BIG DATA ANALYTICS LAB

EXPERIMENT_NO-03

AIM: Write driver code, mapper code, reducer code to count number of words in a given file. (Hint: WordCount Map- Reduce Program)

Description:

1) Open Oracle VM VirtualBox->export cloudera->start

2) Cloudera->settings->system->set processors to “2”, by default it is “1”.

3) To launch “cloudera Express”

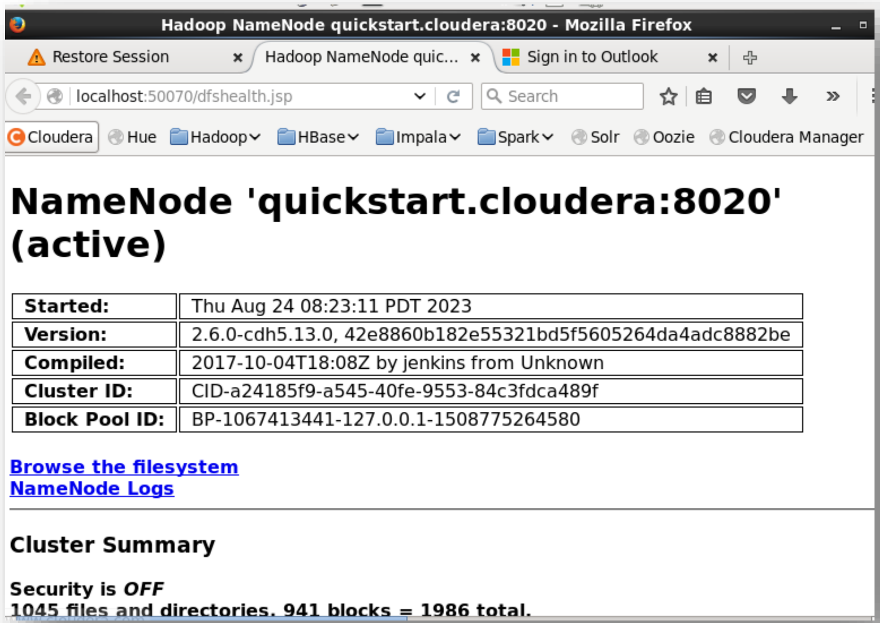
   (i) Open terminal in cloudera and start the server by using

   “sudo service cloudera-sdh-server start”

   (ii) After successful completion click on Cloudera Express->Cloud Manager

   (iii) Both username and password is “cloudera” and then click on Login

4) In browser type “localhost:50070/dfshealth.jsp”



NameNode 'quickstart.cloudera:8020' (active)

Started:	Thu Aug 24 08:23:11 PDT 2023
Version:	2.6.0-cdh5.13.0, 42e8860b182e55321bd5f5605264da4adc8882be
Compiled:	2017-10-04T18:08Z by jenkins from Unknown
Cluster ID:	CID-a24185f9-a545-40fe-9553-84c3fdca489f
Block Pool ID:	BP-1067413441-127.0.0.1-1508775264580

Browse the filesystem
NameNode Logs

Cluster Summary

Security is *OFF*
1045 files and directories, 941 blocks = 1986 total.

5) In eclipse->File->New->Java project->Project name “WordCount”->Finish

6) Create three classes.

Right click on WordCount -> New->class->Name “WCDriver”

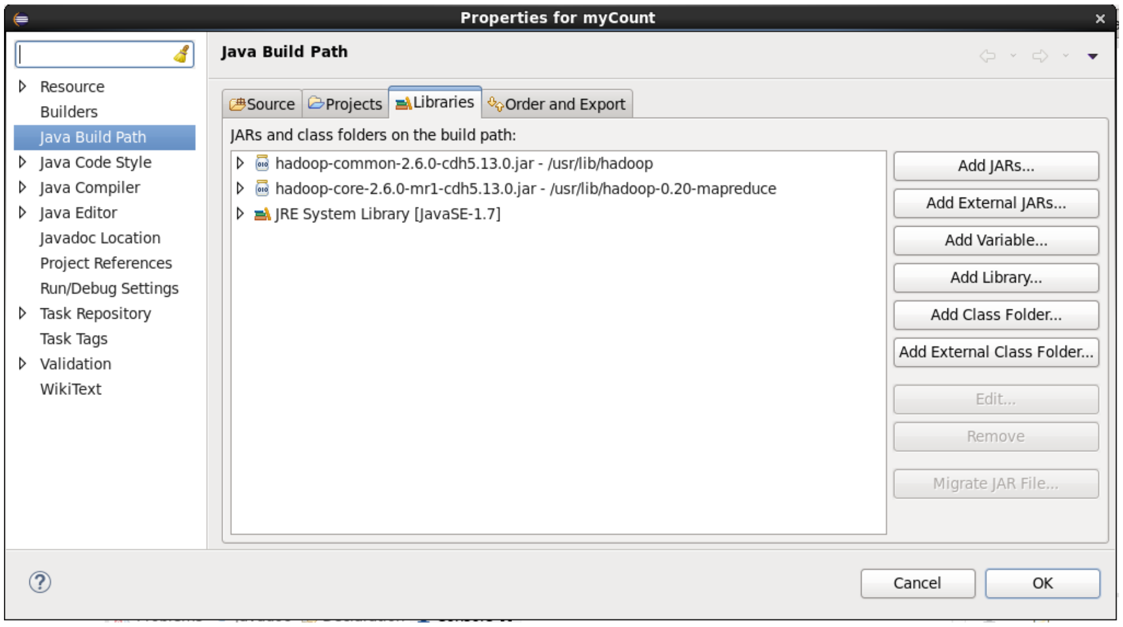
Right click on WordCount -> New->class->Name “WCMapper”

Right click on WordCount -> New->class->Name “WCReducer”

7) Add Hadoop libraries.

Right click on WordCount ->Build path->Configure Build path->Add external JARS. (usr\lib\hadoop\hadoop-common-2.6.0-cdh 5.13.0 jar,

Usr\lib\hadoop\hadoop-core-2.6.0-cdh 5.13.0 jar)



Java Build Path

JARs and class folders on the build path:

hadoop-common-2.6.0-cdh5.13.0.jar - /usr/lib/hadoop
hadoop-core-2.6.0-mr1-cdh5.13.0.jar - /usr/lib/hadoop-0.20-mapreduce
JRE System Library [JavaSE-1.7]

PROGRAM:

WCMapper.java

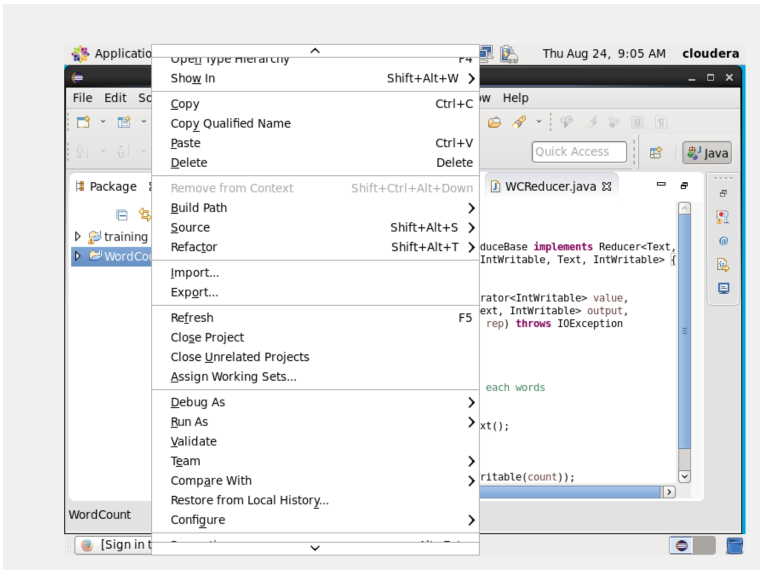
```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;

public class WCMapper extends MapReduceBase implements Mapper<LongWritable,
                                                Text, Text, IntWritable> {

    // Map function
    public void map(LongWritable key, Text value, OutputCollector<Text,
                 IntWritable> output, Reporter rep) throws IOException
    {

        String line = value.toString();

        // Splitting the line on spaces
        for (String word : line.split(" "))
        {
            if (word.length() > 0)
            {
                output.collect(new Text(word), new IntWritable(1));
            }
        }
    }
}
```

WCReducer.java

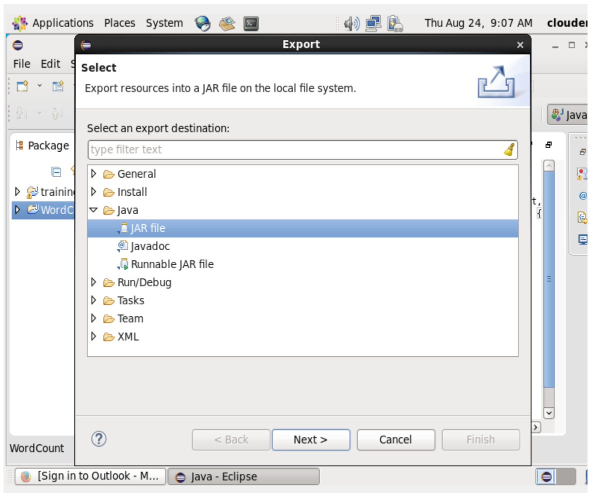
```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;

public class WCReducer extends MapReduceBase implements Reducer<Text,
                                    IntWritable, Text, IntWritable> {

    // Reduce function
    public void reduce(Text key, Iterator<IntWritable> value,
                   OutputCollector<Text, IntWritable> output,
                            Reporter rep) throws IOException
    {

        int count = 0;
        // Counting the frequency of each words
        while (value.hasNext())
        {
            IntWritable i = value.next();
            count += i.get();
        }

        output.collect(key, new IntWritable(count));
    }
}
```

WCDriver.java

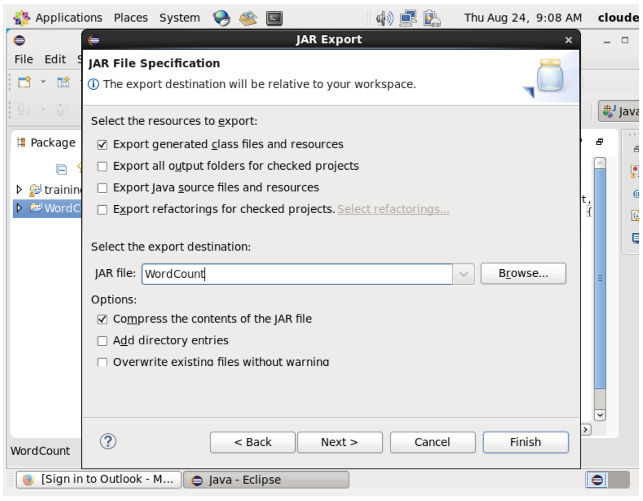
```
import java.io.IOException;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class WCDriver extends Configured implements Tool {

    public int run(String args[]) throws IOException
    {
        if (args.length < 2)
        {
            System.out.println("Please give valid inputs");
            return -1;
        }

        JobConf conf = new JobConf(WCDriver.class);
        FileInputFormat.setInputPaths(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        conf.setMapperClass(WCMapper.class);
        conf.setReducerClass(WCReducer.class);
        conf.setMapOutputKeyClass(Text.class);
        conf.setMapOutputValueClass(IntWritable.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        JobClient.runJob(conf);
        return 0;
    }

    // Main Method
    public static void main(String args[]) throws Exception
    {
        int exitCode = ToolRunner.run(new WCDriver(), args);
        System.out.println(exitCode);
    }
}
```



Right click on WordCount->Export->java->jar file->JAR file:” WordCount”->Finish



Export resources into a JAR file on the local file system.



The export destination will be relative to your workspace.

Select the resources to export:

- [✓] Export generated class files and resources
- [ ] Export all output folders for checked projects
- [ ] Export Java source files and resources
- [ ] Export refactorings for checked projects. Select refactorings...

Select the export destination:

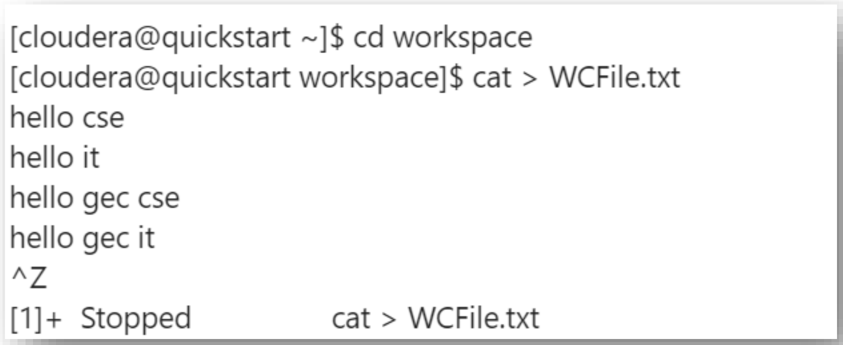
JAR file: WordCount

Options:

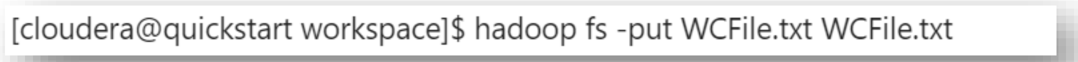
- [✓] Compress the contents of the JAR file
- [ ] Add directory entries
- [ ] Overwrite existing files without warning

OUTPUT:

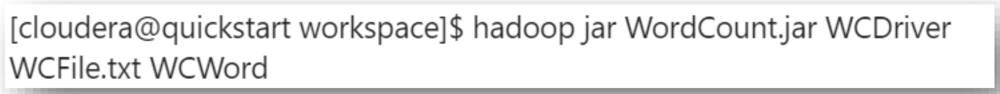
In Terminal



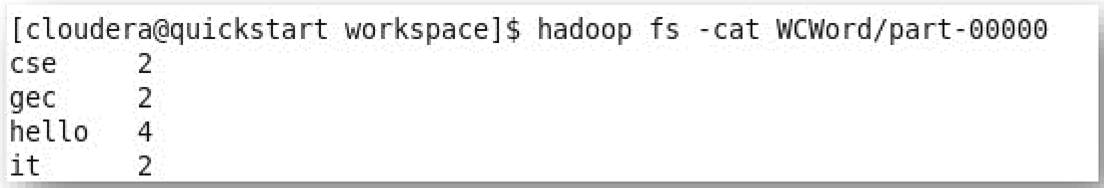
```
[cloudera@quickstart ~]$ cd workspace
[cloudera@quickstart workspace]$ cat > WCFile.txt
hello cse
hello it
hello gec cse
hello gec it
^Z
[1]+  Stopped                 cat > WCFile.txt
```



```
[cloudera@quickstart workspace]$ hadoop fs -put WCFile.txt WCFile.txt
```



```
[cloudera@quickstart workspace]$ hadoop jar WordCount.jar WCDriver
WCFile.txt WCWord
```



```
[cloudera@quickstart workspace]$ hadoop fs -cat WCWord/part-00000
cse     2
gec     2
hello   4
it      2
```