

南京信息工程大学数据库系统原理实验（实习）报告

实验（实习）名称 数据库完整性和安全性 实验（实习）日期 2025-4-30 得分 指导教师
系 计算机学院 专业 计科

一、实验目的

1. 理解关系数据库完整性的基本概念；
2. 掌握 SQL Server 的 6 类约束：NOT NULL、PRIMARY KEY、CHECK、FOREIGN KEY、DEFAULT 和 UNIQUE，能够在创建表时用相应的约束准确地描述实体完整性、参照完整性和用户定义完整性；
3. 掌握增加和删除约束的方法；
4. 加深对数据库安全性的理解；
5. 掌握 SQL Server 中有关登录、（数据库）用户和权限的管理方法。

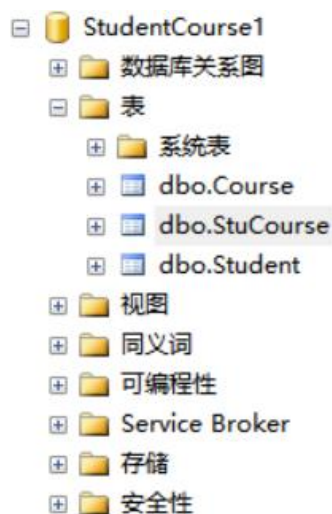
二、实验内容

- （一）数据库完整性
- （二）数据库安全性

三、实验过程与结果

- （一）数据库完整性

- 1、创建数据库 StudentCourse1。



- 2、分别按表 A9.1、表 A9.2 和表 A9.3 所示的结构和约束条件写出创建 3 个表的 CREATE 语句。

设计 SQL 语句：

```
USE StudentCourse1
```

```
GO
```

```
CREATE TABLE Student
```

```
(学号 CHAR(6) PRIMARY KEY,
```

```
姓名 CHAR(12) NOT NULL CONSTRAINT name_unique UNIQUE,
```

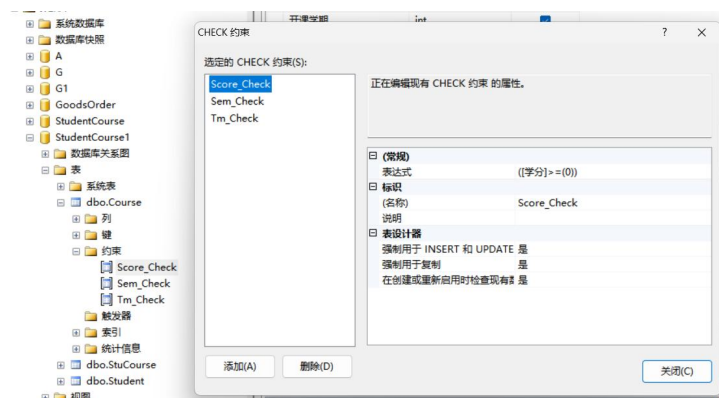
```
专业名 VARCHAR(20),
```

```

性别 CHAR (2) NOT NULL CONSTRAINT gen_Check CHECK (性别='男' OR 性别='女'),
出生时间 SMALLDATETIME,
总学分 INT CONSTRAINT tot_Check CHECK (总学分>=0),
备注 TEXT
)
GO
CREATE TABLE Course
( 课程号 CHAR(4) PRIMARY KEY,
课程名 VARCHAR(40) NOT NULL,
开课学期 INT CONSTRAINT Sem_Check CHECK (开课学期 BETWEEN 1 AND 8),
学时 INT CONSTRAINT Tm_Check CHECK ( 学时>=0),
学分 INT CONSTRAINT Score_Check CHECK ( 学分>=0)
)
GO
CREATE TABLE StuCourse
( 学号 CHAR(6) NOT NULL,
课程号 CHAR(4) NOT NULL,
成绩 INT CONSTRAINT Grade_Check CHECK ( 成绩>=0),
PRIMARY KEY (学号, 课程号),
FOREIGN KEY(学号) REFERENCES Student(学号),
FOREIGN KEY(课程号) REFERENCES Course(课程号)
)
GO

```

3、在对象资源管理器中查看约束。



(1) 主码约束测试

- 分析：向 Student 表中插入如下 SQL 语句，插入失败，因为“学号”字段违反了主码约束，不能为空。
- SQL 语句如下：


```
INSERT INTO Student
VALUES(NULL,'张萍萍','大气科学','女',NULL,NULL,NULL)
```
- 语句执行结果如下：



(2) 外码测试约束。

- 分析：先向 Student 和 Course 表中插入数据，再向 StuCourse 表中插入数据，因为不存在学号为“070105”的学生和课程号为“1003”的课程，违反外码约束条件，所以插入失败；接着向 StuCourse 表中插入数据，再从 Student 表中删除学号为“070101”的学生，结果显示删除失败，因为 StuCourse 中还有选课记录，不可直接删除。

- 设计的 SQL 语句：

```
INSERT INTO Student
VALUES('070101','丁一平','计算机科学与技术','男','1989-05-01',80,'三好学生')
INSERT INTO Student
VALUES('070102','王红','计算机科学与技术','女','1988-12-20',80,NULL)
INSERT INTO Course
VALUES('1001','高等数学','1',80,5)
INSERT INTO Course
VALUES('1002','高等数学','2',80,5)

INSERT INTO StuCourse VALUES('070105','1003',85)

INSERT INTO StuCourse VALUES('070101','1001',85)
INSERT INTO StuCourse VALUES('070101','1002',90)
INSERT INTO StuCourse VALUES('070102','1001',98)

DELETE FROM Student WHERE 学号='070101'
```

- 语句执行结果如下：



(3) 在查询分析器中输入以下插入数据的 T-SQL 语句并执行。

- 分析：插入数据失败，因为 Student 表的 name 属性不可以重复。
- 设计的 SQL 语句：

```
INSERT INTO Student
VALUES('070501','丁一平','大气科学','女',NULL,NULL,NULL)
UPDATE Student SET 姓名='丁一平' WHERE 学号='070102'
```

- 语句执行结果如下：



(4) check 约束测试。

- 分析：分别违反 Student 表中总学分大于等于 0 的约束，性别字段为“男”或“女”的约束；Course 表中学分大于等于 0 的约束，开课学期在 1-8 之间的约束，学时大于等于 0 的约束；StuCourse 表中成绩大于等于 0 的约束。
- 设计的 SQL 语句：

```
INSERT INTO Student
```

```
VALUES('070103','赵玲玲','计算机科学与技术','女','1999-10-2',-20,NULL)
```

```
INSERT INTO Student
```

```
VALUES('070103','赵玲玲','计算机科学与技术','或','1999-10-2',20,NULL)
```

```
INSERT INTO Course VALUES('1004','数据库',9,-1,-2)
```

```
INSERT INTO StuCourse VALUES('070102','1002',-99)
```

- 语句执行结果如下：



(5) NOT NULL 约束测试

- 分析：“课程名”字段值违反了 NOT NULL 约束。
- 设计的 SQL 语句：

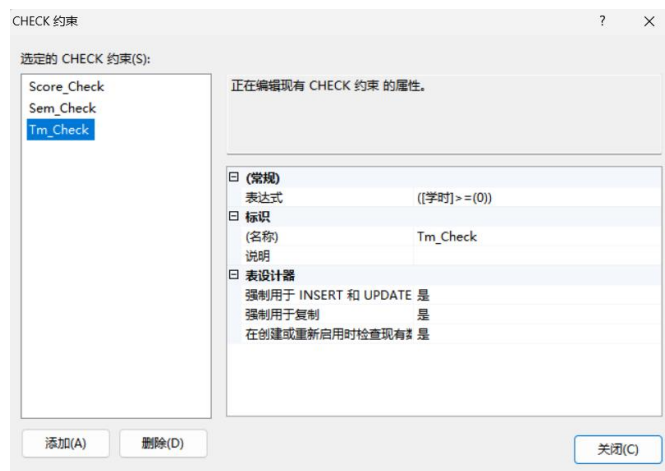
```
INSERT INTO Course VALUES('2008',NULL,3,32,2)
```

- 语句执行结果如下：



4、修改约束定义。

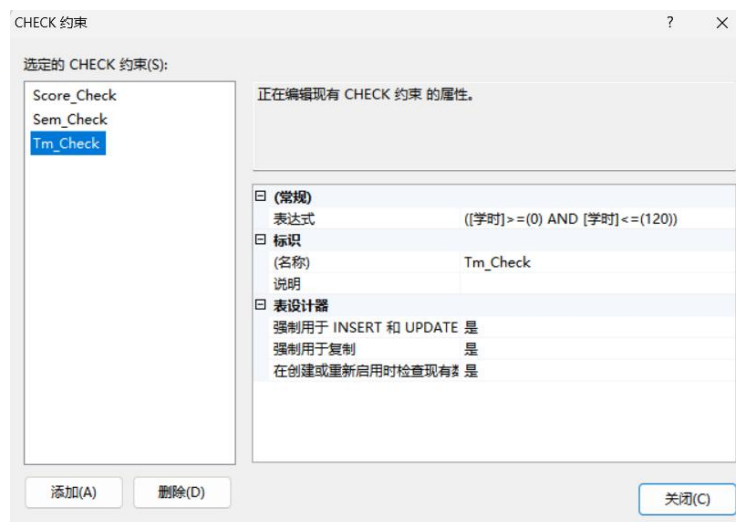
(1) 先查看 Tm_Check 约束内容：



(2) 在查询分析器中输入以下 T-SQL 语句并执行：

```
USE StudentCourse1
ALTER TABLE Course
DROP CONSTRAINT Tm_Check --先删除 Tm_Check 约束
GO
ALTER TABLE Course
ADD CONSTRAINT Tm_Check --再增加 Tm_Check 约束
CHECK (学时>=0 AND 学时<=120)
GO
```

(3) 查看修改后的约束内容：



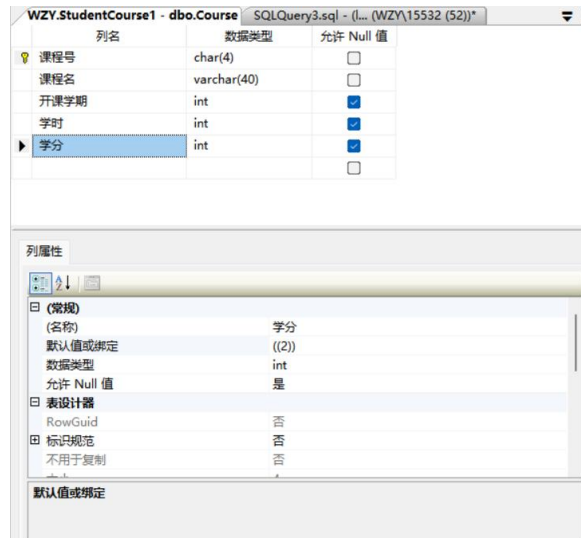
5、为 StudentCourse1 的 Course 表中学分字段增加“默认值为 2”的约束。

(1) 在查询分析器中输入以下 T-SQL 语句并执行：

```
USE StudentCourse1
ALTER TABLE Course
ADD CONSTRAINT Score_default --增加默认值约束 Score_default
Default(2) FOR 学分
```

GO

(2) 查看增加的约束内容:



(3) DEFAULT 约束测试:

在查询分析器中输入以下插入数据的 T-SQL 语句并执行:

```
INSERT INTO Course(课程号,课程名,开课学期,学时) VALUES('2008','数据库系统',3,32)
```

在查询分析器中输入查询语句并执行:

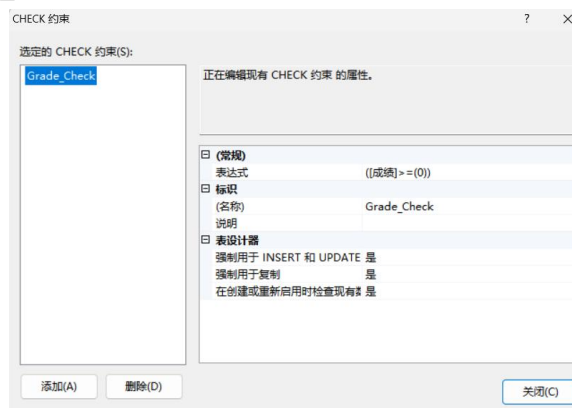
```
SELECT * FROM Course
```

执行结果如下:

	课程号	课程名	开课学期	学时	学分
1	1001	高等数学1	1	80	5
2	1002	高等数学2	2	80	5
3	2008	数据库系统	3	32	2

6、将 StudentCourse1 的 StuCourse 表中 Grade_Check 约束修改为取值范围为 0~100。

(1) 先查看 Grade_Check 约束内容:



(2) 在查询分析器中输入以下 T-SQL 语句并执行:

```
USE StudentCourse1
```

```
--检查是否已存在同名的约束，若有，则删除。
```

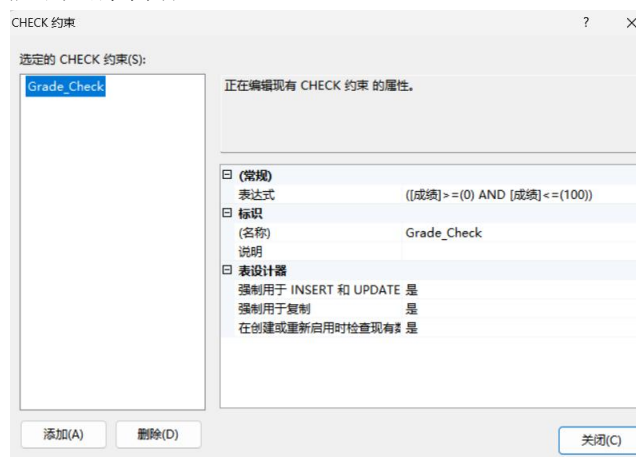
```
IF EXISTS (SELECT name FROM sysobjects
```

```

WHERE name = 'Grade_Check' AND type = 'C')
ALTER TABLE StuCourse
DROP CONSTRAINT Grade_Check --先删除 Grade_Check 约束
GO
ALTER TABLE StuCourse
ADD CONSTRAINT Grade_Check --再增加 Grade_Check 约束
CHECK (成绩>=0 AND 成绩<=100)
GO

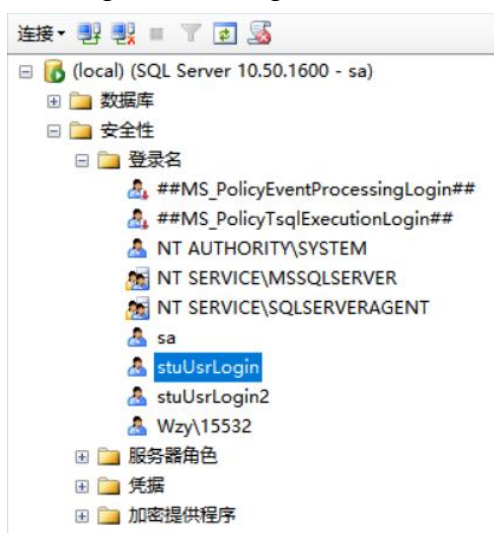
```

(3) 查看修改后的约束内容：



(二) 数据库安全性

- 1、将 SQL Server 服务器的身份验证模式设置为“SQL Server 身份验证模式”。
- 2、创建两个登录名：stuUsrLogin、stuUsrLogin2：



3、以 sa 登录，分别为登录名 stuUsrLogin、stuUsrLogin2 创建映射两个数据库用户 stuUsr、stuUsr2，使这两个登录名可访问 StudentCourse 数据库，并授予他们可以查询 Student 表的权限。SQL 语句如下：

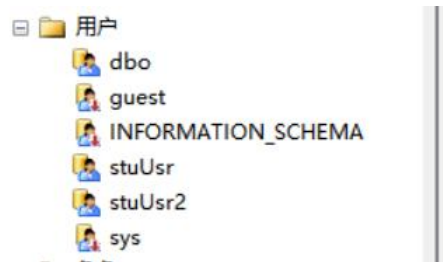
```

USE StudentCourse
CREATE USER stuUsr FOR LOGIN stuUsrLogin
CREATE USER stuUsr2 FOR LOGIN stuUsrLogin2
GRANT SELECT ON Student TO stuUsr, stuUsr2

```


GO

结果如图所示：



4、验证第 3 步的授权。重新以 stuUsrLogin 登录 SQL Server，查看数据库、表信息。



5、重新以 sa 登录，将对 Student 表的 INSERT 权限授予 stuUsr，并允许他将此权限再授予其他用户。语句如下：

```
USE StudentCourse
```

```
GRANT INSERT ON Student TO stuUsr WITH GRANT OPTION
```

```
GO
```

6、重新以 stuUsrLogin 登录，向 Student 表的插入一条记录，能够正常插入记录，表明权限授予正确。语句如下：

```
USE StudentCourse; -- 或你表所在的数据库名
```

```
GO
```

```
INSERT INTO Student Values('070210','张红红','电子信息工程','女','1991-2-10',76,'三好生')
```




```
SQLQuery7.sql - (lo...ntCourse (sa (62)))* SQLQuery6.sql - (l...(stuUsrLogin (60)))*
USE StudentCourse; -- 或你表所在的数据库名
GO
INSERT INTO Student VALUES ('070210', '张红红', '电子信息工程', '女', '1991-2-10', 76,
```

消息

(1 行受影响)

7、将 Student 表的插入权限授予 stuUsr2，语句如下。

```
USE StudentCourse; -- 或你表所在的数据库名
```

```
GO
```

```
GRANT INSERT ON Student TO stuUsr2
```

8、验证对 stuUsr2 的授权。重新以 stuUsrLogin2 登录，向 Student 表的插入一条记录，能够正常插入记录，表明权限授予正确。

语句如下：

```
USE StudentCourse; -- 或你表所在的数据库名
```

```
GO
```

```
INSERT INTO Student Values('070120','李小力','计算机科学与技术','男','1990-5-1',78,'三好生')
```



```
SQLQuery9.sql - (l...tuUsrLogin2 (53))* SQLQuery8.sql - (l...(stuUsrLogin (63)))
USE StudentCourse; -- 或你表所在的数据库名
GO
INSERT INTO Student Values('070120', '李小力', '计算机科学与技术', '男', '1990-5-1', 78, '三好生'
```

消息

(1 行受影响)

9、把授予 stuUsr2 用户对 Student 表插入的权限收回，重新以 stuUsrLogin 登录：

```
USE StudentCourse; -- 或你表所在的数据库名
```

```
GO
```

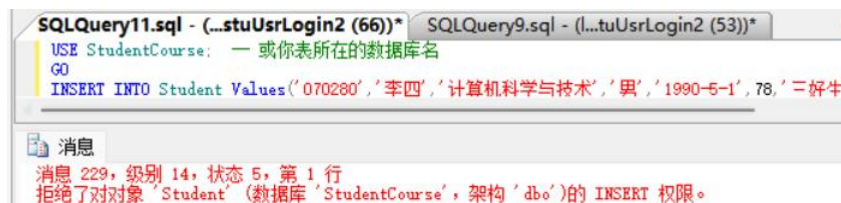
```
REVOKE INSERT ON Student FROM stuUsr2
```

10、验证对 stuUsr2 用户权限收回操作是否成功。重新以 stuUsrLogin2 登录：

```
USE StudentCourse; -- 或你表所在的数据库名
```

```
GO
```

```
INSERT INTO Student Values('070280','李四','计算机科学与技术','男','1990-5-1',78,'三好生')
```



```
SQLQuery11.sql - (...stuUsrLogin2 (66))* SQLQuery9.sql - (l...tuUsrLogin2 (53))*
USE StudentCourse; -- 或你表所在的数据库名
GO
INSERT INTO Student Values('070280', '李四', '计算机科学与技术', '男', '1990-5-1', 78, '三好生'
```

消息

消息 229, 级别 14, 状态 5, 第 1 行
拒绝对对象 'Student' (数据库 'StudentCourse', 架构 'dbo')的 INSERT 权限。

四、实验总结

通过本次数据库完整性与安全性实验，我掌握了常用约束（如 **PRIMARY KEY**、**CHECK**、**FOREIGN KEY** 等）的使用方法，理解了实体完整性、参照完整性和用户自定义完整性的实现方式。同时，我学会了在 **SQL Server** 中创建登录名、用户并设置权限，能熟练使用 **GRANT**、**REVOKE** 等语句实现权限控制。实验增强了我对数据库约束机制和安全管理理解，也提升了实际操作与故障排查能力。