The goal of normalizing a database schema is to reduce data redundancy, improve data integrity, and ensure that it adheres to the rules of each normal form. The 3NF tables were created to achieve these goals. Here's a detailed explanation of why the tables were divided into their current structure in the 3NF:

Author, Publisher, and Genre tables: These tables were created to eliminate data redundancy and maintain data integrity. In the original schema, the author, publisher, and genre information were repeated across the Book, eBook, and Audiobook tables. By creating separate tables for each of these entities and linking them with foreign keys, we avoid redundancy and ensure that any update, insertion, or deletion operation can be performed consistently.

Book, eBook, and Audiobook tables: These tables hold information specific to each medium. They have been structured to contain only data directly relevant to their primary keys. This ensures that all the attributes in each table are fully functionally dependent on the primary key, satisfying 1NF and 2NF requirements.

Book Copy, eBook Copy, and Audiobook Copy tables: These tables were created to separate information about the individual copies of each medium from the general information about the medium itself. This separation helps to maintain data integrity and ensures that each table contains only data relevant to its primary key. This satisfies the 2NF requirement.

Member table: This table contains information about library members. It is structured to maintain data integrity and follows the rules of 1NF, 2NF, and 3NF, ensuring that all attributes are fully functionally dependent on the primary key.

Hold table: This table captures information about hold transactions. It includes foreign keys for member\_ID and the medium IDs (book\_ID, eBook\_ID, audiobook\_ID). By structuring the table this way, we maintain data integrity and ensure that hold transactions only reference valid members and media items.

Loan table: This table contains information about loan transactions. It has foreign keys for member\_ID and the medium copy IDs (book\_copy\_ID, eBook\_copy\_ID, audiobook\_copy\_ID). This structure ensures that each loan transaction has a non-null value for the Member ID and exactly one non-null value for the medium copy IDs, as per the constraints specified. This maintains data integrity and ensures that loan transactions only reference valid members and media items.

By dividing the schema into these 3NF tables, we have reduced data redundancy, improved data integrity, and ensured that the schema adheres to the rules of the first three normal forms. This optimized schema will lead to better performance, easier maintenance, and fewer data anomalies.

Here are the 3NF tables for the library database schema:

**Author table:**

author\_ID (Primary key)

author\_name

**Publisher table:**

publisher\_ID (Primary key)

publisher\_name

**Genre table:**

genre\_ID (Primary key)

genre\_name

**Book table:**

book\_ID (Primary key)

ISBN (UNIQUE constraint)

title

author\_ID (Foreign key)

publisher\_ID (Foreign key)

publication\_year

genre\_ID (Foreign key)

**eBook table:**

eBook\_ID (Primary key)

title

author\_ID (Foreign key)

publisher\_ID (Foreign key)

DOI (UNIQUE constraint)

publication\_year

genre\_ID (Foreign key)

**Audiobook table:**

audiobook\_ID (Primary key)

title

author\_ID (Foreign key)

publisher\_ID (Foreign key)

ISAN (UNIQUE constraint)

publication\_year

genre\_ID (Foreign key)

**Book Copy table:**

book\_copy\_ID (Primary key)

book\_ID (Foreign key, NOT NULL constraint)

status

allowable\_loan\_period

eBook Copy table:

eBook\_copy\_ID (Primary key)

eBook\_ID (Foreign key, NOT NULL constraint)

status

allowable\_loan\_period

**Audiobook Copy table:**

audiobook\_copy\_ID (Primary key)

audiobook\_ID (Foreign key, NOT NULL constraint)

status

allowable\_loan\_period

**Member table:**

member\_ID (Primary key)

first\_name

last\_name

address

contact\_details

membership\_type

membership\_status

**Hold table:**

hold\_ID (Primary key)

member\_ID (Foreign key)

book\_ID (Foreign key, nullable)

eBook\_ID (Foreign key, nullable)

audiobook\_ID (Foreign key, nullable)

hold\_date

hold\_status

**Loan table:**

loan\_ID (Primary key)

member\_ID (Foreign key, NOT NULL constraint)

hold\_ID (Foreign key, DEFAULT NULL constraint)

book\_copy\_ID (Foreign key, nullable)

eBook\_copy\_ID (Foreign key, nullable)

audiobook\_copy\_ID (Foreign key, nullable)

borrow\_date

due\_date

return\_date

**Code:(Create)**

-- 1. Author tableCREATE TABLE Author (

author\_ID INT AUTO\_INCREMENT PRIMARY KEY,

author\_name VARCHAR(255) NOT NULL

);

-- 2. Publisher tableCREATE TABLE Publisher (

publisher\_ID INT AUTO\_INCREMENT PRIMARY KEY,

publisher\_name VARCHAR(255) NOT NULL

);

-- 3. Genre tableCREATE TABLE Genre (

genre\_ID INT AUTO\_INCREMENT PRIMARY KEY,

genre\_name VARCHAR(255) NOT NULL

);

-- 4. Book tableCREATE TABLE Book (

book\_ID INT AUTO\_INCREMENT PRIMARY KEY,

ISBN VARCHAR(13) UNIQUE,

title VARCHAR(255) NOT NULL,

author\_ID INT,

publisher\_ID INT,

publication\_year YEAR,

genre\_ID INT,

FOREIGN KEY (author\_ID) REFERENCES Author(author\_ID),

FOREIGN KEY (publisher\_ID) REFERENCES Publisher(publisher\_ID),

FOREIGN KEY (genre\_ID) REFERENCES Genre(genre\_ID)

);

-- 5. eBook tableCREATE TABLE eBook (

eBook\_ID INT AUTO\_INCREMENT PRIMARY KEY,

title VARCHAR(255) NOT NULL,

author\_ID INT,

publisher\_ID INT,

DOI VARCHAR(255) UNIQUE,

publication\_year YEAR,

genre\_ID INT,

FOREIGN KEY (author\_ID) REFERENCES Author(author\_ID),

FOREIGN KEY (publisher\_ID) REFERENCES Publisher(publisher\_ID),

FOREIGN KEY (genre\_ID) REFERENCES Genre(genre\_ID)

);

-- 6. Audiobook tableCREATE TABLE Audiobook (

audiobook\_ID INT AUTO\_INCREMENT PRIMARY KEY,

title VARCHAR(255) NOT NULL,

author\_ID INT,

publisher\_ID INT,

ISAN VARCHAR(255) UNIQUE,

publication\_year YEAR,

genre\_ID INT,

FOREIGN KEY (author\_ID) REFERENCES Author(author\_ID),

FOREIGN KEY (publisher\_ID) REFERENCES Publisher(publisher\_ID),

FOREIGN KEY (genre\_ID) REFERENCES Genre(genre\_ID)

);

-- 7. Book Copy tableCREATE TABLE Book\_Copy (

book\_copy\_ID INT AUTO\_INCREMENT PRIMARY KEY,

book\_ID INT NOT NULL,

status VARCHAR(50),

allowable\_loan\_period INT,

FOREIGN KEY (book\_ID) REFERENCES Book(book\_ID)

);

-- 8. eBook Copy tableCREATE TABLE eBook\_Copy (

eBook\_copy\_ID INT AUTO\_INCREMENT PRIMARY KEY,

eBook\_ID INT NOT NULL,

status VARCHAR(50),

allowable\_loan\_period INT,

FOREIGN KEY (eBook\_ID) REFERENCES eBook(eBook\_ID)

);

-- 9. Audiobook Copy tableCREATE TABLE Audiobook\_Copy (

audiobook\_copy\_ID INT AUTO\_INCREMENT PRIMARY KEY,

audiobook\_ID INT NOT NULL,

status VARCHAR(50),

allowable\_loan\_period INT,

FOREIGN KEY (audiobook\_ID) REFERENCES Audiobook(audiobook\_ID)

);

-- 10. Member tableCREATE TABLE Member (

member\_ID INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(255) NOT NULL,

last\_name VARCHAR(255) NOT NULL,

address VARCHAR(255),

contact\_details VARCHAR(255),

membership\_type VARCHAR(50),

membership\_status VARCHAR(50)

);

-- 11. Hold tableCREATE TABLE Hold (

hold\_ID INT AUTO\_INCREMENT PRIMARY KEY,

member\_ID INT NOT NULL,

book\_ID INT,

eBook\_ID INT,

audiobook\_ID INT,

hold\_date DATE,

hold\_status VARCHAR(50),

FOREIGN KEY (member\_ID) REFERENCES Member(member\_ID),

FOREIGN KEY (book\_ID) REFERENCES Book(book\_ID),

FOREIGN KEY (eBook\_ID) REFERENCES eBook(eBook\_ID),

FOREIGN KEY (audiobook\_ID) REFERENCES Audiobook(audiobook\_ID)

);

-- 12. Loan tableCREATE TABLE Loan (

loan\_ID INT AUTO\_INCREMENT PRIMARY KEY,

member\_ID INT NOT NULL,

hold\_id INT DEFAULT NULL,

book\_copy\_ID INT,

eBook\_copy\_ID INT,

audiobook\_copy\_ID INT,

borrow\_date DATE,

due\_date DATE,

return\_date DATE,

FOREIGN KEY (member\_ID) REFERENCES Member(member\_ID),

FOREIGN KEY (hold\_id) REFERENCES Hold(hold\_id),

FOREIGN KEY (book\_copy\_ID) REFERENCES Book\_Copy(book\_copy\_ID),

FOREIGN KEY (eBook\_copy\_ID) REFERENCES eBook\_Copy(eBook\_copy\_ID),

FOREIGN KEY (audiobook\_copy\_ID) REFERENCES Audiobook\_Copy(audiobook\_copy\_ID)

);

**Code:(Insert)**

-- Insert authorsINSERT INTO Author (author\_name)VALUES ('George Orwell'),

('J.R.R. Tolkien'),

('Harper Lee');

-- Insert publishersINSERT INTO Publisher (publisher\_name)VALUES ('Secker & Warburg'),

('Allen & Unwin'),

('J.B. Lippincott & Co.');

-- Insert genresINSERT INTO Genre (genre\_name)VALUES ('Dystopian Fiction'),

('Fantasy'),

('Southern Gothic');

-- Insert booksINSERT INTO Book (ISBN, title, author\_ID, publisher\_ID, publication\_year, genre\_ID)VALUES ('978-0451524935', '1984', 1, 1, '1949', 1),

('978-0261102385', 'The Lord of the Rings', 2, 2, '1954', 2),

('978-0060935467', 'To Kill a Mockingbird', 3, 3, '1960', 3);

This code inserts three authors, three publishers, and three genres into their respective tables. Then, it adds three books to the Book table, linking them to the appropriate authors, publishers, and genres using their IDs. Note that you should adjust the author\_ID, publisher\_ID, and genre\_ID values according to the actual IDs generated by the AUTO\_INCREMENT primary keys in your specific database.

-- Insert book copiesINSERT INTO Book\_Copy (book\_ID, status, allowable\_loan\_period)VALUES (1, 'Available', 14),

(1, 'Available', 14),

(2, 'Available', 14),

(2, 'On Loan', 14),

(3, 'Available', 14),

(3, 'Available', 14),

(3, 'On Hold', 14);

-- Insert membersINSERT INTO Member (first\_name, last\_name, address, contact\_details, membership\_type, membership\_status)VALUES ('Alice', 'Johnson', '123 Main St', 'alice@email.com', 'Standard', 'Active'),

('Bob', 'Smith', '456 Oak St', 'bob@email.com', 'Standard', 'Active'),

('Carol', 'Williams', '789 Pine St', 'carol@email.com', 'Premium', 'Active');

This code inserts seven book copies into the Book\_Copy table, with some copies marked as 'Available', one 'On Loan', and one 'On Hold'. Then, it adds three members to the Member table.

Now, let's create some hold and loan transactions:

-- Insert holdsINSERT INTO Hold (member\_ID, book\_ID, hold\_date, hold\_status)VALUES (1, 3, '2023-04-15', 'Active'),

(2, 2, '2023-04-18', 'Active');

-- Insert loansINSERT INTO Loan (member\_ID, book\_copy\_ID, borrow\_date, due\_date)VALUES (3, 4, '2023-04-10', '2023-04-24');

This code creates two hold transactions for members Alice and Bob, and one loan transaction for member Carol.