LIBRARY MANAGEMENT SYSTEM

A Case Study Submitted to

DEPARTMENT
of
COMPUTER SCIENCE AND SYSTEMS ENGINEERING

Submitted by

K.MD SHAFI	**21121A1555**
A.SASHIDHAR	**21121A1503**

Under the Guidance of
Mr. P. Yogendra Prasad
Assistant Professor

SREE VIDYANIKETHAN Engineering College (Autonomous)

Department of Computer Science and Systems Engineering
Sree Vidyanikethan Engineering College (Autonomous)
Sree Sainath Nagar, Tirupati – 517 102
(2022-2023)

SREE VIDYANIKETHAN ENGINEERING COLLEGE
(AUTONOMOUS)
Sree Sainath Nagar, Tirupati

DEPARTMENT OF COMPUTER SCIENCE AND SYSTEMS ENGINEERING

CERTIFICATE

This is to certify that the Case Study report entitled

LIBRARY MANAGEMENT SYSTEM

is the Bonafide work done by

K.MD SHAFI	**21121A1555**
A. SASHIDHAR	**21121A1503**

in the Department of **Computer Science and Systems Engineering**, and submitted to Computer Science and Systems Engineering during the academic year 2022-2023. This work has been carried out under my supervision.

Guide:

Mr. P. Yogendra Prasad
Assistant Professor
Dept. of CSSE

Head:

Dr. K. Ramani
Professor & Head
Dept. of CSSE

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DEPARTMENT OF COMPUTER SCIENCE AND SYSTEMS ENGINEERING

VISION

To become a centre of excellence in Computer Sciences and Systems Engineering through teaching, training, research and innovation to create quality engineering professionals who can solve the growing complex problems of the society.

MISSION

- ✓ Established with the cause of development of technical education in advanced computer sciences and engineering with applications to systems there by serving the society and nation.
- ✓ Transfer of Knowledge through contemporary curriculum and fostering faculty and student development.
- ✓ Create keen interest for research and innovation among students and faculty by understanding the needs of the society and industry.
- ✓ Skill development among diversity of students in technical domains and profession for development of systems and processes to meet the demands of the industry and research.
- ✓ Imbibing values and ethics in students for prospective and promising engineering profession and develop a sense of respect for all.

PROGRAM EDUCATIONAL OBJECTIVES

1. Demonstrate competencies in the Computer Science domain and Management with an ability to comprehend, analyze, design and create software systems for pursuing advanced studies in the areas of interest.
2. Evolve as entrepreneurs or be employed by acquiring required skill sets for developing computer systems and solutions in multi-disciplinary areas.
3. Exhibit progression and professional skill development in Computer programming and systems development with ethical attitude through life-long learning.

PROGRAM SPECIFIC OUTCOMES

PSO1: Employ Systems Approach to model the solutions for real life problems, design and develop software systems by applying Modern Tools.

PSO2: Develop solutions using novel algorithms in High Performance Computing and Data Science.

PSO3: Use emerging technologies for providing security and privacy to design, deploy and manage network systems.

PROGRAM OUTCOMES

1. Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
6. Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

II B. Tech. – II Semester

(20BT40531) DATABASE MANAGEMENT SYSTEMS LAB

COURSE OUTCOMES

CO1. Analyze the requirements of a given database problem and design viable ER-Models for implementation of database.

CO2. Create database schemas, select and apply suitable integrity constraints for querying databases using SQL interface.

CO3. Develop and interpret PL/SQL blocks to centralize database applications for maintainability and reusability.

CO4. Develop database applications for societal applications such as ticket reservation system, employee payroll system using modern tools.

CO5. Work independently and communicate effectively in oral and written forms.

DECLARATION

We hereby declare that this project report titled "Title" is a genuine work carried out by us, in B.Tech (Computer Science and Systems Engineering) degree course of Jawaharlal Nehru Technological University Anantapur and has not been submitted to any other course or University for the award of any degree by us.

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature of the students

1. K.MD SHAFFI
2. A.SASHIDHAR

ABSTRACT

The Library Management System is a comprehensive software solution designed to manage and streamline the day-to-day operations of a library. This project aims to develop a relational database management system to efficiently manage all aspects of a library, including book management, borrower management, circulation management, and reporting. The system is designed to automate routine tasks, such as book acquisition, cataloging, and lending, while also providing users with an intuitive interface to search for books, make reservations, and manage their accounts. The system is built using a robust RDBMS, which ensures reliable data storage, retrieval, and management. The system's reporting capabilities allow librarians to generate detailed reports on book circulation, overdue books, and borrower statistics, providing valuable insights for library management. The proposed Library Management System is an efficient and user-friendly solution for libraries of all sizes, providing an essential tool to streamline operations and improve service quality.

Keywords: SQL Server, Microsoft SQL server management studio, HTML, CSS , java script, PHP server.

TABLE OF CONTENTS

CHAPTER 1. INTRODUCTION

1.1 Introduction to the topic

In an age of rapid technological advancement, libraries have evolved beyond their traditional role as repositories of books and information. Today, libraries are dynamic centers for knowledge dissemination, fostering community engagement, and providing access to diverse resources in various formats. To meet the ever-growing demands of modern library operations, an efficient and comprehensive Library Management System (LMS) becomes a crucial tool.

This introduction aims to present the key features and benefits of a cutting-edge Library Management System, which revolutionizes the way libraries function and empowers librarians to provide enhanced services to their patrons.

In conclusion, a robust Library Management System is essential for libraries to adapt to the evolving information landscape and meet the expectations of today's tech-savvy users. By streamlining operations, integrating digital resources, providing comprehensive analytics, and enabling effective communication, an advanced LMS empowers librarians to deliver enhanced services and experiences to their patrons. Embracing a modern Library Management System is a crucial step towards building a future-ready library that remains relevant and valuable in the digital age.

1.2 Problem Statement

Libraries play a vital role in society by providing access to knowledge, fostering learning, and promoting literacy. However, many libraries struggle with outdated and inefficient management systems, hindering their ability to effectively serve their patrons. This problem statement identifies the key issues faced by libraries in their existing management systems and highlights the need for a modern and efficient Library Management System (LMS).

Manual and Time-consuming Processes:
Many libraries still rely on manual processes for tasks such as cataloging, circulation, and inventory management. This manual approach is time-consuming and prone to errors, leading to inefficiencies in resource management. Librarians spend excessive time on administrative tasks, diverting their attention away from more value-added activities such as assisting patrons and expanding library services.

Limited Accessibility and Searchability:
Outdated library management systems often lack user-friendly interfaces and robust search functionalities. Patrons face challenges in locating resources, as the cataloging systems may be outdated or poorly organized. Limited accessibility to digital resources further restricts users from fully utilizing the library's offerings, especially in an era where online access to information is paramount.

Inadequate Resource Tracking and Security:
Traditional library systems struggle to effectively track and manage borrowed materials. Manual record-keeping may result in misplaced items, overdue notices not being sent, or inefficient handling of fines and penalties. Additionally, outdated security measures may fail to prevent theft or unauthorized borrowing, leading to resource loss and compromised library collections.

1.3 Objectives

This is a ticket booking web portal that includes various travelling availabilities through bus, train, flight. To move from source to destination we will have innumerable options where we can select the travel type based on our interest , availabilities and our economic stability. This website provides the customer all available transports from one place to other place via the required travel type which includes various availabilities like business class ,first class and economy seats regarding flights, sleeper seats and seater seats in train, AC and NON-AC sleeper as well as seater seats in bus.

The objective of this case study is to design and develop a database for the travel agency to maintain the records of various vehicles, admins, and customers who are accessing the website. It also maintains records of the regular customers, customers who have booked and who have cancelled, the confirmation of booking for the customer is done by the admin.

CHAPTER 2. DATABASE DESIGN

2.1 List of Attributes, entities and relationship

1. **Entity Name: Book**

Attributes	**Type**
BookID	int(10)
Titel	varchar(20)
Author	Int
Category	Int
AvailableCopies	Int

2. **Entity Name: Borrower**

Attributes	**Type**
BorrowerID	int(10)
FirstName	varchar(25)
LastName	varchar(25)
Email	varchar(50)
Phone	varchar(10)
Address	varchar(100)

3. **Entity Name: Transactions**

Attributes	**Type**
TransactionID	int(10)
UserID	int (25)
BookID	Int
TransactionDate	Date
ReturnDate	Date
Status	Varchar(100)

4. **Entity Name: Category**

Attributes	**Type**
CategoryID	int(10)
CategoryName	Varchar(50)

5. **Entity Name: Author**

Attributes	**Type**
AuthorID	int(10)
AuthorName	Varchar(50)
Country	Varchar(50)
BirthDate	Date

6. **Entity Name: Staff**

Attributes	**Type**
Staffid	int(10)
Staffname	varchar(25)
position	varchar(25)
email	varchar(55)
mobile	varchar(10)

7. **Entity Name: Finepenalty**

Attributes	**Type**
Fine_id	int(10)
Borrowerid	int(10)
amount	Money
description	Varchar(50)
payment_status	Vachar(20)

8. **Entity Name: Publisher**

Attributes	**Type**
Publisherid	int(10)
Publishername	varchar(40)
address	varchar(20)
contact_number	int(10)
email	varchar(45)

9. **Entity Name: Reservations**

Attributes	**Type**
Reservationsid	int(10)
Bookid	int(10)
UserID	int (10)
reservation_date	date
date	date
status	varchar(50)

10. **Entity Name: Languages**

Attributes	**Type**
Languageid	int(10)
Bookid	int(10)
LanguagesName	Varchar(50)

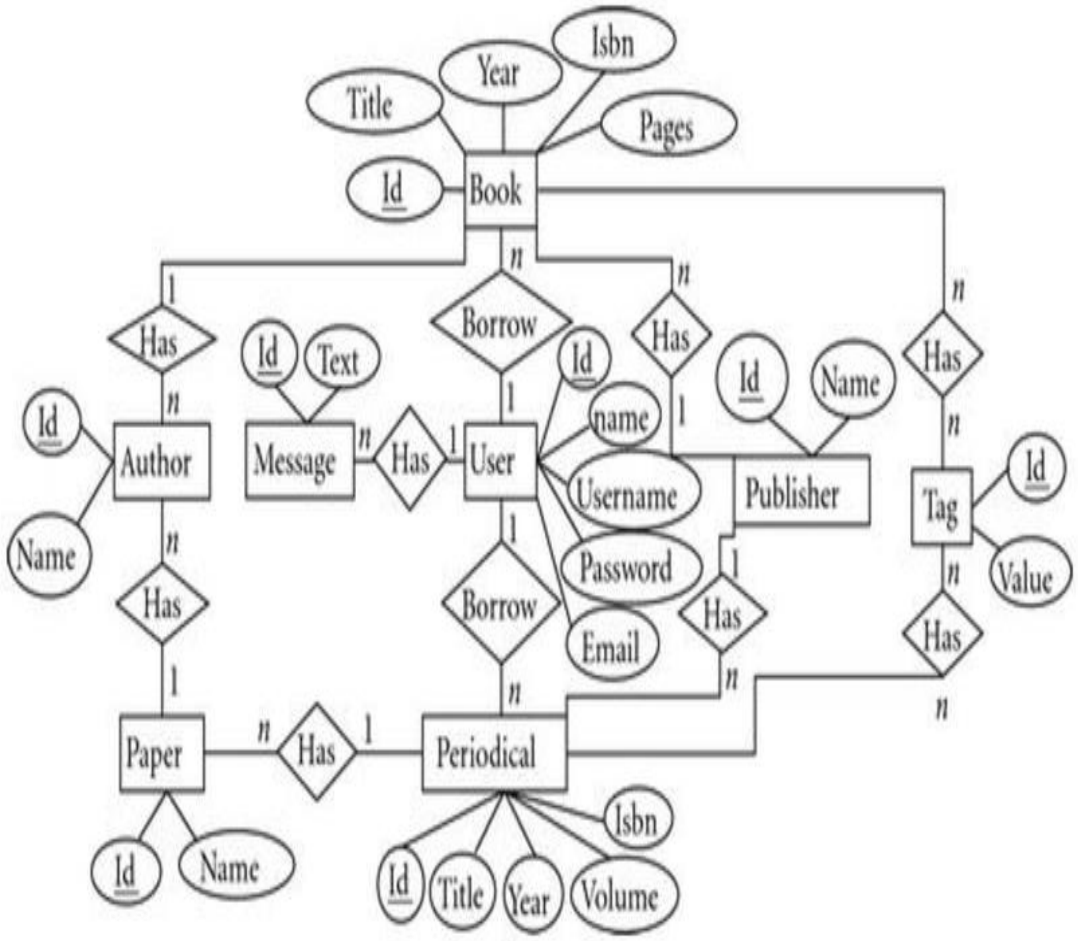
11. **Entity Name: Bookcopies**

Attributes	**Type**
copyid	int(10)
Bookid	Int(10)
location	varchar (50)
availability_status	Varchar(50)

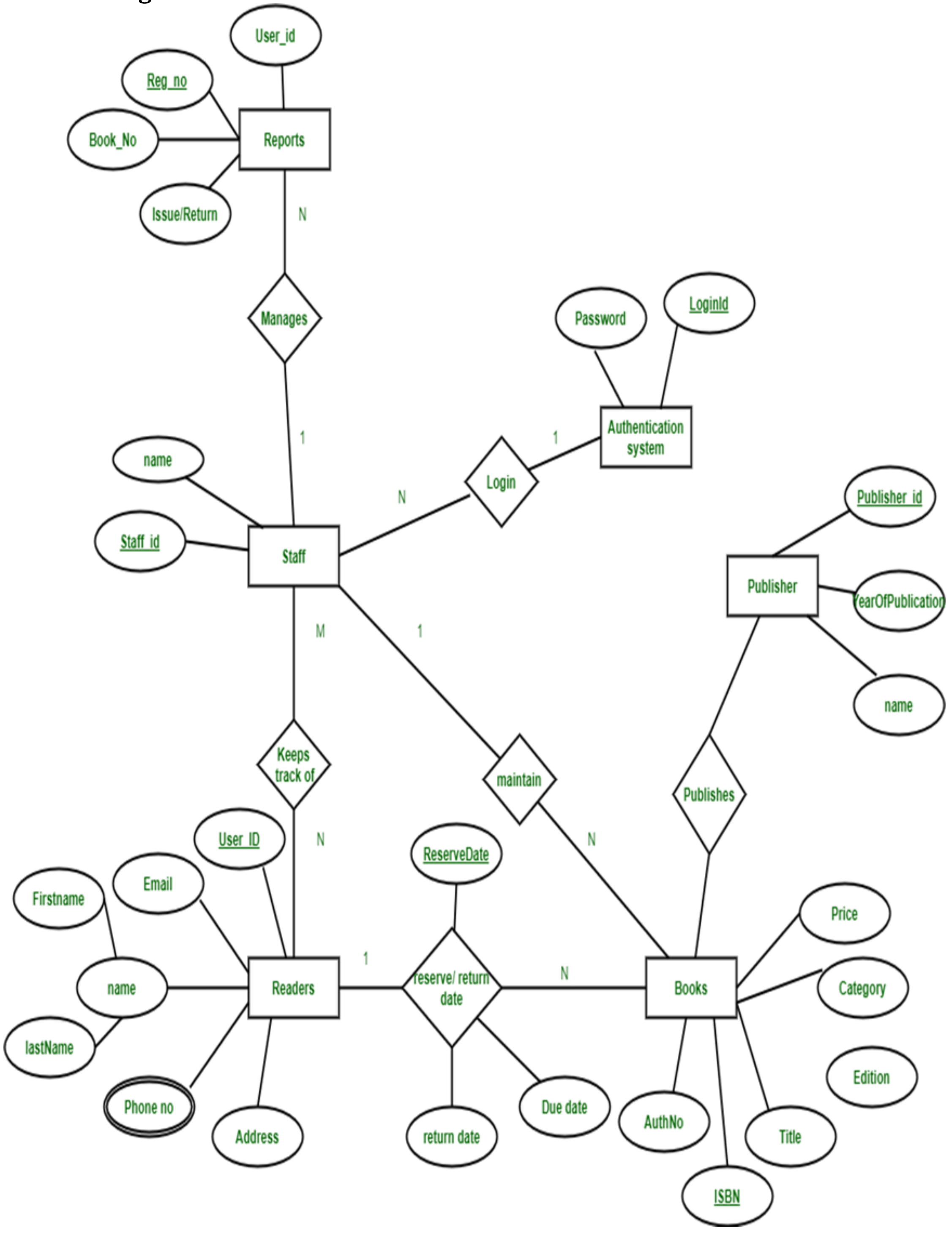
12. **Entity Name: BookReview**

Attributes	**Type**
Reviewid	int(10)
Bookid	int(10)
Borrowerid	int(10)
ReviewText	Varchar(50)
ReviewDate	Date

2.1.1 Entities and their relationships:



2.2 E-R Diagram



CHAPTER 3. RELATIONAL MODEL

3.1 Database languages

Four categories of database languages :

1. Data definition language (DDL)

Data definition language (DDL) creates the framework of the database by specifying the database schema, which is the structure that represents the organization of data. Its common uses include the creation and alteration of tables, files, indexes and columns within the database. This language also allows users to rename or drop the existing database or its components.

Here's a list of DDL statements:

- CREATE: Creates a new database or object, such as a table, index or column.
- ALTER: Changes the structure of the database or object.
- DROP: Deletes the database or existing objects.
- RENAME: Renames the database or existing objects.

2. Data manipulation language (DML)

Data manipulation language (DML) provides operations that handle user requests, offering a way to access and manipulate the data that users store within a database. Its common functions include inserting, updating and retrieving data from the database.

Here's a list of DML statements:

- INSERT: Adds new data to the existing database table.
- UPDATE: Changes or updates values in the table.
- DELETE: Removes records or rows from the table.
- SELECT: Retrieves data from the table or multiple tables.

3. Data control language (DCL)

Data control language (DCL) controls access to the data that users store within a database. Essentially, this language controls the rights and permissions of the database system. It allows users to grant or revoke privileges to the database.

Here's a list of DCL statements:

- GRANT: Gives a user access to the database.
- REVOKE: Removes a user's access to the database.

4. Transaction control language (TCL)

Transaction control language (TCL) manages the transactions within a database. Transactions group a set of related tasks into a single, executable task. All the tasks must succeed in order for the transaction to work. Here's a list of TCL statements:

- COMMIT: Carries out a transaction.
- ROLLBACK: Restores a transaction if any tasks fail to execute.

3.2 Table Description

Following are the tables along with constraints used in Library management system

database.

1. **Book :** The Book table in a library management system typically stores information about the books available in the library

   Constraint: The BookID should be unique for each book in the table.

2. **Borrower :** The Borrower table in a library management system stores information about the library patrons or borrowers. It keeps track of the individuals who borrow books from the library.

   Constraint: The BorrowerID should be unique for each borrower in the table.

3. **Transaction** : The Transaction table in a library management system is used to record the transactions or activities related to borrowing and returning books. It helps in keeping track of the history of book transactions for each borrower.

   Constraint: The Transaction_id should be unique for each transaction in the table. The Userid and Bookid columns should reference existing entries in the respective tables.

4. **Category :** The Category table in a library management system is used to classify books into different categories or genres. It helps in organizing and categorizing books based on their subject matter, allowing users to easily search and locate books of their interest.

   Constraint: The Categoryid should be unique for each category in the table.

5. **Author :** The Author table in a library management system is used to store information about the authors of books available in the library. It helps in maintaining a record of authors and their associated books, allowing users to search for books written by a specific author.

   Constraint: The Authorid should be unique for each author in the table.

6. **Staff** : The Staff table in a library management system is used to store information about the staff members who work in the library. It helps in managing and tracking the library staff, their roles, and their relevant details.

   Constraint: The Staffid should be unique for each staff member in the table.

7. **Finepenalty** : The FinePenalty table in a library management system is used to track fines or penalties imposed on library borrowers for late returns or other violations. It stores information about the fines imposed, the borrowers who incurred the fines, and the related transactions

   Constraint: The Fine_ID should be unique for each fine in the table. The BorrowerID column should reference an existing entry in the Borrower table.

8. **Publisher** : The Publisher table in a library management system stores information about the publishers of books available in the library. It includes details about the publishing companies, such as their names, addresses, and contact information.

   Constraints: Primary Key Constraint (PublisherID): Ensures that each PublisherID value in the table is unique and serves as the primary key for identifying publishers.

   Not Null Constraint (PublisherName): Ensures that the PublisherName column must have a value and cannot be empty.

9. **Bookcopies** : The Bookcopies table in a library management system tracks the individual copies of books available in the library. It stores information about each copy, such as its unique identifier, availability status, associated book details, and any additional attributes.

   Constraints: Ensures that each CopyID value in the table is unique and serves as the primary key for identifying book copies.

10. **Reservations** : The Reservations table in a library management system is used to track reservations made by library users for specific books. It stores information

about the reservation, including the user who made the reservation, the book reserved, the reservation status, and any additional attributes.

Constraints: Ensures that each ReservationID value in the table is unique and serves as the primary key for identifying reservations.

11. **Languages** : The Languages table is used to store information about different languages in which books are available. It helps in categorizing and organizing books based on their language.

Constraints: Ensures that each language name is unique in the table.

Ensures that these columns must have values and cannot be empty.

12. **Bookreview**: The BookReview table is used to store information about reviews or feedback provided by users for books in the library. It allows users to share their opinions and ratings on specific books.

Constraints: Ensures that each Reviewid value in the table is unique and serves as the primary key for identifying reviews.

3.3 Relational Database Scheme

The relational database schema for ***Library management system*** database is as follows:

1. book (bookid, title, author, category, availablecopies)
2. borrower(borrowerid, firstname, lastname, email, phone, address)
3. transactions(transactionsid, userid, bookid, transactiondate, returndate, status)
4. category(categoryid, categoryname)
5. author (authorid, authorname, country, birthdate)
6. staff(staffid, staffname, position, email,mobile)
7. finepenalty(fine_id, borrowerid,amount, description,payment_status)
8. publisher(publisherid, publishername, address, contact_number,email)
9. bookcopies(copyid, bookid,location , availability_status)
10. reservations(reservationid, bookid, userid, reservation_date,status)
11. languages (languageid, bookid, languagename)
12. bookreview(reviewid, bookid, borrowerid reviewtext, reviewdate)

3.4 Relational Queries

```
/* create a table book*/
create table book
(
BookID              int                    primary key,
 Title              varchar(30)           not null,
 Author              INT                 not null
 Category              int                 not null
 AvailableCopies      int                 not null
)
 insert into Book values(101, 'X-Men: God Loves',401, 301, 98)
insert into Book values(102, 'Mike Tyson : Undisputed Truth',402, 302, 654)
insert into Book values(103, 'V for Vendetta',403, 303, 600)
insert into Book values(104, 'When Breath Becomes Air',404, 304, 500)
insert into Book values(105, 'The Great Gatsby',405, 305,120)
  select * from book
```

Output:

```
BOOKID     TITLE                          AUTHOR     CATEGORY    COPYS
101        X-Men: God Loves                 401         301         98
102        Mike Tyson : Undisputed Truth    402         302        655
103        V for Vendetta                   403         303        600
104        When Breath Becomes Air          404         304        500
105        The Great Gatsby                 405         305        120
106        To Kill a Mockingbird            406         304        150
107        Pride and Prejudice              407         302         53
108        Brave New World                  408          303        453
109        The Scarlet Letter               409          305        252
110        The Lord of the Rings            410          301        566
111         Adventures of Tom Sawyer        402          306        370
112         Ben Hur                         409          303        213
113        Baburnama                        404          302        233
114        Ancient Mariner                  407          305        100
115        Arms and the Man                 404          306        54
116        The Moon and Sixpence            402          301        233
```

```
117             Far from the Madding Crowd          404             304           203
118             Geetanjal                           406             302           203
119             Utopi                               403             308           100
/*create a table Borrower*/
create table Borrower
(
BorrowerID        int                  primary key,
 FirstName         varchar(30)        not null,
 LastName         varchar(30)        not null,
 Email              varchar(40)        not null,
 Phone              varchar(20)        not null,
 Address          varchar(200)         not null
)
insert into Borrower values(501, 'Robin', 'Steve', 'robin@gmail.com', 8937783783,
'Tirupati')
insert into Borrower values(502, 'Aadhya', 'Sree', 'aadhya100@gmail.com', 9783787873,
'Hyderabad')
insert into Borrower values(503, 'Sashi', 'Ambati', 'sashiambati62@gmail.com',
8774845673, 'Kurnool')
insert into        Borrower values(504, 'Shaffi', 'Shaik', 'shaffi199@gmail.com',9876548974,
'Guntur')
insert into         Borrower values (505,'Nithin', 'Ambati','nithin@gmail.com', 8769085468 ,
'VIJAYAWADA')

 select * from Borrower
```

Output:

```
ID   FRISTNAME   LASTNAME    EMAIL                   PHONE       ADDRESS
501     Robin         Steve       robin@gmail.com         8937783783   Tirupati
502     Aadhya         Sree       aadhya100@gmail.com     9783787873   Hyderabad
503     Sashi         Ambati      sashiambati62@gmail.com 8774845673   Kurnool
504     Shaffi        Shaik       shaffi199@gmail.com     9876548974   Guntur
505     Nithin        Ambati      nithin@gmail.com        8769085468   VIJAYAWADA
506     Nani          uppala      nani345@gmail.com       8769083543   nandal
507     vishnu         boya       vishnu243@gmail.com     826545468    alur
508     ramya         Ambati      ramya24@gmail.com       8784562132   VIJAYAWADA
509     basha         syed        basha23@gmail.com       9895621235   kurnool
510     asif          syed        asif786@gmail.com       8775658454   adoni
```

```
511   mohammed   qureshi   mohammed44@gmail.com      8907654332    kadapa
512   raju       boya      raju43@gmail.com          7654322123    gooty
513   vishnu     kumar     vishnu@gmail.com          6578905432    puttur
514   mahesh     konda     mahesh45@gmail.com        98765456784   chittoor
515   ravi       kumar     kavi55@gmail.com          6754324567    gudur
516   sudheer    ediga     sudheer45@gmail.com       6754348978    nellore
517   abi        kumar     abi876@gmail.com          9087345621    rajampet
518   kiran      sai       kiran78@gmail.com         5643236789    rayachoty
519   uppesh     kumar     uppesh786@gmail.com       7865453487    ongole
520   raffiq     mohammed  raffiq26@gmail.com        7032279243    ALUR
```

```
/*create a table Transactions*/
create table Transactions
(
  T_ID             int           PRIMARY KEY,
  UserID           int           not null foreign key references Borrower(BorrowerID),
  BookID           int           not null foreign key references Book(BookID),
  TransactionDate  DATE          not null,
  ReturnDate       DATE          not null,
  Status           varchar(100)  not null)
insert into Transactions values(201,503,101,'2023-03-21','2023-03-26','Returned')
insert into Transactions values(202,502,103,'2023-03-01','2023-03-05','Returned')
insert into Transactions values(203,504,105,'2023-04-15','2023-04-20','Not Returned')
insert into Transactions values(204,501,102,'2023-04-25','2023-04-30','Returned')
insert into Transactions values(205,505,104,'2023-02-18','2023-02-23','Not Returned')
select * from  Transactions
```

Output:

```
T_ID userID  BookID  TransactionDate  ReturnDate   Status
201   503     101     2023-03-21       2023-03-26   Returned
202   502     103     2023-03-01       2023-03-05   Returned
203   504     105     2023-04-15       2023-04-20   Not Returned
204   501     102     2023-04-25       2023-04-30   Returned
205   505     104     2023-02-18       2023-02-23   Not Returned
206   506     108     2023-03-01       2023-03-23   Not Returned
207   507     107     2023-03-04       2023-03-22   Returned
208   508     109     2023-03-25       2023-03-31   Not Returned
209   509     110     2023-03-06       2023-03-20   Returned
210   510     110     2023-03-24       2023-03-30   Not Returned
```

```
211     512     117     2023-04-11     2023-04-23     Returned
212     511     113     2023-03-15     2023-03-23     Returned
213     515     115     2023-04-21     2023-04-23     Not Returned
214     518     110     2023-03-04     2023-03-23     Not Returned
215     519     118     2023-03-16     2023-03-20      Returned
216     516     119     2023-02-16     2023-03-27     Not Returned
217     517     117     2023-01-11     2023-03-14      Returned
218     518     118     2023-02-19     2023-03-21      Returned
219     519     119     2023-03-01     2023-03-02     Not Returned
220     520     107     2023-02-04     2023-03-22     Not Returne
```

```
 /* create a table Category */
create table Category
(
       CategoryID         int                PRIMARY KEY,
      CategoryName    varchar(50)     not null
)

insert into Category values(301,'Comics')
insert into Category values(302,'Sports')
insert into Category values(303,' Comics ')
insert into Category values(3045,' Medical ')
insert into Category values(305,' Fiction ')
  select * from Category
```

Output:

```
CategoryID       CategoryName
301                Comics
302                Sports
303                Comics
304                Medical
305                Fiction
306                Adventure
307                Detective
308                Mystery
309                Fiction
```

```
310                    Historical Fiction
```

```
/* create a table Author */
create table Author
(
    AuthorID          int                  PRIMARY KEY,
    AuthorName     varchar(100)        not null,
    Country            varchar(50)          not null,
    BirthDate          DATE,
)

insert into Author values(401,'Chris','INDIA','1987-03-21')
insert into Author values(402,'Alan Moore','AUSTRALIA','1986-08-11')
insert into Author values(403,'Mike Tyson','NEW YORK','1965-06-02')
insert into Author values(404,'F. Scott Fitzgerald','JAPAN','1879-01-14')
insert into Author values(405,'Paul Kalanithi','AMERICA','1908-09-15')
  select * from Author
```

Output:

```
AuthorID      AuthorName            Country                 BirthDate
401              Chris                       INDIA                   1987-03-21
402              Alan Moore              AUSTRALIA           1986-08-11
403              Mike Tyson               NEW YORK            1965-06-02
404              F. Scott Fitzgerald      JAPAN                 1879-01-14
405              Paul Kalanithi            AMERICA              1908-09-15
406              Rabindranath Tagore    INDIA                  1861-06-12
407              Salman Rushdie           INDIA                  1850-02-08
408              Stephen King              AMERICA              1885-05-21
409              Mark Twain                 AMERICA              1770-08-02
410              Leo Tolstoy                 RUSSIA                1828-04-24
```

```
/*create table Staff */
create table Staff
(

  StaffID           INT              Primary Key,
Staffname         varchar(30)     not null,
position            varchar(30)   not null,
email               varchar(50)   not null,
mobile              varchar(10)   not null
)
  insert into Staff values(901,'rajesh','librarian','rajesha22@gamil.com',9936472821)
insert into Staff values(902,'mahesh','libraian','mahesh166@gmail.com',9865473821)
insert into Staff values(903,'babu','libraian assistant','babu55@gmail.com',7898654323)
insert into Staff values(904,'raju','labrary clerk','raju785@gmail.com',8786543221)
insertintoStaffvalues(905,'moham','librarymanager','moham786@gmail.com',7690765432)
insert into Staff values(906,'basha','senior librarian','basha254@gmail.com',8976546732)
select * from Staff
```

Output:

```
StaffID  Staffname   position               email                mobile
901      rajesh      librarian              rajesha22@gamil.com   9936472821
902      mahesh      libraian               mahesh166@gmail.com   9865473821
903      babu        libraian assistant     babu55@gmail.com      7898654323
904      raju        labrary clerk          raju785@gmail.com     8786543221
905      moham       library manager        moham786@gmail.com    7690765432
906      basha       senior librarian       basha254@gmail.com    8976546732
907      sudheer     Library Technician     sudheer54@gmail.com   9976556732
908      sandeep     Library Manager        sandeep4@gmail.com    9976546732
909      shaffi      Reference librarian    shaffi123@gmail.com   7976548732
910       verra      Medical librarian      veera@gmail.com       9976540732
```

```
/* create table FinePenalty */
Create table FinePenalty
(

fine_id            int              Primary Key,
 BorrowerID       int               not null foreign key references Borrower(BorrowerID),
amount            money           not null,
description       varchar(50)     not null,
payment_status    varchar(44)   not null
)

insert into FinePenalty values(601,503,10.5,'Late book return','unpaid')
insert into FinePenalty values(602,504,500,'lost book','paid')
insert into FinePenalty values(603,502,15,'Late book return','unpaid')
insert into FinePenalty values(604,507,50,'Damaged book','paid')
insert into FinePenalty values(605,506,700,'lost book','paid')
insert into FinePenalty values(606,505,150,'Damaged book','paid')
select * from FinePenalty
```

OUTPUT:

```
fine_id   BorrowerID   amount      description          payment_status
601          503          10.50       Late book return        unpaid
602          504          500.00      lost book               paid
603          502          15.00       Late book return        unpaid
604          507          50.00       Damaged book            paid
605          506          700.00      lost book               paid
606          505          150.00      Damaged book            paid
607          507          110.00      Damaged book            unpaid
608          509          100.00      lost book                paid
609          508          44.00       Late book return         paid
610          510          15.00       Late book return         unpaid
611          513          150.00      Late book return         unpaid
612          515          500.00      Llost book                paid
613          516          44.00       Damaged book              unpaid
614          512          100.00      Late book return          paid
```

```
615             518             300.00          lost book               unpaid
616             513             199.00          Late book return        unpaid
617             518             244.00          Damaged book             paid
618             511             143.00          Late book return         unpaid
619             519             22.00           Late book return         paid
620             520             444.00          lost book                unpaid
```

```
/*create table Bookcopies */
create table Bookcopies
(
copyID                  int             Primary Key,
 BookID                 int             not null foreign key references Book(BookID),
location                varchar(50)      not null,
availability_status     varchar(50)       not null
)
insert into Bookcopies values(1,101,'Shelf A-1','Available')
insert into Bookcopies values(2,104,'Shelf A-2','Available')
insert into Bookcopies values(3,103,'Shelf B-2','Borrowed')
insert into Bookcopies values(4,102,'Shelf B-1','Borrowed')
insert into Bookcopies values(5,105,'Shelf C-2','Available')
insert into Bookcopies values(6,106,'Shelf C-2','Available')
```

Output:

```
copyID   BookID    location     availability_status
1         101      Shelf A-1    Available
2         104      Shelf A-2    Available
3         103      Shelf B-2    Borrowed
4         102      Shelf B-1    Borrowed
5         105      Shelf C-2    Available
6         106      Shelf C-2    Available
7         107      Shelf D-1    Borrowed
8         107      Shelf E-2    Available
9         109      Shelf C-3    Available
10        115      Shelf D-2    Borrowed
11        114      Shelf B-1    Available
12        113      Shelf E-2    Available
```

```
13          113          Shelf C-1       Borrowed
14          119          Shelf A-2       Available
15          106          Shelf B-1       Borrowed
16          117          Shelf E-1       Available
17          117          Shelf A-3       Available
18          109          Shelf C-3       Borrowed
19          108          Shelf B-3       Borrowed
20          118          Shelf D-3       Available
```

```
/*create table Reservations*/
create table Reservations
(
ReservationID           int        Primary Key,
BookID                  int        not null foreign key references Book(BookID),
UserID                  int        not null foreign key references Borrower(BorrowerID),
reservation_date        date       not null,
status                  varchar(50)  not null
)
insert into Reservations values(1001,102,502,'2023-03-21','Active')
insert into Reservations values(1002,104,505,'2023-03-27','Active')
insert into Reservations values(1003,101,503,'2023-03-30','Cancelled')
insert into Reservations values(1004,105,501,'2023-04-05','Active')
insert into Reservations values(1005,103,504,'2023-04-21','Cancelled')
insert into Reservations values(1006,106,503,'2023-04-15','Cancelled')
insert into Reservations values(1007,107,507,'2023-04-11','Active')
insert into Reservations values(1008,108,506,'2023-04-05','Cancelled')
select * from  Reservations
```

Output:

ReservationID	BookID	UserID	reservation_date	status
1001	102	502	2023-03-21	Active
1002	104	505	2023-03-27	Active
1003	101	503	2023-03-30	Cancelled
1004	105	501	2023-04-05	Active
1005	103	504	2023-04-21	Cancelled

```
1006                106    503           2023-04-15    Cancelled
1007                107    507           2023-04-11    Active
1008                108    506           2023-04-05    Cancelled
1009                109    508           2023-04-30    Active
1010                110    509           2023-04-24    Active
1011                110    510           2023-05-02    Active
1012                113    518           2023-05-20    Cancelled
1013                113    514           2023-05-04    Cancelled
1014                114    514           2023-05-16    Active
1015                115    511           2023-05-19    Active
1016                114    516           2023-05-21    Active
1017                119    515           2023-05-24    Active
1018                115    520           2023-06-01    Active
1019                119    519           2023-06-15    Cancelled
1020                118    520           2023-06-24    Active
```

```
/*create table Languages*/
create table Languages
(
   LanguageID      INT                 PRIMARY KEY,
   BookID          int                 not null foreign key references Book(BookID),
   LanguageName    VARCHAR(50)         NOT NULL
)
insert into Languages  values(1101,101,'ENGLISH')
insert into Languages  values(1102,102,'SPANISH')
insert into Languages  values(1103,104,'FRENCH')
insert into Languages  values(1104,107,'RUSSIAN')
insert into Languages  values(1105,109,'ARABIC')
insert into Languages  values(1106,108,'ENGLISH')
select * from  Languages
```

Output:

```
LanguageID   BookID    LanguageName
1101          101          ENGLISH
1102          102          SPANISH
1103          104          FRENCH
```

```
1104            107             RUSSIAN
1105            109             ARABIC
1106            108             ENGLISH
1107            109             ENGLISH
1108            110             FRENCH
1109            113             ENGLISH
1110            115             ENGLISH
1111            117             FRENCH
1112            120             ENGLISH
```

SQL QUERIES:

Query1:Retrieve books with available copies greater than or equal to 500.

```
SELECT * FROM book WHERE AvailableCopies >=500;
```

/*output:

```
Book_id      Title                          Author   Category   AvailableCopies
102          Mike Tyson : Undisputed Truth   402      302        654
103          V for Vendetta                  403      303        600
104          When Breath Becomes Air         404      304        500
110          The Lord of the Rings           410      301        566*/
```

Query2: Retrieve the book title and author name for all books.

```
  SELECT Book.Title, Author.AuthorName
FROM Book
JOIN Author ON Book.Author = Author.AuthorID;
```

/*output:

```
  Title                              AuthorName
X-Men: God Loves                       Chris
Mike Tyson : Undisputed Truth          Alan Moore
V for Vendetta                         Mike Tyson
When Breath Becomes Air                F. Scott Fitzgerald
The Great Gatsby                       Paul Kalanithi
To Kill a Mockingbird                  Rabindranath Tagore
Pride and Prejudice                    Salman Rushdie
Brave New World                        Stephen King
```

```
The Scarlet Letter                  Mark Twain
The Lord of the Rings               Leo Tolstoy
Adventures of Tom Sawyer            Alan Moore
Ben Hur                             Mark Twain
Baburnama                           F. Scott Fitzgerald
Ancient Mariner                     Salman Rushdie
Arms and the Man                    F. Scott Fitzgerald
The Moon and Sixpence               Alan Moore
Far from the Madding Crowd          F. Scott Fitzgerald
Geetanjal                           Rabindranath Tagore
Utopi                                Mike Tyson
*/
```

Query3: Retrieve all books borrowed by a specific borrower.

```
SELECT Book.*
FROM Book
JOIN Transactions ON Book.BookID = Transactions.BookID
WHERE Transactions.UserID = 502;
/*output
Bookis    Title             Author  Category   AvailableCopies
  103    V for Vendetta      403     303               600    */
```

Query4: Retrieve the books published by a specific author.

```
SELECT Book.*
FROM Book
JOIN Author ON Book.Author = Author.AuthorID
WHERE Author.AuthorName = 'Chris';
/* output:
Bookis    Title                 Author  Category   AvailableCopies
101     X-Men: God Loves       401        301          98 */
```

Query5: Retrieve the borrowers who have unpaid fine penalties.

```
SELECT Borrower.*
FROM Borrower
JOIN FinePenalty ON Borrower.BorrowerID = FinePenalty.BorrowerID
```

```
WHERE FinePenalty.payment_status = 'unpaid';
```

/* output:

```
Borrowerid  Firstname   Lastname    Email                          Phone        Address
503            Sashi      Ambati      sashiambati62@gmail.com    8774845673   Kurnool
502           Aadhya      Sree        aadhya435@gmail.com       9354978378  Hyderabad
507           vishnu      boya          vishnu254@gmail.com       826545468    alur
510           asif        syed        asif786@gmail.com            8775658454   adoni
513           vishnu      kumar       vishnu@gmail.com             6578905432   puttur
516           sudheer     ediga       sudheer45@gmail.com          6754348978   nellore
518         kiran       sai           kiran78@gmail.com           5643236789  rayachoty
513        vishnu      kumar          vishnu@gmail.com            6578905432   puttur
511      mohammed  qureshi       mohammed44@gmail.com      8907654332   kadapa
520        RAFFIQ    mohammed    raffiq26@gmail.com           7032279243   ALUR
```

Query6: Retrieve the first name, last name, and email of borrowers who have their reservation status as cancelled.

```
SELECT Borrower.FirstName, Borrower.LastName, Borrower.Email
FROM Borrower
INNER JOIN Reservations ON Borrower.BorrowerID = Reservations.BorrowerID
WHERE Reservations.status = 'Cancelled';
```

/* output:

```
Firstname    Lastname          Email
Sashi          Ambati        sashiambati62@gmail.com
Shaffi         Shaik         shaffi199@gmail.com
Sashi          Ambati        sashiambati62@gmail.com
Nani           uppala        nani34@gmail.com
kiran          sai           kiran78@gmail.com
mahesh         konda         mahesh45@gmail.com
uppesh         kumar         uppesh786@gmail.com*/
```

Query7:Retrieve the first name, last name, and email of borrowers who have not returned their borrowed books.

```
SELECT Borrower.FirstName, Borrower.LastName, Borrower.Email
FROM Borrower
INNER JOIN Transactions ON Borrower.BorrowerID = Transactions.BorrowerID
WHERE Transactions.Status = 'Not Returned';
```

```
/* output:
Firstname    Lastname          Email
Shaffi        Shaik          shaffi199@gmail.com
Nithin        Ambati         nithin@gmail.com
Nani          uppala         nani34@gmail.com
ramya         Ambati         ramya24@gmail.com
asif          syed           asif786@gmail.com
ravi          kumar          kavi55@gmail.com
kiran         sai            kiran78@gmail.com
sudheer       ediga          sudheer45@gmail.com
uppesh        kumar          uppesh786@gmail.com
RAFFIQ        mohammed       raffiq26@gmail.com
  */
```

Query8: Retrieve the title and author of books in the "Fiction" category.

```
SELECT Book.Title, Author.AuthorName
FROM Book
INNER JOIN Author ON Book.Author = Author.AuthorID
INNER JOIN Category ON Book.Category = Category.CategoryID
WHERE Category.CategoryName = 'Fiction';
 /* output:
Title                      AuthorName
The Great Gatsby       Paul Kalanithi
The Scarlet Letter     Mark Twain
Ancient Mariner        Salman Rushdie   */
```

Query9: Retrieve the first name, last name, and amount of each borrower with a fine penalty.

```
SELECT Borrower.FirstName, Borrower.LastName, FinePenalty.amount
FROM Borrower
INNER JOIN FinePenalty ON Borrower.BorrowerID = FinePenalty.BorrowerID;
 /* output:
FirstName     LastName    Amount
Sashi          Ambati        10.50
Shaffi         Shaik         500.00
```

```
Aadhya     Sree       15.00
vishnu     boya       50.00
Nani       uppala     700.00
Nithin     Ambati     150.00
vishnu     boya       110.00
basha      syed       100.00
ramya      Ambati     44.00
asif       syed       15.00
vishnu     kumar      150.00
ravi       kumar      500.00
sudheer    ediga      44.00
raju       boya       100.00
kiran      sai        300.00
vishnu     kumar      199.00
kiran      sai        244.00
mohammed   qureshi    143.00
uppesh     kumar      22.00
RAFFIQ     mohammed   444.00     */
```

Query10: Retrieve the average fine amount for each category:

```
SELECT Category.CategoryName, AVG(FinePenalty.amount) AS AverageFine
FROM Category
JOIN Book ON Category.CategoryID = Book.Category
JOIN Transactions ON Book.BookID = Transactions.BookID
JOIN FinePenalty ON Transactions.BorrowerID = FinePenalty.BorrowerID
GROUP BY Category.CategoryName;
```

```
/* output:
CategoryName     AverageFine
Comics             180.8333
Fiction            348.00
Medical            179.00
Sports             134.00*/
```

Query11: Retrieve the books that have not been reserved.

```
SELECT B.Title
```

```
FROM Book B
LEFT JOIN Reservations R ON B.BookID = R.BookID
WHERE R.ReservationID IS NULL;
/* output:
     Title
    Adventures of Tom Sawyer
    Arms and the Man
    The Moon and Sixpence
     Utopi            */
```

Query12: Retrieve the borrowers who have made reservations on a specific date.

```
SELECT DISTINCT Br.BorrowerID, Br.FirstName, Br.LastName
FROM Borrower Br
JOIN Reservations R ON Br.BorrowerID = R.UserID
WHERE R.reservation_date = '2023-04-05';
/*output:
BorrowerID      FirstName      LastNmae
501                Robin              Steve
506                 Nani              uppala  */
```

Query13 Retrieve the number of reservations made for each book.

```
SELECT B.Title, COUNT(*) AS NumberOfReservations
FROM Book B
LEFT JOIN Reservations R ON B.BookID = R.BookID
GROUP BY B.Title;
/* output:
Title                                   NumberOfReservations
Adventures of Tom Sawyer                        1
Ancient Mariner                                 2
Arms and the Man                                1
Baburnama                                       2
Ben Hur                                         2
Brave New World                                 1
Far from the Madding Crowd                      1
Geetanjal                                       2
Mike Tyson : Undisputed Truth                   1
```

```
Pride and Prejudice                          1
The Great Gatsby                             1
The Lord of the Rings                        2
The Moon and Sixpence                        1
The Scarlet Letter                           1
To Kill a Mockingbird                        1
Utopi                                        1
V for Vendetta                               1
When Breath Becomes Air                      1
X-Men: God Loves                             1    */
```

Query14 Retrieve the books with the highest fine amount.

```
SELECT B.Title, MAX(FP.amount) AS HighestFineAmount
FROM Book B
JOIN Transactions T ON B.BookID = T.BookID
JOIN FinePenalty FP ON T.BorrowerID = FP.BorrowerID
GROUP BY B.Title;
```

```
/* output:
Title                          HighestFineAmount
Ancient Mariner                     500.00
Ben Hur                             143.00
Brave New World                     700.00
Far from the Madding Crowd          300.00
Geetanjal                           44.00
Pride and Prejudice                 444.00
The Great Gatsby                    500.00
The Lord of the Rings               300.00
The Moon and Sixpence               100.00
The Scarlet Letter                  44.00
V for Vendetta                      15.00
When Breath Becomes Air             150.00
X-Men: God Loves                    10.50  */
```

Query15 Retrieve the borrowers who have not made any fine payments.

```
SELECT Br.BorrowerID, Br.FirstName, Br.LastName
FROM Borrower Br
```

```
LEFT JOIN FinePenalty FP ON Br.BorrowerID = FP.BorrowerID
WHERE FP.fine_id IS NULL
;/* output:
BorrowerID      FirstName      LastNmae
501                 Robin             Steve
514                mahesh             konda
517                    abi             kumar*/
```

Query16: Retrieve the details of books that are available and located on a specific shelf:

```
SELECT b.Title, bc.location
FROM Book b
JOIN Bookcopies bc ON b.BookID = bc.BookID
WHERE bc.availability_status = 'Available' AND bc.location = 'Shelf A-1';
/* output:
       Title                      Location
  X-Men: God Loves      Shelf A-1     */
```

Query17 Retrieve the books written by authors from a specific country:

```
SELECT b.Title, a.AuthorName
FROM Book b
JOIN Author a ON b.Author = a.AuthorID
WHERE a.Country = 'INDIA';
/* output:
Title                           AuthorName
X-Men: God Loves                  Chris
To Kill a Mockingbird             Rabindranath Tagore
Pride and Prejudice               Salman Rushdie
Ancient Mariner                   Salman Rushdie
Geetanjal                         Rabindranath Tagore     */
```

Query18: Retrieve the borrowers who have never borrowed a book:

```
SELECT bo.FirstName, bo.LastName
FROM Borrower bo
LEFT JOIN Transactions t ON bo.BorrowerID = t.BorrowerID
WHERE t.BorrowerID IS NULL;
```

```
/* output:
FirstName      LastNmae
vishnu               kumar
mahesh               konda   */
```

Query19: Retrieve the details of books with titles starting with 'The':

```
SELECT Title
FROM Book
WHERE Title LIKE 'The%';
```

```
/* output:
Title
The Great Gatsby
The Scarlet Letter
The Lord of the Rings
The Moon and Sixpence   */
```

Query20: Retrieve the details of books borrowed and returned within a specific date range:

```
SELECT b.Title, t.TransactionDate, t.ReturnDate
FROM Book b
JOIN Transactions t ON b.BookID = t.BookID
WHERE t.TransactionDate BETWEEN '2023-01-01' AND '2023-06-01'
  AND t.ReturnDate BETWEEN '2023-01-01' AND '2023-06-01';
```

```
/* output:
  Title                            TransactionDate    ReturnDate
 X-Men: God Loves                      2023-03-21     2023-03-26
V for Vendetta                         2023-03-01     2023-03-05
The Great Gatsby                       2023-04-15     2023-04-20
Mike Tyson : Undisputed Truth          2023-04-25     2023-04-30
When Breath Becomes Air                2023-02-18     2023-02-23
Brave New World                        2023-03-01     2023-03-23
Pride and Prejudice                    2023-03-04     2023-03-22
The Scarlet Letter                     2023-03-25     2023-03-31
The Lord of the Rings                  2023-03-06     2023-03-20
The Lord of the Rings                  2023-03-24     2023-03-30
The Moon and Sixpence                  2023-04-11     2023-04-23
```

```
Ben Hur                         2023-03-15    2023-03-23
Ancient Mariner                 2023-04-21    2023-04-23
The Lord of the Rings           2023-03-04    2023-03-23
Far from the Madding Crowd      2023-03-16    2023-03-20
Geetanjal                       2023-02-16    2023-03-27
The Moon and Sixpence           2023-01-11    2023-03-14
Far from the Madding Crowd      2023-02-19    2023-03-21
Geetanjal                       2023-03-01    2023-03-02
Pride and Prejudice             2023-02-04    2023-03-22*/
```

Query21: Retrive all books with their titles, authors, and categories

```
SELECT Book.Title, Author.AuthorName, Category.CategoryName
FROM Book
JOIN Author ON Book.Author = Author.AuthorID
JOIN Category ON Book.Category = Category.CategoryID;
```

/* output:

```
Title                          AuthorName          CategoryName
X-Men: God Loves               Chris                  Comics
Mike Tyson : Undisputed Truth  Alan Moore             Sports
V for VendettaMike             Tyson                  Comics
When Breath Becomes Air        F. Scott Fitzgerald    Medical
The Great Gatsby               Paul Kalanithi         Fiction
To Kill a Mockingbird          Rabindranath Tagore    Medical
Pride and Prejudice            Salman Rushdie         Sports
Brave New World                Stephen King           Comics
The Scarlet Letter             Mark Twain             Fiction
The Lord of the Rings          Leo Tolstoy            Comics
Adventures of Tom Sawyer       Alan Moore             Adventure
Ben Hur                        Mark Twain             Comics
Baburnama                      F. Scott Fitzgerald    Sports
Ancient Mariner                Salman Rushdie         Fiction
Arms and the Man               F. Scott Fitzgerald    Adventure
The Moon and Sixpence          Alan Moore             Comics
Far from the Madding Crowd     F. Scott Fitzgerald    Medical
Geetanjal                      Rabindranath Tagore    Sports
```

```
Utopi                                   Mike Tyson          Mystery */
```

Query22: Retrive all books with their authors and the number of times each book has been borrowed

```
SELECT Book.Title, Author.AuthorName, COUNT(*) AS BorrowCount
FROM Book
JOIN Transactions ON Book.BookID = Transactions.BookID
JOIN Author ON Book.Author = Author.AuthorID
GROUP BY Book.Title, Author.AuthorName;
```

```
 /* output:
Totle                                   AuthorName      BorrowCount
Mike Tyson : Undisputed Truth             Alan Moore            1
The Moon and Sixpence                     Alan Moore            2
X-Men: God Loves                          Chris                 1
Far from the Madding Crowd                F. Scott Fitzgerald   2
When Breath Becomes Air                   F. Scott Fitzgerald   1
The Lord of the Rings                     Leo Tolstoy           3
Ben Hur                                   Mark Twain            1
The Scarlet Letter                        Mark Twain            1
V for Vendetta                            Mike Tyson            1
The Great Gatsby                          Paul Kalanithi        1
Geetanjal                                 Rabindranath Tagore   2
Ancient Mariner                           Salman Rushdie        1
Pride and Prejudice                       Salman Rushdie        2
Brave New World                           Stephen King          1 */
```

Query23List the authors who have written books in multiple categories

```
SELECT Author.AuthorName
FROM Author
JOIN Book ON Author.AuthorID = Book.Author
GROUP BY Author.AuthorName
HAVING COUNT(DISTINCT Book.Category) > 1;
```

```
/* output:
AuthorName
```

```
Alan Moore
F. Scott Fitzgerald
Mark Twain
Mike Tyson
Rabindranath Tagore
Salman Rushdie*/
```

Query24: List the staff members who have not performed any transactions:

```
SELECT Staff.Staffname
FROM Staff
LEFT JOIN Transactions ON Staff.StaffID = Transactions.UserID
WHERE Transactions.TransactionID IS NULL;
/* output:
StaffName
rajesh
mahesh
babu
raju
mohammed
basha
sudheer
sandeep
shaffi
verra */
```

Query25: Retrieve the books that are available in a specific location:

```
SELECT Book.Title, Bookcopies.location
FROM Book
JOIN Bookcopies ON Book.BookID = Bookcopies.BookID
WHERE Bookcopies.location = 'Shelf B-1';
/* output:
Title                              location
Mike Tyson : Undisputed Truth      Shelf B-1
Baburnama                          Shelf B-1
```

```
To Kill a Mockingbird                Shelf B-1 */
```

Query 26: Find the total number of transactions for each staff member:

```
SELECT Staff.Staffname, COUNT(Transactions.TransactionID) AS TotalTransactions
FROM Staff
LEFT JOIN Transactions ON Staff.StaffID = Transactions.UserID
GROUP BY Staff.Staffname;
/*output:
 Staffname        TotalTransactions
babu                   0
basha                  0
mahesh                 0
mohammed               0
rajesh                 0
raju                   0
Sandeep                0
shaffi                 0
sudheer                0
verra                  0  */
```

Query 27 List the borrowers who have paid all their fines:

```
SELECT Borrower.FirstName, Borrower.LastName
FROM Borrower
LEFT JOIN FinePenalty ON Borrower.BorrowerID = FinePenalty.BorrowerID
GROUP BY Borrower.FirstName, Borrower.LastName
HAVING COUNT(FinePenalty.fine_id) = 0;
/*output:
FirstName   LastName
mahesh        konda
abi           kumar
Robin         Steve*/
```

Query 28: Retrieve the number of books published in each category.

```
SELECT C.CategoryName, COUNT(*) AS NumberOfBooks
FROM Category C
JOIN Book B ON C.CategoryID = B.Category
GROUP BY C.CategoryName;
```

```
/*output:
CategoryName          NumberOfBooks
Adventure                   2
Comics                      6
Fiction                     3
Medical                     3
Mystery                     1
Sports                      4   */
```

Query 29: Retrieve all books with multiple authors.

```
SELECT Book.Title, COUNT(*) AS NumAuthors
FROM Book
JOIN Author ON Book.Author = Author.AuthorID
GROUP BY Book.Title
HAVING COUNT(*) > 1;
```

```
/*output:
Title        NumAuthors
*/
```

Query 30: Retrieve the books with their corresponding languages.

```
SELECT Book.Title, Languages.LanguageName
FROM Book
JOIN Languages ON Book.BookID = Languages.BookID;
```

```
 /*output:
Title                              LanguageName
X-Men: God Loves                      ENGLISH
Mike Tyson : Undisputed Truth         SPANISH
When Breath Becomes Air               FRENCH
Pride and Prejudice                   RUSSIAN
The Scarlet Letter                    ARABIC
Brave New World                       ENGLISH
The Scarlet Letter                    ENGLISH
The Lord of the Rings                 FRENCH
Ben Hur                               ENGLISH
Ancient Mariner                       ENGLISH
```

```
The Moon and Sixpence                FRENCH
Utopi   ENGLISH */
```

Query31: Retrieve the books borrowed by borrowers whose names start with "A"

```
SELECT Book.BookID, Book.Title
FROM Book
JOIN Transactions ON Book.BookID = Transactions.BookID
JOIN Borrower ON Transactions.UserID = Borrower.BorrowerID
WHERE Borrower.FirstName LIKE 'A%';
/* output:
BookID   Title
103       V for Vendetta
110       The Lord of the Rings
117       The Moon and Sixpence  */
```

Query32: Retrieve the staff members who have not borrowed any books:

```
SELECT s.staffid, s.staffname, s.position, s.email, s.mobile
FROM Staff s
WHERE NOT EXISTS (
   SELECT 1
   FROM Transactions t
   WHERE t.UserID = s.StaffID
/* output:
Staffed        staffname    position              email                       mobile
901            rajesh       librarian             rajesha22@gamil.com         9936472821
902            mahesh       libraian              mahesh166@gmail.com         9865473821
903            babu         libraian assistant    babu55@gmail.com            7898654323
904            raju         labrary clerk         raju785@gmail.com           8786543221
905         mohammed        library manager       mohammed786@gmail.com 7690765432
906            basha        senior librarian      basha254@gmail.com 8976546732
907            sudheer    Library Technician      sudheer54@gmail.com         9976556732
908            sandeep      Library Manager       sandeep4@gmail.com 9976546732
909            shaffi       Reference librarian   shaffi123@gmail.com 7976548732
910            verra        Medical librarian     veera@gmail.com             9976540732
*/
```

Query33:Retrieve the reservations with the earliest reservation date:

```
SELECT r. *
FROM Reservations r
WHERE reservation_date = (
   SELECT MIN(reservation_date)
   FROM Reservations
);
```

/* output:

```
ReservationID  BookID   UserID   Reservation_date      Status
1001                 102          502      2023-03-21               Active      */
```

Query 34:Retrieve the reservations made by users with a specific email domain :

```
SELECT r.*
FROM Reservations r
INNER JOIN Borrower u ON r.UserID = u.BorrowerID
WHERE u.Email LIKE 'sashiambati62@gmail.com';
```

/* output:

```
ReservationID  BookID   UserID   Reservation_date      Status
1003             101              503       2023-03-30           Cancelled
1006             106              503       2023-04-15           Cancelled
*/
```

Query35: Retrieve the book copies that are available and have not been reserved:

```
SELECT bc.*
FROM Bookcopies bc
LEFT JOIN Reservations r ON bc.BookID = r.BookID
WHERE bc.availability_status = 'Available'
```

Query

```
AND r.ReservationID IS NULL;
```

/* output:

```
CopyID  BookID     Location    availability_Status
16            117          Shelf E-1        Available
17            117          Shelf A-3        Available
*/
```

Query 36.Retrieve the details of all languages along with the number of books in each language:

```
SELECT l.LanguageID, l.LanguageName, COUNT(b.BookID) AS BookCount
FROM Languages l
LEFT JOIN Book b ON l.BookID = b.BookID
```

Query

```
GROUP BY l.LanguageID, l.LanguageName;
/* output:
LanguageID   LanguageName   BookCount
1101         ENGLISH              1
1102         SPANISH              1
1103         FRENCH               1
1104         RUSSIAN              1
1105         ARABIC               1
1106         ENGLISH              1
1107         ENGLISH              1
1108         FRENCH               1
1109         ENGLISH              1
1110         ENGLISH              1
1111         FRENCH               1
1112         ENGLISH              1
*/
```

Query 37.Retrieve the languages with the highest number of books:

```
SELECT l.LanguageID, l.LanguageName
FROM Languages l
INNER JOIN Book b ON l.BookID = b.BookID
GROUP BY l.LanguageID, l.LanguageName
HAVING COUNT(b.BookID) = (
   SELECT MAX(BookCount)
   FROM (
      SELECT COUNT(b2.BookID) AS BookCount
      FROM Book b2
      GROUP BY b2.BookID
   ) AS SubQuery
```

Query

```
);
```

```
/* output:
LanguageID   LanguageName
1101             ENGLISH
1102             SPANISH
1103             FRENCH
1104             RUSSIAN
1105             ARABIC
1106             ENGLISH
1107             ENGLISH
1108             FRENCH
1109             ENGLISH
1110             ENGLISH
1111             FRENCH
1112             ENGLISH
*/
```

Query38: Retrieve the book copies that have been borrowed and returned within a specific date range:

```
SELECT bc.*
FROM Bookcopies bc
INNER JOIN Transactions t ON bc.BookID = t.BookID
WHERE t.Status = 'Returned'
```

Query

```
AND t.TransactionDate BETWEEN '2023-01-01' AND '2023-06-01';
```

```
/* output:
copyID   BookID    Loacation   availability_Status
1             101    Shelf A-1       Available
3             103    Shelf B-2       Borrowed
4             102    Shelf B-1       Borrowed
12            113    Shelf E-2       Available
13            113    Shelf C-1       Borrowed
16            117    Shelf E-1       Available
17            117    Shelf A-3       Available
*/
```

Query39: Retrieve the reservations with the highest reservation ID:

```
SELECT *
FROM Reservations
WHERE ReservationID = (
   SELECT MAX(ReservationID)
   FROM Reservations
```
Query

```
);
```
/* output:

```
ReservationID   BookID   UserID   Reservation_date      Status
1020              118      520    2023-06-24              Active
*/
```
Query40: Retrieve the books that have been borrowed the most times.

```
SELECT Book.BookID, Book.Title
FROM Book
JOIN Transactions ON Book.BookID = Transactions.BookID
GROUP BY Book.BookID, Book.Title
```
Query

```
HAVING COUNT(Transactions.TransactionID) = (SELECT MAX(TransactionCount)
FROM (SELECT COUNT(TransactionID) AS TransactionCount FROM Transactions
GROUP BY BookID) AS T);
```
/* output:

```
BookID          Title
110         The Lord of the Rings
*/
```
Query41: Retrieve the books that have been reviewed more than 1 time:

```
SELECT b.BookID, b.Title
FROM Book b
JOIN (
   SELECT BookID, COUNT(*) AS ReviewCount
   FROM BookReview
   GROUP BY BookID
   HAVING COUNT(*) > 1
```
Query

```
) AS R ON b.BookID = R.BookID;
```

```
/* output:
BookID          Title
106          To Kill a Mockingbird
*/
```

Query42: Retrieve the member who has written the most book reviews:

```
SELECT bw.BorrowerID, CONCAT(bw.FirstName, ' ', bw.LastName) AS ReviewerName,
COUNT(*) AS ReviewCount
FROM Borrower bw
JOIN BookReview br ON bw.BorrowerID = br.BorrowerID
GROUP BY bw.BorrowerID, FirstName,LastName
HAVING COUNT(*) = (
   SELECT MAX(ReviewCount)
   FROM (
      SELECT bw.BorrowerID, COUNT(*) AS ReviewCount
      FROM Borrower bw
      JOIN BookReview br ON bw.BorrowerID = br.BorrowerID
      GROUP BY bw.BorrowerID
   ) AS T
Query
);
/* output:
BorrowerID   ReviewerName,     ReviewCount
505             Nithin Ambati                    2
*/
```

Query43: Retrieve the books with the most recent reviews:

```
SELECT b.BookID, b.Title, br.ReviewText, br.ReviewDate
FROM Book b
JOIN BookReview br ON b.BookID = br.BookID
WHERE br.ReviewDate = (
   SELECT MAX(ReviewDate)
   FROM BookReview
Query
);
/* output:
```

```
BookID        Title                   Reviewtext                           ReviewDate
  114          Baburnama         The plot twists kept me on the edge of my seat. Highly
recommend!    2023-06-17
*/
```

Query44: Retrieve the books that have at least one review and are currently available:

```
SELECT b.BookID, b.Title
FROM Book b
JOIN BookReview br ON b.BookID = br.BookID
JOIN BookCopies bc ON b.BookID = bc.BookID
```

Query

```
WHERE bc.availability_status = 'Available';
```

/* output:

```
BookID                  Title
104             When Breath Becomes Air
105             The Great Gatsby
106             To Kill a Mockingbird
106             To Kill a Mockingbird
109             The Scarlet Letter
114             Baburnama
118             Far from the Madding Crowd
*/
```

Query45: Retrieve book reviews with their corresponding book titles:

```
SELECT br.ReviewID, br.ReviewText, br.ReviewDate, bo.FirstName, bo.LastName
FROM BookReview br
```

Query

```
JOIN Borrower bo ON br.BorrowerID = bo.BorrowerID;
```

/* output:

```
 ReviewID                    Reviewtext                              ReviewDate  Firstname Lastname
1201    I really enjoyed this book. Highly recommended.        2023-06-15   Aadhya    Sree
1202    The characters were well-developed, but the plot .      2023-06-10    Nithin  Ambati
1203    This book had a slow start, but it picked up towards .2023-06-12     Sashi    Ambati
1204    The author did an excellent job of creating suspenseful. 2023-06-09 kiran      sai
1205    I found the writing style to be captivating.                2023-06-12  mahesh konda
1206    The book was a bit too long for my liking.                2023-06-1     ramya    Ambati
```

```
1207    I could not put this book down.                          2023-06-11    Nithin  Ambati
1208    The plot twists kept me on the edge of my seat.          2023-06-17     ravi    kumar
1209    The characters felt shallow and lacked depth.            2023-06-08    raffiq  moham
1210    I was disappointed by the ending.                        2023-06-16    asif     syed
*/
```

Query46: Retrieve books with no reviews:

```
SELECT b.BookID, b.Title
FROM Book b
LEFT JOIN BookReview br ON b.BookID = br.BookID
```

Query

```
WHERE br.ReviewID IS NULL;
```

/* output:

```
BookID          Title
101             X-Men: God Loves
103             V for Vendetta
107             Pride and Prejudice
108             Brave New World
110             The Lord of the Rings
113             Ben Hur
115             Ancient Mariner
116             Arms and the Man
117             The Moon and Sixpence
119             Geetanjal
*/
```

Query47: Retrieve book reviews written by a specific borrower:

```
SELECT br.ReviewID, br.ReviewText, br.ReviewDate, bo.FirstName, bo.LastName
FROM BookReview br
JOIN Borrower bo ON br.BorrowerID = bo.BorrowerID
```

Query

```
WHERE bo.FirstName = 'mahesh' AND bo.LastName = 'konda';
```

/* output:

```
ReviewID              Reviewtext                         ReviewDate  Firstname  Lastname
1205    I found the writing style to be captivating and lyrical. 2023-06-14   mahesh    konda
*/
```

Query48: Retrieve book reviews for books in a specific category:

```
SELECT br.ReviewID, br.ReviewText, br.ReviewDate, b.Title, c.CategoryName
FROM BookReview br
JOIN Book b ON br.BookID = b.BookID
JOIN Category c ON b.Category = c.CategoryID
```

Query

```
WHERE c.CategoryName = 'Mystery';
```

/* output:

```
ReviewID            Reviewtext                        ReviewDate    Title   CategoryName
1209    The characters felt shallow and lacked depth.    2023-06-08    Utopi    Mystery
*/
```

Query49: Retrieve the total number of reviews for each book:

```
SELECT b.BookID, b.Title, COUNT(br.ReviewID) AS ReviewCount
FROM Book b
LEFT JOIN BookReview br ON b.BookID = br.BookID
```

Query

```
GROUP BY b.BookID, b.Title;
```

/* output:

```
BookID    Title                            ReviewCount
101       X-Men: God Loves                      0
102       Mike Tyson : Undisputed Truth         1
103       V for Vendetta                        0
104       When Breath Becomes Air               1
105       The Great Gatsby                      1
106       To Kill a Mockingbird                 2
107       Pride and Prejudice                   0
108       Brave New World                       0
109       The Scarlet Letter                    1
110       The Lord of the Rings                 0
111       Adventures of Tom Sawyer              1
113       Ben Hur                               0
114       Baburnama                             1
115       Ancient Mariner                       0
116       Arms and the Man                      0
```

```
117     The Moon and Sixpence               0
118     Far from the Madding Crowd          1
119     Geetanjal                           0
120     Utopi                               1
*/
```

Query50: Retrieve the borrowers information and the title of the book they most recently borrowed:

```
SELECT bo.FirstName, bo.LastName, b.Title
FROM Borrower bo
JOIN (
    SELECT t.UserID, t.BookID
    FROM Transactions t
    WHERE t.TransactionDate = (
        SELECT MAX(TransactionDate)
        FROM Transactions
        WHERE UserID = t.UserID
    )
) AS latest ON bo.BorrowerID = latest.UserID
JOIN Book b ON latest.BookID = b.BookID;
```

/* output:

```
Firstname     Lastname       Title
RAFFIQ        mohammed     Pride and Prejudice
Uppesh        kumar        Far from the Madding Crowd
kiran         sai          The Lord of the Rings
abi           kumar        The Moon and Sixpence
sudheer       ediga        Geetanjal
ravi          kumar        Ancient Mariner
raju          boya         The Moon and Sixpence
mohammed      qureshi      Ben Hur
asif          syed         The Lord of the Rings
basha         syed         The Lord of the Rings
ramya         Ambati       The Scarlet Letter
Vishnu        boya         Pride and Prejudice
Nani          uppala       Brave New World
```

```
Nithin      Ambati      When Breath Becomes Air
Shaffi      Shaik       The Great Gatsby
Sashi       Ambati      X-Men: God Loves
Aadhya      Sree        V for Vendetta
Robin       Steve       Mike Tyson : Undisputed Truth

*/
```

CHAPTER 4. CONCLUSION AND FUTUREWORK

4.1 Conclusion

In conclusion, a library management system is a complex system that involves multiple tables and relationships to efficiently manage and organize library resources. The system typically includes tables such as Book, Borrower, Transaction, Category, Author, Staff, FinePenalty, Publisher, BookCopies, Reservations, Languages, and BookReview.

Each table serves a specific purpose and holds relevant information about books, borrowers, transactions, categories, authors, staff members, fines and penalties, publishers, book copies, reservations, languages, and book reviews.

In conclusion, the Library Management System (LMS) database serves as an indispensable tool in the efficient management of libraries, fostering seamless operations and enhancing user experiences. The LMS database effectively organizes and categorizes vast amounts of information, enabling librarians to easily track and manage various resources, such as books, periodicals, multimedia materials, and user records.

By integrating features like search functionalities, borrowing and returning processes, reservation systems, and user profiles, the LMS database streamlines library operations, reduces administrative burdens, and improves accessibility for library patrons. It provides users with the ability to locate and request materials efficiently, while librarians can monitor and update the availability and circulation of resources in real-time.

Furthermore, the LMS database facilitates data-driven decision-making by generating comprehensive reports and analytics. Librarians can leverage this information to assess collection development, understand usage patterns, and identify areas for improvement. By leveraging these insights, libraries can optimize their resources, enhance their collections, and tailor their services to meet the evolving needs of their users.

Moreover, the LMS database also plays a pivotal role in safeguarding the integrity and security of library data. It ensures the privacy of user information, manages authentication and access controls, and protects against data loss or unauthorized modifications. By implementing robust data protection measures, libraries can instill confidence in their users, establishing a foundation of trust and reliability.
In a digital era where information is abundant and diverse, the Library Management System database serves as a cornerstone for libraries to adapt and thrive. It empowers librarians to efficiently manage their resources, deliver personalized experiences to their users, and foster a vibrant and engaging library community. With continuous advancements in technology and the evolving needs of library users, the LMS database will continue to evolve, supporting libraries in their mission to provide knowledge, education, and enrichment to all.

Efficient Resource Management: A library management system helps in efficiently managing library resources such as books, journals, and multimedia materials. It allows librarians to organize and track the availability, circulation, and location of these resources, making it easier for users to find and access the materials they need.

Streamlined Borrowing and Returns: The system automates the process of book borrowing and returns, reducing manual paperwork and streamlining the checkout and check-in procedures. This improves the overall efficiency of library operations and saves time for both library staff and users.

Enhanced User Experience: Library management systems offer features that enhance the user experience. Users can search for books, check their availability, place reservations, and renew borrowed items online. This convenience improves user satisfaction and encourages more frequent library usage.

Accurate Fines and Penalties Calculation: The system automates the calculation of fines and penalties for overdue books, ensuring accuracy and fairness. It tracks the due dates, calculates fines based on predefined rules, and updates the borrower's account accordingly. This reduces disputes and ensures a transparent and efficient fine management process.

Data Analysis and Reporting: Library management systems provide valuable insights through data analysis and reporting capabilities. Librarians can generate reports on various aspects such as book circulation, popular books, user preferences, and overdue items. These insights help in making informed decisions about resource allocation, collection development, and service improvements**.**

4.2 Future Work

Integration with Emerging Technologies: As technology advances, libraries can explore integrating emerging technologies into their management systems. This could include incorporating features like artificial intelligence (AI) for recommendation systems, virtual reality (VR) for immersive learning experiences, or blockchain for secure transactions and authentication.

Mobile Application Development: Developing a mobile application for the library management system would provide users with even more convenience and accessibility. Users can search for books, make reservations, receive notifications, and manage their accounts directly from their mobile devices.

Enhanced accessibility features: Accessibility is a crucial aspect of library services. In the future, the LMS database can be enhanced to support features like text-to-speech functionality for visually impaired users, closed captions for videos, and compatibility with assistive technologies. By prioritizing accessibility, libraries can ensure that all individuals, regardless of their abilities, can fully access and benefit from the resources and services provided.

Integration with digital content platforms: With the increasing availability of digital resources, the LMS database can be expanded to seamlessly integrate with various digital content platforms. This integration would enable users to access and manage e-books, audiobooks, online journals, and other digital materials directly through the library system. Such integration would provide a unified experience for users, allowing them to discover, borrow, and return both physical and digital resources through a single interface.

Data analytics and predictive analysis: Expanding the capabilities of the LMS database to include advanced data analytics and predictive analysis can provide valuable insights for librarians. By analyzing borrowing patterns, user behavior, and resource usage trends, libraries can make data-driven decisions on collection development, resource allocation, and programming. Predictive analysis can help identify potential demand for certain materials, optimize inventory management, and enhance overall resource utilization.

Collaborative and social features: Libraries are not just repositories of books; they are also community spaces that foster interaction and collaboration. The LMS database can incorporate social features to facilitate community engagement, such as discussion forums, book clubs, and user-generated content. By promoting user interaction and knowledge sharing, libraries can create a sense of community and encourage collaboration among library users.

Seamless integration with external systems: To further enhance the efficiency of library operations, the LMS database can be designed to seamlessly integrate with external systems. This could include integrating with financial systems for fine management, student information systems for user authentication and data synchronization, or interlibrary loan networks to facilitate resource sharing among libraries. Such integrations would minimize manual processes, reduce data duplication, and streamline workflows for librarians and users alike.

By continuously improving and expanding the capabilities of the LMS database, libraries can adapt to evolving technology trends, user expectations, and changing needs. The future work outlined above presents opportunities for libraries to leverage technology effectively, enhance user experiences, and remain at the forefront of providing accessible, valuable, and engaging library services to their communities.