数据库系统实验 4 实验报告

数据科学与计算机学院 计算机科学与技术 2016 级
王凯祺 16337233

2018 年 11 月 12 日

1 实验 4 触发器实验

1.1 after 触发器

在 takes 表上定义一个 *update* 触发器，当成绩更新后，自动修改 students 表中的 tot_cred，以保持数据一致性。

```
delimiter //
create trigger takes_cred_update after update on takes
for each row
begin
    if NEW.grade <> 'F' and NEW.grade is not null
        and (OLD.grade = 'F' or OLD.grade is null) then
        update student
        set tot_cred = tot_cred +
            (select credits
            from course
            where course.course_id = NEW.course_id)
        where student.ID = NEW.ID;
    end if;
    if OLD.grade <> 'F' and OLD.grade is not null
        and (NEW.grade = 'F' or NEW.grade is null) then
        update student
        set tot_cred = tot_cred -
            (select credits
            from course
            where course.course_id = OLD.course_id)
        where student.ID = OLD.ID;
    end if;
end;
```

在 takes 表上定义一个 *insert* 触发器，当选课记录插入后，自动修改 students 表中的 tot_cred，以保持数据一致性。

```
delimiter //
create trigger takes_cred_insert after insert on takes
for each row
begin
```

```
    if NEW.grade <> 'F' and NEW.grade is not null then
        update student
        set tot_cred = tot_cred +
            (select credits
            from course
            where course.course_id = NEW.course_id)
        where student.ID = NEW.ID;
    end if;
end;
```

在 takes 表上定义一个 *delete* 触发器，当选课记录删除后，自动修改 students 表中的 tot_cred ，以保持数据一致性。

```
delimiter //
create trigger takes_cred_delete after update on takes
for each row
begin
    if OLD.grade <> 'F' and OLD.grade is not null then
        update student
        set tot_cred = tot_cred -
            (select credits
            from course
            where course.course_id = OLD.course_id)
        where student.ID = OLD.ID;
    end if;
end;
```

验证触发器 takes_cred_update 。

```
select `grade`, `credits`
from takes natural join course
where ID = '1000' and
    course_id = '239' and
    sec_id = '1' and
    semester = 'Fall' and
    year = 2006;

select * from student where ID = 1000;
```

```
+-------+---------+
| grade | credits |
+-------+---------+
| C     |       4 |
+-------+---------+
1 row in set (0.03 sec)

+------+--------+------------+----------+
| ID   | name   | dept_name  | tot_cred |
+------+--------+------------+----------+
| 1000 | Manber | Civil Eng. |       39 |
+------+--------+------------+----------+
1 row in set (0.02 sec)
```

我们可以看到，学生 1000 已获学分 39，某门课的成绩为 C，学分为 4。我们将这门课的成绩修改为 F，观察触发器是否起作用。

```
update `takes`
set grade='F'
where ID = '1000' and
    course_id = '239' and
    sec_id = '1' and
    semester = 'Fall' and
    year = 2006;

select * from student where ID = 1000;
```

运行结果：

```
+------+--------+------------+----------+
| ID   | name   | dept_name  | tot_cred |
+------+--------+------------+----------+
| 1000 | Manber | Civil Eng. |       35 |
+------+--------+------------+----------+
1 row in set (0.04 sec)
```

在触发器的作用下，学号为 1000 的学分被修改。

1.2 before 触发器

在 takes 表上定义一个 *insert* 触发器，当选课记录插入之前，先检查 prereq 表中该课程的前驱课程是否全部已修并合格。

```
delimiter //
create trigger takes_prereq_insert before insert on takes
for each row
begin
    if exists(
        select 1
        from prereq
        where prereq.course_id = NEW.course_id and
            not exists (
                select 1
                from takes
                where prereq.prereq_id = takes.course_id and
                    NEW.ID = takes.ID and
                    takes.grade <> 'F' and
                    takes.grade is not null
            )
    ) then
        signal sqlstate '45001' set message_text = "Prerequisite␣course␣not␣learned.
            ";
    end if;
end;
```

验证触发器 insert 。

随便选一个 prereq 关系。

```
SELECT * FROM lab.prereq limit 1;
```

```
+-----------+-----------+
| course_id | prereq_id |
+-----------+-----------+
| 696       | 101       |
+-----------+-----------+
1 row in set (0.02 sec)
```

我们知道了 696 的前驱课程是 101 。

尝试添加一条记录:

```
insert into takes values ('1000', '696', '1', 'Spring', 2002, null);
```

结果如下:

```
ERROR 1644 (45001): Prerequisite course not learned.
```

由于 1000 未修读 101 课程，插入语句被拒绝。

尝试添加两条记录:

```
insert into takes values ('1000', '101', '1', 'Fall', 2009, 'B');
insert into takes values ('1000', '696', '1', 'Spring', 2002, null);
```

结果如下:

```
Query OK, 1 row affected (0.17 sec)
Query OK, 1 row affected (0.01 sec)
```

1.3 删除触发器

```
drop trigger takes_prereq_insert;
```

1.4 实验总结

触发器的设计其实就是设计 select 语句，当满足一定条件时做特定的事情。

触发器本身的原理不难，但是语法却每种 SQL 语言都不一样。用 Mysql 需要查阅 Mysql 手册才能得知如何执行 rollback ，如何写条件语句等……Mysql 没有 rollback 语句，但是可以 throw exception ，相当于中止执行。目前我还没试过 throw exception 之前执行过的语句是否会被 rollback ，目测是不行。