CSE-3330: Database & File Structures

Project 2-2

Fall 2020

Prof. Guizani

Team Members:

- ✓ **Suman Thapa Magar (Team Leader), Student ID: 1001643016, Section: 003**
  - Overall gave major contributions: Created tables for CUSTOMER & RENTAL.
  - Coded for SQL query no. 5, 6, 7, 8, 9a, 9b
  - Successfully implemented constraints and use of FOREIGN and PRIMARY KEYS
- ✓ **Yunika Upadhayaya, Student ID: 1001631183, Section: 003**
  - Created a table for RATE.
  - Coded for SQL query no. 1, 2, 3, 4a, 4b, 10
  - Found a way to load files to each table.
- ✓ **Pratik Mahato, Student ID: 1001661375, Section: 004**
  - Created a table for VEHICLE.
  - Coded for SQL query no. 11 & 12
  - Created docx file that includes the tasks 1, 2 & 3.

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.

Pratik Mahato | *Yunika Upadhayaya* | *Suman Thapa Magar*

Contents

Task 1 3

Table Commands 3

Explanation on Table Creation 4

Task 2 5

Methodology 5

Verifying by SQL Queries 11

Challenges 12

Task 3 13

Query 1 13

Query 2 14

Query 3 14

Query 4a 15

Query 4b 16

Query 5 17

Query 6 18

Query 7 19

Query 8 21

Query 9a 21

Query 9b 22

Query 10 23

Query 11 23

Query 12 25

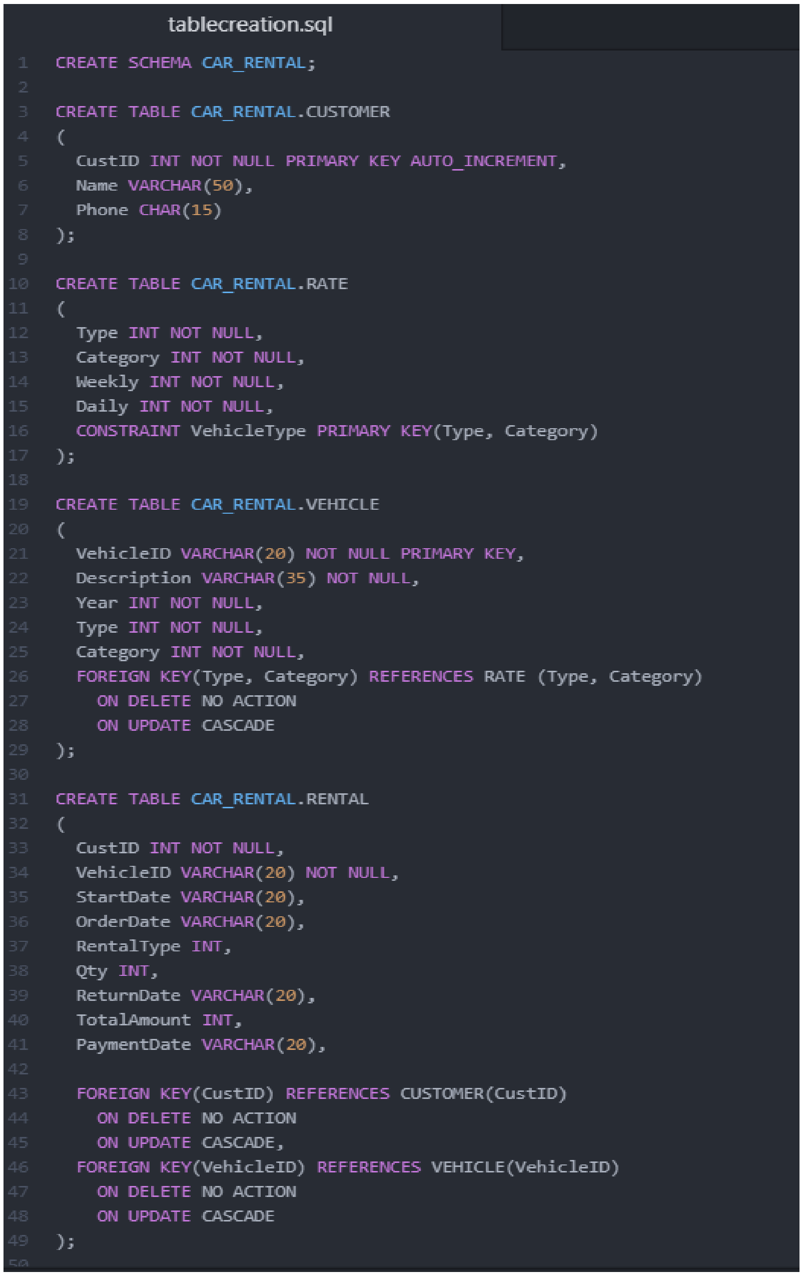
Challenges 25

References 25

Task 1

Table Commands

The screenshot below from the source code shows the necessary commands to create tables for the car rental database schema. After the code was typed, they were selected and ‘’ was clicked on MySQL workbench to create the tables.



```
CREATE SCHEMA CAR_RENTAL;

CREATE TABLE CAR_RENTAL.CUSTOMER
(
  CustID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
  Name VARCHAR(50),
  Phone CHAR(15)
);

CREATE TABLE CAR_RENTAL.RATE
(
  Type INT NOT NULL,
  Category INT NOT NULL,
  Weekly INT NOT NULL,
  Daily INT NOT NULL,
  CONSTRAINT VehicleType PRIMARY KEY(Type, Category)
);

CREATE TABLE CAR_RENTAL.VEHICLE
(
  VehicleID VARCHAR(20) NOT NULL PRIMARY KEY,
  Description VARCHAR(35) NOT NULL,
  Year INT NOT NULL,
  Type INT NOT NULL,
  Category INT NOT NULL,
  FOREIGN KEY(Type, Category) REFERENCES RATE (Type, Category)
    ON DELETE NO ACTION
    ON UPDATE CASCADE
);

CREATE TABLE CAR_RENTAL.RENTAL
(
  CustID INT NOT NULL,
  VehicleID VARCHAR(20) NOT NULL,
  StartDate VARCHAR(20),
  OrderDate VARCHAR(20),
  RentalType INT,
  Qty INT,
  ReturnDate VARCHAR(20),
  TotalAmount INT,
  PaymentDate VARCHAR(20),

  FOREIGN KEY(CustID) REFERENCES CUSTOMER(CustID)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  FOREIGN KEY(VehicleID) REFERENCES VEHICLE(VehicleID)
    ON DELETE NO ACTION
    ON UPDATE CASCADE
);
```

Explanation on Table Creation

Using the given requirement attributes, tables were made for the CAR_RENTAL schema. Primary keys, foreign keys and necessary constraints added for each table is as follows:

Table CUSTOMER:

Primary key is CustID, - this id helps to keep track of rented vehicles by the customer from other tables. It cannot be null, that is each customer is designated with unique CustID. With the help of Auto_Increment, each upcoming row from customer file is incremented for the primary key CustID.

Table RATE:

Type and Category under this table are combined together to form a composite primary key. This composite key is used to reference the daily and weekly rental rates of different kinds of vehicles.

Table VEHICLE:

VehicleID is used as primary key and Type & Category attributes are taken as composite foreign key from table RATE. This foreign key is used to find the associated weekly and monthly rates of vehicles.

Table RENTAL:

CustID from table CUSTOMER and VehicleID from table VEHICLE are considered as foreign keys. These keys help to keep track of the customer information and the vehicles they have rented.

Task 2

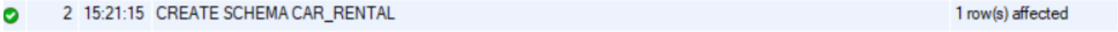
Methodology

The coding for the project was done via **MySQL Workbench.**

- First, the following code was executed to create a schema for our database.

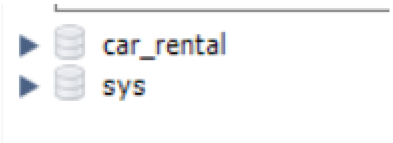
```
CREATE SCHEMA CAR_RENTAL;
```

When execute sign is clicked for above code, following would appear in output console:

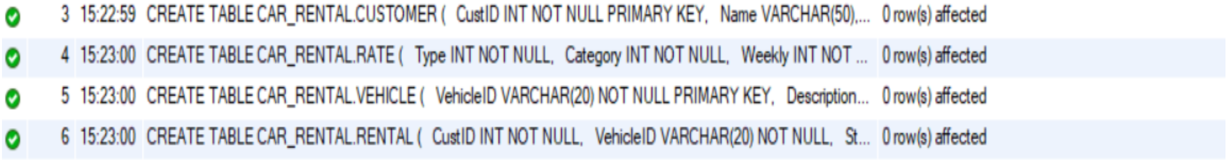


```
2  15:21:15  CREATE SCHEMA CAR_RENTAL                                              1 row(s) affected
```

After this line of code execution, this schema would appear in the Schemas Navigator.

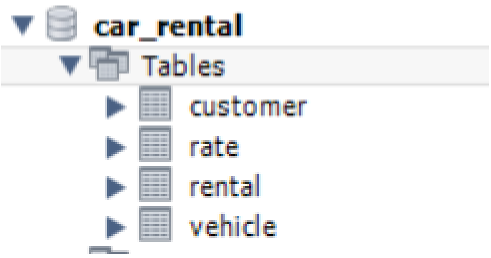


- The tables were created for CUSTOMER, RATE, VEHICLE, RENTAL as per the code shown in our source file. All the code for tables were highlighted separately and executed one after another. Output console:



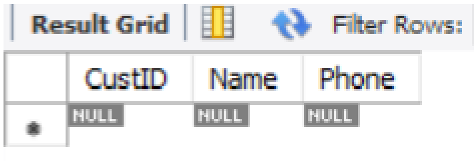
```
3  15:22:59  CREATE TABLE CAR_RENTAL.CUSTOMER (  CustID INT NOT NULL PRIMARY KEY,  Name VARCHAR(50),...  0 row(s) affected
4  15:23:00  CREATE TABLE CAR_RENTAL.RATE (  Type INT NOT NULL,  Category INT NOT NULL,  Weekly INT NOT ...  0 row(s) affected
5  15:23:00  CREATE TABLE CAR_RENTAL.VEHICLE (  VehicleID VARCHAR(20) NOT NULL PRIMARY KEY,  Description...  0 row(s) affected
6  15:23:00  CREATE TABLE CAR_RENTAL.RENTAL (  CustID INT NOT NULL,  VehicleID VARCHAR(20) NOT NULL,  St...  0 row(s) affected
```

After this execution, tables would be created inside the Car_Rental schema and can be accessed under tables in the Navigator.



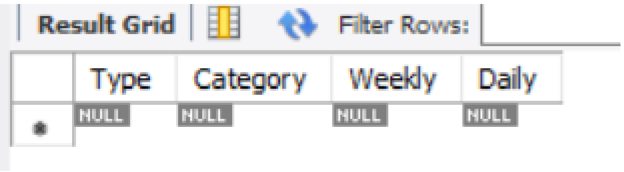
Each table attributes can be confirmed by navigating through them under Schemas:

CUSTOMER:



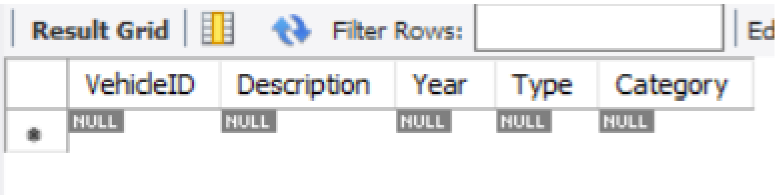
	CustID	Name	Phone
●	NULL	NULL	NULL

RATE:



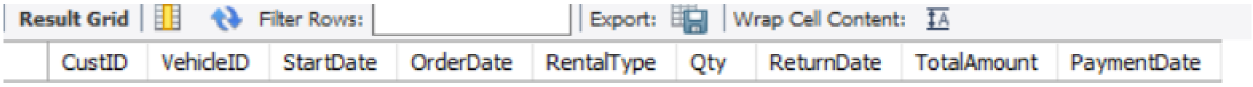
	Type	Category	Weekly	Daily
●	NULL	NULL	NULL	NULL

VEHICLE:



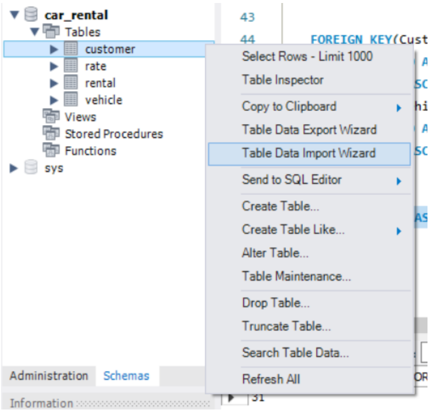
	VehicleID	Description	Year	Type	Category
●	NULL	NULL	NULL	NULL	NULL

RENTAL:



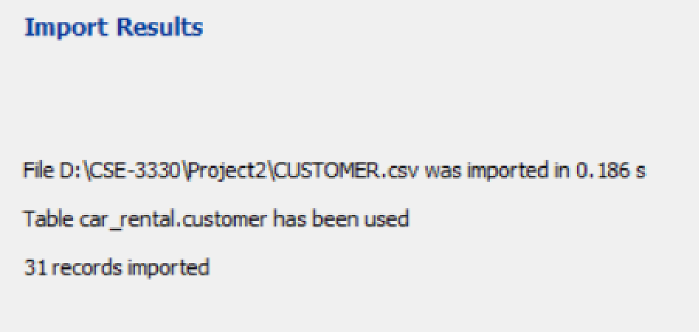
	CustID	VehicleID	StartDate	OrderDate	RentalType	Qty	ReturnDate	TotalAmount	PaymentDate

- The respective files in form of .csv extension were imported into table through Table Data Import Wizard. This method to import files was used for each table.



To verify, if the files have been loaded up, the result grid view of any table can be clicked.
For example:

Result Grid of CUSTOMER:

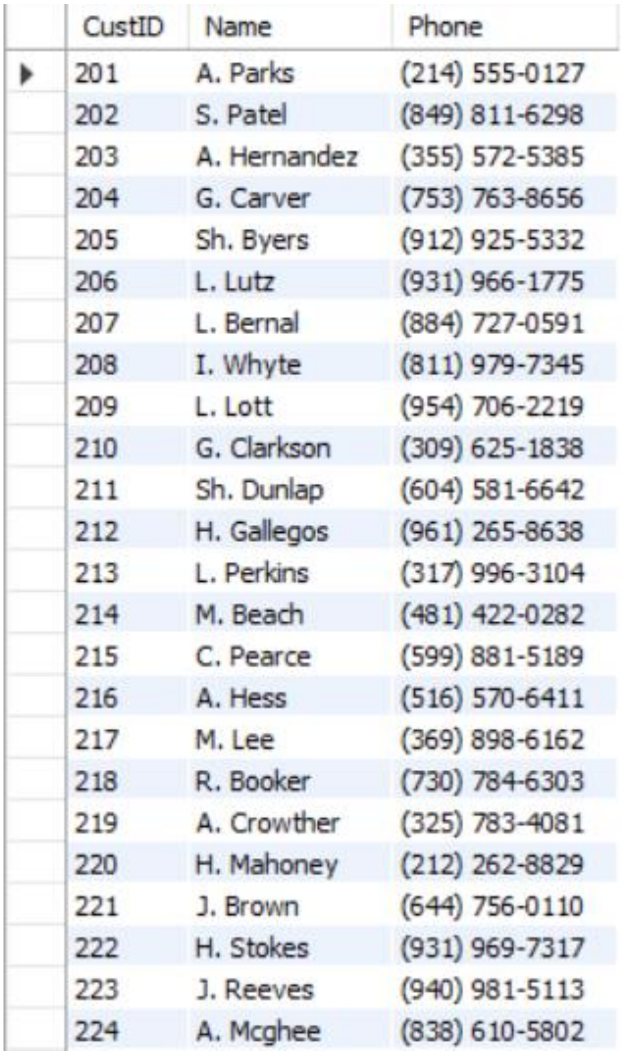


Import Results

File D:\CSE-3330\Project2\CUSTOMER.csv was imported in 0.186 s

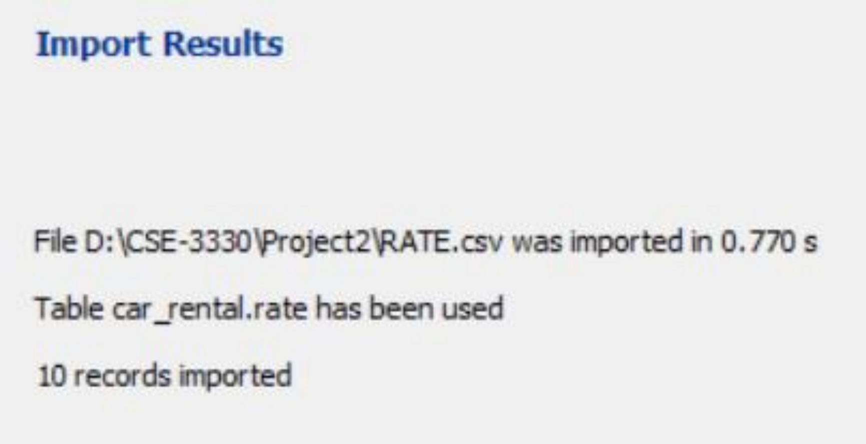
Table car_rental.customer has been used

31 records imported



	CustID	Name	Phone
▶	201	A. Parks	(214) 555-0127
	202	S. Patel	(849) 811-6298
	203	A. Hernandez	(355) 572-5385
	204	G. Carver	(753) 763-8656
	205	Sh. Byers	(912) 925-5332
	206	L. Lutz	(931) 966-1775
	207	L. Bernal	(884) 727-0591
	208	I. Whyte	(811) 979-7345
	209	L. Lott	(954) 706-2219
	210	G. Clarkson	(309) 625-1838
	211	Sh. Dunlap	(604) 581-6642
	212	H. Gallegos	(961) 265-8638
	213	L. Perkins	(317) 996-3104
	214	M. Beach	(481) 422-0282
	215	C. Pearce	(599) 881-5189
	216	A. Hess	(516) 570-6411
	217	M. Lee	(369) 898-6162
	218	R. Booker	(730) 784-6303
	219	A. Crowther	(325) 783-4081
	220	H. Mahoney	(212) 262-8829
	221	J. Brown	(644) 756-0110
	222	H. Stokes	(931) 969-7317
	223	J. Reeves	(940) 981-5113
	224	A. Mcghee	(838) 610-5802

Result Grid of RATE:

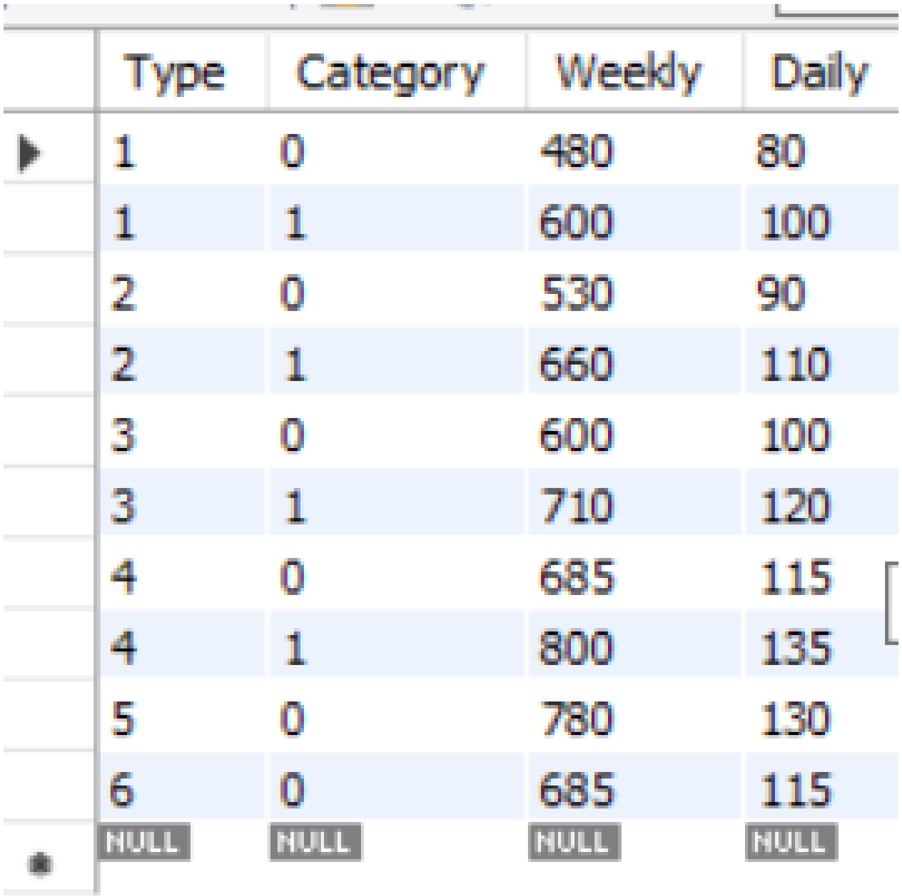


Import Results

File D:\CSE-3330\Project2\RATE.csv was imported in 0.770 s

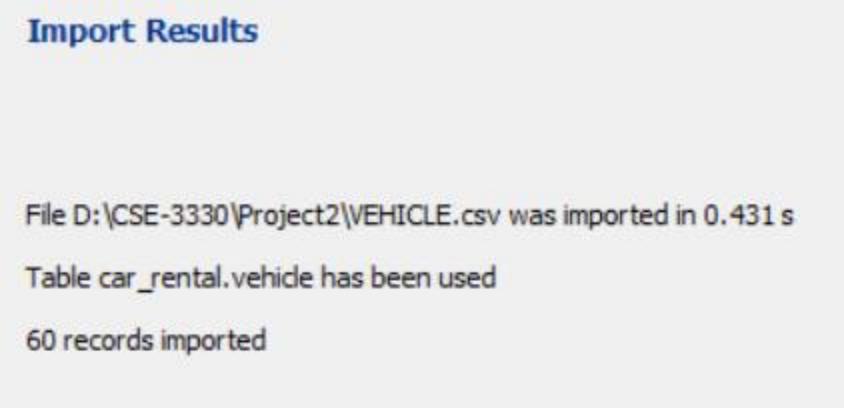
Table car_rental.rate has been used

10 records imported



Type	Category	Weekly	Daily
1	0	480	80
1	1	600	100
2	0	530	90
2	1	660	110
3	0	600	100
3	1	710	120
4	0	685	115
4	1	800	135
5	0	780	130
6	0	685	115
NULL	NULL	NULL	NULL

Result Grid of VEHICLE:

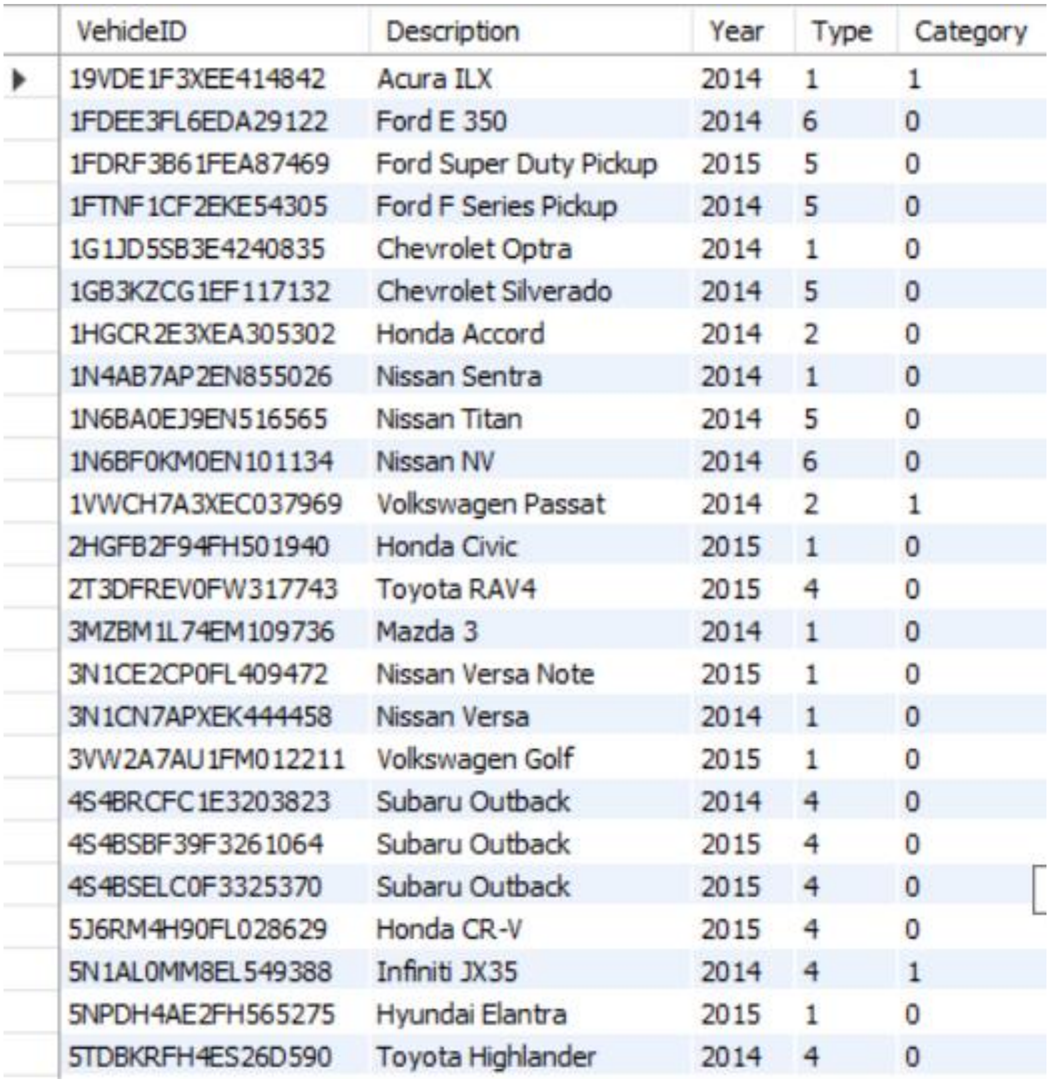


Import Results

File D:\CSE-3330\Project2\VEHICLE.csv was imported in 0.431 s

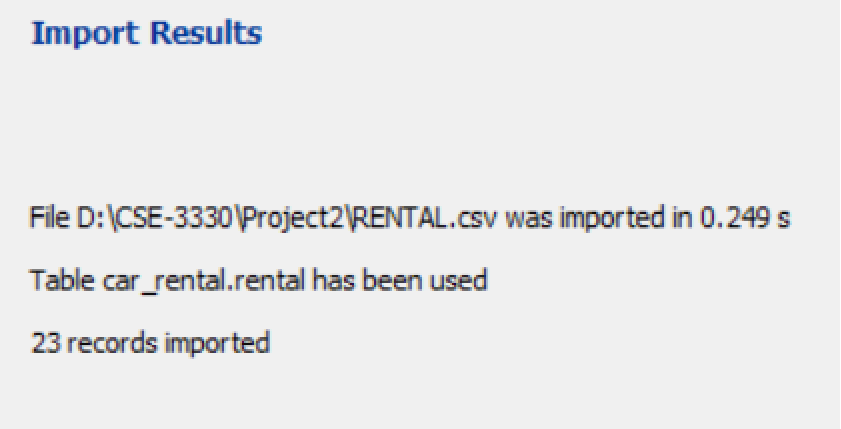
Table car_rental.vehicle has been used

60 records imported



VehicleID	Description	Year	Type	Category
19VDE1F3XEE414842	Acura ILX	2014	1	1
1FDEE3FL6EDA29122	Ford E 350	2014	6	0
1FDRF3B61FEA87469	Ford Super Duty Pickup	2015	5	0
1FTNF1CF2EKE54305	Ford F Series Pickup	2014	5	0
1G1JD5SB3E4240835	Chevrolet Optra	2014	1	0
1GB3KZCG1EF117132	Chevrolet Silverado	2014	5	0
1HGCR2E3XEA305302	Honda Accord	2014	2	0
1N4AB7AP2EN855026	Nissan Sentra	2014	1	0
1N6BA0EJ9EN516565	Nissan Titan	2014	5	0
1N6BF0KM0EN101134	Nissan NV	2014	6	0
1VWCH7A3XEC037969	Volkswagen Passat	2014	2	1
2HGFB2F94FH501940	Honda Civic	2015	1	0
2T3DFREV0FW317743	Toyota RAV4	2015	4	0
3MZBM1L74EM109736	Mazda 3	2014	1	0
3N1CE2CP0FL409472	Nissan Versa Note	2015	1	0
3N1CN7APXEK444458	Nissan Versa	2014	1	0
3VW2A7AU1FM012211	Volkswagen Golf	2015	1	0
4S4BRCFC1E3203823	Subaru Outback	2014	4	0
4S4BSBF39F3261064	Subaru Outback	2015	4	0
4S4BSELC0F3325370	Subaru Outback	2015	4	0
5J6RM4H90FL028629	Honda CR-V	2015	4	0
5N1AL0MM8EL549388	Infiniti JX35	2014	4	1
5NPDH4AE2FH565275	Hyundai Elantra	2015	1	0
5TDBKRFH4ES26D590	Toyota Highlander	2014	4	0

Result Grid of RENTAL:

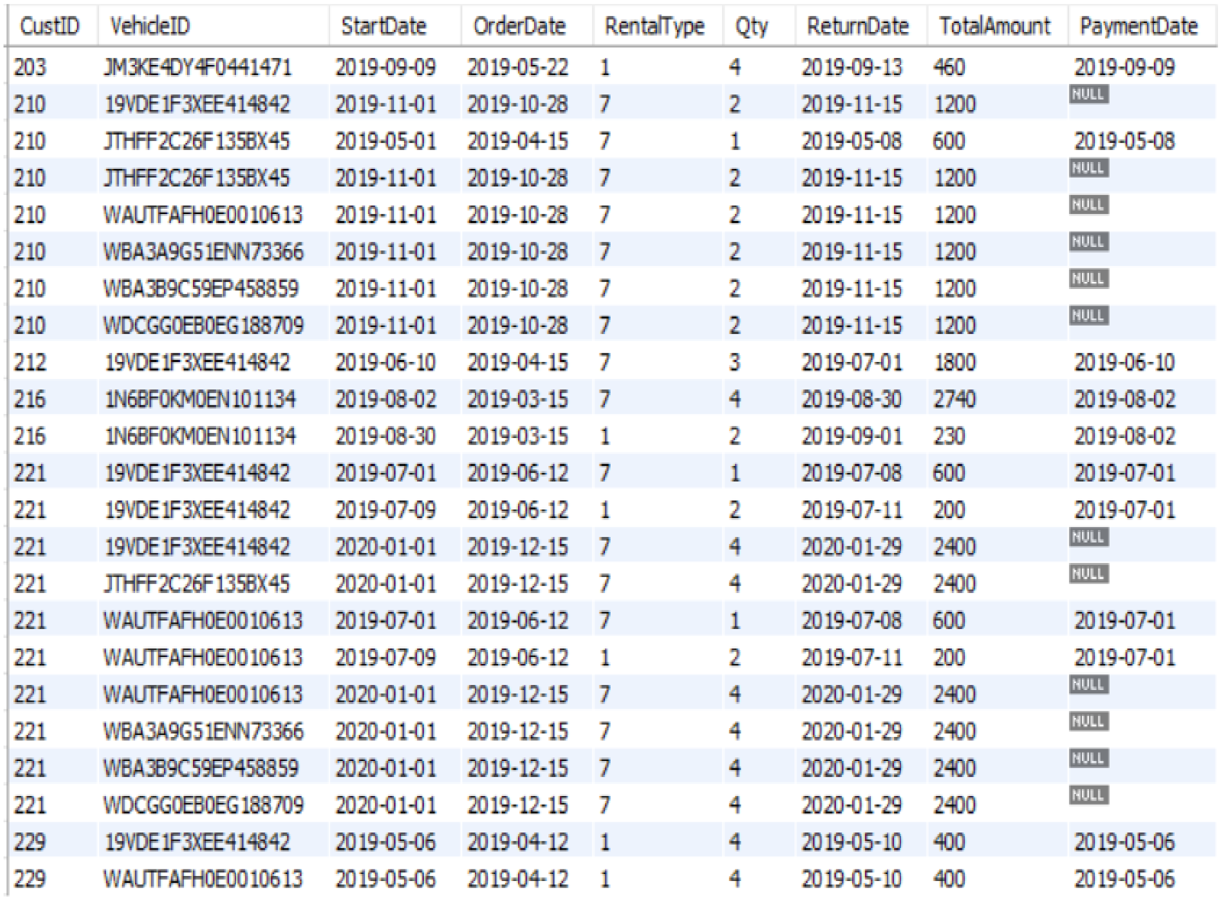


Import Results

File D:\CSE-3330\Project2\RENTAL.csv was imported in 0.249 s

Table car_rental.rental has been used

23 records imported



CustID	VehicleID	StartDate	OrderDate	RentalType	Qty	ReturnDate	TotalAmount	PaymentDate
203	JM3KE4DY4F0441471	2019-09-09	2019-05-22	1	4	2019-09-13	460	2019-09-09
210	19VDE1F3XEE414842	2019-11-01	2019-10-28	7	2	2019-11-15	1200	NULL
210	JTHFF2C26F135BX45	2019-05-01	2019-04-15	7	1	2019-05-08	600	2019-05-08
210	JTHFF2C26F135BX45	2019-11-01	2019-10-28	7	2	2019-11-15	1200	NULL
210	WAUTFAFH0E0010613	2019-11-01	2019-10-28	7	2	2019-11-15	1200	NULL
210	WBA3A9G51ENN73366	2019-11-01	2019-10-28	7	2	2019-11-15	1200	NULL
210	WBA3B9C59EP458859	2019-11-01	2019-10-28	7	2	2019-11-15	1200	NULL
210	WDCGG0EB0EG188709	2019-11-01	2019-10-28	7	2	2019-11-15	1200	NULL
212	19VDE1F3XEE414842	2019-06-10	2019-04-15	7	3	2019-07-01	1800	2019-06-10
216	1N6BF0KM0EN101134	2019-08-02	2019-03-15	7	4	2019-08-30	2740	2019-08-02
216	1N6BF0KM0EN101134	2019-08-30	2019-03-15	1	2	2019-09-01	230	2019-08-02
221	19VDE1F3XEE414842	2019-07-01	2019-06-12	7	1	2019-07-08	600	2019-07-01
221	19VDE1F3XEE414842	2019-07-09	2019-06-12	1	2	2019-07-11	200	2019-07-01
221	19VDE1F3XEE414842	2020-01-01	2019-12-15	7	4	2020-01-29	2400	NULL
221	JTHFF2C26F135BX45	2020-01-01	2019-12-15	7	4	2020-01-29	2400	NULL
221	WAUTFAFH0E0010613	2019-07-01	2019-06-12	7	1	2019-07-08	600	2019-07-01
221	WAUTFAFH0E0010613	2019-07-09	2019-06-12	1	2	2019-07-11	200	2019-07-01
221	WAUTFAFH0E0010613	2020-01-01	2019-12-15	7	4	2020-01-29	2400	NULL
221	WBA3A9G51ENN73366	2020-01-01	2019-12-15	7	4	2020-01-29	2400	NULL
221	WBA3B9C59EP458859	2020-01-01	2019-12-15	7	4	2020-01-29	2400	NULL
221	WDCGG0EB0EG188709	2020-01-01	2019-12-15	7	4	2020-01-29	2400	NULL
229	19VDE1F3XEE414842	2019-05-06	2019-04-12	1	4	2019-05-10	400	2019-05-06
229	WAUTFAFH0E0010613	2019-05-06	2019-04-12	1	4	2019-05-10	400	2019-05-06

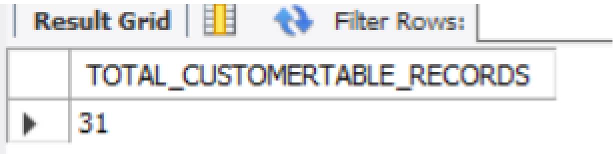
Verifying by SQL Queries

To calculate total number of records in the table:

CUSTOMER TABLE:

```
SELECT COUNT(*) AS TOTAL_CUSTOMERTABLE_RECORDS
FROM CUSTOMER;
```

RESULT:

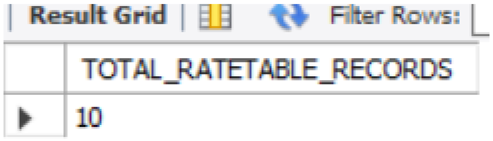


TOTAL_CUSTOMERTABLE_RECORDS
31

RATE TABLE:

```
SELECT COUNT(*) AS TOTAL_RATETABLE_RECORDS
FROM RATE;
```

RESULT:

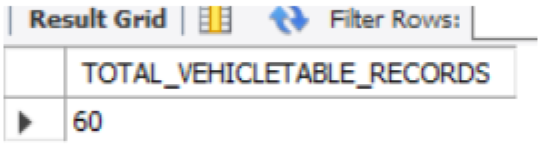


TOTAL_RATETABLE_RECORDS
10

VEHICLE TABLE:

```
SELECT COUNT(*) AS TOTAL_VEHICLETABLE_RECORDS
FROM VEHICLE;
```

RESULT:

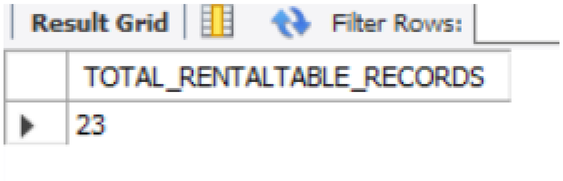


	TOTAL_VEHICLETABLE_RECORDS
▶	60

RENTAL TABLE:

```
SELECT COUNT(*) AS TOTAL_RENTALTABLE_RECORDS
FROM RENTAL;
```

RESULT:



	TOTAL_RENTALTABLE_RECORDS
▶	23

Challenges

We were trying to create tables and import files on each through SQLite terminal, but it was rather hectic to debug when something went wrong while importing. We also had to recreate the database if the data were accidentally inputted wrong. That is why we used MySQL Workbench to easily implement commands to create table and import files into table with necessary GUI provided for it. There were also cases when it was hard to retrieve the original information from database if some data was changed accidentally.

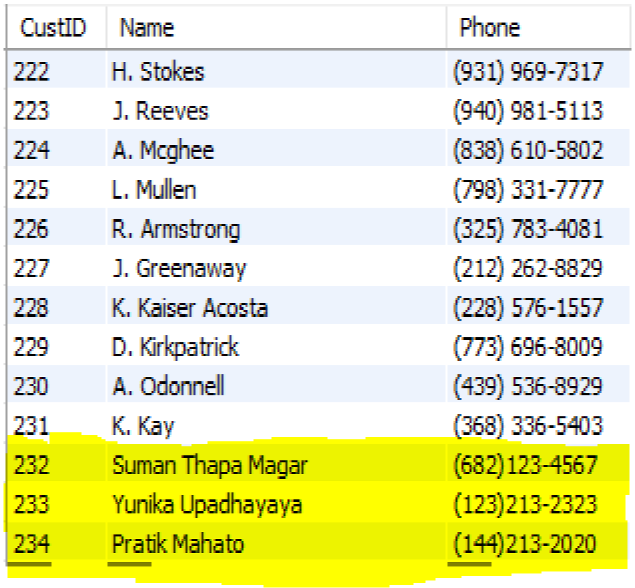
Task 3

Query 1

Insert yourself as a New Customer.Do not provide the customerID in your query.

```
INSERT INTO CUSTOMER(Name, Phone)
VALUES ('Suman Thapa Magar', '(682)123-4567');
INSERT INTO CUSTOMER (Name,Phone)
VALUES('Yunika Upadhayaya', '(123)213-2323');
INSERT INTO CUSTOMER (Name,Phone)
VALUES('Pratik Mahato', '(144)213-2020');
```

RESULT:



CustID	Name	Phone
222	H. Stokes	(931) 969-7317
223	J. Reeves	(940) 981-5113
224	A. Mcghee	(838) 610-5802
225	L. Mullen	(798) 331-7777
226	R. Armstrong	(325) 783-4081
227	J. Greenaway	(212) 262-8829
228	K. Kaiser Acosta	(228) 576-1557
229	D. Kirkpatrick	(773) 696-8009
230	A. Odonnell	(439) 536-8929
231	K. Kay	(368) 336-5403
232	Suman Thapa Magar	(682)123-4567
233	Yunika Upadhayaya	(123)213-2323
234	Pratik Mahato	(144)213-2020

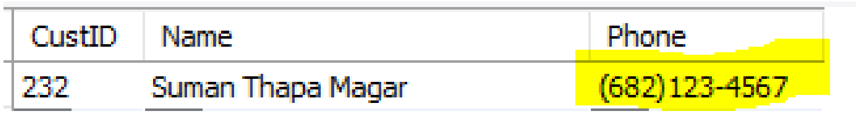
Query 2

Update your phone number to (837)721-8965

```
UPDATE CUSTOMER
SET Phone = '(837)721-8965'
WHERE CustID = 232;
```

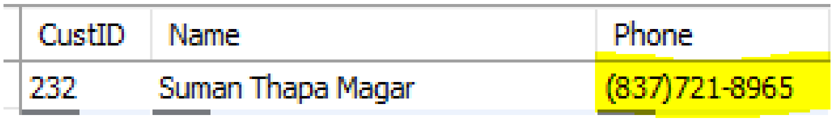
RESULT:

Before update:



CustID	Name	Phone
232	Suman Thapa Magar	(682)123-4567

After running the query:



CustID	Name	Phone
232	Suman Thapa Magar	(837)721-8965

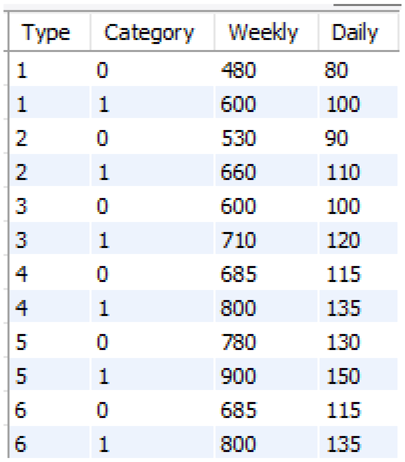
Query 3

Increase only daily rates for luxury vehicles by 5%

```
UPDATE RATE
SET Daily = Daily * 1.05
WHERE Category = '1';
```

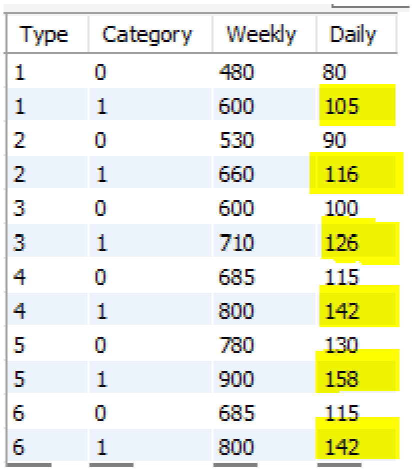
RESULT:

Before Update:



Type	Category	Weekly	Daily
1	0	480	80
1	1	600	100
2	0	530	90
2	1	660	110
3	0	600	100
3	1	710	120
4	0	685	115
4	1	800	135
5	0	780	130
5	1	900	150
6	0	685	115
6	1	800	135

After Update:



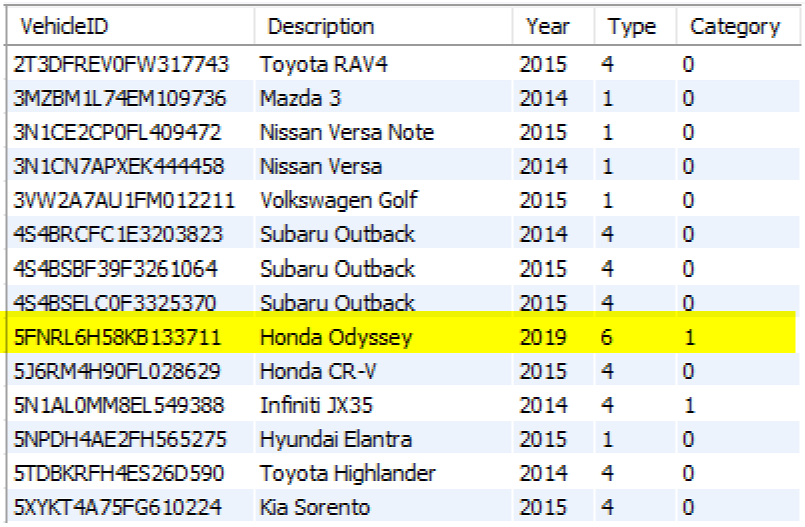
Type	Category	Weekly	Daily
1	0	480	80
1	1	600	105
2	0	530	90
2	1	660	116
3	0	600	100
3	1	710	126
4	0	685	115
4	1	800	142
5	0	780	130
5	1	900	158
6	0	685	115
6	1	800	142

Query 4a

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

```
INSERT INTO VEHICLE(VehicleID, Description, Year, Type ,Category)
VALUES ('5FNRL6H58KB133711', 'Honda Odyssey','2019', '6','1');
```

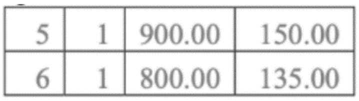
RESULT:



VehicleID	Description	Year	Type	Category
2T3DFREV0FW317743	Toyota RAV4	2015	4	0
3MZBM1L74EM109736	Mazda 3	2014	1	0
3N1CE2CP0FL409472	Nissan Versa Note	2015	1	0
3N1CN7APXEK444458	Nissan Versa	2014	1	0
3VW2A7AU1FM012211	Volkswagen Golf	2015	1	0
4S4BRCFC1E3203823	Subaru Outback	2014	4	0
4S4BSBF39F3261064	Subaru Outback	2015	4	0
4S4BSELC0F3325370	Subaru Outback	2015	4	0
5FNRL6H58KB133711	Honda Odyssey	2019	6	1
5J6RM4H90FL028629	Honda CR-V	2015	4	0
5N1AL0MM8EL549388	Infiniti JX35	2014	4	1
5NPDH4AE2FH565275	Hyundai Elantra	2015	1	0
5TDBKRFH4ES26D590	Toyota Highlander	2014	4	0
5XYKT4A75FG610224	Kia Sorento	2015	4	0

Query 4b

Insert the following rates

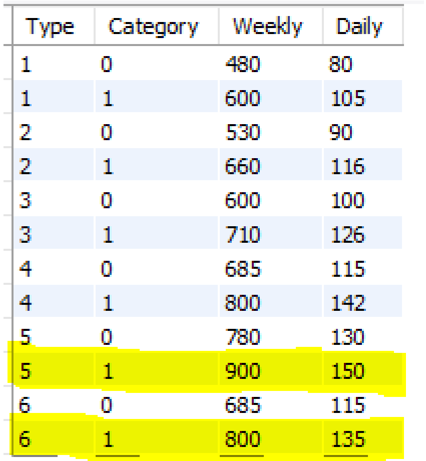


5	1	900.00	150.00
6	1	800.00	135.00

```
INSERT INTO RATE(Type, Category, Weekly, Daily)
VALUES (‘5’, ’1’, ‘900’,’150’);

INSERT INTO RATE (Type, Category, Weekly, Daily)
VALUES (‘ 6’ , ‘1’, ‘800’ , ‘135’ );
```

RESULT:



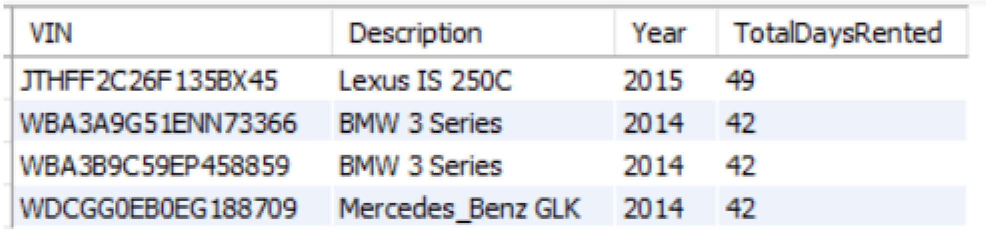
Type	Category	Weekly	Daily
1	0	480	80
1	1	600	105
2	0	530	90
2	1	660	116
3	0	600	100
3	1	710	126
4	0	685	115
4	1	800	142
5	0	780	130
5	1	900	150
6	0	685	115
6	1	800	135

Query 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.

```
SELECT R.VehicleID as VIN, V.Description, V.Year, SUM(R.Qty * R.RentalType) AS TotalDaysRented
FROM (SELECT *
            FROM VEHICLE
            WHERE Type = 1 AND Category = 1) AS V,  RENTAL AS R
WHERE V.VehicleID = R.VehicleID AND
R.VehicleID NOT IN
(
           SELECT RENTAL.VehicleID
           FROM RENTAL
           WHERE (StartDate >= '2019-06-01'AND StartDate <= '2019-06-20') OR (ReturnDate <=
           '2019-06-20' AND ReturnDate >= '2019-06-01')
)
GROUP BY  V.VehicleID;
```

RESULT:



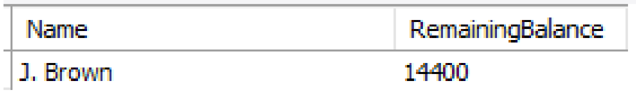
VIN	Description	Year	TotalDaysRented
JTHFF2C26F135BX45	Lexus IS 250C	2015	49
WBA3A9G51ENN73366	BMW 3 Series	2014	42
WBA3B9C59EP458859	BMW 3 Series	2014	42
WDCGG0EB0EG188709	Mercedes_Benz GLK	2014	42

Query 6

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

```
SELECT Name, SUM(TotalAmount)as RemainingBalance
FROM CUSTOMER as C, RENTAL as R
WHERE C.CustID =R.CustID AND C.CustID = '221' AND  R.PaymentDate IS NULL;
```

RESULT:



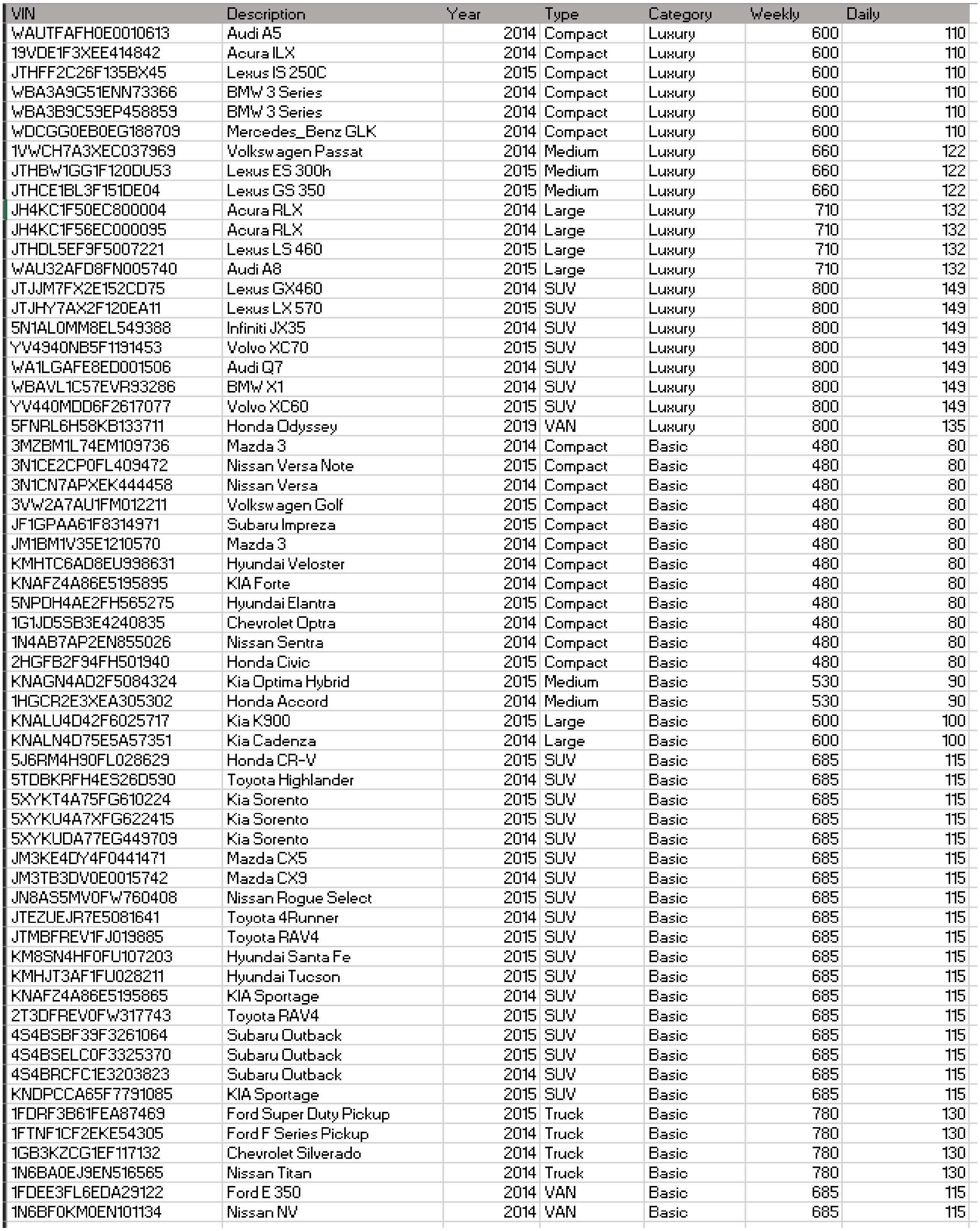
Name	RemainingBalance
J. Brown	14400

Query 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.

```
 SELECT V.VehicleID as VIN , V.Description, V.Year,
CASE V.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 Type ,
CASE V.Category
        WHEN 0 THEN 'Basic'
        WHEN 1 THEN 'Luxury'
END Category,  R.Weekly, R.Daily
FROM RATE AS R
INNER JOIN VEHICLE AS V
ON R.Type = V.Type AND R.CATEGORY = V.Category
ORDER BY V.Category DESC, V.Type;
```

RESULT:



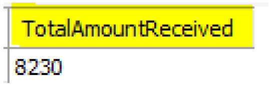
VIN	Description	Year	Type	Category	Weekly	Daily
WAUTFAFH0E0010613	Audi A5	2014	Compact	Luxury	600	110
19VDE1F3XEE414842	Acura ILX	2014	Compact	Luxury	600	110
JTHFF2C26F135BX45	Lexus IS 250C	2015	Compact	Luxury	600	110
WBA3A9G51ENN73366	BMW 3 Series	2014	Compact	Luxury	600	110
WBA3B9C59EP458859	BMW 3 Series	2014	Compact	Luxury	600	110
WDCGG0EB0EG188709	Mercedes_Benz GLK	2014	Compact	Luxury	600	110
1VWCH7A3XEC037969	Volksw agen Passat	2014	Medium	Luxury	660	122
JTHBW1GG1F120DU53	Lexus ES 300h	2015	Medium	Luxury	660	122
JTHCE1BL3F151DE04	Lexus GS 350	2015	Medium	Luxury	660	122
JH4KC1F50EC800004	Acura RLX	2014	Large	Luxury	710	132
JH4KC1F56EC000095	Acura RLX	2014	Large	Luxury	710	132
JTHDL5EF9F5007221	Lexus LS 460	2015	Large	Luxury	710	132
WAU32AFD8FN005740	Audi A8	2015	Large	Luxury	710	132
JTJJM7FX2E152CD75	Lexus GX460	2014	SUV	Luxury	800	149
JTJHY7AX2F120EA11	Lexus LX 570	2015	SUV	Luxury	800	149
5N1AL0MM8EL549388	Infiniti JX35	2014	SUV	Luxury	800	149
YV4940NB5F1191453	Volvo XC70	2015	SUV	Luxury	800	149
WA1LGAFE8ED001506	Audi Q7	2014	SUV	Luxury	800	149
WBAVL1C57EVR93286	BMW X1	2014	SUV	Luxury	800	149
YV440MDD6F2617077	Volvo XC60	2015	SUV	Luxury	800	149
5FNRL6H58KB133711	Honda Odyssey	2019	VAN	Luxury	800	135
3MZBM1L74EM109736	Mazda 3	2014	Compact	Basic	480	80
3N1CE2CP0FL409472	Nissan Versa Note	2015	Compact	Basic	480	80
3N1CN7APXEK444458	Nissan Versa	2014	Compact	Basic	480	80
3VW2A7AU1FM012211	Volksw agen Golf	2015	Compact	Basic	480	80
JF1GPAA61F8314971	Subaru Impreza	2015	Compact	Basic	480	80
JM1BM1V35E1210570	Mazda 3	2014	Compact	Basic	480	80
KMHTC6AD8EU998631	Hyundai Veloster	2014	Compact	Basic	480	80
KNAFZ4A86E5195895	KIA Forte	2014	Compact	Basic	480	80
5NPDH4AE2FH565275	Hyundai Elantra	2015	Compact	Basic	480	80
1G1JD5SB3E4240835	Chevrolet Optra	2014	Compact	Basic	480	80
1N4AB7AP2EN855026	Nissan Sentra	2014	Compact	Basic	480	80
2HGFB2F94FH501940	Honda Civic	2015	Compact	Basic	480	80
KNAGN4AD2F5084324	Kia Optima Hybrid	2015	Medium	Basic	530	90
1HGCR2E3XEA305302	Honda Accord	2014	Medium	Basic	530	90
KNALU4D42F6025717	Kia K900	2015	Large	Basic	600	100
KNALN4D75E5A57351	Kia Cadenza	2014	Large	Basic	600	100
5J6RM4H90FL028629	Honda CR-V	2015	SUV	Basic	685	115
5TDBKRFH4ES26D590	Toyota Highlander	2014	SUV	Basic	685	115
5XYKT4A75FG610224	Kia Sorento	2015	SUV	Basic	685	115
5XYKU4A7XFG622415	Kia Sorento	2015	SUV	Basic	685	115
5XYKUDA77EG449709	Kia Sorento	2014	SUV	Basic	685	115
JM3KE4DY4F0441471	Mazda CX5	2015	SUV	Basic	685	115
JM3TB3DV0E0015742	Mazda CX9	2014	SUV	Basic	685	115
JN8AS5MV0FW760408	Nissan Rogue Select	2015	SUV	Basic	685	115
JTEZUEJR7E5081641	Toyota 4Runner	2014	SUV	Basic	685	115
JTMBFREV1FJ019885	Toyota RAV4	2015	SUV	Basic	685	115
KM8SN4HF0FU107203	Hyundai Santa Fe	2015	SUV	Basic	685	115
KMHJT3AF1FU028211	Hyundai Tucson	2015	SUV	Basic	685	115
KNAFZ4A86E5195865	KIA Sportage	2014	SUV	Basic	685	115
2T3DFREV0FW317743	Toyota RAV4	2015	SUV	Basic	685	115
4S4BSBF39F3261064	Subaru Outback	2015	SUV	Basic	685	115
4S4BSELC0F3325370	Subaru Outback	2015	SUV	Basic	685	115
4S4BRCFC1E3203823	Subaru Outback	2014	SUV	Basic	685	115
KNDPCCA65F7791085	KIA Sportage	2015	SUV	Basic	685	115
1FDRF3B61FEA87469	Ford Super Duty Pickup	2015	Truck	Basic	780	130
1FTNF1CF2EKE54305	Ford F Series Pickup	2014	Truck	Basic	780	130
1GB3KZCG1EF117132	Chevrolet Silverado	2014	Truck	Basic	780	130
1N6BA0EJ9EN516565	Nissan Titan	2014	Truck	Basic	780	130
1FDEE3FL6EDA29122	Ford E 350	2014	VAN	Basic	685	115
1N6BF0KM0EN101134	Nissan NV	2014	VAN	Basic	685	115

Query 8

What is the total of money that customers paid to us until today?

```
SELECT SUM(TotalAmount) as TotalAmountReceived
FROM RENTAL
WHERE RENTAL.PAYMENTDATE IS NOT NULL;
```

<u>RESULT:</u>



```
TotalAmountReceived
8230
```

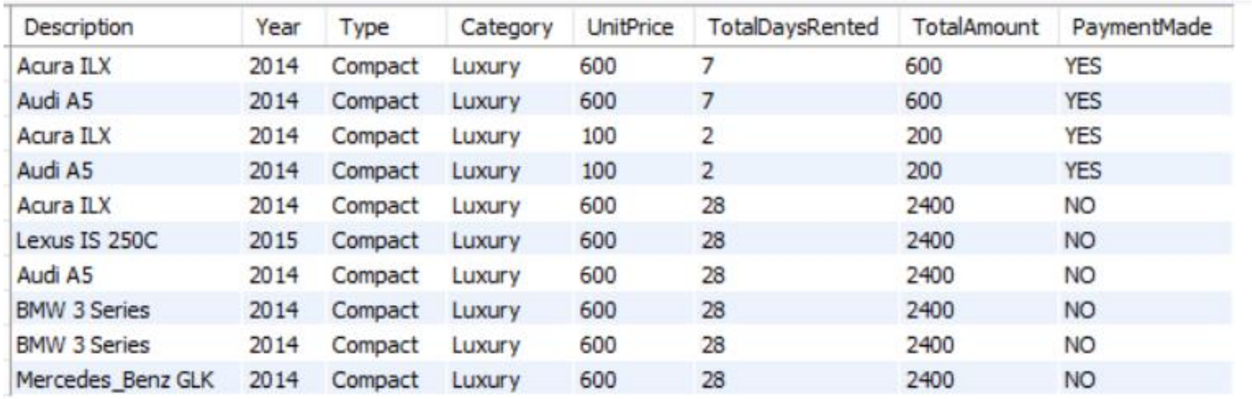
Query 9a

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.

```
SELECT V.Description, V.Year,
CASE V.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 Type ,
CASE V.Category
   WHEN 0 THEN 'Basic'
   WHEN 1 THEN 'Luxury'
END Category,
R.TotalAmount DIV R.Qty, (R.Qty * R.RentalType) AS TotalDaysRented,
R.TotalAmount,
CASE
    WHEN R.PaymentDate IS NULL THEN 'NO'
   ELSE 'YES'
END AS PaymentMade
```

```
FROM RENTAL as R, CUSTOMER as C, VEHICLE AS V
WHERE C.Name = 'J. Brown' AND C.CustID = R.CustID AND R.VehicleID = V.VehicleID
ORDER BY R.StartDate;
s
```

RESULT:



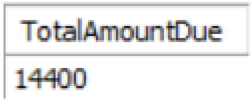
Description	Year	Type	Category	UnitPrice	TotalDaysRented	TotalAmount	PaymentMade
Acura ILX	2014	Compact	Luxury	600	7	600	YES
Audi A5	2014	Compact	Luxury	600	7	600	YES
Acura ILX	2014	Compact	Luxury	100	2	200	YES
Audi A5	2014	Compact	Luxury	100	2	200	YES
Acura ILX	2014	Compact	Luxury	600	28	2400	NO
Lexus IS 250C	2015	Compact	Luxury	600	28	2400	NO
Audi A5	2014	Compact	Luxury	600	28	2400	NO
BMW 3 Series	2014	Compact	Luxury	600	28	2400	NO
BMW 3 Series	2014	Compact	Luxury	600	28	2400	NO
Mercedes_Benz GLK	2014	Compact	Luxury	600	28	2400	NO

Query 9b

For the same customer return the current balance.

```
SELECT SUM(R.TotalAmount) AS TotalAmountDue
FROM RENTAL AS R
WHERE R.PaymentDate IS NULL AND
R.CustID IN
(SELECT C.CustID
FROM CUSTOMER as C
WHERE C.Name = 'J. Brown');
```

RESULT:



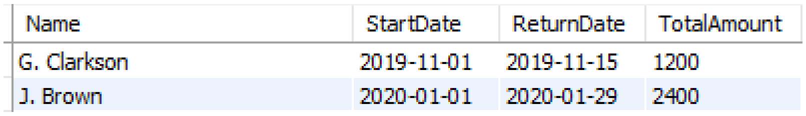
TotalAmountDue
14400

Query 10

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

```
SELECT C.Name, R.StartDate, R.ReturnDate, R.TotalAmount
FROM RENTAL AS R, CUSTOMER AS C
WHERE R.CustID = C.CustID AND R.VehicleID = '19VDE1F3XEE414842' AND R.PaymentDate IS
NULL ;
```

RESULT:



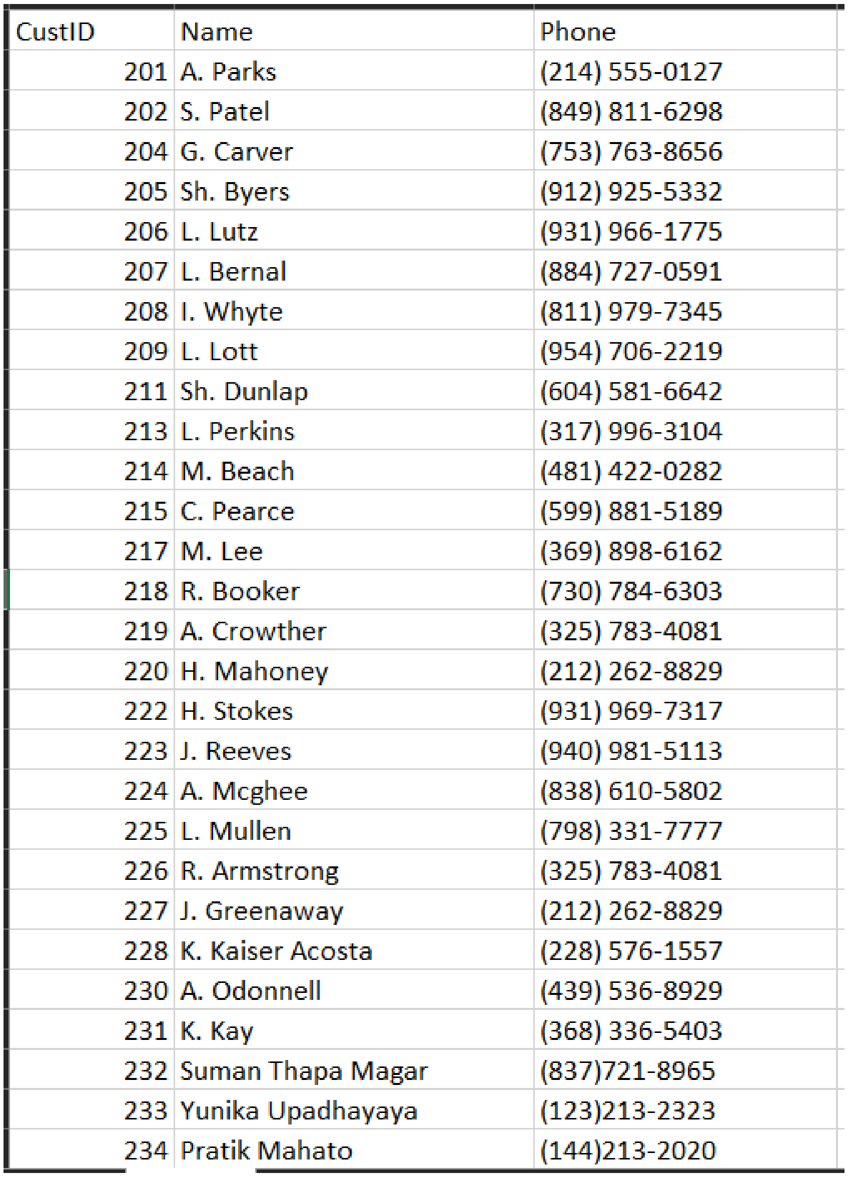
Name	StartDate	ReturnDate	TotalAmount
G. Clarkson	2019-11-01	2019-11-15	1200
J. Brown	2020-01-01	2020-01-29	2400

Query 11

Return all customers that they never rent a vehicle.

```
SELECT *
FROM CUSTOMER
WHERE CustID NOT IN (SELECT CustID FROM RENTAL);
```

RESULT:



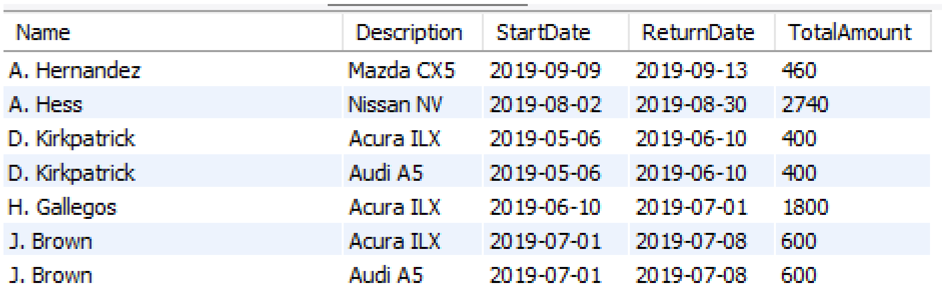
CustID	Name	Phone
201	A. Parks	(214) 555-0127
202	S. Patel	(849) 811-6298
204	G. Carver	(753) 763-8656
205	Sh. Byers	(912) 925-5332
206	L. Lutz	(931) 966-1775
207	L. Bernal	(884) 727-0591
208	I. Whyte	(811) 979-7345
209	L. Lott	(954) 706-2219
211	Sh. Dunlap	(604) 581-6642
213	L. Perkins	(317) 996-3104
214	M. Beach	(481) 422-0282
215	C. Pearce	(599) 881-5189
217	M. Lee	(369) 898-6162
218	R. Booker	(730) 784-6303
219	A. Crowther	(325) 783-4081
220	H. Mahoney	(212) 262-8829
222	H. Stokes	(931) 969-7317
223	J. Reeves	(940) 981-5113
224	A. Mcghee	(838) 610-5802
225	L. Mullen	(798) 331-7777
226	R. Armstrong	(325) 783-4081
227	J. Greenaway	(212) 262-8829
228	K. Kaiser Acosta	(228) 576-1557
230	A. Odonnell	(439) 536-8929
231	K. Kay	(368) 336-5403
232	Suman Thapa Magar	(837)721-8965
233	Yunika Upadhayaya	(123)213-2323
234	Pratik Mahato	(144)213-2020

Query 12

Return all rentals that the customer paid on the StartDate. List Customer Name, Vehicle Descriptions, StartDate, ReturnDate, and TotalAmount. Order by Customer Name.

```
SELECT C.Name, V.Description, R.StartDate, R.ReturnDate, R.TotalAmount
FROM CUSTOMER AS C, RENTAL AS R, VEHICLE AS V
WHERE R.StartDate = R.PaymentDate AND R.CustID = C.CustID AND R.VehicleID = V.VehicleID
ORDER BY C.Name;
```

RESULT:



Name	Description	StartDate	ReturnDate	TotalAmount
A. Hernandez	Mazda CX5	2019-09-09	2019-09-13	460
A. Hess	Nissan NV	2019-08-02	2019-08-30	2740
D. Kirkpatrick	Acura ILX	2019-05-06	2019-06-10	400
D. Kirkpatrick	Audi A5	2019-05-06	2019-06-10	400
H. Gallegos	Acura ILX	2019-06-10	2019-07-01	1800
J. Brown	Acura ILX	2019-07-01	2019-07-08	600
J. Brown	Audi A5	2019-07-01	2019-07-08	600

Challenges

The most significant challenge in the Task 3 was figuring out necessary constraints and the variable declaration needed to run the queries. We had to keep updating tables and importing files if constraint did not work as planned. Out of all queries to made, we would say query no. 5 and query no.9a took lot of time to sort out the answer and debugging. It was also equally crucial to know which attribute should be made foreign key to access primary keys effectively.

References

Elmasri, R., & Navathe, S. (2007). Fundamentals of database systems. Boston: Pearson/Addison Wesley.