WEGMANS DEVAM PATEL

12/13/2018

Wegmans Database Problem

There was a problem in Wegmans database to save the Post Transaction for customers and show on hand quantity.

SUBSET 1

Name: Devam Patel

Company: Wegmans

Location (Headquarter): Rochester, NY

Type of store: Grocery

Website: https://www.wegmans.com

Information About the Company:

Wegmans is a regional supermarket chain with 97 stores: 46 in New York, 17 in Pennsylvania, 9 in New Jersey, 11 in Virginia, 8 in Maryland, and 6 in Massachusetts. It is one of the largest private companies in the U.S. The company has

- 48,000 employees
- Annual sales in 2017 of $8.7 billion
- 31st on the 2017 *Supermarket News* list of the Top 75 Supermarkets based on sales volume

Wegmans is a family-owned company, founded in 1916, headquartered in Rochester, NY. Danny Wegman is chairman; Colleen Wegman, his daughter, is president and CEO. Danny's daughter Nicole Wegman is sr. vice president. Robert Wegman, Danny's father, was chairman until his death in April 2006.

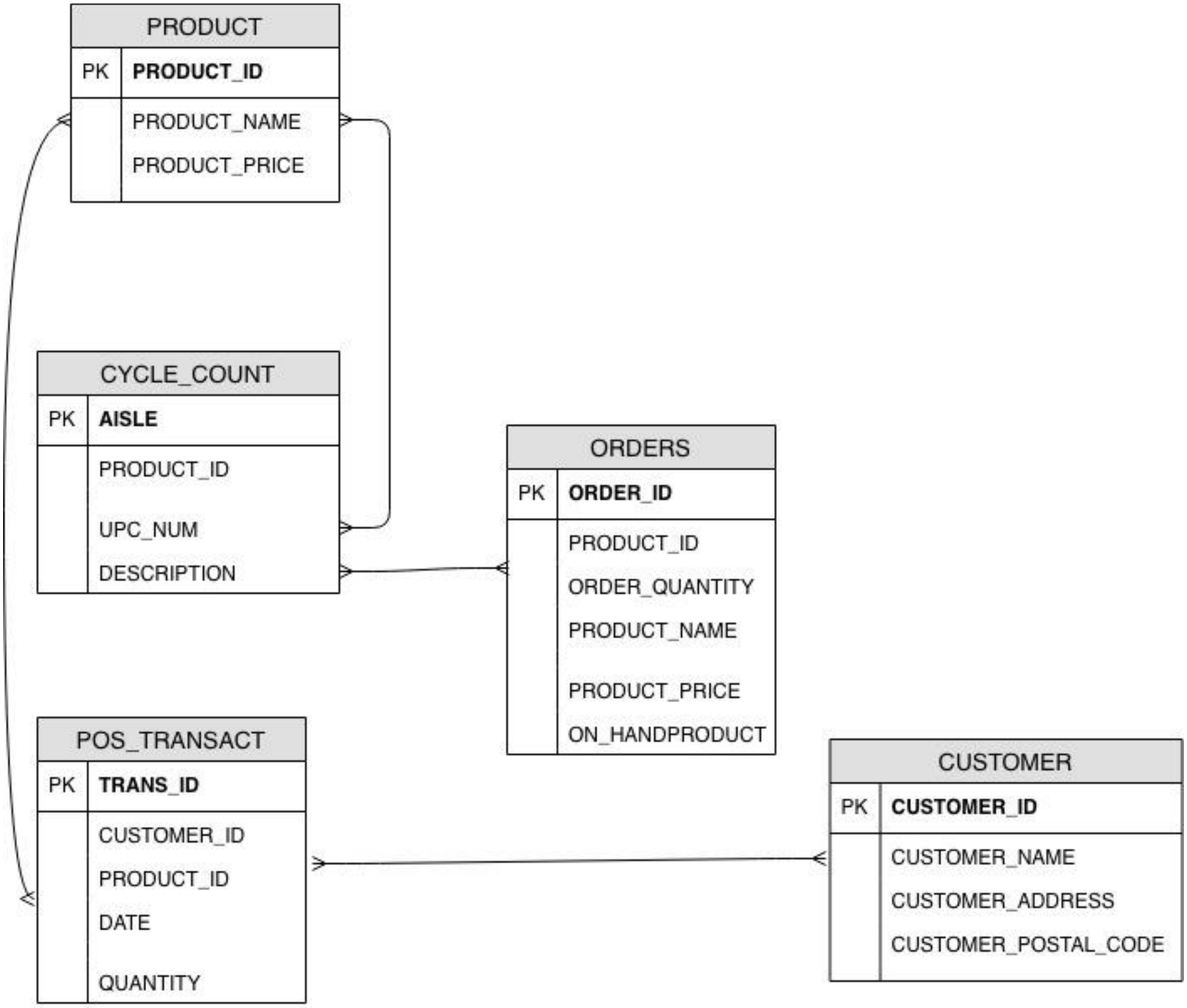
Wegmans Founders

Brothers Walter and John Wegman (Robert Wegman's father and uncle) learned the food business by working in their parents' store in Rochester. John peddled fresh produce from a pushcart, and in 1916, he opened the Rochester Fruit & Vegetable Company, which marks the beginning of Wegmans Food Markets. Walter joined him a year later. Robert Wegman assumed leadership of the company in 1950, guiding it until his death in 2006.

Database Problem

The company database problem is storing the selling item. One employee from each department must count the product every day and see how many they sell on that day, so they can adjust the cycle count on daily basis and employee has an idea about how many to order. That's how they don't get product on daily basis. This problem cause Wegmans and customer. Wegmans gets affect by ordering more, if they order more that pay more and end up with not selling that product, so they must throw out the products that cause the company by money. Customer get effects by dissatisfied when stock runs out of stock. I want to solve this problem by getting the product name and how many they sell by end of the day. If they have to order 10 boxes of butter for Saturday and already, they have around 3 boxes of butter in backstock and 2 boxes of butter they sold on Friday and 2 boxes of butter on shelf, 5 boxes of butter on hands manager of the department need to order just 5 more to have enough butter for customers. I'm thinking to solve this problem by making a new raw for the each product in database, so when it's time to give order for the next day then manager can look that up on the data table and will be able to figure out the amounts of products need for the next day, also I want to make a new row that displays the amount of product in hands by that way employee doesn't have to do cycle count every day. When employee is giving order, they can just look at the table and have an idea of what amount of product to order.

SUBSET 2



PRODUCT

PK	**PRODUCT_ID**
	PRODUCT_NAME PRODUCT_PRICE

CYCLE_COUNT

PK	**AISLE**
	PRODUCT_ID UPC_NUM DESCRIPTION

ORDERS

PK	**ORDER_ID**
	PRODUCT_ID ORDER_QUANTITY PRODUCT_NAME PRODUCT_PRICE ON_HANDPRODUCT

POS_TRANSACT

PK	**TRANS_ID**
	CUSTOMER_ID PRODUCT_ID DATE QUANTITY

CUSTOMER

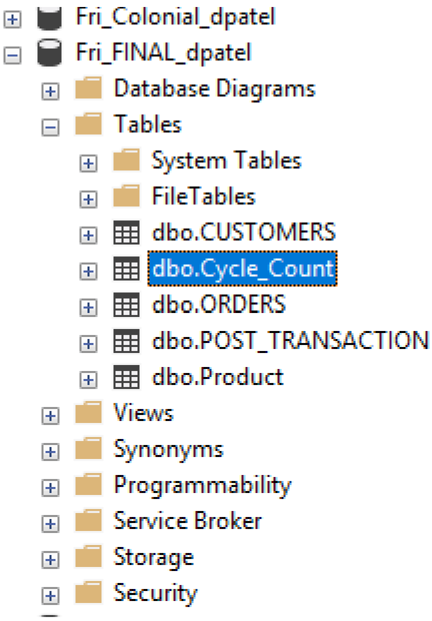
PK	**CUSTOMER_ID**
	CUSTOMER_NAME CUSTOMER_ADDRESS CUSTOMER_POSTAL_CODE

BUSINESS RULE

- ORDERS NEEDS TO BE DONE IN THE CASES, NOT THE QUANTITY.
- DO THE INVENTORY EVERY WEEK
- LOOK FOR THE HOLIDAYS BEFORE ORDERING.

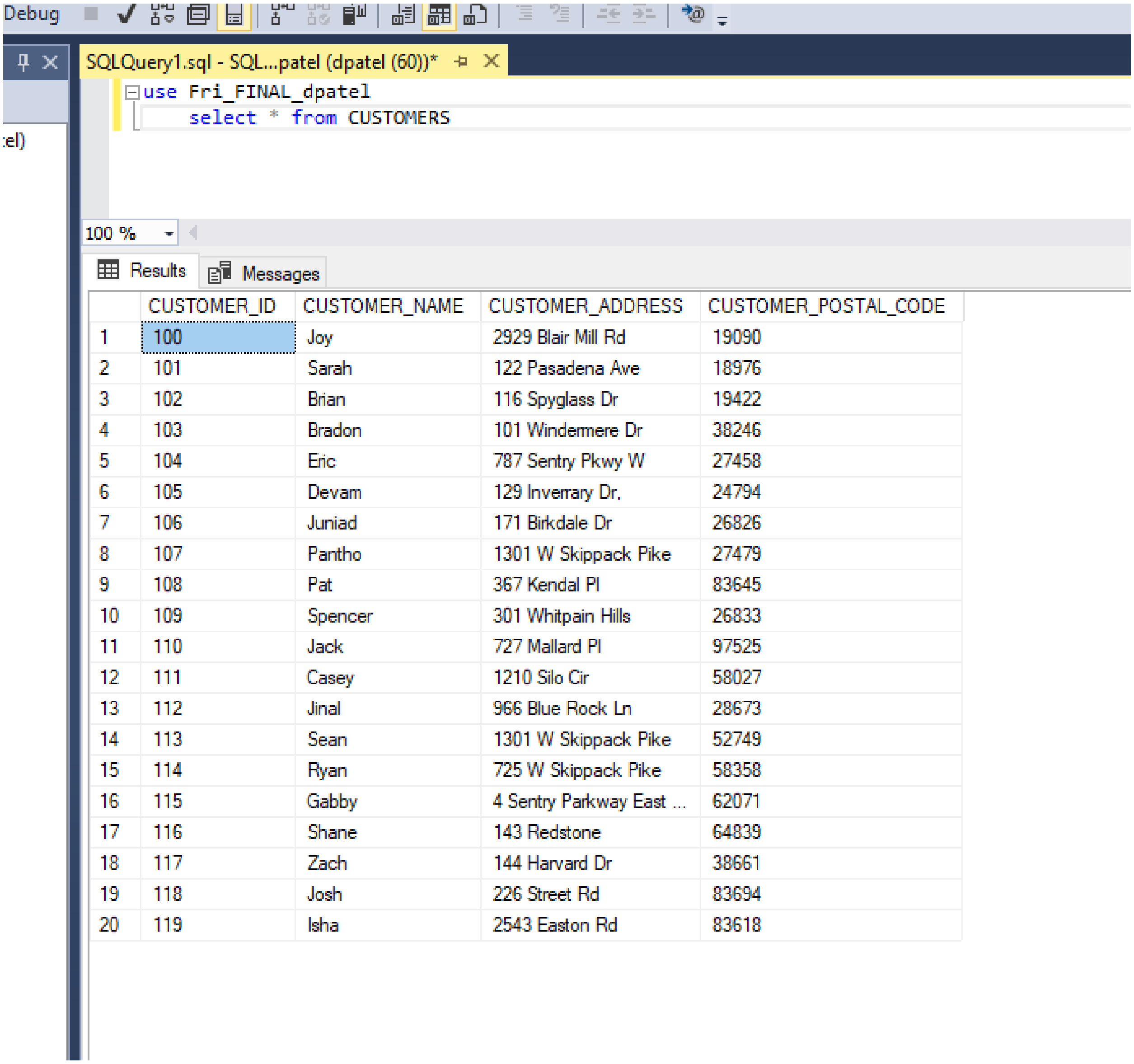
SUBSET 3

5 Tables



- Fri_Colonial_dpatel
- Fri_FINAL_dpatel
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - dbo.CUSTOMERS
    - dbo.Cycle_Count
    - dbo.ORDERS
    - dbo.POST_TRANSACTION
    - dbo.Product
  - Views
  - Synonyms
  - Programmability
  - Service Broker
  - Storage
  - Security

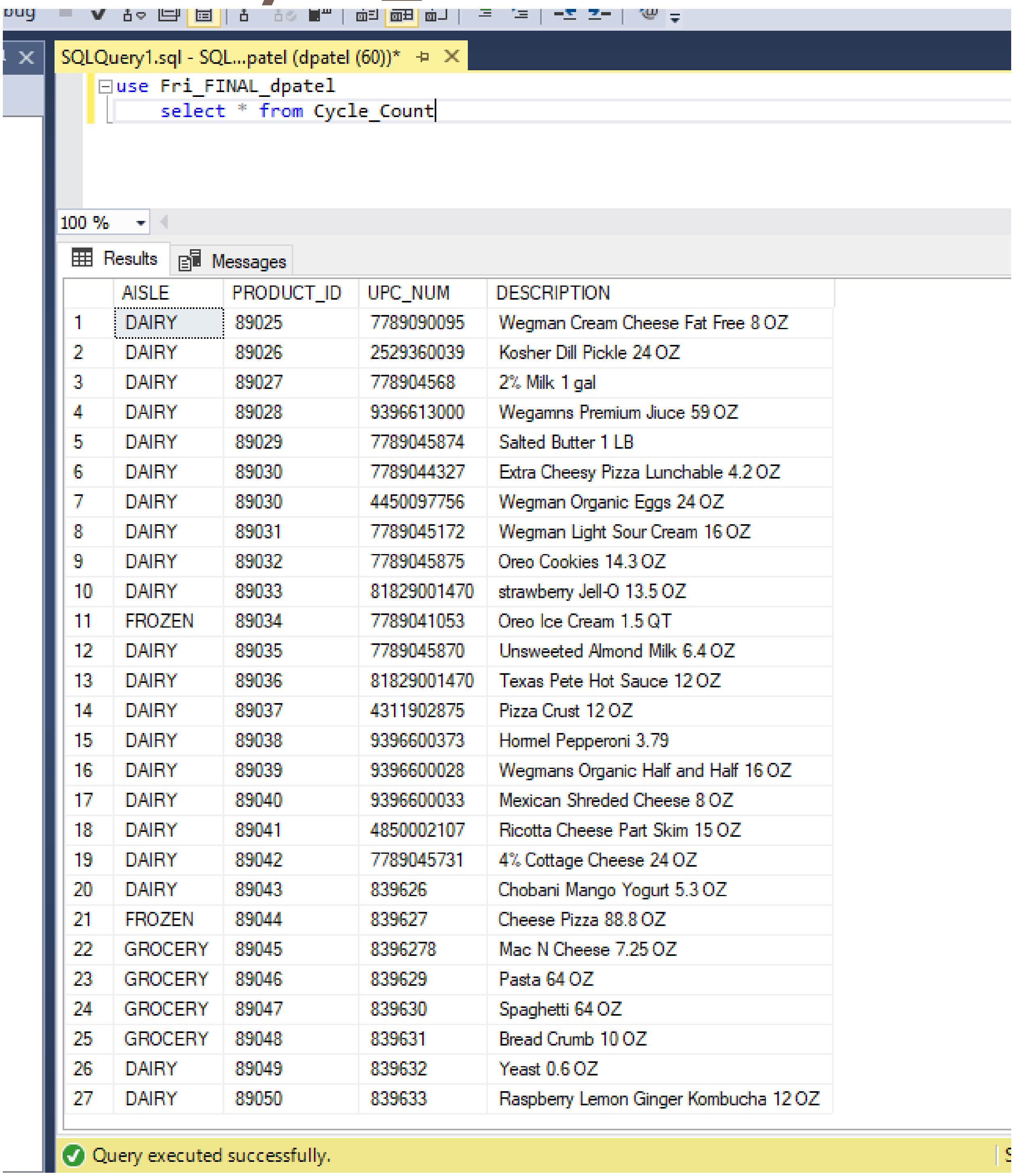
Customers Table



```
use Fri_FINAL_dpatel
    select * from CUSTOMERS
```

	CUSTOMER_ID	CUSTOMER_NAME	CUSTOMER_ADDRESS	CUSTOMER_POSTAL_CODE
1	100	Joy	2929 Blair Mill Rd	19090
2	101	Sarah	122 Pasadena Ave	18976
3	102	Brian	116 Spyglass Dr	19422
4	103	Bradon	101 Windermere Dr	38246
5	104	Eric	787 Sentry Pkwy W	27458
6	105	Devam	129 Inverrary Dr,	24794
7	106	Juniad	171 Birkdale Dr	26826
8	107	Pantho	1301 W Skippack Pike	27479
9	108	Pat	367 Kendal Pl	83645
10	109	Spencer	301 Whitpain Hills	26833
11	110	Jack	727 Mallard Pl	97525
12	111	Casey	1210 Silo Cir	58027
13	112	Jinal	966 Blue Rock Ln	28673
14	113	Sean	1301 W Skippack Pike	52749
15	114	Ryan	725 W Skippack Pike	58358
16	115	Gabby	4 Sentry Parkway East ...	62071
17	116	Shane	143 Redstone	64839
18	117	Zach	144 Harvard Dr	38661
19	118	Josh	226 Street Rd	83694
20	119	Isha	2543 Easton Rd	83618

Cycle_Count Table

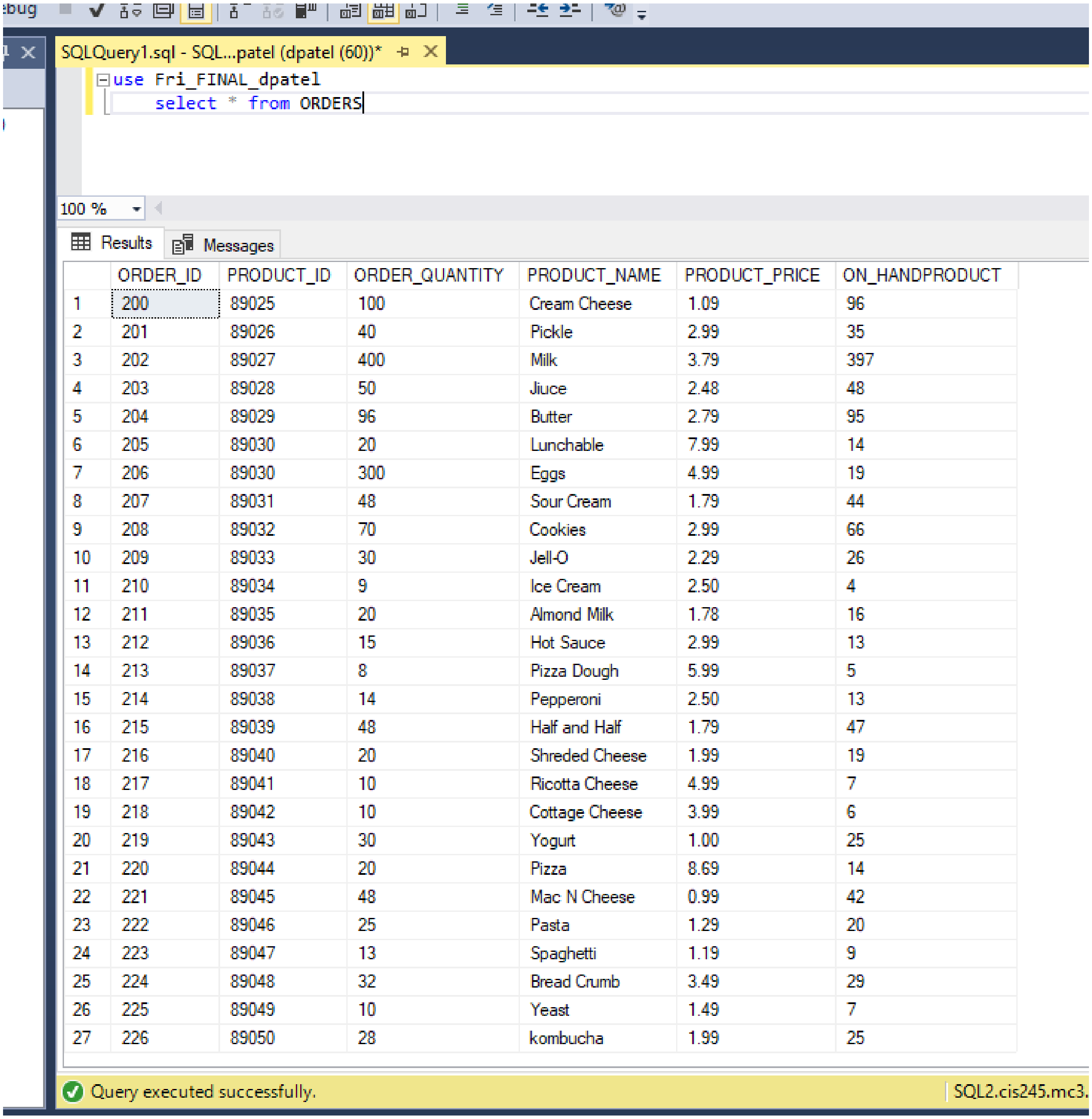


```
use Fri_FINAL_dpatel
    select * from Cycle_Count
```

	AISLE	PRODUCT_ID	UPC_NUM	DESCRIPTION
1	DAIRY	89025	7789090095	Wegman Cream Cheese Fat Free 8 OZ
2	DAIRY	89026	2529360039	Kosher Dill Pickle 24 OZ
3	DAIRY	89027	778904568	2% Milk 1 gal
4	DAIRY	89028	9396613000	Wegamns Premium Jiuce 59 OZ
5	DAIRY	89029	7789045874	Salted Butter 1 LB
6	DAIRY	89030	7789044327	Extra Cheesy Pizza Lunchable 4.2 OZ
7	DAIRY	89030	4450097756	Wegman Organic Eggs 24 OZ
8	DAIRY	89031	7789045172	Wegman Light Sour Cream 16 OZ
9	DAIRY	89032	7789045875	Oreo Cookies 14.3 OZ
10	DAIRY	89033	81829001470	strawberry Jell-O 13.5 OZ
11	FROZEN	89034	7789041053	Oreo Ice Cream 1.5 QT
12	DAIRY	89035	7789045870	Unsweeted Almond Milk 6.4 OZ
13	DAIRY	89036	81829001470	Texas Pete Hot Sauce 12 OZ
14	DAIRY	89037	4311902875	Pizza Crust 12 OZ
15	DAIRY	89038	9396600373	Hormel Pepperoni 3.79
16	DAIRY	89039	9396600028	Wegmans Organic Half and Half 16 OZ
17	DAIRY	89040	9396600033	Mexican Shreded Cheese 8 OZ
18	DAIRY	89041	4850002107	Ricotta Cheese Part Skim 15 OZ
19	DAIRY	89042	7789045731	4% Cottage Cheese 24 OZ
20	DAIRY	89043	839626	Chobani Mango Yogurt 5.3 OZ
21	FROZEN	89044	839627	Cheese Pizza 88.8 OZ
22	GROCERY	89045	8396278	Mac N Cheese 7.25 OZ
23	GROCERY	89046	839629	Pasta 64 OZ
24	GROCERY	89047	839630	Spaghetti 64 OZ
25	GROCERY	89048	839631	Bread Crumb 10 OZ
26	DAIRY	89049	839632	Yeast 0.6 OZ
27	DAIRY	89050	839633	Raspberry Lemon Ginger Kombucha 12 OZ

Query executed successfully.

Orders Table

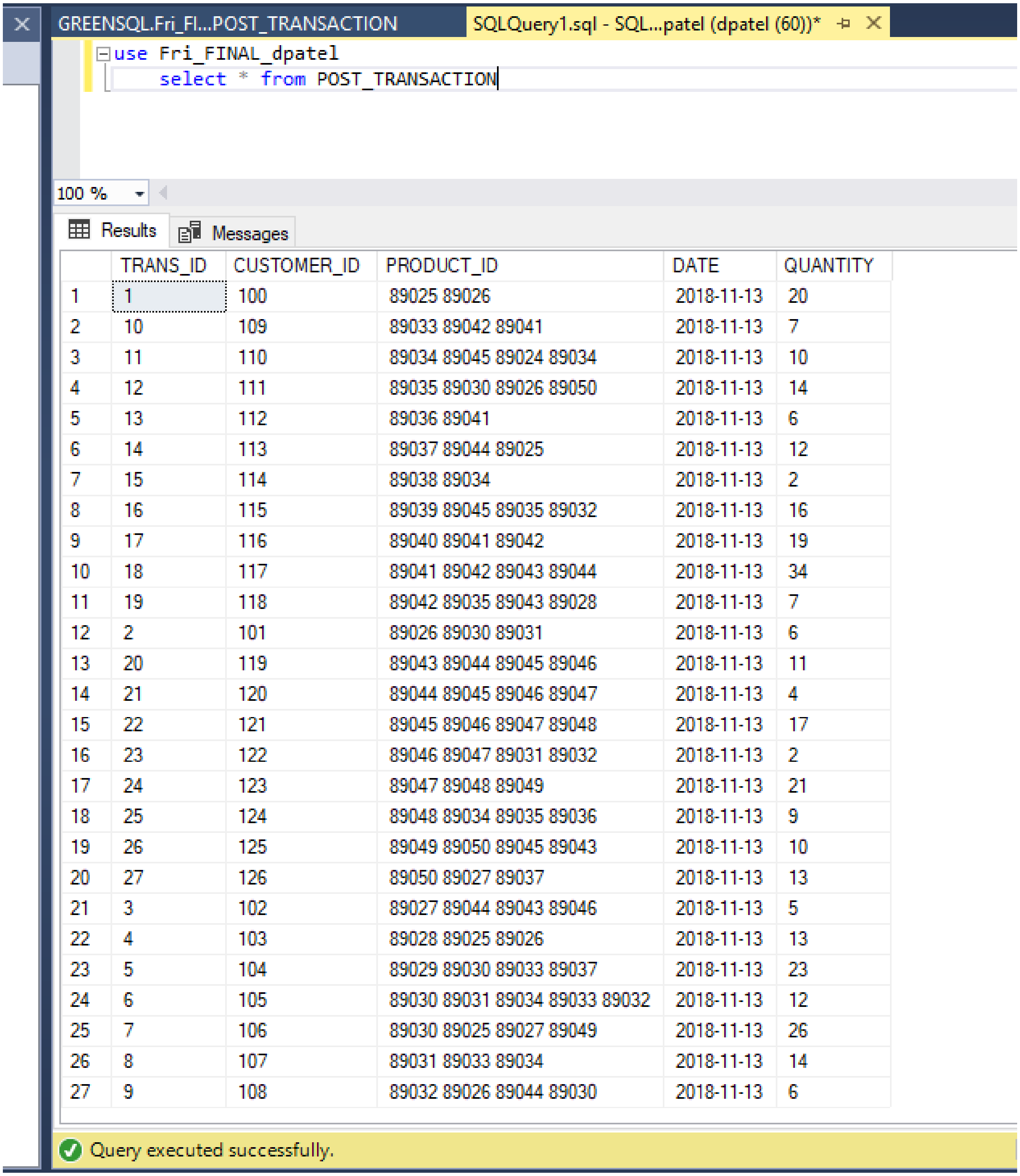


```
use Fri_FINAL_dpatel
    select * from ORDERS
```

	ORDER_ID	PRODUCT_ID	ORDER_QUANTITY	PRODUCT_NAME	PRODUCT_PRICE	ON_HANDPRODUCT
1	200	89025	100	Cream Cheese	1.09	96
2	201	89026	40	Pickle	2.99	35
3	202	89027	400	Milk	3.79	397
4	203	89028	50	Jiuce	2.48	48
5	204	89029	96	Butter	2.79	95
6	205	89030	20	Lunchable	7.99	14
7	206	89030	300	Eggs	4.99	19
8	207	89031	48	Sour Cream	1.79	44
9	208	89032	70	Cookies	2.99	66
10	209	89033	30	Jell-O	2.29	26
11	210	89034	9	Ice Cream	2.50	4
12	211	89035	20	Almond Milk	1.78	16
13	212	89036	15	Hot Sauce	2.99	13
14	213	89037	8	Pizza Dough	5.99	5
15	214	89038	14	Pepperoni	2.50	13
16	215	89039	48	Half and Half	1.79	47
17	216	89040	20	Shreded Cheese	1.99	19
18	217	89041	10	Ricotta Cheese	4.99	7
19	218	89042	10	Cottage Cheese	3.99	6
20	219	89043	30	Yogurt	1.00	25
21	220	89044	20	Pizza	8.69	14
22	221	89045	48	Mac N Cheese	0.99	42
23	222	89046	25	Pasta	1.29	20
24	223	89047	13	Spaghetti	1.19	9
25	224	89048	32	Bread Crumb	3.49	29
26	225	89049	10	Yeast	1.49	7
27	226	89050	28	kombucha	1.99	25

✓ Query executed successfully.

POST_TRANSACTION Table

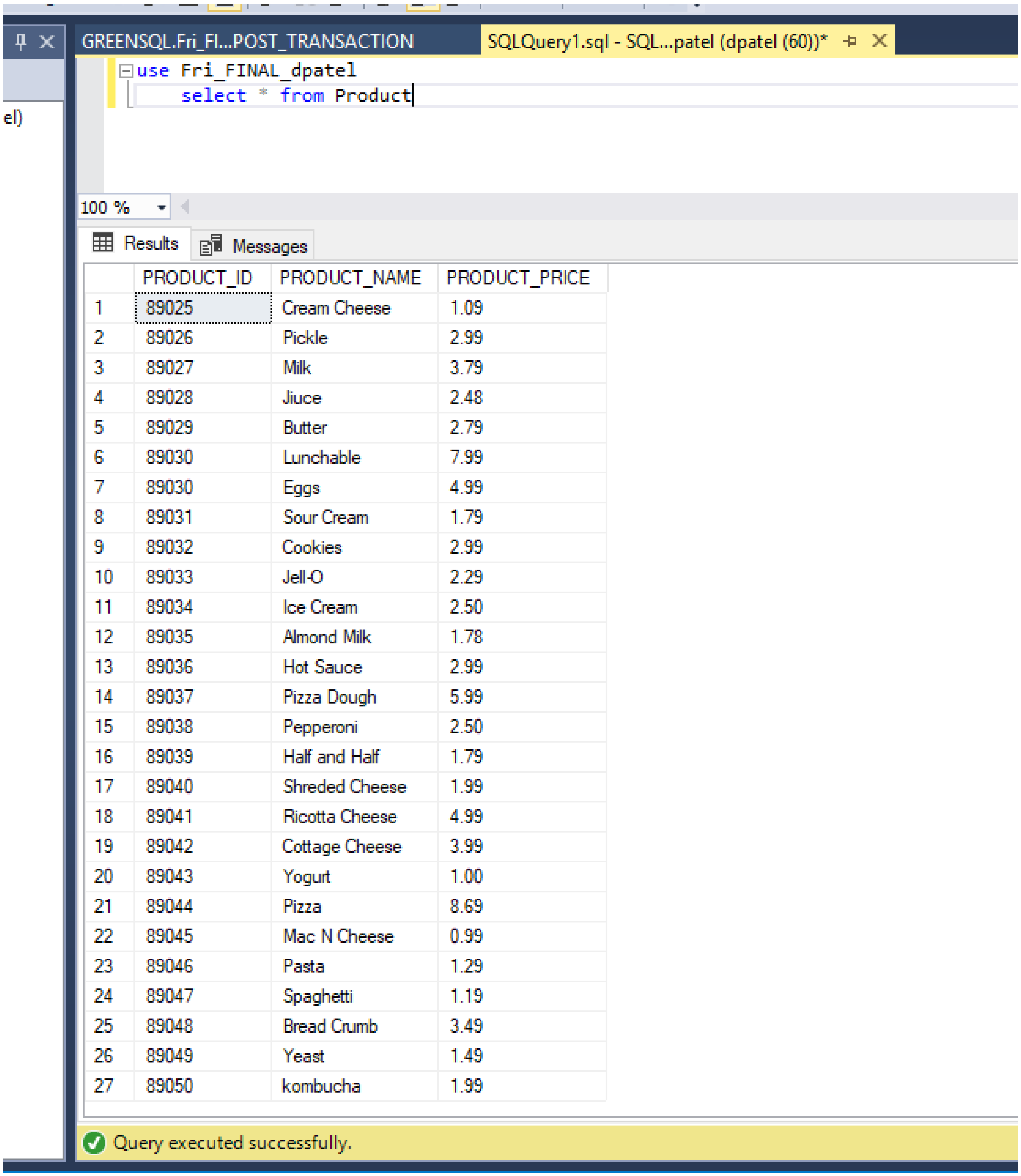

GREENSQL.Fri_FI...POST_TRANSACTION
SQLQuery1.sql - SQL...patel (dpatel (60))*
100 %
Results
Messages

```
use Fri_FINAL_dpatel
    select * from POST_TRANSACTION
```

	TRANS_ID	CUSTOMER_ID	PRODUCT_ID	DATE	QUANTITY
1	1	100	89025 89026	2018-11-13	20
2	10	109	89033 89042 89041	2018-11-13	7
3	11	110	89034 89045 89024 89034	2018-11-13	10
4	12	111	89035 89030 89026 89050	2018-11-13	14
5	13	112	89036 89041	2018-11-13	6
6	14	113	89037 89044 89025	2018-11-13	12
7	15	114	89038 89034	2018-11-13	2
8	16	115	89039 89045 89035 89032	2018-11-13	16
9	17	116	89040 89041 89042	2018-11-13	19
10	18	117	89041 89042 89043 89044	2018-11-13	34
11	19	118	89042 89035 89043 89028	2018-11-13	7
12	2	101	89026 89030 89031	2018-11-13	6
13	20	119	89043 89044 89045 89046	2018-11-13	11
14	21	120	89044 89045 89046 89047	2018-11-13	4
15	22	121	89045 89046 89047 89048	2018-11-13	17
16	23	122	89046 89047 89031 89032	2018-11-13	2
17	24	123	89047 89048 89049	2018-11-13	21
18	25	124	89048 89034 89035 89036	2018-11-13	9
19	26	125	89049 89050 89045 89043	2018-11-13	10
20	27	126	89050 89027 89037	2018-11-13	13
21	3	102	89027 89044 89043 89046	2018-11-13	5
22	4	103	89028 89025 89026	2018-11-13	13
23	5	104	89029 89030 89033 89037	2018-11-13	23
24	6	105	89030 89031 89034 89033 89032	2018-11-13	12
25	7	106	89030 89025 89027 89049	2018-11-13	26
26	8	107	89031 89033 89034	2018-11-13	14
27	9	108	89032 89026 89044 89030	2018-11-13	6

✓ Query executed successfully.

Product Table


GREENSQL.Fri_FI...POST_TRANSACTION
SQLQuery1.sql - SQL...patel (dpatel (60))*
el)
100 %
Results
Messages

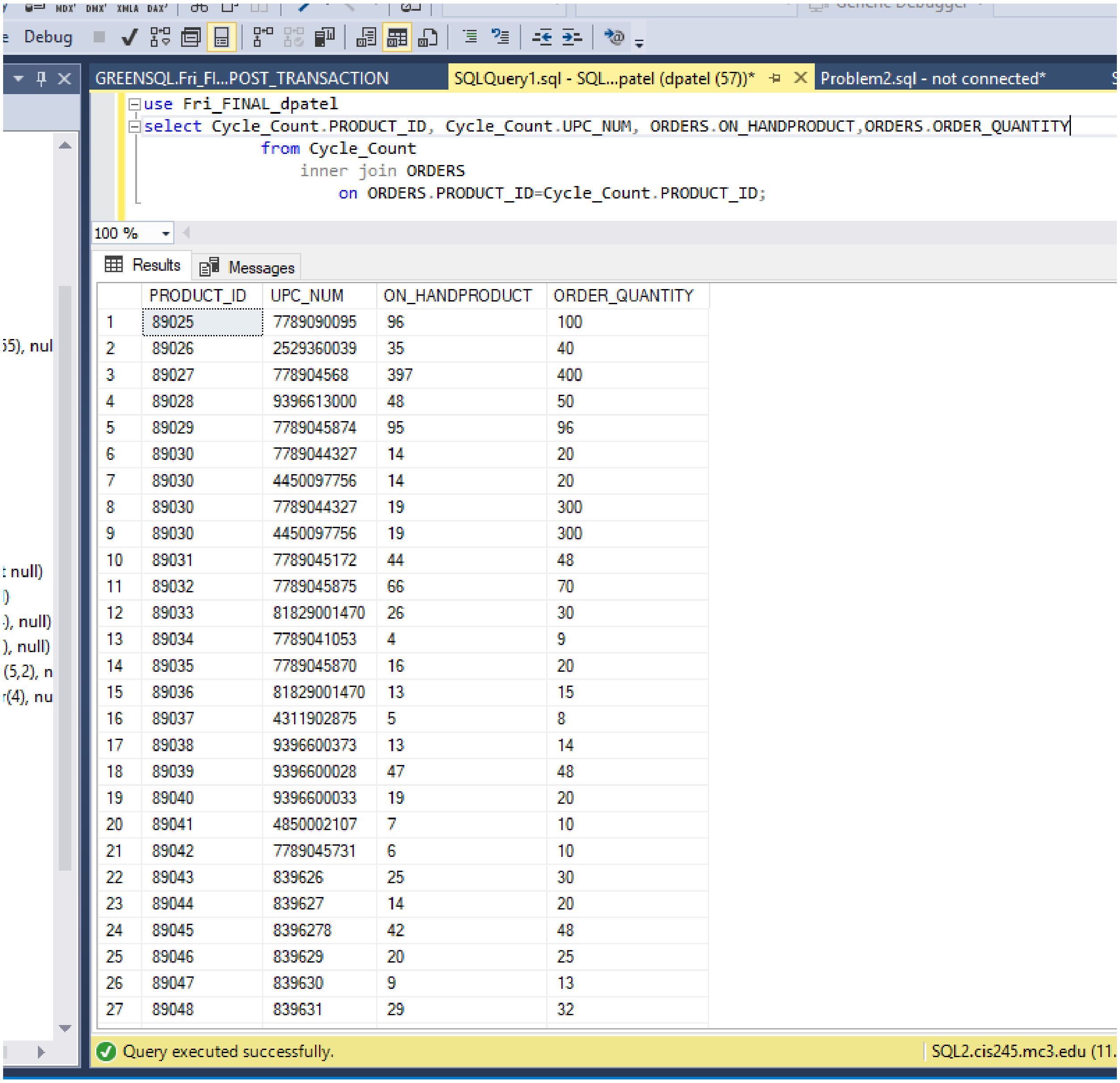
```
use Fri_FINAL_dpatel
    select * from Product
```

	PRODUCT_ID	PRODUCT_NAME	PRODUCT_PRICE
1	89025	Cream Cheese	1.09
2	89026	Pickle	2.99
3	89027	Milk	3.79
4	89028	Jiuce	2.48
5	89029	Butter	2.79
6	89030	Lunchable	7.99
7	89030	Eggs	4.99
8	89031	Sour Cream	1.79
9	89032	Cookies	2.99
10	89033	Jell-O	2.29
11	89034	Ice Cream	2.50
12	89035	Almond Milk	1.78
13	89036	Hot Sauce	2.99
14	89037	Pizza Dough	5.99
15	89038	Pepperoni	2.50
16	89039	Half and Half	1.79
17	89040	Shreded Cheese	1.99
18	89041	Ricotta Cheese	4.99
19	89042	Cottage Cheese	3.99
20	89043	Yogurt	1.00
21	89044	Pizza	8.69
22	89045	Mac N Cheese	0.99
23	89046	Pasta	1.29
24	89047	Spaghetti	1.19
25	89048	Bread Crumb	3.49
26	89049	Yeast	1.49
27	89050	kombucha	1.99

✓ Query executed successfully.

SUBSET4

My created database help will employee of the company when they will order stuff for the next day. I added new column called on hand product which will tell that how many products does the department has on hands which mean in backstock and on the shelf. In the process of doing order for the next day employee from the department had to count the product or had to assume that how much they going to need for the next day. If they assume the on-hand products and order it that will be lost for the company and also for the customers by not providing enough product to the customers and by getting more things in order company had to pay extra and some of the products has less life on shelf and the department end up with throwing the products out which is waste of food and money both. After adding the on-hand column in the order table employee will be able to do order easily. In the product table the I added one more column called ON_HAND_PRODUCT which tells the employee that how many products they have in available for sale at this moment, so now the order table looks like this with this type of column (PRODUCT_ID) (ORDER_QUANTITY) (PRODUCT_NAME) (PRODUCT_PRICE) and (ON_HAND_PRODUCT). Below attached picture shows the code that will help the employees to order easily. In the table there is column called ON_HANDPRODUCT that shows the available amount of product for sale now. That gives an idea to the employee to order the quantity for the next day. As soon as customer buys the product the ON_HANDPRODUCT number decrease, example if customer buy 1 quantity of PRODUCT_ID 89025 then ON_HANDPRODUCT will decrease by 1 and will be 99. ORDER_QUANTITY column shows the actual quantity to order to be ready for the next day.

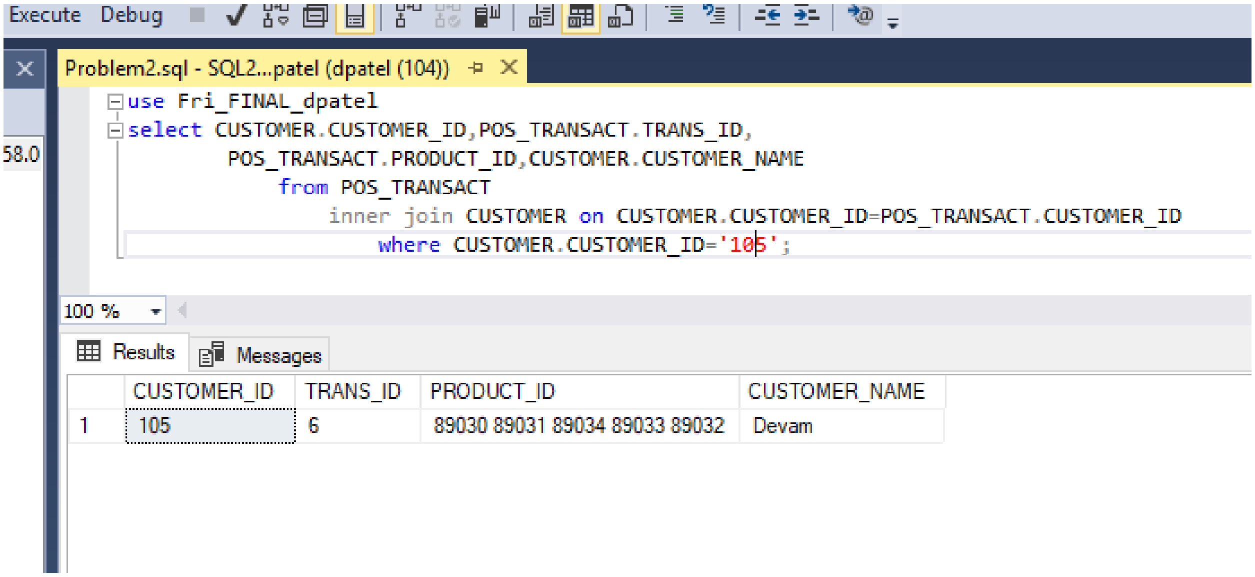

Debug
GREENSQL.Fri_FI...POST_TRANSACTION
SQLQuery1.sql - SQL...patel (dpatel (57))*
Problem2.sql - not connected*
100 %
Results
Messages
SQL2.cis245.mc3.edu (11.

```
use Fri_FINAL_dpatel
select Cycle_Count.PRODUCT_ID, Cycle_Count.UPC_NUM, ORDERS.ON_HANDPRODUCT,ORDERS.ORDER_QUANTITY
            from Cycle_Count
                inner join ORDERS
                    on ORDERS.PRODUCT_ID=Cycle_Count.PRODUCT_ID;
```

	PRODUCT_ID	UPC_NUM	ON_HANDPRODUCT	ORDER_QUANTITY
1	89025	7789090095	96	100
2	89026	2529360039	35	40
3	89027	778904568	397	400
4	89028	9396613000	48	50
5	89029	7789045874	95	96
6	89030	7789044327	14	20
7	89030	4450097756	14	20
8	89030	7789044327	19	300
9	89030	4450097756	19	300
10	89031	7789045172	44	48
11	89032	7789045875	66	70
12	89033	81829001470	26	30
13	89034	7789041053	4	9
14	89035	7789045870	16	20
15	89036	81829001470	13	15
16	89037	4311902875	5	8
17	89038	9396600373	13	14
18	89039	9396600028	47	48
19	89040	9396600033	19	20
20	89041	4850002107	7	10
21	89042	7789045731	6	10
22	89043	839626	25	30
23	89044	839627	14	20
24	89045	8396278	42	48
25	89046	839629	20	25
26	89047	839630	9	13
27	89048	839631	29	32

Query executed successfully.

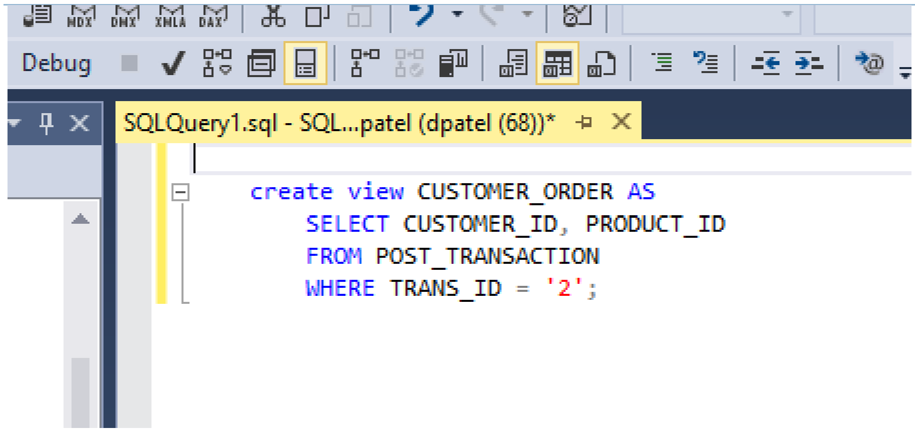
Second database problem for the company was that storing the PRODUCT_IDs that customers bought. If customer lose their receipt they can't return the stuff and they lose money. By that Wegmans was losing the customers. I added new table in the database called POS_TRANSACT. POS_TRANSACT table has 5 column which is (TRANS_ID), (CUSTOMER_ID), (PRODUCT_ID), (DATE) and quantity. TRANS_ID will show the transaction number, CUSTOMER_ID is a number that every customer gets when they shop the products, PRODUCT_ID shows that how many product customers bought. DATE shows the date of the transaction and QUANTITY shows the total amount of quantity customer bought. Now if customer want to return the product and lose the receipt still customer can return the product. Now Wegmans database stores the data now.   Below attached image is the sample database I've created.



```
use Fri_FINAL_dpatel
select CUSTOMER.CUSTOMER_ID,POS_TRANSACT.TRANS_ID,
        POS_TRANSACT.PRODUCT_ID,CUSTOMER.CUSTOMER_NAME
            from POS_TRANSACT
                inner join CUSTOMER on CUSTOMER.CUSTOMER_ID=POS_TRANSACT.CUSTOMER_ID
                    where CUSTOMER.CUSTOMER_ID='105';
```

	CUSTOMER_ID	TRANS_ID	PRODUCT_ID	CUSTOMER_NAME
1	105	6	89030 89031 89034 89033 89032	Devam

This code will give result in CUSTOMER_ID, TRANS_ID, PRODUCT_ID and CUSTOMER_ID. CUSTOMER_ID shows the unique id of the customer. PRODUCT_ID shows the product number, by that it's the proof for the employee and the customer of buying the product.



```
create view CUSTOMER_ORDER AS
    SELECT CUSTOMER_ID, PRODUCT_ID
    FROM POST_TRANSACTION
    WHERE TRANS_ID = '2';
```

I’ve created create view because I don’t want other people to look at the other customers data.

It will restrict other people to look at the data.