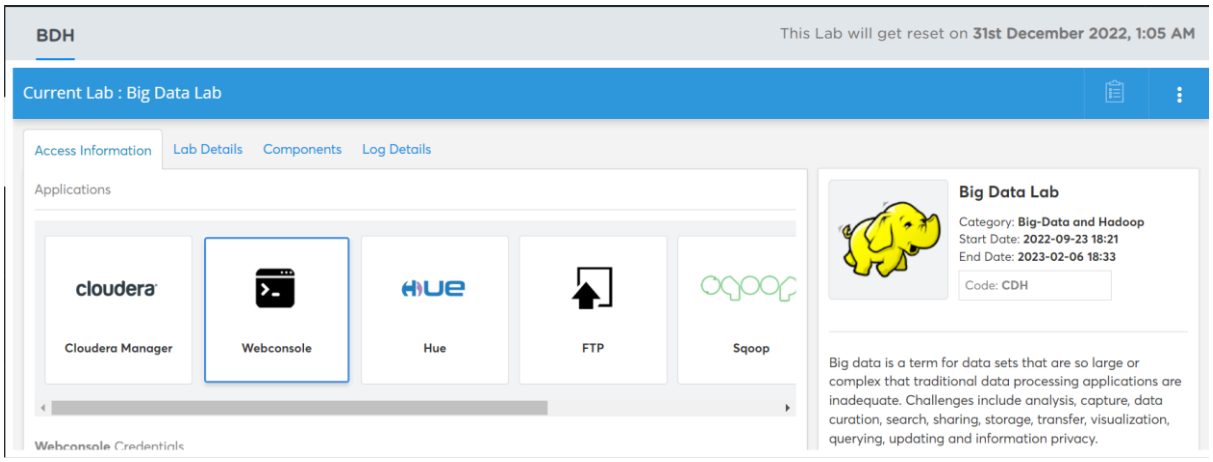
YELLOW Taxi trip analysis using Hive

First-Step: Before Executing any queries inside HDFS, HIVE first we must prepare the Lab. In this project I have used Simplilearn big data cloudera distribution lab.



So from web-Console first login into the web console with proper user name and password.

Username: kaushikdey1984yahoo

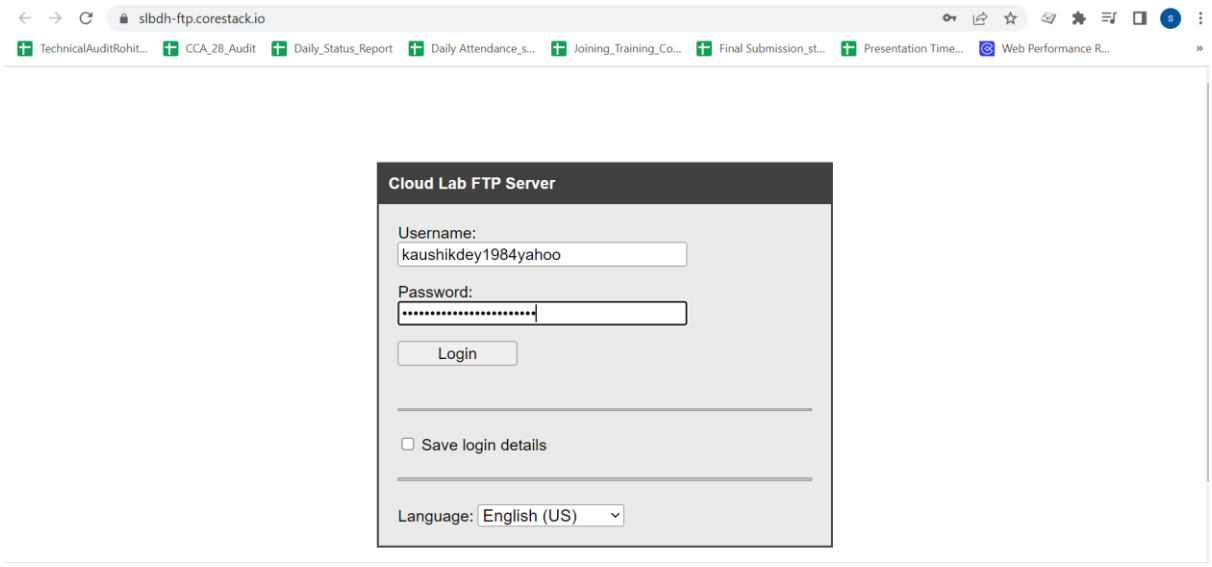
Password : kaushikdey1984yahooodeqx

After that from web-console go to the hive shell with following screen-shot commands.

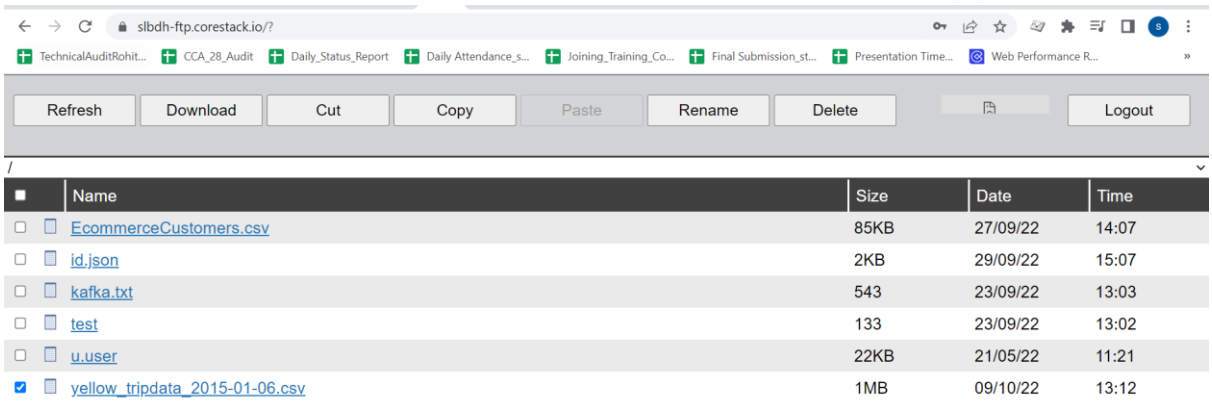
```
ip-10-0-41-79 login: kaushikdey1984yahoo
Password:
Last login: Tue Sep 27 15:44:28 on pts/1292
[kaushikdey1984yahoo@ip-10-0-41-79 ~]$ hive
WARNING: Use "yarn jar" to launch YARN applications.
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/cloudera/parcels/CDH-6.3.2-1.cdh6.3.2.p0.1605554/jars/log4j-slf4j-impl-2.8.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/cloudera/parcels/CDH-6.3.2-1.cdh6.3.2.p0.1605554/jars/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/opt/cloudera/parcels/CDH-6.3.2-1.cdh6.3.2.p0.1605554/jars/hive-common-2.1.1-cdh6.3.2.jar!/hive-log4j2.properties Async: false

WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> Create database mohita;
OK
Time taken: 2.519 seconds
hive> use mohita;
OK
Time taken: 0.064 seconds
hive> show tables;
OK
Time taken: 0.14 seconds
hive>
```

YELLOW Taxi trip analysis using Hive

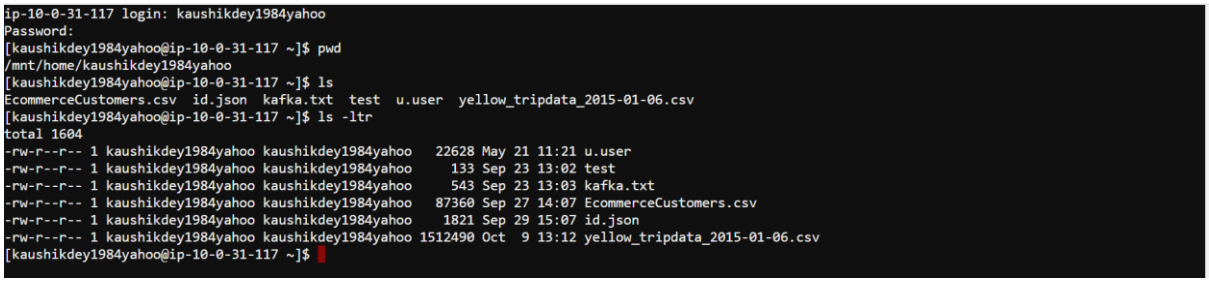
Second-Step: Load Yellow Taxi trip csv data from local system to hdfs via FTP.



Now the tick checkbox data is from yellow_tripdata_2015-01-06.csv.



We have to confirm that from the webconsole shell also. This is the following screen-shot.



Now move this csv file from userpath to hdfs with following commands.

1) Hadoop fs -mkdir newDataFlair

2) Hadoop fs -put yellow_tripdata_2015-01-06.csv newDataFlair

3) Hadoop fs -ls newDataFlair

YELLOW Taxi trip analysis using Hive

```
[kaushikdey1984yahoo@ip-10-0-41-79 ~]$ hadoop fs -ls
Found 5 items
drwx------   - kaushikdey1984yahoo hadoop          0 2022-09-24 14:00 .Trash
drwxr-xr-x   - kaushikdey1984yahoo hadoop          0 2022-09-29 18:00 .sparkStaging
drwxr-xr-x   - kaushikdey1984yahoo hadoop          0 2022-09-27 16:28 Desktop
drwxr-xr-x   - kaushikdey1984yahoo hadoop          0 2022-09-29 15:09 newDataFlair
drwxr-xr-x   - kaushikdey1984yahoo hadoop          0 2022-09-23 13:09 sample_101
[kaushikdey1984yahoo@ip-10-0-41-79 ~]$ hadoop fs -ls /newDataFlair
ls: `/newDataFlair': No such file or directory
[kaushikdey1984yahoo@ip-10-0-41-79 ~]$ hadoop fs -ls newDataFlair
Found 2 items
-rw-r--r--   3 kaushikdey1984yahoo hadoop      87360 2022-09-27 16:02 newDataFlair/EcommerceCustomers.csv
-rw-r--r--   3 kaushikdey1984yahoo hadoop       1821 2022-09-29 15:09 newDataFlair/id.json
[kaushikdey1984yahoo@ip-10-0-41-79 ~]$ hadoop fs -put yellow_tripdata_2015-01-06.csv newDataFlair
[kaushikdey1984yahoo@ip-10-0-41-79 ~]$ hadoop fs -ls newDataFlair
Found 3 items
-rw-r--r--   3 kaushikdey1984yahoo hadoop      87360 2022-09-27 16:02 newDataFlair/EcommerceCustomers.csv
-rw-r--r--   3 kaushikdey1984yahoo hadoop       1821 2022-09-29 15:09 newDataFlair/id.json
-rw-r--r--   3 kaushikdey1984yahoo hadoop    1512490 2022-10-09 13:20 newDataFlair/yellow_tripdata_2015-01-06.csv
[kaushikdey1984yahoo@ip-10-0-41-79 ~]$
```

Third-Step: Now change the shell (that is Hive) and execute the following commands:

1) Create database mohita;
2) Use mohita;
3) Hive>CREATE TABLE IF NOT EXISTS taxidata (vendor_id string, pickup_datetime string, dropoff_datetime string, passenger_count int, trip_distance DECIMAL(9,6), pickup_longitude DECIMAL(9,6), pickup_latitude DECIMAL(9,6), rate_code int, store_and_fwd_flag string, dropoff_longitude DECIMAL(9,6), dropoff_latitude DECIMAL(9,6), payment_type string, fare_amount DECIMAL(9,6), extra DECIMAL(9,6), mta_tax DECIMAL(9,6), tip_amount DECIMAL(9,6), tolls_amount DECIMAL(9,6), total_amount DECIMAL(9,6), trip_time_in_secs int) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED as TEXTFILE TBLPROPERTIES ("skip.header.line.count"="1");
4) Hive>Show tables;

```
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/opt/cloudera/parcels/CDH-6.3.2-1.cdh6.3.2.p0.1605554/jars/hive-common-2.1.1-cdh6.3.2.jar!/hive-log4j
2.properties Async: false

WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> Create database mohita;
OK
Time taken: 2.519 seconds
hive> use mohita;
OK
Time taken: 0.064 seconds
hive> show tables;
OK
Time taken: 0.14 seconds
hive> CREATE TABLE IF NOT EXISTS taxidata  (vendor_id string, pickup_datetime string, dropoff_datetime string, passenger_count int, trip_distance DECIMAL
(9,6), pickup_longitude DECIMAL(9,6), pickup_latitude DECIMAL(9,6), rate_code int, store_and_fwd_flag string, dropoff_longitude DECIMAL(9,6), dropoff_lat
itude DECIMAL(9,6), payment_type string, fare_amount DECIMAL(9,6), extra DECIMAL(9,6), mta_tax DECIMAL(9,6), tip_amount DECIMAL(9,6), tolls_amount DECIMA
L(9,6), total_amount DECIMAL(9,6), trip_time_in_secs int )  ROW FORMAT DELIMITED  FIELDS TERMINATED BY ','  STORED as TEXTFILE  TBLPROPERTIES ("skip.head
er.line.count"="1");
OK
Time taken: 0.193 seconds
hive> show tables;
OK
taxidata
Time taken: 0.045 seconds, Fetched: 1 row(s)
hive>
```

Now we have to load data from hdfs to hive shell.

5) Hive> LOAD DATA INPATH 'newDataFlair/yellow_tripdata_2015-01-06.csv' INTO TABLE taxidata;

Now we have to check that from Hive shell & Hue Editor. (It is from UI SIDE)

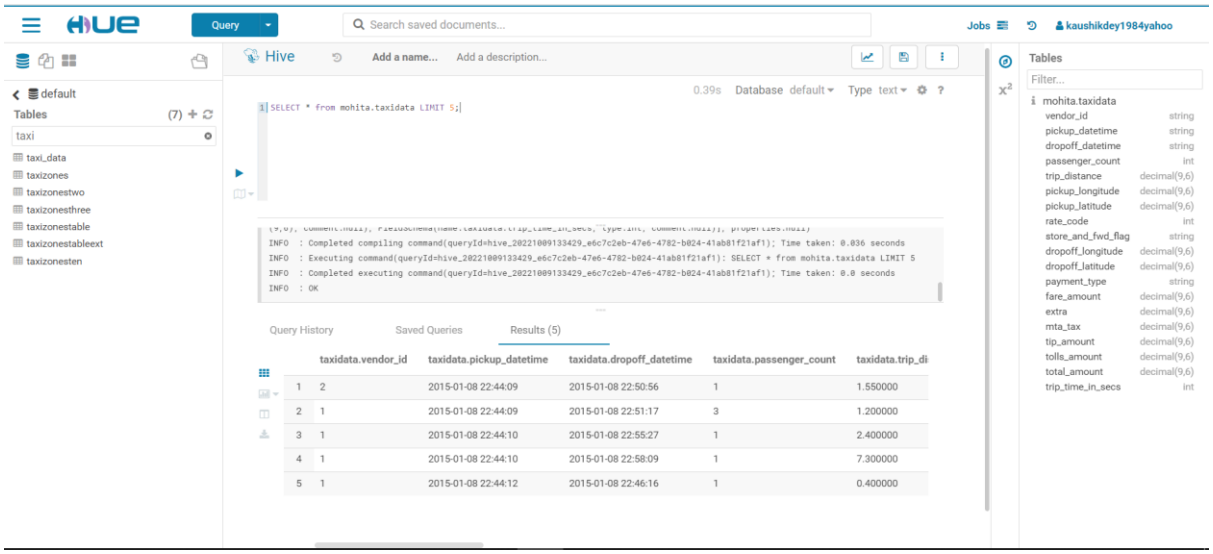
6) Hive> select * from mohita.taxidata LIMIT 5;

YELLOW Taxi trip analysis using Hive

ScreenShot from Hive Shell;

```
Time taken: 0.023 seconds
hive> SELECT * from taxidata LIMIT 5;
OK
2       2015-01-08 22:44:09     2015-01-08 22:50:56     1       1.550000        -73.987686      40.724251       1       N       -73.973763      40.743378
2       7.500000        0.500000        0.500000        0.000000        0.000000        8.800000        5000
1       2015-01-08 22:44:09     2015-01-08 22:51:17     3       1.200000        -73.991570      40.726933       1       N       -74.004105      40.721081
2       7.000000        0.500000        0.500000        0.000000        0.000000        8.300000        5344860
1       2015-01-08 22:44:10     2015-01-08 22:55:27     1       2.400000        -73.981918      40.783443       1       N       -73.952354      40.798199
2       10.500000       0.500000        0.500000        0.000000        0.000000        11.800000       3345464
1       2015-01-08 22:44:10     2015-01-08 22:58:09     1       7.300000        -73.973122      40.743553       1       N       -73.919571      40.832001
2       21.500000       0.500000        0.500000        0.000000        0.000000        22.800000       893933
1       2015-01-08 22:44:12     2015-01-08 22:46:16     1       0.400000        -73.982948      40.766209       1       N       -73.984390      40.764053
2       3.500000        0.500000        0.500000        0.000000        0.000000        4.800000        36864
Time taken: 0.212 seconds, Fetched: 5 row(s)
hive> SELECT * from mohita.taxidata LIMIT 5;
OK
2       2015-01-08 22:44:09     2015-01-08 22:50:56     1       1.550000        -73.987686      40.724251       1       N       -73.973763      40.743378
2       7.500000        0.500000        0.500000        0.000000        0.000000        8.800000        5000
1       2015-01-08 22:44:09     2015-01-08 22:51:17     3       1.200000        -73.991570      40.726933       1       N       -74.004105      40.721081
2       7.000000        0.500000        0.500000        0.000000        0.000000        8.300000        5344860
1       2015-01-08 22:44:10     2015-01-08 22:55:27     1       2.400000        -73.981918      40.783443       1       N       -73.952354      40.798199
2       10.500000       0.500000        0.500000        0.000000        0.000000        11.800000       3345464
1       2015-01-08 22:44:10     2015-01-08 22:58:09     1       7.300000        -73.973122      40.743553       1       N       -73.919571      40.832001
2       21.500000       0.500000        0.500000        0.000000        0.000000        22.800000       893933
1       2015-01-08 22:44:12     2015-01-08 22:46:16     1       0.400000        -73.982948      40.766209       1       N       -73.984390      40.764053
2       3.500000        0.500000        0.500000        0.000000        0.000000        4.800000        36864
Time taken: 0.063 seconds, Fetched: 5 row(s)
hive>
```

ScreenShot from Hue Editor;



Fourth-Step: in Fourth step we have to analysis the following questions & answers.

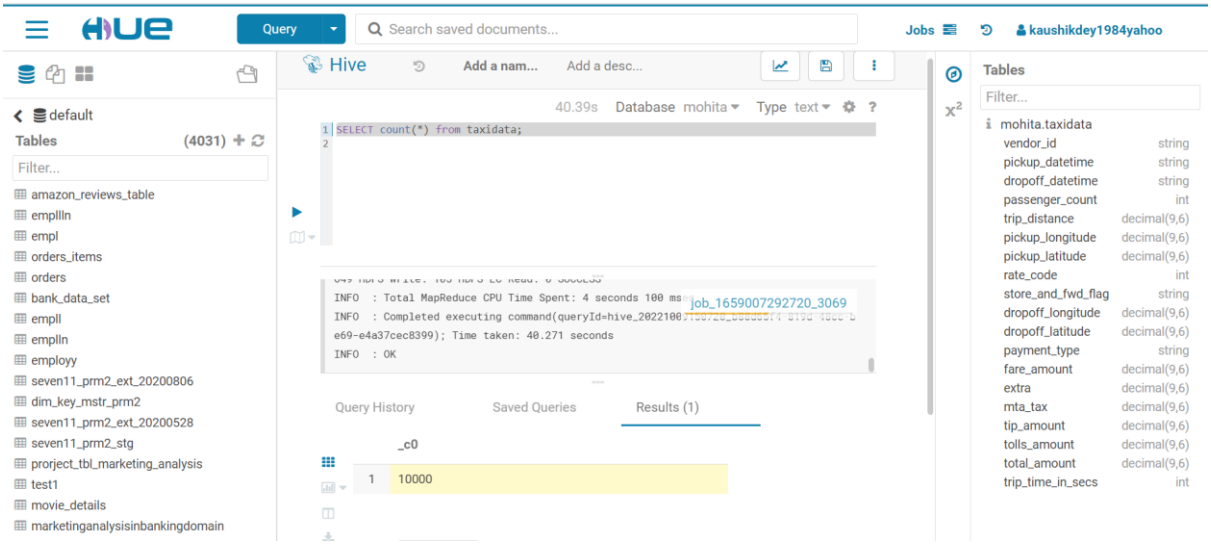
Perform taxi trip analysis by solving the questions below:

1. What is the total Number of trips (equal to the number of rows)?

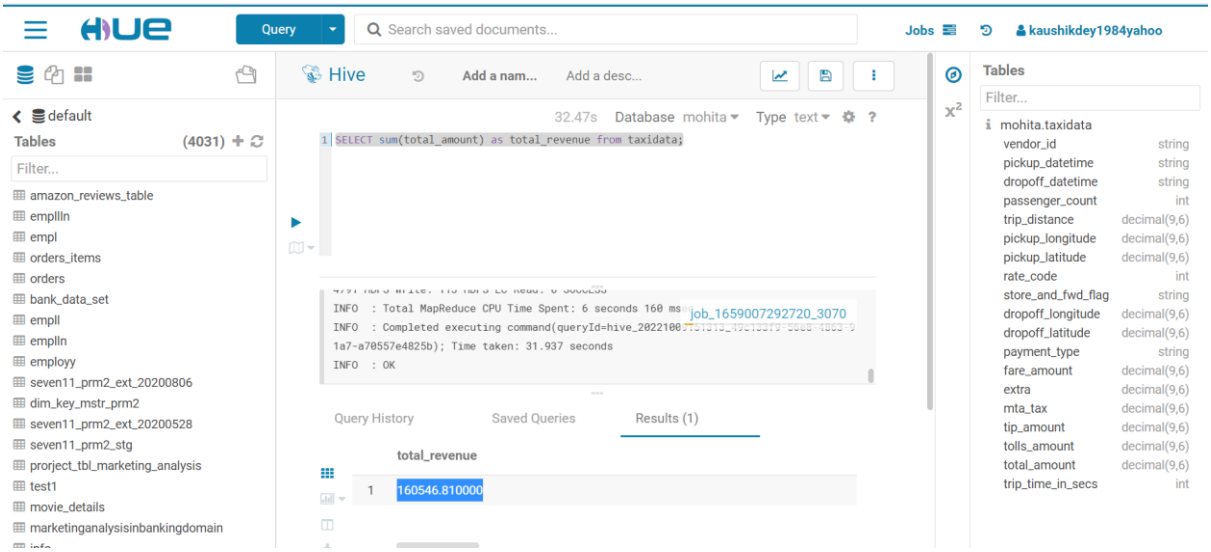
   SELECT count(*) from mohita.taxidata;

   Output : 10000

YELLOW Taxi trip analysis using Hive



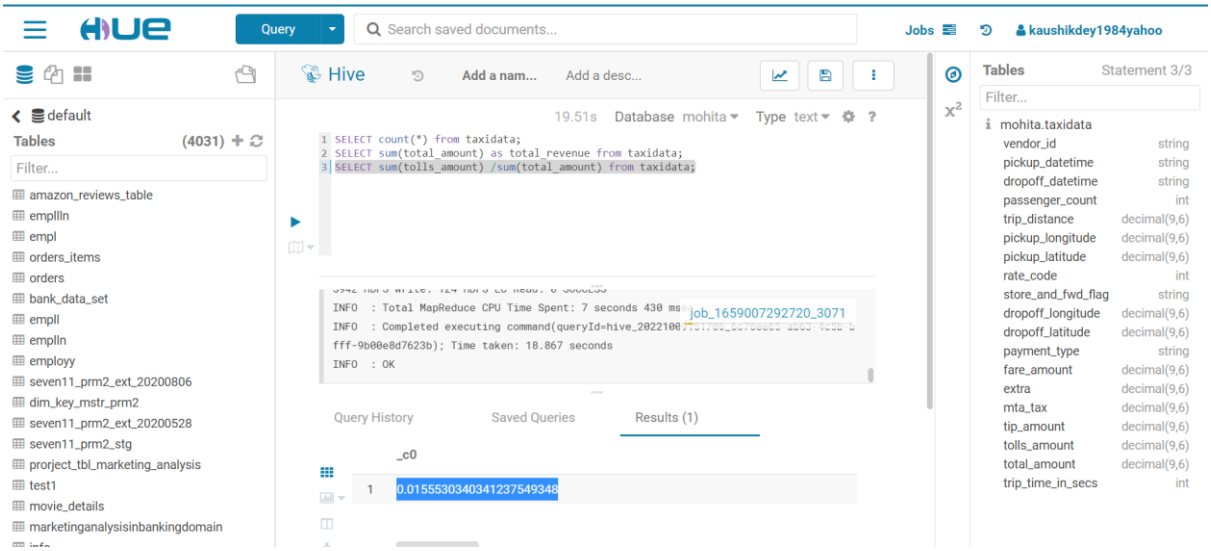
2) What is the total revenue generated by all the trips? The fare is stored in the column total_amount.
Ans : SELECT sum(total_amount) as total_revenue from taxidata;
Output : 160546.810000



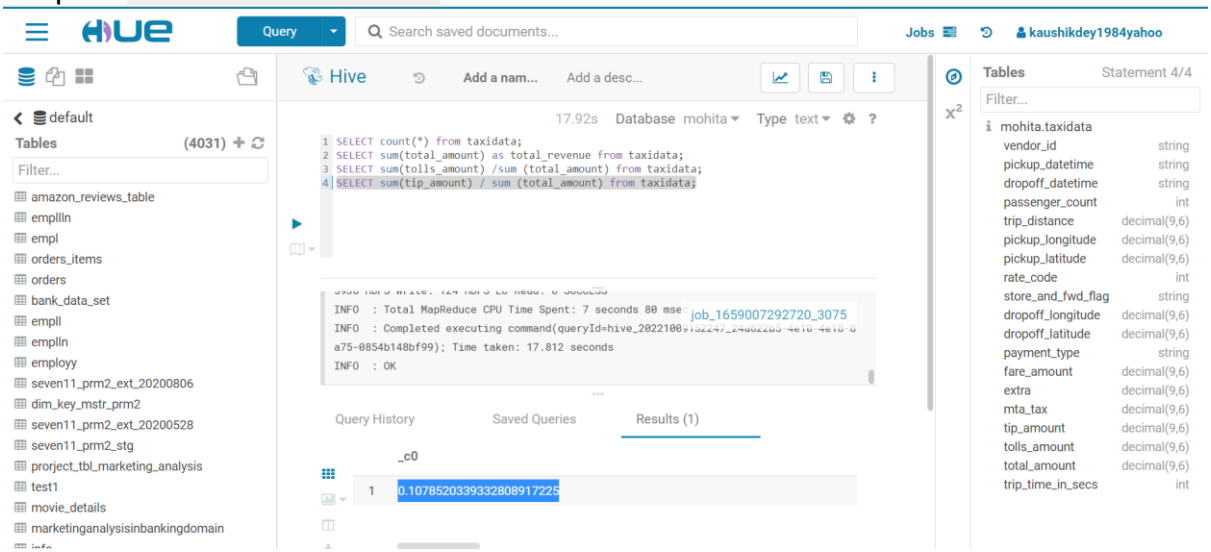
3) What fraction of the total is paid for tolls? The toll is stored in tolls_amount.
Ans : SELECT sum(tolls_amount) /sum(total_amount) from taxidata;

Output : 0.0155530340341237549348

YELLOW Taxi trip analysis using Hive

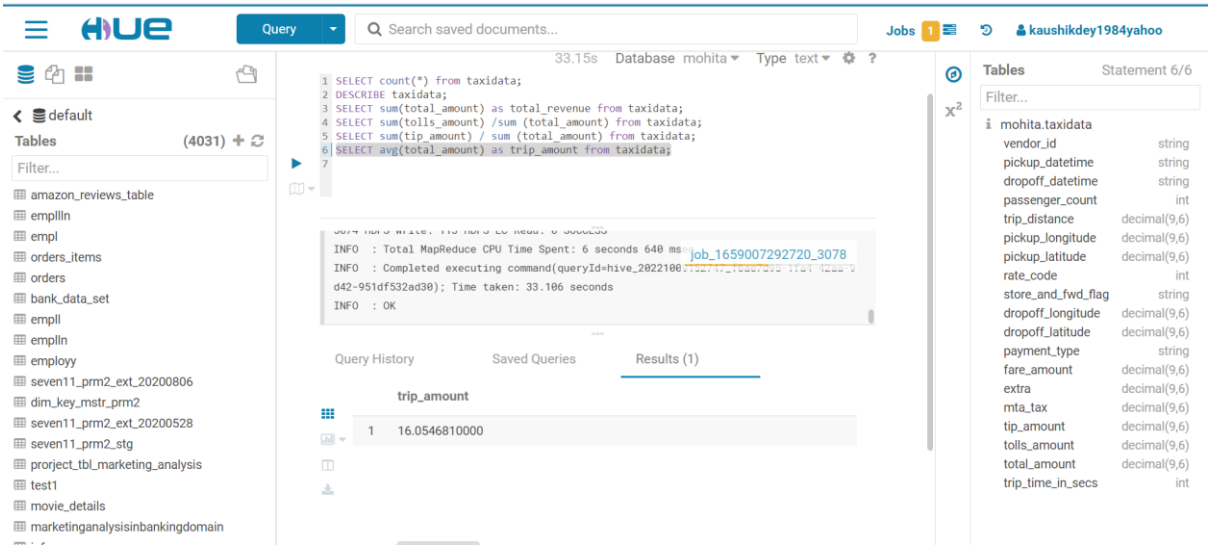


4) What fraction of it is driver tips? The tip is stored in tip_amount.
Ans : SELECT sum(tip_amount) / sum (total_amount) from taxidata;
Output : 0.1078520339332808917225



5) What is the average trip total amount?
Ans : SELECT avg(total_amount) as trip_amount from taxidata;
Output : 16.0546810000

YELLOW Taxi trip analysis using Hive



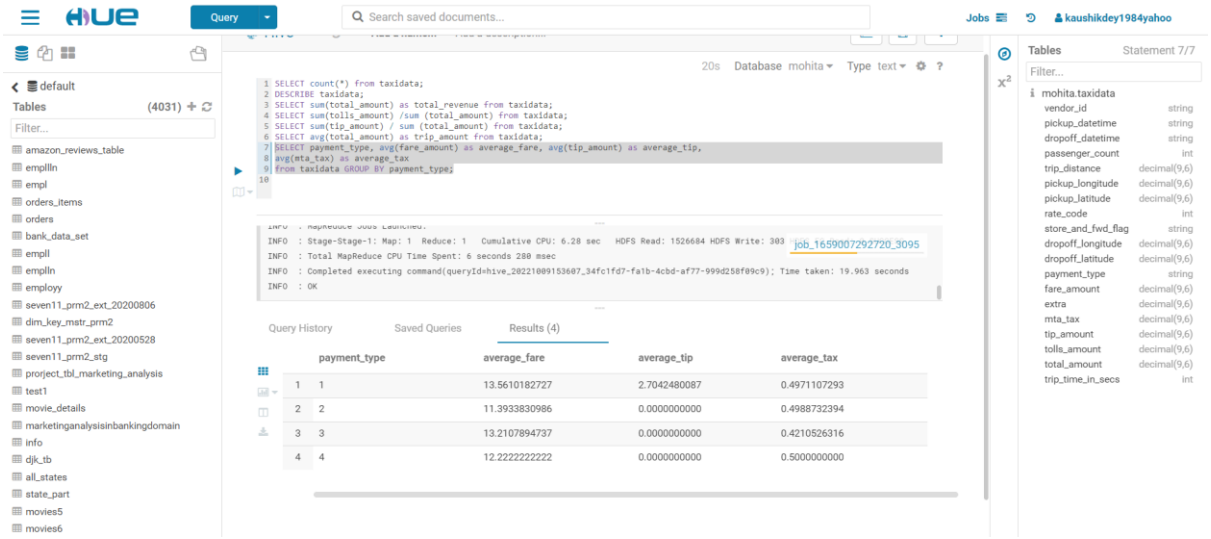
6) For each payment type, display the following details?
   a) Average fare generated
   b) Average tip
   c) Average tax – tax is stored in column mta_tax

Ans : SELECT payment_type, avg(fare_amount) as average_fare,
avg(tip_amount) as average_tip,
avg(mta_tax) as average_tax
from taxidata GROUP BY payment_type;
output :

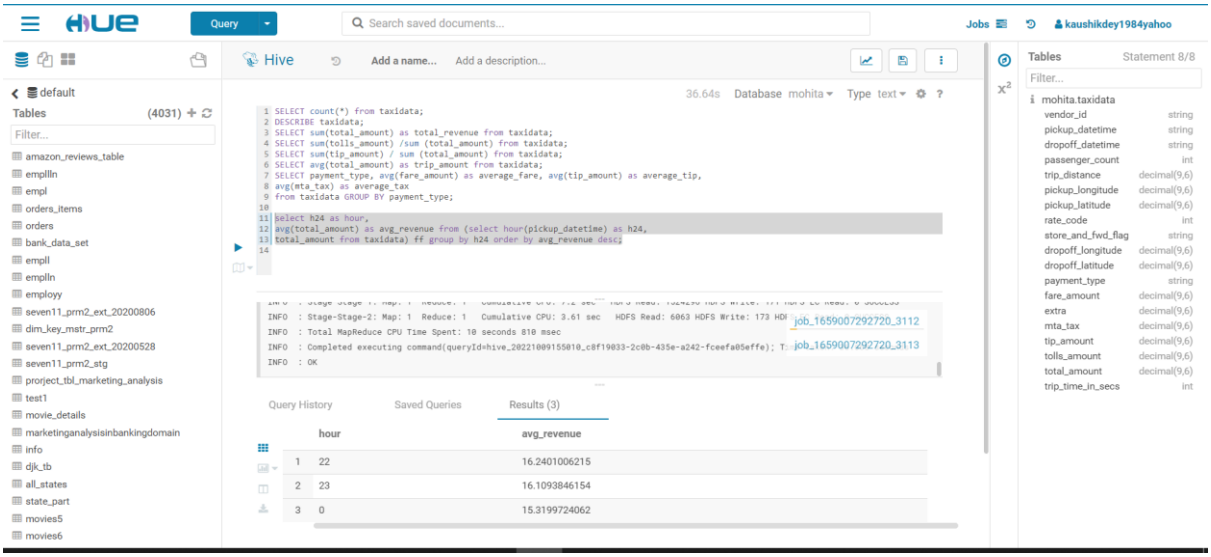
payment_type average_fare average_tip average_tax

	payment_type	average_fare	average_tip	average_tax
1	1	13.5610182727	2.7042480087	0.4971107293
2	2	11.3933830986	0.0000000000	0.4988732394
3	3	13.2107894737	0.0000000000	0.4210526316
4	4	12.2222222222	0.0000000000	0.5000000000

YELLOW Taxi trip analysis using Hive

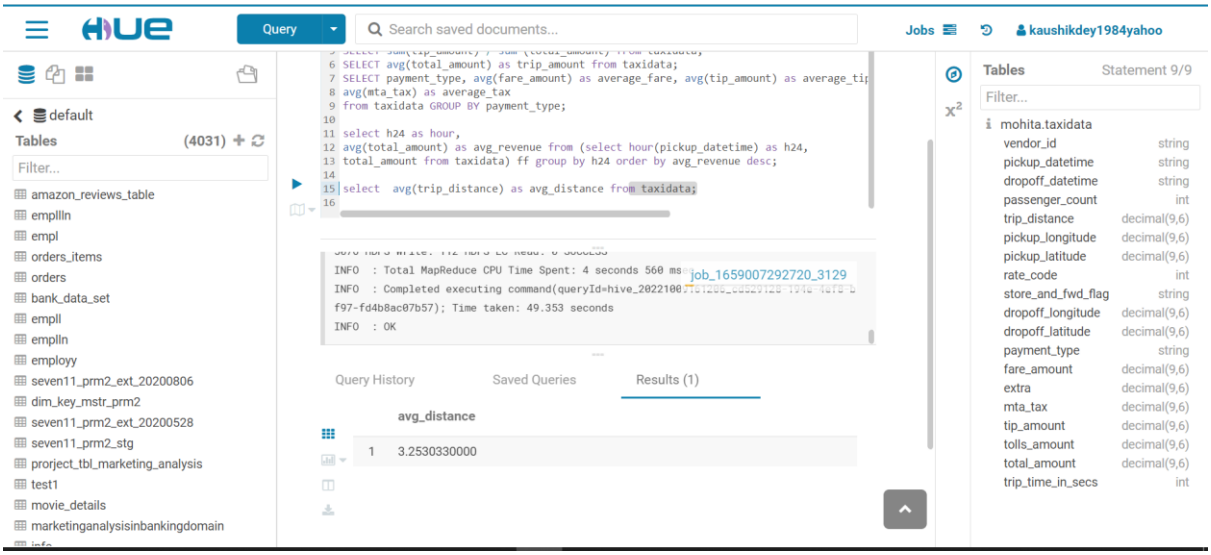


7) On an average which hour of the day generates the highest revenue?
Ans : select h24 as hour,
avg(total_amount) as avg_revenue from (select hour(pickup_datetime) as h24,
total_amount from taxidata) ff group by h24 order by avg_revenue desc;



8) What is the average distance of the trips? Distance is stored in the column trip_distance.
Ans : SELECT avg(trip_distance) as avg_distance from taxidata;
Output: 3.2530330000

YELLOW Taxi trip analysis using Hive



9) How many different payment types are used? Column name – payment_type. select distinct payment_type from taxidata
Ans : SELECT distinct payment_type from taxidata;
Output :