/\* 1.Create AFTER UPDATE trigger to track product price changes\*/

--Create product\_price\_audit table

CREATE TABLE product\_price\_audit (

audit\_id SERIAL PRIMARY KEY,

product\_id INT,

product\_name VARCHAR(40),

old\_price DECIMAL(10,2),

new\_price DECIMAL(10,2),

change\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

user\_name VARCHAR(50) DEFAULT CURRENT\_USER

);

A screenshot of a computer

AI-generated content may be incorrect.

--Create a trigger function

CREATE OR REPLACE FUNCTION log\_price\_change()

RETURNS TRIGGER

AS $$

BEGIN

INSERT INTO product\_price\_audit (

product\_id,

product\_name,

old\_price,

new\_price

)

VALUES (

OLD.product\_id,

OLD.product\_name,

OLD.unit\_price,

NEW.unit\_price

);

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

A screenshot of a computer

AI-generated content may be incorrect.

--Create a row level trigger for below event--AFTER UPDATE OF unit\_price ON products

CREATE TRIGGER trg\_price\_update

AFTER UPDATE OF unit\_price ON products

FOR EACH ROW

WHEN (OLD.unit\_price IS DISTINCT FROM NEW.unit\_price)

EXECUTE FUNCTION log\_price\_change();

A screenshot of a computer

AI-generated content may be incorrect.

--Test the trigger by updating the product price by 10% to any one product\_id.

UPDATE products

SET unit\_price = unit\_price \* 1.10

WHERE product\_id = 70;

A screenshot of a computer

AI-generated content may be incorrect.

SELECT \* FROM products WHERE product\_id = 70;’

A screenshot of a computer

AI-generated content may be incorrect.

--Cross check the audit table

SELECT \* FROM product\_price\_audit

ORDER BY change\_date DESC;

A screenshot of a computer

AI-generated content may be incorrect.

/\*2. Create stored procedure using IN and INOUT parameters to assign tasks to employees\*/

--Create the employee\_tasks table

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

);

A screenshot of a computer

AI-generated content may be incorrect.

--Create the stored procedure

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 task assignment

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

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

-- Count total tasks for this employee

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

FROM employee\_tasks

WHERE employee\_id = p\_employee\_id;

-- Raise notice

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

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

END;

$$;

A screenshot of a computer

AI-generated content may be incorrect.

--Call the procedure to test it

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

A screenshot of a computer program

AI-generated content may be incorrect.

-- Cross check inserted

SELECT \* FROM employee\_tasks

WHERE employee\_id = 1;

A screenshot of a computer

AI-generated content may be incorrect.