图书管理系统数据库设计

### 摘要

图书借阅管理系统通过精确的图书信息管理和借阅记录追踪，确保图书的借阅与归还流程高效有序。系统实现了对图书借阅、归还、库存等环节的全面监控，帮助管理员及时掌握图书库存状态，缩短信息流转时间，确保图书管理的准确性和高效性。系统支持借阅人对图书的查询、借阅与归还操作，同时具备智能预警功能，自动提醒借阅人超期未还书籍，并记录借还情况，以避免图书的损坏和遗失。系统有效降低了人工管理成本，提高了图书馆或书店的运营效率。

当前，随着信息化管理的普及，图书馆和书店等单位也逐渐实施了信息化系统，利用自动化手段提升工作效率，减少人为操作错误。图书借阅管理系统应运而生，针对借阅人、图书及借阅记录等信息进行精细管理，在实现信息共享和查询的同时，也保证了系统的高效性与可靠性。

本课程设计报告展示了图书借阅管理系统的数据库设计与实现过程。设计分为概念模型、逻辑设计和物理设计三个阶段。在概念模型阶段，我们通过 E-R 图分析图书借阅管理需求，并在逻辑设计阶段将其转化为关系模式，设计了包括图书、借阅人、借阅记录等模块的数据库表结构。在物理设计阶段，遵循命名规则并查询表结构，完成了数据表的设计与优化，应用了触发器、视图、存储过程和存储函数等功能。本次设计加深了我对数据库设计理论的理解，并提升了实际操作能力。

该系统为图书馆和书店提供了高效的信息化管理解决方案，帮助管理员更好地管理图书资源，减少人工错误，提升管理效率，增强了系统的可维护性和扩展性。

**关键字**：图书管理、数据库设计、E-R图、触发器、存储过程

### Abstract

The Library Management System ensures the efficient and orderly process of book borrowing and returning through precise book information management and borrowing record tracking. The system implements comprehensive monitoring of book borrowing, returning, and inventory processes, helping administrators keep track of book stock levels in real-time, reduce information flow time, and ensure the accuracy and efficiency of book management. The system supports borrowers in querying, borrowing, and returning books, while also featuring an intelligent warning function that automatically alerts borrowers about overdue books and records the borrowing and returning history to prevent book damage or loss. The system effectively reduces manual management costs and improves the operational efficiency of libraries or bookstores.

With the widespread adoption of information management, libraries and bookstores are gradually implementing automated systems to enhance work efficiency and minimize human errors. The Library Management System has emerged to provide detailed management of borrower information, books, and borrowing records, ensuring efficient information sharing and query capabilities while also guaranteeing the system’s performance and reliability.

This course design report presents the database design and implementation process for the Library Management System. The design is divided into three stages: conceptual model, logical design, and physical design. In the conceptual model stage, we analyzed the management requirements through an E-R diagram, and in the logical design stage, the E-R diagram was transformed into a relational model. Database table structures for key modules such as books, borrowers, and borrowing records were designed. In the physical design stage, naming conventions were followed and table structures were optimized. We applied triggers, views, stored procedures, and stored functions to enhance the system's efficiency. This design has deepened my understanding of database theory and improved my practical skills in database management.

This system provides libraries and bookstores with an efficient information management solution, helping administrators better manage book resources, reduce human errors, improve management efficiency, and enhance the system's maintainability and scalability.

**Keywords:** Library Management, Database Design, E-R Diagram, Triggers, Stored Procedures

# 一、系统需求分析

（一）需求概述

本次课程设计的题目是“图书借阅管理系统”。该系统旨在高效管理图书的借阅与归还，具体要求如下：所有书籍均由唯一的书号进行标识，书籍信息包括书号、书名、作者、出版社、出版日期等属性。用户可以借阅多种书籍，而同一本书籍也可被多个用户借阅，因此借书证号需具备唯一性，以便准确记录每位用户的借阅信息。此外，每个出版社可以出版多种书籍，但同一本书籍仅由一个出版社出版，出版社名称也需具备唯一性。

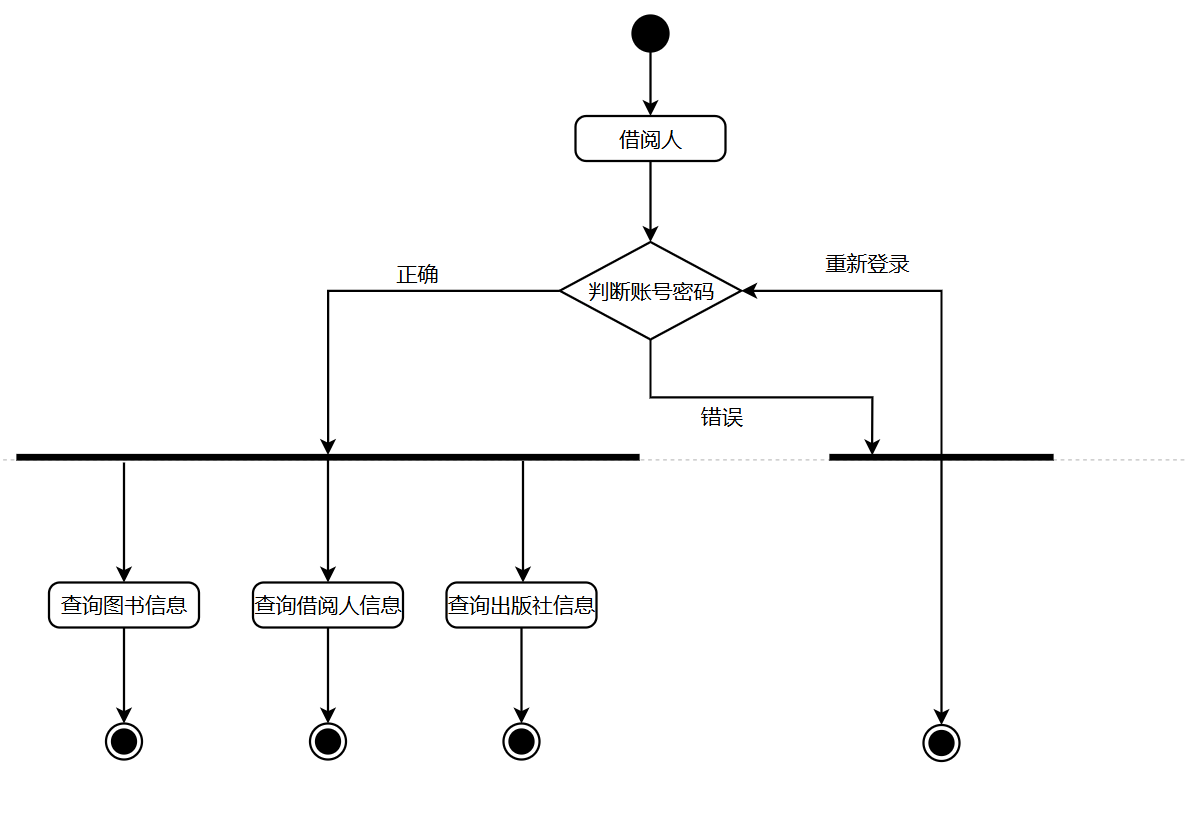
系统的使用者分为管理人员和借书人。他们对该数据库系统有不同的操作：

管理人员：可以随时查询书库中现有书籍的种类、数量及存放位置。此外，他们还可以实时查看书籍的借还情况，包括借阅人的单位、姓名、借书证号、借书日期和还书日期。当需要增购相关书籍时，管理员能够通过数据库中保存的出版社信息，如电报编号、电话、邮政编码及地址等，联系相应的出版社。总之，管理人员具备对数据库系统进行增加、删除、更新和查询的全面权限，以确保图书借阅管理的高效性和准确性。

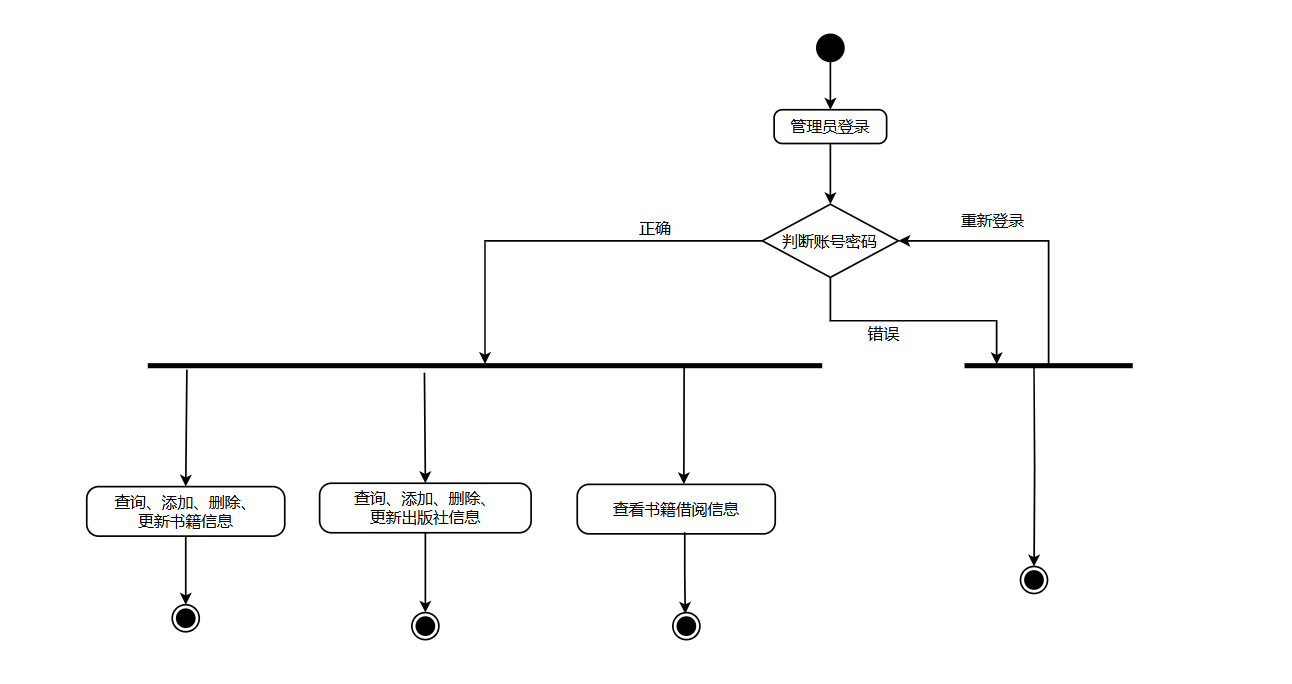
借阅人：借阅人也可以随时查询书库中现有书籍的品种、数量与存放位置。同时通过系统查询书籍的借还情况。当借阅人需要借阅书籍时，数据库将显示相关数据的信息，同时借阅人修改图书馆保存的图书信息，更新自己的借阅信息。学生在归还图书时，应根据书的编码修改借阅信息。总之，借阅人可以对数据库进行查询、修改、删除等操作。

（二）业务流分析

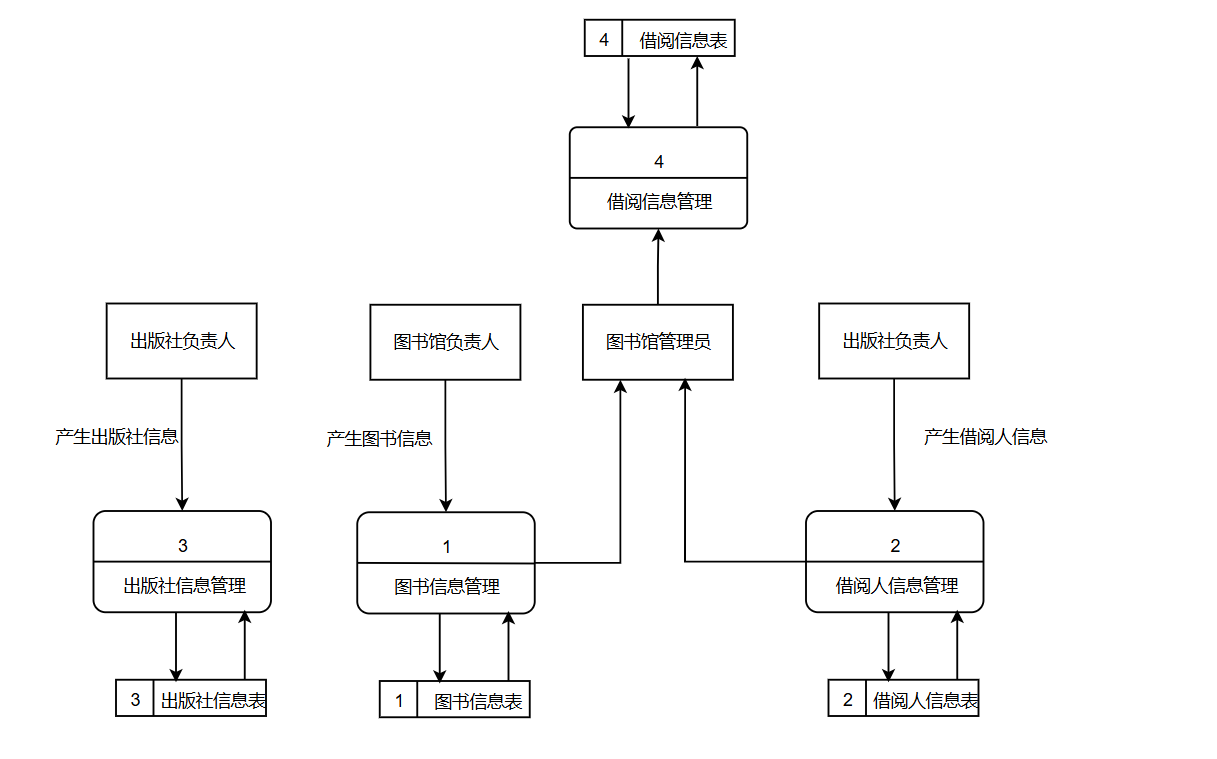
从借阅人角度：



从管理员角度：



（三）数据流分析



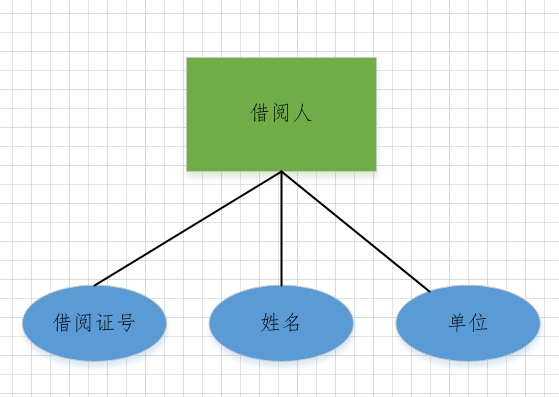
# 二、数据库概念结构设计

（一）实体分析

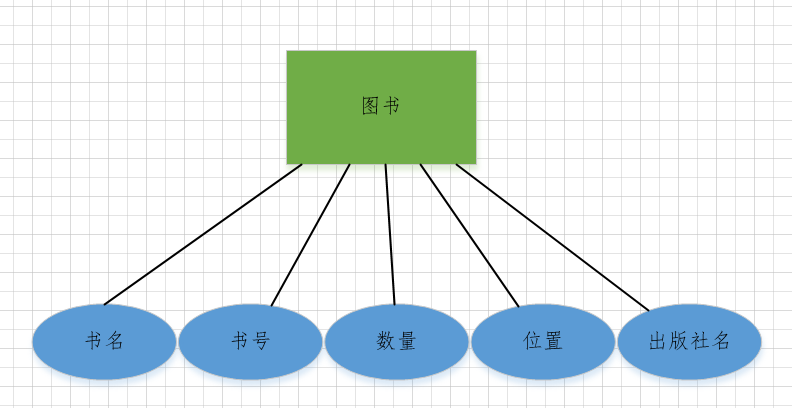
经需求分析，本次课程设计中包含四个实体，他们分别是：借书人实体、图书实体、出版社实体、图书借阅实体。

（二）属性分析

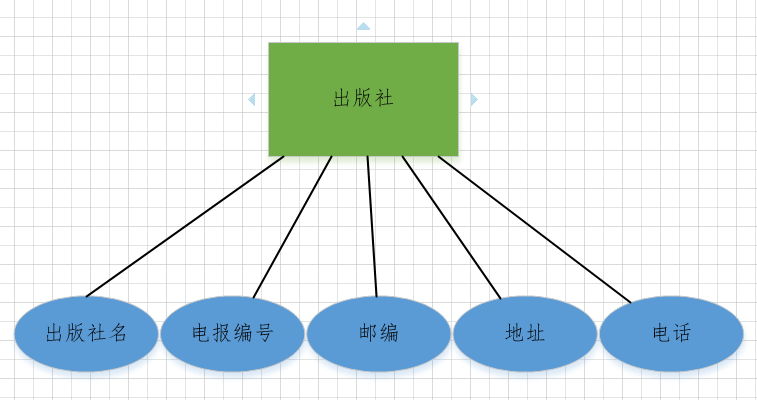
借书人实体属性；借阅证号，姓名，单位；如下；



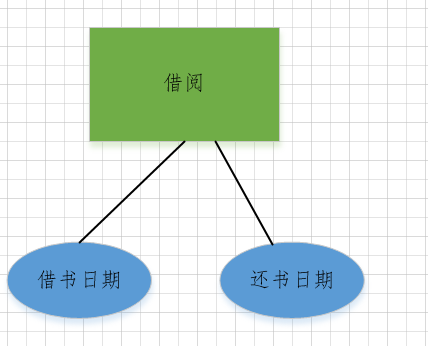
图书实体属性：书号、书名、数量、位置、出版社名。如下：



出版社实体属性：出版社名、电报编号、电话 , 邮编 , 地址。如下：



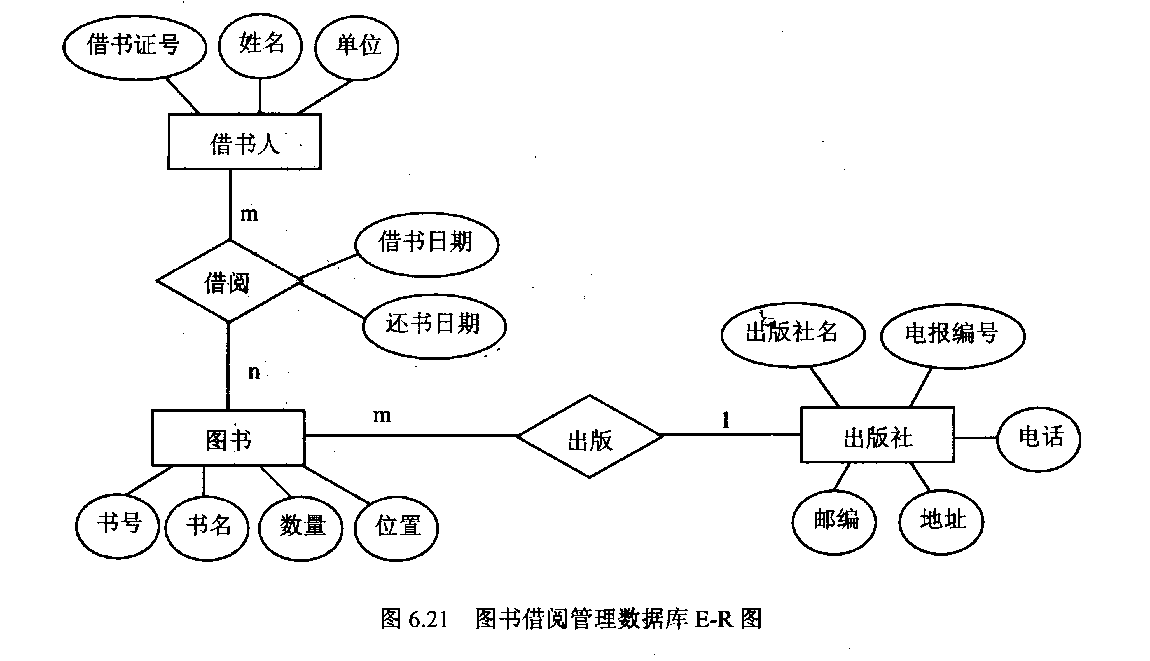
图书借阅实体属性：借书日期、还书日期。如下：



（三）联系分析

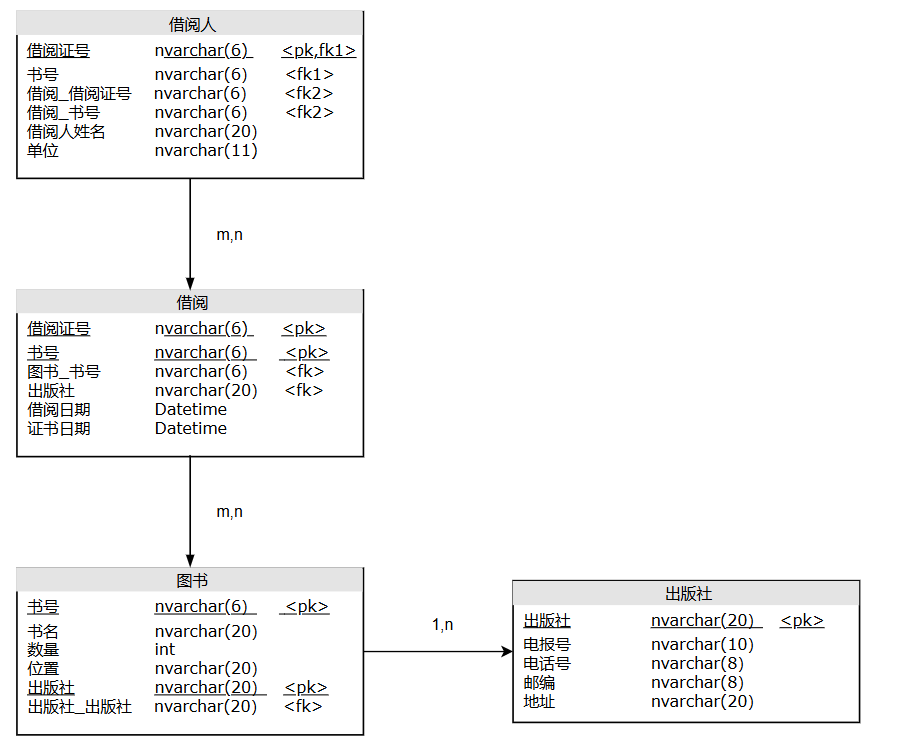
任何人可借多种书，任何一种书可为多个人所借，借书证号具有惟一性，借阅者与图书之间是多对多的关系。一个出版社可出版多种书籍，同一本书仅为一个出版社出版，出版社名具有惟一性，书与出版社之间为一对多关系。所有各类书籍均可由书号惟一标识，书号与书籍之间是一对一的关系。

E-R图分析如下：



（四）概念模型分析（.PDM图）

在经过需求分析和实体属性的分析，以及各实体之间的关系，最终得到概念模型如下：



# 三、数据库逻辑结构设计

（一）概念模型转化为逻辑模型

1.一对一关系的转化

借阅人与借阅证号是一对一的关系

2.一对多关系的转化

一本图书只有一家出版社，但一家出版社可以出版多本图书

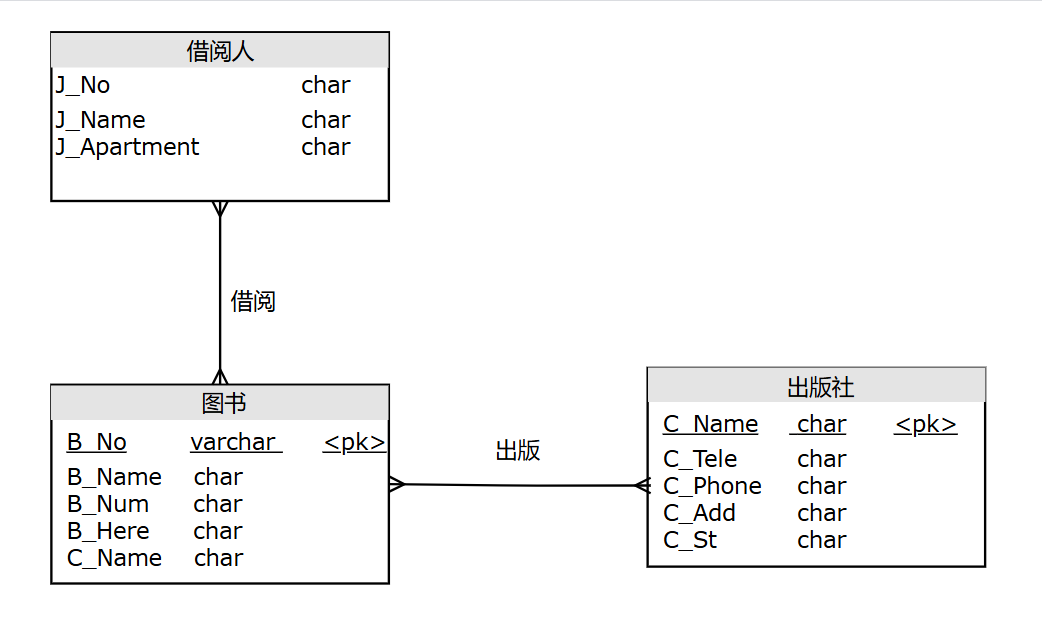
一个借阅人只有一个单位，但一个单位可以有多名借阅人

3.多对多关系的转化

多个图书馆给多个出版社供应多个图书，图书馆与出版社的关系是多对多。

多本图书可以被多个人借阅，图书与借阅人是多对多。

（二）逻辑模型设计（.PDM图）



# 四、数据库物理实现

（一）表设计

表1 单位信息表设计

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段解释 | 数据类型 | 备注 |
| J\_No | 借阅证号 | char(6) | 主键 |
| J\_Name | 借阅人名 | varchar(20) | 非空 |
| J\_Department | 单位 | char(11) | 唯一性 |

表2 借阅人信息表设计

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段解释 | 数据类型 | 备注 |
| B\_Name | 书名 | varchar(20) | 非空 |
| B\_Num | 数量 | int |  |
| B\_Here | 位置 | char(20) |  |
| C\_Name | 出版社名 | char(20) | 外键 |

表3 出版社信息表设计

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段解释 | 数据类型 | 备注 |
| C\_Name | 出版社名称 | char(20) | 主键 |
| C\_Tele | 电报编号 | char(10) | 非空 |
| C\_Phone | 电话 | char(8) |  |
| C\_Addr | 地址 | char(20) |  |
| C\_St | 邮编 | Char(6) |  |

表4 借阅信息表设计：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段解释 | 数据类型 | 备注 |
| J\_No | 借阅证号 | char(6) | 主键 |
| B\_No | 书号 | char(6) | 主键 |
| G\_day | 借阅日期 | Datetime |  |
| R\_day | 还书日期 | Datetime | 唯一性 |

（二）创建表和完整性约束代码设计

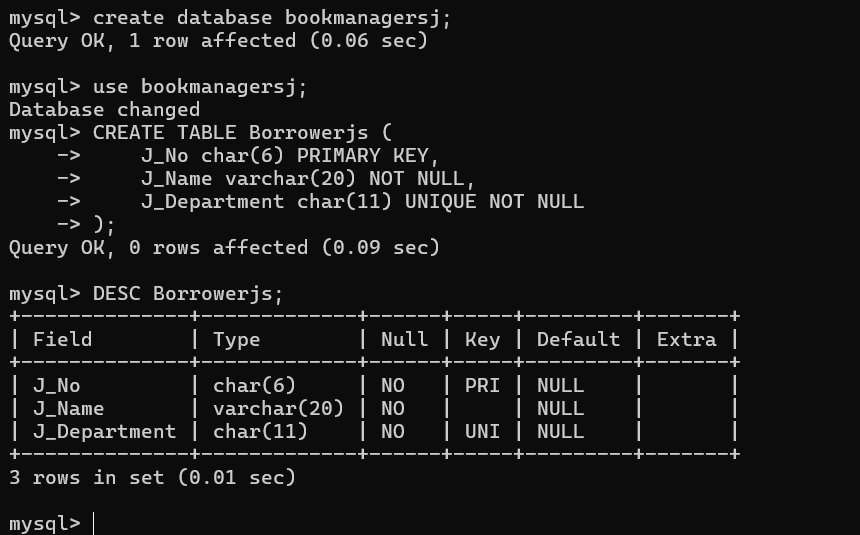
1. 创建借阅信息表及完整性约束：

CREATE TABLE Borrowersj (

J\_No char(6) PRIMARY KEY,

J\_Name varchar(20) NOT NULL,

J\_Department char(11) UNIQUE NOT NULL

);

2. 创建图书信息表及完整性约束：

CREATE TABLE Booksj (

B\_No char(6) PRIMARY KEY,

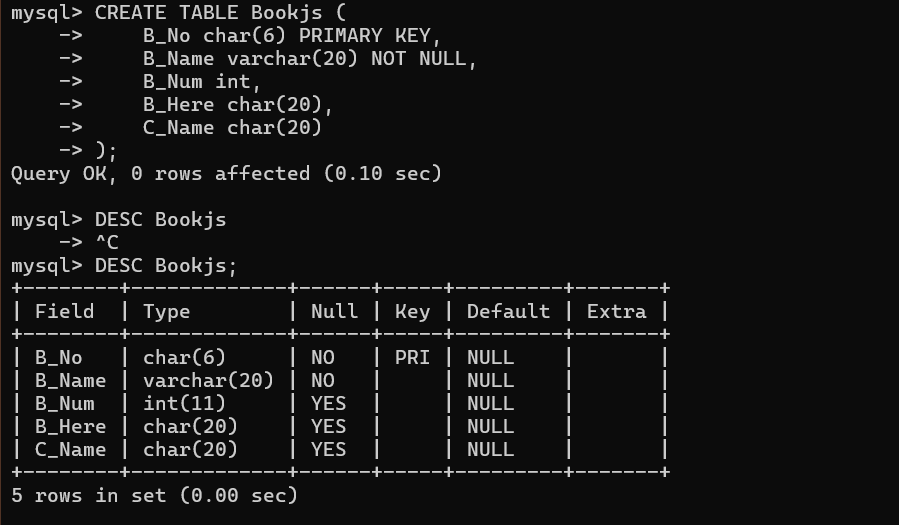
B\_Name varchar(20) NOT NULL,

B\_Num int,

B\_Here char(20),

C\_Name char(20)

);



3. 创建出版社信息及完整性约束：

CREATE TABLE Pubsj(

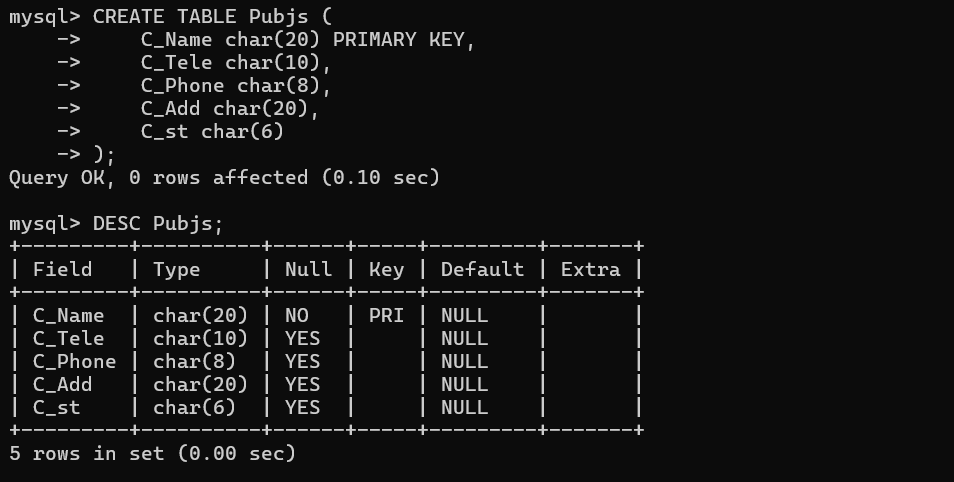
C\_Name char(20) primary key,

C\_Tele char(10),

C\_Phone char(8),

C\_Add char(20),

C\_st char(6)

);

4. 创建借阅关系表及完整性约束：

CREATE TABLE Borrowsj (

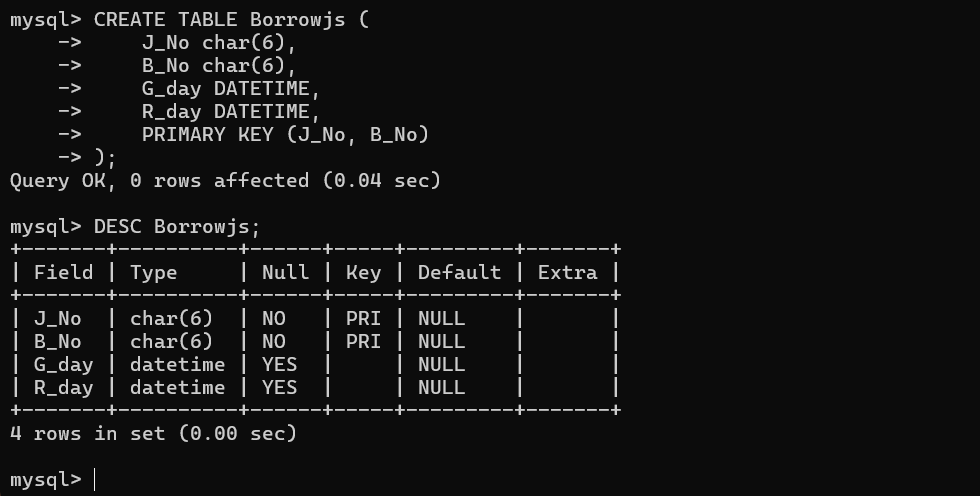
J\_No char(6),

B\_No char(6),

G\_day DATETIME,

R\_day DATETIME,

PRIMARY KEY (J\_No, B\_No)

);

（三）创建视图、索引、存储过程和触发器

1. 创建视图

1.1 创建图书与借阅者的编号信息的视图。

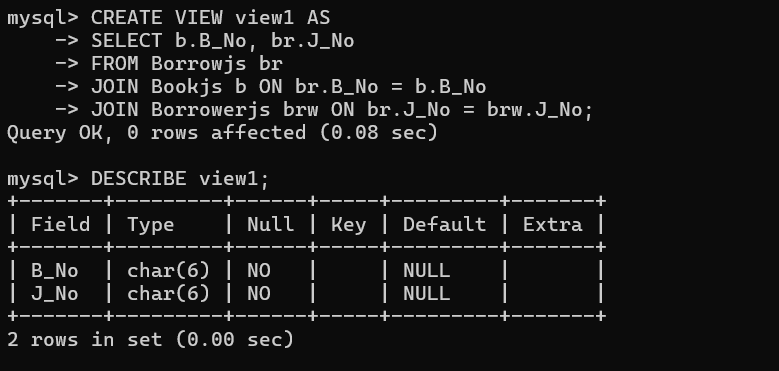
CREATE VIEW view1 AS

SELECT b.B\_No, br.J\_No

FROM Borrowjs br

JOIN Bookjs b ON br.B\_No = b.B\_No

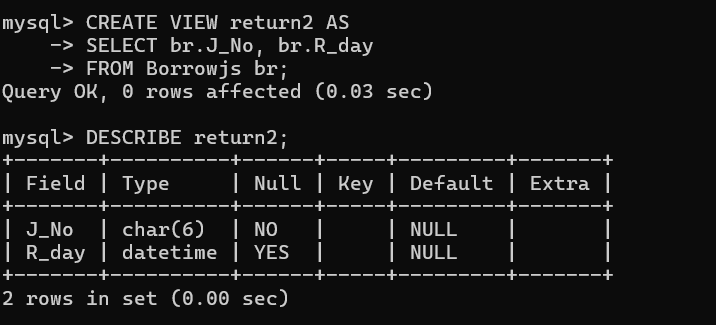
JOIN Borrowerjs brw ON br.J\_No = brw.J\_No;



1.2 创建借阅者归还图书日期信息的视图。

CREATE VIEW return2 AS

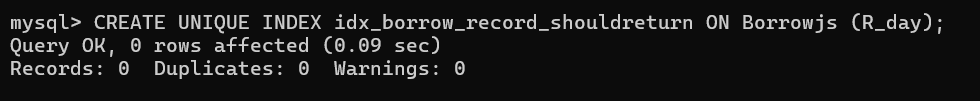
SELECT br.J\_No, br.R\_day

FROM Borrowjs br;

2. 创建索引

为数据库中的Borrowsj建立索引。这两个表都分别按照各自的shouldreturn项目按升序唯一排列。

CREATE UNIQUE INDEX idx\_borrow\_record\_shouldreturn ON Borrowjs (R\_day);



3. 创建存储过程

3.1 建立存储过程，输入读者编号得到该读者的所有借阅信息。

DELIMITER $$

CREATE PROCEDURE reader(

IN readerID VARCHAR(6)

)

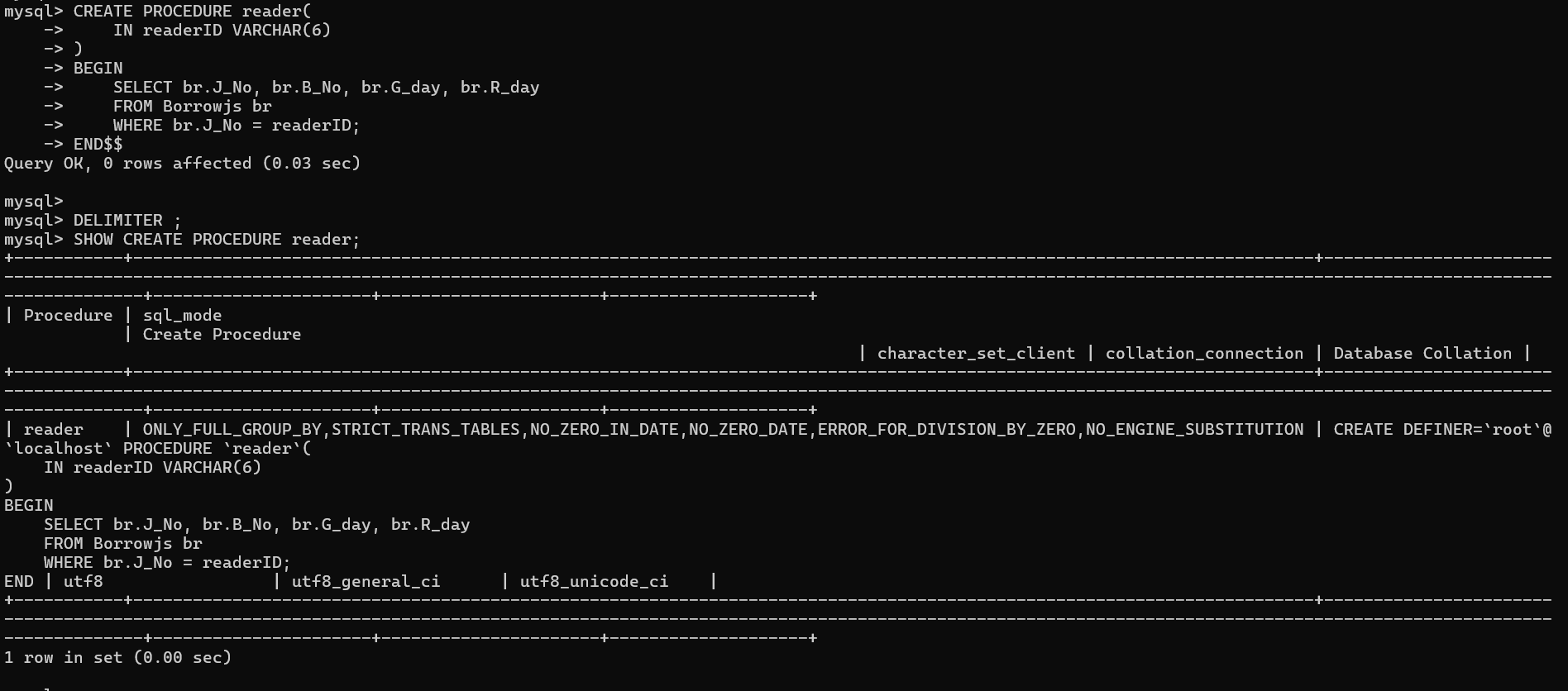
BEGIN

SELECT br.J\_No, br.B\_No, br.G\_day, br.R\_day

FROM Borrowjs br

WHERE br.J\_No = readerID;

END$$

DELIMITER ;

3.2 建立存储过程，输入图书编号得到该图书所有的信息。

DELIMITER $$

CREATE PROCEDURE book(

IN bookID VARCHAR(6)

)

BEGIN

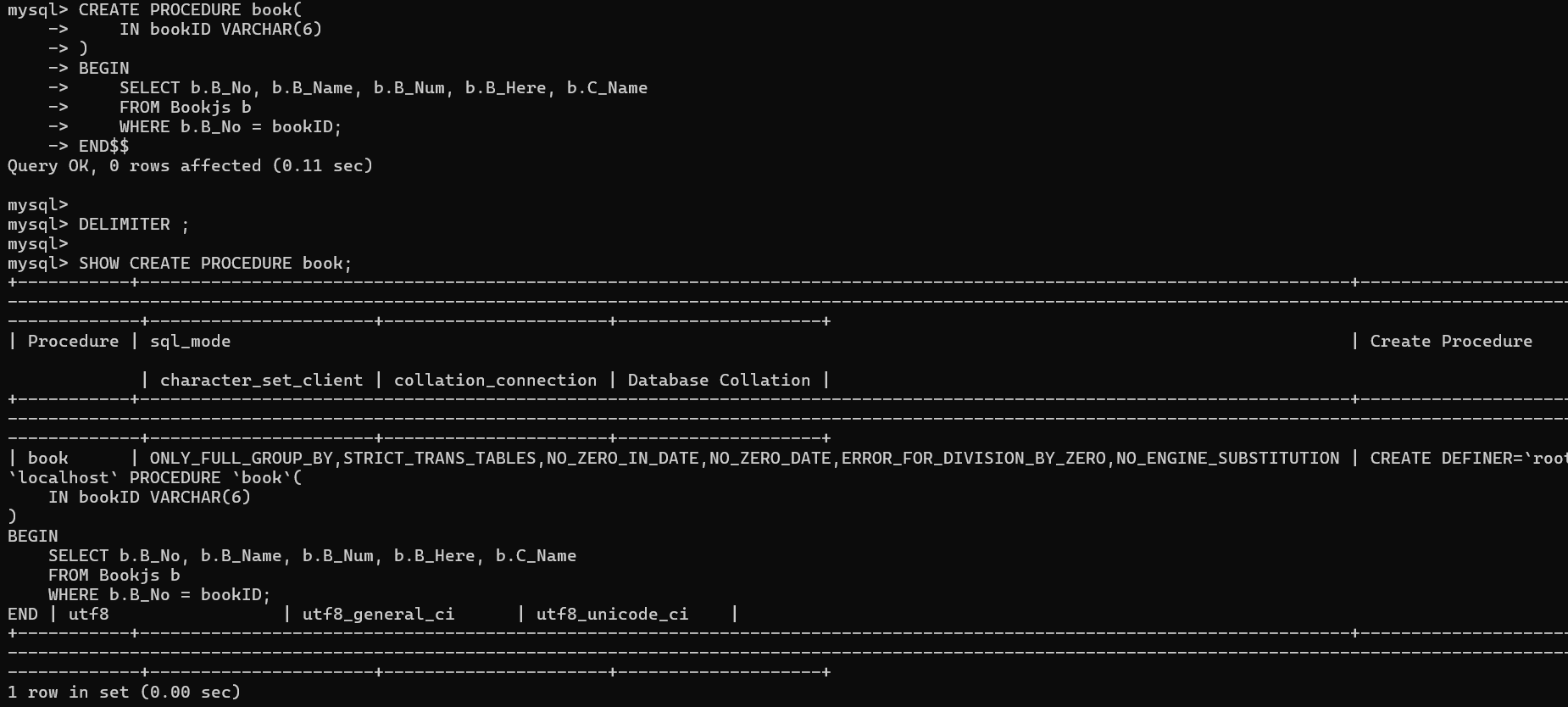
SELECT b.B\_No, b.B\_Name, b.B\_Num, b.B\_Here, b.C\_Name

FROM Bookjs b

WHERE b.B\_No = bookID;

END$$

DELIMITER ;



# 数据库功能调试

1. 数据插入

INSERT INTO Borrowersj (J\_No, J\_Name, J\_Department) VALUES

('J001', '张三', '计算机学院'),

('J002', '李四', '经济学院'),

('J003', '王五', '物理学院'),

('J004', '赵六', '化学学院'),

('J005', '孙七', '数学学院'),

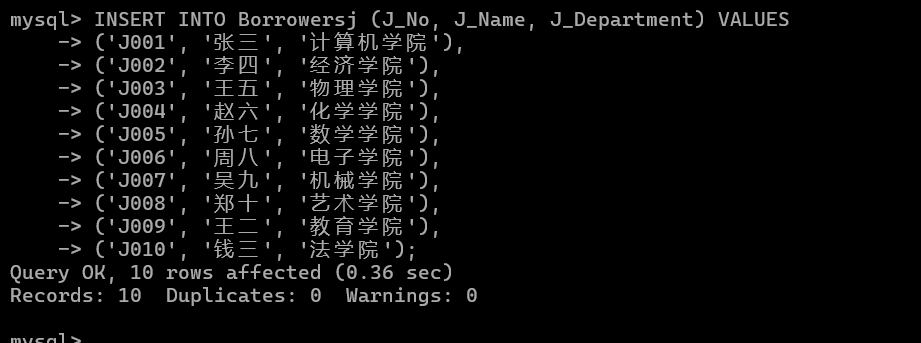
('J006', '周八', '电子学院'),

('J007', '吴九', '机械学院'),

('J008', '郑十', '艺术学院'),

('J009', '王二', '教育学院'),

('J010', '钱三', '法学院');



INSERT INTO Booksj (B\_No, B\_Name, B\_Num, B\_Here, C\_Name) VALUES

('B001', '高等数学', 20, '图书馆1楼', '清华出版社'),

('B002', '线性代数', 15, '图书馆2楼', '高教出版社'),

('B003', '概率论', 10, '图书馆2楼', '人民出版社'),

('B004', '数据库系统', 8, '图书馆3楼', '机械工业出版社'),

('B005', '操作系统', 12, '图书馆3楼', '电子工业出版社'),

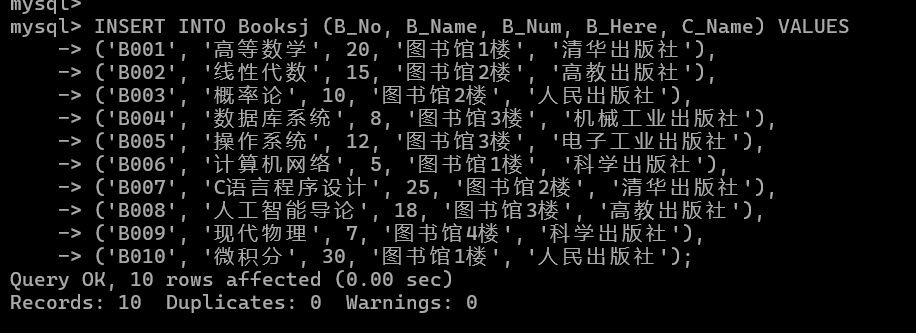
('B006', '计算机网络', 5, '图书馆1楼', '科学出版社'),

('B007', 'C语言程序设计', 25, '图书馆2楼', '清华出版社'),

('B008', '人工智能导论', 18, '图书馆3楼', '高教出版社'),

('B009', '现代物理', 7, '图书馆4楼', '科学出版社'),

('B010', '微积分', 30, '图书馆1楼', '人民出版社');



INSERT INTO Pubsj (C\_Name, C\_Tele, C\_Phone, C\_Add, C\_St) VALUES

('清华出版社', '0101234567', '12345678', '北京市海淀区', '正常'),

('高教出版社', '0107654321', '87654321', '北京市朝阳区', '正常'),

('人民出版社', '0201234567', '22334455', '广州市天河区', '正常'),

('机械工业出版社', '0103333444', '11223344', '北京市丰台区', '正常'),

('电子工业出版社', '0755123456', '55667788', '深圳市南山区', '正常'),

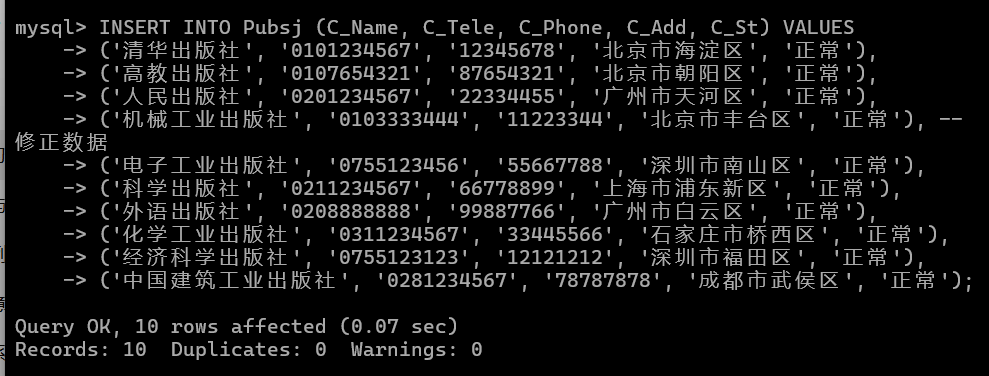
('科学出版社', '0211234567', '66778899', '上海市浦东新区', '正常'),

('外语出版社', '0208888888', '99887766', '广州市白云区', '正常'),

('化学工业出版社', '0311234567', '33445566', '石家庄市桥西区', '正常'),

('经济科学出版社', '0755123123', '12121212', '深圳市福田区', '正常'),

('中国建筑工业出版社', '0281234567', '78787878', '成都市武侯区', '正常');;



INSERT INTO Borrowsj (J\_No, B\_No, G\_day, R\_day) VALUES

('J001', 'B001', '2024-01-01', '2024-01-10'),

('J002', 'B002', '2024-01-05', '2024-01-15'),

('J003', 'B003', '2024-01-10', '2024-01-20'),

('J004', 'B004', '2024-01-12', '2024-01-22'),

('J005', 'B005', '2024-01-15', '2024-01-25'),

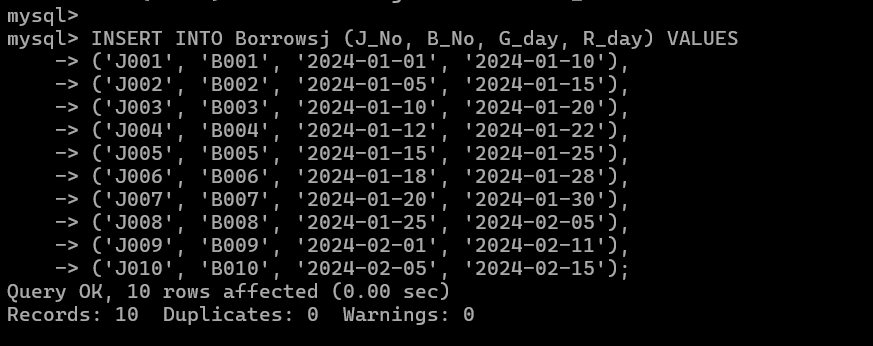
('J006', 'B006', '2024-01-18', '2024-01-28'),

('J007', 'B007', '2024-01-20', '2024-01-30'),

('J008', 'B008', '2024-01-25', '2024-02-05'),

('J009', 'B009', '2024-02-01', '2024-02-11'),

('J010', 'B010', '2024-02-05', '2024-02-15');



1. 查询模块

-- 查询图书表中的几列，并用别名

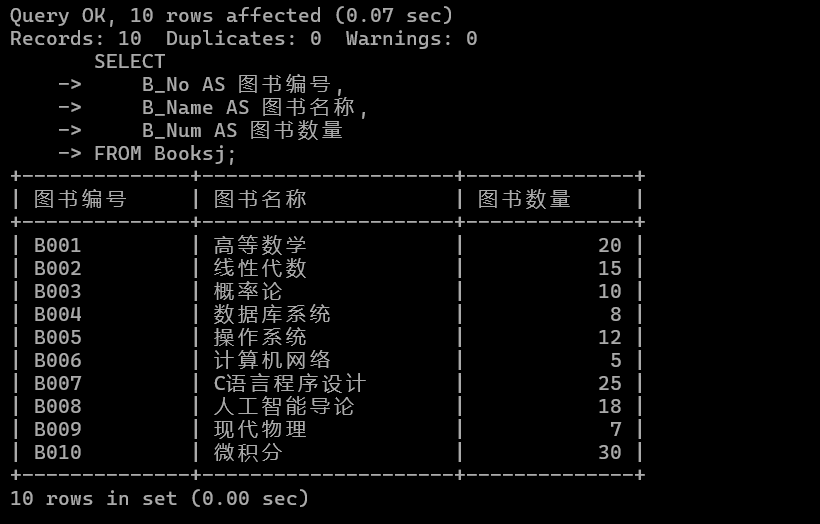
SELECT

B\_No AS 图书编号,

B\_Name AS 图书名称,

B\_Num AS 图书数量

FROM Booksj;



-- 计算库存数量大于10的图书的平均值、最大值和最小值

SELECT

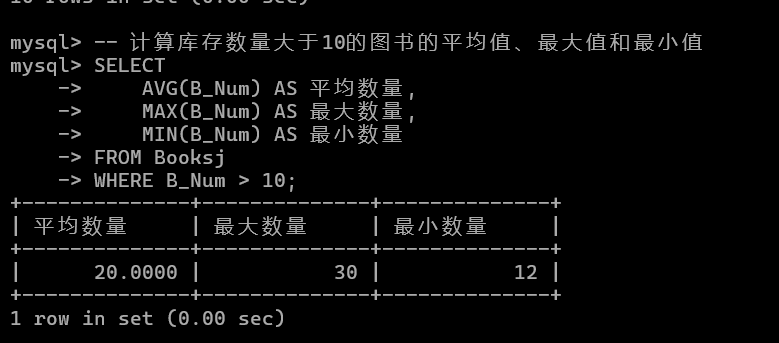
AVG(B\_Num) AS 平均数量,

MAX(B\_Num) AS 最大数量,

MIN(B\_Num) AS 最小数量

FROM Booksj

WHERE B\_Num > 10;

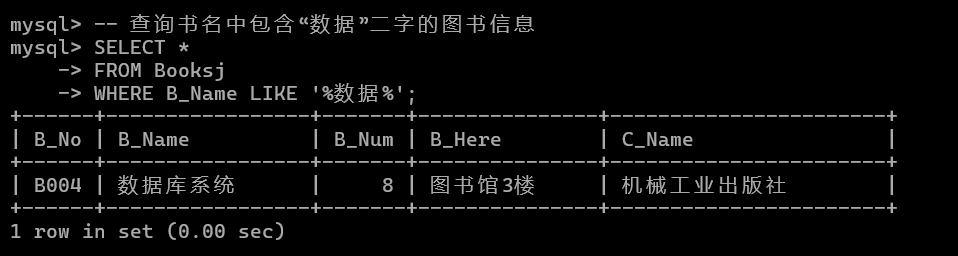


-- like查询书名中包含“数据”二字的图书信息

SELECT \*

FROM Booksj

WHERE B\_Name LIKE '%数据%';



-- 查询所有出版社电话为10位的出版社

SELECT \*

FROM Pubsj

WHERE LENGTH(C\_Tele) = 10;



-- 查询借阅关系中，借阅了图书的借阅人姓名和图书名称

SELECT

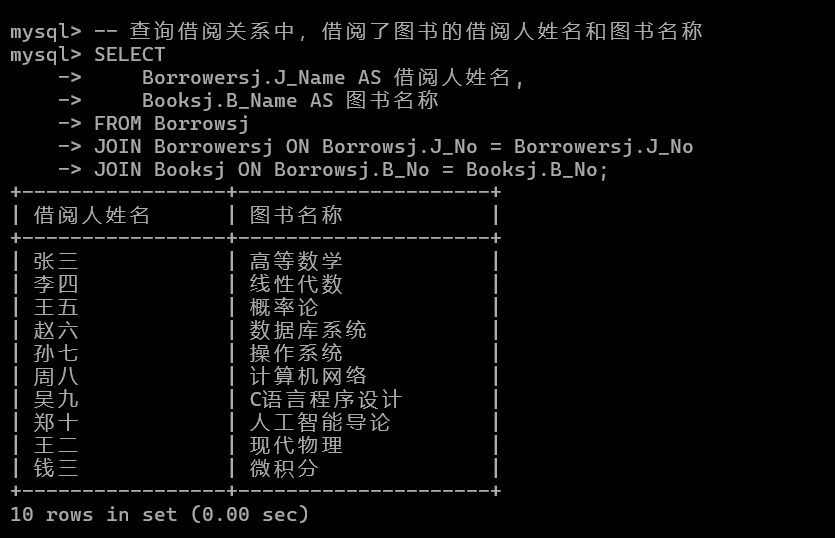
Borrowersj.J\_Name AS 借阅人姓名,

Booksj.B\_Name AS 图书名称

FROM Borrowsj

JOIN Borrowersj ON Borrowsj.J\_No = Borrowersj.J\_No

JOIN Booksj ON Borrowsj.B\_No = Booksj.B\_No;

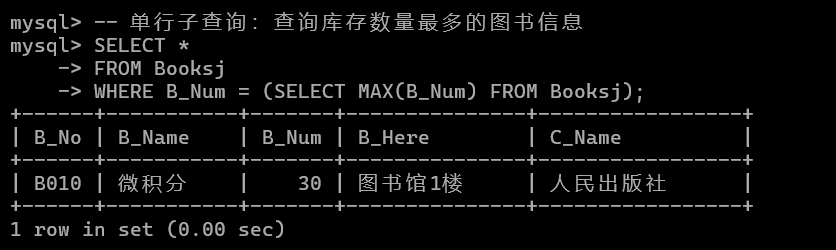


-- 单行子查询：查询库存数量最多的图书信息

SELECT \*

FROM Booksj

WHERE B\_Num = (SELECT MAX(B\_Num) FROM Booksj);

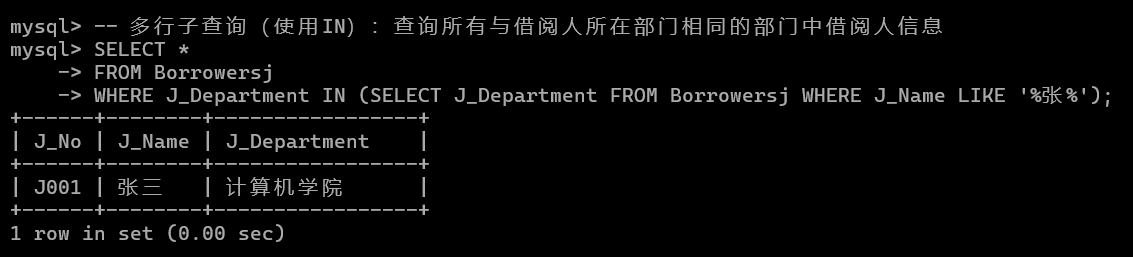


-- 多行子查询（使用IN）：查询所有与借阅人所在部门相同的部门中借阅人信息

SELECT \*

FROM Borrowersj

WHERE J\_Department IN (SELECT J\_Department FROM Borrowersj WHERE J\_Name LIKE '%张%');



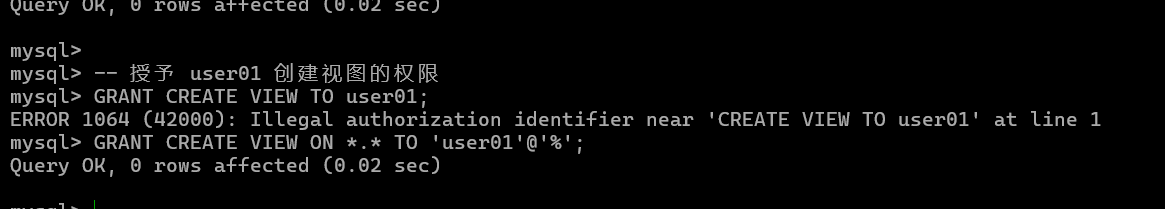
1. 视图模块

-- 创建用户 user01

CREATE USER user01 IDENTIFIED BY 'password';

-- 授予 user01 创建视图的权限

GRANT CREATE VIEW ON \*.\* TO 'user01'@'%';



-- 创建一个视图，查询借阅人及其借阅的图书信息

CREATE VIEW BorrowedBooks AS

SELECT

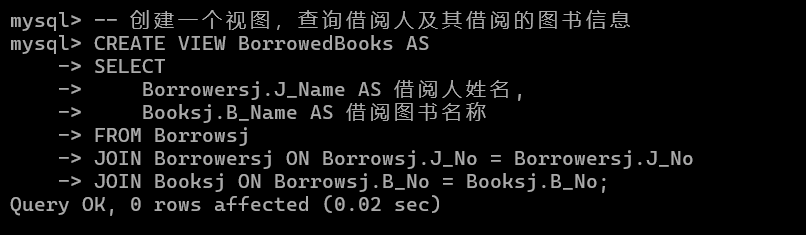
Borrowersj.J\_Name AS 借阅人姓名,

Booksj.B\_Name AS 借阅图书名称

FROM Borrowsj

JOIN Borrowersj ON Borrowsj.J\_No = Borrowersj.J\_No

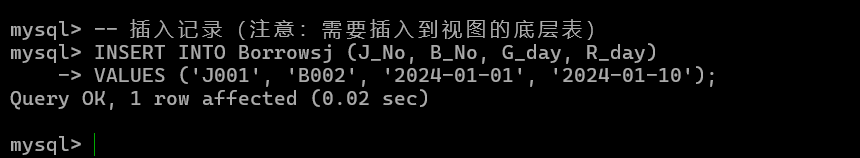
JOIN Booksj ON Borrowsj.B\_No = Booksj.B\_No;



-- 插入记录（注意：需要插入到视图的底层表）

INSERT INTO Borrowsj (J\_No, B\_No, G\_day, R\_day)

VALUES ('J001', 'B002', '2024-01-01', '2024-01-10');

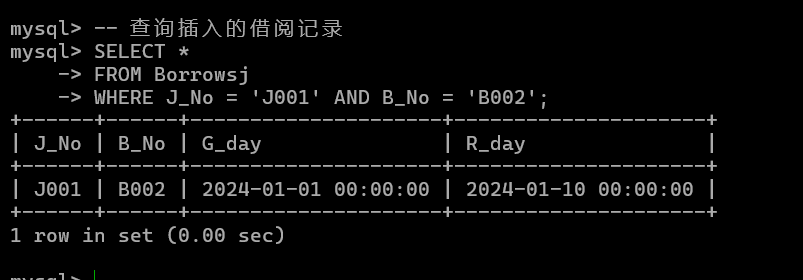


-- 查询插入的借阅记录

SELECT \*

FROM Borrowsj

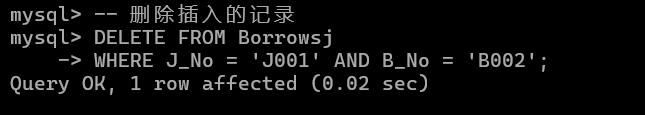
WHERE J\_No = 'J001' AND B\_No = 'B002';



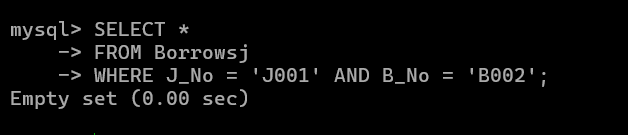
-- 删除插入的记录

DELETE FROM Borrowsj

WHERE J\_No = 'J001' AND B\_No = 'B002';



-- 删除后查询



1. 存储过程

-- 根据图书编号返回图书名称

DELIMITER //

CREATE PROCEDURE GetBookName(IN book\_no CHAR(6), OUT book\_name VARCHAR(20))

BEGIN

SELECT B\_Name INTO book\_name

FROM Booksj

WHERE B\_No = book\_no;

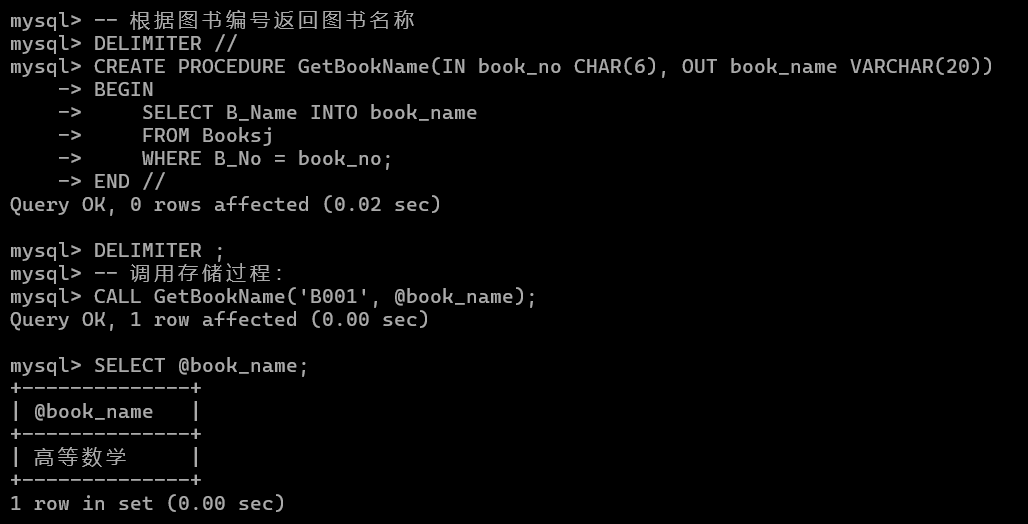
END //

DELIMITER ;

-- 调用存储过程：

CALL GetBookName('B001', @book\_name);

SELECT @book\_name;



**-- 统计库存数量大于10的图书数量**

DELIMITER //

CREATE FUNCTION CountBooksAbove10() RETURNS INT

DETERMINISTIC -- 声明函数是确定性的

READS SQL DATA -- 声明函数只读取数据，不修改数据

BEGIN

DECLARE book\_count INT;

SELECT COUNT(\*) INTO book\_count

FROM Booksj

WHERE B\_Num > 10;

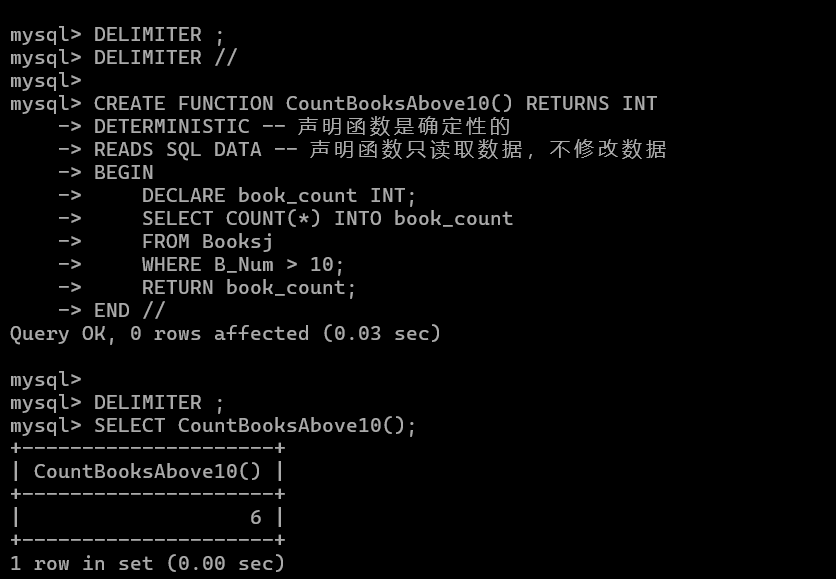
RETURN book\_count;

END //

DELIMITER ;

**-- 调用函数：**

SELECT CountBooksAbove10();



1. 触发器

**-- 触发器：在插入图书时自动设置默认库存数量为0**

DELIMITER //

CREATE TRIGGER SetDefaultBookNum

BEFORE INSERT ON Booksj

FOR EACH ROW

BEGIN

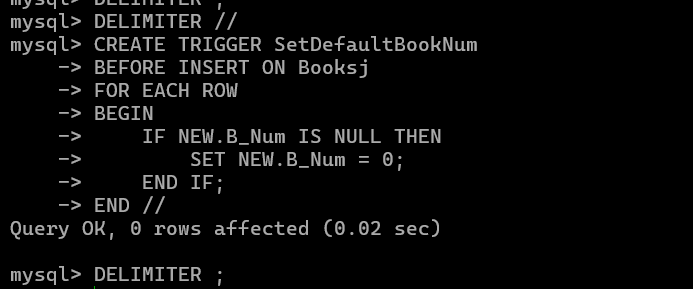
IF NEW.B\_Num IS NULL THEN

SET NEW.B\_Num = 0;

END IF;

END //

DELIMITER ;



**--** 当有新的借阅记录插入到 Borrowsj 表时，自动减少对应图书的库存数量 B\_Num。

DELIMITER //

CREATE TRIGGER UpdateBookStockAfterInsert

AFTER INSERT ON Borrowsj

FOR EACH ROW

BEGIN

-- 减少图书库存数量

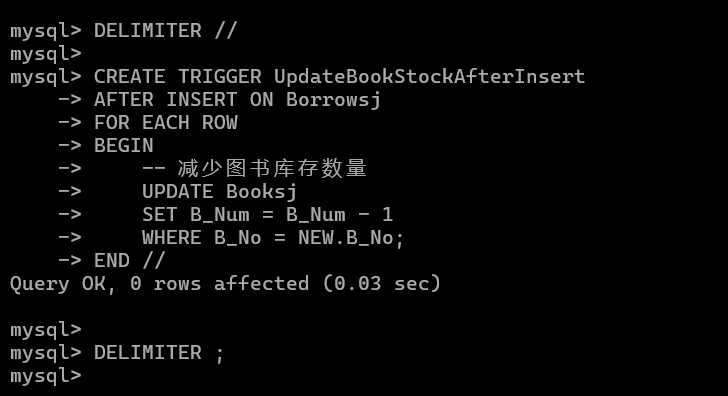
UPDATE Booksj

SET B\_Num = B\_Num - 1

WHERE B\_No = NEW.B\_No;

END //

DELIMITER ;



-- 插入新的借阅记录

INSERT INTO Borrowsj (J\_No, B\_No, G\_day, R\_day)

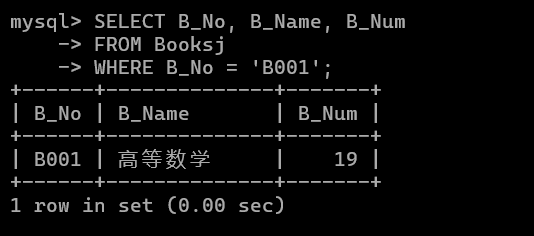
VALUES ('J002', 'B001', '2024-01-01', '2024-01-10');

-- 查看库存是否减少

SELECT B\_No, B\_Name, B\_Num

FROM Booksj

WHERE B\_No = 'B001';



结果：触发器验证成功

-- 如果某本书的库存数量为 0，禁止从 Booksj 表中删除该书。

DELIMITER //

CREATE TRIGGER PreventDeleteBookWithZeroStock

BEFORE DELETE ON Booksj

FOR EACH ROW

BEGIN

-- 如果图书库存为 0，抛出错误

IF OLD.B\_Num = 0 THEN

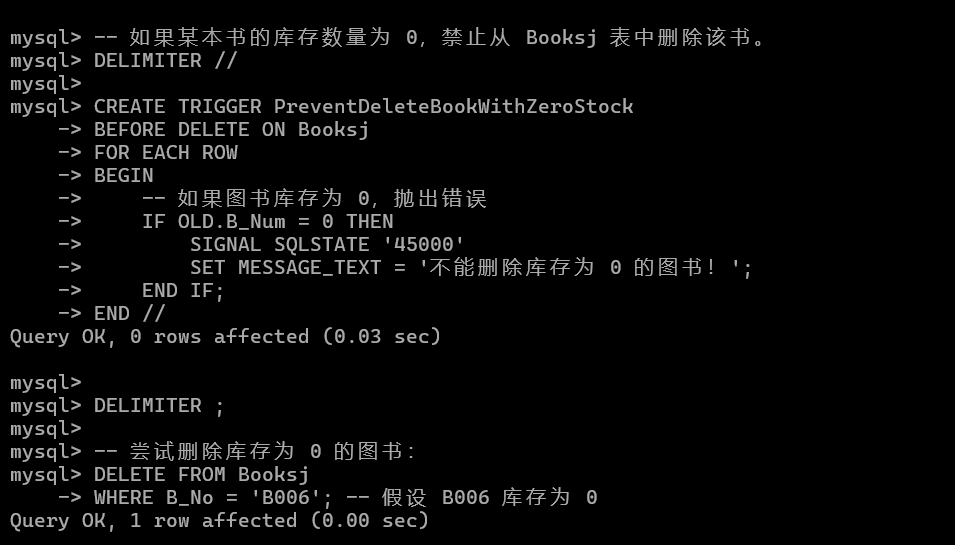
SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = '不能删除库存为 0 的图书！';

END IF;

END //

DELIMITER ;



-- 设置图书库存为 0

UPDATE Booksj

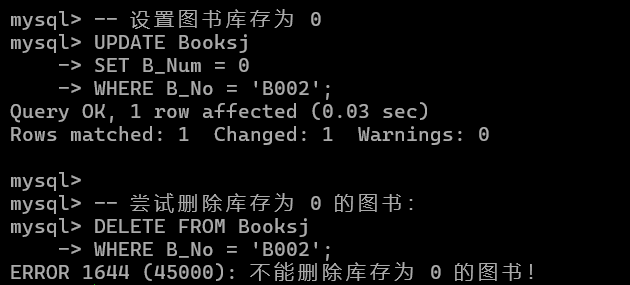
SET B\_Num = 0

WHERE B\_No = 'B002';

-- 尝试删除库存为 0 的图书：

DELETE FROM Booksj

WHERE B\_No = 'B002';



触发器验证成功！

# 六、设计总结

终于完成了这次的实验报告，心里有非常多的感触。通过这次实验报告，回顾到了这学期数据库学到的知识，小到描述一个语句，大到在两个表之间建立连接进行查询工作，我们意识到了所学的知识还有非常多的漏洞。之前也只是在课堂上听老师讲数据库系统的操作，也从来没有亲身的去将他应用到一个完整的系统上实施，也并没有我们所想象的那么容易。查漏补缺，发现问题后思考解决，同时也查阅了大量的资料。在这个过程中不仅是完成了一份数据库的实验报告，同时也提高了学习分析、解决问题的能力，收获非常大。

首先在建立需求分析的时候我们对书籍、借书人和管理人员之间的联系有点不清晰，题目中原本没有管理人员，但完成这个借阅系统的管理运行添加管理员之后会便捷许多。加入管理员后，将三者的关系重新进行了规划，才定下了实体。起步虽然有些困难，但大家一直在积极思考的解决问题，经常是做到中途时出现了问题，又重新翻书回去找。关系模式确立之后，逻辑模型与物理模型分析便可以进行了，之前一直没有完整的使用power designer实现逻辑图，这次也发现没有我想的那么容易。后来在书写SQL语句的时候忘记本来的数据受保护，无论如何也不能做修改。询问了周围的同学之后，又重新建立了新的查询。

通过这次实验让我感受到，如果想完整的去设计出一个查询的系统，首先自己要有扎实的基地，并且需要思路清晰，不能只去思考而不实践，也不能盲目去实践，找到对的方法才会事半功倍。

总的来说，这是非常有意义的一次实验，这次也学习到了非常多的知识，希望下次能做的更好。

# 七、附录-代码

-- 创建数据库

CREATE DATABASE BOOK\_SYS\_SJ;;

USE sj;

-- 创建借阅信息表及完整性约束：

CREATE TABLE Borrowersj (

J\_No char(6) PRIMARY KEY,

J\_Name varchar(20) NOT NULL,

J\_Department char(11) UNIQUE NOT NULL

);

-- 创建图书信息表及完整性约束：

CREATE TABLE Booksj (

B\_No char(6) PRIMARY KEY,

B\_Name varchar(20) NOT NULL,

B\_Num int,

B\_Here char(20),

C\_Name char(20)

);

-- 创建出版社信息及完整性约束：

CREATE TABLE Pubsj(

C\_Name char(20) primary key,

C\_Tele char(10),

C\_Phone char(8),

C\_Add char(20),

C\_st char(6)

);

-- 创建借阅关系表及完整性约束：

CREATE TABLE Borrowsj (

J\_No char(6),

B\_No char(6),

G\_day DATETIME,

R\_day DATETIME,

PRIMARY KEY (J\_No, B\_No)

);

-- 创建图书与借阅者的编号信息的视图。

CREATE VIEW view1 AS

SELECT b.B\_No, br.J\_No

FROM Borrowjs br

JOIN Bookjs b ON br.B\_No = b.B\_No

JOIN Borrowerjs brw ON br.J\_No = brw.J\_No;

-- 创建借阅者归还图书日期信息的视图。

CREATE VIEW return2 AS

SELECT br.J\_No, br.R\_day

FROM Borrowjs br;

-- 为数据库中的Borrowsj建立索引。这两个表都分别按照各自的shouldreturn项目按升序唯一排列。

CREATE UNIQUE INDEX idx\_borrow\_record\_shouldreturn ON Borrowjs (R\_day);

-- 创建存储过程

-- 建立存储过程，输入读者编号得到该读者的所有借阅信息。

DELIMITER $$

CREATE PROCEDURE reader(

IN readerID VARCHAR(6)

)

BEGIN

SELECT br.J\_No, br.B\_No, br.G\_day, br.R\_day

FROM Borrowjs br

WHERE br.J\_No = readerID;

END$$

DELIMITER ;

-- 建立存储过程，输入图书编号得到该图书所有的信息。

DELIMITER $$

CREATE PROCEDURE book(

IN bookID VARCHAR(6)

)

BEGIN

SELECT b.B\_No, b.B\_Name, b.B\_Num, b.B\_Here, b.C\_Name

FROM Bookjs b

WHERE b.B\_No = bookID;

END$$

DELIMITER ;

-- 数据插入

INSERT INTO Borrowersj (J\_No, J\_Name, J\_Department) VALUES

('J001', '张三', '计算机学院'),

('J002', '李四', '经济学院'),

('J003', '王五', '物理学院'),

('J004', '赵六', '化学学院'),

('J005', '孙七', '数学学院'),

('J006', '周八', '电子学院'),

('J007', '吴九', '机械学院'),

('J008', '郑十', '艺术学院'),

('J009', '王二', '教育学院'),

('J010', '钱三', '法学院');

INSERT INTO Booksj (B\_No, B\_Name, B\_Num, B\_Here, C\_Name) VALUES

('B001', '高等数学', 20, '图书馆1楼', '清华出版社'),

('B002', '线性代数', 15, '图书馆2楼', '高教出版社'),

('B003', '概率论', 10, '图书馆2楼', '人民出版社'),

('B004', '数据库系统', 8, '图书馆3楼', '机械工业出版社'),

('B005', '操作系统', 12, '图书馆3楼', '电子工业出版社'),

('B006', '计算机网络', 5, '图书馆1楼', '科学出版社'),

('B007', 'C语言程序设计', 25, '图书馆2楼', '清华出版社'),

('B008', '人工智能导论', 18, '图书馆3楼', '高教出版社'),

('B009', '现代物理', 7, '图书馆4楼', '科学出版社'),

('B010', '微积分', 30, '图书馆1楼', '人民出版社');

INSERT INTO Pubsj (C\_Name, C\_Tele, C\_Phone, C\_Add, C\_St) VALUES

('清华出版社', '0101234567', '12345678', '北京市海淀区', '正常'),

('高教出版社', '0107654321', '87654321', '北京市朝阳区', '正常'),

('人民出版社', '0201234567', '22334455', '广州市天河区', '正常'),

('机械工业出版社', '0103333444', '11223344', '北京市丰台区', '正常'),

('电子工业出版社', '0755123456', '55667788', '深圳市南山区', '正常'),

('科学出版社', '0211234567', '66778899', '上海市浦东新区', '正常'),

('外语出版社', '0208888888', '99887766', '广州市白云区', '正常'),

('化学工业出版社', '0311234567', '33445566', '石家庄市桥西区', '正常'),

('经济科学出版社', '0755123123', '12121212', '深圳市福田区', '正常'),

('中国建筑工业出版社', '0281234567', '78787878', '成都市武侯区', '正常');;

INSERT INTO Borrowsj (J\_No, B\_No, G\_day, R\_day) VALUES

('J001', 'B001', '2024-01-01', '2024-01-10'),

('J002', 'B002', '2024-01-05', '2024-01-15'),

('J003', 'B003', '2024-01-10', '2024-01-20'),

('J004', 'B004', '2024-01-12', '2024-01-22'),

('J005', 'B005', '2024-01-15', '2024-01-25'),

('J006', 'B006', '2024-01-18', '2024-01-28'),

('J007', 'B007', '2024-01-20', '2024-01-30'),

('J008', 'B008', '2024-01-25', '2024-02-05'),

('J009', 'B009', '2024-02-01', '2024-02-11'),

('J010', 'B010', '2024-02-05', '2024-02-15');

-- 查询图书表中的几列，并用别名

SELECT

B\_No AS 图书编号,

B\_Name AS 图书名称,

B\_Num AS 图书数量

FROM Booksj;

-- 计算库存数量大于10的图书的平均值、最大值和最小值

SELECT

AVG(B\_Num) AS 平均数量,

MAX(B\_Num) AS 最大数量,

MIN(B\_Num) AS 最小数量

FROM Booksj

WHERE B\_Num > 10;

-- like查询书名中包含“数据”二字的图书信息

SELECT \*

FROM Booksj

WHERE B\_Name LIKE '%数据%';

-- 查询所有出版社电话为10位的出版社

SELECT \*

FROM Pubsj

WHERE LENGTH(C\_Tele) = 10;

-- 查询借阅关系中，借阅了图书的借阅人姓名和图书名称

SELECT

Borrowersj.J\_Name AS 借阅人姓名,

Booksj.B\_Name AS 图书名称

FROM Borrowsj

JOIN Borrowersj ON Borrowsj.J\_No = Borrowersj.J\_No

JOIN Booksj ON Borrowsj.B\_No = Booksj.B\_No;

-- 单行子查询：查询库存数量最多的图书信息

SELECT \*

FROM Booksj

WHERE B\_Num = (SELECT MAX(B\_Num) FROM Booksj);

-- 多行子查询（使用IN）：查询所有与借阅人所在部门相同的部门中借阅人信息

SELECT \*

FROM Borrowersj

WHERE J\_Department IN (SELECT J\_Department FROM Borrowersj WHERE J\_Name LIKE '%张%');

-- 创建用户 user01

CREATE USER user01 IDENTIFIED BY 'password';

-- 授予 user01 创建视图的权限

GRANT CREATE VIEW ON \*.\* TO 'user01'@'%';

-- 创建一个视图，查询借阅人及其借阅的图书信息

CREATE VIEW BorrowedBooks AS

SELECT

Borrowersj.J\_Name AS 借阅人姓名,

Booksj.B\_Name AS 借阅图书名称

FROM Borrowsj

JOIN Borrowersj ON Borrowsj.J\_No = Borrowersj.J\_No

JOIN Booksj ON Borrowsj.B\_No = Booksj.B\_No;

-- 插入记录（注意：需要插入到视图的底层表）

INSERT INTO Borrowsj (J\_No, B\_No, G\_day, R\_day)

VALUES ('J001', 'B002', '2024-01-01', '2024-01-10');

-- 查询插入的借阅记录

SELECT \*

FROM Borrowsj

WHERE J\_No = 'J001' AND B\_No = 'B002';

-- 删除插入的记录

DELETE FROM Borrowsj

WHERE J\_No = 'J001' AND B\_No = 'B002';

-- 根据图书编号返回图书名称

DELIMITER //

CREATE PROCEDURE GetBookName(IN book\_no CHAR(6), OUT book\_name VARCHAR(20))

BEGIN

SELECT B\_Name INTO book\_name

FROM Booksj

WHERE B\_No = book\_no;

END //

DELIMITER ;

-- 调用存储过程：

CALL GetBookName('B001', @book\_name);

SELECT @book\_name;

**-- 统计库存数量大于10的图书数量**

DELIMITER //

CREATE FUNCTION CountBooksAbove10() RETURNS INT

DETERMINISTIC -- 声明函数是确定性的

READS SQL DATA -- 声明函数只读取数据，不修改数据

BEGIN

DECLARE book\_count INT;

SELECT COUNT(\*) INTO book\_count

FROM Booksj

WHERE B\_Num > 10;

RETURN book\_count;

END //

DELIMITER ;

**-- 调用函数：**

SELECT CountBooksAbove10();

**-- 触发器：在插入图书时自动设置默认库存数量为0**

DELIMITER //

CREATE TRIGGER SetDefaultBookNum

BEFORE INSERT ON Booksj

FOR EACH ROW

BEGIN

IF NEW.B\_Num IS NULL THEN

SET NEW.B\_Num = 0;

END IF;

END //

DELIMITER ;

**--** 当有新的借阅记录插入到 Borrowsj 表时，自动减少对应图书的库存数量 B\_Num。

DELIMITER //

CREATE TRIGGER UpdateBookStockAfterInsert

AFTER INSERT ON Borrowsj

FOR EACH ROW

BEGIN

-- 减少图书库存数量

UPDATE Booksj

SET B\_Num = B\_Num - 1

WHERE B\_No = NEW.B\_No;

END //

DELIMITER ;

-- 插入新的借阅记录

INSERT INTO Borrowsj (J\_No, B\_No, G\_day, R\_day)

VALUES ('J002', 'B001', '2024-01-01', '2024-01-10');

-- 查看库存是否减少

SELECT B\_No, B\_Name, B\_Num

FROM Booksj

WHERE B\_No = 'B001';

-- 如果某本书的库存数量为 0，禁止从 Booksj 表中删除该书。

DELIMITER //

CREATE TRIGGER PreventDeleteBookWithZeroStock

BEFORE DELETE ON Booksj

FOR EACH ROW

BEGIN

-- 如果图书库存为 0，抛出错误

IF OLD.B\_Num = 0 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = '不能删除库存为 0 的图书！';

END IF;

END //

DELIMITER ;

-- 设置图书库存为 0

UPDATE Booksj

SET B\_Num = 0

WHERE B\_No = 'B002';

-- 尝试删除库存为 0 的图书：

DELETE FROM Booksj

WHERE B\_No = 'B002';