ASSIGNMENT – 6

DBMS
ACCESSI–SPHERE

PALAK BHARDWAJ (2022344) - Transactions and admin Functionalities
SWARNIMA PRASAD (2022525) - Admin and User functionalities

E-R diagram link here

1. Conflicting Transaction Pair 1

The database can be accessed by multiple users and admins, and cases may arise when such an allowed access leads to inconsistency.

Transaction 1 -

Admin 1 is trying to to add the quantity -

START TRANSACTION;

UPDATE Product

SET Quantity = Quantity - 15

WHERE Product_ID = 2;

COMMIT;

```
mysql> UPDATE Product
    -> SET Quantity = Quantity - 20
    -> WHERE Product_ID = 2;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

Transaction 2 -

Admin 2 is trying to decrease the quantity -

START TRANSACTION;

UPDATE Product

SET Quantity = Quantity - 15

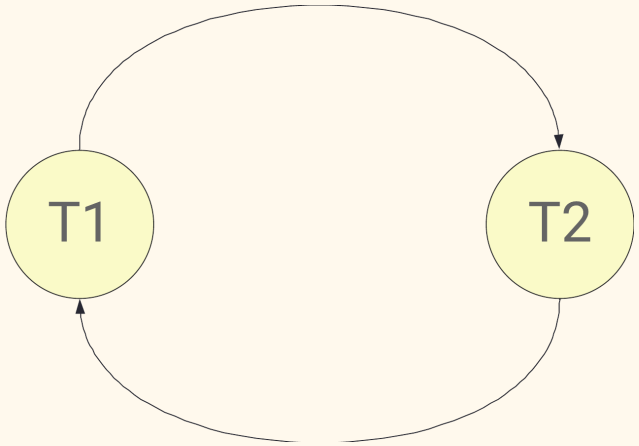
WHERE Product_ID = 2;

-- Commit;

```
mysql> UPDATE Product
    -> SET Quantity = Quantity - 15
    -> WHERE Product_ID = 2;
ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transaction
mysql>
```

This is a schedule of the first transaction mentioned as T1 and the second transaction mentioned as T2.

T1	T2
Read (Quantity_Subtracted)	
Read (Inventory)	
	Read(Quantity_Subtracted)
Inventory=Inventory-Quantity_Subtracted	
	Read (Inventory)
Write(Inventory)	
	Inventory=Inventory-Quantity_Subtracted
	Write(Inventory)
Commit	
	Commit



Operation Types: Both transactions perform write operations on the same data (quantity field of product_id=1).

Data Item: The specific data item being modified (quantity for product_id=1) is the same in both transactions.

Conflict Types
Write-Write Conflict (Direct Conflict): This occurs when two transactions attempt to write to the same data item. The last write overwrites the previous writes, potentially leading to inconsistent data or lost updates depending on the sequence of transaction commits.

2. Conflicting Transaction Pair 2

Transaction 1 -

This admin may want to delete a product from the table to show its discontinuity.

START TRANSACTION;

delete from Product

where Product_ID=11;

Commit;

Transaction 2

Another admin may want to insert a new product into the table, but it is conflicting because two admins would be accessing the same data, leading to inconsistency.

Start Transaction;

INSERT INTO Product (Name, Description, Company_name, Category, Status, SupplierID, Quantity, Price)

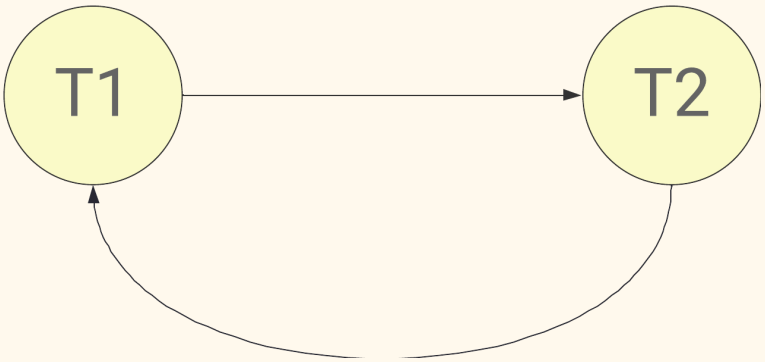
VALUES

('Cochlear Implant', 'An electronic medical device that replaces the function of the damaged inner ear', 'HearCo', 'Hearing', true, 3, 20, 50000);

Commit;

T1	T2
Read (Product)	
	Write (Product)
Read (Product)	
	Read (Product)

Precedence Graph



Acid properties being violated-

Consistency

If both transactions are executed simultaneously, it can lead to inconsistency because one transaction is removing a product while the other is adding a new one. This could result in a state where the database is neither in its initial state nor in its desired state.

Transaction 1 starts

```
mysql> INSERT INTO Product (Name, Description, Company_name, Category, Status, SupplierID, Quantity, Pric
    -> VALUES
    -> ('Cochlear Implant', 'An electronic medical device that replaces the function of the damaged inner
, 'Hearing', true, 3, 20, 50000);
Query OK, 1 row affected (0.00 sec)

mysql>
```

Transaction 2 shows : Lock wait Timeout Exceeded

```
mysql> use accessisphere;
Database changed
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> delete from Product
    -> where Product_ID=11;
ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transaction
mysql>
```

This happens when both transaction run simultaneously in concurrent sessions.

2. Non-Conflicting Transactions

Start Transaction;

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

VALUES ('New Vision', 11, 'An organization promoting eye health', 'newvision@gmail.com', '123 Vision Road, New Delhi, 110011', '1111234567');

Commit;

Start Transaction;

DELETE FROM PremiumMember WHERE UserID = 2;

Commit;

Start Transaction;

SELECT * FROM DeliveryWorker WHERE Salary > 20000;

Commit;

Start Transaction;

SELECT * FROM OrderTable WHERE UserID = 1;

Commit;

All 4 of them are non-conficiting and even if run together would not result into an error

3.Additonal Transaction implemented via cli

Updates order statues for orders which are older than a month

```
def update_order_statuses():
    try:
        update_query = """
        UPDATE OrderTable
        SET OrderStatus = 'Under Review'
        WHERE OrderDate <= %s AND OrderStatus = 'Pending'
        """
        one_month_ago = datetime.now() - timedelta(days=30)
        one_month_ago = one_month_ago.strftime('%Y-%m-%d')  # Format it to match SQL date format
        mycursor.execute(update_query, (one_month_ago,))

        # Commit the changes
        cnx.commit()
        print(f"Updated orders: {mycursor.rowcount} rows affected.")
    except mysql.connector.Error as err:
        print("Error occurred:",err)
        cnx.rollback()  # Rollback the transaction in case of error
```

1. Functionalities of Admin

1. Add Product

Allows the admin to add new products to the inventory.

The admin can input details such as product name, description, company name, category, status, supplier ID, and quantity.

2. Add NGO

Enables the admin to add new Non-Governmental Organizations (NGOs) to the system.

Information such as the NGO's name, description, email, address, and contact number can be added.

3. Show Delivery workers general information

Shows general information about delivery workers, including their ID, name, age, and city.

4. See the number of Active users

Provides the count of active users in the system.

5. Show Analysis of the current inventory

Displays an analysis of the current inventory, including product ID, name, quantity, supplier name, city, and state.

6. Display Users with at least one purchase

Lists users who have made at least one purchase along with their total number of purchases.

7. View Product Sale Stats

Shows statistics related to product sales, including the product name, supplier name, total units sold, and total sales amount.

8. Update the Quantity of Existing Product

Allows the admin to update the quantity of an existing product in the inventory.

9.Update Order Status

Updates the order statuses based on certain conditions, such as orders pending for more than a month being marked as 'Under Review'.

1. Functionalities of User

Function descriptions -

1. Sign In

Collects user information like name, age, contact details, and address.

Validates inputs (e.g., email, phone number).

Inserts user data into the database.

2. Login

Allows existing users to log in with their username and password.

Validates user credentials against the database.

Checks account status (e.g., whether the account is blocked).

3. View products

Displays available product categories.

Shows products under selected categories.

Allows users to view product details and add them to their cart.

4. View Cart

Displays products currently in the user's cart.

Calculates the total price of items in the cart.

Provides options to place an order or exit.

5. View Review-

View Review

6. Give Product Review-

Giving Product Review

7. Upgrade to Premium Member

8.Place Order

Collects payment mode and delivery details from the user.

Generates a transaction ID and inserts order details into the database.

Updates inventory and order status.

Prompts used for confirmation of the theoretical analysis and functionality explanation.