NOVEL NEST

A Simple Database Application of Library Management System

CSE 2211 : Database Management System Lab

Batch : 29 / 2nd Year 2nd Semester 2024

Date of Submission

17 July, 2025

Submitted by

Ariful Islam (Roll-45)
Jubair Ahammad Akter (Roll-59)

Course Instructors

Mr. Abu Ahmed Ferdous (Associate Professor)
Mr. Redwan Ahmed Rizvee (Lecturer)

Department of Computer Science and Engineering

University of Dhaka

Table of Contents

1 Introduction

1.1 Project Overview

This project, **NOVEL NEST**, implements a comprehensive **Library Management System** database that stores and manages data about books, authors, users, transactions, and other relevant information. The system is designed to help users manage books in a library, track borrowing transactions, and maintain records of all library activities efficiently.

1.2 Objectives

The main objectives of this database system are :

- To provide a centralized repository for library information
- To enable efficient searching and filtering of books based on multiple criteria
- To maintain relationships between books, authors, users, and transactions
- To support complex queries that can help users find books matching their preferences
- To ensure data integrity through proper constraints and normalization

1.3 Scope

The database covers the following aspects of library management :

- Basic book details (title, description, genre, price)
- Information about authors (name, biography, nationality)
- User details (name, contact, library card information)
- Transaction records (borrow, return dates, penalties)
- Book availability and inventory status
- Tags for categorization and searching

2 Project Features

- User registration and login
- Admin panel for managing books and users
- Book issue and return management
- Viewing available books and user information
- Secure authentication for users and admins

3 Database Design

3.1 Entity-Relationship Model

The database follows a relational model with multiple entities and relationships. The main entities include :

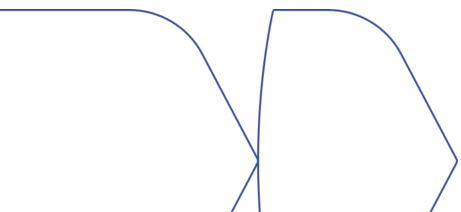
Entities and Attributes

- **Users**
  - Attributes : id (PK), name, username (unique), photo, email (unique), phone, address, password, role, unique_id (unique)
- **Authors**
  - Attributes : id (PK), name, bio, photo, email (unique), nbooks, nnews
- **Books**
  - Attributes : id (PK), title, type, author_id (FK), cover, description, price, amount, location, branch_id (FK), pdf
- **Attendance**
  - Attributes : id (PK), user_id (FK), branch_id (FK), entry_time, exit_time, books_id (FK)
- **Lend**
  - Attributes : id (PK), user_id (FK), branch_id (FK), received_time, return_time, books_id (FK), method
- **Buy**
  - Attributes : id (PK), user_id (FK), branch_id (FK), time, books_id (FK), method, amount, payment_id
- **Branches**
  - Attributes : id (PK), name, address, email (unique), phone
- **News**
  - Attributes : id (PK), title, content, author_id (FK), date

Relationships

- **Authorship** (between Authors and Books)
  - A book can have multiple authors, and an author can write multiple books. This is represented by the `Book_Authors` junction table.
- **News Article Authorship** (between Authors and News)

  - A news article can have multiple authors, and an author can write multiple news articles. This is represented by the `News_Authors` junction table.
- **Attendance Record** (between Users and Attendance)
  - A user can have multiple attendance records. An attendance record is associated with one user.
- **Book Borrowing** (between Books and Attendance)
  - A book can be checked out by multiple users. Each attendance record can refer to multiple books.
- **Book Lending** (between Users, Branches, and Lend)
  - One user can borrow many books from many branches. A lending record is linked to one user and one branch.
- **Book Buying** (between Users, Branches, and Buy)
  - One user can buy books from one or multiple branches. A buying record is linked to one user and one branch.
- **News Article** (between Authors and News)
  - A news article can have multiple authors. A news article is written by one or more authors.



3.2 Schema Diagram

Figure 1 shows the schema diagram of our database with all tables and their relationships.

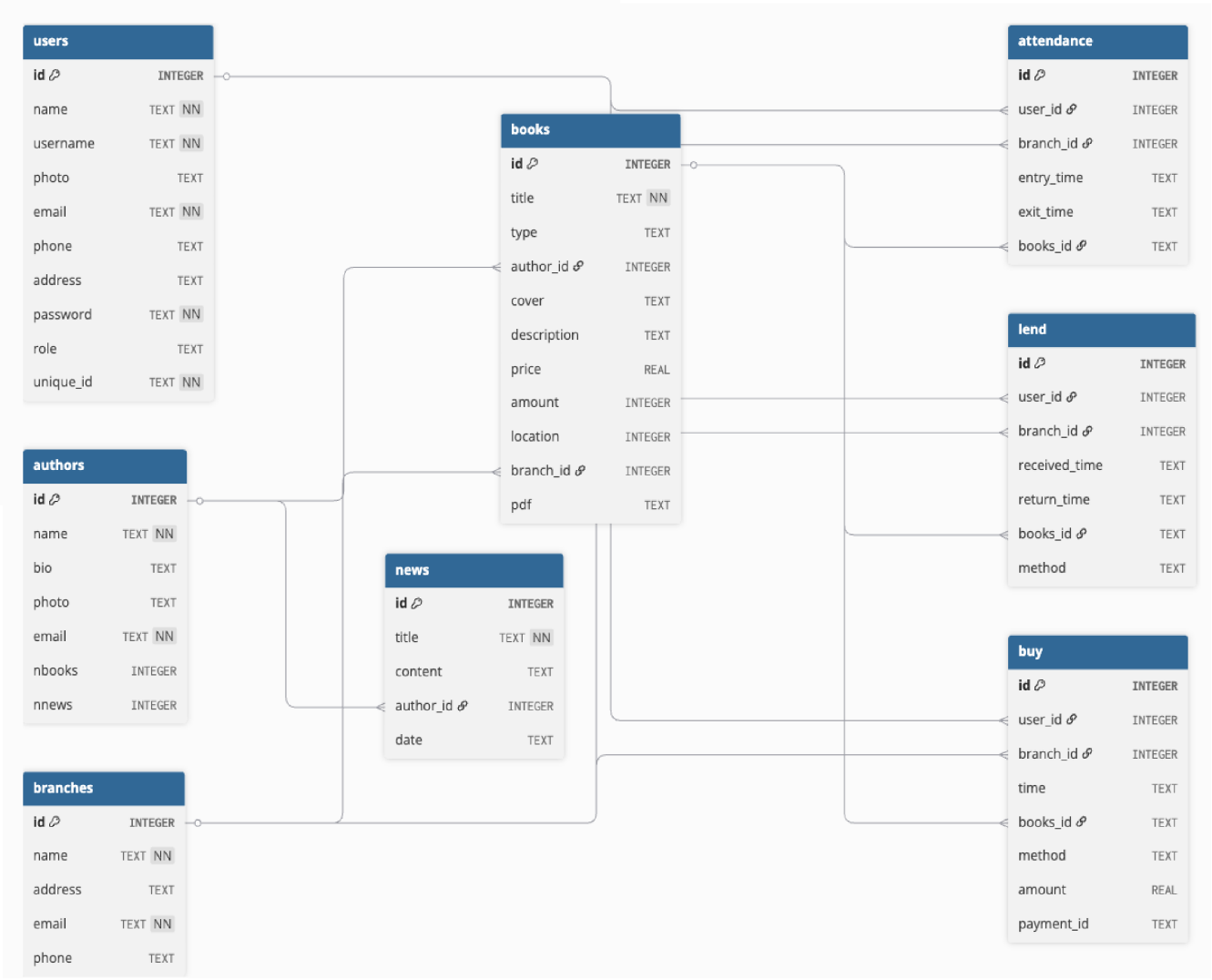


FIGURE 1 – Database Schema Diagram

3.3 Entity-Relationship Diagram

The E-R diagram in Figure 3 illustrates the relationships between different entities in our database system.

article tikz

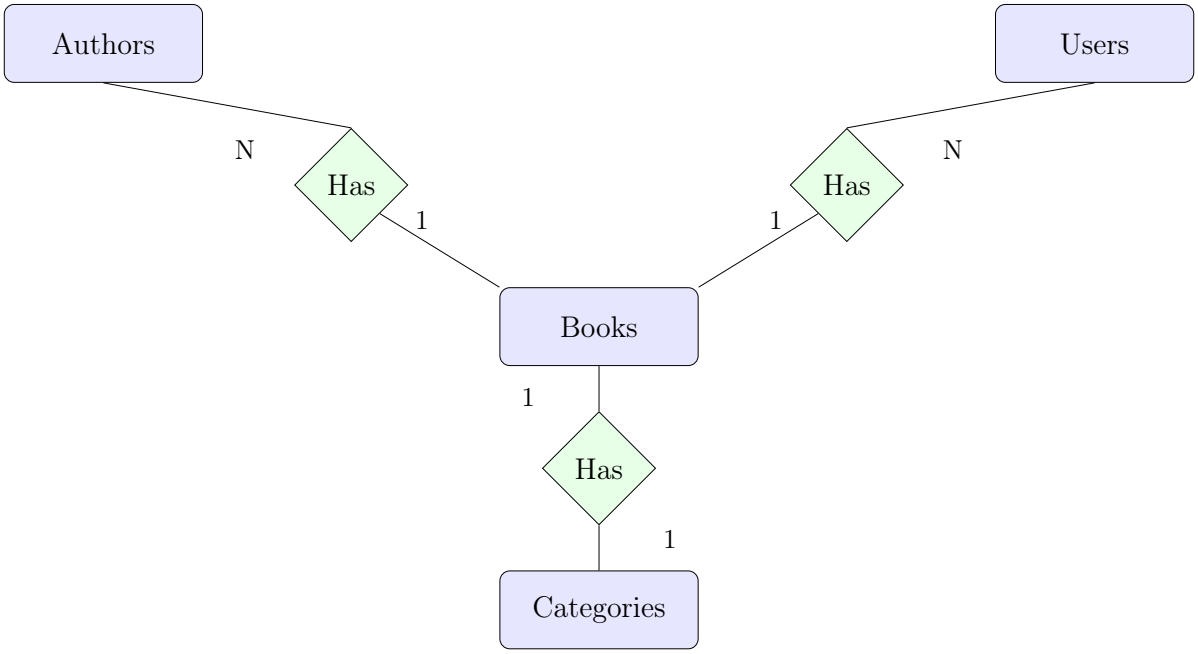


FIGURE 2 – Entity-Relationship Diagram of the Library Management System

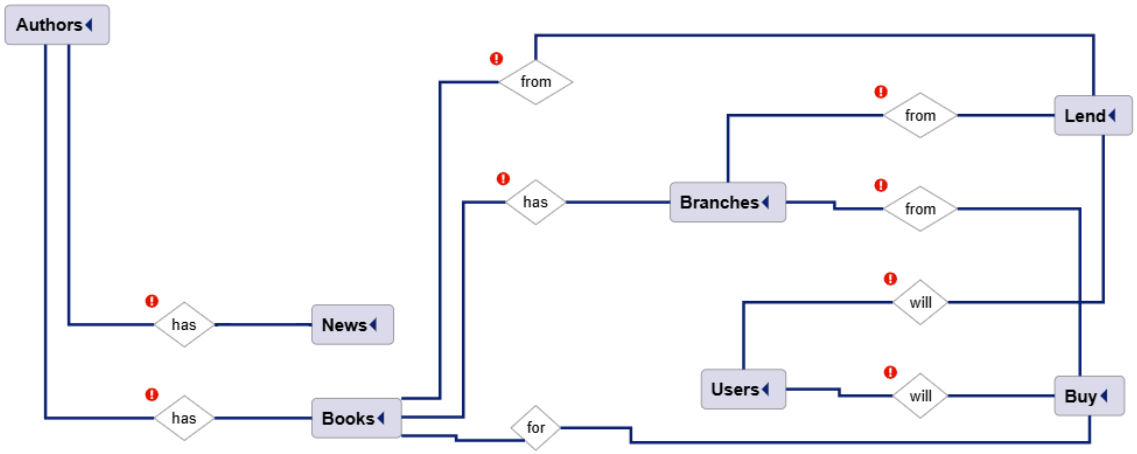


FIGURE 3 – Entity-Relationship Diagram of the Library Management System...2

4 Database Implementation

4.1 Tables and Attributes

The database consists of the following tables with their respective attributes :

4.1.1 Users Table

This table stores information about users.

Attribute	Description	Data Type	Constraints
id	Unique identifier for each user	INTEGER	PRIMARY KEY
name	User's full name	TEXT	NOT NULL
username	User's unique username	TEXT	UNIQUE, NOT NULL
photo	URL of the user's profile photo	TEXT	
email	User's email address	TEXT	UNIQUE, NOT NULL, CHECK (email LIKE '%@%.%')
phone	User's phone number	TEXT	CHECK (LENGTH(phone) > 10)
address	User's address	TEXT	
password	User's password	TEXT	NOT NULL
role	Role of the user (e.g., 'admin', 'user')	TEXT	DEFAULT 'user'
unique_id	Unique identifier for each user	TEXT	UNIQUE, NOT NULL

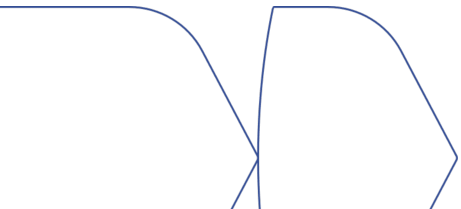
Table 1 – Users Table Structure

4.1.2 Authors Table

This table stores information about authors.

Attribute	Description	Data Type	Constraints
id	Unique identifier for each author	INTEGER	PRIMARY KEY, AUTOINCREMENT
name	Author's full name	TEXT	NOT NULL
bio	Author's biography	TEXT	
photo	URL of the author's photo	TEXT	
email	Author's email address	TEXT	UNIQUE, NOT NULL, CHECK (email LIKE '%@%%')
nbooks	Number of books written by the author	INTEGER	CHECK (nbooks >= 0)
nnews	Number of news articles written by the author	INTEGER	CHECK (nnews >= 0)

Table 2 – Authors Table Structure



4.1.3 Books Table

This table stores information about books.

Attribute	Description	Data Type	Constraints
id	Unique identifier for each book	INTEGER	PRIMARY KEY, AUTOINCREMENT
title	Title of the book	TEXT	NOT NULL
type	Type of the book (e.g., fiction, non-fiction)	TEXT	
author_id	Reference to the author of the book	INTEGER	FOREIGN KEY (author_id) REFERENCES authors(id)
cover	URL of the book's cover image	TEXT	
description	Description of the book	TEXT	
price	Price of the book	REAL	CHECK (price >= 0)
amount	Available stock of the book	INTEGER	CHECK (amount >= 0)
location	Location identifier of the book	INTEGER	
branch_id	Reference to the branch where the book is available	INTEGER	FOREIGN KEY (branch_id) REFERENCES branches(id)
pdf	Availability of the PDF version of the book (yes/no)	TEXT	CHECK (pdf IN ('yes', 'no'))

Table 3 – Books Table Structure

4.1.4 Attendance Table

This table stores information about user attendance.

Attribute	Description	Data Type	Constraints
id	Unique identifier for each attendance record	INTEGER	PRIMARY KEY, AUTOINCREMENT
user_id	Reference to the user who attended	INTEGER	FOREIGN KEY (user_id) REFERENCES users(id)
branch_id	Reference to the branch where the attendance was recorded	INTEGER	FOREIGN KEY (branch_id) REFERENCES branches(id)
entry_time	Time when the user entered the branch	TEXT	
exit_time	Time when the user left the branch	TEXT	
books_id	Reference to the books borrowed during attendance	TEXT	FOREIGN KEY (books_id) REFERENCES books(id)

Table 4 – Attendance Table Structure

4.1.5 Lend Table

This table stores information about books lent to users.

Attribute	Description	Data Type	Constraints
id	Unique identifier for each lending record	INTEGER	PRIMARY KEY, AUTOINCREMENT
user_id	Reference to the user who lent the book	INTEGER	FOREIGN KEY (user_id) REFERENCES users(id)
branch_id	Reference to the branch where the book was lent	INTEGER	FOREIGN KEY (branch_id) REFERENCES branches(id)
received_time	Time when the book was lent to the user	TEXT	
return_time	Time when the book was expected to be returned	TEXT	
books_id	Reference to the book lent	TEXT	FOREIGN KEY (books_id) REFERENCES books(id)
method	Payment method for the lending	TEXT	

Table 5 – Lend Table Structure

4.1.6 Buy Table

This table stores information about books purchased by users.

Attribute	Description	Data Type	Constraints
id	Unique identifier for each purchase record	INTEGER	PRIMARY KEY, AUTOINCREMENT
user_id	Reference to the user who made the purchase	INTEGER	FOREIGN KEY (user_id) REFERENCES users(id)
branch_id	Reference to the branch where the purchase was made	INTEGER	FOREIGN KEY (branch_id) REFERENCES branches(id)
time	Time of purchase	TEXT	
books_id	Reference to the book purchased	TEXT	FOREIGN KEY (books_id) REFERENCES books(id)
method	Payment method for the purchase	TEXT	
amount	Amount paid for the book	REAL	CHECK (amount >= 0)
payment_id	Identifier for the payment transaction	TEXT	

Table 6 – Buy Table Structure

4.1.7 Branches Table

This table stores information about the library branches.

Attribute	Description	Data Type	Constraints
id	Unique identifier for each branch	INTEGER	PRIMARY KEY, AUTOINCREMENT
name	Name of the branch	TEXT	NOT NULL
address	Address of the branch	TEXT	
email	Branch's email address	TEXT	UNIQUE, NOT NULL, CHECK (email LIKE '%@%.%')
phone	Branch's phone number	TEXT	

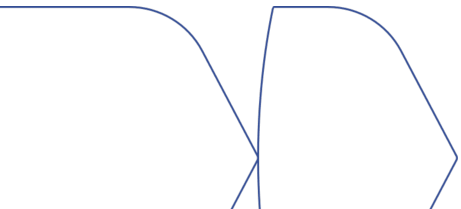
TABLE 7 – Branches Table Structure

4.1.8 News Table

This table stores information about news articles.

Attribute	Description	Data Type	Constraints
id	Unique identifier for each news article	INTEGER	PRIMARY KEY, AUTOINCREMENT
title	Title of the news article	TEXT	NOT NULL
content	Content of the news article	TEXT	
author_id	Reference to the author of the news article	INTEGER	FOREIGN KEY (author_id) REFERENCES authors(id)
date	Date of publication	TEXT	

TABLE 8 – News Table Structure



4.2 SQL DDL Statements

Below are the SQL DDL (Data Definition Language) statements used to create the database tables :

4.2.1 Users Table

```sql
CREATE TABLE IF NOT EXISTS users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    username TEXT UNIQUE NOT NULL,
    photo TEXT,
    email TEXT UNIQUE NOT NULL CHECK (email LIKE '%@%.%'),
    phone TEXT CHECK (LENGTH(phone) > 10),
    address TEXT,
    password TEXT NOT NULL,
    role TEXT DEFAULT 'user' CHECK (role IN ('user', 'admin')),
    unique_id TEXT UNIQUE NOT NULL
);
```

4.2.2 Authors Table

```sql
CREATE TABLE IF NOT EXISTS authors (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    bio TEXT,
    photo TEXT,
    email TEXT UNIQUE NOT NULL CHECK (email LIKE '%@%.%'),
    nbooks INTEGER CHECK (nbooks >= 0),
    nnews INTEGER CHECK (nnews >= 0)
);
```

4.2.3 Books Table

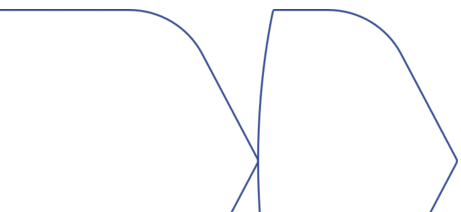
```sql
CREATE TABLE IF NOT EXISTS books (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    title TEXT NOT NULL,
    type TEXT,
    author_id INTEGER,
    cover TEXT,
    description TEXT,
    price REAL CHECK (price >= 0),
    amount INTEGER CHECK (amount >= 0),
    location INTEGER,
    branch_id INTEGER,
    pdf TEXT CHECK (pdf IN ('yes', 'no')),
    FOREIGN KEY(author_id) REFERENCES authors(id),
    FOREIGN KEY(branch_id) REFERENCES branches(id)
);
```

4.2.4 Attendance Table

```sql
CREATE TABLE IF NOT EXISTS attendance (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    user_id INTEGER,
    branch_id INTEGER,
    entry_time TEXT,
    exit_time TEXT,
    books_id TEXT,
    FOREIGN KEY(user_id) REFERENCES users(id),
    FOREIGN KEY(branch_id) REFERENCES branches(id),
    FOREIGN KEY(books_id) REFERENCES books(id)
);
```

4.2.5 Lend Table

```sql
CREATE TABLE IF NOT EXISTS lend (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    user_id INTEGER,
    branch_id INTEGER,
    received_time TEXT,
    return_time TEXT,
    books_id TEXT,
    method TEXT,
    FOREIGN KEY(user_id) REFERENCES users(id),
    FOREIGN KEY(branch_id) REFERENCES branches(id),
```



```
    FOREIGN KEY(books_id) REFERENCES books(id)
);
```

4.2.6 Buy Table

```
CREATE TABLE IF NOT EXISTS buy (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    user_id INTEGER,
    branch_id INTEGER,
    time TEXT,
    books_id TEXT,
    method TEXT,
    amount REAL CHECK (amount >= 0),
    payment_id TEXT,
    FOREIGN KEY(user_id) REFERENCES users(id),
    FOREIGN KEY(branch_id) REFERENCES branches(id),
    FOREIGN KEY(books_id) REFERENCES books(id)
);
```

4.2.7 Branches Table

```
CREATE TABLE IF NOT EXISTS branches (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    address TEXT,
    email TEXT UNIQUE NOT NULL CHECK (email LIKE '%@%.%'),
    phone TEXT
);
```

4.2.8 News Table

```
CREATE TABLE IF NOT EXISTS news (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    title TEXT NOT NULL,
    content TEXT,
    author_id INTEGER,
    date TEXT,
    FOREIGN KEY(author_id) REFERENCES authors(id)
);
```

4.3 Sample Data

Below are examples of data inserted into the tables :

4.3.1 Users Table

```
INSERT INTO users (name, username, photo, email, phone, address,
    password, role, unique_id) VALUES
('John Doe', 'john_doe', 'photo1.jpg', 'john@example.com',
    '12345678901', '123 Elm Street', 'password123', 'user',
    'UID001'),
('Alice Smith', 'alice_smith', 'photo2.jpg',
    'alice@example.com', '12345678902', '456 Oak Street',
    'password456', 'admin', 'UID002'),
('Bob Johnson', 'bob_johnson', 'photo3.jpg', 'bob@example.com',
    '12345678903', '789 Pine Street', 'password789', 'user',
    'UID003'),
('Admin User', 'admin_user', 'photo4.jpg', 'admin@example.com',
    '12345678904', '101 Maple Street', 'adminpassword', 'admin',
    'UID004'),
('Eve Williams', 'eve_williams', 'photo5.jpg',
    'eve@example.com', '12345678905', '202 Birch Lane',
    'password101', 'user', 'UID005'),
('David Brown', 'david_brown', 'photo6.jpg',
    'david@example.com', '12345678906', '303 Cedar Street',
    'password202', 'admin', 'UID006'),
('Grace Lee', 'grace_lee', 'photo7.jpg', 'grace@example.com',
    '12345678907', '404 Maple Street', 'password303', 'user',
    'UID007'),
('Michael Davis', 'michael_davis', 'photo8.jpg',
    'michael@example.com', '12345678908', '505 Oak Avenue',
    'password404', 'admin', 'UID008'),
('Sarah White', 'sarah_white', 'photo9.jpg',
    'sarah@example.com', '12345678909', '606 Pine Road',
    'password505', 'user', 'UID009'),
('Daniel Green', 'daniel_green', 'photo10.jpg',
    'daniel@example.com', '12345678910', '707 Elm Street',
    'password606', 'user', 'UID010'),
('Olivia Clark', 'olivia_clark', 'photo11.jpg',
    'olivia@example.com', '12345678911', '808 Birch Lane',
    'password707', 'admin', 'UID011'),
('James Martin', 'james_martin', 'photo12.jpg',
    'james@example.com', '12345678912', '909 Cedar Street',
    'password808', 'user', 'UID012'),
('Lily Hall', 'lily_hall', 'photo13.jpg', 'lily@example.com',
    '12345678913', '1010 Oak Street', 'password909', 'user',
    'UID013'),
('Lucas Allen', 'lucas_allen', 'photo14.jpg',
    'lucas@example.com', '12345678914', '1111 Pine Road',
    'password1010', 'admin', 'UID014'),
```

```
('Megan King', 'megan_king', 'photo15.jpg', 'megan@example.com',
    '12345678915', '1212 Birch Lane', 'password1111', 'user',
    'UID015');
```

4.3.2 Authors Table

```
INSERT INTO authors (name, bio, photo, email, nbooks, nnews)
    VALUES
('J.K. Rowling', 'British author, best known for the Harry
    Potter series.', 'jk_rowling.jpg', 'jk@example.com', 10, 5),
('George Orwell', 'English novelist and essayist, known for 1984
    and Animal Farm.', 'george_orwell.jpg', 'george@example.com',
    6, 2),
('Mark Twain', 'American writer, humorist, and publisher.',
    'mark_twain.jpg', 'mark@example.com', 12, 3),
('Jane Austen', 'English novelist known for Pride and
    Prejudice.', 'jane_austen.jpg', 'jane@example.com', 7, 1),
('Ernest Hemingway', 'American novelist and short story
    writer.', 'ernest_hemingway.jpg', 'ernest@example.com', 8, 2),
('Haruki Murakami', 'Japanese novelist and translator.',
    'haruki_murakami.jpg', 'haruki@example.com', 15, 6),
('F. Scott Fitzgerald', 'American novelist, best known for The
    Great Gatsby.', 'f_scott_fitzgerald.jpg',
    'fitzgerald@example.com', 5, 4),
('Leo Tolstoy', 'Russian writer, best known for War and Peace.',
    'leo_tolstoy.jpg', 'leo@example.com', 20, 10),
('Charles Dickens', 'English writer and social critic.',
    'charles_dickens.jpg', 'charles@example.com', 30, 15),
('Agatha Christie', 'English crime writer, known for Hercule
    Poirot.', 'agatha_christie.jpg', 'agatha@example.com', 10, 5),
('Isaac Asimov', 'American author and professor of
    biochemistry.', 'isaac_asimov.jpg', 'isaac@example.com', 25,
    8),
('J.R.R. Tolkien', 'English writer and philologist, best known
    for The Lord of the Rings.', 'tolkien.jpg',
    'tolkien@example.com', 5, 2),
('C.S. Lewis', 'British writer and theologian.', 'cs_lewis.jpg',
    'cs_lewis@example.com', 8, 4),
('Virginia Woolf', 'English writer, known for Mrs Dalloway.',
    'virginia_woolf.jpg', 'virginia@example.com', 10, 5),
('Gabriel Garcia Marquez', 'Colombian novelist, known for One
    Hundred Years of Solitude.', 'garcia_marquez.jpg',
    'marquez@example.com', 15, 7);
```

4.3.3 Books Table

```
INSERT INTO books (title, type, author_id, cover, description,
    price, amount, location, branch_id, pdf) VALUES
('Harry Potter and the Philosopher\'s Stone', 'Fantasy', 1,
    'hp1_cover.jpg', 'First book in the Harry Potter series.',
    19.99, 10, 1, 1, 'yes'),
('1984', 'Dystopian', 2, '1984_cover.jpg', 'A novel about a
    totalitarian regime.', 14.99, 8, 2, 2, 'no'),
('The Adventures of Tom Sawyer', 'Adventure', 3,
    'tom_sawyer_cover.jpg', 'A story of a young boy growing up
    along the Mississippi River.', 12.99, 5, 3, 3, 'no'),
('Pride and Prejudice', 'Romance', 4,
    'pride_prejudice_cover.jpg', 'A love story set in the Regency
    Era.', 9.99, 6, 4, 4, 'yes'),
('The Old Man and the Sea', 'Fiction', 5,
    'old_man_sea_cover.jpg', 'A short novel about a fisherman\'s
    struggle.', 11.99, 7, 5, 5, 'no'),
('Harry Potter and the Chamber of Secrets', 'Fantasy', 1,
    'hp2_cover.jpg', 'Second book in the Harry Potter series.',
    19.99, 9, 1, 1, 'yes'),
('Animal Farm', 'Allegory', 2, 'animal_farm_cover.jpg', 'A
    political satire about a group of farm animals.', 10.99, 4,
    2, 2, 'no'),
('The Catcher in the Rye', 'Fiction', 3,
    'catcher_rye_cover.jpg', 'A young man\'s struggles in New
    York.', 13.99, 3, 3, 3, 'no'),
('Emma', 'Romance', 4, 'emma_cover.jpg', 'A story of matchmaking
    and misjudgments.', 8.99, 5, 4, 4, 'yes'),
('A Farewell to Arms', 'War Fiction', 5,
    'farewell_arms_cover.jpg', 'A love story set during World War
    I.', 12.99, 6, 5, 5, 'no'),
('Harry Potter and the Prisoner of Azkaban', 'Fantasy', 1,
    'hp3_cover.jpg', 'The third book in the Harry Potter
    series.', 18.99, 8, 1, 1, 'yes'),
('The Road to Wigan Pier', 'Non-Fiction', 2,
    'wigan_pier_cover.jpg', 'A reflection on life in
    working-class Britain.', 15.99, 5, 2, 2, 'no'),
('Huckleberry Finn', 'Adventure', 3, 'huck_finn_cover.jpg', 'The
    adventures of Huck Finn and Jim down the Mississippi River.',
    14.99, 6, 3, 3, 'no'),
('Sense and Sensibility', 'Romance', 4,
    'sense_sensibility_cover.jpg', 'A story of two sisters
    finding love.', 7.99, 8, 4, 4, 'yes'),
('The Sun Also Rises', 'Fiction', 5, 'sun_also_rises_cover.jpg',
    'A novel about post-war disillusionment.', 13.99, 7, 5, 5,
    'no'),
('Moby Dick', 'Adventure', 3, 'moby_dick_cover.jpg', 'A novel
    about a captains obsessive quest for revenge on a white
    whale.', 16.99, 4, 6, 1, 'no'),
```

```
('The Great Gatsby', 'Classic', 6, 'gatsby_cover.jpg', 'A novel about the American Dream in the 1920s.', 10.99, 7, 2, 2, 'yes'),
('To Kill a Mockingbird', 'Drama', 7, 'mockingbird_cover.jpg', 'A story of racial injustice in the deep South.', 13.49, 8, 3, 3, 'no'),
('The Odyssey', 'Epic', 8, 'odyssey_cover.jpg', 'An ancient Greek epic about Odysseus\' journey home.', 18.49, 6, 4, 4, 'yes'),
('The Hobbit', 'Fantasy', 9, 'hobbit_cover.jpg', 'The prequel to The Lord of the Rings, a story of Bilbo Baggins.', 15.99, 10, 5, 5, 'no'),
('Les Miserables', 'Historical Fiction', 10, 'les_miserables_cover.jpg', 'A story of justice, love, and redemption in 19th-century France.', 19.49, 9, 1, 1, 'yes'),
('Brave New World', 'Dystopian', 2, 'brave_new_world_cover.jpg', 'A utopian society that faces tragic consequences.', 14.99, 11, 2, 2, 'no'),
('Fahrenheit 451', 'Dystopian', 2, 'fahrenheit_451_cover.jpg', 'A society where books are burned to maintain control.', 11.99, 12, 3, 3, 'no'),
('Dracula', 'Horror', 11, 'dracula_cover.jpg', 'The classic vampire novel by Bram Stoker.', 13.99, 9, 4, 4, 'yes'),
('Wuthering Heights', 'Romance', 12, 'wuthering_heights_cover.jpg', 'A tragic love story set in the Yorkshire moors.', 16.99, 6, 5, 5, 'no'),
('Frankenstein', 'Gothic', 13, 'frankenstein_cover.jpg', 'The tale of a scientist who creates a monstrous being.', 14.49, 7, 1, 1, 'yes'),
('The Picture of Dorian Gray', 'Gothic', 14, 'dorian_gray_cover.jpg', 'A man remains young while his portrait ages and bears the consequences of his actions.', 17.49, 8, 2, 2, 'no'),
('The Bell Jar', 'Fiction', 15, 'bell_jar_cover.jpg', 'A semi-autobiographical novel by Sylvia Plath.', 12.99, 7, 3, 3, 'yes'),
('The Secret Garden', 'Children\'s Fiction', 16, 'secret_garden_cover.jpg', 'A young girl discovers a hidden garden and a new life.', 10.99, 9, 4, 4, 'no'),
('Little Women', 'Classic', 17, 'little_women_cover.jpg', 'A novel about four sisters growing up during the Civil War.', 15.49, 10, 5, 5, 'yes'),
('The Grapes of Wrath', 'Historical Fiction', 18, 'grapes_wrath_cover.jpg', 'The story of a family migrating to California during the Great Depression.', 14.99, 8, 1, 1, 'no'),
('The Outsiders', 'Young Adult', 19, 'outsiders_cover.jpg', 'The rivalry between two groups of teenagers in the 1960s.', 9.99, 11, 2, 2, 'yes'),
('The Handmaid\'s Tale', 'Dystopian', 20, 'handmaids_tale_cover.jpg', 'A dystopian novel set in a theocratic society.', 16.49, 7, 3, 3, 'no'),
```

```
('The Catcher in the Rye', 'Classic', 21,
    'catcher_rye_cover.jpg', 'A young man\'s struggles with life
    and alienation in New York.', 12.49, 6, 4, 4, 'yes'),
('One Hundred Years of Solitude', 'Magical Realism', 22,
    'one_hundred_years_cover.jpg', 'A multi-generational saga set
    in a fictional town.', 17.99, 9, 5, 5, 'no'),
('Beloved', 'Historical Fiction', 23, 'beloved_cover.jpg', 'A
    haunting novel about the legacy of slavery.', 15.49, 8, 1, 1,
    'yes'),
('The Brothers Karamazov', 'Philosophical', 24,
    'brothers_karamazov_cover.jpg', 'A philosophical novel
    exploring faith, doubt, and morality.', 18.99, 10, 2, 2,
    'no'),
('The Fountainhead', 'Philosophical Fiction', 25,
    'fountainhead_cover.jpg', 'A story of an architect who
    refuses to compromise on his ideals.', 19.49, 11, 3, 3,
    'yes'),
('Catch-22', 'Satire', 26, 'catch_22_cover.jpg', 'A satirical
    novel about the absurdity of war.', 16.99, 7, 4, 4, 'no'),
('Gone with the Wind', 'Historical Romance', 27,
    'gone_with_the_wind_cover.jpg', 'A love story set during the
    American Civil War.', 14.49, 9, 5, 5, 'yes'),
('The Color Purple', 'Historical Fiction', 28,
    'color_purple_cover.jpg', 'A story about African-American
    women in the early 20th century.', 15.99, 10, 1, 1, 'no'),
('The Road', 'Post-Apocalyptic', 29, 'road_cover.jpg', 'A father
    and son struggle to survive in a post-apocalyptic world.',
    13.99, 11, 2, 2, 'yes'),
('The Shining', 'Horror', 30, 'shining_cover.jpg', 'A family
    encounters supernatural forces at a remote hotel.', 17.49, 8,
    3, 3, 'no'),
('Slaughterhouse-Five', 'Science Fiction', 31,
    'slaughterhouse_five_cover.jpg', 'A soldier travels through
    time and experiences war in a nonlinear fashion.', 12.49, 9,
    4, 4, 'yes'),
('The Jungle Book', 'Children\'s Fiction', 32,
    'jungle_book_cover.jpg', 'The adventures of Mowgli in the
    jungle.', 10.99, 10, 5, 5, 'no'),
('The Call of the Wild', 'Adventure', 33,
    'call_of_the_wild_cover.jpg', 'A dogs journey in the wilds
    of Alaska during the Klondike Gold Rush.', 14.99, 8, 1, 1,
    'yes'),
('The Alchemist', 'Fiction', 34, 'alchemist_cover.jpg', 'A
    boys journey to find his personal legend and treasure.',
    16.99, 9, 2, 2, 'no'),
('The Secret', 'Self-Help', 35, 'secret_cover.jpg', 'A book
    about the law of attraction and manifesting success.', 11.49,
    7, 3, 3, 'yes'),
('The 5th Wave', 'Science Fiction', 36, '5th_wave_cover.jpg', 'A
    post-apocalyptic novel about a girl fighting against alien
    invasion.', 15.49, 8, 4, 4, 'no'),
```

```
('The Witching Hour', 'Fantasy', 37, 'witching_hour_cover.jpg',
   'A novel about a family of witches.', 18.99, 10, 5, 5, 'yes'),
('The Stand', 'Horror', 38, 'stand_cover.jpg', 'A group of
   survivors battles a superflu that wipes out most of
   humanity.', 17.49, 9, 1, 1, 'no'),
('The Girl on the Train', 'Thriller', 39,
   'girl_on_train_cover.jpg', 'A psychological thriller about an
   unreliable narrator.', 12.99, 7, 2, 2, 'yes'),
('Shutter Island', 'Mystery', 40, 'shutter_island_cover.jpg', 'A
   U.S. Marshal investigates a psychiatric hospital on an
   isolated island.', 13.49, 6, 3, 3, 'no'),
('The Girl with the Dragon Tattoo', 'Mystery', 41,
   'dragon_tattoo_cover.jpg', 'A journalist teams up with a
   hacker to solve a missing person case.', 14.99, 8, 4, 4,
   'yes'),
('The Maze Runner', 'Science Fiction', 42,
   'maze_runner_cover.jpg', 'A group of teenagers must escape a
   labyrinth of dangerous trials.', 15.99, 9, 5, 5, 'no');
```

4.3.4 Attendance Table

```
INSERT INTO attendance (user_id, branch_id, entry_time,
   exit_time, books_id) VALUES
(1, 1, '2025-07-16 09:00:00', '2025-07-16 10:00:00', '1'),
(2, 2, '2025-07-16 10:00:00', '2025-07-16 11:00:00', '2'),
(3, 3, '2025-07-16 11:00:00', '2025-07-16 12:00:00', '3'),
(4, 4, '2025-07-16 12:00:00', '2025-07-16 13:00:00', '4'),
(1, 5, '2025-07-16 14:00:00', '2025-07-16 15:00:00', '5'),
(2, 1, '2025-07-17 09:00:00', '2025-07-17 10:00:00', '6'),
(3, 2, '2025-07-17 10:00:00', '2025-07-17 11:00:00', '7'),
(4, 3, '2025-07-17 11:00:00', '2025-07-17 12:00:00', '8'),
(1, 4, '2025-07-17 12:00:00', '2025-07-17 13:00:00', '9'),
(2, 5, '2025-07-17 14:00:00', '2025-07-17 15:00:00', '10'),
(3, 1, '2025-07-18 09:00:00', '2025-07-18 10:00:00', '11'),
(4, 2, '2025-07-18 10:00:00', '2025-07-18 11:00:00', '12'),
(1, 3, '2025-07-18 11:00:00', '2025-07-18 12:00:00', '13'),
(2, 4, '2025-07-18 12:00:00', '2025-07-18 13:00:00', '14'),
(3, 5, '2025-07-18 14:00:00', '2025-07-18 15:00:00', '15');
```

4.3.5 Lend Table

```sql
INSERT INTO lend (user_id, branch_id, received_time,
    return_time, books_id, method) VALUES
(1, 1, '2025-07-16 09:00:00', '2025-07-23 09:00:00', '1',
    'online'),
(2, 2, '2025-07-16 10:00:00', '2025-07-23 10:00:00', '2',
    'offline'),
(3, 3, '2025-07-16 11:00:00', '2025-07-23 11:00:00', '3',
    'online'),
(4, 4, '2025-07-16 12:00:00', '2025-07-23 12:00:00', '4',
    'offline'),
(1, 5, '2025-07-16 14:00:00', '2025-07-23 14:00:00', '5'),
(2, 1, '2025-07-17 09:00:00', '2025-07-23 09:00:00', '6'),
(3, 2, '2025-07-17 10:00:00', '2025-07-23 10:00:00', '7'),
(4, 3, '2025-07-17 11:00:00', '2025-07-23 11:00:00', '8'),
(1, 4, '2025-07-17 12:00:00', '2025-07-23 12:00:00', '9'),
(2, 5, '2025-07-17 14:00:00', '2025-07-23 14:00:00', '10'),
(3, 1, '2025-07-18 09:00:00', '2025-07-23 09:00:00', '11'),
(4, 2, '2025-07-18 10:00:00', '2025-07-23 10:00:00', '12'),
(1, 3, '2025-07-18 11:00:00', '2025-07-23 11:00:00', '13'),
(2, 4, '2025-07-18 12:00:00', '2025-07-23 12:00:00', '14'),
(3, 5, '2025-07-18 14:00:00', '2025-07-23 14:00:00', '15');
```

4.3.6 Buy Table

```sql
INSERT INTO buy (user_id, branch_id, time, books_id, method,
    amount, payment_id) VALUES
(1, 1, '2025-07-16 09:00:00', '1', 'online', 19.99, 'PAY001'),
(2, 2, '2025-07-16 10:00:00', '2', 'offline', 14.99, 'PAY002'),
(3, 3, '2025-07-16 11:00:00', '3', 'online', 12.99, 'PAY003'),
(4, 4, '2025-07-16 12:00:00', '4', 'offline', 9.99, 'PAY004'),
(1, 5, '2025-07-16 14:00:00', '5', 'online', 11.99, 'PAY005'),
(2, 1, '2025-07-17 09:00:00', '6', 'offline', 10.99, 'PAY006'),
(3, 2, '2025-07-17 10:00:00', '7', 'online', 13.99, 'PAY007'),
(4, 3, '2025-07-17 11:00:00', '8', 'offline', 14.49, 'PAY008'),
(1, 4, '2025-07-17 12:00:00', '9', 'online', 17.99, 'PAY009'),
(2, 5, '2025-07-17 14:00:00', '10', 'offline', 18.49, 'PAY010'),
(3, 1, '2025-07-18 09:00:00', '11', 'online', 20.99, 'PAY011'),
(4, 2, '2025-07-18 10:00:00', '12', 'offline', 19.99, 'PAY012'),
(1, 3, '2025-07-18 11:00:00', '13', 'online', 21.49, 'PAY013'),
(2, 4, '2025-07-18 12:00:00', '14', 'offline', 22.99, 'PAY014'),
(3, 5, '2025-07-18 14:00:00', '15', 'online', 23.99, 'PAY015');
```

4.3.7 Branches Table

```sql
INSERT INTO branches (name, address, email, phone) VALUES
('Central Branch', '123 Main Street', 'central@example.com',
   '1234567890'),
('West Branch', '456 Oak Avenue', 'west@example.com',
   '2345678901'),
('East Branch', '789 Pine Road', 'east@example.com',
   '3456789012'),
('South Branch', '101 Maple Street', 'south@example.com',
   '4567890123'),
('North Branch', '202 Birch Lane', 'north@example.com',
   '5678901234'),
('Central West Branch', '125 Main Street',
   'centralwest@example.com', '1234567891'),
('South East Branch', '457 Oak Avenue', 'southeast@example.com',
   '2345678902'),
('North East Branch', '790 Pine Road', 'northeast@example.com',
   '3456789013'),
('West North Branch', '103 Maple Street',
   'westnorth@example.com', '4567890124'),
('Central East Branch', '205 Birch Lane',
   'centraleast@example.com', '5678901235'),
('West South Branch', '300 Elm Street', 'westsouth@example.com',
   '6789012345'),
('East South Branch', '600 Birch Lane', 'eastsouth@example.com',
   '7890123456'),
('Central North Branch', '777 Oak Avenue',
   'centralnorth@example.com', '8901234567'),
('South West Branch', '111 Pine Road', 'southwest@example.com',
   '9012345678'),
('North Central Branch', '333 Cedar Street',
   'northcentral@example.com', '0123456789');
```

4.3.8 News Table

```sql
INSERT INTO news (title, content, author_id, date) VALUES
('New Book Arrival: Harry Potter', 'We have received new copies of Harry Potter and the Philosopher\'s Stone.', 1, '2025-07-16'),
('New Release: 1984', 'We are excited to announce the release of George Orwell\'s 1984.', 2, '2025-07-16'),
('Upcoming Event: Author Meet and Greet', 'Join us for a meet and greet with local authors.', 3, '2025-07-16'),
('Book Review: Pride and Prejudice', 'A review of Jane Austen\'s classic novel.', 4, '2025-07-16'),
('Author Spotlight: Ernest Hemingway', 'A deep dive into the works of Ernest Hemingway.', 5, '2025-07-16'),
('New Book Arrival: The Old Man and the Sea', 'We have received new copies of The Old Man and the Sea.', 6, '2025-07-17'),
('Upcoming Event: Writing Workshop', 'Join us for an upcoming writing workshop with local authors.', 7, '2025-07-17'),
('Author Review: Agatha Christie', 'A review of Agatha Christie\'s classic novels.', 8, '2025-07-17'),
('Featured Author: George Orwell', 'A spotlight on the works of George Orwell.', 9, '2025-07-18'),
('Latest Book Release: The Catcher in the Rye', 'We are excited to announce the release of The Catcher in the Rye.', 10, '2025-07-18'),
('Exclusive Interview: Mark Twain', 'Join us for an exclusive interview with Mark Twain.', 11, '2025-07-18'),
('Book Discussion: Moby Dick', 'Join us for a book discussion on Moby Dick.', 12, '2025-07-18'),
('Author of the Month: J.K. Rowling', 'Our featured author of the month is J.K. Rowling.', 13, '2025-07-19'),
('Book Club Meeting: Animal Farm', 'Join our book club to discuss Animal Farm.', 14, '2025-07-19'),
('New Release: The Great Gatsby', 'We are thrilled to announce the release of The Great Gatsby.', 15, '2025-07-19');
```

5 Query Examples

This section demonstrates various SQL queries that can be executed on our database to retrieve useful information.

5.1 Basic Queries

5.1.1 List all books with their authors

```sql
SELECT b.id AS book_id, b.title, b.description, b.price,
       a.name AS author_name, a.bio
FROM books b
JOIN authors a ON b.author_id = a.id;
```

The corresponding relational algebra expression is :

$$\Pi_{\text{book\_id, title, description, price, author\_name, bio}}(\text{books} \bowtie_{\text{author\_id}} \text{authors})$$

BOOK_ID	TITLE	DESCRIPTION	PRICE	AUTHOR_NAME	BIO
1	Harry Potter and the Philosophers Stone	First book in the Harry Potter series.	19.99	J.K. Rowling	British author, best known for the Harry Potter series.
2	1984	A novel about a totalitarian regime.	14.99	George Orwell	English novelist and essayist, known for 1984 and Animal Farm.
3	The Adventures of Tom Sawyer	A story of a young boy growing up along the Mississippi River.	12.99	Mark Twain	American writer, humorist, and publisher.
4	Pride and Prejudice	A love story set in the Regency Era.	9.99	Jane Austen	English novelist known for Pride and Prejudice.
5	The Old Man and the Sea	A short novel about a fishermans struggle.	11.99	Ernest Hemingway	American novelist and short story writer.
6	Harry Potter and the Chamber of Secrets	Second book in the Harry Potter series.	19.99	J.K. Rowling	British author, best known for the Harry Potter series.
7	Animal Farm	A political satire about a group of farm animals.	10.99	George Orwell	English novelist and essayist, known for 1984 and Animal Farm.
8	The Catcher in the Rye	A young mans struggles in New York.	13.99	Mark Twain	American writer, humorist, and publisher.
9	Emma	A story of matchmaking and misjudgments.	8.99	Jane Austen	English novelist known for Pride and Prejudice.
10	A Farewell to Arms	A love story set during World War I.	12.99	Ernest Hemingway	American novelist and short story writer.
11	Harry Potter and the Prisoner of Azkaban	The third book in the Harry Potter series.	18.99	J.K. Rowling	British author, best known for the Harry Potter series.
12	The Road to Wigan Pier	A reflection on life in working-class Britain.	15.99	George Orwell	English novelist and essayist, known for 1984 and Animal Farm.
13	Huckleberry Finn	The adventures of Huck Finn and Jim down the Mississippi River.	14.99	Mark Twain	American writer, humorist, and publisher.
14	Sense and Sensibility	A story of two sisters finding love.	7.99	Jane Austen	English novelist known for Pride and Prejudice.
15	The Sun Also Rises	A novel about post-war disillusionment.	13.99	Ernest Hemingway	American novelist and short story writer.
16	Moby Dick	A novel about a captain's obsessive quest for revenge on a white whale.	16.99	Mark Twain	American writer, humorist, and publisher.
17	The Great Gatsby	A novel about the American Dream in the 1920s.	10.99	Haruki Murakami	Japanese novelist and translator.
18	To Kill a Mockingbird	A story of racial injustice in the deep South.	13.49	F. Scott Fitzgerald	American novelist, best known for The Great Gatsby.
19	The Odyssey	An ancient Greek epic about Odysseus journey home.	18.49	Leo Tolstoy	Russian writer, best known for War and Peace.
20	The Hobbit	The prequel to The Lord of the Rings, a story of Bilbo Baggins.	15.99	Charles Dickens	English writer and social critic.
21	Les Miserables	A story of justice, love, and redemption in 19th-century France.	19.49	Agatha Christie	English crime writer, known for Hercule Poirot.
22	Brave New World	A utopian society that faces tragic consequences.	14.99	George Orwell	English novelist and essayist, known for 1984 and Animal Farm.
23	Fahrenheit 451	A society where books are burned to maintain control.	11.99	George Orwell	English novelist and essayist, known for 1984 and Animal Farm.
24	Dracula	The classic vampire novel by Bram Stoker.	13.99	Isaac Asimov	American author and professor of biochemistry.
25	Wuthering Heights	A tragic love story set in the Yorkshire moors.	16.99	J.R.R. Tolkien	English writer and philologist, best known for The Lord of the Rings.
26	Frankenstein	The tale of a scientist who creates a monstrous being.	14.49	C.S. Lewis	British writer and theologian.
27	The Picture of Dorian Gray	A man remains young while his portrait ages and bears the consequences of his actions.	17.49	Virginia Woolf	English writer, known for Mrs Dalloway.
28	The Bell Jar	A semi-autobiographical novel by Sylvia Plath.	12.99	Gabriel Garcia Marquez	Colombian novelist, known for One Hundred Years of Solitude.

FIGURE 4 – List all books with their authors

5.1.2 Find all branches serving a specific book (e.g., Animal Farm)

```sql
SELECT br.id, br.name AS branch_name, br.address, br.email
FROM branches br
JOIN books b ON br.id = b.branch_id
WHERE b.title = 'Animal Farm';
```

The corresponding relational algebra expression is :

$$\Pi_{\text{id, branch\_name, address, email}}(\sigma_{\text{title = 'Animal Farm'}}(\text{branches} \bowtie_{\text{branch\_id}} \text{books}))$$

ID	BRANCH_NAME	ADDRESS	EMAIL
2	West Branch	456 Oak Avenue	west@example.com

FIGURE 5 – Find all branches serving 'Animal Farm'

5.2 Join Operations

5.2.1 Natural Join

```sql
SELECT b.title, a.name AS author_name
FROM books b
NATURAL JOIN authors a;
```

The corresponding relational algebra expression is :

$$\Pi_{\text{title, author\_name}}(\text{books} \bowtie \text{authors})$$

TITLE	AUTHOR_NAME
Harry Potter and the Philosophers Stone	J.K. Rowling
1984	George Orwell
The Adventures of Tom Sawyer	Mark Twain
Pride and Prejudice	Jane Austen
The Old Man and the Sea	Ernest Hemingway
Harry Potter and the Chamber of Secrets	Haruki Murakami
Animal Farm	F. Scott Fitzgerald
The Catcher in the Rye	Leo Tolstoy
Emma	Charles Dickens
A Farewell to Arms	Agatha Christie
Harry Potter and the Prisoner of Azkaban	Isaac Asimov
The Road to Wigan Pier	J.R.R. Tolkien
Huckleberry Finn	C.S. Lewis
Sense and Sensibility	Virginia Woolf
The Sun Also Rises	Gabriel Garcia Marquez

FIGURE 6 – Natural Join - Books and Authors

5.2.2 Outer Join

```sql
SELECT b.title, a.name AS author_name
FROM books b
LEFT OUTER JOIN authors a ON b.author_id = a.id;
```

The corresponding relational algebra expression is :

$$\Pi_{\text{title, author\_name}}(\text{books LEFT OUTER JOIN authors})$$

TITLE	AUTHOR_NAME
Harry Potter and the Philosophers Stone	J.K. Rowling
Harry Potter and the Chamber of Secrets	J.K. Rowling
Harry Potter and the Prisoner of Azkaban	J.K. Rowling
1984	George Orwell
Animal Farm	George Orwell
The Road to Wigan Pier	George Orwell
Brave New World	George Orwell
Fahrenheit 451	George Orwell
The Adventures of Tom Sawyer	Mark Twain
The Catcher in the Rye	Mark Twain
Huckleberry Finn	Mark Twain
Moby Dick	Mark Twain
Pride and Prejudice	Jane Austen
Emma	Jane Austen
Sense and Sensibility	Jane Austen

TITLE	AUTHOR_NAME
The Old Man and the Sea	Ernest Hemingway
A Farewell to Arms	Ernest Hemingway
The Sun Also Rises	Ernest Hemingway
The Great Gatsby	Haruki Murakami
To Kill a Mockingbird	F. Scott Fitzgerald
The Odyssey	Leo Tolstoy
The Hobbit	Charles Dickens
Les Miserables	Agatha Christie
Dracula	Isaac Asimov
Wuthering Heights	J.R.R. Tolkien
Frankenstein	C.S. Lewis
The Picture of Dorian Gray	Virginia Woolf
The Bell Jar	Gabriel Garcia Marquez

FIGURE 7 – Outer Join - Books and Authors

5.2.3 Cross Product

```sql
SELECT b.title, a.name AS author_name
FROM books b
CROSS JOIN authors a;
```

The corresponding relational algebra expression is :

$$\text{books} \times \text{authors}$$

TITLE	AUTHOR_NAME
Harry Potter and the Philosophers Stone	J.K. Rowling
1984	J.K. Rowling
The Adventures of Tom Sawyer	J.K. Rowling
Pride and Prejudice	J.K. Rowling
The Old Man and the Sea	J.K. Rowling
Harry Potter and the Chamber of Secrets	J.K. Rowling
Animal Farm	J.K. Rowling
The Catcher in the Rye	J.K. Rowling
Emma	J.K. Rowling
A Farewell to Arms	J.K. Rowling
Harry Potter and the Prisoner of Azkaban	J.K. Rowling
The Road to Wigan Pier	J.K. Rowling
Huckleberry Finn	J.K. Rowling
Sense and Sensibility	J.K. Rowling
The Sun Also Rises	J.K. Rowling
Moby Dick	J.K. Rowling
The Great Gatsby	J.K. Rowling
To Kill a Mockingbird	J.K. Rowling
The Odyssey	J.K. Rowling
The Hobbit	J.K. Rowling
Les Miserables	J.K. Rowling
Brave New World	J.K. Rowling
Fahrenheit 451	J.K. Rowling
Harry Potter and the Philosophers Stone	George Orwell
1984	George Orwell
The Adventures of Tom Sawyer	George Orwell
Pride and Prejudice	George Orwell
The Old Man and the Sea	George Orwell
Harry Potter and the Chamber of Secrets	George Orwell
Animal Farm	George Orwell
The Catcher in the Rye	George Orwell
Emma	George Orwell
A Farewell to Arms	George Orwell
Harry Potter and the Prisoner of Azkaban	George Orwell
The Road to Wigan Pier	George Orwell
Huckleberry Finn	George Orwell
Sense and Sensibility	George Orwell
The Sun Also Rises	George Orwell
Moby Dick	George Orwell
The Great Gatsby	George Orwell
To Kill a Mockingbird	George Orwell
The Odyssey	George Orwell
The Hobbit	George Orwell
Les Miserables	George Orwell
Brave New World	George Orwell

Download CSV

Rows 1 - 50. More rows exist.

FIGURE 8 – Cross Product - Books and Authors

5.3 Nested Queries

5.3.1 Nested Query with (NOT) EXISTS

```sql
SELECT b.title
FROM books b
WHERE EXISTS (
    SELECT 1
    FROM authors a
    WHERE a.id = b.author_id AND a.name = 'George Orwell'
);
```

The corresponding relational algebra expression is :

$$\Pi_{\text{title}}(\sigma_{\text{EXISTS}}(\text{books}))$$

TITLE
1984
Animal Farm
The Road to Wigan Pier
Brave New World
Fahrenheit 451

FIGURE 9 – Nested Query with EXISTS - Books by George Orwell

5.3.2 Nested Query with IN

```sql
SELECT b.title
FROM books b
WHERE b.author_id IN (
    SELECT a.id
    FROM authors a
    WHERE a.name = 'George Orwell'
);
```

The corresponding relational algebra expression is :

$$\Pi_{\text{title}}(\sigma_{\text{author\_id} \in \text{authors.id}}(\text{books}))$$

TITLE
1984
Animal Farm
The Road to Wigan Pier
Brave New World
Fahrenheit 451

FIGURE 10 – Nested Query with IN - Books by George Orwell

5.3.3 Scalar Subquery

```sql
SELECT b.title, b.price
FROM books b
WHERE b.price > (
    SELECT AVG(price)
    FROM books
);
```

The corresponding relational algebra expression is :

$$\Pi_{\text{title, price}}(\sigma_{\text{price}>\text{AVG(price)}}(\text{books}))$$

TITLE	PRICE
Harry Potter and the Philosophers Stone	19.99
1984	14.99
Harry Potter and the Chamber of Secrets	19.99
Harry Potter and the Prisoner of Azkaban	18.99
The Road to Wigan Pier	15.99
Huckleberry Finn	14.99
Moby Dick	16.99
The Odyssey	18.49
The Hobbit	15.99
Les Miserables	19.49
Brave New World	14.99
Wuthering Heights	16.99
The Picture of Dorian Gray	17.49

FIGURE 11 – Scalar Subquery - Books with Price Above Average

5.3.4 WITH Clause

```sql
WITH AuthorAvgPrice AS (
    SELECT author_id, AVG(price) AS avg_price
    FROM books
    GROUP BY author_id
)
SELECT a.name AS author_name, avg_price
FROM authors a
JOIN AuthorAvgPrice ap ON a.id = ap.author_id;
```

The corresponding relational algebra expression is :

$$\Pi_{\text{author\_name, avg\_price}}(\text{authors} \bowtie \text{AuthorAvgPrice})$$

AUTHOR_NAME	AVG_PRICE
Haruki Murakami	10.99
Virginia Woolf	17.49
J.K. Rowling	19.656666666666666666666666666666666666667
F. Scott Fitzgerald	13.49
Gabriel Garcia Marquez	12.99
George Orwell	13.79
Leo Tolstoy	18.49
Isaac Asimov	13.99
J.R.R. Tolkien	16.99
Jane Austen	8.99
Ernest Hemingway	12.99
Agatha Christie	19.49
Mark Twain	14.74
Charles Dickens	15.99
C.S. Lewis	14.49

FIGURE 12 – WITH Clause - Average Price Per Author

5.4 Aggregate Functions

5.4.1 Count, Group By, and Having

```sql
SELECT author_id, COUNT(*) AS book_count
FROM books
GROUP BY author_id
HAVING COUNT(*) > 3;
```

The corresponding relational algebra expression is :

$$\Pi_{\text{author\_id, COUNT}}(\sigma_{\text{COUNT}>3}(\text{books}))$$

AUTHOR_ID	BOOK_COUNT
2	5
3	4

FIGURE 13 – Count and Group By - Books Per Author

5.4.2 Average with Group By

```sql
SELECT author_id, AVG(price) AS avg_price
FROM books
GROUP BY author_id;
```

The corresponding relational algebra expression is :

$$\Pi_{\text{author\_id, AVG(price)}}(\text{books})$$

AUTHOR_ID	AVG_PRICE
6	10.99
14	17.49
1	19.656666666666666666666666666666666666667
7	13.49
15	12.99
2	13.79
8	18.49
11	13.99
12	16.99
4	8.99
5	12.99
10	19.49
3	14.74
9	15.99
13	14.49

FIGURE 14 – Average Price Per Author

5.5 String and Set Operations

5.5.1 String Operations

```sql
SELECT b.title
FROM books b
WHERE b.title LIKE '%Farm%';
```

The corresponding relational algebra expression is :

$$\Pi_{\text{title}}(\sigma_{\text{title LIKE '\%Farm\%'}}(\text{books}))$$

TITLE
Animal Farm

FIGURE 15 – String Operation - Books with 'Farm' in Title

5.5.2 Set Operations

```sql
SELECT title FROM books
UNION
SELECT title FROM news;
```

The corresponding relational algebra expression is :

$$\text{books.title} \cup \text{news.title}$$

TITLE
1984
A Farewell to Arms
Animal Farm
Author Review: Agatha Christie
Author Spotlight: Ernest Hemingway
Author of the Month: J.K. Rowling
Book Club Meeting: Animal Farm
Book Discussion: Moby Dick
Book Review: Pride and Prejudice
Brave New World
Dracula
Emma
Exclusive Interview: Mark Twain
Fahrenheit 451
Featured Author: George Orwell
Frankenstein
Harry Potter and the Chamber of Secrets
Harry Potter and the Philosophers Stone
Harry Potter and the Philosophers Stone
Harry Potter and the Prisoner of Azkaban
Huckleberry Finn
Latest Book Release: The Catcher in the Rye
Les Miserables
Moby Dick
New Book Arrival: Harry Potter
New Book Arrival: The Old Man and the Sea
New Release: 1984
New Release: The Great Gatsby
Pride and Prejudice
Sense and Sensibility
The Adventures of Tom Sawyer
The Bell Jar
The Catcher in the Rye
The Great Gatsby
The Hobbit
The Odyssey
The Old Man and the Sea
The Picture of Dorian Gray
The Road to Wigan Pier
The Sun Also Rises
To Kill a Mockingbird
Upcoming Event: Author Meet and Greet
Upcoming Event: Writing Workshop
Wuthering Heights

Download CSV

43 rows selected.

FIGURE 16 – Set Operation - Union of Books and News Titles

5.6 Update and Delete Operations

5.6.1 Update Operation

```sql
UPDATE books
SET price = price + 5
WHERE title = 'Animal Farm';
```

The corresponding relational algebra expression is :

$$\text{UPDATE}(\text{books})$$

TITLE	PRICE
Animal Farm	10.99

Download CSV

1 row(s) updated.

TITLE	PRICE
Animal Farm	15.99

FIGURE 17 – Update Operation - Increase Price of 'Animal Farm'

5.6.2 Delete Operation

```
DELETE FROM books
WHERE title = 'Animal Farm';
```

The corresponding relational algebra expression is :

$$\text{DELETE}(\text{books})$$

```
DELETE FROM lend WHERE books_id = (SELECT id FROM books WHERE title = 'Animal Farm');
DELETE FROM attendance WHERE books_id = (SELECT id FROM books WHERE title = 'Animal Farm');
DELETE FROM buy WHERE books_id = (SELECT id FROM books WHERE title = 'Animal Farm');
-- We do the work because, books are also used as secondary key on these table

DELETE FROM books
WHERE title = 'Animal Farm';

1 row(s) deleted.
```

FIGURE 18 – Delete Operation - Remove 'Animal Farm' from Books

6 Views

Views provide a way to create virtual tables that can simplify complex queries and provide an additional layer of security.

6.1 Creating Views

6.1.1 Book Author View

This view lists all books with their associated authors :

```sql
CREATE VIEW book_author AS
SELECT b.id AS book_id, b.title, a.name AS author_name
FROM books b
JOIN authors a ON b.author_id = a.id;
```

6.2 Querying Views

Example of querying the book summary view :

```sql
SELECT ba.title, b.price, ba.author_name
FROM book_author ba
JOIN books b ON ba.book_id = b.id
WHERE b.price > 15.0
ORDER BY b.price DESC;
```

TITLE	PRICE	AUTHOR_NAME
Harry Potter and the Philosophers Stone	19.99	J.K. Rowling
Harry Potter and the Chamber of Secrets	19.99	J.K. Rowling
Les Miserables	19.49	Agatha Christie
Harry Potter and the Prisoner of Azkaban	18.99	J.K. Rowling
The Odyssey	18.49	Leo Tolstoy
The Picture of Dorian Gray	17.49	Virginia Woolf
Moby Dick	16.99	Mark Twain
Wuthering Heights	16.99	J.R.R. Tolkien
The Road to Wigan Pier	15.99	George Orwell
The Hobbit	15.99	Charles Dickens

FIGURE 19 – Querying Views

7 Functional Dependencies and Normalization

7.1 Functional Dependencies

The following functional dependencies exist in our database :

7.1.1 Books Table

- **book_id → title, type, cover, description, price, amount, location, branch_id, pdf**
  - The **book_id** uniquely identifies each book, and all other attributes (title, type, cover, description, price, etc.) depend solely on it.

7.1.2 Authors Table

- **author_id → name, bio, photo, email, nbooks, nnews**
  - The **author_id** uniquely identifies each author, and all other attributes (name, bio, etc.) depend solely on it.

7.1.3 Branches Table

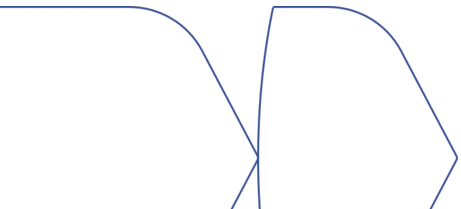
- **id → name, address, email, phone**
  - The **id** uniquely identifies each branch, and all other attributes (name, address, etc.) depend solely on it.

7.1.4 Users Table

- **id → name, username, photo, email, phone, address, password, role, unique_id**
  - The **id** uniquely identifies each user, and all other attributes depend solely on it.

7.1.5 Lend Table

- **id → user_id, branch_id, received_time, return_time, books_id, method**
  - The **id** uniquely identifies each lending record, and all other attributes depend solely on it.



7.1.6 Buy Table

- **id → user_id, branch_id, time, books_id, method, amount, payment_id**
  - The **id** uniquely identifies each purchase record, and all other attributes depend solely on it.

7.1.7 News Table

- **id → title, content, author_id, date**
  - The **id** uniquely identifies each news article, and all other attributes (title, content, etc.) depend solely on it.

7.2 Normalization Analysis

Our database schema is designed to be in **Third Normal Form (3NF)**. Here's the detailed analysis :

7.2.1 First Normal Form (1NF)

All tables satisfy 1NF because :

- **Each table has a primary key :**
  - For example, the **Books** table has **book_id** as its primary key.
- **All attributes contain atomic (indivisible) values :**
  - For example, the **name** attribute in the **Books** table stores a single value (e.g., « The Great Gatsby »).
- **No repeating groups exist :**
  - For example, the **Book Categories** table does not store multiple categories in a single cell.

7.2.2 Second Normal Form (2NF)

All tables satisfy 2NF because :

- **They are in 1NF :**
  - All tables already satisfy the conditions of 1NF.
- **All non-key attributes are fully functionally dependent on the primary key :**
  - For example, in the **Books** table, all attributes (title, description, rating, etc.) depend entirely on the primary key **book_id**.
- **Junction tables (e.g., book_authors, book_categories) have composite primary keys with no non-key attributes :**
  - For example, the **Book_Authors** table has a composite primary key (**book_id, author_id**) and no non-key attributes.

7.2.3 Third Normal Form (3NF)

All tables satisfy 3NF because :

- **They are in 2NF :**
  - All tables already satisfy the conditions of 2NF.
- **No transitive dependencies exist :**
  - Non-key attributes depend only on the primary key, not on other non-key attributes.
  - For example, in the **Books** table, the **price** depends only on **book_id**, not on any other attribute like **title** or **author_id**.

7.2.4 Proof of 3NF

To prove that the database is in 3NF, consider the following :

- **Books Table :**
  - All attributes (title, description, rating, etc.) depend directly on **book_id**.
  - There are no transitive dependencies. For example, **rating** does not depend on **title** or **description**.
- **Authors Table :**
  - All attributes (name, biography, nationality) depend directly on **author_id**.
  - There are no transitive dependencies. For example, **biography** does not depend on **nationality**.
- **Branches Table :**
  - All attributes (name, address, email, phone) depend directly on **id**.
  - There are no transitive dependencies. For example, **category_name** does not depend on any other attribute like **id**.

Thus, the database schema is fully normalized to **Third Normal Form (3NF)**, ensuring data integrity, minimizing redundancy, and optimizing query performance.

8 Technologies Used

- Report : Overleaf for beautiful LaTeX
- Schema-Diagram : Dbdiagram for average schema diagram
- SQL : Oracle for database
- Project : PHP for web implementation.

9 Conclusion

9.1 Summary

In this project, we have designed and implemented a comprehensive database for a Library Management System named « NOVEL NEST ». The main goal was to create a structured and efficient system that handles various aspects of library management, including book inventory, author details, user information, transactions, and more. We went through several stages of database design, including the creation of entities, relationships, and attributes that represent the core data of the library.

We also implemented Functional Dependencies to ensure that the database maintains data integrity, and we performed a detailed Normalization Analysis to bring the schema into Third Normal Form (3NF), ensuring that the database is free from redundancy and transitive dependencies.

Additionally, we demonstrated the creation of various SQL queries and views that help manage the library data efficiently, such as joining Books with Authors, updating book prices, and retrieving records of book transactions. The project provides a solid foundation for building a web-based application with a backend system to interact with the library database.

9.2 Future Plan

In the future, we plan to implement this database as a web-based Library Management Project called « NOVEL NEST ». The implementation will be done using PHP for the backend, which will allow us to create dynamic web pages for library users, including functionalities for book search, borrowing books, user registration, and transaction management.

We aim to integrate this database into a user-friendly interface where both admins and users will have access to different features such as book catalog, book lending/return, book purchase, and news updates related to the library. Additionally, we hope to add features such as real-time notifications, book reservations, and admin dashboards to manage users and books efficiently.

This future phase of the project will make the NOVEL NEST library management system more accessible to users online and provide a comprehensive solution to managing library operations effectively.

THE END