SQL Assignment 8

						\sim	$\overline{}$	п
	\sim	С.	-	ξr	\sim	\ 1	1	-1
\mathbf{r}	()		ı v	, ,	_	71	. ,	-1
	${}^{\circ}$	_	U۶	S I I	◡.	_	v	ı

1.	Create view vw_updatable_products (use same query whatever I used in the training)	1
2.	Transaction: Update the product price for products by 10% in category id=1	2
3.	Create a regular view which will have below details (Need to do joins):	4
4	Create a recursive CTE based on Employee Hierarchy	5

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;

QUERY:

CREATE VIEW vw_updatable_products AS

SELECT product_id, product_name, unit_price, units_in_stock, discontinued

FROM products

WHERE discontinued = 0

WITH CHECK OPTION

UPDATE vw_updatable_products SET unit_price = unit_price * 1.1 WHERE units in stock < 10;

OUTPUT:

Data Output Messages Notifications

=+					
	product_id smallint	product_name character varying (40)	unit_price real	units_in_stock smallint	discontinued integer
1	3	Aniseed Syrup	10	13	0
2	4	Chef Anton's Cajun Seasoning	22	53	0
3	6	Grandma's Boysenberry Spread	25	120	0
4	7	Uncle Bob's Organic Dried Pears	30	15	0
5	8	Northwoods Cranberry Sauce	40	6	0
6	10	Ikura	31	31	0
7	11	Queso Cabrales	21	22	0
8	12	12 Queso Manchego La Pastora		86	0

Data Output Messages Notifications					
=+ L v L v L SQL					
	product_id smallint	product_name character varying (40)	unit_price real	units_in_stock smallint	discontinued integer
59	77	Original Frankfurter grüne Soße	13	32	0
60	8	Northwoods Cranberry Sauce	44	6	0
61	21	Sir Rodney's Scones	11	3	0
62	31	Gorgonzola Telino	13.75	0	0
63	32	Mascarpone Fabioli	35.2	9	0
64	45	Rogede sild	10.45	5	0
65	66	Louisiana Hot Spiced Okra	18.7	4	0
66	68	Scottish Longbreads	13.75	6	0
67	74	Lonalife Tofu	11	4	n
Total	rows: 67	uery complete 00:00:00.120			

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

Try COMMIT and ROLLBACK and observe what happens.

QUERY:

SELECT * FROM products WHERE category_id = 1;

BEGIN;

UPDATE products

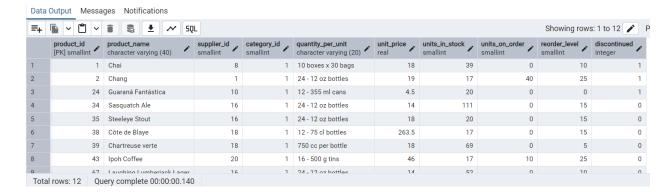
SET unit price = unit price * 1.10

WHERE category_id = 1;

ROLLBACK;

COMMIT;

OUTPUT:







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 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

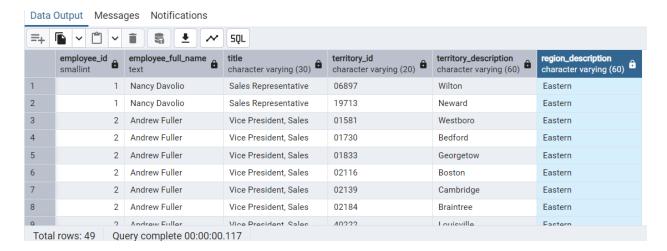
JOIN employee_territories et ON e.employee_id = et.employee_id
```

JOIN territories t ON et. territory id = t. territory id

JOIN region r ON t.region_id = r.region_id;

SELECT * FROM vw_employee_territory;

OUTPUT:



4. Create a recursive CTE based on Employee Hierarchy

QUERY:

WITH RECURSIVE cte_employee_hierarchy AS (

SELECT employee_id, first_name, last_name, reports_to, 0 AS level

FROM employees e

WHERE reports to IS NULL

UNION ALL

SELECT e.employee_id, e.first_name, e.last_name, e.reports_to, eh.level+1

FROM employees e

JOIN cte employee hierarchy eh

ON eh.employee_id = e.reports_to)

SELECT level, employee_id,

e.first_name || ' ' || e.last_name AS employee_full_name

FROM cte_employee_hierarchy

ORDER BY level, employee_id;

OUTPUT:

Data Output Messages Notifications				
=+		v i 8	♣ ~ SQL	
	level integer	employee_id smallint	employee_full_name text	
1	0	2	Andrew Fuller	
2	1	1	Nancy Davolio	
3	1	3	Janet Leverling	
4	1	4	Margaret Peacock	
5	1	5	Steven Buchanan	
6	1	8	Laura Callahan	
7	2	6	Michael Suyama	
8	2	7	Robert King	
Total	rows: 9	Query complet	e 00:00:00.145	