**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).

1. **Users Table (members) –** Stores member details (ID, name, address, membership start

& expiry date, password).

1. **Transactions Table (books\_issued)** – Tracks book borrow records (Book ID, User

ID).

1. **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;

1. **Get the Total Fine Collected per User (GROUP BY)**

SELECT user\_id, SUM(amount) AS total\_fine FROM fines GROUP BY user\_id;

1. **List Books in Alphabetical Order (ORDER BY)**

SELECT \* FROM books ORDER BY title ASC;

1. **Retrieve Recently Borrowed Books (ORDER BY)**

SELECT user\_id, book\_id, borrow\_date FROM transactions ORDER BY borrow\_date DESC;

1. **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;

1. **Selection (σ) – Retrieve Available Books**

SELECT \* FROM books WHERE availability = 'Yes';

**Relational Algebra:** σavailability='Yes'(books)

1. **Projection (π) – Extract Book Titles and Authors**

SELECT title, author FROM books;

**Relational Algebra:** πtitle, author(books)

1. **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)**

1. **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)

1. **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)

1. **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

**Relational Schema for Library Management System**

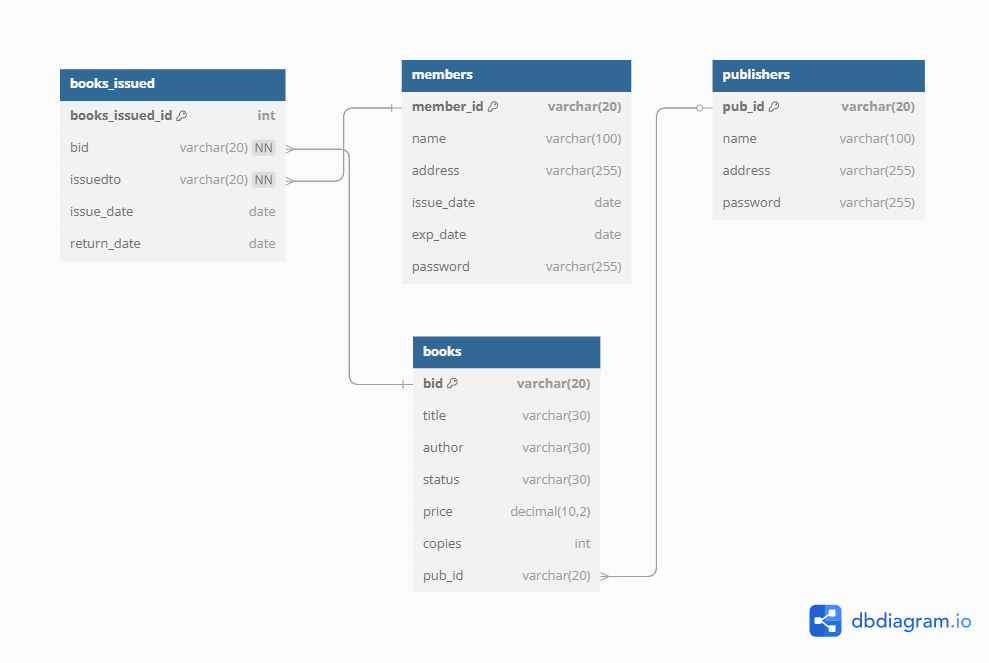
* **Books (books) → (bid PK, title, author, status, price, copies)**
* **Users (members) → (member\_id PK, name, address, issue\_date, exp\_date, password)**
* **Transactions (books\_issued) → (bid PK, issuedto FK (member\_id))**
* **Publishers (publishers) → (pub\_id PK, name, address, password)**

**Primary Keys (PK):**

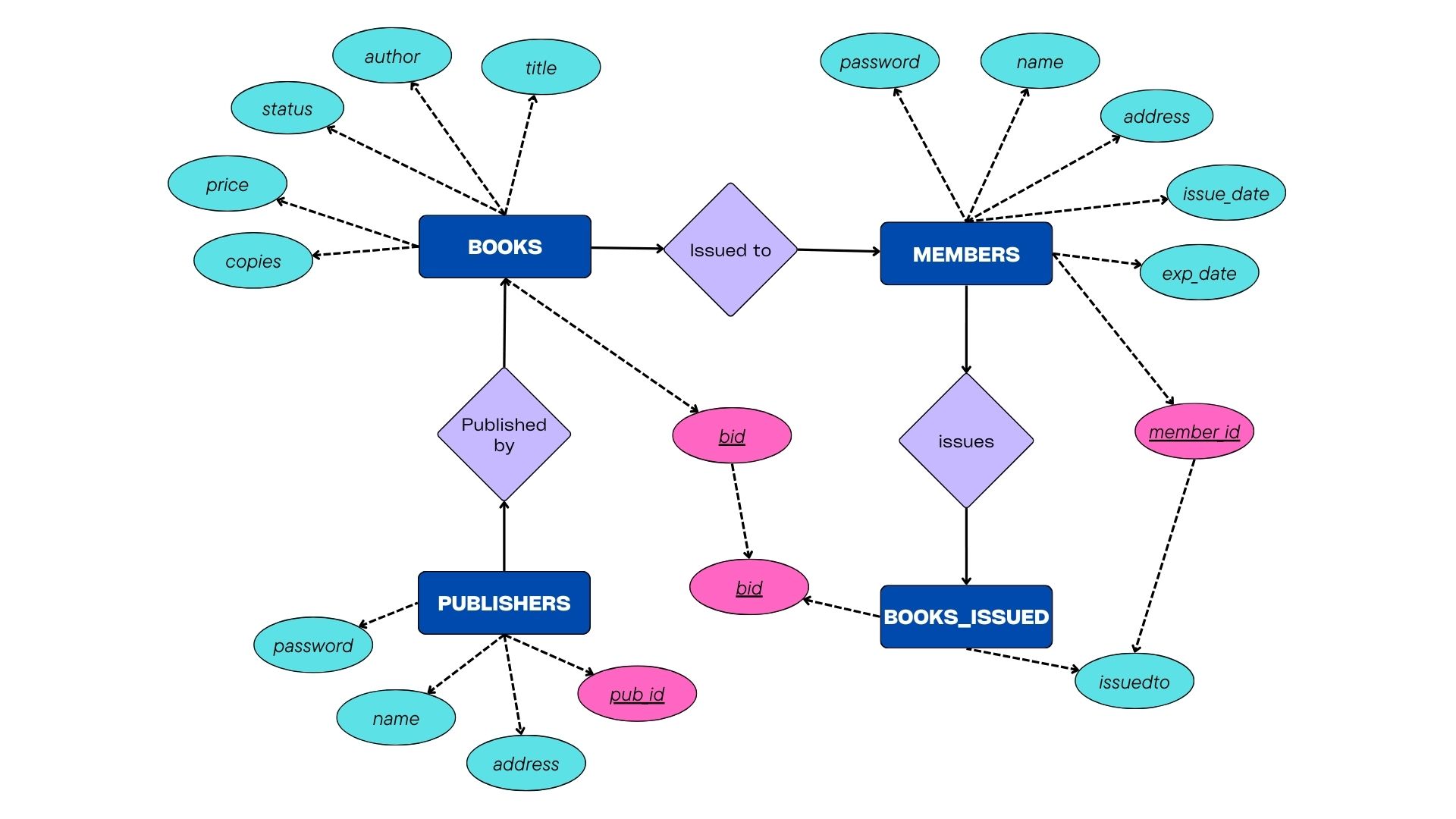
* Books (books) → bid (Primary Key)
* Users (members) → member\_id (Primary Key)
* Transactions (books\_issued) → (bid, issuedto) (Composite Primary Key)
* Publishers (publishers) → pub\_id (Primary Key)

**Foreign Keys (FK):**

* Transactions (books\_issued)
  + bid → References books(bid)
  + issuedto → References members(member\_id)



**RELATIONAL SCHEMA**



**ER DIAGRAM**