Library Management System Database Design Document

C.1. BUSINESS RULES AND ASSUMPTIONS

1. The user should be able to **create an account** with username, email, and password. On creation of every user account, a **unique identifier** will be automatically generated to further identify the user in the application.

2. Each user will have a **timestamp** of when the account was created.

3. A user should be able to **borrow books** from the library system. Each borrowing transaction will generate a **unique record ID**.

4. When borrowing a book, the system will record the **borrowing date** and **expected return date.** The **actual return date** will be recorded when the book is returned.

5. A user should be able to **like books**. Each like will be uniquely identified and associated with both a user and a book.

6. Users can **comment on books**. Each comment will have a **unique identifier** and will contain the comment text.

7. Users can make **reservations** for books that are currently unavailable. Each reservation will have a **unique identifier** and an expiry date.

8. Books will have attributes including book name, authors, publishers, publication date, number of copies available, and ISBN.

9. Each book will be assigned a **unique identifier** for ease of tracking in the system.

10. Books can belong to **one or multiple categories**. Categories are identified by a **unique category ID.**

11. Categories have **names** that describe the genre or subject matter of the books.

12. An a**dministrator** can manage the book inventory and user accounts in the system.

13. Administrators have their own accounts with **unique identifiers**, names, emails, and passwords.

14. The system will track which books **are borrowed** by which users.

15. The system will keep records of which books **are liked** by which users.

16. The system allows **multiple copies** of the same book to be tracked separately for borrowing purposes.

17. A book's **availability** for borrowing is determined by the number of copies available.

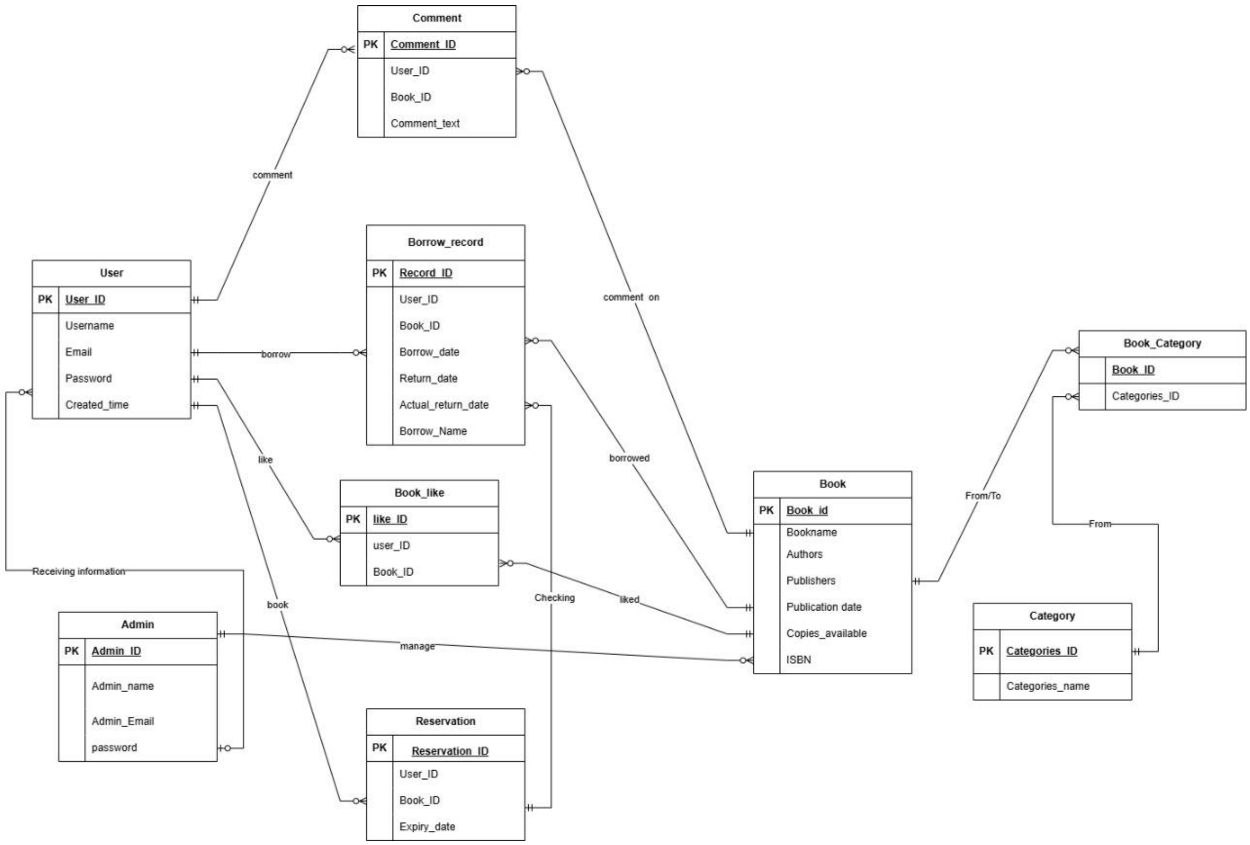
18. The system will **enforce return dates** for borrowed books.

19. Books can be **reserved in advance** if all copies are currently borrowed.

20. The system allows users to **search for books** by various attributes such as title, author, and category.

s

C.2. ENTITY RELATIONSHIP DIAGRAM (ERD)



C.3. RELATIONS (LOGICAL DESIGN / SCHEMA CONVERSION)

user (User_ID, Username, Email, Password, Created_time)

admin (Admin_ID, Admin_name, Admin_Email, password)

book (Book_id, Bookname, Authors, Publishers, Publication_date, Copies_available, ISBN)

borrow_record (Record_ID, User_ID, Book_ID*, Borrow_date, Return_date, Actual_return_date, Borrow_Name)*

User_ID references user

Book_ID references book

book_like (like_ID, user_ID, Book_ID*)*

user_ID references user

Book_ID references book

comment (Comment_ID, User_ID, Book_ID*, Comment_text)*

User_ID references user

Book_ID references book

reservation (Reservation_ID, User_ID, Book_ID*, Expiry_date)*

User_ID references user

Book_ID references book

category (Categories_ID, Categories_name)

book_category (Book_ID, Categories_ID*)*

Book_ID references book

Categories_ID references category

Foreign key references are indicated with an asterisk ()

C.4. LIST OF FUNCTIONAL DEPENDENCIES RELATED TO EACH BUSINESS RULE

USER

BR1: The user should be able to create an account with username, email, and password. On creation of every user account, a unique identifier will be automatically generated.

- User_ID → {Username, Email, Password, Created_time}

BR2: Each user will have a timestamp of when the account was created.

- User_ID → Created_time

Additional functional dependencies:

- Email → {User_ID, Username, Password, Created_time} (assuming email must be unique)

BOOK

BR8: Books will have attributes including book name, authors, publishers, publication date, number of copies available, and ISBN.

- Book_id → {Bookname, Authors, Publishers, Publication_date, Copies_available, ISBN}

BR9: Each book will be assigned a unique identifier for ease of tracking in the system.

- Book_id → {Bookname, Authors, Publishers, Publication_date, Copies_available, ISBN}

Additional functional dependencies:

- ISBN → {Book_id, Bookname, Authors, Publishers, Publication_date, Copies_available} (assuming ISBN must be unique)

BORROW_RECORD

BR3: A user should be able to borrow books from the library system. Each borrowing transaction will generate a unique record ID.

- Record_ID → {User_ID, Book_ID, Borrow_date, Return_date, Actual_return_date, Borrow_Name}

BR4: When borrowing a book, the system will record the borrowing date and expected return date. The actual return date will be recorded when the book is returned.

- Record_ID → {Borrow_date, Return_date, Actual_return_date}

BR14: The system will track which books are borrowed by which users.

- Record_ID → {User_ID, Book_ID}

BOOK_LIKE

BR5: A user should be able to like books. Each like will be uniquely identified and associated with both a user and a book.

- like_ID → {user_ID, Book_ID}

BR15: The system will keep records of which books are liked by which users.

- As Book_like is a weak associative entity with a generated primary key, we have:
- like_ID → {user_ID, Book_ID}

COMMENT

BR6: Users can comment on books. Each comment will have a unique identifier and will contain the comment text.

- Comment_ID → {User_ID, Book_ID, Comment_text}

RESERVATION

BR7: Users can make reservations for books that are currently unavailable. Each reservation will have a unique identifier and an expiry date.

- Reservation_ID → {User_ID, Book_ID, Expiry_date}

BR19: Books can be reserved in advance if all copies are currently borrowed.

- Reservation_ID → {User_ID, Book_ID, Expiry_date}

ADMIN

BR12: An administrator can manage the book inventory and user accounts in the system.

- Admin_ID → {Admin_name, Admin_Email, password}

BR13: Administrators have their own accounts with unique identifiers, names, emails, and passwords.

- Admin_ID → {Admin_name, Admin_Email, password}

- Admin_Email → {Admin_ID, Admin_name, password} (assuming admin email must be unique)

CATEGORY

BR10: Books can belong to one or multiple categories. Categories are identified by a unique category ID.

- Categories_ID → Categories_name

BR11: Categories have names that describe the genre or subject matter of the books.

- Categories_ID → Categories_name
- Categories_name → Categories_ID (assuming category names must be unique)

BOOK_CATEGORY

BR10: Books can belong to one or multiple categories. Categories are identified by a unique category ID.

- As Book_Category is a weak associative entity with no additional attributes, there is no FD aside from and to itself.
- {Book_ID, Categories_ID} → {Book_ID, Categories_ID}

C.5. NORMALIZATION (LOGICAL DESIGN)

User entity

- User_ID → {Username, Email, Password, Created_time}
- Email → {User_ID, Username, Password, Created_time}

1NF: Yes, it has a unique identifier for each row (User_ID) and no repeating attributes. 2NF: Yes, every non-key attribute is functionally dependent on the PK (User_ID). 3NF: Yes, there are no transitive functional dependencies. BCNF: Yes, for two reasons:

1. Even though there is a functional dependency not from the primary key (Email), the determinator is still a superkey. Email is a superkey for the User entity as there cannot be two users with the same email address.
2. When we consider the functional dependency with the determinant Email, it is a trivial dependency.

Therefore, the final set of relation User is: User (User_ID, Username, Email, Password, Created_time)

Book entity

- Book_id → {Bookname, Authors, Publishers, Publication_date, Copies_available, ISBN}
- ISBN → {Book_id, Bookname, Authors, Publishers, Publication_date, Copies_available}

1NF: Yes, it has a unique identifier for each row (Book_id) and no repeating attributes. 2NF: Yes, every non-key attribute is functionally dependent on the PK (Book_id). 3NF: Yes, there are no transitive functional dependencies. BCNF: Yes, for two reasons:

1. Even though there is a functional dependency not from the primary key (ISBN), the determinator is still a superkey. ISBN is a superkey for the Book entity as there cannot be two books with the same ISBN.
2. When we consider the functional dependency with the determinant ISBN, it is a trivial dependency.

Therefore, the final set of relation Book is: Book (Book_id, Bookname, Authors, Publishers, Publication_date, Copies_available, ISBN)

Borrow_record entity

- Record_ID → {User_ID, Book_ID, Borrow_date, Return_date, Actual_return_date, Borrow_Name}

1NF: Yes, it has a unique identifier for each row (Record_ID) and no repeating attributes. 2NF: Yes, every non-key attribute is functionally dependent on the PK (Record_ID). 3NF: Yes, there are no transitive functional dependencies. BCNF: Yes, there are no other functional dependencies not from the primary key on the Borrow_record entity.

Therefore, the final set of relation Borrow_record is: Borrow_record (Record_ID, User_ID*, Book_ID*, Borrow_date, Return_date, Actual_return_date, Borrow_Name)

- User_ID references User
- Book_ID references Book

Book_like entity

- like_ID → {user_ID, Book_ID}

1NF: Yes, it has a unique identifier for each row (like_ID) and no repeating attributes. 2NF: Yes, every non-key attribute is functionally dependent on the PK (like_ID). 3NF: Yes, there are no transitive functional dependencies. BCNF: Yes, there are no other functional dependencies not from the primary key on the Book_like entity.

Therefore, the final set of relation Book_like is: Book_like (like_ID, user_ID*, Book_ID*)

- user_ID references User
- Book_ID references Book

Comment entity

- Comment_ID → {User_ID, Book_ID, Comment_text}

1NF: Yes, it has a unique identifier for each row (Comment_ID) and no repeating attributes. 2NF: Yes, every non-key attribute is functionally dependent on the PK (Comment_ID). 3NF: Yes, there are no transitive functional dependencies. BCNF: Yes, there are no other functional dependencies not from the primary key on the Comment entity.

Therefore, the final set of relation Comment is: Comment (Comment_ID, User_ID*, Book_ID*, Comment_text)

- User_ID references User
- Book_ID references Book

Reservation entity

- Reservation_ID → {User_ID, Book_ID, Expiry_date}

1NF: Yes, it has a unique identifier for each row (Reservation_ID) and no repeating attributes. 2NF: Yes, every non-key attribute is functionally dependent on the PK (Reservation_ID). 3NF: Yes, there are no transitive functional dependencies. BCNF: Yes, there are no other functional dependencies not from the primary key on the Reservation entity.

Therefore, the final set of relation Reservation is: Reservation (Reservation_ID, User_ID*, Book_ID*, Expiry_date)

- User_ID references User
- Book_ID references Book

Admin entity

- Admin_ID → {Admin_name, Admin_Email, password}
- Admin_Email → {Admin_ID, Admin_name, password}

1NF: Yes, it has a unique identifier for each row (Admin_ID) and no repeating attributes. 2NF: Yes, every non-key attribute is functionally dependent on the PK (Admin_ID). 3NF: Yes, there are no transitive functional dependencies. BCNF: Yes, for two reasons:

1. Even though there is a functional dependency not from the primary key (Admin_Email), the determinator is still a superkey. Admin_Email is a superkey for the Admin entity as there cannot be two admins with the same email address.
2. When we consider the functional dependency with the determinant Admin_Email, it is a trivial dependency.

Therefore, the final set of relation Admin is: Admin (Admin_ID, Admin_name, Admin_Email, password)

Category entity

- Categories_ID → Categories_name
- Categories_name → Categories_ID

1NF: Yes, it has a unique identifier for each row (Categories_ID) and no repeating attributes. 2NF: Yes, every non-key attribute is functionally dependent on the PK (Categories_ID). 3NF: Yes, there are no transitive functional dependencies. BCNF: Yes, for two reasons:

1. Even though there is a functional dependency not from the primary key (Categories_name), the determinator is still a superkey. Categories_name is a superkey for the Category entity as there cannot be two categories with the same name.
2. When we consider the functional dependency with the determinant Categories_name, it is a trivial dependency.

Therefore, the final set of relation Category is: Category (Categories_ID, Categories_name)

Book_Category entity

- {Book_ID, Categories_ID} → {Book_ID, Categories_ID}

1NF: Yes, it has a unique identifier for each row (Book_ID and Categories_ID) and no repeating attributes. 2NF: Yes, there are no non-key attributes (the entire relation consists of the composite primary key). 3NF: Yes, there are no transitive functional dependencies. BCNF: Yes, there are no other functional dependencies not from the primary key on the Book_Category entity.

Therefore, the final set of relation Book_Category is: Book_Category (Book_ID*, Categories_ID*)

- Book_ID references Book
- Categories_ID references Category