

Name: Arun Kumar N  
Student ID : 2017CBDE030  
Hive Assignment

Note: Comments denoted with Dark Blue color

**Steps performed in Hive Assignment:**

1. **Copied Parking\_Violations\_Issued\_-\_Fiscal\_Year\_2017.csv** from the S3 bucket using the below command

```
Aws s3 cp s3://hiveassignmentdatabde/Parking_Violations_Issued_-_Fiscal_Year_2017.csv to  
s3://my-bucket-name (Command to copy the dataset to our own S3 bucket)
```

2. **Once the dataset is copied into S3 bucket, created an external table with the below schema**

```
CREATE EXTERNAL TABLE IF NOT EXISTS Parking_Violation_s3(
```

```
`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_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` 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 ',' location 's3a://hivetestarun/'  
TBLPROPERTIES('skip.header.line.count'='1');  
(creating external table using the schema in dataset with row delimited by , and location is to  
fetch data from the s3 bucket and tbl properties is to remove header line)
```

**3. Once the above command is executed, just checking the existence of data inside the table**

```
Select * from parking_violation_2017 limit 10;
```

*(Command to check the data in the table limit is used to display only the n number of values in the output)*

**4. Created an external table and inserted only the 2017 data using the following command:**

```
CREATE TABLE Violation_2017(  
`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_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` 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);
```

**Insert Command:**

```
INSERT into violation_2017  
SELECT * FROM parking_violation_s3 where  
year(from_unixtime(unix_timestamp(issue_date,'MM/dd/yyyy')))= '2017';
```

*(creating external table using the schema in dataset, this table is created only to store 2017 data. It is basically a data cleaning activity, removing all the unwanted data from the previous table and storing only the required data in the current table. Using from\_unixtime function, which stores into the table only if the year equals 2017)*

**5. Created external table and stored it in an orc format**

```
CREATE TABLE Violation_2017_qrc(  
`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_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` 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) STORED AS ORC;
```

**Insert Command:**

```
INSERT into violation_2017_qrc  
Select * from violation_2017;
```

*(creating external table using the schema in dataset, this table will store the data in the ORC (Optimized row columnar) file format. This will store data in columnar format and stores data in form of stripes. Can access data faster)*

6. Then started performing the assignment tasks

**Part-I: Examine the data:**

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

**Query:** SELECT count(\*) as Tickets\_2017 from violation\_2017\_qrc;  
*(violation\_2017\_qrc contains only the 2017 data, getting count(\*) from the above table, will list the total number of tickets in the year 2017)*

Results (1) 🔍 ↗

	tickets_2017
1	5431903

Returned 5431903 rows.  
*(Total number of tickers for the year 2017 is 5431903)*

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

**Query:** SELECT count(DISTINCT registration\_state) as registration\_state\_unique from violation\_2017\_qrc;  
*(violation\_2017\_qrc contains only the 2017 data, getting count(distinct registration\_state) from the above table, will list the unique registration states in the year 2017)*

Results (1) 🔍 ↗

	registration_state_unique
1	65

Returned 65 rows.  
*(Total number of unique states which got parking tickets in the year 2017 is 65)*

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 )

**Query:** SELECT \* FROM violation\_2017\_qrc WHERE street\_code1 IS NULL or street\_code2 IS NULL or street\_code3 IS NULL; *(to get details of ticket)*

**Query:** SELECT count(\*) as Ticket\_count FROM violation\_2017\_qrc WHERE street\_code1 IS NULL or street\_code2 IS NULL or street\_code3 IS NULL; *(to get the count of ticket)*

```
158
159| SELECT * FROM violation_2017_qrc WHERE street_code1 IS NULL or street_code2 IS NULL or street_code3 IS NULL;
```

INFO : line uri to track the job: http://ip-10-0-0-229.ap-south-1.compute.internal:8088/proxy/application\_1529729642008\_0177  
INFO : Starting Job = job\_1529729642008\_0177, Tracking URL = http://ip-10-0-0-229.ap-south-1.compute.internal:8088/proxy/application\_1529729642008\_0177 [job\\_1529729642008\\_0177](#)  
INFO : Kill Command = /opt/cloudera/parcels/CDH-5.14.0-1.cdh5.14.0.p0.24/lib/hadoop/bin/hadoop job -kill job\_1529729642008\_0177  
INFO : Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0  
INFO : 2018-06-23 07:41:24,726 Stage-1 map = 0%, reduce = 0%

Query History Saved Queries Results (4)

☒ COLUMNS (44)

☒ violation\_2017\_qrc.summons\_number

☒ violation\_2017\_qrc.plate\_id

☒ violation\_2017\_qrc.registration\_state

☒ violation\_2017\_qrc.plate\_type

☒ violation\_2017\_qrc.issue\_date

☒ violation\_2017\_qrc.violation\_code

☒ violation\_2017\_qrc.vehicle\_body\_type

☒ violation\_2017\_qrc.vehicle\_make

☒ violation\_2017\_qrc.issuing\_agency

	violation_2017_qrc.summons_number	violation_2017_qrc.plate_id	violation_2017_qrc.registration_state	violation_2017_qrc.plate_type
1	1422528194	CO68446	CT	999
2	1418938300	BGN2846	NY	PAS
3	1416762061	41040PC	NY	APP
4	1416450725	GHN8197	NY	PAS

*( From the results we can find that, 4 parking tickers don't have addresses)*

## Part-II: Aggregation tasks

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

**Query:** SELECT violation\_code, count(\*) as Violation\_count FROM violation\_2017\_qrc  
GROUP BY violation\_code ORDER BY violation\_count desc LIMIT 5;

*(Here to find the top 5 frequency for violation codes, here listing the violation code and its count. Used GROUP BY(a costliest operation) it groups all the violation counts based on violation code and in the descending order listing the violation\_count. Here limit 5 is used to list only the 5 frequent violation\_codes)*

**Answer:**

Results (5) 🔍 📄

	violation_code	violation_count
1	21	768082
2	36	662765
3	38	542079
4	14	476660
5	20	319646

*(From the results, we can find that above mentioned 5 violation codes are most frequent in the NewYork City)*

### 2. i. How often does each vehicle body type get a parking ticket?

**Query:** SELECT vehicle\_body\_type, count(\*) as frequency from violation\_2017\_qrc GROUP BY vehicle\_body\_type ORDER BY frequency desc limit 5;

*(Here to find the top 5 vehicle body type who get parking ticket, here listing the vehicle body type and its count. Used GROUP BY(a costliest operation) it groups all the vehicle body type based on vehicle body type count and in the descending order listing the frequency of vehicle body type parking ticket count. Here limit 5 is used to list only the 5 most vehicle body type who gets parking tickets)*

**Result:**

Results (5) 🔍 📄

	vehicle_body_type	frequency
1	SUBN	1883953
2	4DSD	1547307
3	VAN	724025
4	DELV	358982
5	SDN	194197



(Above query list the top 5 vehicle body type who gets a parking ticket in the year 2017 most of the times)

ii. **How about the vehicle make?**

**Query:** SELECT vehicle\_make, count(\*) as Vehicle\_Make\_Frequency from violation\_2017\_qrc  
GROUP BY vehicle\_make ORDER BY Vehicle\_Make\_Frequency desc limit 5;

(Here to find the top 5 vehicle make,, here listing the vehicle make and its count. Used GROUP BY(a costliest operation) it groups all the vehicle make based on vehicle made count and in the descending order listing the frequency of vehicle made count. Here limit 5 is used to list only the 5 most vehicle make)

**Result:**

Results (5) 🔍 ↗

	vehicle_make	vehicle_make_frequency
1	FORD	636842
2	TOYOT	605290
3	HONDA	538884
4	NISSA	462017
5	CHEVR	356032

(Above query list the top 5 vehicle make and its vehicle make frequency count in the year 2017.)

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

i. **Violating Precincts**

**Query:** SELECT violation\_precinct, count(\*) as violation\_precinct\_count from  
violation\_2017\_qrc GROUP BY violation\_precinct ORDER BY violation\_precinct\_count  
desc limit 5;

(Here the query is to find the top 5 violation precinct and its count. Used group by and order by to list in descending order based on violation\_precinct\_count)

**Results:**

Results (5) 🔍 ↗

	violation_precinct	violation_precinct_count
1	0	925596
2	19	274443
3	14	203552
4	1	174702
5	18	169131

*(above query displays Top 5 violation precinct and its count)*

**ii. Issuer Precincts**

**Query:** SELECT issuer\_precinct, count(\*) as issuer\_precinct\_count from  
violation\_2017\_qrc GROUP BY issuer\_precinct ORDER BY issuer\_precinct\_count desc  
limit 5;

*(Here the query is to find the top 5 issuer precinct and its count. Used group by and order by to list in descending order based on issuer\_precinct\_count)*

**Result:**

Results (5) 🔍 ↗

	issuer_precinct	issuer_precinct_count
1	0	1078403
2	19	266959
3	14	200494
4	1	168740
5	18	162994

*(above query displays Top 5 issuer precinct and its count)*

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?

**Query:** SELECT issuer\_precinct, violation\_code, count(violation\_code) as violation\_count FROM violation\_2017\_qrc A INNER JOIN (SELECT issuer\_precinct, count(issuer\_precinct) as Issuer\_precinct\_count FROM violation\_2017\_qrc GROUP BY issuer\_precinct ORDER BY issuer\_precinct\_count desc limit 3)B ON A.issuer\_precinct = B.issuer\_precinct GROUP BY a.issuer\_precinct, a.violation\_code;

*(Here the query is find the violation code frequency across top 3 issue precinct. Joins were used in the above query and displays the violation code for the top 3 issuer precinct.)*

**Result:**

Results (100+) 🔍

	issuer_precinct	violation_code	violation_code_count
1	0	0	191
2	0	1	1
3	0	3	1
4	0	4	1
5	0	5	48076
6	0	6	28
7	0	7	210175
8	0	8	3
9	0	9	68
10	0	10	227
11	0	11	12
12	0	12	1
13	0	13	41
14	0	14	4222
15	0	15	1

*(It displays 228 rows. Issuer\_Precinct are 0, 14 and 19.)*

**do these precinct zones have an exceptionally high frequency of certain violation codes?**  
Yes.

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

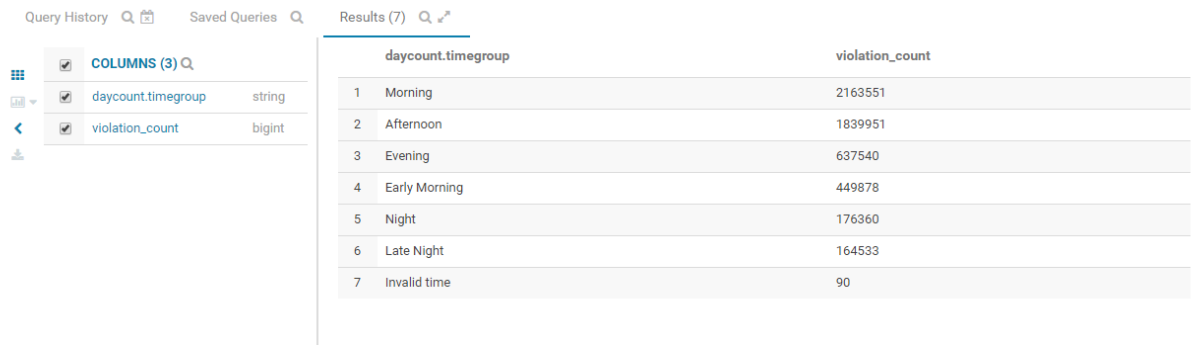
Query:

```
select
daycount.timegroup,
count(*) as violation_count
from
(
select
case
when substr(violation_time,5)='A' and (substr(violation_time,1,2) = '12' or
substr(violation_time,1,2) = '00' or substr(violation_time,1,2) = '01' or substr(violation_time,1,2)
= '02' or substr(violation_time,1,2) = '03') then "Late Night"
when substr(violation_time,5)='A' and (substr(violation_time,1,2) = '04' or
substr(violation_time,1,2) = '05' or substr(violation_time,1,2) = '06' or substr(violation_time,1,2)
= '07') then "Early Morning"
when substr(violation_time,5)='A' and (substr(violation_time,1,2) = '08' or
substr(violation_time,1,2) = '09' or substr(violation_time,1,2) = '10' or substr(violation_time,1,2)
= '11') then "Morning"

when substr(violation_time,5)='P' and (substr(violation_time,1,2) = '12' or
substr(violation_time,1,2) = '00' or substr(violation_time,1,2) = '01' or substr(violation_time,1,2)
= '02' or substr(violation_time,1,2) = '03') then "Afternoon"
when substr(violation_time,5)='P' and (substr(violation_time,1,2) = '04' or
substr(violation_time,1,2) = '05' or substr(violation_time,1,2) = '06' or substr(violation_time,1,2)
= '07') then "Evening"
when substr(violation_time,5)='P' and (substr(violation_time,1,2) = '08' or
substr(violation_time,1,2) = '09' or substr(violation_time,1,2) = '10' or substr(violation_time,1,2)
= '11') then "Night"
else "Invalid time"
end timegroup
from violation_2017_qrc) daycount
group by daycount.timegroup
order by violation_count desc;
```

*(Used case queries here. Question is to find the properties of parking violations across different times of the day. Here case queries checks whether it is AM or PM, based on that then gets its count. Here splitted it as Latenight, Early morning, morning, Afternoon, Evening, Night and the frequencies of tickets. Here daycount alias for CASE statement and timegroup is alias for subquery)*

## Result:



daycount.timegroup	violation_count
1 Morning	2163551
2 Afternoon	1839951
3 Evening	637540
4 Early Morning	449878
5 Night	176360
6 Late Night	164533
7 Invalid time	90

*(Above query listed the total number of tickets on the particular timegroup like, morning, afternoon, evening, early morning, night, latenight and invalid time)*

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

### Query:

```
select hr,violation_code,frequency, rank from(
select hourofday.hr,violation_code,count(*) as frequency,rank() over (PARTITION BY
hourofday.hr ORDER BY count(*) desc) AS rank from(
select violation_code,
case
when substr(violation_time,5)='A' and (substr(violation_time,1,2) = '12' or
substr(violation_time,1,2) = '00' or substr(violation_time,1,2) = '01' or substr(violation_time,1,2)
= '02' or substr(violation_time,1,2) = '03') then "Late Night"

when substr(violation_time,5)='A' and (substr(violation_time,1,2) = '04' or
substr(violation_time,1,2) = '05' or substr(violation_time,1,2) = '06' or substr(violation_time,1,2)
= '07') then "Early Morning"

when substr(violation_time,5)='A' and (substr(violation_time,1,2) = '08' or
substr(violation_time,1,2) = '09' or substr(violation_time,1,2) = '10' or substr(violation_time,1,2)
= '11') then "Morning"

when substr(violation_time,5)='P' and (substr(violation_time,1,2) = '12' or
substr(violation_time,1,2) = '00' or substr(violation_time,1,2) = '01' or substr(violation_time,1,2)
= '02' or substr(violation_time,1,2) = '03') then "Afternoon"

when substr(violation_time,5)='P' and (substr(violation_time,1,2) = '04' or
substr(violation_time,1,2) = '05' or substr(violation_time,1,2) = '06' or substr(violation_time,1,2)
= '07') then "Evening"
```

when substr(violation\_time,5)='P' and (substr(violation\_time,1,2) = '08' or  
substr(violation\_time,1,2) = '09' or substr(violation\_time,1,2) = '10' or substr(violation\_time,1,2)  
= '11') then "Night"

else "Invalid time"

end as hr

from violation\_2017\_qrc)hourofday

group by hourofday.hr,violation\_code)ranking\_table

where ranking\_table.rank<=3;

*(divided 24 hours into 6 discrete bins of equal 4 hours and finding the rank here to find the 3 most commonly occurring violations)*

**Result:**

Results (21+) 🔍 📄

	hr	violation_code	frequency	rank
1	Afternoon	36	286284	1
2	Afternoon	38	240721	2
3	Afternoon	37	167026	3
4	Early Morning	14	74114	1
5	Early Morning	40	60652	2
6	Early Morning	21	57896	3
7	Evening	38	102855	1
8	Evening	14	75902	2
9	Evening	37	70345	3
10	Invalid time	21	32	1
11	Invalid time	94	15	2
12	Invalid time	46	11	3
13	Late Night	21	36957	1
14	Late Night	40	25866	2
15	Late Night	78	15528	3
16	Morning	21	598060	1
17	Morning	36	348165	2
18	Morning	38	176570	3
19	Night	7	26293	1
20	Night	40	22337	2
21	Night	14	21045	3

*(above query lists top 3 commonly occurring violations for all the timegroups)*

**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)**

**Query:**

```
select hr,violation_code,frequency, rank from(
select hourofday.hr,violation_code,count(*) as frequency,rank() over (PARTITION BY
hourofday.hr ORDER BY count(*) desc) AS rank from(
select violation_code,
case
when substr(violation_time,5)='A' and (substr(violation_time,1,2) = '12' or
substr(violation_time,1,2) = '00' or substr(violation_time,1,2) = '01' or substr(violation_time,1,2)
= '02' or substr(violation_time,1,2) = '03') then "Late Night"

when substr(violation_time,5)='A' and (substr(violation_time,1,2) = '04' or
substr(violation_time,1,2) = '05' or substr(violation_time,1,2) = '06' or substr(violation_time,1,2)
= '07') then "Early Morning"

when substr(violation_time,5)='A' and (substr(violation_time,1,2) = '08' or
substr(violation_time,1,2) = '09' or substr(violation_time,1,2) = '10' or substr(violation_time,1,2)
= '11') then "Morning"

when substr(violation_time,5)='P' and (substr(violation_time,1,2) = '12' or
substr(violation_time,1,2) = '00' or substr(violation_time,1,2) = '01' or substr(violation_time,1,2)
= '02' or substr(violation_time,1,2) = '03') then "Afternoon"

when substr(violation_time,5)='P' and (substr(violation_time,1,2) = '04' or
substr(violation_time,1,2) = '05' or substr(violation_time,1,2) = '06' or substr(violation_time,1,2)
= '07') then "Evening"

when substr(violation_time,5)='P' and (substr(violation_time,1,2) = '08' or
substr(violation_time,1,2) = '09' or substr(violation_time,1,2) = '10' or substr(violation_time,1,2)
= '11') then "Night"

else "Invalid time"
end as hr
from violation_2017_qrc)hourofday
where violation_code in (36,21,38)
group by hourofday.hr,violation_code
order by frequency desc)ranking_table
where ranking_table.rank<=1;
(above query displays the most commonly occurring violation code for the timegroup, case
queries and partitioning is used in the above query.)
```

**Result:**

Results (7) 🔍 📄

	hr	violation_code	frequency	rank
1	Morning	21	598060	1
2	Afternoon	36	286284	1
3	Evening	38	102855	1
4	Early Morning	21	57896	1
5	Late Night	21	36957	1
6	Night	38	20347	1
7	Invalid time	21	32	1

8. 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))

i. Query:

```
select
seasoncount.ticketfrequency,
count(*) as ticket_frequency
from
(
select
case
when (substr(issue_date,1,2)='03' or substr(issue_date,1,2) = '04' or substr(issue_date,1,2) =
'05') then "Spring"

when (substr(issue_date,1,2)='06' or substr(issue_date,1,2) = '07' or substr(issue_date,1,2) =
'08') then "Summer"

when (substr(issue_date,1,2)='09' or substr(issue_date,1,2) = '10' or substr(issue_date,1,2) =
'11') then "Fall"

when (substr(issue_date,1,2)='12' or substr(issue_date,1,2) = '01' or substr(issue_date,1,2) =
'02') then "Winter"

else "Invalid Month"

end ticketfrequency
from violation_2017_qrc) seasoncount
group by seasoncount.ticketfrequency
```



order by ticket\_frequency desc;

*(above query splits the whole year into 4 seasons, spring, winter, summer and fall and getting the total number of tickets for the particular season. Used case queries here to split the year into seasons)*

**Result:**

Results (4) 🔍 ↗

seasoncount.ticketfrequency		ticket_frequency
1	Spring	2873380
2	Winter	1704680
3	Summer	852864
4	Fall	979

*(above query displays the total number of tickets for the splitted seasons)*

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

**Query:**

```
select Season,violation_code,frequency, rank from(
select hourofday.Season,violation_code,count(*) as frequency,rank() over (PARTITION BY
hourofday.Season ORDER BY count(*) desc) AS rank from(
select violation_code,
case
when (substr(issue_date,1,2)='03' or substr(issue_date,1,2) = '04' or substr(issue_date,1,2) =
'05') then "Spring"

when (substr(issue_date,1,2)='06' or substr(issue_date,1,2) = '07' or substr(issue_date,1,2) =
'08') then "Summer"

when (substr(issue_date,1,2)='09' or substr(issue_date,1,2) = '10' or substr(issue_date,1,2) =
'11') then "Fall"

when (substr(issue_date,1,2)='12' or substr(issue_date,1,2) = '01' or substr(issue_date,1,2) =
'02') then "Winter"

else "Invalid Month"
end as Season
from violation_2017_qrc)hourofday
group by hourofday.Season,violation_code)ranking_table
where ranking_table.rank<=3;
```

*(above query splits the whole year into 4 seasons, spring, winter, summer and fall and getting the total number of tickets for the particular season. Used case queries here to split the year into seasons and finding the 3 most common violations for each of the season)*

**Result:**

Results (12) 🔍 ↗

	season	violation_code	frequency	rank
1	Fall	46	231	1
2	Fall	21	128	2
3	Fall	40	116	3
4	Spring	21	402424	1
5	Spring	36	344834	2
6	Spring	38	271167	3
7	Summer	21	127350	1
8	Summer	36	96663	2
9	Summer	38	83518	3
10	Winter	21	238180	1
11	Winter	36	221268	2
12	Winter	38	187386	3

*(Above query displays the frequency based on 3 most common violations occurred)*