**LABORATORY PROGRAM – 6**

**Weather Program in HADOOP**

**QUESTIONS**

From the following link extract the weather data

https://github.com/tomwhite/hadoopbook/tree/master/input/ncdc/all

a) Create a MapReduce program to find average temperature for each year from NCDC data set.

b) find the mean max temperature for every month

**Driver Code**

package temp;

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 AverageDriver {

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

if (args.length != 2) {

System.err.println("Please enter both input and output parameters.");

System.exit(-1);

}

// Creating a configuration and job instance

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "Average Calculation");

job.setJarByClass(AverageDriver.class);

// Input and output paths

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

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

// Setting mapper and reducer classes

job.setMapperClass(AverageMapper.class);

job.setReducerClass(AverageReducer.class);

// Output key and value types

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

// Submitting the job and waiting for it to complete

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

}

}

**Mapper Code**

package temp;

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.mapreduce.Mapper;

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

public static final int MISSING = 9999;

@Override

public void map(LongWritable key, Text value, Context context)

throws IOException, InterruptedException {

String line = value.toString();

// Extract year from fixed position

String year = line.substring(15, 19);

int temperature;

// Determine if there's a '+' sign

if (line.charAt(87) == '+') {

temperature = Integer.parseInt(line.substring(88, 92));

} else {

temperature = Integer.parseInt(line.substring(87, 92));

}

// Quality check character

String quality = line.substring(92, 93);

// Only emit if data is valid

if (temperature != MISSING && quality.matches("[01459]")) {

context.write(new Text(year), new IntWritable(temperature));

}

}

}

**Reducer Code**

package temp;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

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

@Override

public void reduce(Text key, Iterable<IntWritable> values,

Context context) throws IOException, InterruptedException {

int sumTemp = 0;

int count = 0;

for (IntWritable value : values) {

sumTemp += value.get();

count++;

}

if (count > 0) {

int average = sumTemp / count;

context.write(key, new IntWritable(average));

}

}

}

**OBSERVATION**







