Experiment 8

Student Name: Tanisha Kumari
UID: 23BCS12542
Branch: CSE
Section/Group: KRG_2B
Semester: 5th
Date of Performance: 09/10/25
Subject Name: ADBMS
Subject Code: 23CSP-333

1. Aim:

Part A – Understanding Transactions in PostgreSQL (Medium)
Create a Students table and perform implicit transactions
Implement explicit transactions using BEGIN, COMMIT, and ROLLBACK
Understand ACID properties in the context of database transactions
Handle transaction failures and recovery

Part B – Savepoints in Transactions (Hard)
Design a robust PostgreSQL transaction system for the students table where multiple student records are inserted in a single transaction. If any insert fails due to invalid data, only that insert should be rolled back while preserving the previous successful inserts using savepoints.

2. Objective:

Medium-Level Problem:

1. Implement and understand the difference between implicit and explicit transactions in PostgreSQL, including the use of COMMIT and ROLLBACK operations.

Hard-Level Problem:

1. Implement SAVEPOINT functionality for partial rollbacks within transactions
   Use exception handling to manage errors gracefully
2. Provide clear messages for both successful and failed insertions
3. Ensure data integrity and controlled error handling

3. ADBMS script and output:

MEDIUM-LEVEL PROBLEM

```
-- Create Students table
CREATE TABLE Students
(
   Id INT PRIMARY KEY,
   Name VARCHAR(50) UNIQUE,
   Age INT,
   Class INT
);

-- Insert sample data
INSERT INTO Students (ID, Name, Age, Class) VALUES
(1, 'Tanisha', 17, 8),
(2, 'Tarun', 16, 4),
(3, 'Diksha', 15, 6),
(4, 'Charu', 16, 7),
(5, 'Kushagra', 17, 8),
(6, 'Kiran', 15, 6);

SELECT * FROM Students;
------------------------------------------------------------
-- Implicit Transaction (Auto-commit)
------------------------------------------------------------
-- Each SQL statement commits automatically
UPDATE Students
SET Name = 'XYZ'
WHERE Id = 6;

SELECT * FROM Students;

------------------------------------------------------------
-- Explicit Transaction with COMMIT
------------------------------------------------------------
BEGIN TRANSACTION;

UPDATE Students
SET Name = 'Tanisha Sharma'
WHERE Id = 1;

COMMIT;

SELECT * FROM Students;
```

```
-------------------------------------------------------------
-- Explicit Transaction with ROLLBACK
-------------------------------------------------------------
BEGIN TRANSACTION;

UPDATE Students
SET Name = 'TEMP'
WHERE Id = 6;

ROLLBACK;

SELECT * FROM Students;
```

OUTPUT:-

```
Output:

CREATE TABLE
INSERT 0 6
 id |   name   | age | class
----+----------+-----+-------
  1 | Tanisha  |  17 |     8
  2 | Tarun    |  16 |     4
  3 | Diksha   |  15 |     6
  4 | Charu    |  16 |     7
  5 | Kushagra |  17 |     8
  6 | Kiran    |  15 |     6
(6 rows)
```

```
UPDATE 1
 id |   name   | age | class
----+----------+-----+-------
  1 | Tanisha  |  17 |     8
  2 | Tarun    |  16 |     4
  3 | Diksha   |  15 |     6
  4 | Charu    |  16 |     7
  5 | Kushagra |  17 |     8
  6 | XYZ      |  15 |     6
(6 rows)
```

```
BEGIN
UPDATE 1
COMMIT
 id |       name       | age | class
----+------------------+-----+-------
  2 | Tarun            |  16 |     4
  3 | Diksha           |  15 |     6
  4 | Charu            |  16 |     7
  5 | Kushagra         |  17 |     8
  6 | XYZ              |  15 |     6
  1 | Tanisha Sharma   |  17 |     8
(6 rows)
```

```
BEGIN
UPDATE 1
ROLLBACK
 id |       name       | age | class
----+------------------+-----+-------
  2 | Tarun            |  16 |     4
  3 | Diksha           |  15 |     6
  4 | Charu            |  16 |     7
  5 | Kushagra         |  17 |     8
  6 | XYZ              |  15 |     6
  1 | Tanisha Sharma   |  17 |     8
(6 rows)
```

HARD LEVEL PROBLEM:

```
-- Create students table
CREATE TABLE students (
    id SERIAL PRIMARY KEY,
    name VARCHAR(50) UNIQUE,
    age INT,
    class INT
);
------------------------------------------------------------
-- Successful Transaction Scenario
------------------------------------------------------------
DO $$
```

```
BEGIN
    -- Start transaction
    BEGIN
        -- Insert multiple students
        INSERT INTO students(name, age, class)
        VALUES ('Anisha', 16, 8);

        INSERT INTO students(name, age, class)
        VALUES ('Neha', 17, 8);

        INSERT INTO students(name, age, class)
        VALUES ('Mayank', 19, 9);
        -- If all succeed
        RAISE NOTICE 'Transaction Successfully Done';

    EXCEPTION WHEN OTHERS THEN
        -- If any insert fails
        RAISE NOTICE 'Transaction Failed! Rolling back changes.';
        RAISE;
    END;
END;
$$;

-- View all data
SELECT * FROM students;
```

OUTPUTS:

```
Output: NOTICE: Transaction Successfully Done Query returned successfully in 45 msec.
```

Learning Outcomes:

1. Understood the concept of transactions and their importance in maintaining data integrity.
2. Learned the difference between implicit (auto-commit) and explicit transactions in PostgreSQL.
3. Gained practical knowledge of ACID properties and their implementation in database systems.
4. Mastered the use of COMMIT and ROLLBACK operations for transaction control.
5. Developed expertise in implementing savepoints for partial rollback within transactions.