DATABASE SYSTEMS- ASSIGNMNET 2- PART3

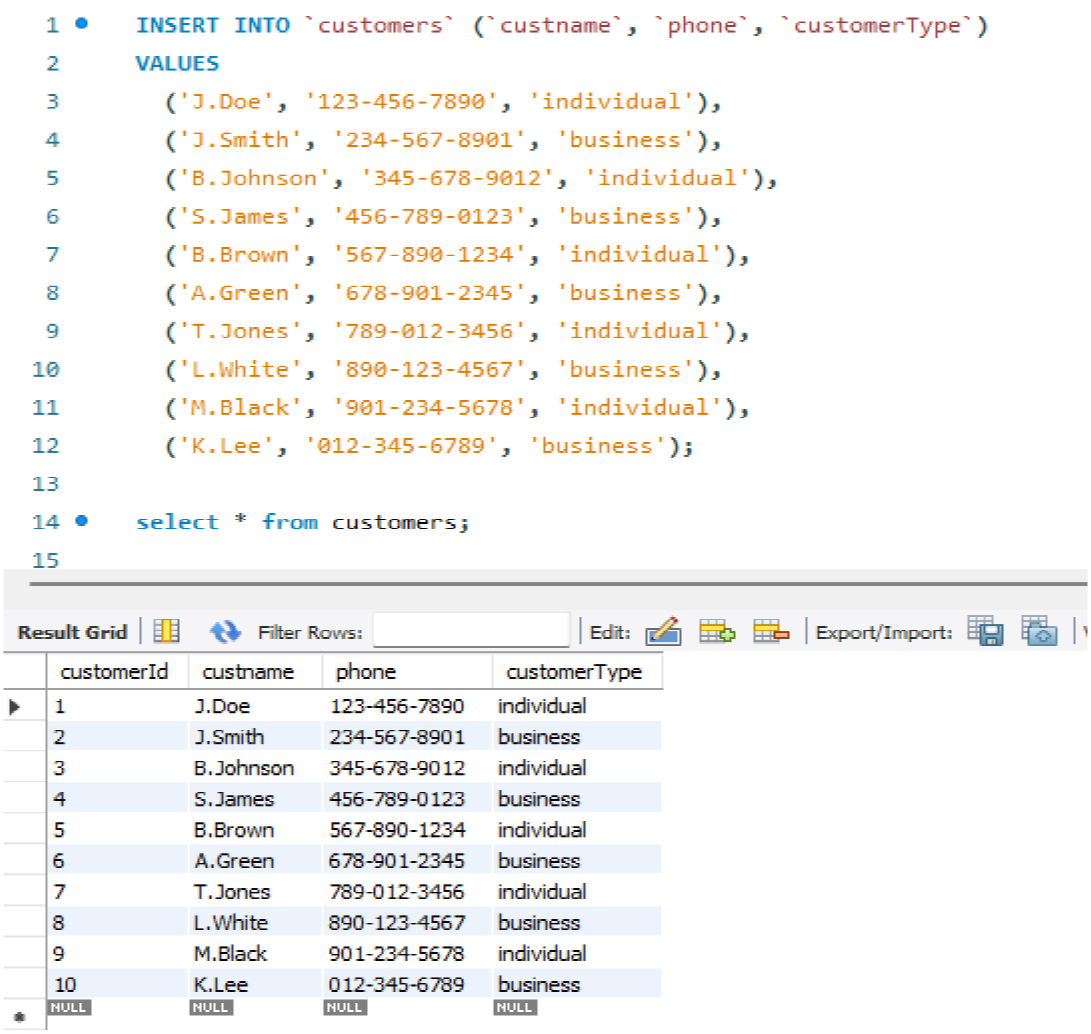
ARCHI SAHU 1002104548

Question1 and Question2:

INSERTING DATA, WRITING QUERIES and RETRIEVING THE OUTPUT

Customer table:

Insert query:

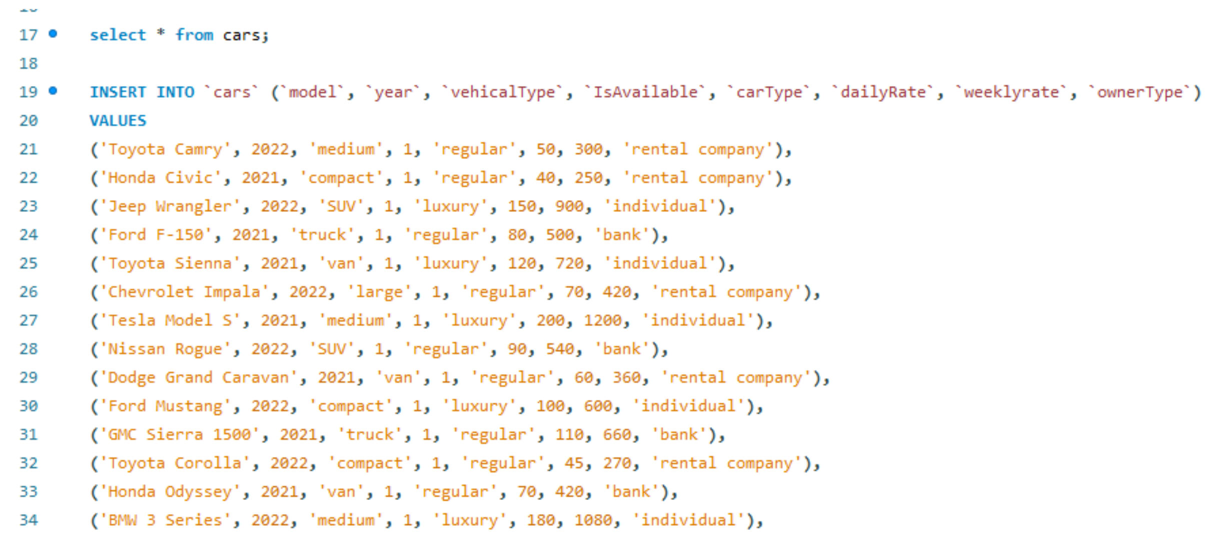


```sql
INSERT INTO `customers` (`custname`, `phone`, `customerType`)
VALUES
  ('J.Doe', '123-456-7890', 'individual'),
  ('J.Smith', '234-567-8901', 'business'),
  ('B.Johnson', '345-678-9012', 'individual'),
  ('S.James', '456-789-0123', 'business'),
  ('B.Brown', '567-890-1234', 'individual'),
  ('A.Green', '678-901-2345', 'business'),
  ('T.Jones', '789-012-3456', 'individual'),
  ('L.White', '890-123-4567', 'business'),
  ('M.Black', '901-234-5678', 'individual'),
  ('K.Lee', '012-345-6789', 'business');

select * from customers;
```

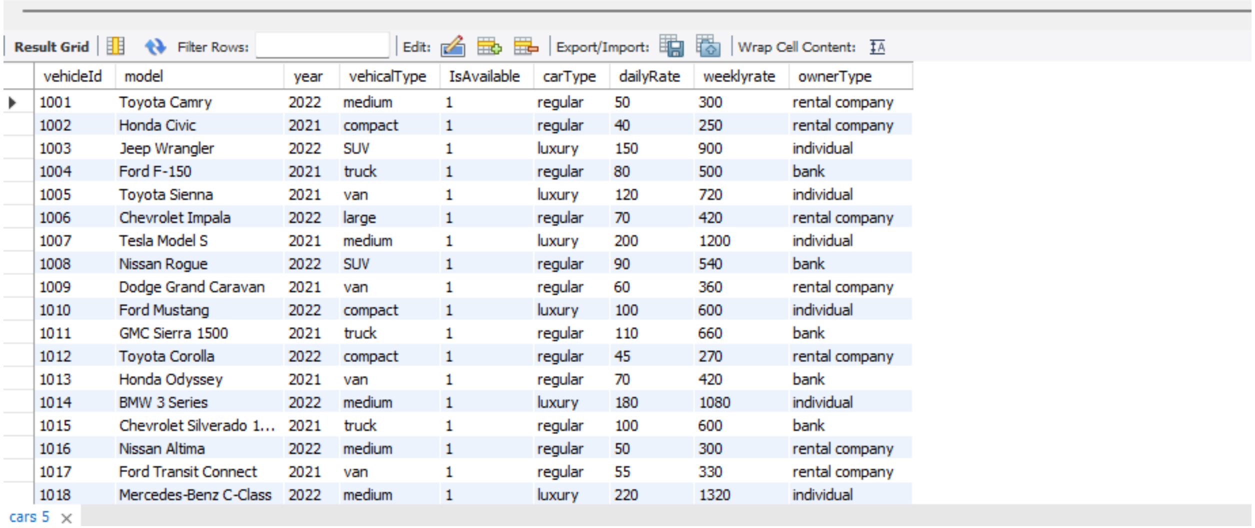
customerId	custname	phone	customerType
1	J.Doe	123-456-7890	individual
2	J.Smith	234-567-8901	business
3	B.Johnson	345-678-9012	individual
4	S.James	456-789-0123	business
5	B.Brown	567-890-1234	individual
6	A.Green	678-901-2345	business
7	T.Jones	789-012-3456	individual
8	L.White	890-123-4567	business
9	M.Black	901-234-5678	individual
10	K.Lee	012-345-6789	business
NULL	NULL	NULL	NULL

Car table:

Insert query:

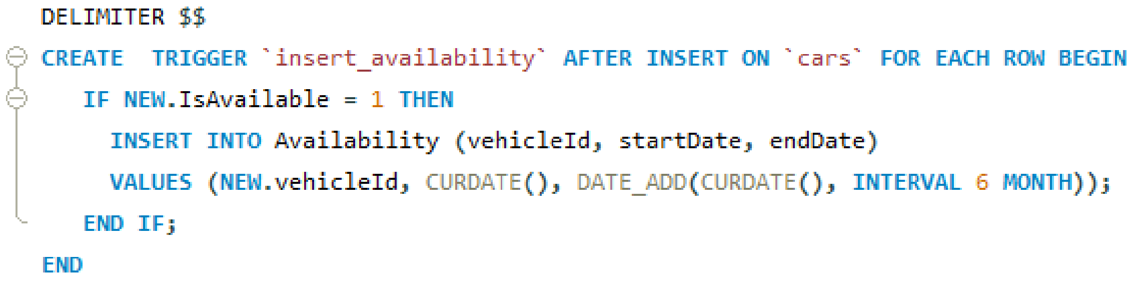


```
select * from cars;

INSERT INTO `cars` (`model`, `year`, `vehicalType`, `IsAvailable`, `carType`, `dailyRate`, `weeklyrate`, `ownerType`)
VALUES
('Toyota Camry', 2022, 'medium', 1, 'regular', 50, 300, 'rental company'),
('Honda Civic', 2021, 'compact', 1, 'regular', 40, 250, 'rental company'),
('Jeep Wrangler', 2022, 'SUV', 1, 'luxury', 150, 900, 'individual'),
('Ford F-150', 2021, 'truck', 1, 'regular', 80, 500, 'bank'),
('Toyota Sienna', 2021, 'van', 1, 'luxury', 120, 720, 'individual'),
('Chevrolet Impala', 2022, 'large', 1, 'regular', 70, 420, 'rental company'),
('Tesla Model S', 2021, 'medium', 1, 'luxury', 200, 1200, 'individual'),
('Nissan Rogue', 2022, 'SUV', 1, 'regular', 90, 540, 'bank'),
('Dodge Grand Caravan', 2021, 'van', 1, 'regular', 60, 360, 'rental company'),
('Ford Mustang', 2022, 'compact', 1, 'luxury', 100, 600, 'individual'),
('GMC Sierra 1500', 2021, 'truck', 1, 'regular', 110, 660, 'bank'),
('Toyota Corolla', 2022, 'compact', 1, 'regular', 45, 270, 'rental company'),
('Honda Odyssey', 2021, 'van', 1, 'regular', 70, 420, 'bank'),
('BMW 3 Series', 2022, 'medium', 1, 'luxury', 180, 1080, 'individual'),
```



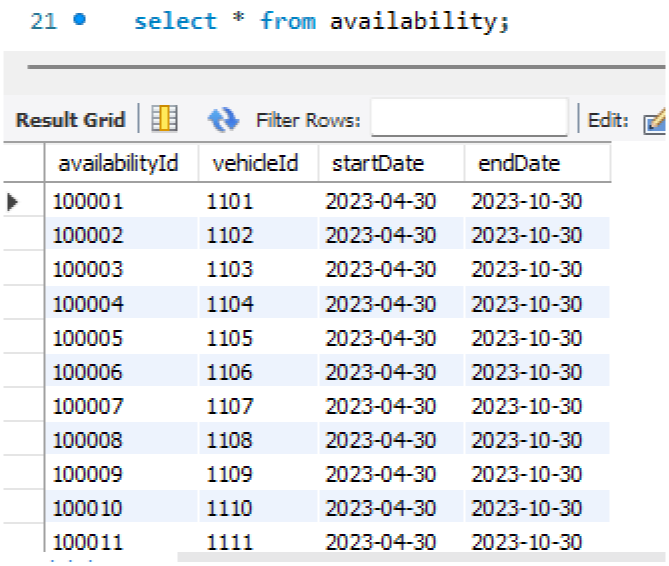
vehicleId	model	year	vehicalType	IsAvailable	carType	dailyRate	weeklyrate	ownerType
1001	Toyota Camry	2022	medium	1	regular	50	300	rental company
1002	Honda Civic	2021	compact	1	regular	40	250	rental company
1003	Jeep Wrangler	2022	SUV	1	luxury	150	900	individual
1004	Ford F-150	2021	truck	1	regular	80	500	bank
1005	Toyota Sienna	2021	van	1	luxury	120	720	individual
1006	Chevrolet Impala	2022	large	1	regular	70	420	rental company
1007	Tesla Model S	2021	medium	1	luxury	200	1200	individual
1008	Nissan Rogue	2022	SUV	1	regular	90	540	bank
1009	Dodge Grand Caravan	2021	van	1	regular	60	360	rental company
1010	Ford Mustang	2022	compact	1	luxury	100	600	individual
1011	GMC Sierra 1500	2021	truck	1	regular	110	660	bank
1012	Toyota Corolla	2022	compact	1	regular	45	270	rental company
1013	Honda Odyssey	2021	van	1	regular	70	420	bank
1014	BMW 3 Series	2022	medium	1	luxury	180	1080	individual
1015	Chevrolet Silverado 1...	2021	truck	1	regular	100	600	bank
1016	Nissan Altima	2022	medium	1	regular	50	300	rental company
1017	Ford Transit Connect	2021	van	1	regular	55	330	rental company
1018	Mercedes-Benz C-Class	2022	medium	1	luxury	220	1320	individual

Trigger on cars table after inserting cars details:



```
DELIMITER $$
CREATE  TRIGGER `insert_availability` AFTER INSERT ON `cars` FOR EACH ROW BEGIN
   IF NEW.IsAvailable = 1 THEN
     INSERT INTO Availability (vehicleId, startDate, endDate)
     VALUES (NEW.vehicleId, CURDATE(), DATE_ADD(CURDATE(), INTERVAL 6 MONTH));
   END IF;
END
```

Availability table:

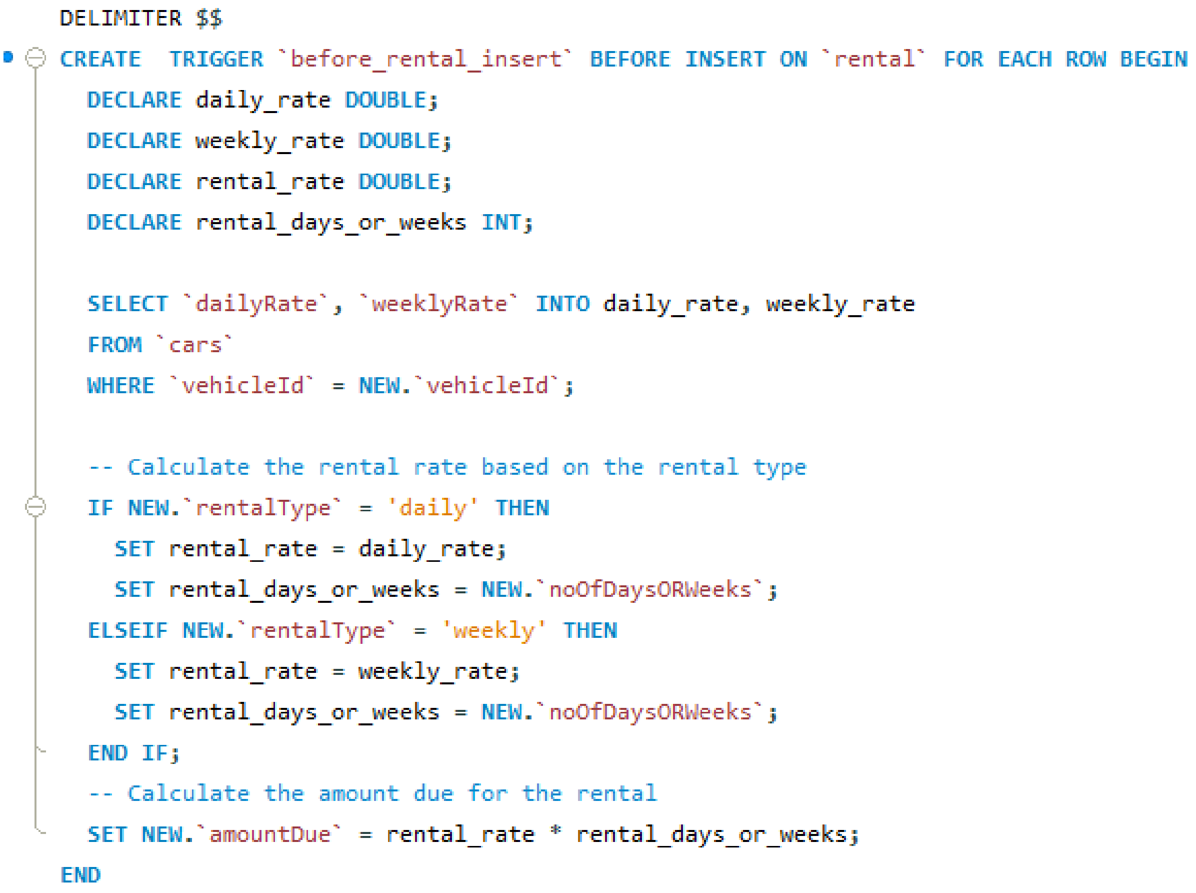


```
select * from availability;
```

availabilityId	vehicleId	startDate	endDate
100001	1101	2023-04-30	2023-10-30
100002	1102	2023-04-30	2023-10-30
100003	1103	2023-04-30	2023-10-30
100004	1104	2023-04-30	2023-10-30
100005	1105	2023-04-30	2023-10-30
100006	1106	2023-04-30	2023-10-30
100007	1107	2023-04-30	2023-10-30
100008	1108	2023-04-30	2023-10-30
100009	1109	2023-04-30	2023-10-30
100010	1110	2023-04-30	2023-10-30
100011	1111	2023-04-30	2023-10-30

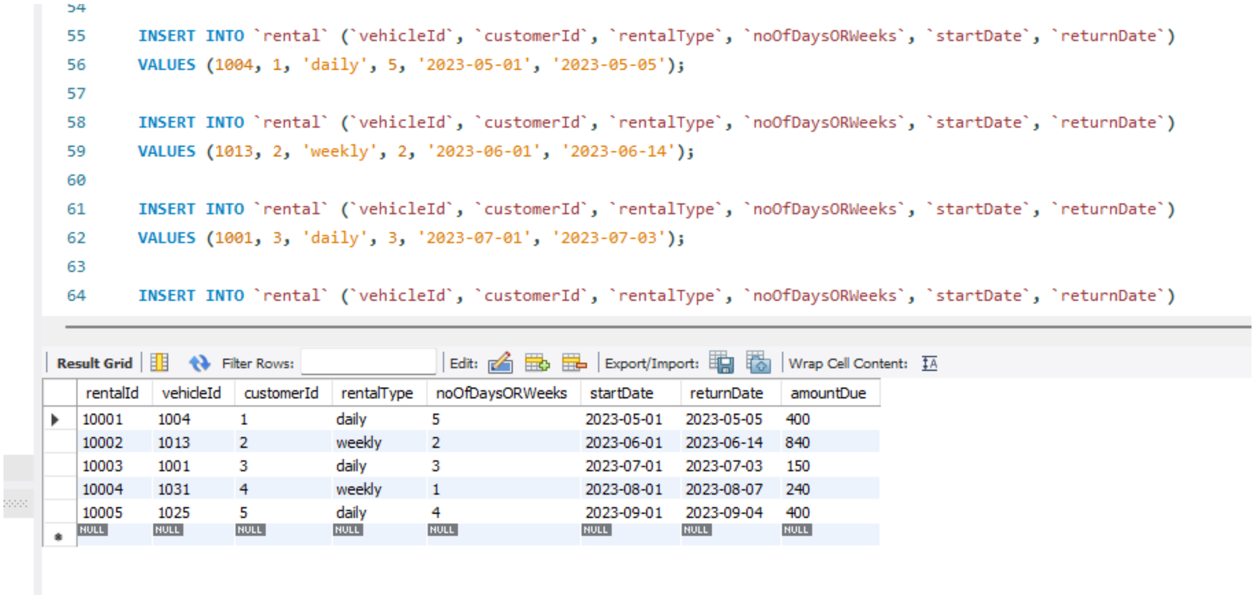
Rental table:

Tigger before inserting data into rental table:



```
DELIMITER $$
CREATE  TRIGGER `before_rental_insert` BEFORE INSERT ON `rental` FOR EACH ROW BEGIN
  DECLARE daily_rate DOUBLE;
  DECLARE weekly_rate DOUBLE;
  DECLARE rental_rate DOUBLE;
  DECLARE rental_days_or_weeks INT;

  SELECT `dailyRate`, `weeklyRate` INTO daily_rate, weekly_rate
  FROM `cars`
  WHERE `vehicleId` = NEW.`vehicleId`;

  -- Calculate the rental rate based on the rental type
  IF NEW.`rentalType` = 'daily' THEN
    SET rental_rate = daily_rate;
    SET rental_days_or_weeks = NEW.`noOfDaysORWeeks`;
  ELSEIF NEW.`rentalType` = 'weekly' THEN
    SET rental_rate = weekly_rate;
    SET rental_days_or_weeks = NEW.`noOfDaysORWeeks`;
  END IF;
  -- Calculate the amount due for the rental
  SET NEW.`amountDue` = rental_rate * rental_days_or_weeks;
END
```

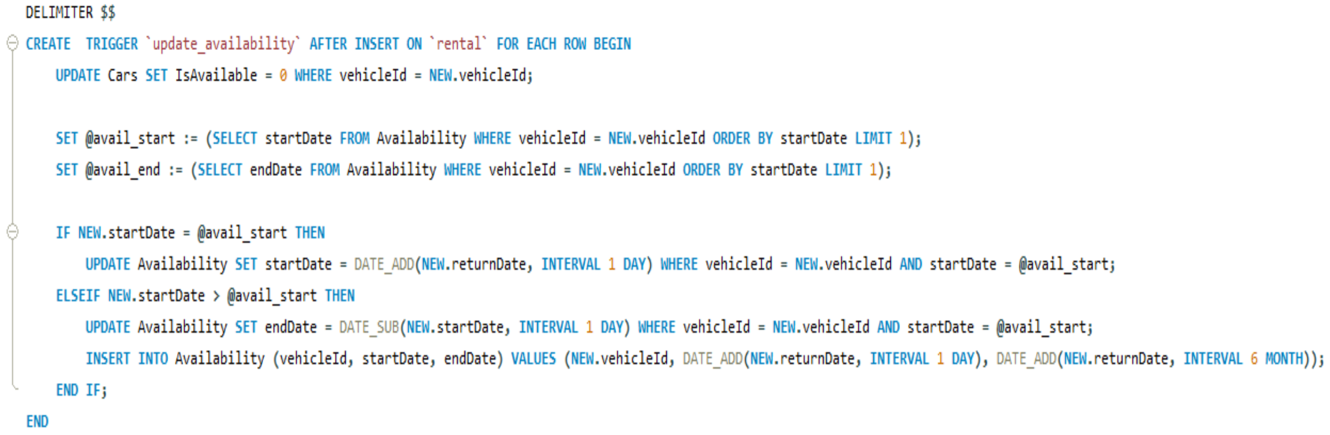
Insert query:



```sql
INSERT INTO `rental` (`vehicleId`, `customerId`, `rentalType`, `noOfDaysORWeeks`, `startDate`, `returnDate`)
VALUES (1004, 1, 'daily', 5, '2023-05-01', '2023-05-05');

INSERT INTO `rental` (`vehicleId`, `customerId`, `rentalType`, `noOfDaysORWeeks`, `startDate`, `returnDate`)
VALUES (1013, 2, 'weekly', 2, '2023-06-01', '2023-06-14');

INSERT INTO `rental` (`vehicleId`, `customerId`, `rentalType`, `noOfDaysORWeeks`, `startDate`, `returnDate`)
VALUES (1001, 3, 'daily', 3, '2023-07-01', '2023-07-03');

INSERT INTO `rental` (`vehicleId`, `customerId`, `rentalType`, `noOfDaysORWeeks`, `startDate`, `returnDate`)
```

rentalId	vehicleId	customerId	rentalType	noOfDaysORWeeks	startDate	returnDate	amountDue
10001	1004	1	daily	5	2023-05-01	2023-05-05	400
10002	1013	2	weekly	2	2023-06-01	2023-06-14	840
10003	1001	3	daily	3	2023-07-01	2023-07-03	150
10004	1031	4	weekly	1	2023-08-01	2023-08-07	240
10005	1025	5	daily	4	2023-09-01	2023-09-04	400
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

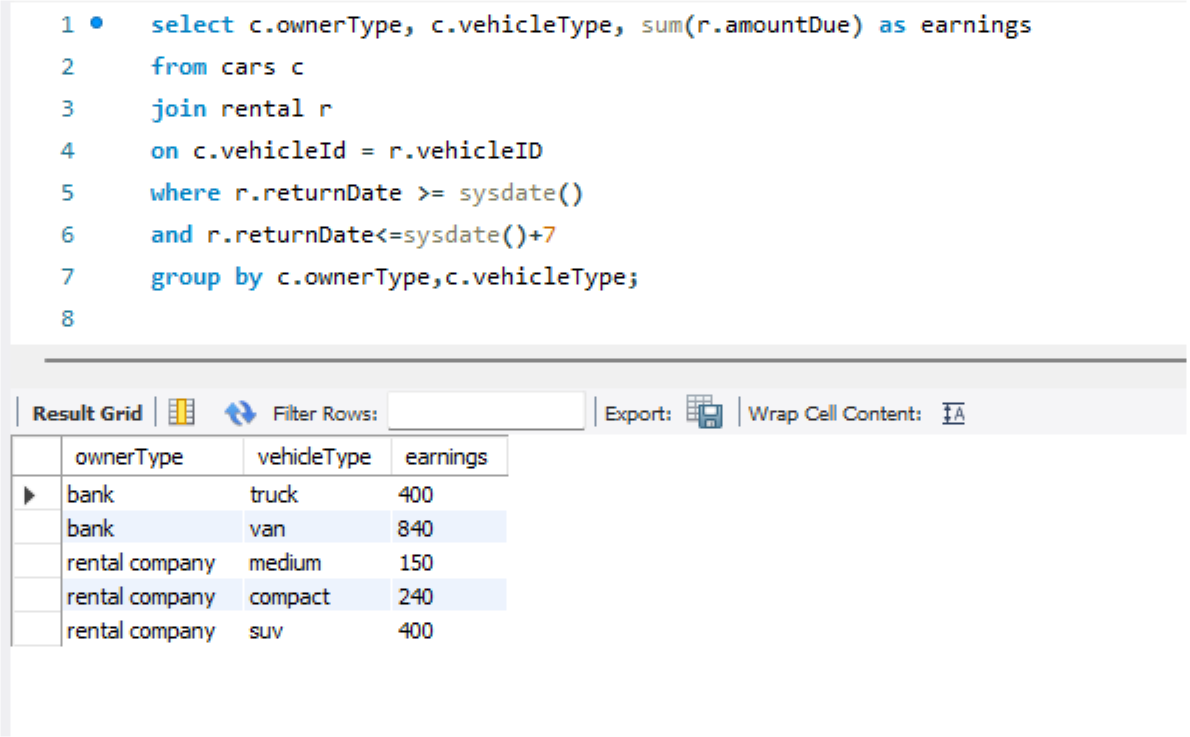
Tigger after inserting data into rental table:



```sql
DELIMITER $$
CREATE  TRIGGER `update_availability` AFTER INSERT ON `rental` FOR EACH ROW BEGIN
    UPDATE Cars SET IsAvailable = 0 WHERE vehicleId = NEW.vehicleId;

    SET @avail_start := (SELECT startDate FROM Availability WHERE vehicleId = NEW.vehicleId ORDER BY startDate LIMIT 1);
    SET @avail_end := (SELECT endDate FROM Availability WHERE vehicleId = NEW.vehicleId ORDER BY startDate LIMIT 1);

    IF NEW.startDate = @avail_start THEN
        UPDATE Availability SET startDate = DATE_ADD(NEW.returnDate, INTERVAL 1 DAY) WHERE vehicleId = NEW.vehicleId AND startDate = @avail_start;
    ELSEIF NEW.startDate > @avail_start THEN
        UPDATE Availability SET endDate = DATE_SUB(NEW.startDate, INTERVAL 1 DAY) WHERE vehicleId = NEW.vehicleId AND startDate = @avail_start;
        INSERT INTO Availability (vehicleId, startDate, endDate) VALUES (NEW.vehicleId, DATE_ADD(NEW.returnDate, INTERVAL 1 DAY), DATE_ADD(NEW.returnDate, INTERVAL 6 MONTH));
    END IF;
END
```

Question3:

Write a query that will prepare a report for weekly earnings by owner, by car type and per car unit that owner owns within that car type.

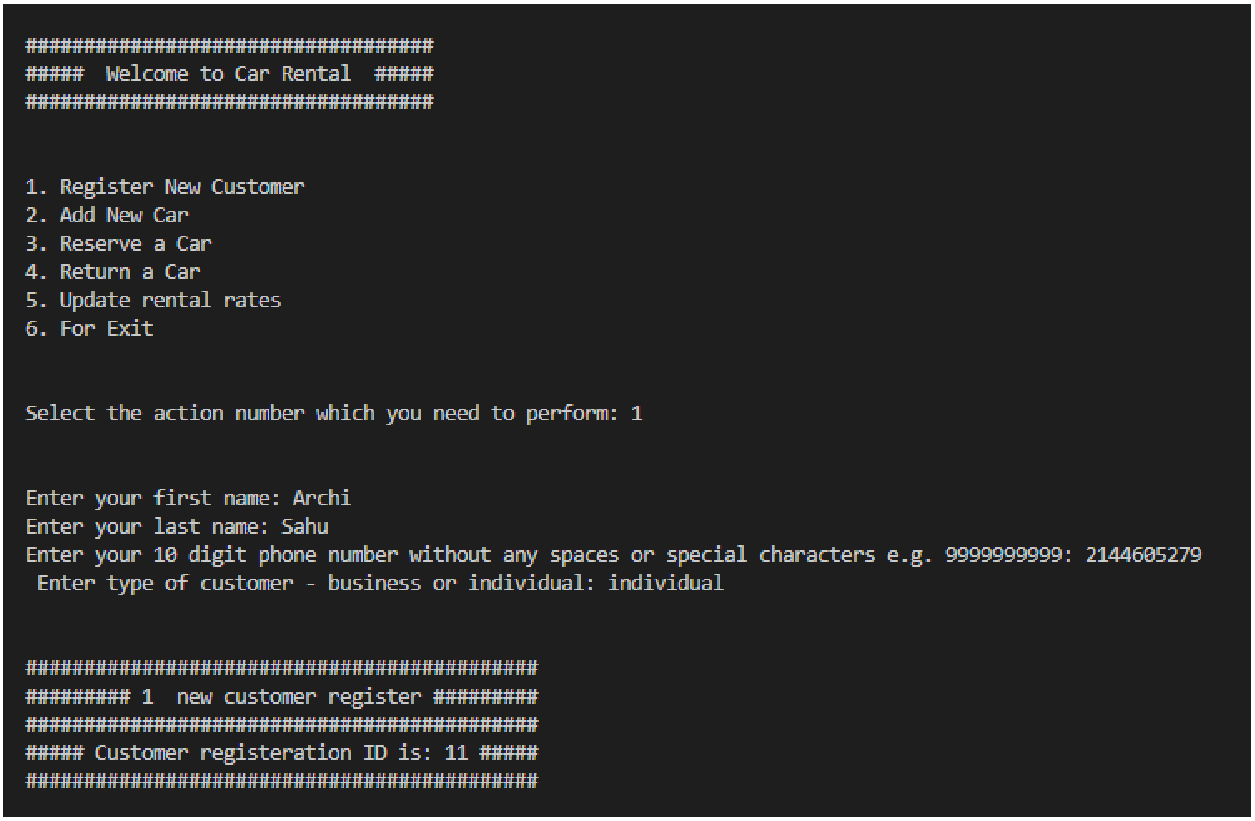


```sql
select c.ownerType, c.vehicleType, sum(r.amountDue) as earnings
from cars c
join rental r
on c.vehicleId = r.vehicleID
where r.returnDate >= sysdate()
and r.returnDate<=sysdate()+7
group by c.ownerType,c.vehicleType;

```

ownerType	vehicleType	earnings
bank	truck	400
bank	van	840
rental company	medium	150
rental company	compact	240
rental company	suv	400

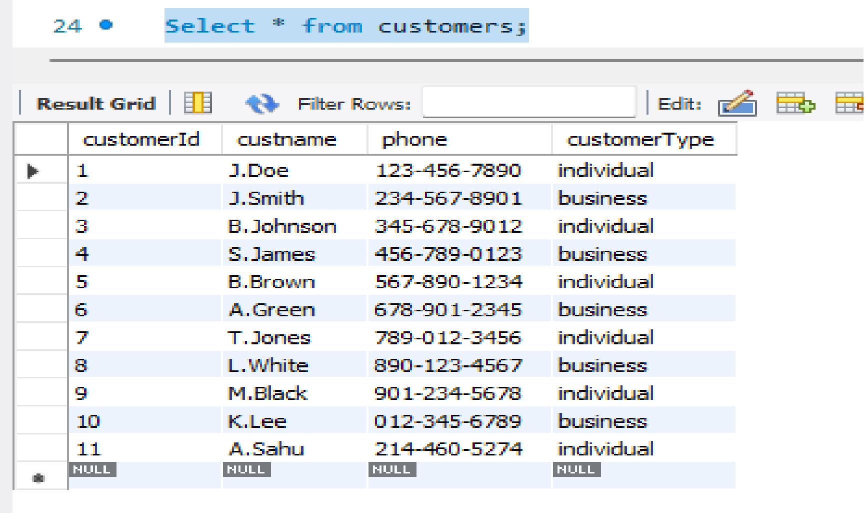
Question4:
Write the following database update transactions using any suitable programming or scripting language (e.g. JAVA/JDBC, Python or PHP).

4.1 The first transaction is to add information about a new CUSTOMER.



```
###################################
#####  Welcome to Car Rental  #####
###################################


1. Register New Customer
2. Add New Car
3. Reserve a Car
4. Return a Car
5. Update rental rates
6. For Exit


Select the action number which you need to perform: 1


Enter your first name: Archi
Enter your last name: Sahu
Enter your 10 digit phone number without any spaces or special characters e.g. 9999999999: 2144605279
 Enter type of customer - business or individual: individual


############################################
######### 1  new customer register #########
############################################
##### Customer registeration ID is: 11 #####
############################################
```

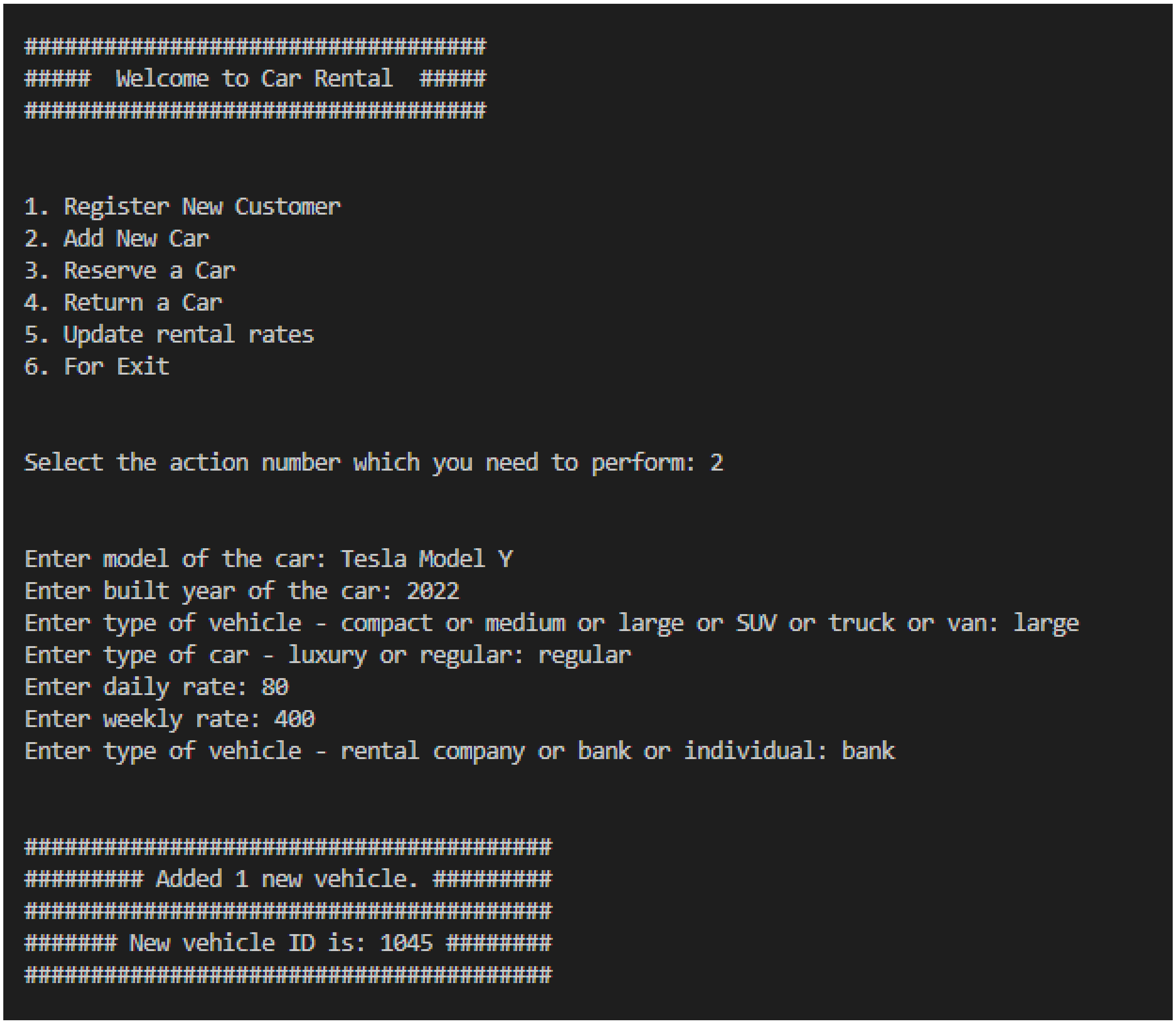
After executing this, you can find the updated data in the SQL as shown below.



```
Select * from customers;
```

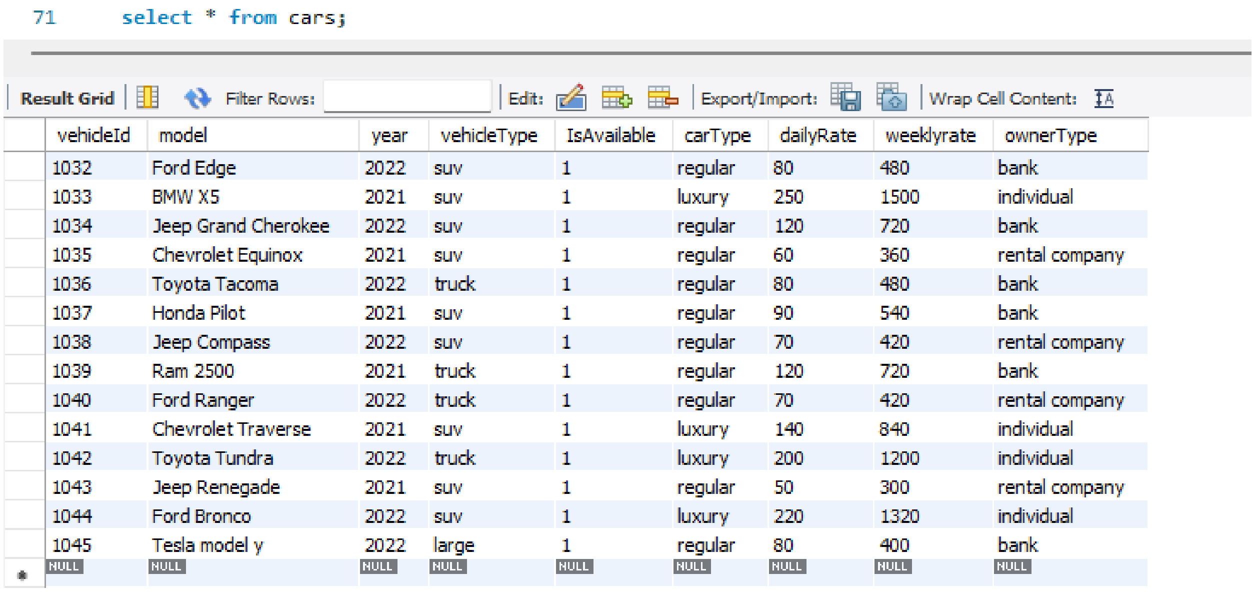
customerId	custname	phone	customerType
1	J.Doe	123-456-7890	individual
2	J.Smith	234-567-8901	business
3	B.Johnson	345-678-9012	individual
4	S.James	456-789-0123	business
5	B.Brown	567-890-1234	individual
6	A.Green	678-901-2345	business
7	T.Jones	789-012-3456	individual
8	L.White	890-123-4567	business
9	M.Black	901-234-5678	individual
10	K.Lee	012-345-6789	business
11	A.Sahu	214-460-5274	individual
NULL	NULL	NULL	NULL

4.2 The second transaction is to add all the information about a new CAR.



```
###################################
#####  Welcome to Car Rental  #####
###################################


1. Register New Customer
2. Add New Car
3. Reserve a Car
4. Return a Car
5. Update rental rates
6. For Exit


Select the action number which you need to perform: 2


Enter model of the car: Tesla Model Y
Enter built year of the car: 2022
Enter type of vehicle - compact or medium or large or SUV or truck or van: large
Enter type of car - luxury or regular: regular
Enter daily rate: 80
Enter weekly rate: 400
Enter type of vehicle - rental company or bank or individual: bank


########################################
######### Added 1 new vehicle. #########
########################################
####### New vehicle ID is: 1045 ########
########################################
```

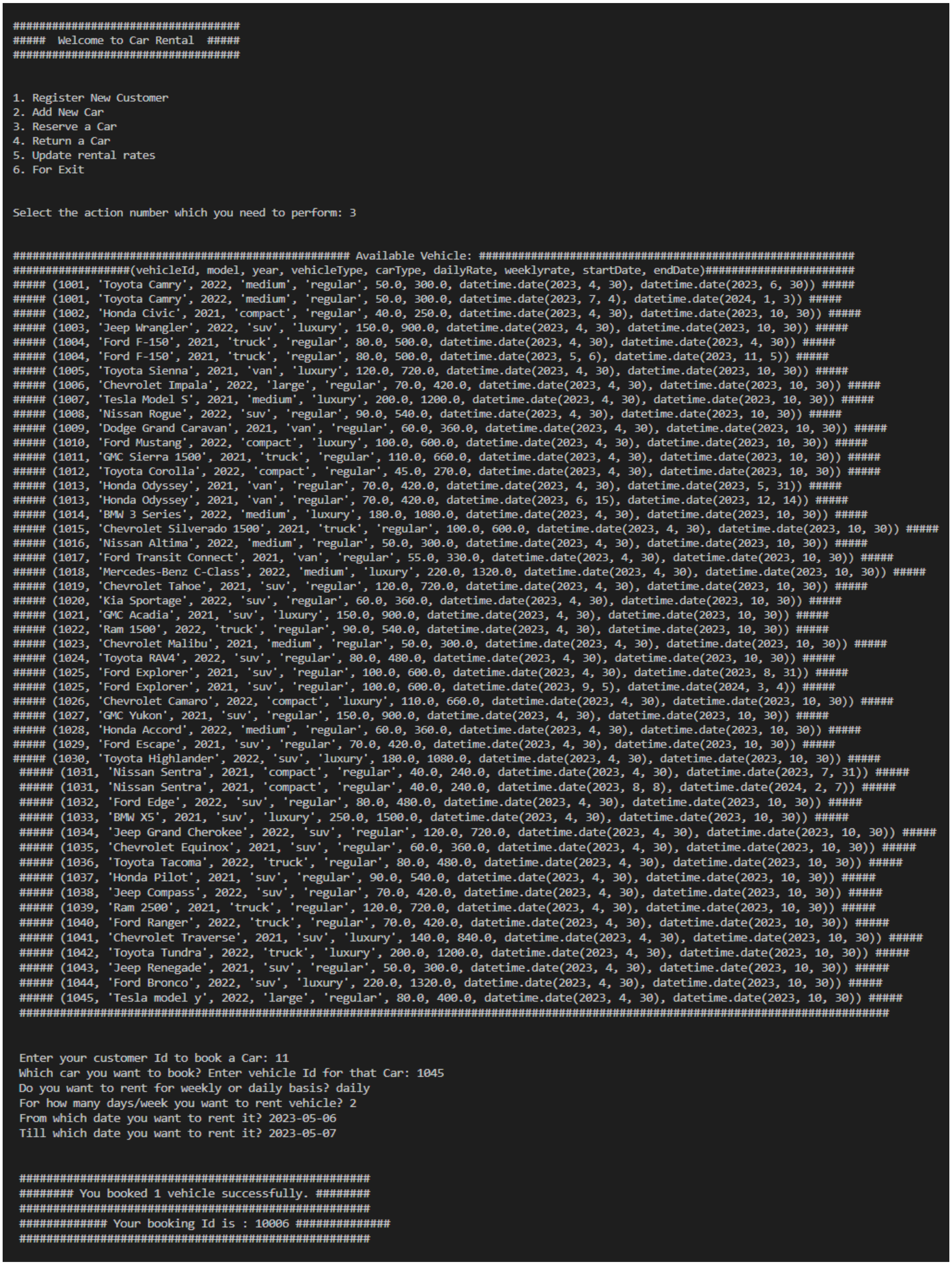
After executing this, you can find the updated data in the SQL as shown below.



```
select * from cars;
```

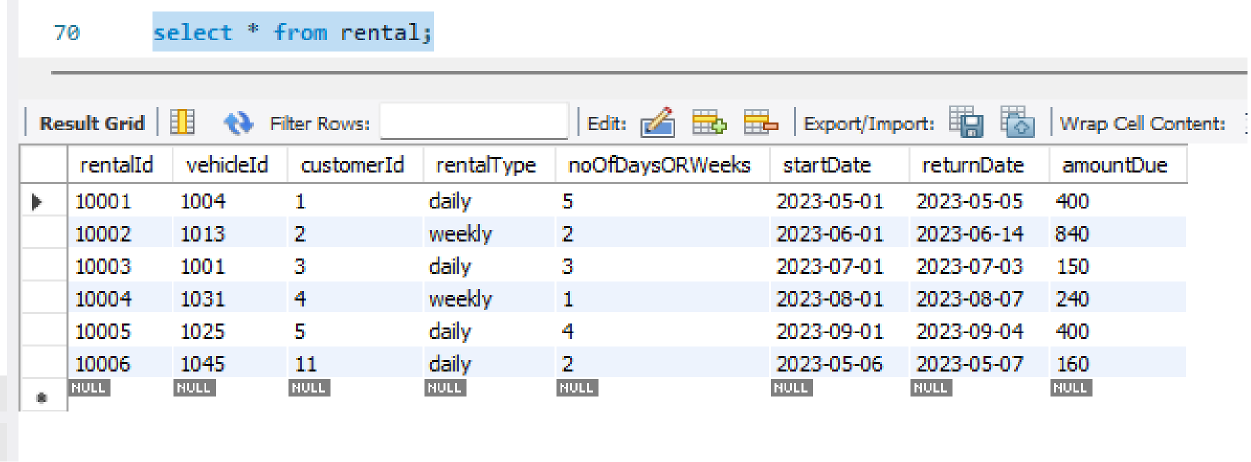
vehicleId	model	year	vehicleType	IsAvailable	carType	dailyRate	weeklyrate	ownerType
1032	Ford Edge	2022	suv	1	regular	80	480	bank
1033	BMW X5	2021	suv	1	luxury	250	1500	individual
1034	Jeep Grand Cherokee	2022	suv	1	regular	120	720	bank
1035	Chevrolet Equinox	2021	suv	1	regular	60	360	rental company
1036	Toyota Tacoma	2022	truck	1	regular	80	480	bank
1037	Honda Pilot	2021	suv	1	regular	90	540	bank
1038	Jeep Compass	2022	suv	1	regular	70	420	rental company
1039	Ram 2500	2021	truck	1	regular	120	720	bank
1040	Ford Ranger	2022	truck	1	regular	70	420	rental company
1041	Chevrolet Traverse	2021	suv	1	luxury	140	840	individual
1042	Toyota Tundra	2022	truck	1	luxury	200	1200	individual
1043	Jeep Renegade	2021	suv	1	regular	50	300	rental company
1044	Ford Bronco	2022	suv	1	luxury	220	1320	individual
1045	Tesla model y	2022	large	1	regular	80	400	bank
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

4.3 The third transaction is to add all the information about a new rental reservation (this must find a free car of the appropriate type for the rental period).



```
##################################
#####  Welcome to Car Rental  #####
##################################


1. Register New Customer
2. Add New Car
3. Reserve a Car
4. Return a Car
5. Update rental rates
6. For Exit


Select the action number which you need to perform: 3


################################################### Available Vehicle: ##########################################################
#################(vehicleId, model, year, vehicleType, carType, dailyRate, weeklyrate, startDate, endDate)######################
##### (1001, 'Toyota Camry', 2022, 'medium', 'regular', 50.0, 300.0, datetime.date(2023, 4, 30), datetime.date(2023, 6, 30)) #####
##### (1001, 'Toyota Camry', 2022, 'medium', 'regular', 50.0, 300.0, datetime.date(2023, 7, 4), datetime.date(2024, 1, 3)) #####
##### (1002, 'Honda Civic', 2021, 'compact', 'regular', 40.0, 250.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1003, 'Jeep Wrangler', 2022, 'suv', 'luxury', 150.0, 900.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1004, 'Ford F-150', 2021, 'truck', 'regular', 80.0, 500.0, datetime.date(2023, 4, 30), datetime.date(2023, 4, 30)) #####
##### (1004, 'Ford F-150', 2021, 'truck', 'regular', 80.0, 500.0, datetime.date(2023, 5, 6), datetime.date(2023, 11, 5)) #####
##### (1005, 'Toyota Sienna', 2021, 'van', 'luxury', 120.0, 720.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1006, 'Chevrolet Impala', 2022, 'large', 'regular', 70.0, 420.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1007, 'Tesla Model S', 2021, 'medium', 'luxury', 200.0, 1200.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1008, 'Nissan Rogue', 2022, 'suv', 'regular', 90.0, 540.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1009, 'Dodge Grand Caravan', 2021, 'van', 'regular', 60.0, 360.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1010, 'Ford Mustang', 2022, 'compact', 'luxury', 100.0, 600.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1011, 'GMC Sierra 1500', 2021, 'truck', 'regular', 110.0, 660.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1012, 'Toyota Corolla', 2022, 'compact', 'regular', 45.0, 270.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1013, 'Honda Odyssey', 2021, 'van', 'regular', 70.0, 420.0, datetime.date(2023, 4, 30), datetime.date(2023, 5, 31)) #####
##### (1013, 'Honda Odyssey', 2021, 'van', 'regular', 70.0, 420.0, datetime.date(2023, 6, 15), datetime.date(2023, 12, 14)) #####
##### (1014, 'BMW 3 Series', 2022, 'medium', 'luxury', 180.0, 1080.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1015, 'Chevrolet Silverado 1500', 2021, 'truck', 'regular', 100.0, 600.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1016, 'Nissan Altima', 2022, 'medium', 'regular', 50.0, 300.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1017, 'Ford Transit Connect', 2021, 'van', 'regular', 55.0, 330.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1018, 'Mercedes-Benz C-Class', 2022, 'medium', 'luxury', 220.0, 1320.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1019, 'Chevrolet Tahoe', 2021, 'suv', 'regular', 120.0, 720.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1020, 'Kia Sportage', 2022, 'suv', 'regular', 60.0, 360.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1021, 'GMC Acadia', 2021, 'suv', 'luxury', 150.0, 900.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1022, 'Ram 1500', 2022, 'truck', 'regular', 90.0, 540.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1023, 'Chevrolet Malibu', 2021, 'medium', 'regular', 50.0, 300.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1024, 'Toyota RAV4', 2022, 'suv', 'regular', 80.0, 480.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1025, 'Ford Explorer', 2021, 'suv', 'regular', 100.0, 600.0, datetime.date(2023, 4, 30), datetime.date(2023, 8, 31)) #####
##### (1025, 'Ford Explorer', 2021, 'suv', 'regular', 100.0, 600.0, datetime.date(2023, 9, 5), datetime.date(2024, 3, 4)) #####
##### (1026, 'Chevrolet Camaro', 2022, 'compact', 'luxury', 110.0, 660.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1027, 'GMC Yukon', 2021, 'suv', 'regular', 150.0, 900.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1028, 'Honda Accord', 2022, 'medium', 'regular', 60.0, 360.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1029, 'Ford Escape', 2021, 'suv', 'regular', 70.0, 420.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
##### (1030, 'Toyota Highlander', 2022, 'suv', 'luxury', 180.0, 1080.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1031, 'Nissan Sentra', 2021, 'compact', 'regular', 40.0, 240.0, datetime.date(2023, 4, 30), datetime.date(2023, 7, 31)) #####
 ##### (1031, 'Nissan Sentra', 2021, 'compact', 'regular', 40.0, 240.0, datetime.date(2023, 8, 8), datetime.date(2024, 2, 7)) #####
 ##### (1032, 'Ford Edge', 2022, 'suv', 'regular', 80.0, 480.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1033, 'BMW X5', 2021, 'suv', 'luxury', 250.0, 1500.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1034, 'Jeep Grand Cherokee', 2022, 'suv', 'regular', 120.0, 720.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1035, 'Chevrolet Equinox', 2021, 'suv', 'regular', 60.0, 360.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1036, 'Toyota Tacoma', 2022, 'truck', 'regular', 80.0, 480.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1037, 'Honda Pilot', 2021, 'suv', 'regular', 90.0, 540.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1038, 'Jeep Compass', 2022, 'suv', 'regular', 70.0, 420.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1039, 'Ram 2500', 2021, 'truck', 'regular', 120.0, 720.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1040, 'Ford Ranger', 2022, 'truck', 'regular', 70.0, 420.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1041, 'Chevrolet Traverse', 2021, 'suv', 'luxury', 140.0, 840.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1042, 'Toyota Tundra', 2022, 'truck', 'luxury', 200.0, 1200.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1043, 'Jeep Renegade', 2021, 'suv', 'regular', 50.0, 300.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1044, 'Ford Bronco', 2022, 'suv', 'luxury', 220.0, 1320.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ##### (1045, 'Tesla model y', 2022, 'large', 'regular', 80.0, 400.0, datetime.date(2023, 4, 30), datetime.date(2023, 10, 30)) #####
 ###################################################################################################################################


 Enter your customer Id to book a Car: 11
 Which car you want to book? Enter vehicle Id for that Car: 1045
 Do you want to rent for weekly or daily basis? daily
 For how many days/week you want to rent vehicle? 2
 From which date you want to rent it? 2023-05-06
 Till which date you want to rent it? 2023-05-07


 #####################################################
 ######## You booked 1 vehicle successfully. ########
 #####################################################
 ############ Your booking Id is : 10006 #############
 #####################################################
```

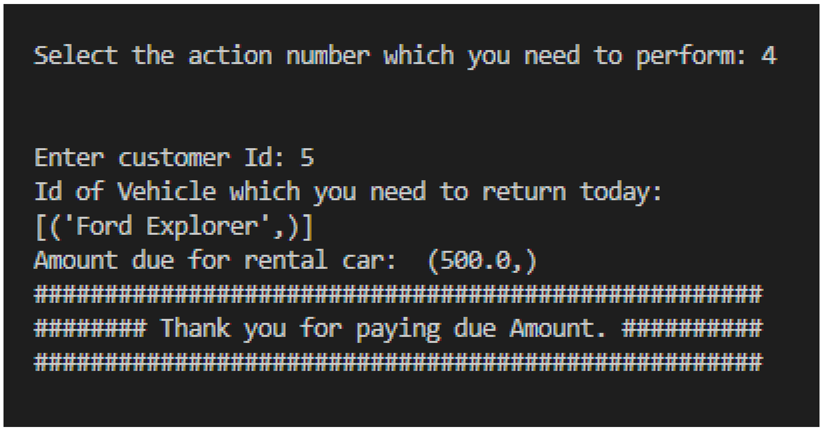
After executing this, you can find the updated data in the SQL as shown below.



```
select * from rental;
```

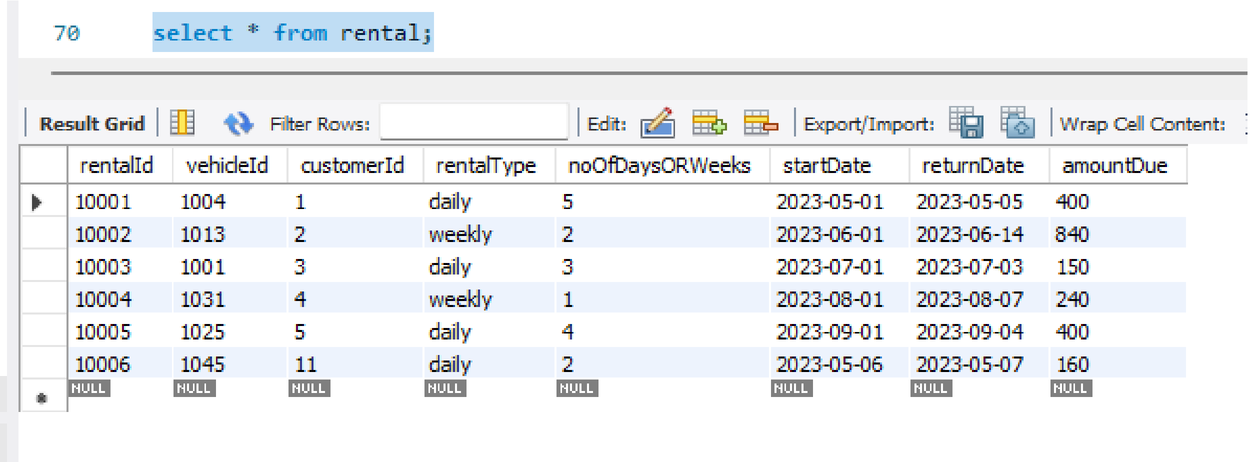
rentalId	vehicleId	customerId	rentalType	noOfDaysORWeeks	startDate	returnDate	amountDue
10001	1004	1	daily	5	2023-05-01	2023-05-05	400
10002	1013	2	weekly	2	2023-06-01	2023-06-14	840
10003	1001	3	daily	3	2023-07-01	2023-07-03	150
10004	1031	4	weekly	1	2023-08-01	2023-08-07	240
10005	1025	5	daily	4	2023-09-01	2023-09-04	400
10006	1045	11	daily	2	2023-05-06	2023-05-07	160
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

4.4 The fourth transaction is to handle the return of a rented car. This transaction should print the total customer payment due for the rental, and enter it in the database.



```
Select the action number which you need to perform: 4


Enter customer Id: 5
Id of Vehicle which you need to return today:
[('Ford Explorer',)]
Amount due for rental car:  (500.0,)
####################################################
######## Thank you for paying due Amount. ##########
####################################################
```

Inserting AmountDue in rental table using trigger before_rental_insert.

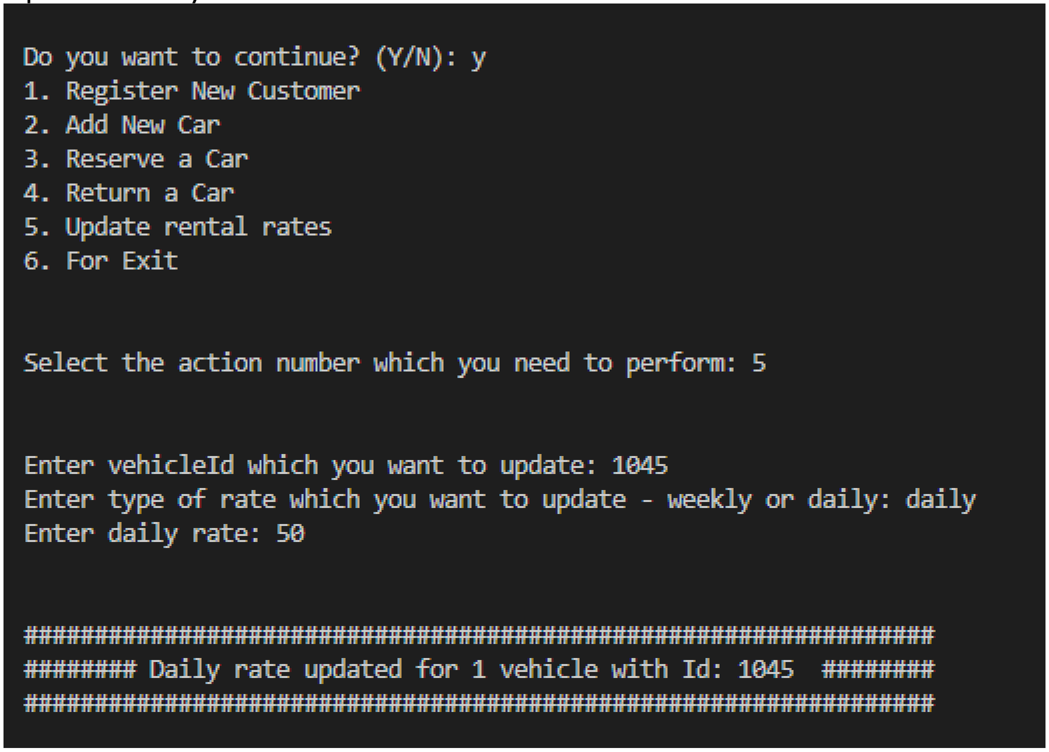


```
select * from rental;
```

rentalId	vehicleId	customerId	rentalType	noOfDaysORWeeks	startDate	returnDate	amountDue
10001	1004	1	daily	5	2023-05-01	2023-05-05	400
10002	1013	2	weekly	2	2023-06-01	2023-06-14	840
10003	1001	3	daily	3	2023-07-01	2023-07-03	150
10004	1031	4	weekly	1	2023-08-01	2023-08-07	240
10005	1025	5	daily	4	2023-09-01	2023-09-04	400
10006	1045	11	daily	2	2023-05-06	2023-05-07	160
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

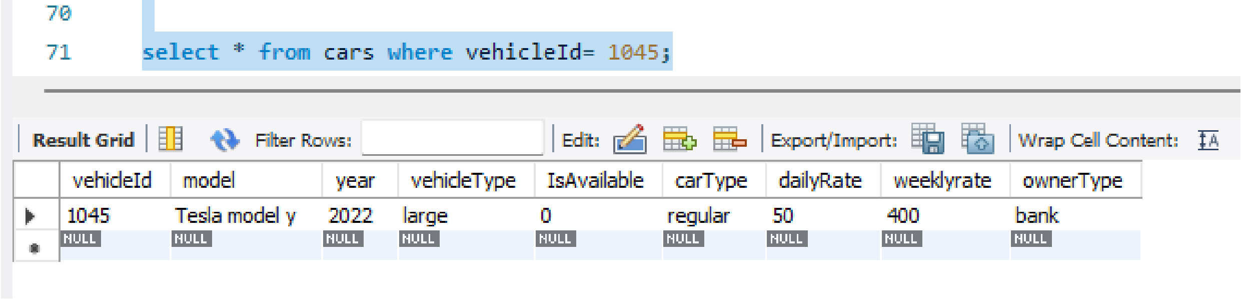
4.5 The fifth transaction is to enter or update the rental rates (daily and weekly) for a type of car.

Updated daily rate:



```
Do you want to continue? (Y/N): y
1. Register New Customer
2. Add New Car
3. Reserve a Car
4. Return a Car
5. Update rental rates
6. For Exit


Select the action number which you need to perform: 5


Enter vehicleId which you want to update: 1045
Enter type of rate which you want to update - weekly or daily: daily
Enter daily rate: 50


#################################################################
######## Daily rate updated for 1 vehicle with Id: 1045  ########
#################################################################
```

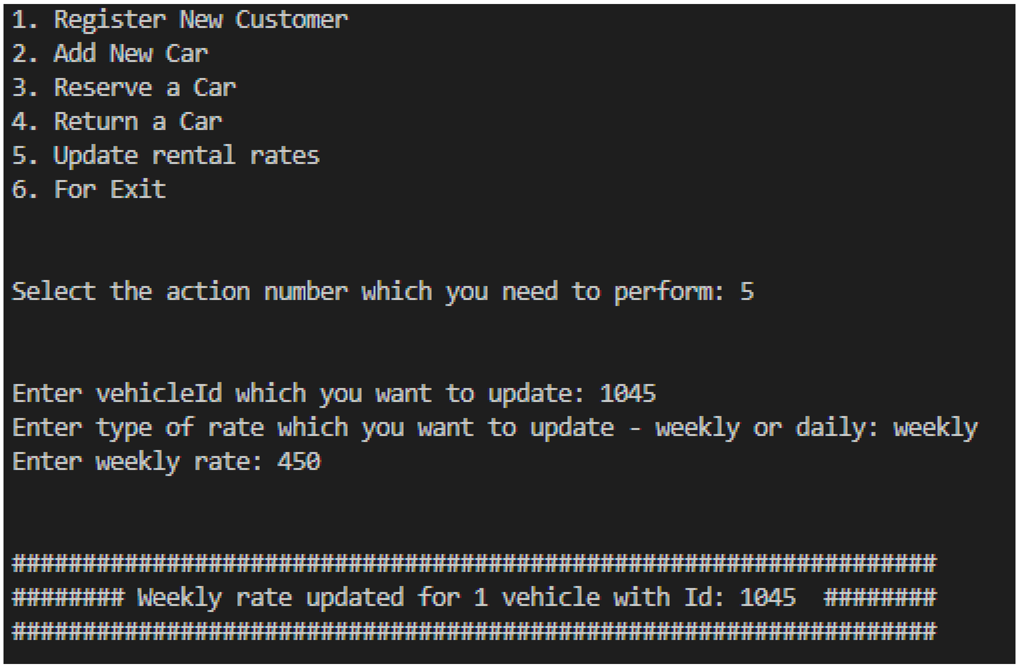
After executing this, you can find the updated data in the SQL as shown below.



```
select * from cars where vehicleId= 1045;
```

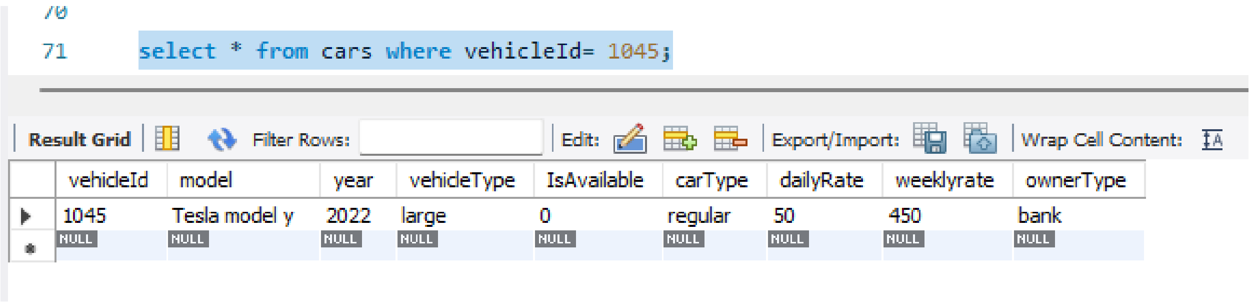
vehicleId	model	year	vehicleType	IsAvailable	carType	dailyRate	weeklyrate	ownerType
1045	Tesla model y	2022	large	0	regular	50	400	bank
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Update weekly rate:



```
1. Register New Customer
2. Add New Car
3. Reserve a Car
4. Return a Car
5. Update rental rates
6. For Exit


Select the action number which you need to perform: 5


Enter vehicleId which you want to update: 1045
Enter type of rate which you want to update - weekly or daily: weekly
Enter weekly rate: 450


###################################################################
######## Weekly rate updated for 1 vehicle with Id: 1045  ########
###################################################################
```

After executing this, you can find the updated data in the SQL as shown below.



```
select * from cars where vehicleId= 1045;
```

	vehicleId	model	year	vehicleType	IsAvailable	carType	dailyRate	weeklyrate	ownerType
▶	1045	Tesla model y	2022	large	0	regular	50	450	bank
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL