**ENGR-E 534 BIG DATA APPLICATIONS**
**Assignment 4**
**Shalini Kothuru (University ID 2001096463)**

**Task 1:** Create a directory on HDFS

Started services and created 'assignment_data' directory

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.22621.2283]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>cd..

C:\Windows>cd..

C:\>cd hadoop-3.2.4

C:\hadoop-3.2.4>cd sbin

C:\hadoop-3.2.4\sbin>start-dfs.cmd

C:\hadoop-3.2.4\sbin>start-yarn.cmd
starting yarn daemons

C:\hadoop-3.2.4\sbin>hdfs dfs -mkdir /assignment_data

C:\hadoop-3.2.4\sbin>
```

Created one more directory named bda_assignment_4

```
C:\hadoop-3.2.4\sbin>hdfs dfs -mkdir /bda_assignment_4

C:\hadoop-3.2.4\sbin>hdfs dfs -ls /
Found 2 items
drwxr-xr-x   - HP supergroup          0 2023-10-08 15:43 /assignment_data
drwxr-xr-x   - HP supergroup          0 2023-10-08 15:50 /bda_assignment_4
```

**Task 2:** Upload files to HDFS

Uploading file.txt, file.csv, file3.json from local machine to directory assignment_data

```
C:\hadoop-3.2.4\sbin>hdfs dfs -mkdir /bda_assignment_4

C:\hadoop-3.2.4\sbin>hdfs dfs -ls /
Found 2 items
drwxr-xr-x   - HP supergroup          0 2023-10-08 15:43 /assignment_data
drwxr-xr-x   - HP supergroup          0 2023-10-08 15:50 /bda_assignment_4

C:\hadoop-3.2.4\sbin>hdfs dfs -copyFromLocal "D:\MSCS Course\1.3\BDA\Assignments\Assignment 4\files\file1.txt" /assignment_data/

C:\hadoop-3.2.4\sbin>hdfs dfs -copyFromLocal "D:\MSCS Course\1.3\BDA\Assignments\Assignment 4\files\file2.csv" /assignment_data/

C:\hadoop-3.2.4\sbin>
C:\hadoop-3.2.4\sbin>hdfs dfs -copyFromLocal "D:\MSCS Course\1.3\BDA\Assignments\Assignment 4\files\file3.json" /assignment_data/
```

**Task 3:** List files in HDFS

Listing files in assignment_data directory

```
C:\hadoop-3.2.4\sbin>hdfs dfs -ls /assignment_data
Found 3 items
-rw-r--r--   1 HP supergroup         21 2023-10-08 15:56 /assignment_data/file1.txt
-rw-r--r--   1 HP supergroup         32 2023-10-08 15:56 /assignment_data/file2.csv
-rw-r--r--   1 HP supergroup         46 2023-10-08 15:56 /assignment_data/file3.json

C:\hadoop-3.2.4\sbin>
```

**Task 4:** View file content in HDFS

Viewing contents of file1.txt using cat

```
C:\hadoop-3.2.4\sbin>hdfs dfs -cat /assignment_data/file1.txt
Big data applications
C:\hadoop-3.2.4\sbin>
```

**Task 5:** Create a new directory in HDFS

Creating a sub directory 'docs' in assignment_data directory and checking if it is craeted

```
C:\hadoop-3.2.4\sbin>hdfs dfs -mkdir /assignment_data/docs

C:\hadoop-3.2.4\sbin>hdfs dfs -ls /assignment_data
Found 4 items
drwxr-xr-x   - HP supergroup          0 2023-10-08 15:59 /assignment_data/docs
-rw-r--r--   1 HP supergroup         21 2023-10-08 15:56 /assignment_data/file1.txt
-rw-r--r--   1 HP supergroup         32 2023-10-08 15:56 /assignment_data/file2.csv
-rw-r--r--   1 HP supergroup         46 2023-10-08 15:56 /assignment_data/file3.json

C:\hadoop-3.2.4\sbin>
```

**Task 6:** Move files to a different directory in HDFS

Moving file2.csv and file3.json to a docs directory and checking if those files are moved.

```
C:\hadoop-3.2.4\sbin>hdfs dfs -mv /assignment_data/file2.csv /assignment_data/docs/

C:\hadoop-3.2.4\sbin>hdfs dfs -mv /assignment_data/file3.json /assignment_data/docs/

C:\hadoop-3.2.4\sbin>hdfs dfs -ls /assignment_data
Found 2 items
drwxr-xr-x   - HP supergroup          0 2023-10-08 16:01 /assignment_data/docs
-rw-r--r--   1 HP supergroup         21 2023-10-08 15:56 /assignment_data/file1.txt

C:\hadoop-3.2.4\sbin>hdfs dfs -ls /assignment_data/docs
Found 2 items
-rw-r--r--   1 HP supergroup         32 2023-10-08 15:56 /assignment_data/docs/file2.csv
-rw-r--r--   1 HP supergroup         46 2023-10-08 15:56 /assignment_data/docs/file3.json
```

**Task 7:** Delete files from HDFS

Deleting file1.txt from assignment_data directory

```
C:\hadoop-3.2.4\sbin>hdfs dfs -rm /assignment_data/file1.txt
Deleted /assignment_data/file1.txt

C:\hadoop-3.2.4\sbin>hdfs dfs -ls /assignment_data
Found 1 items
drwxr-xr-x   - HP supergroup          0 2023-10-08 16:01 /assignment_data/docs

C:\hadoop-3.2.4\sbin>
```

**Task 8:** Check HDFS file status

HDFS status of file2.csv

```
Administrator: Command Prompt

C:\hadoop-3.2.4>hdfs fsck /assignment_data/docs/file2.csv -files -blocks -locations
Connecting to namenode via http://localhost:9870/fsck?ugi=HP&files=1&blocks=1&locations=1&path=%2Fassignment_data%2Fdocs%2Ffile2.csv
FSCK started by HP (auth:SIMPLE) from /127.0.0.1 for path /assignment_data/docs/file2.csv at Sun Oct 08 16:26:01 EDT 2023

/assignment_data/docs/file2.csv 32 bytes, replicated: replication=1, 1 block(s):  OK
0. BP-1536519246-192.168.56.1-1696790261588:blk_1073741826_1002 len=32 Live_repl=1  [DatanodeInfoWithStorage[127.0.0.1:9866,DS-3716e37f-7b95-4c26-8b1e-3afa29acd61b,DISK]]


Status: HEALTHY
 Number of data-nodes:  1
 Number of racks:               1
 Total dirs:                    0
 Total symlinks:                0

Replicated Blocks:
 Total size:    32 B
 Total files:   1
 Total blocks (validated):      1 (avg. block size 32 B)
 Minimally replicated blocks:   1 (100.0 %)
 Over-replicated blocks:        0 (0.0 %)
 Under-replicated blocks:       0 (0.0 %)
 Mis-replicated blocks:         0 (0.0 %)
 Default replication factor:    1
 Average block replication:     1.0
 Missing blocks:                0
 Corrupt blocks:                0
 Missing replicas:              0 (0.0 %)

Erasure Coded Block Groups:
 Total size:    0 B
 Total files:   0
 Total block groups (validated):        0
 Minimally erasure-coded block groups:  0
 Over-erasure-coded block groups:       0
 Under-erasure-coded block groups:      0
 Unsatisfactory placement block groups: 0
 Average block group size:      0.0
 Missing block groups:          0
 Corrupt block groups:          0
14°C
Partly sunny
Search
ENG
IN
16:26
08-10-2023
```

```
Replicated Blocks:
 Total size:    32 B
 Total files:   1
 Total blocks (validated):      1 (avg. block size 32 B)
 Minimally replicated blocks:   1 (100.0 %)
 Over-replicated blocks:        0 (0.0 %)
 Under-replicated blocks:       0 (0.0 %)
 Mis-replicated blocks:         0 (0.0 %)
 Default replication factor:    1
 Average block replication:     1.0
 Missing blocks:                0
 Corrupt blocks:                0
 Missing replicas:              0 (0.0 %)

Erasure Coded Block Groups:
 Total size:    0 B
 Total files:   0
 Total block groups (validated):        0
 Minimally erasure-coded block groups:  0
 Over-erasure-coded block groups:       0
 Under-erasure-coded block groups:      0
 Unsatisfactory placement block groups: 0
 Average block group size:      0.0
 Missing block groups:          0
 Corrupt block groups:          0
 Missing internal blocks:       0
FSCK ended at Sun Oct 08 16:26:01 EDT 2023 in 10 milliseconds


The filesystem under path '/assignment_data/docs/file2.csv' is HEALTHY
```

**Task 9:** Delete a directory from HDFS

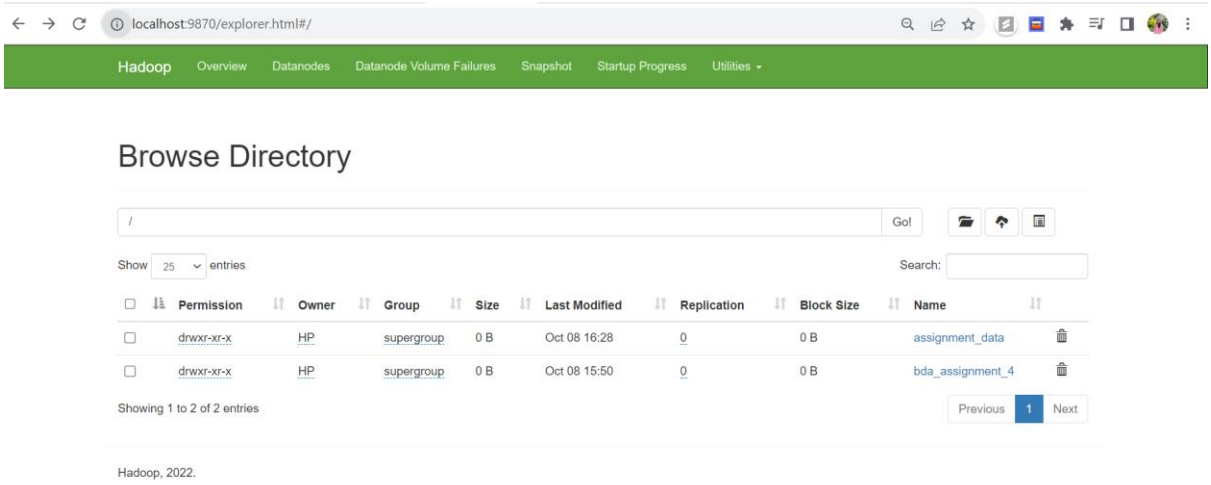
Deleting directory from hdfs and checking if it is deleted.

```
C:\hadoop-3.2.4>
C:\hadoop-3.2.4>hdfs dfs -rm -r /assignment_data/docs
Deleted /assignment_data/docs

C:\hadoop-3.2.4>hdfs dfs -ls /assignment_data

C:\hadoop-3.2.4>
```

Now, we don't have docs directory. Also we can use below interface vis localhost:9870 to view status.




**Task 10: Dataset Overview**

Provide a brief description of the dataset, including the file format, size, and the type of data it contains.

**Data source:**
I choose All beauty product category from link below
https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/#complete-data

**Data description:**
It is the reviews provided by different customers for different beauty products on amazon platform. I am using 2018 version of this dataset. It has 371345 records i.e reviews. The reviews are stored in json file and the file size is 167 MB.

It has following attributes:

- overall : overall rating of product. Range is 1 to 5
- vote: number of votes
- verified: verified or not(values-True/False)
- reviewTime: time of the review
- reviewerID: reviewer ID
- asin: product ID
- reviewerName : reviewer name
- reviewText: review provided by the reviewer.
- summary: summary of review
- unixReviewTime: time of the review in unix time

**Task 11: Word Count**

I used Intelli J IDEA IDE to achieve task 11 and task 12 as we can add dependencies directly using Maven.

## Implementation

## Mapper Class

```java
package bda_4;
//importing necessary libraries
import java.io.IOException;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
import org.codehaus.jackson.JsonNode;
import org.codehaus.jackson.map.ObjectMapper;
1 usage
public class mapper extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
    1 usage
    private final static IntWritable one = new IntWritable( value: 1);
    2 usages
    private Text token = new Text();
    1 usage
    private ObjectMapper map_object = new ObjectMapper();
    public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output,
                    Reporter reporter) throws IOException {
        String line = value.toString();
        try
        {
            //parsing the json data
```

```java
                    Reporter reporter) throws IOException {
        String line = value.toString();
        try
        {
            //parsing the json data
            JsonNode json_node = map_object.readTree(line);
            //extracting the "reviewText" attribute from JSON
            String reviewText = json_node.get("reviewText").asText();
            //removing punctuation and converting words to lowercase
            reviewText = reviewText.replaceAll( regex: "[^a-zA-Z\\s]", replacement: "").toLowerCase();
            //splitting the processed sentences into words
            String[] words = reviewText.split( regex: "\\s+");
            //emitting each word with 1 as its count
            for (String word : words)
            {
                this.token.set(word);
                output.collect(this.token, one);
            }
        }
        //exception handling
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

## Reducer Class

word_count Version control

pom.xml (word_count) mapper.java connect.java reducer.java

```java
package bda_4;
////importing necessary libraries
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;
1 usage
public class reducer  extends MapReduceBase implements Reducer<Text,IntWritable,Text,IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,IntWritable> output,
                       Reporter reporter) throws IOException {
        //initial count of word is set to zero
        int frequency=0;
        //iterating and combining the values of the word to get frequency of that word in total words
        while (values.hasNext())
        {
            frequency+=values.next().get();
        }
        //word wits its frequency
        output.collect(key,new IntWritable(frequency));
    }
}
```

## Connect Class

word_count Version control

pom.xml (word_count) mapper.java connect.java reducer.java

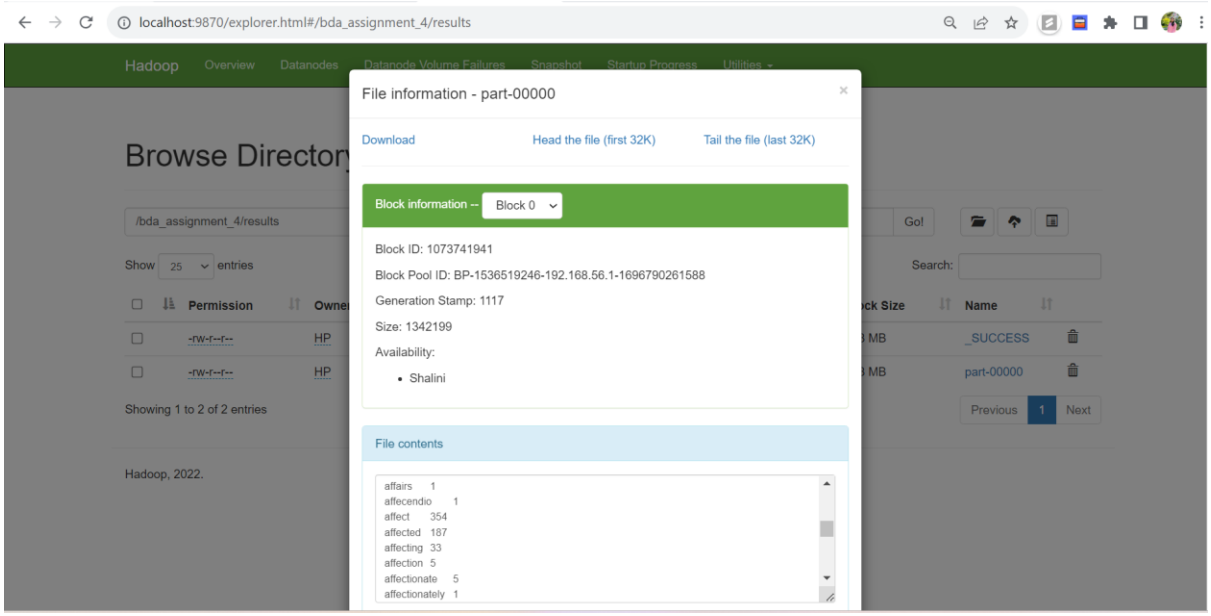
```java
package bda_4;
//importing necessary libraries
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.TextOutputFormat;
public class connect
{
    public static void main(String[] args) throws IOException
    {
        //configuring job
        JobConf job_conf = new JobConf(connect.class);
        job_conf.setJobName("word_count");
        job_conf.setOutputKeyClass(Text.class);
        job_conf.setOutputValueClass(IntWritable.class);
        job_conf.setMapperClass(mapper.class);
        job_conf.setReducerClass(reducer.class);
        job_conf.setInputFormat(TextInputFormat.class);
        job_conf.setOutputFormat(TextOutputFormat.class);
        //taking argument 1 as input
        FileInputFormat.setInputPaths(job_conf,new Path(args[0]));
        //taking argument 2 as output
        FileOutputFormat.setOutputPath(job_conf,new Path(args[1]));
        //running job
        JobClient.runJob(job_conf);
    }
}
```

## Output:

```
PS D:\MSCS Course\1.3\BDA\Assignments\Assignment 4\word_count>

PS D:\MSCS Course\1.3\BDA\Assignments\Assignment 4\word_count> hadoop jar target/word_count-1.0-SNAPSHOT.jar bda_4.connect /bda_assignment_4/dataset/All_Beauty.json /bda
_assignment_4/results
2023-10-09 16:34:51,204 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2023-10-09 16:34:51,464 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2023-10-09 16:34:52,560 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application w
ith ToolRunner to remedy this.
2023-10-09 16:34:52,596 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/HP/.staging/job_1696794597114_0013
2023-10-09 16:34:53,003 INFO mapred.FileInputFormat: Total input files to process : 1
2023-10-09 16:34:53,038 INFO net.NetworkTopology: Adding a new node: /default-rack/127.0.0.1:9866
2023-10-09 16:34:53,143 INFO mapreduce.JobSubmitter: number of splits:2
2023-10-09 16:34:53,404 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1696794597114_0013
2023-10-09 16:34:53,407 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-10-09 16:34:53,618 INFO conf.Configuration: resource-types.xml not found
2023-10-09 16:34:53,620 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2023-10-09 16:34:54,101 INFO impl.YarnClientImpl: Submitted application application_1696794597114_0013
2023-10-09 16:34:54,163 INFO mapreduce.Job: The url to track the job: http://Shalini:8088/proxy/application_1696794597114_0013/
2023-10-09 16:34:54,165 INFO mapreduce.Job: Running job: job_1696794597114_0013
2023-10-09 16:35:05,488 INFO mapreduce.Job: Job job_1696794597114_0013 running in uber mode : false
2023-10-09 16:35:05,490 INFO mapreduce.Job:  map 0% reduce 0%
2023-10-09 16:35:27,917 INFO mapreduce.Job:  map 55% reduce 0%
2023-10-09 16:35:33,035 INFO mapreduce.Job:  map 62% reduce 0%
2023-10-09 16:35:34,054 INFO mapreduce.Job:  map 70% reduce 0%
2023-10-09 16:35:38,132 INFO mapreduce.Job:  map 83% reduce 0%
2023-10-09 16:35:39,151 INFO mapreduce.Job:  map 100% reduce 0%
```

## Seeing results in UI




**Seeing results through cmd:** I returned only first 50 due to huge size

```
Administrator: Command Prompt

C:\hadoop-3.2.4\sbin>hadoop fs -cat /bda_assignment_4/results/part-00000 | more +50
aathank 1
aawesome        1
aawkward        1
aazon   2
ab      39
aback   14
abad    1
abag    1
abaility        1
abajo   1
abandon 8
abandoned       18
abandoning      3
abanoned        1
abate   6
abated  4
abating 2
abberration     1
abbey   1
abbrasions      1
abbreviated     3
abbreviations   1
abc     16
abcderm 1
abck    1
abcs    1
abcsell 1
abd     21
abdomen 44
abdominal       33
abducted        1
abdul   1
abdulaziz       2
abe     3
abel    1
aber    2
abercombie      1
abercrombie     9
```

**Program workflow:**

Prior to executing program: I have added dependencies in xml file. I have uploaded All_Beauty.json file to hdfs using commands.

Coming to program's workflow

**Mapping phase**(mapper java class): It takes All_Beauty.json file as input and extracts reviewText attribute from all records. We then process sentences in reviewText by removing punctuation and convert it lower cases. We then split sentences based on spaces. After splitting we have tokens. Each token is printed with its count as 1.

**Shuffling phase**: Hadoop groups and sort the data obtained from mappers and send it to reducer class.

**Reducer phase**(reducer class file). This program takes sorted tokens as input and combines the sum of all the values which has same key and result in key value pairs where word is the key and corresponding frequency is the value.

**Connect class:** In this class, I defined input to the program and the output it should obtain. It also can be seen as center of all classes. It specifies mapper and reducer class, formats. Basically, it configures the job to be executed by Hadoop.

**Executing program:**
Once all the class files are coded, I generated the jar file for my program and executed the project in terminal using following command:

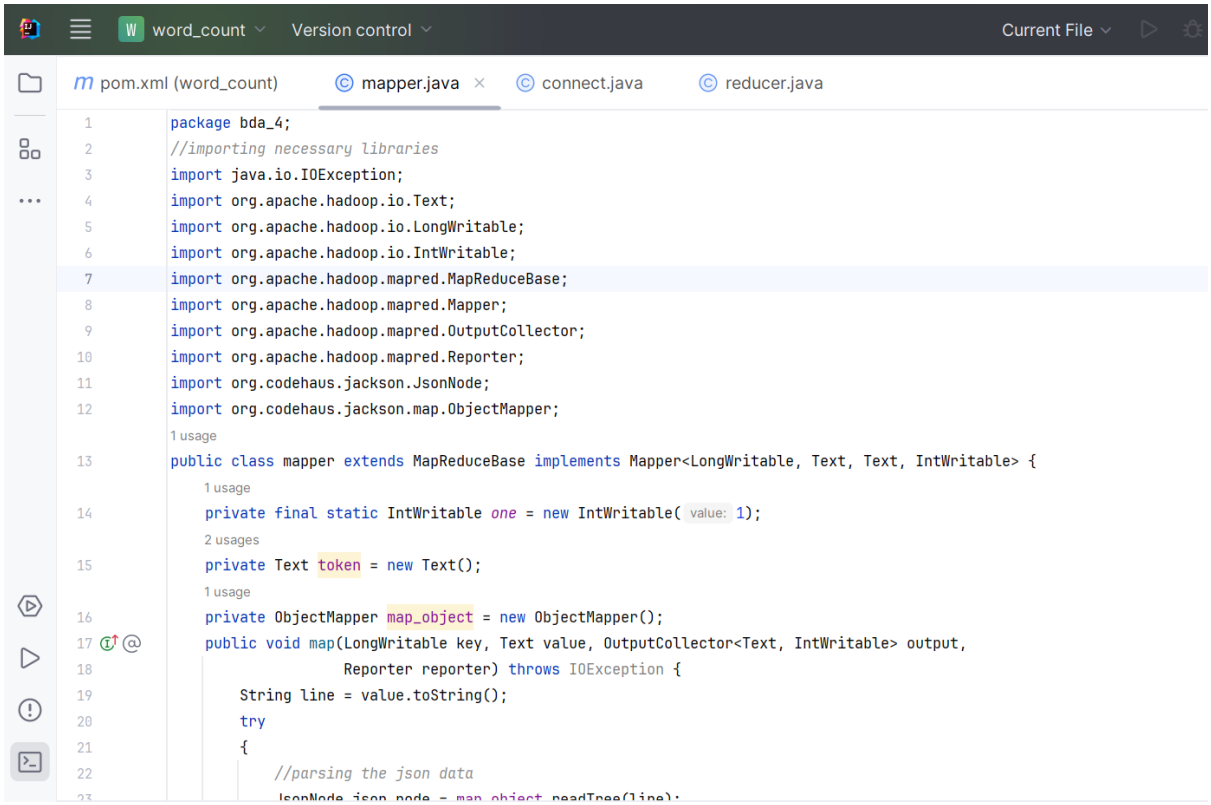
**Cmd to execute the word_count algorithm:**

hadoop jar target/word_count-1.0-SNAPSHOT.jar bda_4.connect /bda_assignment_4/dataset/All_Beauty.json /bda_assignment_4/results

## Task 12: Top N Words

### Implementation

**Mapper Class:** The mapper class is same as task 11 and not altered.




```java
package bda_4;
//importing necessary libraries
import java.io.IOException;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
import org.codehaus.jackson.JsonNode;
import org.codehaus.jackson.map.ObjectMapper;
1 usage
public class mapper extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
    1 usage
    private final static IntWritable one = new IntWritable( value: 1);
    2 usages
    private Text token = new Text();
    1 usage
    private ObjectMapper map_object = new ObjectMapper();
    public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output,
                    Reporter reporter) throws IOException {
        String line = value.toString();
        try
        {
            //parsing the json data
            JsonNode json_node = map_object.readTree(line);
```

word_count Version control Current File

pom.xml (word_count) mapper.java connect.java reducer.java

```java
                    Reporter reporter) throws IOException {
            String line = value.toString();
            try
            {
                //parsing the json data
                JsonNode json_node = map_object.readTree(line);
                //extracting the "reviewText" attribute from JSON
                String reviewText = json_node.get("reviewText").asText();
                //removing punctuation and converting words to lowercase
                reviewText = reviewText.replaceAll( regex: "[^a-zA-Z\\s]",  replacement: "").toLowerCase();
                //splitting the processed sentences into words
                String[] words = reviewText.split( regex: "\\s+");
                //emitting each word with 1 as its count
                for (String word : words)
                {
                    this.token.set(word);
                    output.collect(this.token, one);
                }
            }
            //exception handling
            catch (Exception e)
            {
                e.printStackTrace();
            }
        }
    }
```

## Reducer Class

word_count Version control Current File

pom.xml (word_count) mapper.java connect.java reducer.java

```java
package bda_4;
import java.io.IOException;
import java.util.Comparator;
import java.util.Iterator;
import java.util.Map;
import java.util.TreeMap;
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;
import org.apache.hadoop.mapred.JobConf;
2 usages
public class reducer extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {
    2 usages
    private int N;
    //TreeMap to store word and its frequency
    5 usages
    private TreeMap<Integer, Text> top_N_words;
    2 usages
    private OutputCollector<Text, IntWritable> result;
    public void configure(JobConf job)
    {
        //getting n value from configuration, I gave default value as 3
        N = job.getInt( name: "top_N",  defaultValue: 3);
        // new Treemap for the tokens
        top_N_words = new TreeMap<>(Comparator.reverseOrder());
    }

    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output,
```

```java
        //getting n value from configuration, I gave default value as 3
        N = job.getInt( name: "top_N",  defaultValue: 3);
        // new Treemap for the tokens
        top_N_words = new TreeMap<>(Comparator.reverseOrder());
    }

    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output,
                       Reporter reporter) throws IOException {
        int frequency = 0;
        while (values.hasNext()) {
            frequency += values.next().get();
        }
        //adding word and its frequency to the TreeMap
        top_N_words.put(frequency, new Text(key.toString()));
        //keeping only top N words
        if (top_N_words.size() > N)
        {
            top_N_words.pollLastEntry();
        }
        //storing the OutputCollector for later use in the close method as I can't use output directly in close method
        result = output;
    }
    public void close() throws IOException
    {
        //emitting the top N words in descending order
        for (Map.Entry<Integer, Text> entry : top_N_words.entrySet())
        {
            result.collect(entry.getValue(), new IntWritable(entry.getKey()));
        }
    }
}
```

## Connect Class

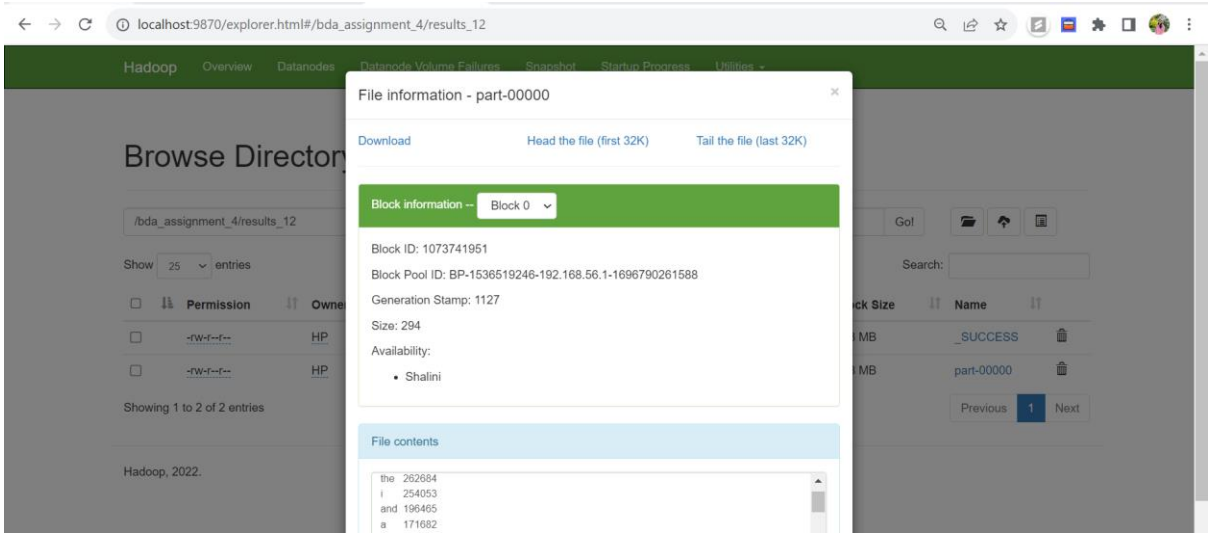
```java
package bda_4;
//importing necessary libraries
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.TextOutputFormat;
public class connect {
    public static void main(String[] args) throws IOException{
        //task 12
        if (args.length != 3) {
            System.err.println("Please give 3 arguments input, output, N value");
            System.exit( status: 1);
        }
        JobConf job_conf = new JobConf(connect.class);
        job_conf.setJobName("word_count");
        job_conf.setOutputKeyClass(Text.class);
        job_conf.setOutputValueClass(IntWritable.class);
        job_conf.setMapperClass(mapper.class);
```

```java
        //task 12
        if (args.length != 3) {
            System.err.println("Please give 3 arguments input, output, N value");
            System.exit( status: 1);
        }
        JobConf job_conf = new JobConf(connect.class);
        job_conf.setJobName("word_count");
        job_conf.setOutputKeyClass(Text.class);
        job_conf.setOutputValueClass(IntWritable.class);
        job_conf.setMapperClass(mapper.class);
        //2nd part of TASK 12
        //job_conf.setCombinerClass(reducer.class);
        job_conf.setReducerClass(reducer.class);
        job_conf.setInputFormat(TextInputFormat.class);
        job_conf.setOutputFormat(TextOutputFormat.class);
        //taking argument 1 as input
        FileInputFormat.setInputPaths(job_conf,new Path(args[0]));
        //taking argument 2 as output
        FileOutputFormat.setOutputPath(job_conf,new Path(args[1]));
        //task 12
        int N = Integer.parseInt(args[2]);
        job_conf.setInt( name: "top_N", N);
        JobClient.runJob(job_conf);
    }
}
```

## Output:

```
PS D:\MSCS Course\1.3\BDA\Assignments\Assignment 4\word_count> hadoop jar target/word_count-1.0-SNAPSHOT.jar bda_4.connect /bda_assignment_4/dataset/All_Beauty.json /bda
_assignment_4/results_12 30
2023-10-09 17:02:51,296 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2023-10-09 17:02:51,671 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2023-10-09 17:02:52,862 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application w
ith ToolRunner to remedy this.
2023-10-09 17:02:52,937 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/HP/.staging/job_1696794597114_0014
2023-10-09 17:02:53,766 INFO mapred.FileInputFormat: Total input files to process : 1
2023-10-09 17:02:53,794 INFO net.NetworkTopology: Adding a new node: /default-rack/127.0.0.1:9866
2023-10-09 17:02:53,876 INFO mapreduce.JobSubmitter: number of splits:2
2023-10-09 17:02:54,089 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1696794597114_0014
2023-10-09 17:02:54,091 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-10-09 17:02:54,395 INFO conf.Configuration: resource-types.xml not found
2023-10-09 17:02:54,396 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2023-10-09 17:02:54,621 INFO impl.YarnClientImpl: Submitted application application_1696794597114_0014
2023-10-09 17:02:54,696 INFO mapreduce.Job: The url to track the job: http://Shalini:8088/proxy/application_1696794597114_0014/
2023-10-09 17:02:54,701 INFO mapreduce.Job: Running job: job_1696794597114_0014
2023-10-09 17:03:07,025 INFO mapreduce.Job: Job job_1696794597114_0014 running in uber mode : false
2023-10-09 17:03:07,027 INFO mapreduce.Job:  map 0% reduce 0%
2023-10-09 17:03:27,337 INFO mapreduce.Job:  map 31% reduce 0%
2023-10-09 17:03:28,344 INFO mapreduce.Job:  map 65% reduce 0%
2023-10-09 17:03:31,424 INFO mapreduce.Job:  map 81% reduce 0%
2023-10-09 17:03:32,440 INFO mapreduce.Job:  map 100% reduce 0%
2023-10-09 17:03:39,542 INFO mapreduce.Job:  map 100% reduce 100%
2023-10-09 17:03:40,573 INFO mapreduce.Job: Job job_1696794597114_0014 completed successfully
```

**Since we were not asked to do complete text preprocessing of reviewText, results contain words like ‘a’, ‘the’ etc**




**Modifications on Word Count to achieve Task 12 - Top N Words:**

I have modified the reducer and connect java class of word count - task 11 to achieve task 12.

- Connect class: It has N integer to take 3rd argument from the user input(input - 1st argument, output-2nd argument, N – 3rd argument)
- Reducer class: It has same functionality but in addition I used a TreeMap with a custom comparator that sorts the entries based on counts in descending order. This allows to efficiently keep track of the top N words and their frequencies. The close method emits these top N words in descending order of counts.

**Cmd to execute the top N words algorithm:** (30 – top 30 words)

hadoop jar target/word_count-1.0-SNAPSHOT.jar bda_4.connect /bda_assignment_4/dataset/All_Beauty.json /bda_assignment_4/results_12 30

## Using Combiner:

The combiner runs on the output of the mapper class. It performs a local aggregation of word counts. It did not change the results but increases the efficiency of the algorithm.

We just add following line to connect java class file

Job_conf.setCombinerClass(reducer.class);

word_count  Version control

pom.xml (word_count)  mapper.java  connect.java  reducer.java

```java
import org.apache.hadoop.mapred.TextOutputFormat;
public class connect {
    public static void main(String[] args) throws IOException{
        //task 12
        if (args.length != 3) {
            System.err.println("Please give 3 arguments input, output, N value");
            System.exit( status: 1);
        }
        JobConf job_conf = new JobConf(connect.class);
        job_conf.setJobName("word_count");
        job_conf.setOutputKeyClass(Text.class);
        job_conf.setOutputValueClass(IntWritable.class);
        job_conf.setMapperClass(mapper.class);
        //2nd part of TASK 12
        job_conf.setCombinerClass(reducer.class);
        job_conf.setReducerClass(reducer.class);
        job_conf.setInputFormat(TextInputFormat.class);
        job_conf.setOutputFormat(TextOutputFormat.class);
        //taking argument 1 as input
        FileInputFormat.setInputPaths(job_conf,new Path(args[0]));
        //taking argument 2 as output
        FileOutputFormat.setOutputPath(job_conf,new Path(args[1]));
        //task 12
        int N = Integer.parseInt(args[2]);
        job_conf.setInt( name: "top_N", N);
        JobClient.runJob(job_conf);
    }
```