2024 Spring CS504 Principles of Data Management and Mining Project Report

Public Library Management System

Name: Anusha Goulla

G Number: G01452111

Anusha Goulla
G01452111

OVERVIEW:

The library management system developed for the Principles of Data Management and Mining project is a solution tailored for the dynamic needs of a public library set. Through database design and implementation, the system efficiently organizes and maintains a vast library resources, ranging from traditional books and magazines to e-books etc. By seamlessly integrating features for membership management, borrowing processes, and analytical insights, the system enhances the overall experience for both library staff and members. Members can easily access and borrow materials while staff can effectively manage circulation and make data-driven decisions regarding resource acquisition and management. With its user-friendly interface and functionality, the library management system represents a significant step forward in optimizing library operations and enriching the experience of library members.

SCOPE:

The project for Principles of Data Management and Mining revolves around designing and implementing a database management system tailored for a public library setting. This project includes concepts ranging from the fundamentals of database design to SQL querying and manipulation techniques. The main goal is to create a practical database system that effectively handles different aspects of running a library, such as organizing materials, managing memberships, handling borrowing activities, and generating reports. The focus is on maintaining data integrity, minimizing redundancy, and optimizing accessibility for both library staff and members. The fundamental entities of the system comprise Material, Catalog, Genre, Borrow, Author, Authorship, Member, and Staff, each equipped with specific attributes and relationships for smooth functioning of library activities.

The system has several key functionalities such as:

Materials Management: Comprehensive information about all library materials, including books, magazines, e-books, and audiobooks, is stored, and maintained. This includes details such as titles, authors, publication dates, and genres.

Membership Management: The system efficiently manages information related to library members, encompassing their names, contact details, membership numbers, and borrowing history.

Borrowing: The system facilitates the borrowing process, enabling members to check out items seamlessly. It equips library staff with the necessary information to manage the circulation of library materials. When a material is checked out, the system records it's borrow date and anticipated due date. Upon return, the system updates the return date accordingly.

Reporting and Analytics: The system generates reports on various aspects of library usage, including popular materials and other relevant statistics. These insights empower library staff to make informed, data-driven decisions regarding resource acquisition and management.

Anusha Goulla
G01452111

Entities and Relationships

1. Material

Represents individual items available in the library, such as books, magazines, e-books,

and audiobooks.

Attributes:

- Material_ID: A unique identifier for each material.
- 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.
- Genre_ID: A reference to the genre of the material.

2. Catalog

Represents a record of library materials with information on their availability and location.

Attributes:

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

3. Genre

Represents the various genres or categories of library materials.

Attributes:

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

4. Borrow

Represents the borrowing activity of library materials by members.

Attributes:

- Borrow_ID: A unique identifier for each borrowing transaction.
- Material_ID: A reference to the borrowed material.
- Member_ID: A reference to the member who borrowed the material.
- Staff_ID: A reference to the staff who processed the transaction.
- 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

Represents authors who have created library materials.

Attributes:

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

6. Authorship

Represents the relationship between authors and the materials they have created.

Attributes:

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

7. Member

Represents library members who can borrow and reserve materials.

Attributes:

- Member_ID: A unique identifier for each member.
- 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

Represents library staff who manage library resources and assist members.

Attributes:

- Staff_ID: A unique identifier for each staff member.
- 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.

Anusha Goulla
G01452111

RELATIONSHIPS:

MATERIAL BELONGS TO CATALOG:

It shows the Many to One relationship. Each catalog has many material entries, but each material is linked to one catalog entry.

Constraints:

- Material exhibits total participation, meaning every material within the library must belong to a catalog.
- Catalog demonstrates partial participation, indicating that not every catalog entry needs to have associated materials.

MATERIAL ASSOCIATED WITH GENRE:

It shows the Many to One relationship. Each genre is linked to many material entries, but each material is linked to one genre entry.

Constraints:

- Material demonstrates total participation, ensuring that every material in the library is associated with a genre.
- Catalog displays partial participation, signifying that not every genre needs to be associated with a material entry.

MATERIAL AUTHORSHIP OF AUTHOR:

It displays Many to One relationship. Where each authorship is linked to many material entries, but each material is linked to one authorship entry.

Constraints:

- Both Material and Authorship exhibit total participation, indicating that every material in the library must belong to an authorship, and conversely, every authorship must be associated with a material.

MATERIAL BORROWED OUT BORROW:

It shows one to many relationship. Every material is linked to one borrow entry, whereas each borrow is linked to many material entries.

Constraints:

- Borrow demonstrates total participation, ensuring that every borrowing activity within the library is associated with a material.
- Material displays partial participation, indicating that not every material entry needs to be associated with a borrowing activity.

BORROW ISSUED BY STAFF:

It shows many to one relationship. Every borrow is linked to one staff entry, whereas

each staff is linked to many borrow entry.

Constraints:

- Borrow exhibits total participation, ensuring that every borrowing activity within the library is associated with a staff member.
- Staff demonstrates partial participation, indicating that not every staff member needs to be associated with a borrowing activity.

BORROW LENT TO MEMBER:

It shows many to one relationship. Every borrow is linked to one member entry, whereas

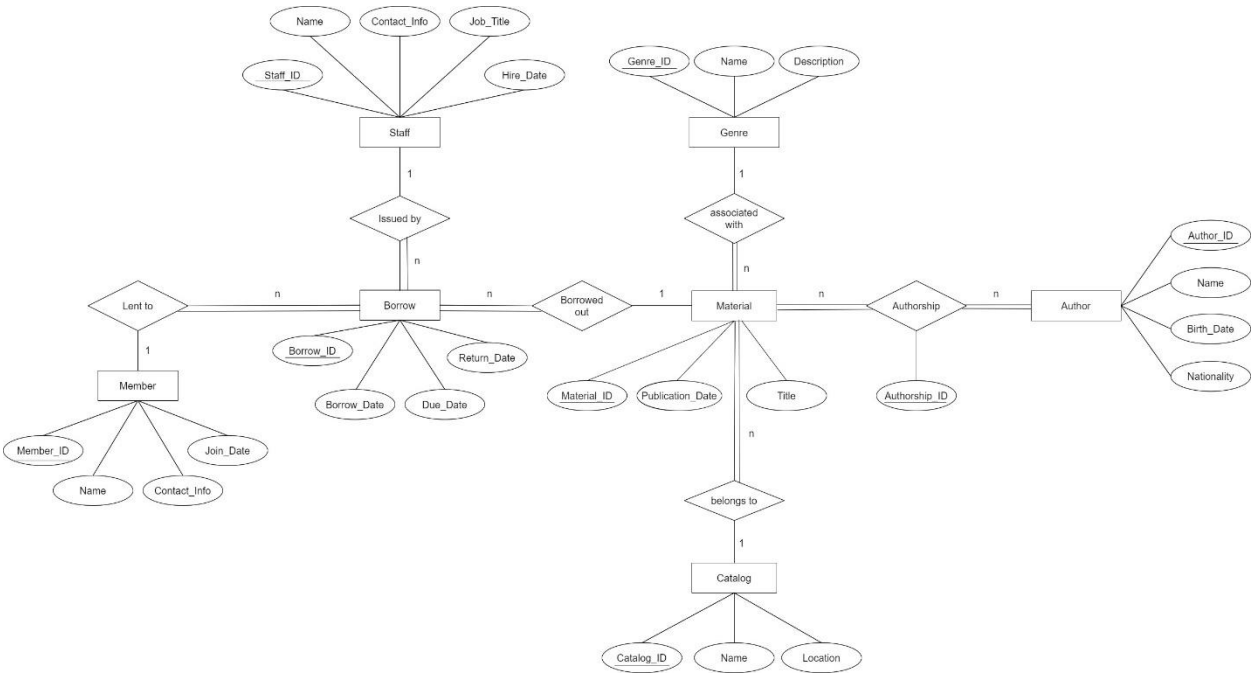
each borrow is linked to many member entries.

Constraints:

- Borrow demonstrates total participation, ensuring that every borrowing activity within the library is associated with a member.
- Member displays partial participation, indicating that not every member needs to be associated with a borrowing activity.

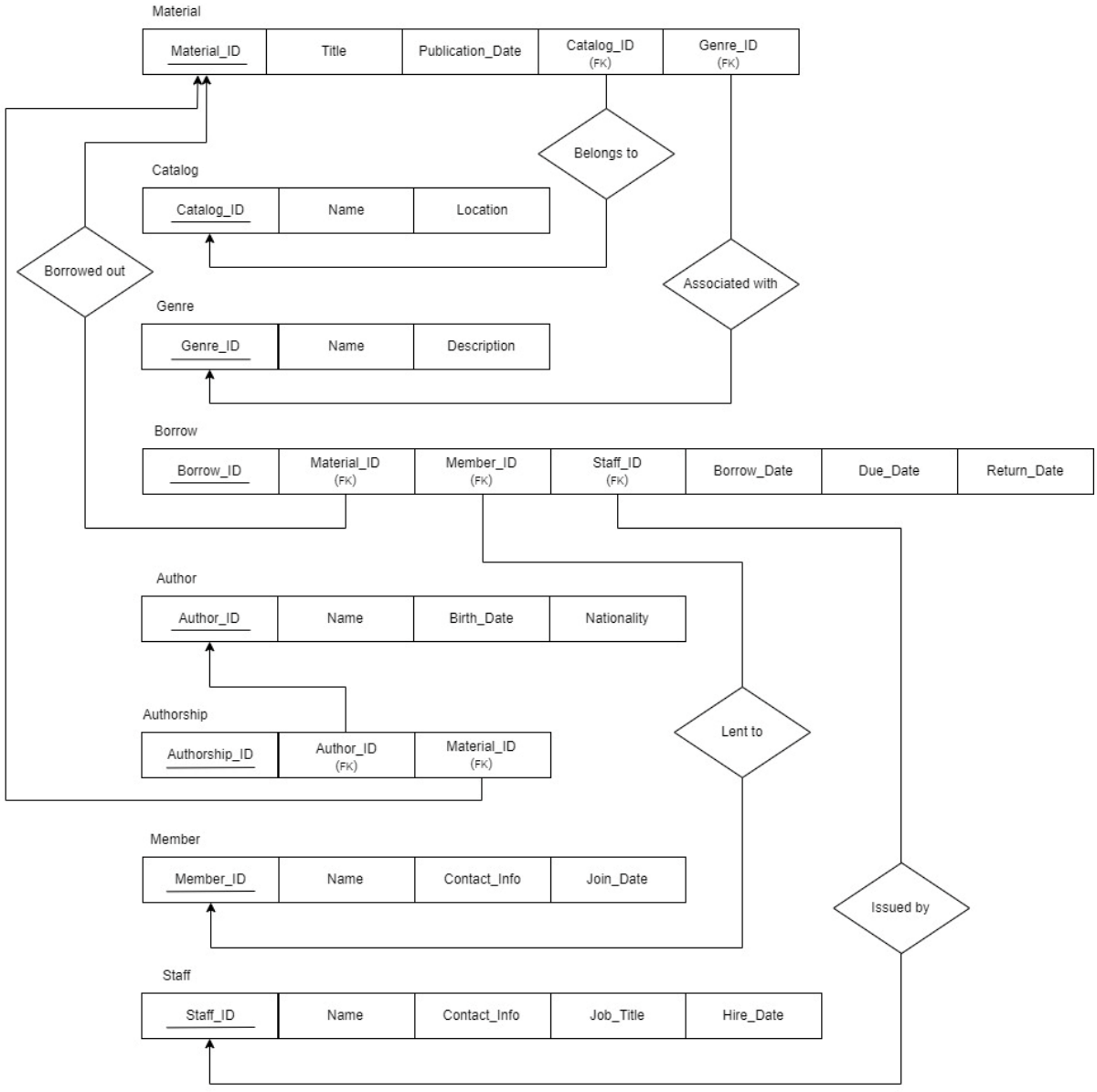
Anusha Goulla
G01452111

ENTITY – RELATIONSHIP DIAGRAM:



Anusha Goulla
G01452111

RELATIONAL SCHEMA:



DATABASE IMPLEMENTATION:

PostgreSQL was selected as the primary database management system (DBMS) for the library management system implementation. This powerful open-source object-relational database offers

a combination of features, high dependability, and good performance, making it an ideal solution for meeting the requirements of the project. PostgreSQL emerges as a practical choice for the project, providing the necessary foundation for building a scalable, reliable, and efficient library management system.

SCHEMA CREATION:

Catalog:

```
-- Create Catalog table

CREATE TABLE Catalog (

    Catalog_ID SERIAL PRIMARY KEY,

    Name VARCHAR(255) NOT NULL,

    Location VARCHAR(255)

);
```

Genre:

```
-- Create Genre table

CREATE TABLE Genre (

    Genre_ID SERIAL PRIMARY KEY,

    Name VARCHAR(255) NOT NULL,

    Description TEXT

);
```

Author:

```
-- Create Author table

CREATE TABLE Author (

    Author_ID SERIAL PRIMARY KEY,

    Name VARCHAR(255) NOT NULL,

    Birth_Date DATE,

    Nationality VARCHAR(100)
```

```
);
```

Member:

```
-- Create Member table
CREATE TABLE Member (
    Member_ID SERIAL PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Contact_Info VARCHAR(255),
    Join_Date DATE
);
```

Staff:

```
-- Create Staff table
CREATE TABLE Staff (
    Staff_ID SERIAL PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Contact_Info VARCHAR(255),
    Job_Title VARCHAR(100),
    Hire_Date DATE
);
```

Material:

```
-- Create Material table
CREATE TABLE Material (
    Material_ID SERIAL PRIMARY KEY,
    Title VARCHAR(255) NOT NULL,
    Publication_Date DATE,
    Catalog_ID INT REFERENCES Catalog(Catalog_ID),
    Genre_ID INT REFERENCES Genre(Genre_ID)
);
```

Borrow:

```
-- Create Borrow table
CREATE TABLE Borrow (
    Borrow_ID SERIAL PRIMARY KEY,
    Material_ID INT REFERENCES Material(Material_ID),
    Member_ID INT REFERENCES Member(Member_ID),
    Staff_ID INT REFERENCES Staff(Staff_ID),
    Borrow_Date DATE,
    Due_Date DATE,
    Return_Date DATE
);
```

Authorship:

```
-- Create Authorship table
CREATE TABLE Authorship (
    Authorship_ID SERIAL PRIMARY KEY,
    Author_ID INT REFERENCES Author(Author_ID),
    Material_ID INT REFERENCES Material(Material_ID)
);
```

DATABASE POPULATION:

Catalog:

```
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');

Genre:

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.');

Author:

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');

Member:

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');

Staff:

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');

Material:

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, '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);

Borrow:

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', NULL),

(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);

Authorship:

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, 22),

(24, 3, 23),

(25, 4, 24),

(26, 5, 25),

(27, 6, 26),

(28, 7, 27),

(29, 8, 28),

(30, 19, 28),

(31, 9, 29),

(32, 10, 30),

(33, 8, 30),

(34, 2, 29);

QUERYING AND MANIPULATION:

Fetch:

```
259
260 SELECT * FROM Member;
261
```

Data Output | Messages | Notifications

	member_id [PK] integer	name character varying (255)	contact_info character varying (255)	join_date date
1	1	Alice Johnson	alice.johnson@email.com	2018-01-10
2	2	Bob Smith	bob.smith@email.com	2018-03-15
3	3	Carol Brown	carol.brown@email.com	2018-06-20
4	4	David Williams	david.williams@email.com	2018-09-18
5	5	Emily Miller	emily.miller@email.com	2019-02-12
6	6	Frank Davis	frank.davis@email.com	2019-05-25
7	7	Grace Wilson	grace.wilson@email.com	2019-08-15
8	8	Harry Garcia	harry.garcia@email.com	2019-11-27
9	9	Isla Thomas	isla.thomas@email.com	2020-03-04

Insert:

```
261 --Insert
262 INSERT INTO Author (Author_ID, Name, Birth_Date, Nationality)
263 VALUES
264     (21, 'Anusha Goulla', '2001-02-20', 'Indian');
265 SELECT * FROM Author WHERE Name like '%Anusha%';
266
267
268
```

Data Output | Messages | Notifications

	author_id [PK] integer	name character varying (255)	birth_date date	nationality character varying (100)
1	21	Anusha Goulla	2001-02-20	Indian

Update:

```
268
269 Update Author set name = 'Anusha Reddy' where author_id = 21;
270 select * from Author where author_id =21;
271
272
```

Data Output | Messages | Notifications

	author_id [PK] integer	name character varying (255)	birth_date date	nationality character varying (100)
1	21	Anusha Reddy	2001-02-20	Indian

Delete:

```
272 -- Delete
273 Delete from Author where author_id = 21;
274 select * from Author where author_id =21;
275
276 -- Query1 --
```

Data Output | Messages | Notifications

author_id [PK] integer	name character varying (255)	birth_date date	nationality character varying (100)

Joins:

```
276 --Joins
277
278 Select m.title, g.name from material m inner join genre g
279 on m.genre_id = g.genre_id ;
280
281
```

Data Output | Messages | Notifications

	title character varying (255)	name character varying (255)
1	The Catcher in the Rye	General Fiction
2	To Kill a Mockingbird	General Fiction
3	The Da Vinci Code	Mystery & Thriller
4	The Hobbit	Science Fiction & Fantasy
5	The Shining	Horror & Suspense
6	Pride and Prejudice	General Fiction
7	The Great Gatsby	General Fiction
8	Moby Dick	General Fiction
9	Crime and Punishment	General Fiction

Aggregation:

Anusha Goulla
G01452111

```
281 -- Aggregation
282 SELECT Genre.name, COUNT(Material.Material_ID) AS Total_Materials
283 FROM Genre
284 LEFT JOIN Material ON Genre.Genre_ID = Material.Genre_ID
285 GROUP BY Genre.Name;
286
287
```

Data Output | Messages | Notifications

	name character varying (255)	total_materials bigint
1	General Fiction	14
2	Classics	3
3	Horror & Suspense	4
4	Dystopian & Apocalyptic	3
5	Historical Fiction	2
6	Mystery & Thriller	1
7	Epic Poetry & Mythology	0
8	Science Fiction & Fantasy	4

Sub Queries:

```
289 SELECT *
290 FROM authorship
291 WHERE Author_ID IN (
292     SELECT Author_ID
293     FROM Author
294     WHERE Nationality = 'American'
295 );
296
297
```

Data Output | Messages | Notifications

	authorship_id [PK] integer	author_id integer	material_id integer
1	2	2	2
2	4	4	4
3	6	6	6
4	11	11	11
5	18	18	18
6	22	2	22
7	25	4	24
8	27	6	26
9	34	2	29

Anusha Goulla
G01452111

Queries/Updates:

1. Which materials are currently available in the library? If a material is borrowed and not returned, it's not considered available.

Query:

SELECT * FROM Material WHERE Material_ID NOT IN (SELECT Material_ID FROM Borrow WHERE Return_Date IS NULL);

```
SELECT * FROM Material WHERE Material_ID NOT IN (SELECT Material_ID FROM Borrow
                                                 WHERE Return_Date IS NULL);
```

	material_id [PK] integer	title character varying (255)	publication_date date	catalog_id integer	genre_id integer
1	3	The Da Vinci Code	2003-04-01	3	2
2	11	1984	1949-06-08	1	5
3	12	Animal Farm	1945-08-17	2	5
4	13	The Haunting of Hill House	1959-10-17	3	4
5	14	Brave New World	1932-08-01	4	5
6	15	The Chronicles of Narnia: The Lion, the Witch and the Wardro…	1950-10-16	5	3
7	16	The Adventures of Huckleberry Finn	1884-12-10	6	1
8	17	Catch-22	1961-10-11	7	1
9	18	The Picture of Dorian Gray	1890-07-01	8	1
10	19	The Call of Cthulhu	1928-02-01	9	4
11	22	A Tale of Two Cities	1859-04-30	7	1
12	23	The Iliad	1750-01-01	8	6
13	24	The Odyssey	1725-01-01	9	6
14	25	The Brothers Karamazov	1880-01-01	10	1
15	26	The Divine Comedy	1320-01-01	6	6
16	27	The Grapes of Wrath	1939-04-14	7	1
17	28	The Old Man and the Sea	1952-09-01	8	1
18	29	The Count of Monte Cristo	1844-01-01	9	1
19	30	A Midsummer Night's Dream	1596-01-01	10	7
20	31	The Tricky Book	1888-01-01	10	7

2. Which materials are currently overdue? Suppose today is 04/01/2023 and show the borrow date and due date of each material.

Query:

SELECT Material_ID, Borrow_Date, Due_Date

FROM Borrow

WHERE Due_Date < '2023-04-01' AND Return_Date IS NULL;

```
304  SELECT Material_ID, Borrow_Date, Due_Date
305  FROM Borrow
306  WHERE Due_Date < '2023-04-01' AND Return_Date IS NULL;
307
```

Data Output | Messages | Notifications

	material_id integer	borrow_date date	due_date date
1	20	2021-10-21	2021-11-11
2	21	2021-11-29	2021-12-20
3	1	2022-12-28	2023-01-18
4	2	2023-01-23	2023-02-13
5	4	2023-03-01	2023-03-22
6	5	2023-03-10	2023-03-31

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.

Query:

SELECT Material.Title, COUNT(Borrow.Material_ID) AS Borrow_Count

FROM Material

INNER JOIN Borrow ON Material.Material_ID = Borrow.Material_ID

GROUP BY Material.Material_ID

ORDER BY Borrow_Count DESC

LIMIT 10;

```
309 SELECT Material.Title, COUNT(Borrow.Material_ID) AS Borrow_Count
310 FROM Material
311 INNER JOIN Borrow ON Material.Material_ID = Borrow.Material_ID
312 GROUP BY Material.Material_ID
313 ORDER BY Borrow_Count DESC
314 LIMIT 10;
315
```

Data Output | Messages | Notifications

	title character varying (255)	borrow_count bigint
1	The Hobbit	3
2	To Kill a Mockingbird	3
3	The Shining	3
4	Pride and Prejudice	3
5	The Catcher in the Rye	3
6	The Da Vinci Code	3
7	The Great Gatsby	2
8	Crime and Punishment	2
9	The Hitchhiker's Guide to the Galaxy	2
10	Moby Dick	2

4. How many materials has the author Lucas Piki written?

Query:

SELECT COUNT(*) AS Total_Materials

FROM Authorship

WHERE Author_ID = (SELECT Author_ID FROM Author WHERE Name = 'Lucas Piki');

```
317 SELECT COUNT(*) AS Total_Materials
318 FROM Authorship
319 WHERE Author_ID = (SELECT Author_ID FROM Author WHERE Name = 'Lucas Piki');
320
```

Data Output | Messages | Notifications

	total_materials bigint
1	1

5. How many materials were written by two or more authors?

Query:

SELECT COUNT(*) AS Total_Materials

FROM (SELECT Material_ID FROM Authorship GROUP BY Material_ID HAVING COUNT(*) >= 2) AS Multi_Authored_Materials;

```
322 SELECT COUNT(*) AS Total_Materials
323 FROM (SELECT Material_ID FROM Authorship GROUP BY Material_ID
324       HAVING COUNT(*) >= 2) AS Multi_Authored_Materials;
325
```

Data Output | Messages | Notifications

	total_materials bigint
1	4

6. What are the most popular genres in the library ranked by the total number of borrowed times of each genre?

Query:

SELECT Genre.Name, COUNT(Borrow.Material_ID) AS Borrow_Count

FROM Genre

INNER JOIN Material ON Genre.Genre_ID = Material.Genre_ID

INNER JOIN Borrow ON Material.Material_ID = Borrow.Material_ID

GROUP BY Genre.Genre_ID

ORDER BY Borrow_Count DESC;

```
326 -- Query6 --
327 SELECT Genre.Name, COUNT(Borrow.Material_ID) AS Borrow_Count
328 FROM Genre
329 INNER JOIN Material ON Genre.Genre_ID = Material.Genre_ID
330 INNER JOIN Borrow ON Material.Material_ID = Borrow.Material_ID
331 GROUP BY Genre.Genre_ID
332 ORDER BY Borrow_Count DESC;
333
```

Data Output | Messages | Notifications

	name character varying (255)	borrow_count bigint
1	General Fiction	22
2	Science Fiction & Fantasy	6
3	Horror & Suspense	5
4	Classics	3
5	Mystery & Thriller	3
6	Historical Fiction	1

7. How many materials had been borrowed from 09/2020-10/2020?

Query:

SELECT COUNT(*) AS Total_Borrowed_Materials

FROM Borrow

WHERE Borrow_Date BETWEEN '2020-09-01' AND '2020-10-31';

```
334 -- Query7 --
335 SELECT COUNT(*) AS Total_Borrowed_Materials
336 FROM Borrow
337 WHERE Borrow_Date BETWEEN '2020-09-01' AND '2020-10-31';
338
```

Data Output | Messages | Notifications

	total_borrowed_materials bigint
1	1

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

Query:

UPDATE Borrow

SET Return_Date = '2023-04-01'

WHERE Material_ID = (SELECT Material_ID FROM Material WHERE Title = 'Harry Potter and the Philosopher''s Stone') AND Return_Date IS NULL;

SELECT * from Borrow WHERE Material_ID = '20'

```
339  -- Query8 --
340  UPDATE Borrow
341  SET Return_Date = '2023-04-01'
342  WHERE Material_ID = (SELECT Material_ID FROM Material
343                       WHERE Title = 'Harry Potter and the Philosopher''s Stone')
344                       AND Return_Date IS NULL;
345  SELECT * from Borrow WHERE Material_ID = '20'
346
```

Data Output | Messages | Notifications

	borrow_id [PK] integer	material_id integer	member_id integer	staff_id integer	borrow_date date	due_date date	return_date date
1	20	20	7	2	2021-10-21	2021-11-11	2023-04-01

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

Query:

DELETE FROM Borrow WHERE Member_ID = 5;

DELETE FROM Member WHERE Name = 'Emily Miller';

SELECT * FROM Member WHERE Name = 'Emily Miller';

```
347  -- Query9 --
348  DELETE FROM Borrow WHERE Member_ID = 5;
349  DELETE FROM Member WHERE Name = 'Emily Miller';
350  SELECT * FROM Member WHERE Name = 'Emily Miller';
351
```

Data Output | Messages | Notifications

member_id [PK] integer	name character varying (255)	contact_info character varying (255)	join_date date

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 Luke

Query:

-- Insert new author

INSERT INTO Author (author_id, Name)

VALUES (21,'Lucas Luke');

select * from author where author_id = 21;

--Insert new material

INSERT INTO Material (material_id, Title, Publication_Date, Catalog_ID,

Genre_ID)

VALUES (32,'New Book', '2020-08-01',

(SELECT Catalog_ID FROM Catalog WHERE Name = 'E-Books'),

(SELECT Genre_ID FROM Genre WHERE Name = 'Mystery & Thriller'));

select * from material where material_id = 32;

Anusha Goulla
G01452111

-- Insert authorship record

INSERT INTO Authorship (Authorship_ID, Author_ID, Material_ID)

VALUES(35,(SELECT Author_ID FROM Author

WHERE Name = 'Lucas Luke'),32);

select * from authorship where authorship_id = 35;

```
353  -- Insert new author
354  INSERT INTO Author (author_id, Name)
355  VALUES (21,'Lucas Luke');
356  select * from author where author_id = 21;
357
```

Data Output | Messages | Notifications

	author_id [PK] integer	name character varying (255)	birth_date date	nationality character varying (100)
1	21	Lucas Luke	[null]	[null]

```
358  --Insert new material
359  INSERT INTO Material (material_id, Title, Publication_Date, Catalog_ID,
360  Genre_ID)
361  VALUES (32,'New Book', '2020-08-01',
362   (SELECT Catalog_ID FROM Catalog WHERE Name = 'E-Books'),
363   (SELECT Genre_ID FROM Genre WHERE Name = 'Mystery & Thriller'));
364  select * from material where material_id = 32;
365
```

Data Output | Messages | Notifications

	material_id [PK] integer	title character varying (255)	publication_date date	catalog_id integer	genre_id integer
1	32	New Book	2020-08-01	3	2

```
366  -- Insert authorship record
367  INSERT INTO Authorship (Authorship_ID, Author_ID, Material_ID)
368  VALUES(35,(SELECT Author_ID FROM Author
369  WHERE Name = 'Lucas Luke'),32);
370  select * from authorship where authorship_id = 35;
371
372
```

Data Output | Messages | Notifications

	authorship_id [PK] integer	author_id integer	material_id integer
1	35	21	32

Design of Extended Features:

1. **Alert staff about overdue materials on a daily basis?**

Identifying Overdue Materials: Writing an SQL query to select materials that are overdue based on the current date and the due date recorded in the database.

Query:

SELECT Material_ID, Title, Due_Date

FROM Borrow

WHERE Return_Date IS NULL AND Due_Date < CURRENT_DATE;

We can implement a notification system using email, SMS, or an in-app notification feature, and schedule a daily task using Task Scheduler to execute a script that queries overdue materials and sends notifications to staff.

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

To automatically deactivate memberships based on members' overdue occurrences (three times or more), and reactivate memberships upon payment of overdue fees, we can implement a monitoring system to track overdue occurrences, suspend memberships programmatically upon reaching the limit, and provide a mechanism for members to reactivate their memberships upon fee payment.

Track Overdue Occurrences: Creating a table to track overdue occurrences for each member. This table stores Member_ID and a count of overdue occurrences.

```
CREATE TABLE OverdueOccurrences (
    Member_ID INT,
    OverdueCount INT,
    PRIMARY KEY (Member_ID)
);
```

Deactivate Memberships: Use a trigger or stored procedure to monitor overdue occurrences and automatically deactivate memberships when the count exceeds the limit (three times).

```
CREATE TRIGGER UpdateOverdueCount AFTER UPDATE ON Borrow
FOR EACH ROW
BEGIN
    IF NEW.Return_Date > NEW.Due_Date THEN
        UPDATE OverdueOccurrences
        SET OverdueCount = OverdueCount + 1
        WHERE Member_ID = NEW.Member_ID;
    END IF;
END;

UPDATE Members
SET MembershipStatus = 'Inactive'
WHERE Member_ID IN (
    SELECT Member_ID
    FROM OverdueOccurrences
```

```
    WHERE OverdueCount >= 3
);
```

Reactivate Memberships: Provide a mechanism for members to reactivate their memberships upon payment of overdue fees. This might involve updating the membership status in the database once the fee is paid.

```
UPDATE Members
SET MembershipStatus = 'Active'
WHERE Member_ID = :member_id;
```

Conclusion:

In summary, the development of the library management system represents an effort to design and implement a functional and efficient database solution for the needs of a public library. Through meticulous database design, implementation, and querying, we have constructed a system capable of managing various library resources, facilitating membership management, streamlining borrowing processes, and providing insightful reporting and analytics.