# Library Management System Documentation

**Team Name:** Team No.5

## Team Members and Contributions

* **[Mostafa Ahmed ]** Documentation, query optimization , Business logic implementation
* **[Abdelrahman Ahmed]:** Database design, system testing
* **[Mina Ashraf]:** schema implementation, Trigger development
* **[Nada Saeed]:** database query optimization, ERD creation

## Project Overview

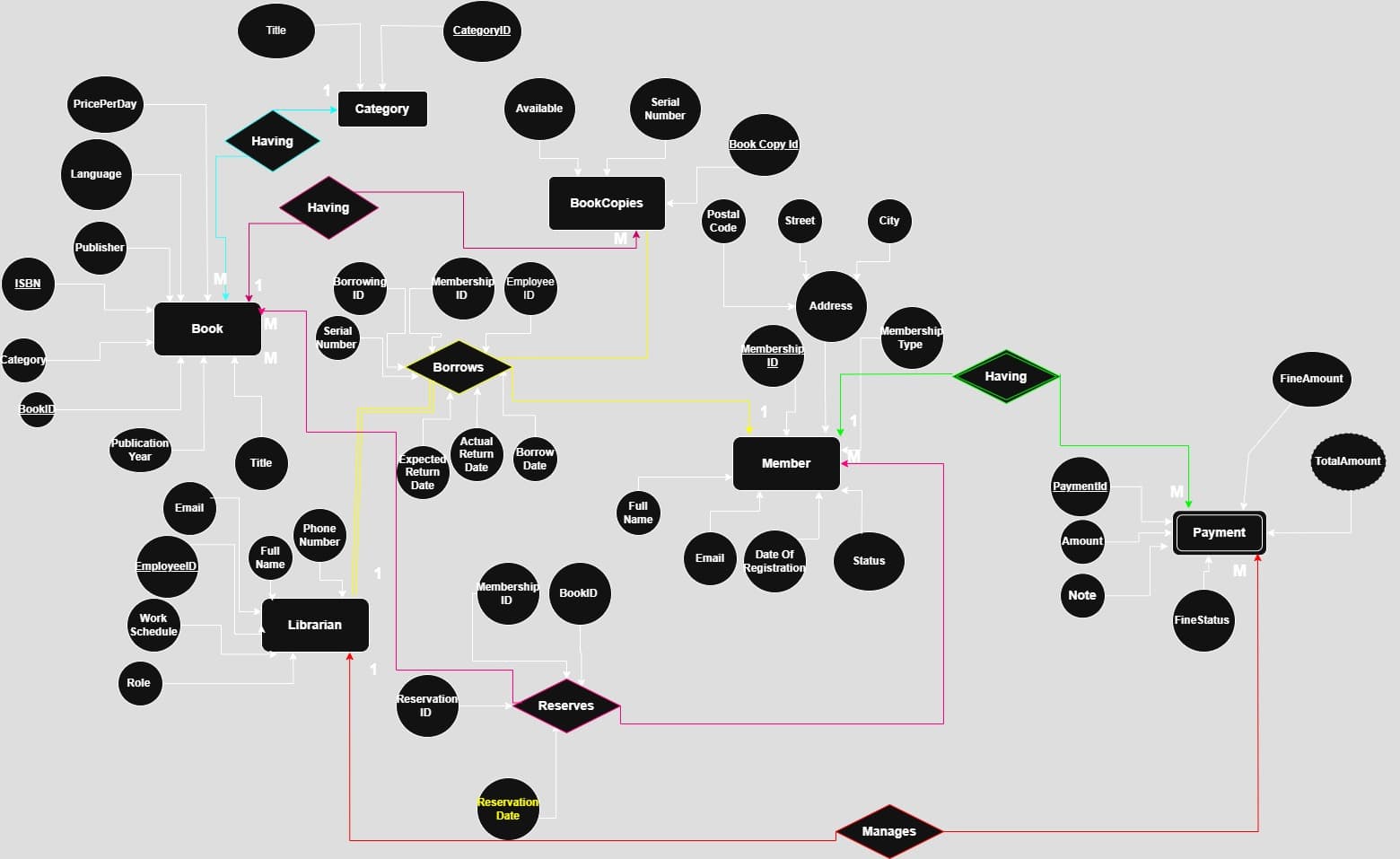
Alexandria Public Library has decided to implement a comprehensive database system to manage their book collection, member information, and borrowing operations. They've hired LibraTech Solutions as database designers to create this system.

The system is designed to track library members, books, book copies, librarians, borrowing transactions, reservations, and payments. It includes functionality for managing the borrowing process, handling overdue returns, automatically fulfilling reservations, and processing payments with late fees when applicable.

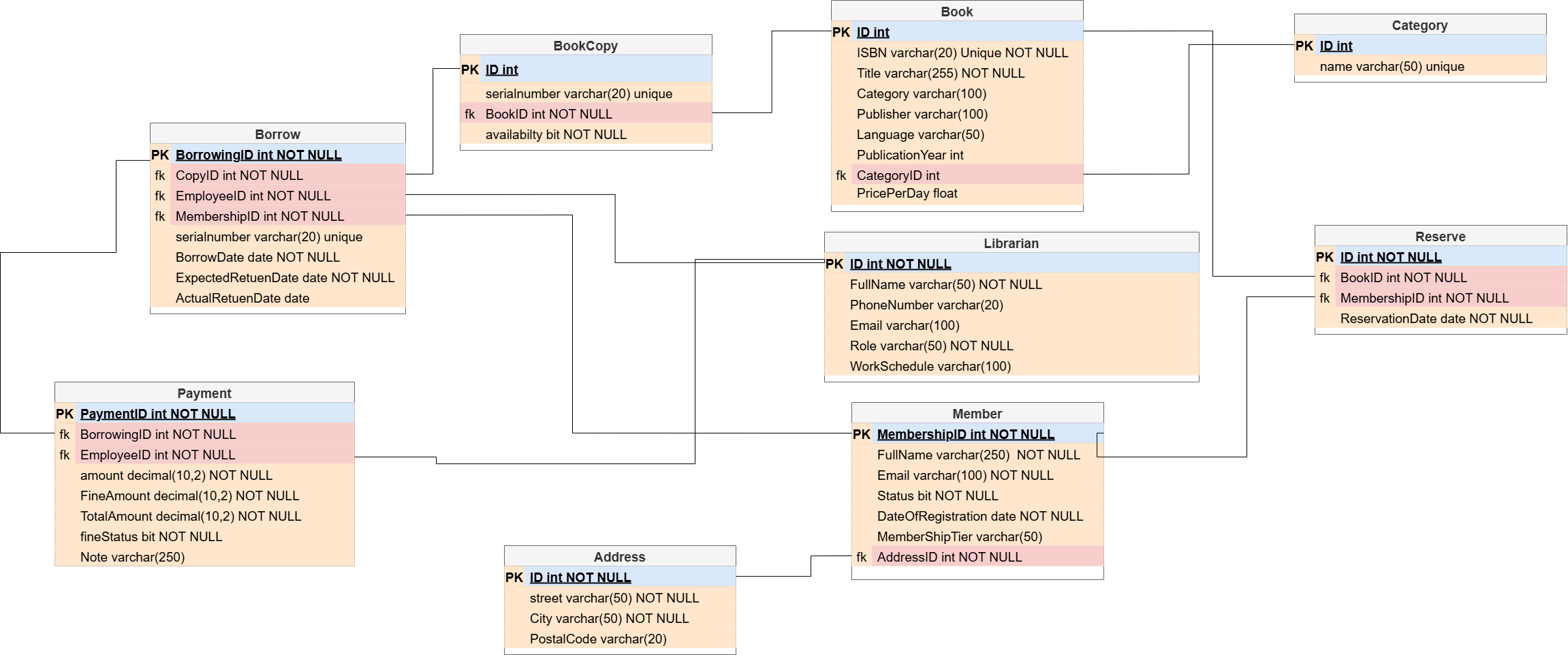
## Business Scenario

* Each member of the library has an ID number, full name, email address, membership status, date of registration, membership type, and a complete address (street, city, postal code).
* Each librarian working at Alexandria has an ID number, full name, phone number, email address, role, and work schedule.
* Each book in the library has an ID number, ISBN, title, category, publisher, language, publication year, and a daily borrowing price.
* Each physical copy of a book has a unique ID, a serial number, and is linked to a specific book title with an availability status.
* Members can borrow multiple book copies, and each book copy can be borrowed by different members over time.
* Each borrowing transaction records which member borrowed which book copy, which librarian processed the transaction, the borrowing date, expected return date, and actual return date when returned.
* Members can reserve books that are currently unavailable, with each reservation tracking the member, book, and reservation date.
* When books are returned late, the system generates payment records with appropriate fees consisting of the borrowing fee plus any late fine amount.
* Each payment record is linked to a specific borrowing transaction and handled by a librarian, with a status indicating whether the payment has been completed.

## Entity-Relationship Diagram (ERD)



## Database Schema



## Business Rules Implementation

### Triggers

1. **Prevent Borrowing with Overdue Books:** Blocks new borrowing transactions if the member has any overdue books.
2. **Mark Copy as Unavailable After Borrowing:** Updates book copy availability status to unavailable when borrowed.
3. **Mark Copy as Available After Return:** Updates book copy availability status to available when returned.
4. **Auto-fulfill Reservation on Book Return:** Automatically assigns returned books to next member in reservation queue.
5. **Create Payment Record for Late Returns:** Generates payment records with fine calculations when books are returned late.
6. **Copy Serial Number to Borrow Record:** Adds book copy serial number to borrowing record for reference.

### Queries

1. **Books Due for Return Soon:** Lists books due for return in the next 7 days to enable proactive reminders.
2. **Overdue Books with Fine Estimates:** Shows overdue books with days overdue and potential fine amounts.
3. **Show Reservation Queue:** Displays waiting list for specific books by reservation date.
4. **Librarian Borrowing Transactions:** Lists all borrowing transactions handled by a specific librarian.
5. **Members with Unpaid Fees:** Lists members with unpaid payments and their complete contact information.
6. **Available Book Copies:** Shows which copies of a particular book are available for borrowing.
7. **Books Borrowed per Member:** Provides statistics on borrowing activity by member.
8. **Books Not Borrowed Recently:** Identifies potentially unpopular books not borrowed in past 6 months.
9. **Most Popular Books:** Identifies most frequently borrowed books for acquisition planning.
10. **New Members:** Lists library members who registered in the current year.
11. **Currently Reserved Books:** Shows books with active reservation requests.
12. **Most Active Librarians:** Identifies librarians who processed the most borrowing transactions.
13. **Late Return Details:** Reports on books returned late for fine assessment.
14. **Inactive Members:** Identifies members who have never borrowed books.
15. **Book Availability Status:** Shows the availability status of all book copies.
16. **Member Details:** Fetches complete member information including address by name.
17. **Book Availability by Title:** Shows availability status of all copies of a specific book.
18. **Member Borrowing History:** Displays the complete borrowing history for a specific member.
19. **Member Account Summary:** Provides a comprehensive overview of a member's account status.
20. **Reservations by Book Title:** Lists all reservations for a specific book title.
21. **Librarian Details by Role:** Fetches librarian information filtered by role.
22. **Category Book Counts:** Shows how many books are in each category.
23. **Empty Categories:** Identifies categories with no books assigned.
24. **Books in Specific Category:** Shows all books in a particular category.
25. **Total Categories Count:** Counts the total number of book categories.
26. **Available Books by Category:** Shows how many available copies exist in each category.
27. **Popular Categories:** Identifies which book categories are most frequently borrowed.
28. **Unpaid Fines List:** Shows all outstanding payments that need to be collected.
29. **Borrowing Revenue:** Calculates total income from basic borrowing fees.
30. **Late Fee Revenue:** Shows total income generated from late return fines.
31. **Highest Value Patrons:** Identifies members with highest total payments.
32. **Member Payment History:** Shows complete payment history for a specific member.
33. **Monthly Revenue Report:** Generates payment totals by month for financial reporting.
34. **High Fine Payments:** Identifies unusually high fine amounts for review.
35. **Librarian Payment Processing:** Shows payments handled by specific librarians.
36. **Category Fine Analysis:** Shows which book categories generate highest fines.
37. **Payment Status Summary:** Provides overview of paid vs unpaid payments.

## Trigger Testing and Validation

### Test Cases for Triggers

1. **Test for Overdue Borrow Prevention**
   * **Purpose:** Verify that members with overdue books cannot borrow additional books
   * **Test Method:** Attempt to create a new borrowing record for a member with overdue books
   * **Expected Result:** Error message preventing the transaction
   * **SQL Example:**

sql

INSERT INTO borrow (member\_id, copy\_id, employee\_id, borrowdate, expectedreturndate)

VALUES (1032, 77, 2012, GETDATE(), DATEADD(DAY, 14, GETDATE()));

1. **Test for Book Availability Status Changes**
   * **Purpose:** Verify that book copy availability is updated correctly after borrowing and returning
   * **Test Method:** Check availability before and after return operation
   * **Expected Result:** Availability changes from 0 to 1 after return
   * **SQL Example:**

sql

SELECT \* FROM bookcopy WHERE id = 11; *-- Should show availability = 0*

UPDATE borrow SET actualreturndate = GETDATE() WHERE copy\_id = 11;

SELECT \* FROM bookcopy WHERE id = 11; *-- Should show availability = 1*

1. **Test for Reservation Fulfillment**
   * **Purpose:** Verify that the reservation system automatically assigns returned books to waiting members
   * **Test Method:** Create a reservation and then return a matching book
   * **Expected Result:** New borrowing record created for the member with reservation
   * **SQL Example:**

sql

INSERT INTO reserve (id, book\_id, member\_id, reservationdate)

VALUES (15, 117, 1037, GETDATE());

UPDATE borrow SET actualreturndate = GETDATE()

WHERE copy\_id IN (SELECT id FROM bookcopy WHERE book\_id = 117);

SELECT \* FROM borrow WHERE member\_id = 1037 ORDER BY borrowdate DESC;

1. **Test for Payment/Fine Creation**
   * **Purpose:** Verify that payment records with appropriate fines are created for late returns
   * **Test Method:** Return a book after its due date and check payment records
   * **Expected Result:** New payment record with calculated fine amounts
   * **SQL Example:**

sql

UPDATE borrow SET actualreturndate = '2025-04-10'

WHERE member\_id = 1030 AND copy\_id = 10;

SELECT \* FROM Payment WHERE BorrowingID IN (

SELECT id FROM borrow WHERE member\_id = 1030 AND copy\_id = 10

);