AMAZON CLONE DATABASE DESIGN

**Abstract:**

The database is designed for the scenario of Internet Shopping where customers browse goods online and buy the product if it meets the demands. It is similar to E-commerce giant Amazon.

The overall view is that customers orders the products they like online. On the other end, the products are shipped to the customers based on their availability. The products are initially stored in warehouses.

**-Student Name:** Sanjana Kyasrahalli Sreenivasamurthy

**-SUID:** 79454536

**-Instructor:** Ehat Erchanli

-**Class:** Fall 2019

1. **INTRODUCTION**

In today’s fast changing world, its extremely important to be able to satisfy customers needs in most timely manner. If customers wish to see the products online they must be able to do so in most systematic way. And we should be able to provide them with correct information.

To do so the system must be extremely robust to provide clients with information of products they are looking for. One of the main pillar to do so is to design a good database that represents this situation and store data in a very efficient way so that data retrieval and data consistency is maintained.

An online store is a virtual store on the Internet where customers can browse the catalog and select products of interest. The ordered items are then shipped from the warehouse. Keeping this real world scenario as base, I have tried to design database that includes Customers, Products and many other database objects.

This project helps in understanding the creation of a good database and effort that goes into building a normalized clean database.

1. **DESIGN CONSIDERATIONS**

To generate the Amazon Clone database system with effective functionality, we need to be careful while selecting the appropriate attributes for our system. The following are the basic attributes identified as a stepping stone for crafting the system:

**Customers:**

CustomerId | Name | Email| Phone | Address Order information | Date ordered | Reviews

**Products:**

ProductId | ProductName | Price | CustomerRatings

**Suppliers:**

SupplierId | Name | Address | Phone | Email

Products Information

**Orders**

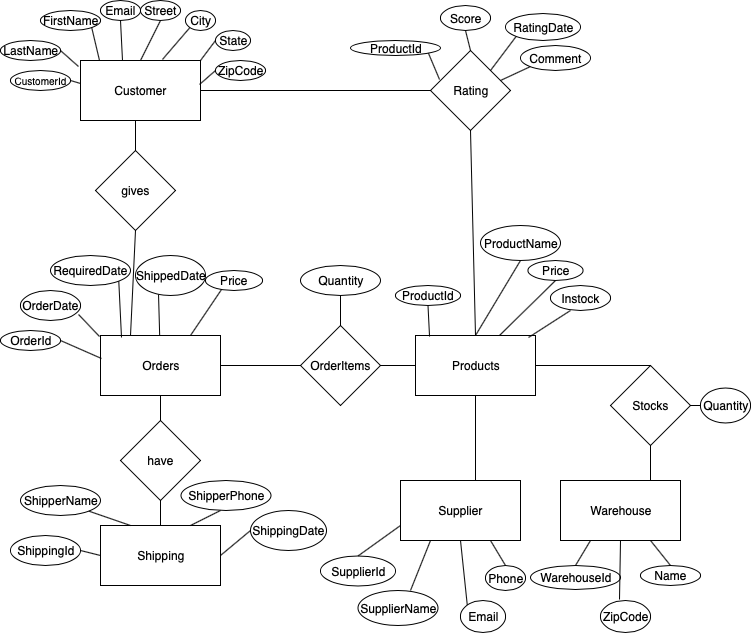
OrderNumber | Status | OrderDate | OrderItems TotalPrice | Shipping Service | Shipping Address

Shipping Information

**Warehouse:**

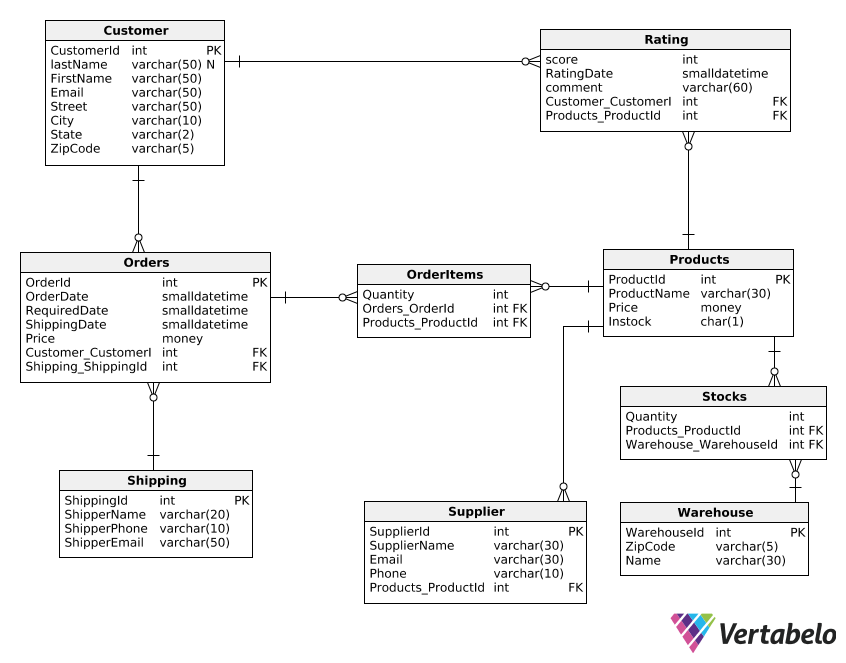
WarehouseId | Address | StoredProducts

**ER diagram**



1. **SCHEMA**

After normalizing the database to 3rd normal form, the schema of the database looks like:

****

1. **IMPLEMENTATION**

/\*Code\*/

use master

go

if db\_id('AmazonClone') is not null

drop database AmazonClone;

go

/\*create database named AmazonClone\*/

create database AmazonClone;

go

use AmazonClone

go

/\*creating tables for AmazonClone database\*/

create table Customer(

CustomerId int primary key identity not null,

LastName varchar(50),

FirstName varchar(50) not null,

Email varchar(50) not null,

Street varchar(50),

City varchar(10),

State varchar(2),

ZipCode varchar(5)

);

create table Shipping(

ShippingId int primary key identity not null,

ShipperName varchar(20),

ShipperPhone varchar(10),

ShipperEmail varchar(50)

);

create table Orders(

OrderId int primary key identity not null,

OrderDate smalldatetime not null,

RequiredDate smalldatetime,

ShippingDate smalldatetime not null,

Price money not null check(Price>0),

CustomerId int not null references Customer(CustomerId),

ShippingId int not null references Shipping(ShippingId)

);

create table Supplier(

SupplierId int not null primary key identity,

SupplierName varchar(30),

Email varchar(30),

Phone varchar(10)

);

create table Products(

ProductId int primary key identity not null,

ProductName varchar(30) not null,

Price money,

Instock char(1),

SupplierId int not null references Supplier(SupplierId)

);

create table OrderItems(

OrderId int not null references Orders(OrderId),

ProductId int not null references Products(ProductId),

Quantity int

primary key(OrderId,ProductId)

);

create table Rating(

ProductId int not null references Products(ProductId),

CustomerId int not null references Customer(CustomerId),

score int,

RatingDate smalldatetime,

comment varchar(60)

);

create table Warehouse(

WarehouseId int not null primary key identity,

ZipCode varchar(5),

Name varchar(30)

);

create table Stocks(

WarehouseId int not null references Warehouse(WarehouseId),

ProductId int not null references Products(ProductId),

Quantity int not null

);

create table OrdersArchive(

    OrderId int primary key identity not null,

    OrderDate smalldatetime  not null,

    RequiredDate smalldatetime,

    ShippingDate smalldatetime not null,

    Price money not null check(Price>0),

    CustomerId int not null,

    ShippingId int not null

);

/\*Inserting values into tables of AmazonClone database\*/

insert into Customer values ('Lee','Christine','lee@gmail.com','Havard Pl','Syracuse','NY','13210');

insert into Customer values ('Lee','Vicki','Vickilee@gmail.com','Havard Pl','New York','NY','13211');

insert into Customer values ('Anna','Grace','annagrace@gmail.com','Columbus av','Auburn','AL','14154');

insert into Customer values ('Carol','Sue','craolsue@gmail.com','Berkley dr','Phoenix','AR','14256');

insert into Customer values ('Lou','Tressa','tressalou@gmail.com','Comstock av','Prescott','AR','13210');

insert into Customer values ('Mark','John','markjohn@gmail.com','Colvin Street','Brooklyn','NY','13212');

insert into Customer values ('Jack','Bobby','bobbyjack@gmail.com','Burnet av','Buffalo','NY','13215');

insert into Customer values ('Allen','Ford','fordallen@gmail.com','Euclid av','Beacon','NY','13216');

insert into Customer values ('Brock','Will','willbrock@gmail.com','James street','Charlotte','NC','67210');

insert into Shipping values ('Brenda Enterprise','4567893456','brendaent@gmail.com');

insert into Shipping values ('Vishal Enterprise','2564789105','vishalent@gmail.com');

insert into Shipping values ('Hut Enterprise','2425347495','hutent@gmail.com');

insert into Shipping values ('Vinayaka Enterprise','1453623789','vinayakaent@gmail.com');

insert into Shipping values ('Ganesh Enterprise','3451789021','ganeshent@gmail.com');

insert into Shipping values ('Vignesh Enterprise','3564729101','vigneshent@gmail.com');

insert into Shipping values ('Srinivas Enterprise','3452617891','srinivasent@gmail.com');

insert into Shipping values ('zeroone','8972615345','zeroone@gmail.com');

insert into Shipping values ('Namana Enterprise','6782435617','namanaent@gmail.com');

insert into Shipping values ('Shiva Enterprise','8930236781','shivaent@gmail.com');

insert into Orders values ('2017-04-8','2017-4-15','2017-4-12',240.0,1,3);

insert into Orders values ('2017-04-9','2017-4-13','2017-4-11',500.0,1,4);

insert into Orders values ('2017-04-15','2017-4-20','2017-4-16',643.0,2,1);

insert into Orders values ('2017-05-12','2017-05-18','2017-05-17',890.0,3,1);

insert into Orders values ('2017-06-08','2017-6-13','2017-4-10',1000.0,4,2);

insert into Orders values ('2017-06-23','2017-06-25','2017-6-23',340.0,5,5);

insert into Orders values ('2017-07-9','2017-7-15','2017-7-11',456.0,6,6);

insert into Orders values ('2017-01-12','2017-1-20','2017-1-15',344.0,7,6);

insert into Orders values ('2017-01-15','2017-1-25','2017-4-18',235.0,8,7);

insert into Orders values ('2017-02-14','2017-2-20','2017-4-15',678.0,9,8);

insert into Supplier values('Namana','namana@gmail.com','3456789865');

insert into Supplier values('Mendoz','mendoz@gmail.com','4561243456');

insert into Supplier values('allen','allen@gmail.com','4561236783');

insert into Supplier values('Brady','brady@gmail.com','6732451678');

insert into Supplier values('Channing','channing@gmail.com','8935671234');

insert into Supplier values('Carson','crason@gmail.com','9075677845');

insert into Supplier values('Cohen','cohen@gmail.com','6782341567');

insert into Supplier values('Axton','axton@gmail.com','9845678934');

insert into Supplier values('Barnes','barnes@gmail.com','7834561789');

insert into Supplier values('Hepburn','hepburn@gmail.com','9034568734');

insert into Products values('Ann Bath Towel',150,1,1);

insert into Products values('Vanilla Bean lotion',240,1,2);

insert into Products values('Champagne toast lotion',567,1,3);

insert into Products values('Winter candy wash',567,1,4);

insert into Products values('gingham body wah',345,1,5);

insert into Products values('Twisted mint wash',567,1,6);

insert into Products values('Cherry blossom lotion',432,1,6);

insert into Products values('Ann mixer',678,1,7);

insert into Products values('black cherry juice',345,1,7);

insert into Products values('Ann Grinder',789,1,8);

insert into OrderItems values(1,1,2);

insert into OrderItems values(2,2,2);

insert into OrderItems values(3,2,4);

insert into OrderItems values(4,3,3);

insert into OrderItems values(5,3,4);

insert into OrderItems values(6,3,5);

insert into OrderItems values(7,4,2);

insert into OrderItems values(8,4,3);

insert into OrderItems values(9,5,2);

insert into OrderItems values(10,6,5);

insert into Rating values(1,1,4,'2017-11-09','Good');

insert into Rating values(2,1,5,'2017-11-09','Good');

insert into Rating values(2,2,1,'2017-11-09','Bad quailty');

insert into Rating values(3,3,6,'2017-11-09','Good');

insert into Rating values(3,4,7,'2017-11-09','Satisfactory');

insert into Rating values(3,5,10,'2017-11-09','Excellent');

insert into Rating values(4,6,4,'2017-11-09','Good');

insert into Rating values(4,7,5,'2017-11-09','Good');

insert into Rating values(5,8,6,'2017-11-09','Good');

insert into Rating values(6,9,7,'2017-11-09','Good');

insert into Warehouse values('34561','Droom');

insert into Warehouse values('45671','Drexel');

insert into Warehouse values('34567','Aurra');

insert into Warehouse values('45678','Ayala');

insert into Warehouse values('12345','Airen');

insert into Warehouse values('98765','Cade');

insert into Warehouse values('23876','Bardan');

insert into Warehouse values('34903','Dengar');

insert into Warehouse values('23907','Deckard');

insert into Warehouse values('34215','Malcom');

insert into Stocks values(1,1,15);

insert into Stocks values(2,1,23);

insert into Stocks values(3,1,4);

insert into Stocks values(4,1,5);

insert into Stocks values (5,1,5);

insert into Stocks values(6,1,5);

insert into Stocks values(7,1,5);

insert into Stocks values(8,1,5);

insert into Stocks values(1,2,20);

insert into Stocks values(2,3,15);

insert into Stocks values(3,4,30);

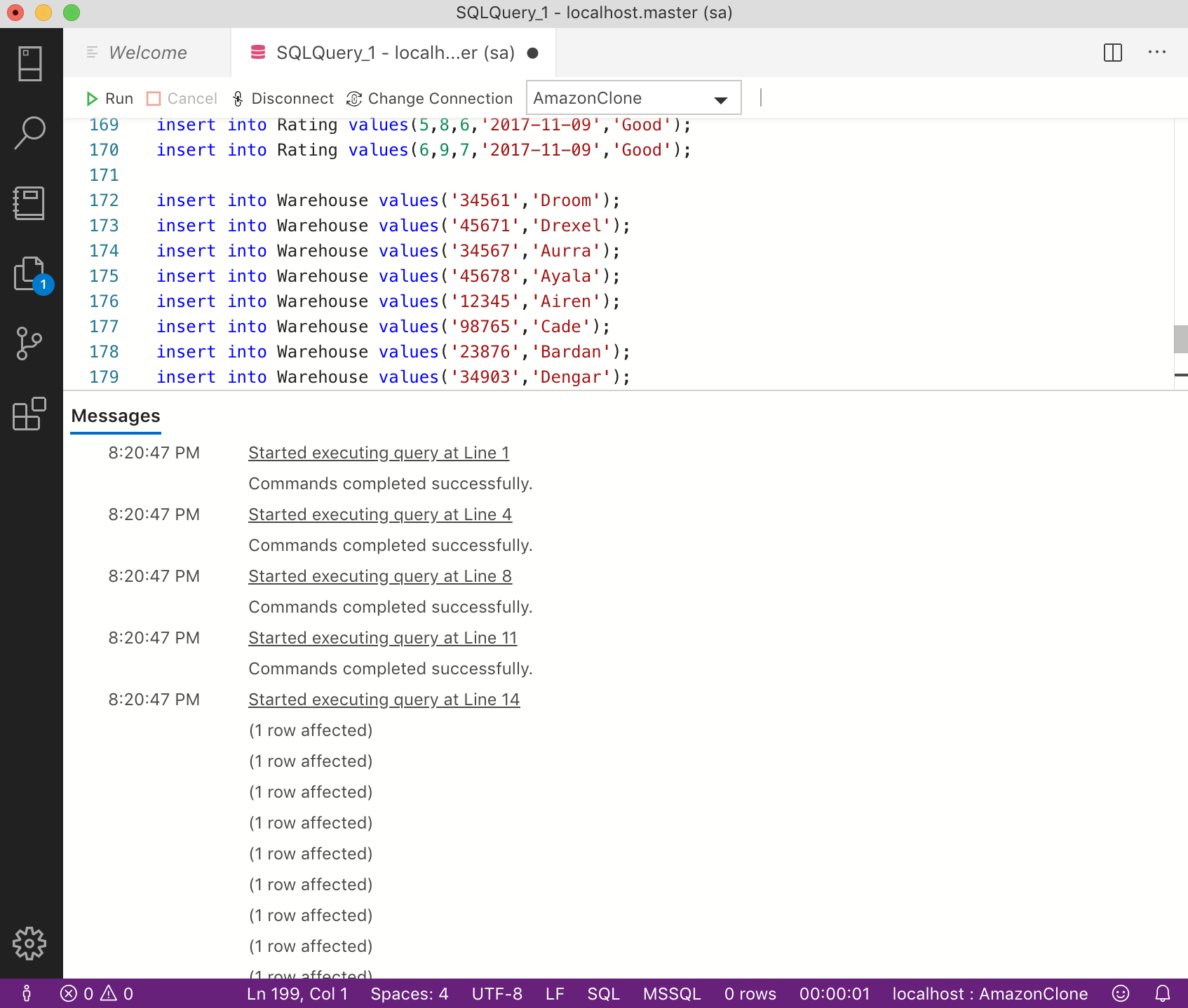
insert into Stocks values(4,5,16);

insert into Stocks values (5,6,30);

insert into Stocks values(6,7,20);

insert into Stocks values(7,8,45);

insert into Stocks values(8,9,20);

****

1. **SCENARIOS(Testing)**

**Scenario 1:**

**Select all Customers who have ordered product with ProductId 4**

**Code:**

use AmazonClone

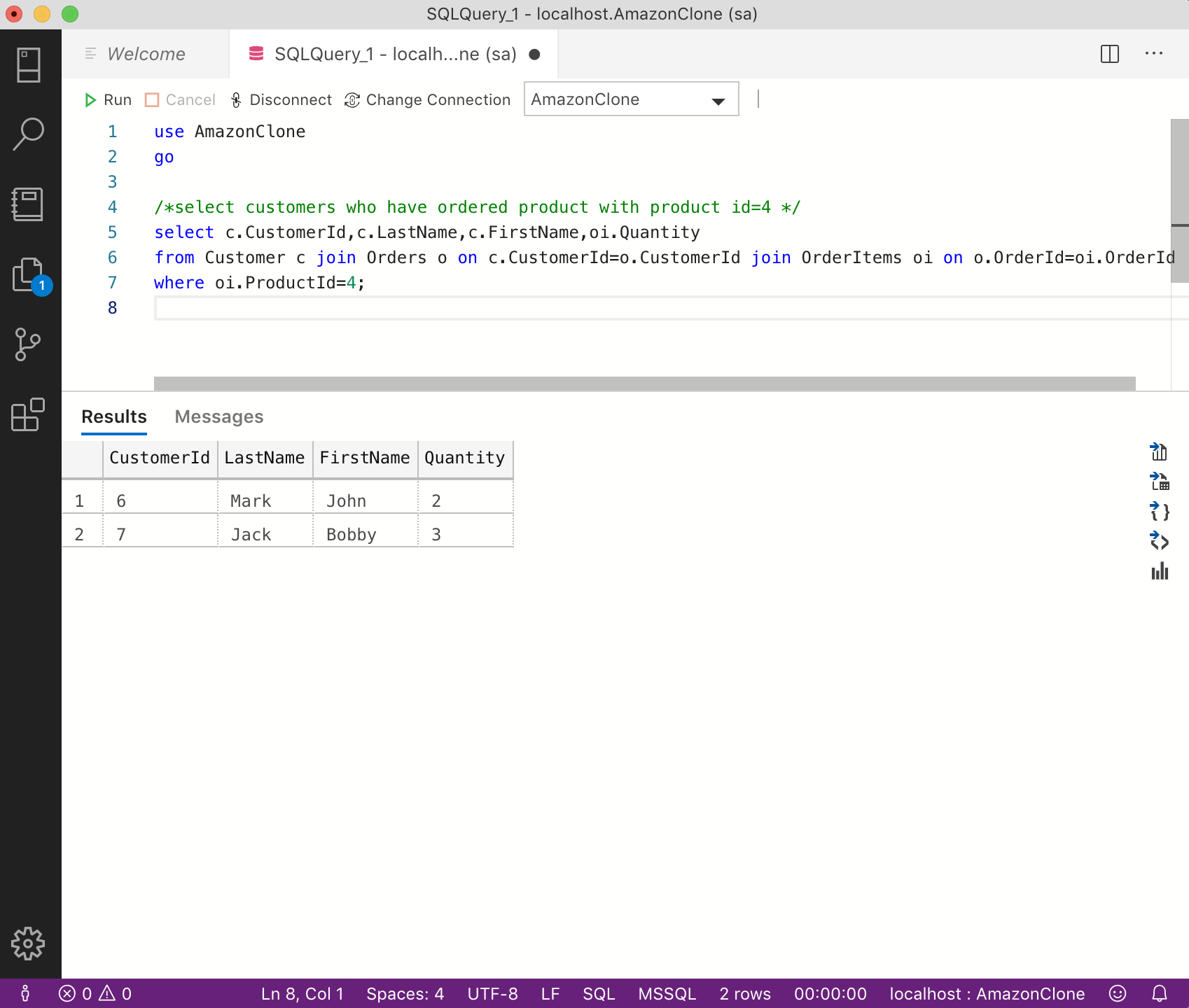
go

select c.CustomerId,c.LastName,c.FirstName,oi.Quantity

from Customer c join Orders o on c.CustomerId=o.CustomerId join OrderItems oi on o.OrderId=oi.OrderId

where oi.ProductId=4;

**Screenshot:**



**Comments:**

The query selects Customer information who have ordered product with product Id 4.

**Scenario 2:**

**Select product which has highest rating**

**Code:**

use AmazonClone

GO

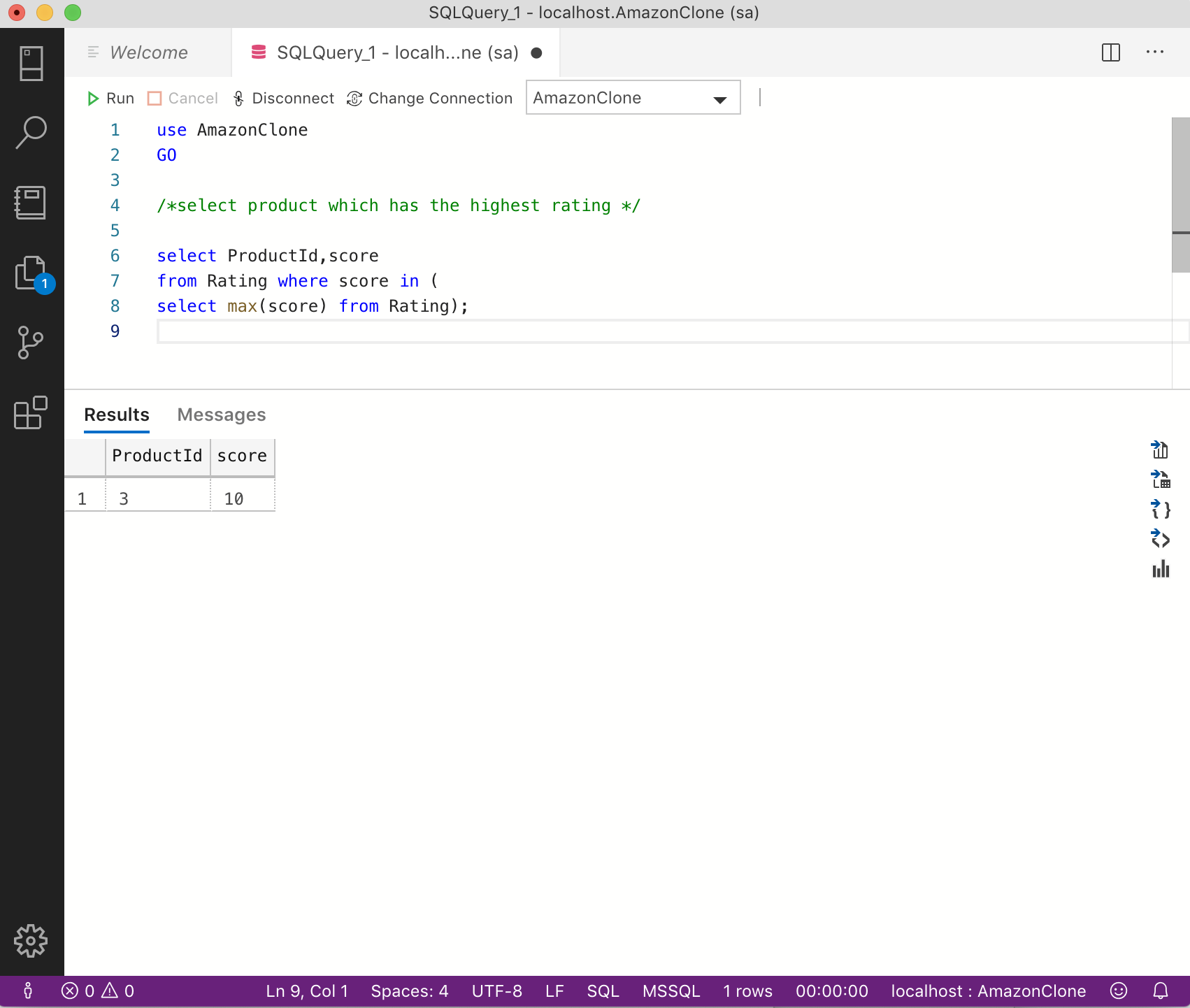
/\*select product which has the highest rating \*/

select ProductId,score

from Rating where score in (

select max(score) from Rating);

**Screenshot:**

****

**Comments:**

The query selects the product which has highest rating. This is most useful when admin wants to check which product is satisfying customers the most.

**Scenario 3:**

**Select customers who have ordered product with productid 2 during ‘2017-04-15’**

**Code:**

use AmazonClone

GO

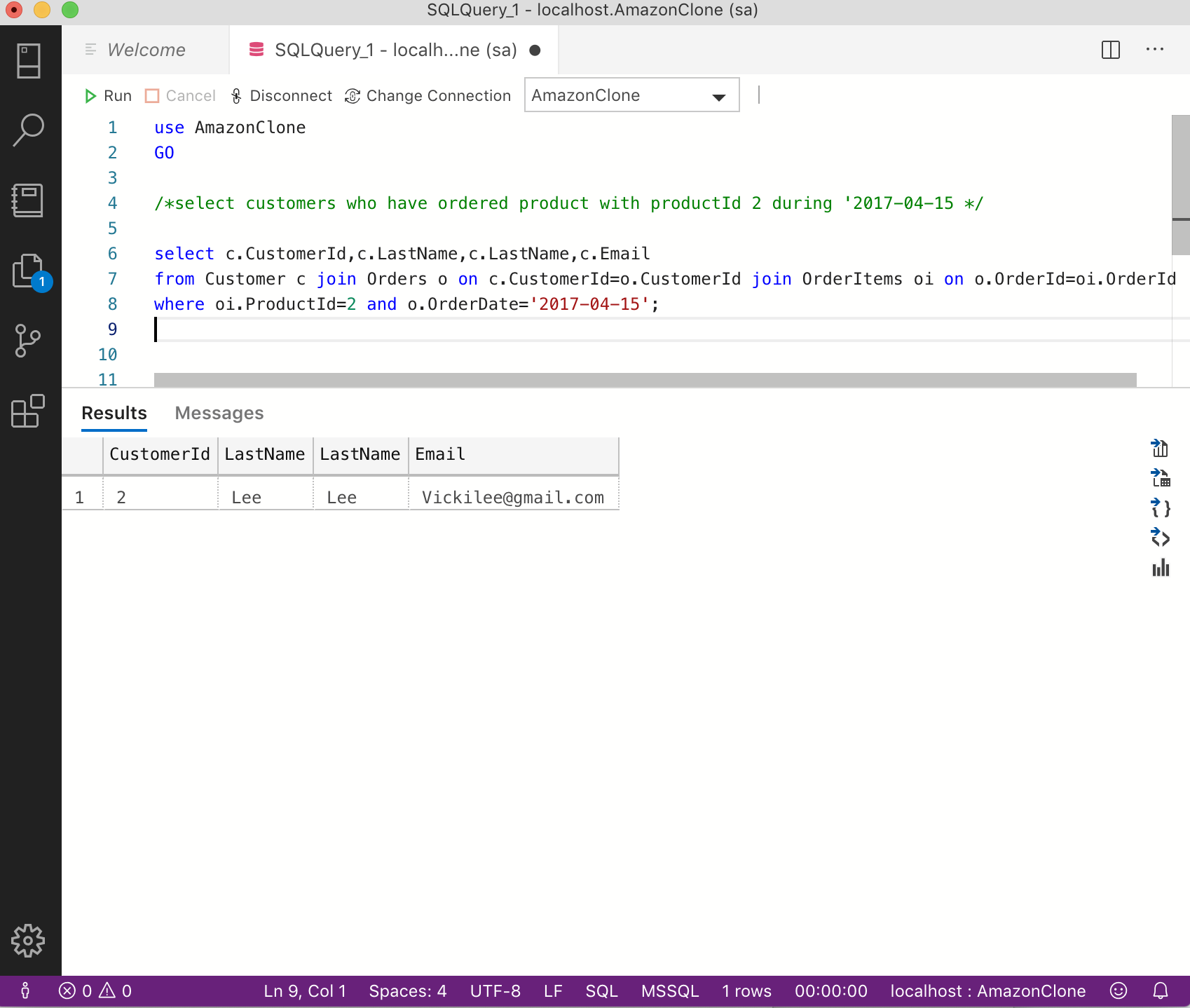
/\*select customers who have ordered product with productId 2 during '2017-04-15 \*/

select c.CustomerId,c.LastName,c.LastName,c.Email

from Customer c join Orders o on c.CustomerId=o.CustomerId join OrderItems oi on o.OrderId=oi.OrderId

where oi.ProductId=2 and o.OrderDate='2017-04-15';

**Screenshot:**

****

**Comments:**

The query selects customer information who have ordered product with productid 2 during particular date. This is useful when admin wants to know what kind of customers orders particular product on particular date.

**Scenario 4:**

**Select product which has been ordered maximum number of times**

**Code:**

use AmazonClone

GO

/\*select product which has been ordered maximum number of times\*/

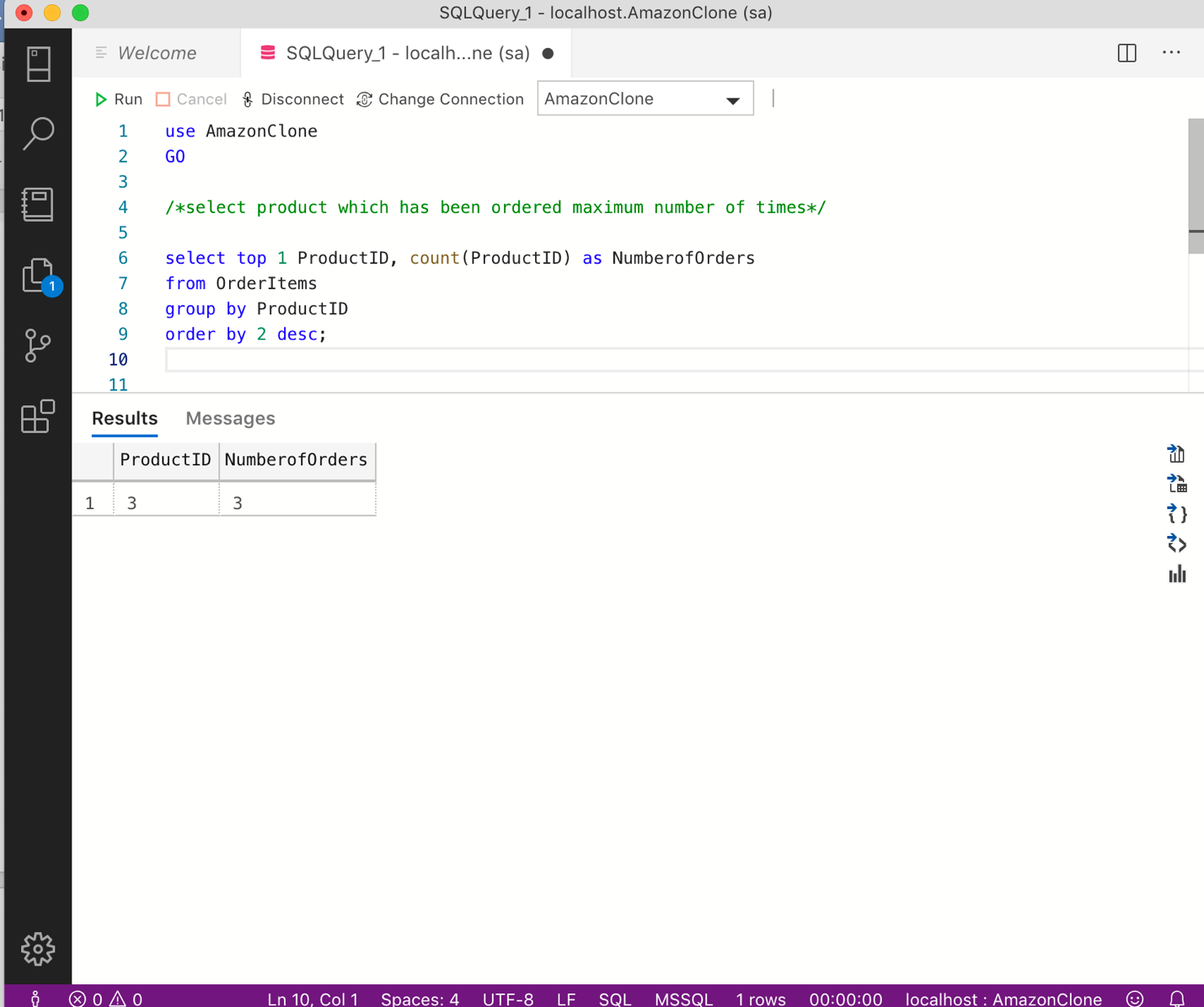
select top 1 ProductID, count(ProductID) as NumberofOrders

from OrderItems

group by ProductID

order by 2 desc;

**Screenshot:**



**Comments:**

The query selects the product which has been ordered maximum number of times.

**Scenario 5:**

**Create a view that selects highest ratings of the products**

**Code:**

create view HighestRatings

as

select max(score) as highestScore,r.ProductId

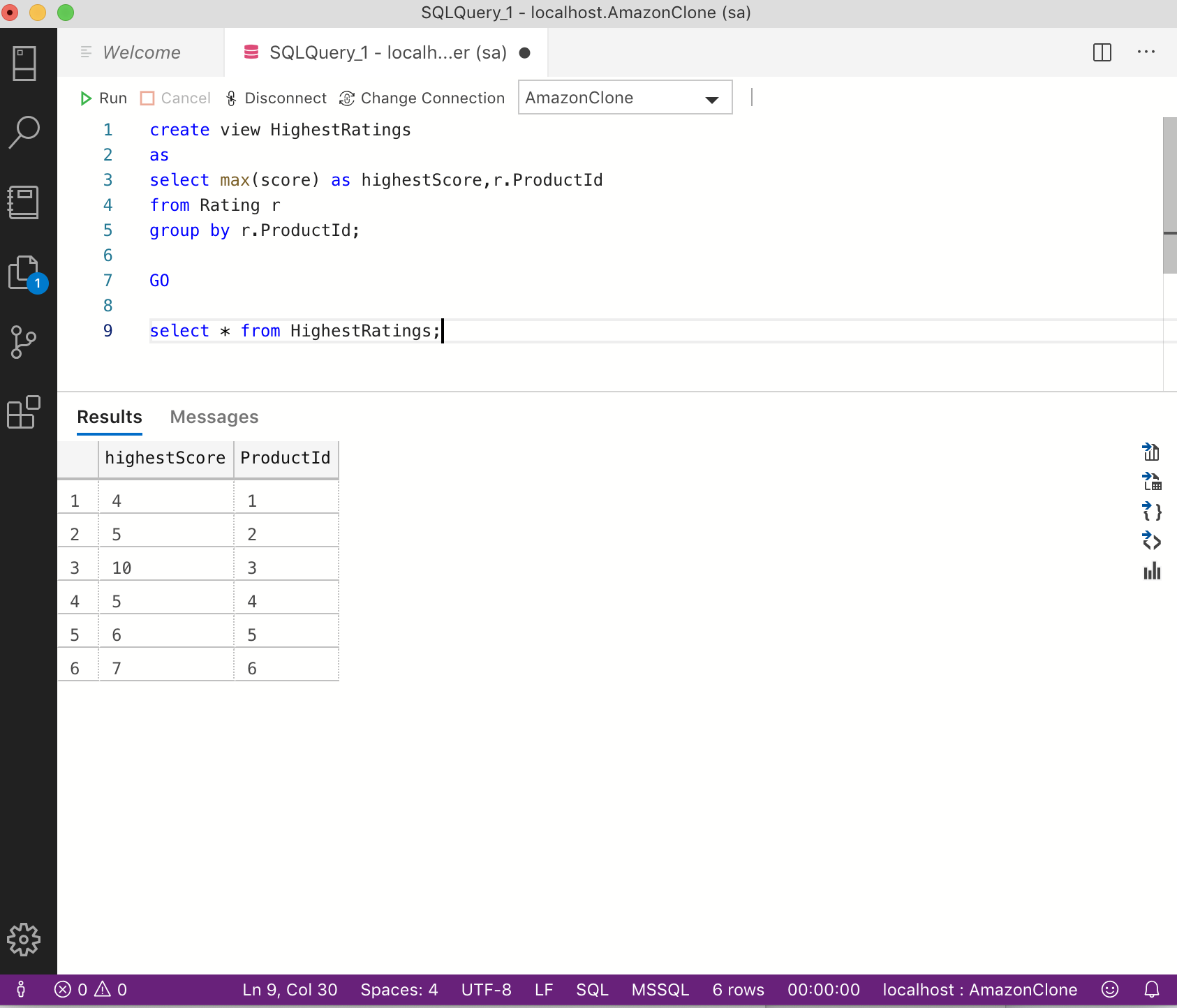
from Rating r

group by r.ProductId;

GO

select \* from HighestRatings;

**Screenshot:**



**Comments:**

The view displays highest rating received for each product.

**Scenario 6:**

**Create a view that displays number of orders for each product**

**Code:**

create view NumofOrders AS

select count(\*) as numberofOrders,p.ProductId

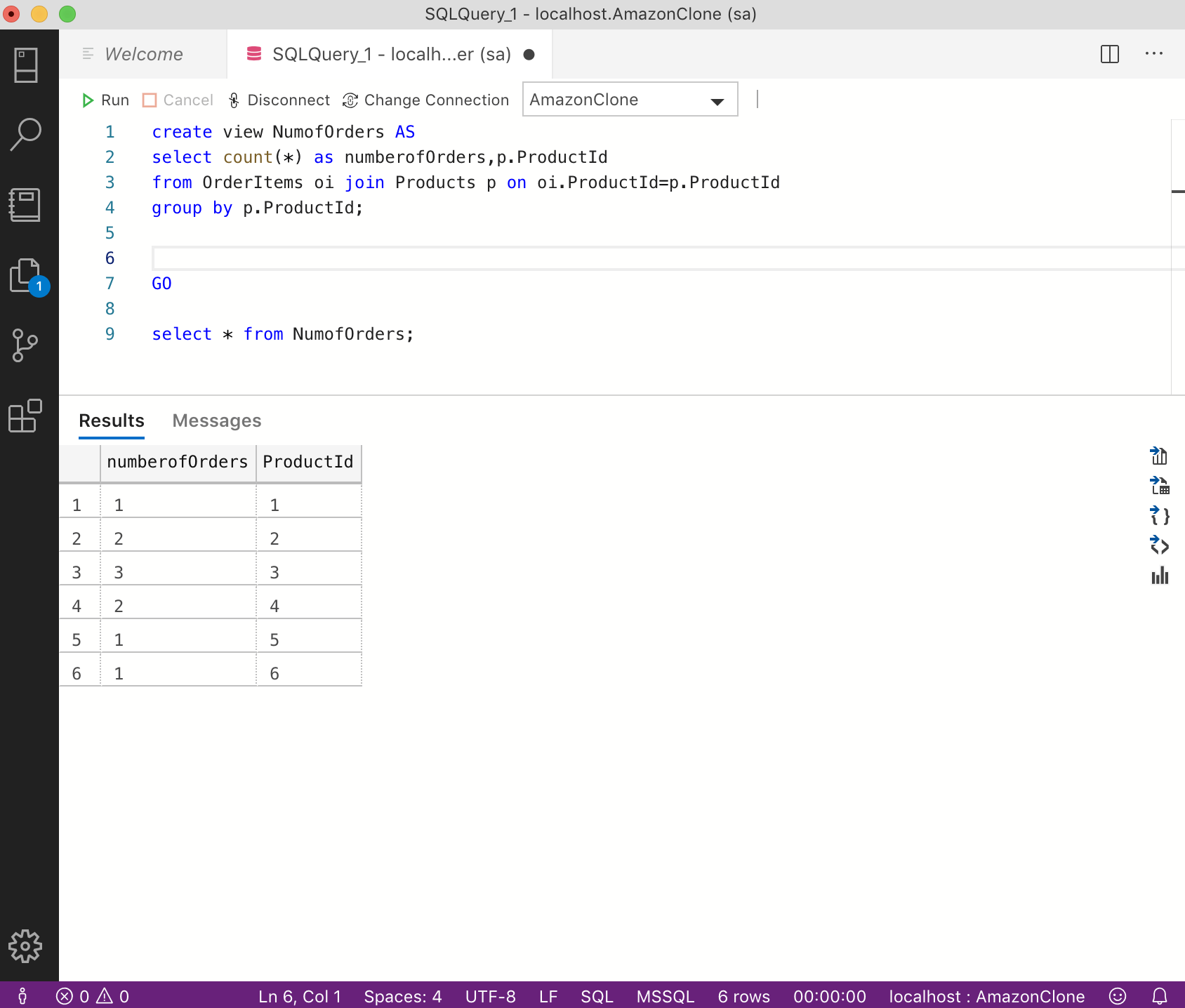
from OrderItems oi join Products p on oi.ProductId=p.ProductId

group by p.ProductId;

GO

select \* from NumofOrders;

**Screenshot:**

****

**Comments:**

The view displays number of orders for each product.

**Scenario 7:**

**Create a view to select Warehouse details of products**

**Code:**

create view WarehouseDetails

as

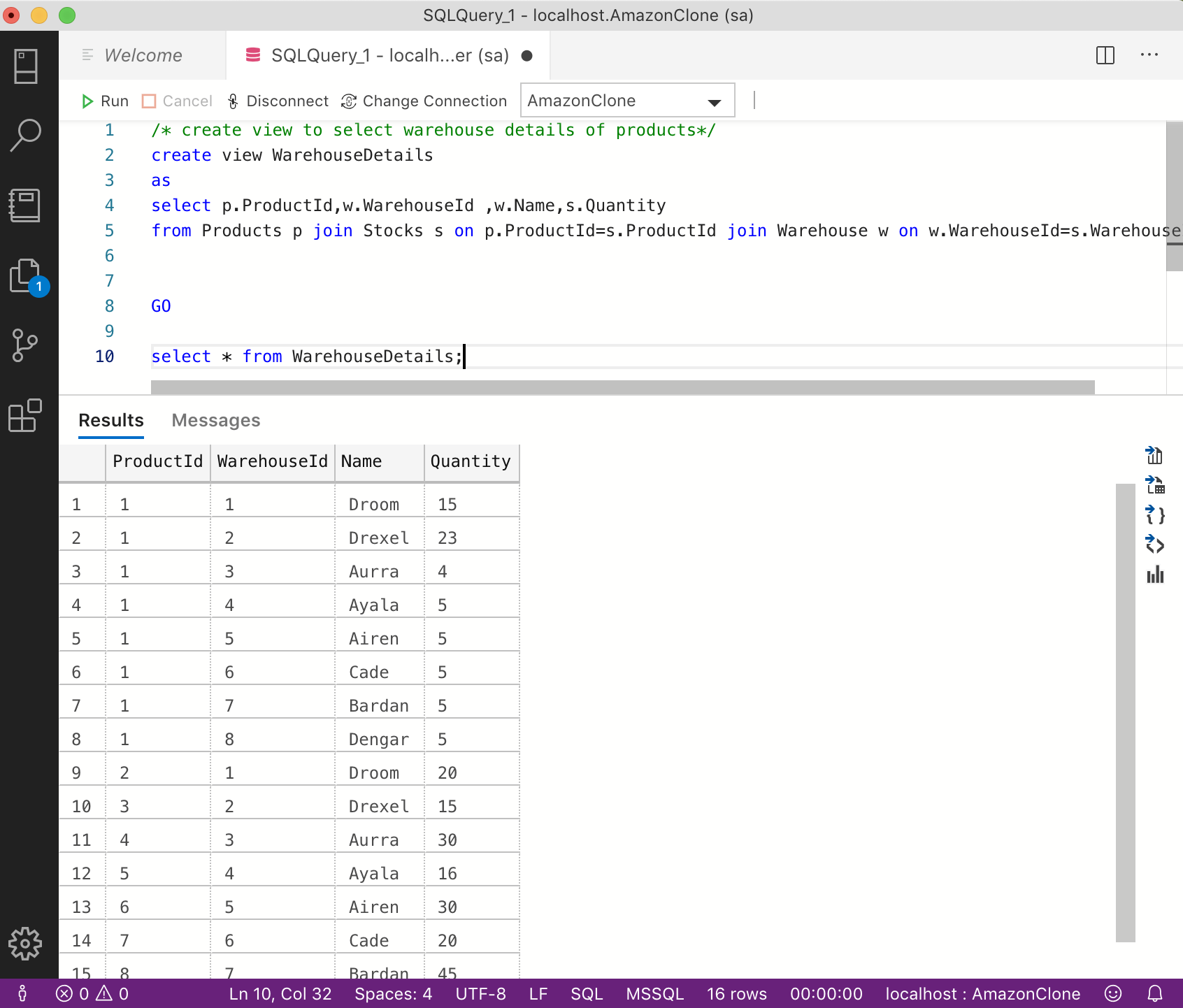
select p.ProductId,w.WarehouseId ,w.Name,s.Quantity

from Products p join Stocks s on p.ProductId=s.ProductId join Warehouse w on w.WarehouseId=s.WarehouseId;

GO

select \* from WarehouseDetails;

**Screenshot:**



**Comments:**

The view returns the warehouse information and number of quantity in stock for each product in each warehouse.

**Scenario 8:**

**Create a view that selects products ordered by people in New York**

**Code:**

create view ProductsOrderState

as

select p.ProductId,p.ProductName,c.State

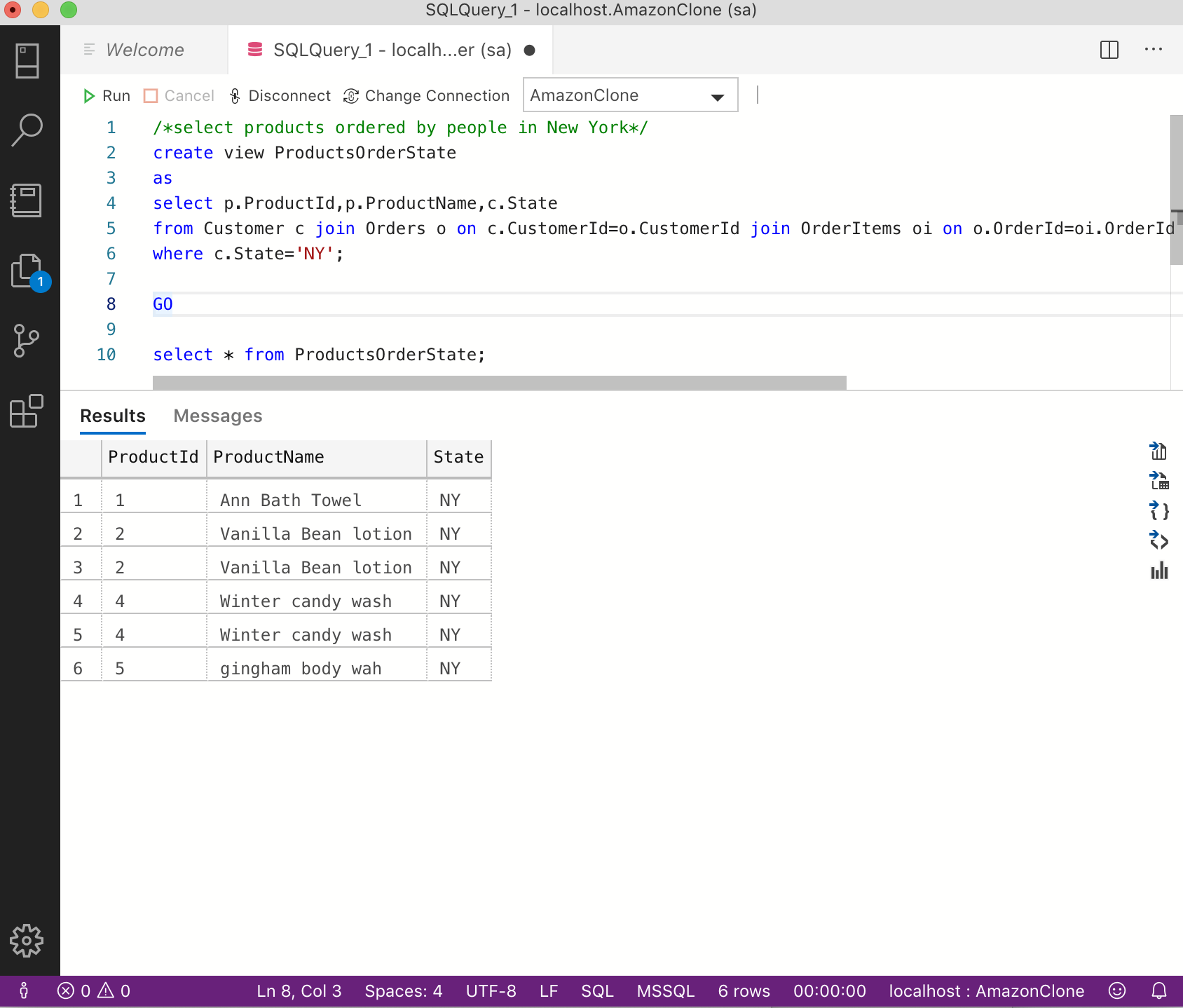
from Customer c join Orders o on c.CustomerId=o.CustomerId join OrderItems oi on o.OrderId=oi.OrderId join Products p on oi.ProductId=p.ProductId

where c.State='NY';

GO

select \* from ProductsOrderState;

**Screenshot:**

****

**Comments:**

The view returns the products ordered by people in New York

**Scenario 9:**

**Create a procedure to display instock products**

**Code:**

create PROCEDURE InstockProducts

as

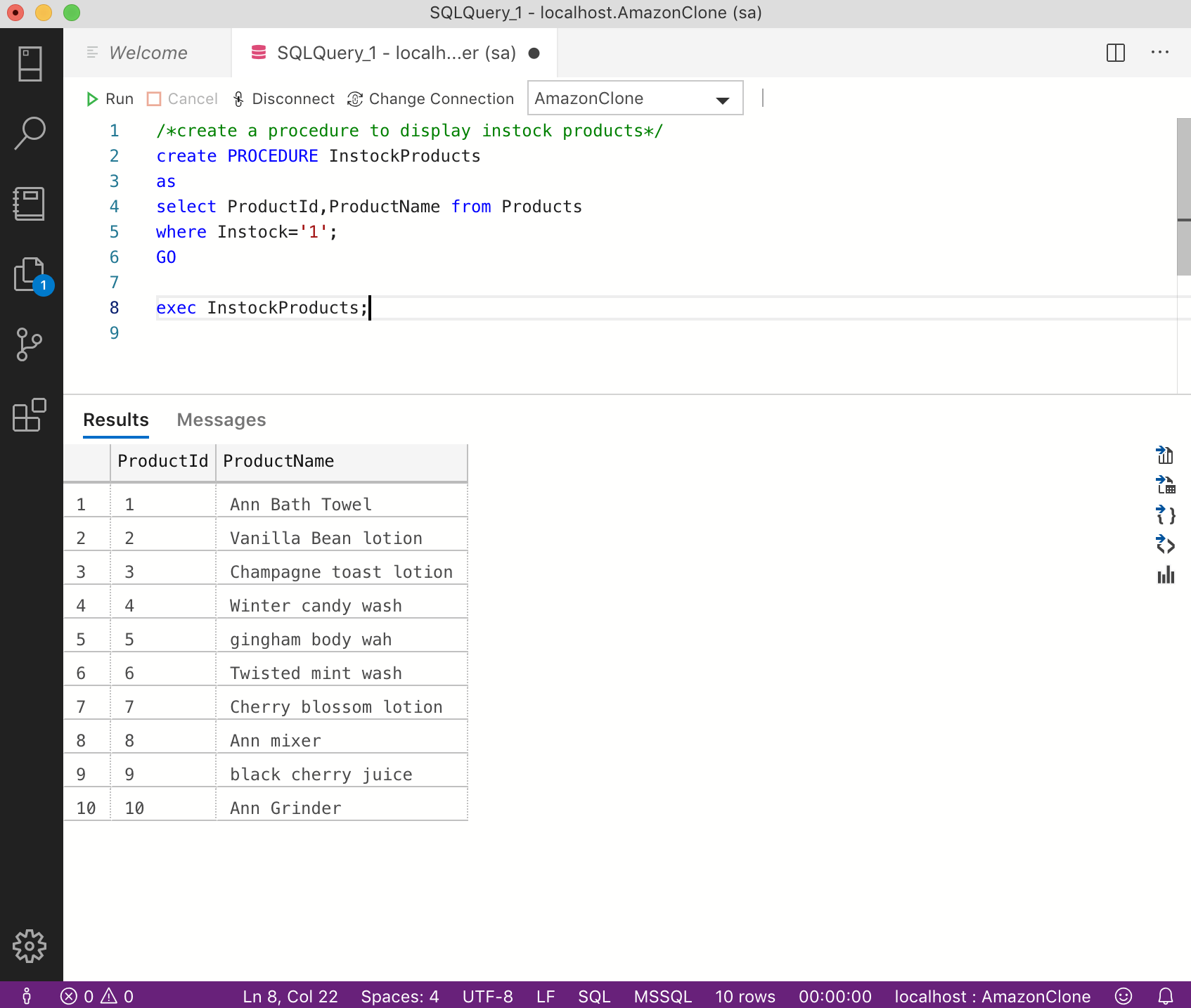
select ProductId,ProductName from Products

where Instock='1';

GO

exec InstockProducts;

**Screenshot:**



**Comments:**

The query creates a procedure that returns the Products that are instock.

**Scenario 10:**

**Create a procedure that returns the quantity of particular product ordered by customer with given customerId**

**CODE:**

create PROCEDURE OrderQuantity

    @CustomerId int,@ProductId int,

    @Quantity int OUTPUT

as

select @Quantity=oi.Quantity

from Customer c join Orders o on c.CustomerId=o.CustomerId join OrderItems oi on o.OrderId=oi.OrderId

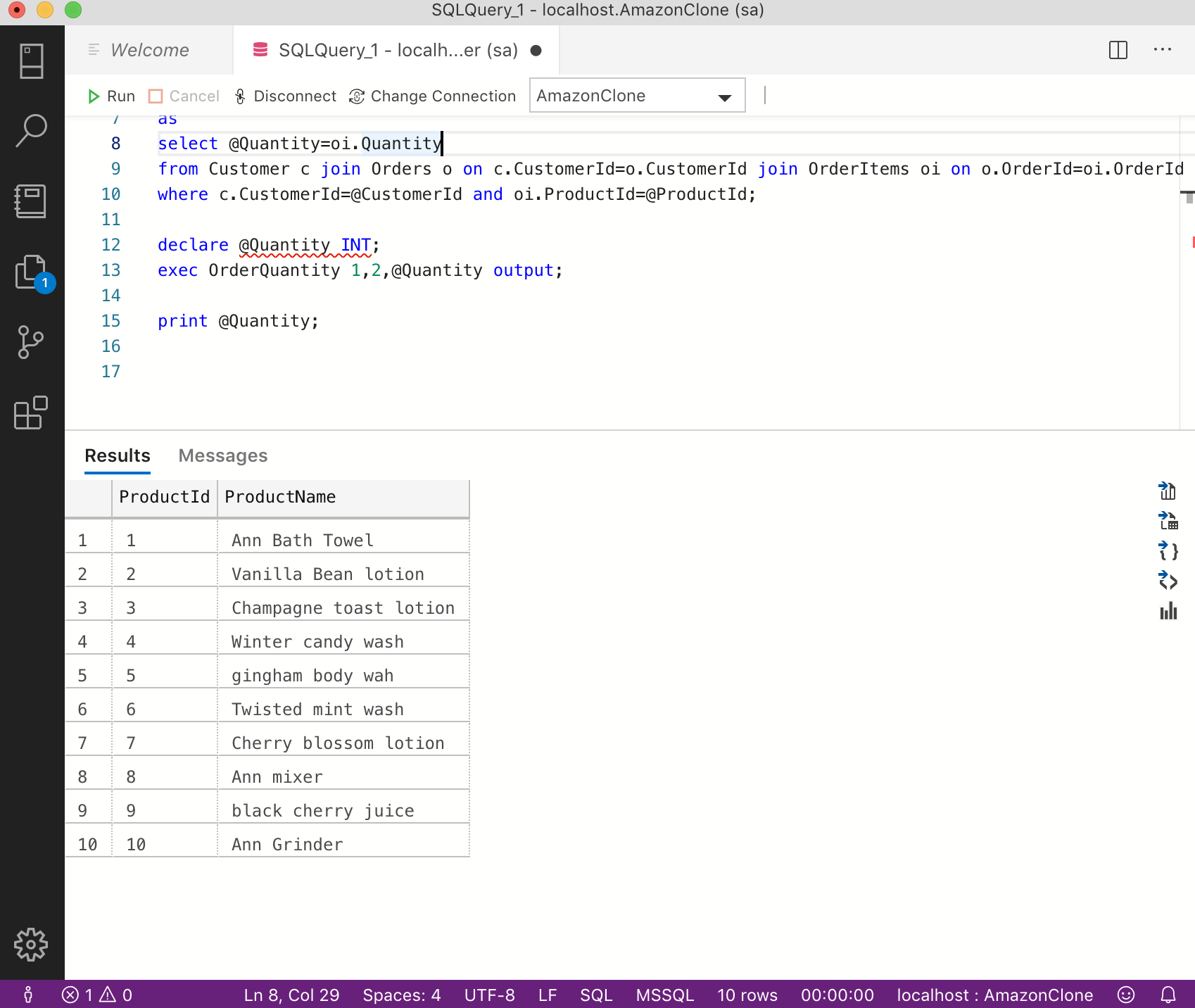
where c.CustomerId=@CustomerId and oi.ProductId=@ProductId;

declare @Quantity INT;

exec OrderQuantity 1,2,@Quantity output;

print @Quantity;

**Screenshot:**



**Comments:**

The query creates a procedure that returns the quantity of particular product ordered by customer with given CustomerID. This helps admin to know the buying trends of the customer

**Scenario 11:**

**Create a procedure to display total amount of order during particular date**

**Code:**

create procedure  TotalAmountOrder

        @Date smallDateTime

as

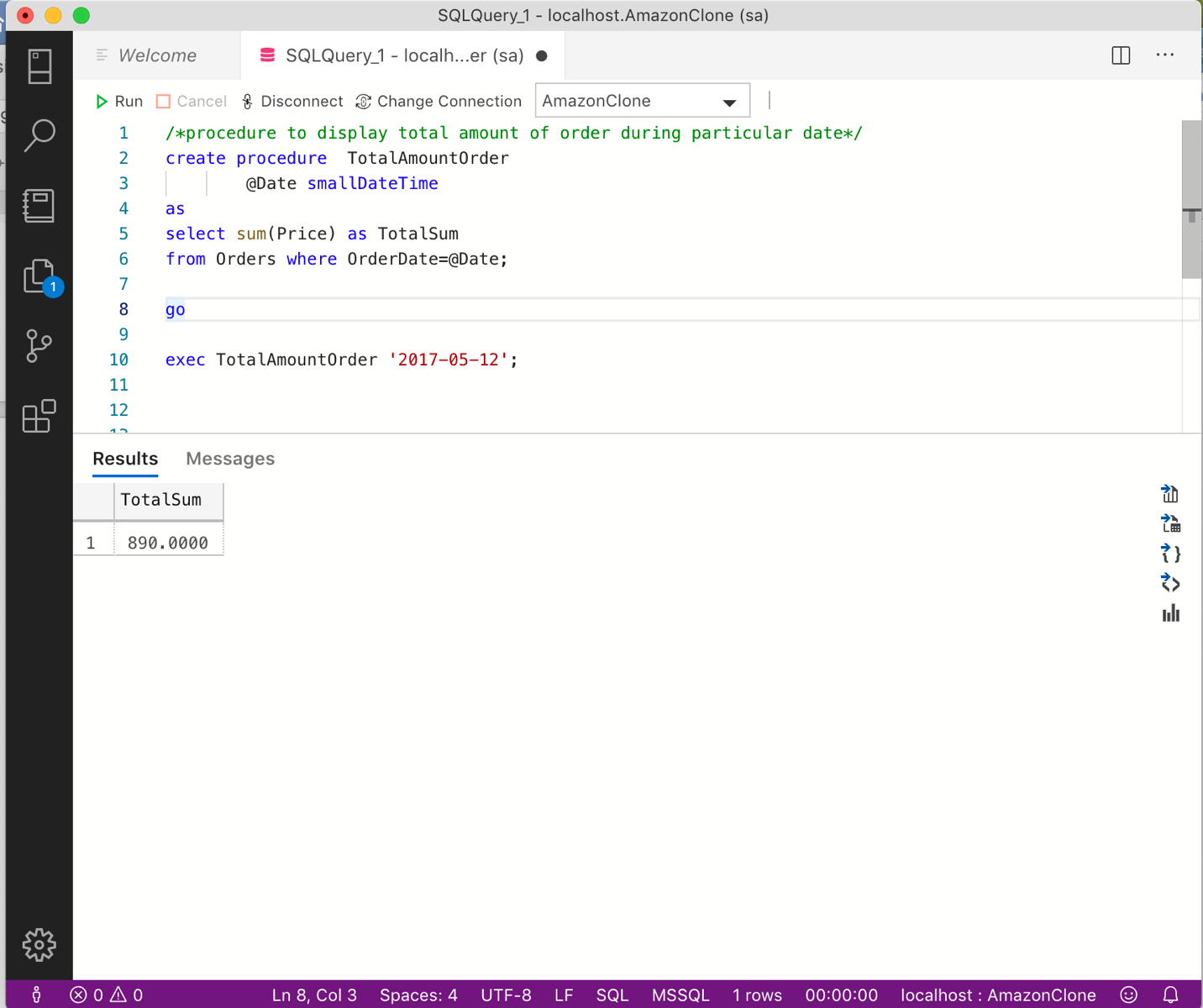
select sum(Price) as TotalSum

from Orders where OrderDate=@Date;

go

exec TotalAmountOrder '2017-05-12';

**Screenshot:**

****

**Comments:**

The procedure returns total amount of order during particular date.

This helps admin to know when the sales are good and when stocks should be there.

**Scenario 12:**

**Create a procedure to return highest rating of a particular product**

**Code:**

create procedure HighestRatingProduct

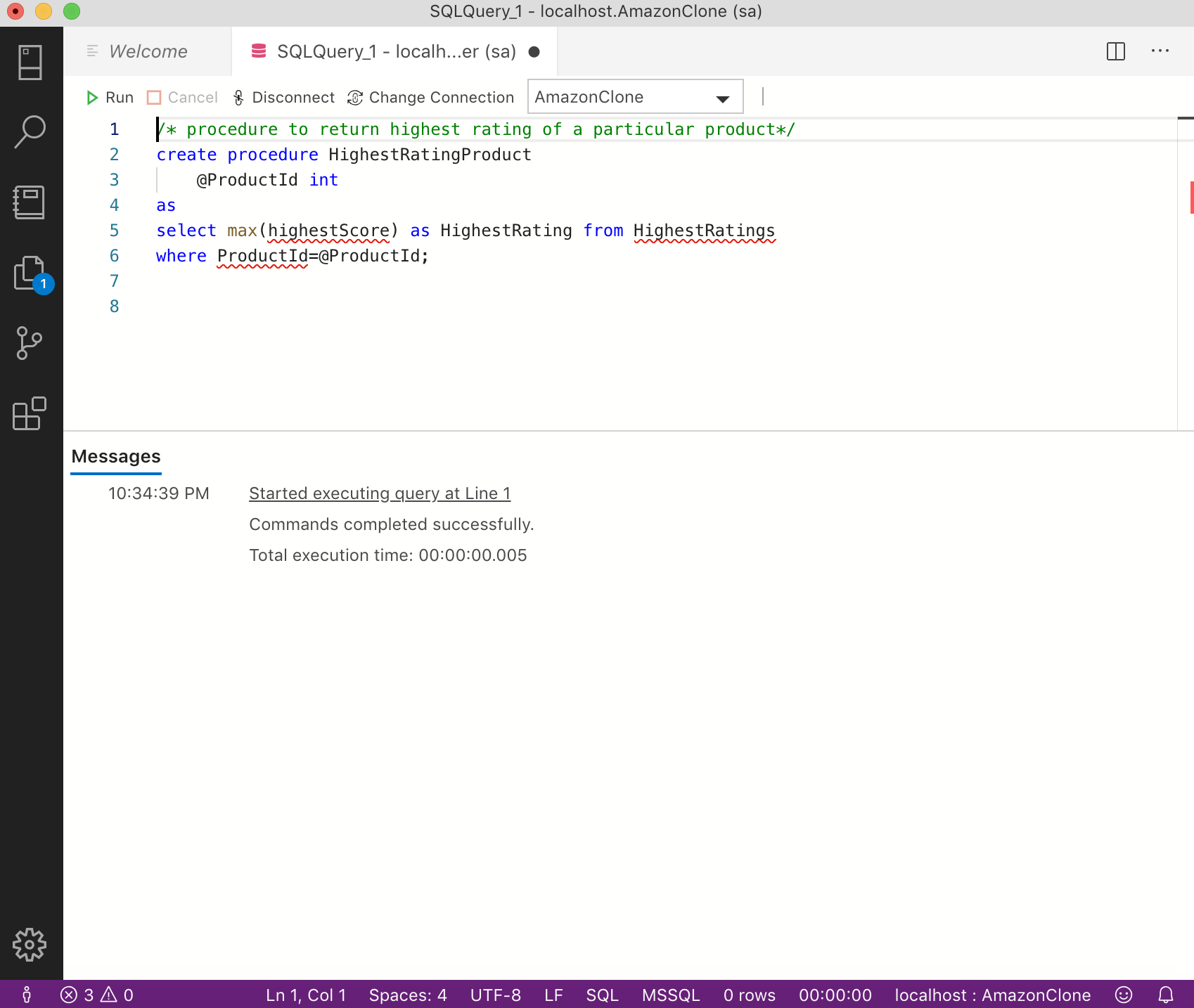
    @ProductId int

as

select max(highestScore) as HighestRating from HighestRatings

where ProductId=@ProductId;

**Screenshot:**

****

**Comments:**

The procedure returns the highest rating of a particular product.

**Scenario 13:**

**Function that takes ProductName and returns its ProductId.**

Code:

use AmazonClone

go

create FUNCTION fnProductId (@ProductName varchar(30)) returns INT

BEGIN

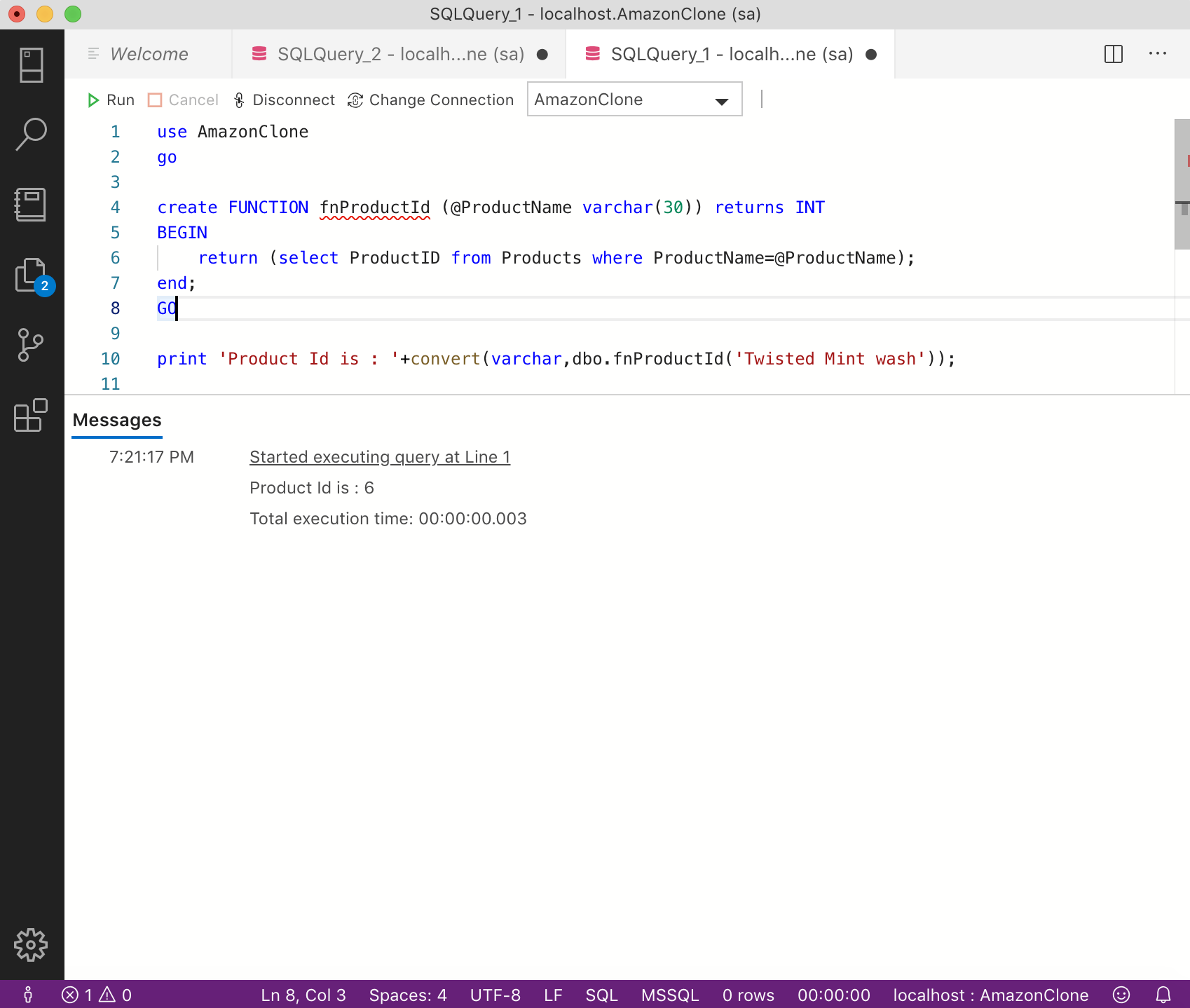
    return (select ProductID from Products where ProductName=@ProductName);

end;

GO

print 'Product Id is : '+convert(varchar,dbo.fnProductId('Twisted Mint wash'));

Screenshot:



**Comments:**

The function returns the ProductId, given the ProductName.

**Scenario 14**

**Function to check whether product is instock when ProductName is given.**

**Code:**

use AmazonClone

go

create function IsInstock(@ProductName varchar(30)) returns char

begin

return(select InStock

from Products

where ProductId=dbo.fnProductId(@ProductName));

END;

GO

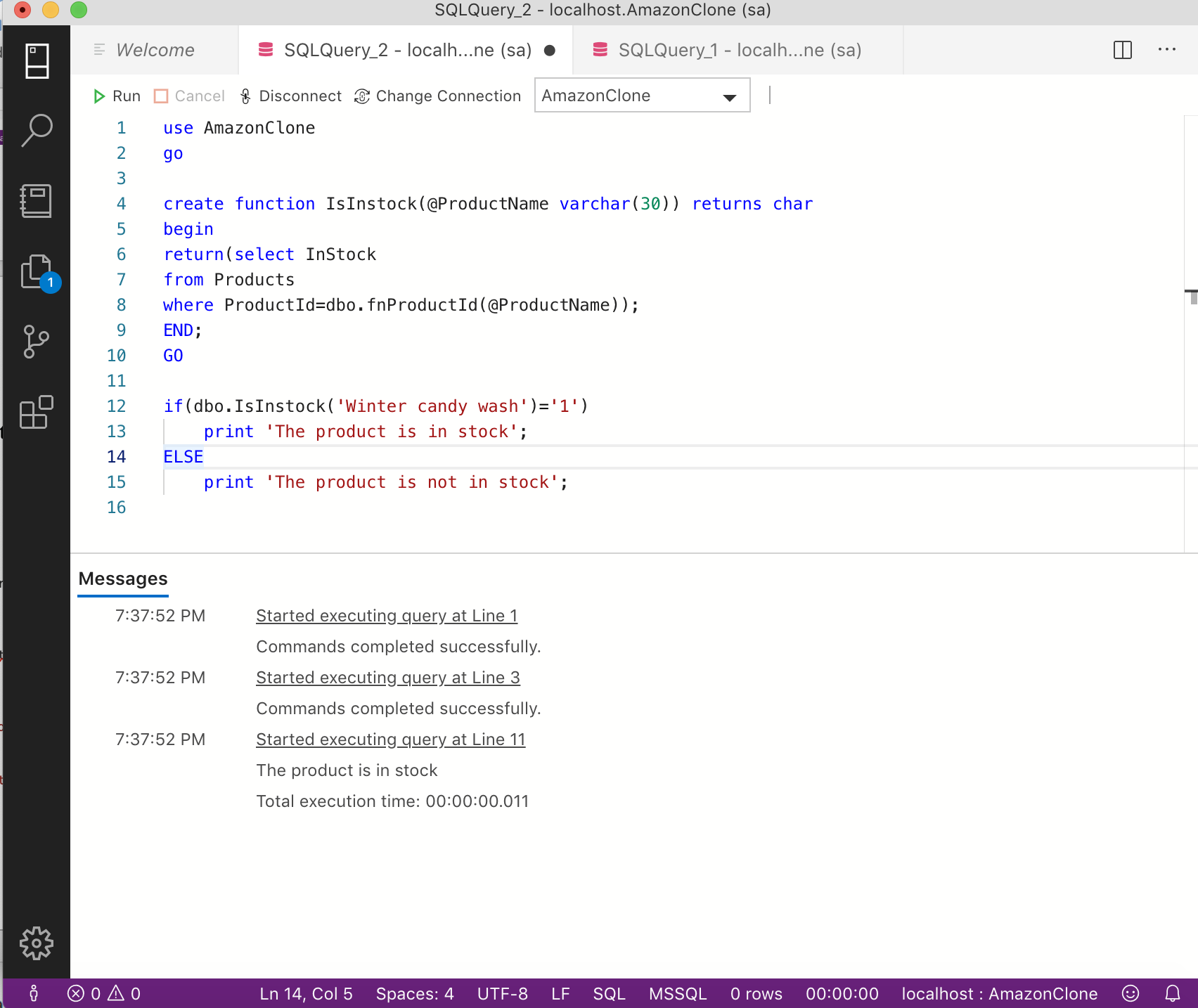
if(dbo.IsInstock('Winter candy wash')='1')

    print 'The product is in stock';

ELSE

    print 'The product is not in stock';

**Screenshot:**



**Comments:**

The function returns whether product is instock when ProductName is given.

**Scenario 15**

**Function that returns highest rating of given product**

**CODE:**

use AmazonClone

go

create function fnHighestRatingProduct(@ProductId int) returns int

begin

return (select max(highestScore) as HighestRating from HighestRatings

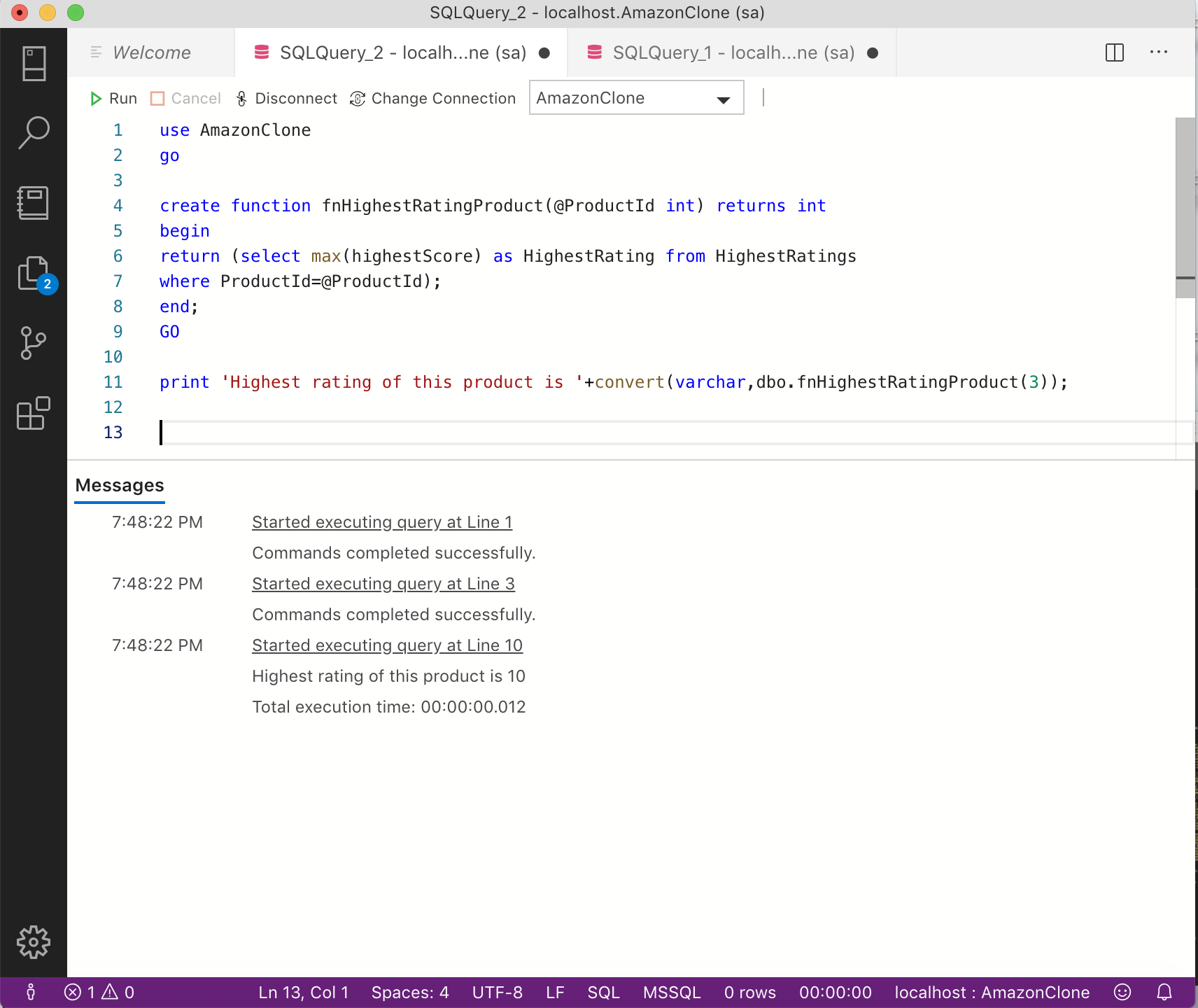
where ProductId=@ProductId);

end;

GO

print 'Highest rating of this product is '+convert(varchar,dbo.fnHighestRatingProduct(3));

**Screenshot:**

****

**Comments:**

The function returns the highest rating of given product.

**Scenario 16:**

**Function that returns the totalSumOrder during the given date**

**Code:**

use AmazonClone

go

create function fnTotalAmountOrder(@Date smallDateTime) returns money

BEGIN

return (select sum(Price) as TotalSum

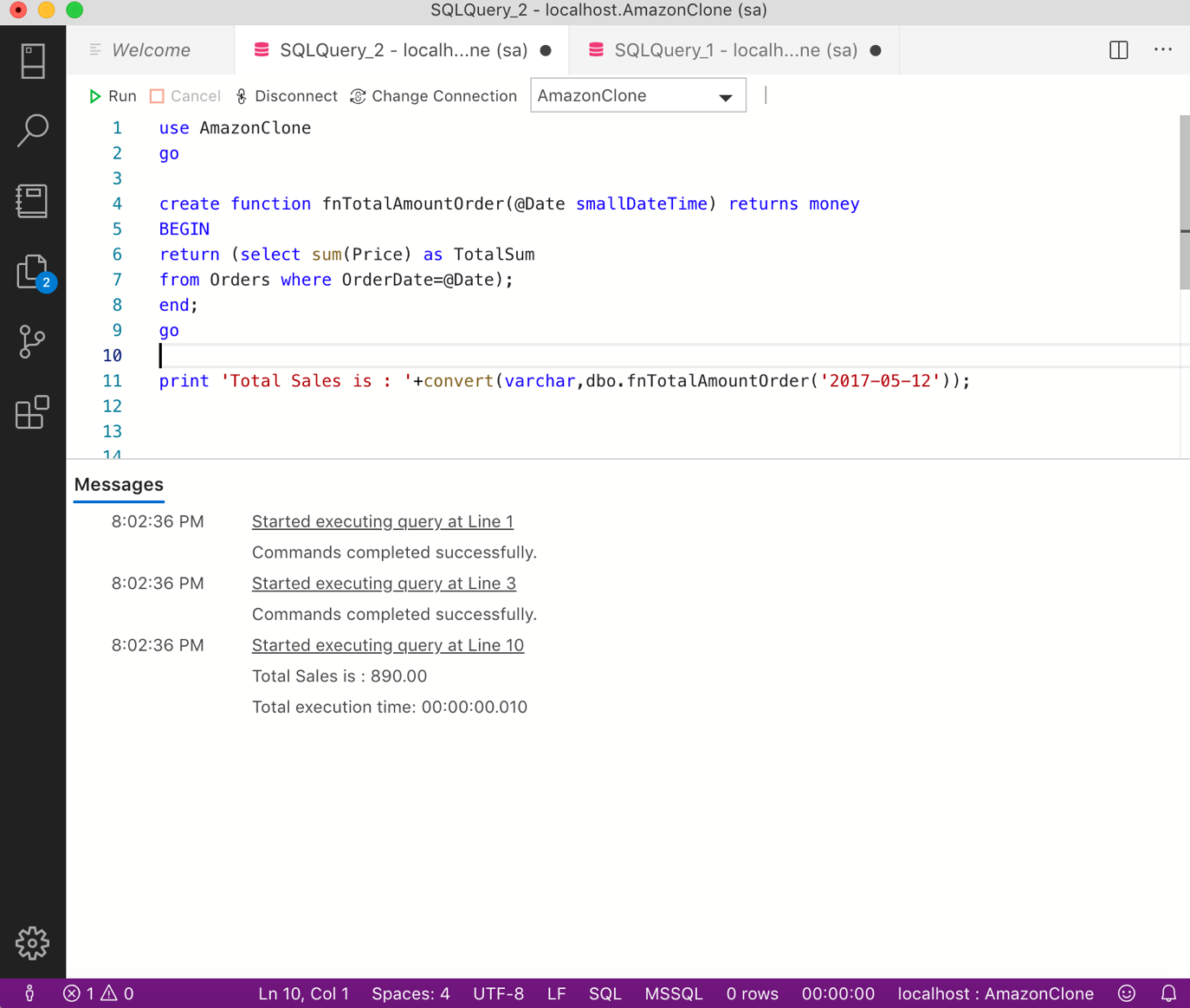
from Orders where OrderDate=@Date);

end;

go

print 'Total Sales is : '+convert(varchar,dbo.fnTotalAmountOrder('2017-05-12'));

**Screenshot:**

****

**Comments:**

The function returns the totalSumOrder during the given date.

**Scenario 17:**

**Trigger to change the State to Uppercase for consistency after insert or update:**

**Code:**

use AmazonClone

GO

create TRIGGER StateUpper

on Customer

after insert,update

AS

update Customer

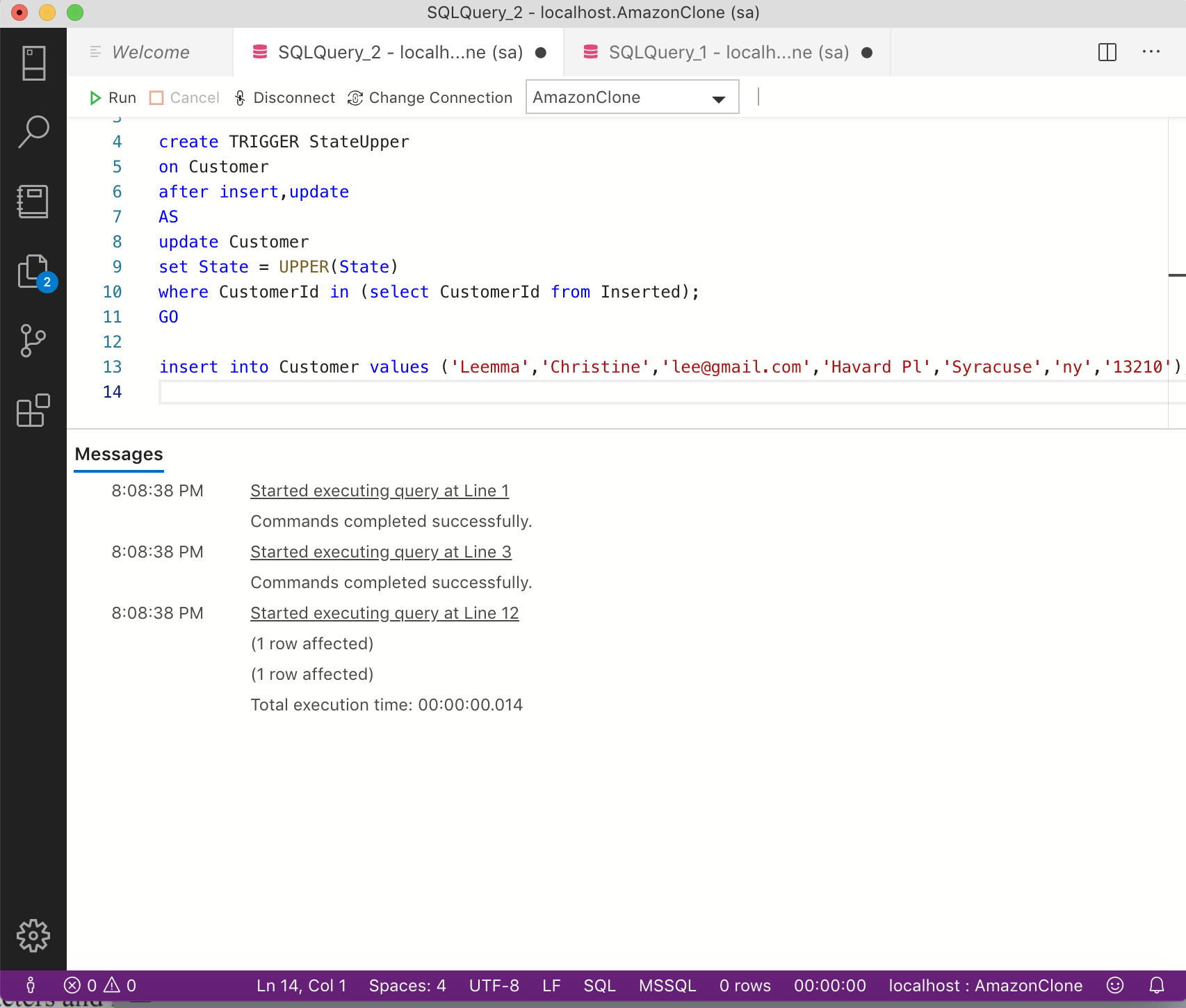
set State = UPPER(State)

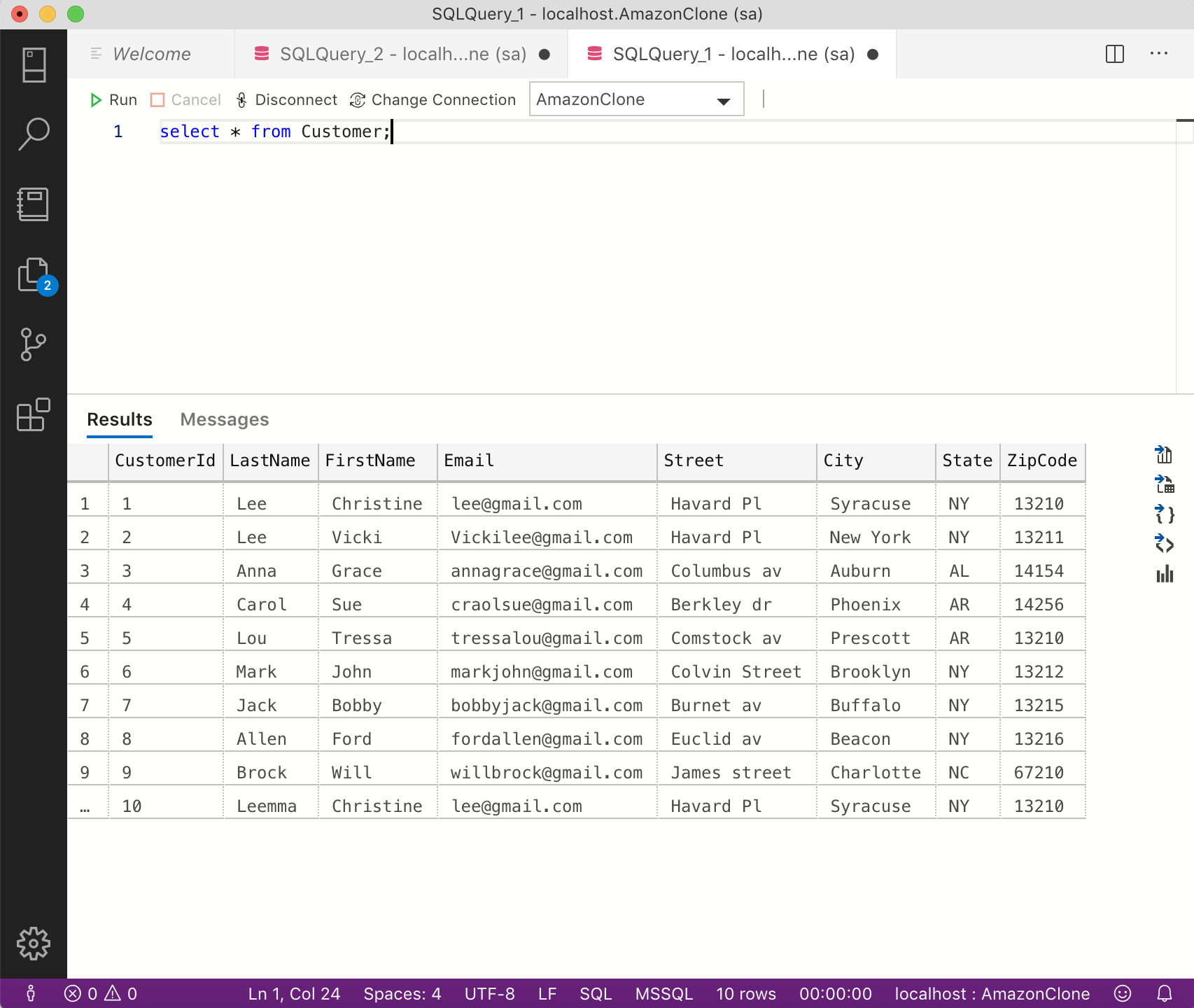
where CustomerId in (select CustomerId from Inserted);

GO

insert into Customer values ('Leemma','Christine','lee@gmail.com','Havard Pl','Syracuse','ny','13210');

**Screenshot:**





**Comments:**

The query creates a trigger to change the state to uppercase for consistency after insert or update.

**Scenario 18:**

**Trigger to insert order to OrdersArchive table after it has been deleted from Orders table.**

**CODE:**

create trigger ArchiveOrder

on Orders

after delete

AS

insert into OrdersArchive(

    OrderId,

    OrderDate,

    RequiredDate,

    ShippingDate,

    Price,

    CustomerId,

    ShippingId

) select

    OrderId,

    OrderDate,

    RequiredDate,

    ShippingDate,

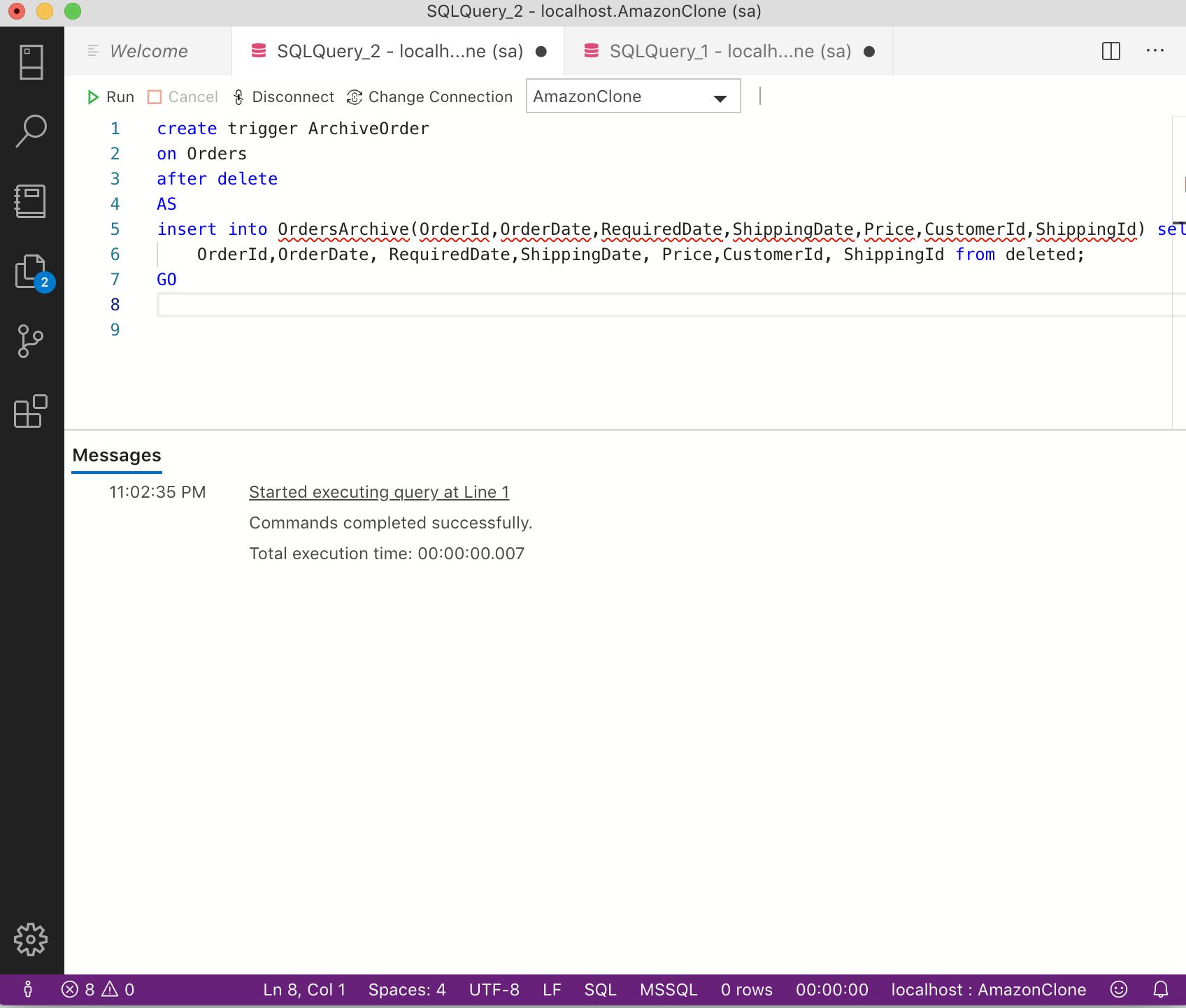
    Price,

    CustomerId,

    ShippingId

     from deleted;

**Screenshot:**

****

**Comments:**

The query creates a trigger to insert order to OrdersArchive table after it has been deleted from Orders table.

**Scenario 19:**

**Create role OrderEntry and give update permission to the new role for the Orders table,update and insert permission for the OrderItems table and select permission for all users table.**

**Code:**

create role OrderEntry;

grant UPDATE

on dbo.Orders

to OrderEntry

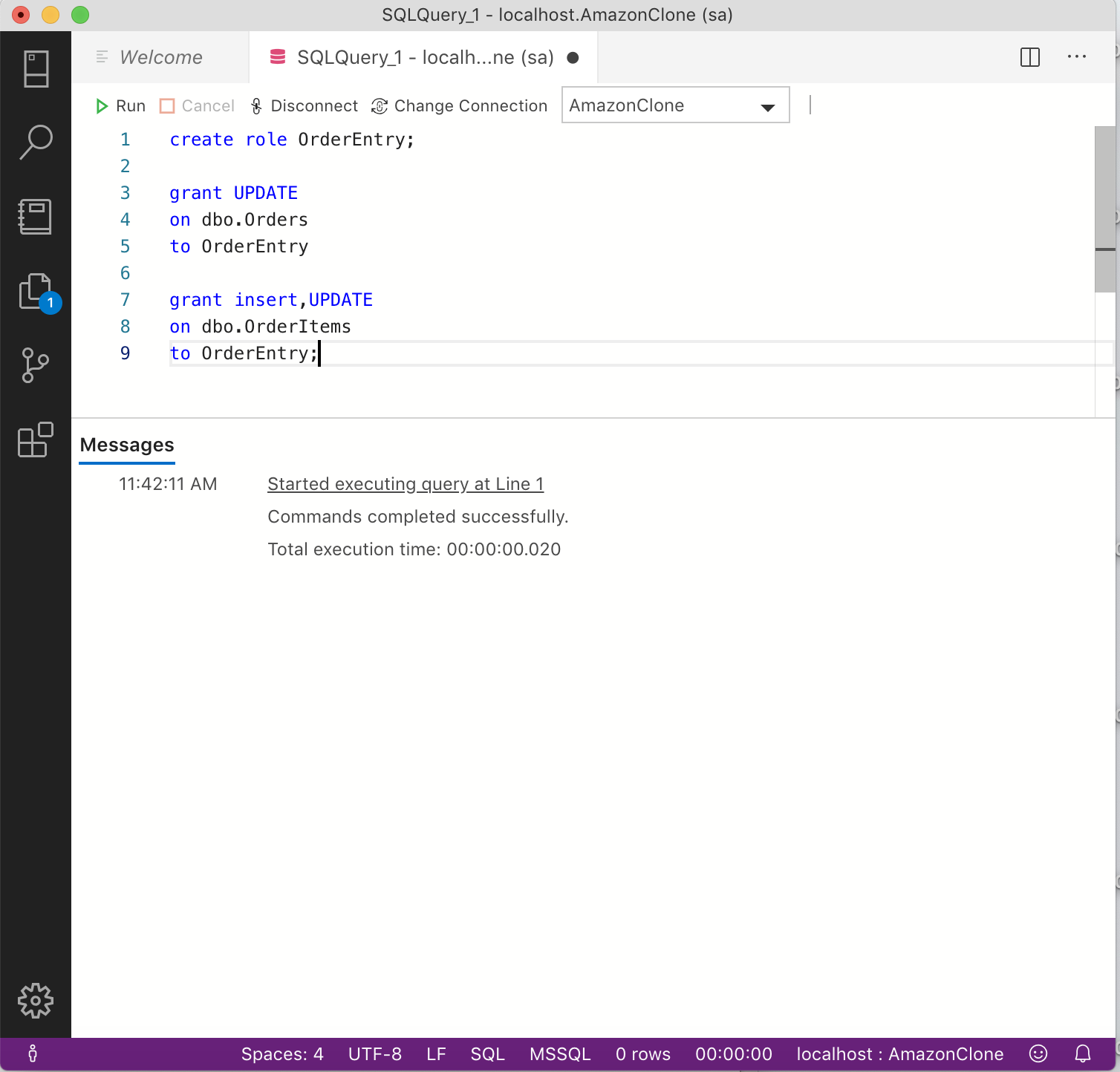
grant insert,UPDATE

on dbo.OrderItems

to OrderEntry;

alter role db\_datareader add member OrderEntry;

**Screenshot:**

****

****

**Comments:**

The script creates a role named OrderEntry and gives update permission to the new role for the Orders table, update and insert permission for the OrderItems table and selet permission for all users table.

**Scenario 20:**

**Create a login with ID Watson and password ‘HarryPotter@1’ and set the default database to AmazonClone database. Create a user ‘Emma’ for the login and assign the user to the OrderEntry role.**

**Code:**

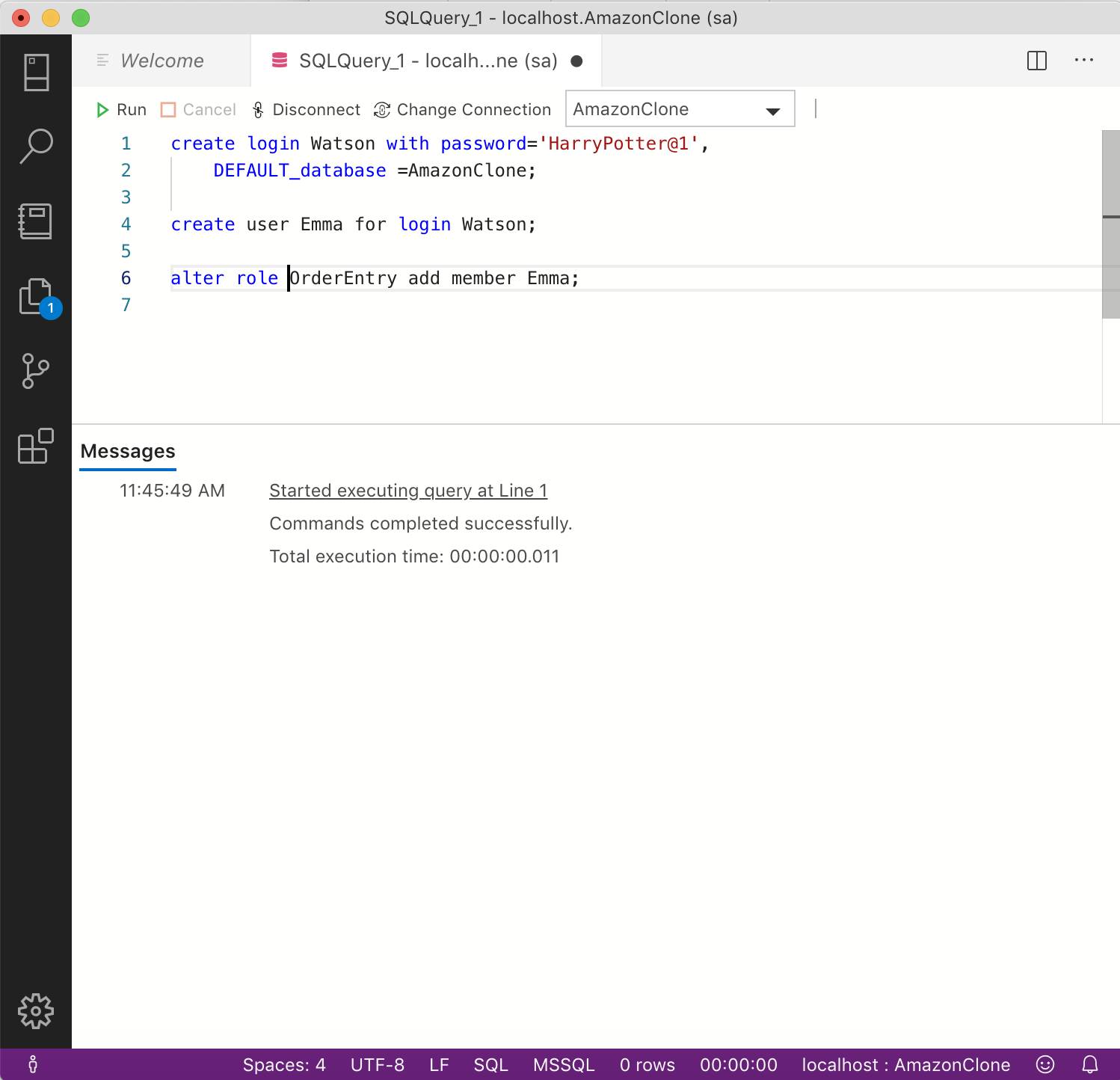
create login Watson with password='HarryPotter@1',

    DEFAULT\_database =AmazonClone;

create user Emma for login Watson;

alter role OrderEntry add member Emma;

**Screenshot:**

****

**Comments:**

The script a login with ID Watson and password ‘HarryPotter@1’ and sets the default database to AmazonClone database. The next statement creates user Emma for the login and assigns the user to the OrderEntry role.

1. **CONCLUSION**

The project helped in gaining real world experience in designing a database. From identifying tables and normalizing the database to improve the perfprmance of the database by using functions and procedures. The process of transforming a real world scenario into database model involves several intricate steps that gets better with each enhancement to the database.