Library Management System

By Neha Reddy Yenugu

# INTRODUCTION

To make knowledge more accessible for learning, education and training, libraries have a crucial role to play. A successful library is one that is well-managed. Employees of the library need to keep track of resources, newly issued books, books that are borrowed and returned along with the due dates. When the data is in bulk, it is librarians’ responsibility to maintain the thousands of volumes that are maintained in library. The librarian must put a lot of effort into issue, receive, reserve, and maintain the book. And manually doing this takes a lot of time and energy. Here, digitalizing comes into picture, where we can maintain all the records.

A library management system is software that helps the library staff to keep track of all the resources, such as books, articles and digital content, number of books in the library, how many are issued, how many are returned or renewed, how many have late fines assessed, etc. The system also provides reports on usage of resources which will allow the staff members to plan about whether to order new resources and in what quantity they need to order. Overall, a library management system is an important tool for librarians to efficiently manage their library and provide access to information for their members. Creating a library management system software is the best approach to systematically maintain, organize and handle uncountable books.

# SCOPE OF THE PROJECT

The purpose of a library management system is to take charge of the library's resources, maintain records of all borrowed resources, and organize these resources. The goal of this project is to create a database schema for running a public library.

Uses:

* We can maintain detailed information on the library's book collection, staff, and their due dates utilizing this system.
* Managing responsibilities like borrowing and keeping track of library items.
* Dealing with late fees and overdue books to make sure that library materials are returned on schedule.

The ultimate purpose of this project is to provide library staff with a time-saving method of managing library resources and services.

# REQUIREMENTS

## Software Requirements:

The software requirements for completing this project are MySQL for coding and draw.io for Entity Relationship Diagram.

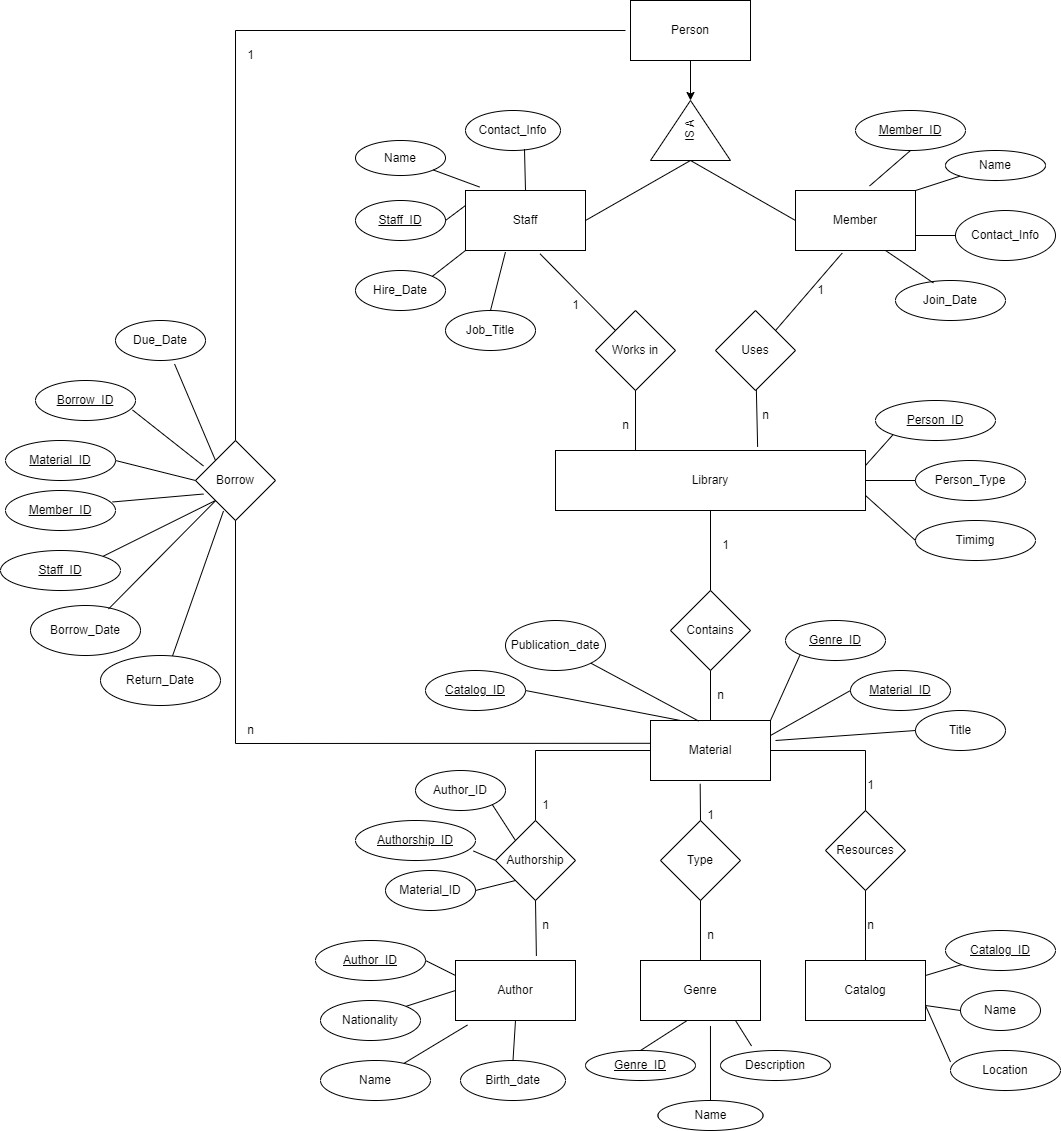
## Hardware Requirements:

For completing the project along with software requirements, we do need a system with internet connect.

## Project Requirements:

* **Materials Management:** The system should store and maintain information about all library materials, such as books, magazines, e-books, and audiobooks, including their titles, authors, publication dates, and genres.
* **Membership Management**: The system should store and manage information about library members, including their names, contact information, membership numbers, and borrowing history.
* **Borrowing:** The system should facilitate the borrowing process, allowing members to check out items, and provide library staff with the necessary information to manage the circulation of library materials. Once material is checked out, a librarian should record it borrow date, anticipated due date. And once the material is returned, its return date should be updated.
* **Reporting and Analytics:** The system should generate reports on library usage, popular materials, and other relevant statistics, enabling the library staff to make data-driven decisions about resource acquisition and management.

# ENTITY RELATIONSHIP DIAGRAM



**ENTITIES AND RELATIONSHIPS**

For the Library Management System, we have the following entities.

1. Materials
2. Staff
3. Author
4. Library
5. Catalog
6. Genre
7. Person
8. Member

and consists of the following relationships.

1. WorksIn
2. Uses
3. Borrow
4. Contains
5. Authorship
6. Type
7. Resources

# Entities

### Material –

It consists of resources such as books, magazines, e-books and audio books which has different attributes such as Title, PublicationDate etc.

### Staff –

The employees who work in the library come under this table. They maintain the library and ensure consistency of the incoming books and borrowing materials. The primary key is Staff\_ID.

### Author –

All the resources that are available in the material are written or published by the authors with the attributes Author\_ID, Author\_Name, Publication\_Date etc.

### Library –

It is a public library and contains a lot of materials. It is managed by the staff and the members of the library can rent the materials available in the library.

### Catalog –

This entity consists of the data about the materials location inside the library. This makes it easier for the members to easily locate a book they need.

### Genre –

This entity contains the type of the genre of the book in the library. A list of available materials can be sorted based on the genre as well.

### Person –

This entity defines the ISA relationship with Member and the Staff.

### Member –

It consists of the members who are the customers of the library.

# Relationships

### WorksIn –

This relationship is between the Staff and the Library. The relationship is one to many. This means that the n number of Staff can work in the library.

### Uses –

It determines the relationship between the Member and the Library. The relationship is one to many.

### Borrow –

It defines the relationship between the Person and the Materials. The person may be a staff member or the member of the library. The relationship is one to many.

### Contains –

This relationship is between the Library and the Materials. The relationship is one to many. Which means the library can have n number of materials.

### Authorship –

It determines the relation between materials and author. The relationship is one to many. For each material there can be any number of authors.

### Type –

It is a relationship between the materials and the genre. This is also a one-to-many relationship.

### Resources –

It defines the relation between the materials and the catalog. This is also a one-to-many relationship.

# DATABASE IMPLEMENTATION

The MySQL software is downloaded from the official website and the MySQL Workbench is used for writing the code and database connection.

# DDL

## Creation of Tables:

### Table 1:

Creation of Material Table

CREATE TABLE material (

`Material\_ID` INT NOT NULL,

`Title` VARCHAR(100) NULL,

`Publication\_Date` DATE NULL,

`Catalog\_ID` INT NULL,

`Genre\_ID` INT NULL, PRIMARY KEY (`Material\_ID`),

UNIQUE INDEX `Material\_ID\_UNIQUE` (`Material\_ID` ASC) VISIBLE, INDEX `Catalog\_ID\_idx` (`Catalog\_ID` ASC) VISIBLE,

INDEX `Genre\_ID\_idx` (`Genre\_ID` ASC) VISIBLE, CONSTRAINT `Catalog\_ID`

FOREIGN KEY (`Catalog\_ID`) REFERENCES `catalog` (`Catalog\_ID`) ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `Genre\_ID` FOREIGN KEY (`Genre\_ID`)

REFERENCES `genre` (`Genre\_ID`) ON DELETE CASCADE

ON UPDATE CASCADE);

### Table 2:

Creation of Catalog table

CREATE TABLE catalog (

`Catalog\_ID` INT NOT NULL,

`Name` VARCHAR(45) NULL,

`Location` VARCHAR(45) NULL, PRIMARY KEY (`Catalog\_ID`),

UNIQUE INDEX `Catalog\_ID\_UNIQUE` (`Catalog\_ID` ASC) VISIBLE);

### Table 3:

Creation of Genre table

CREATE TABLE genre (

`Genre\_ID` INT NOT NULL,

`Name` VARCHAR(45) NULL,

`Description` LONGTEXT NULL, PRIMARY KEY (`Genre\_ID`),

UNIQUE INDEX `Genre\_ID\_UNIQUE` (`Genre\_ID` ASC) VISIBLE);

### Table 4:

Creation of Borrow Table

CREATE TABLE borrow (

`Borrow\_ID` INT NOT NULL,

`Material\_ID` INT NULL,

`Member\_ID` INT NULL,

`Staff\_ID` INT NULL,

`Borrow\_Date` DATE NULL,

`Due\_Date` DATE NULL,

`Return\_Date` DATE NULL, PRIMARY KEY (`Borrow\_ID`),

UNIQUE INDEX `Borrow\_ID\_UNIQUE` (`Borrow\_ID` ASC) VISIBLE, INDEX `Material\_id\_idx` (`Material\_ID` ASC) VISIBLE,

INDEX `Member\_ID\_idx` (`Member\_ID` ASC) VISIBLE, INDEX `Staff\_ID\_idx` (`Staff\_ID` ASC) VISIBLE, CONSTRAINT `MaterialId`

FOREIGN KEY (`Material\_ID`) REFERENCES `material` (`Material\_ID`) ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `Member\_ID` FOREIGN KEY (`Member\_ID`)

REFERENCES `member` (`Member\_ID`) ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `Staff\_ID` FOREIGN KEY (`Staff\_ID`) REFERENCES `staff` (`Staff\_ID`) ON DELETE CASCADE

ON UPDATE CASCADE);

### Table 5:

Creation of Author Table

CREATE TABLE author (

`Author\_ID` INT NOT NULL,

`Name` VARCHAR(45) NULL,

`Birth\_Date` DATE NULL,

`Nationality` VARCHAR(45) NULL, PRIMARY KEY (`Author\_ID`),

UNIQUE INDEX `Author\_ID\_UNIQUE` (`Author\_ID` ASC) VISIBLE);

### Table 6:

Creation of Authorship Table

CREATE TABLE authorship (

`Authorship\_ID` INT NOT NULL,

`Author\_ID` INT NULL,

`Material\_ID` INT NULL, PRIMARY KEY (`Authorship\_ID`),

UNIQUE INDEX `Authorship\_ID\_UNIQUE` (`Authorship\_ID` ASC) VISIBLE, INDEX `Author\_ID\_idx` (`Author\_ID` ASC) VISIBLE,

INDEX `Material\_ID\_idx` (`Material\_ID` ASC) VISIBLE, CONSTRAINT `Author\_ID`

FOREIGN KEY (`Author\_ID`) REFERENCES `author` (`Author\_ID`) ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `Material\_ID` FOREIGN KEY (`Material\_ID`)

REFERENCES `material` (`Material\_ID`) ON DELETE CASCADE

ON UPDATE CASCADE);

### Table 7:

Creation of Member Table

CREATE TABLE Member( Member\_ID INT PRIMARY KEY, Name VARCHAR(45) NULL,

Contact\_info VARCHAR(100) NULL, Join\_Date DATE NULL,

UNIQUE (Member\_ID)

);

### Table 8:

Creation of Staff Table

CREATE TABLE staff (

`Staff\_ID` INT NOT NULL,

`Name` VARCHAR(45) NULL,

`Contact\_Info` VARCHAR(100) NULL,

`Job\_Title` VARCHAR(45) NULL,

`Hire\_Date` DATE NULL, PRIMARY KEY (`Staff\_ID`),

UNIQUE INDEX `Staff\_ID\_UNIQUE` (`Staff\_ID` ASC) VISIBLE);

# POPULATE OF DATABASE

### Inserting values into Material table:

INSERT INTO Member VALUES(1,"Alice Johnson","[alice.johnson@email.com](mailto:alice.johnson@email.com)", "2018-01-10"), (2,"Bob Smith","[bob.smith@email.com](mailto:bob.smith@email.com)","2018-03-15"),

(3,"Carol Brown","[carol.brown@email.com](mailto:carol.brown@email.com)","2018-06-20"), (4,"David Williams","[david.williams@email.com](mailto:david.williams@email.com)","2018-09-18"), (5,"Emily Miller","[emily.miller@email.com](mailto:emily.miller@email.com)","2019-02-12"), (6,"Frank Davis","[frank.davis@email.com](mailto:frank.davis@email.com)","2019-05-25"), (7,"Grace Wilson","[grace.wilson@email.com](mailto:grace.wilson@email.com)","2019-08-15"), (8,"Harry Garcia","[harry.garcia@email.com](mailto:harry.garcia@email.com)","2019-11-27"), (9,"Isla Thomas","[isla.thomas@email.com](mailto:isla.thomas@email.com)","2020-03-04"), (10,"Jack Martinez","[jack.martinez@email.com](mailto:jack.martinez@email.com)","2020-07-01"), (11,"Kate Anderson","[kate.anderson@email.com](mailto:kate.anderson@email.com)","2020-09-30"), (12,"Luke Jackson","[luke.jackson@email.com](mailto:luke.jackson@email.com)","2021-01-18"), (13,"Mia White","[mia.white@email.com](mailto:mia.white@email.com)","2021-04-27"), (14,"Noah Harris","[noah.harris@email.com](mailto:noah.harris@email.com)","2021-07-13"), (15,"Olivia Clark","[olivia.clark@email.com](mailto:olivia.clark@email.com)","2021-10-05"), (16,"Peter Lewis","[peter.lewis@email.com](mailto:peter.lewis@email.com)","2021-12-01"), (18,"Rachel Young","[rachel.young@email.com](mailto:rachel.young@email.com)","2022-06-17"), (17,"Quinn Hall","[quinn.hall@email.com](mailto:quinn.hall@email.com)","2022-02-28"),

(19,"Sam Walker","[sam.walker@email.com](mailto:sam.walker@email.com)","2022-09-25"), (20,"Tiffany Allen","[tiffany.allen@email.com](mailto:tiffany.allen@email.com)","2022-12-10");

### Inserting values into Catalog Table:

INSERT INTO catalog (`Catalog\_ID`, `Name`, `Location`) VALUES ('1', 'Books', 'A1.1'),

('2', 'Magazines', 'B2.1'),

('3', 'E-Books', 'C3.1'),

('4', 'Audiobooks', 'D4.1'),

('5', 'Journals', 'E5.1'),

('6', 'Newspaper', 'F6.1'),

('7', 'Maps', 'G7.1'),

('8', 'Novels', 'H8.1'),

('9', 'Sheet Music', 'I9.1'),

('10', 'Educational', 'J10.1');

### Inserting values into Genre Table:

INSERT INTO genre (`Genre\_ID`, `Name`, `Description`) VALUES

('1', 'General Fiction', 'Literary works with a focus on character and plot development exploring various themes and human experiences.'),

('2', 'Mystery & Thriller', ' Suspenseful stories centered around crime, investigation, or espionage with an emphasis on tension and excitement.'),

('3', 'Science Fiction & Fantasy', 'Imaginative works that explore alternate realities, futuristic concepts, and magical or supernatural elements.'),

('4', ' Horror & Suspense ', 'Stories designed to evoke fear, unease, or dread, often featuring supernatural or psychological elements.'),

('5', 'Dystopian & Apocalyptic', 'Depictions of societies in decline or collapse, often exploring themes of political and social oppression or environmental disaster.'),

('6', 'Classics', 'Enduring works of literature that have stood the test of time, often featuring rich language and complex themes.'),

('7', 'Historical Fiction', 'Fictional stories set in the past, often based on real historical events or figures, and exploring the customs and experiences of that time.'),

('8', 'Epic Poetry & Mythology', 'Ancient or traditional stories and poems, often featuring heroes, gods, and mythical creatures, and exploring cultural values and beliefs');

### Inserting values into Borrow table:

INSERT INTO borrow (`Borrow\_ID`, `Material\_ID`, `Member\_ID`, `Staff\_ID`, `Borrow\_Date`,

`Due\_Date`, `Return\_Date`) VALUES

('1', '1', '1', '1', '2018-09-12', '2018-10-03', '2018-09-30'),

('2', '2', '2', '1', '2018-10-15', '2018-11-05', '2018-10-29'),

('3', '3', '3', '1', '2018-12-20', '2019-01-10', '2019-01-08'),

('4', '4', '4', '1', '2019-03-11', '2019-04-01', '2019-03-27'),

('5', '5', '5', '1', '2019-04-20', '2019-05-11', '2019-05-05'),

('6', '6', '6', '1', '2019-07-05', '2019-07-26', '2019-07-21'),

('7', '7', '7', '1', '2019-09-10', '2019-10-01', '2019-09-25'),

('8', '8', '8', '1', '2019-11-08', '2019-11-29', '2019-11-20'),

('9', '9', '9', '1', '2020-01-15', '2020-02-05', '2020-02-03'),

('10', '10', '10', '1', '2020-03-12', '2020-04-02', '2020-03-28'),

('11', '1', '11', '2', '2020-05-14', '2020-06-04', '2020-05-28'),

('12', '2', '12', '2', '2020-07-21', '2020-08-11', '2020-08-02'),

('13', '3', '13', '2', '2020-09-25', '2020-10-16', '2020-10-15'),

('14', '4', '1', '2', '2020-11-08', '2020-11-29', '2020-11-24'),

('15', '5', '2', '2', '2021-01-03', '2021-01-24', '2021-01-19'),

('16', '6', '3', '2', '2021-02-18', '2021-03-11', '2021-03-12'),

('17', '17', '4', '2', '2021-04-27', '2021-05-18', '2021-05-20'),

('18', '18', '5', '2', '2021-06-13', '2021-07-04', '2021-06-28'),

('19', '19', '6', '2', '2021-08-15', '2021-09-05', '2021-09-03'),

('20', '20', '7', '2', '2021-10-21', '2021-11-11', '2021-11-05'),

('21', '21', '1', '3', '2021-11-29', '2021-12-20',null),

('22', '22', '2', '3', '2022-01-10', '2022-01-31', '2022-01-25'),

('23', '23', '3', '3', '2022-02-07', '2022-02-28', '2022-02-23'),

('24', '24', '4', '3', '2022-03-11', '2022-04-01', '2022-03-28'),

('25', '25', '5', '3', '2022-04-28', '2022-05-19', '2022-05-18'),

('26', '26', '6', '3', '2022-06-22', '2022-07-13', '2022-07-08'),

('27', '27', '7', '3', '2022-08-04', '2022-08-25', '2022-08-23'),

('28', '28', '8', '3', '2022-09-13', '2022-10-04', '2022-09-28'),

('29', '29', '9', '3', '2022-10-16', '2022-11-06', '2022-11-05'),

('30', '30', '8', '3', '2022-11-21', '2022-12-12', '2022-12-05'),

('31', '1', '9', '4', '2022-12-28', '2023-01-18', null),

('32', '2', '1', '4', '2023-01-23', '2023-02-13', null),

('33', '3', '10', '4', '2023-02-02', '2023-02-23', '2023-02-17'),

('34', '4', '11', '4', '2023-03-01', '2023-03-22', null),

('35', '5', '12', '4', '2023-03-10', '2023-03-31', null),

('36', '6', '13', '4', '2023-03-15', '2023-04-05', null),

('37', '7', '17', '4', '2023-03-25', '2023-04-15', null),

('38', '8', '8', '4', '2023-03-30', '2023-04-20', null),

('39', '9', '9', '4', '2023-03-26', '2023-04-16', null),

('40', '10', '20', '4', '2023-03-28', '2023-04-18', null);

### Inserting values into Author Table:

INSERT INTO author (`Author\_ID`, `Name`, `Birth\_Date`,

`Nationality`) VALUES

('1', ' Jane Austen', '1775-12-16', 'British'),

('2', 'Ernest Hemingway', '1899-07-21', 'American'),

('3', 'George Orwell', '1903-06-25', 'British'),

('4', ' Scott Fitzgerald', '1896-09-24', 'American'),

('5', ' J.K. Rowling', '1965-07-31', 'British'),

('6', 'Mark Twain', '1835-11-30', 'American'),

('7', 'Leo Tolstoy', '1828-09-09', 'Russian'),

('8', 'Virginia Woolf', '1882-01-25', 'British'),

('9', ' Gabriel Márquez', '1927-03-06', 'Colombian'),

('10', 'Charles Dickens', '1812-02-07', 'British'),

('11', ' Harper Lee', '1926-04-28', 'American'),

('12', 'Oscar Wilde', '1854-10-16', 'Irish'),

('13', 'William Shakespeare', '1564-04-26', 'British'),

('14', ' Franz Kafka', '1883-07-03', 'Czech'),

('15', 'James Joyce', '1882-02-02', 'Irish'),

('16', 'J.R.R. Tolkien', '1892-01-03', 'British'),

('17', ' Emily Brontë', '1818-07-30', 'British'),

('18', 'Toni Morrison', '1931-02-18', 'American'),

('19', ' Fyodor Dostoevsky', '1821-11-11', 'Russian'),

('20', 'Lucas Piki', '1847-10-16', 'British');

### Inserting into Authorship Table:

INSERT INTO authorship (`Authorship\_ID`, `Author\_ID`, `Material\_ID`) VALUES ('1', '1', '1'),

('2', '2', '2'),

('3', '3', '3'),

('4', '4', '4'),

('5', '5', '5'),

('6', '6', '6'),

('7', '7', '7'),

('8', '8', '8'),

('9', '9', '9'),

('10', '10', '10'),

('11', '11', '11'),

('12', '12', '12'),

('13', '13', '13'),

('14', '14', '14'),

('15', '15', '15'),

('16', '16', '16'),

('17', '17', '17'),

('18', '18', '18'),

('19', '19', '19'),

('20', '20', '20'),

('21', '1', '21'),

('22', '2', '22'),

('23', '3', '23'),

('24', '5', '24'),

('25', '5', '25'),

('26', '6', '26'),

('27', '7', '27'),

('28', '8', '28'),

('29', '19', '28'),

('30', '9', '29'),

('31', '10', '30'),

('32', '8', '30'),

('33', '2', '29');

### Inserting into Member Table:

INSERT INTO Member VALUES(1,"Alice Johnson","[alice.johnson@email.com](mailto:alice.johnson@email.com)", "2018-01-10")

,(2,"Bob Smith","[bob.smith@email.com](mailto:bob.smith@email.com)","2018-03-15")

,(3,"Carol Brown","[carol.brown@email.com](mailto:carol.brown@email.com)","2018-06-20")

,(4,"David Williams","[david.williams@email.com](mailto:david.williams@email.com)","2018-09-18")

,(5,"Emily Miller","[emily.miller@email.com](mailto:emily.miller@email.com)","2019-02-12")

,(6,"Frank Davis","[frank.davis@email.com](mailto:frank.davis@email.com)","2019-05-25")

,(7,"Grace Wilson","[grace.wilson@email.com](mailto:grace.wilson@email.com)","2019-08-15")

,(8,"Harry Garcia","[harry.garcia@email.com](mailto:harry.garcia@email.com)","2019-11-27")

,(9,"Isla Thomas","[isla.thomas@email.com](mailto:isla.thomas@email.com)","2020-03-04")

,(10,"Jack Martinez","[jack.martinez@email.com](mailto:jack.martinez@email.com)","2020-07-01")

,(11,"Kate Anderson","[kate.anderson@email.com](mailto:kate.anderson@email.com)","2020-09-30")

,(12,"Luke Jackson","[luke.jackson@email.com](mailto:luke.jackson@email.com)","2021-01-18")

,(13,"Mia White","[mia.white@email.com](mailto:mia.white@email.com)","2021-04-27")

,(14,"Noah Harris","[noah.harris@email.com](mailto:noah.harris@email.com)","2021-07-13")

,(15,"Olivia Clark","[olivia.clark@email.com](mailto:olivia.clark@email.com)","2021-10-05")

,(16,"Peter Lewis","[peter.lewis@email.com](mailto:peter.lewis@email.com)","2021-12-01")

,(18,"Rachel Young","[rachel.young@email.com](mailto:rachel.young@email.com)","2022-06-17")

,(17,"Quinn Hall","[quinn.hall@email.com](mailto:quinn.hall@email.com)","2022-02-28")

,(19,"Sam Walker","[sam.walker@email.com](mailto:sam.walker@email.com)","2022-09-25")

,(20,"Tiffany Allen","[tiffany.allen@email.com](mailto:tiffany.allen@email.com)","2022-12-10");

# QUERIES

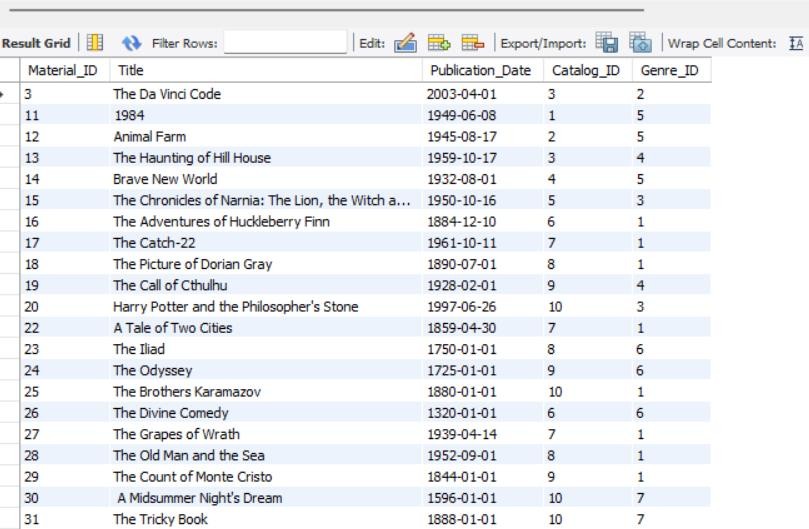
### Query 1:

Which materials are currently available in the library?

### Solution:

SELECT \* FROM Material WHERE Material\_ID NOT IN (SELECT Material\_ID FROM Borrow where Return\_Date IS NULL);

### Output:



**Query 2:**

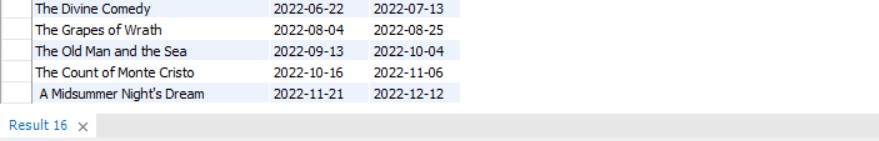
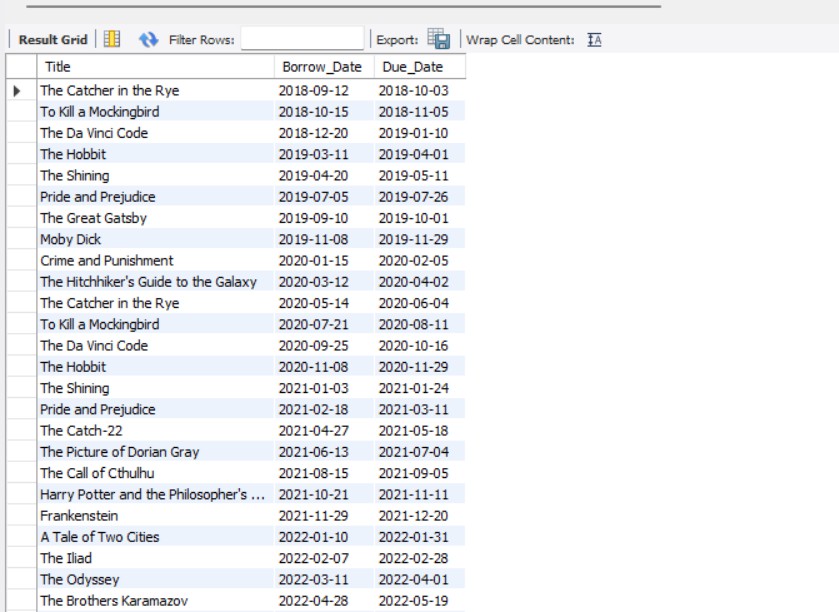
Which materials are currently overdue?

Suppose today is 04/01/2023, and show the borrow date and due date of each material

### Solution:

SELECT m.Title,b.Borrow\_Date,b.Due\_Date FROM Borrow b, Material m WHERE b.Due\_Date <= '20230104' AND b.Material\_ID = m.Material\_ID;

### Output:



**Query 3:**

What are the top 10 most borrowed materials in the library? Show the title of each material and order them based on their available counts

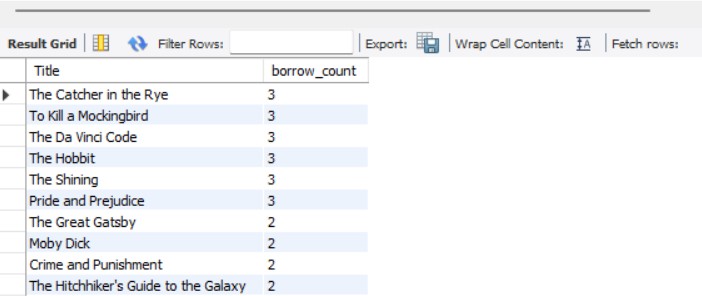
### Solution:

SELECT m.Title, COUNT(\*) AS BorrowCount

FROM Material m JOIN Borrow b ON m.Material\_ID=b.Material\_ID GROUP BY m.Material\_ID

ORDER BY BorrowCount DESC LIMIT 10;

### Output:



**Query 4:**

How many books has the author Lucas Piki written?

### Solution:

SELECT a.Name, COUNT(\*) AS BooksWritten FROM Material m JOIN Authorship aShip

ON aShip.Material\_ID = m.Material\_ID JOIN Author a

ON a.Author\_ID=aShip.Author\_ID WHERE a.Name='Lucas Piki' GROUP BY a.Author\_ID; **Output:**



### Query 5:

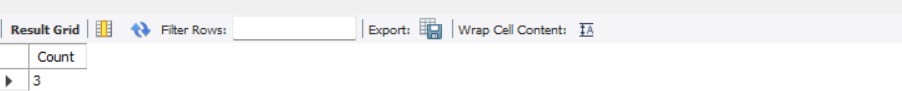
How many books were written by two or more authors?

### Solution:

SELECT COUNT(\*) as COUNT from (SELECT m.Title, count(\*) as noOfAuthors from Material m JOIN Authorship aShip

ON aShip.Material\_ID = m.Material\_ID JOIN Author a

ON a.Author\_ID=aShip.Author\_ID GROUP BY aShip.Material\_ID HAVING noOfAuthors>1) ans; **Output:**



### Query 6:

What are the most popular genres in the library?

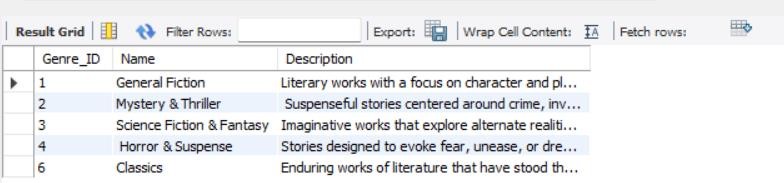
### Solution:

SELECT g.Genre\_ID,g.Name,g.Description FROM

(SELECT m.Genre\_ID FROM Borrow b,Material m WHERE m.Material\_ID=b.Material\_ID) t,Genre g WHERE t.Genre\_ID=g.Genre\_ID

GROUP BY g.Genre\_ID ORDER BY g.Genre\_ID ASC LIMIT 5;

### Output:



**Query 7:**

How many materials have been borrowed from 09/2020-10/2020?

### Solution:

SELECT COUNT(\*) AS Count FROM(

SELECT m.Title, b.Borrow\_Date FROM Material m

JOIN Borrow b

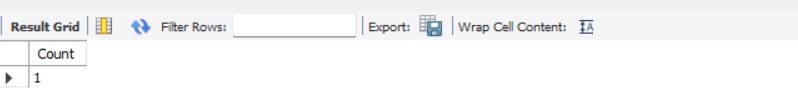
ON m.Material\_ID=b.Material\_ID

WHERE date\_format(b.Borrow\_Date,'%Y-%m')

BETWEEN DATE\_FORMAT(STR\_TO\_DATE(CONCAT('01/', '09/2020'), '%d/%m/%Y'), '%Y-%m') AND DATE\_FORMAT(STR\_TO\_DATE(CONCAT('01/', '10/2020'), '%d/%m/%Y'), '%Y-%m')

) answer;

### Output:



**Query 8:**

How do you update the “Harry Potter and the Philosopher's Stone” when it is returned on 04/01/2023?

### Solution:

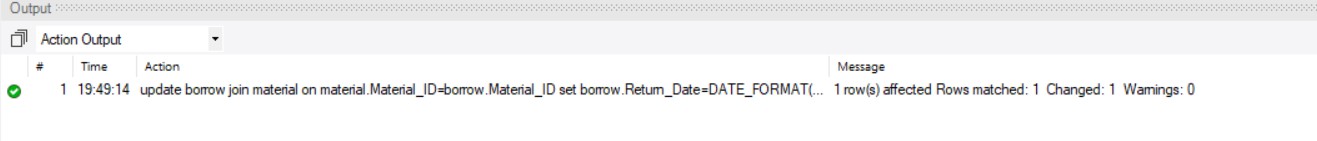
UPDATE Borrow b JOIN Material m

ON m.Material\_ID = b.Material\_ID

SET b.Return\_Date=DATE\_FORMAT(STR\_TO\_DATE('04/01/2023', '%d/%m/%Y'), '%Y-%m-%d')

WHERE m.Title='Harry Potter and the Philosopher\'s Stone';

### Output:



**Query 9:**

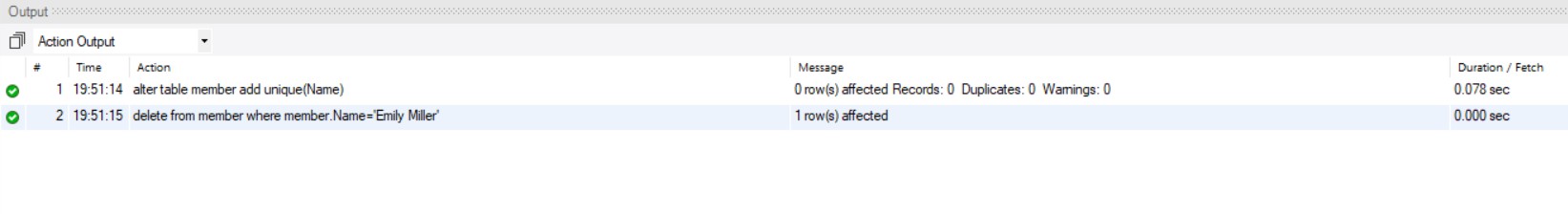
How do you delete the member Emily Miller and all her related records from the database?

### Solution:

ALTER TABLE Member ADD UNIQUE(Name);

DELETE FROM Member m WHERE m.Name='Emily Miller';

### Output:



**Query 10:**

How do you add the following material to the database? Title: New book

Date: 2020-08-01

Catalog: E-Books

Genre: Mystery & Thriller Author: Lucas Pipi **Solution:**

SELECT Catalog\_ID FROM Catalog c WHERE c.Name='E-Books' INTO @Catalog\_ID;

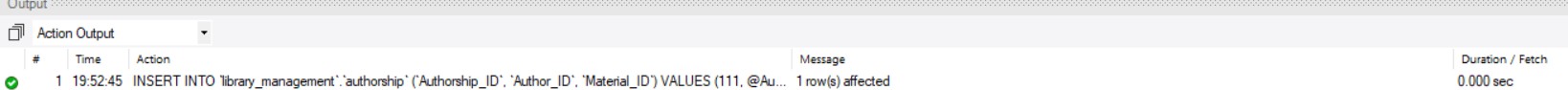
SELECT Genre\_ID FROM Genre g WHERE g.Name='Mystery & Thriller' INTO @Genre\_ID;

INSERT INTO `material` (`Material\_ID`, `Title`, `Publication\_Date`, `Catalog\_ID`, `Genre\_ID`) VALUES (111,'New book', '2020-08-01', @Catalog\_ID , @Genre\_ID);

SELECT m.Material\_ID FROM Material m WHERE m.Title='New book' INTO @Material\_ID; SELECT a.Author\_ID FROM Author a WHERE a.Name='Lucas Piki' INTO @Author\_ID;

INSERT INTO `authorship` (`Authorship\_ID`,`Author\_ID`,`Material\_ID`) VALUES (111, @Author\_ID, @Material\_ID);

**Output:**



# FUTURE WORK

### Alert staff about overdue materials daily.

A notification can be sent to the staff when a new borrow due date has been passed and the due date past books can be checked on daily basis and update these materials as due books. Once the staff logins to their system they observe the materials that are past the due dates and the books that are due on a particular date.

Query that should be run for every 24hrs.

SELECT m.Maetrial\_ID,m.Title , mem.Member\_ID, mem.Name FROM Material m, Borrow b, Member mem

WHERE b.Return\_Date<CURRENT\_DATE and m.Material\_ID=b.Material\_ID and mem.Member\_ID=b.Member\_ID;

### Automatically deactivate the membership based on the member’s overdue occurrence (>= three times). And reactivate the membership once the member pays the overdue fee.

For deactivation we can add a new column to the member table. The column can be Status and the values would be Active or Inactive. The books borrowing can be done only when the status is Active, and the borrowing can be stopped when the status is inactive. For toggle of the active and inactive status we need to count the number of overdue books. For this we will create a new table named Overdue table consisting of Material\_ID, Member\_ID, Due\_Date. Once a book is past the Due\_Date this material id , member id and due date will be inserted into the Overdue table. For toggling the status in Member table, we will add a trigger to the Overdue table for insertion and deletion. Every time we insert a new record, we will check for the number of over due items for that member and once the count is greater than or equals to 3 the status of the Member will be changed to Inactive and once we delete a record we again check for the number of over due items and if the count is less than 3 the member status would be changed to the active status.

# INSTRUCTIONS TO RUN THE CODE

1. First create a database and use the database.
2. Then create all the tables using the queries.
3. Then insert all the values into the tables.
4. After insertion execute the queries for the given questions.

# CONSLUSION

In conclusion, I have developed a Library Management System which will be helpful for the management of the library materials that are stored and borrowed by the members. In this by using the MySQL I have combined the DDL (Data Definition Language) and DML (Data Manipulation Language) such as creation, insertion and updating the values. I have used the primary keys and foreign keys which will be helpful in maintaining the consistency of the database. I also ensured of no data redundancy.