VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BCSE302P – DATABASE SYSTEMS

Submitted By

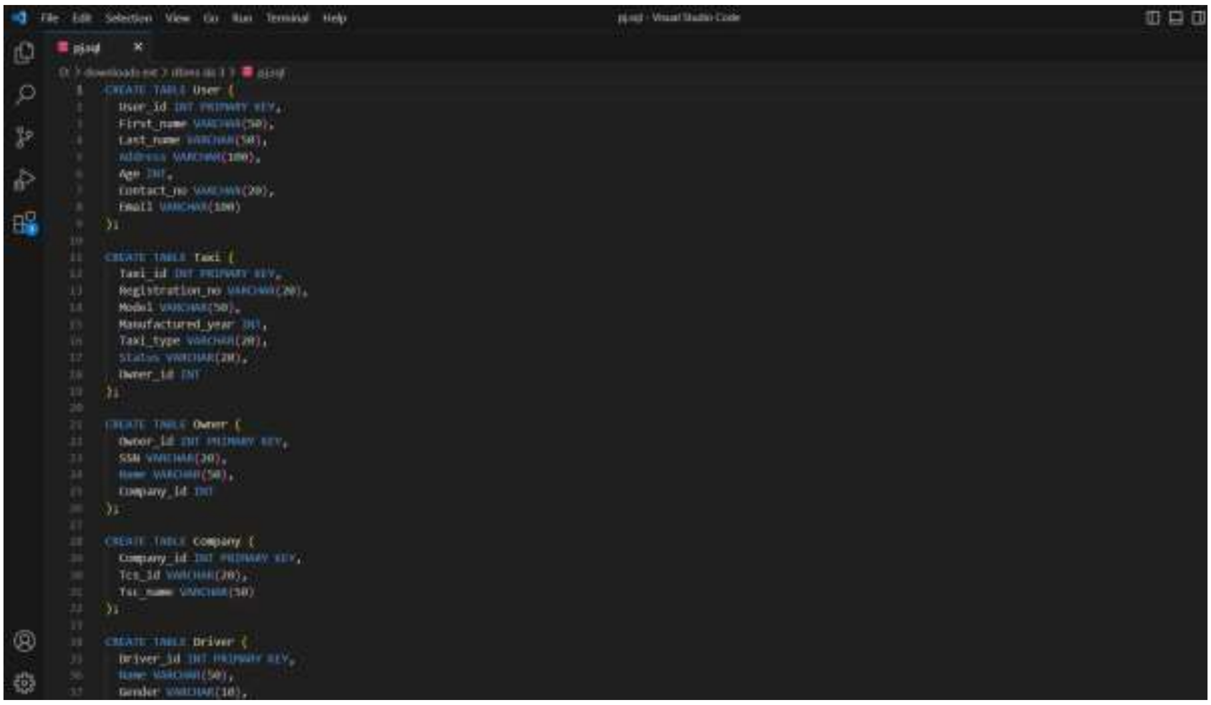
21BLC1411 Utkarsh Vyas

21BLC1377 Priyanshu Pandey

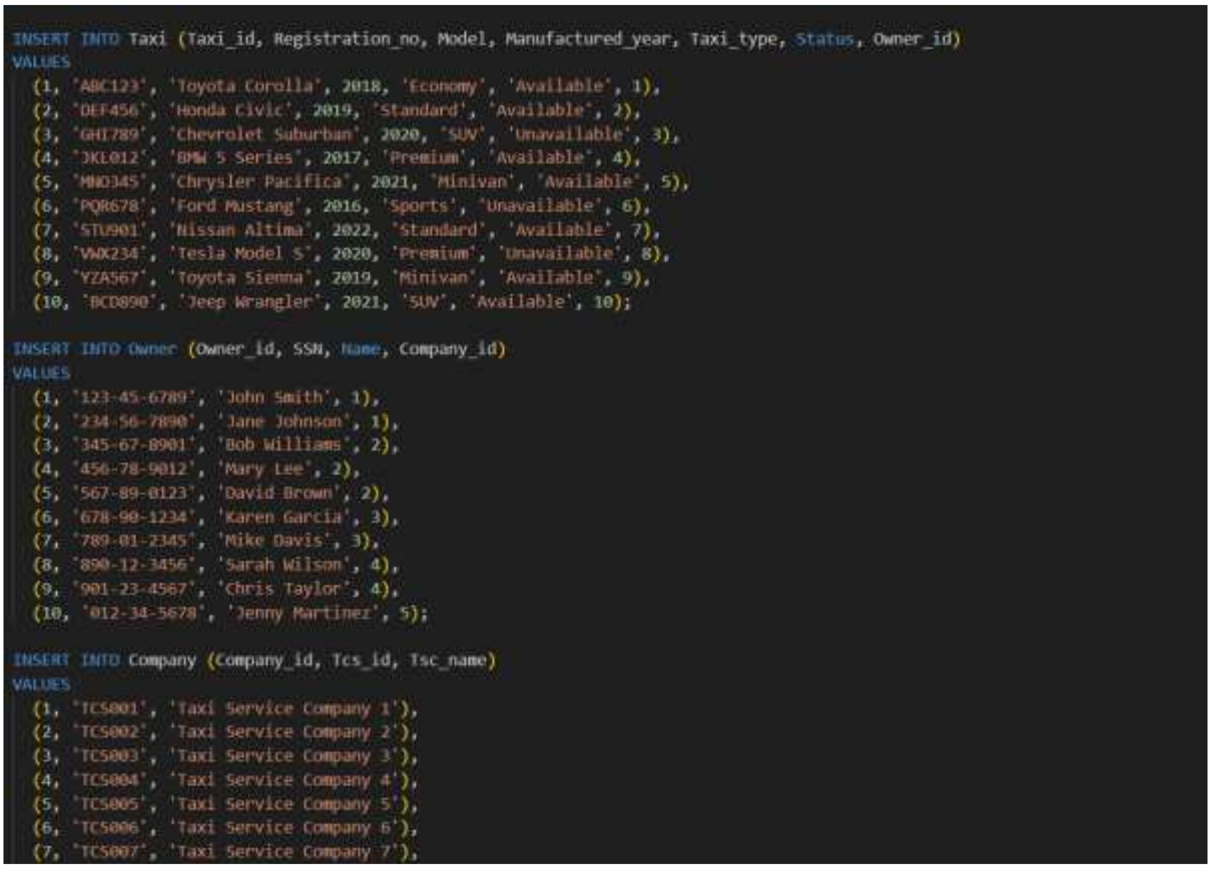
Submitted To

Dr. Sherley A

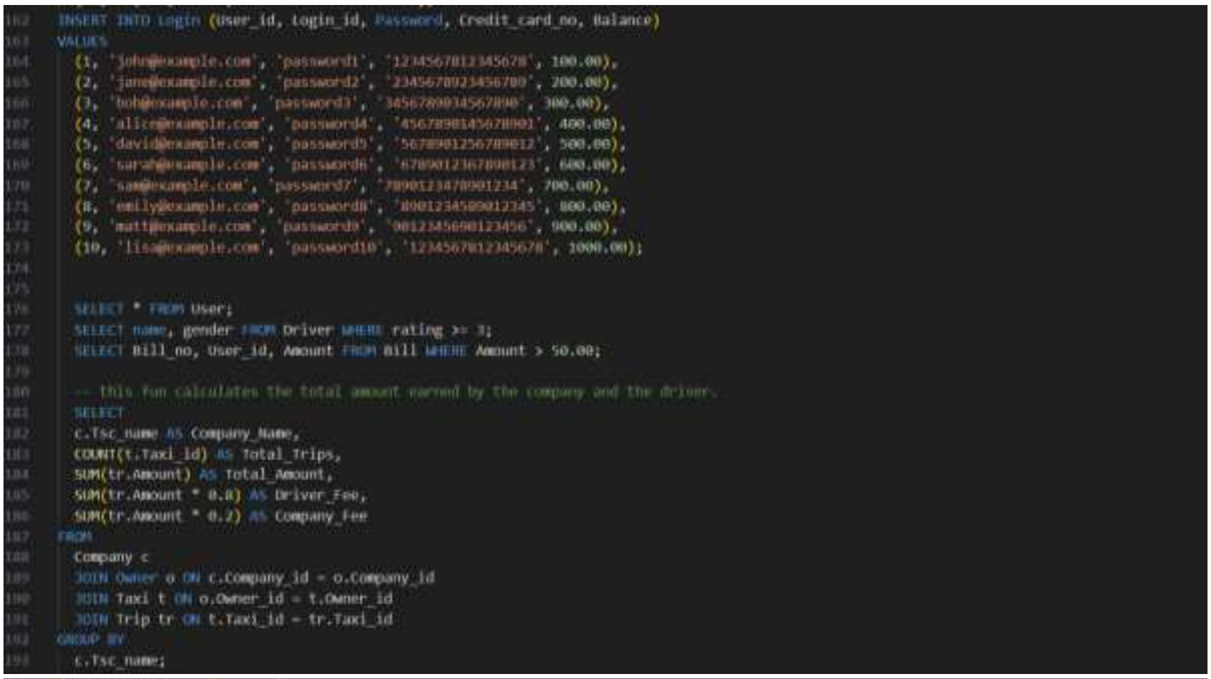
Code Snippets :


File Edit Selection View Go Run Terminal Help
pj.sql - Visual Studio Code
pj.sql

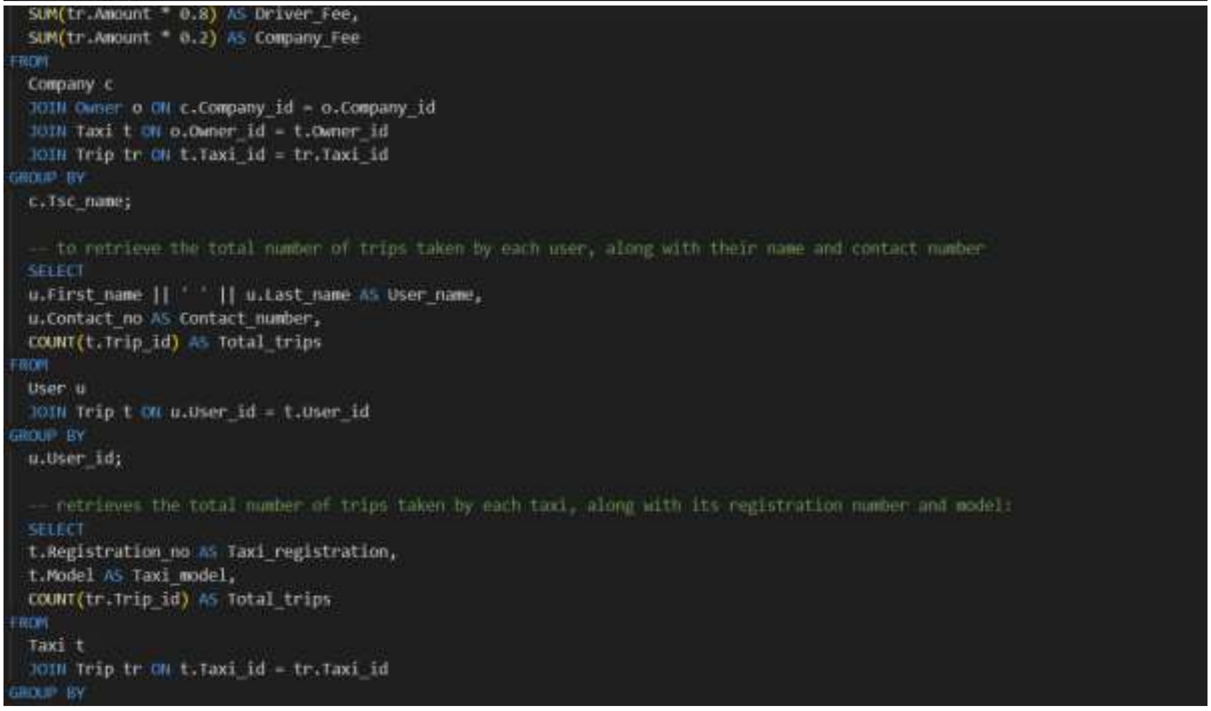
```sql
CREATE TABLE User (
  User_id INT PRIMARY KEY,
  First_name VARCHAR(50),
  Last_name VARCHAR(50),
  Address VARCHAR(100),
  Age INT,
  Contact_no VARCHAR(20),
  Email VARCHAR(100)
);

CREATE TABLE Taxi (
  Taxi_id INT PRIMARY KEY,
  Registration_no VARCHAR(20),
  Model VARCHAR(50),
  Manufactured_year INT,
  Taxi_type VARCHAR(20),
  Status VARCHAR(20),
  Owner_id INT
);

CREATE TABLE Owner (
  Owner_id INT PRIMARY KEY,
  SSN VARCHAR(20),
  Name VARCHAR(50),
  Company_id INT
);

CREATE TABLE Company (
  Company_id INT PRIMARY KEY,
  Tcs_id VARCHAR(20),
  Tsc_name VARCHAR(50)
);

CREATE TABLE Driver (
  Driver_id INT PRIMARY KEY,
  Name VARCHAR(50),
  Gender VARCHAR(10),
```



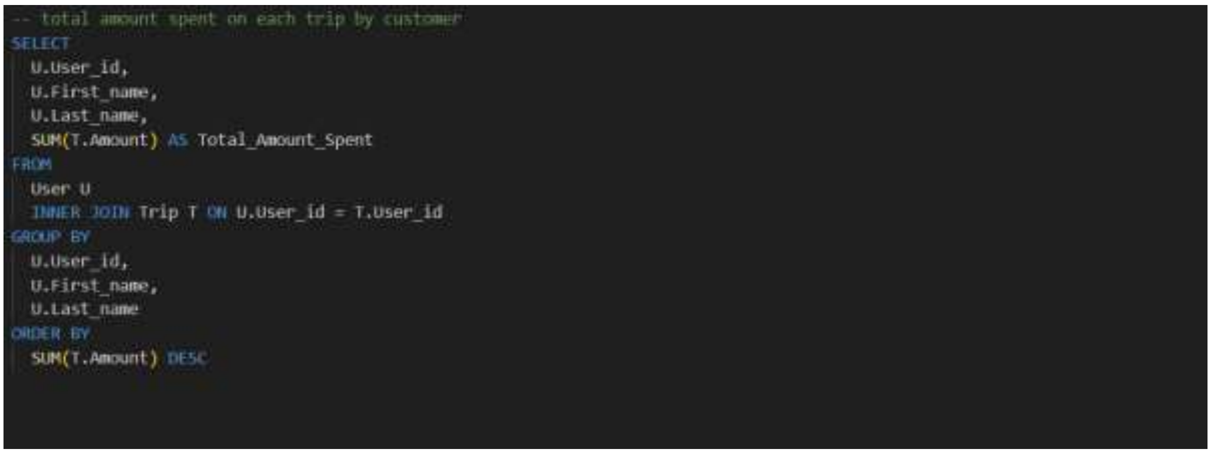
```sql
INSERT INTO Taxi (Taxi_id, Registration_no, Model, Manufactured_year, Taxi_type, Status, Owner_id)
VALUES
  (1, 'ABC123', 'Toyota Corolla', 2018, 'Economy', 'Available', 1),
  (2, 'DEF456', 'Honda Civic', 2019, 'Standard', 'Available', 2),
  (3, 'GHI789', 'Chevrolet Suburban', 2020, 'SUV', 'Unavailable', 3),
  (4, 'JKL012', 'BMW 5 Series', 2017, 'Premium', 'Available', 4),
  (5, 'MNO345', 'Chrysler Pacifica', 2021, 'Minivan', 'Available', 5),
  (6, 'PQR678', 'Ford Mustang', 2016, 'Sports', 'Unavailable', 6),
  (7, 'STU901', 'Nissan Altima', 2022, 'Standard', 'Available', 7),
  (8, 'VWX234', 'Tesla Model S', 2020, 'Premium', 'Unavailable', 8),
  (9, 'YZA567', 'Toyota Sienna', 2019, 'Minivan', 'Available', 9),
  (10, 'BCD890', 'Jeep Wrangler', 2021, 'SUV', 'Available', 10);

INSERT INTO Owner (Owner_id, SSN, Name, Company_id)
VALUES
  (1, '123-45-6789', 'John Smith', 1),
  (2, '234-56-7890', 'Jane Johnson', 1),
  (3, '345-67-8901', 'Bob Williams', 2),
  (4, '456-78-9012', 'Mary Lee', 2),
  (5, '567-89-0123', 'David Brown', 2),
  (6, '678-90-1234', 'Karen Garcia', 3),
  (7, '789-01-2345', 'Mike Davis', 3),
  (8, '890-12-3456', 'Sarah Wilson', 4),
  (9, '901-23-4567', 'Chris Taylor', 4),
  (10, '012-34-5678', 'Jenny Martinez', 5);

INSERT INTO Company (Company_id, Tcs_id, Tsc_name)
VALUES
  (1, 'TCS001', 'Taxi Service Company 1'),
  (2, 'TCS002', 'Taxi Service Company 2'),
  (3, 'TCS003', 'Taxi Service Company 3'),
  (4, 'TCS004', 'Taxi Service Company 4'),
  (5, 'TCS005', 'Taxi Service Company 5'),
  (6, 'TCS006', 'Taxi Service Company 6'),
  (7, 'TCS007', 'Taxi Service Company 7'),
```



```sql
INSERT INTO Login (User_id, Login_id, Password, Credit_card_no, Balance)
VALUES
  (1, 'john@example.com', 'password1', '1234567812345678', 100.00),
  (2, 'jane@example.com', 'password2', '2345678923456789', 200.00),
  (3, 'bob@example.com', 'password3', '3456789034567890', 300.00),
  (4, 'alice@example.com', 'password4', '4567890145678901', 400.00),
  (5, 'david@example.com', 'password5', '5678901256789012', 500.00),
  (6, 'sarah@example.com', 'password6', '6789012367890123', 600.00),
  (7, 'sam@example.com', 'password7', '7890123478901234', 700.00),
  (8, 'emily@example.com', 'password8', '8901234589012345', 800.00),
  (9, 'matt@example.com', 'password9', '9012345690123456', 900.00),
  (10, 'lisa@example.com', 'password10', '1234567812345678', 1000.00);


  SELECT * FROM User;
  SELECT name, gender FROM Driver WHERE rating >= 3;
  SELECT Bill_no, User_id, Amount FROM Bill WHERE Amount > 50.00;

  -- this fun calculates the total amount earned by the company and the driver.
  SELECT
  c.Tsc_name AS Company_Name,
  COUNT(t.Taxi_id) AS Total_Trips,
  SUM(tr.Amount) AS Total_Amount,
  SUM(tr.Amount * 0.8) AS Driver_Fee,
  SUM(tr.Amount * 0.2) AS Company_Fee
FROM
  Company c
  JOIN Owner o ON c.Company_id = o.Company_id
  JOIN Taxi t ON o.Owner_id = t.Owner_id
  JOIN Trip tr ON t.Taxi_id = tr.Taxi_id
GROUP BY
  c.Tsc_name;
```



```sql
  SUM(tr.Amount * 0.8) AS Driver_Fee,
  SUM(tr.Amount * 0.2) AS Company_Fee
FROM
  Company c
  JOIN Owner o ON c.Company_id = o.Company_id
  JOIN Taxi t ON o.Owner_id = t.Owner_id
  JOIN Trip tr ON t.Taxi_id = tr.Taxi_id
GROUP BY
  c.Tsc_name;

  -- to retrieve the total number of trips taken by each user, along with their name and contact number
  SELECT
  u.First_name || ' ' || u.Last_name AS User_name,
  u.Contact_no AS Contact_number,
  COUNT(t.Trip_id) AS Total_trips
FROM
  User u
  JOIN Trip t ON u.User_id = t.User_id
GROUP BY
  u.User_id;

  -- retrieves the total number of trips taken by each taxi, along with its registration number and model:
  SELECT
  t.Registration_no AS Taxi_registration,
  t.Model AS Taxi_model,
  COUNT(tr.Trip_id) AS Total_trips
FROM
  Taxi t
  JOIN Trip tr ON t.Taxi_id = tr.Taxi_id
GROUP BY
```



```sql
-- total amount spent on each trip by customer
SELECT
  U.User_id,
  U.First_name,
  U.Last_name,
  SUM(T.Amount) AS Total_Amount_Spent
FROM
  User U
  INNER JOIN Trip T ON U.User_id = T.User_id
GROUP BY
  U.User_id,
  U.First_name,
  U.Last_name
ORDER BY
  SUM(T.Amount) DESC
```

```
CREATE TABLE User (
  User_id INT PRIMARY KEY,
  First_name VARCHAR(50),
  Last_name VARCHAR(50),
  Address VARCHAR(100),
  Age INT,
  Contact_no VARCHAR(20),
  Email VARCHAR(100)
);

CREATE TABLE Taxi (
  Taxi_id INT PRIMARY KEY,
  Registration_no VARCHAR(20),
  Model VARCHAR(50),
  Manufactured_year INT,
  Taxi_type VARCHAR(20),
  Status VARCHAR(20),
  Owner_id INT
);

CREATE TABLE Owner (
  Owner_id INT PRIMARY KEY,
  SSN VARCHAR(20),
  Name VARCHAR(50),
  Company_id INT
);

CREATE TABLE Company (
  Company_id INT PRIMARY KEY,
  Tcs_id VARCHAR(20),
  Tsc_name VARCHAR(50)
);

CREATE TABLE Driver (
  Driver_id INT PRIMARY KEY,
  Name VARCHAR(50),
  Gender VARCHAR(10),
  Contact_no VARCHAR(20),
  Rating INT,
  Age INT
);

CREATE TABLE Trip (
  Trip_id INT PRIMARY KEY,
  User_id INT,
  Taxi_id INT,
  Start_time DATETIME,
  End_time DATETIME,
  Amount DECIMAL(10,2),
  Promotional_code VARCHAR(20),
  Feedback VARCHAR(200),
  Driver_id INT
);

CREATE TABLE Bill (
  Bill_no INT PRIMARY KEY,
  User_id INT,
  Driver_id INT,
  Amount DECIMAL(10,2),
  Date DATETIME
);
```

```
CREATE TABLE Login (
  User_id INT PRIMARY KEY,
  Login_id VARCHAR(50),
  Password VARCHAR(50),
  Credit_card_no VARCHAR(16),
  Balance DECIMAL(10,2)
);
INSERT INTO User (User_id, First_name, Last_name, Address, Age, Contact_no, Email)
VALUES
  (1, 'John', 'Doe', '123 Main St', 30, '555-555-5555', 'johndoe@email.com'),
  (2, 'Jane', 'Smith', '456 Elm St', 25, '555-555-1234', 'janesmith@email.com'),
  (3, 'Bob', 'Johnson', '789 Oak St', 40, '555-555-6789', 'bobjohnson@email.com'),
  (4, 'Mary', 'Williams', '321 Maple St', 35, '555-555-4321', 'marywilliams@email.com'),
  (5, 'David', 'Lee', '654 Pine St', 28, '555-555-9876', 'davidlee@email.com'),
  (6, 'Karen', 'Taylor', '987 Oak St', 50, '555-555-1111', 'karentaylor@email.com'),
  (7, 'Mike', 'Brown', '456 Pine St', 45, '555-555-2222', 'mikebrown@email.com'),
  (8, 'Sarah', 'Davis', '789 Maple St', 30, '555-555-3333', 'sarahdavis@email.com'),
  (9, 'Chris', 'Wilson', '123 Elm St', 35, '555-555-4444', 'chriswilson@email.com'),
  (10, 'Jenny', 'Garcia', '321 Oak St', 25, '555-555-5555', 'jennygarcia@email.com');

INSERT INTO Taxi (Taxi_id, Registration_no, Model, Manufactured_year, Taxi_type, Status, Owner_id)
VALUES
  (1, 'ABC123', 'Toyota Corolla', 2018, 'Economy', 'Available', 1),
  (2, 'DEF456', 'Honda Civic', 2019, 'Standard', 'Available', 2),
  (3, 'GHI789', 'Chevrolet Suburban', 2020, 'SUV', 'Unavailable', 3),
  (4, 'JKL012', 'BMW 5 Series', 2017, 'Premium', 'Available', 4),
  (5, 'MNO345', 'Chrysler Pacifica', 2021, 'Minivan', 'Available', 5),
  (6, 'PQR678', 'Ford Mustang', 2016, 'Sports', 'Unavailable', 6),
  (7, 'STU901', 'Nissan Altima', 2022, 'Standard', 'Available', 7),
  (8, 'VWX234', 'Tesla Model S', 2020, 'Premium', 'Unavailable', 8),
  (9, 'YZA567', 'Toyota Sienna', 2019, 'Minivan', 'Available', 9),
  (10, 'BCD890', 'Jeep Wrangler', 2021, 'SUV', 'Available', 10);

INSERT INTO Owner (Owner_id, SSN, Name, Company_id)
VALUES
  (1, '123-45-6789', 'John Smith', 1),
  (2, '234-56-7890', 'Jane Johnson', 1),
  (3, '345-67-8901', 'Bob Williams', 2),
  (4, '456-78-9012', 'Mary Lee', 2),
  (5, '567-89-0123', 'David Brown', 2),
  (6, '678-90-1234', 'Karen Garcia', 3),
  (7, '789-01-2345', 'Mike Davis', 3),
  (8, '890-12-3456', 'Sarah Wilson', 4),
  (9, '901-23-4567', 'Chris Taylor', 4),
  (10, '012-34-5678', 'Jenny Martinez', 5);

INSERT INTO Company (Company_id, Tcs_id, Tsc_name)
VALUES
  (1, 'TCS001', 'Taxi Service Company 1'),
  (2, 'TCS002', 'Taxi Service Company 2'),
  (3, 'TCS003', 'Taxi Service Company 3'),
  (4, 'TCS004', 'Taxi Service Company 4'),
  (5, 'TCS005', 'Taxi Service Company 5'),
  (6, 'TCS006', 'Taxi Service Company 6'),
  (7, 'TCS007', 'Taxi Service Company 7'),
  (8, 'TCS008', 'Taxi Service Company 8'),
  (9, 'TCS009', 'Taxi Service Company 9'),
  (10, 'TCS010', 'Taxi Service Company 10');

INSERT INTO Driver (Driver_id, Name, Gender, Contact_no, Rating, Age)
VALUES
  (1, 'John Doe', 'M', '555-555-5555', 4, 30),
  (2, 'Jane Smith', 'F', '555-555-1234', 5, 25),
  (3, 'Bob Johnson', 'M', '555-555-6789', 3, 40),
  (4, 'Mary Williams', 'F', '555-555-4321', 4, 35),
  (5, 'David Lee', 'M', '555-555-9876', 5, 28),
  (6, 'Karen Taylor', 'F', '555-555-1111', 4, 50),
  (7, 'Mike Brown', 'M', '555-555-2222', 3, 45),
  (8, 'Sarah Davis', 'F', '555-555-3333', 4, 30),
  (9, 'Chris Wilson', 'M', '555-555-4444', 3, 35),
  (10, 'Jenny Garcia', 'F', '555-555-5555', 5, 25);



  -- This code inserts a new trip into the Trip table
INSERT INTO Trip (Trip_id, User_id, Taxi_id, Start_time, End_time, Amount, Promotional_code,
Feedback, Driver_id)
VALUES
  (1, 1, 1, '2023-07-16 12:00:00', '2023-07-16 12:30:00', 25.50, 'SUMMER2023', 'Great ride!', 1),
  (2, 2, 2, '2023-07-16 13:00:00', '2023-07-16 13:30:00', 30.00, 'JULY2023', 'Driver was friendly', 2),
  (3, 3, 3, '2023-07-16 14:00:00', '2023-07-16 14:30:00', 40.25, '', '', 3),
  (4, 4, 4, '2023-07-16 15:00:00', '2023-07-16 15:30:00', 55.75, 'LOYALTY2023', 'Driver was on time',
4),
  (5, 5, 5, '2023-07-16 16:00:00', '2023-07-16 16:30:00', 20.00, '', '', 5),
  (6, 6, 6, '2023-07-16 17:00:00', '2023-07-16 17:30:00', 50.00, '', '', 6),
  (7, 7, 7, '2023-07-16 18:00:00', '2023-07-16 18:30:00', 35.50, 'DISCOUNT2023', 'Driver was
professional', 7),
  (8, 8, 8, '2023-07-16 19:00:00', '2023-07-16 19:30:00', 75.25, '', '', 8),
  (9, 9, 9, '2023-07-16 20:00:00', '2023-07-16 20:30:00', 22.50, '', '', 9),
  (10, 10, 10, '2023-07-16 21:00:00', '2023-07-16 21:30:00', 60.00, '', '', 10);


INSERT INTO Bill (Bill_no, User_id, Driver_id, Amount, Date)
VALUES
  (1, 1, 1, 25.50, '2023-07-16 12:30:00'),
  (2, 2, 2, 30.00, '2023-07-16 13:30:00'),
  (3, 3, 3, 40.25, '2023-07-16 14:30:00'),
  (4, 4, 4, 55.75, '2023-07-16 15:30:00'),
  (5, 5, 5, 20.00, '2023-07-16 16:30:00'),
  (6, 6, 6, 50.00, '2023-07-16 17:30:00'),
  (7, 7, 7, 35.50, '2023-07-16 18:30:00'),
  (8, 8, 8, 75.25, '2023-07-16 19:30:00'),
  (9, 9, 9, 22.50, '2023-07-16 20:30:00'),
  (10, 10, 10, 60.00, '2023-07-16 21:30:00');
INSERT INTO Login (User_id, Login_id, Password, Credit_card_no, Balance)
VALUES
  (1, 'john@example.com', 'password1', '1234567812345678', 100.00),
  (2, 'jane@example.com', 'password2', '2345678923456789', 200.00),
  (3, 'bob@example.com', 'password3', '3456789034567890', 300.00),
  (4, 'alice@example.com', 'password4', '4567890145678901', 400.00),
  (5, 'david@example.com', 'password5', '5678901256789012', 500.00),
  (6, 'sarah@example.com', 'password6', '6789012367890123', 600.00),
  (7, 'sam@example.com', 'password7', '7890123478901234', 700.00),
  (8, 'emily@example.com', 'password8', '8901234589012345', 800.00),
  (9, 'matt@example.com', 'password9', '9012345690123456', 900.00),
  (10, 'lisa@example.com', 'password10', '1234567812345678', 1000.00);


  SELECT * FROM User;
  SELECT name, gender FROM Driver WHERE rating >= 3;
  SELECT Bill_no, User_id, Amount FROM Bill WHERE Amount > 50.00;

  -- this fun calculates the total amount earned by the company and the driver.
  SELECT
  c.Tsc_name AS Company_Name,
  COUNT(t.Taxi_id) AS Total_Trips,
  SUM(tr.Amount) AS Total_Amount,
  SUM(tr.Amount * 0.8) AS Driver_Fee,

  SUM(tr.Amount * 0.2) AS Company_Fee

FROM

  Company c

  JOIN Owner o ON c.Company_id = o.Company_id

  JOIN Taxi t ON o.Owner_id = t.Owner_id

  JOIN Trip tr ON t.Taxi_id = tr.Taxi_id

GROUP BY

  c.Tsc_name;


  -- to retrieve the total number of trips taken by each user, along with their name and contact
number

  SELECT

  u.First_name || ' ' || u.Last_name AS User_name,

  u.Contact_no AS Contact_number,

  COUNT(t.Trip_id) AS Total_trips

FROM

  User u

  JOIN Trip t ON u.User_id = t.User_id

GROUP BY

  u.User_id;


  -- retrieves the total number of trips taken by each taxi, along with its registration number and
model:

  SELECT

  t.Registration_no AS Taxi_registration,

  t.Model AS Taxi_model,

  COUNT(tr.Trip_id) AS Total_trips

FROM

  Taxi t

  JOIN Trip tr ON t.Taxi_id = tr.Taxi_id

GROUP BY
```

```
  t.Taxi_id;


-- Generate bill for a specific trip

INSERT INTO Bill (User_id, Driver_id, Amount, Date)

SELECT

  t.User_id,

  t.Driver_id,

  t.Amount * 0.8,

  NOW() AS Date

FROM

  Trip t

WHERE

  t.Trip_id = 1234;



SELECT * FROM Bill WHERE User_id = 5678 AND Driver_id = 9012 AND Date = (SELECT MAX(Date)
FROM Bill);


-- total amount spent on each trip by customer

SELECT

  U.User_id,

  U.First_name,

  U.Last_name,

  SUM(T.Amount) AS Total_Amount_Spent

FROM

  User U

  INNER JOIN Trip T ON U.User_id = T.User_id

GROUP BY

  U.User_id,

  U.First_name,

  U.Last_name
```

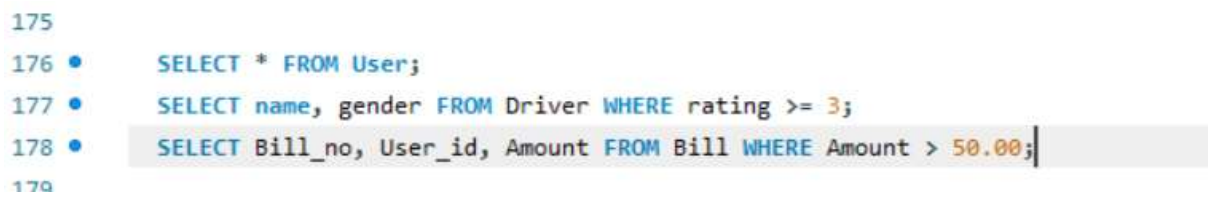
```
ORDER BY
 SUM(T.Amount) DESC
```

Code Explanation along with its output :

- An SQL script in the provided code produces several tables in a relational database. The attributes in each table, which each represent a different entity, explain the traits of that thing.
- Attributes like User_id, First_name, Last_name, Address, Age, Contact_no, and Email are contained in the "User" table. Users of a taxi service can have their information stored in this table.
- Taxi_id, Registration_no, Model, Manufactured_year, Taxi_type, Status, and Owner_id are some of the elements in the "Taxi" table. Information on taxis, including their owners, can be kept in this table.
- Attributes in the "Owner" table include Owner_id, SSN, Name, and Company_id. Information on the taxi proprietors can be kept in this table.
- Attributes like Company_id, Tcs_id, and Tsc_name are contained in the "Company" table.
- This is an SQL query that inserts data into a table called "Login". The table has columns named "User_id", "Login_id", "Password", "Credit_card_no", and "Balance".

- The values being inserted into the table are:

– - For the first row: User_id is 1, Login_id is 'john@example.com', Password is 'password1', Credit_card_no is '1234567812345678', and Balance is 100.00.
– - For the second row: User_id is 2, Login_id is 'jane@example.com', Password is 'password2', Credit_card_no is '2345678923456789', and Balance is 200.00.
– - For the third row: User_id is 3, Login_id is 'bob@example.com', Password is 'password3', Credit_card_no is '3456789034567890', and Balance is 300.00.
– - For the fourth row: User_id is 4, Login_id is 'alice@example.com', Password is 'password4', Credit_card_no is '4567890145678901', and Balance is 400.00.
– - For the fifth row: User_id is 5, Login_id is 'david@example.com', Password is 'password5', Credit_card_no is '5678901256789012', and Balance is 500.00.
– - For the sixth row: User_id is 6, Login_id is 'sarah@example.com', Password is 'password6', Credit_card_no is '6789012367890123', and Balance is 600.00.
– - For the seventh row: User_id is 7, Login_id is 'sam@example.com', Password is 'password7', Credit_card_no is '7890123478901234', and Balance is 700.00.
– - For the eighth row: User_id is 8, Login_id is 'emily@example.com', Password is 'password8', Credit_card_no is '8901234589012345', and Balance is 800.00.
– - For the ninth row: User_id is 9, Login_id is 'matt@example.com', Password is 'password9', Credit_card_no is '9012345690123456', and Balance is 900.00.
– - For the tenth row: User_id is 10, Login_id is 'lisa@example.com', Password is 'password10', Credit_card_no is '1234567812345678', and Balance is 1000.00.



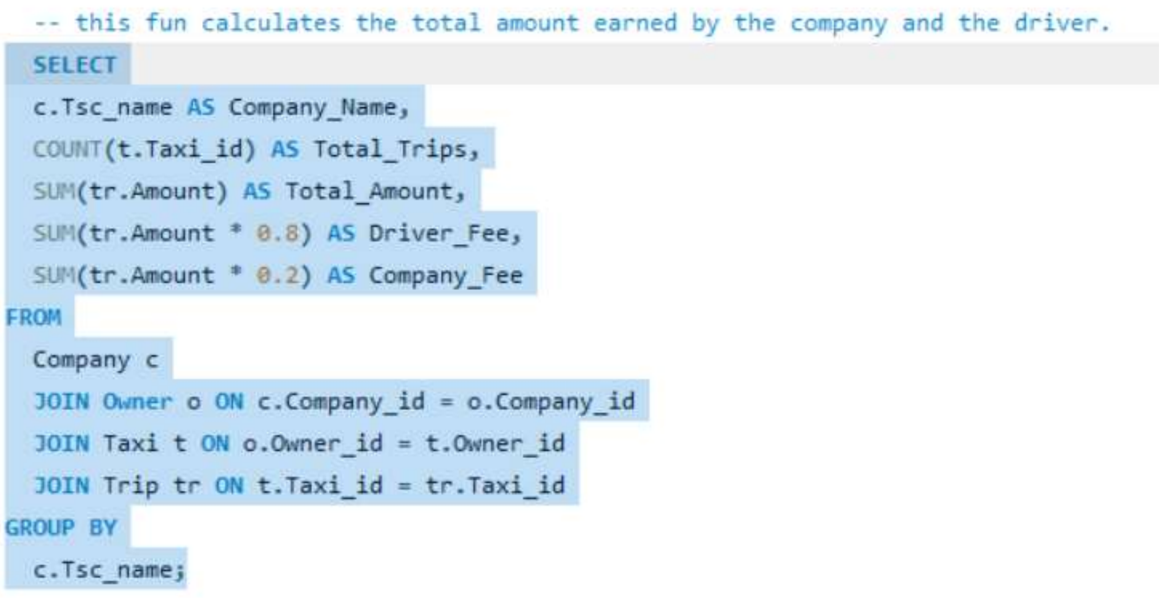
```sql
SELECT * FROM User;
SELECT name, gender FROM Driver WHERE rating >= 3;
SELECT Bill_no, User_id, Amount FROM Bill WHERE Amount > 50.00;
```

These are three separate SQL queries:

1. "SELECT * FROM User": This query selects all columns from a table called "User". It will return all rows and columns from that table.

2. "SELECT name, gender FROM Driver WHERE rating >= 3": This query selects only the "name" and "gender" columns from a table called "Driver" where the "rating" column is greater than or equal to 3. It will return only the rows that meet this condition.

3. "SELECT Bill_no, User_id, Amount FROM Bill WHERE Amount > 50.00": This query selects the "Bill_no", "User_id", and "Amount" columns from a table called "Bill" where the "Amount" column is greater than 50.00. It will return only the rows that meet this condition.



```sql
-- this fun calculates the total amount earned by the company and the driver.
SELECT
  c.Tsc_name AS Company_Name,
  COUNT(t.Taxi_id) AS Total_Trips,
  SUM(tr.Amount) AS Total_Amount,
  SUM(tr.Amount * 0.8) AS Driver_Fee,
  SUM(tr.Amount * 0.2) AS Company_Fee
FROM
  Company c
  JOIN Owner o ON c.Company_id = o.Company_id
  JOIN Taxi t ON o.Owner_id = t.Owner_id
  JOIN Trip tr ON t.Taxi_id = tr.Taxi_id
GROUP BY
  c.Tsc_name;
```

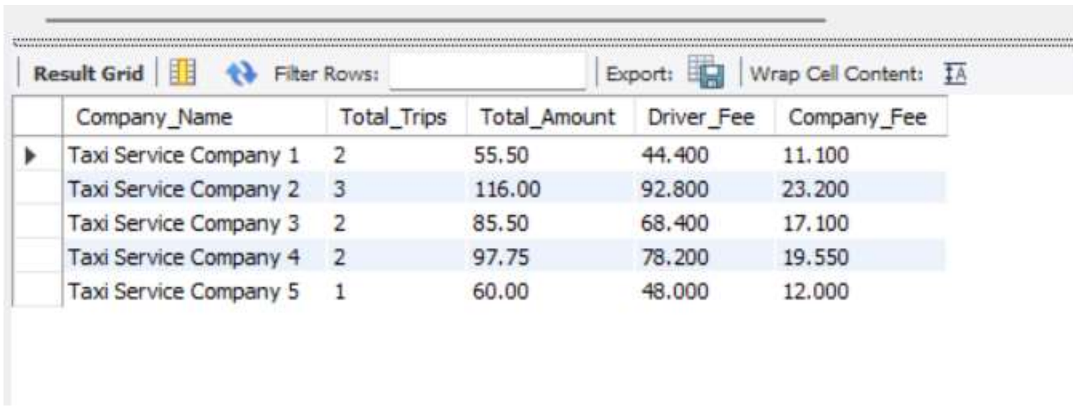
This is an SQL query that retrieves data from several tables related to taxi trips and fees, and groups the results by company name. Here is a breakdown of the query:

- The SELECT clause specifies the columns to retrieve and some computed values. The alias "Company_Name" is used for the "Tsc_name" column of the "Company" table. "Total_Trips" is the count of all taxi trips for the company. "Total_Amount" is the sum of all trip amounts for the company. "Driver_Fee" is 80% of the total trip amount for the company, representing the fee paid to the driver. "Company_Fee" is 20% of the total trip amount for the company, representing the fee paid to the company.

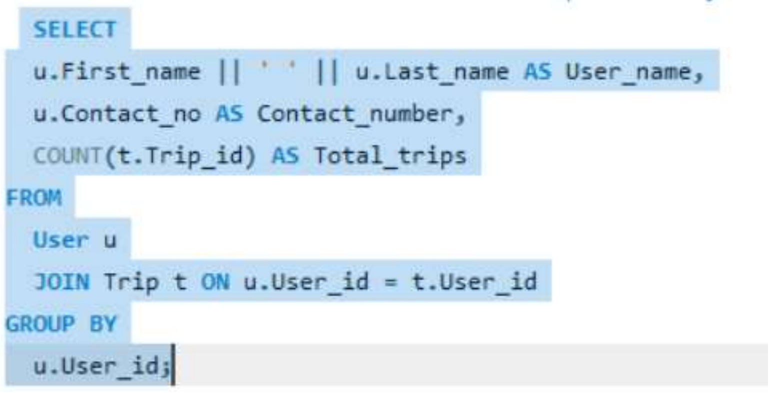
- The FROM clause specifies the tables to use for the query and the relationships between them. Four tables are joined: "Company", "Owner", "Taxi", and "Trip". The "Company" and "Owner" tables are joined on their respective ID columns. The "Owner" and "Taxi" tables are joined on their respective ID columns. The "Taxi" and "Trip" tables are joined on their respective ID columns.

- The GROUP BY clause groups the result set by the company name, which is the alias used for the "Tsc_name" column of the "Company" table. This means that the query will return one row per company, with the computed values aggregated over all taxi trips for that company.

Overall, this query computes the total number of trips, the total amount, and the respective fees paid to the driver and the company for each taxi company in the dataset.



Company_Name	Total_Trips	Total_Amount	Driver_Fee	Company_Fee
Taxi Service Company 1	2	55.50	44.400	11.100
Taxi Service Company 2	3	116.00	92.800	23.200
Taxi Service Company 3	2	85.50	68.400	17.100
Taxi Service Company 4	2	97.75	78.200	19.550
Taxi Service Company 5	1	60.00	48.000	12.000

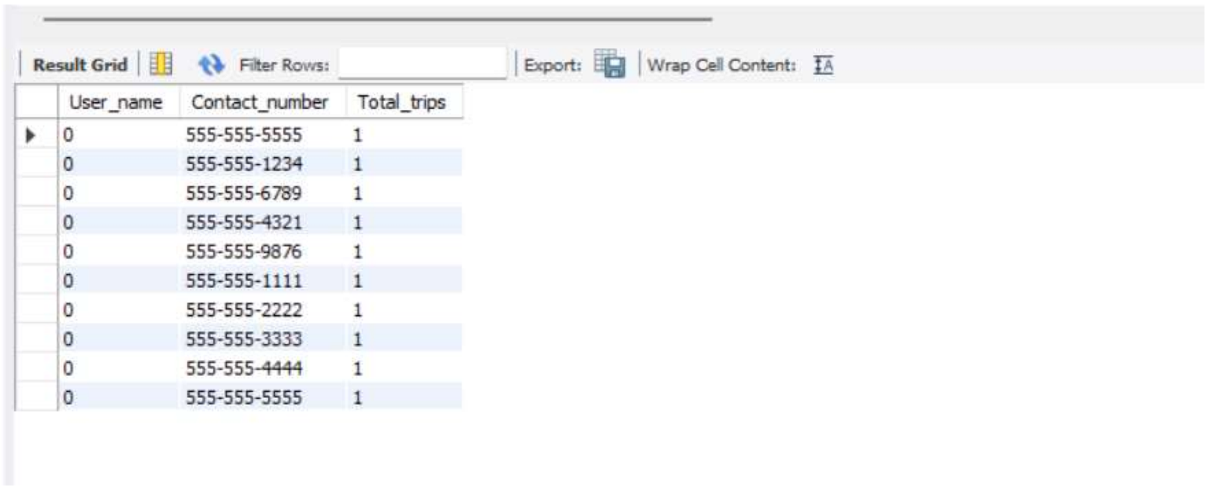

```
SELECT
  u.First_name || ' ' || u.Last_name AS User_name,
  u.Contact_no AS Contact_number,
  COUNT(t.Trip_id) AS Total_trips
FROM
  User u
  JOIN Trip t ON u.User_id = t.User_id
GROUP BY
  u.User_id;
```

This is an SQL query that retrieves data from two tables related to taxi trips and users, and groups the results by user. Here is a breakdown of the query:

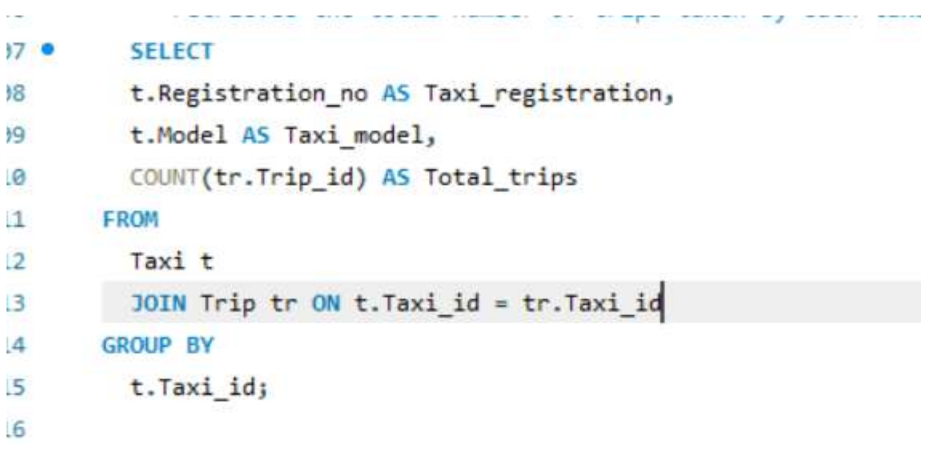
- The SELECT clause specifies the columns to retrieve and some computed values. The "User_name" column is a concatenation of the "First_name" and "Last_name" columns of the "User" table, separated by a space. The "Contact_number" column is the "Contact_no" column of the "User" table. "Total_trips" is the count of all taxi trips for the user.

- The FROM clause specifies the tables to use for the query and the relationship between them. Two tables are joined: "User" and "Trip". The "User" and "Trip" tables are joined on their respective ID columns.

- The GROUP BY clause groups the result set by the user ID, which is the unique identifier of each user in the "User" table. This means that the query will return one row per user, with the computed values aggregated over all taxi trips for that user.



User_name	Contact_number	Total_trips
0	555-555-5555	1
0	555-555-1234	1
0	555-555-6789	1
0	555-555-4321	1
0	555-555-9876	1
0	555-555-1111	1
0	555-555-2222	1
0	555-555-3333	1
0	555-555-4444	1
0	555-555-5555	1



```sql
  SELECT
  t.Registration_no AS Taxi_registration,
  t.Model AS Taxi_model,
  COUNT(tr.Trip_id) AS Total_trips
FROM
  Taxi t
  JOIN Trip tr ON t.Taxi_id = tr.Taxi_id
GROUP BY
  t.Taxi_id;
```

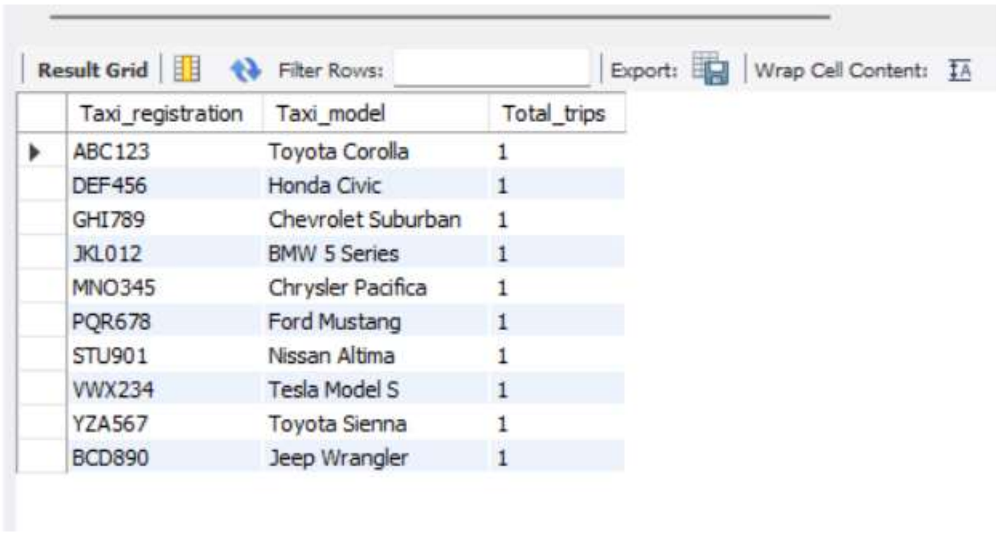
This is an SQL query that retrieves data from two tables related to taxi trips and taxis, and groups the results by taxi. Here is a breakdown of the query:

- The SELECT clause specifies the columns to retrieve and some computed values. The "Taxi_registration" column is the "Registration_no" column of the "Taxi" table. The "Taxi_model" column is the "Model" column of the "Taxi" table. "Total_trips" is the count of all taxi trips made with the taxi.

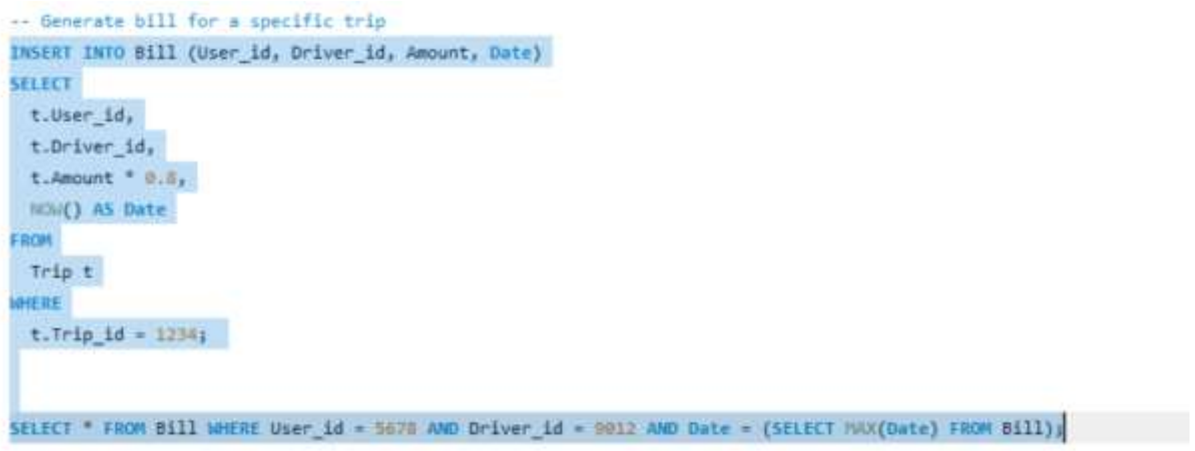
- The FROM clause specifies the tables to use for the query and the relationship between them. Two tables are joined: "Taxi" and "Trip". The "Taxi" and "Trip" tables are joined on their respective ID columns.

- The GROUP BY clause groups the result set by the taxi ID, which is the unique identifier of each taxi in the "Taxi" table. This means that the query will return one row per taxi, with the computed values aggregated over all taxi trips made with that taxi.

Overall, this query computes the total number of trips made by each taxi in the dataset, along with its registration number and model. By grouping the results by taxi, the query provides a summary of each taxi's activity in the system.



Taxi_registration	Taxi_model	Total_trips
ABC123	Toyota Corolla	1
DEF456	Honda Civic	1
GHI789	Chevrolet Suburban	1
JKL012	BMW 5 Series	1
MNO345	Chrysler Pacifica	1
PQR678	Ford Mustang	1
STU901	Nissan Altima	1
VWX234	Tesla Model S	1
YZA567	Toyota Sienna	1
BCD890	Jeep Wrangler	1



```sql
-- Generate bill for a specific trip
INSERT INTO Bill (User_id, Driver_id, Amount, Date)
SELECT
  t.User_id,
  t.Driver_id,
  t.Amount * 0.8,
  NOW() AS Date
FROM
  Trip t
WHERE
  t.Trip_id = 1234;


SELECT * FROM Bill WHERE User_id = 5678 AND Driver_id = 9012 AND Date = (SELECT MAX(Date) FROM Bill);
```

This is an SQL query that inserts a new row into the "Bill" table, using data from the "Trip" table. Here is a breakdown of the query:

- The INSERT INTO clause specifies the table to insert data into, which is the "Bill" table. The columns to insert data into are not explicitly specified, but they are assumed to be all columns of the table.

- The SELECT clause specifies the data to insert into the table. The "User_id" column is the "User_id" column of the "Trip" table. The "Driver_id" column is the "Driver_id" column of the "Trip" table. The "Amount" column is 80% of the "Amount" column of the "Trip" table, which represents the fee paid to the driver. The "Date" column is set to the current date and time using the NOW() function.

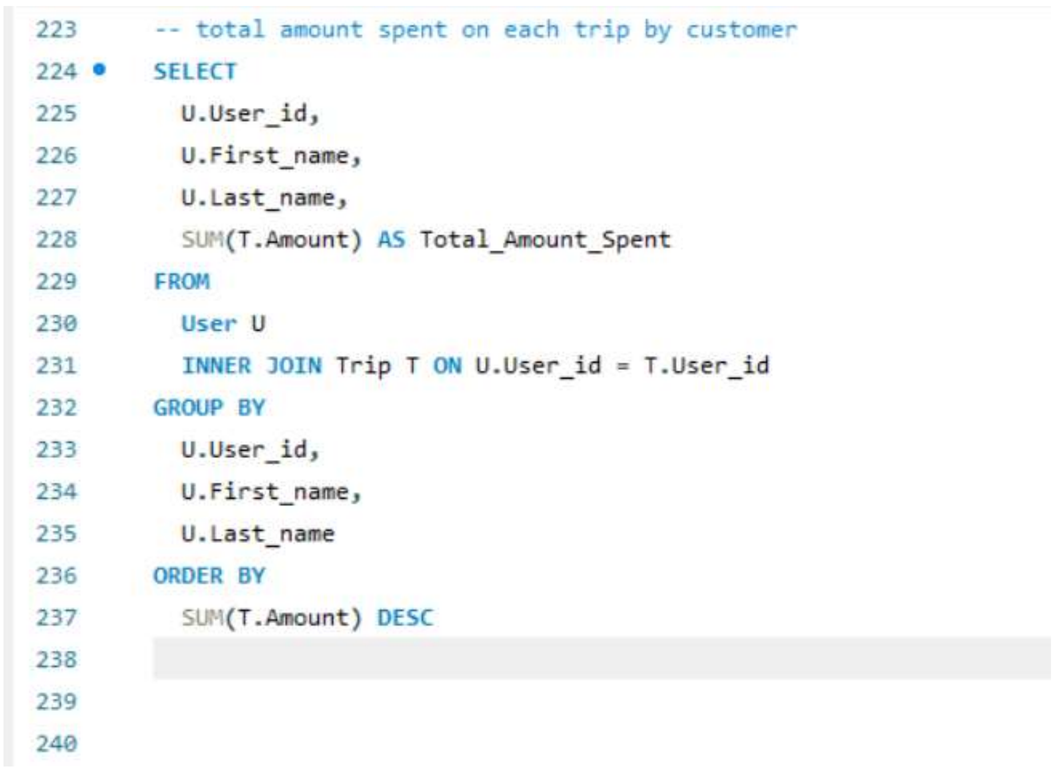
- The WHERE clause specifies which row of the "Trip" table to use for the data insertion, based on the "Trip_id" column. In this case, only the trip with ID 1234 will be considered for the data insertion.

The second query:

- The SELECT clause retrieves all columns from the "Bill" table.

- The WHERE clause specifies the conditions for the rows to retrieve. In this case, only the rows where "User_id" is 5678, "Driver_id" is 9012, and "Date" is the maximum date in the table will be retrieved. This means that the query will return the most recent bill for the specified user and driver.

Overall, these two queries work together to insert a new bill into the system for a specific trip, and then retrieve the most recent bill for a specified user and driver.



```sql
-- total amount spent on each trip by customer
SELECT
  U.User_id,
  U.First_name,
  U.Last_name,
  SUM(T.Amount) AS Total_Amount_Spent
FROM
  User U
  INNER JOIN Trip T ON U.User_id = T.User_id
GROUP BY
  U.User_id,
  U.First_name,
  U.Last_name
ORDER BY
  SUM(T.Amount) DESC
```

- The query retrieves data from two tables: "User" and "Trip".

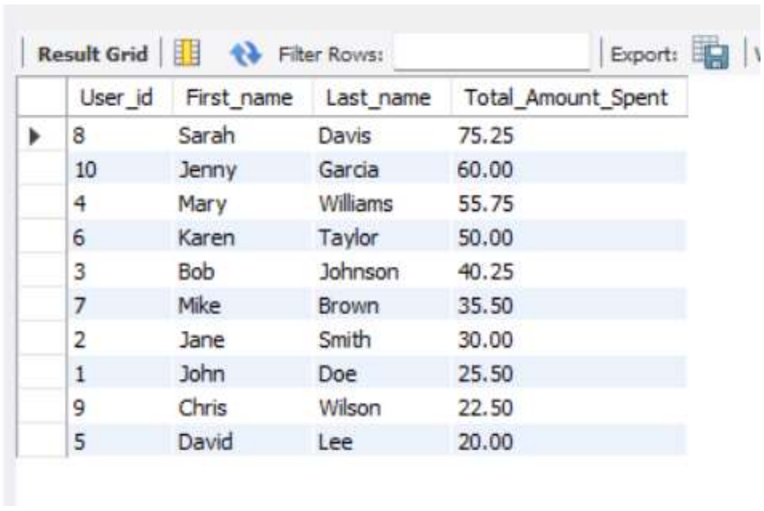
- It joins the tables on their respective ID columns.

- The query groups the result set by user ID, first name, and last name.

- It computes the total amount spent on taxi trips by each user.

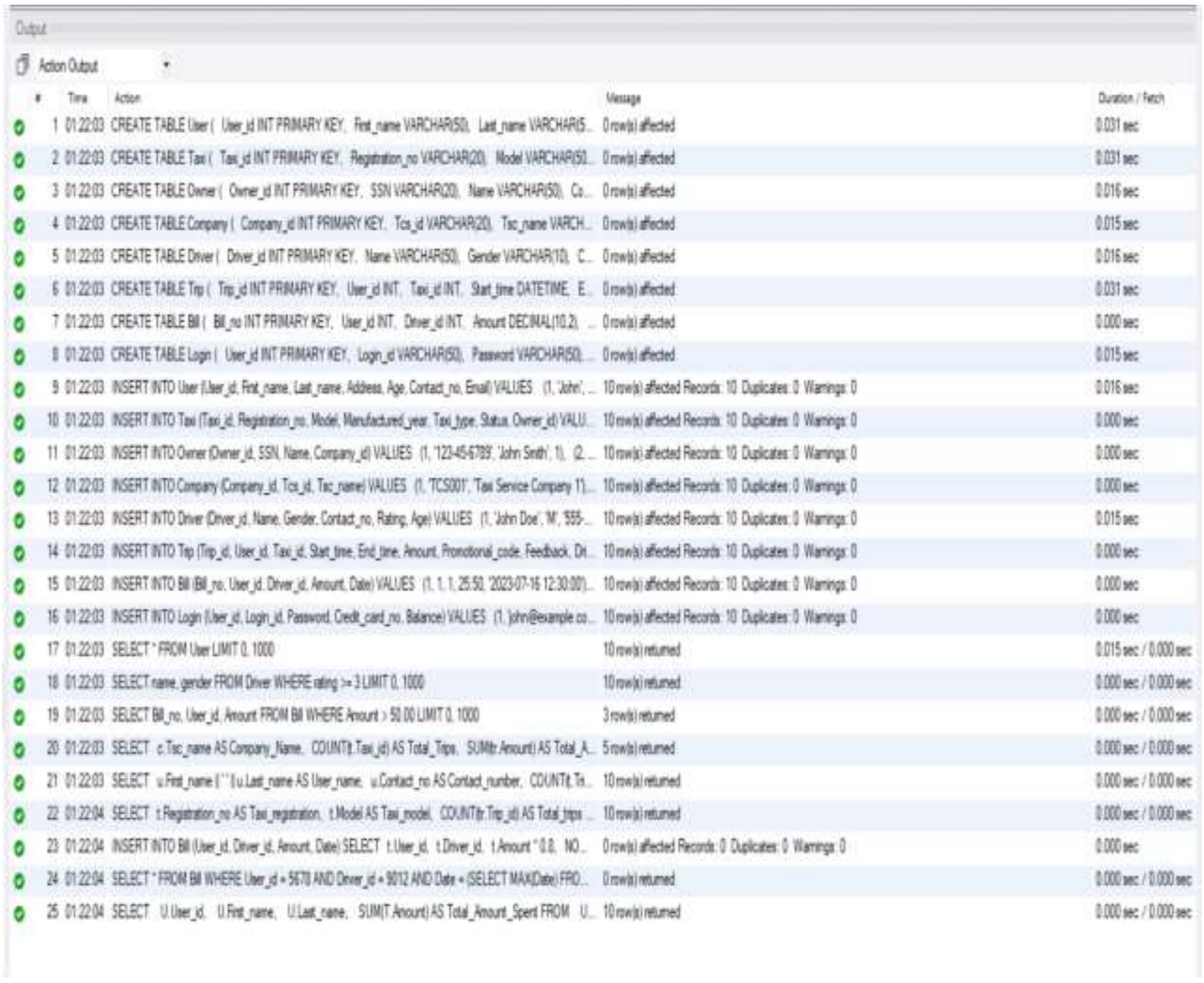
- The results are sorted in descending order of the total amount spent.

- The query returns one row per user, with the computed values aggregated over all taxi trips for that user.

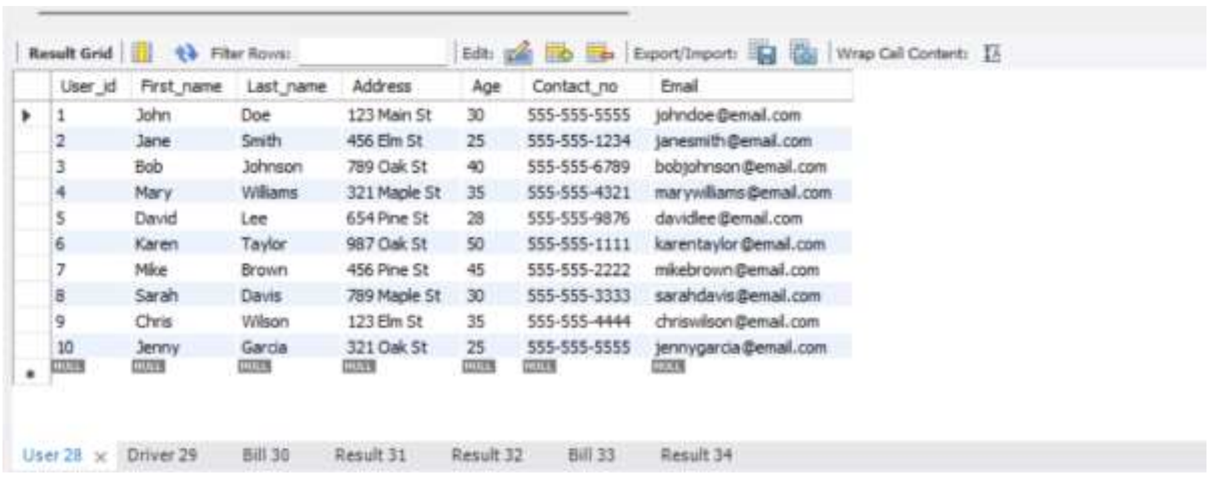


User_id	First_name	Last_name	Total_Amount_Spent
8	Sarah	Davis	75.25
10	Jenny	Garcia	60.00
4	Mary	Williams	55.75
6	Karen	Taylor	50.00
3	Bob	Johnson	40.25
7	Mike	Brown	35.50
2	Jane	Smith	30.00
1	John	Doe	25.50
9	Chris	Wilson	22.50
5	David	Lee	20.00

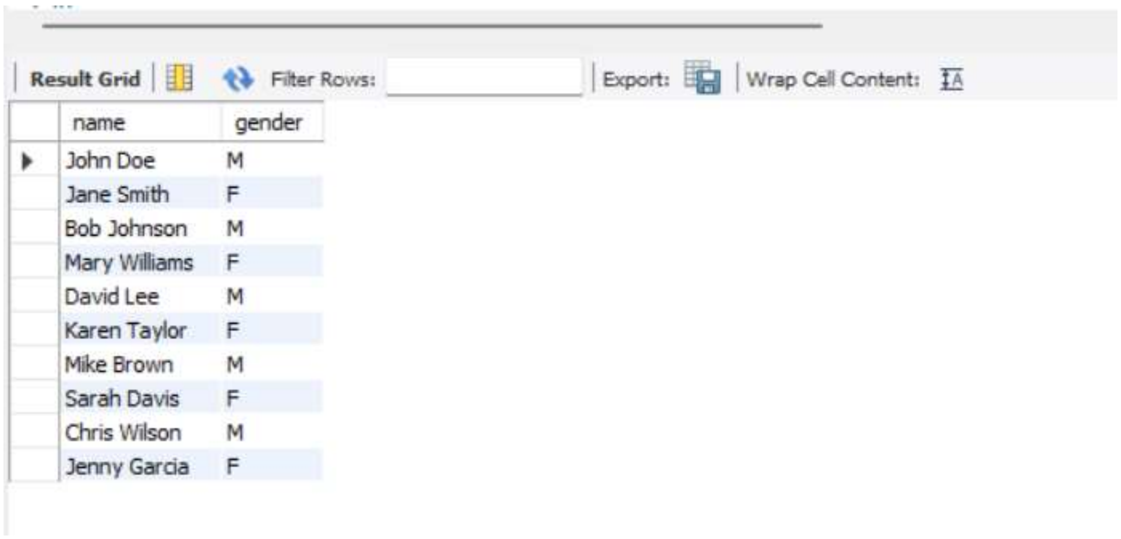
<u>Output snippets</u> :



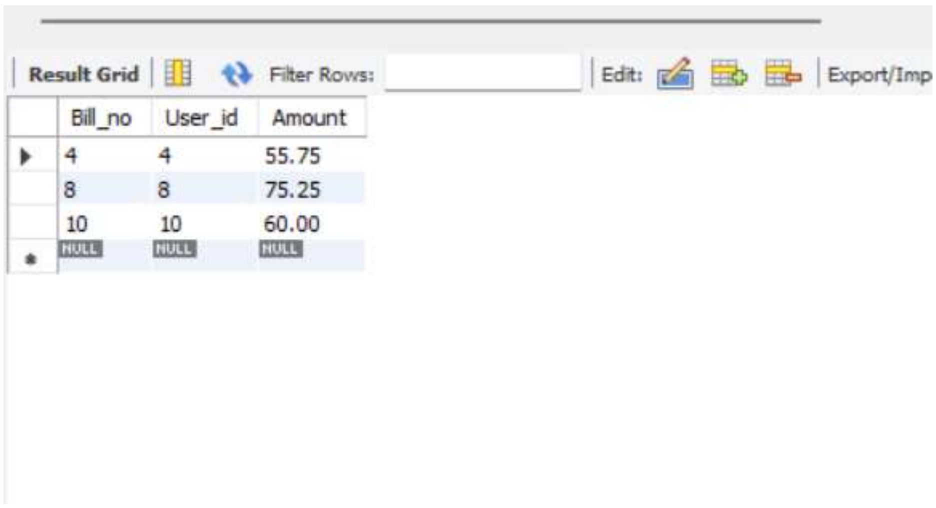
```
#  Time      Action                                                                                                  Message                                                    Duration / Fetch
1  01:22:03  CREATE TABLE User (  User_id INT PRIMARY KEY,  First_name VARCHAR(50),  Last_name VARCHAR(5...          0 row(s) affected                                          0.031 sec
2  01:22:03  CREATE TABLE Taxi (  Taxi_id INT PRIMARY KEY,  Registration_no VARCHAR(20),  Model VARCHAR(50...        0 row(s) affected                                          0.031 sec
3  01:22:03  CREATE TABLE Owner (  Owner_id INT PRIMARY KEY,  SSN VARCHAR(20),  Name VARCHAR(50),  Co...            0 row(s) affected                                          0.016 sec
4  01:22:03  CREATE TABLE Company (  Company_id INT PRIMARY KEY,  Tcs_id VARCHAR(20),  Tsc_name VARCH...            0 row(s) affected                                          0.015 sec
5  01:22:03  CREATE TABLE Driver (  Driver_id INT PRIMARY KEY,  Name VARCHAR(50),  Gender VARCHAR(10),  C...        0 row(s) affected                                          0.016 sec
6  01:22:03  CREATE TABLE Trip (  Trip_id INT PRIMARY KEY,  User_id INT,  Taxi_id INT,  Start_time DATETIME,  E...   0 row(s) affected                                          0.031 sec
7  01:22:03  CREATE TABLE Bill (  Bill_no INT PRIMARY KEY,  User_id INT,  Driver_id INT,  Amount DECIMAL(10,2),  ... 0 row(s) affected                                          0.000 sec
8  01:22:03  CREATE TABLE Login (  User_id INT PRIMARY KEY,  Login_id VARCHAR(50),  Password VARCHAR(50), ...        0 row(s) affected                                          0.015 sec
9  01:22:03  INSERT INTO User (User_id, First_name, Last_name, Address, Age, Contact_no, Email) VALUES  (1, 'John', ... 10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0   0.016 sec
10 01:22:03  INSERT INTO Taxi (Taxi_id, Registration_no, Model, Manufactured_year, Taxi_type, Status, Owner_id) VALU... 10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0  0.000 sec
11 01:22:03  INSERT INTO Owner (Owner_id, SSN, Name, Company_id) VALUES  (1, '123-45-6789', 'John Smith', 1),  (2, ... 10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0   0.000 sec
12 01:22:03  INSERT INTO Company (Company_id, Tcs_id, Tsc_name) VALUES  (1, 'TCS001', 'Taxi Service Company 1')... 10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0    0.000 sec
13 01:22:03  INSERT INTO Driver (Driver_id, Name, Gender, Contact_no, Rating, Age) VALUES  (1, 'John Doe', 'M', '555-... 10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0  0.015 sec
14 01:22:03  INSERT INTO Trip (Trip_id, User_id, Taxi_id, Start_time, End_time, Amount, Promotional_code, Feedback, Dri... 10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0  0.000 sec
15 01:22:03  INSERT INTO Bill (Bill_no, User_id, Driver_id, Amount, Date) VALUES  (1, 1, 1, 25.50, '2023-07-16 12:30:00')... 10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0  0.000 sec
16 01:22:03  INSERT INTO Login (User_id, Login_id, Password, Credit_card_no, Balance) VALUES  (1, 'john@example.co... 10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0    0.000 sec
17 01:22:03  SELECT * FROM User LIMIT 0, 1000                                                                      10 row(s) returned                                         0.015 sec / 0.000 sec
18 01:22:03  SELECT name, gender FROM Driver WHERE rating >= 3 LIMIT 0, 1000                                       10 row(s) returned                                         0.000 sec / 0.000 sec
19 01:22:03  SELECT Bill_no, User_id, Amount FROM Bill WHERE Amount > 50.00 LIMIT 0, 1000                          3 row(s) returned                                          0.000 sec / 0.000 sec
20 01:22:03  SELECT  c.Tsc_name AS Company_Name,  COUNT(t.Taxi_id) AS Total_Trips,  SUM(tr.Amount) AS Total_A...    5 row(s) returned                                          0.000 sec / 0.000 sec
21 01:22:03  SELECT  u.First_name || ' ' || u.Last_name AS User_name,  u.Contact_no AS Contact_number,  COUNT(t.Tri... 10 row(s) returned                                     0.000 sec / 0.000 sec
22 01:22:04  SELECT  t.Registration_no AS Taxi_registration,  t.Model AS Taxi_model,  COUNT(tr.Trip_id) AS Total_trips ... 10 row(s) returned                                 0.000 sec / 0.000 sec
23 01:22:04  INSERT INTO Bill (User_id, Driver_id, Amount, Date) SELECT  t.User_id,  t.Driver_id,  t.Amount * 0.8,  NO... 0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0   0.000 sec
24 01:22:04  SELECT * FROM Bill WHERE User_id = 5678 AND Driver_id = 9012 AND Date = (SELECT MAX(Date) FRO...          0 row(s) returned                                          0.000 sec / 0.000 sec
25 01:22:04  SELECT  U.User_id,  U.First_name,  U.Last_name,  SUM(T.Amount) AS Total_Amount_Spent FROM  U...          10 row(s) returned                                         0.000 sec / 0.000 sec
```



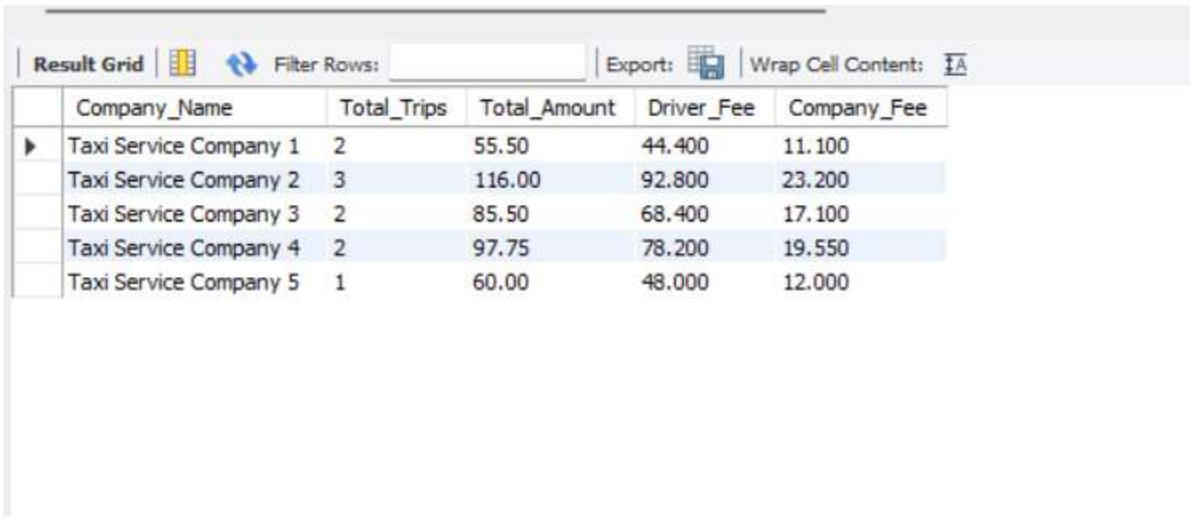
User_id	First_name	Last_name	Address	Age	Contact_no	Email
1	John	Doe	123 Main St	30	555-555-5555	johndoe@email.com
2	Jane	Smith	456 Elm St	25	555-555-1234	janesmith@email.com
3	Bob	Johnson	789 Oak St	40	555-555-6789	bobjohnson@email.com
4	Mary	Williams	321 Maple St	35	555-555-4321	marywilliams@email.com
5	David	Lee	654 Pine St	28	555-555-9876	davidlee@email.com
6	Karen	Taylor	987 Oak St	50	555-555-1111	karentaylor@email.com
7	Mike	Brown	456 Pine St	45	555-555-2222	mikebrown@email.com
8	Sarah	Davis	789 Maple St	30	555-555-3333	sarahdavis@email.com
9	Chris	Wilson	123 Elm St	35	555-555-4444	chriswilson@email.com
10	Jenny	Garcia	321 Oak St	25	555-555-5555	jennygarcia@email.com
NULL	NULL	NULL	NULL	NULL	NULL	NULL



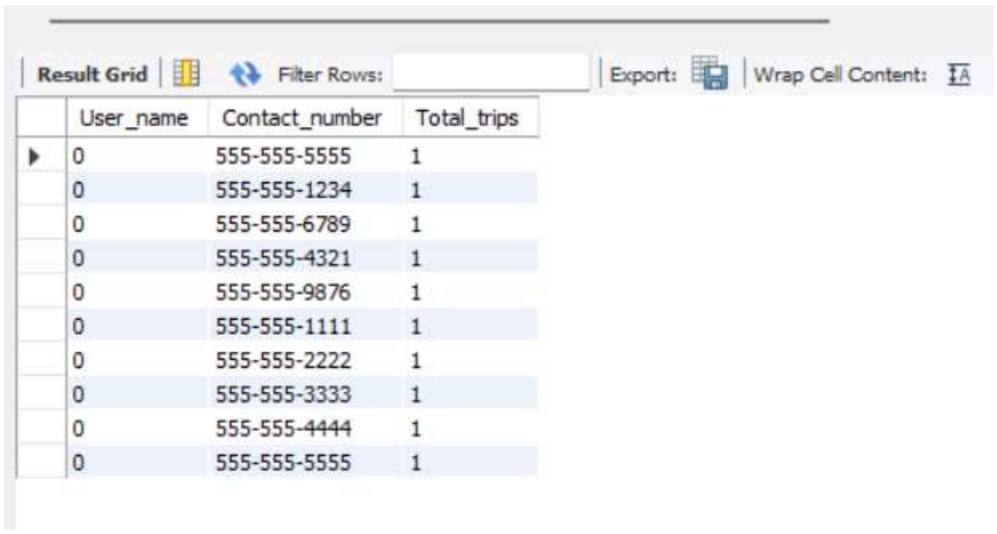
name	gender
John Doe	M
Jane Smith	F
Bob Johnson	M
Mary Williams	F
David Lee	M
Karen Taylor	F
Mike Brown	M
Sarah Davis	F
Chris Wilson	M
Jenny Garcia	F



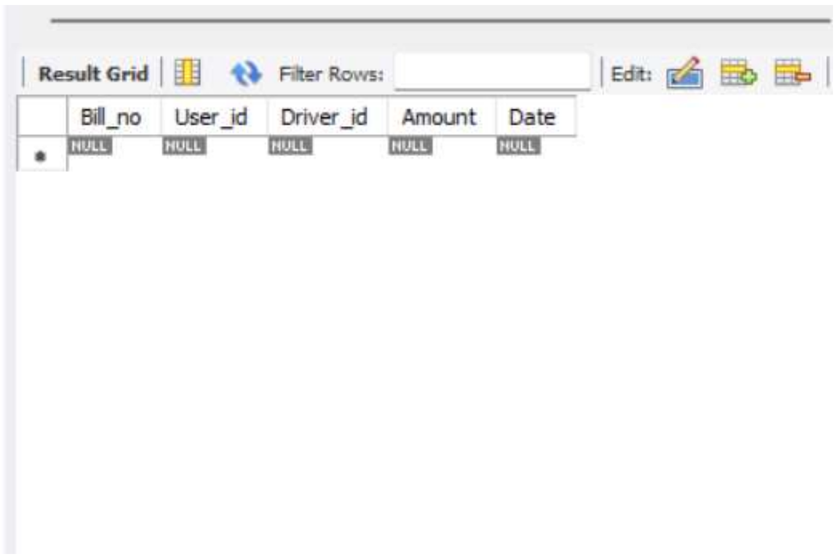
Bill_no	User_id	Amount
4	4	55.75
8	8	75.25
10	10	60.00
NULL	NULL	NULL



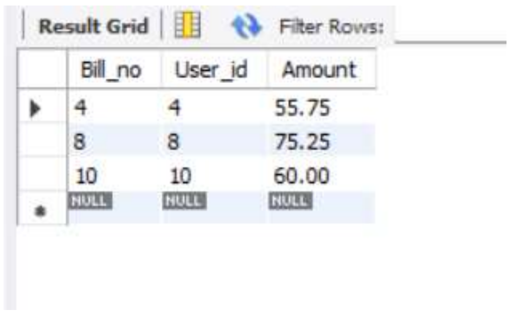
Company_Name	Total_Trips	Total_Amount	Driver_Fee	Company_Fee
Taxi Service Company 1	2	55.50	44.400	11.100
Taxi Service Company 2	3	116.00	92.800	23.200
Taxi Service Company 3	2	85.50	68.400	17.100
Taxi Service Company 4	2	97.75	78.200	19.550
Taxi Service Company 5	1	60.00	48.000	12.000



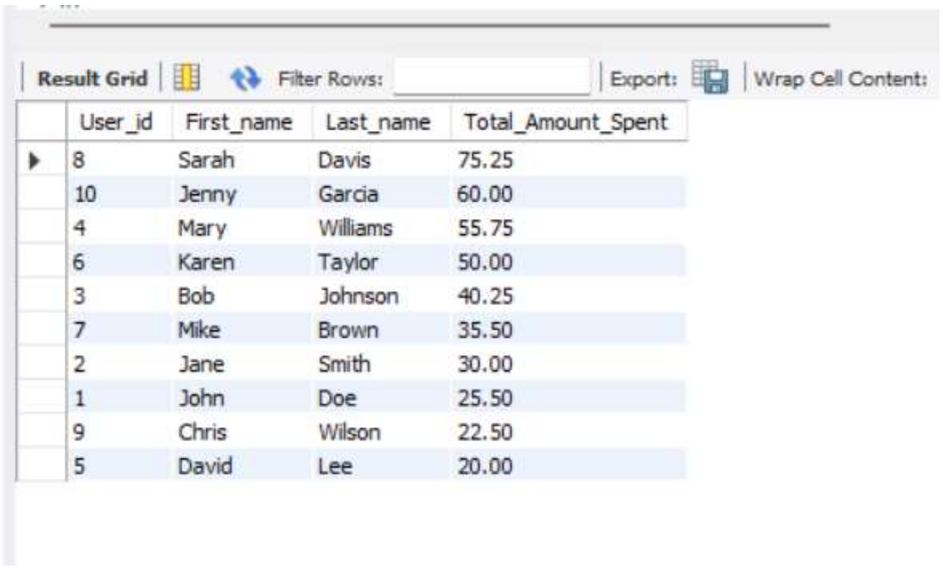
User_name	Contact_number	Total_trips
0	555-555-5555	1
0	555-555-1234	1
0	555-555-6789	1
0	555-555-4321	1
0	555-555-9876	1
0	555-555-1111	1
0	555-555-2222	1
0	555-555-3333	1
0	555-555-4444	1
0	555-555-5555	1



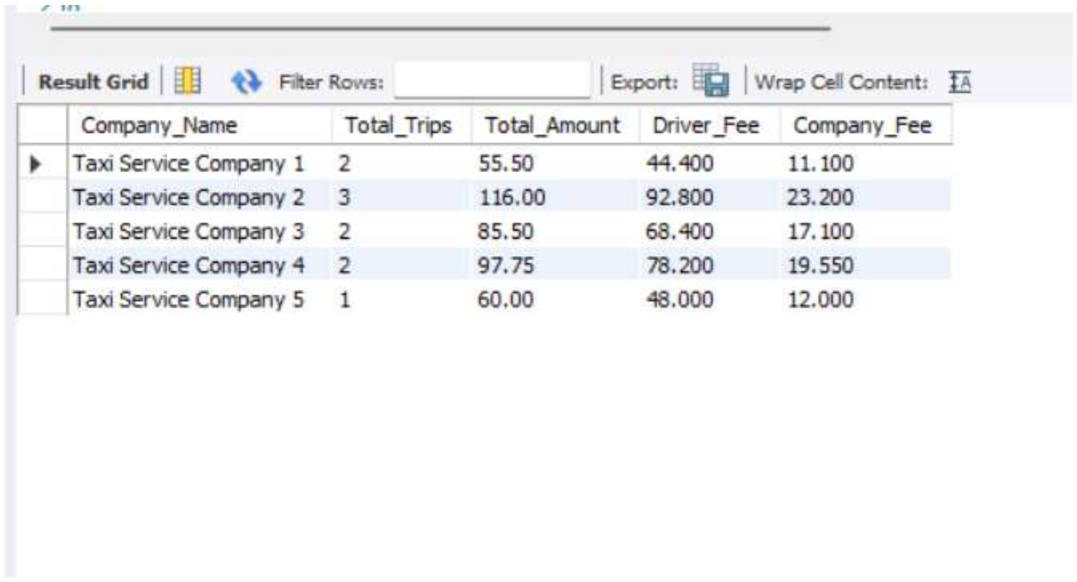
	Bill_no	User_id	Driver_id	Amount	Date
*	NULL	NULL	NULL	NULL	NULL



	Bill_no	User_id	Amount
▶	4	4	55.75
	8	8	75.25
	10	10	60.00
*	NULL	NULL	NULL

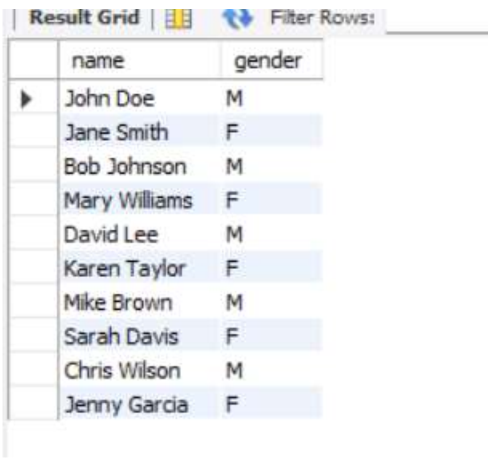


	User_id	First_name	Last_name	Total_Amount_Spent
▶	8	Sarah	Davis	75.25
	10	Jenny	Garcia	60.00
	4	Mary	Williams	55.75
	6	Karen	Taylor	50.00
	3	Bob	Johnson	40.25
	7	Mike	Brown	35.50
	2	Jane	Smith	30.00
	1	John	Doe	25.50
	9	Chris	Wilson	22.50
	5	David	Lee	20.00



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Company_Name	Total_Trips	Total_Amount	Driver_Fee	Company_Fee
Taxi Service Company 1	2	55.50	44.400	11.100
Taxi Service Company 2	3	116.00	92.800	23.200
Taxi Service Company 3	2	85.50	68.400	17.100
Taxi Service Company 4	2	97.75	78.200	19.550
Taxi Service Company 5	1	60.00	48.000	12.000



Result Grid | Filter Rows:

name	gender
John Doe	M
Jane Smith	F
Bob Johnson	M
Mary Williams	F
David Lee	M
Karen Taylor	F
Mike Brown	M
Sarah Davis	F
Chris Wilson	M
Jenny Garcia	F