**Assignment 3.2**

**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.

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

**Three class** (Code is check-in GITHUB)

Mapper Class(MapperForTotalUnitsSoldForCompany.java)

**package** mapreduce.television;

**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** MapperForTotalUnitsSoldForCompany **extends** Mapper<LongWritable, Text, Text, IntWritable> {

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

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

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

String line = value.toString();

String NA = "NA";

String[] words = line.split("\\|");

String companyName = words[0];

String productName = words[1];

**if** (!(companyName.equals(NA) || productName.equals(NA))) {

Text outputKey = **new** Text(companyName.trim());

IntWritable outputValue = **new** IntWritable(1);

context.write(outputKey, outputValue);

}

}

}

**Reducer Class(ReducerForTotalUnitsSoldForCompany.java)**

**package** mapreduce.television;

**import** java.io.IOException;

**import** org.apache.hadoop.io.IntWritable;

**import** org.apache.hadoop.io.Text;

**import** org.apache.hadoop.mapreduce.Reducer;

**public** **class** ReducerForTotalUnitsSoldForCompany **extends** Reducer<Text,IntWritable,Text,IntWritable>

{

**public** **void** reduce(Text word, Iterable<IntWritable> values, Context con) **throws** IOException, InterruptedException {

**int** sum = 0;

**for** (IntWritable value : values) {

sum += value.get();

}

con.write(word, **new** IntWritable(sum));

}

}

**Driver Class(TotalUnitsSoldForCompany.java)**

**package** mapreduce.television;

**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.output.FileOutputFormat;

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

**public** **class** TotalUnitsSoldForCompany {

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

Configuration c = **new** Configuration();

Job totalUnitsSoldJob = **new** Job(c, "TotalUnitsSoldForCompany");

totalUnitsSoldJob.setJarByClass(TotalUnitsSoldForCompany.**class**);

totalUnitsSoldJob.setMapperClass(MapperForTotalUnitsSoldForCompany.**class**);

totalUnitsSoldJob.setReducerClass(ReducerForTotalUnitsSoldForCompany.**class**);

totalUnitsSoldJob.setOutputKeyClass(Text.**class**);

totalUnitsSoldJob.setOutputValueClass(IntWritable.**class**);

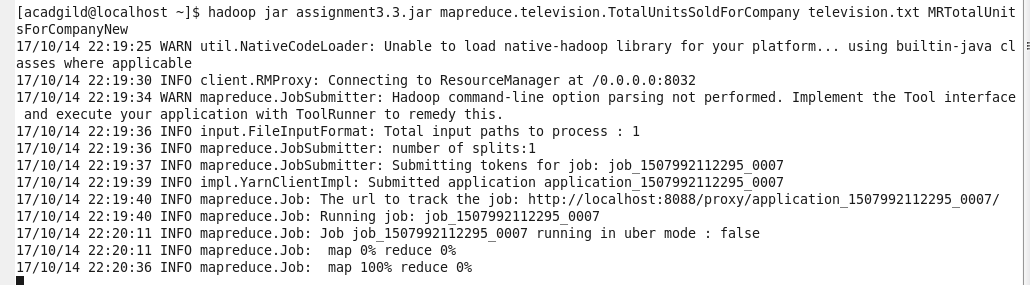
FileInputFormat.*addInputPath*(totalUnitsSoldJob, **new** Path(args[0]));

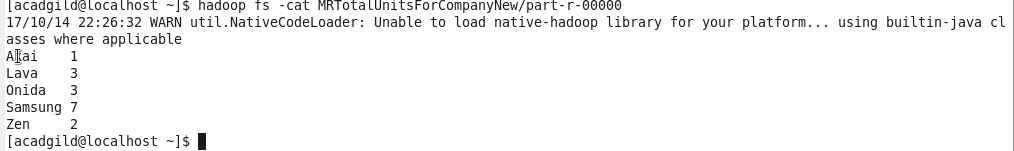
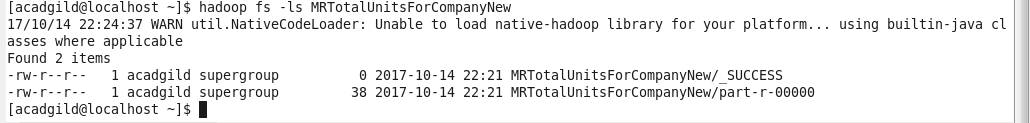
FileOutputFormat.*setOutputPath*(totalUnitsSoldJob, **new** Path(args[1]));

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

}

}





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

**Three class** (Code is check-in GITHUB)

TotalUnitsSoldForStateForOnida

**package** mapreduce.television;

**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.output.FileOutputFormat;

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

**public** **class** TotalUnitsSoldForStateForOnida {

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

Configuration c = **new** Configuration();

Job totalUnitsSoldJob = **new** Job(c, "TotalUnitsSoldForStateForOnida");

totalUnitsSoldJob.setJarByClass(TotalUnitsSoldForStateForOnida.**class**);

totalUnitsSoldJob.setMapperClass(MapperTotalUnitsSoldForStateForOnida.**class**);

totalUnitsSoldJob.setReducerClass(ReducerTotalUnitsSoldForStateForOnida.**class**);

totalUnitsSoldJob.setOutputKeyClass(Text.**class**);

totalUnitsSoldJob.setOutputValueClass(IntWritable.**class**);

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

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

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

}

}

Mapper -MapperTotalUnitsSoldForStateForOnida.java

**package** mapreduce.television;

**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;

// Mapper class to calculate the total units sold in each state for Onida company.

**public** **class** MapperTotalUnitsSoldForStateForOnida **extends** Mapper<LongWritable, Text, Text, IntWritable> {

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

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

// Onida|Lucid|18|Uttar Pradesh|232401|16200

String line = value.toString();

String NA = "NA";

String ONIDA="Onida";

String[] words = line.split("\\|");

String companyName = words[0];

String productName = words[1];

**if** (!(companyName.equals(NA) || productName.equals(NA))) {

**if** (companyName.equals(ONIDA)) {

String stateName = words[3];

Text outputKey = **new** Text(stateName.trim());

IntWritable outputValue = **new** IntWritable(1);

context.write(outputKey, outputValue);

}

}

}

}

**Reducer**

ReducerTotalUnitsSoldForStateForOnida.java

**package** mapreduce.television;

**import** java.io.IOException;

**import** org.apache.hadoop.io.IntWritable;

**import** org.apache.hadoop.io.Text;

**import** org.apache.hadoop.mapreduce.Reducer;

**public** **class** ReducerTotalUnitsSoldForStateForOnida **extends** Reducer<Text,IntWritable,Text,IntWritable>

{

**public** **void** reduce(Text word, Iterable<IntWritable> values, Context con) **throws** IOException, InterruptedException {

**int** sum = 0;

**for** (IntWritable value : values) {

sum += value.get();

}

con.write(word, **new** IntWritable(sum));

}

}

**Output**

