Look Inna Book

COMP 3005 Project Report
Due December 10, 2021
Instructor Ahmed El-Roby

By Sara Shikhhassan 101142208,
Sam Al Zoubi 101140949

Table of Contents

1 Conceptual Design

1.1 ER Diagram

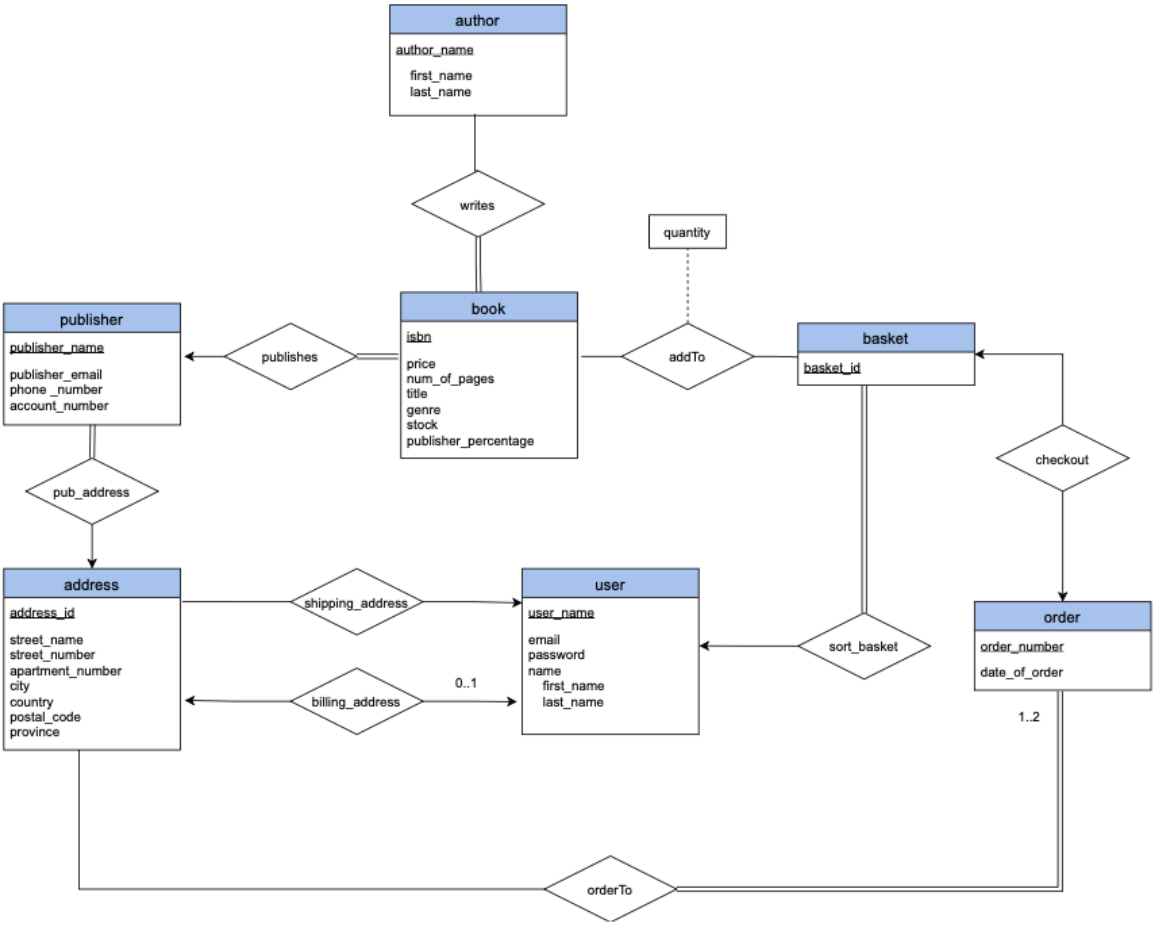

Figure 1. ER Diagram

1.2 Entity Sets

- Address
- The entity set $address$ represents the street address of the $user$ in which the user's purchase would be delivered to and the payment would be billed to. It also holds the $publisher's$ address. The $address$ entity set has a primary key address_id to distinguish between addresses as users can have the same address' thus, making every user's address to be unique through the address id.

- User
- The entity set $user$ represents a bookstore user in which a customer would have made an account online. It has attributes that hold the customer's information such as username, email, password and full name composed of first and last name. The $user$ entity set has a primary key user_name to ensure a unique username to every online customer of the bookstore as some customers might have the same email, password and full name.

- Basket
- The entity set $basket$ represents the user's shopping cart in the online bookstore. It's attribute basket_id is the primary key, which will be a unique ID for every shopping cart in the online store.

- Order
- The entity set $order$ represents the checked out shopping cart, which holds the user's order. It also holds the tracking number of a user's order using the order number. The Primary key of this entity set is order_number as every order placed will have a unique order number, and this unique order number can be used to track their shipped order.

- Book
- The entity set $book$ represents books in the online store. It's primary key is the isbn attribute which is a unique international book number allocating to an edition or variation of the publication of the set of books. Even if a book does not belong to a series, it will still have a unique isbn. This will allow customers of the online bookstore to easily find a book through this number. The other attributes of the $book$ entity set is price, number of pages, title, genre, and stock can also be used to filter out searches for the customers when searching for books in the online store. It's other attribute publisher percentage is used to keep track of the percentage owned to the publisher through the book sales.

- Author
- The entity set $author$ represents the author of a published book. It's primary key is the author_name attribute which is composed of the author's first and last name. The assumption here is that no two authors can have the same name.

- Publisher
- The entity set $publisher$ represents the publisher of the books found in the online store. It's primary key is the publisher_name entity set which makes every published book carry a unique publisher name. It's other attribute is account_number which is the publisher's bank account number.

- Banking
- The entity set $banking$ represents the bank account associated with a publisher. It's primary key is account_number corresponding to the unique account number of the publisher.

1.3 Relationship Sets

- pub_banking
- The pub_banking relation relates the $publisher$ with the $banking$. This allows the publisher to receive a percentage of their published book sales through their banking account. The attributes of this relation is the $banking$ primary key, account_number, and the $publisher$primary key, publisher_name. The primary key of this relation is the attribute account_number as a publisher can have many bank accounts, but a bank account can only belong to one publisher. Therefore, this is a many to one relationship. This relation is a total relation from both sides because a publisher and banking must participate with each other and exist.

- pub_address
- The pub_address relation relates the $publisher$ with the $address$. This allows the owners of the bookstore to store the publisher's address information as per the project requirement. A publisher can have many addresses but an address can only have one publisher corresponding to it. Thus, the primary key of this relation is publisher_name. It's other attribute is address_id which comes from the primary key of the $address$ entity set. This relation is a many to one relation and is total on the $publisher$ side. A publisher must participate meaning a publisher must have an address

- writes
- The writes relation relates an $author$ to the $book$ they have written. Since an author can write many books and many books can be written by the same author, the primary key of this relation is the isbn attribute. It's other attribute is the author_name which is the primary key of the $author$ entity set. This relation is many to many, and is total on the book side because a book cannot exist if it is not written by an author.

- publishes
- The publishes relation relates a $book$ to it's $publisher$. Since a publisher can publish many books, but a book can only be published by one publisher, the primary key of this relation is the primary key of the $book$ entity set, isbn. Its other attribute is the publisher_name which comes from the $publisher$ primary key. This relation is many to one and is total on the book side because a book must be published to exist.

- shipping_address
- The shipping_address relation relates a user with the address used to ship their order to. The primary key of this relation is address_id because a user can have only one address to ship their order to, but a shipping address can have multiple users associated with it. It's other attribute is the primary key of the $user$ entity set, user_name. This relation is many to one and partial because if a user does not checkout, then the user does not need to participate with the address entity set.

- billing_address
- The billing_address relation relates a user with the address used to bill their order to. The primary key of this relation is address_id because a user can have one billing address if different than their shipping address, or zero billing addresses. This relation is one to one, and is partial because the entities may not participate in any of the relationships in the relationship set. In this case, if the user does not have a billing address and the billing address does not have a user, then they do not participate with each other.

- orderTo
- The orderTo relation relates an order to its corresponding address. The primary key in this relation is order_number because an order can only be delivered to one address but an address can have multiple orders. Therefore, this relation is many to one and is total on the order side because an order must participate with the address in order to be shipped.

- sort_basket
- The sort_basket relation relates a user to their shopping basket. The primary key in this relation is basket_id because every user can have multiple baskets over time, but the baskets can only belong to one user at a time. This relation is many to one and is total on the basket side because a basket must have a user to exist, but a user does not need to have a basket.

- addTo
- The addTo relation relates a book to the basket. This relation allows a user to add a given book to their basket. The primary keys in this relation are isbn, basket_id.

2 Reduction To Relation Schemas

user(<u>username</u>, email, password, first_name,last_name)
book(<u>isbn</u>, publisher_name, price, num_of_pages, title, genre, stock, publisher_percentage)
address(<u>address_id</u>, street_name, street_number, apartment_number, city, country, postal_code, province)
publisher(<u>publisher_name</u>,account_number, address_id, publisher_email, phone_number)
author(<u>first_name</u>, <u>last_name</u>)
basket(<u>basket_id</u>, user_name)
order(<u>order_number</u>, date_of_order)
writes(<u>first_name</u>,<u>last_name</u> ,<u>isbn)</u>
shipping_address(<u>address_id</u>, user_name)
billing_address(<u>address_id</u>, user_name)
addTo(<u>isbn</u>, <u>basket_id</u>, quantity)
checkout(<u>order_number</u>, basket_id)
orderTo(<u>address_id,</u> <u>order_number</u>)

2.1 Normalization of Relation Schemas

- user(<u>username</u>, email, password, first_name,last_name)

F = { username→email,password,first_name,last_name}

The user relation is already in good form. The attribute closure of $(username)^+$ is {email,password,first_name,last_name} and since this is the only functional dependency for this relation and it is a superkey, it is in BCNF.

- book(<u>isbn</u>, publisher_name, price, num_of_pages, title, genre, stock, publisher_percentage)

F = { isbn→publisher_name,price, num_of_pages, title, genre, stock, publisher_percentage}

The book relation is already in good form. The attribute closure of $(isbn)^+$ is {publisher_name,price, num_of_pages, title, genre, stock, publisher_percentage} and since this is the only functional dependency for this relation and isbn is a superkey, it is in BCNF.

- address(<u>address_id</u>, street_name, street_number, apartment_number, city, country, postal_code, province)

F = {address_id →street_name, street_number, apartment_number, city, country, postal_code, province}

The address relation is already in good form. The attribute closure of $(address\_id)^+$ is {street_name, street_number, apartment_number, city, country, postal_code, province} and since this is the only functional dependency for this relation and address_id is a superkey, it is in BCNF.

- publisher(<u>publisher_name</u>, account_number,address_id, publisher_email, phone_number)

F = {publisher_name →address_id, publisher_email, phone_number, account_number
publisher_email →publisher_name
phone_number → publisher_name
account_number → publisher_name}

The publisher relation is already in good form. The attribute closure of $(publisher\_name)^+$ is {address_id, publisher_email, phone_number, account_number}. So we know that publisher_name is a superkey and for all other functional dependencies, if you apply transitivity with the first functional dependency (publisher_name →address_id,

publisher_email, phone_number, account_number). Attributes publisher_email, phone_number and count_number will also be superkeys and therefore it is in BCNF.

- author(first_name, last_name)

F ={ first_name,last_name →first_name,last_name}

The author relation is already in good form. The only functional dependency present is a trivial functional dependency and therefore this relation is in BCNF.

- basket(basket_id, user_name)

F ={ basket_id→ user_name}

The basket relation is already in good form. The attribute closure of $(basket_id)^+$is {basket_id, user_name} and therefore basket_id is a superkey and the relation is in BCNF.

- order(order_number, date_of_order)

F ={ order_number → date_of_order}

The order relation is already in good form. The attribute closure of $(order_number)^+$is {order_number, date_of_order} and therefore order_number is a superkey and the relation is in BCNF.

- writes(first_name,last_name,isbn)

F = {first_name, last_name → first_name, last_name
Isbn → isbn}

The writes relation is already in good form. Both functional dependencies are trivial functional dependencies and therefore the relation is in BCNF.

- shipping_address(address_id, user_name)

F = { address_id → user_name}

The shipping_address relation is already in good form. The attribute closure of $(address_id)^+$is {address_id, user_name} and therefore address_id is a superkey and the relation is in BCNF.

- billing_address(<u>user_name</u>, address_id)

F = {user_name → address_id}

The billing_address relation is already in good form. The attribute closure of $(user_name)^+$is {address_id, user_name} and therefore user_name is a superkey and the relation is in BCNF.

- addTo(<u>isbn</u>, <u>basket_id</u>, quantity)

F ={ isbn, basket_id → quantity}

The addTo relation is already in good form. The attribute closure of $(basket_id, isbn)^+$is {isbn, basket_id, quantity} and therefore (isbn,basket_id) is a superkey and the relation is in BCNF.
Note: $(basket\_id, isbn)^+$ is also a candidate key because $(basket\_id)^+$= $(isbn)^+$ = {isbn, basket_id, quantity} and so individually both basket_id and isbn are superkeys.

- checkout(<u>order_number</u>, basket_id)

F = {order_number → basket_id}

The checkout relation is already in good form. The attribute closure of $(order_number)^+$is {order_number, basket_id} and therefore order_number is a superkey and the relation is in BCNF.

- orderTo(<u>address_id, order_number</u>)

F = {address_id, order_number → address_id, order_number}

The orderTo relation is already in good form. The only functional dependency present is a trivial functional dependency and therefore this relation is in BCNF.

2.2 Database Schema Diagram

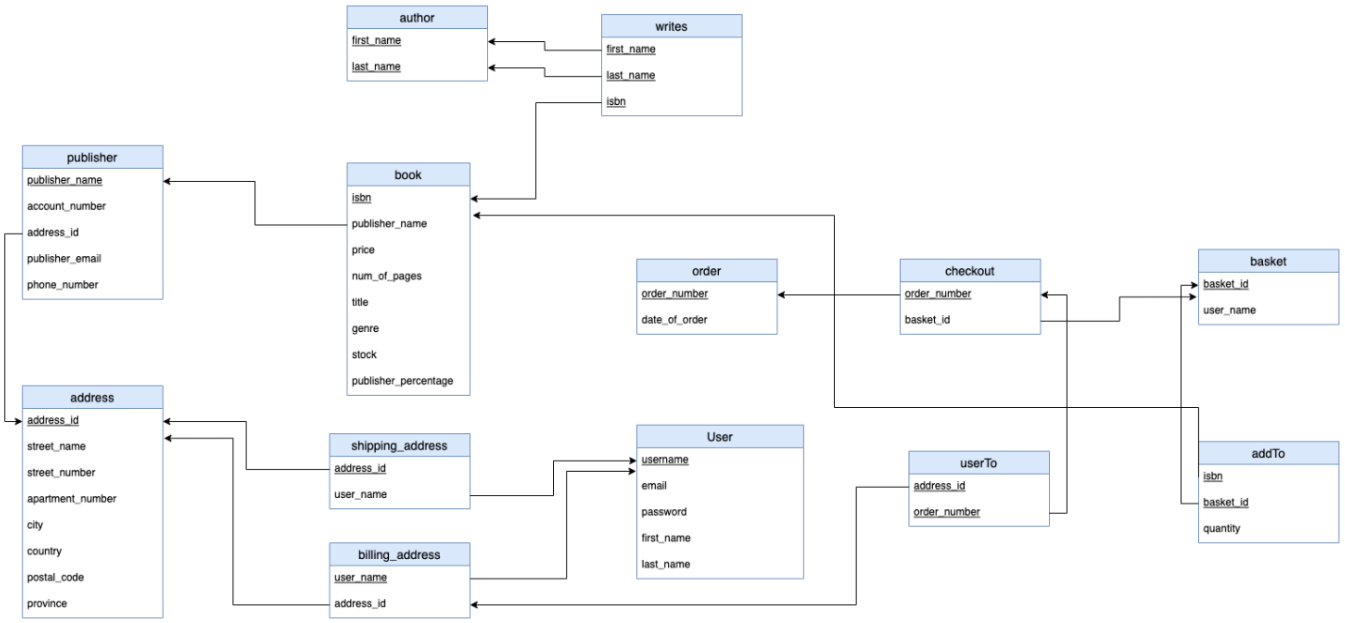

Figure 2: Database Schema Diagram

3 Implementation

3.1 Work Flow Diagram

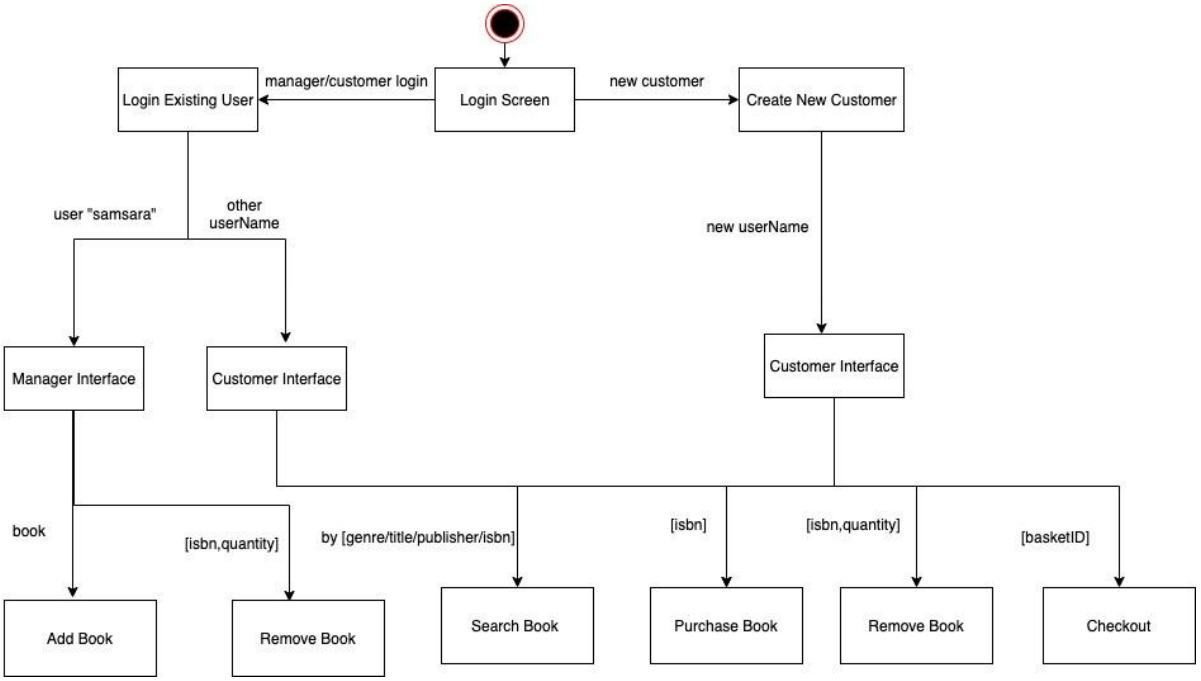

Figure 3: Workflow Diagram

3.2 Login Screen

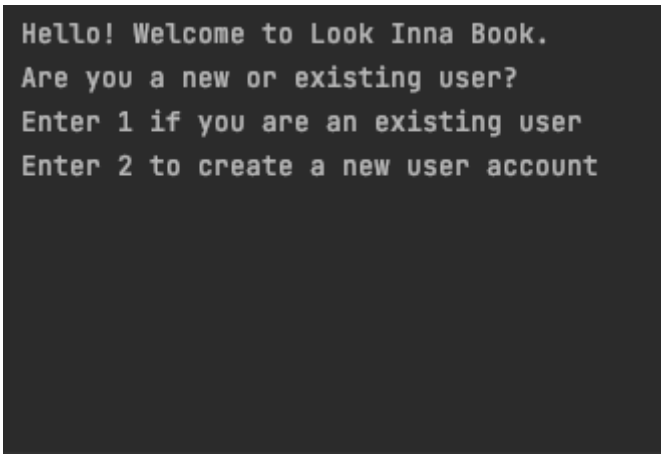

Figure 4: Initial Login Screen

The login screen interface is the initial console output the user is presented with when the user runs the program and accesses the store. The user must choose if they are an existing user or a new one. If the user is a new user, then the program will run the “New User Case” scenario and add the new customer to the database with the corresponding information. If the user already exists, then the user may log in either as a manager or as a customer. To be a

manager the manager's username must be "samsara" as there may be only one manager in this system, with set credentials.

3.2.1 Case New User Login Screen

```
Hello! Welcome to Look Inna Book.
Are you a new or existing user?
Enter 1 if you are an existing user
Enter 2 to create a new user account
2
Enter a username:
samzoubi
Enter a password:
kellykelly
Enter first name:
sam
Enter last name
zoubi
Enter email address:
samalzoubi@cmail.carleton.ca
```

Figure 5: Creating a New User

If the user enters the system as a new user then the user will be directed to answer a few questions concerning the users credentials in order to be added to the system and use it's services. After the user has entered all the required information correctly, the user will be added to the database.

3.2.2 Customer Interface Login

```
Welcome Customer!

Enter 0 to find a book, 1 to purchase a book, 2 to proceed to checkout.
```

Figure 6: Interface Login

This is the login screen of a customer. The customer has the option to either find books based on multiple factors such as the genre of the book, the title, the publisher name and the book's isbn. The customer also has the option to purchase a book based on the book's isbn and to checkout a cart if the user wishes.

3.2.3 Manager Login Interface (User samsara)

```
Welcome back, manager.

Enter 0 to add books to the store, enter 1 to delete books from store
```

Figure 7: Login for Managers

This is the login screen of a manager. The manager has the option to add books to its store or remove books from the store.

3.3 Search Book By Genre Senario

```
Welcome Customer!

Enter 0 to find a book, 1 to purchase a book, 2 to proceed to checkout.
0
Enter 0 to search by ISBN, 1 to search by title, 2 to search by genre, 3 to search by publisher name, 4 to return to main menu
2
Enter the book's genre
```

Figure 8: Searching by Genre

This is an example scenario in regards to the customer interface, if the customer wishes to search for a book based on genres. The output of this display will be all the books that are

present in the database with the given genre and the user will either be given the option to purchase the book, to continue looking or to return to the main menu.

3.4 Purchase Book Senario

```
Enter 0 to purchase book, 1 to continue browsing, 2 for main menu
2
Enter 0 to find a book, 1 to purchase a book, 2 to proceed to checkout.
1
What is the book ISBN?
5555
How many would you like to purchase?
4
```

Figure 9: Purchasing a Book

This is an example scenario in regards to the customer interface of the system, if the customer wishes to purchase a book given a book's unique ISBN number. The customer will be asked to clarify how many copies of the book the customer wishes to purchase. The program will search if the book exists in the database and if the book does not exist then the customer shall be notified via a print statement.

3.5 Add Book Senario

```
Enter 0 to add books to the store, enter 1 to delete books from store
0
Enter book isbn:
5555
Enter amount to add:
7

Please enter the book's isbn, title, number of pages, price, publisher earnings, stock, genre, and publisher name

Separated by commas ','
```

Figure 10: Adding a Book

This is an example scenario in regards to the manager interface, of what the manager will be asked for when requesting to add more books to the store, the manager must enter all the relevant book,publisher, and author information related to the book that will be added in order to successfully update the database and add the new corresponding entries.

3.6 Remove Book Senario

```
Welcome back, manager.

Enter 0 to add books to the store, enter 1 to delete books from store
1
Enter book isbn:
4444
Enter amount to remove:
5
```

Figure 11: Removing a book

This is an example scenario in regards to the manager interface, of when a manager wishes to remove a book from the store. The manager will be prompted to enter the book's isbn and amount to remove. Once the manager has done so, the database will be successfully updated and those corresponding entries will be removed.

4 GitHub Repository

The following link can be used to access the team member's GitHub Repository: https://github.com/samizoubi/COMP-3005-LookInnaBook

Appendix I

Our group's availability from 9am to 5pm for a 20 minutes demonstration of our work on December 11th is from 11:00am-11:20am, 12:00pm-12:20pm, and 1:00pm-1:20pm.