/\*Q1. 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 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;

select \* from vw\_updatable\_products where units\_in\_stock <10;

select \* from products where discontinued = 0 and units\_in\_stock <10;

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

/\*

2. Transaction:

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

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

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

A screenshot of a computer screen

AI-generated content may be incorrect.

begin;

update products

set unit\_price =unit\_price\*1.1

where category\_id =1;

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

A screenshot of a computer

AI-generated content may be incorrect.

rollback;

A screenshot of a computer screen

AI-generated content may be incorrect.

commit;

/\*Q3. 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\_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 r.region\_id =t.region\_id

select \* from Employee\_details

A screenshot of a computer

AI-generated content may be incorrect.

/\*4. Create a recursive CTE based on Employee Hierarchy\*/

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 e.reports\_to = eh.employee\_id)

select level, employee\_id, reports\_to from cte\_employee\_hierarchy

A screenshot of a computer

AI-generated content may be incorrect.