Assignment 11

Step 1- To start :

* start-all.sh or ( start-dfs.sh & start-yarn.sh)
* To verify, type : jps
* Output (NameNode must be there):

5776 DataNode

6211 ResourceManager

6760 Jps

5980 SecondaryNameNode

5613 NameNode

6351 NodeManager

Step 2- Create Directory :

* hadoop fs -mkdir /user7
* hadoop fs -mkdir /user7/input
* To verify, go to : localhost:9870 -> Utilities->Browse File System

Step 3 - Create input.txt

* touch input.txt
* To verify, type : ls
* nano input.txt
* Type the input then, Ctrl X -> y ->enter

Step 4 – Put input.txt in the directory

* Hadoop fs -put input.txt /user7/input

Step 5 – Create Java file

* touch WordCount.java
* nano WordCount.java
* Code:

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;

import java.io.IOException;

import java.util.StringTokenizer;

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().replaceAll("[^a-zA-Z]", "").toLowerCase()); // Clean words

if (!word.toString().isEmpty()) {

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);

    }

 }

* To verify, type: ls
* javac -classpath $(hadoop classpath) WordCount.java
* jar cf wordcount.jar WordCount\*.class
* hadoop jar wordcount.jar WordCount /user7/input /user7/output

Step 6 – To view output

* hadoop fs -cat /user7/output/part-r-00000