LIBRARY MANAGEMENT SYSTEM

PROJECT REPORT
for
21CSS205P - DATABASE MANAGEMENT SYSTEMS

Submitted by

Vishwajeet Kumar [RA2311003010624]
Aarjav Godha [RA2311003010628]
Paridhi Tiwari [RA2311003010647]

Under the Guidance of

Dr. Jeyasekar A
(Associate Professor, Department of computing technologies)

In partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING

DEPARTMENT OF COMPUTING TECHNOLOGIES
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR- 603203
March 2025

LIBRARY MANAGEMENT SYSTEM

Project Description

The **Library Management System** is a database-driven application designed to automate and streamline library operations. It enables efficient book cataloging, member registration, book borrowing, return tracking, and inventory management. The system ensures smooth coordination between librarians and library users, reducing manual errors and improving overall efficiency.

Objectives

- **Automate Library Processes** – Minimize manual work in book transactions and record-keeping.
- **Efficient Book Tracking** – Maintain accurate records of book availability and borrowing history.
- **User Management** – Implement role-based access control for librarians and members.
- **Fine Calculation** – Automate overdue fine calculation and notifications.
- **Inventory Management** – Ensure real-time updates on book stock and generate reports.

Scope of the Project

1. **Book Management** – Add, update, remove, and categorize books.
2. **User Management** – Register and manage members with unique IDs.
3. **Borrowing & Returning** – Automate book issue and return processes.
4. **Fine Management** – Calculate fines based on due dates and notify users.
5. **Inventory & Reports** – Maintain book stock and generate transaction reports.

Database Design & SQL Operations

Tables & Relationships

1. **Books Table (books)** – Stores book details (ID, title, author, availability, price, copies).
2. **Users Table (members) –** Stores member details (ID, name, address, membership start
   & expiry date, password).
3. **Transactions Table (books_issued)** – Tracks book borrow records (Book ID, User ID).
4. **Publishers Table (publishers)** – Stores publisher details (ID, name, address, password).

SQL Operations & Set Operations

1. **Count the Number of Books in Each Category (GROUP BY)**
   SELECT category, COUNT(book_id) AS total_books FROM books GROUP BY category;
2. **Get the Total Fine Collected per User (GROUP BY)**
   SELECT user_id, SUM(amount) AS total_fine FROM fines GROUP BY user_id;

3. **List Books in Alphabetical Order (ORDER BY)**
   SELECT * FROM books ORDER BY title ASC;

4. **Retrieve Recently Borrowed Books (ORDER BY)**
   SELECT user_id, book_id, borrow_date FROM transactions ORDER BY borrow_date DESC;

5. **Get the Number of Borrowed Books per User (GROUP BY & ORDER BY)**
   SELECT user_id, COUNT(book_id) AS books_borrowed FROM transactions GROUP BY user_id ORDER BY books_borrowed DESC;

6. **Selection (σ) – Retrieve Available Books**
   SELECT * FROM books WHERE availability = 'Yes';
   Relational Algebra: σavailability='Yes'(books)

7. **Projection (π) – Extract Book Titles and Authors**
   SELECT title, author FROM books;
   Relational Algebra: πtitle, author(books)

8. **Set Difference (-) – Find Members Who Haven't Borrowed Any Books**
   SELECT member_id FROM members WHERE member_id NOT IN (SELECT issuedto FROM books_issued);
   Relational Algebra: πmember_id(members) - πissuedto(books_issued)

9. **Union (∪) – Get a Combined List of Publisher and Member Addresses**
   SELECT address FROM publishers UNION SELECT address FROM members;
   Relational Algebra: πaddress(publishers) ∪ πaddress(members)

10. **Intersection (∩) – Find Users Who Have an Active Membership and Borrowed Books**
    SELECT member_id FROM members INTERSECT SELECT issuedto FROM books_issued;
    Relational Algebra: πmember_id(members) ∩ πissuedto(books_issued)

11. **Join (⋈) – Retrieve Borrowed Books with User Names**
    SELECT members.member_id, members.name, books.title FROM members INNER JOIN books_issued ON members.member_id =

books_issued.issuedto INNER JOIN books ON books_issued.bid = books.bid;

Relational Algebra: members ⋈members.member_id = books_issued.issuedto books_issued   ⋈books_issued.bid = books.bid books

<u>Relational Schema for Library Management System</u>

- **Books (books) → (bid PK, title, author, status, price, copies)**
- **Users (members) → (member_id PK, name, address, issue_date, exp_date, password)**
- **Transactions (books_issued) → (bid PK, issuedto FK (member_id))**
- **Publishers (publishers) → (pub_id PK, name, address, password)**

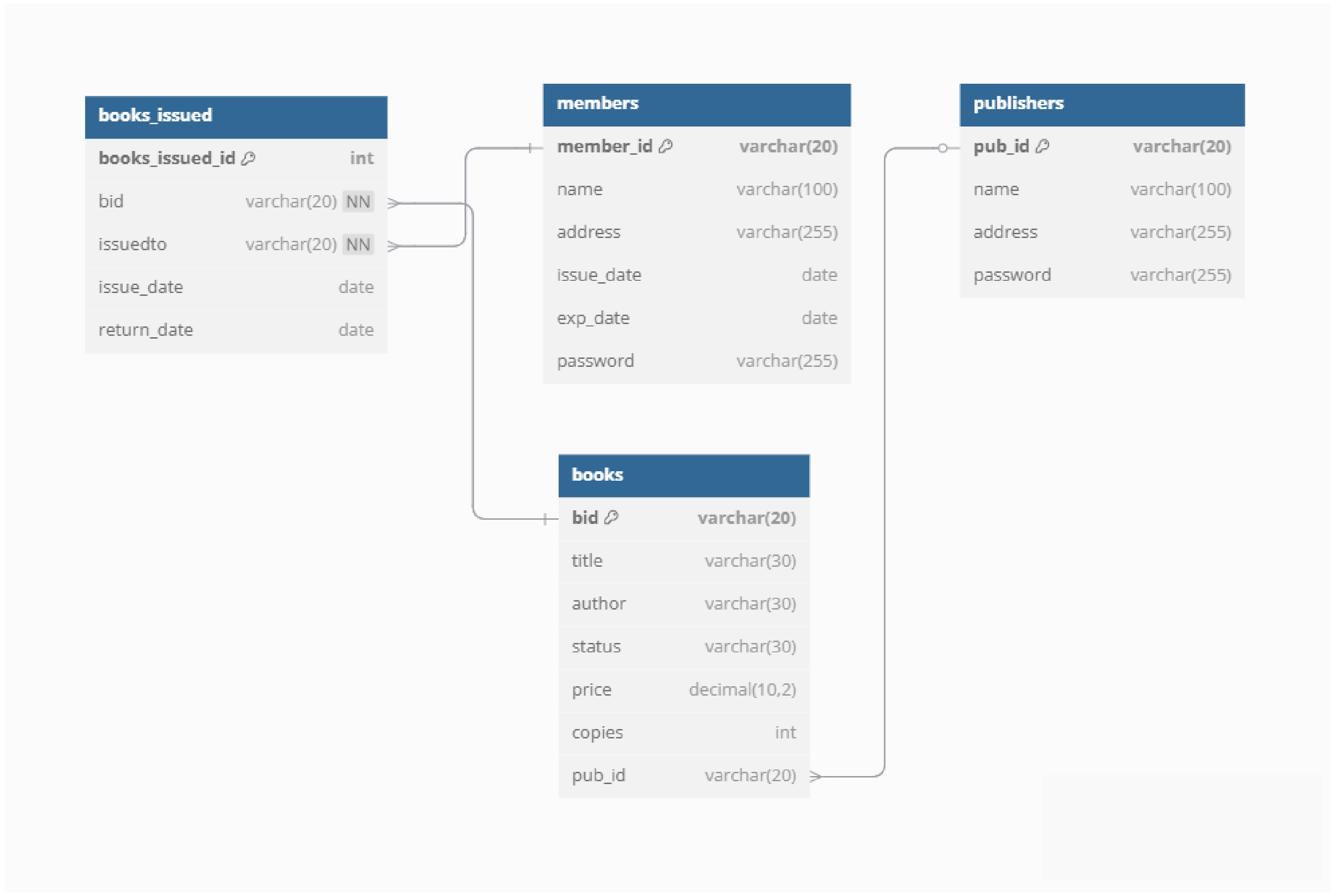
<u>Primary Keys (PK):</u>

- Books (books) → bid (Primary Key)
- Users (members) → member_id (Primary Key)
- Transactions (books_issued) → (bid, issuedto) (Composite Primary Key)
- Publishers (publishers) → pub_id (Primary Key)

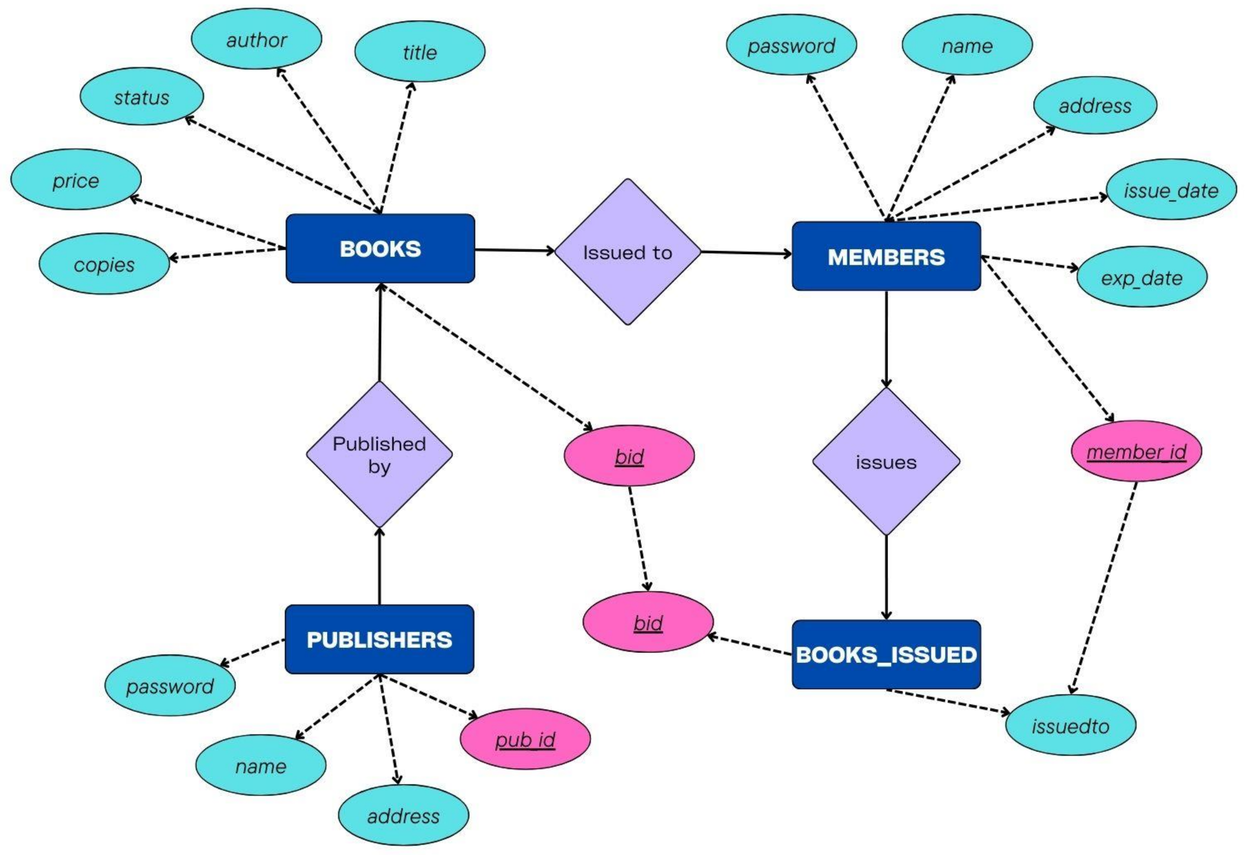
<u>Foreign Keys (FK):</u>

- Transactions (books_issued)
    - bid → References books(bid)
    - issuedto → References members(member_id)

RELATIONAL SCHEMA



ER DIAGRAM


author
title
status
price
copies
BOOKS
Issued to
MEMBERS
password
name
address
issue_date
exp_date
Published by
bid
issues
member id
PUBLISHERS
bid
BOOKS_ISSUED
password
name
address
pub id
issuedto