INTRODUCTION

The New York City Taxi has released a staggeringly detailed historical dataset covering over 1.1 billion individual taxi trips in the city from January 2009 through June 2015.As the Data is huge, I have taken data from Jan 2016 to June 2016 for my analysis. Taken as a part of data How much are the earnings by each Taxi Companies, what are the Peak trip Times, What is the maximum distance covered by a driver?

DATASET LINK:

http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

MAPREDUCE ANALYSIS

Analysis 1:

This analysis calculates the peak Trip Time every day for the month of January 2016. Peak Trip Time is calculated based on the number of rides passenger used the Green Taxi.

PTTWritableKey:

(Hour, Time)

Mapper:

Emits (Date, PTTWritablekey)

Reducer:

Emits (Date, PTTWritablekey)
Sorting based on the no of rides done in reducer
Used SortedHashMap

Output:

2016-01-01	Peak Hour:01-Total Rides6858
2016-01-02	Peak Hour:22-Total Rides3238
2016-01-03	Peak Hour:00-Total Rides3257
2016-01-04	Peak Hour:18-Total Rides3498
2016-01-05	Peak Hour:18-Total Rides3289
2016-01-06	Peak Hour:18-Total Rides3154
2016-01-07	Peak Hour:18-Total Rides3094
2016-01-08	Peak Hour:18-Total Rides3621
2016-01-09	Peak Hour:23-Total Rides3537
2016-01-10	Peak Hour:00-Total Rides3899
2016-01-11	Peak Hour:18-Total Rides3077
2016-01-12	Peak Hour:18-Total Rides2897
2016-01-13	Peak Hour:19-Total Rides3433
2016-01-14	Peak Hour:18-Total Rides3257
2016-01-15	Peak Hour:18-Total Rides3746
2016-01-16	Peak Hour:23-Total Rides3918
2016-01-17	Peak Hour:00-Total Rides4047
2016-01-18	Peak Hour:18-Total Rides2786
2016-01-19	Peak Hour:18-Total Rides3289
2016-01-20	Peak Hour:18-Total Rides3120
2016-01-21	Peak Hour:19-Total Rides3618
2016-01-22	Peak Hour:19-Total Rides4426
2016-01-23	Peak Hour:00-Total Rides2987
2016-01-24	Peak Hour:18-Total Rides2125
2016-01-25	Peak Hour:18-Total Rides3280
2016-01-26	Peak Hour:18-Total Rides3427
2016-01-27	Peak Hour:18-Total Rides3510
2016-01-28	Peak Hour:18-Total Rides3562
2016-01-29	Peak Hour:19-Total Rides4368
2016-01-30	Peak Hour:23-Total Rides4507

Analysis 2:

This analysis compares the No of rides each day for Yellow Taxi and Green Taxi in the month of Jan 2016.

YellowRecordCount:

Mapper:
Emits(Date,1)
Reducer:
Emits (Date, Count)

GreenRecordCount:

Mapper:
Emits(Date,1)
Reducer:
Emits (Date, Count)

The output of the above two programs were used to perform inner join on the yellow and green taxi for the month of Jan 2016.

YellowMapper:
Emits (Date, Y-Count)
GreenMapper:
Emits (Date, G-Count)
Reducer:
Emits (Date, Y-Count: G-Count)

Output:

2016-01-01	G64396,Y345037
2016-01-02	G46958,Y312831
2016-01-03	G44270,Y302878
2016-01-04	G43740,Y316171
2016-01-05	G42342,Y343251
2016-01-06	G42025,Y348516
2016-01-07	G42811,Y364894
2016-01-08	G50987,Y392070
2016-01-09	G56861,Y405825
2016-01-10	G49877,Y351788
2016-01-11	G41603,Y342651
2016-01-12	G41124,Y367390
2016-01-13	G45839,Y395090
2016-01-14	G45846,Y396473
2016-01-15	G52719,Y401289
2016-01-16	G59626,Y411899
2016-01-17	G52803,Y379156
2016-01-18	G38913,Y341481
2016-01-19	G43963,Y385187
2016-01-20	G43734,Y382105
2016-01-21	G48606,Y399654
2016-01-22	G57552,Y420162
2016-01-23	G11031,Y78133
2016-01-24	G22010,Y159766
2016-01-25	G42944,Y282087
2016-01-26	G44321,Y327655
2016-01-27	G45792,Y359180
2016-01-28	G47519,Y383326
2016-01-29	G57370,Y414039
2016-01-30	G65415,Y435369

Analysis 3:

This analysis sort the records based on distance travelled and then by fare amount earned.

DistanceFare:

(distance, fare)

DistanceFareRecord:

(Date, distance, fare)

Key Comparator:

(Distance)

Group Comparator:

(Distance, Fare)

Mapper:

Emits (DistanceFare, DistanceFareRecord)

Reducer:

Emits (DistanceFareRecord, Null)

Output:

```
2016-01-26,0.0,1.1
2016-01-25,0.0,1.1
2016-01-25,0.0,1.1
2016-01-18,0.0,1.2
2016-01-20,0.0,1.1
2016-01-27,0.0,1.1
2016-01-22,0.0,1.1
2016-01-20,0.0,1.0
2016-01-26,0.0,1.1
2016-01-22,0.0,1.2
2016-01-21,0.0,1.3
2016-01-30,0.0,1.0
2016-01-22,0.0,1.0
2016-01-30,0.0,1.0
2016-01-25,0.0,1.1
2016-01-20,0.0,1.9
2016-01-22,0.0,1.2
2016-01-20,0.0,1.0
2016-01-28,0.0,1.7
2016-01-19,0.0,1.1
2016-01-21,0.0,1.0
2016-01-29,0.0,1.0
```

Analysis 4:

This analysis partitions the record based on the binning patterns and also adds two more columns to the output based on latitude and longitudes - Pickup Location and Drop Location. This output will be fed as an input to the next two analysis

Mapper:

Emit (record, bins) along with the below conditions added to the records.

```
if((pickup_long > -74.016309 && pickup_long<-73.943986
        pickup="Manhattan";
}else if((pickup_long>-73.996224 && pickup_long< -73.9
    pickup="Brooklyn";
}else if((pickup_long>-73.929015 && pickup_long< -73.7
   pickup="Queen";
}else if((pickup_long>-73.909672 && pickup_long< -73.8
    pickup="Bronx";
}else if((pickup_long>-74.241142 && pickup_long< -74.0
    pickup="Staten Island";
}else if((pickup_long>-73.815939 && pickup_long< -73.7
    pickup="JFK Airport";
}else{
    pickup="UnKnown";
}

if((drop_long > -74.016309 && drop_long<-73.943986) &&
        drop="Manhattan";
}else if((drop_long>-73.996224 && drop_long< -73.90435
    drop="Brooklyn";
}else if((drop_long>-73.929015 && drop_long< -73.73246
   drop="Queen";
}else if((drop_long>-73.909672 && drop_long< -73.81443
    drop="Bronx";
}else if((drop_long>-74.241142 && drop_long< -74.09344
    drop="Staten Island";
}else if((drop_long>-73.815939 && drop_long< -73.76469
    drop="JFK Airport";
}else{
    drop="UnKnown";
}
```

Output:

Bins are separated based on weekday. As the input split size is more two records are generated for each bins

1	128 MB	Friday-m-00000
1	128 MB	Friday-m-00001
1	128 MB	Monday-m-00000
1	128 MB	Monday-m-00001
1	128 MB	Saturday-m-00000
1	128 MB	Saturday-m-00001
1	128 MB	Sunday-m-00000
1	128 MB	Sunday-m-00001
1	128 MB	Thursday-m-00000

Analysis 5:

This analysis performs bloom filter on the analysis 4 output permitting the below pattern of output to the next MapReduce job. Filters the bloom filter specified pickup drop combination. The output of the bloom filter is fed into another MapReduce function to calculate average tip of the driver per hour between Brooklyn and Manhattan.

Bloom Filter:

Filter (Pickup – Brooklyn, Drop – Manhattan)

Bloom Filter Mapper (Mapper 1):

Emits (Above Filter Records)

AverageTuple:

(hour, tip)

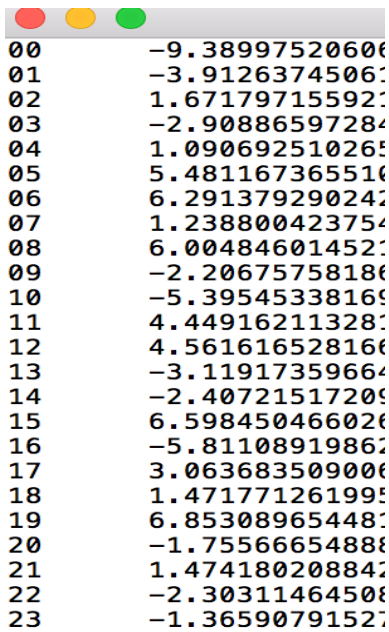
Mapper:

Emits (date, AverageTuple)

Reducer:

Emits (date, (Tip) Double Writable)

Output:



00	-9.3899752060
01	-3.9126374506
02	1.67179715592
03	-2.9088659728
04	1.09069251026
05	5.481167365510
06	6.291379290242
07	1.23880042375
08	6.00484601452
09	-2.2067575818
10	-5.3954533816
11	4.44916211328
12	4.561616528166
13	-3.1191735966
14	-2.4072151720
15	6.598450466026
16	-5.8110891986
17	3.063683509006
18	1.47177126199
19	6.85308965448
20	-1.7556665488
21	1.474180208842
22	-2.3031146450
23	-1.3659079152

Analysis 6:

This analysis uses the output analysis 4 output to calculate the no of rides between any two places in NYC.

Mapper:

Emits (date, AverageTuple)

Reducer:

Emits (date, (Tip) Double Writable)

Output:

```
Bronx-Bronx     29119
Bronx-Brooklyn  133
Bronx-JFK Airport       93
Bronx-Manhattan 2362
Bronx-Queen     443
Bronx-Staten Island     3
Brooklyn-Bronx  264
Brooklyn-Brooklyn       239420
Brooklyn-JFK Airport    2758
Brooklyn-Manhattan      58369
Brooklyn-Queen  12440
Brooklyn-Staten Island  54
JFK Airport-Bronx       3
JFK Airport-Brooklyn    13
JFK Airport-JFK Airport 178
JFK Airport-Manhattan   11
JFK Airport-Queen       70
Manhattan-Bronx 5147
Manhattan-Brooklyn      33774
Manhattan-JFK Airport   2312
Manhattan-Manhattan     238692
Manhattan-Queen 19759
Manhattan-Staten Island 21
Queen-Bronx     996
Queen-Brooklyn  7371
Queen-JFK Airport       3287
Queen-Manhattan 27141
Queen-Queen     292897
Queen-Staten Island     10
Staten Island-Brooklyn  12
Staten Island-JFK Airport       2
Staten Island-Queen     1
Staten Island-Staten Island     90
```

Analysis 7:

This analysis calculates the largest trip of a driver during the month of Jan 2016.

Mapper:

Emits (Driver ID, Distance)

Reducer:

Emits (Driver ID, Distance)

Max is calculated in reducer

Output:

```
G001    100.39
G002    101.3
G003    45.1
G004    40.86
G005    48.31
G006    133.59
G007    76.8
G008    147.5
G009    61.71
G010    79.54
G011    59.26
```

HIVE ANALYSIS

Analysis of Green Taxi Data Using Hive.

CREATE TABLE GREEN TAXI:

```
CREATE TABLE greentaxi
 (
   vendorid        int,
   pick_up_date    string,
   drop_date       string,
   flag            CHAR(1),
   rate_code       INT,
   pick_up_long    string,
   pick_up_lat     string,
   drop_off_long   string,
   drop_off_lat    string,
   passenger_count INT,
   trip_distance   DECIMAL(5,2),
   fare_amount     DECIMAL(5,2),
   extra           DECIMAL(5,2),
   tax             DECIMAL(5,2),
   tip             DECIMAL(5,2),
   tolls           DECIMAL(5,2),
   surcharge       DECIMAL(5,2),
   total_amount    DECIMAL(5,2),
   payment_type    int,
   trip_type       int
```

```
)
comment 'Data about Green NYC Taxi for the year 2016-Jan'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

LOAD DATA INTO TABLE:

```
LOAD DATA LOCAL INPATH '/Users/nirmal/Desktop/hive_greenTaxi.csv' overwrite into TABLE
greentaxi;
```

bin — java -Xmx256m -Djava.net.preferIPv4Stack=true -Dhadoop.log.dir=/usr/local/...

```
Time taken: 0.027 seconds, Fetched: 3 row(s)
hive> CREATE TABLE greentaxi
    >   (
    >      vendorid        int,
    >      pick_up_date    string,
    >      drop_date       string,
    >      flag            CHAR(1),
    >      rate_code       INT,
    >      pick_up_long    string,
    >      pick_up_lat     string,
    >      drop_off_long   string,
    >      drop_off_lat    string,
    >      passenger_count INT,
    >      trip_distance   DECIMAL(5,2),
    >      fare_amount     DECIMAL(5,2),
    >      extra           DECIMAL(5,2),
    >      tax             DECIMAL(5,2),
    >      tip             DECIMAL(5,2),
    >      tolls           DECIMAL(5,2),
    >      surcharge       DECIMAL(5,2),
    >      total_amount    DECIMAL(5,2),
    >      payment_type    int,
    >      trip_type       int
    >   )
    > comment 'Data about Green NYC Taxi for the year 2016-Jan'
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE;
OK
Time taken: 0.035 seconds
hive> LOAD DATA LOCAL INPATH '/Users/nirmal/Desktop/hive_greenTaxi.csv' overwrite int
o TABLE greentaxi;
Loading data to table default.greentaxi
Table default.greentaxi stats: [numFiles=1, numRows=0, totalSize=123266582, rawDataSi
ze=0]
OK
Time taken: 0.424 seconds
```

ANALYSIS TO CALCULATE TOTAL AMOUNT EARNED PER DAY BY GREEN TAXI RIDES:

```
select g.pickup_date,sum(g.total_amount) from
(select TO_DATE(from_unixtime(UNIX_TIMESTAMP(pick_up_date,'mm/dd/yy hh:mm')))
as pickup_date ,total_amount from greentaxi) g group by g.pickup_date;
```

```
hive> select g.pickup_date,sum(g.total_amount) from
    > (select TO_DATE(from_unixtime(UNIX_TIMESTAMP(pick_up_date,'mm/dd/yy hh:mm')))
as pickup_date ,total_amount from greentaxi) g group by g.pickup_date;
Query ID = nirmal_20170423173929_3dd865fd-392d-4b40-9515-4962ac63a55c
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1492974919297_0006, Tracking URL = http://thiyagarajans-MacBook-P
ro.local:8088/proxy/application_1492974919297_0006/
Kill Command = /usr/local/bin/hadoop-2.5.2//bin/hadoop job  -kill job_1492974919297_
0006
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2017-04-23 17:39:34,412 Stage-1 map = 0%,  reduce = 0%
2017-04-23 17:39:43,621 Stage-1 map = 100%,  reduce = 0%
2017-04-23 17:39:47,694 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_1492974919297_0006
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   HDFS Read: 123277439 HDFS Write: 459 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
2016-01-01      979082.44
2016-01-02      656240.85
2016-01-03      633191.69
2016-01-04      603734.3
2016-01-05      593993.79
2016-01-06      594275.04
2016-01-07      609830.1
2016-01-08      732898.9
2016-01-09      821240.72
2016-01-10      719753.71
2016-01-11      585793.59
2016-01-12      589505.27
2016-01-13      648550.86
2016-01-14      663275.93
2016-01-15      784792.66
2016-01-16      868727.81
2016-01-17      758297.91
2016-01-18      529132.23
2016-01-19      616183.72
2016-01-20      621652.26
2016-01-21      694884.55
2016-01-22      711476.04
Time taken: 18.94 seconds, Fetched: 22 row(s)
```

ANALYSIS TO CALCULATE NO OF RIDES PER DAY:

select g.pickup_date,sum(g.passenger_count) from (select TO_DATE(from_unixtime(UNIX_TIMESTAMP(pick_up_date,'mm/dd/yy hh:mm'))) as pickup_date ,passenger_count from greentaxi) g group by g.pickup_date;

```
hive> select g.pickup_date,sum(g.passenger_count) from
    > (select TO_DATE(from_unixtime(UNIX_TIMESTAMP(pick_up_date,'mm/dd/yy hh:mm')))
as pickup_date ,passenger_count from greentaxi) g group by g.pickup_date;
Query ID = nirmal_20170423174139_234fbecc-6892-4cc4-a443-e07c56747f3c
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1492974919297_0007, Tracking URL = http://thiyagarajans-MacBook-P
ro.local:8088/proxy/application_1492974919297_0007/
Kill Command = /usr/local/bin/hadoop-2.5.2//bin/hadoop job  -kill job_1492974919297_
0007
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2017-04-23 17:41:45,042 Stage-1 map = 0%,  reduce = 0%
2017-04-23 17:41:54,205 Stage-1 map = 100%,  reduce = 0%
2017-04-23 17:41:59,302 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_1492974919297_0007
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   HDFS Read: 123277214 HDFS Write: 374 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
2016-01-01      91243
2016-01-02      64687
2016-01-03      60679
2016-01-04      59229
2016-01-05      56470
2016-01-06      56768
2016-01-07      57851
2016-01-08      68983
2016-01-09      78346
2016-01-10      69068
2016-01-11      55954
2016-01-12      54807
2016-01-13      62067
2016-01-14      61792
2016-01-15      70679
2016-01-16      82671
2016-01-17      73525
2016-01-18      52941
2016-01-19      59035
2016-01-20      58591
2016-01-21      65817
2016-01-22      66786
Time taken: 20.352 seconds, Fetched: 22 row(s)
```

PIG ANALYSIS:

1)Total Amount earned per pay by Green Taxi divided into Fare Amount and Trip Amount.

```
thiyagarajans-MacBook-Pro:bin nirmal$ cat amountPerDay.pig
greenTaxi = LOAD 'hdfs://localhost:9000/FinalProject/Pig/Pig_greenTaxi.csv' USING PigStora
ge(',');
dataTaxi = FOREACH greenTaxi GENERATE $1 as date,$13 as fare,$16 as tip;
grp = GROUP dataTaxi BY date;
cnt = FOREACH grp GENERATE group,SUM(dataTaxi.fare),SUM(dataTaxi.tip);
STORE cnt INTO ' hdfs://localhost:9000/FinalProject/Pig/Output1 ' USING PigStorage (',');
thiyagarajans-MacBook-Pro:bin nirmal$
```

Output: Stored in HDFS

part-r-00000 (4)

```
1/1/16,826813.6800000003,74418.79000000111
1/2/16,553494.6399999999,51496.86000000068
1/3/16,532954.1499999997,51416.49000000043
1/4/16,498916.6999999997,47915.44000000039
1/5/16,490476.51999999984,49628.80000000021
1/6/16,490811.90999999986,49469.17000000035
1/7/16,502291.1099999999,52505.669999999875
1/8/16,603486.8200000001,62940.56000000003
1/9/16,683626.2900000002,74328.29000000104
1/10/16,598108.8999999998,66474.50000000047
1/11/16,482282.19,50366.46000000017
1/12/16,485847.85999999987,51097.33999999998
1/13/16,532369.8899999997,57721.420000000646
1/14/16,546080.1699999999,58481.59000000035
1/15/16,645820.6900000001,69378.79000000065
1/16/16,723613.37,78008.63000000127
1/17/16,631714.07,68160.39000000026
1/18/16,440789.49,46979.28000000011
1/19/16,507025.3599999999,53607.310000000354
1/20/16,509135.91999999987,56470.640000000094
1/21/16,569413.9500000001,62273.44000000031
1/22/16,584622.9199999998,61606.83000000004
```

2. What is the maximum fare, maximum trip and maximum total amount earned per day.

```
[thiyagarajans-MacBook-Pro:bin nirmal$ cat maxTripRide.pig
taxi = LOAD 'hdfs://localhost:9000/FinalProject/Pig/Pig_greenTaxi.csv' USING PigStorage(',
');
data = FOREACH taxi GENERATE $1 as pickupDate,$13 as fare,$16 as trip,$19 as total;
filtered = FILTER data BY $1*$2*$3 is not null;
grped = GROUP filtered by pickupDate;
amt = FOREACH grped GENERATE group,MAX(filtered.fare),MAX(filtered.trip),MAX(filtered.tota
l);
STORE amt INTO ' hdfs://localhost:9000/FinalProject/Pig/Output2 ' USING PigStorage (',');
thiyagarajans-MacBook-Pro:bin nirmal$
```

OUTPUT:

```
1/1/16,499.0,100.0,499.0
1/2/16,490.0,110.8,492.0
1/3/16,499.0,219.8,499.0
1/4/16,499.0,112.59,499.34
1/5/16,533.5,174.25,755.09
1/6/16,936.0,85.0,971.8
1/7/16,260.0,100.0,278.04
1/8/16,499.0,400.0,499.0
1/9/16,498.77,250.0,498.88
1/10/16,483.23,400.0,483.23
1/11/16,636.5,400.0,648.3
1/12/16,650.0,145.0,700.0
1/13/16,499.0,350.0,690.6
1/14/16,497.0,103.5,497.0
1/15/16,483.93,100.0,483.93
1/16/16,497.29,195.0,497.29
1/17/16,492.89,197.0,504.96
1/18/16,462.0,100.0,462.0
1/19/16,704.5,168.51,730.92
1/20/16,510.5,116.0,532.3
1/21/16,350.0,100.0,350.0
1/22/16,507.0,101.56,609.36
```

3. Tip Percent of Every Ride

```
thiyagarajans-MacBook-Pro:bin nirmal$ cat TipPercent.pig
tipData = LOAD 'hdfs://localhost:9000/FinalProject/Pig/Pig_greenTaxi.csv' USING PigStorage
(',');
filterData = FOREACH tipData GENERATE $1 as pickupDate,$2 as pickupTime,$16 as tip,$19 as
total;
tipPercent = FOREACH filterData GENERATE $0,$1,($2*100)/($3-$2);
STORE tipPercent INTO ' hdfs://localhost:9000/FinalProject/Pig/Output3 ' USING PigStorage
(',');
thiyagarajans-MacBook-Pro:bin nirmal$
```

OUTPUT:

part-m-

```
1/1/16,0:29,10.75268817204301
1/1/16,0:19,0.0
1/1/16,0:19,22.47191011235955
1/1/16,0:22,0.0
1/1/16,0:24,0.0
1/1/16,0:32,0.0
1/1/16,0:34,0.0
1/1/16,0:31,0.0
1/1/16,0:24,15.037593984962406
1/1/16,0:28,9.70873786407767
1/1/16,0:32,13.698630136986303
1/1/16,0:37,0.0
1/1/16,0:21,0.0
1/1/16,0:34,0.0
1/1/16,0:26,16.260162601626014
1/1/16,0:35,17.24137931034483
1/1/16,0:25,0.0
1/1/16,0:17,17.857142857142858
1/1/16,0:31,0.0
1/1/16,0:25,0.0
1/1/16,0:30,0.0
1/1/16,0:21,24.5398773006135
1/1/16,0:18,0.0
1/1/16,0:10,12.605042016806722
1/1/16,0:30,29.126213592233007
1/1/16,0:24,0.0
1/1/16,0:30,17.699115044247787
1/1/16,0:32,0.0
1/1/16,0:34,0.0
1/1/16,0:19,0.0
```

Availability:

HBASE ANALYSIS:

1.Table Creation HBase:

create 'greenTaxi','date','lat','long','dist','fare'

```
hbase(main):004:0> create 'greenTaxi','date','lat','long','dist','fare'
 0 row(s) in 2.2860 seconds

 => Hbase::Table - greenTaxi
hbase(main):005:0> thiyagarajans-MacBook-Pro:bin nirmal$ pwd
```

2.Load data into Table

```
thiyagarajans-MacBook-Pro:bin nirmal$ ./hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.columns="HBASE_ROW_KEY,date:pick_d,date:drop_
d,long:pick_lo,lat:pick_la,long:drop_lo,lat:drop_la,dist:distance,fare:trip,fare:tip,fare:total" greenTaxi hdfs://localhost:9000/FinalProject/Hbase
/H_greenTaxi.txt
2017-04-25 23:34:03,574 WARN  [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes wh
ere applicable
2017-04-25 23:34:03,842 INFO  [main] zookeeper.RecoverableZooKeeper: Process identifier=hconnection-0x276438c9 connecting to ZooKeeper ensemble=loc
alhost:2181
2017-04-25 23:34:03,949 INFO  [main] zookeeper.ZooKeeper: Client environment:zookeeper.version=3.4.6-1569965, built on 02/20/2014 09:09 GMT
2017-04-25 23:34:03,950 INFO  [main] zookeeper.ZooKeeper: Client environment:host.name=10.110.128.136
2017-04-25 23:34:03,950 INFO  [main] zookeeper.ZooKeeper: Client environment:java.version=1.8.0_101
2017-04-25 23:34:03,950 INFO  [main] zookeeper.ZooKeeper: Client environment:java.vendor=Oracle Corporation
```

3.Count Operation

```
hbase(main):001:0> count 'greenTaxi'
Current count: 1000, row: 100897

Current count: 2000, row: 101797

Current count: 3000, row: 102697

Current count: 4000, row: 103597

Current count: 5000, row: 104497

Current count: 6000, row: 105397

Current count: 7000, row: 106297

Current count: 8000, row: 107197

Current count: 9000, row: 108097
```

```
Current count: 569000, row: 92621
Current count: 570000, row: 93521
Current count: 571000, row: 94421
Current count: 572000, row: 95321
Current count: 573000, row: 96221
Current count: 574000, row: 97121
Current count: 575000, row: 98021
Current count: 576000, row: 98922
Current count: 577000, row: 99822
577195 row(s) in 13.4220 seconds
=> 577195
```

Get Command:

```
hbase(main):002:0> get 'greenTaxi','171850'
COLUMN                          CELL
 date:drop_d                    timestamp=1493177643550, value=1/14/16
 date:pick_d                    timestamp=1493177643550, value=1/14/16
 dist:distance                  timestamp=1493177643550, value=1.87
 fare:tip                       timestamp=1493177643550, value=0
 fare:total                     timestamp=1493177643550, value=10.3
 fare:trip                      timestamp=1493177643550, value=8.5
 lat:drop_la                    timestamp=1493177643550, value=40.81887436
 lat:pick_la                    timestamp=1493177643550, value=40.79721069
 long:drop_lo                   timestamp=1493177643550, value=-73.93723297
 long:pick_lo                   timestamp=1493177643550, value=-73.94693756
1 row(s) in 0.2410 seconds

hbase(main):003:0> get 'greenTaxi','171850',{COLUMN=>'fare:trip'}
COLUMN                          CELL
 fare:trip                      timestamp=1493177643550, value=8.5
1 row(s) in 0.0180 seconds

hbase(main):004:0>
```

MAPREDUCECODE:

Analysis 1:

PTTWritable.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package PeakTripTime;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.Writable;
import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.io.WritableUtils;

/**
 *
 * @author nirmal
 */
public class PTTWritable implements Writable{

    private String time;
    private int count;

    public PTTWritable(){
        time=new String();
        count=0;
    }

    public PTTWritable(String time, int count) {
        this.time = time;
        this.count = count;
    }



    public String getTime() {
        return time;
    }

    public int getCount() {
        return count;
    }

    public void setTime(String time) {
        this.time = time;
    }

    public void setCount(int count) {
        this.count = count;
    }



    @Override
    public String toString() {
```

```
        return (new StringBuilder().append("Peak Hour:").append(time)
                .append("-").append("Total Rides").append(count).toString());
    }

    @Override
    public void write(DataOutput d) throws IOException {
        WritableUtils.writeString(d, time);
      WritableUtils.writeVInt(d, count);
    }

    @Override
    public void readFields(DataInput di) throws IOException {
        this.time = WritableUtils.readString(di);
       this.count = WritableUtils.readVInt(di);
    }


}
```

PeakTripTimeMapper:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package PeakTripTime;

import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

/**
 *
 * @author nirmal
 */
public class PeakTripTimeMapper extends Mapper<Object,Text,Text,PTTWritable>{

    private PTTWritable pttTuple=new PTTWritable();

    @Override
    protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {

        String tokens[]=value.toString().split(",");
        if(tokens.length==21){
        if(!(tokens[0].equals("")) && !(tokens[0].equalsIgnoreCase("VendorID"))){

        String time=tokens[1].split(" ")[1].split(":")[0];
        pttTuple.setTime(time);
        pttTuple.setCount(1);
        context.write(new Text(tokens[1].split(" ")[0]), pttTuple);
        }
        }

    }

}
```

PeakTripTimeReducer:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
```

```
 */
package PeakTripTime;

import java.io.EOFException;
import java.io.IOException;
import java.util.Collections;
import java.util.Comparator;
import java.util.HashMap;
import java.util.Iterator;
import java.util.LinkedHashMap;
import java.util.LinkedList;
import java.util.List;
import java.util.Map;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nirmal
 */
public class PeakTripTimeReducer extends Reducer<Text,PTTWritable,Text,PTTWritable>{

   private PTTWritable pttw=new PTTWritable();

   @Override
   protected void reduce(Text key, Iterable<PTTWritable> values, Context context) throws IOException, InterruptedException ,EOFException{

      HashMap<String,Integer> ptHash=new HashMap<String,Integer>();
      for(PTTWritable ptt:values){
         if(ptHash.containsKey(ptt.getTime())){
            ptHash.put(ptt.getTime(), ptHash.get(ptt.getTime()) + 1);
         }else{
            ptHash.put(ptt.getTime(), ptt.getCount());
         }
      }

      List list = new LinkedList(ptHash.entrySet());

     // Defined Custom Comparator here
     Collections.sort(list, new Comparator() {
         public int compare(Object o1, Object o2) {
            return ((Comparable) ((Map.Entry) (o1)).getValue())
               .compareTo(((Map.Entry) (o2)).getValue()) * -1;
         }
     });

     // Here I am copying the sorted list in HashMap
     // using LinkedHashMap to preserve the insertion order
     HashMap sortedHashMap = new LinkedHashMap();
     for (Iterator it = list.iterator(); it.hasNext();) {
          Map.Entry entry = (Map.Entry) it.next();
          sortedHashMap.put(entry.getKey(), entry.getValue());
     }

     String str=(String) sortedHashMap.keySet().iterator().next();
     int val=(int) sortedHashMap.get(str);
     pttw.setTime(str);
     pttw.setCount(val);

     context.write(key, pttw);

   }


   }
```

PeakTripTime:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package PeakTripTime;

import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author nirmal
 */
public class PeakTripTime {

   public static void main(String args[]) throws IOException, InterruptedException, ClassNotFoundException{
      Configuration conf = new Configuration();
         Job job = Job.getInstance(conf, "PeakTripTime");
         job.setJarByClass(PeakTripTime.class);

         job.setMapperClass(PeakTripTimeMapper.class);
         job.setMapOutputKeyClass(Text.class);
         job.setMapOutputValueClass(PTTWritable.class);

        job.setCombinerClass(PeakTripTimeReducer.class);
         job.setReducerClass(PeakTripTimeReducer.class);
         job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(PTTWritable.class);
         //job.setNumReduceTasks(0);
         FileInputFormat.addInputPath(job, new Path(args[0]));
         FileOutputFormat.setOutputPath(job, new Path(args[1]));

         System.exit(job.waitForCompletion(true) ? 0 : 1);
   }

}
```

Analysis 2:

TripsMapper

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package TripsPerDay;

import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

/**
 *
 * @author nirmal
 */
```

```
public class TripsMapper extends Mapper<Object,Text,Text,IntWritable>{

    @Override
    protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {

        String tokens[]=value.toString().split(",");
        if(!(tokens[0].equals("")) && !(tokens[0].equalsIgnoreCase("VendorID"))){
            context.write(new Text(tokens[1].split(" ")[0]), new IntWritable(1));
        }

    }



}
```

TripsReducer

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package TripsPerDay;

import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nirmal
 */
public class TripsReducer extends Reducer<Text,IntWritable,Text,IntWritable>{

     private IntWritable result = new IntWritable();
    @Override
    protected void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {

        int sum = 0;
        for(IntWritable val : values){
            sum += val.get();
        }
        result.set(sum);

       context.write(key,result);
    }



}
```

Trips

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package TripsPerDay;

import CarrerGroupCount.ACReducer;
import PeakTripTime.PTTWritable;
import PeakTripTime.PeakTripTime;
```

```
import PeakTripTime.PeakTripTimeMapper;
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author nirmal
 */
public class Trips {

   public static void main(String args[]) throws IOException, InterruptedException, ClassNotFoundException{
      Configuration conf = new Configuration();
         Job job = Job.getInstance(conf, "PeakTripTime");
         job.setJarByClass(Trips.class);

         job.setMapperClass(TripsMapper.class);
         job.setMapOutputKeyClass(Text.class);
         job.setMapOutputValueClass(IntWritable.class);

        job.setReducerClass(TripsReducer.class);
      job.setOutputKeyClass(Text.class);
      job.setOutputValueClass(IntWritable.class);

         FileInputFormat.addInputPath(job, new Path(args[0]));
         FileOutputFormat.setOutputPath(job, new Path(args[1]));

         System.exit(job.waitForCompletion(true) ? 0 : 1);
   }

}
```

GreenMapper:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package CompareTripsPerDay;

import java.io.IOException;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

/**
 *
 * @author nirmal
 */
public class GreenMapper extends Mapper<Object,Text,Text,Text>{

   private Text outKey=new Text();
   private Text outValue=new Text();

   @Override
   protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {
      String tokens[]=value.toString().split("\\t");
      outKey.set(tokens[0]);
      outValue.set("G"+tokens[1]);
      context.write(outKey,outValue);
   }
```

YellowMapper

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package CompareTripsPerDay;

import java.io.IOException;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

/**
 *
 * @author nirmal
 */
public class YellowMapper extends Mapper<Object,Text,Text,Text>{

    private Text outKey=new Text();
   private Text outValue=new Text();


   @Override
   protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {

      String tokens[]=value.toString().split("\\t");
      outKey.set(tokens[0]);
      outValue.set("Y"+tokens[1]);
      context.write(outKey,outValue);

   }



}
```

YGReducer

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package CompareTripsPerDay;

import java.io.IOException;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nirmal
 */
public class YGReducer extends Reducer<Text,Text,Text,Text>{

   private Text tmp=new Text();
   private String greenTaxi,yellowTaxi;

   @Override
   protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {

      while(values.iterator().hasNext()){
            tmp=values.iterator().next();
            if(tmp.charAt(0)=='G'){
              greenTaxi= tmp.toString();
```

```
            }
            else if(tmp.charAt(0)=='Y'){
               yellowTaxi=tmp.toString();

            }

         }

      if(greenTaxi!=null && yellowTaxi!=null){
           context.write(key, new Text(greenTaxi+","+yellowTaxi));
      }
      else if(greenTaxi==null){
         context.write(key, new Text("G-0"+","+yellowTaxi));
      }
      else if(yellowTaxi==null){
           context.write(key, new Text(greenTaxi+","+"Y-0"));
      }


   }



}
```

TripsJoin

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package CompareTripsPerDay;

import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.MultipleInputs;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

/**
 *
 * @author nirmal
 */
public class TripsJoin {

   public static void main(String args[]) throws IOException, InterruptedException, ClassNotFoundException{
      Configuration conf=new Configuration();
      Job job=Job.getInstance(conf, "InnerJoin");
      job.setJarByClass(TripsJoin.class);

      MultipleInputs.addInputPath(job, new Path(args[0]),TextInputFormat.class, GreenMapper.class);
      MultipleInputs.addInputPath(job, new Path(args[1]),TextInputFormat.class, YellowMapper.class);

      job.setReducerClass(YGReducer.class);
      job.setOutputFormatClass(TextOutputFormat.class);
      TextOutputFormat.setOutputPath(job, new Path(args[2]));
      job.setOutputKeyClass(Text.class);
      job.setOutputValueClass(Text.class);
      System.exit(job.waitForCompletion(true) ? 0 : 1);
   }

}
```

Analysis 3:

DistanceFareMapper

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package DistanceFareSorting;

import java.io.IOException;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

/**
 *
 * @author nirmal
 */
public class DistanceFareMapper extends Mapper<Object,Text,DistanceFare,DistanceFareRecord>{

   private DistanceFare df=new DistanceFare();
   private DistanceFareRecord dfr=new DistanceFareRecord();

   @Override
   protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {


      String tokens[]=value.toString().split(",");
      if(tokens.length==21){
      if(!(tokens[0].equals("")) && !(tokens[0].equalsIgnoreCase("VendorID"))){

      String date=tokens[1].split(" ")[0];





      double dist=Double.parseDouble(tokens[10]);
      double far=Double.parseDouble(tokens[11]);

      int distance=(int) dist;
      int fare=(int) far;
      if(fare>0){

      df.setDistance(distance);
      df.setFare(fare);

      dfr.setDate(date);
      dfr.setDistance(dist);
      dfr.setFare(far);

      context.write(df, dfr);
      }

      }
      }

   }



}
```

DistanceFareReducer

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package DistanceFareSorting;

import java.io.IOException;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nirmal
 */
public class DistanceFareReducer extends Reducer<DistanceFare,DistanceFareRecord,NullWritable,DistanceFareRecord>{

    @Override
    protected void reduce(DistanceFare key, Iterable<DistanceFareRecord> values, Context context) throws IOException, InterruptedException {

        for (DistanceFareRecord dfr:values){
            context.write(NullWritable.get(),dfr);
        }

    }



}
```

DistanceFareRecord

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package DistanceFareSorting;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import org.apache.hadoop.io.Writable;
import org.apache.hadoop.io.WritableUtils;

/**
 *
 * @author nirmal
 */
public class DistanceFareRecord implements Writable{

    private String date;
    private double distance;
    private double fare;

    public DistanceFareRecord(){
        this.date=null;
        this.distance=0;
        this.fare=0;
    }

    public String getDate() {
        return date;
    }
```

```
    public void setDate(String date) {
        this.date = date;
    }

    public double getDistance() {
        return distance;
    }

    public void setDistance(double distance) {
        this.distance = distance;
    }

    public double getFare() {
        return fare;
    }

    public void setFare(double fare) {
        this.fare = fare;
    }



    @Override
    public void write(DataOutput d) throws IOException {
         //To change body of generated methods, choose Tools | Templates.
         WritableUtils.writeString(d, date);
        d.writeDouble(distance);
        d.writeDouble(fare);
    }

    @Override
    public void readFields(DataInput di) throws IOException {
        date = WritableUtils.readString(di);
        distance = di.readDouble();
        fare=di.readDouble();
    }

    public String toString(){
        return (new StringBuilder().append(date)
                .append(",").append(distance).append(",").append(fare).toString());

        }
}
```

DistancePartitioner

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package DistanceFareSorting;

import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.mapreduce.Partitioner;

/**
 *
 * @author nirmal
 */
public class DistancePartitioner extends Partitioner<DistanceFare,NullWritable>{

    @Override
    public int getPartition(DistanceFare key, NullWritable value, int i) {
```

```
        //To change body of generated methods, choose Tools | Templates.
        return  (key.getDistance() % i);
    }

}
```

GroupComparator

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package DistanceFareSorting;

import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.io.WritableComparator;

/**
 *
 * @author nirmal
 */
public class GroupComparator extends WritableComparator{

    protected GroupComparator()
    {
        super(DistanceFare.class,true);
    }

    @Override
    public int compare(WritableComparable w1, WritableComparable w2){
        DistanceFare cw1 = (DistanceFare) w1;
        DistanceFare cw2 = (DistanceFare) w2;

        return cw1.getDistance().compareTo(cw2.getDistance());
    }

}
```

KeyComparator

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package DistanceFareSorting;

import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.io.WritableComparator;

/**
 *
 * @author nirmal
 */
public class KeyComparator extends WritableComparator{

    protected KeyComparator(){
        super(DistanceFare.class,true);
    }

    @Override
    public int compare(WritableComparable a, WritableComparable b) {

        DistanceFare df1 = (DistanceFare) a;
        DistanceFare df2 = (DistanceFare) b;
```

```
        int cmp = df1.getDistance().compareTo(df2.getDistance());
            if (cmp != 0) {
                return cmp;
             }
        return df1.getFare().compareTo(df2.getFare());

    }



}
```

DFSorting

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package DistanceFareSorting;

import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author nirmal
 */
public class DFSorting {

    public static void main(String args[]) throws IOException, InterruptedException, ClassNotFoundException{


        Configuration conf = new Configuration();
            Job job = Job.getInstance(conf,"SecondarySort");
            job.setJarByClass(DFSorting.class);

            job.setMapperClass(DistanceFareMapper.class);
            job.setMapOutputKeyClass(DistanceFare.class);
            job.setMapOutputValueClass(DistanceFareRecord.class);

            job.setSortComparatorClass(KeyComparator.class);
            job.setPartitionerClass(DistancePartitioner.class);
            job.setGroupingComparatorClass(GroupComparator.class);

            job.setReducerClass(DistanceFareReducer.class);
            job.setOutputKeyClass(NullWritable.class);
            job.setOutputValueClass(DistanceFareRecord.class);

            FileInputFormat.addInputPath(job, new Path(args[0]));
            FileOutputFormat.setOutputPath(job, new Path(args[1]));



            System.exit(job.waitForCompletion(true) ? 0 : 1);
    }

}
```

Analysis 4:

DOWMapper

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package DayOfWeekRides;

import java.io.IOException;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.Locale;
import java.util.logging.Level;
import java.util.logging.Logger;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.lib.output.MultipleOutputs;

/**
 *
 * @author nirmal
 */
public class DOWMapper extends Mapper<Object,Text,Text,NullWritable>{

   public MultipleOutputs<Text,NullWritable> mos=null;
   private Text output;
   @Override
   protected void cleanup(Context context) throws IOException, InterruptedException {
      mos.close();
   }

   @Override
   protected void map(Object key, Text value, Context context) throws IOException, InterruptedException{

      try {

         float pickup_long,pickup_lat,drop_long,drop_lat;
         String pickup,drop;

         String tokens[]=value.toString().split(",");
         if(!(tokens[0].equals("")) && !(tokens[0].equalsIgnoreCase("VendorID"))){
         String s=tokens[1];
         SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
         Date date=formatter.parse(s);
         Calendar cal = Calendar.getInstance();
         cal.setTime(date);
         String dayOfWeek = cal.getDisplayName( Calendar.DAY_OF_WEEK ,Calendar.LONG, Locale.getDefault());

         pickup_long=Float.parseFloat(tokens[5]);
         pickup_lat=Float.parseFloat(tokens[6]);
         drop_long=Float.parseFloat(tokens[7]);
         drop_lat=Float.parseFloat(tokens[8]);

   if((pickup_long > -74.016309 && pickup_long<-73.943986) && (pickup_lat>40.703363 && pickup_lat< 40.822683)) {
         pickup="Manhattan";
   }else if((pickup_long>-73.996224 && pickup_long< -73.904354) && (pickup_lat>40.590195 && pickup_lat< 40.699933)){
      pickup="Brooklyn";
   }else if((pickup_long>-73.929015 && pickup_long< -73.732468) && (pickup_lat>40.667349 && pickup_lat< 40.782815)){
     pickup="Queen";
```

```
}else if((pickup_long>-73.909672 && pickup_long< -73.814439) && (pickup_lat>40.808118 && pickup_lat< 40.910298)){
   pickup="Bronx";
}else if((pickup_long>-74.241142 && pickup_long< -74.093445) && (pickup_lat>40.506272 && pickup_lat< 40.643899)){
   pickup="Staten Island";
}else if((pickup_long>-73.815939 && pickup_long< -73.764694) && (pickup_lat>40.636008 && pickup_lat< 40.662522)){
   pickup="JFK Airport";
}else{
   pickup="UnKnown";
}

if((drop_long > -74.016309 && drop_long<-73.943986) && (drop_lat>40.703363 && drop_lat< 40.822683)) {
      drop="Manhattan";
}else if((drop_long>-73.996224 && drop_long< -73.904354) && (drop_lat>40.590195 && drop_lat< 40.699933)){
   drop="Brooklyn";
}else if((drop_long>-73.929015 && drop_long< -73.732468) && (drop_lat>40.667349 && drop_lat< 40.782815)){
  drop="Queen";
}else if((drop_long>-73.909672 && drop_long< -73.814439) && (drop_lat>40.808118 && drop_lat< 40.910298)){
   drop="Bronx";
}else if((drop_long>-74.241142 && drop_long< -74.093445) && (drop_lat>40.506272 && drop_lat< 40.643899)){
   drop="Staten Island";
}else if((drop_long>-73.815939 && drop_long< -73.764694) && (drop_lat>40.636008 && drop_lat< 40.662522)){
   drop="JFK Airport";
}else{
   drop="UnKnown";
}


    output=new Text(value+","+pickup+","+drop);

      if(dayOfWeek.equalsIgnoreCase("Monday")){
        mos.write("bins", output, NullWritable.get(), "Monday");
      }
      if(dayOfWeek.equalsIgnoreCase("Tuesday")){
        mos.write("bins", output, NullWritable.get(), "Tuesday");
      }
      if(dayOfWeek.equalsIgnoreCase("Wednesday")){
        mos.write("bins", output, NullWritable.get(), "Wednesday");
      }
      if(dayOfWeek.equalsIgnoreCase("Thursday")){
        mos.write("bins", output, NullWritable.get(), "Thursday");
      }
      if(dayOfWeek.equalsIgnoreCase("Friday")){
        mos.write("bins", output, NullWritable.get(), "Friday");
      }
      if(dayOfWeek.equalsIgnoreCase("Saturday")){
        mos.write("bins", output, NullWritable.get(), "Saturday");
      }
      if(dayOfWeek.equalsIgnoreCase("Sunday")){
        mos.write("bins", output, NullWritable.get(), "Sunday");
      }


      }

   } catch (ParseException ex) {
      Logger.getLogger(DOWMapper.class.getName()).log(Level.SEVERE, null, ex);
   }

}

@Override
protected void setup(Context context) throws IOException, InterruptedException {
   mos=new MultipleOutputs(context);
}
```

```
}
```

DOW

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package DayOfWeekRides;

import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.MultipleOutputs;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

/**
 *
 * @author nirmal
 */
public class DOW {

   public static void main(String args[]) throws IOException, InterruptedException, ClassNotFoundException
   {
   Configuration conf = new Configuration();
      Job job = Job.getInstance(conf, "Bining Pattern");
      job.setJarByClass(DOW.class);
      job.setMapperClass(DOWMapper.class);


      MultipleOutputs.addNamedOutput(job, "bins", TextOutputFormat.class, Text.class, NullWritable.class);
      MultipleOutputs.setCountersEnabled(job, true);

      job.setNumReduceTasks(0);

      FileInputFormat.addInputPath(job, new Path(args[0]));
      FileOutputFormat.setOutputPath(job, new Path(args[1]));
      System.exit(job.waitForCompletion(true) ? 0 : 1);

   }

}
```

Analysis 5:

BloomFilterMapper

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package BLFilter;

import com.google.common.base.Charsets;
import com.google.common.hash.BloomFilter;
import com.google.common.hash.Funnel;
import com.google.common.hash.Sink;
```

```
import java.io.IOException;
import java.nio.charset.Charset;
import java.util.ArrayList;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

/**
 *
 * @author nirmal
 */
public class BloomFilterMapper extends Mapper<Object,Text,Text,NullWritable>{

   Funnel<PickUpDrop> pd=new Funnel<PickUpDrop>() {
      @Override
      public void funnel(PickUpDrop t, Sink sink) {
        sink.putString(t.Pickup, Charsets.UTF_8).putString(t.drop,Charsets.UTF_8);
      }
   };

   BloomFilter<PickUpDrop> filter=BloomFilter.create(pd,500,0.1);

   @Override
   protected void setup(Context context) throws IOException, InterruptedException {
       //To change body of generated methods, choose Tools | Templates.
       PickUpDrop pt=new PickUpDrop("Brooklyn", "Manhattan");
      ArrayList<PickUpDrop> list = new ArrayList<PickUpDrop>();
      list.add(pt);

      for(PickUpDrop p:list){
         filter.put(p);
      }
   }



   @Override
   protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {
      String line[] = value.toString().split(",");
      PickUpDrop pudp;
      String pickup=line[21];
      String drop=line[22];
      if((!pickup.equalsIgnoreCase("")||pickup!=null || !pickup.equalsIgnoreCase("null"))
            &&
         (!drop.equalsIgnoreCase("")||drop!=null || !drop.equalsIgnoreCase("null"))){

         pudp = new PickUpDrop(pickup,drop);


         if(filter.mightContain(pudp)){
            context.write(value,NullWritable.get());
         }

   }


   }
```

PickUpDrop

```
}

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package BLFilter;
```

```
/**
 *
 * @author nirmal
 */
public class PickUpDrop {

    final String Pickup;
    final String drop;

    public PickUpDrop(String pick,String dr){
        this.Pickup=pick;
        this.drop=dr;
    }

}
```

AverageTuple

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package BLFilter;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import org.apache.hadoop.io.Writable;
import org.apache.hadoop.io.WritableUtils;

/**
 *
 * @author nirmal
 */
public class AverageTuple implements Writable{

    private int count;
    private double tip;

    public AverageTuple(int c,double d){
        this.count=c;
        this.tip=d;
    }

    public AverageTuple() {
        count=0;
        tip=0;

// throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.
    }

    public int getCount() {
        return count;
    }

    public void setCount(int count) {
        this.count = count;
    }

    public double getTip() {
        return tip;
    }

    public void setTip(double tip) {
        this.tip = tip;
    }
```

```
    @Override
    public void write(DataOutput d) throws IOException {
        d.writeDouble(tip);
        d.writeInt(count);
    }

    @Override
    public void readFields(DataInput di) throws IOException {

        count=di.readInt();
        tip=di.readDouble();

    }



}
```

TipHourlyMapper

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package BLFilter;

import PeakTripTime.PTTWritable;
import java.io.IOException;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

/**
 *
 * @author nirmal
 */
public class TipHourlyMapper extends Mapper<Object,Text,Text,AverageTuple>{

    private AverageTuple avg=new AverageTuple();

    @Override
    protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {

        String tokens[]=value.toString().split(",");
        String time=tokens[1].split(" ")[1].split(":")[0];

        avg.setTip(Double.parseDouble(tokens[14]));
        avg.setCount(1);

        context.write(new Text(time),avg);


    }



}
```

TipHourlyReducer

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package BLFilter;

import java.io.IOException;
```

```
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nirmal
 */
public class TipHourlyReducer extends Reducer<Text,AverageTuple,Text,DoubleWritable>{

    private AverageTuple avgTuple=new AverageTuple();

   @Override
   protected void reduce(Text key, Iterable<AverageTuple> values, Context context) throws IOException, InterruptedException {

      double sum=0;
      int count=0;

      for(AverageTuple tuple:values){
         sum+=tuple.getCount()*tuple.getTip();
         count+=tuple.getCount();

      }

      double average=sum/count;


      context.write(key,new DoubleWritable(average));


   }



}
```

BLFilter:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package BLFilter;

import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author nirmal
 */
public class BLFilter {

   public static void main(String[] args) throws IOException, InterruptedException, ClassNotFoundException {
      // TODO code application logic here
      Configuration conf = new Configuration();
      Job job = Job.getInstance(conf,"Bloom Filter");
      job.setJarByClass(BLFilter.class);
      job.setMapperClass(BloomFilterMapper.class);
```

```
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(NullWritable.class);
        job.setNumReduceTasks(0);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        boolean complete = job.waitForCompletion(true);

       Configuration conf2 = new Configuration();
       Job job2 = Job.getInstance(conf2, "chaining");
       if (complete) {
          job2.setJarByClass(BLFilter.class);
          job2.setMapperClass(TipHourlyMapper.class);
          job2.setMapOutputKeyClass(Text.class);
          job2.setMapOutputValueClass(AverageTuple.class);

          job2.setReducerClass(TipHourlyReducer.class);
          job2.setOutputKeyClass(Text.class);
          job2.setOutputValueClass(DoubleWritable.class);

          FileInputFormat.addInputPath(job2, new Path(args[1]));
          FileOutputFormat.setOutputPath(job2, new Path(args[2]));

          System.exit(job2.waitForCompletion(true) ? 0 : 1);
       }
    }

}
```

Analysis 6:

TopPickupDropMapper

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package TopPickUpDrops;

import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

/**
 *
 * @author nirmal
 */
public class TopPickupDropMapper extends Mapper<Object,Text,Text,IntWritable>{

    private Text outkey=new Text();
   // private PickDropTuple pd=new PickDropTuple();

    @Override
    protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {
        String tokens[]=value.toString().split(",");
        String pickup,drop;
        pickup=tokens[21];
        drop=tokens[22];
        if((!pickup.equalsIgnoreCase("UnKnown")) && (!drop.equalsIgnoreCase("UnKnown"))){
           // pd.setPick(pickup);
           // pd.setDrop(drop);
            outkey=new Text(pickup+"-"+drop);
            context.write(outkey,new IntWritable(1));
```

```
        }
    }



}
```

TopPickupDropReducer

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package TopPickUpDrops;

import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nirmal
 */
public class TopPickupDropReducer extends Reducer<Text,IntWritable,Text,IntWritable>{

    @Override
    protected void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {

            int sum=0;
            for(IntWritable val:values){
                sum=sum+val.get();
            }
            context.write(key, new IntWritable(sum));

        }



}
```

PickUpDrops

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package TopPickUpDrops;

import PeakTripTime.PTTWritable;
import PeakTripTime.PeakTripTime;
import PeakTripTime.PeakTripTimeMapper;
import PeakTripTime.PeakTripTimeReducer;
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
```

```
 *
 * @author nirmal
 */
public class PickUpDrops {

   public static void main(String args[]) throws IOException, InterruptedException, ClassNotFoundException{

       Configuration conf = new Configuration();
         Job job = Job.getInstance(conf, "PeakTripTime");
         job.setJarByClass(PickUpDrops.class);

         job.setMapperClass(TopPickupDropMapper.class);
         job.setMapOutputKeyClass(Text.class);
         job.setMapOutputValueClass(IntWritable.class);


         job.setReducerClass(TopPickupDropReducer.class);
         job.setOutputKeyClass(Text.class);
         job.setOutputValueClass(IntWritable.class);

         FileInputFormat.addInputPath(job, new Path(args[0]));
         FileOutputFormat.setOutputPath(job, new Path(args[1]));

         System.exit(job.waitForCompletion(true) ? 0 : 1);
   }

}
```

Analysis 7:

MDTMapper

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package MostDistanceTravelled;

import java.io.IOException;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

/**
 *
 * @author nirmal
 */
public class MDTMapper extends Mapper<Object,Text,Text,FloatWritable>
{
   FloatWritable distance;

   @Override
   protected void map(Object key, Text value, Context context) throws IOException, InterruptedException {
      String tokens[]=value.toString().split(",");
      if(!("VendorID".equalsIgnoreCase(tokens[0]))){
      distance=new FloatWritable(Float.parseFloat(tokens[10]));

      context.write(new Text(tokens[21]),distance);
      }
   }

}
```

MDTReducer

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package MostDistanceTravelled;

import java.io.IOException;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nirmal
 */
public class MDTReducer extends Reducer<Text,FloatWritable,Text,FloatWritable>{



    @Override
    protected void reduce(Text key, Iterable<FloatWritable> values, Context context) throws IOException, InterruptedException {

        float max=0;
        for(FloatWritable val:values){
            if(val.get()>max){
                max=val.get();
            }
        }
        context.write(key, new FloatWritable(max));
    }



}
```

MDT:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package MostDistanceTravelled;

import CarrerGroupCount.ACGroup;
import CarrerGroupCount.ACMapper;
import CarrerGroupCount.ACReducer;
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author nirmal
 */
public class MDT {
```

```
  public static void main(String args[]) throws IOException, InterruptedException, ClassNotFoundException{

  Configuration conf = new Configuration();
    Job job = Job.getInstance(conf,"IP Access");
    job.setJarByClass(MDT.class);
    job.setMapperClass(MDTMapper.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(FloatWritable.class);
    job.setReducerClass(MDTReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(FloatWritable.class);

    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }

}
```

HIVE CODE

```
CREATE TABLE greentaxi
 (
   vendorid       int,
   pick_up_date    string,
   drop_date       string,
   flag            CHAR(1),
   rate_code       INT,
   pick_up_long    string,
   pick_up_lat     string,
   drop_off_long   string,
   drop_off_lat    string,
   passenger_count INT,
   trip_distance   DECIMAL(5,2),
   fare_amount     DECIMAL(5,2),
   extra           DECIMAL(5,2),
   tax             DECIMAL(5,2),
   tip             DECIMAL(5,2),
   tolls           DECIMAL(5,2),
   surcharge       DECIMAL(5,2),
   total_amount    DECIMAL(5,2),
   payment_type    int,
   trip_type       int
 )
comment 'Data about Green NYC Taxi for the year 2016-Jan'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

Select t.ts,sum(t.value) from
(select TO_DATE(from_unixtime(UNIX_TIMESTAMP(dates,'mm/dd/yy hh:mm'))) as ts ,value from dates) t group by
t.ts;


select g.pickup_date,sum(g.total_amount) from
```

```
(select TO_DATE(from_unixtime(UNIX_TIMESTAMP(pick_up_date,'mm/dd/yy hh:mm'))) as pickup_date
,total_amount from greentaxi) g group by g.pickup_date;


select g.pickup_date,sum(g.passenger_count) from
(select TO_DATE(from_unixtime(UNIX_TIMESTAMP(pick_up_date,'mm/dd/yy hh:mm'))) as pickup_date
,passenger_count from greentaxi) g group by g.pickup_date;
```

Pig Scripts:

AmountPerDay:

```
greenTaxi = LOAD 'hdfs://localhost:9000/FinalProject/Pig/Pig_greenTaxi.csv' USING PigStorage(',');
dataTaxi = FOREACH greenTaxi GENERATE $1 as date,$13 as fare,$16 as tip;
grp = GROUP dataTaxi BY date;
cnt = FOREACH grp GENERATE group,SUM(dataTaxi.fare),SUM(dataTaxi.tip);
STORE cnt INTO ' hdfs://localhost:9000/FinalProject/Pig/Output1 ' USING PigStorage (',');
```

MaxTripRide:

```
taxi = LOAD 'hdfs://localhost:9000/FinalProject/Pig/Pig_greenTaxi.csv' USING PigStorage(',');
data = FOREACH taxi GENERATE $1 as pickupDate,$13 as fare,$16 as trip,$19 as total;
filtered = FILTER data BY $1*$2*$3 is not null;
grped = GROUP filtered by pickupDate;
amt = FOREACH grped GENERATE group,MAX(filtered.fare),MAX(filtered.trip),MAX(filtered.total);
STORE amt INTO ' hdfs://localhost:9000/FinalProject/Pig/Output2 ' USING PigStorage (',');
```

TipPercent:

```
tipData = LOAD 'hdfs://localhost:9000/FinalProject/Pig/Pig_greenTaxi.csv' USING PigStorage(',');
filterData = FOREACH tipData GENERATE $1 as pickupDate,$2 as pickupTime,$16 as tip,$19 as total;
tipPercent = FOREACH filterData GENERATE $0,$1,($2*100)/($3-$2);
STORE tipPercent INTO ' hdfs://localhost:9000/FinalProject/Pig/Output3 ' USING PigStorage (',');
```