记录长度

MySQL 中规定：任何一条记录最长不超过 65535 个字节，这意味着varchar永远达不到理论最大值。

那么，varchar实际存储长度能达到多大呢？这由编码字符集决定。

下面，以varchar在UTF-8和GBK的情况为例，执行如下 SQL 语句，进行演示：

-- 求出 varchar 在 utf8 和 gbk 字符集下的实际最大值

create table my\_utf8(

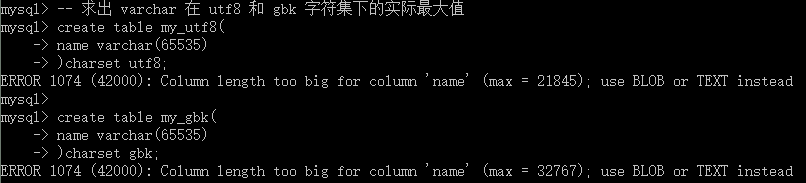
name varchar(65535)

)charset utf8;

create table my\_gbk(

name varchar(65535)

)charset gbk;

[](https://camo.githubusercontent.com/f8a89d641080ca272ed0049b6f943cf963a569aa/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353232313631393039383233)

观察上面的结果，发现咱们定义的字段name的长度超过限制啦，并且提示了其在utf8和gbk字符集下各自的最大值。那么，咱们修改 SQL 语句如下，并再次执行：

-- 求出 varchar 在 utf8 和 gbk 字符集下的实际最大值

create table my\_utf8(

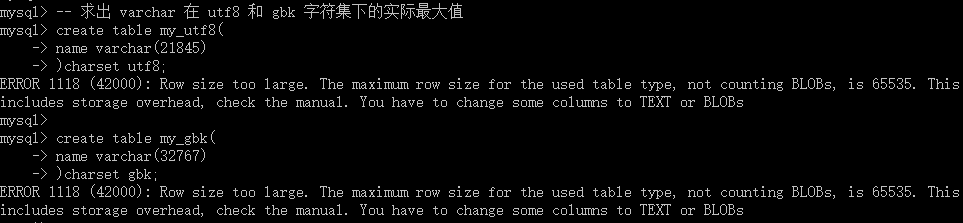
name varchar(21845)

)charset utf8;

create table my\_gbk(

name varchar(32767)

)charset gbk;

[](https://camo.githubusercontent.com/546fadf05556091716119801983ba7ab0a48a06b/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353232313632333230393334)

观察上面的执行结果，好吧，仍然在报错，为什么呢？观察如下 SQL 语句，并执行：

-- 求出 varchar 在 utf8 和 gbk 字符集下的实际最大值

create table my\_utf8(

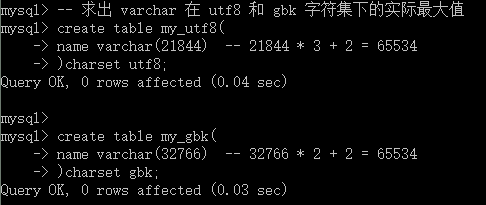
name varchar(21844) -- 21844 \* 3 + 2 = 65534

)charset utf8;

create table my\_gbk(

name varchar(32766) -- 32766 \* 2 + 2 = 65534

)charset gbk;

[](https://camo.githubusercontent.com/e1e3737fdb0854b2b040a3c01f9c28f1b7bb57e4/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353232313633393238363434)

如上图所示，咱们已经创建成功啦！至于什么定义字段name的长度为21844和32766是由于：

21845 \* 3 + 2 = 65537 > 65535

32767 \* 2 + 2 = 65536 > 65535

因此，在提示的最大值的基础上各自减1. 至于，为什么还要加2，则是因为varchar为变长字符串，在其定义的时候，也就是说在分配存储空间的时候，都会自动多分配1到2个字节空间，因为咱们想要算最大的存储范围，所以加2.

在这里，细心的同学会发现一个问题，那就是：在咱们创建表my\_utf8和my\_gbk的时候，咱们仅用了65534个字节，还剩余一个字节。现在，如果咱们想要将65535个字节都用了，怎么办呢？好说，增加一个tinyint类型的字段即可：

-- 求出 varchar 在 utf8 和 gbk 字符集下的实际最大值

create table my\_utf81(

stuno tinyint, -- 1

name varchar(21844) -- 21844 \* 3 + 2 = 65534

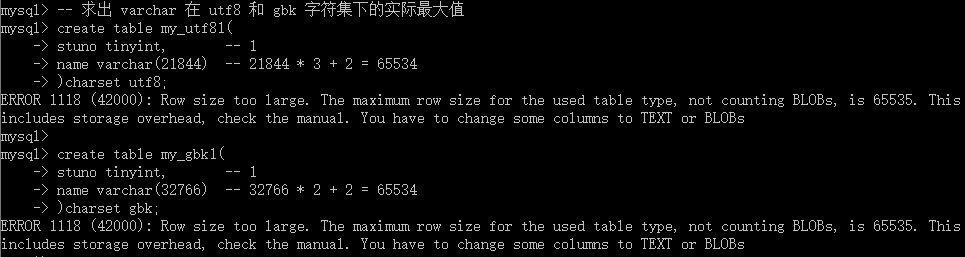
)charset utf8;

create table my\_gbk1(

stuno tinyint, -- 1

name varchar(32766) -- 32766 \* 2 + 2 = 65534

)charset gbk;

[](https://camo.githubusercontent.com/9dd67f3b34aaf7c52c6732977994e67dcd880566/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353232313635303531353839)

观察上面的结果，呃，竟然又出错啦！为什么啊？65534 + 1 = 65535，并没有超出范围啊！其实吧，之所以会出现这样的问题，是因为：在 MySQL 的记录中，如果有任何一个字段允许为空，那么系统就会自动从整个记录中保留一个字节来存储null，若想释放null所占的字节，则必须保证所有字段都不允许为空。

-- 求出 varchar 在 utf8 和 gbk 字符集下的实际最大值

create table my\_utf82(

stuno tinyint not null, -- 1

name varchar(21844) not null -- 21844 \* 3 + 2 = 65534

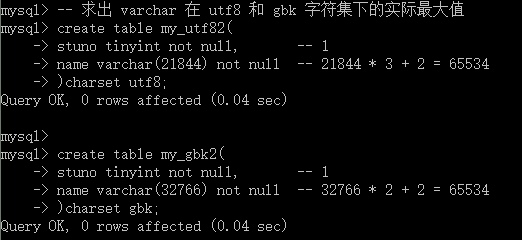
)charset utf8;

create table my\_gbk2(

stuno tinyint not null, -- 1

name varchar(32766) not null -- 32766 \* 2 + 2 = 65534

)charset gbk;

[](https://camo.githubusercontent.com/7e4acf2a884dfdc6d912d9ef176454207a5eb83d/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353232313635373033313535)

如上图所示，咱们已经成功创建了表my\_utf82和my\_gbk2.

此外，在 MySQL 中，text文本字符串不占用记录长度，额外存储，但是text文本字符串也是属于记录的一部分，无论是在utf8还是在gbk字符集之中，其都占用记录中的10个字节长度，用来保存数据的地址以及长度