

Hackathon Submission Template (Level-2-Solution)

Use Case Title:

Library Management System

Student Name:

Manikandan

Register Number:

C2S11661

Institution:

Arulmigu Palaniandavar College of Arts and Culture

Department:

Bachelor of Computer Applications

Date of Submission:

07-04-2025

1. Problem Statement

In traditional libraries, manual record-keeping leads to issues like misplaced books, inefficient tracking, lack of real-time availability, and no fine calculation system for overdue returns. These inefficiencies affect both students and librarians, especially in large institutions.

This project proposes a **Smart Library Management System** that uses a structured SQL database to automate key processes like:

- Managing books and student data

- Handling issue and return of books
- Automatically calculating late return fines
- Requesting books that are out of stock

2. Database Design & Implementation

2.1 Database Creation & Tables

```
-- Create the database
create database mani;
USE mani;

-- Table: Students
CREATE TABLE Students (
    StudentID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(100),
    Email VARCHAR(100) UNIQUE,
    Department VARCHAR(100),
    Year INT
);

-- Table: Authors
CREATE TABLE Authors (
    AuthorID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(100)
);

-- Table: Categories
CREATE TABLE Categories (
    CategoryID INT AUTO_INCREMENT PRIMARY KEY,
    CategoryName VARCHAR(100)
);
```

```
    CategoryName VARCHAR(100)
);

-- Table: Books

CREATE TABLE Books (
    BookID INT AUTO_INCREMENT PRIMARY KEY,
    Title VARCHAR(255),
    AuthorID INT,
    CategoryID INT,
    ISBN VARCHAR(50) UNIQUE,
    Quantity INT CHECK (Quantity >= 0),
    FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID),
    FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID)
);

-- Table: IssuedBooks

CREATE TABLE IssuedBooks (
    IssueID INT AUTO_INCREMENT PRIMARY KEY,
    BookID INT,
    StudentID INT,
    IssueDate DATE,
    DueDate DATE,
    ReturnDate DATE,
    Fine DECIMAL(10,2),
    FOREIGN KEY (BookID) REFERENCES Books(BookID),
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID)
);
```

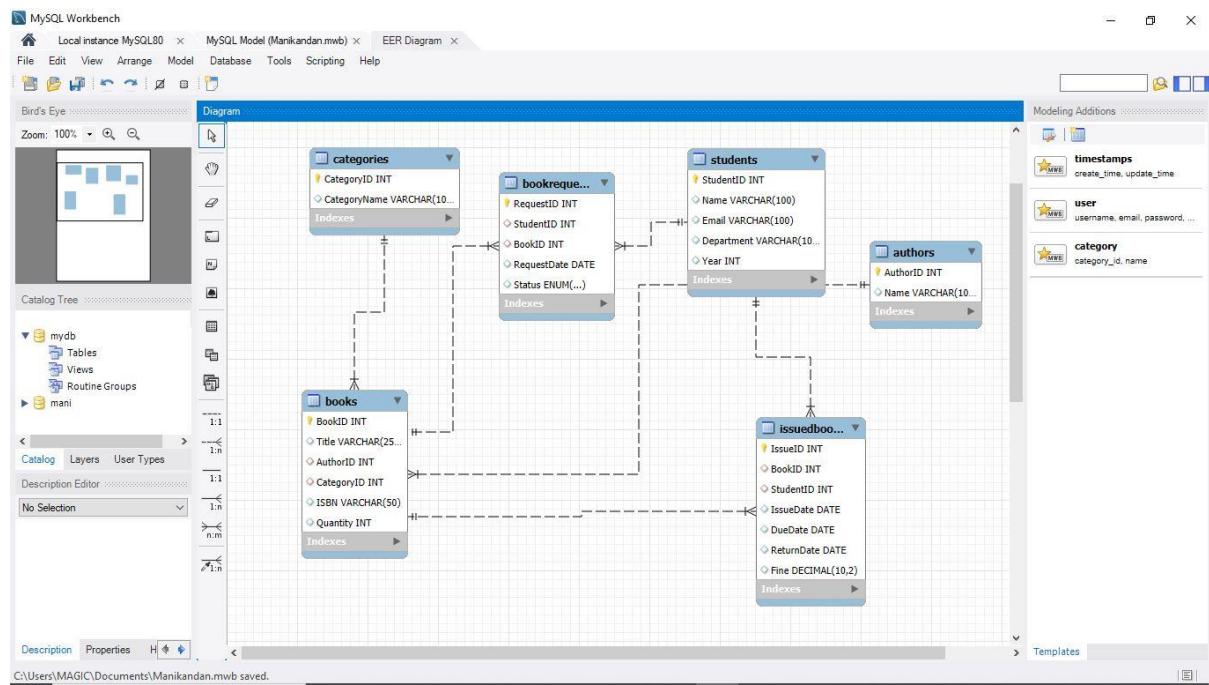
```
-- Table: BookRequests (NEW)

CREATE TABLE BookRequests (
    RequestID INT AUTO_INCREMENT PRIMARY KEY,
    StudentID INT,
    BookID INT,
    RequestDate DATE,
    Status ENUM('Pending', 'Approved', 'Rejected') DEFAULT 'Pending',
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    FOREIGN KEY (BookID) REFERENCES Books(BookID)
);
```

2.2. ER Diagram (Reverse Engineered):

ERD Description:

- Students → IssuedBooks (1:M)
- Books → IssuedBooks (1:M)
- Books → BookRequests (1:M)
- Students → BookRequests (1:M)
- Books ↔ Authors & Categories (M:1)



3. Queries for Data Management

3.1 Insert Sample Data

-- Insert Authors

INSERT INTO Authors (Name) VALUES

('J.K. Rowling'), ('George Orwell'), ('Sundar Raj');

-- Insert Categories

```
INSERT INTO Categories (CategoryName) VALUES
```

('Fiction'), ('Science'), ('History');

-- Insert Books

```
INSERT INTO Books (Title, AuthorID, CategoryID, ISBN, Quantity) VALUES
```

('Harry Potter', 1, 1, 'HP001', 3),

('1984', 2, 1, '1984X', 2),

('The Indian Economy', 3, 3, 'IE2024', 1);

```
-- Insert Students
```

```
INSERT INTO Students (Name, Email, Department, Year) VALUES
```

```
('Arun Kumar', 'arun@example.com', 'CSE', 2),  
('Divya R', 'divya@example.com', 'ECE', 3),  
('Ravi M', 'ravi@example.com', 'IT', 1);
```

```
-- Issue a Book
```

```
INSERT INTO IssuedBooks (BookID, StudentID, IssueDate, DueDate, ReturnDate, Fine)
```

```
VALUES (1, 1, '2024-03-01', '2024-03-10', '2024-03-15', 25.00);
```

```
-- Book Request Example
```

```
INSERT INTO BookRequests (StudentID, BookID, RequestDate)
```

```
VALUES (2, 3, '2024-04-01');
```

3.2 Retrieval Queries (Advanced)

1. List all books by genre "Fiction":

```
SELECT Title  
FROM Books  
JOIN Categories ON Books.CategoryID = Categories.CategoryID  
WHERE CategoryName = 'Fiction';
```

2. Find overdue books with fines:

```
SELECT Students.Name, Books.Title, IssuedBooks.Fine  
FROM IssuedBooks  
JOIN Students ON IssuedBooks.StudentID = Students.StudentID
```

```
JOIN Books ON IssuedBooks.BookID = Books.BookID  
WHERE ReturnDate > DueDate;
```

3. Show pending book requests

```
SELECT Students.Name, Books.Title, RequestDate  
FROM BookRequests  
JOIN Students ON BookRequests.StudentID = Students.StudentID  
JOIN Books ON BookRequests.BookID = Books.BookID  
WHERE Status = 'Pending';
```

4. Top 3 most issued books:

```
SELECT Books.Title, COUNT(*) AS TimesIssued  
FROM IssuedBooks  
JOIN Books ON IssuedBooks.BookID = Books.BookID  
GROUP BY IssuedBooks.BookID  
ORDER BY TimesIssued DESC  
LIMIT 3;
```

5. List all books with available stock > 0:

```
SELECT Title, Quantity  
FROM Books  
WHERE Quantity > 0;
```

6. Find books issued by a specific student (Example: Arun Kumar)

```
SELECT Books.Title, IssueDate, DueDate, ReturnDate  
FROM IssuedBooks  
JOIN Books ON IssuedBooks.BookID = Books.BookID  
JOIN Students ON IssuedBooks.StudentID = Students.StudentID
```

```
WHERE Students.Name = 'Arun Kumar';
```

7. Show students who have requested a specific book (Example: 'The Indian Economy')

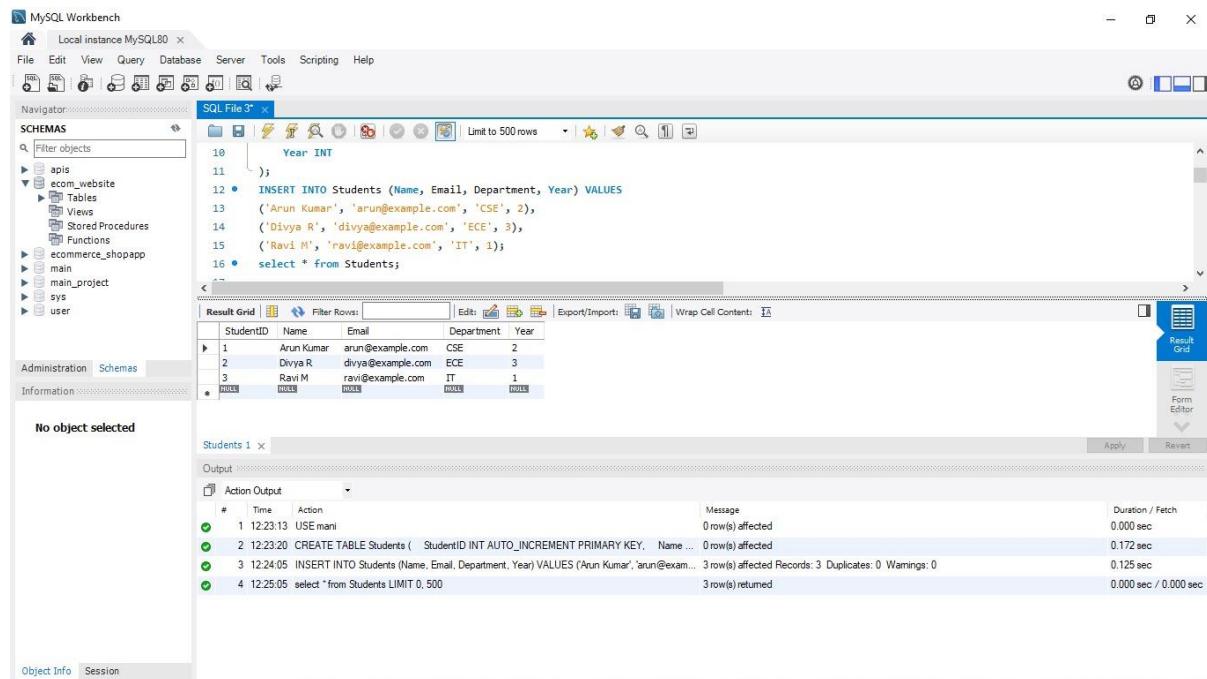
```
SELECT Students.Name, RequestDate, Status  
FROM BookRequests  
  
JOIN Students ON BookRequests.StudentID = Students.StudentID  
  
JOIN Books ON BookRequests.BookID = Books.BookID  
  
WHERE Books.Title = 'The Indian Economy';
```

4. Implementation & Results

4.1 Execution Environment

- **Database Management System:** MySQL Workbench
- **SQL Execution Tool:** MySQL Workbench

4.2 Screenshots of Execution Results



The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar displays the Navigator with Schemas (api, ecom_website, ecommerce_shopapp, main, main_project, sys, user) and Administration tabs. The main area has a SQL tab with the following code:

```
10     Year INT  
11 );  
12 • INSERT INTO Students (Name, Email, Department, Year) VALUES  
13 ('Arun Kumar', 'arun@example.com', 'CSE', 2),  
14 ('Divya R', 'divya@example.com', 'ECE', 3),  
15 ('Ravi M', 'ravi@example.com', 'IT', 1);  
16 • select * from Students;
```

The Result Grid shows the following data:

StudentID	Name	Email	Department	Year
1	Arun Kumar	arun@example.com	CSE	2
2	Divya R	divya@example.com	ECE	3
3	Ravi M	ravi@example.com	IT	1

The Output tab shows the Action Output log:

#	Time	Action	Message	Duration / Fetch
1	12:23:13	USE main	0 row(s) affected	0.000 sec
2	12:23:20	CREATE TABLE Students (StudentID INT AUTO_INCREMENT PRIMARY KEY, Name ...)	0 row(s) affected	0.172 sec
3	12:24:05	INSERT INTO Students (Name, Email, Department, Year) VALUES ('Arun Kumar', 'arun@example.com', 'CSE', 2)	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0	0.125 sec
4	12:25:05	select * from Students LIMIT 0, 500	3 row(s) returned	0.000 sec / 0.000 sec

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- apis
- ecom_website
 - Tables
 - Views
 - Stored Procedures
 - Functions
- ecommerce_shopapp
- main
- main_project
- sys
- user

Administration Schemas

No object selected

Information

SQL File 3*

```

19
20 -- Table: Authors
21 CREATE TABLE Authors (
22     AuthorID INT AUTO_INCREMENT PRIMARY KEY,
23     Name VARCHAR(100)
24 );
25
26 INSERT INTO Authors (Name) VALUES
27     ('J.K. Rowling'), ('George Orwell'), ('Sundar Raj');
28
29 select * from Authors;
30

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid

AuthorID	Name
1	J.K. Rowling
2	George Orwell
3	Sundar Raj
*	NULL

Authors 2 x

Action Output

#	Time	Action	Message	Duration / Fetch
3	12:24:05	INSERT INTO Students (Name, Email, Department, Year) VALUES ('Arun Kumar', 'arun@example.com', 'Computer Science', 'Year 1')	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0	0.125 sec
4	12:25:05	select * from Students LIMIT 0, 500	3 row(s) returned	0.000 sec / 0.000 sec
5	12:28:00	CREATE TABLE Authors (AuthorID INT AUTO_INCREMENT PRIMARY KEY, Name VARCHAR(100))	0 row(s) affected	0.062 sec
6	12:28:42	INSERT INTO Authors (Name) VALUES ('J.K. Rowling'), ('George Orwell'), ('Sundar Raj');	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0	0.062 sec
7	12:28:45	select * from Authors LIMIT 0, 500	3 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- apis
- ecom_website
 - Tables
 - Views
 - Stored Procedures
 - Functions
- ecommerce_shopapp
- main
- main_project
- sys
- user

Administration Schemas

No object selected

Information

SQL File 3*

```

30
31
32 CREATE TABLE Categories (
33     CategoryID INT AUTO_INCREMENT PRIMARY KEY,
34     CategoryName VARCHAR(100)
35 );
36
37 INSERT INTO Categories (CategoryName) VALUES
38     ('Fiction'), ('Science'), ('History');
39
40 select * from Categories;
41

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid

CategoryID	CategoryName
1	Fiction
2	Science
3	History
*	NULL

Categories 5 x

Action Output

#	Time	Action	Message	Duration / Fetch
10	12:34:19	CREATE TABLE Categories (CategoryID INT AUTO_INCREMENT PRIMARY KEY, CategoryName VARCHAR(100))	0 row(s) affected	0.079 sec
11	12:34:23	INSERT INTO Categories (CategoryName) VALUES ('Fiction'), ('Science'), ('History')	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0	0.016 sec
12	12:34:26	select * from Categories LIMIT 0, 500	3 row(s) returned	0.000 sec / 0.000 sec
13	12:38:26	select * from Categories LIMIT 0, 500	3 row(s) returned	0.000 sec / 0.000 sec
14	12:38:29	select * from Categories LIMIT 0, 500	3 row(s) returned	0.031 sec / 0.000 sec

Object Info Session

MySQL Workbench

Local instance MySQL80 X

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- Filter objects
- apis
- econ_website
- Tables
- Views
- Stored Procedures
- Functions
- ecommerce_shopapp
- main
- main_project
- sys
- user

SQL File 3* X

```

50     Quantity INT CHECK (Quantity >= 0),
51     FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID),
52     FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID)
53   );
54
55 •   INSERT INTO Books (Title, AuthorID, CategoryID, ISBN, Quantity) VALUES
56   ('Harry Potter', 1, 1, 'HP001', 3),
57   ('1984', 2, 1, '1984X', 2),
58   ('The Indian Economy', 3, 3, 'IE2024', 1);
59
60 •   select * From Books;
61

```

Administration Schemas

No object selected

Information

Result Grid | Filter Rows: [] | Edit: [] | Export/Import: [] | Wrap Cell Content: []

BookID	Title	AuthorID	CategoryID	ISBN	Quantity
1	Harry Potter	1	1	HP001	3
2	1984	2	1	1984X	2
3	The Indian Economy	3	3	IE2024	1
•	NULL	NULL	NULL	NULL	NULL

Books 6 X

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:41:22	CREATE TABLE Books (BookID INT AUTO_INCREMENT PRIMARY KEY, Title VARCHAR(255), AuthorID INT, CategoryID INT, ISBN VARCHAR(13), Quantity INT CHECK (Quantity >= 0), FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID), FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID));	0 row(s) affected	0.265 sec
2	12:41:39	INSERT INTO Books (Title, AuthorID, CategoryID, ISBN, Quantity) VALUES ('Harry Potter', 1, 1, 'HP001', 3)	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0	0.110 sec
3	12:41:43	select * From Books LIMIT 0, 500	3 row(s) returned	0.000 sec / 0.000 sec

Result Grid

MySQL Workbench

Local instance MySQL80 X

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- Filter objects
- apis
- econ_website
- Tables
- Views
- Stored Procedures
- Functions
- ecommerce_shopapp
- main
- main_project
- sys
- user

SQL File 3* X

```

68     DueDate DATE,
69     ReturnDate DATE,
70     Fine DECIMAL(10,2),
71     FOREIGN KEY (BookID) REFERENCES Books(BookID),
72     FOREIGN KEY (StudentID) REFERENCES Students(StudentID)
73   );
74
75 •   INSERT INTO IssuedBooks (BookID, StudentID, IssueDate, DueDate, ReturnDate, Fine)
76   VALUES (1, 1, '2024-03-01', '2024-03-10', '2024-03-15', 25.00);
77
78 •   select * from IssuedBooks;
79

```

Administration Schemas

No object selected

Information

Result Grid | Filter Rows: [] | Edit: [] | Export/Import: [] | Wrap Cell Content: []

IssueID	BookID	StudentID	IssueDate	DueDate	ReturnDate	Fine
1	1	1	2024-03-01	2024-03-10	2024-03-15	25.00
•	NULL	NULL	NULL	NULL	NULL	NULL

IssuedBooks 7 X

Output

Action Output

#	Time	Action	Message	Duration / Fetch
2	12:41:39	INSERT INTO Books (Title, AuthorID, CategoryID, ISBN, Quantity) VALUES ('Harry Potter', 1, 1, 'HP001', 3)	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0	0.110 sec
3	12:41:43	select * From Books LIMIT 0, 500	3 row(s) returned	0.000 sec / 0.000 sec
4	12:43:34	CREATE TABLE IssuedBooks (IssueID INT AUTO_INCREMENT PRIMARY KEY, BookID INT, StudentID INT, IssueDate DATE, DueDate DATE, ReturnDate DATE, Fine DECIMAL(10,2), FOREIGN KEY (BookID) REFERENCES Books(BookID), FOREIGN KEY (StudentID) REFERENCES Students(StudentID));	0 row(s) affected	0.265 sec
5	12:43:39	INSERT INTO IssuedBooks (BookID, StudentID, IssueDate, DueDate, ReturnDate, Fine) VALUES (1, 1, '2024-03-01', '2024-03-10', '2024-03-15', 25.00)	1 row(s) affected	0.047 sec
6	12:43:41	select * from IssuedBooks LIMIT 0, 500	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

The screenshot shows the MySQL Workbench interface. In the top-left, the Navigator pane displays the database schema with several schemas listed. The central area contains a SQL editor window titled "SQL File 3" with the following SQL code:

```

84     BookID INT,
85     RequestDate DATE,
86     Status ENUM('Pending', 'Approved', 'Rejected') DEFAULT 'Pending',
87     FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
88     FOREIGN KEY (BookID) REFERENCES Books(BookID)
89   );
90
91 • INSERT INTO BookRequests (StudentID, BookID, RequestDate)
92   VALUES (2, 3, '2024-04-01');
93
94 • select * from BookRequests;

```

Below the SQL editor is a Result Grid showing the inserted data:

RequestID	StudentID	BookID	RequestDate	Status
1	NULL	2	3	Pending
			2024-04-01	

On the right side, the "Output" pane shows the execution log for the actions performed:

Action	Time	Message	Duration / Fetch
CREATE TABLE BookRequests (RequestID INT AUTO_INCREMENT PRIMARY KEY, ...)	1 12:45:13	0 row(s) affected	0.250 sec
INSERT INTO BookRequests (StudentID, BookID, RequestDate) VALUES (2, 3, '2024-04-01')	2 12:45:17	1 row(s) affected	0.031 sec
select * from BookRequests LIMIT 0, 500	3 12:45:19	1 row(s) returned	0.000 sec / 0.000 sec

5. GitHub Repository

5.1 Repository Link

<https://github.com/mku2062022bca11/Hackathonlevel2.git>

5.2 Uploaded Files in Repository

📁 Smart-Library-Management-System/

```

├── docs/
│   ├── ER_Diagram.png
│   ├── Hackathon_Level2_Report.pdf
│   ├── database_design.md
│   └── queries_explained.md
└── sql_scripts /
    └── create_tables.sql

```

```
|   └── insert_data.sql  
|   └── sample_queries.sql  
└── screenshots/  
    ├── table_creation /  
    └── query_results /  
└── README.md
```