/* 数据类型（列类型） */ ------------------

1. 数值类型

-- a. 整型 ----------

类型	字节	范围（有符号位）	
tinyint	1字节	-128 ~ 127	无符号位：0 ~ 255
smallint	2字节	-32768 ~ 32767	
mediumint	3字节	-8388608 ~ 8388607	
int	4字节		
bigint	8字节		

int(M) M表示总位数

- 默认存在符号位，unsigned 属性修改
- 显示宽度，如果某个数不够定义字段时设置的位数，则前面以0补填，zerofill 属性修改
  例：int(5) 插入一个数'123'，补填后为'00123'
- 在满足要求的情况下，越小越好。
- 1表示bool值真，0表示bool值假。MySQL没有布尔类型，通过整型0和1表示。常用tinyint(1)表示布尔型。

-- b. 浮点型 ----------

类型	字节	范围
float(单精度)	4字节	
double(双精度)	8字节	

浮点型既支持符号位 unsigned 属性，也支持显示宽度 zerofill 属性。
  不同于整型，前后均会补填0.

定义浮点型时，需指定总位数和小数位数。
  float(M, D) double(M, D)
  M表示总位数，D表示小数位数。
  M和D的大小会决定浮点数的范围。不同于整型的固定范围。
  M既表示总位数（不包括小数点和正负号），也表示显示宽度（所有显示符号均包括）。
  支持科学计数法表示。
  浮点数表示近似值。

-- c. 定点数 ----------

decimal -- 可变长度

decimal(M, D) M也表示总位数，D表示小数位数。

保存一个精确的数值，不会发生数据的改变，不同于浮点数的四舍五入。

将浮点数转换为字符串来保存，每9位数字保存为4个字节。

2. 字符串类型

-- a. char, varchar ----------

char 定长字符串，速度快，但浪费空间

varchar 变长字符串，速度慢，但节省空间

M表示能存储的最大长度，此长度是字符数，非字节数。

不同的编码，所占用的空间不同。

char,最多255个字符，与编码无关。

varchar,最多65535字符，与编码有关。

一条有效记录最大不能超过65535个字节。
  utf8 最大为21844个字符，gbk 最大为32766个字符，latin1 最大为65532个字符

varchar 是变长的，需要利用存储空间保存 varchar 的长度，如果数据小于255个字节，则采用一个字节来保存长度，反之需要两个字节来保存。

varchar 的最大有效长度由最大行大小和使用的字符集确定。

最大有效长度是65532字节，因为在varchar存字符串时，第一个字节是空的，不存在任何数据，然后还需两个字节来存放字符串的长度，所以有效长度是64432-1-2=65532字节。

例：若一个表定义为 CREATE TABLE tb(c1 int, c2 char(30), c3 varchar(N)) charset=utf8; 问N的最大值是多少？ 答：(65535-1-2-4-30*3)/3

```
-- b. blob, text ----------
```

blob 二进制字符串（字节字符串）

tinyblob, blob, mediumblob, longblob

text 非二进制字符串（字符字符串）

tinytext, text, mediumtext, longtext

text 在定义时，不需要定义长度，也不会计算总长度。

text 类型在定义时，不可给default值

```
-- c. binary, varbinary ----------
```

类似于char和varchar，用于保存二进制字符串，也就是保存字节字符串而非字符字符串。

char, varchar, text 对应 binary, varbinary, blob.

3. 日期时间类型

一般用整型保存时间戳，因为PHP可以很方便的将时间戳进行格式化。

datetime	8字节	日期及时间	1000-01-01 00:00:00 到 9999-12-31 23:59:59
date	3字节	日期	1000-01-01 到 9999-12-31
timestamp	4字节	时间戳	19700101000000 到 2038-01-19 03:14:07
time	3字节	时间	-838:59:59 到 838:59:59
year	1字节	年份	1901 - 2155

```
datetime    YYYY-MM-DD hh:mm:ss
timestamp   YY-MM-DD hh:mm:ss
            YYYYMMDDhhmmss
            YYMMDDhhmmss
            YYYYMMDDhhmmss
            YYMMDDhhmmss
date        YYYY-MM-DD
            YY-MM-DD
            YYYYMMDD
            YYMMDD
            YYYYMMDD
            YYMMDD
time        hh:mm:ss
            hhmmss
            hhmmss
year        YYYY
            YY
            YYYY
            YY
```

4. 枚举和集合

```
-- 枚举(enum) ----------
enum(val1, val2, val3...)
```

在已知的值中进行单选。最大数量为65535.

枚举值在保存时，以2个字节的整型(smallint)保存。每个枚举值，按保存的位置顺序，从1开始逐一递增。

```
    表现为字符串类型，存储却是整型。
    NULL值的索引是NULL。
    空字符串错误值的索引值是0。
-- 集合（set） ----------
set(val1, val2, val3...)
    create table tab ( gender set('男', '女', '无') );
    insert into tab values ('男, 女');
    最多可以有64个不同的成员。以bigint存储，共8个字节。采取位运算的形式。
    当创建表时，SET成员值的尾部空格将自动被删除。
```

NULL 值处理：

- **IS NULL: 当列的值是 NULL,此运算符返回 true。**
- **IS NOT NULL: 当列的值不为 NULL, 运算符返回 true。**
- **<=>: 比较操作符（不同于=运算符），当比较的的两个值为 NULL 时返回 true。**

在 MySQL 中，NULL 值与任何其它值的比较（即使是 NULL）永远返回 false，即 NULL = NULL 返回false 。

建表规范

```
/* 建表规范 */ ------------------
    -- Normal Format, NF
        - 每个表保存一个实体信息
        - 每个具有一个ID字段作为主键
        - ID主键 + 原子表
    -- 1NF, 第一范式
        字段不能再分，就满足第一范式。
    -- 2NF, 第二范式
        满足第一范式的前提下，不能出现部分依赖。
        消除符合主键就可以避免部分依赖。增加单列关键字。
    -- 3NF, 第三范式
        满足第二范式的前提下，不能出现传递依赖。
        某个字段依赖于主键，而有其他字段依赖于该字段。这就是传递依赖。
        将一个实体信息的数据放在一个表内实现。
```