Scenario

WhatABook, a small used book store has decided to expand their presence by offering customers a solution they can install on their personal computers. WhatABook would like to establish a console application that allows customers to browse their in-store book listing, add books to their Wishlist, and view store hours and locations. Right now, they only have one location, but the owner, Hugo Dopsen, is emphatic about expanding. Hugo would like the new application's interface to be as simplistic as possible. Most of their users will be middle aged customers with minimal computer experience. Customers will need to register for a free account by providing their email address, first, and last name. The program should support options for viewing a list of their in-store books, adding books to their Wishlist, and viewing store hours and locations.

Biz rules

* a User can have many Wishlist books.
* many Book(s) can be added to a User(s) Wishlist.
* many Book(s) can be added to one Wishlist.
* WhatABook has many Book(s).
* many Book(s) are owned by WhatABook.
* WhatABook can have many Location(s).

User Interface Requirements

Menu

1. View Books
2. View Store Locations
3. My Account
   * Prompt the user to enter a user\_id
     1. 1, 2, or 3
     2. Note: before a user can access their account, they must enter a valid user\_id.
   * Wishlist Menu
     1. Wishlist
     2. Add Book
        + Display the available books
          - The output will show the book\_id, book\_name, author, and details
        + Prompt the user to select a book to add to their wishlist
          - To add a book to the user's Wishlist, the insert statement will need the book\_id and user\_id
     3. Main Menu
        + This option takes users back to the main menu
4. Exit Program
   * Exits the program

Database Guidelines

* You do not need to create a table for WhatABook because the database will only support one company and it is inferred that Book(s) are owned by WhatABook.
* You will need to create a separate table for WhatABook’s hours and locations because at some point they will expand to multiple locations.
* There should only be four tables.
* The courses GitHub repository has an initialization script to compare your code against mine. But, only use this as a last resort. To truly learn these topics, you will need to make an effort and try to complete the project on your own.

Query Guidance

* To display a user’s Wishlist you will need to use two INNER JOINs to combine the user and book tables.
* Store, Book, Wishlist, User.
* To add a book to the users wishlist you will need to capture the selected book\_id and user\_id.
* To view a list of books not in the users wishlist you will need to use the NOT IN operator with a nested query.
* WHERE book\_id NOT IN ( SELECT book\_id FROM wishlist WHERE user\_id = ).
* If you get stuck, the course's GitHub repository has an SQL script that includes all of the SQL queries. But, again, only use this as a last resort. To truly learn these topics, you will need to make an effort and try to complete the project on your own.

User Interface Guidance/Hints

* Create a method to for “show\_menu()”
* Create a method for “show\_books(\_cursor)”
* Create a method for “show\_locations(\_cursor)”
* Create a method for “validate\_user()”
* Create a method for “show\_account\_menu()”
* Create a method for “show\_wishlist(\_cursor, \_user\_id)”
* Create a method for “show\_books\_to\_add(\_cursor, \_user\_id)”
* Create a method for “add\_book\_to\_wishlist(\_cursor, \_user\_id, \_book\_id)”
* Create a method to display the account menu
* Use variables to capture the user’s entry for user\_id
* Use variables to capture the user’s entry for book\_id
* If you get stuck, the courses GitHub repository has the final solution to compare your code against. But, again, only use this as a last resort. To truly learn these topics, you will need to make an effort and try to complete the project on your own.