SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of Artificial Intelligence and Data Science

Course Name – 23ITB204-Modern Database Management Systems
II Year / III Semester

Topic - Transaction support in SQL

Transaction support in SQL and database management systems (DBMS) is crucial for ensuring data integrity and consistency.

What is a Transaction?

A transaction is a sequence of one or more SQL operations that are treated as a single unit of work. Transactions are essential for ensuring that a series of operations either complete fully or not at all, which helps maintain data integrity.

ACID Properties

Transactions in SQL are governed by the ACID properties:

1. **Atomicity**: Ensures that all operations within a transaction are completed successfully. If any operation fails, the entire transaction is rolled back to its previous state.
2. **Consistency**: Guarantees that a transaction takes the database from one valid state to another, preserving data integrity.
3. **Isolation**: Ensures that transactions are executed in isolation from one another. Intermediate results of a transaction are not visible to other transactions until it is committed.
4. **Durability**: Guarantees that once a transaction is committed, its changes are permanent and will survive system failures.

SQL Commands for Transactions

In SQL, you typically manage transactions using the following commands:

BEGIN TRANSACTION or **START TRANSACTION**: Initiates a new transaction.

COMMIT: Saves all the changes made during the transaction to the database.

ROLLBACK: Undoes all changes made during the current transaction, restoring the database to its previous state.

Example

Here’s a simple example demonstrating transaction support:

```
BEGIN TRANSACTION;
UPDATE Accounts SET balance = balance - 100 WHERE account_id = 1;
UPDATE Accounts SET balance = balance + 100 WHERE account_id = 2;
-- Check for errors
IF @@ERROR <> 0
BEGIN
    ROLLBACK;
    PRINT 'Transaction failed, changes were rolled back.';
END
ELSE
BEGIN
    COMMIT;
    PRINT 'Transaction successful, changes were committed.';
END
```

Isolation Levels

Different isolation levels determine how transactions interact with each other, impacting performance and data integrity. Common isolation levels include:

READ UNCOMMITTED: Allows reading uncommitted changes made by other transactions.

READ COMMITTED: Ensures that only committed changes are read.

REPEATABLE READ: Ensures that if a transaction reads a value, it will see the same value if it reads it again before the transaction ends.

SERIALIZABLE: Provides the strictest isolation, ensuring transactions are completely isolated from one another.

Thank You