Database-System Experiment-10

21307289 刘森元

1. 实验目的

学习实体完整性的建立，以及实践违反实体完整性的结果；学习建立外键，以及利用 FOREIGN KEY…REFERENCES子句以及各种约束保证参照完整性。

2. 实验环境

Macbook Pro 2021 (Apple M1 Pro)

macOS Ventura 13.5.2

PostgreSQL 15.4 (Homebrew)

zsh 5.9

3. 实验步骤

重建数据库

```
-- 重建数据库
DROP DATABASE "school";
CREATE DATABASE "school";
\c school
\cd '/Users/qiu_nangong/Documents/Github/Database-System/Experiment-10'
\i STUDENTS.sql
\i TEACHERS.sql
\i COURSES.sql
\i CHOICES.sql
```

课内实验

1）在数据库 school中建立表Stu_Union，进行主键约束，在没有违反实体完整性的前提下插入并更新一条记录。(参考代码如下：)

```
CREATE TABLE Stu_Union(
  sno CHAR(5) NOT NULL UNIQUE,
  sname CHAR(8),
  ssex CHAR(1),
  sage INT,
  sdept CHAR(20),
  CONSTRAINT PK PRIMARY KEY(sno)
);
```

```
insert Stu_Union values('10000','王敏','1',23,'cs);

UPDATE Stu_Union SET sno='' WHERE sdept='CS';
UPDATE Stu_Union SET sno='95002' WHERE sname='王敏';

select * from Stu_Union;
```

按照参考代码建立表格并插入数据

```
school=# \d stu_union
              数据表 "public.stu_union"
 栏位   |      类型       | 校对规则 |   可空的   | 预设
-------+-----------------+----------+----------+------
 sno   | character(5)    |          | not null |
 sname | character(8)    |          |          |
 ssex  | character(1)    |          |          |
 sage  | integer         |          |          |
 sdept | character(20)   |          |          |
索引：
    "pk" PRIMARY KEY, btree (sno)
```

```
school=# SELECT * FROM stu_union
school-# ;
  sno   |   sname    | ssex | sage |        sdept
-------+------------+------+------+----------------------
 10000 | 王敏       | 1    |   23 | cs
(1 行记录)
```

可见表格成功创建，并带有主键约束

不违反实体完整性的前提下插入并更新一条记录

```
UPDATE Stu_Union SET sno = '95002' WHERE sname = '王敏';
```

```
school=# UPDATE Stu_Union SET sno = '95002' WHERE sname = '王敏';
UPDATE 1
school=# SELECT * FROM stu_union;
  sno  |   sname    | ssex | sage |        sdept
-------+------------+------+------+---------------------
 95002 | 王敏       | 1    |   23 | cs
(1 行记录)
```

```
UPDATE Stu_Union SET sno = '' WHERE sname = '王敏';
```

```
school=# UPDATE Stu_Union SET sno = '' WHERE sname = '王敏';
UPDATE 1
school=# SELECT * FROM stu_union;
  sno  |   sname    | ssex | sage |        sdept
-------+------------+------+------+---------------------
       | 王敏       | 1    |   23 | cs
(1 行记录)
```

2）演示违反实体完整性的插入操作。

```
INSERT INTO Stu_Union VALUES (NULL, '李强', '1', 25 ,'cg');
```

```
school=# INSERT INTO Stu_Union VALUES (NULL, '李强', '1', 25 ,'cg');
ERROR:  null value in column "sno" of relation "stu_union" violates not-null constraint
描述:  Failing row contains (null, 李强      , 1, 25, cg                ).
```

```
INSERT INTO Stu_Union VALUES ('10000', '李强', '0', 25, 'cg');
```

```
school=# INSERT INTO Stu_Union VALUES ('10000', '李强', '0', 25, 'cg');
ERROR:  duplicate key value violates unique constraint "pk"
描述:  Key (sno)=(10000) already exists.
```

3）演示违反实体完整性的更新操作。

```
UPDATE Stu_Union SET sno = NULL WHERE sname = '王敏';
```

```
school=# UPDATE Stu_Union SET sno = NULL WHERE sname = '王敏';
ERROR:  null value in column "sno" of relation "stu_union" violates not-null constraint
描述:  Failing row contains (null, 王敏      , 1, 23, cs                ).
```

```
UPDATE Stu_Union SET sno = '10000' WHERE sage = 25;
```

```
school=# SELECT * FROM stu_union ;
  sno   |   sname    | ssex | sage |        sdept
--------+------------+------+------+---------------------
 10000  | 王敏       | 1    |   23 | cs
 10001  | 李强       | 0    |   25 | cg
(2 行记录)

school=# UPDATE Stu_Union SET sno = '10000' WHERE sage = 25;
ERROR:  duplicate key value violates unique constraint "pk"
描述:  Key (sno)=(10000) already exists.
```

4）为演示参照完整性，建立表 Course,令 cno 为其主键，并在 Stu_Union 中插入数据。为下面的实验步骤做预先准备。(参考代码如下：)

```
insert into Stu_Union values('10001','李勇','0',24,'EE');

select * from Stu_Union;

create TABLE Course(
  cno char(4)NOT NULL UNIQUE,
  cname varchar(50)NOT NULL,
  cpoints int,
  CONSTRAINT PK_course primary KEY(cno)
);

insert into Course values('0001','ComputerNetworks',2);
insert into Course values('0002','Databsae',3);
```

```
school=# SELECT * FROM Course
school-# ;
 cno   |       cname        | cpoints
-------+--------------------+---------
 0001  | ComputerNetworks   |       2
 0002  | Databsae           |       3
(2 行记录)
```

5) 建立表 SC，令 sno 和 cno 分别为参照 stu union 表以及 Course 表的外键，设定为 级联删除，并令(sno,cno) 为其主键。在不违反参照完整性的前提下，插入数据。(参考代码如下：)

```
CREATE TABLE SC(
    Sno CHAR(5) REFERENCES Stu_Union(sno) ON DELETE CASCADE,
    Cno CHAR(4) REFERENCES Course(cno) ON DELETE CASCADE,
    grade INT,
    CONSTRAINT PK_sc PRIMARY KEY(sno, cno)
);
INSERT INTO SC VALUES('10001', '0001', 2);
INSERT INTO SC VALUES('10001', '0002', 2);

SELECT * FROM SC;
```

```
  sno   | cno  | grade
-------+------+-------
 10001 | 0001 |     2
 10001 | 0002 |     2
(2 行记录)
```

演示不违反参照完整性的插入数据

```
INSERT INTO SC VALUES('10000', '0001', 2);
INSERT INTO SC VALUES('10000', '0002', 2);
```

```
school=# SELECT * FROM SC;
  sno   | cno  | grade
-------+------+-------
 10001 | 0001 |     2
 10001 | 0002 |     2
 10000 | 0001 |     2
 10000 | 0002 |     2
(4 行记录)
```

6）演示违反参照完整性的插入数据。

```
INSERT INTO SC VALUES('95002', '0001', 2);
INSERT INTO SC VALUES('95002', '0002', 2);
```

```
school=# INSERT INTO SC VALUES('95002', '0001', 2);
INSERT INTO SC VALUES('95002', '0002', 2);
ERROR:  insert or update on table "sc" violates foreign key constraint "sc_sno_fkey"
描述：  Key (sno)=(95002) is not present in table "stu_union".
ERROR:  insert or update on table "sc" violates foreign key constraint "sc_sno_fkey"
描述：  Key (sno)=(95002) is not present in table "stu_union".
```

这是因为 `sno = '95002'` 的项不存在

7）在 Stu_Union 中删除数据，演示级联删除。

先插入用于演示的行

```
INSERT INTO Stu_Union VALUES('95002', 'TEST', '', NULL, NULL);
INSERT INTO SC VALUES('95002', '0001', 2);
INSERT INTO SC VALUES('95002', '0002', 2);
```

```
school=# SELECT * FROM SC;
  sno   | cno   | grade
-------+------+-------
 10001 | 0001 |     2
 10001 | 0002 |     2
 10000 | 0001 |     2
 10000 | 0002 |     2
 95002 | 0001 |     2
 95002 | 0002 |     2
(6 行记录)

school=# SELECT * FROM Stu_Union;
  sno   |   sname    | ssex | sage |        sdept
-------+------------+------+------+----------------------
 10000 | 王敏       | 1    |   23 | cs
 10001 | 李勇       | 0    |   24 | EE
 95002 | TEST       |      |      |
(3 行记录)
```

进行级联删除

```
DELETE FROM Stu_Union WHERE sname = 'TEST';
```

```
school=# SELECT * FROM SC;
  sno   | cno  | grade
-------+------+-------
 10001 | 0001 |     2
 10001 | 0002 |     2
 10000 | 0001 |     2
 10000 | 0002 |     2
(4 行记录)

school=# SELECT * FROM Stu_Union;
  sno   |   sname    | ssex | sage |        sdept
-------+------------+------+------+----------------------
 10000 | 王敏       | 1    |   23 | cs
 10001 | 李勇       | 0    |   24 | EE
(2 行记录)
```

8）在 Course 中删除数据，演示级联删除。

先插入用于演示的行

```
INSERT INTO Course VALUES('TEST', 'TEST', NULL);
INSERT INTO SC VALUES('10000', 'TEST', 2);
INSERT INTO SC VALUES('10001', 'TEST', 2);
```

```
school=# SELECT * FROM SC;
  sno   | cno  | grade
-------+------+-------
 10001 | 0001 |     2
 10001 | 0002 |     2
 10000 | 0001 |     2
 10000 | 0002 |     2
 10000 | TEST |     2
 10001 | TEST |     2
(6 行记录)

school=# SELECT * FROM Stu_Union;
  sno   |   sname    | ssex | sage |        sdept
-------+------------+------+------+---------------------
 10000 | 王敏       | 1    |   23 | cs
 10001 | 李勇       | 0    |   24 | EE
(2 行记录)
```

进行级联删除

```
DELETE FROM Course WHERE cno = 'TEST';
```

```
school=# SELECT * FROM SC;
  sno   | cno  | grade
-------+------+-------
 10001 | 0001 |     2
 10001 | 0002 |     2
 10000 | 0001 |     2
 10000 | 0002 |     2
(4 行记录)

school=# SELECT * FROM Stu_Union;
  sno   |   sname    | ssex | sage |        sdept
-------+------------+------+------+---------------------
 10000 | 王敏       | 1    |   23 | cs
 10001 | 李勇       | 0    |   24 | EE
(2 行记录)
```

自我实践

1) 用 ALTER TABLE 语句将SC 表中的 ON DELETE cascade 改为 ON DELETE NO ACTION,重新插 入SC 的数据。重复课内实验中7.和8.,观察结果，分析原因。

先观察`SC`表中目前已有的约束名

```
school=# \d SC
                数据表 "public.sc"
 栏位   |     类型      | 校对规则 |  可空的   | 预设
-------+--------------+----------+----------+------
 sno   | character(5) |          | not null |
 cno   | character(4) |          | not null |
 grade | integer      |          |          |
索引：
    "pk_sc" PRIMARY KEY, btree (sno, cno)
外部键(FK)限制：
    "sc_cno_fkey" FOREIGN KEY (cno) REFERENCES course(cno) ON DELETE CASCADE
    "sc_sno_fkey" FOREIGN KEY (sno) REFERENCES stu_union(sno) ON DELETE CASCADE
```

很显然，我们接下来关注的就是它的外键约束`sc_cno_fkey`和`sc_sno_fkey`。

需要注意的是，我们不能在已有外键约束的基础上直接进行修改，作为替代，我们需要先行删除已有的外键约束，然后新增我们所期望的外键约束，利用`ALTER TABLE`操作，具体如下：

```
ALTER TABLE SC DROP CONSTRAINT sc_cno_fkey;
ALTER TABLE SC DROP CONSTRAINT sc_sno_fkey;

ALTER TABLE SC ADD CONSTRAINT sc_cno_fkey
FOREIGN KEY (cno) REFERENCES course (cno) ON DELETE NO ACTION;

ALTER TABLE SC ADD CONSTRAINT sc_sno_fkey
FOREIGN KEY (sno) REFERENCES stu_union (sno) ON DELETE NO ACTION;
```

```
school=# \d SC
                数据表 "public.sc"
 栏位   |     类型      | 校对规则 |  可空的   | 预设
-------+--------------+----------+----------+------
 sno   | character(5) |          | not null |
 cno   | character(4) |          | not null |
 grade | integer      |          |          |
索引：
    "pk_sc" PRIMARY KEY, btree (sno, cno)
外部键(FK)限制：
    "sc_cno_fkey" FOREIGN KEY (cno) REFERENCES course(cno) ON DELETE CASCADE
    "sc_sno_fkey" FOREIGN KEY (sno) REFERENCES stu_union(sno) ON DELETE CASCADE
```

重复 7)

```
INSERT INTO Stu_Union VALUES('95002', 'TEST', '', NULL, NULL);
INSERT INTO SC VALUES('95002', '0001', 2);
INSERT INTO SC VALUES('95002', '0002', 2);
DELETE FROM Stu_Union WHERE sname = 'TEST';
```

```
school=# INSERT INTO Stu_Union VALUES('95002', 'TEST', '', NULL, NULL);
INSERT INTO SC VALUES('95002', '0001', 2);
INSERT INTO SC VALUES('95002', '0002', 2);
DELETE FROM Stu_Union WHERE sname = 'TEST';
INSERT 0 1
INSERT 0 1
INSERT 0 1
ERROR:  update or delete on table "stu_union" violates foreign key constraint "sc_sno_fkey" on table "sc"
描述：  Key (sno)=(95002) is still referenced from table "sc".
```

删除失败，违反的是**参照性约束**，要删除该行，必须先将从表中所有对此行的引用都删除之后，才可行。

只能由用户先自行删除从表数据，在删除主表数据，如下：

```
DELETE from SC where sno = '95002';
DELETE from stu_union where sname = 'test';
```

```
school=# SELECT * FROM Stu_union
school-# ;
  sno   |   sname     | ssex | sage |        sdept
--------+-------------+------+------+----------------------
 10000  | 王敏        | 1    |   23 | cs
 10001  | 李勇        | 0    |   24 | EE
 95002  | TEST        |      |      |
(3 行记录)
```

重复 8)

```
INSERT INTO Course VALUES('TEST', 'TEST', NULL);
INSERT INTO SC VALUES('10000', 'TEST', 2);
INSERT INTO SC VALUES('10001', 'TEST', 2);
DELETE FROM Course WHERE cno = 'TEST';
```

```
school=# INSERT INTO Course VALUES('TEST', 'TEST', NULL);
INSERT INTO SC VALUES('10000', 'TEST', 2);
INSERT INTO SC VALUES('10001', 'TEST', 2);
DELETE FROM Course WHERE cno = 'TEST';
INSERT 0 1
INSERT 0 1
INSERT 0 1
ERROR:  update or delete on table "course" violates foreign key constraint "sc_cno_fkey" on table "sc"
描述：  Key (cno)=(TEST) is still referenced from table "sc".
```

失败原因与 7) 同理。

2) 使用 ALTER TABLE 语句将 SC 表中的 ON DELETE cascade 改为 ON DELETE SET NULL,重 新插入 SC 的数据。
重复课内实验中7.和8.,观察结果，分析原因。

修改约束

```
ALTER TABLE SC DROP CONSTRAINT sc_cno_fkey;
ALTER TABLE SC DROP CONSTRAINT sc_sno_fkey;

ALTER TABLE SC ADD CONSTRAINT sc_cno_fkey
FOREIGN KEY (cno) REFERENCES course (cno) ON DELETE SET NULL;

ALTER TABLE SC ADD CONSTRAINT sc_sno_fkey
FOREIGN KEY (sno) REFERENCES stu_union (sno) ON DELETE SET NULL;
```

```
school=# \d sc
                数据表 "public.sc"
 栏位   |     类型      | 校对规则 |  可空的   | 预设
-------+--------------+----------+----------+------
 sno   | character(5) |          | not null |
 cno   | character(4) |          | not null |
 grade | integer      |          |          |
索引：
    "pk_sc" PRIMARY KEY, btree (sno, cno)
外部键(FK)限制：
    "sc_cno_fkey" FOREIGN KEY (cno) REFERENCES course(cno) ON DELETE SET NULL
    "sc_sno_fkey" FOREIGN KEY (sno) REFERENCES stu_union(sno) ON DELETE SET NULL
```

重复 7)

```
INSERT INTO Stu_Union VALUES('95002', 'TEST', '', NULL, NULL);
INSERT INTO SC VALUES('95002', '0001', 2);
INSERT INTO SC VALUES('95002', '0002', 2);
DELETE FROM Stu_Union WHERE sname = 'TEST';
```

```
school=# INSERT INTO Stu_Union VALUES('95002', 'TEST', '', NULL, NULL);
INSERT INTO SC VALUES('95002', '0001', 2);
INSERT INTO SC VALUES('95002', '0002', 2);
DELETE FROM Stu_Union WHERE sname = 'TEST';
ERROR:  duplicate key value violates unique constraint "pk"
描述：  Key (sno)=(95002) already exists.
INSERT 0 1
INSERT 0 1
ERROR:  null value in column "sno" of relation "sc" violates not-null constraint
描述：  Failing row contains (null, 0001, 2).
背景：  SQL statement "UPDATE ONLY "public"."sc" SET "sno" = NULL WHERE $1 OPERATOR(pg_catalog.=) "sno""
school=#
```

可以看到，依旧报错，无法删除成功，这是因为在表格`SC`中，列`sno`不允许被设置成`NULL`，因此`ON DELETE SET NULL`无法成功执行，主表的删除操作也无法成功。

重复 8)

```
INSERT INTO Course VALUES('TEST', 'TEST', NULL);
INSERT INTO SC VALUES('10000', 'TEST', 2);
INSERT INTO SC VALUES('10001', 'TEST', 2);
DELETE FROM Course WHERE cno = 'TEST';
```

```
school=# INSERT INTO Course VALUES('TEST', 'TEST', NULL);
INSERT INTO SC VALUES('10000', 'TEST', 2);
INSERT INTO SC VALUES('10001', 'TEST', 2);
DELETE FROM Course WHERE cno = 'TEST';
ERROR:  duplicate key value violates unique constraint "pk_course"
描述:  Key (cno)=(TEST) already exists.
ERROR:  duplicate key value violates unique constraint "pk_sc"
描述:  Key (sno, cno)=(10000, TEST) already exists.
ERROR:  duplicate key value violates unique constraint "pk_sc"
描述:  Key (sno, cno)=(10001, TEST) already exists.
ERROR:  null value in column "cno" of relation "sc" violates not-null constraint
描述:  Failing row contains (10000, null, 2).
背景:  SQL statement "UPDATE ONLY "public"."sc" SET "cno" = NULL WHERE $1 OPERATOR(pg_catalog.=) "cno""
```

失败原因与 7) 同理。

4. 实验心得

在本次实验中，我学习了实体完整性的建立以及参照完整性的实践。通过创建表格和插入数据的操作，我深入了解了实体完整性和参照完整性的概念和作用。首先，我创建了一个名为"Stu_Union"的表格，并为其指定了主键约束。通过插入和更新记录的操作，我观察到在不违反实体完整性的前提下，成功地插入和更新了记录。接着，我创建了一个名为"Course"的表格，并为其指定了主键约束。在"Stu_Union"表格中插入了一些数据作为参照完整性的准备。然后，我创建了一个名为"SC"的表格，并为其指定了外键约束，参照了"Stu_Union"和"Course"表格的主键。通过插入数据的操作，我演示了不违反参照完整性的情况下成功插入数据。接下来，我演示了违反参照完整性的插入数据操作。由于插入的数据在主表中不存在，因此违反了参照完整性约束，插入操作失败。随后，我演示了级联删除的操作。通过在主表中删除数据，观察到从表中相关数据也被自动删除，实现了级联删除的效果。在自我实践部分，我使用ALTER TABLE语句修改了"SC"表的外键约束，将ON DELETE CASCADE改为ON DELETE NO ACTION和ON DELETE SET NULL。通过重复之前的插入和删除操作，我观察到在ON DELETE NO ACTION约束下，删除操作失败，违反了参照完整性约束；而在ON DELETE SET NULL约束下，由于列不允许设置为NULL，删除操作同样失败。通过本次实验，我深入理解了实体完整性和参照完整性的概念和作用，掌握了在数据库中建立约束和实践参照完整性的方法。这对于保证数据库数据的完整性和一致性非常重要，也为后续数据库设计和操作提供了基础。