EXPERIMENT NO. 4

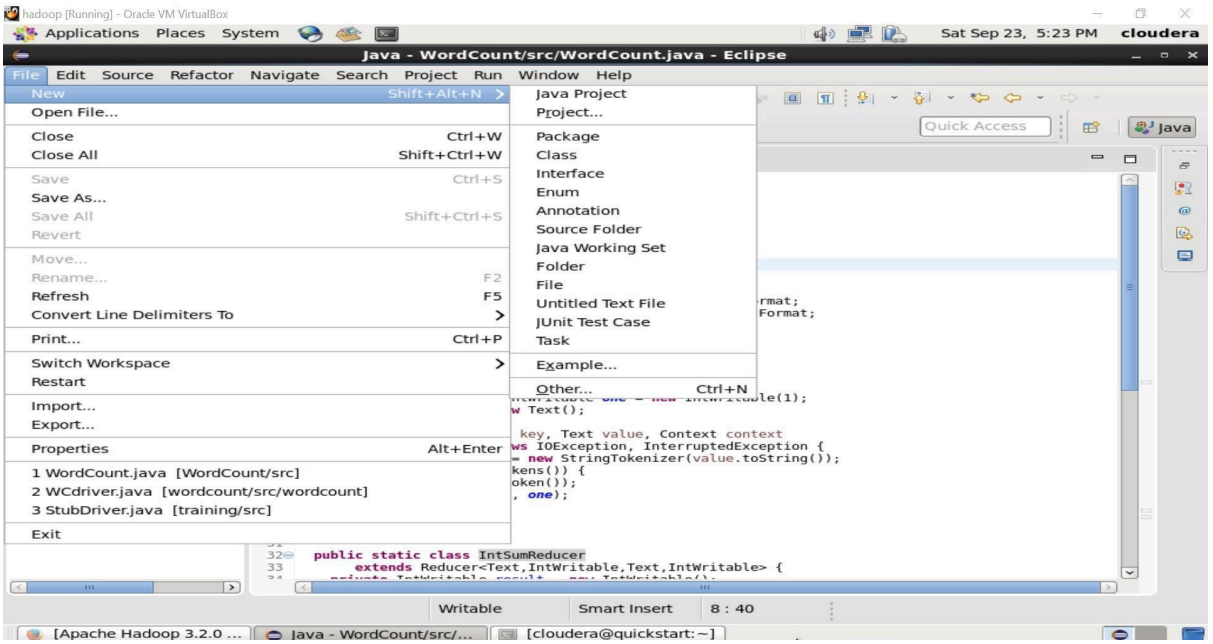
Aim: Execution of MapReduce program for sorting of numbers and counting word occurrences in a text file.

Theory:

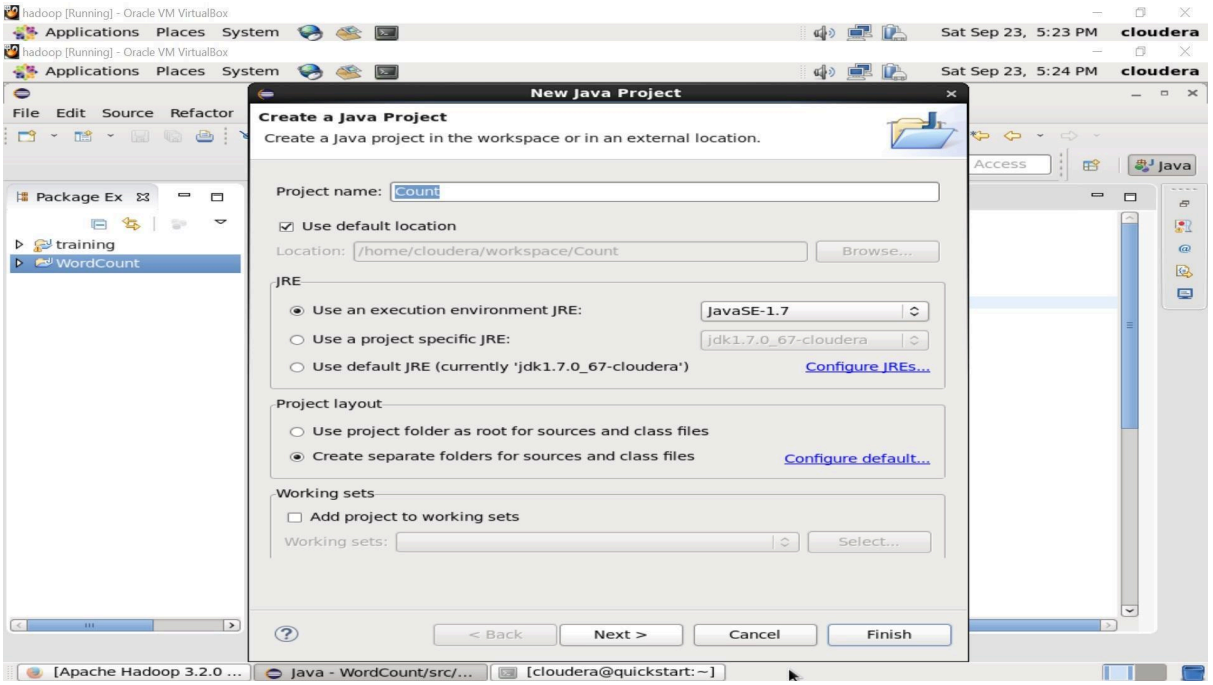
1. **Explain the Use of Every Hadoop Ecosystem Component.**

Practical

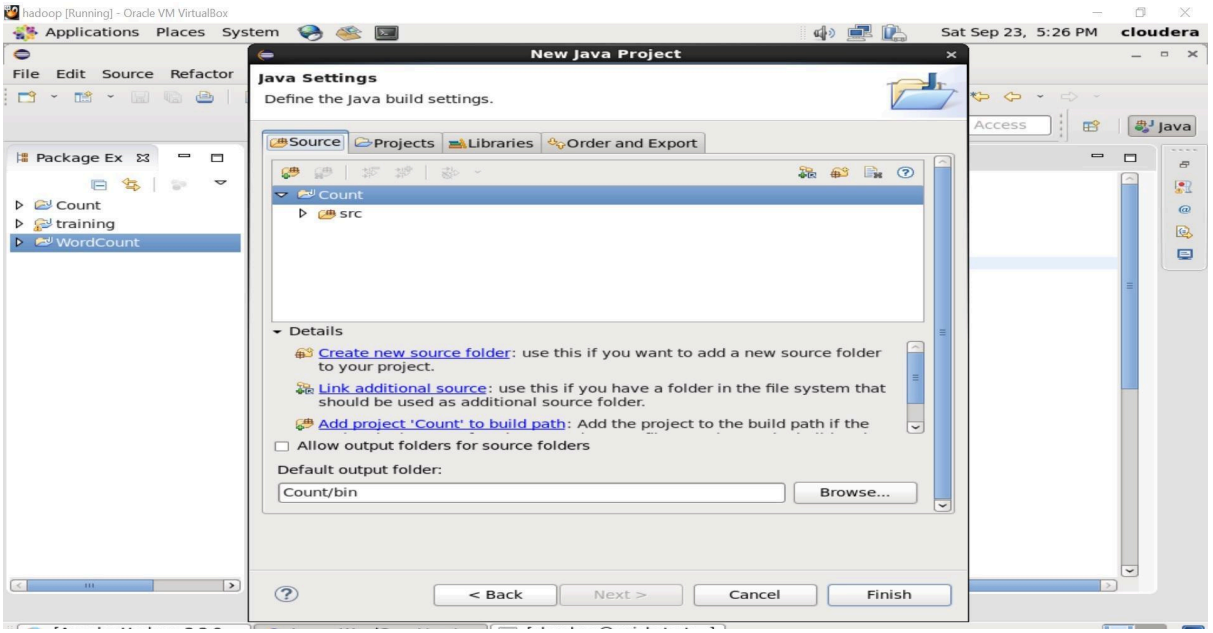
1.In cloudera → open eclipse —>File —> java Project



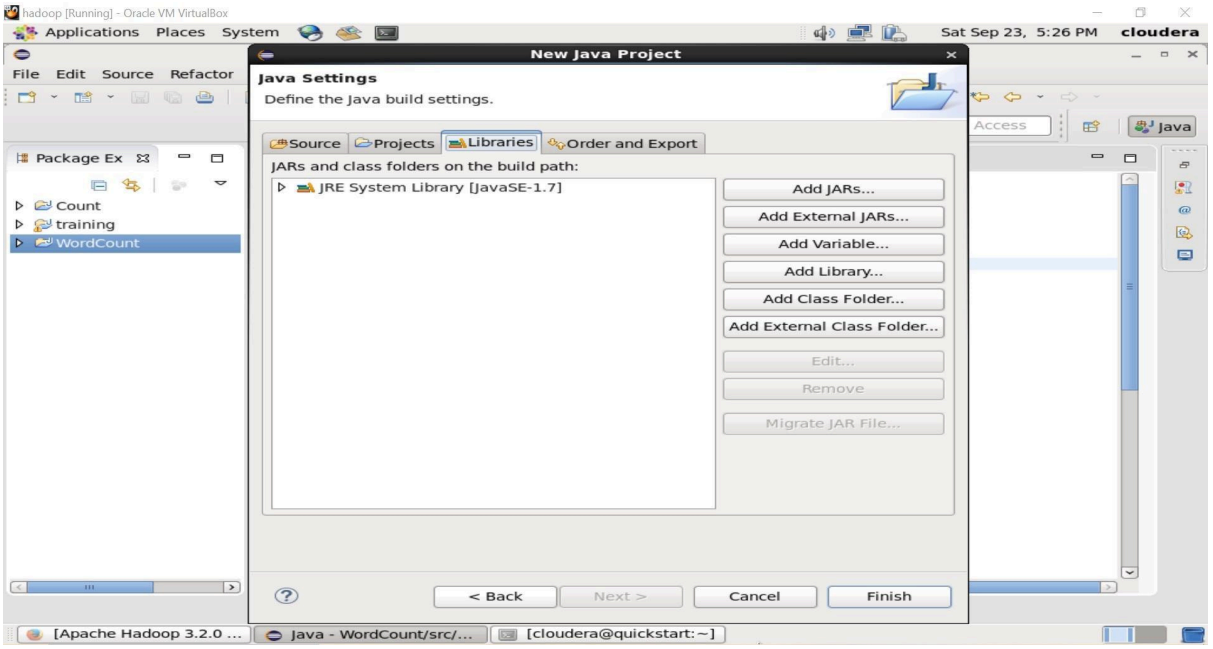
2. Save as file name "Count". Don't click on the finish button but continue with the next button.



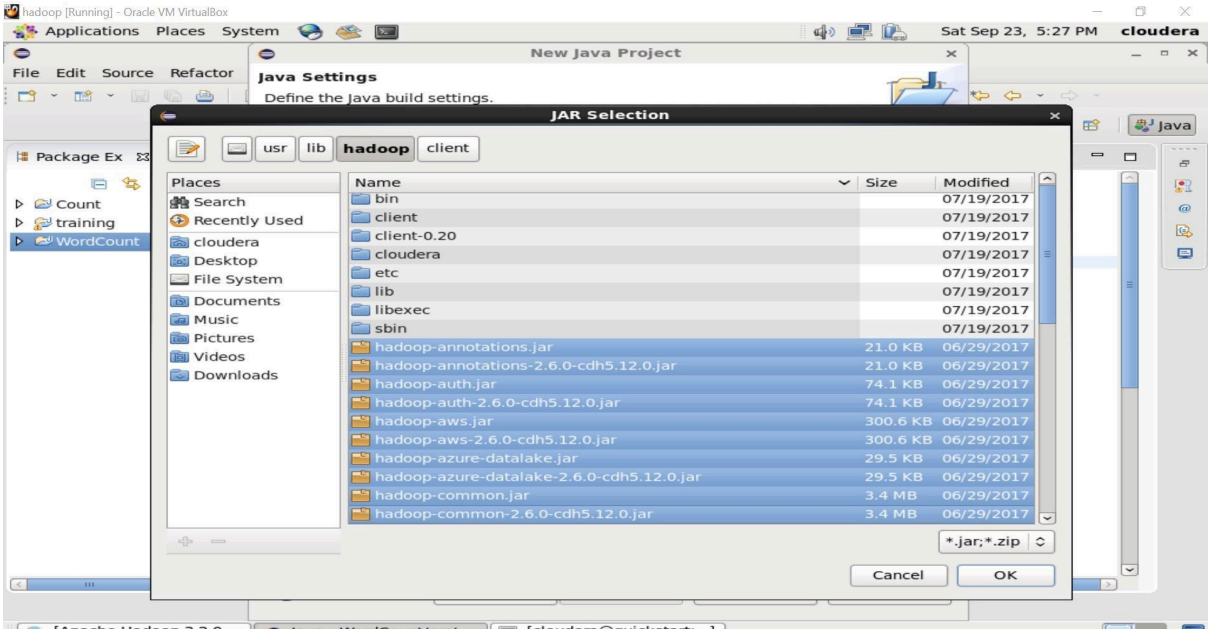
3. click on the Libraries tab.



4. Click on Add External JARS files.

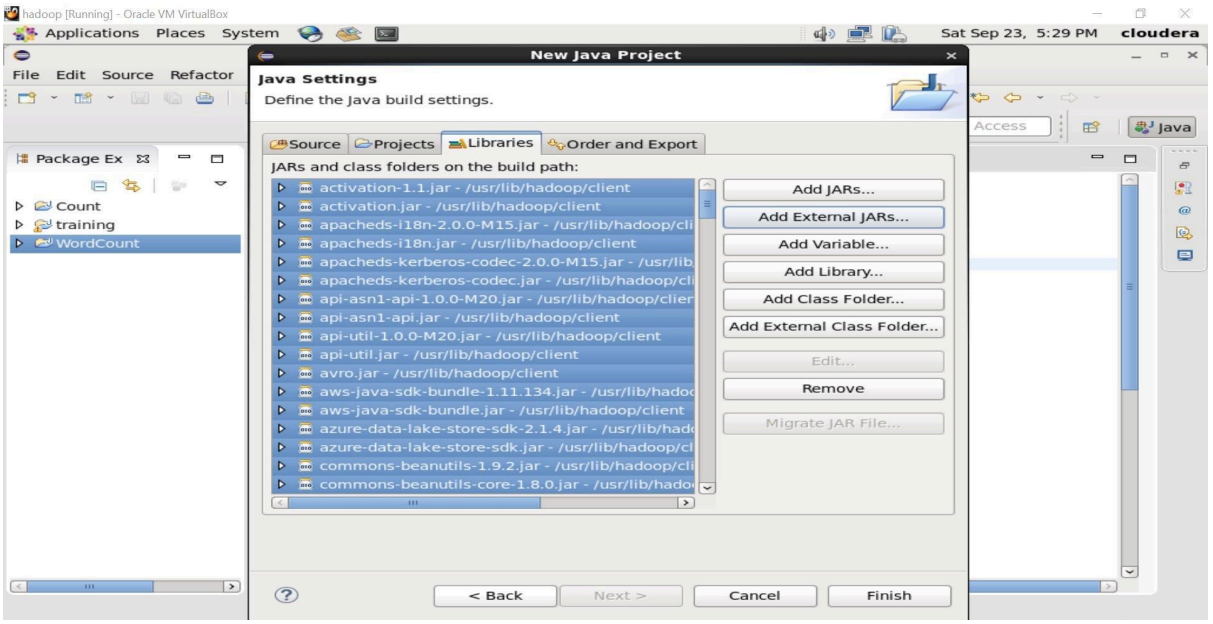


5. Open the file from Filesystem→ /usr/lib/hadoop/select the all jar files. (shfit+ctrl)



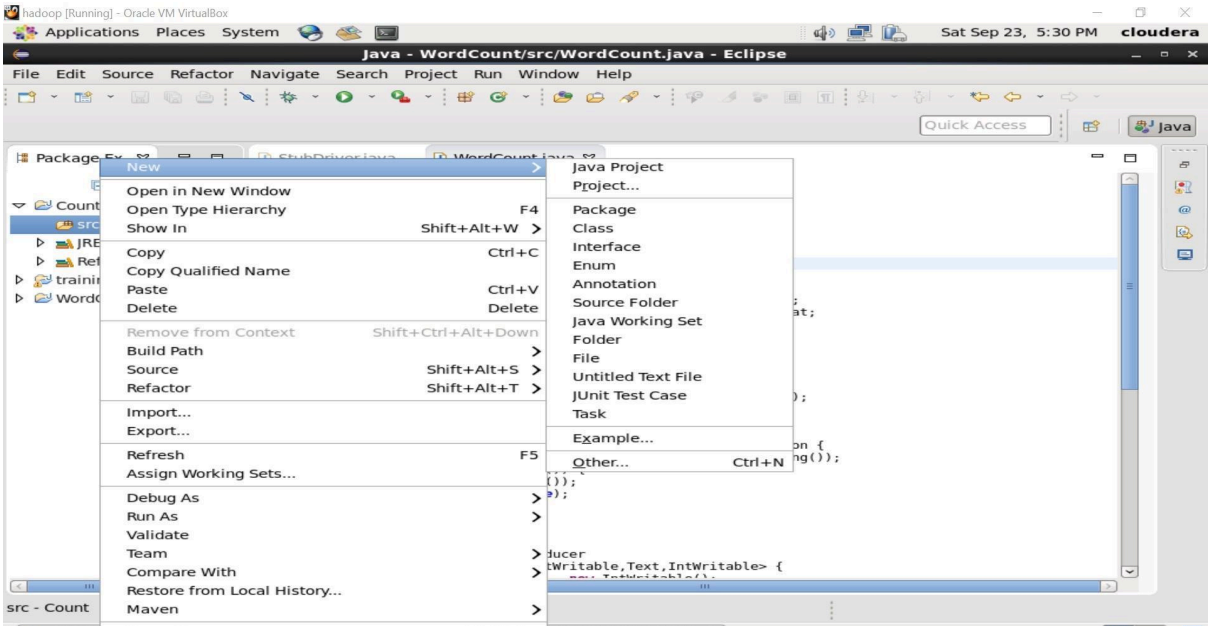
Option 5 is not for Cloudera_1.5 so directly jump on option 6.

5. Again click on Add external JARS tab.
Open the file from Filesystem→ /usr/lib/hadoop/client/select all jar files.

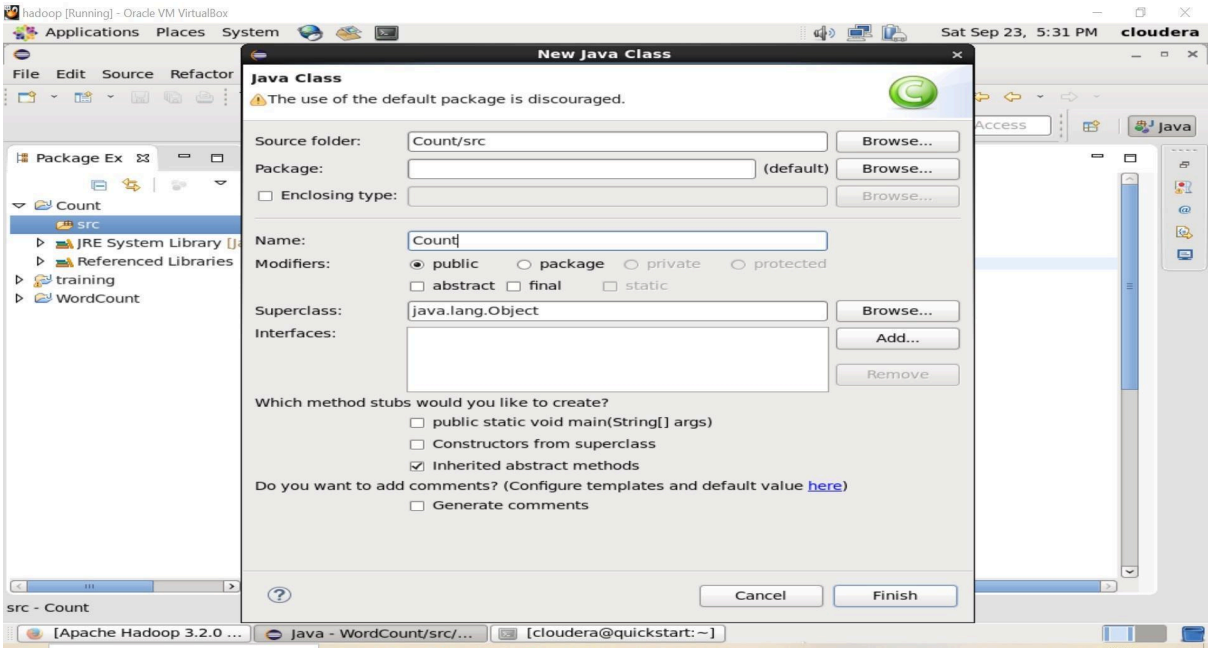


Click the finish button.

6. Right-click on the Src file→NEW→ Class.



7.Save as Count java class name



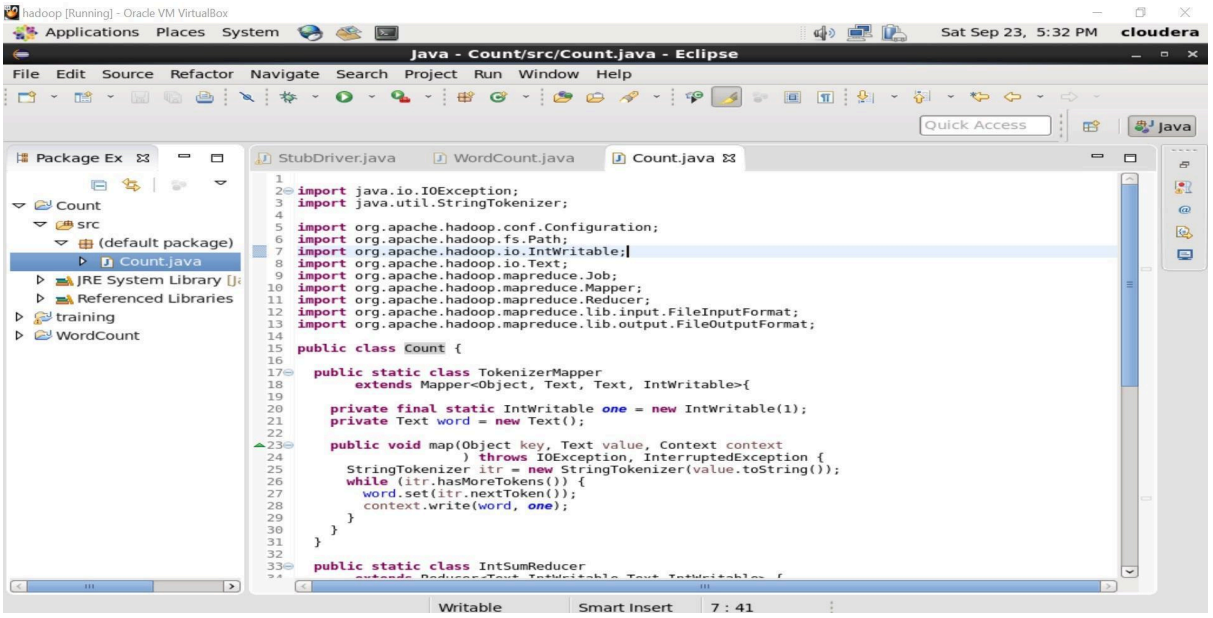
It will open the count.java file

8. Write MapReduce Program for word count in the java program.

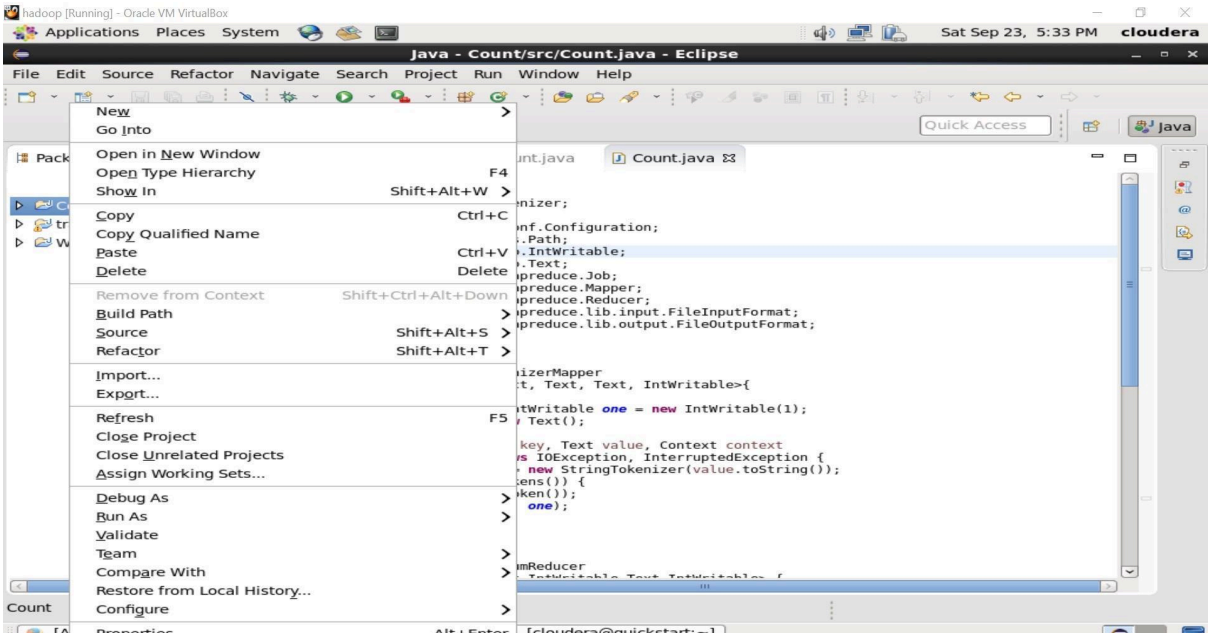
```
import java.io.IOException;
import java.util.StringTokenizer;

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.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class shortcount2025 {

        public static class TokenizerMapper
    extends Mapper<Object, Text, Text, IntWritable>{

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

  public void map(Object key, Text value, Context context
                  ) throws IOException, InterruptedException {
    StringTokenizer itr = new StringTokenizer(value.toString());
```

```
    while (itr.hasMoreTokens()) {
      word.set(itr.nextToken());
      context.write(word, one);
    }
  }
}

public static class IntSumReducer
     extends Reducer<Text,IntWritable,Text,IntWritable> {
  private IntWritable result = new IntWritable();

  public void reduce(Text key, Iterable<IntWritable> values,
                     Context context
                     ) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
      sum += val.get();
    }
    result.set(sum);
    context.write(key, result);
  }
}

public static void main(String[] args) throws Exception {
  Configuration conf = new Configuration();
  Job job = new Job(conf, "short count 2025");
  //Job job = Job.getInstance(conf, "shortcount2025");
  job.setJarByClass(shortcount2025.class);
  job.setMapperClass(TokenizerMapper.class);
  job.setCombinerClass(IntSumReducer.class);
  job.setReducerClass(IntSumReducer.class);
  job.setOutputKeyClass(Text.class);
  job.setOutputValueClass(IntWritable.class);
  FileInputFormat.addInputPath(job, new Path(args[0]));
  FileOutputFormat.setOutputPath(job, new Path(args[1]));
  System.exit(job.waitForCompletion(true) ? 0 : 1);
 }
}
```

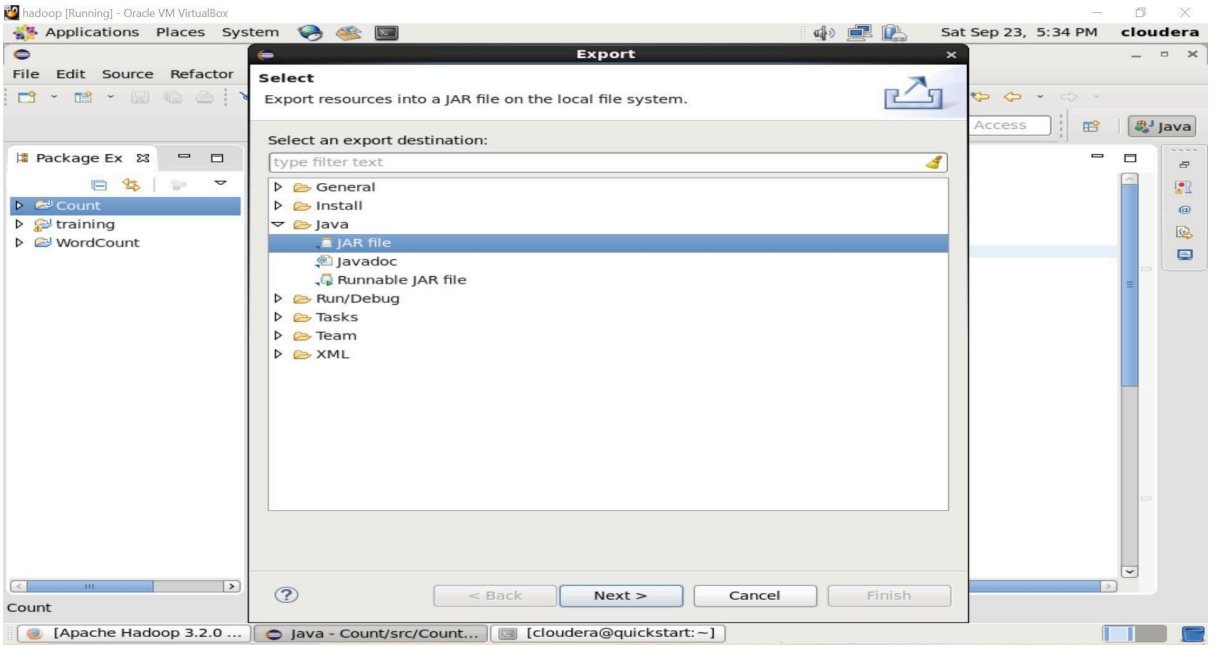
9.Save the file. (ctrl+S}



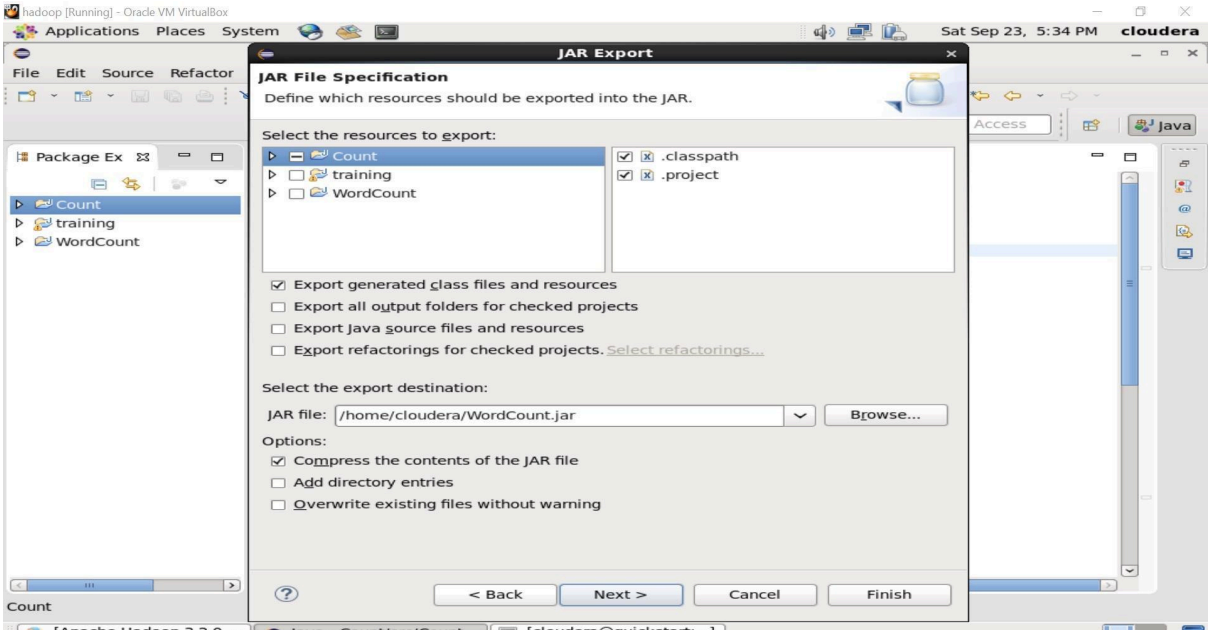
10.Click on File —> Export.



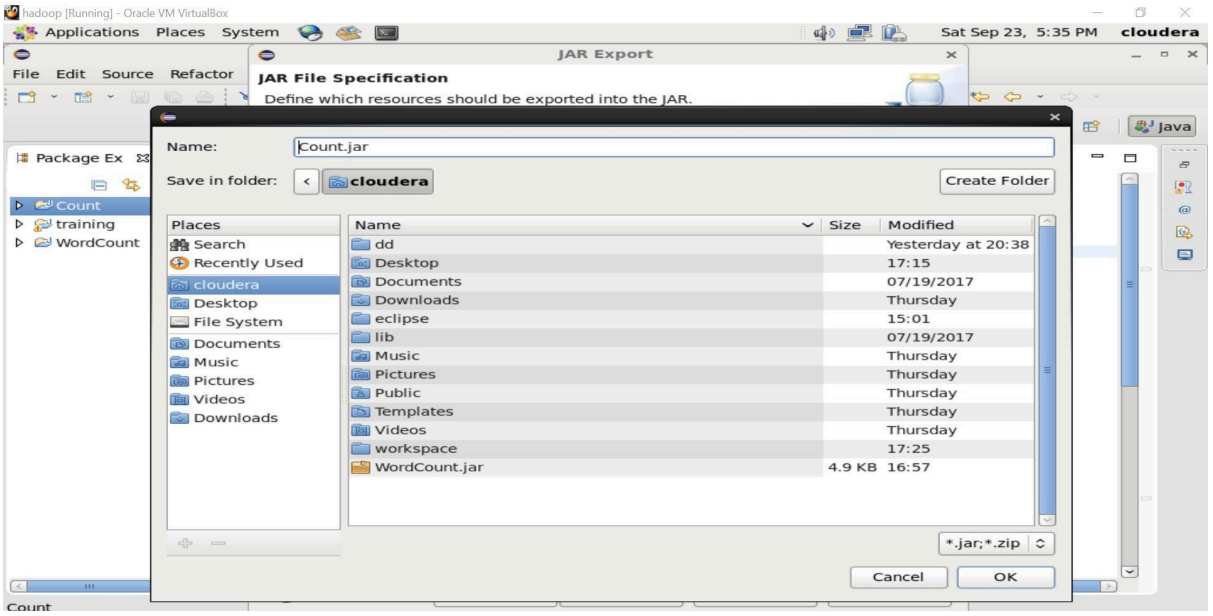
11. Export resources into Java→ a JAR file on the local file system.



12. Click on NEXT. Select project as " Count". Next Select Export Destination.



13.Save as Count.jar file in cloudera directory.(home/user/cloudera)



14. check the Cloudera Directory Count.jar available or not.

14. Go to Terminal type command below

```
pwd
```

15.Create one text file for mapreduce count program

```
nano  cn.txt
```

Enter some word like

Sakec
Sakec
Vesp
Vesp
Vesit
Djsce
Djsce

Ctrl+x →type "yes" → Enter(file save)

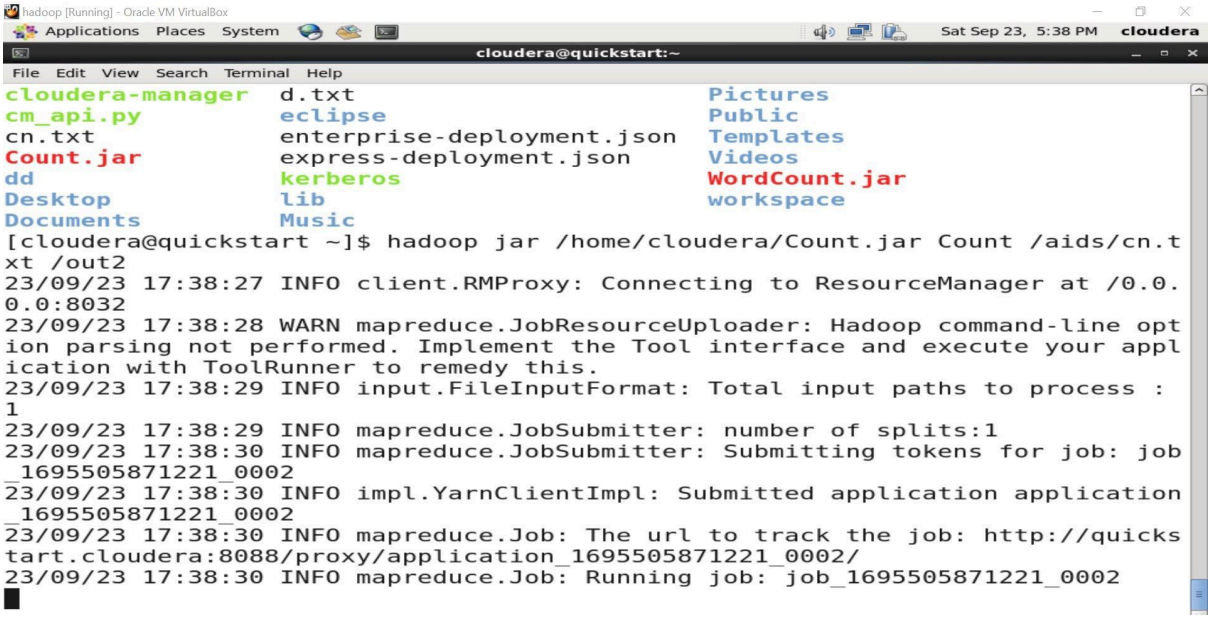
16. shift cn.txt on Hadoop. Before that create one directory aids

```
hadoop  dfs -mkdir /aids
```

17.shift the cn.txt on this directory on hadoop.

```
hadoop  dfs -put /home/usr/cloudera/cn.txt  /aids
```

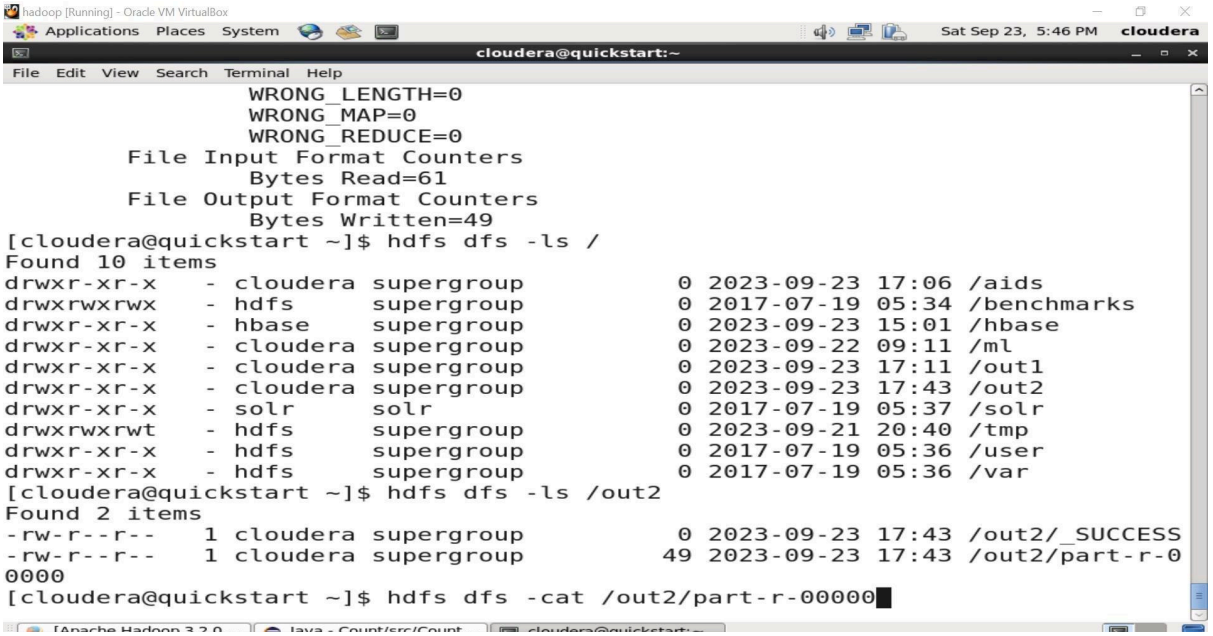
18.hadoop jar /home/clouldera/Count.jar Count /aids/cn.txt /out2(initially take 1 min.)



19.Checkout out2 file being created.

hadoop dfs -ls /

19.hadoop dfs -ls /aids/out2

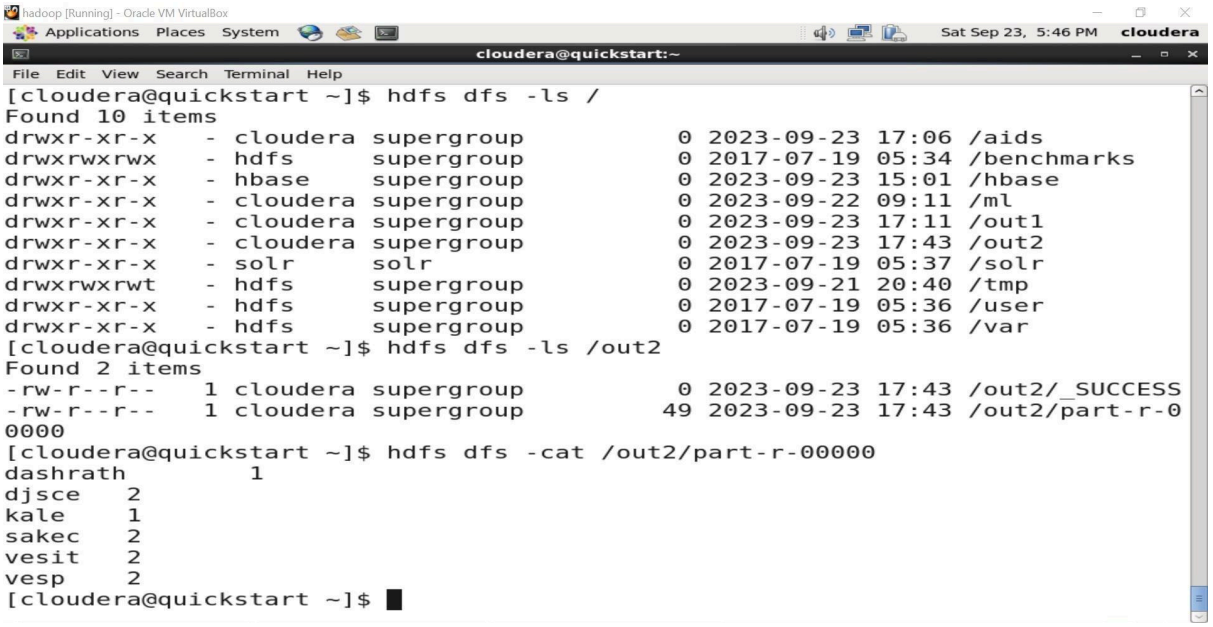


20. Status of Out2 file

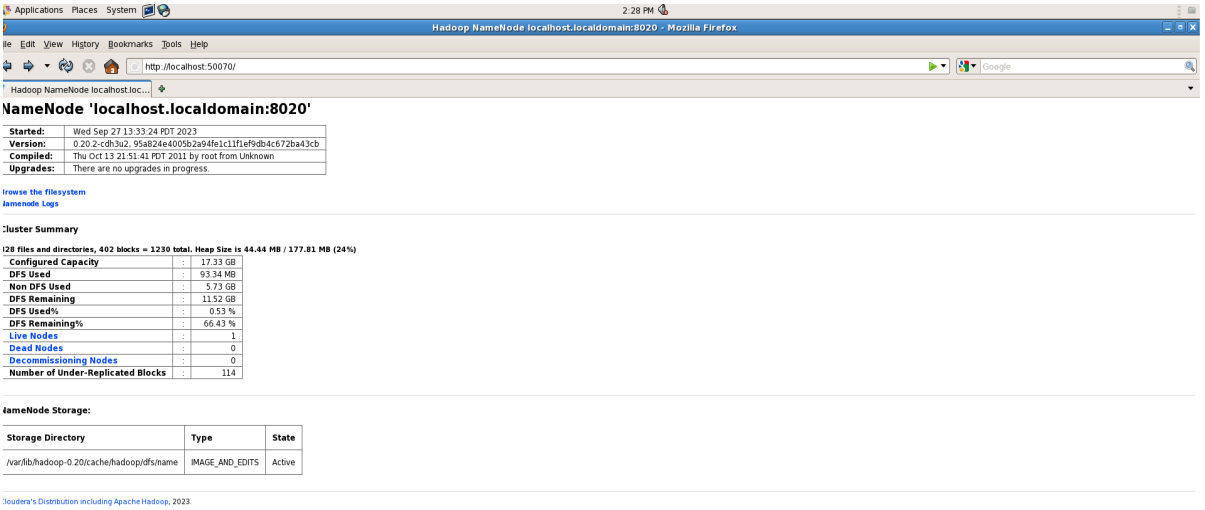
hadoop dfs -ls /out2

21. Finally run command

hadoop dfs -cat /out2/part-r-00000



23. Want see on Browser put url : localhost:50070



Conclusion: hence we study how to run word count programs using MapReduce module on Hadoop.

References:

1.https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

2. https://youtu.be/Wb5p8S5jZCc?si=5J8rmuyMzWMowzbX

3.https://www.youtube.com/watch?v=uH5y6nTo_04&t=2s