ProjectReport

by Sumedh Saraf

Submission date: 26-Apr-2017 11:56AM (UTC-0400)
Submission ID: 805274192
File name: ProjectReport.docx (512.28K)
Word count: 2747
Character count: 23427

Dataset Link: http://stat-computing.org/dataexpo/2009/the-data.html

Year chosen for data download-2008. Dataset contains information pertaining to each flight trip.

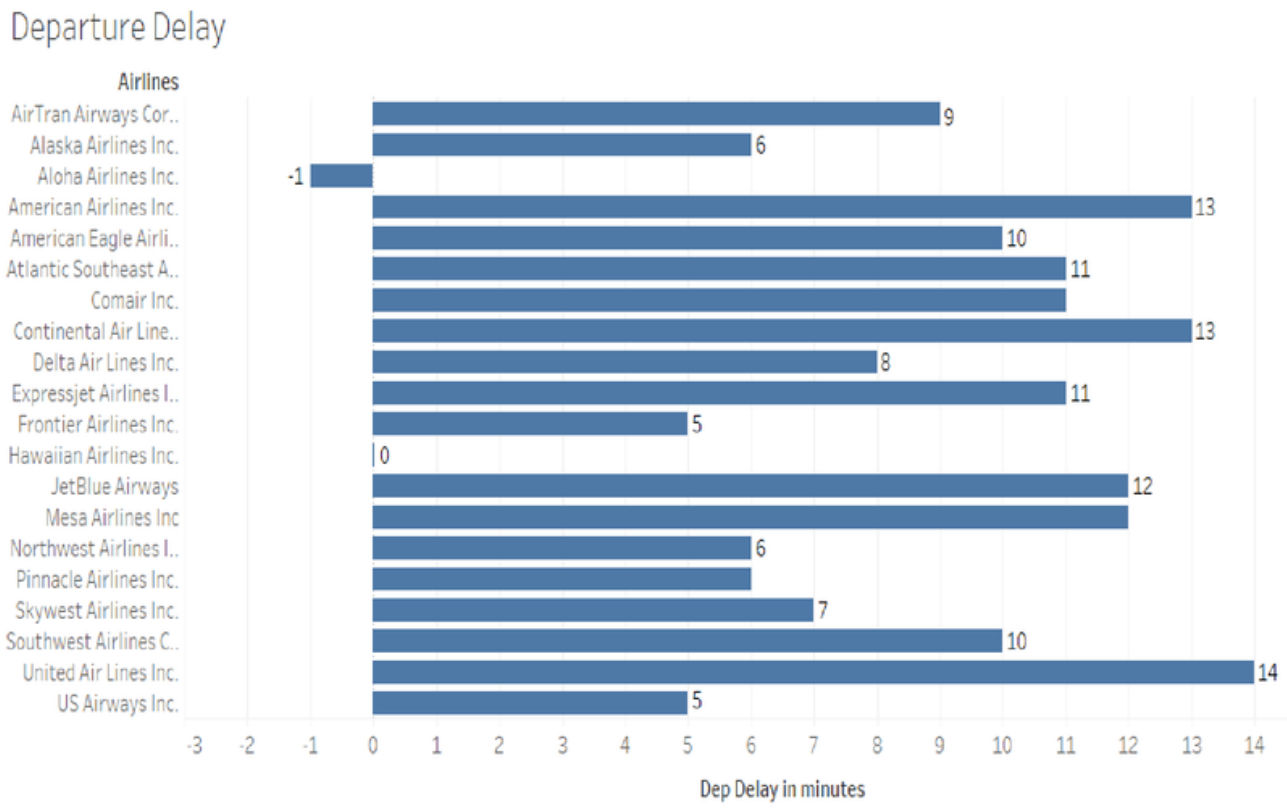
The project contains 7 analysis, which try to answer question stated at the beginning of each analysis.

Analysis 1

Which airlines have worst and best departure delays?

Finding the Departure Delay for each airline company for the year 2008

- This was done using Map reduce framework by finding average departure delay for each airline company.
- Custom Writable class is used to achieve Combiner Optimization.
- Inner join is used to join the airline company code to airline company name.
- Both the above tasks are done in one go using map reduce chaining.
- TextInputFormat and MultipleInputFiles are used for reading input files.
- Visualization is achieved using Tableau
- Time taken by the analysis was 3 min 20sec for the dataset of size 657MB with 6 input splits and 1 data node. After using combiner optimization the time taken to run analysis was reduced by 40 seconds .This Analysis was run on Virtual Machine with RAM - 4GB.



Sum of Dep Delay in minutes for each Airlines.

Its clear from the above analysis that United Airlines has worst departure delay -14min

Aloha Airlines has departure of 1 min early. Hawaiian Airlines Inc has no departure delay

Analysis 2

Which airlines have flights travelling distances between 0- 50 miles and 900-1400?

- BloomFilter can be used for this analysis. Hot values will be the distance values lying between 900 -1400 and 0-50.
- The list of hot values are put in a csv file which is put into HDFS. DistributedCache class is used to read these hot values and filter the input values. False positivity probablility is set to 0.1% and bloomfilter will accept around 500 values.
- IdentityReducer is used to capture values in one single file ,which will ease the process of data visualization further.
- Tableau is used for data visualization

Filtered Distances

Description

C2	AirTran Airway..	Alaska Airlines ..	American Airlin..	American Eagle..	Comair Inc.	Continental Air..	JetBlue Airways	Pinnacle Airlin..	Skywest Airlin..	United Air Line..
24								Abc		Abc
28										Abc
30										Abc
31										Abc
36										Abc
905										Abc
1003									Abc	Abc
1182	Abc			Abc	Abc	Abc			Abc	
1379		Abc	Abc			Abc	Abc	Abc		

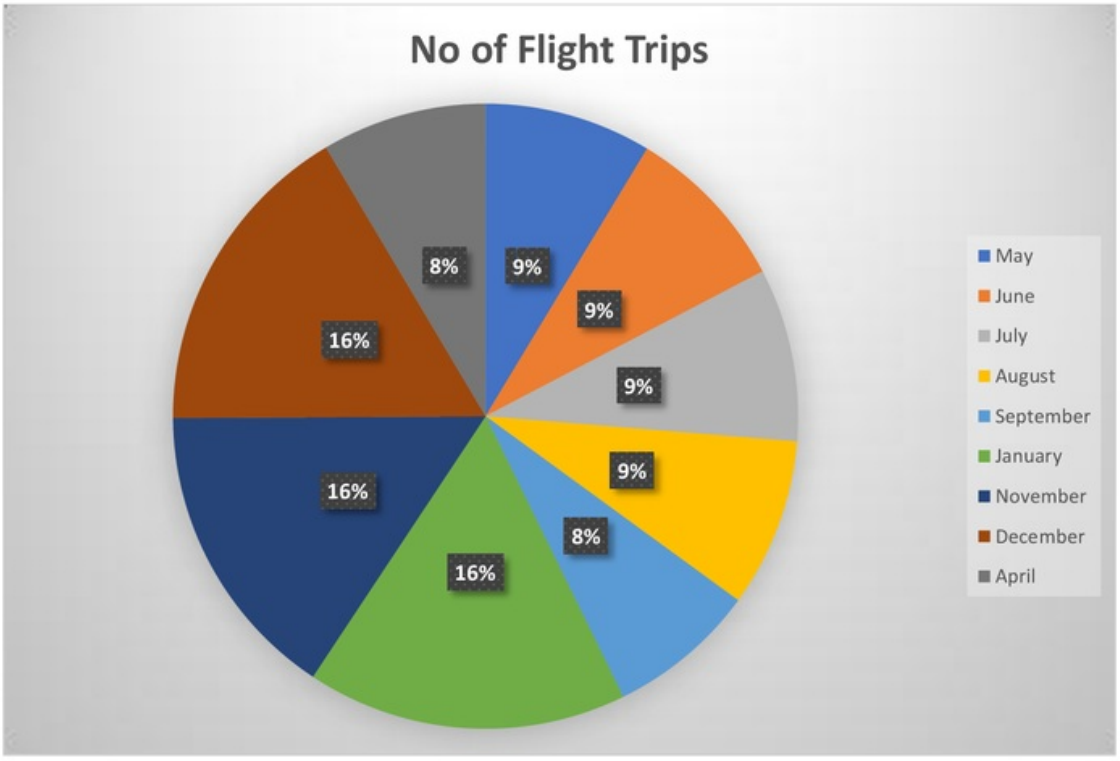
The view is broken down by Description vs. C2.

From above analysis its clear that United Airlines mostly has flights which travels lesser distance.
Continenetal Airlines flies flights ranging larger distance as shown in the above diagram

Analysis 3

What will be the Air Traffic for each month in USA?

- Partitioner is used for this analysis by keeping the key as Month number. There will be 12 separate files which will contain flight traffic details for that particular month.
- CustomWritable class is used to read the origin, destination, Airline company and month for eacxh trip.
- No of reducers is set to 12 as there are 12 months



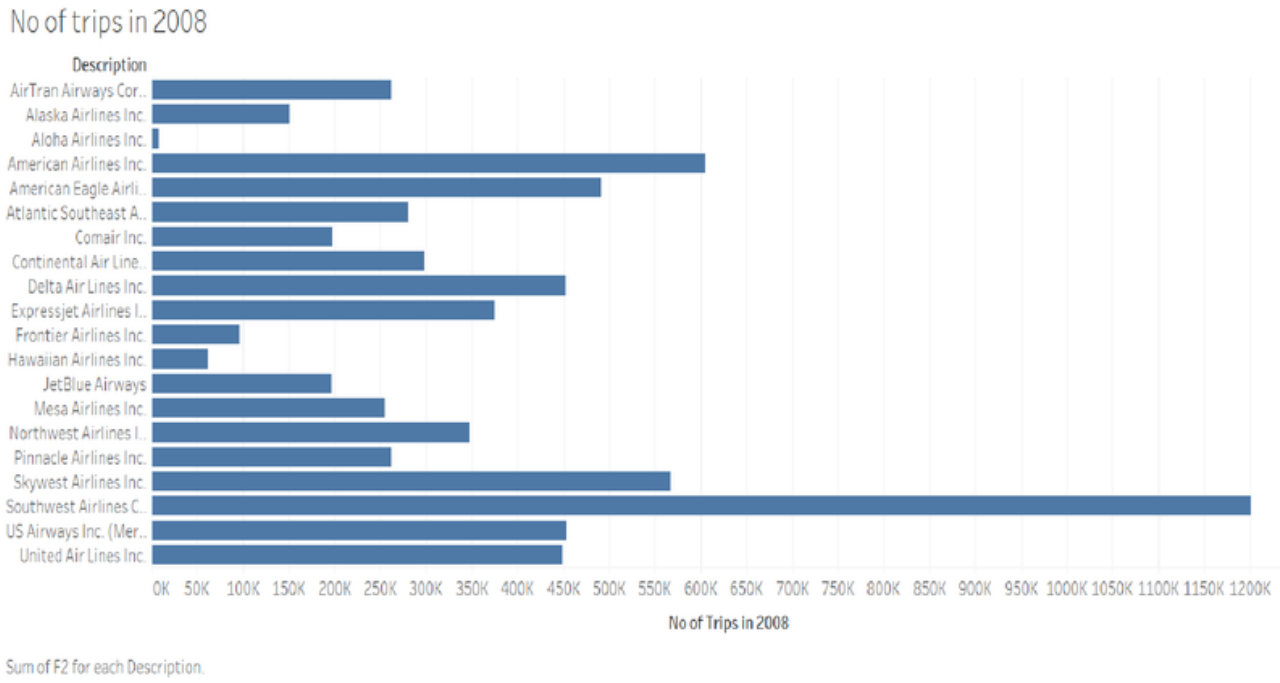
From above analysis its clear that people in USA travel more in November and December when compared to other months.

Analysis 4

Which airline company offered maximum number of trips in 2008 throughout USA?

This can be analysed by counting no of times the unique carrier code appears in the input file.

- Analyzed using counters as the the number of airline companies are fixed and their number is around 20 for the year 2008
- Reducer is not required since its count only operation using Counters provided by map reduce framework.
- Output can be seen directly in the terminal



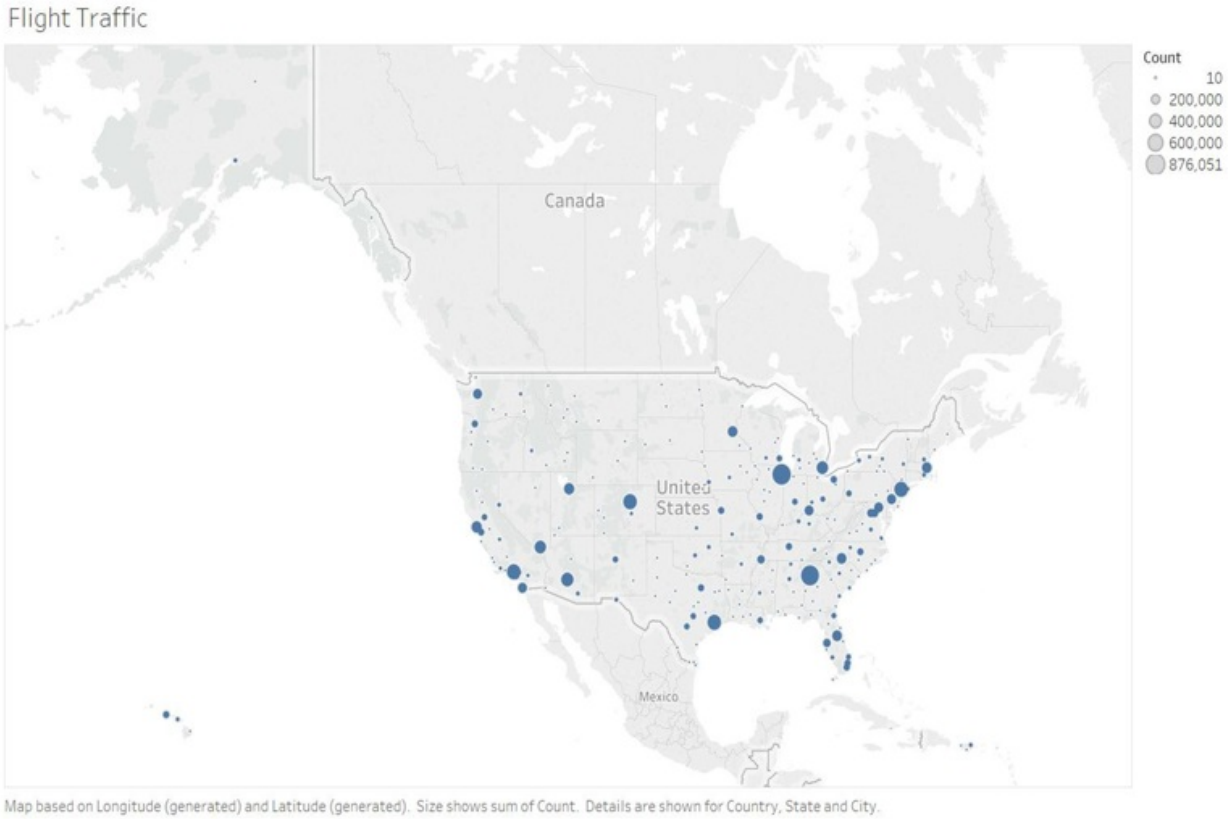
From the above analysis its clear that Southwest Airllines offered maximum number of trips for the year 2008.Aloha Airlines offered the least number of trips

Analysis 5

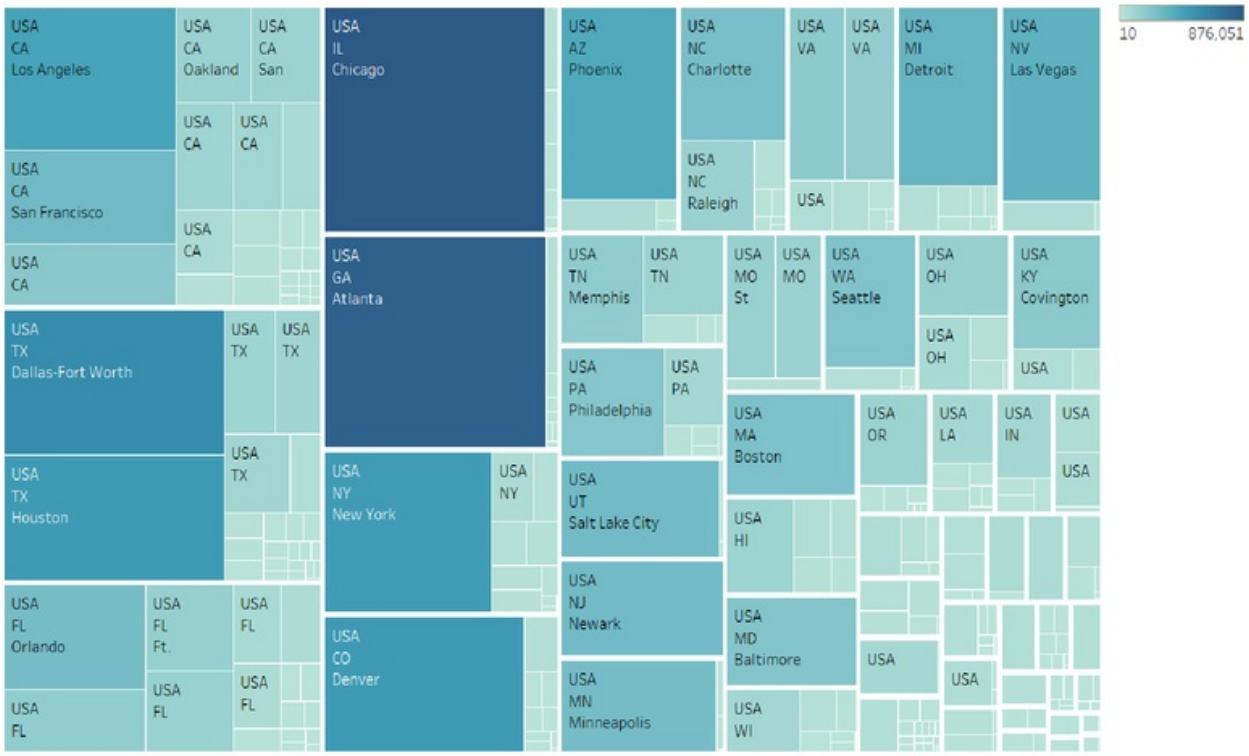
Which city had the highest flight traffic in 2008?

This analysis is performed using pig scripts

- Input File is loaded to get the count of flights arriving at each station
- Flights leaving from each station is also calculated
- Inner join is performed on above mentioned files.
- For each city code number of flights leaving and no of flights arriving is calculated to determine the total traffic



Flight Traffic



Country, State and City. Color shows sum of Count. Size shows sum of Count. The marks are labeled by Country, State and City.

From the above analysis its clear that Atlanta and Chicago had the highest flight traffic for the year 2008.

Analysis 6

Recommend a particular flights for a particular day if the given flight has no seats left.

This is achieved by putting data into hbase using Java APIs and later reading them depending on the input request.

Consider the input data shown below

Month	Day of month	Origin	Destination	Flight number	Tail number	Carrier
1	1	IAD	TPA	335	N432WN	WN
1	1	IAD	TPA	444	N2349E	9E
1	1	IAD	TPA	7766	N335XE	XE
1	2	IAD	DEN	3322	NA	OO
1	2	IAD	DEN	4452	N223UA	NA

This data will be inserted into hbase with rowkey = month+day of month +origin+Destination

Row key	Column Family-FlightDetails		
11IADTPA	Flight No:335	Tail No: N432WN	Carrier:WN
	Flight No:444	Tail No: N2349E	Carrier:9E
	Flight No:7766	Tail No: N335XE	Carrier:XE
12IADDEN	Flight No:3322	Carrier:OO	
	Flight No:4452	Tail No:N223UA	

Whenever flight information between two cities on a particular day is asked, row key is formed by appending those values, the values returned by the key will give flight information.

Analysis 7

Finding flights that have travelled for maximum amount of time

This analysis is done using top ten pattern.The reputation is chosen as Elapsed time to find out the maximum values.

Conclusion:

Hawaiian Airlines has the flights which travel maximum distance per trip

Code

Analysis1

```
/*
 * 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 final1;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import java.util.ArrayList;
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.Text;
import org.apache.hadoop.io.Writable;
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.input.MultipleInputs;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
```

```
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

/**
 *
 * @author sumedh
 */
public class Final1 {

        public static class AverageCountTuple implements Writable {

                private int average;
                private int count;

    public int getAverage() {
       return average;
    }

    public void setAverage(int average) {
       this.average = average;
    }

    public int getCount() {
       return count;
    }

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

```
        @Override
        public void readFields(DataInput in) throws IOException {
                average = in.readInt();
                count = in.readInt();

        }

        @Override
        public void write(DataOutput out) throws IOException {
                // TODO Auto-generated method stub
                out.writeInt(average);
    out.writeInt(count);
        }

@Override
public String toString() {
   return String.valueOf(this.average);
}


    }

public static class Map1 extends Mapper<Object, Text, Text, AverageCountTuple>{
IntWritable delaytimeh = new IntWritable();
Text carrierh = new Text();
```

```
    public void map(Object key, Text value, Context context){
       String carrier;
       Integer delaytime;
       try{
          AverageCountTuple out = new AverageCountTuple();
          String row[] = value.toString().split(",");
           carrier = row[8];
           delaytime = Integer.parseInt(row[15]);
           carrierh.set(carrier);
           delaytimeh.set(delaytime);
           out.setCount(1);
           out.setAverage(delaytime);
           context.write(carrierh, out);

       }catch(Exception e){


       }
    }
}

     public static class Reduce1 extends Reducer<Text,AverageCountTuple,Text,AverageCountTuple>{

    private IntWritable totalDelay = new IntWritable();
    private IntWritable res = new IntWritable();
    private AverageCountTuple newrs = new AverageCountTuple();
```

```
        protected void reduce(Text key, Iterable<AverageCountTuple> values, Context context) throws
IOException, InterruptedException {
          //int sum = 0;
          int count = 0;
          int avg = 0;
                        int sum = 0;
               int cnt = 0;
                        for (AverageCountTuple val : values) {
                                  sum += val.getCount()* val.getAverage();
                                  cnt += val.getCount();
                        }
               if (cnt != 0)
               {
                        avg = sum / cnt;
               }
                        res.set(avg);
               newrs.setAverage(avg);

            context.write(key, newrs);
        }


  }

      public static class JoinMapper1 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[] separatedInput = value.toString().split("\t");


        String id = separatedInput[0].trim();
    System.out.println(id);
        if (id == null)
        {
           return;
        }
        outKey.set(id);
        outValue.set("@"+value );
        context.write(outKey, outValue);


}
}

public static class JoinMapper2 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 id = value.toString().split(",")[0].trim();

        String Id = id.substring(1,id.length()-1);

        System.out.println(Id);


      if (id == null)

      {

        return;

      }

      outKey.set(Id);

      outValue.set("$"+value );

      context.write(outKey, outValue);


    }


}


 public static class JoinReducer extends Reducer<Text,Text,Text,Text>

     {

       private static final Text EMPTY_TEXT = new Text();

       private Text tmp = new Text();


       private ArrayList<Text> listA = new ArrayList<Text>();

                private ArrayList<Text> listB = new ArrayList<Text>();

                private String joinType = null;
```

```
    // @Override
     protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {
       listA.clear();
       listB.clear();

       while(values.iterator().hasNext())
       {
          tmp = values.iterator().next();
          if (tmp.charAt(0) == '@')
          {
             listA.add(new Text(tmp.toString().substring(1)));
          } else if (tmp.charAt(0) == '$')
          {
             listB.add(new Text(tmp.toString().substring(1)));
          }
       }

       executeJoinLogic(context);

       }

     private void executeJoinLogic(Context context) throws IOException, InterruptedException {

         if(!listA.isEmpty() && !listB.isEmpty())
         {
         for (Text A : listA) {
```

```
            {
               for (Text B : listB) {
                  context.write(A,B);

               }

            }
         }
         }

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

        Configuration conf = new Configuration();
        Job job1 = Job.getInstance(conf,"Highest Delay");
        job1.setJarByClass(Final1.class);
        job1.setMapperClass(Map1.class);
        job1.setMapOutputKeyClass(Text.class);
        job1.setMapOutputValueClass(AverageCountTuple.class);
        job1.setCombinerClass(Reduce1.class);
        job1.setReducerClass(Reduce1.class);
        job1.setOutputKeyClass(Text.class);
        job1.setOutputValueClass(AverageCountTuple.class);

        FileInputFormat.addInputPath(job1,new Path(args[0]));
        FileOutputFormat.setOutputPath(job1, new Path(args[1]));
```

```
        boolean complete = job1.waitForCompletion(true);


    Configuration conf2 = new Configuration();
    Job job2 = Job.getInstance(conf2 ,"chaining");
    if(complete)
    {

     job2.setJarByClass(Final1.class);
     MultipleInputs.addInputPath(job2, new Path(args[1]), TextInputFormat.class, JoinMapper1.class);
     MultipleInputs.addInputPath(job2, new Path(args[2]), TextInputFormat.class, JoinMapper2.class);
     job2.setMapOutputKeyClass(Text.class);
     job2.setMapOutputValueClass(Text.class);
    //  job2.getConfiguration().set("join.type","inner");
     job2.setReducerClass(JoinReducer.class);
     job2.setOutputFormatClass(TextOutputFormat.class);

     TextOutputFormat.setOutputPath(job2,new Path(args[3]));
     job2.setOutputKeyClass(Text.class);
     job2.setOutputValueClass(Text.class);

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

    }

    }

}
```

Analysis 2

```
/*
 * 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 bloomfilter_lab6;

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.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.net.URI;
import java.net.URISyntaxException;
import org.apache.commons.lang.math.NumberUtils;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.filecache.DistributedCache;
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.Mapper;
```

```
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author sumedh
 */
public class BloomFilter_lab6 {

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

   public static class BloomFilterMapper extends Mapper<Object,Text,Text,NullWritable>
   {
      Funnel<Person> p = new Funnel<Person>() {
         @Override
         public void funnel(Person person, Sink into) {
           into.putInt(person.id);
         }
      };

      private BloomFilter<Person> friends = BloomFilter.create(p, 500, 0.1);

      @Override
      protected void setup(Context context) throws IOException  {
```

```
        Path[] files = DistributedCache.getLocalCacheFiles(context.getConfiguration());
          if (files != null && files.length > 0) {


            for (Path file : files) {


              try {
                File myFile = new File(file.toUri());
                BufferedReader bufferedReader = new BufferedReader(new
FileReader(myFile.toString()));
                String line = null;
                while ((line = bufferedReader.readLine()) != null) {
                  String[] split = line.split(",");
                  for (String i:split)
                  {


                    Integer id = Integer.parseInt(i.trim());


                  Person p = new Person(id.intValue());
                  friends.put(p);
                  }
                }
              } catch (IOException ex) {
                System.err.println("Exception while reading  file: " + ex.getMessage());
              }
            }
          }
```

```
    }

    @Override
    protected void map(Object key, Text value, Context context) throws IOException,
InterruptedException {
        String values[] = value.toString().split(",");


         if (NumberUtils.isNumber(values[18].trim()))
       {
        Person p = new Person(Integer.parseInt(values[18]));
        if(friends.mightContain(p))
        {
           Text val = new Text();
           String airline =  values[8].toString()+"\t"+values[18].toString();
           val.set(airline);
           context.write(val, NullWritable.get());
        }
    }

    }

    public class Reducer_stock extends Reducer<Text, NullWritable, Text, NullWritable> {


  public void reduce(Text key, Iterable<NullWritable> values,
                Context context
```

```
    ) throws IOException, InterruptedException {



       context.write(key, NullWritable.get());
    }
}



    }
    public static void main(String[] args) throws URISyntaxException {
            Configuration conf = new Configuration();

       Job job;
       try {
          job = Job.getInstance(conf, "bloom filter");

       job.setJarByClass(BloomFilter_lab6.class);
       job.setMapperClass(BloomFilterMapper.class);
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(NullWritable.class);

         DistributedCache.addCacheFile(new URI(args[0]), job.getConfiguration());

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(NullWritable.class);



       FileInputFormat.addInputPath(job, new Path(args[1]));
```

```
        FileOutputFormat.setOutputPath(job, new Path(args[2]));

        boolean success = job.waitForCompletion(true);
        System.out.println(success);

        }
        catch (IOException|InterruptedException|ClassNotFoundException ex) {

        }
    }

}
```

Analysis 3

```
/*
 * 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 final3;

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.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
```

```
import org.apache.commons.lang.math.NumberUtils;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
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.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author sumedh
 */
public class Final3 {



  public static class Mapper1 extends Mapper<Object,Text,Text,CustomWritable>
  {
    //Text fightDetails = new Text();
    Text month = new Text();
    @Override
    protected void map(Object key, Text value, Context context) throws IOException,
InterruptedException {
        String values[] = value.toString().split(",");
```

```
        if (NumberUtils.isNumber(values[2].trim()) && NumberUtils.isNumber(values[9].trim()))
       {



        CustomWritable cw = new CustomWritable();
       month.set(values[1].toString());
       //Text cw = new Text(values[16].toString());
       cw.setOrigin(values[16].toString());
       cw.setDestination(values[17].toString());
       cw.setFlightNUmber(Integer.parseInt(values[9].toString()));
       cw.setFlight(values[8].toString());
       context.write(month, cw);



     }


     }


   }


    public static class Reduce1 extends Reducer<Text,CustomWritable,Text,CustomWritable>
  {


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

```
            //Text nw = new Text(value.toString());
            context.write(key,value);
        }

    }









}


public static void main(String[] args) {


    Configuration conf = new Configuration();

    Job job;
    try {
     job = Job.getInstance(conf, "Divide");
     job.setJarByClass(Final3.class);
     job.setMapperClass(Mapper1.class);
     job.setMapOutputKeyClass(Text.class);
     job.setMapOutputValueClass(CustomWritable.class);
     job.setNumReduceTasks(12);
     job.setReducerClass(Reduce1.class);
```

```
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(CustomWritable.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        boolean success = job.waitForCompletion(true);


      }
      catch (IOException|InterruptedException|ClassNotFoundException ex) {

      }

      }

      }
```

Analysis 4

```
 /*
 * 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 final4;

import org.apache.hadoop.mapreduce.Mapper;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
```

```
import java.util.HashSet;
import java.util.logging.Level;
import java.util.logging.Logger;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Counter;
import org.apache.hadoop.mapreduce.Counters;
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.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author sumedh
 */
public class Final4 {

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


  public static class CountMapper extends Mapper<Object, Text, NullWritable, NullWritable> {
```

```
public static final String STATE_COUNTER_GROUP = "state";
public static final String UNKNOWN_COUNTER = "";
public static final String EMPTY_COUNTER = "";
public String[] sArray = new String[]{"WN","AQ","AS","AA","B6","CO","DL","EV",
                                      "F9","FL","HA","MQ","NW","OH","OO","UA",
                                      "US","XE","YV","9E"};

private HashSet<String> states = new HashSet<String>(Arrays.asList(sArray));

public void map(Object key, Text value, Context context) throws IOException, InterruptedException
{

  //Map<String, String> parsed = MRDPUtils.transformXMLToMap(value.toString());
   String[] row = value.toString().split(",");
  String airline = row[8].trim();

  if (airline != null
     && !airline.isEmpty())

  {
    // String[] tokens = airline.toUpperCase().split("\\s");

    // boolean unknown = true;
```

```
            //for (String state : tokens) {
              if (states.contains(airline)) {
                  context.getCounter(STATE_COUNTER_GROUP, airline).increment(1);
               //    unknown = false;
                 //  break;

               }
              else
              {
                 context.getCounter(STATE_COUNTER_GROUP, UNKNOWN_COUNTER).increment(1);
              }
            }

//              if (unknown) {
//                 context.getCounter(abc, UNKNOWN_COUNTER).increment(1);
//              }
//
//          }
//          else {
//                 context.getCounter(abc , EMPTY_COUNTER).increment(1);
//                 }
      }


   }



    public static void main(String[] args) throws IOException, InterruptedException,
ClassNotFoundException {
       Configuration conf = new Configuration();
```

```
  Job job1 = Job.getInstance(conf,"Counter");
  job1.setJarByClass(Final4.class);
  job1.setMapperClass(CountMapper.class);
  job1.setMapOutputKeyClass(NullWritable.class);
  job1.setMapOutputValueClass(NullWritable.class);

  //job1.setCombinerClass(Reduce1.class);
  //job1.setReducerClass(Reduce1.class);
  //job1.setOutputKeyClass(Text.class);
  //job1.setOutputValueClass(AverageCountTuple.class);

  FileInputFormat.addInputPath(job1,new Path(args[0]));
  FileOutputFormat.setOutputPath(job1, new Path(args[1]));
  //System.exit(job1.waitForCompletion(true) ? 0:1);



  //FileOutputFormat.setOutputPath(job1, new Path(args[1]));

int code = job1.waitForCompletion(true) ? 0:1;

if (code == 0)
{
   for(Counter counter : job1.getCounters().getGroup(CountMapper.STATE_COUNTER_GROUP))
   {
      System.out.println(counter.getDisplayName()+"\t"+counter.getValue());
   }
}
```

```
   }

}
```

Analysis 5

```
log =     LOAD '/home/sumedh/2008.csv' using PigStorge(',');
grpd = GROUP log BY $16;
cntd = FOREACH grpd GENERATE group,COUNT(log);
STORE cntd INTO 'outputcount';

log1 =    LOAD '/home/sumedh/2008.csv' using PigStorge(',');
grpd1 = GROUP log1 BY $16;
cntd1 = FOREACH grpd1 GENERATE group,COUNT(log1);
STORE cntd1 INTO 'outputcount1';

t2 = LOAD '/home/sumedh/Join/t2' AS (place2,count2);
t1 = LOAD '/home/sumedh/Join/t1' AS (place1,count1);
joined = JOIN t1 BY place1, t2 BY place2;
store joined INTO 'joinedoutput';

log2 = LOAD '/home/sumedh/Join/t3 AS (name1,value1,name2,value2);
sum2 = FOREACH log2 GENERATE (name1,value1+value2);
store sum2 INTO 'finalOutput';
```

Analysis 6

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

```
 Configuration con = HBaseConfiguration.create();
 HBaseAdmin admin = new HBaseAdmin(con);
 HTableDescriptor tableDescriptor = new
 HTableDescriptor(TableName.valueOf("Flight"));
 tableDescriptor.addFamily(new HColumnDescriptor("FlightDetails"));
admin.createTable(tableDescriptor);
 System.out.println(" Table created ");

public class GettingData {

  public static void main(String args[]) throws IOException
  {
   Configuration config = HBaseConfiguration.create();

   // Instantiating HTable class
   HTable table = new HTable(config, "Flight");

   // Instantiating Get class
   Get g = new Get(Bytes.toBytes("11IADTPA"));

   // Reading the data
   Result result = table.get(g);

   // Reading values from Result class object
   byte [] value = result.getValue(Bytes.toBytes("FlightDetails"),Bytes.toBytes("FlightNo"));
```

```
    byte [] value1 = result.getValue(Bytes.toBytes("FlightDetails"),Bytes.toBytes("TailNo"));

    byte [] value2 = result.getValue(Bytes.toBytes("FlightDetails"),Bytes.toBytes("Carrier"));


    // Printing the values
    String FlightNo = Bytes.toString(value);
    String TailNo = Bytes.toString(value1);
    String Carrier = Bytes.toString(value2);

    System.out.println("flightno: " + FlightNo + " TailNo: " + TailNo + "Carrier:" + Carrier);
```

Analysis 7

```
/*
 * 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 final5;
import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import java.util.ArrayList;
import java.util.TreeMap;
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.io.Writable;
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.input.MultipleInputs;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.commons.lang.math.NumberUtils;
/**
 *
 * @author sumedh
 */
public class Final5 {

 public static class TopMapper extends Mapper<Object,Text,
                                NullWritable,Text>
 {

    private TreeMap<Integer,Text> reputation =
        new TreeMap<Integer, Text>();

     @Override
     protected void map(Object key, Text value, Context context) throws IOException,
InterruptedException {
        String row[] = value.toString().split(",");
```

```
        if (NumberUtils.isNumber(row[11].trim()))
        {

         reputation.put(Integer.parseInt(row[11].toString())
                , new Text(value.toString()));
        if (reputation.size() > 10)
        {
           reputation.remove(reputation.firstKey());
      }
      }
      }

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

         for(Text t:reputation.descendingMap().values())
    {
       context.write(NullWritable.get(),t);
    }

      }


}


public static class Reduce1 extends Reducer<NullWritable,Text,NullWritable,Text>
 {
```

```
    private TreeMap<Integer, Text> repToRecordMap = new TreeMap<Integer, Text>();


    public void reduce(NullWritable key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {


     {
        for (Text value : values) {


   repToRecordMap.put(Integer.parseInt(value.
         toString().split(",")[11]),new Text(value));
        }


        if (repToRecordMap.size() > 10)
        {
         repToRecordMap.remove(repToRecordMap.firstKey());
        }


        for(Text t:repToRecordMap.descendingMap().values())
        {
           context.write(NullWritable.get(), t);
        }
      }
      }
```

```
}

public static void main(String[] args) {



   Configuration conf = new Configuration();


   Job job;
   try {
    job = Job.getInstance(conf, "Topten");
    job.setJarByClass(Final5.class);
    job.setMapperClass(TopMapper.class);
    job.setMapOutputKeyClass(NullWritable.class);
    job.setMapOutputValueClass(Text.class);
   //  job.setNumReduceTasks(12);
    job.setReducerClass(Reduce1.class);
    job.setOutputKeyClass(NullWritable.class);
    job.setOutputValueClass(Text.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    boolean success = job.waitForCompletion(true);


   }
```

```
        catch (IOException|InterruptedException|ClassNotFoundException ex) {

        }

    }

}
```