**Assignment No:**

**Assignment Title:** Design a distributed application using MapReduce and Hadoop for:

1. Character counting in a given text file.
2. Counting no. of occurrences of every word in a given text file.

Objective:

1. To learn the how to use the MapReduce and Hadoop
2. To learn how to write MapReduce Program in Java or C++ or Python

Input: Hello Hadoop

Expected Output: Total Number of characters are 14.

**Implementation Details:**

1. Open Eclipse and open new Java Project
2. Create Three Java Classes into the project.
   1. Name them CharCountDriver (having the main function), CharCountMapper, CharCountReducer.

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

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 CharCountMapper extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWritable>{

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

String line = value.toString();

String tokenizer[] = line.split("");

for(String SingleChar : tokenizer)

{

Text charKey = new Text(SingleChar);

IntWritable One = new IntWritable(1);

output.collect(charKey, One);

}

} }

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

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 CharCountReducer extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {

public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException

{

int sum = 0;

while (values.hasNext()) {

sum += values.next().get();

}

output.collect(key, new IntWritable(sum));

}

}

**Driver Code: You have to copy-paste this below program into the CharCountDriver Java Class file.**

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 CharCountDriver {

public static void main(String[] args)

throws IOException

{

JobConf conf = new JobConf(CharCountDriver.class);

conf.setJobName("CharCount");

conf.setOutputKeyClass(Text.class);

conf.setOutputValueClass(IntWritable.class);

conf.setMapperClass(CharCountMapper.class);

conf.setCombinerClass(CharCountReducer.class);

conf.setReducerClass(CharCountReducer.class);

conf.setInputFormat(TextInputFormat.class);

conf.setOutputFormat(TextOutputFormat.class);

FileInputFormat.setInputPaths(conf, new Path(args[0]));

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

JobClient.runJob(conf);

}

**Step 3: Add the external jar which are need to import in program**

First, Check the Hadoop Version with the command: hadoop version

Based on the Hadoop version Download the jar package Hadoop Common

<https://mvnrepository.com/artifact/org.apache.hadoop/hadoop-common>

and Hadoop MapReduce Core

https://mvnrepository.com/artifact/org.apache.hadoop/hadoop-mapreduce-client-core

**Step 4: Now we add these external jars to our CharCount project.**

Right Click on CharCount -> then select Build Path-> Click on Configure Build Path and select Add External jars…. and add jars from it’s download location then click -> Apply and Close.

**Step 5: Now export the project as a jar file.**

Right-click on **CharCount** choose **Export**.. and go to **Java** -> **JAR file** click -> **Next** and choose your export destination then click -> **Next**. Choose Main Class as **CharCount** by clicking -> **Browse** and then click -> **Finish** -> **Ok**.

Once the Jar file is successfully created and saved at /Documents directory for example name is charectercount.jar

**Step 6: Create a simple text file and add the text into it.**

We can create the text file by gedit or vim in ubuntu.

**Step 7: Start our Hadoop Daemons**

start-dfs.sh

start-yarn.sh

**Step 8: Move your test.txt file to the Hadoop HDFS.**

**Syntax:** hdfs dfs -put /file\_path /destination

In below command / shows the root directory of our HDFS.

hdfs dfs -put /home/user/Documents/test.txt /

Check the file is present in the root directory of HDFS or not.

hdfs dfs -ls /

**Step 9: Now Run your Jar File with the below command and produce the output in CharCountResult File**.

**Syntax:**

hadoop jar /jar\_file\_location /dataset\_location\_in\_HDFS /output-file\_name

**Command:**

hadoop jar /home/user/Documents/charectercount.jar /test.txt /CharCountResult

**Step 10: Now Move to localhost:50070/, under utilities select Browse the file system and download part-r-00000 in /CharCountResult directory to see result**. we can also check the result i.e. that part-r-00000 file with cat command as shown below.

hdfs dfs -cat /CharCountResult/part-00000

**Conclusion:**

**Sample Question for oral evaluation:**

**References:**

1. YouTube Video

[HOW TO INSTALL HADOOP 2 9 0 SINGLE NODE CLUSTER AND RUN WORDCOUNT PROGRAM ON UBUNTU 16 04 - YouTube](https://www.youtube.com/watch?v=5rJTPMLKsq0)