

Cloudera CDH 5.8 Installation

On Oracle Virtual Box



Prerequisites:

- 1) 64bit Host Operating System.
- 2) Minimum Recommended Memory for Cloudera is 4GB*
- 3) Oracle Virtual Box on your Computer

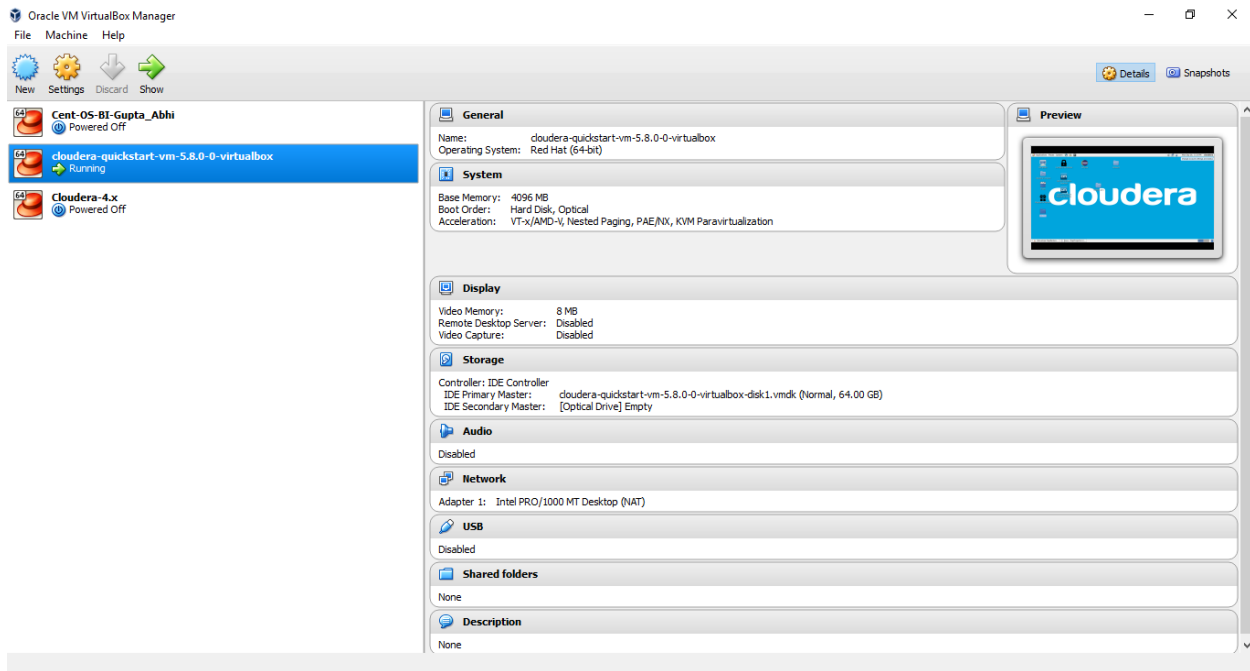
Download Cloudera from: http://www.cloudera.com/downloads/quickstart_vms/5-8.html

Step 1:

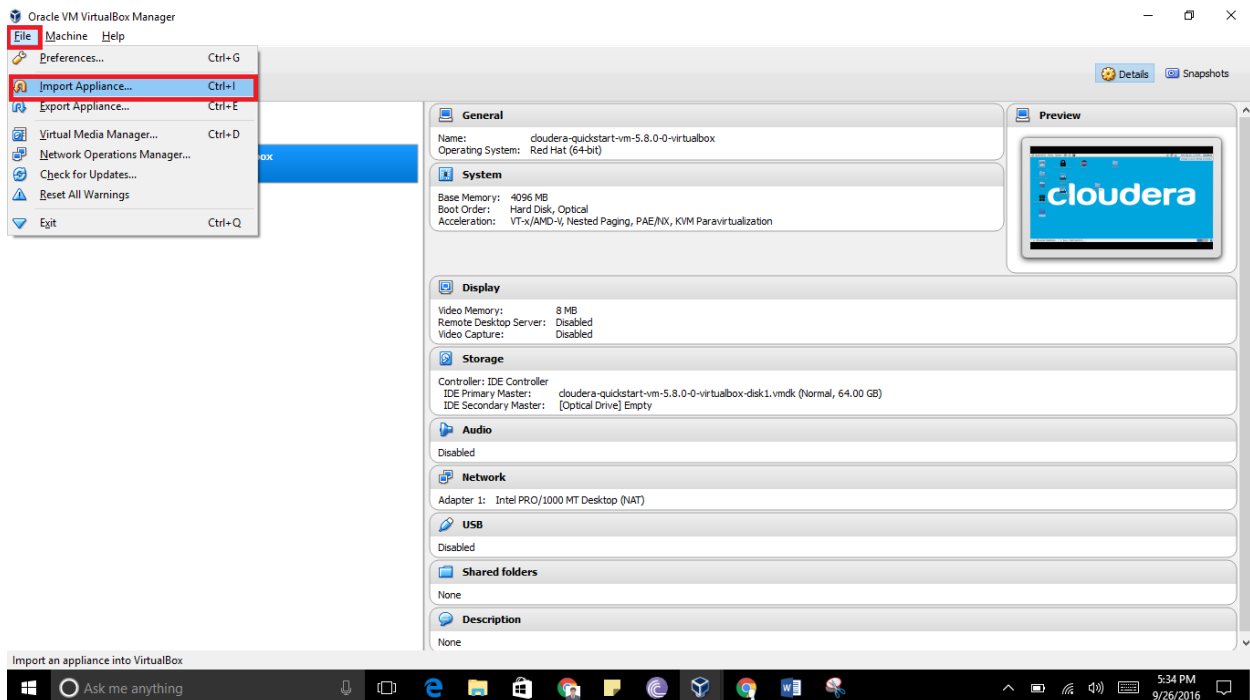
After Downloading Unzip the folder. You will find these two files in your folder. Make sure you have virtualbox.ovf (Open Virtualization Format Package) file present in the folder.

| <input type="checkbox"/> | Name | Date modified | Type | Size |
|---|--|-------------------|------------------------------------|--------------|
|  | cloudera-quickstart-vm-5.8.0-0-virtualbox.ovf | 9/24/2016 5:53 PM | Open Virtualization Format Package | 15 KB |
|  | cloudera-quickstart-vm-5.8.0-0-virtualbox-disk1.vmdk | 9/24/2016 5:53 PM | VMware virtual disk file | 5,199,931 KB |

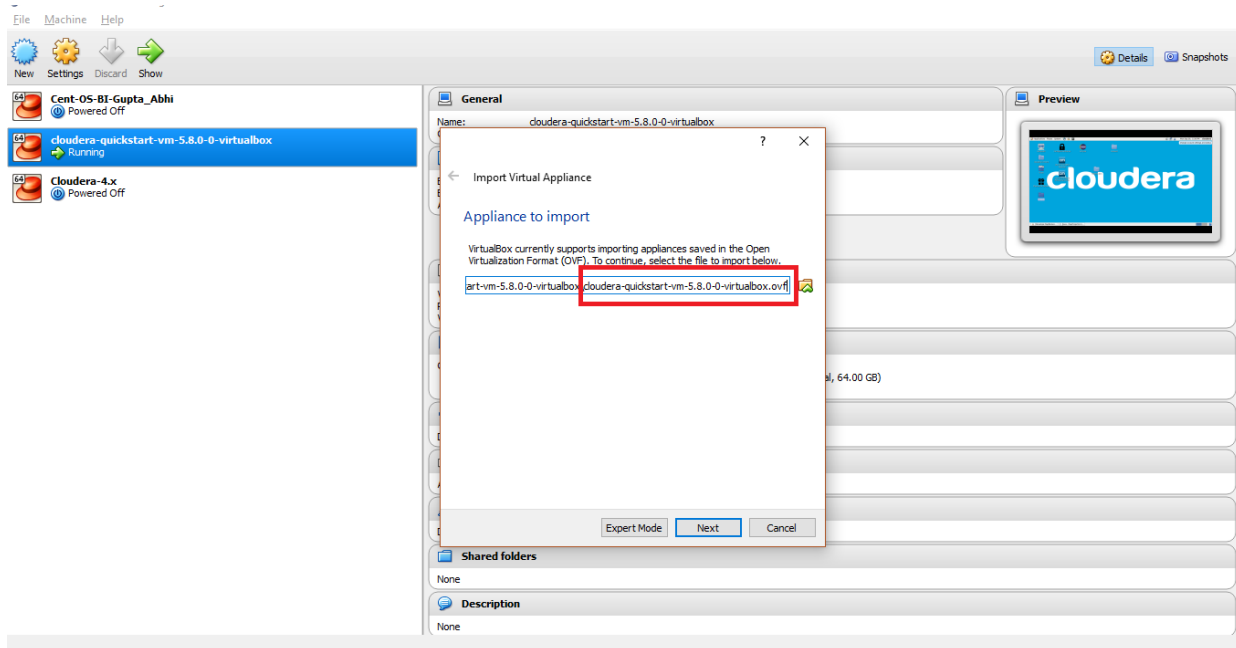
Step 2: Open Virtual Box:



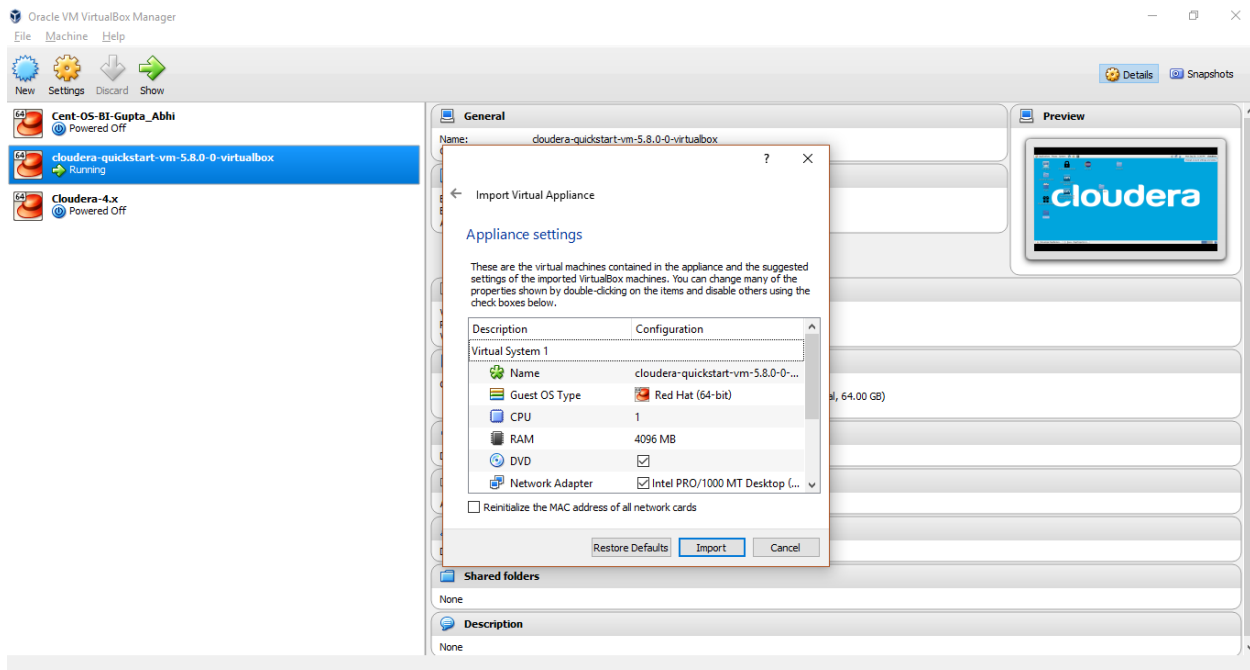
Step 3: Click on file → Import Appliance



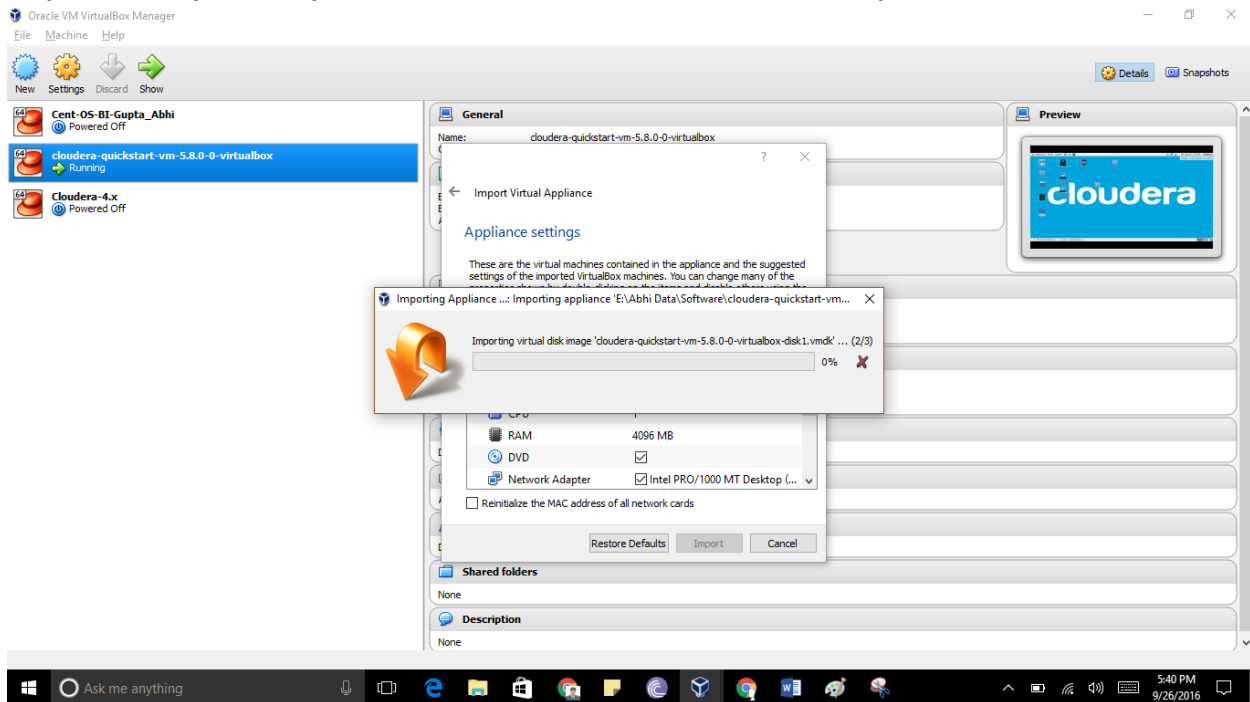
Step 4: In the Pop-up box select the .ovf file from the folder where you have unzipped the Cloudera.



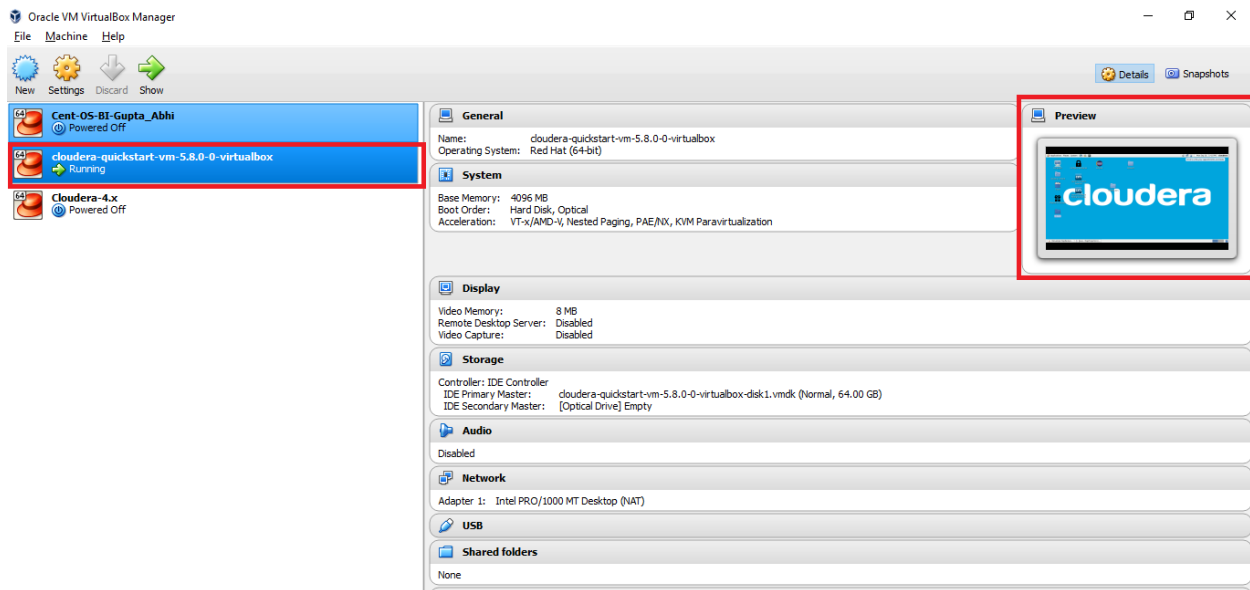
Step 5: Click Next by choosing next button. The below popup will show the Setting of your new virtual Cloudera Machine.



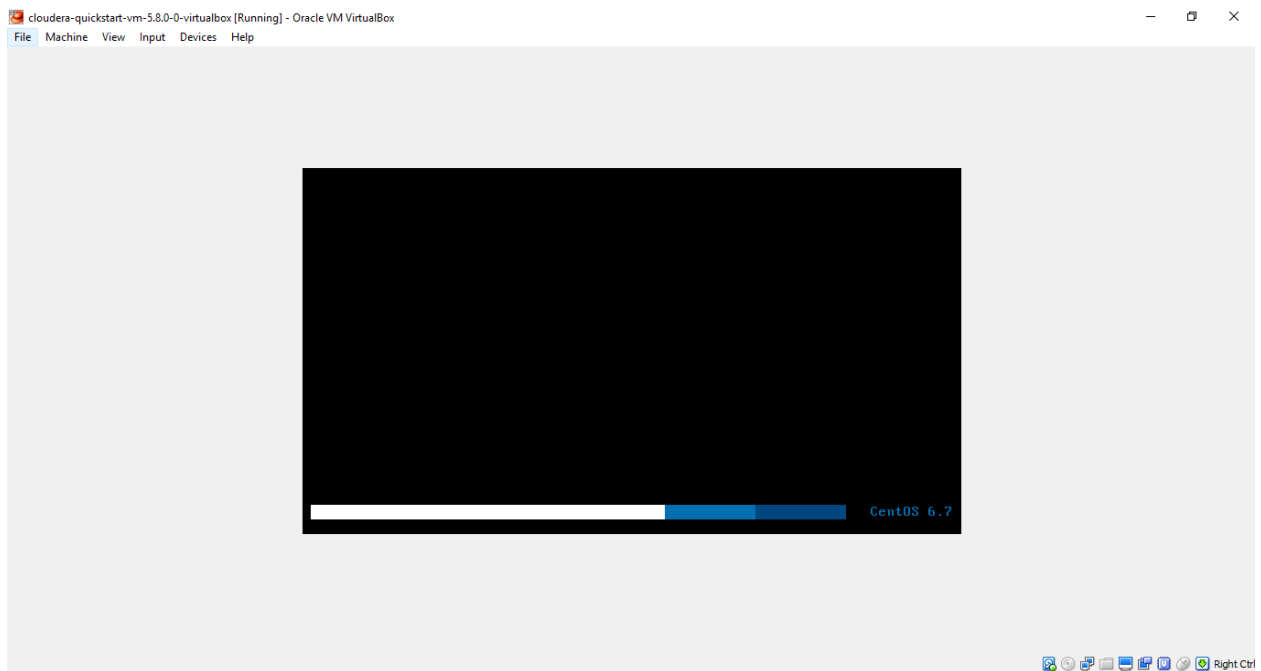
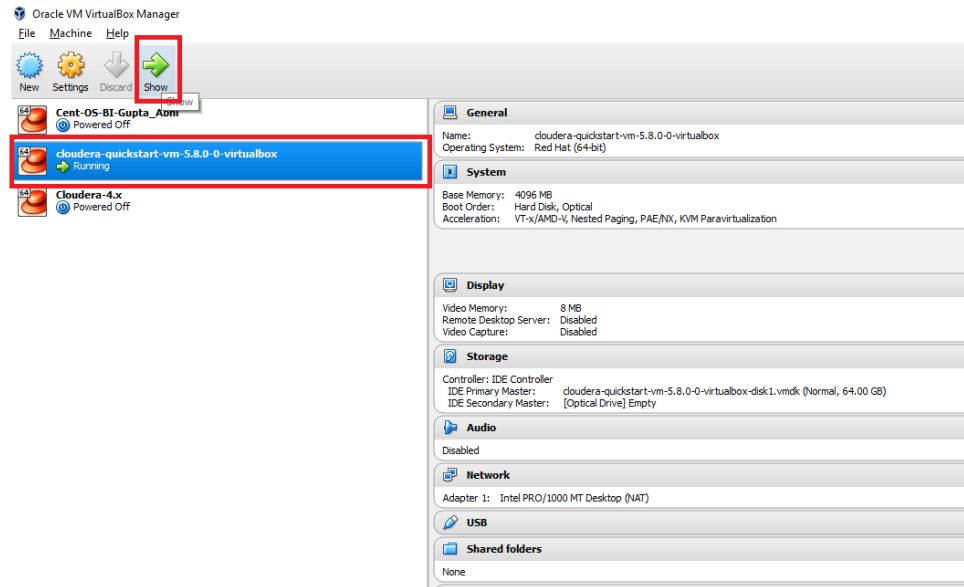
Step 6: Click Import. The process will take almost 45-60 Minutes to complete.

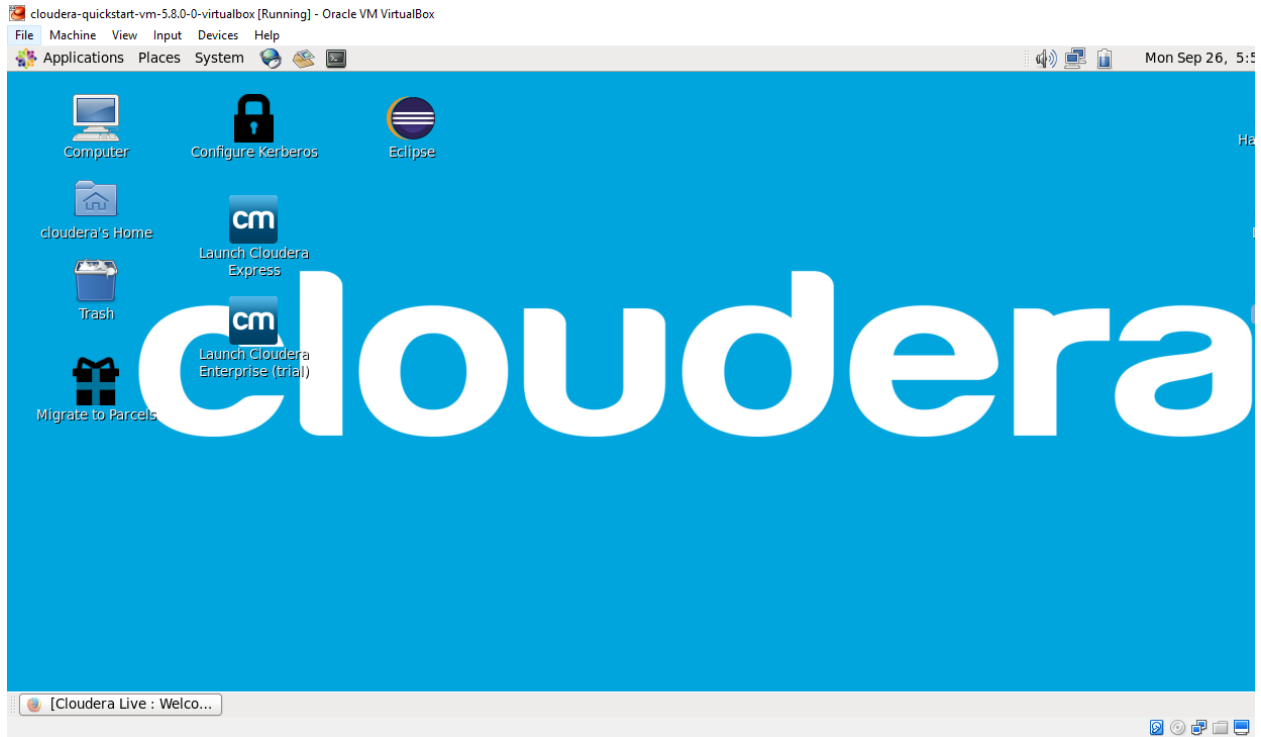


Step 7: After Completion you will find the below virtual machine in your Oracle virtual box.



Step 8: Select the Machine and click on Start. It will take a little time to start

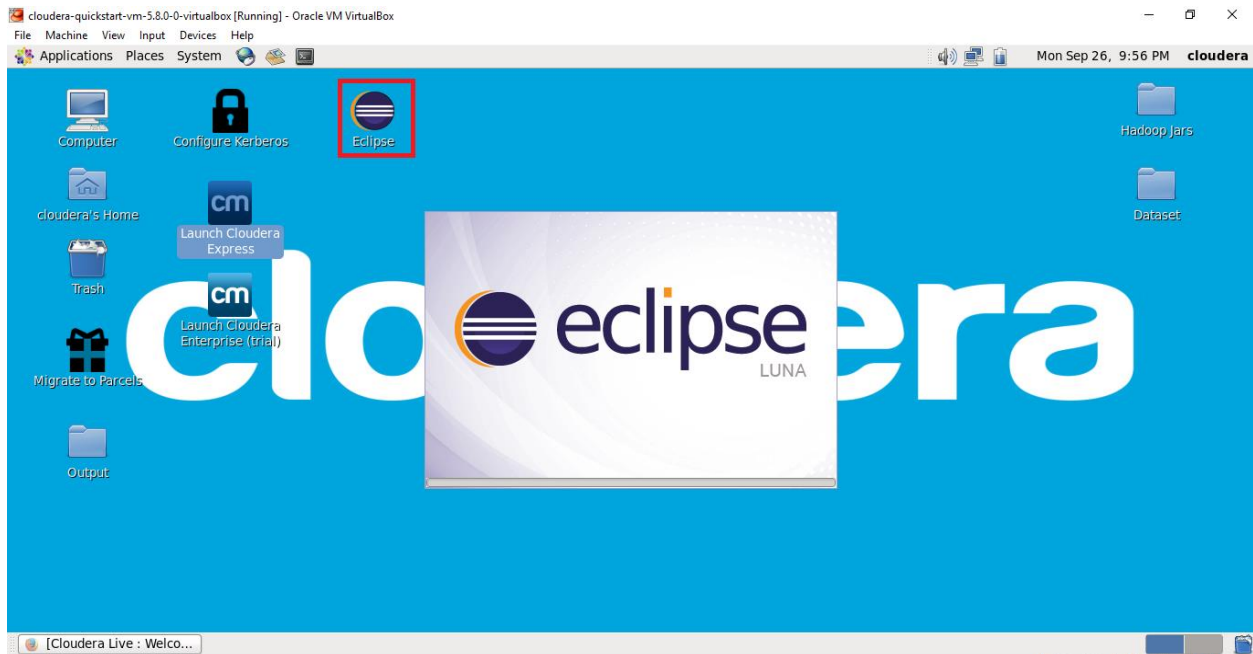




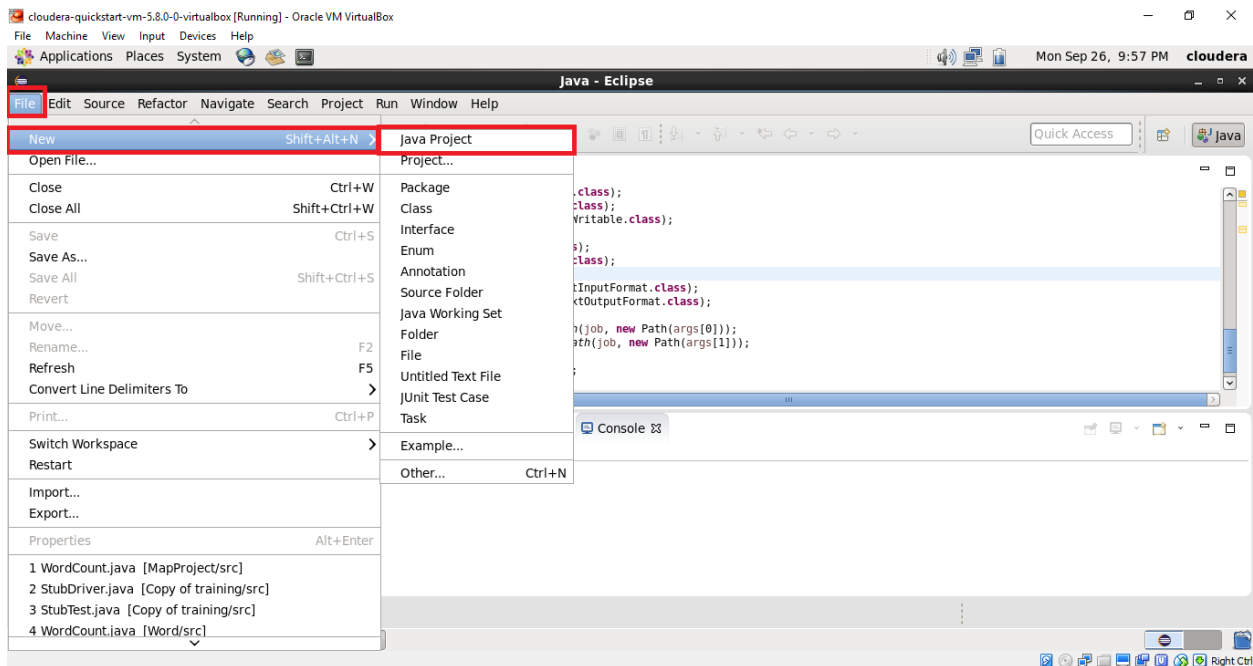
This ends the installation of Cloudera CDH5 version on Oracle Virtual box

How to Create Map Reduce Program in Eclipse

Step 1: Go to Desktop and open Eclipse



Step 2: Go to menu bar and click on File → New → Java Project.



Step 3: Give the name for the project and click next.

New Java Project

Create a Java project in the workspace or in an external location.

Project name:

☒ Use default location

Location: [Browse...](#)

JRE

☒ Use an execution environment JRE:

☐ Use a project specific JRE:

☐ Use default JRE (currently 'jdk1.7.0_67-cloudera') [Configure JREs...](#)

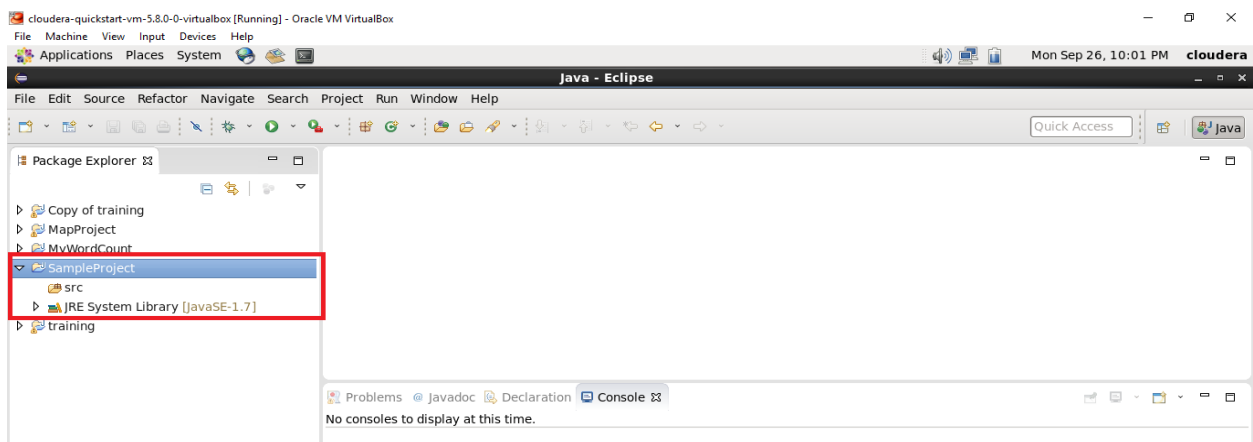
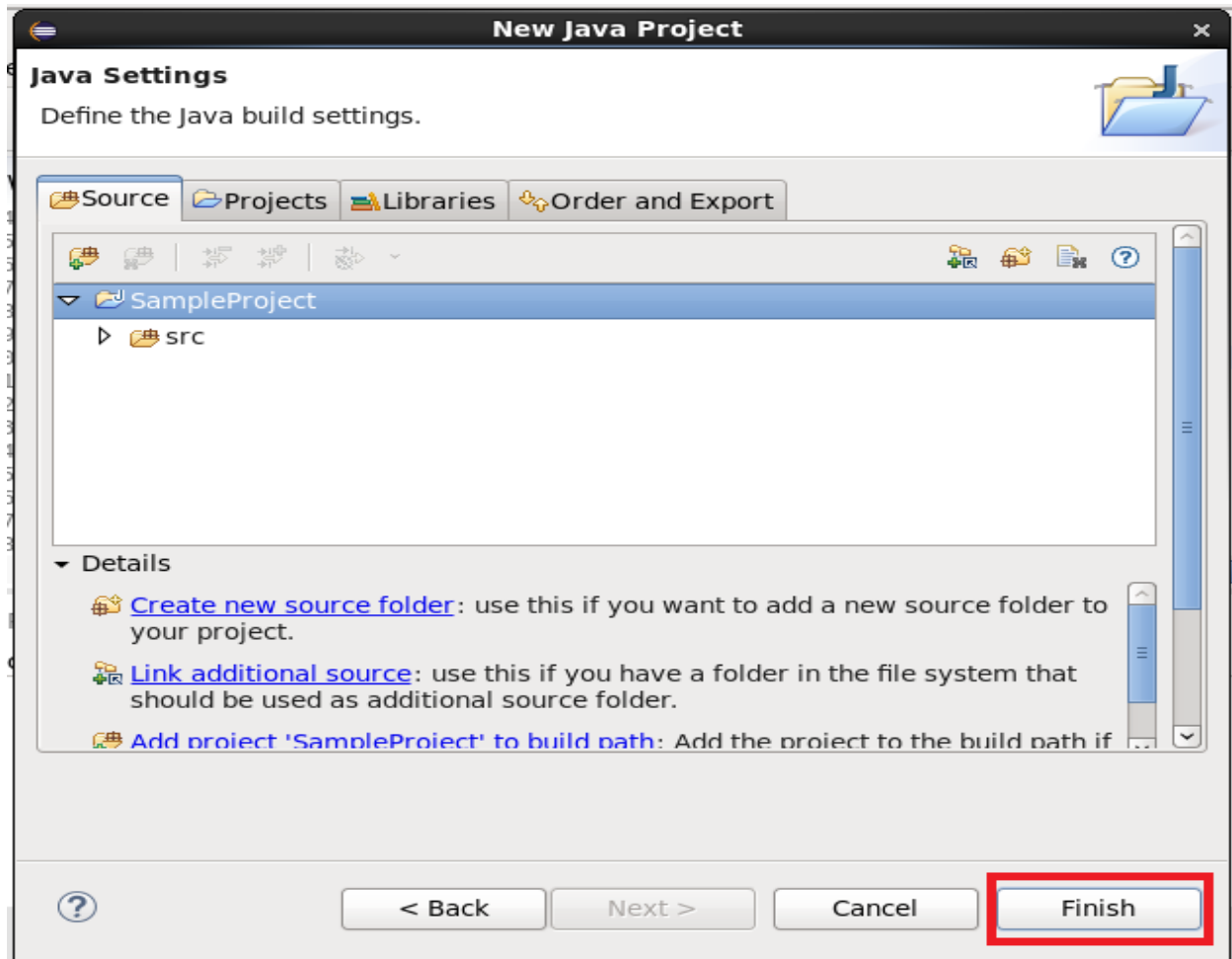
Project layout

☐ Use project folder as root for sources and class files

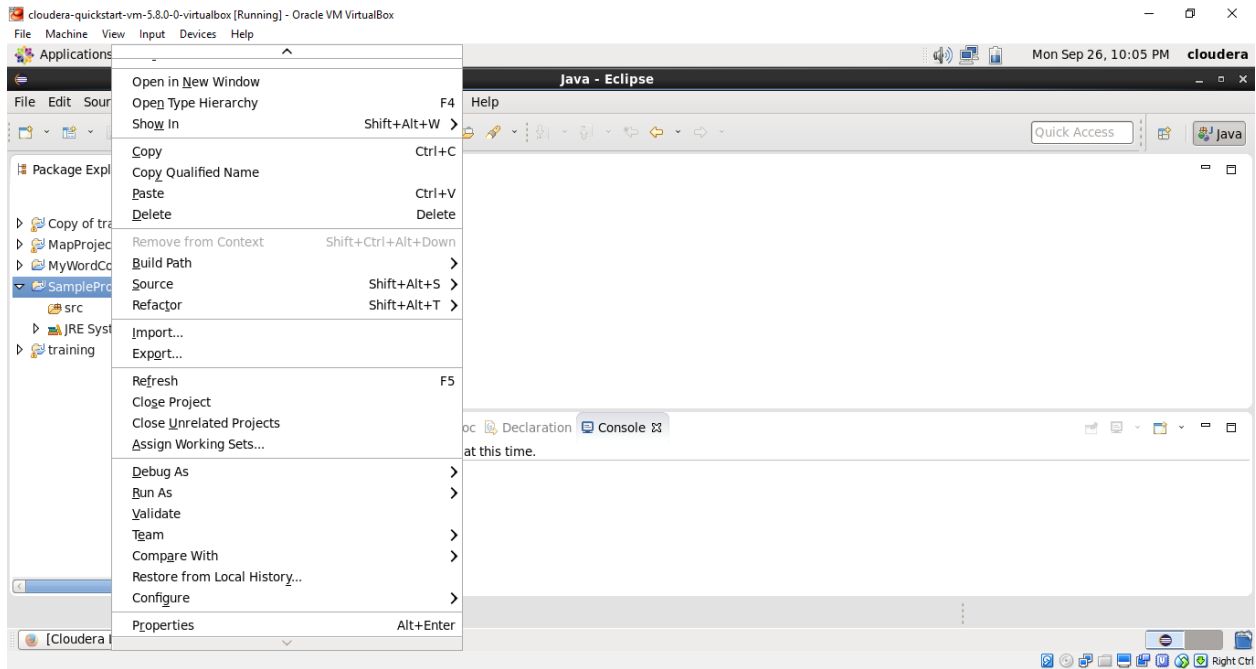
☒ Create separate folders for sources and class files [Configure default...](#)

[?](#) [< Back](#) **[Next >](#)** [Cancel](#) [Finish](#)

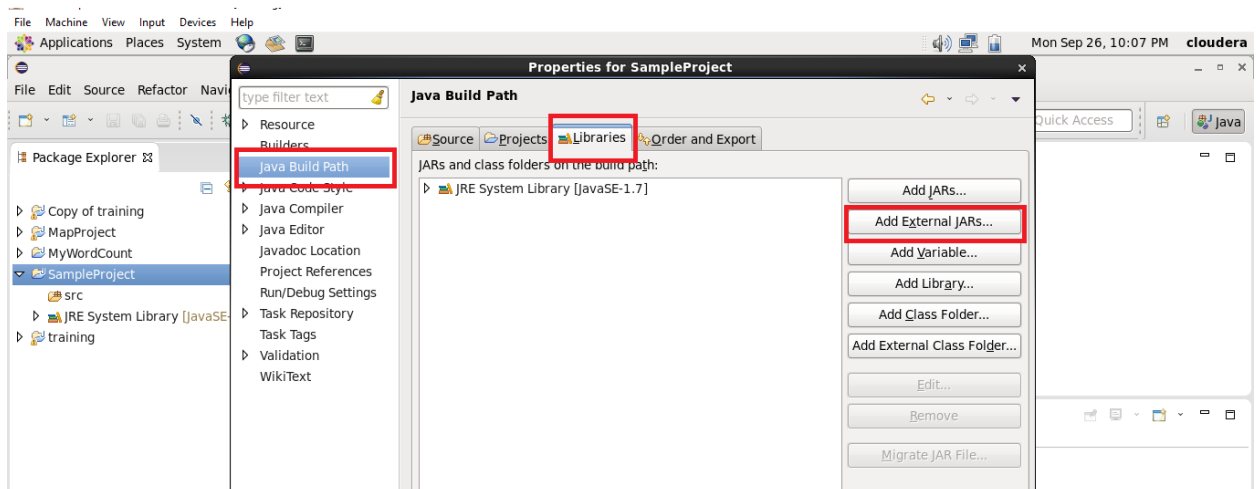
Step 4: Click Finish. After that you will find your project in Package Explorer.



Step 5: Now It's Time to add Hadoop jars, that are required to run MapReduce project. Right Click on the project and click properties



Step 6: Go to Java Build Path, Click on Libraries and then Click Add External Jars



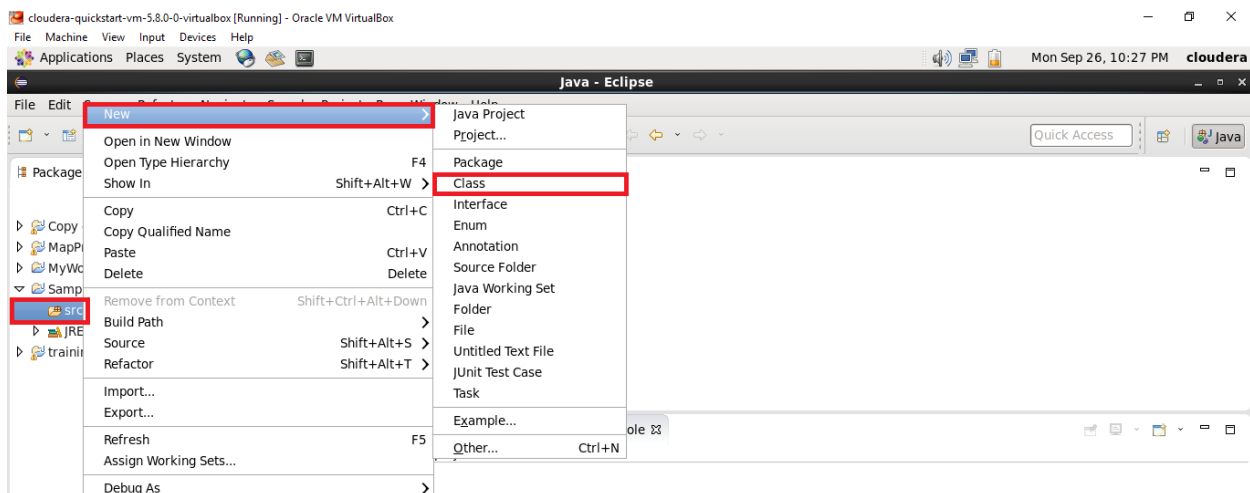
Step 7: Add the following Jars by going to these locations:

Usr/lib/Hadoop/Client-0.20 - All Jars from this folder

Usr/lib/Hadoop/ 3 jars from this folder 1) Hadoop-Annotations.jar 2) Hadoop-Auth.jar 3) Hadoop-common.jar

Usr/lib/Hadoop/lib -only one jar - commons-httpclient-3.1.jar

Step 8: Create a java class in SRC folder and give the mane of the class as WordCount and click finish.



Java Class

⚠ The use of the default package is discouraged.

Source folder:

Package:

☐ Enclosing type:

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass:

Interfaces:

Which method stubs would you like to create?

☐ public static void main(String[] args)

☐ Constructors from superclass

Step 9: Copy and paste the below code in the class.

```
import java.io.IOException; import java.util.*;
import java.lang.Object;

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 static final int UPPER_CASE = 0;
    private static final int LOWER_CASE = 0;
    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);

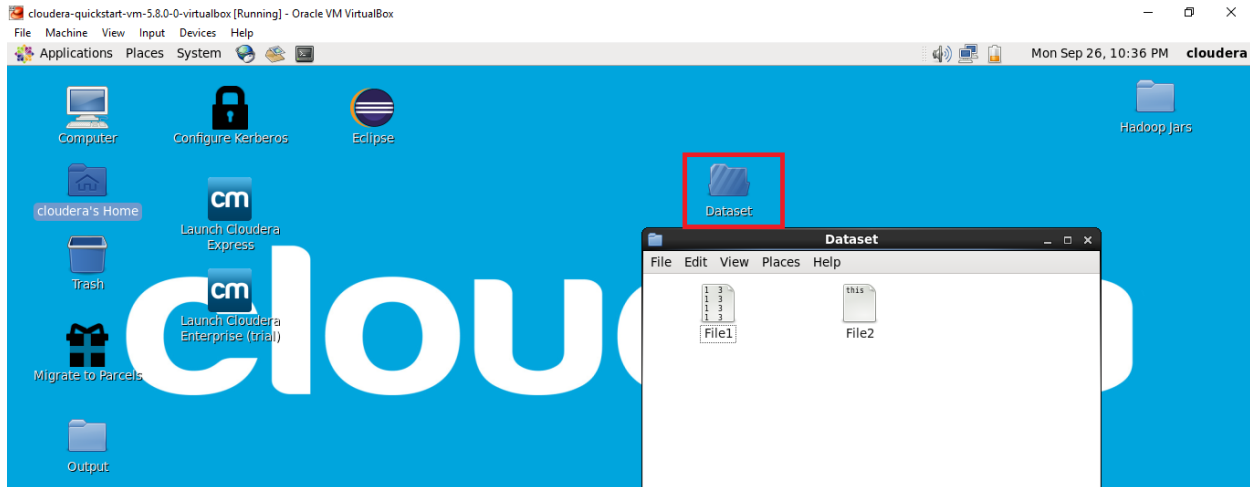
    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);

    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));

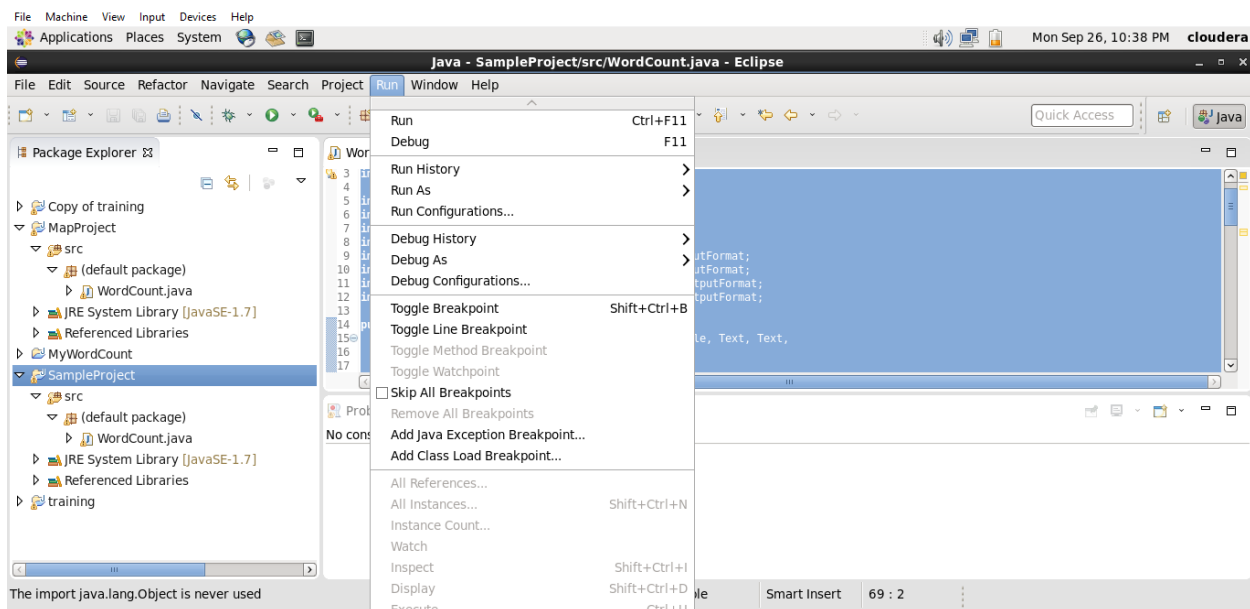
    job.waitForCompletion(true);
}
}

```

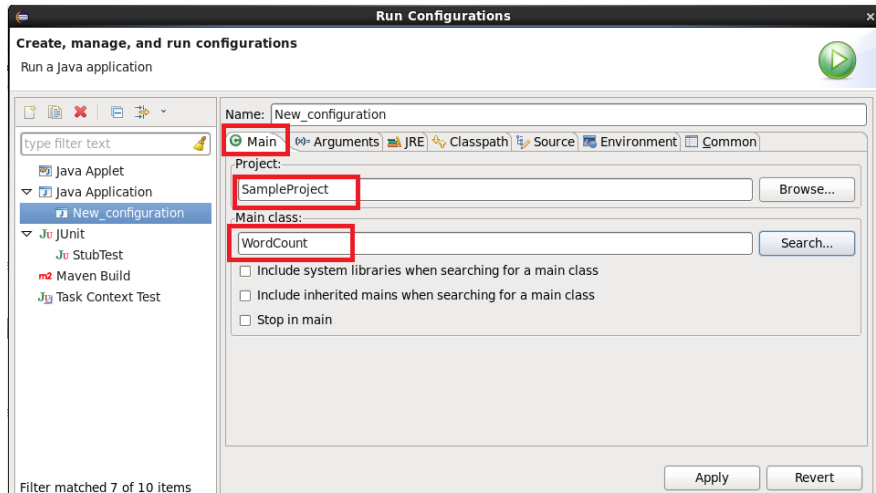
Step 10: Create a folder on Desktop and create multiple input file in the folder which contains different text.



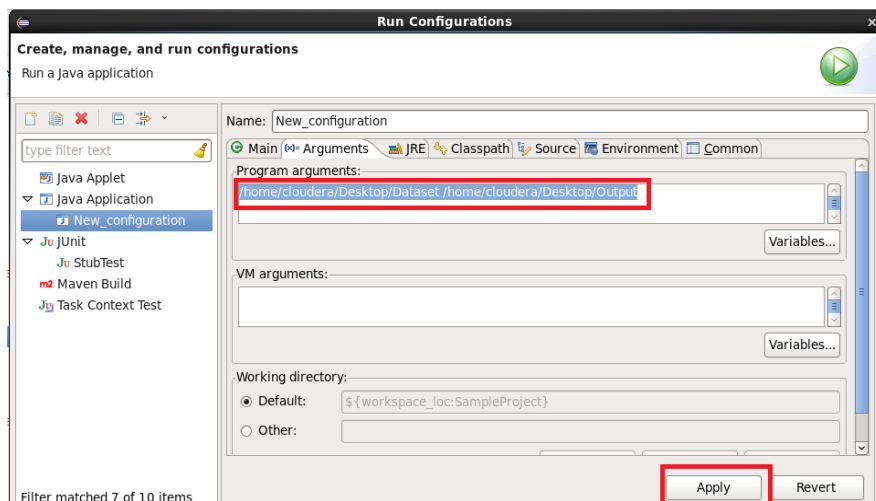
Step 11: Go back to Eclipse and click on run from menu bar and select Run Configuration



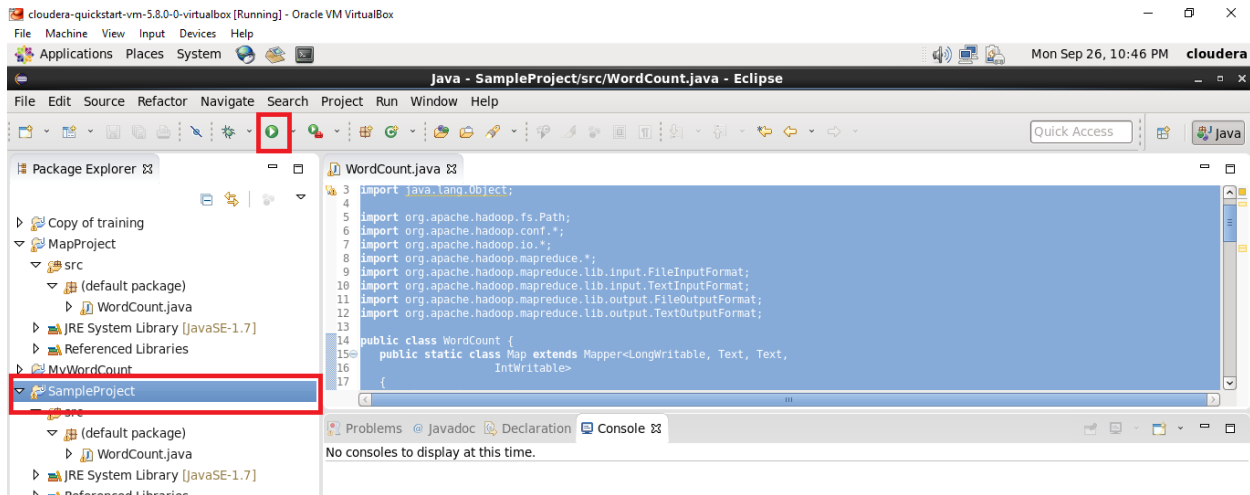
Step 12: In Main section select the project that we have created and also select the main class from that project.



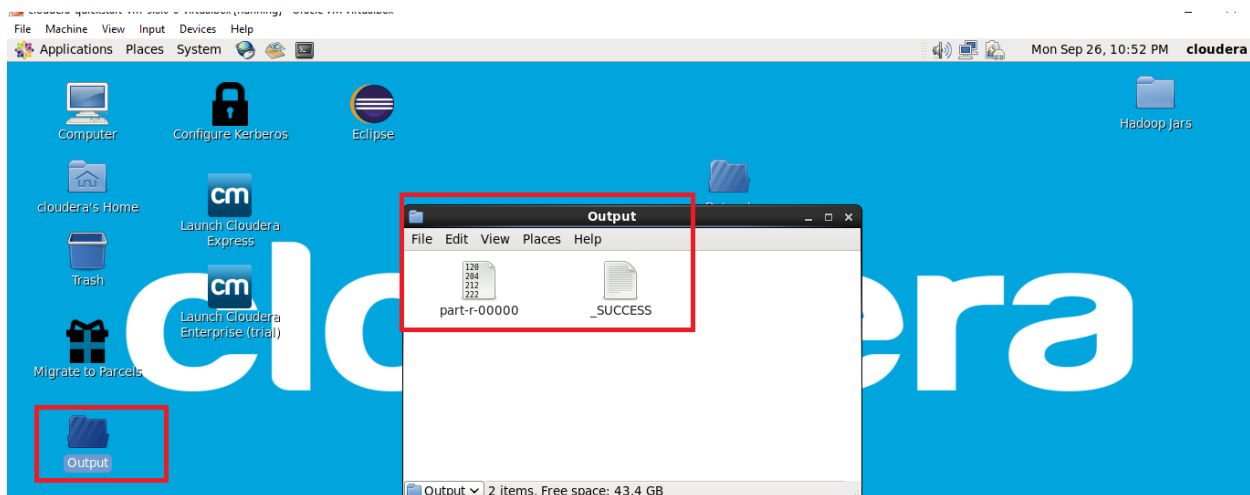
Step 13: Go to argument tab and provide the path of input and output folder separated by space (Do not create output folder It will be automatically created in the provided path) and then click apply
home/cloudera/Desktop/Dataset home/cloudera/Desktop/Output



Step 14: Run the program by selecting the project and clicking the run button.



Step 15: If there are no errors in the program you will be able to see the output folder in the provided location with the following output files.



For more details visit

<http://blog.cloudera.com/blog/2013/08/how-to-use-eclipse-with-mapreduce-in-clouderas-quickstart-vm/>