**Green Taxi:**

**Loading data into Hive Tables for Green:**

Creating tables for taxi data:

create table green (VendorID int, lpep\_pickup\_datetime timestamp,

lpep\_dropoff\_datetime timestamp, store\_and\_fwd\_flag string, ratecodeid

int, pulocationid int, dolocationid int, passenger\_count int,

trip\_distance float, fare\_amount float, extra float, mta\_tax float,

tip\_amount float, tolls\_amount float, ehail\_fee string ,

improve\_surcharge float, total\_amount float, payment\_type int,

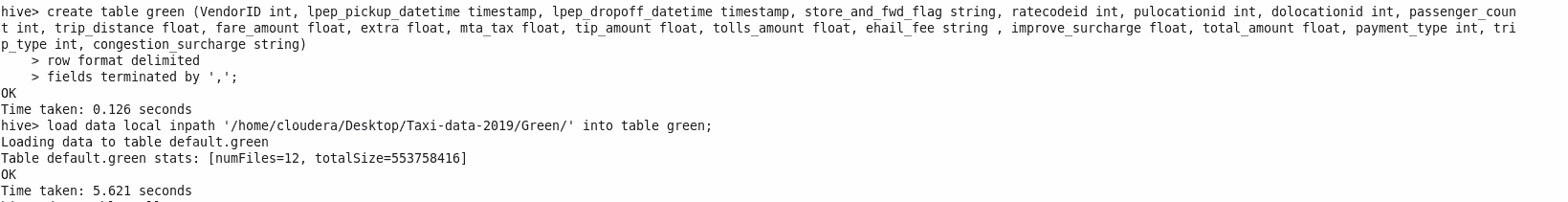
trip\_type int, congestion\_surcharge string)

row format delimited

fields terminated by ',';

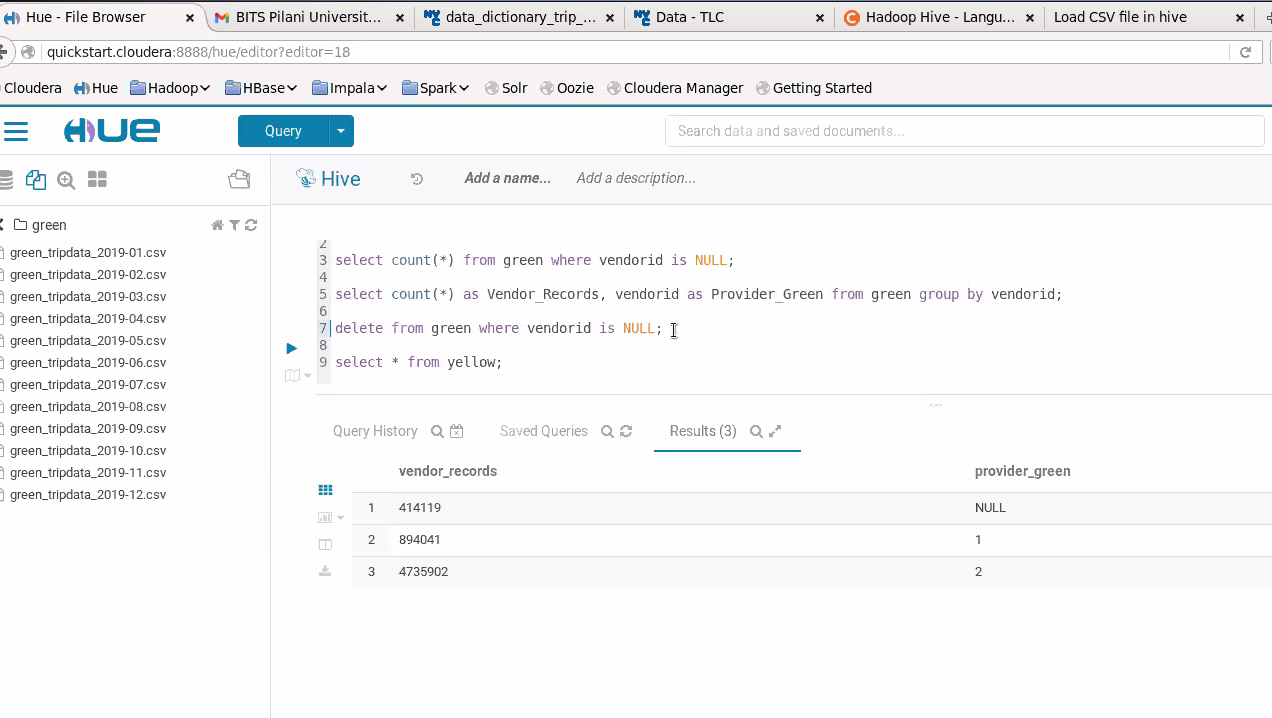
Loading data from csv for all months of 2019:

load data local inpath '/home/cloudera/Desktop/Taxi-data-2019/Green/' into table green;



Total records provided by each vendor:

select count(\*) as Vendor\_Records, vendorid as Provider\_Green from green group by vendorid;



From above result, We see there are some null records for Vendor

**1) Which vendor provides the most useful data?**

Below query is for getting the vendor who is giving more inappropriate data(i.e nulls or negative values in CSV ):

select vendorid, count(\*) from green

where vendorid is null or passenger\_count is null or passenger\_count<=0 or trip\_distance <= 0

or RateCodeID is null or fare\_amount <= 0 or extra < 0

or mta\_tax not in (0,0.5) or tip\_amount < 0

or tolls\_amount < 0

or total\_amount < 0 or year(lpep\_pickup\_datetime) <> 2019 group by vendorid;



From above we can see there is no much difference but vendor 2= VeriFone Inc. is providing more useful data

**Cleaning the data for further processing:**

Now creating another table to load valid and appropriate data by removing nulls and negatives:

Creating table with partition by month:

create table if not exists green\_partition (

vendorid int,

lpep\_pickup\_datetime string,

lpep\_dropoff\_datetime string,

store\_and\_fwd\_flag string,

ratecodeid int,

pulocationid int,

dolocationid int,

passenger\_count int,

trip\_distance double,

fare\_amount double,

extra double,

mta\_tax double,

tip\_amount double,

tolls\_amount double,

improvement\_surcharge double,

total\_amount double,

payment\_type int

) partitioned by (mnth int);

SET hive.exec.max.dynamic.partitions=100000;

SET hive.exec.max.dynamic.partitions.pernode=100000;

SET hive.exec.dynamic.partition=true;

SET hive.exec.dynamic.partition.mode=nonstrict;

Loading data to Green\_partition table:

insert overwrite table green\_partition partition(mnth)

select vendorid, lpep\_pickup\_datetime, lpep\_dropoff\_datetime, store\_and\_fwd\_flag, ratecodeid, pulocationid, dolocationid, passenger\_count,

trip\_distance,

fare\_amount, extra, mta\_tax,

tip\_amount, tolls\_amount, improve\_surcharge, total\_amount, payment\_type, month(lpep\_pickup\_datetime) as mnth

from green

where passenger\_count>0 and trip\_distance > 0

and RateCodeID is not null and fare\_amount > 0 and extra >= 0

and mta\_tax in (0,0.5) and tip\_amount >= 0

and tolls\_amount >= 0 and improve\_surcharge = 0.30

and total\_amount >= 0 and year(lpep\_pickup\_datetime) = 2019;

Counting the number of rows in which any column has NULL values

select count(\*) as Total\_NULL\_records from green\_partition

where vendorid is null

or tpep\_pickup\_datetime is null

or tpep\_pickup\_datetime is null

or passenger\_count is null

or trip\_distance is null

or ratecodeid is null

or store\_and\_fwd\_flag is null

or pulocationid is null

or dolocationid is null

or payment\_type is null

or fare\_amount is null

or extra is null

or mta\_tax is null

or tip\_amount is null

or tolls\_amount is null

or improvement\_surcharge is null

or total\_amount is null

or mnth is null;

There are no nulls in the data

**2) Find the month wise trip count, average distance and average passenger count**

**from the trips completed by yellow and green taxis in 2019. Summary**

**visualizations will be preferred for better analysis.**

- Compare the overall trip count monthwise: Based on the results we can see most number of trips are done in early months of the year and are getting reduced by the end o the year

select mnth as Month, count(\*) as TripCoun from green\_partition

group by mnth order by mnth;

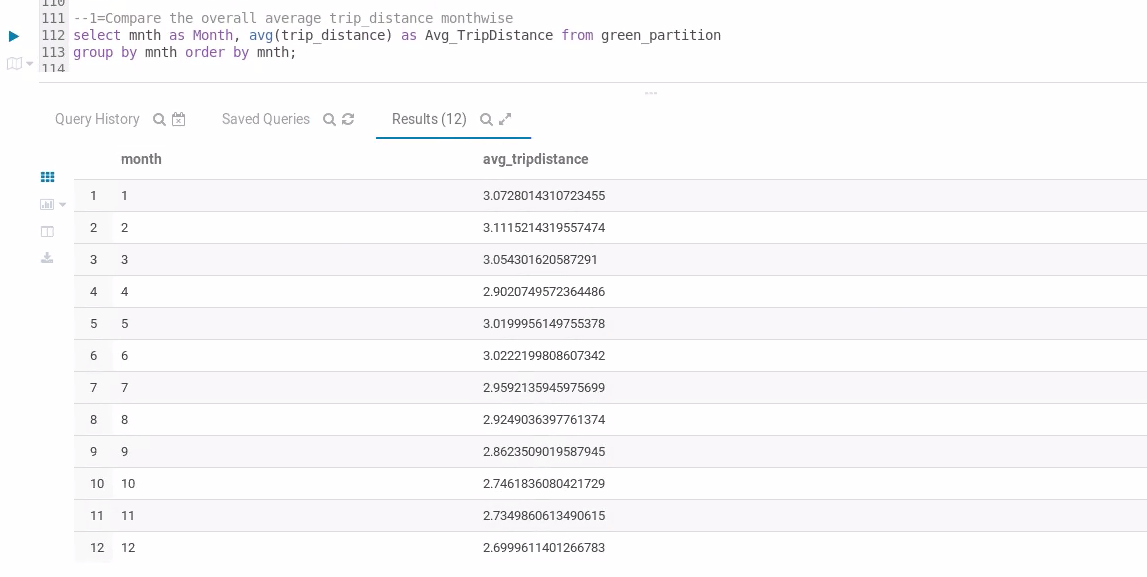


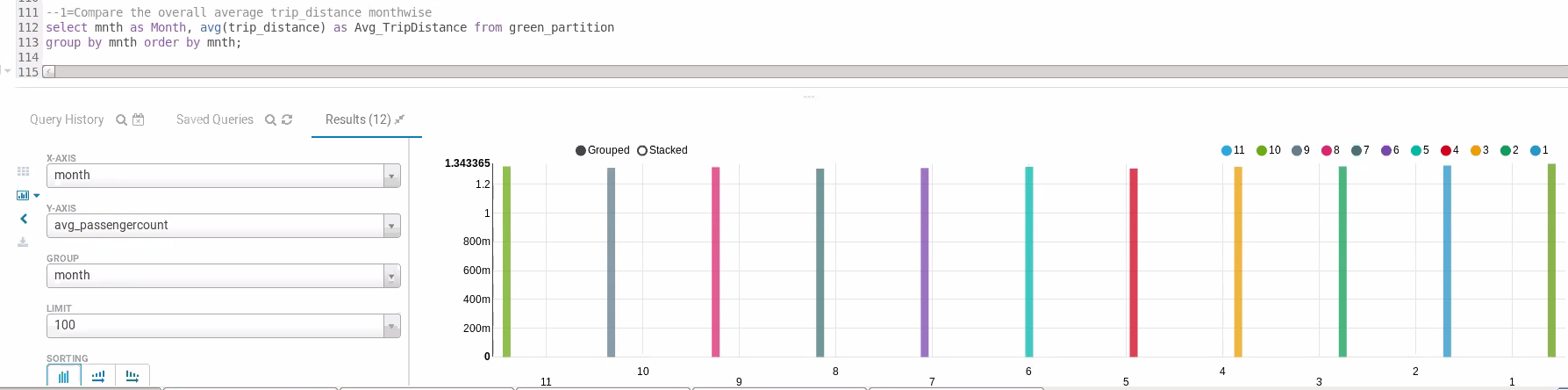


- Compare the overall average trip\_distance monthwise: Based on the results, we can see it is almost similar 2 - 3 miles

select mnth as Month, avg(trip\_distance) as Avg\_TripDistance from green\_partition

group by mnth order by mnth;

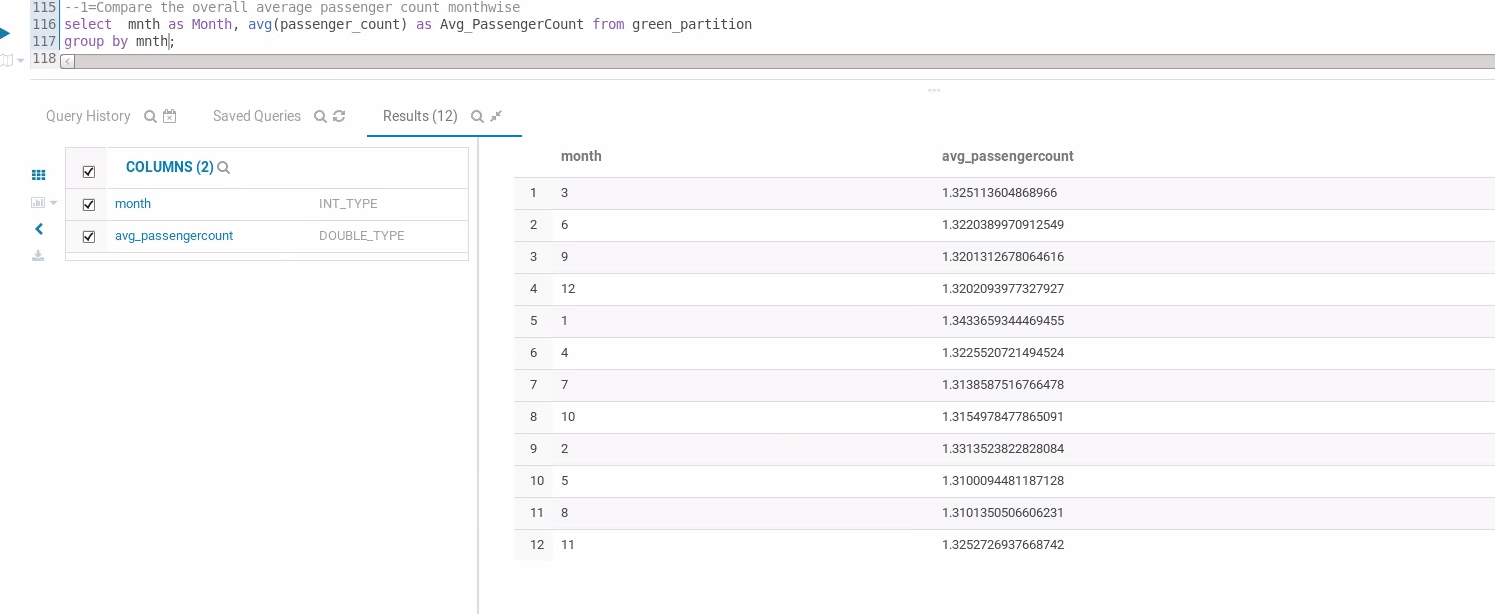




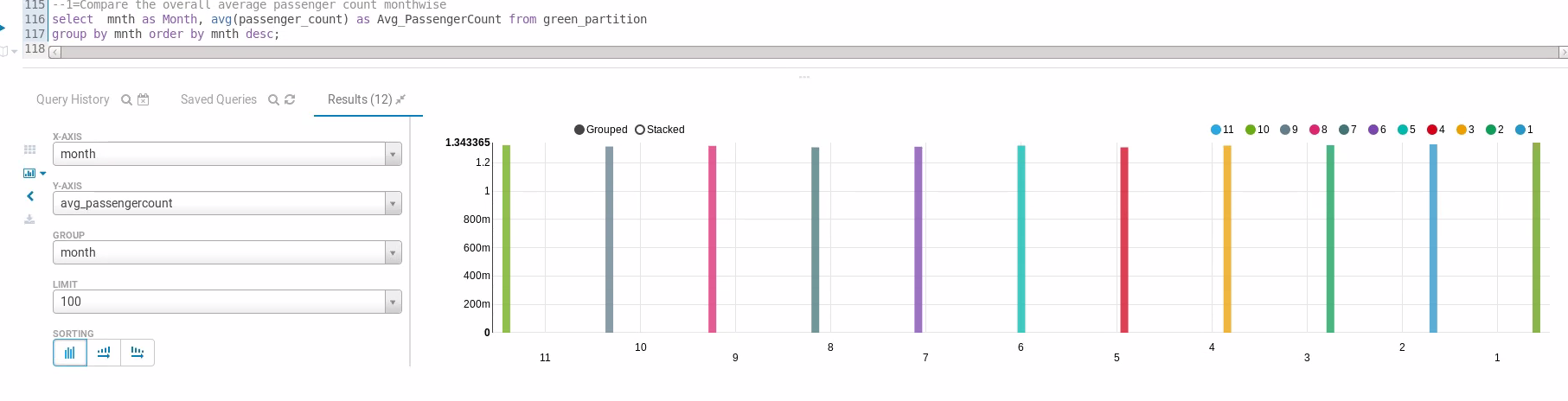
- Compare the overall average passenger count monthwise: From below we can see it is almost similar for all months

select mnth as Month, avg(passenger\_count) as Avg\_PassengerCount from green\_partition

group by mnth order by mnth desc;



Visual bar graph for average passenger count. It is almost same across months



**Loading Location Name data:**

create table taxi\_loc\_data (LocationID int,Borough string,Zone string,service\_zone string)

row format delimited

fields terminated by ',';

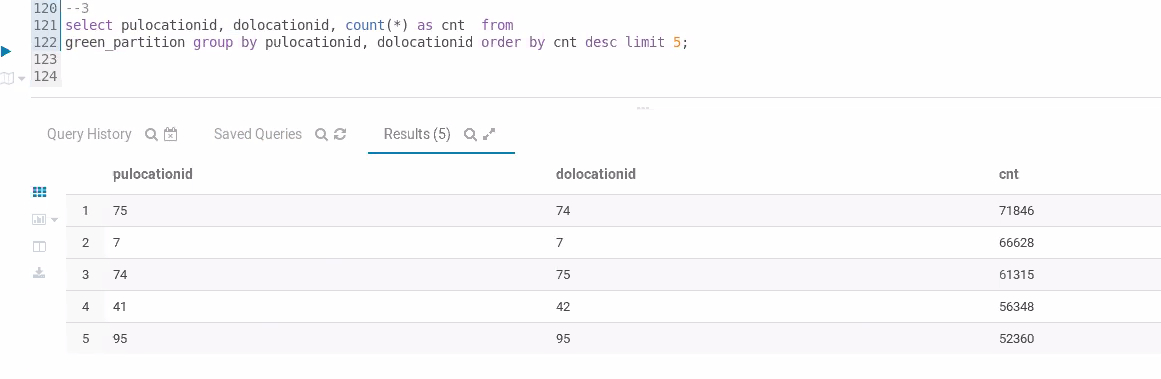
load data local inpath '/home/cloudera/Desktop/Taxi-data-2019/taxi+\_zone\_lookup.csv' into table taxi\_loc\_data;

**3) Find out the five busiest routes served by the yellow and green taxis during 2019.**

**The name of start and drop points to be provided.**

select pulocationid, dolocationid, count(\*) as cnt from

green\_partition group by pulocationid, dolocationid order by cnt desc limit 5;

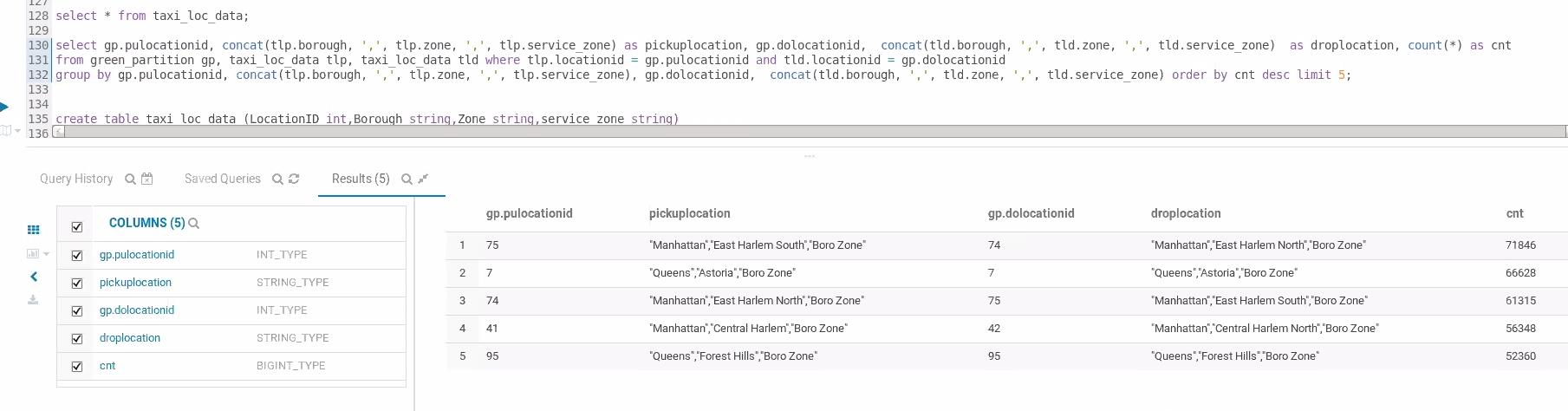


Populating Location names:

select gp.pulocationid, concat(tlp.borough, ',', tlp.zone, ',', tlp.service\_zone) as pickuplocation, gp.dolocationid, concat(tld.borough, ',', tld.zone, ',', tld.service\_zone) as droplocation, count(\*) as cnt

from green\_partition gp, taxi\_loc\_data tlp, taxi\_loc\_data tld where tlp.locationid = gp.pulocationid and tld.locationid = gp.dolocationid

group by gp.pulocationid, concat(tlp.borough, ',', tlp.zone, ',', tlp.service\_zone), gp.dolocationid, concat(tld.borough, ',', tld.zone, ',', tld.service\_zone) order by cnt desc limit 5;

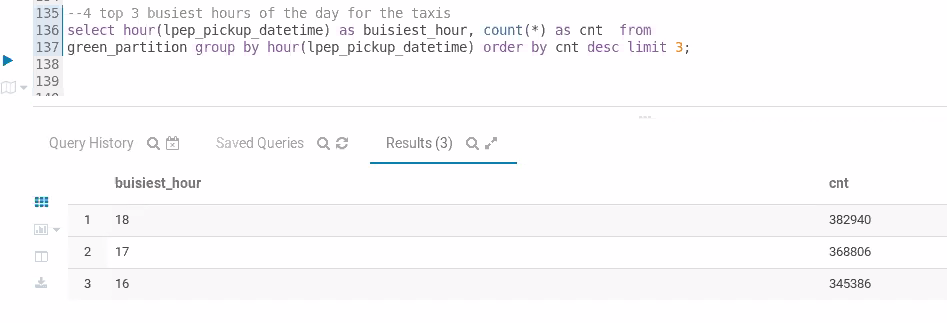


**4) What are the top 3 busiest hours of the day for the taxis?**

From Below we can see evening 4, 5, 6 PM are the busiest hours

select hour(lpep\_pickup\_datetime) as buisiest\_hour, count(\*) as cnt from

green\_partition group by hour(lpep\_pickup\_datetime) order by cnt desc limit 3;



**5) What is the most preferred way of payment used by the passengers? What are**

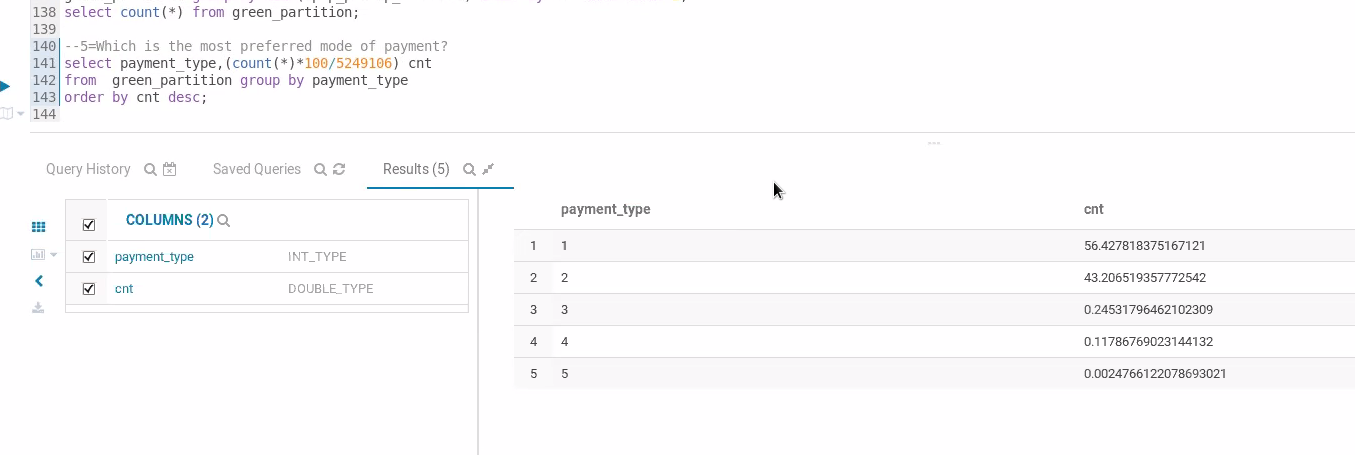
**the weekly trends observed for the methods of payments?**

Mostly used payment mode is credit cards with 56%

select payment\_type,(count(\*)\*100/5249106) cnt

from green\_partition group by payment\_type

order by cnt desc;



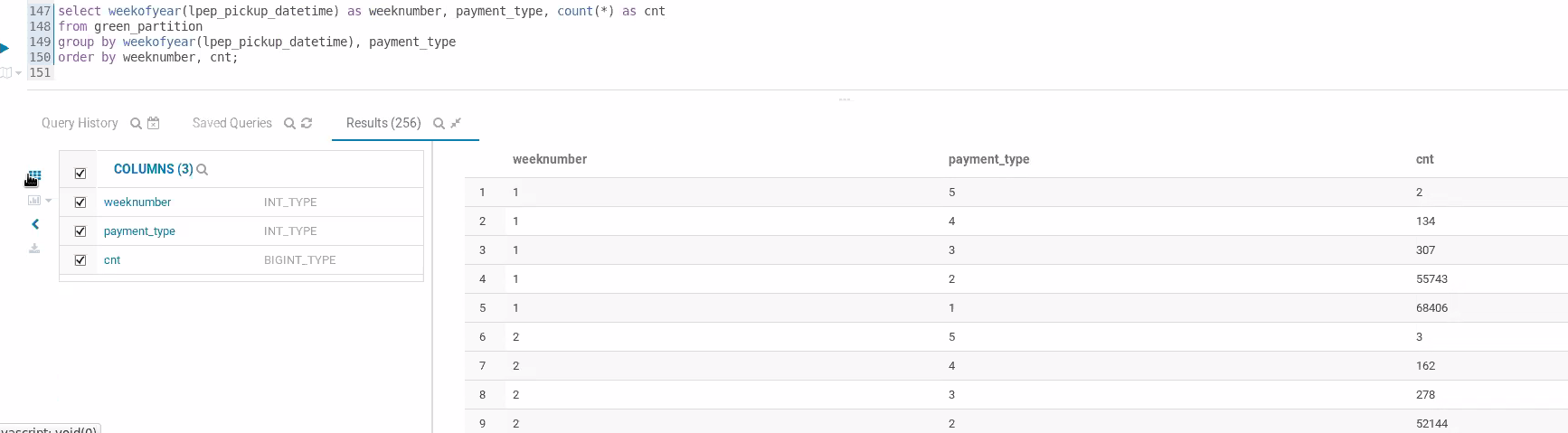
Below is the Weekly trends followed by customers for payments:

select weekofyear(lpep\_pickup\_datetime) as weeknumber, payment\_type, count(\*) as cnt

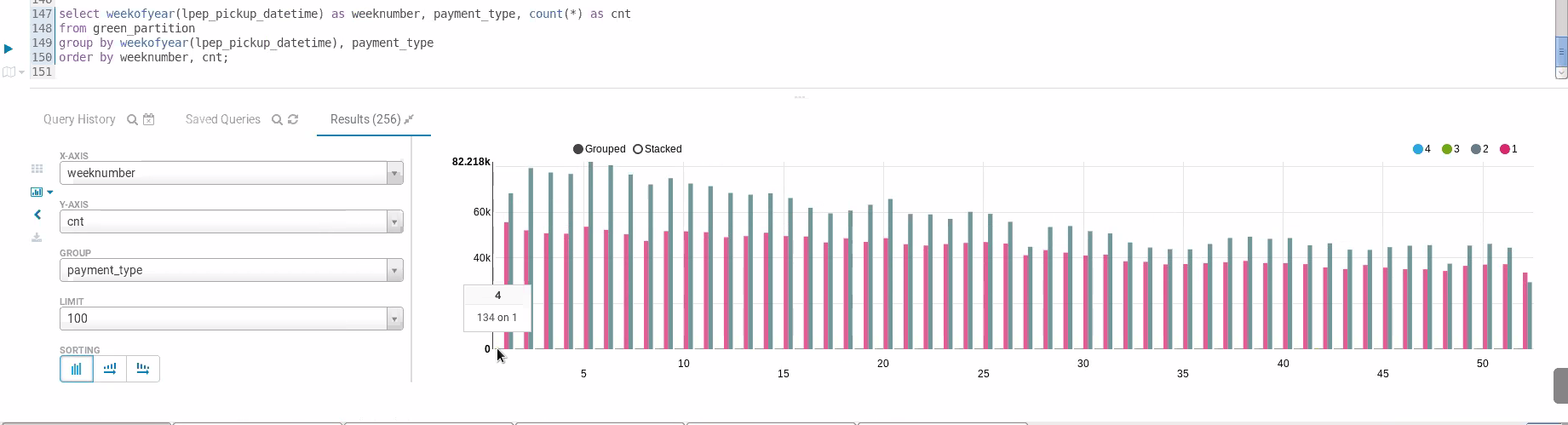
from green\_partition

group by weekofyear(lpep\_pickup\_datetime), payment\_type

order by weeknumber, cnt;



View



**Yellow Taxi:**

Creating table for yellow taxi data:

create table yellow (VendorID int, tpep\_pickup\_datetime timestamp,

tpep\_dropoff\_datetime timestamp, passenger\_count int, trip\_distance

float, ratecodeid int, store\_and\_fwd\_flag string, pulocationid int,

dolocationid int, payment\_type int, fare\_amount float, extra float,

mta\_tax float, tip\_amount float, tolls\_amount float, improve\_surcharge

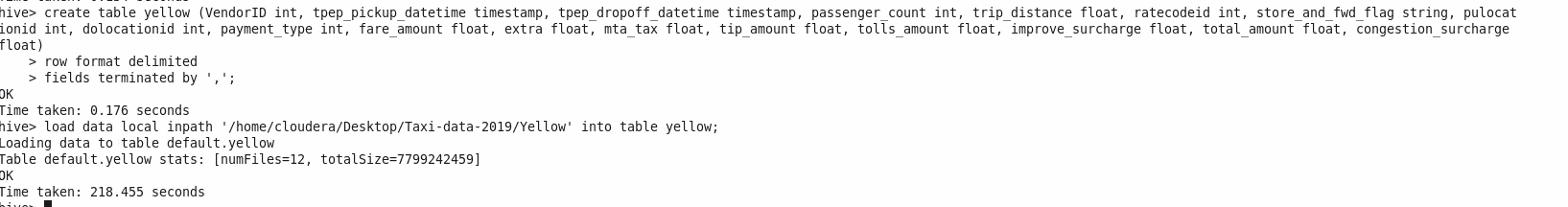
float, total\_amount float, congestion\_surcharge float)

row format delimited

fields terminated by ',';

Loading Data from csv for all months of 2019:

load data local inpath '/home/cloudera/Desktop/Taxi-data-2019/Yellow' into table yellow;



Total records provided by each vendor:

select count(\*) as vendor\_records, vendorid as provider\_yellow from yellow group by vendorid;

We see there are some null records and VendorId 4 for Vendor which are invalid



**1) Which vendor provides the most useful data?**

Below query is for getting the vendor who is giving more inappropriate data(i.e nulls or negative values in CSV ):

select vendorid, count(\*) from yellow

where vendorid is null or passenger\_count is null or passenger\_count<=0 or trip\_distance <= 0

or RateCodeID is null or fare\_amount <= 0 or extra < 0

or mta\_tax not in (0,0.5) or tip\_amount < 0

or tolls\_amount < 0

or total\_amount < 0 or year(lpep\_pickup\_datetime) <> 2019 group by vendorid;

From above we can see there is no much difference but vendor 2= VeriFone Inc. is providing more useful data

**Cleaning the data for further processing:**

Now creating another table to load valid and appropriate data by removing nulls and negatives:

Creating table with partition by month:

create table if not exists yellow\_partition (

vendorid int,

tpep\_pickup\_datetime string,

tpep\_dropoff\_datetime string,

passenger\_count int,

trip\_distance double,

ratecodeid int,

store\_and\_fwd\_flag string,

pulocationid int,

dolocationid int,

payment\_type int

fare\_amount double,

extra double,

mta\_tax double,

tip\_amount double,

tolls\_amount double,

improvement\_surcharge double,

total\_amount double,

) partitioned by (mnth int);

SET hive.exec.max.dynamic.partitions=100000;

SET hive.exec.max.dynamic.partitions.pernode=100000;

SET hive.exec.dynamic.partition=true;

SET hive.exec.dynamic.partition.mode=nonstrict;

Loading data to Yellow\_partition table:

Not able to run the yellow partition insertion and queries. It is taking too much time and System is crashing. All Yellow taxi queries are updated above. Results will be similar to Green taxi details

insert overwrite table yellow\_partition partition(mnth)

select vendorid, tpep\_pickup\_datetime, tpep\_dropoff\_datetime, passenger\_count,

trip\_distance, ratecodeid

store\_and\_fwd\_flag, pulocationid, dolocationid, payment\_type,fare\_amount, extra, mta\_tax,

tip\_amount, tolls\_amount, improve\_surcharge, total\_amount, month(tpep\_pickup\_datetime) as mnth

from yellow

where vendorid is not null and vendorid <> 4 and

passenger\_count>0 and trip\_distance > 0

and RateCodeID is not null and fare\_amount > 0 and extra >= 0

and mta\_tax in (0,0.5) and tip\_amount >= 0

and tolls\_amount >= 0 and improve\_surcharge = 0.30

and total\_amount >= 0 and year(tpep\_pickup\_datetime) = 2019;

Counting the number of rows in which any column has NULL values

select count(\*) as Total\_NULL\_records from yellow\_partition

where vendorid is null

or tpep\_pickup\_datetime is null

or tpep\_pickup\_datetime is null

or passenger\_count is null

or trip\_distance is null

or ratecodeid is null

or store\_and\_fwd\_flag is null

or pulocationid is null

or dolocationid is null

or payment\_type is null

or fare\_amount is null

or extra is null

or mta\_tax is null

or tip\_amount is null

or tolls\_amount is null

or improvement\_surcharge is null

or total\_amount is null

or mnth is null;

There are no nulls in the data

**2) Find the month wise trip count, average distance and average passenger count**

**from the trips completed by yellow and green taxis in 2019. Summary**

**visualizations will be preferred for better analysis.**

- Compare the overall trip count monthwise: Based on the results we can see most number of trips are done in early months of the year and are getting reduced by the end of the year

select mnth as Month, count(\*) as TripCoun from yellow\_partition

group by mnth order by mnth;

- Compare the overall average trip\_distance monthwise: Based on the results, we can see it is almost similar 2 - 3 miles

select mnth as Month, avg(trip\_distance) as Avg\_TripDistance from yellow\_partition

group by mnth order by mnth;

- Compare the overall average passenger count monthwise: From below we can see it is almost similar for all months

select mnth as Month, avg(passenger\_count) as Avg\_PassengerCount from yellow\_partition

group by mnth order by mnth desc;

Visual bar graph for average passenger count. It is almost same across months

**3) Find out the five busiest routes served by the yellow and green taxis during 2019.**

**The name of start and drop points to be provided.**

select pulocationid, dolocationid, count(\*) as cnt from

yellow\_partition group by pulocationid, dolocationid order by cnt desc limit 5;

Populating Location names:

select gp.pulocationid, concat(tlp.borough, ',', tlp.zone, ',', tlp.service\_zone) as pickuplocation, gp.dolocationid, concat(tld.borough, ',', tld.zone, ',', tld.service\_zone) as droplocation, count(\*) as cnt

from green\_partition gp, taxi\_loc\_data tlp, taxi\_loc\_data tld where tlp.locationid = gp.pulocationid and tld.locationid = gp.dolocationid

group by gp.pulocationid, concat(tlp.borough, ',', tlp.zone, ',', tlp.service\_zone), gp.dolocationid, concat(tld.borough, ',', tld.zone, ',', tld.service\_zone) order by cnt desc limit 5;

**4) What are the top 3 busiest hours of the day for the taxis?**

From Below we can see evening 4, 5, 6 PM are the busiest hours

select hour(tpep\_pickup\_datetime) as buisiest\_hour, count(\*) as cnt from

yellow\_partition group by hour(tpep\_pickup\_datetime) order by cnt desc limit 3;

**5) What is the most preferred way of payment used by the passengers? What are**

**the weekly trends observed for the methods of payments?**

Mostly used payment mode is credit cards with 56%

select payment\_type,(count(\*)\*100/5249106) cnt

from yellow\_partition group by payment\_type

order by cnt desc;

Below is the Weekly trends followed by customers for payments:

select weekofyear(tpep\_pickup\_datetime) as weeknumber, payment\_type, count(\*) as cnt

from yellow\_partition

group by weekofyear(tpep\_pickup\_datetime), payment\_type

order by weeknumber, cnt;

Not able to run the yellow partition insertion and queries. It is taking too much time and System is crashing. All Yellow taxi queries are updated above. Results will bi similar to Green taxi details