Functional Requirements for a Library Management System

The following are the functional requirements for a Library Management System:

1.  Ability to add and remove books from the library

2.  Ability to search for books in the library by title, author, or ISBN

3.  Ability to check out and return books

4.  Ability to display a list of all books in the library

5.  Ability to store and retrieve information about library patrons, including their name and ID number

6.  Ability to track which books are currently checked out and when they are due to be returned

7.  Ability to generate reports on library usage and checkouts

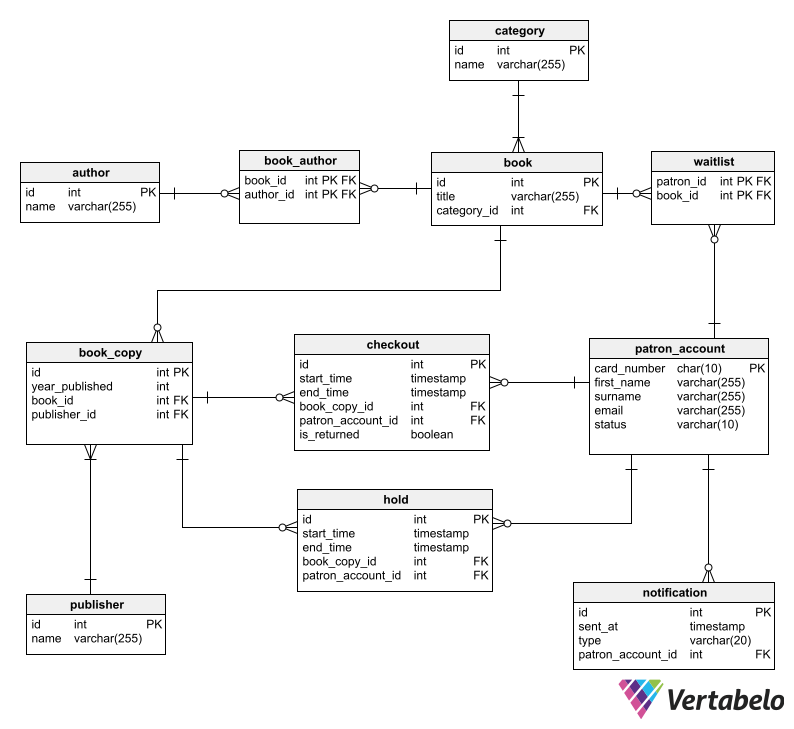
Entities

Book: Each book in the library is represented by a Book entity.

Library: The collection of books in the library is represented by the Library entity.

Patron: Each library patron is represented by a Patron entity.

Checkout: Each checkout transaction is represented by a Checkout entity.



class Library {

    String name;

    Address location;

    List<BookItem> books;

}

class Book {

    String uniqueIdNumber;

    String title;

    List<Author> authors;

    BookType bookType;

}

class BookItem extends Book {

    String barcode;

    Date publicationDate;

    Rack rackLocation;

    BookStatus bookStatus;

    BookFormat bookFormat;

    Date issueDate;

}

class Address {

    int pinCode //ZipCode

    String street;

    String city;

    String state;

    String country;

}

public enum BookType {

    SCI\_FI, ROMANTIC, FANTASY, DRAMA;

}

public enum BookFormat {

    HARDCOVER, PAPERBACK, NEWSPAPER, JOURNAL;

}

public enum BookStatus {

    ISSUED, AVAILABLE, RESERVED, LOST;

}

class Rack {

    int number;

    String locationId;

}

class Person {

    String firstName;

    String lastName;

}

class Author extends Person {

    List<Book> booksPublished;

}

class SystemUser extends Person {

    String Email;

    String phoneNumber;

    Account account;

}

class Member extends SystemUsers {

    int totalBookCheckedOut;

    Search searchObj;

    BookIssueService issueService;

}

class Librarian extends SystemUsers {

    Search searchObj;

    BookIssueService issueService;

    public void addBookItem(BookItem bookItem);

    public BookItem deleteBookItem(String barcode);

    public BookItem editBookItem(BookItem bookItem);

}

class Account {

    String userName;

    String password;

    int accountId;

}

class Search {

    public List<BookItem> geBookByTitle(String title);

    public List<BookItem> geBookByAuthor(Author author);

    public List<BookItem> geBookByType(BookType bookType);

    public List<BookItem> geBookByPublicationDate(Date publicationDate);

}

class BookIssueService {

    Fine fine;

    public BookReservationDetail getReservationDetail(BookItem book);

    public void updateReservationDetail(BookReservationDetail bookReservationDetail);

    public BookReservationDetail reserveBook(BookItem book, SystemUser user);

    public BookIssueDetail issueBook(BookItem book, SystemUser user);

    // it will internaly call the issueBook function after basic validations

    public BookIssueDetail renewBook(BookItem book, SystemUser user);

    public void returnBook(BookItem book, SystemUser user);

}

class BookLending {

    BookItem book;

    Date startDate;

    SystemUser user;

}

class BookReservationDetail extends BookLending {

    ReservationStatus reservationStatus;

}

class BookIssueDetail extends BookLending {

    Date dueDate;

}

class Fine {

    Date fineDate;

    BookItem book;

    SystemUser user;

    public double calculateFine(int days);

}