## 一Mysql 数据类型选择

### Char与varchar



由于char是固定长度的，所以存取速度比varchar快。缺点是浪费存储空间，程序需要对行尾进行处理，所以对那些长度变化不大，并且对查询速度有较高要求的数据可以通过char来存取。当然这个也是由不同的存储引擎来决定的。

MyISAM 存储引擎：建议使用固定长度的数据列代替可变长度的数据列。

MEMORY 存储引擎：目前都使用固定长度的数据行存储，因此无论使用 CHAR 或VARCHAR 列都没有关系。两者都是作为 CHAR 类型处理。

InnoDB 存储引擎：建议使用 VARCHAR 类型。对于 InnoDB 数据表，内部的行存储格式没有区分固定长度和可变长度列（所有数据行都使用指向数据列值的头指针），因此在

本质上，使用固定长度的 CHAR 列不一定比使用可变长度 VARCHAR 列性能要好。因而，主要的性能因素是数据行使用的存储总量。由于 CHAR 平均占用的空间多于 VARCHAR，因此使用 VARCHAR 来最小化需要处理的数据行的存储总量和磁盘 I/O 是比较好的

### Text与Blob

一般在保存少量字符串的时候，我们会选择 CHAR 或者 VARCHAR；而在保存较大文本时，通常会选择使用 TEXT 或者 BLOB，二者之间的主要差别是 BLOB 能用来保存二进制数据，比如照片；而 TEXT 只能保存字符数据，比如一篇文章或者日记。

BLOB 和 TEXT 值会引起一些性能问题，特别是在执行了大量的删除操作时。删除操作会在数据表中留下很大的“空洞”，以后填入这些“空洞”的记录在插入的性能上会有影响。为了提高性能，建议定期使用 OPTIMIZE TABLE 功能对这类表进行碎片整理，避免因为“空洞”导致性能问题。

可以使用合成的（Synthetic）索引来提高大文本字段（BLOB 或 TEXT）的查询性能。

简单来说，合成索引就是根据大文本字段的内容建立一个散列值，并把这个值存储在单独的

数据列中，接下来就可以通过检索散列值找到数据行了。

在不必要的时候避免检索大型的 BLOB 或 TEXT 值。

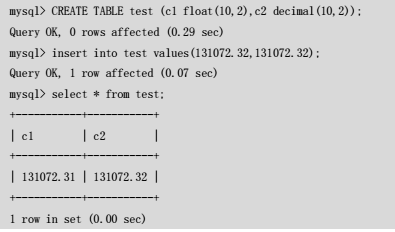
把 BLOB 或 TEXT 列分离到单独的表中。

### 1.3定点数与浮点数

浮点数一般用于表示含有小数部分的数值。当一个字段被定义为浮点类型后，如果插入数据的精度超过该列定义的实际精度，则插入值会被四舍五入到实际定义的精度值，然后插入，四舍五入的过程不会报错。在 MySQL 中 float、double（或 real）用来表示浮点数。

定点数不同于浮点数.

定点数实际上是以字符串形式存放的，所以定点数可以更加精确的保存数据。如果实际插入的数值精度大于实际定义的精度，则 MySQL 会进行警告但是数据按照实际精度四舍五入后插入.



1浮点数存在误差问题.

2对货币等对精度敏感的数据，应该用定点数表示或存储.

3在编程中，如果用到浮点数，要特别注意误差问题，并尽量避免做浮点数比较.

4要注意浮点数中一些特殊值的处理。

### 1.4日期

YEAR:根据实际需要选择能够满足应用的最小存储的日期类型。如果应用只需要记录“年

份”，那么用 1 个字节来存储的 YEAR 类型完全可以满足，而不需要用 4 个字节来

存储的 DATE 类型。这样不仅仅能节约存储，更能够提高表的操作效率。

DATETIME:如果要记录年月日时分秒，并且记录的年份比较久远，那么最好使用 DATETIME，而不要使用 TIMESTAMP。因为 TIMESTAMP 表示的日期范围比 DATETIME 要短得多。

TIMESTAMP 如果记录的日期需要让不同时区的用户使用，那么最好使用 TIMESTAMP，因为日期类型中只有它能够和实际时区相对应。

