Assignment 2: Design a database schema for a library system, including tables, fields, and constraints like NOT NULL, UNIQUE, and CHECK. Include primary and foreign keys to establish relationships between tables.

Database Schema for Library System

1. Tables and Fields

1) Books

- book_id (INT, PRIMARY KEY, AUTO_INCREMENT)
- title (VARCHAR(255), NOT NULL)
- author (VARCHAR(255), NOT NULL)
- isbn (VARCHAR(13), NOT NULL, UNIQUE)
- published_year (YEAR, NOT NULL)
- genre (VARCHAR(100))
- available_copies (INT, NOT NULL, CHECK (available_copies >= 0))

2) Members

- member_id (INT, PRIMARY KEY, AUTO_INCREMENT)
- first_name (VARCHAR(100), NOT NULL)
- last_name (VARCHAR(100), NOT NULL)
- email (VARCHAR(255), NOT NULL, UNIQUE)
- phone (VARCHAR(15))
- address (VARCHAR(255))
- join_date (DATE, NOT NULL)

3) Loans

- loan_id (INT, PRIMARY KEY, AUTO_INCREMENT)
- book_id (INT, NOT NULL, FOREIGN KEY REFERENCES Books(book_id))
- member_id (INT, NOT NULL, FOREIGN KEY REFERENCES Members(member_id))
- loan_date (DATE, NOT NULL)
- due_date (DATE, NOT NULL)
- return_date (DATE)

4) Authors

- author_id (INT, PRIMARY KEY, AUTO_INCREMENT)
- name (VARCHAR(255), NOT NULL, UNIQUE)
- birthdate (DATE)

5) Genres

- genre_id (INT, PRIMARY KEY, AUTO_INCREMENT)
- genre_name (VARCHAR(100), NOT NULL, UNIQUE)

6) BookGenres

- book_id (INT, NOT NULL, FOREIGN KEY REFERENCES Books(book_id))
- genre_id (INT, NOT NULL, FOREIGN KEY REFERENCES Genres(genre_id))
- PRIMARY KEY (book_id, genre_id)

SQL STATEMENT

```
-- Create Books table
CREATE TABLE Books (
   book_id INT PRIMARY KEY AUTO_INCREMENT,
   title VARCHAR(255) NOT NULL,
   author VARCHAR(255) NOT NULL,
   isbn VARCHAR(13) NOT NULL UNIQUE,
   published_year YEAR NOT NULL,
   genre VARCHAR(100),
   available_copies INT NOT NULL CHECK (available_copies >= 0)
);

-- Create Members table
CREATE TABLE Members (
   member_id INT PRIMARY KEY AUTO_INCREMENT,
   first_name VARCHAR(100) NOT NULL,
   last_name VARCHAR(100) NOT NULL,
   email VARCHAR(255) NOT NULL UNIQUE,
   phone VARCHAR(15),
   address VARCHAR(255),
   join_date DATE NOT NULL
);

-- Create Loans table
CREATE TABLE Loans (
   loan_id INT PRIMARY KEY AUTO_INCREMENT,
   book_id INT NOT NULL,
   member_id INT NOT NULL,
   loan_date DATE NOT NULL,
   due_date DATE NOT NULL,
   return_date DATE,
   FOREIGN KEY (book_id) REFERENCES Books(book_id),
   FOREIGN KEY (member_id) REFERENCES Members(member_id)
);

-- Create Authors table
CREATE TABLE Authors (
   author_id INT PRIMARY KEY AUTO_INCREMENT,
   name VARCHAR(255) NOT NULL UNIQUE,
   birthdate DATE
);

-- Create Genres table
CREATE TABLE Genres (
   genre_id INT PRIMARY KEY AUTO_INCREMENT,
   genre_name VARCHAR(100) NOT NULL UNIQUE
);
```

```
-- Create BookGenres table
CREATE TABLE BookGenres (
    book_id INT NOT NULL,
    genre_id INT NOT NULL,
    PRIMARY KEY (book_id, genre_id),
    FOREIGN KEY (book_id) REFERENCES Books(book_id),
    FOREIGN KEY (genre_id) REFERENCES Genres(genre_id)
);
```

2. **Explanation of Constraints:**
   - NOT NULL: Ensures that the field cannot be left empty.
   - UNIQUE: Ensures that all values in a column are distinct.
   - CHECK: Ensures that all values in a column satisfy a specific condition.
   - PRIMARY KEY: Uniquely identifies each row/record in a table.
   - FOREIGN KEY: Ensures referential integrity by linking one table to another.

3. **Relationships:**
   - Books to Loans: One-to-Many (One book can have many loans).
   - Members to Loans: One-to-Many (One member can have many loans).
   - Books to Genres: Many-to-Many (A book can belong to multiple genres, and a genre can include multiple books) via BookGenres.