**Day 8 Assignment**

--1. Create view vw\_updatable\_products (use same query whatever I used in the training)

--Try updating view with below query and see if the product table also gets updated.

--Update query:

--UPDATE updatable\_products SET unit\_price = unit\_price \* 1.1 WHERE units\_in\_stock < 10;

CREATE OR REPLACE VIEW vw\_updatable\_products AS

Create the View vw\_updatable products

CREATE OR REPLACE VIEW vw\_updatable\_products AS

SELECT

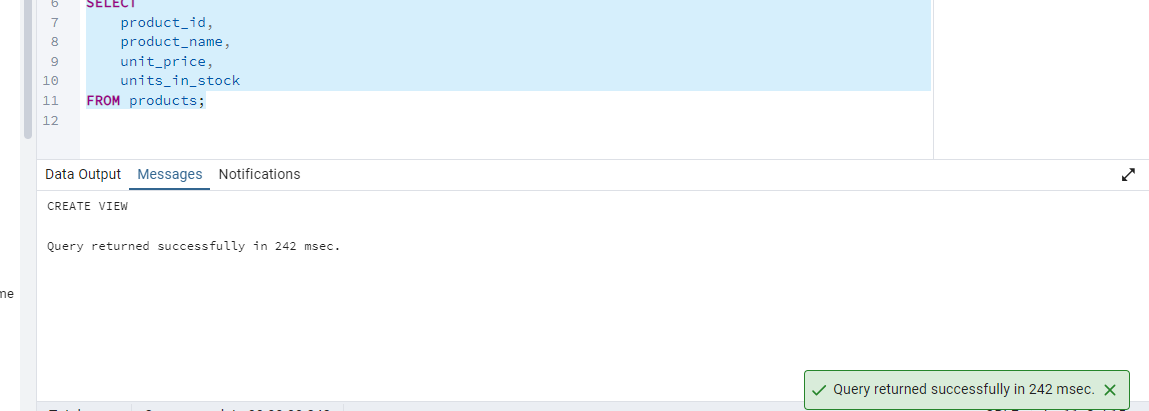
productid,

productname,

unitprice,

units\_in\_stock

FROM products;

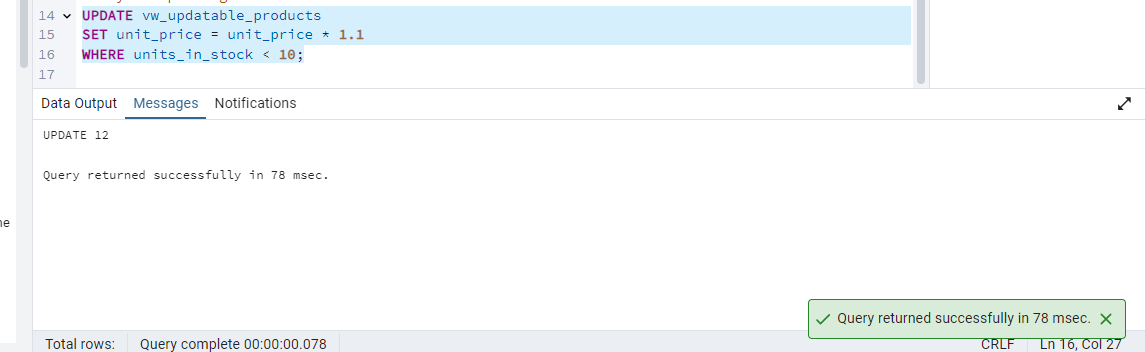


Try to Updating View

UPDATE vw\_updatable\_products

SET unit\_price = unit\_price \* 1.1

WHERE units\_in\_stock < 10;



### Verify if the main table(Products) is updated:

SELECT

product\_id,

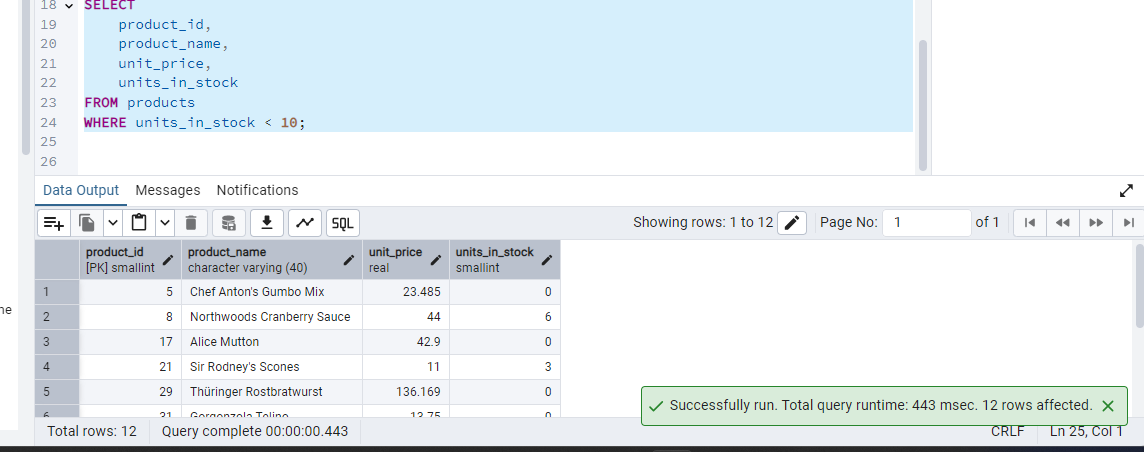
product\_name,

unit\_price,

units\_in\_stock

FROM products

WHERE units\_in\_stock < 10;



2. Transaction:

–Update the product price for products by 10% in category id=1

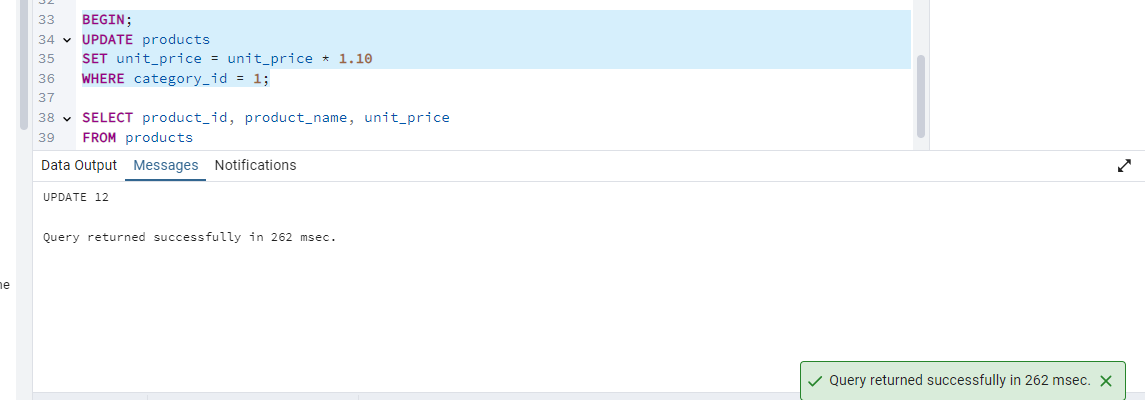
–Try COMMIT and ROLLBACK and observe what happens.

BEGIN;

UPDATE products

SET unit\_price = unit\_price \* 1.10

WHERE category\_id = 1;

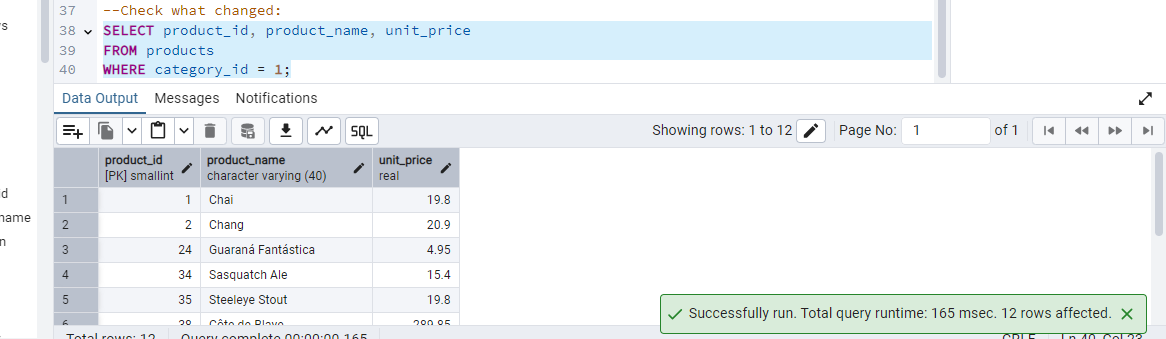


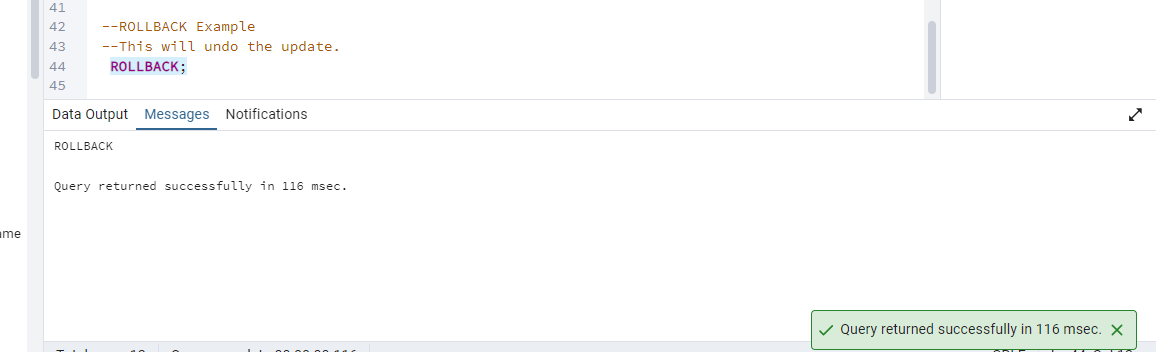
--Check what changed:

SELECT product\_id, product\_name, unit\_price

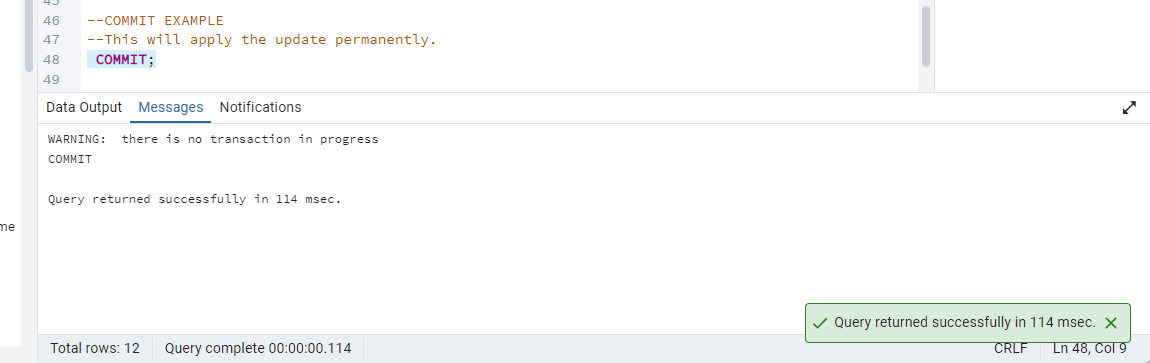
FROM products

WHERE category\_id = 1;





COMMIT



3. Create a regular view which will have below details (Need to do joins):

Employee\_id,

Employee\_full\_name,

Title,

Territory\_id,

territory\_description,

Region\_description

Query:-

CREATE VIEW vw\_employee\_territory\_details AS

SELECT

e.employee\_id,

e.first\_name || ' ' || e.last\_name AS employee\_full\_name,

e.title,

t.territory\_id,

t.territory\_description,

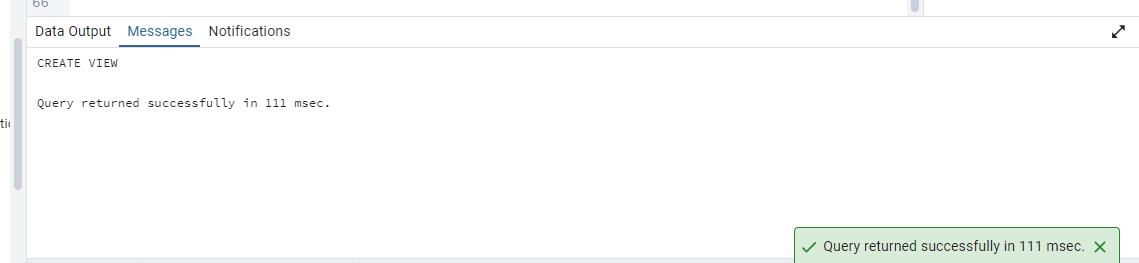
r.region\_description

FROM employees e

JOIN employee\_territories et ON e.employee\_id = et.employee\_id

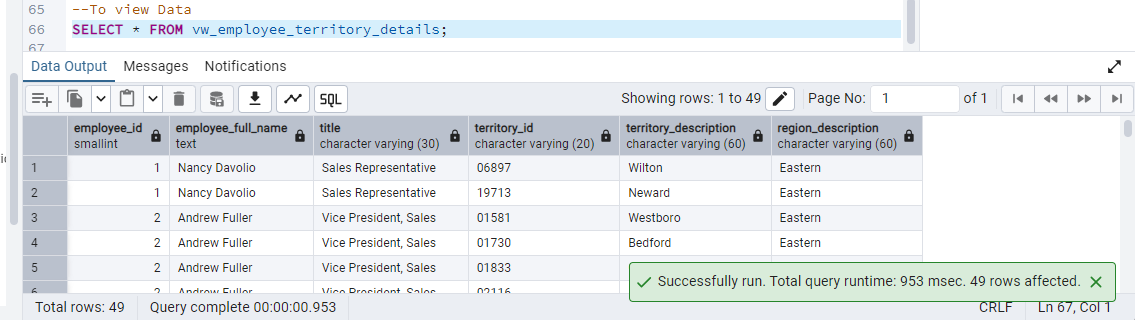
JOIN territories t ON et.territory\_id = t.territory\_id

JOIN region r ON t.region\_id = r.region\_id;



To view Data

SELECT \* FROM vw\_employee\_territory\_details;



4. Create a recursive CTE based on Employee Hierarchy

Query:

WITH RECURSIVE employee\_hierarchy AS (

-- Base case: top-level employees (no manager)

SELECT

employee\_id,

first\_name || ' ' || last\_name AS employee\_name,

reports\_to AS manager\_id,

1 AS level

FROM employees

WHERE reports\_to IS NULL

UNION ALL

-- Recursive case: employees who report to others

SELECT

e.employee\_id,

e.first\_name || ' ' || e.last\_name AS employee\_name,

e.reports\_to AS manager\_id,

eh.level + 1

FROM employees e

INNER JOIN employee\_hierarchy eh ON e.reports\_to = eh.employee\_id

)

SELECT \* FROM employee\_hierarchy

ORDER BY level, manager\_id;

