**作业一**

**任务：**

**用mapper和reducer统计每个班年龄最小和最大的人以及这个班的人数。**

**输出格式为：**

**班级号 最小生日 最小生日姓名 最大生日 最大生日姓名 班级人数**

**MAP阶段代码**

java

public static class ClassMapper extends Mapper<LongWritable, Text, Text, Text> {

private Text classNumber = new Text(); // 用于存储班级号

private Text birthAndName = new Text(); // 用于存储生日和姓名

@Override

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

String line = value.toString(); // 将输入行转换为字符串

String[] fields = line.split(","); // 按逗号分割字段

if(fields.length == 3) { // 确保有三个字段

String classNum = fields[0].trim(); // 班级号

String birthDate = fields[1].trim(); // 生日

String name = fields[2].trim(); // 姓名

classNumber.set(classNum); // 设置班级号

birthAndName.set(birthDate + "," + name); // 设置生日和姓名

context.write(classNumber, birthAndName); // 输出班级号和生日姓名

}

}

}

```

**REDUCE阶段代码**

java

public static class ClassReducer extends Reducer<Text, Text, NullWritable, Text> {

private SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss"); // 日期格式

private Text result = new Text(); // 用于存储最终结果

@Override

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

String classNum = key.toString(); // 获取班级号

int count = 0; // 计数班级人数

String oldestName = ""; // 年龄最大的人姓名

String youngestName = ""; // 年龄最小的人姓名

Date oldestDate = null; // 年龄最大的人生日

Date youngestDate = null; // 年龄最小的人生日

// 遍历所有值

for(Text val : values) {

count++; // 计数

String[] birthAndName = val.toString().split(",", 2); // 分割生日和姓名

String birthDateStr = birthAndName[0]; // 生日字符串

String name = birthAndName[1]; // 姓名

try {

Date birthDate = dateFormat.parse(birthDateStr); // 解析生日

// 更新年龄最大的人

if(oldestDate == null || birthDate.before(oldestDate)) {

oldestDate = birthDate;

oldestName = name;

}

// 更新年龄最小的人

if(youngestDate == null || birthDate.after(youngestDate)) {

youngestDate = birthDate;

youngestName = name;

}

} catch(ParseException e) {

e.printStackTrace(); // 异常处理

}

}

// 构造输出字符串

String output = classNum + "\t" + dateFormat.format(youngestDate) + "\t" + youngestName + "\t" +

dateFormat.format(oldestDate) + "\t" + oldestName + "\t" + count;

result.set(output); // 设置结果

context.write(NullWritable.get(), result); // 输出结果

}

}

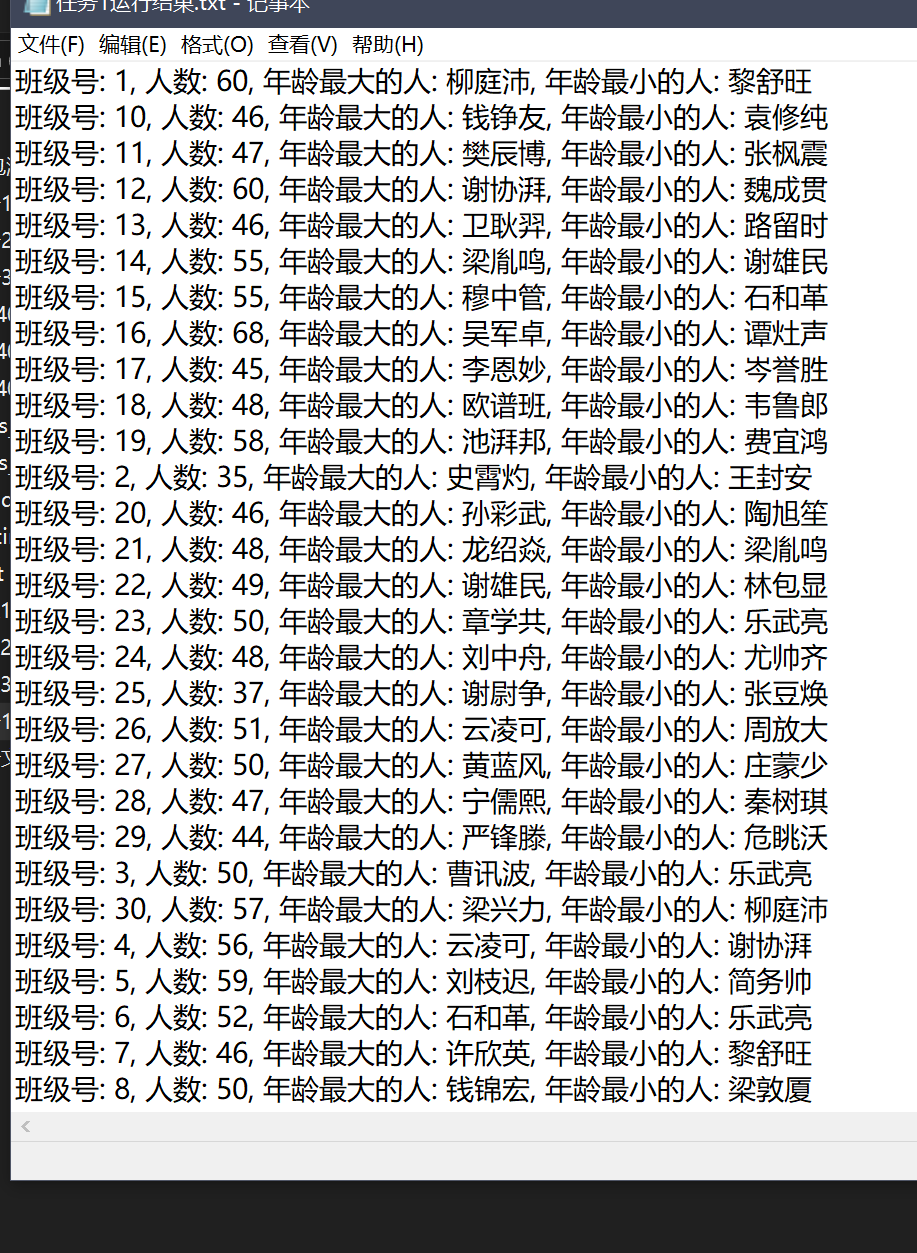


图1 输出结果

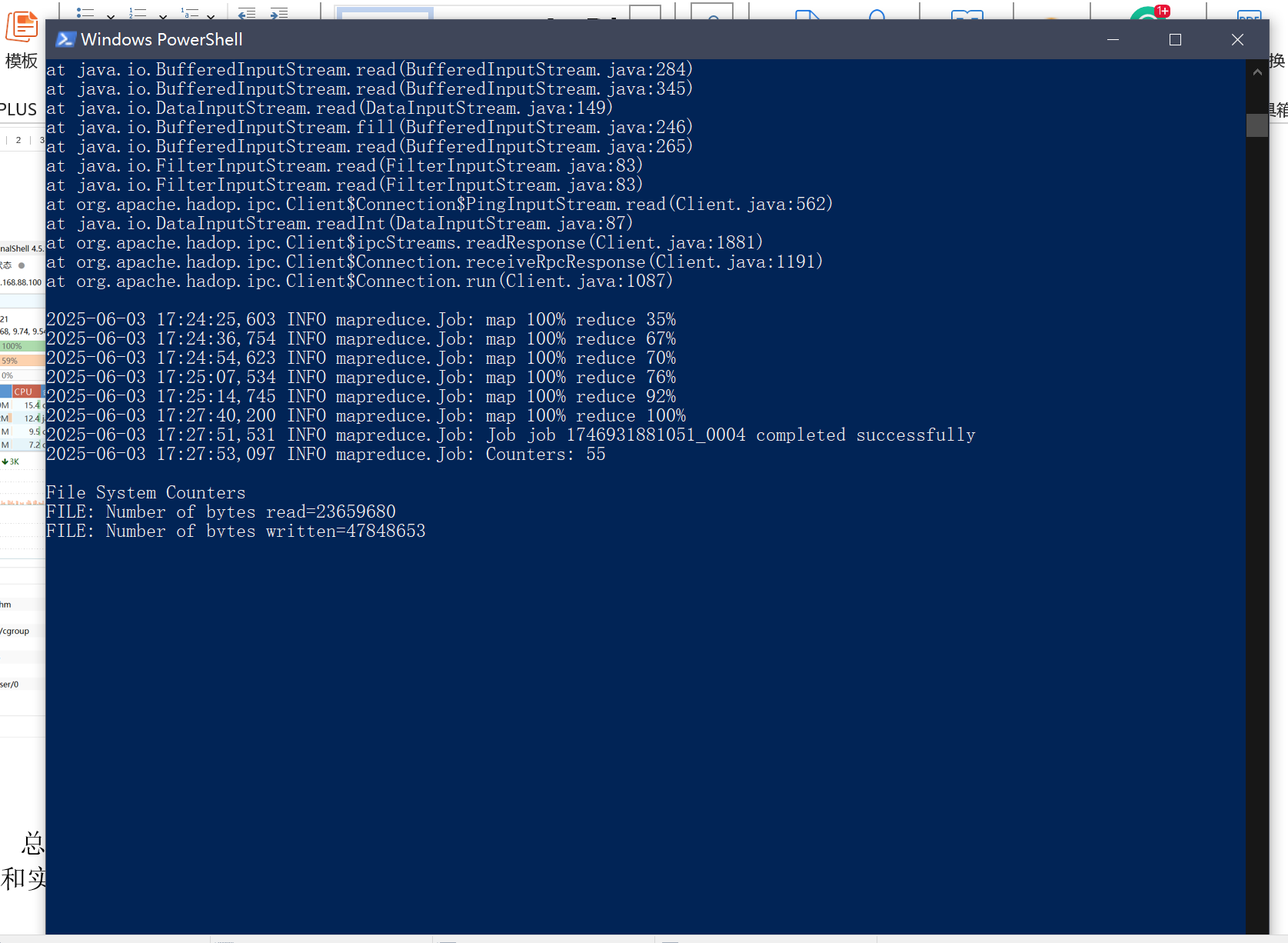


图2 MapReduce运行界面

**实验总结**

在本次实验中，实现了一个基于MapReduce的程序，用于统计每个班级的年龄最小和最大的人以及班级人数。通过Mapper阶段，将输入数据解析并输出班级号及对应的生日和姓名。在Reducer阶段，对同一班级的所有记录进行汇总，计算出班级人数、年龄最小和最大的人及其姓名。最终，按照指定格式输出结果。通过这个实验，深入理解了MapReduce编程模型的基本原理和数据处理流程，并掌握了如何在分布式环境中进行数据统计与分析。