Online Bookstore Data Model

1. Overview

This document translates the business rules of an online bookstore into a structured data model. Each rule informs the creation of entities, attributes, and relationships, and by identifying these components, we capture the essential elements and connections within the system. This design balances normalization and efficiency to manage data integrity, flexibility, and scalability.

2. Identified Entities, Attributes, and Keys

Each entity represents a primary component of the bookstore, including `Book`, `Author`, `Publisher`, `Category`, `Customer`, `Review`, `Order`, and `Shipping Method`. Below is a summary of each entity, attributes, and keys.

2.1 Book

- Attributes: BookID (PK), Title, Genre, Price, StockQuantity, ISBN, PublicationDate

- Relationships: Many-to-many with `Author`, one-to-many with `Publisher`, many-to-many with `Category`, one-to-many with `Review`, many-to-many with `Order`

- Explanation: Each book has a unique identifier (`BookID`) and several attributes relevant to bookstore management, such as stock, genre, and ISBN.

2.2 Author

- Attributes: AuthorID (PK), Name, Address, Email, PhoneNumber

- Relationships: Many-to-many with `Book`

- Explanation: An author can write multiple books, and each book can be written by multiple authors, necessitating a many-to-many relationship.

2.3 Publisher

- Attributes: PublisherID (PK), Name, Address, Email, PhoneNumber

- Relationships: One-to-many with `Book`

- Explanation: Each book has one publisher, but a publisher can publish multiple books.

2.4 Category

- Attributes: CategoryID (PK), Name, ParentCategoryID (FK)

- Relationships: One-to-many with `Subcategory`, many-to-many with `Book`

- Explanation: Categories are hierarchical, with parent categories potentially containing multiple subcategories. Each book can belong to multiple categories.

2.5 Customer

- Attributes: CustomerID (PK), Name, Email, Password, Address, PhoneNumber, RegistrationDate

- Relationships: One-to-many with `Order`, one-to-many with `Review`

- Explanation: Each customer can place multiple orders and write reviews for various books.

2.6 Review

- Attributes: ReviewID (PK), Rating, Comment, ReviewDate, CustomerID (FK), BookID (FK)

- Relationships: One-to-many with `Customer` and one-to-many with `Book`

- Explanation: Reviews are unique to a combination of a customer and a book, capturing customer feedback for specific books.

2.7 Order

- Attributes: OrderID (PK), CustomerID (FK), OrderDate, ShippingAddress, TotalAmount, Status, TrackingInfo

- Relationships: Many-to-many with `Book` (through an associative entity `OrderDetail`), one-to-many with `Customer`, one-to-one with `ShippingMethod`

- Explanation: Each order can include multiple books with specified quantities and prices, linking with `Book` through an associative entity.

2.8 Shipping Method

- Attributes: ShippingMethodID (PK), Type, Cost, EstimatedDeliveryTime

- Relationships: One-to-many with `Order`

- Explanation: Each order has a specified shipping method, which determines the cost and delivery time.

3. Relationships and Foreign Keys

The following are the key relationships and the foreign keys involved:

- Books and Authors: Many-to-many, requiring an associative entity `BookAuthor` with foreign keys `BookID` and `AuthorID`.

- Books and Categories: Many-to-many, requiring an associative entity `BookCategory` with foreign keys `BookID` and `CategoryID`.

- Books and Orders: Many-to-many, requiring an associative entity `OrderDetail` with foreign keys `OrderID` and `BookID` to specify quantity and unit price.

- Books and Publishers: One-to-many, with `PublisherID` as a foreign key in `Book`.

- Categories and Subcategories: One-to-many, with `ParentCategoryID` as a self-referencing foreign key in `Category`.

- Customers and Orders: One-to-many, with `CustomerID` as a foreign key in `Order`.

- Customers and Reviews: One-to-many, with `CustomerID` and `BookID` as foreign keys in `Review`.

- Orders and Shipping Method: One-to-one, with `ShippingMethodID` as a foreign key in `Order`.

4. Design Decisions

The design decisions focused on ensuring data integrity and flexibility:

1. Normalization: Each entity is designed to reduce redundancy by normalizing data structures. For example, separating `Author` and `Publisher` details prevents duplicated information across books.

2. Many-to-Many Relationships: Associative entities such as `BookAuthor`, `BookCategory`, and `OrderDetail` handle many-to-many relationships effectively, preserving normalization while allowing complex associations.

3. Hierarchical Categories: The parent-child relationship in `Category` allows for flexible categorization, accommodating various genres and subgenres.

4. Order Details: The `OrderDetail` associative entity manages specific details of each ordered item, providing clear tracking for quantities and unit prices.

5. Customer Reviews: `Review` is linked to both `Customer` and `Book` to accurately capture unique reviews for each book by each customer.

Conclusion

This data model captures the essential components and relationships in an online bookstore. By focusing on normalization, flexibility, and scalability, the design efficiently meets business rules, accommodates many-to-many and hierarchical relationships, and ensures organized data flow, enabling robust database operations for managing the bookstore's diverse needs.