Transactions Oracle SQL

By Dhandapani Yedappalli Krishnamurthi Sep 30, 2025

🔹 What is a Transaction?

A **transaction** is a sequence of one or more SQL operations that are executed as a single unit of work.

- Either **all changes succeed** (COMMIT)
- Or **all changes are undone** (ROLLBACK)

👉 A transaction ensures **data consistency** and follows the **ACID properties** (Atomicity, Consistency, Isolation, Durability).

🔹 Key Transaction Control Statements in Oracle

1. **COMMIT** → Save changes permanently.
2. **ROLLBACK** → Undo changes since last COMMIT or ROLLBACK.
3. **SAVEPOINT** → Set a marker inside a transaction, so you can rollback partially.
4. **SET TRANSACTION** → Define properties of a transaction (read-only, isolation level, etc.).

🛠️ Example 1: Create Table and Insert Data

```
-- Step 1: Create a sample accounts table
CREATE TABLE accounts (
   acc_id    NUMBER PRIMARY KEY,
   acc_name  VARCHAR2(50),
   balance   NUMBER
);
-- Step 2: Insert Indian names with balances
INSERT INTO accounts VALUES (1, 'Kapil', 1000);
INSERT INTO accounts VALUES (2, 'Tharun', 1500);
INSERT INTO accounts VALUES (3, 'Sneha', 2000);
INSERT INTO accounts VALUES (4, 'Sangeetha', 2500);
COMMIT;  -- save the initial data
```

📊 Table now looks like:

ACC_ID	ACC_NAME	BALANCE
1	Kapil	1000
2	Tharun	1500
3	Sneha	2000
4	Sangeetha	2500

🛠️ Example 2: Transaction with COMMIT

-- Kapil transfers ₹200 to Tharun
UPDATE accounts SET balance = balance - 200 WHERE acc_name = 'Kapil';
UPDATE accounts SET balance = balance + 200 WHERE acc_name = 'Tharun';
-- Save permanently
COMMIT;
✅ **Transaction is complete. Kapil's balance decreases, Tharun's increases.**
✅ Changes are permanent now.

🛠️ Example 3: Transaction with ROLLBACK

-- Sneha tries to transfer ₹500 to Sangeetha
UPDATE accounts SET balance = balance - 500 WHERE acc_name = 'Sneha';
UPDATE accounts SET balance = balance + 500 WHERE acc_name = 'Sangeetha';
-- Oops! Cancel the whole transaction
ROLLBACK;
✅ Sneha and Sangeetha's balances go back to their original values.

🛠️ Example 4: SAVEPOINT and Partial Rollback

-- Tharun pays Kapil ₹100
UPDATE accounts SET balance = balance - 100 WHERE acc_name = 'Tharun';
UPDATE accounts SET balance = balance + 100 WHERE acc_name = 'Kapil';
SAVEPOINT step1;
-- Tharun pays Sneha ₹200
UPDATE accounts SET balance = balance - 200 WHERE acc_name = 'Tharun';
UPDATE accounts SET balance = balance + 200 WHERE acc_name = 'Sneha';
SAVEPOINT step2;
-- Tharun tries to pay extra ₹300 to Sangeetha
UPDATE accounts SET balance = balance - 300 WHERE acc_name = 'Tharun';
UPDATE accounts SET balance = balance + 300 WHERE acc_name = 'Sangeetha';
-- Realizes mistake, rollback to step2
ROLLBACK TO step2;
COMMIT;
✅ **Only the last wrong transfer (₹300 to Sangeetha) is undone.**

✅ The rollback only undoes the last deduction, but keeps earlier updates.

🛠️ Example 5: SET TRANSACTION

-- Open a read-only transaction
SET TRANSACTION READ ONLY;
SELECT * FROM accounts;
COMMIT;
👉 **Ensures no accidental changes while viewing balances.**
👉 This ensures no updates are allowed — only read operations.

🎯 Key Takeaways

- A **transaction begins automatically** when you execute the first DML statement (INSERT, UPDATE, DELETE).
- Use **COMMIT** to save and **ROLLBACK** to undo.
- Use **SAVEPOINT** for partial rollbacks.
- Transactions ensure **data consistency and reliability**.