DAY -9

1) .Create AFTER UPDATE trigger to track product price changes.

**QUERY:**

**Create Table for logging:**

CREATING TABLE CREATE TABLE product\_price\_log

( log\_id SERIAL PRIMARY KEY,

product\_id INT, old\_price NUMERIC,

new\_price NUMERIC,

changed\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP );



Create Trigger Function:

CREATE OR REPLACE FUNCTION log\_price\_change()

RETURNS TRIGGER AS $$

BEGIN

IF OLD.unit\_price <> NEW.unit\_price

THEN INSERT INTO product\_price\_log (product\_id, old\_price, new\_price)

VALUES (OLD.product\_id, OLD.unit\_price, NEW.unit\_price);

END IF;

RETURN NEW;

END;

$$

LANGUAGE plpgsql;

**Output:**



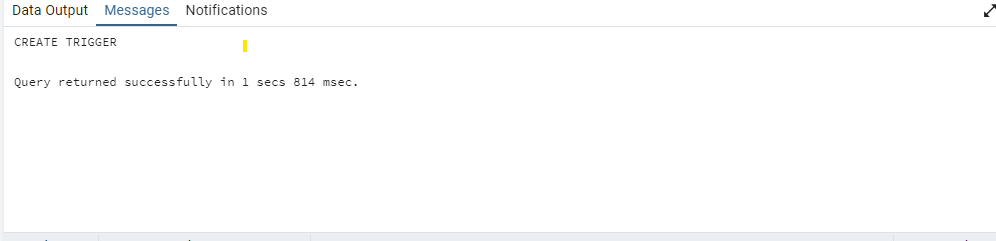
**Create the trigger on the products table**

CREATE TRIGGER after\_price\_update

AFTER UPDATE ON products

FOR EACH ROW EXECUTE FUNCTION log\_price\_change();

Output:



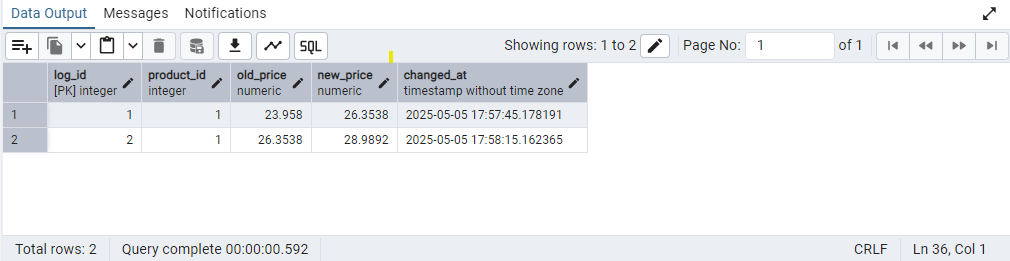
**Update price by 10% and verify the audit log:**

**QUERY:**

UPDATE products SET unit\_price = unit\_price \* 1.10 WHERE product\_id = 1;

SELECT \* FROM product\_price\_log WHERE product\_id = 1;

**OUTPUT:**



2) Create stored procedure using IN and INOUT parameters to assign tasks to employees

**Query:**

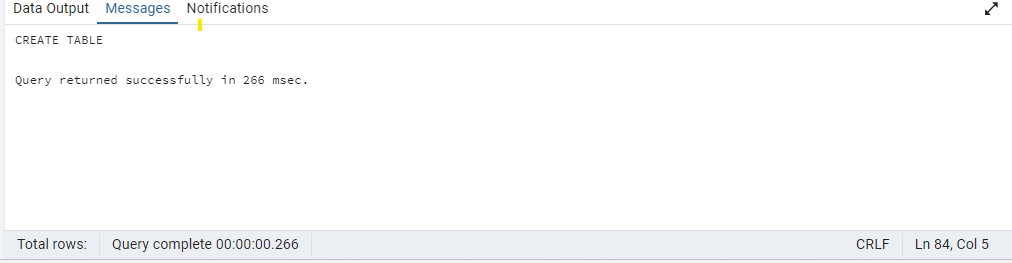
CREATE TABLE IF NOT EXISTS employee\_tasks (

task\_id SERIAL PRIMARY KEY,

employee\_id INT,

task\_name VARCHAR(50),

assigned\_date DATE DEFAULT CURRENT\_DATE)



CREATE OR REPLACE PROCEDURE assign\_task(

IN p\_employee\_id INT,

IN p\_task\_name VARCHAR(50),

INOUT p\_task\_count INT DEFAULT 0

)

LANGUAGE plpgsql

AS $$

BEGIN

INSERT INTO employee\_tasks (employee\_id, task\_name)

VALUES (p\_employee\_id, p\_task\_name);

SELECT COUNT(\*) INTO p\_task\_count

FROM employee\_tasks

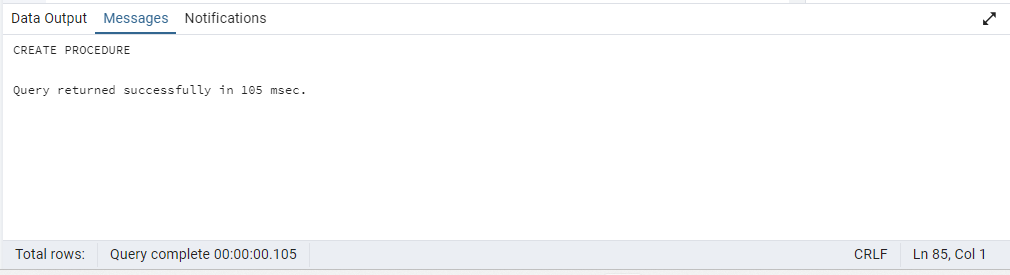
WHERE employee\_id = p\_employee\_id;

RAISE NOTICE 'Task "%" assigned to employee %. Total tasks: %',

p\_task\_name, p\_employee\_id, p\_task\_count;

END;

$$;



CALL assign\_task(1, 'Review Reports', 0);

DO $$

DECLARE

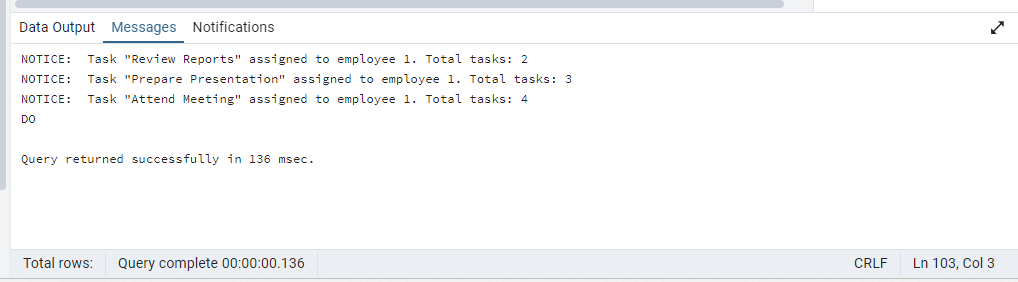
task\_count INT := 0;

BEGIN

CALL assign\_task(1, 'Prepare Presentation', task\_count);

CALL assign\_task(1, 'Attend Meeting', task\_count);

END; $$



SELECT \* FROM employee\_tasks ORDER BY task\_id;

