EXPERIMENT - 0 8

Student Name: Mishika
UID: 23BCS13811
Branch: BE-CSE
Section/Group: 2BCS_krg_2a
Semester: 5th
Date of Performance: 09/10/25
Subject Name: ADBMS
Subject Code: 23CSP-333

1. AIM:

. i) Understanding Transactions in PostgreSQL (Medium)

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

Tasks:

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

ii) Savepoints in Transactions (Hard)

Problem Statement: 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.

Objective:

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

.

2. Tools Used : PostGres

Solutions:

Q1)

```
--CREATING A TABLE
```

Creating Table and Basic Transactions

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

-- Inserting sample data
INSERT INTO Students (ID, Name, Age, Class) VALUES
(1,'Aarav', 17, 8),
(2,'Vikram', 16, 4),
(3,'Priya', 15, 6),
(4,'Rohan', 16, 7),
(5,'Sita', 17, 8),
(6,'Kiran', 15, 6);

SELECT * FROM STUDENTS;
```

Implicit Transaction (Auto-commit):

```
-- This UPDATE commits automatically
UPDATE STUDENTS
SET NAME = 'XYZ'
WHERE ID = 6;

-- Changes are permanent immediately
SELECT * FROM STUDENTS;
```

Explicit Transaction with COMMIT:

```
-- Start explicit transaction
BEGIN TRANSACTION;

UPDATE STUDENTS
SET NAME = 'AMAN'
WHERE ID = 1;

-- Make changes permanent
COMMIT;

SELECT * FROM STUDENTS;
```

Explicit Transaction with ROLLBACK:

```
-- Start transaction
BEGIN TRANSACTION;

UPDATE STUDENTS
SET NAME = 'TEMP'
WHERE ID = 6;

-- Undo all changes
ROLLBACK;

-- Original data is preserved
SELECT * FROM STUDENTS;
```

Advanced: Exception Handling with Savepoints (Hard Level)

```
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;
$$;

SELECT * FROM students;
```

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

4. 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.