CSE - 581 Intro to DBMS

Project -2

COVER PAGE

Title:

RideHailing Database

Abstract:

**Ride hailing** services are services that use online-enabled platforms to connect between passengers and drivers using their personal vehicles. This document is to present the how designing of database by applying third normal form principles and maintaining relationship between tables. Implementation of the design using MSSQL database with SQL server. Also addressing issues and demonstration issues regarding security for customers and drivers. Implemented Stored procedures, triggers, Functions and views that are useful to this database users.

Name: Divya Sai Sekhar, Mullapudi

SUID : #7556584447

Table of Contents

Abstract ............................................................................................................................................ 1

Design ................................................................................................................................................ 3

Introduction ....................................................................................................................................... 3

Design Considerations with 3rd normal form ..................................................................................... 3

Relationships Between Tables ............................................................................................................4

One to One Relationship ....................................................................................................................4

One– Many Relationships .................................................................................................................. 4

Entity Relation Diagram .................................................................................................................... 5

Implementation ................................................................................................................................. 6

Database Creation................................................................................................................................6

Table Creation ..................................................................................................................................... 6

Triggers...................................................................................................................................................9

Testing Healthcare Database .................................................................................................................. 25

Function and Stored Procedure…………………. .......................................................................................... 25

Demonstrating Reliability of the Database ............................................................................................. 33

Security Issues ....................................................................................................................................33

Conclusion and Remarks………………………………………………………………………………………………………………….35

**Design:**

**Introduction:**

The objective of the database design is to generate logical and physical layer of design models . Designing of database is helpful in creating a structure model of data for easy flow and segregation of data from one real world entity to other entities. TO achieve these goals data in real worlds entities are considered as a table and all the attributes that define the model are columned to that table. To maintain easy flow of data form one table to other table are normalized to third normal form. Here in this Ride hailing physical design model refer to considering data can be sub divided onto physical entities like Car, Customers, Drivers, Drivers Status, Insurance, Car Insurance, Licence, Trips, Banking, Customer Credit Card. Logical design model explains various relationships these entities.

**Tables:**

Based on business requirement below are the design consideration an divide into following tables.

Address: Address are the entities which are determined by GPS location. Many entities in the database can requires them. Attributes are door number, street, City, State, Pin code.

Banking: Banking is an entity that is require by the driver to receive income earned form the trips. It contains bank name, routing number, account number, account type and percent share to be given into that account.

Car: Car is another entity which is possessed by the driver. It have make, model company, release ear, color, Max Passenger count, Max baggage count as attributes.

Car Insurance: Car insurance is the required for the each car and can be obtained by any company, policy number, date of expiry.

Customer Credit Card: Credit card are to maintained each customer can have any number of card and may use as per requirement during trips.

Customer Rating: Rating that customer receiver at end of each of the trips from the corresponding driver in the trip.

Customers: Customer are unique entity each person who are using the riderailing for requesting the trips.

Drivers: Drivers are entity who respond the request and perform services for the customers through this ride hailing application.

Driver Rating: Rating that driver receiver at end of each of the trips from the corresponding customer in the trip.

DriverStatus: To know whether driving is waiting for an request of in a trip or inactive.

Insurance: Each individual driver can have any number of insurance based on categories.

Licence: each driver requires an licence entity to check whether he is a legal driver to give a trip request.

Trips: To maintain details all the trips that had happened using the ride hailing.

**Relationships:**

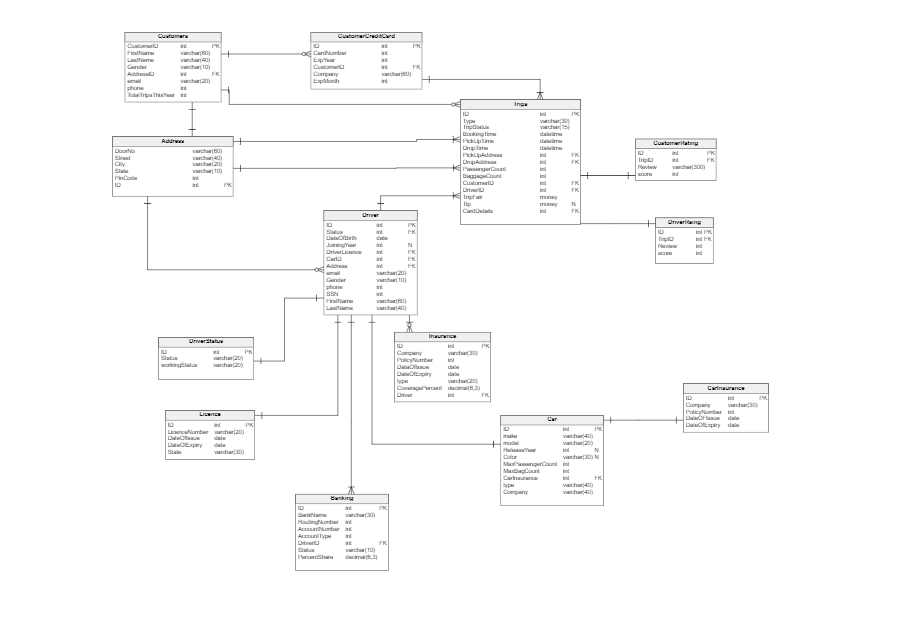
**One to one relationship:**

Address to trips, Drivers to Drivers Status, Driver to License, Driver to Car, Car to Car Insurance, Trips to Customer Rating, Trips to Driver rating,

One to Many Relationship:

Customer to CreditCards, Customer to trips, Address to Trips,Driver to Trips, Drivers to Insurance, Driver to Banking. Trips to Cards.

**E/R diagram:**



**Implementation**

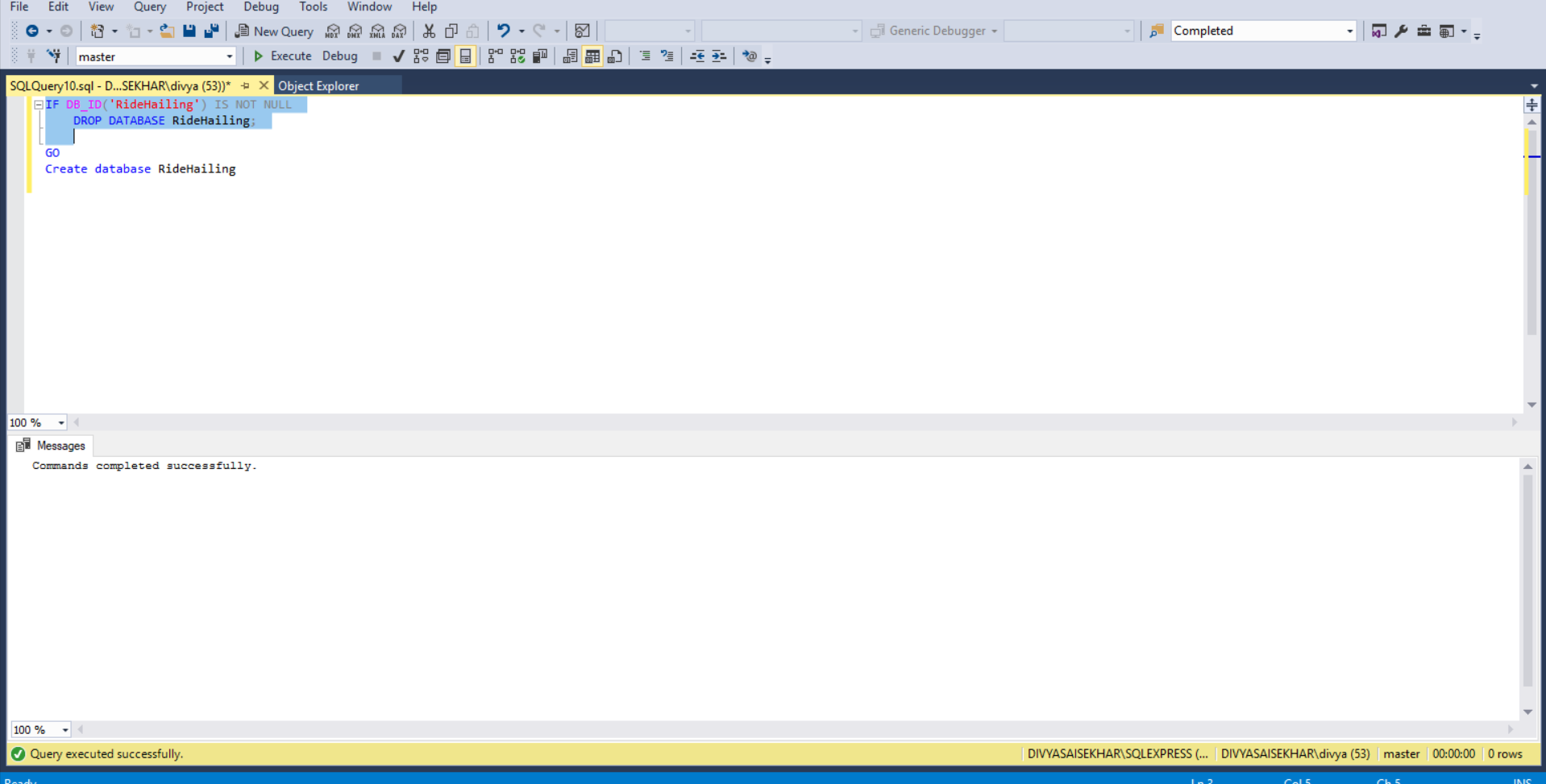
**Database Creations:**

IF DB\_ID('RideHailing') IS NOT NULL

DROP DATABASE RideHailing;

GO

Create database RideHailing



**Tables Creations:**

GO

USE RideHailing;

Go

CREATE TABLE Addresss (

ID int not null primary key,

DoorNumber varchar(60) not null,

Street varchar(40) not null,

City varchar(20) not null,

State varchar(20) not null,

PinCode int not null

);

Go

CREATE TABLE CarInsurance (

ID int not null primary key,

Company varchar(40) not null,

PolicyNumber int not null,

DateOFIssue date not null,

DateOFExpiry date not null,

);

Go

CREATE TABLE Car (

ID int not null primary key ,

Company varchar(40) not null,

make varchar(40) not null,

model varchar(20) not null,

ReleaseYear int,

Color varchar(30),

MaxPassengerCount int not null default 4,

MaxBagCount int not null check(MaxBagCount > 0),

type varchar(40) not null,

CarInsurance int not null foreign key references CarInsurance(ID),

);

Go

CREATE TABLE Licence (

ID int not null primary key ,

LicenceNumber varchar(20),

DateOfIssue date,

DateOfExpiry date,

State varchar(30),

);

Go

CREATE TABLE DriverStatus (

ID int not null primary key ,

status varchar(20) not null,

WorkingStatus varchar(20) not null,

);

Go

CREATE TABLE Driver (

ID int not null primary key ,

status int foreign key references DriverStatus(ID),

DateOfBirth date not null,

JoiningYear int,

DriverLicence int foreign key references Licence(ID),

CarID int foreign key references Car(ID),

Address int not null foreign key references Addresss(ID),

email varchar(30) not null,

Gender varchar(10) not null,

Phone int not null,

SSN int not null,

FirstName varchar(60) not null,

LastName varchar(40) not null,

);

Go

CREATE TABLE Banking (

ID int not null primary key ,

BankName varchar(30) not null,

RoutingNumber int not null,

AccountNumber int not null,

AccountType varchar(20) not null,

DriverID int not null foreign key references Driver(ID),

Status varchar(30) not null,

PercentShare decimal(6,3) null default 000.000

);

Go

CREATE TABLE Insurance (

ID int not null primary key ,

Company varchar(40) not null,

PolicyNumber int not null,

DateOFIssue date not null,

DateOFExpiry date not null,

type varchar(20) not null,

CoveragePercent decimal(6,3) not null default 100.000,

Driver int not null foreign key references Driver(ID)

);

Go

CREATE TABLE Customers (

CustomerID int not null primary key,

FirstName varchar(60) not null,

LastName varchar(40) not null,

Gender varchar(10) not null,

AddressID int not null foreign key references Addresss(ID),

email varchar(20) not null,

Phone int null,

TotalTripsThisYear int default 0);

Go

CREATE TABLE CustomerCreditCard (

ID int not null primary key,

CardNumber varchar(16) not null,

ExpYear int not null,

CustomerID int not null foreign key references Customers(CustomerID),

Company varchar(40),

ExpMonth int not null

);

Go

CREATE TABLE Trips (

ID int not null primary key,

TripStatus varchar(30) not null,

BookingTime datetime not null,

PickUpTime datetime not null,

DropTime datetime not null,

PickUpAddress int not null foreign key references Addresss(ID),

DropAddress int not null foreign key references Addresss(ID),

PassengerCount int not null,

BaggageCount int not null,

CustomerID int not null foreign key references Customers(CustomerID),

DriverID int not null foreign key references Driver(ID),

TripFair money not null CHECK(TripFair > 0),

Tip money default 0 ,

CardDetails int not null foreign key references CustomerCreditCard(ID)

);

Go

CREATE TABLE CustomerRating (

ID int not null primary key ,

TripID int not null foreign key references Trips(ID),

Review varchar(300),

score int check (score >0 and score <11 )

);

Go

CREATE TABLE DriverRating (

ID int not null primary key ,

TripID int not null foreign key references Trips(ID),

Review varchar(300),

score int check (score >0 and score <11 )

);



**Triggers::**

**Trigger1 for UpdateTripsThisYear**

Update total number of trips the customer travelled using this ride hailing application in the current year.

GO

USE RideHailing

GO

CREATE TRIGGER UpdateTripsthisYear

on dbo.Trips

For INSERT, UPDATE

AS

BEGIN

DECLARE @CustomerID int

DECLARE @Completed varchar(30)

DECLARE @LastTripDate date

DECLARE @CurrentDate date

DECLARE @LastTripID int

DECLARE @TripsCount int

Declare @currentTripID int

SELECT @CustomerID = CustomerId, @CurrentDate = DropTime, @Completed= TripStatus, @currentTripID= ID from inserted

Select @TripsCount=TotalTripsThisYear from Customers where CustomerId = @CustomerID

Select @LastTripDate=MAX(DropTime) from Trips where CustomerId = @CustomerID

IF (DATEDIFF(year, @LastTripDate, @CurrentDate) >0)

BEGIN

IF(@Completed = 'Completed')

BEGIN

UPDATE Customers SET TotalTripsThisYear = 1 where CustomerId = @CustomerID;

END

END

ELSE

BEGIN

IF(@Completed = 'Completed')

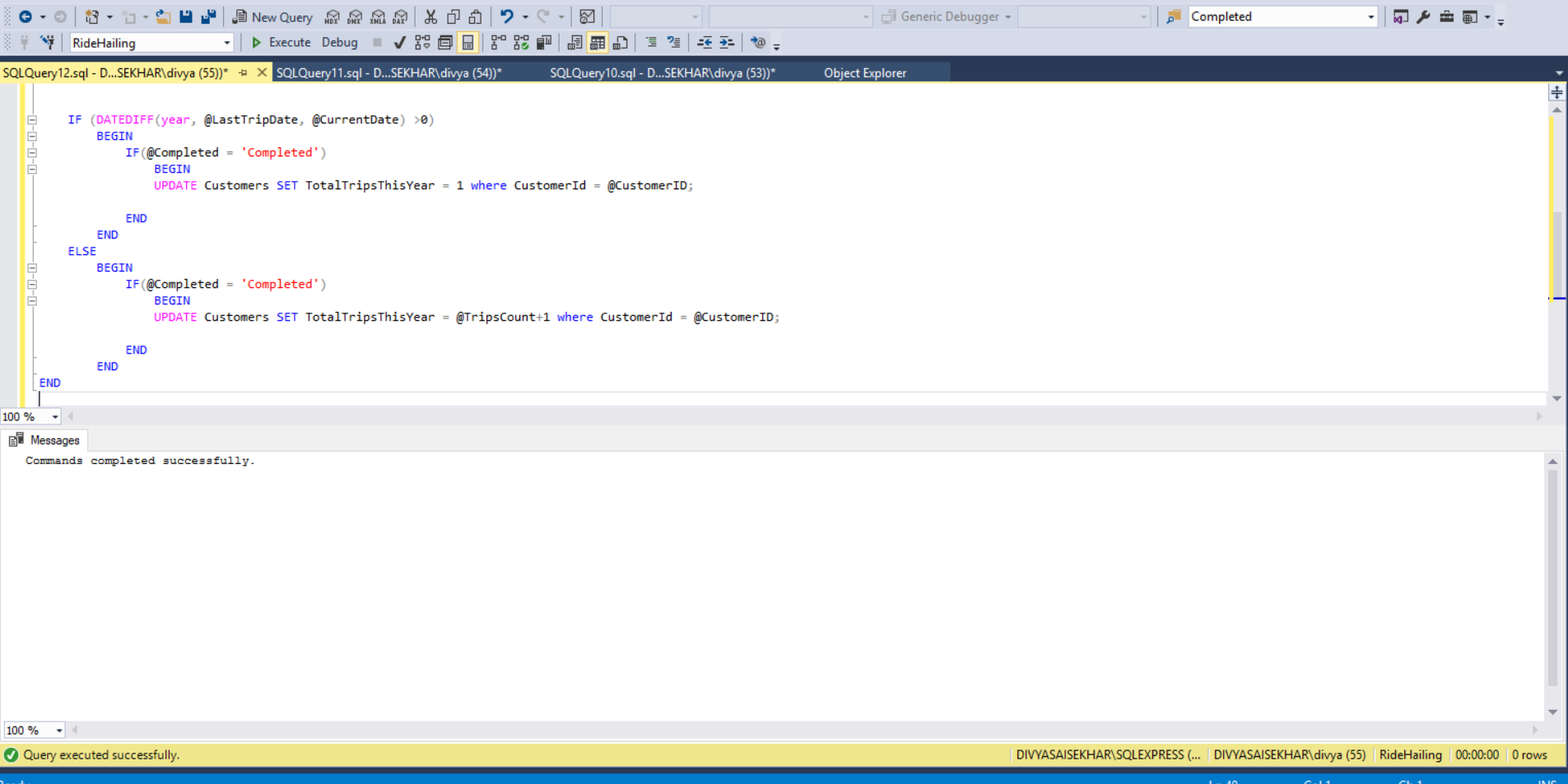
BEGIN

UPDATE Customers SET TotalTripsThisYear = @TripsCount+1 where CustomerId = @CustomerID;

END

END

END



**Triggers 2::**

This trigger is use to automatically update status of the driver from on trip to available and make hm available for next trip.

GO

USE RideHailing;

GO

Create trigger ChangeDriverStatus

on dbo.Trips

After INSERT, UPDATE

AS

BEGIN

Declare @ID int

Declare @customerId int

Declare @Completed varchar(30)

select @ID=DriverID, @Completed=TripStatus from inserted

BEGIN

IF(@Completed = 'Completed')

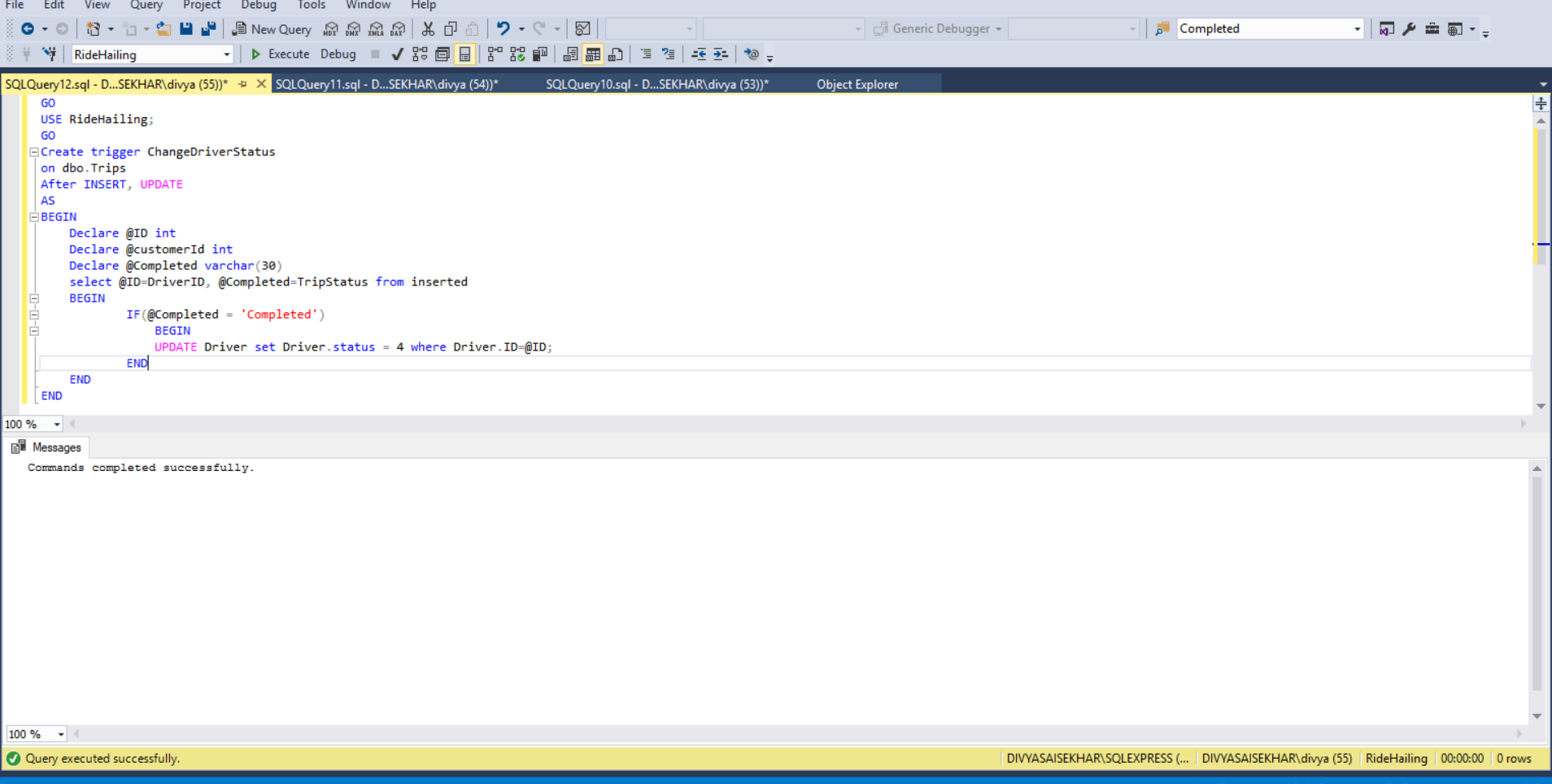
BEGIN

UPDATE Driver set Driver.status = 4 where Driver.ID=@ID;

END

END

END



**Testing::**

GO

USE RideHailing;

GO

insert into DriverStatus values (1,'Working','On Trip');

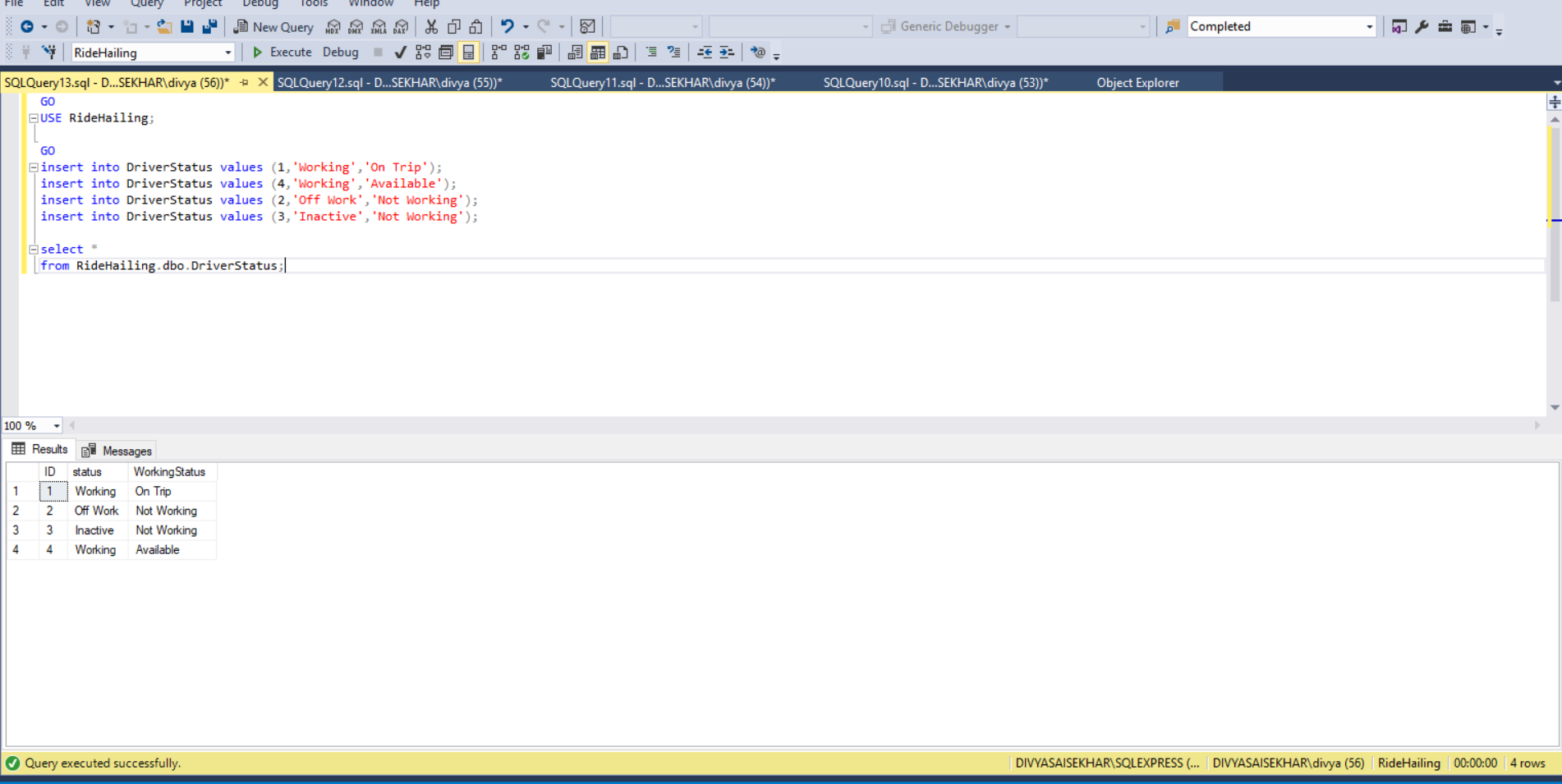
insert into DriverStatus values (4,'Working','Available');

insert into DriverStatus values (2,'Off Work','Not Working');

insert into DriverStatus values (3,'Inactive','Not Working');

select \*

from RideHailing.dbo.DriverStatus;



GO

USE RideHailing;

Go

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (1,'708','S Beech Street', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (2,'250','Buckingham Ave', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (3,'429','Columbus Ave', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (4,'529','Columbus Ave', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (5,'535 ','WestCott Ave', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (6,'400','GreenWood place', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (7,'800','S Bufbs Street', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (8,'250','Rutger Ave', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (9,'429','Crouse Ave', 'Syracuse','New Nork', 13210);

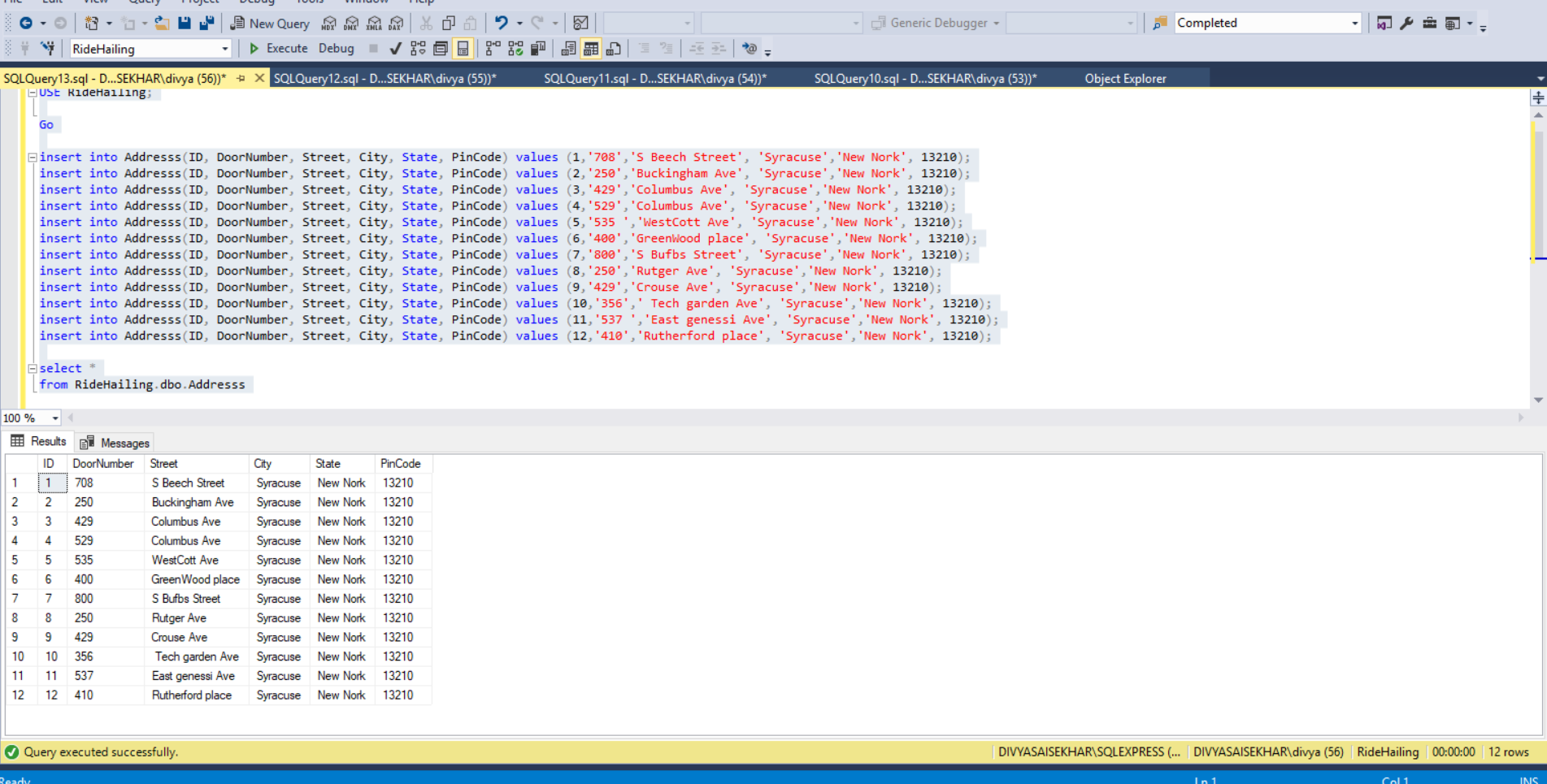
insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (10,'356',' Tech garden Ave', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (11,'537 ','East genessi Ave', 'Syracuse','New Nork', 13210);

insert into Addresss(ID, DoorNumber, Street, City, State, PinCode) values (12,'410','Rutherford place', 'Syracuse','New Nork', 13210);

select \*

from RideHailing.dbo.Addresss



GO

USE RideHailing;

Go

insert into Customers values(1,'Sekhar','Mullapudi','Male',2,'saisekhar@gmail.com',1156789099,0);

insert into Customers values(2,'Nitesh','Bhutani','Male',5,'nbhuthan@gmail.com',1157298587,0);

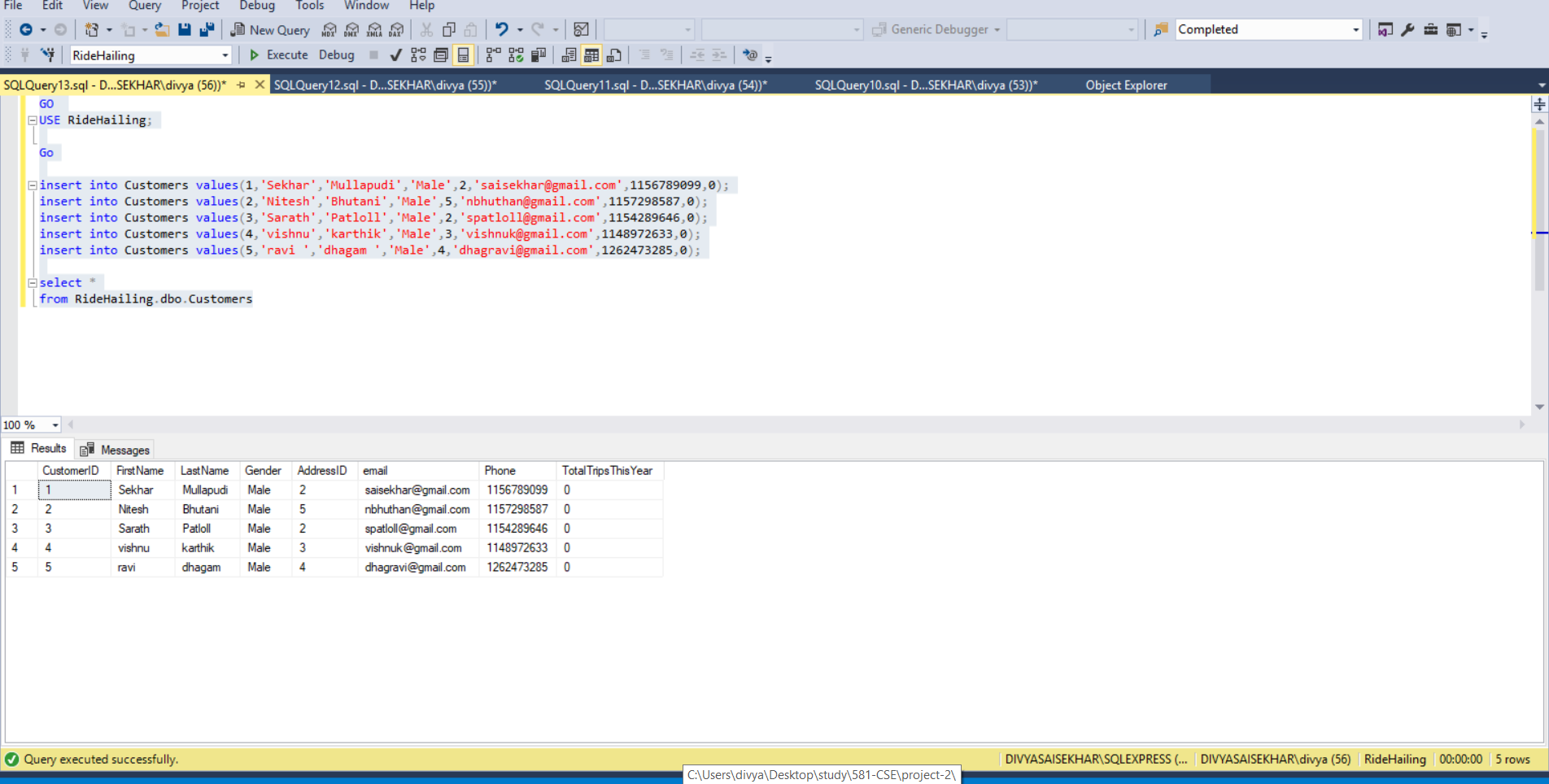
insert into Customers values(3,'Sarath','Patloll','Male',2,'spatloll@gmail.com',1154289646,0);

insert into Customers values(4,'vishnu','karthik','Male',3,'vishnuk@gmail.com',1148972633,0);

insert into Customers values(5,'ravi ','dhagam ','Male',4,'dhagravi@gmail.com',1262473285,0);

select \*

from RideHailing.dbo.Customers



GO

USE RideHailing;

Go

insert into CustomerCreditCard values(1,'1122345678903579',2023,1,'AmericanExpress',2);

insert into CustomerCreditCard values(2,'1122903579345678',2020,1,'Discover',2);

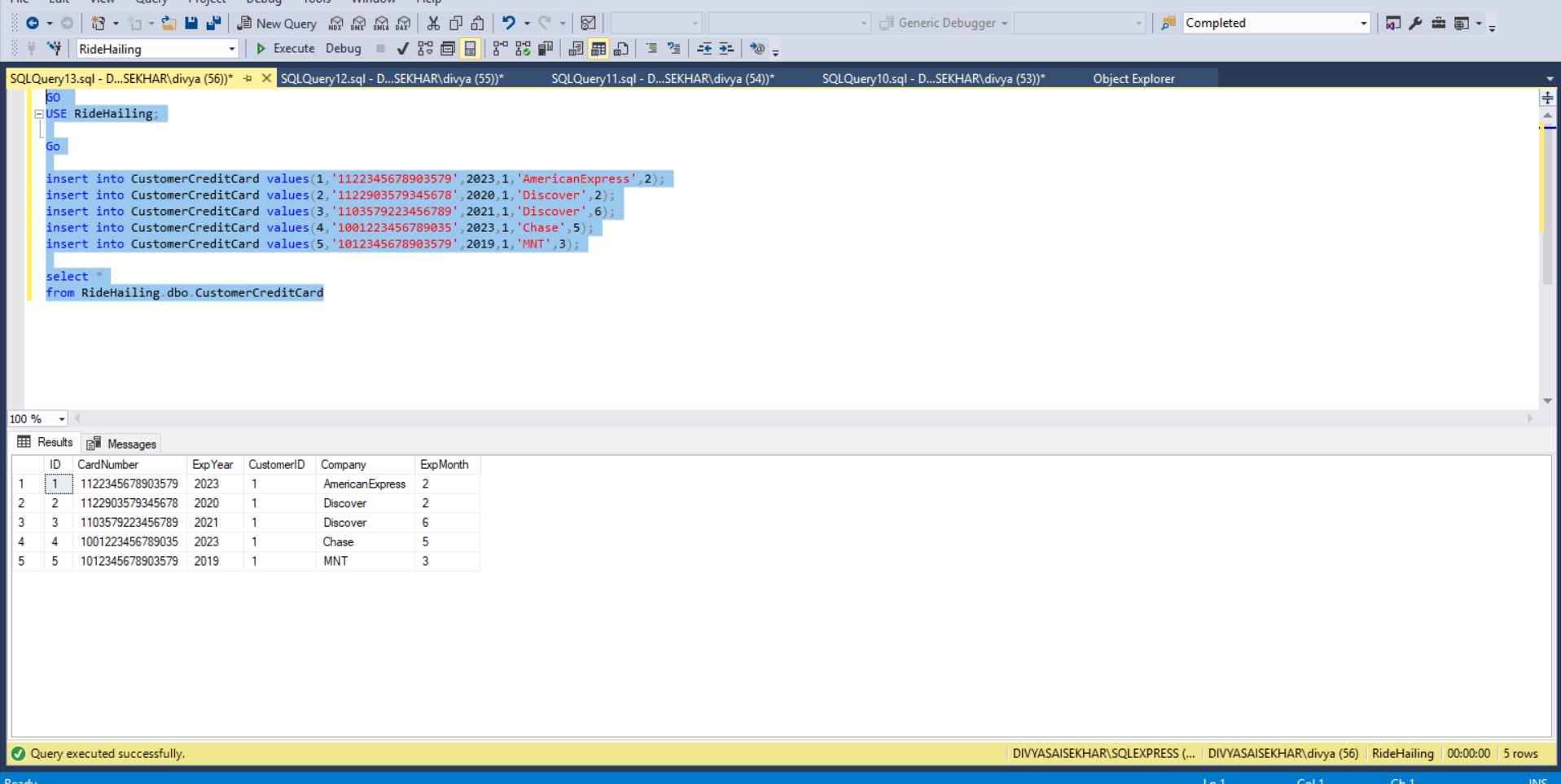
insert into CustomerCreditCard values(3,'1103579223456789',2021,1,'Discover',6);

insert into CustomerCreditCard values(4,'1001223456789035',2023,1,'Chase',5);

insert into CustomerCreditCard values(5,'1012345678903579',2019,1,'MNT',3);

select \*

from RideHailing.dbo.CustomerCreditCard



GO

USE RideHailing;

Go

Insert Into Licence values

(1, '123478CN3A','04/11/2014', '04/14/2024', 'New Jersey' ),

(2,'45BC6N678A','01/11/2018', '01/11/2028','New York' ),

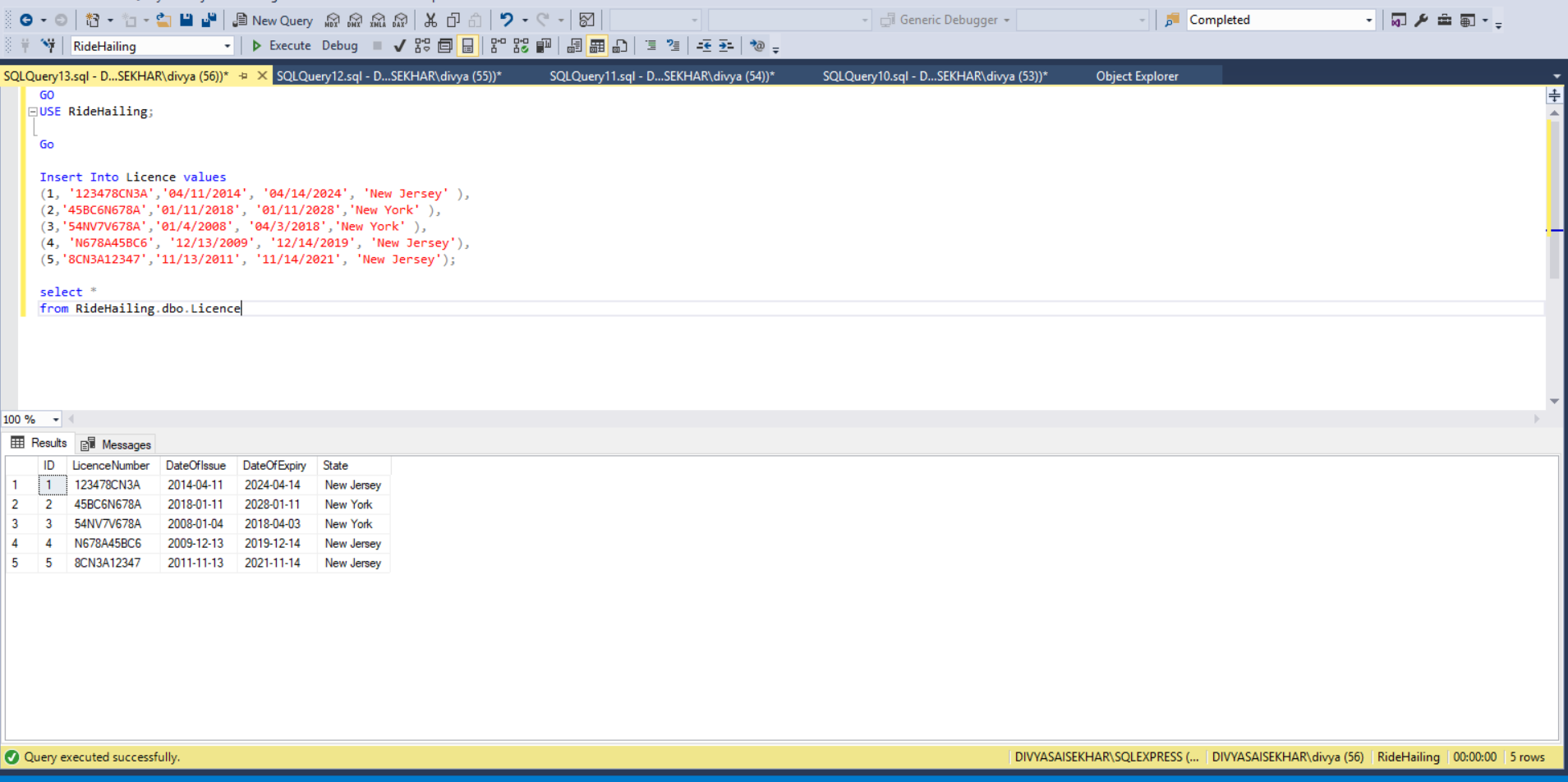
(3,'54NV7V678A','01/4/2008', '04/3/2018','New York' ),

(4, 'N678A45BC6', '12/13/2009', '12/14/2019', 'New Jersey'),

(5,'8CN3A12347','11/13/2011', '11/14/2021', 'New Jersey');

select \*

from RideHailing.dbo.Licence



GO

USE RideHailing;

Go

Insert Into CarInsurance values

(1, 'Bajaj',3245342,'04/11/2018', '04/14/2020'),

(2,'Mahindra',3678945,'01/11/2019', '01/11/2021'),

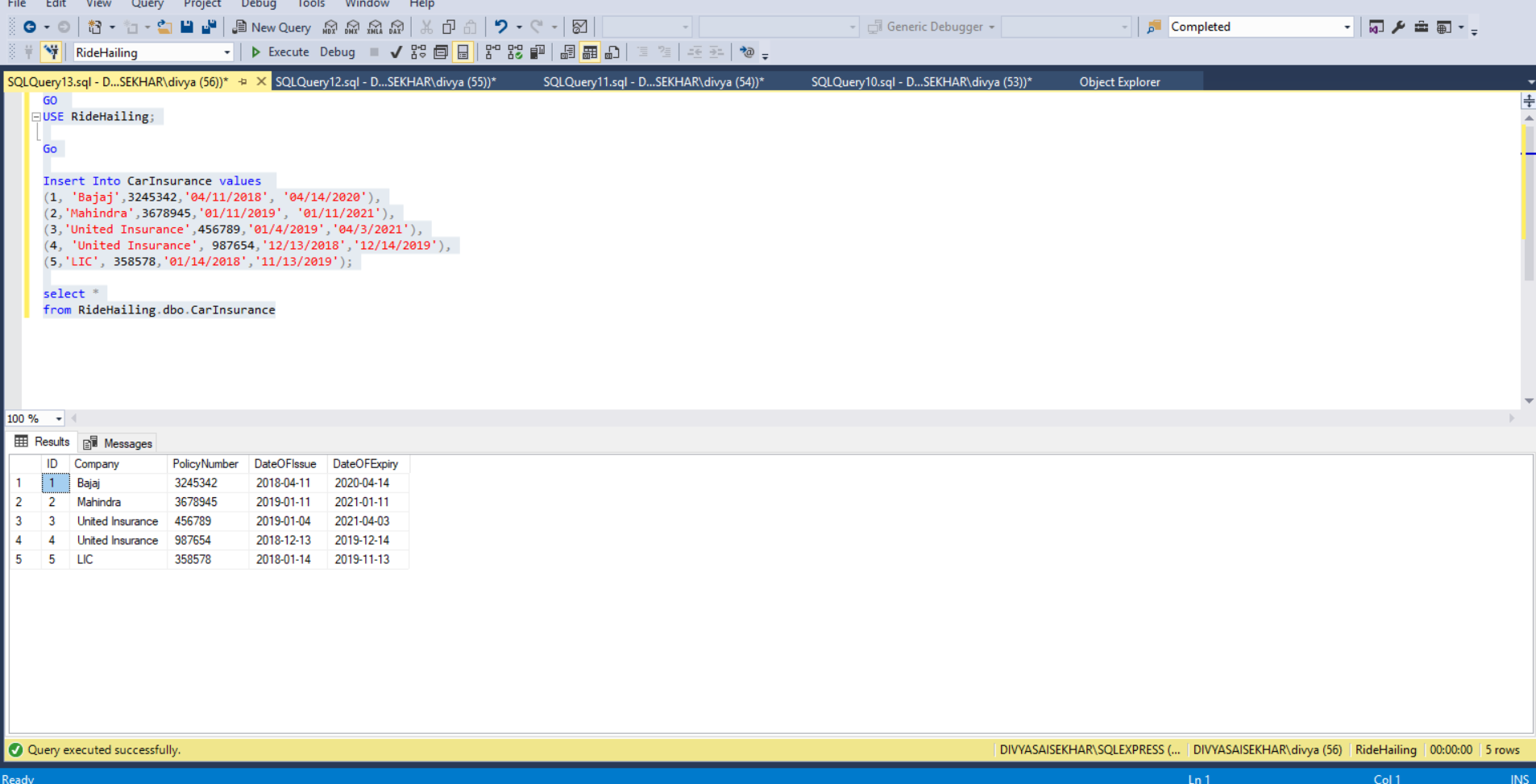
(3,'United Insurance',456789,'01/4/2019','04/3/2021'),

(4, 'United Insurance', 987654,'12/13/2018','12/14/2019'),

(5,'LIC', 358578,'01/14/2018','11/13/2019');

select \*

from RideHailing.dbo.CarInsurance



GO

USE RideHailing;

Go

Insert Into Car values

(1, 'Audi','ACJ7308','JOSFN89897',2004,'Red',4,2,'luxury',1),

(2,'BMW','AKX5657','OKASFGe27497',2017,'White',4,3,'SUV',5),

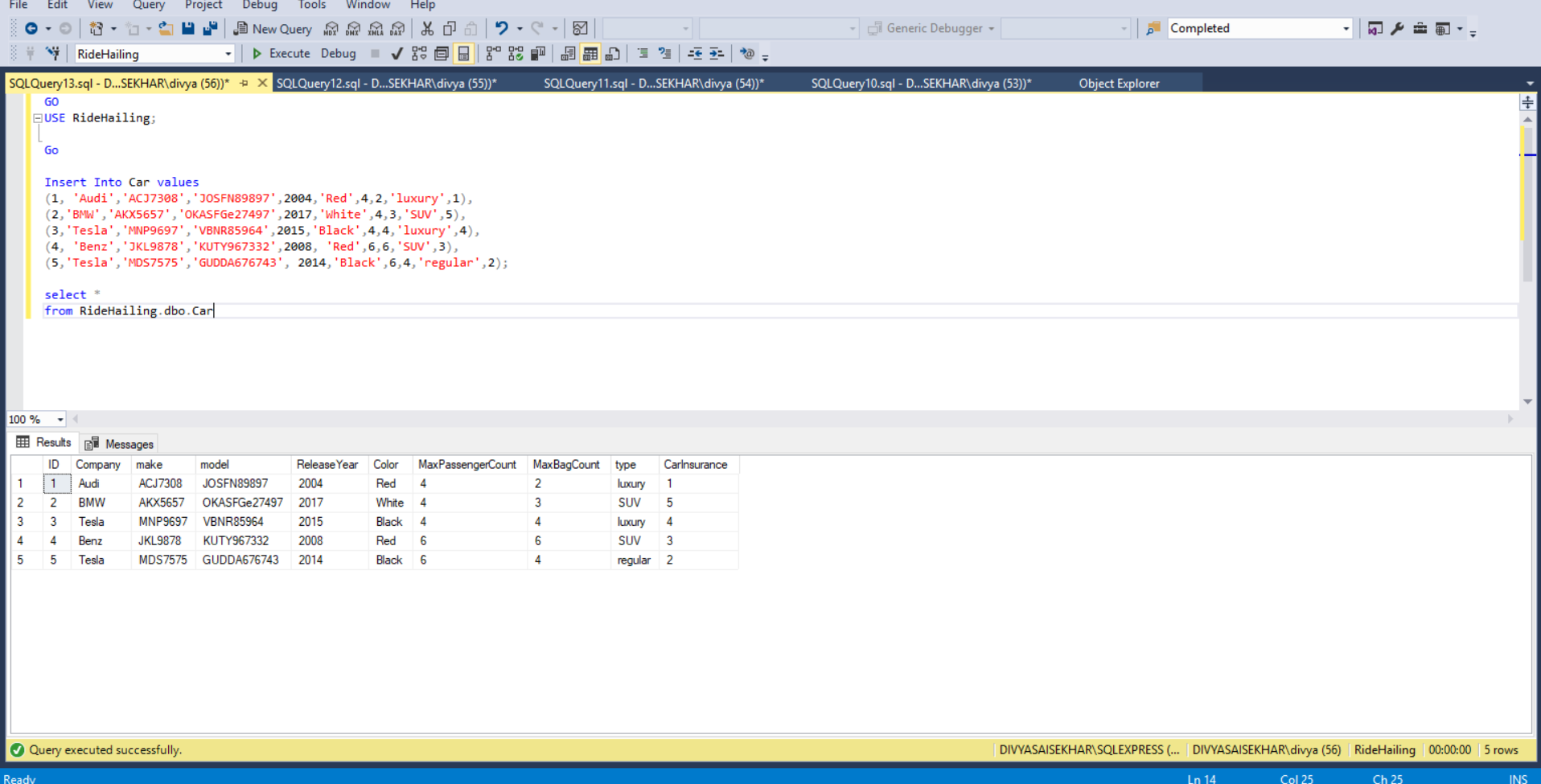
(3,'Tesla','MNP9697','VBNR85964',2015,'Black',4,4,'luxury',4),

(4, 'Benz','JKL9878','KUTY967332',2008, 'Red',6,6,'SUV',3),

(5,'Tesla','MDS7575','GUDDA676743', 2014,'Black',6,4,'regular',2);

select \*

from RideHailing.dbo.Car



GO

USE RideHailing;

Go

Insert Into Driver values

(1,1,'04/14/1992',2010,2,1,7,'loafr@gmail.com','male',1000234,100004,'rahul','ksb'),

(2,2,'04/14/1989',2014,3,2,9,'behnchod@gmail.com','male',1009873,1002334,'ananth','naga'),

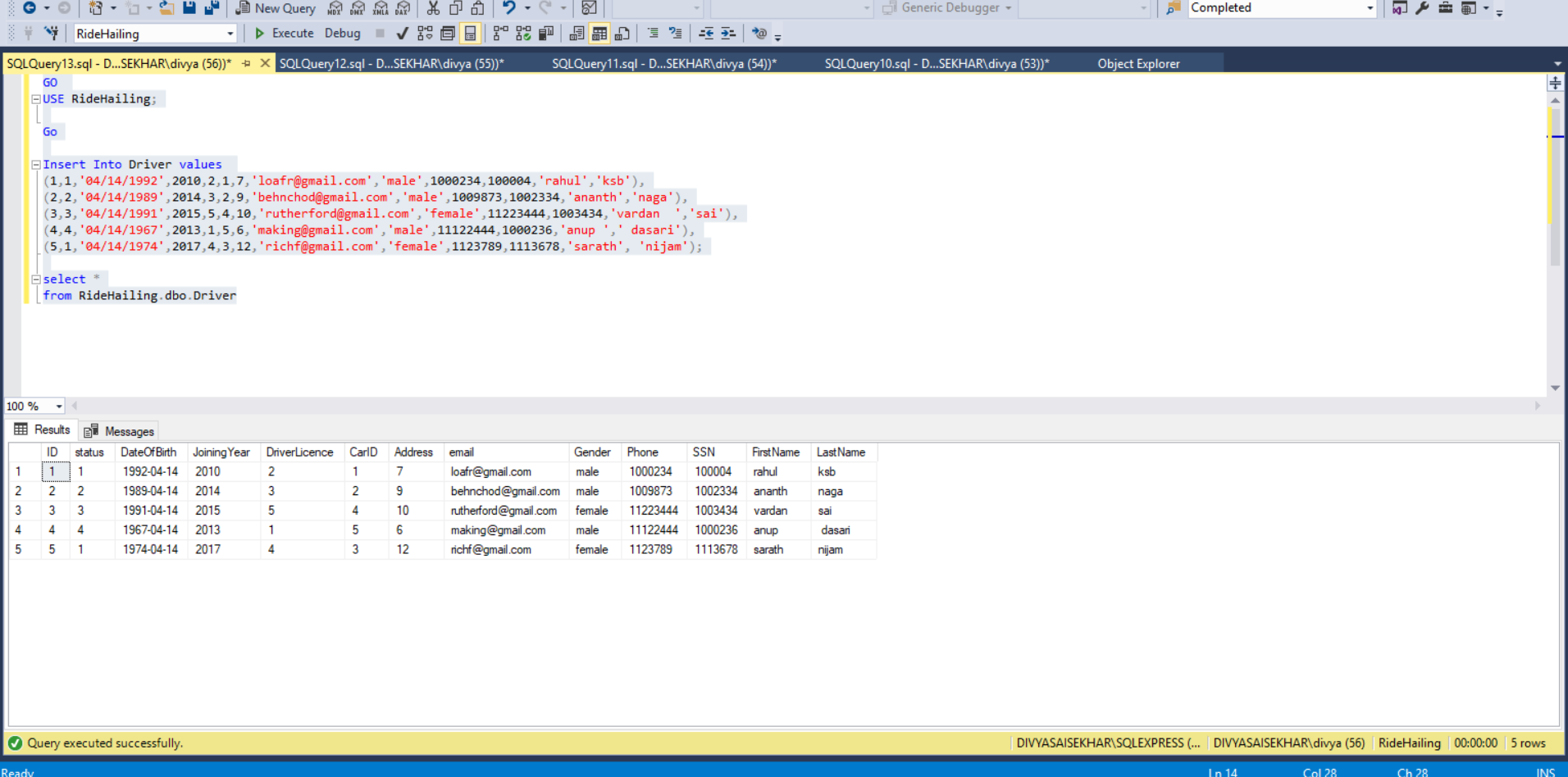
(3,3,'04/14/1991',2015,5,4,10,'rutherford@gmail.com','female',11223444,1003434,'vardan ','sai'),

(4,4,'04/14/1967',2013,1,5,6,'making@gmail.com','male',11122444,1000236,'anup ',' dasari'),

(5,1,'04/14/1974',2017,4,3,12,'richf@gmail.com','female',1123789,1113678,'sarath', 'nijam');

select \*

from RideHailing.dbo.Driver



GO

USE RideHailing;

Go

Insert Into Insurance values

(1, 'Aetna',3245342,'04/11/2018', '04/14/2020', 'ordinary',72.989,1),

(2,'Aetna',3678945,'01/11/2019', '01/11/2021','Preimum',48.968, 2),

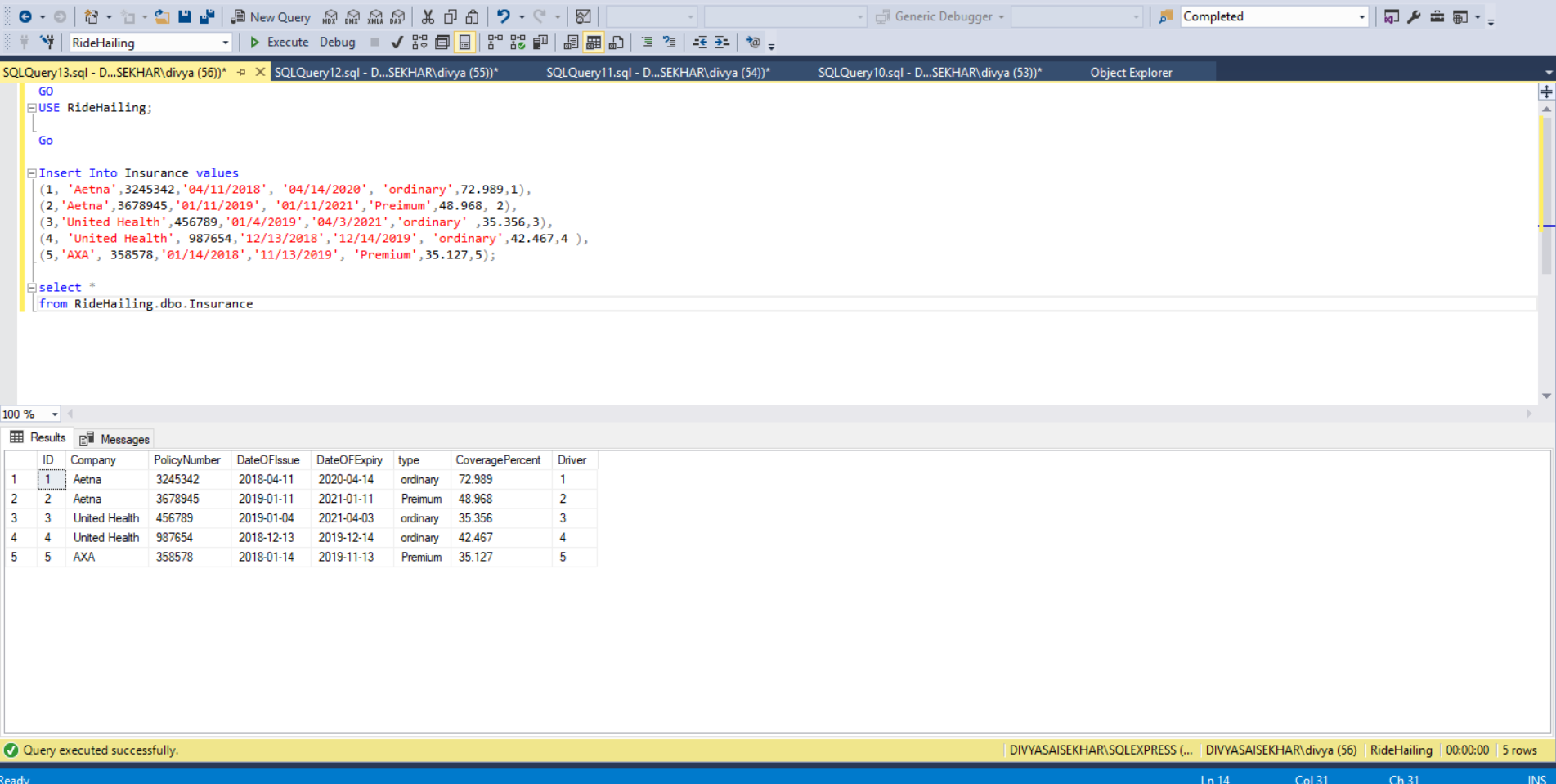
(3,'United Health',456789,'01/4/2019','04/3/2021','ordinary' ,35.356,3),

(4, 'United Health', 987654,'12/13/2018','12/14/2019', 'ordinary',42.467,4 ),

(5,'AXA', 358578,'01/14/2018','11/13/2019', 'Premium',35.127,5);

select \*

from RideHailing.dbo.Insurance



Go

Insert Into Banking values

(1, 'Chase',32454,12345,'Checking', 1,'Active' ,100.000),

(2,'MNT',36784,34567,'Checking', 2,'Active',100.000),

(3,'Chase',45678,23458,'Checking',3,'Active' ,100.000),

(4, 'Key', 98764,345678,'Checking',4, 'Active',100.000 ),

(5,'Key', 35878,234567,'Checking',5,'Active',100.000);

select \*

from RideHailing.dbo.Banking



GO

USE RideHailing;

GO

insert into Trips values (0,'Completed','2011-12-18 13:13:17','2011-12-18 13:23:17','2011-12-18 13:31:17',1,5,5,4,2,4,52.43,6,2);

insert into Trips values (1,'ON Trip','2011-12-18 13:11:17','2011-12-12 13:33:17','2011-12-18 13:35:17',4,8,3,2,3,1,17.34,2,3);

insert into Trips values (2,'Completed' ,'2019-05-03 18:58:37.580','2019-05-03 18:58:37.580', '2019-05-03 18:58:37.580', 3, 5, 3, 1, 4, 1, 52.30, 6.00, 4);

insert into Trips values (3.,'Cancelled','2019-05-03 19:00:14.960','2019-05-03 19:00:14.960','2019-05-03 19:00:14.960', 3, 5, 3, 1, 1, 1, 2.00, 0.00, 1);

insert into Trips values (4,'Completed' ,'2019-05-03 19:02:45.340', '2019-05-03 19:02:45.340','2019-05-03 19:02:45.340', 2, 4, 2, 2, 1, 1, 12.00, 0.00, 1);

insert into Trips values (5,'ON Trip','2011-12-18 13:13:17','2011-12-18 13:23:17','2011-12-18 13:31:17',5,8,2,4,5,5,34.30,2,5);

insert into Trips values (6 ,'Completed','2019-05-03 19:03:35.527', '2019-05-03 19:03:35.527', '2019-05-03 19:03:35.527', 2, 5, 2, 2, 4, 4, 17.00, 3.00, 4);

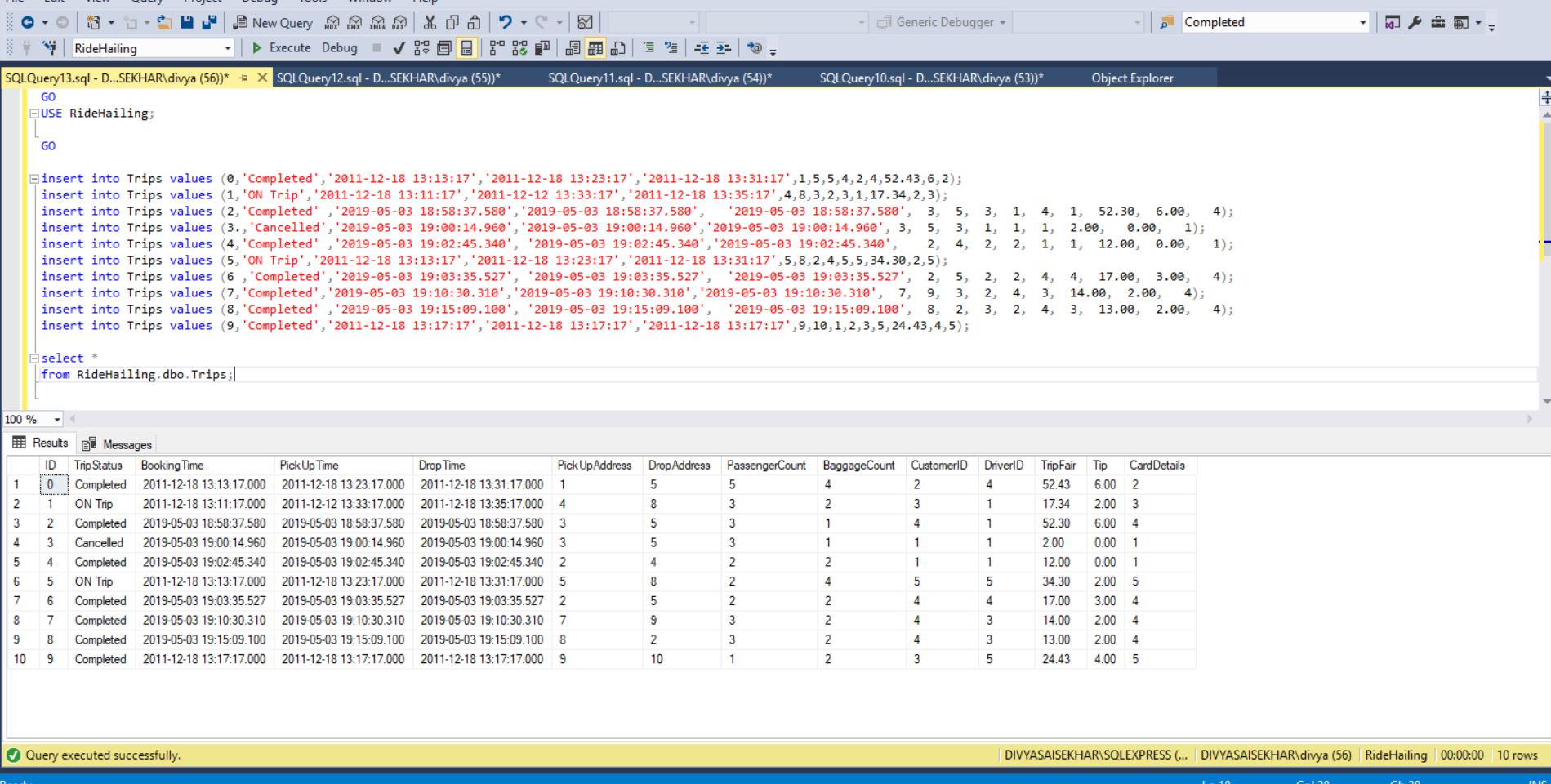
insert into Trips values (7,'Completed','2019-05-03 19:10:30.310','2019-05-03 19:10:30.310','2019-05-03 19:10:30.310', 7, 9, 3, 2, 4, 3, 14.00, 2.00, 4);

insert into Trips values (8,'Completed' ,'2019-05-03 19:15:09.100', '2019-05-03 19:15:09.100', '2019-05-03 19:15:09.100', 8, 2, 3, 2, 4, 3, 13.00, 2.00, 4);

insert into Trips values (9,'Completed','2011-12-18 13:17:17','2011-12-18 13:17:17','2011-12-18 13:17:17',9,10,1,2,3,5,24.43,4,5);

select \*

from RideHailing.dbo.Trips;



GO

USE RideHailing;

GO

insert into DriverRating values(1,1,'decent',4);

insert into DriverRating values(2,2,'hchdna',5);

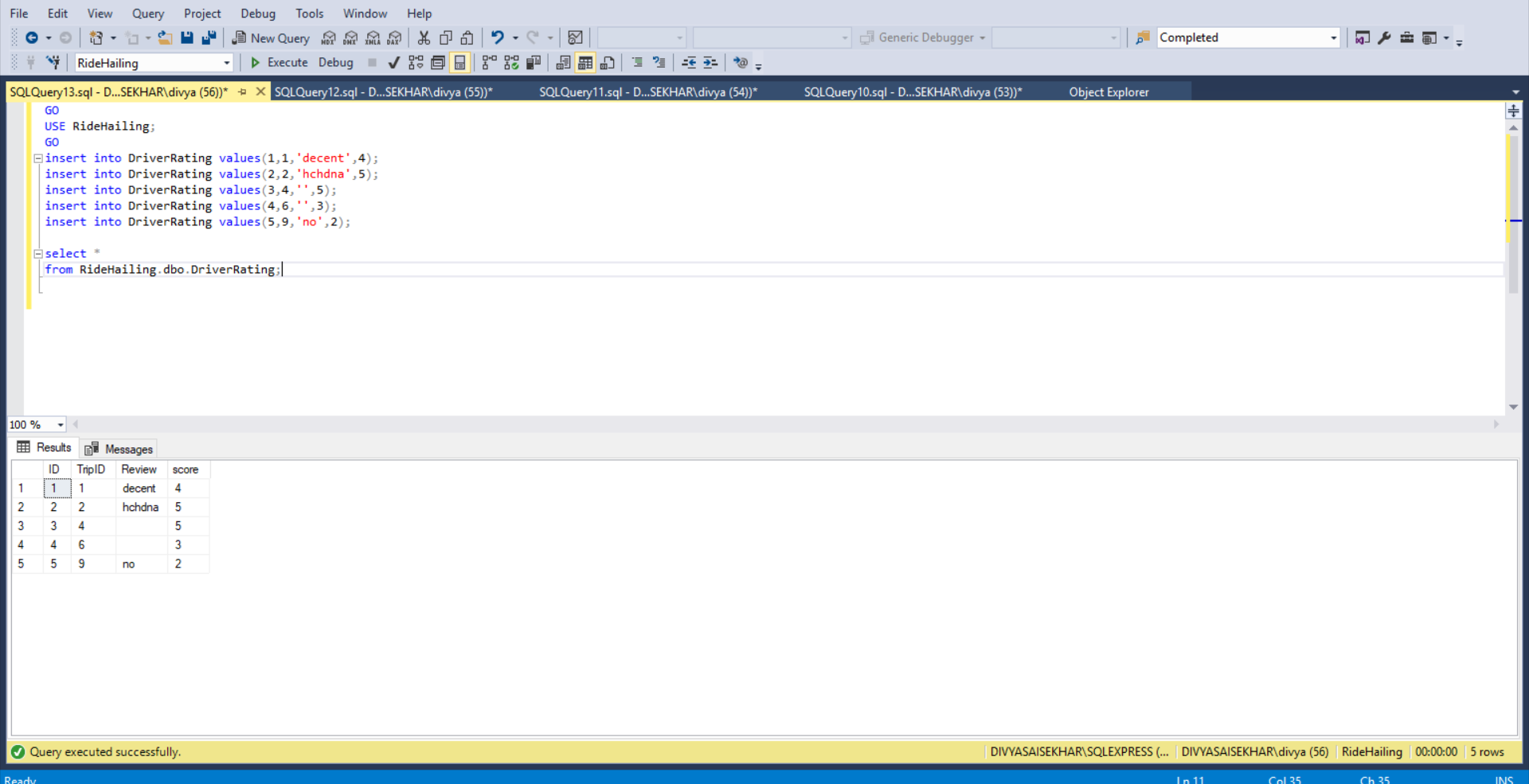
insert into DriverRating values(3,4,'',5);

insert into DriverRating values(4,6,'',3);

insert into DriverRating values(5,9,'no',2);

select \*

from RideHailing.dbo.DriverRating;



GO

USE RideHailing;

Go

insert into CustomerRating values(1,1,'everythn is good',5);

insert into CustomerRating values(2,2,'',4);

insert into CustomerRating values(3,4,'',4);

insert into CustomerRating values(4,6,'',5);

insert into CustomerRating values(5,9,'not interested',3);

select \*

from RideHailing.dbo.CustomerRating;



**Demonstration :**

**View ::**

**View 1:**

This view is the display the average rating of all the customers whoa re using ride hailing application based on the review they received from the drivers.

GO

USE RideHailing;

GO

Go

Create view CustomerAverageRating

as

select Customers.CustomerID,Customers.FirstName, AVG(CustomerRating.score) as rating

from Trips join Customers on Trips.CustomerID = Customers.CustomerID

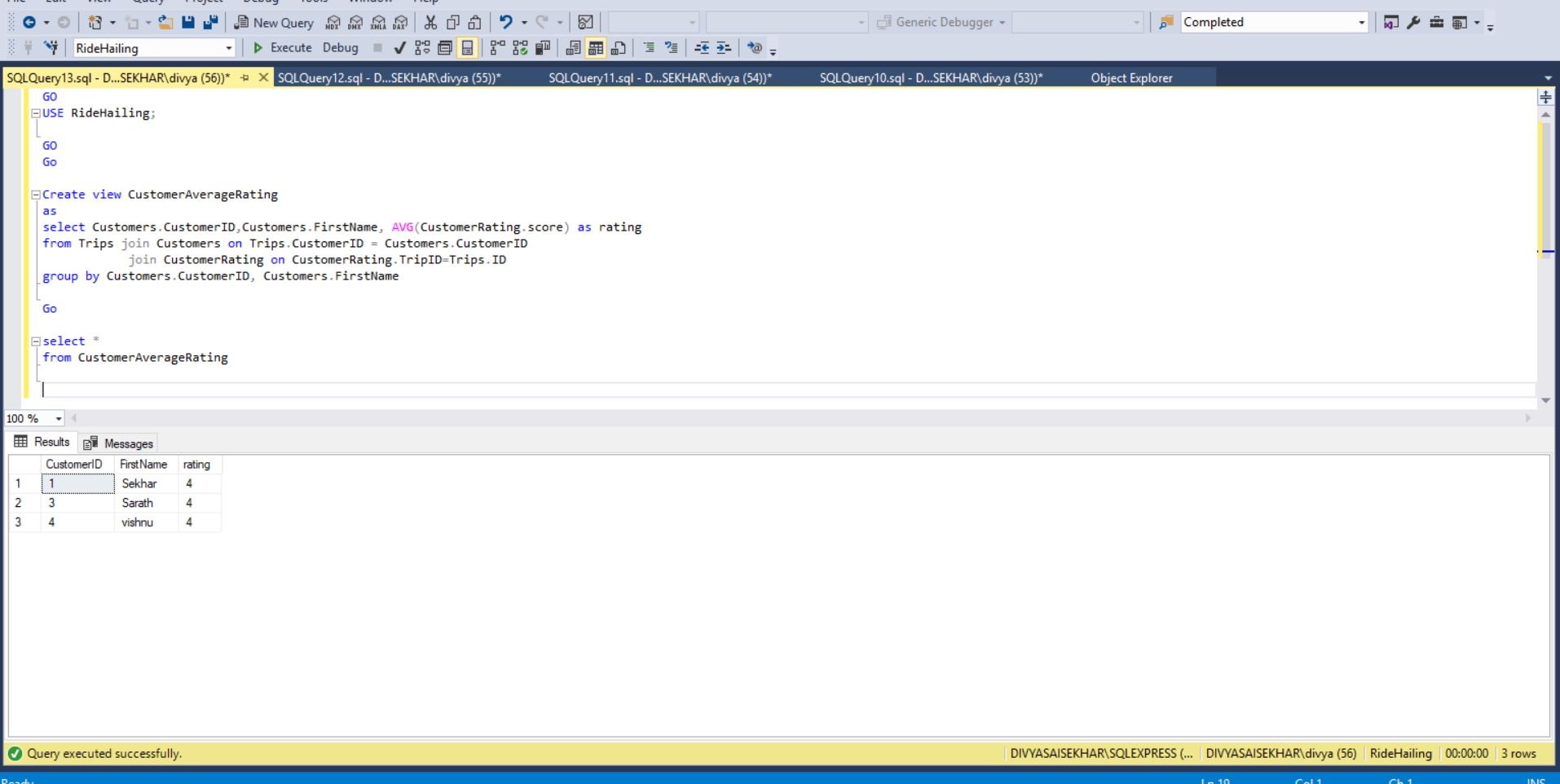
join CustomerRating on CustomerRating.TripID=Trips.ID

group by Customers.CustomerID, Customers.FirstName

Go

select \*

from CustomerAverageRating



**View2:**

TO know the driver and information of that he was register with and its insurance details.

GO

USE RideHailing;

GO

Create view DriverCarInsurance as

select Driver.ID, Driver.FirstName, Driver.LastName, Driver.email,Car.type, CarInsurance.Company, CarInsurance.PolicyNumber

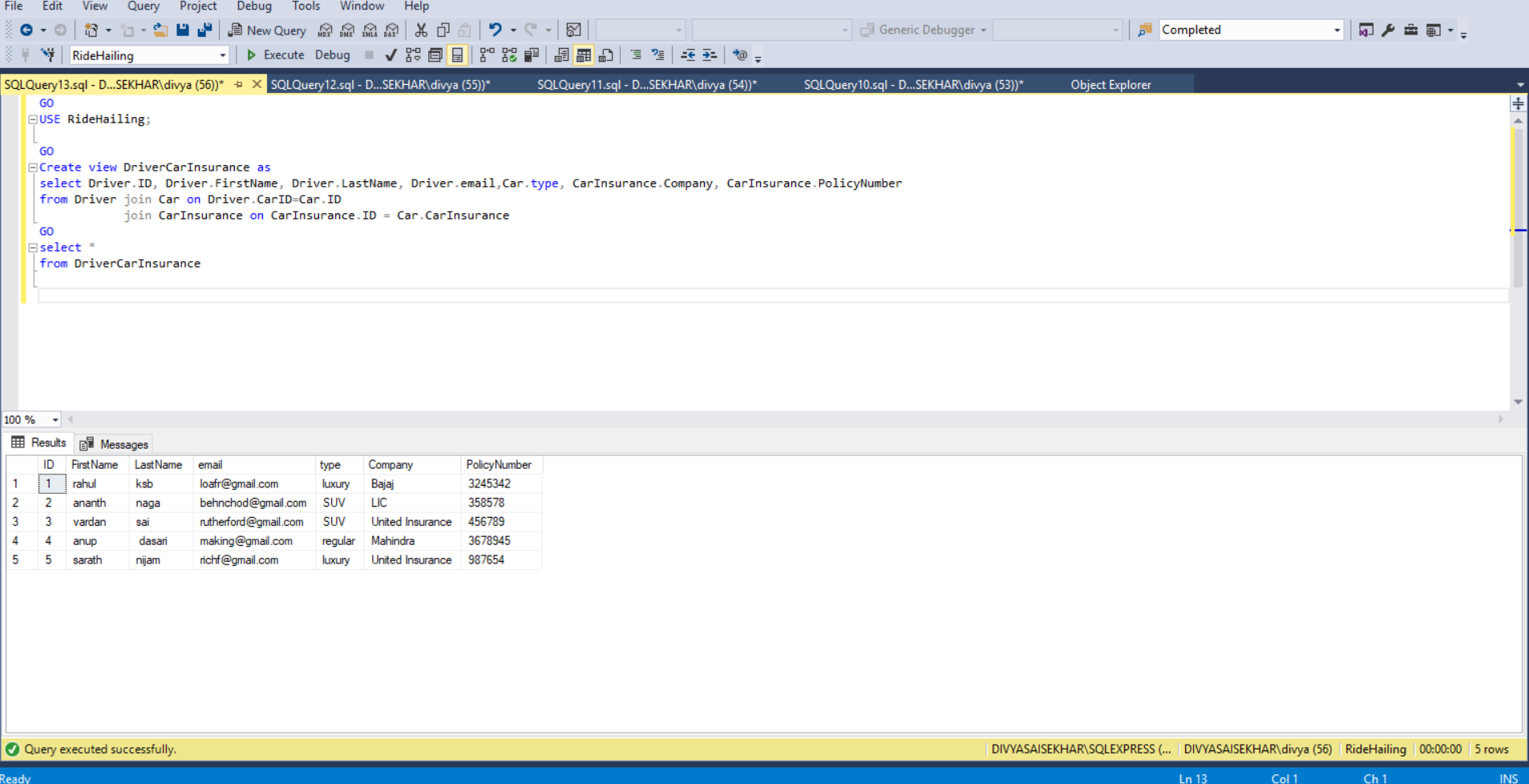
from Driver join Car on Driver.CarID=Car.ID

join CarInsurance on CarInsurance.ID = Car.CarInsurance

GO

select \*

from DriverCarInsurance



**View 3:**

TO know the licence details for all the drivers.

GO

USE RideHailing;

GO

Create view DriverLicenceInfo as

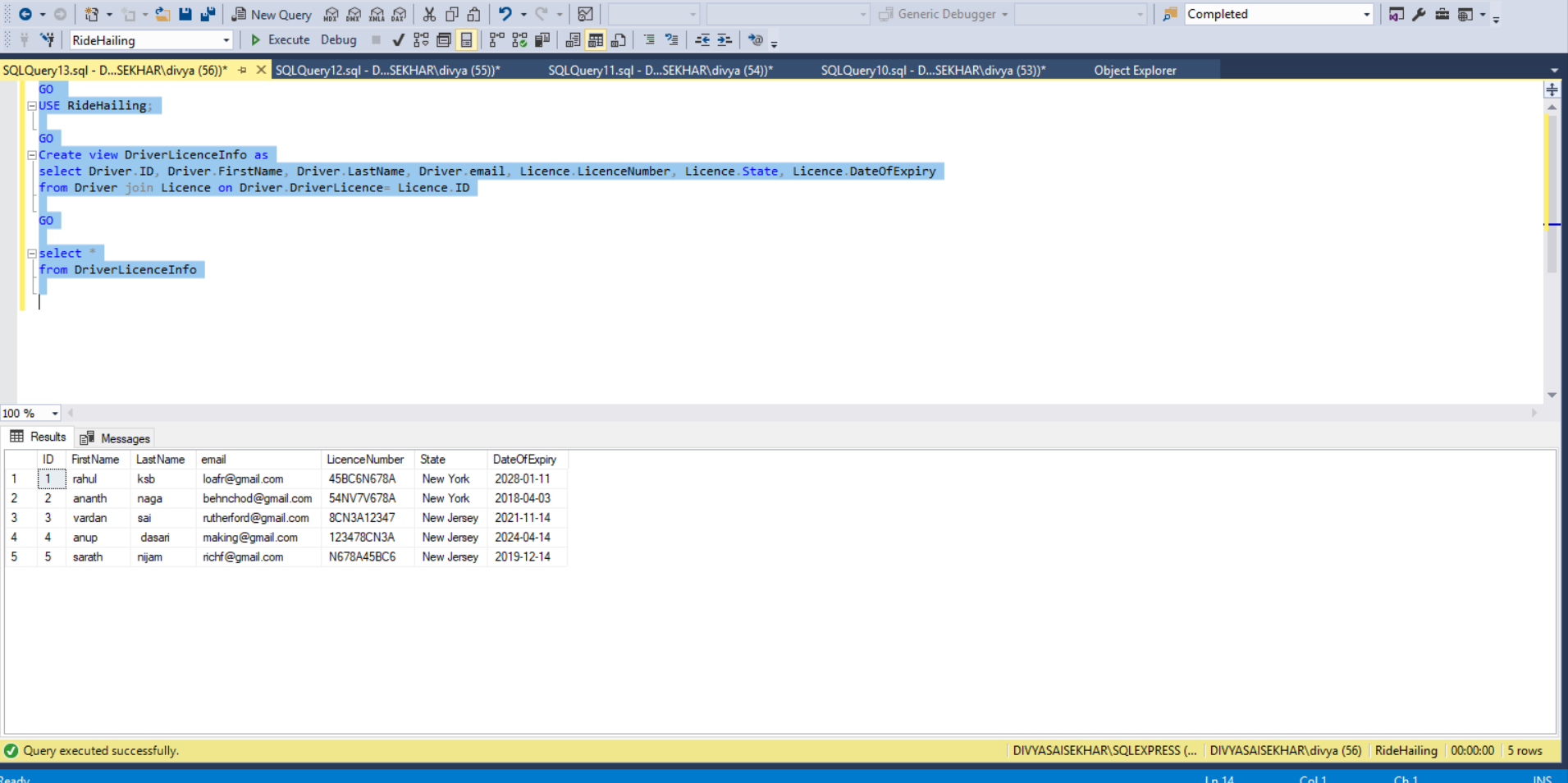
select Driver.ID, Driver.FirstName, Driver.LastName, Driver.email, Licence.LicenceNumber, Licence.State, Licence.DateOfExpiry

from Driver join Licence on Driver.DriverLicence= Licence.ID

GO

select \*

from DriverLicenceInfo



**View 4::**

This view gives details of all the drivers so that all trips they have completed.

GO

Create view DriverTripsCounts as

select Driver.ID, Driver.FirstName, Driver.LastName, count(\*) as TotalTrips

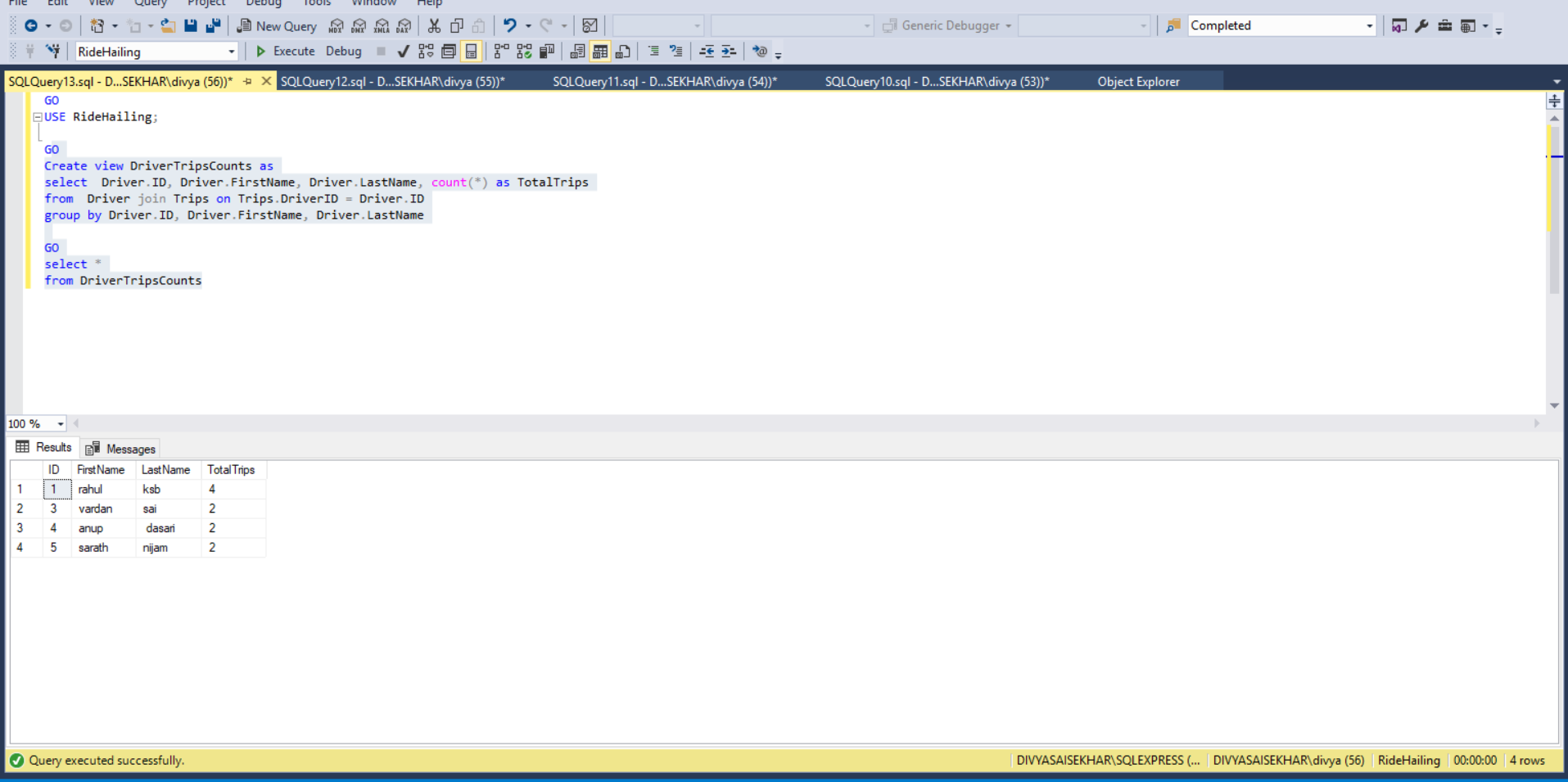
from Driver join Trips on Trips.DriverID = Driver.ID

group by Driver.ID, Driver.FirstName, Driver.LastName

GO

select \*

from DriverTripsCounts



**Functions::**

**Function1:**

**For a particular driver input his corresponding drivers income was returned.**

GO

USE RideHailing;

Go

Create Function DriverIncome(@DriverID int)

Returns money

as

begin

declare @trip money

declare @tip money

select @trip=SUM(TripFair),@tip=SUM(Tip)

from Trips

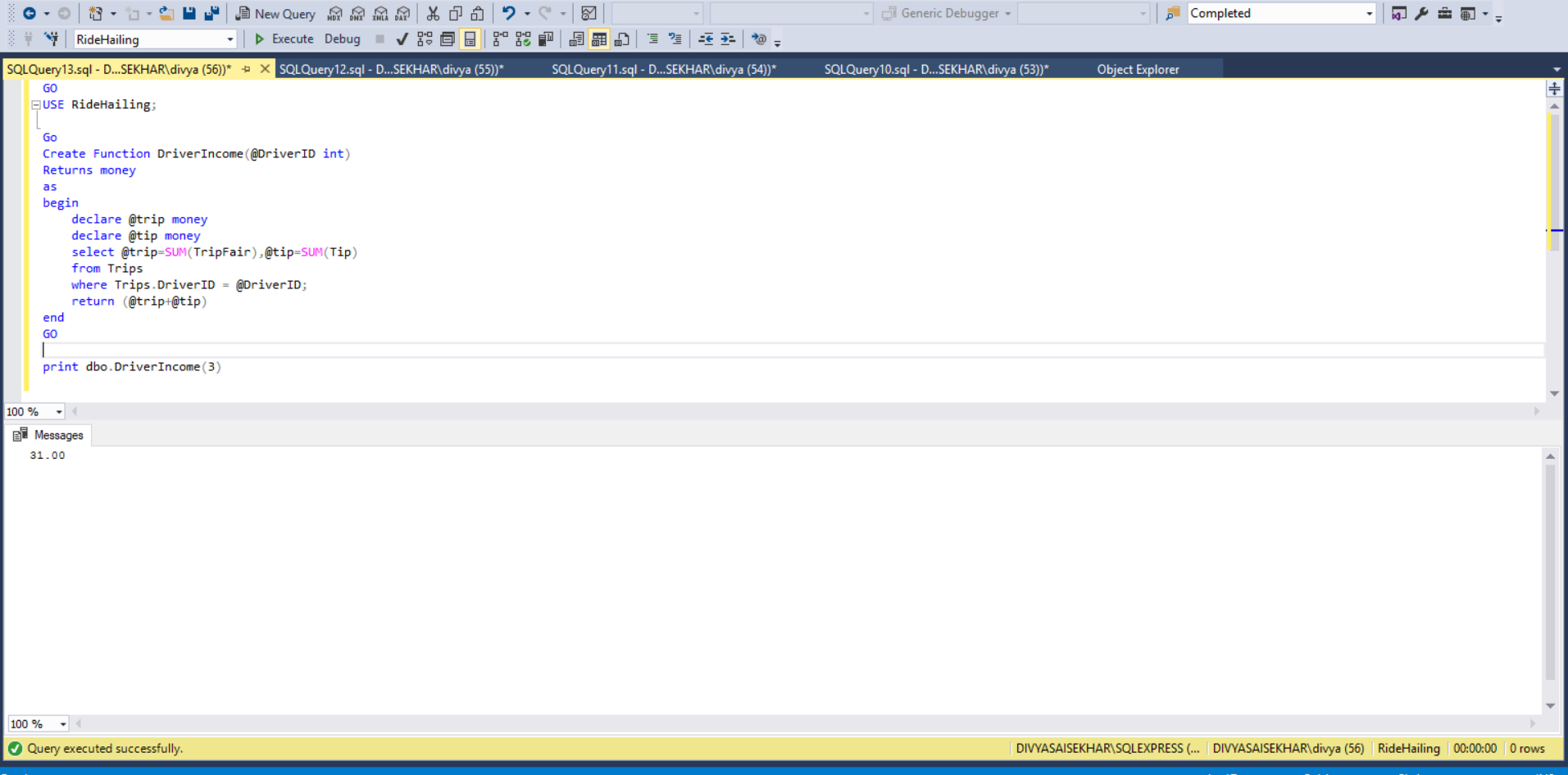
where Trips.DriverID = @DriverID;

return (@trip+@tip)

end

GO

print dbo.DriverIncome(3)



**Function 2:**

**To determined total number of drivers available.**

GO

USE RideHailing;

Go

Create Function AvailableDrivers()

Returns int

as

begin

return (select Count(\*)

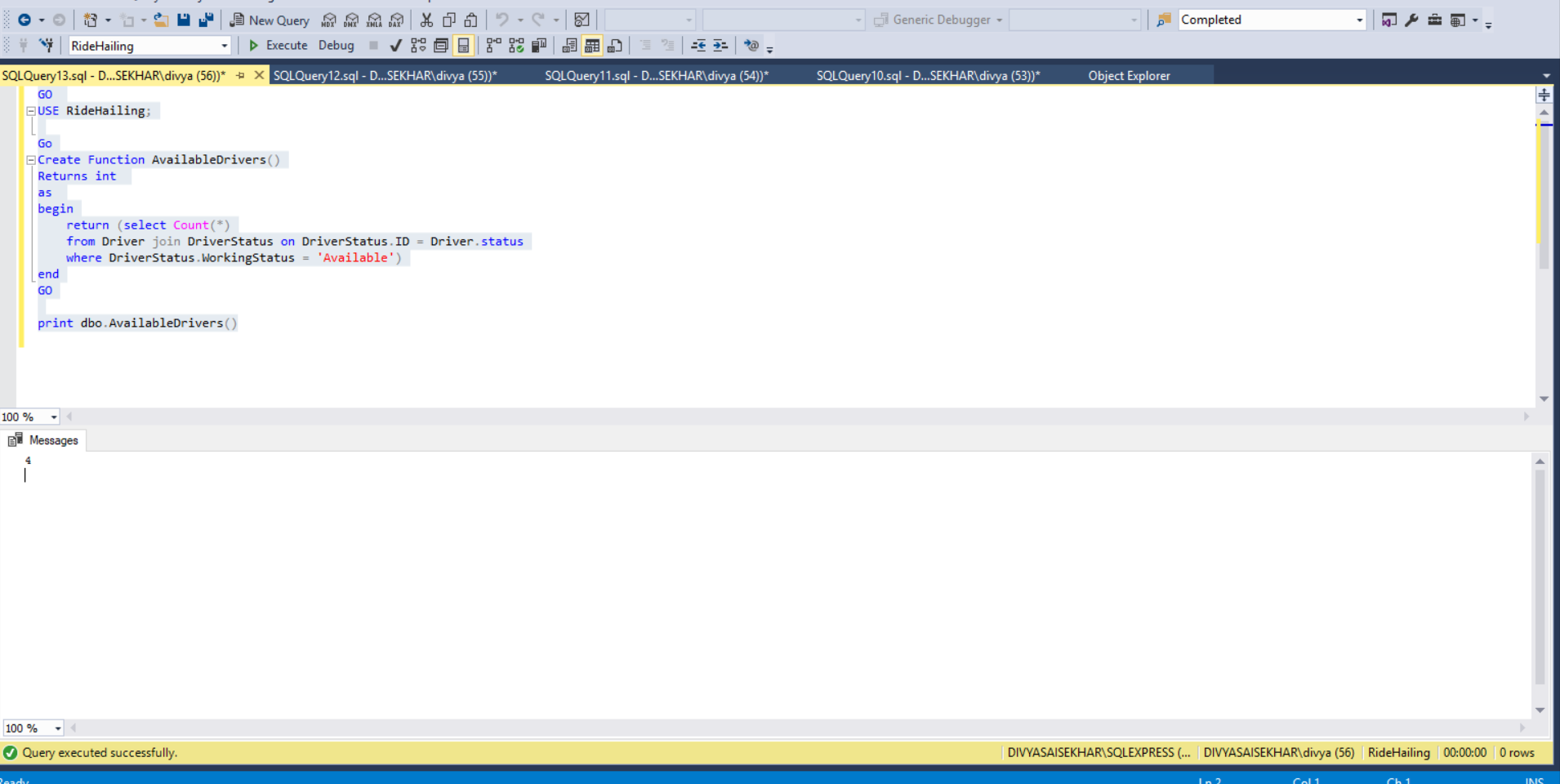
from Driver join DriverStatus on DriverStatus.ID = Driver.status

where DriverStatus.WorkingStatus = 'Available')

end

GO

print dbo.AvailableDrivers()



**Store Procedure:**

**Store Procedure 1:**

**To get the customer rating for a customer.**

Go

Create PROC CustomerRate

@id int,

@score int OUTPUT

as

begin

select @score=AVG(CustomerRating.score)

from Trips join Customers on Trips.CustomerID = Customers.CustomerID

join CustomerRating on CustomerRating.TripID=Trips.ID

where Customers.CustomerID = @id

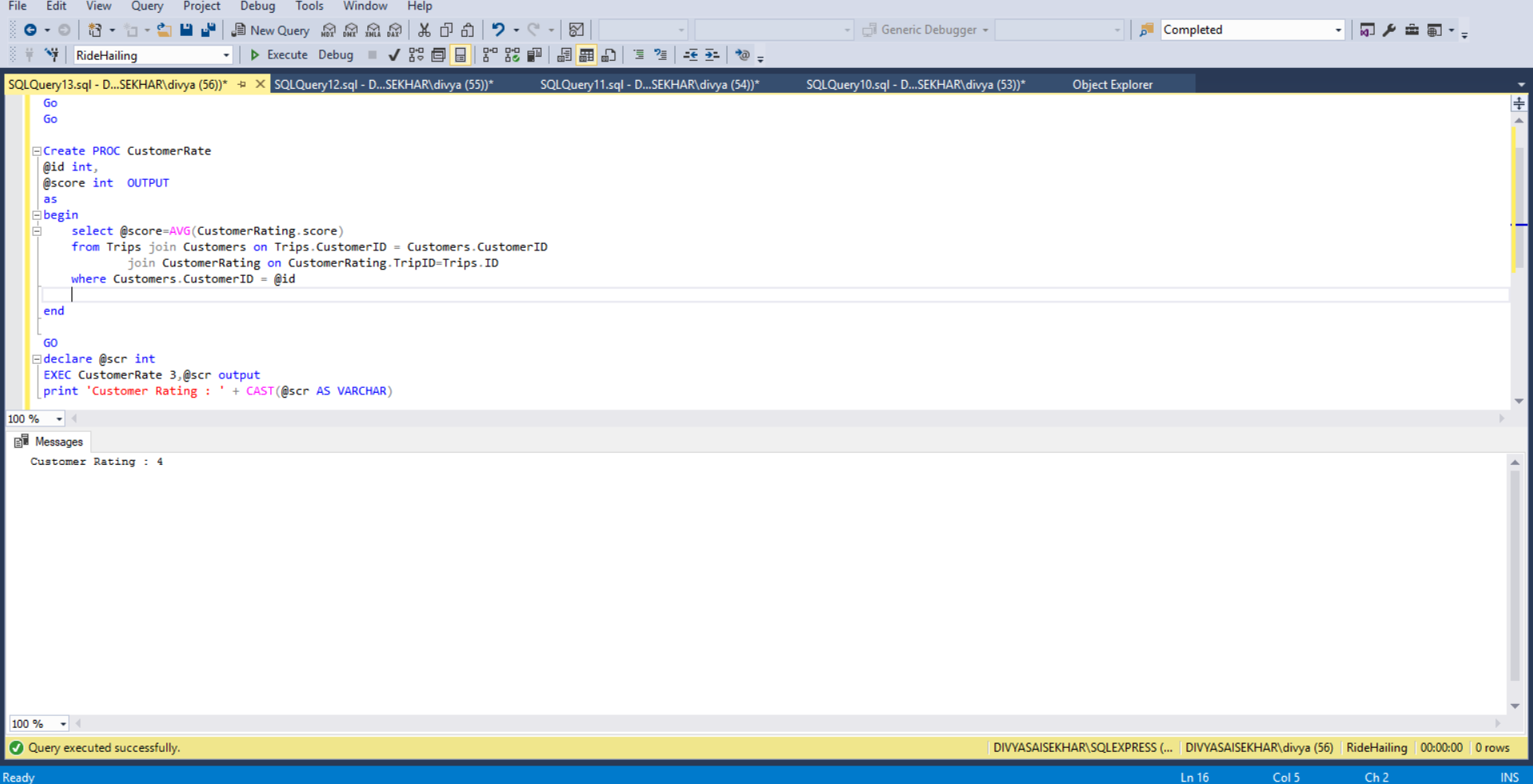
end

GO

declare @scr int

EXEC CustomerRate 3,@scr output

print 'Customer Rating : ' + CAST(@scr AS VARCHAR)



**Store Procedure 1:**

**To get the total number of available cars .**

GO

Create PROC AvailableCar

@outp int output

as

begin

select @outp= count(\*)

from Driver join DriverStatus on Driver.status = DriverStatus.ID

where DriverStatus.WorkingStatus='Available'

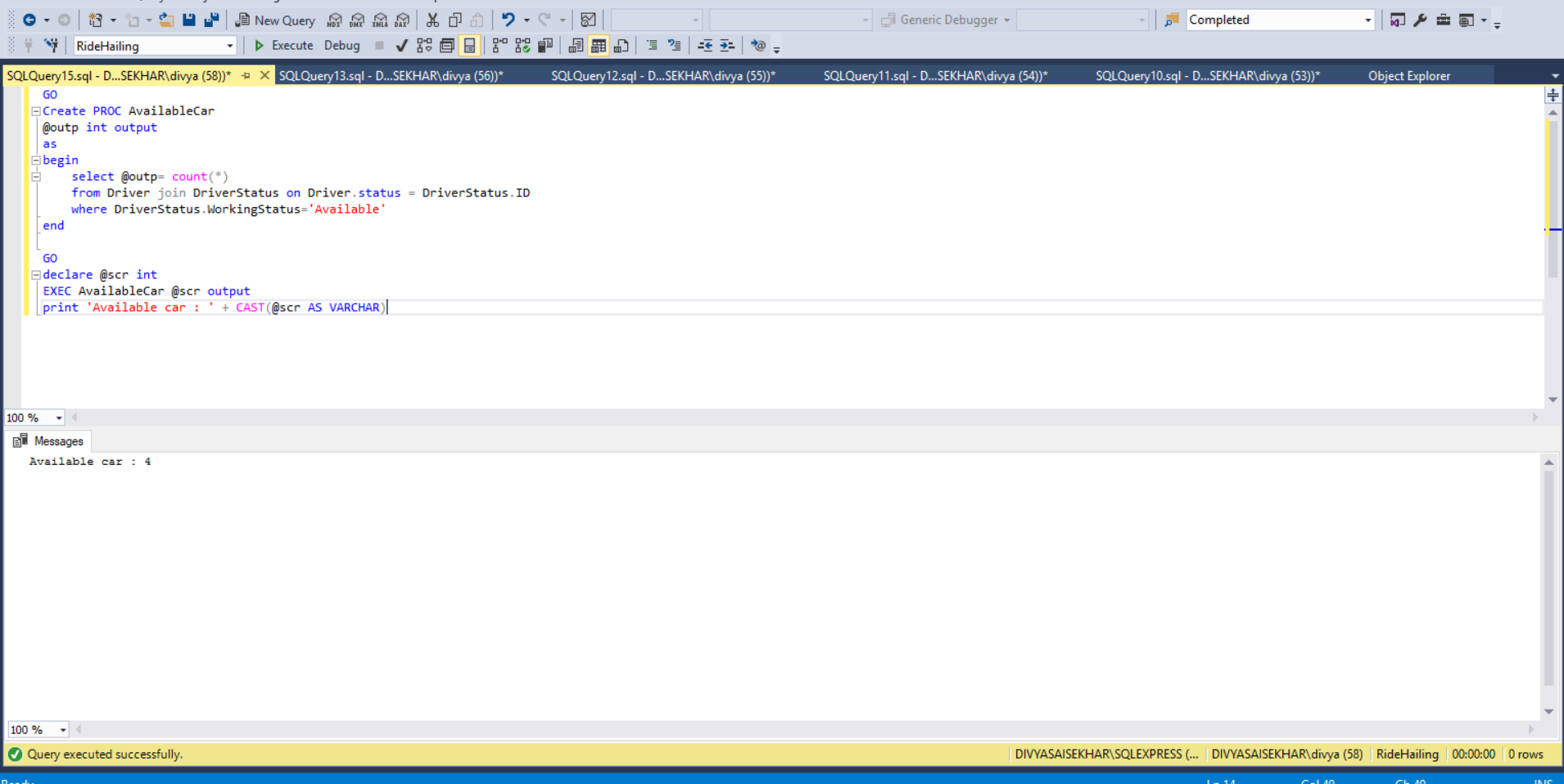
end

GO

declare @scr int

EXEC AvailableCar @scr output

print 'Available car : ' + CAST(@scr AS VARCHAR)



**Roles:**

**Creating a role to assign to a user with permissions for insert on drivers and insert and update Trips.**

Go

USE RideHailing

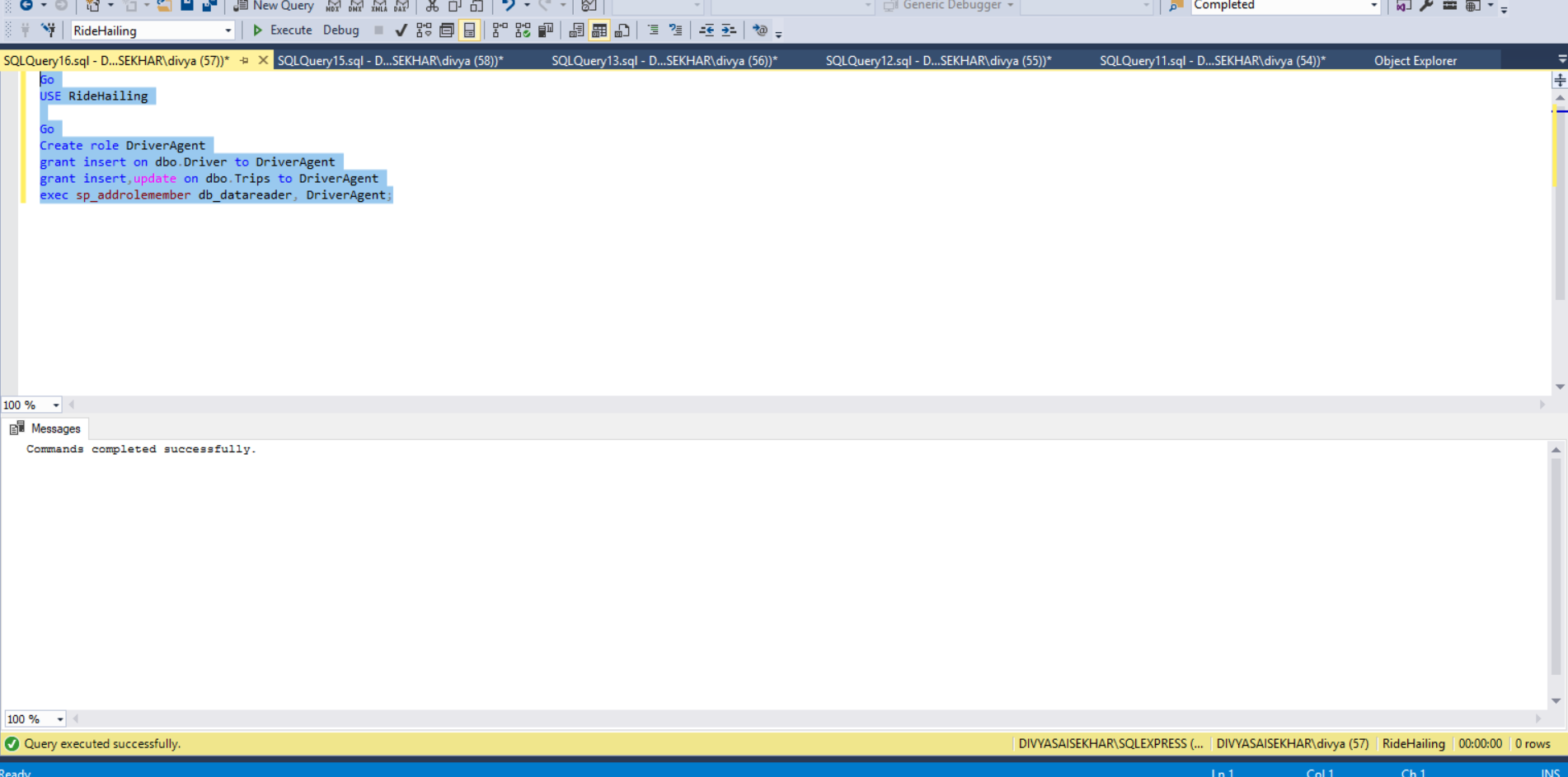
Go

Create role DriverAgent

grant insert on dbo.Driver to DriverAgent

grant insert,update on dbo.Trips to DriverAgent

exec sp\_addrolemember db\_datareader, DriverAgent;



**Roles2:**

Create login divyasai with username divyas assigning role DriverAgent to this user

Go

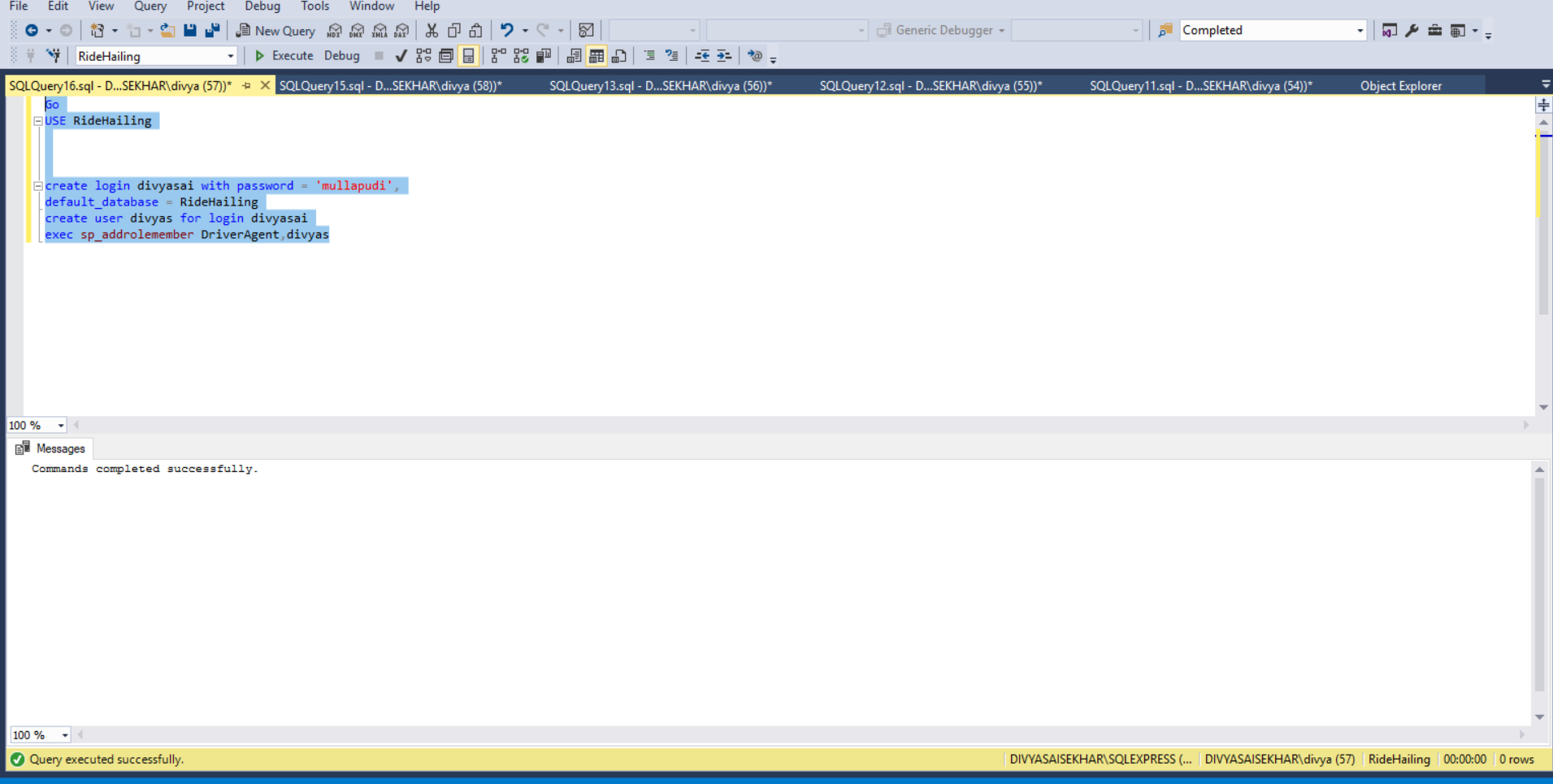
USE RideHailing

create login divyasai with password = 'mullapudi',

default\_database = RideHailing

create user divyas for login divyasai

exec sp\_addrolemember DriverAgent,divyas



**Concluions and Remarks:**

Created a cloned database for the Uber as Ride hailing. Understood the business requirement. Implemented the and designed the database as E/R diagrams, following third normal forms and in SQL server. Created triggers, views, Stored Procedures , functions and roles for testing and population data using DDL , DML , DRL for above tasks. Understood the performance, complexity, security integrity and roles to that have importance in the database design, implementation and testing

The measures of performance, security and integrity are being addressed during the design, implementation and testing