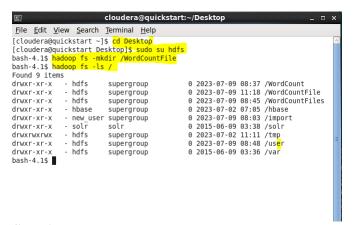
## 7(a): MapReduce application for word counting on Hadoop cluster.

- **Step 1:** Start Cloudera and open Eclipse. Create a new Java Project. Enter the project name and Check the default output folder and Click Next
- **Step 2:** Add external jars to compile the code. **Click on Libraries Tab** and "Add External Jars". Right Select all the jars present in folder /usr/lib/hadoop/client, /usr/lib/Hadoop, /usr/lib/hadoop/lib.
- Step 3: Select File system and click on usr/lib/Hadoop/lib. Select all Jar files and click on Ok and Finish.
- **Step 4:** Project is created with name Word Count. Right Click on the project new Class.
- **Step 5 :** Create a jar file of the program. Right click on project select "Export" and then click on "Jar File" under Java folder. Give the location path where you want to store your .jar file.
- **Step 6:** Write the java program using Map, Reduce and Driver methods and save it.

## Code:

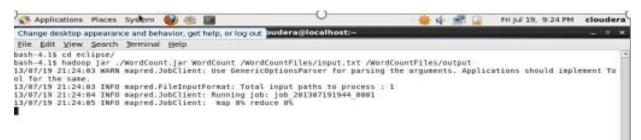
```
import java.io.IOException;import
java.util.*;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
public class WordCount {
       public static class WcMap extends Mapper<LongWritable, Text, Text, IntWritable> {private
               final static IntWritable one = new IntWritable(1);
               private Text word = new Text();
               public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
                       String line=value.toString();
                       StringTokenizer tokenizer = new StringTokenizer(line); while
                       (tokenizer.hasMoreTokens()) {
                               word.set(tokenizer.nextToken());
                               context.write(word, one);
                       }
```

```
public static class WcReduce extends Reducer<Text, IntWritable, Text, IntWritable> {public
                 void reduce(Text key,Iterable<IntWritable> values, Context context)
         throws IOException, InterruptedException {
                 int sum = 0;
                 for(IntWritable val:
                 values){sum += val.get();
                 context.write(key, new IntWritable(sum));
         }
 public static void main(String[] args) throws Exception {Configuration
         conf = new Configuration();
                 Job job = Job.getInstance(conf,"Word Count");
                 job.setOutputKeyClass(Text.class);
                 job.setOutputValueClass(IntWritable.class);
                 job.setMapperClass(WcMap.class);
                 job.setReducerClass(WcReduce.class);
                 job.setInputFormatClass(TextInputFormat.class);
                job.setOutputFormatClass(TextOutputFormat.class);
                 FileInputFormat.addInputPath(job, new Path(args[0]));
                 FileOutputFormat.setOutputPath(job, new Path(args[1]));
                 job.setJarByClass(WordCount.class); job.waitForCompletion(true);
 Step7:Create Input file. Open the terminal and create a input file which is a huge text file.
   $vim input.txt
Step 8 : Make a new file directory on HDFS (Hadoop Distributed File System)
$ cd Desktop
$ sudo su hdfs
hadoop fs -mkdir /WordCountFile
hadoop fs -ls /
```



Step 9: Copy this file on the NameNode i.e., on HDFS \$ hdfs dfs -copyFromLocal input.txt

**Step 10:** Run the program using the hadoop command. Open **new terminal** type **cd Desktop** press enter and type \$ **hadoop jar WordCount.jar WordCount/WordCountFile /output2.** Map Reduce program starts to run. We can see the percentage of mapping and reducing the program is doing on the command line.



Step 11: Display the output. \$\\$hadoop fs \( -\cat{VordCountFiles/ouput/part-0000} \)

```
File Edit View Search Terminal Help
Mirror 22
Next
       1
Not
       1
Now
Prince 2
Queen
      107
Queen, 36
Queen. 12
       22
Seven
She
       24
Since 11
Snow
       176
Soon
       11
The
      117
Then
      11
They
       22
This
       12
Time
      1
when
      11
White 124
White's 5
White, 33
```