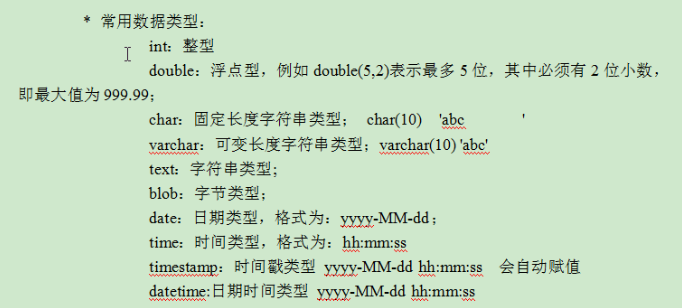
MySQL中的数据类型

# MySQL中常用的常用的数据类型



# MySQL中的数值类型：整型和浮点型

## int类型

有：**tinyint（1个字节最大127=2^7-1）、smallint（2个字节最大2^15-1）、mediumint（3个字节最大2^23-1）、int（4个字节最大2^31-1）、bigint（8个字节最大2^63-1）。**

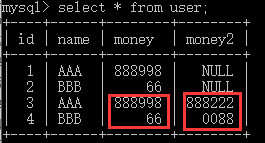
## int() : int后加圆括号的作用：

定义列的数据类型时，int后加小括号如int(6)表示，显示的位数。

**注意**：对于位数多的并没有截断，且没有对原始数据有所修改，如果位数少，即高位为0时,则0不会显示出来。如果要求高位补0的话，在定义列的数据类型的时候，在后面可以添加zerofill。

如：alter table user add money int(4);

alter table user add money2 int(4) **zerofill**;

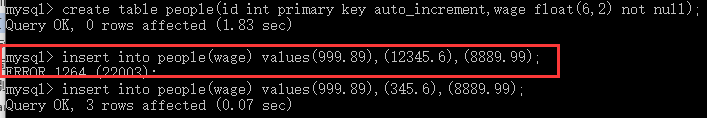


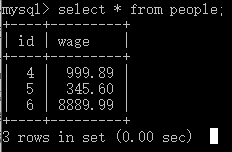
## decimal数据类型

## float（4个字节）和double（8个字节）

**float(m,n)** : m表示所有位数（整数位加上小数位），n表示小数位。

create table people(id int primary key auto\_increment,wage **float(6,2)** not null);

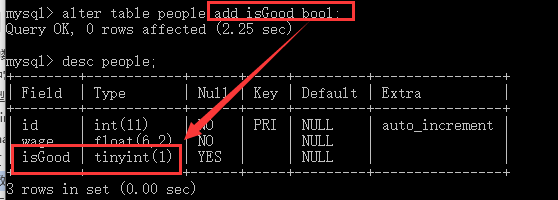


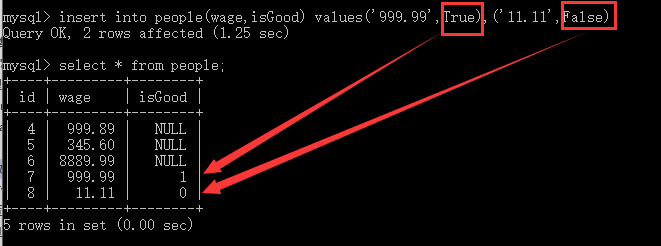


**当数据超过float定义的位数时，会报错。有些版本不报错，对于小数部分四舍五入，对于正数取最大值，一般去99\*\*999。**

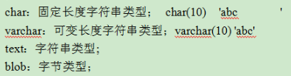
## bool数据类型本质就是tinyint

**True 对应的就是1，False对应的就是0。**





# MySQL中的字符串类型



## char：固定位数的字符串，不足的用空格补充

定义时，一般需要指定长度，否则默认为1.

**alter table user add name2 char;**

**alter table user add name3 char(10);**

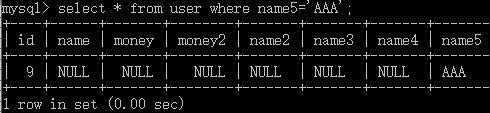


**在后面添加binary，表示该字段内的字符串区分大小写，同样适用于varchar。**

**alter table user add name5 varchar(10) binary;**

insert user(name5) values('aaa'),(‘AAA’)

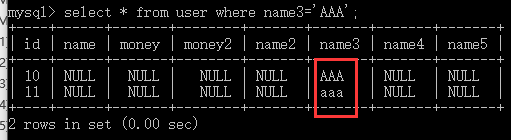
select \* from user where name5='AAA';结果只能选择出一条;



对于name3字段：

alter table user add name3 char(10);

insert user(name3) values('AAA'),('aaa');



## varchar：在指定长度内，长度可变，节省存储空间。

在严格模式下，存储的数据长度不能长于指定长度。

**在后面添加binary，表示该字段内的字符串区分大小写**

## char与varchar的区别：

定义这两个数据类型的时候，需要在后面**添加位数**，如char(100)、varchar(100)。

### 共同点：两者使用中，都不允许超过这个字符的限定，否则都会报出错误：

在**非严格模式**下，可以截取前面的进行存储，不会报错；在**严格模式**下，不能存储，直接报错。



### 区别：char是固定位数的字符串，如果添加位数少于指定位数，则自动利用空字符补齐，如果多于指定位数，则报错。varchar是根据设置的字符保存数据，不会自动添加空字符。

### char的长度是固定的，所以（它的处理速度）性能更高些，但是其缺点就是浪费了存储空间；程序需要对行尾空格进行处理。由于MySQL不断升级，varchar数据类型的性能不断改进并提高，所以平时一般使用varchar较多。

**简单总结： char 性能高，浪费存储空间；varchar 性能低，节省存储空间。**

## text：存放大量文本。还有tinytext。

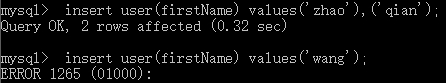
## blob：存放大量二进制数据。还有longblob。

# 枚举类型enum与集合set类型

## 枚举类型enum

定义字段为枚举类型，需要指定枚举的所有元素，插入时，只能插入该枚举中的一个元素。

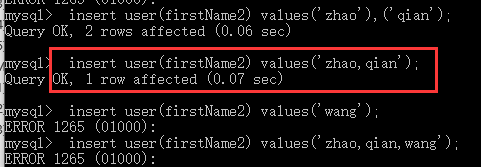
**alter table user add firstName enum('zhao','qian','sun','li');**





## 集合类型：set

定义字段为集合set类型，需要指定集合的所有元素，插入时，可以插入该集合中的一个或多个元素。



## 枚举类型enum与集合set类型的区别

**共同点：都是只能插入限定的元素；**

**不同点：枚举只可以插入一个元素，而集合可以插入一个或多个元素。**

# MySQL中的日期与时间类型

## 简单介绍(主要是5个)

### date：只是表示年月日，格式为yyyy-MM-dd。

### time：只是表示时分秒，格式为hh:mm:ss。

### datetime：表示年月日和时分秒，格式为yyyy-MM-dd hh:mm:ss。

### timestamp:时间戳类型。



**在未赋值时，默认填充是当前时间。**

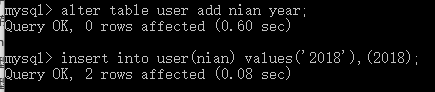
**填充方式1：insert into user(timeStamp) values('2017-08-08 12:12:12')**

**填充方式2：insert into user(timeStamp) values('2017-08-08 12:12:12'),(20180808121212);**

### year：表示年，YYYY

如：alter table user add nian year;

insert into user(nian) values('2018'),(2018);

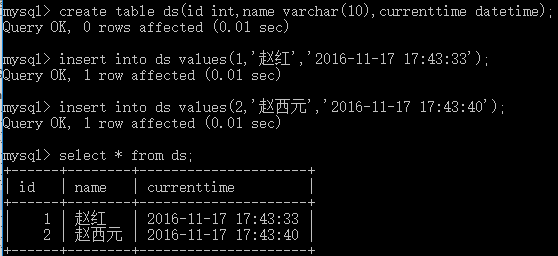


## 插入方法两种：字符串或者一串数字。

### 第一种插入方法：

插入date、time、datetime等数据类型值时，和**插入字符串**一样，注意格式即可，**yyyy-MM-dd hh:mm:ss**

对于时间类型的值，虽然是以**字符串**形式插入的，**但是插入的时候仍会检验插入的值是否合法，如不合法，则报错**。





### 第二种插入方式：直接输入一串数字。

注意：这串数字既可以利用引号括起来，也可以不括。



**insert into user(timeStamp) values('2017-08-08 12:12:12'), ('20180808121212'),(20190909131313);**

## 时间的大小比较：用于选择某一个时间段的数据查询。

限定**时间范围**进行查询数据。

利用**> 、<、<=、>=、=、!=、或<>**即可。

注意： **等于用单个 = 表示；不等号既可以利用!=也可以利用<>。**

**例如：select \* from user where timeStamp >'2017-09-07 11-11-11';**

**select \* from user where timeStamp>20170906111111;**

**select \* from user where timeStamp>'20170906111111';**

# MySQL常用数据类型详解

主要分为三大类：

## 数值类型：bit、tinyint、bool、boolean、smalllint、int、bigint、float、double。

## 文本、二进制类型：char、varchar、blob、text

## 日期时间类型：date、time、datetime、timestamp、year



注意：时间日期类型四种：**date 、time、datetime、timestamp**。

**date ： 2016-12-12**

**time：11:11:11**

**datetime： 2016-12-12 11:11:11**

**timestamp：2016-12-28 10:56:33**  注意：timestamp是自动添加当前时间的。格式和datetime一样。**在添加数据的时候，自动添加该字段**。不向该字段赋值或赋值为null，该字段都会自动设置为当前时间。当然该字段也可以像datetime一样使用，向该字段设置任意时间。

