**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.

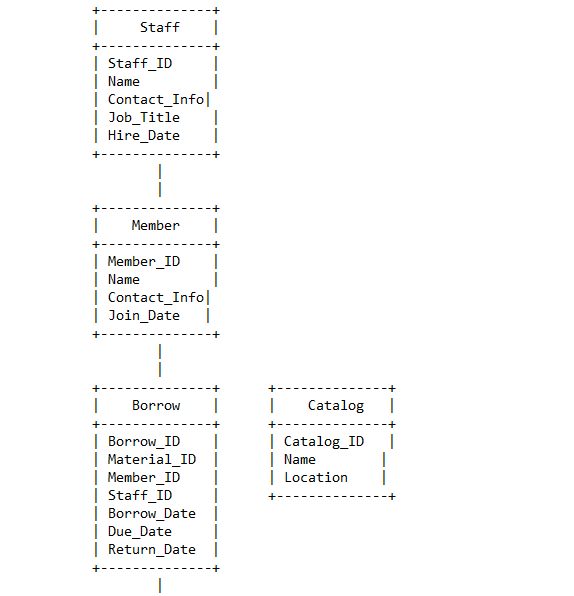
Author: An individual who has created library materials.

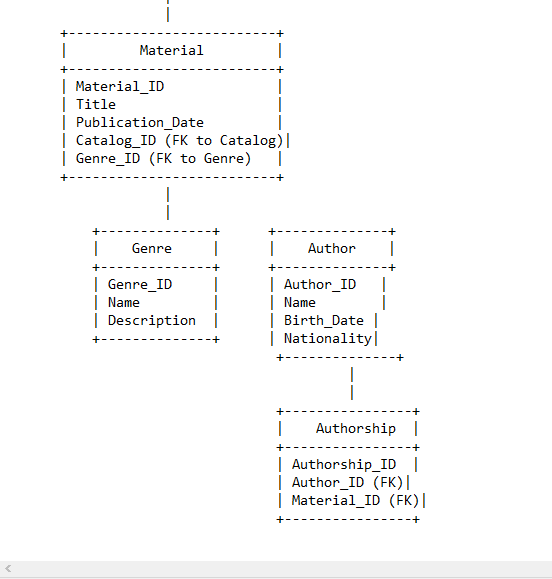
Authorship: The relationship between authors and the materials they have created.

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:**





**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.

**DDL COMMANDS**

CREATE TABLE Authors (

Author\_ID INT PRIMARY KEY,

First\_Name VARCHAR(50),

Last\_Name VARCHAR(50)

);

CREATE TABLE Authorship (

Material\_ID INT,

Author\_ID INT,

CONSTRAINT pk\_authorship PRIMARY KEY (Material\_ID, Author\_ID),

CONSTRAINT fk\_authorship\_material FOREIGN KEY (Material\_ID)

REFERENCES Material(Material\_ID)

ON DELETE CASCADE,

CONSTRAINT fk\_authorship\_author FOREIGN KEY (Author\_ID)

REFERENCES Authors(Author\_ID)

ON DELETE CASCADE

);

CREATE TABLE Material (

Material\_ID INT PRIMARY KEY,

Title VARCHAR(255),

Publication\_Date DATE,

Catalog\_ID INT,

Genre\_ID INT,

CONSTRAINT fk\_material\_catalog FOREIGN KEY (Catalog\_ID)

REFERENCES Catalog(Catalog\_ID),

CONSTRAINT fk\_material\_genre FOREIGN KEY (Genre\_ID)

REFERENCES Genre(Genre\_ID)

);

CREATE TABLE Catalog (

Catalog\_ID INT PRIMARY KEY,

Name VARCHAR(50),

Location VARCHAR(50)

);

CREATE TABLE Genre (

Genre\_ID INT PRIMARY KEY,

Name VARCHAR(50),

Description VARCHAR(255)

);

CREATE TABLE Borrow (

Borrow\_ID INT PRIMARY KEY,

Material\_ID INT,

Member\_ID INT,

Staff\_ID INT,

Borrow\_Date DATE,

Due\_Date DATE,

Return\_Date DATE,

CONSTRAINT fk\_borrow\_material FOREIGN KEY (Material\_ID)

REFERENCES Material(Material\_ID)

ON DELETE CASCADE,

CONSTRAINT fk\_borrow\_member FOREIGN KEY (Member\_ID)

REFERENCES Member(Member\_ID)

ON DELETE CASCADE,

CONSTRAINT fk\_borrow\_staff FOREIGN KEY (Staff\_ID)

REFERENCES Staff(Staff\_ID)

ON DELETE CASCADE

);

**QUERIES**

Which materials are currently available in the library?

SELECT \*

FROM material

WHERE available\_count > 0;

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

SELECT material.material\_id, material.title, borrow.borrow\_date, borrow.due\_date

FROM material

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

WHERE borrow.return\_date IS NULL AND borrow.due\_date < '2023-04-01';

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?

SELECT material.title, COUNT(borrow.borrow\_id) AS borrow\_count, material.available\_count

FROM material

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

GROUP BY material.material\_id

ORDER BY borrow\_count DESC, material.available\_count DESC

LIMIT 10;

How many books has the author Lucas Piki written?

SELECT COUNT(\*)

FROM material

JOIN book\_author ON material.material\_id = book\_author.book\_id

JOIN author ON book\_author.author\_id = author.author\_id

WHERE author.name = 'Lucas Piki' AND material.type = 'book';

How many books were written by two or more authors?

SELECT COUNT(DISTINCT book\_id)

FROM book\_author

GROUP BY book\_id

HAVING COUNT(author\_id) >= 2;

What are the most popular genres in the library?

SELECT genre, COUNT(material\_id) AS material\_count

FROM material\_genre

GROUP BY genre

ORDER BY material\_count DESC;

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

SELECT COUNT(\*)

FROM borrow

WHERE borrow\_date >= '2020-09-01' AND borrow\_date < '2020-11-01';

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 = '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;

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

DELETE FROM member WHERE name = 'Emily Miller';

DELETE FROM borrow WHERE member\_id NOT IN (SELECT member\_id FROM member);

How do you add the following material to the database?

INSERT INTO material (title, type, total\_count, available\_count, publish\_year)

VALUES ('The Catcher in the Rye', 'book', 5, 5, 1951);

INSERT INTO book (material\_id, ISBN, edition)

VALUES (LAST\_INSERT\_ID(), '9780316769174', 'First Edition');

INSERT INTO book\_author (book\_id, author\_id)

VALUES (LAST\_INSERT\_ID(), (SELECT author\_id FROM author WHERE name = 'J.D. Salinger'));

INSERT INTO material\_genre (material\_id, genre)

VALUES (LAST\_INSERT\_ID(), 'Fiction');