**CS504-INDIVIDUAL PROJECT**

**-Mounika Jaggampudi**

**G01359781**

Table of Contents

[Database Design: 2](#_Toc134038157)

[Entity-Relationship (ER) diagram: 3](#_Toc134038158)

[Relational Schema Diagram: 4](#_Toc134038159)

[Database normalization: 4](#_Toc134038160)

[Procedure for Executing Queries: 5](#_Toc134038161)

[QUERIES: 5](#_Toc134038162)

[Design 12](#_Toc134038163)

# Database Design:

Scope of the project:

The scope of the project is to design and implement a database management system for a public library. The system will manage library resources, including books, magazines, digital media, and other materials, and provide efficient access to member information and facilitate borrowing and tracking of library materials.

The entities and their relationships are:

Material: A library item available for borrowing, such as books, magazines, e-books, and audiobooks.

Catalog: A record of library materials with information on their availability and location.

Genre: A category or type of library materials.

Borrow: The borrowing activity of library materials by members.

Author: An individual who has created library materials.

Authorship: The relationship between authors and the materials they have created.

Member: A person who is a member of the library and can borrow and reserve materials.

Staff: A staff member who manages library resources and assists members.

# Entity-Relationship (ER) diagram:

**Diagram

Description automatically generated**

# Relational Schema Diagram:

**Diagram

Description automatically generated**

# Database normalization:

Database normalization is the process of organizing the attributes and tables in a database to minimize redundancy and ensure data integrity. Normalization ensures that the database is free from inconsistencies, update anomalies, and data redundancy, which can affect database performance and accuracy.

The database schema presented in the ER diagram is already normalized. The entities and their relationships are in a third normal form (3NF), which means that each attribute is dependent only on the primary key and there are no transitive dependencies. Therefore, further normalization is not required.

# Procedure for Executing Queries:

I used Oracle Database platform for creating public library database and executed all the following queries. Along with every query, there is an image of my execution attached below depicting the result of the query. [The tables that I have created are Material, Catalog, Genre, Author, Member, Staff, Borrow, along with the relationship Authorship. Also inserted the sample data given, I have classified the source codes into two SQL files namely, create\_PublicLibrary for creating tables and Insert\_PublicLibrary for inserting the sample data.

# QUERIES:

1. Which materials are currently available in the library?

SELECT Title

FROM Material

WHERE Material\_ID NOT IN (

SELECT Material\_ID

FROM Borrow

WHERE Return\_Date IS NULL

);Graphical user interface, application

Description automatically generated

2. Which materials are currently overdue?

SELECT Material.Title, Borrow.Borrow\_Date, Borrow.Due\_Date

FROM Borrow

JOIN Material ON Borrow.Material\_ID = Material.Material\_ID

WHERE Borrow.Return\_Date IS NULL AND Borrow.Due\_Date < DATE '2023-04-01';

Graphical user interface, table

Description automatically generated

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

SELECT \* FROM(

SELECT m.title,COUNT(b.material\_id) AS borrowed\_count

FROM borrow b

INNER JOIN material m ON b.material\_id=m.material\_id

GROUP BY m.title

ORDER BY borrowed\_count DESC)

WHERE rownum <= 10;

Table

Description automatically generated

-- 4. How many books has the author Lucas Piki written?

SELECT COUNT(material\_id) AS books\_count

FROM authorship a

INNER JOIN author au ON a.author\_id=au.author\_id

WHERE au.name='Lucas Piki';

Graphical user interface, application, Teams

Description automatically generated

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

SELECT SUM(COUNT (DISTINCT Material\_ID)) as "Number of books by more than one author"

FROM Authorship

GROUP BY Material\_ID

HAVING COUNT(Material\_ID) > 1;

Graphical user interface, application, Teams

Description automatically generated

-- 6. What are the most popular genres in the library?

SELECT Genre.Name, COUNT(\*) AS Borrow\_Count

FROM Borrow

JOIN Material ON Borrow.Material\_ID = Material.Material\_ID

JOIN Genre ON Material.Genre\_ID = Genre.Genre\_ID

GROUP BY Genre.Genre\_ID

ORDER BY Borrow\_Count DESC;

Table

Description automatically generated

SELECT genre.name, COUNT(borrow.borrow\_id) AS borrow\_count

FROM genre

JOIN material ON genre.genre\_id = material.genre\_id

JOIN borrow ON material.material\_id = borrow.material\_id

WHERE borrow.return\_date IS NOT NULL

GROUP BY genre.name

ORDER BY borrow\_count DESC;

Table

Description automatically generated

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

SELECT COUNT(DISTINCT Material\_ID)

FROM Borrow

WHERE Borrow\_Date BETWEEN '2020-09-01' AND '2020-10-31';

Graphical user interface, application

Description automatically generated

-- 8. How do you update the “Harry Potter and the Philosopher's Stone” when it is returned on

-- 04/01/2023?

UPDATE Borrow

SET Return\_Date = 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;

Graphical user interface, application, Teams

Description automatically generated

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

DELETE FROM Borrow

WHERE Member\_ID = (

SELECT Member\_ID

FROM Member

WHERE Name = 'Emily Miller'

);

DELETE FROM Member

WHERE Name = 'Emily Miller';

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

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

INSERT INTO Material (Material\_ID, Title, Publication\_Date, Catalog\_ID, Genre\_ID) VALUES(32, 'New Book', DATE'2020-08-01',3,2);

INSERT INTO Authorship (Authorship\_ID, Author\_ID, Material\_ID) VALUES(34, 20, 32);

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

# Design:

1. I can add a script or a stored procedure that runs every day to check for any late materials in the database, extending the current database system to notify personnel about overdue materials daily. To locate any items that still have a member-checked-out status but have a due date that is earlier than today, the script would need to query the database. The script can notify staff members via email or another preferred mode of communication once it discovers any late items.
2. I can construct a trigger that is invoked when a member checks out a material to automatically deactivate a membership depending on the member's overdue occurrence (>= three times) and revive it once the member pays the overdue cost. If the member has any late occurrences larger than or equal to three, the trigger will need to check that fact. If so, the member's status would be changed to "deactivated." You can change the status to "active" after the member pays the past-due charge.