```
-- IMPORTANT: BEFORE CREATING ANY TABLE, MAKE SURE YOU RUN THIS COMMAND
ADD JAR /opt/cloudera/parcels/CDH/lib/hive/lib/hive-hcatalog-core-1.1.0-
cdh5.11.2.jar;

-- Drop table if exists
DROP TABLE nyc_taxi_data;


-- create an external table

CREATE EXTERNAL TABLE IF NOT EXISTS nyc_taxi_data(
`vendorid` int,
`tpep_pickup_datetime` timestamp,
`tpep_dropoff_datetime` timestamp,
`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)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/common_folder/nyc_taxi_data'
TBLPROPERTIES ("skip.header.line.count"="1");

-- Check for the created tables

select * FROM nyc_taxi_data limit 10;
describe nyc_taxi_data

-- BASIC DATA QUALITY CHECKS

---------------------------------------------------------------------------
----------------------------
--      1. How many records has each TPEP provider provided?
--      Write a query that summarises the number of records of each
provider.
---------------------------------------------------------------------------
----------------------------

SELECT vendorid,count(*) AS count
FROM nyc_taxi_data
GROUP BY vendorid;

----------------------------------------------
--            vendorid       count
--            1               527386
--            2               647183
----------------------------------------------

---------------------------------------------------------------------------
----------------------------
```

```
--              2. The data provided is for months November and December
only.
--              Check whether the data is consistent, and if not, identify
the data quality issues.
--              Mention all data quality issues in comments.
----------------------------------------------------------------------------
----------------------------

-- DATETIME ISSUES

-- Analyzing pickup time

SELECT  vendorid,YEAR(tpep_pickup_datetime) AS yr,
MONTH(tpep_pickup_datetime) AS mnth,count(*) AS total
FROM nyc_taxi_data
GROUP BY vendorID, YEAR(tpep_pickup_datetime), MONTH(tpep_pickup_datetime)
ORDER BY vendorID, yr, mnth;

-- Analyzing drop time

SELECT  vendorid,YEAR(tpep_dropoff_datetime) AS yr,
MONTH(tpep_dropoff_datetime) AS mnth,count(*) AS total
FROM nyc_taxi_data
GROUP BY vendorID, YEAR(tpep_dropoff_datetime),
MONTH(tpep_dropoff_datetime)
ORDER BY vendorID, yr, mnth;

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

--      There are data with years 2003, 2008, 2009, 2018, 2019
--      Data with 2018 with month 1 can be accetable as the trip has
started in 31 DEC 2017 and ended up in 1 Jan 2018
--      But Data with other years with random months are erroneous and
vendor 2 is providing almost all of them.

----------------------------------------------------------------------------
----------------------------
--              3. You might have encountered unusual or erroneous rows in
the dataset.
--              Can you conclude which vendor is doing a bad job in
providing the records?
----------------------------------------------------------------------------
----------------------------

--      We have only 2 Vendor IDs. To review the job done by the vendors,
we can use the rate code IDs
--      Analyzing the Rate code IDs

SELECT  vendorid, ratecodeid, count(*) as count
FROM nyc_taxi_data
GROUP BY vendorid, ratecodeid
ORDER BY vendorid, ratecodeid;


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

--              vendorid       ratecodeid             _c2

--              1                              1                       513991
--          1               2         10544
--          1               3         1186
--          1               4         230
--          1               5         1425
--          1               6         2
--          1               99        8

--          2               1         628287
--          2               2         14794
--          2               3         1376
--          2               4         356
--          2               5         2368
--          2               6         1
--          2               99        1

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

--  Both the vendors are not performing a good job as the rating is 1 for a
huge share of trips
--     Also, both the vendors are seeding invalid ratecodeIDs,
--     in which vendor ID 1 (Creative Mobile Technologies) is top with 8
records.

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

-- UNUSUAL OR ERRONEOUS PASSENGER COUNT

SELECT vendorid, passenger_count, count(*) AS count
FROM nyc_taxi_data
GROUP BY vendorid, passenger_count
ORDER BY vendorID, passenger_count;

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

--     Vendor 1 and 2 are seeding unusual passenger_count i.e equal to 0
--  The number for Vendor 1 is higher compared to Vendor 2

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

-- UNUSUAL OR ERRONEOUS FARE AMOUNT

SELECT vendorid, count(*) AS count_error, sum(fare_amount) AS sum_error
FROM nyc_taxi_data
WHERE fare_amount<0
GROUP BY vendorid;

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

--          vendorid      count_error          sum_error
--          2                   558                        -4917.38

--     fare_amount was unusually charged by vendor 2, i.e 558 times with
total of -$4917(negative)
```

```
--------------------------------------------------------------------------------------------------

--      UNUSUAL OR ERRONEOUS EXTRA CHARGE

SELECT vendorid, count(*) AS count_error, sum(extra) AS sum_error
FROM nyc_taxi_data
WHERE extra NOT IN (0,0.5,1)
GROUP BY vendorid;

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

--      Extra charges are also unusual somewhere, where Both are charging
some unusual Extra as 1823 and 3033
--      times respectively, where vendor 2 (VeriFone Inc) has done this
more times with approx. 4000 USD

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

-- UNUSUAL OR ERRONEOUS MTA TAX

SELECT vendorid, count(*) AS count_error,sum(mta_tax) AS sum_error
FROM nyc_taxi_data
WHERE mta_tax NOT IN (0,0.5)
GROUP BY vendorid;

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

--      mta_tax was also unusually charged, mostly from vendor 2, i.e 547
times with total of -$263(negative)

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

-- UNUSUAL OR ERRONEOUS TIP AMOUNT

SELECT vendorid, count(*) AS count_error,sum(tip_amount) AS sum_error
FROM nyc_taxi_data
WHERE tip_amount<0
GROUP BY vendorid;

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

--      vendorid       count_error            sum_error
--      2                      4                              -3.5

-- tip_amount was unusually charged in a few cases, all from vendor 2, i.e
4 times with total of -$3.5 (negative)

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

-- UNUSUAL OR ERRONEOUS TIP AMOUNT IMPROVEMENT_SURCHARGE

SELECT vendorid, count(*) AS count_error,sum(improvement_surcharge) AS
sum_error
FROM nyc_taxi_data
```

```
WHERE improvement_surcharge NOT IN (0,0.3)
GROUP BY vendorid;

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

--      improvement_surcharge was also unusually charged, all from vendor
2, i.e 562 times with total of -$163.4 (negative)

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

-- UNUSUAL OR ERRONEOUS PICKUP AND DROP TIME

SELECT vendorid ,count(*) AS count_error
FROM nyc_taxi_data
WHERE
UNIX_TIMESTAMP(tpep_pickup_datetime)>UNIX_TIMESTAMP(tpep_dropoff_datetime)
GROUP BY vendorid;

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

--      vendorid       count_error
--      1                      73

-- Vendor 1 is seeding unusual pickup time i.e greater than drop time

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

-- UNUSUAL OR ERRONEOUS TRIP DURATION

SELECT vendorid ,count(*) as count_error
FROM nyc_taxi_data
WHERE trip_distance <= 0
GROUP BY vendorid;

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

--      vendorid       count_error
--      1                      3185
--      2                      4217

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

-------------------- Final Conclusion on unusual or erroneous rows in the
dataset --------------------

-- There are unusual or erroneous rows in dataset with respect to:

-- RATE CODES
-- PASSENGER COUNT
-- FARE AMOUNT
-- TRIP DURATION
-- PASSENGER COUNT
-- PICKUP & DROP TIME
-- MTA TAX
-- EXTRA CHARGE
-- TIP AMOUNT
-- FARE AMOUNT
-- IMPROVEMENT SURCHARGE

--      Detail about these unusual or erroneous rows has been discussed
above with the queries

--      Note: Both 1 and 2, are corrrect answers as per the accountability
of the erroneous points

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

-- CREATING A CLEAN, ORC PARTITIONED TABLE FOR ANALYSIS

-- CREATING A PARTITIONED TABLE WITH FORMATTED DATA AND IN ORC COMPRESSION
-- IMPORTANT: BEFORE PARTITIONING ANY TABLE, MAKE SURE YOU RUN THESE
COMMANDS

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

-- Drop table if already exists
DROP TABLE IF EXISTS nyc_taxi_data_partitioned_orc;

-- Creating table with required datatypes(columns), partition settings and
compressd format configuration

CREATE EXTERNAL TABLE IF NOT EXISTS nyc_taxi_data_partitioned_orc(
`vendorid` int,
`tpep_pickup_datetime` timestamp,
`tpep_dropoff_datetime` timestamp,
`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 (yr int, mnth int)
STORED AS ORC
LOCATION '/user/hive/warehouse/nyc_taxi_data_partitioned_orc'
TBLPROPERTIES ("orc.compress"="SNAPPY");

SELECT * FROM nyc_taxi_data_partitioned_orc;

-- Setting to allow partition limits and insert permission into partition
SET hive.exec.dynamic.partition=true;
SET hive.exec.dynamic.partition.mode=nonstrict;


-- Inserting data to the partitioned table
INSERT OVERWRITE TABLE nyc_taxi_data_partitioned_orc PARTITION(yr, mnth)
```

```
SELECT `vendorid` int,
`tpep_pickup_datetime` timestamp,
`tpep_dropoff_datetime` timestamp,
`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,
YEAR(`tpep_pickup_datetime`) AS yr,
MONTH(`tpep_pickup_datetime`) AS mnth
FROM nyc_taxi_data
WHERE YEAR(`tpep_pickup_datetime`)= 2017 AND MONTH(`tpep_pickup_datetime`)
in (11,12)
AND `passenger_count` NOT IN (0, 192)
AND `ratecodeid` IN (1,2,3,4,5,6);

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

--      Note: We have removed the erroneous data pertaining to columns:
tpep_pickup_datetime, passenger_count and ratecodeid

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

--      Checking for data in table

SELECT * FROM nyc_taxi_data_partitioned_orc LIMIT 10;

--      Checking for total records available

SELECT vendorid, COUNT(*) AS count
FROM nyc_taxi_data_partitioned_orc
GROUP BY vendorid
ORDER BY vendorid;

---------------------------------------------
--             vendorid       count (previous)
--             1               520571 (527386)
--             2               647157 (647183)
---------------------------------------------

-- Note: Here we have used partitions, buckets and ORC AS well
-- We don't expect the same from student, even if they create "Partitions"
and "ORC" individually, we'll provide them full marks

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

-- ANALYSIS-I
```

```
----------------------------------------------------------------------------
----------------------------
-- 1. Compare the average fare for November and December.
----------------------------------------------------------------------------
----------------------------

SELECT mnth, round(avg(fare_amount),2) AS Average_fare_amount
FROM nyc_taxi_data_partitioned_orc
WHERE fare_amount>0
GROUP BY mnth;

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

--      mnth    average_fare_amount
--      11              13.12
--      12              12.91

--      Average Fare amount in November is slightly higher than the same in
December.

----------------------------------------------------------------------------
----------------------------
--      2. Explore the number of passengers per trip - how many trips are
made by each level
--      of Passenger_count? Do most people travel solo or with other
people?
----------------------------------------------------------------------------
----------------------------

SELECT passenger_count, count(*) AS Total
FROM nyc_taxi_data_partitioned_orc
WHERE isnotnull(passenger_count) and passenger_count>0
GROUP BY passenger_count
ORDER BY passenger_count;

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

--      passenger_count        total
--      1                                      827485
--      2                                      176871
--      3                                      50693
--      4                                      24951
--      5                                      54567
--      6                                      33145
--      7                                      12
--      8                                      3
--      9                                      1

--      Most of the people travel solo, as we can see the count is highest
(827485) when passenger_count is 1

----------------------------------------------------------------------------
----------------------------
-- 3. Which is the most preferred mode of payment?
----------------------------------------------------------------------------
----------------------------

SELECT payment_type, count(*) AS Total
FROM nyc_taxi_data_partitioned_orc
```

```
GROUP BY payment_type;

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

--      payment_mode            total
--      Credit card (1)         785728
--      Cash (2)                        374179
--      No charge (3)           6187
--      Dispute (4)                     1634

--      Credit Card is most preffered mode of payment

----------------------------------------------------------------------------
----------------------------
-- 4. What is the average tip paid? Compare the average tip with the 25th,
50th and 75th percentiles
-- and comment whether the  average tip  is a representative statistic (of
the central tendency)
-- of tip amount paid.
----------------------------------------------------------------------------
----------------------------

SELECT AVG(tip_amount) AS average_tip
FROM nyc_taxi_data_partitioned_orc
WHERE tip_amount >= 0
AND fare_amount > 0;

-- Average tip_amount paid
--      1.8545392085777652

SELECT  PERCENTILE_APPROX(`tip_amount`, 0.25) AS 25_per,
PERCENTILE_APPROX(`tip_amount`, 0.50) AS 50_Per,
PERCENTILE_APPROX(`tip_amount`, 0.75) AS 75_per,
AVG(`tip_amount`) AS Average
FROM nyc_taxi_data_partitioned_orc
WHERE `tip_amount` >= 0
AND `fare_amount` > 0;

----------------------------------------------------------------------------
----------------------------
--      25th, 50th and 75th percentiles of tip amount
--      [0.0,1.36,2.45]
----------------------------------------------------------------------------
----------------------------

-- Average tip_amount paid       : 1.85
-- 50th percentile of tip amount : 1.36
-- The  average tip  is not a representative statistic (of the central
tendency) of  tip amount paid .

----------------------------------------------------------------------------
----------------------------
-- 5. Explore the Extra (charge) variable - what is the fraction of total
trips where an extra charge is levied?
----------------------------------------------------------------------------
----------------------------

SELECT SUM(IF( extra > 0, 1 , 0 ) )/ COUNT(*) * 100 AS
Fraction_When_Levied_Extra
FROM nyc_taxi_data_partitioned_orc
```

```
---------------------------------------------------------------------------
----------------------------

--      46.21% of trips happen when extra charge is levied, means people
like to travel when no extra charge is levied.

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

-- ANALYSIS-II

---------------------------------------------------------------------------
----------------------------
-- 1.What is the correlation between the number of passengers and tip paid?
Do multiple travellers
-- pay more compared to solo travellers?
---------------------------------------------------------------------------
----------------------------
-- Finding correlation

SELECT CORR(tip_amount, passenger_count)
from nyc_taxi_data_partitioned_orc
WHERE tip_amount>=0 AND passenger_count>0

-------------- OUTPUT --------------
--      -0.0047482397788550035
------------------------------------

-- Verifying correlation by Correlation Coefficient(r)=Cov(x,y)/Sx*Sy

SELECT covar_pop(tip_amount,
passenger_count)/(stddev_pop(tip_amount)*stddev_pop(passenger_count))
from nyc_taxi_data_partitioned_orc
WHERE tip_amount>=0 AND passenger_count>0

-------------- OUTPUT --------------
--             -0.00474823977885501
------------------------------------

-- CONCLUSION

--  Correlation between the number of passengers and tip paid: -
0.0047482397788550035
--  It indicates Weak Negative Correlation.
--  It means as number of passengers increases, the tip amount decreases
very slightly.
--  Based on correlation value, solo travellers pay more compared to
multiple travellers.

---------------------------------------------------------------------------
--------------------
--      Q2. Segregate the data into five segments of â€˜tip paidâ€™: [0-5),
[5-10), [10-15) , [15-20) and >=20.
--      Calculate the percentage share of each bucket (i.e. the fraction of
trips falling in each bucket).
---------------------------------------------------------------------------
--------------------

-- Below query returns the fraction of trips falling in each segment.
```

```
SELECT (SUM(IF(tip_amount >=0 AND tip_amount < 5, 1,0))/COUNT(*))*100 AS
`[0-5)`,
       (SUM(IF(tip_amount >=5 AND tip_amount < 10, 1,0))/COUNT(*))*100 AS
`[5-10)`,
       (SUM(IF(tip_amount >=10 AND tip_amount < 15, 1,0))/COUNT(*))*100 AS
`[10-15)`,
       (SUM(IF(tip_amount >=15 AND tip_amount < 20, 1,0))/COUNT(*))*100 AS
`[15-20)`,
       (SUM(IF(tip_amount >=20, 1,0))/COUNT(*))*100 AS `>=20`
FROM nyc_taxi_data_partitioned_orc
WHERE tip_amount >= 0
AND fare_amount  > 0;


-- CONCLUSION

-- Fraction of Trips Falling in Bucket [0-5)   - 92.11106369047926
-- Fraction of Trips Falling in Bucket [5-10)  - 5.654481849610322
-- Fraction of Trips Falling in Bucket [10-15) - 1.8930259105189817
-- Fraction of Trips Falling in Bucket [15-20) - 0.23687462613402444
-- Fraction of Trips Falling in Bucket >=20    - 0.10455392325742033

---------------------------------------------------------------------------
--------------------
--      Q3. Which month has a greater average  speed  - November or
December? Note that the
--      variable  speed  will have to be derived from other metrics.
---------------------------------------------------------------------------
--------------------

-- Below query returns average  speed  for November & December 2017.

SELECT mnth,
AVG(trip_distance/((UNIX_TIMESTAMP(tpep_dropoff_datetime) -
UNIX_TIMESTAMP(tpep_pickup_datetime))/3600)) as Avg_Speed_MPH
FROM nyc_taxi_data_partitioned_orc
WHERE trip_distance >= 0
GROUP BY mnth;


-- CONCLUSION
-------------------
-- November Month has average  speed  as 10.97802043563046 Miles Per Hour
-- December Month has average  speed  as 11.073593998600314 Miles Per Hour
-- Based on average  speed  values, December Month has a greater average
speed

---------------------------------------------------------------------------
--------------------
--      Q4. Analyse the average speed of the most happening days of the
year i.e. 31st December
--      (New year's eve) and 25th December (Christmas Eve) and compare it
with the overall average.
---------------------------------------------------------------------------
--------------------

-- Below query returns overall average speed for both November & December
2017.

SELECT AVG(trip_distance/((UNIX_TIMESTAMP(tpep_dropoff_datetime) -
UNIX_TIMESTAMP(tpep_pickup_datetime))/3600)) as Avg_Speed_MPH
```

```
FROM nyc_taxi_data_partitioned_orc
WHERE trip_distance >= 0
AND YEAR(tpep_dropoff_datetime) IN (2017, 2018);

-- overall average speed of the trips: 11.026369911646409 Miles Per Hour

-- Below query returns average speed on 31st December 2017 & 25th December
2017.

SELECT FROM_UNIXTIME(UNIX_TIMESTAMP(tpep_pickup_datetime), 'dd-MMM-yyyy')
as `Happening_date`,
AVG(trip_distance/((UNIX_TIMESTAMP(tpep_dropoff_datetime) -
UNIX_TIMESTAMP(tpep_pickup_datetime))/3600)) as Avg_Speed_MPH
FROM nyc_taxi_data_partitioned_orc
WHERE trip_distance >= 0
AND mnth = 12
AND DAY(tpep_pickup_datetime) IN (25,31)
AND YEAR(tpep_dropoff_datetime) IN (2017, 2018)
GROUP BY FROM_UNIXTIME(UNIX_TIMESTAMP(tpep_pickup_datetime), 'dd-MMM-
yyyy');


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

-- CONCLUSION

-- On 25-Dec-2017, the average  speed  was 15.24030794591516 Miles Per Hour
-- On 31-Dec-2017, the average  speed  was 13.202755584924587 Miles Per
Hour
-- Overall average  speed  was 11.026369911646409
-- Based on average  speed  values analysis, the average speed  was vewry
high on 25-Dec-2017.

---------------------------------------------------------------------------
--------------------
```