BIG DATA HADOOP AND SPARK DEVELOPMENT

ASSIGNMENT – 4

Table of Contents:

BIG DATA HADOOP AND SPARK DEVELOPMENT

1. Introduction

In this assignment, the given task is performed and Output of the task is performed and Screenshots are attached.

2. Objective

This assignment consolidates the deeper understanding of the Session – 4 Introduction to MapReduce.

3. Associated Data Files

```
Samsung|Optima|14|Madhya Pradesh|132401|14200 Onida|Lucid|18|Uttar
Pradesh|232401|16200 Akai|Decent|16|Kerala|922401|12200
Lava|Attention|20|Assam|454601|24200
Zen|Super|14|Maharashtra|619082|9200 Samsung|Optima|14|Madhya
Pradesh|132401|14200 Onida|Lucid|18|Uttar Pradesh|232401|16200
Onida|Decent|14|Uttar Pradesh|232401|16200
Onida|NA|16|Kerala|922401|12200 Lava|Attention|20|Assam|454601|24200
Zen|Super|14|Maharashtra|619082|9200 Samsung|Optima|14|Madhya
Pradesh|132401|14200 NA|Lucid|18|Uttar Pradesh|232401|16200
Samsung|Decent|16|Kerala|922401|12200
Lava|Attention|20|Assam|454601|24200
Samsung|Super|14|Maharashtra|619082|9200
Samsung|Super|14|Maharashtra|619082|9200
Samsung|Super|14|Maharashtra|619082|9200
```

4. 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.

- **Task 2**

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

- **Task 3**

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

5. Expected Output

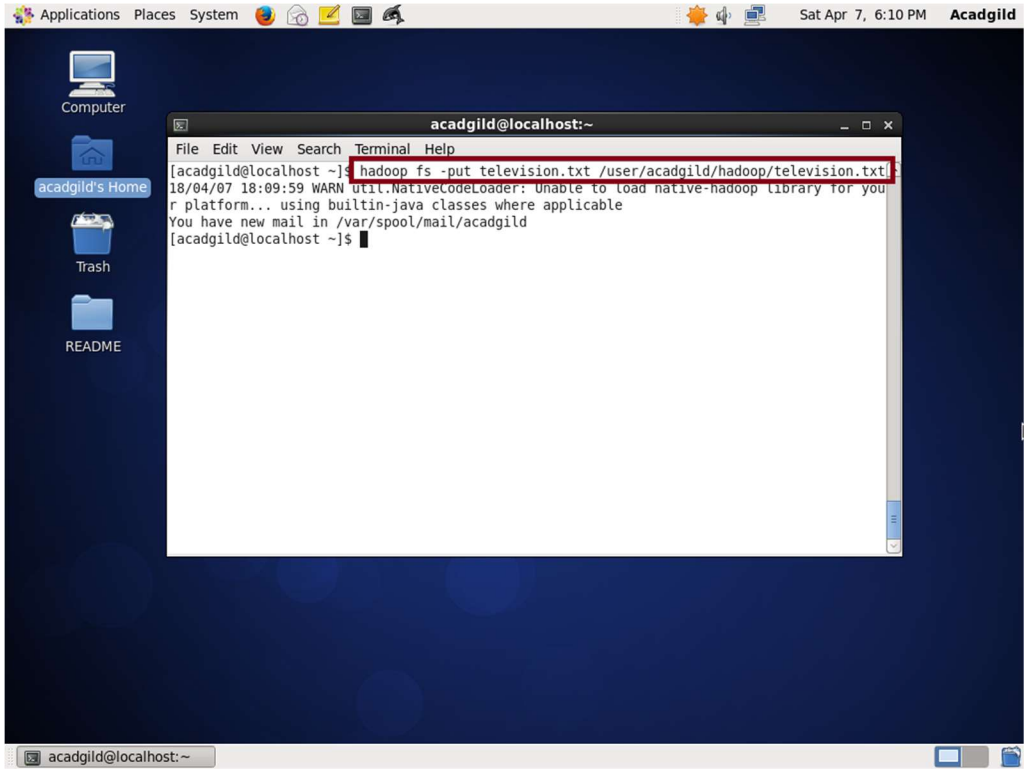
Preparing to perform tasks:

By copying a text file (television.txt) with the size of 312Mb from local to Acadglid VM.

hdfs dfs –put television.txt /user/acadgild/hadoop/television.txt

by using **–put** command the **television.txt** is copied from local to **Acadgild VM**

The following screenshot show the process of copying **television.txt** from local to **Acadgild VM**.



- Task 1

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

  A Map/Reduce program that filters the NA values from the input file.

Removeinvalid.java (Driver class)

```
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
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.InvalidInputException;
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 Removeinvalid {

public static void main(String[] args) throws ClassNotFoundException,
InterruptedException,InvalidInputException, IOException
            {
                  Configuration conf = new Configuration();
                  Job job = new Job(conf, "Remove NA value");
                  job.setJarByClass(Removeinvalid.class);
                  job.setMapperClass(RemoveinvalidMapper.class);

                  // Specify the number of reducer to 0
                  job.setNumReduceTasks(0);

                  //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);

                  //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);
                    }
            }
```

Remove invalid Mapper.java (Mapper program)

```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;


public class RemoveinvalidMapper extends Mapper<LongWritable, Text, Text,
Text>
{
      private Text word = new Text();

      public void map(LongWritable key, Text value, Context context)
                  throws IOException, InterruptedException
                  {
                  String line = value.toString();
                  int count=0;
                  StringTokenizer tokenizer = new StringTokenizer(line,"|");

            while (tokenizer.hasMoreTokens())
            {
                word.set(tokenizer.nextToken());
                if(word.toString().equalsIgnoreCase("NA"))
                {

                  count=count++;
                }
            }
            if(count==0)
            {
            Text t = new Text(line);
            context.write(t,null);
            }
            }
}
```

hadoop [--config conf dir]

jar <jar> run a jar file

by using the above syntax, we can run the jar file

hadoop <hadoop jar file path> <path of the file name> <directory name where the output can be stored>

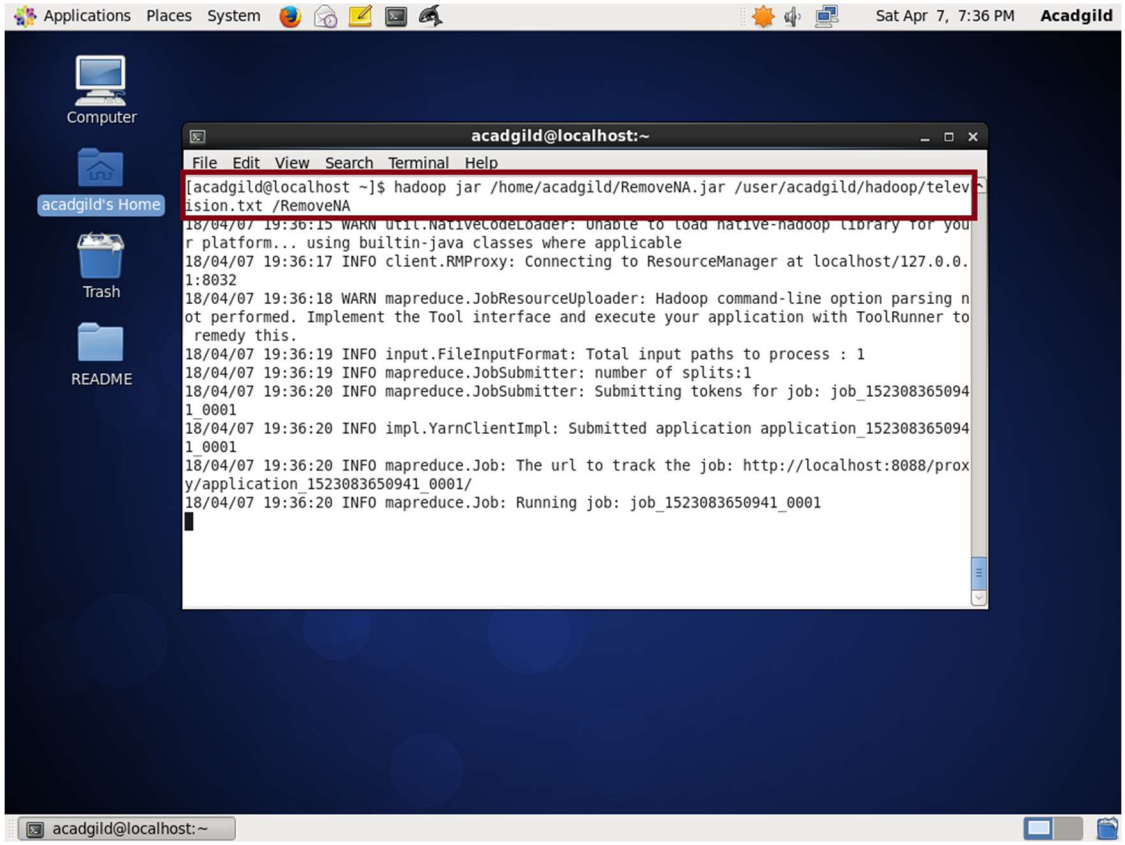
the above is the syntax for the jar file to run and output save in the directory.

To run Map/Reduce program

- jar File path is /home/acadgild/RemoveNA.jar
- Input File path is /user/acadgild/hadoop/television.txt
- Output File path is /RemoveNA

The command which is used here is

hadoop jar /home/acadgild/RemoveNA.jar /user/acadgild/hadoop/television.txt /RemoveNA



Map/Reduce process is performed and output are saved in the RemoveNA

Applications Places System Sat Apr 7, 7:38 PM Acadgild

acadgild@localhost:~

File Edit View Search Terminal Help

```
[acadgild@localhost ~]$ hadoop jar /home/acadgild/RemoveNA.jar /user/acadgild/hado
ision.txt /RemoveNA
18/04/07 19:36:15 WARN util.NativeCodeLoader: Unable to load native-hadoop library
r platform... using builtin-java classes where applicable
18/04/07 19:36:17 INFO client.RMProxy: Connecting to ResourceManager at localhost/
1:8032
18/04/07 19:36:18 WARN mapreduce.JobResourceUploader: Hadoop command-line option p
ot performed. Implement the Tool interface and execute your application with ToolR
 remedy this.
18/04/07 19:36:19 INFO input.FileInputFormat: Total input paths to process : 1
18/04/07 19:36:19 INFO mapreduce.JobSubmitter: number of splits:1
18/04/07 19:36:20 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1523
1_0001
18/04/07 19:36:20 INFO impl.YarnClientImpl: Submitted application application_1523
1_0001
18/04/07 19:36:20 INFO mapreduce.Job: The url to track the job: http://localhost:8
y/application_1523083650941_0001/
18/04/07 19:36:20 INFO mapreduce.Job: Running job: job_1523083650941_0001
18/04/07 19:36:36 INFO mapreduce.Job: Job job_1523083650941_0001 running in uber m
lse
18/04/07 19:36:36 INFO mapreduce.Job:  map 0% reduce 0%
18/04/07 19:36:45 INFO mapreduce.Job:  map 100% reduce 0%
18/04/07 19:36:46 INFO mapreduce.Job: Job job_1523083650941_0001 completed success
18/04/07 19:36:46 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=107367
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=855
                HDFS: Number of bytes written=646
                HDFS: Number of read operations=5
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Data-local map tasks=1
```

acadgild@localhost:~

Applications Places System Sat Apr 7, 7:40 PM Acadgild

acadgild@localhost:~

File Edit View Search Terminal Help

```
18/04/07 19:36:46 INFO mapreduce.Job: Job job_1523083650941_0001 completed success
18/04/07 19:36:46 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=107367
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=855
                HDFS: Number of bytes written=646
                HDFS: Number of read operations=5
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=7030
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=7030
                Total vcore-milliseconds taken by all map tasks=7030
                Total megabyte-milliseconds taken by all map tasks=7198720
        Map-Reduce Framework
                Map input records=18
                Map output records=16
                Input split bytes=122
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=72
                CPU time spent (ms)=730
                Physical memory (bytes) snapshot=92213248
                Virtual memory (bytes) snapshot=2056757248
                Total committed heap usage (bytes)=32571392
        File Input Format Counters
                Bytes Read=733
        File Output Format Counters
                Bytes Written=646
[acadgild@localhost ~]$
```

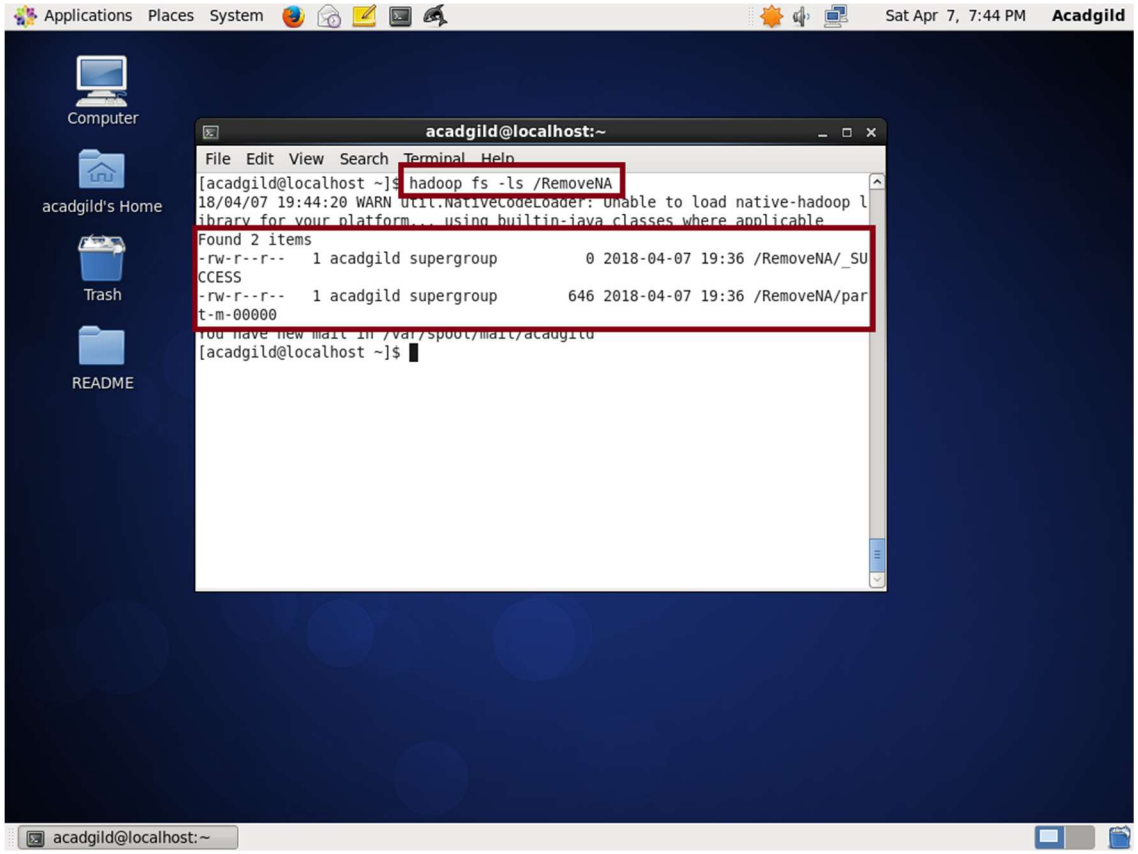
acadgild@localhost:~

The Output is saved in the RemoveNA directory.

By using,

hadoop fs –ls /RemoveNA

All the files are saved in the RemoveNA directory.



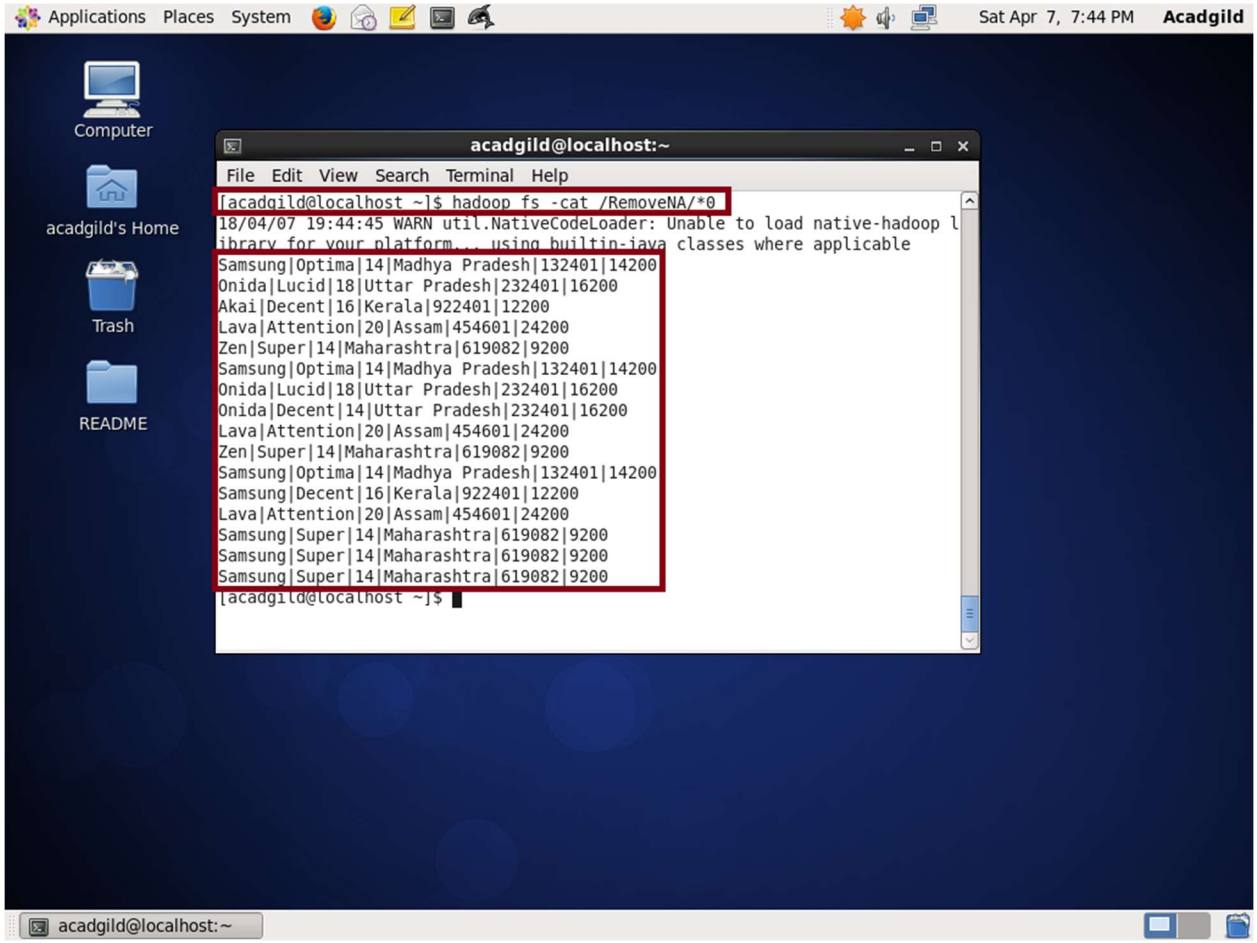
By using,

The following command we can see the output of the processed input text file.

hadoop fs –cat /RemoveNA/*0

Or

hadoop fs –cat /RemoveNA/part-m-00000



- Task 2

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

Totalunitsold.java (driver class)

```
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.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.InvalidInputException;
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 Totalunitsold {
      public static void main(String[] args) throws ClassNotFoundException,
InterruptedException,InvalidInputException, IOException
      {
            Configuration conf = new Configuration();
            @SuppressWarnings("deprecation")
            Job job = new Job(conf, "Remove NA value");
            job.setJarByClass(Totalunitsold.class);
            job.setMapperClass(TotalunitsoldMapper.class);
            job.setReducerClass(TotalunitsoldReducer.class);

            // Specify the number of reducer to 0
            job.setNumReduceTasks(1);

            //Set the combiner
            job.setCombinerClass(TotalunitsoldReducer.class);

            //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);

            //set the input and output format class
            job.setInputFormatClass(TextInputFormat.class);
            job.setOutputFormatClass(TextOutputFormat.class);

            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);

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

TotalunitsoldMapper.java(Mapper)

```
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 TotalunitsoldMapper extends
Mapper<LongWritable,Text,Text,IntWritable>
{
private final static IntWritable one =new IntWritable(1);

public void map(LongWritable key, Text value, Context context)throws
IOException, InterruptedException
{
            String line [] = value.toString().split("\\|");
            Text t1 = new Text(line [0]);
            context.write(t1, one);
}
}
```

TotalunitsoldReducer.java(Reducer)

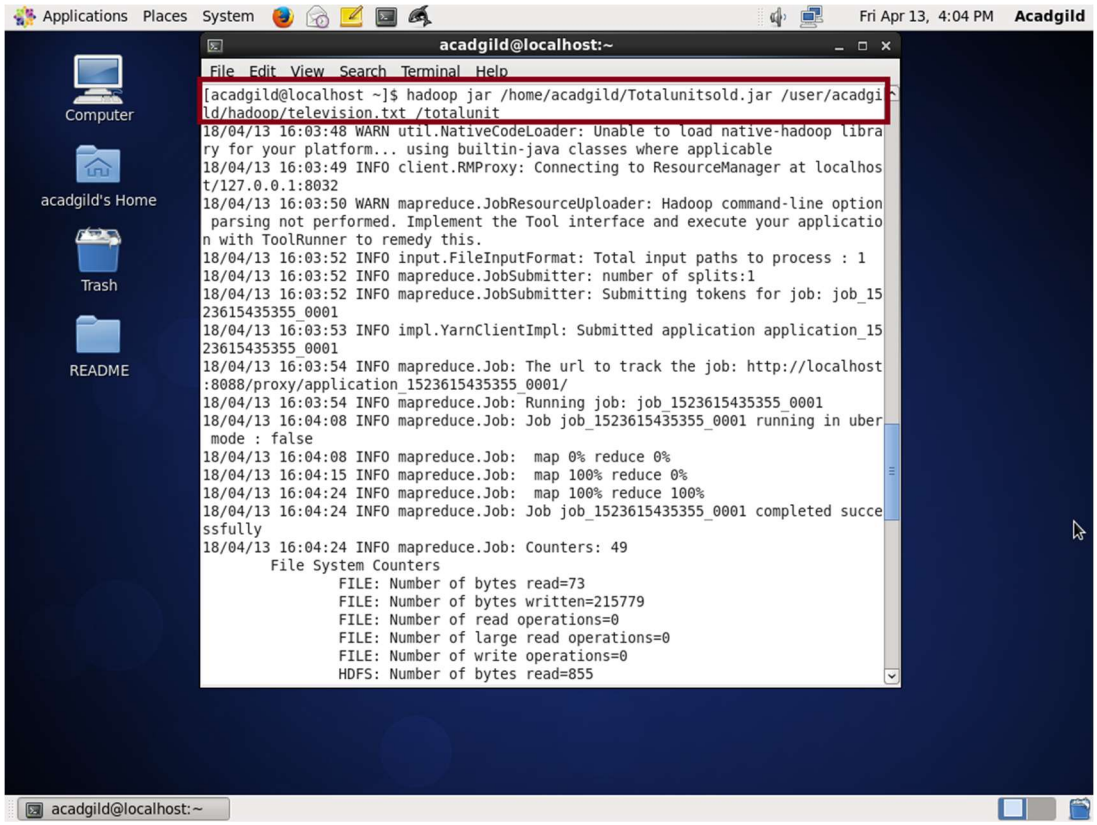
```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class TotalunitsoldReducer extends Reducer<Text,IntWritable, Text,
IntWritable>
{
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));
        }
}
```

To run Map/Reduce program

- jar File path is /home/acadgild/Totalunitsold.jar
- Input File path is /user/acadgild/hadoop/television.txt
- Output File path is /totalunit

The command which is used here is

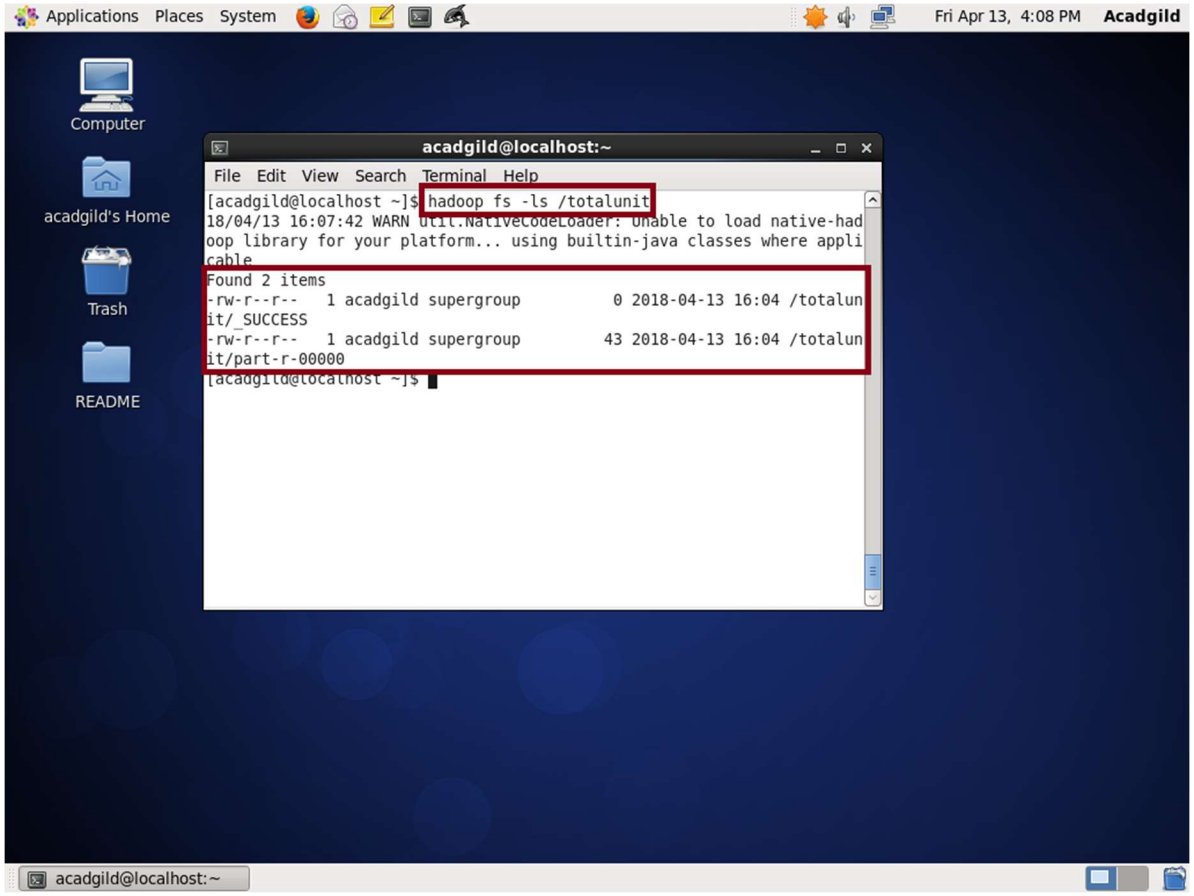
hadoop jar /home/acadgild/Totalunitsold.jar /user/acadgild/hadoop/television.txt /Totalunit



```
                HDFS: Number of bytes written=43
                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)=5302
                Total time spent by all reduces in occupied slots (ms)=5869
                Total time spent by all map tasks (ms)=5302
                Total time spent by all reduce tasks (ms)=5869
                Total vcore-milliseconds taken by all map tasks=5302
                Total vcore-milliseconds taken by all reduce tasks=5869
                Total megabyte-milliseconds taken by all map tasks=5429248
                Total megabyte-milliseconds taken by all reduce tasks=6009856
        Map-Reduce Framework
                Map input records=18
                Map output records=18
                Map output bytes=183
                Map output materialized bytes=73
                Input split bytes=122
                Combine input records=18
                Combine output records=6
                Reduce input groups=6
                Reduce shuffle bytes=73
                Reduce input records=6
                Reduce output records=6
                Spilled Records=12
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=136
```

```
                Map input records=18
                Map output records=18
                Map output bytes=183
                Map output materialized bytes=73
                Input split bytes=122
                Combine input records=18
                Combine output records=6
                Reduce input groups=6
                Reduce shuffle bytes=73
                Reduce input records=6
                Reduce output records=6
                Spilled Records=12
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=136
                CPU time spent (ms)=1590
                Physical memory (bytes) snapshot=300367872
                Virtual memory (bytes) snapshot=4118200320
                Total committed heap usage (bytes)=170004480
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=733
        File Output Format Counters
                Bytes Written=43
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$
```

Map/Reduce process is performed and output are saved in the totalunit



The Output is saved in the totalunit directory.

By using,

hadoop fs –ls /totalunit

All the files are saved in the totalunit directory.

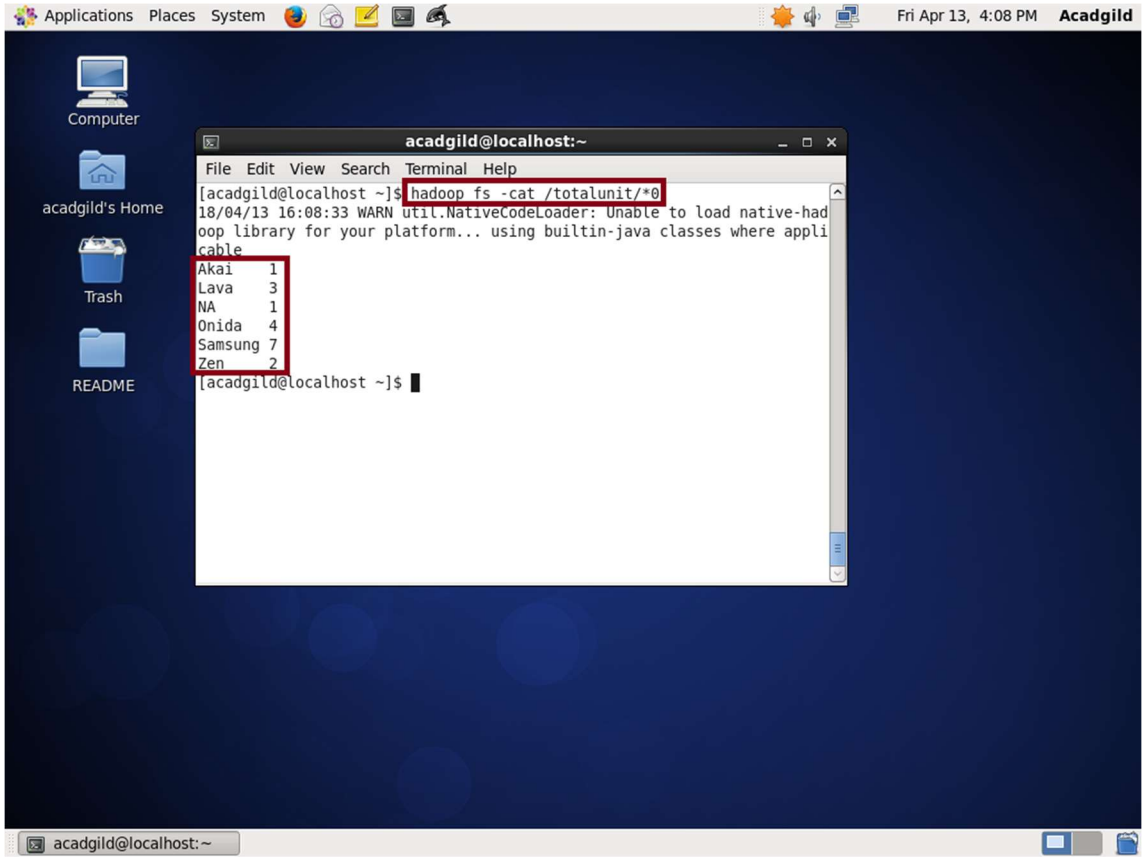
By using,

The following command we can see the output of the processed input text file.

hadoop fs –cat /totalunit/*0

Or

hadoop fs –cat /totalunit/part-m-00000



- Task 3

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

Onida.java (Driver class program)

```
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.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.InvalidInputException;
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 Onida {
      public static void main(String[] args) throws ClassNotFoundException,
InterruptedException,InvalidInputException, IOException
      {
            Configuration conf = new Configuration();
            @SuppressWarnings("deprecation")
            Job job = new Job(conf, "Remove NA value");
            job.setJarByClass(Onida.class);
            job.setMapperClass(OnidaMapper.class);
            job.setReducerClass(OnidaReducer.class);

            //Specify the number of reducer to 0
            job.setNumReduceTasks(1);

            //Set the combiner
            job.setCombinerClass(OnidaReducer.class);

            //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);

            //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);
  }}
```

OnidaMapper.java (Mapper class program)

```
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>
{
private final static IntWritable one = new IntWritable(1);
private final static IntWritable zero = new IntWritable(0);

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

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

            if(line[0].equalsIgnoreCase("ONIDA"))
          {
            Text t1 = new Text(line[3]);
            context.write(t1,one);
          }
            else
            {
                  Text t2 = new Text(line[3]);
                  context.write(t2,zero);
            }
}
}
```

OnidaReducer.java (Reducer class program)

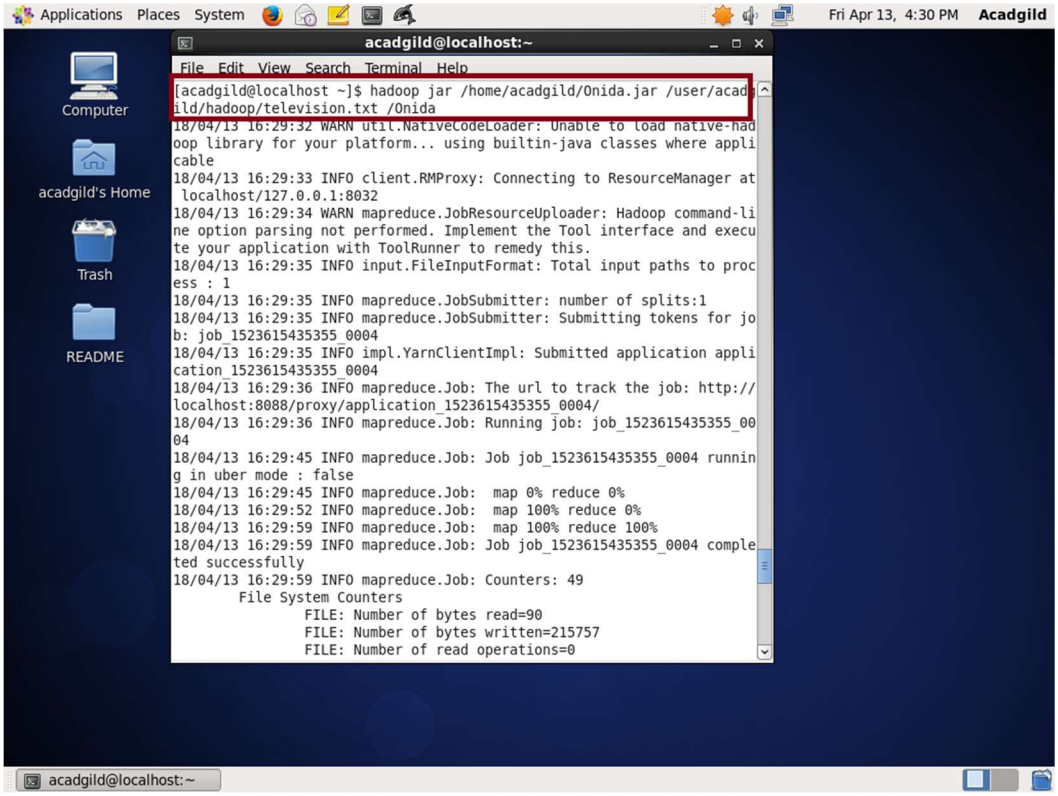
```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class OnidaReducer extends Reducer<Text,IntWritable, Text,
IntWritable>
{
 public void reduce(Text 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));
        }
}
```

To run Map/Reduce program

- jar File path is /home/acadgild/Onida.jar
- Input File path is /user/acadgild/hadoop/television.txt
- Output File path is /Onida

The command which is used here is

hadoop jar /home/acadgild/Onida.jar /user/acadgild/hadoop/television.txt /Onida



```
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=855
                HDFS: Number of bytes written=64
                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)=461
5
                Total time spent by all reduces in occupied slots (ms)=
5260
                Total time spent by all map tasks (ms)=4615
                Total time spent by all reduce tasks (ms)=5260
                Total vcore-milliseconds taken by all map tasks=4615
                Total vcore-milliseconds taken by all reduce tasks=5260
                Total megabyte-milliseconds taken by all map tasks=4725
760
                Total megabyte-milliseconds taken by all reduce tasks=5
386240
        Map-Reduce Framework
                Map input records=18
                Map output records=18
                Map output bytes=272
                Map output materialized bytes=90
                Input split bytes=122
                Combine input records=18
                Combine output records=5
                Reduce input groups=5
```

```
        Map-Reduce Framework
                Map input records=18
                Map output records=18
                Map output bytes=272
                Map output materialized bytes=90
                Input split bytes=122
                Combine input records=18
                Combine output records=5
                Reduce input groups=5
                Reduce shuffle bytes=90
                Reduce input records=5
                Reduce output records=5
                Spilled Records=10
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=142
                CPU time spent (ms)=1400
                Physical memory (bytes) snapshot=295604224
                Virtual memory (bytes) snapshot=4118204416
                Total committed heap usage (bytes)=170004480
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=723
        File Output Format Counters
                Bytes Written=64
[acadgild@localhost ~]$
```

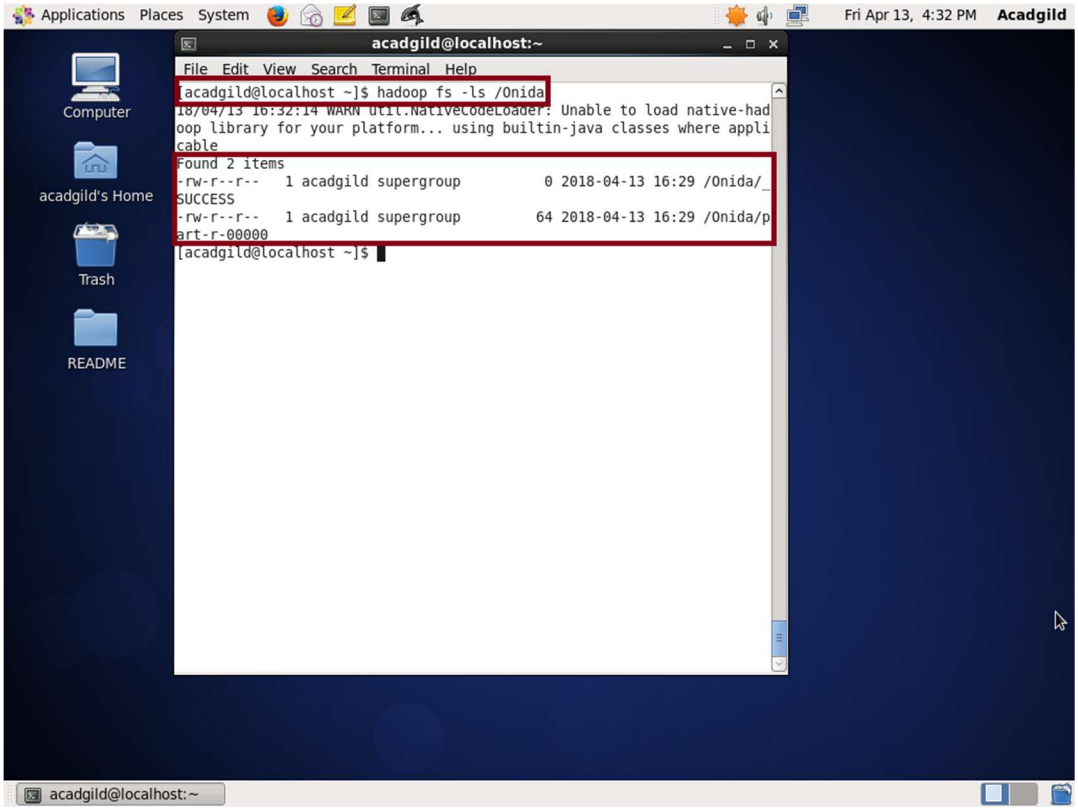
Map/Reduce process is performed and output are saved in the Onida

The Output is saved in the Onida directory.

By using,

hadoop fs –ls /Onida

All the files are saved in the Onida directory



By using,

The following command we can see the output of the processed input text file.

hadoop fs –cat /Onida/*0

Or

hadoop fs –cat /Onida/part-m-00000