DPA – Lab 7

//AverageReducer.java

package stubs;

public class AverageReducer extends Reducer<Text, IntWritable, Text, Text> {

public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {

int sum = 0;

int count = 0;

for (IntWritable val : values) {

sum += val.get();

count++;

}

double average = (double) sum / count;

context.write(key, new Text(String.format("%.2f", average)));

}

}

//AverageWordLength.java

package stubs;

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.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class AverageWordLength {

public static void main(String[] args) throws Exception {

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "average word length");

job.setJarByClass(AvgWordLength.class);

job.setMapperClass(LetterMapper.class);

job.setReducerClass(AverageReducer.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);

}

}

//LetterMapper.java

public class LetterMapper extends Mapper<LongWritable, Text, Text, IntWritable> {

private final static IntWritable wordLength = new IntWritable();

private final Text firstLetter = new Text();

public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

for (String word : value.toString().split("\s+")) {

if (!word.isEmpty()) {

firstLetter.set(word.substring(0, 1));

wordLength.set(word.length());

context.write(firstLetter, wordLength);

}

}

}

}

1. Total number of rows in the datasource:

SELECT COUNT(\*) FROM wikipedia;

2. Number of distinct pages:

SELECT COUNT(DISTINCT "page") FROM wikipedia;

3. Most accessed language based on channel:

SELECT channel, COUNT(\*) AS access\_count

FROM wikipedia

GROUP BY channel

ORDER BY access\_count DESC

LIMIT 1;

4. Number of robot accesses:

SELECT COUNT(\*) FROM wikipedia

WHERE isRobot = 1;

5. Top 5 most accessed pages:

SELECT "page", COUNT(\*) AS access\_count

FROM wikipedia

GROUP BY "page"

ORDER BY access\_count DESC

LIMIT 5;

6. Distinct robot identity names:

SELECT DISTINCT "robot" FROM wikipedia

WHERE isRobot = 1;

7. Number of edits by anonymous users:

SELECT COUNT(\*) FROM wikipedia

WHERE isAnonymous = 1;

8. Number of edits by logged-in users:

SELECT COUNT(\*) FROM wikipedia

WHERE isAnonymous = 0;

9. Top 5 users by number of edits:

SELECT "user", COUNT(\*) AS edits

FROM wikipedia

GROUP BY "user"

ORDER BY edits DESC

LIMIT 5;

10. Edits per day:

SELECT FLOOR(\_\_time TO DAY) AS day, COUNT(\*) AS edits

FROM wikipedia

GROUP BY day

ORDER BY day ASC;