**Maharishi University of Management**

**CS522 – Big Data**

**Prof. Premchand Nair**

**Set Up a Single Node Cluster**

**September 20th, 2017**

|  |  |  |
| --- | --- | --- |
| Author | ID | Date |
| Minh Hieu Nguyen | 985949 | 09/20/2017 |

In this manual, we're going to see how to set up a single node cluster and optionally an eclipse development environment to create and test your programs on Microsoft Windows.

1. **Install VMWARE**

First, go to <https://www.vmware.com/products/workstation/workstation-evaluation.html>

### Under VMware Workstation 12.5 Pro for Windows 64-bit, click on Download now for windows

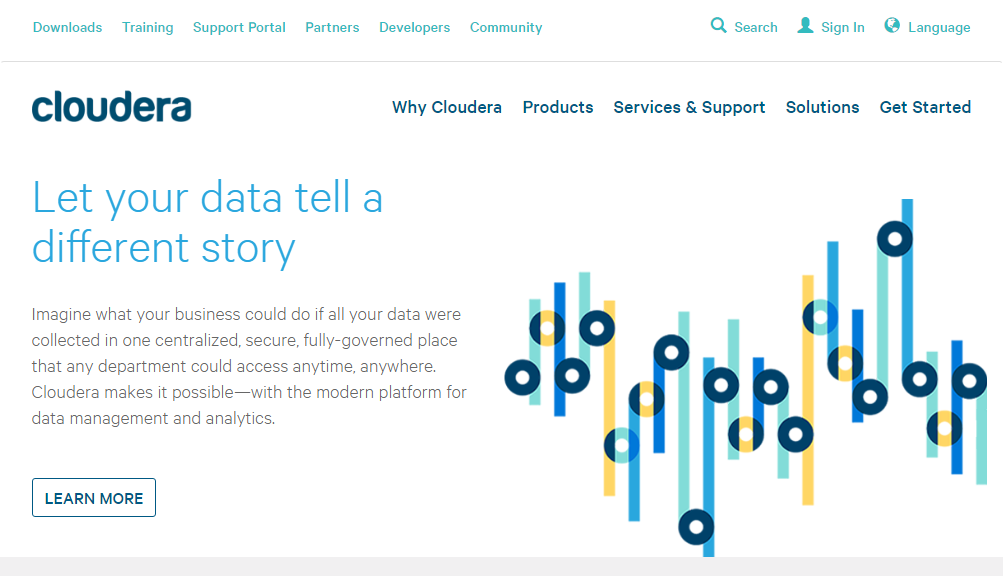
### 

Then after download, double click on setup file to perform the VMWARE installation

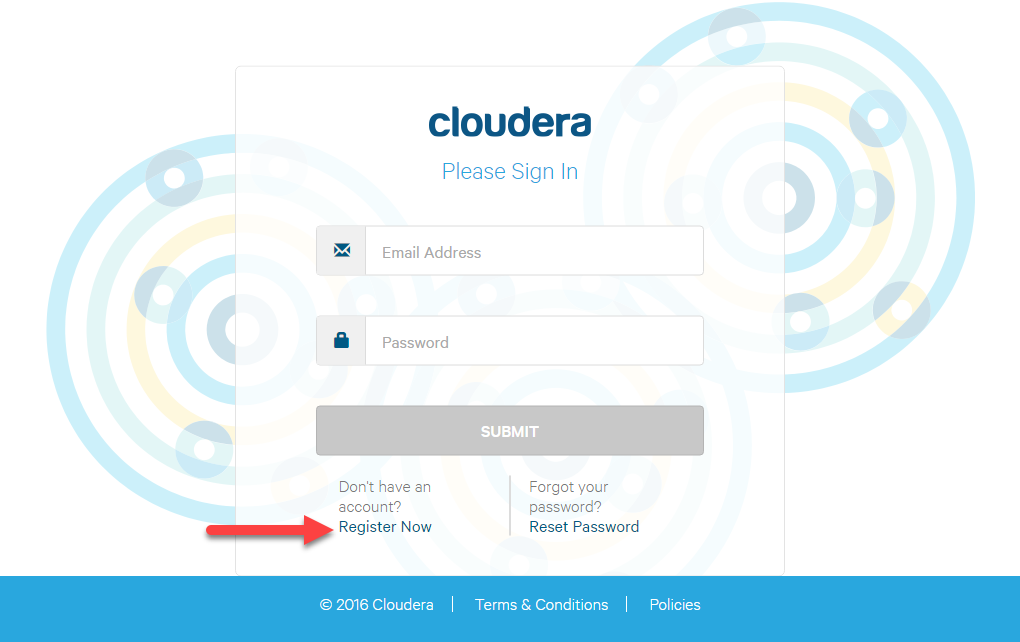


After finished installation, you will able to run the program.

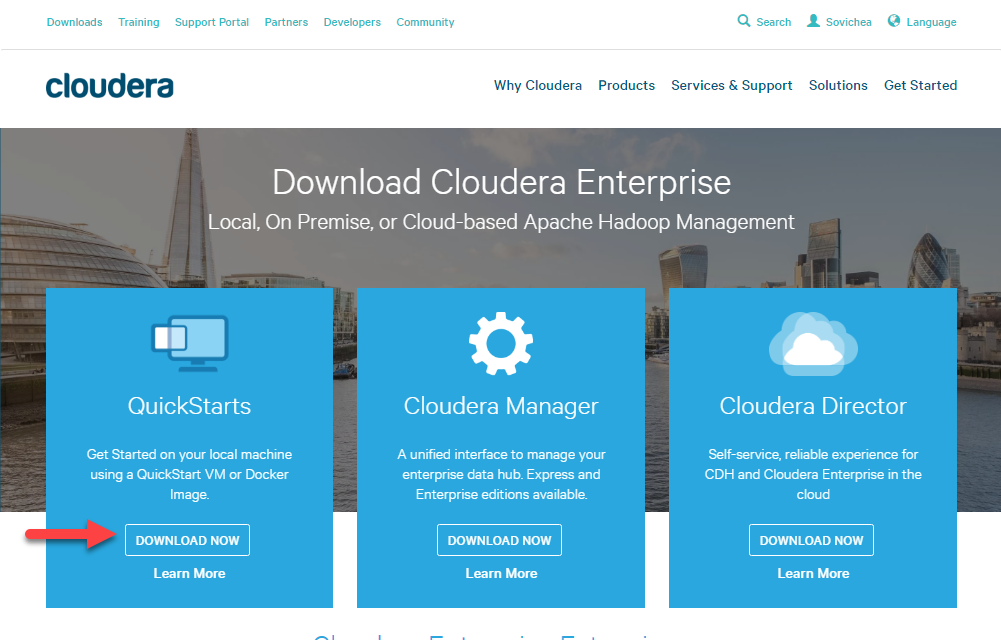
1. **Download & Install Cloudera**
   1. **Get Cloudera**

First, go to <http://www.cloudera.com/> 

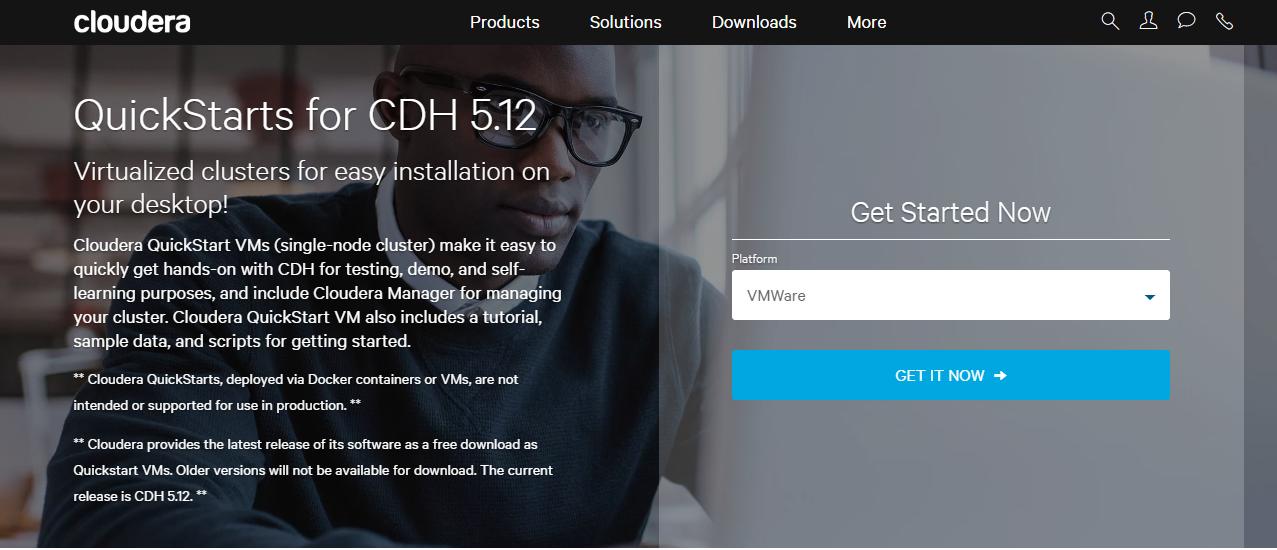
Click on **Sign In**, you will need an account to download Cloudera, if you don't have an account yet, click on **Register Now**.



After finished Register or Sign In, go to <http://www.cloudera.com/downloads.html> to download Cloudera. Under Quick Start, click on **Download Now**



Select Version and Platform: Virtual Box, then click **Get It Now**

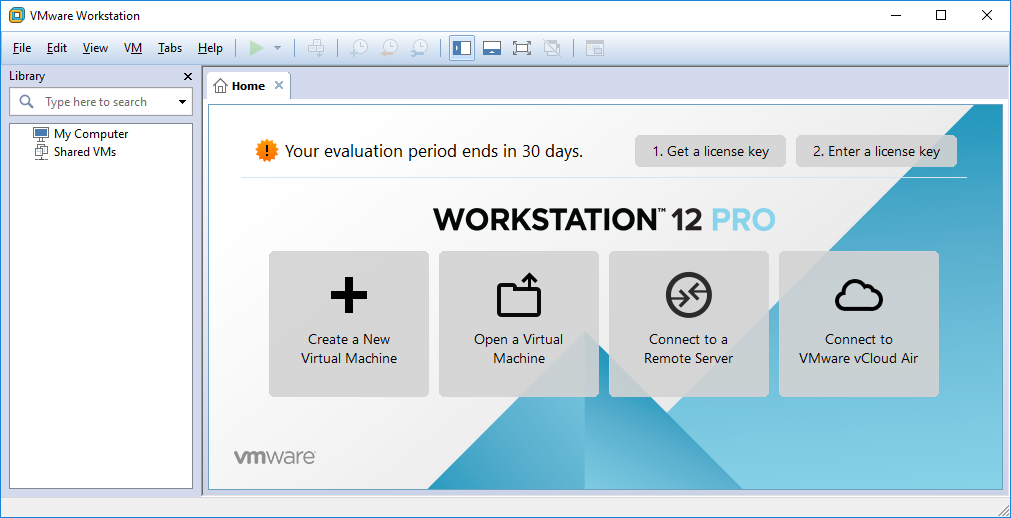


Download will start, you will get the zip file after finished

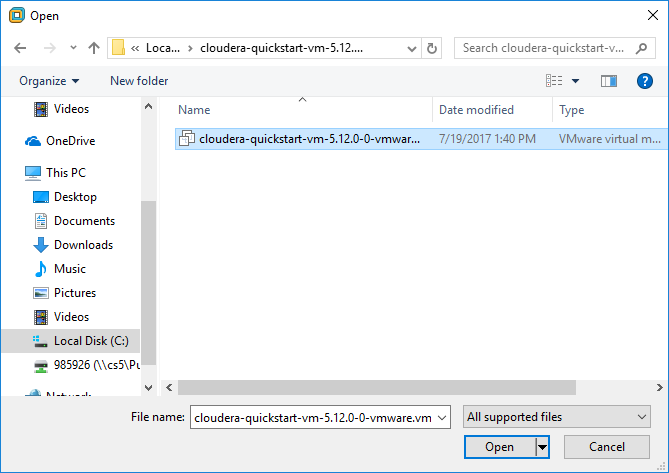


* 1. **Install Cloudera**

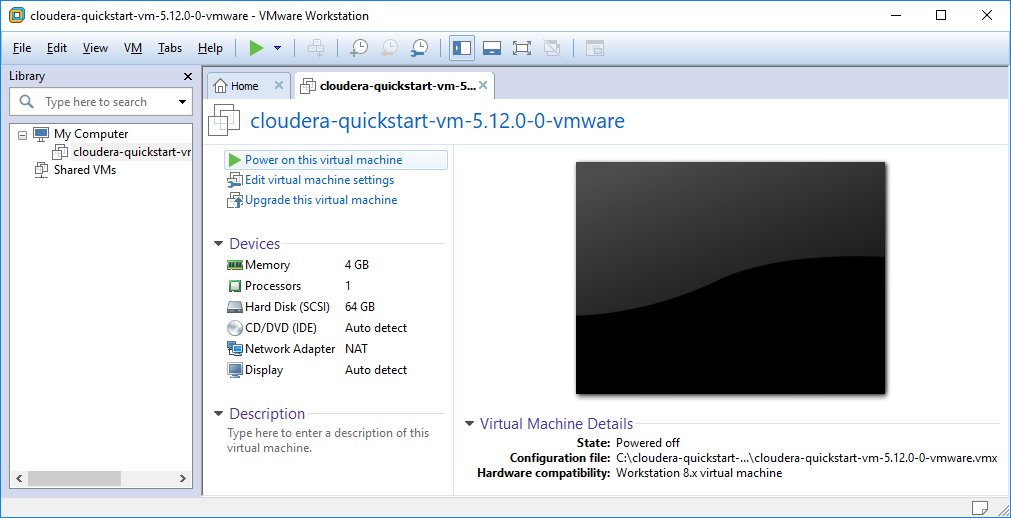
After you VMWARE image file, you can import it to VMWARE. Run WMWARE, then click open a virtual machine



Extract "cloudera-quickstart-vm-5.12.0-0-vmware.zip" to an appropriate location.

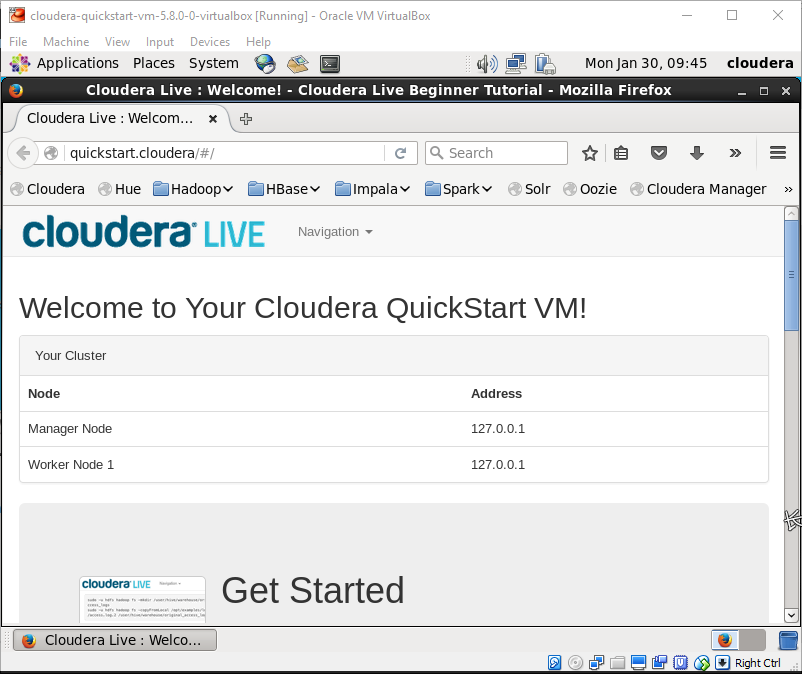


After finished open, you will able to see your Virtual Machine. Click Power to start the VM.

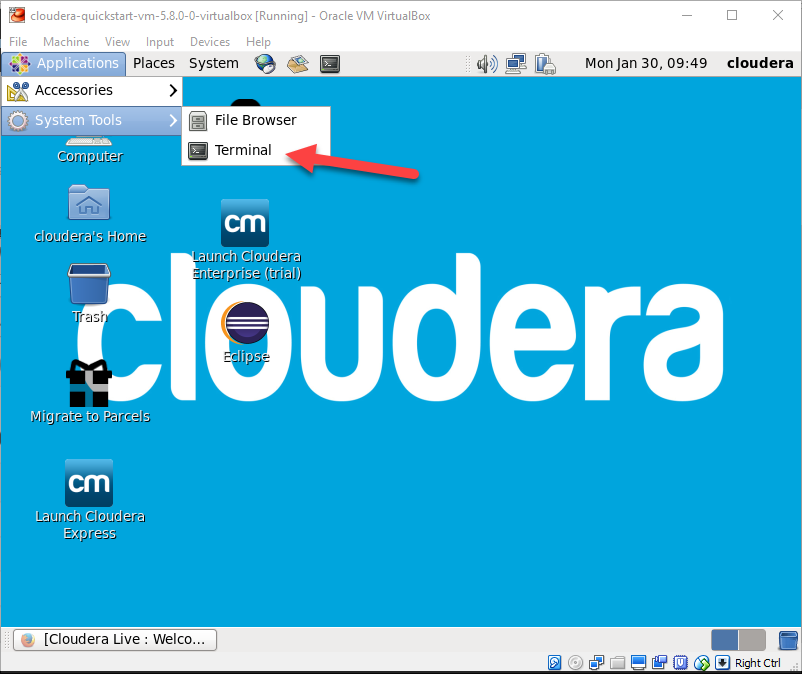


You will able to get to Red Hat CentOS after finish running

Cloudera Welcome page will pop up automatically.



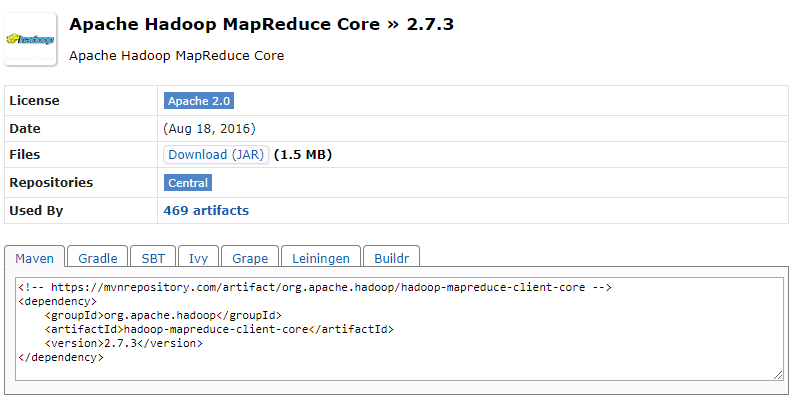
Go to Application -> System Tools -> Terminal

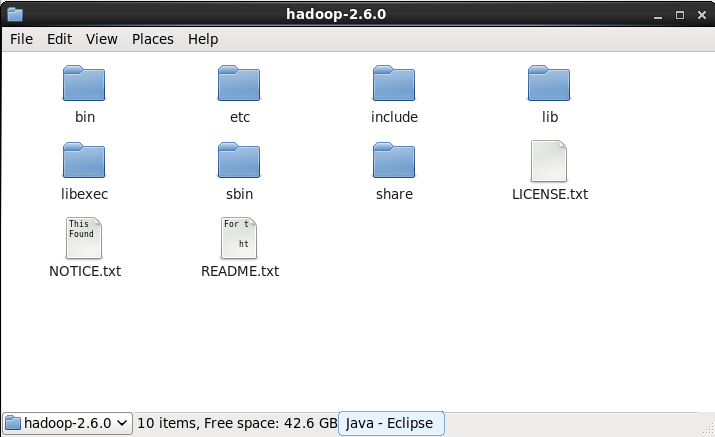


1. **Test Run WordCount**
   1. **Download Hadoop Libraries**

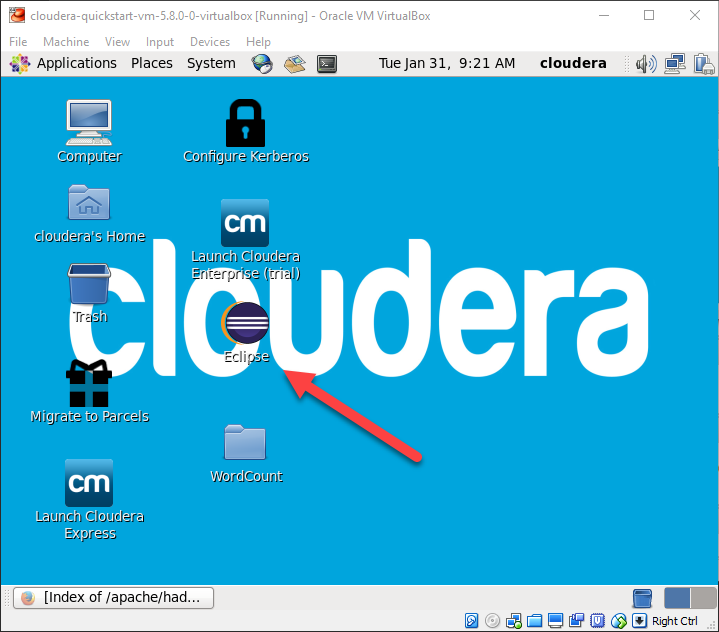
You will require hadoop libraries to run WordCount.



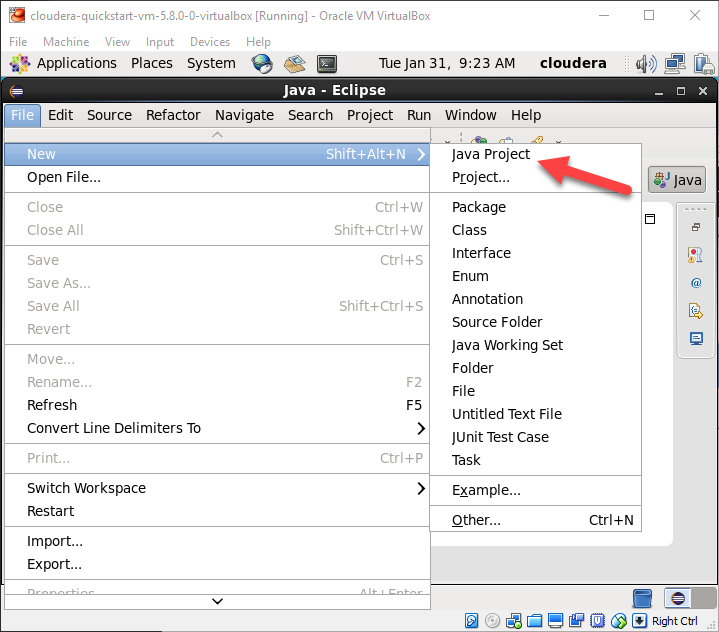


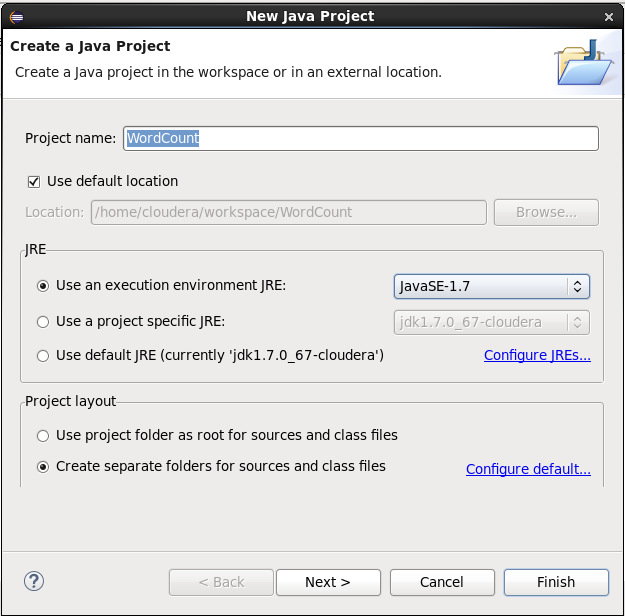
You will get the file as below, and you will need it to import to Eclipse later

* 1. **Eclipse Project Setup**

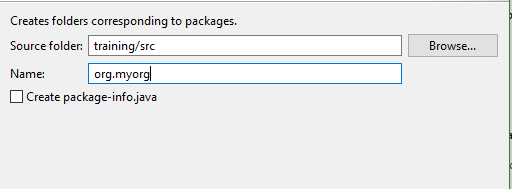
By default, Eclipse is available on your VM machine

Run the Eclipse IDE and then create Java Project

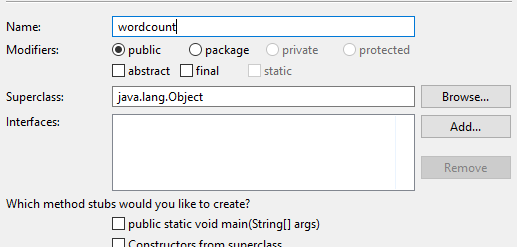


Setup project name and JRE

Create Package name



Create WordCount class



* 1. **WordCount Source Code and Library Setup**

WordCount maps (extract) words from an input source and reduces the result, return a count of each word. You can find source code below or on the internet

|  |
| --- |
| package org.myorg;  import java.io.IOException;  import java.util.\*;    import org.apache.hadoop.fs.Path;  import org.apache.hadoop.conf.\*;  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 Map 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 Reduce 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 = new Job(conf, "wordcount");  job.setJarByClass(WordCount.class);    job.setOutputKeyClass(Text.class);  job.setOutputValueClass(IntWritable.class);    job.setMapperClass(Map.class);  job.setReducerClass(Reduce.class);    job.setInputFormatClass(TextInputFormat.class);  job.setOutputFormatClass(TextOutputFormat.class);    FileInputFormat.addInputPath(job, new Path(args[0]));  FileOutputFormat.setOutputPath(job, new Path(args[1]));    job.waitForCompletion(true);  }  } |

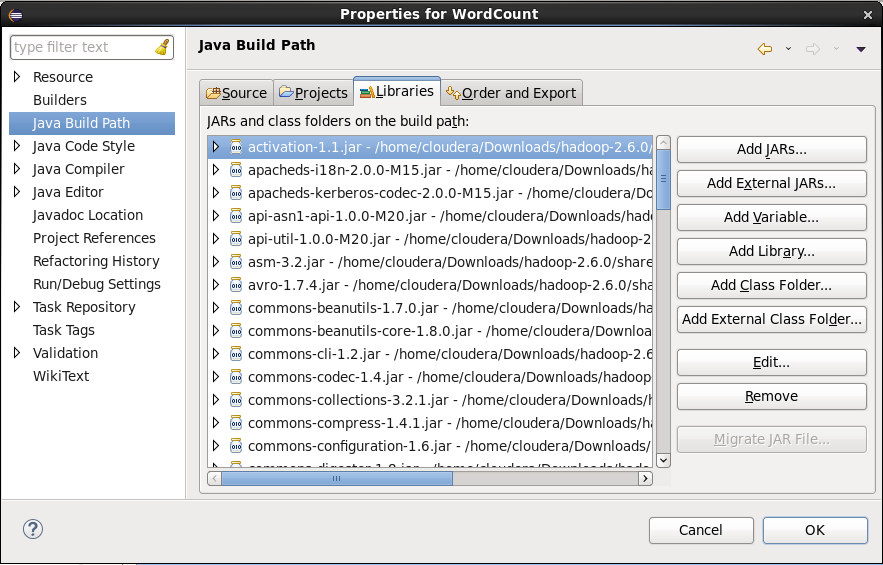
Add all required libraries as below

share/hadoop/common/hadoop-common-\*.jar

share/hadoop/mapreduce/hadoop-mapreduce-client-core-\*.jar

share/hadoop/mapreduce/hadoop-mapreduce-client-jobclient-\*.jar

share/hadoop/common/hadoop-common-lib-\*.jar (all files)



* 1. **Running WordCount**

Before you run, you must create input and output locations in HDFS. Use the following commands to create input directory /user/cloudera/wordcount/input in HDFS:

$ sudo su hdfs

$ hadoop fs -mkdir /user/cloudera

$ hadoop fs -chown cloudera /user/cloudera

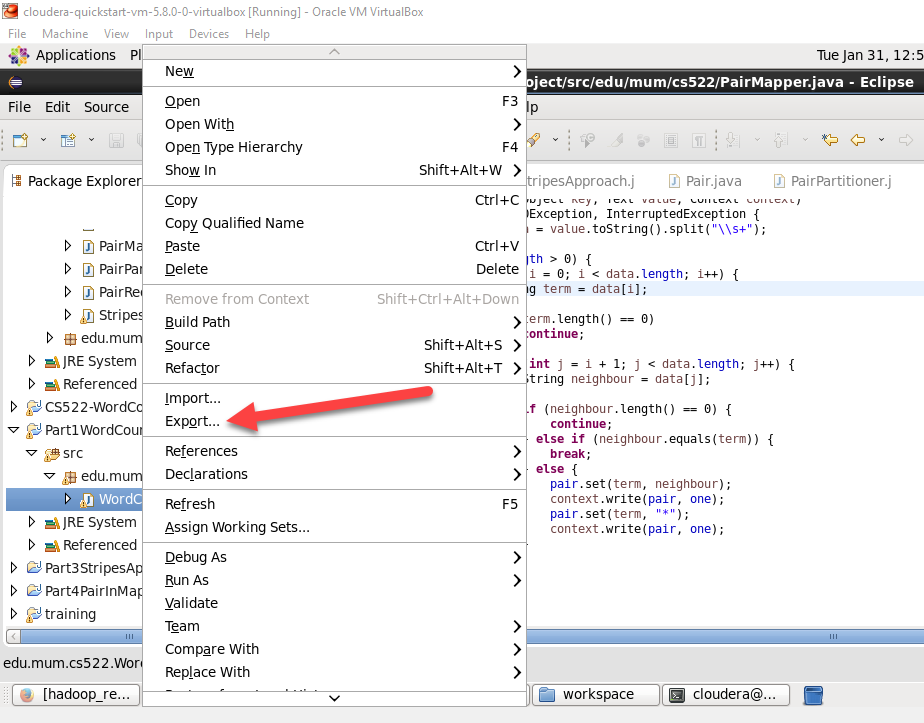
$ exit$ sudo su cloudera

$ hadoop fs -mkdir /user/cloudera/wordcount /user/cloudera/wordcount/input

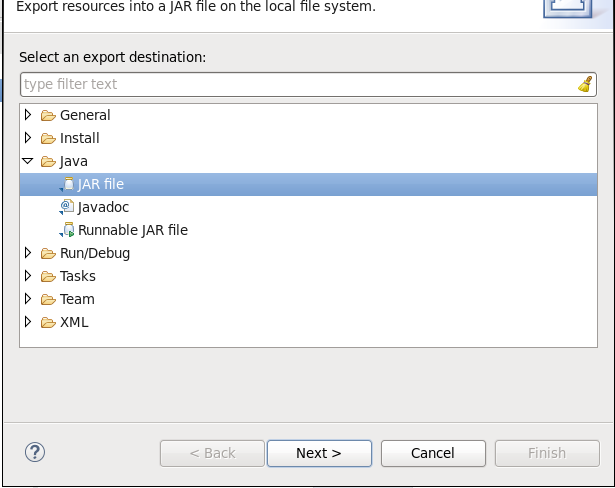
Create sample text files to use as input, and move them to the /user/cloudera/wordcount/input directory in HDFS. You can use any files you choose; for convenience, the following shell commands create a few small input files for illustrative purposes.

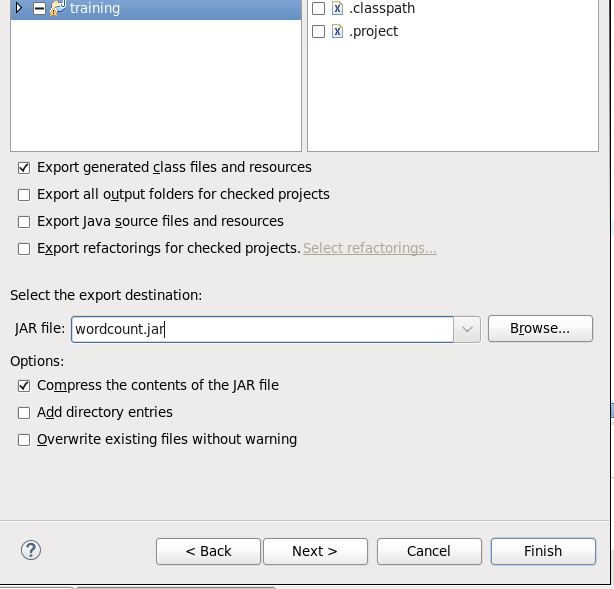
Compile WordCount class.

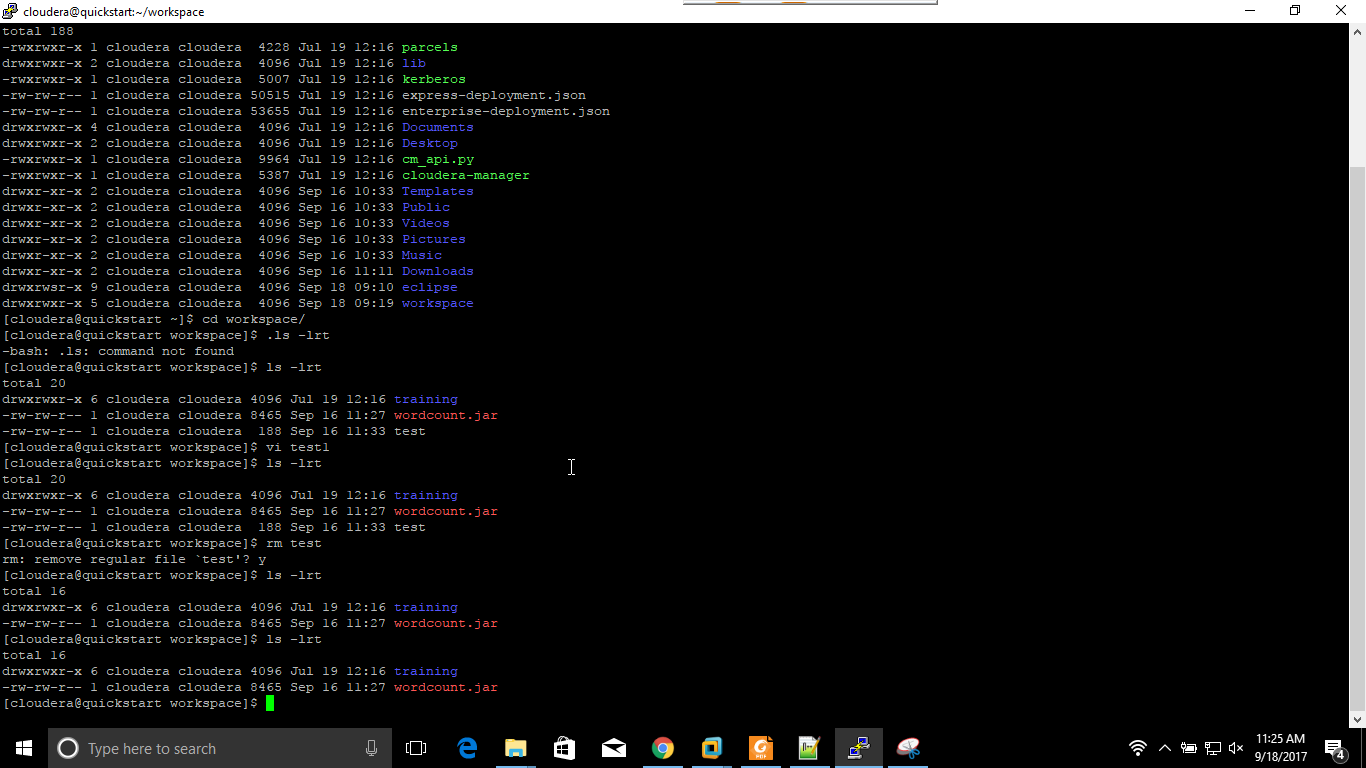
In Eclipse, right click on your class file -> Choose Export



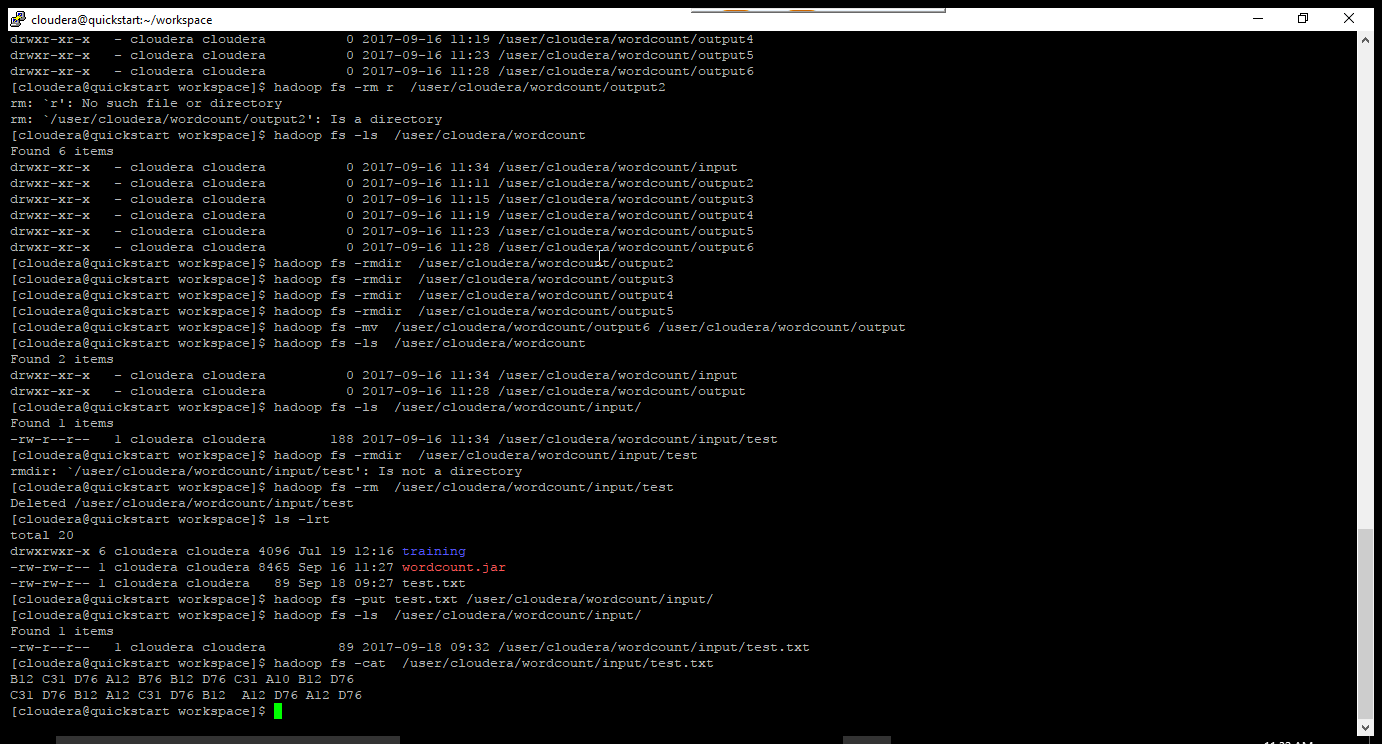
Select Java -> JAR file







Create input file on Hadoop

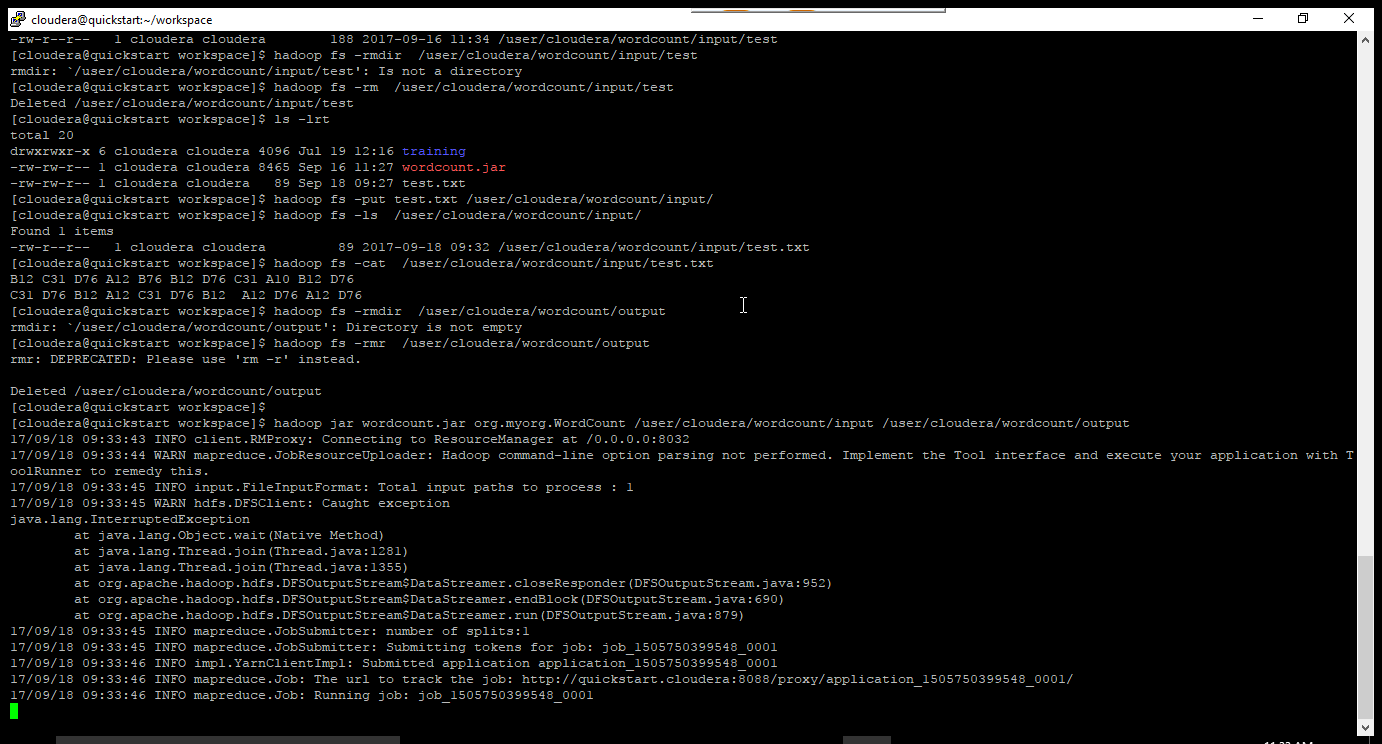


Select your project configuration and export destination. After that, click Finish.

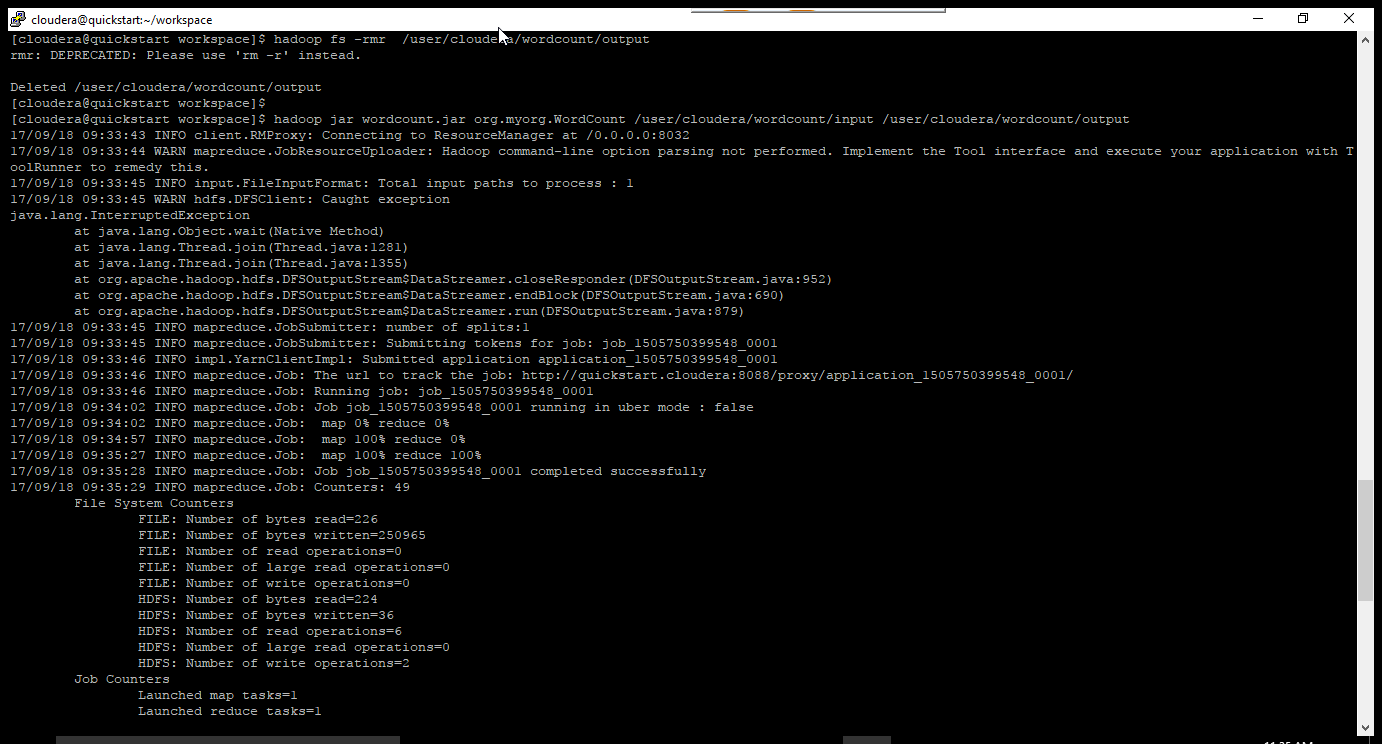
To run the WordCount application from JAR file, passing the paths to the input and output directories in HDFS.

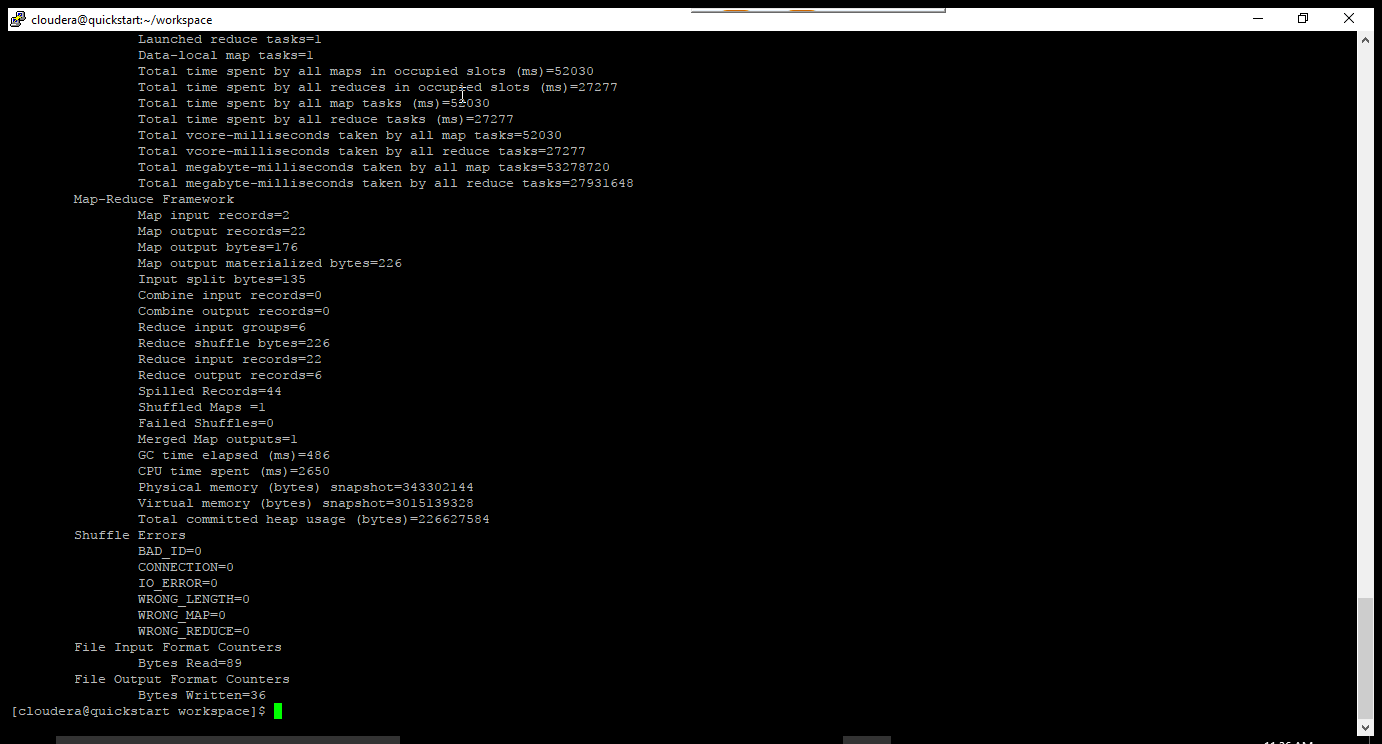
Run command

$ hadoop jar wordcount.jar org.myorg.WordCount /user/cloudera/wordcount/input /user/cloudera/wordcount/output



You can see some output result as below





Output is in attachments