A DATABASE MANAGEMENT SYSTEM FOR THE LIBRARY

Abhishiek Kurra

CS504

PRINCIPLES OF DATA MANAGEMENT & MINING

Binqian Yin, Ph.D.

Introduction:

Managing a library manually is a herculean task. The management must keep track of every book, magazine, paper, member, and library staff. This is quite difficult to manage manually. The Database Management System (DBMS) is a handy tool that can potentially make keeping track of all the data easy. However, for that, we must make sure that the DBMS is structured and maintained well. A well-structured DBMS can accommodate constant changes in the data because of management of materials, membership, changes made to database due to borrowing of the materials by the members. A good DBMS must also be able to provide an analysis when required. This ensures availability and optimal usage of the library's resources.

Entities:

Below are the few entities that are present in the current library use case:

1. Material: These are the individual entries of books, magazines, e-books etc.
2. Catalog: This is the record of availability and location of a material
3. Genre: Category/Genre of the library material
4. Author: Records of the authors of the library material
5. Member: Details of members who can borrow/book the materials
6. Staff: Details of staff members who manage the library

Relationships:

The following are the relationships for the given library use case:

1. Borrow: A member **borrows material** under the supervision of a **staff** member.
2. Authorship: An **author** has **authorship** over **material**.
3. Within: A **material** is within a **Catalog.**
4. Is of: A **material** is of a **genre**.

Tables:

Given below are the tables along with their attributes:

1. Material:

● Material_ID: A unique identifier for each material. (Primary)

● Title: The title of the material.

● Publication_Date: The date of publication of the material.

● Catalog_ID: A reference to the catalog entry for the material. (Foreign)

● Genre_ID: A reference to the genre of the material. (Foreign)

2. Catalog:

- Catalog_ID: A unique identifier for each catalog entry. (Primary)
- Name: The name of the catalog.
- Location: The location of the material within the library.

3. Genre

- Genre_ID: A unique identifier for each genre. (Primary)
- Name: The name of the genre.
- Description: The brief introduction of the genre.

4. Borrow:

- Borrow_ID: A unique identifier for each borrowing transaction. (Primary)
- Material_ID: A reference to the borrowed material. (Foreign)
- Member_ID: A reference to the member who borrowed the material. (Foreign)
- Staff_ID: A reference to the staff who processed the transaction. (Foreign)
- Borrow_Date: The date the material was borrowed.
- Due_Date: The date the material is due.
- Return_Date: The date the material is returned.

5. Author:

- Author_ID: A unique identifier for each author. (Primary)
- Name: The name of the author.
- Birth_Date: The birth date of the author.
- Nationality: The nationality of the author.

6. Authorship:

- Authorship_ID: A unique identifier for each authorship record. (Primary)
- Author_ID: A reference to the author. (Foreign)
- Material_ID: A reference to the material authored. (Foreign)

7. Member:

- Member_ID: A unique identifier for each member. (Primary)
- Name: The name of the member.

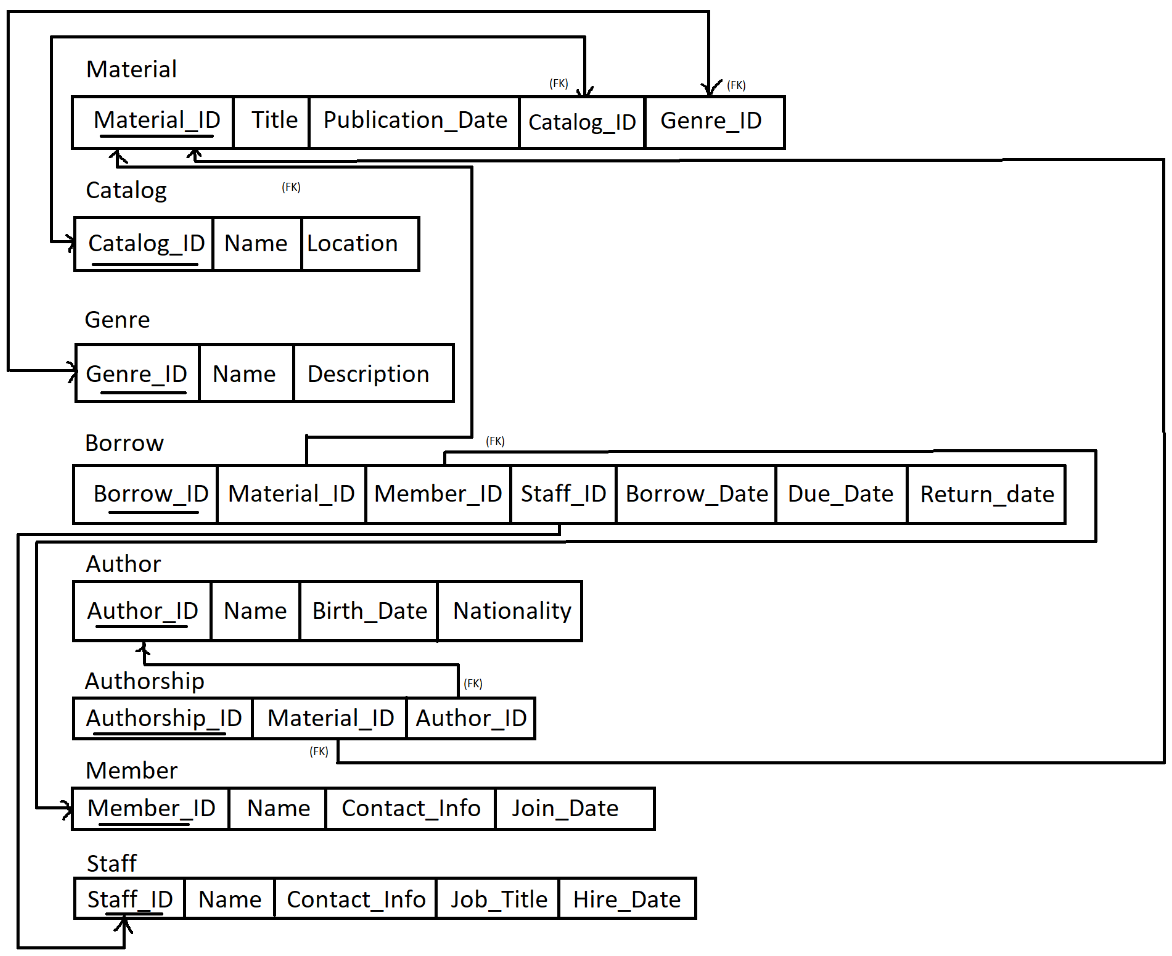
- Contact_Info: Email address (or phone number) of the member.
- Join_Date: The date the member joined the library.

8. Staff:

- Staff_ID: A unique identifier for each staff member. (Primary)
- Name: The name of the staff member.
- Contact_Info: Email address (or phone number) of the member.
- Job_Title: The job title of the staff member (e.g., librarian, assistant librarian).
- Hire_Date: The date the staff member was hired by the library.

Database Schema:

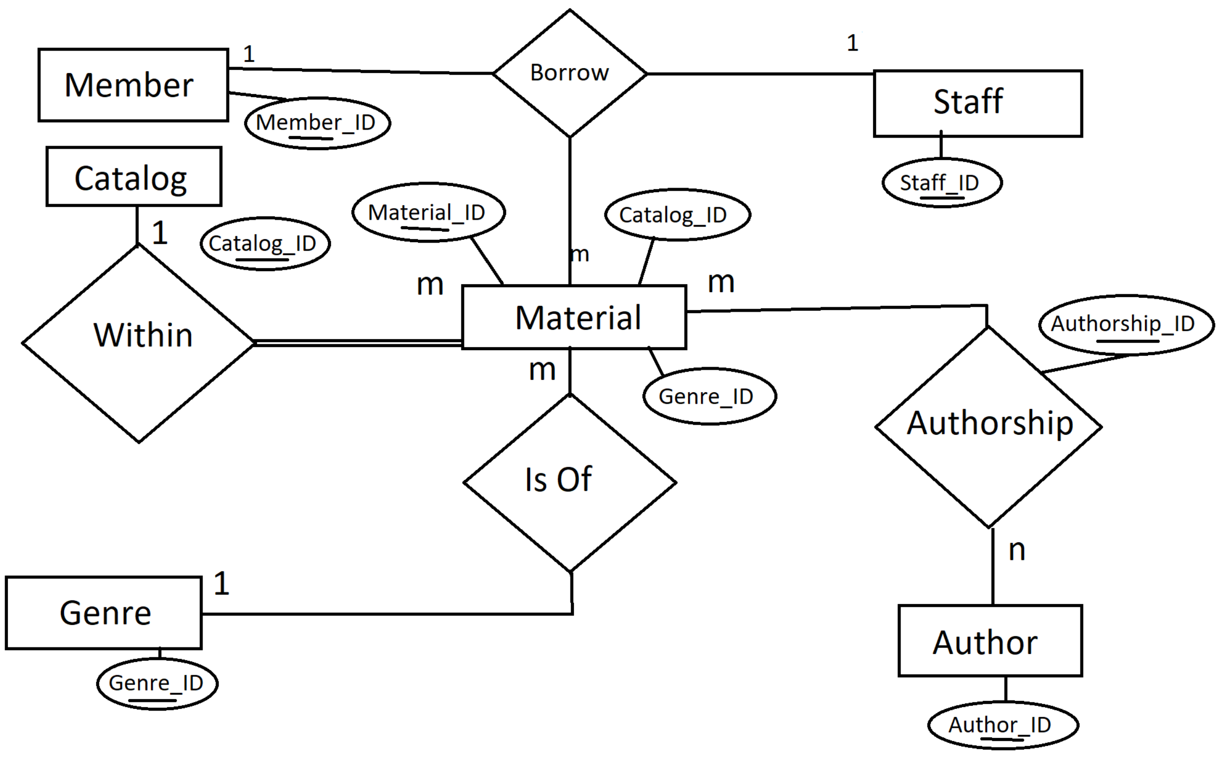
Given below is the schema for the given database.



Assumptions/Constraints upon the Databse:

1. All materials are within a single catalog, so all materials have complete participation.
2. All the materials have at least one genre, but a genre might not belong to any material.
3. Any number of materials can be borrowed by a member under the supervision of staff.
4. A material has at least one author. However, an author may or may not have authorship over material.

Entity-Relationship Diagram:



Chosen DBMS:

The Relational database system chosen for this is the MySQL database server. This is because the library data is local and does not require highly complicated set of management requirements. This rather simplifies the database by using SQL as Data Definition Language and Data Management Language.

Data Definition Language:

The following is the code used to in MySQL to define and input the initial data into the database.

```
CREATE DATABASE Library;

#Selecting a Database to use it for the rest of the DDL.

USE Library;


CREATE TABLE Staff (

  Staff_ID INT NOT NULL PRIMARY KEY,

  Name VARCHAR(255),

  Contact_Info VARCHAR(255),

  Job_Title VARCHAR(255),

  Hire_Date DATE

);


INSERT INTO Staff (Staff_ID, Name, Contact_Info, Job_Title, Hire_Date)

VALUES (1, 'Amy Green', 'amy.green@email.com', 'Librarian', '2017-06-01'),

       (2, 'Brian Taylor', 'brian.taylor@email.com', 'Library Assistant', '2018-11-15'),

       (3, 'Christine King', 'chris.king@email.com', 'Library Assistant', '2019-05-20'),

       (4, 'Daniel Wright', 'dan.wright@email.com', 'Library Technician', '2020-02-01');


CREATE TABLE Member (

  Member_ID INT NOT NULL PRIMARY KEY,

  Name VARCHAR(255),

  Contact_Info VARCHAR(255),

  Join_Date DATE

);


INSERT INTO Member (Member_ID, Name, Contact_Info, Join_Date)

VALUES

  (1, 'Alice Johnson', 'alice.johnson@email.com', '2018-01-10'),

  (2, 'Bob Smith', 'bob.smith@email.com', '2018-03-15'),

  (3, 'Carol Brown', 'carol.brown@email.com', '2018-06-20'),

  (4, 'David Williams', 'david.williams@email.com', '2018-09-18'),

  (5, 'Emily Miller', 'emily.miller@email.com', '2019-02-12'),

  (6, 'Frank Davis', 'frank.davis@email.com', '2019-05-25'),

  (7, 'Grace Wilson', 'grace.wilson@email.com', '2019-08-15'),

  (8, 'Harry Garcia', 'harry.garcia@email.com', '2019-11-27'),

  (9, 'Isla Thomas', 'isla.thomas@email.com', '2020-03-04'),
  (10, 'Jack Martinez', 'jack.martinez@email.com', '2020-07-01'),
  (11, 'Kate Anderson', 'kate.anderson@email.com', '2020-09-30'),
  (12, 'Luke Jackson', 'luke.jackson@email.com', '2021-01-18'),
  (13, 'Mia White', 'mia.white@email.com', '2021-04-27'),
  (14, 'Noah Harris', 'noah.harris@email.com', '2021-07-13'),
  (15, 'Olivia Clark', 'olivia.clark@email.com', '2021-10-05'),
  (16, 'Peter Lewis', 'peter.lewis@email.com', '2021-12-01'),
  (17, 'Quinn Hall', 'quinn.hall@email.com', '2022-02-28'),
  (18, 'Rachel Young', 'rachel.young@email.com', '2022-06-17'),
  (19, 'Sam Walker', 'sam.walker@email.com', '2022-09-25'),
  (20, 'Tiffany Allen', 'tiffany.allen@email.com', '2022-12-10');


CREATE TABLE Author (
  Author_ID INT NOT NULL PRIMARY KEY,
  Name VARCHAR(255),
  Birth_Date DATE,
  Nationality VARCHAR(255)
);


INSERT INTO Author (Author_ID, Name, Nationality, Birth_date)
VALUES
    (1, 'Jane Austen', 'British', '1775-12-16'),
    (2, 'Ernest Hemingway', 'American', '1899-07-21'),
    (3, 'George Orwell', 'British', '1903-06-25'),
    (4, 'Scott Fitzgerald', 'American', '1896-09-24'),
    (5, 'J.K. Rowling', 'British', '1965-07-31'),
    (6, 'Mark Twain', 'American', '1835-11-30'),
    (7, 'Leo Tolstoy', 'Russian', '1828-09-09'),
    (8, 'Virginia Woolf', 'British', '1882-01-25'),
    (9, 'Gabriel Márquez', 'Colombian', '1927-03-06'),
    (10, 'Charles Dickens', 'British', '1812-02-07'),
    (11, 'Harper Lee', 'American', '1926-04-28'),
    (12, 'Oscar Wilde', 'Irish', '1854-10-16'),
    (13, 'William Shakespeare', 'British', '1564-04-26'),
    (14, 'Franz Kafka', 'Czech', '1883-07-03'),
    (15, 'James Joyce', 'Irish', '1882-02-02'),
    (16, 'J.R.R. Tolkien', 'British', '1892-01-03'),
    (17, 'Emily Brontë', 'British', '1818-07-30'),

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

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

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


CREATE TABLE Catalog (

  Catalog_ID INT NOT NULL PRIMARY KEY,

  Name VARCHAR(255),

  Location VARCHAR(255)

);


INSERT INTO Catalog (Catalog_ID, Name, Location) VALUES

(1, 'Books', 'A1.1'),

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

(3, 'E-Bookscatalog', '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');


CREATE TABLE Genre (

  Genre_ID INT NOT NULL PRIMARY KEY,

  Name VARCHAR(255) ,

  Description VARCHAR(255)

);


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.');
```

```
CREATE TABLE Material (
  Material_ID INT PRIMARY KEY,
  Title VARCHAR(255),
  Publication_Date DATE ,
  Catalog_ID INT ,
  Genre_ID INT,
  FOREIGN KEY (Catalog_ID) REFERENCES Catalog(Catalog_ID) ON DELETE CASCADE,
  FOREIGN KEY (Genre_ID) REFERENCES Genre(Genre_ID) ON DELETE CASCADE
);

INSERT INTO Material (Material_ID, Title, Publication_Date, Catalog_ID, Genre_ID) VALUES
(1, 'The Catcher in the Rye', '1951-07-16', 1, 1),
(2, 'To Kill a Mockingbird', '1960-07-11', 2, 1),
(3, 'The Da Vinci Code', '2003-04-01', 3, 2),
(4, 'The Hobbit', '1937-09-21', 4, 3),
(5, 'The Shining', '1977-01-28', 5, 4),
(6, 'Pride and Prejudice', '1813-01-28', 1, 1),
(7, 'The Great Gatsby', '1925-04-10', 2, 1),
(8, 'Moby Dick', '1851-10-18', 3, 1),
(9, 'Crime and Punishment', '1866-01-01', 4, 1),
(10, "The Hitchhiker's Guide to the Galaxy", '1979-10-12', 5, 3),
(11, '1984', '1949-06-08', 1, 5),
(12, 'Animal Farm', '1945-08-17', 2, 5),
(13, 'The Haunting of Hill House', '1959-10-17', 3, 4),
(14, 'Brave New World', '1932-08-01', 4, 5),
(15, 'The Chronicles of Narnia: The Lion, the Witch and the Wardrobe', '1950-10-16', 5, 3),
(16, 'The Adventures of Huckleberry Finn', '1884-12-10', 6, 1),
(17, 'The Catch-22', '1961-10-11', 7, 1),
(18, 'The Picture of Dorian Gray', '1890-07-01', 8, 1),
(19, 'The Call of Cthulhu', '1928-02-01', 9, 4),
(20, "Harry Potter and the Philosopher's Stone", '1997-06-26', 10, 3),
(21, 'Frankenstein', '1818-01-01', 6, 4),
(22, 'A Tale of Two Cities', '1859-04-30', 7, 1),
(23, 'The Iliad', '1750-01-01', 8, 6),
(24, 'The Odyssey', '1725-01-01', 9, 6),
(25, 'The Brothers Karamazov', '1880-01-01', 10, 1),
(26, 'The Divine Comedy', '1320-01-01', 6, 6),
(27, 'The Grapes of Wrath', '1939-04-14', 7, 1),
(28, 'The Old Man and the Sea', '1952-09-01', 8, 1),
(29, 'The Count of Monte Cristo', '1844-01-01', 9, 1),

(30, "A Midsummer Night's Dream", '1596-01-01', 10, 7),

(31, "The Tricky Book", "1888-01-01",10,7);


CREATE TABLE Borrow (

  Borrow_ID INT NOT NULL PRIMARY KEY,

  Material_ID INT,

  Member_ID INT,

  Staff_ID INT,

  Borrow_Date DATE,

  Due_Date DATE,

  Return_Date DATE,

  FOREIGN KEY (Material_ID) REFERENCES Material(Material_ID) ON DELETE CASCADE,

  FOREIGN KEY (Member_ID) REFERENCES Member(Member_ID) ON DELETE CASCADE,

  FOREIGN KEY (Staff_ID) REFERENCES Staff(Staff_ID) ON DELETE CASCADE

);


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);


CREATE TABLE Authorship (

  Authorship_ID INT NOT NULL PRIMARY
KEY,

  Author_ID INT,

  Material_ID INT,

  FOREIGN KEY (Author_ID) REFERENCES
Author(Author_ID) ON DELETE CASCADE,

  FOREIGN     KEY     (Material_ID)
REFERENCES Material(Material_ID) ON
DELETE CASCADE

);


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, 4, 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);
```

Data Management Language (DML):

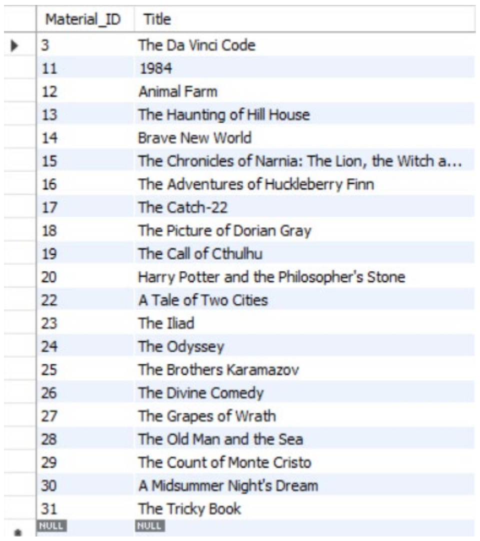
Below is the code for a variety of different queries run upon the database with successful output. However, before we start querying, we need to make sure that the correct database is selected an being used. For this we use " USE Library; " before we run al the queries

Q1. Which materials are currently available in the library?

Code:

```
USE Library;


#Query 1

#Query 1

SELECT m.Material_ID, m.Title

FROM Material m

WHERE Material_ID NOT IN (


    SELECT DISTINCT Material_ID

    FROM Borrow

    WHERE Return_Date IS NULL

);
```

Output:



```
Material_ID  Title
3            The Da Vinci Code
11           1984
12           Animal Farm
13           The Haunting of Hill House
14           Brave New World
15           The Chronicles of Narnia: The Lion, the Witch a...
16           The Adventures of Huckleberry Finn
17           The Catch-22
18           The Picture of Dorian Gray
19           The Call of Cthulhu
20           Harry Potter and the Philosopher's Stone
22           A Tale of Two Cities
23           The Iliad
24           The Odyssey
25           The Brothers Karamazov
26           The Divine Comedy
27           The Grapes of Wrath
28           The Old Man and the Sea
29           The Count of Monte Cristo
30           A Midsummer Night's Dream
31           The Tricky Book
NULL         NULL
```

Q2. Which materials are currently overdue?

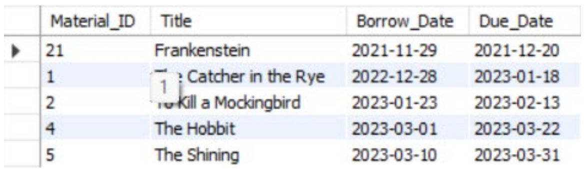
Suppose today is 04/01/2023, and show the borrow date and due date of each material.

Code:

```
USE Library;
```

```
#Query 2

SELECT m.Material_ID, m.Title,
b.Borrow_Date, b.Due_Date

FROM Material AS m, BORROW AS b

WHERE m.Material_ID = b.Material_ID
AND b.Return_Date IS NULL AND
b.Due_Date < '2023-04-01';
```

Output:



Material_ID	Title	Borrow_Date	Due_Date
21	Frankenstein	2021-11-29	2021-12-20
1	The Catcher in the Rye	2022-12-28	2023-01-18
2	To Kill a Mockingbird	2023-01-23	2023-02-13
4	The Hobbit	2023-03-01	2023-03-22
5	The Shining	2023-03-10	2023-03-31

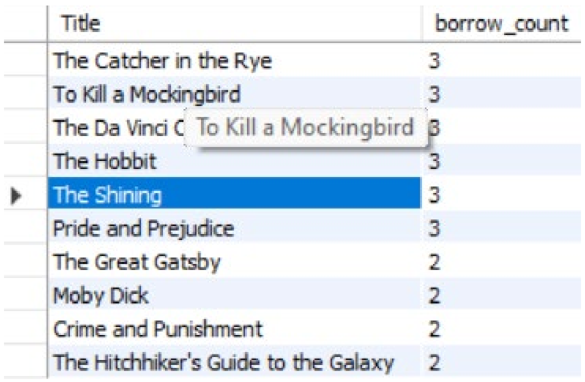
Q3. What are the top 10 most borrowed materials in the library?

Show the title of each material and order them based on their available counts

Code:

```
USE Library;


#Query 3

SELECT m.Title, count(b.material_id)
AS Materials_Borrowed

FROM borrow as b, material as m

WHERE m.material_id = b.material_id

GROUP BY m.material_id

LIMIT 10;
```

Output:



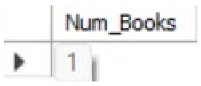
Title	borrow_count
The Catcher in the Rye	3
To Kill a Mockingbird	3
The Da Vinci C	3
The Hobbit	3
The Shining	3
Pride and Prejudice	3
The Great Gatsby	2
Moby Dick	2
Crime and Punishment	2
The Hitchhiker's Guide to the Galaxy	2

Q4. How many books has the author Lucas Piki written?

Code:

```
USE Library;


#Query 4

SELECT count(Material_ID) AS
Num_Books

FROM Author AS a, Authorship AS asp

WHERE a.Name = "Lucas Piki" AND
a.Author_ID = asp.Author_ID;
```

Output:



Num_Books
1

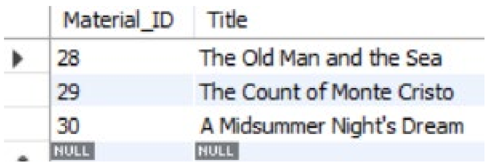
Q5. How many books were written by two or more authors?

Code:

```
USE Library;


#Query 5

SELECT m.Title

FROM Material AS m
```

```
JOIN AUTHORSHIP AS asp ON
m.material_ID = asp.material_ID

GROUP BY m.material_ID

Having count(asp.material_ID) >= 2;
```

Output:



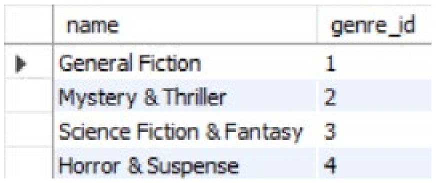
Material_ID	Title
28	The Old Man and the Sea
29	The Count of Monte Cristo
30	A Midsummer Night's Dream
NULL	NULL

Q6. What are the most popular genres in the library?

Code:

```
USE Library;


#Query 6

SELECT name, genre_id

FROM Genre

WHERE Genre_ID IN (

                              SELECT
m.Genre_id

                              FROM
Material AS m,borrow AS b

                              WHERE
m.Material_ID = b.Material_id

                              GROUP
BY m.material_id

                    HAVING
count(m.material_id) > 1

                    );
```

Output:



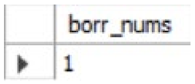
name	genre_id
General Fiction	1
Mystery & Thriller	2
Science Fiction & Fantasy	3
Horror & Suspense	4

Q7. How many materials have been borrowed from 09/2020-10/2020?

Code:

```
USE Library;


#Query 7

SELECT count(Material_ID) AS
borr_nums

FROM Borrow

WHERE Borrow_Date > '2020-09-01' AND
Borrow_Date < '2020-10-01';
```

Output:



borr_nums
1

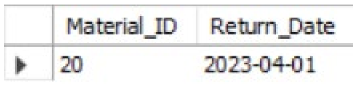
Q8. How do you update the "Harry Potter and the Philosopher's Stone"

when it is returned on 04/01/2023?

Code:

```
USE Library;


#Query 8

UPDATE Borrow

SET Return_Date = "2023-04-01"

WHERE Material_ID IN (

            SELECT Material_ID
            FROM Material

        WHERE Title = "Harry Potter
and the Philosopher's Stone"

);


SELECT m.Material_ID, b.Return_Date

FROM material AS m, Borrow AS B

WHERE m.Material_ID = b.Material_ID
AND Title = "Harry Potter and the
Philosopher's Stone";
```

Output:



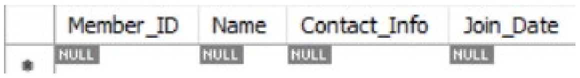
	Material_ID	Return_Date
▶	20	2023-04-01

Q9. How do you delete the member Emily Miller and all her related records from the database?

Code:

```
USE Library;


#Query 9


#In case of errors due to
incompatibility with the SQL editor,
execute the line below by removing
the "#"

#ALTER DATABASE Library SET
lo_compat_privileges TO on;

SET SQL_SAFE_UPDATES = 0;


DELETE FROM Member

WHERE Name = 'Emily Miller';


SELECT *

FROM member

WHERE name = "Emily Miller";
```

Output:



	Member_ID	Name	Contact_Info	Join_Date
✱	NULL	NULL	NULL	NULL

Q10. How do you add the following material to the database?

Title: New book

Date: 2020-08-01

Catalog: E-Books

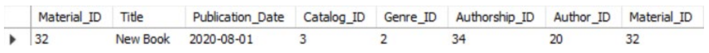
Genre: Mystery & Thriller

Author: Lucas Pipi

Code:

```
USE Library;


#Query 10

INSERT INTO material

VALUES

(32,"New Book", "2020-08-01", 3,2);


INSERT INTO authorship

VALUES

(34,20,32);


SELECT*

FROM Material AS m, Authorship AS a

WHERE m.Material_ID = 32 AND
m.Material_ID = a.Material_ID;
```

Output:



	Material_ID	Title	Publication_Date	Catalog_ID	Genre_ID	Authorship_ID	Author_ID	Material_ID
▶	32	New Book	2020-08-01	3	2	34	20	32

Design of The Database:

We can use the NOTIFY clause to create pop-up notification alerts that provide the staff with notifications about materials overdue. Using a select statement to identify un-returned books whose present day is past the due date will give regular alerts for the staff members.

```
LISTEN dues;


IF( SELECT *

FROM Borrow as b

WHERE Return_Date = NULL AND Due_Date < CURRENT_DATE) > 0

NOTIFY dues, “Over-dues For Today”;

END IF;
```

Automating a task in SQL is possible using the CREATE TRIGGER CLAUSE.
However, for this task, we will need to create another table for defaulters who have defaulted the return for more than 3 times.

```
CREATE TABLE Defaulters (
Member_ID INT NOT NULL,
Name VARCHAR(255),
Contact_Info VARCHAR(255),
Join_Date DATE,
Due_Fee_paid BINARY,
FOREIGN KEY (Member_ID) REFERENCES Member(Member_ID) ON DELETE CASCADE,
FOREIGN KEY (Name) REFERENCES Member(Name),
FOREIGN KEY (Contact_Info) REFERENCES Member(Contact_Info),
FOREIGN KEY (Join_Date) REFERENCES Member(Join_Date));
```

Here, we can add multiple steps of tasks for the client to perform on a regular basis such as a SELECT query or an UPDATE query, for our case:

```
CREATE TRIGGER Defaults_update ON Members
AFTER UPDATE
AS
BEGIN

UPDATE Defaulters
```

```
VALUES(
        SELECT *
        FROM Member
        WHERE Member_ID IN(
                SELECT Member_ID
                FROM borrow
                WHERE Member_ID = Member_ID
                AND Return_Date NOT NULL AND Return_Date > Due_Date
                GROUP BY Member_ID
                HAVING count(*) > 3))
DELETE MEMBERS
WHERE Member_ID IN (
                SELECT Member_ID
                FROM borrow
                WHERE Member_ID = Member_ID
                AND Return_Date NOT NULL AND Return_Date > Due_Date
                GROUP BY Member_ID
                HAVING count(*) > 3))

END;
```

This will automatically run a series of SQL queries if an UPDATE clause is used on the Member Table. However, for removing the values from the defaulters table, we can use the same TRIGGER clause when there is an update on defaulters table. Below is the code demonstration of how to do this:

```
CREATE TRIGGER Defaults_Delete ON Defaulters
AFTER UPDATE
AS
BEGIN
        UPDATE Members
        VALUES (
                SELECT *
                FROM Defaulters
                WHERE Member_ID IN(
                                SELECT Member_ID
                                FROM borrow
                                WHERE Member_ID = Member_ID
                                AND Return_Date NOT NULL AND Return_Date >
Due_Date
                                GROUP BY Member_ID
                                HAVING count(*) > 3))

        DELETE FROM Defaulters
        Where Due_Fee_paid = 1
End;
```