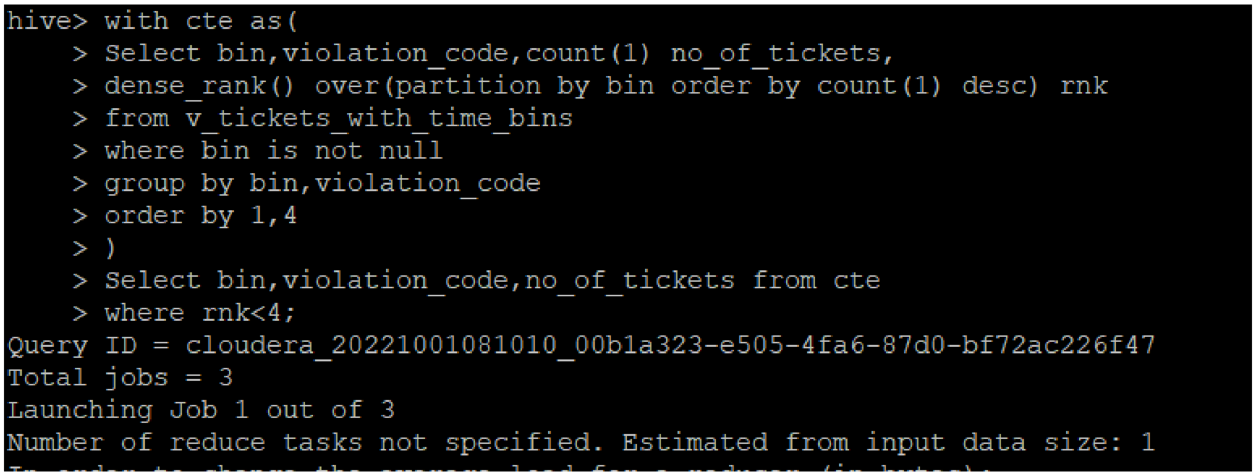
```
hive> select summons_number,
    > violation_time,
    > from_unixtime(unix_timestamp(concat(violation_time,'M'),'HHmmaaa'),'HH:mm aaa') converted_violation_time
    > from nyc_parking_violations limit 10;
OK
summons_number  violation_time  converted_violation_time
4631633384      1211P   12:11 PM
4631184358      1207P   12:07 PM
4007039033      1037A   10:37 AM
7662736064      0101P   01:01 AM
8539360652      0602P   06:02 AM
8463518175      1130A   11:30 AM
8520357982      0907A   09:07 AM
4634732270      1223P   12:23 PM
7079846778      0627A   06:27 AM
8517065839      0913A   09:13 AM
Time taken: 0.879 seconds, Fetched: 10 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.

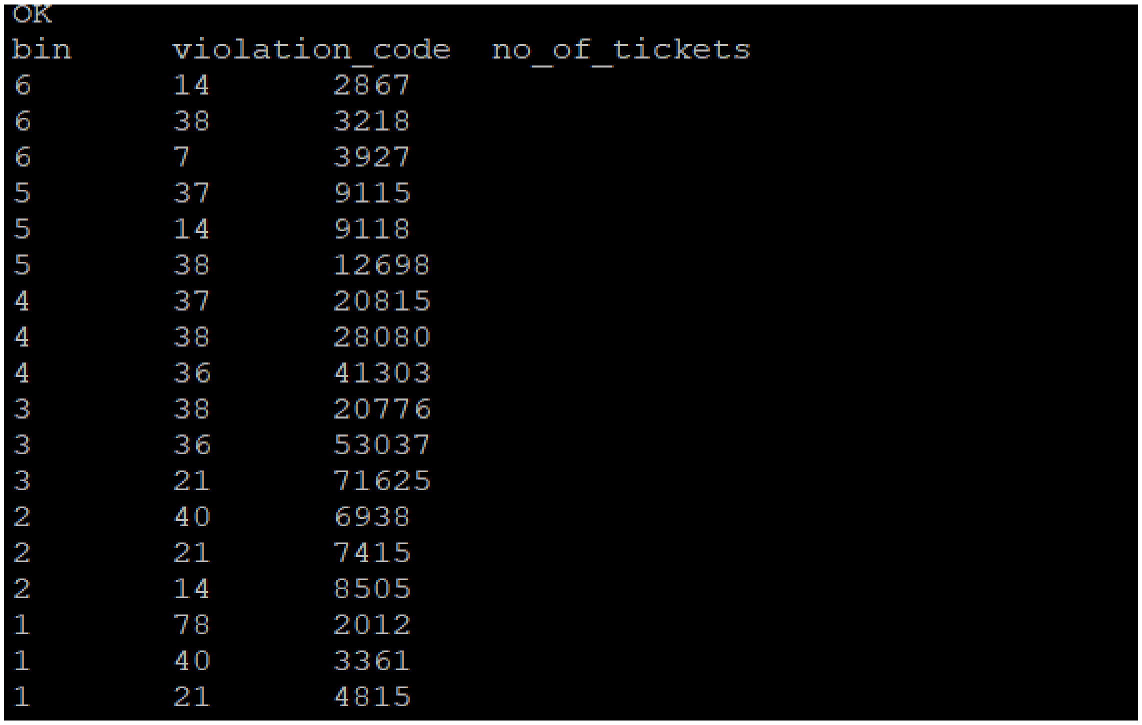
Bin	Time Interval
1	12:00 AM to 04:00 AM
2	04:00 AM to 08:00 AM
3	08:00 AM to 12:00 PM
4	12:00 PM to 04:00 PM
5	04:00 PM to 08:00 PM
6	08:00 PM to 12:00 AM



```
hive> create view v_tickets_with_time_bins as
    > Select summons_number,violation_code,
    > case when substring(violation_time,1,2) in ('00','12','01','02','03') and substring(violation_time,5)='A' then 1
    > when substring(violation_time,1,2) in ('04','05','06','07') and substring(violation_time,5)='A' then 2
    > when substring(violation_time,1,2) in ('08','09','10','11') and substring(violation_time,5)='A' then 3
    > when substring(violation_time,1,2) in ('00','12','01','02','03') and substring(violation_time,5)='P' then 4
    > when substring(violation_time,1,2) in ('04','05','06','07') and substring(violation_time,5)='P' then 5
    > when substring(violation_time,1,2) in ('08','09','10','11') and substring(violation_time,5)='P' then 6
    > end as bin,violation_time
    > from nyc_parking_violations;
OK
summons_number  violation_code  bin     violation_time
Time taken: 0.766 seconds
hive>
```

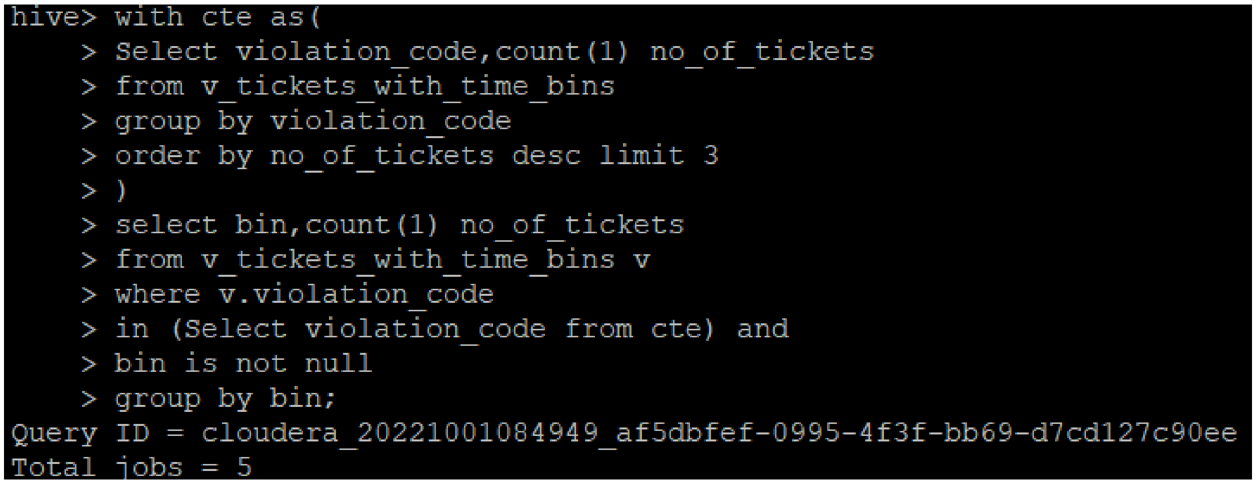


```
hive> with cte as(
    > Select bin,violation_code,count(1) no_of_tickets,
    > dense_rank() over(partition by bin order by count(1) desc) rnk
    > from v_tickets_with_time_bins
    > where bin is not null
    > group by bin,violation_code
    > order by 1,4
    > )
    > Select bin,violation_code,no_of_tickets from cte
    > where rnk<4;
Query ID = cloudera_20221001081010_00b1a323-e505-4fa6-87d0-bf72ac226f47
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks not specified. Estimated from input data size: 1
```

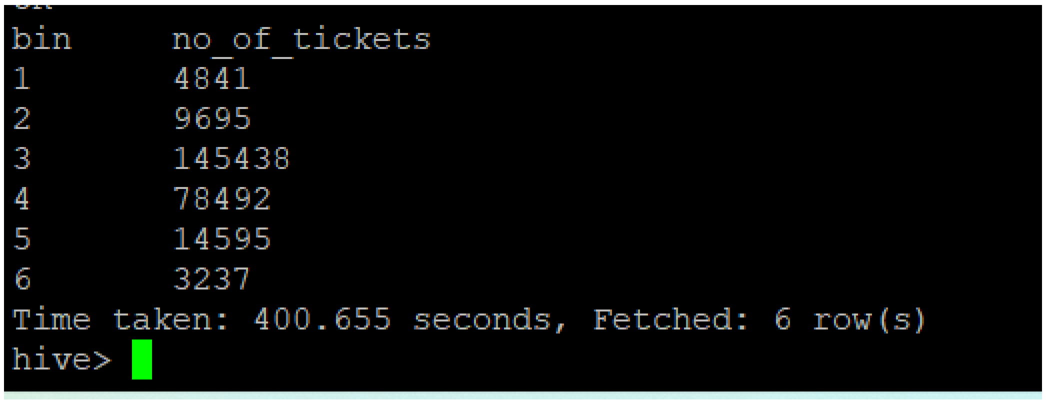


```
OK
bin     violation_code  no_of_tickets
6       14      2867
6       38      3218
6       7       3927
5       37      9115
5       14      9118
5       38      12698
4       37      20815
4       38      28080
4       36      41303
3       38      20776
3       36      53037
3       21      71625
2       40      6938
2       21      7415
2       14      8505
1       78      2012
1       40      3361
1       21      4815
```

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

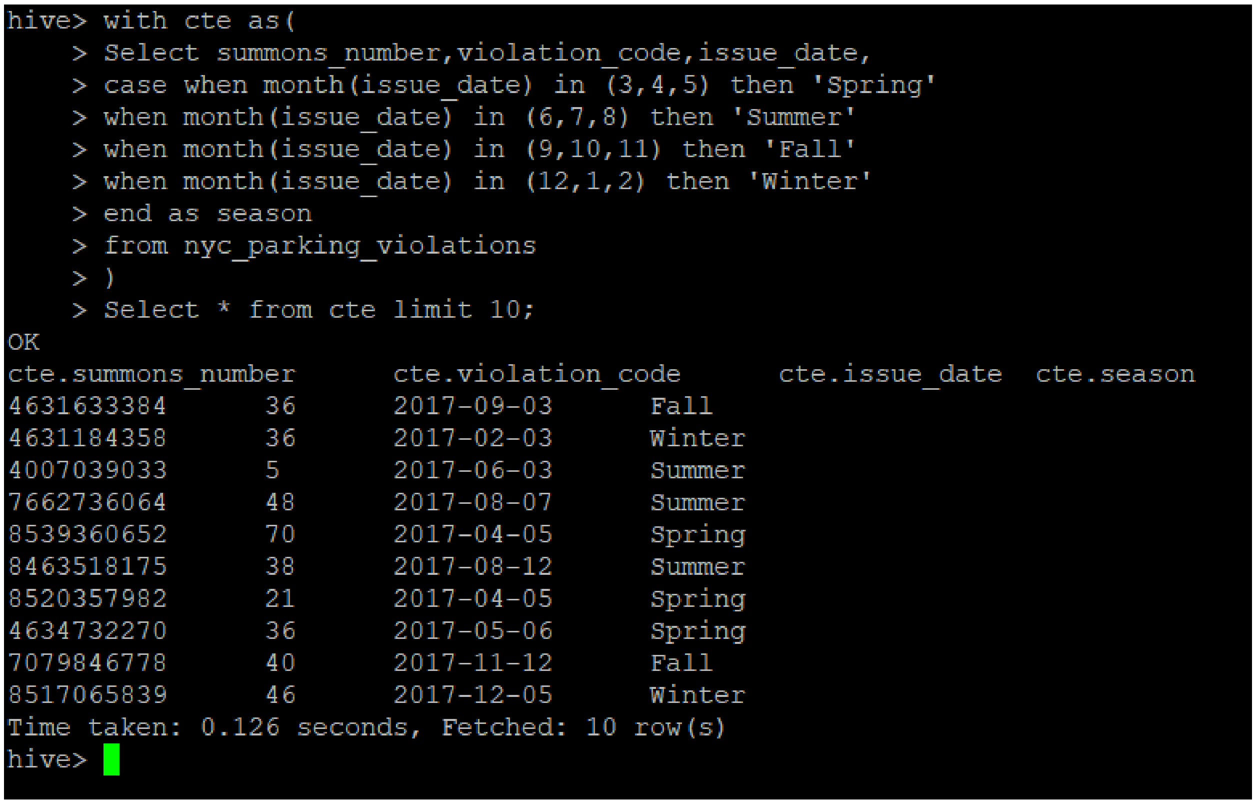


```
hive> with cte as(
    > Select violation_code,count(1) no_of_tickets
    > from v_tickets_with_time_bins
    > group by violation_code
    > order by no_of_tickets desc limit 3
    > )
    > select bin,count(1) no_of_tickets
    > from v_tickets_with_time_bins v
    > where v.violation_code
    > in (Select violation_code from cte) and
    > bin is not null
    > group by bin;
Query ID = cloudera_20221001084949_af5dbfef-0995-4f3f-bb69-d7cd127c90ee
Total jobs = 5
```



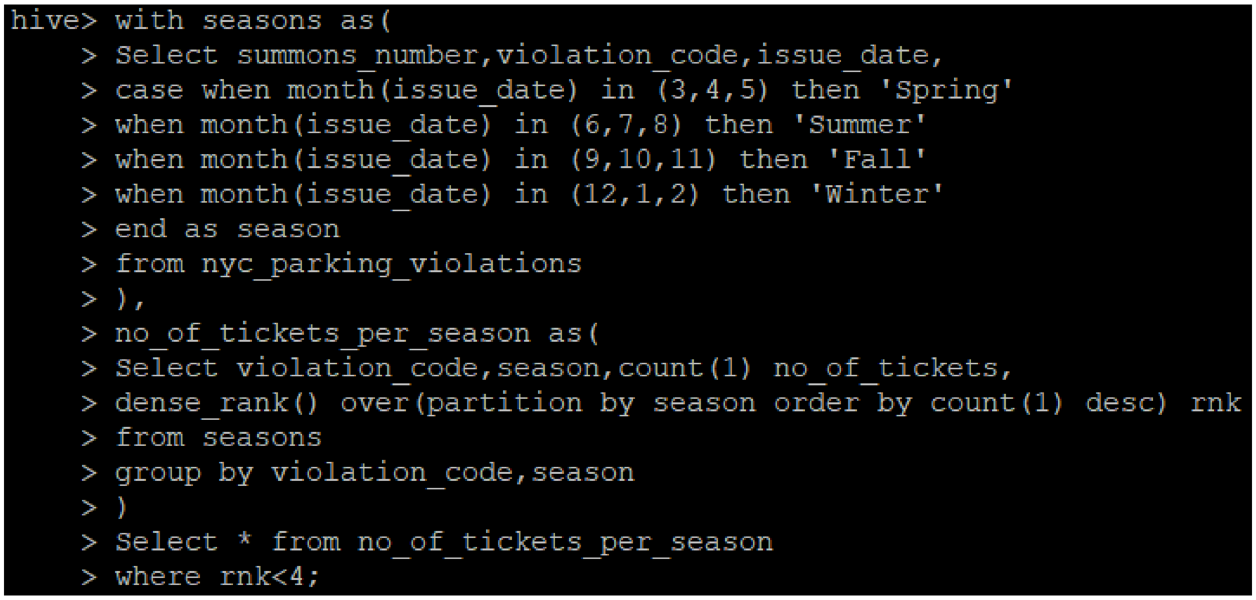
```
bin     no_of_tickets
1       4841
2       9695
3       145438
4       78492
5       14595
6       3237
Time taken: 400.655 seconds, Fetched: 6 row(s)
hive>
```

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))

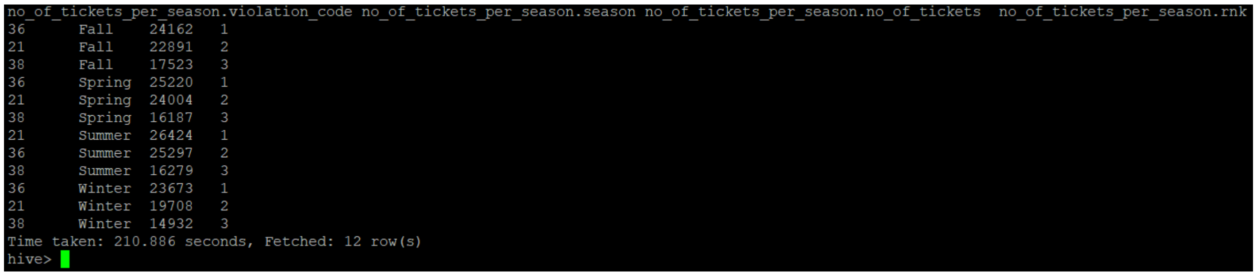


```
hive> with cte as(
    > Select summons_number,violation_code,issue_date,
    > case when month(issue_date) in (3,4,5) then 'Spring'
    > when month(issue_date) in (6,7,8) then 'Summer'
    > when month(issue_date) in (9,10,11) then 'Fall'
    > when month(issue_date) in (12,1,2) then 'Winter'
    > end as season
    > from nyc_parking_violations
    > )
    > Select * from cte limit 10;
OK
cte.summons_number      cte.violation_code      cte.issue_date  cte.season
4631633384      36      2017-09-03      Fall
4631184358      36      2017-02-03      Winter
4007039033      5       2017-06-03      Summer
7662736064      48      2017-08-07      Summer
8539360652      70      2017-04-05      Spring
8463518175      38      2017-08-12      Summer
8520357982      21      2017-04-05      Spring
4634732270      36      2017-05-06      Spring
7079846778      40      2017-11-12      Fall
8517065839      46      2017-12-05      Winter
Time taken: 0.126 seconds, Fetched: 10 row(s)
hive>
```

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



```
hive> with seasons as(
    > Select summons_number,violation_code,issue_date,
    > case when month(issue_date) in (3,4,5) then 'Spring'
    > when month(issue_date) in (6,7,8) then 'Summer'
    > when month(issue_date) in (9,10,11) then 'Fall'
    > when month(issue_date) in (12,1,2) then 'Winter'
    > end as season
    > from nyc_parking_violations
    > ),
    > no_of_tickets_per_season as(
    > Select violation_code,season,count(1) no_of_tickets,
    > dense_rank() over(partition by season order by count(1) desc) rnk
    > from seasons
    > group by violation_code,season
    > )
    > Select * from no_of_tickets_per_season
    > where rnk<4;
```



```
no_of_tickets_per_season.violation_code no_of_tickets_per_season.season no_of_tickets_per_season.no_of_tickets  no_of_tickets_per_season.rnk
36      Fall    24162   1
21      Fall    22891   2
38      Fall    17523   3
36      Spring  25220   1
21      Spring  24004   2
38      Spring  16187   3
21      Summer  26424   1
36      Summer  25297   2
38      Summer  16279   3
36      Winter  23673   1
21      Winter  19708   2
38      Winter  14932   3
Time taken: 210.886 seconds, Fetched: 12 row(s)
hive> 
```