Data Hadoop & Spark Training - ACADGILD

Assignment 3.2

18-Nov-17

BIG DATA

DATA SET DESCRIPTION

Name of the file - television.txt

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

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.

In this assignment, I have only taken Mapper because we just have to filter records containing "NA".

Mapper Code: Task1Mapper.java

```
/*
* All the comments in the program are highlighted in Green.
* @author Sahil Khurana <sahilkhurana369@gmail.com
*/
```

```
// Package Declared
package Assignment_3;
```

```
import java.io.IOException;
```

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

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

```
import org.apache.hadoop.mapreduce.Mapper;
```

```
// class is extended to have the arguments keyIn as LongWritable and ValueIn as Text and KeyOut as Text and ValueOut as Text.
public class Task1Mapper extends Mapper<LongWritable,Text,Text,Text> {
```

```
// overriding the map method which will run one time for every line.
@Override
public void map(LongWritable key, Text value, Context context)
```

```
// storing the line in a string variable
String line=value.toString();
```

```
// splitting the line by using comma "|" delimiter and storing the values in a String Array so that all the columns in a row are stored in the string array.
```

```
String[] words=line.split("\\|");
// string array declared with position 0
String Company_Name=words[0];
// string array declared with position 1
String Product_Name=words[1];
// if loop condition with string array not equal to NA
if (!(Company_Name.equals("NA")|| Product_Name.equals("NA")))
// obtaining Text as value to the context.
context.write(value, new Text());
    }  // map class closed
}  // class Task1Mapper closed
```

Driver Code: Task1.java

```
/*
* All the comments in the program are highlighted in Green.
* @author Sahil Khurana <sahilkhurana369@gmail.com
*/
// Package Declared
package Assignment_3;
// Import the Configuration of system parameters.
import org.apache.hadoop.conf.Configuration;
// used to Names a file or directory in a AbstractFileSystem for hdfs.
import org.apache.hadoop.fs.Path;
// This class stores text using standard UTF8 encoding.
import org.apache.hadoop.io.Text;
// The job submitter's view of the Job
import org.apache.hadoop.mapreduce.Job;
// FilterInputFormat is a convenience class that wraps InputFormat.
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
// FilterInputFormat is a convenience class that wraps OutputFormat.
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
// Class declared
public class Task1 {
// The @SuppressWarnings annotation disables certain compiler warnings. In this case, the warning about deprecated code ("deprecation")
```

```
@SuppressWarnings({ "deprecation" })
// main method started
public static void main(String[] args) throws Exception {
// Create a configuration object for the job
Configuration conf = new Configuration();
// create new object named job
Job job = new Job(conf, "Task1");
// Set a name of the Job
job.setJobName("Assignment_3.2_Task1");
// Set input directories using command line arguments,arg[0] = name of input directory on HDFS
FileInputFormat.addInputPath(job, new Path(args[0]));
// Set input directories using command line arguments,arg[1] = name of output directory on HDFS
FileOutputFormat.setOutputPath(job, new Path(args[1]));
// Specify names of Mapper Class
job.setMapperClass(Task1Mapper.class);
// Sets reducer tasks to 0
job.setNumReduceTasks(0);
// Specify data type of output key
job.setOutputKeyClass(Text.class);
// Specify data type of output value
job.setOutputValueClass(Text.class);
// Submit the job, then poll for progress until the job is complete
job.waitForCompletion(true);
  }  // main method closed
} // class Task1 closed
```

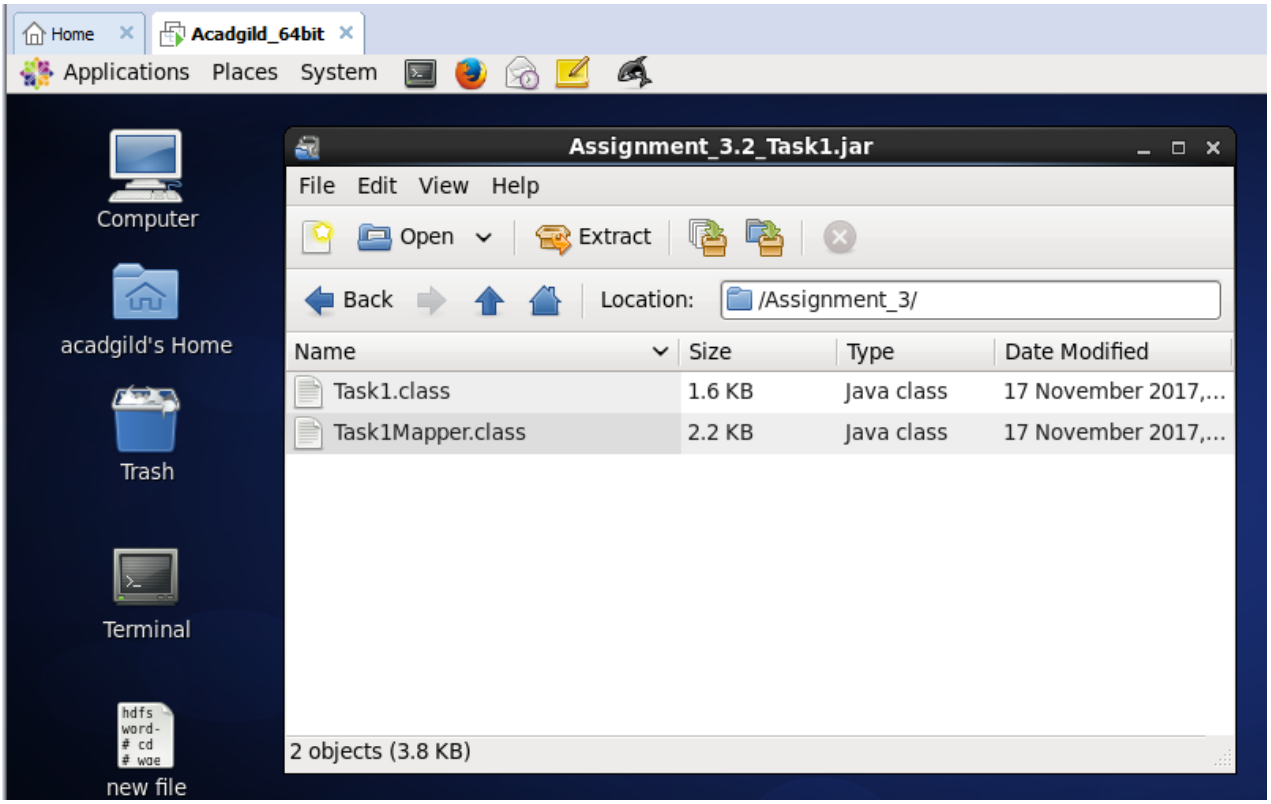
Steps to Run Mapreduce program on command prompt

Step 1:

Create a Jar, which will contain

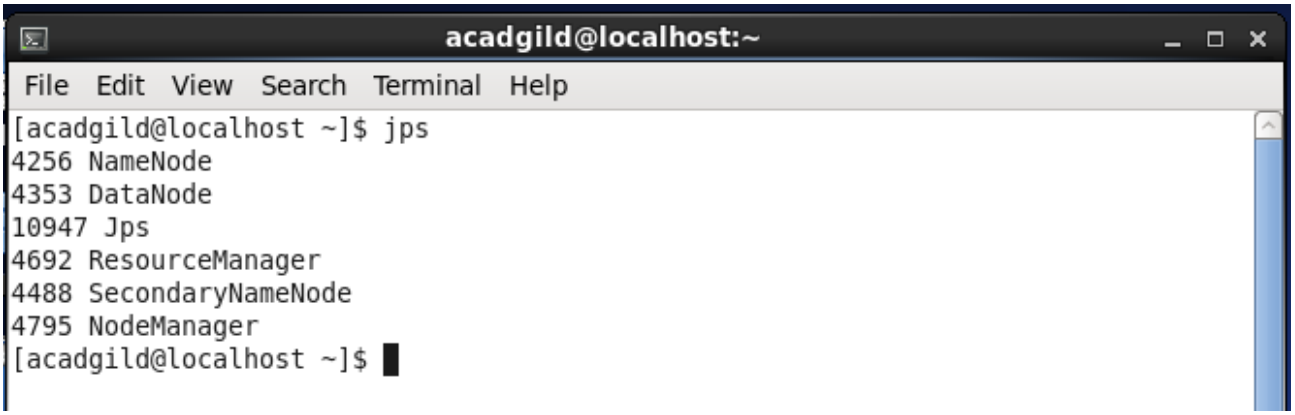
Driver Code: Task1.java

Mapper Code: Task1Mapper.java



Step 2:

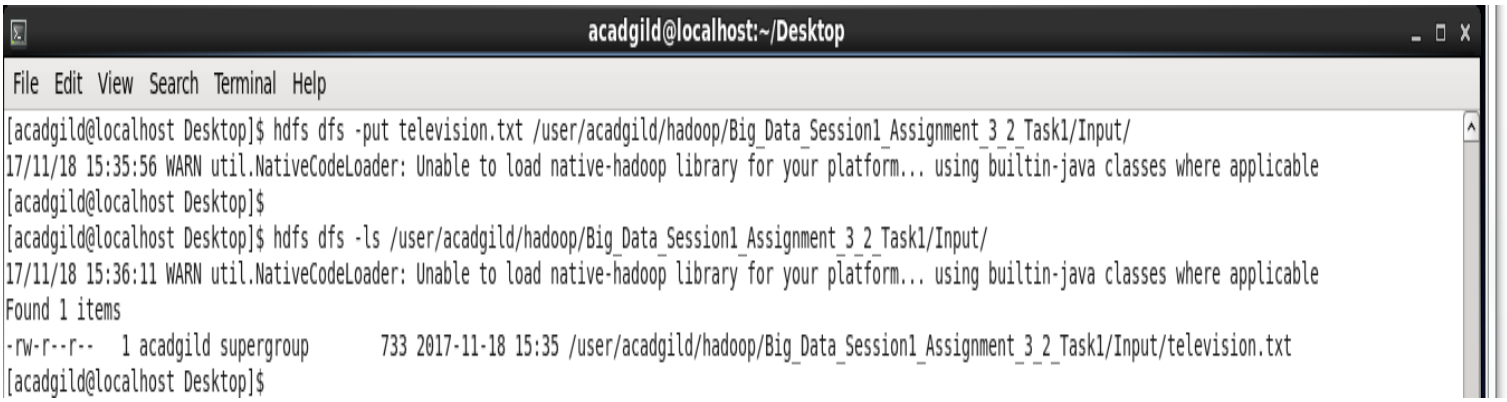
Check the all the Hadoop services are running or not by typing jps command



Step 3:

Transfer the file “television.txt” to HDFS filesystem.

hdfs dfs -put television.txt /user/acadgild/hadoop/Big_Data_Session1_Assignment_3_2_Task1/Input/



Step 4:

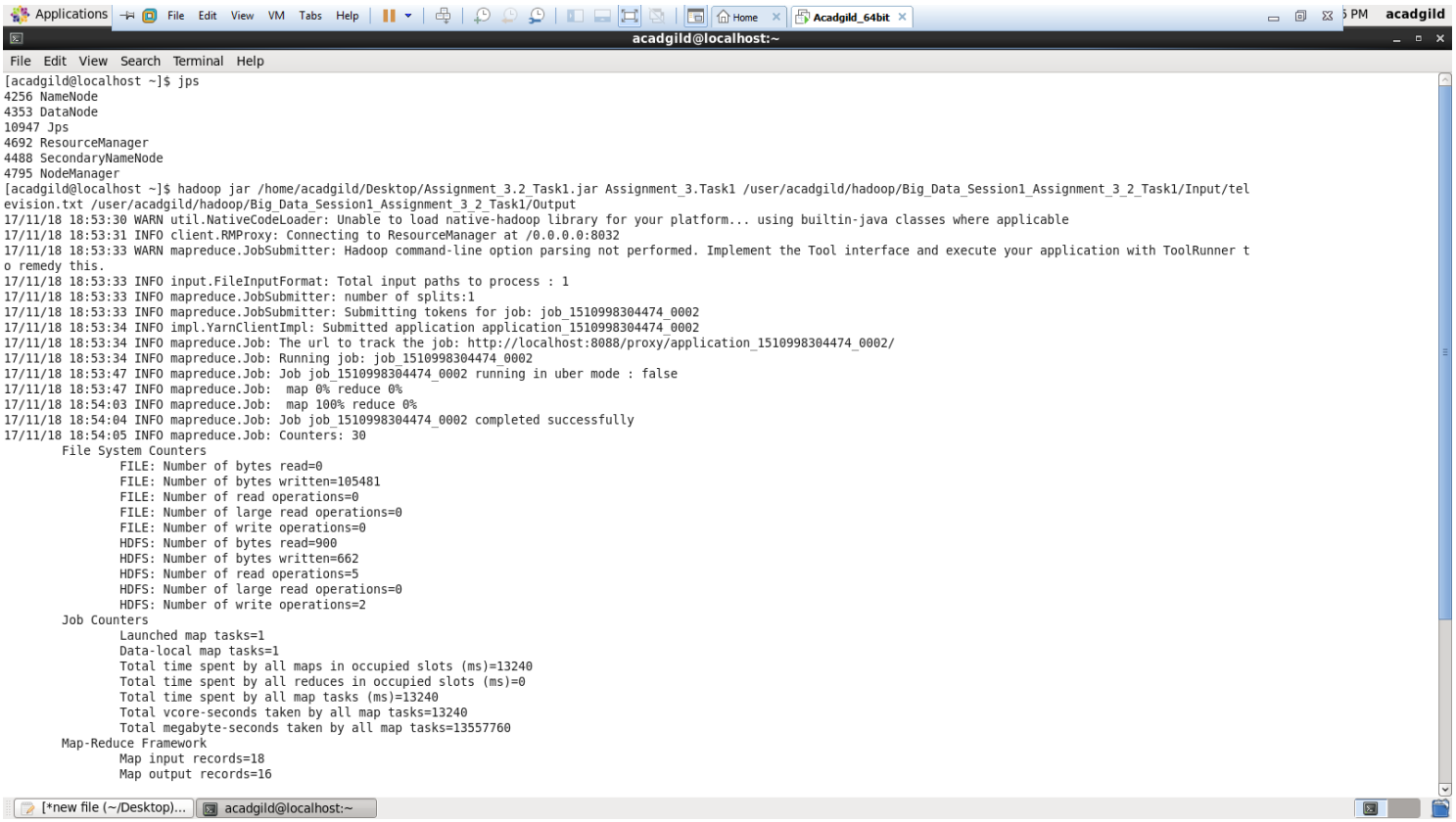
Run the MapReduce Job

Generic Command:-

hadoop jar <JAR file Path > <PackageName.MainClass> <Inputfile> <outputDir>

In case of Assignment 3.2:-

hadoop jar /home/acadgild/Desktop/Assignment_3.2_Task1.jar Assignment_3.Task1 /user/acadgild/hadoop/Big_Data_Session1_Assignment_3_2_Task1/Input/television.txt /user/acadgild/hadoop/Big_Data_Session1_Assignment_3_2_Task1/Output



```
[acadgild@localhost ~]$ hadoop jar /home/acadgild/Desktop/Assignment_3.2_Task1.jar Assignment_3.Task1
/user/acadgild/hadoop/Big_Data_Session1_Assignment_3_2_Task1/Input/television.txt
/user/acadgild/hadoop/Big_Data_Session1_Assignment_3_2_Task1/Output

17/11/18 18:53:30 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where
applicable

17/11/18 18:53:31 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032

17/11/18 18:53:33 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and
execute your application with ToolRunner to remedy this.

17/11/18 18:53:33 INFO input.FileInputFormat: Total input paths to process : 1

17/11/18 18:53:33 INFO mapreduce.JobSubmitter: number of splits:1

17/11/18 18:53:33 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1510998304474_0002

17/11/18 18:53:34 INFO impl.YarnClientImpl: Submitted application application_1510998304474_0002

17/11/18 18:53:34 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1510998304474_0002/

17/11/18 18:53:34 INFO mapreduce.Job: Running job: job_1510998304474_0002

17/11/18 18:53:47 INFO mapreduce.Job: Job job_1510998304474_0002 running in uber mode : false

17/11/18 18:53:47 INFO mapreduce.Job:  map 0% reduce 0%

17/11/18 18:54:03 INFO mapreduce.Job:  map 100% reduce 0%

17/11/18 18:54:04 INFO mapreduce.Job: Job job_1510998304474_0002 completed successfully

17/11/18 18:54:05 INFO mapreduce.Job: Counters: 30

        File System Counters

                FILE: Number of bytes read=0

                FILE: Number of bytes written=105481

                FILE: Number of read operations=0

                FILE: Number of large read operations=0

                FILE: Number of write operations=0

                HDFS: Number of bytes read=900

                HDFS: Number of bytes written=662

                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)=13240

                Total time spent by all reduces in occupied slots (ms)=0

                Total time spent by all map tasks (ms)=13240

                Total vcore-seconds taken by all map tasks=13240

                Total megabyte-seconds taken by all map tasks=13557760

        Map-Reduce Framework

                Map input records=18
```

Map output records=16

Input split bytes=167

Spilled Records=0

Failed Shuffles=0

Merged Map outputs=0

GC time elapsed (ms)=1251

CPU time spent (ms)=510

Physical memory (bytes) snapshot=92958720

Virtual memory (bytes) snapshot=2055663616

Total committed heap usage (bytes)=30474240

File Input Format Counters

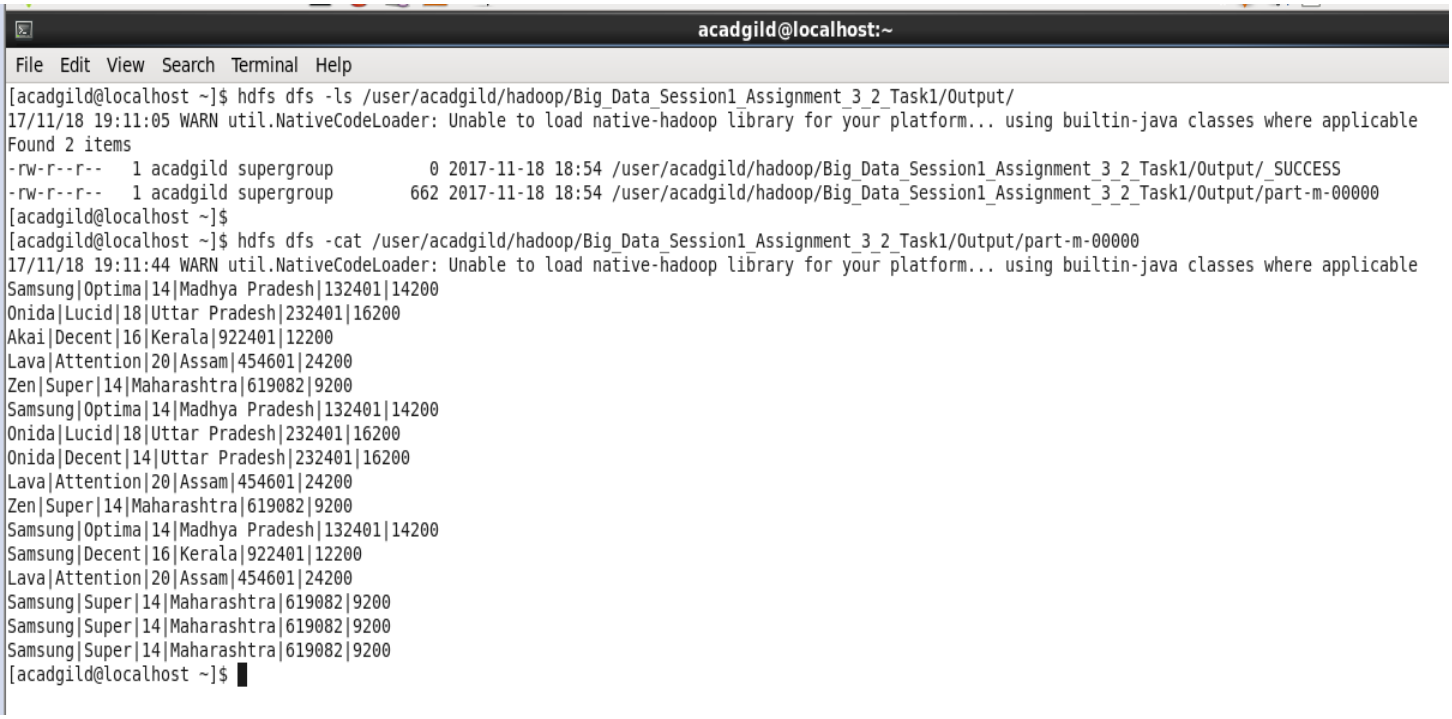
Bytes Read=733

File Output Format Counters

Bytes Written=662

Step 5:

After execution, the result will be stored on HDFS location:-
/user/acadgild/hadoop/Big_Data_Session1_Assignment_3_2_Task1/Output



"NA" is removed from the data set

part-m-00000

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

Lava|Attention|20|Assam|454601|24200

Zen|Super|14|Maharashtra|619082|9200

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

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

Results can also be seen via web interface as-