实验二 MySQL 建表实验-电子杂志订阅表

一、实验目的

能根据文字提示，完成对应数据表的创建，并可以对数据表中的数据进行简单的增、删、改、查操作

二、实验需求

1. 在 mysql 数据库中新建一张电子杂志订阅表（subscribe）
2. 电子杂志订阅表中要包含 4 个字段，分别为编号（id）、订阅右键的邮箱地址（email）、用户是否确认订阅（status，使用数字 1 表示已确认，0 表示未确认）、邮箱确认的验证码（code）。
3. 为杂志订阅表添加 5 条测试数据，见课本中的表。

三、实验要求

1. 动手完成实验
2. 完成实验报告

四、实验过程及结果

步骤 1:启动数据库，输入 net start mysql 就能启动数据库。启动的结果如下图 1 所示。

```
C:\WINDOWS\system32>net start mysql
MySQL 服务正在启动 .
MySQL 服务已经启动成功。
```

图 1 数据库启动成功

步骤 2：登录数据库，输入 mysql -u root -p 就能进入数据库登录验证步骤，之后在其上输入登陆密码，能能够成功登入数据库。登陆成功截图如下图 2 所示。

```
C:\WINDOWS\system32>mysql -u root -p
Enter password: ********
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.31-log MySQL Community Server (GPL)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

图 2 登录数据库成功

步骤 3：创建数据库，如果要打开的的数据库不存在，则需要创建。由于本人数据库并未创建过，因此没有现存数据库供选择，因此需要先创建数据库。创建的命令为：CREATE DATABASE mydb;但是考虑到如果本人忘记了，以前的确创建了数据库，而且名字也叫 mydb，此时为了避免程序报错，需要做一个万全考虑，因此在创建的时候加一个判断 mydb 是否存在的语句，若不存在就创建 mydb，命令为：CREATE DATABASE IF NOT EXISTS mydb;不报错表示

已经成功。成功的截图如下图 3 所示。

```
mysql> CREATE DATABASE IF NOT EXISTS mydb;
Query OK, 1 row affected, 1 warning (0.00 sec)
```

图 3 创建 mydb 数据库成功

步骤 4：使用数据库。就是选择数据库到当前操作的对象。比如当数据库中有两个对象，一个是 mydb 和 mydb1，如果当前是在 mydb1 的状态下，需要修改 mydb 的内容，就需要修改数据库状态。命令为 USE + 数据库名。如下图 4 所示。提示 Database changed 表示切换成功。

```
mysql> USE mydb
Database changed
```

图 4 切换数据库成功

步骤 5：创建电子杂志订阅表，此订阅表中包括四个字段。通过“CRAEATE TABLE+表格名称”的命令完成创建。格式为字段代表的含义、邮箱地址、状态（是否确认）、验证码，最后需要确定编码方式为 UTF-8，否则有可能发生乱码。没有报错表示已经创建成功。当我继续创建 subscribe 为名称时发生了系统报错“Table 'subscribe' already exists”，这是由于我之前创建过 subscribe 表，因此在这次复现过程中，将该表格命名为 subscribe1 以示区别。创建成功的截图如下图 5 所示。

```
mysql> CREATE TABLE subscribe(
    -> id INT COMMENT '编号',
    -> email VARCHAR(60) COMMENT '邮件订阅的邮箱地址',
    -> status INT COMMENT '是否确认，0未确认，1已确认',
    ->  code VARCHAR(10) COMMENT '邮箱确认的验证码'
    ->  )DEFAULT CHARSET=utf8;
ERROR 1050 (42S01): Table 'subscribe' already exists
mysql> CREATE TABLE subscribe1(
    -> id INT COMMENT '编号',
    -> email VARCHAR(60) COMMENT '邮件订阅的邮箱地址',
    -> status INT COMMENT '是否确认，0未确认，1已确认',
    ->  code VARCHAR(10) COMMENT '邮箱确认的验证码'
    ->  )DEFAULT CHARSET=utf8;
Query OK, 0 rows affected (0.03 sec)
```

图 5 表格创建成功

步骤 6：向步骤 5 创建好的表格中添加数据，由于上述表格中规定了四个字段，因此添加数据的时候也需要对应着添加四个字段。添加的内容必须要和字段需要的类型一致，否则会报错。如图 6 是添加字段时，由于需要字段类型不一致报错的截图，步骤 5 中要求的是第三个字段必须是整型数字类型，而在实际输入中不是；第四个字段要求的是字符串类型，但是实际输入的并不是字符串（字符串要用英文引号”引起来），因此会图 6 的错误。若字段超出了关于规定的字段时，也会报错。如图 7 是添加的字段长度大于规定的字段长度时的报错。同理，当字段少于规定字段时也会报相同的错误。因此这一步要万分小心谨慎。最终没有报错表示输入成功，建表成功。如图 8 所示。

```
mysql> INSERT INTO subscribe VALUES
    -> (1, 'gosling123456@qq.com', a, QAZWSX);
ERROR 1054 (42S22): Unknown column 'a' in 'field list'
```

图 6 添加字段类型不同时报错

```
mysql> INSERT INTO subscribe VALUES
    -> (1, 'gosling123456@qq.com', a, QAZWSX,1);
ERROR 1136 (21S01): Column count doesn't match value count at row 1
```

图 7 字段溢出报错

```
mysql> INSERT INTO subscribe VALUES(1, 'gosling123456@qq.com', 1, 'QAZWSX'),(2, '1336245271@qq.com', 1
, 'EDCRFV'),(3, 'gosling123456@lookout.com', 0, 'YHNTGB'),(4, 'gosling123456@github.com', 0, 'UJMRFV')
,(5, 'gosling123456@163.com', 1, 'TGBEDC');
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

图 8 向表格中添加数据成功

步骤 7： 验证步骤 6 是否成功，可以通过查看表 subscribe1 的内容来完成。可以通过通配符*来全选所有的数据。输入命令为：SELECT * FROM subscribe1; 结果如图 9 所示。从图表中能非常清晰的看出在步骤 5 创建的字段相当于第一行的所有列坐标。在步骤 6 中添加的数据是每一行数据，分别于步骤 5 的字段对应，规范整齐。

```
mysql> SELECT * FROM subscribe1;
+------+--------------------------+--------+--------+
| id   | email                    | status | code   |
+------+--------------------------+--------+--------+
|    1 | gosling123456@qq.com     |      1 | QAZWSX |
|    2 | 1336245271@qq.com        |      1 | EDCRFV |
|    3 | gosling123456@lookout.com|      0 | YHNTGB |
|    4 | gosling123456@github.com |      0 | UJMRFV |
|    5 | gosling123456@163.com    |      1 | TGBEDC |
+------+--------------------------+--------+--------+
5 rows in set (0.00 sec)
```

图 9 数据添加成功后显示的表格

步骤 8：实现查找的操作。在上述的表 subscribe1 中，已经有四个字段，因此可以通过四个字段和他们对应的值进行查找。其格式为“SELECT * FROM + 表格名”。在这里，表二名为 subscribe1。图 10 为按照 status 字段为 1 查找的订阅内容、图 11 为按照 id 字段为 3 查找的订阅内容、图 12 为按照 email 名称查找 email 名为 gosling123456@qq.com 的订阅内容、图 13 为按照验证码 code 名称查找 code 名为 EDCRFV 的订阅内容。

```
mysql> SELECT * FROM subscribe1 WHERE status=1;
+------+-------------------------+--------+--------+
| id   | email                   | status | code   |
+------+-------------------------+--------+--------+
|    1 | gosling123456@qq.com    |      1 | QAZWSX |
|    2 | 1336245271@qq.com       |      1 | EDCRFV |
|    5 | gosling123456@163.com   |      1 | TGBEDC |
+------+-------------------------+--------+--------+
3 rows in set (0.00 sec)
```

图 10 按照 status 查找 status 为 1 的订阅内容

```
mysql> SELECT * FROM subscribe1 WHERE id = 3;
+------+--------------------------+--------+--------+
| id   | email                    | status | code   |
+------+--------------------------+--------+--------+
|    3 | gosling123456@lookout.com|      0 | YHNTGB |
+------+--------------------------+--------+--------+
1 row in set (0.00 sec)
```

图 11 按照 id 查找 id 为 3 的订阅内容

```
mysql> SELECT * FROM subscribe1 WHERE email = 'gosling123456@qq.com';
+------+----------------------+--------+--------+
| id   | email                | status | code   |
+------+----------------------+--------+--------+
|    1 | gosling123456@qq.com |      1 | QAZWSX |
+------+----------------------+--------+--------+
1 row in set (0.00 sec)
```

图 12 按照 email 名称查找 email 名为 gosling23456@qq.com 的内容

```
mysql> SELECT * FROM subscribe1 WHERE code = 'EDCRFV';
+------+--------------------+--------+--------+
| id   | email              | status | code   |
+------+--------------------+--------+--------+
|    2 | 13362452?1@qq.com  |      1 | EDCRFV |
+------+--------------------+--------+--------+
1 row in set (0.00 sec)
```

图 13 按照 code 名称查找 code 为 EDCRFV 的订阅内容

步骤 9：实现修改操作。在输入时有个在 email 字段下的订阅邮箱输入错了，其原本的形式应该是'gosling123456@outlook.com'，但是在第 6 步，向建好的表里面添加字段的时候输入成'gosling123456@lookout.com'，因此需要将其改正过来。因此需要选择数据库中的 UPDATA 关键词命令。输入 UPDATE subscribe1 SET + "需要修改的数据属于的字段" = + "要修改为的字段的内容" + WHERE + "主键的字段" = + "需要修改的对象主键字段的内容"。例如需要将 email 字段内容为'gosling123456@lookout.com'的订阅邮箱修改为'gosling123456@outlook.com'的操作为：UPDATE subscribe1 SET email = 'gosling123456@outlook.com' WHERE email = 'gosling123456@lookout.com'。修改前后的截图如下图 14 和图 15 所示，其中图 14 为修改前的 subscribe1 表，图 15 为修改后的 subscribe1 表。

```
mysql> SELECT * FROM subscribe1;
+------+--------------------------+--------+--------+
| id   | email                    | status | code   |
+------+--------------------------+--------+--------+
|    1 | gosling123456@qq.com     |      1 | QAZWSX |
|    2 | 13362452?1@qq.com        |      1 | EDCRFV |
|    3 | gosling123456@lookout.com|      0 | YHNTGB |
|    4 | gosling123456@github.com |      0 | UJMRFV |
|    5 | gosling123456@163.com    |      1 | TGBEDC |
+------+--------------------------+--------+--------+
5 rows in set (0.00 sec)
```

图 14 修改前的 subscribe1 表

```
mysql> SELECT * FROM subscribe1;
+------+--------------------------+--------+--------+
| id   | email                    | status | code   |
+------+--------------------------+--------+--------+
|    1 | gosling123456@qq.com     |      1 | QAZWSX |
|    2 | 13362452?1@qq.com        |      1 | EDCRFV |
|    3 | gosling123456@outlook.com|      0 | YHNTGB |
|    4 | gosling123456@github.com |      0 | UJMRFV |
|    5 | gosling123456@163.com    |      1 | TGBEDC |
+------+--------------------------+--------+--------+
5 rows in set (0.00 sec)
```

图 15 修改后的 subscribe1 表

步骤 10： 实现删除操作。删除的命令为 DELETE FROM + 表名 + 对应数据的相应字段和内容。例如如果要删除 subscribe1 表中的第 5 个元素，命令为 DELETE FROM subscribe1 WHERE id = 5 这样就能删除掉该组数据，结果的截

图如图 16 所示；如果要删除 subscribe1 表中的 email 为’1336245271@qq.com’的元素，命令为 DELETE FROM subscribe1 WHERE email = ’1336245271@qq.com’、这样就能删除掉该组数据，结果的截图如图 17 所示；如果要删除 subscribe1 表中的 status 为 1 的元素，命令为 DELETE FROM subscribe1 WHERE status = 1,此时所有 status 为 1 的元素都会被删除这样就能删除掉该组数据，结果的截图如图 18 所示；如果要删除 subscribe1 表中的 code 为 QAZWSX 的元素，命令为 DELETE FROM subscribe1 WHERE code = QAZWSX 这样就能删除掉该组数据，结果的截图如图 19 所示;

```
mysql> DELETE FROM subscribe1 WHERE id = 5;
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM subscribe1;
+------+---------------------------+--------+--------+
| id   | email                     | status | code   |
+------+---------------------------+--------+--------+
|    1 | gosling123456@qq.com      |      1 | QAZWSX |
|    2 | 1336245271@qq.com         |      1 | EDCRFV |
|    3 | gosling123456@outlook.com |      0 | YHNTGB |
|    4 | gosling123456@github.com  |      0 | UJMRFV |
+------+---------------------------+--------+--------+
4 rows in set (0.00 sec)
```

图 16 删除 id 为 5 的结果

```
mysql> DELETE FROM subscribe1 WHERE email = '1336245271@qq.com';
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM subscribe1;
+------+---------------------------+--------+--------+
| id   | email                     | status | code   |
+------+---------------------------+--------+--------+
|    1 | gosling123456@qq.com      |      1 | QAZWSX |
|    3 | gosling123456@outlook.com |      0 | YHNTGB |
|    4 | gosling123456@github.com  |      0 | UJMRFV |
|    5 | gosling123456@163.com     |      1 | TGBEDC |
+------+---------------------------+--------+--------+
4 rows in set (0.00 sec)
```

图 17 删除 email 为'1336245271@qq.com'的结果

```
mysql> DELETE FROM subscribe1 WHERE status = 1;
Query OK, 3 rows affected (0.04 sec)

mysql> SELECT * FROM subscribe1;
+------+---------------------------+--------+--------+
| id   | email                     | status | code   |
+------+---------------------------+--------+--------+
|    3 | gosling123456@outlook.com |      0 | YHNTGB |
|    4 | gosling123456@github.com  |      0 | UJMRFV |
+------+---------------------------+--------+--------+
2 rows in set (0.00 sec)
```

图 18 删除 status 为 1 的结果

```
mysql> DELETE FROM subscribe1 WHERE code = 'QAZWSX';
Query OK, 1 row affected (0.04 sec)

mysql> SELECT * FROM subscribe1;
+------+---------------------------+--------+---------+
| id   | email                     | status | code    |
+------+---------------------------+--------+---------+
|    2 | 1336245271@qq.com         |      1 | EDCRFV  |
|    3 | gosling123456@outlook.com |      0 | YHNTGB  |
|    4 | gosling123456@github.com  |      0 | UJMRFV  |
|    5 | gosling123456@163.com     |      1 | TGBEDC  |
+------+---------------------------+--------+---------+
4 rows in set (0.00 sec)
```

图 19 删除 code 为 QAZWSX 的结果

步骤 11：扩展操作，实现数据库中特定关键字的排序，升序和降序。排序是利用 SORT 关键词命令来实现的，其格式为：SORT * FROM + 表名 ORDER BY + 关键字段 + （升序 asc 或降序 desc）。下图分别是通过对 id、状态排序（升序和降序的操作图和操作后的表）。

```
mysql> select * from subscribe1 order by id desc;
+------+---------------------------+--------+---------+
| id   | email                     | status | code    |
+------+---------------------------+--------+---------+
|    5 | gosling123456@163.com     |      1 | TGBEDC  |
|    4 | gosling123456@github.com  |      0 | UJMRFV  |
|    3 | gosling123456@outlook.com |      0 | YHNTGB  |
|    2 | 1336245271@qq.com         |      1 | EDCRFV  |
|    1 | gosling123456@qq.com      |      1 | QAZWSX  |
+------+---------------------------+--------+---------+
5 rows in set (0.00 sec)
```

图 20 id 降序排序结果

```
mysql> select * from subscribe1 order by id asc;
+------+---------------------------+--------+---------+
| id   | email                     | status | code    |
+------+---------------------------+--------+---------+
|    1 | gosling123456@qq.com      |      1 | QAZWSX  |
|    2 | 1336245271@qq.com         |      1 | EDCRFV  |
|    3 | gosling123456@outlook.com |      0 | YHNTGB  |
|    4 | gosling123456@github.com  |      0 | UJMRFV  |
|    5 | gosling123456@163.com     |      1 | TGBEDC  |
+------+---------------------------+--------+---------+
5 rows in set (0.00 sec)
```

图 21 id 升序排序结果

```
mysql> select * from subscribe1 order by status asc;
+------+---------------------------+--------+---------+
| id   | email                     | status | code    |
+------+---------------------------+--------+---------+
|    3 | gosling123456@outlook.com |      0 | YHNTGB  |
|    4 | gosling123456@github.com  |      0 | UJMRFV  |
|    2 | 1336245271@qq.com         |      1 | EDCRFV  |
|    5 | gosling123456@163.com     |      1 | TGBEDC  |
|    1 | gosling123456@qq.com      |      1 | QAZWSX  |
+------+---------------------------+--------+---------+
5 rows in set (0.00 sec)
```

图 22 status 升序排序结果

```
mysql> select * from subscribe1 order by status desc;
+------+---------------------------+--------+--------+
| id   | email                     | status | code   |
+------+---------------------------+--------+--------+
|    2 | 1336245271@qq.com         |      1 | EDCRFV |
|    5 | gosling123456@163.com     |      1 | TGBEDC |
|    1 | gosling123456@qq.com      |      1 | QAZWSX |
|    3 | gosling123456@outlook.com |      0 | YHNTGB |
|    4 | gosling123456@github.com  |      0 | UJMRFV |
+------+---------------------------+--------+--------+
5 rows in set (0.00 sec)
```

图 23 status 降序排序结果

五、实验总结

本次实验我学会了数据库的创建、添加、删除、修改、查找和排序。本次实验主要是按照书上的步骤一个一个地敲命令，过程虽然有些繁琐，甚至还有一个步骤出现好多错误然后翻来覆去反反复复的修改，不过幸运的是，总算把所有功能都实现了，甚至还在查找资料的过程中发现了一个新的操作——排序，以及其实现方法。在敲命令的过程中，我越发觉得这和书上的内容遥相呼应，在我写到设置四个字段的时候，我想起来第一节理论课上的 excel 表的表头，当我写到 SELECT * FROM WHERE 的时候，我想到可以通过控制关键字的值（比如控制 status 为 1）来控制视图等等，通过实验，我发现理论与实践结合的越发密切了。

虽说本次实验最终是完成了，但是一路上还是花了不少功夫的。虽然是对着课本内容敲命令，但是所谓学而不思则罔，我的目标不只是为了完成作业，而是要通过完成作业来夯实数据库的基础。在写命令的过程中，我明白了，数据库的语法规则，基本上是靠关键字来实现的，例如创建数据库就是 CREATE，添加元素就是 INSERT，删除元素就是 DELETE，修改元素就是 UPDATE，查找元素就是 SELECT，其后缀都是用来确定元素的位置，以便于操作。在敲命令期间，我忘记开启大写锁定，但是写的命令还是通过了。于是我明白了，数据库的操作命令是不区分大小写的，那以后在写数据库命令的时候就不需要来回切换大小写了，也算是节省了精力。此外，我在敲命令的时候发现了*的作用，我记得在 python 语言里面，*的作用是通配符，在数据库中，*代表的是全部的意思，也算是通配符的含义。

通过本次实验，我明白了数据库的学习在于多看多学多想多动手，要将理论与实践相结合才能学好数据库，真正将数据库变成自己最熟悉的工具。