



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Experiment 6

**Student Name:** Sambhav Mahajan

**UID:** 23BCS11290

**Branch:** B.E. C.S.E.

**Section/Group:** KRG-2B

**Semester:** 5th

**Date of Performance:**

**Subject Name:** ADBMS

**Subject Code:** 23CSP-333

**1. Aim:** To create a stored procedure in PostgreSQL that dynamically counts the number of employees based on a gender input, providing HR with an automated tool for gender diversity reporting.

## 2. Objective:

1. Design and create an employee's table with sample HR data (name, gender, department, etc.).
2. Implement a stored procedure `get_employee_count_by_gender` that:
  - Accepts a gender value (Male or Female) as input.
  - Returns the total number of employees for that gender.
3. Demonstrate calling the procedure and displaying results.
4. Show how this solution helps in HR decision-making for workforce diversity monitoring.

## 3. DBMS Script:

```
-- =====  
-- Step 1: Setup Employees Table with Sample Data  
-- =====
```

DROP TABLE IF EXISTS employees;

```
CREATE TABLE employees (  
    emp_id    SERIAL PRIMARY KEY,  
    emp_name  VARCHAR(100) NOT NULL,  
    gender    VARCHAR(10) CHECK (gender IN ('Male', 'Female')),  
    department VARCHAR(50),  
    joining_date DATE DEFAULT CURRENT_DATE  
);
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
-- Insert sample employees
INSERT INTO employees (emp_name, gender, department) VALUES
('Amit Sharma', 'Male', 'IT'),
('Priya Nair', 'Female', 'HR'),
('Ravi Kumar', 'Male', 'Finance'),
('Neha Singh', 'Female', 'IT'),
('Arjun Mehta', 'Male', 'Marketing'),
('Shreya Iyer', 'Female', 'Finance');
```

```
-- =====
-- Step 2: Create Stored Procedure for Employee Count by Gender
-- =====
```

```
DROP PROCEDURE IF EXISTS get_employee_count_by_gender(VARCHAR, OUT INT);
```

```
CREATE OR REPLACE PROCEDURE get_employee_count_by_gender(
    IN in_gender VARCHAR,    -- input parameter
    OUT emp_count INT       -- output parameter
)
LANGUAGE plpgsql
AS $$
BEGIN
    SELECT COUNT(*) INTO emp_count
    FROM employees
    WHERE gender = in_gender;

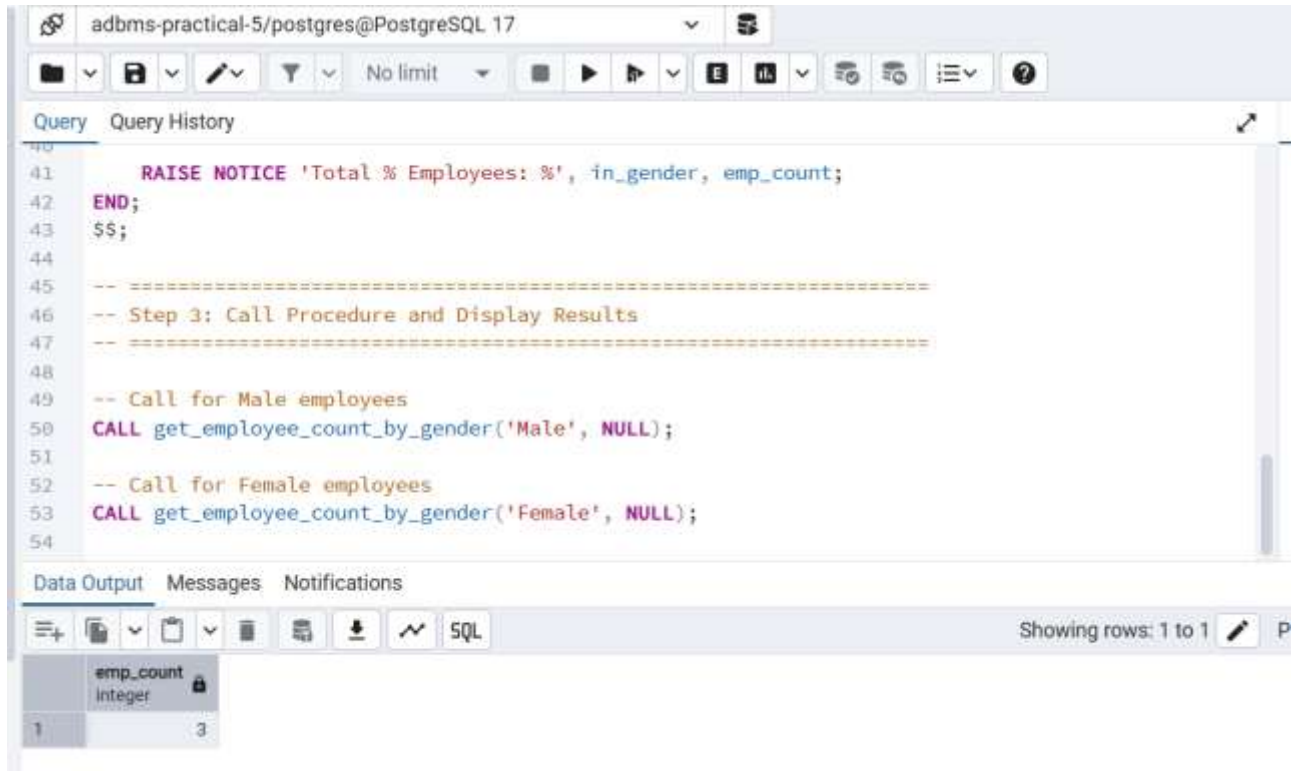
    RAISE NOTICE 'Total % Employees: %', in_gender, emp_count;
END;
$$;
```

```
-- =====
-- Step 3: Call Procedure and Display Results
-- =====
```

```
-- Call for Male employees
CALL get_employee_count_by_gender('Male', NULL);

-- Call for Female employees
CALL get_employee_count_by_gender('Female', NULL);
```

## 4. Output:



The screenshot shows a PostgreSQL query editor window titled 'adbms-practical-5/postgres@PostgreSQL 17'. The query editor contains the following SQL code:

```

41 RAISE NOTICE 'Total % Employees: %', in_gender, emp_count;
42 END;
43 $$;
44
45 -- =====
46 -- Step 3: Call Procedure and Display Results
47 -- =====
48
49 -- Call for Male employees
50 CALL get_employee_count_by_gender('Male', NULL);
51
52 -- Call for Female employees
53 CALL get_employee_count_by_gender('Female', NULL);
54

```

Below the query editor, the 'Data Output' tab is active, showing a table with the following data:

emp_count	integer
1	3

The 'Messages' tab is also visible, showing the output of the RAISE NOTICE statement.

Figure 1

## 5. Learning Outcomes:

1. Create and populate HR-related datasets in PostgreSQL.
2. Develop stored procedures with input and output parameters.
3. Use PL/pgSQL procedural logic to perform computations and return results.
4. Implement dynamic HR reporting queries for workforce analytics.
5. Understand how database-driven automation can support real-world HR decision-making (e.g., diversity monitoring).