**PROMPT :- Give me detailed documentation of TCL COMMAND in postgerSQL and syntax. Also, give two example of TCL COMMAND for beignner.**

**What is TCL in PostgreSQL?**

TCL (Transaction Control Language) commands in PostgreSQL are used to manage transactions in a database. Transactions are units of work that are executed as a single operation — either completely successful or completely rolled back (undone).

PostgreSQL is ACID-compliant, meaning it ensures Atomicity, Consistency, Isolation, and Durability — and TCL commands help enforce this.

**Main TCL Commands in PostgreSQL**

| Command | Description |
| --- | --- |
| BEGIN | Starts a new transaction |
| COMMIT | Saves all changes made during the current transaction |
| ROLLBACK | Cancels all changes made during the current transaction |
| SAVEPOINT | Sets a savepoint within a transaction (like a mini checkpoint) |
| ROLLBACK TO | Rolls back to a specific savepoint (not the whole transaction) |
| RELEASE | Deletes a savepoint (optional, PostgreSQL does it automatically at commit) |

Syntax of TCL Commands

**BEGIN**;

-- DML operations (INSERT, UPDATE, DELETE)

-- Optional: SAVEPOINT savepoint\_name;

-- To confirm changes

**COMMIT**;

-- OR to undo changes

**ROLLBACK**;

-- OR to undo part of it

ROLLBACK TO savepoint\_name;

**Why Use Transactions?**

* To ensure data integrity.
* To group multiple operations and either succeed completely or fail together.
* To recover from errors without corrupting the database.

**Example 1: Basic Transaction with COMMIT**

Suppose we have a BankAccounts table:

CREATE TABLE BankAccounts (

account\_id INT PRIMARY KEY,

holder\_name VARCHAR(50),

balance REAL

);

Now insert two rows and commit:

BEGIN;

INSERT INTO BankAccounts (account\_id, holder\_name, balance)

VALUES (1, 'Ravi Kumar', 10000);

INSERT INTO BankAccounts (account\_id, holder\_name, balance)

VALUES (2, 'Sneha Shah', 12000);

COMMIT;

**Explanation:**

* Both inserts are treated as a single transactio.
* If any insert fails, you could use ROLLBACK instead of COMMIT.

**Example 2: Transaction with ROLLBACK and SAVEPOINT**

BEGIN;

INSERT INTO BankAccounts (account\_id, holder\_name, balance)

VALUES (3, 'Amit Mehta', 8000);

SAVEPOINT sp1;

-- Intentional error: duplicate primary key

INSERT INTO BankAccounts (account\_id, holder\_name, balance)

VALUES (3, 'Priya Rao', 9500);

-- Error occurs here, rollback to savepoint

ROLLBACK TO sp1;

-- Now continue with valid insert

INSERT INTO BankAccounts (account\_id, holder\_name, balance)

VALUES (4, 'Priya Rao', 9500);

COMMIT;

**Explanation:**

* If an error happens after sp1, we rollback to it instead of the whole transaction.
* Finally, we commit only the valid changes.
* Key Points
* Use transactions when making multiple changes that depend on each other.
* Always test your operations inside BEGIN ... COMMIT in critical applications.