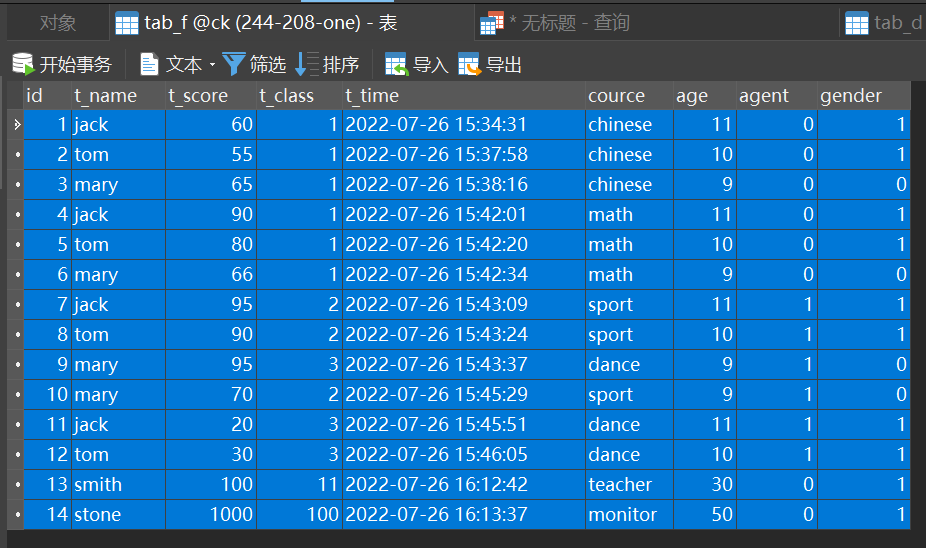
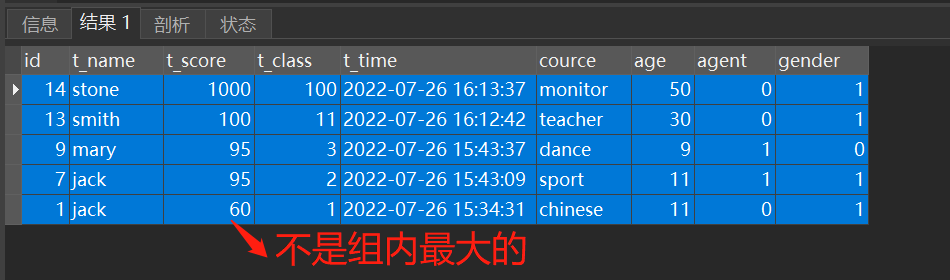
这一页文档里展示的，是初学者自以为是、其实错误的做法！正确做法看下一页文档：

id是自增、主键，其他是普通字段。

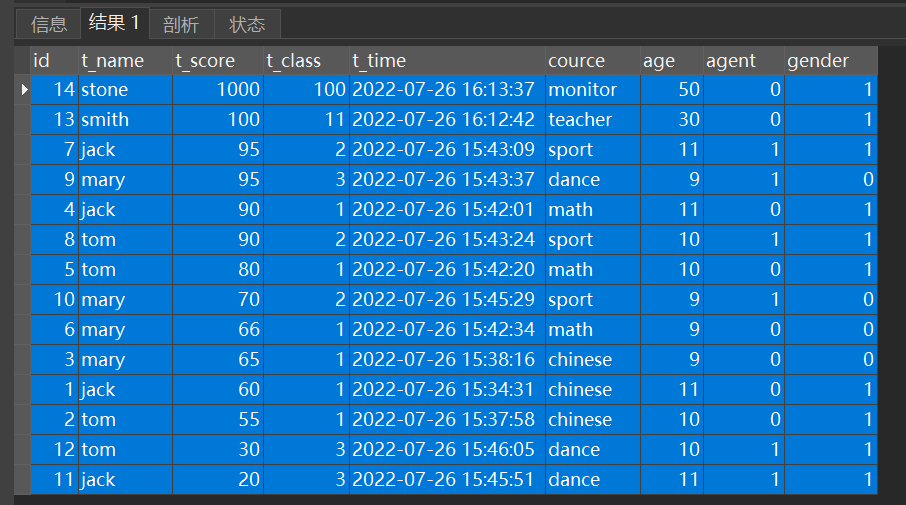


SELECT \* FROM tab\_f GROUP BY t\_class ORDER BY t\_score DESC;

（每组让GROUP BY自动只取一条，取到的也不是组内排序的第一条。**不符合预期**）



SELECT \* FROM tab\_f GROUP BY t\_class,id ORDER BY t\_score DESC; （**不符合预期**）

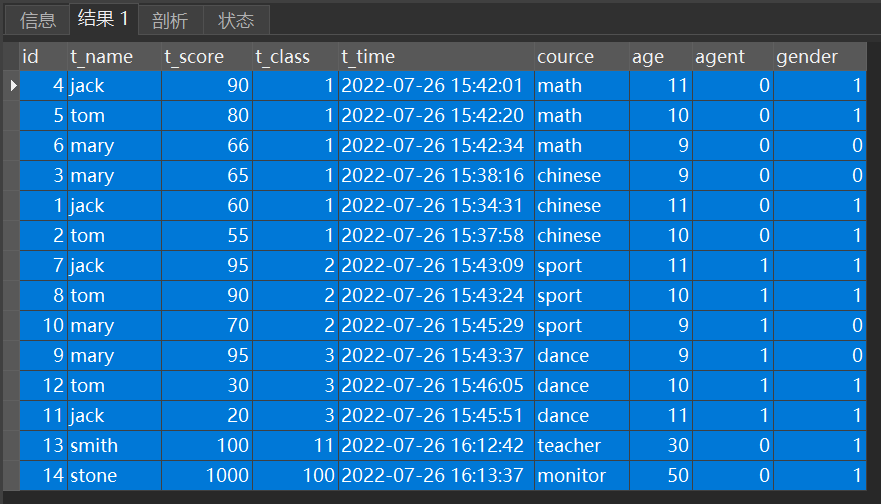


============================= 一 组内排序 =====================================

想要对组内排序，可采用的一种方法是，对组内再分组，同时对这些再分组进行组间排序，从而达到从最外层看做到了组内排序的效果。如下：

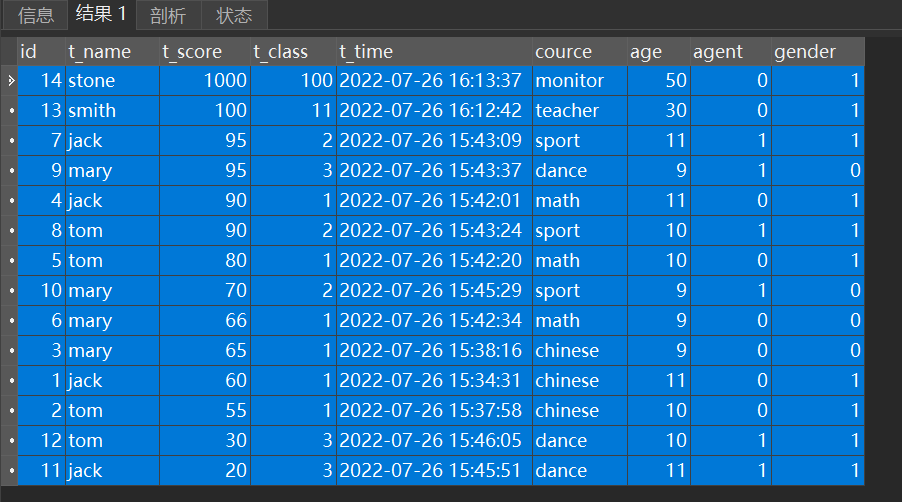
SELECT \* FROM tab\_f GROUP BY t\_class, t\_score, id ORDER BY t\_class, t\_score DESC; （符合预期）

（ORDER BY后面的t\_class是默认升序，t\_score后面指明了DESC则是降序）



*注意：上述sql中不能少了ORDER BY t\_class,否则无法指明ORDER\_BY后面的t\_score去对里层的分组进行组间排序，否则情况如下：*

*SELECT \* FROM tab\_f GROUP BY t\_class, t\_score,id ORDER BY t\_score DESC; （不符合预期）*



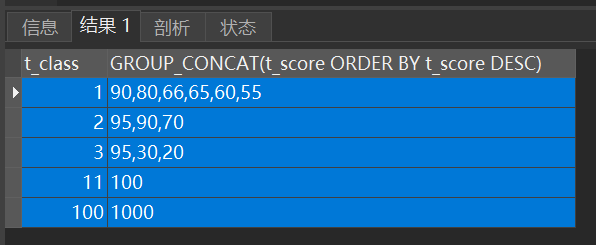
============================= 二 组内排序 =====================================

另一种方式是GROUP\_CONCAT：

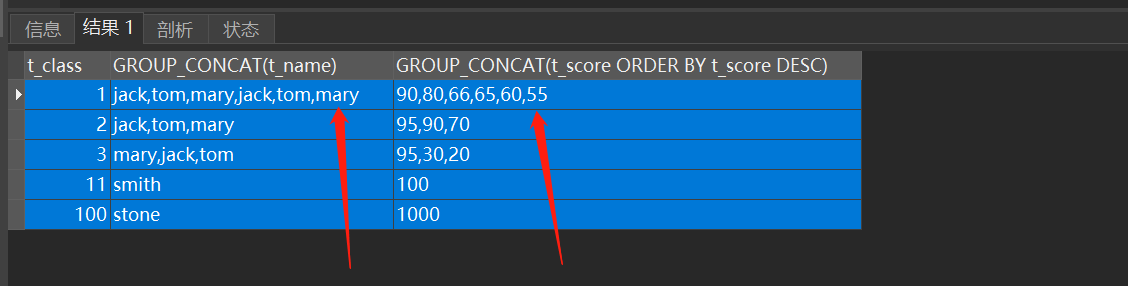
（看下面，也可参考https://www.cnblogs.com/qi-yuan-008/p/12788213.html ）

（看下面，或可参考https://www.cnblogs.com/guo10432/p/13620739.html ）

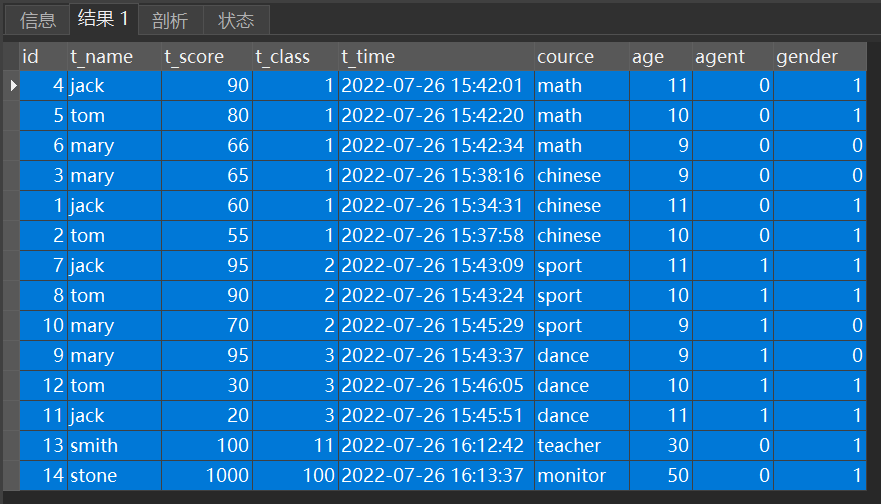
SELECT t\_class,GROUP\_CONCAT(t\_score ORDER BY t\_score DESC) FROM tab\_f GROUP BY t\_class;（符合预期）



SELECT t\_class,GROUP\_CONCAT(t\_name),GROUP\_CONCAT(t\_score ORDER BY t\_score DESC) FROM tab\_f GROUP BY t\_class;（此时不符合预期，两个GROUP\_CONCAT中的数据列表无法做到逐一对应）

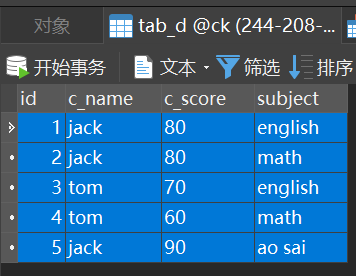


看是否符合预期，可以看看前文第一种方式得到的正确结果：



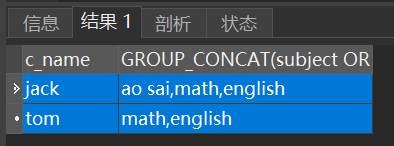
再举一例 分组-GROUP\_CONCAT：

id是自增、主键，其他是一般字段。



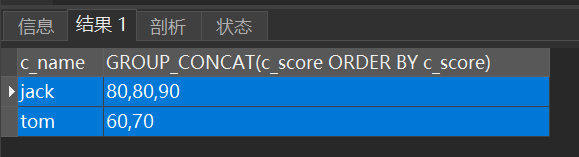
SELECT c\_name, GROUP\_CONCAT(subject ORDER BY id DESC) FROM tab\_d GROUP BY c\_name;

（GROUP\_CONCAT中的ORDER BY 是对内部subject的排序）



SELECT c\_name,GROUP\_CONCAT(c\_score ORDER BY c\_score) FROM tab\_d GROUP BY c\_name;

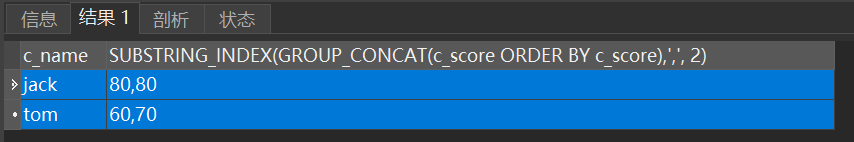
（GROUP\_CONCAT中的ORDER BY 是对内部c\_score的排序）



【GROUP\_CONCAT组内的截取问题】

【调用者主动截取】

SELECT c\_name, SUBSTRING\_INDEX(GROUP\_CONCAT(c\_score ORDER BY c\_score),',', 2) FROM tab\_d GROUP BY c\_name;（结果符合预期）



其中的MySQL函数—— SUBSTRING\_INDEX(str,delim,count)，用于获取子字符串

-- str:待处理字符串  
-- delim:分隔符  
-- count:计数值，若count是正数，则从左往右数，取第count个分隔符左边的字符串  
若count是负数，则从右往左数，第count个分隔符右边的字符串

）

-substring\_index可以嵌套，

如substring\_index(substring\_index(“颐、和、园”,`、`,**2**),`、`,-1)

注意：用了group\_concat后，select里如果使用了limit是对组间起作用，不会对group\_concat起作用，也不要想着在sql语句中的group\_concat的括号里使用limit。

【MySQL的默认截取】

用group\_concat连接字段的时候是有长度限制的，并不是有多少连多少，一旦查询的数据超过MySQL默认的1024，就会自动截取，但可以更改相应的系统变量group\_concat\_max\_len的值：

1、在MySQL的配置文件中加上 group\_concat\_max\_len=102400 //数字可自定义

2、也可以简单一点，在执行sql语句前

SET GLOBAL group\_concat\_max\_len=102400;后直接跟包含group\_concat 的sql语句即可。（这种是全局的）  
SET SESSION group\_concat\_max\_len=102400;后直接跟包含group\_concat 的sql语句即可。（这种是当前和MySQL的会话的）  
若设置了最大长度，则结果被截至这个最大长度。

【用关键词SEPARATOR修改group\_concat的默认分隔符】

select GROUP\_CONCAT(f\_a SEPARATOR '\_') from t\_one GROUP BY f\_b;

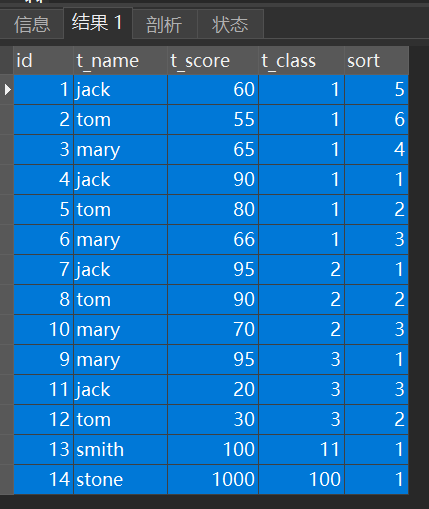
select GROUP\_CONCAT(f\_a ORDER BY id SEPARATOR '\_') from t\_one GROUP BY f\_b;

============================= 三 组内排序 =====================================

第三种方式是表的自相JOIN：（参考https://www.cnblogs.com/qi-yuan-008/p/12788213.html ）

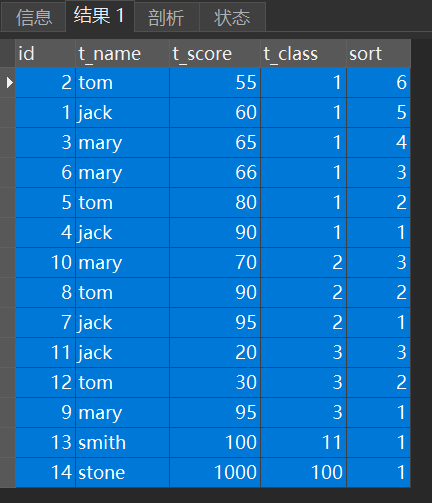
通过JOIN后COUNT(\*) AS sort获得排序依据。{{{但这种方式因联表会过多占用内存}}}

SELECT tab1.id,tab1.t\_name, tab1.t\_score, tab1.t\_class, COUNT(\*) AS sort FROM tab\_f tab1 JOIN tab\_f tab2 ON tab1.t\_class = tab2.t\_class AND tab1.t\_score <= tab2.t\_score GROUP BY tab1.t\_class, tab1.id; （符合预期）【GROUP\_BY的后面若无ORDER\_BY，组间默认升序】

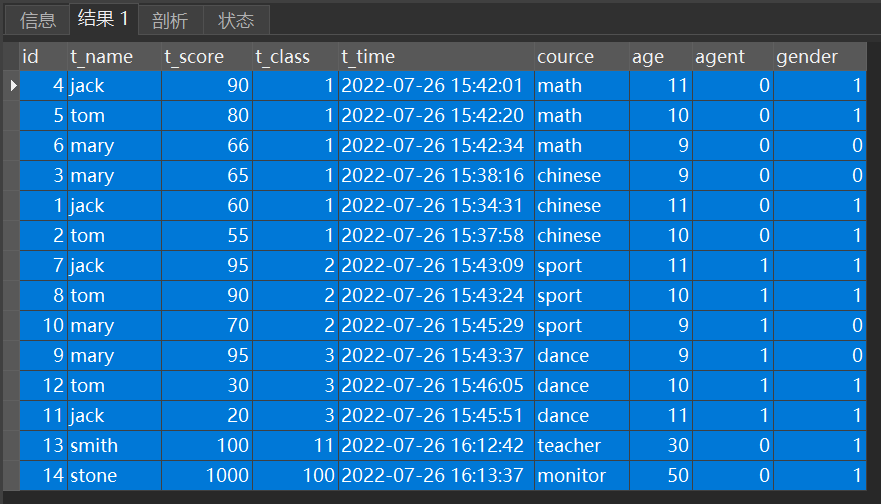


但如果是

SELECT ANY\_VALUE(tab1.id) ,tab1.t\_name, tab1.t\_score, tab1.t\_class, COUNT(\*) AS sort FROM tab\_f tab1 JOIN tab\_f tab2 ON tab1.t\_class = tab2.t\_class AND tab1.t\_score <= tab2.t\_score GROUP BY tab1.t\_class, tab1.t\_score; （符合预期）【GROUP\_BY的后面若无ORDER\_BY，组间会默认升序】



看是否符合预期，可以看看前文第一种方式得到的正确结果：



============================= 四 组内排序 =====================================

MySQL8.0 有了窗口函数,就可以使用序号函数 rank() , over是关键字,用来指定函数执行的窗口范围，括号里指定用partition by 进分组,order by 进行排序

select score,class,student\_name RANK() OVER (partition by class order by score desc) from student\_score

窗口函数的另一个示例：

select dept\_no,name,salary,ROW\_NUMBER() OVER(partition by dept\_no order by salary desc) as sort from employee

ROW\_NUMBER() OVER(partition by dept\_no order by salary desc)  
表示根据dept\_no分组，在分组内部根据 salary排序，而此函数计算的值就表示每组内部排序后的顺序编号（组内连续的唯一的)

============================= 五 组内排序 =====================================

更多可参考

https://wenku.baidu.com/view/edb330aacf22bcd126fff705cc17552707225e2f?aggId=9170f13f13a6f524ccbff121dd36a32d7375c793

(完)