数据库实验lab9

曹广杰

15352015 数据科学与计算机

授课教师：刘玉葆

2017/10/20

Content

实验目的

1. 学习建立外键
2. 了解利用FOREIGN KEY和REFERENCES子句以及各种约束保证参照完整性

实验内容

修改键的约束

本次实验中由于涉及到关于删除表单的操作，所以需要对输入的命令进行更为谨慎的操作，在进行实验之前，笔者使用了事务的架构：

```
use School
set xact_abort on
begin transaction  tmp
...
commit transaction tmp
```

把具体的操作和实现过程都写在中间的部分，这样，在当前的部分尚未完成之前，最大可能地避免了错误的删除，这就避免了很多无谓的修复操作。

修改约束与插入删除操作

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（按照实验一）。再删除Stu_Union中sno为'10001'的数据。观察结果，并分析原因。

```
alter table sc
    drop constraint FK__sc__cno__4F7CD00D
alter table sc
    add constraint FK_sc_cno foreign key (cno)
        references course(cno) on delete no action
alter table sc
    drop constraint FK__sc__sno__4E88ABD4
alter table sc
    add constraint FK_sc_sno foreign key (sno)
        references stu_union(sno) on delete no action
```

之后就是重新插入SC的数据的操作了——

```
insert into sc values ('95002', '0001', 2)
insert into sc values ('95002', '0002', 2)
insert into sc values ('10001', '0001', 2)
insert into sc values ('10001', '0002', 2)
```

由于在数据库的表格内已经有该数据了，所以对于重复的数据插入并不能成功。

删除Stu_Union中sno为'10001'的数据：

```
delete
from stu_union
where sno = '10001'
```

删除操作失败：`消息 547，级别 16，状态 0，第 1 行DELETE 语句与 REFERENCE 约";FK_sc_sno";冲突。该冲突发生于数据库";School";，表";dbo.sc";，column ';sno';。`这是因为我们设置当前的属性约束为on delete no action，因此在该项数据属性有数据存在时不允许删除。

修改键的约束为**on delete set NULL**

用alter table语句将SC表中的on delete no action改为on delete set NULL，重新插入SC的数据（按照实验一）。再删除Stu_Union中sno为'10001'的 数据。观察结果，并分析原因

```
alter table sc
    drop constraint FK_sc_cno
alter table sc
    add constraint FK_sc_cno foreign key (cno)
        references course(cno) on delete set null
alter table sc
    drop constraint FK_sc_sno
alter table sc
    add constraint FK_sc_sno foreign key (sno)
        references stu_union(sno) on delete set null
```

实验结果显示，修改约束的行为失败：`由于一个或多个引用列不可为 Null，因此无法使用 SET NULL 引用操作创建外键";FK_sc_cno";`，这是因为我们在创建这关系的时候对于以上的变量sno和cno都设置为不可为NULL，这与当前的操作发生矛盾，故而不能修改。

综上：

1. 可以为关系中的属性设置约束
2. 目前涉及到的属性有
    - on delete cascade
    - on delete no action
    - on delete set null

    其实对应的，on update也有同样的属性约束。

句法：

1. 删除约束

    `alter table` 修改的关系名

    `drop constraint` 删除的属性约束名

2. 添加外键约束

    `alter table` 修改的关系名

    `add constraint` 添加的约束名称 `foreign key` (当前关系内被选中的外键)

    `references` 另一个关系(另一个关系的主键) `on delete set null`

修改约束与插入删除操作

修改ICBC_Card表的外键属性，使其变为on delete set NULL, 尝试删除students表中一条记录。

```
alter table icbc_card
    drop constraint FK__icbc_card__stu_c__5535A963
alter table icbc_card
    add constraint FK__icbc_card__stu_c foreign key (stu_card_id)
        references stu_card(card_id) on delete set null
```

这里可以实现修改的操作，但是在删除的时候，会出现：“DELETE语句与REFERENCE约束

"FK_CHOICES_STUDENTS"冲突。该冲突发生于数据库"School"，表"dbo.CHOICES", column 'sid'。”

这是因为删除的时候，由于choices中的属性与students中的属性关联了外键，使得不能实现删除操作。

参照完整性与互相参照

自定义实现表的自参照

创建一个班里的学生互助表，规定：包括学生编号，学生姓名，学生的帮助对 象，每个学生有且仅有一个帮助对象，帮助对象也必须是班里的学生。（表的自参照问题）

```
create table stu_help(
    stu_id char(6),
    sname varchar(20),
    recipient char(6)
    constraint pk_stu_help primary key(stu_id)
    )
alter table stu_help
    add constraint fk_stu_help foreign key(recipient)
                                    references stu_help(stu_id)
```

题意分析：

由于接受资助者也属于当前的关系，为了避免自身的调用，就一定要将。

- 如果保留字作用在score上，就意味着一个score只能出现一次，那么有多个学号有同样的分数，学号也就只能出现一次——这就造成了信息的丢失。
- 如果作用在保留字上，一个学生可能有多个成绩——但是由笔者实验得知，在本次实验中，每位学生至多有两个成绩，而且有两个成绩的学号中一定有一个成绩显示为Null。

为了保持程序的正常运行，如果我是设计者，我会选择使用第二种方案，事实上sql 的设计也是这样——distinct只能添加在sid的前面作为限定。

这里就涉及到Null与已有成绩的取舍问题。在一个人有两个成绩二其中一个是Null的时候，我们在查询中希望见到哪一个成绩呢？很显然，Null的成绩是没有意义的，故而在本次查询中，查询的信息就只是非Null的成绩，Null的部分被覆盖了。

实现表的互相参照

学校学生会的每个部门都有一个部长，每个部长领导多个部员，每个部只有一 个部员有评测部长的权利，请给出体现这两种关系（领导和评测）的两张互参照的表的定义。（两个表互相参照的问题）

```
create table manage(
    member_id char(6),
    manager_id char(6),
    constraint pk_manage primary key(member_id)
    )
create table supervis(
    viser_id char(6),
    vised_id char(6),
    constraint pk_supervis primary key(viser_id, vised_id)
    )
```

由于实验要求每一位部长只能有一位成员具有监督权利，因此，“部长”与“监督者”之间的关系是典型的函数映射，为此，一定要实现一个新的关系以保证监督者的唯一性。此后的实现就是基于普通的互相参照，是两个表格之间的关系了：

```sql
create table employer(
    leader_id char(6),
    follower_id char(6),
    supervis char (6),
    constraint pk_employer primary key(leader_id),
    constraint fk_employer_vis foreign key(supervis)
                            references supervis(viser_id),
    constraint fk_employer_led foreign key(leader_id)
                            references supervis(vised_id)
    )
create table employee(
    follower_id char(6),
    leader_id char(6),
    constraint pk_employee primary key(follower_id),
    constraint fk_employee_fllw foreign key(follower_id)
                                  references manage(member_id)
    constraint fk_employee_lead foreign key(leader_id)
                                  references manage(manager_id)
    )

```

两个表格中的变量分别描述当前的实体集的状态信息，使用外键的约束是为了实现单元之间真正的约束，表示几个角色之间的关系。

综上可见，在sql的设计中，基本的运算单元为了可以大程度上地表示数据的趋势，都是对Null视而不见的，这样就避免了平时数据中经常出现的Null对于数据分析的不利影响。