SQL & PL/SQL 实验报告

PB15111662 李双利

一、 实验目的

1. 学会使用 oracle 进行表的建立、更改和删除。
2. 掌握用 PL/SQL 实现从数据表中进行级联查询、嵌套查询等较为复杂的查询操作。
3. 理解 PL/SQL 的存储过程和触发器，并进行实现。

二、 实验要求

（1）新建三个基本表

a）Book（ID: char(8)，name:varchar2(10)，author:varchar2(10)，price:float，status: int） 图书号 ID 为主键，书名不能为空。状态（status）为 1 表示书被借出，0 表示在馆， 默认值为 0。

b）Reader（ID:char(8)，name:varchar2(10)，age:int，address:varchar2(20)）读者号 ID 为主键。

c）Borrow（book_ID:char(8)，Reader_ID:char(8)，Brrrow_Date:date，Return_Date:date） 其中：还期 Return_Date 为 NULL 表示该书未还。主键为（图书号，读者号），图 书号为外键，引用图书表的图书号，读者号为外键，引用读者表的读者号

（2）完整性验证

插入测试数据，并且设计例子，验证实体完整性、参照完整性、用户自定义完整性。

（3）实现以下查询操作

a）检索读者 Rose 的读者号和地址；

b）检索读者 Rose 所借阅读书（包括已还和未还图书）的图书名和借期；

c）检索未借阅图书的读者姓名；

d）检索 Ullman 所写的书的书名和单价；

e）检索读者“李林”借阅未还的图书的图书号和书名；

f）检索借阅图书数目超过 3 本的读者姓名；

g）检索没有借阅读者“李林”所借的任何一本书的读者姓名和读者号；

h）检索书名中包含“Oracle”的图书书名及图书号；

i）创建一个读者借书信息的视图，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者号以及所借阅的不同图书数；

（4）设计存储过程

实现对 Book 表的 ID 的修改。

（5）设计触发器

实现当一本书被借出时，自动将 Book 表中相应图书的 status 修改为 1；当某本书被归还时，自动将 status 改为 0。

三、 实验内容和过程

（1） 表的建立和数据插入

用 sql 的 create 语句分别建立三个表即可，比如 Book 的建立如下：

```
create table Book(
    ID char(8) Constraint PK_B Primary Key,
    name varchar2(10) Not null,
    author varchar2(10),
    price float,
    status int default 0
);
```

需要注意的是每个表的约束要求，指定约束名可以方便数据库的管理。对于 Borrow 表而言，其两个属性值 Book_ID 和 Reader_ID 需要与 Book 表和 Reader 表建立外码的约束关系。

在建好的三个表中实现插入过程，用 sql 的 insert 语句即可，比如：
insert into Reader values('RMA099', '李林', 21, '中国');
其他数据的插入改变 values 和表名即可。插入多条数据后，得到的三个数据表如下：

ID	NAME	AGE	ADDRESS
RMA001	Navas	32	哥斯达黎加
RMA020	Asensio	22	西班牙
RMA004	Ramos	32	西班牙
RMA007	Ronaldo	33	葡萄牙
RMA009	Benzema	31	法国
RMA011	Bale	32	威尔士
RMA008	Kroos	28	德国
RMA010	Modric	33	克罗地亚
RMA098	Rose	24	英国
RMA099	李林	21	中国

```
SQL> select * from Book;

ID       NAME       AUTHOR          PRICE     STATUS
-------- ---------- ---------- ---------- ----------
BK001    白夜行     东野圭吾           35          1
BK002    恶意       东野圭吾           28          1
BK003    Oracle 1   Ullman             39          0
BK004    Oracle 2   Ullman             39          0
BK005    Oracle 3   Ullman             39          1
BK006    Mysql      Ullman             41          0
BK007    ML         Murphy             64          1
BK008    DL         GoodFellow        118          1
```

```
SQL> select * from Borrow;

BOOK_ID  READER_I BRRROW_DATE    RETURN_DATE
-------- -------- -------------- --------------
BK001    RMA098   02-1月 -18     02-3月 -18
BK002    RMA098   05-2月 -18
BK006    RMA098   22-1月 -18     08-4月 -18
BK007    RMA099   15-2月 -18
BK008    RMA099   16-2月 -18
BK005    RMA099   17-2月 -18
BK001    RMA099   04-3月 -18     02-4月 -18
BK003    RMA007   15-8月 -17     02-1月 -18
BK004    RMA007   18-3月 -17     02-1月 -18
BK005    RMA007   24-5月 -17     02-1月 -18
BK006    RMA007   15-3月 -17     02-1月 -18

BOOK_ID  READER_I BRRROW_DATE    RETURN_DATE
-------- -------- -------------- --------------
BK007    RMA007   04-1月 -17     02-1月 -18
BK001    RMA008   19-4月 -17     22-11月-17
BK002    RMA010   28-7月 -17     12-12月-18
BK001    RMA011   06-4月 -18
BK002    RMA011   04-2月 -17     16-3月 -17
BK004    RMA011   15-1月 -17     12-2月 -17
BK005    RMA011   14-2月 -17     10-4月 -18
```

（2） 完整性的验证过程

a） 实体完整性

根据实体完整性质，主码不能为空且唯一，所以设计如下测试：

insert into Reader values('RMA098', 'Rose1', 20, 'England');

（主码'RMA098’已经存在表 Reader 中）

得到报错结果：

```
begin
  insert into Reader values('RMA098', 'Rose1', 20, 'England');
end;
错误报告 -
ORA-00001: 违反唯一约束条件 (SYSTEM.PK)
ORA-06512: 在 line 2
```

b） 参照完整性

由参照完整性，参照关系 R 的任一个外码值必须 等于被参照关系 S 中所参照的候选码的某个值。所以设计如下测试：

```
update Reader
set id = 'RMA096'
where id = 'RMA098' ;
```

得到报错结果：

```
begin
  update Reader
  set id = 'RMA096'
  where id = 'RMA098' ;
end;
错误报告 -
ORA-02292: 违反完整约束条件 (SYSTEM.FK_RI) - 已找到子记录
ORA-06512: 在 line 2
```

c） 用户自定义完整性

根据 Book 中用户自定义的一个约束为“书名不能为空”，所以尝试插

入一条书名为空的记录进行测试：

insert into Book values('BK099', '', 'Tom', 22, 0);

得到报错结果：

```
begin
  insert into Book values('BK099', '', 'Tom', 22, 0);
end;
错误报告 -
ORA-01400: 无法将 NULL 插入 ("SYSTEM"."BOOK"."NAME")
ORA-06512: 在 line 2
```

（3） 查询操作的实现过程

a） 检索读者 Rose 的读者号和地址；

新建一个变量 s1，直接选择 name 为’Rose’的记录即可。

```
select ID, address into s1 from Reader where name = 'Rose';
```

结果为：

```
(1)检索读者Rose 的读者号和地址: RMA098  英国
```

b） 检索读者 Rose 所借阅读书（包括已还和未还图书）的图书名和借期；

新建一个 s2 游标存放选择结果，对 Book 和 Borrow 进行级联查询，并且选择其中读者 ID 为 Rose 的 ID 的记录。

```
cursor s2 is select Book.name, Borrow.brrrow_date
             from Book, Borrow
             where Book.ID = Borrow.book_ID and Borrow.Reader_ID = s1.id;
```

结果为：

```
(2)Rose所借的图书的图书名和借期：
-------------------------
白夜行  02-1月 -18
恶意    05-2月 -18
Mysql   22-1月 -18
-------------------------
```

c） 检索未借阅图书的读者姓名；

新建一个 s3 游标存放选择结果，首先找到所有借阅过图书的读者 ID，然后从 reader 表里查找 id 不在其中的记录。

```
cursor s3 is select name
             from Reader
             where id not in (select distinct Reader_ID from Borrow);
```

结果为：

```
(3)未借阅图书的读者姓名：
-------------------------
Navas
Benzema
Asensio
Ramos
-------------------------
```

d） 检索 Ullman 所写的书的书名和单价；

新建一个 s4 游标存放选择结果，直接从 Book 表找查找 author 为’Ullman 的’记录即可。

```
cursor s4 is select name, price
             from Book
             where author = 'Ullman';
```

结果为：

```
（4）Ullman所写的书的书名和单价：
-------------------------
Oracle 1  39
Oracle 2  39
Oracle 3  39
Mysql     41
-------------------------
```

e） 检索读者“李林”借阅未还的图书的图书号和书名；
新建一个 s5 游标存放选择结果，由于涉及到读者姓名、图书名和借阅记录，所以考虑对三个表进行级联查询，连接后选择 name 为’李林’并且返回日期为空的记录。

```
cursor s5 is select book.id, book.name
             from Book, Reader, Borrow
             where Reader.name = '李林' and Reader.id = Borrow.Reader_ID
             and Borrow.Book_ID = Book.id and Borrow.return_date is null;
```

结果为：

```
（5）李林借阅未还的图书的图书号和书名：
-------------------------
BK007     ML
BK008     DL
BK005     Oracle 3
-------------------------
```

f） 检索借阅图书数目超过 3 本的读者姓名；
新建一个 s6 游标存放选择结果，首先从 Borrow 中根据读者号进行分组，然后统计每个读者号对应的记录数，这样找到了读者号和借阅数的子表，然后通过嵌套查询，从其中选出借阅数大于 3 的记录。

```
cursor s6 is select name
             from Reader, (select Reader_ID, count(Book_ID) as count_book
                           from Borrow
                           group by Reader_ID) Borrow2
             where Reader.id = Borrow2.Reader_ID and Borrow2.count_book > 3;
```

结果为：

```
（6）借阅图书数目超过 3 本的读者姓名：
-------------------------
Ronaldo
李林
Bale
-------------------------
```

g） 检索没有借阅读者“李林”所借的任何一本书的读者姓名和读者号；
新建一个 s7 游标存放选择结果，考虑到直接查找逻辑较为复杂，所以先查找借阅过李林所借的任何一本书的读者，然后再用集合的差运算，即可选出要求的记录。

```
cursor s7 is (select name, id from Reader)
             minus
            (select distinct Reader.name as name, Reader.id as id
             from (select distinct Borrow.Book_ID as id
                   from Borrow, Reader
                   where Borrow.Reader_ID = Reader.id and Reader.name = '李林') BookID,
                   Borrow,Reader
             where Borrow.Book_ID in BookID.id and Borrow.Reader_ID = Reader.id);
```

结果为：

```
（7）没有借阅读者"李林"所借的任何一本书的读者姓名和读者号：
-------------------------
RMA020    Asensio
RMA009    Benzema
RMA010    Modric
RMA001    Navas
RMA004    Ramos
-------------------------
```

h）检索书名中包含“Oracle”的图书书名及图书号；

新建一个 s8 游标存放选择结果，用 like 进行相似性的匹配查询即可。

```
cursor s8 is select id, name
             from Book
             where name like '%Oracle%';
```

结果为：

```
（8）书名中包含“Oracle”的图书书名及图书号：
-------------------------
BK003     Oracle 1
BK004     Oracle 2
BK005     Oracle 3
-------------------------
```

i） 创建一个读者借书信息的视图，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者号以及所借阅的不同图书数；

在新建视图后，选择最近一年内借阅记录，所以用当前日期减去借阅日期，然后判断该值是否小于 365，即可得到一年内的借阅记录，再进行分组计数即可。

```
for c1 in(select Reader_id, count(distinct Book_id) as br_num
          from borrow_info
          where sysdate - Borrow_date < 365 group by Reader_id) loop
  DBMS_OUTPUT.PUT_LINE(rpad(c1.Reader_id, 10) || c1.br_num);
end loop;
```

结果为：

```
（9）最近一年所有读者的读者号以及所借阅的不同图书数：
-------------------------
RMA010    1
RMA007    2
RMA099    4
RMA011    1
RMA098    3
-------------------------
```

（4） 存储过程

需要实现对 Book 里的图书号进行更新，但是由于 Book 和 Borrow 里的

读书号存在外码的联系，所以同时要对 Borrow 的图书号也进行更新。但是由于存在参照完整性，所以不能直接在一个存储过程里两个表的更新，这是因为每次更新一个表之后会立刻检查约束关系，不满足参照完整性就会保存。我想到了两种解决方案：

1. 先把要更新后的 Book 新纪录插入到 Book 表中，然后在 Borrow 表里直接进行更新就行了（因为要更新的 id 在 book 表中可以找到），最后删除原来 book_id 的那条记录即可。
2. 还有一种简单的方案就是将其约束修改为延迟检查，即在 update Book 之后不立刻检查是否满足约束，这样就不会报错，在更新完两个表之后恢复其约束为立刻检查即可。

（5） 触发器设计

根据实验要求，每次插入一条借阅记录，要将 book 里的对应 status 改为 1，如果归还的话更改为 0。所以设计一个触发器对两种行为的发生进行触发操作，每次触发后对 Book 和 Borrow 进行级联查询，找到 Book 的 status 为 0 并且在 Borrow 里的 return_date 为空的记录（说明这是一条新插入的借阅记录），更新 status 为 1 即可。

同样的，查找 Book 的 status 为 1 并且在 Borrow 里的 return_date 不为空的记录（说明这是一本刚被归还还未修改 status 的书），更新 status 为 0 即可。

四、 实验总结

此次实验，首先是完整的配置了 oracle 的开发环境以及 PL/SQL 的 IDE，然后用 PL/SQL 语言进行了数据库的建表，查询，插入和更新等基本操作，并且还实现了存储过程和触发器的设计。通过此次实验，我对 SQL 的语法更加熟悉，对数据库的原理有了更深入的理解，同时，在实验过程中进行的“级联更新”（oracle 自身不支持）操作也让我学会了对于数据库的设计过程要从多个角度理解。