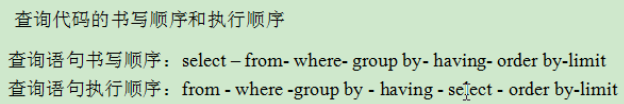
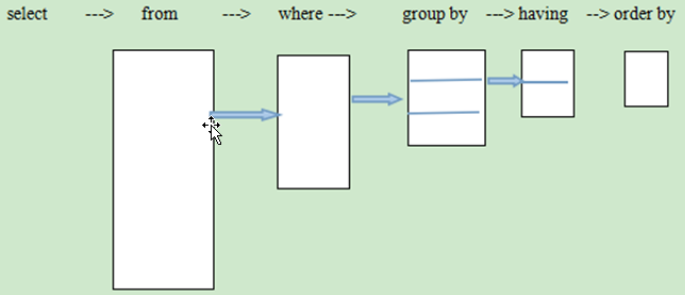
MySQL的排序查询与分组查询

# 查询语句的书写顺序与执行顺序





# 排序查询： order by 和asc/desc

## 功能：对select查询出的数据记录按照某个或多个字段排序。

默认情况下，是升序查询，即asc。**asc: 正序，顺序； desc：反序，倒序**

## 语法1： order by 字段名 [asc/desc] 默认为asc

**注意：**如果有where选择数据，**order by**一定要放在where的后面。

## 语法2：可以支持多条件排序（根据多字段进行排序）

**ORDER BY field [asc/desc],field2 [asc/desc],…;**

升序asc 是默认的，可以不写。

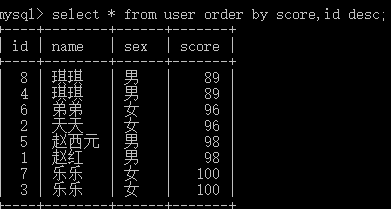
首先根据**第一个field进行**排序，如果**出现具有相同的**，再根据**第二个字段**进行排序，否则，不再进行判断，依次类推。

### select \* from ds order by currenttime desc,money2,name desc;

先按照currenttime降序排序，currenttime相同的再按照money2升序排序，currenttime和money2都相同的，再按照name排序。

### select \* from users order by PROVINCEID asc; 按照provinceid升序排序

### select \* from user order by score [asc],id desc; # 先按照成绩升序（asc可以省略），再按照id降序排序



# 分组查询：group by

## 功能：GROUP BY关键字可以将select的查询结果按照某个字段或多个字段进行分组。

## 分组依据：字段中值相等的为一组，组内可以根据第二个字段再分成小组。

在MySQL中，还可以按照**多个字段**进行分组。分组过程中，先按照第一个指定字段进行分组，遇到该字段的值相等的情况时，再把该字段相等的记录按照**另外一个字段**进行分组。

## 语法：GROUP BY field1 [,field2,...] [WITH ROLLUP] [HAVING 条件表达式]

### field:分组的依据，从左向右依次分类；

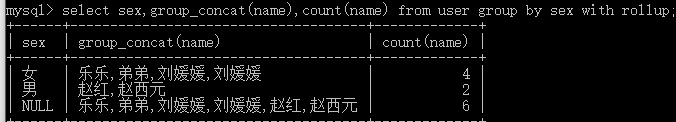
### HAVING 条件表达式：用来限制分组后的显示，符合条件表达式的结果将被显示。

**注意：having后面跟的条件表达式中的字段必须是前面分组的依据多用的字段。**

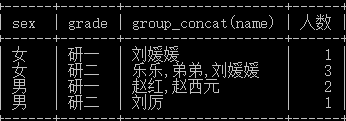
### WITH ROLLUP：将会在所有记录的最后加上一条记录。加上的这一条记录是上面所有记录的总和。

**with rollup 必须紧挨着group by ，在group by 之后，在having 之前。放在having之后是错误的。**

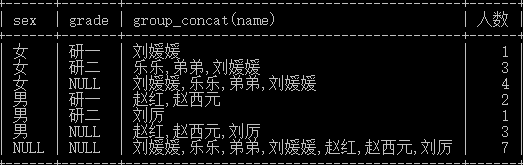
select sex,group\_concat(name),count(name) from user **group by sex** **with rollup**;



select sex,grade,group\_concat(name),count(name) 人数 from user **group by sex,grade**;

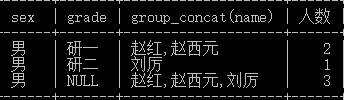


select sex,grade,group\_concat(name),count(name) 人数 from user group by sex,grade **with rollup**;



注意：**每一次分组都会在最后添加一行**。

select sex,grade,group\_concat(name),count(name) 人数 from user group by sex,grade **with rollup having** sex = '男';



## group by 与集合函数一起使用

### group by 与group\_concat()函数一起使用（注意不是concat()函数，concat()函数与group\_concat()函数联合使用，格式化输出）

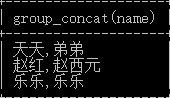
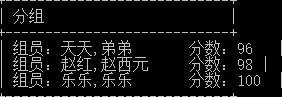
**GROUP\_CONCAT()函数**会把每个分组内部指定的字段值用逗号组合在一起并显示出来；而**concat()函数**是字符与字段值的组合拼接。

如果某字段是**分组的依据字段**，则是唯一的；但是如果输出的字段不是分组依据字段，则其具有不同的值，但是硬要输出来 ，则只可以输出该分组的第一个数据的该字段值，这一般是没有多大意义的。所以，一般单独输出的字段是**分组依据字段**，其他字段利用**GROUP\_CONCAT（field）**函数输出。

如：select **group\_concat(name)** from user group by score having score > 90;

select **concat('组员：',group\_concat(name),'\t分数：',score)** as '分组' from user group by score having score > 90;

select concat(name) from user group by score having score > 90;#没有任何效果

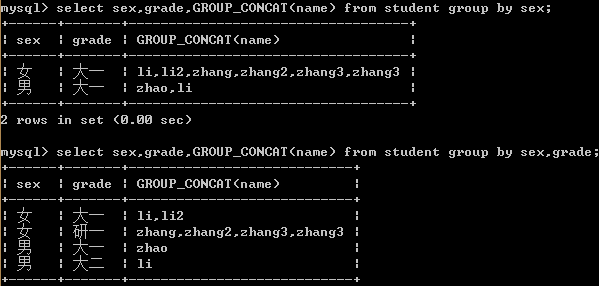
  

**GROUP BY关键字**单独使用时，**只能查询出每个分组的一条记录，这样做的意义不大**。因此，一般在使用**集合函数时**才使用GROUP BY关键字。

A. select sex,grade,**GROUP\_CONCAT**(name) from student **group by** sex;

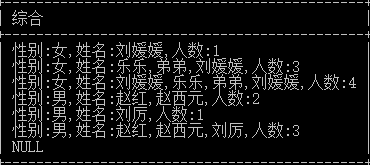
B. select sex,grade**,GROUP\_CONCAT**(name) from student **group by** sex,grade;

首先根据sex分组，然后在分好的各个组内，再根据grade进行分组

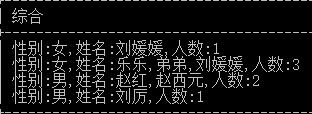


### concat()函数与group\_concat()函数的联合使用

select **concat**('性别:',sex,',姓名:',**group\_concat(name)**,',人数:',count(name)) as 综合 from user group by sex,grade with rollup;



select concat('性别:',sex,',姓名:',**group\_concat(name)**,',人数:',count(name)) as 综合 from user group by sex,grade;



### GROUP BY关键字通常与聚合函数一起使用。聚合函数包括COUNT()函数、SUM()函数、AVG()函数、MAX()函数和MIN()函数等。

A. **COUNT()函数**：用于统计记录的条数。

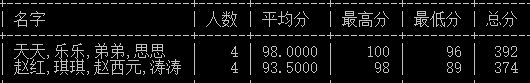
B. **SUM()函数**：用于计算字段的值的总和。

C. **AVG()函数**：用于计算字段的值的平均值。

D. **MAX()函数**：用于查询字段的最大值。

E. **MIN()函数**：用于查询字段的最小值。

select **group\_concat(name) '名字'**,count(name) '人数',avg(score) '平均分',max(score) '最高分',min(score) '最低分',sum(score) '总分' from user group by sex; #按照性别分成两组，计算人数、平均分、最高分、最低分、总分。

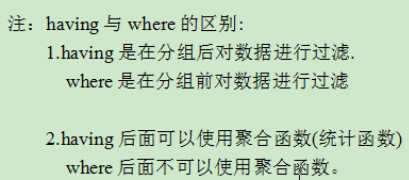


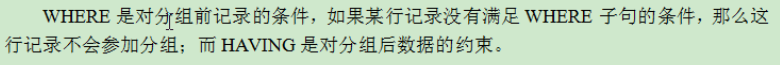
## group by与having 条件的联合使用

使用GROUP BY关键字时，如果加上“HAVING 条件表达式”，则可以限制输出哪些组。只有符合条件表达式的组才会显示。

### HAVING表达式与WHERE表达式的区别

“HAVING 条件表达式”与“WHERE 条件表达式”都是用于限制显示的。但是，两者起作用的地方不一样。**WHERE 条件表达式**：作用于表或者视图，是表和视图的查询条件。**HAVING 条件表达式**：作用于分组后的记录，用于选择符合条件的组。**简单说，having用于筛选组（分组后的筛选），后接条件可以用聚合函数，where用于筛选数据记录（分组前的筛选），后接条件不可以使用聚合函数。where 语句要在group by之前，而having 要在group by之后。where筛选出的数据再进行分组，然后利用having筛选出特定的组。**





select group\_concat(name) '名字',sum(score) '总分' from user group by sex **having sum(score)>340**; #having 后可以使用聚合函数，而where后面不可以。

