|  |  |
| --- | --- |
|  | |
| import java.io.IOException; | |
|  | import org.apache.hadoop.io.IntWritable; |
|  | import org.apache.hadoop.io.Text; |
|  | import org.apache.hadoop.io.LongWritable; |
|  | import org.apache.hadoop.mapreduce.Job; |
|  | import org.apache.hadoop.mapreduce.Mapper; |
|  | import org.apache.hadoop.mapreduce.Partitioner; |
|  | import org.apache.hadoop.mapreduce.Reducer; |
|  | import org.apache.hadoop.conf.\*; |
|  | import org.apache.hadoop.fs.\*; |
|  | import org.apache.hadoop.mapreduce.lib.input.\*; |
|  | import org.apache.hadoop.mapreduce.lib.output.\*; |
|  | import org.apache.hadoop.util.Tool; |
|  | import org.apache.hadoop.util.ToolRunner; |
|  |  |
|  |  |
|  | public class partitioner extends Configured implements Tool |
|  | { |
|  |  |
|  | public static class MapClass extends Mapper<LongWritable,Text,Text,Text> |
|  | { |
|  | public void map(LongWritable key, Text value, Context context) |
|  | { |
|  | try{ |
|  | String[] str = value.toString().split(","); |
|  | String gender= str[3]; |
|  | //String name= str[1]; |
|  | //String age= str[2]; |
|  | //String salary= str[4]; |
|  | //String myvalue= name + ',' + salary; |
|  | context.write(new Text(gender),new Text(value)); |
|  |  |
|  | } |
|  | catch(Exception e) |
|  | { |
|  | System.out.println(e.getMessage()); |
|  | } |
|  | } |
|  | } |
|  |  |
|  | public static class ReduceClass extends Reducer<Text,Text,Text,IntWritable> |
|  | { |
|  | public int max= 0; |
|  | private Text outputkey= new Text(); |
|  |  |
|  | public void reduce(Text key, Iterable<Text> values,Context context) throws IOException, InterruptedException { |
|  |  |
|  | max= 0; |
|  | for (Text val: values) |
|  | { |
|  | //outputkey.set(key); |
|  | String [] str = val.toString().split(","); |
|  | if(Integer.parseInt(str[4])>max) |
|  | { |
|  | max= Integer.parseInt(str[4]); |
|  | String mykey= str[3] + ',' + str[1] + ',' + str[2]; |
|  | outputkey.set(mykey); |
|  | } |
|  |  |
|  |  |
|  | } |
|  |  |
|  | context.write(outputkey, new IntWritable(max)); |
|  |  |
|  | } |
|  | } |
|  | public static class Coderpartitioner extends Partitioner <Text, Text> |
|  | { |
|  | public int getPartition(Text key, Text value, int numReducetasks) |
|  | { |
|  | String[] str= value.toString().split(","); |
|  | int age= Integer.parseInt(str[2]); |
|  | if(age<=20) |
|  | { |
|  | return 0; |
|  | } |
|  | else if(age >20 && age<=30) |
|  | { |
|  | return 1; |
|  | } |
|  | else |
|  | { |
|  | return 2; |
|  | } |
|  | } |
|  | } |
|  |  |
|  |  |
|  | public int run(String[] args) throws Exception { |
|  | Configuration conf = new Configuration(); |
|  | //conf.set(name", "value") |
|  | Job job = Job.getInstance(conf); |
|  | job.setJarByClass(partitioner.class); |
|  | job.setJobName("Top salaried employees"); |
|  | FileInputFormat.setInputPaths(job, new Path(args[0])); |
|  | FileOutputFormat.setOutputPath(job, new Path(args[1])); |
|  |  |
|  | job.setMapperClass(MapClass.class); |
|  | job.setMapOutputKeyClass(Text.class); |
|  | job.setMapOutputValueClass(Text.class); |
|  |  |
|  | job.setPartitionerClass(Coderpartitioner.class); |
|  | job.setReducerClass(ReduceClass.class); |
|  | job.setNumReduceTasks(3); |
|  | job.setInputFormatClass(TextInputFormat.class); |
|  | job.setOutputFormatClass(TextOutputFormat.class); |
|  |  |
|  |  |
|  | job.setOutputKeyClass(Text.class); |
|  | job.setOutputValueClass(Text.class); |
|  |  |
|  | System.exit(job.waitForCompletion(true)? 0 : 1); |
|  | return 0; |
|  | } |
|  | public static void main(String ar[]) throws Exception |
|  | { |
|  | int res = ToolRunner.run(new Configuration(), new partitioner(),ar); |
|  | System.exit(0); |
|  | } |
|  | } |
|  |  |
|  |  |