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;

select \* from products;

create view update\_products\_view as

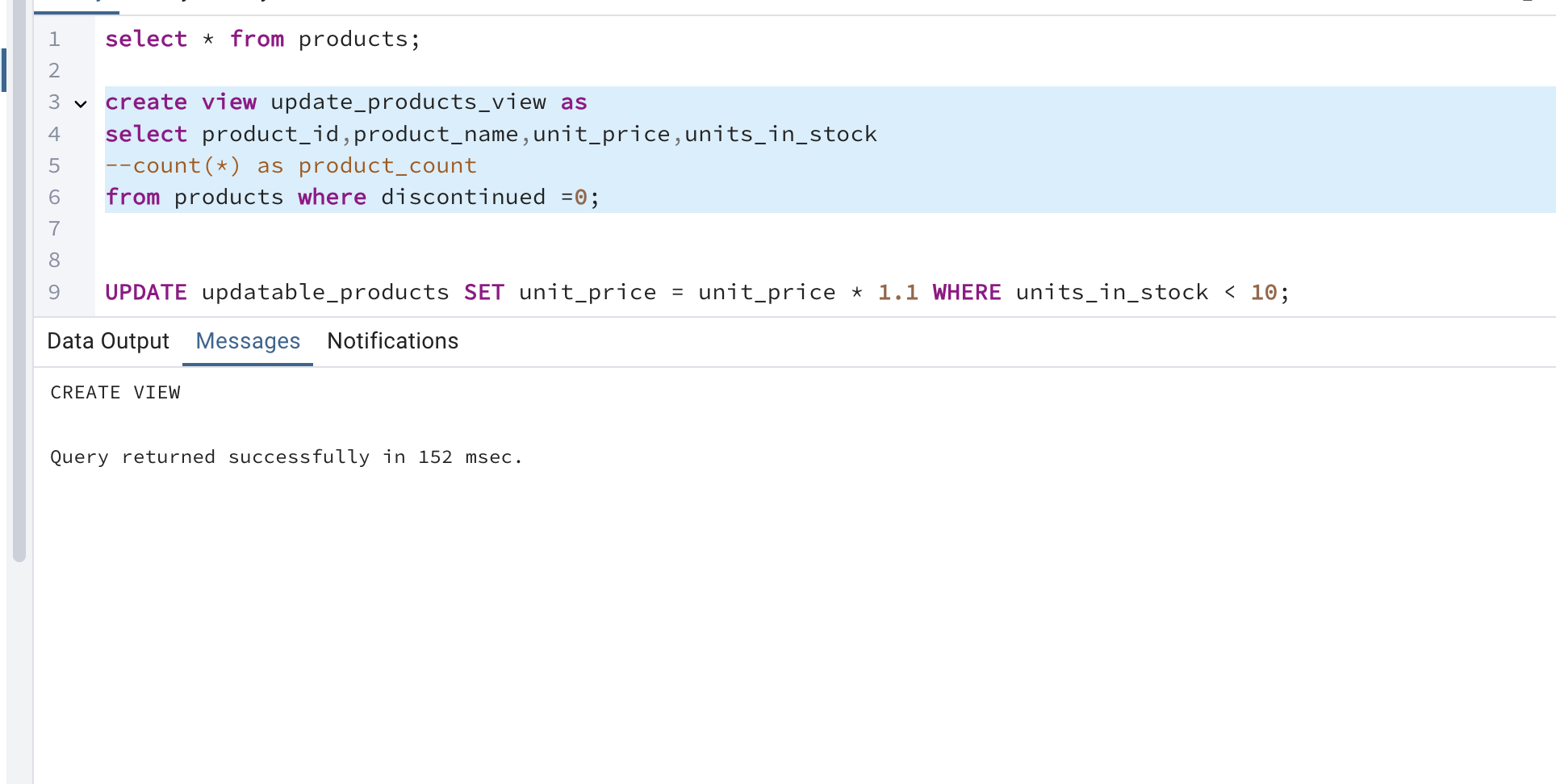
select product\_id,product\_name,unit\_price,units\_in\_stock

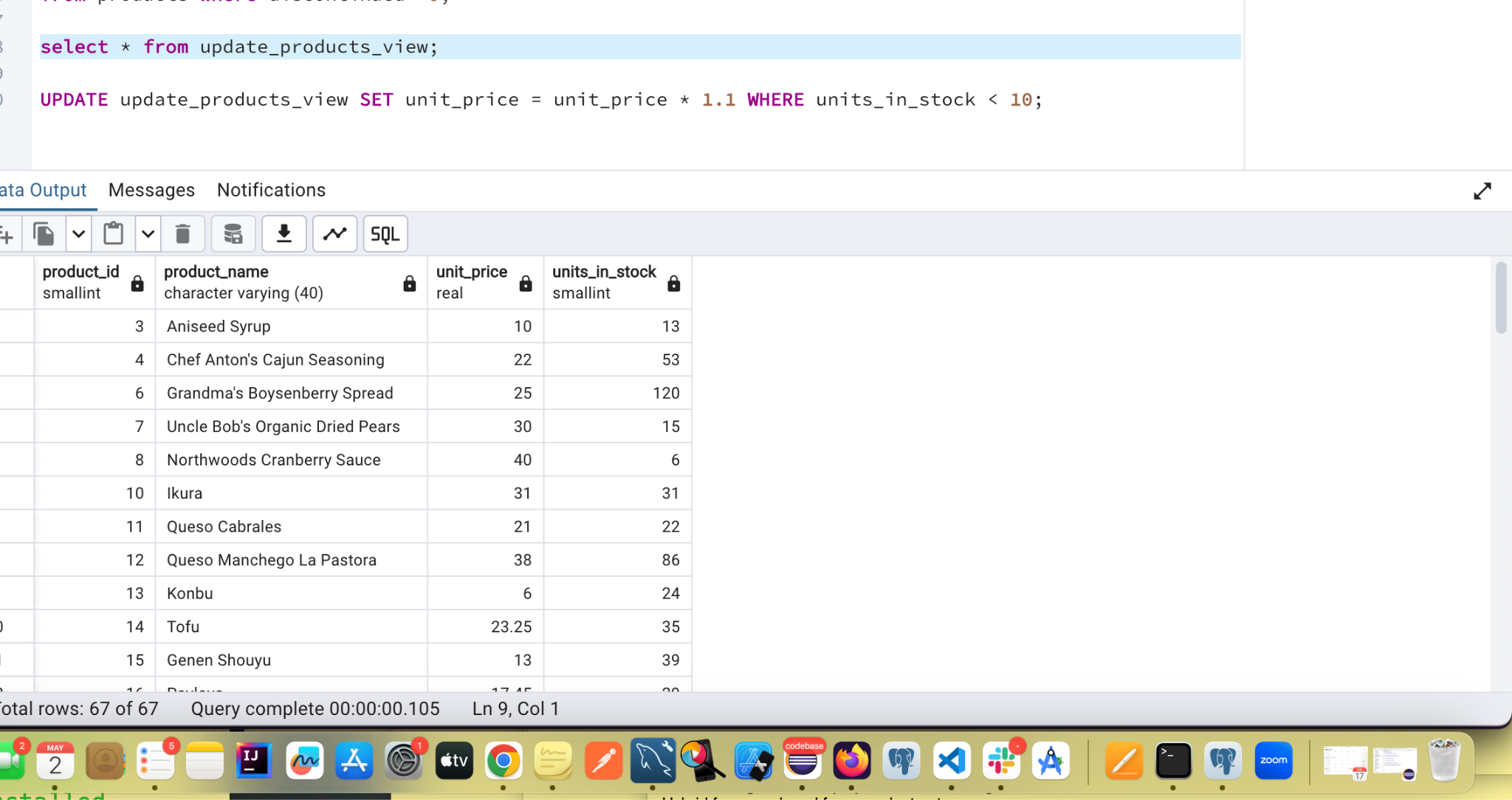
--count(\*) as product\_count

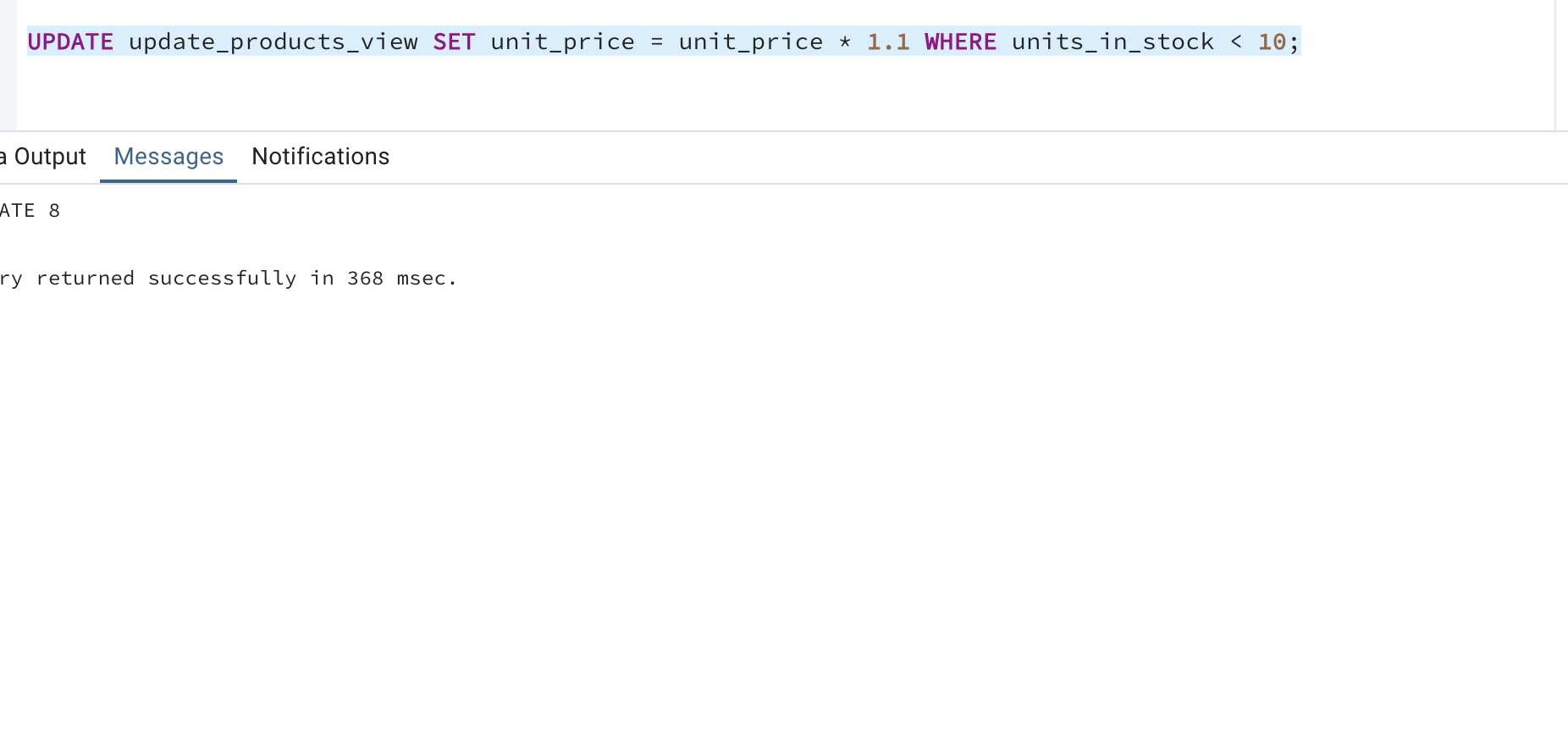
from products where discontinued =0;

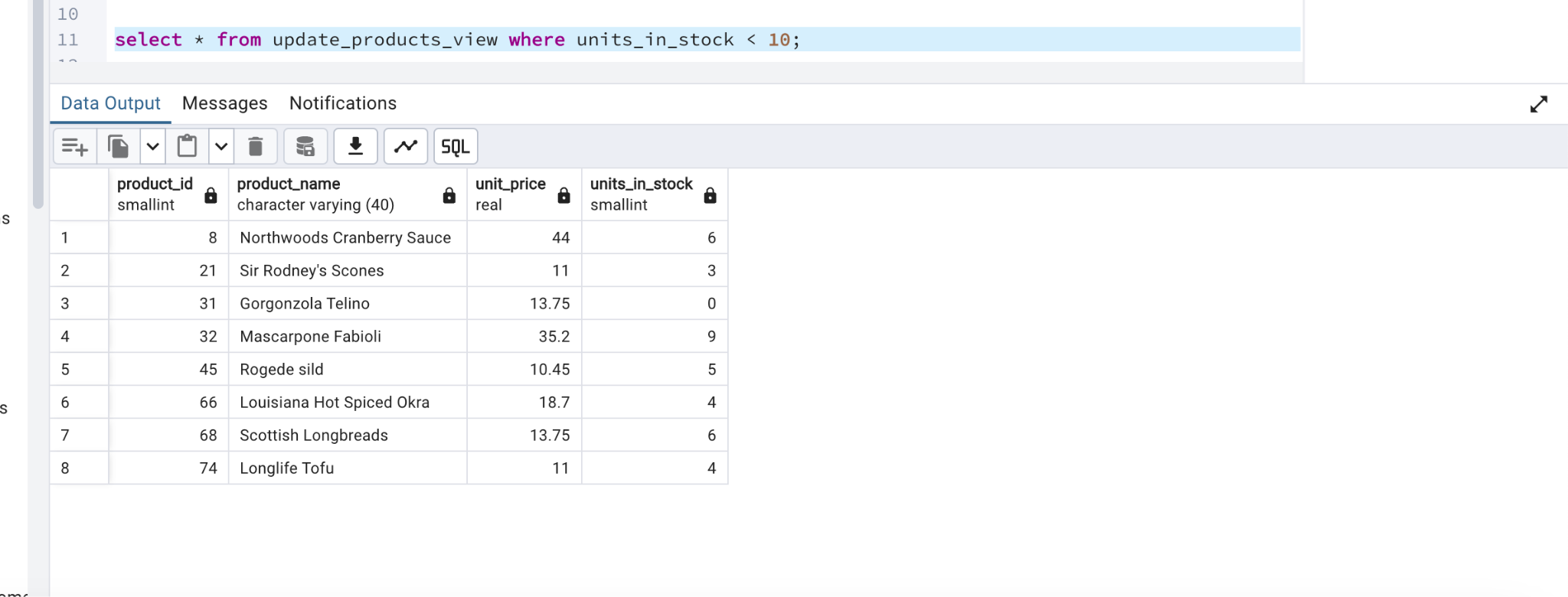
UPDATE update\_products\_view SET unit\_price = unit\_price \* 1.1 WHERE units\_in\_stock < 10;

select \* from update\_products\_view 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\*1.1 where category\_id=1;

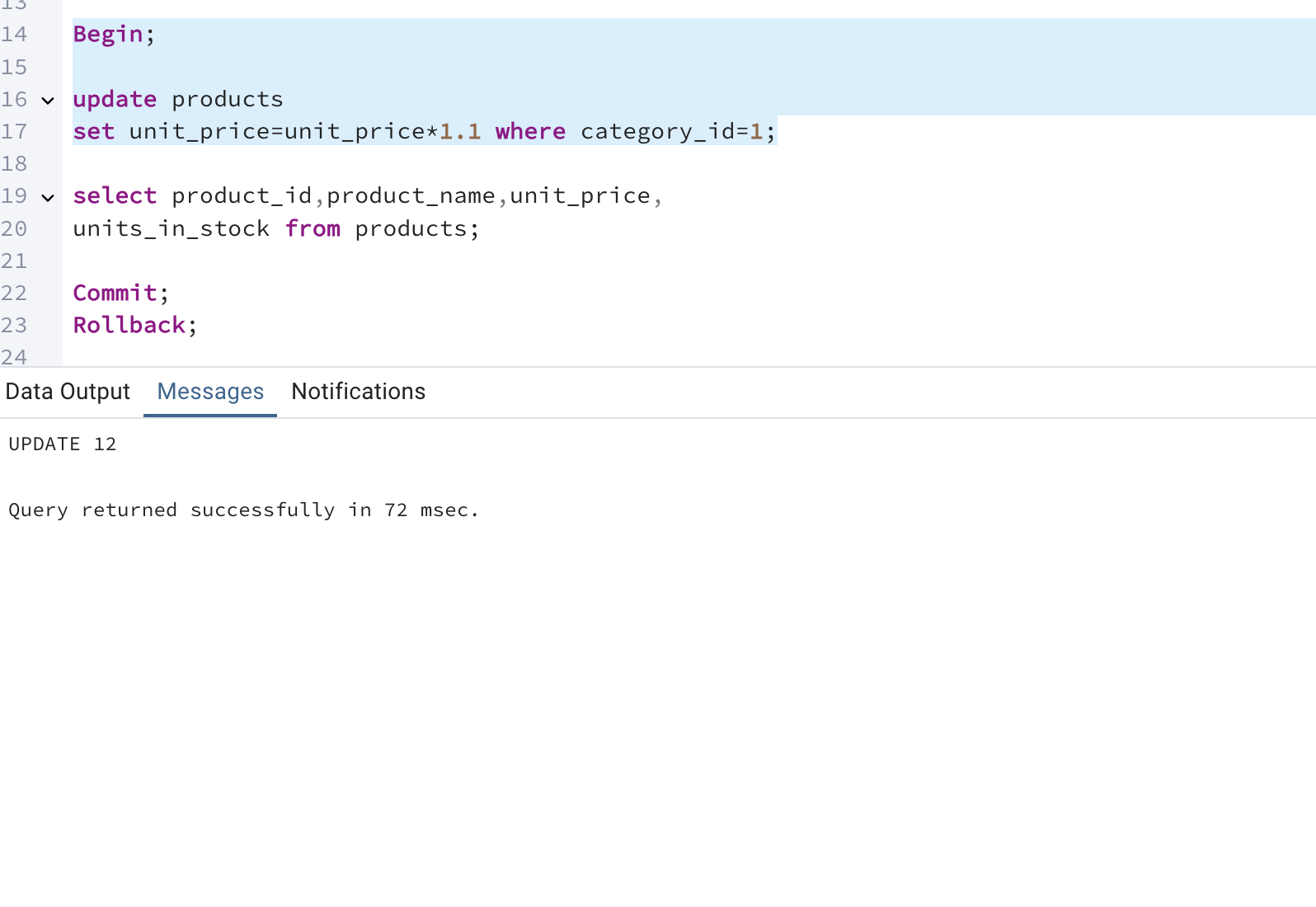
select product\_id,product\_name,unit\_price,

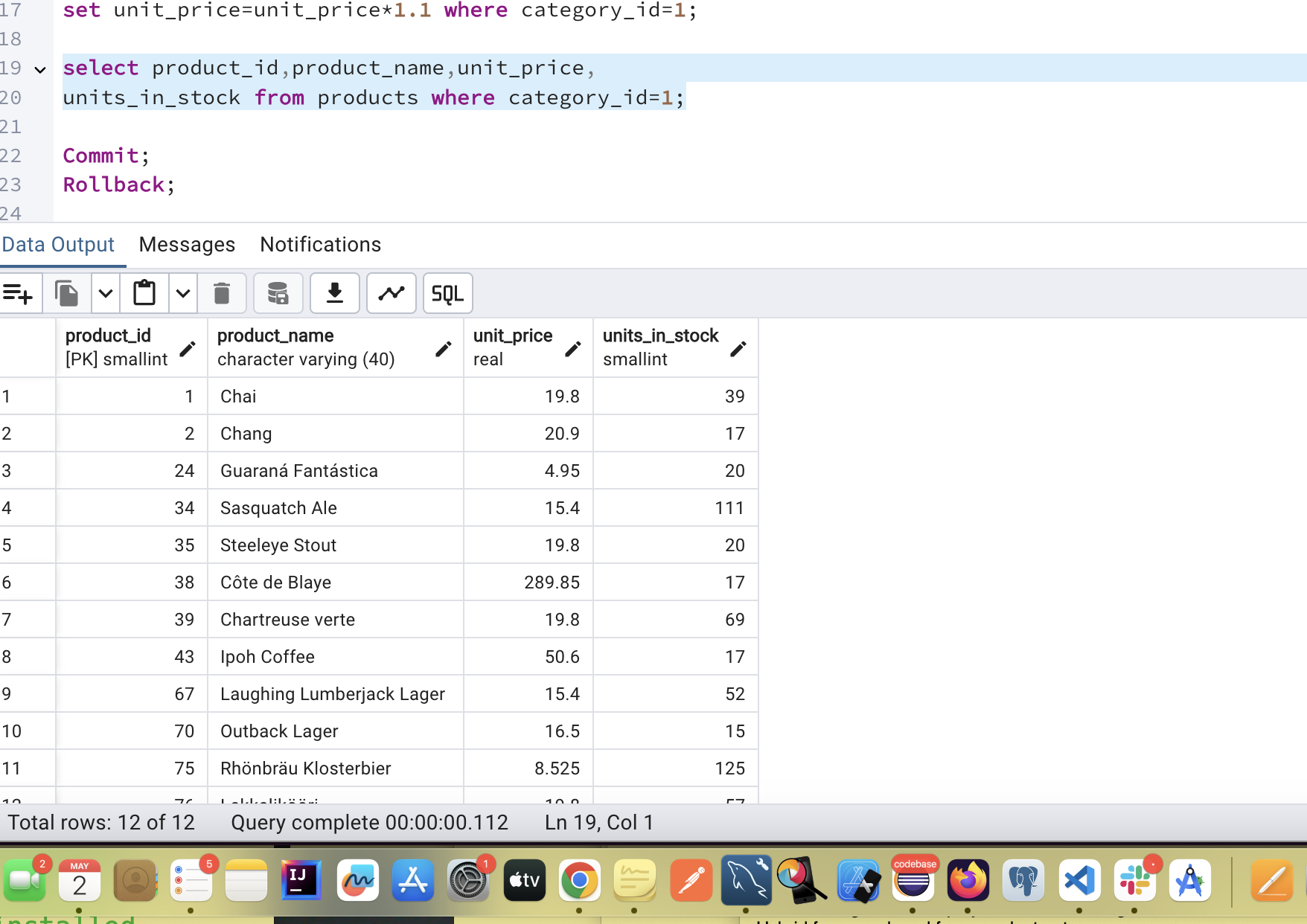
units\_in\_stock from products where category\_id=1;

Commit;

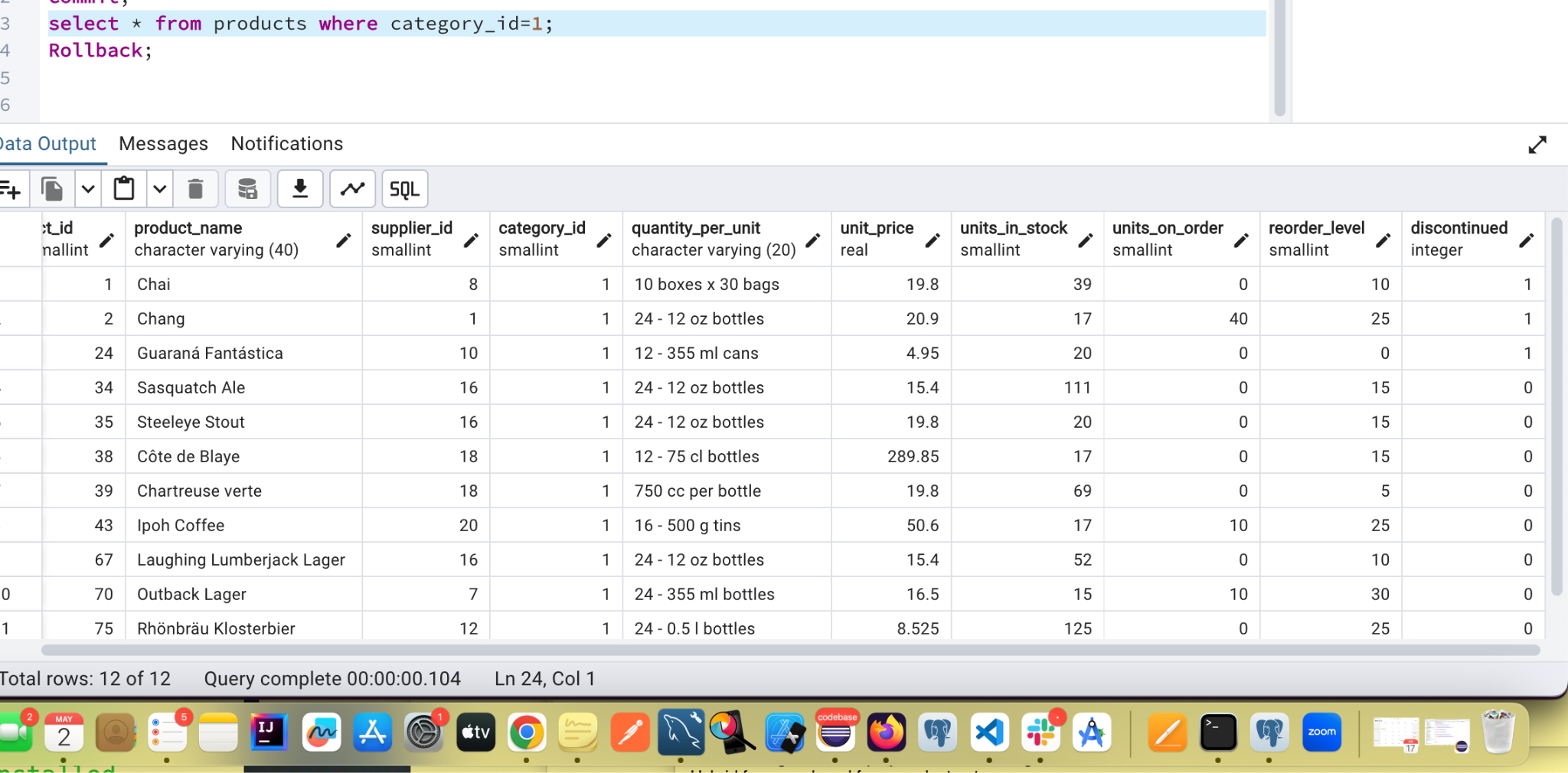
select \* from products where category\_id=1;

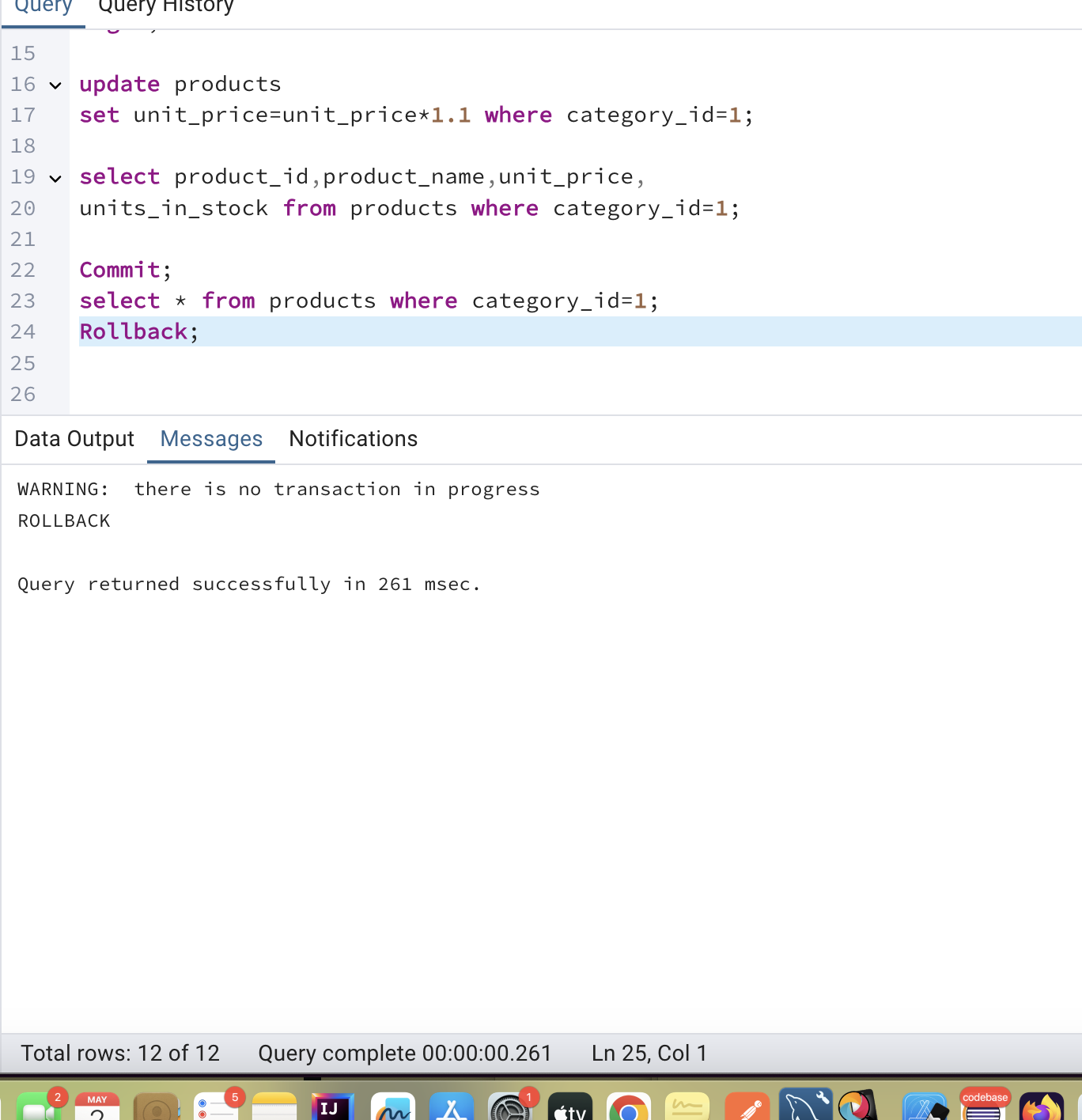
Rollback;



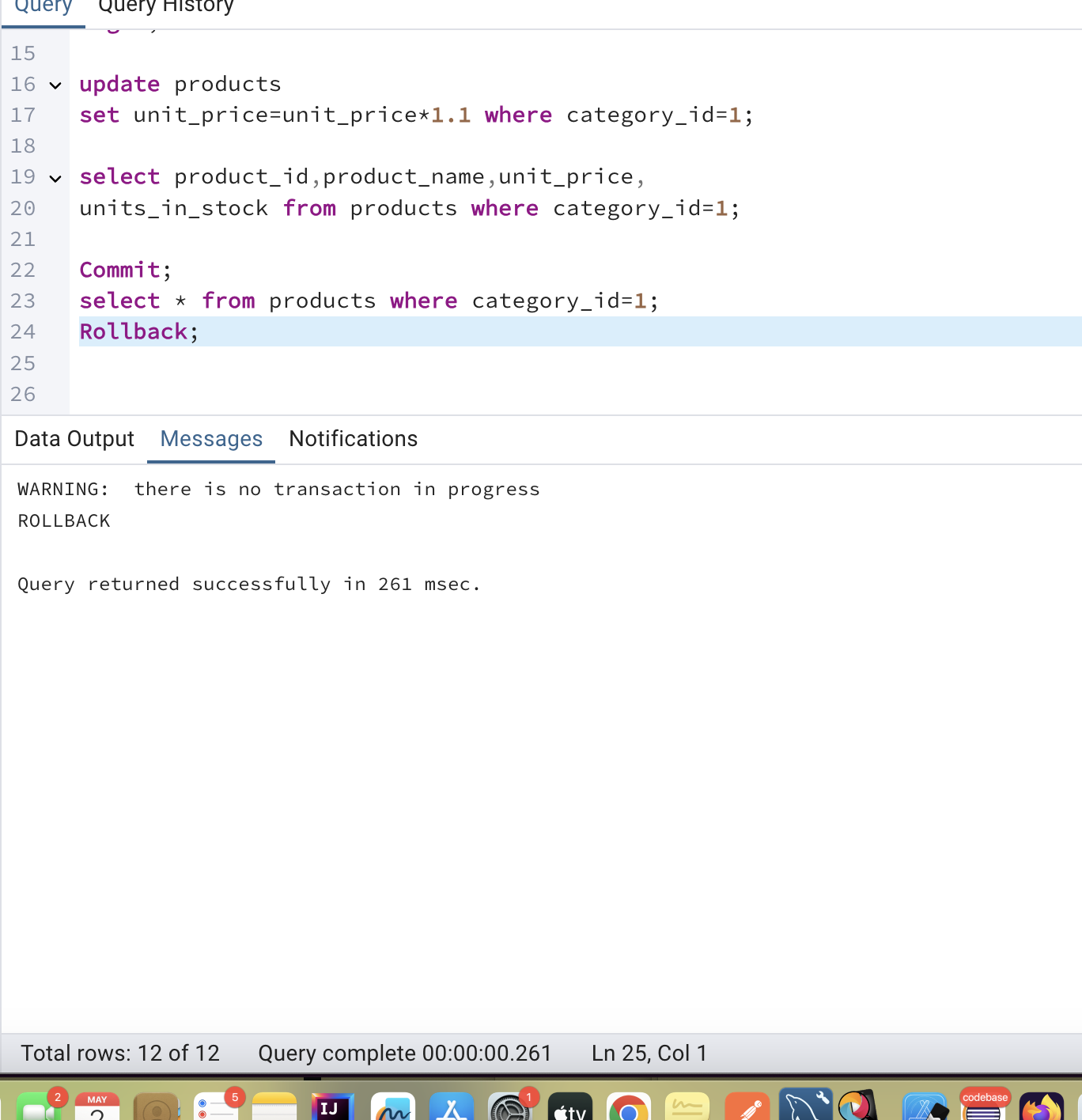








\



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..

create view employee\_terrritory\_region as

select e.employee\_id,

e.last\_name || '' || e.first\_name as employee\_fullname,

e.title,

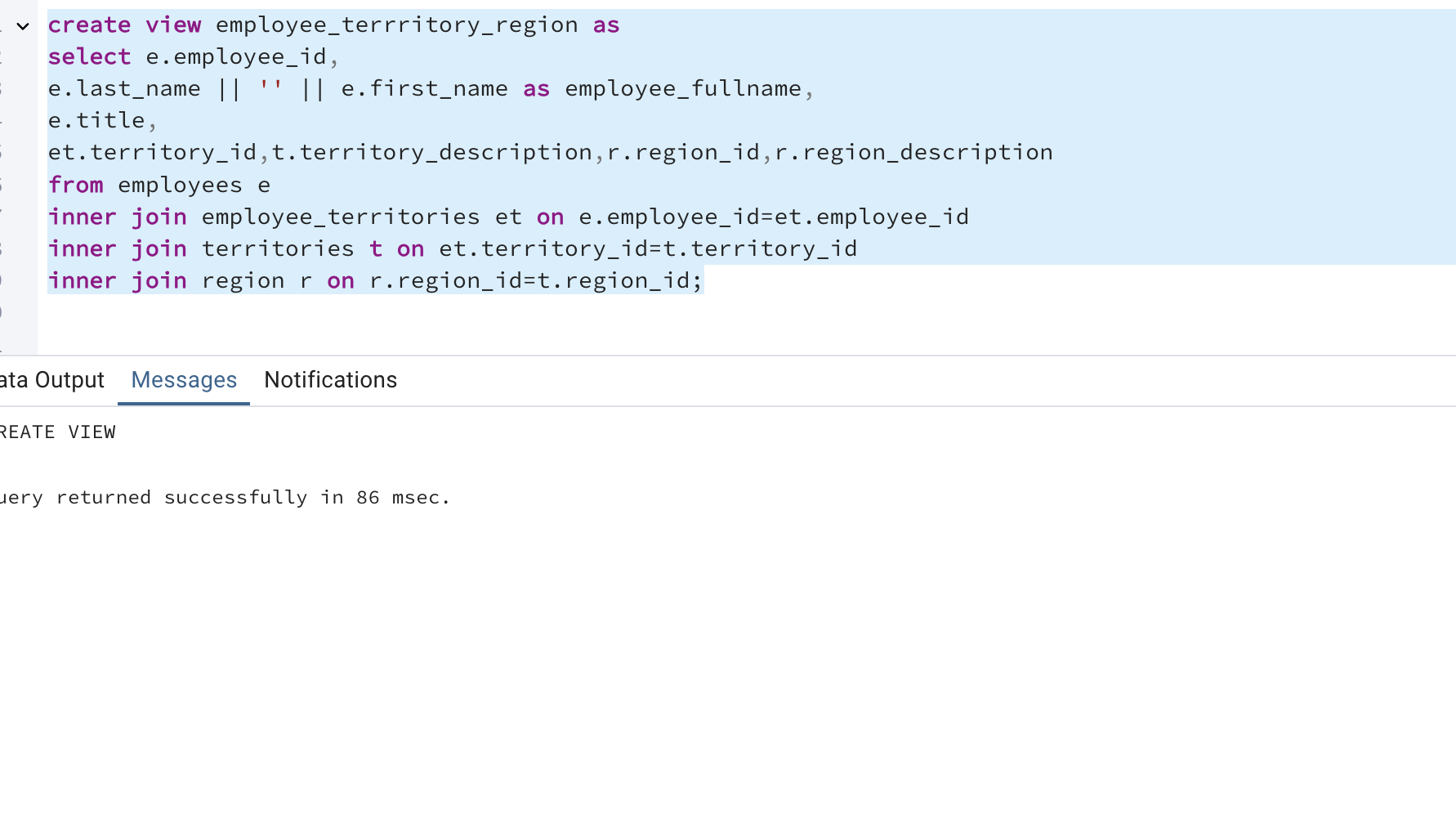
et.territory\_id,t.territory\_description,r.region\_id,r.region\_description

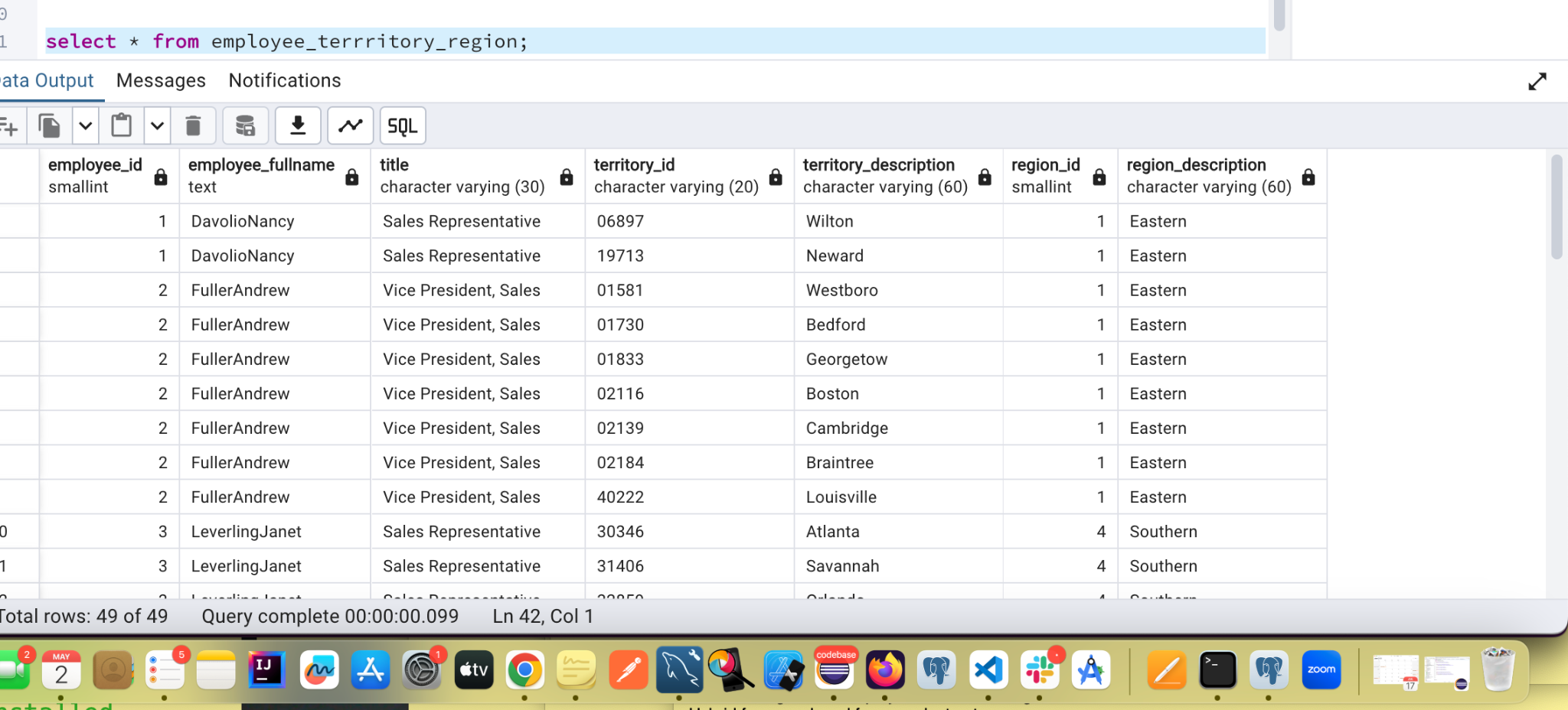
from employees e

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

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

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





4. Create a recursive CTE based on Employee Hierarchy

with recursive cte\_employees\_hierarchy as

( select employee\_id,

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

reports\_to,

1 as level

from employees

where reports\_to is NULL

UNION ALL

select e.employee\_id,

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

e.reports\_to,eh.level +1

from employees e

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

)

SELECT \* FROM cte\_employees\_hierarchy

ORDER BY level, reports\_to, employee\_id;

