**Sql Assignment 4**

1. Create a stored procedure in the Northwind database that will calculate the average

value of Freight for a specified customer.Then, a business rule will be added that will

be triggered before every Update and Insert command in the Orders controller,and

will use the stored procedure to verify that the Freight does not exceed the average

freight. If it does, a message will be displayed and the command will be cancelled.

Sol.

create procedure sp\_ValidateFreight

-- inputted customer

@CustomerID nvarchar(5),

-- returned average freight

@AverageFreight money output

as

begin

select @AverageFreight = AVG(Freight)

from Orders

where CustomerID = @CustomerID

end

go

Create trigger tr\_VerifyFreightForInsert

on Orders

Instead of insert

as

begin

Declare @AvgFreightOfOrders money

Declare @CustID nchar(5)

Declare @Freight money

Select @CustId=CustomerID from inserted

Select @Freight=Freight from inserted

-- execute stored procedure

exec sp\_ValidateFreight @CustID,

@AverageFreight = @AvgFreightOfOrders output

-- check the freight

if @AvgFreightOfOrders is not null

and @AvgFreightOfOrders < @Freight

begin

Raiserror('Invalid data as Freight value exceeds the average freight value',16,1)

return

end

end

Create trigger tr\_VerifyFreightForUpdate

on Orders

Instead of update

as

begin

Declare @AvgFreightOfOrders money

Declare @CustID nchar(5)

Declare @Freight money

Select @CustId=CustomerID from inserted

Select @Freight=Freight from inserted

-- execute stored procedure

exec sp\_ValidateFreight @CustID,

@AverageFreight = @AvgFreightOfOrders output

-- check the freight

if @AvgFreightOfOrders is not null

and @AvgFreightOfOrders < @Freight

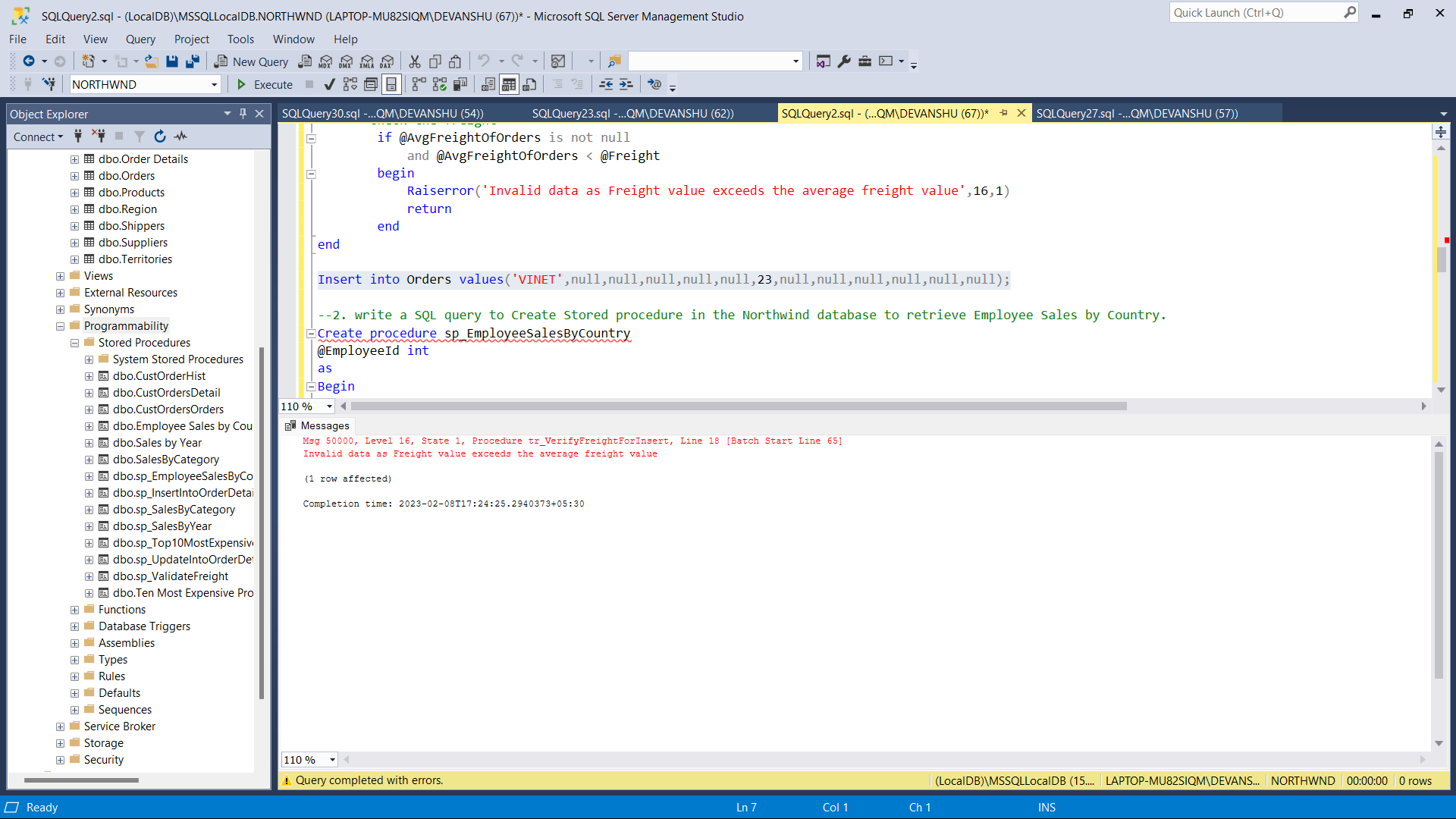
begin

Raiserror('Invalid data as Freight value exceeds the average freight value',16,1)

return

end

end



2. write a SQL query to Create Stored procedure in the Northwind database to retrieve

Employee Sales by Country

Sol.

Create procedure sp\_EmployeeSalesByCountry

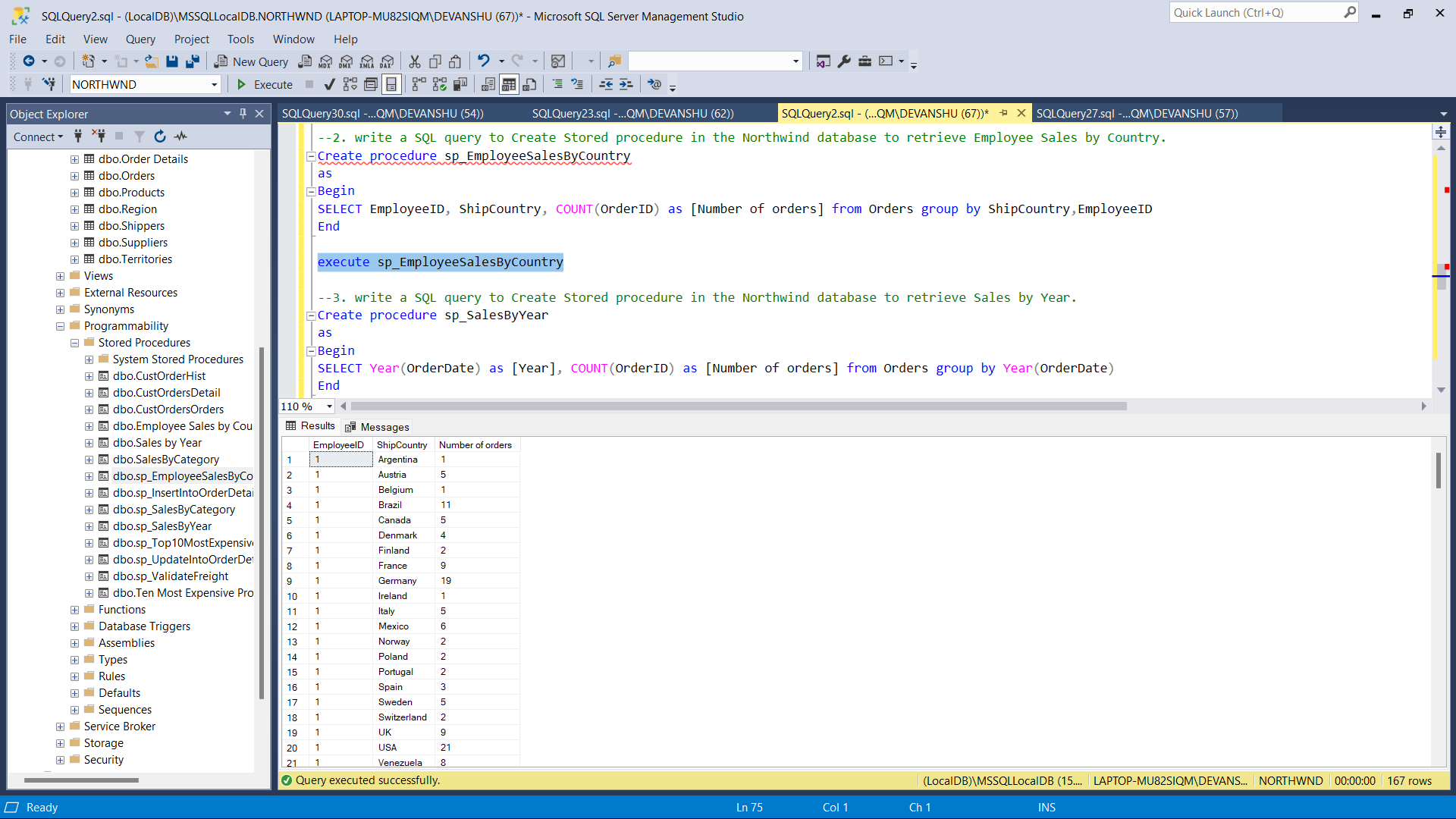
@EmployeeId int

as

Begin

SELECT EmployeeID, ShipCountry, COUNT(OrderID) as [Number of orders] from Orders group by ShipCountry,EmployeeID

End



3. write a SQL query to Create Stored procedure in the Northwind database to retrieve

Sales by Year.

Sol.

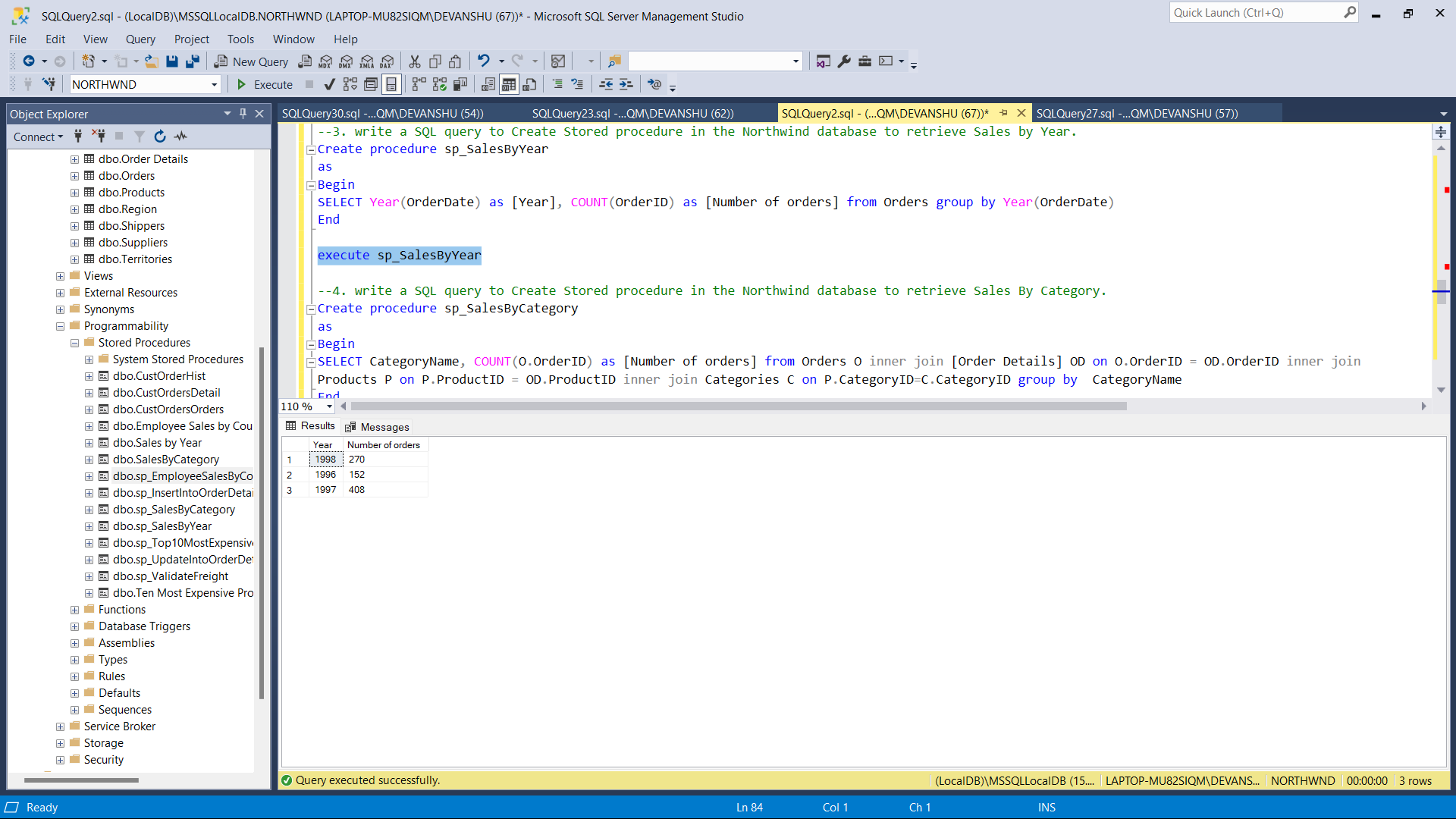
Create procedure sp\_SalesByYear

as

Begin

SELECT Year(OrderDate) as [Year], COUNT(OrderID) as [Number of orders] from Orders group by Year(OrderDate)

End



4. write a SQL query to Create Stored procedure in the Northwind database to retrieve

Sales By Category.

Sol.

Create procedure sp\_SalesByCategory

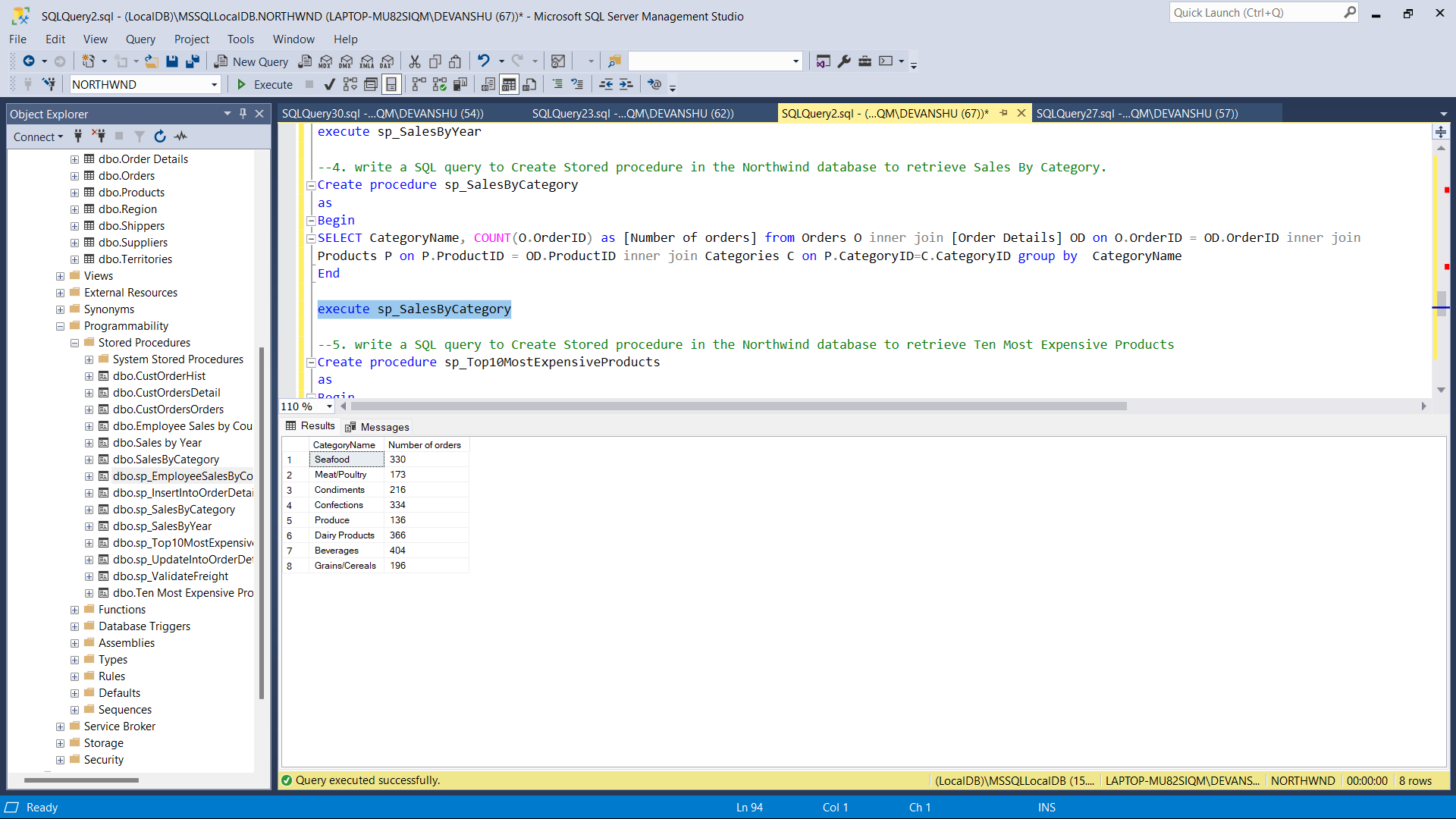
as

Begin

SELECT CategoryName, COUNT(O.OrderID) as [Number of orders] from Orders O inner join [Order Details] OD on O.OrderID = OD.OrderID inner join

Products P on P.ProductID = OD.ProductID inner join Categories C on P.CategoryID=C.CategoryID group by CategoryName

End



5. write a SQL query to Create Stored procedure in the Northwind database to retrieve

Ten Most Expensive Products.

Sol.

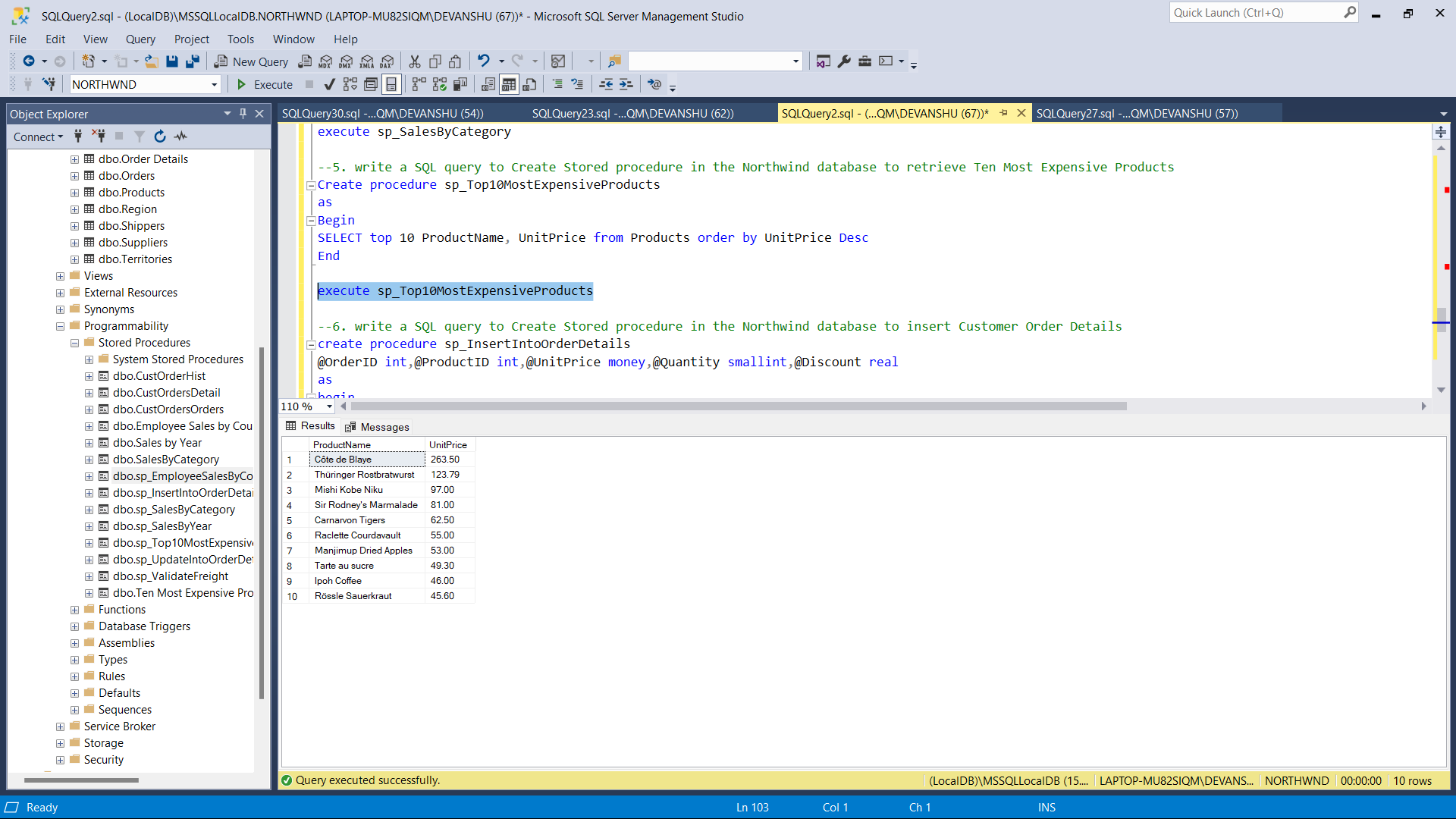
Create procedure sp\_Top10MostExpensiveProducts

as

Begin

SELECT top 10 ProductName, UnitPrice from Products order by UnitPrice Desc

End



6. write a SQL query to Create Stored procedure in the Northwind database to insert

Customer Order Details.

Sol.

create procedure sp\_InsertIntoOrderDetails

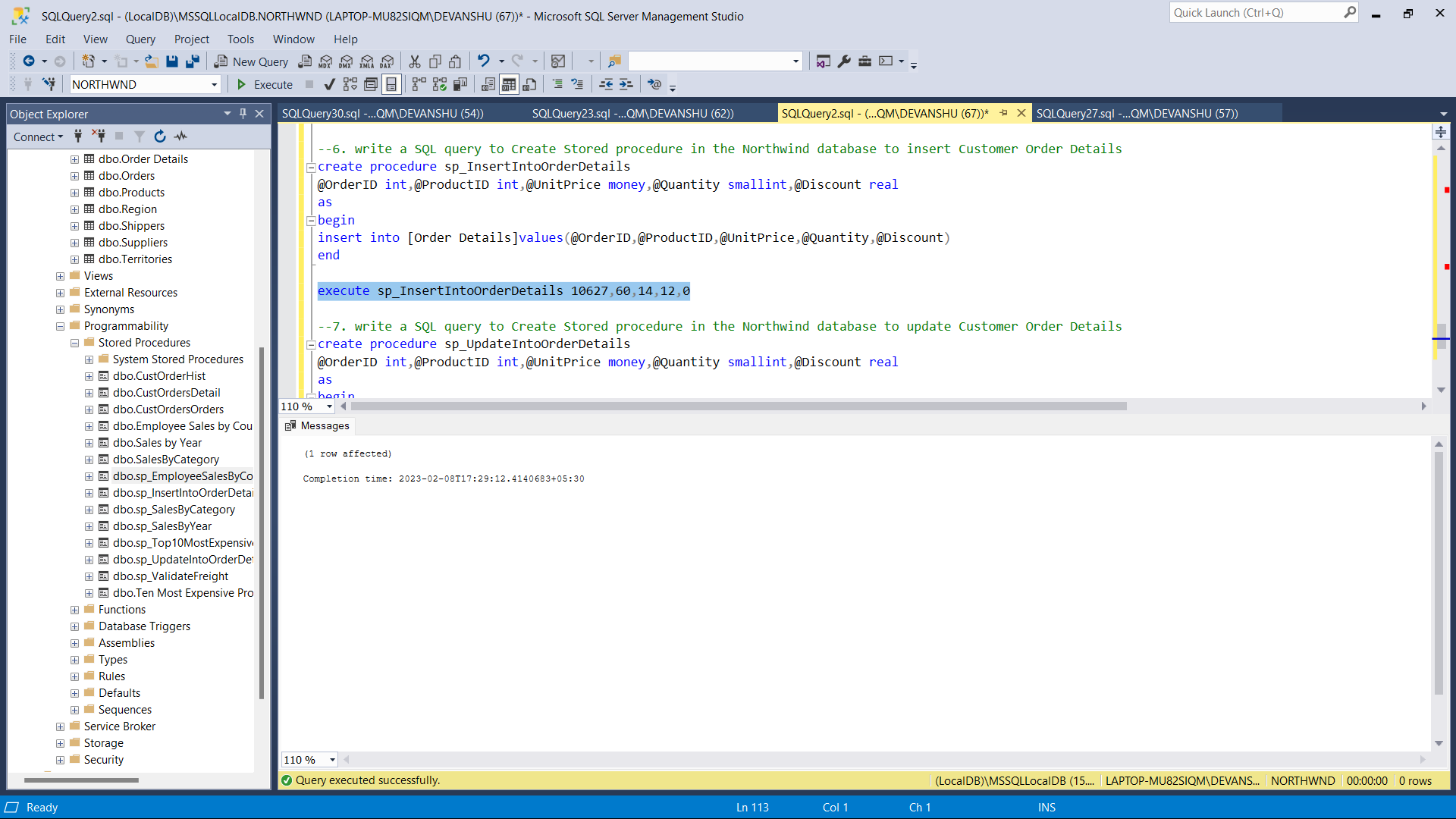
@OrderID int,@ProductID int,@UnitPrice money,@Quantity smallint,@Discount real

as

begin

insert into [Order Details]values(@OrderID,@ProductID,@UnitPrice,@Quantity,@Discount)

end



7. write a SQL query to Create Stored procedure in the Northwind database to update

Customer Order Details.

Sol.

create procedure sp\_UpdateIntoOrderDetails

@OrderID int,@ProductID int,@UnitPrice money,@Quantity smallint,@Discount real

as

begin

update [Order Details] set UnitPrice=@UnitPrice, Quantity=@Quantity, Discount=@Discount

where OrderID=@OrderID and ProductID=@ProductID

end

