

Lab 01 数据库 SQL 与过程化 SQL 实验

设某图书馆数据库包含下面的基本表：

- **图书表 Book**(bid:char(8), bname:varchar(100), author:varchar(50), price:float, bstatus:int, borrow_Times:int, reserve_Times:int). 其中:
 - A. 图书号 bid 为主键;
 - B. 书名 bname 不能为空;
 - C. 状态 bstatus 为 0 表示可借, 1 表示书被借出, 2 表示已被预约, 默认值为 0;
 - D. borrow_Times 表示图书有史以来的总借阅次数, reserve_Times 表示图书当前的预约人数, 默认值都为 0;
 - E. 规定一本书只能被一个人借阅, 但可以被多个人预约。
- **读者表 Reader**(rid:char(8), rname:varchar(20), age:int, address:varchar(100)).
读者号 rid 为主键。
- **借阅表 Borrow**(book_ID:char(8), reader_ID:char(8), borrow_Date:date, return_Date:date). 其中:
 - A. 还期 return_Date 为 NULL 表示该书未还;
 - B. 主键为 (图书号book_ID, 读者号reader_ID, 借阅日期borrow_Date);
 - C. 图书号book_ID为外键, 引用图书表的图书号;
 - D. 读者号reader_ID为外键, 引用读者表的读者号。
- **预约表 Reserve**(book_ID:char(8), reader_ID:char(8), reserve_Date:date, take_Date:date).
 - A. 其中主键为(图书号book_ID, 读者号reader_ID, 预约日期reserve_Date);
 - B. reserve_Date 默认为当前日期;
 - C. take_Date 为预约取书的日期且要求晚于 reserve_Date。

1、 创建上述基本表, 并插入给定测试样例;

2、 用 SQL 语言完成下面小题, 并测试运行结果:

(1) 查询读者 Rose 借过的书 (包括已还和未还) 的图书号、书名和借期;

(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;

(3) 查询被借阅次数最多的作者 (注意一个作者可能写了多本书);

(使用两种方法: A. 使用借阅表 borrow 中的借书记录;

B. 使用图书表 book 中的 borrow_times)

(思考: 哪种方法更好?)

(4) 查询目前借阅未还的书名中包含 “MySQL” 的图书号和书名;

(5) 查询借阅图书数目超过 3 本的读者姓名;

(6) 查询没有借阅过任何一本 J.K. Rowling 所著的图书的读者号和姓名;

(7) 查询 2024 年借阅图书数目排名前 3 名的读者号、姓名以及借阅图书数;

(8) 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 (对于没有借过图书的读者, 是否包含在该视图中均可); 并使用该视图查询2024年所有读者的读者号以及所借阅的不同图书数;

3、设计一个存储过程 `updateReaderID`, 实现对读者表的 ID 的修改
(本题要求不得使用外键定义时的 `on update cascade` 选项, 因为该选项不是所有 DBMS 都支持)。

使用该存储过程: 将读者ID中 ‘R006’ 改为 ‘R999’ 。

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借书处理。要求:

A. 一个读者最多只能借阅 3 本图书, 意味着如果读者已经借阅了 3 本图书并且未归还则不允许再借书;

B. 同一天不允许同一个读者重复借阅同一本读书;

C. 如果该图书存在预约记录, 而当前借阅者没有预约, 则不许借阅;

(思考: 在实现时, 处理借书请求的效率是否和 A、B、C 的实现顺序有关系?)

D. 如果借阅者已经预约了该图书, 则允许借阅, 但要求借阅完成后删除借阅者对该图书的预约记录;

E. 借阅成功后图书表中的 `times` 加 1;

F. 借阅成功后修改 `bstatus`。

使用该存储过程处理:

(1) ID为 ‘R001’ 的读者借阅ID为 ‘B008’ 的书的请求 (未预约), 显示借阅失败信息。

(2) ID为 ‘R001’ 的读者借阅ID为 ‘B001’ 的书的请求 (已预约), 显示借阅成功信息, 并展示预约表相关预约记录被删除, 以及图书表对应书籍的`borrow_times`和`bstatus`属性的变化。

(3) ID为 ‘R001’ 的读者再次借阅ID为 ‘B001’ 的书的请求 (同一天已经借阅过), 显示借阅失败信息。

(4) ID为 ‘R005’ 的读者借阅ID为 ‘B008’ 的书的请求 (已借三本书未还), 显示借阅失败信息。

(以上借阅日期默认为 ‘2024-03-28’)

5、参考 4，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。要求：

- A. 还书后补上借阅表 borrow 中对应记录的 return_date;
- B. 还书后将图书表 book 中对应记录的 bstatus 修改为 0（没有其他预约）或 2（有其他预约）。

使用该存储过程处理：

- (1) ID为‘R001’的读者归还ID为‘B008’的书的请求（未借阅），并展示还书失败信息。
 - (2) ID为‘R001’的读者归还ID为‘B001’的书的请求，并展示相关书籍在book表中的bstatus以及在borrow表中的return_date的变化。
- （以上还书日期默认为‘2024-03-29’）

6、设计触发器，实现：

- A. 当一本书被预约时，自动将图书表 book 中相应图书的 bstatus 修改为 2，并增加 reserve_Times;
- B. 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;
- C. 当某本书的最后一位预约者取消预约且该书未被借出（修改前 bstatus 为 2）时，将 bstatus 改为 0。

定义并创建该触发器，然后处理：

- (1) ID为‘R001’的读者预约ID为‘B012’的书，再取消预约的请求，展示过程中reserve_Times 和 bstatus 的变化。