package scdx.bigdata.mr;

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.Text;

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

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 java.io.IOException;

public class WordCountDriver {

static class WordCountMapper extends Mapper<LongWritable,Text,Text,IntWritable>{

// <LongWritable, Text, Text, IntWritable>LongWritable表示的是读取的每一样文本数据的下标，第一个Text表示对应的那行数据， 第二个Text 表示的是单词，IntWritable表示的是输出的类型

@Override

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

String[] words=value.toString().split(" ");//拿到数据并切分为每个单词

for(String word:words){

context.write(new Text(word),new IntWritable(1));//输出每个单词类型都为1，例如：<"aaa",1><"bbb",1>

}

}

}

static class WordCountReducer extends Reducer<Text,IntWritable,Text,IntWritable>{

//第一组Text,IntWritable 从map端传过来的数据类型 ；第二组Text,IntWritable 是要输出到文件系统的

@Override

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

//key和values都是从map传过来的，中途将相同值的key合并

int count=0;

for(IntWritable value:values){

count+=value.get();//出现次数累加

}

context.write(key,new IntWritable(count));//输出每个单词及出现的次数

}

}

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

Configuration configuration=new Configuration();

Job job=Job.getInstance(configuration);// 我们提交给集群的是一个job任务，集群也是通过yarn来调度的job，所以new一个job出来

job.setJarByClass(WordCountDriver.class);

job.setNumReduceTasks(2);

job.setMapperClass(WordCountMapper.class);

job.setReducerClass(WordCountReducer.class);// 将mapper、reducer封装进job

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);// 配置mapper输出数据类型

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);// 配置reduce输出数据类型

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

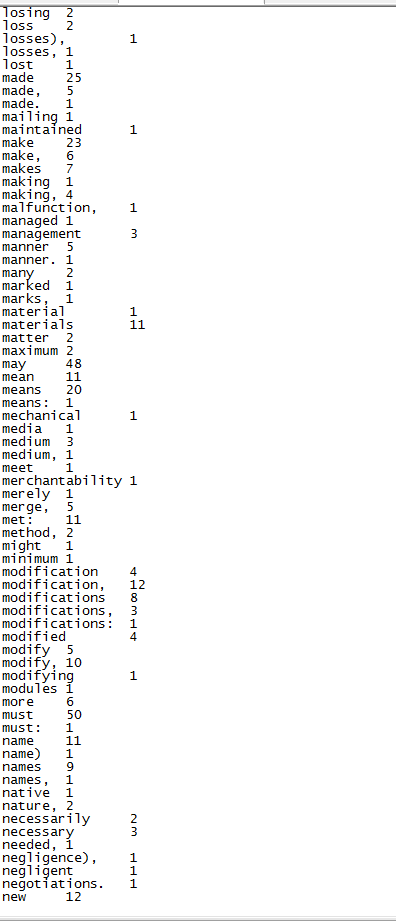
FileOutputFormat.setOutputPath(job,new Path(args[1])); // 配置输入输出文件目录

boolean flag = job.waitForCompletion(true); // 线程一直阻塞，等到执行完后返回结果

System.exit(flag ? 1:0);

}

}



Drive：用来驱动和分配任务

Map：用来阶段接收并发任务

Reduce：用来阶段汇总任务

Job和Configuration：初始化任务