一、数据库简介

1、概念

数据库中持久存储数据的介质。

计算机中的数据库，就是能够高效的组织与管理数据的介质(内存、硬盘)

2、数据库的发展史

网状数据库：是指以处理记录类型为节点的网络模型来存储与管理数据的。

层次数据库：层次数据库是网状数据库之后的一种数据库，将数据组织成一种有序有向的树状结构。

关系数据库：就使用二维表模型来存储与管理数据库的。

3、关系型数据库(纯理论)

关系型数据库就是使用关系模型来组织与管理数据的数据库。

关系模型就是二维表模型，一个关系型数据库是由多个二维表及表之间的联系组成的数据库的集合。

关系：关系就是具有行列结构的表格，每个关系都有一个关系名，对应数据库中的数据表。

元组：元组可以理解为二维表中的一行，也就是数据表中的一条记录。

属性：属性可以理解为二维表中的一列，也就是数据表中的一个字段

域：是限制属性的取值范围，也就是对列的一种约束。

关键字：能够唯一标识元组(记录)，一组属性(字段)

4、常见的关系型数据库

关系型数据库：大型数据库：Oracle(甲骨文)、DB2(IBM公司)

中型数据库：SQL Server(微软公司)、**MySQL**(甲骨文)

小型：Access(微软公司)

非关系型数据库noSQL、MangoDB、Redis

5、SQL简介

**S**tructure **Q**uery **L**anguage结构化查询语言，主要用于对数据进行操作

DML Data Manuplation Language 数据操作语言 insert、delete、update

DQL Data Query Language 数据查询语言 select

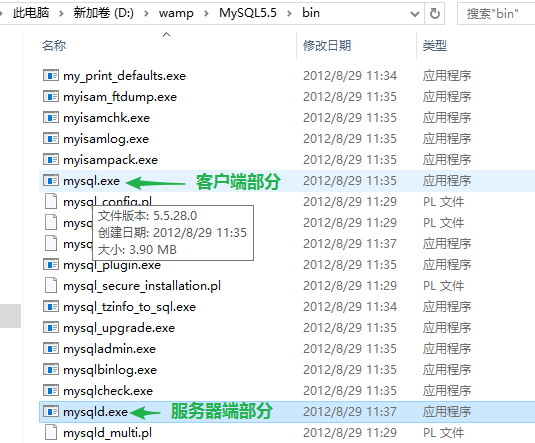
DDL Data Defination Language 数据库定义语言 create

DCL Data Control Language 数据控制语言 grant

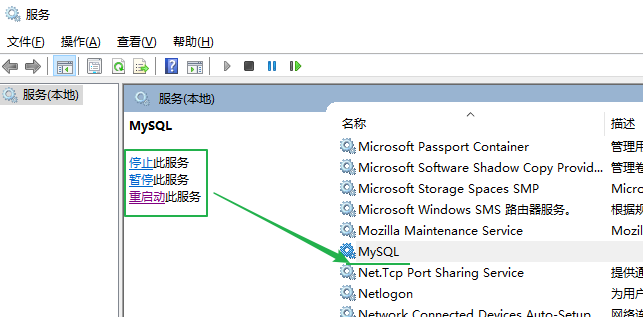
TCL Transaction Control Language 事务控制语言

二、访问数据库

1、MySQL服务器的管理



可以通过Window的服务管理器：



2、登陆MySQL数据库服务器

工具：客户端部分mysql.exe文件，运行环境是cmd窗口。

语法：

mysql -h服务器的地址 -P端口号 -u用户名 -p密码

说明：

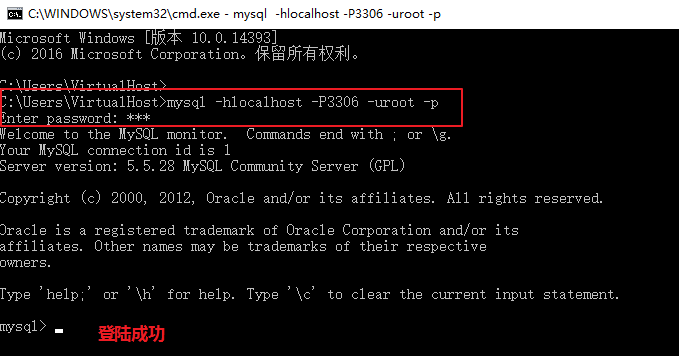
-h mysql服务器的地址 localhost表示本机 127.0.0.1表示本机，如果是本机可以省略

-P 端口号，默认是3306 如果是默认可以省略

-u 用户名 默认的帐户root是管理员帐户

-p 密码 不建议在-p后直接输入密码(注意)

示例：



3、MySQL的层次结构

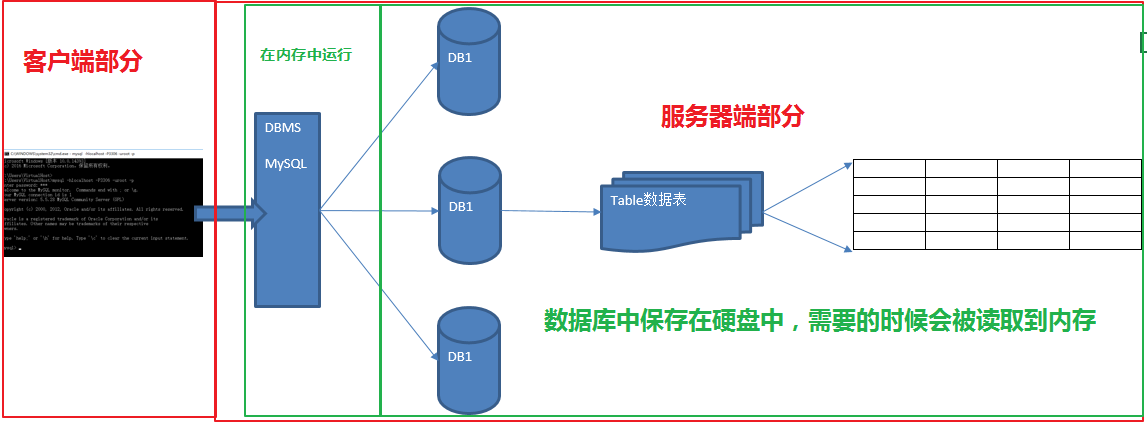
DBMS DataBase Management System数据库管理系统

DB DataBase 数据库

Table 数据表(对应现实中的表格)

Record 数据表中的一行

Fields 数据表中的一列



三、数据库的基本操作

1、查看数据库

语法： show database**s**;

说明： 查看当前有多少个数据库存在。



test是一个测试数据库，可以删除。其他余三个数据库MySQL的系统数据库。

2、创建数据库

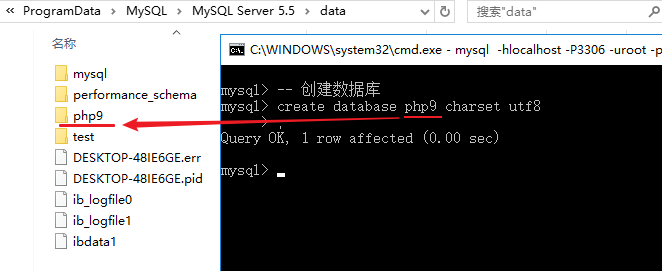
语法： create database 数据库名 【库选项】

说明：用于创建数据库，数据库名不要与MySQL的关键字重复，如果真的需要使用关键字，必须使用反引号(``)将数据库名括起来。

【库选项】

charset 用于设置存储字符集

collate 用于设置校验集



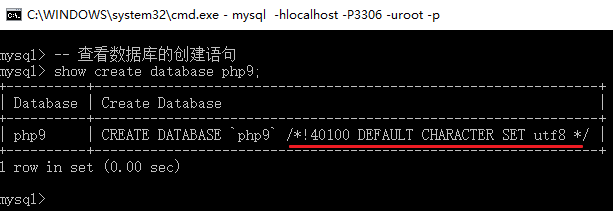
当创建一个数据库后，会在硬盘中创建同名的一个文件夹。

MySQL的语句规则：**每一条语句后必须使用;结束。分号与加车的作用是将客户cmd窗口中输出的命令或数据传递远程的服务器。**

3、查看数据库的创建语句

语法： show create database 数据库名;

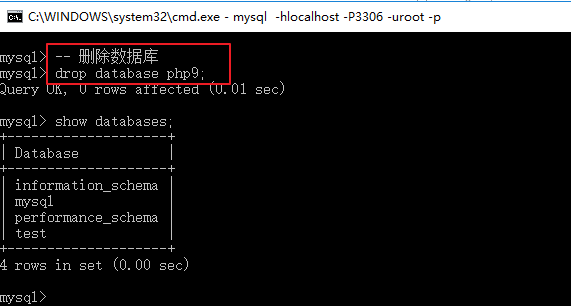
说明：用于查看数据库的创建语句，可能获取库选项的相关信息。



4、删除数据库

语法： drop database 数据库名;

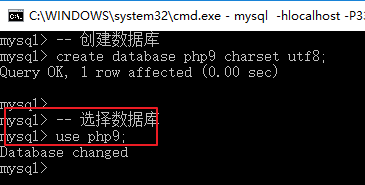
示例：



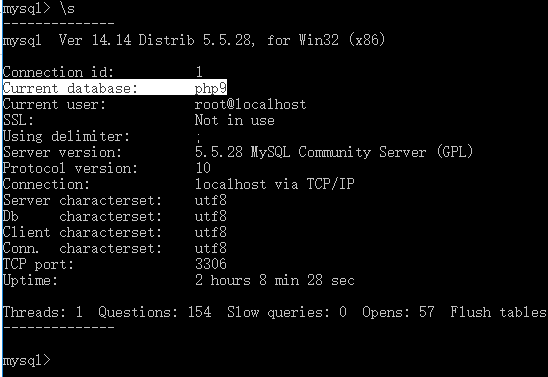
5、选择数据库

语法： use 数据库名;

说明：由于表是存储在某个具体的库中的，所以要对表操作，需先进入到指定的数据库



扩展： \s 可以查看数据库的信息，其中包含当前所在的数据库信息



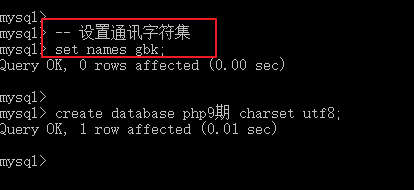
四、数据表的基本操作

1、设置客户端的通讯编码

语法： set names 客户端的字符集

说明：用于明确的告诉服务器，当前客户端与之通讯的编码是什么。

示例：



2、查看数据表

语法： show table**s**;

说明：查看当前数据库中有多少个数据表存在



3、创建数据表

语法： create table 表名(

字段名 列类型 【列属性】,

字段名 列类型 【列属性】,

….

)【表选项】

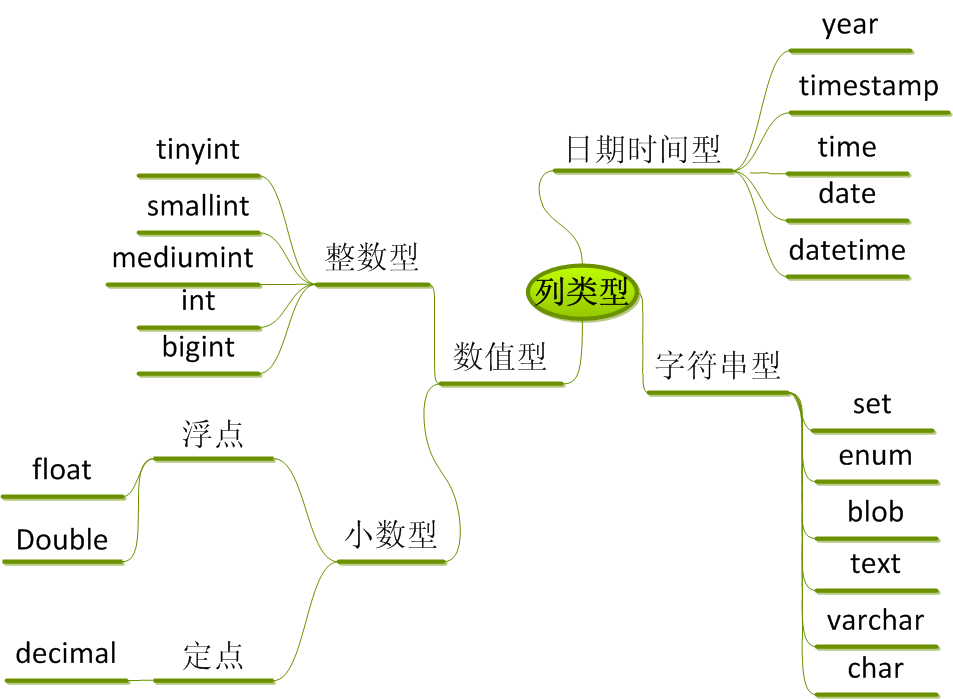
说明：列类型就是所定义的列的取值限制（对应程序中的数据类型）

【表选项】 charset 设置存储字符集

collate 设置校验集

engine 存储引擎

提示：当创建一个表后，仅是创建了一个表的结构(表头)



4、查看表结构

语法： desc 表名;

5、查看表的创建语句

语法： show create table 表名;

6、删除表

语法： drop table 表名;

五、数据的基本操作

1、增加数据

语法： insert into 表名【(字段列表)】 values (值列表)

说明：字段列表中字段名不需要加引号

值列表中的字符型的值必须加引号。

字段列表中字段的个数要与值列表中值的个数一致，而且顺序也一要致。

字段列表也可以省略，如果省略那表示全部字段插入数据。值列表中值的顺序要与表中的字段的顺序一致。

2、查看数据

①、语法：

select \*|字段列表 from 表名 [where] [group by] [having] [order by] [limit]

说明： \* 所有的字段

字段列表 用于查看部分字段。

②、where子句

【where子句】

根据where子句的表达式，对记录进行筛选。即是表达式，就会涉及到运算符。

MySQL的运算符：

>、<、>=、<=、<>、=、and、or、not、between m and n、in(值列表)、is null、like

= 用于判断是不相等，同也是赋值运算符

and、or 逻辑与、逻辑或

between m and n 字段的值介于m与n之间

in(值列表) 字段的值等于值列表中的某一个

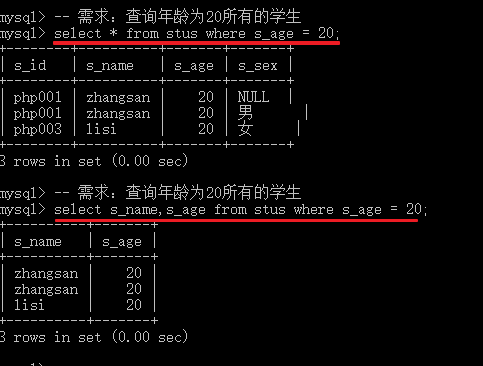
is null 专用于判断null值的运算符

like 用于模糊查询，必须与两个占位符进匹配使用 占位符：%、\_

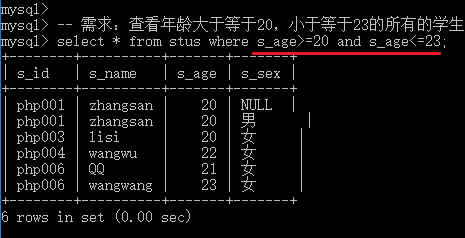
% 代表当前位置及其后0个或多个字符

\_ 代表当前位置的1个字符

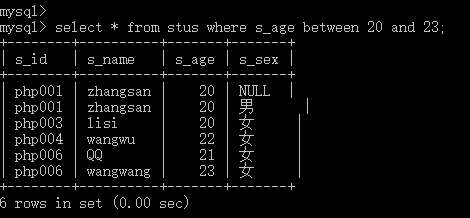
示例： =



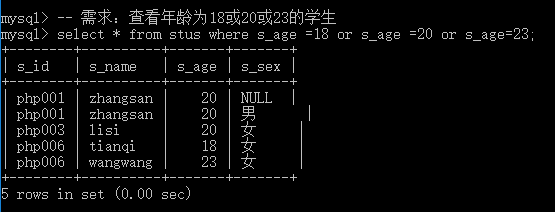
示例：and



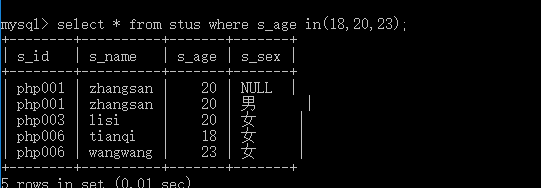
示例：between m and n



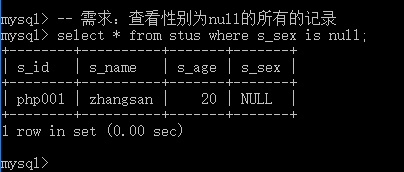
示例：or



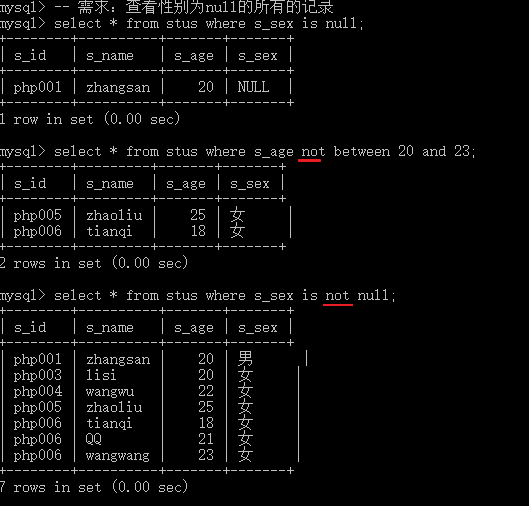
示例：in



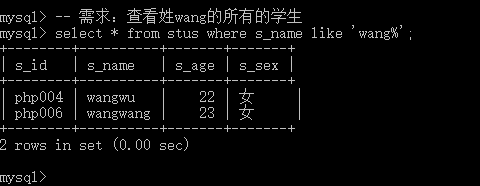
示例：is null



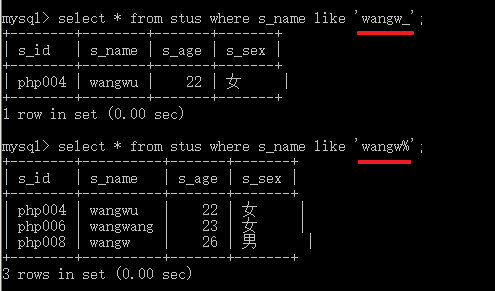
示例：not



示例：like %



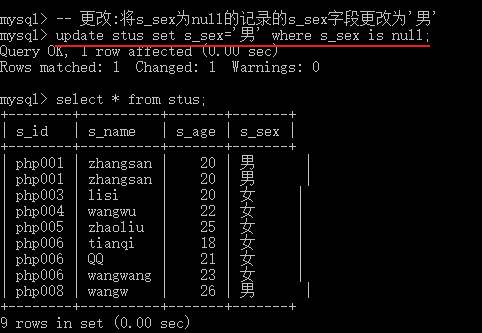
示例：like \_



3、修改数据

语法： update 表名 set 字段1=值, 字段2=值,… 【where 子句】

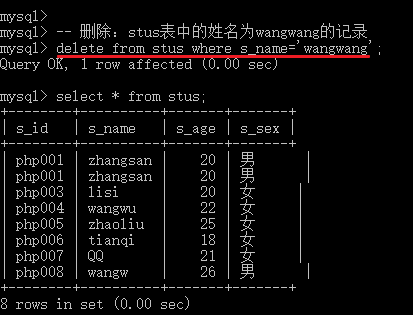
说明：根据where子句的条件匹配到的记录进行修改。如果省略where表示所有的记录都匹配。



4、删除数据

语法：delete from 表名 【where子句】

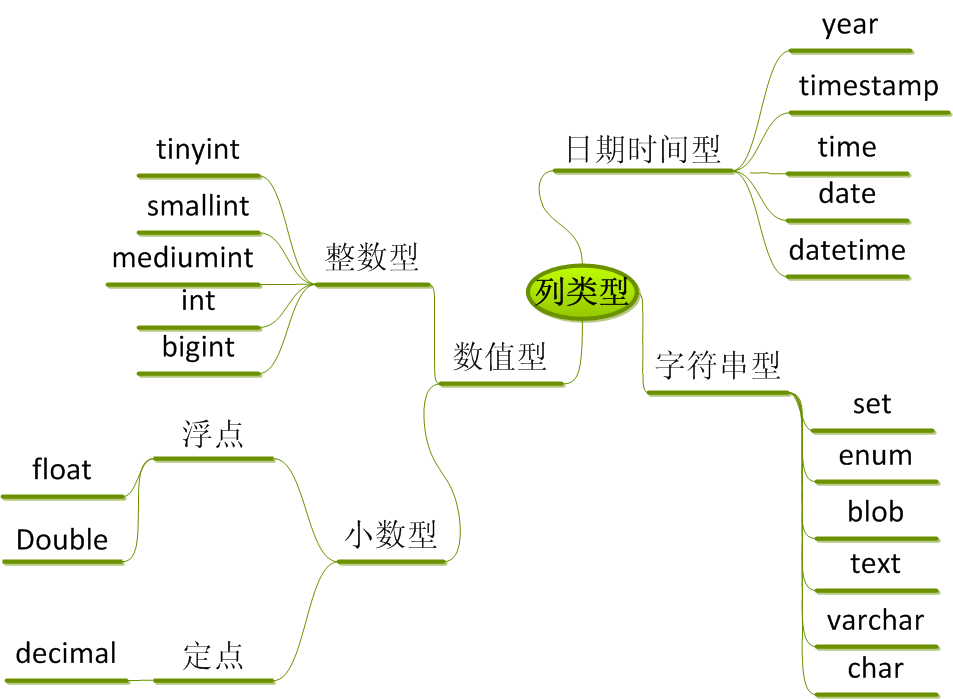
说明：将where子句中匹配到的记录进行删除，



对于结构的查看：使用的是show

对于数据的查看：使用的是select关键字

六、列类型——详解



1、数值型

①、整数型

整数型是一个系列，都是整数，唯一的区别是所能表示的数大小为同而已。

|  |  |  |  |
| --- | --- | --- | --- |
| **类型** | **字节** | **最小值（有符号）** | **最大值（有符号）** |
| **最小值（无符号）** | **最大值（无符号）** |
| TINYINT | 1 | -128 | 127 |
| 0 | 255 |
| SMALLINT | 2 | -32768 | 32767 |
| 0 | 65535 |
| MEDIUMINT | 3 | -8388608 | 8388607 |
| 0 | 16777215 |
| INT | 4 | -2147483648 | 2147483647 |
| 0 | 4294967295 |
| BIGINT | 8 | -9223372036854775808 | 9223372036854775807 |
| 0 | 18446744073709551615 |

在使用整数来表示实现中数据时，有时现实中数据不可能出现负数。所以可以使用unsinged关键字来将有符号位的整型类型，更改为无符号位。

②、小数型

**小数型又分为浮点与定点(了解)：**

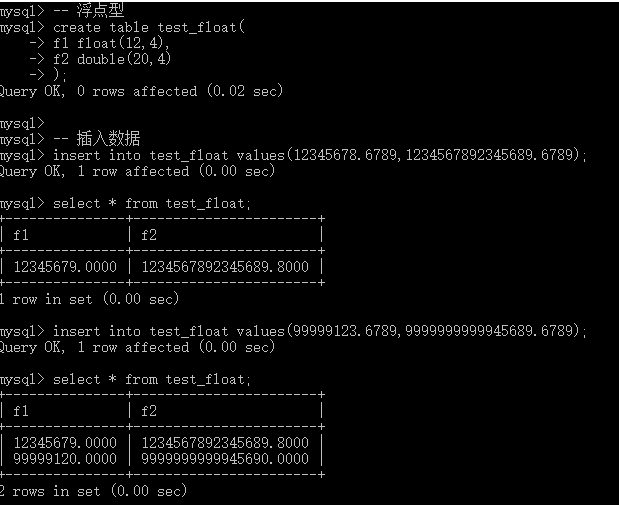
**浮点类型分为float与double**

|  |  |  |  |
| --- | --- | --- | --- |
| **类型** | **字节** | **最小值** | **最大值** |
| Float | 4 | -3.402823466E+38 | 3.402823466E+38 |
| Double | 8 | -1.7976931348623157E+308 | 1.7976931348623157E+308 |

语法： float(m,d) double(m,d)

说明： m（表示的是总位数）、d（表示的是小数位）、整数位 m-d

示例：



说明：

对于浮点型会出现精度丢失的问题。

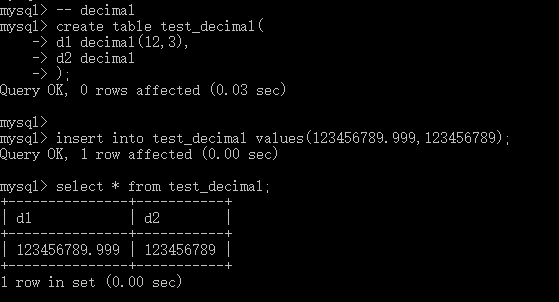
对于float类型仅能精确到5至6位，多于的位会使用0代替

对于double类型仅能精确到14至15位，多于的位会使用0代替

decimal(定点型)：语法： decimal(m,d)

说明： m（表示总位数，m最大的取值为65）、d（表示小数位，d最大的取值为30）

示例：



2、字符型

①、定长

语法： char(len)

说明： len表示字符的个数。最大是255。

|  |  |  |  |
| --- | --- | --- | --- |
| 表的存储字符集 | 设置的列属性 | 实际存储的数据 | 实际所占的空间 |
| Utf8 | Char(10) | 5 | 10\*3=30 |

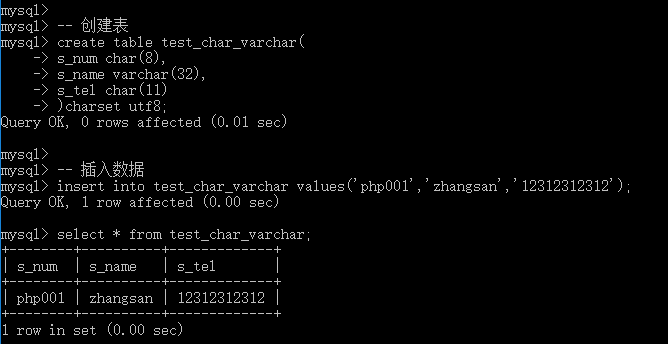
②、变长

语法： varchar(max)

说明： max表示字符的个数。理论是65535(会受很多条件限制)

|  |  |  |  |
| --- | --- | --- | --- |
| 表的存储字符集 | 设置的列属性 | 实际存储的数据 | 实际所占的空间 |
| Utf8 | varchar(10) | 5 | 5\*3=30 |

示例：



③、文本类型

text也是一个系列，包含

tinytext、text、mediumtext、longtext，唯一的不同就是所能存储的字符的多少不同。

④、blob类型

blob也是一系列，包含

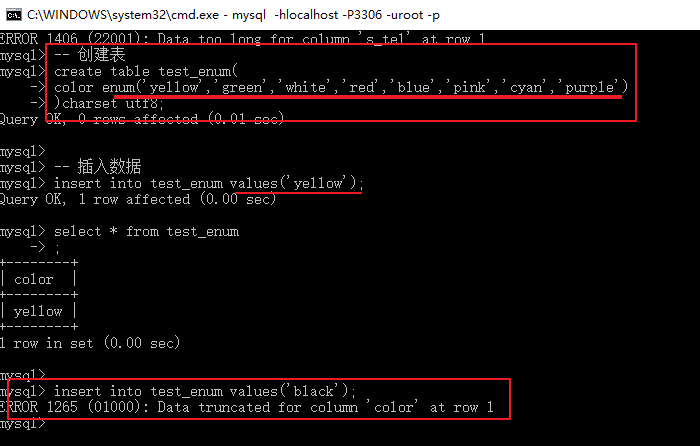
tinyblob、blob、mediumblob、longblob，唯一的不同就是所能存储的数据的多少不同。

blog可以存储二进制数据

⑤、枚举类型

用于规范数据，只允许用户从多个预定的数据中，选取一个。

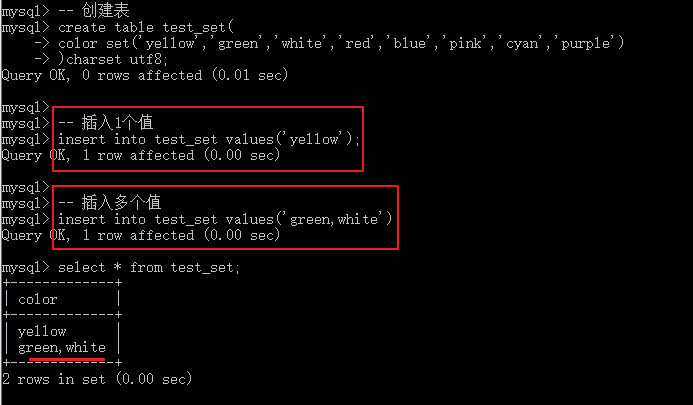
语法：enum(值列表)



⑥、集合类型

用于规范数据，只允许用户从多个预定的数据中，选取多个。

语法： set(值列表)



3、日期时间型

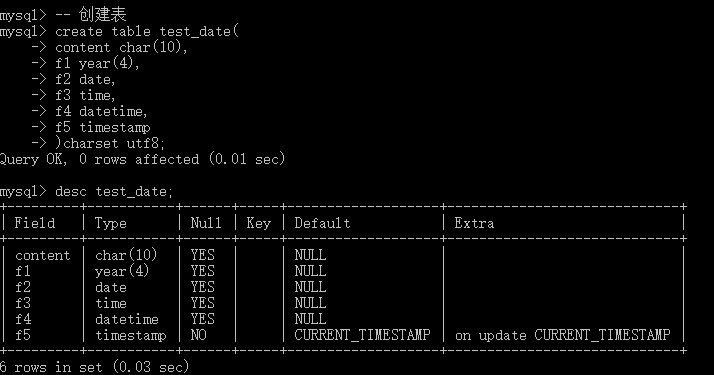
year(2或4) 年份 2表示2位的年份;4表示4位的年份

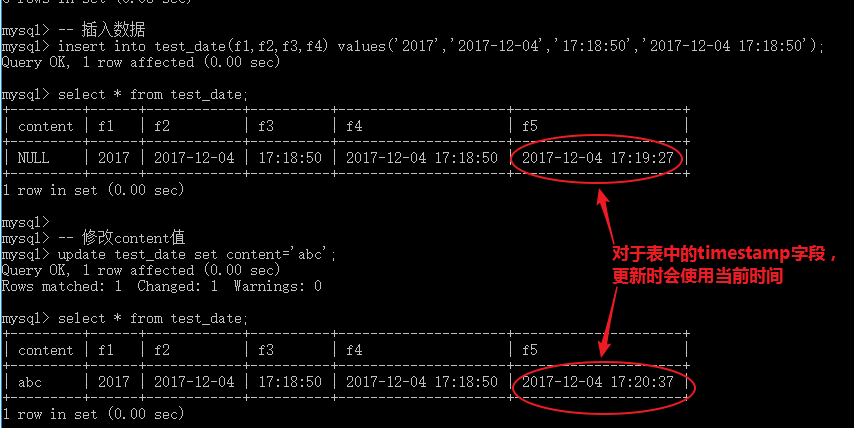
date 日期 格式： '2017-12-4'

time 时间 格式： '17:14:59'

datetime 日期时间型 格式： '2017-12-4 17:14:59'

timestamp 时间戳，主要的作用是用于记录数据表中记录最后一次被修改的时间。





七、字符集与校对集

1、字符集

①、查看MySQL所支持的字符集

语法： show charset;

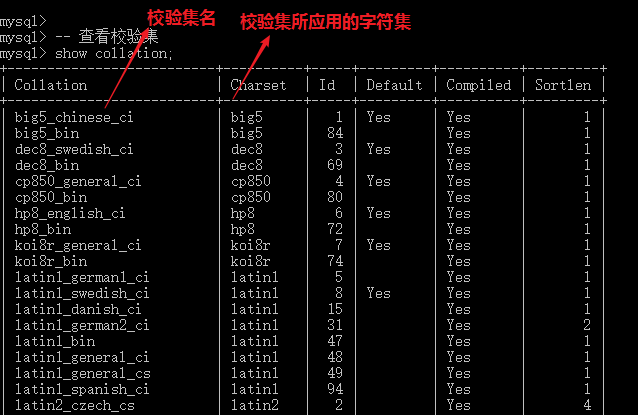


2、校验集

校验集就是某一种字符集中的比较规则。也就是是否区分大小写。

查看校验集

语法： show collation;



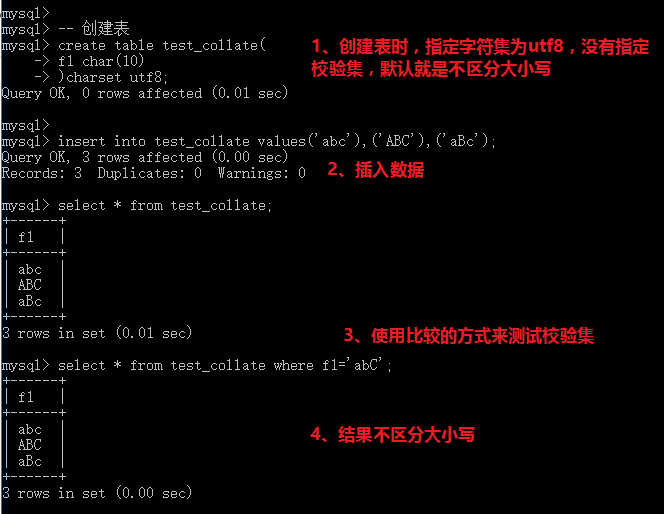
提示： 虽然有很多校验集，但通过观察发现这么多的校验集是以三种方式结尾：

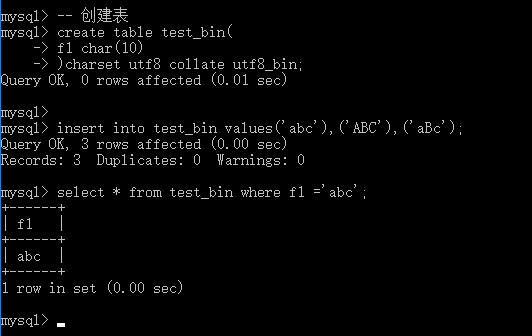
\_cs CaseSensitive 大小写敏感(区分大小写)

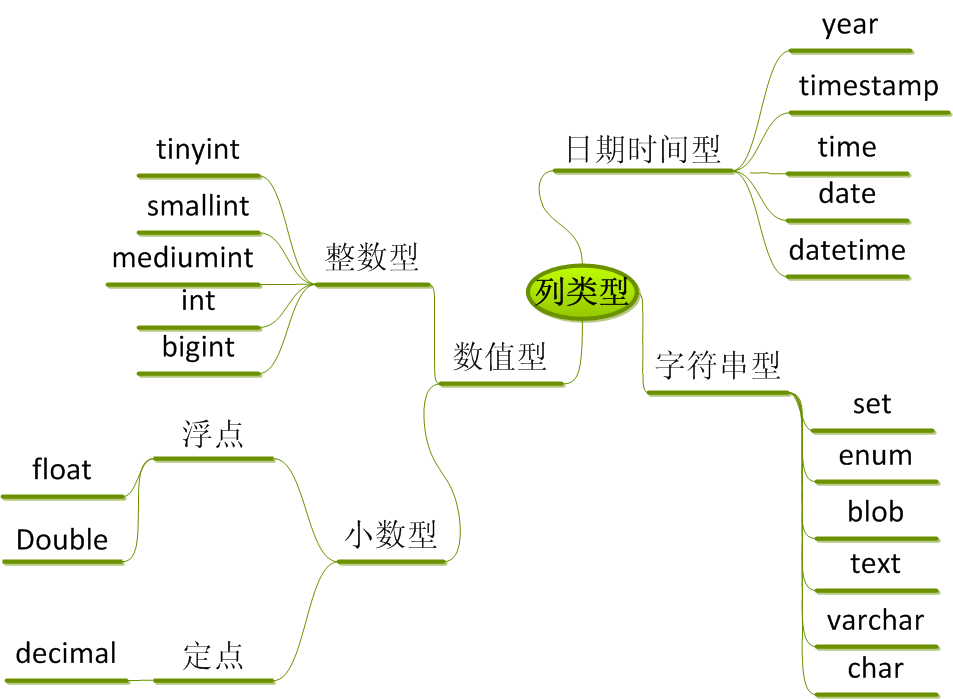
\_ci CaseInsensitive 大小写不敏感(不区分大小写)

\_bin 二进制 区分大小写

核验集主要是体现在比较上。



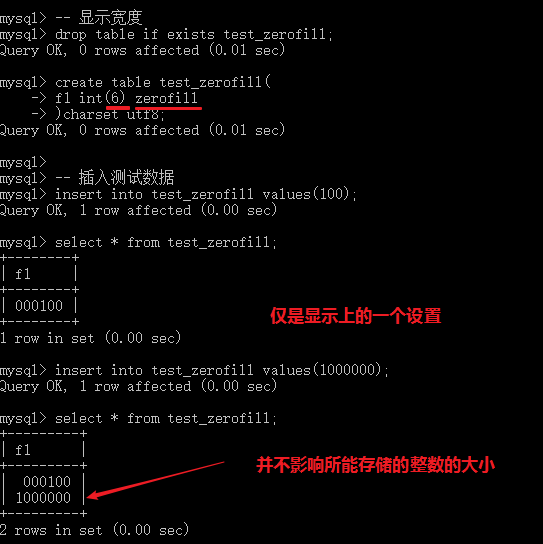




一、显示宽度与0填充

在我们为某个字段设置整数类型时，可以为其指定显示宽度。必须与zerofill联合使用

语法：整型(显示宽度)



二、列属性

create table 表名(

字段名 列类型 【列属性】,

字段名 列类型 【列属性】,

…..

)【表选项】

1、列类型 VS 列属性

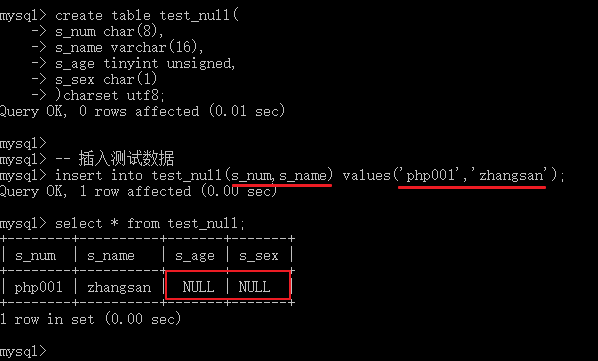
列类型主要用于约束单独的一个字段在整行中的取值限制。

列属性主查用于约束单独的一个字段在整列中的取值限制。

2、null

MySQL系统的默认值，

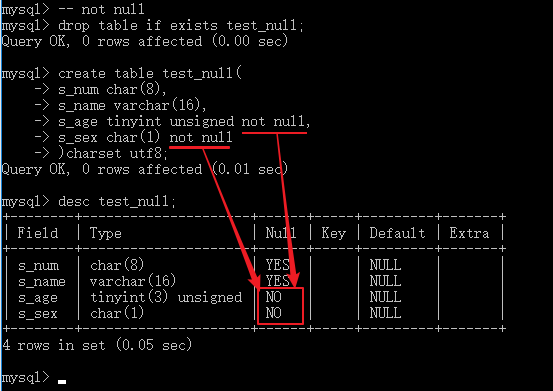
默认，在插入记录时当我们没有为某个字段提供数据时，那么mysql会使用null值为填充。



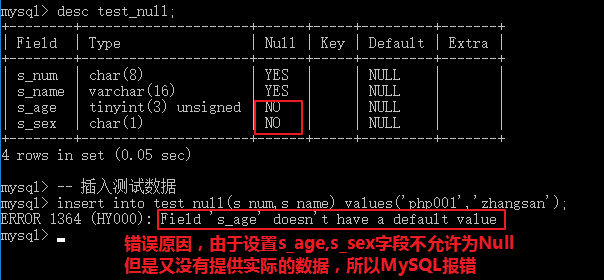
提示：一般我们在创建表时，不允许字段的值为null。

3、not null

主要用于约束当前的字段不允许为null值(不允许MySQL使用null值进行填充)



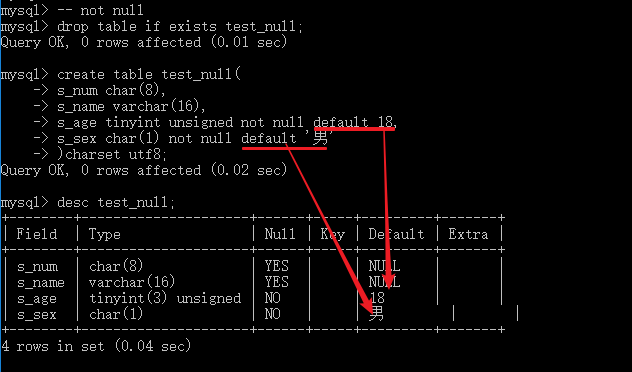
插入测试数据：



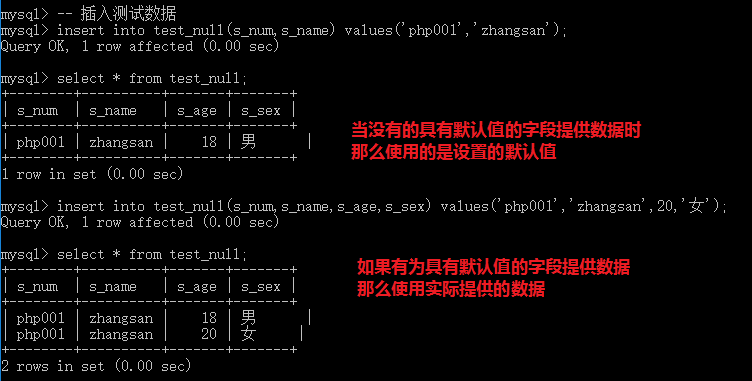
当为某个字段设置了not null约束后，要么在插入数据时必须提供数据，要么为not null字段设置一个默认值。

4、default

自定义默认值。

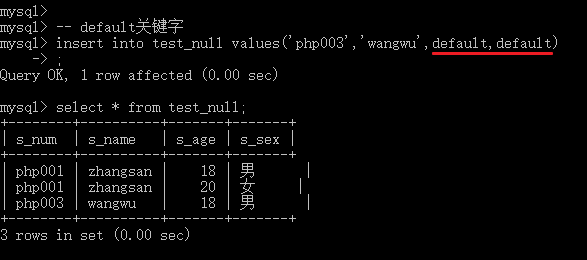


插入测试数据：



default关键字

当我们插入记录时，也可以使用default关键字。来代替默认值。



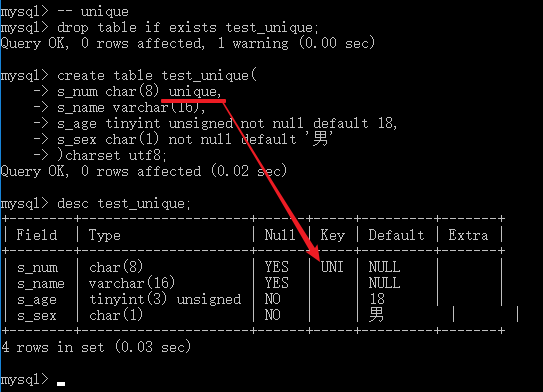
提示：

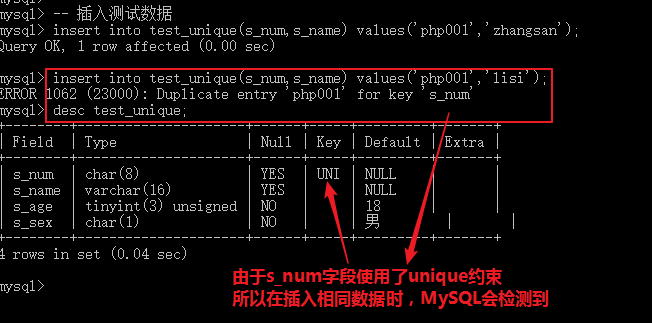
在设置默认值时，如果有很大的机率能确定这个值就使用确定的值。

如果没有办法确定，对于字段型的字段使用default ‘’，对于数值型使用default 0

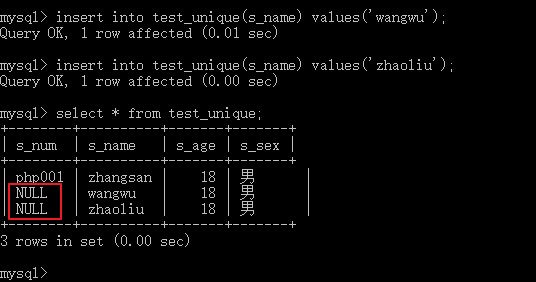
5、unique

用于约束当前的字段在整列中都是唯一的。





示例：unique的特点



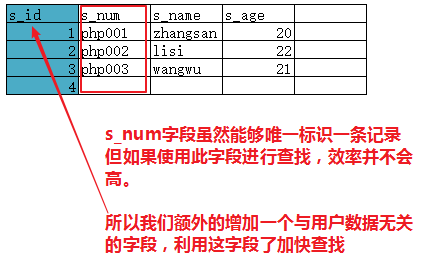
提示：

unique约束，不允许字段的值重复，但是允许为null，原因是null值根本不等于null

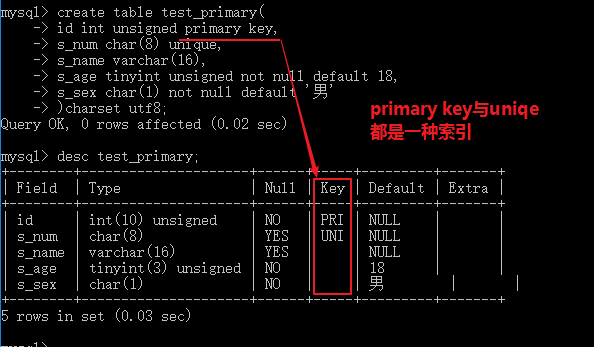
6、primary key

Primary（主要的意思）、key（键，字段）

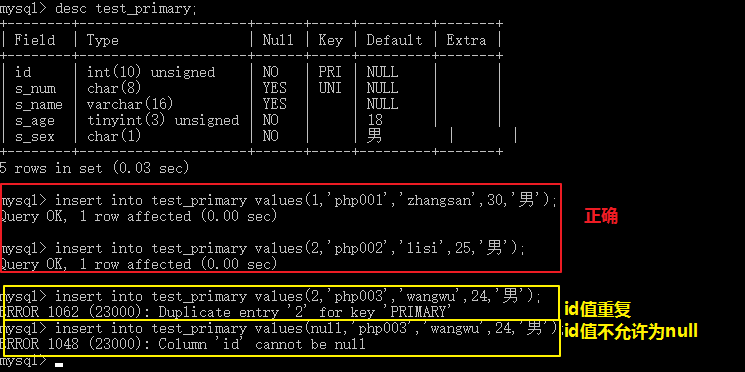
primary key 重要的主要的字段。此处的重要并不说对用户重要，而是对MySQL来说重要。主要体现在对记录的查找上。



示例：

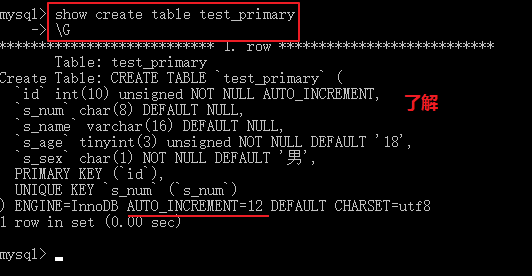


示例：插入测试



**提示：主键的特点，不允许重复，且不允许为null值**

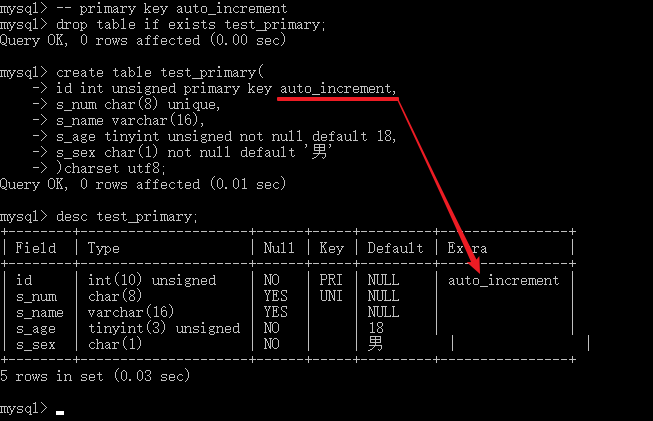
查看下一个自增长的主键的值。



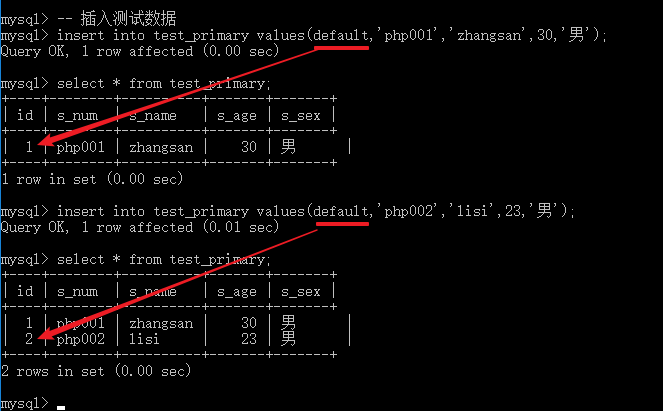
7、auto\_increment

对于主键的值，我们选择了整数型，就是想达到一种自动增长，这样就不会重复。而且能唯一标识一条记录。

auto\_increment，是自增长，就是用于解决主键值的问题。



示例：

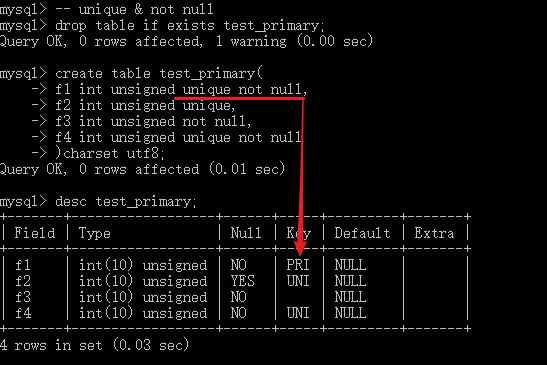


提示：

对于auto\_increment属性，必须应用在整型字段并且是primary key 或者unique。但是绝大多数与primary key联合使用。

8、模拟主键(了解)

当一个表中如果没有设置primary key，那么MySQL会在表中寻找第1个具有unique与not null约束的字段，作为主键字段。



提示：

一个表中能有1个主键

一个表中能有多个唯一键

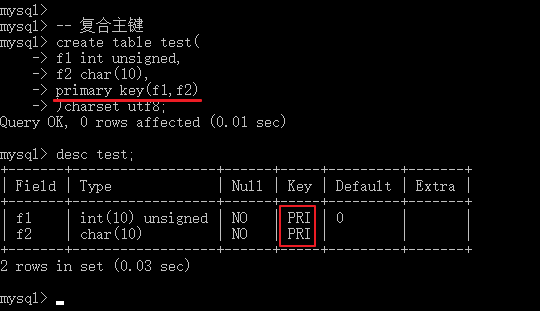
一个表中能有1个自增长

9、复合主键(了解)

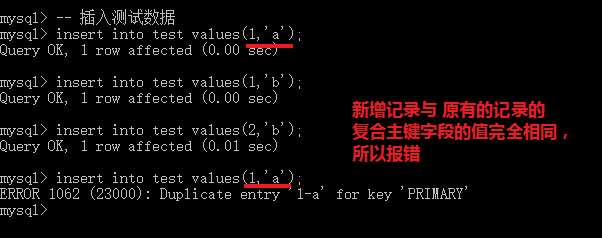
MySQL中可以使用多个字段，共同组成主键字段。

语法： primary key(字段1,字段2)

示例：

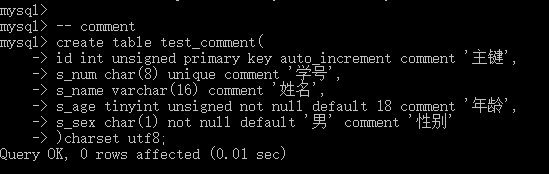


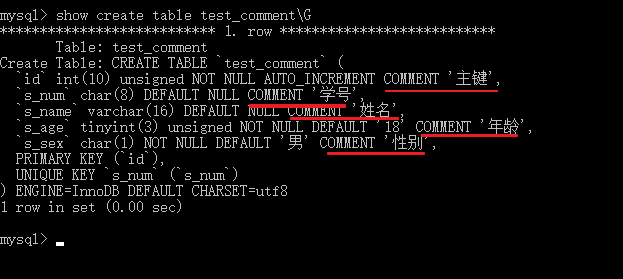
对于复合主键的值，只要当组成复合主键的字段的值完全相同才算是重复。



10、备注

语法： comment ‘备注’



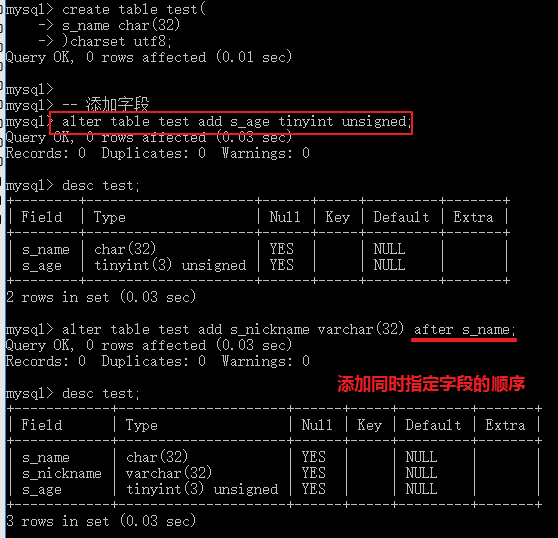


三、修改表结构

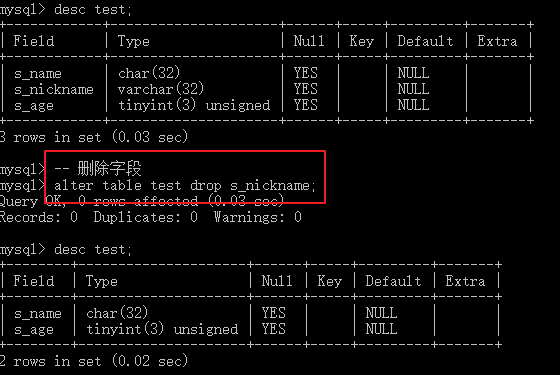
**总的语法：** alter table 表名 add column|add primary key|drop|drop primary key|drop key|modify|change|rename|

1、增加与删除字段

①、添加alter table 表名 add 【column】 字段名 列类型 列属性 【first| afterp字段名】

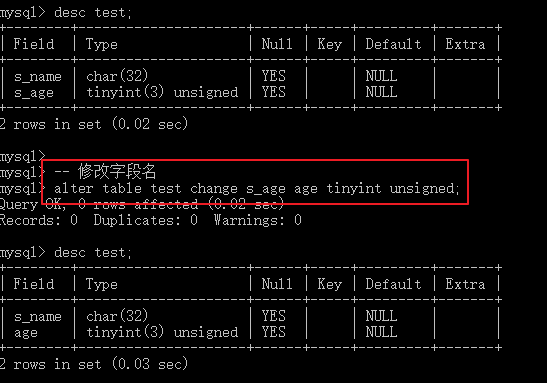


②、删除：alter table 表名 drop 【column】 字段名;



2、修改字段名

alter table 表名 change 原字段名 新字段名 列类型 列属性。

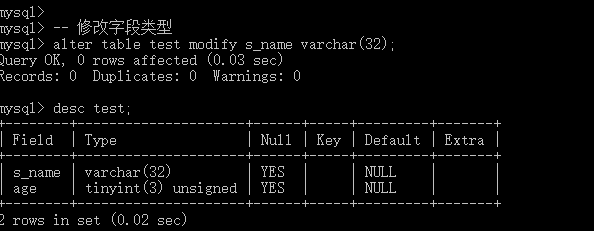


提示：就算是仅修改字段名，那么字段的原类型，原属性也要重新书写，不然就变为删除。

3、修改列类型

alter table 表名 modify 字段名 列类型 列属性 【first|after】

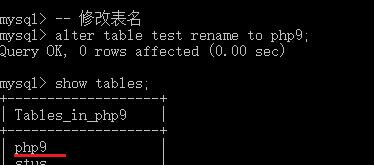
示例：



4、修改表名

alter table 表名 rename to 新名

示例：

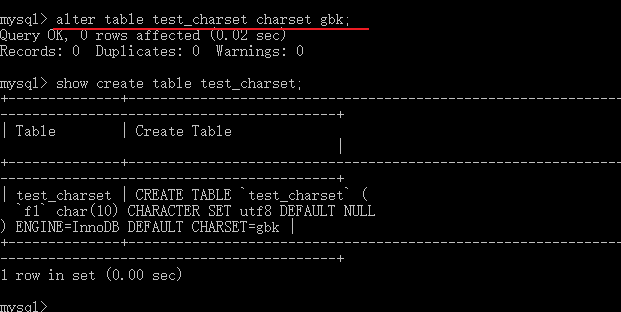


5、修改表选项

create table 表名( )【表选项】

Engine（存储引擎）、Charset（存储字符集）、collate（校验集）

语法： alter table 表名 engine 存储引擎|charset 存储字符集|collate校验集



提示： 虽然MySQL提供了修改存储字符集的命令，但是如果一个表中已有数据，那么不要执行修改存储字符集的命令。

6、修改列属性

列属性包含not null、default、unique、auto\_increment、primary key。

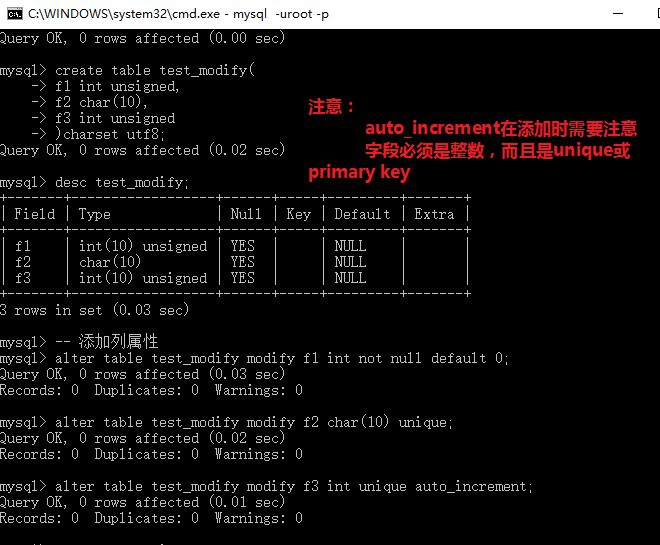
当一个表创建之后，对于列属性的操作，我们可以使用alter table 表名 modify来操作，在操作时，如果书写了列属性就是添加列属性，如果没有书写就是删除原有的列属性。

但是有两个特殊的列属性，primary key 与unique。

①、增加列属性

**普通的列属性**

alter table 表名 modify 字段名 not null defult 值 unique、auto\_increment



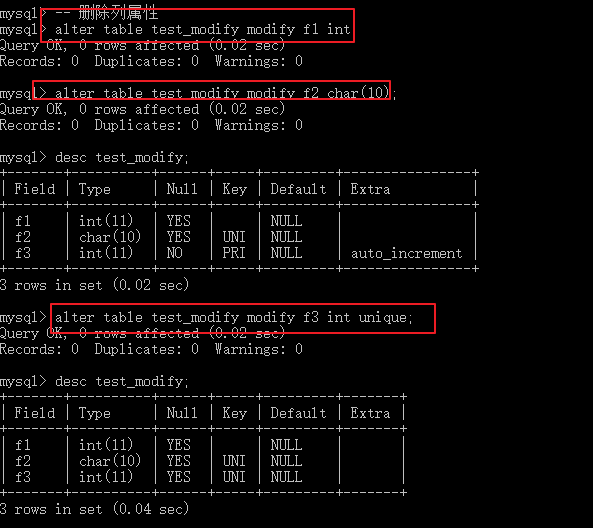
提示：注意unique与default不能联合使用。

**主键字段的添加**

语法： alter table 表名 add primary key(字段名)

②、删除属性

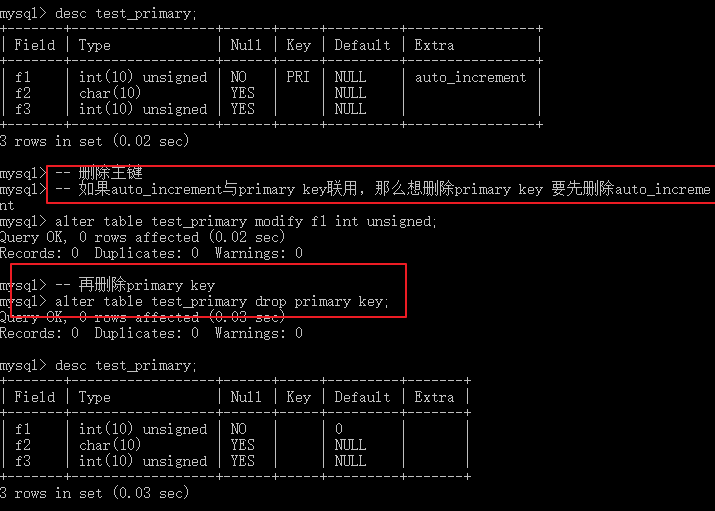
**普通的列属性**



**对于not null、default、auto\_increment不书写就是删除。**

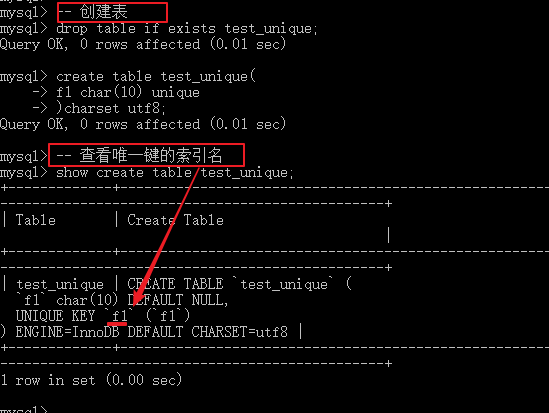
primary key与unique的删除

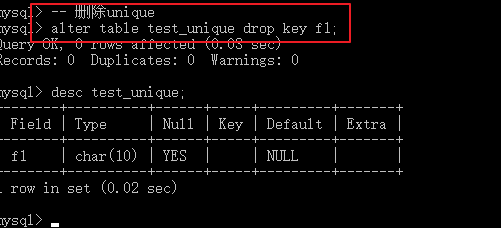
alter table 表名 drop primary key;



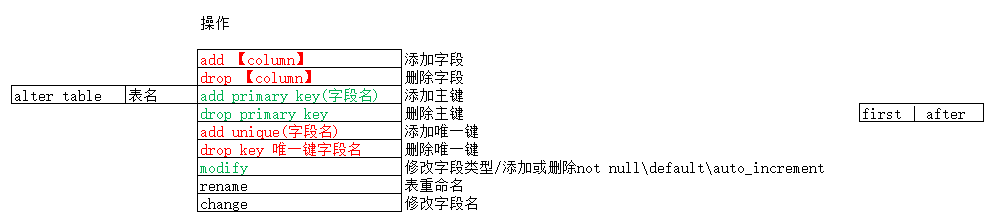
alter table表名 drop key 唯一键的索引名;

默认唯一键的索引名与字段名相同。





小节：



四、实体与实体关系(了解)

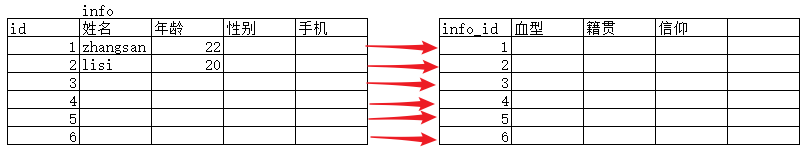
实体：

现实世界中的任何一个事物都是一个实体。在使用MySQL保存现实中的事物的信息时，那么一个事物(一个实体)的信息就保存在一个表中。

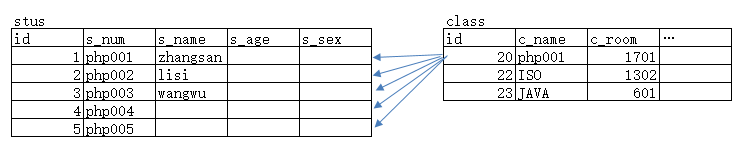
现实中的实体与实体之间具有关系，那么MySQL数据库中，保存实体的表与表之间也会反应出现实中实体与实体之间的关系。

这样的关系可以分为以下三类：

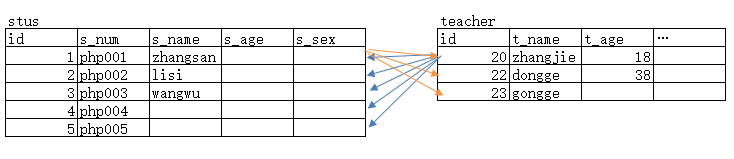
一对一



一对多



多对多



五、高级操作

1、复制表结构

语法：

create table 表B like 表A;

说明：  
 使用表A的结构重新创建一个表B

只是复制结构，不包含表的数据。

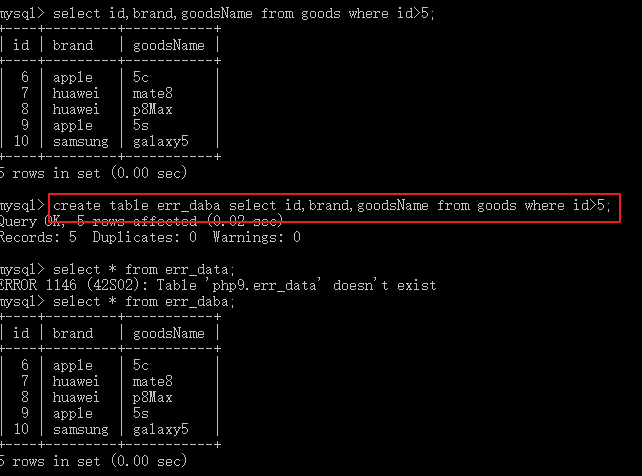


2、备份SQL执行结果

语法：

create table 表名 select 语句;

说明：将一条select语句得到的结果保存到一个新创建的表中。



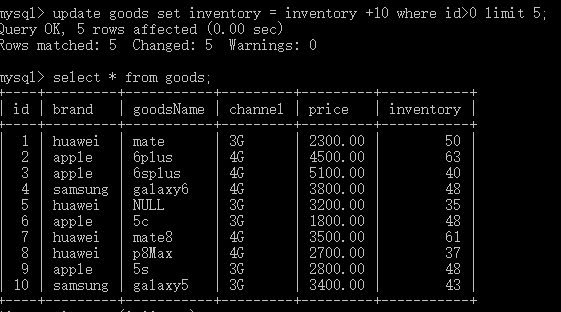
3、限制更新

语法：

update 表名 set 字段=值 【where子句】【limit子句】

说明：

用于将where子句匹配到记录，仅更改limit子句限制的条数。



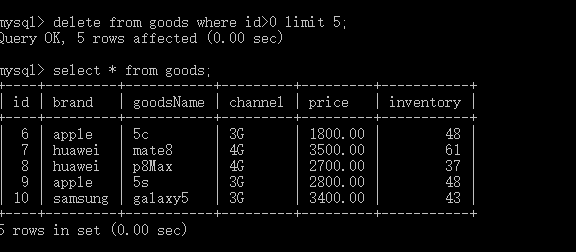
4、限制删除

语法：

delete from 表名 【where子句】【limit子句】

说明：

用于将where子句匹配到记录，仅删除limit子句限制的条数。



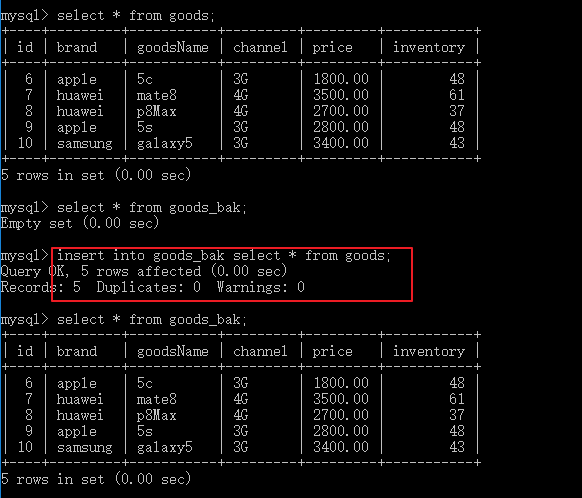
5、批量插入

语法：

insert into 表A【(字段列表)】 select 字段列表 from 表B;

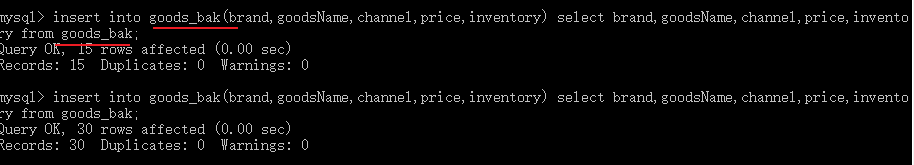
说明：

将select语句得到的数据，插入到表A中。



6、蠕虫复制

蠕虫复制是批量插入的一个特殊情况，在使用批量插入语句时，如果插入的表与数据查询的表是同一个表，那么这种情况，术语就叫蠕虫复制。



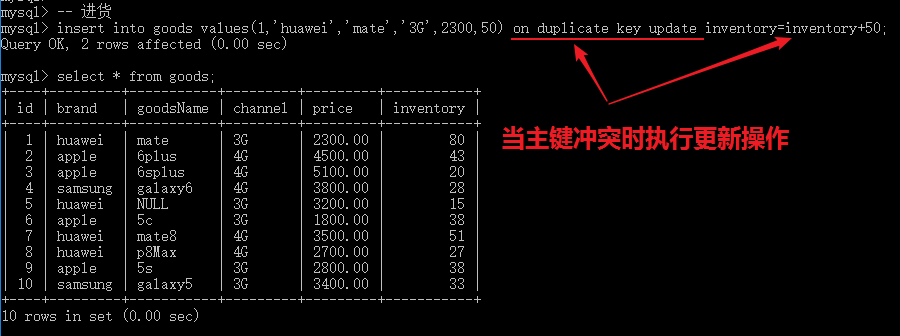
7、主键冲突

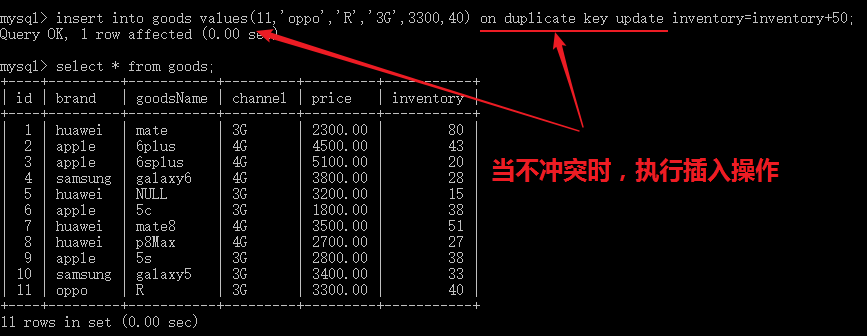
主要讲述的就是主键冲突的解决方案》

①、冲突更新。

语法：

insert into 表名【(字段列表)】 values(值列表) **on duplicate key update** 字段=值,字段=值…;





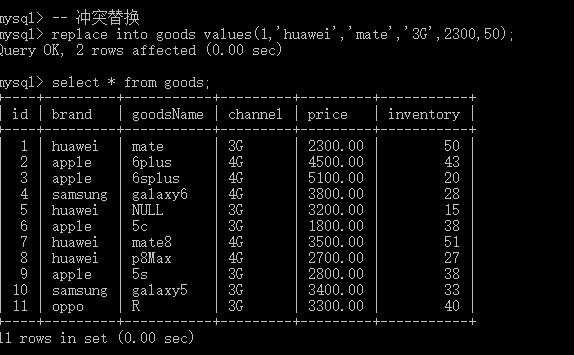
②、冲突替换

语法：

replace into 表名【(字段列表)】 values(值列表)

说明：

如果主键冲突，会将所有的字段替换掉



8、清空表

语法：

delete from 表名;

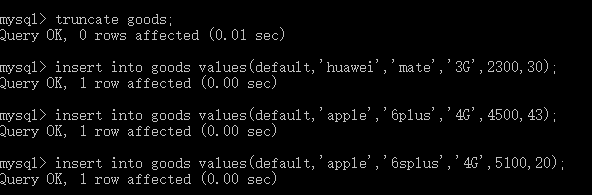
说明：

只会将表中的数据删除，并不会重建索引。

语法：

truncate 表名;

说明：  
 不但会删除表中的数据，而且还会重建索引。





一、高级查询

数据库主要是用于对数据进存储，更新、删除、查询操作CURD。在这几种操作中，最重要的就是查询。

select 【all|distinct】 字段列表|\*|字段名 【as】 别名 from 数据源 【as】 表别名 【where子句】【group by子句】【having子句】【order by子句】【limit子句】

1、select选项

【all|distinct】

说明：

all (默认)在显示结果中包含重复的数据

distinct 在显示结果中去掉重复的数据。

此处的重复是在显示结果中的整行来对比。

示例：



示例：



2、字段别名

在查行数据查询时，我们可以对查询出来的字段名，在显示时设置一个别名。

示例：



3、数据源

所谓的数据源就是数据的来源。

数据源可以分为单表数据源，多表数据源、子查询数据源。

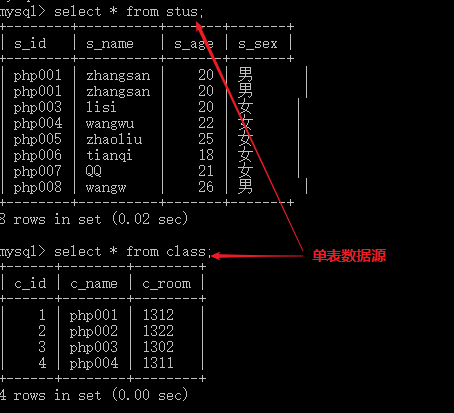
单表数据源： 数据的来源只来源于一个表

多表数据源： 数据的来源来源于多个表

子查询数据源： from后数据的来源是另一个select语句。

①、单表数据源

示例：单表数据源



②、多表数据源

语法：

select \* from 表A,表B

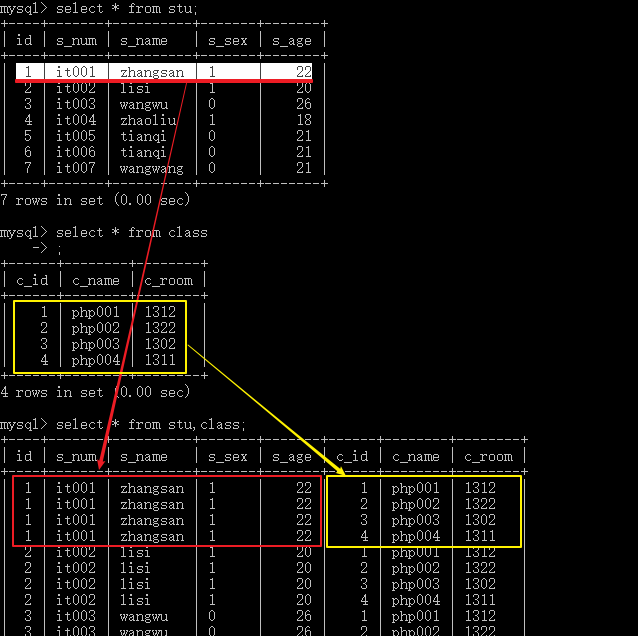
说明：

多表数据源产生的结果是一个迪卡尔积，

迪卡尔积的形式是两个表的字段数相加，记录数相乘。

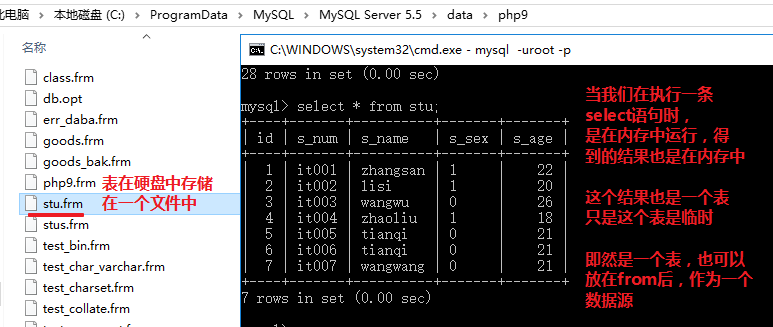
迪卡尔积，没有实际的意义，但是以后所要讲的连接查询都是以迪卡尔积为基础的。

示例：

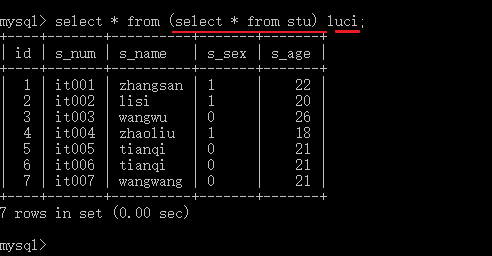


③、子查询数据源

表中的数据是存储在硬盘中的一个文件中。



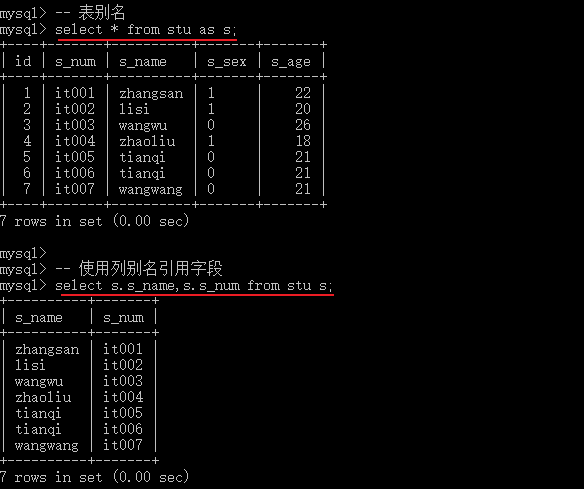
示例：



4、表别名

在一条select语句中引用一个表时，也可以为这个表起一个表别名，主要应用在连接查询中。

示例：



5、【where子句】

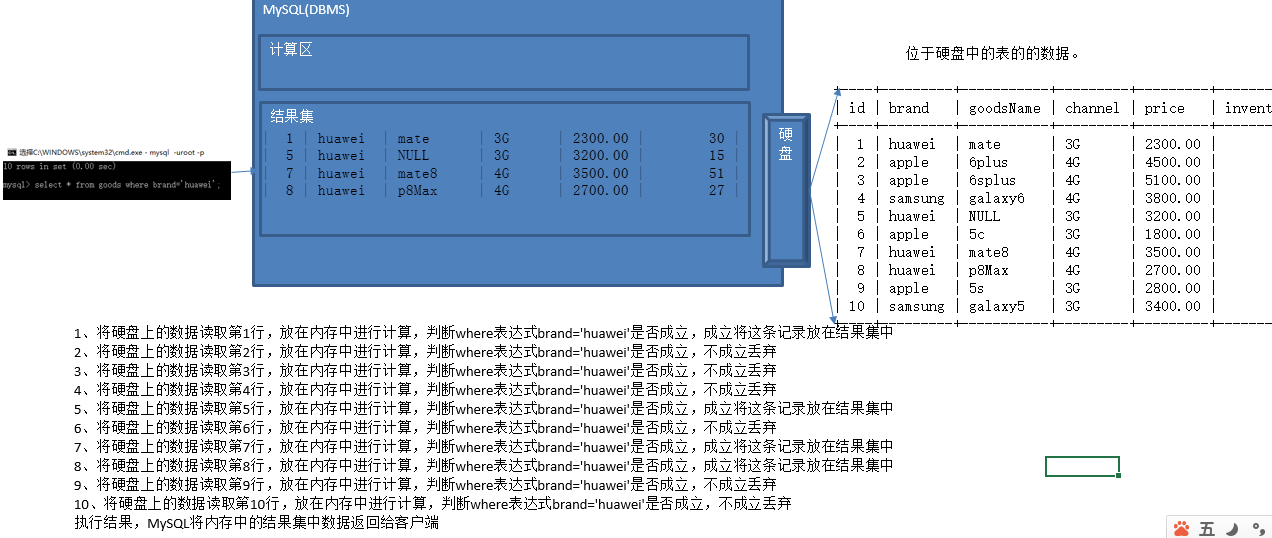
【where子句】【group by子句】【order by子句】【having子句】【limit子句】

这些子句称这五子句，五子句可以随意的组合使用，但是组合中的顺序必须按语法的顺序书写。

①、使用

where子句，根据表达式，在将数据从硬盘中读取到内存时进行第1次筛选。

②、原理



提示：

如果省略where子句，表示所有的记录都匹配

6、【group by子句】

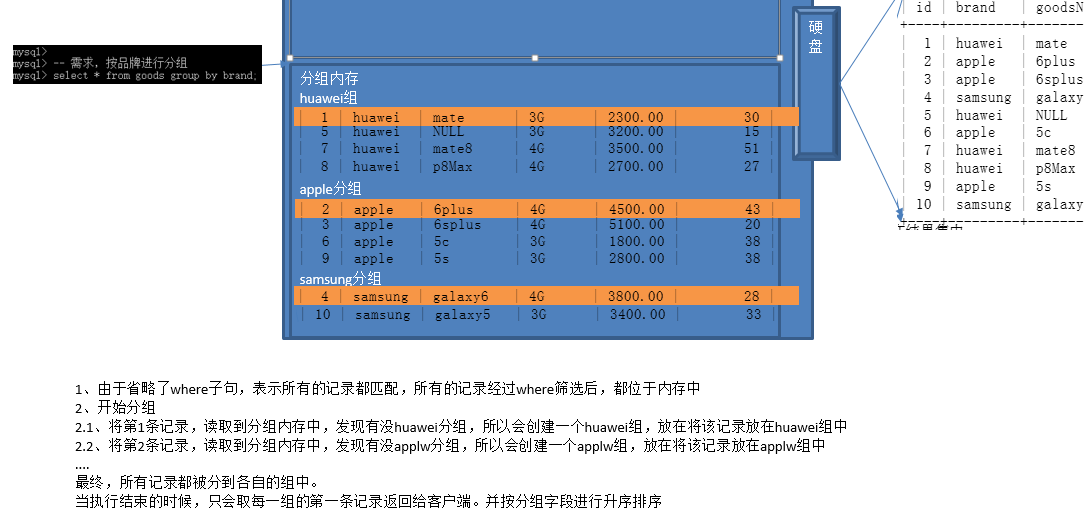
语法：

group by 字段1,字段2..

①、作用

group by 是对where子句得到的结果进行，分组统计

②、原理



提示：

如果省略group by表示，相当于所有的记录为一个大组，但是要把所有的记录都返回。

③、统计函数

group by 重要是在统计上，MySQL对分组的数据提供了很多的统计函数

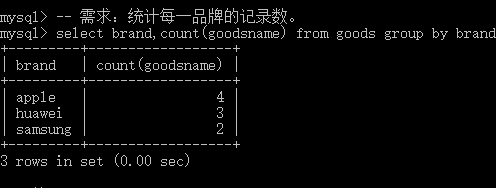
count(字段名|\*)

对group by得到的每一组，求每一组的记录数。

count(字段名)

按指定的字段进行统计，如果某一条记录的被统计的字段是Null值，这条记录并不会参与统计。

示例：count(字段名)



count(\*)

统计记录数，并不受null值的影响。

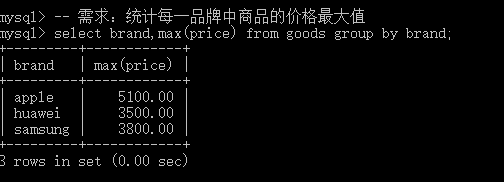
示例：



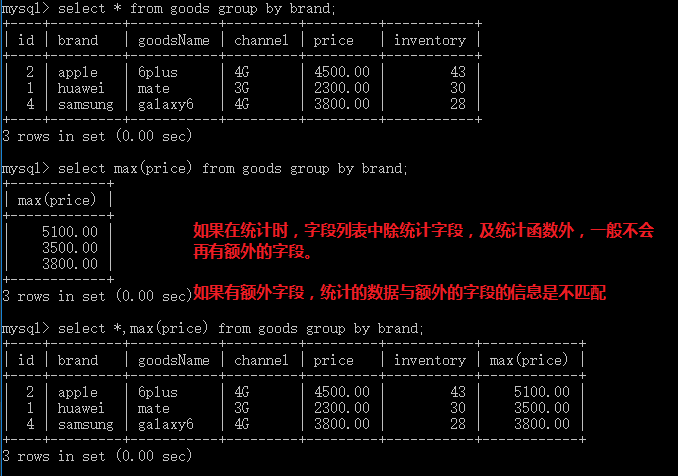
max(字段名)

统计每一组中的指定的字段的最大值。

示例：



示例：



min(字段名)

用于统计，每一组中指定字段的最小值。

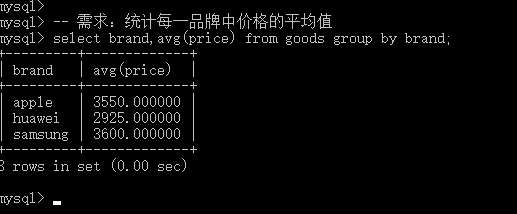
示例：



avg(字段名)

用于统计每一组中指定字段的平均值。

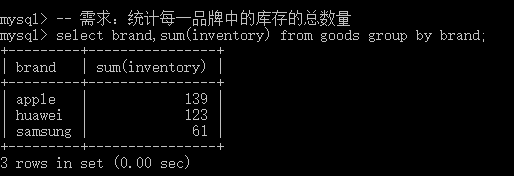
示例：



sum(字段名)

用于统计每一组中指定字段的和。

示例：



扩展：

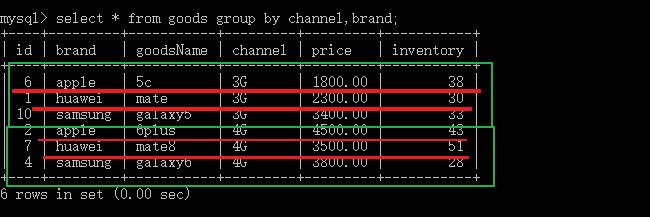
所有的统计函数后都可以使用一个别名来表示。



④、多字段分组

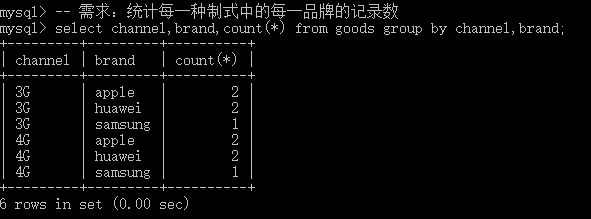
group by 字段1,字段2…

当group by后指定多个字段时，会进行多字段分组



在多字段分组的同时使用统计函数。

示例：



分析：

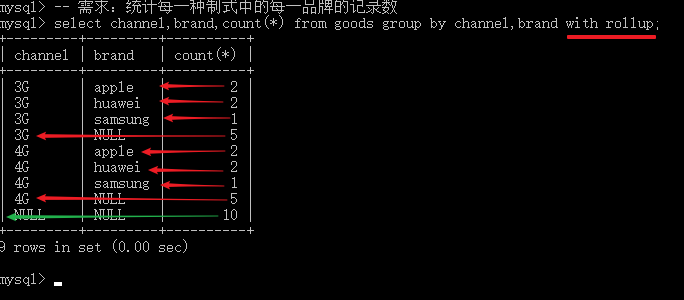
在多字段进行分组时，最终的统计数据，是针对最小组统计。如上例中是针对品牌所划分的组。

⑤、回溯统计

with rollup

多字段进行分组，统计函数默认是应用在最小组上的，如果想对包含最小组(最小组之上)的大组进行同样的统计，那么with rollup就可以实现这个需求。

示例：



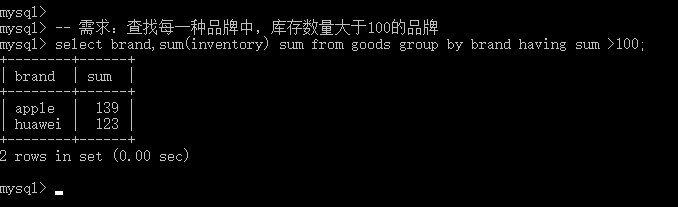
7、【having 子句】

作用：

where是在将硬盘中的数据读取到内存时进行第1次筛选。

group by是针对where子句匹配以的记录进行分组统计

having是针对group by分组统计得到的结果再进行第2次筛选。



扩展(了解)：

统计函数，是不能出现在where子句中。

having可以代替where子句，但是强烈不建议这么用。

8、【order by 子句】

语法：

order by 字段1【asc|desc】【, 字段2【asc|desc】】…

说明：

order by 对where子句、group by子句、having子句得到的结果进行一个显示顺序上的控制。

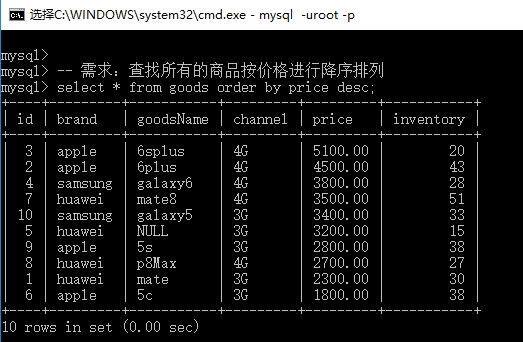
asc 缺省的升序

desc 降序

示例：asc



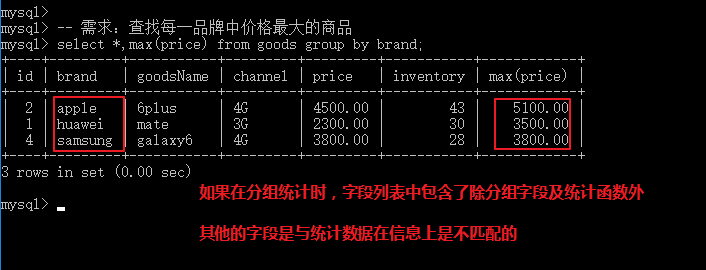
示例：desc

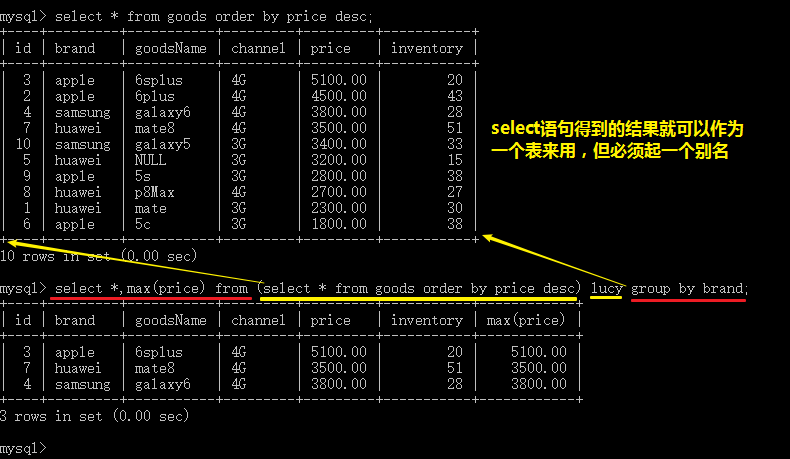


示例：多字段排序



扩展(了解)：





9、【limit 子句】

语法：

limit 【offset,】rows

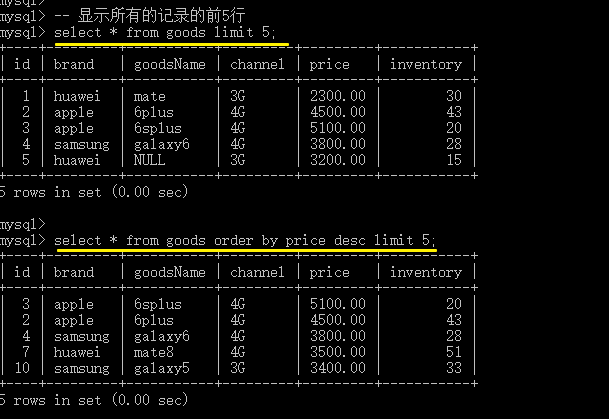
说明：

limit是对面where子句、group by子句、having子句、order by子句得到的结果进行一个显示行数的一个限制。

offset 偏移量，第1条记录偏移量为0，第2条记录偏移量为1，依此类推。如果省略表示0

rows 显示的行数

示例：



数据分页：

分页相关的因素：

每一页显示的记录数： rowsPerPage 人为设定的

当前的页码数： curPage 用户所点击的页码

假设

rowsPerPage=3 表示每一页显示3条记录

curPage 1 2 3 4 5 …

数据分页的公式：

select \* from goods limit (curPage-1)\*rowsPerPage,rowsPerPage;

二、联合查询

语法：

select语句A

union 【all|distinct】

select语句Ｂ

说明：

分表存储联合查询。为什么要分表查询，为了解决查询1条记录更快，但是查询单条记录快了，解决解决所有呢，所以才就出现联合查询的语法。

【all|distinct】 union选项

all 表示所有

distinct (默认的)表示去重

在联合查询中的两条select语句，所查询出来的字段的个数必须一致。

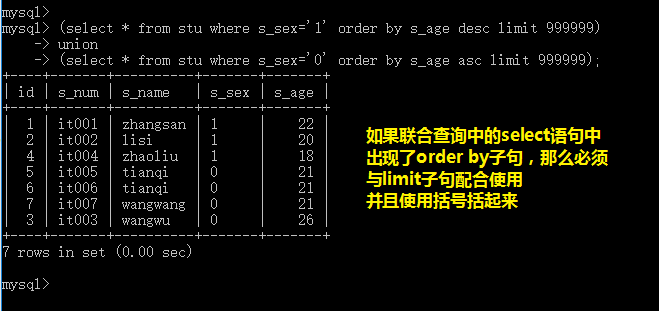
示例：



联合查询的注意事项：

联合查询还可以解决，对同一个表的不同部分进行不同的操作。

如果联合查询的select语句中有order by子句，那么必须配合limit使用。



三、存储引擎(了解)

create table 表名(表的定义)【表选项】

【表选项】

charset 存储字符集

collate 校验集

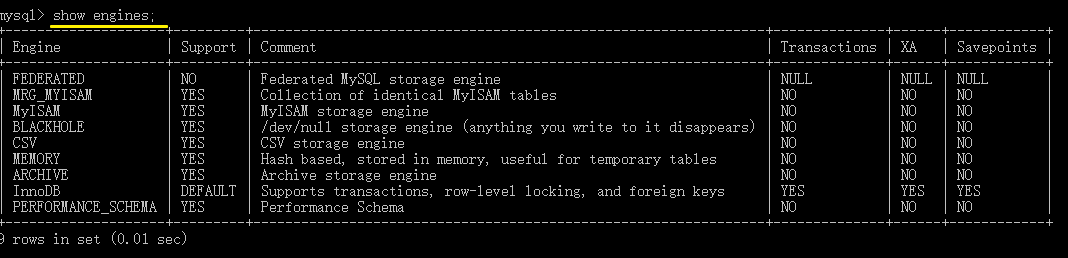
engine 存储引擎

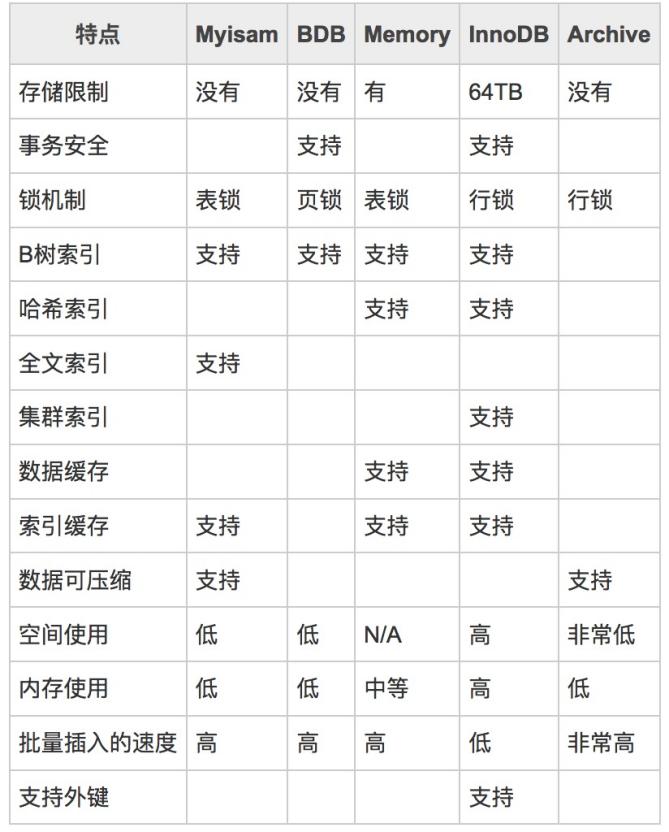
存储引擎：

汽车的引擎是用于驱动汽车运行的。而MySQL的存储引擎，用于驱动数据的从硬盘进行读取或写的。

查看存储引擎：

语法：  
 show engines;



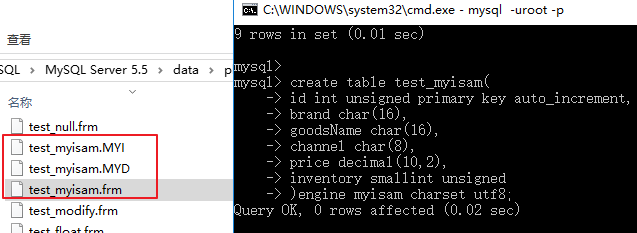


MySIAM

不支持事务安全

不支持外键

MySQL低版本默认的存储引擎



如果是MyISAM存储引擎，那么在创建一个表时会创建三个文件。

.frm 表的结构文件。

.MYD 存储的是表的中的数据

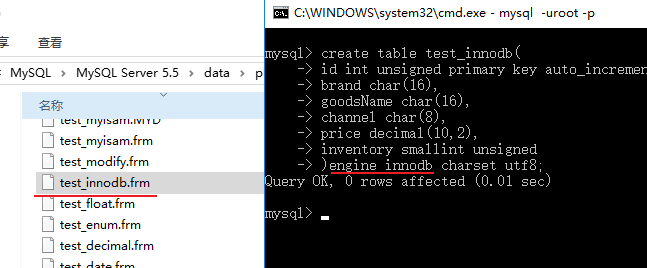
.MYI 存储的表的索引。

InnoDB

支持事务安全

支持外键

MySQL高版本默认的存储引擎



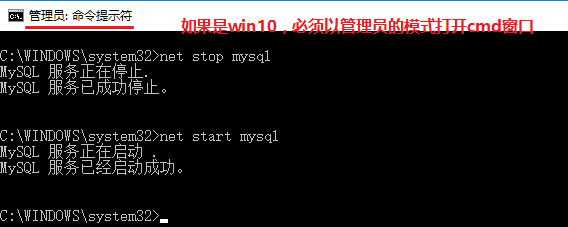
如果使用innodb存储引擎，在创建一个表时，只会有结构文件，其他的数据文件与索引文件放在ibdata1中。

Window的对系统服务的维护。

window的net命令，运行环境是cmd窗口

net stop 服务名

net start 服务名

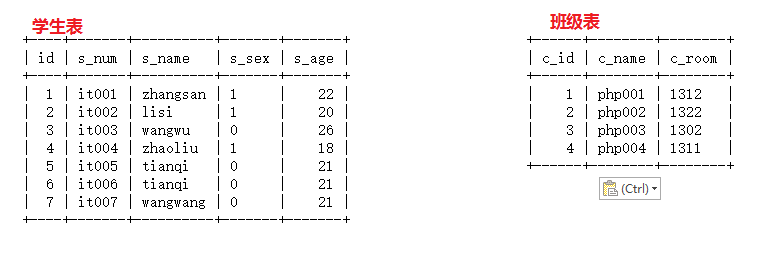


一、外键

1、业务场景

表关系：就是告诉大家在为某个项目创建数据表时，一个数据表就存储一个实体的信息。

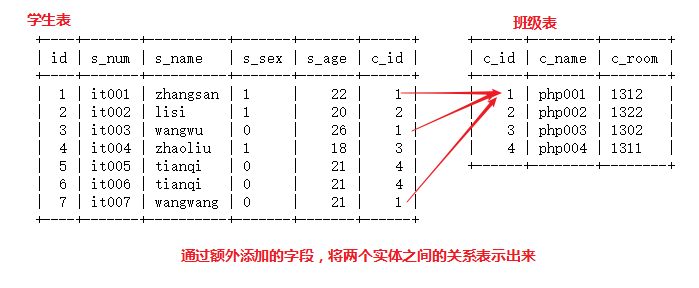
根据表关系的学习，我们将两个实体分别存在两个表中

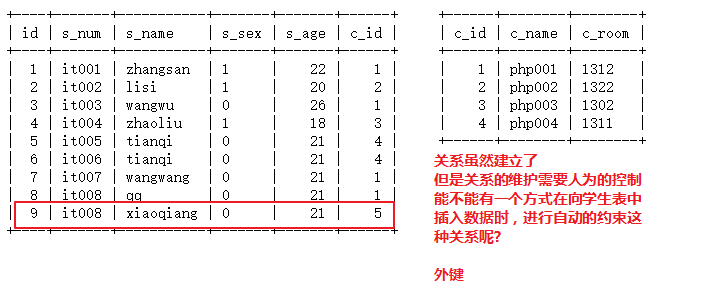


现实中学生与班级之间具有关系，如何在数据表中之间也将这种关系表示出来呢？

解决方案：

在学生表中添加一个字段，用于存储这个学生所属的班级的id，这样的话就建立了表与表之间关系，也就是学生与班级之间的关系。





2、外键

表B的某个字段的取值，是表A中主键字段值中的1个。我们就称这个字段就是外键字段。

表B中被表A中约束的字段就是外键字段。

表B称之为从表

表A称之为主表

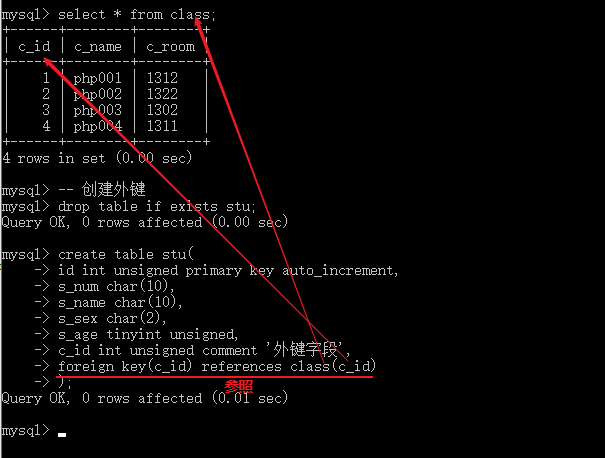
3、创建外键

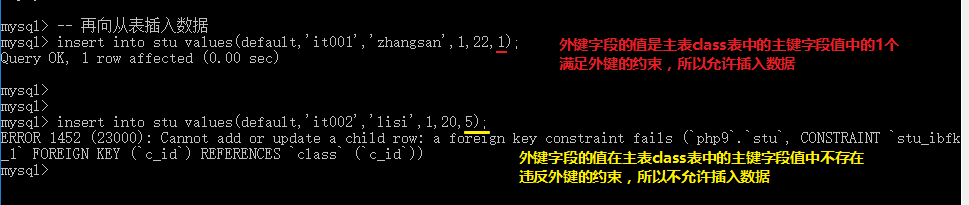
要创建外键，必须创建主表

语法：

foreign key(外键字段) references 主表(主键字段)

示例：





提示：

外键的主要的作用就是建立一种强制的约束。

4、外键的约束

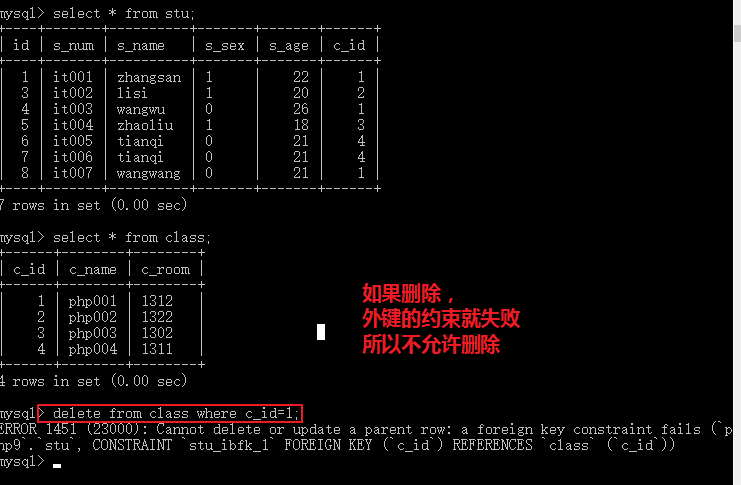
①、从表被主表约束

外键字段的类型必须与主表中的主键字段的类型一致。

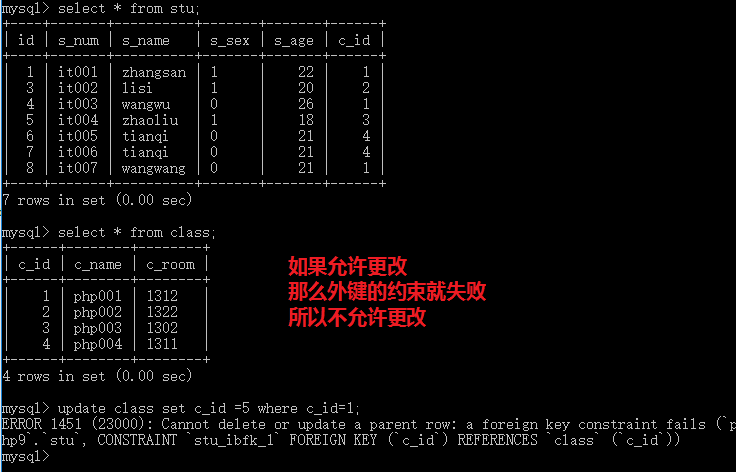
从表中外键字段的值必须是主表中主键字段值中的1个。

②、主表被从表约束

不能随意的删除主表中的记录



主表中主键字段的值也不能随意的更改



③、级联约束与置空约束

默认的主表中的记录不能随意删除，主表中的主键字段的值也不能随意修改，但实际开发中确实有这种需求。那么如何解决？级联约束与置空约束就是解决这种需求。

on update cascade

当主表的主键字段的值被修改后，从表中与之相关的外键字段的值也同时修改。

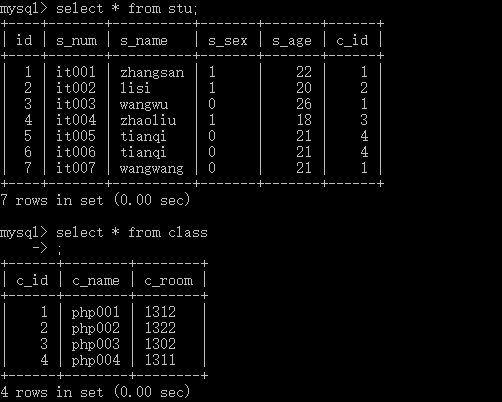
on delete set null

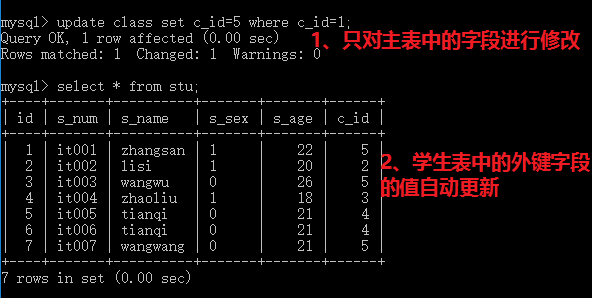
当主表中的记录被删除时，从表中的与之相关的外键字段的值设置为null

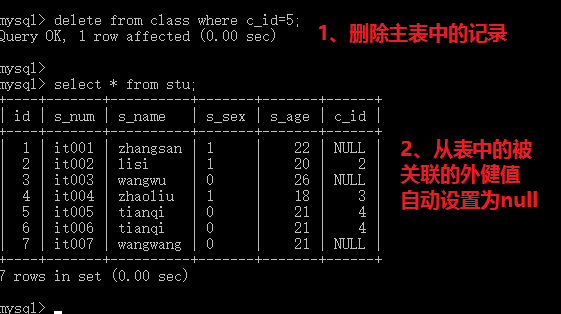
示例：设置级联与置空约束



测试：

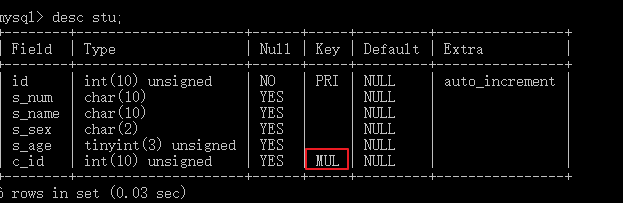






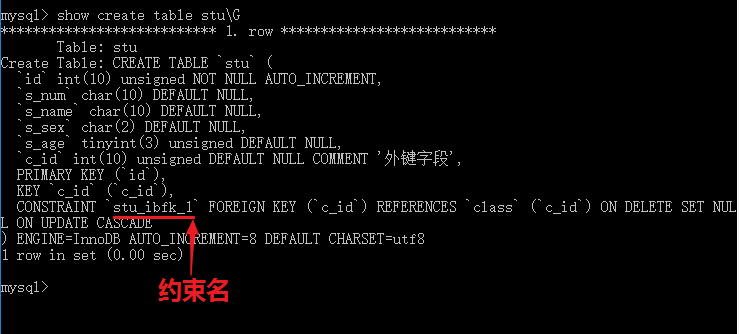
5、删除外键

查看外键：

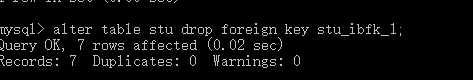


语法：  
 alter table 表名 drop foreign key 外键的约束名。

查看外键的约束名：



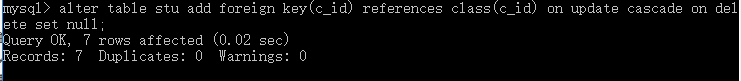
删除：



6、添加外键

语法：

alter table 表名 add foreign key(外键字段) references 主表(主键字段) 【on update cascade】【on delete set null】



二、子查询

1、定义

在一个select语句中包含另一个select语句，被包含的select语句就是子查询。

2、子查询的分类

有两种分类方式：

根据子查询的出现的位置

where子查询

from子查询

根据子查询查询出来的数据

标量子查询

列子查询

行子查询

以上三个子查询都是where子查询

表子查询

表子查询是from子查询

3、子查询详解

①、理解where子句的表达式

where表达式的本质，就是字段名与值的等值匹配情况，有以下几种：

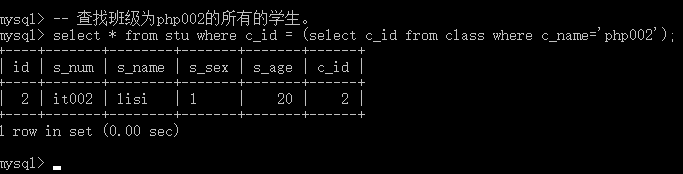
|  |  |  |
| --- | --- | --- |
| 字段个数 | 值的个数 | 需要使用的运算符 |
| 1 | 1 | = |
| 1 | 1 | In |
| 1 | n | in |
| n | n | (字段列表)**=**(值列表) |

②、标量子查询

子查询的书写方式：先写最终想得到的那个select，另一个子查询其实就是where表达式中的一个值。

查询出来的结果是1行1列。

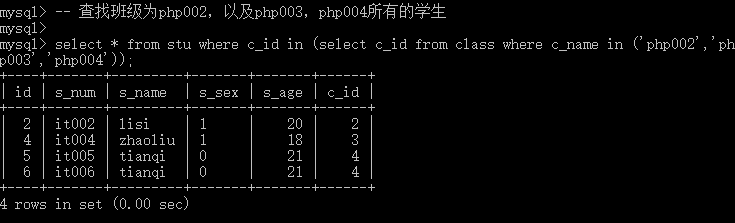
示例：



③、列子查询

子查询得到的结果是1列多行

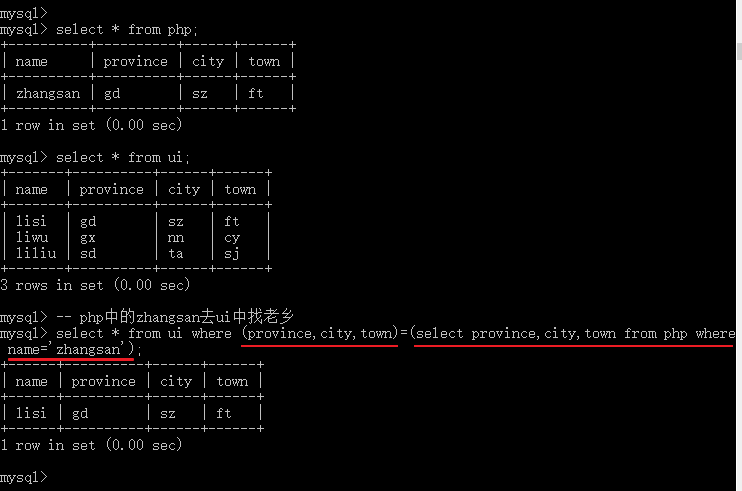
示例：



④、行子查询

子查询得到的结果1行多列。

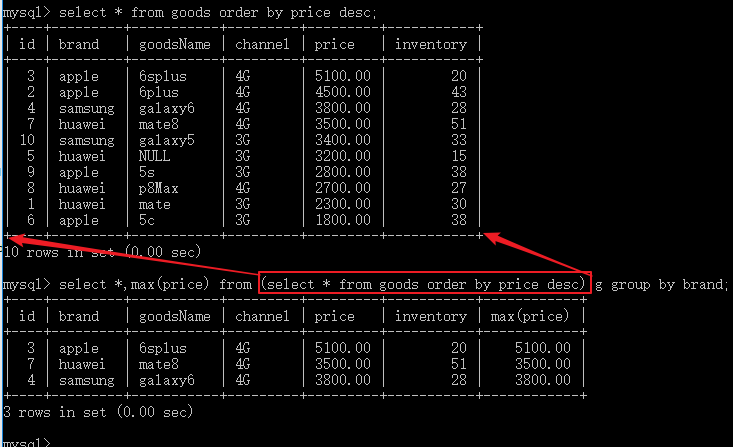
找老乡



提示：  
 以上是where子查询

⑤、表子查询

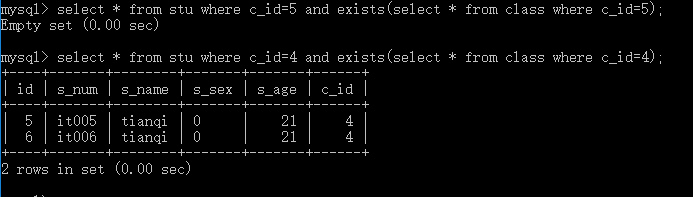
子查询得到的数据是多行多列，主要用在from后作为一个数据源。



**提示：**

**所有的子查询都必须使用括号括起来，但是只有表子查询必须使用别名。**

⑥、exists子查询



4、all、any、some

主要用于where表达式中的不等值比较。

all与any、some主要用于一个字段与多个值的不等值比较。

all 所有

any 任何一个 some(是any的一个同义词)

字段 >all(多个值) >最大的 一个字段多个值的and max()

字段 <all(多个值) <最小的 一个字段多个值的and min()

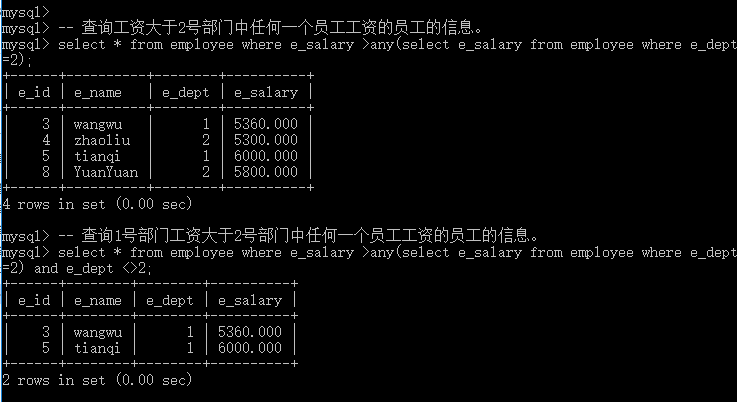
字段 <>all(多个值) not in

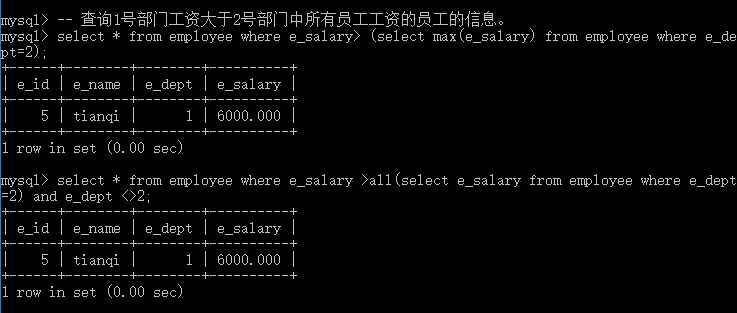
字段 >any(多个值) >最小值 一个字段多个值的or min()

字段 <any(多个值 ) <最大值 一个字段多个值的or max()

字段 =any(多个值) in

示例：





三、连接查询

子查询的特点：虽然涉及到多个表的查询，但最终得到的结果数据只来源于一个表的数据。

连接查询的特点：最终数据是来源于多个表。

1、内连接

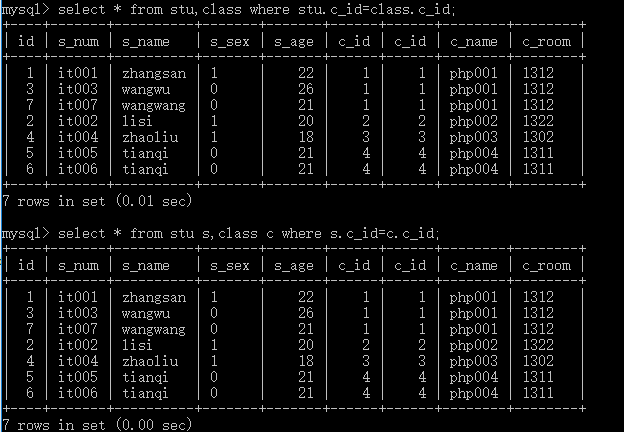
①、隐式内连接

语法：

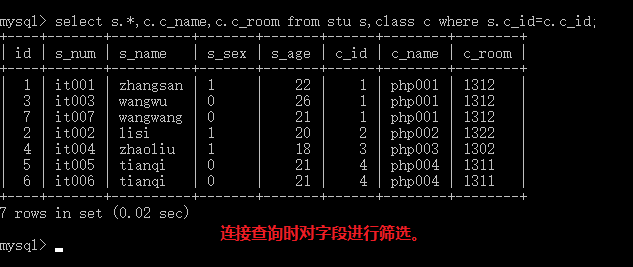
select \* from 表A,表B 连接条件;

说明：

默认多表数据源产生的结果是迪卡尔积，但如果指定连接条件，那么就可以将有意义的记录筛选出来。



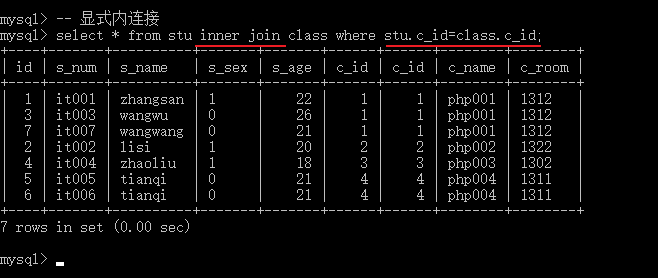
示例：



②、显式内连接

语法：  
 select \* from 表A 【inner 】 join 表B 连接条件;

示例：



提示：

内连接的特点，得到结果就是两个连接的表中匹配成功的记录，对于匹配不成功的丢弃。

内连接也称之为等值连接，根据连接条件相等的记录才保留。

2、外连接

①、左外连接

语法：

select \* from 表L **left join** 表R **on** 连接条件

②、右外连接

语法：

select \* from 表L **right join** 表R **on** 连接条件

说明：

无论是左外连接还是右外连接，匹配成功的记录都会被保留。

左外连接与右外连接的匹配是在匹配不成功时的处理方式。

匹配不成功有两种情况：

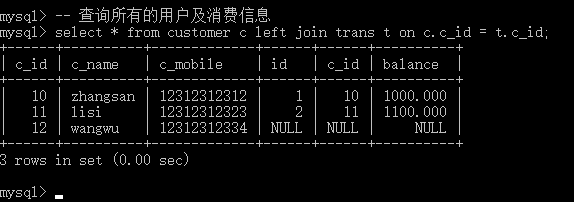
左表中的记录在右表中找不到匹配的记录。

右表中的记录在左表中找不到匹配的记录

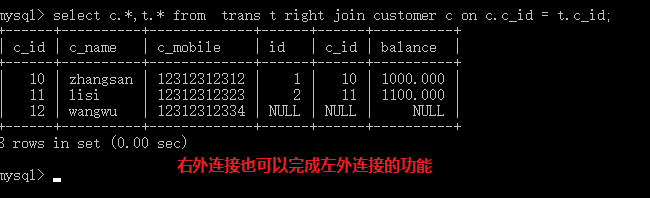
对于左外连接，会保留左表中匹配不成功的记录。

对于右外连接，会保留右表中匹配不成功的记录。

示例：



示例：



小节：

内连接，只保留匹配成功的

外连接，不但保留匹配成功的，还会保留匹配不成功。

在左外连接与右外连接，主要体现在匹配不成功的记录上。

3、自然连接(扩展了解）

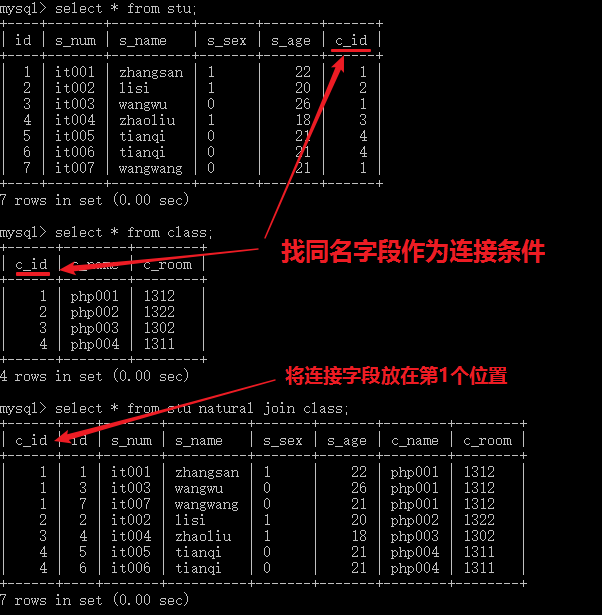
自然连接就是自动匹配，**不需要人为的指定连接字段**。自然连接会自动找进行连接的两个表中**同名的字段进行连接**。

自然连接就是对内连接与外连接的一种简化。

①、自然内连接。

语法：  
 select \* from 表A natural join 表B;

示例：



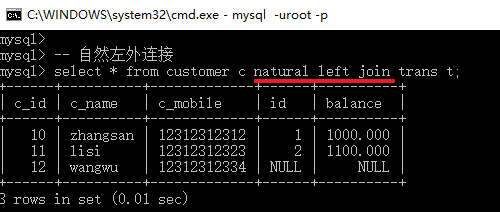
②、自然左外连接

自然左外连接就是对左外连接的简化。

语法：

select \* from 表L natural left join 表R;

示例：



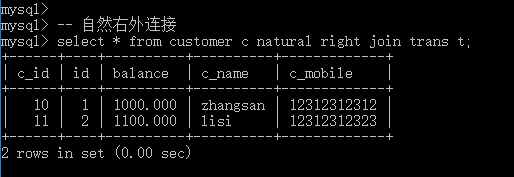
②、自然右外连接

自然右外连接就是对右外连接的简化。

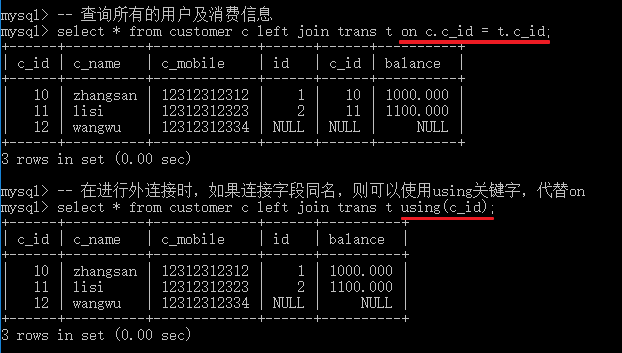
语法：

select \* from 表L natural right join 表R;

示例：



using关键字



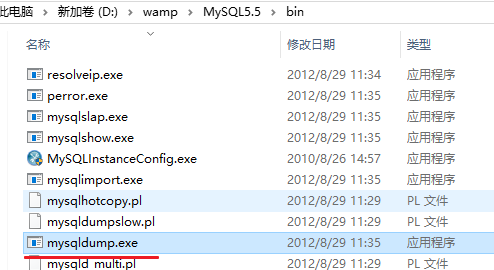
四、数据备份

1、sql备份与还原

①、sql备份

我们是想对数据进行备份，任何一个时刻，数据库的数据表中数据都可以使用一条insert语句来表示。sql备份备份的是sql语句。

备份工具是mysqldump.exes

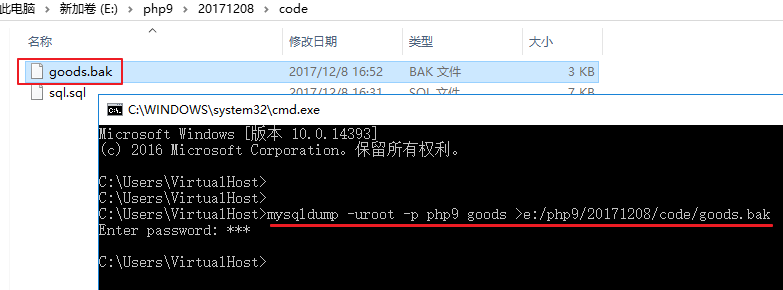


运行环境在cmd窗口

语法：

mysqldump –uroot –p 数据库名 【数据表】> 备份的文件名

示例：

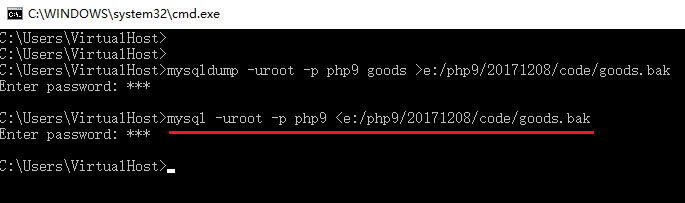


②、还原

工具mysql.exe

语法：

mysql –uroot –p 数据库 < 还原的文件(备份文件)



2、数据备份

①、备份

语法：

select 字段名 from 表名 where 条件 into outfile ‘备份文件’ 【fields选项】【lines选项】

说明：

fields用于设置字段的相关信息

以fields开头

terminated by 设置字段与字段之间的分隔符 默认为tab

enclosed by 设置字段的包裹 默认是空字符’’

escaped by 设置null值使用什么代替 默认\N

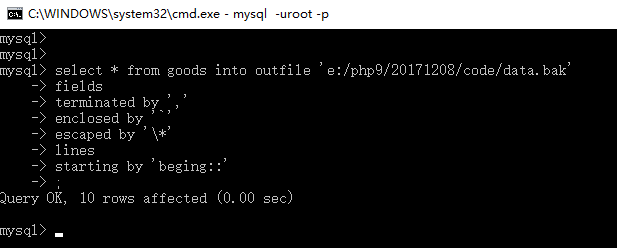
lines用于设置记录的相关信息

以lines开头

starting by 设置一行的开始符号

terminated by 设置一行的结束符号 默认\r\n

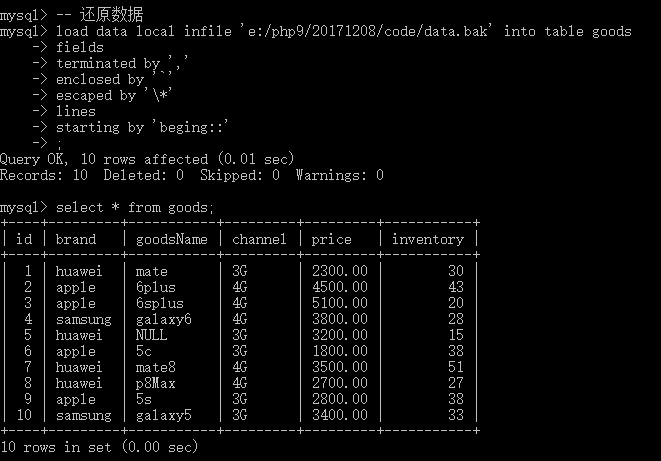
示例：



②、还原

语法：

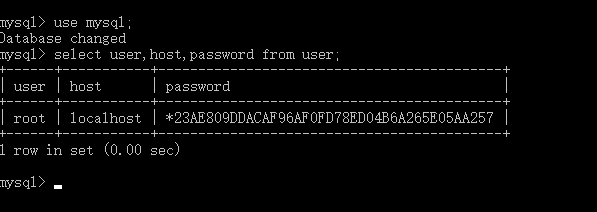
load data local infile ‘还原的文件名’ into table 表名 【fields选项】【lines选项】



五、用户管理

1、查看用户

数据库中的用户数据是保存在mysql数据库中的user数据表中。



2、创建用户

语法：

create user ‘登陆名’@’标识’ identified by ‘密码’;

说明：

登陆名 用户名

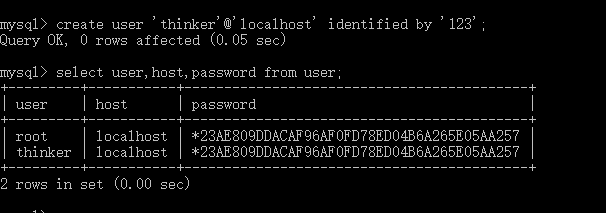
标识 作用是用于标记这个用户所能登陆位置，取值：

localhost 只能在本机登陆

ip地址 220.231.114.115

% 不限制登陆位置

示例：



3、授予权限

语法：

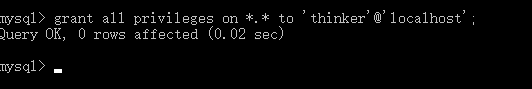
grant 权限或all privileges on 数据库名【.数据表名】 to ‘登陆名’@’标识’

说明：

all privileges 所有的权限

权限 create、drop、alter、select、update、insert……

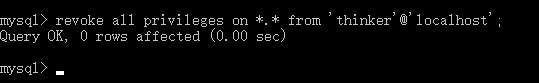
示例：



4、收回权限

语法：

revoke 权限或all privileges on 数据库名【.数据表名】 from ‘登陆名’@’标识’



5、删除用户

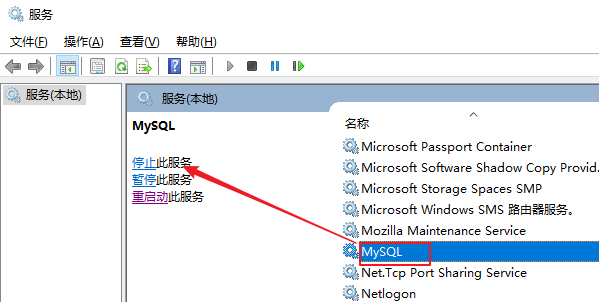
语法：

drop user ‘登陆名‘@’标识’



6、重置root密码

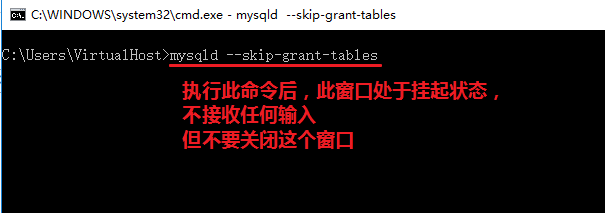
①、停止mysql服务



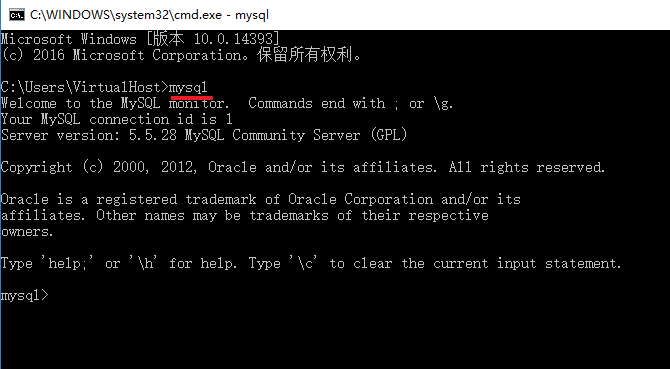
②、启动

使用mysqladmin.exe工具来启动mysql服务，mysqladmin.exe工具在启动msyql时可以跳过授权表。

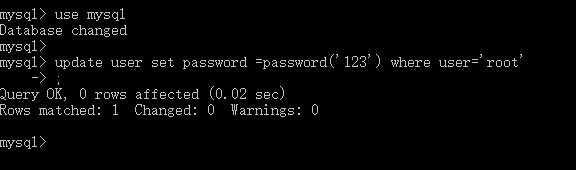
语法  
 mysqld --skip-grant-tables



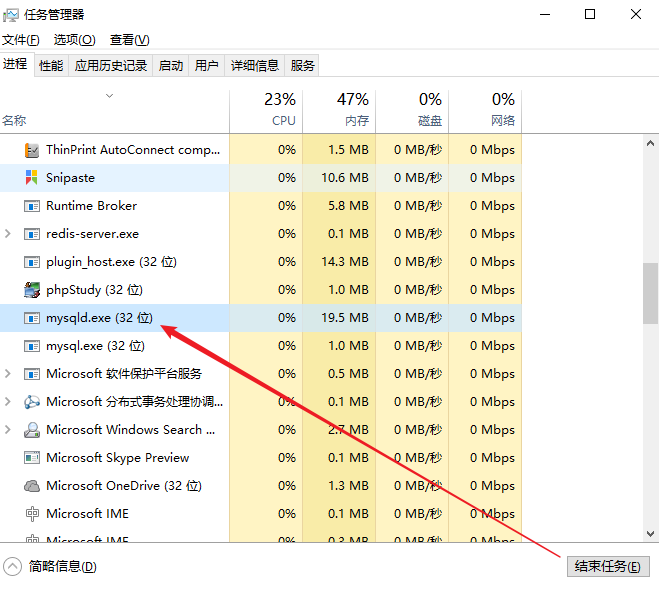
③、登陆mysql



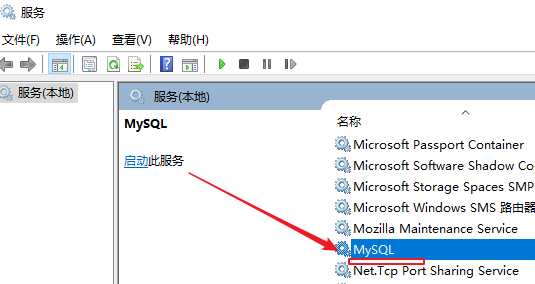
④、修改密码



⑤、关闭第2步中处于挂起的窗口，并结束mysql服务。



⑥、重启mysql服务



⑦、使用新密码进行登陆