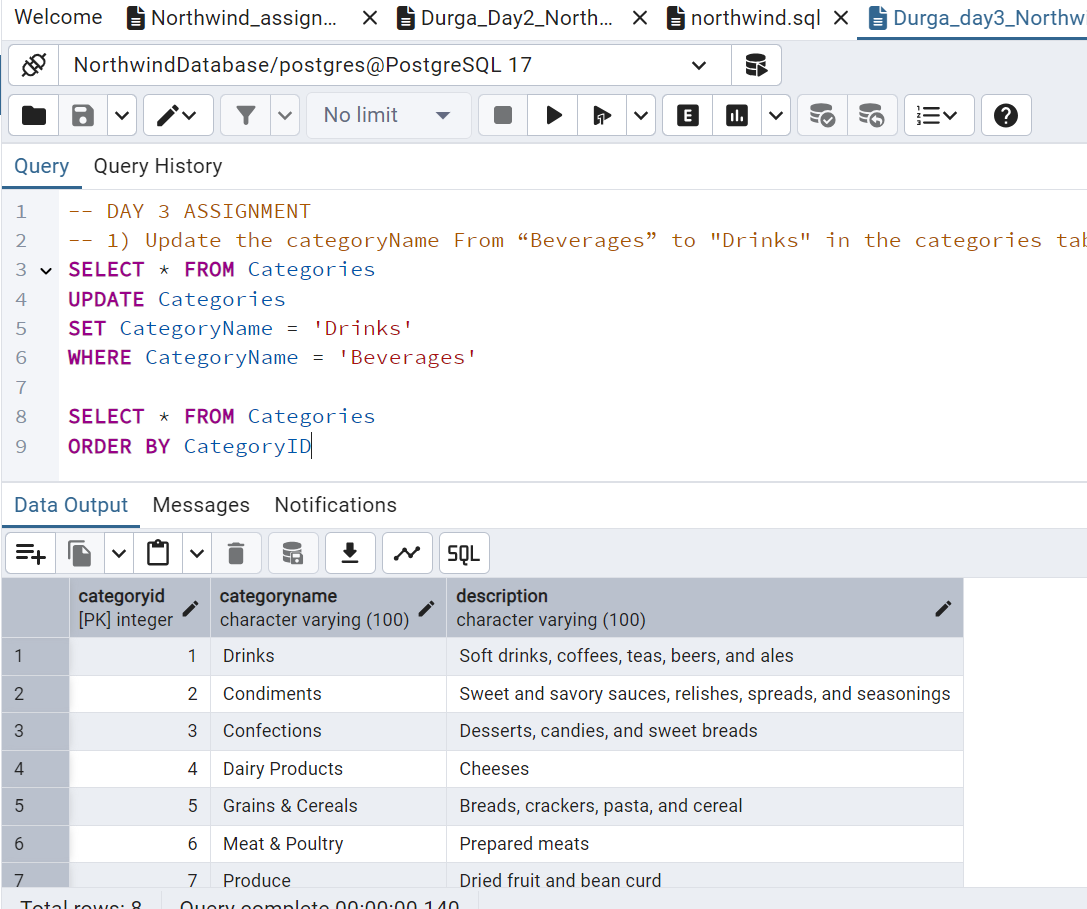
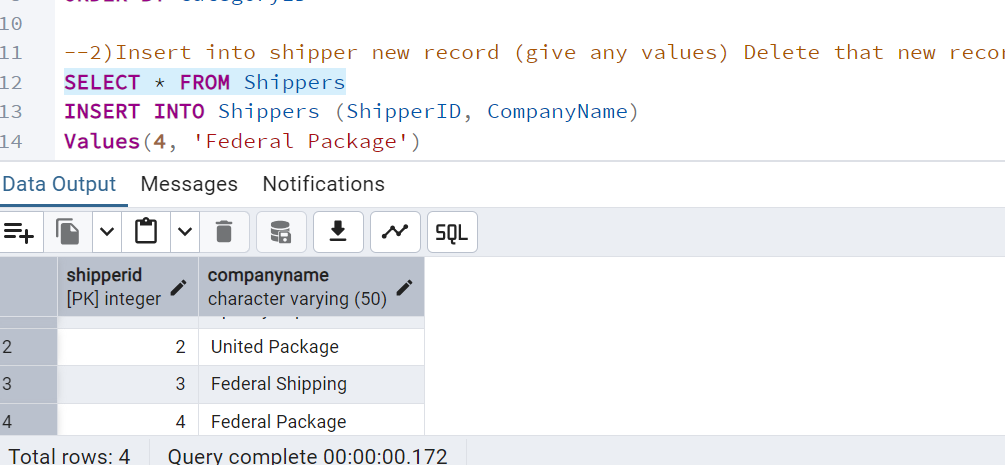
**Day 3**

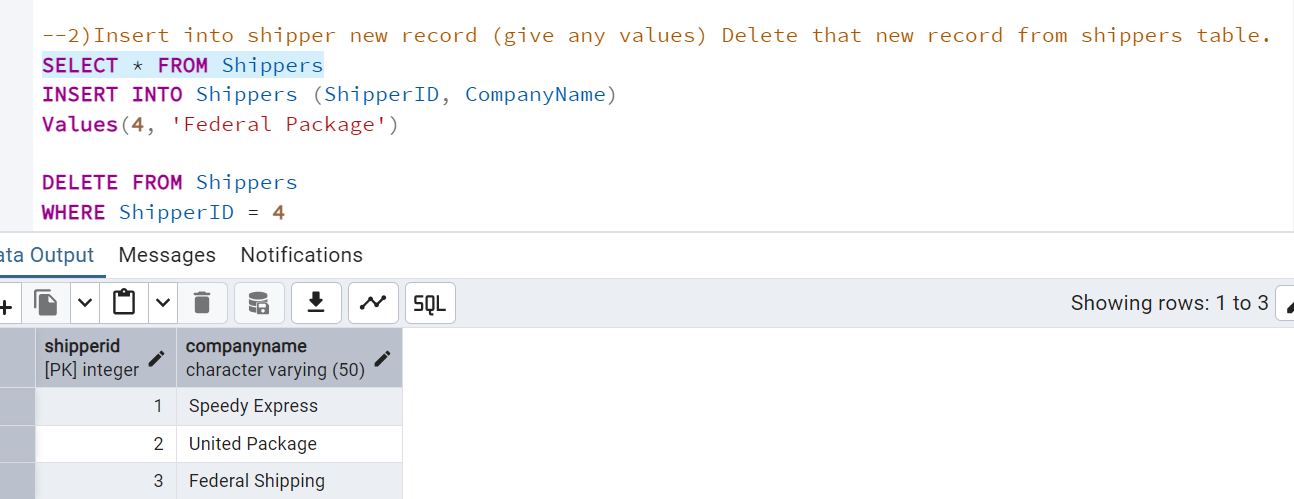
**USE Northwind from Kaggle:**

1)      Update the CategoryName From “Beverages” to "Drinks" in the categories table.



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



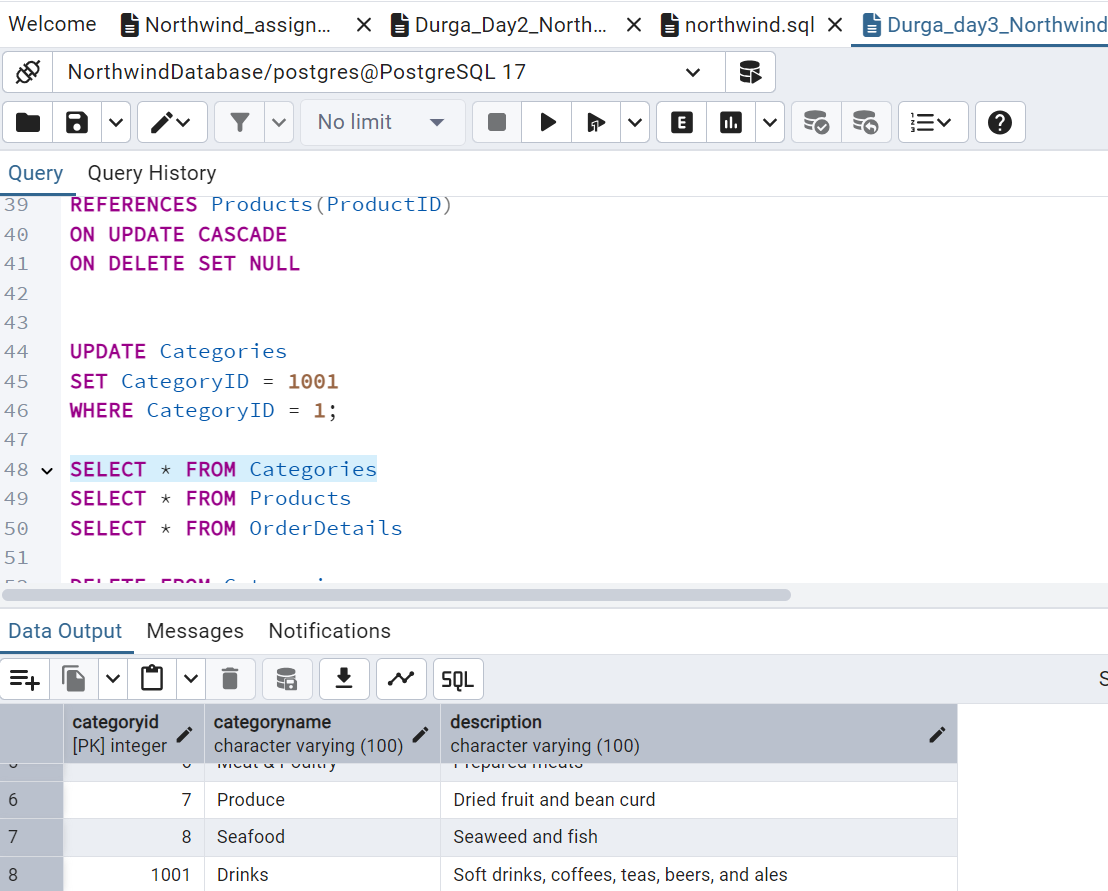


3)      Update categoryID=1 to categoryID=1001. Make sure related products update their categoryID too. Display the both category and products table to show the cascade.

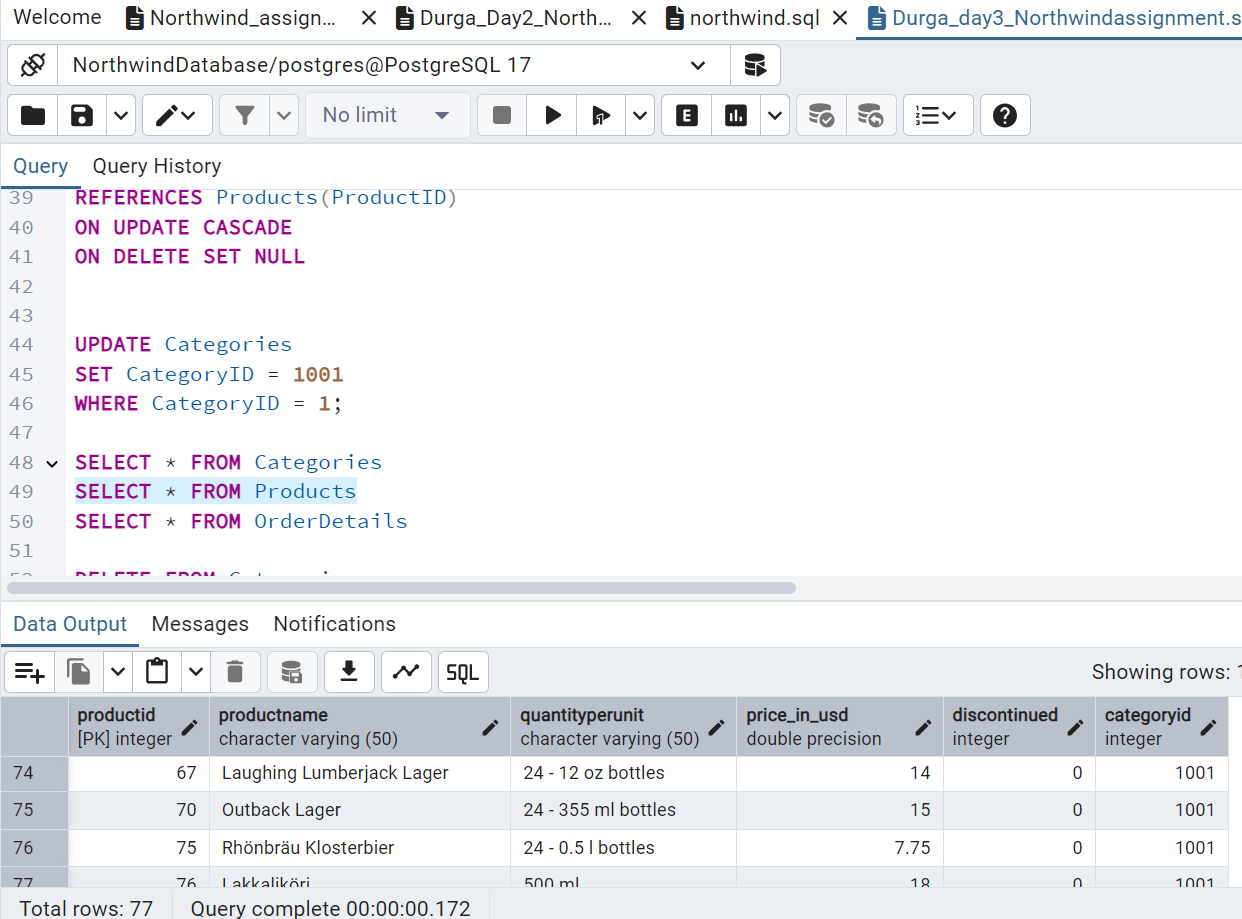
 Delete the categoryID= “3”  from categories. Verify that the corresponding records are deleted automatically from products.

 (HINT: Alter the foreign key on products(categoryID) to add ON UPDATE CASCADE, ON DELETE CASCADE)

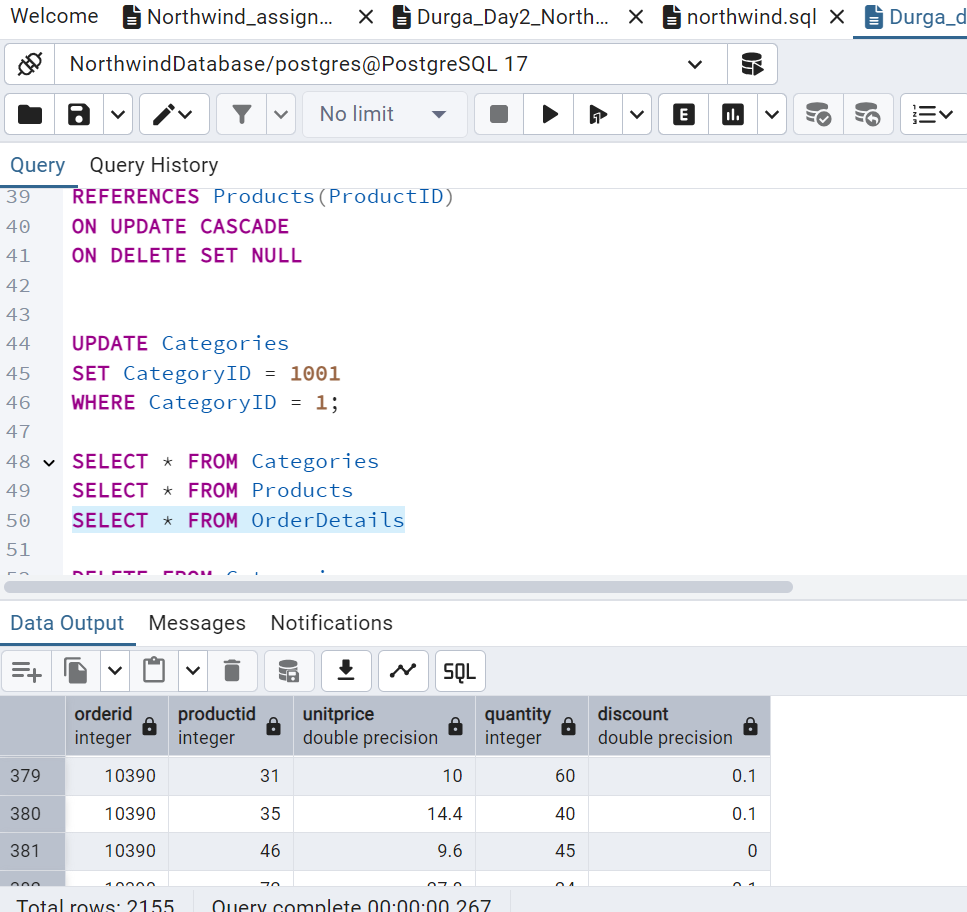
Below is the Category table where categoryid value 1 is changed to 1001



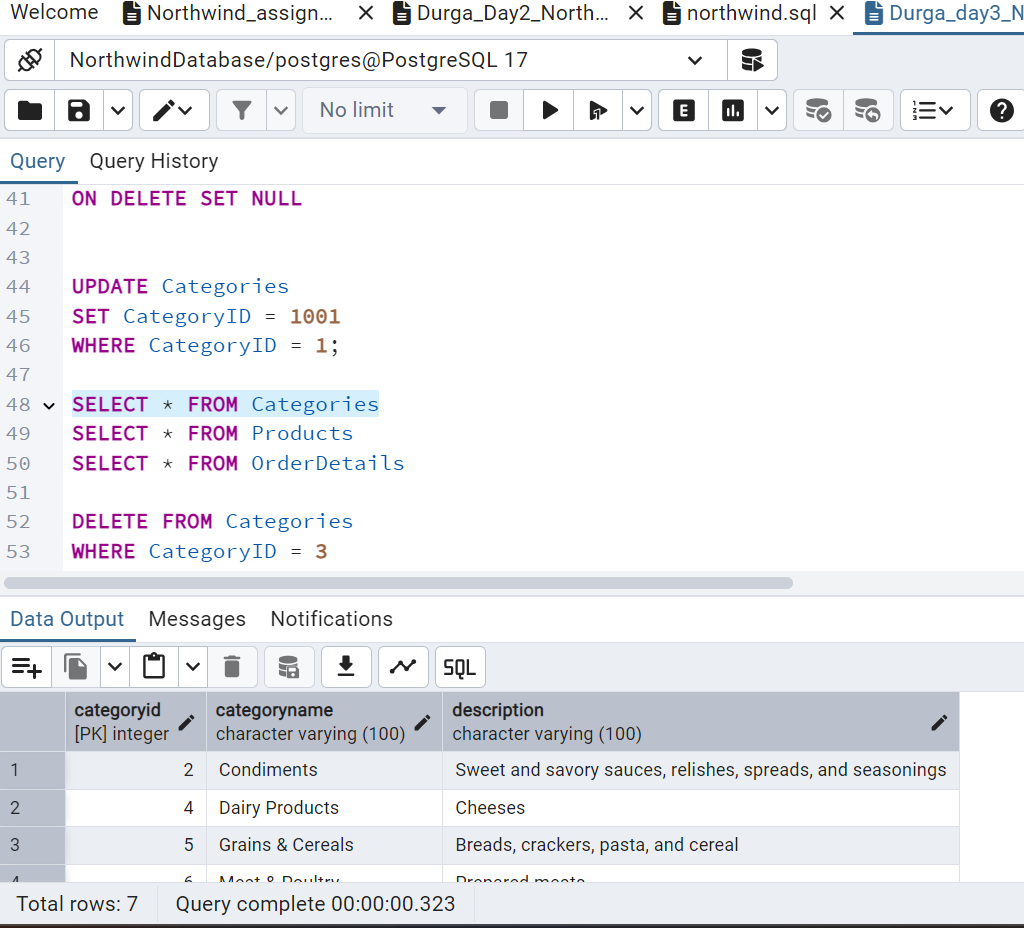
Below is the products table where categoryid value is updated from 1 to 1001



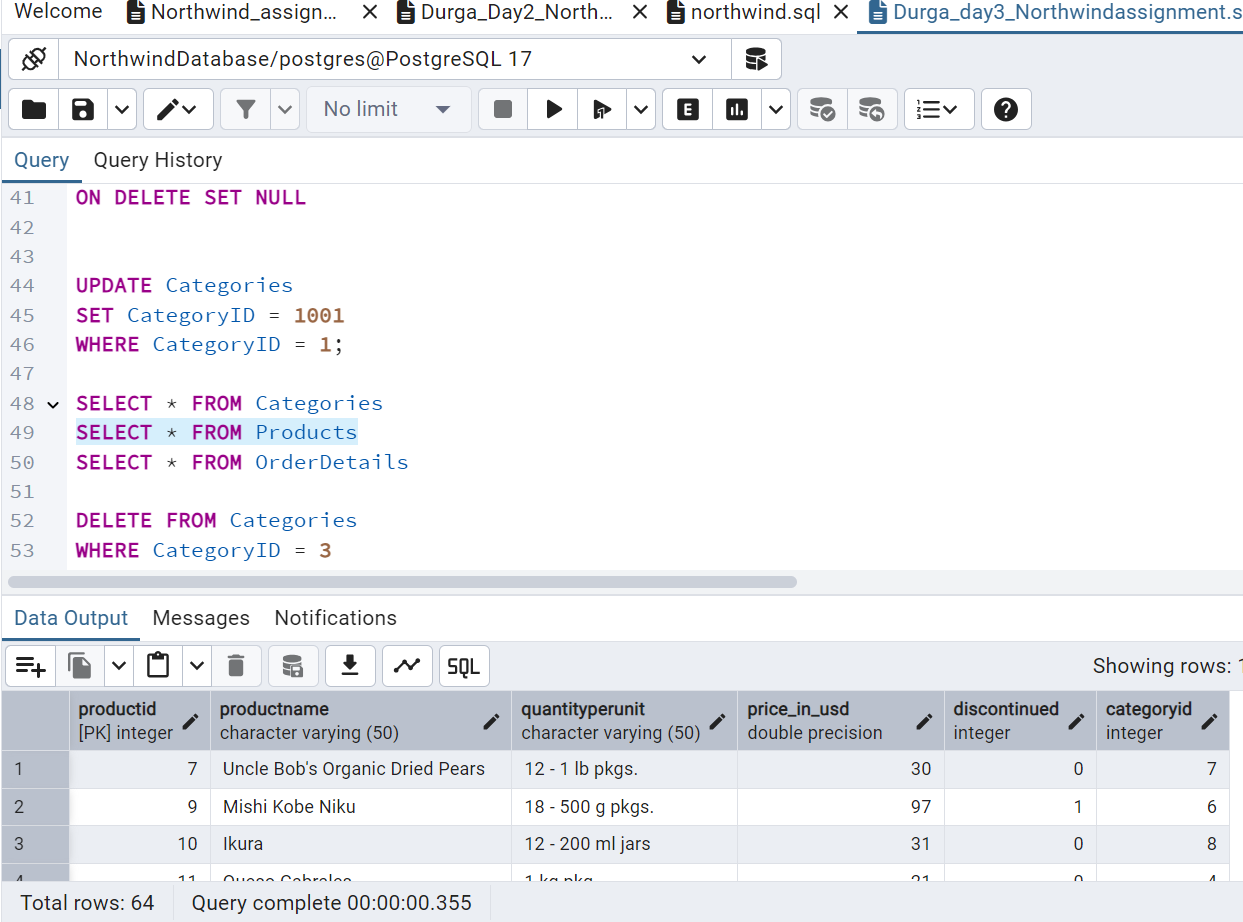
Below is the Orderdetails table updating categoryid value doesn’t affect this table as there is no direct connection.



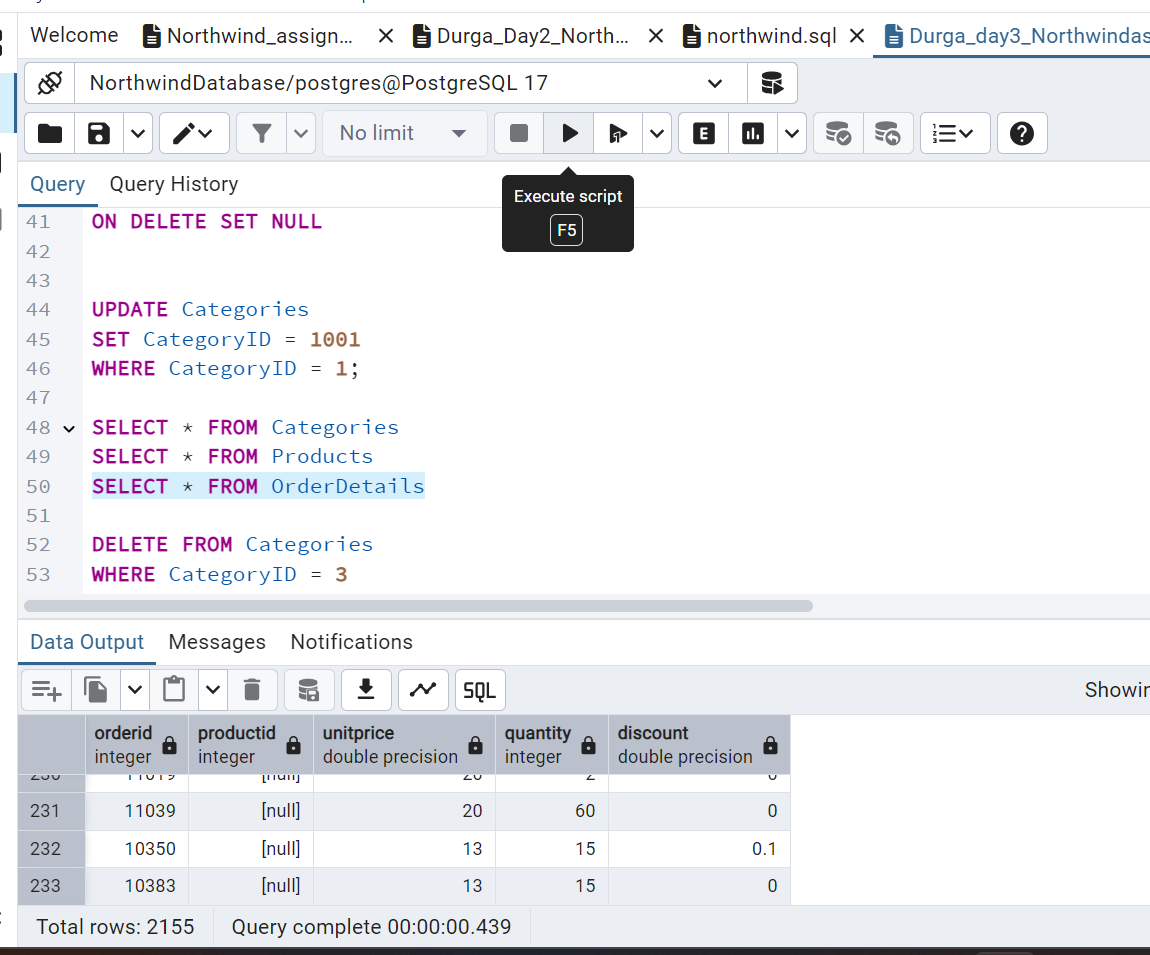
After Deleting CategoryID=3 in Categories table the categoryid value is deleted.



Below is the products table after deleting categoryid=3. Since we have set “ON DELETE CASCADE” for products table when deleting categoryid = 3 corresponding row values are deleted in products table. If you see below total rows reduced to 64 (before deletion it was 77 can see the screenshot of Updating value) after deletion.

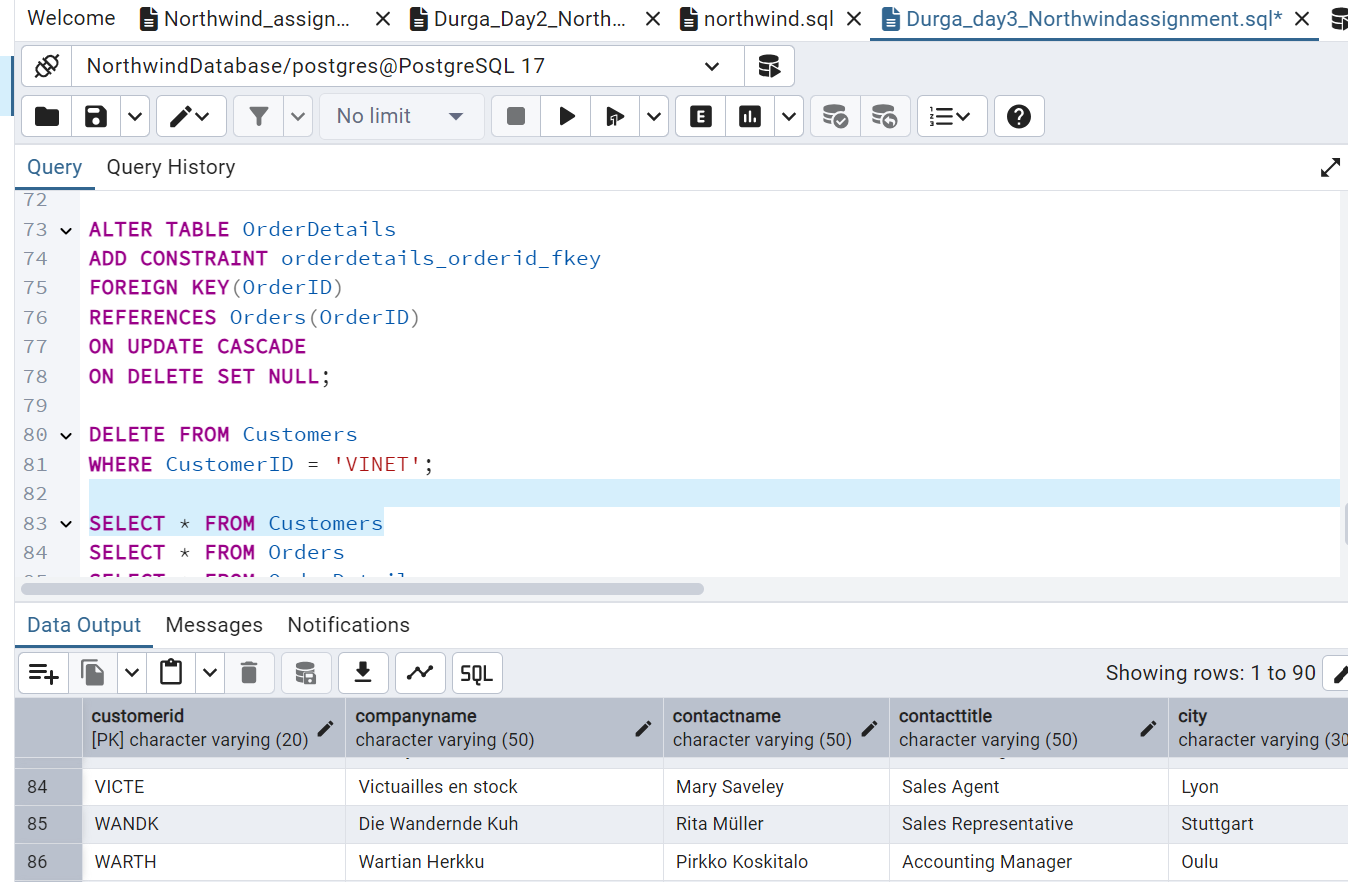


Below is Orderdetails table where the (deleted caategoryid = 3) corresponding productid values are set null. This is because we set “ON DELETE SET NULL”

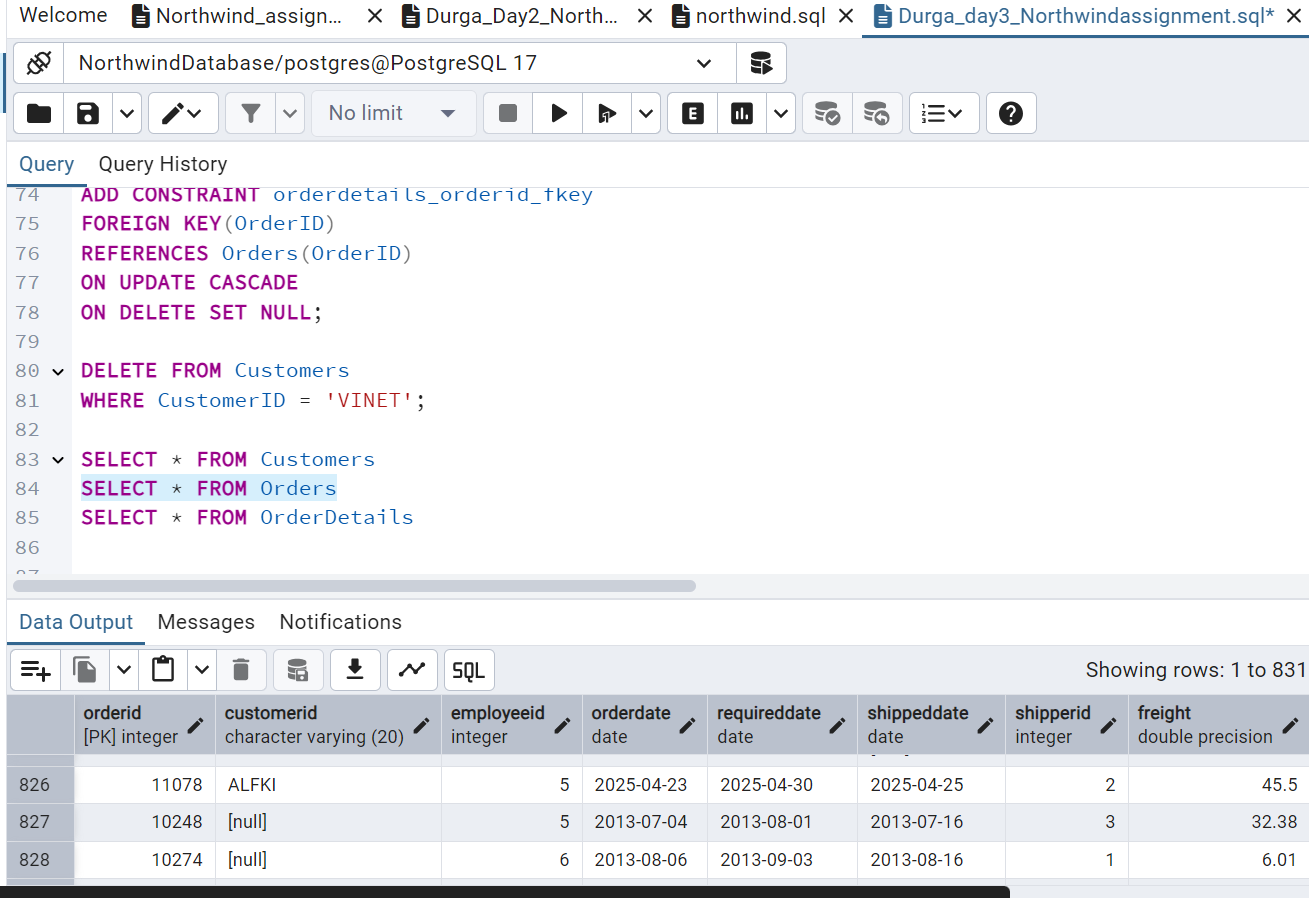


4)      Delete the customer = “VINET”  from customers. Corresponding customers in orders table should be set to null (HINT: Alter the foreign key on orders(customerID) to use ON DELETE SET NULL)

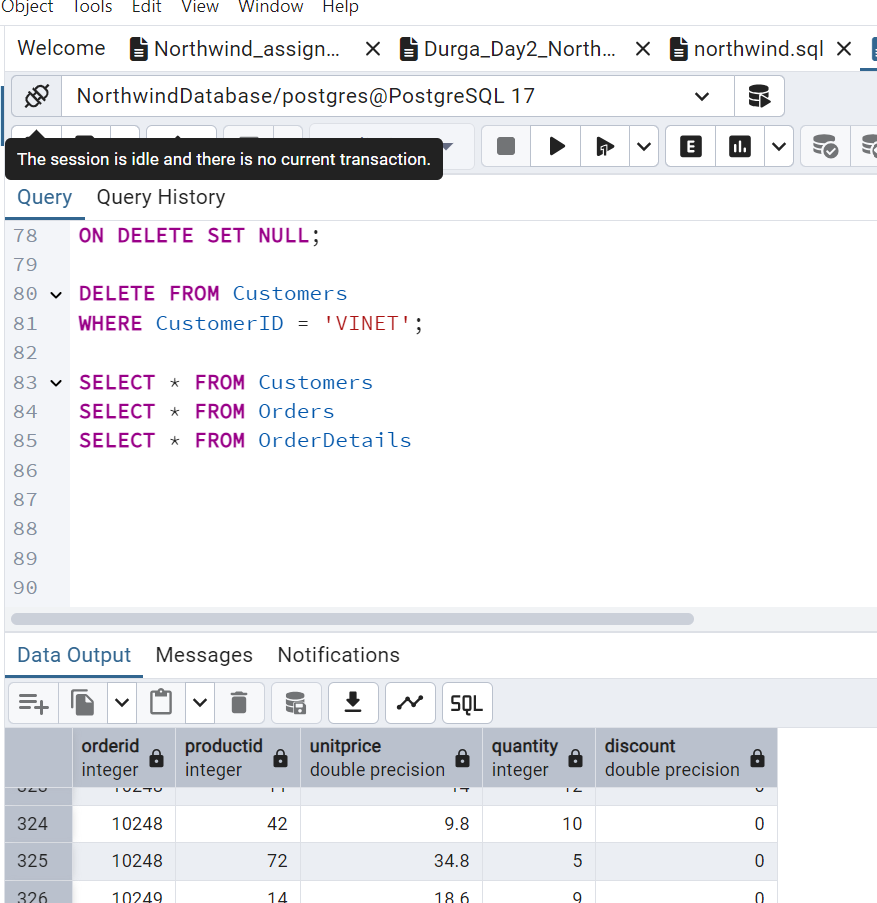
 After deleting CustomerID = ‘VINET’ below is the change in output in Customers table CustomerID VINET is deleted



After deleting CustomerID = ‘VINET’ below is the change in output in Orders table setting “ON DELETE SET NULL” corresponding customerid values in orders table is set NULL

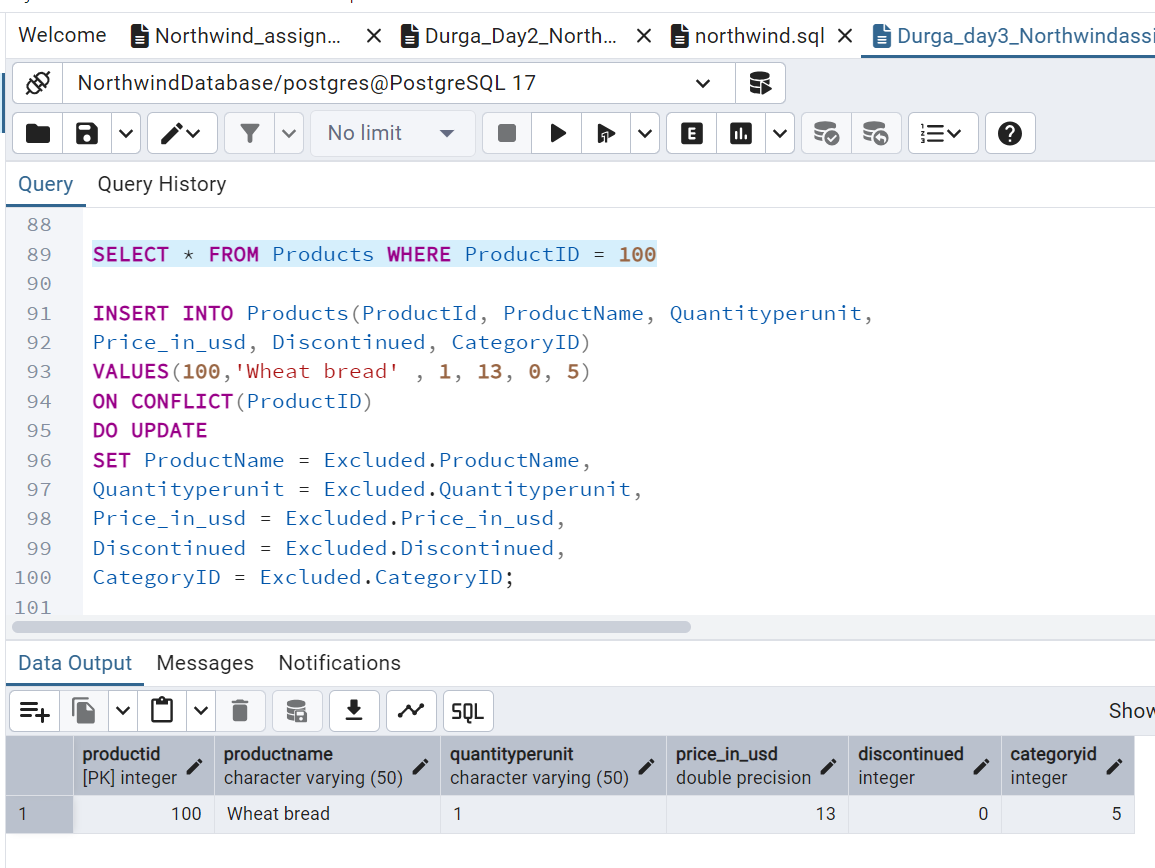


After deleting CustomerID = ‘VINET’ there won’t be any change in output in OrdersDetails table unless we delete OrderID or if we had set “ON DELETE CASCADE” in Orders table.

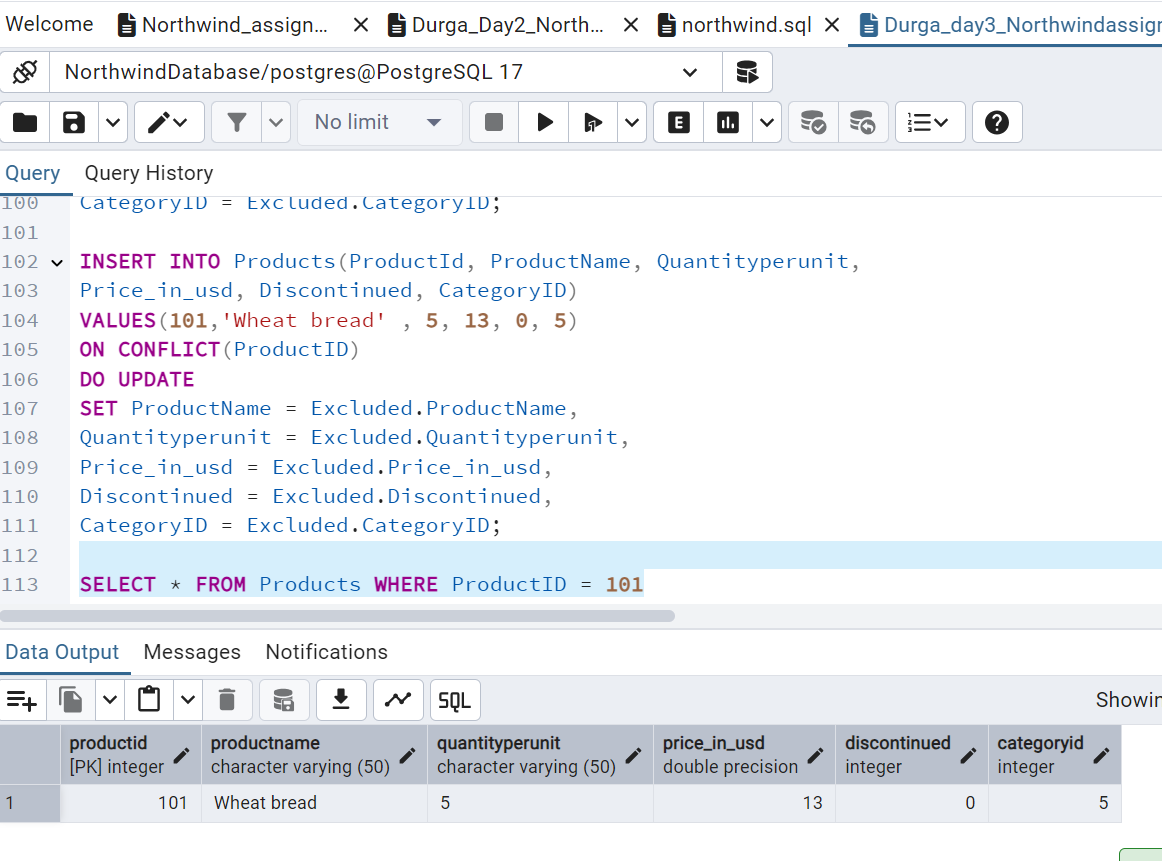


5)      Insert the following data to Products using UPSERT:

product\_id = 100, product\_name = Wheat bread, quantityperunit=1,unitprice = 13, discontinued = 0, categoryID=5

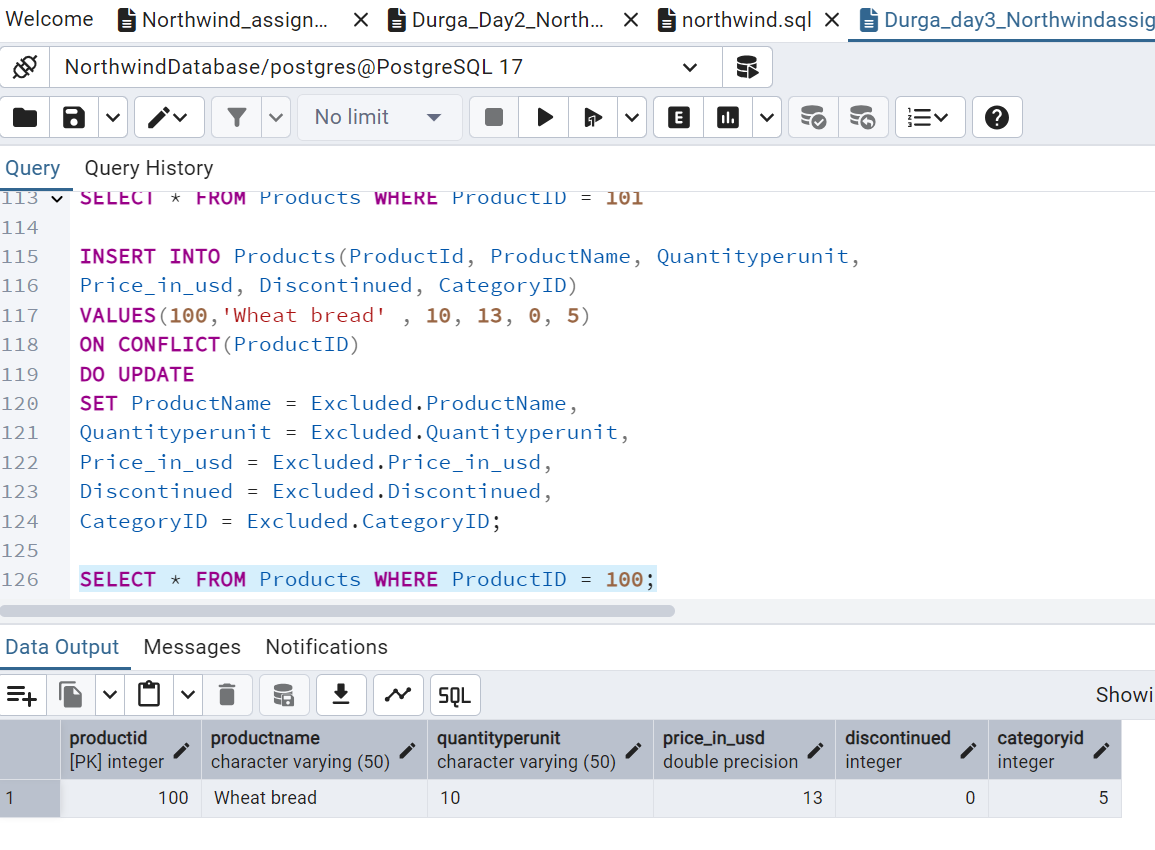


product\_id = 101, product\_name = White bread, quantityperunit=5 boxes,unitprice = 13, discontinued = 0, categoryID=5



product\_id = 100, product\_name = Wheat bread, quantityperunit=10 boxes,unitprice = 13, discontinued = 0, categoryID=5

(this should update the quantityperunit for product\_id = 100)



6)      Write a **MERGE query**:

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

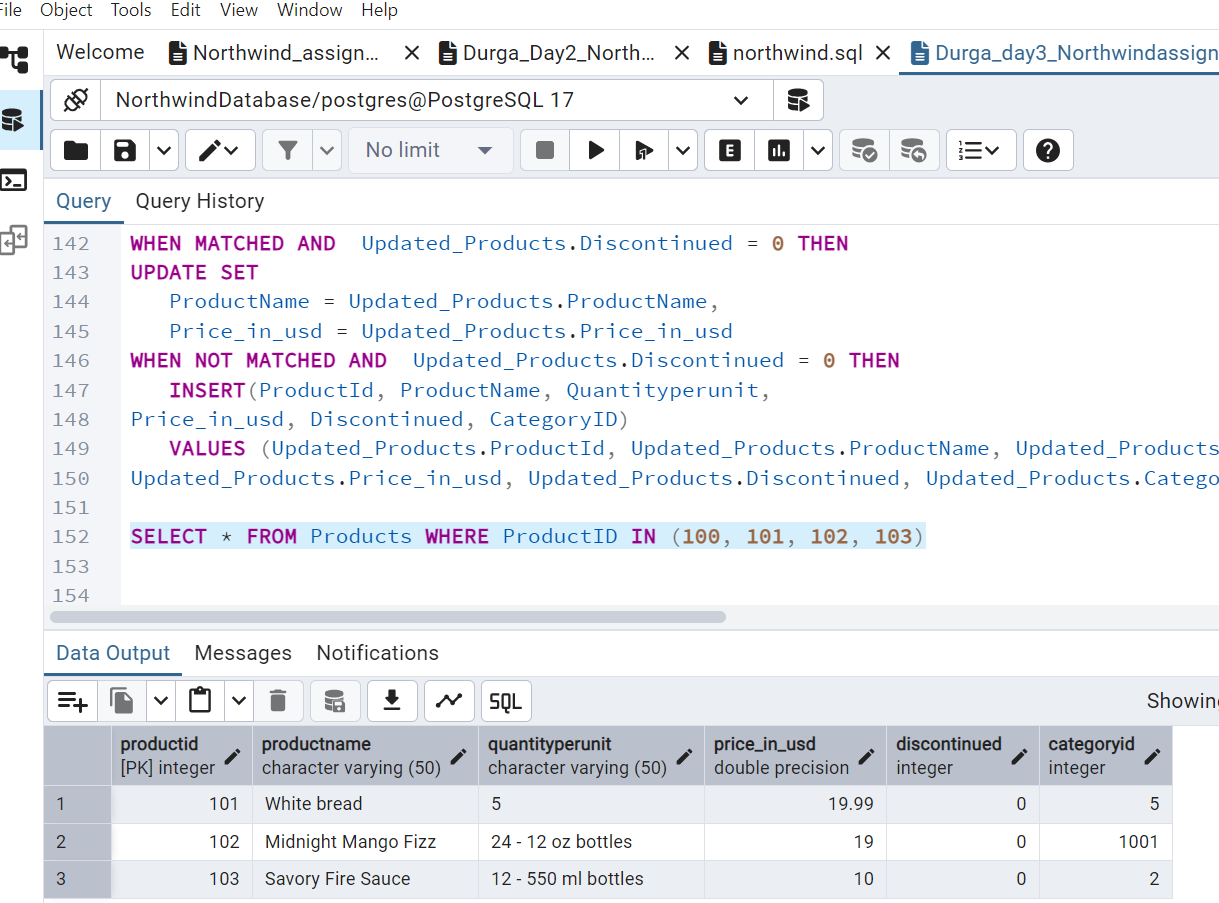
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| productID | productName | quantityPerUnit | unitPrice | discontinued | categoryID |
| 100 | Wheat bread | 10 | 20 | 1 | 5 |
| 101 | White bread | 5 boxes | 19.99 | 0 | 5 |
| 102 | Midnight Mango Fizz | 24 - 12 oz bottles | 19 | 0 | 1 |
| 103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

In the previous query we inserted productid 100 and 101. So productid has values 100 and 101 already. Hence, After executing the query the productid with value=100 will be deleted and productid =101 row value will be updated to the new value and two new productid (102, 103)row values will be inserted in Products table



**USE NEW Northwind DB:**

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

