Lab01 数据库 SQL 与过程化 SQL 实验

李远航 PB20000137

一、实验目的

掌握基本的sql的查询以及过程化sql的设计

二、实验环境

```
MySQL : 8.0.32-0ubuntu0.20.04.2 (Ubuntu)
```

三、实验过程和结果

1. 创建基本表

```
create table `lab1`.`Book`(
    `ID` char(8) not null,
    `name` varchar(10) not null,
    `author` varchar(10) default null,
    `price` float,
    `status` int default 0,
    `borrowTimes` int default 0,
    `reserveTimes` int default 0,
    primary key(`ID`),
    check(`status` in (0,1,2,3))
);

create table `lab1`.`Reader`(
    `ID` char(8) not null,
    `name` varchar(10) not null,
    `age` int,
    `addresss` varchar(20),
    primary key(`ID`)
);

create table `lab1`.`Borrow`(
    `bookID` char(8) not null,
    `readerID` char(8) not null,
    `borrowDate` date not null,
    `returnDate` date,
    primary key(`borrowDate`,`bookID`,`readerID`),
    foreign key(`bookID`) references `lab1`.`Book` (`ID`),
    foreign key(`readerID`) references `lab1`.`Reader` (`ID`)
);

create table `lab1`.`Reserve`(
    `bookID` char(8) not null,
    `readerID` char(8) not null,
    `reserveDate` date not null default (now()),
    `takeDate` date,
    primary key(`bookID`,`readerID`,`reserveDate`),
    foreign key(`bookID`) references `lab1`.`Book` (`ID`),
    foreign key(`readerID`) references `lab1`.`Reader` (`ID`),
```

```
    check (`reserveDate` < `takeDate`)
);
```

2. 查询

    1. 查询读者Rose借过的图书的图书号，书名和借期

    ```
    select bo.bookID, b.name, bo.borrowDate
    from `Book` b, `Reader` r, `Borrow` bo
    where b.ID = bo.bookID and r.ID = bo.readerID and r.name = 'Rose';
    ```

    ```
    +--------+----------------------+------------+
    | bookID | name                 | borrowDate |
    +--------+----------------------+------------+
    | b11    | 三体                 | 2022-01-11 |
    | b16    | 中国2185             | 2022-01-11 |
    | b1     | 数据库系统实现       | 2022-02-22 |
    | b2     | 数据库系统概念       | 2022-02-22 |
    | b19    | HowWeThink           | 2023-04-08 |
    +--------+----------------------+------------+
    ```

    2. 查询从没有借过图书也从没有预约过图书的读者号和读者姓名

    ```
    select ID, name from `Reader`
    where not exists (select * from `Borrow` where ID = `Borrow`.readerID)
        and not exists (select * from `Reserve` where ID =
    `Reserve`.readerID);
    ```

    ```
    +-----+-----------+
    | ID  | name      |
    +-----+-----------+
    | r10 | 汤大晨    |
    | r22 | 张悟      |
    +-----+-----------+
    ```

    3. 查询被借阅次数最多的读者

    ```
    select author from `Book`, `Borrow`
    where `Book`.ID = `Borrow`.bookID
    group by `Book`.author
    having count(*) = (
        select max(cn) from (
            select count(*) as cn from `Book`, `Borrow`
            where `Book`.ID = `Borrow`.bookID
            group by `Book`.author
        ) as newtable
    );
    ```

    ```
    +-----------+
    | author    |
    +-----------+
    | 刘慈欣    |
    +-----------+
    ```

    4. 查询目前借阅未还的书名中包含"MySQL"的图书号和书名

```
select `Book`.ID, `Book`.name from `Book`, `Borrow`
where `Book`.ID = `Borrow`.bookID
and `Borrow`.returnDate is null and `Book`.name like '%MySQL%';
```

```
+-----+------------+
| ID  | name       |
+-----+------------+
| b14 | Perl&MySQL |
+-----+------------+
```

5. 查询借阅图书数目超过10本的读者姓名

```
select `Reader`.name from `Reader`, `Borrow`
where `Reader`.ID = `Borrow`.readerID
group by `Reader`.ID
having count(*) >= 10;
```

```
+--------+
| name   |
+--------+
| 王林   |
| David  |
| 王林   |
+--------+
```

6. 查询没有借阅过任何一本John所著的图书的读者号和姓名

```
select r.ID, r.name from `Reader` r
where not exists(
    select * from `Book`, `Borrow`
    where `Borrow`.bookID = `Book`.ID
        and r.ID = `Borrow`.readerID and `Book`.author = 'John'
)
```

```
+-----+-----------+
| ID  | name      |
+-----+-----------+
| r1  | 王林      |
| r10 | 汤大晨    |
| r11 | 李平      |
| r12 | Lee       |
| r14 | Bob       |
| r15 | 李晓      |
| r17 | Mike      |
| r18 | 范维      |
| r19 | David     |
| r20 | Vipin     |
| r21 | 林立      |
| r22 | 张悟      |
| r23 | 袁平      |
| r4  | Mora      |
| r6  | 李一一    |
| r8  | 赵四      |
+-----+-----------+
```

7. 查询2022年借阅图书数目排名前10名的读者号、姓名以及借阅图书数

```sql
select r.ID, r.name, count(*) as cn from `Reader` r, `Borrow` b
where r.ID = b.readerID and year(b.borrowDate) = 2022 group by r.ID
order by cn desc
limit 10;
```

```
+-----+-----------+----+
| ID  | name      | cn |
+-----+-----------+----+
| r11 | 李平      |  4 |
| r3  | 罗永平    |  4 |
| r2  | Rose      |  4 |
| r23 | 袁平      |  3 |
| r1  | 王林      |  3 |
| r7  | 王二狗    |  3 |
| r8  | 赵四      |  3 |
| r9  | 魏心      |  3 |
| r13 | Jack      |  2 |
| r4  | Mora      |  2 |
+-----+-----------+----+
```

8. 创建一个读者借书信息的试图，该视图包含读者号、姓名、所借图书号、图书名和借期；并使用该视图查询最近一年所有读者的读书号以及所借阅的不同图书数

```sql
create view c_view(readerID, readername, bookID, bookname, borrowdate)
as
select `Reader`.ID, `Reader`.name, Book.ID, Book.name,
`Borrow`.borrowDate
from `Reader`, Book, `Borrow`
where `Reader`.ID = `Borrow`.readerID and Book.ID = `Borrow`.bookID;

select readerID, count(distinct bookID) from c_view
where borrowdate >= date_sub(curdate(), interval 1 year)
group by readerID;
```

```
+----------+------------------------+
| readerID | count(distinct bookID) |
+----------+------------------------+
| r11      |                      4 |
| r12      |                      1 |
| r13      |                      2 |
| r14      |                      2 |
| r15      |                      1 |
| r16      |                      1 |
| r17      |                      1 |
| r19      |                      1 |
| r2       |                      1 |
| r23      |                      3 |
| r4       |                      1 |
| r5       |                      3 |
| r6       |                      2 |
| r9       |                      2 |
+----------+------------------------+
```

3. 设计一个存储过程updateReaderID，实现对读者表的ID的修改

   考虑oldid不存在和新id已存在的情况

```
create procedure updateReaderID(in oldid char(8), in newid char(8), out ret
int)
begin
    set ret = 0;
    start transaction;
    if exists(select * from `Reader` where ID = oldid)
    and not exists(select * from Reader where ID = newid) then
        set FOREIGN_KEY_CHECKS = 0;
        update `Reader` set ID = newid where ID = oldid;
        update `Borrow` set readerID = newid where readerID = oldid;
        update `Reserve` set readerID = newid where readerID = oldid;
        set FOREIGN_KEY_CHECKS = 1;
        select 'Success!';
        commit;
    else
        set ret = -1;
        select 'Fail!';
        rollback;
    end if;
end
```

4. 设计一个存储过程borrowBook，当读者借书时调用该存储过程完成结束处理

   一步步讨论读者是否重复借书，是否存在预约记录，读者同天借书数目是否超过3本，等等

```
create procedure borrowBook(in rdid char(8), in boid char(8), out ret int)
label:begin
    declare tmp date;
    declare respeople char(8);
    declare books int;
    set ret = 0;
    start transaction;

    if not exists(select * from Reader where ID = rdid)
    or not exists(select * from Book where ID = boid) then
        set ret = -1;
        select 'not exists reader or book!'
        rollback;
        leave label;
    end if;

    select borrowDate into tmp from `Reader`,`Borrow`
    where `Reader`.ID = `Borrow`.readerID and `Reader`.ID = rdid
            and `Borrow`.borrowDate = Date(now());
    if tmp is not null then
        set ret = -1;
        select 'borrow too many times one day!'
        rollback;
        leave label;
    end if;

    if exists(select * from Book where ID = boid and status = 1) then
        set ret = -1;
        select 'can not borrow!'
        rollback;
        leave label;
```

```
    end if;

    if exists(select * from `Reserve` where bookID = boid and readerID != rdid) and
    not exists(select * from `Reserve` where bookID = boid and readerID = rdid) then
        set ret = -1;
        select 'have not reserve!'
        rollback;
        leave label;
    end if;

    select count(*) into books from `Borrow`
    where readerID = rdid and returnDate is null group by readerID;
    if books >= 3 then
        set ret = -1;
        select 'borrow too many books!'
        rollback;
        leave label;
    end if;

    select readerID into respeople from `Reserve` where bookID = boid and readerID = rdid;
    if respeople is null then
        insert into Borrow value(boid, rdid, Date(now()), NULL);
        update `Book` set status = 1, borrowTimes = borrowTimes + 1 where ID = boid;
        select 'Success!';
    else
        insert into Borrow value(boid, rdid, Date(now()), NULL);
        update `Book` set status = 1, borrowTimes = borrowTimes + 1 where ID = boid;
        delete from `Reserve` where bookID = boid and readerID = rdid;
        select 'Success!';
    end if;
    commit;
end
```

5. 设计一个存储过程 returnBook，当读者还书时调用该存储过程完成还书处理

   考虑是否存在可还书的记录

```
create procedure returnbook(in rdid char(8), in boid char(8), out ret int)
begin
    set ret = 0;
    start transaction;

    if not exists(select * from `Borrow` where boid = bookID and rdid = readerID and returnDate is null) then
        set ret = -1;
        select 'Error!';
        rollback;

    else
        update `Book` set status = IF(exists(select * from Reserve where bookID = boid),2,0) where ID = boid;
```

```
13          update `Borrow` set returnDate = Date(now()) where bookID = boid and
    readerID = rdid and returnDate is null;
14          select 'Success';
15          commit;
16      end if;
17  end
```

6. 设计一个触发器，实现：当一本书被预约时, 自动将 Book 表中相应图书的 status 修改为 2，并增加 reserve_Times；当某本预约的书被借出时或者读者取 消预约时, 自动减少 reserve_Times

   考虑还书后这本书是否被预约来更新status

```
1   create trigger updateRe after insert on `Reserve` for each row
2   begin
3       update `Book` set status = if(status = 1, 1, 2), reserveTimes =
    reserveTimes + 1 where ID = new.bookID;
4   end
5
6   create trigger updateBo after delete on `Reserve` for each row
7   begin
8       update `Book` set reserveTimes = reserveTimes - 1 where ID = old.bookID;
9   end
```

四、实验总结与思考

1. 对mysql的建表查询有了更深刻的理解
2. 对存储过程和触发器的设计有了进一步的认识