*Big Data and Hadoop Development*

**1. Introduction**

In this assignment you need to perform the tasks given.

**2. Objective**

This assignment will help you to consolidate the concepts learnt in the session 4.

**3. Prerequisites:**

None

**4. Associated Data Files:**

https://drive.google.com/file/d/0Bxr27gVaXO5sVjQ5QW0wQ3RCTUU/view?usp=sharing

**5. 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 filter out the invalid records. Map only job will fit for this

context.

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.lib.output.TextOutputFormat;

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

import org.apache.hadoop.mapreduce.Job;

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

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

public class Session3.Assignment {

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

if (args.length != 2) {

System.err.println("Usage: WordCount <input path> <output path>");

System.exit(-1);

}

//Job Related Configurations

Configuration conf = new Configuration();

//set Task name

Job job = new Job(conf, "Filterd NA Record");

job.setJarByClass(Session3.Assignment.class);

/\*set output key and value for map class\*/

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

/\*set class for map task\*/

//To set the mapper and reducer of this job

job.setMapperClass(FRMapper.class);

// Specify the number of reducer to 2

//job.setNumReduceTasks(2);

//Provide paths to pick the input file for the job

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

//set the input and output format class

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

//Provide paths to pick the output file for the job, and delete it if already present

Path outputPath = new Path(args[1]);

FileOutputFormat.setOutputPath(job, outputPath);

outputPath.getFileSystem(conf).delete(outputPath, true);

//Set the combiner

// job.setCombinerClass(Assignment3Reducer.class);

//set the input and output format class

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

//set up the output key and value classes

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(Text.class);

//execute the job

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

}

}

**Mapper class**

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 FRMapper extends Mapper<LongWritable, Text, Text, Text> {

@Override

protected 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") ))

{

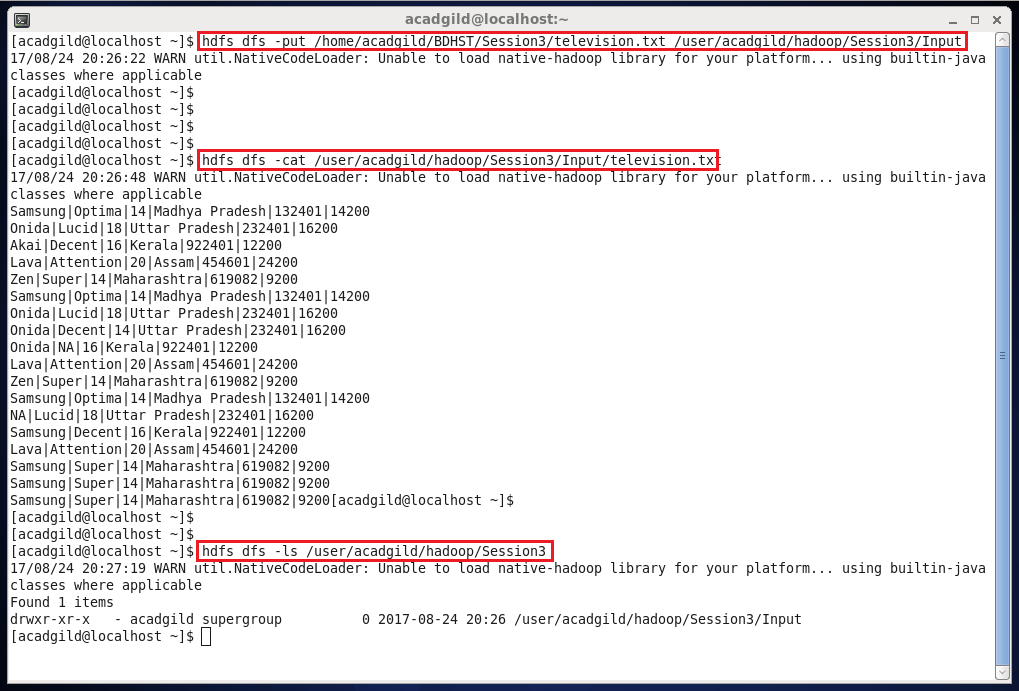
Text line = new Text(value);

context.write(new Text(line),new Text(""));

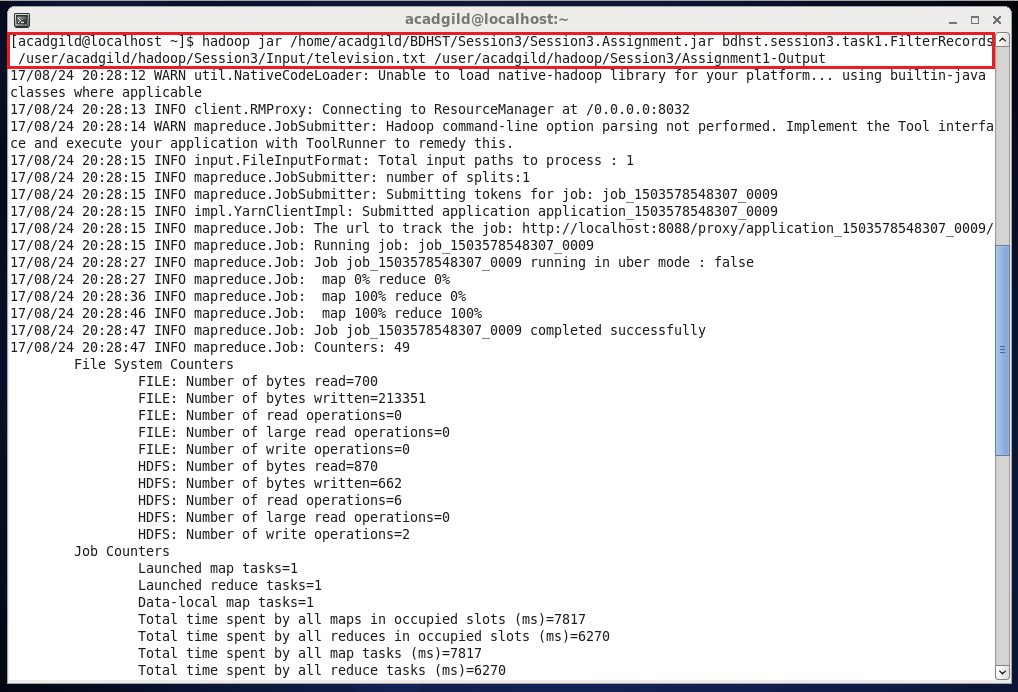
}

}

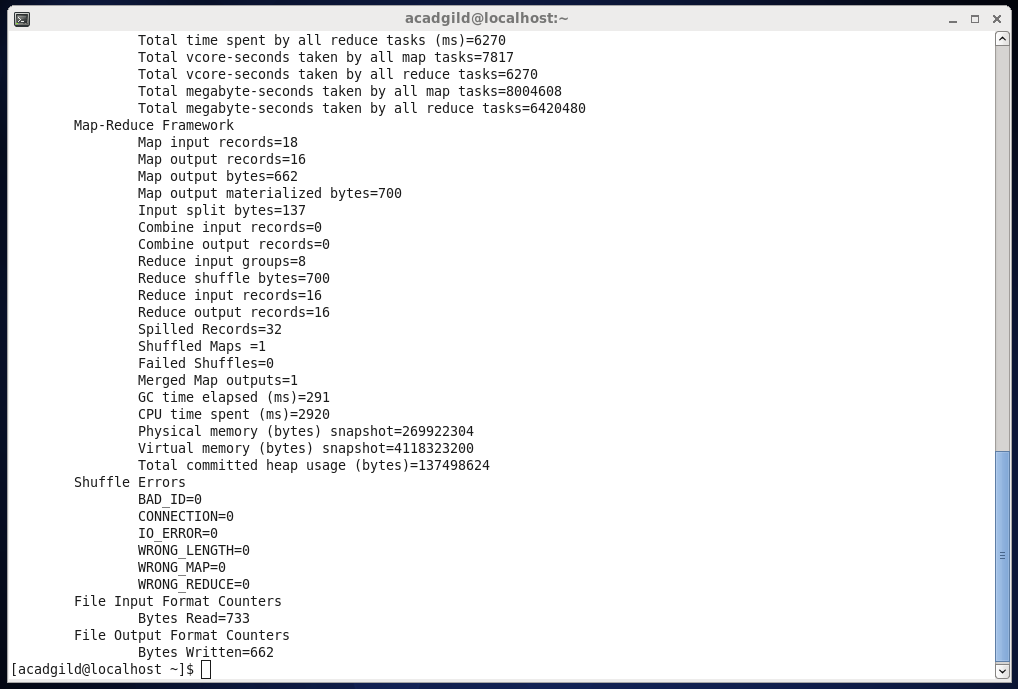
1. Move the data file to the HDFS in **/user/acadgild/Session3/Input**



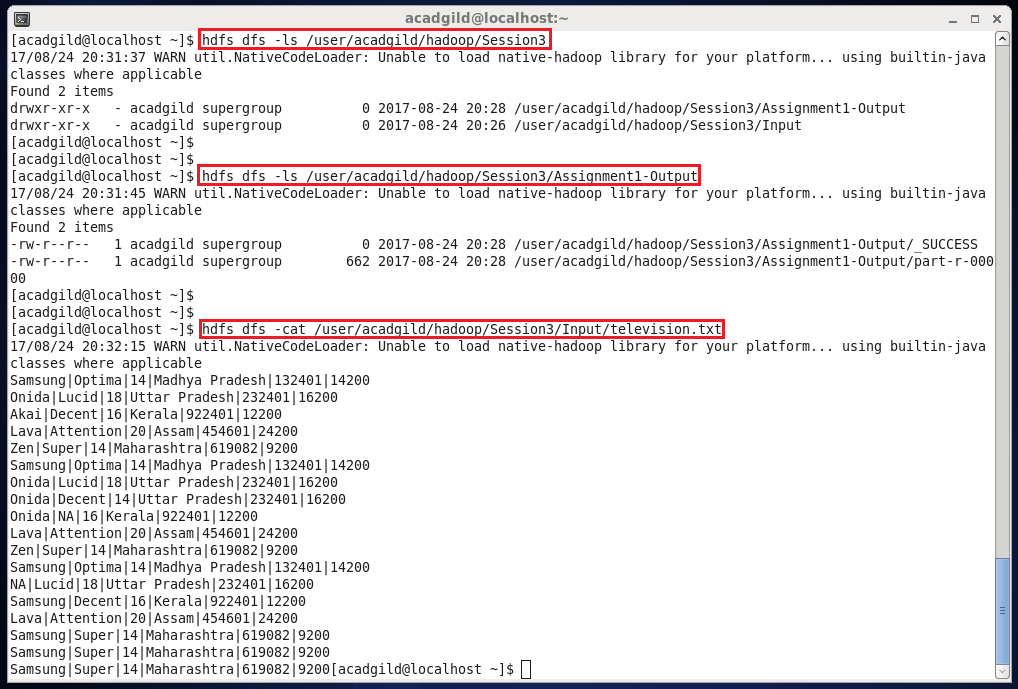
1. Execute the .jar file



1. Successfully executed .jar file (Creates Output in output folder provided in HDFS)



1. Output folder successfully created in Session3
2. Assignment1-Output folder contains success message file and result file (part-r-00000)



1. Contents of Result file (part-r-00000) Invalid Records, i.e. records with ‘NA’, are removed.