

实验报告10

实验十 数据库完整性

实验目的

课内实验

自我实践

实验心得

实验十 数据库完整性

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 FOREIGNKEY...REFERENCES子句以及各种约束保证参照完整性。

会用到以下几种类型的 SQL 语句：

1. 创建表 (CREATE TABLE) :

```
1 CREATE TABLE Stu_Union(  
2     sno CHAR(5) NOT NULL UNIQUE,  
3     sname CHAR(8),  
4     ssex CHAR(1),  
5     sage INT,  
6     sdept CHAR(20),  
7     CONSTRAINT PK_Stu_Union PRIMARY KEY(sno)  
8 );
```

2. 插入数据 (INSERT INTO) :



Plain Text |

```
1  INSERT INTO Stu_Union VALUES('10000','王敏','1',23,'cs');
2  INSERT INTO Course VALUES('0001','ComputerNetworks',2);
```

3. 更新数据 (UPDATE) :



Plain Text |

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

4. 查询数据 (SELECT) :



Plain Text |

```
1  SELECT * FROM Stu_Union;
```

5. 删除数据 (DELETE FROM) :



Plain Text |

```
1  DELETE FROM Stu_Union WHERE sno = '某个学生的学号';
```

6. 创建外键约束 (CREATE TABLE with FOREIGN KEY) :



Plain Text |

```
1  CREATE TABLE SC(
2      Sno CHAR(5) REFERENCES Stu_Union(sno) ON DELETE CASCADE,
3      Cno CHAR(4) REFERENCES Course(cno) ON DELETE CASCADE,
4      grade INT,
5      CONSTRAINT PK_SC PRIMARY KEY(sno,cno)
6  );
```

7. 修改表结构 (ALTER TABLE) :



Plain Text |

```
1  ALTER TABLE SC
2  DROP FOREIGN KEY fk_SC_Stu_Union;
3
4  ALTER TABLE SC
5  ADD CONSTRAINT fk_SC_Stu_Union
6  FOREIGN KEY (Sno) REFERENCES Stu_Union(sno)
7  ON DELETE SET NULL;
```

8. 添加外键约束 (ALTER TABLE with ADD CONSTRAINT) :

▼ Plain Text |

```
1 ALTER TABLE SC
2 ADD CONSTRAINT fk_SC_Course
3 FOREIGN KEY (Cno) REFERENCES Course(cno)
4 ON DELETE SET NULL;
```

9. 删除外键约束 (ALTER TABLE with DROP FOREIGN KEY) :

▼ Plain Text |

```
1 ALTER TABLE SC
2 DROP FOREIGN KEY fk_SC_Stu_Union;
```

10. 修改外键约束的行为 (ALTER TABLE with MODIFY) :

▼ Plain Text |

```
1 ALTER TABLE SC
2 MODIFY Sno CHAR(5) REFERENCES Stu_Union(sno) ON DELETE NO ACTION;
```

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。

修正了原来代码中的一些小问题

```

1 CREATE TABLE Stu_Union(
2   sno CHAR(5) NOT NULL UNIQUE,
3   sname CHAR(8),
4   ssex CHAR(1),
5   sage INT,
6   sdept CHAR(20),
7   CONSTRAINT PK_Stu_Union PRIMARY KEY(sno)
8 );
9
10
11 insert into Stu_Union values('10000','王敏','1',23,'cs');
12
13 UPDATE Stu_Union SET sno='' WHERE sdept='CS';
14
15 UPDATE Stu_Union SET sno='95002' WHERE sname='王敏';
16
17 select * from Stu_Union;

```

数据输出 消息 通知

	sno [PK] character (5)	sname character (8)	ssex character (1)	sage integer	sdept character (20)
1	95002	王敏	1	23	cs

2. 演示违反实体完整性的插入操作。

```

1 INSERT INTO Stu_Union VALUES ('95002', '李四', '0', 22, 'math');
2

```

错误: 键值"(sno)=(10000)" 已经存在重复键违反唯一约束"pk_stu_union"

错误: 重复键违反唯一约束"pk_stu_union"

SQL 状态: 23505

详细: 键值"(sno)=(10000)" 已经存在

3. 演示违反实体完整性的更新操作。

```

1 UPDATE Stu_Union SET sno='95002' WHERE sname='李四';

```

错误: 键值"(sno)=(95002)" 已经存在重复键违反唯一约束"pk_stu_union"

错误: 重复键违反唯一约束"pk_stu_union"

SQL 状态: 23505

详细: 键值"(sno)=(95002)" 已经存在

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。

```
1
2 create table Course(
3   cno char(4)NOT NULL UNIQUE,
4   cname varchar(50)NOT NULL,
5   cpoints int,
6   constraint PK primary key(cno)
7 );
8 insert into Course values('0001','ComputerNetworks',2);
9 insert into Course values('0002','Databsae',3);
```

数据输出 消息 通知

	sno [PK] character (5)	sname character (8)	ssex character (1)	sage integer	sdept character (20)
1	95002	王敏	1	23	cs
2	10000	李四	0	22	math
3	10001	李勇	0	24	EE

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。

```

1 CREATE TABLE SC(
2   Sno CHAR(5) REFERENCES Stu_Union(sno) on delete cascade,
3   Cno CHAR(4) REFERENCES Course(cno) on delete cascade,
4   grade INT,
5   CONSTRAINT PK_SC PRIMARY KEY(sno,cno)
6 );
7 insert into sc values('95002','0001',2);
8 insert into sc values('95002','0002',2);
9 insert into sc values('10001','0001',2);
10 insert into sc values('10001','0002',2);
11 Select * From SC;

```

数据输出 消息 通知

	sno [PK] character (5)	cno [PK] character (4)	grade integer
1	95002	0001	2
2	95002	0002	2
3	10001	0001	2
4	10001	0002	2

6. 演示违反参照完整性的插入数据。

```

1 INSERT INTO SC VALUES ('10003', '0003', 3);

```

数据输出 消息 通知

错误: 键值对(sno)=(10003)没有在表"stu_union"中出现.插入或更新表 "sc" 违反外键约束 "sc_sno_fkey"

错误: 插入或更新表 "sc" 违反外键约束 "sc_sno_fkey"

SQL 状态: 23503

详细: 键值对(sno)=(10003)没有在表"stu_union"中出现.

尝试插入的记录 ('10003', '0003', 3) 将违反参照完整性, 因为:

1. Sno '10003' 不存在于 Stu_Union 表中。
2. Cno '0003' 不存在于 Course 表中。

数据库将拒绝这个插入操作, 并返回一个错误, 因为它违反了外键约束。

7. 在 Stu_Union 中删除数据，演示级联删除。

Python

```
1 DELETE FROM Stu_Union WHERE sno = '95002';
```

sc表中的查询，删除前

数据输出 消息 通知

	sno [PK] character (5)	cno [PK] character (4)	grade integer
1	95002	0001	2
2	95002	0002	2
3	10001	0001	2
4	10001	0002	2

删除后

数据输出 消息 通知

	sno [PK] character (5)	cno [PK] character (4)	grade integer
1	10001	0001	2
2	10001	0002	2

执行上述删除操作后，SC 表中所有引用了这个 sno 的记录也会被自动删除。

8. 在Course 中删除数据，演示级联删除。

不仅删除了course表中数据，cs数据也删除了

Python

```
1 DELETE FROM Course WHERE cno = '0001';
```

sc表中的查询，删除前

数据输出 消息 通知			
	sno [PK] character (5)	cno [PK] character (4)	grade integer
1	10001	0001	2
2	10001	0002	2

删除后

数据输出 消息 通知			
	sno [PK] character (5)	cno [PK] character (4)	grade integer
1	10001	0002	2

执行上述删除操作后，SC 表中所有引用了这个 cno 的记录也会被自动删除。

自我实践

重新插入后

数据输出 消息 通知			
	sno [PK] character (5)	cno [PK] character (4)	grade integer
1	10001	0002	2
2	95002	0001	2
3	95002	0002	2
4	10001	0001	2

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。


```

1 ALTER TABLE SC
2 DROP CONSTRAINT sc_sno_fkey;
3
4 ALTER TABLE SC
5 ADD CONSTRAINT PK_SC PRIMARY KEY (Sno, Cno),
6 ADD CONSTRAINT fk_Stu_Union_sno FOREIGN KEY (Sno) REFERENCES Stu_Union(sno)
  ON DELETE NO ACTION,
7 ADD CONSTRAINT fk_Course_cno FOREIGN KEY (Cno) REFERENCES Course(cno) ON D
  ELETE NO ACTION;
8
9 DELETE FROM Stu_Union WHERE sno = '95002';
10

```

错误：键值对(sno)=(95002)仍然是从表"sc"引用的,在 "stu_union" 上的更新或删除操作违反了在 "sc" 上的外键约束 "fk_stu_union_sno"

错误：在 "stu_union" 上的更新或删除操作违反了在 "sc" 上的外键约束 "fk_stu_union_sno"

SQL 状态：23503

详细：键值对(sno)=(95002)仍然是从表"sc"引用的。

将 `on delete` 行为改为 `NO ACTION`。这意味着如果尝试删除 `Stu_Union` 或 `Course` 表中的记录，而这些记录被 `SC` 表中的外键引用，数据库将不允许删除操作，以防止违反参照完整性。

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。

```

1 drop constraint fk_Stu_Union_sno
2 drop constraint fk_Course_cno
3
4 ADD CONSTRAINT fk_SC_Stu_Union
5 FOREIGN KEY (Sno) REFERENCES Stu_Union(sno)
6 ON DELETE SET NULL;
7
8
9 ALTER TABLE SC
10 ADD CONSTRAINT fk_SC_Course
11 FOREIGN KEY (Cno) REFERENCES Course(cno)
12 ON DELETE SET NULL;

```

由于course表和student表中设置了sno和cno不能为null值

所以数据库将把 SC 表中对应的 Sno 字段设置为 NULL 的操作失败

数据输出 消息 通知

错误: 失败, 行包含(10001, null, 2).null value in column "cno" of relation "sc" violates not-null constraint

错误: null value in column "cno" of relation "sc" violates not-null constraint

SQL 状态: 23502

详细: 失败, 行包含(10001, null, 2).

上下文: SQL 语句 "UPDATE ONLY "public"."sc" SET "cno" = NULL WHERE \$1 OPERATOR(pg_catalog.=) "cno""

数据输出 消息 通知

错误: 失败, 行包含(null, 0001, 2).null value in column "sno" of relation "sc" violates not-null constraint

错误: null value in column "sno" of relation "sc" violates not-null constraint

SQL 状态: 23502

详细: 失败, 行包含(null, 0001, 2).

上下文: SQL 语句 "UPDATE ONLY "public"."sc" SET "sno" = NULL WHERE \$1 OPERATOR(pg_catalog.=) "sno""

实验心得

在进行数据库实验时, 我经常会遇到一些意想不到的错误, 这些错误初看之下似乎指向我操作的失误。例如, 当我尝试执行一个看似简单的删除操作时, 数据库却返回了一个违反外键约束的错误。一开始, 我可能会误以为是自己输入了错误的命令或者遗漏了某些步骤。

然而, 仔细阅读错误信息并回顾数据库的工作原理后, 我意识到这些错误实际上是数据库在严格执行参照完整性的结果。在设置了级联删除或设置为 NULL 的外键约束后, 数据库会自动处理相关联的记录, 以保持数据的一致性和完整性。这种机制在防止数据丢失和错误方面发挥了重要作用。

通过这些经历, 我学会了在遇到错误时, 不仅要检查自己的操作, 还要深入理解数据库的工作原理和约束规则。这种分析和思考的过程, 不仅帮助我解决了眼前的问题, 也加深了我对数据库设计和操作的理解。每次实验都是一个学习的机会, 即使是遇到错误和挑战, 也能从中获得宝贵的经验和知识。

