Problem Statement

We have a dataset of sales of different TV sets across different locations.

Records look like:

Samsung|Optima|14|Madhya Pradesh|132401|14200

The fields are arranged like:

Company Name|Product Name|Size in inches|State|Pin Code|Price

There are some invalid records which contain 'NA' in either Company Name or Product Name.

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

**drive code**

package TotalUnitSale;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

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

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

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

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

public class TotalUnitSale

{

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

{

Configuration conf = new Configuration();

Job job = new Job(conf, "TV TotalUnitSale");// the job runs under this

job.setJarByClass(TotalUnitSale.class);

job.setMapOutputKeyClass(Text.class); //mapper key output

job.setMapOutputValueClass(IntWritable.class); //mapper output value

job.setOutputKeyClass(Text.class);// output key of the mapreduce

job.setOutputValueClass(IntWritable.class);//output value of the mapreduce

job.setMapperClass(TotalUnitSaleMapper.class);// Mapper class

job.setReducerClass(TotalUnitSaleReducer.class);//reducer class

job.setNumReduceTasks(2);

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);

}

}

**Mapper Code :**

package TotalUnitSale;

import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

public class TotalUnitSaleMapper extends Mapper<LongWritable, Text, Text, IntWritable>

{

private final static IntWritable unit = new IntWritable(1); // declaring the Mapper value

private Text CompanyName = new Text(); //declaring the Mapper key

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

{

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

StringTokenizer tokenizer=new StringTokenizer(Linearray[0]); //we have used the String Tokenizer class which takes array into single word/token.

while(tokenizer.hasMoreTokens()) // the while loop checks for the more tokens/words, if we have next token it will continue the loop

{

CompanyName.set(tokenizer.nextToken());

}

context.write(CompanyName, unit); // output of the Mapper Key and Value

}

}

**Reducer code**

package TotalUnitSale;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class TotalUnitSaleReducer extends Reducer<Text, IntWritable, Text, IntWritable>

{

public void reduce(Text CompanyName, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException

{

int sum=0; // declaring a variable sum

for(IntWritable value:values) // the for loop get the iterable values and counting the values

{

sum+=value.get();

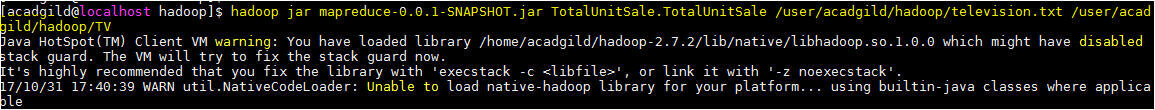
}

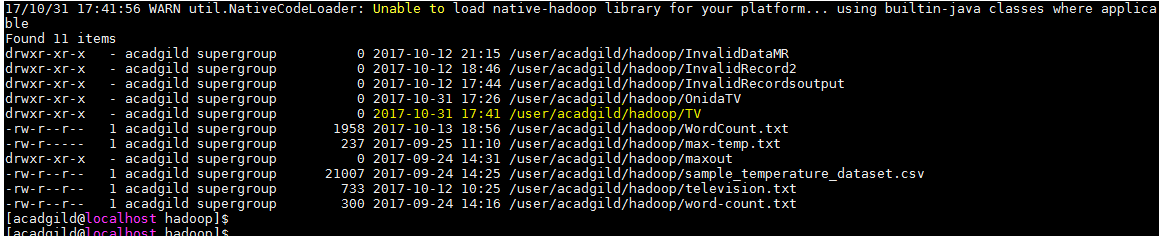
context.write(CompanyName, new IntWritable(sum)); // output of the the Key and value

}

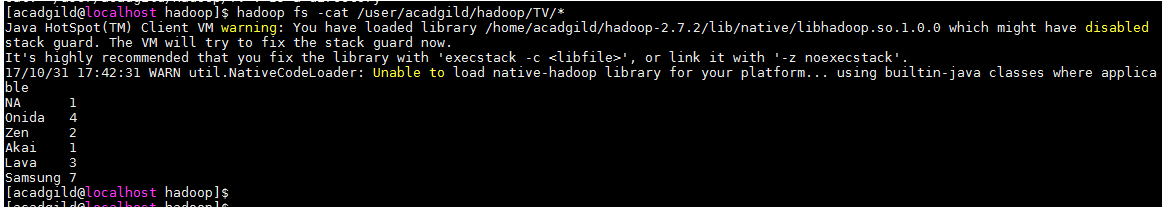
}

**hadoop jar mapreduce-0.0.1-SNAPSHOT.jar TotalUnitSale.TotalUnitSale/user/acadgild/hadoop/television.txt /user/acadgild/hadoop/TV**





**Output screen shot**



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

**company.**

**Driver Code**

package OnidaTotalUnit;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

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

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

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

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

public class OnidaTotalUnit

{

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

{

Configuration conf = new Configuration();

Job job = new Job(conf, "Onida Total Unit");// the job runs under this

job.setJarByClass(OnidaTotalUnit.class);

job.setMapOutputKeyClass(Text.class); //mapper key output

job.setMapOutputValueClass(IntWritable.class); //mapper output value

job.setOutputKeyClass(Text.class);//output key of the mapreduce

job.setOutputValueClass(IntWritable.class); //output value of the mapreduce

job.setMapperClass(OnidaMapper.class); // mapper class

job.setReducerClass(OnidaReducer.class);// reducer class

job.setNumReduceTasks(2);

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);

}

}

**Mapper Code**

package OnidaTotalUnit;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

public class OnidaMapper extends Mapper<LongWritable, Text, Text, IntWritable>

{

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

{

String[] Linearray = value.toString().split("\\|"); //the array is split into string value and stored in Linearray

if(Linearray[0].equals("Onida")) // checking the word Onida in the linearray[0], if it is Onida print the state name in linearray[3]and unit value

{

Text State = new Text(Linearray[3]);

IntWritable unit= new IntWritable(1);

context.write(State, unit);

}

}

}

**Reducer Code**

package OnidaTotalUnit;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class OnidaReducer extends Reducer<Text, IntWritable, Text, IntWritable>

{

public void reduce(Text State, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException

{

int sum = 0; // declaring the variable sum

for(IntWritable value:values) // the for loop get the iterable values and counting the values

{

sum+= value.get();

}

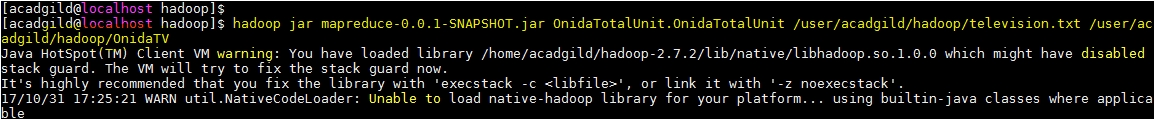
context.write(State, new IntWritable(sum)); // print the state name which is the key and the number of units stored in the sum

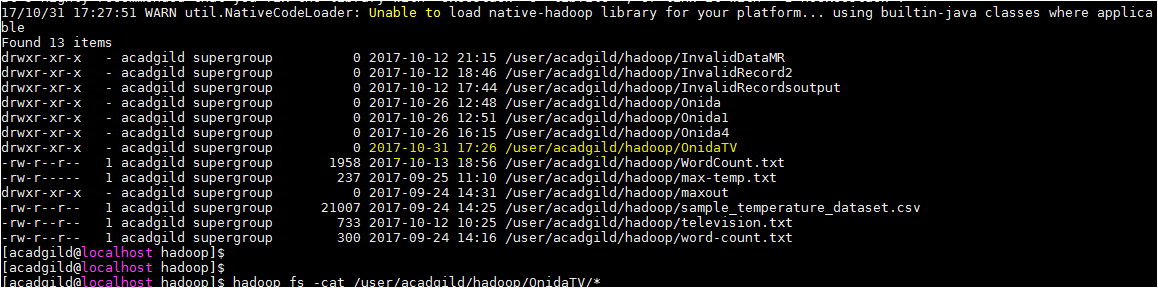
}

}

## **Command to run the jar**

## **hadoop jar mapreduce-0.0.1-SNAPSHOT.jar OnidaTotalUnit.OnidaTotalUnit/user/acadgild/hadoop/television.txt/user/acadgild/hadoop/OnidaTV**





## **Output screen**

