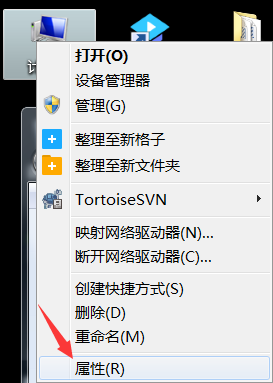
# 第三阶段

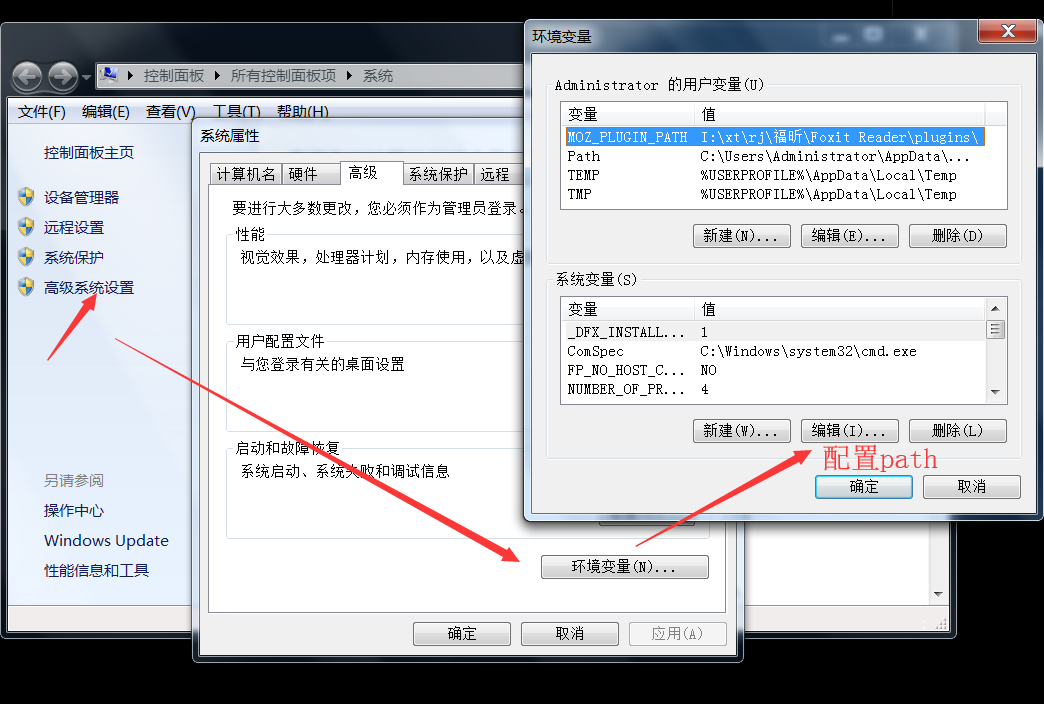
## Day17 MySQL入门

### 1．环境变量:win7环境:

1．1. 打开系统界面

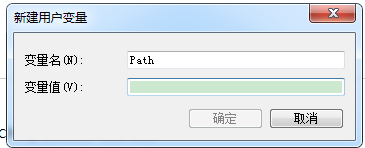


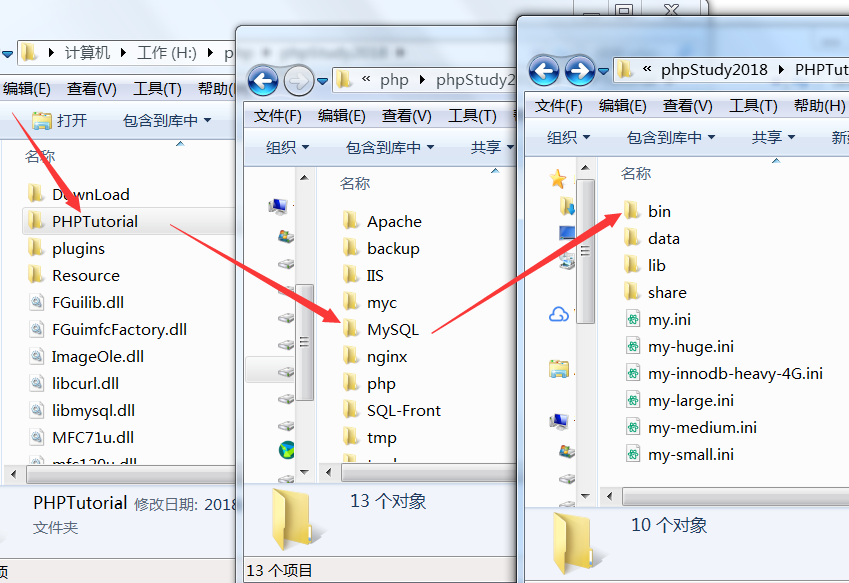
1．2. 进入环境变量

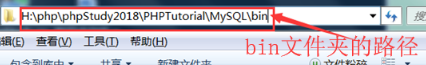


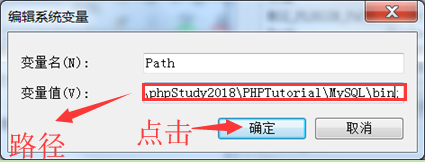
1．3. 配置环境变量

如果没有可以点击新建,变量名写Path, 变量值为MySQL下的bin文件夹的地址:



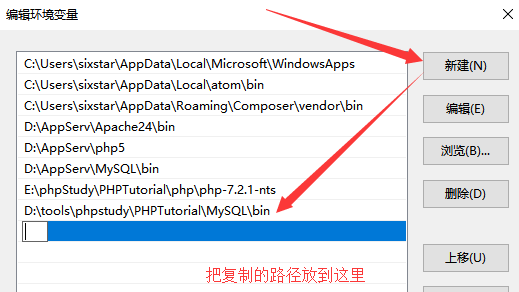






Win7 的看前面的环境变量后面有没有分号,没有就要加上分号。注意不要修改前面的环境变量的值。

Win10的可以直接复制到这里就可以了:

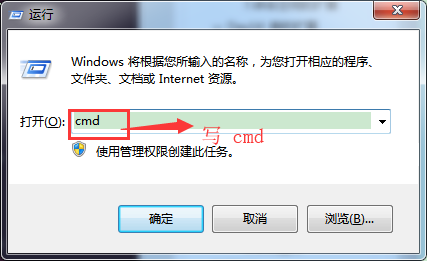


### 进入命令行

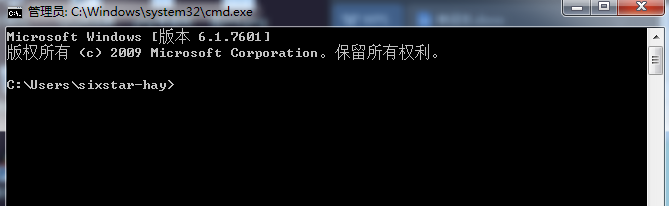
第一种方式:



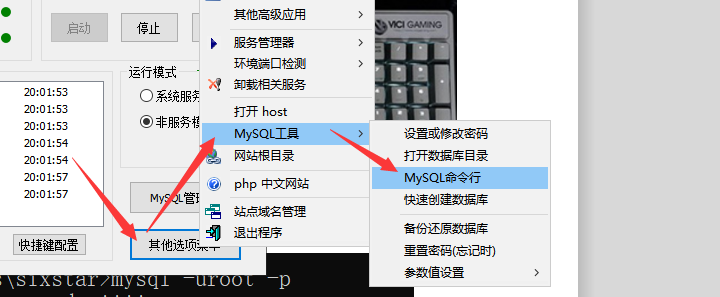
点击win+r键进入运行



点击确定进入命令行:



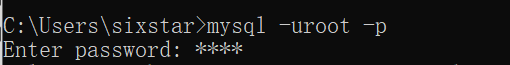
第二种方式(直接进入数据库,不需要环境变量):



### 进入数据库的几种方式

1. 常用方式(需要使用环境变量,用户名和密码默认是root):

进入数据库:Mysql -uroot -p



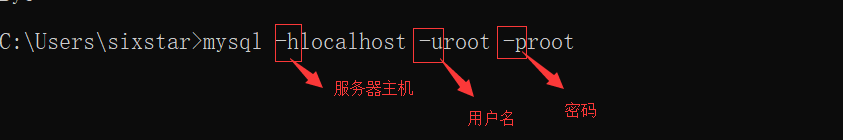
2.使用完整的路径(不需要使用环境变量):



3.直接使用phpstudy的命令行,输入密码(结合上面进入命令行的第二种方式使用):

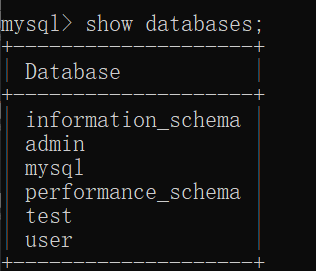


4.使用完整的格式进入(各个符号表示的意思);

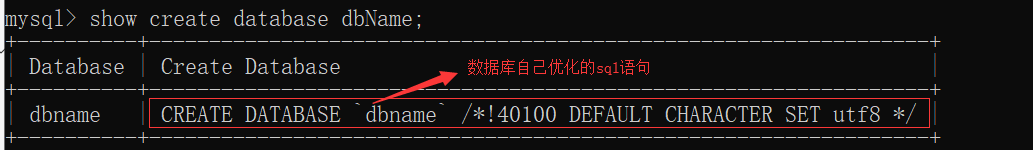


### 4.库操作

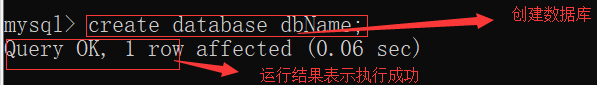
1. 查询数据库 show databases;



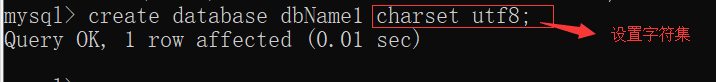
查询字符集: show create database dbName;



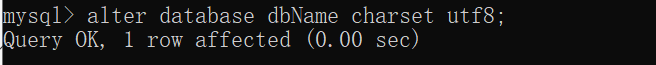
1. 创建数据库 create database dbName;



创建数据库 设置字符集 create database dbName charset utf8;

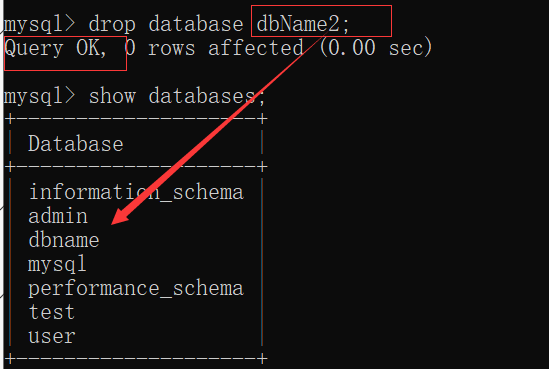


1. 修改数据库: alter database dbName charset utf8;



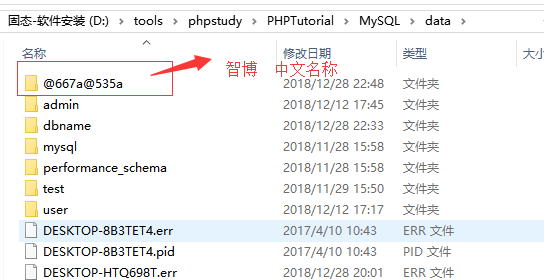
注意修改的是字符集;

1. 删除数据库: drop database dbName;

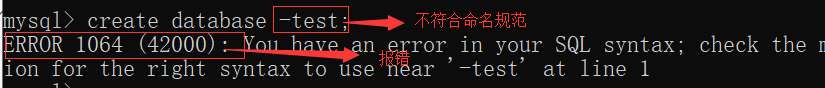


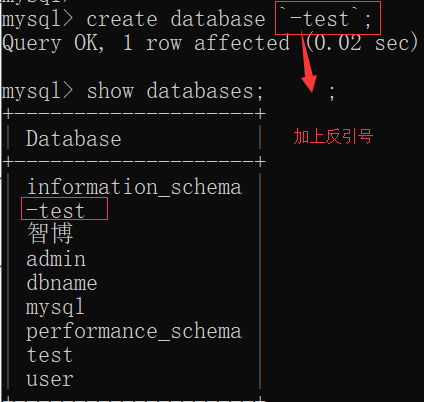
补充:

注意:中文的命名不建议写,如果不设置字符集会报错,创建后在data文件中不会显示中文(data文件夹中就是数据库存储的信息)



数据库的命名也要符合命名规范,如果碰到不符合的情况可以使用反引号 ·· (tab上面的键) 来访问:





## Day18表与数据的简单操作

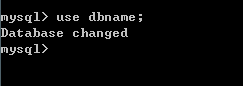
进入数据库的第一步:设置字符集 set names gbk;



1.表操作

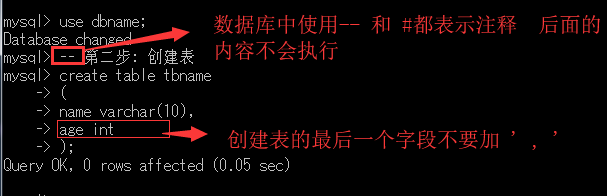
1.1选择数据库:

选择了数据库之后,就是对该数据库进行操作



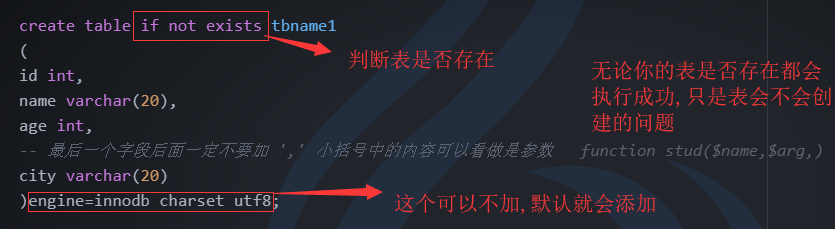
1.2 创建表

第一种常用创建:

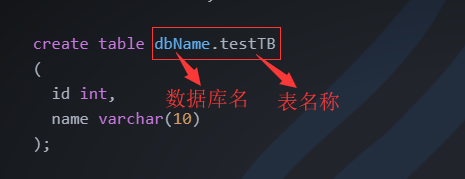


注释: ‘ -- ’ 和’ # ’都表示注释;

第二种 判断表是否存在创建:



第三种 在没有选择数据库的时创建表 (补充,了解即可,一般不用):

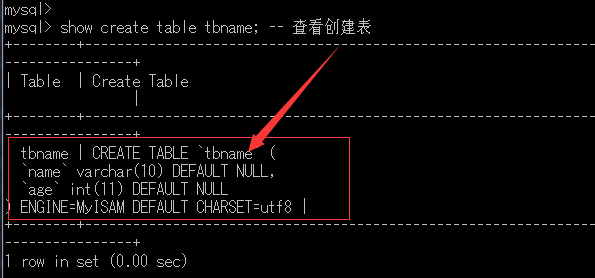


1.3 查看表

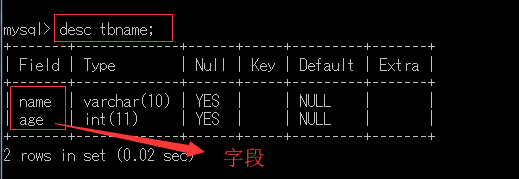
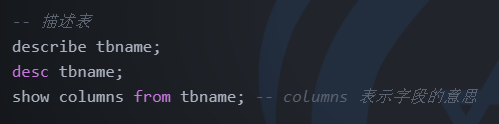
第一种:查看所有的表:



第二种:查看表的创建

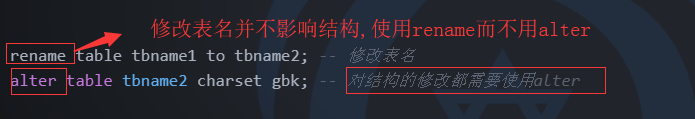


第三种:查看表的字段

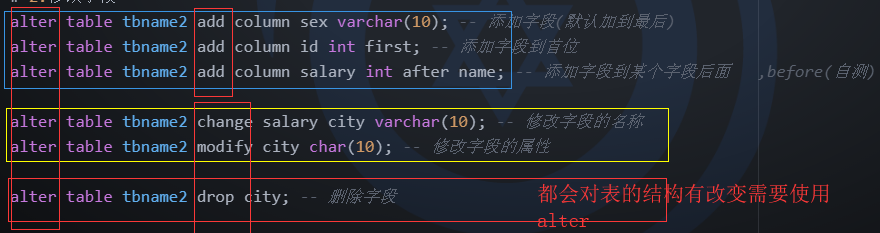


1.4修改表

1).修改表本身

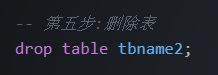


2).修改表的字段(字段的增删改)



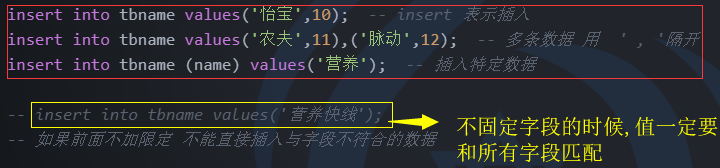
1.5 删除表

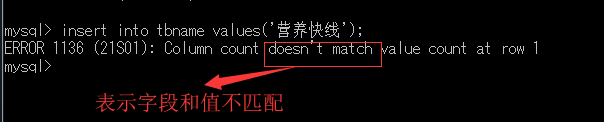
操作要谨慎！！！



2.数据操作

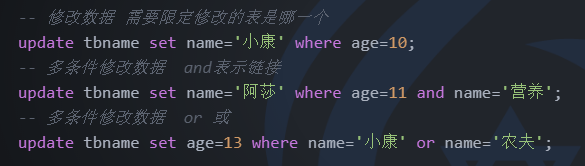
2.1新增数据(插入数据)





由于数据库的报错可能会不够清晰,因此从数据库的报错很难直接找到问题,需要细心以及测试,建议可以在记事本或编辑器上写SQL语句,然后复制过来测试,避免重复编写浪费时间。

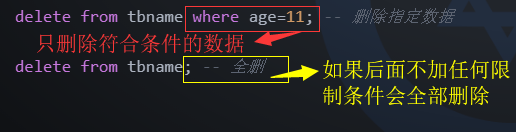
2.2 修改数据



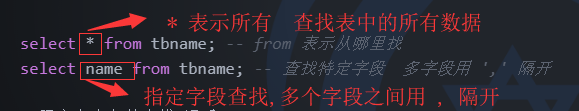
and 表示连接,并且,如果使用,必须多个条件都满足才能符合条件

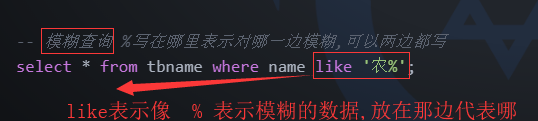
or 表示或者,当数据中有一个符合该条件的都可以执行

2.3 删除数据



2.4 查看数据





‘ \_ ’ 也表示模糊查询,但是他只能表示单个的字符,而’ % ’可以表示所有的字符,所以一般不用 ‘ \_ ’ ,了解一下就可以了。另外数据表也可以使用模糊查询。

3.数据类型

3.1整型

tinyint -- 迷你整型 1 字节 -2 ^7 ~ 2^7 -1 -128~127

当你设置该字段无符号的时候 最大值会变成 255 (unsigned)

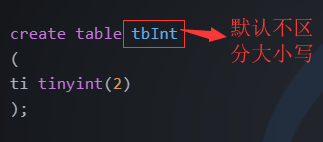
smallint -- 小整型 2字节 -2^15 ~ 2^15 -1 -32,768

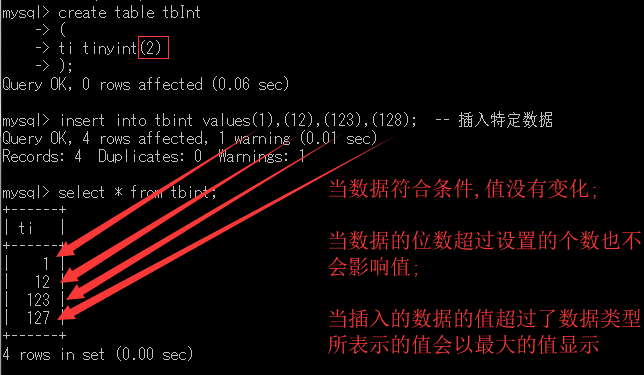
mediumint -- 中等整型 3字节 -2^23 ~ 2^23 -1 -8388608

int -- 整型 4 字节 -2^31 ~ 2^31 -1 -2,147,483,648

bigint -- 大整型 8字节 -2^63 ~ 2^63 -1 -9223372036854775807

记住大概范围就可以了



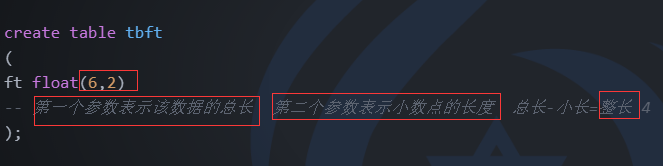


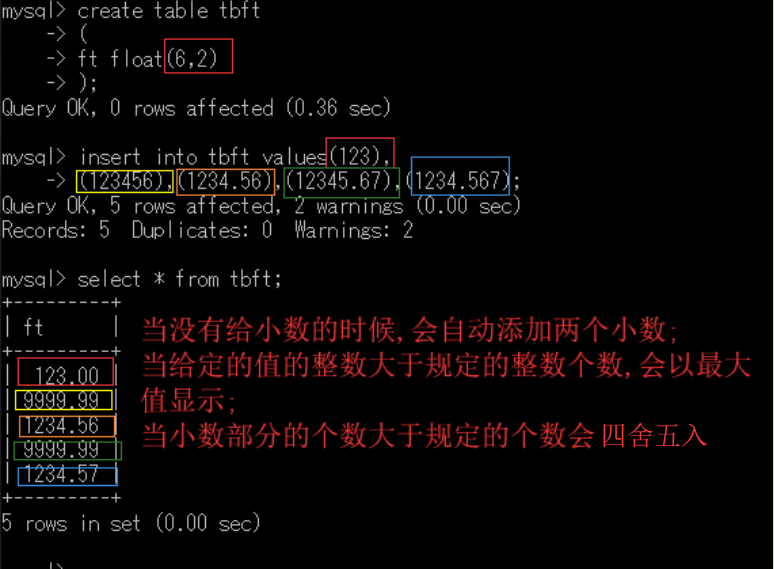
整数型数据类型后面小括号的值表示的只是一种格式,不会影响值的大小。而字符串后面的值则是字符的个数,不要记混了。

3.2 小数型

float 单精度 4字节表示,包含小数的一定是小数型; 超过整型范围的也是小数型。

double 双精度 8字节

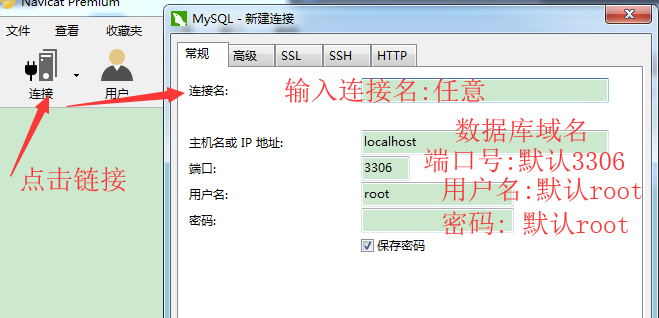


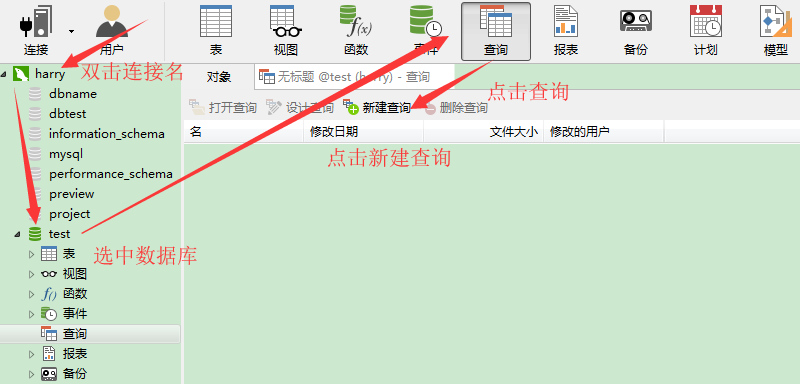


小数型括号中的第一个参数表示该数据的总长 第二个参数表示小数点的长度 而 整长=总长-小长

## Day19

### 1.navicate的基本使用







### 2.时间类型

datetime 时间型 完整的时间 2019-01-01 12:00:00

date 日期型 日期(不包括时间) 2019-01-01

time 时间型 时间段,表示从以前的某个时间到现在的某个时间 可以为负数

TIMESTAMP 时间戳 不是真正完整的额时间戳,时间不以毫秒计算(数据库)

时间戳 : 从格林威治的1970-现在的时间以毫秒来计算(时间戳)

year 年 可以有参数,2或者4 分别表示年份的最后两位以及完整的格式



不同的时间类型显示的时间格式都不一样。

### 3.字符串类型

char varchar enum set text

3.1 char与varchar

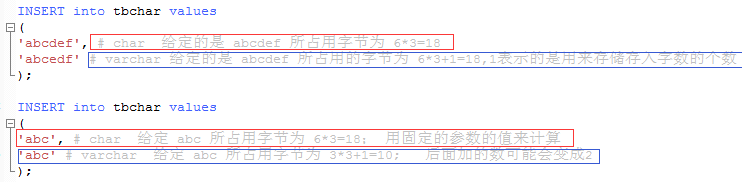
Char : 定长字符串, 给定参数为多少,就一定会占据给定的值

Varchar : 变长字符串, 给定多少不一定全部用, 按照需求占据空间

text能存储字符的量非常大,一般不会用到 ,

utf8字符集下cahr大概存储255个字符, varchar大概存储21845个字符

字节与字符的关系 在gbk下 : 1字符=2字节; utf8下 : 1字符=3字节



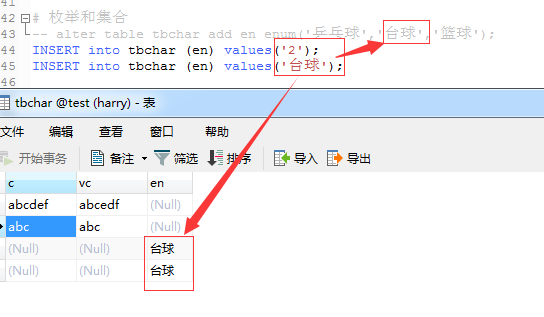
一个字节能存储的值最大为255,当你的字符的值超过255的时候需要使用2来存储字符的值。

当你获取varchar里面的值,先要读取加上的一个 '1' 来了解我存储了几个字符 ,然后才能读取对应的值 ,相对来说降低了效率。

Cahr一般用在字符个数确定的地方:比如身份证号,手机号等。

3.2 enum与set

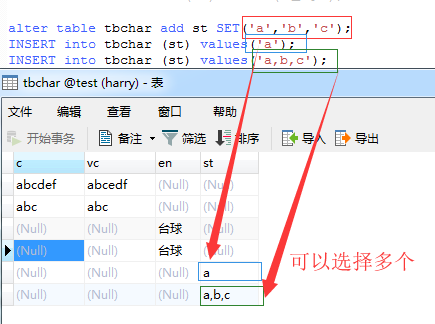
enum 枚举 : 插入的数据必须是 枚举设置的值中的一个,不能超出,否则插入不了, enum是单选按钮,只能选一个。



枚举创建完成之后 数据库会对其进行自动保存,保存之后会给特定的标签来获取,这里的2就是对应台球的值。

为什么给定额外的标签来存取:因为一个数最多占3个字节,'乒乓球' 会占用9个字节,存的值越多节省的空间越多。在保存值的时候,有一个保存操作,会浪费时间,降低了效率。

SET set集合是一个复选框,可以选择多个值 :

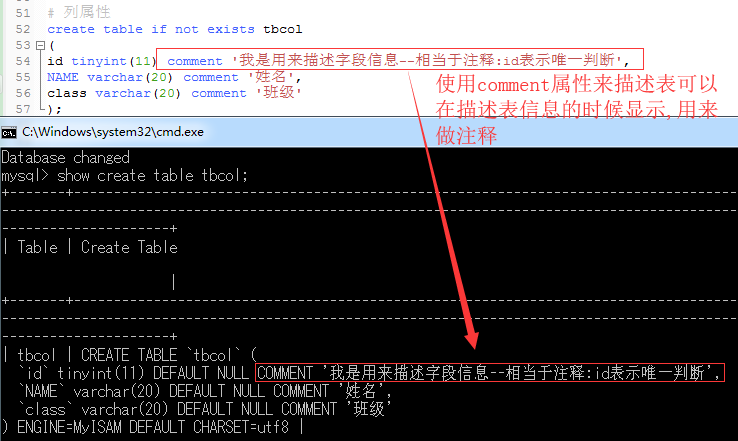


了解: 当有多个值被选中的时候,会以’ 1 ’来标识,没有被选中的会被’ 0 ’标识,然后翻转过来转换成十进制就能选中多个。

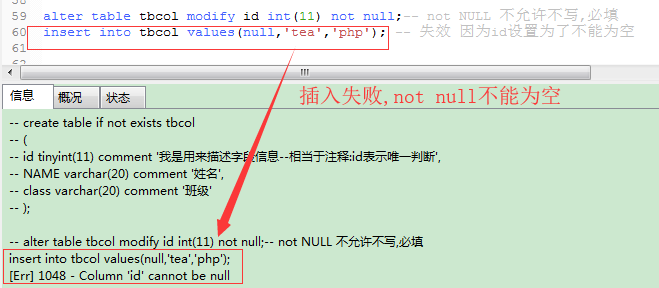
### 4.列属性

列属性: 字段的属性。

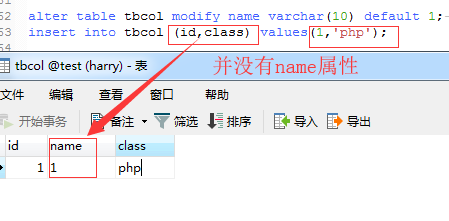
Comment描述表信息:



Null/not null:



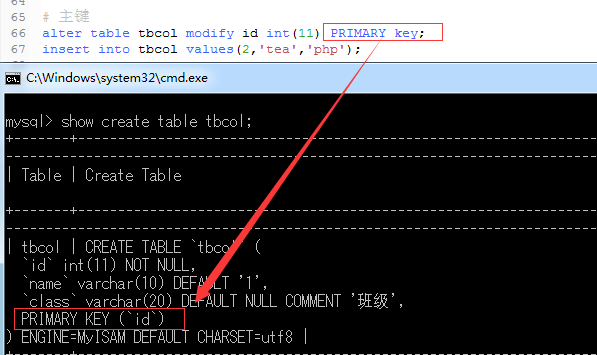
Default:

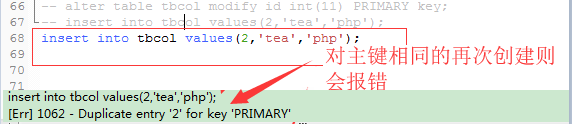


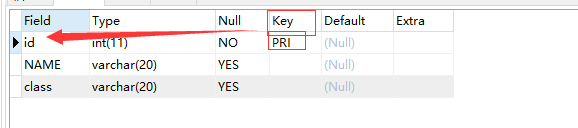
### 5.主键 primary key

字段中能唯一确定数据的字段:设置了该键为主键后,该键不能重复,如果重复则无法插入该数据。

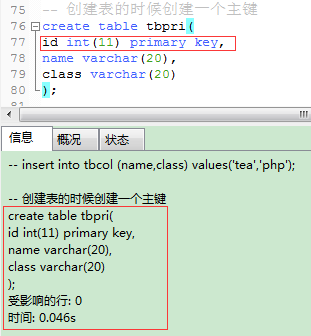
第一种: 创建表之后创建主键







第二种: 创建表的时候创建主键



第三种: 多个字段创建为一个主键



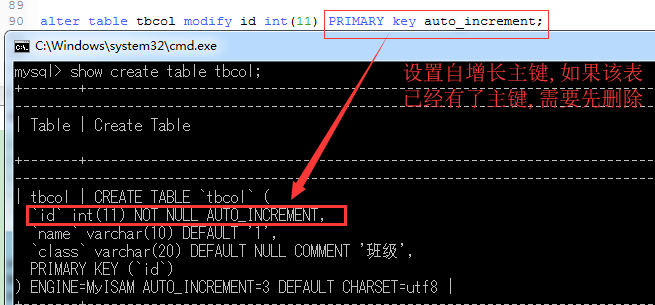
也可以在创建完表之后创建多字段主键。

虽然主键字段有两个,但是主键只有一种,只有两个字段的值都一样,才是同一主键;也就是说多字段主键键可以是单一的值不一样。

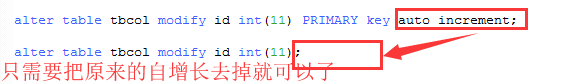
unique key 用法和主键一样,表示唯一键,唯一键可以存在多个,数据也不能重复,但是可以为空。一个表只能存在一个主键,但可以存在多个唯一键。

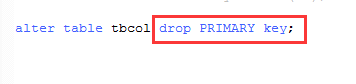
### 6.自增长auto\_increment

自增长是相对于主键来说的:



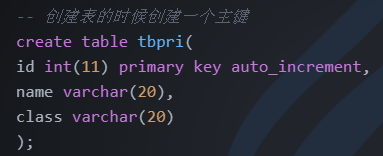
删除主键, 需要先删除自增长。





然后使用drop删除自增长。

创建表的时候创建主键自增长:

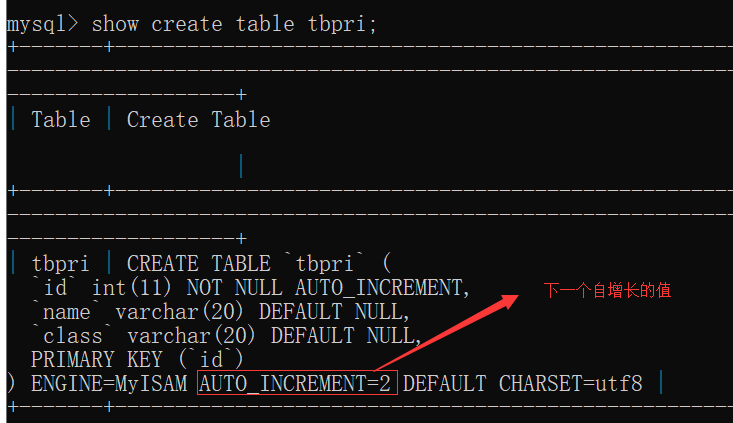


查看自增长的信息:

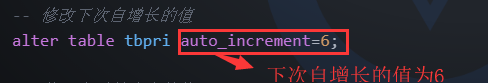


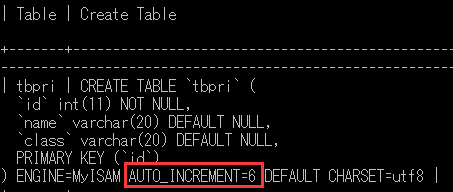


查看下次自增长的值:



修改下次自增长的值:





修改自增长的初始与增加的值:

