**PROJECT 2: PHASE 3**

**December 16th ,2020**

**Team Members:**

* **Anish Thakur**
* **Unique Karanjit**
* **Darpan Dhimal**

**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.

Signed by: Anish Thakur, Unique Karanjit, Darpan Dhimal

**Task 1: Execute the following queries on the CarRental2019 database tables:**

**Query 1:**

#adding Returned row to RENTAL table

#cursor.execute("ALTER TABLE RENTAL ADD COLUMN Returned INT")

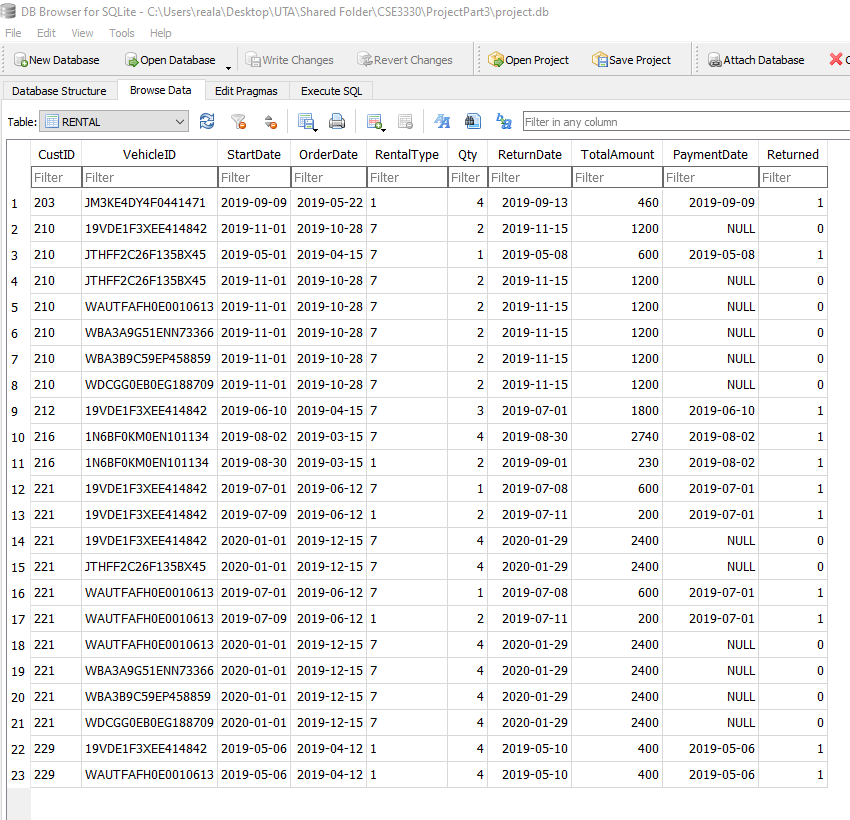
#updating values in Returned column

#change Returned to 0 if no PayementDate

cursor.execute('UPDATE RENTAL SET Returned=0 WHERE PaymentDate="NULL"')

#change Returned to 1 if there is PayementDate

cursor.execute('UPDATE RENTAL SET Returned=1 WHERE PaymentDate!="NULL"')

**OUTPUT**

**Query 2**

CREATE VIEW vRentalInfo AS

SELECT RENTAL.OrderDate, RENTAL.StartDate, RENTAl.ReturnDate,

RENTAL.qty\*RENTAL.RentalType AS TotalDays,

RENTAL.VehicleID AS VIN,

VEHICLE.DESCRIPTION AS VEHICLE,

CASE VEHICLE.Type

WHEN 1 THEN 'Compact'

WHEN 2 THEN 'Medium'

WHEN 3 THEN 'Large'

WHEN 4 THEN 'SUV'

WHEN 5 THEN 'Truck'

WHEN 6 THEN 'VAN'

END AS Type,

CASE VEHICLE.Category

WHEN 1 THEN 'LUXURY'

WHEN 0 THEN 'BASIC'

END AS Category,

CUSTOMER.CustID AS CustomerID,

CUSTOMER.Name AS CustomerName,

RENTAL.TotalAmount AS OrderAmount,

CASE

WHEN RENTAL.PaymentDate IS NOT NULL THEN 0

ELSE Rental.TotalAmount

END AS RentalBalance

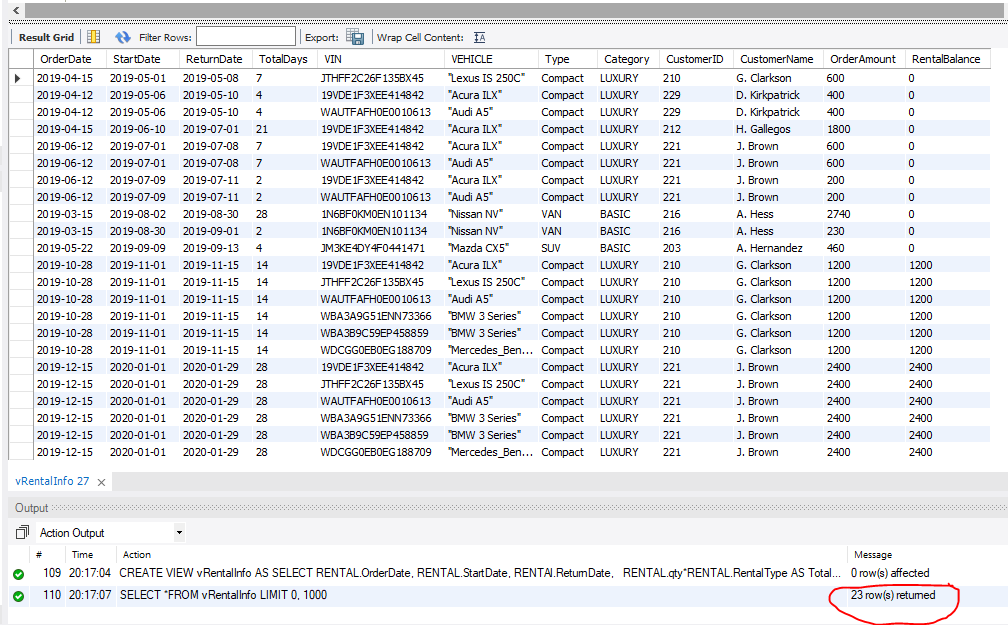
FROM VEHICLE, RENTAL,CUSTOMER

WHERE VEHICLE.VehicleID=RENTAL.VehicleID AND CUSTOMER.CustID=RENTAL.CustID

ORDER BY RENTAL.StartDate;

2. SELECT \*FROM vRentalInfo;

**OUTPUT**



**Task 2: Create a GUI for the CarRental2019 database:**

Requirements:

1. *The first requirement is to add information about a new customer. Do not provide the customer ID in your query. Submit your editable SQL query that your code executes.*

cursor.execute("””INSERT INTO CUSTOMER(Name,Phone) VALUES(:name,:phone)””",

{

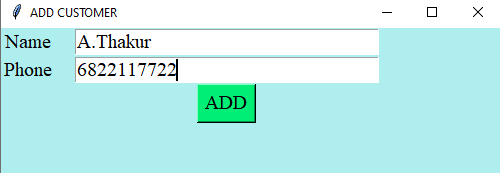
'name':name.get(),

'phone':phone.get()

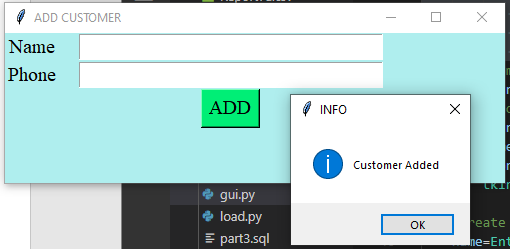
})

Screenshot:

*Before Adding:*



*After Adding:*



1. *The second requirement is to add all the information about a new vehicle. Submit your editable SQL query that your code executes*

cursor.execute("””INSERT INTO VEHICLE VALUES(:VIN,:Description,:Year,:Type,:Category)””",

{

'VIN':VIN.get(),

'Description':Description.get(),

'Year':Year.get(),

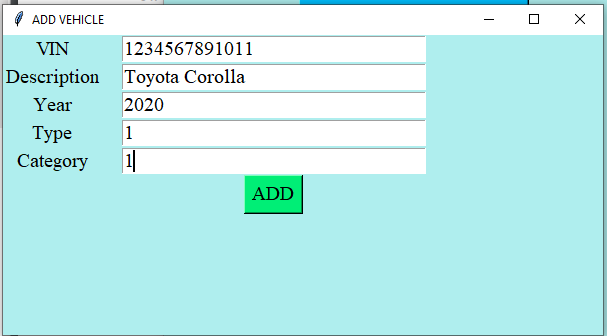
'Type':Type.get(),

'Category':Category.get()

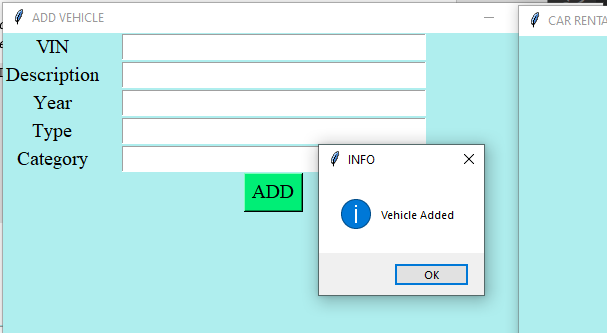
})

*Screenshot*

*Before Adding:*



*After Adding:*



1. *The third requirement is to add all the information about a new rental reservation (this must find a free vehicle of the appropriate type and category for a specific rental period). We assume that the customer has the right either to pay at the order or return date. Submit your editable SQL queries (select available vehicles & insert rental) that your code executes*
2. *The fourth requirement is to handle the return of a rented car. This transaction should print the total customer payment due for that rental, enter it in the database and update the returned attribute accordingly. You need to be able to retrieve a rental by the return date, customer name (the table needs the id), and vehicle info. Submit your editable SQL queries (retrieve & update rental) that your code executes.*
3. *List for every customer the ID, name, and if there is any remaining balance. The user has the right to search either by a customer’s ID, name, part of the name, or to run the query with no filters/criteria. The amount needs to be in US dollars. For customers with zero (0) or NULL balance, you need to return zero dollars ($0.00). Make sure that your query returns meaningful attribute names. In the case that the user decides not to provide any filters, order the results based on the balance amount. Make sure that you return all records. Submit your editable SQL query that your code executes*

*cursor=conn.cursor()*

*if len(custName.get())==0 or len(custId.get())==0:*

*cursor.execute("""SELECT Name,*

*CASE*

*WHEN R.PaymentDate="NULL" THEN SUM(TotalAmount)*

*ELSE "0.00"*

*END AS Balance*

*FROM CUSTOMER AS C, RENTAL AS R*

*WHERE R.CustID=C.CustID*

*GROUP BY NAME*

*ORDER BY Balance""")*

*result=cursor.fetchall()*

*else:*

*cursor.execute("""SELECT Name,*

*CASE*

*WHEN R.PaymentDate="NULL" THEN SUM(TotalAmount)*

*ELSE "0.00"*

*END AS Balance*

*FROM CUSTOMER AS C, RENTAL AS R*

*WHERE R.CustID=C.CustID AND C.CustID=:custID*

*GROUP BY NAME""",*

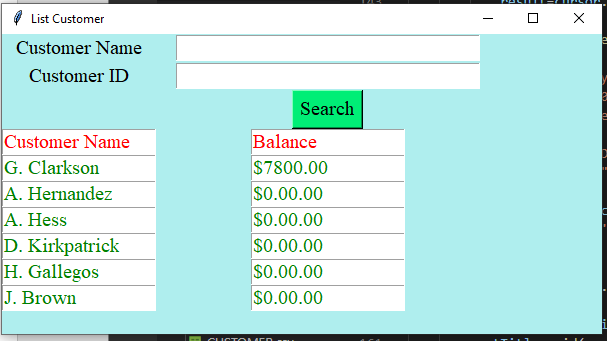
*{*

*'custID':custId.get(),*

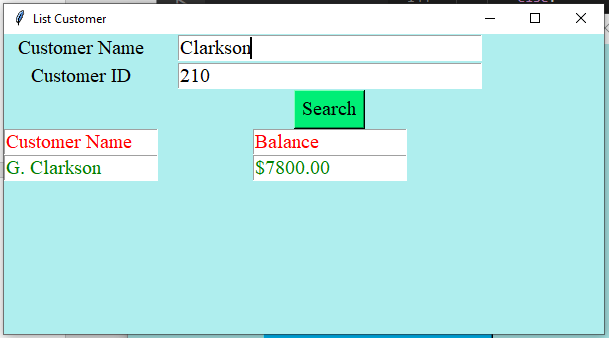
*'custName':custId.get()})*

*Screenshots:*

*Without any user input, after hitting search:*



*With user input, after hitting search:*



1. *List for every vehicle the VIN, the description, and the average DAILY price. The user has the right either to search by the VIN, vehicle’s description, part of the description, or to run the query with no filters/criteria. An example criterion would be all ‘BMW’ vehicles. The amount needs to be in US dollars. The average DAILY price derives from the rental table, and the amount needs to have two decimals as well as the dollar ‘$’ sign. For vehicles that they do not have any rentals, you need to substitute the NULL value with a ‘Non-Applicable’ text. Make sure that your query returns meaningful attribute names. In the case that the user decides not to provide any filters, order the results based on the average daily price. Submit your editable SQL query that your code executes*

cursor.execute("""SELECT V.VehicleID,Description,

CASE

WHEN V.VehicleID NOT IN (SELECT RENTAL.VehicleID FROM RENTAL) THEN "NON APPLICABLE"

ELSE CAST(Daily AS NUMERIC(10,2))\*100/100

END AS DailyPrice

FROM VEHICLE AS V,RATE AS R, RENTAL

WHERE V.Type=R.Type AND V.Category=R.Category

GROUP BY V.VehicleID

ORDER BY DailyPrice

""")

result=cursor.fetchall()

else:

cursor.execute("""SELECT V.VehicleID,Description,

CASE

WHEN V.VehicleID NOT IN (SELECT RENTAL.VehicleID FROM RENTAL) THEN "NON APPLICABLE"

ELSE Daily

END AS DailyPrice

FROM VEHICLE AS V,RATE AS R, RENTAL

WHERE V.Type=R.Type AND V.Category=R.Category

AND V.VehicleID=:VIN

GROUP BY V.VehicleID

ORDER BY DailyPrice

""",

{

'VIN':VIN.get(),

})

*Screenshots:*

*Without any user input, after hitting search:*

*With user input, after hitting search:*

