



UBER COMPETITOR DATABASE DESIGN

Abstract

This is the project of designing, implementing and testing a database for uber competitor. All requirements listed are satisfied. It is aggregated all of knowledges I learned in Database management system class in 2019 Spring Semester.

NAME:TIANXIANG YI

SUID:312944880

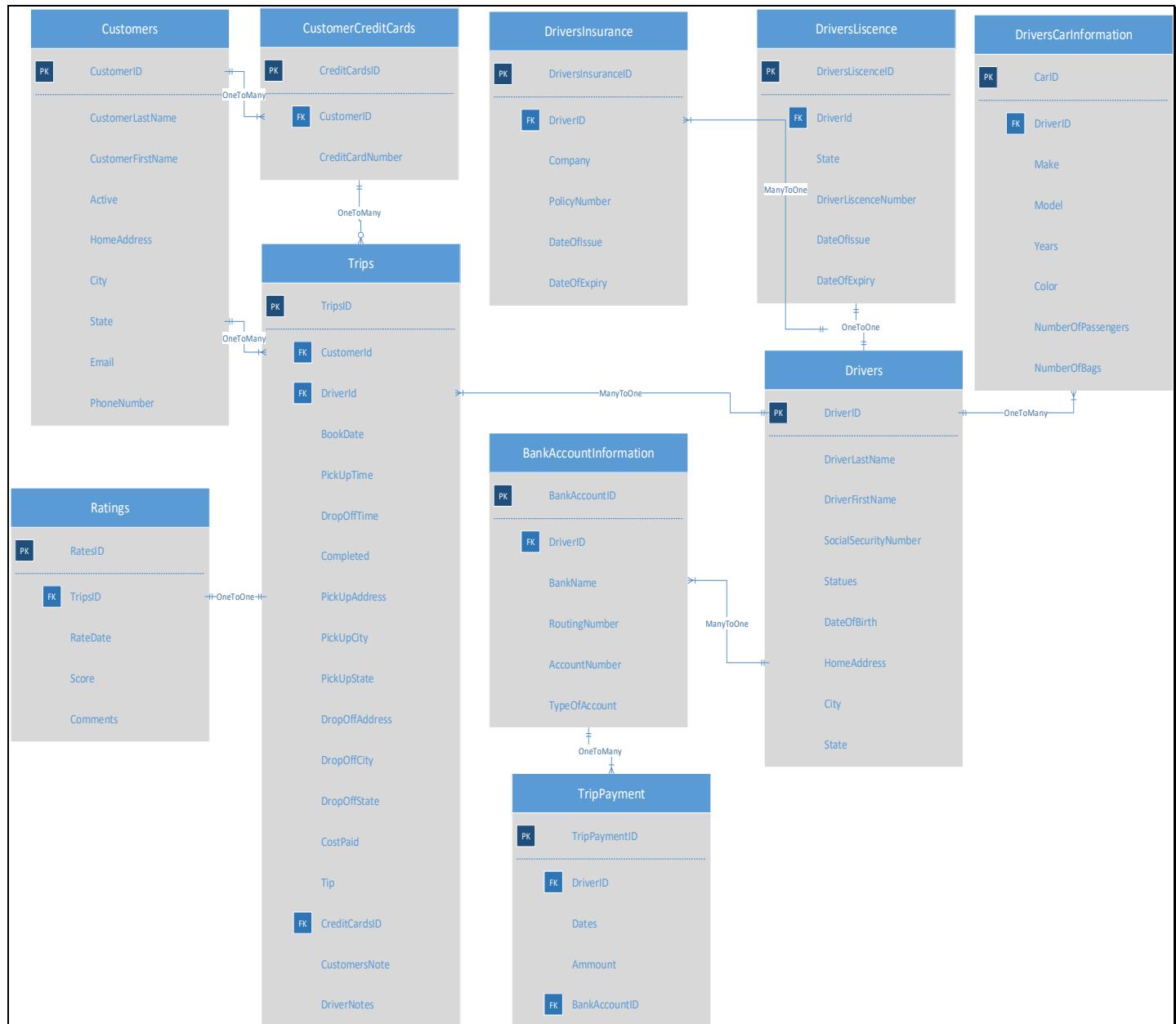
1. Design

1.1 Introduction

The Uber Competitor Database Design maps the logical data model to the target database management system with consideration to the system's performance requirements. The Database Design converts logical data like customer, driver, trips and rates to physical storage constructs (tables) of the target DBMS

1.2 Description

In this Uber Competitor Database Design Project, I implement 10 tables and each table have lots of attribute.



This is Entity/Relationship Diagram of my Design. You could see that it store all of the necessary information need to store, each table has relationship between them and some tables get Foreign Key Constraints to satisfy business logic requirement, all the table names are satisfy the ‘CamelCase’ requirement. In the following code, you could see I build four business report with four Views:BadRatingTrip, DriverDetailedsInformation, CardInformation and HistoryTrips. I using scripts to estabilish the security level with Database Login ‘TianxiangYi’ and password ‘123456’. Also, I create two stored procedures to show ‘Total number of customer oders’ and ‘Driver pickup records in one month’ and two functions to show company income ,expendure and balance in one month.

2. Implementation

2.1 Brief Description

In the implementation part, first I normalize all the table to 3rd Normal Form. For example, I divide the address to address, city, state and divide the name to first name and last name to satisfy the attribute atomic. Then I decide which attribute is only depend on the Customer, Driver and Trips to group them in different table to satisfy 2nd and 3rd Normal Form.

Second, I determines primary keys, each attribute/column datatype, nullabilities in table. Every table I generate primary key like 'XXX'ID for the purpose of querying fast. Then I consider the relationship between tables and tables.and set foreign key constrain. For example, the Relationship between Customers and Drivers in many to many relationship. So I implement a new table Trips which is relationship table to make each to Trips is one to many relationship and set CustomerID and DriverID is foreign key in Table Trips.

Third, I consider some potential integrity and security issues about the implementation I done. I find out that because I implement lots of foreign constraints, so when you drop tables, you should consider it carefully. Also in some table like CustomerCreditCard, I should use the composite primary key composed of CustomerID and StoredCardSequence which is more safety for paying money. But I use the primary key CardID, it is more efficient in querying but lower safety level.

2.2 Code of creation of the database

Create 10 table with right datatypes, nullabilities and relationship in SQL

```
“
CREATE DATABASE UberCompetitor
GO

USE UberCompetitor
CREATE TABLE Customers
(CustomerID INT PRIMARY KEY IDENTITY,
CustomerLastName VARCHAR(50) NOT NULL,
CustomerFirstName VARCHAR(50) NOT NULL,
Active BIT NOT NULL,
HomeAddress VARCHAR(50) NOT NULL,
City VARCHAR(50) NOT NULL,
State VARCHAR(50) NOT NULL,
Email VARCHAR(50) NOT NULL,
PhoneNumber VARCHAR(50) NOT NULL,
)

CREATE TABLE CustomerCreditCards
(CreditCardID INT PRIMARY KEY IDENTITY,
```

```
CustomerID INT FOREIGN KEY REFERENCES Customers(CustomerID) NOT NULL,  
CreditCardNumber VARCHAR(50) NOT NULL)
```

```
CREATE TABLE Drivers(  
DriverID INT PRIMARY KEY IDENTITY,  
DriverLastName VARCHAR(50) NOT NULL,  
DriverFirstName VARCHAR(50) NOT NULL,  
SocialSecurityNumber VARCHAR(50) NOT NULL,  
Statuses VARCHAR(50) NOT NULL,  
DateOfBirth SMALLDATETIME NOT NULL,  
  
HomeAddress VARCHAR(50) NOT NULL ,  
City VARCHAR(50) NOT NULL,  
State VARCHAR(50) NOT NULL  
)
```

```
CREATE TABLE DriverLiscence(  
DriverLiscenceID INT PRIMARY KEY IDENTITY,  
DriverID INT FOREIGN KEY REFERENCES Drivers(DriverID) NOT NULL,  
State VARCHAR(50) NOT NULL,  
DriverLiscenceNumber VARCHAR(50) NOT NULL,  
DateOfIssue SMALLDATETIME NOT NULL,  
DateOfExpiry SMALLDATETIME NOT NULL  
)
```

```
CREATE TABLE DriversCarInformation(  
CarID INT PRIMARY KEY IDENTITY,  
DriverID INT FOREIGN KEY REFERENCES Drivers(DriverID) NOT NULL,  
Make VARCHAR(50) NOT NULL,  
Model VARCHAR(50) NULL,  
Years INT NOT NULL,  
Color VARCHAR(50) NOT NULL,  
NumberOfPassengers INT NULL,  
NumberOfBags INT NULL)
```

```
CREATE TABLE DriverInsurance(  
DriverInsuranceID INT PRIMARY KEY IDENTITY,  
DriverID INT FOREIGN KEY REFERENCES Drivers(DriverID) NOT NULL,  
Company VARCHAR(50) NOT NULL,  
PolicyNumber VARCHAR(50) NOT NULL,  
DateOfIssue SMALLDATETIME NOT NULL,  
DateOfExpiry SMALLDATETIME NOT NULL)
```

```
CREATE TABLE BankAccountInformation(
```

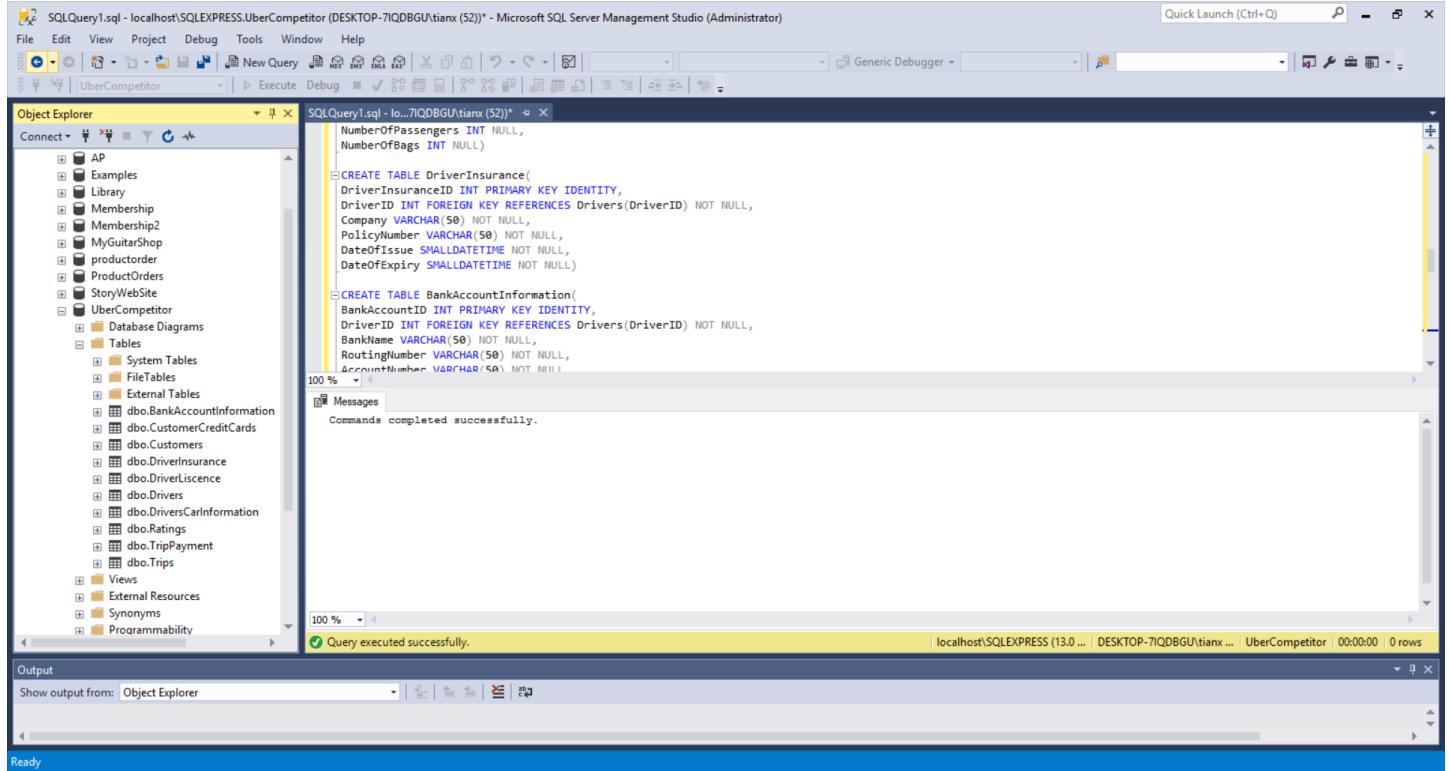
```
BankAccountID INT PRIMARY KEY IDENTITY,
DriverID INT FOREIGN KEY REFERENCES Drivers(DriverID) NOT NULL,
BankName VARCHAR(50) NOT NULL,
RoutingNumber VARCHAR(50) NOT NULL,
AccountNumber VARCHAR(50) NOT NULL,
TypeOfAccount VARCHAR(50) NOT NULL
)

CREATE TABLE TripPayment(
TripPaymentID INT PRIMARY KEY IDENTITY,
DriverID INT FOREIGN KEY REFERENCES Drivers(DriverID) NOT NULL,
Dates SMALLDATETIME NOT NULL,
Ammount MONEY NOT NULL,
BankAccountID INT FOREIGN KEY REFERENCES BankAccountInformation(BankAccountID) NOT NULL
)

CREATE TABLE Trips(
TripsID INT PRIMARY KEY IDENTITY,
CustomerID INT FOREIGN KEY REFERENCES Customers(CustomerID) NOT NULL,
DriverID INT FOREIGN KEY REFERENCES Drivers(DriverID) NOT NULL,
BookDate SMALLDATETIME NOT NULL,
PickUpTime SMALLDATETIME NULL,
DropOffTime SMALLDATETIME NULL,
Completed BIT NOT NULL,
PickUpAddress VARCHAR(50) NOT NULL,
PickUpCity VARCHAR(50) NOT NULL,
PickUPState VARCHAR(50) NOT NULL,
DropOffAddress VARCHAR(50) NOT NULL,
DropOffCity VARCHAR(50) NOT NULL,
DropOffState VARCHAR(50) NOT NULL,
CostPaid MONEY NULL,
Tip MONEY NULL,
CreditCardID INT FOREIGN KEY REFERENCES CustomerCreditCards(CreditCardID) NOT NULL,
CustomerNote Text NULL,
DriverNote Text NULL)

CREATE TABLE Ratings(
RatesID INT PRIMARY KEY IDENTITY,
TripsID INT FOREIGN KEY REFERENCES Trips(TripsID) NOT NULL,
RateDate SMALLDATETIME NULL,
Score INT NULL,
Comments TEXT NULL
)
"
```

Script result:



2.3 Code about data loading

Each table insert 5 records data which has real meaning

“**INSERT INTO** Customers

```
VALUES('Du', 'Kevin', 1, '508 Ivy Ridge Road', 'Syracuse', 'New York', 'kd123@gmail.com', '315-283-9999'),  
      ('Yi', 'Tianxiang', 1, '919 East Genesee Street', 'Syracuse', 'New York', 'tyi100@gmail.com', '315-  
278-3890'),  
      ('Qiu', 'Qinru', 1, '60 Presidential Road', 'Syracuse', 'New York', 'qinruqiu@gmail.com', '115-183-  
9919'),  
      ('Chen', 'Roger', 1, '113 Croyden Avenue', 'Syracuse', 'New York', 'Cr554@gmail.com', '232-121-6666'),  
      ('Fawcett', 'Jim', 1, '150 East Fayette Street', 'Syracuse', 'New York', 'jim@outlook.com', '666-663-  
1101')
```

```
INSERT INTO CustomerCreditCards
```

```
VALUES(1,8333145767418817),(1,9458507466425494),(2,1334883558174385),(3,6029043252798221),(4,21884645  
26151113),(5,1636454729655352)
```

INSERT INTO Drivers

```
VALUES('Brandon','Seed','111-22-1234','working-available','1986-05-05','615 James Street','Syracuse','New York'),  
      ('Jorge','Gkatz','848-82-6883','working-available','1976-10-03','417 Comstock Avenue','Syracuse','New York'),  
      ('Christine','Tina','308-68-5366','working-available','1982-02-21','404 University Avenue','Syracuse','New York'),  
      ('Jeff','Teague','672-87-5442','working-available','1967-07-16','908 Harrison Street','Syracuse','New York'),  
      ('Rodney','Tawfek','602-73-4381','working-available','1988-08-07','205AHawley Avenue','Syracuse','New York')
```

```
INSERT INTO DriversCarInformation  
VALUES(1,'Toyotal','Camry',1,'Blue',4,2),  
(2,'GMC','Terrian',1,'White',6,4),  
(3,'Volkswage','Passat',1,'Black',4,2),  
(4,'BMW','X5',1,'Black',4,2),  
(5,'Audi','Q9',1,'Black',4,4)
```

```
INSERT INTO DriverLiscence  
VALUES(1,'New York','A1270461','2015-08-01','2023-08-01'),  
(2,'New York','A2272878','2012-10-01','2020-10-01'),  
(3,'New York','C5456709','2019-05-01','2027-05-1'),  
(4,'New York','A1330424','2017-2-20','2025-2-20'),  
(5,'New York','A9626829','2011-11-11','2019-11-11')
```

```
INSERT INTO DriverInsurance  
VALUES(1,'Progressive','65405181','2018-05-05','2020-05-05'),  
(2,'QuoteWizard','26211066','2019-01-10','2022-01-10'),  
(3,'State Farm','73026700','2019-02-01','2021-02-01'),  
(4,'Esurance','15495412','2017-03-03','2020-03-03'),  
(5,'Liberty Mutual','78411751','2018-10-25','2019-10-25')
```

```
INSERT INTO BankAccountInformation  
VALUES(1,'JP Morgan Chase','322271627','229822663559','Checking'),  
(2,'Bank of America','102001017','145492286476','Checking'),  
(3,'Citigroup','021100361','856780521214','Saving'),  
(4,'Wells Fargo','267084131','843192652788','Saving'),  
(5,'Goldman Sachs','998100771','306596591043','Checking')
```

```
INSERT INTO TripPayment  
VALUES(1,'2019-05-01',90,1),(2,'2019-05-01',80,2),(3,'2019-05-01',85,3),(4,'2019-05-01',90,4),(5,'2019-05-01',95,5)
```

```
INSERT INTO Trips
```

```

VALUES(1,1,'2019-04-25','2019-04-25 18:06','2019-04-25 18:30',1,'439 Harrison Street','Syracuse','New York','Carousel Center','Syracuse','New York',10,2,1,NULL,NULL),
(2,4,'2019-04-25','2019-04-25 12:20','2019-04-25 13:00',1,'46 Harborside Drive','Syracuse','New York','994 East Genesee Street','Syracuse','New York',20,2,3,NULL,NULL),
(3,2,'2019-04-29','2019-04-29 15:00','2019-04-29 16:30',1,'500 University Place','Syracuse','New York','1200 Ivy Ridge Road','Syracuse','New York',30,4.5,4,NULL,NULL),
(4,5,'2019-04-23','2019-04-23 1:00','2019-04-23 1:10',1,'211 Clarke Street','Syracuse','New York','107 Monstrose Avenue','Syracuse','New York',15,5,5,NULL,NULL),
(5,3,'2019-04-22','2019-04-22 9:00','2019-04-22 9:30',1,'Mount Olympus Drive','Syracuse','New York','Destiny USA','Syracuse','New York',20,4,6,NULL,NULL)

```

INSERT INTO Ratings

```

VALUES(1,'2019-04-25 18:35',100,NULL),
(2,'2019-04-25 14:30',90,'Smells bad!'),
(3,'2019-04-29 18:30',100,NULL),
(4,'2019-04-25 00:00',50,'Too slow'),
(5,'2019-04-23 09:30',100,NULL)
"
```

Here is the result of Code query:

```

--USE UberCompetitor
--INSERT INTO Customers
VALUES('Du','Kevin',1,'88 Toy Ridge Road','Syracuse','New York','kdu12@gmail.com','315-283-9999'),
('Glu','Glorv',2,'999 East Genesee Street','Syracuse','New York','glu12@gmail.com','315-278-3999'),
('Qui','Quirv',1,'68 Presidential Road','Syracuse','New York','qiu12@gmail.com','315-183-9919'),
('Chen','Roger',1,'113 Croyden Avenue','Syracuse','New York','cr54@gmail.com','232-121-6666'),
('Fawcett','Jill',1,'158 East Fayette Street','Syracuse','New York','jin12outlook.com','666-663-1081')

--INSERT INTO CustomerCreditCards
VALUES(1,833314576418817),(1,9458507466425494),(2,1334883558174385),(3,6029043252798221),(4,2184464526151113),(5,1636454729655352)

--INSERT INTO Drivers
VALUES('Brandon','Seed',111-22-1234,'working-available','1986-05-05','615 James Street','Syracuse','New York'),
('Jorge','Gkatz',848-82-6883,'working-available','1976-10-03','417 Comstock Avenue','Syracuse','New York'),
('Christine','Tina',380-68-5366,'working-available','1982-02-21','484 University Avenue','Syracuse','New York'),
('Jeffrey','Kane',477-47-4247,'working-available','1987-07-11','448 Harrison Street','Syracuse','New York')

```

Query executed successfully.

Here is 3 important table demonstration:Trips,Customers,Drivers

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including tables like AP, Examples, Library, Membership, Membership2, MyGuitarShop, producerder, ProductOrders, StoryWebSite, and UberCompetitor, along with their respective system and external tables.

In the center, the Results pane displays the output of the following query:

```
SELECT * FROM Trips
```

The results show 5 rows of data from the Trips table:

TripID	CustomerID	DriverID	BookDate	PickUpTime	DropOffTime	Completed	PickUpAddress	PickUpCity	PickUpState	DropOffAddress	DropOffCity	DropOffState	CostP
1	1	1	2015-04-20 00:00:00	2015-04-20 12:00:00	2015-04-20 13:00:00	1	40 Harrison Street	Syracuse	New York	Counsel Center	Syracuse	New York	10.00
2	2	2	2015-04-20 00:00:00	2015-04-25 12:00:00	2015-04-25 13:00:00	1	40 Harrison Drive	Syracuse	New York	994 East Genesee Street	Syracuse	New York	20.00
3	3	3	2015-04-29 00:00:00	2015-04-29 15:00:00	2015-04-29 16:30:00	1	500 University Place	Syracuse	New York	1200 Ivy Ridge Road	Syracuse	New York	30.00
4	4	4	2015-04-23 00:00:00	2015-04-23 01:00:00	2015-04-23 01:10:00	1	211 Clarkie Street	Syracuse	New York	107 Monroe Avenue	Syracuse	New York	15.00
5	5	5	2015-04-22 00:00:00	2015-04-22 09:00:00	2015-04-22 09:30:00	1	Mount Olympus Drive	Syracuse	New York	Destry USA	Syracuse	New York	20.00

Below the results, a message indicates "Query executed successfully." and "localhost\SQLExpress (13.0... DESKTOP-7QD8GU\tianxiangyi... UberCompetitor 00:00:00 5 rows".

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including tables like AP, Examples, Library, Membership, Membership2, MyGuitarShop, producerder, ProductOrders, StoryWebSite, and UberCompetitor, along with their respective system and external tables.

In the center, the Results pane displays the output of the following query:

```
SELECT * FROM Customers
```

The results show 5 rows of data from the Customers table:

CustomerID	CustomerLastName	CustomerFirstName	Active	HomeAddress	City	State	Email	PhoneNumber
1	Du	Kevin	1	508 Ivy Ridge Road	Syracuse	New York	kd123@gmail.com	315-283-9999
2	Yi	Tianxiang	1	919 East Genesee Street	Syracuse	New York	ty100@gmail.com	315-278-3990
3	Qiu	Guru	1	60 Presidential Road	Syracuse	New York	qmu99@gmail.com	115-183-9919
4	Chen	Roger	1	113 Croyden Avenue	Syracuse	New York	C554@gmail.com	232-121-6666
5	Fawcett	Jm	1	150 East Fayette Street	Syracuse	New York	jn@outlook.com	666-663-1101

Below the results, a message indicates "Query executed successfully." and "localhost\SQLExpress (13.0... DESKTOP-7QD8GU\tianxiangyi... UberCompetitor 00:00:00 5 rows".

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including tables like AP, Examples, Library, Membership, Membership2, MyGuitarShop, producerder, ProductOrders, StoryWebSite, and UberCompetitor, along with their respective system and external tables.

In the center, the Results pane displays the output of the following query:

```
SELECT * FROM Drivers
```

The results show 5 rows of data from the Drivers table:

DriverID	DriverLastName	DriverFirstName	SocialSecurityNumber	Status	DateOfBirth	HomeAddress	City	State
1	Brandon	Seed	911-22-0254	working-available	1985-01-20 00:00:00	215 James Street	Syracuse	New York
2	Jeff	Lee	848-65-8883	working-available	1976-03-20 00:00:00	417 Clinton Avenue	Syracuse	New York
3	Catherine	Tina	300-65-5366	working-available	1982-03-21 00:00:00	404 University Avenue	Syracuse	New York
4	Jeff	Teague	6721-67-5442	working-available	1967-07-16 00:00:00	908 Harrison Street	Syracuse	New York
5	Rodney	Taufek	602-73-4381	working-available	1988-05-07 00:00:00	2054 Hawley Avenue	Syracuse	New York

Below the results, a message indicates "Query executed successfully." and "localhost\SQLExpress (13.0... DESKTOP-7QD8GU\tianxiangyi... UberCompetitor 00:00:00 5 rows".

3. Testing

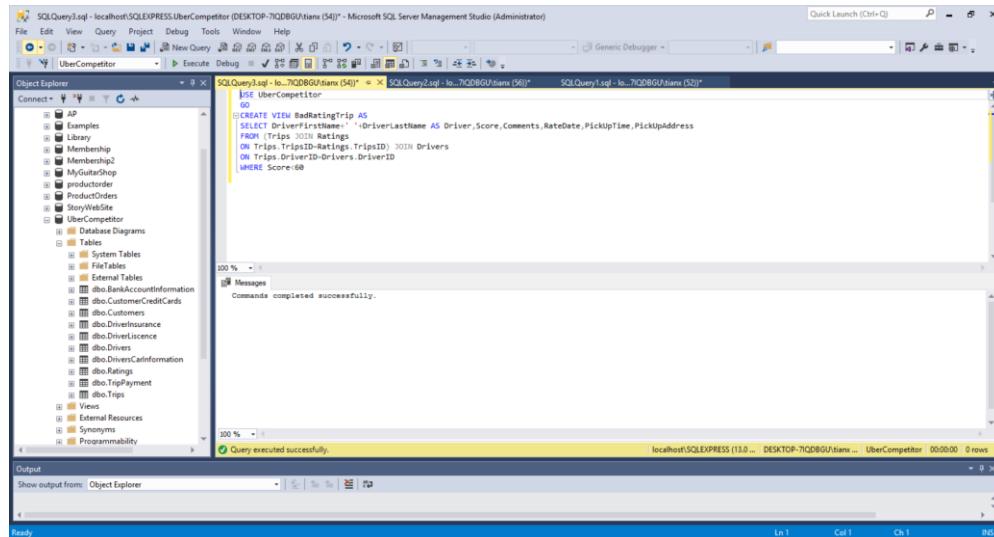
3.1 Views

(1) List of bad comments given by customer, I design not to show Customer Name which is supporting anonymous comments. This view helps customer to feedback to company and other customer to choose better driver.

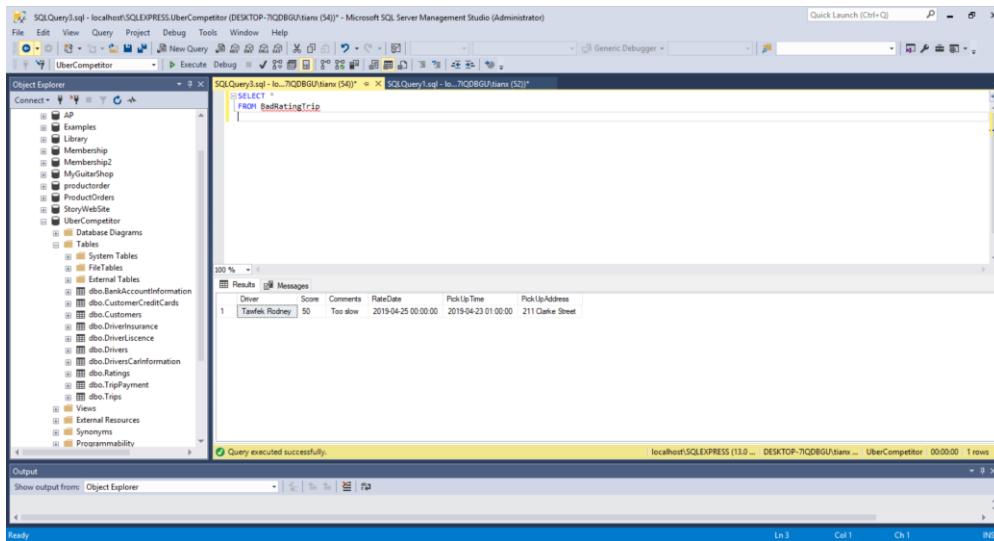
Code:

```
"USE UberCompetitor
GO
CREATE VIEW BadRatingTrip AS
SELECT DriverFirstName+' '+DriverLastName AS Driver,Score,Comments,RateDate,PickUpTime,PickUpAddress
FROM (Trips JOIN Ratings
ON Trips.TripsID=Ratings.TripsID) JOIN Drivers
ON Trips.DriverID=Drivers.DriverID
WHERE Score<60
GO
SELECT *
FROM BadRatingTrip"
```

Execution Screenshot:



Result Verification



(2) This View help customers to choose the driver that satisfy their requirement(Seats and Bags) and desriminate the car when booked throgh color and make. What's more it could help company to manage driver information more easily.

Code:

```
"USE UberCompetitor
GO

CREATE VIEW DriverDetailedsInformation AS
SELECT DriverFirstName+ ' ' + DriverLastName AS Driver, SocialSecurityNumber, DriverLiscenceNumber AS
DriverLiscence ,
Company + ':' + PolicyNumber AS DriverInsurance, Make + ' ' + model + 'Color:' + Color AS
CarInformation, NumberOfPassengers, numberofBags
FROM((Drivers JOIN DriversCarInformationON
Drivers.DriverID=DriversCarInformation.DriverID)JOIN DriverLiscence
ON Drivers.DriverID=DriverLiscence.DriverID) JOIN DriverInsurance
ON Drivers.DriverID=DriverInsurance.DriverID
USE UberCompetitor
GO

SELECT *
FROM DriverDetailedsInformation
"
```

ExecutionScreenshot:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, the database 'UberCompetitor' is selected. In the center, two queries are running in separate panes:

```

USE UberCompetitor
GO

CREATE VIEW DriverDetailedInformation AS
SELECT DriverFirstName + ' ' + DriverLastName AS Driver, SocialSecurityNumber, DriverLiscenceNumber AS DriverLiscence,
Company + ':' + PolicyNumber AS DriverInsurance, Make + ' ' + model + 'Color:' + Color AS CarInformation, NumberOfPassengers, numberOfBags
FROM(Drivers JOIN DriversCarInformation ON
Drivers.DriverID=DriversCarInformation.DriverID)JOIN DriverLiscence
ON Drivers.DriverID=DriverLiscence.DriverID JOIN DriverInsurance
ON Drivers.DriverID=DriverInsurance.DriverID

```

The status bar at the bottom indicates "Query executed successfully." and "localhost\SQLEXPRESS (13.0 ... | DESKTOP-7IQDBGU\tianx ... | UberCompetitor | 00:00:00 | 0 rows".

Result Verification:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, the database 'UberCompetitor' is selected. In the center, a query is running in a single-pane interface:

```

USE UberCompetitor
GO

SELECT *
FROM DriverDetailedInformation

```

The results pane displays the following data:

Driver	SocialSecurityNumber	DriverLiscence	DriverInsurance	CarInformation	NumberOfPassengers	numberOfBags
Seed Brandon	111-22-1234	A1270461	Progressive:65405181	Toyota Camry:Color:Blue	4	2
Gkatz Jorge	848-82-6883	A2272878	QuoteWizard:26211066	GMC Terrain:Color:White	6	4
Tina Christine	308-69-5366	C5456709	State Farm:73026700	Volkswage Passat:Color:Black	4	2
Teague Jeff	672-87-5442	A1330424	Esurance:15495412	BMW X5:Color:Black	4	2
Tawfek Rodney	602-73-4381	A9626829	Liberty Mutual:78411751	Audi Q9:Color:Black	4	4

The status bar at the bottom indicates "Query executed successfully." and "localhost\SQLEXPRESS (13.0 ... | DESKTOP-7IQDBGU\tianx ... | UberCompetitor | 00:00:00 | 5 rows".

(3) This View union the customer card information and driver account information. Help company to set its security level higher and carefully for the reason of its importance.

Code:

```
"USE UberCompetitor
```

```
GO
```

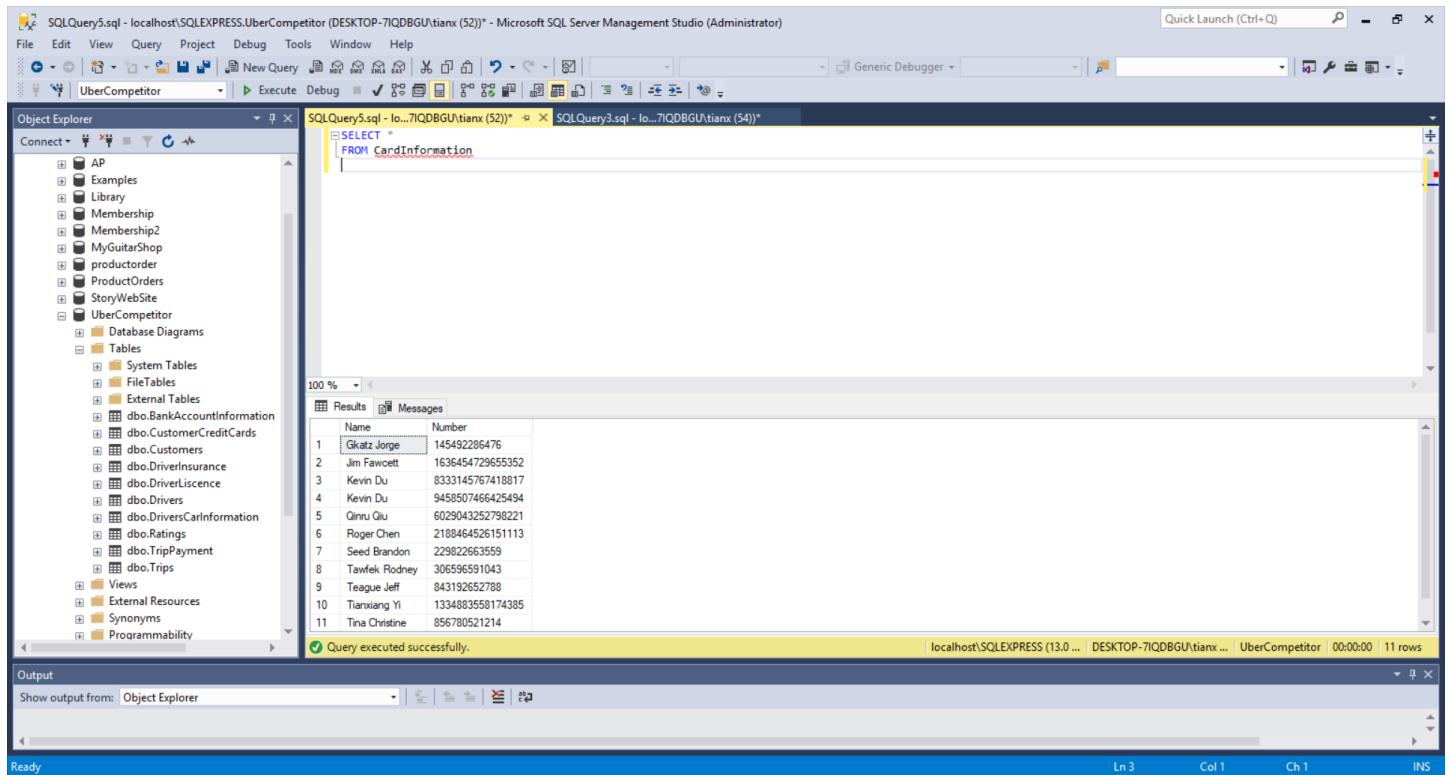
```

CREATE VIEW CardInformation AS
SELECT DriverFirstName+ ' ' + DriverLastName AS Name, AccountNumber AS Number
FROM Drivers JOIN BankAccountInformation ON
Drivers.DriverID=BankAccountInformation.DriverID
UNION
SELECT CustomerFirstName+ ' '+CustomerLastName AS Name, CreditCardNumber AS Number
FROM Customers JOIN CustomerCreditCards
ON Customers.CustomerID=CustomerCreditCards.CustomerID
GO
SELECT *
FROM CardInformation
"
```

Execution Screenshot:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'UberCompetitor' is selected, and the 'Tables' node under it is expanded, showing various tables like AP, Examples, Library, Membership, etc. In the center pane, a query window titled 'SQLQuery1.sql - localhost\SQLExpress.UberCompetitor (DESKTOP-7IQDBGU\tianx (52))' contains the SQL code provided in the text block. The code creates a view named 'CardInformation' by selecting names and account numbers from drivers and customers. The 'Messages' pane at the bottom shows the message 'Commands completed successfully.' and the status bar indicates 'Query executed successfully.' and 'localhost\SQLExpress (13.0 ... | DESKTOP-7IQDBGU\tianx ... | UberCompetitor | 00:00:00 | 0 rows'.

Result Verification:



(4) This View is made for customer who want to check his past record of travel and for company to manage all the past record trips.

Code:

```
"USE UberCompetitor
GO
CREATE VIEW HistoryTrips AS
SELECT CustomerFirstName+ ' '+CustomerLastName AS Customer, PickupTime,Dropofftime
,PickUpAddress+', '+PickUpCity+', '+PickUPState+' -- '+DropOffAddress+', '+DropOffCity+', '+DropOffState
AS Expeirence,Costpaid+ Tip AS TotalMoney
, DriverFirstName+ ' '+ DriverLastName AS Driver,Completed
FROM (Trips JOIN Drivers
ON Trips.DriverID=Drivers.DriverID)JOIN Customers
ON Trips.CustomerID=Customers.CustomerID
USE UberCompetitor
GO
SELECT *
FROM HistoryTrips
"
```

Execution Screenshot:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'UberCompetitor' is selected. In the center pane, a T-SQL script is being run:

```

USE UberCompetitor
GO
CREATE VIEW HistoryTrips AS
SELECT Customer.FirstName + Customer.LastName AS Customer, PickupTime,Dropofftime
,PickUpAddress+','+PickUpCity+','+PickUpState+','+DropOffAddress+','+DropOffCity+','+DropOffState AS Experience,CostPaid+ Tip AS TotalMoney
, Driver.FirstName + Driver.LastName AS Driver,Completed
FROM Trips JOIN Drivers
ON Trips.DriverID=Drivers.DriverID JOIN Customers
ON Trips.CustomerID=Customers.CustomerID
    
```

The 'Messages' tab shows the message: "Commands completed successfully." The status bar at the bottom right indicates "Query executed successfully." and "localhost:SQLEXPRESS (13.0 ... DESKTOP-7IQDBGU\tianx ... UberCompetitor | 00:00:00 | 0 rows".

Result Verification:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'UberCompetitor' is selected. In the center pane, a T-SQL query is being run:

```

USE UberCompetitor
GO
SELECT *
FROM HistoryTrips
    
```

The results are displayed in the 'Results' tab, showing five rows of trip history data:

	Customer	Pickup Time	Dropofftime	Experience	TotalMoney	Driver	Completed
1	Kevin Du	2019-04-25 18:06:00	2019-04-25 18:30:00	439 Harrison Street,Syracuse,New York - Carous...	12.00	Seed Brandon	1
2	Tianxiang Yi	2019-04-25 12:20:00	2019-04-25 13:00:00	46 Harbordale Drive,Syracuse,New York - 994 Ea...	22.00	Teague Jeff	1
3	Qiruo Qu	2019-04-29 15:00:00	2019-04-29 16:30:00	500 University Place,Syracuse,New York - 1200 L...	34.50	Gkatz Jorge	1
4	Roger Chen	2019-04-23 01:00:00	2019-04-23 01:10:00	211 Clake Street,Syracuse,New York - 107 Mon...	20.00	Tawek Rodney	1
5	Jim Fawcett	2019-04-22 09:00:00	2019-04-22 09:30:00	Mount Olympus Drive,Syracuse,New York - Deat...	24.00	Tina Christine	1

The status bar at the bottom right indicates "Query executed successfully." and "localhost:SQLEXPRESS (13.0 ... DESKTOP-7IQDBGU\tianx ... UberCompetitor | 00:00:00 | 5 rows".

3.2 Functions

(1)One month income of company

Code:

"**USE** UberCompetitor

GO

CREATE FUNCTION fnIncomeOneMonth

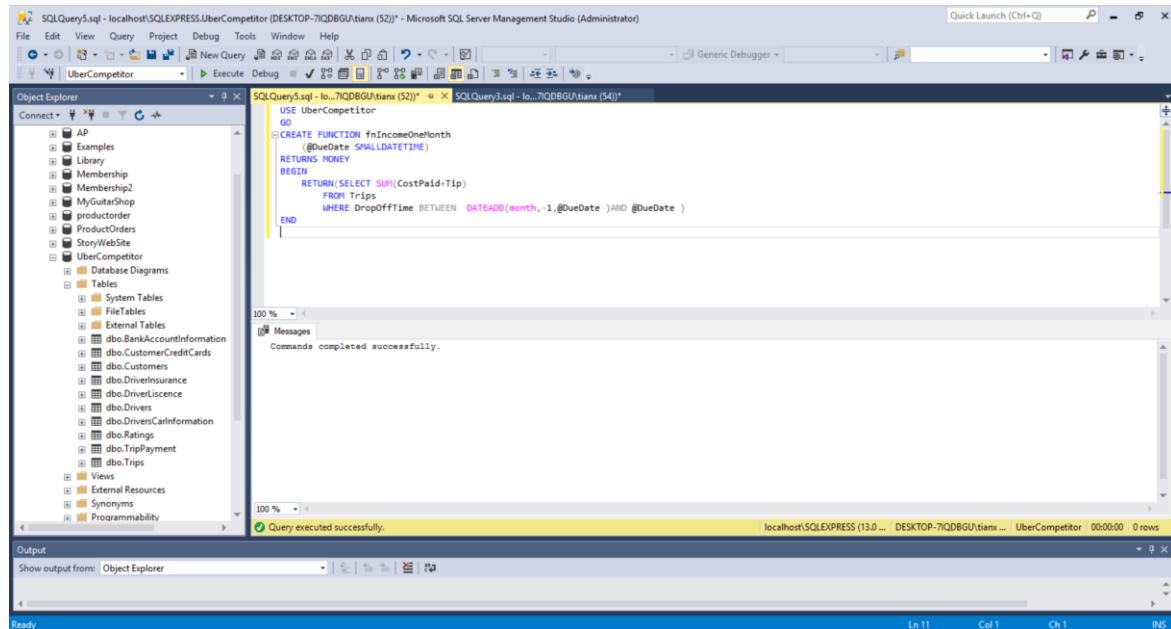
(@DueDate SMALLDATETIME)

```

RETURNS MONEY
BEGIN
    RETURN(SELECT SUM(CostPaid+Tip)
        FROM Trips
        WHERE DropOffTime BETWEEN DATEADD(month,-1,@DueDate )AND @DueDate )
END"

```

Execution Screenshot:



(2)Expendure (give salary to driver)

Code:

```

"USE UberCompetitor
GO
CREATE FUNCTION fnExpendureOneMonth
    (@DueDate SMALLDATETIME)
RETURNS MONEY
BEGIN
    RETURN(SELECT SUM(Ammount)
        FROM TripPayment
        WHERE Dates BETWEEN DATEADD(month,-1,@DueDate )AND @DueDate )
END
"

```

Execution Screenshot:

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure for the 'UberCompetitor' database, including tables like AP, Examples, Library, Membership, and UberCompetitor. The central pane displays a T-SQL script for creating a function:

```

USE UberCompetitor
GO
CREATE FUNCTION fnExpenditureOneMonth
    (@DueDate SMALLDATETIME)
RETURNS MONEY
BEGIN
    RETURN (SELECT SUM(Amount)
            FROM TripPayment
            WHERE Dates BETWEEN DATEADD(month,-1,@DueDate )AND @DueDate )
END

```

The 'Messages' pane below the script shows the message "Commands completed successfully." The status bar at the bottom indicates "Query executed successfully." and "localhost\SQLEXPRESS (13.0... | DESKTOP-7IQDBGU\tianx... | UberCompetitor | 00:00:00 | 0 rows".

Result Verification:

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure for the 'UberCompetitor' database. The central pane displays a T-SQL script that executes a stored procedure to calculate income, expenditure, and balance due for May 2019:

```

USE UberCompetitor
GO
DECLARE @DueDate SMALLDATETIME
SET @DueDate='2019-05-01'
PRINT 'Income: $'+CONVERT(varchar,dbo.fnIncomeOneMonth(@DueDate) )
PRINT 'Expenditure: $'+CONVERT(varchar,dbo.fnExpendureOneMonth(@DueDate) )
PRINT 'BalanceDue: $'+CONVERT(varchar,dbo.fnIncomeOneMonth(@DueDate)-dbo.fnExpendureOneMonth(@DueDate) )

```

The 'Messages' pane below the script shows the results: "Income: \$112.50", "Expenditure: \$440.00", and "BalanceDue: \$-327.50". The status bar at the bottom indicates "Query executed successfully." and "localhost\SQLEXPRESS (13.0... | DESKTOP-7IQDBGU\tianx... | UberCompetitor | 00:00:00 | 0 rows".

Explanation:

In my design of database, driver get their salary every first day of new month. So this function could help company to evaluate what's the total balance due of one month which is income from all tirps minus all drvier amount of the salary.

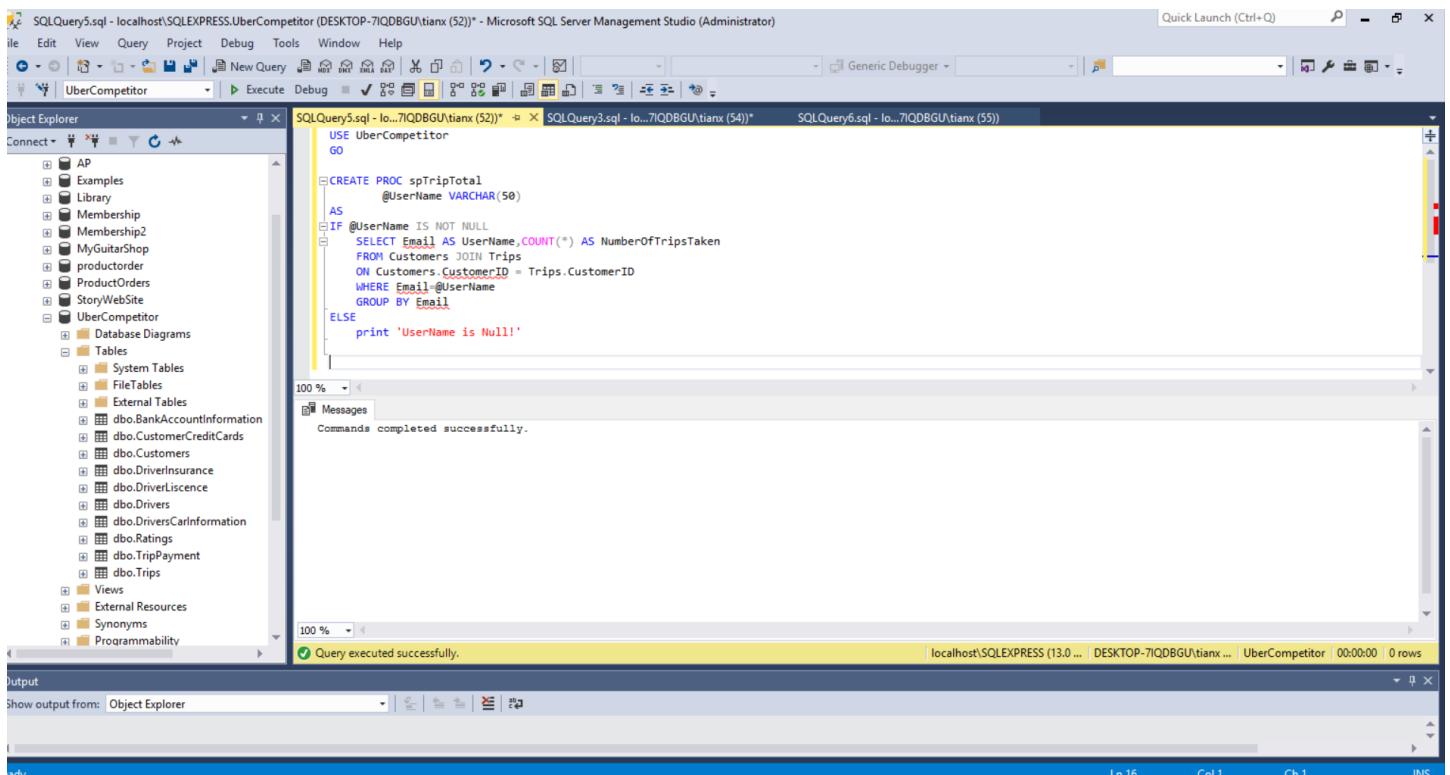
3.3 Stored procedure

(1) This is the procedure to specified customer to find his total number trips more quick and efficient.

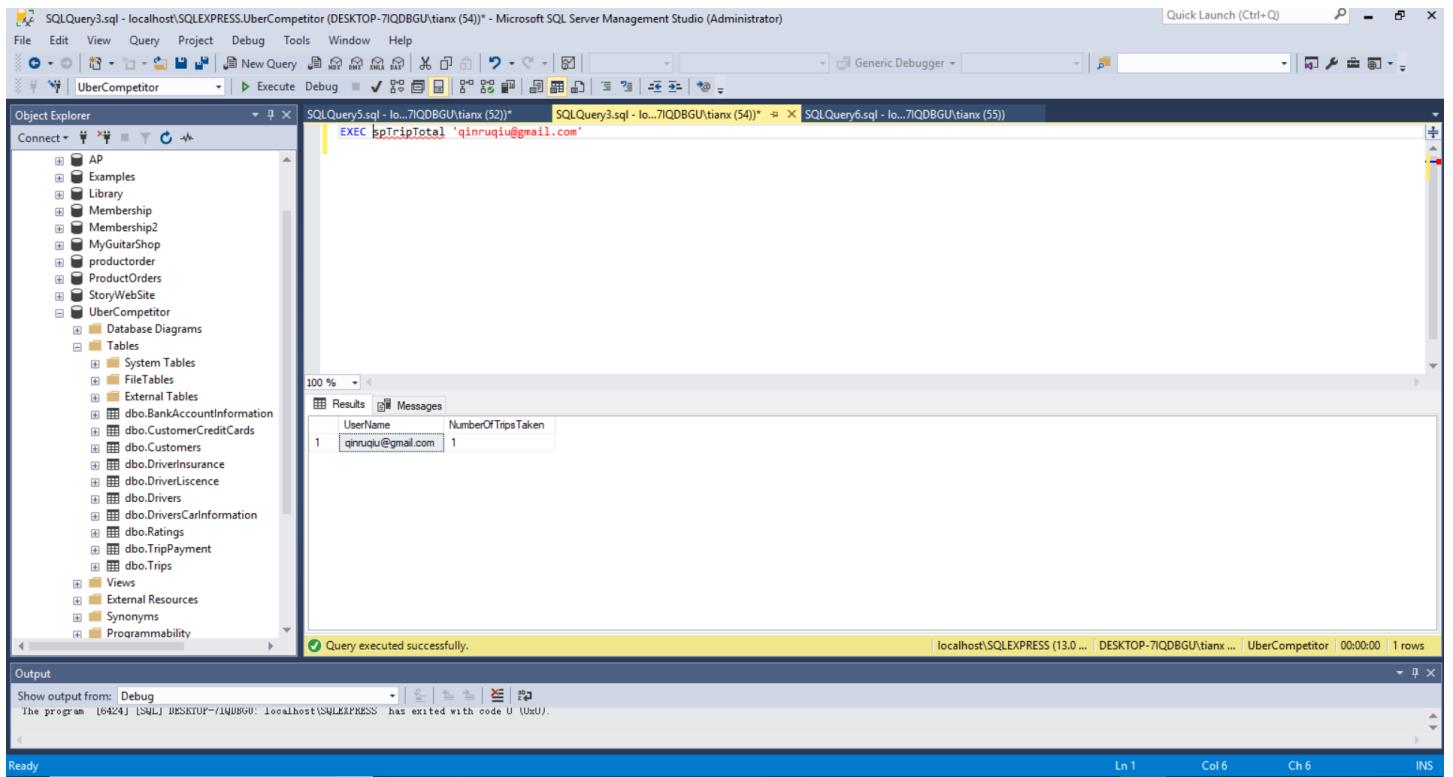
Code:

```
"USE UberCompetitor
GO
CREATE PROC spTripTotal
    @UserName VARCHAR(50)
AS
IF @UserName IS NOT NULL
    SELECT Email AS UserName,COUNT(*) AS NumberOfTripsTaken
    FROM Customers JOIN Trips
    ON Customers.CustomerID = Trips.CustomerID
    WHERE Email=@UserName
    GROUP BY Email
ELSE
    print 'UserName is Null!'
GO
EXEC spTripTotal 'qinruqiu@gmail.com'
"
```

Execution Screenshot:



Result Verification:



(2) This is procedure for driver to record his past one month trip. If this variable @Date is date driver got paid, this could be the record for all pickup of the pay.

Code:

```
"USE UberCompetitor
GO
```

```
CREATE PROC spTripsPastOneMonth
    @Driver INT,
    @Date SMALLDATETIME
AS
SELECT DriverFirstName + ' ' + DriverLastName AS DriverName, PickUpAddress, PickUpTime
FROM Drivers JOIN Trips
ON Drivers.DriverID=Trips.DriverID
WHERE BookDate BETWEEN DATEADD(month, -1, @Date) and @Date
    AND Drivers.DriverID=@Driver
GO
EXEC spTripsPastOneMonth 4, '2019-05-01'
"
```

Execution Screenshot:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'UberCompetitor' is selected. In the center pane, a query window displays the following T-SQL code:

```

FROM Drivers JOIN Trips
ON Drivers.DriverID=Trips.DriverID
WHERE BookDate BETWEEN DATEADD(month,-1,@Date) and @Date
AND Drivers.DriverID=@Driver
  
```

The status bar at the bottom indicates "Query executed successfully." and "localhost\SQLExpress (13.0 ... | DESKTOP-7IQDBGU\tianx ... | UberCompetitor | 00:00:00 | 0 rows".

Result Verification:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'UberCompetitor' is selected. In the center pane, a query window displays the following T-SQL code:

```

EXEC spTripsPastOneMonth 4, '2019-05-01'
  
```

The results pane shows a single row of data:

	DriverName	PickUpAddress	PickUpTime
1	Teague Jeff	46 Harborside Drive	2019-04-25 12:20:00

The status bar at the bottom indicates "Query executed successfully." and "localhost\SQLExpress (13.0 ... | DESKTOP-7IQDBGU\tianx ... | UberCompetitor | 00:00:00 | 1 rows".

3.4 Scripts

(1) Password assignment

Code:

```
'CREATE LOGIN TianxiangYi WITH PASSWORD='123456' ,
```

```
DEFAULT_DATABASE = UberCompetitor'
```

Execution Screenshot:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'master' database is selected. In the center pane, a query window contains the command to create a login:

```
'CREATE LOGIN TianxiangYi WITH PASSWORD='123456' ,
DEFAULT_DATABASE = UberCompetitor'
```

The 'Messages' pane at the bottom displays the message "Commands completed successfully." The 'Output' pane shows the executed query and its results.

(2)User roles

Code:

```
CREATE USER Jim FOR LOGIN TianxiangYi
```

Execution Screenshot:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'master' database is selected. In the center pane, a query window contains the command to create a user:

```
'CREATE USER Jim FOR LOGIN TianxiangYi'
```

The 'Messages' pane at the bottom displays the message "Commands completed successfully." The 'Output' pane shows the executed query and its results.

4. Conclusion

This is all the Uber Competitor Database I designed. It include everything provided in Project 2 Statement file. It could access data from client side, driver side. The test situation looks well without error. The backward may be the data loading volum is too small. For futher modify and implementation should wait for it to use.

Remarks

The requirement of this Uber competitor database is all satisfied. It work well with test data.