



下载APP



## 03 | 表：怎么创建和修改表？

2021-03-13 朱晓峰

MySQL 必知必会

[进入课程 >](#)**讲述：朱晓峰**

时长 12:29 大小 11.44M



你好，我是朱晓峰。今天，我们来聊一聊怎么创建和修改数据表。

创建和修改数据表，是数据存储过程中的重要一环。我们不仅需要把表创建出来，还需要正确地设置限定条件，这样才能确保数据的一致性和完整性。同时，表中的数据会随着业务需求的变化而变化，添加和修改相应的字段也是常见的操作。这节课，我们就来学习下具体的方法。

在我们的超市项目里，客户经常需要进货，这就需要在 MySQL 数据库里面创建一个表来管理进货相关的数据。我们先看看这个表里有什么内容。



假设这个表叫做进货单头表（importhead），如下图所示：

listnumber (单号)	supplierid (供货商编号)	stocknumber (仓库编号)	importtype (进货方式)	quantity (进货数量)	importvalue (进货金额)	Recorder (录入人编号)	recordingdate (录入时间)
1234	1	1	1	10	100	1	2020-12-20
2345	1	1	2	20	200	1	2020-12-21
3456	1	1	3	5	50	1	2020-12-25

这里的 1、2、3 表示门店的 3 种进货方式，分别是配送中心配送、门店采买和供货商直供。

其中，“1（配送中心配送）”是标准进货方式。因为超市是连锁经营，为了确保商品质量和品类一致，超过 9 成的门店进货，是通过配送中心进行配送的。因此，我们希望这个字段的值能够默认是 1，这样一来，除非有特别的指定，否则，门店进货单的进货方式，就自动设置成“1”了。

现在，客户需要一个类似的表来存储进货数据，而且进货方式还有 3 个可能的取值范围，需要设置默认值，那么，应该怎么创建这个表呢？另外，创建好表以后，又该怎么进行修改呢？

## 如何创建数据表？

首先，我们要知道 MySQL 创建表的语法结构：

复制代码

```
1 CREATE TABLE <表名>
2 {
3  字段名1 数据类型 [字段级别约束] [默认值],
4  字段名2 数据类型 [字段级别约束] [默认值],
5  .....
6  [表级别约束]
7  };
```


在这里，我们通过定义表名、表中的字段、表的属性等，把一张表创建出来。

你可能注意到了，在 MySQL 创建表的语法结构里面，有一个词叫做“约束”。“约束”限定了表中数据应该满足的条件。MySQL 会根据这些限定条件，对表的操作进行监

控，阻止破坏约束条件的操作执行，并提示错误，从而确保表中数据的唯一性、合法性和完整性。这是创建表时不可缺少的一部分。


下面我来带你创建刚刚提到的进货单表。需要注意的是，这里我们需要定义默认值，也就是要定义默认值约束，除此之外，还有很多种约束，一会儿我再细讲。

我们先来看基本的数据表创建流程，创建代码如下：


 复制代码

```
1 CREATE TABLE demo.importhead
2 (
3   listnumber INT,
4   supplierid INT,
5   stocknumber INT,
6   --我们在字段importtype定义为INT类型的后面，按照MySQL创建表的语法，加了默认值1。
7   importtype INT DEFAULT 1,
8   quantity DECIMAL(10,3),
9   importvalue DECIMAL(10,2),
10  recorder INT,
11  recordingdate DATETIME
12 );
```

运行这个 SQL 语句，表 demo.importhead 就按照我们的要求被创建出来了。

在创建表的时候，字段名称要避开 MySQL 的  系统关键字，原因是 MySQL 系统保留的关键字都有特定的含义，如果作为字段名称出现在 SQL 语句中，MySQL 会把这个字段名称理解为系统关键字，从而导致 SQL 语句无法正常运行。比如，刚刚我们把进货金额设置为 “importvalue”，而不是 “value”，就是因为，“value” 是 MySQL 的系统关键字。

好了，现在我们尝试往刚刚创建的表里插入一条记录，来验证一下对字段 “importtype” 定义的默认值约束是否起了作用。

 复制代码

```
1 INSERT INTO demo.importhead
2 (
3   listnumber,
4   supplierid,
5   stocknumber,
6   -- 这里我们没有插入字段importtype的值
7   quantity,
```

```
8 importvalue,  
9 recorder,  
10 recordingdate  
11 )  
12 VALUES  
13 (  
14 3456,  
15 1,  
16 1,  
17 10,  
18 100,  
19 1,  
20 '2020-12-10'  
21 );
```

插入完成后，我们来查询一下表的内容：

```
1 SELECT *  
2 FROM demo.importhead
```

[复制代码](#)

运行结果如下：

```
1 mysql> select * from demo.importhead;  
2 +-----+-----+-----+-----+-----+-----+  
3 | listnumber | supplierid | stocknumber | importtype | quantity | importvalue  
4 +-----+-----+-----+-----+-----+-----+  
5 |          1234 |          1 |          1 |          1 |    10.000 |     100.00  
6 |          2345 |          1 |          1 |          2 |    20.000 |    2000.00  
7 |          3456 |          1 |          1 |          1 |    20.000 |    2000.00  
8 +-----+-----+-----+-----+-----+-----+  
9 3 rows in set (0.00 sec)
```

[复制代码](#)

你会发现，字段 importtype 的值已经是 1 了。这样，通过在创建表的时候设置默认值，我们就实现了将字段的默认值定义为 1 的目的。

到这里，表就被创建出来了。

## 都有哪些约束？

刚刚这种给字段设置默认值的做法，就是默认约束。设置了默认约束，插入数据的时候，如果不明确给字段赋值，那么系统会把设置的默认值自动赋值给字段。

除了默认约束，还有主键约束、外键约束、非空约束、唯一性约束和自增约束。

我们在 [🔗 上节课](#) 里学的主键，其实就是主键约束，我就不多说了。外键约束涉及表与表之间的关联，以及确保表的数据一致性的问题，内容比较多，后面我在讲“关联表”的时候，再给你具体解释。

现在，我来重点给你介绍一下非空约束、唯一性约束和自增约束。

## 1. 非空约束

非空约束表示字段值不能为空，如果创建表的时候，指明某个字段非空，那么添加数据的时候，这个字段必须有值，否则系统就会提示错误。

## 2. 唯一性约束

唯一性约束表示这个字段的值不能重复，否则系统会提示错误。跟主键约束相比，唯一性约束要更加弱一些。

在一个表中，我们可以指定多个字段满足唯一性约束，而主键约束则只能有一个，这也是 MySQL 系统决定的。另外，**满足主键约束的字段，自动满足非空约束，但是满足唯一性约束的字段，则可以是空值。**

为了方便你理解，我来举个例子。比如，我们有个商品信息表 goodsmaster，如下所示：

barcode	goodsname	price
0001	book	89
0002	pen	12

barcode 代表条码，goodsname 代表名称。为了防止条码重复，我们可以定义字段“barcode”满足唯一性约束。这样一来，条码就不能重复，但是可以为空，而且只能有一条记录条码为空。

同样道理，为了防止名称重复，我们也可以定义字段“goodsname”满足唯一性约束。但是，由于无论名称和条码都可能重用，或者可能为空，都不适合做主键。因此，对这个表来说，可以添加一个满足唯一性要求的新字段来做主键。

### 3. 自增约束

自增约束可以让 MySQL 自动给字段赋值，且保证不会重复，非常有用，只是不容易用好。所以，我借助一个例子来给你具体讲一讲。

假如我们有这样一个商品信息表：

barcode	goodsname	price
0001	书	89
0002	馒头	1.5
0002	花卷	1.8

从这个表中，我们可以看到，barcode、goodsname 和 price 都不能确保唯一性，所以没有任何一个字段可以做主键，因此，我们可以自己添加一个字段 itemnumber，并且每次添加一条数据的时候，要给值增加 1。怎么实现呢？我们就可以通过定义自增约束的方式，让系统自动帮我们赋值，从而满足唯一性，这样就可以做主键了。

itemnumber	barcode	goodsname	price
1	0001	书	89
2	0002	馒头	1.5
3	0002	花卷	1.8

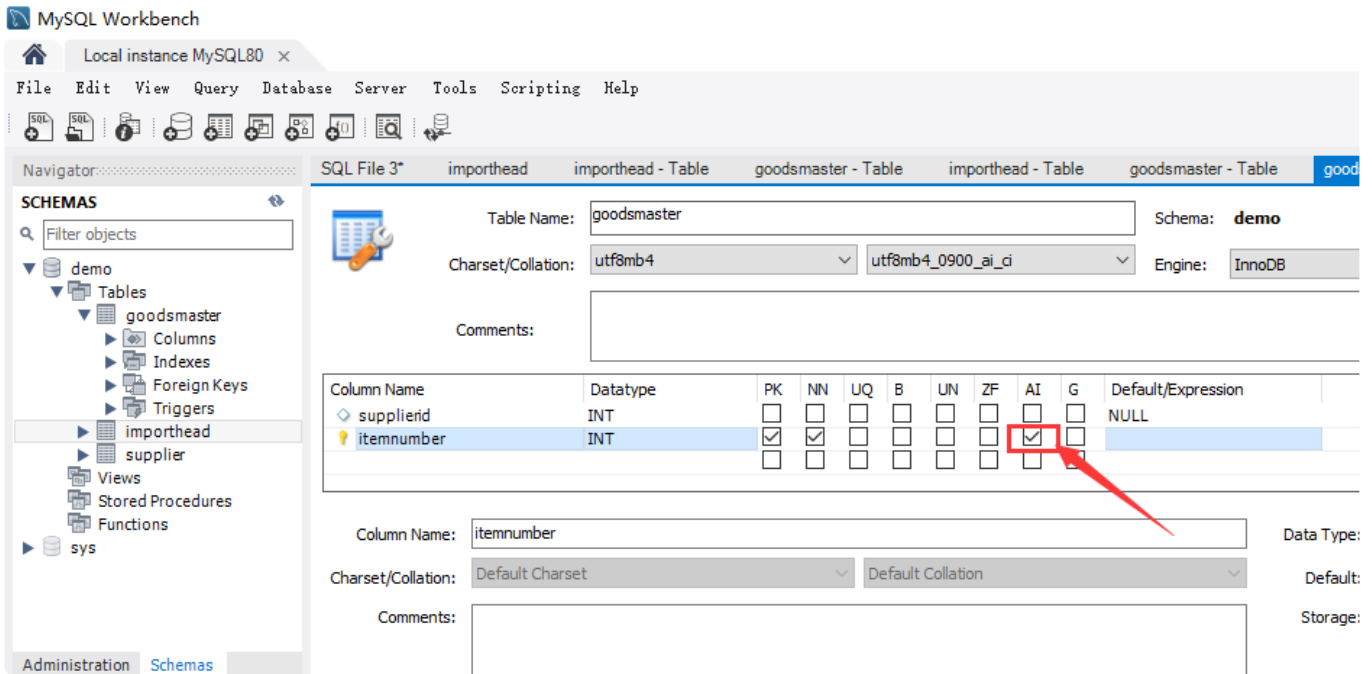
这里，有 2 个问题需要你注意一下。

第一，在数据表中，只有整数类型的字段（包括 TINYINT、SMALLINT、MEDIUMINT、INT 和 BIGINT），才可以定义自增约束。自增约束的字段，每增加一条数据，值自动增加 1。



第二，你可以给自增约束的字段赋值，这个时候，MySQL 会重置自增约束字段的自增基数，下次添加数据的时候，自动以自增约束字段的最大值加 1 为新的字段值。

举个例子，我们通过 Workbench 把数据表 demo.goodsmaster 中的字段 itemnumber，定义为满足自增约束，如下图所示：




然后，我们插入一条测试记录：

复制代码

```
1 INSERT INTO demo.goodsmaster
2 (
3 itemnumber,
4 barcode,
5 goodsname,
6 specification,
7 unit,
8 price
9 )
10 VALUES
11 (
12 -- 指定商品编号为100:
13 100,
14 '0003',
15 '测试1',
16 '',
17 '个',
18 10
19 );
```




运行这条语句之后，查看表的内容，我们得到：

 复制代码

```
1 mysql> select * from demo.goodsmaster;
2 +-----+-----+-----+-----+-----+
3 | itemnumber | barcode | goodsname | specification | unit | price |
4 +-----+-----+-----+-----+-----+
5 |          1 | 0001    | 书        | 16开          | 本   | 89.00 |
6 |          2 | 0002    | 地图      | NULL          | 张   | 9.90  |
7 |          3 | 0003    | 笔        | 10支          | 包   | 3.00  |
8 |         100 | 0003    | 测试1     |               | 个   | 10.00 |
9 +-----+-----+-----+-----+-----+
10 4 rows in set (0.02 sec)
```


这个时候，字段“itemnumber”的值不连续，最大值是我们刚刚插入的 100。

接着，我们再插入一条数据：

 复制代码

```
1 INSERT INTO demo.goodsmaster
2 (
3 -- 不指定自增字段itemnumber的值
4 barcode,
5 goodsname,
6 specification,
7 unit,
8 supplierid,
9 price
10 )
11 VALUES
12 (
13 '0004',
14 '测试2',
15 '',
16 '个',
17 1,
18 20
19 );
```

运行这条语句之后，我们再查看表的内容：

 复制代码

```
1 mysql> select * from demo.goodsmaster;
```

```
2 +-----+-----+-----+-----+
3 | itemnumber | barcode | goodsname | specification | unit | price |
4 +-----+-----+-----+-----+
5 |          1 | 0001    | 书        | 16开          | 本   | 89.00 |
6 |          2 | 0002    | 地图      | NULL          | 张   | 9.90  |
7 |          3 | 0003    | 笔        | 10支          | 包   | 3.00  |
8 |         100 | 0003    | 测试1     |               | 个   | 10.00 |
9 |         101 | 0004    | 测试2     |               | 个   | 20.00 |
10 +-----+-----+-----+-----+
11 5 rows in set (0.00 sec)
```

可以看到，系统自动给自增字段 “itemnumber” ，在最大值的基础之上加了 1，赋值为 101。

好了，到这里，我们就学会了创建表和定义约束的方法。约束要根据业务需要定义在相应的字段上，这样才能保证数据是准确的，你一定要注意它的使用方法。

## 如何修改表？

创建完表以后，我们经常还需要修改表，下面我们就来学习下修改表的方法。

在咱们的超市项目中，当我们创建新表的时候，会出现这样的情况：我们前面创建的进货单表，是用来存储进货数据的。但是，我们还要创建一个进货单历史表

(importtheadhist)，用来存储验收过的进货数据。这个表的结构跟进货单表类似，只是多了 2 个字段，分别是验收人 (confirmer) 和验收时间 (confirmdate)。针对这种情况，我们很容易想到可以通过复制表结构，然后在这个基础上通过修改表结构，来创建新的表。具体怎么实现呢？接下来，我就给你讲解一下。

首先，我们要把原来的表结构复制一下，代码如下：

```
1 CREATE TABLE demo.importtheadhist
2 LIKE demo.importthead;
```

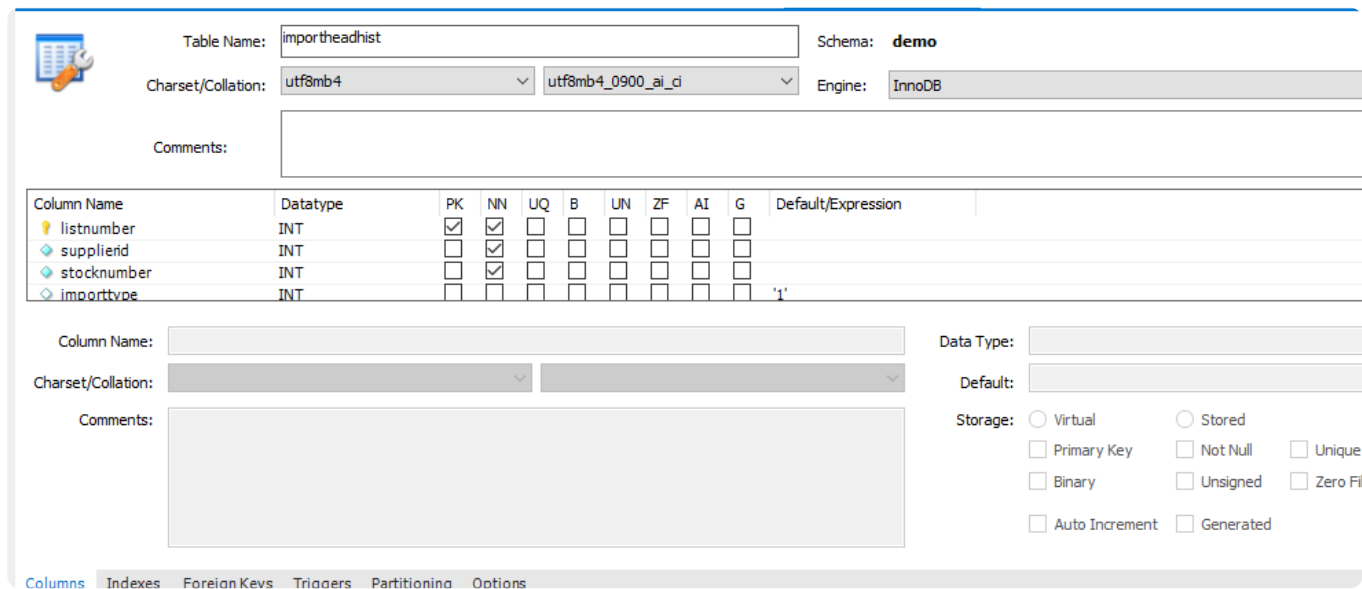
 复制代码

运行这个语句之后，一个跟 demo.importthead 有相同表结构的空表 demo.importtheadhist，就被创建出来了。

这个新创建出来的表，还不是我们需要的表，我们需要对这个表进行修改，通过添加字段和修改字段，来获得我们需要的“进货单历史表”。

## 添加字段

你可能会想到，我们可以通过 Workbench，用可视化操作来修改表的结构，如下图所示：



这样当然没问题，但是我想给你讲一个更方便灵活的方式：**用 SQL 语句来修改表的结构。**

现在我要给这个新的表增加 2 个字段：confirmer 和 confirmdate，就可以用下面的代码：

复制代码

```
1 mysql> ALTER TABLE demo.importheadhist
2     -> ADD confirmer INT; -- 添加一个字段confirmer, 类型INT
3 Query OK, 0 rows affected (0.04 sec)
4 Records: 0 Duplicates: 0 Warnings: 0
5
6 mysql> ALTER TABLE demo.importheadhist
7     -> ADD confirmdate DATETIME; -- 添加一个字段confirmdate, 类型是DATETIME
8 Query OK, 0 rows affected (0.04 sec)
9 Records: 0 Duplicates: 0 Warnings: 0
```

运行这个 SQL 语句，查看表的结构，你会发现，在字段的最后，多了两个字段：

“confirmer”，数据类型是 INT；

“confirmdate”，类型是 DATETIME。

我们来查看一下表结构：

[复制代码](#)

```
1 mysql> DESCRIBE demo.importtheadhist;
2 +-----+-----+-----+-----+-----+-----+
3 | Field          | Type          | Null | Key | Default | Extra |
4 +-----+-----+-----+-----+-----+-----+
5 | listnumber     | int           | NO   | PRI | NULL    |       |
6 | supplierid     | int           | NO   |     | NULL    |       |
7 | stocknumber    | int           | NO   |     | NULL    |       |
8 | importtype     | int           | YES  |     | 1       |       |
9 | quantity       | decimal(10,3) | YES  |     | NULL    |       |
10 | importvalue    | decimal(10,2) | YES  |     | NULL    |       |
11 | recorder       | int           | YES  |     | NULL    |       |
12 | recordingdate  | datetime      | YES  |     | NULL    |       |
13 | confirmer      | int           | YES  |     | NULL    |       |
14 | confirmdate    | datetime      | YES  |     | NULL    |       |
15 +-----+-----+-----+-----+-----+-----+
16 10 rows in set (0.02 sec)
```

这样，通过简单增加 2 个字段，我们就获得了进货单历史表。

## 修改字段


除了添加字段，我们可能还要修改字段，比如，我们要把字段名称 “quantity” 改成 “importquantity”，并且把字段类型改为 DOUBLE，该怎么操作呢？

我们可以通过修改表结构语句 ALTER TABLE，来修改字段：

[复制代码](#)

```
1 mysql> ALTER TABLE demo.importtheadhist
2     -> CHANGE quantity importquantity DOUBLE;
3 Query OK, 0 rows affected (0.15 sec)
4 Records: 0 Duplicates: 0 Warnings: 0
```

运行这个 SQL 语句，查看表的结构，我们会得到下面的结果：

 复制代码


```

1 mysql> DESCRIBE demo.importtheadhist;
2 +-----+-----+-----+-----+-----+-----+
3 | Field          | Type          | Null | Key | Default | Extra |
4 +-----+-----+-----+-----+-----+-----+
5 | listnumber     | int           | NO   | PRI | NULL    |       |
6 | supplierid     | int           | NO   |     | NULL    |       |
7 | stocknumber    | int           | NO   |     | NULL    |       |
8 | importtype     | int           | YES  |     | 1       |       |
9 | importquantity | double        | YES  |     | NULL    |       |
10 | importvalue    | decimal(10,2) | YES  |     | NULL    |       |
11 | recorder       | int           | YES  |     | NULL    |       |
12 | recordingdate  | datetime      | YES  |     | NULL    |       |
13 | confirmer      | int           | YES  |     | NULL    |       |
14 | confirmdate    | datetime      | YES  |     | NULL    |       |
15 +-----+-----+-----+-----+-----+-----+
16 10 rows in set (0.02 sec)

```

可以看到，字段名称和字段类型全部都改过来了。

如果你不想改变字段名称，只想改变字段类型，例如，把字段 “importquantity” 类型改成 DECIMAL(10,3)，你可以这样写：


 复制代码

```

1 ALTER TABLE demo.importtheadhist
2 MODIFY importquantity DECIMAL(10,3);

```

运行 SQL 语句，查看表结构，你会发现，已经改过来了。

 复制代码

```

1 mysql> DESCRIBE demo.importtheadhist;
2 +-----+-----+-----+-----+-----+-----+
3 | Field          | Type          | Null | Key | Default | Extra |
4 +-----+-----+-----+-----+-----+-----+
5 | listnumber     | int           | NO   | PRI | NULL    |       |
6 | supplierid     | int           | NO   |     | NULL    |       |
7 | stocknumber    | int           | NO   |     | NULL    |       |
8 | importtype     | int           | YES  |     | 1       |       |
9 | importquantity | decimal(10,3) | YES  |     | NULL    |       |
10 | importvalue    | decimal(10,2) | YES  |     | NULL    |       |
11 | recorder       | int           | YES  |     | NULL    |       |
12 | recordingdate  | datetime      | YES  |     | NULL    |       |
13 | confirmer      | int           | YES  |     | NULL    |       |
14 | confirmdate    | datetime      | YES  |     | NULL    |       |
15 +-----+-----+-----+-----+-----+-----+

```

```
16  10 rows in set (0.02 sec)
```

我们还可以通过 SQL 语句**向表中添加一个字段**，我们甚至可以指定添加字段在表中的位置。

比如说，在字段 supplierid 之后，添加一个字段 suppliername，数据类型是 TEXT。我们可以这样写 SQL 语句：

复制代码

```
1 ALTER TABLE demo.importheadhist
2 ADD suppliername TEXT AFTER supplierid;
```

运行 SQL 语句，查看表结构：

复制代码

```
1 mysql> DESCRIBE demo.importheadhist;
2 +-----+-----+-----+-----+-----+-----+
3 | Field          | Type          | Null | Key | Default | Extra |
4 +-----+-----+-----+-----+-----+-----+
5 | listnumber     | int           | NO   | PRI | NULL    |       |
6 | supplierid     | int           | NO   |     | NULL    |       |
7 | suppliername   | text          | YES  |     | NULL    |       |
8 | stocknumber    | int           | NO   |     | NULL    |       |
9 | importtype     | int           | YES  |     | 1       |       |
10 | importquantity | decimal(10,3) | YES  |     | NULL    |       |
11 | importvalue    | decimal(10,2) | YES  |     | NULL    |       |
12 | recorder       | int           | YES  |     | NULL    |       |
13 | recordingdate  | datetime      | YES  |     | NULL    |       |
14 | confirmer      | int           | YES  |     | NULL    |       |
15 | confirmdate    | datetime      | YES  |     | NULL    |       |
16 +-----+-----+-----+-----+-----+-----+
17 11 rows in set (0.02 sec)
```

到这里，我们就完成了修改字段在表中位置的操作。

## 总结

这节课，我们学习了创建和修改数据表的具体方法。在讲创建表时，我讲到了一个重要的概念，就是约束，包括默认约束、非空约束、唯一性约束和自增约束等。

默认值约束：就是给字段设置一个默认值。


非空约束：就是声明字段不能为空值。

唯一性约束：就是声明字段不能重复。

自增约束：就是声明字段值能够自动加 1，且不会重复。

在修改表时，我们可以通过修改已经存在的表创建新表，也可以通过添加字段、修改字段的方式来修改数据表。

最后，我给你汇总了一些常用的创建表的 SQL 语句，你一定要牢记。


 复制代码

```
1 CREATE TABLE
2 (
3  字段名 字段类型 PRIMARY KEY
4 );
5 CREATE TABLE
6 (
7  字段名 字段类型 NOT NULL
8 );
9 CREATE TABLE
10 (
11  字段名 字段类型 UNIQUE
12 );
13 CREATE TABLE
14 (
15  字段名 字段类型 DEFAULT 值
16 );
17 -- 这里要注意自增类型的条件，字段类型必须是整数类型。
18 CREATE TABLE
19 (
20  字段名 字段类型 AUTO_INCREMENT
21 );
22 -- 在一个已经存在的表基础上，创建一个新表
23 CREATE demo.importtheadhist LIKE demo.importthead;
24 -- 修改表的相关语句
25 ALTER TABLE 表名 CHANGE 旧字段名 新字段名 数据类型;
26 ALTER TABLE 表名 ADD COLUMN 字段名 字段类型 FIRST|AFTER 字段名;
27 ALTER TABLE 表名 MODIFY 字段名 字段类型 FIRST|AFTER 字段名;
```

对于初学者来说，掌握了今天的内容，就足够对数据表进行操作了。不过，MySQL 支持的数据表操作不只这些，我来举几个简单的小例子，你可以了解一下，有个印象。



比如，你可以在表一级指定表的存储引擎：

 复制代码

```
1 ALTER TABLE 表名 ENGINE=INNODB;
```

你还可以通过指定关键字 `AUTO_EXTENDSIZE`，来指定存储文件自增空间的大小，从而提高存储空间的利用率。

在 MySQL 8.0.23 之后的版本中，你甚至还可以通过 `INVISIBLE` 关键字，使字段不可见，但却可以使用。

如果你想了解更多有关数据表的操作，我也给你提供两份资料：[MySQL 创建表文档](#)和[MySQL 修改表文档](#)。这些都是 MySQL 的官方文档，相信会对你有所帮助。

## 思考题

请你写一个 SQL 语句，将表 `demo.goodsmaster` 中的字段 `“salesprice”` 改成不能重复，并且不能为空。

欢迎在留言区写下你的思考和答案，我们一起交流讨论。如果你觉得今天的内容对你有所帮助，欢迎你把它分享给你的朋友或同事，我们下节课见。

提建议

12.12 大促

# 每日一课 VIP 年卡

10分钟，解决你的技术难题

¥159/年 ¥365/年

每日一课  
VIP 年卡

仅3天，【点击】图片，立即抢购 >>>

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 02 | 字段：这么多字段类型，该怎么定义？

下一篇 04 | 增删改查：如何操作表中的数据？

## 精选留言 (13)

写留言



朱晓峰 置顶

2021-03-29

你好，我是朱晓峰，下面我就来公布一下上节课思考题的答案：

上节课，我们学习了MySQL中的字段类型，包括整数类型、浮点数和定点数类型、文本类型、日期和时间类型等。下面是思考题的答案：

...

展开 ∨



1



lesserror

2021-03-13

朱老师，周末好。

表 demo.importhead 中插入数据的时候，SQL 语句是不是少了两条插入记录？文稿中你查询出来的是三条记录。

...

展开 ▾

作者回复: 感谢你的提醒，CREATE 语句确实少了 TABLE 关键字，我已经改过来了。我会尽快把思考题的答案发布一下，方便大家学习。



6

**右耳朵猫咪**

2021-03-15

老师好，创建表字段的时候是不是应该把多个单词用下划线分开比较好？比如，listnumber 改成 list\_number。

展开 ▾

作者回复: 命名规则应该统一，便于理解和书写。比如 listnumber，拆分成 list\_number，我认为意义不大，因为意思比较单一。

2

3

**yaomon**

2021-03-14

我们可以定义字段 “barcode” 满足唯一性约束。这样一来，条码就不能重复，但是可以为空，而且只能有一条记录条码为空。

-----

null 和任何值不相等，包括 null，所以不是可以有多条记录都是 null 值吗，是 8.0 规则变了？

展开 ▾



3

**Jun**

2021-03-24

想请教一下老师，为什么 importquantity 用 decimal 而不是 int，难道进货数量会有小数吗？

作者回复: 有的，超市有生鲜，会有小数的



1

**Harry**

2021-03-17

change 和 first/after 学习了

展开 ∨

1

1

**小鱼干**

2021-03-15

int类型，不写字段大小和写字段大小实际存储是一样吗

展开 ∨

作者回复: int 后面指定的那个数字表示显示的位数，与实际存储空间无关，比如int(4)，表示显示4位，如果不足4位，在左边用空格补齐。



1

1

**星空下**

2021-03-13

这节好实用，都是开发中在用的操作

展开 ∨

作者回复: 在实践中学习，记得牢



1

**小白**

2021-03-23

ALTER TABLE demo.goodsmaster MODIFY salesprice INT UNIQUE;  
-- 不能重复设置如上，非空设置不成功。没有找到好的解决办法。望老师解答

**Harry**

2021-03-17

交作业：

alter table demo.goodsmaster

modify salesprice decimal(10,3) not null...

展开 ∨



王坤祥

2021-03-14

goodsmaster表结构里面突然就多出几个字段，导致插入失败。另外，复制表结构的语句也有问题。

跟其他那些优秀的课程还有一点差距。



青生先森

2021-03-13

老师你好，修改表结构是不是对性能有影响呢？比如页分裂或者合并。

作者回复: 是的



Better

2021-03-13

老师有个问题，虽然建表改表语句是基础，但是，现在开发很多年的人都未必能准确快速的用命令操作，，基本都是用可视化的方式去修改配置，datagrip, navicat, workbench等，这些dms,集成了强大的可视化功能，那么记住这些操作命令有必要么？(当然对于dba,命令行界面的服务器管理，这个是特别必要的)

展开 ∨

作者回复: 可视化的集成工具使用方便，容易上手。但是这些工具对平台资源要求较高，命令行不受这些限制，建议你尽量掌握。

