Experiment 1.6

Student Name: Vaibhav Kumar   **UID:** 23BCS13386

Branch: B.E-C.S.E   **Section/Group:** 23BCS_KRG-2B

Semester: 5th   **Date of Performance:** 24 Sept, 2025

Subject Name: ADBMS   **Subject Code:** 23CSP-333

MEDIUM - LEVEL

1. **Problem Title:** HR-Analytics: Employee count based on dynamic gender passing

2. **Procedure (Step-by-Step):**

    TechSphere Solutions, a growing IT services company with offices across India, wants to **track and monitor gender diversity** within its workforce. The HR department frequently needs to know the **total number of employees by gender** (Male or Female) .

    To solve this problem, the company needs an **automated database-driven solution** that can instantly return the count of employees by gender through a stored procedure that:

    1. Create a PostgreSQL stored procedure that:
    2. Takes a **gender** (e.g., 'Male' or 'Female') as input.
    3. Calculates the **total count of employees** for that gender.
    4. Returns the result as an **output parameter**.
    5. Displays the result clearly for HR reporting purposes.

3. **SQL Commands:**

```sql
1
2  -----------------------MEDIUM PROBLEM--------------------------
3  CREATE TABLE employees (
4      emp_id SERIAL PRIMARY KEY,
5      emp_name VARCHAR(100),
6      gender VARCHAR(10)
7  );
8
9  -- Sample data
10 INSERT INTO employees (emp_name, gender) VALUES
11 ('Amit', 'Male'),
12 ('Priya', 'Female'),
13 ('Ravi', 'Male'),
14 ('Sneha', 'Female'),
15 ('Karan', 'Male');
16
17 select * from EMPLOYEES;
18
19 ----CREATING A PROCEDURE----
20 CREATE OR REPLACE PROCEDURE count_employees_by_gender(
21      IN input_gender VARCHAR,
22     OUT total_count int
23 )
24 LANGUAGE plpgsql
25 AS $$
26 BEGIN
27     SELECT COUNT(*) INTO total_count
28     FROM employees
29     WHERE gender = input_gender;
30 END;
31 $$;
32
33 ---CALLING THE PROCEDURE-----
34 DO $$
35 DECLARE
36     result INT;
37 BEGIN
38     CALL count_employees_by_gender('Male', result);
39     RAISE NOTICE 'TOTAL EMPLOYEES OF GENDER Male ARE %', result;
40 END;
41 $$;
42
```

4. **Output:**

STDIN

Input for the program (Optional)

Output:

```
CREATE TABLE
INSERT 0 5
 emp_id | emp_name | gender
--------+----------+--------
      1 | Amit     | Male
      2 | Priya    | Female
      3 | Ravi     | Male
      4 | Sneha    | Female
      5 | Karan    | Male
(5 rows)

CREATE PROCEDURE
DO

psql:commands.sql:40: NOTICE:  TOTAL EMPLOYEES OF GENDER Male ARE 3
```

HARD - LEVEL

1. **Problem Title:** SmartStore Automated Purchase System

2. **Procedure (Step-by-Step):**

   SmartShop is a modern retail company that sells electronic gadgets like smartphones, tablets, and laptops.
   The company wants to **automate its ordering and inventory management process**.
   Whenever a customer places an order, the system must:
   1. **Verify stock availability** for the requested product and quantity.
   2. If sufficient stock is available:
      - **Log the order** in the sales table with the ordered quantity and total price.
      - **Update the inventory** in the products table by reducing quantity_remaining and increasing   quantity_sold.
      - Display a **real-time confirmation message**: "Product sold successfully!"
   3. If there is **insufficient stock**, the system must:
      - **Reject the transaction** and display: Insufficient Quantity Available!"

3. SQL Commands:

commands.sql   EXPERIMENT-6(HARD) ✏

```sql
1  ---------------------------HARD PROBLEM ----------------------------------
2  CREATE TABLE products (
3      product_id SERIAL PRIMARY KEY,
4      product_name VARCHAR(100),
5      price NUMERIC(10,2),
6      quantity_remaining INT,
7      quantity_sold INT DEFAULT 0
8  );
9
10 INSERT INTO products (product_name, price, quantity_remaining) VALUES
11 ('Smartphone', 30000, 50),
12 ('Tablet', 20000, 30),
13 ('Laptop', 60000, 20);
14
15 CREATE TABLE sales (
16     sale_id SERIAL PRIMARY KEY,
17     product_id INT REFERENCES products(product_id),
18     quantity INT,
19     total_price NUMERIC(10,2),
20     sale_date TIMESTAMP DEFAULT NOW()
21 );
22
23 CREATE OR REPLACE PROCEDURE place_order(
24     IN p_product_id INT,
25     IN p_quantity INT
26 )
27 LANGUAGE plpgsql
28 AS $$
29 DECLARE
30     available_stock INT;
31     product_price NUMERIC(10,2);
32 BEGIN
33     SELECT quantity_remaining, price
34     INTO available_stock, product_price
35     FROM products
36     WHERE product_id = p_product_id;
37
38     IF available_stock IS NULL THEN
39        RAISE NOTICE 'Product ID % does not exist!', p_product_id;
40     ELSIF available_stock >= p_quantity THEN
41        -- LOGGING THE ORDER
42        INSERT INTO sales (product_id, quantity, total_price)
43        VALUES (p_product_id, p_quantity, p_quantity * product_price);
44
45        UPDATE products
46        SET quantity_remaining = quantity_remaining - p_quantity,
47            quantity_sold = quantity_sold + p_quantity
48        WHERE product_id = p_product_id;
49
50        RAISE NOTICE 'Product sold successfully!';
51     ELSE
52        RAISE NOTICE 'Insufficient Quantity Available!';
53     END IF;
54 END;
55 $$;
56
57
58 CALL PLACE_ORDER(2,20); --PRODUCT SOLD SUCCESSFULLY AND QUANTITY_REMAINING COLUMN SET TO -20 AND
59 SELECT * FROM SALES;
60 SELECT * FROM PRODUCTS;
61 CALL PLACE_ORDER(3,100); --INSUFFICIENT QUANTITY AVAILABLE
```

4. Output:

STDIN

Input for the program (Optional)

Output:

```
CREATE TABLE
INSERT 0 3
CREATE TABLE
CREATE PROCEDURE
CALL
 sale_id | product_id | quantity | total_price |          sale_date
---------+------------+----------+-------------+----------------------------
       1 |          2 |       20 |   400000.00 | 2025-09-28 15:08:21.159238
(1 row)


 product_id | product_name |  price   | quantity_remaining | quantity_sold
------------+--------------+----------+--------------------+---------------
          1 | Smartphone   | 30000.00 |                 50 |             0
          3 | Laptop       | 60000.00 |                 20 |             0
          2 | Tablet       | 20000.00 |                 10 |            20
(3 rows)

CALL

psql:commands.sql:58: NOTICE:  Product sold successfully!
psql:commands.sql:61: NOTICE:  Insufficient Quantity Available!
```