**Following is the map reduce code to block the NA records and sales of tvs for each state.**

**2. Write a Map Reduce program to calculate the total units sold for each Company.**

*Company will be first parameter. So it is linearray[0] will be taken in mapper class*

package Session4;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

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 Task4

{

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

{

private final static IntWritable one = new IntWritable(1);

private Text word = new Text();

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

) throws IOException, InterruptedException {

String linearray[] = value.toString().split("\\|");

if(!linearray[0].equals("NA") && !linearray[1].equals("NA"))

{

word.set(linearray[0]);

context.write(word,one);

}

}

}

public static class IntSumReducer

extends Reducer<Text,IntWritable,Text,IntWritable> {

private IntWritable result = new IntWritable();

public void reduce(Text key, Iterable<IntWritable> values,

Context context

) throws IOException, InterruptedException {

int sum = 0;

for (IntWritable val : values) {

sum += val.get();

}

result.set(sum);

context.write(key, result);

}

}

public static class Task4Partitioner extends Partitioner<Text, IntWritable>

{

@Override

public int getPartition(Text key, IntWritable value, int numreducetasks) {

// TODO Auto-generated method stub

String str=key.toString();

char ch = str.charAt(0);

if(ch>='A' && ch<='F')

return 0;

else if(ch >='G' && ch <='L')

return 1;

else if(ch>='M' && ch <='R')

return 2;

else

return 3;

}

}

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

{

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "Task4");

job.setJarByClass(Task4.class);

job.setMapperClass(Map.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);

job.setNumReduceTasks(4);

job.setPartitionerClass(Task4Partitioner.class);

job.setReducerClass(IntSumReducer.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

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

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

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

}

}