BookHaven

Your escape from reality, one book at a time

Transaction Analysis

Non Conflicting transactions

- Book Browsing

  Since browsing our collection only requires a *read* query, this will not be a conflicting action. It is done using the following query

```
SELECT * FROM book
```

- Login

  Login only requires checking whether the input credentials are valid or not and is done using the following query

```
SELECT * FROM Admin WHERE username=user_name AND passkey=pass_key
SELECT * FROM Customer WHERE username=user_name AND passkey=pass_key
```

- Cart

  /cart display the logged in user's shopping cart, done with

```
SELECT cart_id FROM cart WHERE customer_id = customerId
SELECT book.book_id, book.book_name, book.author_name, book.genre, book.price, cart_items.count
FROM cart_items
JOIN book ON cart_items.book_id = book.book_id
WHERE cart_items.cart_id = cartId
```

- Admin Dashboard

  displays the admin dashboard, which help them manage books, customers, and orders; another *read*

```
SELECT * FROM book
SELECT * FROM customer
SELECT * FROM orders
```

Conflicting Transactions

- New User Registration

  There may be a situation where two different users register with same username. This may lead to a conflict. To prevent this, we have locked tables from updating for other users. This will prevent another session from updating any table while a user is registering.

```
START TRANSACTION
LOCK TABLES book READ, orders READ, cart_items READ, cart READ, customer WRITE, reviews READ, admin READ, browses READ
INSERT INTO customer (first_name, last_name, pincode, address, contact, date_of_birth, email, passkey, amount, owned_books) V
COMMIT
UNLOCK TABLES
```

- Order Checkout

  When two users are simultaneously checking out, tables will be updated with different values in different session at the same time. This may lead to *data_curruption* and *data_inconsistency*. So we ensure that only one user checks-out at a time. This transaction is also made **ACID** through the following

```
START TRANSACTION
START TRY
    LOCK TABLES book WRITE, orders WRITE, cart_items WRITE, cart WRITE, customer READ, reviews READ, admin READ, browses READ
    INSERT INTO orders (quantity, total_cost, book_id, customer_id) VALUES (count, price, bookId, userId)
    DELETE FROM cart_items WHERE book_id = bookId AND cart_id = cartId
    UPDATE book SET stock = stock - count WHERE book_id = bookId
    COMMIT
    UNLOCK TABLES
END TRY
START CATCH
    ROLLBACK
END CATCH
```