Name : Soham Kumawat

Roll No. : TCOB57

Assignment No. : 12 (B2)

Subject : DSBDA Lab

### 1. LogFileMapper.java (Used for mapping the IP addresses from the input CSV file)

package LogFileCountry;

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.\*;

public class LogFileMapper extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {

private final static IntWritable one = new IntWritable(1);

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

String valueString = value.toString();

String[] SingleIpData = valueString.split("-");

output.collect(new Text(SingleIpData[0]), one);

}

}

### 2. LogFileReduce.java (Used for reducing the data received from the mapper process to the final output)

package LogFileCountry;

import java.io.IOException;

import java.util.\*;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapred.\*;

public class LogFileReducer extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {

public void reduce(Text t\_key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {

Text key = t\_key;

int frequencyForIp = 0;

while (values.hasNext()) {

// replace type of value with the actual type of our value

IntWritable value = (IntWritable) values.next();

frequencyForIp += value.get();

}

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

}

}

### 3. LogFileCountryDriver.java (The driver code to run MapReduce on HDFS)

package LogFileCountry;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.\*;

import org.apache.hadoop.mapred.\*;

public class LogFileCountryDriver {

public static void main(String[] args) {

JobClient my\_client = new JobClient();

// Create a configuration object for the job

JobConf job\_conf = new JobConf(LogFileCountryDriver.class);

// Set a name for the job

job\_conf.setJobName("LogFileIP");

// Specify data type of output key and value

job\_conf.setOutputKeyClass(Text.class);

job\_conf.setOutputValueClass(IntWritable.class);

// Specify names of Mapper and Reducer Class

job\_conf.setMapperClass(LogFileCountry.LogFileMapper.class);

job\_conf.setReducerClass(LogFileCountry.LogFileReducer.class);

// Specify formats of the data type of Input and Output

job\_conf.setInputFormat(TextInputFormat.class);

job\_conf.setOutputFormat(TextOutputFormat.class);

// Set input and output directories using command line arguments,

// arg[0] = name of input directory on HDFS, and arg[1] = name of output directory to be created to store the output file.

FileInputFormat.setInputPaths(job\_conf, new Path(args[0]));

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

my\_client.setConf(job\_conf);

try {

// Run the job

JobClient.runJob(job\_conf);

} catch (Exception e) {

e.printStackTrace();

}

}

}

### Log File Input Sample (log\_file.txt):

0.223.157.186 - - [15/Jul/2009:20:50:32 -0700] "GET /assets/js/the-associates.js HTTP/1.1" 304 -

10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/home-logo.png HTTP/1.1" 304 -

10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/dummy/primary-news-2.jpg HTTP/1.1" 304 -

10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/dummy/primary-news-1.jpg HTTP/1.1" 304 -

10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/home-media-block-placeholder.jpg HTTP/1.1" 304 -

10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/dummy/secondary-news-4.jpg HTTP/1.1" 304 -

10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/loading.gif HTTP/1.1" 304 -

10.223.157.186 - - [15/Jul/2009:20:50:33 -0700] "GET /assets/img/search-button.gif HTTP/1.1" 304 -

### Output Sample (part-00000.txt on Hadoop):

10.1.1.236

10.1.181.142

10.1.232.315

10.10.55.142

10.102.101.661

10.103.184.104

10.103.190.8153

10.103.63.291

10.104.73.511

10.105.160.1831

10.108.91.15115

10.109.21.761

10.11.131.40

10.111.71.208

10.112.227.1846

10.114.74.30

10.115.118.78

10.117.224.230