SIMATS ENGINEERING

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

CAPSTONE PROJECT REPORT

PROJECT TITLE

LIBRARY MANAGEMENT SYSTEM

TECHNIQUE

TEAM MEMBERS

S MURALI KRISHNAN 192321093

V KAUSHIK NARAYANAN 192321047

COURSE CODE / NAME

CSA0556 / DATABASE MANAGEMENT SYSTEM FOR RELATIONAL DATABASE

Task 1: Entity Identification and Attributes

1. Book

- **Attributes**:
  - BookID: Unique identifier for each book (Primary Key).
  - Title: The title of the book.
  - Author: The author(s) of the book.
  - Publisher: The publisher of the book.
  - ISBN: International Standard Book Number.
  - YearPublished: The year the book was published.
  - Genre: The genre or category of the book.
  - CopiesAvailable: Number of copies available in the library.

2. Member (Patron)

- **Attributes**:
  - MemberID: Unique identifier for each library member (Primary Key).
  - FirstName: The first name of the member.
  - LastName: The last name of the member.
  - Address: The home address of the member.
  - PhoneNumber: Contact number of the member.
  - Email: Email address of the member.
  - MembershipDate: The date the membership was started.
  - MembershipType: Type of membership (e.g., Regular, Premium).

3. Librarian

- **Attributes**:
  - LibrarianID: Unique identifier for each librarian (Primary Key).
  - FirstName: The first name of the librarian.
  - LastName: The last name of the librarian.
  - EmployeeNumber: Unique employee number.
  - PhoneNumber: Contact number of the librarian.
  - Email: Email address of the librarian.
  - HireDate: The date the librarian was hired.

4. Loan

- **Attributes**:
  - LoanID: Unique identifier for each loan transaction (Primary Key).
  - BookID: Foreign key referencing the Book entity.
  - MemberID: Foreign key referencing the Member entity.
  - LoanDate: The date when the book was borrowed.
  - DueDate: The date when the book is due to be returned.
  - ReturnDate: The date when the book was actually returned.

5. Reservation

- **Attributes**:
  - ReservationID: Unique identifier for each reservation (Primary Key).
  - BookID: Foreign key referencing the Book entity.

    - MemberID: Foreign key referencing the Member entity.
    - ReservationDate: The date when the reservation was made.
    - Status: Status of the reservation (e.g., Pending, Completed).

6. Fine

- **Attributes**:
    - FineID: Unique identifier for each fine imposed (Primary Key).
    - MemberID: Foreign key referencing the Member entity.
    - LoanID: Foreign key referencing the Loan entity.
    - Amount: Amount of the fine.
    - DateImposed: The date when the fine was imposed.
    - Status: Status of the fine (e.g., Paid, Unpaid).

Task 2: Relationship Modeling

1. Book - Loan Relationship

- **Type**: One-to-Many
- **Description**: A book can be loaned out multiple times, but each loan transaction pertains to a single book.
- **Entities Involved**: Book and Loan
- **Foreign Key**: BookID in the Loan table references BookID in the Book table.

2. Member - Loan Relationship

- **Type**: One-to-Many
- **Description**: A member can borrow multiple books, and each loan transaction is associated with a single member.
- **Entities Involved**: Member and Loan
- **Foreign Key**: MemberID in the Loan table references MemberID in the Member table.

3. Book - Reservation Relationship

- **Type**: One-to-Many
- **Description**: A book can have multiple reservations, but each reservation pertains to a single book.
- **Entities Involved**: Book and Reservation
- **Foreign Key**: BookID in the Reservation table references BookID in the Book table.

4. Member - Reservation Relationship

- **Type**: One-to-Many
- **Description**: A member can reserve multiple books, and each reservation is associated with a single member.
- **Entities Involved**: Member and Reservation

- **Foreign Key**: MemberID in the Reservation table references MemberID in the Member table.

5. Loan - Fine Relationship

- **Type**: One-to-One
- **Description**: Each loan can result in a single fine, and each fine is associated with a specific loan.
- **Entities Involved**: Loan and Fine
- **Foreign Key**: LoanID in the Fine table references LoanID in the Loan table.

6. Member - Fine Relationship

- **Type**: One-to-Many
- **Description**: A member can have multiple fines, and each fine is associated with a single member.
- **Entities Involved**: Member and Fine
- **Foreign Key**: MemberID in the Fine table references MemberID in the Member table.

7. Librarian - Loan Relationship

- **Type**: One-to-Many
- **Description**: A librarian can manage multiple loan transactions, but each loan transaction is handled by a single librarian.
- **Entities Involved**: Librarian and Loan
- **Foreign Key**: LibrarianID in the Loan table references LibrarianID in the Librarian table.

8. Librarian - Reservation Relationship

- **Type**: One-to-Many
- **Description**: A librarian can manage multiple reservations, but each reservation is handled by a single librarian.
- **Entities Involved**: Librarian and Reservation
- **Foreign Key**: LibrarianID in the Reservation table references LibrarianID in the Librarian table.

Task 3: ER Diagram Design

ER Diagram:

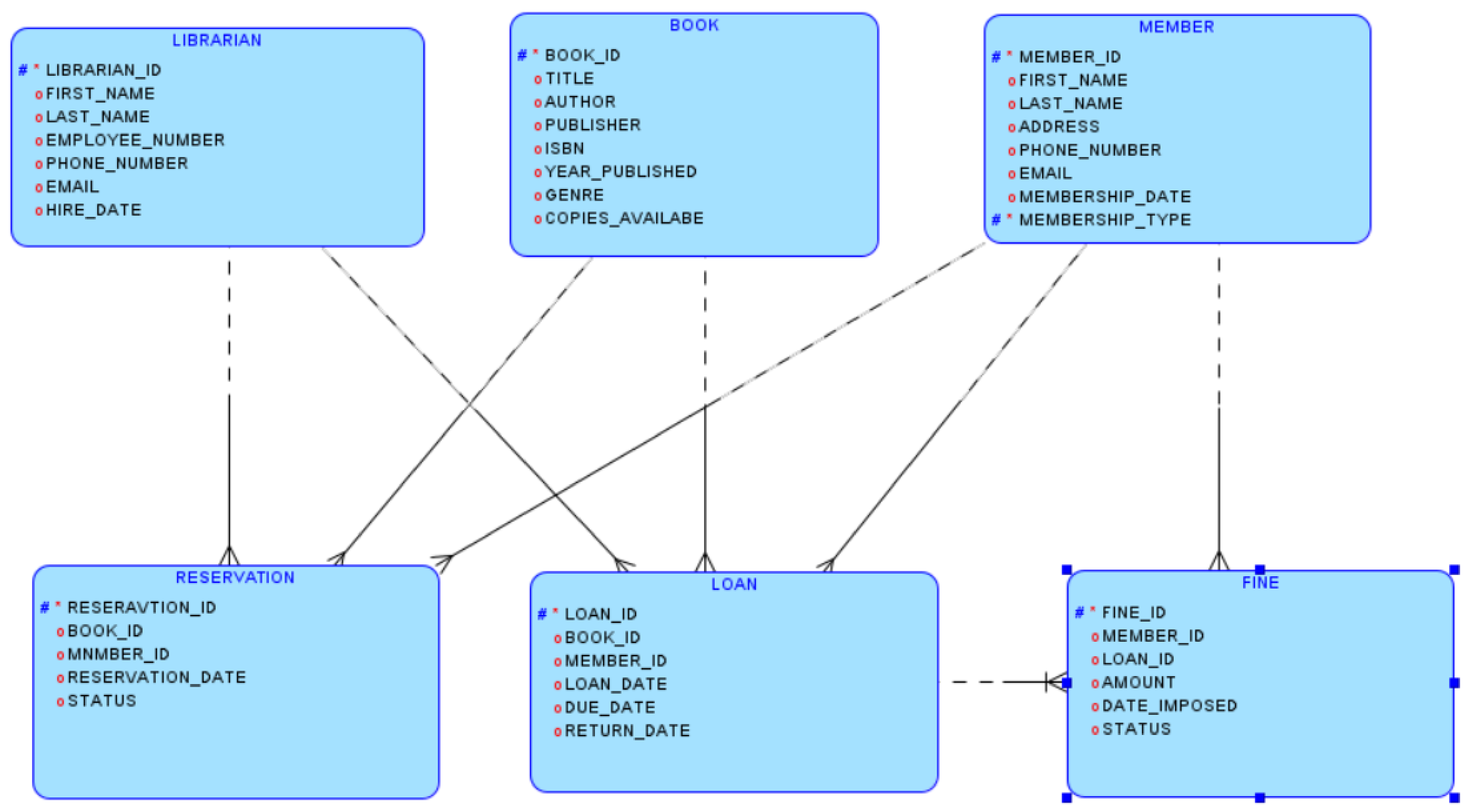
Entities: Book, Member, Librarian, Loan, Reservation, Fine.

Relationships: Connect the entities as described in Task 2, ensuring primary keys and foreign keys are correctly labeled.

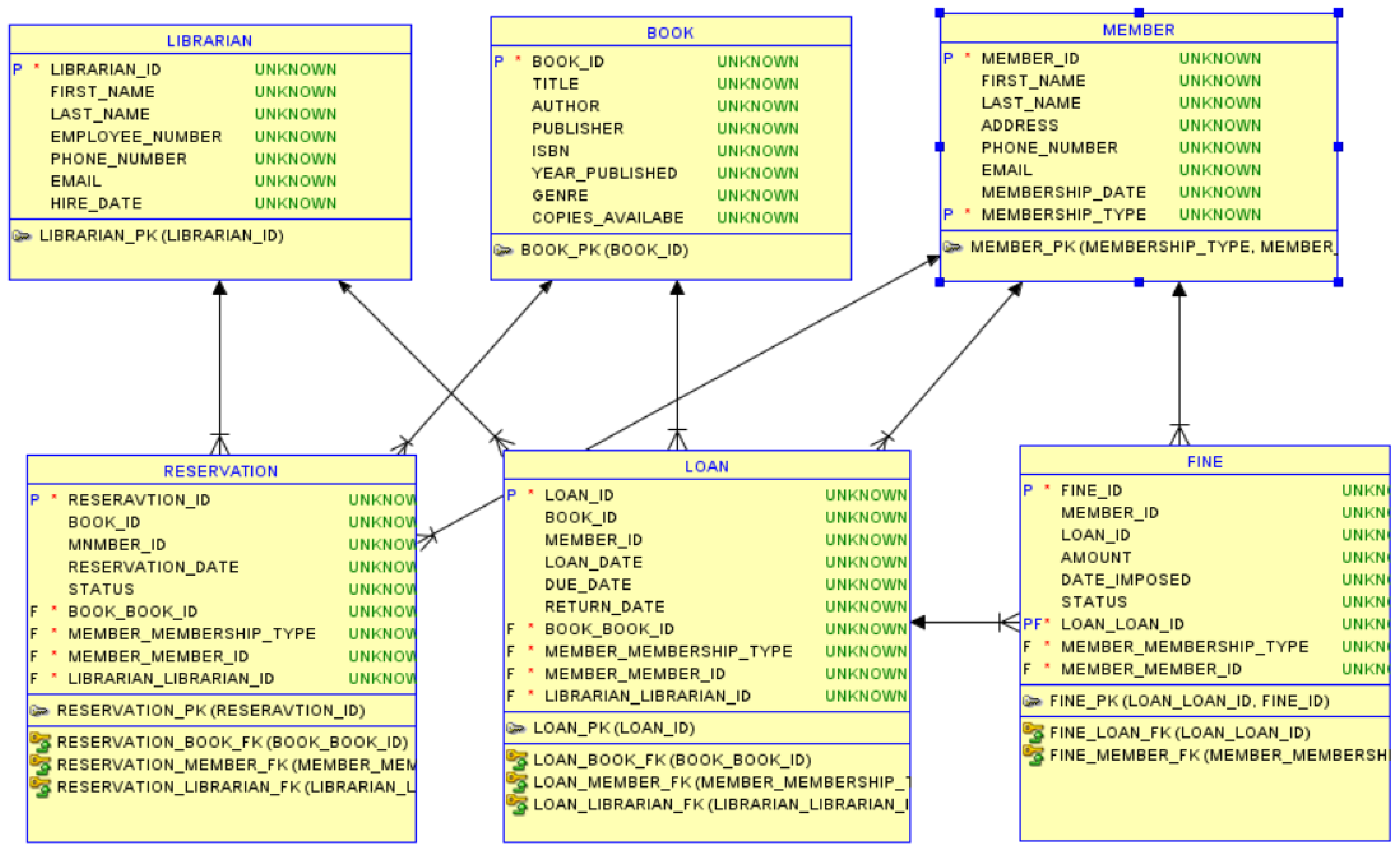
Primary Keys (PK): Unique identifiers for each entity.

Foreign Keys (FK): References to related entities.

Logical Model



Relational Model



Task 4: Justification and Normalization

1. Justification of the Design:

The design of the Library Management System (LMS) is grounded in the need for efficient, accurate, and reliable management of library resources and operations. Here’s why the design choices were made:

- **Entity Identification**: The chosen entities (Book, Member, Librarian, Loan, Reservation, Fine) reflect the core components of a library system. Each entity corresponds to a real-world object or concept essential for library operations.
- **Relationship Modeling**: The relationships between entities were defined to mirror the interactions in a typical library. For instance, books are loaned to members, fines are associated with overdue loans, and librarians manage these operations. This modeling ensures that the system can handle tasks such as tracking who borrowed which book, when it’s due back, and whether a fine is imposed.
- **Scalability**: The one-to-many relationships allow the system to scale. For example, a Member can borrow many Books, and a Book can be reserved multiple times. This flexibility is crucial as libraries grow and their operations become more complex.

2. Normalization:

Normalization is the process of organizing the fields and tables of a relational database to minimize redundancy and dependency. Here’s how the database schema for the LMS is normalized through different normal forms:

First Normal Form (1NF):

- **Objective**: Ensure that the table is organized such that each column contains atomic values, and each record is unique.
- **Implementation**:
  - Each table (e.g., Book, Member, Loan) has a primary key (BookID, MemberID, LoanID) that uniquely identifies each record.
  - All attributes hold single, indivisible values (e.g., Title, Author, ISBN in Book).
  - Repeating groups or arrays are not allowed; each piece of data is stored in its column.

Second Normal Form (2NF):

- **Objective**: Achieve 1NF and ensure that all non-key attributes are fully functionally dependent on the primary key.
- **Implementation**:
  - In the Loan table, attributes like LoanDate, DueDate, and ReturnDate are fully dependent on LoanID.

- No partial dependencies exist. For instance, in the Loan table, BookID and MemberID together uniquely identify the loan, but non-key attributes like LoanDate depend solely on the composite key.

Third Normal Form (3NF):

- **Objective**: Achieve 2NF and ensure that no transitive dependencies exist between the attributes.
- **Implementation**:
  - In the Fine table, Amount is directly dependent on FineID and not on other non-key attributes, eliminating transitive dependencies.
  - The Member table's attributes (e.g., FirstName, LastName, PhoneNumber) are all dependent directly on the MemberID and not on other non-key attributes.

Advantages of Normalization:

- **Data Integrity**: By ensuring that data is stored in only one place, normalization reduces the risk of data anomalies and ensures consistency across the database.
- **Efficiency**: Normalized tables typically require less storage space and can be queried more efficiently, which is crucial for a system managing potentially large datasets like a library.
- **Maintenance**: Changes in data, such as updating a member's contact information, need to be made in only one place, simplifying maintenance and reducing the chance of errors.

Task 5: SQL Query Optimization

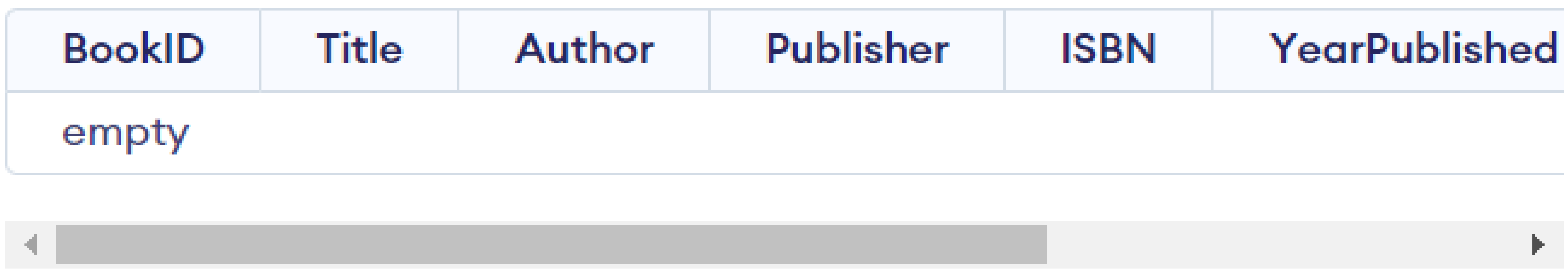
SQL Table Creation with Indexing:

CODE:

```
CREATE TABLE Book (

  BookID INT PRIMARY KEY,

  Title VARCHAR(255) NOT NULL,

  Author VARCHAR(255) NOT NULL,

  Publisher VARCHAR(255),

  ISBN VARCHAR(13) UNIQUE NOT NULL,
  YearPublished INT,
  Genre VARCHAR(100),
  CopiesAvailable INT DEFAULT 0
);
CREATE TABLE Member (
  MemberID INT PRIMARY KEY,
  FirstName VARCHAR(100) NOT NULL,
  LastName VARCHAR(100) NOT NULL,
  Address VARCHAR(255),
  PhoneNumber VARCHAR(15),
  Email VARCHAR(100) UNIQUE NOT NULL,
  MembershipDate DATE NOT NULL,
  MembershipType VARCHAR(50) NOT NULL
);
CREATE TABLE Librarian (
  LibrarianID INT PRIMARY KEY,
  FirstName VARCHAR(100) NOT NULL,
  LastName VARCHAR(100) NOT NULL,
  EmployeeNumber VARCHAR(50) UNIQUE NOT NULL,
  PhoneNumber VARCHAR(15),
  Email VARCHAR(100) UNIQUE NOT NULL,
  HireDate DATE NOT NULL
);
CREATE TABLE Loan (
  LoanID INT PRIMARY KEY,
  BookID INT NOT NULL,
  MemberID INT NOT NULL,
  LibrarianID INT NOT NULL,
  LoanDate DATE NOT NULL,
  DueDate DATE NOT NULL,
  ReturnDate DATE,
  FOREIGN KEY (BookID) REFERENCES Book(BookID),
  FOREIGN KEY (MemberID) REFERENCES Member(MemberID),
  FOREIGN KEY (LibrarianID) REFERENCES Librarian(LibrarianID)
);
CREATE TABLE Reservation (
  ReservationID INT PRIMARY KEY,
  BookID INT NOT NULL,
  MemberID INT NOT NULL,
  ReservationDate DATE NOT NULL,
  Status VARCHAR(50) NOT NULL,
  FOREIGN KEY (BookID) REFERENCES Book(BookID),
  FOREIGN KEY (MemberID) REFERENCES Member(MemberID)
);
CREATE TABLE Fine (
  FineID INT PRIMARY KEY,
  MemberID INT NOT NULL,
  LoanID INT NOT NULL,
  Amount DECIMAL(10, 2) NOT NULL,
  DateImposed DATE NOT NULL,
    Status VARCHAR(50) NOT NULL,
    FOREIGN KEY (MemberID) REFERENCES Member(MemberID),
    FOREIGN KEY (LoanID) REFERENCES Loan(LoanID)
);
CREATE INDEX idx_book_title ON Book(Title);
CREATE INDEX idx_book_author ON Book(Author);
CREATE UNIQUE INDEX idx_book_isbn ON Book(ISBN);
CREATE UNIQUE INDEX idx_member_email ON Member(Email);
CREATE INDEX idx_loan_duedate ON Loan(DueDate);
CREATE INDEX idx_reservation_status ON Reservation(Status);
CREATE INDEX idx_fine_status ON Fine(Status);
```

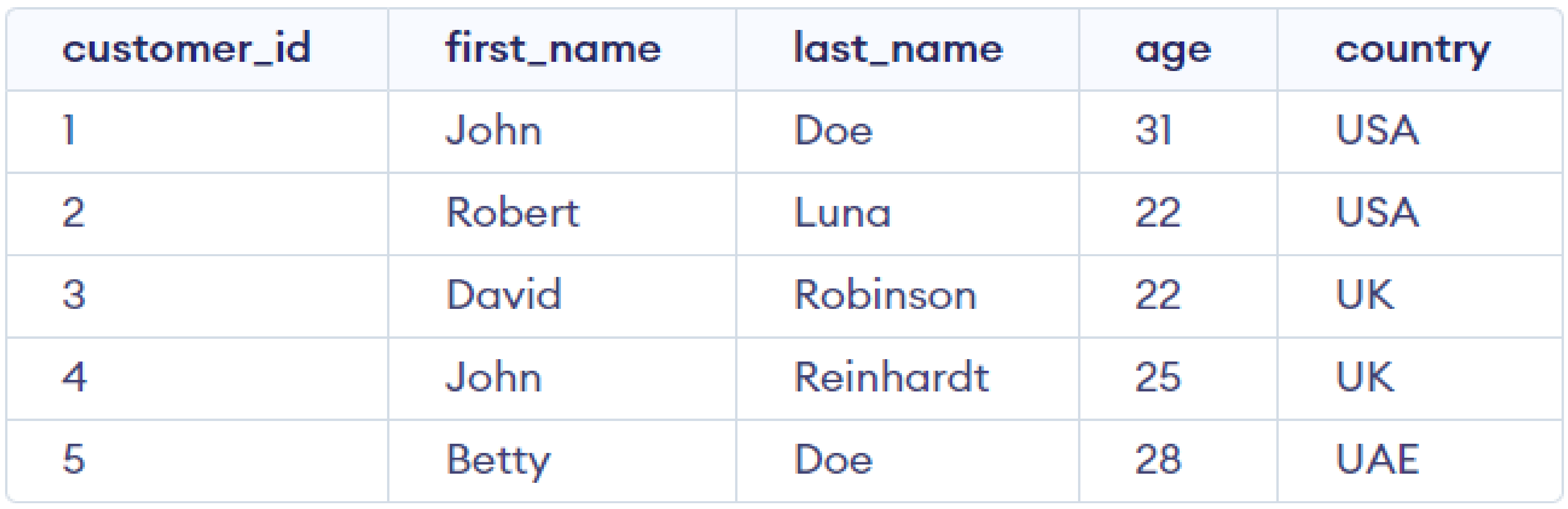
OUTPUT

Book



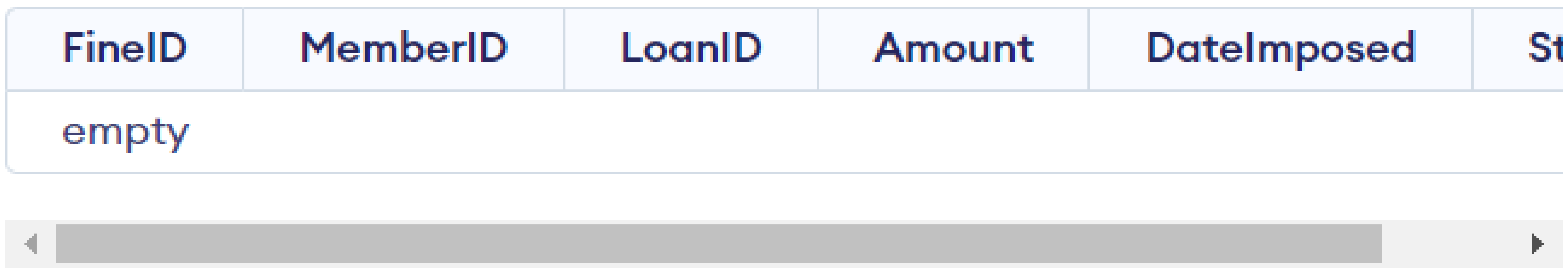
BookID	Title	Author	Publisher	ISBN	YearPublished
empty					

Customers

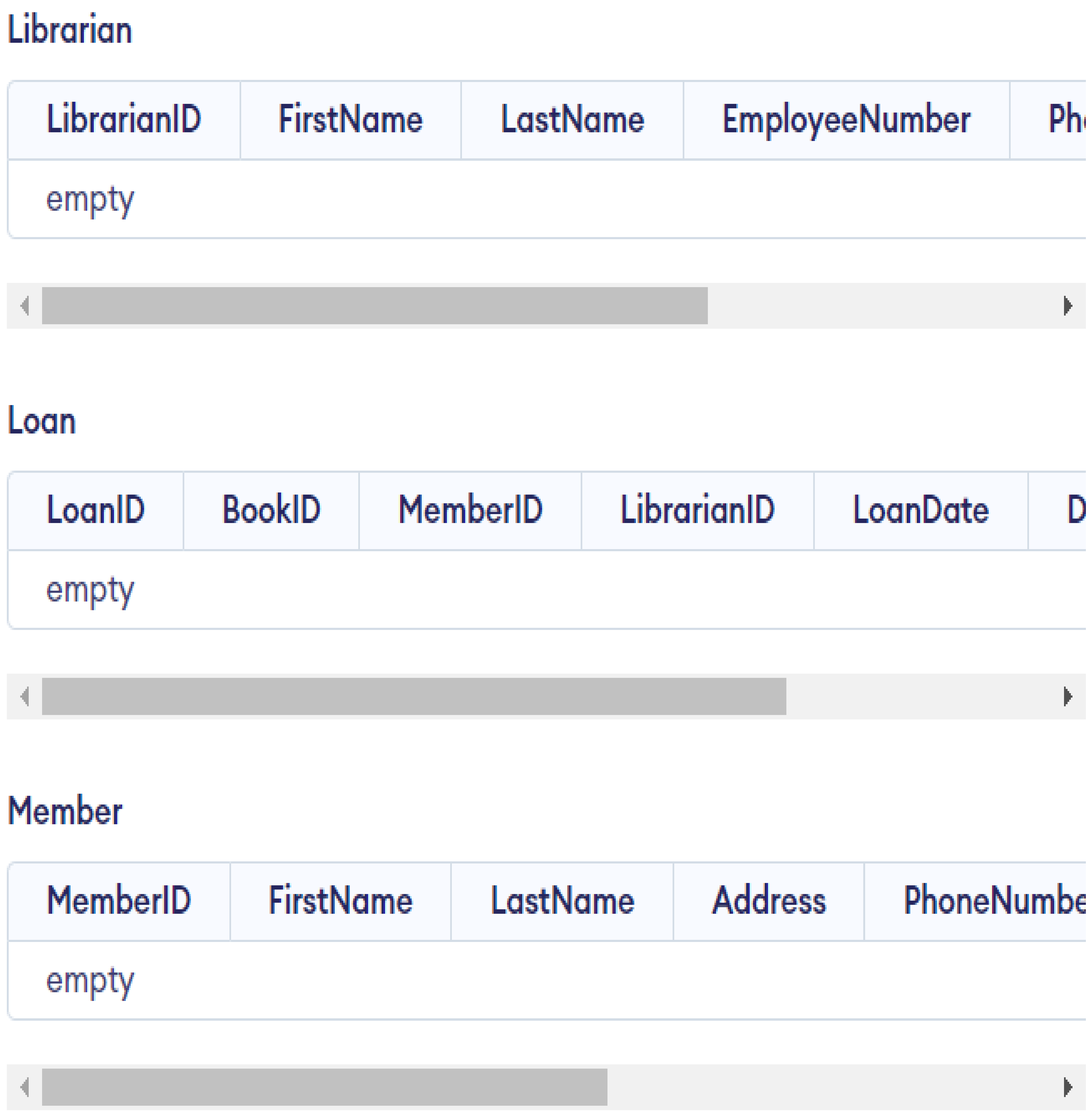


customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Fine



FineID	MemberID	LoanID	Amount	DateImposed	St
empty					



Librarian

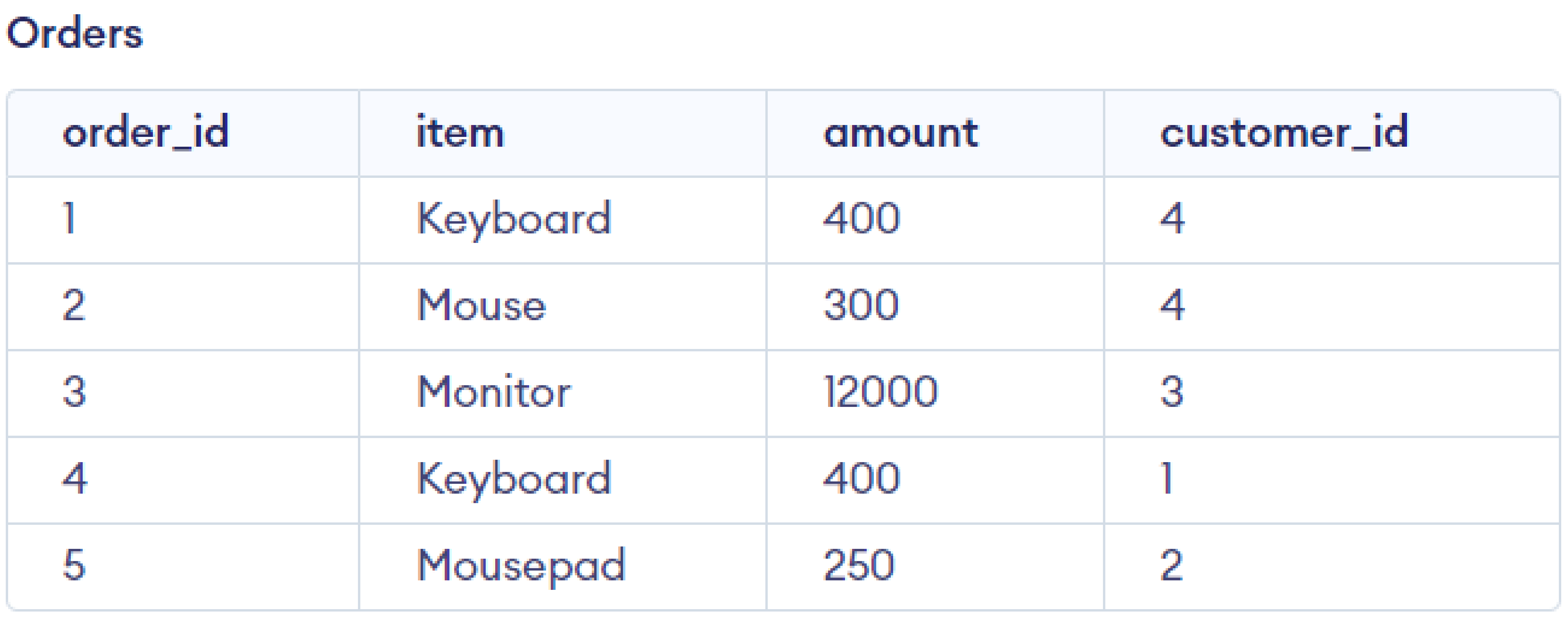
LibrarianID	FirstName	LastName	EmployeeNumber	Ph
empty				

Loan

LoanID	BookID	MemberID	LibrarianID	LoanDate	D
empty					

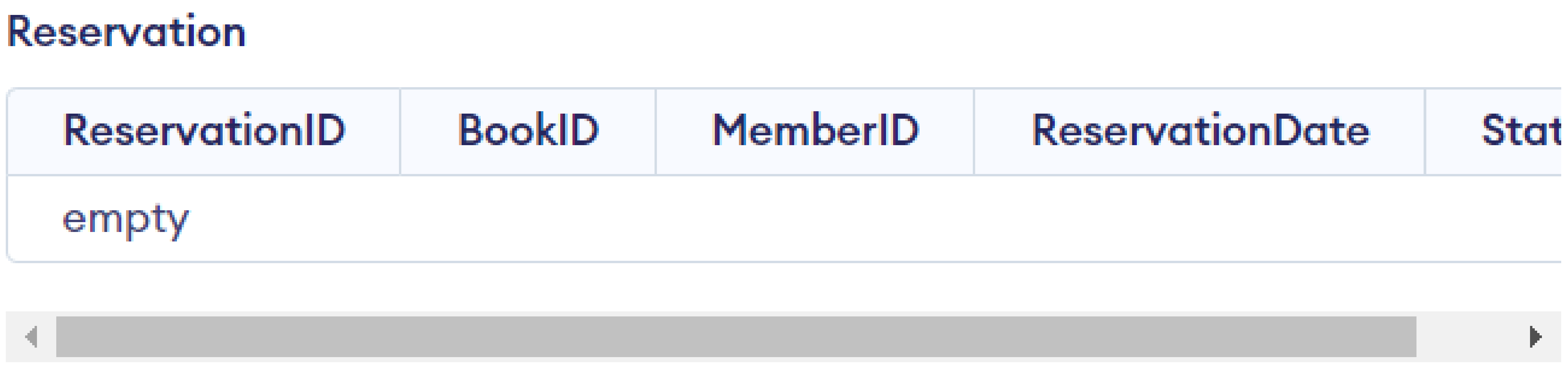
Member

MemberID	FirstName	LastName	Address	PhoneNumbe
empty				



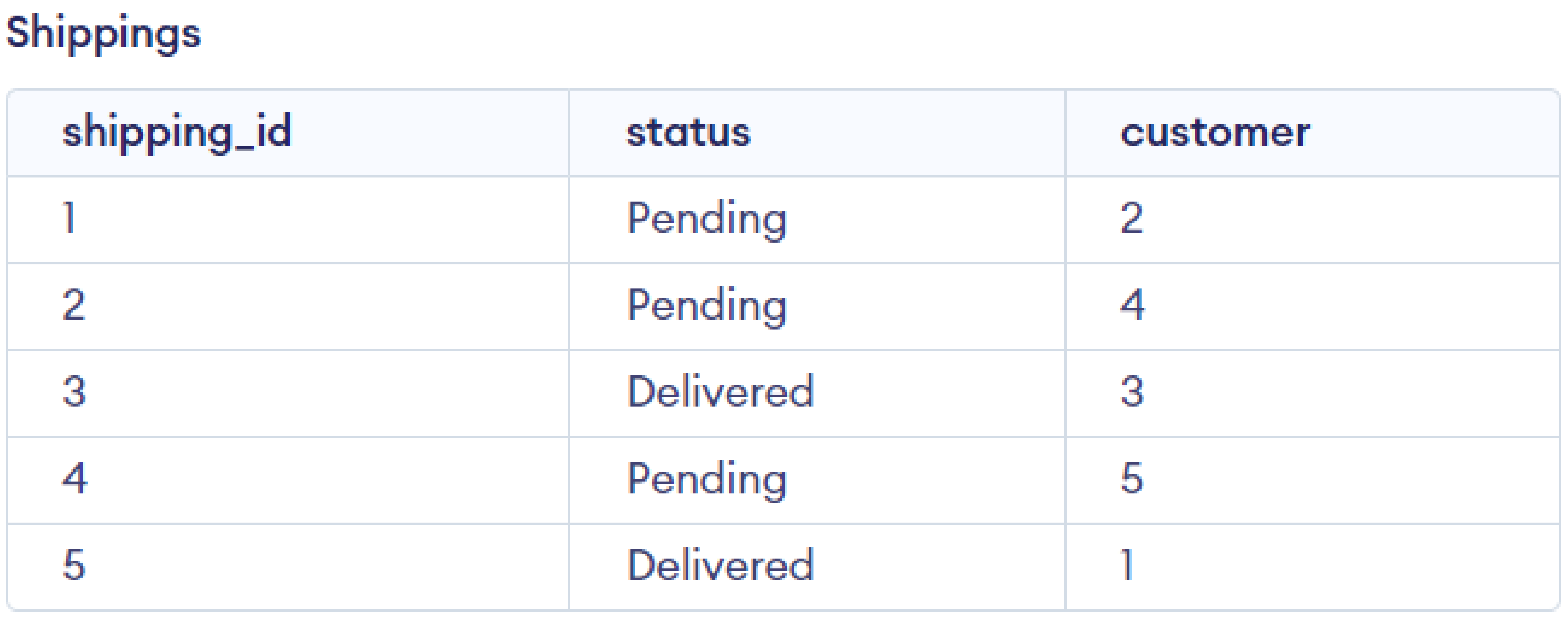
Orders

order_id	item	amount	customer_id
1	Keyboard	400	4
2	Mouse	300	4
3	Monitor	12000	3
4	Keyboard	400	1
5	Mousepad	250	2



Reservation

ReservationID	BookID	MemberID	ReservationDate	Stat
empty				



Shippings

shipping_id	status	customer
1	Pending	2
2	Pending	4
3	Delivered	3
4	Pending	5
5	Delivered	1

Considerations:

- **Indexing Trade-offs**: While indexes improve query performance, they can slow down data modification operations like INSERT, UPDATE, and DELETE because the indexes need to be updated as well. Therefore, indexes should be used on columns that are frequently searched or filtered in queries but not overly indexed to avoid performance degradation.
- **Primary and Foreign Keys**: Primary keys are automatically indexed, ensuring quick lookups and maintaining referential integrity.

Conclusion

The design and implementation of the Library Management System (LMS) database have been carefully crafted to ensure efficiency, scalability, and data integrity. By identifying key entities such as Book, Member, Librarian, Loan, Reservation, and Fine, and establishing clear relationships between them, the system effectively models the real-world operations of a library. The process of normalization has been applied to eliminate redundancy and ensure that data is stored in the most efficient way possible.

Moreover, strategic indexing on critical fields enhances query performance, making the system responsive even as the volume of data grows. The careful balance between normalization and indexing ensures that the database is both easy to maintain and optimized for the most common and essential operations, such as searching for books, managing loans, and handling fines.