**NUMPY NINJA SQL BOOTCAMP APRIL 2025**

**DAY 3 SQL ASSIGNMENT**

**Greeshma Ravula DA 164**

**QUESTION 1)**

Update the categoryName From “Beverages” to "Drinks" in the categories table.

**QUERY DETAILS**

UPDATE categories

SET categoryname='Drinks'

WHERE categoryname='Beverages';

**SCREENSHOTS**

****

**QUESTION 2)**

Insert into shipper new record (give any values) Delete that new record from shippers table.

**QUERY DETAILS**

INSERT INTO shippers(shipperid,companyname) VALUES (4,'DTC shipping')

DELETE FROM shippers WHERE shipperid = 4;

**SCREENSHOTS**



**QUESTION 3)**

Update categoryID=1 to categoryID=1001. Make sure related products update their categoryID too. Display the both category and products table to show the cascade.

Delete the categoryID= “3” from categories. Verify that the corresponding records are deleted automatically from products.

**QUERY DETAILS**

*—ALTERING THE PRODUCTS TABLE*

ALTER TABLE products

DROP CONSTRAINT IF EXISTS products\_categoryid\_fkey;

ALTER TABLE products

ADD CONSTRAINT products\_categoryid\_fkey

FOREIGN KEY (categoryID)

REFERENCES categories(categoryID)

ON UPDATE CASCADE

ON DELETE CASCADE;

*--UPDATING CATEGORYID*

UPDATE categories

SET categoryid = 1001

WHERE categoryid = 1;

*--CASCADE ON CATEGORY AND PRODUCTS TABLE*

select \* from categories



select \* from products;



*-- Delete category categoryid = 3*

*-- Throws Error message so we should alter the table*

*– ALTERING Order\_details*

ALTER TABLE order\_details

DROP CONSTRAINT IF EXISTS order\_details\_productid\_fkey;

ALTER TABLE order\_details

ADD CONSTRAINT order\_details\_productid\_fkey

FOREIGN KEY (productid)

REFERENCES products(productid)

ON DELETE CASCADE;

– DELETING CATEGORYID=3

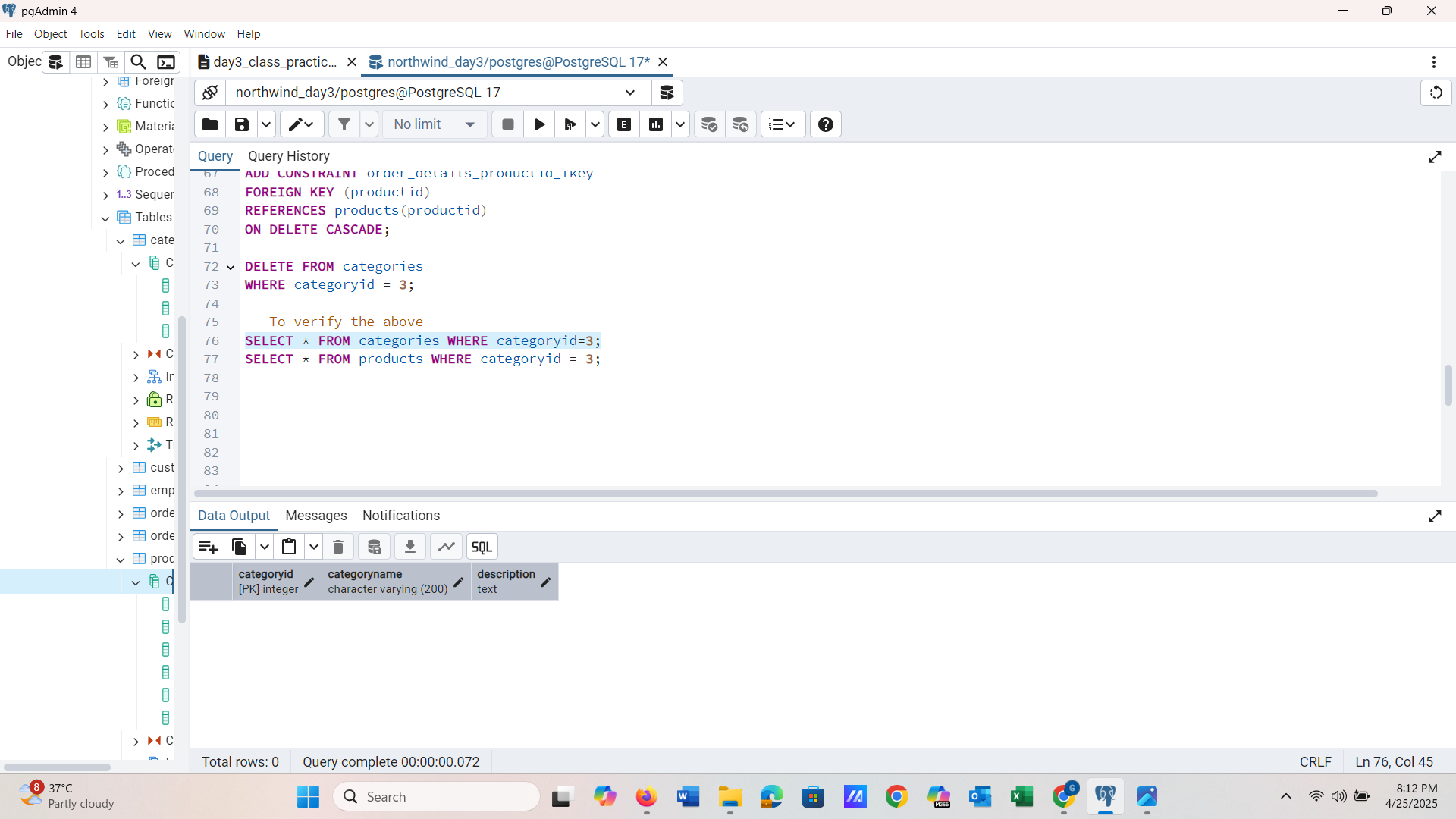
DELETE FROM categories

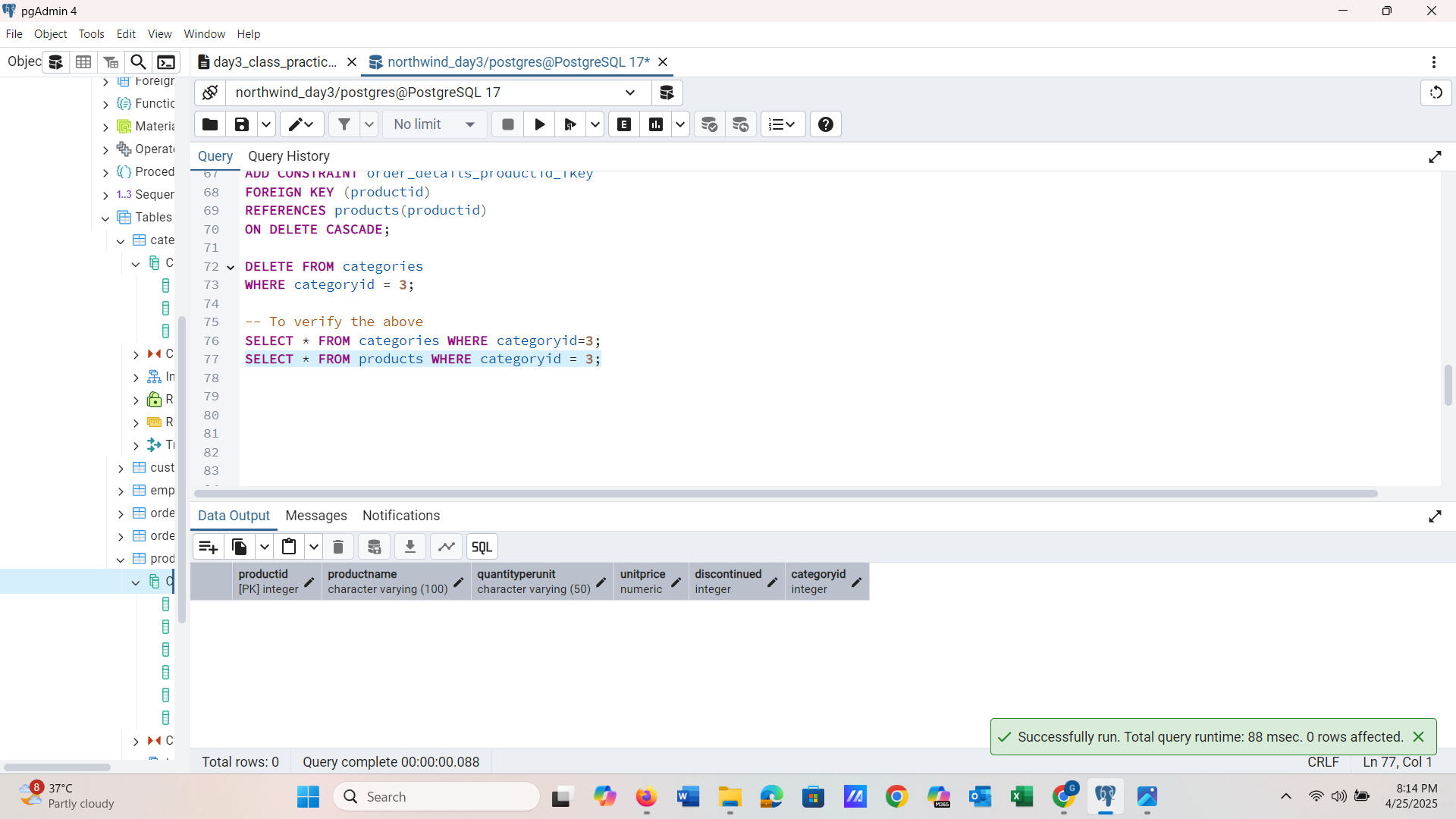
WHERE categoryid = 3;

-- To verify the above

SELECT \* FROM categories WHERE categoryid=3;

SELECT \* FROM products WHERE categoryid = 3;





**QUESTION 4)**

Delete the customer = “VINET” from customers. Corresponding customers in orders table should be set to null

**QUERY DETAILS**

– ALTERING ORDERS TABLE

ALTER TABLE orders

DROP CONSTRAINT IF EXISTS orders\_customerid\_fkey;

ALTER TABLE orders

ADD CONSTRAINT orders\_customerid\_fkey

FOREIGN KEY (customerid)

REFERENCES customers(customerid)

ON DELETE SET NULL;

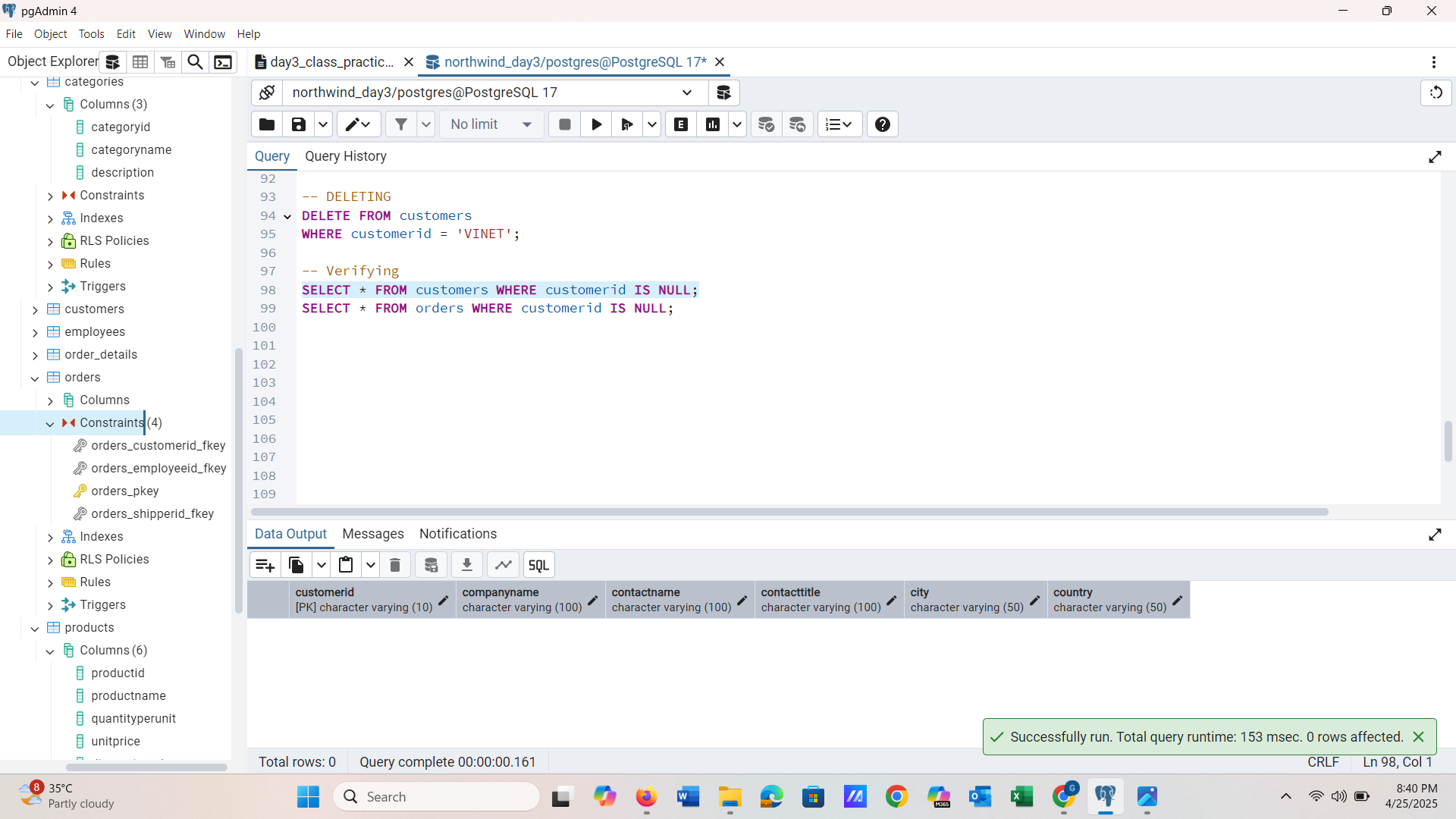
-- DELETING

DELETE FROM customers

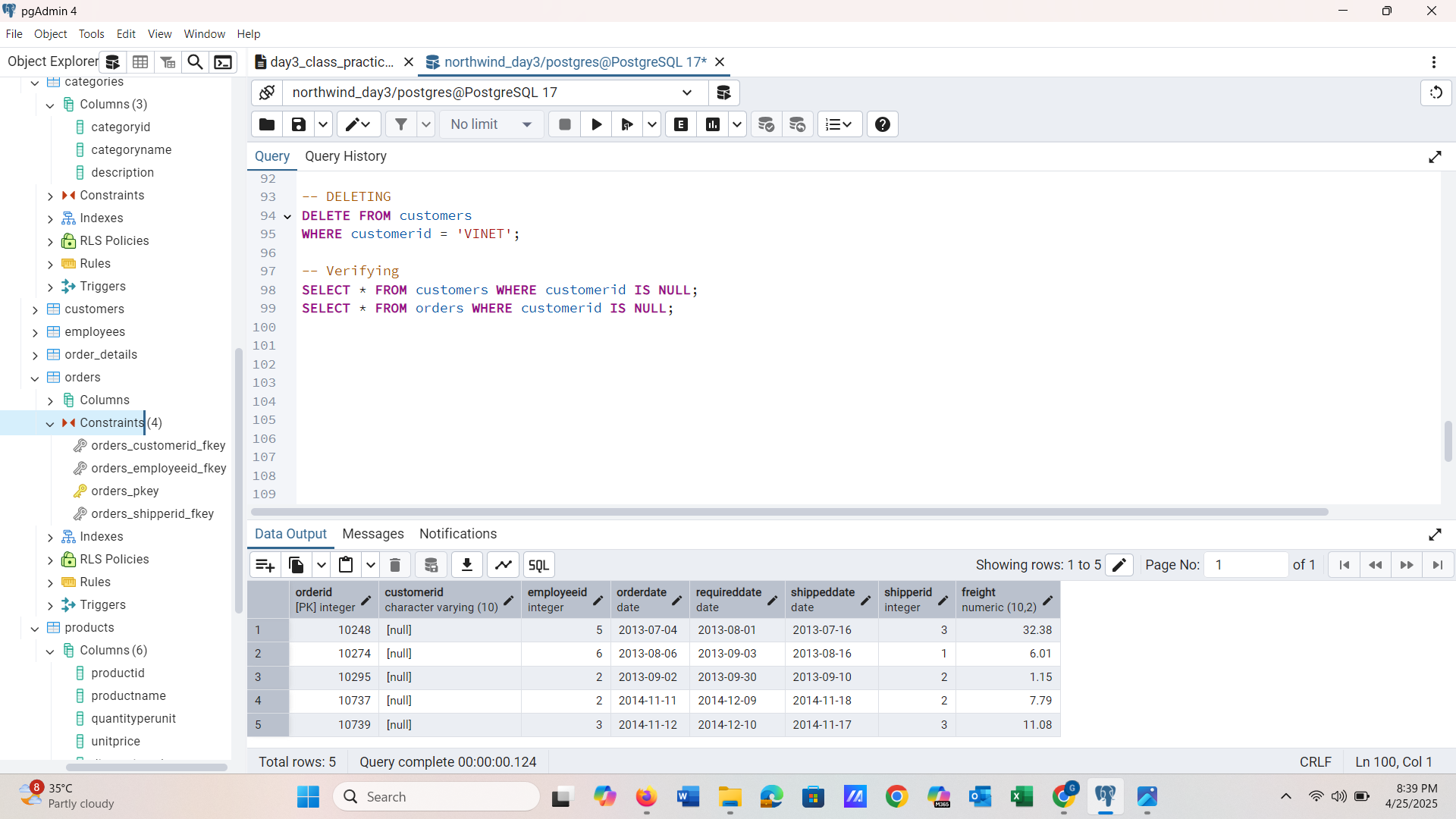
WHERE customerid = 'VINET';

**SCREENSHOTS**

SELECT \* FROM customers WHERE customerid IS NULL;



SELECT \* FROM orders WHERE customerid IS NULL;



**QUESTION 5)**

Insert the following data to Products using UPSERT:

**QUERY DETAILS**

***---Insert product\_id 100***

INSERT INTO products (productid, productname, quantityperunit, unitprice, discontinued, categoryid)

VALUES (100, 'Wheat bread', '1', 13, 0, 5)

ON CONFLICT (productid)

DO UPDATE SET

quantityperunit = EXCLUDED.quantityperunit;

***---Insert product\_id 101***

INSERT INTO products (productid, productname, quantityperunit, unitprice, discontinued, categoryid)

VALUES (101, 'White bread', '5 boxes', 13, 0, 5)

ON CONFLICT (productid)

DO UPDATE SET quantityperunit = EXCLUDED.quantityperunit;

***-- Update product\_id 100 again with new quantityperunit***

INSERT INTO products (productid, productname, quantityperunit, unitprice, discontinued, categoryid)

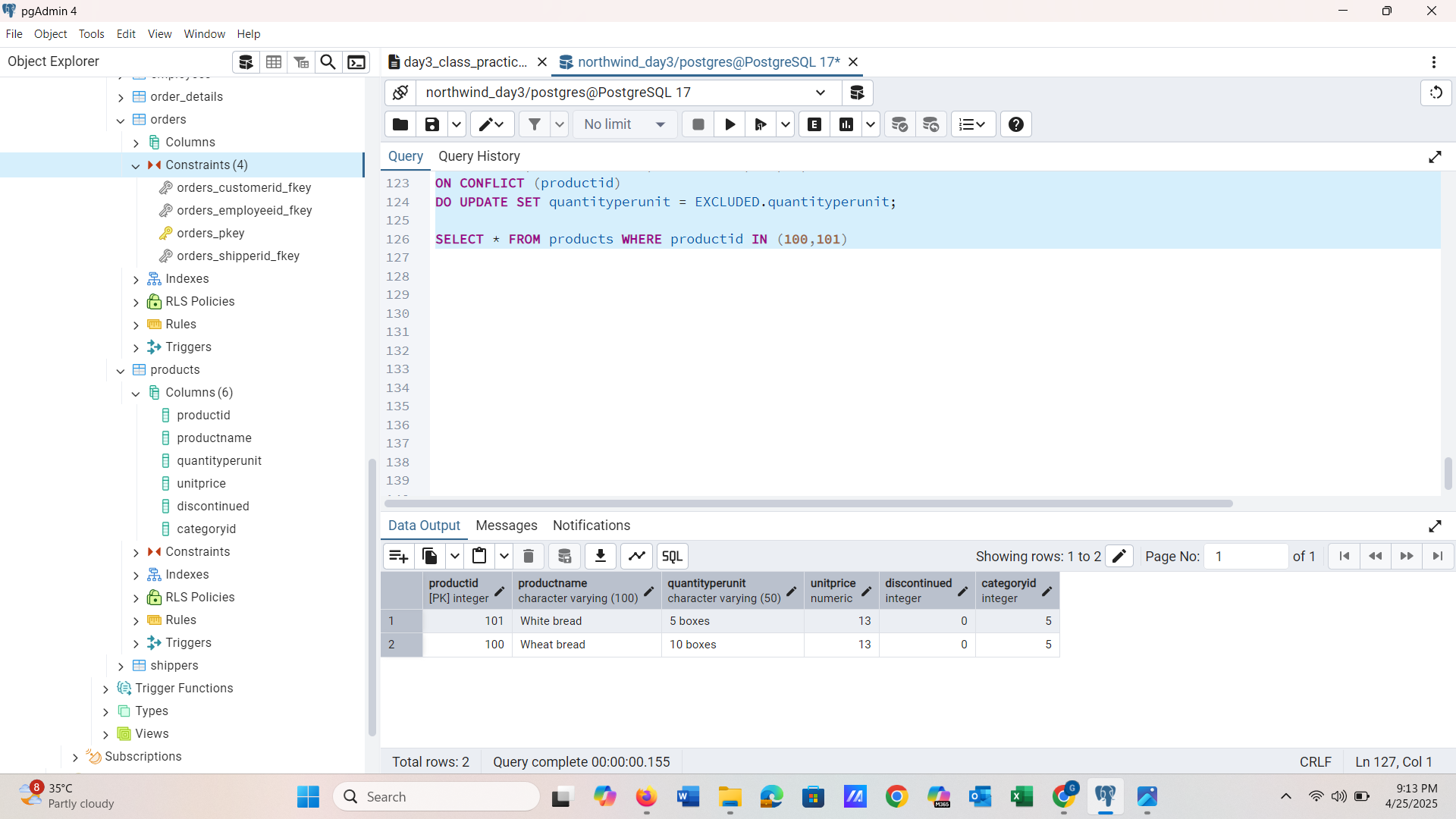
VALUES (100, 'Wheat bread', '10 boxes', 13, 0, 5)

ON CONFLICT (productid)

DO UPDATE SET quantityperunit = EXCLUDED.quantityperunit;

**SCREENSHOTS**

SELECT \* FROM products WHERE productid IN (100,101)



**QUESTION 6)**

Write a **MERGE query**:

Create **temp table with name:**  ‘updated\_products’ and insert values as below:

| productID | productName | quantityPerUnit | unitPrice | discontinued | categoryID |
| --- | --- | --- | --- | --- | --- |
| 100 | Wheat bread | 10 | 20 | 1 | 5 |
| 101 | White bread | 5 boxes | 19.99 | 0 | 5 |
| 102 | Midnight Mango Fizz | 24 - 12 oz bottles | 19 | 0 | 1 |
| 103 | Savory Fire Sauce | 12 - 550 ml bottles | 10 | 0 | 2 |

* Update the price and discontinued status for from below table ‘updated\_products’ only if there are matching products and updated\_products .discontinued =0
* If there are matching products and updated\_products .discontinued =1 then delete

* Insert any new products from updated\_products that don’t exist in products only if updated\_products .discontinued =0.

**QUERY DETAILS**

-- Temporary table

CREATE TEMP TABLE updated\_products (

productid INT PRIMARY KEY,

productName VARCHAR(100),

quantityPerUnit VARCHAR(50),

unitPrice NUMERIC,

discontinued INT,

categoryid INT

);

-- Inserting the values

INSERT INTO updated\_products (productid, productName, quantityPerUnit, unitPrice, discontinued, categoryid)

VALUES

(100, 'Wheat bread', '10', 20, 1, 5),

(101, 'White bread', '5 boxes', 19.99, 0, 5),

(102, 'Midnight Mango Fizz', '24 - 12 oz bottles', 19, 0, 1),

(103, 'Savory Fire Sauce', '12 - 550 ml bottles', 10, 0, 2);

/\*ALTERING TABLE\*/

ALTER TABLE products

DROP CONSTRAINT products\_categoryid\_fkey;

ALTER TABLE products

ADD CONSTRAINT products\_categoryid\_fkey

FOREIGN KEY (categoryID) REFERENCES categories(categoryID)

ON DELETE SET NULL;

/\*Inserting Missing Categories\*/

SELECT DISTINCT categoryID FROM updated\_products;

SELECT categoryID FROM categories;

INSERT INTO categories (categoryID, categoryName, description)

VALUES

(1, 'Beverages', 'Auto-added for foreign key'),

(2, 'Condiments', 'Auto-added for foreign key'),

(5, 'Bakery', 'Auto-added for foreign key')

ON CONFLICT (categoryID) DO NOTHING;

/\*MERGING\*/

MERGE INTO products AS p

USING updated\_products AS u

ON p.productID = u.productID

-- 1. Update if discontinued = 0

WHEN MATCHED AND u.discontinued = 0 THEN

UPDATE SET

unitPrice = u.unitPrice,

discontinued = u.discontinued

-- 2. Delete if discontinued = 1

WHEN MATCHED AND u.discontinued = 1 THEN

DELETE

-- 3. Insert if not exists and discontinued = 0

WHEN NOT MATCHED AND u.discontinued = 0 THEN

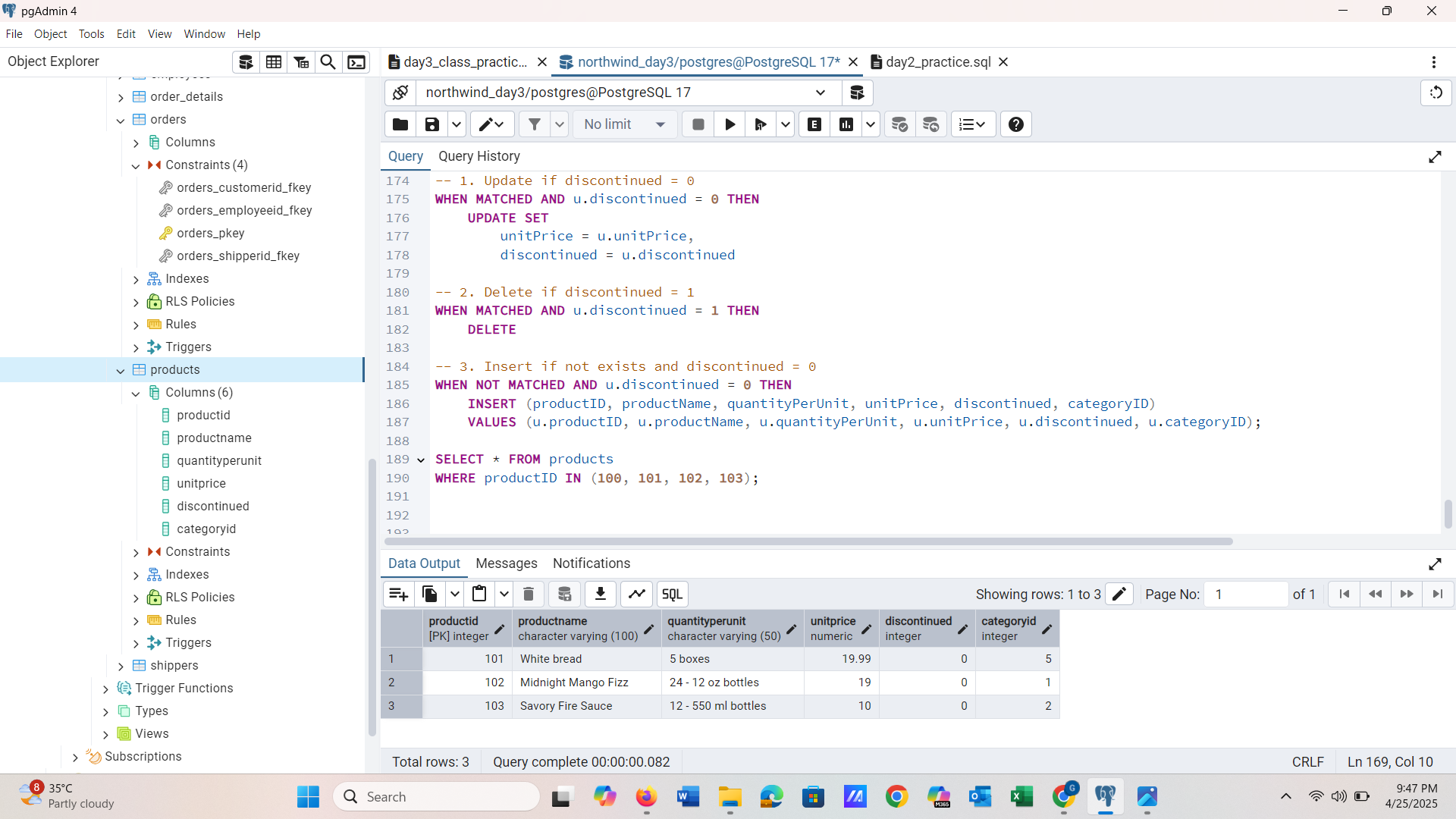
INSERT (productID, productName, quantityPerUnit, unitPrice, discontinued, categoryID)

VALUES (u.productID, u.productName, u.quantityPerUnit, u.unitPrice, u.discontinued, u.categoryID);

**SCREENSHOT**

SELECT \* FROM products

WHERE productID IN (100, 101, 102, 103);

****

**QUESTION 7) List all orders with employee full names. (Inner join)**

**NEW DATABASE**

**QUERY DETAILS**

SELECT

orders.order\_id,

orders.order\_date,

employees.first\_name || ' ' || employees.last\_name AS employee\_full\_name

FROM

orders

INNER JOIN

employees ON orders.employee\_id = employees.employee\_id;

**SCREENSHOT**

