Exp No: 10 **Date:**

HADOOP

DEMONSTRATE THE MAP REDUCE PROGRAMMING MODEL BY COUNTING THE NUMBER OF WORDS IN A FILE

AIM:

To demonstrate the MAP REDUCE programming model for counting the number of words in a file.

PROCEDURE

Step 1 - Open Terminal

$ su hduser

Password:

Step 2 - Start dfs and mapreduce services

$ cd /usr/local/hadoop/hadoop-2.7.2/sbin

$ start-dfs.sh

$ start-yarn.sh

$ jps

Step 3 - Check Hadoop through web UI

// Go to browser type http://localhost:8088 – All Applications Hadoop Cluster

// Go to browser type http://localhost:50070 – Hadoop Namenode

Step 4 – Open New Terminal

$ cd Desktop/

$ mkdir inputdata

$ cd inputdata/

$ echo “Hai, Hello, How are you? How is your health?” >> hello.txt

$ cat>> hello.txt

Step 5 – Go back to old Terminal

$ hadoop fs –copyFromLocal /home/hduser/Desktop/inputdata/hello.txt /folder/hduser // Check in hello.txt in Namenode using Web UI

Step 6 – Download and open eclipse by creating workspace

Create a new java project.

Step 7 – Add jar to the project

You need to remove dependencies by adding jar files in the hadoop source folder. Now Click on Project tab and go to Properties.Under Libraries tab, click Add External JARs and select all the jars in the folder (click on 1st jar, and Press Shift and Click on last jat to select all jars in between and click ok)

/usr/local/hadoop/hadoop-2.7.2/share/hadoop/commonand

/usr/local/hadoop/hadoop-2.7.2/share/hadoop/mapreduce folders.

Step -8 – WordCount Program

Create 3 java files named

- WordCount.java
- WordCountMapper.java
- WordCountReducer.java

WordCount.java

```
import org.apache.hadoop.conf.Configured;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

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;
import org.apache.hadoop.io.Text;

public class WordCount extends Configured implements Tool {

        @Override
        public int run(String[] arg0) throws Exception {
                // TODO Auto-generated method
                stub if(arg0.length<2)
                {
System.out.println("check the command line arguments");
                }
                JobConf conf=new JobConf(WordCount.class);
                FileInputFormat.setInputPaths(conf, new Path(arg0[0]));
                        FileOutputFormat.setOutputPath(conf, new
Path(arg0[1])); conf.setMapperClass(WordMapper.class);
conf.setReducerClass(WordReducer.class);

                        conf.setOutputKeyClass(Text.class);
                        conf.setOutputValueClass(IntWritable.class);
                        conf.setOutputKeyClass(Text.class);
```

```
                conf.setOutputValueClass(IntWritable.class);
                JobClient.runJob(conf);


        return 0;
    }
    public static void main(String args[]) throws Exception
    {
        int exitcode=ToolRunner.run(new WordCount(),
        args); System.exit(exitcode);

    }
}
```

WordCountMapper.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.Mapper;
public class WordCountMapper extends MapReduceBase implements
```

```
Mapper<LongWritable,Text,Text,IntWritable>
{
        @Override
        public void map(LongWritable arg0, Text arg1, OutputCollector<Text,
IntWritable> arg2, Reporter arg3)
                        throws IOException {
                // TODO Auto-generated method stub

                String s=arg1.toString();
                for(String word:s.split(" "))
                {
arg2.collect(new Text(word),new IntWritable(1));
                }
        }
}
```

WordCountReducer.java

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

```
public class WordCountReducer implements
        Reducer<Text,IntWritable,Text,IntWritable> {  @Override
public void configure(JobConf arg0) {
        // TODO Auto-generated method stub
    }
    @Override
    public void close() throws IOException {
        // TODO Auto-generated method stub
    }
    @Override
public void reduce(Text arg0, Iterator<IntWritable> arg1,
OutputCollector<Text, IntWritable> arg2, Reporter arg3)
            throws IOException {
        // TODO Auto-generated method
        stub int count=0;
        while(arg1.hasNext())
        {
            IntWritable i=arg1.next();
            count+=i.get();
        }
        arg2.collect(arg0,new IntWritable(count));

    }
}
```

Step 9 - Create JAR file

Now Click on the Run tab and click Run-Configurations. Click on New Configuration button on the left top side and Apply after filling the following properties.

Step 10 - Export JAR file

Now click on File tab and select Export. under Java, select Runnable Jar.

In Launch Config – select the config fie you created in Step 9 (WordCountConfig).

➢ Select an export destination (let's say desktop.)

➢ Under Library handling, select Extract Required Libraries into generated JAR and click Finish. ➢ Right-Click the jar file, go to Properties and under Permissions tab, Check Allow executing file

as a program. and give Read and Write access to all the users

Step 11 – Go back to old Terminal for Execution of WordCount Program $hadoop jar wordcount.jar/usr/local/hadoop/input/usr/local/hadoop/output

Step 12 – To view results in old Terminal
$hdfs dfs -cat /usr/local/hadoop/output/part-r-00000

Step 13 - To Remove folders created using hdfs

$ hdfs dfs -rm -R /usr/local/hadoop/output

OUTPUT:

```
hayagreevan@fedora:~/cc$ ls
hello.txt  mapper.java  reducer.java  word_count.java
hayagreevan@fedora:~/cc$ hdfs dfs -ls /
Found 7 items
drwxr-xr-x   - hayagreevan supergroup          0 2024-08-26 19:30 /exp2
drwxr-xr-x   - hayagreevan supergroup          0 2024-08-28 12:29 /exp3
drwxr-xr-x   - hayagreevan supergroup          0 2024-08-28 13:04 /exp4
drwxrwxrwx   - hayagreevan supergroup          0 2024-09-13 14:31 /exp6
drwxr-xr-x   - hayagreevan supergroup          0 2024-10-09 14:13 /project
drwxrwxr-x   - hayagreevan supergroup          0 2024-10-09 13:12 /tmp
drwxr-xr-x   - hayagreevan supergroup          0 2024-08-28 14:58 /user
hayagreevan@fedora:~/cc$ hdfs dfs -mkdir /cc
hayagreevan@fedora:~/cc$ hdfs dfs -put hello.txt /cc
hayagreevan@fedora:~/cc$ hdfs dfs -ls /cc
Found 1 items
-rw-r--r--   1 hayagreevan supergroup         51 2024-11-16 18:25 /cc/hello.txt
```

```
hayagreevan@fedora:~/cc$ javac -classpath $HADOOP_HOME/share/hadoop/common/*:$HADOOP
_HOME/share/hadoop/mapreduce/*:. -d . WordCountMapper.java WordCountReducer.java Wor
dCount.java
hayagreevan@fedora:~/cc$ jar -cvf WordCount.jar -C . .
added manifest
adding: hello.txt(in = 51) (out= 46)(deflated 9%)
adding: WordCountMapper.java(in = 711) (out= 319)(deflated 55%)
adding: WordCountReducer.java(in = 866) (out= 344)(deflated 60%)
adding: WordCount.java(in = 1221) (out= 475)(deflated 61%)
adding: WordCountMapper.class(in = 1736) (out= 683)(deflated 60%)
adding: WordCountReducer.class(in = 1659) (out= 656)(deflated 60%)
adding: WordCount.class(in = 1698) (out= 861)(deflated 49%)
hayagreevan@fedora:~/cc$
```

```
hayagreevan@fedora:~/cc$ hadoop jar ./wordcount.jar WordCount /cc/hello.txt /cc/output
2024-11-16 23:48:37,901 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-11-16 23:48:42,003 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2024-11-16 23:48:42,584 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hayagreevan/.staging/job_1731780985282_0001
2024-11-16 23:48:46,567 INFO input.FileInputFormat: Total input files to process : 1
2024-11-16 23:48:48,563 INFO mapreduce.JobSubmitter: number of splits:1
2024-11-16 23:48:50,097 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1731780985282_0001
2024-11-16 23:48:50,100 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-11-16 23:48:50,916 INFO conf.Configuration: resource-types.xml not found
2024-11-16 23:48:50,918 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2024-11-16 23:48:52,349 INFO impl.YarnClientImpl: Submitted application application_1731780985282_0001
2024-11-16 23:48:52,554 INFO mapreduce.Job: The url to track the job: http://fedora:8088/proxy/application_1731780985282_0001/
2024-11-16 23:48:52,555 INFO mapreduce.Job: Running job: job_1731780985282_0001
2024-11-16 23:49:36,582 INFO mapreduce.Job: Job job_1731780985282_0001 running in uber mode : false
2024-11-16 23:49:36,585 INFO mapreduce.Job:  map 0% reduce 0%
2024-11-16 23:50:03,820 INFO mapreduce.Job:  map 100% reduce 0%
2024-11-16 23:50:19,262 INFO mapreduce.Job:  map 100% reduce 100%
2024-11-16 23:50:21,445 INFO mapreduce.Job: Job job_1731780985282_0001 completed successfully
2024-11-16 23:50:21,983 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=111
                FILE: Number of bytes written=553365
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=150
                HDFS: Number of bytes written=63
                HDFS: Number of read operations=8
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=21213
                Total time spent by all reduces in occupied slots (ms)=12246
                Total time spent by all map tasks (ms)=21213
                Total time spent by all reduce tasks (ms)=12246
                Total vcore-milliseconds taken by all map tasks=21213
```

```
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=21213
                Total time spent by all reduces in occupied slots (ms)=12246
                Total time spent by all map tasks (ms)=21213
                Total time spent by all reduce tasks (ms)=12246
                Total vcore-milliseconds taken by all map tasks=21213
                Total vcore-milliseconds taken by all reduce tasks=12246
                Total megabyte-milliseconds taken by all map tasks=21722112
                Total megabyte-milliseconds taken by all reduce tasks=12539904
        Map-Reduce Framework
                Map input records=1
                Map output records=9
                Map output bytes=87
                Map output materialized bytes=111
                Input split bytes=99
                Combine input records=0
                Combine output records=0
                Reduce input groups=8
                Reduce shuffle bytes=111
                Reduce input records=9
                Reduce output records=8
                Spilled Records=18
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=859
                CPU time spent (ms)=6320
                Physical memory (bytes) snapshot=558489600
                Virtual memory (bytes) snapshot=5174104064
                Total committed heap usage (bytes)=433061888
                Peak Map Physical memory (bytes)=320790528
                Peak Map Virtual memory (bytes)=2583846912
                Peak Reduce Physical memory (bytes)=237699072
                Peak Reduce Virtual memory (bytes)=2590257152
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=51
        File Output Format Counters
                Bytes Written=63
hayagreevan@fedora:~/cc$
```

```
hayagreevan@fedora:~/cc$ hdfs dfs -cat /cc/output/*
Hello,  1
How     2
are     1
health?"        1
is      1
you?    1
your    1
“Hai,   1
```


RESULT

Thus a word count program in java is implemented using Map Reduce.