–1,1. --Write a function to Calculate the total stock value for a given category:

--(Stock value=ROUND(SUM(unit\_price \* units\_in\_stock)::DECIMAL, 2)

--Return data type is DECIMAL(10,2)

CREATE OR REPLACE FUNCTION get\_stock\_value\_by\_category(

p\_category\_id INT

)

RETURNS DECIMAL(10,2)

LANGUAGE plpgsql

AS $$

DECLARE

v\_stock\_value DECIMAL(10,2);

BEGIN

SELECT ROUND(SUM(unit\_price \* units\_in\_stock)::DECIMAL, 2)

INTO v\_stock\_value

FROM products

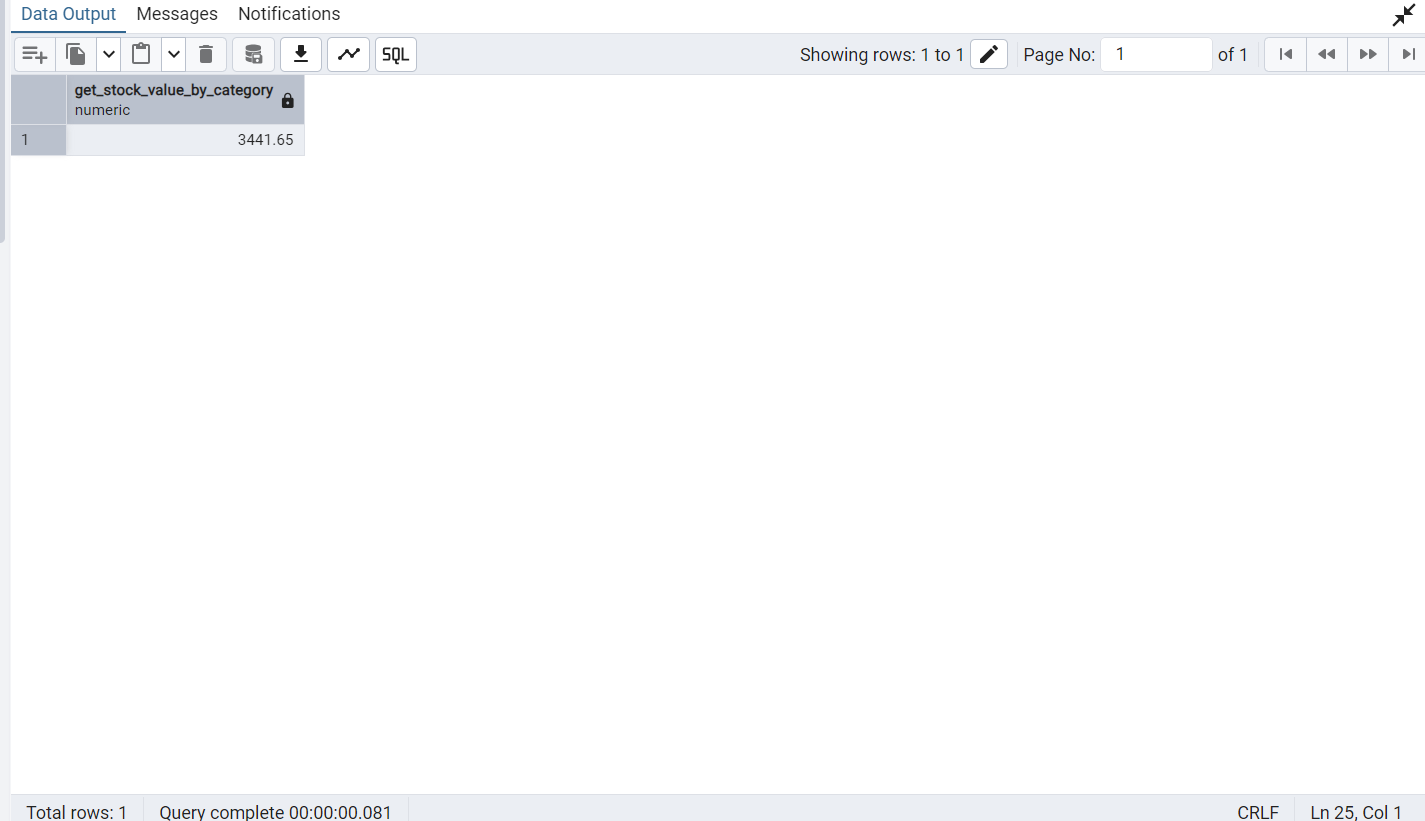
WHERE category\_id = p\_category\_id;

RETURN COALESCE(v\_stock\_value, 0.00);

END;

$$;

SELECT get\_stock\_value\_by\_category(7);



--2. Try writing a cursor query which I executed in the training

--This cursor will iterate through products and adjust prices based on units in stock

CREATE OR REPLACE PROCEDURE update\_prices\_with\_cursor()

LANGUAGE plpgsql

AS $$

DECLARE

product\_cursor CURSOR FOR

SELECT product\_id,product\_name,unit\_price,units\_in\_stock

FROM products

WHERE discontinued =0 ;

product\_record RECORD;

v\_new\_price DECIMAL(10,2);

BEGIN

--open the cursor

OPEN product\_cursor;

LOOP

**--fetch the next row**

FETCH product\_cursor INTO product\_record;

--Exit whn no more rows to fetch

EXIT WHEN NOT FOUND;

**--calculate new price**

IF product\_record.units\_in\_stock < 10 THEN

v\_new\_price := product\_record.unit\_price \* 1.1;-- 10% increase

ELSE

v\_new\_price := product\_record.unit\_price \* 0.95;-- 5% increase

END IF;

**--update the product**

UPDATE products

SET unit\_price = ROUND (v\_new\_price,2)

WHERE product\_id =product\_record.product\_id;

**--log the change**

RAISE NOTICE 'updated% price from % to %',

product\_record.product\_name,

product\_record.unit\_price,

v\_new\_price;

END LOOP;

**-- close the cursor**

CLOSE product\_cursor;

END;

$$;

**--TO execute:**

CALL update\_prices\_with\_cursor();

