Experiment- 8

Student Name: Anuj Kumar Singh
Branch: CSE
Semester: 5th
Subject Name: ADBMS

UID: 23BCS10098
Section/Group: KRG 1-B
Subject Code: 23CSP-333

1. Aim:

Design a robust PostgreSQL transaction system for the students table where multiple student records are inserted in a single transaction.

a. If any insert fails due to invalid data, only that insert should be rolled back.
b. Previous successful inserts should remain intact.
c. Use savepoints to manage partial rollbacks.
d. Provide clear messages for successful and failed insertions.

2. Objective:

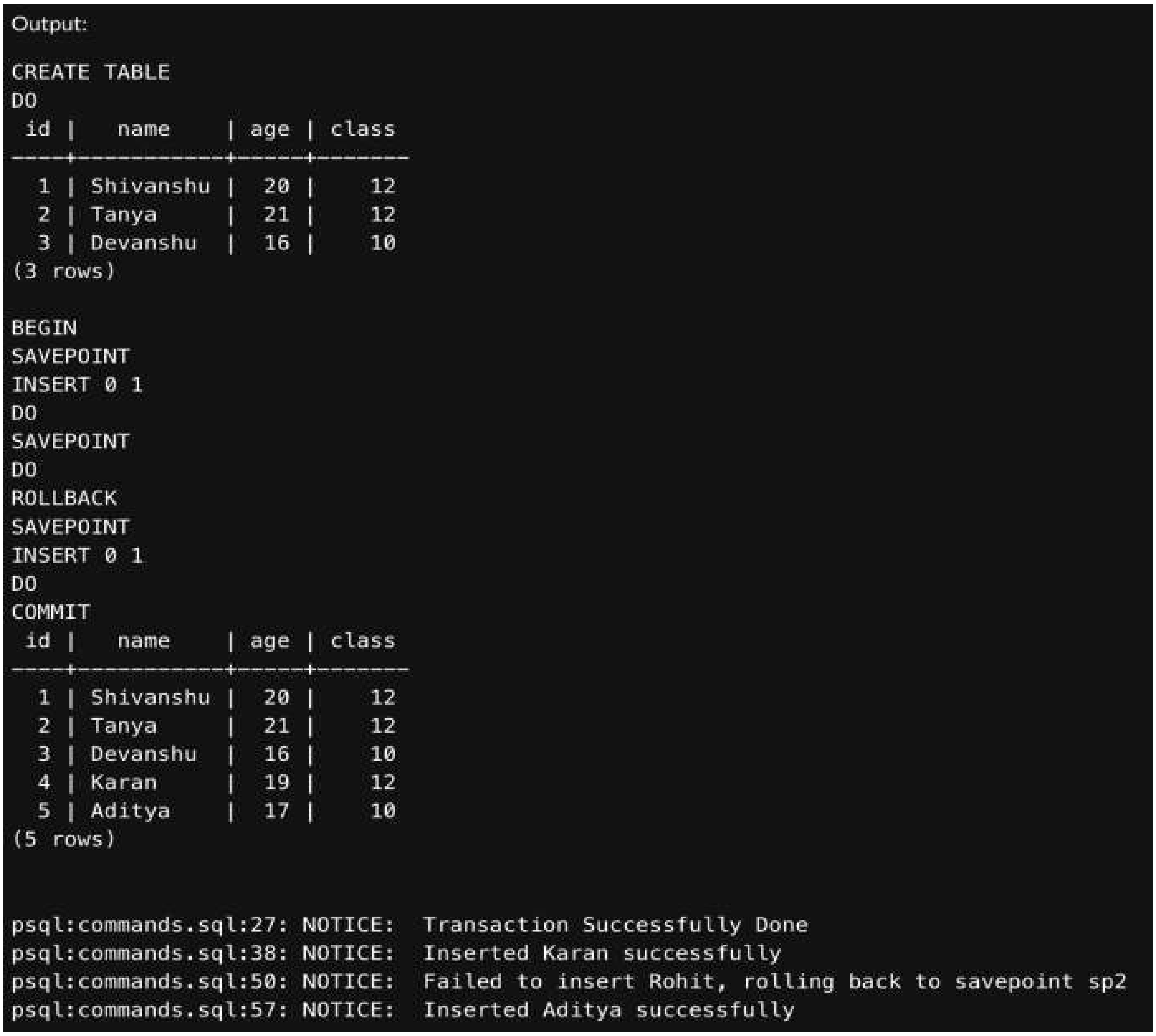
- Understand Transaction Management in PostgreSQL
- Learn Partial Rollback Using Savepoints
- Handle Errors Gracefully
- Provide Feedback on Database Operations
- **Develop Robust and Fault-tolerant Database Systems**

3. Code:

```
-- Create table
CREATE TABLE students (
   id SERIAL PRIMARY KEY,
   name VARCHAR(50),
   age INT,
   class INT
);

-- Insert multiple students in one transaction
DO $$
BEGIN
   BEGIN
      INSERT INTO students(name, age, class) VALUES ('Shivanshu',20,12);
      INSERT INTO students(name, age, class) VALUES ('Tanya',21,12);
      INSERT INTO students(name, age, class) VALUES ('Devanshu',16,10);
      RAISE NOTICE 'Transaction Successfully Done';
   EXCEPTION
      WHEN OTHERS THEN
```

```
RAISE NOTICE 'Transaction Failed..! Rolling back all changes.'; RAISE;
   END;
END;
$$;

SELECT * FROM students;

-- Transaction with Savepoints

BEGIN;  -- Start transaction

-- Savepoint 1: Karan
SAVEPOINT sp1;
INSERT INTO students(name, age, class) VALUES ('Karan',19,12);
DO $$ BEGIN RAISE NOTICE 'Inserted Karan successfully'; END $$;

-- Savepoint 2: Rohit (invalid insert)
SAVEPOINT sp2;
DO $$
BEGIN
   BEGIN
      INSERT INTO students(name, age, class) VALUES ('Rohit','wrong',12);
   EXCEPTION WHEN OTHERS THEN
      RAISE NOTICE 'Failed to insert Rohit, rolling back to savepoint sp2';
   END;
END;
$$;
-- Rollback the failed insert in SQL
ROLLBACK TO SAVEPOINT sp2;

-- Savepoint 3: Aditya
SAVEPOINT sp3;
INSERT INTO students(name, age, class) VALUES ('Aditya',17,10);
DO $$ BEGIN RAISE NOTICE 'Inserted Aditya successfully'; END $$;

-- Commit all successful inserts
COMMIT;

SELECT * FROM students;
```

4. Output:


```
Output:

CREATE TABLE
DO
 id |    name    | age | class
----+------------+-----+-------
  1 | Shivanshu  |  20 |    12
  2 | Tanya      |  21 |    12
  3 | Devanshu   |  16 |    10
(3 rows)

BEGIN
SAVEPOINT
INSERT 0 1
DO
SAVEPOINT
DO
ROLLBACK
SAVEPOINT
INSERT 0 1
DO
COMMIT
 id |    name    | age | class
----+------------+-----+-------
  1 | Shivanshu  |  20 |    12
  2 | Tanya      |  21 |    12
  3 | Devanshu   |  16 |    10
  4 | Karan      |  19 |    12
  5 | Aditya     |  17 |    10
(5 rows)


psql:commands.sql:27: NOTICE:  Transaction Successfully Done
psql:commands.sql:38: NOTICE:  Inserted Karan successfully
psql:commands.sql:50: NOTICE:  Failed to insert Rohit, rolling back to savepoint sp2
psql:commands.sql:57: NOTICE:  Inserted Aditya successfully
```

4. Learning Outcomes:

- Master Transaction Control
- Implement Partial Rollbacks with Savepoints
- Error Handling in Database Operations
- Provide Clear Feedback and Maintain Data Consist