**Assignment – 4**

* **Check constraint on order details table**

ALTER TABLE tblOrderDetails

ADD CONSTRAINT CHK\_tblOrderDetails\_Unit\_Price CHECK( unit\_price > 0)

ALTER TABLE tblOrderDetails

ADD CONSTRAINT CHK\_tblOrderDetails\_Quantity CHECK( quantity > 0)

ALTER TABLE tblOrderDetails

ADD CONSTRAINT DF\_tblOrderDetails\_Discount DEFAULT 0 FOR discount

Q1) Average Freight. Check before update and insert.

==>

CREATE PROCEDURE usp\_Orders\_AverageFreight

AS

BEGIN

RETURN ( SELECT AVG(freight) FROM tblOrders )

END

ALTER PROCEDURE usp\_Orders\_BeforeInsertUpdate ( @CustomerID VARCHAR(10), @Freight REAL = NULL )

AS

BEGIN

IF NOT EXISTS( SELECT customer\_id FROM tblCustomers WHERE customer\_id = @CustomerID )

BEGIN

RAISERROR(N'Given customer ID %s does not exists', 16, 1, @CustomerID)

RETURN

END

DECLARE @AvgFreight REAL

EXEC usp\_Orders\_AverageFreight @AvgFreight OUTPUT

IF @Freight IS NULL

BEGIN

SELECT @Freight = freight FROM tblOrders WHERE customer\_id = @CustomerID

END

IF @Freight > @AvgFreight

BEGIN

RAISERROR('Given freight exceeds average freight. Insert/Update will be cancelled.',

16, 1)

END

ELSE

BEGIN

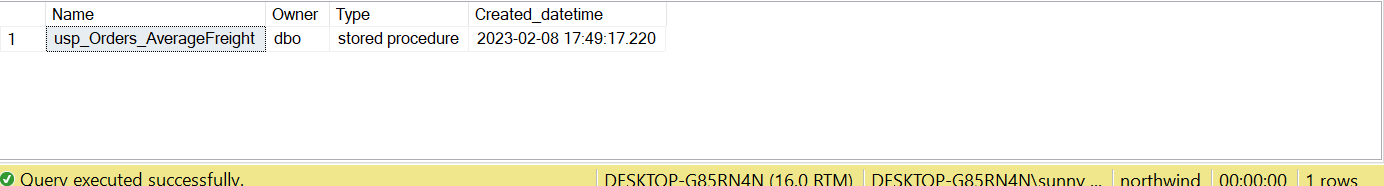
PRINT('Given freight is accepted Business Rule Value.')

END

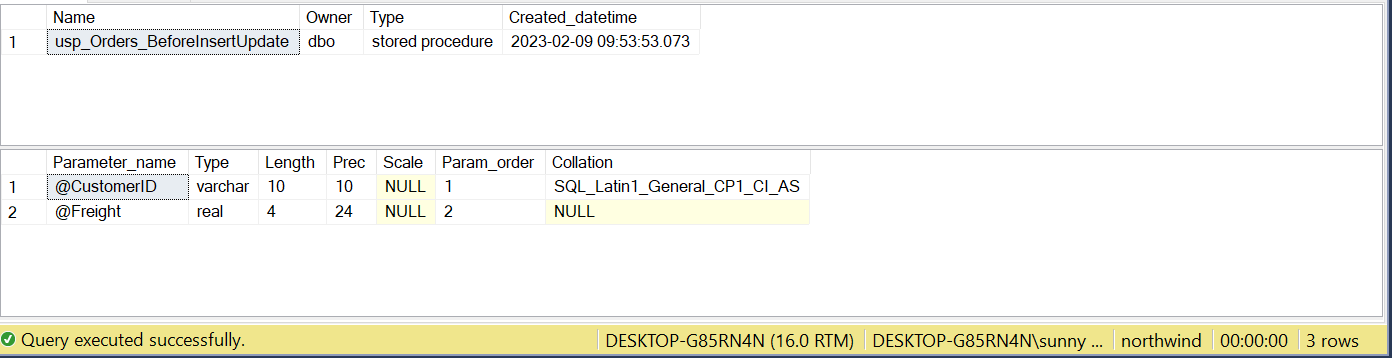
END

Output:

EXEC sp\_help usp\_Orders\_AverageFreight



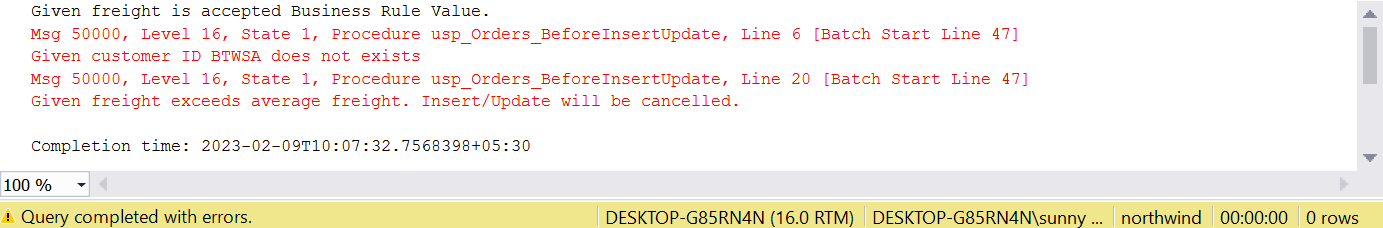
EXECUTE sp\_help usp\_Orders\_BeforeInsertUpdate



EXECUTE usp\_Orders\_BeforeInsertUpdate 'ALFKI'

EXECUTE usp\_Orders\_BeforeInsertUpdate 'BTWSA'

EXECUTE usp\_Orders\_BeforeInsertUpdate 'AROUT', 80



Q2) Write a SQL query to Create Stored procedure in the Northwind database to retrieve Employee Sales by Country.

==>

CREATE PROCEDURE usp\_Employee\_EmployeeSalesByCountry

AS

BEGIN

SELECT emp.employee\_id, emp.first\_name + ' ' + emp.last\_name "Employee Name", ord.ship\_country "Country", COUNT(ord.order\_id) "Total Order",

ROUND(SUM(od.quantity \* od.unit\_price - ( od.quantity \* od.unit\_price \* od.discount)), 2) "Total Sale"

FROM dbo.tblEmployees emp JOIN dbo.tblOrders ord

ON emp.employee\_id = ord.employee\_id

JOIN dbo.tblOrderDetails od

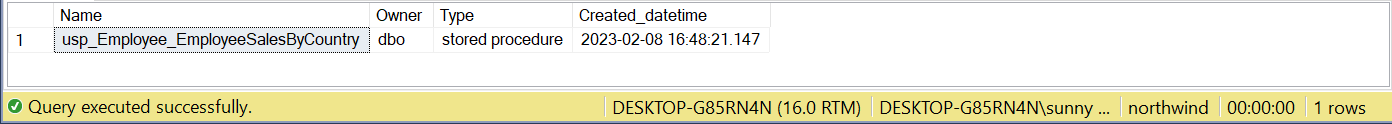
ON ord.order\_id = od.order\_id

GROUP BY emp.employee\_id, emp.first\_name + ' ' + emp.last\_name, ord.ship\_country

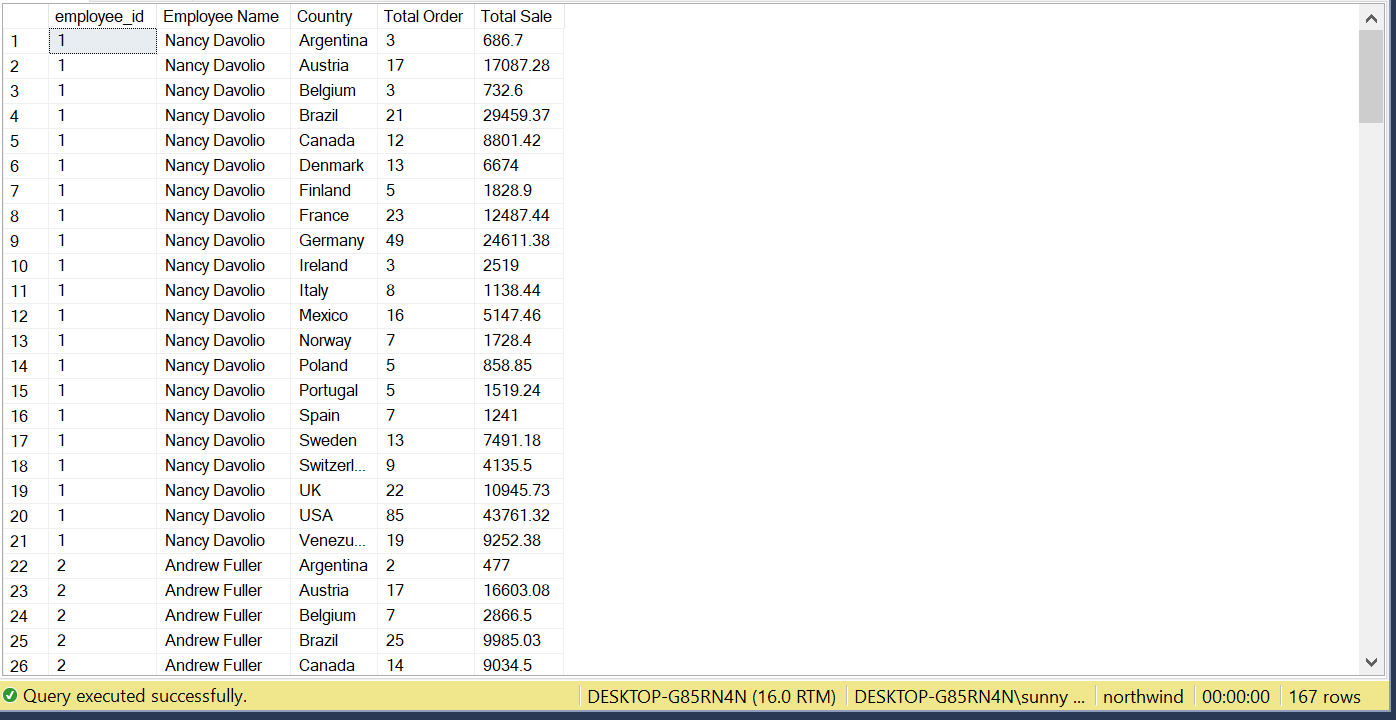
END

Output:

EXEC sp\_help usp\_Employee\_EmployeeSalesByCountry



EXEC usp\_Employee\_EmployeeSalesByCountry



Q3) write a SQL query to Create Stored procedure in the Northwind database to retrieve Sales by Year.

==>

CREATE PROCEDURE usp\_Order\_SalesByYear( @Year SMALLINT )

AS

BEGIN

IF @YEAR NOT IN ( SELECT YEAR(tblOrders.order\_date) FROM tblOrders GROUP BY YEAR(tblOrders.order\_date))

BEGIN

RAISERROR(N'No transaction found for given year: %d',16, 1, @Year)

RETURN

END

SELECT YEAR( ord.order\_date ) "Year", COUNT(ord.order\_id) "Total Order",

ROUND(SUM(od.quantity \* od.unit\_price - ( od.quantity \* od.unit\_price \* od.discount)), 2) "Total Sale"

FROM tblOrders ord JOIN tblOrderDetails od

ON ord.order\_id = od.order\_id

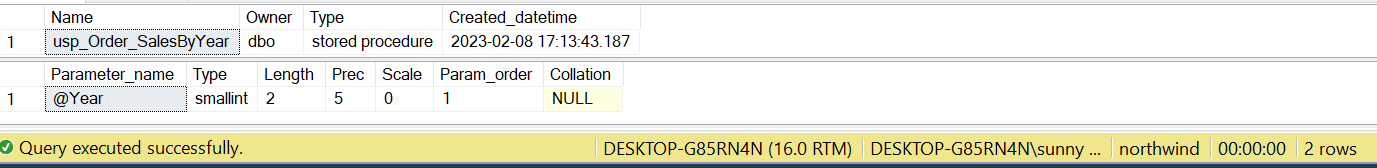
WHERE YEAR(ORD.order\_date) = @Year

GROUP BY YEAR(ord.order\_date)

END

Output:

EXEC sp\_help usp\_Order\_SalesByYear

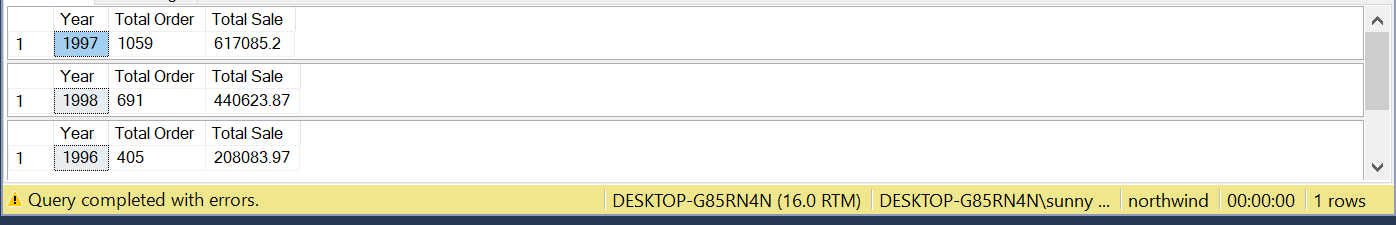


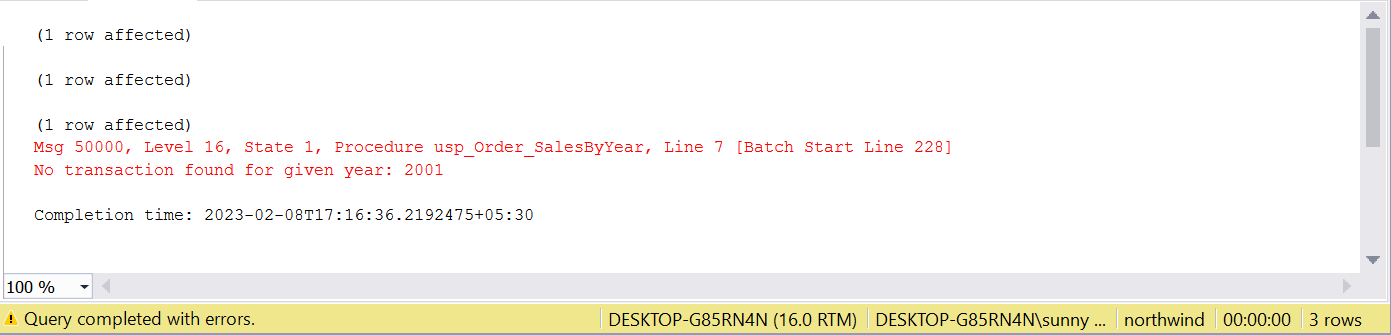
EXEC usp\_Order\_SalesByYear 1997

EXEC usp\_Order\_SalesByYear 1998

EXEC usp\_Order\_SalesByYear 1996

EXEC usp\_Order\_SalesByYear 2001





Q4) Write a SQL query to Create Stored procedure in the Northwind database to retrieve Sales By Category.

==>

CREATE PROCEDURE usp\_Category\_SalesByCategory( @Category VARCHAR(20) = NULL)

AS

BEGIN

IF @Category IS NULL

BEGIN

SELECT category.CategoryID, category.CategoryName,

ROUND(SUM(od.quantity \* od.unit\_price - ( od.quantity \* od.unit\_price \* od.discount)), 2) "Total Sale"

FROM tblOrderDetails od JOIN tblProducts product

ON od.product\_id = product.ProductID

JOIN tblCategories category

ON category.CategoryID = product.CategoryID

GROUP BY category.CategoryID, category.CategoryName

END

ELSE

BEGIN

SELECT category.CategoryID, category.CategoryName,

ROUND(SUM(od.quantity \* od.unit\_price - ( od.quantity \* od.unit\_price \* od.discount)), 2) "Total Sale"

FROM tblOrderDetails od JOIN tblProducts product

ON od.product\_id = product.ProductID

JOIN tblCategories category

ON category.CategoryID = product.CategoryID

WHERE category.CategoryName LIKE '%'+@Category+'%'

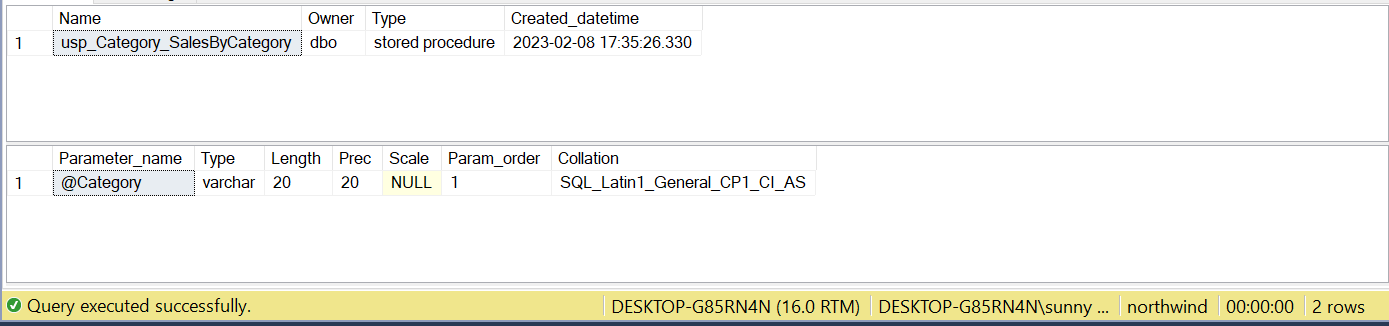
GROUP BY category.CategoryID, category.CategoryName

END

END

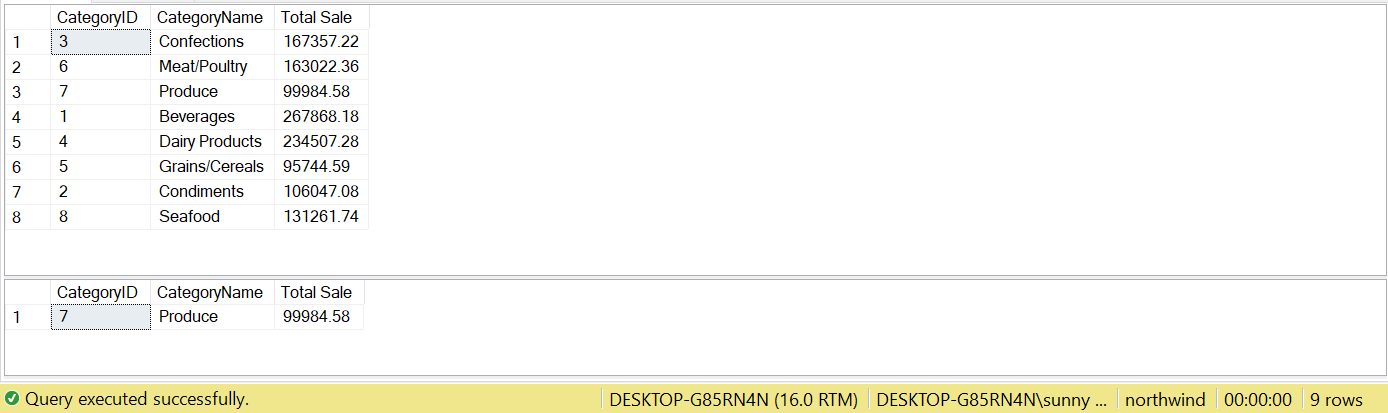
Output:

EXEC sp\_help usp\_Category\_SalesByCategory



EXEC usp\_Category\_SalesByCategory

EXEC usp\_Category\_SalesByCategory 'Produce'



Q5) Write a SQL query to Create Stored procedure in the Northwind database to retrieve Ten Most Expensive Products.

==>

CREATE PROCEDURE usp\_Products\_TopProductsByPrice ( @NumOfRecord INT = 10)

AS

BEGIN

SELECT TOP (@NumOfRecord) \*

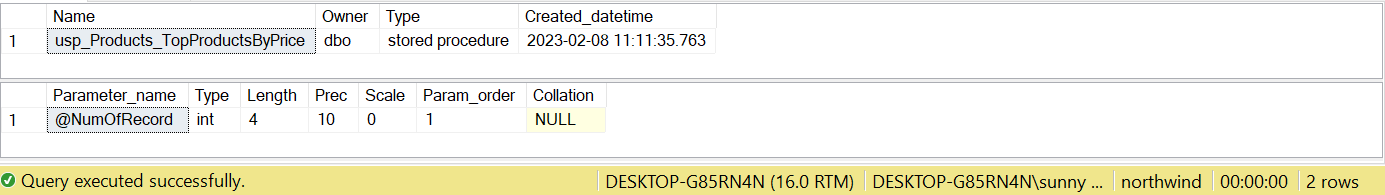
FROM dbo.tblProducts

ORDER BY UnitPrice DESC

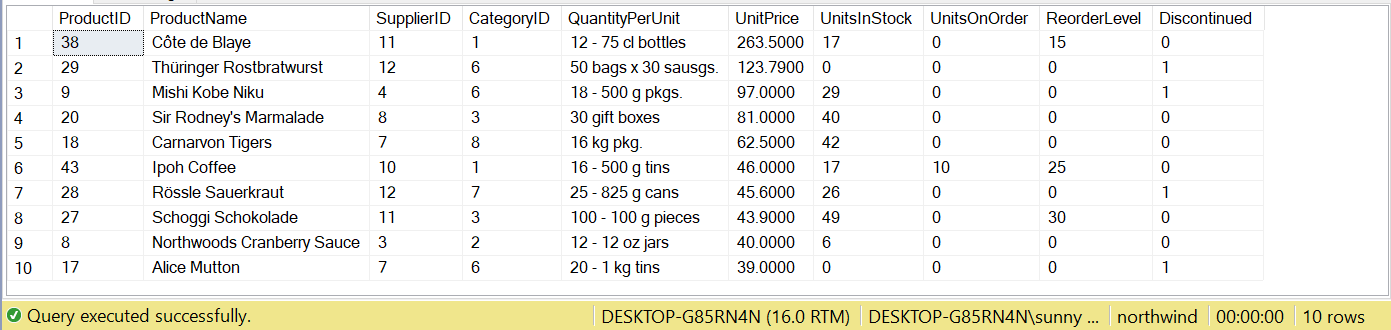
END

Output:

EXEC sp\_help usp\_Products\_TopProductsByPrice



EXEC usp\_Products\_TopProductsByPrice



Q6) Write a SQL query to Create Stored procedure in the Northwind database to insert Customer Order Details.

==>

CREATE PROCEDURE usp\_OrderDetails\_Insert( @OrderID SMALLINT,

@ProductID INT,

@UnitPrice REAL,

@Quantity SMALLINT,

@Discount REAL = 0

)

AS

BEGIN

IF EXISTS(SELECT order\_id FROM tblOrderDetails WHERE order\_id = @OrderID AND product\_id = @ProductID)

BEGIN

RAISERROR(N'Given ORDER ID: %d and product ID: %d already exists', 16, 1, @OrderID, @ProductID)

RETURN

END

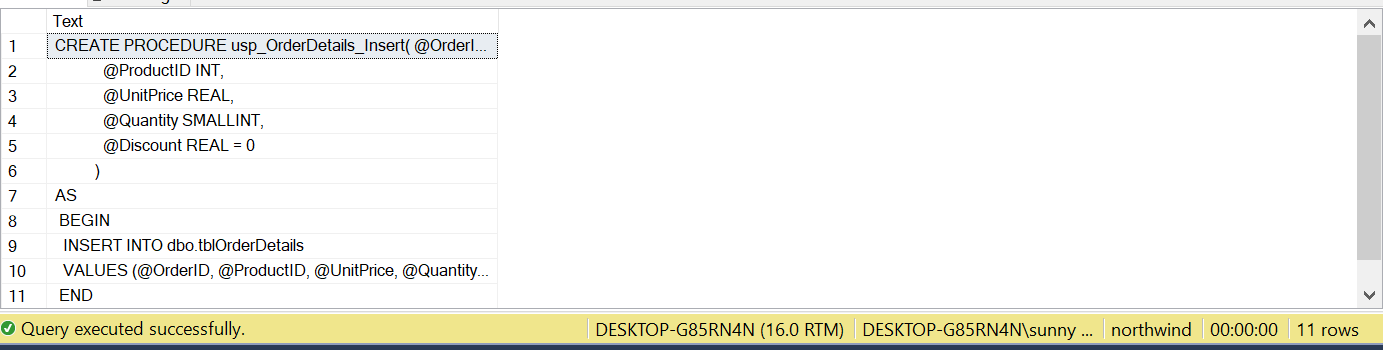
INSERT INTO dbo.tblOrderDetails

VALUES (@OrderID, @ProductID, @UnitPrice, @Quantity, @Discount)

END

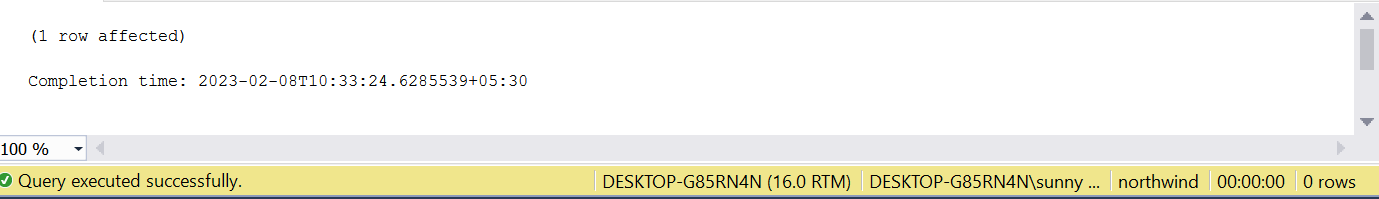
**Output:**

**EXEC sp\_helptext usp\_OrderDetails\_Insert**



Inserting Record in Order Details

EXEC usp\_OrderDetails\_Insert 10268, 10, 20, 30, 0.18



Q7) Write a SQL query to Create Stored procedure in the Northwind database to update Customer Order Details.

==>

CREATE PROCEDURE usp\_OrderDetails\_Update(

@OrderID SMALLINT,

@ProductID INT,

@UnitPrice REAL = NULL,

@Quantity SMALLINT = NULL,

@Discount REAL = NULL

)

AS

BEGIN

IF NOT EXISTS(SELECT order\_id FROM tblOrderDetails WHERE order\_id = @OrderID AND product\_id = @ProductID)

BEGIN

RAISERROR(N'Given ORDER ID: %d and product ID: %d does not exists', 16, 1, @OrderID, @ProductID)

RETURN

END

IF @UnitPrice IS NULL AND @Quantity IS NULL AND @Discount IS NULL

BEGIN

RAISERROR('Nothing to update', 16, 1);

RETURN

END

DECLARE @oldPrice REAL, @oldQuantity SMALLINT, @oldDiscount REAL

SELECT @oldPrice = unit\_price, @oldQuantity = quantity, @oldDiscount = discount

FROM dbo.tblOrderDetails

WHERE order\_id = @OrderID AND product\_id = @ProductID

IF (@UnitPrice IS NULL)

BEGIN

SET @UnitPrice = @oldPrice

END

IF (@Quantity IS NULL)

BEGIN

SET @Quantity = @oldQuantity

END

IF (@Discount IS NULL)

BEGIN

SET @Discount = @oldDiscount

END

UPDATE dbo.tblOrderDetails

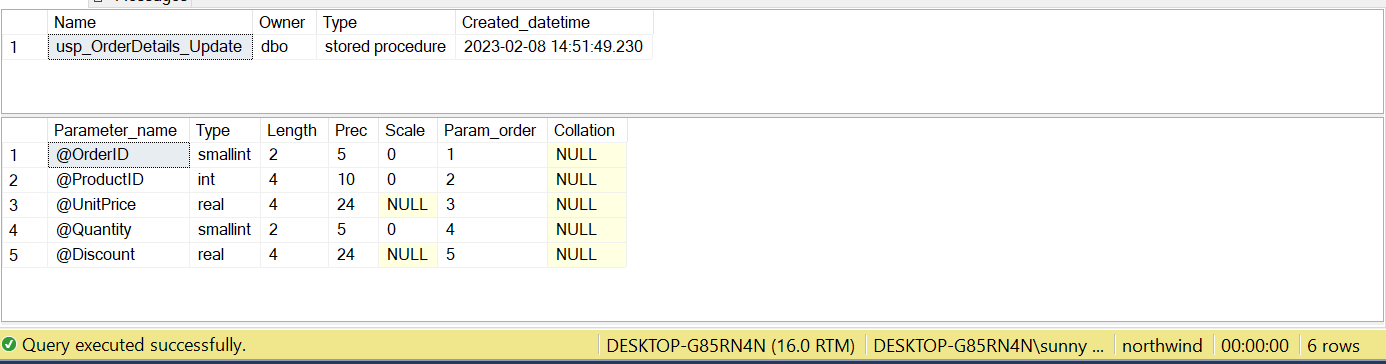
SET unit\_price = @UnitPrice, quantity = @Quantity, discount = @Discount

WHERE order\_id = @OrderID AND product\_id = @ProductID

END

Output:

EXEC sp\_help usp\_OrderDetails\_Update



EXEC usp\_OrderDetails\_Update 10248, 11, 14, 12, 0.13

