**ASSIGNMENT 11**

**Aim:** Write a Hadoop program that counts the number of occurrences of each word in a text file.

### Pre-requisites:

* + - 1. Hadoop installation must be completed successfully.
      2. Single node Hadoop cluster must be configured and running.
      3. Eclipse must be installed as the MapReduce word count will be run from eclipse IDE.

### Objectives:

To learn the configuration of single node Hadoop clustering & implementation of word count.

**Theory:** In Hadoop, MapReduce is a computation that decomposes large manipulation jobs into individual tasks that can be executed in parallel cross a cluster of severs. The result of tasks can be joined together to compute final results.

### MapReduce consist of 2 Steps:

**Map Function:** It takes a set of data and converts it into another set of data, where individual elements are broken down into tuples.

**Reduce Function:** Takes the output from Map as an input and combines those data tuples into a smaller set of tuples.

Hadoop WordCount operation occurs in 3 stages:

### Mapper Phase:

The text from an input text file is tokenized into words to form a key value pair with all the words present is input text file. The key is the word from the input file and value is 1.

Eg: If you consider the sentence “An elephant is an animal “.The mapper phase will split the string into individual words. In this case, the entire sentence will be split into 5 tokens with a value 1 as shown below.

Key-value pairs from Hadoop Map Phase Execution: (an,1)

(elephant,1) (is,1)

(an,1) (animal,1)

### Shuffle phase

After the map phase execution is completed successfully shuffle phase is executed automatically wherein the key-value pairs generated in the map phase are taken as input and then sorted in alphabetical order. After shuffle phase output will be:

(an,1) (animal,1) (elephant,1) (is,1)

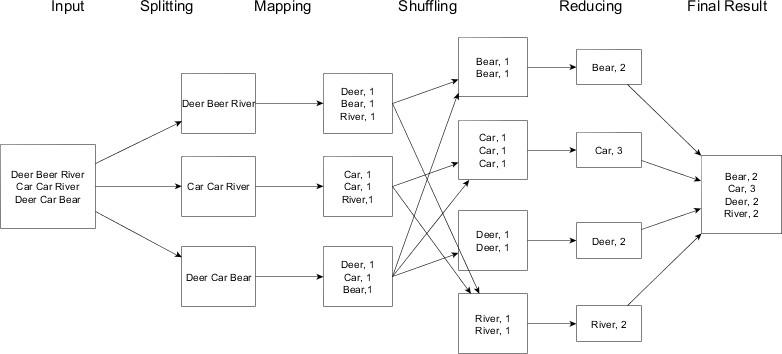
### Reduce Phase

In this phase, as the keys are grouped together and the values for similar keys are added up to find the occurences for a particular word. It is like an aggregation phase for the keys generated by the map phase. The reducer phase takes the output of shuffle phase as input and than reduces the key –value pairs to unique keys with values added up. After the execution of reduce phase, output will be

(an,2) (animal,1) (elephant,1) (is,1)

This is how MapReduce program executes & counts the number of occurences of a word in any given input file.

Wordcount Process using MapReduce:



**Code:**

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]));

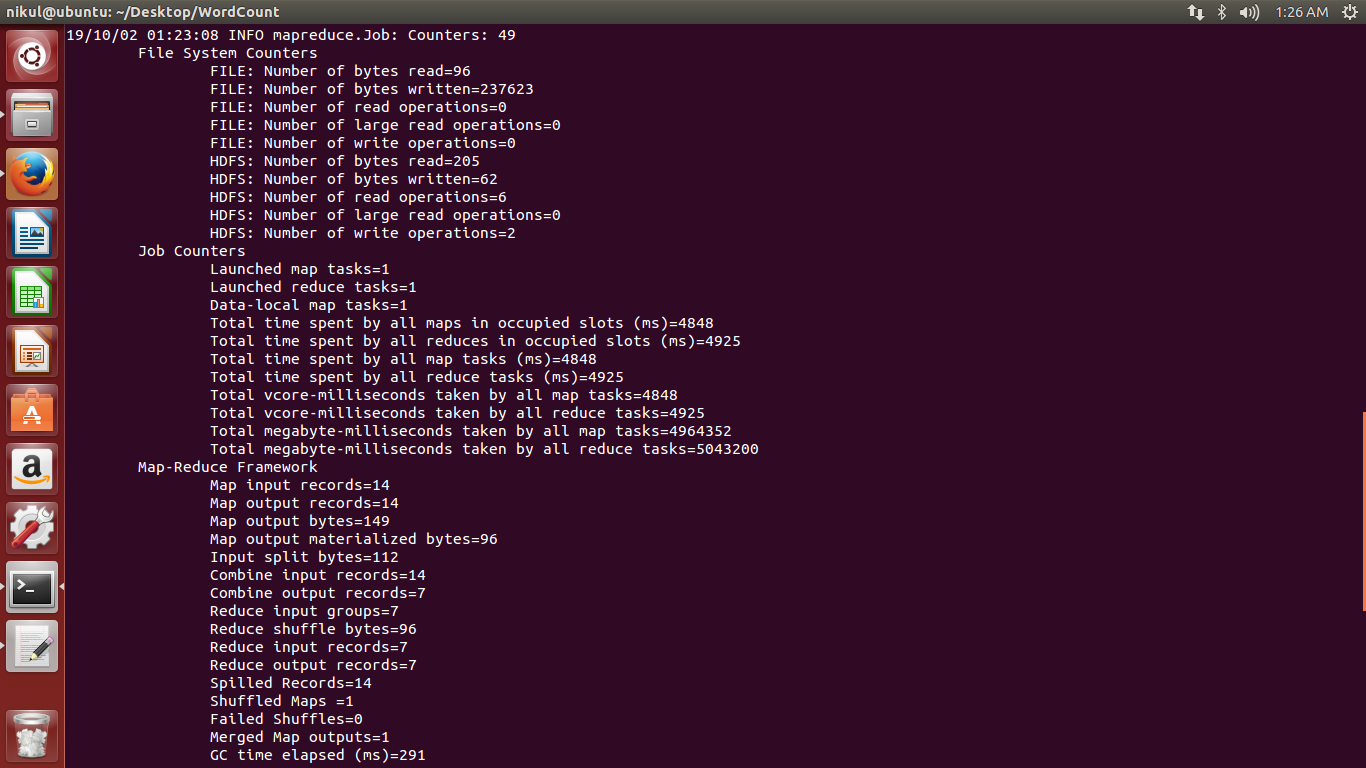
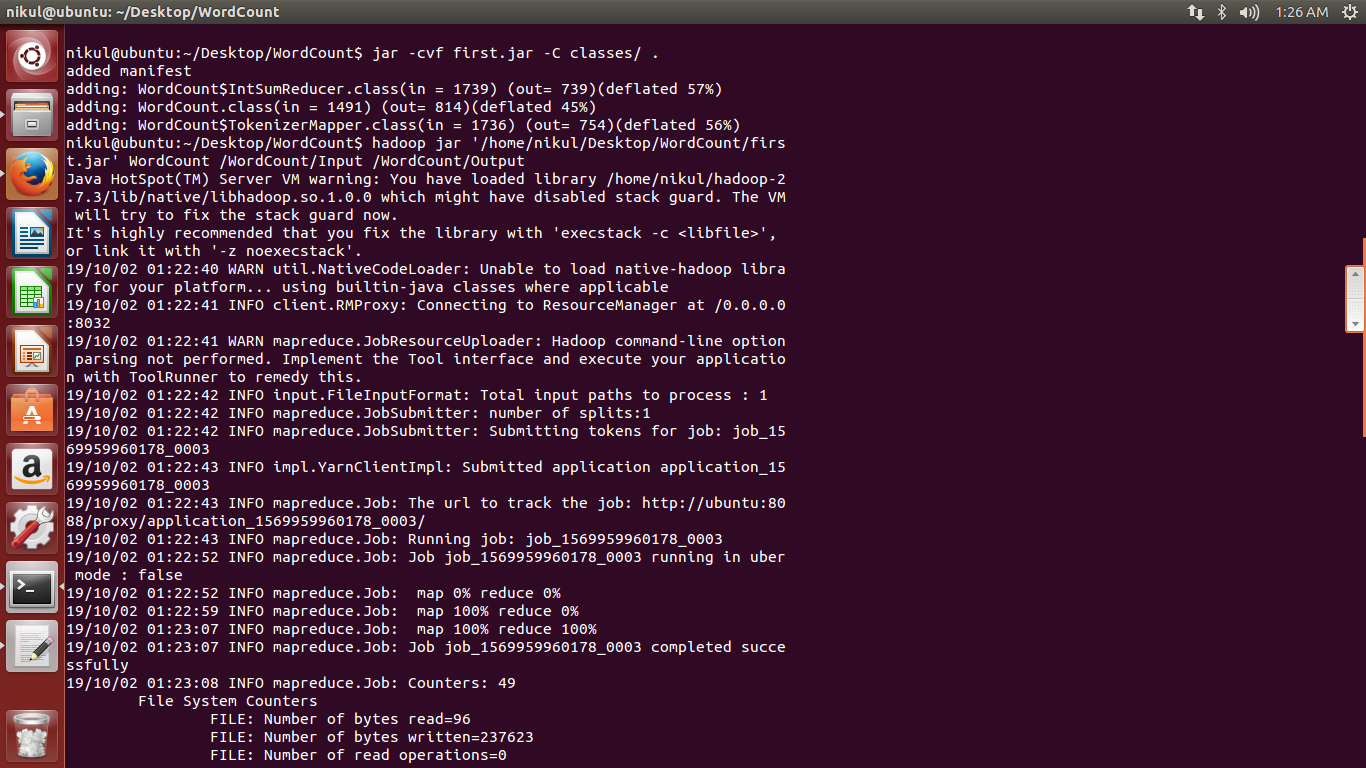
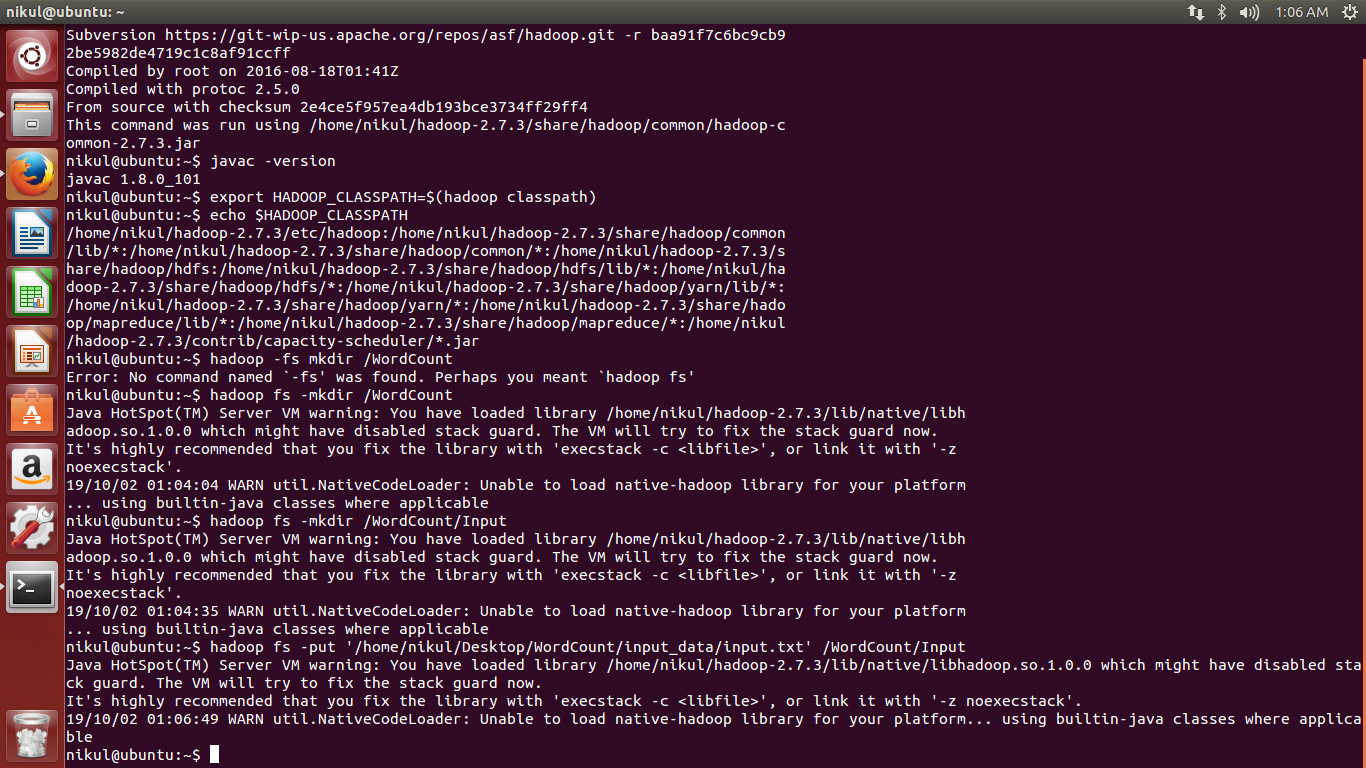
FileOutputFormat.setOutputPath(job, new Path(args[1]));

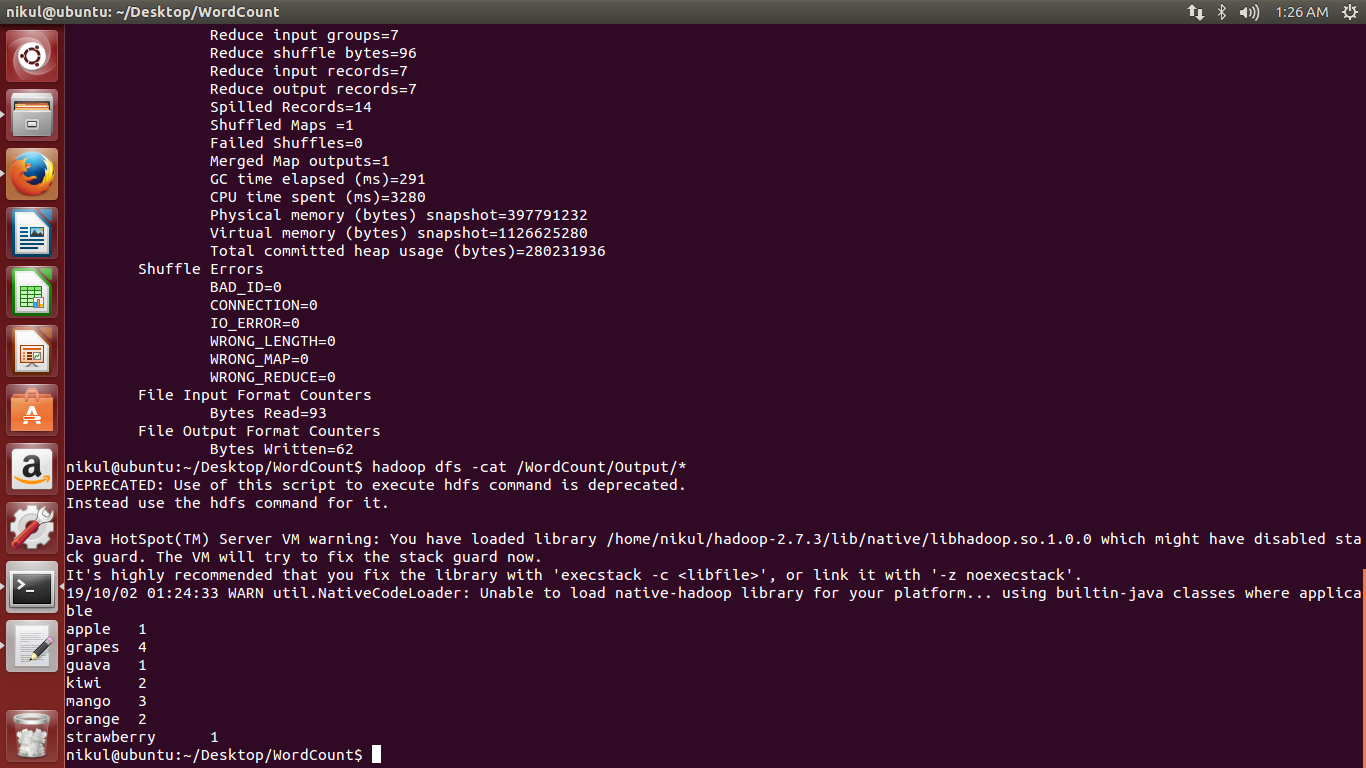
System.exit(job.waitForCompletion(true) ? 0 : 1);

}

}

**Output:**

****

****

### Conclusion:

Thus, we have implemented MapReduce that counts the no. of occurrences of each word in a file.