University of Texas at Arlington

CSE 3330 Database Project 2 Part 2

By

Shishir Pathak

Ujjwal Bajagain

HONOR CODE

I pledge, on my honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values

hard work and honest effort in the pursuit of academic excellence.

I promise that I will submit only work that I personally create or that I contribute to group collaborations,

and I will appropriately reference any work from other sources. I will follow the highest standards of

integrity and uphold the spirit of the Honor Code.

This document contains Screen shots of some of the codes used in the CSE 3330 Database project.

To review the entire source code please visit https://github.com/ujjwalbgn/cse3330

**Tools Used:**

Database: SQlite3

Programming Language: python

Database tool: DB Browser for SQLite

Version Control: Git (GitHub link: [**https://github.com/ujjwalbgn/cse3330**](https://github.com/ujjwalbgn/cse3330))

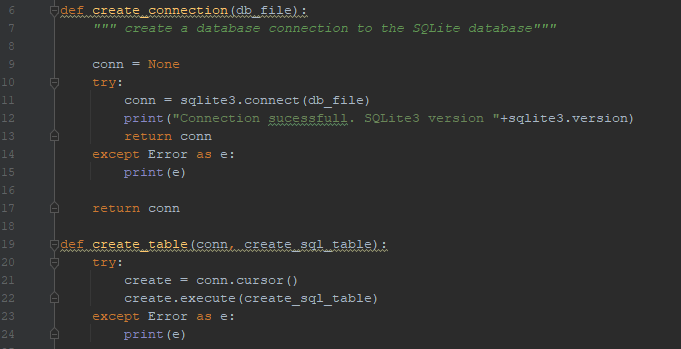
To recreate the program please follow the README.md associated with this program.

The data was copied to the python script where those data were inserted into the inserted functions which inserted the data into the tables.

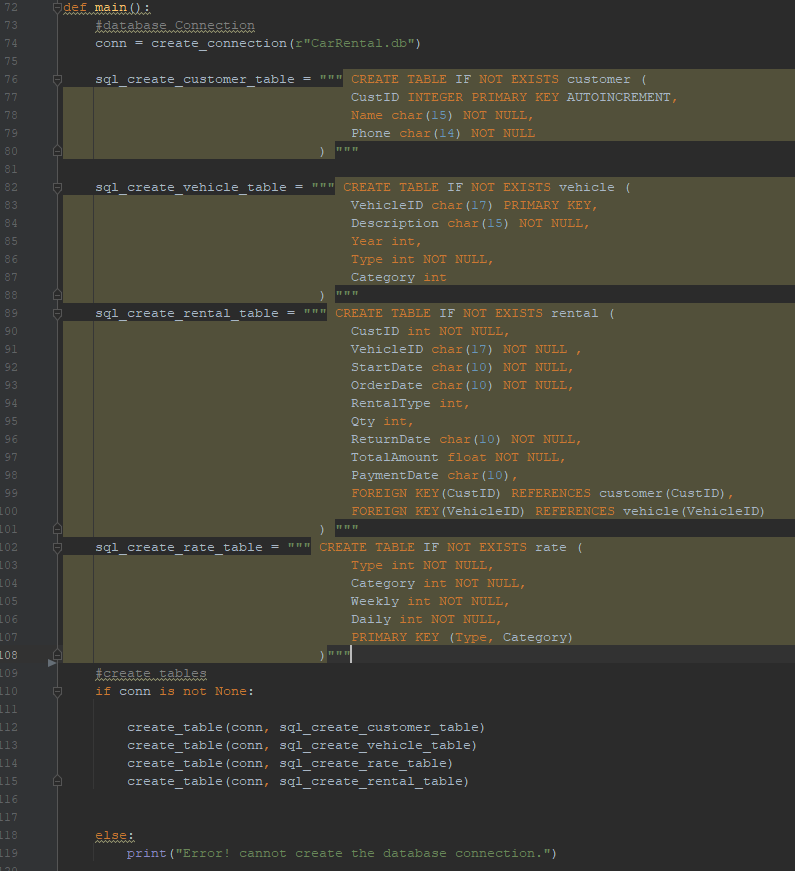
The queries were ran in DB Browser for SQLite and the screenshot of the query with the respected results are listed in this document.

**TASK 1: Creating database tables**

The database and table are create by running python script ***create\_insert.py*** . Here is the screenshot of the **python script used to create database and function used create tables**.



**Query for creating table and passing the query into the create table function:**



**Explanation:** While creating customer table, we add Auto Increment to primary key so that the database could automatically generate primary key when its not being passes into the insert query.

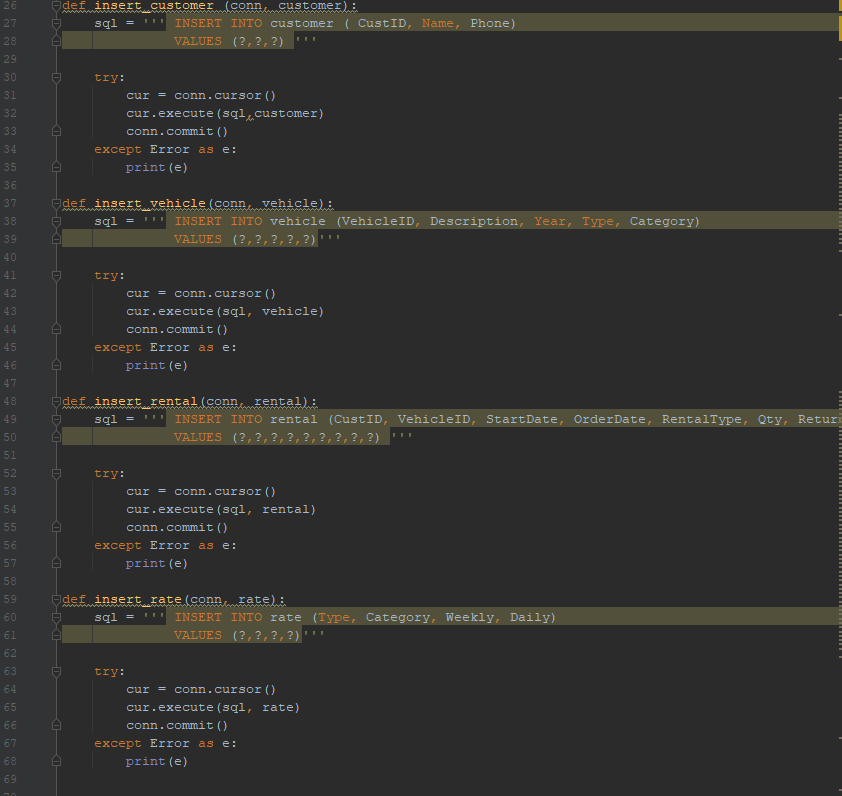
For most fields Char data type is used as most data were short string values. And int and float were used for number values according to the need of the data.

In rate table Composite Key (multiple columns are defined as primary key) of Type and Category is used. Using composite key for rate makes it easier to look up the rental cost of the vehicle by using type and category of the vehicle.

Not Null is used for the fields that are required for the business logic to work.

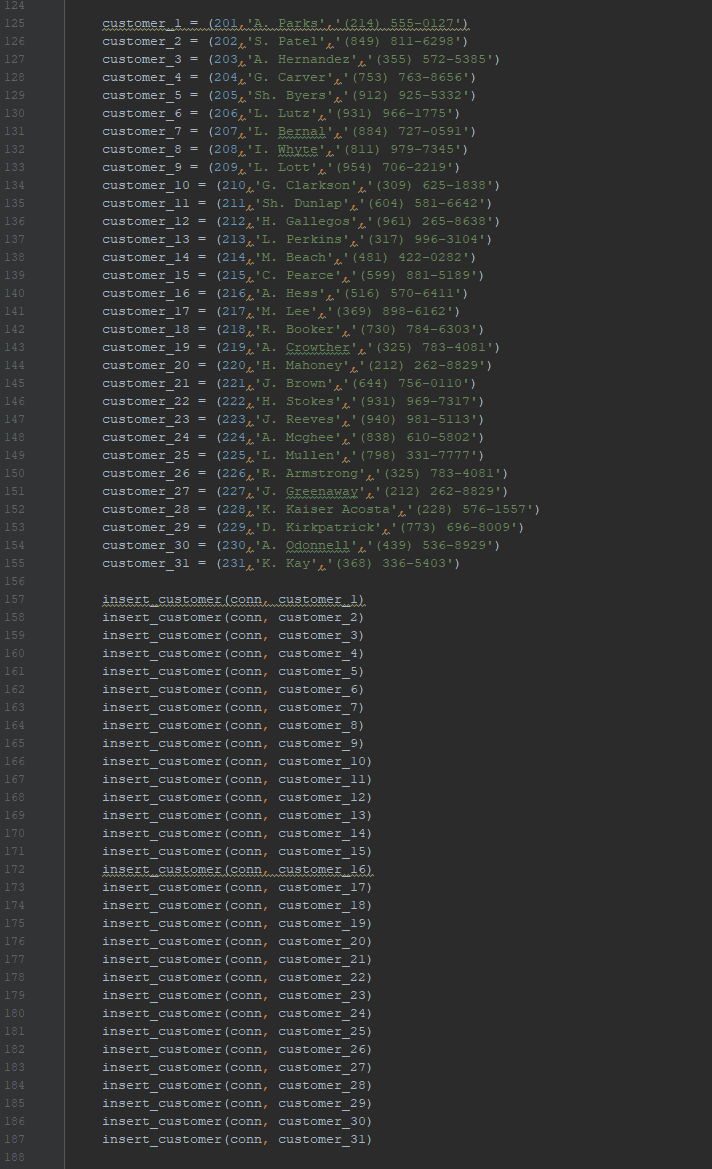
**Task 2: Inserting Data into Tables:**

Below is the screenshot of function which are used to pass data into the table.



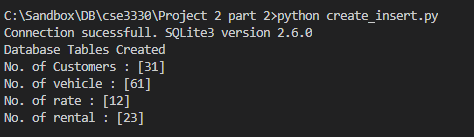
**Explanation**: To inset data into tables we wrote python function with sql insert queries within them and called in these function and passed in the data from the text files to the respective function.

Examples of passing data into insert function:

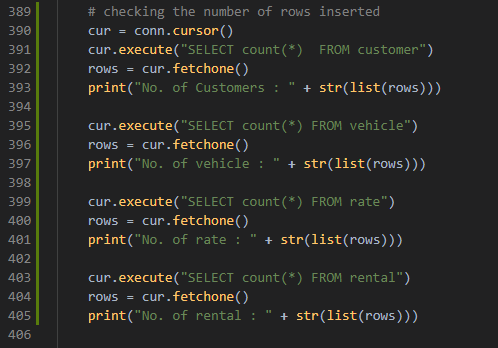


**Explanation:** This is an example of inserting data into customer table where we started by adding data from the text file as a python variable and passing the variable to the insert function to create a table. Similar approach is used to insert data into other tables as well.

Now we run the ***create\_insert.py*** python script which will create the database, create tables and insert data and **print out the number of rows in the table**.



To get the number of rows following query was used:

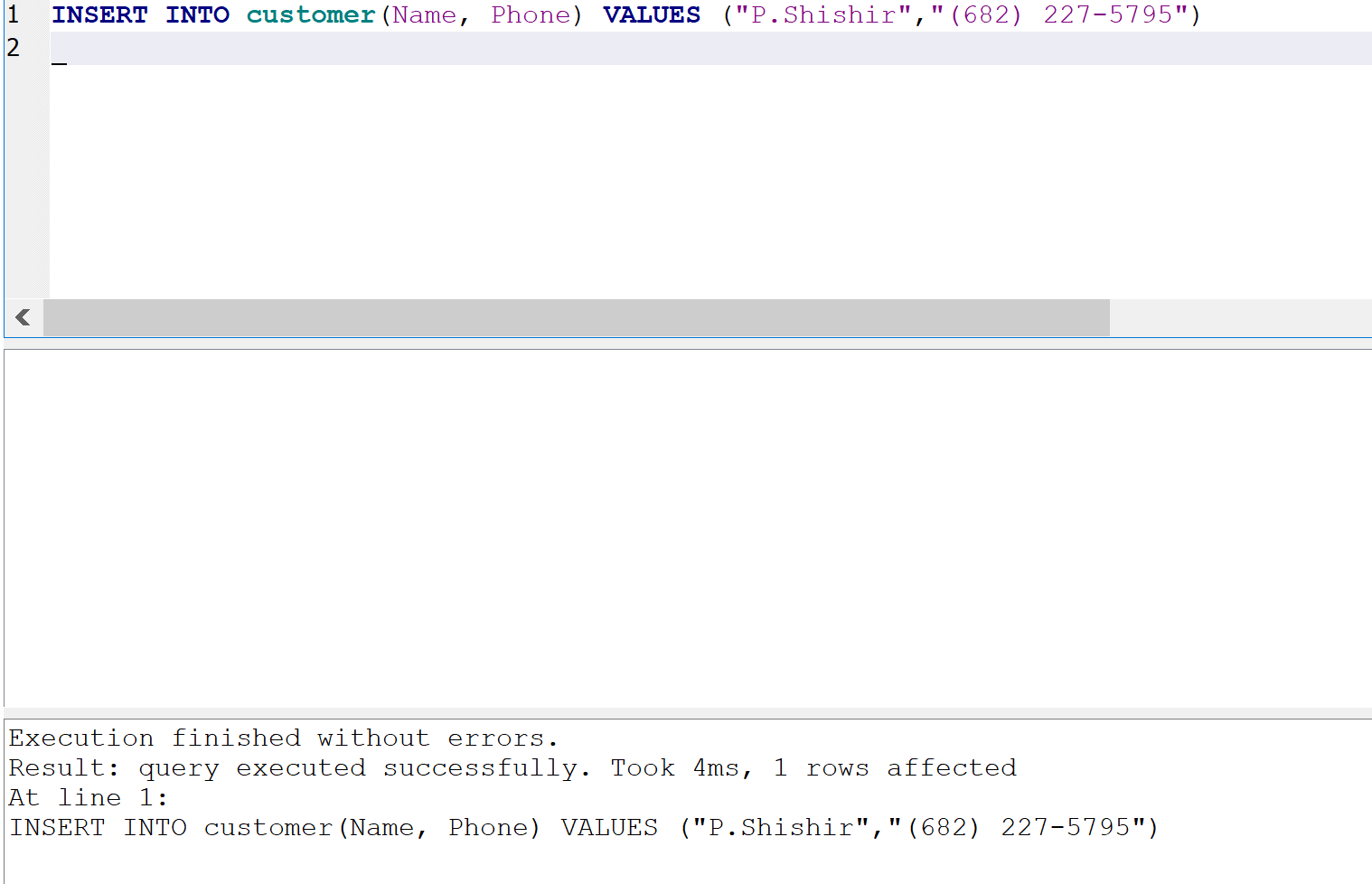


**Task 3: Queries**

**Question 1: Insert yourself as a New Customer. Do not provide the CustomerID in your query.**

**Query:**

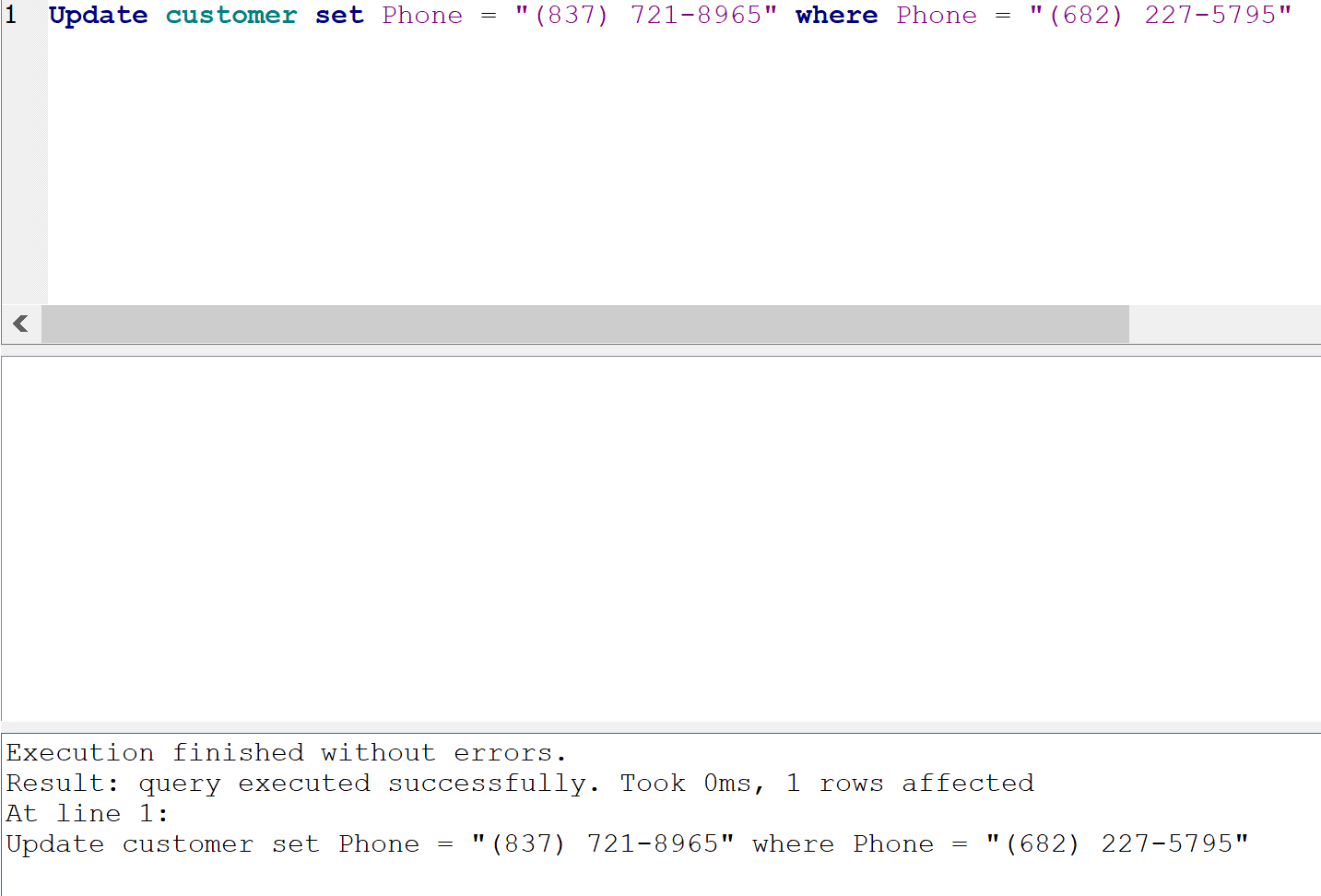
INSERT INTO customer(Name, Phone) VALUES ("P.Shishir","(682) 227-5795")



**Question 2: Update your phone number to (837) 721-8965**

**Query:**

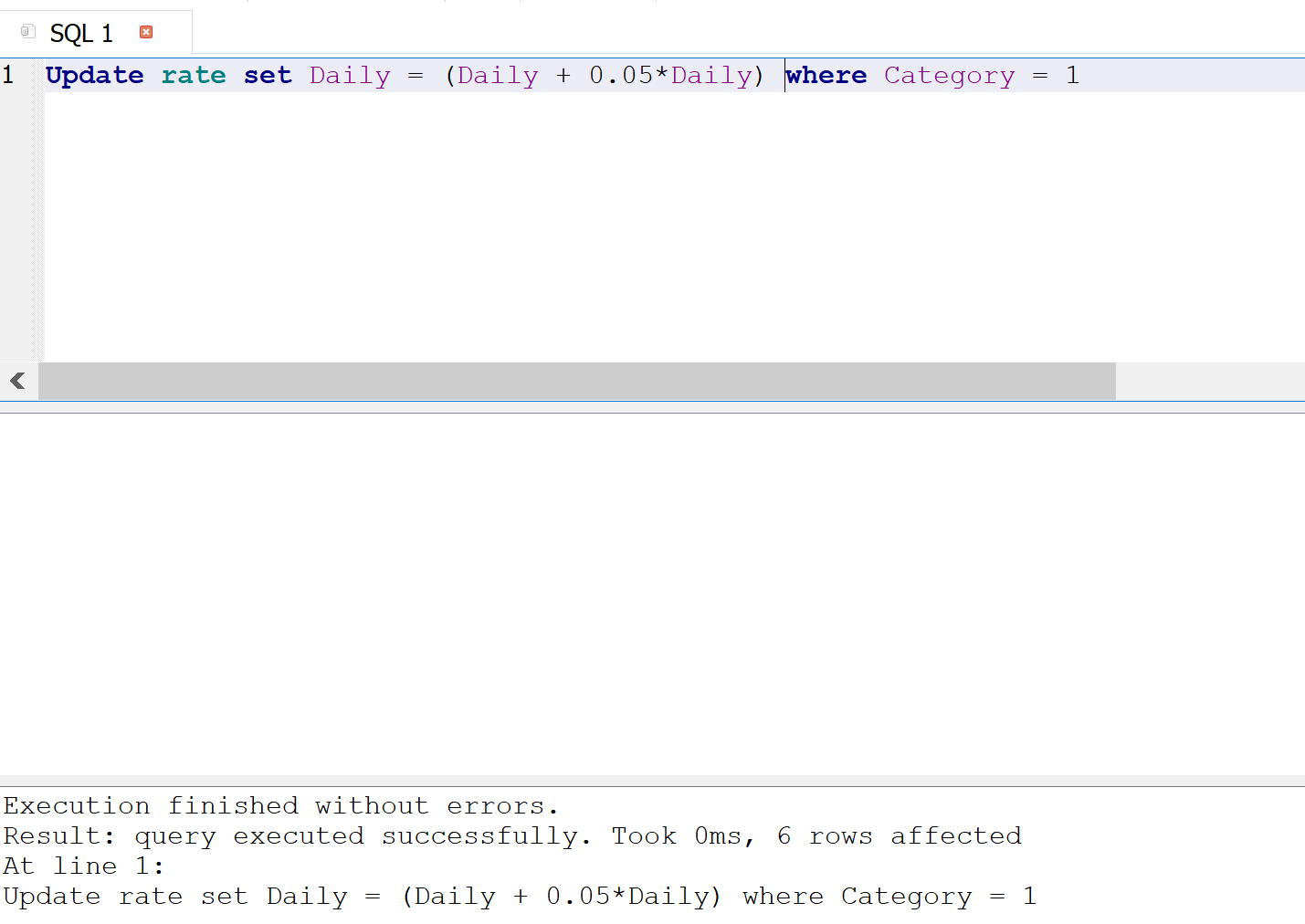
Update customer set Phone = "(837) 721-8965" where Phone = "(682) 227-5795"



**Question 3: Increase only daily rates for luxury vehicles by 5%**

**Query:**

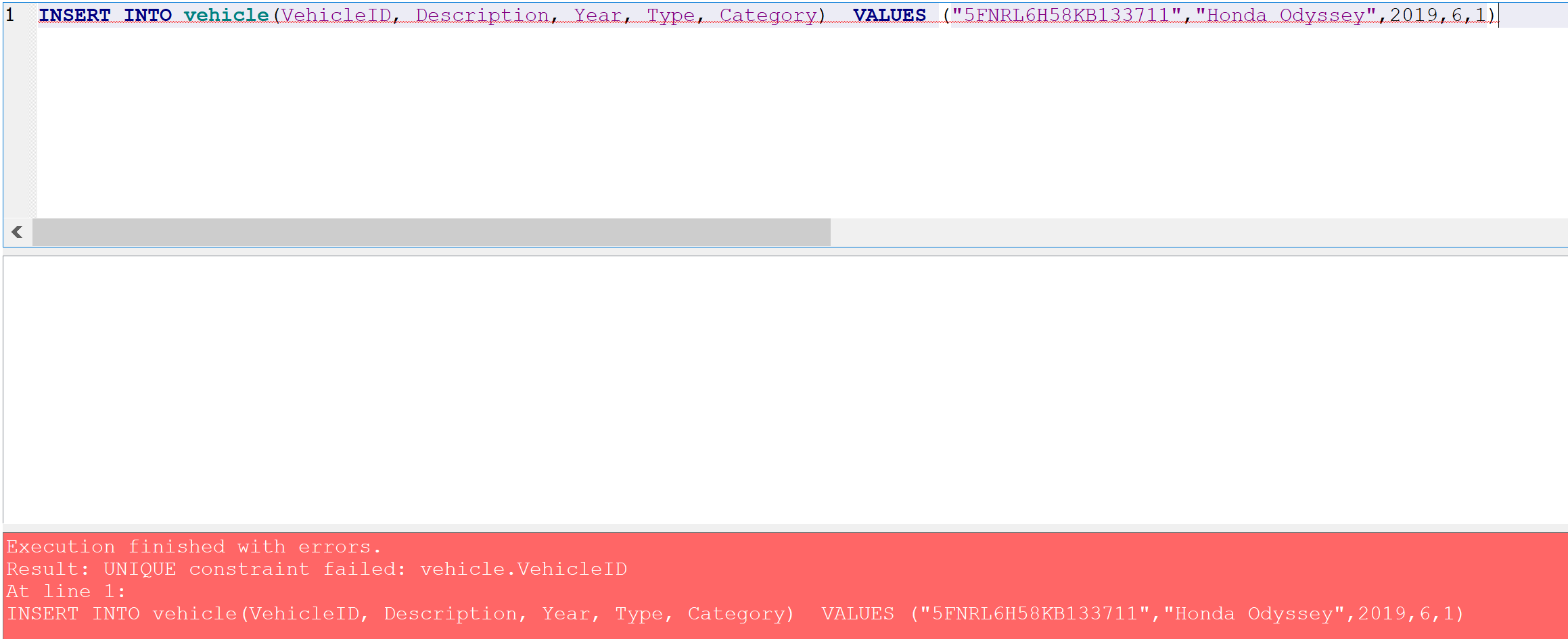
Update rate set Daily = (Daily + 0.05\*Daily) where Category = 1



**Question 4-a: Insert a new luxury van with the following info: Honda Odyssey 2019, vehicle id: 5FNRL6H58KB133711**

**Query**:

INSERT INTO vehicle(VehicleID, Description, Year, Type, Category) VALUES ("5FNRL6H58KB133711","Honda Odyssey",2019,6,1)



**Explanation**: Insert statement fails as the vehicle id already exists in the database and database is designed such that there is unique vehicle id.

**Question 4-b: You also need to insert the following rates**

**Query:**

INSERT INTO rate(Type, Category, Weekly, Daily) VALUES (5,1,900.0,150.00),(6,1,800.00,135.00)



The inset statement fails as the Composite Key (multiple columns are defined as primary key) which is the Type and Category already exist in the table.

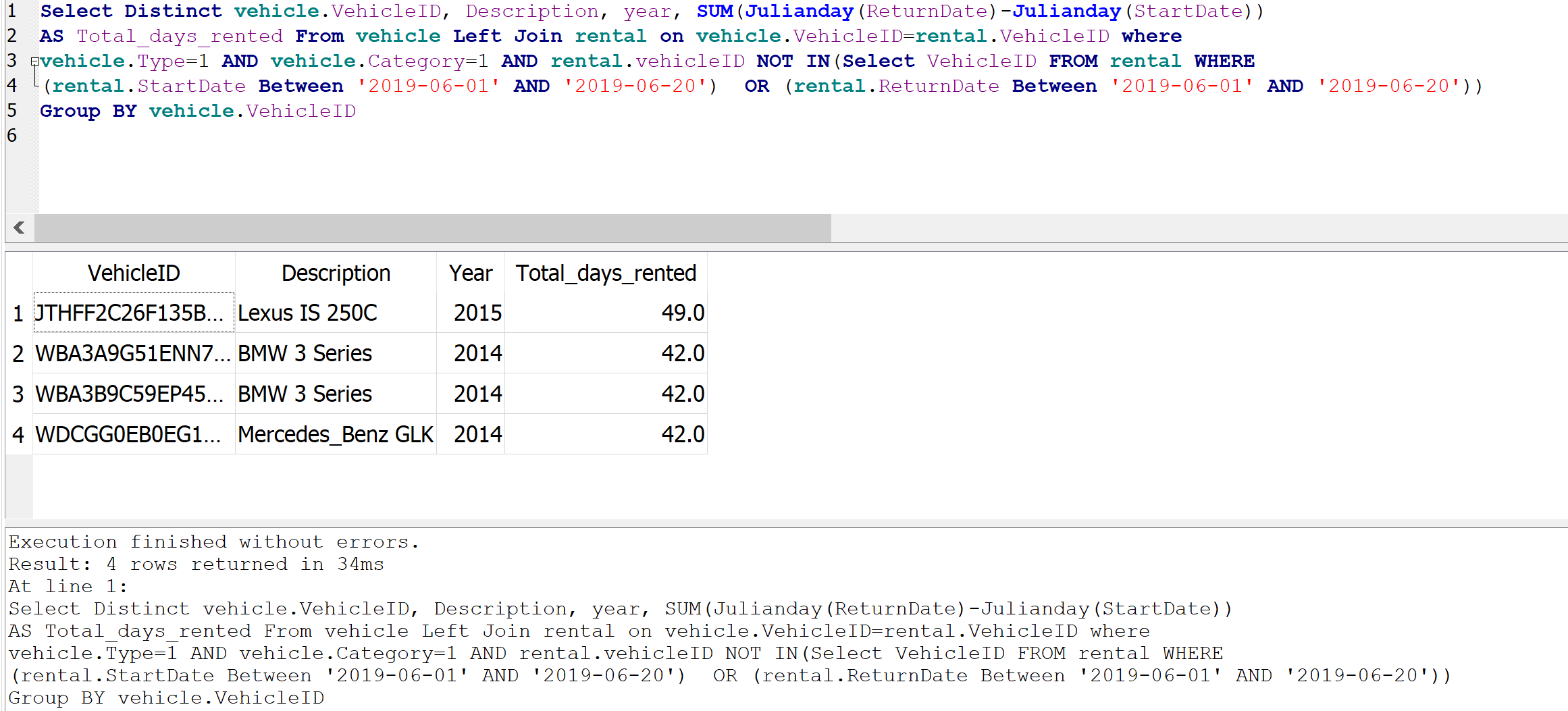
**Question 5: Return all Compact(1) & Luxury(1) vehicles that were available for rent from June 01, 2019**

**until June 20, 2019. List VechicleID as VIN, Description, year, and how many days have been rented so**

**far. You need to change the weeks into days.**

**Query:**

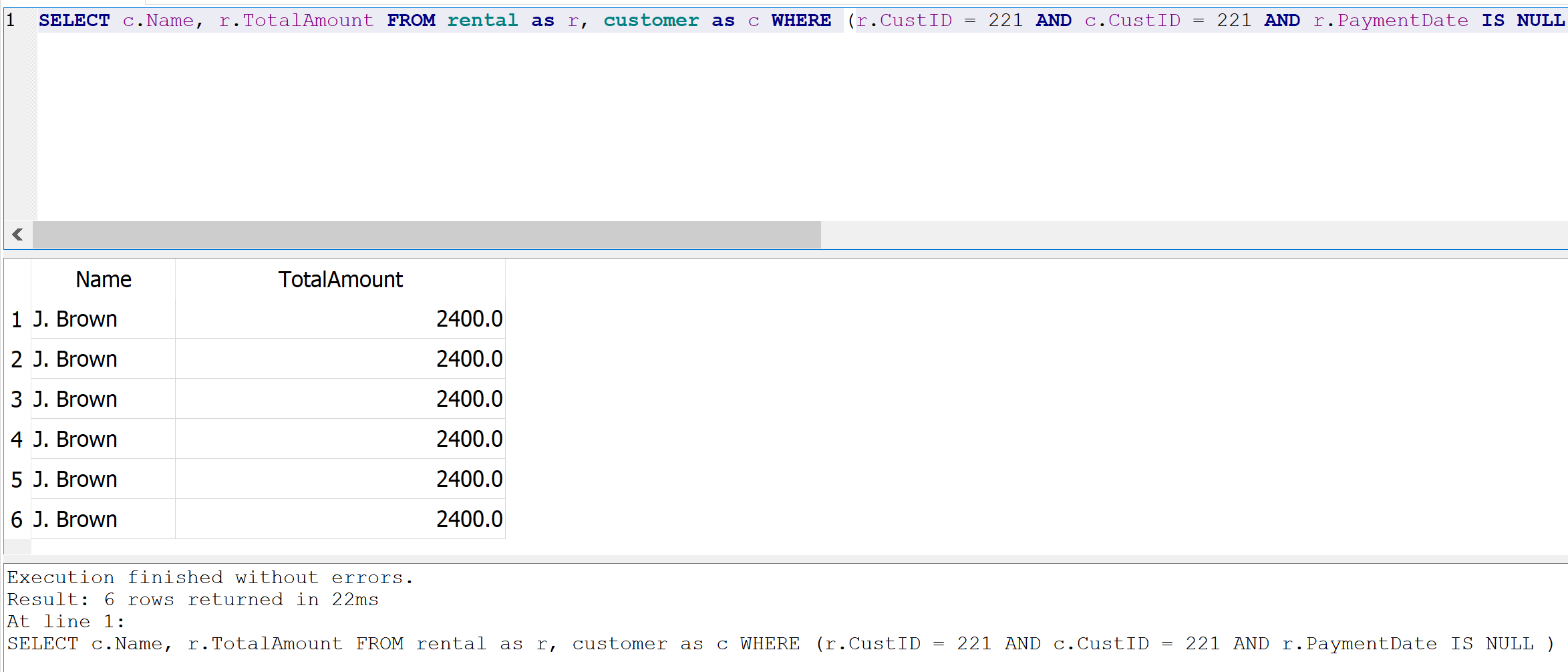
Select Distinct vehicle.VehicleID, Description, year, SUM(Julianday(ReturnDate)-Julianday(StartDate)) AS Total\_days\_rented From vehicle Left Join rental on vehicle.VehicleID=rental.VehicleID where vehicle.Type=1 AND vehicle.Category=1 AND rental.vehicleID NOT IN(Select VehicleID FROM rental WHERE (rental.StartDate Between '2019-06-01' AND '2019-06-20') OR (rental.ReturnDate Between '2019-06-01' AND '2019-06-20')) Group BY vehicle.VehicleID



**Question 6: Return a list with the remaining balance for the customer with the id ‘221’. List customer name, and the balance.**

**Query:**

SELECT c.Name, r.TotalAmount FROM rental as r, customer as c WHERE (r.CustID = 221 AND c.CustID = 221 AND r.PaymentDate IS NULL )



**Question 7: Create a report that will return all vehicles. List the VehicleID as VIN, Description, Year, Type, Category, and Weekly and Daily rates. For the vehicle Type and Category, you need to use the SQL Case statement to substitute the numbers with text. Order your results based on Category (first Luxury and then Basic) and Type based on the Type number, not the text.**

**Query:**

SELECT vehicle.VehicleID as VIN, vehicle.Description, Year, rate.Weekly, rate.Daily,

CASE WHEN vehicle.Type = 1 THEN 'Compact'

WHEN vehicle.Type = 2 THEN 'Medium'

WHEN vehicle.Type = 3 THEN 'Large'

WHEN vehicle.Type = 4 THEN 'SUV'

WHEN vehicle.Type = 5 THEN 'Truck'

WHEN vehicle.Type = 6 THEN 'Van'

END As Type,

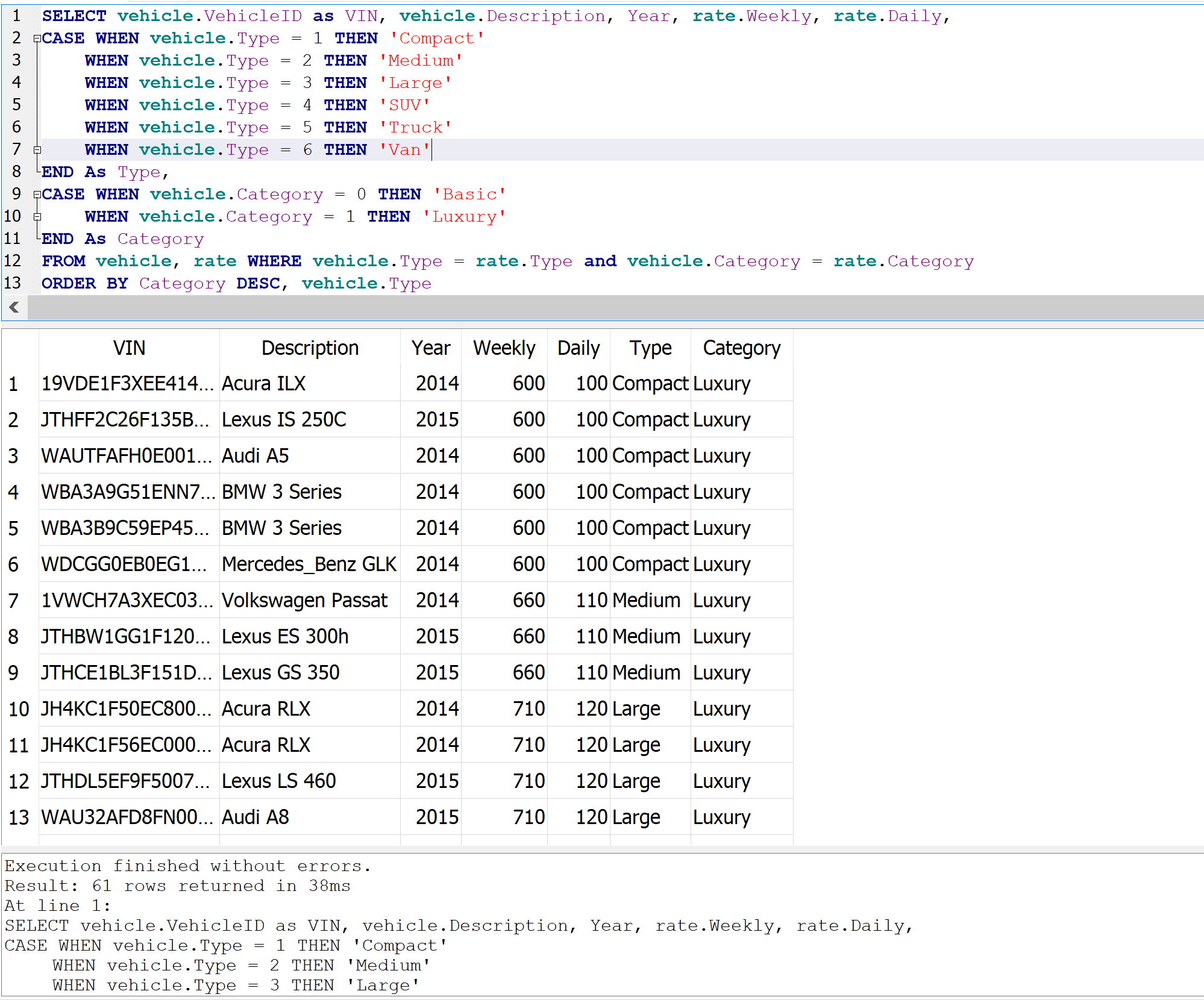
CASE WHEN vehicle.Category = 0 THEN 'Basic'

WHEN vehicle.Category = 1 THEN 'Luxury'

END As Category

FROM vehicle, rate WHERE vehicle.Type = rate.Type and vehicle.Category = rate.Category

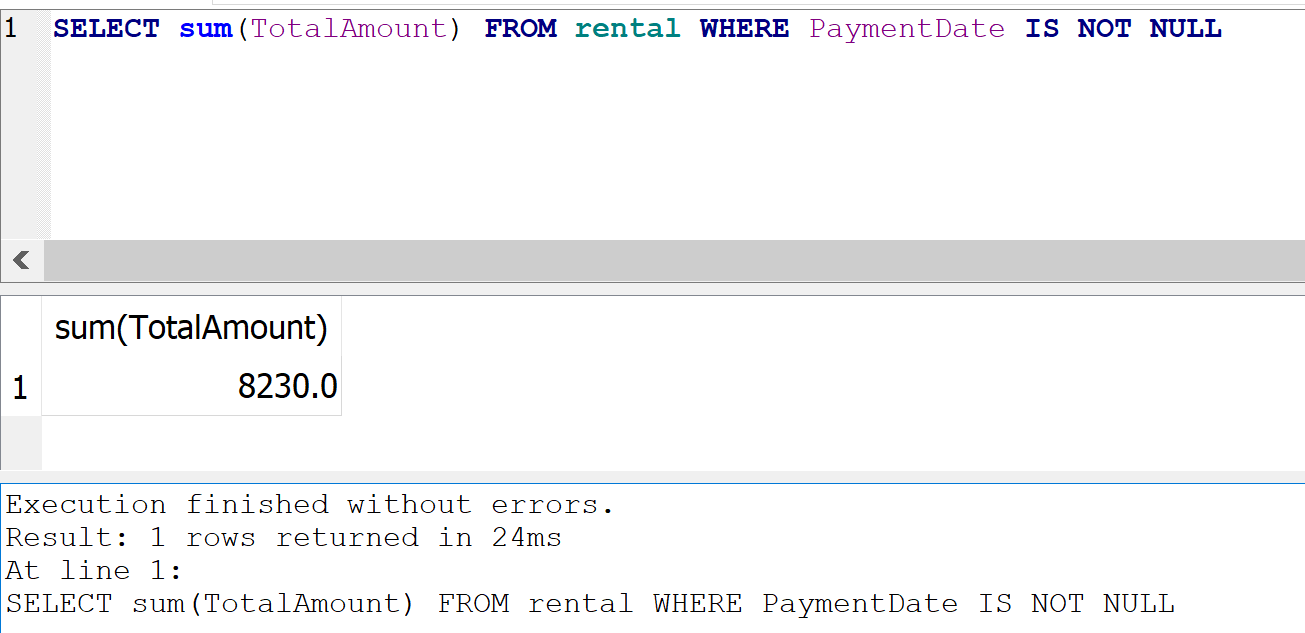
ORDER BY Category DESC, vehicle.Type



**Question 8: What is the total of money that customers paid to us until today?**

**Query:**

SELECT sum(TotalAmount) FROM rental WHERE PaymentDate IS NOT NULL



**Question 9-a: Create a report for the J. Brown customer with all vehicles he rented. List the description, year, type, and category. Also, calculate the unit price for every rental, the total duration mention if it is on weeks or days, the total amount, and if there is any payment. Similarly, as in Question 7, you need to change the numeric values to the corresponding text. Order the results by the StartDate.**

**Query:**

SELECT vehicle.Description, Year, (Julianday(ReturnDate)-Julianday(StartDate)) AS "Total Duration in Days",

Weekly as "Weekly Rates", Daily as "Daily Rates", TotalAmount as "Total Amount",

CASE WHEN PaymentDate != 'NULL' THEN 'Paid'

ELSE 'UnPaid'

END As "Payment Status",

CASE WHEN RentalType = 1 THEN 'Daily'

WHEN RentalType = 7 THEN 'Weekly'

END As "Rental Type",

CASE WHEN vehicle.Type = 1 THEN 'Compact'

WHEN vehicle.Type = 2 THEN 'Medium'

WHEN vehicle.Type = 3 THEN 'Large'

WHEN vehicle.Type = 4 THEN 'SUV'

WHEN vehicle.Type = 5 THEN 'Truck'

WHEN vehicle.Type = 6 THEN 'Van'

END As "Vehicle Type",

CASE WHEN vehicle.Category = 0 THEN 'Basic'

WHEN vehicle.Category = 1 THEN 'Luxury'

END As Category

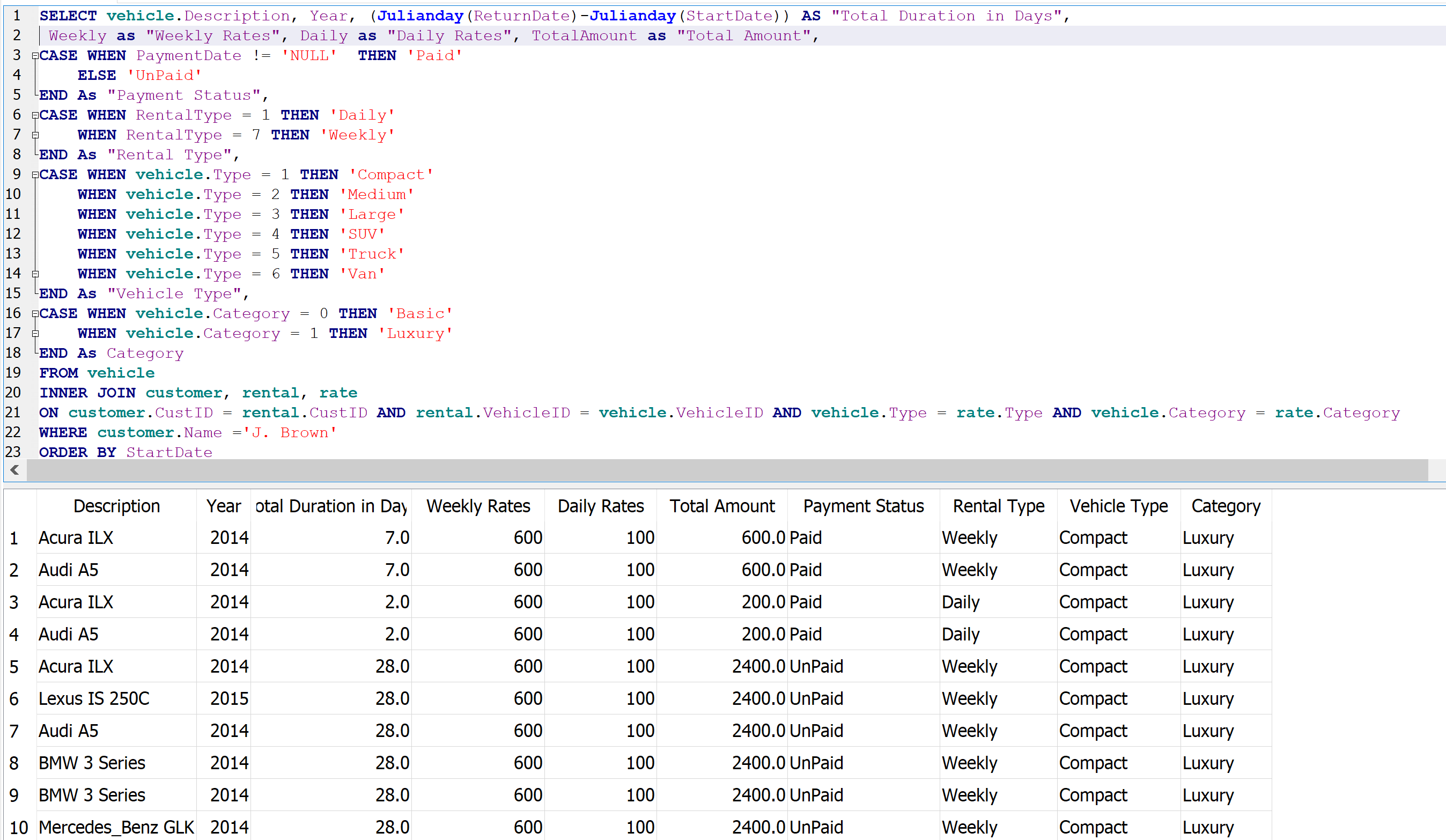
FROM vehicle

INNER JOIN customer, rental, rate

ON customer.CustID = rental.CustID AND rental.VehicleID = vehicle.VehicleID AND vehicle.Type = rate.Type AND vehicle.Category = rate.Category

WHERE customer.Name ='J. Brown'

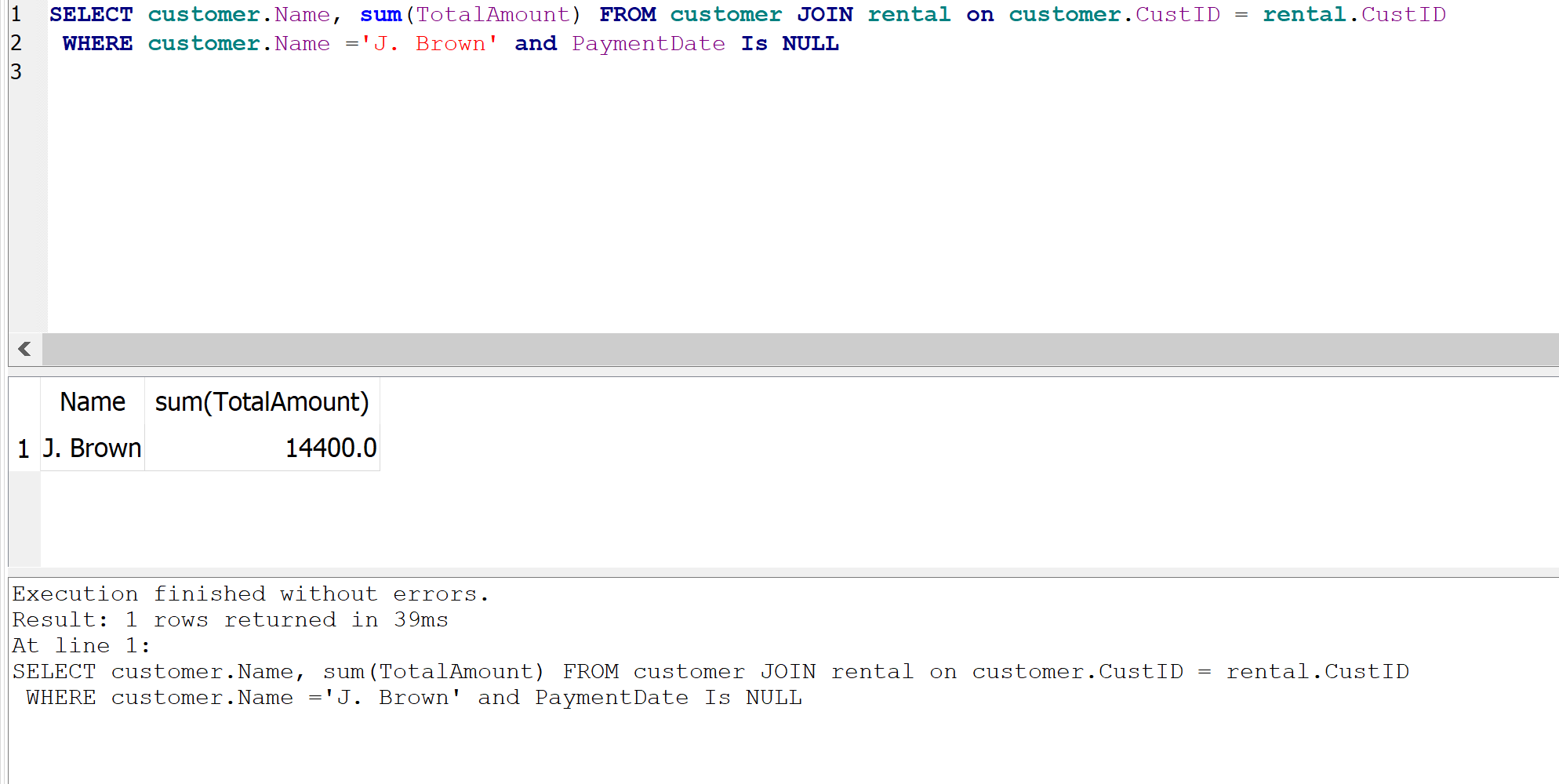
ORDER BY StartDate



**Question 9-b: For the same customer return the current balance**

**Query:**

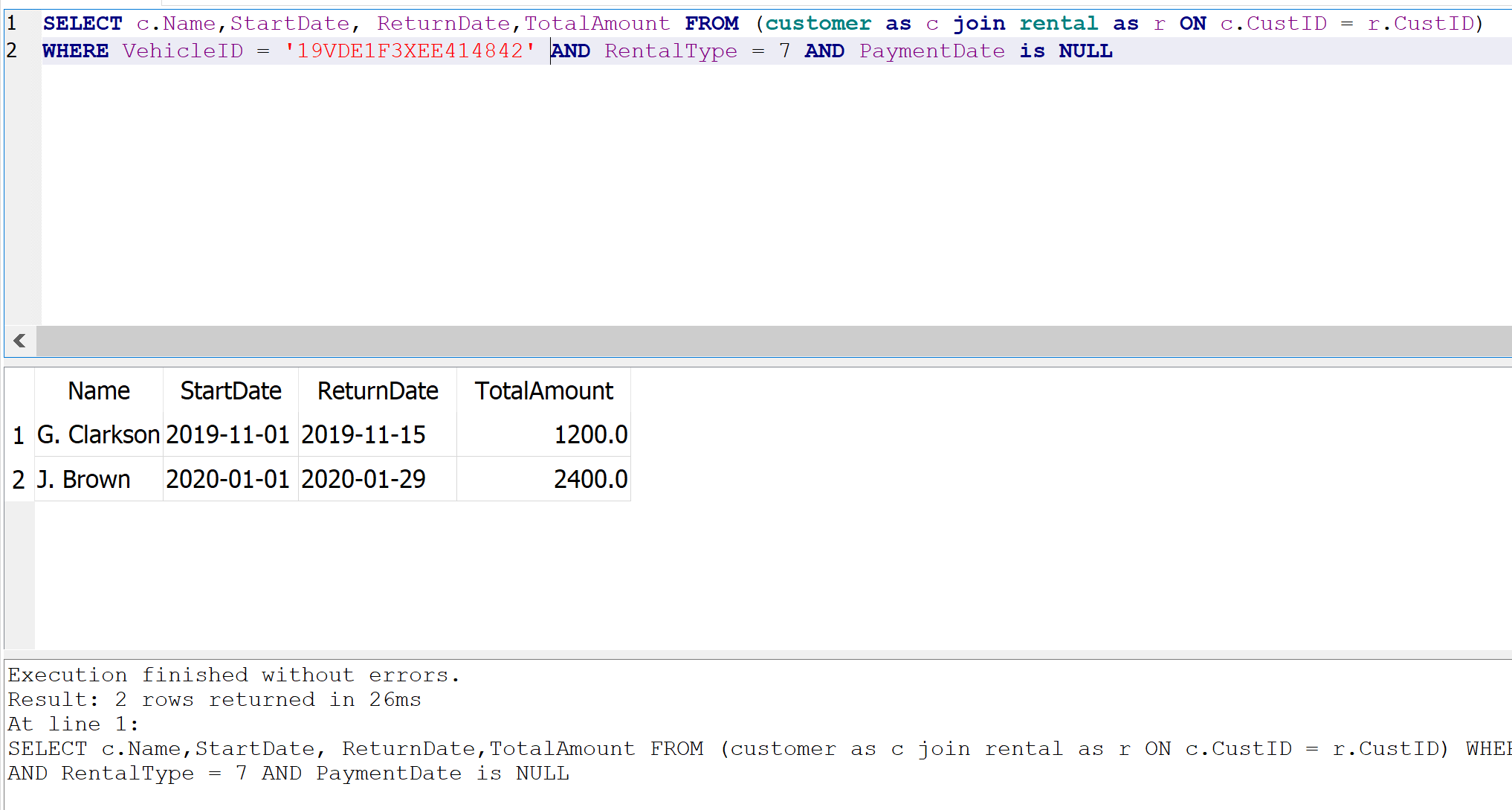
SELECT customer.Name, sum(TotalAmount) FROM customer JOIN rental on customer.CustID = rental.CustID WHERE customer.Name ='J. Brown' and PaymentDate Is NULL



**Question 10: Retrieve all weekly rentals for the vechicleID ‘19VDE1F3XEE414842’ that are not paid yet. List the Customer Name, the start and return date, and the amount.**

**Query**:

SELECT c.Name,StartDate, ReturnDate,TotalAmount FROM (customer as c join rental as r ON c.CustID = r.CustID) WHERE VehicleID = '19VDE1F3XEE414842' AND RentalType = 7 AND PaymentDate is NULL



**Question 11: Return all customers that they never rent a vehicle**

**Query:**

SELECT Name FROM customer WHERE CustID NOT IN(SELECT CustID FROM rental)



**Question 12: Return all rentals that the customer paid on the StartDate. List Customer Name, Vehicle Description, StartDate, ReturnDate, and TotalAmount. Order by Customer Name.**

**Query:**

SELECT c.Name, v.Description, r.StartDate, r.ReturnDate,r.TotalAmount

FROM (customer as c , rental as r, vehicle as v)

WHERE r.PaymentDate = r.StartDate AND c.CustID = r.CustID AND r.VehicleID = v.VehicleID

ORDER BY name

