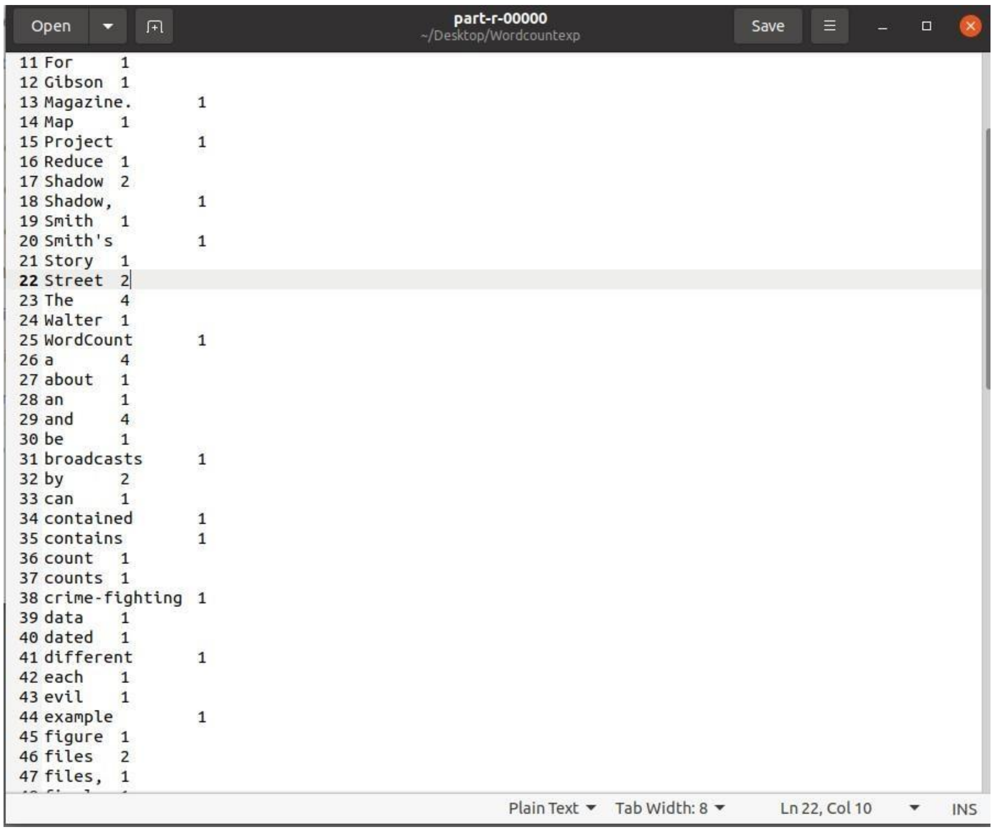
GROUP B
EXPERIMENT: 1

WordCount.java

```
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 WordCount {

 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 = Job.getInstance(conf, "word count");
  job.setJarByClass(WordCount.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);
  }
}
```

output: output file (part-r-00000)



```
For     1
Gibson  1
Magazine.       1
Map     1
Project         1
Reduce  1
Shadow  2
Shadow,         1
Smith   1
Smith's         1
Story   1
Street  2
The     4
Walter  1
WordCount       1
a       4
about   1
an      1
and     4
be      1
broadcasts      1
by      2
can     1
contained       1
contains        1
count   1
counts  1
crime-fighting  1
data    1
dated   1
different       1
each    1
evil    1
example         1
figure  1
files   2
files,  1
```

GROUP B
EXPERIMENT: 2

Code:

1> <u>LogFileMapper.java</u> (Use for mapping the IP addresses from input csv file**)**

```
package LogFileCountry;
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.*;

public class LogFileMapper extends MapReduceBase implements Mapper<LongWritable, Text, Text,
IntWritable> {
        private final static IntWritable one = new IntWritable(1);

        public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter
reporter) throws IOException {

                String valueString = value.toString();
                String[] SingleIpData = valueString.split("-");
                output.collect(new Text(SingleIpData[0]), one);
        }
}
```

2><u>LogFileReduce.java</u> (Use for reducing data received from mapper process to final output)

```
package LogFileCountry;
import java.io.IOException;
import java.util.*;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.*;

public class LogFileReducer extends MapReduceBase implements Reducer<Text, IntWritable, Text,
IntWritable> {public void reduce(Text t_key, Iterator<IntWritable> values, OutputCollector<Text,IntWritable>
output, Reporter reporter) throws IOException {
                Text key = t_key;
                int frequencyForIp = 0;
                while (values.hasNext()) {
                        // replace type of value with the actual type of our value
                        IntWritable value = (IntWritable) values.next();
                        frequencyForIp += value.get();
                }
                output.collect(key, new IntWritable(frequencyForIp));
        }
}
```

3>**LogFileCountryDriver.java** (The driver code to run map-reduce on hdfs)

```
package LogFileCountry;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;

public class LogFileCountryDriver {
	public static void main(String[] args) {
		JobClient my_client = new JobClient();
		// Create a configuration object for the job
		JobConf job_conf = new JobConf(LogFileCountryDriver.class);

		// Set a name of the Job
		job_conf.setJobName("LogFileIP");
		// Specify data type of output key and value
		job_conf.setOutputKeyClass(Text.class);
		job_conf.setOutputValueClass(IntWritable.class);
		// Specify names of Mapper and Reducer Class
		job_conf.setMapperClass(LogFileCountry.LogFileMapper.class);
		job_conf.setReducerClass(LogFileCountry.LogFileReducer.class);

		// Specify formats of the data type of Input and output
		job_conf.setInputFormat(TextInputFormat.class);
		job_conf.setOutputFormat(TextOutputFormat.class);

		// Set input and output directories using command line arguments,
		//arg[0] = name of input directory on HDFS, and arg[1] = name of output directory to be
created to store the output file.

		FileInputFormat.setInputPaths(job_conf, new Path(args[0]));
		FileOutputFormat.setOutputPath(job_conf, new Path(args[1]));

		my_client.setConf(job_conf);
		try { // Run the job
			JobClient.runJob(job_conf);
		} catch (Exception e) {
			e.printStackTrace();
		}
	}
}
```

4> **log_file.txt** (Input file sample)

```
0.223.157.186 - - [15/Jul/2009:20:50:32 -0700] "GET /assets/js/the-associates.js HTTP/1.1" 304 -
10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/home-logo.png HTTP/1.1" 304 -
10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/dummy/primary-news-2.jpg HTTP/1.1" 304 -
10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/dummy/primary-news-1.jpg HTTP1.1" 304 -
10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/home-media-block-placeholder.jpg HTTP/1.1" 304
-
10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/dummy/secondary-news-4.jpg HTTP/1.1" 304 -
10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/loading.gif HTTP/1.1" 304 -
10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/search-button.gif HTTP/1.1" 304 –
```

5> **<u>Output</u>** (part-00000.txt On Hadoop) (sample)

```
10.1.1.236      7
10.1.181.142    14
10.1.232.31     5
10.10.55.142    14
10.102.101.66   1
10.103.184.104  1
10.103.190.81   53
10.103.63.29    1
10.104.73.51    1
10.105.160.183  1
10.108.91.151   15
10.109.21.76    1
10.11.131.40    1
10.111.71.20    8
10.112.227.184  6
10.114.74.30    1
10.115.118.78   1
10.117.224.230  1
```

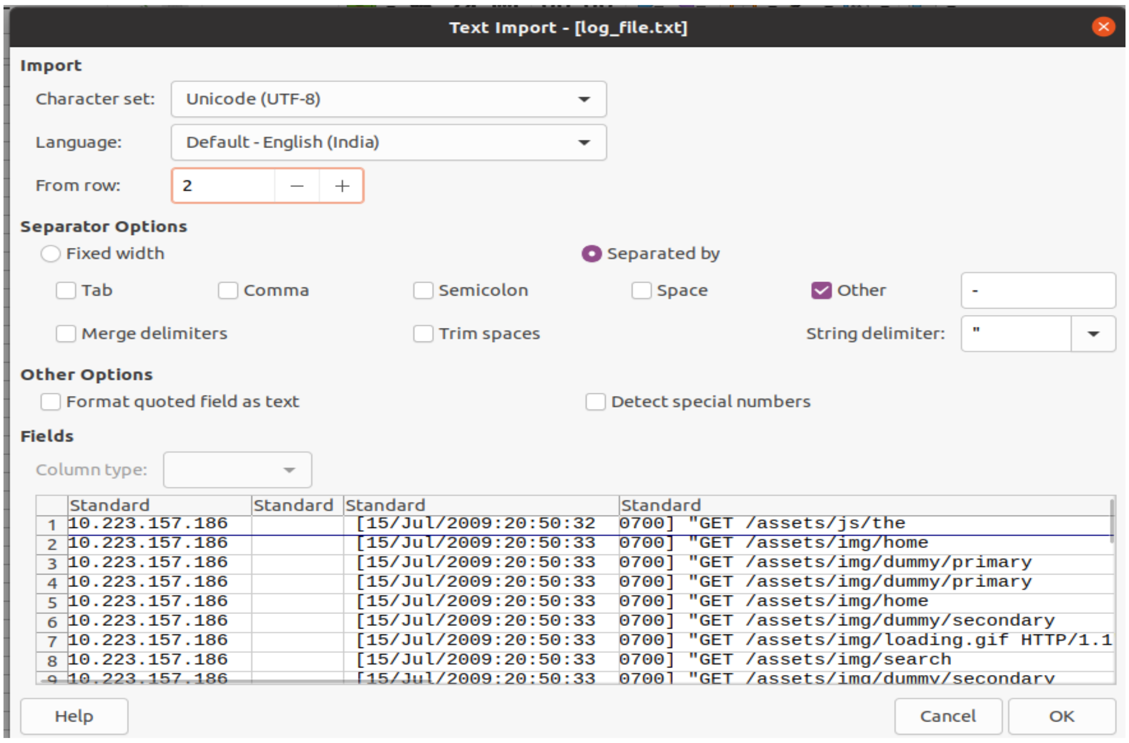
Step For Logs File Code:

1. Starting Hadoop and check if it is started.

   $ start-all.sh

2. Create folder “LogFileTut”. Copy the log_file.txt given and create the java files.
   i. LogFileMapper.java
   ii. LogFileReducer.java
   iii. LogFileCountryDriver.java

3. Convert the log_file.txt to .csv file. Open LibreOffice Calc-> Open -> log_file.txt. Save As .csv in the LogFileTut folder.



Text Import - [log_file.txt]

Import

Character set: Unicode (UTF-8)

Language: Default - English (India)

From row: 2

Separator Options

Fixed width [ ] [✓ Separated by]

Tab [ ] Comma [ ] Semicolon [ ] Space [ ] [✓ Other] -

Merge delimiters [ ] Trim spaces [ ] String delimiter: "

Other Options

Format quoted field as text [ ] Detect special numbers [ ]

Fields

Column type:

	Standard	Standard	Standard	Standard
1	10.223.157.186		[15/Jul/2009:20:50:32	0700] "GET /assets/js/the
2	10.223.157.186		[15/Jul/2009:20:50:33	0700] "GET /assets/img/home
3	10.223.157.186		[15/Jul/2009:20:50:33	0700] "GET /assets/img/dummy/primary
4	10.223.157.186		[15/Jul/2009:20:50:33	0700] "GET /assets/img/dummy/primary
5	10.223.157.186		[15/Jul/2009:20:50:33	0700] "GET /assets/img/home
6	10.223.157.186		[15/Jul/2009:20:50:33	0700] "GET /assets/img/dummy/secondary
7	10.223.157.186		[15/Jul/2009:20:50:33	0700] "GET /assets/img/loading.gif HTTP/1.1
8	10.223.157.186		[15/Jul/2009:20:50:33	0700] "GET /assets/img/search
9	10.223.157.186		[15/Jul/2009:20:50:33	0700] "GET /assets/img/dummy/secondary

4. Give Read permission to all the files in directories.

   $ sudo chmod +r *.*

5. Set HADOOP_CLASSPATH environment variable.

   $ export HADOOP_CLASSPATH=$(hadoop classpath)

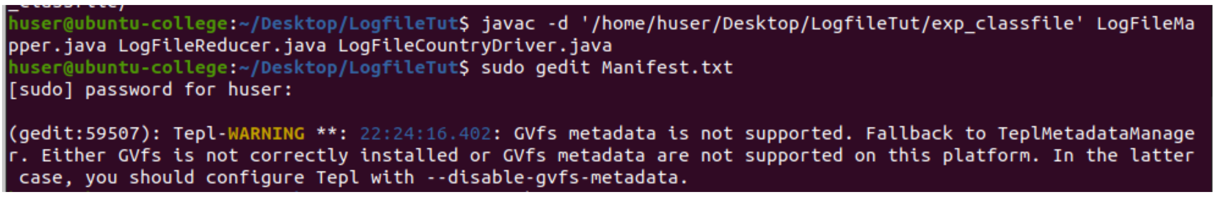
   or

   $ export CLASSPATH= "$HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-client-core-3.2.2.jar: $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-client-common-3.2.2.jar: $HADOOP_HOME/share/hadoop/common/hadoop-common-3.2.2.jar: $HADOOP_HOME/lib/*: ~/home/huser/Desktop/LogFileTut/*"

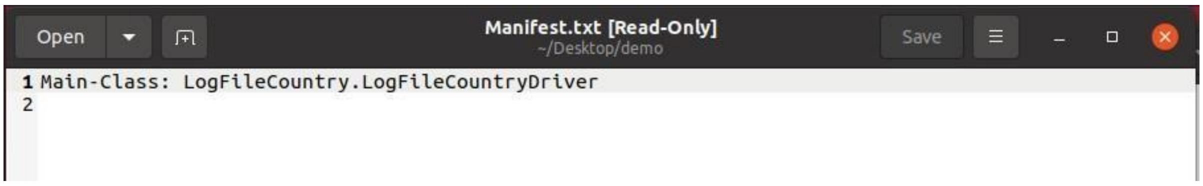
6. Compile the java code:

   $ javac -classpath $(HADOOP_CLASSPATH) -d ‘/home/huser/Desktop/LogFileTut/exp_classfile .*java

7. Create Manifest.txt file.

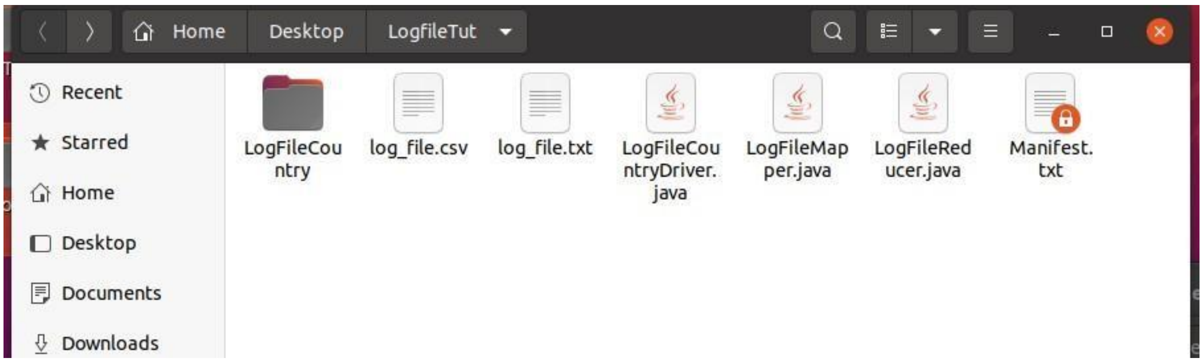


```
huser@ubuntu-college:~/Desktop/LogfileTut$ javac -d '/home/huser/Desktop/LogfileTut/exp_classfile' LogFileMa
pper.java LogFileReducer.java LogFileCountryDriver.java
huser@ubuntu-college:~/Desktop/LogfileTut$ sudo gedit Manifest.txt
[sudo] password for huser:

(gedit:59507): Tepl-WARNING **: 22:24:16.402: GVfs metadata is not supported. Fallback to TeplMetadataManage
r. Either GVfs is not correctly installed or GVfs metadata are not supported on this platform. In the latter
 case, you should configure Tepl with --disable-gvfs-metadata.
```



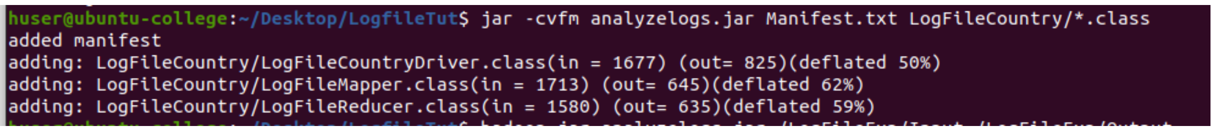
Manifest.txt [Read-Only]
~/Desktop/demo

```
Main-Class: LogFileCountry.LogFileCountryDriver

```

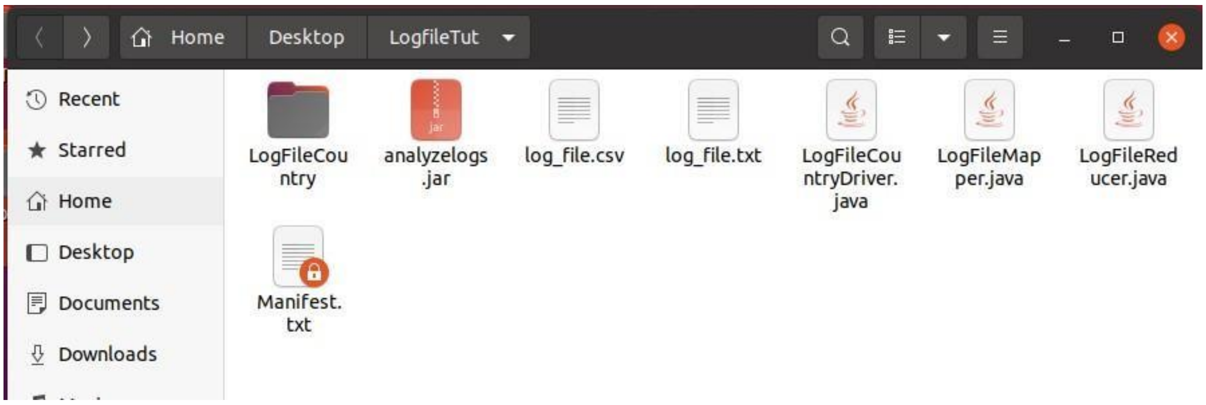


8. Creation .jar file of classes:

$ jar -cvfm analyzelogs.jar Manifest.txt LogFileCountry/*.class



```
huser@ubuntu-college:~/Desktop/LogfileTut$ jar -cvfm analyzelogs.jar Manifest.txt LogFileCountry/*.class
added manifest
adding: LogFileCountry/LogFileCountryDriver.class(in = 1677) (out= 825)(deflated 50%)
adding: LogFileCountry/LogFileMapper.class(in = 1713) (out= 645)(deflated 62%)
adding: LogFileCountry/LogFileReducer.class(in = 1580) (out= 635)(deflated 59%)
```



9. Create a directory on HDFS .And check on localhost:9870

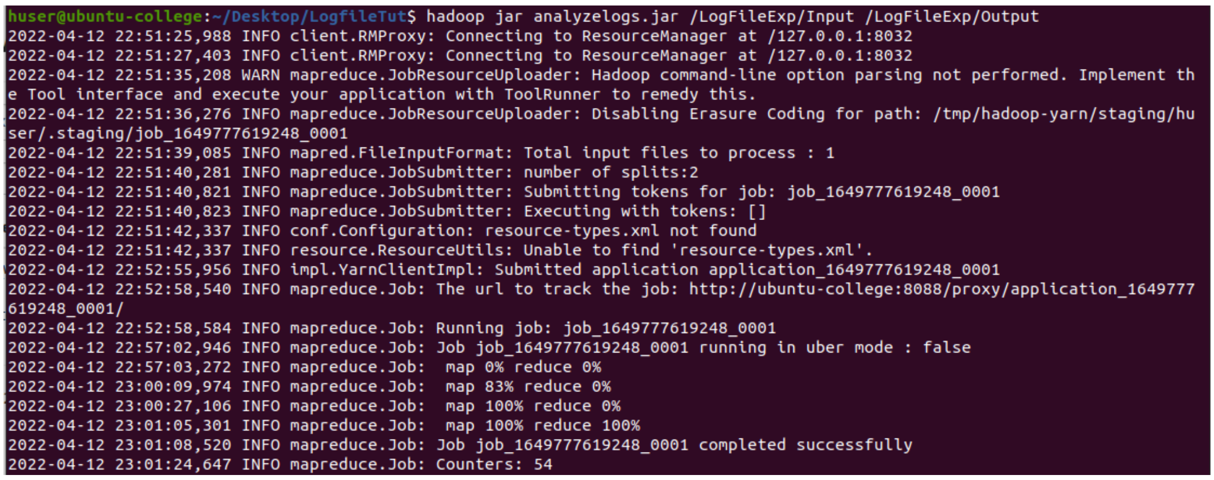
$ hdfs dfs -mkdir / LogFileExp
$ hdfs dfs -mkdir / LogFileExp/Input
$ hdfs dfs -mkdir / LogFileExp/Output

10. Upload the log_file.csv in hadoop dir /LogFileExp/Input

$ hdfs dfs -put '/home/huser/Desktop/LogFileTut/log_file.csv' /LogFileExp/Input

11. Running the jar file on Hadoop.

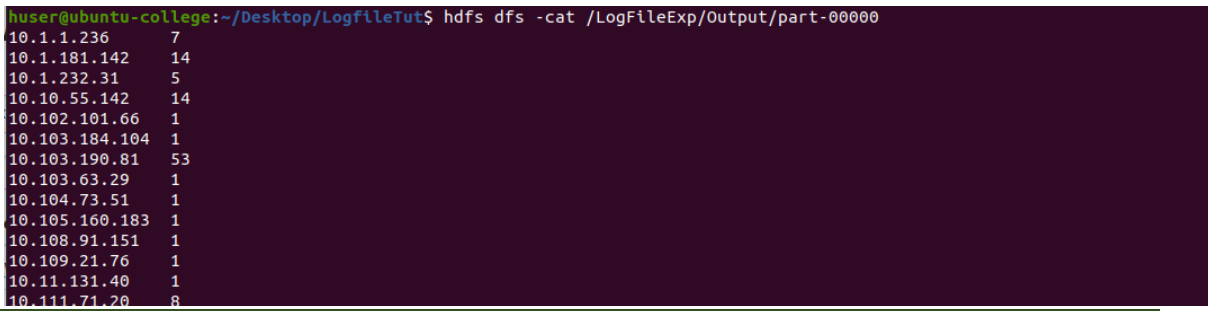
$ hadoop jar analyzelogs.jar /LogFileExp/Input /LogFileExp/Output



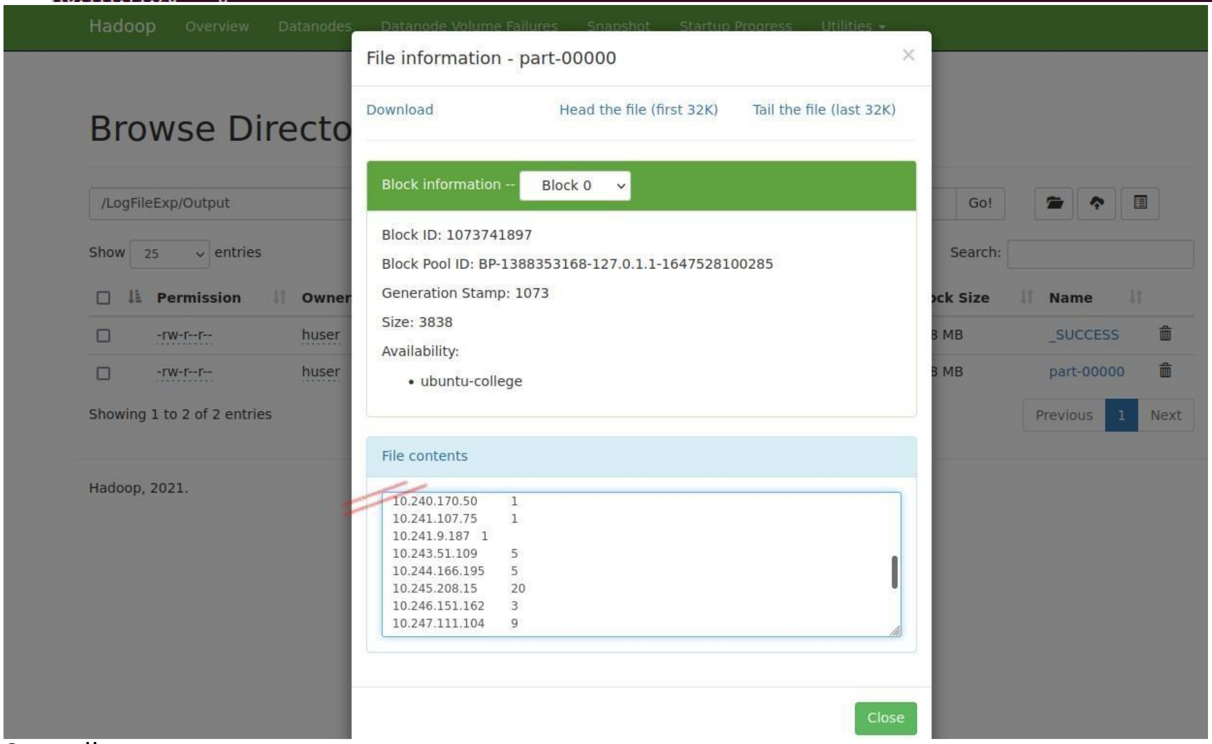
```
huser@ubuntu-college:~/Desktop/LogfileTut$ hadoop jar analyzelogs.jar /LogFileExp/Input /LogFileExp/Output
2022-04-12 22:51:25,988 INFO client.RMProxy: Connecting to ResourceManager at /127.0.0.1:8032
2022-04-12 22:51:27,403 INFO client.RMProxy: Connecting to ResourceManager at /127.0.0.1:8032
2022-04-12 22:51:35,208 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement th
e Tool interface and execute your application with ToolRunner to remedy this.
2022-04-12 22:51:36,276 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hu
ser/.staging/job_1649777619248_0001
2022-04-12 22:51:39,085 INFO mapred.FileInputFormat: Total input files to process : 1
2022-04-12 22:51:40,281 INFO mapreduce.JobSubmitter: number of splits:2
2022-04-12 22:51:40,821 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1649777619248_0001
2022-04-12 22:51:40,823 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-04-12 22:51:42,337 INFO conf.Configuration: resource-types.xml not found
2022-04-12 22:51:42,337 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2022-04-12 22:52:55,956 INFO impl.YarnClientImpl: Submitted application application_1649777619248_0001
2022-04-12 22:52:58,540 INFO mapreduce.Job: The url to track the job: http://ubuntu-college:8088/proxy/application_1649777
619248_0001/
2022-04-12 22:52:58,584 INFO mapreduce.Job: Running job: job_1649777619248_0001
2022-04-12 22:57:02,946 INFO mapreduce.Job: Job job_1649777619248_0001 running in uber mode : false
2022-04-12 22:57:03,272 INFO mapreduce.Job:  map 0% reduce 0%
2022-04-12 23:00:09,974 INFO mapreduce.Job:  map 83% reduce 0%
2022-04-12 23:00:27,106 INFO mapreduce.Job:  map 100% reduce 0%
2022-04-12 23:01:05,301 INFO mapreduce.Job:  map 100% reduce 100%
2022-04-12 23:01:08,520 INFO mapreduce.Job: Job job_1649777619248_0001 completed successfully
2022-04-12 23:01:24,647 INFO mapreduce.Job: Counters: 54
```

12. Check the Output file.

$ hdfs dfs -cat /LogFileExp/Output/part-00000



```
huser@ubuntu-college:~/Desktop/LogfileTut$ hdfs dfs -cat /LogFileExp/Output/part-00000
10.1.1.236      7
10.1.181.142    14
10.1.232.31     5
10.10.55.142    14
10.102.101.66   1
10.103.184.104  1
10.103.190.81   53
10.103.63.29    1
10.104.73.51    1
10.105.160.183  1
10.108.91.151   1
10.109.21.76    1
10.11.131.40    1
10.111.71.20    8
```



File information - part-00000

Download   Head the file (first 32K)   Tail the file (last 32K)

Block information -- Block 0

Block ID: 1073741897

Block Pool ID: BP-1388353168-127.0.1.1-1647528100285

Generation Stamp: 1073

Size: 3838

Availability:

- ubuntu-college

File contents

```
10.240.170.50   1
10.241.107.75   1
10.241.9.187    1
10.243.51.109   5
10.244.166.195  5
10.245.208.15   20
10.246.151.162  3
10.247.111.104  9
```

Close

/LogFileExp/Output   Go!

Show 25 entries   Search:

Permission	Owner	ck Size	Name
-rw-r--r--	huser	8 MB	_SUCCESS
-rw-r--r--	huser	8 MB	part-00000

Showing 1 to 2 of 2 entries   Previous 1 Next

Hadoop, 2021.

13. Stop all processes :

$ stop-all.sh

Conclusion: Thus, we successfully implement, distributed application using MapReduce which processes a log file of a system.

GROUP B
EXPERIMENT: 3

Weather.java

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
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.mapred.KeyValueTextInputFormat;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.util.*;
/**
 * This is an Hadoop Map/Reduce application for Working on weather data It reads
 * the text input files, breaks each line into stations weather data and finds
 * average for temperature, dew point , wind speed. The output is a locally
 * sorted list of stations and its 12 attribute vector of average temp , dew ,
 * wind speed of 4 sections for each month.
*/
public class Weather extends Configured implements Tool {
        final long DEFAULT_SPLIT_SIZE = 128 * 1024 * 1024;
        /**
         * Map Class for Job 1
         * For each line of input, emits key value pair with
         * station_yearmonth_sectionno as key and 3 attribute vector with
         * temperature , dew point , wind speed as value.Map method will strip the
         * day and hour from field and replace it with section no (
         * <b>station_yearmonth_sectionno</b>, <b><temperature,dew point , wind
 * speed></b>).
 */
public static class MapClass extends MapReduceBase
                    implements Mapper<LongWritable, Text, Text, Text> {
        private Text word = new Text();
        private Text values = new Text();
        public void map(LongWritable key, Text value,
                                        OutputCollector<Text, Text> output,
                                        Reporter reporter) throws IOException {
                String line = value.toString();
                StringTokenizer itr = new StringTokenizer(line);
                int counter = 0;
                String key_out = null;
                String value_str = null;
                boolean skip = false;
                loop:while (itr.hasMoreTokens() && counter<13) {
                        String str = itr.nextToken();
                        switch (counter) {
                        case 0:
                                key_out = str;
                                if(str.contains("STN")){//Ignoring rows where stationid is all 9
                                        skip = true;
                                        break loop;
                                }else{
                                        break;
                                }
                        case 2:
                                int hour = Integer.valueOf(str.substring(str.lastIndexOf("_")+1, str.length()));
                                str = str.substring(4,str.lastIndexOf("_")-2);
                                if(hour>4 && hour<=10){
                                        str = str.concat("_section1");
                                }else if(hour>10 && hour<=16){
                                        str = str.concat("_section2");
                                }else if(hour>16 && hour<=22){
                                        str = str.concat("_section3");
                                } else{ str = str.concat("_section4");
                                }

                                key_out = key_out.concat("_").concat(str);
                                break;
                        case 3://Temperature
                                if(str.equals("9999.9")){//Ignoring rows temperature is all 9
                                        skip = true;
                                        break loop;
                                }else{
                                        value_str = str.concat(" ");
                                        break;
                                }
                        case 4://Dew point
                                if(str.equals("9999.9")){//Ignoring rows where dewpoint all 9
                                        skip = true;
                                        break loop;
                    }else{
                        value_str = value_str.concat(str).concat(" ");
                        break;
                    }
                case 12://Wind speed
                    if(str.equals("999.9")){//Ignoring rows wind speed is all 9
                        skip = true;
                        break loop;
                    }else{ value_str = value_str.concat(str).concat(" ");
                        break;
                    }
                default: break;
                }
                counter++;
            }
            if(!skip){
                word.set(key_out);
                values.set(value_str);
                output.collect(word, values);
            }
        }
}
/**
 * Reducer Class for Job 1
 * A reducer class that just emits 3 attribute vector with average
 * temperature , dew point , wind speed for each of the section of the month for each input
 */
public static class Reduce extends MapReduceBase
            implements Reducer<Text, Text, Text, Text> {
        private Text value_out_text = new Text();
        public void reduce(Text key, Iterator<Text> values,
                OutputCollector<Text, Text> output, Reporter reporter) throws IOException {
            double sum_temp = 0;
            double sum_dew = 0;
            double sum_wind = 0;
            int count = 0;

            while (values.hasNext()) {
                String str = values.next().toString();
                StringTokenizer itr = new StringTokenizer(str);
                int count_vector = 0;
                while (itr.hasMoreTokens()) {
                    String nextToken = itr.nextToken(" ");
                    if(count_vector==0){
                        sum_temp += Double.valueOf(nextToken);
                    }
                    if(count_vector==1){
                        sum_dew += Double.valueOf(nextToken);
                    }
                    if(count_vector==2){
                        sum_wind += Double.valueOf(nextToken);
                    }
```

```
                        count_vector++;
                    }
                    count++;
                }
                double avg_tmp = sum_temp / count;
                double avg_dew = sum_dew / count;
                double avg_wind = sum_wind / count;
                System.out.println(key.toString()+" count is "+count+" sum of temp is "+sum_temp+" sum of
dew is "+sum_dew+" sum of wind is "+sum_wind+"\n");
                String                    value_out                    =                    String.valueOf(avg_tmp).concat("
").concat(String.valueOf(avg_dew)).concat(" ").concat(String.valueOf(avg_wind));
                value_out_text.set(value_out);
                output.collect(key, value_out_text);
            }
    }
    static int printUsage() {
            System.out.println("weather [-m  <maps>]  [-r  <reduces>]  <job_1  input>  <job_1  output>  <job_2
output>");
            ToolRunner.printGenericCommandUsage(System.out);
            return -1;
    }
    /**
     * The main driver for weather map/reduce program.
     * Invoke this method to submit the map/reduce job.
     * @throws IOException When there is communication problems with the job tracker.
     */
    public int run(String[] args) throws Exception {
            Configuration config = getConf();
            // We need to lower input block size by factor of two
            JobConf conf = new JobConf(config, Weather.class);
            conf.setJobName("Weather Job1");

            // the keys are words (strings)
            conf.setOutputKeyClass(Text.class);
            // the values are counts (ints)
            conf.setOutputValueClass(Text.class);

            conf.setMapOutputKeyClass(Text.class);
            conf.setMapOutputValueClass(Text.class);
            conf.setMapperClass(MapClass.class);
            //conf.setCombinerClass(Combiner.class);
            conf.setReducerClass(Reduce.class);
            List<String> other_args = new ArrayList<String>();
            for(int i=0; i < args.length; ++i) {
                try {
                    if ("-m".equals(args[i])) {
                        conf.setNumMapTasks(Integer.parseInt(args[++i]));
                    } else if ("-r".equals(args[i])) {
                        conf.setNumReduceTasks(Integer.parseInt(args[++i]));
                    } else {
                        other_args.add(args[i]);
                    }
                } catch (NumberFormatException except) {
                        System.out.println("ERROR: Integer expected instead of " + args[i]);
                        return printUsage();
                } catch (ArrayIndexOutOfBoundsException except) {
                        System.out.println("ERROR: Required parameter missing from " +
                                args[i-1]);
                        return printUsage();
                }
        }
        // Make sure there are exactly 2 parameters left.
        FileInputFormat.setInputPaths(conf, other_args.get(0));
        FileOutputFormat.setOutputPath(conf, new Path(other_args.get(1)));
        JobClient.runJob(conf);
        return 0;
    }
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new Weather(), args);
        System.exit(res);
    }
}
```

Input: <u>sample_weather.txt</u> (sample)

```
690190 13910  20060201_0  51.75    33.0 24  1006.3 24   943.9 24   15.0 24   10.7 24   22.0   28.9    0.00I 999.9  000000
690190 13910  20060201_1  54.74    33.0 24  1006.3 24   943.9 24   15.0 24   10.7 24   22.0   28.9    0.00I 999.9  00000
690190 13910  20060201_2  50.59    33.0 24  1006.3 24   943.9 24   15.0 24   10.7 24   22.0   28.9    0.00I 999.9  00000
690190 13910  20060201_3  51.67    33.0 24  1006.3 24   943.9 24   15.0 24   10.7 24   22.0   28.9    0.00I 999.9  000000
690190 13910  20060201_4  65.67    33.0 24  1006.3 24   943.9 24   15.0 24   10.7 24   22.0   28.9    0.00I 999.9  000000
690190 13910  20060201_5  55.37    33.0 24  1006.3 24   943.9 24   15.0 24   10.7 24   22.0   28.9    0.00I 999.9  000000
690190 13910  20060201_6  49.26    33.0 24  1006.3 24   943.9 24   15.0 24   10.7 24   22.0   28.9    0.00I 999.9  000000
690190 13910  20060201_7  55.44    33.0 24  1006.3 24   943.9 24   15.0 24   10.7 24   22.0   28.9    0.00I 999.9  000000
690190 13910  20060201_8  64.05    33.0 24  1006.3 24   943.9 24   15.0 24   10.7 24   22.0   28.9    0.00I 999.9  000000
```

Output: part-00000.txt (on Hadoop)

```
690190_02_section1  53.87166666666666 25.899999999999995 7.774999999999998
690190_02_section2  54.76125000000001 25.900000000000006 7.774999999999999
690190_02_section3  53.25041666666667 25.899999999999995 7.774999999999996
690190_02_section4  52.44708333333333 25.900000000000006 7.774999999999999
```

Weather Data Analysis Steps to run:

14. Starting Hadoop

    $ start-all.sh

15. Made A folder “WeatherAssi” and write Weather.java code.
28. Create new folder for input data.
29. Add input text file in the input data folder.
30. Create new folder to hold java class files.
31. Set HADOOP_CLASSPATH environment variable.

    $ export HADOOP_CLASSPATH=$(hadoop classpath)

32. Create a directory on HDFS

    $ hdfs dfs -mkdir /WeatherTut

    $ hdfs dfs -mkdir /WeatherTut/Input

33. Checking on localhost:9870
34. Upload the input file (device) to that directory.

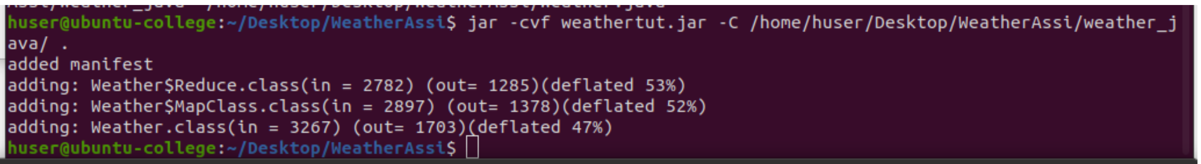
    $ hdfs dfs -put input_data/sample_wheater.txt /WeatherTuT/Input

35. Compile the java code:

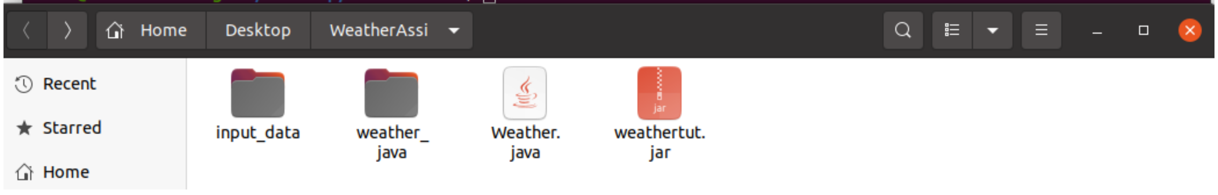
    $ javac -classpath $(HADOOP_CLASSPATH) -d '/home/huser/Desktop/WeatherAssi/ weather_java' /home/huser/Desktop/WeatherAssi/Weather.java

36. Creation .jar file of classes:

    $ jar -cvf weathertut.jar -C /home/huser/Desktop/WeatherAssi/weather_java/ .

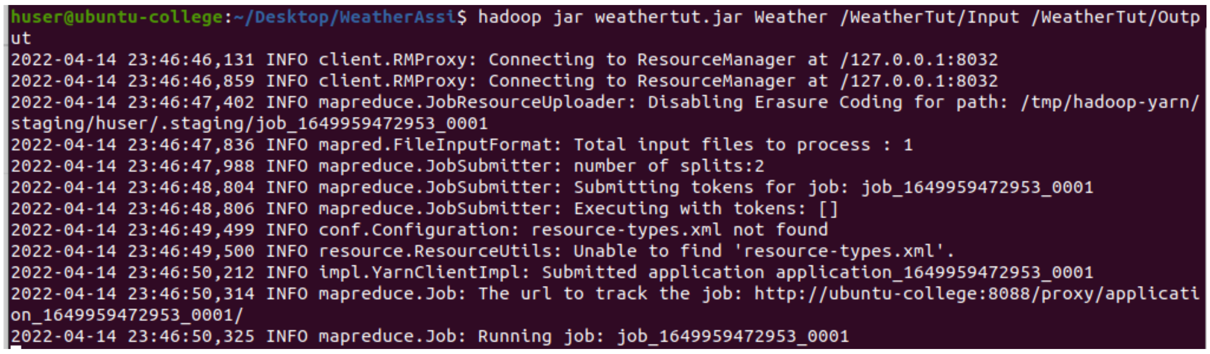


```
huser@ubuntu-college:~/Desktop/WeatherAssi$ jar -cvf weathertut.jar -C /home/huser/Desktop/WeatherAssi/weather_j
ava/ .
added manifest
adding: Weather$Reduce.class(in = 2782) (out= 1285)(deflated 53%)
adding: Weather$MapClass.class(in = 2897) (out= 1378)(deflated 52%)
adding: Weather.class(in = 3267) (out= 1703)(deflated 47%)
huser@ubuntu-college:~/Desktop/WeatherAssi$
```



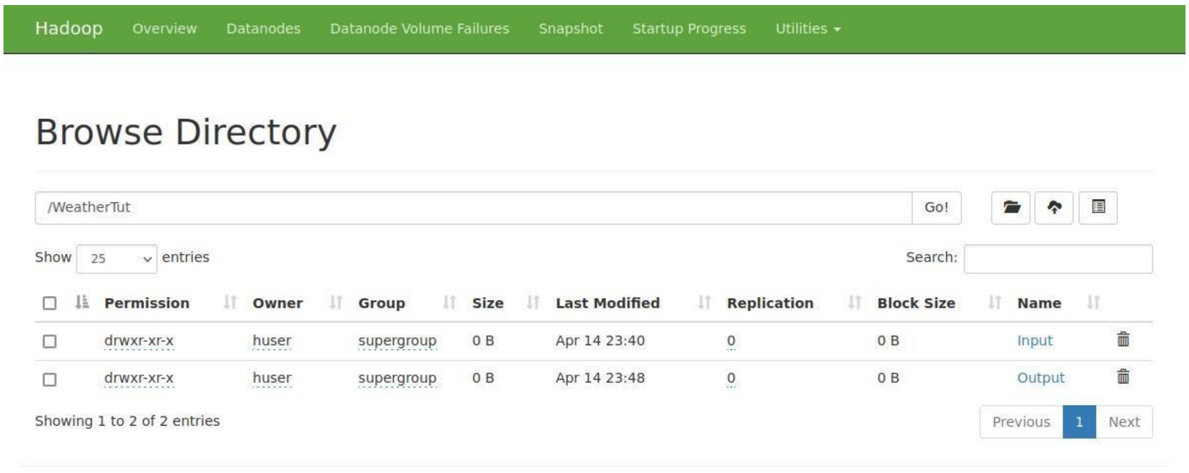
37. Running the jar file on Hadoop

    $ hadoop jar weather.jar Weather /WeatherTut/Input /WeatherTut/Ouput



```
huser@ubuntu-college:~/Desktop/WeatherAssi$ hadoop jar weathertut.jar Weather /WeatherTut/Input /WeatherTut/Outp
ut
2022-04-14 23:46:46,131 INFO client.RMProxy: Connecting to ResourceManager at /127.0.0.1:8032
2022-04-14 23:46:46,859 INFO client.RMProxy: Connecting to ResourceManager at /127.0.0.1:8032
2022-04-14 23:46:47,402 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/
staging/huser/.staging/job_1649959472953_0001
2022-04-14 23:46:47,836 INFO mapred.FileInputFormat: Total input files to process : 1
2022-04-14 23:46:47,988 INFO mapreduce.JobSubmitter: number of splits:2
2022-04-14 23:46:48,804 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1649959472953_0001
2022-04-14 23:46:48,806 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-04-14 23:46:49,499 INFO conf.Configuration: resource-types.xml not found
2022-04-14 23:46:49,500 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2022-04-14 23:46:50,212 INFO impl.YarnClientImpl: Submitted application application_1649959472953_0001
2022-04-14 23:46:50,314 INFO mapreduce.Job: The url to track the job: http://ubuntu-college:8088/proxy/applicati
on_1649959472953_0001/
2022-04-14 23:46:50,325 INFO mapreduce.Job: Running job: job_1649959472953_0001
```

38. Check output on localhost:9870 /localhost:50070



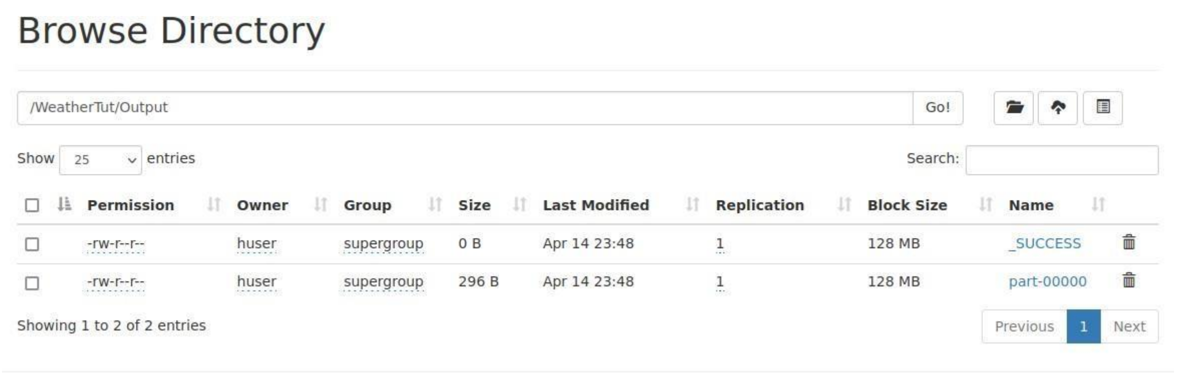
Browse Directory

/WeatherTut

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	huser	supergroup	0 B	Apr 14 23:40	0	0 B	Input
drwxr-xr-x	huser	supergroup	0 B	Apr 14 23:48	0	0 B	Output

Showing 1 to 2 of 2 entries



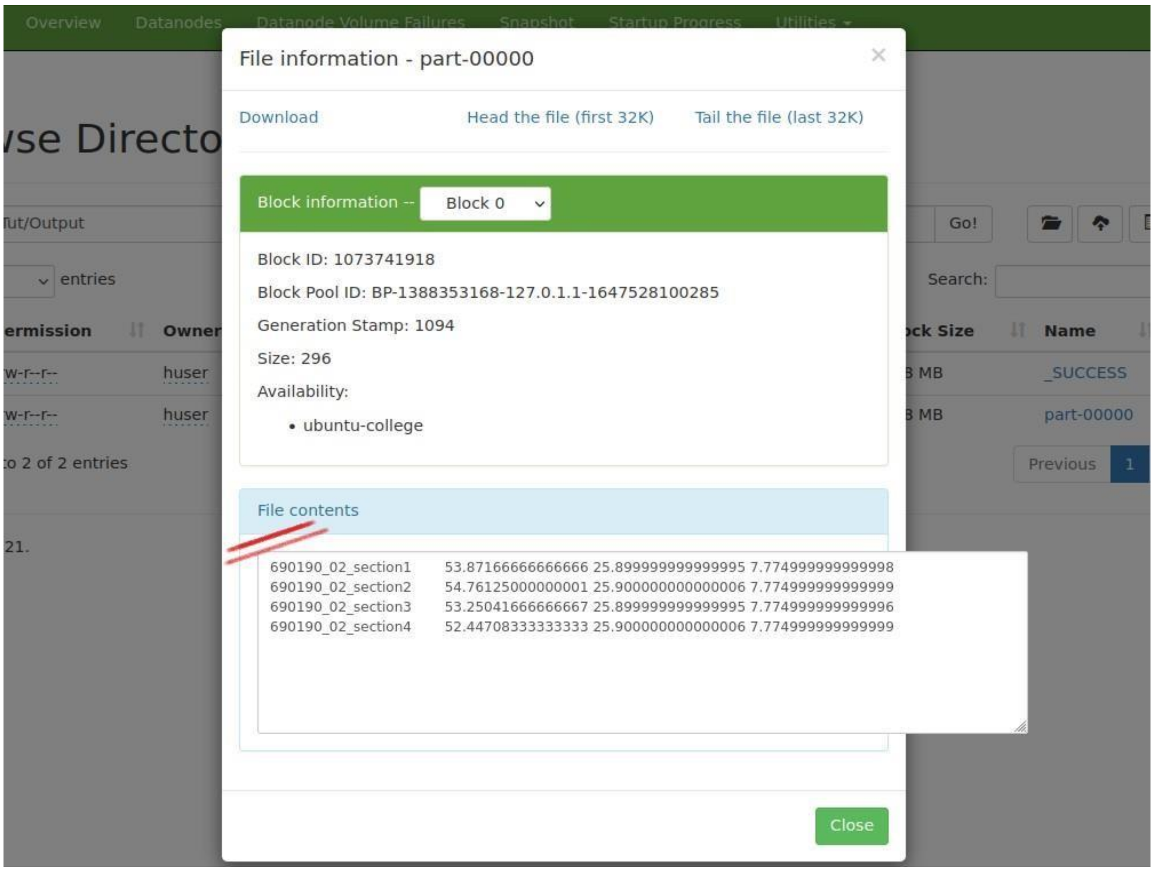
Browse Directory

/WeatherTut/Output

Show 25 entries

	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
☐	-rw-r--r--	huser	supergroup	0 B	Apr 14 23:48	1	128 MB	_SUCCESS
☐	-rw-r--r--	huser	supergroup	296 B	Apr 14 23:48	1	128 MB	part-00000

Showing 1 to 2 of 2 entries



File information - part-00000

Download Head the file (first 32K) Tail the file (last 32K)

Block information -- Block 0

Block ID: 1073741918

Block Pool ID: BP-1388353168-127.0.1.1-1647528100285

Generation Stamp: 1094

Size: 296

Availability:

- ubuntu-college

File contents

```
690190_02_section1	53.87166666666666 25.899999999999995 7.774999999999998
690190_02_section2	54.76125000000001 25.900000000000006 7.774999999999999
690190_02_section3	53.25041666666667 25.899999999999995 7.774999999999996
690190_02_section4	52.44708333333333 25.900000000000006 7.774999999999999
```

39. Stop Hadoop services:

    $ stop-all.sh

Conclusion: Thus, we successfully, Locate dataset (e.g., sample_weather.txt) for working on weather data which reads the text input files and finds average for temperature, dew point and wind speed.