一. 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。

二. 实验内容

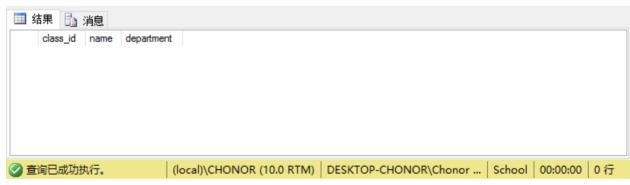
- 在数据库 school中建立表 Stu_Union,进行主键约束,在没有违反实体完整性的前提下插入并更新一条记录。
- 理解违反实体完整性的插入操作。
- 理解违反实体完整性的更新操作。
- 事务的处理,包括事务的建立、处理,以及出错时的事务回退。
- 通过建立 Scholarship表,插入数据,演示当与现有的数据环境不符时,无法建立实体完整性和参照完整性。

三. 实验结果

1. 在school数据库中建立一张新表class,包括class_id(varchar(4)), name(varchar(10)),department(varchar(20))三个列,并约束class_id为主键。

首先查询一下选修C++的学生总人数

```
CREATE TABLE class (class_id VARCHAR(4) NOT NULL UNIQUE,
name VARCHAR(10),
department VARCHAR(20)
CONSTRAINT PK_class PRIMARY KEY(class_id)
);
SELECT * FROM class;
```



2. 创建事务T3, 在事务中插入一个元组('00001','01CSC','CS'),并在T3中嵌套创建事务T4, T4也插入和T3一样的元组,编写代码测试,查看结果。

```
SET XACT_ABORT ON

BEGIN TRANSACTION T3

INSERT INTO CLASS VALUES ('0001','01CSC','CS');

SELECT * FROM CLASS;

SET XACT_ABORT ON

BEGIN TRANSACTION T4

INSERT INTO CLASS VALUES ('0001','01CSC','CS');

SELECT * FROM CLASS;

COMMIT TRANSACTION T4

COMMIT TRANSACTION T3
```



此时我们可以看到结果为:



但是实际上如果我们在使用:

```
SELECT * FROM CLASS;
```

此时:



我们是查询不到任何消息的因为此时 SET XACT_ABORT ON 的影响整个查询语句都回滚,所以我们需要修改为:

```
SET XACT_ABORT ON

BEGIN TRANSACTION T3

INSERT INTO CLASS VALUES ('0001','01CSC','CS');

SELECT * FROM CLASS;

SET XACT_ABORT OFF

BEGIN TRANSACTION T4

INSERT INTO CLASS VALUES ('0001','01CSC','CS');

SELECT * FROM CLASS;

COMMIT TRANSACTION T4

COMMIT TRANSACTION T3
```



此时因为SET XACT_ABORT OFF 所以T4的错误不会回滚,T3中的插入才会保留。

四. 实验感想

这次实验主要在于完整性的理解,实现起来很简单,主要就是在理解部分,在使用SQL的时候我们需要考虑到 其对于整个数据表的完整性的影响,如果破坏了完整性,那么此时的SQL语句不会成功执行。之后实验中还有事务 的使用以及事务的回滚的基本语法,理解了 SET XACT_ABORT ON的用法。