自动增长

自动增长：auto\_increment，当对应的字段，不给值，或者是默认值，或者是null的时候，就会自动的被系统触发，系统会从当前字段中取已有的最大值再进行+1操作，得到新的字段值。

自增长通过跟主键进行搭配使用，其特点为：

任何字段要做自增长，前提其本身必须是一个索引，即key栏有值；

自增长字段必须是数字（整型）；

每张表最多有一个自增长字段。

执行如下 SQL 语句，进行测试：

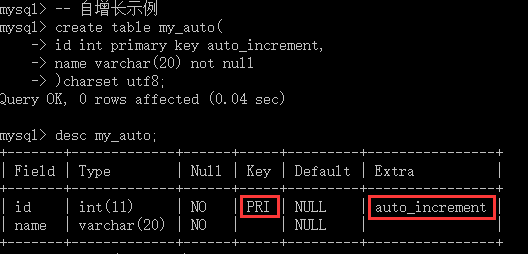
-- 自增长示例

create table my\_auto(

id int primary key auto\_increment,

name varchar(20) not null

)charset utf8;

[](https://camo.githubusercontent.com/64fde169000c8ca6eb295ece88080e591b41d1f1/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353233323235343531303438)

使用自增长

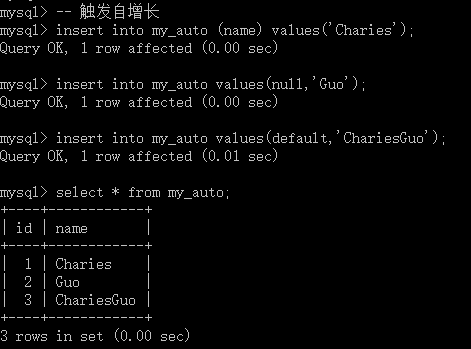
当自增长给定的值为null或者默认值的时候，就是触发自动增长。

-- 触发自增长

insert into my\_auto (name) values('Charies');

insert into my\_auto values(null,'Guo');

insert into my\_auto values(default,'ChariesGuo');

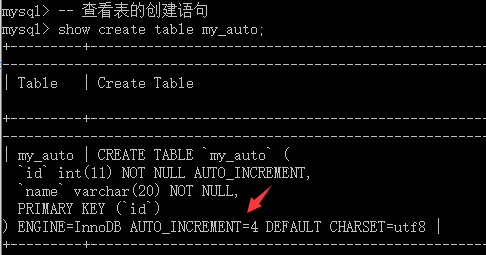
[](https://camo.githubusercontent.com/9ab4ef35b896ecc48aa1b2d884419902d234be15/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353233323330323134343435)

如上图所示，自增长的第一个元素默认是1，自增长每次都是自增1。此外，如果自增长对应的字段输入了值，那么自增长失效，但是下一次还是能够正确的自增长，即从取该字段的最大值+1.

那么，如何确定下一次自增长的数值是什么呢？简单，咱们可以通过查看表的创建语句看到。

-- 查看表的创建语句

show create table my\_auto;

[](https://camo.githubusercontent.com/74178c38fb9e3a38d106f85a29fe67f90ffd99d4/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353233323330393038353031)

如上图所示，显然在咱们之前插入的 3 条记录之后，下一次自增长是4，符合咱们的结论。

修改自增长

自增长如果是涉及到字段改变，就必须先删除自增长，然后再增加自增长，因为每张表只能有一个自增长字段。

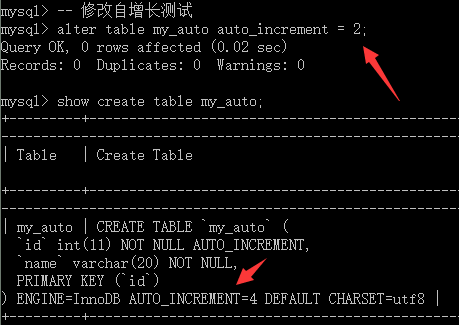
如果修改当前自增长字段已经存在的值，则只能修改比当前已有自增长字段中的最大值更大，不能更小，因为更小不生效。

基本语法：alter table + 表名 + auto\_increment = 值;

执行如下 SQL 语句，进行测试：

-- 修改自增长测试

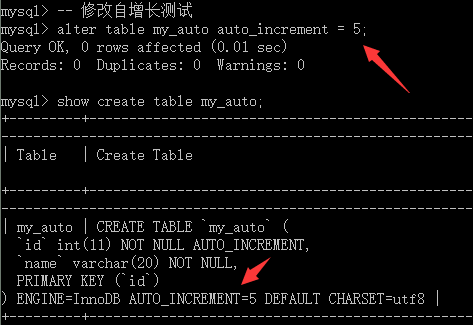
alter table my\_auto auto\_increment = 2;

[](https://camo.githubusercontent.com/8ae7c97c7552b30eb0eabeea04265d693fbb878a/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353233323333303334383437)

如上图所示，当咱们修改自增长小于当前自增长字段中的最大值时，虽然显示 SQL 语句执行成功，但实际上并没有修改成功。下面，在执行如下 SQL 语句，进行测试：

-- 修改自增长测试

alter table my\_auto auto\_increment = 5;

[](https://camo.githubusercontent.com/b1aed215b0bc889706d0c31f15098df79d86c087/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353233323333323530373034)

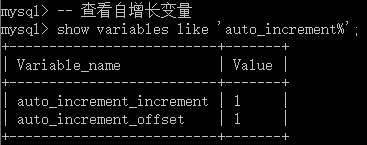
如上图所示，显然当咱们修改自增长大于当前自增长字段中的最大值时，修改成功。

在这里，咱们不妨思考一下，为什么自增长是从1开始呢？为什么每次都自增1呢？虽然现在咱们不知道是什么原因导致上面的表示形式，但是咱们知道，所有系统的表现（如字符集、校对集）都是由系统内部的变量进行控制的，因此咱们可以查看自增长对应的变量：

基本语法：show variables like 'auto\_increment%';

-- 查看自增长变量

show variables like 'auto\_increment%';

[](https://camo.githubusercontent.com/ccee35ffa44ff66fe1bf375abb0a3301578f2c94/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353233323334313535313333)

如上图所示，其中auto\_increment\_increment表示步长，auto\_increment\_offset表示初始值.

因此，咱们可以通过修改上面的两个变量实现不同的效果，但是需要注意的是：修改是会话级别，并且修改的是整个数据库，而不是单张表。

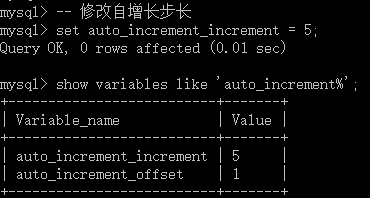
基本语法：set auto\_increment\_increment = 值;

执行如下 SQL 语句，进行测试：

-- 修改自增长步长

set auto\_increment\_increment = 5;

insert into my\_auto values (null,'Guobinhit');

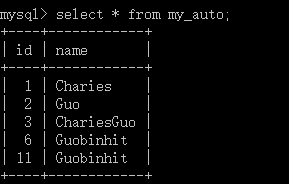
[](https://camo.githubusercontent.com/ed745311bbac2d589bc6fd0649cb4e821ccb6dc4/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353233323334373532313638)

如上图所示，咱们已经修改成功！接下来，执行如下 SQL 语句，进行测试：

-- 插入测试数据

insert into my\_auto values (null,'Guobinhit');

insert into my\_auto values (default,'Guobinhit');

[](https://camo.githubusercontent.com/8c98703513f4602a31e5e8b91f474f7be0e633fe/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353233323335303437303434)

如上图所示，咱们插入的第一个Guobinhit的id值竟然是6，第二个Guobinhit的id值倒是在6的基础上增加了5为11正常。这个现象产生的原因是：咱们之前的修改会导致系统产生一个误差，从初始值1开始计算加5，因此第一个Guobinhit的id值是6.

不过说实话，修改自增长并没有什么实际的意义，在此只是想让大家知道：自增长的初始值和步长都是可以修改的而已。

删除自增长

自增长是字段的一个属性，因此可以通过modify来进行修改。想要删除自增长的话，只需要保证字段没有auto\_increment即可。

基本语法：alter table + 表名 + modify + 字段 + 类型;

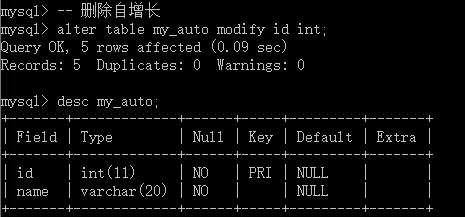
执行如下 SQL 语句，进行测试：

-- 删除自增长

alter table my\_auto modify id int primary key;

[IMG_264](https://camo.githubusercontent.com/bb27c6a86dd0ef404c10bfb100badd161ed8f426/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353234303030353138333839)

如上图所示，呃，好吧，出错啦！这是因为表my\_auto之前已经定义了主键，所以在执行上述 SQL 语句的时候，系统会认为咱们想要定义多个主键，自然就要报错啦！解决方法就是，去掉上述 SQL 语句中的primary key即可。

[](https://camo.githubusercontent.com/882b67b53b11d3e72dd74d1d37bfdd49b032f26d/687474703a2f2f696d672e626c6f672e6373646e2e6e65742f3230313730353234303030383236363238)

如上图所示，显然咱们已经成功删除自增长啦！