NSD DBA1 DAY02

1 修改表结构

1.1 问题

本案例要求熟悉MySQL库中表的字段修改，主要练习以下操作：

- 添加字段
- 修改字段名
- 修改字段类型
- 删除字段

1.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：添加字段

在studb中创建tea6表

```
mysql> CREATE TABLE studb.tea6(id int(4) PRIMARY KEY,
    -> name varchar(4) NOT NULL,
    -> age int(2) NOT NULL
    -> );
Query OK, 0 rows affected (0.34 sec)
```

为tea6表添加一个address字段

添加前：

```
mysql> DESC tea6;
+-------+------------+------+-----+---------+-------+
| Field | Type       | Null | Key | Default | Extra |
+-------+------------+------+-----+---------+-------+
| id    | int(4)     | NO   | PRI | NULL    |       |
| name  | varchar(4) | NO   |     | NULL    |       |
```

Top

```
| age   | int(2)     | NO   |     | NULL    |       |
+-------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

添加address字段：

```
mysql> ALTER TABLE tea6 ADD address varchar(48);
Query OK, 0 rows affected (0.84 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

添加后（默认作为最后一个字段）：

```
mysql> DESC tea6;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| id      | int(4)      | NO   | PRI | NULL    |       |
| name    | varchar(4)  | NO   |     | NULL    |       |
| age     | int(2)      | NO   |     | NULL    |       |
| address | varchar(48) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

3）在tea6表的age列之后添加一个gender字段

添加操作：

```
mysql> ALTER TABLE tea6 ADD gender enum('boy','girl') AFTER age;
Query OK, 0 rows affected (0.59 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

确认添加结果：

Top

```
mysql> DESC tea6;
+---------+-------------------+------+-----+---------+-------+
| Field   | Type              | Null | Key | Default | Extra |
+---------+-------------------+------+-----+---------+-------+
| id      | int(4)            | NO   | PRI | NULL    |       |
| name    | varchar(4)        | NO   |     | NULL    |       |
| age     | int(2)            | NO   |     | NULL    |       |
| gender  | enum('boy','girl') | YES  |     | NULL    |       |
| address | varchar(48)       | YES  |     | NULL    |       |
+---------+-------------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

步骤二：修改字段名和字段类型

将tea6表的gender字段改名为sex，并添加非空约束

修改操作：

```
mysql> ALTER TABLE tea6 CHANGE gender
    -> sex enum('boy','girl') NOT NULL;
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

确认修改结果：

```
mysql> DESC tea6;
+---------+-------------------+------+-----+---------+-------+
| Field   | Type              | Null | Key | Default | Extra |
+---------+-------------------+------+-----+---------+-------+
| id      | int(4)            | NO   | PRI | NULL    |       |
| name    | varchar(4)        | NO   |     | NULL    |       |
| age     | int(2)            | NO   |     | NULL    |       |
| sex     | enum('boy','girl') | NO   |     | NULL    |       |
| address | varchar(48)       | YES  |     | NULL    |       |
+---------+-------------------+------+-----+---------+-------+
```

Top

```
11.    5 rows in set (0.00 sec)
```

步骤三：删除字段

删除tea6表中名为sex的字段：

```
01.    mysql> ALTER TABLE tea6 DROP sex;                    //删除操作
02.    Query OK, 0 rows affected (0.52 sec)
03.    Records: 0  Duplicates: 0  Warnings: 0
04.
05.    mysql> DESC tea6;                                    //确认删除结果
06.    +---------+-------------+------+-----+---------+-------+
07.    | Field   | Type        | Null | Key | Default | Extra |
08.    +---------+-------------+------+-----+---------+-------+
09.    | id      | int(4)      | NO   | PRI | NULL    |       |
10.    | name    | varchar(4)  | NO   |     | NULL    |       |
11.    | age     | int(2)      | NO   |     | NULL    |       |
12.    | address | varchar(48) | YES  |     | NULL    |       |
13.    +---------+-------------+------+-----+---------+-------+
14.    4 rows in set (0.00 sec)
```

2 MySQL索引创建与删除

2.1 问题

本案例要求熟悉MySQL索引的类型及操作方法，主要练习以下任务：

- 普通索引、唯一索引、主键索引的创建/删除
- 自增主键索引的创建/删除
- 建立员工表yg、工资表gz，数据内容如表-1、表-2所示，设置外键实现同步更新与同步删除

表-1 员工表yg的数据

yg_id	name
1	Jerry
2	Tom

Top

表-2 工资表gz的数据

gz_id	Name	gz
1	Jerry	12000
2	Tom	8000

2.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：索引的创建与删除

创建表的时候指定INDEX索引字段

创建库home：

```
mysql> create database home;
Query OK, 1 row affected (0.00 sec)
```

允许有多个INDEX索引字段。比如，以下操作在home库中创建了tea4表，将其中的id、name作为索引字段：

```
mysql> USE home;
Database changed
mysql> CREATE TABLE tea4(
    -> id char(6) NOT NULL,
    -> name varchar(6) NOT NULL,
    -> age int(3) NOT NULL,
    -> gender ENUM('boy','girl') DEFAULT 'boy',
    -> INDEX(id),INDEX(name)
    -> );
Query OK, 0 rows affected (0.59 sec)
```

查看新建tea4表的字段结构，可以发现两个非空索引字段的KEY标志为MUL：

```
mysql> DESC tea4;
+--------+--------------------+------+-----+---------+-------+
| Field  | Type               | Null | Key | Default | Extra |
+--------+--------------------+------+-----+---------+-------+
```

Top

```
| id     | char(6)           | NO   | MUL | NULL    |       |
| name   | varchar(6)        | NO   | MUL | NULL    |       |
| age    | int(3)            | NO   |     | NULL    |       |
| gender | enum('boy','girl') | YES  |     | boy     |       |
+--------+-------------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

2）删除现有表的某个INDEX索引字段

比如，删除tea4表中名称为named的INDEX索引字段：

```
mysql> drop INDEX name ON tea4;                    //删除name字段的索引
Query OK, 0 rows affected (0.18 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> DESC tea4;                                  //确认删除结果
+--------+-------------------+------+-----+---------+-------+
| Field  | Type              | Null | Key | Default | Extra |
+--------+-------------------+------+-----+---------+-------+
| id     | char(6)           | NO   | MUL | NULL    |       |
| name   | varchar(6)        | NO   |     | NULL    |       |
| age    | int(3)            | NO   |     | NULL    |       |
| gender | enum('boy','girl') | YES  |     | boy     |       |
+--------+-------------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

3）在已有的某个表中设置INDEX索引字段

比如，针对tea4表的age字段建立索引，名称为 nianling：

```
mysql> CREATE INDEX nianling ON tea4(age);     //针对指定字段创建索
Query OK, 0 rows affected (0.62 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> DESC tea4;                              //确认创建结果 Top
+--------+-------------------+------+-----+---------+-------+
```

```
| Field  | Type              | Null | Key | Default | Extra |
+--------+-------------------+------+-----+---------+-------+
| id     | char(6)           | NO   | MUL | NULL    |       |
| name   | varchar(6)        | NO   |     | NULL    |       |
| age    | int(3)            | NO   | MUL | NULL    |       |
| gender | enum('boy','girl') | YES  |     | boy     |       |
+--------+-------------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

4）查看指定表的索引信息

使用SHOW INDEX 指令：

```
mysql> SHOW INDEX FROM tea4\G
*************************** 1. row ***********************
        Table: tea4
   Non_unique: 1
     Key_name: id
  Seq_in_index: 1
  Column_name: id
    Collation: A
  Cardinality: 0
     Sub_part: NULL
       Packed: NULL
         Null:
   Index_type: BTREE                    //使用B树算法
      Comment:
Index_comment:
*************************** 2. row ***********************
        Table: tea4
   Non_unique: 1
     Key_name: nianling                 //索引名称
  Seq_in_index: 1
  Column_name: age                      //字段名称
    Collation: A                                          Top
  Cardinality: 0
     Sub_part: NULL
```

```
25.           Packed: NULL
26.             Null:
27.       Index_type: BTREE
28.          Comment:
29.    Index_comment:
30.    2 rows in set (0.00 sec)
```

5）创建表的时候指定UNIQUE索引字段

UNIQUE表示唯一性的意思，同一个表中可以有多个字段具有唯一性。

比如，创建tea5表，将id、name字段建立设置UNIQUE索引，age字段设置INDEX索引：

```
01.    mysql> CREATE TABLE tea5(
02.        -> id char(6),
03.        -> name varchar(4) NOT NULL,
04.        -> age int(3) NOT NULL,
05.        -> UNIQUE(id),UNIQUE(name),INDEX(age)
06.        -> );
07.    Query OK, 0 rows affected (0.30 sec)
```

查看新建tea5表的字段结构，可发现UNIQUE字段的KEY标志为UNI；另外，由于字段name必须满足“NOT NULL”的非空约束，所以将其设置为UNIQUE后会自动变成了PRIMARY KEY主键字段：

```
01.    mysql> DESC tea5;                              //确认设置结果
02.    +-------+------------+------+-----+---------+-------+
03.    | Field | Type       | Null | Key | Default | Extra |
04.    +-------+------------+------+-----+---------+-------+
05.    | id    | char(6)    | YES  | UNI | NULL    |       |
06.    | name  | varchar(4) | NO   | PRI | NULL    |       |
07.    | age   | int(3)     | NO   | MUL | NULL    |       |
08.    +-------+------------+------+-----+---------+-------+
09.    3 rows in set (0.03 sec)
```

Top

6）删除UNIQUE索引、在已有的表中设置UNIQUE索引字段

先删除tea5表name字段的唯一索引（与删除INDEX索引的方法相同）：

```
mysql> DROP INDEX name ON tea5;                    //清除UNIQUE索引
Query OK, 0 rows affected (0.97 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> DESC tea5;                                  //确认删除结果
+-------+------------+------+-----+---------+-------+
| Field | Type       | Null | Key | Default | Extra |
+-------+------------+------+-----+---------+-------+
| id    | char(6)    | YES  | UNI | NULL    |       |
| name  | varchar(4) | NO   |     | NULL    |       |
| age   | int(3)     | NO   | MUL | NULL    |       |
+-------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

重新为tea5表的name字段建立UNIQUE索引，并确认结果：

```
mysql> CREATE UNIQUE INDEX name ON tea5(name);     //建立UNIQUE
Query OK, 0 rows affected (0.47 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> DESC tea5;                                  //确认设置结果
+-------+------------+------+-----+---------+-------+
| Field | Type       | Null | Key | Default | Extra |
+-------+------------+------+-----+---------+-------+
| id    | char(6)    | YES  | UNI | NULL    |       |
| name  | varchar(4) | NO   | PRI | NULL    |       |
| age   | int(3)     | NO   | MUL | NULL    |       |
+-------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Top

7）建表时设置PRIMARY KEY主键索引

主键索引实际上在前面已经接触过了，建表的时候可以直接指定。如果表内一开始没

有主键字段，则新设置的非空UNIQUE字段相当于具有PRIMARY KEY主键约束。

每个表中的主键字段只能有一个。

建表的时候，可以直接在某个字段的“约束条件”部分指定PRIMARY KEY；也可以在最后指定PRIMARY KEY(某个字段名)。比如：

```
mysql> CREATE TABLE biao01(
    -> id int(4) PRIMARY KEY,                    //直接在字段定义时约束
    -> name varchar(8)
    -> );
Query OK, 0 rows affected (0.19 sec)
```

或者：

```
mysql> CREATE TABLE biao02(
    -> id int(4),
    -> name varchar(8),
    -> PRIMARY KEY(id)                           //所有字段定义完，最后指定
    -> );
Query OK, 0 rows affected (0.17 sec)
```

在建表的时候，如果主键字段为int类型，还可以为其设置AUTO_INCREMENT自增属性，这样当添加新的表记录时，此字段的值会自动从1开始逐个增加，无需手动指定。比如，新建一个tea6表，将id列作为自增的主键字段：

```
mysql> CREATE TABLE tea6(
    -> id int(4) AUTO_INCREMENT,
    -> name varchar(4) NOT NULL,
    -> age int(2) NOT NULL,
    -> PRIMARY KEY(id)
    -> );
Query OK, 0 rows affected (0.29 sec)
```

Top

8）删除现有表的PRIMARY KEY主键索引

如果要移除某个表的PRIMARY KEY约束，需要通过ALTER TABLE指令修改。比如，以

下操作将清除biao01表的主键索引。

清除前（主键为id）：

```
mysql> DESC biao01;
+-------+------------+------+-----+---------+-------+
| Field | Type       | Null | Key | Default | Extra |
+-------+------------+------+-----+---------+-------+
| id    | int(4)     | NO   | PRI | NULL    |       |
| name  | varchar(8) | YES  |     | NULL    |       |
+-------+------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

清除操作：

```
mysql> ALTER TABLE biao01 DROP PRIMARY KEY;
Query OK, 0 rows affected (0.49 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

清除后（无主键）：

```
mysql> DESC biao01;
+-------+------------+------+-----+---------+-------+
| Field | Type       | Null | Key | Default | Extra |
+-------+------------+------+-----+---------+-------+
| id    | int(4)     | NO   |     | NULL    |       |
| name  | varchar(8) | YES  |     | NULL    |       |
+-------+------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

当尝试删除tea6表的主键时，会出现异常：

Top

```
mysql> ALTER TABLE tea6 DROP PRIMARY KEY;
ERROR 1075 (42000): Incorrect table definition; there can be only one a
```

这是因为tea6表的主键字段id具有AUTO_INCREMNET自增属性，提示这种字段必须作为主键存在，因此若要清除此主键必须先清除自增属性——修改id列的字段定义：

```
mysql> ALTER TABLE tea6 MODIFY id int(4) NOT NULL;
Query OK, 0 rows affected (0.75 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

然后再清除主键属性就OK了：

```
mysql> ALTER TABLE tea6 DROP PRIMARY KEY;                    //清除主键
Query OK, 0 rows affected (0.39 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc tea6;                                            //确认清除结果
+-------+------------+------+-----+---------+-------+
| Field | Type       | Null | Key | Default | Extra |
+-------+------------+------+-----+---------+-------+
| id    | int(4)     | NO   |     | NULL    |       |
| name  | varchar(4) | NO   |     | NULL    |       |
| age   | int(2)     | NO   |     | NULL    |       |
+-------+------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

9）为现有表添加PRIMARY KEY主键索引

重新为tea6表指定主键字段，仍然使用id列：

```
mysql> ALTER TABLE tea6 ADD PRIMARY KEY(id);                 //设置主键字
Query OK, 0 rows affected (0.35 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Top

```
04.
05.    mysql> DESC tea6;                                    //确认设置结果
06.    +-------+------------+------+-----+---------+-------+
07.    | Field | Type       | Null | Key | Default | Extra |
08.    +-------+------------+------+-----+---------+-------+
09.    | id    | int(4)     | NO   | PRI | NULL    |       |
10.    | name  | varchar(4) | NO   |     | NULL    |       |
11.    | age   | int(2)     | NO   |     | NULL    |       |
12.    +-------+------------+------+-----+---------+-------+
13.    3 rows in set (0.00 sec)
```

步骤二：创建数据库并设置外键实现同步更新与同步删除

根据实验任务要求，两个表格的字段结构如表-1、表-2所示。

1）创建yg表，用来记录员工工号、姓名

其中yg_id列作为主键，并设置自增属性

```
01.    mysql> CREATE TABLE yg(
02.        -> yg_id int(4) AUTO_INCREMENT,
03.        -> name char(16) NOT NULL,
04.        -> PRIMARY KEY(yg_id)
05.        -> );
06.    Query OK, 0 rows affected (0.15 sec)
```

2）创建gz表，用来记录员工的工资信息

其中gz_id需要参考员工工号，即gz表的gz_id字段设为外键，将yg表的yg_id字段作为参考键：

```
01.    mysql> CREATE TABLE gz(
02.        -> gz_id int(4) NOT NULL,
03.        -> name char(16) NOT NULL,
04.        -> gz float(7,2) NOT NULL DEFAULT 0,
05.        -> INDEX(name),
06.        -> FOREIGN KEY(gz_id) REFERENCES yg(yg_id)
07.        -> ON UPDATE CASCADE ON DELETE CASCADE
```

Top

```
08.     -> );
09.  Query OK, 0 rows affected (0.23 sec)
```

3）为yg表添加2条员工信息记录

因yg_id有AUTO_INCREMENT属性，会自动填充，所以只要为name列赋值就可以了。

插入表记录可使用INSERT指令，这里先执行下列操作，具体在下一章学习：

```
01.  mysql> INSERT INTO yg(name) VALUES('Jerry'),('Tom');
02.  Query OK, 2 rows affected (0.16 sec)
03.  Records: 2  Duplicates: 0  Warnings: 0
```

确认yg表的数据记录：

```
01.  mysql> SELECT * FROM yg;
02.  +-------+-------+
03.  | yg_id | name  |
04.  +-------+-------+
05.  |     1 | Jerry |
06.  |     2 | Tom   |
07.  +-------+-------+
08.  2 rows in set (0.00 sec)
```

4）为gz表添加2条工资信息记录

同上，数据参考图-2，插入相应的工资记录（gz_id字段未指定默认值，也未设置自增属性，所以需要手动赋值）：

```
01.  mysql> INSERT INTO gz(gz_id,name,gz)
02.     -> VALUES(1,'Jerry',12000),(2,'Tom',8000)
03.     -> ;
04.  Query OK, 2 rows affected (0.06 sec)
05.  Records: 2  Duplicates: 0  Warnings: 0
```

Top

确认gz表的数据记录：

```
mysql> SELECT * FROM gz;
+-------+-------+----------+
| gz_id | name  | gz       |
+-------+-------+----------+
|     1 | Jerry | 12000.00 |
|     2 | Tom   |  8000.00 |
+-------+-------+----------+
2 rows in set (0.05 sec)
```

5）验证表记录的UPDATE更新联动

将yg表中Jerry用户的yg_id修改为1234：

```
mysql> update yg SET yg_id=1234 WHERE name='Jerry';
Query OK, 1 row affected (0.05 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

确认修改结果：

```
mysql> SELECT * FROM yg;
+-------+-------+
| yg_id | name  |
+-------+-------+
|     2 | Tom   |
|  1234 | Jerry |
+-------+-------+
2 rows in set (0.00 sec)
```

同时也会发现，gz表中Jerry用户的gz_id也跟着变了：

Top

```
mysql> SELECT * FROM gz;
+-------+-------+----------+
```

```
| gz_id | name  | gz       |
+-------+-------+----------+
|  1234 | Jerry | 12000.00 |
|     2 | Tom   |  8000.00 |
+-------+-------+----------+
2 rows in set (0.00 sec)
```

6）验证表记录的DELETE删除联动

删除yg表中用户Jerry的记录：

```
mysql> DELETE FROM yg WHERE name='Jerry';
Query OK, 1 row affected (0.05 sec)
```

确认删除结果：

```
mysql> SELECT * FROM yg;
+-------+------+
| yg_id | name |
+-------+------+
|     2 | Tom  |
+-------+------+
1 row in set (0.00 sec)
```

查看gz表中的变化（Jerry的记录也没了）：

```
mysql> SELECT * FROM gz;
+-------+------+---------+
| gz_id | name | gz      |
+-------+------+---------+
|     2 | Tom  | 8000.00 |
+-------+------+---------+
1 row in set (0.00 sec)
```

Top

7）删除指定表的外键约束

先通过SHOW指令获取表格的外键约束名称：

```
mysql> SHOW CREATE TABLE gz\G
*************************** 1. row ***********************
       Table: gz
Create Table: CREATE TABLE `gz` (
  `gz_id` int(4) NOT NULL,
  `name` char(16) NOT NULL,
  `gz` float(7,2) NOT NULL DEFAULT '0.00',
  KEY `name` (`name`),
  KEY `gz_id` (`gz_id`),
  CONSTRAINT `gz_ibfk_1` FOREIGN KEY (`gz_id`) REFERENCES `yg` (`yg
) ENGINE=InnoDB DEFAULT CHARSET=utf8
1 row in set (0.00 sec)
```

其中gz_ibfk_1即删除外键约束时要用到的名称。

删除操作：

```
mysql> ALTER TABLE gz DROP FOREIGN KEY gz_ibfk_1;
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

确认删除结果：

```
mysql> SHOW CREATE TABLE gz\G
*************************** 1. row ***********************
       Table: gz
Create Table: CREATE TABLE `gz` (
  `gz_id` int(4) NOT NULL,
  `name` char(16) NOT NULL,
  `gz` float(7,2) NOT NULL DEFAULT '0.00',
  KEY `name` (`name`),
  KEY `gz_id` (`gz_id`)
```

Top

```
10.     ) ENGINE=InnoDB DEFAULT CHARSET=utf8
11.     1 row in set (0.00 sec)
```

Top