数据库系统课程实验报告

实验名称:	实验八 触发器
实验日期:	2022/5/5
实验地点:	四号楼
提交日期:	2022/5/6

学号:	陈奕培
姓名:	22920202202879
专业年级:	软工 2020 级
学年学期:	2021-2022 学年第二学期

1.实验目的

- 掌握数据库的特点（字体：华文仿宋，字号：四号，下同）

2.实验内容和步骤

（1）创建表

```
CREATE TABLE Teacher(
    ID CHAR(5) PRIMARY KEY,
    JOB VARCHAR(20) NOT NULL,
    SAL NUMBER(7,2)
);
```

```
test=# CREATE TABLE Teacher(
    ID CHAR(5) PRIMARY KEY,
    JOB VARCHAR(20) NOT NULL,
    SAL NUMBER(7,2)
);test(# test(# test(# test(#
NOTICE:  CREATE TABLE / PRIMAR
eacher"
CREATE TABLE
```

（2）插入数据

```
INSERT INTO Teacher VALUES
('10001', '教授', 3800),
('10002', '教授', 4100),
('10003', '副教授', 3500),
('10004', '助理教授', 3000);
```

```
test=# INSERT INTO Teacher VALUES
('10001', '教授', 3800),
('10002', '教授', 4100),
('10003', '副教授', 3500),
('10004', '助理教授', 3000);test-# test-# test-# test-#
INSERT 0 4
```

（3）添加触发器

①添加函数

```
CREATE OR REPLACE FUNCTION UPDATE_SAL()
RETURNS TRIGGER AS $$
DECLARE
BEGIN
    IF(NEW.SAL<4000)AND(NEW.JOB = '教授')
```

```
    THEN NEW.SAL := 4000;
    END IF;
    RETURN NEW;
END
$$ LANGUAGE PLPGSQL;
```

②添加触发器

```
CREATE TRIGGER INSERT_OR_UPDATE_SAL
BEFORE INSERT OR UPDATE ON Teacher
FOR EACH ROW
EXECUTE PROCEDURE UPDATE_SAL();
```

```
test=# CREATE TRIGGER INSERT_OR_UPDATE_SAL
BEFORE INSERT OR UPDATE ON Teacher
FOR EACH ROW
EXECUTE PROCEDURE UPDATE_SAL();test-# test-# test-#
CREATE TRIGGER
```

（4）验证触发器是否工作

```
INSERT INTO Teacher VALUES
('10005', '教授', 3999),
('10006', '教授', 4000);

UPDATE Teacher SET SAL=3900 WHERE ID = '10002';

SELECT * FROM Teacher;
```

```
test=# INSERT INTO Teacher VALUES
('10005', '教授', 3999),
('10006', '教授', 4000);test-# test-#
INSERT 0 2
test=# SELECT * FROM Teacher;
  id   |   job    |   sal
-------+----------+---------
 10001 | 教授     | 3800.00
 10002 | 教授     | 4100.00
 10003 | 副教授   | 3500.00
 10004 | 助理教授 | 3000.00
 10005 | 教授     | 4000.00
 10006 | 教授     | 4000.00
(6 rows)
```

插入检查，成功将工作改为 4000

```
test=# UPDATE Teacher SET SAL=3900 WHERE ID = '10002';
UPDATE 1
test=# SELECT * FROM Teacher;
  id   |   job    |   sal
-------+----------+---------
 10001 | 教授     | 3800.00
 10003 | 副教授   | 3500.00
 10004 | 助理教授 | 3000.00
 10005 | 教授     | 4000.00
 10006 | 教授     | 4000.00
 10002 | 教授     | 4000.00
(6 rows)
```

修改检查，成功将工资改为 4000

（5）完成教材内容

①建表

```
test=# CREATE TABLE SC_U(
    Sno CHAR(9),
    Cno CHAR(4),
    Oldgrade SMALLINT,
    Newgrade SMALLINT,
    FOREIGN KEY(Sno) REFERENCES Student(Sno),
    FOREIGN KEY(Cno) REFERENCES Course(CNO)
);test(# test(# test(# test(# test(# test(# test(#
CREATE TABLE
```

②创建触发器

```
CREATE OR REPLACE FUNCTION INSERT_SC_U()
RETURNS TRIGGER AS $$
DECLARE
BEGIN
    IF(NEW.Grade >= 1.1*OLD.Grade)
    THEN INSERT INTO SC_U VALUES(OLD.Sno,OLD.Cno,OLD.Grade,NEW.Grade);
    END IF;
    RETURN NEW;
END
$$ LANGUAGE PLPGSQL;
```

```
test=# CREATE OR REPLACE FUNCTION INSERT_SC_U()
RETURNS TRIGGER AS $$
DECLARE
BEGIN
    IF(NEW.Grade >= 1.1*OLD.Grade)
    THEN INSERT INTO SC_U VALUES(OLD.Sno,OLD.Cno,OLD.Grade,NEW.Grade);
    END IF;
    RETURN NEW;
END
$$ LANGUAGE PLPGSQL;test-# test$# test$# test$# test$# test$# test$# tes
CREATE FUNCTION
```

```
CREATE TRIGGER SC_AFTER_UPDATE
AFTER UPDATE ON SC
FOR EACH ROW
EXECUTE PROCEDURE INSERT_SC_U();
```

```
CREATE TRIGGER SC_AFTER_UPDATE
AFTER UPDATE ON SC
FOR EACH ROW
EXECUTE PROCEDURE INSERT_SC_U();tes
CREATE TRIGGER
```

③检验

```
UPDATE SC SET Grade=100 WHERE Sno='201215122' AND Cno='2';
UPDATE SC SET Grade=90 WHERE Sno='201215121' AND Cno='2';
SELECT * FROM SC_U;
```

```
test=# UPDATE SC SET Grade=100 WHERE Sno='201215122' AND Cno='2';
UPDATE 1
test=# SELECT * FROM SC_U;
    sno     | cno  | oldgrade | newgrade
------------+------+----------+----------
 201215122 | 2    |       90 |      100
(1 row)
```

```
test=# UPDATE SC SET Grade=90 WHERE Sno='201215121' AND Cno='2';
UPDATE 1
test=# SELECT * FROM SC_U;
    sno     | cno  | oldgrade | newgrade
------------+------+----------+----------
 201215122 | 2    |       90 |      100
(1 row)
```

第一条修改满足了条件，被记录下来，第二条 85->90 不满足 1.1，没被记录下来

（6）查看触发器

```
SELECT * FROM PG_TRIGGER;
```

```
test=# SELECT * FROM PG_TRIGGER;
 tgrelid |            tgname             | tgfoid | tgtype | tg
gargs | tgqual | tgowner
---------+-------------------------------+--------+--------+---
------+--------+---------
   16942 | insert_or_update_sal          |  16954 |     23 | 0
x     |        |      10
```

```
   17040 | RI_ConstraintTrigger_c_17051 |   1644 |      5 | 0        | t          |
x     |        |      10
   17040 | RI_ConstraintTrigger_c_17052 |   1645 |     17 | 0        | t          |
x     |        |      10
   17025 | sc_after_update              |  17053 |     17 | 0        | f          |
x     |        |      10
(22 rows)
```

（7）禁用触发器

①数据复原

```
DELETE FROM SC_U WHERE Newgrade=100;
UPDATE SC SET Grade=90 WHERE Sno='201215122' AND Cno='2';
UPDATE SC SET Grade=85 WHERE Sno='201215121' AND Cno='2';
```

```
test=# DELETE FROM SC_U WHERE Newgrade=100;
UPDATE SC SET Grade=90 WHERE Sno='201215122' AND Cno='2';
UPDATE SC SET Grade=85 WHERE Sno='201215121' AND Cno='2';DELETE 1
test=# UPDATE 1
```

②修改表

```
ALTER TABLE SC DISABLE TRIGGER SC_AFTER_UPDATE;
```

```
=> ALTER TABLE SC DISABLE TRIGGER SC_AFTER_UPDATE;
TER TABLE
```

重新修改数据，SC_U 中无内容

```
UPDATE SC SET Grade=100 WHERE Sno='201215122' AND Cno='2';
UPDATE SC SET Grade=90 WHERE Sno='201215121' AND Cno='2';
SELECT * FROM SC_U;
```

```
test=# SELECT * FROM SC_U;
 sno | cno | oldgrade | newgrade
-----+-----+----------+----------
(0 rows)
```

（8）删除触发器

```
DROP TRIGGER INSERT_OR_UPDATE_SAL ON Teacher;
DROP TRIGGER SC_AFTER_UPDATE ON SC;
```

```
test=# DROP TRIGGER SC_AFTER_UPDATE ON SC;
DROP TRIGGER
```

```
test=#
test=# DROP TRIGGER INSERT_OR_UPDATE_SAL ON Teacher;
DROP TRIGGER
```

思考：在对表的数据进行改动的时候，可以提前做出判断，避免错误的发生；在改动数据的时候，对该项改动进行记录；

3.实验总结

3.1 完成的工作

建表;

创建函数;

创建触发器;

对触发器进行测试;

3.2 对实验的认识

学会了触发器的使用;

3.3 遇到的困难及解决方法

无