PRANEETH RAVIRALA
G01448129

CS504-PROJECT

OVERVIEW:

The Library Management System project is a software designed to enhance the operational efficiency of libraries and literary resource centers. This system automates the management of books, magazines, e-books, and other resources, thereby streamlining processes to enhance user and staff experiences and accelerate various management tasks. It simplifies intricate activities like cataloging, borrowing, and returning items, and facilitates better inventory control, user registration, and fee collection, ensuring a more organized and accessible library environment.

SCOPE:

The project involves developing a database system for a public library to simplify how materials, member information, and borrowing details are managed. This system will feature tools for generating reports and analyzing data. It will organize key components like books, catalogs, genres, loans, authors, their connections to books, library members, and staff. Each component will have specific characteristics and established connections to help run the library smoothly. This setup aims to make operations more efficient, supporting the library staff in providing better service to the community by ensuring an organized and easily accessible resource system.

Place Catalog of Materials:

- Manages a variety of materials including books, magazines, e-books, and audiobooks.
- Catalogs details like title, publication date, genre, and storage location.

Managing Members Record:

- Keeps records of member names, contact information, and dates of membership initiation.
- Offers a straightforward registration process and effective account management.

Designing System for Managing Borrowing Returning Details of Materials:

- Supports the checkout of materials with monitoring of checkout and due dates.
- Manages returns efficiently to keep the system updated on material availability.

Managing Record of Authors and their Authorships:

- Records comprehensive data on authors such as names, birth dates, and nationalities.
- Maps relationships between authors and their works.

Managing Staff Records:

- Maintains staff records including job titles, contact details, and dates of hiring.
- Distributes roles to enhance operational efficiency.

Querying Platform for enhanced Searching:

- Provides an easy-to-use search interface for finding materials by title, author, or genre.
- Includes advanced querying options for creating administrative reports.

ENTITIES:

Material:

Represents items available in the library such as books, e-books, etc.

Attributes:

- Material_ID (unique identifier)
- Title
- Publication_Date
- Catalog_ID (a reference to the catalog entry for the material)
- Genre_ID (a reference to the genre of the material)

Catalog:

Records the availability and location of library materials.

Attributes:

- Catalog_ID (unique identifier)
- Name
- Location

Genre:

Categorizes materials into different genres like Fiction, Non-fiction, etc.

Attributes:

- Genre_ID (unique identifier)
- Name
- Description

Borrow:

Details the borrowing activity of members, including dates and staff involvement.

Attributes:

- Borrow_ID (unique identifier)
- Material_ID (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
- Due_Date
- Return_Date

Author:

Contains information about the authors of the materials.

Attributes:

- Author_ID (unique identifier)
- Name
- Birth_Date
- Nationality

Authorship:

Associates authors with the materials they have created.

Attributes:

- Authorship_ID (unique identifier)
- Author_ID (reference to the author)
- Material_ID (a reference to the material authored)

Member:

Represents the library members who borrow materials.

Attributes:

- Member_ID (unique identifier)
- Name
- Contact_Info
- Join_Date

Staff:

Represents the employees of the library responsible for managing various tasks.

Attributes:

- Staff_ID (unique identifier)
- Name
- Contact_Info
- Job_Title
- Hire_Date

Relationship & Participation:

1. Material - This entity represents the items available in the library. It relates to Catalog with a one-to-many relationship indicating that each entry in the catalog can correspond to multiple materials (partial participation for Material, total for Catalog), and to Authorship with a many-to-many relationship through Types indicating that each material may have multiple authorships, and each authorship can be associated with multiple materials (partial participation for both).

2. Authorship - Represents the authorship details of the materials. It is in a many-to-one relationship with the Author (partial participation for Authorship, total for Author), indicating that

each authorship record is related to exactly one author, but an author can have multiple authorships.

3. Author - An entity representing individuals who author materials. The relationship to Authorship indicates that authors can have none or many authorship records (partial participation).

4. Genre - Represents different genres or categories that materials can belong to. It is related to Material through Types in a many-to-many relationship (partial participation for both), meaning material can belong to multiple genres, and a genre can include multiple materials.

5. Catalog - Contains catalog records in the library and is in a one-to-many relationship with Borrowing (partial participation for Catalog, total for Borrowing), which means a catalog entry is associated with none or many borrowing records.

6. Borrow - Represents individual borrow transactions. It has a many-to-one relationship with Member and Staff through Borrows and Manages respectively, indicating that a borrow record is managed by one staff member and taken out by one member (total participation for both Member and Staff).

7. Member - Represents library members. Members can have none or many borrow transactions (partial participation).

8. Staff- Represents library staff. Each staff member can manage none or many borrow transactions (partial participation).

Database Implementation:

The implementation of the library management system utilizes PostgreSQL, an open-source object-relational database management system (DBMS). Renowned for its robustness, reliability, and high performance, PostgreSQL is an ideal choice for this project. It fulfills the necessary criteria to manage library data effectively, providing strong functionality and dependable performance. This DBMS supports the system's need to organize and maintain comprehensive records on materials, members, staff, and borrowing transactions efficiently. Overall, PostgreSQL's capabilities make it a superior solution for facilitating the streamlined operations required by the library management system.

SCHEMA CREATION:

Material:

```
create table Material(
```

```
Material_ID numeric not null,
Title varchar(100),
Publication_Date date,
Catalog_ID numeric,
Genre_ID numeric,
primary key(Material_ID),
foreign key(Catalog_ID) references Catalog(Catalog_ID) on delete cascade on update cascade,
foreign key(Genre_ID) references Genre(Genre_ID) on delete cascade on update cascade
);

LOAD DATA INFILE 'C:/Users/91957/Desktop/Material.csv'
INTO TABLE Material
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;

select * from Material;
```

Catalog:

```
create table Catalog(
Catalog_ID numeric not null,
Name Varchar(100),
Location Varchar(100),
primary key(Catalog_ID));

LOAD DATA INFILE 'C:/Users/91957/Desktop/Catalog.csv'
INTO TABLE Catalog
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;
```

select * from catalog;

Genre:

```
create table Genre(
Genre_ID numeric not null,
Name Varchar(100),
Description text,
primary key(Genre_ID));

LOAD DATA INFILE 'C:/Users/91957/Desktop/Genre.csv'
INTO TABLE Genre
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;

select * from Genre;
```

Borrow:

```
create table Borrow(
Borrow_ID numeric not null,
Material_ID numeric,
Member_ID numeric,
Staff_ID numeric,
Borrow_Date date,
Due_Date date,
Return_Date date,
primary key(Borrow_ID),
foreign key(Material_ID) references Material(Material_ID) on delete cascade on update cascade,
foreign key(Member_ID) references Member(Member_ID) on delete cascade on update
cascade,
foreign key(Staff_ID) references Staff(Staff_ID) on delete cascade on update cascade
```

```
);

LOAD DATA INFILE 'C:/Users/91957/Desktop/Borrow.csv'
INTO TABLE Borrow
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;

update  Borrow
set Return_Date=null
where Return_Date='0000-00-00';

select * from Borrow;
```

Author:

```
create table Author(
Author_ID numeric not null,
Name Varchar(20),
Birth_Date date,
Nationality varchar(10),
primary key(Author_ID));

LOAD DATA INFILE 'C:/Users/91957/Desktop/Author.csv'
INTO TABLE Author
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;

select * from Author;
```

Authorship:

```
create table Authorship(
Authorship_ID numeric not null,
Author_ID numeric,
Material_ID numeric,
primary key(Authorship_ID),
foreign key(Author_ID) references Author(Author_ID) on delete cascade on update cascade,
foreign key(Material_ID) references Material(Material_ID) on delete cascade on update
cascade);

LOAD DATA INFILE 'C:/Users/91957/Desktop/Authorship.csv'
INTO TABLE Authorship
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;

select * from Authorship;
```

Member:

```
create table Member(
Member_ID numeric not null,
Name varchar(20),
Contact_Info varchar(30),
Join_Date date,
primary key(Member_ID)
);
LOAD DATA INFILE 'C:/Users/91957/Desktop/Member.csv'
INTO TABLE Member
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;
```

```
select * from Member;
```

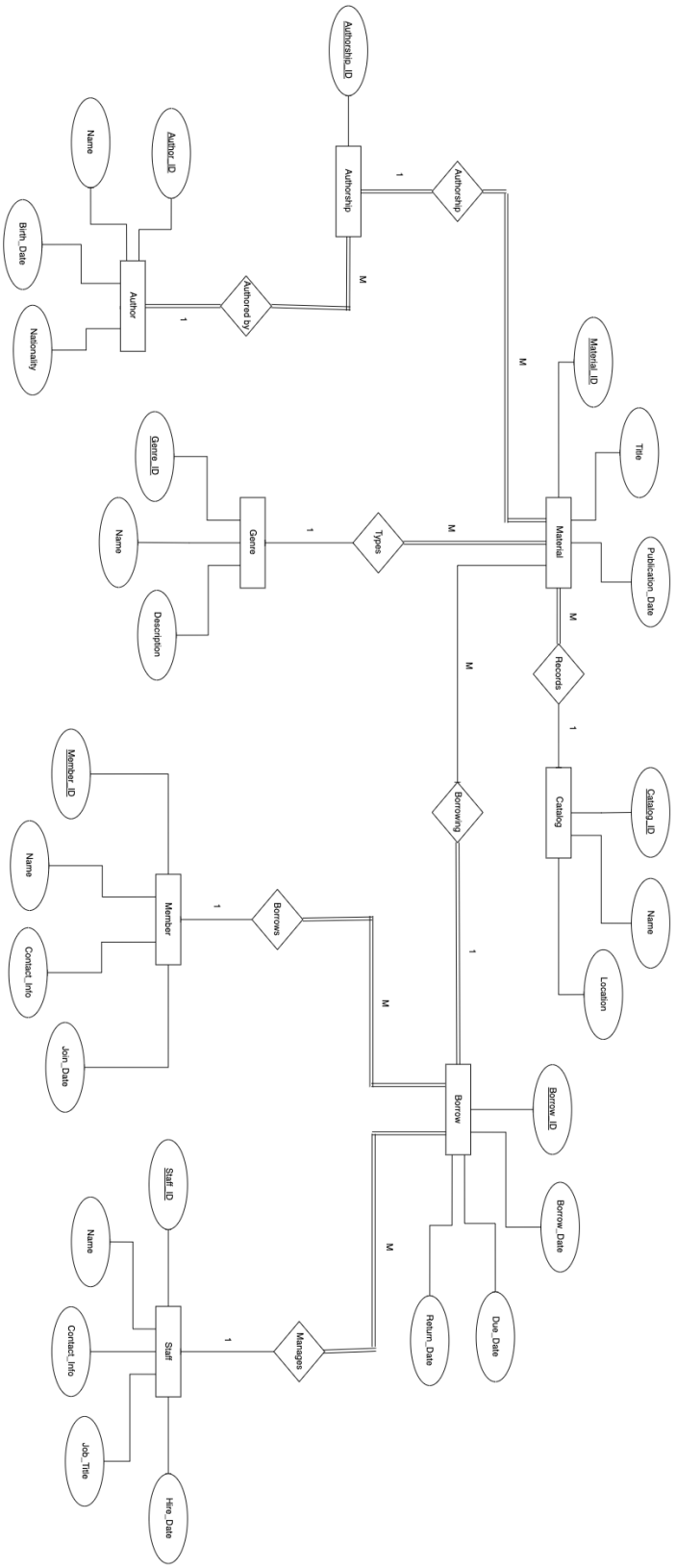
Staff:

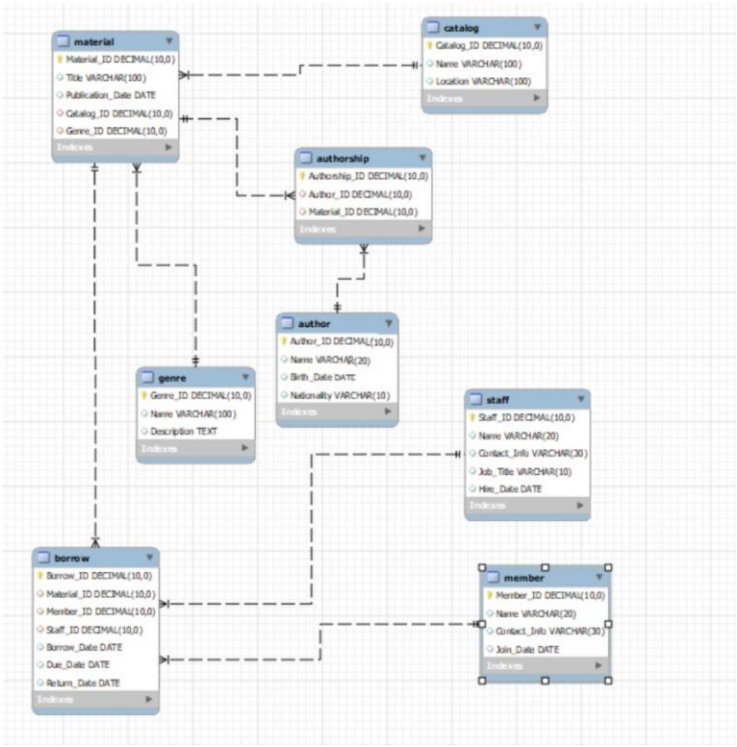
```
create table Staff(
Staff_ID numeric not null,
Name varchar(20),
Contact_Info varchar(30),
Job_Title varchar(10),
Hire_Date date,
primary key(Staff_ID)
);

LOAD DATA INFILE 'C:/Users/91957/Desktop/Staff.csv'
INTO TABLE Staff
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;

select * from Staff;
```

Display of Tables in MySQL ‘library’ database schema.

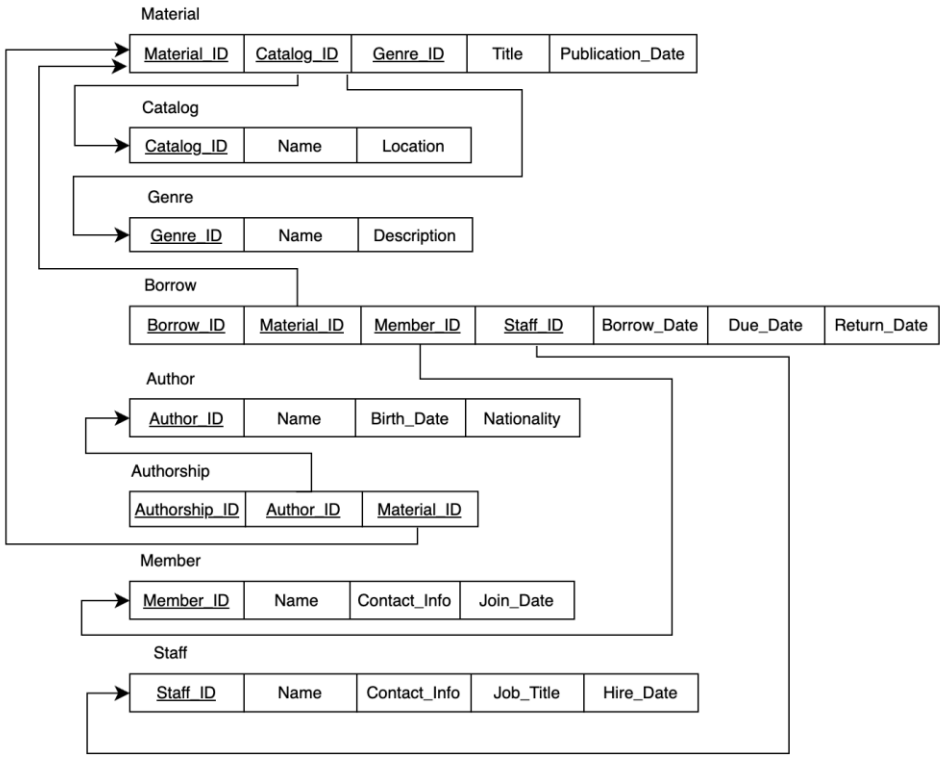

library
Tables
author
authorship
borrow
catalog
genre
material
member
staff
Views
Stored Procedures
Functions

ER DIAGRAM:


Authorship_ID
Authorship
Name
Author_ID
Author
Birth_Date
Nationality
Authored by
Authorship
Material_ID
Material
Title
Publication_Date
Genre_ID
Genre
Name
Description
Types
Records
Catalog
Catalog_ID
Name
Location
Member_ID
Member
Name
Contact_Info
Join_Date
Borrows
Borrowing
Borrow
Borrow_ID
Borrow_Date
Return_Date
Due_Date
Staff_ID
Staff
Name
Contact_Info
Job_Title
Hire_Date
Manages
1
M



RELATIONAL SCHEMA:



QUERYING TECHNIQUES:

1. Searching

```
146     # Quering and Manipulation
147     #1. Searching
148 •   select Catalog_ID,Genre_ID from Material
149     where Title='The Shining';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Catalog_ID	Genre_ID
5	4

2. Inserting

```
151     #2.Inserting
152 •   insert into Staff values (5,'Praneeth Ravirala','praneeth2108ravirala@gmail.com','Librarian','2001-08-21');
153 •   select * from Staff where Name ='Praneeth Ravirala';
154
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Staff_ID	Name	Contact_Info	Job_Title	Hire_Date
5	Praneeth Ravirala	praneeth2108ravirala@gmail.com	Librarian	2001-08-21
NULL	NULL	NULL	NULL	NULL

3. Updating

```
155     #3. Updating
156 •   update Staff
157     set Name='Praneeth'
158     where Name='Praneeth Ravirala';
159 •   select * from Staff where Name='Praneeth';
160
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Staff_ID	Name	Contact_Info	Job_Title	Hire_Date
5	Praneeth	praneeth2108ravirala@gmail.com	Librarian	2001-08-21
NULL	NULL	NULL	NULL	NULL

4. Deleting

```
161    #Deleting
162    Delete from Staff
163    where Name='Praneeth';
164    select * from Staff where Name='Praneeth';
165
```

Result Grid | Filter Rows: | Edit: | Export/Import:

Staff_ID	Name	Contact_Info	Job_Title	Hire_Date
NULL	NULL	NULL	NULL	NULL

5. Inner Join

```
166    #Inner Join
167    select ma.Title,ca.Name from Material as ma
168    inner join Catalog as ca on ma.Catalog_ID=ca.Catalog_ID
169    where ca.Name='Magazines';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Title	Name
To Kill a Mockingbird	Magazines
The Great Gatsby	Magazines
Animal Farm	Magazines

6. Aggregation

```
171    #Aggregation
172    select count(ma.Title) as Count_Title,ca.Name from Material as ma
173    inner join Catalog as ca on ma.Catalog_ID=ca.Catalog_ID
174    group by ca.Name
175    order by Count_Title asc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Count_Title	Name
3	Maps
3	Audiobooks
3	Books
3	Sheet Music
3	Newspaper
3	E-Books
3	Novels
3	Journals
3	Magazines
4	Educational

7. Sub querying

```
177     #Subqueries
178 •   select Title from Material
179     Where Genre_ID= (select Genre_ID from Genre
180                 where Name='Horror & Suspense');
181
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Title
The Shining
The Haunting of Hill House
The Call of Cthulhu
Frankenstein

SQL QUERYING:

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

```
183     #Query1
184 •   select ma.Material_ID,ma.Title
185     from Material as ma
186     where ma.Material_ID not in (
187         select bo.Material_ID
188         from Borrow as bo
189         where bo.Return_Date is null
190     );
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cel

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 an...
16	The Adventures of Huckleberry Finn
17	The Catch-22
18	The Picture of Dorian Gray
19	The Call of Cthulhu

```
183     #Query1
184 •   select ma.Material_ID,ma.Title
185     from Material as ma
186     where ma.Material_ID not in (
187         select bo.Material_ID
188         from Borrow as bo
189         where bo.Return_Date is null
190     );
191
192     #2.
193
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Material_ID	Title
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

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

```
193     #Query2
194 •   select ma.Material_ID,ma.Title,bo.Borrow_Date, bo.Due_Date from Material as ma
195     inner join Borrow as bo on ma.Material_ID=bo.Material_ID
196     where bo.Return_Date is null and bo.Due_Date<='2023-04-01';
197
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Material_ID	Title	Borrow_Date	Due_Date
20	Harry Potter and the Philosopher's Stone	2021-10-21	2021-11-11
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

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.

```
197  #Query3
198  select ma.Material_ID,ma.Title, count(bo.Material_ID) as CountMaterial from Material as ma
199  inner join Borrow as bo on ma.Material_ID = bo.Material_ID
200  group by ma.Title,ma.Material_ID
201  order by CountMaterial desc limit 10;
202
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Material_ID	Title	CountMaterial
4	The Hobbit	3
1	The Catcher in the Rye	3
6	Pride and Prejudice	3
3	The Da Vinci Code	3
2	To Kill a Mockingbird	3
10	The Hitchhiker's Guide to the Galaxy	2
7	The Great Gatsby	2
9	Crime and Punishment	2
8	Moby Dick	2
5	The Shining	2

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

```
203  #Query4
204  select au.Name,count(distinct ma.Material_ID) as authors_materials from Author as au
205  inner join Authorship as ap on au.Author_ID = ap.Author_ID
206  inner join Material as ma on ap.Material_ID = ma.Material_ID
207  where au.Name = 'Lucas Piki'
208  group by au.NAME;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	authors_materials
Lucas Piki	1

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

```
210  #Query5
211  select ma.Material_ID,ma.Title,count(au.Author_ID) as Authors_Count from Author as au
212  inner join Authorship as ap on au.Author_ID = ap.Author_ID
213  inner join  Material as Ma on ap.Material_ID = ma.Material_ID
214  group by ma.Title,ma.Material_ID
215  having count(ap.Author_ID) >= 2
216  order by ma.MATERIAL_ID desc;
217
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Material_ID	Title	Authors_Count
30	A Midsummer Night's Dream	2
29	The Count of Monte Cristo	2
28	The Old Man and the Sea	2
22	A Tale of Two Cities	2

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

```
218     #Query6
219 •   select gn.Name, Count(bo.Borrow_ID) as Borrowed_Count from Genre as gn
220     inner join Material as ma on gn.Genre_ID=ma.Genre_ID
221     inner join Borrow as bo on ma.Material_ID=bo.Material_ID
222     group by gn.Name
223     order by Borrowed_Count desc;
224
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	Borrowed_Count
General Fiction	20
Science Fiction & Fantasy	6
Horror & Suspense	4
Classics	3
Mystery & Thriller	3
Historical Fiction	1

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

```
225     #Query7
226 •   select Material_ID, count(distinct Material_ID) as Borrowed_Count from Borrow
227     where Borrow_Date >= '2020-09-01' and Borrow_Date<= '2020-10-01'
228     group by Material_ID
229     order by Borrowed_Count desc;
230
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Material_ID	Borrowed_Count
3	1

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

```
230
231     #Query8
232 •   update Borrow as bo
233     set bo.Return_Date = '2023-04-01'
234     where bo.Material_ID = (
235         select ma.Material_ID
236         from Material as ma
237         where ma.Title = 'Harry Potter and the Philosopher''s Stone') and bo.Return_Date is null;
238 •   select * from Borrow
239     where Return_Date='2023-04-01';
240
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Borrow_ID	Material_ID	Member_ID	Staff_ID	Borrow_Date	Due_Date	Return_Date
20	20	7	2	2021-10-21	2021-11-11	2023-04-01
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

```
241
242     #Query9
243 •   delete from Member where Name='Emily Miller';
244 •   select * from Member
245     where Name='Emily Miller';
246
```

Result Grid | Filter Rows: | Edit: | Export/Import:

Member_ID	Name	Contact_Info	Join_Date
NULL	NULL	NULL	NULL

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

```
247     #Query10
248 •   begin;
249 •   insert into Material (Material_ID,Title,Publication_Date,Catalog_ID,Genre_ID)
250     select (select max(Material_ID)+1 from Material),
251             'New Book','2020-08-01',(select Catalog_ID from Catalog where Name='E-Books'),
252             (select Genre_ID from Genre where Name='Mystery & Thriller');
253 •   select * from material
254     where Title='New Book';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Material_ID	Title	Publication_Date	Catalog_ID	Genre_ID
32	New Book	2020-08-01	3	2
NULL	NULL	NULL	NULL	NULL

```
256 •   insert into Author (Author_ID, Name)
257     select (select max(Author_ID)+1 from Author), 'Lucas Luke';
258 •   select * from Author
259     where Name='Lucas Luke';
```

Result Grid | Filter Rows: | Edit: | Export/Import:

Author_ID	Name	Birth_Date	Nationality
21	Lucas Luke	NULL	NULL
NULL	NULL	NULL	NULL

```
261 •   insert into Authorship (Authorship_ID,Author_ID, Material_ID)
262     select (select max(Authorship_ID)+1 from Authorship),(select Author_ID from Author where Name='Lucas Luke'),
263             (select max(Material_ID) from Material);
264 •   select * from Authorship
265     where Author_ID= (select Author_ID from Author where Name='Lucas Luke');
266 •   commit;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Authorship_ID	Author_ID	Material_ID
35	21	32
NULL	NULL	NULL

DESIGN:

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

```
select Material.Title from Material inner join Borrow
on Material.Material_ID=Borrow.Material_ID
where Borrow.Due_Date< current_date and Borrow.Return_Date is null;
```

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.

1.Create table for storing information about about due date, membership status, recent payment status and overdue count.

```
create table MembershipStatus(
ID numeric not null,
Due_Date date,
Status Varchar(10) default 'active',
Overdue_Count numeric default 0,
Recent_Payment_Status varchar(5),
primary key(ID),
foreign key(ID) references Member(Member_ID) on update cascade on delete cascade
);
```

2. Insert values from Member table and also additional details about subscription status.

```
Insert into MembershipStatus
select (select Member_ID from Member),('2024-04-10'),('active'),(0),('Yes');
```

3.step to deactivate membership

```
update MembershipStatus
set Status='deactive', Overdue_Count=Overdue_Count+1,Recent_Payment_Status='NO'
where Due_Date < current_date and Overdue_Count > 3;
```

4. step to reactive membership.

```
update MembershipStatus
set Status='active',Overdue_Count=0
where Recent_Payment_Status='Yes';
```