Name: Heli Shah
SU ID: 733914949
Course: CSE 581 Introduction to Database Management System
Semester: Fall 2019

# PROJECT 2
# AMAZON COMPETITOR DATABASE

An Amazon-like database for it's competitor

## 1. ABSTRACT:

For the project 2,
The Amazon Competitor Database is created keeping in mind the functionality of Amazon and its day to day operations. The database contains various tables and there is a very complex relationship among them.
The purpose of this database is to bring a model for an Amazon competitor by bringing into design all the important aspects that a company like Amazon would need. The database contains critical information of a particular company's data and the data can be used in a number of ways as per the need of the company. This database is implemented in Azure Data Studio. This relational database that has been implemented provides more than enough relationship and tables to store relevant information for the company's growth and survival. The database takes care of the Order Management, Shipping, Warehousing and Stocking information along with the Customer and Product information.
However, the database is designed in a way that it is just not limited to the tables provided in here and can be upgraded to include more features. This is due to the ease of adding the referential integrity of SQL.The tables are created and the sample entries are already added to the tables for the better understanding of the structure.
In addition to that, store procedures, functions, triggers, column constraints while making the tables and making different views are created and implemented on this database. Certain **security features** are also added to the database. This database is all set to serve a company which is inline the operations of Amazon.

# 2. DESIGN:

## 2.1 INTRODUCTION:

In this project a database for an Amazon competitor is designed, tested and verified. The purpose of such a database is to obtain the necessary information a company in the online-store field would require. Firstly, the database is designed with a number of essential features and then built upon by adding relevant tables, tables that add critical information to the database. It is ensured that a database with a number of entities should be normalized. Here all these entities provide the vital information to construct a logistics database. This competitor will be able to test the database with a number of T-SQL commands and various views and queries. The main purpose to design a proper database is to ensure that it is normalized and the entities are not directly dependent on each other, which indeed is a very crucial factor. The database for the competitor has a number of relationships including one-to-many, many-to-many, many-to-one and one-to-one. All these relationships provide necessary information and tell how the data is to be stored. Use of such a database increases the efficiency of the business operation and reduces overall costs. The database uses the perfect schema to allow updating and viewing data at ease. Also, one of the reasons why this design supersedes other databases is its security information. Moreover, the database design ensures that redundant information is not stored in it. When less redundant information is stored on the database the database efficiency increases by speed and storage. More the data stored in the memory more difficult it would become to access data. A unique feature of this database is that it gives vital information for each and every entity and its attributes. The design is made in such a way that any important attribute can be added to the particular entity easily using SQL server. Since this database has been designed in SQL server data retrieval and data access have never become easier since. Queries used here are powerful tools that provide information instantly. In this database most of the entities one can look for in the substantial growth of the company has been identified and entered and also newer entities can be added when necessary due the primary key and foreign key constraints this database has provided. The Business problem in this industry is uniquely identified in each domain and then the tables have been constructed to solve these problems. A reason would be to make the database more understandable for the user. Overall the database has been carefully designed, implemented and tested to tackle most of the business problems an Amazon competitor would endure.

## 2.2 DESIGN EXPLAINATION:

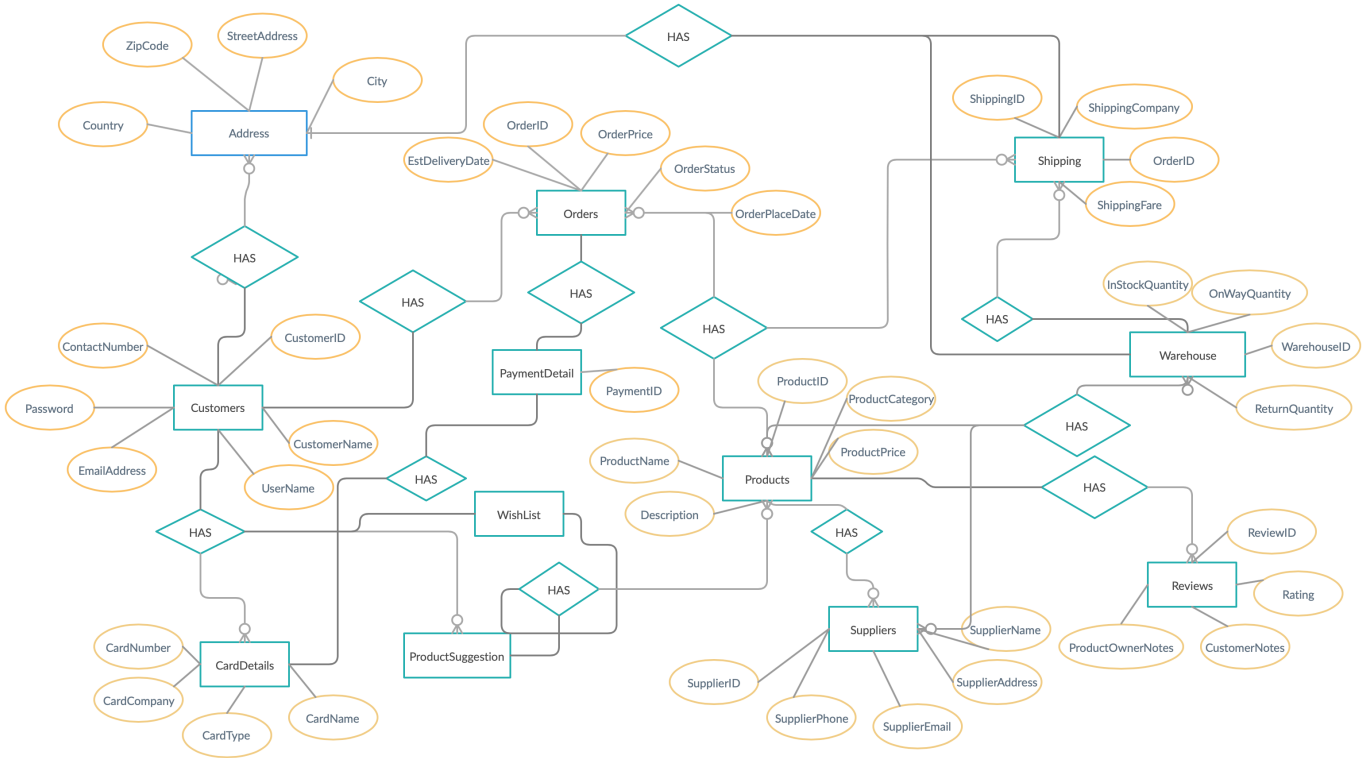
The database has been designed by taking into account a number of considerations to solve the given business problem. There are a total of 17 tables taken into consideration while addressing a company like Amazon. There are many relationships among the tables: one-to-one, one-to-many and many-to-many. These relationships can be clearly seen in the Entity-Relationship diagram below. the testing is done using a number of SQL queries and T-SQL commands such as **stored procedures**, **functions**, **triggers** and also with views. There are many constraints that are taken care of while designing the tables like columns having NOT NULL so that there is an entry mandatorily, providing default values to some columns in case values are not added. Each table record is additionally identified uniquely by its Primary Key. A number of Foreign Key constraints have been added to provide a proper data flow between the related tables. There is no redundancy in the tables which is ensured as the database is reduced to its 3rd normal form. Thus, it is also ensured that all tables have only one primary key and the foreign key is related only to the primary key of another table.

This database contains the personal information of customers (or the account holders) like their name, address, contact information like email, phone number and address and also their payment method details. The address of the customers is whole new entity that has the Street address, City, State and Zip Code. The card details contain the Card number, Card Company, Type and name on the card. The customers personal information also contains the login information of the customer like username and password. The other important aspect of the database is the Orders. This entity contains OrderID, OrderStatus, OrderPrice, OrderPlaceDate, EstimatedDeliveryDate. This entity is related to the customers as customers have many orders. Order is also related to the PaymentDetails entity which is in turn related to the CardDetails entity. Customers have a WishList and ProductSuggestion entity. Both of them contains the product information in the respective details. Another important entity introduced here is the Products. This products entity contains the ProductID, ProductName, Description, ProductCategory and Product Price. It is related to Orders entity in a many-to-many manner. Two important entities that are added here are Suppliers and Warehouse. The Suppliers entity has attributes like ID, Contact, Address and is related to the product in a many-to-many way. The reason for this is one product may be supplied by more than one supplier and one supplier can deliver more than one type of product. Coming to Warehouse, it is related to the products with the exact same relation as Suppliers. It just contains the entities: ID, InStock Quantity of products, OnTheWay to deliver the products and the return products. One more relation is

between Warehouse and entity Shipping. It is linked in a way that one warehouse unit is connected to many shipping units. The Shipping unit is in turn related to the Orders unit where one order can have more than one shipping label (if the items are being shipped in many different parts). There is one more entity of reviews that is related to the Products entity in a way that each product can have many reviews.

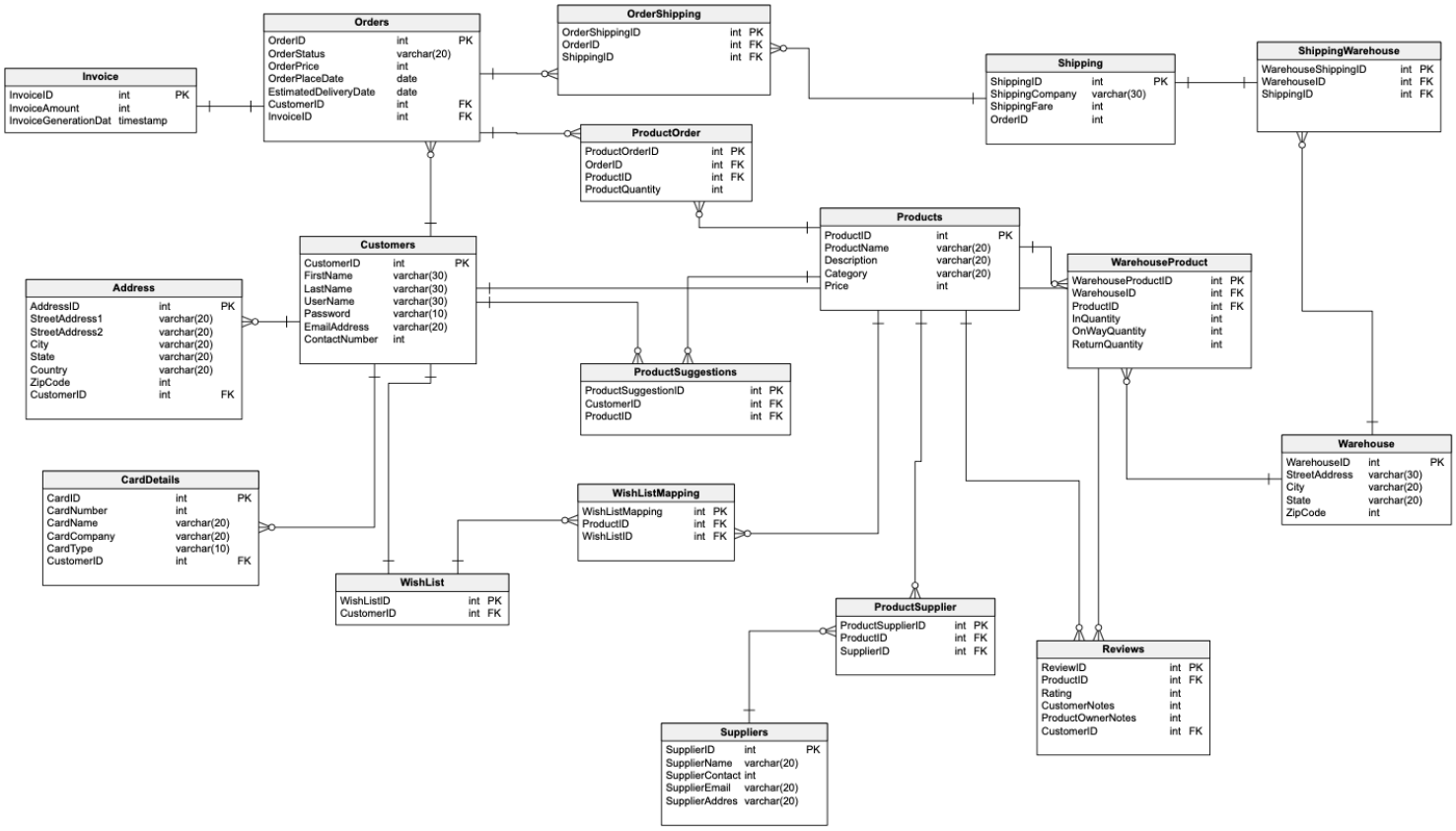
## 2.3 ENTITY RELATIONSHIP DESIGN:

This whole relationship is described through ER diagram which is posted below:




This Entity Relationship diagram depicts all the relationships among the entities and also the attributes of each of the entities.

## 2.4 DATABASE DESIGN:




This Database Design Diagram is reduced to its 3rd Normal form and so every reference is made to only the primary key of another table. Many-to-many relationships are taken care of and so are the redundancies.

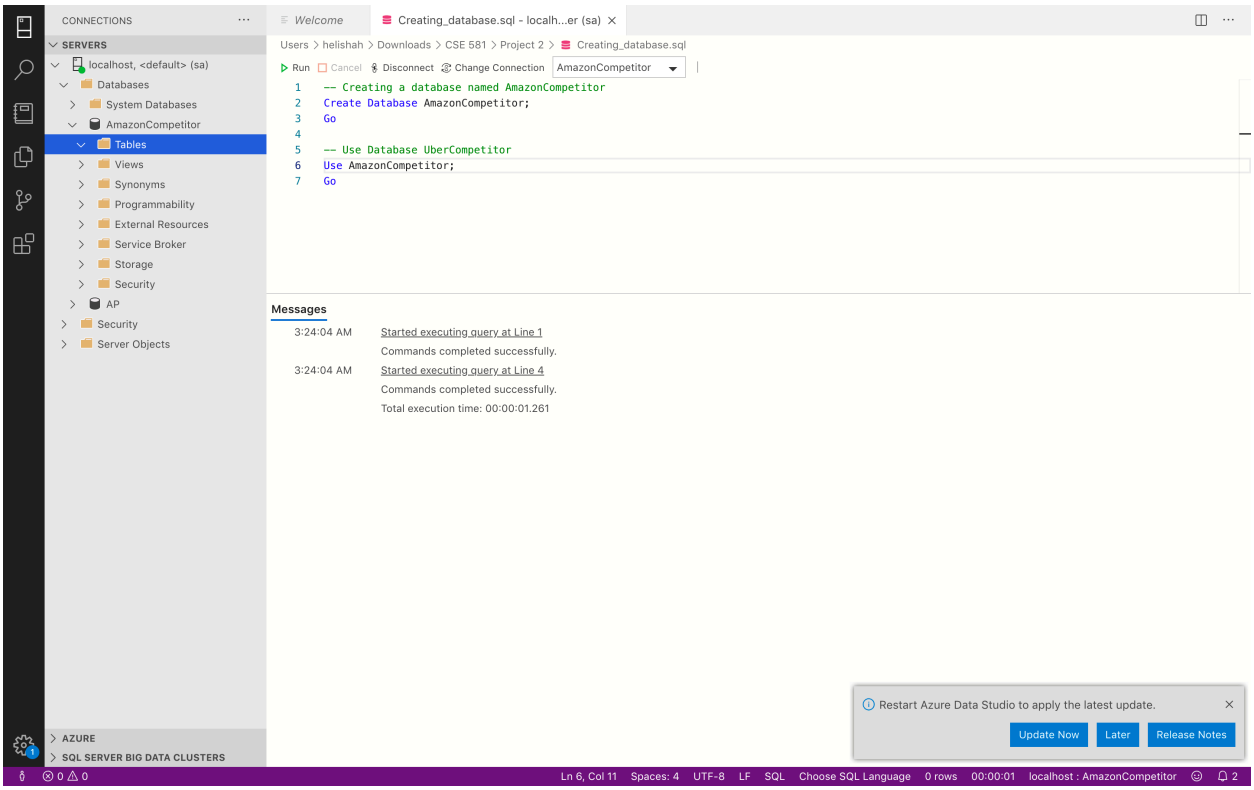
# 3. IMPLEMENTATION:

In the actual implementation of the above mentioned design for the Amazon Competitor Database is carried out in Azure Data Studio. A new database is created and all the tables are defined with it. After that, all the tabled are fed with 10 records each so as to make this database feel as an actual one and perform testing on it.

For each implementation, respective code for explanation is given with the query and the output is also shown at needed places. All this is included in the screenshots below:

## 3.1 Creation of Database:

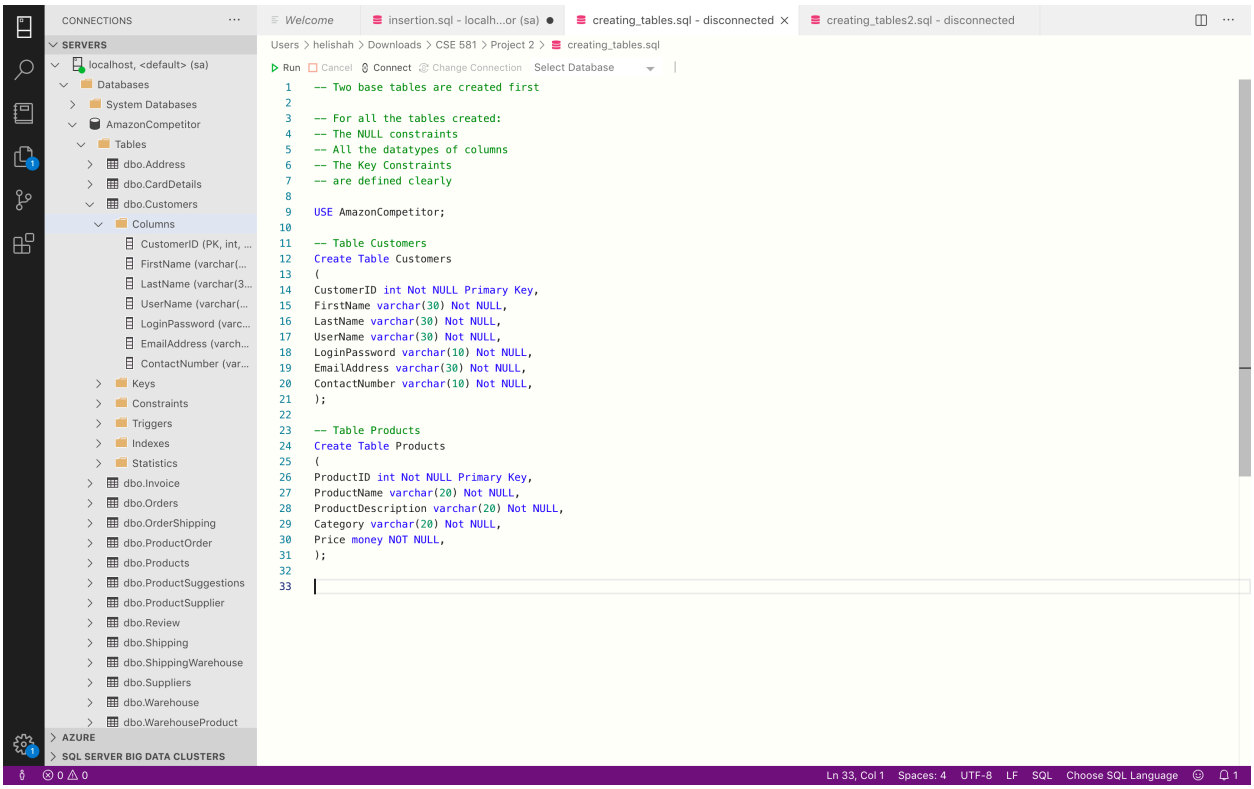
The database named AmazonCompetitor is created and then used the same.



## 3.2 Creating the Tables:

The necessary tables are created by using the ‘Create Table’ command here the necessary constraints, primary key and foreign key are provided. Also, all the nullability constraint for each table is included.

A. First, the base tables around which the whole database revolves, are made: Customers and Products



B. Creating all the remaining 16 tables keeping in mind their constraints and Foreign References:

The tables created follow a set of **constraints** and some foreign references. They are all depicted in this SQL code:

```
--
 All other tables are created now and given thier respective ref
erences according to the design

-- For all the tables created:
-- The NULL constraints
-- All the datatypes of columns
-- The Key Constraints
-- are defined clearly

USE AmazonCompetitor;

-- Table Address for storing Customer's Address
Create Table Address
(
AddressID int Not NULL Primary Key,
CustomerID int Not NULL Foreign Key References Customers(Custome
rID),
StreetAddress1 varchar(30) Not NULL,
StreetAddress2 varchar(30) Not NULL,
City varchar(30) Not NULL,
AddressState varchar(10) Not NULL,
Country varchar(30) Not NULL,
ZipCode int Not NULL,
);

-- Table CardDetails for storing Customer's Payment Information
Create Table CardDetails
(
AddressID int Not NULL Primary Key,
```

```
CustomerID int Not NULL Foreign Key References Customers(Custome
rID),
StreetAddress1 varchar(30) Not NULL,
StreetAddress2 varchar(30) Not NULL,
City varchar(30) Not NULL,
AddressState varchar(10) Not NULL,
Country varchar(30) Not NULL,
ZipCode varchar(5) Not NULL,
);

-- Table Invoice for storing Invoice Information for an Order
Create Table Invoice
(
InvoiceID int Not NULL Primary Key,
InvoiceAmount money Not NULL,
InvoiceGenerationDate date Not NULL,
);

-- Table Orders for storing Customer's Order Information
Create Table Orders
(
OrderID int Not NULL Primary Key,
CustomerID int Not NULL Foreign Key References Customers(Custome
rID),
OrderStatus varchar(20) Not NULL,
OrderPrice money Not NULL,
OrderPlaceDate date Not NULL,
EstDeliveryDate date Not NULL,
InvoiceID int Not NULL Foreign Key References Invoice(InvoiceID)
,
);

--
 Table WishList for storing Customer's WishList Products Informa
tion
Create Table WishList
```

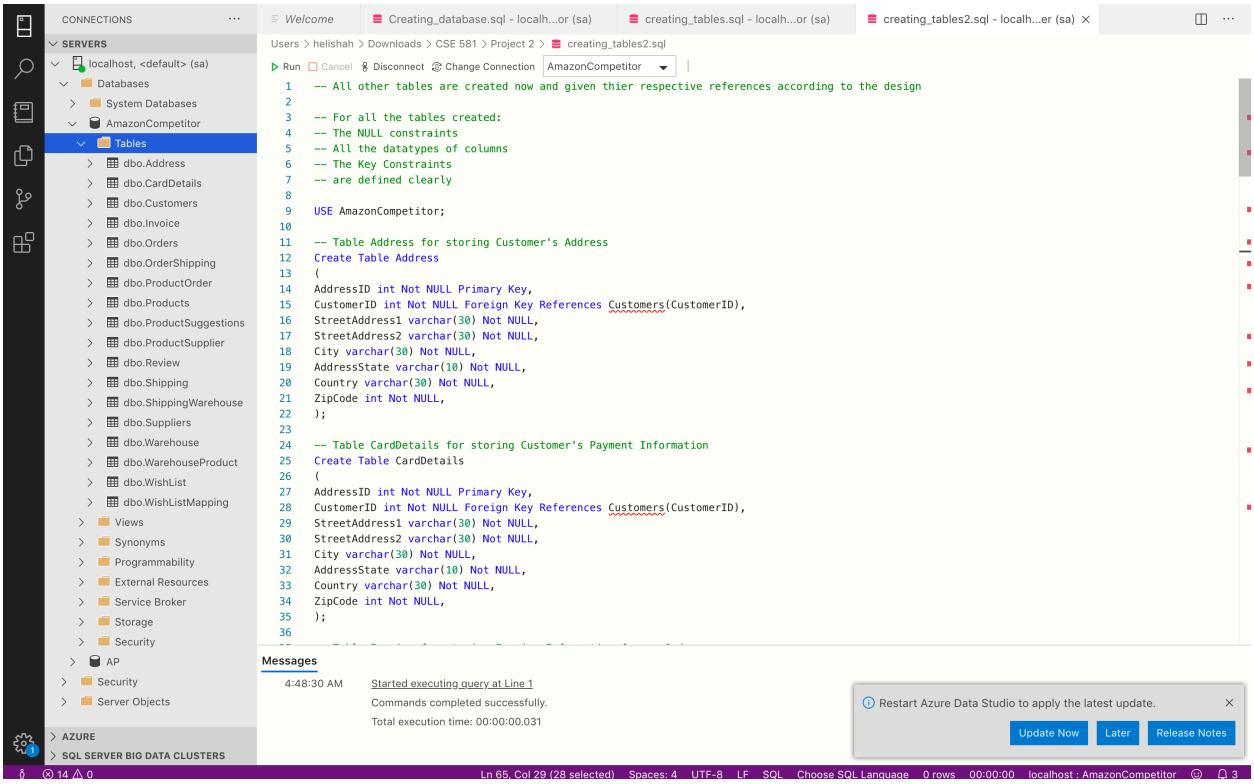
```
(
WishListID int Not NULL Primary Key,
CustomerID int Not NULL Foreign Key References Customers(Custome
rID),
);

--
 Table WishListMapping for mapping Customer's WishList Informati
on to Product Information
Create Table WishListMapping
(
WishListMappingID int Not NULL Primary Key,
WishListID int Not NULL Foreign Key References WishList(WishList
ID),
ProductID int Not NULL Foreign Key References Products(ProductID
),
);

-- Table ProductSuggestions for storing the mapping of Product-
Customer
-- According to the Products suggested to the Customer
Create Table ProductSuggestions
(
ProductSuggestionID int Not NULL Primary Key,
CustomerID int Not NULL Foreign Key References Customers(Custome
rID),
ProductID int Not NULL Foreign Key References Products(ProductID
),
);

--
 Table Shipping for storing the shipping Information for an Orde
r
Create Table Shipping
(
ShippingID int Not NULL Primary Key,
```

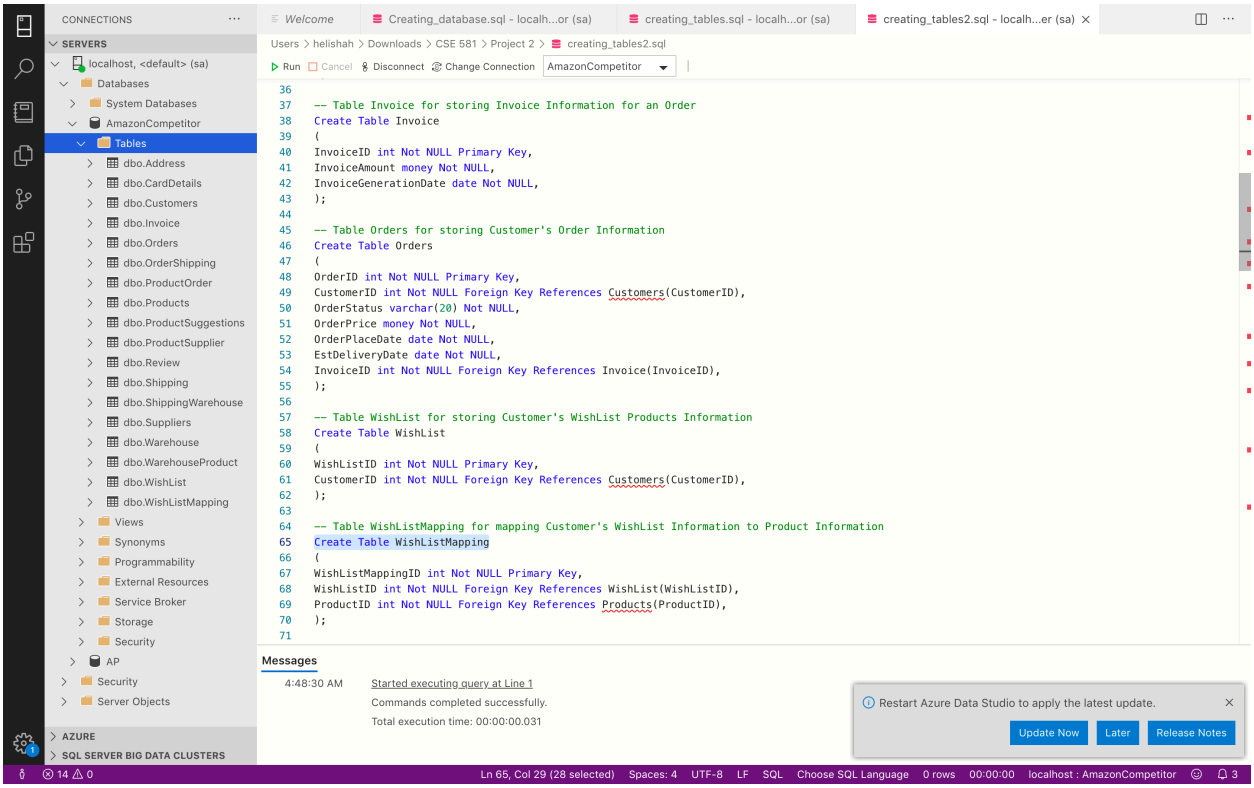
```
ShippingCompany varchar(30) Not NULL,
ShippingFare money Not NULL,
OrderID int Not NULL Foreign Key References Orders(OrderID),
);

--
 Table ProductOrder for mapping the Product Information for an O
rder
-- There can be
Create Table ProductOrder
(
ProductOrderID int Not NULL Primary Key,
OrderID int Not NULL Foreign Key References Orders(OrderID),
ProductID int Not NULL Foreign Key References Products(ProductID
),
ProductQuantity int Not NULL,
);

--
 Table OrderShipping for mapping the Shipping Information for an
 Order
-- There can be more than one Shipping for just one Order
Create Table OrderShipping
(
ProductOrderID int Not NULL Primary Key,
OrderID int Not NULL Foreign Key References Orders(OrderID),
ProductID int Not NULL Foreign Key References Products(ProductID
),
);

--
 Table Reviews for storing the reviews for a particular Product
-- There can be more than one reviwes for just one Product
--
 The comments can be by both concerned parties: the customer and
 the product owner
```

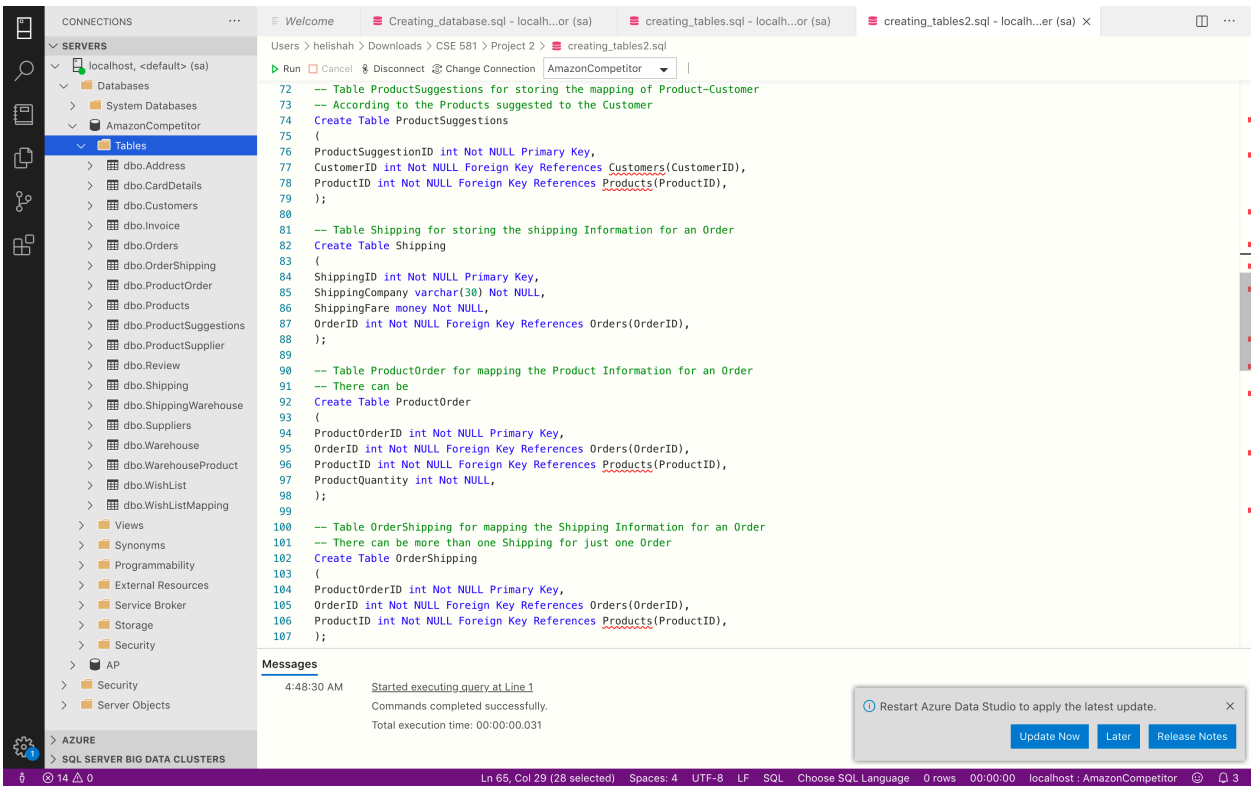
```
Create Table Review
(
ReviewID int Not NULL Primary Key,
CustomerID int Not NULL Foreign Key References Orders(OrderID),
ProductID int Not NULL Foreign Key References Products(ProductID
),
Rating int Not NULL,
CustomerNotes varchar(30),
ProductOwnerNotes varchar(30),
);

--
 Table Suppliers for storing the Information about the Suppliers
 for a particular Product
Create Table Suppliers
(
SupplierID int Not NULL Primary Key,
SupplierName varchar(30) Not NULL,
SupplierContact int Not NULL,
SupplierEmail varchar(30) Not NULL,
SupplierAddress varchar(50) Not NULL,
);

--
 Table ProductSupplier for mapping the Suppliers to the Products
-- Many-to-Many relationship
Create Table ProductSupplier
(
ProductSupplierID int Not NULL Primary Key,
SupplierName varchar(30) Not NULL,
ProductID int Not NULL Foreign Key References Products(ProductID
),
SupplierID int Not NULL Foreign Key References Suppliers(Supplie
rID),
);
```

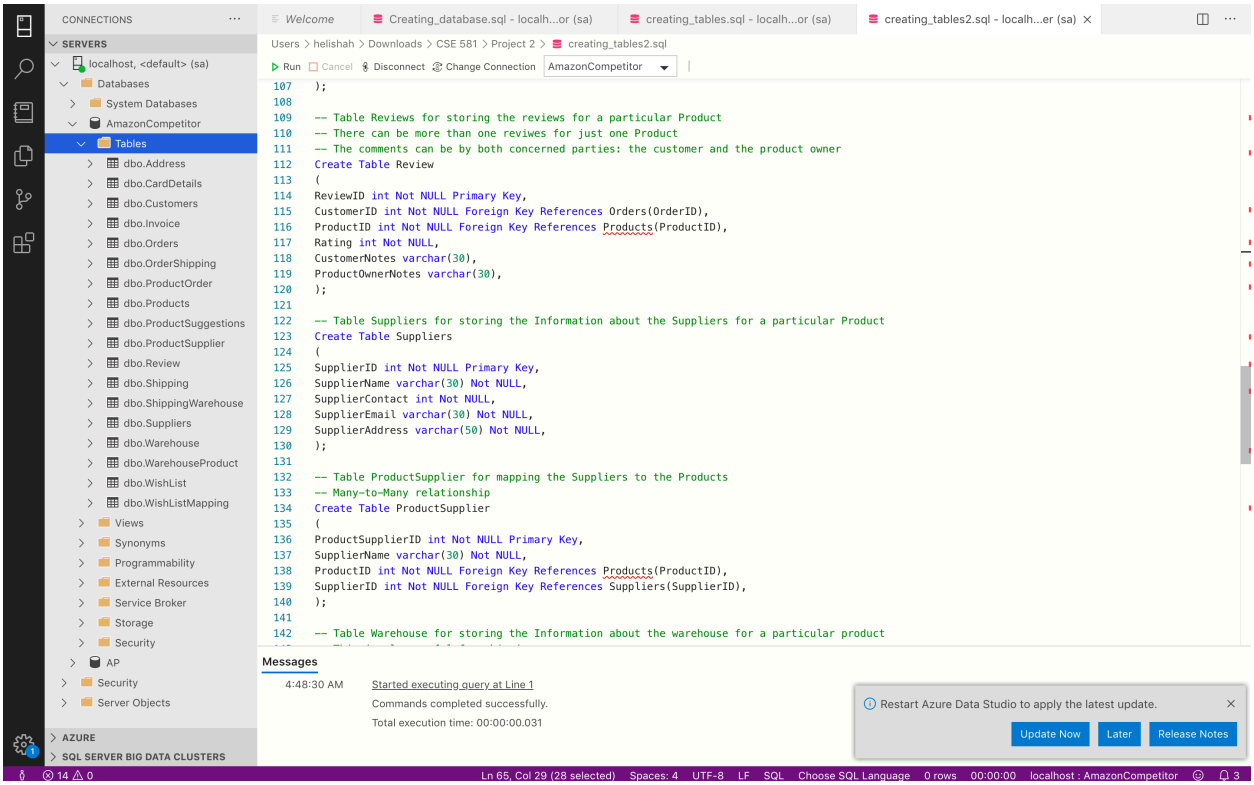
```
--
 Table Warehouse for storing the Information about the warehouse
 for a particular product
-- This is also useful for shipping
Create Table Warehouse
(
WarehouseID int Not NULL Primary Key,
StreetAddress varchar(30) Not NULL,
City varchar(30) Not NULL,
WarehouseState varchar(30) Not NULL,
ZipCode int Not NULL,
);

--
 Table WarehouseProduct for mapping the products to a particular
 warehouse
-- Same products maybe in multiple warehouses
Create Table WarehouseProduct
(
WarehouseProductID int Not NULL Primary Key,
WarehouseID int Not NULL Foreign Key References Warehouse(Wareho
useID),
ProductID int Not NULL Foreign Key References Products(ProductID
),
ProductInQuantity int Not NULL,
ProductOnWayQuantity int Not NULL,
ReturnQuantity int Not NULL,
);

--
 Table ShippingWarehouse for mapping the shipping order to a par
ticular warehouse
-- Same orders may have different shipping
-- But, same shipping must have same warehouses
Create Table ShippingWarehouse
(
```

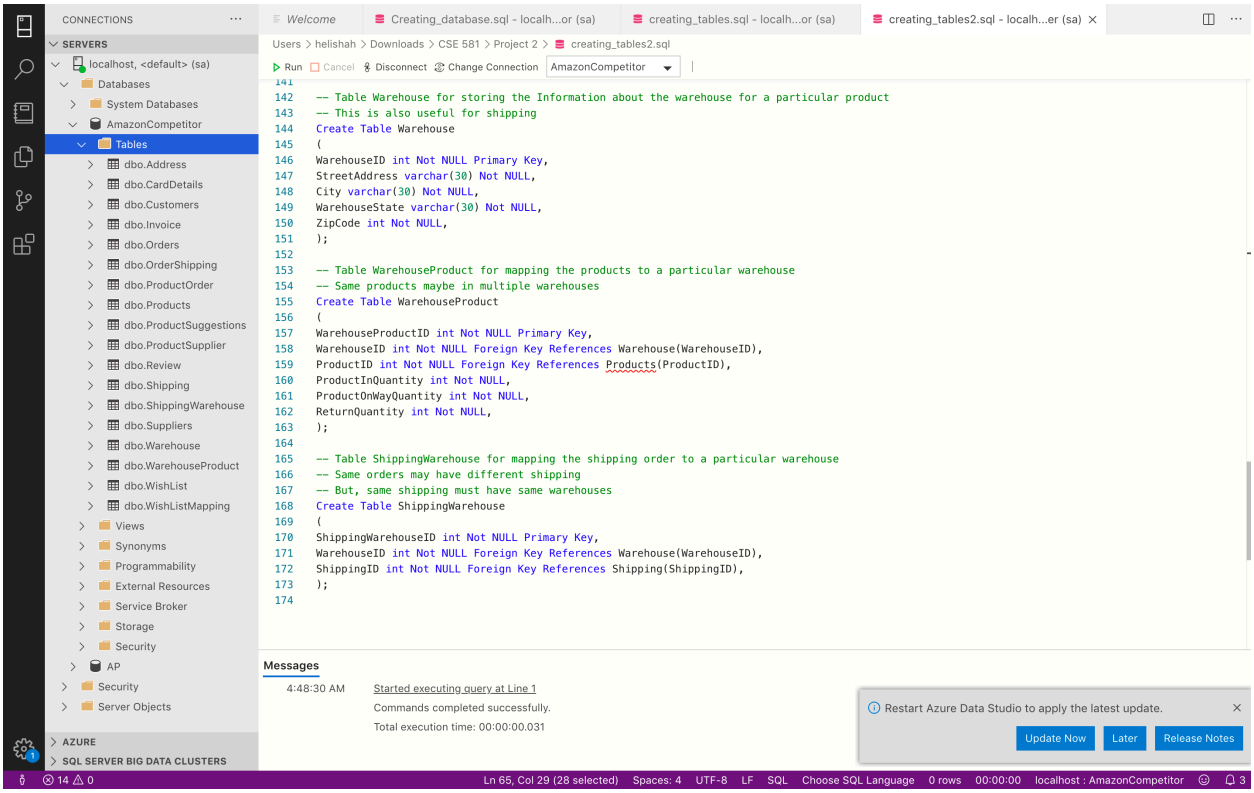
```
ShippingWarehouseID int Not NULL Primary Key,
WarehouseID int Not NULL Foreign Key References Warehouse(Wareho
useID),
ShippingID int Not NULL Foreign Key References Shipping(Shipping
ID),
);
```




CONNECTIONS
SERVERS
localhost, <default> (sa)
Databases
System Databases
AmazonCompetitor
Tables
dbo.Address
dbo.CardDetails
dbo.Customers
dbo.Invoice
dbo.Orders
dbo.OrderShipping
dbo.ProductOrder
dbo.Products
dbo.ProductSuggestions
dbo.ProductSupplier
dbo.Review
dbo.Shipping
dbo.ShippingWarehouse
dbo.Suppliers
dbo.Warehouse
dbo.WarehouseProduct
dbo.WishList
dbo.WishListMapping
Views
Synonyms
Programmability
External Resources
Service Broker
Storage
Security
AP
Security
Server Objects
AZURE
SQL SERVER BIG DATA CLUSTERS
Welcome
Creating_database.sql - localh...or (sa)
creating_tables.sql - localh...or (sa)
creating_tables2.sql - localh...er (sa)
Users > helishah > Downloads > CSE 581 > Project 2 > creating_tables2.sql
Run
Cancel
Disconnect
Change Connection
AmazonCompetitor
36
37 -- Table Invoice for storing Invoice Information for an Order
38 Create Table Invoice
39 (
40 InvoiceID int Not NULL Primary Key,
41 InvoiceAmount money Not NULL,
42 InvoiceGenerationDate date Not NULL,
43 );
44
45 -- Table Orders for storing Customer's Order Information
46 Create Table Orders
47 (
48 OrderID int Not NULL Primary Key,
49 CustomerID int Not NULL Foreign Key References Customers(CustomerID),
50 OrderStatus varchar(20) Not NULL,
51 OrderPrice money Not NULL,
52 OrderPlaceDate date Not NULL,
53 EstDeliveryDate date Not NULL,
54 InvoiceID int Not NULL Foreign Key References Invoice(InvoiceID),
55 );
56
57 -- Table WishList for storing Customer's WishList Products Information
58 Create Table WishList
59 (
60 WishListID int Not NULL Primary Key,
61 CustomerID int Not NULL Foreign Key References Customers(CustomerID),
62 );
63
64 -- Table WishListMapping for mapping Customer's WishList Information to Product Information
65 Create Table WishListMapping
66 (
67 WishListMappingID int Not NULL Primary Key,
68 WishListID int Not NULL Foreign Key References WishList(WishListID),
69 ProductID int Not NULL Foreign Key References Products(ProductID),
70 );
71
Messages
4:48:30 AM
Started executing query at Line 1
Commands completed successfully.
Total execution time: 00:00:00.031
Restart Azure Data Studio to apply the latest update.
Update Now
Later
Release Notes
14 0
Ln 65, Col 29 (28 selected)
Spaces: 4
UTF-8
LF
SQL
Choose SQL Language
0 rows
00:00:00
localhost : AmazonCompetitor
3


CONNECTIONS
SERVERS
localhost, <default> (sa)
Databases
System Databases
AmazonCompetitor
Tables
dbo.Address
dbo.CardDetails
dbo.Customers
dbo.Invoice
dbo.Orders
dbo.OrderShipping
dbo.ProductOrder
dbo.Products
dbo.ProductSuggestions
dbo.ProductSupplier
dbo.Review
dbo.Shipping
dbo.ShippingWarehouse
dbo.Suppliers
dbo.Warehouse
dbo.WarehouseProduct
dbo.WishList
dbo.WishListMapping
Views
Synonyms
Programmability
External Resources
Service Broker
Storage
Security
AP
Security
Server Objects
AZURE
SQL SERVER BIG DATA CLUSTERS
Welcome
Creating_database.sql - localh...or (sa)
creating_tables.sql - localh...or (sa)
creating_tables2.sql - localh...er (sa)
Users > helishah > Downloads > CSE 581 > Project 2 > creating_tables2.sql
Run
Cancel
Disconnect
Change Connection
AmazonCompetitor
72 -- Table ProductSuggestions for storing the mapping of Product-Customer
73 -- According to the Products suggested to the Customer
74 Create Table ProductSuggestions
75 (
76 ProductSuggestionID int Not NULL Primary Key,
77 CustomerID int Not NULL Foreign Key References Customers(CustomerID),
78 ProductID int Not NULL Foreign Key References Products(ProductID),
79 );
80
81 -- Table Shipping for storing the shipping Information for an Order
82 Create Table Shipping
83 (
84 ShippingID int Not NULL Primary Key,
85 ShippingCompany varchar(30) Not NULL,
86 ShippingFare money Not NULL,
87 OrderID int Not NULL Foreign Key References Orders(OrderID),
88 );
89
90 -- Table ProductOrder for mapping the Product Information for an Order
91 -- There can be
92 Create Table ProductOrder
93 (
94 ProductOrderID int Not NULL Primary Key,
95 OrderID int Not NULL Foreign Key References Orders(OrderID),
96 ProductID int Not NULL Foreign Key References Products(ProductID),
97 ProductQuantity int Not NULL,
98 );
99
100 -- Table OrderShipping for mapping the Shipping Information for an Order
101 -- There can be more than one Shipping for just one Order
102 Create Table OrderShipping
103 (
104 ProductOrderID int Not NULL Primary Key,
105 OrderID int Not NULL Foreign Key References Orders(OrderID),
106 ProductID int Not NULL Foreign Key References Products(ProductID),
107 );
Messages
4:48:30 AM
Started executing query at Line 1
Commands completed successfully.
Total execution time: 00:00:00.031
Restart Azure Data Studio to apply the latest update.
Update Now
Later
Release Notes
14 0
Ln 65, Col 29 (28 selected)
Spaces: 4
UTF-8
LF
SQL
Choose SQL Language
0 rows
00:00:00
localhost : AmazonCompetitor
3




With this, the implementation of our database is completed and now we shall go to the phase of testing the database.

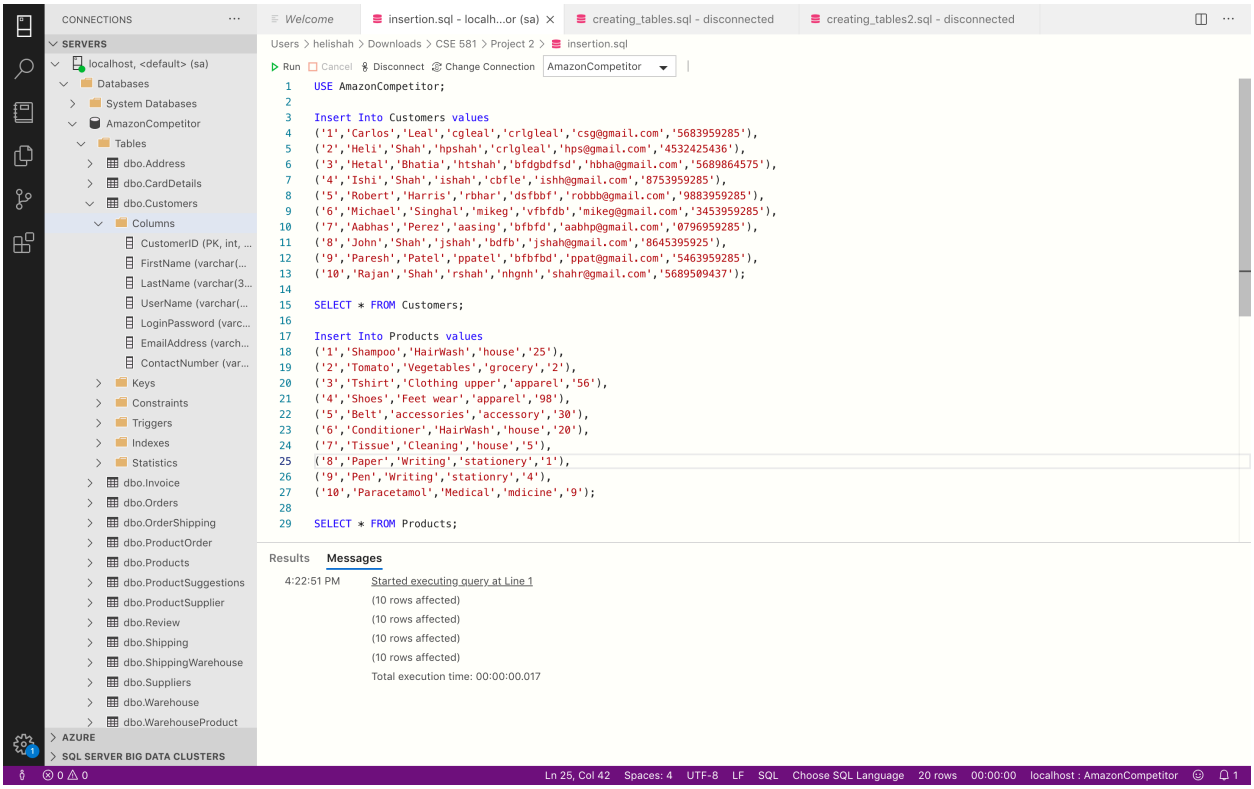
# 4. TESTING THE DATABASE:

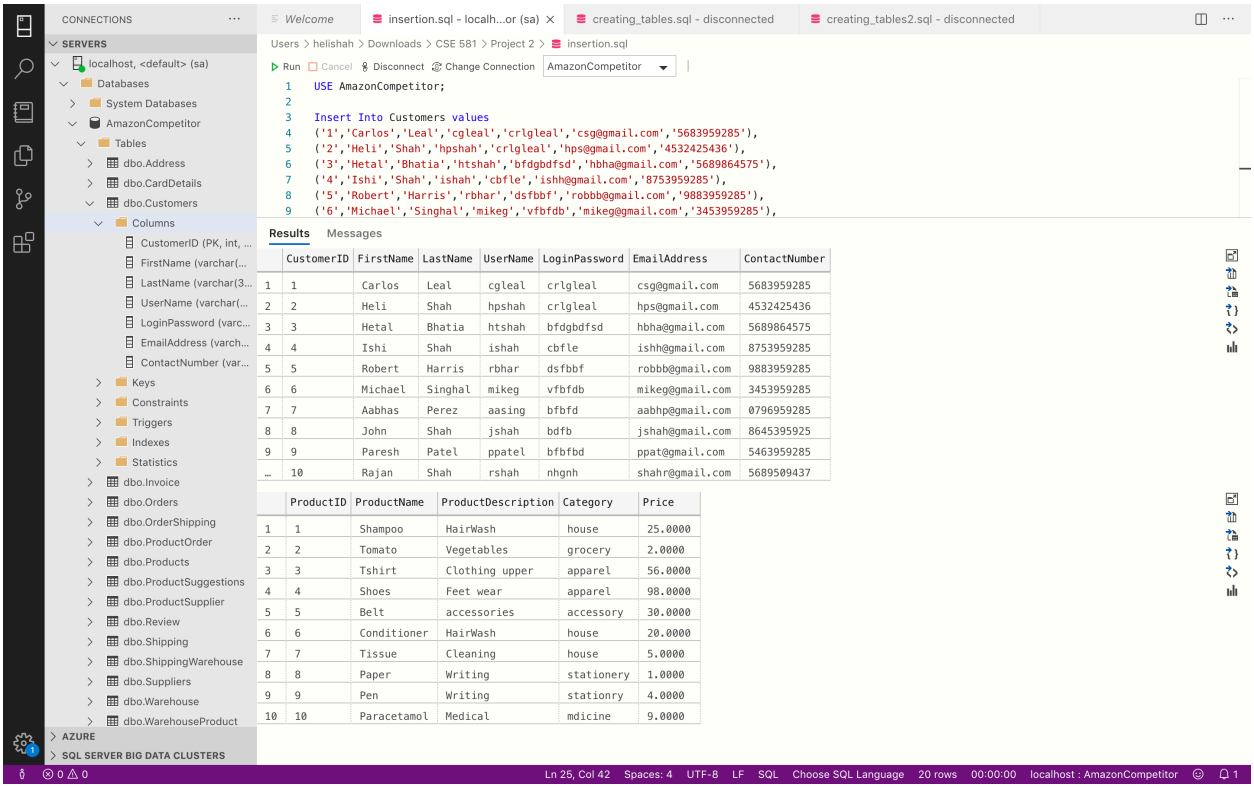
## 4.1 Inserting Values:

Inserting into Customers and Products:

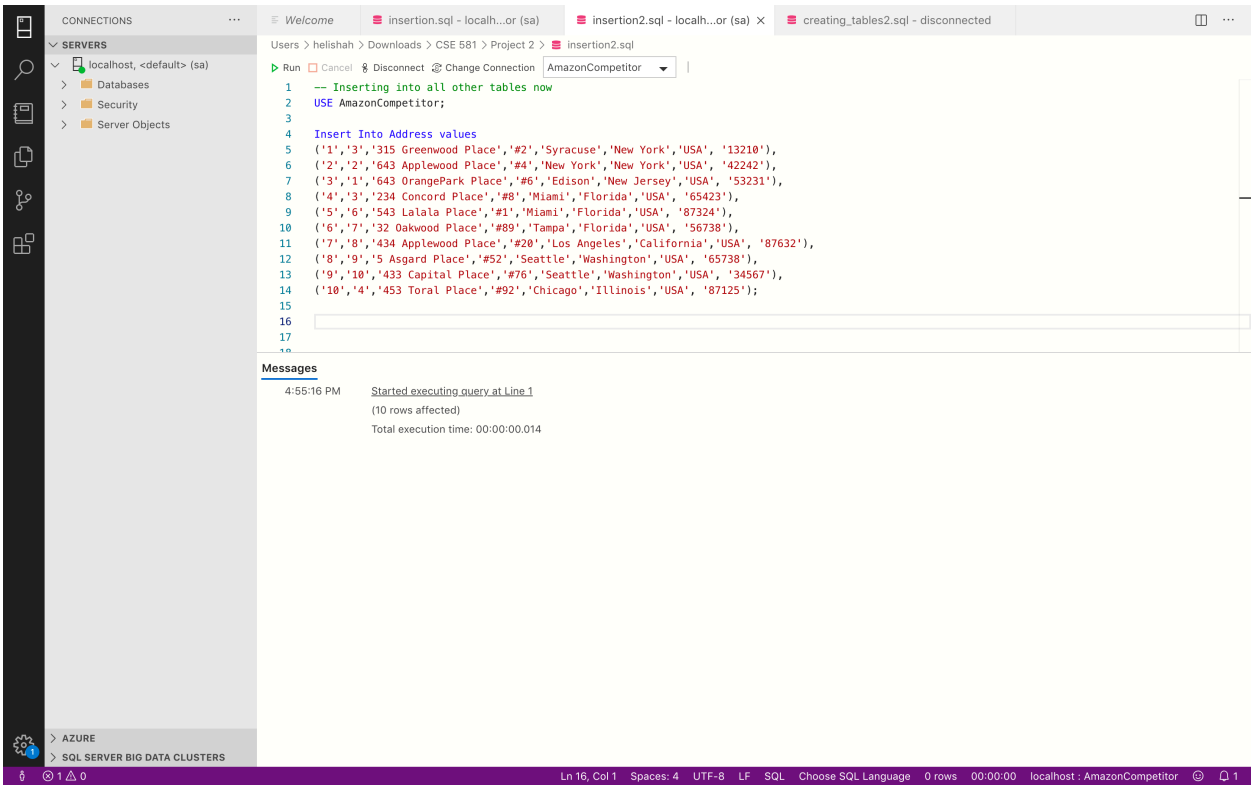
```
USE AmazonCompetitor;
Insert Into Customers values
('1','Carlos','Leal','cgleal','crlgleal','csg@gmail.com','5683959285')
,
('2','Heli','Shah','hpshah','crlgleal','hps@gmail.com','4532425436'),
('3','Hetal','Bhatia','htshah','bfdgbdfsd','hbha@gmail.com','568986457
85'),
('4','Ishi','Shah','ishah','cbfle','ishh@gmail.com','8753959285'),
('5','Robert','Harris','rbhar','dsfbbf','robbb@gmail.com','9883959285'
),
('6','Michael','Singhal','mikeg','vfbfdb','mikeg@gmail.com','345395928
5'),
('7','Aabhas','Perez','aasing','bfbfd','aabhp@gmail.com','0796959285')
,
('8','John','Shah','jshah','bdfb','jshah@gmail.com','86453959285'),
('9','Paresh','Patel','ppatel','bfbfbd','ppat@gmail.com','5463959285')
,
('10','Rajan','Shah','rshah','nhgnh','shahr@gmail.com','56895094367');

Insert Into Products values
('1','Shampoo','HairWashShampoo','house','25'),
('2','Tomato','HairWashShampoo','grocery','2'),
('3','Tshirt','HairWashShampoo','apparel','56'),
('4','Shoes','HairWashShampoo','apparel','98'),
('5','Belt','HairWashShampoo','accessory','30'),
('6','Conditioner','HairWashShampoo','house','20'),
('7','Tissue','HairWashShampoo','house','5'),
('8','Paper','HairWashShampoo','stationery','1'),
('9','Pen','HairWashShampoo','stationry','4'),
('10','Paracetamol','HairWashShampoo','mdicine','9');
```

## Insertion in Products and Customers results:

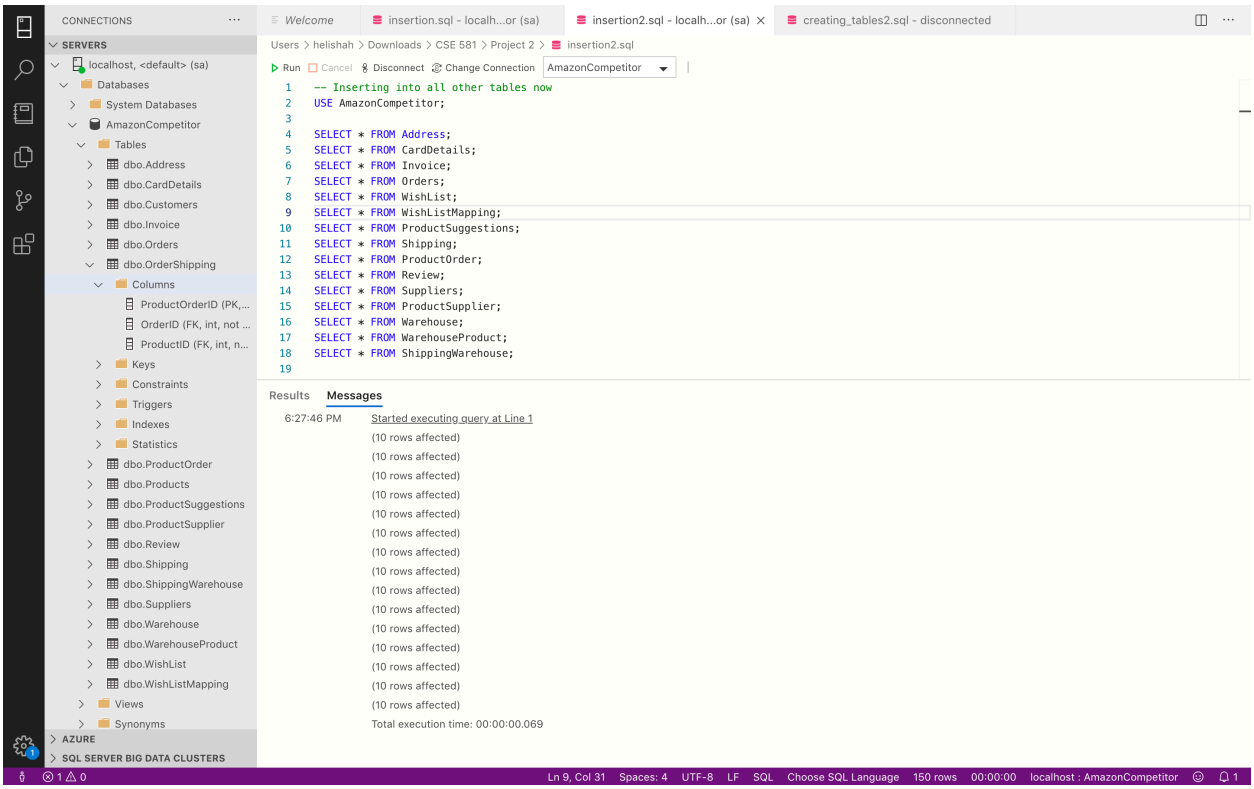


The insert statement format for a table:

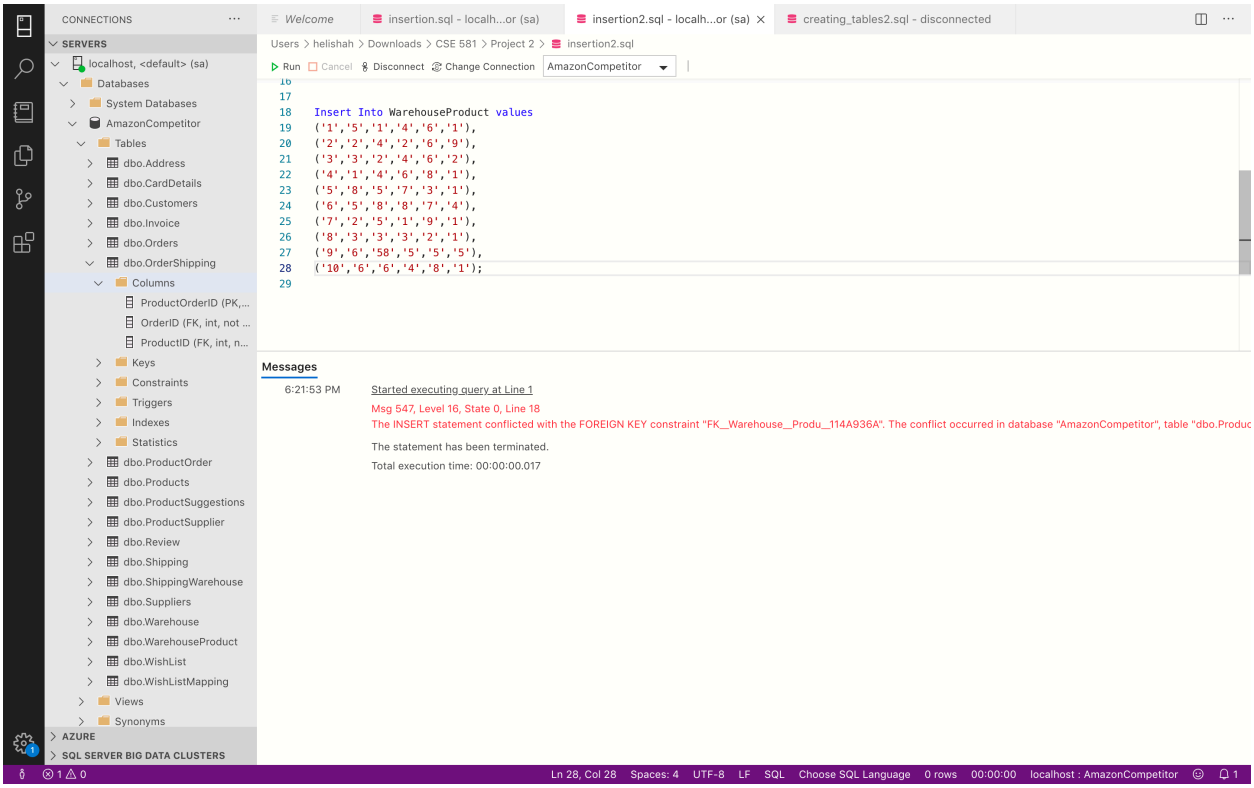



Inserting values into all other remaining tables:



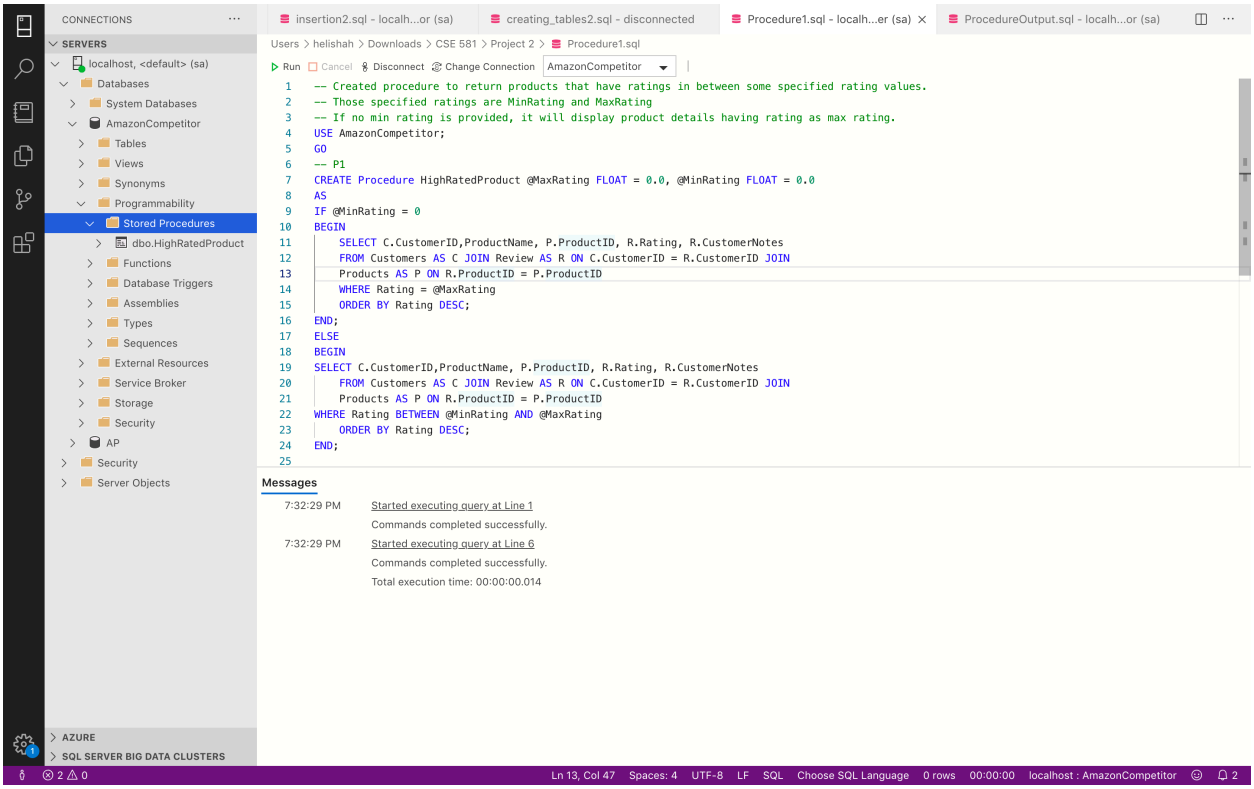
While implementing the database tables, the Foreign Ket constraints have been have been implemented correctly. Here is a proof that where the constraints are enforced, SQL won't allow any incorrect value:

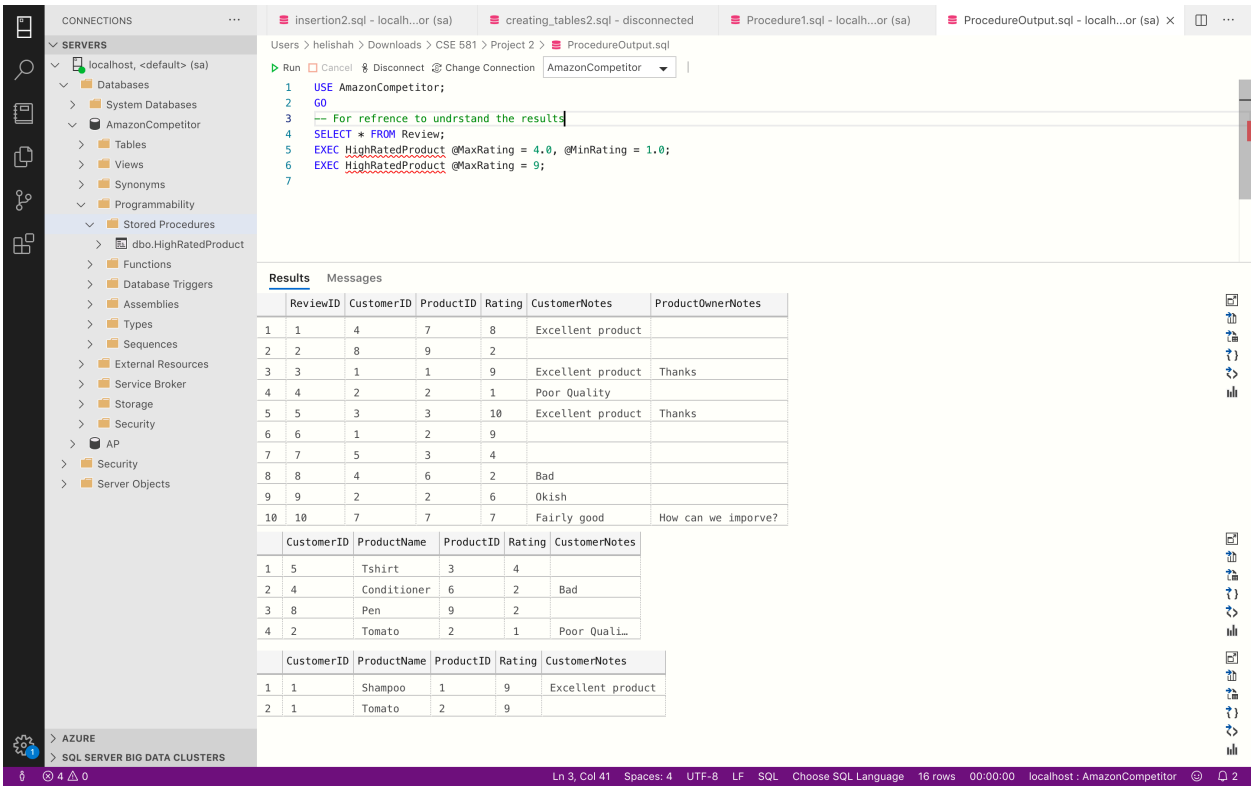
Here, for the table WarehouseProduct, the third Column is referenced to ProductID of Products table and '58' is not an entry of any ProductID in Products table (can refer the above screenshot of Products table entries). So, it gave the error for Foreign Key constraint.



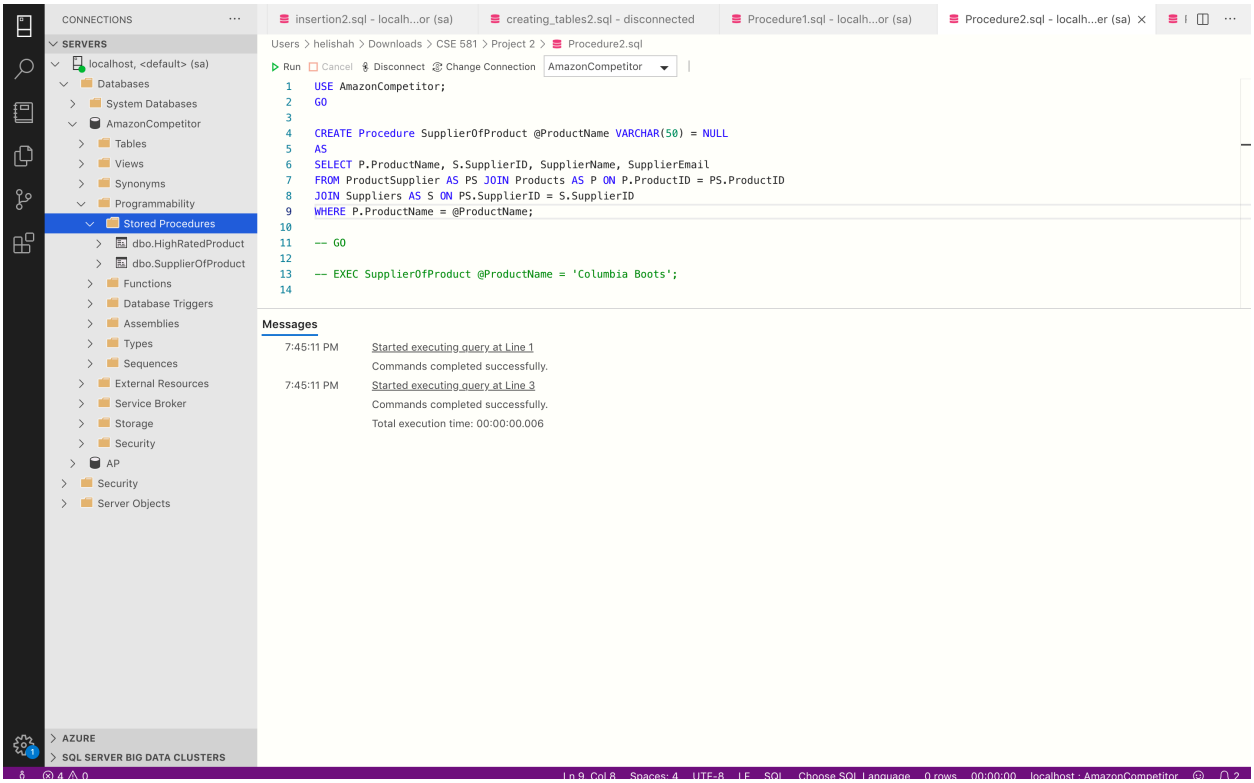
## 4.2 Stored Procedures:

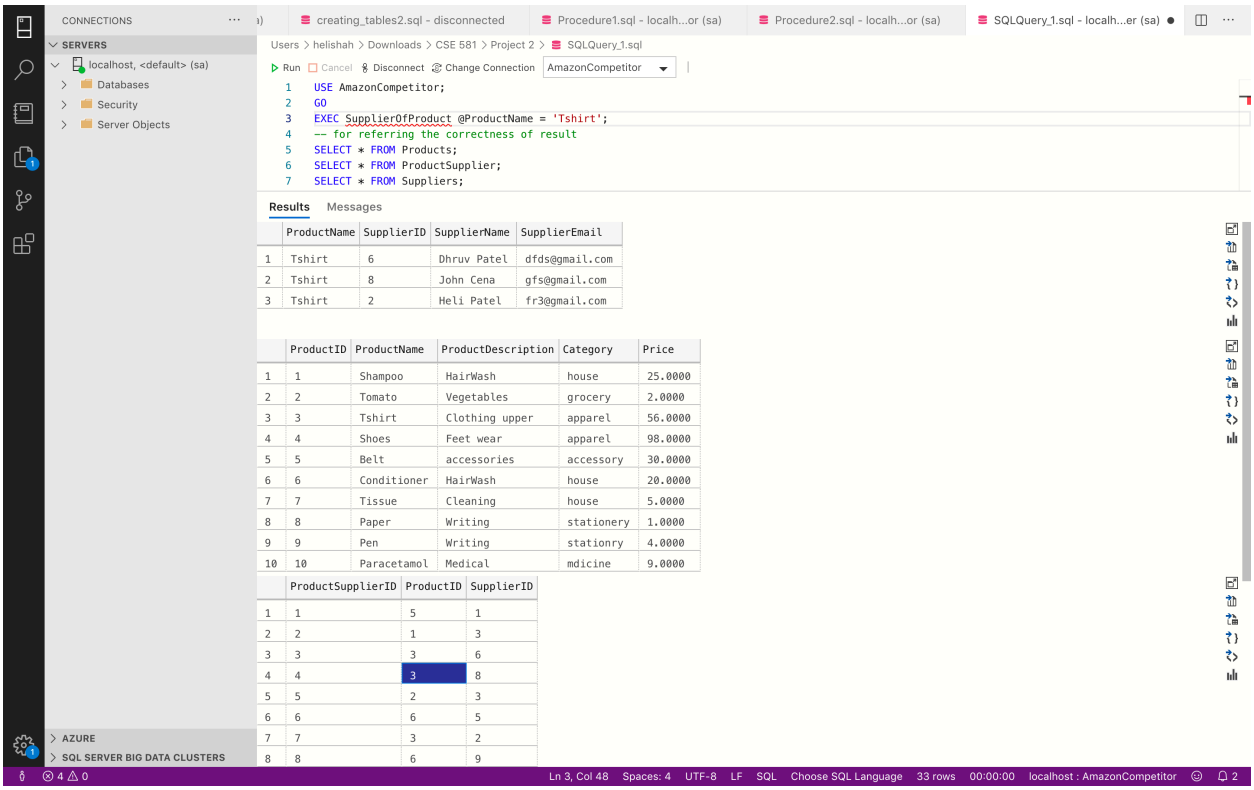
### 4.2.1 This procedure gives us the products rated between the max and min rating provided. While executing, the max or min or both ratings are provided.



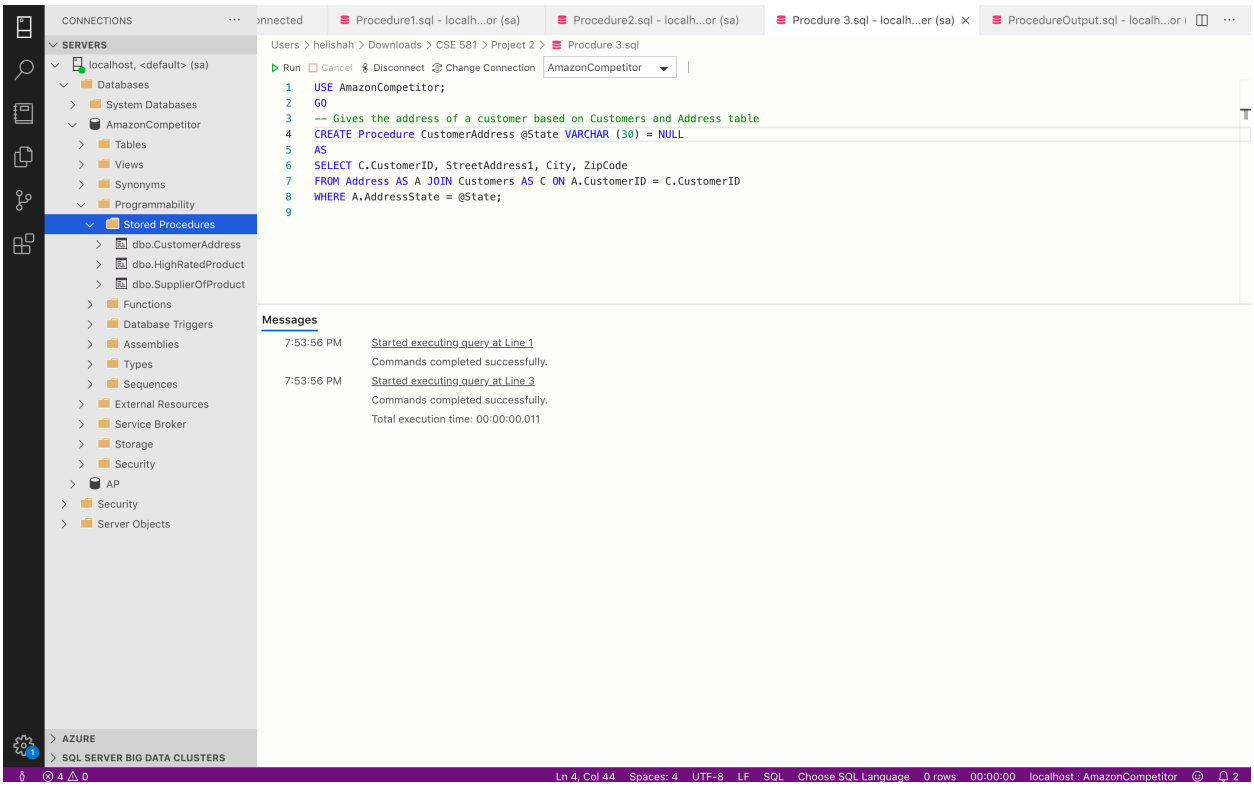



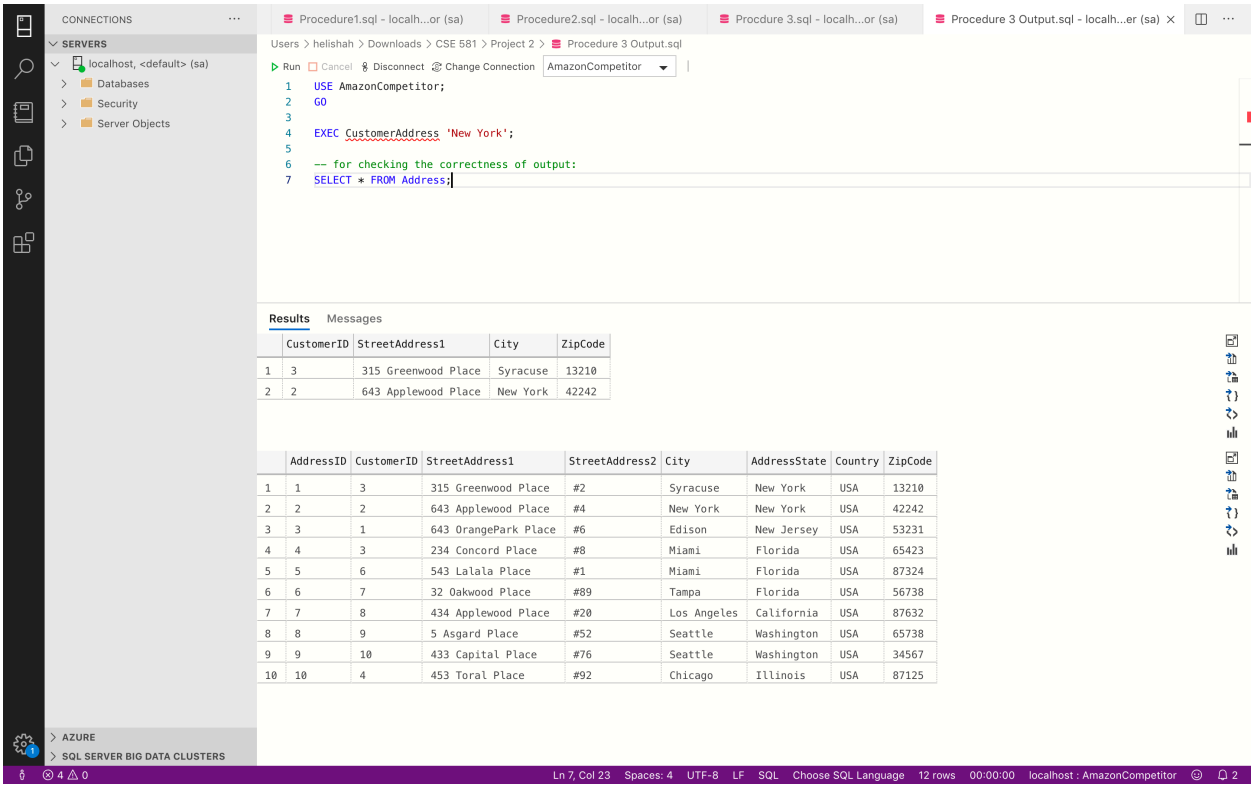
## 4.2.2 This procedure gives details of all the suppliers of a particular product



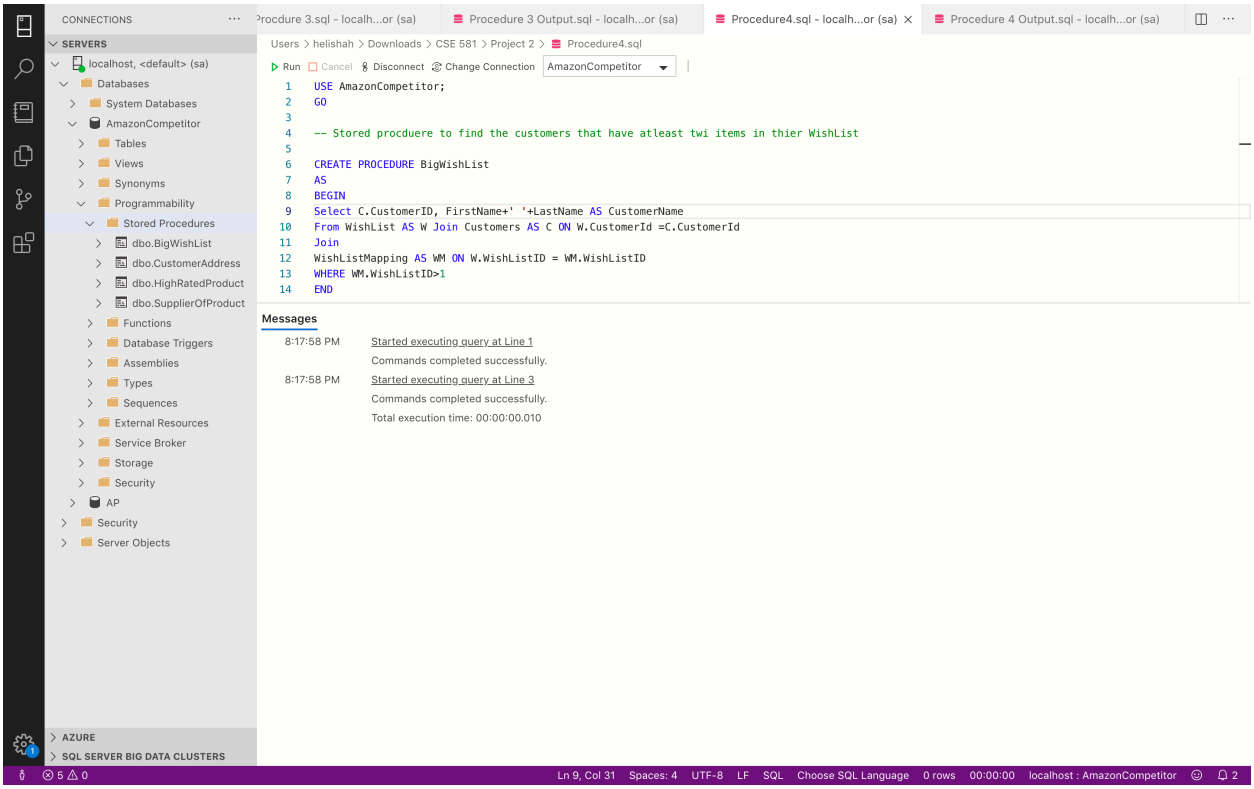



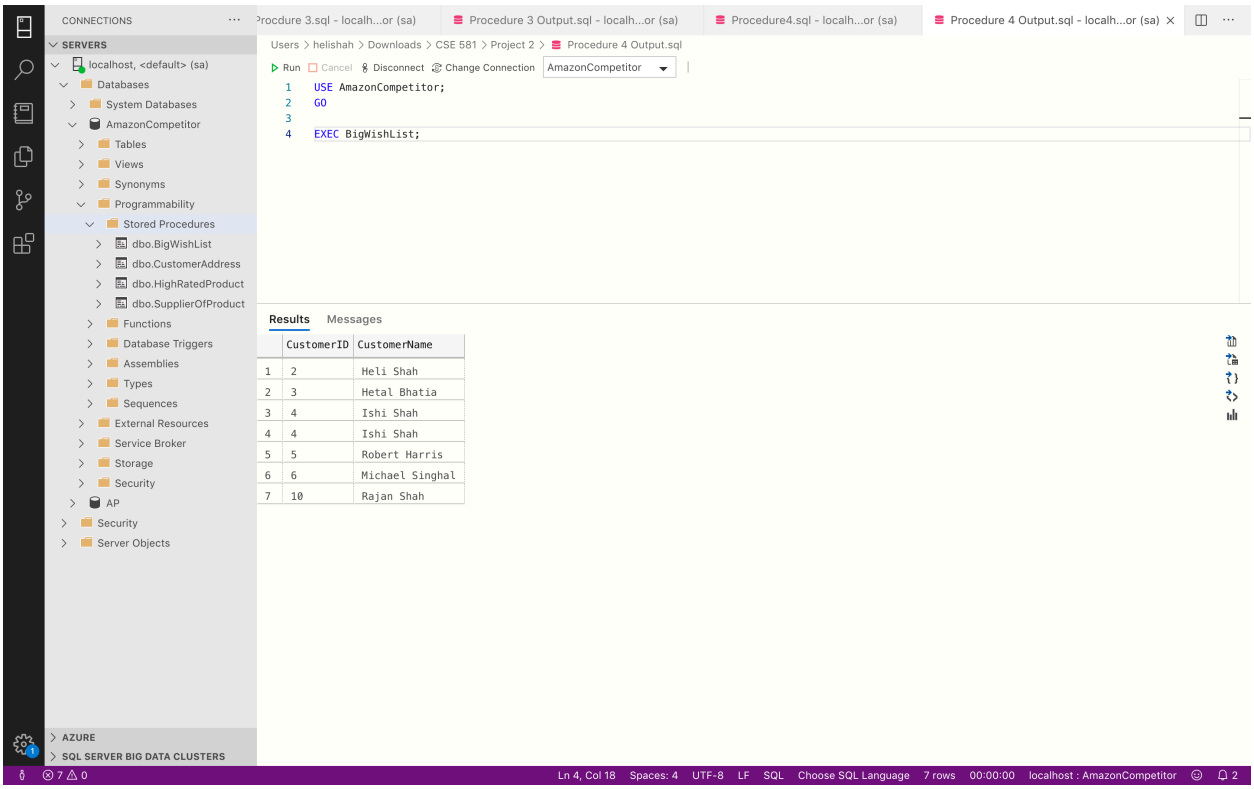
## 4.2.3 This procedure lists the customers living in the state New York



4.2.4 This procedure gives all the customers those have more than 2 products in their WishList

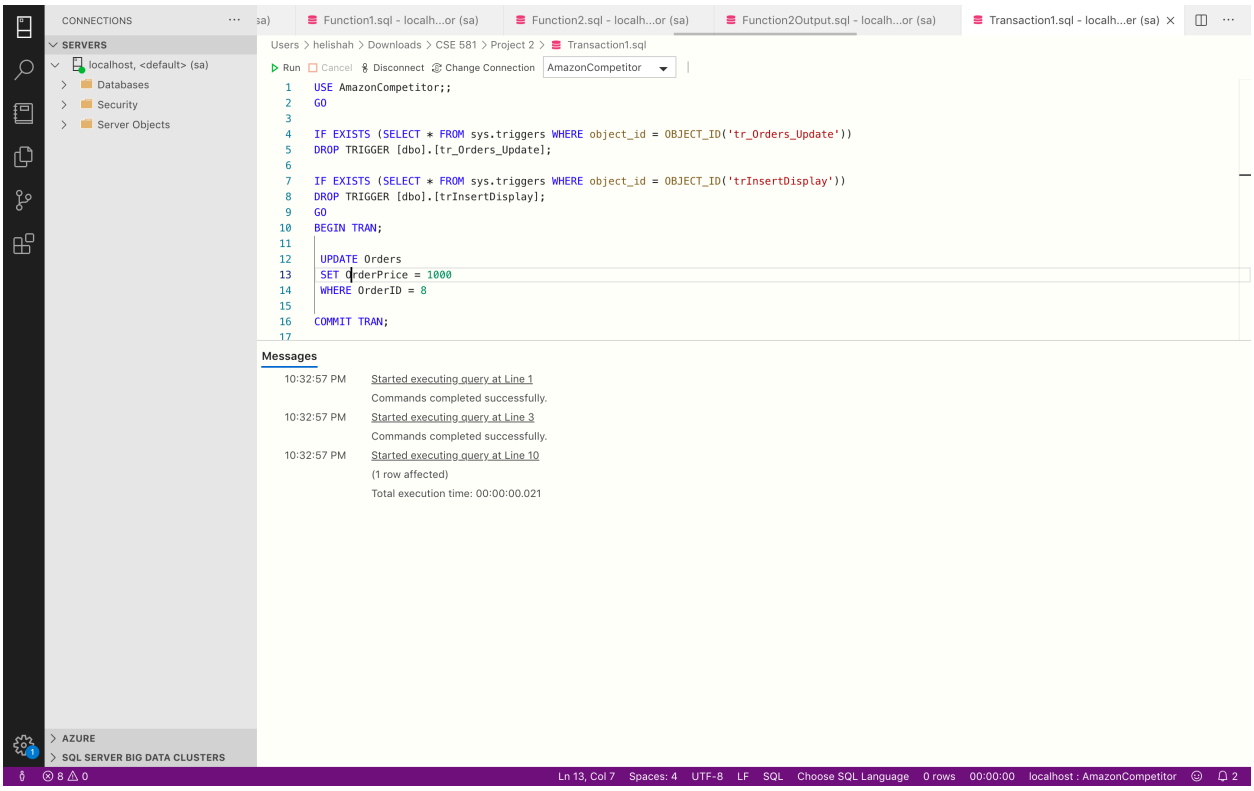


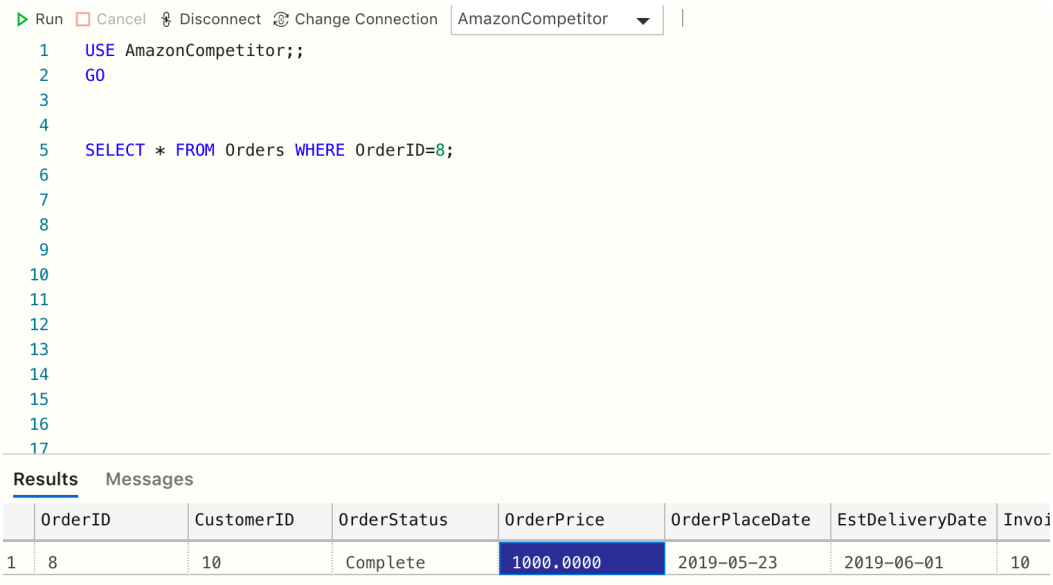
## 4.3 TRANSACTIONS:

### 4.3.1

This transaction updates the OrderPrice for a particular OrderID. It just basically makes sure that it changes it and then allows any other statement to access that record.




The following row is the changed record:

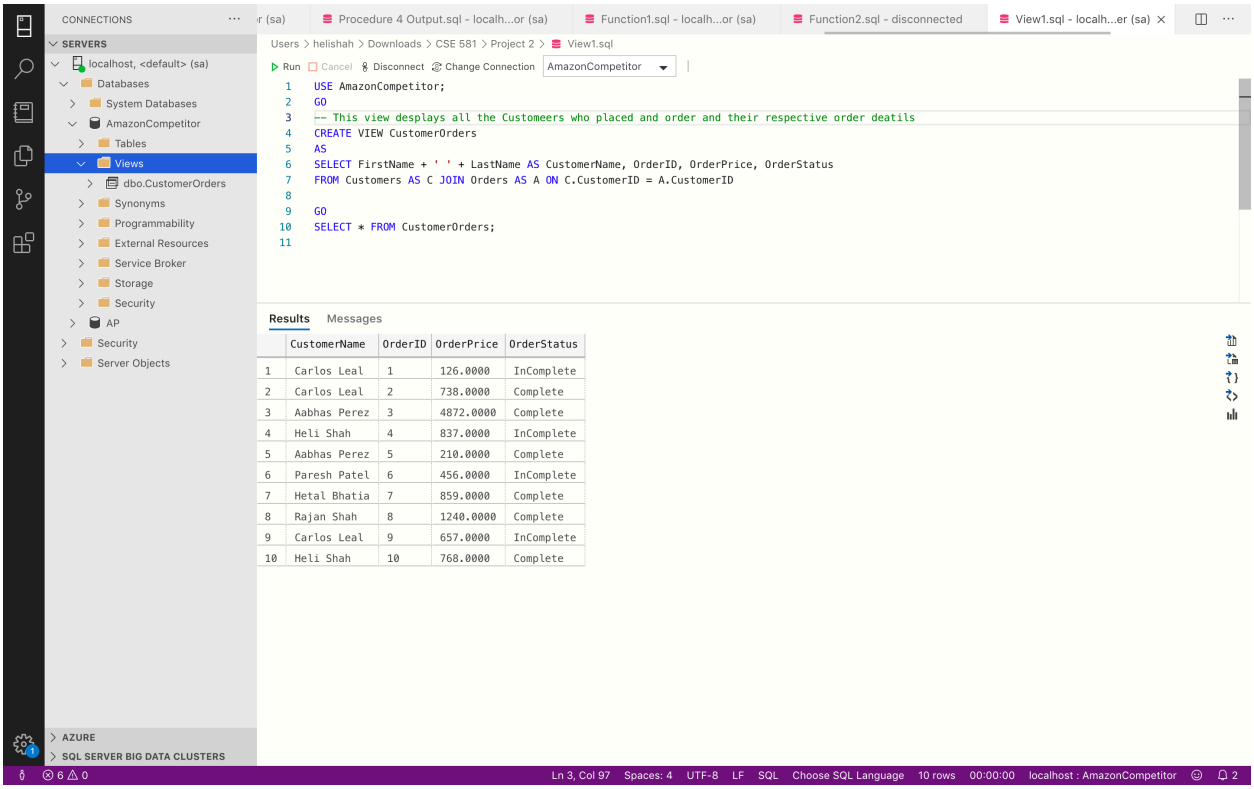


## 4.4 VIEWS AND REPORTS USEFUL FOR THE BUSINESS:

All these views can be also taken in regards as a report that is important for the business analysis for a company like Amazon. Many important patterns about customers-product relation can be deduced from these reports/views.

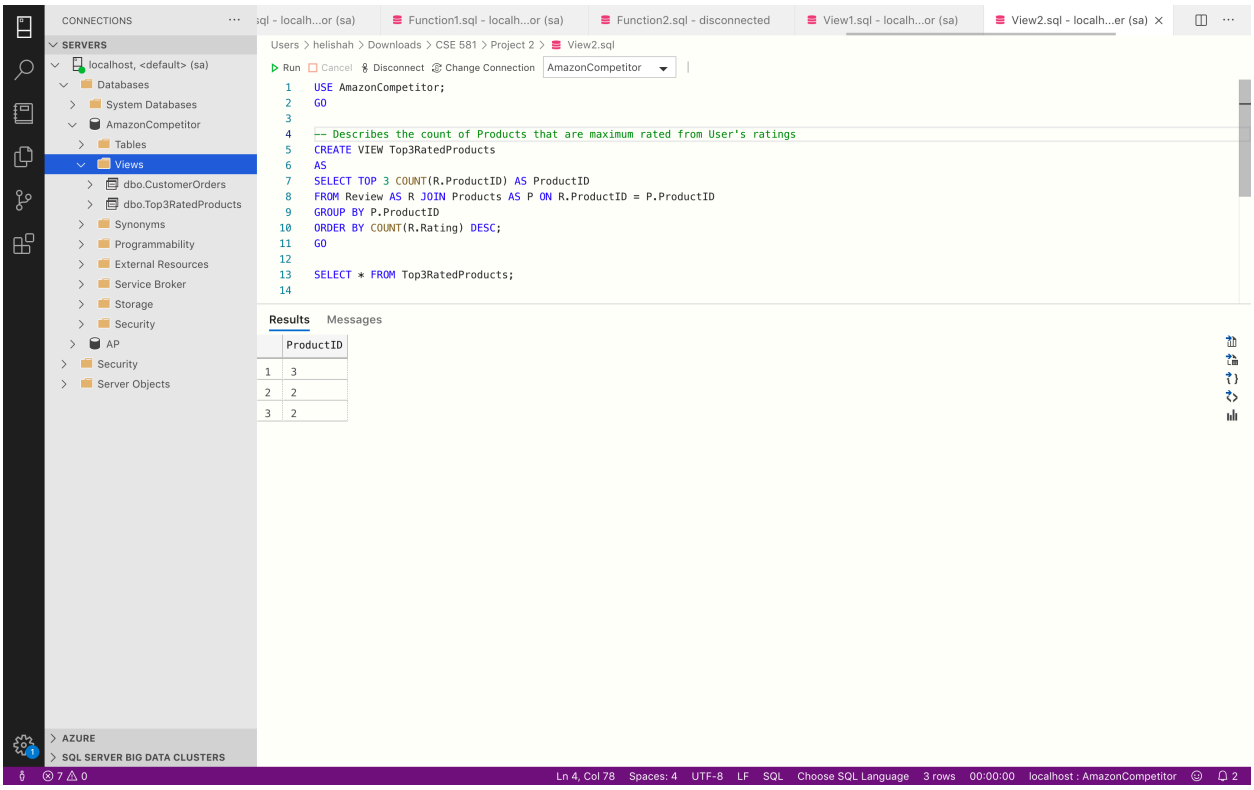
### 4.4.1

This view Customer Orders gives us the information about all those customers who have placed an order along with their order details.



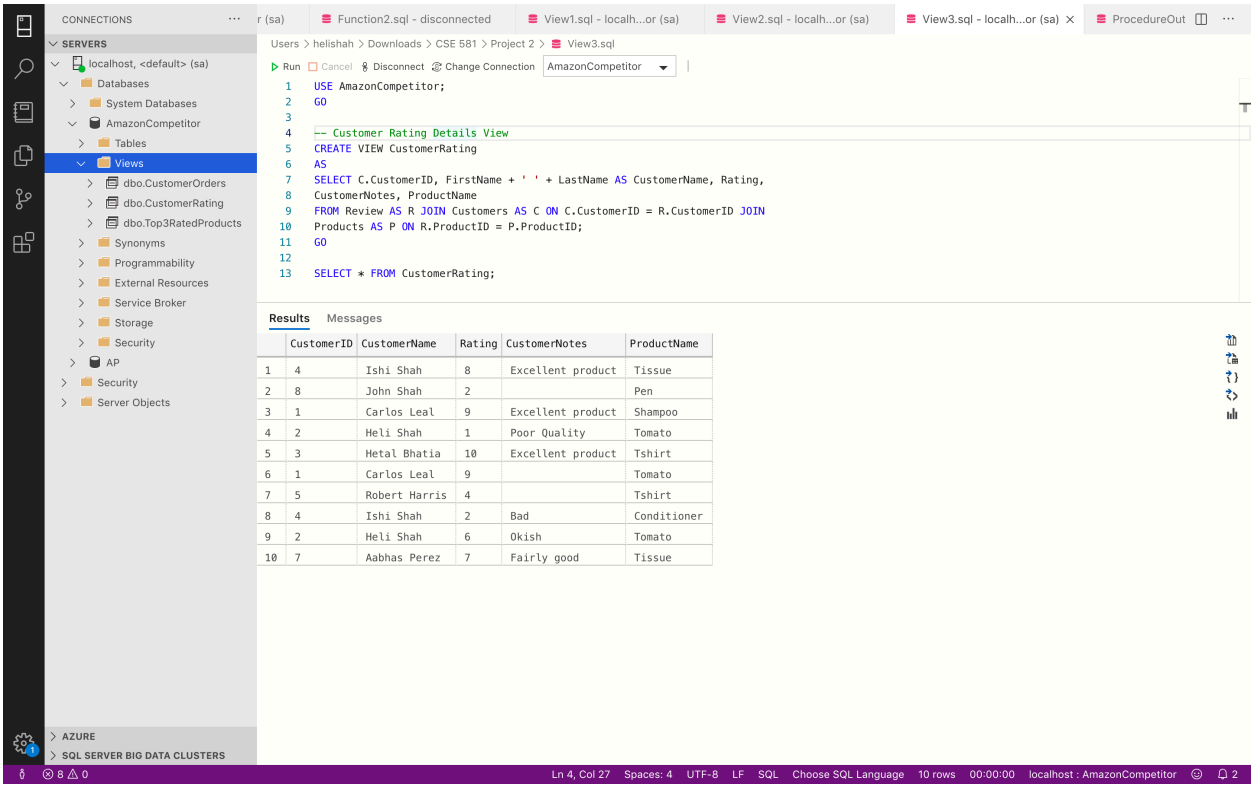
4.4.2

This view Top3PatedProducts extracts the products from Reviews that have the top rating and outputs the Product Number.



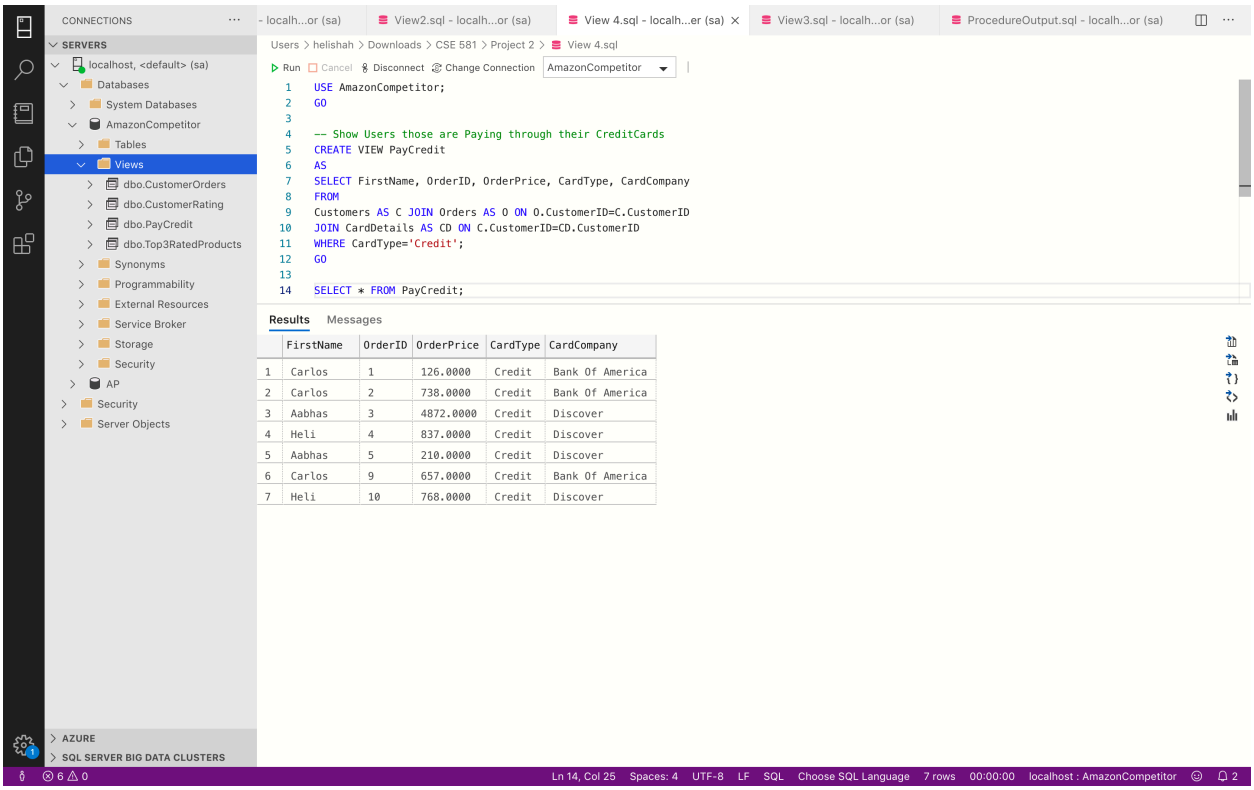
4.4.3

This View describes the Details of a Customer and his/her ratings for a particular product and the notes for the product.



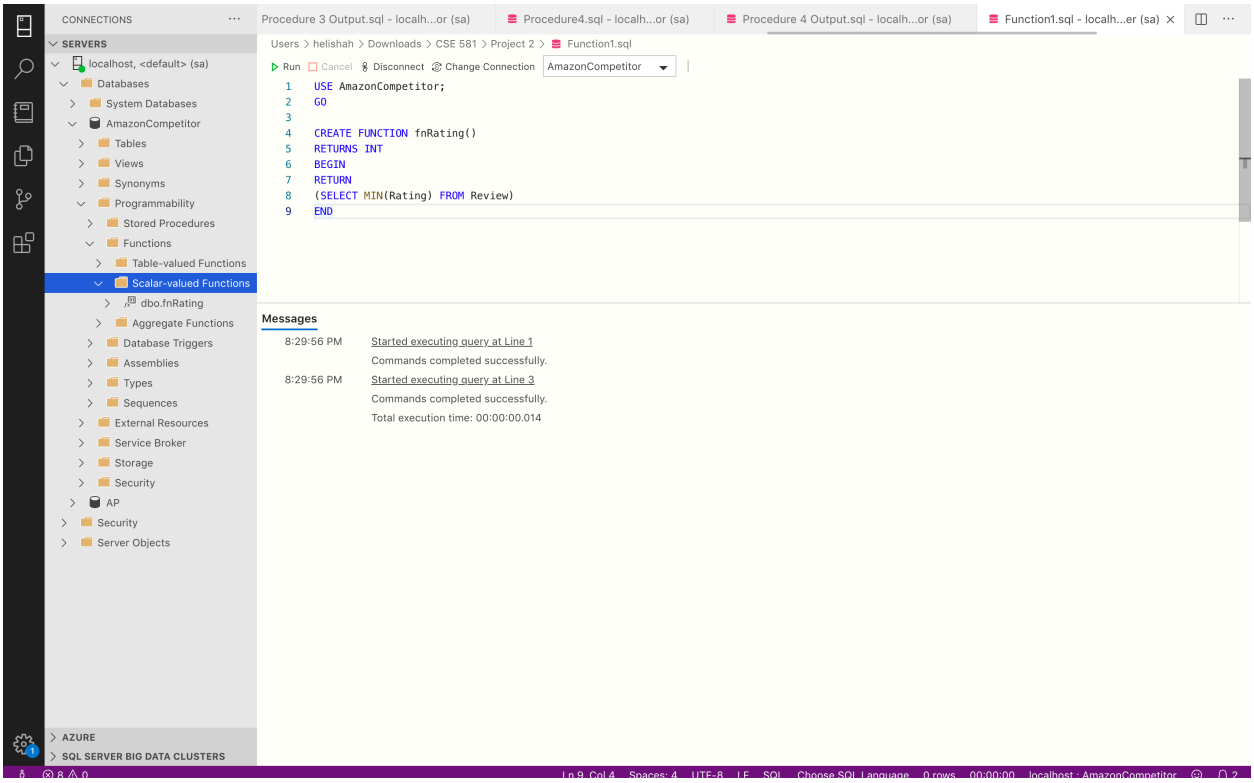
4.4.4

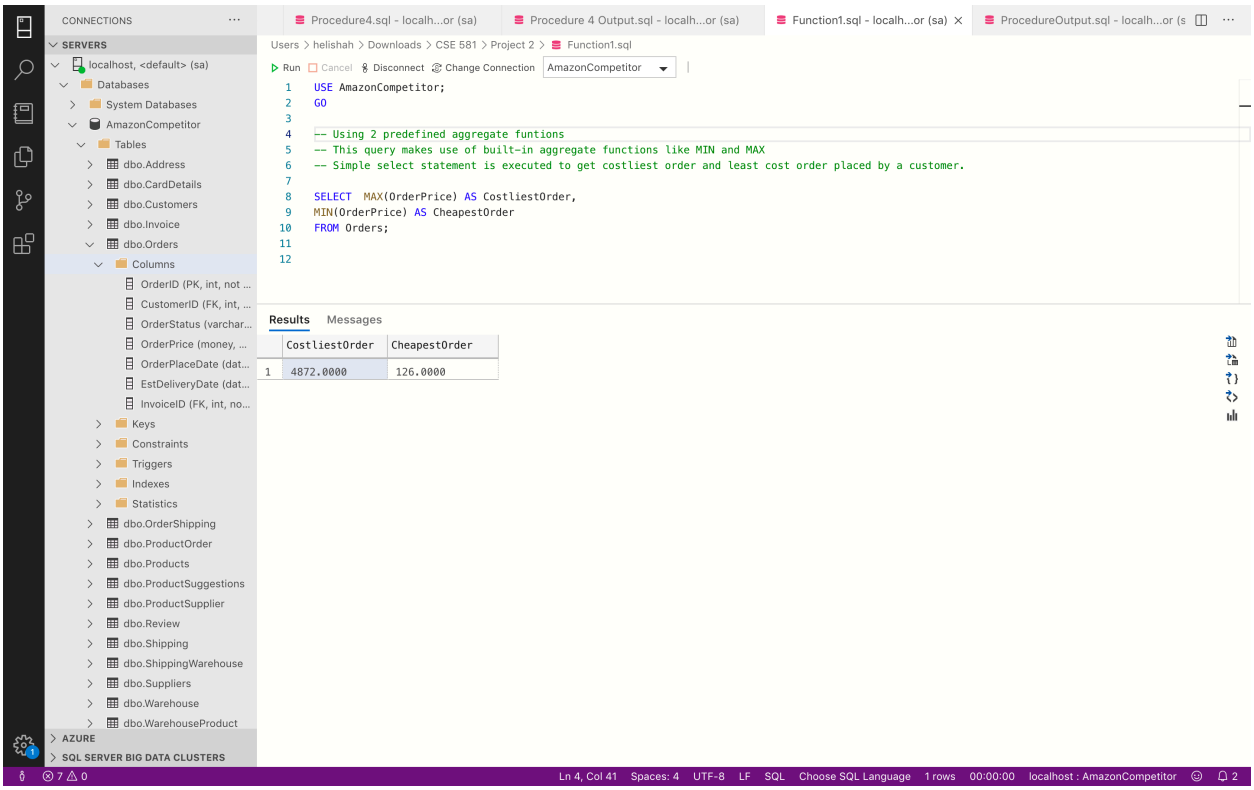
This view gives in the output the customers and their orders who have placed an order and that have their payment method set as Credit Card for that Order.



# 4.5 FUNCTIONS:

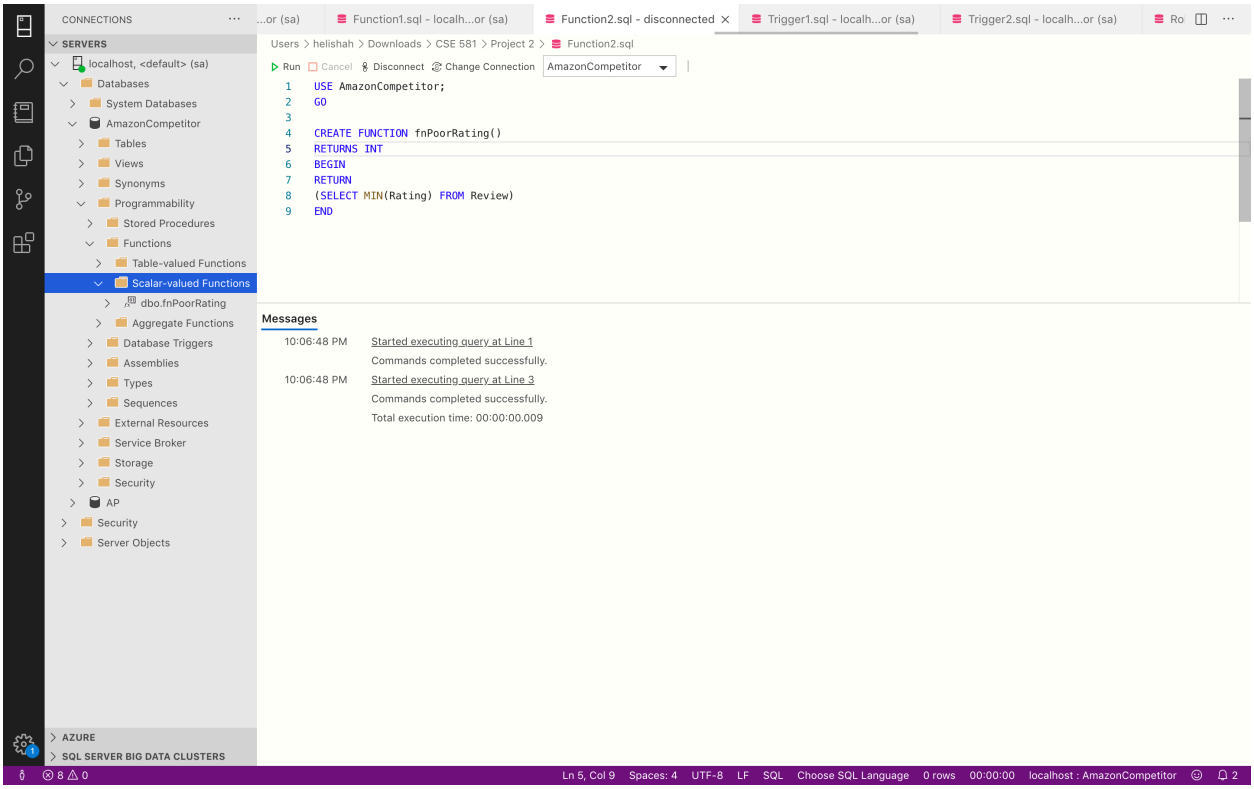
## 4.5.1 Using in-built Aggregate Functions to generate important statistical results. Here, these functions generate the cheapest order and the costliest order placed.



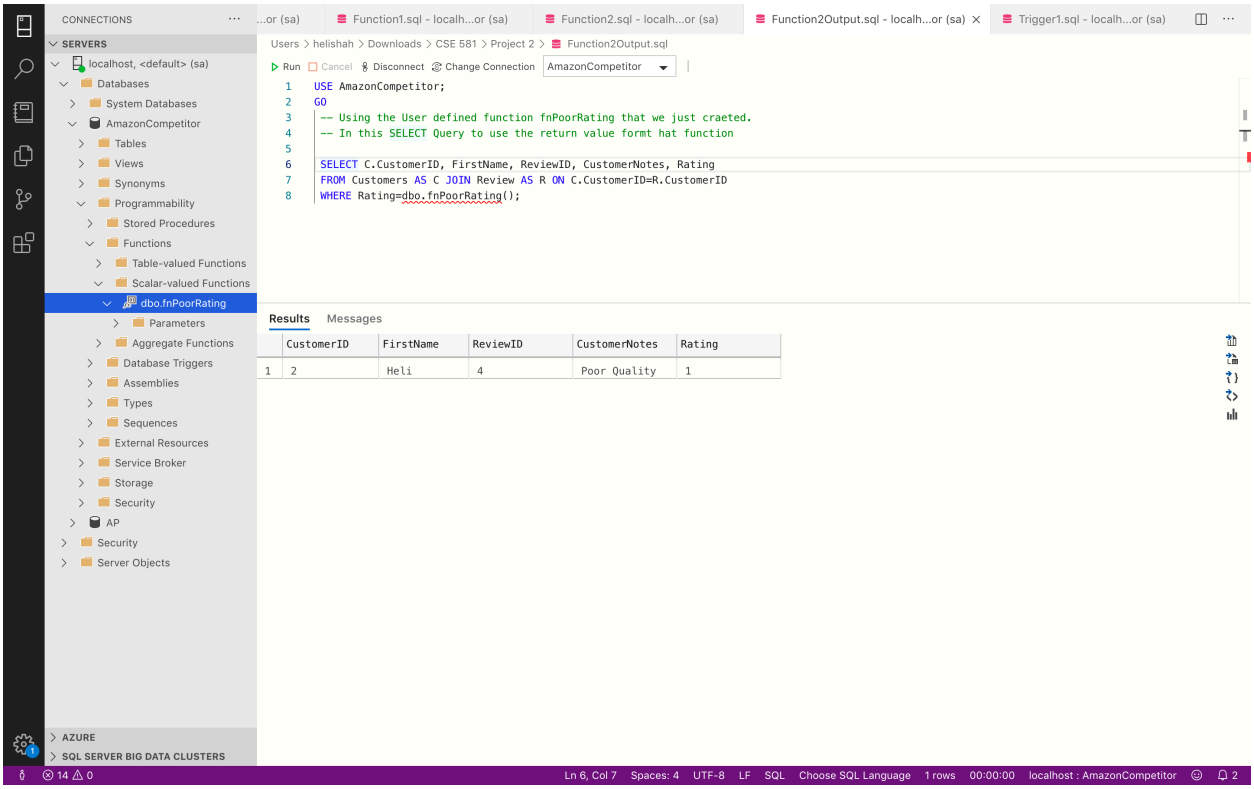



4.5.2

The user defined function fnPoorRating defines a function that returns the value of minimum rating and then this rating can be used for some task. (Maybe to improve the quality of the product, or to satisfy the customer).



Then, in this select statement, the above defined function is called and used to generate the customer information and his/her relevant product review information.



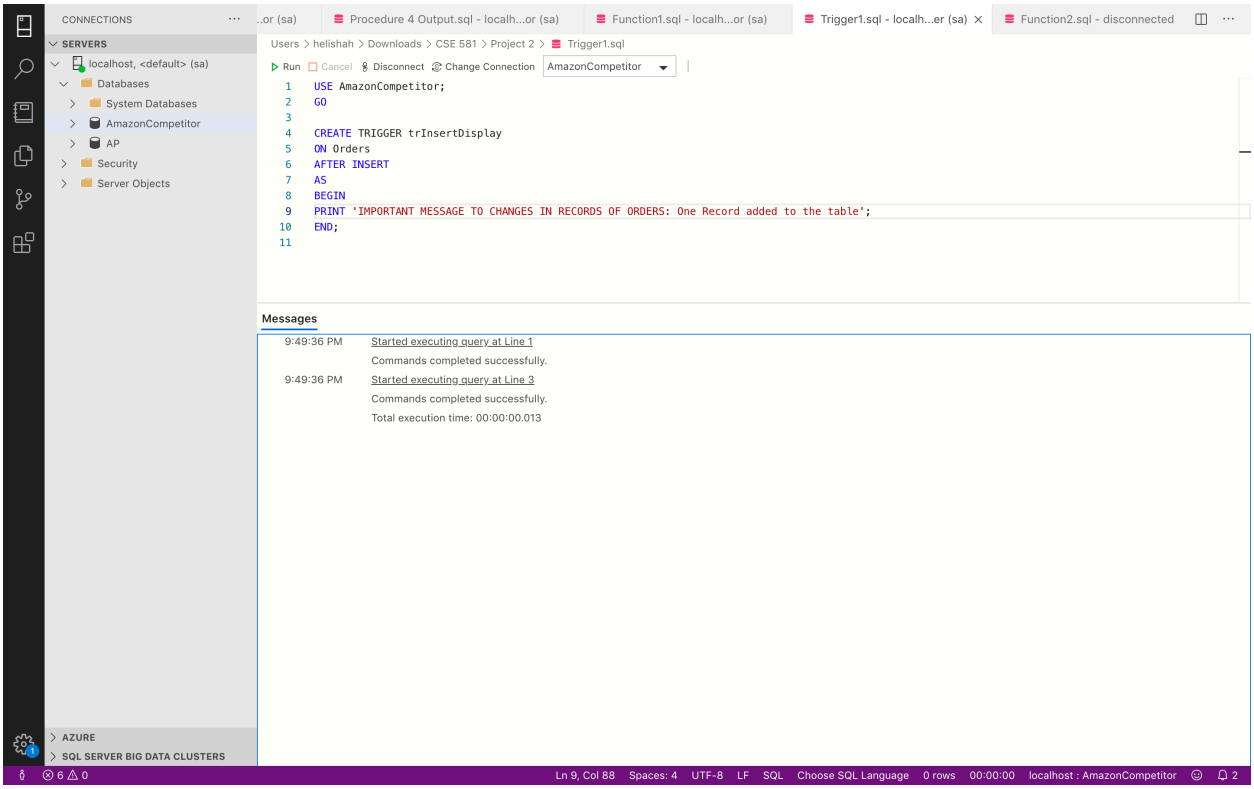
## 4.6 CREATING TRIGGERS:

A **trigger** is a special type of stored procedure that automatically runs when an event occurs in the database server. DML triggers run when a user tries to modify data through a data manipulation language (DML) event. DML events are INSERT, UPDATE, or DELETE statements on a table or view. These triggers fire when any valid event fires, whether table rows are affected or not.

We here will be create 2 triggers one is basic and other is complex trigger, both on same tables to have better understanding of the operation trigger.

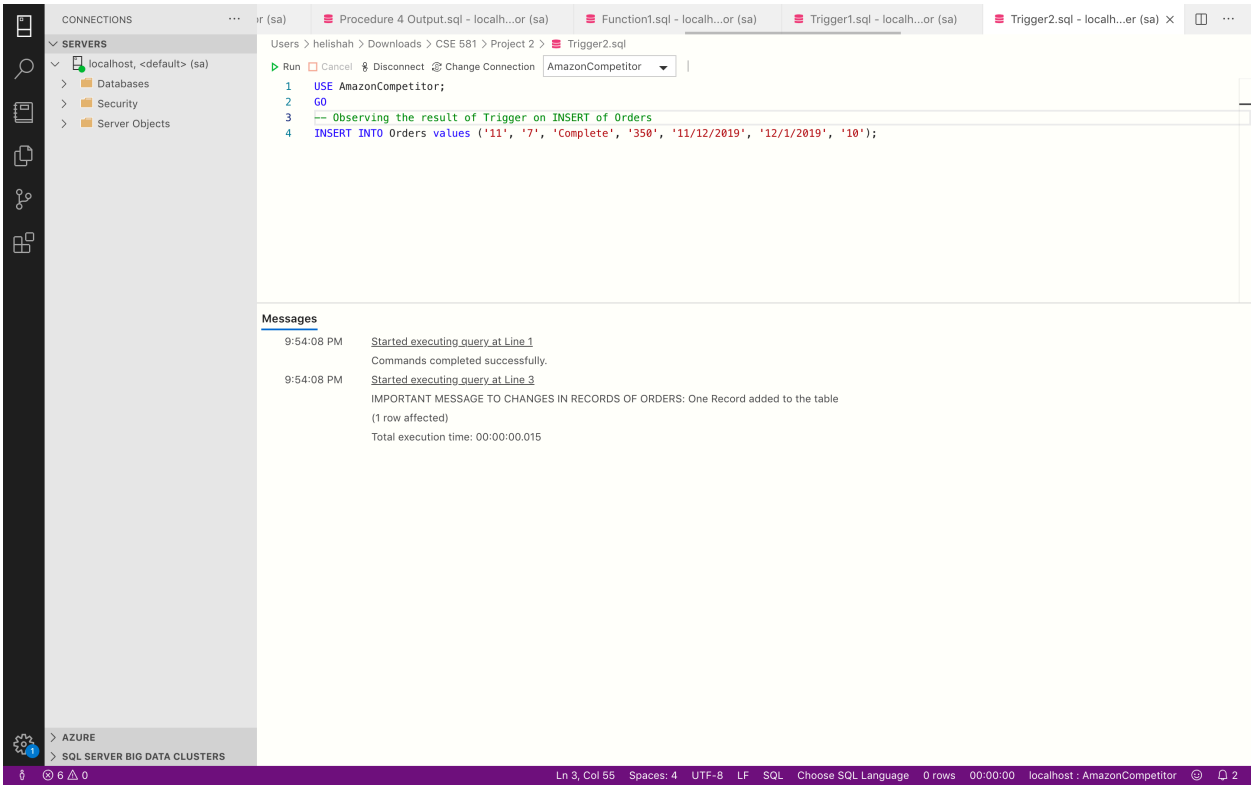
Trigger 1

This trigger is created on Orders table which will get automatically executed when an insert operation gets performed on Orders table. When this trigger fires it will display a statement defined in print clause.



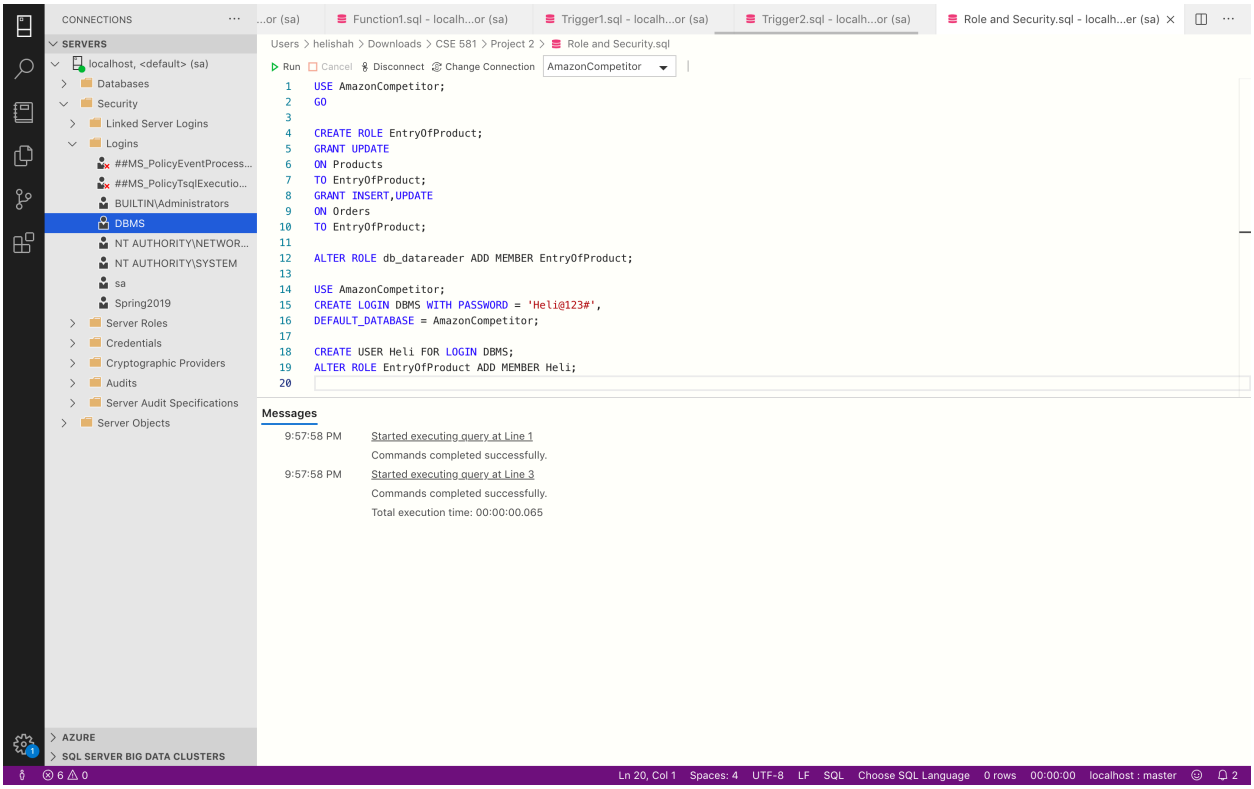

Here our Trigger is completed successfully, so when we fire Insert on Orders table, this will be executed.

Now, see the output while inserting a record to the Orders table.
The trigger will be fired first and then the statement of Insert will execute.



## 4.7 ROLES, LOGIN AND PASSWORD ENCRYPTION: (SECURITY LEVELS FOR THE DATABASE)

This statements create a role named EntryOf Product which has been granted Update permission on Products.
It has Insert, Update permission on OrderItems tables.
This role also got permission to read data from all tables in current database by adding as a member db-datareader.




Next, we have assigned a password to LOGIN named DBMS using create login script and assigning default database as our current database, i.e. AmazonCompetitor. We also created member named Heli for DBMS login and added him as member to EntryOfProduct by which all grant permissions of EntryOfProduct got transferred to member Heli.

# 5. CONCLUSION:

## 5.1 Project Analysis:

The project Amazon Competitor Database has been created and various tables have been created. Sample data was input to the database and it was tested using stored procedures, function, views, triggers and various constraints have been used. Overall the working of the project gives the required outcome. The database was normalized and there is no redundancy in it this makes the database simple and efficient to use. The project has been designed in such a way that it could meet future requirements.

## 5.2 Remarks:

The Amazon competitor database introduces all the concepts of creating, implementing and testing a database. It also made me learn about thee actual industrial requirements in the companies these days. For implementing, the labs and the concepts introduced in class helped in developing the database. This project was a very good practice for all the topics covered in the lecture. The in-class presentations were also a lot helpful to solve this problems in the project.

I used https://my.vertabelo.com/drive for designing the database.
I used https://app.creately.com/manage/project/home to create my ER diagram.