Database Project Part 2

Advanced Queries, Views, and Stored Procedures

Course: Database Management Systems

Prerequisites: Completion of Database Project Part 1

Topics Covered: Joins, Aggregation Functions, Grouping, Views, Stored Procedures

Building upon the Library Management System you designed in Part 1, you will now implement advanced database operations to extract meaningful insights and automate common tasks.

Section 1: Complex Queries with Joins (40 points)

Write SQL queries to retrieve the following information:

1. Library Book Inventory Report

Display library name, total number of books, number of available books, and number of books currently on loan for each library.

2. Active Borrowers Analysis

List all members who currently have books on loan (status = 'Issued' or 'Overdue'). Show member name, email, book title, loan date, due date, and current status.

3. Overdue Loans with Member Details

Retrieve all overdue loans showing member name, phone number, book title, library name, days overdue (calculated as difference between current date and due date), and any fines paid for that loan.

4. Staff Performance Overview

For each library, show the library name, staff member names, their positions, and count of books managed at that library.

5. Book Popularity Report

Display books that have been loaned at least 3 times. Include book title, ISBN, genre, total number of times loaned, and average review rating (if any reviews exist).

6. Member Reading History

Create a query that shows each member's complete borrowing history including: member name, book titles borrowed (including currently borrowed and previously returned), loan dates, return dates, and any reviews they left for those books.

7. Revenue Analysis by Genre

Calculate total fine payments collected for each book genre. Show genre name, total number of loans for that genre, total fine amount collected, and average fine per loan.

Section 2: Aggregate Functions and Grouping (30 points)

Write queries using aggregate functions and GROUP BY:

8. Monthly Loan Statistics

Generate a report showing the number of loans issued per month for the current year. Include month name, total loans, total returned, and total still issued/overdue.

9. Member Engagement Metrics

For each member, calculate: total books borrowed, total books currently on loan, total fines paid, and average rating they give in reviews. Only include members who have borrowed at least one book.

10. Library Performance Comparison

Compare libraries by showing: library name, total books owned, total active members (members with at least one loan), total revenue from fines, and average books per member.

11. High-Value Books Analysis

Identify books priced above the average book price in their genre. Show book title, genre, price, genre average price, and difference from average.

12. Payment Pattern Analysis

Group payments by payment method and show: payment method, number of transactions, total amount collected, average payment amount, and percentage of total revenue.

Section 3: Views Creation (15 points)

Create the following views:

13. vw_CurrentLoans

A view that shows all currently active loans (status 'Issued' or 'Overdue') with member details, book details, loan information, and calculated days until due (or days overdue).

14. vw_LibraryStatistics

A comprehensive view showing library-level statistics including total books, available books, total members, active loans, total staff, and total revenue from fines.

15. vw_BookDetailsWithReviews

A view combining book information with aggregated review data (average rating, total reviews, latest review date) and current availability status.

Section 4: Stored Procedures (15 points)

Create stored procedures for the following operations:

16. sp_IssueBook

Input Parameters: MemberID, BookID, DueDate

Functionality:

- Check if book is available
- Check if member has any overdue loans
- If validations pass, create a new loan record and update book availability
- Return appropriate success or error message

17. sp_ReturnBook

Input Parameters: LoanID, ReturnDate

Functionality:

- Update loan status to 'Returned' and set return date
- Update book availability to TRUE
- Calculate if there's a fine (e.g., $2 per day overdue)
- If fine exists, automatically create a payment record with 'Pending' status
- Return total fine amount (if any)

18. sp_GetMemberReport

Input Parameters: MemberID

Output: Multiple result sets showing:

- Member basic information
- Current loans (if any)
- Loan history with return status
- Total fines paid and any pending fines
- Reviews written by the member

19. sp_MonthlyLibraryReport

Input Parameters: LibraryID, Month, Year

Output: Comprehensive report showing:

- Total loans issued in that month
- Total books returned in that month
- Total revenue collected
- Most borrowed genre
- Top 3 most active members (by number of loans)

Submission Requirements

1. SQL Script File

A single .sql file containing:

- All 19 queries/views/procedures properly commented
- Each section clearly labeled
- Test data inserts demonstrating that your procedures work correctly

2. Documentation Document

Including:

- Brief explanation of your approach for complex queries (queries 5, 6, 7)
- Screenshots of query results for at least 5 different queries
- Explanation of how your stored procedures handle edge cases

- Any assumptions you made

3. Testing Evidence

For each stored procedure, provide at least 2 test cases:

- One successful execution
- One that demonstrates error handling or validation

Evaluation Criteria

- **Correctness:** Queries return accurate results (40%)
- **Efficiency:** Appropriate use of joins and avoiding unnecessary subqueries (20%)
- **Code Quality:** Proper formatting, naming conventions, and comments (15%)
- **Error Handling:** Stored procedures properly validate inputs and handle errors (15%)
- **Documentation:** Clear explanations and evidence of testing (10%)

Important Notes

- You must use your own database from Part 1
- Ensure you have sufficient test data (at least 3 libraries, 20 books, 10 members, 15 loans)
- All date calculations should work with SQL date functions
- Stored procedures must include proper transaction handling where needed
- Views should not contain hardcoded values

Deadline: [To be announced by instructor]

This project requires understanding of joins, aggregation, grouping, views, and stored procedures.
Simply providing this specification to an AI will not produce a working solution - you must

understand
your database structure from Part 1 and adapt these requirements accordingly.