//4)Which top 5 employers file the most petitions each year?

package h1b;

import java.io.IOException;

import java.util.TreeMap;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.LongWritable;

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.Partitioner;

import org.apache.hadoop.mapreduce.Reducer;

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

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

public class question4 {

public static class MyMapper extends

Mapper<LongWritable, Text, Text, Text> {

public void map(LongWritable key, Text value, Context context)

throws IOException, InterruptedException {

String[] str = value.toString().split("\t");

String year = (str[7]);

String petition\_employer = str[3];

context.write(new Text(year + "\t" + str[2]), new Text(

petition\_employer));

}

}

public static class year\_partitionerclass extends Partitioner<Text, Text> {

@Override

public int getPartition(Text key, Text value, int numReduceTasks) {

String[] rec = key.toString().split("\t");

String year = rec[0];

if (year.equals("2011")) {

return 0;

} else if (year.equals("2012")) {

return 1;

} else if (year.equals("2013")) {

return 2;

} else if (year.equals("2014")) {

return 3;

} else if (year.equals("2015")) {

return 4;

} else {

return 5;

}

}

}

public static class MyReducer extends

Reducer<Text, Text, NullWritable, Text> {

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

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

throws IOException, InterruptedException {

int count = 0;

for (Text val : values) {

count++;

}

String mytotal = String.format("%d", count);

String myValue = key.toString();

myValue = myValue + ',' + mytotal;

repToRecordMap.put(new Long(count), new Text(myValue));

if (repToRecordMap.size() > 5) {

repToRecordMap.remove(repToRecordMap.firstKey());

}

}

protected void cleanup(Context context) throws IOException,

InterruptedException {

for (Text t : repToRecordMap.values()) {

context.write(NullWritable.get(), t);

}

}

}

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

ClassNotFoundException, InterruptedException {

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "");

job.setJarByClass(question4.class);

job.setMapperClass(MyMapper.class);

job.setPartitionerClass(year\_partitionerclass.class);

job.setReducerClass(MyReducer.class);

job.setNumReduceTasks(6);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

job.setOutputKeyClass(NullWritable.class);

job.setOutputValueClass(Text.class);

FileInputFormat.setInputPaths(job, new Path(args[0]));

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

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

}

}