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.

Task 1:

Write a Map Reduce program to filter out the invalid records. Map only job will fit for this

context.

Put television.txt file to Hadoop file system

For that we can use this command

Hadoop fs –put television.txt /television.txt

To run Task using Hadoop we can use

Hadoop jar <Name of the jar> /location of input file /destination

To view the output we can use

Hadoop fs –cat /output/part-r-00000

Driver Class-:

```
package com.hem.hadoop.Assignment4;


import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.NullWritable;
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 DriverClass {

 public static void main(String[] args) throws Exception {
  if (args.length != 2) {
   System.err.println("Usage: Filter Invalid Records <input path> <output path>");
   System.exit(-1);
  }

        //Job Related Configurations
        Configuration conf = new Configuration();
        Job job = new Job(conf, "Filter Invalid Records");
        job.setJarByClass(DriverClass.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]));


  //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);
```

```
    //To set the mapper and reducer of this job
    job.setMapperClass(InvalidRecords.class);
    //job.setReducerClass(WordCountReducer.class);

    //Set the combiner
    //job.setCombinerClass(WordCountReducer.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(NullWritable.class);//Using Null Writable as we just want to filter records

    //execute the job
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

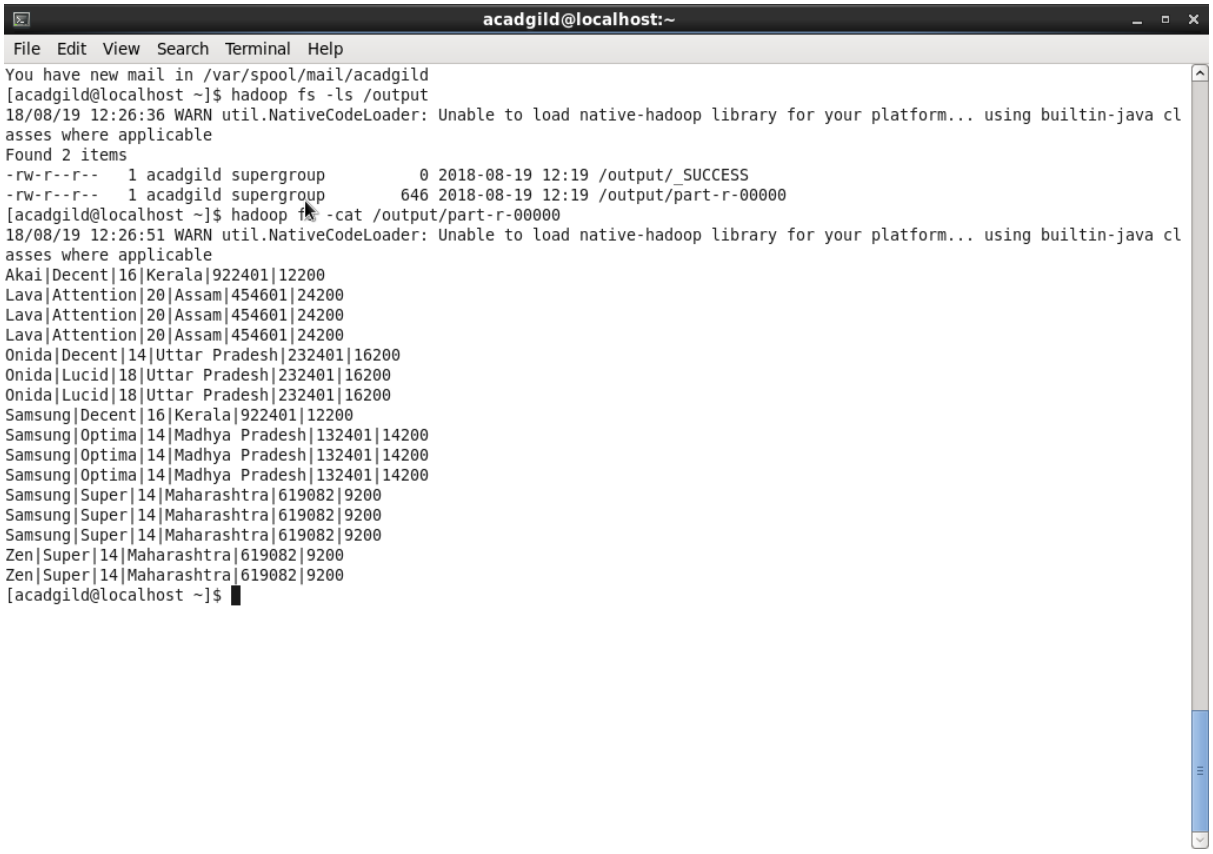
Mapper Class-:

```
package com.hem.hadoop.Assignment4;

import java.io.IOException;

import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class InvalidRecords extends Mapper<LongWritable, Text, Text, NullWritable>
{

        private Text word = new Text();

        @Override
```

```
        public void map(LongWritable key, Text value, Context context)
            throws IOException, InterruptedException {
             boolean found =false;
             String line = value.toString();
//           StringTokenizer tokenizer = new StringTokenizer(line);
//           while (tokenizer.hasMoreTokens()) {
//               word.set(tokenizer.nextToken());
//               context.write(word, one);
//           }
             System.out.println("This is output to mapper:"+key.toString());
             String words[] = line.split("\n");
             for(String wordSplit:words){
                    String tempWord[]=wordSplit.split("\\|");
                    for(String temp:tempWord){
                           if(temp.equalsIgnoreCase("NA")){
                                  found=true;
                                  break;
                           }

                    }
                    if(!found){
                           word.set(wordSplit);
                           context.write(word, NullWritable.get());
                    }

             }
      }
}
```

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
[acadgild@localhost ~]$ hadoop fs -put television.txt /television.txt
18/08/19 12:17:36 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ hadoop jar FilterInvalidRecord.jar /television.txt /output
18/08/19 12:18:28 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
18/08/19 12:18:30 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/08/19 12:18:32 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool in
terface and execute your application with ToolRunner to remedy this.
18/08/19 12:18:32 INFO input.FileInputFormat: Total input paths to process : 1
18/08/19 12:18:32 INFO mapreduce.JobSubmitter: number of splits:1
18/08/19 12:18:33 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1534649382244_0024
18/08/19 12:18:33 INFO impl.YarnClientImpl: Submitted application application_1534649382244_0024
18/08/19 12:18:33 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1534649382244_0024/
18/08/19 12:18:33 INFO mapreduce.Job: Running job: job_1534649382244_0024
18/08/19 12:18:46 INFO mapreduce.Job: Job job_1534649382244_0024 running in uber mode : false
18/08/19 12:18:46 INFO mapreduce.Job:  map 0% reduce 0%
18/08/19 12:18:55 INFO mapreduce.Job:  map 100% reduce 0%
18/08/19 12:19:07 INFO mapreduce.Job:  map 100% reduce 100%
18/08/19 12:19:07 INFO mapreduce.Job: Job job_1534649382244_0024 completed successfully
18/08/19 12:19:07 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=684
                FILE: Number of bytes written=216647
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=834
                HDFS: Number of bytes written=646
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=107120
                Total time spent by all reduces in occupied slots (ms)=64704
```

Output



Task 2:

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

Driver Class-:

```
package com.hem.hadoop.Assignment4;

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;

import com.hem.hadoop.WordCount.WordCountReducer;

public class DriverClass {
```

```
  public static void main(String[] args) throws Exception {
    if (args.length != 2) {
      System.err.println("Usage: Filter Records <input path> <output path>");
      System.exit(-1);
    }

       //Job Related Configurations
       Configuration conf = new Configuration();
       Job job = new Job(conf, "Filter Records");
       job.setJarByClass(DriverClass.class);


    // Specify the number of reducer to 2
    job.setNumReduceTasks(1);

    //Provide paths to pick the input file for the job
    FileInputFormat.setInputPaths(job, new Path(args[0]));


    //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);


    //To set the mapper and reducer of this job
    job.setMapperClass(TotalUnitSold.class);
    job.setReducerClass(TotalUnitSoldReducer.class);

    //Set the combiner
    job.setCombinerClass(TotalUnitSoldReducer.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(IntWritable.class);

    //execute the job
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

Mapper Class-:

```
package com.hem.hadoop.Assignment4;


import java.io.IOException;


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

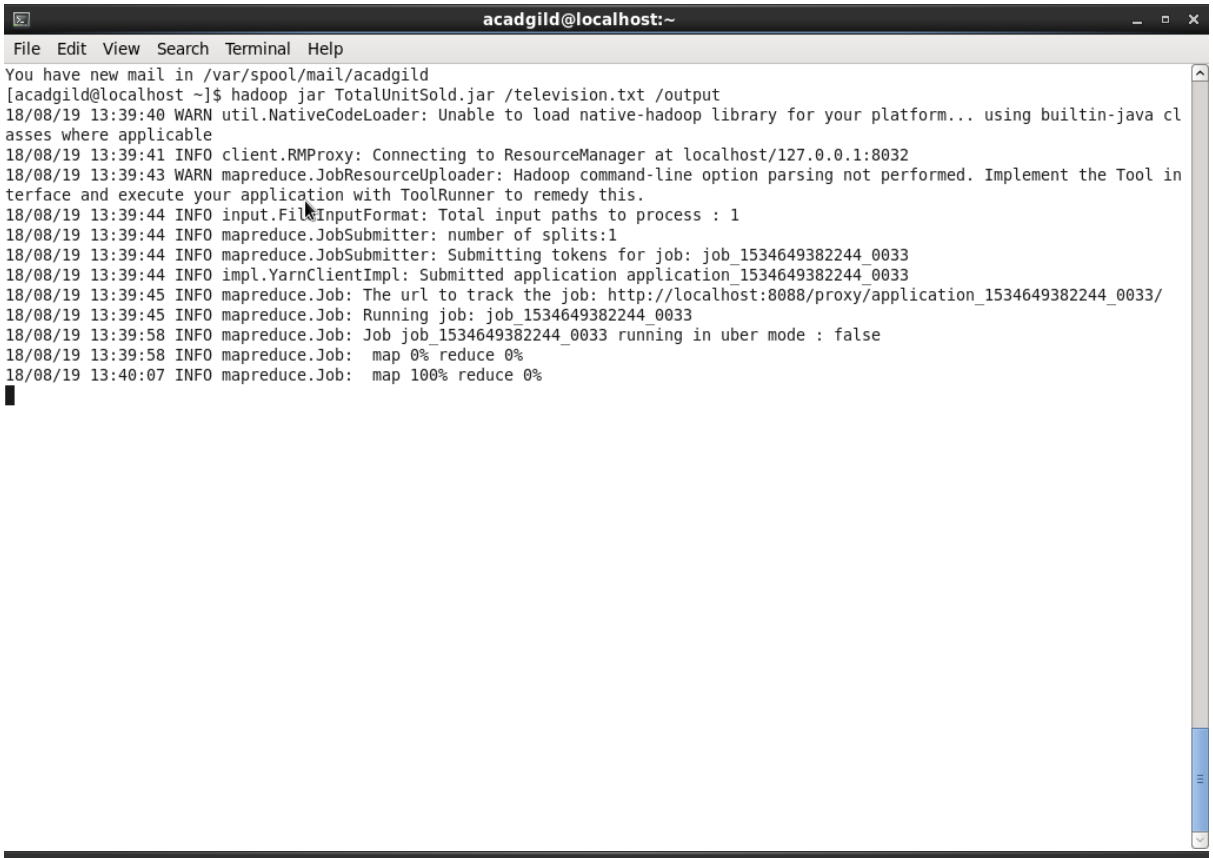
```
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class TotalUnitSold extends Mapper<LongWritable, Text, Text, IntWritable> {

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

	@Override
	public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
		String line = value.toString();
		System.out.println("This is output to mapper:" + key.toString());
		String words[] = line.split("\n");
		for (String wordSplit : words) {
			String tempWord[] = wordSplit.split("\\|");
			word.set(tempWord[0]);
			context.write(word, one);

		}

	}
}
```

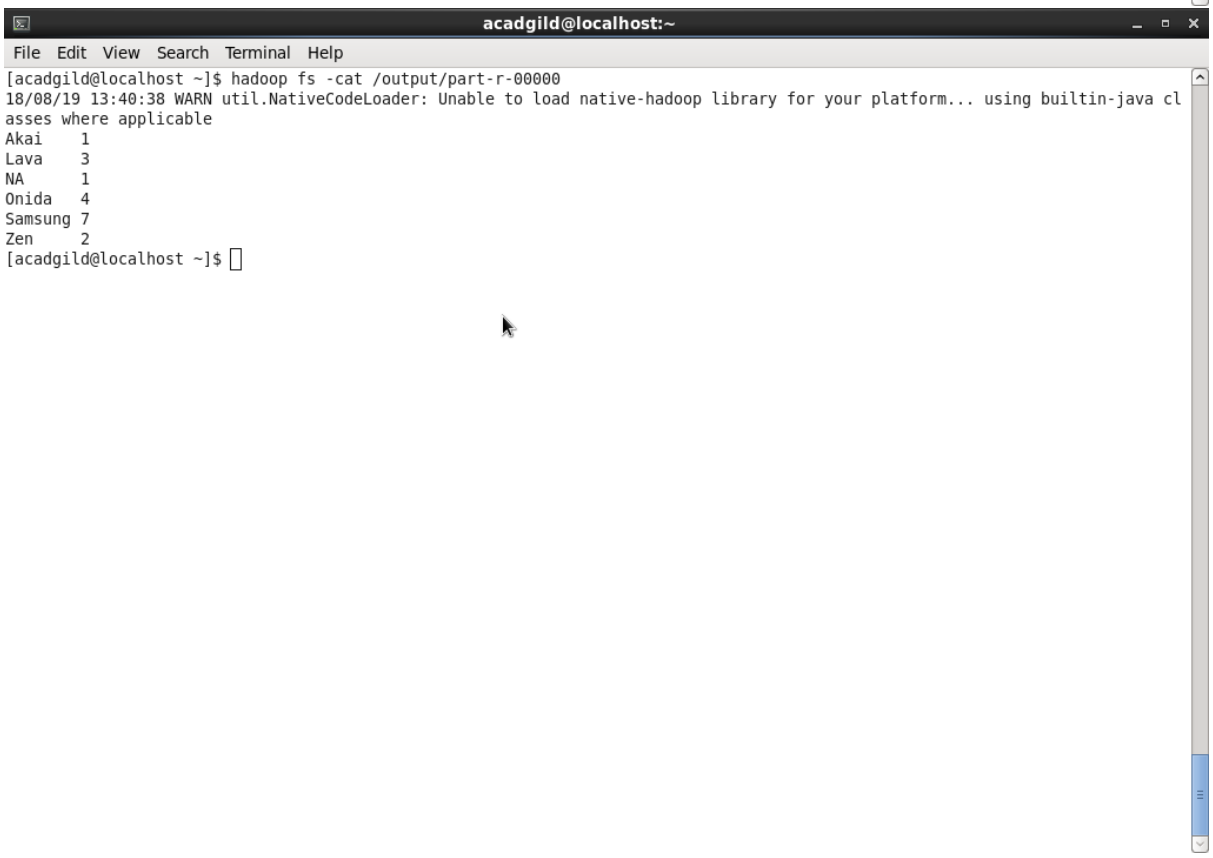
Reducer Class-:

```
package com.hem.hadoop.Assignment4;



import java.io.IOException;
```

```
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;



public class TotalUnitSoldReducer
 extends Reducer<Text, IntWritable, Text, IntWritable> {

 @Override
 public void reduce(Text key, Iterable<IntWritable> values,
    Context context)
    throws IOException, InterruptedException {
    System.out.println("From The Reducer=>"+key) ;

    int sum = 0;
    for (IntWritable value : values) {
                    sum+=value.get();
     }
     context.write(key, new IntWritable(sum));
 }
}
```

Output-:


acadgild@localhost:~
File Edit View Search Terminal Help
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ hadoop jar TotalUnitSold.jar /television.txt /output
18/08/19 13:39:40 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
18/08/19 13:39:41 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/08/19 13:39:43 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool in
terface and execute your application with ToolRunner to remedy this.
18/08/19 13:39:44 INFO input.FileInputFormat: Total input paths to process : 1
18/08/19 13:39:44 INFO mapreduce.JobSubmitter: number of splits:1
18/08/19 13:39:44 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1534649382244_0033
18/08/19 13:39:44 INFO impl.YarnClientImpl: Submitted application application_1534649382244_0033
18/08/19 13:39:45 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1534649382244_0033/
18/08/19 13:39:45 INFO mapreduce.Job: Running job: job_1534649382244_0033
18/08/19 13:39:58 INFO mapreduce.Job: Job job_1534649382244_0033 running in uber mode : false
18/08/19 13:39:58 INFO mapreduce.Job: map 0% reduce 0%
18/08/19 13:40:07 INFO mapreduce.Job: map 100% reduce 0%


acadgild@localhost:~
File Edit View Search Terminal Help
[acadgild@localhost ~]$ hadoop fs -cat /output/part-r-00000
18/08/19 13:40:38 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Akai 1
Lava 3
NA 1
Onida 4
Samsung 7
Zen 2
[acadgild@localhost ~]$

Task 3:

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

Driver Class-:

```
package com.hem.hadoop.Assignment4;

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;

import com.hem.hadoop.WordCount.WordCountReducer;

public class DriverClass {

  public static void main(String[] args) throws Exception {
    if (args.length != 2) {
      System.err.println("Usage: Filter Records <input path> <output path>");
      System.exit(-1);
    }

       //Job Related Configurations
       Configuration conf = new Configuration();
       Job job = new Job(conf, "Filter Records");
       job.setJarByClass(DriverClass.class);


    // Specify the number of reducer to 2
    job.setNumReduceTasks(1);

    //Provide paths to pick the input file for the job
    FileInputFormat.setInputPaths(job, new Path(args[0]));


    //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);


    //To set the mapper and reducer of this job
    job.setMapperClass(TotalUnitSoldOnida.class);
    job.setReducerClass(TotalUnitSoldReducer.class);

    //Set the combiner
    job.setCombinerClass(TotalUnitSoldReducer.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(IntWritable.class);

    //execute the job
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

Mapper Class-:

```
package com.hem.hadoop.Assignment4;

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 TotalUnitSoldOnida extends Mapper<LongWritable, Text, Text,
IntWritable> {

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

       @Override
       public void map(LongWritable key, Text value, Context context) throws
IOException, InterruptedException {
              String line = value.toString();
              System.out.println("This is output to mapper:" + key.toString());
              String words[] = line.split("\n");
              for (String wordSplit : words) {
                     String tempWord[] = wordSplit.split("\\|");
                     if("Onida".equalsIgnoreCase(tempWord[0])){
                            word.set(tempWord[0]+" "+tempWord[3]);
                            context.write(word, one);
                     }


              }

       }
}
```

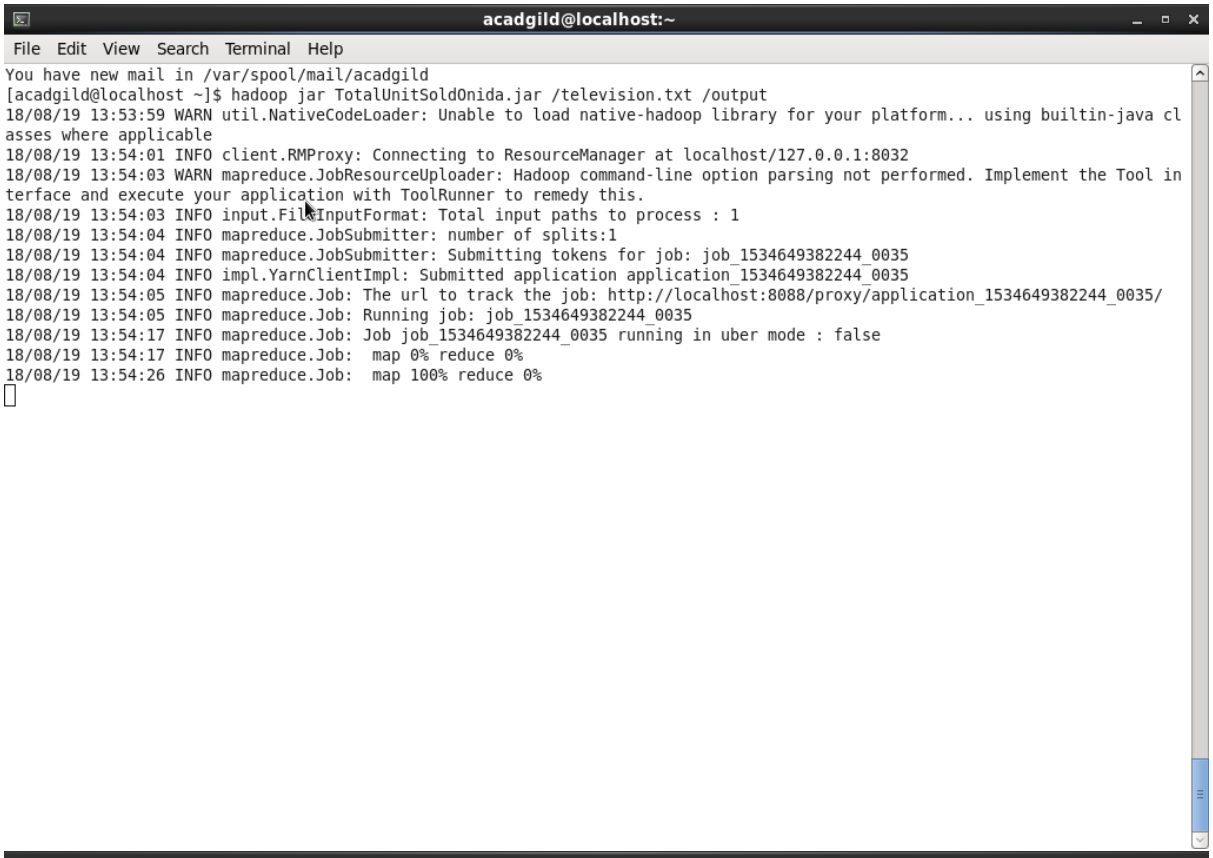
Reducer Class-:

package com.hem.hadoop.Assignment4;

import java.io.IOException;

```
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;



public class TotalUnitSoldReducer
 extends Reducer<Text, IntWritable, Text, IntWritable> {

 @Override
 public void reduce(Text key, Iterable<IntWritable> values,
    Context context)
    throws IOException, InterruptedException {
    System.out.println("From The Reducer=>"+key) ;

    int sum = 0;
    for (IntWritable value : values) {
                    sum+=value.get();
     }
     context.write(key, new IntWritable(sum));
 }
}
```

Output-:


acadgild@localhost:~
File Edit View Search Terminal Help
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ hadoop jar TotalUnitSoldOnida.jar /television.txt /output
18/08/19 13:53:59 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
18/08/19 13:54:01 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/08/19 13:54:03 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool in
terface and execute your application with ToolRunner to remedy this.
18/08/19 13:54:03 INFO input.FileInputFormat: Total input paths to process : 1
18/08/19 13:54:04 INFO mapreduce.JobSubmitter: number of splits:1
18/08/19 13:54:04 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1534649382244_0035
18/08/19 13:54:04 INFO impl.YarnClientImpl: Submitted application application_1534649382244_0035
18/08/19 13:54:05 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1534649382244_0035/
18/08/19 13:54:05 INFO mapreduce.Job: Running job: job_1534649382244_0035
18/08/19 13:54:17 INFO mapreduce.Job: Job job_1534649382244_0035 running in uber mode : false
18/08/19 13:54:17 INFO mapreduce.Job: map 0% reduce 0%
18/08/19 13:54:26 INFO mapreduce.Job: map 100% reduce 0%


acadgild@localhost:~
File Edit View Search Terminal Help
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ hadoop fs -cat /output/part-r-00000
18/08/19 13:54:54 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Onida Kerala   1
Onida Uttar Pradesh   3
[acadgild@localhost ~]$