CIS5200 Term Project Tutorial

Authors: Anjali Baldawa; Dhruvi Patel; Digeshkumar Kansara

Instructor: Jongwook Woo

Date: 12/15/2019

Lab Tutorial

Anjali Baldawa (abaldaw@calstatela.edu)

Dhruvi Patel (dpatel86@calstatela.edu)

Digeshkumar Kansara (dkansar@calstatela.edu)

12/15/2019

New York Yellow Taxi Analysis Using Hadoop

Objectives

In this hands-on lab, you will learn how to:

- Upload and Download file from the local system to Hadoop HDFS and vice versa.
- Create table in HDFS using HiveQL.
- Create HiveQL commands to perform the analysis.
- Visualization result in Tableau and Excel using different charts.

Platform Spec

- Hadoop Cluster version : Hadoop 2.8.5-amzn-4
- HDFS capacity : 147 GB
- Storage : 678 GB
- Hive Version : Hive 2.3.5-amzn-1
- Cluster number of nodes : 5
- Memory size :139949854720 (130.34 GB)
- CPU Speed :2.28 GHZ
- CPU :20 vCPU

Step 1: Download Data and Upload to Amazon AWS HDFS

1) **The following steps explain how to download data from website.**

- To analyze New York taxi data we need to download dataset from the following link. https://chriswhong.com/open-data/foil_nyc_taxi/
- There are two types of dataset called **trip data and fare data.** We need to download both.

Note: The size of the files are huge so you need to have a good internet connection to download the dataset. Approximately, it would take 1 hour to download 1 file. The download time depends on the internet speed.

Once you download files into your local pc, you need to unzip the file in your local system.

2) **We need to upload the dataset from the file on remotely located Amazon AWS Hadoop Cluster.**

Note: The path after SCP is your file path of your pc where file is currently located. EXAMPLE: Our files are located on desktop and in 5200 pro folder. (Desktop/5200 pro)

- For username instead of **dkansar** you need to give **your own username**.
- Also, need to change **IP address** if you have different one.

SCP command to upload file in Hadoop cluster.

1) scp Desktop/5200 pro trip_data_1.csv dkansar@34.221.40.43:home/dkansar
2) scp Desktop/5200 pro trip_data_2.csv dkansar@34.221.40.43:home/dkansar
3) scp Desktop/5200 pro trip_data_3.csv dkansar@34.221.40.43:home/dkansar
4) scp Desktop/5200 pro trip_data_4.csv dkansar@34.221.40.43:home/dkansar

5) scp Desktop/fare/trip_fare_1.csv dkansar@34.221.40.43:/home/dkansar;
6) scp Desktop/fare/trip_fare_2.csv dkansar@34.221.40.43:/home/dkansar;
7) scp Desktop/fare/trip_fare_3.csv dkansar@34.221.40.43:/home/dkansar;

3) We need to remotely access our AWS EMR Hadoop cluster that we execute in our AWS account. Using putty or terminal (in windows, Linux or MAC), using the following command.

C:\Users\Asus>ssh **dkansar@34.221.40.43**

Note: Do not forget to change username and if you have different ip address.

And if we successfully connected then we can see the following output.

```
C:\Users\Asus>ssh dkansar@34.221.40.43
dkansar@34.221.40.43's password:
Last login: Sat Nov 30 18:25:01 2019 from 47.139.67.81

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
36 package(s) needed for security, out of 50 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R:::::::::::::::R
EE:::::EEEEEEEEE:::E M::::::::M       M::::::::M R:::::RRRRRR:::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R:::R      R::::R
  E:::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R:::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R::::::::::RR
  E:::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R:::RRRRRR::::R
  E::::E             M:::::M    M:::M    M:::::M   R:::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R:::R      R::::R
EE:::::EEEEEEEE:::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

-bash-4.2$
```

4)**List current directory to check if all the files uploaded successfully using 'ls -al'.**

```
-bash-4.2$ ls -al
-rw-r--r--  1 dkansar dkansar 2459600863 Oct 29 02:07 trip_data_1.csv
-rw-r--r--  1 dkansar dkansar 2328673265 Oct 29 02:19 trip_data_2.csv
-rw-r--r--  1 dkansar dkansar 2622301287 Oct 29 02:31 trip_data_3.csv
-rw-r--r--  1 dkansar dkansar 2515040578 Oct 29 02:41 trip_data_4.csv
-rw-r--r--  1 dkansar dkansar 2545680024 Oct 29 02:58 trip_data_5.csv
-rw-r--r--  1 dkansar dkansar 1681610043 Nov  8 20:34 trip_fare_1.csv
-rw-r--r--  1 dkansar dkansar 1593003695 Nov  8 20:40 trip_fare_2.csv
```

-rw-r--r-- 1 dkansar dkansar 1794836351 Nov 8 20:47 trip_fare_3.csv

5) Once the file uploaded we need to create a directory to store the trip data for analysis. Upload trip_data*.CSV file to the trip2 directory using following commands.
(Instead * need to put 1, 2, 3 and 4 for file name)

a) Create directory name **project.**
- bash-4.2$ hdfs dfs -mkdir project

b) Create directory name **trip 2** under project directory.
-bash-4.2$ hdfs dfs -mkdir project/trip2

c) Put **trip_data_*.csv** file from home directory to **project/trip2** directory.
-bash-4.2$ hdfs dfs -put trip_data_1.csv project/trip2
-bash-4.2$ hdfs dfs -put trip_data_2.csv project/trip2
-bash-4.2$ hdfs dfs -put trip_data_3.csv project/trip2
-bash-4.2$ hdfs dfs -put trip_data_4.csv project/trip2

d) To check file uploaded successfully, use below command.
-bash-4.2$ hdfs dfs -ls project/trip2
Found 4 items
-rw-r--r-- 1 dkansar hadoop 2459600863 2019-11-05 03:56 project/trip2/trip_data_1.csv
-rw-r--r-- 1 dkansar hadoop 2328673265 2019-11-05 03:54 project/trip2/trip_data_2.csv
-rw-r--r-- 1 dkansar hadoop 2622301287 2019-11-05 06:16 project/trip2/trip_data_3.csv
-rw-r--r-- 1 dkansar hadoop 2515040578 2019-11-05 06:17 project/trip2/trip_data_4.csv

6) Repeat the above step to upload trip_fare*.csv to the fair1 directory.
(Instead * need to put 1, 2, 3 and 4 for file name)

a) Create directory name **fair.**
- bash-4.2$ hdfs dfs -mkdir fair.

b) Create directory name **fair1** under project directory.
-bash-4.2$ hdfs dfs -mkdir fair/fair1

c) Put trip_fair_*.csv file from home directory to fair/fair1 folder:
-bash-4.2$ hdfs dfs -put trip_fair_1.csv fair/fair1
-bash-4.2$ hdfs dfs -put trip_fair_2.csv fair/fair1
-bash-4.2$ hdfs dfs -put trip_fair_3.csv fair/fair1

d) To check file uploaded successfully, use below command.
-bash-4.2$ hdfs dfs -ls fair/fair1
Found 3 items
-rw-r--r-- 1 dkansar hadoop 1681610043 2019-11-08 22:34 fair/fair1/trip_fare_1.csv
-rw-r--r-- 1 dkansar hadoop 1593003695 2019-11-08 22:34 fair/fair1/trip_fare_2.csv

-rw-r--r-- 1 dkansar hadoop 1794836351 2019-11-08 22:35 fair/fair1/trip_fare_3.csv

Step 2: Create Tables for Analysis and Analysis using tools

1) The following hive statement creates two new table named taxi and fair, by describing the fields within the file the delimiter (Comma) between fields. External table preserves data in original file format, while allowing hive to perform query against data.

Open Beeline Command Line Interface using following command to run hive queries.

beeline -u jdbc:hive2://localhost:10000/default -n **your_username**

Note: Instead of dkansar you need to use your_username.

```
-bash-4.2$ beeline -u jdbc:hive2://localhost:10000/default -n dkansar
Connecting to jdbc:hive2://localhost:10000/default
Connected to: Apache Hive (version 2.3.5-amzn-1)
Driver: Hive JDBC (version 2.3.5-amzn-1)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 2.3.5-amzn-1 by Apache Hive
0: jdbc:hive2://localhost:10000/default>
```

Now you have to create your database to separate your tables from other users table. For example, you can create database using the following command.

Note: Instead of dkansar you need to use **your_username.**

0: jdbc:hive2://localhost:10000/default> CREATE DATABASE IF NOT EXISTS **dkansar;**
No rows affected (0.231 seconds)
0: jdbc:hive2://localhost:10000/default>

Use the below command to check if database created or not.

0: jdbc:hive2://localhost:10000/default> show databases;

```
+----------------+
| database_name  |
+----------------+
| abaldaw        |
| amedin62       |
| bfaraji        |
| bliang12       |
| calipio        |
| default        |
| dkansar        |
```

Now you need to use your database, which you created before using the following command.

0: jdbc:hive2://localhost:10000/default> use **dkansar;**
No rows affected (0.323 seconds)

Note: Do not forget to change the username instead of dkansar.

Table 1) In the beeline CLI Copy and Paste the following HiveQL and code to use your database and to create external table taxi.

Note: Do not forget to change the username instead of dkansar.

CREATE EXTERNAL TABLE IF NOT EXISTS taxi (
Medallion STRING, Hack_License STRING, Vendor_Id STRING, Rate_Code INT, Store_fwd_flag STRING, Pickup_date TIMESTAMP, Passenger_Count BIGINT, Trip_Time_in_sec BIGINT, Trip_Distance BIGINT, Pickup_Longitude STRING, Pickup_Latitude STRING, Dropoff_Longitude STRING, Dropoff_Latitude STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/**dkansar**/project/trip2'
TBLPROPERTIES ('skip.header.line.count'='2');

After create query check if taxi table created successfully or not. For that use the below query
0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| sunday             |
| taxi               |
+--------------------+
62 rows selected (0.312 seconds)
```

If you can't see the table name then table is not created and you have to follow the same step again.

After successfully creating the table. Now we can query the contents of taxi table.

0: jdbc:hive2://localhost:10000/default> select * from taxi limit 5;

```
0: jdbc:hive2://localhost:10000/default> select * from taxi limit 5;
+-----------------------------------+-----------------------------------+----------------+----------------+----------------------+-----------------------+-----------------------+-----------------
--------+-----------------------+-------------------+-----------------------+-----------------------+-----------------------+-----------------------+
|         taxi.medallion            |         taxi.hack_license         | taxi.vendor_id | taxi.rate_code | taxi.store_fwd_flag |   taxi.dropoff_date    | taxi.passenger
_count | taxi.trip_time_in_sec | taxi.trip_distance | taxi.pickup_longitude | taxi.pickup_latitude | taxi.dropoff_longitude | taxi.dropoff_latitude |
+-----------------------------------+-----------------------------------+----------------+----------------+----------------------+-----------------------+-----------------
--------+-----------------------+-------------------+-----------------------+-----------------------+-----------------------+-----------------------+
| 0BD7C8F5BA12B88E0B67BED28BEA73D8  | 9FD8F69F0804BDB5549F40E9DA1BE472  | CMT            | 1              | N                    | 2013-01-06 00:18:35.0 | NULL
       | 1                     | 259               | 1.50                  | -74.006683            | 40.731781             | -73.994499            |
| 0BD7C8F5BA12B88E0B67BED28BEA73D8  | 9FD8F69F0804BDB5549F40E9DA1BE472  | CMT            | 1              | N                    | 2013-01-05 18:49:41.0 | NULL
       | 1                     | 282               | 1.10                  | -74.004707            | 40.73777              | -74.009834            |
| DFD2202EE08F7A8DC9A57B02ACB81FE2  | 51EE87E3205C985EF8431D850C786310  | CMT            | 1              | N                    | 2013-01-07 23:54:15.0 | NULL
       | 2                     | 244               | .70                   | -73.974602            | 40.759945             | -73.984734            |
| DFD2202EE08F7A8DC9A57B02ACB81FE2  | 51EE87E3205C985EF8431D850C786310  | CMT            | 1              | N                    | 2013-01-07 23:25:03.0 | NULL
       | 1                     | 560               | 2.10                  | -73.97625             | 40.748528             | -74.002586            |
| 20D9ECB2CA0767CF7A01564DF2844A3E  | 598CCE5B9C1918568DEE71F43CF26CD2  | CMT            | 1              | N                    | 2013-01-07 15:27:48.0 | NULL
       | 1                     | 648               | 1.70                  | -73.966743            | 40.764252             | -73.983322            |
+-----------------------------------+-----------------------------------+----------------+----------------+----------------------+-----------------------+-----------------
--------+-----------------------+-------------------+-----------------------+-----------------------+-----------------------+-----------------------+
5 rows selected (0.393 seconds)
0: jdbc:hive2://localhost:10000/default>
```

We can also see the structure of the table.

```
0: jdbc:hive2://localhost:10000/default> describe  taxi;
+---------------------+------------+----------+
|      col_name       | data_type  | comment  |
+---------------------+------------+----------+
| medallion           | string     |          |
| hack_license        | string     |          |
| vendor_id           | string     |          |
| rate_code           | int        |          |
| store_fwd_flag      | string     |          |
| dropoff_date        | timestamp  |          |
| passenger_count     | bigint     |          |
| trip_time_in_sec    | bigint     |          |
| trip_distance       | bigint     |          |
| pickup_longitude    | string     |          |
| pickup_latitude     | string     |          |
| dropoff_longitude   | string     |          |
| dropoff_latitude    | string     |          |
+---------------------+------------+----------+
13 rows selected (0.288 seconds)
0: jdbc:hive2://localhost:10000/default>
```

Table 2) In beeline CLI Copy and Paste the following HiveQL to create external table fair.

Note: Do not forget to change the username instead of dkansar.

CREATE EXTERNAL TABLE IF NOT EXISTS fair (

Medallion STRING, Hack_License STRING, Venodr_Id STRING, Pickup_Datetime TIMESTAMP, Payment_Type STRING, Fare_Amount FLOAT, Surcharge FLOAT, Mta_Tax FLOAT, Tip_Amount FLOAT, Tolls_Amout FLOAT, Total_Amount FLOAT)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

STORED AS TEXTFILE LOCATION '/user/dkansar/fair/fair1'

TBLPROPERTIES ('skip.header.line.count'='2');

Now we can check if fair table created or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| fair               |
| firstmonth         |
| taxi               |
+--------------------+
61 rows selected (0.324 seconds)
```

We can also see the structure of the fair table.

0: jdbc:hive2://localhost:10000/default> describe fair;

```
0: jdbc:hive2://localhost:10000/default> describe fair;
+------------------+------------+----------+
|     col_name     | data_type  | comment  |
+------------------+------------+----------+
| medallion        | string     |          |
| hack_license     | string     |          |
| venodr_id        | string     |          |
| pickup_datetime  | timestamp  |          |
| payment_type     | string     |          |
| fare_amount      | float      |          |
| surcharge        | float      |          |
| mta_tax          | float      |          |
| tip_amount       | float      |          |
| tolls_amout      | float      |          |
| total_amount     | float      |          |
| pickup_date      | timestamp  |          |
+------------------+------------+----------+
12 rows selected (0.312 seconds)
0: jdbc:hive2://localhost:10000/default>
```

We can check the content of the fair table too.

0: jdbc:hive2://localhost:10000/default> select * from fair limit 5;

```
0: jdbc:hive2://localhost:10000/default> select * from fair limit 5;
+----------------------------------+----------------------------------+----------------+-----------------------+-------------------+-----------------+-------------
--+--------------+-----------------+-----------------+-----------------+-----------------+
|          fair.medallion          |        fair.hack_license         | fair.venodr_id |  fair.pickup_datetime  | fair.payment_type | fair.fare_amount | fair.surcharge
  | fair.mta_tax | fair.tip_amount | fair.tolls_amout | fair.total_amount | fair.pickup_date |
+----------------------------------+----------------------------------+----------------+-----------------------+-------------------+-----------------+-------------
--+--------------+-----------------+-----------------+-----------------+-----------------+
| 0BD7C8F5BA12B88E0B67BED28BEA73D8 | 9FD8F69F0804BDB5549F40E9DA1BE472 | CMT            | 2013-01-06 00:18:35.0 | CSH               | 6.0              | 0.5
  | 0.5          | 0.0             | 0.0             | 7.0             | NULL             |
| 0BD7C8F5BA12B88E0B67BED28BEA73D8 | 9FD8F69F0804BDB5549F40E9DA1BE472 | CMT            | 2013-01-05 18:49:41.0 | CSH               | 5.5              | 1.0
  | 0.5          | 0.0             | 0.0             | 7.0             | NULL             |
| DFD2202EE08F7A8DC9A57B02ACB81FE2 | 51EE87E3205C985EF8431D850C786310 | CMT            | 2013-01-07 23:54:15.0 | CSH               | 5.0              | 0.5
  | 0.5          | 0.0             | 0.0             | 6.0             | NULL             |
| DFD2202EE08F7A8DC9A57B02ACB81FE2 | 51EE87E3205C985EF8431D850C786310 | CMT            | 2013-01-07 23:25:03.0 | CSH               | 9.5              | 0.5
  | 0.5          | 0.0             | 0.0             | 10.5            | NULL             |
| 20D9ECB2CA0767CF7A01564DF2844A3E | 598CCE5B9C1918568DEE71F43CF26CD2 | CMT            | 2013-01-07 15:27:48.0 | CSH               | 9.5              | 0.0
  | 0.5          | 0.0             | 0.0             | 10.0            | NULL             |
+----------------------------------+----------------------------------+----------------+-----------------------+-------------------+-----------------+-------------
--+--------------+-----------------+-----------------+-----------------+-----------------+
5 rows selected (0.394 seconds)
0: jdbc:hive2://localhost:10000/default>
```

Table 3) We are creating another table called "Analysis" by selecting the following fields hack_licence,Fare_Amount,surcharge.mta_tax,tip,tolls and total amount. And it extracts pickup_date , Day and pickup hour from the TIMESTAMP.

Note: Do not forget to change the username instead of dkansar.

CREATE EXTERNAL TABLE IF NOT EXISTS analysis (

Hack_License STRING, Fare_Amount FLOAT, Surcharge FLOAT, Mta_Tax FLOAT, Tip_Amount FLOAT, Tolls_Amout FLOAT, Total_Amount FLOAT, pickup_date DATE, Day INT, pickup_hour STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

STORED AS TEXTFILE LOCATION '/user/dkansar/analysis/'

TBLPROPERTIES ('skip.header.line.count'='2');

This query overwrite data into analysis table from the fair table. You need to copy and paste to hive beeline CLI.

INSERT OVERWRITE TABLE analysis SELECT Hack_License,Fare_Amount,Surcharge,Mta_Tax,Tip_Amount,Tolls_Amout,Total_Amount, TO_DATE(pickup_datetime),date_format(pickup_datetime ,'u'),hour(pickup_datetime) FROM fair;

Now check if table created or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| afternoon          |
| amount             |
| analysis           |
| wednesday          |
+--------------------+
61 rows selected (0.336 seconds)
```

Note: Number of tables differ from one database to another.

We can also see the structure of the fair table.

0: jdbc:hive2://localhost:10000/default> describe analysis;

```
0: jdbc:hive2://localhost:10000/default> describe analysis;
+--------------+-----------+----------+
|   col_name   | data_type | comment  |
+--------------+-----------+----------+
| hack_license | string    |          |
| fare_amount  | float     |          |
| surcharge    | float     |          |
| mta_tax      | float     |          |
| tip_amount   | float     |          |
| tolls_amout  | float     |          |
| total_amount | float     |          |
| pickup_date  | date      |          |
| day          | int       |          |
| pickup_hour  | string    |          |
+--------------+-----------+----------+
10 rows selected (0.301 seconds)
```

We can check the content of the fair table too.

0: jdbc:hive2://localhost:10000/default> select * from analysis limit 5;

```
0: jdbc:hive2://localhost:10000/default> select * from analysis limit 5;
+----------------------------------+----------------------+---------------------+-------------------+-----------------
--+----------------------+---------------+----------------------+
|       analysis.hack_license       | analysis.fare_amount  | analysis.surcharge  | analysis.mta_tax  | analysis.tip_amo
  | analysis.pickup_date  | analysis.day  | analysis.pickup_hour  |
+----------------------------------+----------------------+---------------------+-------------------+-----------------
--+----------------------+---------------+----------------------+
| DF142C5256392C3CDA67C7DFFA5B88E2  | 25.5                 | 0.5                 | 0.5               | 6.26
  | 2013-03-01            | 5             | 0                    |
| 10D2E58D75E07D6B5AAFA329BF5A4CAC  | 6.5                  | 0.5                 | 0.5               | 1.0
  | 2013-03-01            | 5             | 0                    |
| A38CE69F84E515A71DF18F9786F13690  | 15.0                 | 0.5                 | 0.5               | 2.0
  | 2013-03-01            | 5             | 0                    |
| 3DD549B1A5F1EF31372BF2933B2C8D54  | 10.0                 | 0.5                 | 0.5               | 2.0
  | 2013-03-01            | 5             | 0                    |
| DF4C8A343928E47C39E93A012920ABD3  | 11.5                 | 0.5                 | 0.5               | 3.75
  | 2013-03-01            | 5             | 0                    |
+----------------------------------+----------------------+---------------------+-------------------+-----------------
--+----------------------+---------------+----------------------+
5 rows selected (0.397 seconds)
```

Table 4)

Analysis 1:

Step1 : We are creating a table and running queries to find out which day of the week has more number of trips.

Following hive query creates day table.

Note: Do not forget to change the username instead of dkansar.

CREATE EXTERNAL TABLE IF NOT EXISTS day (

day int, trip_count bigint)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

STORED AS TEXTFILE LOCATION '/user/dkansar/day/'

TBLPROPERTIES ('skip.header.line.count'='2');

Following hive query Overwrite data into day table from fair table.

INSERT OVERWRITE TABLE day select day, count (day) from analysis GROUP BY day ORDER BY day ASC;

We can check table created successfully or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| afternoon          |
| clienterrors       |
| day                |
+--------------------+
61 rows selected (0.372 seconds)
```

Note: No of tables different for different database based on how many, you have created.

We can check the content of the table.

0: jdbc:hive2://localhost:10000/default> select * from day;

```
0: jdbc:hive2://localhost:10000/default> select * from day;
+----------+-----------------+
| day.day  | day.trip_count  |
+----------+-----------------+
| 3        | 6419452         |
| 4        | 6739668         |
| 5        | 6913028         |
| 6        | 6916630         |
| 7        | 6060201         |
+----------+-----------------+
5 rows selected (0.321 seconds)
```

Step 2: After creation and insertion of data in a table we need to download file in your local system to visualize the table in tableau.

a) Run the following HDFS shell at beeline CLI to list what file exists at day directory "/user/dkanar/day" that is actually the location of Hive table. It is a file named "000000_0":

Note: Do not forget to change the username instead of dkansar.

0: jdbc:hive2://localhost:10000/default> dfs -ls day;

```
0: jdbc:hive2://localhost:10000/default> dfs -ls day;
+----------------------------------------------------+
|                     DFS Output                     |
+----------------------------------------------------+
| Found 1 items                                      |
| -rwxr-xr-x   1 dkansar hadoop         70 2019-12-03 17:04 day/000000_0 |
+----------------------------------------------------+
2 rows selected (0.01 seconds)
```

b) Quit from the Beeline CLI. And, in shell CLI, you need execute hdfs dfs –get to download HDFS file 000000_0 to your AWS master node, which is a file named "day.csv" below:

Note: Do not forget to change the username instead of dkansar.

-bash-4.2$ hdfs dfs -get /user/dkansar/day/000000_0

Note: If you run above command and it give you error File already exists then follow following command. **Which remove 000000_0 file from Hadoop.**

rm –rf 000000_0

-bash-4.2$ hdfs dfs -get day/000000_0 day.csv

-bash-4.2$ pwd

/home/dkansar

-bash-4.2$ ls

```
-bash-4.2$ ls
\              afternoon.csv  Drop_analysis.csv  Fmonth.csv       labPigETL
000000_0       amount.csv     dropoff.csv        genre.java       location.csv
000001_0       analysis.csv   dropoffl.csv       geolocation.csv  long_distance.csv
000002_0       day.csv        first_etl.pi       hour1.csv        __MACOSX
ad_data1.txt   daytime.csv    first_etl.pig      hour2.csv        midnight.csv
```

c) Now need to download file in the local system.

It is easy for shell terminal – Git Bash, Minty, Linux/Mac terminal - using scp:

Note: Do not forget to change the username instead of dkansar.

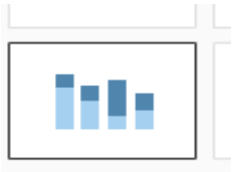
C:\Users\Asus\Desktop\analysis>scp dkansar@34.221.40.43:~/day.csv .
dkansar@34.221.40.43's password:
day.csv 100% 1596 1.6KB/s 00:00

Step 3: Once the file downloaded successfully you have to upload the csv file in tableau. you can follow the below steps to upload file to tableau.

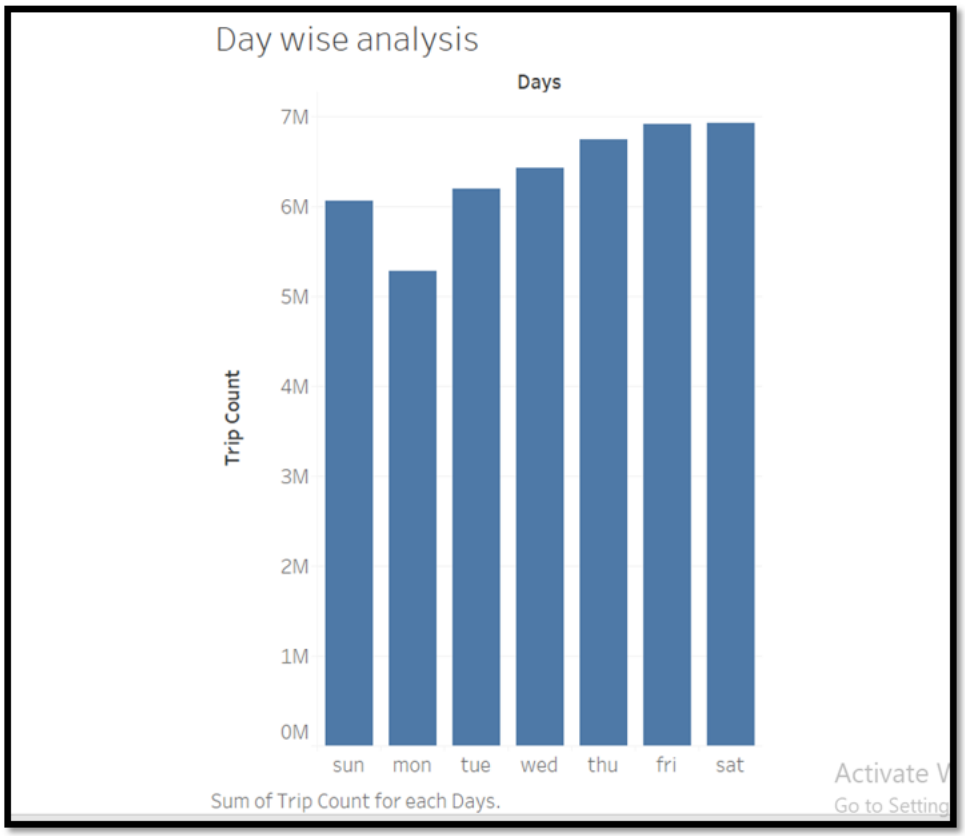
- Open tableau
- In left side **connect** menu in **In File** menu go to more option.
- Where you can choose your day.csv file.
- Open this file.
- Need to rename all fields.
- Go to sheet1.

Step 4: Visualizing data in tableau for day.csv file.

- In this file, you need to drag dimension to the column part and trip_count to the row part. And need to select following bar chart.



And you can see the following chart.



The column represents a day of the week and row represents trip count.

By seeing the graph, we can conclude that Monday is having the lowest number of booked trips, while Friday, Saturday has the highest number of trips in a week.

Table 5)
Analysis 2:

Step1: We are creating a new table and running queries to find out the Trends in number of trips over the 3 months.

Following hive query creates **dates** table.
Note: Do not forget to change the username instead of dkansar.

```
CREATE EXTERNAL TABLE IF NOT EXISTS dates (
pickup_date DATE, hour INT, trip_count BIGINT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/dkansar/dates/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

Following hive query Overwrite data into dates table from analysis table.

```
INSERT OVERWRITE TABLE dates SELECT pickup_date, pickup_hour, COUNT (*) no_of_trips
FROM analysis GROUP BY pickup_date, pickup_hour order by pickup_date ASC;
```

We can check table created successfully or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| clienterrors       |
| dates              |
| day|
+--------------------+
```

We can check the content of the table.

0: jdbc:hive2://localhost:10000/default> select * from dates limit 5;

```
0: jdbc:hive2://localhost:10000/default> sselect * from dates limit 5;
+--------------------+-------------+-------------------+
| dates.pickup_date  | dates.hour  | dates.trip_count  |
+--------------------+-------------+-------------------+
| 2013-01-01         | 4           | 18271             |
| 2013-01-01         | 15          | 19734             |
| 2013-01-01         | 20          | 16363             |
| 2013-01-01         | 12          | 18551             |
| 2013-01-01         | 23          | 11079             |
+--------------------+-------------+-------------------+
5 rows selected (0.36 seconds)
```

Step 2: After creation and insertion of data in a table we need to download file in your local system to visualize the table in tableau.

Note: Step 2 to download file in local system is same for all analysis so we need to Follow step 2 from page 12 (Part a, b, c). Do not forget to change current table name

Dates.

Note: Do not forget to change current table name dates.

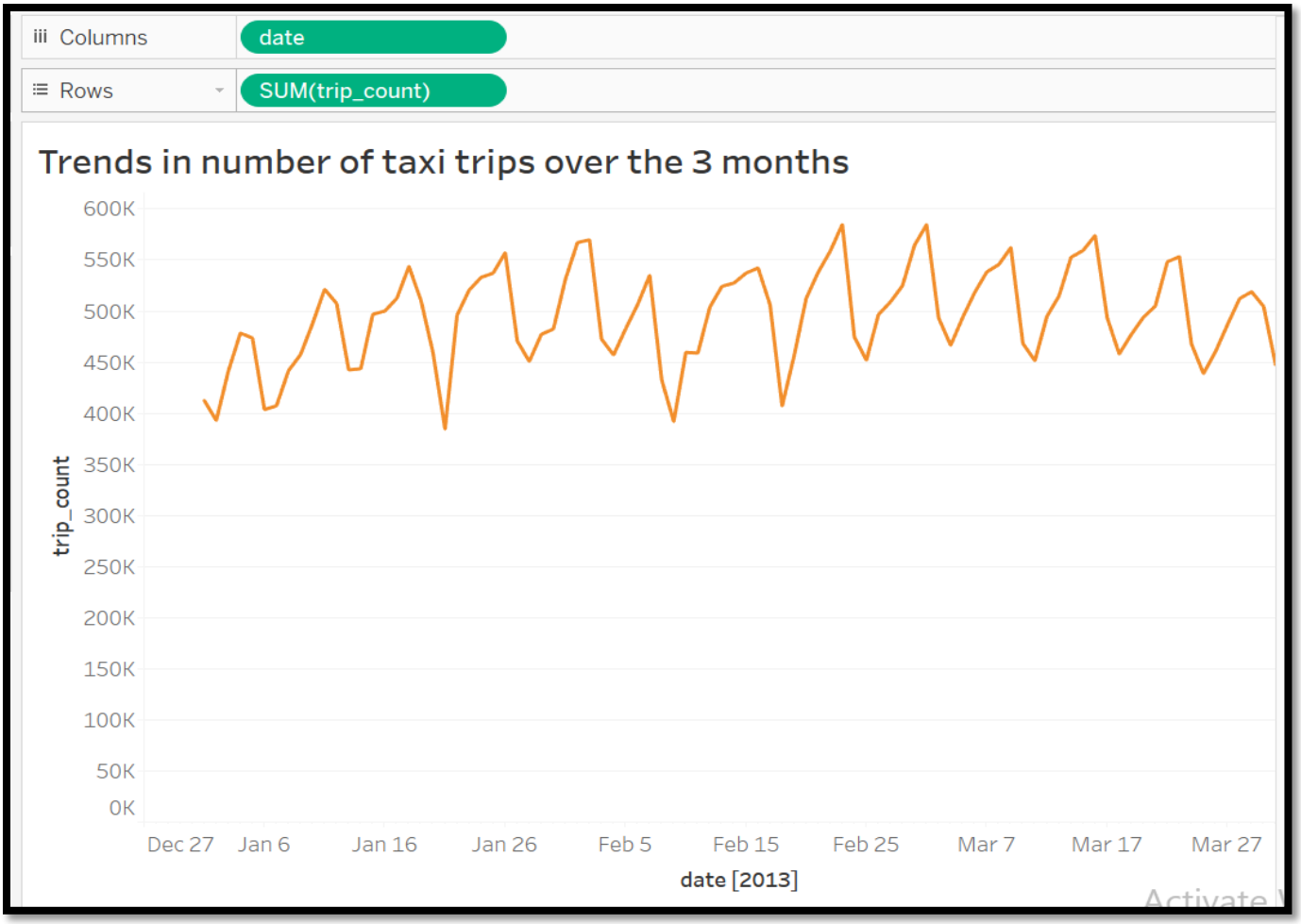
Step 3: Once the file downloaded successfully you, have to upload the csv file in tableau. You can follow the below steps to upload file to tableau.

Note: Step 3 to download upload file to tableau is same for all analysis. So we need to Follow step 3 from analysis 1 page 13.

Note: Do not forget to change current file name dates.csv.

Step 4: Visualizing data in tableau for dates.csv file.

- In this file, you need to drag dimension date to the column and measures trip_count to the row and need to select following line chart.



- The above graph shows the analysis of trip count during the three months on a particular day. The row represents the trip count and the columns represents dates of three months.
- The orange line in the graph represents the trip count on a particular day for the three months.
- We analyzed some pattern from the chart.
- It can be depicted that trip count for Monday is low and for Saturday, it is the highest. Also on 21 January on Martin Luther King day and 14th February on president day the trip count was low. We can assume that trips are low on public holidays.

Table 6) following dropoff_day table extracts hour and day from timestamp and selects trip_distance from taxi table. This table use for further analysis.

Following hive query creates **dropoff_day** table.

Note: Do not forget to change the username instead of dkansar.

CREATE EXTERNAL TABLE IF NOT EXISTS dropoff_day(

total_distance BIGINT day INT, hour STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

STORED AS TEXTFILE LOCATION '/user/dkansar/dropoff_day/'

TBLPROPERTIES ('skip.header.line.count'='2');

Following hive query Overwrite data into dropoff_day table from taxi table.

INSERT OVERWRITE TABLE dropoff_day SELECT trip_distance,date_format(Dropoff_Date ,'u'),hour(Dropoff_Date) FROM taxi ;

We can check table created successfully or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| drop_analysis      |
| dropoff_day        |
| dropoff_location   |
| fair               |
+--------------------+
51 rows selected (0.372 seconds)
```

Table 6)

Analysis 3:

Step1: we are creating a table and running queries to analyze pattern of trip_count During particular day and hour of the day.

Following hive query creates **drop_analysis** table.

Note: Do not forget to change the username instead of dkansar.

CREATE EXTERNAL TABLE IF NOT EXISTS Drop_analysis (

trip_distance Day INT, hour INT, Trip_count BIGINT)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

STORED AS TEXTFILE LOCATION '/user/dkansar/Drop_analysis/'

TBLPROPERTIES ('skip.header.line.count'='2');

Following hive query Overwrite data into drop_analysis table from dropoff_day table.

INSERT OVERWRITE TABLE Drop_analysis SELECT Day, hour, count (*) from taxi GROUP BY day, hour ORDER BY Day, hour Asc;

We can check table created successfully or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| dictionary         |
| drivers            |
| drop_analysis      |
| dropoff_day        |
+--------------------+
61 rows selected (0.372 seconds)
```

We can check the content of the table.

0: jdbc:hive2://localhost:10000/default> select * from drop_analysis limit 10;

```
0: jdbc:hive2://localhost:10000/default> select * from drop_analysis limit 10;
+--------------------+---------------------+---------------------------+
| drop_analysis.day  | drop_analysis.hour  | drop_analysis.trip_count  |
+--------------------+---------------------+---------------------------+
| 1                  | 10                  | 365209                    |
| 1                  | 11                  | 354838                    |
| 1                  | 12                  | 373290                    |
| 1                  | 13                  | 369786                    |
| 1                  | 14                  | 400676                    |
| 1                  | 15                  | 399090                    |
| 1                  | 16                  | 354966                    |
| 1                  | 17                  | 429289                    |
| 1                  | 18                  | 516706                    |
| 1                  | 19                  | 501097                    |
+--------------------+---------------------+---------------------------+
10 rows selected (0.379 seconds)
```

Step 2: After creation and insertion of data in a table we need to download file in your local system to visualize the table in tableau.

Note: Step 2 to download file in local system is same for all analysis so we need to Follow step 2 from page 12 (Part a, b, c).

Note: Do not forget to change current table name drop_analysis.

Step 3: Once the file downloaded successfully you, have to upload the csv file in tableau. You can follow the below steps to upload file to tableau.

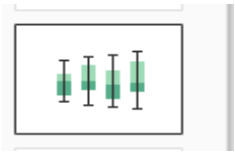
Note: Step 3 to download upload file to tableau is same for all analysis. So we need to

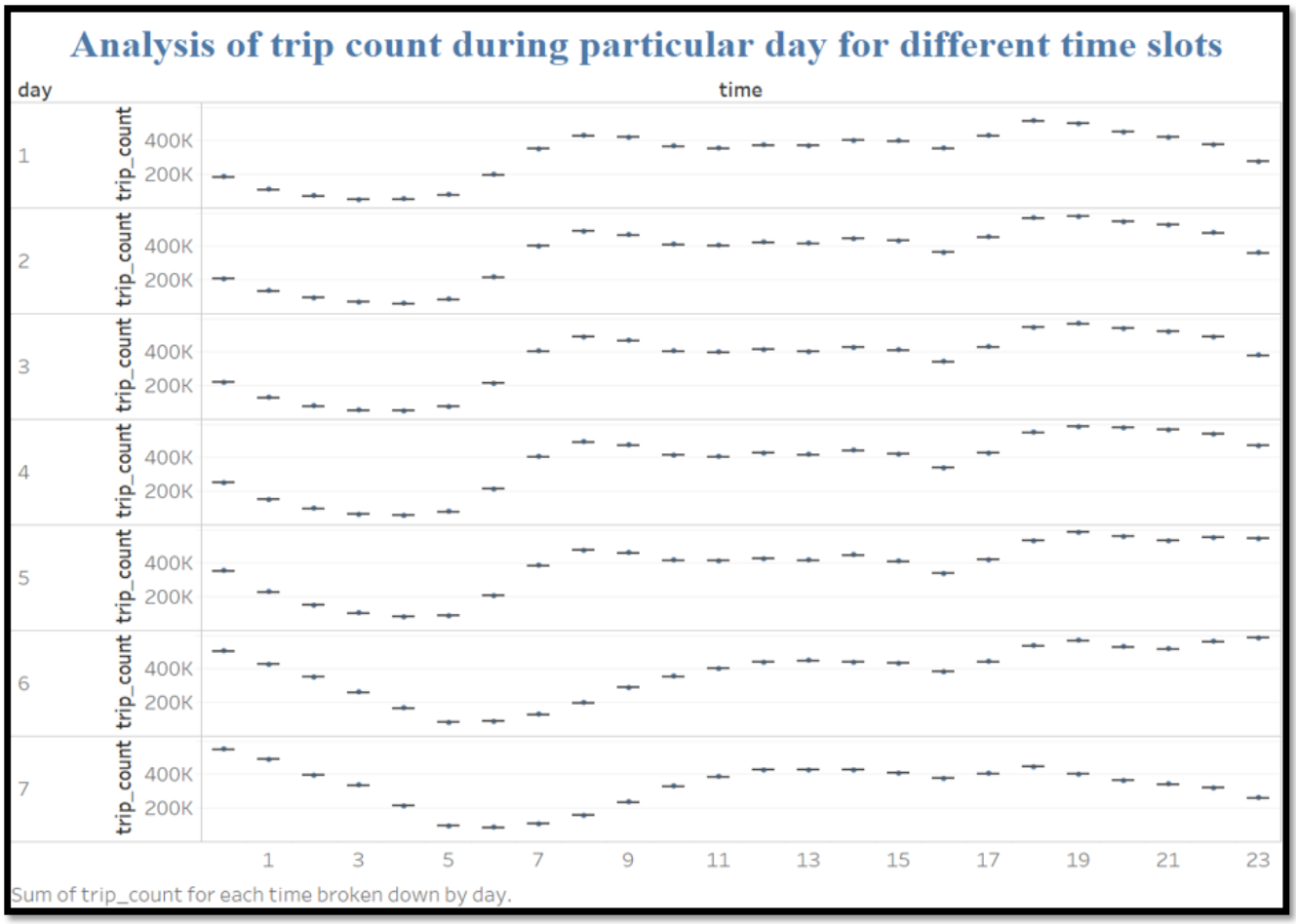
Follow step 3 from analysis 1 page 13.

Note: Do not forget to change current file name drop_analysis.csv.

Step 4: **Analysis for drop_analysis.csv file.**

- In this analysis need to drag time and day from measures to dimension.
- After that need to drag time to the column and then day and trip_count to row part.
- And need to select box-and-whisker-plots like following.





- In the above graph, the column represents time (24-hour format) and row represents trip count.

- The following graph depicts the trip count during each day of the week for different time slot.
- Through this graph, we can say that peak hours are higher during the night for all the days whereas in the morning the trip count is not that high as compared to the night.
- During Saturday's and Sunday's the trip count is the highest during 12:00AM – 2.00 AM compared to other days in the week.
- From the graph, we can conclude that in weekdays trip count gradually start increasing from 5 AM, reaches peak at 9AM, there is slight drop from 10 AM to 5PM (17:00), and again it reaches peak point at 6PM (18:00) to 8 PM (20:00).

Table 7)
Analysis 4a:
Step 1: We are creating the table to find out when people prefer short_distance trip During particular hour.

Following hive query creates **short_distance** table.

Note: Do not forget to change the username instead of dkansar.

```
CREATE EXTERNAL TABLE IF NOT EXISTS short_distance (
day INT, hour INT, trip_count BIGINT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/dkansar/short_distance/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

Following hive query Overwrite data into short_distance table from dropoff_day table.

```
INSERT OVERWRITE TABLE short_distance SELECT day,hour,count(total_distance) from
dropoff_day where total_distance<40 group by day, hour order by day ,hour ASC;
```

We can check table created successfully or not.

```
0: jdbc:hive2://localhost:10000/default> show tables;
```

```
+--------------------+
|      tab_name      |
+--------------------+
| short_distance     |
| sunday             |
| taxi               |
+--------------------+
68 rows selected (0.352 seconds)
```

We can check the content of the table.

0: jdbc:hive2://localhost:10000/default> select * from short_distance limit 10;

```
0: jdbc:hive2://localhost:10000/default> select * from short_distance limit 10;
+---------------------+----------------------+----------------------------+
| short_distance.day  | short_distance.hour  | short_distance.trip_count  |
+---------------------+----------------------+----------------------------+
| 1                   | 10                   | 1514                       |
| 1                   | 11                   | 1584                       |
| 1                   | 12                   | 1634                       |
| 1                   | 13                   | 1761                       |
| 1                   | 14                   | 2008                       |
| 1                   | 15                   | 2154                       |
| 1                   | 16                   | 2217                       |
| 1                   | 17                   | 2225                       |
| 1                   | 18                   | 2016                       |
| 1                   | 19                   | 1937                       |
+---------------------+----------------------+----------------------------+
10 rows selected (0.337 seconds)
```

Step 2: After creation and insertion of data in a table we need to download file in your local system to visualize the table in tableau.

Note: Step 2 to download file in local system is same for all analysis so we need to Follow step 2 from page 12 (Part a, b, c).

Note: Do not forget to change current table name short_distance.

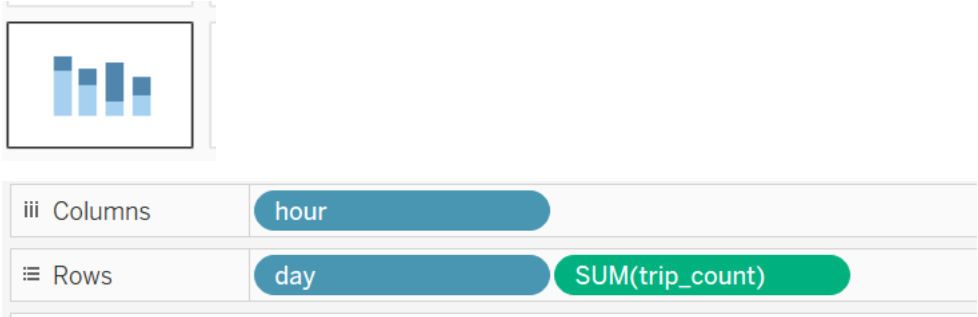
Step 3: Once the file downloaded successfully you, have to upload the csv file in tableau. You can follow the below steps to upload file to tableau.

Note: Step 3 to download upload file to tableau is same for all analysis. So we need to Follow step 3 from analysis 1 page 13.

Note: Do not forget to change current file name short_distance.csv.

Step 4: **Analysis for short_distance.csv file.**

- In this analysis need to drag time and day from measures to dimension.
- After that need to drag time to the column and then day and trip_count to row part.
- And need to select **bar** like following.



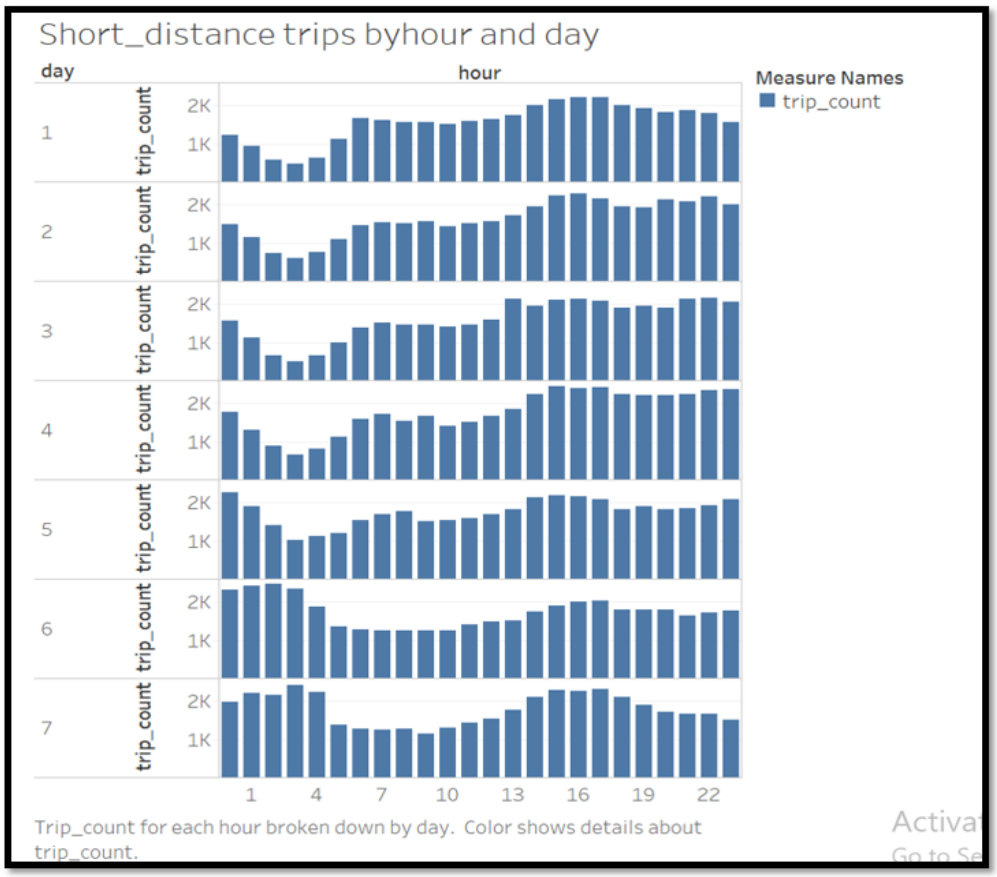


Table 8)

Analysis 4b)

Step 1: We are creating the table "long-distance" which find out when people like to Take more long_distance trip during particular hour.

Following hive query creates **long_distance** table.

Note: Do not forget to change the username instead of dkansar.

```
CREATE EXTERNAL TABLE IF NOT EXISTS long_distance (
day INT, hour INT, trip_count BIGINT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/dkansar/long_distance/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

Following hive query Overwrite data into long_distance table from dropoff_day table.

INSERT OVERWRITE TABLE long_distance SELECT day,hour,count(total_distance) from dropoff_day where total_distance<40 group by day, hour order by day ,hour ASC;

We can check table created successfully or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| long_distance      |
| manhattan          |
| midnight           |
+--------------------+
68 rows selected (0.352 seconds)
```

0: jdbc:hive2://localhost:10000/default> select * from long_distance limit 10;

```
0: jdbc:hive2://localhost:10000/default>  select * from long_distance limit 10;
+--------------------+---------------------+--------------------------+
| long_distance.day  | long_distance.hour  | long_distance.trip_count |
+--------------------+---------------------+--------------------------+
| 1                  | 10                  | 11172                    |
| 1                  | 11                  | 10660                    |
| 1                  | 12                  | 11974                    |
| 1                  | 13                  | 12178                    |
| 1                  | 14                  | 12399                    |
| 1                  | 15                  | 12946                    |
| 1                  | 16                  | 13651                    |
| 1                  | 17                  | 14263                    |
| 1                  | 18                  | 15516                    |
| 1                  | 19                  | 16931                    |
+--------------------+---------------------+--------------------------+
10 rows selected (0.361 seconds)
```

Step 2: After creation and insertion of data in a table we need to download file in your local system to visualize the table in tableau.

Note: Step 2 to download file in local system is same for all analysis so we need to Follow step 2 from page 12 (Part a, b, c).

Note: Do not forget to change current table name long_distance.

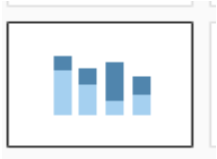
Step 3: Once the file downloaded successfully you, have to upload the csv file in tableau. You can follow the below steps to upload file to tableau.

Note: Step 3 to download upload file to tableau is same for all analysis. So we need to Follow step 3 from analysis 1 page 13.

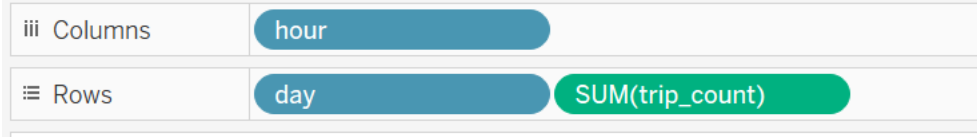
Note: Do not forget to change current file name long_distance.csv.

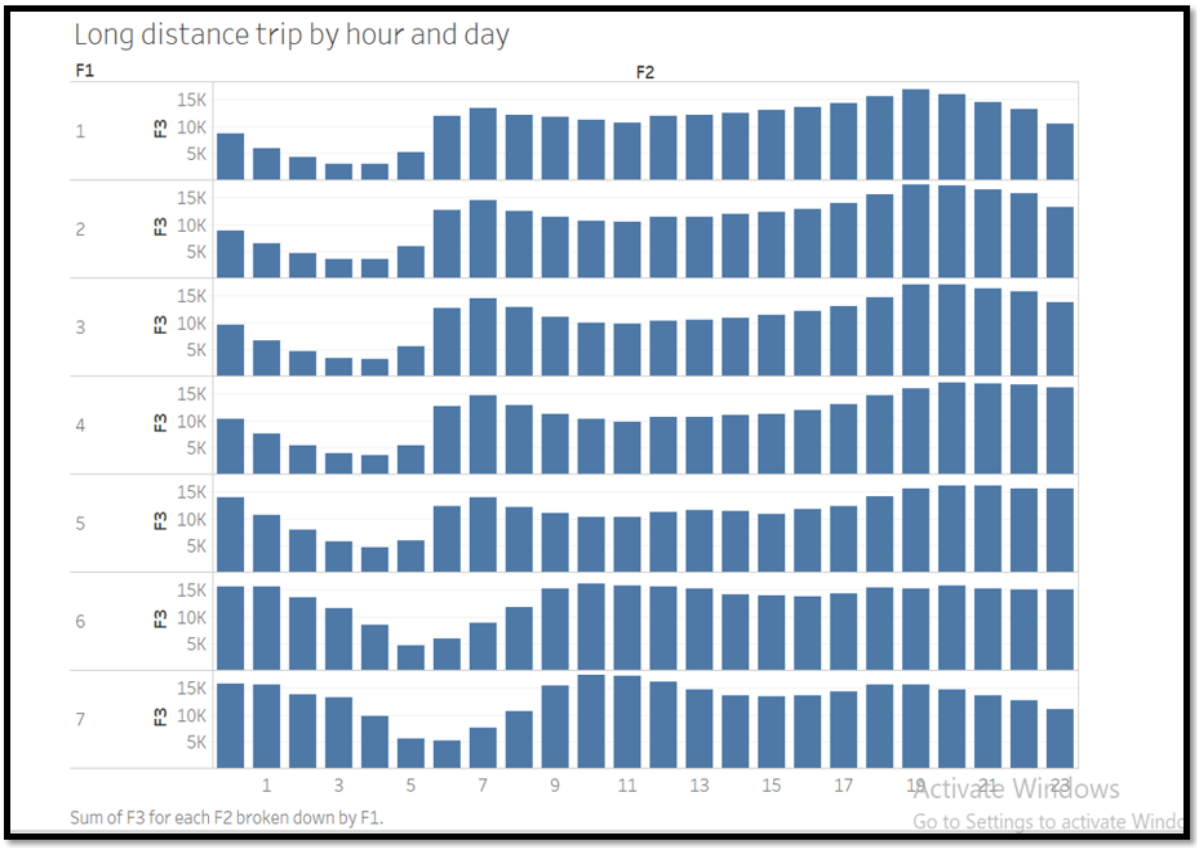
Step 4: **Analysis for long_distance.csv file.**

- In this analysis need to drag time and day from measures to dimension.
- After that need to drag time to the column and then day and trip_count to row part.
- And need to select bar like following.



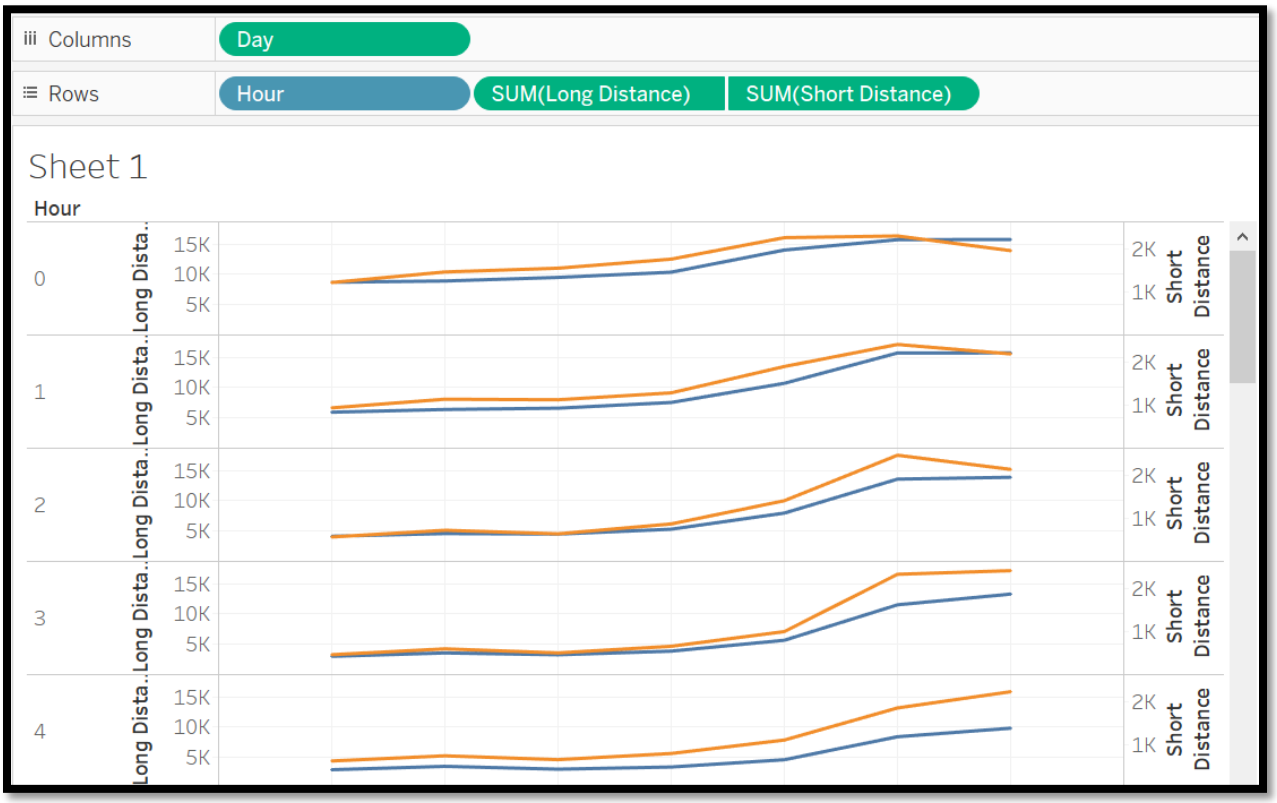
And you can see the following chart.

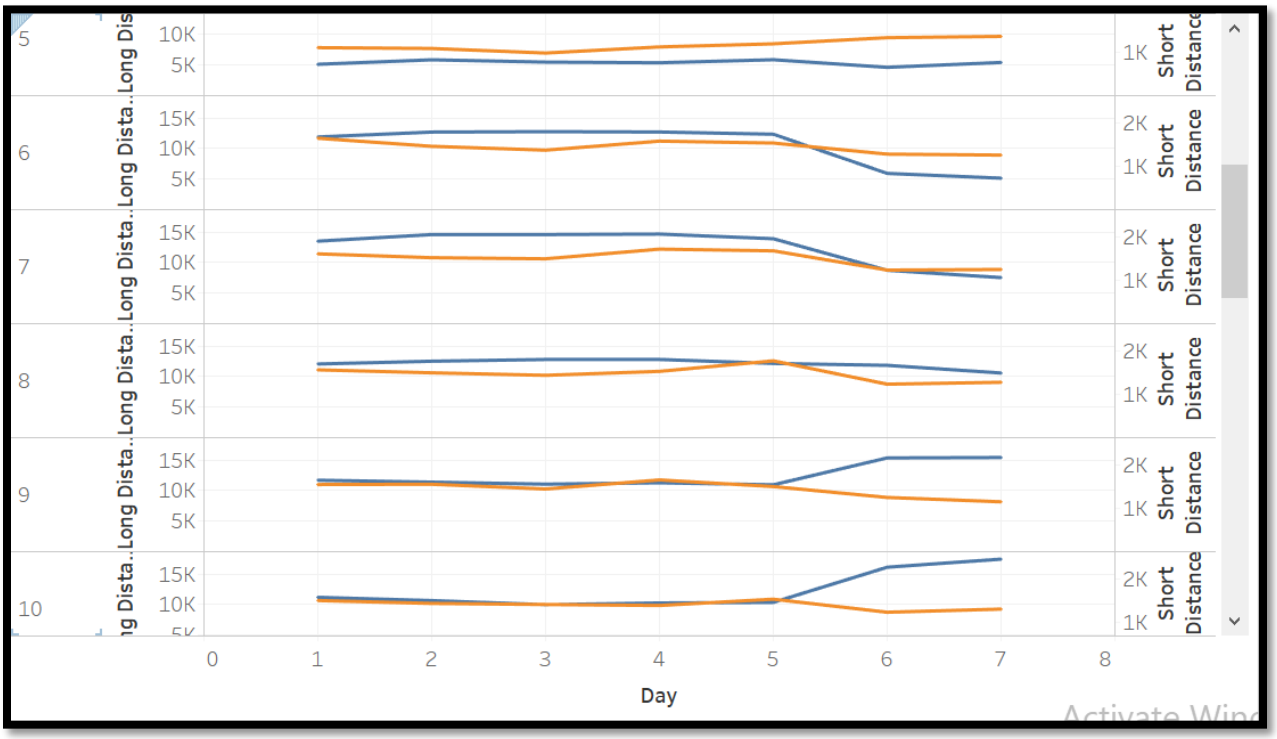


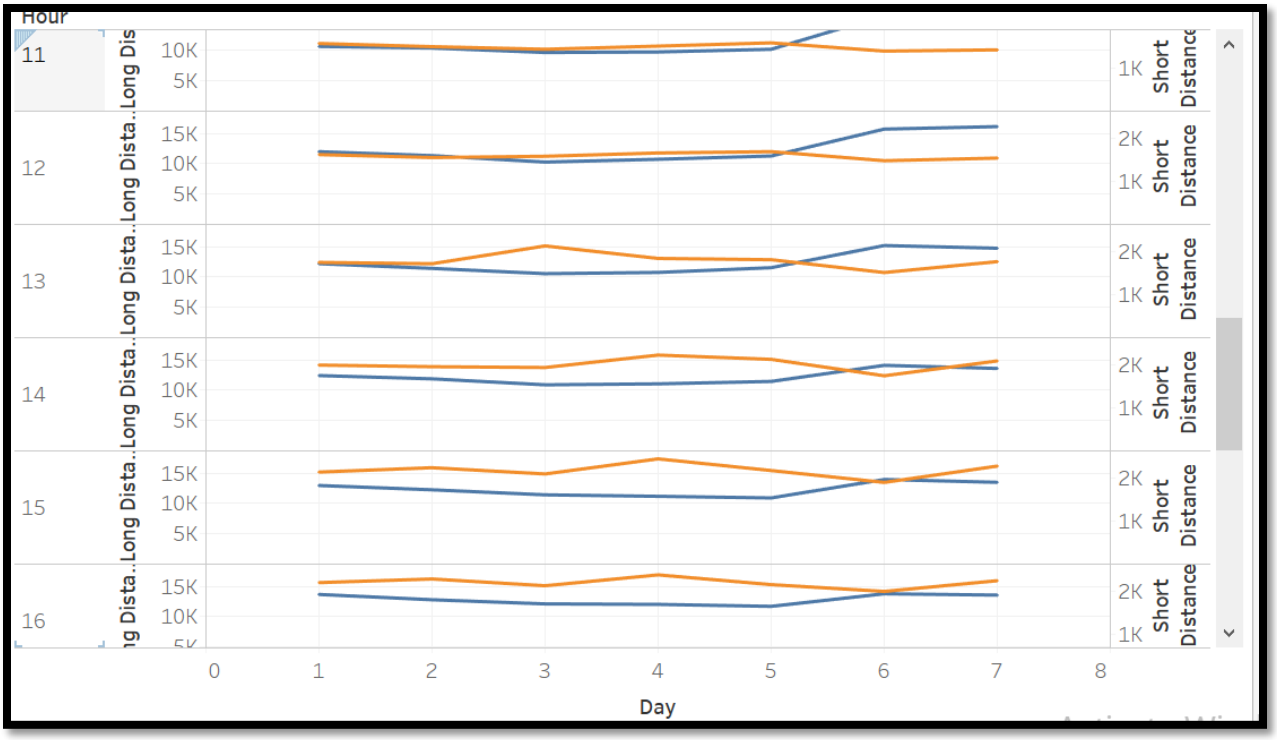


- In the above graph row represents the trip count and column represents the hours (24 Hour Format).
- By comparing both the short and long distance it has been observed that people prefer short distance trip in the morning whereas long distance trip is mostly preferred during evening hours.
- During weekends long distance trip are higher than short distance trip in mornings.

Comparison chart for Short_distance and long_distance







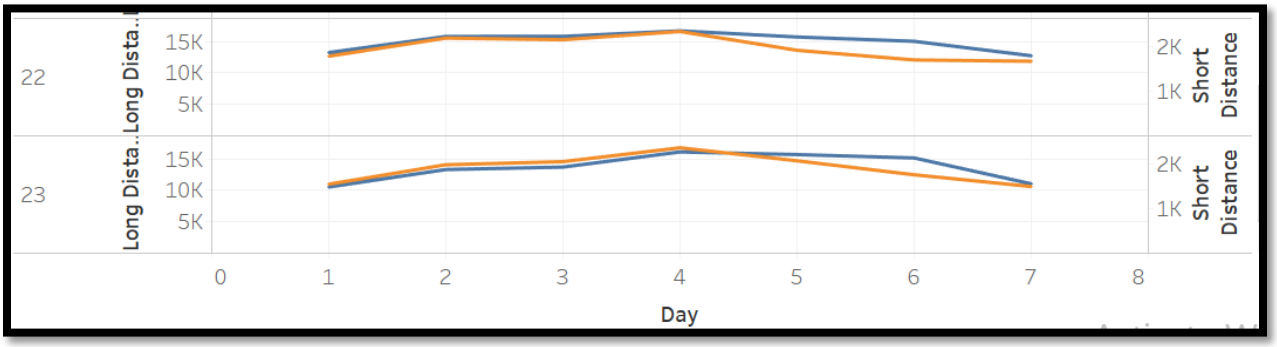


Table 9)
Analysis 5)

Step 1: We are creating the table "payment" find out how many people like to pay by Cash and card two different payment method.

Following hive query creates **payment** table.

Note: Do not forget to change the username instead of dkansar.

```
CREATE EXTERNAL TABLE IF NOT EXISTS payment (
method STRING, method_count BIGINT, Tip_count BIGINT
```

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

STORED AS TEXTFILE LOCATION '/user/dkansar/payment/'

TBLPROPERTIES ('skip.header.line.count'='2');

Following hive query Overwrite data into payment table from fair table.

INSERT OVERWRITE TABLE payment SELECT payment_type,count(payment_type),sum(tip_amount) from fair GROUP BY payment_type ORDER BY payment_type Asc;

We can check table created successfully or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| nighttime          |
| occupation         |
| payment            |
| products           |
+--------------------+
61 rows selected (0.372 seconds)
```

Step 2: After creation and insertion of data in a table we need to download file in your local system to visualize the table in tableau.

Note: Step 2 to download file in local system is same for all analysis so we need to Follow step 2 from page 12 (Part a, b, c). Do not forget to change current table name

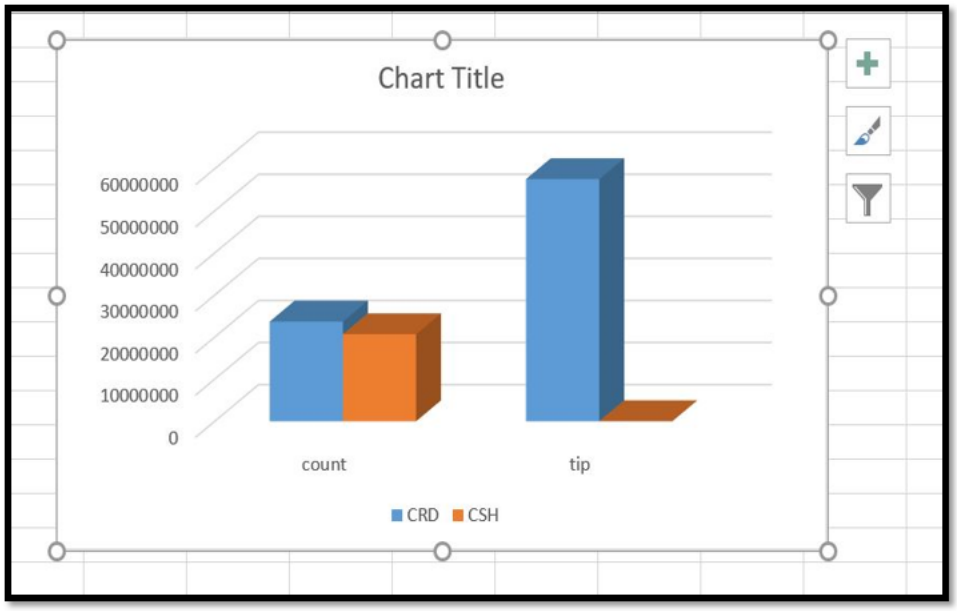
Dates.

Note: Do not forget to change current table name payment.

Step 3: Once the file downloaded successfully you, have to upload the csv file in excel. You can follow the below steps to upload file to excel.

Step 4: **Analysis for payment.csv file.**

- **In excel from insert menu from chart menu select 3D chart.**



- This bar graph represents the payment method that is used by people while paying their fair amount. Most convenient way of paying the fares is either through cash or through card.
- The blue color symbolizes the payment done through card and the orange color symbolizes the payment done through cash.
- By analyzing the graph we can conclude that there is very little difference between the fares paid by cash and card whereas people prefer paying tip through card rather than cash.

Table 10)
Analysis 6)
Step 1: We are creating the table "manhattan" find out drop off location in location central
Park during four month on particular day.

Following hive query creates **Manhattan** table.

Note: Do not forget to change the username instead of dkansar.

CREATE EXTERNAL TABLE IF NOT EXISTS manhattan (

trip_count BIGINT, dropoff_latitude FLOAT, dropoff_longitude FLOAT, dropoff_date TIMESTAMP)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ','

STORED AS TEXTFILE LOCATION '/user/dkansar/manhattan/'

TBLPROPERTIES ('skip.header.line.count'='2');

Following hive query Overwrite data into manhattan table from taxi table.

INSERT OVERWRITE TABLE manhattan select count(*),dropoff_latitude , dropoff_longitude,dropoff_date from taxi where dropoff_latitude Like '-73.971%' AND dropoff_longitude Like'40.783%' GROUP By dropoff_date,dropoff_latitude ,dropoff_longitude;

We can check table created successfully or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| long_distance      |
| manhattan          |
| midnight           |
+--------------------+
68 rows selected (0.352 seconds)
```

0: jdbc:hive2://localhost:10000/default> select * from manhattan limit 10;

```
0: jdbc:hive2://localhost:10000/default> select * from manhattan limit 10;
+----------------------+----------------------------+-----------------------------+-------------------------+
| manhattan.trip_count | manhattan.dropoff_latitude | manhattan.dropoff_longitude | manhattan.dropoff_date  |
+----------------------+----------------------------+-----------------------------+-------------------------+
| 1                    | -73.97186                  | 40.783813                   | 2013-01-01 12:47:33.0   |
| 1                    | -73.9715                   | 40.78342                    | 2013-01-01 13:21:00.0   |
| 1                    | -73.971306                 | 40.783203                   | 2013-01-01 17:24:25.0   |
| 1                    | -73.97197                  | 40.783142                   | 2013-01-01 17:57:00.0   |
| 1                    | -73.971146                 | 40.783367                   | 2013-01-02 06:47:28.0   |
| 1                    | -73.971214                 | 40.78398                    | 2013-01-02 13:07:24.0   |
| 1                    | -73.97118                  | 40.783466                   | 2013-01-02 13:50:01.0   |
| 1                    | -73.971504                 | 40.783405                   | 2013-01-02 15:37:00.0   |
| 1                    | -73.97122                  | 40.783237                   | 2013-01-02 15:47:59.0   |
| 1                    | -73.9711                   | 40.78363                    | 2013-01-03 07:18:24.0   |
+----------------------+----------------------------+-----------------------------+-------------------------+
10 rows selected (0.397 seconds)
```

Step 2: After creation and insertion of data in a table we need to download file in your local system to visualize the table in tableau.

Note: Step 2 to download file in local system is same for all analysis so we need to

Follow step 2 from page 12 (Part a, b, c).

Note: Do not forget to change current table name manhattan.

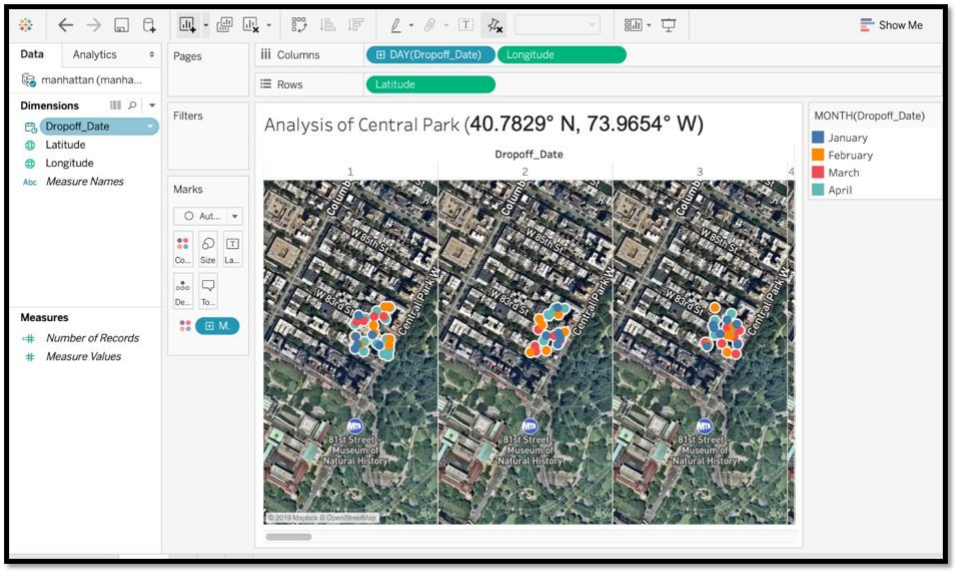
Step 3: Once the file downloaded successfully you, have to upload the csv file in tableau. You can follow the below steps to upload file to tableau.

Note: Step 3 to download upload file to tableau is same for all analysis. So we need to Follow step 3 from analysis 1 page 13.

Note: Do not forget to change current file name manhattan.csv.

Step 4: **Analysis for manhattan.csv file.**

- You need to drag longitude and latitude to the dimension part and if you hover to longitude can see little reverse tringle click on this.
- From the drop down menu select geolocation role then select longitude, and do same procedure for latitude too.
- Now drag longitude and dropoff_date to column field and latitude to raw field.
- Then in column field dropoff_date menu select **day** field.
- Next step is from dimensions drag dropoff_date to color in marks field.



- We have chosen latitudes and longitudes as 40.7829 N , 73.9654 W and Day(Dropoff_date) to visualize the data. Each different color dot represents different months**.**
- The above visualization shows the drop off taxi records for a particular area i.e Central Park for each day for 4 months. We did analysis and visualize every record through tableau maps.

Table 11)
Analysis 7)

Step 1: We are creating the table "short" table to find out dropoff location in New York City for short trip.(trip<40)

Following hive query creates **short** table.

Note: Do not forget to change the username instead of dkansar.

```
CREATE EXTERNAL TABLE IF NOT EXISTS short (
trip_distance BIGINT, dropoff_latitude FLOAT, dropoff_longitude FLOAT,)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/dkansar/short/'
TBLPROPERTIES ('skip.header.line.count'='2');
```

Following hive query Overwrite data into short table from taxi table.

```
INSERT OVERWRITE TABLE short SELECT trip_distance,dropoff_latitude,dropoff_longitude
from taxi where trip_distance<40 limit 30000;
```

We can check table created successfully or not.

0: jdbc:hive2://localhost:10000/default> show tables;

```
+--------------------+
|      tab_name      |
+--------------------+
| secondmonth        |
| secondmontha       |
| short              |
+--------------------+
64 rows selected (0.365 seconds)
```

Step 2: After creation and insertion of data in a table we need to download file in your local system to visualize the table in tableau.

Note: Step 2 to download file in local system is same for all analysis so we need to Follow step 2 from page 12 (Part a, b, c).

Note: Do not forget to change current table name short.

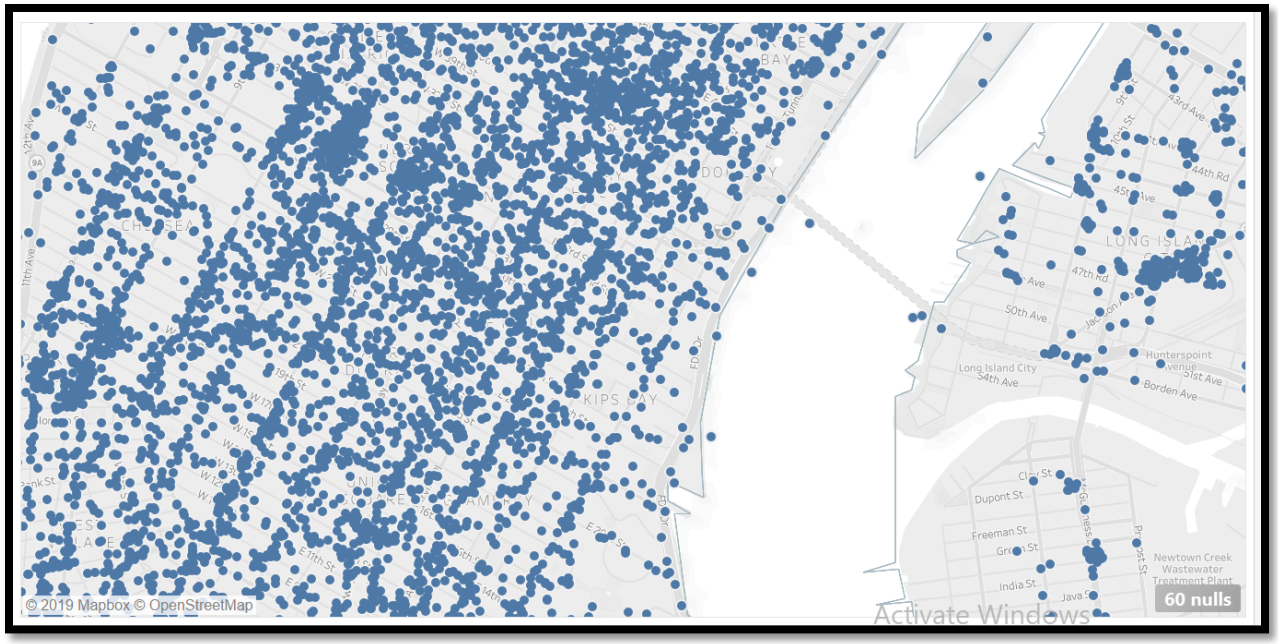
Step 3: Once the file downloaded successfully you, have to upload the csv file in tableau. You can follow the below steps to upload file to tableau.

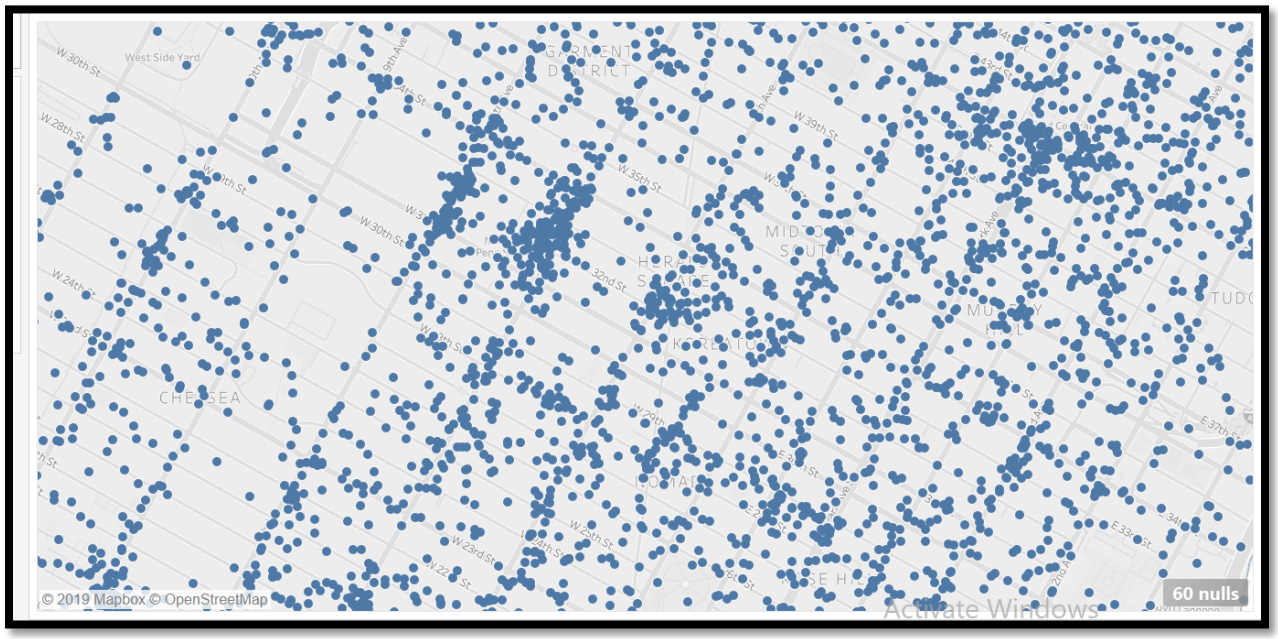
Note: Step 3 to download upload file to tableau is same for all analysis. So we need to Follow step 3 from analysis 1 page 13.

Note: Do not forget to change current file name short.csv.

Step 4: **Analysis for short.csv file.**

- You need to drag longitude and latitude to the dimension part and if you hover to longitude can see little reverse tringle click on this.
- From the drop down menu select geolocation role then select longitude, and do same procedure for latitude too.
- Now drag longitude and dropoff_date to column field and latitude to raw field.


© 2019 Mapbox © OpenStreetMap
60 nulls
Activate Windows
LONG ISLAND
Hunterspoint Avenue
Long Island City
Newtown Creek Wastewater Treatment Plant


© 2019 Mapbox © OpenStreetMap
60 nulls
Activate Windows
GARMENT DISTRICT
CHELSEA
West Side Yard

Summary

In this tutorial we have learned that how taxi trip count increases and decreases in a week and varies at particular time of a day. This analysis can be used for making business decisions which can be further used as a key element for marketing. We had downloaded the dataset and uploaded to the HDFS , later the data was manipulated and analyzed in HDFS using Hadoop and visualization of the results was done in excel and tableau maps.

References

1. URL of data source: https://chriswhong.com/open-data/foil_nyc_taxi/

2. URL of GitHub: https://github.com/anjalibaldawa/NYC-Taxi-Data

3. URL of references:

A) https://www.tableau.com/academic/students

B) https://www.kdnuggets.com/2017/02/data-science-nyc-taxi-trips.html

C) https://toddwschneider.com/posts/analyzing-1-1-billion-nyc-taxi-and-uber-trips-with-a-vengeance/