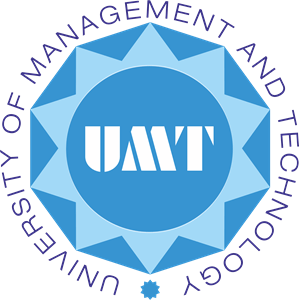
**DATABASE MANAGEMENT – Lab**

**BS(AI) 4th Semester**

**FINAL TERM PROJECT REPORT**

****

**Submitted By:**

**M khalid ur rehman**

**Submitted To:**

**Zeenat tanveer**

## 1. Project Title

**Library Management System using SQL**

## 2. Introduction

A library is a curated collection of sources of information and similar resources, made accessible to a defined community. In educational institutes and universities, libraries are a vital component of academic and student life.

This project aims to build a **Library Management System** using SQL that handles the recording, retrieval, and management of data concerning books, authors, members, and book issuance in a library.

The system includes:

* Managing records for books and their authors.
* Registering members.
* Issuing and returning books.
* Keeping track of book availability.
* Generating useful reports and queries.

## 3. Objectives

* To create a relational database for managing a library's data.
* To maintain book and author details.
* To record member details and memberships.
* To manage the issuing and returning of books.
* To generate reports such as overdue books, most borrowed books, and member borrowing history.
* To ensure data integrity using foreign keys and constraints.

## 4. Tools Used

* **SQL (MySQL / PostgreSQL / SQLite)** – Primary database language.
* **DBMS Concepts** – Relationships, constraints, keys.
* **Optional Tools** – DBeaver, phpMyAdmin, MySQL Workbench (for interface and visualization).

## 5. Entity Descriptions

* **Books**: Represents the physical or digital books available in the library.
* **Authors**: Represents the writers of the books.
* **Members**: Users or patrons who borrow books.
* **Issued\_Books**: Represents the book lending records.

## 6. Entity Relationship Diagram (ERD) – Description

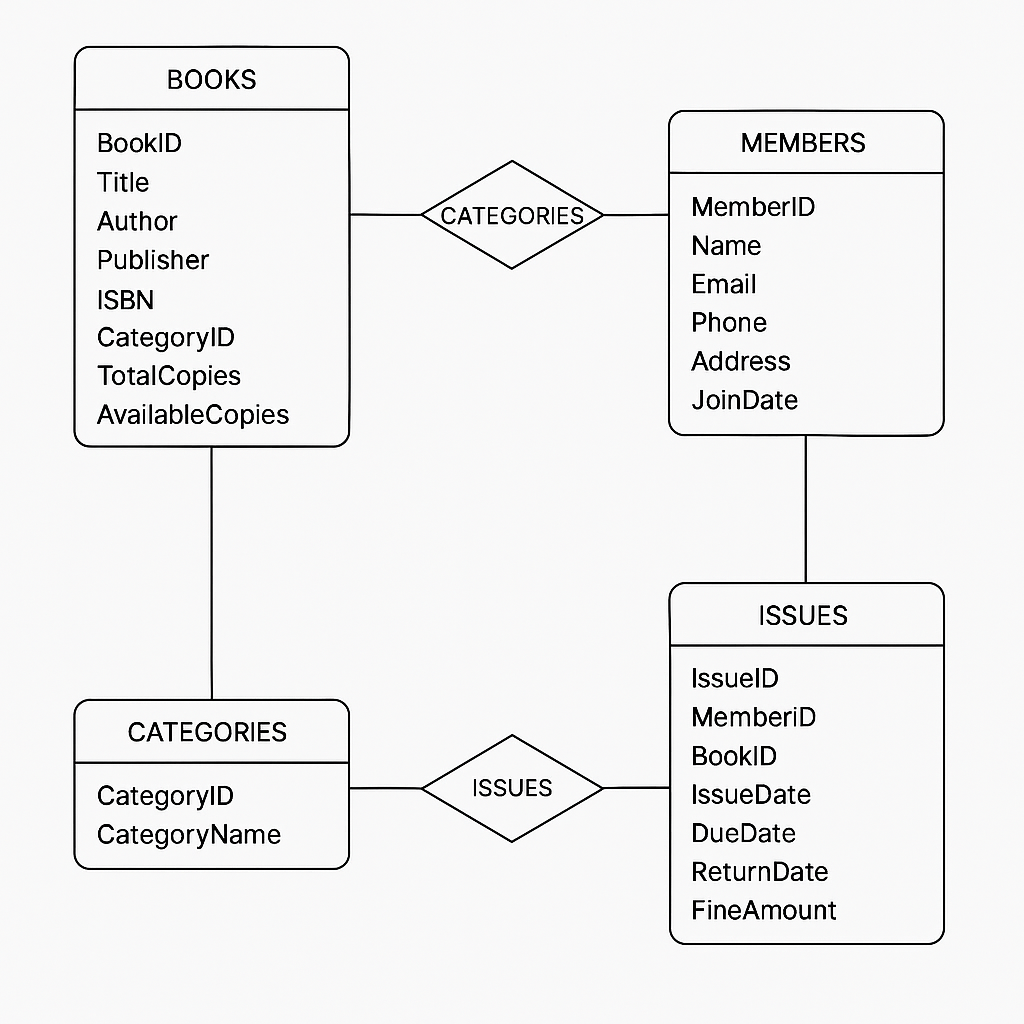
text

CopyEdit

Authors (author\_id) ←── Books (book\_id) ──→ Issued\_Books (issue\_id) ←── Members (member\_id)

### Relationships:

* One **author** can write many **books**.
* One **book** can be issued to many **members** (historically).
* One **member** can borrow many **books**.

****>**DIAGRAM**

## 7. Table Descriptions

### 7.1 Authors

| **Column** | **Data Type** | **Description** |
| --- | --- | --- |
| author\_id | INT | Unique ID for the author |
| name | VARCHAR | Name of the author |
| country | VARCHAR | Country of origin |

### 7.2 Books

| **Column** | **Data Type** | **Description** |
| --- | --- | --- |
| book\_id | INT | Unique book ID |
| title | VARCHAR | Title of the book |
| author\_id | INT | Foreign key from Authors |
| genre | VARCHAR | Genre of the book |
| total\_copies | INT | Total number of copies owned |
| available\_copies | INT | Currently available copies |

### 7.3 Members

| **Column** | **Data Type** | **Description** |
| --- | --- | --- |
| member\_id | INT | Unique member ID |
| name | VARCHAR | Member’s full name |
| email | VARCHAR | Contact email |
| phone | VARCHAR | Phone number |
| membership\_date | DATE | Date of joining |

### 7.4 Issued\_Books

| **Column** | **Data Type** | **Description** |
| --- | --- | --- |
| issue\_id | INT | Unique issue transaction ID |
| book\_id | INT | Foreign key to Books |
| member\_id | INT | Foreign key to Members |
| issue\_date | DATE | Date when book was issued |
| due\_date | DATE | Expected return date |
| return\_date | DATE | Actual return date (nullable) |

## 8. SQL Functional Workflow

1. **Insert Authors**
   * Populate the Authors table with well-known writers.
2. **Insert Books**
   * Add books to the catalog and link to authors via author\_id.
3. **Add Members**
   * Register library users with email, name, and phone.
4. **Issue Books**
   * Insert a record into Issued\_Books.
   * Decrease available\_copies in Books.
5. **Return Books**
   * Update return\_date in Issued\_Books.
   * Increase available\_copies.

## 9. Key SQL Queries (With Explanation)

### 9.1 View All Books

sql

CopyEdit

SELECT \* FROM Books;

**Purpose**: List of all books in the library.

### 9.2 View Available Books

sql

CopyEdit

SELECT \* FROM Books WHERE available\_copies > 0;

**Purpose**: Shows only books that can currently be borrowed.

### 9.3 View Issued Books

sql

CopyEdit

SELECT b.title, m.name, i.issue\_date, i.due\_date

FROM Issued\_Books i

JOIN Books b ON i.book\_id = b.book\_id

JOIN Members m ON i.member\_id = m.member\_id

WHERE i.return\_date IS NULL;

**Purpose**: Displays which books are currently borrowed, and by whom.

### 9.4 Overdue Books

sql

CopyEdit

SELECT b.title, m.name, i.due\_date

FROM Issued\_Books i

JOIN Books b ON i.book\_id = b.book\_id

JOIN Members m ON i.member\_id = m.member\_id

WHERE i.return\_date IS NULL AND i.due\_date < CURRENT\_DATE;

**Purpose**: Lists all overdue books not yet returned.

### 9.5 Member Borrowing History

sql

CopyEdit

SELECT b.title, i.issue\_date, i.return\_date

FROM Issued\_Books i

JOIN Books b ON i.book\_id = b.book\_id

WHERE i.member\_id = 201;

**Purpose**: Shows the full borrow history for a specific member.

## 10. Sample Output (Example)

### View of Current Issued Books

| **issue\_id** | **title** | **member\_name** | **issue\_date** | **due\_date** |
| --- | --- | --- | --- | --- |
| 301 | Harry Potter... | Alice Johnson | 2025-06-10 | 2025-06-20 |
| 302 | 1984 | Bob Smith | 2025-06-15 | 2025-06-25 |

### Available Books Output

| **book\_id** | **title** | **available\_copies** |
| --- | --- | --- |
| 103 | Pride and Prejudice | 8 |
| 104 | Huckleberry Finn | 6 |

## 11. Enhancements & Future Scope

* Add **categories** or **sub-genres** of books.
* Implement **book reservation** system.
* Include **fine calculation** for late returns.
* Build a **web or mobile interface** using PHP, Python, or Java.
* Allow **admin roles** for librarians to approve or reject membership.

## 12. Benefits of This System

* Efficient tracking of book issues and returns.
* Easy to scale and maintain.
* Simple UI can be built on top of this logic.
* Prevents over-borrowing of unavailable books.
* Ensures referential integrity using foreign keys.

## 13. Security Considerations

* Use constraints to prevent duplicate or invalid data.
* Sanitize inputs if building a web interface.
* Use user roles and authentication for sensitive actions (like delete/update).

## 14. Conclusion

This Library Management System in SQL successfully simulates the basic functionality required to run a physical or digital library. It is a relational database system supporting multiple operations like inserting book records, issuing books to members, updating return dates, and running advanced queries to generate reports. It is scalable and forms a solid base for any front-end integration using web or desktop technologies.

This project serves as a practical application of DBMS concepts like normalization, joins, primary & foreign keys, and real-time data manipulation.