浙江大学

数据库系统实验报告

作业名称：SQL 数据完整性

姓　　名：谢集

学　　号：3220103501

电子邮箱：1436572990@qq.com

联系电话：13567793981

指导老师：孙建伶

2024 年 4 月 1 日

实验 3　SQL 数据完整性

一、实验目的

1. 熟悉通过 SQL 进行数据完整性控制的方法。

二、实验环境

操作系统：Windows11 22H2。

实验平台：MySQL。

三、实验流程

打开命令行，输入 `mysql -u sanaka -p`。输入密码，完成 Lab1 中所创建账号的登录。使用命令 `create database Lab3` 创建新数据库 Lab3.

我选择创建的数据库是关于部门管理的，包含两个表：

1、部门（departments）

- d_id: 部门的 ID，我将它设为主键。
- name: 部门的名字，我令它不能为 NULL 或空字符串。

2、雇员（employee）

- e_id: 雇员的 ID，我将它设为主键。
- name: 雇员的名字，我令它不能为 NULL 或空字符串。
- d_id: 雇员所属部门的 ID，我将它设为外键，引用部门表格中的信息。
- sex: 雇员的性别，我在它上面设置 check，检查其是否属于’F’或者’M’。

用 SQL 代码进行表格创建：

```
create table departments (
    d_id int primary key,
    name varchar(255) not null,
    check (name != '')
);

create table employees (
    e_id int primary key,
    name varchar(255) not null,
    sex varchar(1) not null,
    d_id int,
    foreign key (d_id) references departments(d_id)
        on delete set null
        on update cascade,
    check (name != ''),
    check (sex in ('F', 'M'))
);
```

这里我在外键上设置了约束 on delete set null 和 on update cascade，意图是：如果部门被删除，那么雇员的 d_id 会被设置成 NULL；如果部门被修改，那么在雇员表格上进行串联修改。

使用如下命令插入数据：

```
insert into departments (d_id, name) values (1, 'D1'), (2, 'D2'), (3, ''), (4, 'D4');
```

```
ERROR 3819 (HY000): Check constraint 'departments_chk_1' is violated.
```

这说明我们的 check 生效了，不能让 name 字段为空字符串。删除不规范数据后重新插入。

使用如下命令插入数据：

```
insert into employees (e_id, name, d_id, sex) value
    (1, 'sanaka', 'M', 1),  (2, 'cdj', 'm', 1),  (3, 'pxy', 'F', 2), (4, 'yzy', 'M', 2);
```

没有报错，可以发现数据库不区分大小写（即使是数据也一样？）。

```
mysql> insert into employees (e_id, name, sex, d_id) value
    ->  (5, 'sanaka', 's', 1)
    -> ;
ERROR 3819 (HY000): Check constraint 'employees_chk_2' is violated.
```

同样可以发现，我们的 check 很好地检查了性别的正确性。

使用下面命令插入数据：

```
insert into employees (e_id, name, sex, d_id) value
    (6, 'sana', 'f', 3);
```

```
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`lab2
`.`employees`, CONSTRAINT `employees_ibfk_1` FOREIGN KEY (`d_id`) REFERENCES `departments`
(`d_id`) ON DELETE SET NULL ON UPDATE CASCADE)
```

这说明外键设置生效了。由于我们没有在部门表格里插入 d_id 为 3 的记录（因为属性错误被删除了），所以这样的插入被 MySQL 拒绝了。

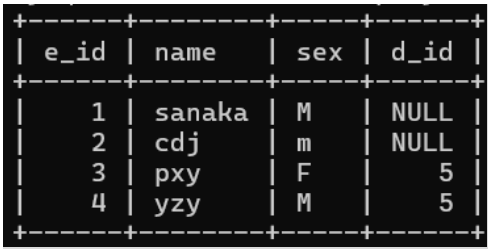
使用如下命令删除数据，观察 employees 的变化：

```
delete from departments where d_id = 1;
select * from employees;
```

```
+------+--------+-----+------+
| e_id | name   | sex | d_id |
+------+--------+-----+------+
|    1 | sanaka | M   | NULL |
|    2 | cdj    | m   | NULL |
|    3 | pxy    | F   |    2 |
|    4 | yzy    | M   |    2 |
+------+--------+-----+------+
```

可以发现，set NULL 实现了期望的效果，被删除的 d_id=1，让引用表中原本是 d_id=1 的记录都变成了 NULL。同样的：

```
update departments
    set d_id = 5 where d_id = 2;
select * from employees;
```



e_id	name	sex	d_id
1	sanaka	M	NULL
2	cdj	m	NULL
3	pxy	F	5
4	yzy	M	5

可以发现，cascade 也实现了期望的效果。我们让被引用表中的 d_id 从 2 变成 5，那么引用表（雇员）的 d_id 字段也发生了从 2 到 5 的变化。

四、遇到的问题及解决方法

本次实验相比而言任务量较小，因此我没有遇到太大的问题。唯一的问题就是使用 update 语句和 insert 语句还是有些不熟练，希望自己在下次实验中可以一次性写对这些基础的实验。

五、总结

通过本次实验，我深入了解了主键、外键和 check 约束在维护数据库完整性中的重要作用，尤其是外键的 on delete 和 on update 子句在处理复杂数据关系时的强大功能。这进一步增强了我对数据库实际应用的认识，为日后的课程学习奠定了坚实的基础。