########################################################################

import java.io.IOException;

import java.util.Map;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.NullWritable;

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 org.apache.hadoop.util.GenericOptionsParser;

public class CrimeDriver {

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

Configuration conf = new Configuration();

String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();

conf.set("fs.defaultFS", "hdfs://localhost:54310");

conf.set("fs.hdfs.impl", org.apache.hadoop.hdfs.DistributedFileSystem.class.getName() );

Job job = new Job(conf, " CrimeData");

job.setJarByClass(CrimeDriver.class);

job.setMapperClass(CrimeMapper.class);

job.setReducerClass(CrimeReducer.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

FileInputFormat.addInputPath(job, new Path("hdfs://localhost:54310/user/hduser/crime\_data1/part-m-00000"));

FileOutputFormat.setOutputPath(job, new Path("hdfs://localhost:54310/home/hduser/Downloads/output4/"));

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

}

}

#####################################################################################

import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.NullWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Mapper.Context;

public class CrimeReducer extends Reducer<Text, IntWritable, Text, IntWritable>

{

@Override

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

{

// Iterate each value to find the total

int sum = 0;

for (IntWritable value : values) {

sum += value.get();

}

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

}

}

#############################################################

import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.NullWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Mapper.Context;

public class CrimeMapper extends Mapper<IntWritable, Text, IntWritable, Text>

{

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

private Text agency = new Text();

@Override

public void map(Text key, Text value,

Mapper.Context context) throws IOException, InterruptedException

{

String line = value.toString();

String[] tokens = line.split(",");

agency.set(tokens[1]);

total = tokens[4];

context.write(agency, total);

}

}

#################################################################