Library Management System - Database Design Document

Introduction

The Library Management System (LMS) is a comprehensive database solution designed to automate and streamline library operations. The system is structured to efficiently manage books, members, staff, and various library processes while ensuring data accuracy, integrity, and security. By leveraging this system, libraries can modernize their administrative functions, enhance user experience, and make data-driven decisions to improve their services.

Purpose

The primary objectives of the Library Management System are:

- Automating routine library operations such as book lending, returns, and inventory management.
- Maintaining accurate records of books, members, and transactions.
- Enabling efficient book circulation management.
- Facilitating seamless tracking of library resources and availability.
- Supporting data-driven decision-making for library administration.
- Improving the user experience for both staff and library members through a structured and intuitive system.

Scope

The system encompasses a variety of core functionalities, including:

- **Book Inventory Management**: Tracking book details, availability, and conditions.
- **Member Registration and Management**: Managing user accounts, borrowing privileges, and status.
- **Staff Operations Tracking**: Monitoring staff responsibilities and activities.
- **Transaction Processing**: Handling book borrowing and return transactions.
- **Reservation Handling**: Managing book reservations and waitlists.
- **Fine Calculation and Tracking**: Automating penalties for overdue books.
- **Feedback Collection and Management**: Allowing users to review and provide feedback on books.
- **Publisher and Author Information Management**: Keeping track of book publishers and authors.

Target Users

The system is designed for the following user groups:

Library Staff

- **Librarians**: Oversee library operations, manage book inventory, and assist members.
- **Administrative Staff**: Handle membership, fines, and feedback management.
- **Management**: Generate reports, oversee library efficiency, and make strategic decisions.

Library Members

- **Regular Patrons**: General users who borrow books and utilize library services.
- **Students**: Educational users requiring academic resources and study materials.
- **Faculty**: Professors and educators utilizing resources for teaching and research.
- **Public Users**: General users who access public library services.

---

1. Business Problems Addressed

The Library Management System is designed to address the following challenges:

- **Tracking Physical Book Inventory**: Ensuring accurate records of book copies and availability.
- **Managing Member Registrations and Borrowing Activities**: Handling user accounts, lending history, and borrowing privileges.
- **Handling Staff Operations**: Assigning responsibilities and tracking staff-managed operations.
- **Processing Reservations and Waitlists**: Allowing users to reserve books and managing availability.
- **Maintaining Comprehensive Book Metadata**: Tracking book details, authors, publishers, and genres.
- **Tracking Book Copies and Conditions**: Ensuring the proper maintenance of books.
- **Collecting and Managing User Feedback**: Gathering member reviews and ratings for books.
- **Calculating and Managing Fines**: Automatically calculating penalties for overdue returns.
- **Generating Reports**: Providing insights on library usage, book popularity, and member activity.

---

2. Entity Details and Relationships

The Library Management System comprises the following primary entities and their relationships:

1. LIBRARY

- **Purpose**: Represents the central institution managing the library system.
- **Attributes**: `library_id`, `name`, `location`.
- **Relationships**:
    - Has multiple MEMBERS (N:M)
    - Employs multiple STAFF (1:N)
    - Contains multiple BOOKS (1:N)
    - Manages overall operations (1:N)

2. MEMBER

- **Purpose**: Tracks library patrons.
- **Attributes**: `member_id`, `first_name`, `last_name`, `email`, `phone`, `join_date`, `status`.
- **Relationships**:
    - Belongs to multiple LIBRARIES (M:N)
    - Engages in multiple TRANSACTIONS (1:N)
    - Creates multiple RESERVATIONS (1:N)
    - Provides multiple FEEDBACK entries (1:N)

3. BOOK

- **Purpose**: Stores book metadata.
- **Attributes**: `book_id`, `library_id`, `title`, `ISBN`, `category_id`, `publisher_id`, `publication_date`, `availability_status.`
- **Relationships**:
    - Belongs to one LIBRARY (N:1)
    - Has multiple COPIES (1:N)
    - Has multiple AUTHORS (M:N through BOOK_AUTHOR)
    - Published by one PUBLISHER (N:1)
    - Belongs to one GENRE (N:1)
    - Receives multiple FEEDBACK entries (1:N)

3.1. BOOK_AUTHOR (Associative Entity)

- **Purpose**: Manages many-to-many relationship between books and authors.
- **Attributes**:book_id, author_id.
- **Relationships**:
    - Links BOOK to AUTHOR (M:N)

4. AUTHOR

- **Purpose**: Stores publisher details.
- **Attributes**:author_id, first_name, last_name.
- **Relationships**:
    - Writes multiple BOOKS (M:N through BOOK_AUTHOR).

5. PUBLISHER

- **Purpose**: Stores publisher details.
- **Attributes**: publisher_id, name, address, mobile_number, email.
- **Relationships**:
    - Publishes multiple BOOKS (1:N)

6. COPY

- **Purpose**: Represents individual physical copies of books.
- **Attributes**: copy_id, book_id, condition, purchase_date.
- **Relationships**:
    - Belongs to one BOOK (N:1)
    - Involved in multiple TRANSACTIONS (1:N)

7. STAFF

- **Purpose**: Manages library operations.
- **Attributes**: staff_id, library_id, first_name, last_name, role, email, phone.
- **Relationships**:
    - Works for one LIBRARY (N:1)
    - Handles multiple TRANSACTIONS (1:N)
    - Manages multiple RESERVATIONS (1:N)

8. TRANSACTION

- **Purpose**: Records book borrowing and returning activities.
- **Attributes**: `transaction_id`, `member_id`, `copy_id`, `staff_id`, `borrow_date`, `due_date`, `return_date`, `transaction_status`.
- **Relationships**:
  - Made by a `MEMBER` (N:1)
  - Involves a `COPY` (N:1)
  - Managed by `STAFF` (N:1)
  - Associated with `FINES` (1:1)

9. RESERVATION

- **Purpose**: Handles book reservations.
- **Attributes**: `reservation_id`, `member_id`, `book_id`, `staff_id`, `reservation_date`, `reservation_status`.
- **Relationships**:
  - Created by a `MEMBER` (N:1)
  - Associated with a `BOOK` (N:1)
  - Handled by `STAFF` (N:1)

10. FINE

- **Purpose**: Tracks penalties for overdue books.
- **Attributes**: `fine_id`, `transaction_id`, `amount`, `issue_date`, `status`.
- **Relationships**:
  - Associated with a `TRANSACTION` (1:1)

11. FEEDBACK

- **Purpose**: Collects user ratings and reviews.
- **Attributes**: `feedback_id`, `book_id`, `member_id`, `rating`, `comment`, `feedback_date`.
- **Relationships**:
  - Given by a `MEMBER` (N:1)
  - Related to a `BOOK` (N:1)

12. GENRE

- **Purpose**: Categorizes books.
- **Attributes**: `genre_id`, `genre_type`, `description`.
- **Relationships**:
  - Contains multiple `BOOKS`. (1:N)

---

3. Key Design Decisions

Relationship Implementation

- BOOK_AUTHOR implemented as a junction table to properly handle many-to-many relationships.
- Direct relationships established between LIBRARY and other entities (BOOK, STAFF, MEMBER).
- Hierarchical relationship structure maintained through proper foreign key implementations.

Status Tracking

- Enhanced status tracking for COPY entity to monitor book availability.
- Comprehensive date tracking across all temporal operations.
- Status fields added to relevant entities for state management.

Library Management

- Centralized LIBRARY entity with enhanced attributes.
- Clear hierarchical structure for staff management.
- Comprehensive tracking of book ownership and location.

Data Integrity

- Foreign key constraints ensure referential integrity.
- Unique constraints on relevant fields (ISBN, member email, etc.)
- Proper handling of composite relationships through junction tables.

---

4. Implementation Considerations

Cardinality Enforcement

- One-to-many relationships implemented through foreign keys.
- Many-to-many relationships handled through junction tables.
- Mandatory relationships enforced through NOT NULL constraints.

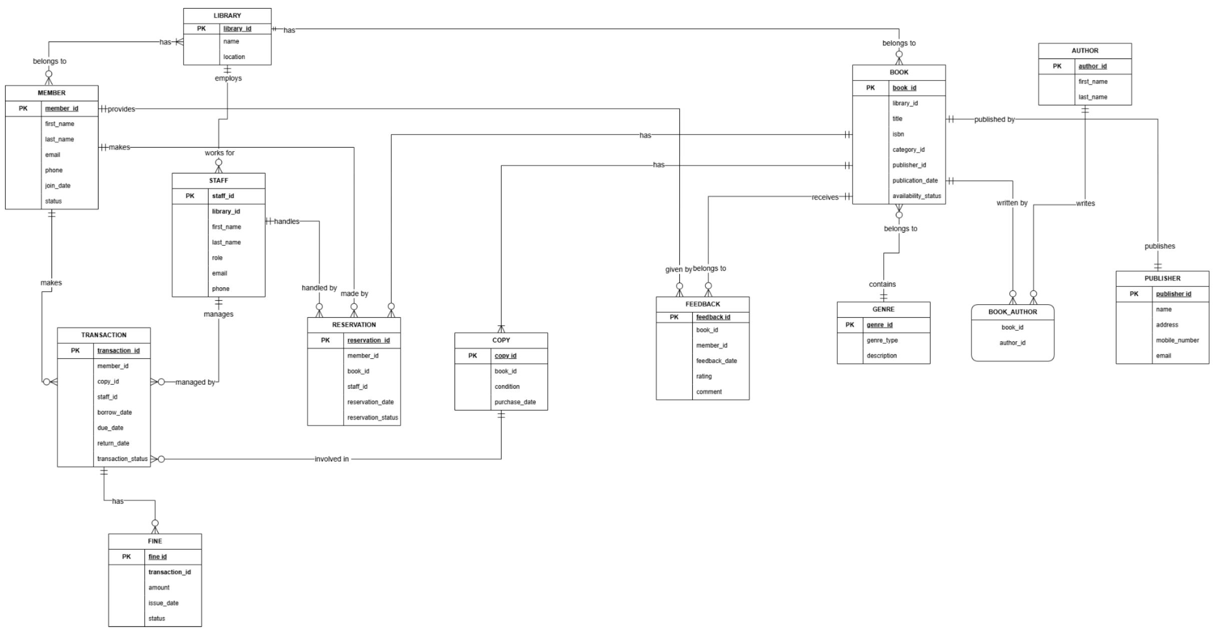
Data Consistency

- Transaction management for related operations.
- Cascading updates/deletes where appropriate.
- Status consistency maintained across related entities.

Assumptions:

1. Copy_id is unique and can be used to trace back multiple copies of a single book_id.

Entity-Relation Diagram:



[For Better Viewing Use This Link – https://shorturl.at/Nljgw (Page 1)]

Group 10 (Team Members):

1) Name: Rutwik Ganagi
Email: ganagi.r@northeastern.edu

2) Name: Sachin Vishaul Baskar
Email: baskar.sa@northeastern.edu

3) Name: Ashwin Badamikar
Email: badamikar.a@northeastern.edu

4) Name: Dennis sharon
Email: cheruvathoorshaj.d@northeastern.edu

5) Name: Chetan Warad
Email: warad.c@northeastern.edu

Github URL - https://github.com/rutwikganagi2000/DMDD-Group-10