

Music Data

Data Set

```
-----  
111115|222|0|1|0  
111113|225|1|0|0  
111117|223|0|1|1  
111115|225|1|0|0
```

Dataset is sample data of songs heard by users on an online streaming platform. The Description of data set attached in musicdata.txt as follows:

- 1st column – User Id
- 2nd Column – Track Id
- 3rd Column – Songs Share status (1 for shared, 0 for not shared)
- 4th Column – Listening Platform (Radio or Web – 0 for radio, 1 for web)
- 5th Column – Song Listening Status (0 for skipped, 1 for fully heard)

We will process the above Music Data from Hadoop Map Reduce.

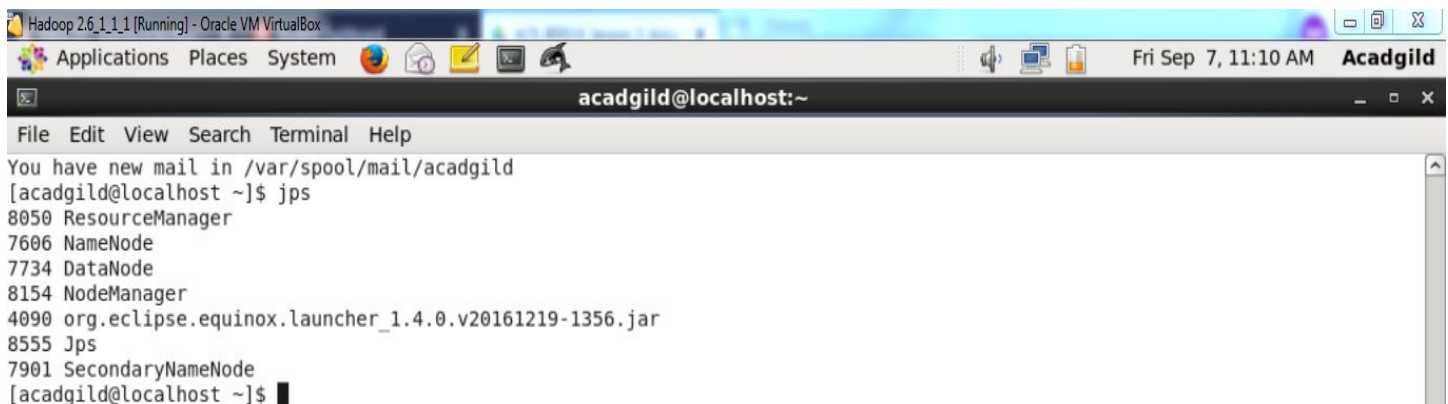
1. First start hadoop daemons in Virtual Machine. The command to start daemons are as follows:

\$start-all.sh

```
[acadgild@localhost ~]$ start-all.sh  
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh  
18/09/07 11:06:10 WARN util.NativeCodeLoader: Unable to load native-hadoop libra  
ry for your platform... using builtin-java classes where applicable  
Starting namenodes on [localhost]  
localhost: starting namenode, logging to /home/acadgild/install/hadoop/hadoop-2.  
6.5/logs/hadoop-acadgild-namenode-localhost.localdomain.out  
localhost: starting datanode, logging to /home/acadgild/install/hadoop/hadoop-2.  
6.5/logs/hadoop-acadgild-datanode-localhost.localdomain.out  
Starting secondary namenodes [0.0.0.0]  
0.0.0.0: starting secondarynamenode, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/hadoop-acadgild-secondaryname  
node-localhost.localdomain.out  
18/09/07 11:06:32 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl  
asses where applicable  
starting yarn daemons  
starting resourcemanager, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/yarn-acadgild-resourcemanager-localhost.  
localdomain.out  
localhost: starting nodemanager, logging to /home/acadgild/install/hadoop/hadoop-2.6.5/logs/yarn-acadgild-nodemanager-localhost.localdomain.out  
You have new mail in /var/spool/mail/acadgild  
[acadgild@localhost ~]$
```

2. We can verify the started daemons with **jps** command, as follows:

\$jps



```
Hadoop 2.6.1.1_1 [Running] - Oracle VM VirtualBox  
Applications Places System  
Fri Sep 7, 11:10 AM Acadgild  
acadgild@localhost:~  
File Edit View Search Terminal Help  
You have new mail in /var/spool/mail/acadgild  
[acadgild@localhost ~]$ jps  
8050 ResourceManager  
7606 NameNode  
7734 DataNode  
8154 NodeManager  
4090 org.eclipse.equinox.launcher_1.4.0.v20161219-1356.jar  
8555 Jps  
7901 SecondaryNameNode  
[acadgild@localhost ~]$
```

Task 1: Find the number of unique listeners in the data set.

1. For this task to do, we need to write three Java source files, *MusicData.java*, *MusicDataMapper.java* and *MusicDataReducer.java*.
2. Here *MusicData.java* is a Driver class which reads the input and calls the *MusicDataMapper.java* methods and *MusicDataReducer.java* methods.
3. *MusicDataMapper.java* implements the Mapper functions of the Map Reduce process.
4. *MusicDataReducer.java* implements the Reducer functions of the Map Reduce process

1. The Source code of the *MusicData.java* is as follows:

```
package MusicData;
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 MusicData {

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

        //Job Related Configurations
        Configuration conf = new Configuration();
        //Job job = new Job.getInstance(conf, "My Word Count with combiner");
        Job job = Job.getInstance(conf, "My Word Count with combiner");
        job.setJarByClass(MusicData.class);

        // Specify the number of reducer to 1
        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(MusicDataMapper.class);
        job.setReducerClass(MusicDataReducer.class);

        //Set the combiner
        job.setCombinerClass(MusicDataReducer.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);
}
}

```

2. The source code of the *MusicDataMapper.java* is as follows:

```

package MusicData;
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;
import java.util.*;

public class MusicDataMapper
    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();
        StringTokenizer tokenizer = new StringTokenizer(line);
        word.set(tokenizer.nextToken("|"));
        context.write(word, one);
    }
}

```

3. The *MusicDataReducer.java* source code is as follows:

```

package MusicData;
import java.io.IOException;

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

public class MusicDataReducer
    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);
```

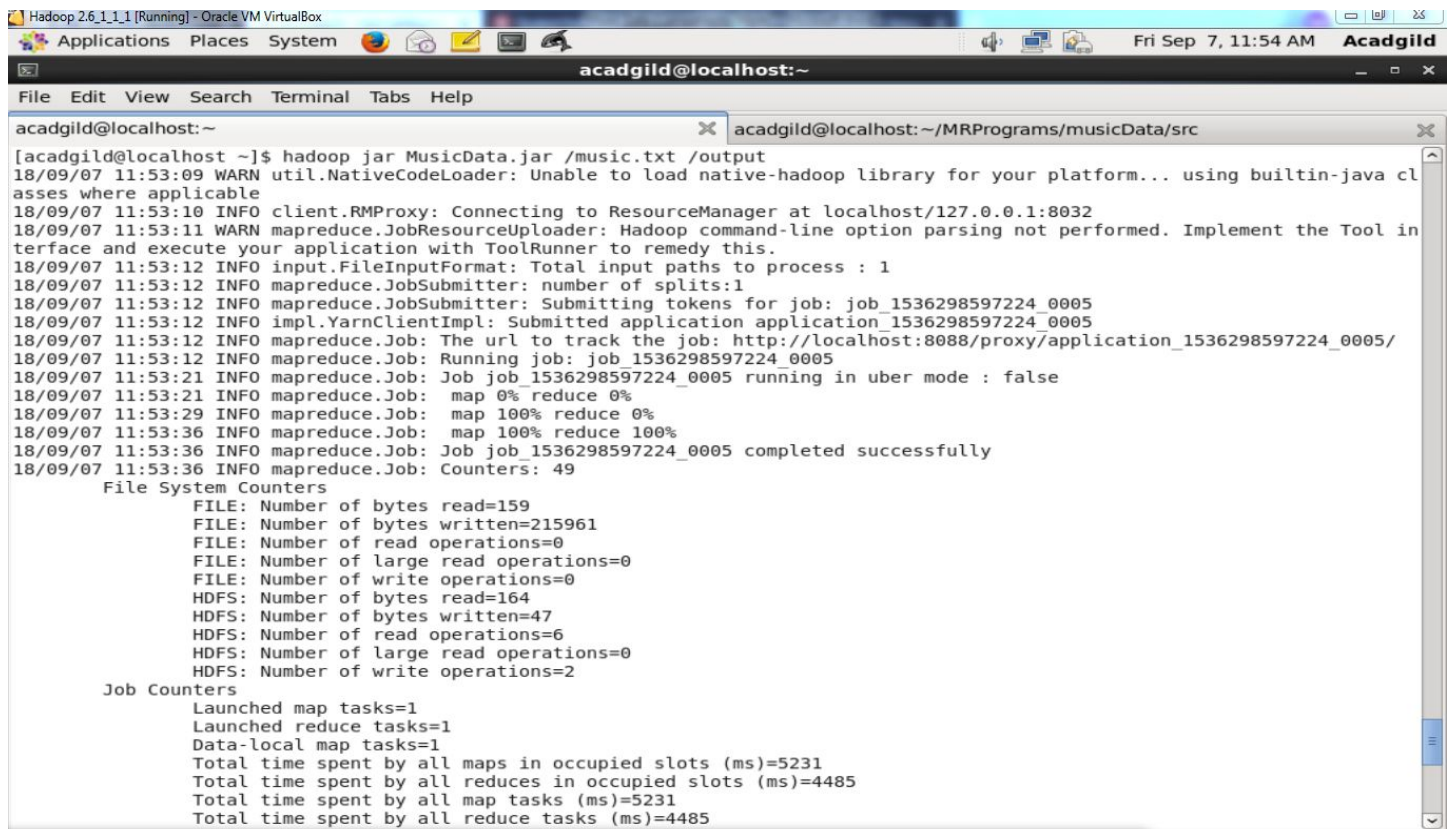
```
int sum = 0;  
int unique = 0;
```

```
for (IntWritable value : values) {  
    sum += value.get();  
    unique += 1;  
}
```

```
key.set("Number of unique listeners in the data set: ");  
context.write(key, new IntWritable(unique));  
}
```

```
}
```

4. Create a jar file **MusicData.jar** with the above source files **MusicData.java**, **MusicDataMapper.java** and **MusicDataReducer.java**.
5. Run the jar file with the input data as **/music.txt** and output will be present in the **/output** directory.
\$hadoop jar MusicData.jar /music.txt /output



```
acacgild@localhost:~  
[acacgild@localhost ~]$ hadoop jar MusicData.jar /music.txt /output  
18/09/07 11:53:09 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl  
asses where applicable  
18/09/07 11:53:10 INFO client.RMPProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032  
18/09/07 11:53:11 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool in  
terface and execute your application with ToolRunner to remedy this.  
18/09/07 11:53:12 INFO input.FileInputFormat: Total input paths to process : 1  
18/09/07 11:53:12 INFO mapreduce.JobSubmitter: number of splits:1  
18/09/07 11:53:12 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1536298597224_0005  
18/09/07 11:53:12 INFO impl.YarnClientImpl: Submitted application application_1536298597224_0005  
18/09/07 11:53:12 INFO mapreduce.Job: The url to track the job: http://localhost:8080/proxy/application_1536298597224_0005/  
18/09/07 11:53:12 INFO mapreduce.Job: Running job: job_1536298597224_0005  
18/09/07 11:53:21 INFO mapreduce.Job: Job job_1536298597224_0005 running in uber mode : false  
18/09/07 11:53:21 INFO mapreduce.Job: map 0% reduce 0%  
18/09/07 11:53:29 INFO mapreduce.Job: map 100% reduce 0%  
18/09/07 11:53:36 INFO mapreduce.Job: map 100% reduce 100%  
18/09/07 11:53:36 INFO mapreduce.Job: Job job_1536298597224_0005 completed successfully  
18/09/07 11:53:36 INFO mapreduce.Job: Counters: 49  
File System Counters  
FILE: Number of bytes read=159  
FILE: Number of bytes written=215961  
FILE: Number of read operations=0  
FILE: Number of large read operations=0  
FILE: Number of write operations=0  
HDFS: Number of bytes read=164  
HDFS: Number of bytes written=47  
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)=5231  
Total time spent by all reduces in occupied slots (ms)=4485  
Total time spent by all map tasks (ms)=5231  
Total time spent by all reduce tasks (ms)=4485
```

```

Hadoop 2.6.1_1.1 [Running] - Oracle VM VirtualBox
Applications Places System
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~ acadgild@localhost:~/MRPrograms/musicData/src
Total vcore-milliseconds taken by all map tasks=5231
Total vcore-milliseconds taken by all reduce tasks=4485
Total megabyte-milliseconds taken by all map tasks=5356544
Total megabyte-milliseconds taken by all reduce tasks=4592640
Map-Reduce Framework
  Map input records=4
  Map output records=4
  Map output bytes=44
  Map output materialized bytes=159
  Input split bytes=96
  Combine input records=4
  Combine output records=3
  Reduce input groups=1
  Reduce shuffle bytes=159
  Reduce input records=3
  Reduce output records=1
  Spilled Records=6
  Shuffled Maps =1
  Failed Shuffles=0
  Merged Map outputs=1
  GC time elapsed (ms)=187
  CPU time spent (ms)=2180
  Physical memory (bytes) snapshot=424574976
  Virtual memory (bytes) snapshot=4165435392
  Total committed heap usage (bytes)=302514176
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=68
File Output Format Counters
  Bytes Written=47
[acadgild@localhost ~]$

```

6. Now we will verify the **/output** directory for Map Reduce process status and output.

\$hadoop fs -ls /output

```

Hadoop 2.6.1_1.1 [Running] - Oracle VM VirtualBox
Applications Places System
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~ acadgild@localhost:~/MRPrograms/musicData/src
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ hadoop fs -ls /output
18/09/07 11:59:58 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Found 2 items
-rw-r--r-- 1 acadgild supergroup 0 2018-09-07 11:53 /output/_SUCCESS
-rw-r--r-- 1 acadgild supergroup 47 2018-09-07 11:53 /output/part-r-00000
[acadgild@localhost ~]$

```

7. From the **_SUCCESS** flag we can understand that the process is success. And the output is present in the **part-r-00000** file.

8. Open the file **part-r-00000** as: ***\$hadoop fs -cat /output/part-r-00000***

```

Hadoop 2.6.1_1.1 [Running] - Oracle VM VirtualBox
Applications Places System
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~ acadgild@localhost:~/MRPrograms/musicData/src
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ hadoop fs -cat /output/part-r-00000
18/09/07 14:08:14 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Number of unique listeners in the data set: 3
[acadgild@localhost ~]$

```

9. We can observe that the message “**Number of unique listeners in the data set: 3**” represents the number of listeners in the given Musig data set.

Task 2: What are the number of times a song was heard fully.

1. For the task 2, we no need to change the **MusicData.java** driver source code.
2. But we need to change the source code of the **MusicDataMapper.java** source code. The changes are as follows:

```
for(int i = 0; i < 4; i++) {  
    tokenizer.nextToken("|");  
}  
String str = tokenizer.nextToken();
```

```
if(str.equals("1")) {  
    word.set(str);  
    context.write(word, one);  
}
```

3. Other wise we can use the following **MusicDataMapper.java** with above changes.

```
package MusicData;  
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;  
import java.util.*;
```

```
public class MusicDataMapper  
    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();  
    StringTokenizer tokenizer = new StringTokenizer(line);
```

```
    for(int i = 0; i < 4; i++) {  
        tokenizer.nextToken("|");  
    }  
    String str = tokenizer.nextToken();
```

```
    if(str.equals("1")) {  
        word.set(str);  
        context.write(word, one);  
    }  
}
```

4. The **MusicDataReducer.java** also requires some changes. The changes are as follows:

```
key.set("Number of times a song was heard fully: ");
```

5. Other wise we can use the following **MusicDataReducer.java** with the above changes:

```
package MusicData;  
import java.io.IOException;
```

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



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

```
public class MusicDataReducer  
    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();
```

```
        }
```

```
        key.set("Number of times a song was heard fully: ");
```

```
        context.write(key, new IntWritable(sum));
```

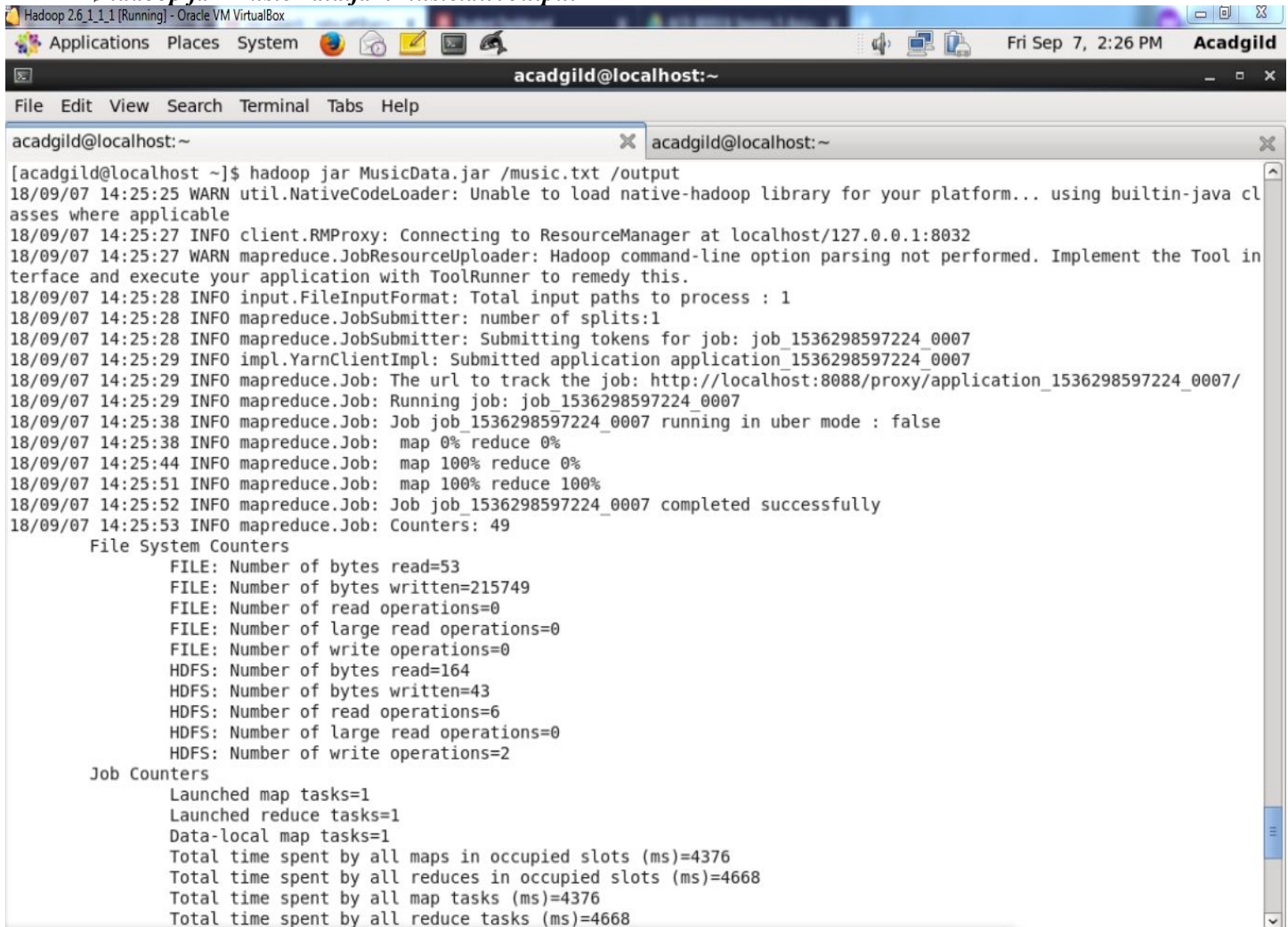
```
    }
```

```
}
```

6. Now create a jar file **MusicData.jar** with the above three source files, **MusicData.java**, **MusicDataMapper.java** and **MusicDataReducer.java**.

7. Run the jar file **MusicData.jar** as follows:

```
$hadoop jar MusicData.jar /music.txt /output
```



```
[acadgild@localhost ~]$ hadoop jar MusicData.jar /music.txt /output
18/09/07 14:25:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
18/09/07 14:25:27 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/09/07 14:25:27 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool in
terface and execute your application with ToolRunner to remedy this.
18/09/07 14:25:28 INFO input.FileInputFormat: Total input paths to process : 1
18/09/07 14:25:28 INFO mapreduce.JobSubmitter: number of splits:1
18/09/07 14:25:28 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1536298597224_0007
18/09/07 14:25:29 INFO impl.YarnClientImpl: Submitted application application_1536298597224_0007
18/09/07 14:25:29 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1536298597224_0007/
18/09/07 14:25:29 INFO mapreduce.Job: Running job: job_1536298597224_0007
18/09/07 14:25:38 INFO mapreduce.Job: Job job_1536298597224_0007 running in uber mode : false
18/09/07 14:25:38 INFO mapreduce.Job: map 0% reduce 0%
18/09/07 14:25:44 INFO mapreduce.Job: map 100% reduce 0%
18/09/07 14:25:51 INFO mapreduce.Job: map 100% reduce 100%
18/09/07 14:25:52 INFO mapreduce.Job: Job job_1536298597224_0007 completed successfully
18/09/07 14:25:53 INFO mapreduce.Job: Counters: 49
File System Counters
  FILE: Number of bytes read=53
  FILE: Number of bytes written=215749
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=164
  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)=4376
  Total time spent by all reduces in occupied slots (ms)=4668
  Total time spent by all map tasks (ms)=4376
  Total time spent by all reduce tasks (ms)=4668
```

```

Hadoop 2.6.1_1.1 [Running] - Oracle VM VirtualBox
Applications Places System
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~ acadgild@localhost:~
Total vcore-milliseconds taken by all map tasks=4376
Total vcore-milliseconds taken by all reduce tasks=4668
Total megabyte-milliseconds taken by all map tasks=4481024
Total megabyte-milliseconds taken by all reduce tasks=4780032
Map-Reduce Framework
  Map input records=4
  Map output records=1
  Map output bytes=6
  Map output materialized bytes=53
  Input split bytes=96
  Combine input records=1
  Combine output records=1
  Reduce input groups=1
  Reduce shuffle bytes=53
  Reduce input records=1
  Reduce output records=1
  Spilled Records=2
  Shuffled Maps =1
  Failed Shuffles=0
  Merged Map outputs=1
  GC time elapsed (ms)=162
  CPU time spent (ms)=2220
  Physical memory (bytes) snapshot=423776256
  Virtual memory (bytes) snapshot=4160446464
  Total committed heap usage (bytes)=290979840
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=68
File Output Format Counters
  Bytes Written=43
[acadgild@localhost ~]$

```

8. We can observe that the MR process is done.
9. Now lets open the */output* folder to check the status of the process and output of the process.
10. Open the */output* folder for status:

\$hadoop fs -ls /output

```

Hadoop 2.6.1_1.1 [Running] - Oracle VM VirtualBox
Applications Places System
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~ acadgild@localhost:~
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ hadoop fs -ls /output
18/09/07 14:26:08 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Found 2 items
-rw-r--r-- 1 acadgild supergroup          0 2018-09-07 14:25 /output/_SUCCESS
-rw-r--r-- 1 acadgild supergroup       43 2018-09-07 14:25 /output/part-r-00000
[acadgild@localhost ~]$

```

11. We can observe that the output is *_SUCCESS* indicates that the process executed successfully.
12. Now open the file */output/part-r-00000* for final output.

\$hadoop fs -cat /output/part-r-00000

```

[acadgild@localhost ~]$ hadoop fs -cat /output/part-r-00000
18/09/07 14:26:14 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Number of times a song was heard fully:      1
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$

```

13. We can observe that the message *Number of times a song was heard fully: 1* indicates that only one song was fully listened by the user.

Task 3: What are the number of times a song was shared.

1. For this task there is nothing to be changed for the *MusicData.java* driver program.
2. But for the *MusicDataMapper.java* program the following changes are required:

```
for(int i = 0; i < 2; i++) {  
    tokenizer.nextToken("|");  
}  
String str = tokenizer.nextToken("|");
```

```
if(str.equals("1")) {  
    word.set(str);  
    context.write(word, one);  
}
```

3. Other wise the following *MusicDataMapper.java* can be used with the above changes:

```
package MusicData;
```

```
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;  
import java.util.*;
```

```
public class MusicDataMapper  
    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();  
    StringTokenizer tokenizer = new StringTokenizer(line);  
    for(int i = 0; i < 2; i++) {  
        tokenizer.nextToken("|");  
    }  
    String str = tokenizer.nextToken("|");  
    if(str.equals("1")) {  
        word.set(str);  
        context.write(word, one);  
    }  
}
```

```
}
```

4. And the *MusicDataReducer.java* source code needs to be changed with the following changes:
int sum = 0;

```
for(IntWritable value : values) {  
    sum += value.get();  
}  
key.set("Number of times a song was shared: ");  
context.write(key, new IntWritable(sum));
```

5. Otherwise, use the following *MusicDataReducer.java* source code with the above changes:

```

package MusicData;
import java.io.IOException;

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

public class MusicDataReducer
    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();
        }
        key.set("Number of times a song was shared: ");
        context.write(key, new IntWritable(sum));
    }
}

```

6. Now create a jar **MusicData.jar** with the above three source files: **MusicData.java**, **MusicDataMapper.java** and **MusicDataReducer.java**.
7. Now run the jar file **MusicData.jar** as follows:
\$hadoop jar MusicData.jar /music.txt /output

```

Hadoop 2.6.1_1_1 [Running] - Oracle VM VirtualBox
Applications Places System
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~ acadgild@localhost:~
[acadgild@localhost ~]$ hadoop jar MusicData.jar /music.txt /output
18/09/07 14:58:09 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
18/09/07 14:58:10 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/09/07 14:58:11 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool in
terface and execute your application with ToolRunner to remedy this.
18/09/07 14:58:11 INFO input.FileInputFormat: Total input paths to process : 1
18/09/07 14:58:11 INFO mapreduce.JobSubmitter: number of splits:1
18/09/07 14:58:12 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1536298597224_0012
18/09/07 14:58:12 INFO impl.YarnClientImpl: Submitted application application_1536298597224_0012
18/09/07 14:58:12 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1536298597224_0012/
18/09/07 14:58:12 INFO mapreduce.Job: Running job: job_1536298597224_0012
18/09/07 14:58:21 INFO mapreduce.Job: Job job_1536298597224_0012 running in uber mode : false
18/09/07 14:58:21 INFO mapreduce.Job: map 0% reduce 0%
18/09/07 14:58:27 INFO mapreduce.Job: map 100% reduce 0%
18/09/07 14:58:35 INFO mapreduce.Job: map 100% reduce 100%
18/09/07 14:58:36 INFO mapreduce.Job: Job job_1536298597224_0012 completed successfully
18/09/07 14:58:36 INFO mapreduce.Job: Counters: 49
File System Counters
  FILE: Number of bytes read=48
  FILE: Number of bytes written=215739
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=164
  HDFS: Number of bytes written=38
  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)=4239
  Total time spent by all reduces in occupied slots (ms)=4837
  Total time spent by all map tasks (ms)=4239
  Total time spent by all reduce tasks (ms)=4837

```

```

Total vcore-milliseconds taken by all map tasks=4239
Total vcore-milliseconds taken by all reduce tasks=4837
Total megabyte-milliseconds taken by all map tasks=4340736
Total megabyte-milliseconds taken by all reduce tasks=4953088
Map-Reduce Framework
  Map input records=4
  Map output records=2
  Map output bytes=12
  Map output materialized bytes=48
  Input split bytes=96
  Combine input records=2
  Combine output records=1
  Reduce input groups=1
  Reduce shuffle bytes=48
  Reduce input records=1
  Reduce output records=1
  Spilled Records=2
  Shuffled Maps =1
  Failed Shuffles=0
  Merged Map outputs=1
  GC time elapsed (ms)=171
  CPU time spent (ms)=2130
  Physical memory (bytes) snapshot=421109760
  Virtual memory (bytes) snapshot=4163706880
  Total committed heap usage (bytes)=289406976
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=68
File Output Format Counters
  Bytes Written=38
You have new mail in /var/spool/mail/acadgild

```

8. Now the Map Reduce process is successfully completed.
9. Open the **/output** folder to check the success of the process and final output of the process.
10. Open the **/output** folder:

\$hadoop fs -ls /output

The screenshot shows a terminal window titled 'Hadoop 2.6.1_1.1 [Running] - Oracle VM VirtualBox'. The user 'Acadgild' is logged in. The terminal output shows the command 'hadoop fs -ls /output' being executed. It returns a warning from 'util.NativeCodeLoader' and then lists two files in the /output directory: 'SUCCESS' and 'part-r-000000'.

```

Hadoop 2.6.1_1.1 [Running] - Oracle VM VirtualBox
Applications Places System
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ hadoop fs -ls /output
18/09/07 15:13:01 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Found 2 items
-rw-r--r-- 1 acadgild supergroup          0 2018-09-07 14:58 /output/_SUCCESS
-rw-r--r-- 1 acadgild supergroup        38 2018-09-07 14:58 /output/part-r-000000
[acadgild@localhost ~]$

```

11. Open the **/output/part-r-000000** file for the final output.

\$hadoop fs -cat /output/part-r-000000

The screenshot shows the same terminal window. The user has executed 'hadoop fs -cat /output/part-r-000000'. The output is 'Number of times a song was shared: 2'.

```

Hadoop 2.6.1_1.1 [Running] - Oracle VM VirtualBox
Applications Places System
acadgild@localhost:~
File Edit View Search Terminal Tabs Help
acadgild@localhost:~
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ hadoop fs -cat /output/part-r-000000
18/09/07 15:19:33 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Number of times a song was shared:      2
[acadgild@localhost ~]$

```

12. We can observe with the message **“Number of times a song was shared: 2”**, indicates that the song was shared 2 times.