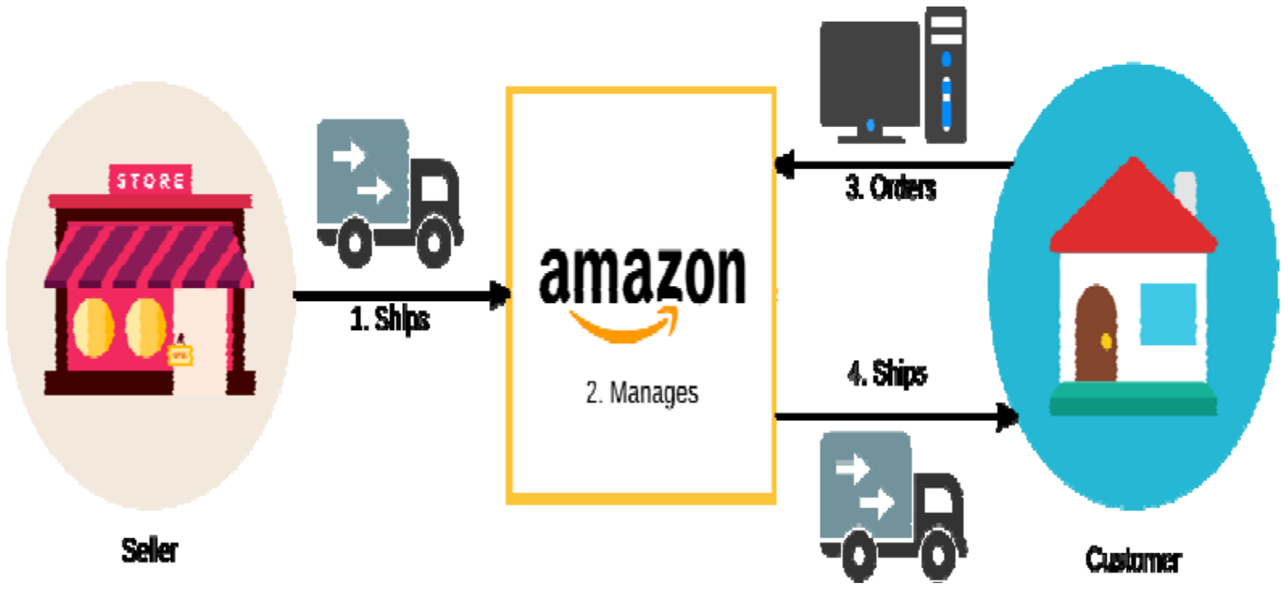
CS699 SQL PROJECT

Classic Version

Louie Angelo Castro Padilla

TABLE OF CONTENTS

INTRODUCTION



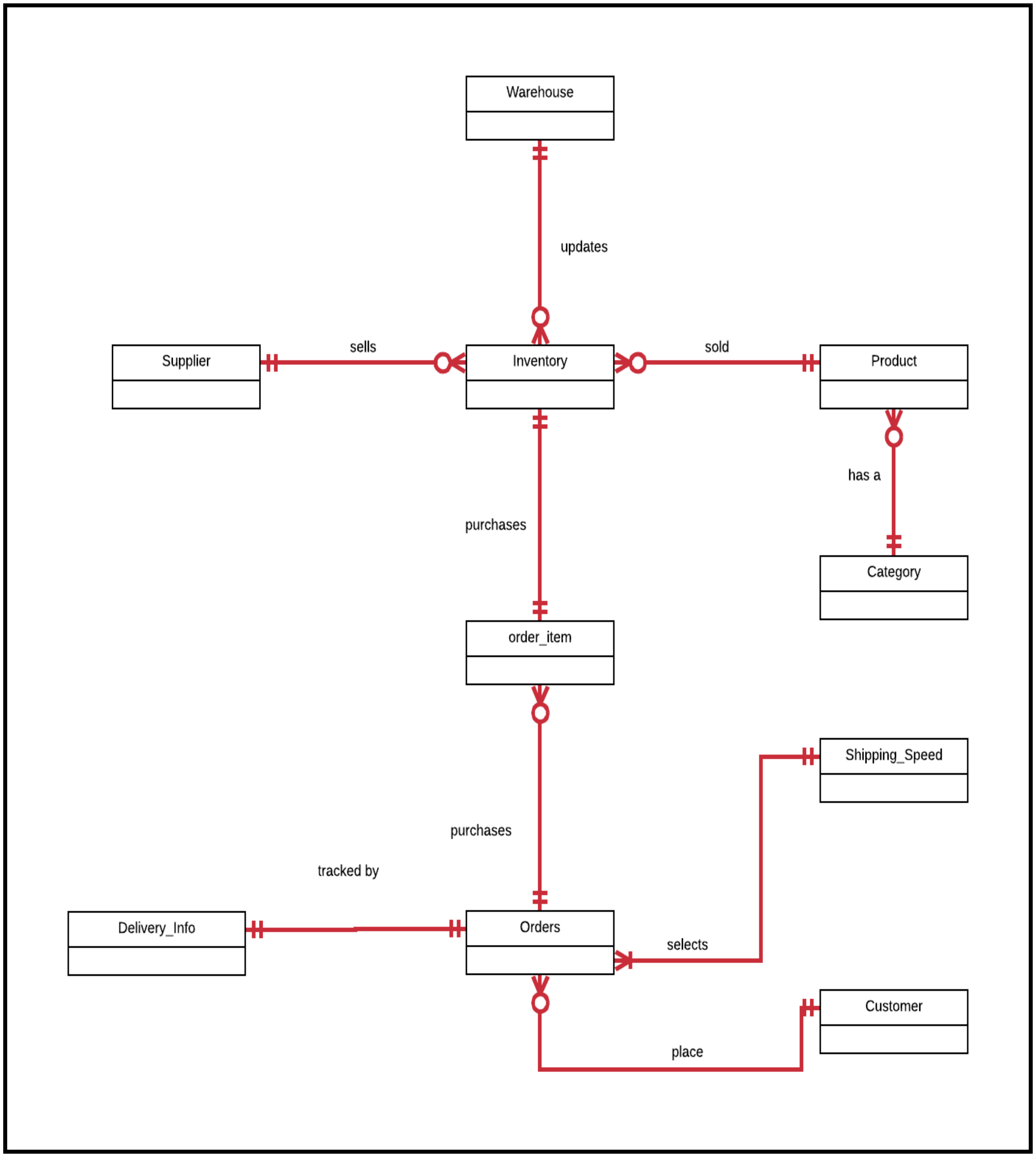
Abstract

This project implements what we have learned during this course term. Using the skills and concepts, we apply them by replicating and crafting a part of Amazon’s database design. I am using MS SQL Server as my database for this project with the focus on the classic version. This includes the design of my database from the business rules to the physical coding. This will also cover uses cases and aspect problems given. Using MS SQL Server the solution to each problem is demonstrated.

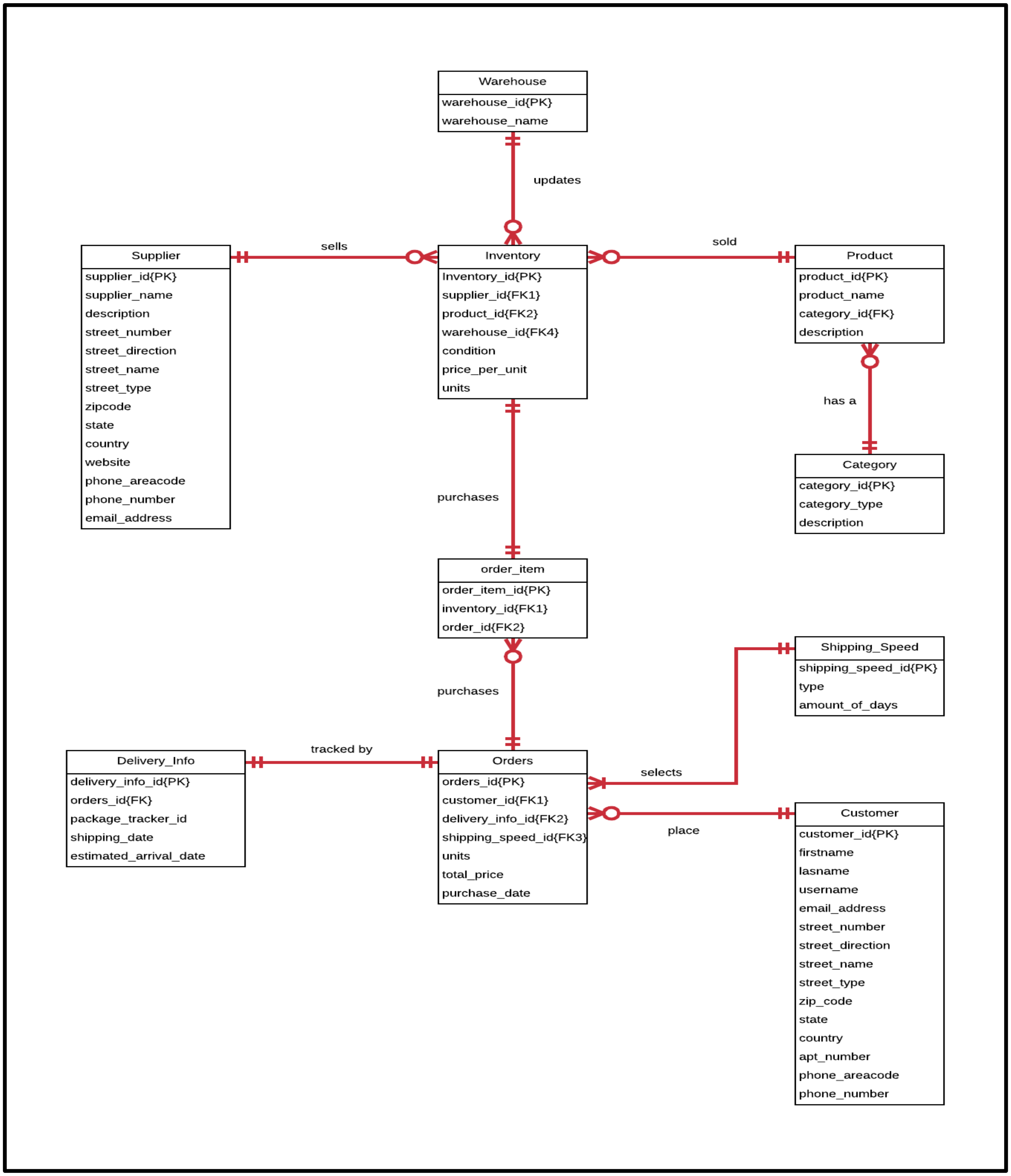
Business Rules

- A warehouse must store at least 1 product; a product must be stored in at least 1 warehouse (M:n)
- A supplier may sell 0 or more products (0:m); a product must be sold by at least one seller (1:m) The bridge entity would be ‘Inventory’ that holds products, supplier’s IDs and more attributes
- A supplier may be listed 0 or more times in the inventory (1:m); an inventory must list at least 1 or many suppliers. (1:m)
- A product may be listed in the inventory 0 or more times(1:M); an inventory must list at least 1 or many products (1:m)
- A product may be purchased through 0 or more orders; an order purchases 0 or more products (M:N) Connected through the inventory bridging entity
- A product must have a category (1:1); a category may have many products (1:m)
- A customer may place 0 to many orders(0:M); an order must be placed by 1 customer (1:1)
- An order must include one delivery information (1:1); a delivery information must include 1 order (1:1)
- An order must have one shipping type (1:1); a shipping type must be in the order (1:1)
- An order must contain one or more order_item(1:m); an order_item must be linked to one order information (1:1)

Conceptual ERD



Logical ERD


Warehouse
warehouse_id{PK}
warehouse_name
updates
Supplier
supplier_id{PK}
supplier_name
description
street_number
street_direction
street_name
street_type
zipcode
state
country
website
phone_areacode
phone_number
email_address
sells
Inventory
Inventory_id{PK}
supplier_id{FK1}
product_id{FK2}
warehouse_id{FK4}
condition
price_per_unit
units
sold
Product
product_id{PK}
product_name
category_id{FK}
description
has a
Category
category_id{PK}
category_type
description
purchases
order_item
order_item_id{PK}
inventory_id{FK1}
order_id{FK2}
purchases
Shipping_Speed
shipping_speed_id{PK}
type
amount_of_days
Delivery_Info
delivery_info_id{PK}
orders_id{FK}
package_tracker_id
shipping_date
estimated_arrival_date
tracked by
Orders
orders_id{PK}
customer_id{FK1}
delivery_info_id{FK2}
shipping_speed_id{FK3}
units
total_price
purchase_date
selects
place
Customer
customer_id{PK}
firstname
lasname
username
email_address
street_number
street_direction
street_name
street_type
zip_code
state
country
apt_number
phone_areacode
phone_number

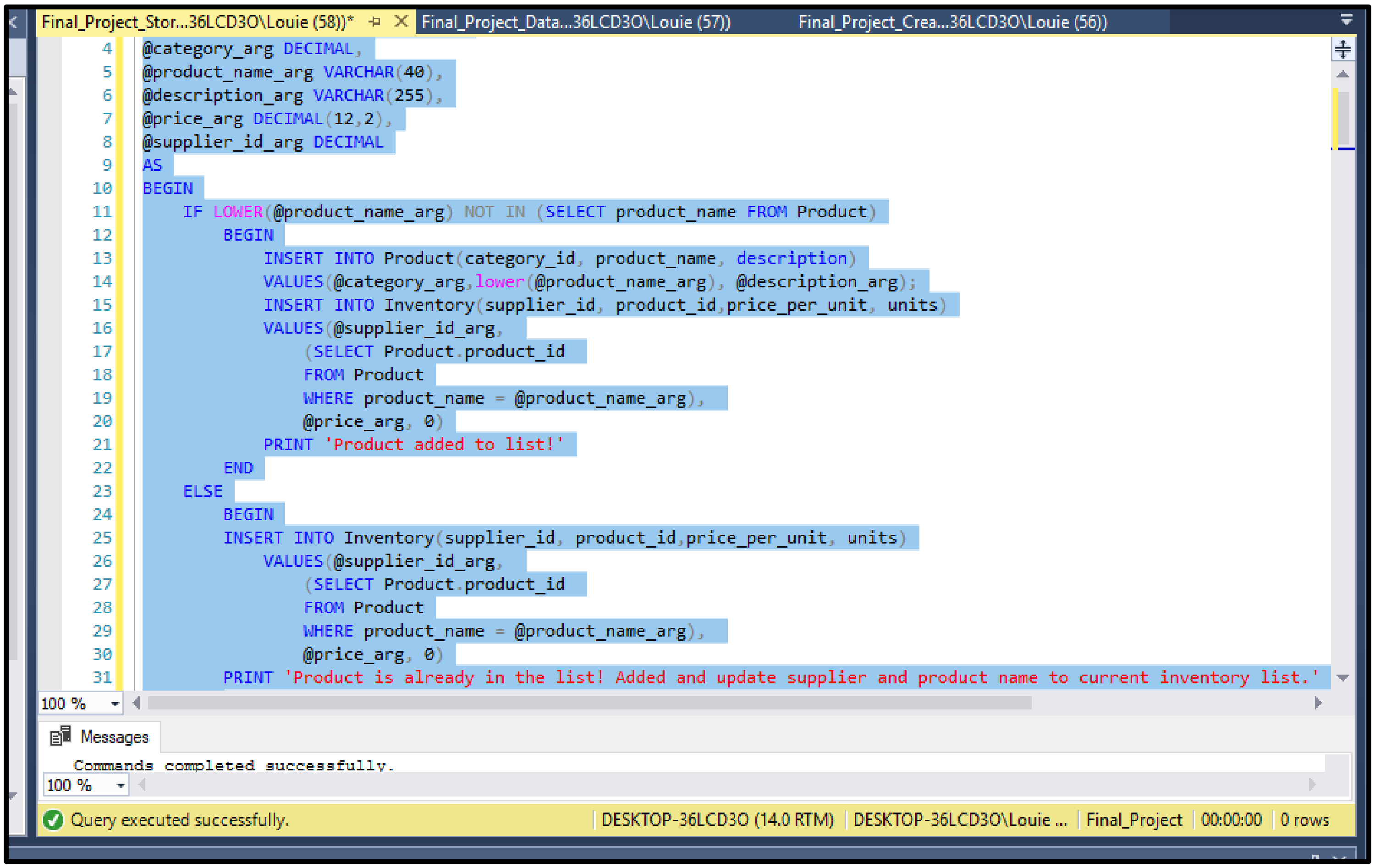
The logical ERD contains how the database is designed. The main entities are Customer, Orders, Shipping_speed, Delivery_info, Product, Category, Inventory, Warehouse, and Supplier. The Order_item is considered as a bridging entity that shows the relationship between customer orders and what they have purchased. In this design you will see attributes that could be in its entity, for example the address could be a separate entity both for customer and supplier. For this project I chose to keep them within their perspective entities since they are unique to either the supplier id or the customer id. Just by knowing those two IDs you will be able to see the address of each supplier or customer.

Aspect 1

New Product Use Case – This occurs when a seller plans to sell a product it has not sold before.

1. *The seller searches Amazon's product list to determine if another seller is already selling the product.*
2. *If a different seller is already selling the product, a new listing is not required; the seller re-uses the same listing.*
3. *If the product is not yet sold on Amazon, a new listing is created with the product's name, description, price, and other relevant items. Every product added is linked to a product category (all categories are predefined by Amazon), for example, "Computers", "Electronics", "Appliances", and similar.*

Stored Procedure



Final_Project_Stor...36LCD3O\Louie (58))* Final_Project_Data...36LCD3O\Louie (57)) Final_Project_Crea...36LCD3O\Louie (56))

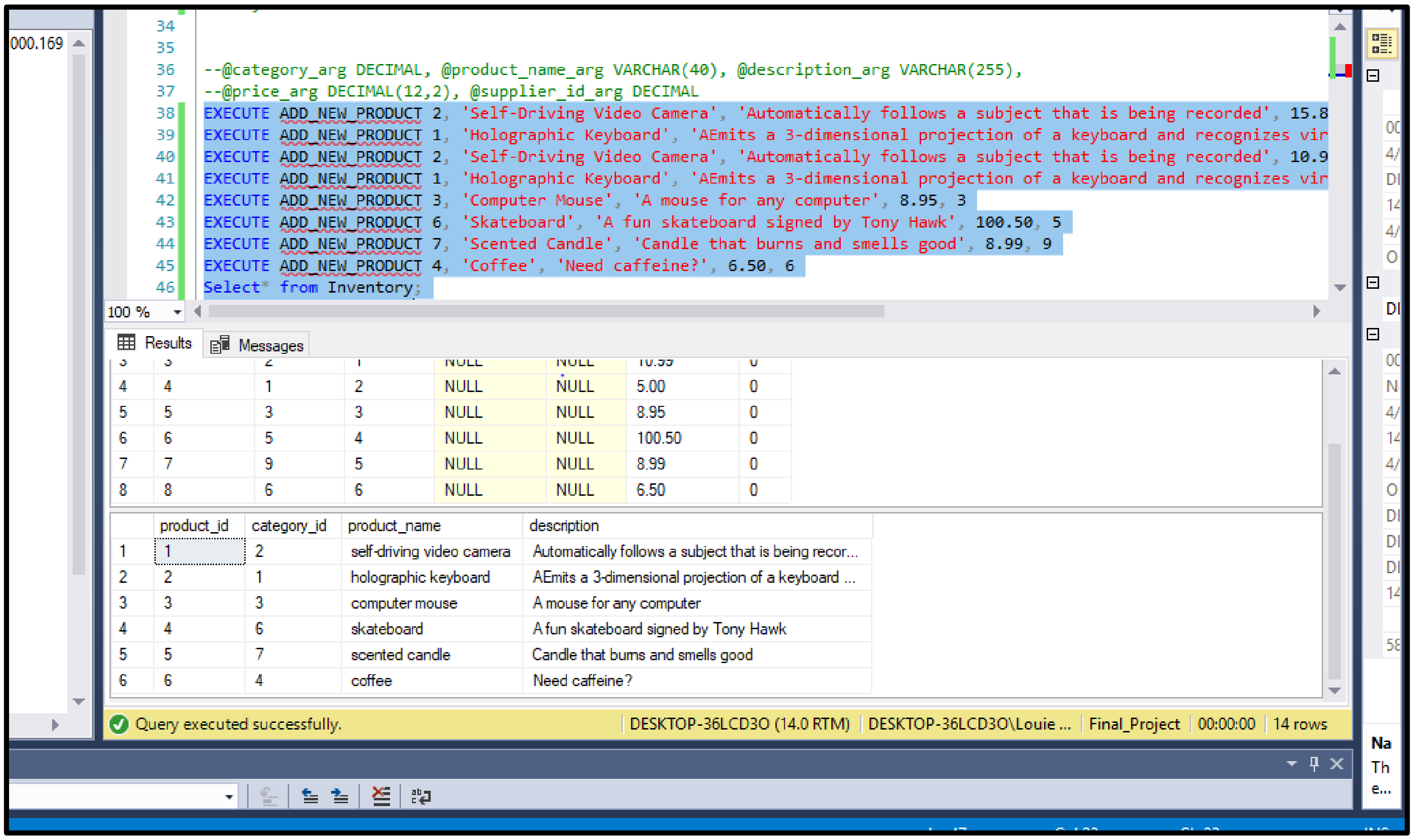
```sql
@category_arg DECIMAL,
@product_name_arg VARCHAR(40),
@description_arg VARCHAR(255),
@price_arg DECIMAL(12,2),
@supplier_id_arg DECIMAL
AS
BEGIN
    IF LOWER(@product_name_arg) NOT IN (SELECT product_name FROM Product)
        BEGIN
            INSERT INTO Product(category_id, product_name, description)
            VALUES(@category_arg,lower(@product_name_arg), @description_arg);
            INSERT INTO Inventory(supplier_id, product_id,price_per_unit, units)
            VALUES(@supplier_id_arg,
                (SELECT Product.product_id
                FROM Product
                WHERE product_name = @product_name_arg),
                @price_arg, 0)
            PRINT 'Product added to list!'
        END
    ELSE
        BEGIN
        INSERT INTO Inventory(supplier_id, product_id,price_per_unit, units)
            VALUES(@supplier_id_arg,
                (SELECT Product.product_id
                FROM Product
                WHERE product_name = @product_name_arg),
                @price_arg, 0)
        PRINT 'Product is already in the list! Added and update supplier and product name to current inventory list.'
```

Messages

```
Commands completed successfully.
```

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 0 rows

Aspect 1C. A seller adds two new products. The first is a self-driving video camera which automatically follows a subject that is being recorded. The second is a holographic keyboard that emits a three-dimensional projection of a keyboard and recognizes virtual key presses from the typist. Invoke the stored procedure twice to add these products, keeping in mind that products have at a minimum a name, description, price, and category.



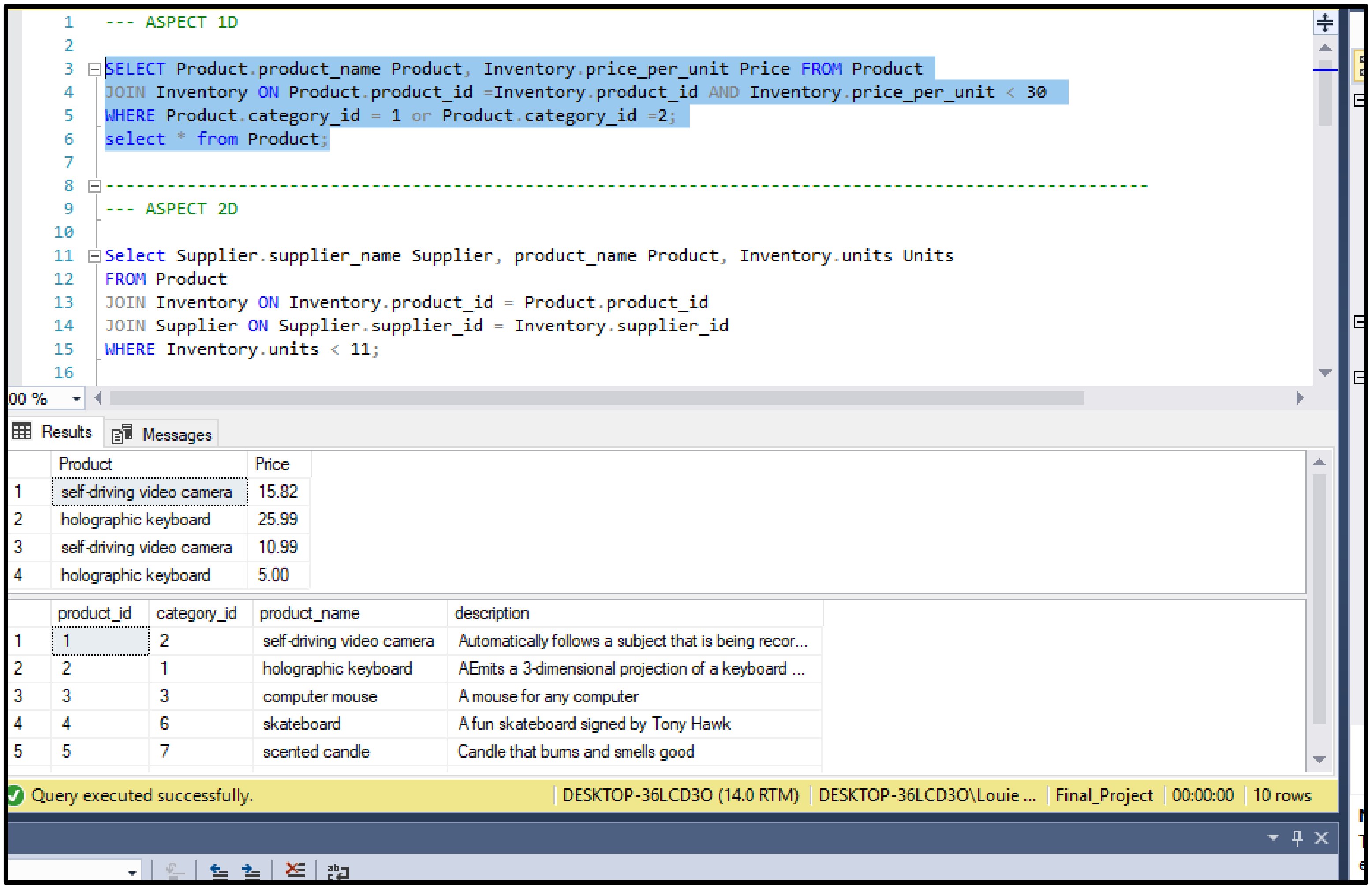
```
--@category_arg DECIMAL, @product_name_arg VARCHAR(40), @description_arg VARCHAR(255),
--@price_arg DECIMAL(12,2), @supplier_id_arg DECIMAL
EXECUTE ADD_NEW_PRODUCT 2, 'Self-Driving Video Camera', 'Automatically follows a subject that is being recorded', 15.8
EXECUTE ADD_NEW_PRODUCT 1, 'Holographic Keyboard', 'AEmits a 3-dimensional projection of a keyboard and recognizes vir
EXECUTE ADD_NEW_PRODUCT 2, 'Self-Driving Video Camera', 'Automatically follows a subject that is being recorded', 10.9
EXECUTE ADD_NEW_PRODUCT 1, 'Holographic Keyboard', 'AEmits a 3-dimensional projection of a keyboard and recognizes vir
EXECUTE ADD_NEW_PRODUCT 3, 'Computer Mouse', 'A mouse for any computer', 8.95, 3
EXECUTE ADD_NEW_PRODUCT 6, 'Skateboard', 'A fun skateboard signed by Tony Hawk', 100.50, 5
EXECUTE ADD_NEW_PRODUCT 7, 'Scented Candle', 'Candle that burns and smells good', 8.99, 9
EXECUTE ADD_NEW_PRODUCT 4, 'Coffee', 'Need caffeine?', 6.50, 6
Select* from Inventory;
```

3	3	2	1	NULL	NULL	10.99	0
4	4	1	2	NULL	NULL	5.00	0
5	5	3	3	NULL	NULL	8.95	0
6	6	5	4	NULL	NULL	100.50	0
7	7	9	5	NULL	NULL	8.99	0
8	8	6	6	NULL	NULL	6.50	0

	product_id	category_id	product_name	description
1	1	2	self-driving video camera	Automatically follows a subject that is being recor...
2	2	1	holographic keyboard	AEmits a 3-dimensional projection of a keyboard ...
3	3	3	computer mouse	A mouse for any computer
4	4	6	skateboard	A fun skateboard signed by Tony Hawk
5	5	7	scented candle	Candle that burns and smells good
6	6	4	coffee	Need caffeine?

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 14 rows

Aspect 1D. A seller is considering developing a new electronic product and requests a list of existing products in the “Computers” or “Electronics” categories that cost $30 or less. Develop and execute a single query that provides this information.



```
--- ASPECT 1D

SELECT Product.product_name Product, Inventory.price_per_unit Price FROM Product
JOIN Inventory ON Product.product_id =Inventory.product_id AND Inventory.price_per_unit < 30
WHERE Product.category_id = 1 or Product.category_id =2;
select * from Product;

-----------------------------------------------------------------------------------------------------
--- ASPECT 2D

Select Supplier.supplier_name Supplier, product_name Product, Inventory.units Units
FROM Product
JOIN Inventory ON Inventory.product_id = Product.product_id
JOIN Supplier ON Supplier.supplier_id = Inventory.supplier_id
WHERE Inventory.units < 11;
```

	Product	Price
1	self-driving video camera	15.82
2	holographic keyboard	25.99
3	self-driving video camera	10.99
4	holographic keyboard	5.00

	product_id	category_id	product_name	description
1	1	2	self-driving video camera	Automatically follows a subject that is being recor...
2	2	1	holographic keyboard	AEmits a 3-dimensional projection of a keyboard ...
3	3	3	computer mouse	A mouse for any computer
4	4	6	skateboard	A fun skateboard signed by Tony Hawk
5	5	7	scented candle	Candle that burns and smells good

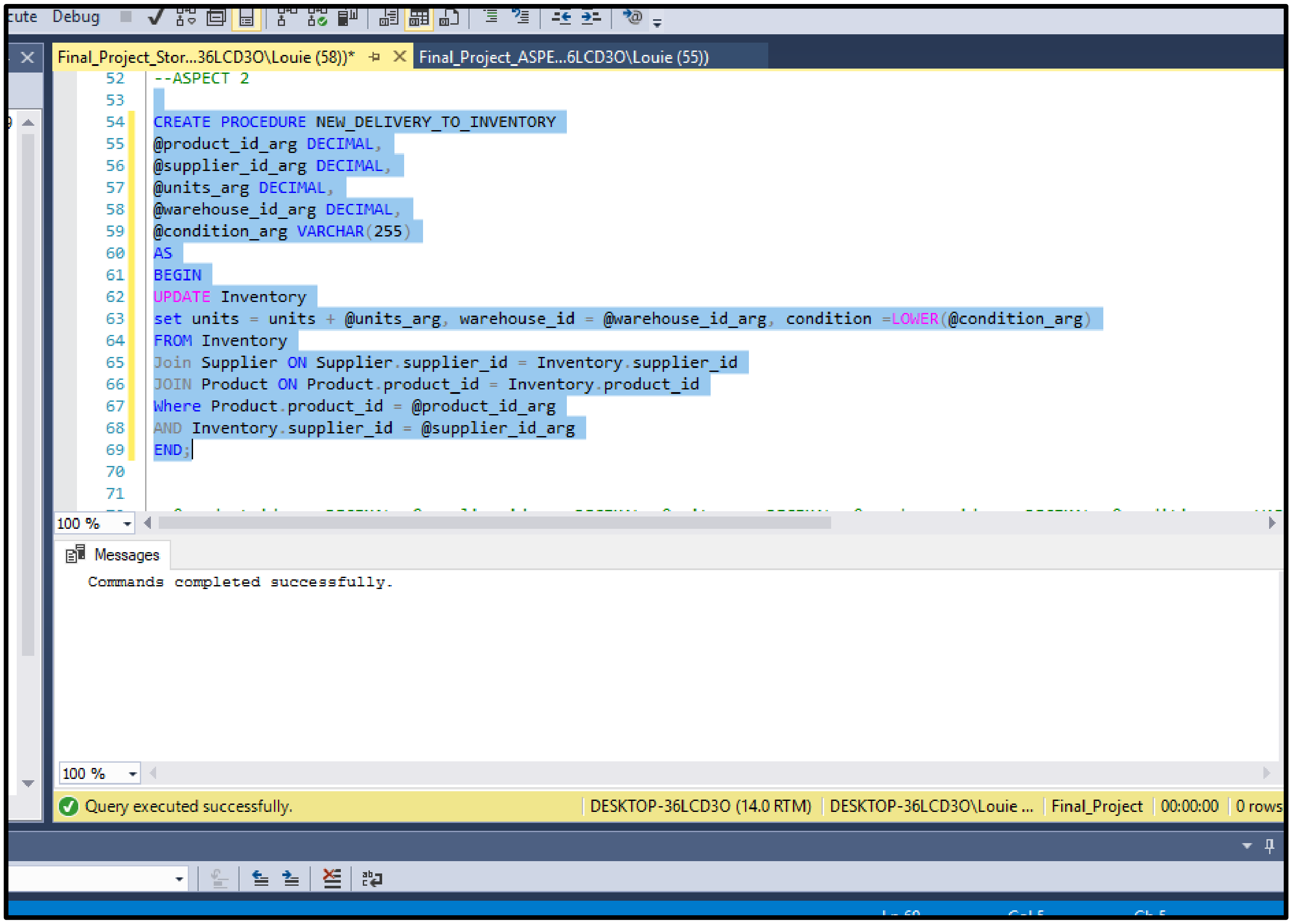
Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 10 rows

Aspect 2

Product Delivery Use Case – This occurs when a seller sends one or more units of a product to Amazon so that they can be sold.

1. *The seller ships one or more units of a product to Amazon's warehouse, along with information that indicates to Amazon what the product is, how many units there are, and the condition (new, used, etc …).*
2. *After Amazon receives the product(s), it updates the seller's inventory so that customers can purchase the product.*

Stored Procedure

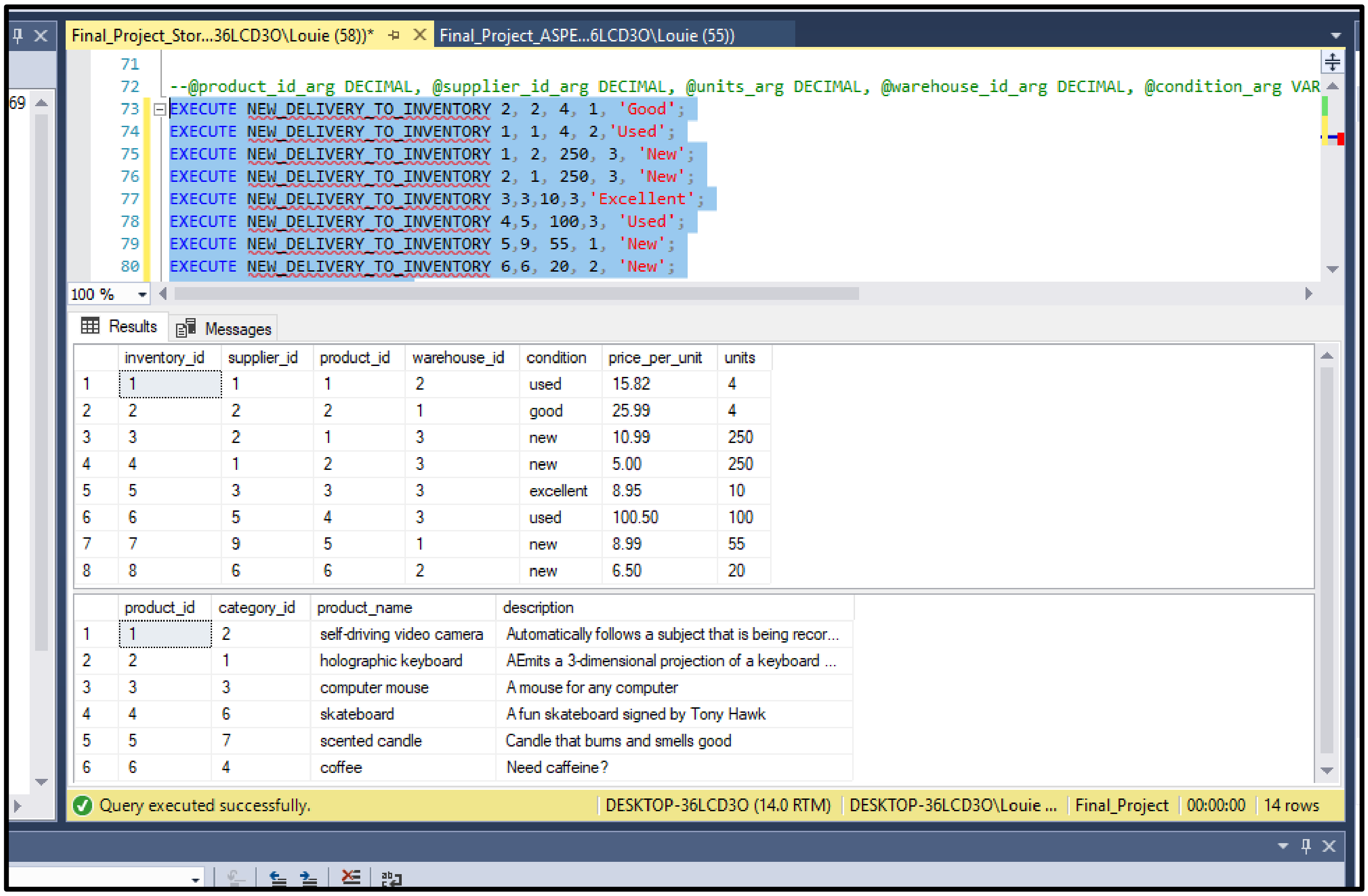


```
--ASPECT 2

CREATE PROCEDURE NEW_DELIVERY_TO_INVENTORY
@product_id_arg DECIMAL,
@supplier_id_arg DECIMAL,
@units_arg DECIMAL,
@warehouse_id_arg DECIMAL,
@condition_arg VARCHAR(255)
AS
BEGIN
UPDATE Inventory
set units = units + @units_arg, warehouse_id = @warehouse_id_arg, condition =LOWER(@condition_arg)
FROM Inventory
Join Supplier ON Supplier.supplier_id = Inventory.supplier_id
JOIN Product ON Product.product_id = Inventory.product_id
Where Product.product_id = @product_id_arg
AND Inventory.supplier_id = @supplier_id_arg
END;
```

```
Commands completed successfully.
```

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 0 rows

Aspect 2C. A seller delivers four each of the two new products added in Aspect 1 (the self-driving video camera and the holographic keyboard). Invoke the stored procedure twice to update the inventory of these products for a seller of your choosing.



Final_Project_Stor...36LCD3O\Louie (58))* Final_Project_ASPE...6LCD3O\Louie (55))

```sql
--@product_id_arg DECIMAL, @supplier_id_arg DECIMAL, @units_arg DECIMAL, @warehouse_id_arg DECIMAL, @condition_arg VAR
EXECUTE NEW_DELIVERY_TO_INVENTORY 2, 2, 4, 1, 'Good';
EXECUTE NEW_DELIVERY_TO_INVENTORY 1, 1, 4, 2,'Used';
EXECUTE NEW_DELIVERY_TO_INVENTORY 1, 2, 250, 3, 'New';
EXECUTE NEW_DELIVERY_TO_INVENTORY 2, 1, 250, 3, 'New';
EXECUTE NEW_DELIVERY_TO_INVENTORY 3,3,10,3,'Excellent';
EXECUTE NEW_DELIVERY_TO_INVENTORY 4,5, 100,3, 'Used';
EXECUTE NEW_DELIVERY_TO_INVENTORY 5,9, 55, 1, 'New';
EXECUTE NEW_DELIVERY_TO_INVENTORY 6,6, 20, 2, 'New';
```

100 %

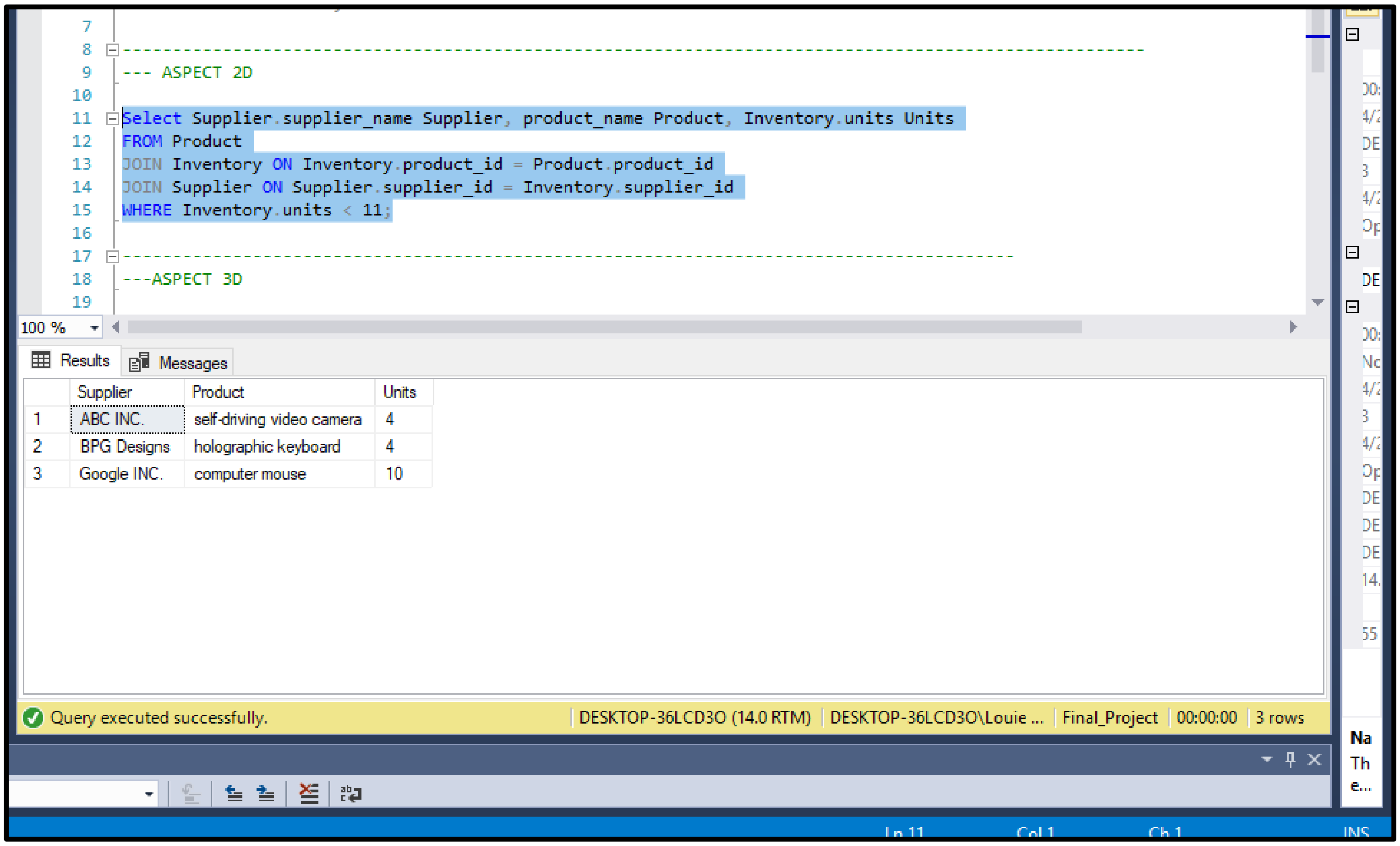
Results Messages

	inventory_id	supplier_id	product_id	warehouse_id	condition	price_per_unit	units
1	1	1	1	2	used	15.82	4
2	2	2	2	1	good	25.99	4
3	3	2	1	3	new	10.99	250
4	4	1	2	3	new	5.00	250
5	5	3	3	3	excellent	8.95	10
6	6	5	4	3	used	100.50	100
7	7	9	5	1	new	8.99	55
8	8	6	6	2	new	6.50	20

	product_id	category_id	product_name	description
1	1	2	self-driving video camera	Automatically follows a subject that is being recor...
2	2	1	holographic keyboard	AEmits a 3-dimensional projection of a keyboard ...
3	3	3	computer mouse	A mouse for any computer
4	4	6	skateboard	A fun skateboard signed by Tony Hawk
5	5	7	scented candle	Candle that burns and smells good
6	6	4	coffee	Need caffeine?

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 14 rows

Aspect 2D. The seller from b above requests a listing of all of its products that have an inventory of 11 or less. Develop and execute a single query that provides this information (the self-driving video camera and holographic keyboard should be among those listed).



```
----------------------------------------------------------------------------------------------------
--- ASPECT 2D

Select Supplier.supplier_name Supplier, product_name Product, Inventory.units Units
FROM Product
JOIN Inventory ON Inventory.product_id = Product.product_id
JOIN Supplier ON Supplier.supplier_id = Inventory.supplier_id
WHERE Inventory.units < 11;

-------------------------------------------------------------------------------------------
---ASPECT 3D
```

	Supplier	Product	Units
1	ABC INC.	self-driving video camera	4
2	BPG Designs	holographic keyboard	4
3	Google INC.	computer mouse	10

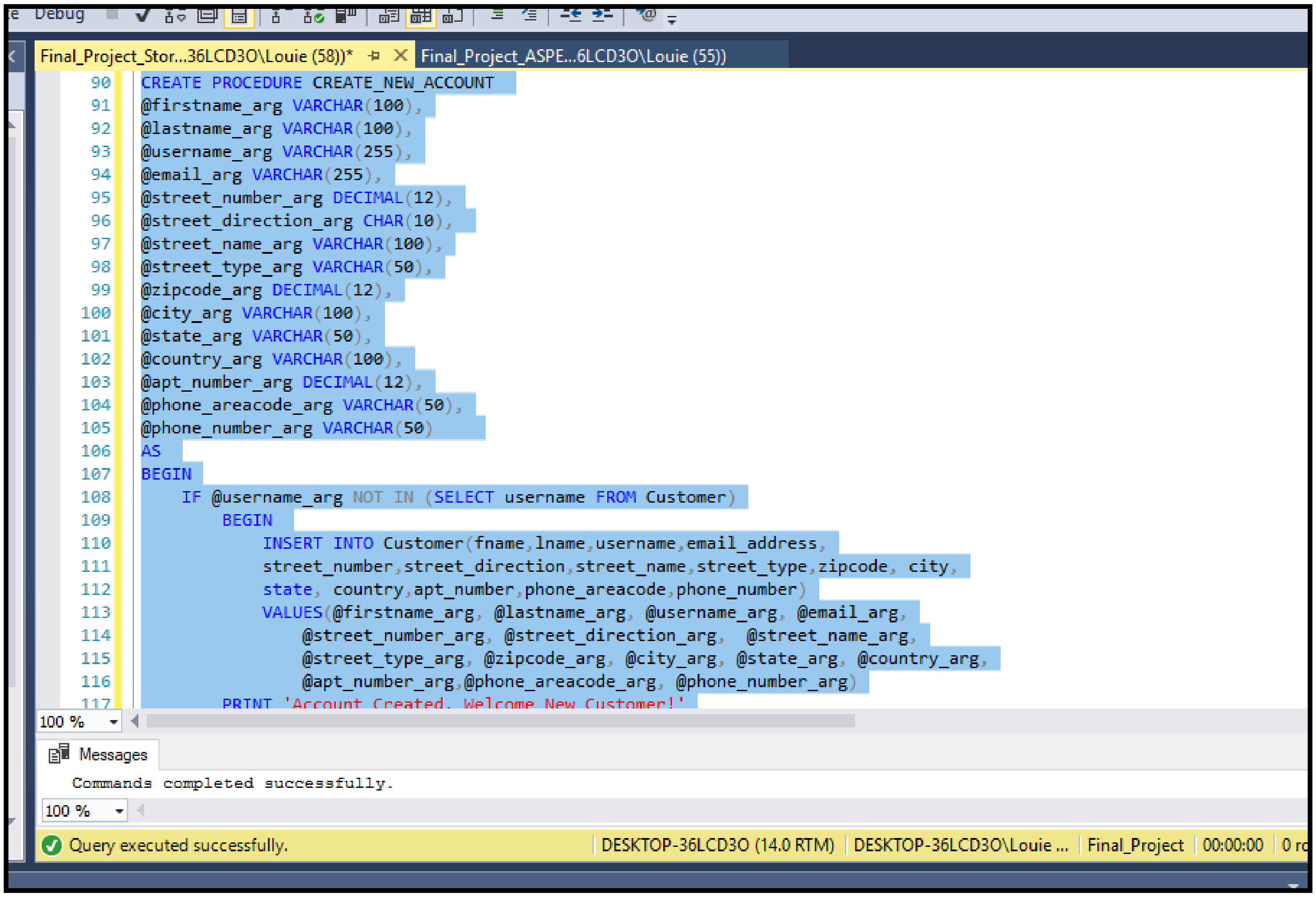
Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 3 rows

Aspect 3

New Customer Account Use Case – This occurs when a customer signs up for an account on Amazon, so they can begin purchasing products.

1. *The customer provides Amazon with basic information including a username, an address, phone number, and an email address.*
2. *Amazon creates an account for the customer, enabling the customer to purchase products.*

Stored Procedure

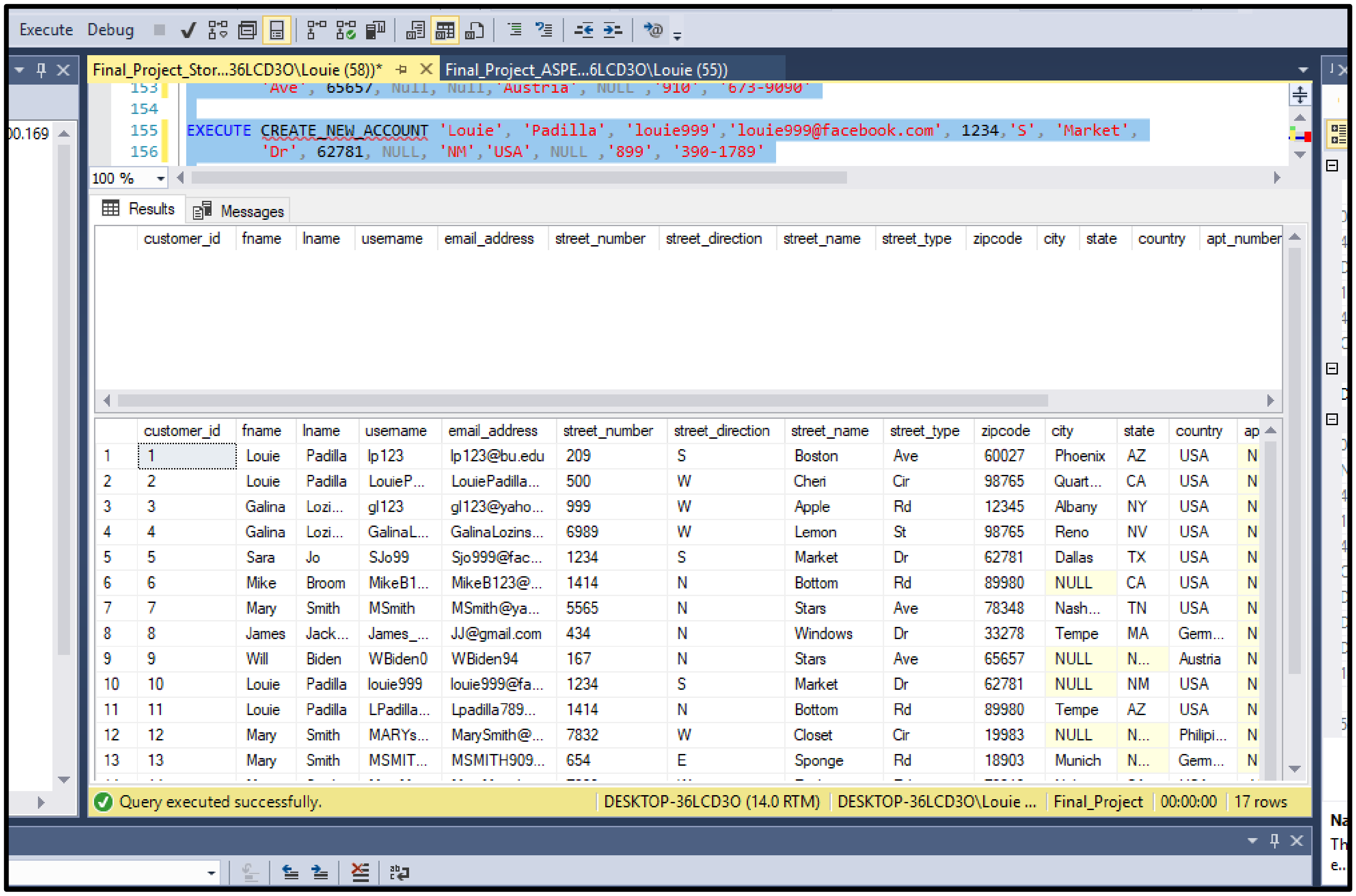

Debug
Final_Project_Stor...36LCD3O\Louie (58))*
Final_Project_ASPE...6LCD3O\Louie (55))
100 %
Messages

```
CREATE PROCEDURE CREATE_NEW_ACCOUNT
@firstname_arg VARCHAR(100),
@lastname_arg VARCHAR(100),
@username_arg VARCHAR(255),
@email_arg VARCHAR(255),
@street_number_arg DECIMAL(12),
@street_direction_arg CHAR(10),
@street_name_arg VARCHAR(100),
@street_type_arg VARCHAR(50),
@zipcode_arg DECIMAL(12),
@city_arg VARCHAR(100),
@state_arg VARCHAR(50),
@country_arg VARCHAR(100),
@apt_number_arg DECIMAL(12),
@phone_areacode_arg VARCHAR(50),
@phone_number_arg VARCHAR(50)
AS
BEGIN
    IF @username_arg NOT IN (SELECT username FROM Customer)
        BEGIN
            INSERT INTO Customer(fname,lname,username,email_address,
            street_number,street_direction,street_name,street_type,zipcode, city,
            state, country,apt_number,phone_areacode,phone_number)
            VALUES(@firstname_arg, @lastname_arg, @username_arg, @email_arg,
                @street_number_arg, @street_direction_arg,  @street_name_arg,
                @street_type_arg, @zipcode_arg, @city_arg, @state_arg, @country_arg,
                @apt_number_arg,@phone_areacode_arg, @phone_number_arg)
        PRINT 'Account Created. Welcome New Customer!'
```

```
Commands completed successfully.
```

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 0 rc

Aspect 3C. You and your facilitator sign up for new accounts on Amazon. Invoke the stored procedure twice to add you and your facilitator as customers.



Final_Project_Stor...36LCD3O\Louie (58))*   Final_Project_ASPE...6LCD3O\Louie (55))

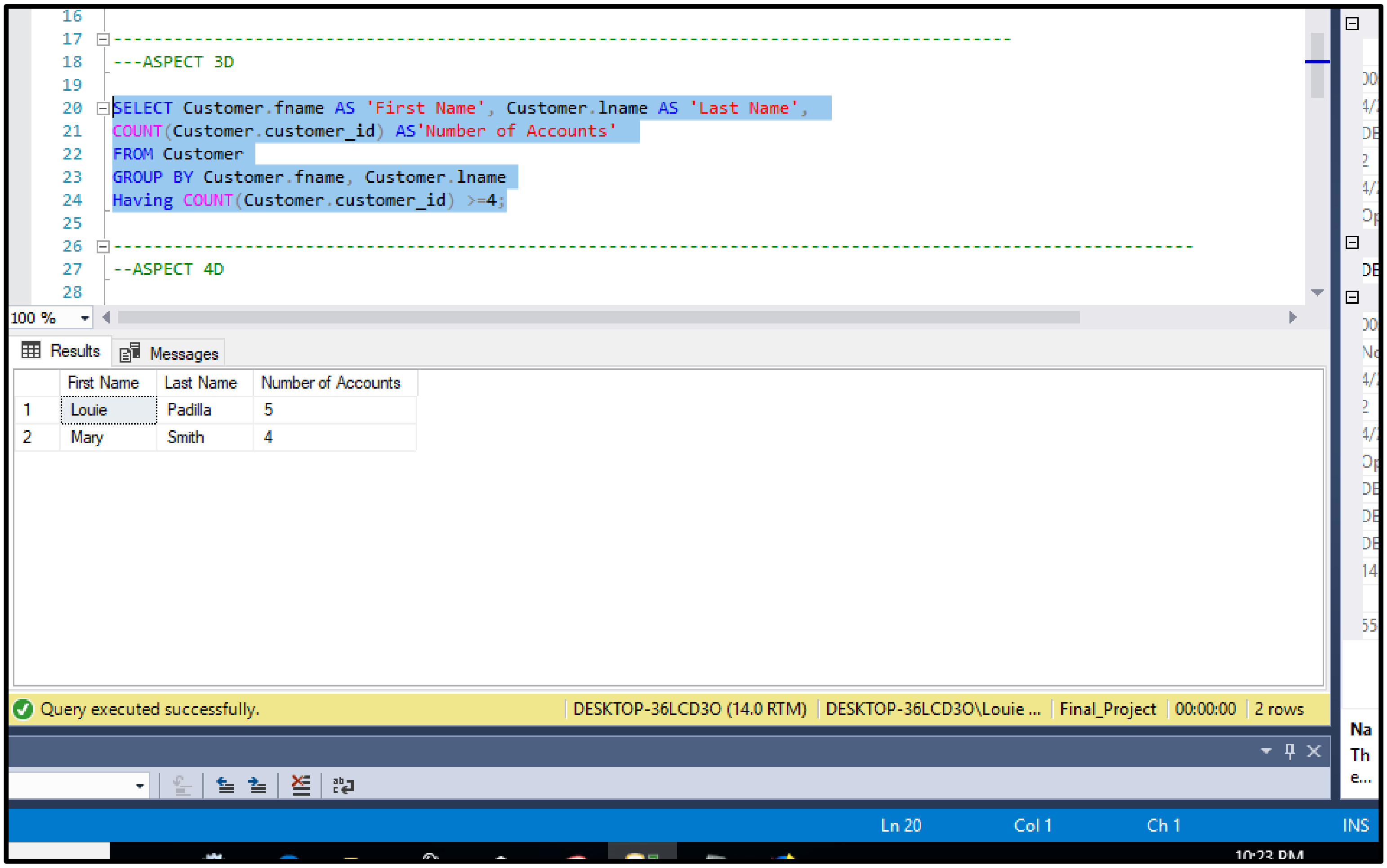
```
        'Ave', 65657, NUll, NUll,'Austria', NULL ,'910', '673-9090'

EXECUTE CREATE_NEW_ACCOUNT 'Louie', 'Padilla', 'louie999','louie999@facebook.com', 1234,'S', 'Market',
        'Dr', 62781, NULL, 'NM','USA', NULL ,'899', '390-1789'
```

	customer_id	fname	lname	username	email_address	street_number	street_direction	street_name	street_type	zipcode	city	state	country	apt_number

	customer_id	fname	lname	username	email_address	street_number	street_direction	street_name	street_type	zipcode	city	state	country	ap
1	1	Louie	Padilla	lp123	lp123@bu.edu	209	S	Boston	Ave	60027	Phoenix	AZ	USA	N
2	2	Louie	Padilla	LouieP...	LouiePadilla...	500	W	Cheri	Cir	98765	Quart...	CA	USA	N
3	3	Galina	Lozi...	gl123	gl123@yaho...	999	W	Apple	Rd	12345	Albany	NY	USA	N
4	4	Galina	Lozi...	GalinaL...	GalinaLozins...	6989	W	Lemon	St	98765	Reno	NV	USA	N
5	5	Sara	Jo	SJo99	Sjo999@fac...	1234	S	Market	Dr	62781	Dallas	TX	USA	N
6	6	Mike	Broom	MikeB1...	MikeB123@...	1414	N	Bottom	Rd	89980	NULL	CA	USA	N
7	7	Mary	Smith	MSmith	MSmith@ya...	5565	N	Stars	Ave	78348	Nash...	TN	USA	N
8	8	James	Jack...	James_...	JJ@gmail.com	434	N	Windows	Dr	33278	Tempe	MA	Germ...	N
9	9	Will	Biden	WBiden0	WBiden94	167	N	Stars	Ave	65657	NULL	N...	Austria	N
10	10	Louie	Padilla	louie999	louie999@fa...	1234	S	Market	Dr	62781	NULL	NM	USA	N
11	11	Louie	Padilla	LPadilla...	Lpadilla789...	1414	N	Bottom	Rd	89980	Tempe	AZ	USA	N
12	12	Mary	Smith	MARYs...	MarySmith@...	7832	W	Closet	Cir	19983	NULL	N...	Philipi...	N
13	13	Mary	Smith	MSMIT...	MSMITH909...	654	E	Sponge	Rd	18903	Munich	N...	Germ...	N

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 17 rows

Aspect 3D. For research purposes, Amazon requests the last names of customers where there are least 4 accounts associated with the last name. Amazon would like to see the actual number of accounts associated with those last names. Develop and execute a single query that provides this information.



```
------------------------------------------------------------------------------------------

---ASPECT 3D

SELECT Customer.fname AS 'First Name', Customer.lname AS 'Last Name',
COUNT(Customer.customer_id) AS'Number of Accounts'
FROM Customer
GROUP BY Customer.fname, Customer.lname
Having COUNT(Customer.customer_id) >=4;

------------------------------------------------------------------------------------------

--ASPECT 4D
```

	First Name	Last Name	Number of Accounts
1	Louie	Padilla	5
2	Mary	Smith	4

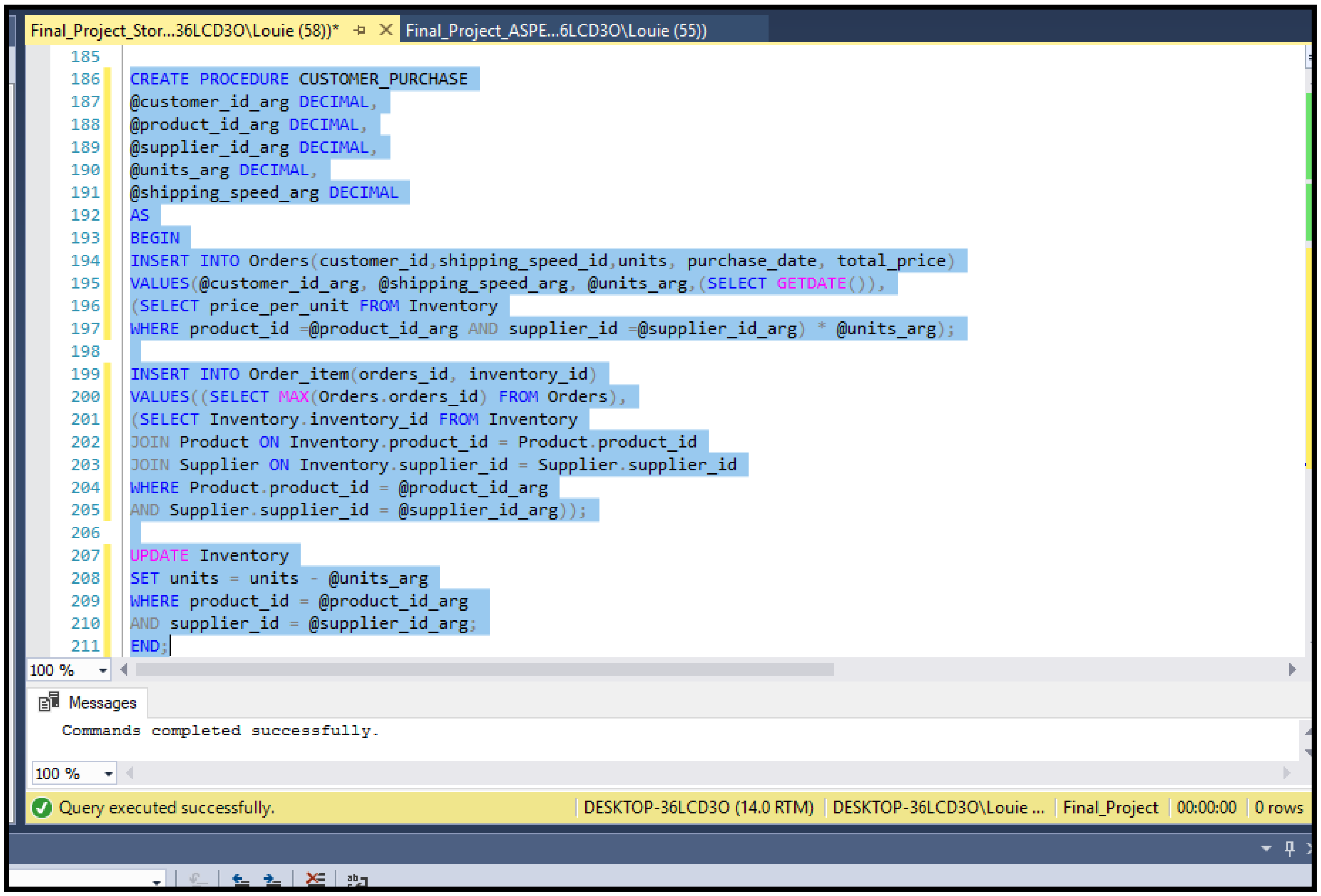
Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 2 rows

Aspect 4

Product Purchase Use Case – This occurs when a customer purchases a product from Amazon that was provided by a seller.

1. *The user logs in to Amazon under their account.*
2. *A customer selects one or more products on Amazon's website. When selecting a product, the customer is actually selecting a particular seller's inventory while doing so, though they might not realize this because the process is seamless on Amazon's website.*
3. *The customer selects a shipping speed (super saver shipping, standard shipping, two-day, one-day) and finalizes their choices.*
4. *Amazon decrements the seller's inventory for the products purchased.*
5. *Amazon creates an order which tracks which customer purchased which products from which sellers.*

Stored Procedure

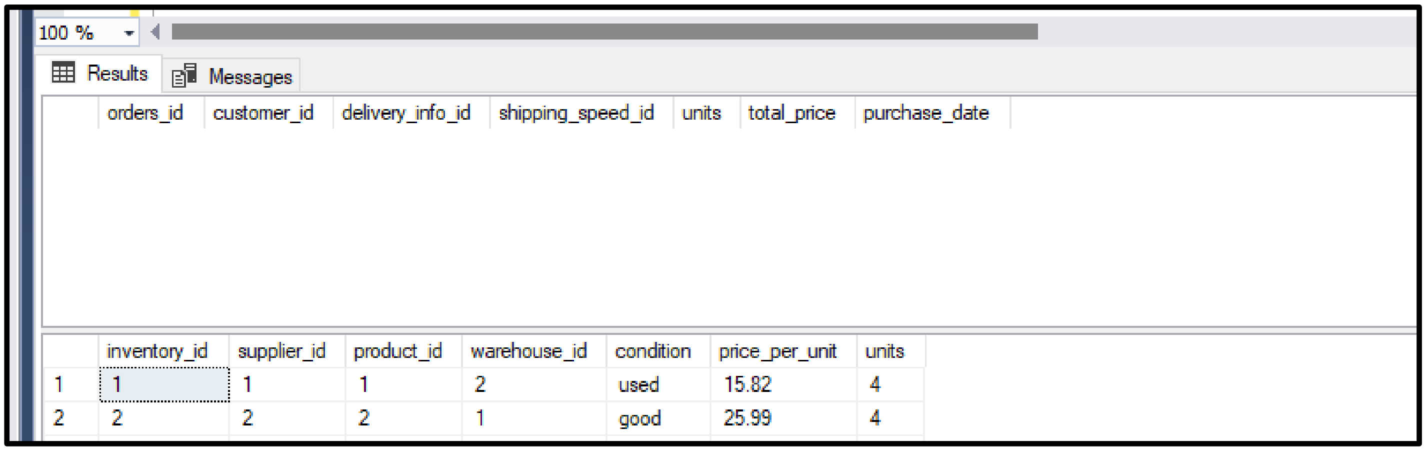

Final_Project_Stor...36LCD3O\Louie (58))*
Final_Project_ASPE...6LCD3O\Louie (55))
100 %
Messages

```
CREATE PROCEDURE CUSTOMER_PURCHASE
@customer_id_arg DECIMAL,
@product_id_arg DECIMAL,
@supplier_id_arg DECIMAL,
@units_arg DECIMAL,
@shipping_speed_arg DECIMAL
AS
BEGIN
INSERT INTO Orders(customer_id,shipping_speed_id,units, purchase_date, total_price)
VALUES(@customer_id_arg, @shipping_speed_arg, @units_arg,(SELECT GETDATE()),
(SELECT price_per_unit FROM Inventory
WHERE product_id =@product_id_arg AND supplier_id =@supplier_id_arg) * @units_arg);

INSERT INTO Order_item(orders_id, inventory_id)
VALUES((SELECT MAX(Orders.orders_id) FROM Orders),
(SELECT Inventory.inventory_id FROM Inventory
JOIN Product ON Inventory.product_id = Product.product_id
JOIN Supplier ON Inventory.supplier_id = Supplier.supplier_id
WHERE Product.product_id = @product_id_arg
AND Supplier.supplier_id = @supplier_id_arg));

UPDATE Inventory
SET units = units - @units_arg
WHERE product_id = @product_id_arg
AND supplier_id = @supplier_id_arg;
END;
```

```
Commands completed successfully.
```

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 0 rows

Aspect 4C. You purchase a self-driving video camera (from Aspect 1), and your facilitator purchases three holographic keyboards. Invoke the stored procedure twice, once for each purchase.

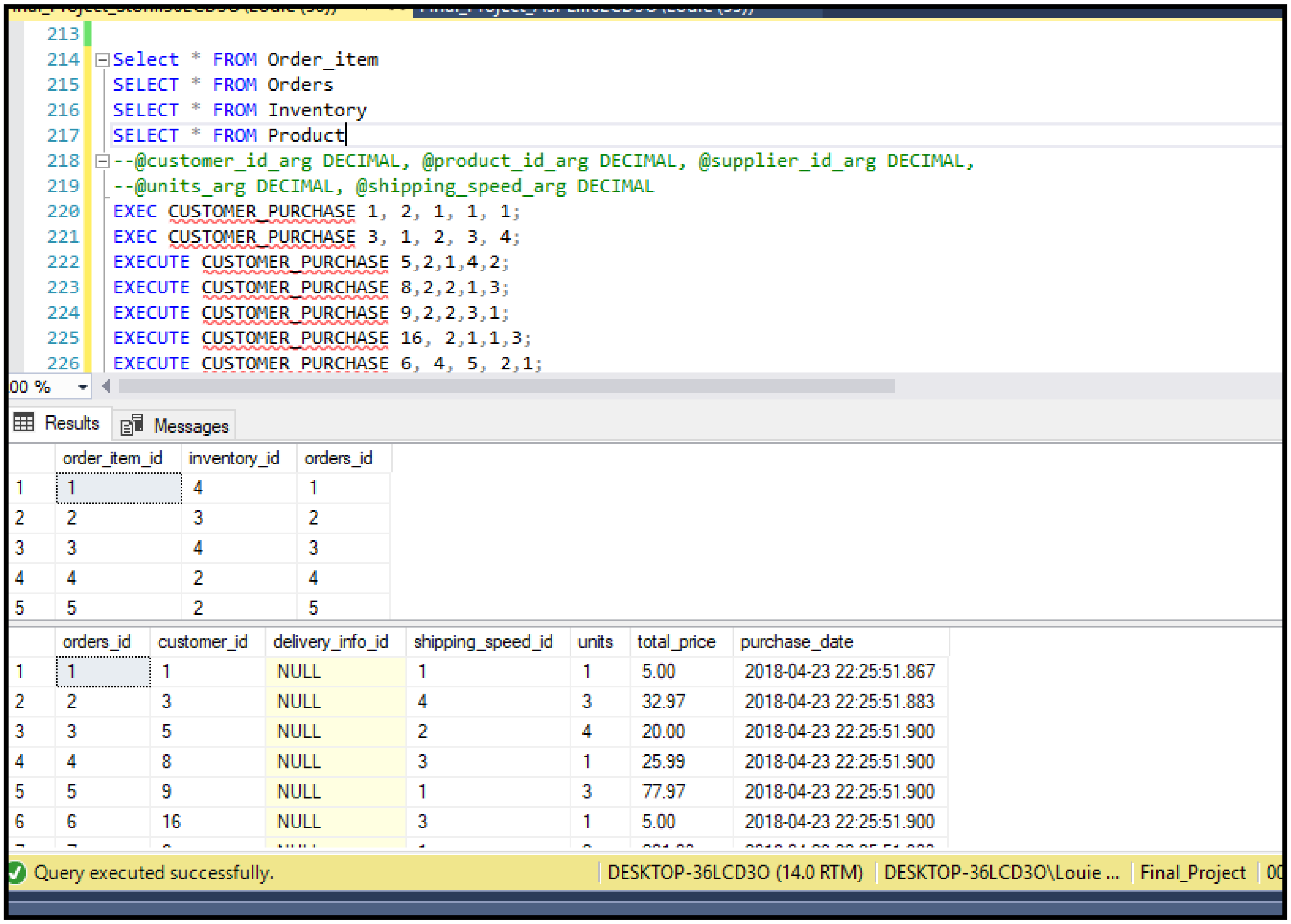


100 %

Results | Messages

orders_id	customer_id	delivery_info_id	shipping_speed_id	units	total_price	purchase_date

	inventory_id	supplier_id	product_id	warehouse_id	condition	price_per_unit	units
1	1	1	1	2	used	15.82	4
2	2	2	2	1	good	25.99	4



```sql

Select * FROM Order_item
SELECT * FROM Orders
SELECT * FROM Inventory
SELECT * FROM Product
--@customer_id_arg DECIMAL, @product_id_arg DECIMAL, @supplier_id_arg DECIMAL,
--@units_arg DECIMAL, @shipping_speed_arg DECIMAL
EXEC CUSTOMER_PURCHASE 1, 2, 1, 1, 1;
EXEC CUSTOMER_PURCHASE 3, 1, 2, 3, 4;
EXECUTE CUSTOMER_PURCHASE 5,2,1,4,2;
EXECUTE CUSTOMER_PURCHASE 8,2,2,1,3;
EXECUTE CUSTOMER_PURCHASE 9,2,2,3,1;
EXECUTE CUSTOMER_PURCHASE 16, 2,1,1,3;
EXECUTE CUSTOMER_PURCHASE 6, 4, 5, 2,1;
```

00 %

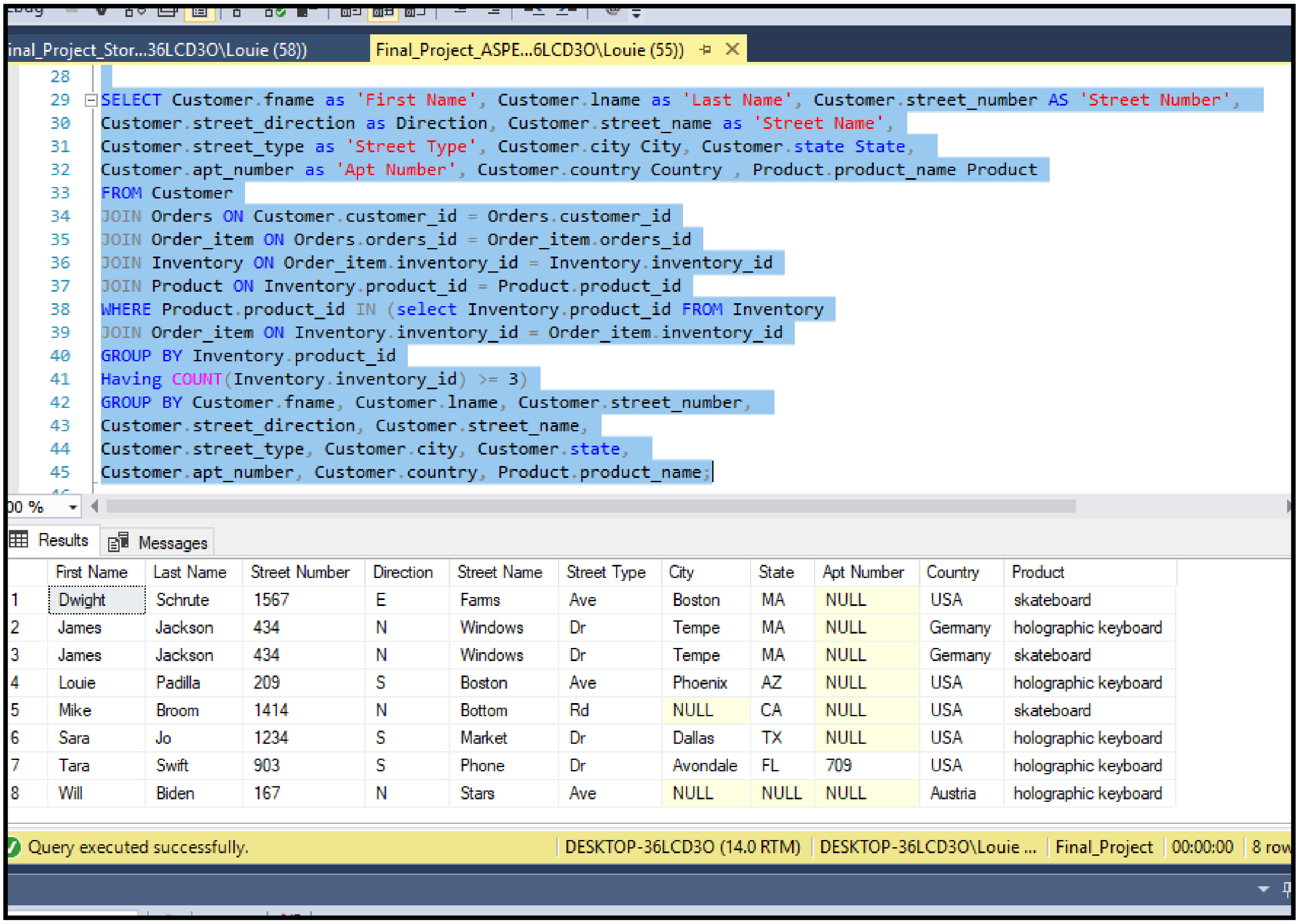
Results | Messages

	order_item_id	inventory_id	orders_id
1	1	4	1
2	2	3	2
3	3	4	3
4	4	2	4
5	5	2	5

	orders_id	customer_id	delivery_info_id	shipping_speed_id	units	total_price	purchase_date
1	1	1	NULL	1	1	5.00	2018-04-23 22:25:51.867
2	2	3	NULL	4	3	32.97	2018-04-23 22:25:51.883
3	3	5	NULL	2	4	20.00	2018-04-23 22:25:51.900
4	4	8	NULL	3	1	25.99	2018-04-23 22:25:51.900
5	5	9	NULL	1	3	77.97	2018-04-23 22:25:51.900
6	6	16	NULL	3	1	5.00	2018-04-23 22:25:51.900

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00

Aspect 4D. The marketing department at Amazon wants to reach out to customers who buy popular products. The department requests the names and addresses of all customers who bought any product that was purchased by at least three different people. Develop and execute a single query that provides this information.


inal_Project_Stor...36LCD3O\Louie (58))
Final_Project_ASPE...6LCD3O\Louie (55))
00 %
Results
Messages

```sql
SELECT Customer.fname as 'First Name', Customer.lname as 'Last Name', Customer.street_number AS 'Street Number',
Customer.street_direction as Direction, Customer.street_name as 'Street Name',
Customer.street_type as 'Street Type', Customer.city City, Customer.state State,
Customer.apt_number as 'Apt Number', Customer.country Country , Product.product_name Product
FROM Customer
JOIN Orders ON Customer.customer_id = Orders.customer_id
JOIN Order_item ON Orders.orders_id = Order_item.orders_id
JOIN Inventory ON Order_item.inventory_id = Inventory.inventory_id
JOIN Product ON Inventory.product_id = Product.product_id
WHERE Product.product_id IN (select Inventory.product_id FROM Inventory
JOIN Order_item ON Inventory.inventory_id = Order_item.inventory_id
GROUP BY Inventory.product_id
Having COUNT(Inventory.inventory_id) >= 3)
GROUP BY Customer.fname, Customer.lname, Customer.street_number,
Customer.street_direction, Customer.street_name,
Customer.street_type, Customer.city, Customer.state,
Customer.apt_number, Customer.country, Product.product_name;
```

	First Name	Last Name	Street Number	Direction	Street Name	Street Type	City	State	Apt Number	Country	Product
1	Dwight	Schrute	1567	E	Farms	Ave	Boston	MA	NULL	USA	skateboard
2	James	Jackson	434	N	Windows	Dr	Tempe	MA	NULL	Germany	holographic keyboard
3	James	Jackson	434	N	Windows	Dr	Tempe	MA	NULL	Germany	skateboard
4	Louie	Padilla	209	S	Boston	Ave	Phoenix	AZ	NULL	USA	holographic keyboard
5	Mike	Broom	1414	N	Bottom	Rd	NULL	CA	NULL	USA	skateboard
6	Sara	Jo	1234	S	Market	Dr	Dallas	TX	NULL	USA	holographic keyboard
7	Tara	Swift	903	S	Phone	Dr	Avondale	FL	709	USA	holographic keyboard
8	Will	Biden	167	N	Stars	Ave	NULL	NULL	NULL	Austria	holographic keyboard

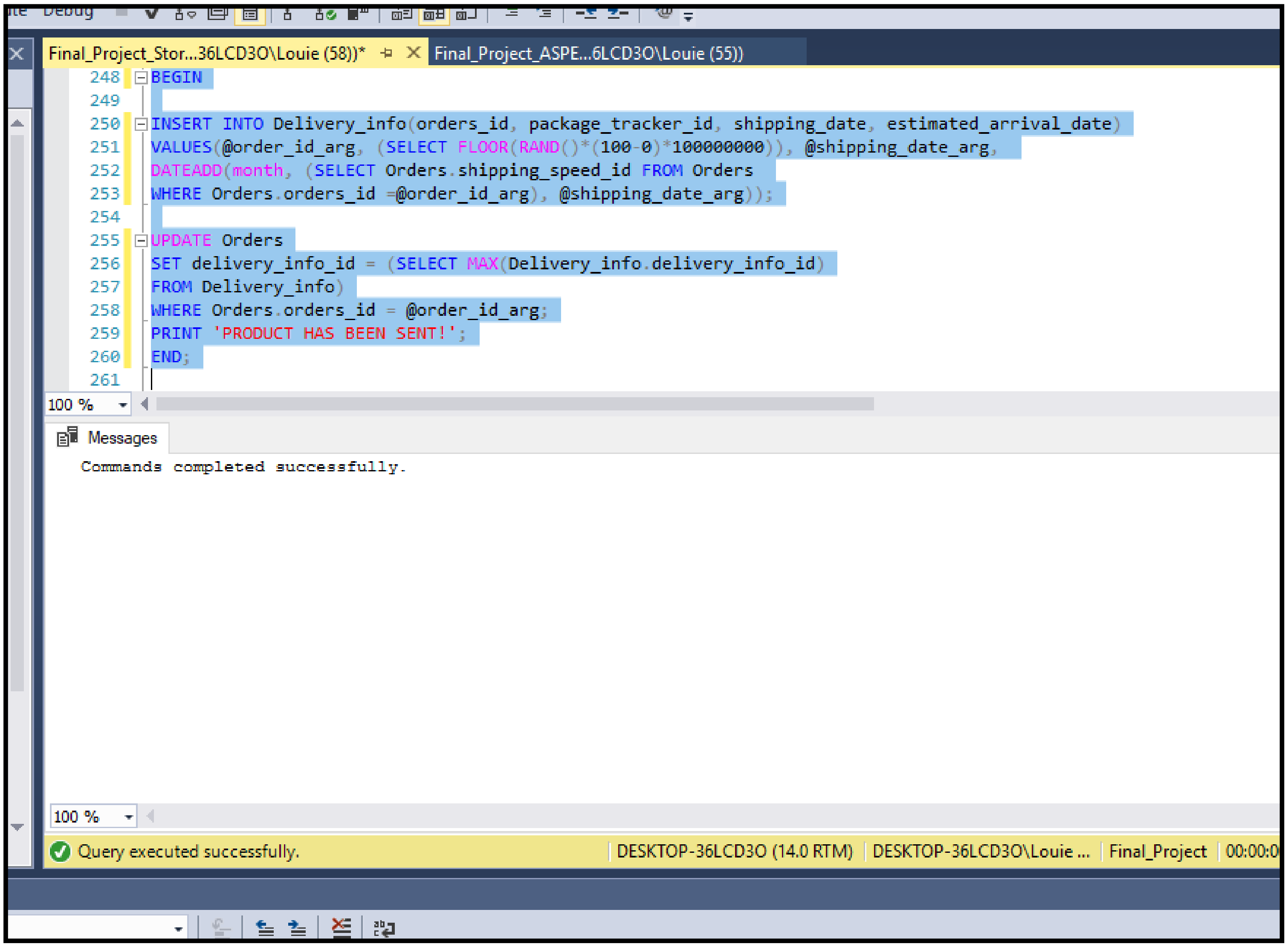
```
Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 8 row
```

Aspect 5

Product Shipment Use Case – This occurs when Amazon ships the products a customer purchased.

1. *Amazon packages up the purchased products and assigns an identifier to package so that it can be tracked.*
2. *Amazon links the package to the customer's order.*
3. *Amazon ships the package to the default address linked to the customer's account.*
4. *Amazon notifies the customer that it has been shipped and provides the customer with the tracking ID.*

Stored Procedure



Final_Project_Stor...36LCD3O\Louie (58))*   Final_Project_ASPE...6LCD3O\Louie (55))

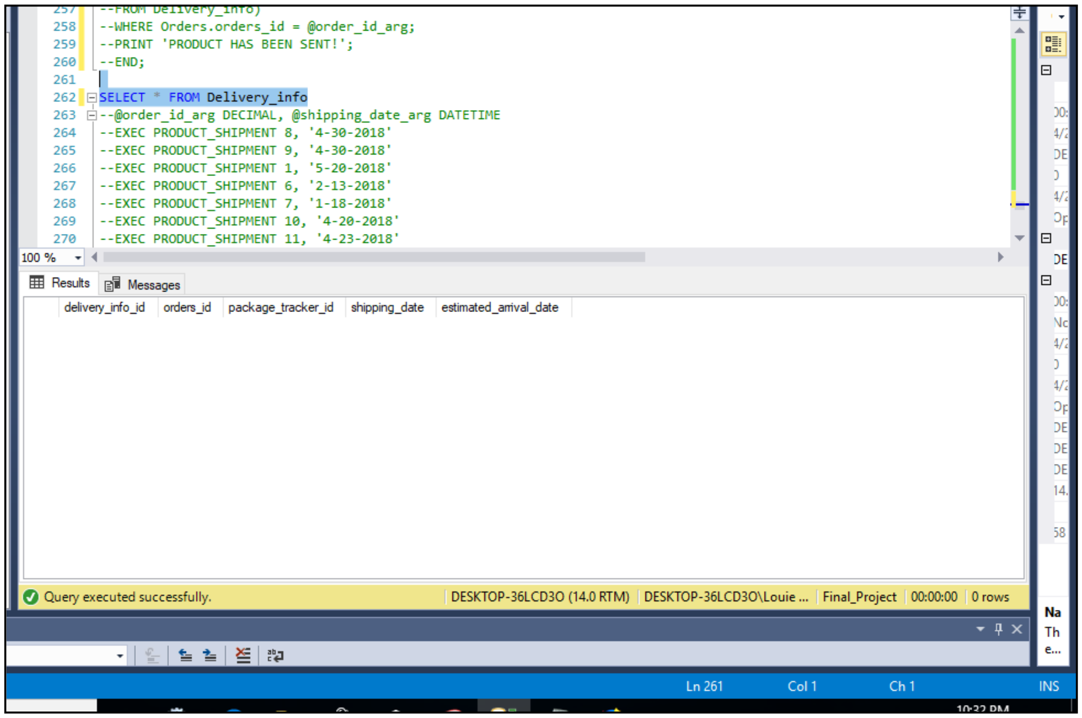
```
BEGIN

INSERT INTO Delivery_info(orders_id, package_tracker_id, shipping_date, estimated_arrival_date)
VALUES(@order_id_arg, (SELECT FLOOR(RAND()*(100-0)*100000000)), @shipping_date_arg,
DATEADD(month, (SELECT Orders.shipping_speed_id FROM Orders
WHERE Orders.orders_id =@order_id_arg), @shipping_date_arg));

UPDATE Orders
SET delivery_info_id = (SELECT MAX(Delivery_info.delivery_info_id)
FROM Delivery_info)
WHERE Orders.orders_id = @order_id_arg;
PRINT 'PRODUCT HAS BEEN SENT!';
END;
```

Messages

```
Commands completed successfully.
```

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:0

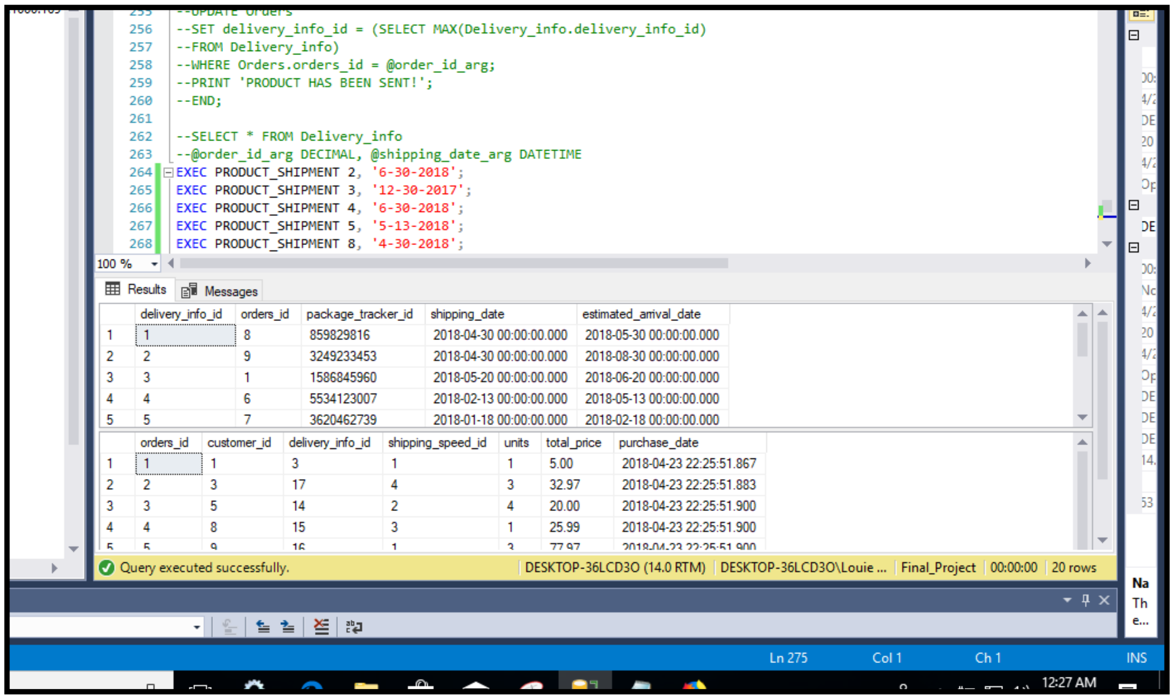
Aspect 5C. Amazon ships the orders listed in Aspect 4, one to you and the other to your facilitator. Invoke the stored procedure twice, once for each order.



```
--FROM Delivery_info)
--WHERE Orders.orders_id = @order_id_arg;
--PRINT 'PRODUCT HAS BEEN SENT!';
--END;

SELECT * FROM Delivery_info
--@order_id_arg DECIMAL, @shipping_date_arg DATETIME
--EXEC PRODUCT_SHIPMENT 8, '4-30-2018'
--EXEC PRODUCT_SHIPMENT 9, '4-30-2018'
--EXEC PRODUCT_SHIPMENT 1, '5-20-2018'
--EXEC PRODUCT_SHIPMENT 6, '2-13-2018'
--EXEC PRODUCT_SHIPMENT 7, '1-18-2018'
--EXEC PRODUCT_SHIPMENT 10, '4-20-2018'
--EXEC PRODUCT_SHIPMENT 11, '4-23-2018'
```

delivery_info_id	orders_id	package_tracker_id	shipping_date	estimated_arrival_date

```
Query executed successfully.   DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 0 rows
Ln 261   Col 1   Ch 1   INS
```



```
--UPDATE Orders
--SET delivery_info_id = (SELECT MAX(Delivery_info.delivery_info_id)
--FROM Delivery_info)
--WHERE Orders.orders_id = @order_id_arg;
--PRINT 'PRODUCT HAS BEEN SENT!';
--END;

--SELECT * FROM Delivery_info
--@order_id_arg DECIMAL, @shipping_date_arg DATETIME
EXEC PRODUCT_SHIPMENT 2, '6-30-2018';
EXEC PRODUCT_SHIPMENT 3, '12-30-2017';
EXEC PRODUCT_SHIPMENT 4, '6-30-2018';
EXEC PRODUCT_SHIPMENT 5, '5-13-2018';
EXEC PRODUCT_SHIPMENT 8, '4-30-2018';
```

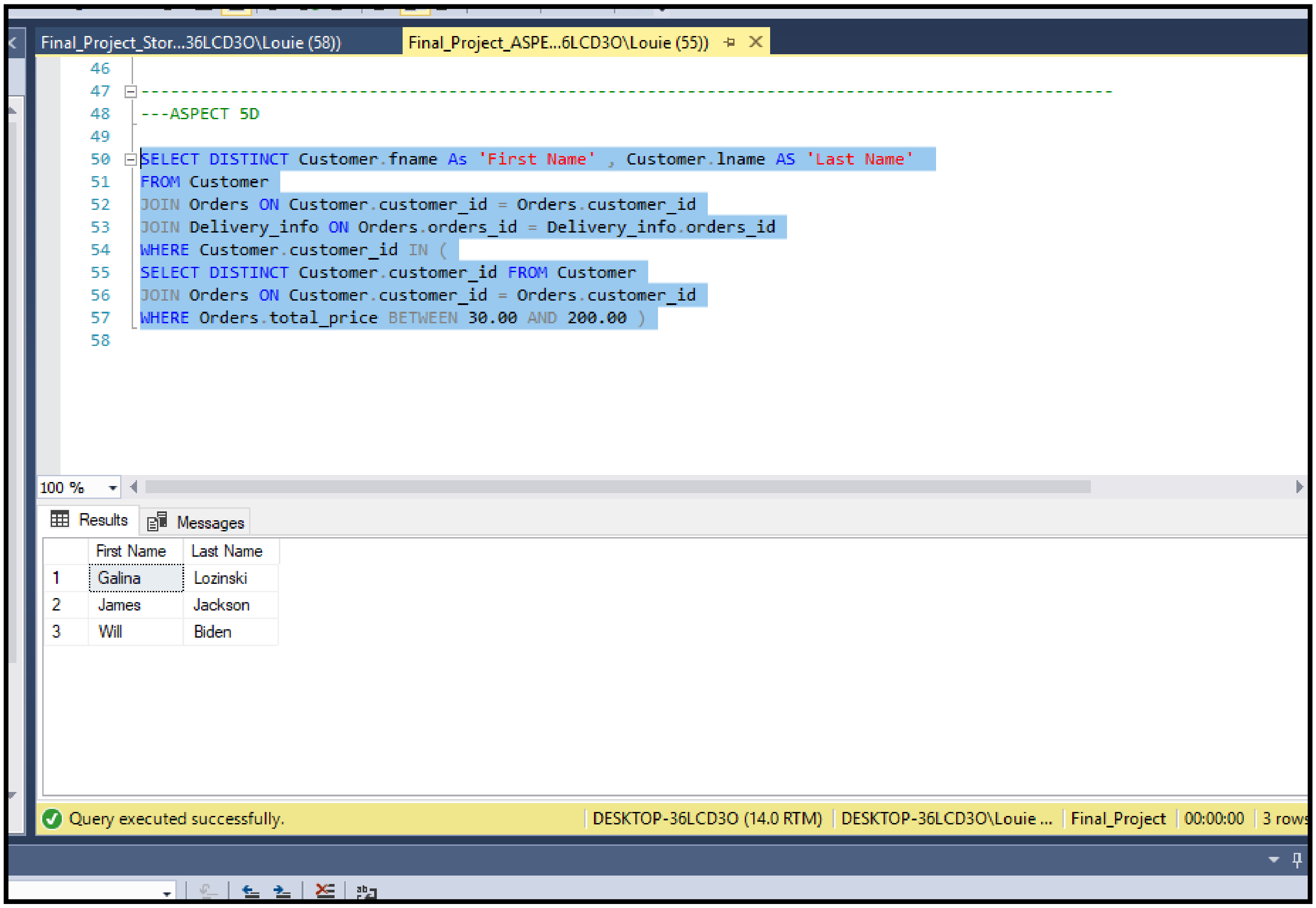
	delivery_info_id	orders_id	package_tracker_id	shipping_date	estimated_arrival_date
1	1	8	859829816	2018-04-30 00:00:00.000	2018-05-30 00:00:00.000
2	2	9	3249233453	2018-04-30 00:00:00.000	2018-08-30 00:00:00.000
3	3	1	1586845960	2018-05-20 00:00:00.000	2018-06-20 00:00:00.000
4	4	6	5534123007	2018-02-13 00:00:00.000	2018-05-13 00:00:00.000
5	5	7	3620462739	2018-01-18 00:00:00.000	2018-02-18 00:00:00.000

	orders_id	customer_id	delivery_info_id	shipping_speed_id	units	total_price	purchase_date
1	1	1	3	1	1	5.00	2018-04-23 22:25:51.867
2	2	3	17	4	3	32.97	2018-04-23 22:25:51.883
3	3	5	14	2	4	20.00	2018-04-23 22:25:51.900
4	4	8	15	3	1	25.99	2018-04-23 22:25:51.900
5	5	9	16	1	3	77.97	2018-04-23 22:25:51.900

```
Query executed successfully.   DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 20 rows
Ln 275   Col 1   Ch 1   INS
```

Aspect 5D. Here you define you own query. Define a request for information for this aspect that is implemented with either aggregation or with a subquery. Then develop and execute a single query that provides this information.

Look for orders shipped that are between $30.00 and $200.00. Name the customer and price.



Final_Project_Stor...36LCD3O\Louie (58))
Final_Project_ASPE...6LCD3O\Louie (55))

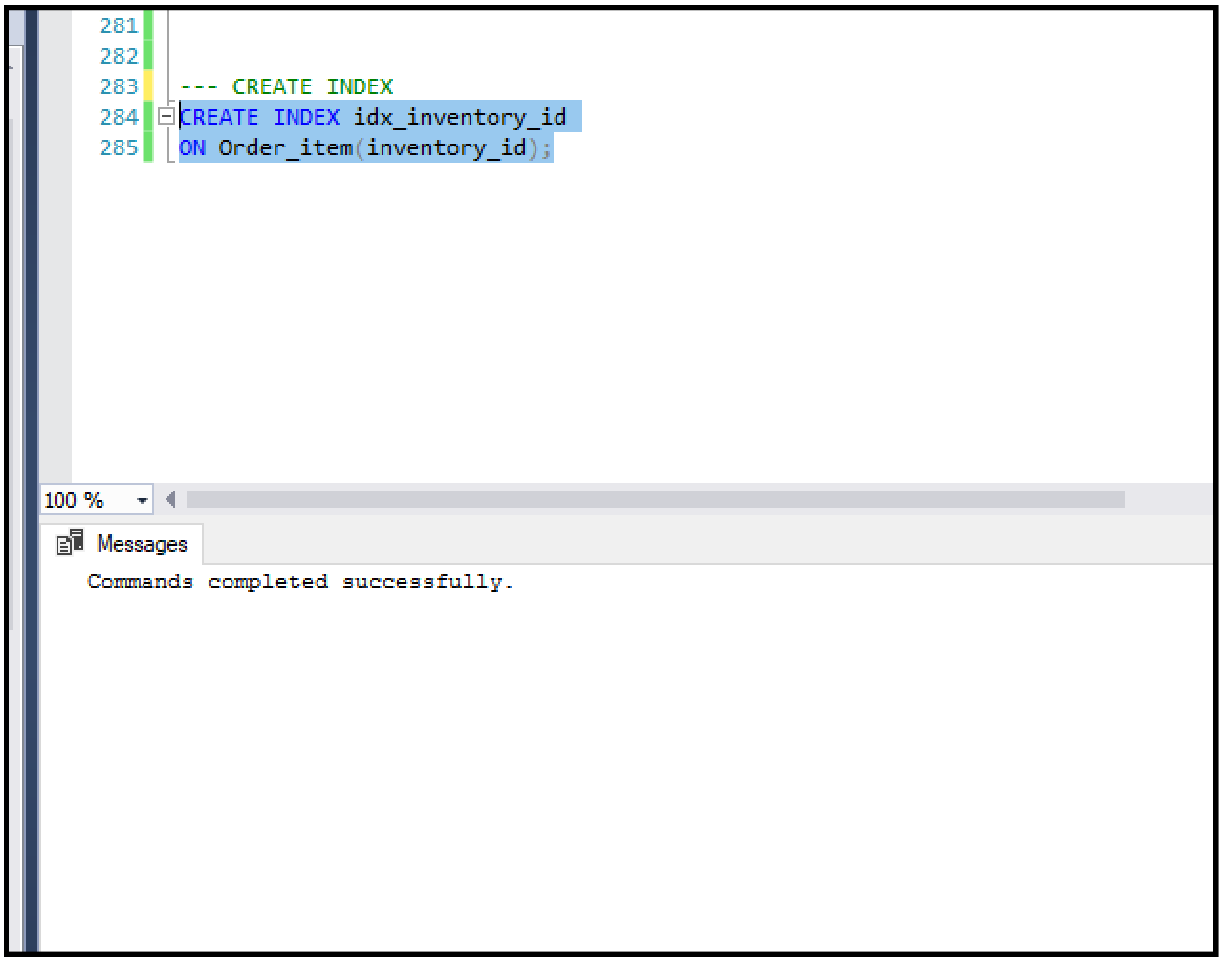
```
---------------------------------------------------------------------------------------------------
---ASPECT 5D

SELECT DISTINCT Customer.fname As 'First Name' , Customer.lname AS 'Last Name'
FROM Customer
JOIN Orders ON Customer.customer_id = Orders.customer_id
JOIN Delivery_info ON Orders.orders_id = Delivery_info.orders_id
WHERE Customer.customer_id IN (
SELECT DISTINCT Customer.customer_id FROM Customer
JOIN Orders ON Customer.customer_id = Orders.customer_id
WHERE Orders.total_price BETWEEN 30.00 AND 200.00 )
```

	First Name	Last Name
1	Galina	Lozinski
2	James	Jackson
3	Will	Biden

Query executed successfully. | DESKTOP-36LCD3O (14.0 RTM) | DESKTOP-36LCD3O\Louie ... | Final_Project | 00:00:00 | 3 rows

Index Creation

The index was created using the inventory ID inside the Order_item entity. This was chosen because there was a potential for large sets of data to be housed inside this table. Having the index on the inventory will allow us to easily find information about the customer who ordered a product and the supplier they chose to buy from. This can be done since the inventory table contains the product ID. This product ID will identify the product that was sold. We can also find out who the Order_item belongs to by joining the Order_item table to Orders then look into the customer ID column. Using the inventory ID as an index will speed up the process of searching for the customer that bought a certain product.



```
--- CREATE INDEX
CREATE INDEX idx_inventory_id
ON Order_item(inventory_id);
```

```
Commands completed successfully.
```

Files

- *Final_Project_ASPECTS*
- *Final_Project_CreateTables_Constraints*
- *Final_Project_Data_Insert*
- *Final_Project_StoredProcedures_Index_Executes*