大数据处理综合实验

实验三

Hive表的Group By操作

小组编号：01

组长姓名：张玲 201220052

2023年5月13日

任务一：客户信息表的Group By操作

1.实验目的

编写一个MapReduce程序，实现客户信息表的Group By操作，统计所在国家标识符相同的所有客户的账户余额的总和，并输出所有国家标识符所对应的余额总和信息，即实现

```
select c_nationkey,sum(c_acctbal) from customer group by c_nationkey;
```

输入数据为customer.tbl，包含8个属性，以｜分割，包含客户唯一标识符、客户名称、客户地址、客户所在国家的唯一标识符、客户电话号码、客户账户余额、客户所属市场细分、客户注释。

输出数据将其上传至HDFS个人目录中，使用Hive建表管理保存国家标识符及所对应的余额总和信息。

2.Map和Reduce的设计思路

设计代码使用MapReduce框架实现的客户数据处理程序。在Map阶段将所有输入数据映射到一个Reducer，然后在Reduce阶段对相同c_nationkey的Customer对象进行聚合，并计算相应的c_acctbal总和。最终的输出结果是每个国家标识符（c_nationkey）对应的c_acctbal总和。

Map函数（CustomerMap类）：

输入的数据按行读取，每行代表一个客户信息。map方法首先获取当前输入记录所属的文件名，即数据来源的文件。然后，将输入行解析为一个Customer对象，并将该对象作为值，空文本作为键写入上下文中。

Key为空文本（Text类型），在程序中没有实际的含义，只是为了将所有的输入数据分配到同一个Reducer进行处理。

Value为Customer对象（Customer类型），为输入数据中的一个客户信息。

Reduce函数（CustomerReducer类）：

接收Map阶段输出的键值对，在Reduce阶段，将迭代器中的所有Customer对象存储在一个Vector容器中。通过遍历Vector容器中的Customer对象进行聚合操作。对于每个Customer对象，首先检查其vis属性是否为1，如果是，则跳过该对象。如果vis属性为0，将该对象的c_acctbal属性添加到sum变量中，并将其vis属性设置为1，表示已访问过。对于Vector中剩余的未访问过的Customer对象，如果其c_nationkey与当前对象相同，则也将其c_acctbal添加到sum变量中，并将其vis属性设置为1。将聚合结果构建成一个键值对。将聚合结果写入上下文。

Key是一个Text对象，表示聚合结果的键。在该程序中，键的格式是"c_nationkey sum"，即国家关键字和对应的账户余额总和。

Value是一个NullWritable对象，值可以为空。

3.Map和Reduce的伪代码

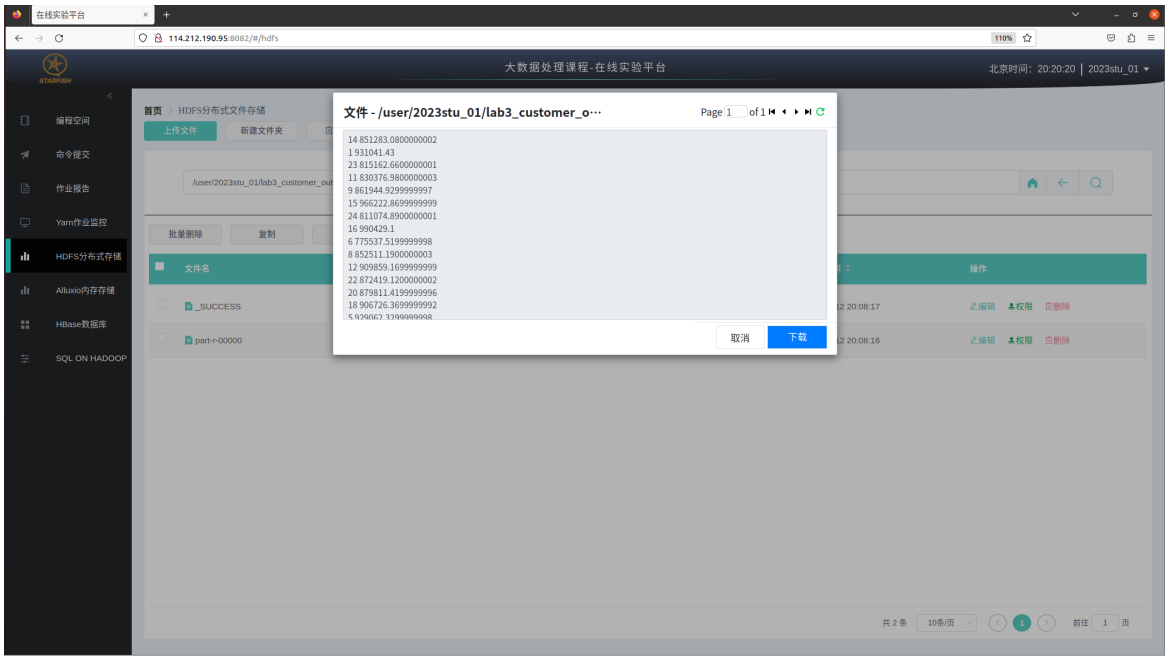
Map部分的伪代码：

```
class CustomerMap:
    function map(key, value, context):
        fileSplit = context.getInputSplit()
        fileName = fileSplit.getPath().getName()//获取当前输入记录所属的文件名
        itr = StringTokenizer(value.toString(), "\n")
        while itr.hasMoreTokens():
            line = itr.nextToken()//获取下一行
            valueInfo = Customer()//创建一个新的Customer对象
            valueInfo.setCustomer(line)
            keyInfo = Text()//空文本
            context.write(keyInfo, valueInfo)//输出键值对
```

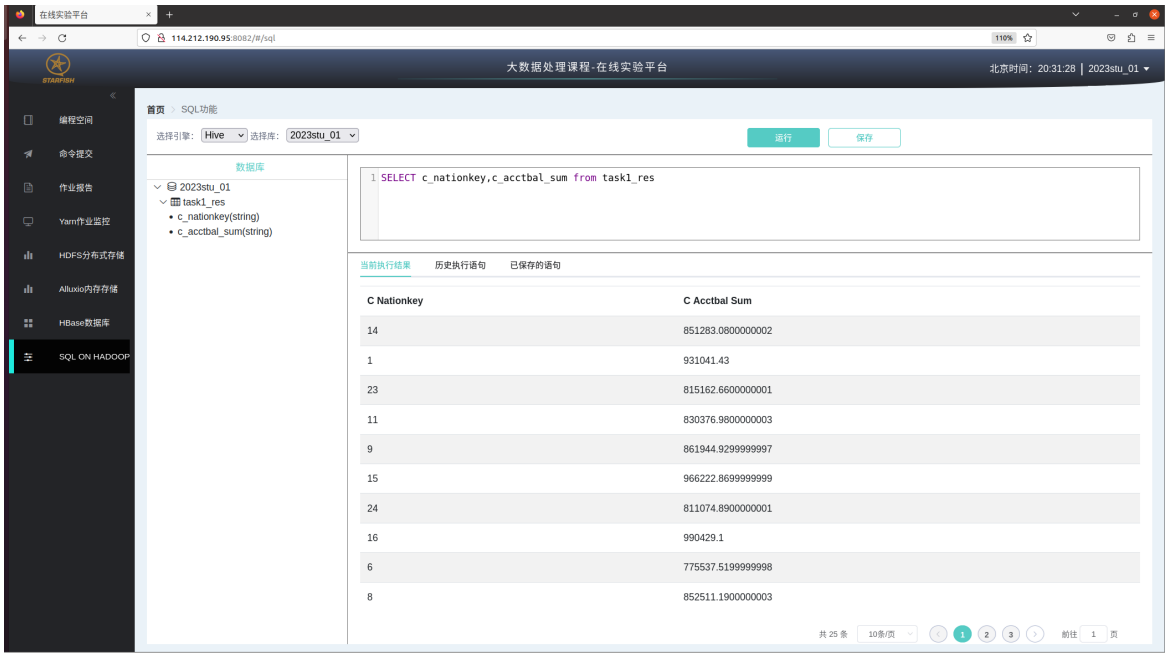
Reduce部分的伪代码：

```
function reduce(key, values):
    customersList  nc// 创建一个新的Vector容器
    for each value in values:
        c = nc//创建一个新的Customer对象，将value复制给c,将c添加到customersList容器中
    for i = 0 to customersList.size():
        if customersList[i].getVis() = 1://若等于 "1"，继续下一次循环
        sum = 0
        sum += customersList[i].getC_acctbal//将customersList[i].getC_acctbal转换为double类型
        customersList[i].setVis()
        for j = i + 1 to customersList.size():
            if customersList[j].getVis() = 1: //等于 "1"，继续下一次循环
            if customersList[j].getC_nationkey = customersList[i].getC_nationkey:
                sum += customersList[j].getC_acctbal
                customersList[j].setVis()
        // 构建键值对
        k = customersList[i].getC_nationkey + " " + String.valueOf(sum)
        //输出键值对 (k, NullWritable)
```

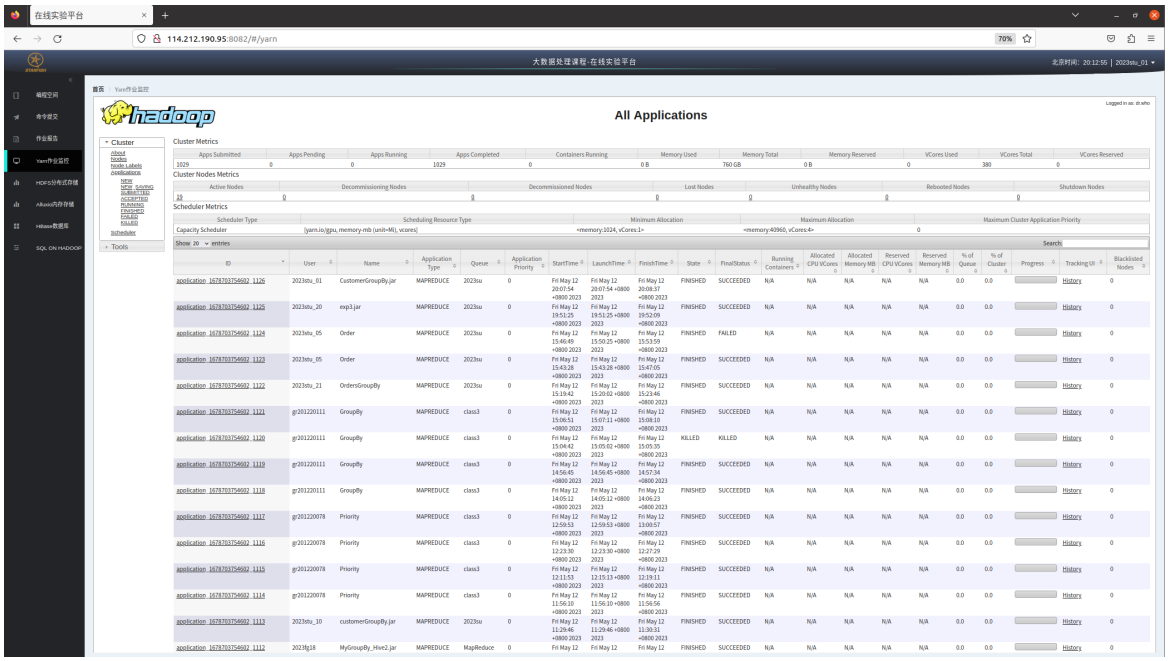
4.Hive输出结果文件的部分截图



输出结果文件在HDFS上路径：/user/2023stu_01/lab3_customer_output


在线实验平台
114.212.190.95:8082/#/sql
大数据处理课程-在线实验平台
北京时间：20:31:28 | 2023stu_01
编程空间
命令提交
作业报告
Yarn作业监控
HDFS分布式存储
Alluxio内存存储
HBase数据库
SQL ON HADOOP
首页 > SQL功能
选择引擎：Hive 选择库：2023stu_01
运行
保存
数据库
2023stu_01
task1_res
c_nationkey(string)
c_acctbal_sum(string)
1 SELECT c_nationkey,c_acctbal_sum from task1_res
当前执行结果
历史执行语句
已保存的语句
C Nationkey
C Acctbal Sum
14 851283.0800000002
1 931041.43
23 815162.6600000001
11 830376.9800000003
9 861944.9299999997
15 966222.8699999999
24 811074.8900000001
16 990429.1
6 775537.5199999998
8 852511.1900000003
共 25 条
10条/页
前往 1 页

5.WebUI执行报告


在线实验平台
114.212.190.95:8082/#/yarn
大数据处理课程-在线实验平台
All Applications
Cluster
Cluster Metrics
Cluster Nodes Metrics
Scheduler Metrics

任务二：订单信息表的Group By操作

1.实验目的

编写一个MapReduce程序，实现订单信息表的Group By操作，统计相同订单优先级的订单中订单发货优先级最高的订单信息，并输出订单的信息，包括订单标识符、优先级和最高发货优先级，即实现

```
select order_key, order_priority, max(ship_priority) from orders group by
order_priority;
```

输入数据为order.tbl，包含9个属性字段，以|分割，包含订单唯一标识符、下单客户唯一标识符、订单状态、订单总价、下单日期、订单优先级、处理订单的职员、订单发货优先级、订单注释。

输出数据将其上传至HDFS个人目录中，使用Hive建表管理保存订单唯一标识符、订单优先级和订单发货优先级。

2.Map和Reduce的设计思路

设计一个MapReduce程序，主要实现了对输入数据进行分组和筛选的功能。对订单数据进行分组，并筛选出每个订单优先级最高且发货优先级最高的订单。

Map函数（OrderMap类）：

在map方法中，主要目的是将每一行数据解析为一个Order对象，并将其作为值，与空键一起写入上下文。首先获取当前处理的输入文件名，然后，将输入的文本数据进行逐行解析。对于每一行数据，首先创建一个Order对象，然后将其字段值解析。将键值对(key, value)写入上下文。

Key是一个Text类型的变量，可能需要根据实际需求进行设置。Key并没有赋值，在每次写入时，键值对的键都将为空。

Reduce函数（OrderReducer类）：

在reduce方法中，目的是用于对订单数据进行分组，并筛选出每个订单优先级最高且发货优先级最高的订单。先创建一个Vector容器ordersList，用于存储具有相同键的Order对象。通过对输入的values进行迭代，将每个Order对象的副本添加到ordersList容器中。对于容器中的每个Order，首先检查其vis属性是否为"1"，如果是，则跳过当前对象的处理。如果vis属性为"0"，则将其设置为"1"，表示已访问。然后创建一个新的Vector容器orders，并将当前对象的副本添加到其中。对于容器中剩余的未访问对象，如果其order_priority与当前对象的order_priority相同，则将其vis属性设置为"1"。然后根据ship_priority与当前对象的ship_priority的大小关系，进行不同的处理：如果ship_priority小于当前对象的ship_priority，则清空orders容器，并更新ship_priority为当前对象的ship_priority，将当前对象的副本添加到orders容器中。如果ship_priority等于当前对象的ship_priority，则将当前对象的副本添加到orders容器中。最后，对于orders容器中的每个Order对象，构建键值对k，将键值对写入上下文。

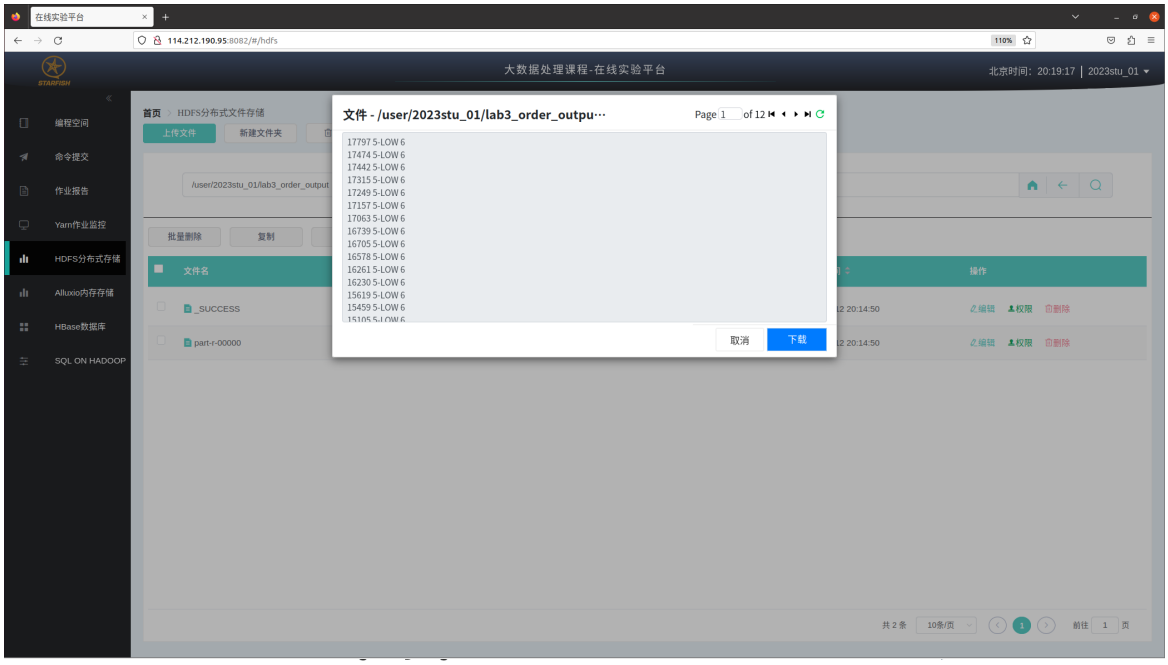
Key类型是Text，表示订单的分组标识。输入的键值对会按照key进行分组，相同的key会被分到同一个reduce任务中进行处理。

values的类型是Iterable<Order>，表示具有相同key的订单数据的迭代器。

3.Map和Reduce的伪代码

Map部分的伪代码：

```
class OrderMap:
    // 定义键值对的变量
    keyInfo = new Text()
    valueInfo = new Order()
    method map(key, value, context):
        fileName = context.getInputSplit().getPath().getName()// 获取当前处理的文件名
        lines = split(value, "\n")// 将输入的文本数据按行进行切分
        for line in lines:
            valueInfo.setOrder(line)// 将行数据设置到valueInfo对象中
            context.write(keyInfo, valueInfo)// 将key和value写入上下文
```

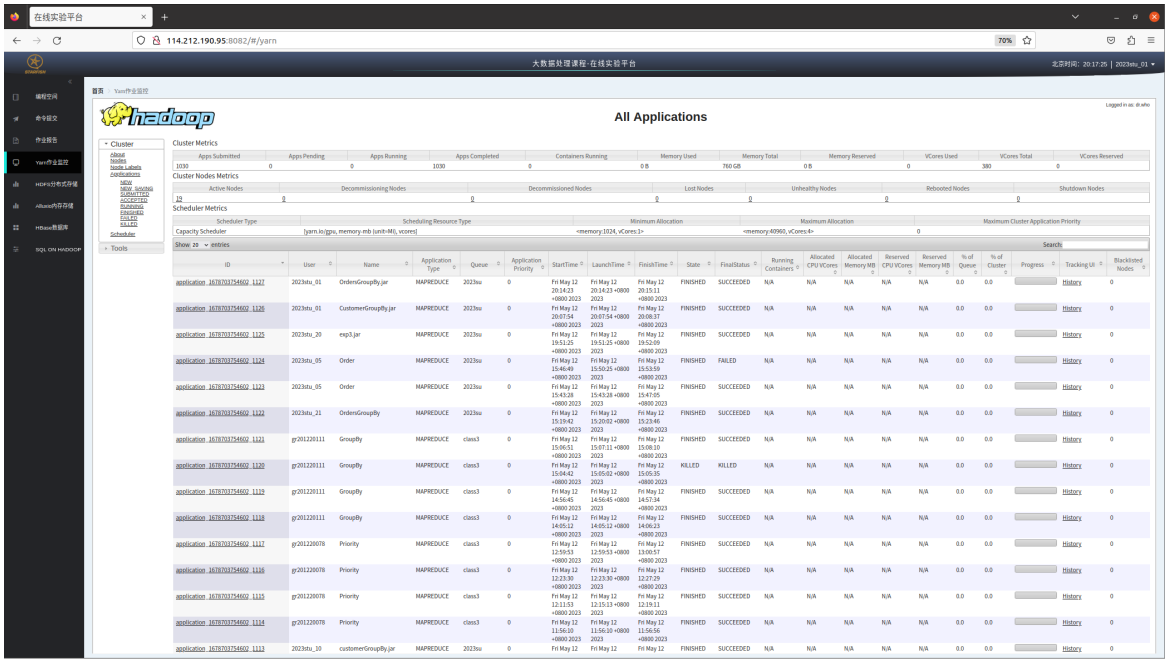


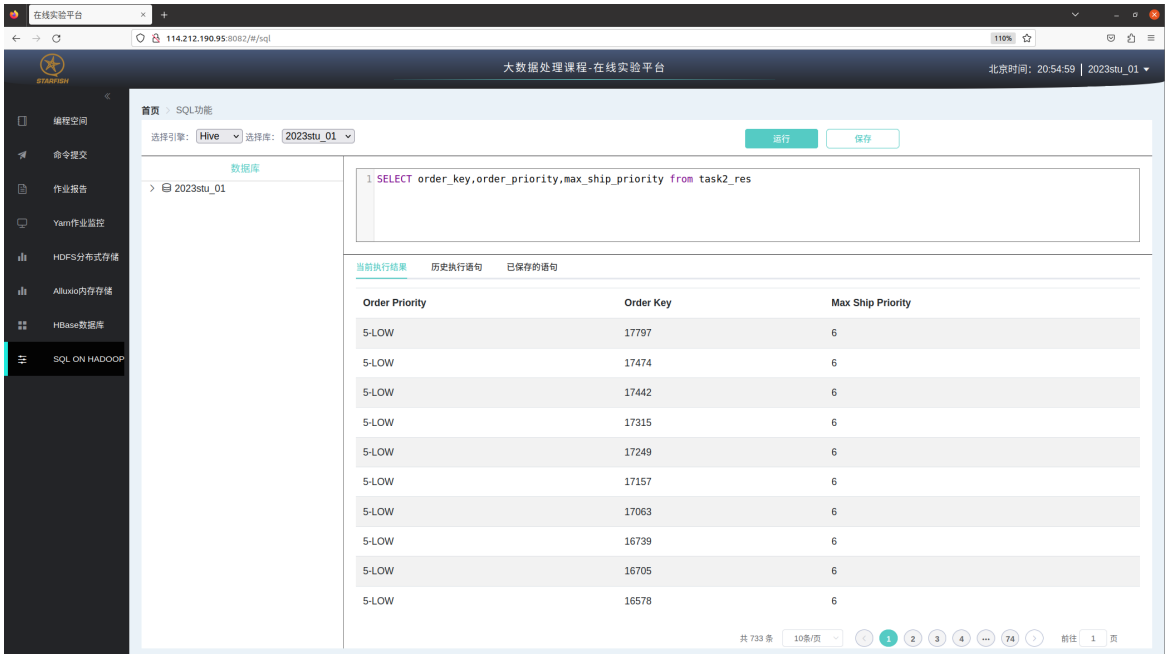

```
            continue
        // 如果当前订单的订单优先级与当前订单相同
        if ordersList.get(j).getOrder_priority() == ordersList.get(i).getOrder_priority():
            ordersList.get(j).setVis() // 将当前订单标记为已访问
            // 如果当前订单的发货优先级高于之前记录的发货优先级
            if Integer.valueOf(ordersList.get(j).getShip_priority()) > ship_priority:
                orders.clear()// 清空新订单列表，并更新发货优先级
                ship_priority = Integer.valueOf(ordersList.get(j).getShip_priority())
                temp = new Order(ordersList.get(j))// 将当前订单添加到新订单列表中
                orders.add(temp)
            // 如果当前订单的发货优先级与之前记录的发货优先级相同
            else if Integer.valueOf(ordersList.get(j).getShip_priority()) == ship_priority:
                temp = new Order(ordersList.get(j))// 将当前订单添加到新订单列表中
                orders.add(temp)
            // 如果当前订单的发货优先级低于之前记录的发货优先级，跳过
            else:
                continue

    // 遍历新订单列表中的订单
    for o in orders:
        k.set(o.getOrder_key() + " " + o.getOrder_priority() + " " + o.getShip_priority())
        context.write(k, NullWritable.get())// 输出键值对到上下文
```

Reduce部分的伪代码：

4.Hive输出结果文件的部分截图

输出结果文件在HDFS上路径：/user/2023stu_01/lab3_order_output


在线实验平台
114.212.190.95:8082/#/yarn
大数据处理课程-在线实验平台
All Applications
Cluster Metrics
Cluster Nodes Metrics
Scheduler Metrics


在线实验平台
114.212.190.95:8082/#/sql
大数据处理课程-在线实验平台
北京时间: 20:54:59 | 2023stu_01
首页 SQL功能
选择引擎: Hive 选择库: 2023stu_01
运行
保存
数据库
2023stu_01
SELECT order_key,order_priority,max_ship_priority from task2_res
当前执行结果
历史执行语句
已保存的语句
Order Priority
Order Key
Max Ship Priority
5-LOW 17797 6
5-LOW 17474 6
5-LOW 17442 6
5-LOW 17315 6
5-LOW 17249 6
5-LOW 17157 6
5-LOW 17063 6
5-LOW 16739 6
5-LOW 16705 6
5-LOW 16578 6
共 733 条
10条/页
前往 1 页

5.WebUI执行报告