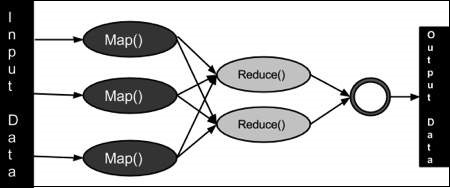
**MapReduce Tutorial:**

*MapReduce* is a framework using which we can write applications to process huge amounts of data, in parallel, on large clusters of commodity hardware in a reliable manner.

**Map stage** − The map or mapper’s job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS).

**Reduce stage** − This stage is the combination of the **Shuffle**stage and the **Reduce** stage. The Reducer’s job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.



* **PayLoad** − Applications implement the Map and the Reduce functions, and form the core of the job.
* **Mapper** − Mapper maps the input key/value pairs to a set of intermediate key/value pair.
* **NamedNode** − Node that manages the Hadoop Distributed File System (HDFS).
* **DataNode** − Node where data is presented in advance before any processing takes place.
* **MasterNode** − Node where JobTracker runs and which accepts job requests from clients.
* **SlaveNode** − Node where Map and Reduce program runs.
* **JobTracker** − Schedules jobs and tracks the assign jobs to Task tracker.
* **Task Tracker** − Tracks the task and reports status to JobTracker

.

* **Job** − A program is an execution of a Mapper and Reducer across a dataset

.

* **Task** − An execution of a Mapper or a Reducer on a slice of data.
* **Task Attempt** − A particular instance of an attempt to execute a task on a SlaveNode.

**Word Count Program With MapReduce and Java:**

**DriverCode**

// Importing libraries

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);

    }

}

**Reducer Code:** You have to copy paste this program into the WCReducer Java Class file.

// Importing 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;

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));

    }

}

**Mapper Code:** You have to copy paste this program into the WCMapper Java Class file.

// Importing libraries

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));

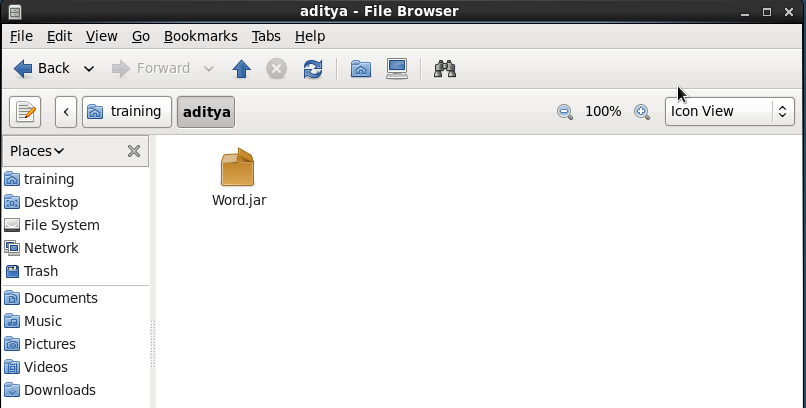
            }

        }

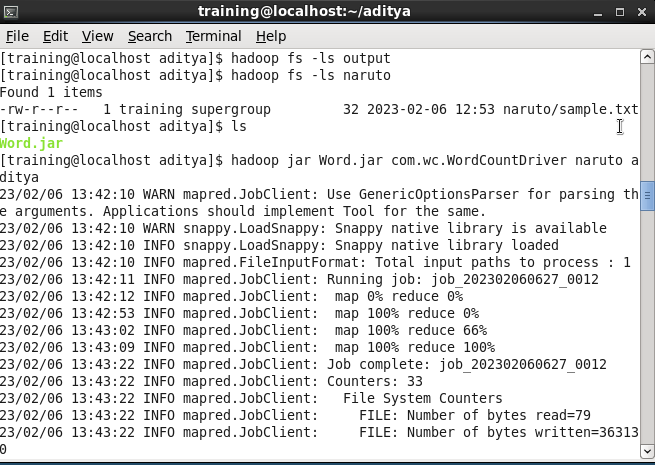
    }

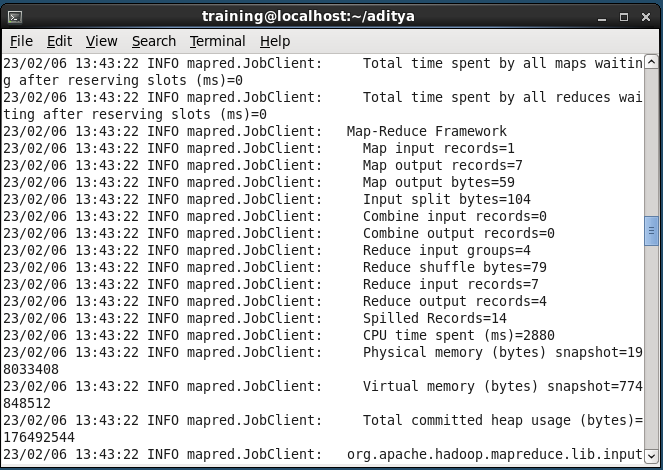
}

**Create .jar file:**

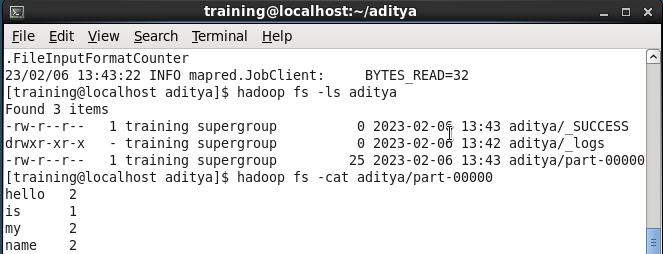
****

**Output of word count:**

****

****

**Word count final output:**

****