Project 2 (Part – 2)

Task 1: Create the following 4 tables for the Car Rental Database

The four tables built for the car rental database are as follows:

CUSTOMER, RENTAL, VEHICLE & RATE.

```
CREATE TABLE VEHICLE
(
   VehicleID  TEXT  NOT NULL,
   Description  TEXT,
   Year  INT,
   Type  INT,
   Category  INT,
   PRIMARY KEY(VehicleID)
);

CREATE TABLE RATE
(
   Type  INT,
   Category  INT,
   Weekly  INT,
   Daily  INT
);

CREATE TABLE CUSTOMER
(
   CustID INTEGER NOT NULL,
   Name  TEXT,
   Phone  TEXT,
```

```
    PRIMARY KEY(CustID)
);
CREATE TABLE RENTAL
(
    CustID INTEGER NOT NULL,
    VehicleID  TEXT  NOT NULL,
    StartDate  DATE,
    OrderDate DATE,
    RentalType  INT,
    Qty  INT,
    ReturnDate  DATE,
    TotalAmount  INT,
    PaymentDate  DATE,
    FOREIGN KEY(CustID) REFERENCES CUSTOMER(CustID),
    FOREIGN KEY(VehicleID) REFERENCES VEHICLE(VehicleID)
);
```

We have decided to create tables for all entities to relate to each other by mapping common values through tables. We decided to have CustID as a primary key for the Customer table, VehicleID as a primary key for Vehicle table, and CustID and VehicleID would be foreign keys in the rental relation. One challenge that we faced while creating the tables was we wanted to create a composite primary key for Rental by having (CustID, VehicleID); however, we were getting errors and that data wasn't getting loaded because of a constraint issue, so we decided to not have a primary key for the rental relation.

Task 2:

The data was stored in .csv files and to import the files we use commands .import table_name to get the data stored in the car rental database.

If we didn't have data stored in .csv files then we would have to create a new table by CREATE TABLE_NAME command and then populate data into the database using the INSERT INTO TABLE_NAME command.

```
.mode csv
```

.import RATE.csv RATE

.import RENTAL.csv RENTAL

.import VEHICLE.csv VEHICLE

.import CUSTOMER.csv CUSTOMER

To check if the data is properly populated we use the command .schema to check the tables that were imported in the proper format and to check if all the data is transferred we can check with printing all the data from the table by putting the command SELECT * FROM TABLE_NAME;

Query to calculate the total number of records per table:

SELECT COUNT (*) AS COUNT FROM VEHICLE;

COUNT: 60

SELECT COUNT (*) AS COUNT FROM RATE;

COUNT: 10

SELECT COUNT (*) AS COUNT FROM CUSTOMER;

COUNT: 32

SELECT COUNT (*) AS COUNT FROM RENTAL;

COUNT: 24

Task 3:

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

INSERT INTO CUSTOMER (CustID, Name, Phone)

VALUES (NULL, 'A. Sakallah', '1001846204');

```
Question 1:
CustID  Name         Phone
------  -----------  ----------
232     A. Sakallah  1001846204
```

Number of rows affected: 1

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

UPDATE CUSTOMER

SET PHONE = '(837) 721-8965'

WHERE NAME = 'A. Sakallah';

```
Question 2:
CustID  Name         Phone
------  -----------  --------------
232     A. Sakallah  (837) 721-8965
```

Number of rows affected: 1

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

UPDATE RATE

SET DAILY = DAILY*0.05 + DAILY

WHERE CATEGORY = 1;

```
Question 3:
Type  Category  Weekly  Daily
----  --------  ------  ------
1     0         480     80
1     1         600     105
2     0         530     90
2     1         660     115.5
3     0         600     100
3     1         710     126
4     0         685     115
4     1         800     141.75
5     0         780     130
6     0         685     115
```

Number of rows affected: 10

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

INSERT INTO VEHICLE

VALUES('5FNRL6H58KB133711' , 'Honda Odyssey' , 2019, 6, 1);

```
Question 4a:
VehicleID          Description    Year  Type  Category
```

```
-----------------  -------------  ----  ----  --------
5FNRL6H58KB133711  Honda Odyssey  2019  1     6
```

Number of rows affected: 1

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

INSERT INTO RATE

VALUES

(5, 1, 900.00, 150.00),

(6, 1, 800.00, 135.00);

```
Question 4b:
Type  Category  Weekly  Daily
----  --------  ------  ------
1     0         480     80
1     1         600     105
2     0         530     90
2     1         660     115.5
3     0         600     100
3     1         710     126
4     0         685     115
4     1         800     141.75
5     0         780     130
6     0         685     115
5     1         900     150
6     1         800     135
```

Number of rows affected: 12

Question 5: Return all Compact(1) & Luxury(1) vehicles that were available for rent from June 01, 2019, until June 20, 2019. List VehicleID as VIN, Description, year, and how many days have been rented so far. You need to change the weeks into days.

SELECT V.VEHICLEID AS 'VIN', V.DESCRIPTION, V.YEAR, SUM(julianday(returndate) - julianday(startdate)) AS 'Rental Days'

FROM VEHICLE AS V JOIN RENTAL AS R ON V.VEHICLEID = R.VEHICLEID

WHERE V.TYPE = 1 AND CATEGORY = 1 AND R.ORDERDATE AND

(SELECT R1.VEHICLEID

FROM RENTAL AS R1

WHERE StartDate BETWEEN '2019-06-01' AND '2019-06-20')

GROUP BY V.VEHICLEID;

```
Question 5:
VIN                Description         Year  Rental Days
-----------------  ------------------  ----  -----------
19VDE1F3XEE414842  Acura ILX           2014  76.0
JTHFF2C26F135BX45  Lexus IS 250C       2015  49.0
WAUTFAFH0E0010613  Audi A5             2014  55.0
WBA3A9G51ENN73366  BMW 3 Series        2014  42.0
WBA3B9C59EP458859  BMW 3 Series        2014  42.0
WDCGG0EB0EG188709  Mercedes_Benz GLK   2014  42.0
```

Number of rows affected: 6

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

SELECT C.NAME, SUM(TOTALAMOUNT) AS 'Remaining Balance'

FROM CUSTOMER AS C

JOIN RENTAL AS R ON C.CUSTID = R.CUSTID

WHERE R.PaymentDate = 'NULL' AND R.CUSTID = 221

GROUP BY C.NAME;

```
Question 6:
Name      Remaining Balance
--------  -----------------
J. Brown  14400
```

Number of rows affected: 1

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.

SELECT DISTINCT V.VEHICLEID AS VIN, V.DESCRIPTION, V.YEAR, V.TYPE, V.CATEGORY, R.DAILY, R.WEEKLY,

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 AS MODEL,
CASE V.CATEGORY
WHEN 1 THEN 'LUXURY'
ELSE 'BASIC'
END AS LB
FROM VEHICLE AS V, RATE AS R
GROUP BY V.VEHICLEID
ORDER BY V.CATEGORY DESC;
```

Question 7:

VIN	Description	Year	Type	Category	Daily	Weekly	MODEL	LB
19VDE1F3XEE414842	Acura ILX	2014	1	1	80	480	COMPACT	LUXURY
1VWCH7A3XEC037969	Volkswagen Passat	2014	2	1	80	480	MEDIUM	LUXURY
5FNRL6H58KB133711	Honda Odyssey	2019	6	1	80	480	VAN	LUXURY
5N1AL0MM8EL549388	Infiniti JX35	2014	4	1	80	480	SUV	LUXURY
JH4KC1F50EC800004	Acura RLX	2014	3	1	80	480	LARGE	LUXURY
JH4KC1F56EC000095	Acura RLX	2014	3	1	80	480	LARGE	LUXURY
JTHBW1GG1F120DU53	Lexus ES 300h	2015	2	1	80	480	MEDIUM	LUXURY
JTHCE1BL3F151DE04	Lexus GS 350	2015	2	1	80	480	MEDIUM	LUXURY
JTHDL5EF9F5007221	Lexus LS 460	2015	3	1	80	480	LARGE	LUXURY
JTHFF2C26F135BX45	Lexus IS 250C	2015	1	1	80	480	COMPACT	LUXURY

JTJHY7AX2F120EA11 Lexus LX 570 2015 4 1 80
480 SUV LUXURY
JTJJM7FX2E152CD75 Lexus GX460 2014 4 1 80
480 SUV LUXURY
WA1LGAFE8ED001506 Audi Q7 2014 4 1 80
480 SUV LUXURY
WAU32AFD8FN005740 Audi A8 2015 3 1 80
480 LARGE LUXURY
WAUTFAFH0E0010613 Audi A5 2014 1 1 80
480 COMPACT LUXURY
WBA3A9G51ENN73366 BMW 3 Series 2014 1 1 80
480 COMPACT LUXURY
WBA3B9C59EP458859 BMW 3 Series 2014 1 1 80
480 COMPACT LUXURY
WBAVL1C57EVR93286 BMW X1 2014 4 1 80
480 SUV LUXURY
WDCGG0EB0EG188709 Mercedes_Benz GLK 2014 1 1 80
480 COMPACT LUXURY
YV440MDD6F2617077 Volvo XC60 2015 4 1 80
480 SUV LUXURY
YV4940NB5F1191453 Volvo XC70 2015 4 1 80
480 SUV LUXURY
1FDEE3FL6EDA29122 Ford E 350 2014 6 0 80
480 VAN BASIC
1FDRF3B61FEA87469 Ford Super Duty Pickup 2015 5 0 80
480 TRUCK BASIC
1FTNF1CF2EKE54305 Ford F Series Pickup 2014 5 0 80
480 TRUCK BASIC
1G1JD5SB3E4240835 Chevrolet Optra 2014 1 0 80
480 COMPACT BASIC
1GB3KZCG1EF117132 Chevrolet Silverado 2014 5 0 80
480 TRUCK BASIC
1HGCR2E3XEA305302 Honda Accord 2014 2 0 80
480 MEDIUM BASIC
1N4AB7AP2EN855026 Nissan Sentra 2014 1 0 80
480 COMPACT BASIC
1N6BA0EJ9EN516565 Nissan Titan 2014 5 0 80
480 TRUCK BASIC
1N6BF0KM0EN101134 Nissan NV 2014 6 0 80
480 VAN BASIC
2HGFB2F94FH501940 Honda Civic 2015 1 0 80
480 COMPACT BASIC
2T3DFREV0FW317743 Toyota RAV4 2015 4 0 80
480 SUV BASIC
3MZBM1L74EM109736 Mazda 3 2014 1 0 80
480 COMPACT BASIC
3N1CE2CP0FL409472 Nissan Versa Note 2015 1 0 80
480 COMPACT BASIC
3N1CN7APXEK444458 Nissan Versa 2014 1 0 80
480 COMPACT BASIC
3VW2A7AU1FM012211 Volkswagen Golf 2015 1 0 80
480 COMPACT BASIC

4S4BRCFC1E3203823 Subaru Outback 2014 4 0 80
480 SUV BASIC
4S4BSBF39F3261064 Subaru Outback 2015 4 0 80
480 SUV BASIC
4S4BSELC0F3325370 Subaru Outback 2015 4 0 80
480 SUV BASIC
5J6RM4H90FL028629 Honda CR-V 2015 4 0 80
480 SUV BASIC
5NPDH4AE2FH565275 Hyundai Elantra 2015 1 0 80
480 COMPACT BASIC
5TDBKRFH4ES26D590 Toyota Highlander 2014 4 0 80
480 SUV BASIC
5XYKT4A75FG610224 Kia Sorento 2015 4 0 80
480 SUV BASIC
5XYKU4A7XFG622415 Kia Sorento 2015 4 0 80
480 SUV BASIC
5XYKUDA77EG449709 Kia Sorento 2014 4 0 80
480 SUV BASIC
JF1GPAA61F8314971 Subaru Impreza 2015 1 0 80
480 COMPACT BASIC
JM1BM1V35E1210570 Mazda 3 2014 1 0 80
480 COMPACT BASIC
JM3KE4DY4F0441471 Mazda CX5 2015 4 0 80
480 SUV BASIC
JM3TB3DV0E0015742 Mazda CX9 2014 4 0 80
480 SUV BASIC
JN8AS5MV0FW760408 Nissan Rogue Select 2015 4 0 80
480 SUV BASIC
JTEZUEJR7E5081641 Toyota 4Runner 2014 4 0 80
480 SUV BASIC
JTMBFREV1FJ019885 Toyota RAV4 2015 4 0 80
480 SUV BASIC
KM8SN4HF0FU107203 Hyundai Santa Fe 2015 4 0 80
480 SUV BASIC
KMHJT3AF1FU028211 Hyundai Tucson 2015 4 0 80
480 SUV BASIC
KMHTC6AD8EU998631 Hyundai Veloster 2014 1 0 80
480 COMPACT BASIC
KNAFZ4A86E5195865 KIA Sportage 2014 4 0 80
480 SUV BASIC
KNAFZ4A86E5195895 KIA Forte 2014 1 0 80
480 COMPACT BASIC
KNAGN4AD2F5084324 Kia Optima Hybrid 2015 2 0 80
480 MEDIUM BASIC
KNALN4D75E5A57351 Kia Cadenza 2014 3 0 80
480 LARGE BASIC
KNALU4D42F6025717 Kia K900 2015 3 0 80
480 LARGE BASIC
KNDPCCA65F7791085 KIA Sportage 2015 4 0 80
480 SUV BASIC

Number of rows affected: 61

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

SELECT SUM(TOTALAMOUNT) AS TOTAL_MONEY FROM RENTAL;

```
Question 8:
TOTAL_MONEY
-----------
29830.0
```

Number of rows affected: 1

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.

SELECT DISTINCT C.NAME, V.DESCRIPTION, V.YEAR, V.TYPE, V.CATEGORY, R.TOTALAMOUNT/(julianday(RETURNDATE) - julianday(STARTDATE)) AS UnitPrice, R.STARTDATE, R.RETURNDATE, R.PAYMENTDATE,

CASE RA.TYPE

WHEN RA.TYPE = 1 THEN 'DAILY'

WHEN RA.TYPE = 7 THEN 'WEEKLY'

ELSE 'WEEKLY' END AS DURATION

FROM VEHICLE AS V, CUSTOMER AS C, RENTAL AS R, RATE AS RA

WHERE C.CUSTID = R.CUSTID AND R.VEHICLEID = V.VEHICLEID AND C.NAME = 'J. Brown'

ORDER BY R.STARTDATE;

```
Question 9a:
Name      Description         Year  Type  Category  UnitPrice
StartDate   ReturnDate  PaymentDate  DURATION
--------  -----------------  ----  ----  --------  ----------------  -
---------  ----------  -----------  --------
J. Brown  Acura ILX          2014  1     1         85.7142857142857
2019-07-01  2019-07-08  2019-07-01   DAILY
```

```
J. Brown  Audi A5            2014  1     1         85.7142857142857
2019-07-01  2019-07-08  2019-07-01   DAILY
J. Brown  Acura ILX          2014  1     1         85.7142857142857
2019-07-01  2019-07-08  2019-07-01   WEEKLY
J. Brown  Audi A5            2014  1     1         85.7142857142857
2019-07-01  2019-07-08  2019-07-01   WEEKLY
J. Brown  Acura ILX          2014  1     1         100.0
2019-07-09  2019-07-11  2019-07-01   DAILY
J. Brown  Audi A5            2014  1     1         100.0
2019-07-09  2019-07-11  2019-07-01   DAILY
J. Brown  Acura ILX          2014  1     1         100.0
2019-07-09  2019-07-11  2019-07-01   WEEKLY
J. Brown  Audi A5            2014  1     1         100.0
2019-07-09  2019-07-11  2019-07-01   WEEKLY
J. Brown  Acura ILX          2014  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         DAILY
J. Brown  Lexus IS 250C      2015  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         DAILY
J. Brown  Audi A5            2014  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         DAILY
J. Brown  BMW 3 Series       2014  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         DAILY
J. Brown  Mercedes_Benz GLK  2014  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         DAILY
J. Brown  Acura ILX          2014  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         WEEKLY
J. Brown  Lexus IS 250C      2015  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         WEEKLY
J. Brown  Audi A5            2014  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         WEEKLY
J. Brown  BMW 3 Series       2014  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         WEEKLY
J. Brown  Mercedes_Benz GLK  2014  1     1         85.7142857142857
2020-01-01  2020-01-29  NULL         WEEKLY
```

Number of rows affected: 18

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

```
SELECT C.NAME, SUM(TOTALAMOUNT) AS CURRENT_BALANCE
FROM CUSTOMER AS C
JOIN RENTAL AS R ON C.CUSTID = R.CUSTID
WHERE R.PAYMENTDATE = 'NULL' AND C.NAME = 'J. Brown';
```

```
Question 9b:
```

```
Name       CURRENT_BALANCE
--------   ---------------
J. Brown   14400
```

Number of rows affected: 1

Question 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 CUSTOMER AS C JOIN RENTAL AS R ON C.CUSTID = R.CUSTID
WHERE R.VEHICLEID = '19VDE1F3XEE414842' AND R.PAYMENTDATE = 'NULL';
```

```
Question 10:
Name          StartDate   ReturnDate  TotalAmount
-----------   ----------  ----------  -----------
G. Clarkson   2019-11-01  2019-11-15  1200
J. Brown      2020-01-01  2020-01-29  2400
```

Number of rows affected: 2

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

```
SELECT C.NAME
FROM CUSTOMER AS C
WHERE NOT EXISTS (SELECT *
FROM RENTAL AS R
WHERE C.CustID= R.CustID);
```

```
Question 11:
Name
---------------
A. Parks
S. Patel
G. Carver
Sh. Byers
L. Lutz
L. Bernal
I. Whyte
L. Lott
Sh. Dunlap
```

```
L. Perkins
M. Beach
C. Pearce
M. Lee
R. Booker
A. Crowther
H. Mahoney
H. Stokes
J. Reeves
A. Mcghee
L. Mullen
R. Armstrong
J. Greenaway
K. Kaiser Acosta
A. Odonnell
K. Kay
A. Sakallah
```

Number of rows affected: 26

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.

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

```
Question 12:
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  2019-05-10  400
D. Kirkpatrick  Audi A5      2019-05-06  2019-05-10  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
```

Number of rows affected: 7