
Step-1: Open Virtual box and then start **clouderaquickstart**

Step-2: Open **Eclipse** present on the cloudera desktop

Step-3: Creating Java Project

3.1: File-> New -> Project ->**Java Project** -> Next

(“WordCount” is the Project name)

Step-4: Adding the Hadoop libraries to the Project

4.1: Click on **Add External Jars** button, the, File File System >usr> lib>**Hadoop**, select all the libraray (jar) files and then click OK button

4.2: Now Click on Add External Jars button, the, File File System >usr> lib> Hadoop>**client**, select all the libraray (jar) files and then click OK button

4.3: Click **finish** button

Step-5: Create Java MapperReduce program

5.1: In the explorer panel Right click on “src” folder of the project WordCount

New> Class>**Name** textfield give as “WordCount” and click Finish button.

5.2: Type the code for WordCount program with import files, Mapper class, Reducer class Driver class with main method

Step-6: Export the project as JAR

6.1: Right click WordCount project and select “Export” >> Java >> Jar file >> Next>> in the JAR file textfield give as /home/cloudera/WordCount.jar, click Finish button>> OK

6.2: Open **cloudera@quickstart** terminal and verify the jar file using **ls** command

Step-7: Create the input file for the MapReduce program by typing command

**cat > /home/cloudera/inputFile.txt**

Verify the data contents by **cat /home/cloudera/inputFile.txt**

Step-8: Move the input file created in local system to hdfs store by

**Hdfsdfs -put /home/cloudera/inputFile.txt /WCInput/**

View the contents of the file moved to hdfs by typing command

**hdfsdfs -cat /WCInput/inputFile.txt**

Step-9: Run MapReduce program on Hadoop by typing command

hadoop jar **/home/hadoop/InvertedIndex.jar InvertedIndex/invInput/inputFile.txt /InvOutput**

Each time you run the above command; you need to give different name for the output directory.

Step-10: View the output directory content by hdfsdfs -ls /WCOutput of the program/job executed, **hdfsdfs -cat /WCOutput/part-r-00000**

**//word count java program code using mapreduce in hadoop framework**

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 //driver class

{ //Mapper class

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

{ StringTokenizeritr = new StringTokenizer(value.toString());

while (itr.hasMoreTokens())

{ word.set(itr.nextToken());

context.write(word, one);

}

}

} //End Mapper class

//Reducer class

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 (IntWritableval : values)

{ sum += val.get();

}

result.set(sum);

context.write(key, result);

}

} //End Reducer class

//Driver class main()

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);

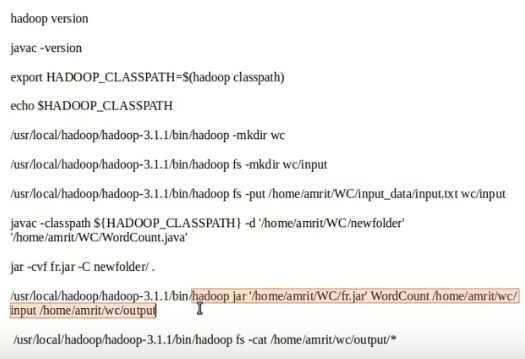
FileInputFormat.addInputPath(job, new Path(args[0]));

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

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

} //End main()

} //End WordCount class

****