Deadline-6

DBMS

INCLUSI-SHOP

Yashovardhan Singhal(2022591)

Conflicting Transactions

-- -- Transaction 1: Update quantity to 30

Start transaction;

UPDATE product

SET quantity = quantity+30

WHERE product_id=1;

Commit;

```
mysql> UPDATE product
    -> SET quantity = 30
    -> WHERE product_id=1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

Transaction 2: Update quantity to 40

Start transaction;

UPDATE product

SET quantity = quantity+40

WHERE product_id=1;

commit;

```
mysql> use inclusishop;
Database changed
mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE product
    -> SET quantity = 40
    -> WHERE product_id=1;
ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transaction
mysql>
```

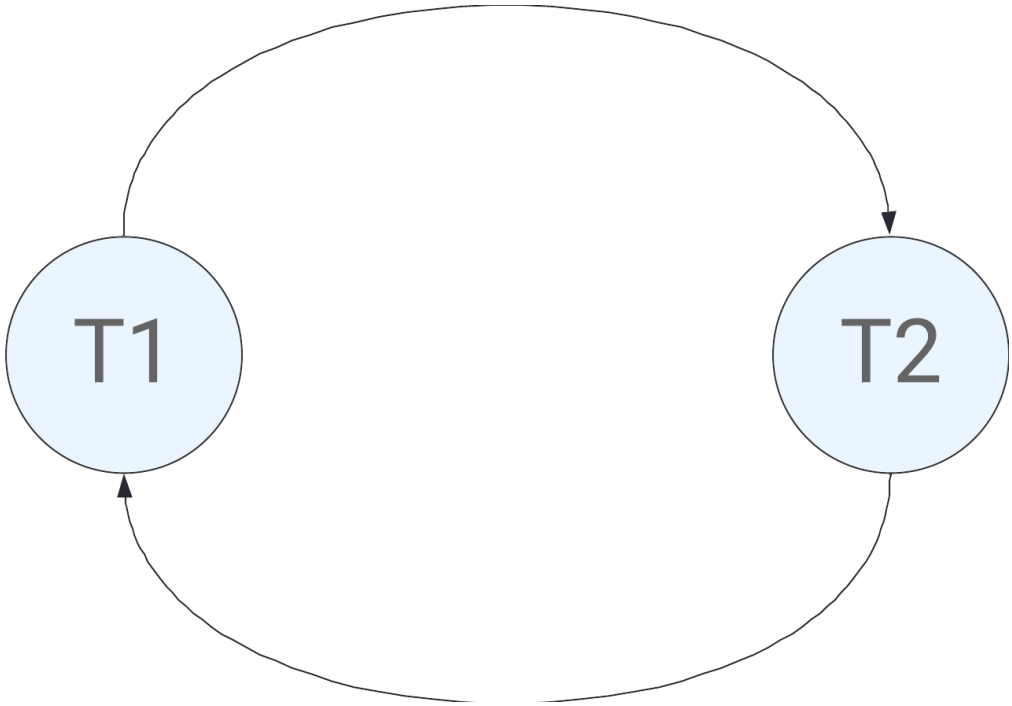
Explaination

In this example, we have two SQL UPDATE statements that are attempting to modify the quantity of the product_id=1.

Depending on the isolation level and concurrency control mechanism of the database system, executing these two transactions

concurrently could lead to a conflict.

T1	T2
Read (Quantity_Added)	
Read (Inventory)	
	Read(Quantity_Added)
Inventory=Inventory+Quantity_Added	
	Read (Inventory)
Write(Inventory)	
	Inventory=Inventory+Quantity_Added
	Write(Inventory)
Commit	
	Commit



Non-conflicting Transactions

----------NON CONFLICTING TRANSACTIONS--------

----------Pair 1----------

START TRANSACTION;

UPDATE Product SET Price = new_price WHERE ProductID = product_id;

COMMIT;

START TRANSACTION;

UPDATE Product SET Description = new_description WHERE ProductID = product_id;

COMMIT;

Explanation

In this case, Transaction 1 updates the Price attribute of the Product table, while Transaction 2 updates the Description attribute.

Since they're updating different columns, they won't conflict with each other, allowing them to be executed concurrently.

----------Pair2----------

INSERT INTO NGOs (Name, Description, Email, Address, ContactNumber)

VALUES ('EmpowerAbility Foundation', 'Empowering people with disabilities to achieve their full potential through education and employment opportunities.', 'info@empowerability.org', '789 Freedom Street, Delhi, New Delhi', '+9876543216');

Commit;

DELETE FROM ProductReview WHERE Review_ID = 7;

Commit;

Explanation

These transactions involve different tables and operations, ensuring that they don't conflict with each other.

Features

The script provided includes a substantial amount of SQL and Python code, which outlines a robust system involving user and admin interactions with a database. Let's break down the functionalities provided for both user and admin roles as detailed in your code:

Admin Functionalities

1. **Admin Login**: Admins can log in using their credentials. After successful login, they can access various management functionalities.

2. Product Management:
   - Add Product: Admins can add new products to the database, specifying details such as name, description, supplier, category, and quantity.

3. NGO Management:
   - Add NGO: Admins can register new NGOs into the system, providing details like name, description, email, address, and contact number.

4. Employee and Delivery Agents Management:
   - Display Delivery Workers' Information: Admins can view general information about delivery agents, including their IDs, names, ages, genders, cities, and contact details.

5. User Management:
   - Display Active Users: Shows the number of active users.
   - Display Users with at Least One Purchase: This function provides details of users who have made at least one purchase, helping in identifying active and potentially premium customers.

6. Sales and Inventory Analysis:
   - Inventory Analysis: Admins can examine the inventory, including products, their descriptions, and supplier information.
   - View Product Sales Stats: Provides statistics on product sales, helping admins understand sales performance.

7. Logging Out: Admins can logout from their session.

User Functionalities

1. User Sign-Up and Login:

- Sign Up: New users can register by providing personal and contact details. Validation checks are performed for phone numbers and email formats.
- Login: Users can log into the system using their credentials.

2. Product Interaction:
- View Products: Users can browse products by categories and see details like product names and IDs.
- Add to Cart: Users can add selected products to their cart with specified quantities.

3. Shopping Cart Management:
- View Cart: Users can view the products added to their cart.

4. Order Management:
- Functions related to viewing, adding, or ordering products from the cart might be expected here, although specific details are not provided in the scripts.

5. Logging Out: Allows users to log out of the system.

Additional Notes

- Error Handling: The system includes error handling for database operations, ensuring robust operation and feedback on duplicate entries, format errors, etc.
- **Security Concerns**: Passwords in the script use checks for complexity, improving security.
- Database Interaction: Uses MySQL for database operations involving products, users, orders, reviews, and NGOs.

Conclusion

The functionality for both roles is well-defined, with admins managing various aspects of the platform and users interacting with products and managing their profiles and orders. This setup supports an e-commerce or service platform focusing on inclusive products, potentially for disabled individuals, given the context of NGOs and specialized products.

Contributions -

Yashovardhan Singhal