Assignment-Day-9-10

Assignment 1: Write a SELECT query to retrieve all columns from a 'customers' table, and modify it to return only the customer name and email address for customers in a specific city.

Solution:

Step 1: Retrieve All Columns from the customers Table

SELECT * FROM customers;

Step 2: Modify the Query to Return Only the Customer Name and Email Address for Customers in a Specific City

SELECT customer_name, email_address

FROM customers

WHERE city = 'Noida';

Assignment 2: Craft a query using an INNER JOIN to combine 'orders' and 'customers' tables for customers in a specified region, and a LEFT JOIN to display all customers including those without orders.

Solution:

SELECT c.customer_name, c.email_address, o.order_date

FROM customers c

LEFT JOIN orders o ON c.customer_id = o.customer_id

WHERE c.region = 'Noida';

Assignment 3: Utilize a subquery to find customers who have placed orders above the average order value, and write a UNION query to combine two SELECT statements with the same number of columns.

Solution:

SELECT customer_id, customer_name FROM customers c

WHERE c.customer_id IN (

```
SELECT customer_id
FROM orders o
WHERE o.order_value > (
  SELECT AVG(order_value)
  FROM orders
  )
);
```

Assignment 4: Compose SQL statements to BEGIN a transaction, INSERT a new record into the 'orders' table, COMMIT the transaction, then UPDATE the 'products' table, and ROLLBACK the transaction.

Solution:

```
BEGIN TRANSACTION;

-- Insert a new record into the 'orders' table (replace with actual column names and values)
INSERT INTO orders (order_id, customer_id, order_date, order_total)
VALUES (1001, 1, '21-May-2024', 250.00);

-- Assuming the insert is successful, Commit the transaction
COMMIT;

--Update the 'products' table (replace with your actual column names and values)
--This update will be rolled back since we use ROLLBACK later
UPDATE products
SET price = price * 1.10  -- Increase price by 10%
WHERE product_id = 101;

--Rollback the entire transaction, effectively undoing the insert
ROLLBACK;
```

Assignment 5: Begin a transaction, perform a series of INSERTs into 'orders', setting a SAVEPOINT after each, rollback to the second SAVEPOINT, and COMMIT the overall transaction.

Solution:

```
BEGIN TRANSACTION;
```

--Insert 1st order (replace with your actual column names and values)
INSERT INTO orders (order_id, customer_id, order_date, order_total)
VALUES (1001, 1, '21-May-2024', 250.00);

--Savepoint after 1st insert (name can be anything descriptive)
SAVEPOINT order1_inserted;

--Insert 2nd order
INSERT INTO orders (order_id, customer_id, order_date, order_total)
VALUES (1002, 2, '22-May-2024', 150.00);

--Savepoint after 2nd insert
SAVEPOINT order2_inserted;

--Insert 2nd order
INSERT INTO orders (order_id, customer_id, order_date, order_total)
VALUES (1003, 3, '23-May-2024', 300.00);

--Simulate an issue after the 3rd insert (can be removed for actual use)
RAISE WARNING ('Issue encountered processing order!');

--Rollback to the second savepoint (order2_inserted)
ROLLBACK TO SAVEPOINT order2_inserted;

--We only want the first two orders, so commit the transaction from here
COMMIT;

Assignment 6: Draft a brief report on the use of transaction logs for data recovery and create a hypothetical scenario where a transaction log is instrumental in data recovery after an unexpected shutdown.

Solution:

Transaction logs are a critical component in database management systems (DBMS). They record all transactions and modifications made to the database, ensuring data integrity and facilitating recovery in case of system failures, such as unexpected shutdowns.

Importance of Transaction Logs:

1.Data Integrity and Consistency: Transaction logs maintain a detailed record of all operations, which helps ensure the database can be restored to a consistent state after a failure.

2.Recovery from Failures: In the event of hardware malfunctions, power outages, or software crashes, transaction logs enable the database to be restored to the last known good state.

3.Audit Trail: They provide a chronological record of all database changes, which is valuable for auditing and troubleshooting purposes.

4.Rollback Mechanism: Transaction logs support rollback operations, allowing incomplete or erroneous transactions to be undone.

How Transaction Logs Work:

Recording Transactions: Each transaction's details are written sequentially to the transaction log. This includes the start of the transaction, the changes made (inserts, updates, deletes), and the end of the transaction.

Commit and Rollback: When a transaction is committed, a commit record is written to the log. If a transaction is rolled back, a rollback record is added.

Checkpointing: Periodically, the DBMS writes a checkpoint to the transaction log. A checkpoint is a snapshot of the database state at a specific point in time, which helps speed up the recovery process.

Scenario: Data Recovery Using Transaction Logs:

Scenario: A financial institution experiences an unexpected shutdown due to a power failure at their data center. The shutdown occurs at 3:00 PM while several transactions are in progress, including critical updates to customer accounts.

Data Recovery Process:

1.Restart the DBMS: Upon power restoration, the database management system (DBMS) is restarted.

2.Analyze the Transaction Log:

- The DBMS reads the transaction log from the last checkpoint, which was written at 2:45 PM.
- All transactions recorded after the checkpoint are analyzed. These include transactions that were committed, in progress, or rolled back before the shutdown.

3.Redo Committed Transactions:

- The DBMS identifies transactions that were committed after the checkpoint but before the shutdown.
- These transactions are reapplied to the database to ensure that all committed changes are reflected.

4.Undo Incomplete Transactions:

- Transactions that were in progress at the time of the shutdown are rolled back.
- This ensures that any partial changes made by these transactions do not leave the database in an inconsistent state.

5.Verify Data Integrity:

- The DBMS verifies the integrity and consistency of the database using the transaction log and checkpoint data.
- Additional integrity checks are performed to ensure no data corruption has occurred.

6.Resume Operations:

- Once the recovery process is complete, the database is brought back online for normal operations.
- Transactions that were rolled back can be resubmitted by the application or users if necessary.

Conclusion:

Transaction logs are indispensable for database recovery and maintaining data integrity. In scenarios like unexpected shutdowns, they provide a reliable mechanism to restore the database to a consistent state, preventing data loss and ensuring business continuity.