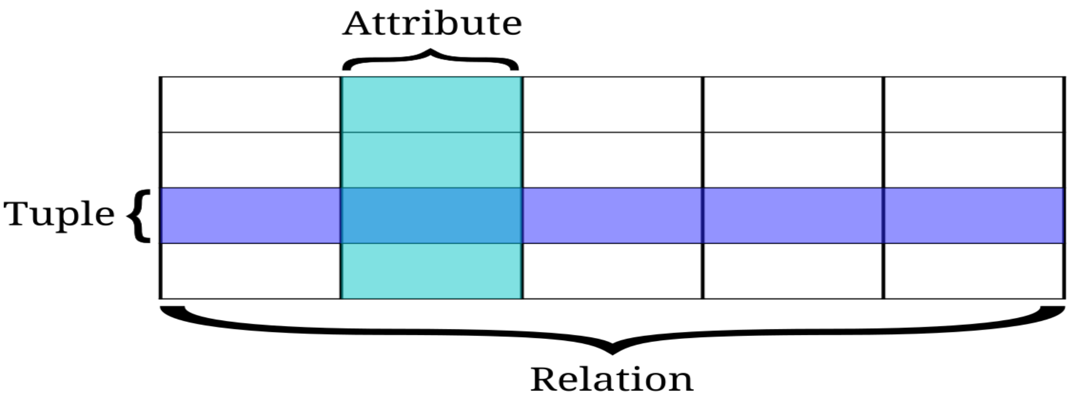
MY SQL PROJECT

“LIBRARY MANAGEMENT SYSTEM”



Project Type:	Data base Management system
Name:	Fatiha Maryam
Date:	2025/02/25
Email:	fatihaniazi555@email.com
GitHub:	**https://github.com/Fatiha-maryam**
LinkedIn:	**www.linkedin.com/in/fatiha-maryam-945880305**

INTRODUCTION:

The Library Management System is a project designed to manage essential library operations, such as issuing books, maintaining student records, tracking book availability, and handling late return fines. This project follows a structured database approach, where we have designed an Entity-Relationship Diagram (ERD), defined relationships, and ensured proper normalization. Using MySQL, we will implement the database, create tables, establish relationships, and apply key DBMS concepts in a practical way. This project helps us understand how databases work in real-world applications and how data is efficiently stored and managed.

PURPOSE:

The main goal of this project is to apply database management system (DBMS) concepts in a real-world scenario. Through this Library Management System, we aim to understand how data is stored, updated, and retrieved efficiently. This project will help us gain hands-on experience in designing structured databases, managing relationships between entities, and implementing key database operations. By working on this system, we will strengthen our practical understanding of database management, which is essential for handling large-scale applications.

IMPORTANCE:

In today’s digital world, effective data management is essential for any organization. A well-structured database allows for easy storage, retrieval, and management of information, leading to better efficiency and decision-making. Database Management Systems (DBMS) play a key role in handling large volumes of data. By developing this Library Management System, we are enhancing our understanding of database design, normalization, and relationships.

This project will serve as a strong foundation for working with large-scale database systems in industries and real-world applications.

PROBLEM STATEMENT:

Traditional library management systems often face issues such as data redundancy, inconsistency, and inefficiencies in handling books, borrowers, and transactions. Without a structured database, searching for books, tracking issued materials, and maintaining accurate records become error-prone and time-consuming, leading to mismanagement and operational delays.

This project aims to overcome these challenges by developing a centralized **Library Management System** using MySQL. The system ensures data consistency, accuracy, and efficiency by providing a well-organized way to store, access, and manage book records and borrower details. It will streamline essential library operations, enabling faster retrieval of information, seamless tracking of issued books, and reduced manual workload. By automating key functions, this system will improve operational efficiency and enhance the overall user experience.

OBJECTIVES

1. **Efficient Book Management:**
   - Store and manage book details, including title, author, ISBN, category, and availability.
2. **User and Borrower Management:**
   - Maintain records of students and other borrowers, tracking their borrowing history and book returns.
3. **Minimizing Data Redundancy and Inconsistency:**
   - Use a relational database model with proper normalization to eliminate duplicate records and ensure data integrity.
4. **Easy Book Tracking:**
   - Allow librarians to check which books are borrowed, by whom, and when they are due for return.
5. **Automated Record-Keeping:**

- Maintain accurate records of book issues, returns, and overdue books to reduce manual effort.

6. **Efficient Query Execution:**

- Implement optimized SQL queries for fast and efficient data retrieval.

7. **User-Friendly Interface (Optional for Future Expansion):**

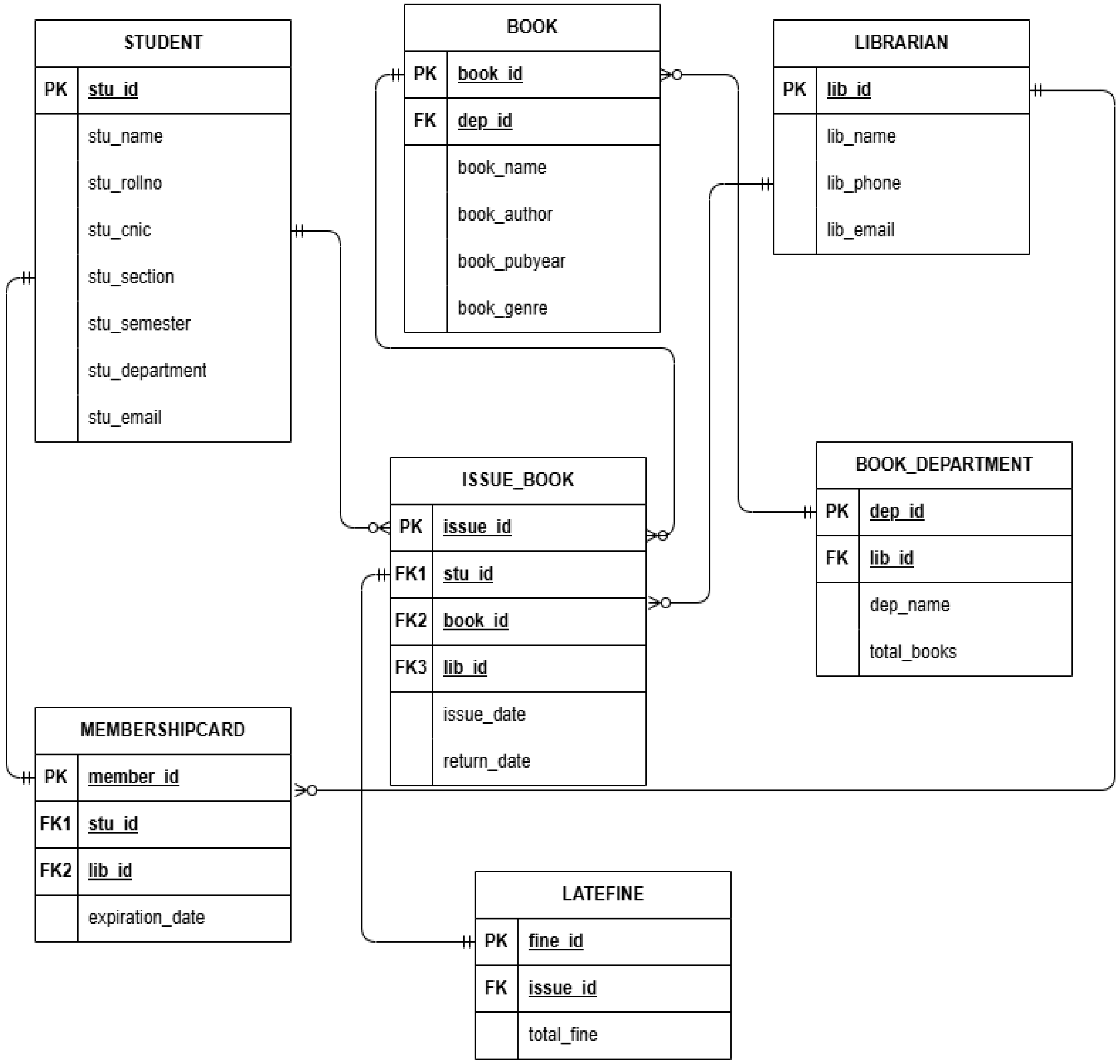
- The system can be extended with a frontend interface in the future to improve user interaction.

DESIGN;

Conceptual Schema:

The conceptual schema represents the high-level structure of the Library Management System, defining entities, attributes, and their relationships without considering implementation details. The main entities include **Student, Book, Librarian, MembershipCard, and LateFine**, each with its respective attributes. The system also establishes relationships such as **Issue_Book (tracks book issuance), Manages_Book (links librarians to books), Generate_Fine (handles late return fines), and Make_MembershipCard (manages student memberships).** This schema ensures a well-structured database foundation for efficient data organization and retrieval.

ERD:



LOGICAL SCHEMA:

- **STUDENT**

- **stu_id** (Primary Key)
- name
- roll_no
- cnic
- semester
- department
- email

□ **BOOK**

- **book_id** (Primary Key)
- name
- author
- genre
- publish_year
- **dep_id** (Foreign Key referencing BOOK_DEPARTMENT)

□ **LIBRARIAN**

- **lib_id** (Primary Key)
- name
- phone_no
- email
- **dep_id** (Foreign Key referencing BOOK_DEPARTMENT)

□ **BOOK_DEPARTMENT**

- **dep_id** (Primary Key)
- dep_name
- total_books
- **lib_id** (Foreign Key referencing LIBRARIAN)

□ **ISSUE_BOOK**

- **issue_id** (Primary Key)
- issue_date
- return_date
- **stu_id** (Foreign Key referencing STUDENT)
- **book_id** (Foreign Key referencing BOOK)
- **lib_id** (Foreign Key referencing LIBRARIAN)

□ **LATEFINE**

- **fine_id** (Primary Key)
- total_fine
- **issue_id** (Foreign Key referencing ISSUE_BOOK)

□ **MEMBERSHIPCARD**

- **mem_id** (Primary Key)
- expiration_date
- university
- **stu_id** (Foreign Key referencing STUDENT)

PHYSICAL SCHEMA:

IMPLEMENTATION IN MY SQL:

```sql
/* creating my database called "my_library" then using it*/
create database my_library;
use my_library;

/*creating tables*/
/*creating student table with 7 attributes */

create table STUDENT(
stu_id int primary key,
name_ varchar(70),
roll_no int,
section char(1),
semester int,
department varchar(70),
email varchar(70));

/*inserting values into student table*/
```

```
insert into STUDENT values (1, 'Bethina', 44, 'A', 6, 'CS',
'bwybern0@sitemeter.com'),
 (2, 'Sara-ann', 47, 'B', 8, 'AI', 'schardin1@wikimedia.org'),
(3, 'Andria', 92,  'C', 4, 'IT', 'asummerside2@ning.com'),
 (4, 'Freddy', 95, 'D', 1, 'psychology', 'fglison3@springer.com'),
 (5, 'Ros', 64,  'E', 7, 'maths', 'rivery4@pinterest.com'),
 (6, 'Tammy', 72,  'F', 1, 'Media', 'tgeale5@thetimes.co.uk'),
 (7, 'Hugues', 9,  'C', 6, 'ENglish', 'hsprigings6@businesswire.com'),
 (8, 'Alix', 76, 'A', 5, 'Business', 'atrodden7@hhs.gov'),
 (9, 'Dianne', 90,  'B', 1, 'Finance', 'dgaskins8@slashdot.org'),
 (10, 'Hayley', 54,  'B',3, 'Fintach', 'hstatter9@upenn.edu'),
 (11, 'Burton', 68,  'F', 2, 'BBA', 'bmaltera@1688.com'),
(12, 'Georges', 9,  'C', 6, 'CS', 'gschiementzb@creativecommons.org'),
(13, 'Garth', 78,  'E', 4, 'IT', 'gmaccomec@shareasale.com'),
 (14, 'Rici', 60,  'C', 3, 'IT', 'rtrembletd@people.com.cn');

 /*Displaying the table */
 select * from STUDENT;

 /* creating table book having 6 attributes */

create table BOOK(
book_id int primary key,
title varchar(70),
author varchar(70),
genre varchar(70),

pubyear year,

dep_id int ,

foreign key (dep_id) references BOOKDEPARTMENT(dep_id));


/*inserting data into BOOK table*/

insert into BOOK values

 (1, 'MATHS', 'Sasha Camel', 'real', 2013, 1),

 (2, 'Science', 'Hieronymus Van Schafflaer', 'funny', 0, 2),

 (3, 'Psychology', 'Xylia Rebbeck', 'normal', 2014, 3),

 (4, 'Sigment frued', 'Kiel McCartan', 'modren', 2008, 4),

 (5, 'fatihalife', 'Gary Jiracek', 'fiction', 2013, 5),

 (6, 'Awaislife', 'Casar Mathouse', 'frightened', 2015, 6),

 (7, 'mom', 'Demetris Blunsen', 'classic', 2012, 7),

 (8, 'merimom', 'Barby Gregoretti', 'romatic', 2011, 8),

 (9, 'business', 'Florrie Stranks', 'sad', 2015, 9),

 (10, 'linear', 'George Elcoat', 'funny', 2010, 10),

 (11, 'AI', 'Lyndsey Yoxall', 'sad', 2013, 11),

 (12, 'CS', 'Alphonso Lombardo', 'happy', 2011, 12),

 (13, 'It', 'Tamera Alcalde', 'thriller', 2013, 13),

 (14, 'Entrepreneurship', 'Moshe Spolton', 'action', 2014, 14);

select * from BOOK;


/* creating table Librarian with 4 attributes*/
```

```
create table LIBRARIAN(
lib_id int primary key,
lib_name varchar(70),
phone varchar(15),
email varchar(70));

/*inserting data into Librarian*/

insert into LIBRARIAN values
 (1, 'Maureene', '770-226-9098', 'mshury0@imageshack.us'),
 (2, 'Emelita', '673-948-7160', 'edutton1@constantcontact.com'),
(3, 'Ophelie', '611-264-8920', 'ogianinotti2@cdbaby.com'),
 (4, 'Anstice', '278-909-1662', 'akoopman3@adobe.com'),
(5, 'Anthony', '463-516-5755', 'apietruszka4@amazon.com'),
 (6, 'Ceil', '533-560-2780', 'cmontier5@skyrock.com'),
 (7, 'Marys', '112-905-0701', 'mhold6@cmu.edu'),
 (8, 'Mariquilla', '197-616-1609', 'mpritchitt7@g.co'),
 (9, 'Laughton', '822-768-3606', 'lgrog8@nifty.com'),
 (10, 'Felipe', '330-345-5320', 'felcom9@ustream.tv'),
 (11, 'Peterus', '925-823-3735', 'pwallwooda@ebay.co.uk'),
 (12, 'Antonie', '446-365-6522', 'amccreeryb@upenn.edu'),
 (13, 'Elizabeth', '483-477-7533', 'epolinic@chicagotribune.com'),
 (14, 'Olympia', '372-155-5230', 'omoyced@bluehost.com');
/*displaying whole librarian table*/
```

```
select * from LIBRARIAN;

/*creating table bookdepartment with four attributes */
create table BOOKDEPARTMENT(
dep_id int primary key,
dep_name varchar(70),
total_books int,
lib_id int,
foreign key (lib_id)references LIBRARIAN(lib_id));

/* inserting data into BOOKDEPARTMENT*/

insert into BOOKDEPARTMENT values
(1, 'Training', 63, 1),
(2, 'Marketing', 31, 2),
 (3, 'Product Management', 33, 3),
 (4, 'Training', 74, 4),
 (5, 'Research and Development', 30, 5),
(6, 'Engineering', 35, 6),
(7, 'Training', 34, 7),
(8, 'Accounting', 74, 8),
 (9, 'Research and Development', 34, 9),
 (10, 'Marketing', 69, 10),
 (11, 'Research and Development', 60, 11),
(12, 'Support', 72, 12),
(13, 'Legal', 84, 13),
 (14, 'Engineering', 59, 14);
/*displaying the full table */
select * from BOOKDEPARTMENT;

/*creating table issuebook 6 attributes*/

create table ISSUEBOOK(
issue_id int primary key,
issue_date date,
return_date date,
stu_id int,
book_id int,
lib_id int,
foreign key(stu_id)references STUDENT(stu_id),
foreign key(book_id)references BOOK(book_id),
foreign key (lib_id)references LIBRARIAN(lib_id));

/*inserting data into issuebook table*/

insert into ISSUEBOOK values
 (1, '1997-06-15', '2005-07-20', 1, 1, 1),
(2, '2007-03-10', '2012-09-25', 2, 2, 2),
(3, '2011-11-05', '2010-04-30', 3, 3, 3),
(4, '2017-08-21', '2016-12-10', 4, 4, 4),
(5, '2009-05-18', '2017-10-22', 5, 5, 5),
(6, '2013-02-14', '2011-06-30', 6, 6, 6),
(7, '2012-04-09', '2005-11-15', 7, 7, 7),
(8, '2009-09-01', '2016-01-19', 8, 8, 8),
(9, '2014-12-05', '2020-02-29', 9, 9, 9),
(10, '2014-07-20', '2014-07-21', 10, 10, 10),
(11, '2008-10-14', '2019-05-16', 11, 11, 11),
(12, '2016-03-07', '2011-08-23', 12, 12, 12),
(13, '2013-05-22', '2014-09-11', 13, 13, 13),
(14, '2017-01-29', '2016-06-17', 14, 14, 14);

select * from ISSUEBOOK;

/*create the table of MEMBERSHIPCARD*/

create table MEMBERSHIPCARD(
mem_id int primary key,
expiration_date date,
stu_id int,
lib_id int,
foreign key (stu_id)references STUDENT (stu_id),
foreign key (lib_id) references LIBRARIAN(lib_id) );
```

```
/*insert data into MEMBERSHIPCARD*/

insert into MEMBERSHIPCARD values
(1, '2024-09-05', 1, 1),
(2, '2025-02-13', 2, 2),
(3, '2024-08-04', 3, 3),
(4, '2024-07-30', 4, 4),
(5, '2024-04-15', 5, 5),
(6, '2025-02-13', 6, 6),
(7, '2024-10-28', 7, 7),
(8, '2024-03-07', 8, 8),
(9, '2025-02-15', 9, 9),
(10, '2025-01-28', 10, 10),
(11, '2024-11-23', 11, 11),
(12, '2024-12-14', 12, 12),
(13, '2024-11-22', 13, 13),
(14, '2024-06-26', 14, 14);

select * from MEMBERSHIPCARD;

/*creating table Latefine having 4 attributes*/

create table LATEFINE(
```

```sql
fine_id int primary key,

total_fine int,

issue_id int,

foreign key (issue_id)references ISSUEBOOK(issue_id));


/*inserting data into LATEFINE*/


insert into LATEFINE values

(1, 66, 1),

(2, 54, 2),

(3, 30, 3),

(4, 76, 4),

(5, 59, 5),

(6, 100, 6),

(7, 377, 7),

(8, 199, 8),

(9, 94, 9),

(10, 98, 10),

(11, 72, 11),

(12, 87, 12),

(13, 70, 13),

(14, 32, 14);

/*Displaying whole data */

select * from LATEFINE;
```

```
/*BASIC QUERIES (CRUD OPERATIONS)*/
/* INSERT, Update, DELETE,Select operations */

/*PROBLEM STATEMENT 1*/
 /*Insert a new student in student table*/
 insert into STUDENT(stu_id,name_ ,roll_no ,section ,semester ,department ,email)
 values (23,'Amna Mehmood',34,'A',5,'Social sciences','amna@email.com');

select * from student;

/*PROBLEM STATEMENT 2 */
/*Update a student's email and departmnet*/
update STUDENT
set  email='Sam@email.com',department ='CS'
where stu_id=11;

select * from STUDENT;

/*PROBLEM STATEMENT 3*/
/* check which books are not available for issue */

-- as now all books are issued so i will insert new book to book table and will not enter them in issuetable
```

```
insert into BOOK (book_id,title,author,genre,pubyear,dep_id)
values(16,'DATABASE SYSTEM','Johnson','Computersci',2004,1);

/* check which books are not available for issue */
select book_id,title,author,genre,pubyear from BOOK
where book_id not in (select book_id from ISSUEBOOK);

/*PROBLEM STATEMENT 4*/
/*check which book belong to which department and how many total books are there */
select title,dep_name,total_books from BOOK
 join BOOKDEPARTMENT on BOOK.dep_id= BOOKDEPARTMENT.dep_id
where book_id=2;

/*PROBLEM STATEMENT 5*/
/*check which book is issued to which student by which librarian*/
select B.title, S.name_ ,L.lib_name from ISSUEBOOK I
join BOOK B on I.book_id =B.book_id
join STUDENT S on I.stu_id=S.stu_id
join LIBRARIAN L on I.lib_id=L.lib_id
where I.issue_id=6;

/*PROBLEM STATEMENT 6*/
/*check which librarian generate which student a fine and what's total fine*/
select S.name_ AS student_name, L.lib_name AS librarian_name,
LA.total_fine  from LATEFINE LA
join ISSUEBOOK I on LA.issue_id = I.issue_id
join STUDENT S on I.stu_id = S.stu_id
join LIBRARIAN L on I.lib_id = L.lib_id
where L.lib_id = 12;

/* PROBLEM STATEMENT 7*/
/*check which book is issued but not returned*/
SELECT B.title AS book_name
FROM ISSUEBOOK I
JOIN BOOK B ON I.book_id = B.book_id
LEFT JOIN LATEFINE L ON I.issue_id = L.issue_id
WHERE L.issue_id IS NULL;

/*PROBLEM STATEMENT 8*/
/* checks how many students have membershipcard*/
select count(M.stu_id) as total_membership_cards from MEMBERSHIPCARD M;

/*PROBLEM STATEMENT 9*/
/*What is expiration date of Membershipcard of specific student*/
select S.name_ , M.expiration_date from MEMBERSHIPCARD  M
 join STUDENT S on M.stu_id=S.stu_id
where M.stu_id =9;
```

```
/*PROBLEM STATEMENT 10*/
/*FIND ALL STUDENTS WHO HAVE NOT ISSUED ANY BOOK*/
SELECT S.stu_id, S.name_
FROM STUDENT S
LEFT JOIN ISSUEBOOK I ON S.stu_id = I.stu_id
WHERE I.issue_id IS NULL;

/*PROBLEM STATEMENT 11*/
/* find most issued book in library */
SELECT B.title, COUNT(I.book_id) AS issue_count
FROM ISSUEBOOK I
JOIN BOOK B ON I.book_id = B.book_id
GROUP BY B.title
ORDER BY issue_count DESC
 LIMIT 1;

/*PROBLEM STATEMENT 12*/
/*Find student who have pending fine*/
SELECT S.stu_id, S.name_, L.total_fine
FROM LATEFINE L
JOIN ISSUEBOOK I ON L.issue_id = I.issue_id
JOIN STUDENT S ON I.stu_id = S.stu_id
WHERE L.total_fine  >0;
```

```
/*PROBLEM STATEMENT 13*/

/*Find librarian who has issue most books*/

SELECT L.lib_name, COUNT(I.issue_id) AS total_issued

FROM ISSUEBOOK I

JOIN LIBRARIAN L ON I.lib_id = L.lib_id

GROUP BY L.lib_id

ORDER BY total_issued DESC

LIMIT 1;
```

END