LARGE SCALE DATA PROCESSING (CSE 3025)
WIN SEMESTER 2017-18

NAME: B.SHUBANKAR

REGNO: 15BCE1123

PROFESSOR: DR. Maheswari N

LAB-7

1. Display top k records

Salary details are maintained in the text file. Identify the top 10 salary details from the file. Write the details in HDFS.

PROGRAM:

```
import java.io.IOException;

import java.util.*;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.conf.*;

import org.apache.hadoop.io.*;

import org.apache.hadoop.mapreduce.*;

import org.apache.hadoop.mapreduce.lib.input.*;

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

```
public class l7_1 {


      public static class Map extends Mapper<LongWritable,Text,NullWritable,Text>
{
       private TreeMap<Integer, Text> salary = new TreeMap<Integer, Text>();
       public void map(LongWritable key, Text value, Context context) throws
IOException, InterruptedException
{
String line = value.toString();
       String[] elements=line.split(",");
       int i= Integer.parseInt(elements[1]);
       salary.put(i,new Text(value));
       if (salary.size() > 10) {
       salary.remove(salary.firstKey());
}
}
       protected void cleanup(Context context) throws IOException,
InterruptedException
{
       for ( Text name : salary.values() ) {
context.write(NullWritable.get(), name);
```

```
}

}

}

    public static class Reduce extends Reducer<NullWritable, Text, NullWritable,
Text>

{

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

InterruptedException {

    TreeMap<Integer, Text> salary = new TreeMap< Integer,Text>();

    for (Text value : values) {

String line = value.toString();

    String[] elements=line.split(",");

    int

i= Integer.parseInt(elements[1]);

    salary.put(i, new Text(value));

    if (salary.size() > 10) {

salary.remove(salary.firstKey());

}

}

    for (Text t : salary.values()) {
```

```
context.write(NullWritable.get(), t);
}
}
}
        public static void main(String[] args) throws Exception {
            Configuration conf = new Configuration();
            Job job = new Job(conf, "Customer");
            job.setJarByClass(l7_1.class);
            job.setOutputKeyClass(NullWritable.class);
            job.setOutputValueClass(Text.class);
            job.setMapperClass(Map.class);
job.setNumReduceTasks(1);
            job.setInputFormatClass(TextInputFormat.class);
            job.setOutputFormatClass(TextOutputFormat.class);
            FileInputFormat.addInputPath(job, new Path(args[0]));
            FileOutputFormat.setOutputPath(job,new Path(args[1]));
            job.waitForCompletion(true);
}
}
```

2.Counters

Salary details are maintained in the text file. Create the user defined counters. Count the number of persons having zero years of experience and write the details of country name and the salary with zero years of experience in HDFS. Also count the number of persons earning the salary greater than 50,000.

PROGRAM:

```
import java.io.IOException;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.conf.*;

import org.apache.hadoop.io.*;

import org.apache.hadoop.mapreduce.*;

import org.apache.hadoop.mapreduce.lib.input.*;

import org.apache.hadoop.mapreduce.lib.output.*;

 public class l7_2 {

 public enum ct

{

 cnt,nt

};
```

```
 public static class Map extends Mapper<LongWritable,Text, Text,
IntWritable>

{

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

String line = value.toString();

 String[] elements=line.split(",");

 int

i= Integer.parseInt(elements[1]);

 float

exp= Float.parseFloat(elements[0]);

 Text tt=new Text(elements[3]);

 if(exp==0.0)

{

 context.getCounter(ct.cnt).increment(1);

 context.write(tt,new IntWritable(i));

}

 if(i > 50000)
```

```
{
 context.getCounter(ct.nt).increment(1);
}
}
}
  public static void main(String[] args) throws Exception {
          Configuration conf = new Configuration();
          Job job = new Job(conf, "Customer");
          job.setJarByClass(l7_2.class);
          job.setOutputKeyClass(Text.class);
          job.setOutputValueClass(IntWritable.class);
          job.setMapperClass(Map.class);
                  job.setNumReduceTasks(1);
          job.setInputFormatClass(TextInputFormat.class);
          job.setOutputFormatClass(TextOutputFormat.class);
          FileInputFormat.addInputPath(job, new Path(args[0]));
          FileOutputFormat.setOutputPath(job,new Path(args[1]));
          job.waitForCompletion(true);
```

```
        Counters cn=job.getCounters();
        cn.findCounter(ct.cnt).getValue();
        cn.findCounter(ct.nt).getValue();
}
}
```

OUTPUT: