MySQL的distinct去重与limit限制查询数目

# distinct去重

## distinct的作用：去除重复的字段；

## distinct的使用语法：

**distinct 后面可以直接跟\*，也可以跟一个字段或多个字段。**

### distinct \* ：只有所有字段都一样的才算重复。

### distinct field1:去除field1字段中重复的。

### distinct field1，field2 ：只有field1、field2都一样才算是重复。

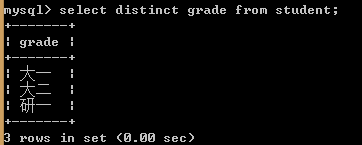
### distinct field1，field2，field3…… ：只有field1、field2、field3、…都一样才算是重复。

## 示例

### select distinct \* from student; 所有字段都相同时才会去除。

### select distinct grade from student; 去除grade字段相同的值。



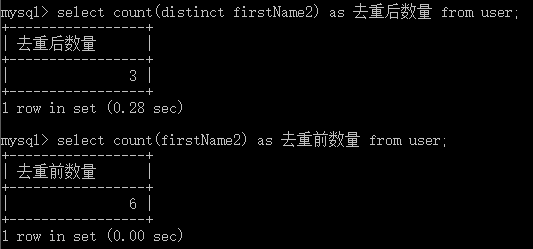
### select distinct sex,grade from student; 两个字段都相同的才算重复，才会去除。



### 去重后的数量按照剩下的不重复的数量

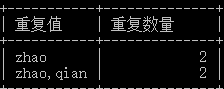
select count(distinct fristName2) as 去重后数量 from user;

select count(firstName2) as 去重前数量 from user;



### 如果需要统计每个重复值对应的记录数目，利用distinct无法完成。需要利用group by分组功能。

select firstName2 重复值,count(firstName2) 重复数量 from user group by firstName2 having firstName2 like 'zhao%';



# limit关键字：强制select语句返回指定的记录数。

## limit的功能：

limit 子句用于强制 SELECT 语句返回指定的记录数。

## limit的语法：

**limit [offset,]** rows **或者 limit** rows **offset** offset

大写（**LIMIT [offset,] rows 或者 LIMIT** rows **OFFSET** offset）

**LIMIT 接受一个或两个数字参数**。参数必须是**整数常量**。如果给定两个参数，第一个参数指定第一个返回记录行的偏移量offset，从0开始；第二个参数指定返回记录行的最大数目。初始记录行的偏移量是 0(而不是 1)。

为了与 PostgreSQL 兼容，MySQL 也支持句法： LIMIT # OFFSET #。

### offset指定要返回的第一行的偏移量，可以不指定，默认为0。

**即limit 0，num; 等价于 limit num; 查询前面的num个记录**

**MySQL不支持：limit offset,-1;表示从偏移后至结束。**

### rows第二个指定返回行的最大数目。

rows表示的是记录个数（即行数），不是结束的行标。

### 初始行的偏移量是0(不是1)。

查询第一个：**select \* from users limit 0,1；**

**select \* from users limit 1；**

**查询第二个：select \* from users limit 1,1；**

## 示例：

### 查出前4条记录

select \* from user limit 0,4;

select \* from user limit 4;

select \* from user limit 4 offset 0;

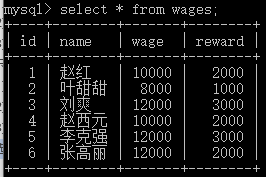
### 查出第4-7条记录

select \* from user limit 3,4;

select \* from user limit 4 offset 3;

## 选择第二名、第三名等：利用排序+limit获取

### 数据库表格



### 目标分析

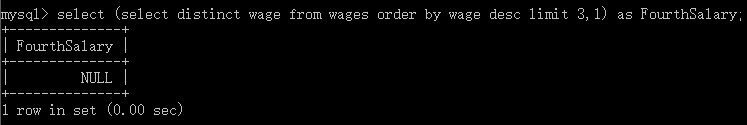
**获取最大值或最小值，直接利用max()或min()函数就可以，但是获取工资排名为第二名、第三名...或倒数（升序即可）第二名、第三名...的工资数目，需要嵌套查询。且有可能不在第二名、第三名等的情况，若不存在则返回NULL**。

注意：**去重，不存在要返回NULL**，起个别名。

**对于NULL处理说明**：不存在的，mysql返回的是空集合即**Empty set**，所以需要人为指定为**NULL**。

**方法：再在外面嵌套一个select语句，即可返回NULL**，还可以再利用**IFNULL函数更加明确的指明更好**。

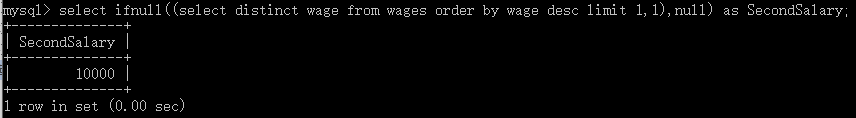




### SQL语句

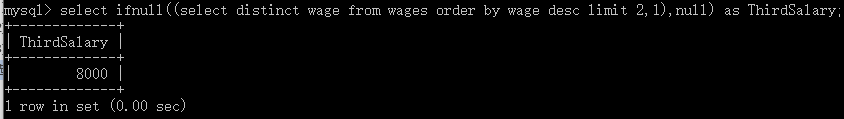
* 1. **返回第二Salary值**

**select ifnull((select distinct wage from wages order by wage desc limit 1,1),null) as SecondSalary;**



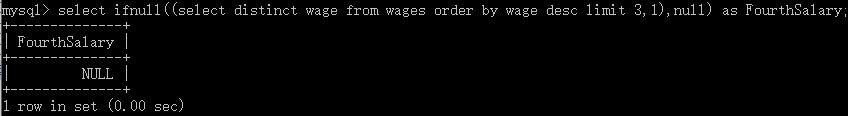
* 1. **返回第三Salary值**

**select ifnull((select distinct wage from wages order by wage desc limit 2,1),null) as ThirdSalary;**

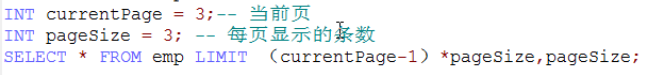


* 1. **返回第四Salary值**

**select ifnull((select distinct wage from wages order by wage desc limit 3,1),null) as FourthSalary;**



## MySQL的分页查询语句及性能分析



### 最基本的分页方式（中小数据量足够）

注意：最好使用索引。举例来说，如果实际SQL类似下面语句，那么在category\_id, id两列上建立复合索引比较好：

**SELECT** \* **FROM** articles **WHERE** category\_id = 123 **ORDER** **BY** id LIMIT 50, 10

### 子查询的分页方式

随着数据量的增加，页数会越来越多，查看后几页的SQL就可能类似：

**SELECT** \* **FROM** articles **WHERE** category\_id = 123 **ORDER** **BY** id LIMIT 10000, 10;

一言以蔽之，就是越往后分页，LIMIT语句的偏移量就会越大，速度也会明显变慢。

此时，我们可以通过子查询的方式来提高分页效率，大致如下：

**SELECT** \* **FROM** articles **WHERE**  id >= (**SELECT** id **FROM** articles  **WHERE** category\_id = 123 **ORDER** **BY** id LIMIT 10000, 1) LIMIT 10;

### JOIN分页方式