

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

EXPERIMENT - 8

Student Name: RAVI
Branch: CSE
Semester: 5th
Subject Name: ADBMS

UID: 23BCS10340
Section/Group: KRG 3-A
Date of Performance: 09/10/2025
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.

- If any insert fails due to invalid data, only that insert should be rolled back.
- Previous successful inserts should remain intact.
- Use savepoints to manage partial rollbacks.
- 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 students (  
    id SERIAL PRIMARY KEY,  
    name VARCHAR(50),  
    age INT,  
    class INT  
);  
  
DO $$  
BEGIN  
    INSERT INTO students(name, age, class) VALUES ('Ravi', 22, 12);  
    INSERT INTO students(name, age, class) VALUES ('Sohneyo', 21, 12);  
    INSERT INTO students(name, age, class) VALUES ('Rias', 18, 11);  
    RAISE NOTICE 'Transaction Successfully Done ';  
EXCEPTION  
    WHEN OTHERS THEN  
        RAISE NOTICE 'Transaction Failed..! Rolling back all changes ';  
        RAISE;
```

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

```
END;
END;
$$;

SELECT * FROM students;

BEGIN;

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

SAVEPOINT sp2;
DO $$
BEGIN
    BEGIN
        INSERT INTO students(name, age, class) VALUES ('Rohan', 'wrong', 10);
    EXCEPTION
        WHEN OTHERS THEN
            RAISE NOTICE 'Failed to insert Rohan, rolling back to savepoint sp2 ';
    END;
END;
$$;

ROLLBACK TO SAVEPOINT sp2;

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

COMMIT;

SELECT * FROM students;
```

4. Output:

Output:

```
CREATE TABLE
```

```
DO
```

id	name	age	class
1	Ravi	22	12
2	Sohneyo	21	12
3	Rias	18	11

(3 rows)

```
BEGIN
```

```
SAVEPOINT
```

```
INSERT 0 1
```

```
DO
```

```
SAVEPOINT
```

```
DO
```

```
ROLLBACK
```

```
SAVEPOINT
```

```
INSERT 0 1
```

```
DO
```

```
COMMIT
```

id	name	age	class
1	Ravi	22	12
2	Sohneyo	21	12
3	Rias	18	11
4	Hinata	19	12
5	Aditya	17	10

(5 rows)

```
psql:commands.sql:21: NOTICE: Transaction Successfully Done
```

```
psql:commands.sql:29: NOTICE: Inserted Hinata successfully
```

```
psql:commands.sql:41: NOTICE: Failed to insert Rohan, rolling back to savepoint sp2
```

```
psql:commands.sql:47: NOTICE: Inserted Aditya successfully
```

5.Learning Outcomes:

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