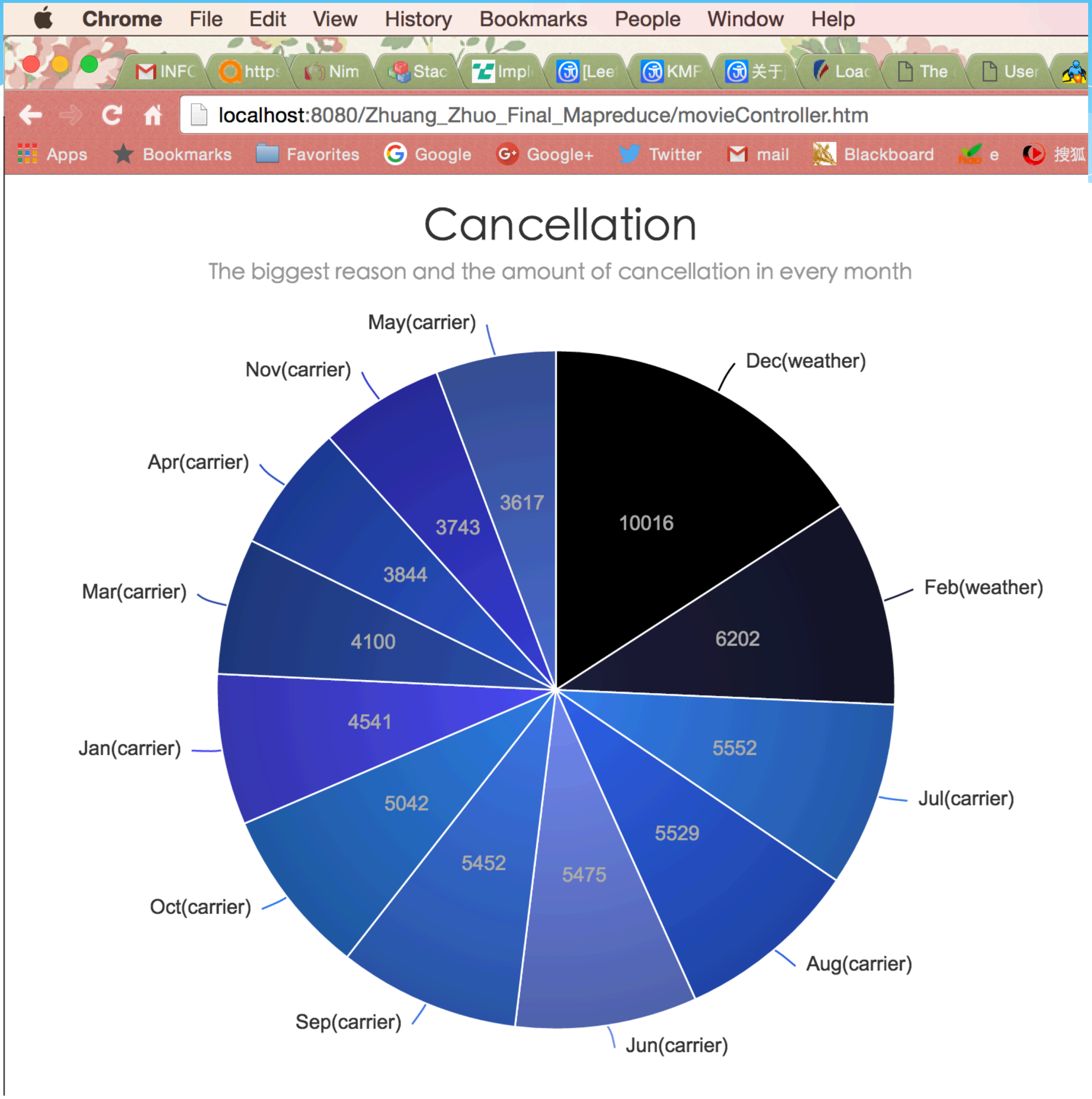
Contents

Introduction of dataset

- Airlines (2006)
- http://stat-computing.org/dataexpo/2009/the-data.html

Name		Description
1	Year	1987-2008
2	Month	1-12
3	DayofMonth	1-31
4	DayOfWeek	1 (Monday) - 7 (Sunday)
5	DepTime	actual departure time (local, hhmm)
6	CRSDepTime	scheduled departure time (local, hhmm)
7	ArrTime	actual arrival time (local, hhmm)
8	CRSArrTime	scheduled arrival time (local, hhmm)
9	UniqueCarrier	unique carrier code
10	FlightNum	flight number
11	TailNum	plane tail number
12	ActualElapsedTime	in minutes
13	CRSElapsedTime	in minutes
14	AirTime	in minutes
15	ArrDelay	arrival delay, in minutes
16	DepDelay	departure delay, in minutes
17	Origin	origin IATA airport code
18	Dest	destination IATA airport code
19	Distance	in miles
20	TaxiIn	taxi in time, in minutes
21	TaxiOut	taxi out time in minutes
22	Cancelled	was the flight cancelled?
23	CancellationCode	reason for cancellation (A = carrier, B = weather, C = NAS, D = security)
24	Diverted	1 = yes, 0 = no
25	CarrierDelay	in minutes
26	WeatherDelay	in minutes
27	NASDelay	in minutes
28	SecurityDelay	in minutes
29	LateAircraftDelay	in minutes

Mapreduce with HDFS (1)



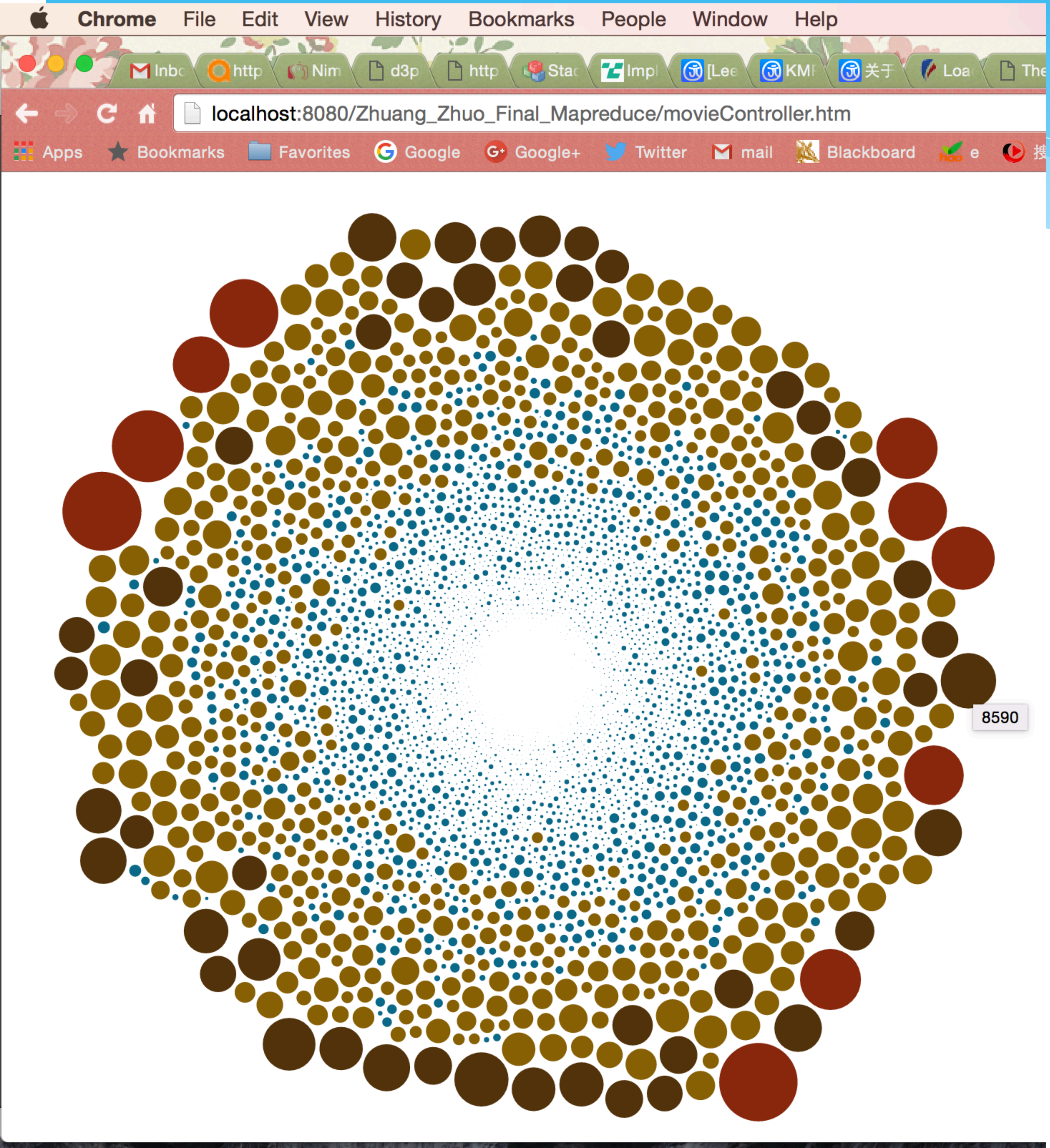
Cancellation

The biggest reason and the amount of cancellation in every month

Dec(weather)	10016
Feb(weather)	6202
Jul(carrier)	5552
Aug(carrier)	5529
Jun(carrier)	5475
Sep(carrier)	5452
Oct(carrier)	5042
Jan(carrier)	4541
Mar(carrier)	4100
Apr(carrier)	3844
Nov(carrier)	3743
May(carrier)	3617

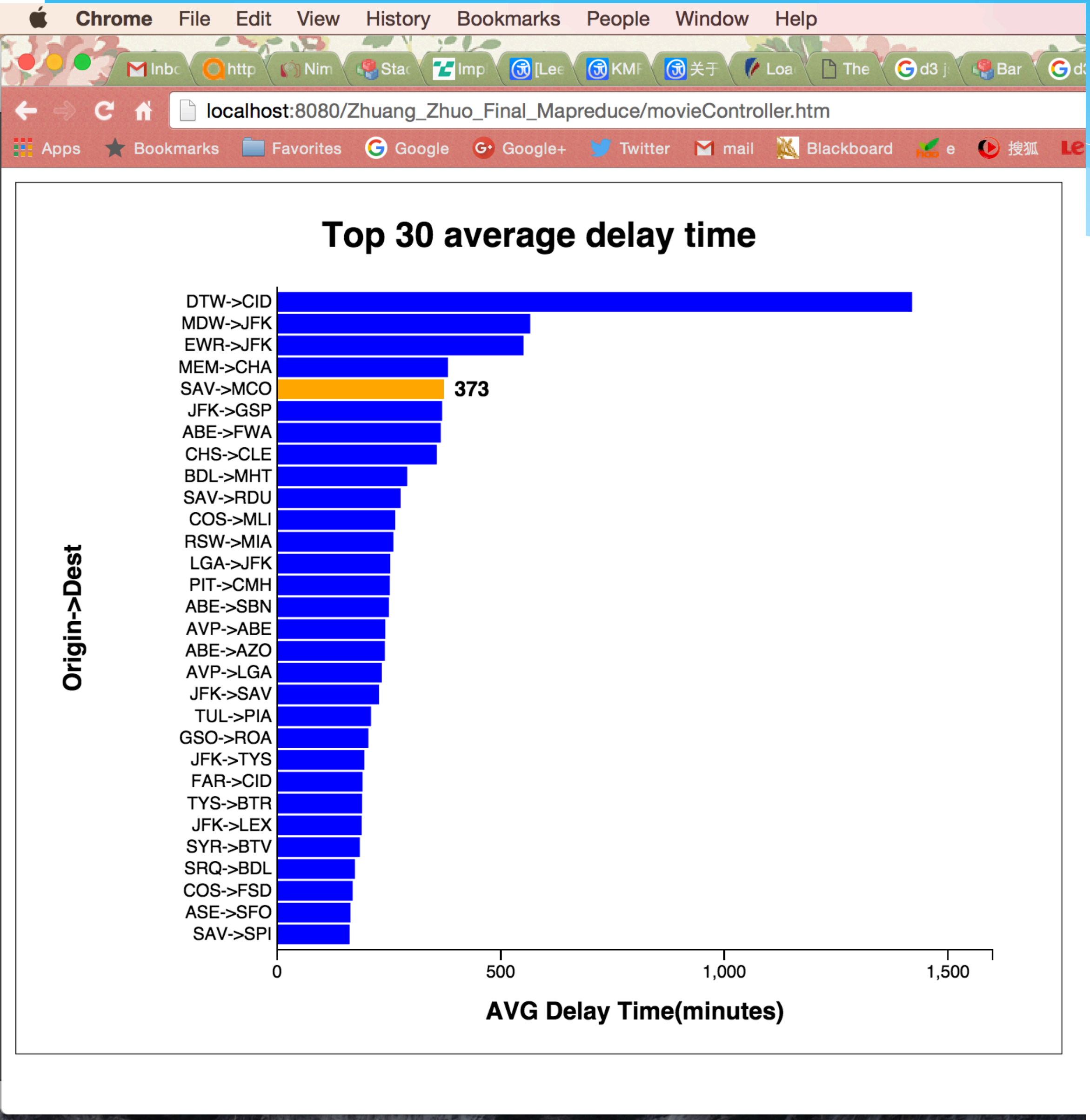
- 1. The biggest reason for cancelation and its amount of cancelation Flights in every month
- This is a chain Mapreduce
- Use spring mvc and d3.js to show the data

Mapreduce with HDFS (2)


Chrome File Edit View History Bookmarks People Window Help
localhost:8080/Zhuang_Zhuo_Final_Mapreduce/movieController.htm
Apps
Bookmarks
Favorites
Google
Google+
Twitter
mail
Blackboard
8590

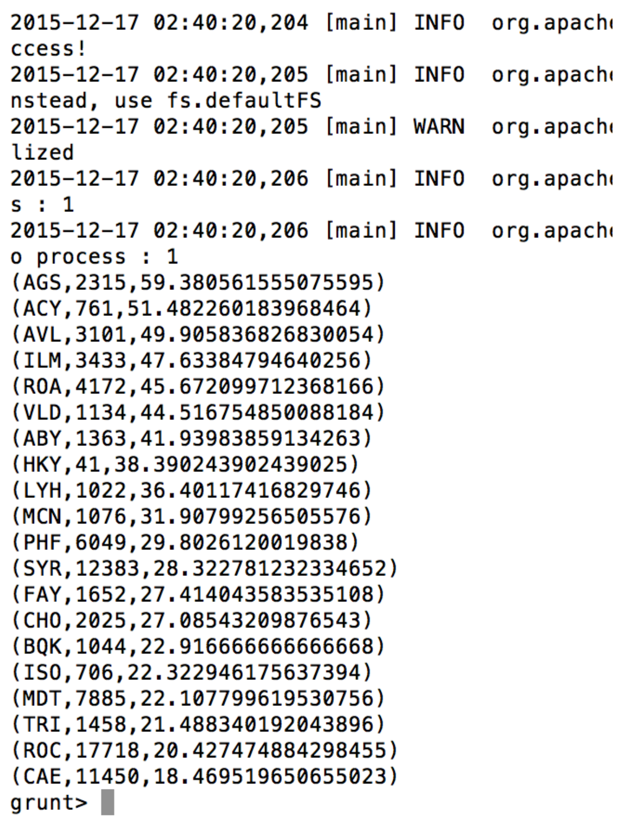
- 2.1 the delay time and the amount of flights of the airlines(from Origin to destination)
- This is the first job of the chain mapreduce.
- The radius of the circles is the total delay time of the airlines, and the outer circles have more flights than the inner ones.

Mapreduce with HDFS (2)



- 2.2 the top 3 oaverage delay time of the airlines(from Origin to destination).
- This is the third job of the chain mapreduce.

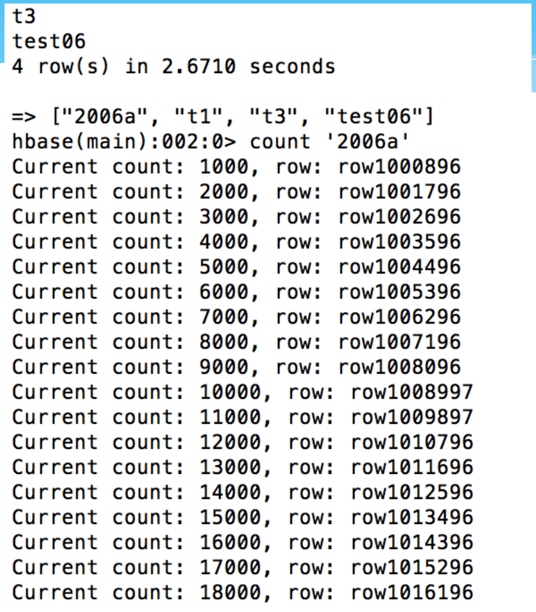
Analysis with pig



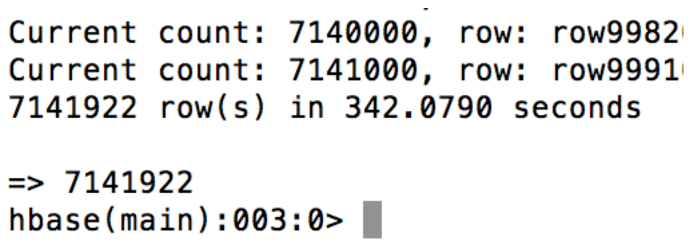
```
2015-12-17 02:40:20,204 [main] INFO  org.apach
ccess!
2015-12-17 02:40:20,205 [main] INFO  org.apach
nstead, use fs.defaultFS
2015-12-17 02:40:20,205 [main] WARN  org.apach
lized
2015-12-17 02:40:20,206 [main] INFO  org.apach
s : 1
2015-12-17 02:40:20,206 [main] INFO  org.apach
o process : 1
(AGS,2315,59.380561555075595)
(ACY,761,51.482260183968464)
(AVL,3101,49.905836826830054)
(ILM,3433,47.63384794640256)
(ROA,4172,45.672099712368166)
(VLD,1134,44.516754850088184)
(ABY,1363,41.93983859134263)
(HKY,41,38.390243902439025)
(LYH,1022,36.40117416829746)
(MCN,1076,31.90799256505576)
(PHF,6049,29.8026120019838)
(SYR,12383,28.322781232334652)
(FAY,1652,27.414043583535108)
(CHO,2025,27.08543209876543)
(BQK,1044,22.916666666666668)
(ISO,706,22.322946175637394)
(MDT,7885,22.107799619530756)
(TRI,1458,21.488340192043896)
(ROC,17718,20.427474884298455)
(CAE,11450,18.469519650655023)
grunt>
```

- Use pig script to implement the top 20 the average number of Taxi in and the amount of flights of a destination
- The data store in the HDFS
- A1 = LOAD '/data/2006a.csv' USING PigStorage(',') AS (Year:chararray,Month:int,DayofMonth:int,DayOfWeek:int,DepTime:chararray,CRSDepTime:chararray,ArrTime:chararray,CRSArrTime:chararray,UniqueCarrier:chararray,FlightNum:chararray,TailNum:chararray,ActualElapsedTime:int,CRSElapsedTime:int,AirTime:int,ArrDelay:int,DepDelay:int,Origin:chararray,Dest:chararray,Distance:int,TaxiIn:int,TaxiOut:int,Cancelled:chararray,CancellationCode:chararray,Diverted:int,CarrierDelay:int,WeatherDelay:int,NASDelay:int,SecurityDelay:int,LateAircraftDelay:int);
- Used the pig UDF to realize counting the amount of a column.
- REGISTER '/pig/PigUDF.jar’

The use of Hbase



```
t3
test06
4 row(s) in 2.6710 seconds

=> ["2006a", "t1", "t3", "test06"]
hbase(main):002:0> count '2006a'
Current count: 1000, row: row1000896
Current count: 2000, row: row1001796
Current count: 3000, row: row1002696
Current count: 4000, row: row1003596
Current count: 5000, row: row1004496
Current count: 6000, row: row1005396
Current count: 7000, row: row1006296
Current count: 8000, row: row1007196
Current count: 9000, row: row1008096
Current count: 10000, row: row1008997
Current count: 11000, row: row1009897
Current count: 12000, row: row1010796
Current count: 13000, row: row1011696
Current count: 14000, row: row1012596
Current count: 15000, row: row1013496
Current count: 16000, row: row1014396
Current count: 17000, row: row1015296
Current count: 18000, row: row1016196
```



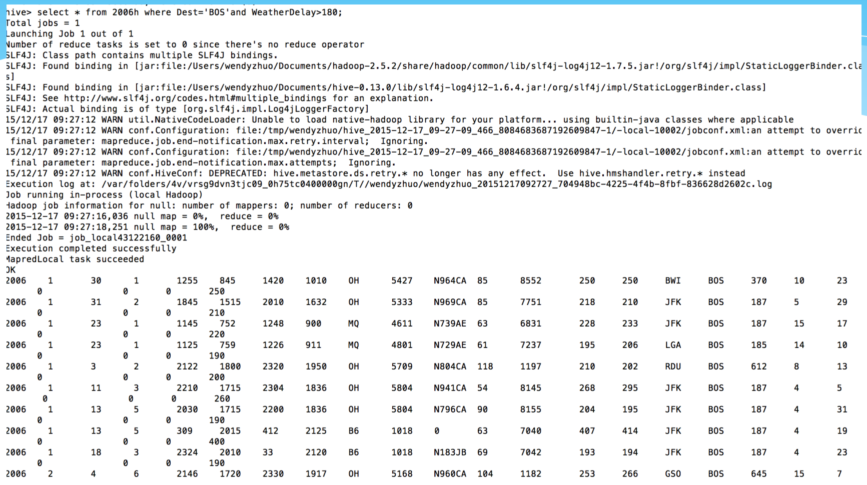
```
Current count: 7140000, row: row9982
Current count: 7141000, row: row9991
7141922 row(s) in 342.0790 seconds

=> 7141922
hbase(main):003:0> █
```

* Use pig script load data to Hbase
* STORE raw_data INTO 'hbase://2006a' USING org.apache.pig.backend.hadoop.hbase.HBaseStorage('t_data:Year,t_data:Month,t_data:DayofMonth,t_data:DayOfWeek,t_data:DepTime,t_data:CRSDepTime, t_data:ArrTime,t_data:CRSArrTime,t_data:UniqueCarrier,t_data:FlightNum,t_data:TailNum,t_data:ActualElapsedTime,t_data:CRSElapsedTime,t_data:AirTime,t_data:ArrDelay,t_data:DepDelay,t_data:Origin,t_data:Dest,t_data:Distance,t_data:TaxiIn, t_data:TaxiOut,t_data:Cancelled,t_data:CancellationCode,t_data:Diverted,t_data:CarrierDelay,t_data:WeatherDelay,t_data:NASDelay,t_data:SecurityDelay,t_data:LateAircraftDelay')
* We can also use the java api to load data to the Hbase

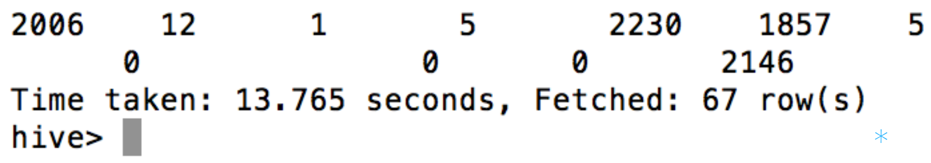
HBaseIntegration

* CREATE TABLE t5(key int,Year STRING,Month INT,DayofMonth INT,DayOfWeek INT,DepTime STRING,CRSDepTime STRING,ArrTime STRING,CRSArrTime STRING,UniqueCarrier STRING,FlightNum STRING,TailNum STRING,ActualElapsedTime INT,CRSElapsedTime INT,AirTime INT,ArrDelay INT,DepDelay INT,Origin STRING,Dest STRING,Distance INT,TaxiIn INT,TaxiOut INT,Cancelled STRING,CancellationCode STRING,Diverted INT,CarrierDelay INT,WeatherDelay INT,NASDelay INT,SecurityDelay INT,LateAircraftDelay INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’ STORED BY‘org.apache.hadoop.hive.hbase.HBaseStorageHandler’ WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key, a:Year, a:Month , a:DayofMonth , a:DayOfWeek , a:DepTime , a:CRSDepTime , a:ArrTime , a:CRSArrTime , a:UniqueCarrier , a:FlightNum , a:TailNum , a:ActualElapsedTime , a:CRSElapsedTime , a:AirTime , a:ArrDelay , a:DepDelay , a:Origin , a:Dest , a:Distance , a:TaxiIn , a:TaxiOut , a:Cancelled , a:CancellationCode , a:Diverted , a:CarrierDelay , a:WeatherDelay , a:NASDelay , a:SecurityDelay , a:LateAircraftDelay") TBLPROPERTIES ("hbase.table.name" = "2006a");

Analysis with Hive(1)



```
hive> select * from 2006h where Dest='BOS'and WeatherDelay>180;
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/Users/wendyzhuo/Documents/hadoop-2.5.2/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.cla
s]
SLF4J: Found binding in [jar:file:/Users/wendyzhuo/Documents/hive-0.13.0/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
15/12/17 09:27:12 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
15/12/17 09:27:12 WARN conf.Configuration: file:/tmp/wendyzhuo/hive_2015-12-17_09-27-09_466_8084683687192609847-1/-local-10002/jobconf.xml:an attempt to overric
 final parameter: mapreduce.job.end-notification.max.retry.interval;  Ignoring.
15/12/17 09:27:12 WARN conf.Configuration: file:/tmp/wendyzhuo/hive_2015-12-17_09-27-09_466_8084683687192609847-1/-local-10002/jobconf.xml:an attempt to overric
 final parameter: mapreduce.job.end-notification.max.attempts;  Ignoring.
15/12/17 09:27:12 WARN conf.HiveConf: DEPRECATED: hive.metastore.ds.retry.* no longer has any effect.  Use hive.hmshandler.retry.* instead
Execution log at: /var/folders/4v/vrsg9dvn3tjc09_0h75tc0400000gn/T//wendyzhuo/wendyzhuo_20151217092727_704948bc-4225-4f4b-8fbf-836628d2602c.log
Job running in-process (local Hadoop)
Hadoop job information for null: number of mappers: 0; number of reducers: 0
2015-12-17 09:27:16,036 null map = 0%,  reduce = 0%
2015-12-17 09:27:18,251 null map = 100%,  reduce = 0%
Ended Job = job_local43122160_0001
Execution completed successfully
MapredLocal task succeeded
OK
2006    1       30      1       1255    845     1420    1010    OH      5427    N964CA  85      8552    250     250     BWI     BOS     370     10      23
    0               0       0       250
2006    1       31      2       1845    1515    2010    1632    OH      5333    N969CA  85      7751    218     210     JFK     BOS     187     5       29
    0               0       0       210
2006    1       23      1       1145    752     1248    900     MQ      4611    N739AE  63      6831    228     233     JFK     BOS     187     15      17
    0               0       0       220
2006    1       23      1       1125    759     1226    911     MQ      4801    N729AE  61      7237    195     206     LGA     BOS     185     14      10
    0               0       0       190
2006    1       3       2       2122    1800    2320    1950    OH      5709    N804CA  118     1197    210     202     RDU     BOS     612     8       13
    0               0       0       200
2006    1       11      3       2210    1715    2304    1836    OH      5804    N941CA  54      8145    268     295     JFK     BOS     187     4       5
     0               0       0       260
2006    1       13      5       2030    1715    2200    1836    OH      5804    N796CA  90      8155    204     195     JFK     BOS     187     4       31
    0               0       0       190
2006    1       13      5       309     2015    412     2125    B6      1018    0       63      7040    407     414     JFK     BOS     187     4       19
    0               0       0       400
2006    1       18      3       2324    2010    33      2120    B6      1018    N183JB  69      7042    193     194     JFK     BOS     187     4       23
    0               0       0       190
2006    2       4       6       2146    1720    2330    1917    OH      5168    N960CA  104     1182    253     266     GSO     BOS     645     15      7
```

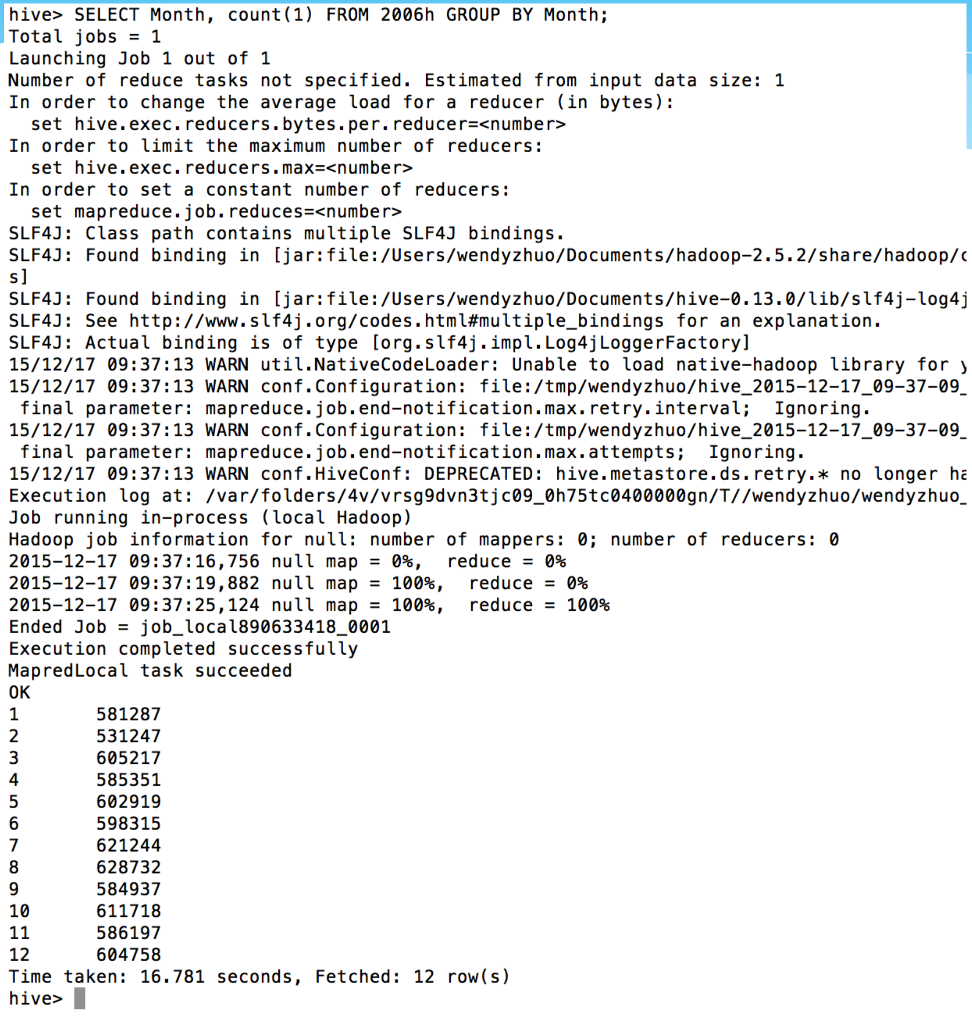


```
2006    12      1       5       2230    1857    5
        0               0       0       2146
Time taken: 13.765 seconds, Fetched: 67 row(s)
hive> █
```

- Except using the HBaseIntegration to share data with hbase, we can easily load data from hdfs into Hive in seconds.
- The query:
- CREATE EXTERNAL TABLE 2006h (Year STRING,Month INT,DayofMonth INT,DayOfWeek INT,DepTime STRING,CRSDepTime STRING,ArrTime STRING,CRSArrTime STRING,UniqueCarrier STRING,FlightNum STRING,TailNum STRING,ActualElapsedTime INT,CRSElapsedTime INT,AirTime INT,ArrDelay INT,DepDelay INT,Origin STRING,Dest STRING,Distance INT,TaxiIn INT,TaxiOut INT,Cancelled STRING,CancellationCode STRING,Diverted INT,CarrierDelay INT,WeatherDelay INT,NASDelay INT,SecurityDelay INT,LateAircraftDelay INT)ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION '/data/2006a.csv';

- Hive can query with SQL-style language.
- select * from 2006h where Dest='BOS'and WeatherDelay>180;

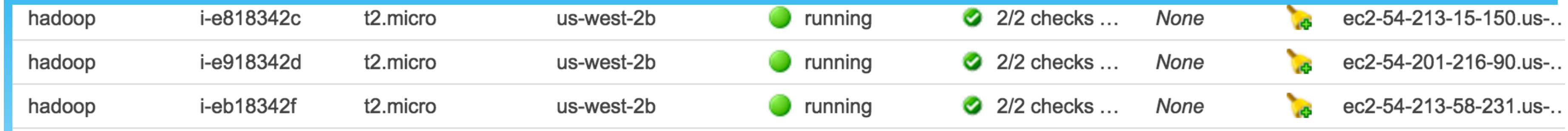
Analysis with Hive(2)



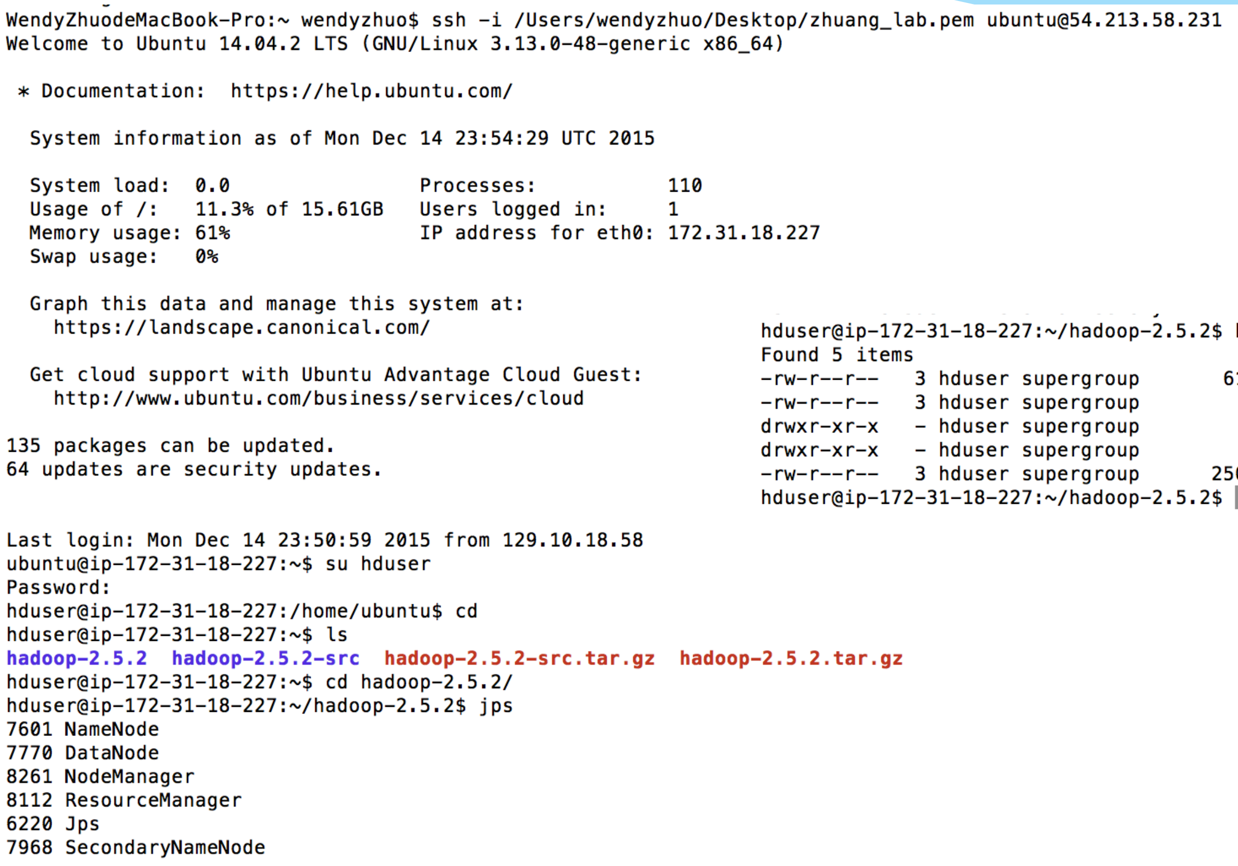
```
hive> SELECT Month, count(1) FROM 2006h GROUP BY Month;
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit 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>
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/Users/wendyzhuo/Documents/hadoop-2.5.2/share/hadoop/c
s]
SLF4J: Found binding in [jar:file:/Users/wendyzhuo/Documents/hive-0.13.0/lib/slf4j-log4j
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
15/12/17 09:37:13 WARN util.NativeCodeLoader: Unable to load native-hadoop library for y
15/12/17 09:37:13 WARN conf.Configuration: file:/tmp/wendyzhuo/hive_2015-12-17_09-37-09_
 final parameter: mapreduce.job.end-notification.max.retry.interval;  Ignoring.
15/12/17 09:37:13 WARN conf.Configuration: file:/tmp/wendyzhuo/hive_2015-12-17_09-37-09_
 final parameter: mapreduce.job.end-notification.max.attempts;  Ignoring.
15/12/17 09:37:13 WARN conf.HiveConf: DEPRECATED: hive.metastore.ds.retry.* no longer ha
Execution log at: /var/folders/4v/vrsg9dvn3tjc09_0h75tc0400000gn/T//wendyzhuo/wendyzhuo_
Job running in-process (local Hadoop)
Hadoop job information for null: number of mappers: 0; number of reducers: 0
2015-12-17 09:37:16,756 null map = 0%,  reduce = 0%
2015-12-17 09:37:19,882 null map = 100%,  reduce = 0%
2015-12-17 09:37:25,124 null map = 100%,  reduce = 100%
Ended Job = job_local890633418_0001
Execution completed successfully
MapredLocal task succeeded
OK
1       581287
2       531247
3       605217
4       585351
5       602919
6       598315
7       621244
8       628732
9       584937
10      611718
11      586197
12      604758
Time taken: 16.781 seconds, Fetched: 12 row(s)
hive> 
```

- Hive can query with SQL-style language:
- SELECT Month, count(1) FROM 2006h GROUP BY Month;

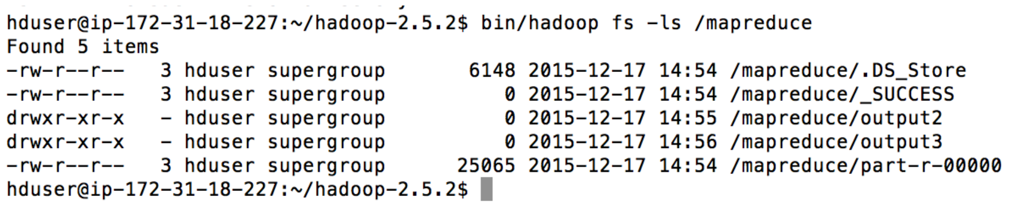
AWS



hadoop	i-e818342c	t2.micro	us-west-2b	● running	✓ 2/2 checks …	*None*		ec2-54-213-15-150.us-..
hadoop	i-e918342d	t2.micro	us-west-2b	● running	✓ 2/2 checks …	*None*		ec2-54-201-216-90.us-..
hadoop	i-eb18342f	t2.micro	us-west-2b	● running	✓ 2/2 checks …	*None*		ec2-54-213-58-231.us-..

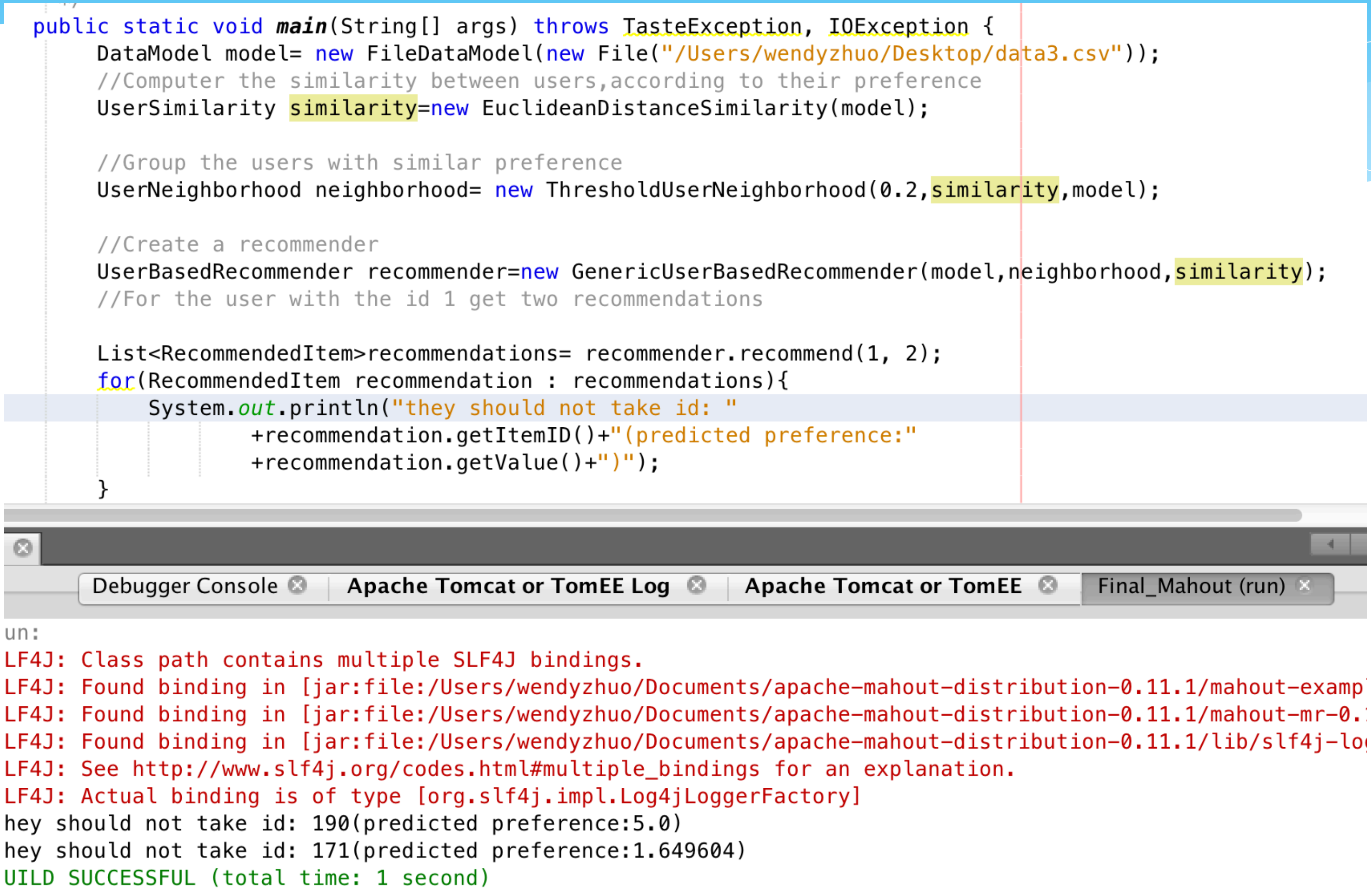


```
WendyZhuodeMacBook-Pro:~ wendyzhuo$ ssh -i /Users/wendyzhuo/Desktop/zhuang_lab.pem ubuntu@54.213.58.231
Welcome to Ubuntu 14.04.2 LTS (GNU/Linux 3.13.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Mon Dec 14 23:54:29 UTC 2015

  System load:  0.0               Processes:           110
  Usage of /:   11.3% of 15.61GB   Users logged in:     1
  Memory usage: 61%               IP address for eth0: 172.31.18.227
  Swap usage:   0%

  Graph this data and manage this system at:
    https://landscape.canonical.com/

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

135 packages can be updated.
64 updates are security updates.


Last login: Mon Dec 14 23:50:59 2015 from 129.10.18.58
ubuntu@ip-172-31-18-227:~$ su hduser
Password:
hduser@ip-172-31-18-227:/home/ubuntu$ cd
hduser@ip-172-31-18-227:~$ ls
hadoop-2.5.2  hadoop-2.5.2-src  hadoop-2.5.2-src.tar.gz  hadoop-2.5.2.tar.gz
hduser@ip-172-31-18-227:~$ cd hadoop-2.5.2/
hduser@ip-172-31-18-227:~/hadoop-2.5.2$ jps
7601 NameNode
7770 DataNode
8261 NodeManager
8112 ResourceManager
6220 Jps
7968 SecondaryNameNode
```



```
hduser@ip-172-31-18-227:~/hadoop-2.5.2$ bin/hadoop fs -ls /mapreduce
Found 5 items
-rw-r--r--   3 hduser supergroup       6148 2015-12-17 14:54 /mapreduce/.DS_Store
-rw-r--r--   3 hduser supergroup          0 2015-12-17 14:54 /mapreduce/_SUCCESS
drwxr-xr-x   - hduser supergroup          0 2015-12-17 14:55 /mapreduce/output2
drwxr-xr-x   - hduser supergroup          0 2015-12-17 14:56 /mapreduce/output3
-rw-r--r--   3 hduser supergroup      25065 2015-12-17 14:54 /mapreduce/part-r-00000
hduser@ip-172-31-18-227:~/hadoop-2.5.2$
```

Mahout Machine Learning



```
public static void main(String[] args) throws TasteException, IOException {
    DataModel model= new FileDataModel(new File("/Users/wendyzhuo/Desktop/data3.csv"));
    //Computer the similarity between users,according to their preference
    UserSimilarity similarity=new EuclideanDistanceSimilarity(model);

    //Group the users with similar preference
    UserNeighborhood neighborhood= new ThresholdUserNeighborhood(0.2,similarity,model);

    //Create a recommender
    UserBasedRecommender recommender=new GenericUserBasedRecommender(model,neighborhood,similarity);
    //For the user with the id 1 get two recommendations

    List<RecommendedItem>recommendations= recommender.recommend(1, 2);
    for(RecommendedItem recommendation : recommendations){
        System.out.println("they should not take id: "
                +recommendation.getItemID()+"(predicted preference:"
                +recommendation.getValue()+")");
    }
```

```
un:
LF4J: Class path contains multiple SLF4J bindings.
LF4J: Found binding in [jar:file:/Users/wendyzhuo/Documents/apache-mahout-distribution-0.11.1/mahout-examp
LF4J: Found binding in [jar:file:/Users/wendyzhuo/Documents/apache-mahout-distribution-0.11.1/mahout-mr-0.
LF4J: Found binding in [jar:file:/Users/wendyzhuo/Documents/apache-mahout-distribution-0.11.1/lib/slf4j-lo
LF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
LF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
hey should not take id: 190(predicted preference:5.0)
hey should not take id: 171(predicted preference:1.649604)
UILD SUCCESSFUL (total time: 1 second)
```

Mapreduce3.java

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.LinkedHashMap;
import java.util.Map;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.chain.ChainMapper;
import org.apache.hadoop.mapreduce.lib.chain.ChainReducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/*
 * 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.
 */

/**
 *
 * @author Zhuang Zhuo <zhuo.z@husky.neu.edu>
 */
public class MapReduce3 {

   /**
    * @param args the command line arguments
    */

   static class TempMapper extends  Mapper<LongWritable, Text, CompositeKey_wd, IntWritable> {
       CompositeKey_wd wd = new CompositeKey_wd();

       @Override
       public void map(LongWritable key, Text value, Context context)
               throws IOException, InterruptedException {
                              //System.out.println("Before Mapper: " + key + "." + value);
           String line = value.toString();

           try {
               String[] lineSplit = line.split(",");
               // String requestUrl = line.substring(0, 10);
                wd.setDayOfWeek(lineSplit[1]);
              wd.setDest(lineSplit[22]);
               String requestUrl = lineSplit[21];
              if(requestUrl.equals("1")){
                  context.write(wd, new IntWritable(1));
              }
             //  System.out.println("" + "After Mapper:" + new Text(requestUrl) + "," + new IntWritable(1));

           } // context.write(out,one);
           catch (java.lang.ArrayIndexOutOfBoundsException e) {
               //  context.getCounter(Counter.LINESKIP).increment(1);

           }

       }

   }
```

Mapreduce3.java

```
static class TempReducer extends Reducer<CompositeKey_wd, IntWritable, CompositeKey_wd, IntWritable> {
  @Override
  public void reduce(CompositeKey_wd key, Iterable<IntWritable> values,
      Context context) throws IOException, InterruptedException {

    // System.out.println("Before Reduce:" + key + ",");
    int count = 0;
    for (IntWritable v : values) {
      count = count + v.get();
    }
    try {
      context.write(key, new IntWritable(count));
    //   System.out.println( ""+ "After Reduce:" + key + "," + count);

    } catch (InterruptedException e) {
      e.printStackTrace();
    }
  }

}

static class TempMapper2 extends  Mapper<LongWritable, Text, IntWritable, CompositeKey_wd> {
    CompositeKey_wd wd = new CompositeKey_wd();
  @Override
  protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {


      String line = value.toString();

    try {
      String[] lineSplit = line.split("\t");
      // String requestUrl = line.substring(0, 10);
      String requestUrl = lineSplit[0];
        String[] lineSplit2 = requestUrl.split(",");
          wd.setDayOfWeek(lineSplit2[0]);
     wd.setDest(lineSplit2[1]);

       int val = Integer.parseInt(lineSplit[1]);

      context.write(new IntWritable(val),wd);
    } // context.write(out,one);
    catch (java.lang.ArrayIndexOutOfBoundsException e) {
      //  context.getCounter(Counter.LINESKIP).increment(1);

    }
  }

}
```

Mapreduce3.java

```
static class TempReduce2 extends  Reducer<IntWritable, CompositeKey_wd,Text, IntWritable> {
    ArrayList<LinkedHashMap<String,Integer>> tm = new ArrayList<LinkedHashMap<String,Integer>>();
        int count=0;
     @Override
     protected void reduce(IntWritable key,Iterable<CompositeKey_wd> values, Context context) throws IOException, InterruptedException {
      for(CompositeKey_wd v :values){
      int a = Integer.parseInt(v.getDayOfWeek())-1;
    if(tm.size()<7){
       for(int i=0;i<12; i++){

        LinkedHashMap<String,Integer> f  = new LinkedHashMap<>();
        tm.add(f);
      }

    }
      LinkedHashMap<String,Integer> fin = tm.get(a);
        fin.put(v.toString(), key.get());
System.out.println("" + "reduce2:" + a + " || " +v.toString() + "| "+ fin.size()+" |"+key.get());
    // context.write(result, key);
      }
     }

     @Override
                       protected void cleanup(Context context) throws IOException,
                                                                    InterruptedException {

            for(LinkedHashMap<String,Integer> f: tm){
              int i=0;
              for (Map.Entry<String,Integer> entry : f.entrySet()) {
                 i++;
                 if(i==f.size()){
                    Text fi = new Text(entry.getKey());
                     context.write(fi,new IntWritable(entry.getValue()));
                 }
              }
            }


         }
  }
```

Mapreduce3.java

```
        public static void main(String[] args) throws Exception {
                        String dst = "hdfs://localhost:9000/data/2006a.csv";

//  String dstOut = "hdfs://localhost:9000/mapreduce/result3/1";
         String dstOut = "hdfs://localhost:9000/mapreduce/output3/1";
        String outFiles = "hdfs://localhost:9000/mapreduce/output3/2";
        Configuration hadoopConfig = new Configuration();

        hadoopConfig.set("fs.hdfs.impl",
            org.apache.hadoop.hdfs.DistributedFileSystem.class.getName()
        );

        hadoopConfig.set("fs.file.impl",
            org.apache.hadoop.fs.LocalFileSystem.class.getName()
        );

        Job job = new Job(hadoopConfig);
        Job job2 = new Job(hadoopConfig);

        FileInputFormat.addInputPath(job, new Path(dst));
        FileOutputFormat.setOutputPath(job, new Path(dstOut));
        FileInputFormat.addInputPath(job2, new Path(dstOut));
        FileOutputFormat.setOutputPath(job2, new Path(outFiles));
        JobConf map1Conf = new JobConf(false);
        ChainMapper.addMapper(job,TempMapper.class,LongWritable.class,Text.class,CompositeKey_wd.class,IntWritable.class,map1Conf);
        JobConf reduceConf = new JobConf(false);
         ChainReducer.setReducer(job,TempReducer.class,CompositeKey_wd.class,IntWritable.class,CompositeKey_wd.class,IntWritable.class,reduceConf);

         JobConf map2Conf = new JobConf(false);
        ChainMapper.addMapper(job2,TempMapper2.class,LongWritable.class,Text.class,IntWritable.class,CompositeKey_wd.class,map2Conf);
       JobConf map3Conf = new JobConf(false);
        ChainReducer.setReducer(job2,TempReduce2.class,IntWritable.class,CompositeKey_wd.class,Text.class,IntWritable.class,map3Conf);
//
     // JobClient.runJob(job);

        //指定自定义的Mapper和Reducer作为两个阶段的任务处理类
//      job.setMapperClass(TempMapper.class);
//
//      job.setReducerClass(TempReducer.class);
job.setOutputKeyClass(CompositeKey_wd.class);

        job.setOutputValueClass(IntWritable.class);

        job2.setMapOutputKeyClass(IntWritable.class);
     job2.setMapOutputValueClass(CompositeKey_wd.class);

 // job2.setSortComparatorClass(LongWritable.DecreasingComparator.class);
job.waitForCompletion(true);
        System.out.println("Finished1");
    job2.waitForCompletion(true);
        System.out.println("Finished2");

    }

}
```

CompositeKey_wd.java

```
import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import org.apache.hadoop.io.WritableComparable;

/**
 *
 * @author Zhuang Zhuo <zhuo.z@husky.neu.edu>
 */
public class CompositeKey_wd implements WritableComparable<CompositeKey_wd>{

    private String dayOfWeek ;
    private String dest ;

    public CompositeKey_wd() {
    }

    public CompositeKey_wd(String week, String dest) {
        this.dayOfWeek = week;
        this.dest = dest;
    }

    @Override
    public String toString() {
        return (new StringBuilder()).append(dayOfWeek).append(", ").append(dest).toString();
    }
    @Override
    public void readFields(DataInput in) throws IOException {
        dayOfWeek = in.readUTF();
        dest = in.readUTF();
    }

    @Override
    public void write(DataOutput out) throws IOException {
        out.writeUTF(dayOfWeek);
        out.writeUTF(dest);
    }

    @Override
    public int compareTo(CompositeKey_wd o) {
        int result = dayOfWeek.compareTo(o.dayOfWeek);
        if (0 == result) {
            result = dest.compareTo(o.dest);
        }
        return result;
    }

    public String getDayOfWeek() {
        return dayOfWeek;
    }

    public void setDayOfWeek(String dayOfWeek) {
        this.dayOfWeek = dayOfWeek;
    }

    public String getDest() {
        return dest;
    }

    public void setDest(String dest) {
        this.dest = dest;
    }

}
```

CompositeKey_mc.java

```
import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import org.apache.hadoop.io.WritableComparable;

/**
 *
 * @author Zhuang Zhuo <zhuo.z@husky.neu.edu>
 */
public class CompositeKey_mc implements WritableComparable<CompositeKey_mc>{

     private int dayOfWeek ;
     private int dest ;

    public CompositeKey_mc() {
    }

    public CompositeKey_mc(int week, int dest) {
        this.dayOfWeek = week;
        this.dest = dest;
    }

    @Override
    public String toString() {
        return (new StringBuilder()).append(dayOfWeek).append(", ").append(dest).toString();
    }

    @Override
    public void readFields(DataInput in) throws IOException {
        dayOfWeek = in.readInt();
        dest = in.readInt();
    }

    @Override
    public void write(DataOutput out) throws IOException {
        out.writeInt(dayOfWeek);
        out.writeInt(dest);
    }
    @Override
    public int compareTo(CompositeKey_mc o) {
        int result =  -(new Integer(dayOfWeek)).compareTo(o.dayOfWeek);
        if (o == result) {
            result =-(new Integer(dest)).compareTo(o.dest);
        }
        return result;
    }

    public int getDayOfWeek() {
        return dayOfWeek;
    }

    public void setDayOfWeek(int dayOfWeek) {
        this.dayOfWeek = dayOfWeek;
    }

    public int getDest() {
        return dest;
    }

    public void setDest(int dest) {
        this.dest = dest;
    }

}
```

Mapreduce1.java

```
import com.sun.jersey.core.header.InBoundHeaders;
import java.io.IOException;
import java.util.HashMap;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

/*
 * 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.
 */

/**
 *
 * @author Zhuang Zhuo <zhuo.z@husky.neu.edu>
 */
public class Mapreduce1 {

    /**
     * @param args the command line arguments
     */



    static class TempMapper extends Mapper<LongWritable, Text, CompositeKey_wd, IntWritable> {
        CompositeKey_wd wd = new CompositeKey_wd();
     @Override
        public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
            try {
               //  String[] lineSplit = line.split(" ");
               String []words=value.toString().split(",");
               // requestUrl = requestUrl.substring(0,requestUrl.indexOf(' ')+1);
                // Text out = new Text(requestUrl);
               int a = Integer.parseInt(words[27]);
              if(a>0){
               wd.setDayOfWeek(words[3]);
               wd.setDest(words[17]);
              //  System.out.println("After Mapper:"+ wd.getDayOfWeek() + "," + wd.getDest()+" | "+new IntWritable(1));
                context.write(wd, new IntWritable(a));
              }
            } // context.write(out,one);
            catch (java.lang.ArrayIndexOutOfBoundsException e) {
                //  context.getCounter(Counter.LINESKIP).increment(1);

            }

        }

    }
```

Mapreduce1.java

```
static class TempReducer extends Reducer<CompositeKey_wd, IntWritable, Text, IntWritable> {
    HashMap<String,Integer> f = new HashMap<String, Integer>();
    int i=0;
        @Override
        public void reduce(CompositeKey_wd key, Iterable<IntWritable> values,Context context) throws IOException, InterruptedException {

          // System.out.println("Before Reduce:" + key + ",");
             int count = 0;
             for (IntWritable v : values) {
                 count = count + v.get();
             }
             try {
               if(f.containsKey(key.getDest())){
                    Text t = new Text(key.getDayOfWeek()+","+f.get(key.getDest())+",");

              double a = count*0.01;
                    if(a<5){
                    context.write(t, new DoubleWritable(a));}
                    else{
                        context.write(t, new DoubleWritable(5));
                    }
                }
                else{
                   f.put(key.getDest(), i);
                   i++;
                    Text t = new Text(key.getDayOfWeek()+", "+f.get(key.getDest())+",");
                    double a = count*0.01;
                    if(a<5){
                    context.write(t, new DoubleWritable(a));}              }

             } catch (InterruptedException e) {
                 e.printStackTrace();
             }
          }
        }

  public static void main(String[] args) throws Exception {
        //输入路径
         String dst = "hdfs://localhost:9000/data/2006a.csv";
        //输出路径，必须是不存在的，空文件加也不行。
         String dstOut = "hdfs://localhost:9000/Number6";
        String outFiles = "/Users/wendyzhuo/NetBeansProjects/final_Hadoop/src/output";
        Configuration hadoopConfig = new Configuration();

        hadoopConfig.set("fs.hdfs.impl",  org.apache.hadoop.hdfs.DistributedFileSystem.class.getName());

        hadoopConfig.set("fs.file.impl",org.apache.hadoop.fs.LocalFileSystem.class.getName());

        Job job = new Job(hadoopConfig);

        FileInputFormat.addInputPath(job, new Path(dst));
        //FileOutputFormat.setOutputPath(job, new Path(dstOut));
FileOutputFormat.setOutputPath(job, new Path(outFiles));
        //指定自定义的Mapper和Reducer作为两个阶段的任务处理类

         job.setMapperClass(TempMapper.class);

        job.setReducerClass(TempReducer.class);

job.setMapOutputKeyClass(CompositeKey_wd.class);
        job.setOutputKeyClass(Text.class);

        job.setOutputValueClass(IntWritable.class);
        //执行job，直到完成
         job.waitForCompletion(true);

        System.out.println("Finished");

    }

}
```

Mapreduce4.java

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.LinkedHashMap;
import java.util.Map;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.chain.ChainMapper;
import org.apache.hadoop.mapreduce.lib.chain.ChainReducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/*
 * 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.
 */

/**
 *
 * @author Zhuang Zhuo <zhuo.z@husky.neu.edu>
 */
public class MapReduce4 {

    /**
     * @param args the command line arguments
     */
    static class TempMapper extends  Mapper<LongWritable, Text, CompositeKey_wd,IntWritable > {
        CompositeKey_wd wd = new CompositeKey_wd();
        @Override
        protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

            String line = value.toString();

            try {
                String[] lineSplit = line.split(",");

                 wd.setDayOfWeek(lineSplit[16]);
               wd.setDest(lineSplit[17]);
               int a = Integer.parseInt(lineSplit[14]);
               int b = Integer.parseInt(lineSplit[15]);
              int c=a+b;
               context.write(wd,new IntWritable(c));

            }
            catch (java.lang.ArrayIndexOutOfBoundsException e) {

            }
    }
```

Mapreduce4.java

```
*      static class TempReducer extends Reducer<CompositeKey_wd, IntWritable, CompositeKey_wd, Text> {

*              @Override
*              protected void reduce(CompositeKey_wd key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {
*               int count = 0;
*                int count1 = 0;
*              for (IntWritable v : values) {
*                 count = count + v.get();
*                  count1++;
*              }
*              try {
*                 Text a = new Text(count + ","+count1);
*
*                 context.write(key,a);
*               //  System.out.println( ""+ "After Reduce1:" + key.toString() + "||" + count+", "+count1);

*              } catch (InterruptedException e) {
*                 e.printStackTrace();
*              }
*              }

*          }
*      static class TempMapper2 extends  Mapper<LongWritable, Text, CompositeKey_wd, CompositeKey_mc> {
*              CompositeKey_wd wd = new CompositeKey_wd();
*              CompositeKey_mc mc = new CompositeKey_mc();
*              @Override
*             protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
*                String line = value.toString();
*              try {
*                 String[] lineSplit = line.split("\t");
*                 String requestUrl = lineSplit[0];
*                 String requestUrl1 = lineSplit[1];
*                   String[] lineSplit2 = requestUrl.split(",");
*                     wd.setDayOfWeek(lineSplit2[0]);
*                wd.setDest(lineSplit2[1]);
*                 String[] lineSplit3 = requestUrl1.split(",");
*                 int a = Integer.parseInt(lineSplit3[0]);
*                int b = Integer.parseInt(lineSplit3[1]);
*                 mc.setDayOfWeek(a);
*                 mc.setDest(b);
*                 context.write(wd,mc);
*              } // context.write(out,one);
*              catch (java.lang.ArrayIndexOutOfBoundsException e) {
*              }
*              }
*
*          }
```

Mapreduce4.java

```
static class TempReduce2 extends  Reducer<CompositeKey_wd, CompositeKey_mc,CompositeKey_wd, IntWritable> {

      @Override
      protected void reduce(CompositeKey_wd key, Iterable<CompositeKey_mc> values, Context context) throws IOException, InterruptedException {
        int count = 0;

      for (CompositeKey_mc v : values) {
          count = v.getDayOfWeek()/v.getDest();
      }
      try {
       //  System.out.println( ""+ "After Reduce2:" + key.toString() + "," + count);
          context.write(key,new IntWritable(count));
      } catch (InterruptedException e) {
          e.printStackTrace();
      }
      }
  }

  static class TempMapper3 extends  Mapper<LongWritable, Text, IntWritable, Text> {
      CompositeKey_wd wd = new CompositeKey_wd();

      @Override
      protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
            String line = value.toString();

      try {
          String[] lineSplit = line.split("\t");
          // String requestUrl = line.substring(0, 10);
          String requestUrl = lineSplit[0];
            String[] lineSplit2 = requestUrl.split(",");
              wd.setDayOfWeek(lineSplit2[0]);
         wd.setDest(lineSplit2[1]);

           int val = Integer.parseInt(lineSplit[1]);
         //  System.out.println( ""+ "After Reduce2:" +val + "," + wd.toString());
          context.write(new IntWritable(val),new Text(wd.toString()));
      } // context.write(out,one);
      catch (java.lang.ArrayIndexOutOfBoundsException e) {
          //  context.getCounter(Counter.LINESKIP).increment(1);

      }

    }
  }
```

Mapreduce4.java

```
static class TempReduce3 extends  Reducer<IntWritable, Text, IntWritable,Text> {
     LinkedHashMap<String,Integer> tm = new LinkedHashMap<String,Integer>();
    int count=0;
      @Override
      public void reduce(IntWritable key,  Iterable<Text> values, Context context)
              throws IOException, InterruptedException {

              for (Text val : values) {
                  count++;
     String a = val.toString();
      int b =key.get();
       tm.put(a, b);
              }
      }
      @Override
                              protected void cleanup(Context context) throws IOException,
                                                                                                                InterruptedException {
                  ArrayList<LinkedHashMap<String,Integer>> a = new ArrayList();
          int i=0;
           for (Map.Entry<String,Integer> entry : tm.entrySet()) {
              i++;
              if(i>count-31){
                    LinkedHashMap<String,Integer> t = new LinkedHashMap<String,Integer>();
                    t.put(entry.getKey(), entry.getValue());
                   a.add(t);
              }

           }
           for(int j=a.size()-1;j>0;j--){
                    Text result = new Text();

                   for (Map.Entry<String,Integer> entry : a.get(j).entrySet()) {
                         result.set(entry.getKey());
                  context.write(new IntWritable(entry.getValue()),result);
              }

      }
              }
   }
public static void main(String[] args) throws Exception {

      //输入路径
       String dst = "hdfs://localhost:9000/data/2006a.csv";

      //输出路径，必须是不存在的，空文件加也不行。
     // String dstOut = "hdfs://localhost:9000/mapreduce/result3/1";
       String dstOut = "/Users/wendyzhuo/NetBeansProjects/final_Hadoop/src/output4/1";
      String outFiles = "/Users/wendyzhuo/NetBeansProjects/final_Hadoop/src/output4/2";
      String outFiles2 = "/Users/wendyzhuo/NetBeansProjects/final_Hadoop/src/output4/3";
      Configuration hadoopConfig = new Configuration();

      hadoopConfig.set("fs.hdfs.impl",
              org.apache.hadoop.hdfs.DistributedFileSystem.class.getName()
      );

      hadoopConfig.set("fs.file.impl",
              org.apache.hadoop.fs.LocalFileSystem.class.getName()
      );
```

Mapreduce4.java

```
      Job job = new Job(hadoopConfig);
          Job job2 = new Job(hadoopConfig);
          Job job3 = new Job(hadoopConfig);

          FileInputFormat.addInputPath(job, new Path(dst));
          FileOutputFormat.setOutputPath(job, new Path(dstOut));
          FileInputFormat.addInputPath(job2, new Path(dstOut));
          FileOutputFormat.setOutputPath(job2, new Path(outFiles));
           FileInputFormat.addInputPath(job3, new Path(outFiles));
          FileOutputFormat.setOutputPath(job3, new Path(outFiles2));
      JobConf map1Conf = new JobConf(false);
          ChainMapper.addMapper(job,TempMapper.class,LongWritable.class,Text.class,CompositeKey_wd.class,IntWritable.class,map1Conf);
          JobConf reduceConf = new JobConf(false);
           ChainReducer.setReducer(job,TempReducer.class,CompositeKey_wd.class,IntWritable.class,CompositeKey_wd.class,IntWritable.class,reduceConf);

           JobConf map2Conf = new JobConf(false);
          ChainMapper.addMapper(job2,TempMapper2.class,LongWritable.class,Text.class,CompositeKey_wd.class,CompositeKey_mc.class,map2Conf);
         JobConf map3Conf = new JobConf(false);
          ChainReducer.setReducer(job2,TempReduce2.class,CompositeKey_wd.class,CompositeKey_mc.class,CompositeKey_wd.class,IntWritable.class,map3Conf);

        JobConf maplConf = new JobConf(false);
          ChainMapper.addMapper(job3,TempMapper3.class,LongWritable.class,Text.class,IntWritable.class,Text.class,maplConf);
         JobConf mapllConf = new JobConf(false);
          ChainReducer.setReducer(job3,TempReduce3.class,IntWritable.class,Text.class,IntWritable.class,Text.class,mapllConf);

          job.setOutputKeyClass(CompositeKey_wd.class);

          job.setOutputValueClass(IntWritable.class);

          job2.setMapOutputKeyClass(CompositeKey_wd.class);
        job2.setMapOutputValueClass(CompositeKey_mc.class);

       job3.setMapOutputKeyClass(IntWritable.class);
        job3.setMapOutputValueClass(Text.class);
          job.waitForCompletion(true);
          System.out.println("Finished1");
       job2.waitForCompletion(true);
          System.out.println("Finished2");

          job3.waitForCompletion(true);
          System.out.println("Finished2");
       }


       }
    }
```

Pig.script

* A1 = LOAD '/data/2006a.csv' USING PigStorage(',') AS (Year:chararray,Month:int,DayofMonth:int,DayOfWeek:int,DepTime:chararray,CRSDepTime:chararray,ArrTime:chararray,CRSArrTime:chararray,UniqueCarrier:chararray,FlightNum:chararray,TailNum:chararray,ActualElapsedTime:int,CRSElapsedTime:int, AirTime:int,ArrDelay:int,DepDelay:int,Origin:chararray,Dest:chararray,Distance:int,TaxiIn:int,TaxiOut:int,Cancelled:chararray,CancellationCode:chararray,Diverted:int,CarrierDelay:int,WeatherDelay:int,NASDelay:int,SecurityDelay:int,LateAircraftDelay:int);
* A = FOREACH A1 GENERATE DayofMonth, Dest,TaxiIn;
* B = GROUP A BY Dest;
* C = FOREACH B GENERATE group AS Dest,
* AVG(A.TaxiIn) AS avgIn;
* REGISTER '/pig/PigUDF.jar'
* E = JOIN C BY Dest, A1.pigCount(Dest) BY Dest;
* report = FOREACH E GENERATE C::Dest,count,avgIn;
* report = ORDER report BY avgIn DESC, count DESC;
* top20 = LIMIT report 20;

pigUDF

```
/*
 * 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 movies;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

import org.apache.pig.Accumulator;
import org.apache.pig.Algebraic;
import org.apache.pig.EvalFunc;
import org.apache.pig.FuncSpec;
import org.apache.pig.PigException;
import org.apache.pig.backend.executionengine.ExecException;
import org.apache.pig.data.DataBag;
import org.apache.pig.data.DataType;
import org.apache.pig.data.Tuple;
import org.apache.pig.data.TupleFactory;
import org.apache.pig.impl.logicalLayer.FrontendException;
import org.apache.pig.impl.logicalLayer.schema.Schema;

/**
 *
 * @author Zhuang Zhuo <zhuo.z@husky.neu.edu>
 */

public class pigCount extends EvalFunc<Long> implements Algebraic, Accumulator<Long>{
  private static TupleFactory mTupleFactory = TupleFactory.getInstance();

  @Override
  public Long exec(Tuple input) throws IOException {
    try {
      DataBag bag = (DataBag)input.get(0);
      if(bag==null)
        return null;

      Iterator it = bag.iterator();
      long cnt = 0;
      while (it.hasNext()){
          Tuple t = (Tuple)it.next();
          if (t != null && t.size() > 0 && t.get(0) != null )
              cnt++;
      }
      return cnt;
    } catch (ExecException ee) {
      throw ee;
    } catch (Exception e) {
      int errCode = 2106;
      String msg = "Error while computing count in " + this.getClass().getSimpleName();
      throw new ExecException(msg, errCode, PigException.BUG, e);
    }
  }

  public String getInitial() {
    return Initial.class.getName();
  }

  public String getIntermed() {
    return Intermediate.class.getName();
  }
```

pigUDF

```
    public String getFinal() {
        return Final.class.getName();
    }

    static public class Initial extends EvalFunc<Tuple> {

        @Override
        public Tuple exec(Tuple input) throws IOException {
            // Since Initial is guaranteed to be called
            // only in the map, it will be called with an
            // input of a bag with a single tuple - the
            // count should always be 1 if bag is non empty
            DataBag bag = (DataBag)input.get(0);
            Iterator it = bag.iterator();
            if (it.hasNext()){
                Tuple t = (Tuple)it.next();
                if (t != null && t.size() > 0 && t.get(0) != null)
                    return mTupleFactory.newTuple(Long.valueOf(1));
            }
            return mTupleFactory.newTuple(Long.valueOf(0));
        }
    }

    static public class Intermediate extends EvalFunc<Tuple> {

        @Override
        public Tuple exec(Tuple input) throws IOException {
            try {
                return mTupleFactory.newTuple(sum(input));
            } catch (ExecException ee) {
                throw ee;
            } catch (Exception e) {
                int errCode = 2106;
                String msg = "Error while computing count in " + this.getClass().getSimpleName();
                throw new ExecException(msg, errCode, PigException.BUG, e);
            }
        }
    }

    static public class Final extends EvalFunc<Long> {
        @Override
        public Long exec(Tuple input) throws IOException {
            try {
                return sum(input);
            } catch (Exception ee) {
                int errCode = 2106;
                String msg = "Error while computing count in " + this.getClass().getSimpleName();
                throw new ExecException(msg, errCode, PigException.BUG, ee);
            }
        }
    }

    static protected Long sum(Tuple input) throws ExecException, NumberFormatException {
        DataBag values = (DataBag)input.get(0);
        long sum = 0;
        for (Iterator<Tuple> it = values.iterator(); it.hasNext();) {
            Tuple t = it.next();
            sum += (Long)t.get(0);
        }
        return sum;
    }
```

pigUDF

```
    @Override
    public Schema outputSchema(Schema input) {
        return new Schema(new Schema.FieldSchema(null, DataType.LONG));
    }


    @Override
    public List<FuncSpec> getArgToFuncMapping() throws FrontendException {
        List<FuncSpec> funcList = new ArrayList<FuncSpec>();
        Schema s = new Schema();
        s.add(new Schema.FieldSchema(null, DataType.BAG));
        funcList.add(new FuncSpec(this.getClass().getName(), s));
        return funcList;
    }

    /* Accumulator interface implementation */
    private long intermediateCount = 0L;

    @Override
    public void accumulate(Tuple b) throws IOException {
        try {
            DataBag bag = (DataBag)b.get(0);
            Iterator it = bag.iterator();
            while (it.hasNext()){
                Tuple t = (Tuple)it.next();
                if (t != null && t.size() > 0 && t.get(0) != null) {
                    intermediateCount += 1;
                }
            }
        } catch (ExecException ee) {
            throw ee;
        } catch (Exception e) {
            int errCode = 2106;
            String msg = "Error while computing min in " + this.getClass().getSimpleName();
            throw new ExecException(msg, errCode, PigException.BUG, e);
        }
    }

    @Override
    public void cleanup() {
        intermediateCount = 0L;
    }

    @Override
    public Long getValue() {
        return intermediateCount;
    }

}
```

Hbase

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.HColumnDescriptor;
import org.apache.hadoop.hbase.HTableDescriptor;
import org.apache.hadoop.hbase.TableName;
import org.apache.hadoop.hbase.client.HBaseAdmin;
import org.apache.hadoop.hbase.client.HTable;
import org.apache.hadoop.hbase.client.Put;
import org.apache.hadoop.hbase.util.Bytes;

/**
 *
 * @author xuefanrong
 */
public class Final_Hbase {
   public static void main(String[] args) throws IOException {

     Configuration con = HBaseConfiguration.create();
     HBaseAdmin admin = new HBaseAdmin(con);
     HTableDescriptor tableDescriptor = new HTableDescriptor(TableName.valueOf("2006a"));
     tableDescriptor.addFamily(new HColumnDescriptor("a_data"));
     admin.createTable(tableDescriptor);
     System.out.println(" Table created ");
     Configuration config = HBaseConfiguration.create();
     HTable hTable = new HTable(config, "2006a");

             Path pt=new Path("hdfs://localhost:9000/data/2006a.csv");
 Configuration confi = new Configuration();

          FileSystem fs = FileSystem.get(confi);
                    BufferedReader b=new BufferedReader(new InputStreamReader(fs.open(pt)));
     String test = null;
     int count = 0;
     while((test = b.readLine())!=null){
        count++;
     }
```

hbase

```
BufferedReader br=new BufferedReader(new InputStreamReader(fs.open(pt)));
    for(int i = 1;i<=count;i++){
      Put p = new Put(Bytes.toBytes("row"+i));
    String s=null;
    if((s=br.readLine())!=""){
      String[] insert = s.split(",");
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("Year"), Bytes.toBytes(insert[0]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("Month"), Bytes.toBytes(insert[1]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("DayofMonth"), Bytes.toBytes(insert[2]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("DayOfWeek"), Bytes.toBytes(insert[3]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("DepTime"), Bytes.toBytes(insert[4]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("CRSDepTime"), Bytes.toBytes(insert[5]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("ArrTime"), Bytes.toBytes(insert[6]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("CRSArrTime"), Bytes.toBytes(insert[7]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("UniqueCarrier"), Bytes.toBytes(insert[8]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("FlightNum"), Bytes.toBytes(insert[9]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("TailNum"), Bytes.toBytes(insert[10]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("ActualElapsedTime"), Bytes.toBytes(insert[11]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("CRSElapsedTime"), Bytes.toBytes(insert[12]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("Airtime"), Bytes.toBytes(insert[13]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("ArrDelay"), Bytes.toBytes(insert[14]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("DepDelay"), Bytes.toBytes(insert[15]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("Origin"), Bytes.toBytes(insert[16]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("Dest"), Bytes.toBytes(insert[17]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("Disntance"), Bytes.toBytes(insert[18]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("TaxiIn"), Bytes.toBytes(insert[19]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("TaxiOut"), Bytes.toBytes(insert[20]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("Cancelled"), Bytes.toBytes(insert[21]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("CancellationCode"), Bytes.toBytes(insert[22]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("Diverted"), Bytes.toBytes(insert[23]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("CarrierDelay"), Bytes.toBytes(insert[24]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("WeatherDelay"), Bytes.toBytes(insert[25]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("NASDelay"), Bytes.toBytes(insert[26]));
            p.add(Bytes.toBytes("a_data"), Bytes.toBytes("SecurityDelay"), Bytes.toBytes(insert[27]));
            p.add(Bytes.toBytes("a_data"),Bytes.toBytes("LateAircraftDelay"),Bytes.toBytes(insert[28]));
            // p.add(Bytes.toBytes("a_data"),Bytes.toBytes("LateAircraftDelay"),Bytes.toBytes(insert[29]));
      // Saving the put Instance to the HTable.
      hTable.put(p);
    }
    }

    // closing HTable
    hTable.close();

    }

  }
```