MySQL常见数据类型——数值型篇

最近翻阅MySQL笔记，发现里面数据类型的东西好多哦，特此整理一份，以供参考。MySQL提供了多种数据类型，主要包括：数值型、字符型、日期型等。不同的MySQL版本支持的数据类型可能稍有不同。

**使用的三大原则：★**

1.更小的通常更好，尽量使用可以正确存储数据的最小数据类型。

2.简单就好，简单数据类型的操作通常需要更少的cpu周期。

3.尽量避免null，如果查询中包含可为null的列，对MySQL来说更难优化，因为可为null的列使得索引、索引统计和值比较都更复杂。

**数值型主要包括：整型、小数和位类型，下面详细介绍下各种具体类型。**

**整型：**

|  |  |  |
| --- | --- | --- |
| **整数类型** | **字节** | **范围** |
| **Tinyint** | 1 | 有符号：-128~127  无符号：0~255 |
| **Smallint** | 2 | 有符号：-32768~32767  无符号：0~65535 |
| **Mediumint** | 3 | 有符号：-8388608~8388607  无符号：0~1677215  **(好吧，反正很大，不用记住)** |
| **Int、integer** | 4 | 有符号：- 2147483648~2147483647  无符号：0~4294967295  **(好吧，反正很大，不用记住)** |
| **Bigint** | 8 | 有符号：  -9223372036854775808  ~9223372036854775807  无符号：0~  9223372036854775807\*2+1  (**好吧，反正很大，不用记住)** |

**说明：**

①如果不指定是无符号还是有符号，则默认是有符号。可以手动添加UNSIGNED指定为无符号整型，限制负数

②如果超出类型范围的操作，会发生out of range错误提示。插入的值显示为最大值

示例：字段id，字段类型 tinyint，插入数值：128。 （超过最大值127）

错误提示：C:\Users\liyuting\AppData\Roaming\Tencent\Users\1285983250\QQ\WinTemp\RichOle\B3TSNDC8~{XVVI7BG0$O45A.png

插入表中的值为：127

③MySQL支持在类型名称后面的小括号内指定显示宽度。如果数字位数足够或大于，则忽略宽度，如果数字位数不够的空间用字符“0”填满，但要搭配zerofill使用

如果不手动指定宽度，则有默认宽度

**示例如下：**

CREATE TABLE IF NOT EXISTS myTab(

c1 TINYINT,

c2 SMALLINT,

c3 MEDIUMINT,

c4 INT,

c5 BIGINT,

c6 TINYINT UNSIGNED,

c7 SMALLINT UNSIGNED,

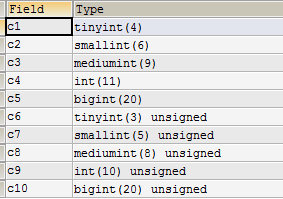
c8 MEDIUMINT UNSIGNED,

c9 INT UNSIGNED,

c10 BIGINT UNSIGNED

)

**查看表结构宽度如下：**



**插入数值：**

INSERT INTO mytab VALUES(200,200,200,200,200,200,200,200,200,200);

**显示数值：**

why？没有宽度效果？

C:\Users\liyuting\AppData\Roaming\Tencent\Users\1285983250\QQ\WinTemp\RichOle\]P~KWIUVA]0G9WP%$`2VM9P.png

**答：因为没有搭配zerofill ！！！**

**搭配上zerofill，重新创建表。示例如下：**

CREATE TABLE IF NOT EXISTS myTab(

c1 TINYINT ZEROFILL,

c2 SMALLINT ZEROFILL,

c3 MEDIUMINT ZEROFILL,

c4 INT ZEROFILL,

c5 BIGINT ZEROFILL,

c6 TINYINT UNSIGNED ZEROFILL,

c7 SMALLINT UNSIGNED ZEROFILL,

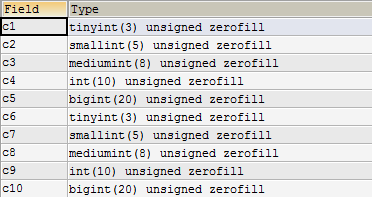
c8 MEDIUMINT UNSIGNED ZEROFILL,

c9 INT UNSIGNED ZEROFILL,

c10 BIGINT UNSIGNED ZEROFILL

)

**查看表结构宽度如下：**



**惊奇地发现，类型自动变为无符号！！！**

**重新插入数值：**

INSERT INTO mytab VALUES(200,200,200,200,200,200,200,200,200,200);

**显示数值：**

C:\Users\liyuting\AppData\Roaming\Tencent\Users\1285983250\QQ\WinTemp\RichOle\1I%DHB}O1V2JP`{)`5LQW[G.png

④整型还有一个属性：AUTO\_INCREMENT。该属性只能用于整型。AUTO\_INCREMENT标识列一般从1开始，每行增加1，不用我们手动插入，但必须定义为PRIMARY\_KEY或UNIQUE键

**小数：**

MySQL有两种方式：浮点数和定点数。

|  |  |  |
| --- | --- | --- |
| **浮点数类型** | **字节** | **范围** |
| **float** | **4** | **±1.75494351E-38~±3.402823466E+38**  **(好吧，反正很大，不用记住)** |
| **double** | **8** | **±2.2250738585072014E-308~**  **±1.7976931348623157E+308**  **(好吧，反正很大，不用记住)** |
| **定点数类型** | **字节** | **范围** |
| **DEC(M,D)**  **DECIMAL(M,D)** | **M+2** | **最大取值范围与double相同，给定decimal的有效取值范围由M和D决定** |

**说明：**

①浮点数和定点数都可以用类型名称后加“（M,D）”的方式来表示。

M：精度 ，该值的整数位+小数位一共显示M位数字

D：标度， 小数位数一共显示D位数字，如果不够后面用0补齐，如果超过，则四舍五入

**示例如下：**

CREATE TABLE mytab2(

f1 FLOAT(5,3),

f2 DOUBLE(5,3),

f3 DECIMAL(5,3)

)

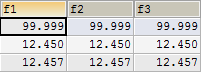
**插入数值：**

INSERT INTO mytab2 VALUES(123.456,123.456,123.456);

INSERT INTO mytab2 VALUES(12.45,12.45,12.45);

INSERT INTO mytab2 VALUES(12.4567,12.4567,12.4567);

**显示结果：**



**②浮点数和定点数的区别：**

1. 定点数在MySQL内部以字符串形式存放，比浮点数更精确，适合用于表示货币等精度高的数据
2. 在不指定精度时，浮点数默认会按照实际的精度来显示，而定点数在不指定精度时，默认M=10，D=0
3. 浮点数如果数据超越了精度和标度值，则自动将四舍五入后的结果插入，系统不会报错；

定点数则会报错！

**位类型**

|  |  |  |
| --- | --- | --- |
| **位类型** | **字节** | **范围** |
| **Bit(M)** | 1~8 | Bit(1)~bit(8) |

**说明：**

①将数值转换成二进制数存入。M的范围：1~64，如果超过，会报Out of range，如果不写，默认M=1

②根据1个字节8位换算，1—8位占1个字节，9—16位占2个字节，依次类推

57—64位占8个字节

③一般搭配bin()(显示为二进制格式)或hex()(显示为十六进制格式)函数读取

**示例：**

CREATE TABLE IF NOT EXISTS myTab(

b1 BIT,

b2 BIT(8)

)

**插入数值：**

INSERT INTO mytab VALUES(2,100);

**显示结果：**

