

数值类型

整形 5个 tinyint smallint mediumint,int,bigint

unsigned.

浮点型

float

double

decimal

type(M,D) M是总共的宽度 ,D表示 小数的宽度

日期类型

datetime

timestamp 时间戳,范围最大到2038年,它是一个从1970年1月1日到现在的一个秒数.

date

time 时间 还表示间隔

year

字符串类型

char

varchar type(M)

set 多选 1,2,4,8,16,…

enum 单选 1,2,3,4…

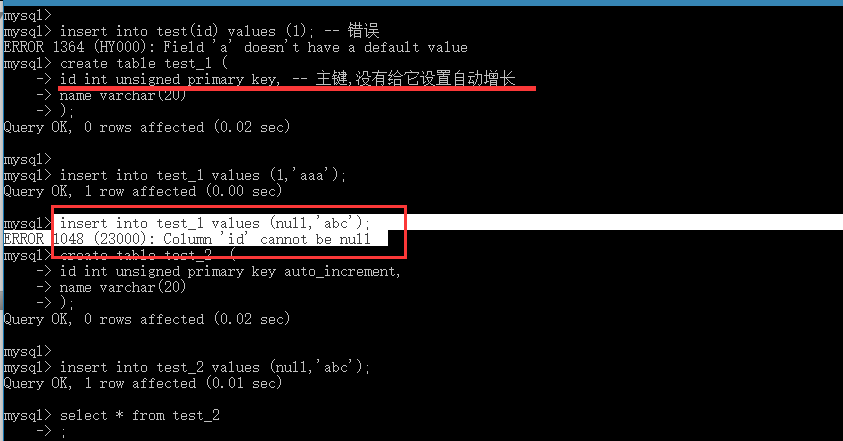
列属性

null 和 not null not null是列属性的数据,再插入数据的时候必须得有值

default

primary key 一张表最多有一个 primary key.

primary key不能为null



primary key 一般情况下我们都习惯于 单独定义一个 id,作为我们的主键.

我们也可以 定义多个字段为我们的主键, primary key(字段列表).

多个字段的值都相等的时候,才冲突

组合主键 (名字-身高)

auto\_increment 可以设置默认值,必须为整形,必须是一个索引

unique key 唯一的. 可以有多个字段为unique key,

字段的值可以为 null

实体与实体的关系

1:1

1:n 学生表 开设班级

n:m 学生表 老师表 思维定式: 将多对多的情况转换成多对1 或者是 1对多的情况

学生表

姓名 性别 年龄 老师

张三 男 20 刘将\齐瑞华\

老师表

姓名 性别 年龄 学生

齐瑞华 男 30 张三\李四\王五

老师字段 学生字段

齐瑞华 张三

齐瑞华 李四

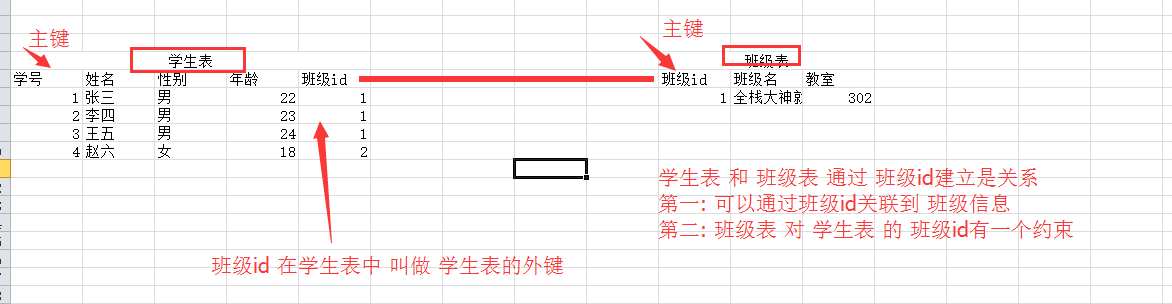
齐瑞华 王五

刘将 张三

刘将 李四

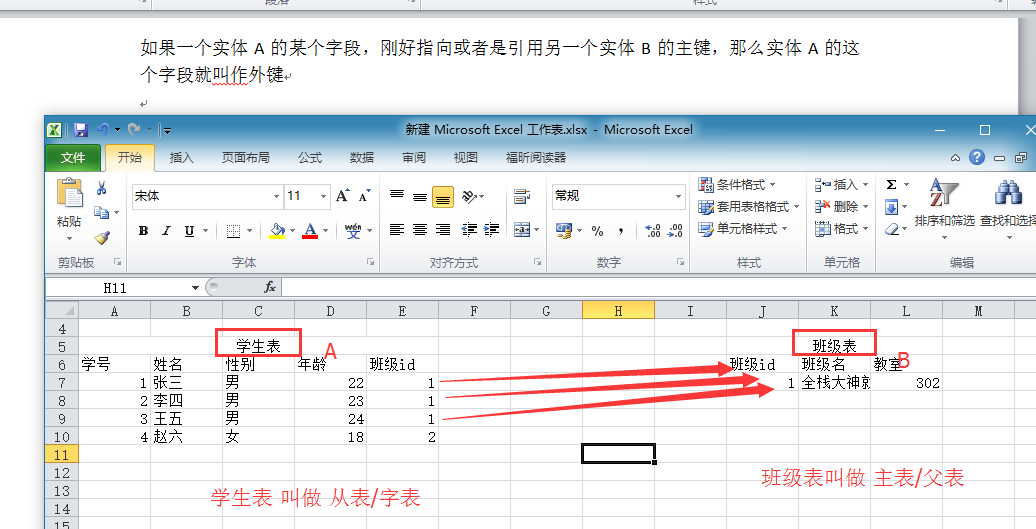
刘将 王五

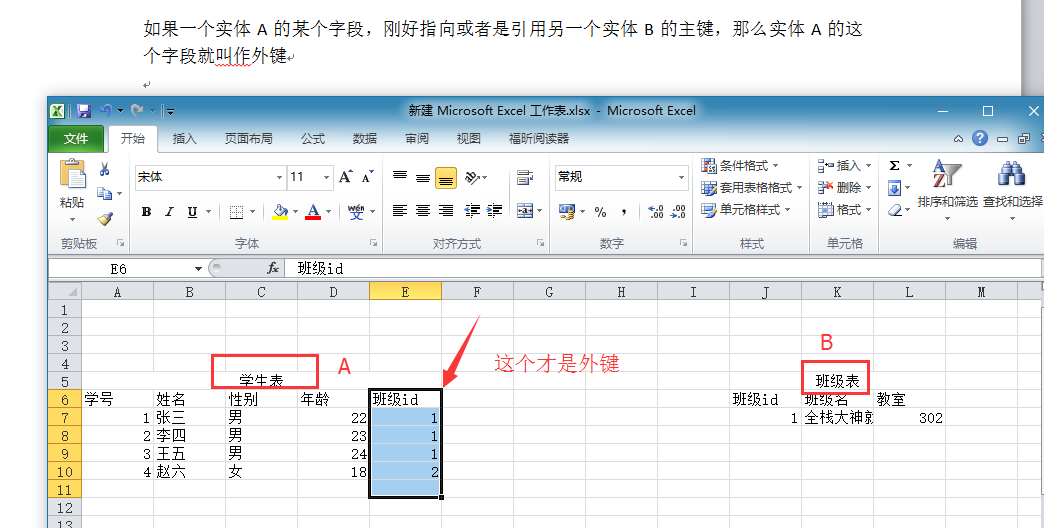
# 第15讲 外键



## 概念

如果一个实体A的某个字段，刚好指向或者是引用另一个实体B的主键，那么实体A的这个字段就叫作外键



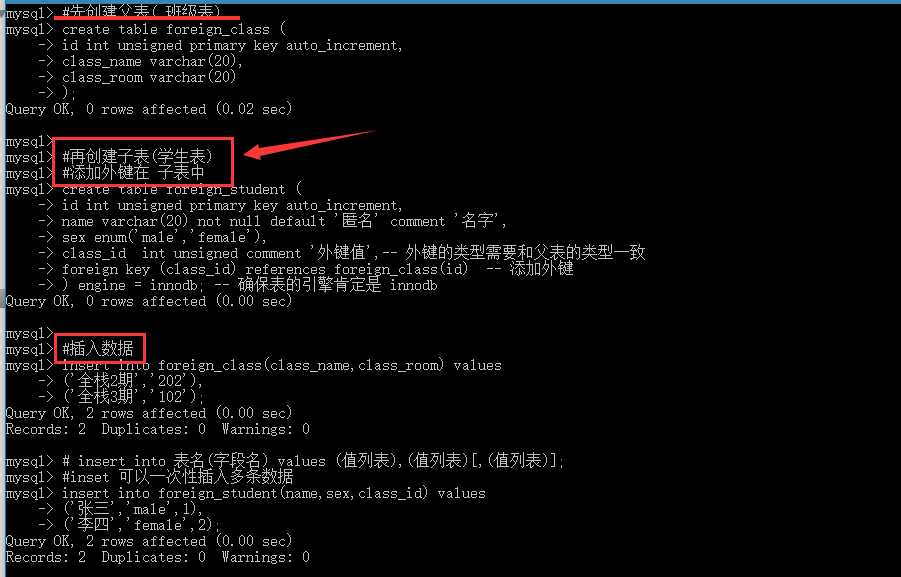


## 语法

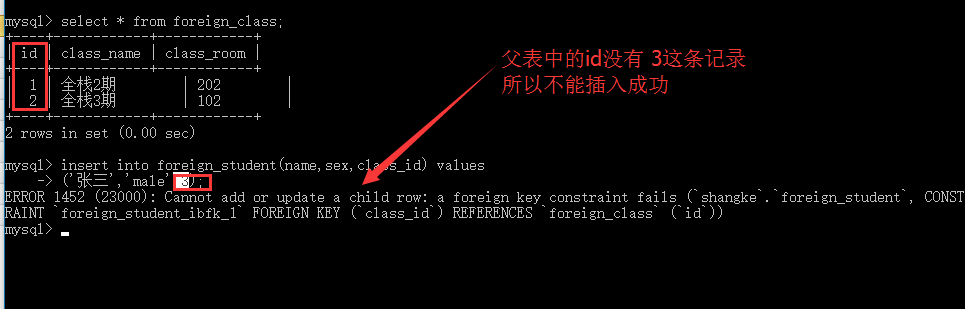
这一句话应该加在 A中

foreign key (子表的字段) references 父表(父表的主键) [选项];

初始工作:



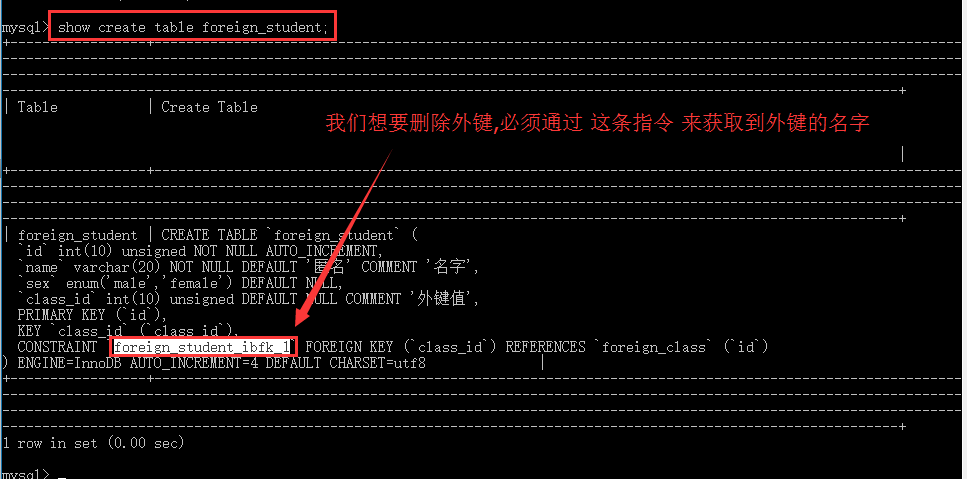
当我们插入一个 父表不存在的id的时候,是插入不进去的



## 删除外键和添加外键

### 删除外键

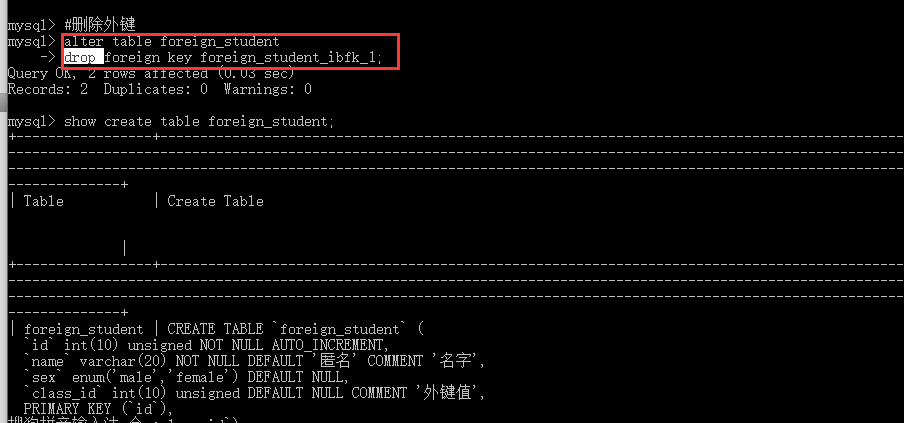
第一步: 通过 show create table 指令来获取外键的名字



第二步: 通过 alter table 表名 + drop foreign key 外键名 来删除外键

alter table 表名

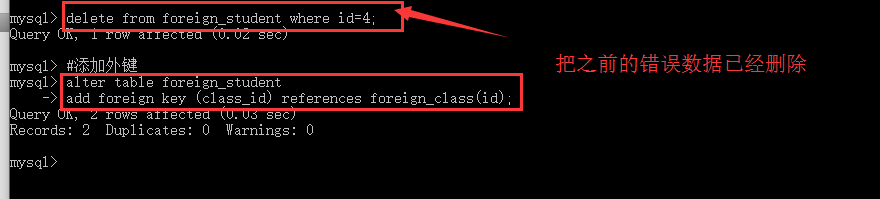
drop foreign key 外键名;



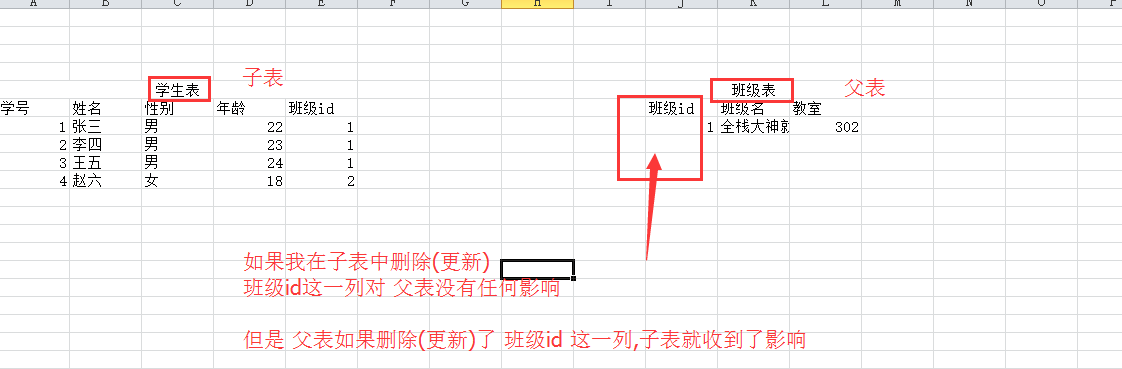
### 添加外键

alter table 表名

add foreign key(子表字段名) references 父表名(父表字段) [选项];



## 级联操作



### 主表更新

foreign key(子表字段) references 父表(父表字段) on update [选项];

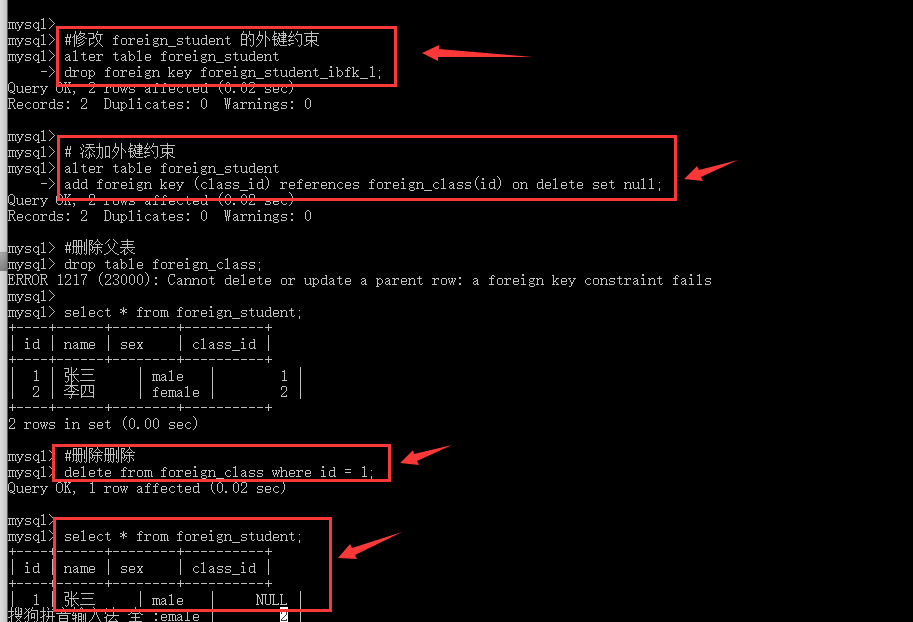
### 主表删除

foreign key(子表字段) references 父表(父表字段) on delete [选项];

### 级联操作

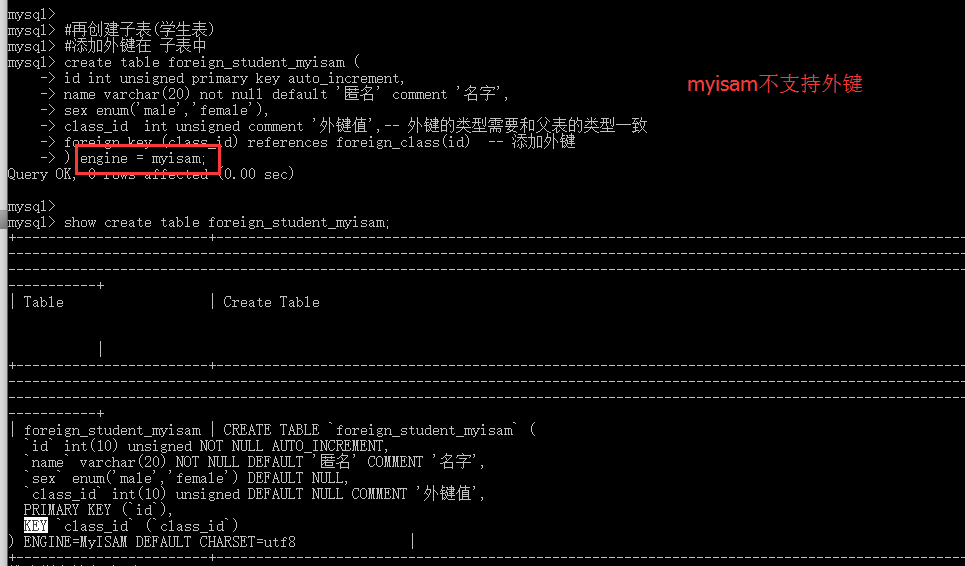
级联操作一共有三种形式：

* **cascade：**同步操作，或者串联操作，也就是当父表的主键字段更新或者删除的时候，子表的外键字段也进行相应的更新或者删除！
* **set null：**设置为null，也就是当父表的主键字段更新或者删除的时候，子表的外键字段就设置为null，当然，前提是子表中的外键字段没有非空约束！
* **restrict：**就是拒绝主表更新或者删除！



## 注意事项

* 外键约束只有InnoDB存储引擎才支持



# 第16讲 数据操作详细

## 插入数据

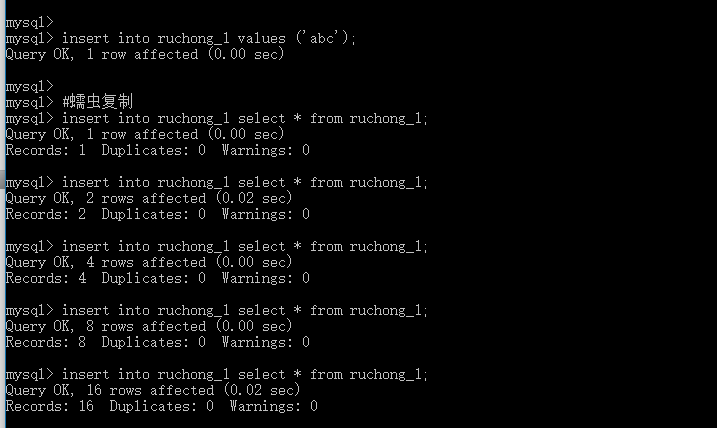
标准语法:

insert into 表名(字段列表) values (值列表)[,(值列表)];

### 蠕(ru)虫复制

* 语法规则

insert into 表名(字段列表) select (字段列表) from 表名;



* 作用

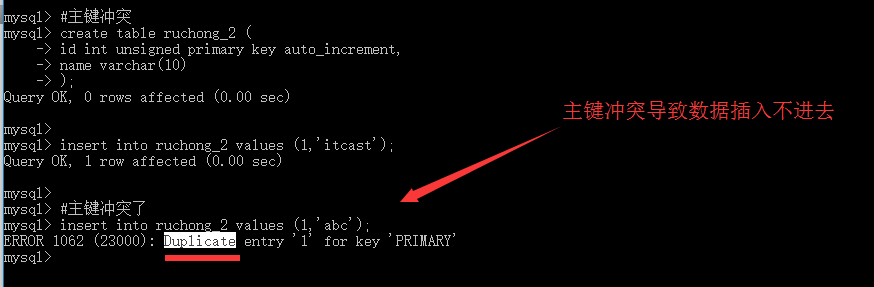
快速的生成测试数据

测试一下数据库的效率

注意事项:

insert的字段列表 需要和 select的字段列表 一一对应.

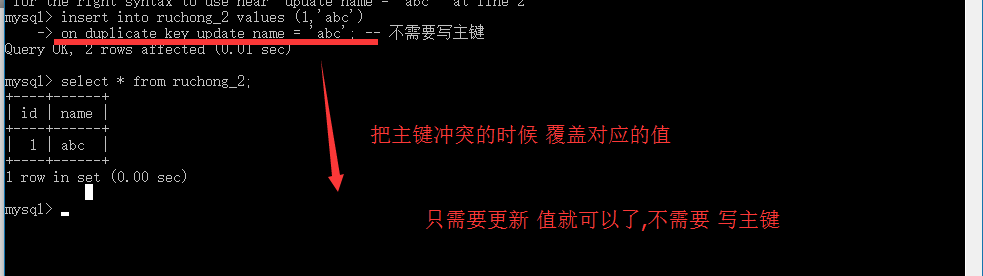
### 主键重复



* 直接更新

insert into 表名(字段列表) values (值列表)

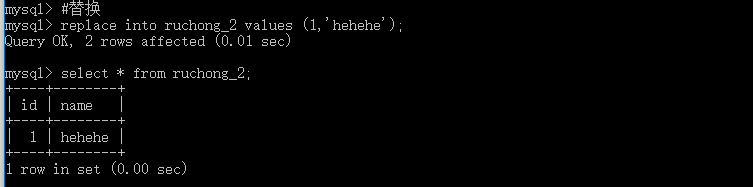
on duplicate key update 字段=值[,字段=值];

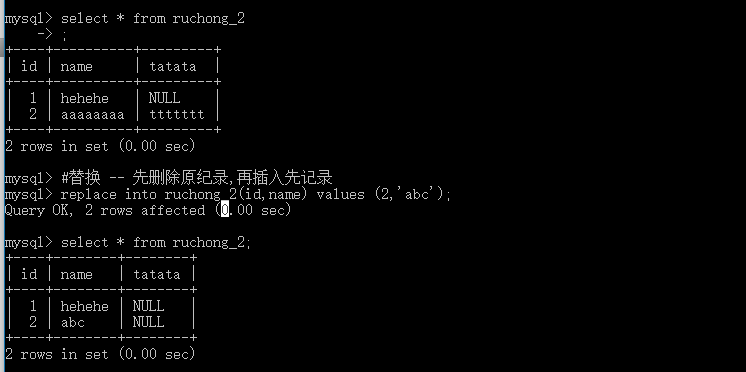


* 直接删除原纪录再插入

replace into 表名(字段列表) values (值列表);

replace的原理就是 先删除原纪录,再插入





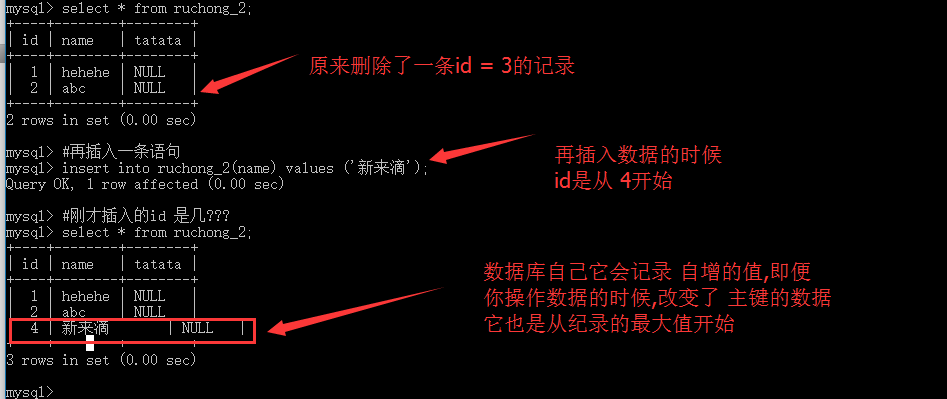
## 删除数据

### 删除满足条件

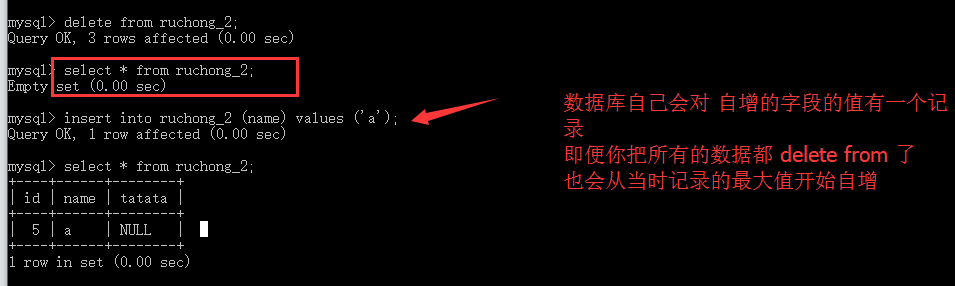
delete from 表名 [条件判断];

### truncate清空

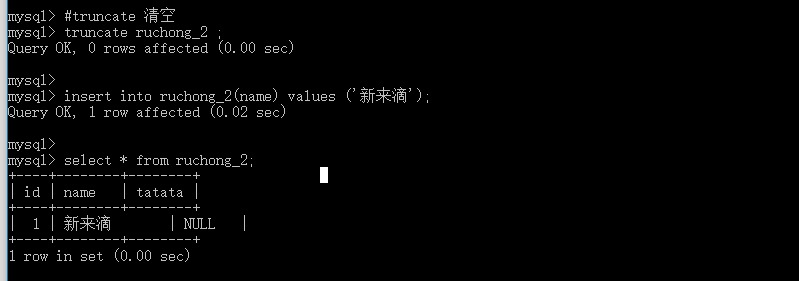
主键的id只能 自动增长



如果我们把所有的数据都删了, 主键 还是从 最大的那个开始



* 只能清空操作，不能使用where判断。
* 清空数据表自增长字段置1.



### 删除语句比较

* drop 删除数据表结构。同时数据全部消失。
* delete 逐行删除记录。通常添加限定条件where。删除全部行，自增长字段不置1.
* truncate 清空数据表数据。自增长字段置1.不能添加where判断。

## 查询数据

select \*|字段列表 from 表名

where 子句

group by 子句

having 子句

order by 子句

limit 子句;

五子查询的顺序必须是按照以上的顺序来!!!

# 第17讲 select选项和别名

## select选项 all/distinct

select all|distinct \* from 表名;

all: 表示所有,默认是 all

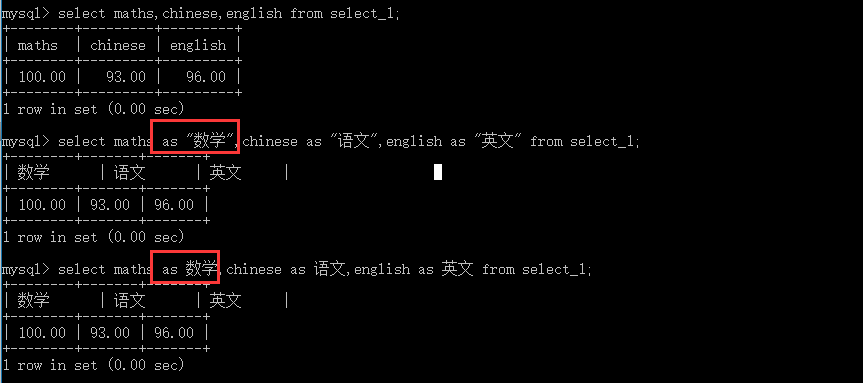
distinct: 去重(去掉重复)



## select别名

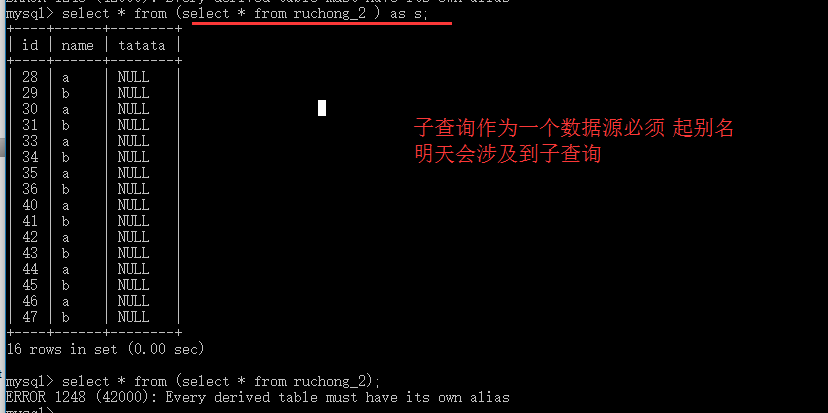
### 别名

就是给字段或者其他表达式起一个名字



### 别名的作用:

* 如果出现多表查询的时候，往往不同的表会有一样的字段名，如果使用相同的字段名，不好区分
* 常都要给表达式起一个别名，来代表该表达式的含义
* 在真实的项目中，表名往往都比较长,可以简化
* 当数据源是一个子查询的时候，必须给该子查询语句起一个别名



# 第18讲 where子句

## 语法

select \*|字段列表 from 表名 where 条件判断;

## 功能

通过限定条件对数据进行筛选，得到想要的数据结果集！

## 流程

逐一从数据表中取出每一条记录，先通过当前的记录来计算where后面表达式的值，如果计算的结果为真（非0），就返回该记录，如果计算的结果为假（0），就不返回该记录，相当于对该表的所有的记录做了一次遍历！

## MySql运算符

### 关系运算符

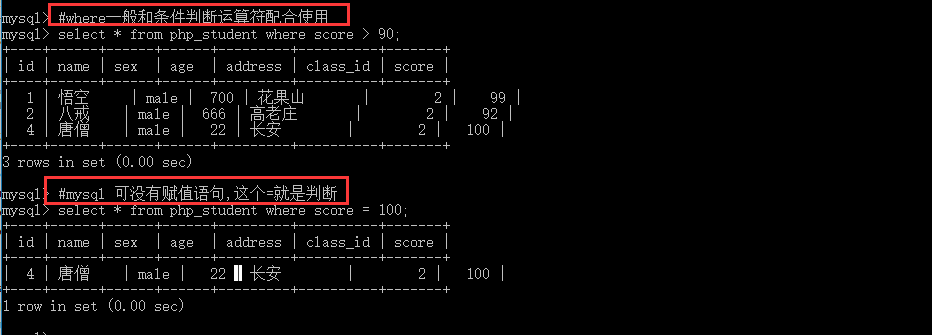
<

>

<=

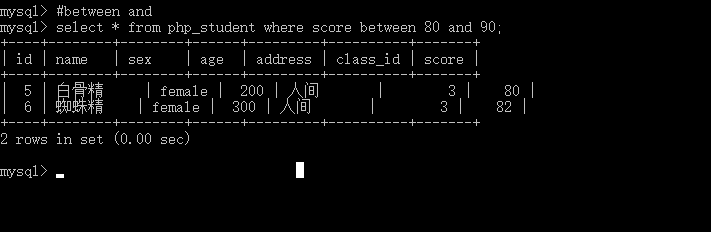
>=

= (mysql中的= 是进行判断)

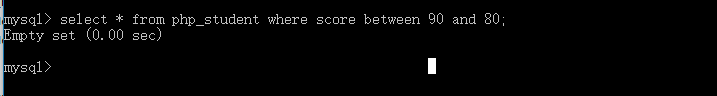


### between and

between A andB 意思就是在 A和B之间 , A和B都是闭区间(A,和B都是在范围之内的)



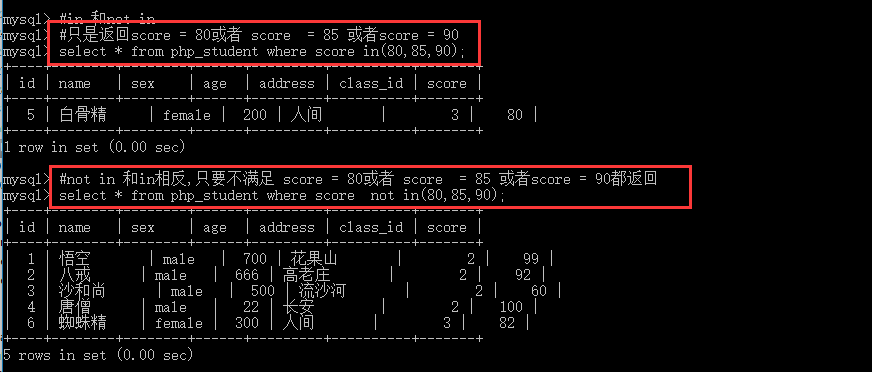
一定要注意 A<=B



### in和not in

in(值的集合)

举例来说明: in(2,4,8) .就是只满足2或者4或者8

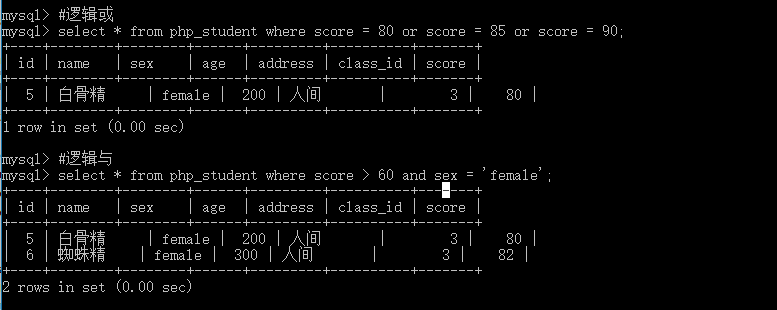


### 逻辑运算符

&& and

|| or

! not



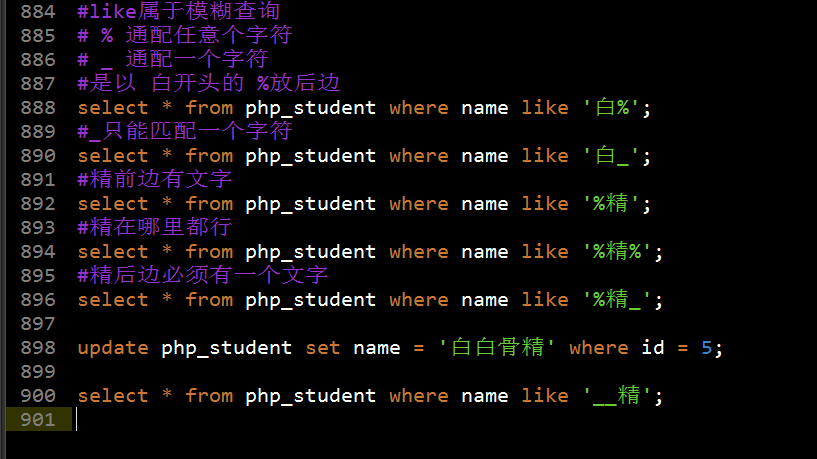
### like

like模糊查询

like 后边必须跟字符串 以 like ‘’ 的形式

通配符：

* % ：使用任意多个字符进行匹配
* \_:使用单个字符进行匹配



# 第19讲 group by子句

select \*|字段列表 from 表名

where 子句

group by子句

having子句

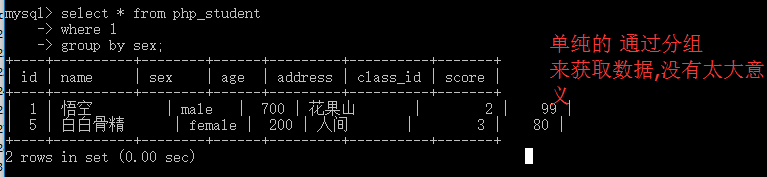
order by 子句

limit 子句;

按照字段进行分组，然后统计数据。分组便于统计数据。

## 语法

group by 字段

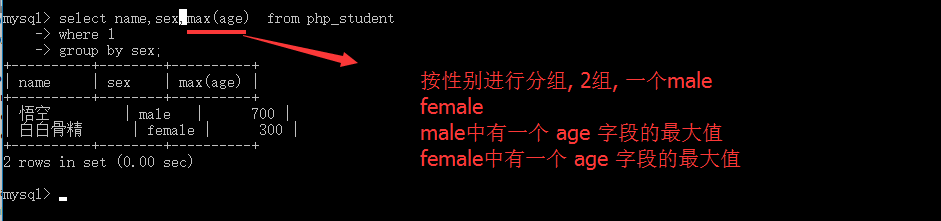


## 聚合函数

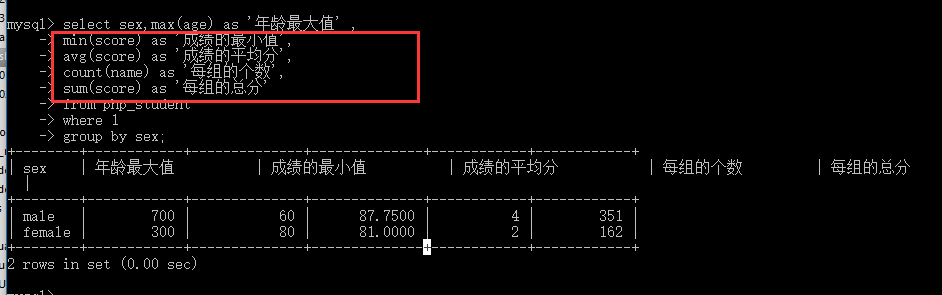
SQL中聚合函数对一组值执行计算，并返回单个值。聚合函数都会忽略空值count(colm),但是count（\*）为记录总行数。聚合函数经常与 SELECT 语句的 GROUP BY 子句一起使用。

* avg()，返回指定组中的平均值，空值被忽略。
* count()，返回指定组中项目的数量；
* max()，返回指定组中数据最大值；
* min()，返回指定组中数据最小值；
* sum()，返回指定组中数据的和

聚合函数中小括号填写的字段,是什么,就去获取相应的字段的 对应的值



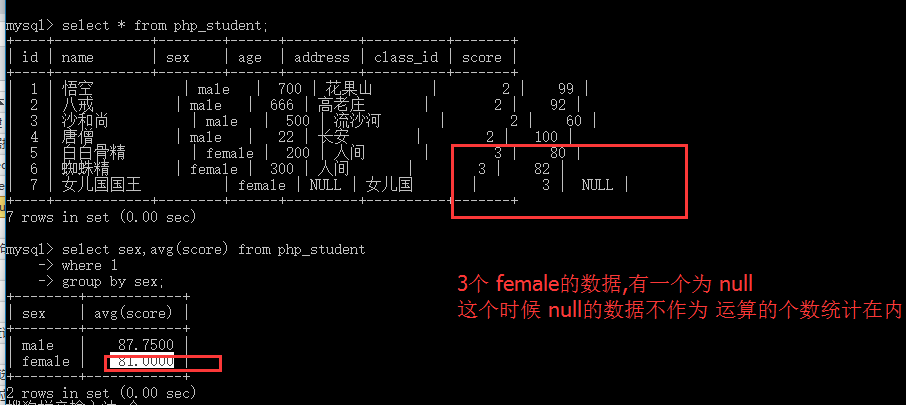
五个聚合的函数



关于数据的值中有null的情况

avg 和count 涉及到 null

avg: null值是忽略的



count 函数对null也是不统计的



如果分组默认的数据就是一个大的组

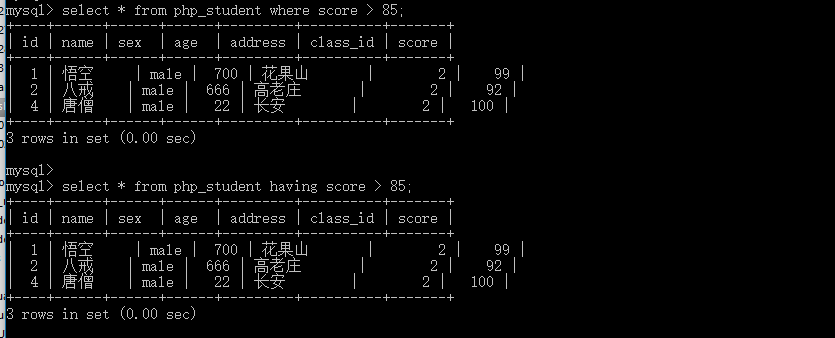
# 第20讲 having子句

having的含义可以理解成为和where一样.

## 分组之后再进行筛选，只能使用having子句

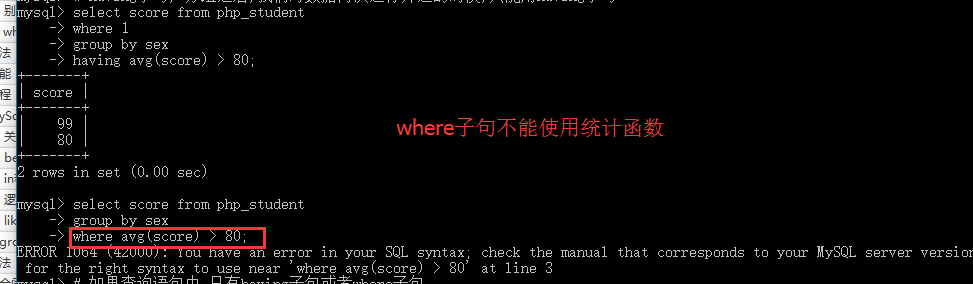
## having子句与where子句的比较:

### 如果语句中只有having或者where子句的话,它们的作用基本相同



### 二者本质区别是: where子句把磁盘上的数据筛选到内存.having子句是把内存中的数据再次进行筛选.

### Where子句不能使用统计函数.having子句可以.



# 第21讲 order by子句

order 排序

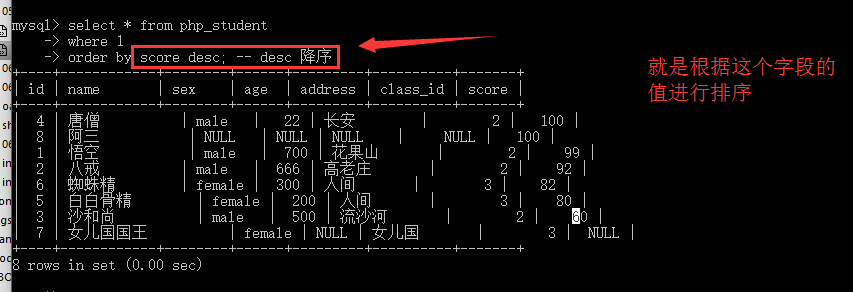
## 语法

oder by 字段 [desc|asc];

desc: 降序, 就是数值有大到小

asc: 升序,就是由小到到的排序

就是按照字段来进行排序



## 多字段排序

order by 字段1 [desc|asc], 字段2 [desc|asc],…

先按照字段1进行排序,如果字段1的值相同,我们再按照字段2的进行排序



# 第22讲 limit子句

limit 限制的意思

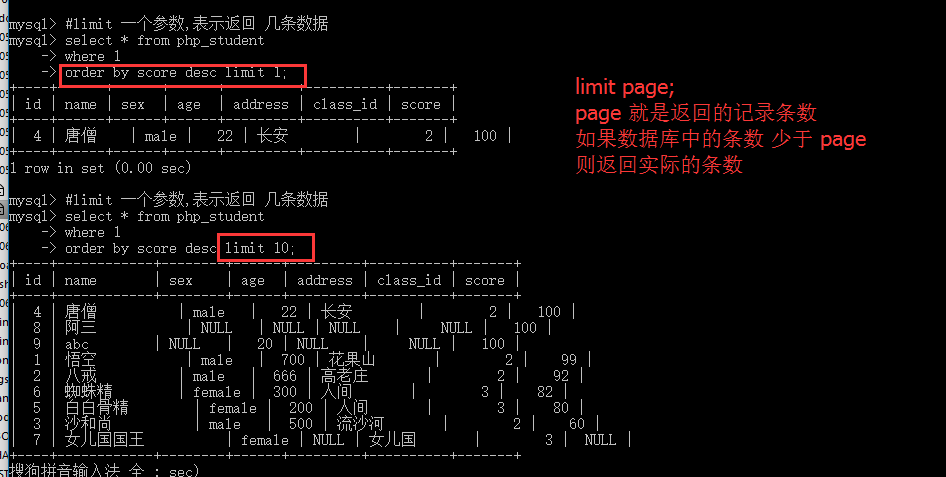
## 语法

limit有两个语法

第一个语法: 一个参数

limit page;

page的含义就是返回多少条记录



第二个语法就是 2个参数:这个是使用最多的

limit start,numofperpage;

start: 表示从第几行开始

numofperpage: 表示 一页显示多少条记录

## 分页原理

limit 0,5; 表示 从记录的第1个开始,获取5条记录

limit 5,5; 表示 从记录的第6个开始,获取5条记录

limit 10,5;表示 从记录的第11个开始,获取5条记录.

limit 15,5; 表示 从记录的第16个开始,获取5条记录

我们会把 每页返回的条数,和对应的页面作为一个参数进行传递

$numofperpage = 5; 我们传递的一个参数

$page = 3

limit ($page -1)\*$numofperpage ,numofperpage; -- 想要获取第1页(一页五条数据)的数据

limit ($page -1)\*$numofperpage, numofperpage; -- 想要获取第2页(一页五条数据)的数据

limit ($page -1)\*$numofperpage, numofperpage; -- 想要获取第3页(一页五条数据)的数据

limit ($page -1)\*$numofperpage, numofperpage; -- 想要获取第4页(一页五条数据)的数据

# 范式

## 概念

范式就是我们设计表的基本规范！Normal Format

### 作用

通过合理的数据存储，从而使得数据的冗余度最小化以及运行效率的最大化！

### 分层

根据不同的需求标准，一层一层的严格递进，一层比一层严格，理论上来说，范式一共有6层！

* 比如：第一范式，第二范式……
* 要想满足第二范式，首先要满足第一范式，依次类推！但是，后面的范式实在是太严格了，很难达到，所以在数据库中，只引入三层范式！
* 满足三层范式的数据库就是设计比较合理的数据库！

## 第一范式

### 概念

是最容易满足的，要求把各种数据设计成一个一个的单独的字段，不能再分割！

第一范式也叫满足“原子性”

例如: 性别和年龄设计在一个字段就不满足第一范式

## 第二范式

### 概念

就是在满足第一范式的基础上再满足以下的两个条件：

* 表中的每一行都具有唯一可区分的特性（就是不存在完全相同的记录）
* 不能有部分依赖

### 什么叫依赖？

如果确定表中的某个字段A，就一定能确定该表的另一个字段B，那么我们就说B依赖于A

例如: 所有的非主键字段都依赖于主键

### 什么叫做部分依赖？

假如一个表的组合主键是（A，B），其他字段都应该依赖于（A，B），但是如果此时有一个字段C，它只依赖于A，也就是说，只要A确定了，C也就确定了，这种情况就叫作部分依赖

例如: 想要知道学生的选修分数,必须先知道选修的课程. 选修的学分就部分依赖于课程和学生

## 第三范式

### 概念

在满足第二范式的基础之上消除传递依赖

### 什么是传递依赖？

如果一个表中有某个字段不直接依赖于主键，而是依赖于其他的字段，就叫作传递依赖！

* 假如主键是A，B依赖于A，而C又依赖于B！
* 注意：并不是说C字段不依赖于主键A，而是C只能等B确定后才依赖于A！

例如: 想要了解一个学生他的院系电话和院系地址,这个时候需要先确定院系

# 联合查询

## 联合查询介绍

数据表数据量比较大时，存储时按照一定的方式分割数据表，将数据存放在不同的数据表中。可以使用union查询满足一定条件的全部信息

## 语法

两个语句的字段列表中字段顺序、数目必须一致

## union选项 all/distinct

## union应用