1) CREATE TABLE product\_price\_audit (

audit\_id SERIAL PRIMARY KEY,

productid INT,

old\_price DECIMAL(10, 2),

new\_price DECIMAL(10, 2),

changed\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE OR REPLACE FUNCTION log\_price\_change()

RETURNS TRIGGER AS $$

BEGIN

-- Only log if the price actually changed

IF OLD.unitprice IS DISTINCT FROM NEW.unitprice THEN

INSERT INTO product\_price\_audit (productid, old\_price, new\_price)

VALUES (OLD.productid, OLD.unitprice, NEW.unitprice);

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

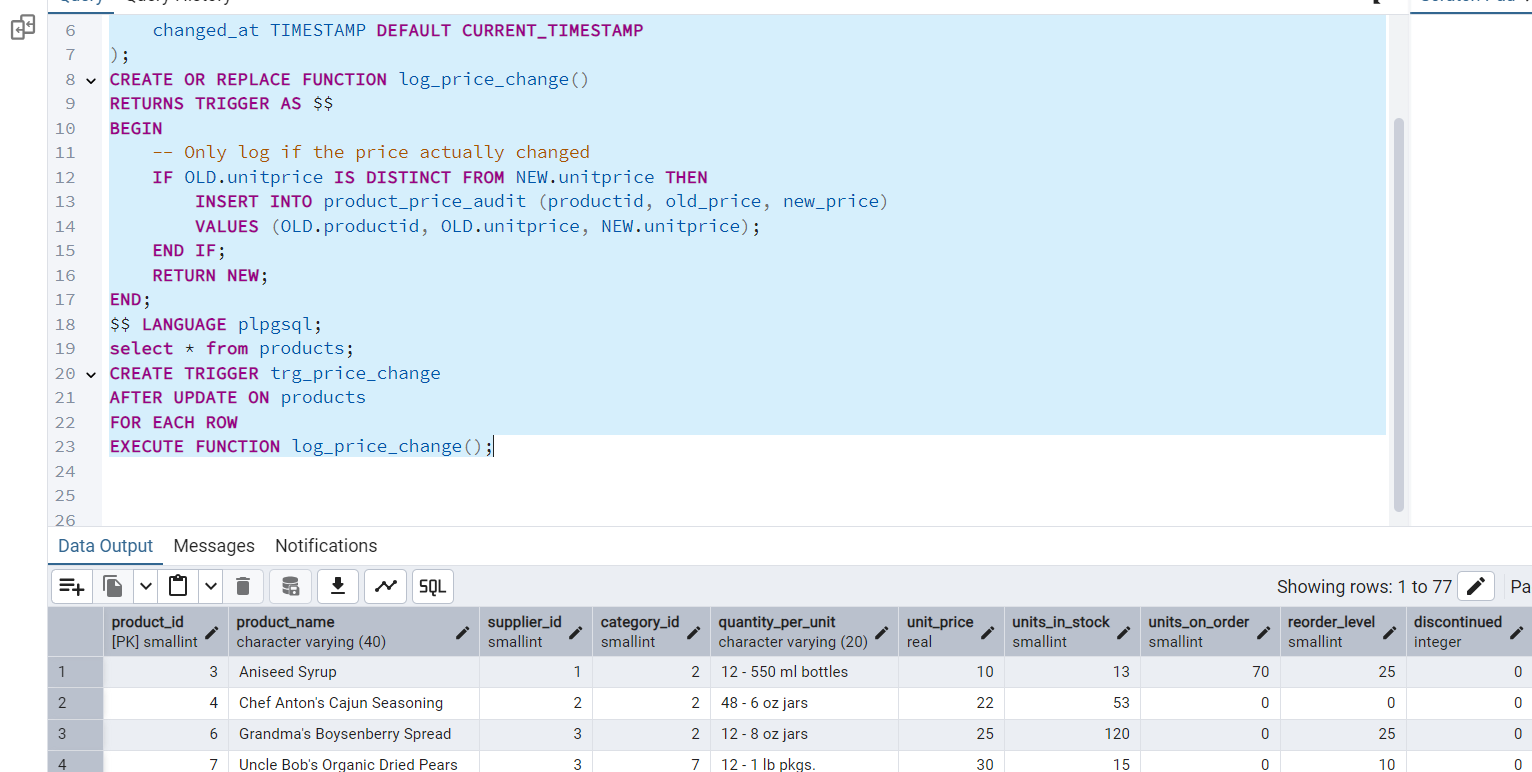
select \* from products;

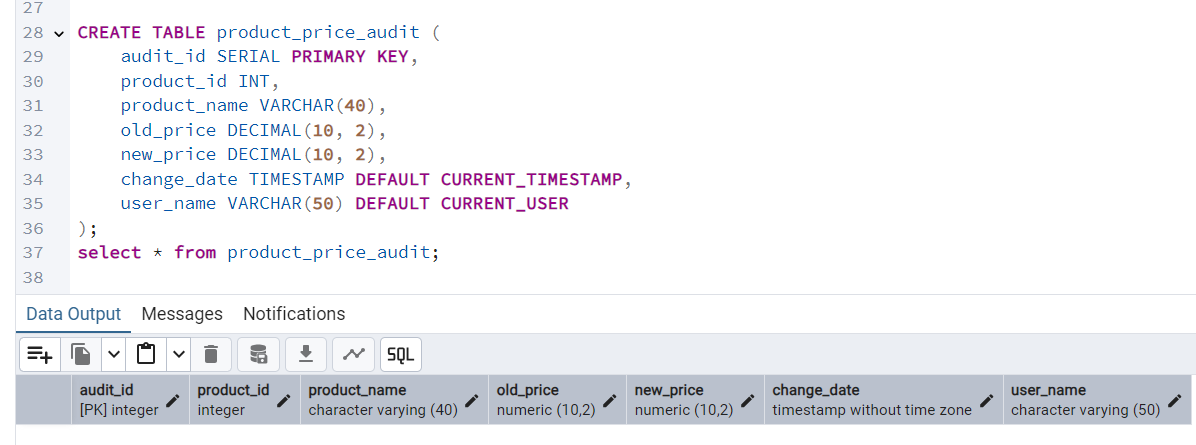
CREATE TRIGGER trg\_price\_change

AFTER UPDATE ON products

FOR EACH ROW

EXECUTE FUNCTION log\_price\_change();





CREATE OR REPLACE FUNCTION log\_product\_price\_change()

RETURNS TRIGGER AS $$

BEGIN

-- Only log if the price has actually changed

IF OLD.unit\_price IS DISTINCT FROM NEW.unit\_price THEN

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

);

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg\_log\_unit\_price\_change

AFTER UPDATE OF unit\_price ON products

FOR EACH ROW

EXECUTE FUNCTION log\_product\_price\_change();

select \* from products;

UPDATE products

SET unit\_price = unit\_price \* 1.10

WHERE product\_id = 3;

SELECT \* FROM product\_price\_audit

ORDER BY change\_date DESC

LIMIT 5;

2)

CREATE OR REPLACE PROCEDURE assign\_task\_to\_employee(

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;

$$;

SELECT \* FROM information\_schema.tables

WHERE table\_name = 'employee\_tasks';

CREATE TABLE employee\_tasks (

task\_id SERIAL PRIMARY KEY,

employee\_id INT,

task\_name VARCHAR(50),

assigned\_date DATE DEFAULT CURRENT\_DATE

);

DO $$

DECLARE

task\_total INT := 0;

BEGIN

CALL assign\_task\_to\_employee(101, 'Submit Status Report', task\_total);

END;

$$;

select \* from employee\_tasks;