**ASSIGNMENT - 8**

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

**Try updating the view with the 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;**

**QUERY**

--Create a view

create view vw\_updatable\_products as

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

from products;

--update query

update vw\_updatable\_products

set unit\_price = unit\_price \* 1.1

where units\_in\_stock < 10;

--Verifying the products table after updating the view

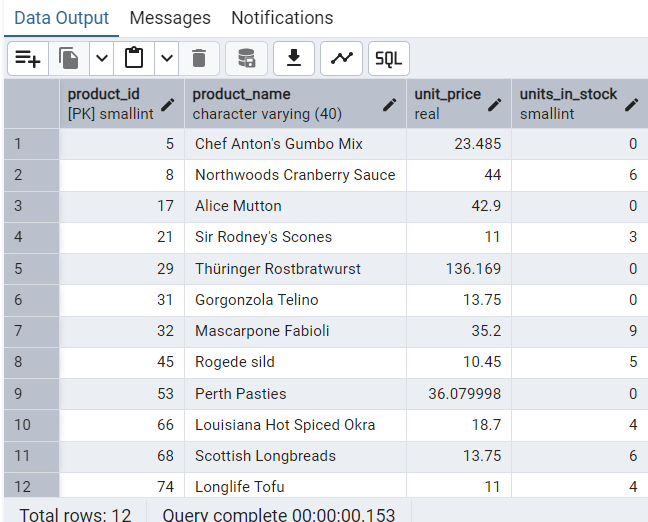
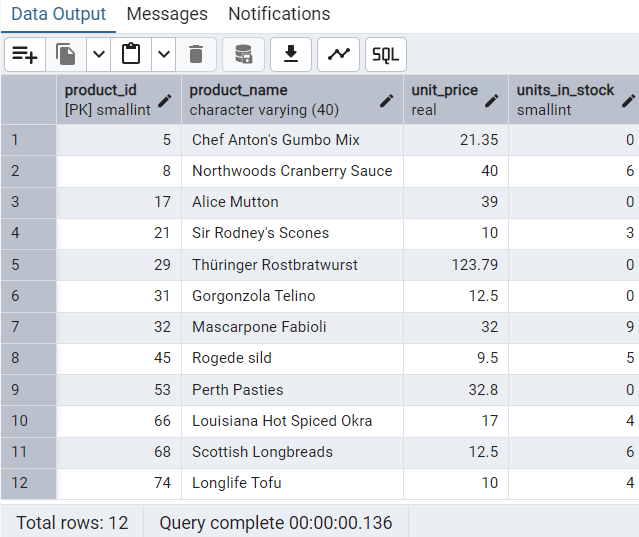
select product\_id,product\_name,unit\_price,units\_in\_stock

from products

where units\_in\_stock<10;

**OUTPUT**

*AFTER UPDATING*  *BEFORE UPDATING*

**–Qn2.Transaction: Update the product price for products by 10% in category id=1**

**Try COMMIT and ROLLBACK and observe what happens.**

**QUERY**

begin;

update products

set unit\_price = unit\_price \* 1.1

where category\_id = 1;

-- View changes before committing

select product\_id,product\_name,unit\_price

from products

where category\_id = 1;

--committing the transaction

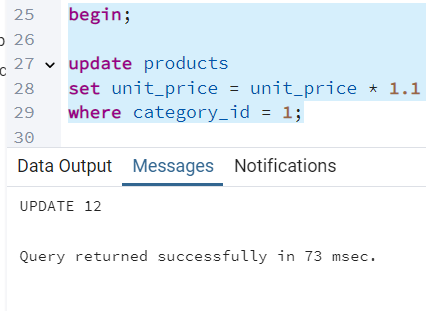
COMMIT;

--rollback

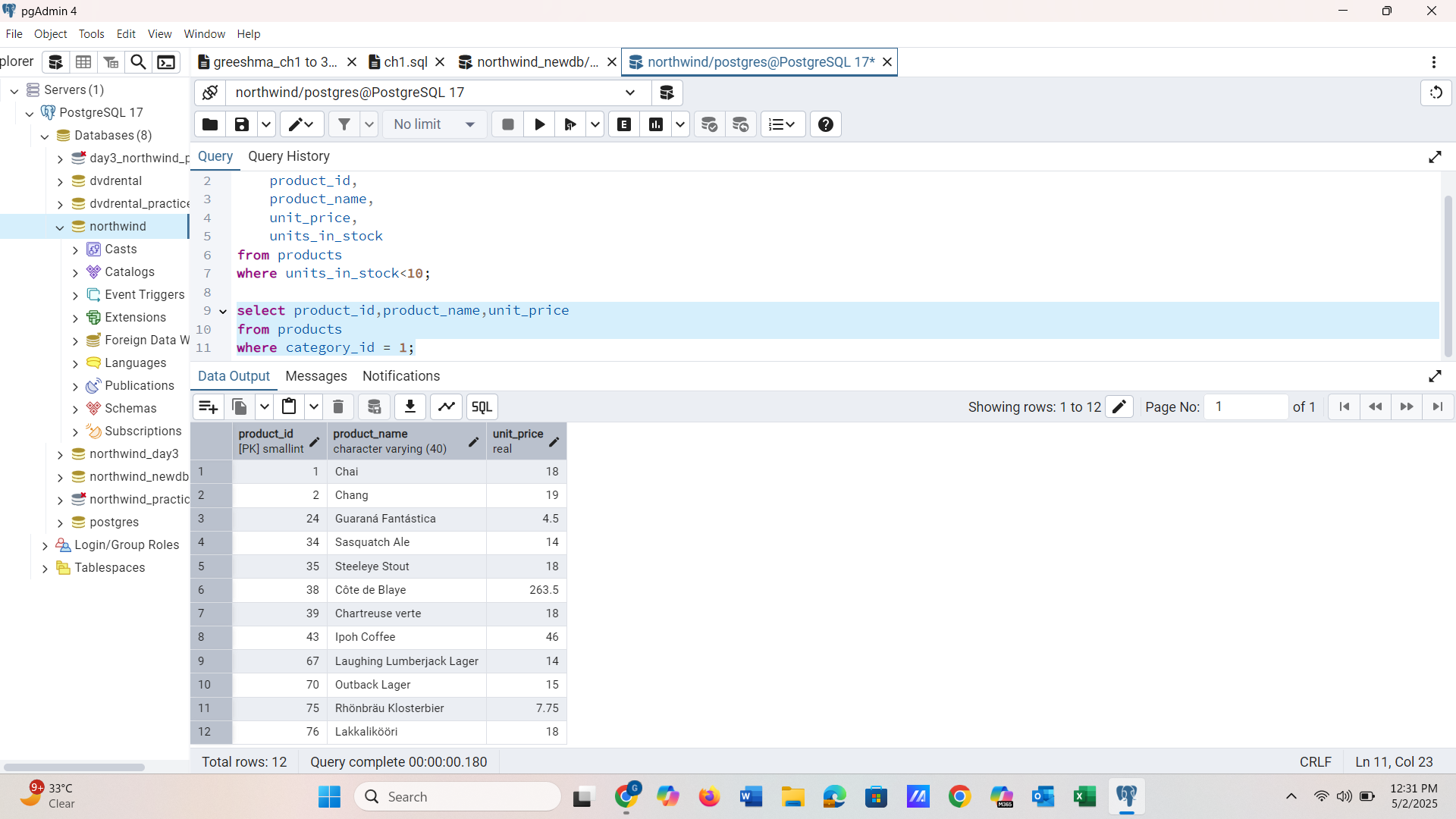
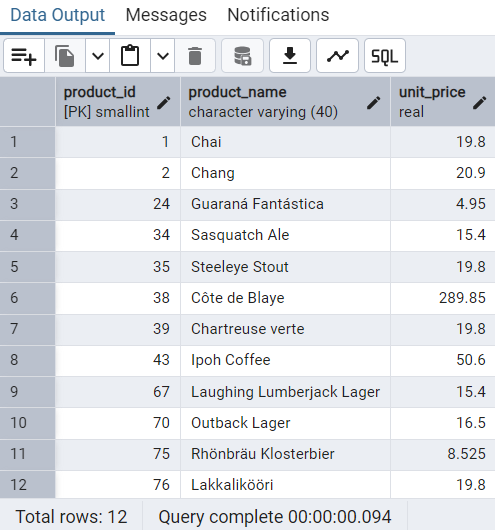
ROLLBACK;

**OUTPUT**

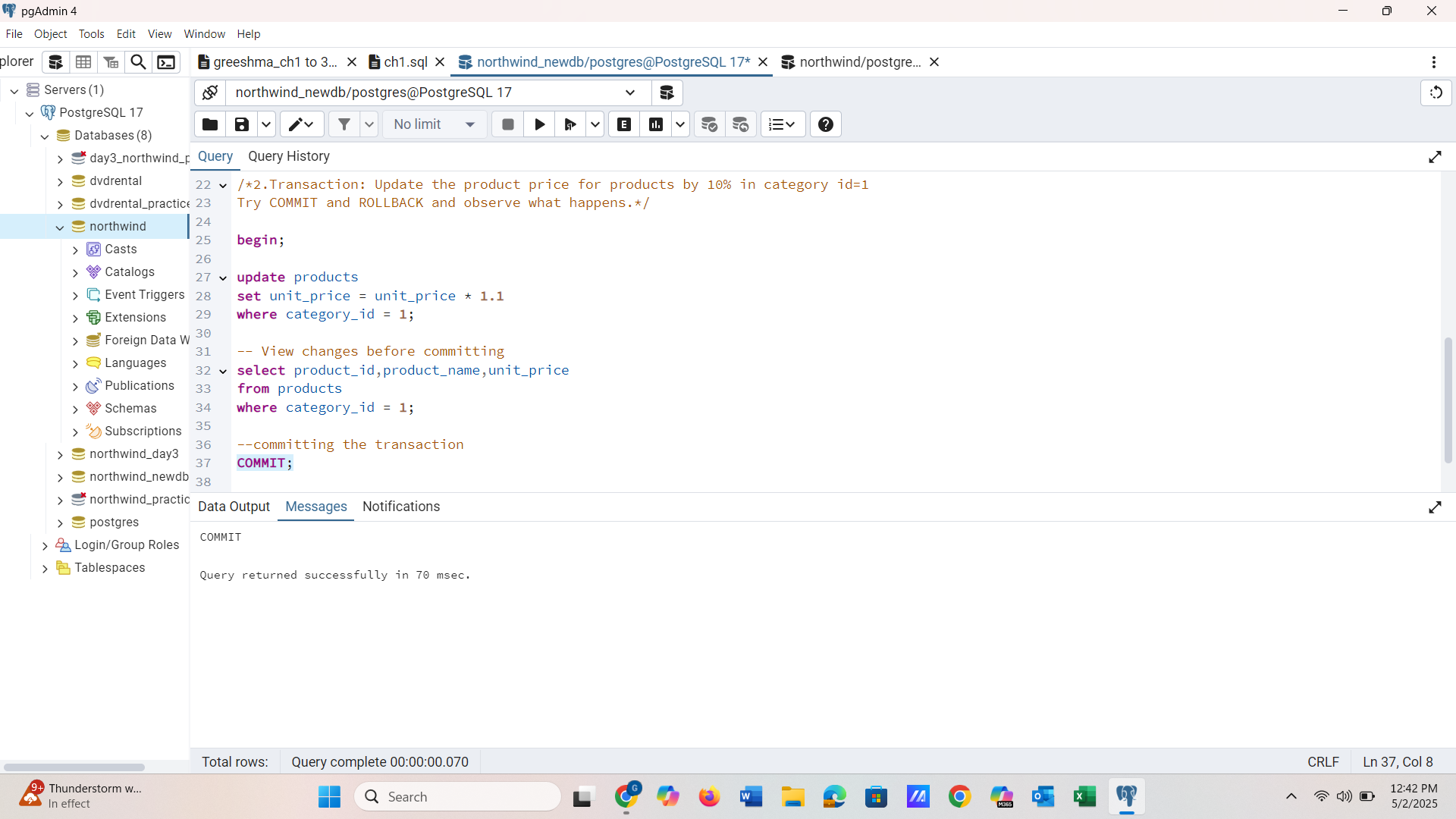
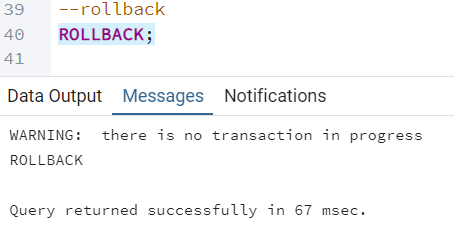
*Updating category\_id*



*Before updating category\_id*  *After updating category\_id*

*Committing Transaction*   *Rollback*

**–Qn3.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\_territories as

select

e.employee\_id,

concat(e.first\_name, ' ', e.last\_name) as employee\_full\_name,

e.title,

et.territory\_id,

t.territory\_description,

r.region\_description

from employees e

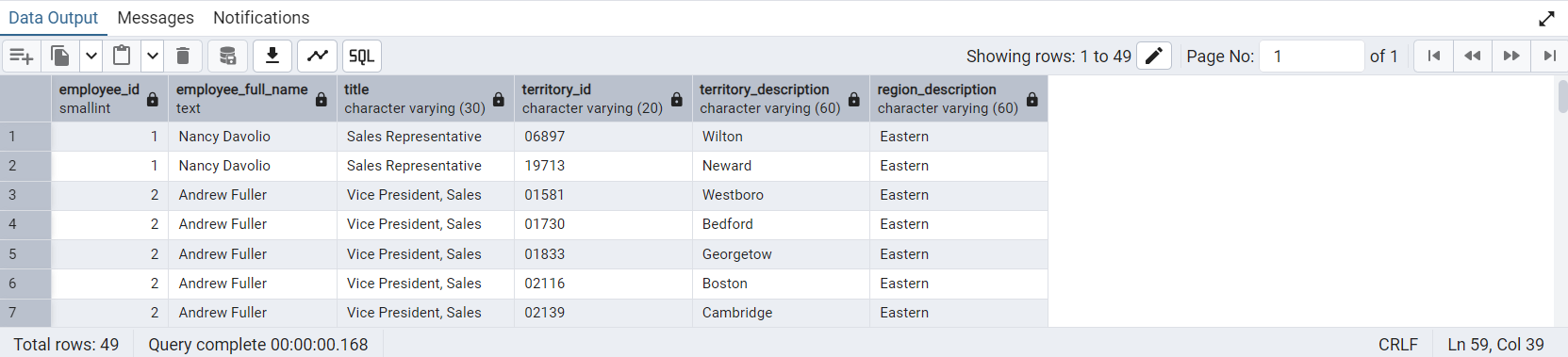
join employee\_territories et on e.employee\_id = et.employee\_id

join territories t on et.territory\_id = t.territory\_id

join region r on t.region\_id = r.region\_id;

select \* from vw\_employee\_territories;

**OUTPUT**



**–Qn4.Create a recursive CTE based on Employee Hierarchy**

**QUERY**

*-- Anchor member: top-level employees (no manager)*

with recursive employee\_hierarchy as (

select employee\_id,first\_name,last\_name,reports\_to,1 as level

from employees

where reports\_to is null

union all

*-- Recursive member: employees reporting to others*

select e.employee\_id,e.first\_name,e.last\_name,e.reports\_to,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, employee\_id;

**OUTPUT**

