实验报告——数据库的完整性和安全性

一、实验目的

1.加深对数据库完整性的理解。
2.研究具体DBMS提供的完整性措施，通过SQL对数据进行完整性控制，掌握不同的设置数据库完整性的方法。

二、实验平台

1.数据库管理系统

MySQL 5.7.20

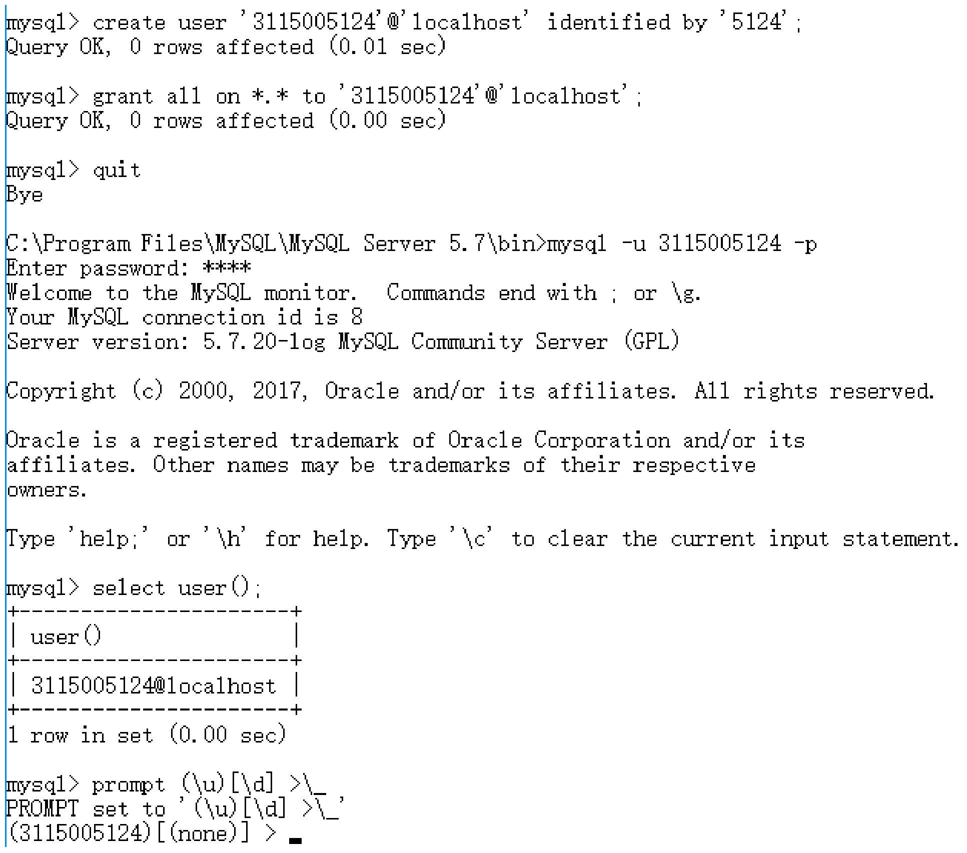
2.可视化管理工具

3.操作系统

Windows 10（企业版）

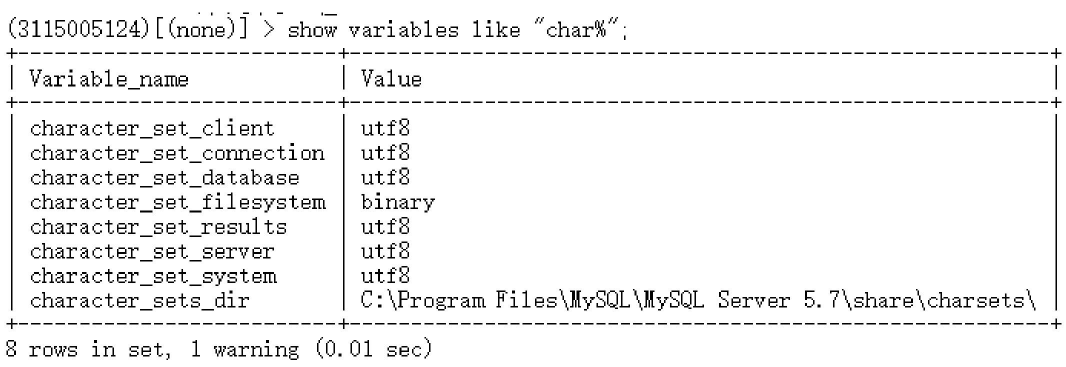
三、实验准备

创建用户:



```
mysql> create user '3115005124'@'localhost' identified by '5124';
Query OK, 0 rows affected (0.01 sec)

mysql> grant all on *.* to '3115005124'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> quit
Bye

C:\Program Files\MySQL\MySQL Server 5.7\bin>mysql -u 3115005124 -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 5.7.20-log MySQL Community Server (GPL)

Copyright (c) 2000, 2017, 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> select user();
+----------------------+
| user()               |
+----------------------+
| 3115005124@localhost |
+----------------------+
1 row in set (0.00 sec)

mysql> prompt (\u)[\d] >\_
PROMPT set to '(\u)[\d] >\_'
(3115005124)[(none)] > _
```

中文编码问题:



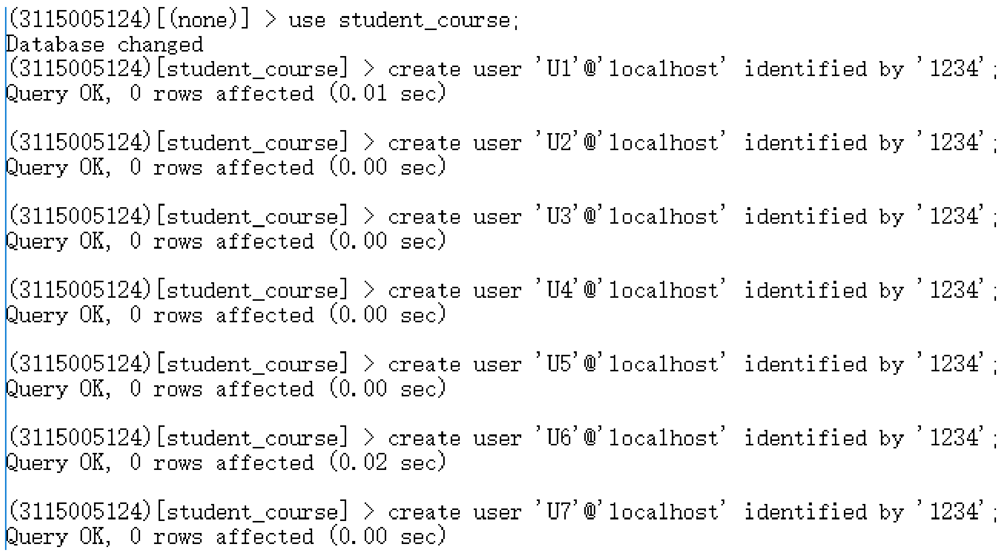
```
(3115005124)[(none)] > show variables like "char%";
+--------------------------+-----------------------------------------------------------+
| Variable_name            | Value                                                     |
+--------------------------+-----------------------------------------------------------+
| character_set_client     | utf8                                                      |
| character_set_connection | utf8                                                      |
| character_set_database   | utf8                                                      |
| character_set_filesystem | binary                                                    |
| character_set_results    | utf8                                                      |
| character_set_server     | utf8                                                      |
| character_set_system     | utf8                                                      |
| character_sets_dir       | C:\Program Files\MySQL\MySQL Server 5.7\share\charsets\  |
+--------------------------+-----------------------------------------------------------+
8 rows in set, 1 warning (0.01 sec)
```

四、实验内容

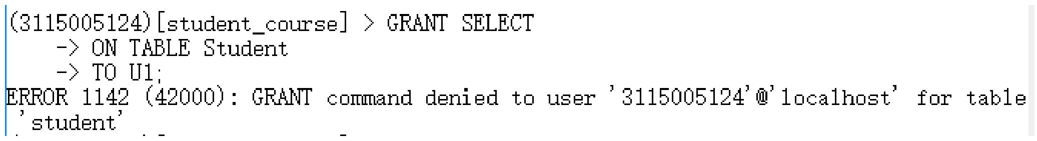
第四章 数据库安全性

授权：授予与收回

首先建立u1-u7的用户：

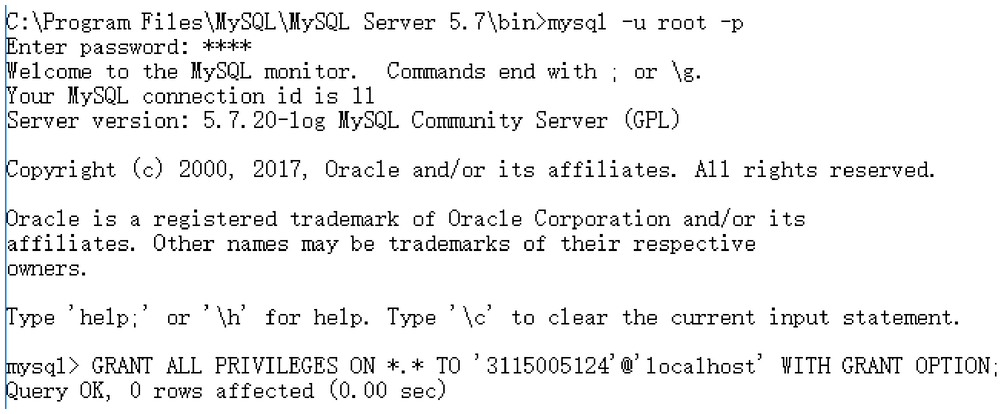


```
(3115005124)[(none)] > use student_course;
Database changed
(3115005124)[student_course] > create user 'U1'@'localhost' identified by '1234';
Query OK, 0 rows affected (0.01 sec)

(3115005124)[student_course] > create user 'U2'@'localhost' identified by '1234';
Query OK, 0 rows affected (0.00 sec)

(3115005124)[student_course] > create user 'U3'@'localhost' identified by '1234';
Query OK, 0 rows affected (0.00 sec)

(3115005124)[student_course] > create user 'U4'@'localhost' identified by '1234';
Query OK, 0 rows affected (0.00 sec)

(3115005124)[student_course] > create user 'U5'@'localhost' identified by '1234';
Query OK, 0 rows affected (0.00 sec)

(3115005124)[student_course] > create user 'U6'@'localhost' identified by '1234';
Query OK, 0 rows affected (0.02 sec)

(3115005124)[student_course] > create user 'U7'@'localhost' identified by '1234';
Query OK, 0 rows affected (0.00 sec)
```



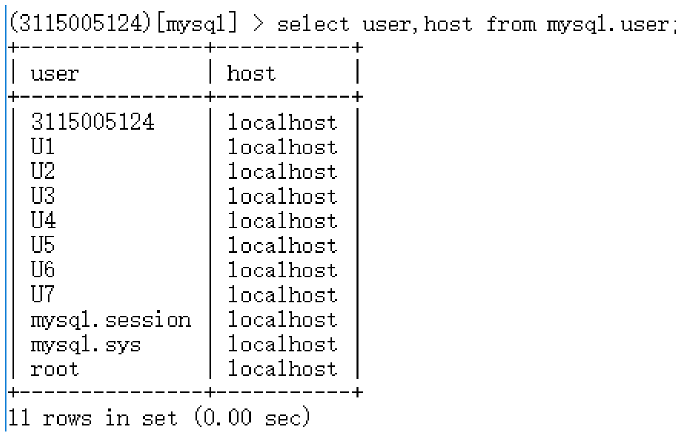
```
(3115005124)[student_course] > GRANT SELECT
    -> ON TABLE Student
    -> TO U1;
ERROR 1142 (42000): GRANT command denied to user '3115005124'@'localhost' for table
 'student'
```

错误原因：3115005124没有GRANT权限，解决方法：使用root用户赋予3115005124@localhost用户GRANT权限。



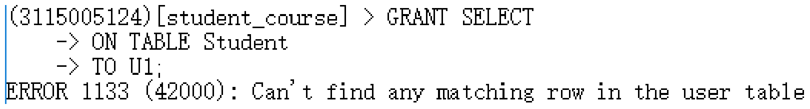
```
C:\Program Files\MySQL\MySQL Server 5.7\bin>mysql -u root -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 5.7.20-log MySQL Community Server (GPL)

Copyright (c) 2000, 2017, 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> GRANT ALL PRIVILEGES ON *.* TO '3115005124'@'localhost' WITH GRANT OPTION;
Query OK, 0 rows affected (0.00 sec)
```

查询用户列表：



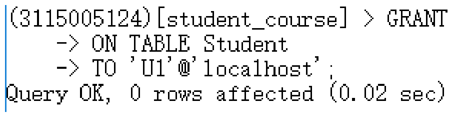
```
(3115005124)[mysql] > select user,host from mysql.user;
+---------------+-----------+
| user          | host      |
+---------------+-----------+
| 3115005124    | localhost |
| U1            | localhost |
| U2            | localhost |
| U3            | localhost |
| U4            | localhost |
| U5            | localhost |
| U6            | localhost |
| U7            | localhost |
| mysql.session | localhost |
| mysql.sys     | localhost |
| root          | localhost |
+---------------+-----------+
11 rows in set (0.00 sec)
```

【4.1】把查询Student表的权限授给用户U1。



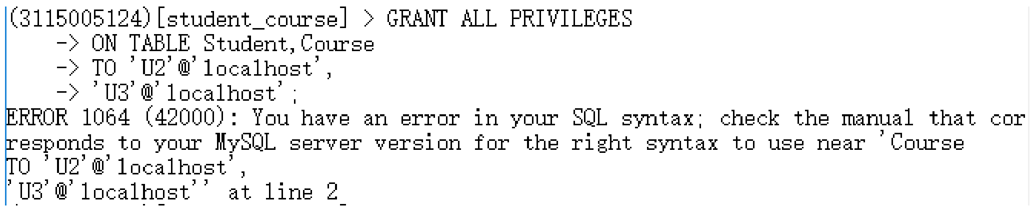
```
(3115005124)[student_course] > GRANT SELECT
    -> ON TABLE Student
    -> TO U1;
ERROR 1133 (42000): Can't find any matching row in the user table
```

错误原因：不能在用户表中找到u1这个用户。在Mysql中用户名必须为创建时的全名，解决方法如下：



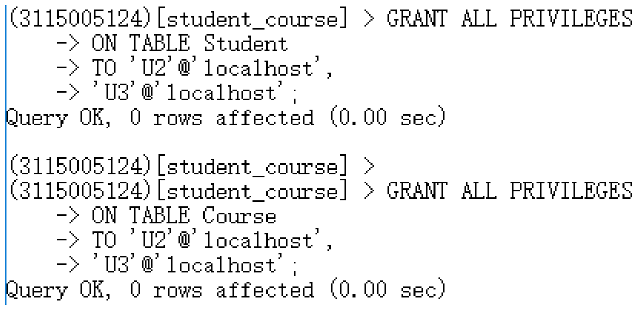
```
(3115005124)[student_course] > GRANT SELECT
    -> ON TABLE Student
    -> TO 'U1'@'localhost';
Query OK, 0 rows affected (0.02 sec)
```

【4.2】把对Student表和Course表的全部操作权限授予用户U2和U3。



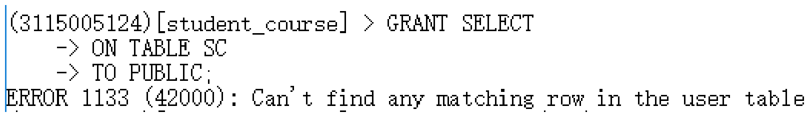
```
(3115005124)[student_course] > GRANT ALL PRIVILEGES
    -> ON TABLE Student,Course
    -> TO 'U2'@'localhost',
    -> 'U3'@'localhost';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that cor
responds to your MySQL server version for the right syntax to use near 'Course
TO 'U2'@'localhost',
'U3'@'localhost'' at line 2
```

错误原因：Mysql不支持同时将两张表的权限授予用户，解决方法如下：



```
(3115005124)[student_course] > GRANT ALL PRIVILEGES
    -> ON TABLE Student
    -> TO 'U2'@'localhost',
    -> 'U3'@'localhost';
Query OK, 0 rows affected (0.00 sec)

(3115005124)[student_course] >
(3115005124)[student_course] > GRANT ALL PRIVILEGES
    -> ON TABLE Course
    -> TO 'U2'@'localhost',
    -> 'U3'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

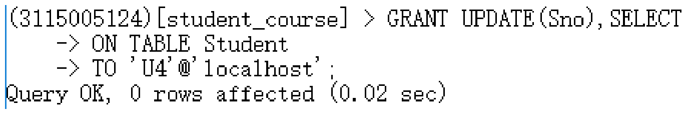
【4.3】把对表SC的查询权限授予所有用户。



```
(3115005124)[student_course] > GRANT SELECT
    -> ON TABLE SC
    -> TO PUBLIC;
ERROR 1133 (42000): Can't find any matching row in the user table
```

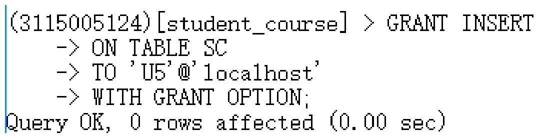
错误原因：PUBLIC 不是用户表里面的用户，也就是说Mysql没有用户组的概念，不能用PUBLIC来代替所有的用户，解决方法暂为一个个授权。

【4.4】把查询Student表和修改学生学号的权限授给用户U4。



```
(3115005124)[student_course] > GRANT UPDATE(Sno),SELECT
    -> ON TABLE Student
    -> TO 'U4'@'localhost';
Query OK, 0 rows affected (0.02 sec)
```

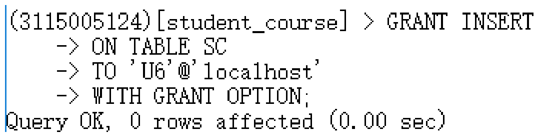
【4.5】把对表SC的INSERT权限授予U5用户，并允许将此权限再授予其他用户。



```
(3115005124)[student_course] > GRANT INSERT
    -> ON TABLE SC
    -> TO 'U5'@'localhost'
    -> WITH GRANT OPTION;
Query OK, 0 rows affected (0.00 sec)
```

【4.6】

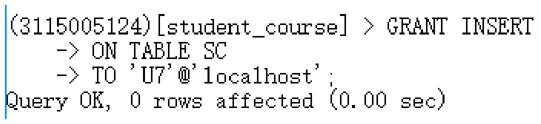
```
GRANT INSERT
ON TABLE SC
TO U6
WITH GRANT OPTION;
```



```
(3115005124)[student_course] > GRANT INSERT
    -> ON TABLE SC
    -> TO 'U6'@'localhost'
    -> WITH GRANT OPTION;
Query OK, 0 rows affected (0.00 sec)
```

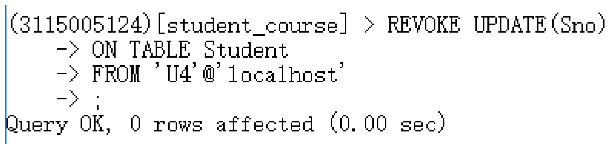
【4.7】

```
GRANT INSERT
ON TABLE SC
TO U7;
```



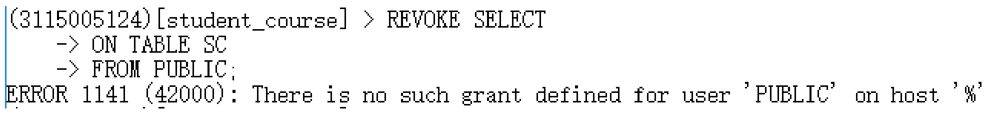
```
(3115005124)[student_course] > GRANT INSERT
    -> ON TABLE SC
    -> TO 'U7'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

【4.8】把用户U4修改学生学号的权限收回。



```
(3115005124)[student_course] > REVOKE UPDATE(Sno)
    -> ON TABLE Student
    -> FROM 'U4'@'localhost'
    -> ;
Query OK, 0 rows affected (0.00 sec)
```

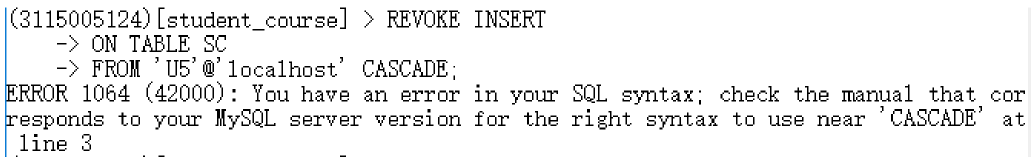
【4.9】收回所有用户对表SC的查询权限。



```
(3115005124)[student_course] > REVOKE SELECT
    -> ON TABLE SC
    -> FROM PUBLIC;
ERROR 1141 (42000): There is no such grant defined for user 'PUBLIC' on host '%'
```

错误原因：PUBLIC 不是用户表里面的用户，也就是说Mysql没有用户组的概念。

【4.10】把用户U5对SC表的INSERT权限收回。

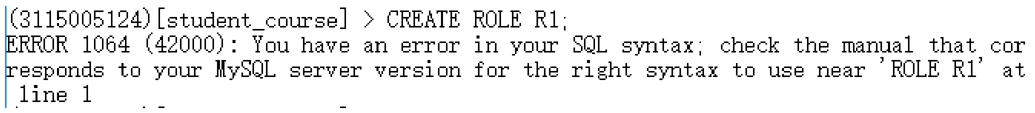


```
(3115005124)[student_course] > REVOKE INSERT
    -> ON TABLE SC
    -> FROM 'U5'@'localhost' CASCADE;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that cor
responds to your MySQL server version for the right syntax to use near 'CASCADE' at
 line 3
```

错误原因：MySQL不支持CASCADE，所以这里报错。

数据库角色

【4.11】通过角色来实现将一组权限授予一个用户。



```
(3115005124)[student_course] > CREATE ROLE R1;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that cor
responds to your MySQL server version for the right syntax to use near 'ROLE R1' at
 line 1
```

错误原因：Mysql不支持角色的创建，所以这里与角色有关的操作都无法进行。

【4.12】角色的权限修改。

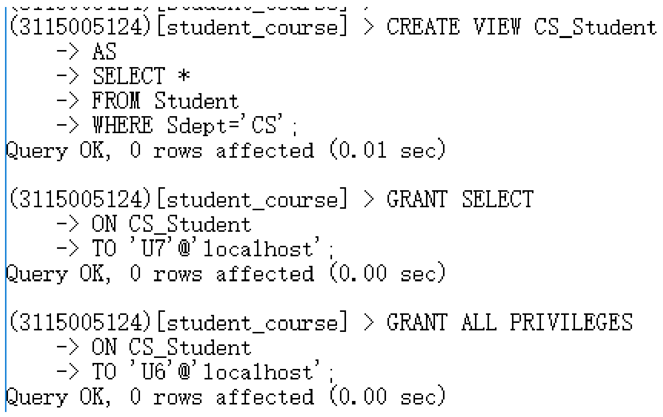
```
GRANT DELETE
ON TABLE Student
TO R1;
```

【4.13】

```
REVOKE SELECT
ON TABLE Student
FROM R1;
```

视图机制

【4.14】建立计算机系学生的视图，把对该视图的SELECT权限授予王平，把该视图上单所有操作权限授予张明。

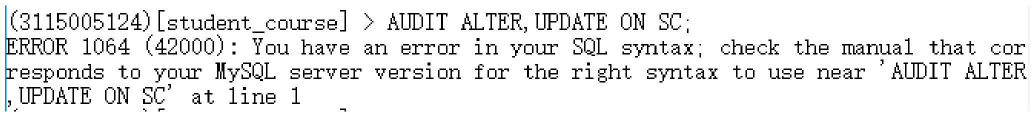


```
(3115005124)[student_course] > CREATE VIEW CS_Student
    -> AS
    -> SELECT *
    -> FROM Student
    -> WHERE Sdept='CS';
Query OK, 0 rows affected (0.01 sec)

(3115005124)[student_course] > GRANT SELECT
    -> ON CS_Student
    -> TO 'U7'@'localhost';
Query OK, 0 rows affected (0.00 sec)

(3115005124)[student_course] > GRANT ALL PRIVILEGES
    -> ON CS_Student
    -> TO 'U6'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

建立计算机系学生的视图，把该视图的select权限授予u7，再将权限授予u6。

审计

【4.15】对修改SC表结构或修改SC表数据的操作进行审计。



```
(3115005124)[student_course] > AUDIT ALTER,UPDATE ON SC;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that cor
responds to your MySQL server version for the right syntax to use near 'AUDIT ALTER
,UPDATE ON SC' at line 1
```

错误原因：MySQL服务器自身没有提供审计功能。

【4.16】取消对SC表的一切审计。

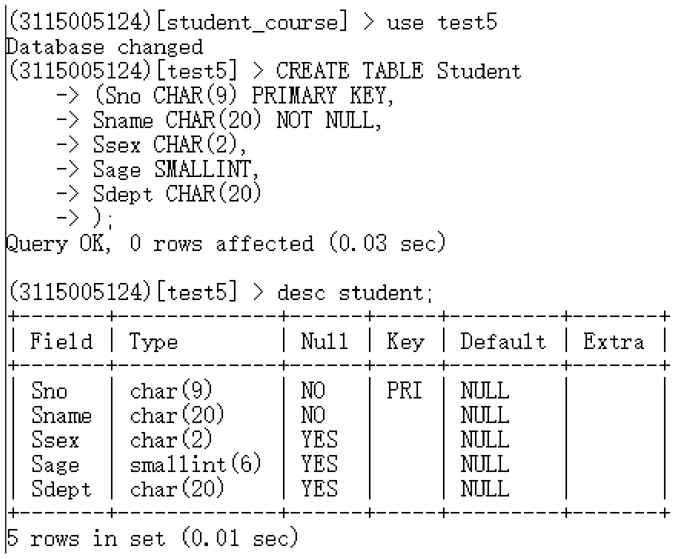
```
NOAUDIT ALTER,UPDATE
ON SC;
```

第五章 数据库完整性

定义实体完整性

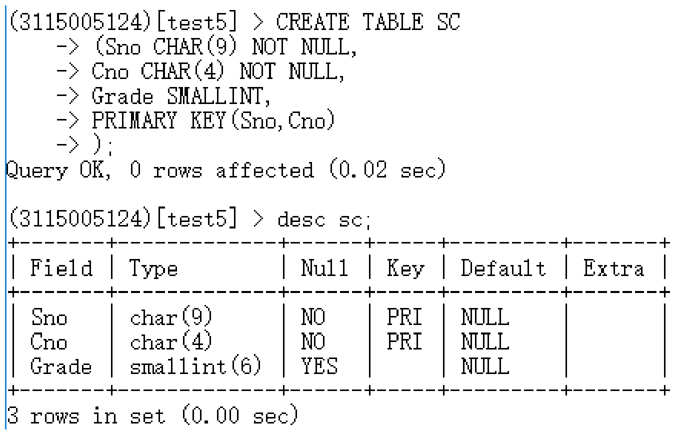
由于本章实验设计大量删表建表操作，为了不对之前student——course数据库造成不可恢复的修改，此章定义新的数据库test5。

【5.1】将Student表中的Sno属性定义为码。



```
(3115005124)[student_course] > use test5
Database changed
(3115005124)[test5] > CREATE TABLE Student
    -> (Sno CHAR(9) PRIMARY KEY,
    -> Sname CHAR(20) NOT NULL,
    -> Ssex CHAR(2),
    -> Sage SMALLINT,
    -> Sdept CHAR(20)
    -> );
Query OK, 0 rows affected (0.03 sec)

(3115005124)[test5] > desc student;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| Sno   | char(9)     | NO   | PRI | NULL    |       |
| Sname | char(20)    | NO   |     | NULL    |       |
| Ssex  | char(2)     | YES  |     | NULL    |       |
| Sage  | smallint(6) | YES  |     | NULL    |       |
| Sdept | char(20)    | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

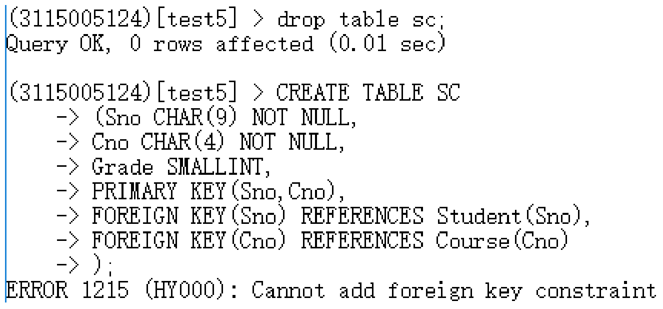
【5.2】将SC表中的Sno、Cno属性组定义为码。



```
(3115005124)[test5] > CREATE TABLE SC
    -> (Sno CHAR(9) NOT NULL,
    -> Cno CHAR(4) NOT NULL,
    -> Grade SMALLINT,
    -> PRIMARY KEY(Sno,Cno)
    -> );
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > desc sc;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| Sno   | char(9)     | NO   | PRI | NULL    |       |
| Cno   | char(4)     | NO   | PRI | NULL    |       |
| Grade | smallint(6) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

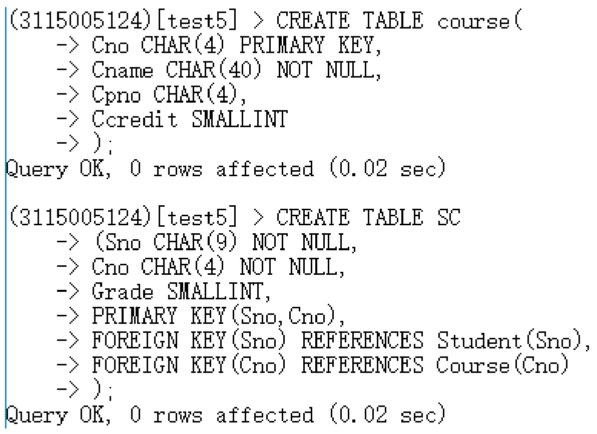
定义参照完整性

【5.3】定义SC表中的参照完整性。



```
(3115005124)[test5] > drop table sc;
Query OK, 0 rows affected (0.01 sec)

(3115005124)[test5] > CREATE TABLE SC
    -> (Sno CHAR(9) NOT NULL,
    -> Cno CHAR(4) NOT NULL,
    -> Grade SMALLINT,
    -> PRIMARY KEY(Sno,Cno),
    -> FOREIGN KEY(Sno) REFERENCES Student(Sno),
    -> FOREIGN KEY(Cno) REFERENCES Course(Cno)
    -> );
ERROR 1215 (HY000): Cannot add foreign key constraint
```

错误原因：test5数据库中没有Course表。解决方法如下：先新建Course表，再完成有关操作。



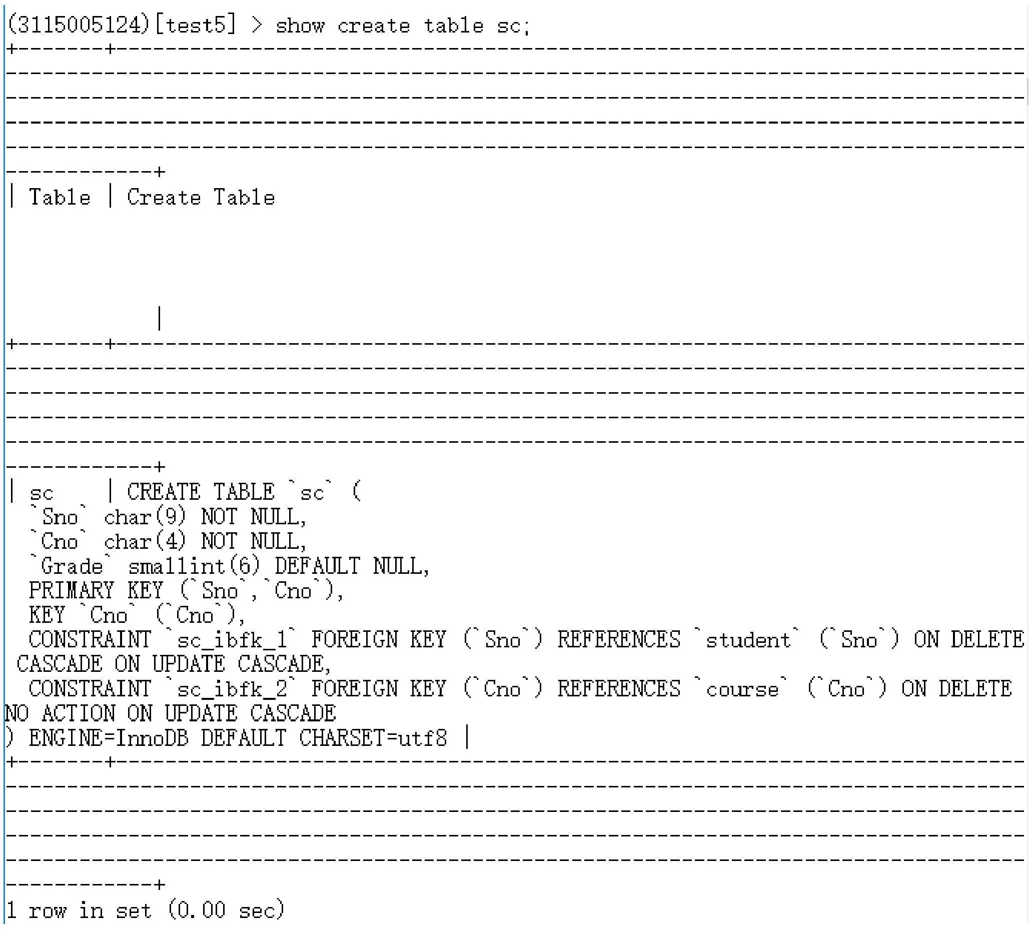
```
(3115005124)[test5] > CREATE TABLE course(
    -> Cno CHAR(4) PRIMARY KEY,
    -> Cname CHAR(40) NOT NULL,
    -> Cpno CHAR(4),
    -> Ccredit SMALLINT
    -> );
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > CREATE TABLE SC
    -> (Sno CHAR(9) NOT NULL,
    -> Cno CHAR(4) NOT NULL,
    -> Grade SMALLINT,
    -> PRIMARY KEY(Sno,Cno),
    -> FOREIGN KEY(Sno) REFERENCES Student(Sno),
    -> FOREIGN KEY(Cno) REFERENCES Course(Cno)
    -> );
Query OK, 0 rows affected (0.02 sec)
```

```
(3115005124)[test5] > show create table sc;
+--------+-----------------------------------------------------------------
--------------------------------------------------------------------------
--------------------------------------------------------------------------
--------------------------------------------------------------------------
---------------------+
| Table | Create Table




                     |
+--------+-----------------------------------------------------------------
--------------------------------------------------------------------------
--------------------------------------------------------------------------
--------------------------------------------------------------------------
---------------------+
| sc    | CREATE TABLE `sc` (
  `Sno` char(9) NOT NULL,
  `Cno` char(4) NOT NULL,
  `Grade` smallint(6) DEFAULT NULL,
  PRIMARY KEY (`Sno`,`Cno`),
  KEY `Cno` (`Cno`),
  CONSTRAINT `sc_ibfk_1` FOREIGN KEY (`Sno`) REFERENCES `student` (`Sno`),
  CONSTRAINT `sc_ibfk_2` FOREIGN KEY (`Cno`) REFERENCES `course` (`Cno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+--------+-----------------------------------------------------------------
--------------------------------------------------------------------------
--------------------------------------------------------------------------
--------------------------------------------------------------------------
---------------------+
1 row in set (0.00 sec)
```

参照完整性检查和违约处理

【5.4】显式说明参照完整性的违约处理示例。

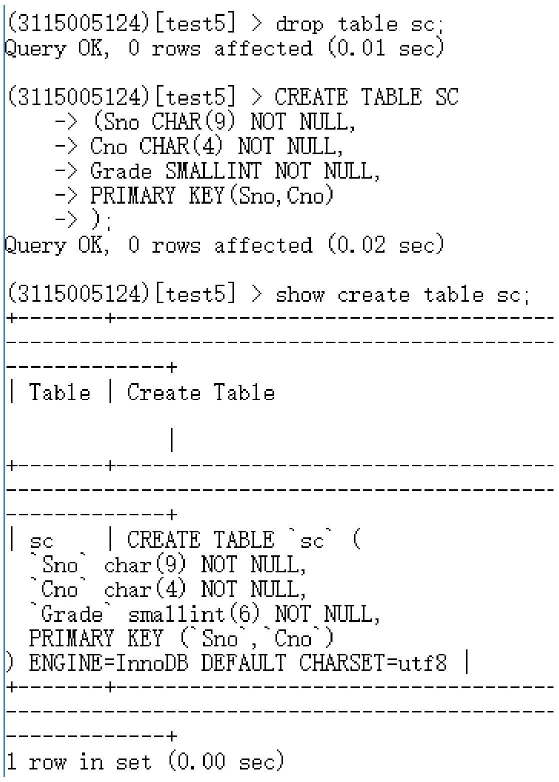
```
(3115005124)[test5] > drop table sc;
Query OK, 0 rows affected (0.01 sec)

(3115005124)[test5] > CREATE TABLE SC
    -> (Sno CHAR(9) NOT NULL,
    -> Cno CHAR(4) NOT NULL,
    -> Grade SMALLINT,
    -> PRIMARY KEY(Sno,Cno),
    -> FOREIGN KEY(Sno) REFERENCES Student(Sno)
    -> ON DELETE CASCADE
    ->
    -> ON UPDATE CASCADE,
    -> FOREIGN KEY(Cno) REFERENCES Course(Cno)
    -> ON DELETE NO ACTION
    ->
    -> ON UPDATE CASCADE
    -> );
Query OK, 0 rows affected (0.02 sec)
```



```
(3115005124)[test5] > show create table sc;
+--------+-------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-------------+
| Table | Create Table





             |
+--------+-------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-------------+
| sc    | CREATE TABLE `sc` (
  `Sno` char(9) NOT NULL,
  `Cno` char(4) NOT NULL,
  `Grade` smallint(6) DEFAULT NULL,
  PRIMARY KEY (`Sno`,`Cno`),
  KEY `Cno` (`Cno`),
  CONSTRAINT `sc_ibfk_1` FOREIGN KEY (`Sno`) REFERENCES `student` (`Sno`) ON DELETE
 CASCADE ON UPDATE CASCADE,
  CONSTRAINT `sc_ibfk_2` FOREIGN KEY (`Cno`) REFERENCES `course` (`Cno`) ON DELETE
NO ACTION ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+--------+-------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-----------------------------------------------------------------------------------
-------------+
1 row in set (0.00 sec)
```

属性上的约束条件

【5.5】在定义SC表时，说明Sno、Cno、Grade属性不允许取空值。



```
(3115005124)[test5] > drop table sc;
Query OK, 0 rows affected (0.01 sec)

(3115005124)[test5] > CREATE TABLE SC
    -> (Sno CHAR(9) NOT NULL,
    -> Cno CHAR(4) NOT NULL,
    -> Grade SMALLINT NOT NULL,
    -> PRIMARY KEY(Sno,Cno)
    -> );
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > show create table sc;
+--------+------------------------------------
---------------------------------------------
--------------+
| Table | Create Table

              |
+--------+------------------------------------
---------------------------------------------
--------------+
| sc    | CREATE TABLE `sc` (
  `Sno` char(9) NOT NULL,
  `Cno` char(4) NOT NULL,
  `Grade` smallint(6) NOT NULL,
  PRIMARY KEY (`Sno`,`Cno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+--------+------------------------------------
---------------------------------------------
--------------+
1 row in set (0.00 sec)
```

【5.6】建立部门表DEPT，要求部门名称Dname列取值唯一，部门编号Deptno列为主码。

```
(3115005124)[test5] > CREATE TABLE DEPT
    -> (Deptno NUMERIC(2),
    -> Dname CHAR(9) UNIQUE NOT NULL,
    -> Location CHAR(10),
    -> PRIMARY KEY(Deptno)
    -> );
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > show create table dept;
+--------+------------------------------------
----------------------------------------------
----------------------------------------------
| Table  | Create Table


+--------+------------------------------------
----------------------------------------------
----------------------------------------------
| dept   | CREATE TABLE `dept` (
  `Deptno` decimal(2,0) NOT NULL,
  `Dname` char(9) NOT NULL,
  `Location` char(10) DEFAULT NULL,
  PRIMARY KEY (`Deptno`),
  UNIQUE KEY `Dname` (`Dname`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+--------+------------------------------------
----------------------------------------------
----------------------------------------------
1 row in set (0.00 sec)
```

【5.7】Student表的Ssex只允许取"男"或"女"。

```
(3115005124)[test5] > drop table student;
Query OK, 0 rows affected (0.01 sec)

(3115005124)[test5] > CREATE TABLE Student
    -> (Sno CHAR(9) PRIMARY KEY,
    -> Sname CHAR(8) NOT NULL,
    -> Ssex CHAR(2) CHECK (Ssex IN('男','女')),
    ->
    -> Sage SMALLINT,
    -> Sdept CHAR(20)
    -> );
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > show create table student;
+----------+----------------------------------
----------------------------------------------
----------------------------------------------
+
| Table    | Create Table


|
+----------+----------------------------------
----------------------------------------------
----------------------------------------------
+
| student | CREATE TABLE `student` (
  `Sno` char(9) NOT NULL,
  `Sname` char(8) NOT NULL,
  `Ssex` char(2) DEFAULT NULL,
  `Sage` smallint(6) DEFAULT NULL,
  `Sdept` char(20) DEFAULT NULL,
  PRIMARY KEY (`Sno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+----------+----------------------------------
----------------------------------------------
----------------------------------------------
+
1 row in set (0.00 sec)
```

【5.8】SC表的Grade的值应该在0和100之间。

```
(3115005124)[test5] > drop table sc;
Query OK, 0 rows affected (0.01 sec)

(3115005124)[test5] > CREATE TABLE SC
    -> (Sno CHAR(9),
    -> Cno CHAR(4),
    -> Grade SMALLINT CHECK (Grade>=0 AND Grade<=100),
    -> PRIMARY KEY(Sno,Cno),
    -> FOREIGN KEY(Sno) REFERENCES Student(Sno),
    -> FOREIGN KEY(Cno) REFERENCES Course(Cno)
    -> );
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > show create table sc;
+--------+-----------------------------------------------------------------
---------------------------------------------------------------------------
---------------------------------------------------------------------------
---------------------------------------------------------------------------
----------------------+
| Table  | Create Table



                     |
+--------+-----------------------------------------------------------------
---------------------------------------------------------------------------
---------------------------------------------------------------------------
---------------------------------------------------------------------------
----------------------+
| sc     | CREATE TABLE `sc` (
  `Sno` char(9) NOT NULL,
  `Cno` char(4) NOT NULL,
  `Grade` smallint(6) DEFAULT NULL,
  PRIMARY KEY (`Sno`,`Cno`),
  KEY `Cno` (`Cno`),
  CONSTRAINT `sc_ibfk_1` FOREIGN KEY (`Sno`) REFERENCES `student` (`Sno`),
  CONSTRAINT `sc_ibfk_2` FOREIGN KEY (`Cno`) REFERENCES `course` (`Cno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+--------+-----------------------------------------------------------------
```

元组上约束条件的定义

【5.9】当学生的性别是男时，其名字不能以Mr.s打头。

```
(3115005124)[test5] > drop table student;
ERROR 1217 (23000): Cannot delete or update a parent row: a foreign key constraint
fails
```

错误原因：在之前建立SC表的时候，SC表的外键Sno与student表的主键约束，所以此时无法删除student表，此时可以先删除SC表，再删除student表。

```
(3115005124)[test5] > drop table sc;
Query OK, 0 rows affected (0.01 sec)

(3115005124)[test5] > drop table student;
Query OK, 0 rows affected (0.01 sec)

(3115005124)[test5] > CREATE TABLE Student
    -> (Sno CHAR(9),
    -> Sname CHAR(8) NOT NULL,
    -> Ssex CHAR(2),
    -> Sage SMALLINT,
    -> Sdept CHAR(20),
    -> PRIMARY KEY(Sno),
    -> CHECK(Ssex='女' OR Sname NOT LIKE 'Mr.%')
    -> );
Query OK, 0 rows affected (0.03 sec)

(3115005124)[test5] > show create table student;
+---------+--------------------------------------
--------------------------------------------------
--------------------------------------------------
+
| Table   | Create Table


|
+---------+--------------------------------------
--------------------------------------------------
--------------------------------------------------
+
| student | CREATE TABLE `student` (
  `Sno` char(9) NOT NULL,
  `Sname` char(8) NOT NULL,
  `Ssex` char(2) DEFAULT NULL,
  `Sage` smallint(6) DEFAULT NULL,
  `Sdept` char(20) DEFAULT NULL,
  PRIMARY KEY (`Sno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+---------+--------------------------------------
```

重新建立SC表：

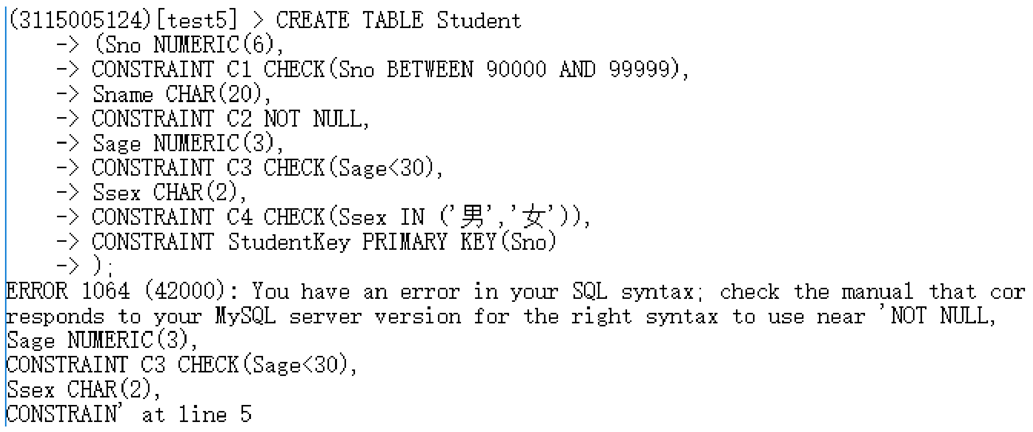
```
(3115005124)[test5] > CREATE TABLE SC
    -> (Sno CHAR(9),
    -> Cno CHAR(4),
    -> Grade SMALLINT CHECK (Grade>=0 AND Grade<=100),
    -> PRIMARY KEY(Sno,Cno),
    -> FOREIGN KEY(Sno) REFERENCES Student(Sno),
    -> FOREIGN KEY(Cno) REFERENCES Course(Cno)
    -> );
Query OK, 0 rows affected (0.02 sec)
```

5.4完整性约束命名子句

【5.10】建立学生登记表Student，要求学号在90000~99999之间，姓名不能取空值，年龄小于30，性别只能是“男”或“女”。

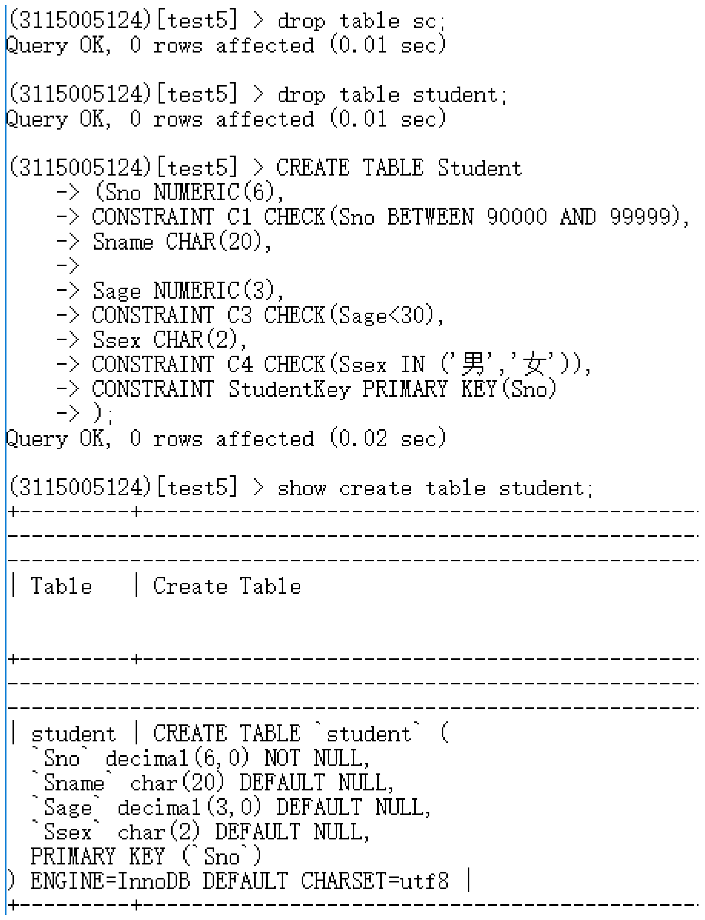
```
(3115005124)[test5] > CREATE TABLE Student
    -> (Sno NUMERIC(6)
    -> CONSTRAINT C1 CHECK(Sno BETWEEN 90000 AND 99999),
    -> Sname CHAR(20)
    -> CONSTRAINT C2 NOT NULL,
    -> Sage NUMERIC(3)
    -> CONSTRAINT C3 CHECK(Sage<30),
    -> Ssex CHAR(2)
    -> CONSTRAINT C4 CHECK(Ssex IN ('男','女')),
    -> CONSTRAINT StudentKey PRIMARY KEY(Sno)
    -> );
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that cor
responds to your MySQL server version for the right syntax to use near 'CONSTRAINT
C1 CHECK(Sno BETWEEN 90000 AND 99999),
Sname CHAR(20)
CONSTRAINT C2 N' at line 3
```

错误原因：少了逗号将CONSTRAINT与属性隔开。
根据错误原因将 SQL语句 进行简单修改，执行结果如下：



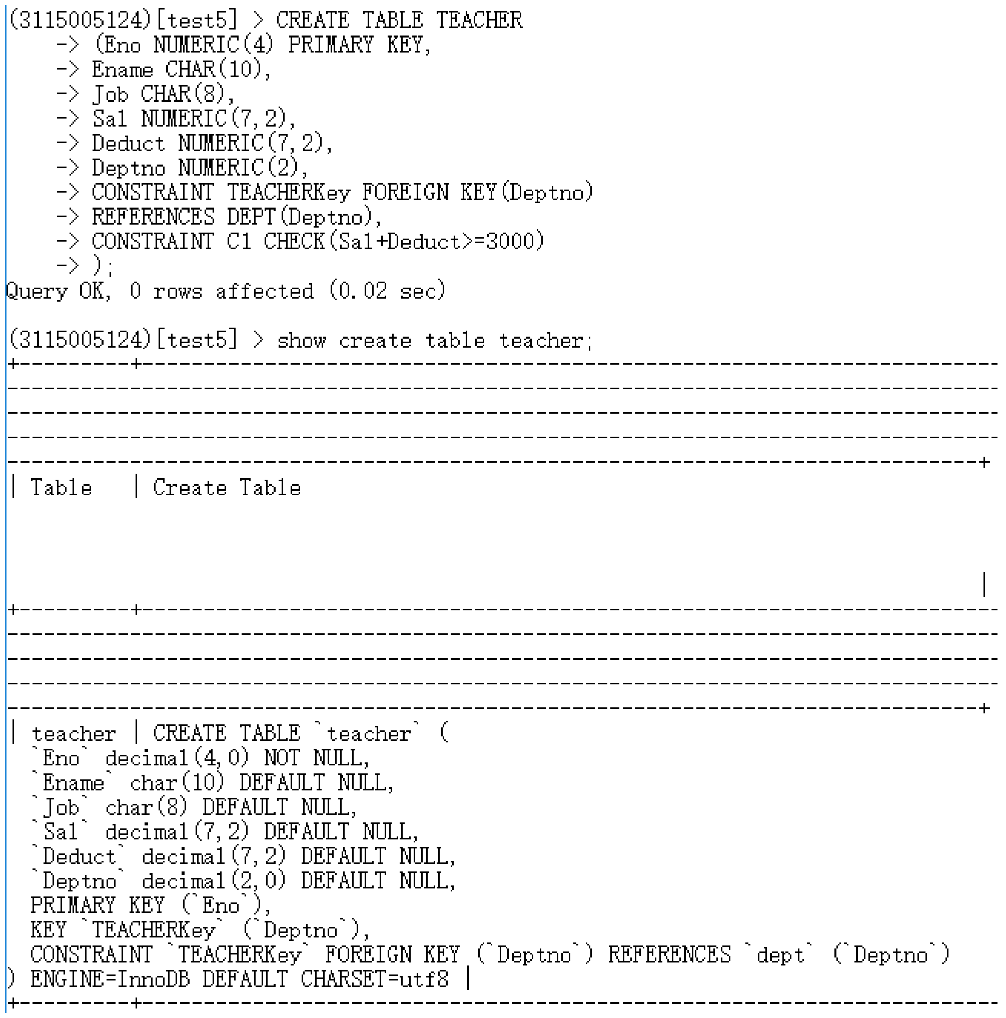
```
(3115005124)[test5] > CREATE TABLE Student
    -> (Sno NUMERIC(6),
    -> CONSTRAINT C1 CHECK(Sno BETWEEN 90000 AND 99999),
    -> Sname CHAR(20),
    -> CONSTRAINT C2 NOT NULL,
    -> Sage NUMERIC(3),
    -> CONSTRAINT C3 CHECK(Sage<30),
    -> Ssex CHAR(2),
    -> CONSTRAINT C4 CHECK(Ssex IN ('男','女')),
    -> CONSTRAINT StudentKey PRIMARY KEY(Sno)
    -> );
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that cor
responds to your MySQL server version for the right syntax to use near 'NOT NULL,
Sage NUMERIC(3),
CONSTRAINT C3 CHECK(Sage<30),
Ssex CHAR(2),
CONSTRAIN' at line 5
```

错误原因：Sname char(20),constraint C2 not null 这句报错，说明CONSTRAINT的列级约束键后不能为空，否则将产生错误。
最终修改后的 SQL语句 如下：



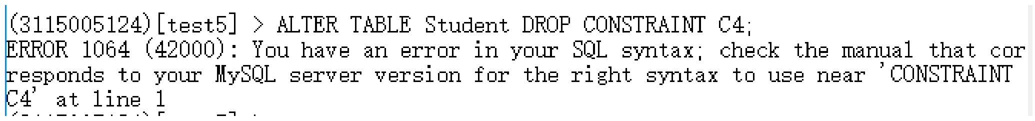
```
(3115005124)[test5] > drop table sc;
Query OK, 0 rows affected (0.01 sec)

(3115005124)[test5] > drop table student;
Query OK, 0 rows affected (0.01 sec)

(3115005124)[test5] > CREATE TABLE Student
    -> (Sno NUMERIC(6),
    -> CONSTRAINT C1 CHECK(Sno BETWEEN 90000 AND 99999),
    -> Sname CHAR(20),
    ->
    -> Sage NUMERIC(3),
    -> CONSTRAINT C3 CHECK(Sage<30),
    -> Ssex CHAR(2),
    -> CONSTRAINT C4 CHECK(Ssex IN ('男','女')),
    -> CONSTRAINT StudentKey PRIMARY KEY(Sno)
    -> );
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > show create table student;
+---------+---------------------------------------------
--------------------------------------------------------
--------------------------------------------------------
| Table   | Create Table


+---------+---------------------------------------------
--------------------------------------------------------
--------------------------------------------------------
| student | CREATE TABLE `student` (
  `Sno` decimal(6,0) NOT NULL,
  `Sname` char(20) DEFAULT NULL,
  `Sage` decimal(3,0) DEFAULT NULL,
  `Ssex` char(2) DEFAULT NULL,
  PRIMARY KEY (`Sno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+---------+---------------------------------------------
```

【5.11】建立教室表TEACHER，要求每个教师的应发工资不低于3000元。应发工资是工资列Sal与扣除项Deduct之和。



```
(3115005124)[test5] > CREATE TABLE TEACHER
    -> (Eno NUMERIC(4) PRIMARY KEY,
    -> Ename CHAR(10),
    -> Job CHAR(8),
    -> Sal NUMERIC(7,2),
    -> Deduct NUMERIC(7,2),
    -> Deptno NUMERIC(2),
    -> CONSTRAINT TEACHERKey FOREIGN KEY(Deptno)
    -> REFERENCES DEPT(Deptno),
    -> CONSTRAINT C1 CHECK(Sal+Deduct>=3000)
    -> );
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > show create table teacher;
+---------+--------------------------------------------------------------------
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
--------------------------------------------------------------------------------+
| Table   | Create Table




                                                                                |
+---------+--------------------------------------------------------------------
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
--------------------------------------------------------------------------------+
| teacher | CREATE TABLE `teacher` (
  `Eno` decimal(4,0) NOT NULL,
  `Ename` char(10) DEFAULT NULL,
  `Job` char(8) DEFAULT NULL,
  `Sal` decimal(7,2) DEFAULT NULL,
  `Deduct` decimal(7,2) DEFAULT NULL,
  `Deptno` decimal(2,0) DEFAULT NULL,
  PRIMARY KEY (`Eno`),
  KEY `TEACHERKey` (`Deptno`),
  CONSTRAINT `TEACHERKey` FOREIGN KEY (`Deptno`) REFERENCES `dept` (`Deptno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+---------+--------------------------------------------------------------------
```

【5.12】去掉例5.10 Student表中对性别的限制。



```
(3115005124)[test5] > ALTER TABLE Student DROP CONSTRAINT C4;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that cor
responds to your MySQL server version for the right syntax to use near 'CONSTRAINT
C4' at line 1
```

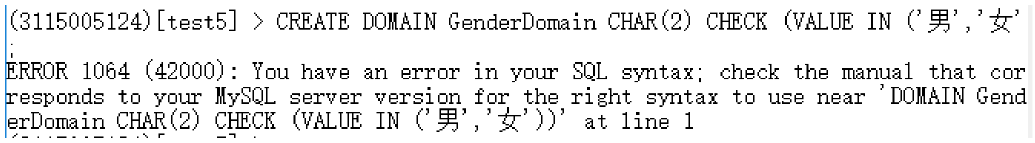
错误原因：在MySQL中，没有删除CONSTRAINT的语句，所以在读到CONSTRAINT时就会报错 域中完整性约束。

【5.13】修改表Student中的约束条件，要求学号改为在900000~999999之间，年龄由小于30改为小于40。

```
ALTER TABLE Student DROP CONSTRAINT C1;
ALTER TABLE Student ADD CONSTRAINT C1 CHECK(Sno BETWEEN 900000 AND 999999);
ALTER TABLE Student DROP CONSTRAINT C3;
ALTER TABLE Student ADD CONSTRAINT C3 CHECK(Sage<40);
```

域中的完整性限制

【5.14】建立一个性别域，并声明性别域的取值范围。



```
(3115005124)[test5] > CREATE DOMAIN GenderDomain CHAR(2) CHECK (VALUE IN ('男','女')
;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that cor
responds to your MySQL server version for the right syntax to use near 'DOMAIN Gend
erDomain CHAR(2) CHECK (VALUE IN ('男','女'))' at line 1
```

错误原因：MySQL不支持DOMAIN语句。

【5.15】建立一个性别域 GenderDomain，并对其中的限制命名。

```
CREATE DOMAIN GenderDomain CHAR(2)
CONSTRAINT GD CHECK(VALUE IN ('男','女'));
```

【5.16】删除域 GenderDomain的限制条件GD。

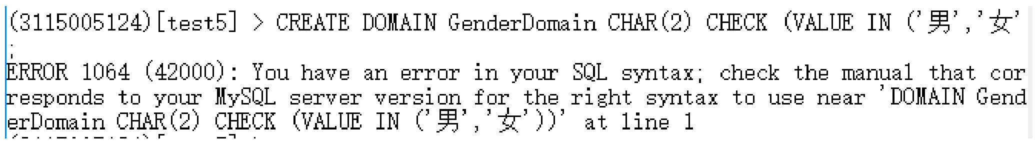
```
ALTER DOMAIN GenderDomain
DROP CONSTRAINT GD;
```

【5.17】在域 GenderDomain上增加性别的限制条件GDD。

```
ALTER DOMAIN GenderDomain
ADD CONSTRAINT GDD CHECK(VALUE IN ('1','0'));
```

断言

【5.18】限制数据库课程最多60名学生选修。



```
(3115005124)[test5] > CREATE DOMAIN GenderDomain CHAR(2) CHECK (VALUE IN ('男','女'
;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that cor
responds to your MySQL server version for the right syntax to use near 'DOMAIN Gend
erDomain CHAR(2) CHECK (VALUE IN ('男','女'))' at line 1
```

错误原因：MySQL不支持ASSERTION语句。

【5.19】限制每一门课程最多60名学生选修。

```
CREATE ASSERTION ASSE_SC_DB_CNUM1
    CHECK(60>=ALL(SELECT count(*)
                    FROM SC
                    GROUP by cno)
                );
```

【5.20】限制每个学期每一门课程最多60名学生选修。

```
ALTER TABLE SC ADD TERM DATE;

CREATE ASSERTION ASSE_SC_DB_CNUM2
    CHECK(60>=ALL (select count(*) from SC group by cno, TERM));
```

触发器

书上的例子无法在MySQL上顺利运行，且没有例程输入输出验证结果。此处阅读MySQL帮助文档REFMAN，运行文档中的TRIGGER例程。

【5.21】当对表SC的Grade属性进行修改时，若分数增加了10%，则将此次操作记录到另一个表SC_U(Sno、Cno、Oldgrade、Newgrade)中，其中 Oldgrade是修改前的分数，Newgrade是修改后的分数。

```
CREATE TRIGGER SC_T
AFTER UPDATE OF Grade ON SC
REFERENCING
    OLDROW AS OldTuple,
    NEWROW AS NewTuple
FOR EACH ROW
WHEN(NewTuple.Grade>=1.1*OLDROW.Grade)
    INSERT INTO SC_U(Sno,Cno,OldGrade,NewGrade)
    VALUE(OldTuple.Sno,OldTuple.Cno,OldTuple.Grade,NewTuple.Grade)
```

【5.22】将每次对表Student的插入操作所增加的学生个数记录到表Student-InsertLog中。

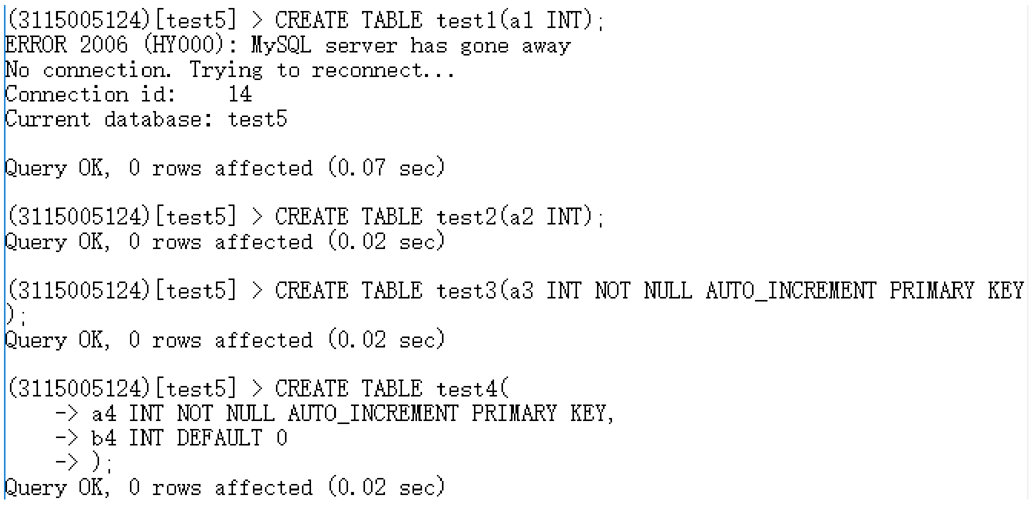
```
CREATE TRIGGER Student_Count
AFTER INSERT ON Student
REFERENCING
    NEW TABLE AS DELTA
    FOR EACH STATEMENT
```

```
        INSERT INTO StudentInsertLog(Numbers)
        SELECT COUNT(*) FROM DELTA
```

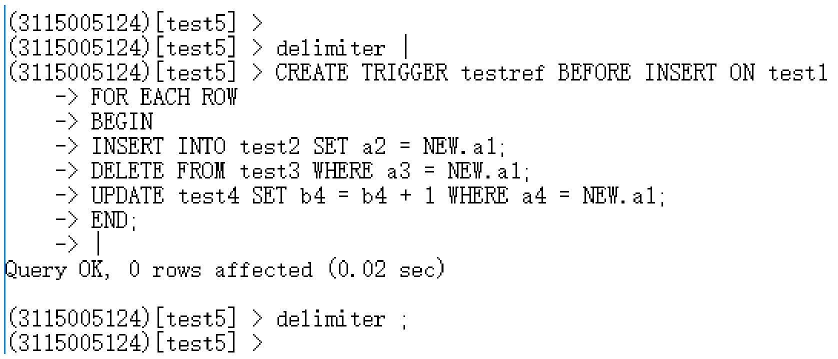
【5.23】定义一个BEFORE行级触发器，为教室表Teacher定义完整性规则“教授的工资不得低于4000元，如果低于4000元，自动改为4000元”。

```
CREATE TRIGGER Insert_Or_Update_Sal
BEFORE INSERT OR UPDATE ON Teacher
REFERENCING NEW row AS newTuple
FOR EACH ROW
BEGIN
    IF(newtuple.Job='教授')AND(newtuple.Sal<4000)

        THEN newtuple.Sal:=4000;
    END IF;
END;
```

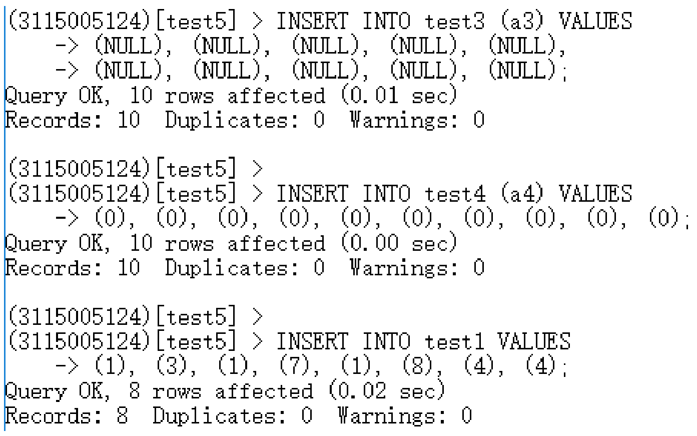
MySQL的触发器也是比较困难的一部分，书上的例子无法在MySQL上顺利运行，通过学习后才知道MySQL的触发器需要放在关键字DELIMITER里面，而且还有好多约束。此处通过学习并且运行MySQL帮助文档REFMAN文档 里的例程。



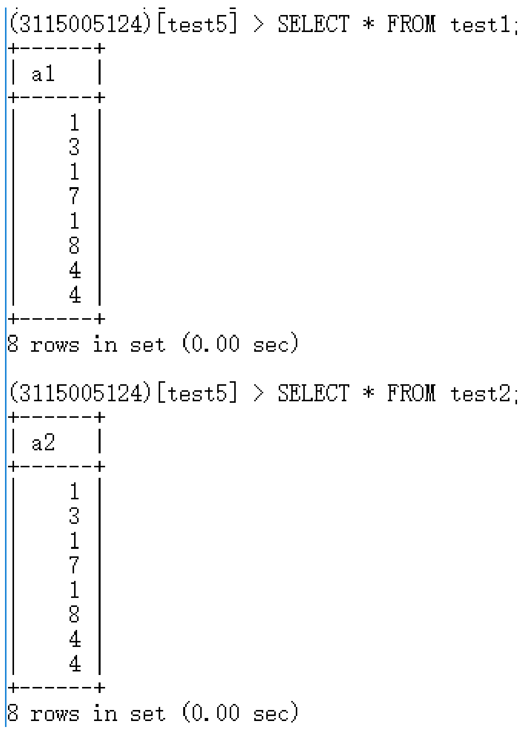
```
(3115005124)[test5] > CREATE TABLE test1(a1 INT);
ERROR 2006 (HY000): MySQL server has gone away
No connection. Trying to reconnect...
Connection id:    14
Current database: test5

Query OK, 0 rows affected (0.07 sec)

(3115005124)[test5] > CREATE TABLE test2(a2 INT);
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > CREATE TABLE test3(a3 INT NOT NULL AUTO_INCREMENT PRIMARY KEY
);
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > CREATE TABLE test4(
    -> a4 INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    -> b4 INT DEFAULT 0
    -> );
Query OK, 0 rows affected (0.02 sec)
```



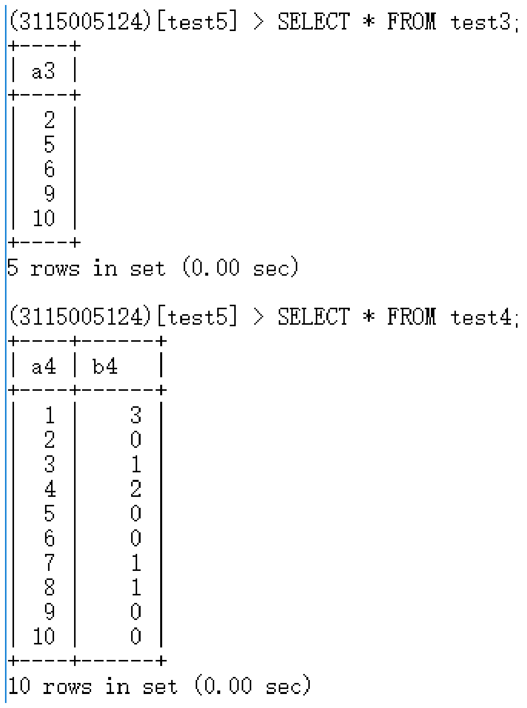
```
(3115005124)[test5] >
(3115005124)[test5] > delimiter |
(3115005124)[test5] > CREATE TRIGGER testref BEFORE INSERT ON test1
    -> FOR EACH ROW
    -> BEGIN
    -> INSERT INTO test2 SET a2 = NEW.a1;
    -> DELETE FROM test3 WHERE a3 = NEW.a1;
    -> UPDATE test4 SET b4 = b4 + 1 WHERE a4 = NEW.a1;
    -> END;
    -> |
Query OK, 0 rows affected (0.02 sec)

(3115005124)[test5] > delimiter ;
(3115005124)[test5] >
```



```
(3115005124)[test5] > INSERT INTO test3 (a3) VALUES
    -> (NULL), (NULL), (NULL), (NULL), (NULL),
    -> (NULL), (NULL), (NULL), (NULL), (NULL);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

(3115005124)[test5] >
(3115005124)[test5] > INSERT INTO test4 (a4) VALUES
    -> (0), (0), (0), (0), (0), (0), (0), (0), (0), (0);
Query OK, 10 rows affected (0.00 sec)
Records: 10  Duplicates: 0  Warnings: 0

(3115005124)[test5] >
(3115005124)[test5] > INSERT INTO test1 VALUES
    -> (1), (3), (1), (7), (1), (8), (4), (4);
Query OK, 8 rows affected (0.02 sec)
Records: 8  Duplicates: 0  Warnings: 0
```



```
(3115005124)[test5] > SELECT * FROM test1;
+------+
| a1   |
+------+
|    1 |
|    3 |
|    1 |
|    7 |
|    1 |
|    8 |
|    4 |
|    4 |
+------+
8 rows in set (0.00 sec)

(3115005124)[test5] > SELECT * FROM test2;
+------+
| a2   |
+------+
|    1 |
|    3 |
|    1 |
|    7 |
|    1 |
|    8 |
|    4 |
|    4 |
+------+
8 rows in set (0.00 sec)
```



```
(3115005124)[test5] > SELECT * FROM test3;
+----+
| a3 |
+----+
|  2 |
|  5 |
|  6 |
|  9 |
| 10 |
+----+
5 rows in set (0.00 sec)

(3115005124)[test5] > SELECT * FROM test4;
+----+------+
| a4 | b4   |
+----+------+
|  1 |    3 |
|  2 |    0 |
|  3 |    1 |
|  4 |    2 |
|  5 |    0 |
|  6 |    0 |
|  7 |    1 |
|  8 |    1 |
|  9 |    0 |
| 10 |    0 |
+----+------+
10 rows in set (0.00 sec)
```

五、实验总结

书上所说的SQL语句和实际操作的MySQL语句还是有一定差别的。走了很多的弯路。需要具有从MySQL帮助文档中解决问题的能力。

出BUG的例子：
【4.3】原因：Mysql没有用户组的概念
【4.9】原因：Mysql没有用户组的概念
【4.10】原因：MySQL不支持CASCADE
【4.11】原因：Mysql不支持角色的创建
【4.12】原因：Mysql不支持角色的创建 【4.13】原因：Mysql不支持角色的创建
【4.15】原因：MySQL服务器自身没有提供审计功能
【4.16】原因：MySQL服务器自身没有提供审计功能

【5.10】原因：CONSTRAINT的列级约束键后不能为空，否则将产生错误
【5.12】原因：在MySQL中，没有删除CONSTRAINT的语句
【5.13】原因：在MySQL中，没有删除CONSTRAINT的语句
【5.14】原因：MySQL不支持DOMAIN语句
【5.15】原因：MySQL不支持DOMAIN语句

【5.16】原因：MySQL不支持DOMAIN语句
【5.17】原因：MySQL不支持DOMAIN语句
【5.18】原因：MySQL不支持ASSERTION语句
【5.19】原因：MySQL不支持ASSERTION语句
【5.20】原因：MySQL不支持ASSERTION语句