**数据库课后作业**

**2212046 王昱 4.22**

**1.学校图书馆在线管理系统，该系统需要满足以下需求：**

（1）用户能够通过系统检索图书，查看图书的详细信息，如作者、出版日期、ISBN号等。

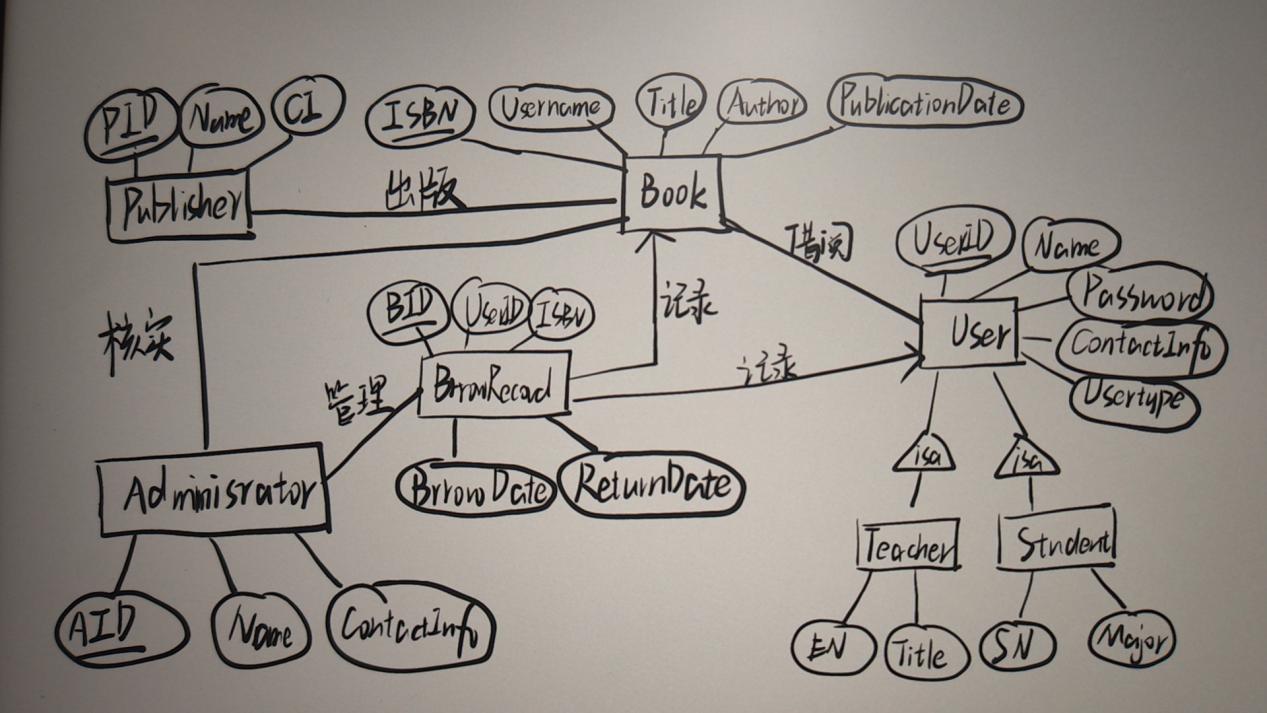
（2）用户可以借阅图书，还书，以及查询自己的借阅历史。

（3）图书管理员可以管理图书库存，包括添加新书、删除旧书、更新图书信息等。

（4）系统需要记录每本书的借阅历史，包括借阅日期、归还日期等。

**2.ER图和关系模式设计**

**a.ER模型图：**



**b.关系模式**

实体：

Book (ISBN,Title Author ,PublicationDate,StockQuantity ,CategoryID ,PublisherID);

User (UserID,Username,Password,ContactInfo,UserType);

由于学校图书馆只有老师学生，所以分为两个子类：

Student (UserID,StudentNumber,Major);

Teacher (UserID,EmployeeNumber,Title);

外键：FOREIGN KEY (UserID) REFERENCES User(UserID)

BorrowRecord (BorrowID,UserID,ISBN,BorrowDate,ReturnDate);

外键：

FOREIGN KEY (UserID) REFERENCES User(UserID),FOREIGN KEY (ISBN) REFERENCES Book(ISBN)；

Administrator (AdminID,Name,ContactInfo);

Publisher(PublisherID,PublisherName,ContactInfo);

关系：

借阅(UserID,ISBN);

记录(BorrowID,UserID);

记录(BorrowID,ISBN);

出版(PublisherID,ISBN);

管理(AdminID,BorrowID);

**c.用SQL语句创建上述关系模式：**

CREATE TABLE Book (

ISBN VARCHAR(13) PRIMARY KEY,

Title VARCHAR(255),

Author VARCHAR(100),

PublicationDate DATE,

StockQuantity INT,

CategoryID INT,

PublisherID INT

);

CREATE TABLE User (

UserID INT PRIMARY KEY,

Username VARCHAR(50),

Password VARCHAR(50),

ContactInfo VARCHAR(100),

UserType VARCHAR(50)

);

CREATE TABLE BorrowRecord (

BorrowID INT PRIMARY KEY,

UserID INT,

ISBN VARCHAR(13),

BorrowDate DATE,

ReturnDate DATE,

FOREIGN KEY (UserID) REFERENCES User(UserID),

FOREIGN KEY (ISBN) REFERENCES Book(ISBN)

);

CREATE TABLE Administrator (

AdminID INT PRIMARY KEY,

Name VARCHAR(100),

ContactInfo VARCHAR(100)

);

CREATE TABLE Publisher (

PublisherID INT PRIMARY KEY,

PublisherName VARCHAR(100),

ContactInfo VARCHAR(100)

);

-- 子类表

CREATE TABLE Student (

UserID INT PRIMARY KEY,

StudentNumber VARCHAR(50),

Major VARCHAR(100),

FOREIGN KEY (UserID) REFERENCES User(UserID)

);

CREATE TABLE Teacher (

UserID INT PRIMARY KEY,

EmployeeNumber VARCHAR(50),

Title VARCHAR(100),

FOREIGN KEY (UserID) REFERENCES User(UserID)

);

**d.给出五个查询样例：**

1.单表查询：查询所有图书的标题和作者

SELECT Title, Author FROM Book;

2.多表连接查询：查询所有借阅记录，包括用户信息和图书标题

SELECT u.Username, b.Title, br.BorrowDate

FROM BorrowRecord br

JOIN User u ON br.UserID = u.UserID

JOIN Book b ON br.ISBN = b.ISBN;

3.多表嵌套查询：查询借阅了超过3本书的用户

SELECT Username FROM User

WHERE UserID IN (

SELECT UserID FROM BorrowRecord

GROUP BY UserID

HAVING COUNT(ISBN) > 3

);

4.EXIST查询：查询至少借阅过一本书的用户

SELECT Username FROM User u

WHERE EXISTS (

SELECT \* FROM BorrowRecord br

WHERE br.UserID = u.UserID

);

5.聚合操作查询：查询每个分类下的图书数量

SELECT c.CategoryName, COUNT(b.ISBN) AS NumberOfBooks

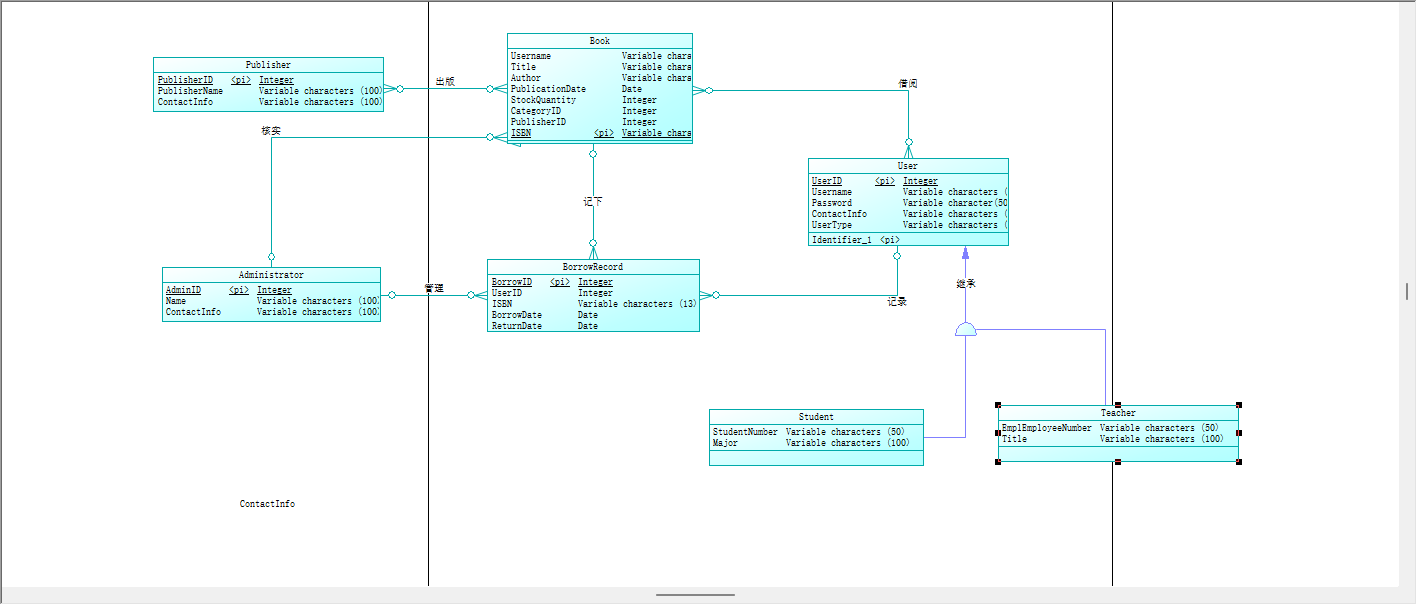
FROM Category c

JOIN Book b ON c.CategoryID = b.CategoryID

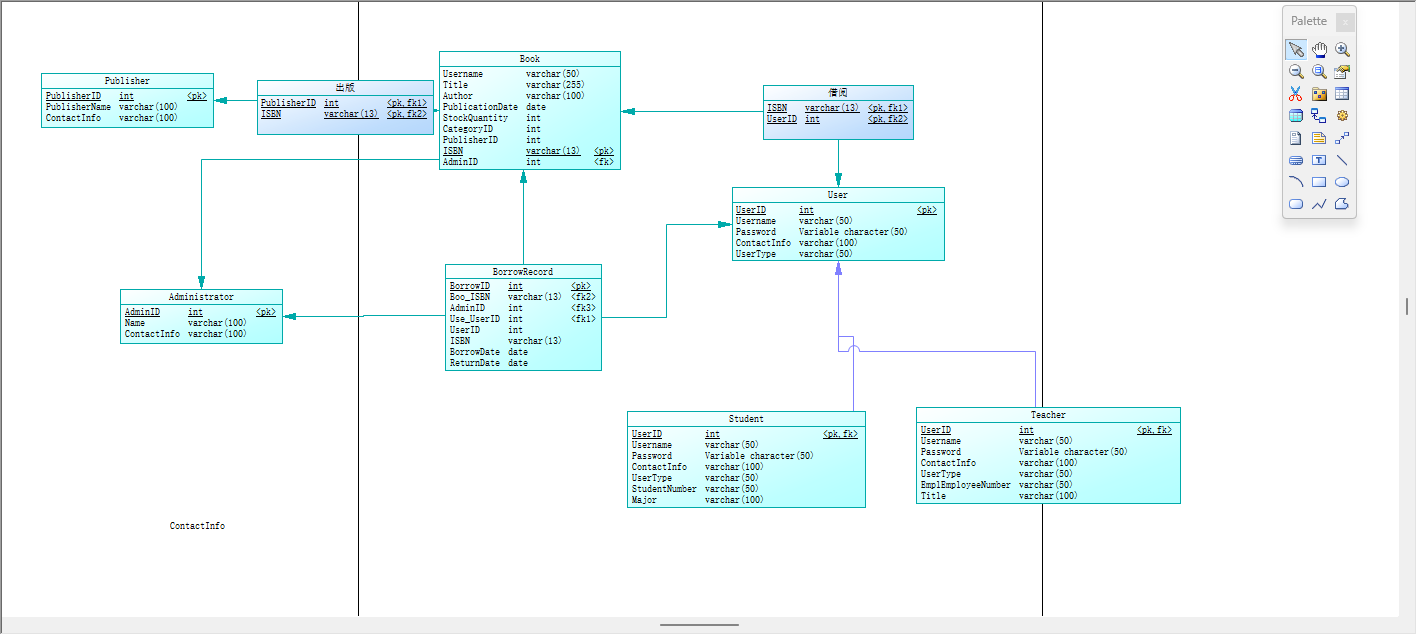
GROUP BY c.CategoryName;

**3.使用PowerDesign实现：**

**a.ER模型图：**

****

**b.关系模型图：**



**c.生成SQL语句：**

/\*==============================================================\*/

/\* DBMS name: MySQL 5.0 \*/

/\* Created on: 2024/4/22 0:44:36 \*/

/\*==============================================================\*/

drop table if exists Administrator;

drop table if exists Book;

drop table if exists BorrowRecord;

drop table if exists Publisher;

drop table if exists Student;

drop table if exists Teacher;

drop table if exists User;

drop table if exists 借阅;

drop table if exists 出版;

/\*==============================================================\*/

/\* Table: Administrator \*/

/\*==============================================================\*/

create table Administrator

(

AdminID int not null,

Name varchar(100),

ContactInfo varchar(100),

primary key (AdminID)

);

/\*==============================================================\*/

/\* Table: Book \*/

/\*==============================================================\*/

create table Book

(

Username varchar(50),

Title varchar(255),

Author varchar(100),

PublicationDate date,

StockQuantity int,

CategoryID int,

PublisherID int,

ISBN varchar(13) not null,

AdminID int,

primary key (ISBN)

);

/\*==============================================================\*/

/\* Table: BorrowRecord \*/

/\*==============================================================\*/

create table BorrowRecord

(

BorrowID int not null,

Boo\_ISBN varchar(13),

AdminID int,

Use\_UserID int,

UserID int,

ISBN varchar(13),

BorrowDate date,

ReturnDate date,

primary key (BorrowID)

);

/\*==============================================================\*/

/\* Table: Publisher \*/

/\*==============================================================\*/

create table Publisher

(

PublisherID int not null,

PublisherName varchar(100),

ContactInfo varchar(100),

primary key (PublisherID)

);

/\*==============================================================\*/

/\* Table: Student \*/

/\*==============================================================\*/

create table Student

(

UserID int not null,

Username varchar(50),

Password Variable character(50),

ContactInfo varchar(100),

UserType varchar(50),

StudentNumber varchar(50),

Major varchar(100),

primary key (UserID)

);

/\*==============================================================\*/

/\* Table: Teacher \*/

/\*==============================================================\*/

create table Teacher

(

UserID int not null,

Username varchar(50),

Password Variable character(50),

ContactInfo varchar(100),

UserType varchar(50),

EmplEmployeeNumber varchar(50),

Title varchar(100),

primary key (UserID)

);

/\*==============================================================\*/

/\* Table: User \*/

/\*==============================================================\*/

create table User

(

UserID int not null,

Username varchar(50),

Password Variable character(50),

ContactInfo varchar(100),

UserType varchar(50),

primary key (UserID)

);

/\*==============================================================\*/

/\* Table: 借阅 \*/

/\*==============================================================\*/

create table 借阅

(

ISBN varchar(13) not null,

UserID int not null,

primary key (ISBN, UserID)

);

/\*==============================================================\*/

/\* Table: 出版 \*/

/\*==============================================================\*/

create table 出版

(

PublisherID int not null,

ISBN varchar(13) not null,

primary key (PublisherID, ISBN)

);

alter table Book add constraint FK\_核实 foreign key (AdminID)

references Administrator (AdminID) on delete restrict on update restrict;

alter table BorrowRecord add constraint FK\_管理 foreign key (AdminID)

references Administrator (AdminID) on delete restrict on update restrict;

alter table BorrowRecord add constraint FK\_记下 foreign key (Boo\_ISBN)

references Book (ISBN) on delete restrict on update restrict;

alter table BorrowRecord add constraint FK\_记录 foreign key (Use\_UserID)

references User (UserID) on delete restrict on update restrict;

alter table Student add constraint FK\_继承 foreign key (UserID)

references User (UserID) on delete restrict on update restrict;

alter table Teacher add constraint FK\_继承2 foreign key (UserID)

references User (UserID) on delete restrict on update restrict;

alter table 借阅 add constraint FK\_借阅 foreign key (ISBN)

references Book (ISBN) on delete restrict on update restrict;

alter table 借阅 add constraint FK\_借阅2 foreign key (UserID)

references User (UserID) on delete restrict on update restrict;

alter table 出版 add constraint FK\_出版 foreign key (PublisherID)

references Publisher (PublisherID) on delete restrict on update restrict;

alter table 出版 add constraint FK\_出版2 foreign key (ISBN)

references Book (ISBN) on delete restrict on update restrict;

**4.比较分析：**

**a.设计差异：**

(1)自动化程度：

教材方法：教材通常详细介绍了将 ER 图手动转化为关系模式的步骤，例如实体到表、属性到字段、关系到外键等。这种方法通常需要开发人员根据教材中的指导手动进行转换。

PowerDesigner：使用PowerDesigner等工具可以自动将 ER 图转换为关系模式。该工具提供了一些自动化的转换功能，例如自动生成表和外键约束等。

影响：PowerDesigner的自动化程度较高，转换过程更快，且可能减少人为错误。然而，自动化转换可能不一定完全符合项目的特定需求。

(2)灵活性：

教材方法：教材的方法通常为手动转换，可以根据具体需求进行调整和优化。

PowerDesigner：使用 PowerDesigner 时，转换过程可能受到工具的预设规则的约束。虽然 PowerDesigner 提供了一定的定制化选项，但不如手动转换灵活。

影响：教材方法允许开发人员根据项目需求调整设计，从而更好地优化数据库结构。PowerDesigner 的预设规则可能对设计的灵活性产生一些限制。

(3)可视化设计：

教材方法：教材通常以文字和图表的形式解释转换过程，但可能缺乏直观的可视化设计工具。

PowerDesigner：PowerDesigner 提供了图形化的设计界面，使得设计过程更加直观和可视化。

影响：PowerDesigner 的可视化设计界面可以帮助开发人员更好地理解和设计数据库结构，提高工作效率。

(4)复杂关系的处理：

教材方法：教材方法需要开发人员自己处理复杂的关系和约束，例如多对多关系、弱实体等。

PowerDesigner：PowerDesigner 通常具有内置功能来处理复杂关系，例如自动创建关联表和外键约束。

影响：PowerDesigner 在处理复杂关系方面可能更方便，但开发人员需要熟悉工具的具体功能。

**b.PowerDesigner生成的SQL语句特点：**

(1)PowerDesigner 工具生成的 SQL 语句有什么样的特点？

PowerDesigner生成的 SQL 语句通常是基于 ER 图的关系模型设计自动生成的。这些 SQL 语句通常符合 SQL 标准，结构合理。

PowerDesigner会自动创建表、索引、外键、约束等内容。此外，SQL 语句还可能包含数据类型定义、约束定义、默认值设置等，以确保数据完整性和一致性。

(2)为什么会出现一些附加语句？

附加语句是为了确保数据库设计的完整性和稳定性。它们包括定义外键、唯一约束、检查约束、默认值、索引等。

这些附加语句可以帮助保持数据完整性、防止数据违规，提高数据库性能，并确保数据查询的准确性和效率。

(3)附加语句的作用是什么？

数据完整性：附加语句（例如外键和约束）可以确保数据库中的数据具有一致性和完整性，避免出现非法数据。

数据性能：附加的索引和优化语句有助于加快数据库查询速度，提高数据库性能。

数据安全性：附加语句可以设置数据访问权限和验证逻辑，确保数据安全和准确。

自动化管理：附加语句可以用于设置默认值、触发器等逻辑，简化数据管理和维护过程。