DBMS ASSIGNMENT

Title: Online Vehicle Booking Management System

By

Team Members:

Devaram Sohan – 21BCE8402

Tharakanadh Sanjay – 21BCE8336

Abdul Aziz – 21BCE8621

Badrinath – 21BCE7620

Team Lead: Devaram Sohan

Slot Number: C1+TC1

Submission Date: 06-04-2023

Course Teacher: Dr.E. Anupriya

School of Computer Science and Engineering

ONLINE VEHICLE BOOKING MANAGEMENT SYSTEM

Introduction:

The Online Vehicle Booking Management System is designed to manage the process of purchasing and selling vehicles through a network of dealers and showrooms. This system provides a centralized platform for storing customer information, vehicle details, sales information, maintenance records, insurance policies, and resale details. The system is designed to handle day-to-day operations and ensure smooth management of the vehicle business.

Functionalities:

The system is capable of performing the following tasks:

- Storing customer information such as name, address, and mobile number.
- Assigning a unique customer ID to each customer.
- Allowing dealers to search for vehicles based on customer specifications such as price, color, and mileage.
- Displaying vehicle images, prices, discounts, and services available from various showrooms.
- Allowing dealers to finalize the deal with the showroom seller.
- Allowing customers to pay for the vehicle through the showroom and recording the purchase information in the sales section.
- Calculating taxes based on the price of the vehicle and storing the tax information in the tax ID section.
- Registering the purchased vehicle with the registration office and storing registration ID, vehicle number, driver's license, and

customer details.

- Handling vehicle maintenance records, including maintenance ID, date, mileage, type, and cost.
- Providing an associated insurance policy for every vehicle that includes insurance ID and coverage.
- Allowing customers to sell their old vehicles to a resale showroom and store information such as vehicle condition, model name, vehicle number, and customer details.

Database Design:

The database for the Online Vehicle Booking Management System is designed to store data in a structured and organized manner. The database consists of the following tables:

- Customers
- Vehicles
- Dealers
- Showrooms
- Sales
- Taxes
- Registrations
- Maintenance
- Insurance
- Resales

Normalization:

The tables in the database are normalized to reduce data redundancy and improve data integrity. The functional dependencies of each table are identified and normalized based on FD, Keys, and day-to-day operations.

Conclusion:

The Online Vehicle Booking Management System is designed to provide a comprehensive solution for managing the process of purchasing and selling vehicles. The system is capable of handling day-to-day operations and ensures smooth management of the vehicle business.

The database is designed to store data in a structured and organized manner, and normalization ensures data integrity and reduces redundancy. The system provides a centralized platform for storing customer information, vehicle details, sales information, maintenance records, insurance policies, and resale details.

Problem Statement:

The system required is for managing the process of purchasing and selling vehicles through a network of dealers and showrooms. The system should be capable of handling tasks such as storing customer information such as their name, address, and mobile number, and assigning a unique customer ID to each customer.

The system should also allow dealers to search for vehicles based on customer specifications such as price, color, and mileage. It should be able to display vehicle images, prices, discounts, and services available from various showrooms.

Once the customer and dealer agree on a vehicle, the system should allow the dealer to finalize the deal with the showroom seller. Customers should be able to pay for the vehicle through the showroom, and the system should record the purchase information in the sales section.

Additionally, the system should calculate taxes based on the price of the vehicle and store the tax information in the tax ID section. It should also register the purchased vehicle with the registration office and store registration ID, vehicle number, driver's license, and customer details.

The system should be capable of handling vehicle maintenance records, including maintenance ID, date, mileage, type, and cost. Every vehicle should have an associated insurance policy that includes insurance ID and coverage.

Furthermore, the system should allow customers to sell their old vehicles to a resale showroom and store information such as vehicle condition, model name, vehicle number, and customer details.

Problem Summary:

Based on the provided problem statement, it appears that a system is needed to manage the process of buying and selling vehicles through a network of dealers and showrooms. The system should be able to handle the following tasks:

Customer Management: The system should be able to store customer information such as name, address, and mobile number, and assign a unique customer ID to each customer.

Vehicle Search and Selection: The system should allow dealers to search for vehicles based on customer specifications such as price, color, and mileage. The system should also display vehicle images, prices, discounts, and services available from various showrooms.

Sales and Payment Processing: Once the customer and dealer agree on a vehicle, the system should allow the dealer to make the final deal with the showroom seller. The customer should be able to pay for the vehicle through the showroom, and the system should record the purchase information in the sales section.

Taxation: The system should calculate taxes based on the price of the vehicle and store the tax information in the tax ID section.

Vehicle Registration: The system should register the purchased vehicle with the registration office and store registration ID, vehicle number, driver's license, and customer details.

Maintenance and Insurance: The system should be able to handle vehicle maintenance records, including maintenance ID, date, mileage, type, and cost. Additionally, every vehicle should have an associated insurance policy that includes insurance ID and coverage.

Resale: The system should allow customers to sell their old vehicles to a resale showroom. The system should store information such as vehicle condition, model name, vehicle number, and customer details.

Overall, the system should be able to handle the buying and selling process of vehicles, including search and selection, payment processing, taxation, registration, maintenance, and resale.

Business Rules:

Based on the provided problem statement, the following business rules can be defined:

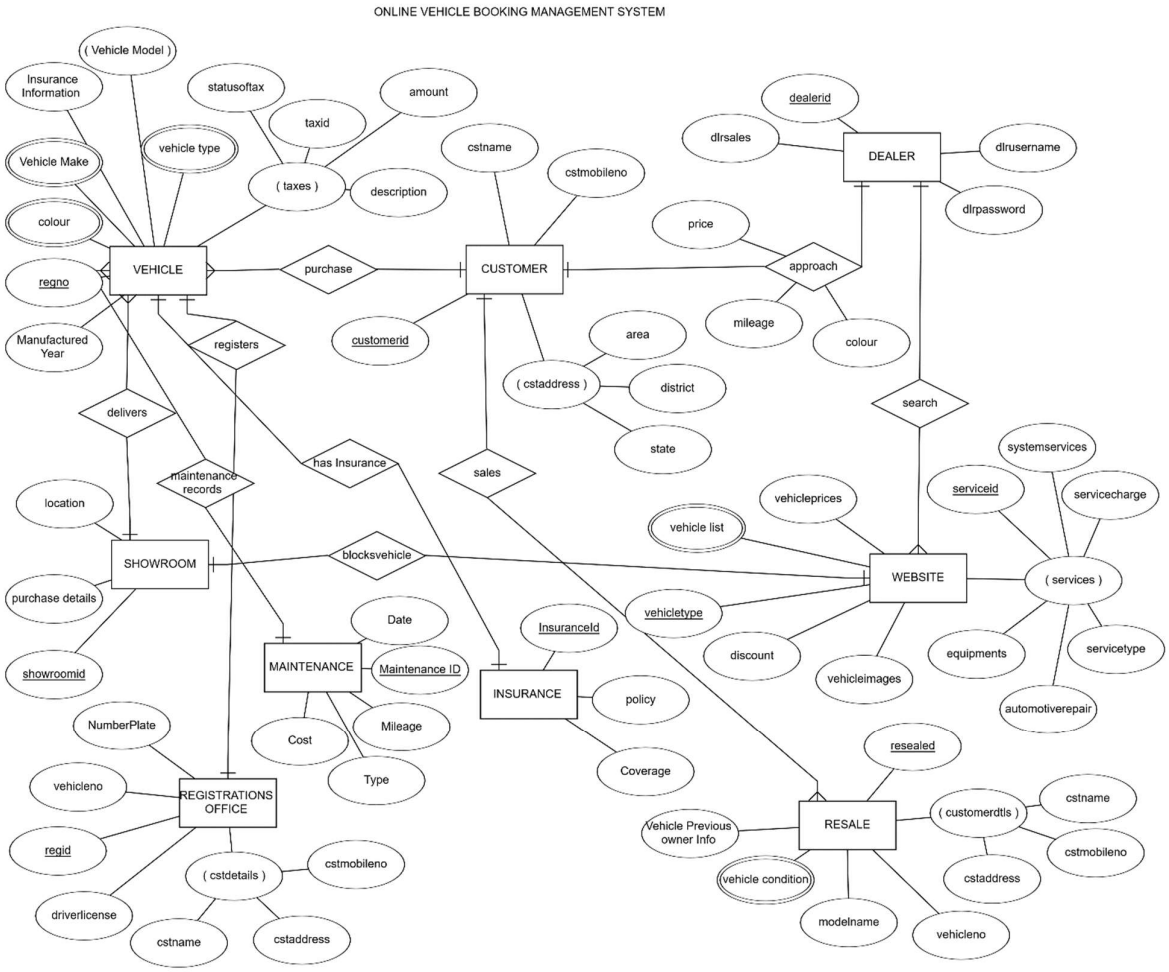
- Each customer can have one or more vehicles.
- Each dealer can have one or more customers.
- Each vehicle can be associated with one or more showrooms.
- Each showroom can have one or more vehicles.
- Each vehicle can have one or more services associated with it.
- Each service can be associated with one or more vehicles.
- Each vehicle can have one or more maintenance records associated with it.
- Each maintenance record is associated with only one vehicle.
- Each vehicle can have one or more insurance policies associated with it.
- Each insurance policy is associated with only one vehicle.

- Each vehicle can be associated with one or more sales transactions.
- Each sales transaction is associated with only one vehicle.
- Each tax ID can be associated with one or more sales transactions.
- Each sales transaction can be associated with only one tax ID.
- Each customer can sell one or more vehicles.
- Each resale showroom can purchase one or more vehicles from customers.
- Each purchased vehicle can be registered with only one registration ID.
- Each registration ID is associated with only one purchased vehicle.
- Each customer can have only one registration ID for each purchased vehicle.
- Each customer can have only one insurance policy for each purchased vehicle.
- Each vehicle can have only one resale record associated with it.
- Each resale record is associated with only one vehicle.

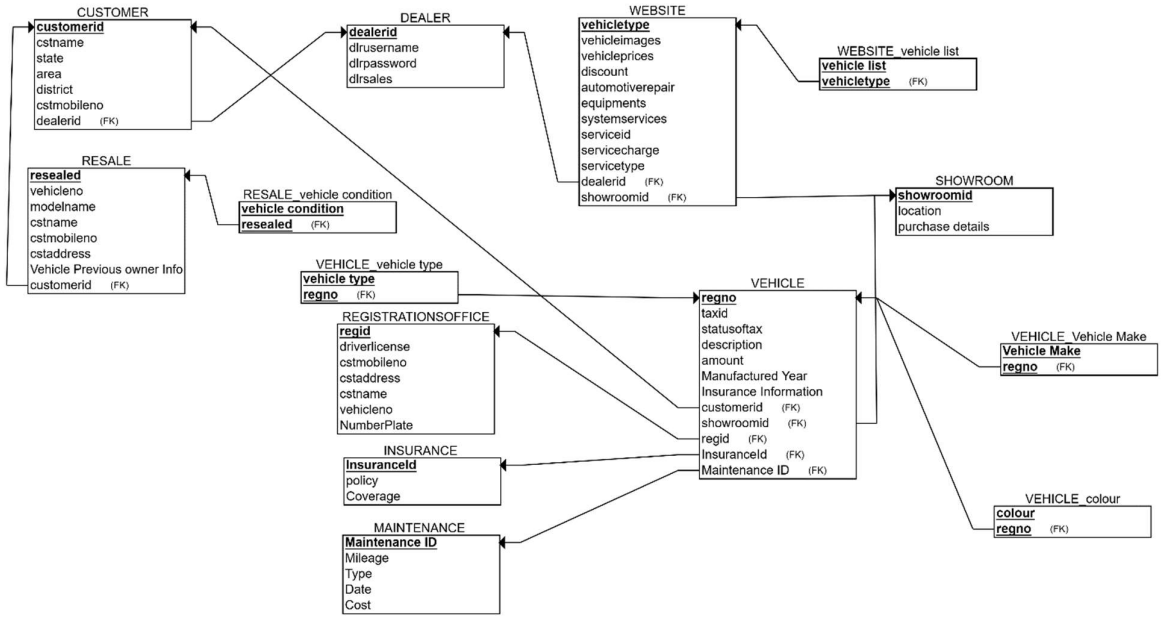
These business rules help to ensure data integrity and consistency in the online vehicle booking management system by defining relationships and constraints between various entities and their attributes. They also help to ensure that the system operates in a

way that is consistent with the business processes involved in buying, selling, registering, and maintaining vehicles.

Entity – Relationship (E-R) Diagram:



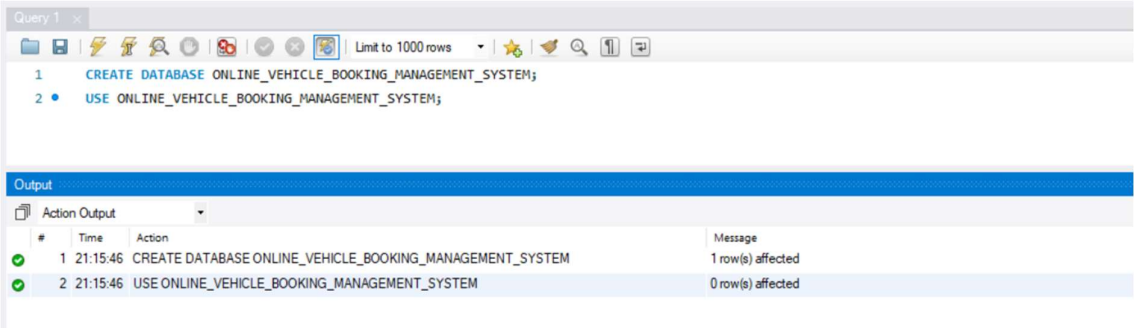
E-R To Relational Schema:



CREATION OF TABLES & CONSTRAINTS IN MYSQL:

CREATE DATABASE ONLINE_VEHICLE_BOOKING_MANAGEMENT_SYSTEM;

USE ONLINE_VEHICLE_BOOKING_MANAGEMENT_SYSTEM;




```
CREATE TABLE customers (
    customer_id INT PRIMARY KEY,
    name VARCHAR(50),
    address VARCHAR(100),
    mobile_number VARCHAR(20)
);
CREATE TABLE showrooms (
    showroom_id INT PRIMARY KEY,
    name VARCHAR(50),
    location VARCHAR(100),
    phone_number VARCHAR(20)
);
CREATE TABLE vehicles (
    vehicle_id INT PRIMARY KEY,
    make VARCHAR(50),
    model VARCHAR(50),
    color VARCHAR(20),
    mileage INT,
    price DECIMAL(10,2),
```

```
    discount DECIMAL(10,2),
    showroom_id INT,
    FOREIGN KEY (showroom_id) REFERENCES showrooms(showroom_id)
);
```

SQL File 5* | online_vehicle_booking_manag_

Limit to 1000 rows

```sql
CREATE TABLE customers (
    customer_id INT PRIMARY KEY,
    name VARCHAR(50),
    address VARCHAR(100),
    mobile_number VARCHAR(20)
);

CREATE TABLE showrooms (
    showroom_id INT PRIMARY KEY,
    name VARCHAR(50),
    location VARCHAR(100),
    phone_number VARCHAR(20)
);

CREATE TABLE vehicles (
    vehicle_id INT PRIMARY KEY,
    make VARCHAR(50),
    model VARCHAR(50),
    color VARCHAR(20),
    mileage INT,
    price DECIMAL(10,2),
    discount DECIMAL(10,2),
    showroom_id INT,
    FOREIGN KEY (showroom_id) REFERENCES showrooms(showroom_id)
);
```

Output

Action Output

#	Time	Action	Message
1	14:45:33	CREATE TABLE customers (customer_id INT PRIMARY KEY, name VARCHAR(50), address VARCHAR(...	0 row(s) affected
2	14:45:33	CREATE TABLE showrooms (showroom_id INT PRIMARY KEY, name VARCHAR(50), location VARCHA...	0 row(s) affected
3	14:45:33	CREATE TABLE vehicles (vehicle_id INT PRIMARY KEY, make VARCHAR(50), model VARCHAR(50), ...	0 row(s) affected

```
CREATE TABLE registrations (
    registration_id INT PRIMARY KEY,
    vehicle_number VARCHAR(20),
    driver_license_number VARCHAR(20),
    customer_id INT,
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);
CREATE TABLE deals (
    deal_id INT PRIMARY KEY,
    customer_id INT,
    vehicle_id INT,
    purchase_date DATE,
    total_cost DECIMAL(10,2),
    tax DECIMAL(10,2),
    registration_id INT,
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id),
    FOREIGN KEY (vehicle_id) REFERENCES vehicles(vehicle_id),
    FOREIGN KEY (registration_id) REFERENCES registrations(registration_id)
);
CREATE TABLE maintenance (
    maintenance_id INT PRIMARY KEY,
    vehicle_id INT,
    maintenance_date DATE,
    mileage INT,
    maintenance_type VARCHAR(50),
    cost DECIMAL(10,2),
    FOREIGN KEY (vehicle_id) REFERENCES vehicles(vehicle_id)
);
```

SQL File 5* online_vehicle_booking_manag...

Limit to 1000 rows

```sql
CREATE TABLE registrations (
    registration_id INT PRIMARY KEY,
    vehicle_number VARCHAR(20),
    driver_license_number VARCHAR(20),
    customer_id INT,
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);

CREATE TABLE deals (
    deal_id INT PRIMARY KEY,
    customer_id INT,
    vehicle_id INT,
    purchase_date DATE,
    total_cost DECIMAL(10,2),
    tax DECIMAL(10,2),
    registration_id INT,
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id),
    FOREIGN KEY (vehicle_id) REFERENCES vehicles(vehicle_id),
    FOREIGN KEY (registration_id) REFERENCES registrations(registration_id)
);

CREATE TABLE maintenance (
    maintenance_id INT PRIMARY KEY,
    vehicle_id INT,
    maintenance_date DATE,
    mileage INT,
    maintenance_type VARCHAR(50),
    cost DECIMAL(10,2),
    FOREIGN KEY (vehicle_id) REFERENCES vehicles(vehicle_id)
);
```

Output

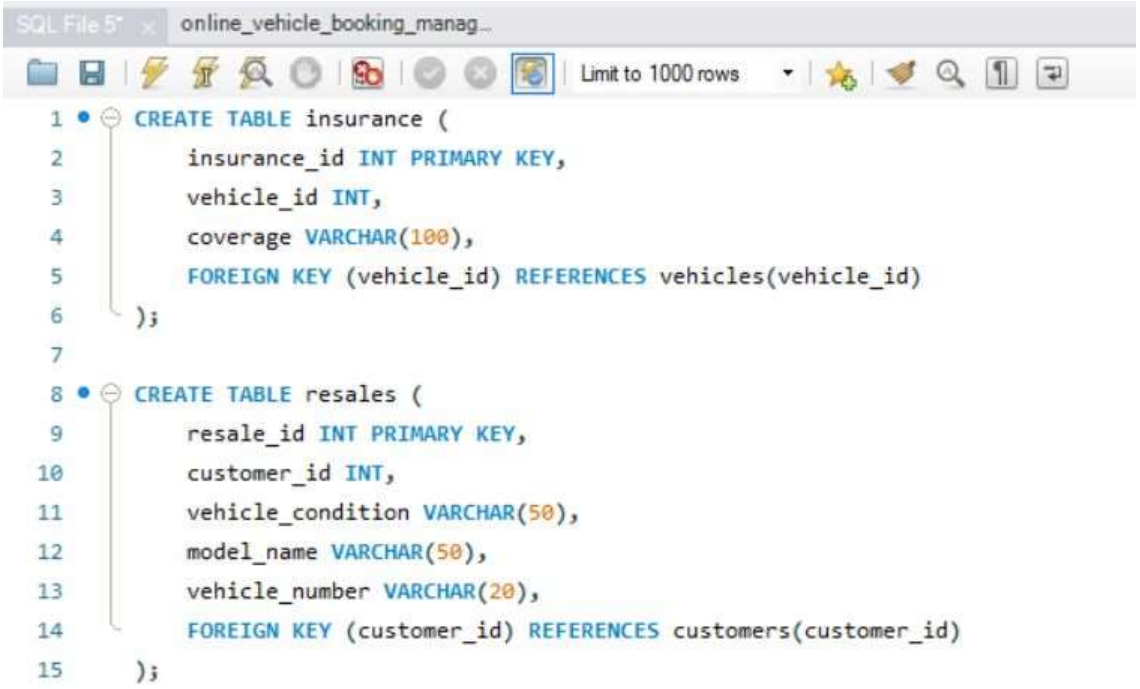
Action Output

#	Time	Action	Message
1	14:50:27	CREATE TABLE registrations (registration_id INT PRIMARY KEY, vehicle_number VARCHAR(20), driver...	0 row(s) affected
2	14:50:28	CREATE TABLE deals (deal_id INT PRIMARY KEY, customer_id INT, vehicle_id INT, purchase_date ...	0 row(s) affected
3	14:50:28	CREATE TABLE maintenance (maintenance_id INT PRIMARY KEY, vehicle_id INT, maintenance_date ...	0 row(s) affected

```sql
CREATE TABLE insurance (
    insurance_id INT PRIMARY KEY,
    vehicle_id INT,
```

```
    coverage VARCHAR(100),
    FOREIGN KEY (vehicle_id) REFERENCES vehicles(vehicle_id)
);

CREATE TABLE resales (
    resale_id INT PRIMARY KEY,
    customer_id INT,
    vehicle_condition VARCHAR(50),
    model_name VARCHAR(50),
    vehicle_number VARCHAR(20),
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);
```



INSERTION OF DATA INTO TABLES:

SQL File 5* online_vehicle_booking_manag...

Limit to 1000 rows

```sql
INSERT INTO Customers VALUES (1, 'Abdul Aziz', 'Old City', '1234567890');
INSERT INTO Customers VALUES (2, 'Sanjay Das', 'Hi-Tech City', '9876409212');
INSERT INTO Customers VALUES (3, 'Badrinath', 'Gachibowli', '7821204908');
INSERT INTO Customers VALUES (4, 'Sohan Reddy', 'Old Guntur', '855567534');
INSERT INTO Customers VALUES (5, 'Hemanth Reddy', 'Vijayawada', '876310934');
```

Output

Action Output

#	Time	Action	Message
1	16:50:35	INSERT INTO Customers VALUES (1, 'Abdul Aziz', 'Old City', '1234567890')	1 row(s) affected
2	16:50:35	INSERT INTO Customers VALUES (2, 'Sanjay Das', 'Hi-Tech City', '9876409212')	1 row(s) affected
3	16:50:35	INSERT INTO Customers VALUES (3, 'Badrinath', 'Gachibowli', '7821204908')	1 row(s) affected
4	16:50:35	INSERT INTO Customers VALUES (4, 'Sohan Reddy', 'Old Guntur', '855567534')	1 row(s) affected
5	16:50:35	INSERT INTO Customers VALUES (5, 'Hemanth Reddy', 'Vijayawada', '876310934')	1 row(s) affected

SQL File 5* online_vehicle_booking_manag...

Limit to 1000 rows

```sql
INSERT INTO Showrooms (showroom_id, name, location, phone_number)
VALUES
    (101, 'Silver Motors', 'Film Nagar', '8888444422'),
    (102, 'Apachi Autos', 'Jubliee Hills', '9999888833'),
    (103, 'Big Boy Toys', 'Gachibowli', '7777666655'),
    (104, 'Kun Exclusive', 'Banjara Hills', '5555999921'),
    (105, 'Sky Motors', 'Kukatpally', '4444222211');
```

Output

Action Output

#	Time	Action	Message
1	17:21:26	INSERT INTO Showrooms (showroom_id, name, location, phone_number) VALUES (101, 'Silver Motors',...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0

SQL File 5* online_vehicle_booking_manag...

Limit to 1000 rows

```sql
INSERT INTO Vehicles (vehicle_id, make, model, color, mileage, price, discount, showroom_id)
VALUES  (1001, 'Toyota', 'Innova', 'Red', 20, 3000000.00, 20000, 101),
        (1002, 'Honda', 'City', 'White', 15, 2500000.00, 50000, 102),
        (1003, 'Ford', 'Mustang','Black', 3, 7000000.00, 100000, 103),
        (1004, 'Nissan', 'Sunny', 'Blue', 16, 3000000.00, 40000, 104),
        (1005, 'Mercedes', 'Benz-C-Class','Grey', 7, 5000000.00, 70000, 105);
```

Output

Action Output

#	Time	Action	Message
1	17:28:48	INSERT INTO Vehicles (vehicle_id, make, model, color, mileage, price, discount, showroom_id) VALUES (1...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```sql
INSERT INTO Resales(resale_id,customer_id,vehicle_condition,model_name,vehicle_number)
    VALUES  (10001, 3,'Good', 'C200 Benz', 'TS D 4567'),
            (10002, 4,'Brand New', 'Audi R8', 'TS Y 3456'),
            (10003, 5,'Well Maintained', 'Fortuner', 'TS Z 2345'),
            (10004, 1,'Average', 'Corolla', 'AP W 4567'),
            (10005, 2,'Minor Defects', 'Median', 'AP F 1923');
```

Output

Action Output

#	Time	Action	Message
1	17:50:13	INSERT INTO Resales(resale_id,customer_id,vehicle_condition,model_name,vehicle_number) VALUES (10...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```sql
INSERT INTO maintenance(maintenance_id,vehicle_id,maintenance_date,mileage,maintenance_type,cost)
    VALUES  (2001, 1005, '2022-08-01', 7,'Oil change', 100000.00),
            (2002, 1004, '2022-03-01', 16, 'Tire rotation', 40000.00),
            (2003, 1003, '2022-06-05', 3, 'Brake pads replacement', 200000.00),
            (2004, 1002, '2022-04-01', 15, 'Air filter replacement', 25000.00),
            (2005, 1001, '2022-09-01', 20, 'Air filter replacement', 40000.00);
```

Output

Action Output

#	Time	Action	Message
1	18:03:55	INSERT INTO maintenance(maintenance_id,vehicle_id,maintenance_date,mileage,maintenance_type,cost) ...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```sql
INSERT INTO insurance(insurance_id,vehicle_id,coverage)
    VALUES  (3001, 1005, 'Third-Party-Liability Only Cover'),
            (3002, 1004, 'Comprehensive Car Insurance'),
            (3003, 1003, 'Collison Damage'),
            (3004, 1002, 'Personal Accident Cover'),
            (3005, 1001, 'Zero Depreciation Insurance');
```

Output

Action Output

#	Time	Action	Message
1	18:10:40	INSERT INTO insurance(insurance_id,vehicle_id,coverage) VALUES (3001, 1005, 'Third-Party-Liability Only Co...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```sql
INSERT INTO Registrations(registration_id,vehicle_number,driver_license_number,customer_id)
    VALUES  (5001, 'AP M 0001', '260953', 5),
            (5002, 'AP N 1101', '123453', 4),
            (5003, 'AP X 9999', '567834', 3),
            (5004, 'AP Y 8888', '467893', 2),
            (5005, 'AP Z 7766', '329872', 1);
```

Output

Action Output

#	Time	Action	Message
1	18:26:16	INSERT INTO Registrations(registration_id,vehicle_number,driver_license_number,customer_id) VALUES (5...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0

```sql
INSERT INTO Deals (deal_id, customer_id, vehicle_id, purchase_date, total_cost, tax, registration_id)
VALUES
    (4001, 1, 1003, '2022-01-10',3000000.00,400000,5001),
    (4002, 3, 1002, '2022-01-20',2500000.00,300000,5002),
    (4003, 4, 1004, '2022-02-05',7000000.00,1000000,5003),
    (4004, 5, 1005, '2022-02-28',3000000.00,300000,5004),
    (4005, 2, 1001, '2022-02-28',5000000.00,500000,5005);
```

Output

Action Output

#	Time	Action	Message
1	18:27:56	INSERT INTO Deals (deal_id, customer_id, vehicle_id, purchase_date, total_cost, tax, registration_id) VAL...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0

DISPLAYING DATA IN TABLES:

SELECT*FROM CUSTOMERS;

SELECT*FROM SHOWROOMS;

SELECT*FROM VEHICLES;

SELECT*FROM REGISTRATIONS;

SELECT*FROM DEALS;

SELECT*FROM MAINTENANCE;

SELECT*FROM INSURANCE;

SELECT*FROM RESALES;

```sql
select*from customers;
```

Result Grid

customer_id	name	address	mobile_number
1	Abdul Aziz	Old City	1234567890
2	Sanjay Das	Hi-Tech City	9876409212
3	Badrinath	Gachibowli	7821204908
4	Sohan Reddy	Old Guntur	855567534
5	Hemanth Reddy	Vijayawada	876310934
NULL	NULL	NULL	NULL

customers 4

Output

Action Output

#	Time	Action	Message
1	21:30:31	select*from customers LIMIT 0, 1000	5 row(s) returned

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
1 • select*from showrooms;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

showroom_id	name	location	phone_number
101	Silver Motors	Film Nagar	8888444422
102	Apachi Autos	Jubliee Hills	9999888833
103	Big Boy Toys	Gachibowli	7777666655
104	Kun Exclusive	Banjara Hills	5555999921
105	Sky Motors	Kukatpally	4444222211
NULL	NULL	NULL	NULL

showrooms 10

Output

Action Output

#	Time	Action	Message
1	18:31:35	select*from showrooms LIMIT 0, 1000	5 row(s) returned

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
1 • select*from vehicles;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

vehicle_id	make	model	color	mileage	price	discount	showroom_id
1001	Toyota	Innova	Red	20	3000000.00	20000.00	101
1002	Honda	City	White	15	2500000.00	50000.00	102
1003	Ford	Mustang	Black	3	7000000.00	100000.00	103
1004	Nissan	Sunny	Blue	16	3000000.00	40000.00	104
1005	Mercedes	Benz-C-Class	Grey	7	5000000.00	70000.00	105
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

vehicles 11

Output

Action Output

#	Time	Action	Message
1	18:32:01	select*from vehicles LIMIT 0, 1000	5 row(s) returned

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
select*from registrations;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

registration_id	vehicle_number	driver_license_number	customer_id
5001	AP M 0001	260953	5
5002	AP N 1101	123453	4
5003	AP X 9999	567834	3
5004	AP Y 8888	467893	2
5005	AP Z 7766	329872	1
NULL	NULL	NULL	NULL

registrations 8

Output

Action Output

#	Time	Action	Message
1	18:30:33	select*from registrations LIMIT 0, 1000	5 row(s) returned

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
select*from deals;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

deal_id	customer_id	vehicle_id	purchase_date	total_cost	tax	registration_id
4001	1	1003	2022-01-10	3000000.00	400000.00	5001
4002	3	1002	2022-01-20	2500000.00	300000.00	5002
4003	4	1004	2022-02-05	7000000.00	1000000.00	5003
4004	5	1005	2022-02-28	3000000.00	300000.00	5004
4005	2	1001	2022-02-28	5000000.00	500000.00	5005
NULL	NULL	NULL	NULL	NULL	NULL	NULL

deals 5

Output

Action Output

#	Time	Action	Message
1	18:29:01	select*from deals LIMIT 0, 1000	5 row(s) returned

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
1 • select*from deals;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

deal_id	customer_id	vehicle_id	purchase_date	total_cost	tax	registration_id
4001	1	1003	2022-01-10	3000000.00	400000.00	5001
4002	3	1002	2022-01-20	2500000.00	300000.00	5002
4003	4	1004	2022-02-05	7000000.00	1000000.00	5003
4004	5	1005	2022-02-28	3000000.00	300000.00	5004
4005	2	1001	2022-02-28	5000000.00	500000.00	5005
NULL	NULL	NULL	NULL	NULL	NULL	NULL

deals 5

Output

Action Output

#	Time	Action	Message
1	18:29:01	select*from deals LIMIT 0, 1000	5 row(s) returned

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
1 • select*from maintenance;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

maintenance_id	vehicle_id	maintenance_date	mileage	maintenance_type	cost
2001	1005	2022-08-01	7	Oil change	100000.00
2002	1004	2022-03-01	16	Tire rotation	40000.00
2003	1003	2022-06-05	3	Brake pads replacement	200000.00
2004	1002	2022-04-01	15	Air filter replacement	25000.00
2005	1001	2022-09-01	20	Air filter replacement	40000.00
NULL	NULL	NULL	NULL	NULL	NULL

maintenance 7

Output

Action Output

#	Time	Action	Message
1	18:30:04	select*from maintenance LIMIT 0, 1000	5 row(s) returned

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
select*from insurance;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

insurance_id	vehicle_id	coverage
3001	1005	Third-Party-Liability Only Cover
3002	1004	Comprehensive Car Insurance
3003	1003	Collison Damage
3004	1002	Personal Accident Cover
3005	1001	Zero Depreciation Insurance
NULL	NULL	NULL

insurance 6

Output

Action Output

#	Time	Action	Message
1	18:29:36	select*from insurance LIMIT 0, 1000	5 row(s) returned

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
select*from resales;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

resale_id	customer_id	vehicle_condition	model_name	vehicle_number
10001	3	Good	C200 Benz	TS D 4567
10002	4	Brand New	Audi R8	TS Y 3456
10003	5	Well Maintained	Fortuner	TS Z 2345
10004	1	Average	Corolla	AP W 4567
10005	2	Minor Defects	Median	AP F 1923
NULL	NULL	NULL	NULL	NULL

resales 9

Output

Action Output

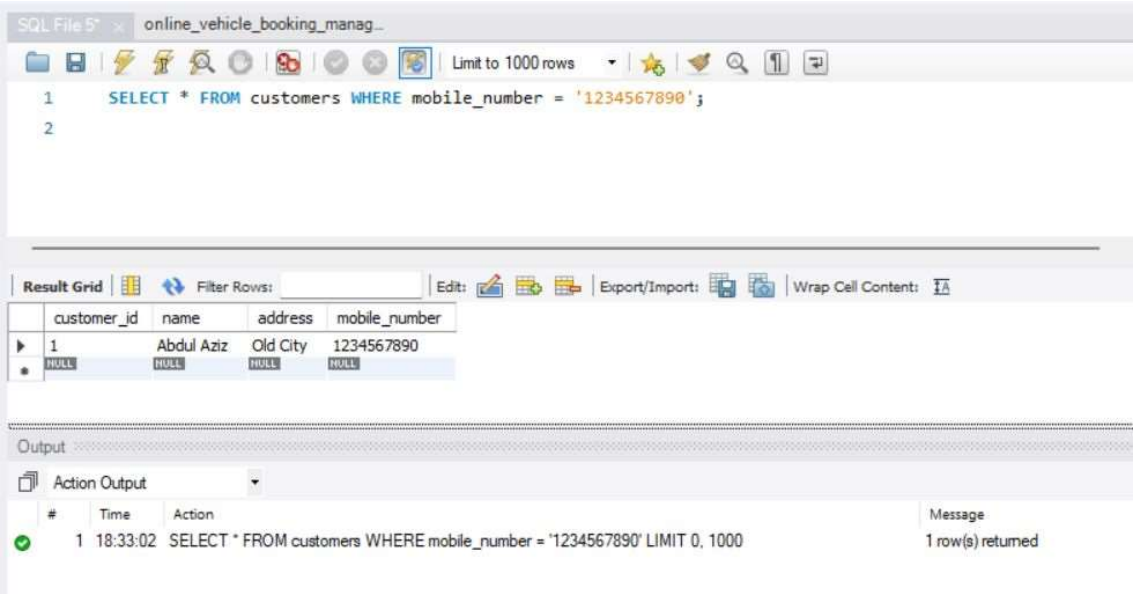
#	Time	Action	Message
1	18:31:07	select*from resales LIMIT 0, 1000	5 row(s) returned

SQL QUERIES & RELATIONAL ALGEBRA QUERIES:

- Simple queries with condition:

1. Find all customers who have the mobile number '1234567890':

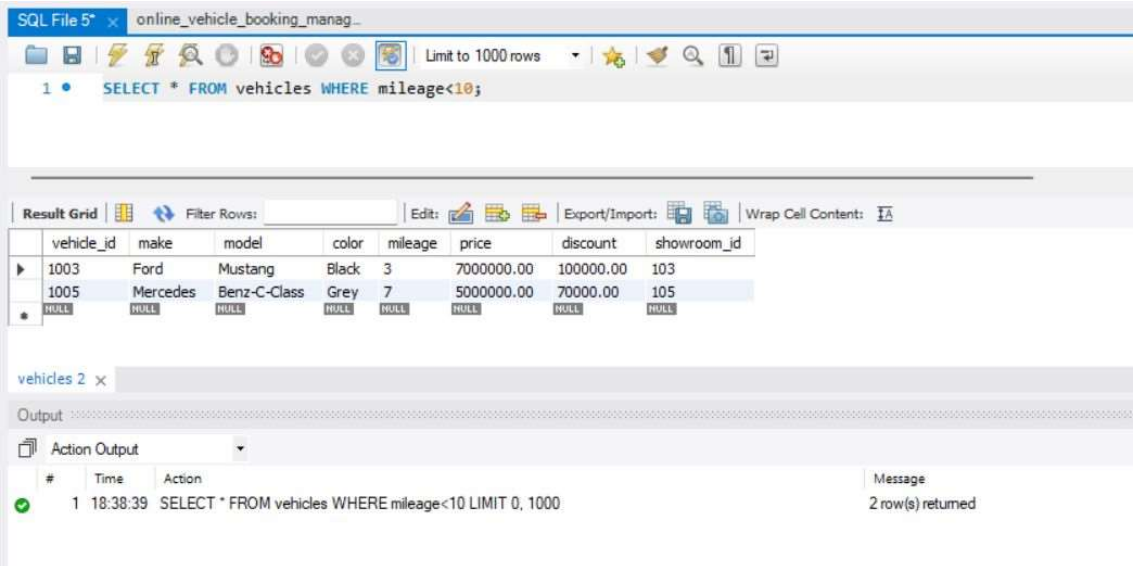
➢ SELECT * FROM customers WHERE mobile_number = '1234567890';



➢ RA QUERY: σ mobile_number='1234567890'(customers)

2. Find all vehicles with mileage less than 10:

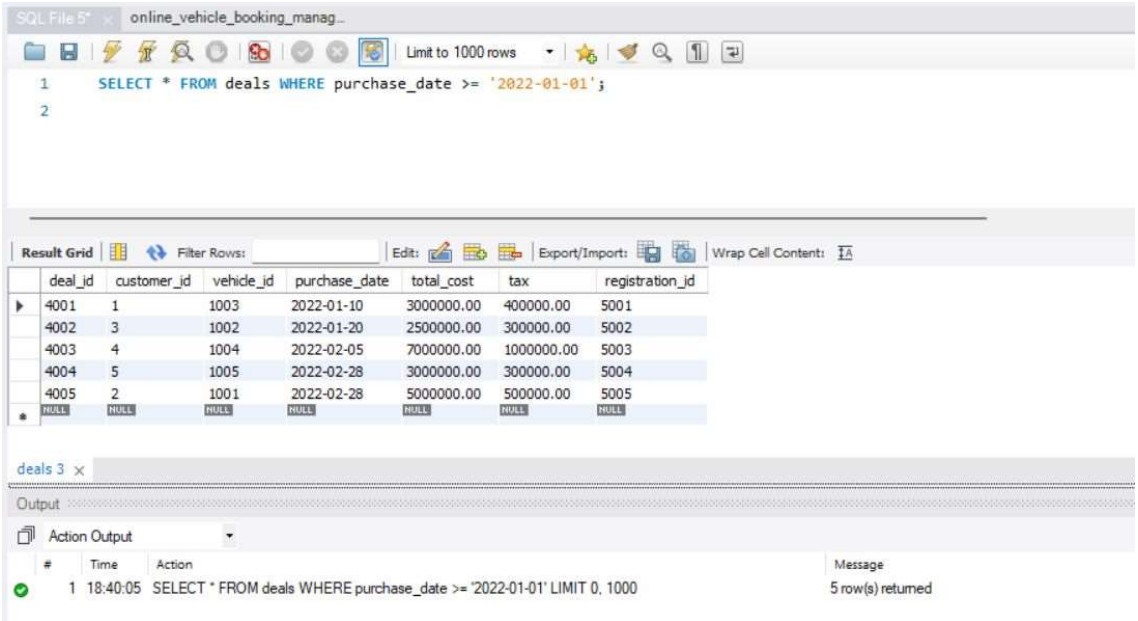
➢ SELECT * FROM vehicles WHERE mileage <10;



➢ RA QUERY: σ mileage<10(vehicles)

3. Find all deals made on or after January 1, 2022:

- SELECT * FROM deals WHERE purchase_date >= '2022-01-01';

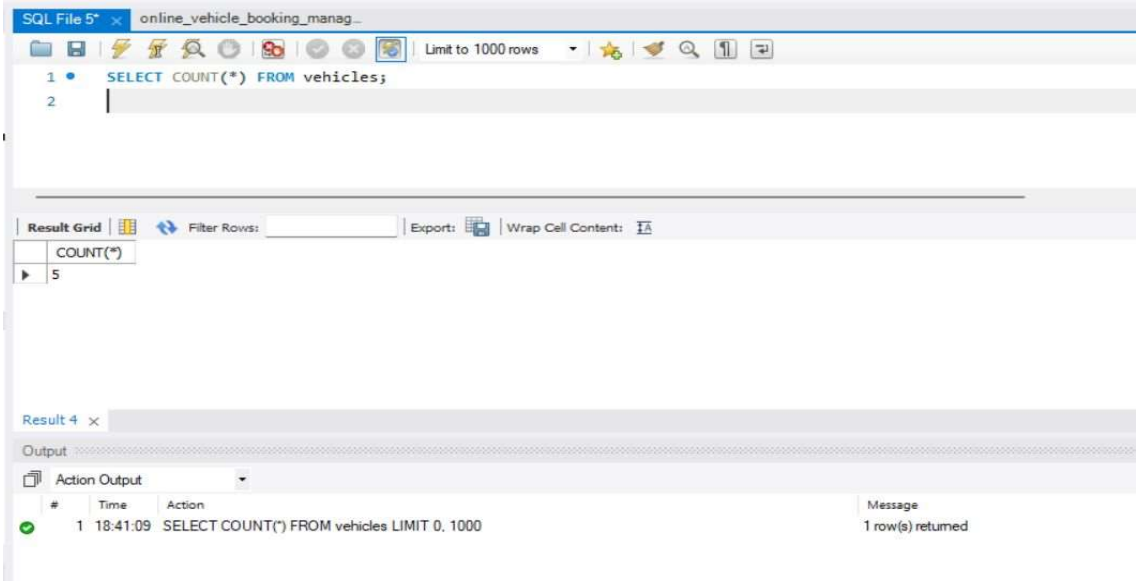


- RA QUERY: σ purchase_date>='2022-01-01'(deals)

- <u>Aggregate Queries:</u>

4. Find the total number of vehicles in the database:

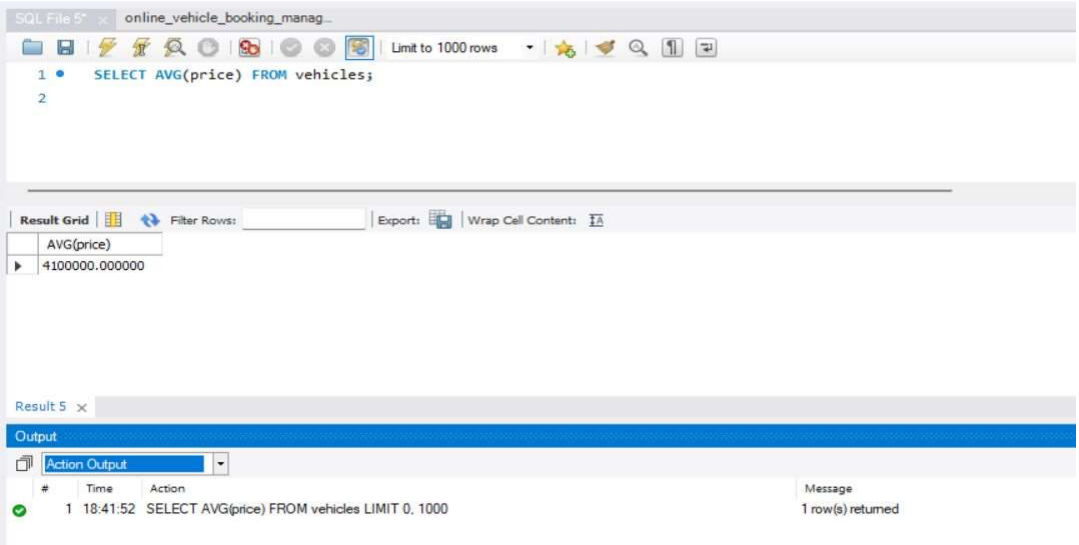
- SELECT COUNT(*) FROM vehicles;



- RA QUERY: πcount(vehicles)

5. Find the average price of vehicles in the database:

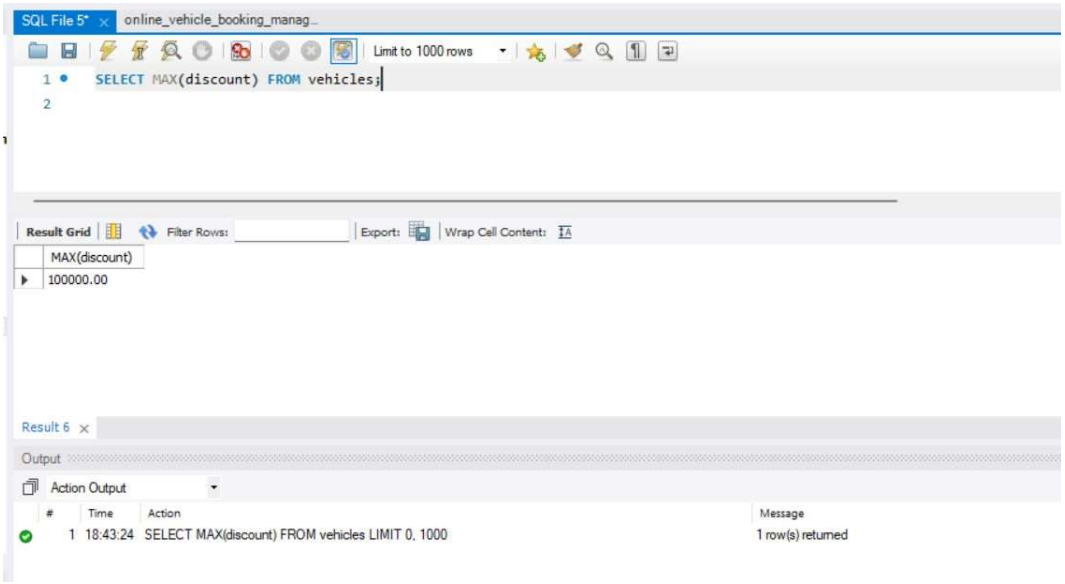
- SELECT AVG(price) FROM vehicles;



- RA QUERY: πavg(price)(vehicles)

6. Find the maximum discount offered on any vehicle:

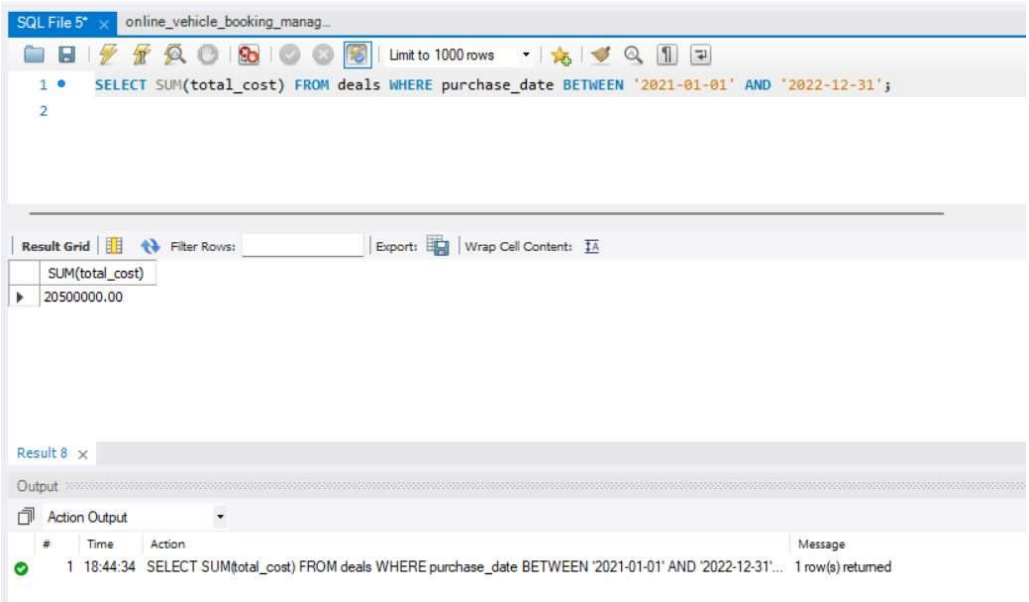
- SELECT MAX(discount) FROM vehicles;



- RA QUERY: πmax(discount)(vehicles)

7. Find the total cost of all deals made in 2021:

- SELECT SUM(total_cost) FROM deals WHERE purchase_date BETWEEN '2021-01-01' AND '2021-12-31';

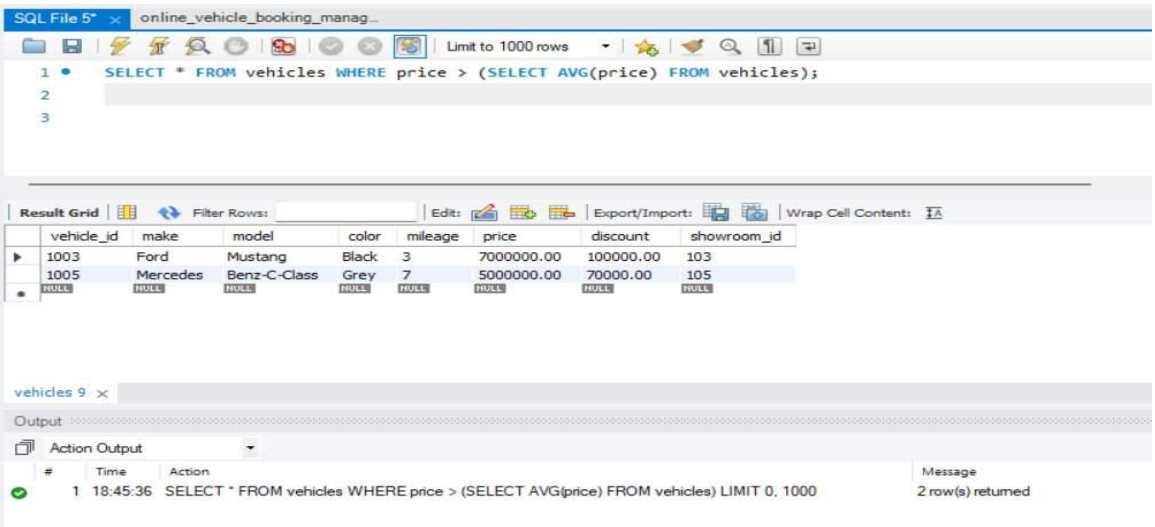


- RA QUERY: σpurchase_date BETWEEN ‘2021-01-01’ AND ‘2021-12-31’(deals) ⋈πsum(total_cost)(deals)

- <u>Sub queries or complex queries:</u>

8. Find all vehicles with a price greater than the average price of all vehicles in the database:

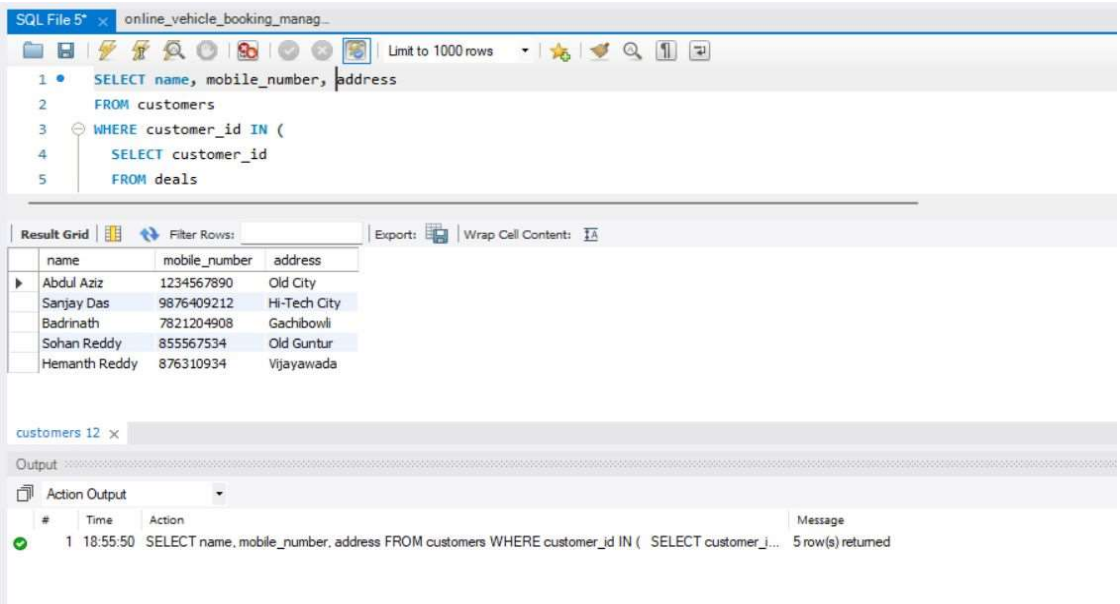
- SELECT * FROM vehicles WHERE price > (SELECT AVG(price) FROM vehicles);



- RA QUERY: ρv1(πprice(vehicles)) ⋈ρv2(πavg(price)(vehicles)) (σprice>avg(price)(v1))

9. Retrieve the names, phone numbers, and addresses of all customers who have made a purchase with a total cost greater than 1,000,000.

- SELECT customer_name, customer_phone, customer_address
  FROM customers
  WHERE customer_id IN (
  SELECT customer_id
  FROM deals
  WHERE total_cost > 1000000);



- RA QUERY: πcustomer_name, customer_phone, customer_address(σcustomer_id ∈ (σtotal_cost > 1000000(deals)) (customers))

10. Find all vehicles with the highest mileage for each make:

- SELECT * FROM vehicles v1 WHERE mileage = (SELECT MAX(mileage) FROM vehicles v2 WHERE v1.make = v2.make);

- RA QUERY: ρv1(vehicles) ⋈ρv2(πmake,max(mileage)(vehicles)) (σmileage=max(mileage)(v2)∧v1.make=v2.make(v1))

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
SELECT * FROM vehicles v1 WHERE mileage =
    (SELECT MAX(mileage) FROM vehicles v2 WHERE v1.make = v2.make);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

vehicle_id	make	model	color	mileage	price	discount	showroom_id
1001	Toyota	Innova	Red	20	3000000.00	20000.00	101
1002	Honda	City	White	15	2500000.00	50000.00	102
1003	Ford	Mustang	Black	3	7000000.00	100000.00	103
1004	Nissan	Sunny	Blue	16	3000000.00	40000.00	104
1005	Mercedes	Benz-C-Class	Grey	7	5000000.00	70000.00	105
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

vehicles 11

Output

Action Output

#	Time	Action	Message
1	18:50:24	SELECT * FROM vehicles v1 WHERE mileage = (SELECT MAX(mileage) FROM vehicles v2 WHERE v...	5 row(s) returned

- <u>Join queries:</u>

11. Find all customers who have made a deal, along with the details of the vehicle they purchased:

➢ SELECT customers.*, vehicles.* FROM customers
JOIN deals ON customers.customer_id = deals.customer_id
JOIN vehicles ON deals.vehicle_id = vehicles.vehicle_id;

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
SELECT customers.*, vehicles.* FROM customers
    JOIN deals ON customers.customer_id = deals.customer_id
    JOIN vehicles ON deals.vehicle_id = vehicles.vehicle_id;
```

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

customer_id	name	address	mobile_number	vehicle_id	make	model	color	mileage	price	discount	showroom_id
1	Abdul Aziz	Old City	1234567890	1003	Ford	Mustang	Black	3	7000000.00	100000.00	103
2	Sanjay Das	Hi-Tech City	9876409212	1001	Toyota	Innova	Red	20	3000000.00	20000.00	101
3	Badrinath	Gachibowli	7821204908	1002	Honda	City	White	15	2500000.00	50000.00	102
4	Sohan Reddy	Old Guntur	855567534	1004	Nissan	Sunny	Blue	16	3000000.00	40000.00	104
5	Hemanth Reddy	Vijayawada	876310934	1005	Mercedes	Benz-C-Class	Grey	7	5000000.00	70000.00	105

Result 13

Output

Action Output

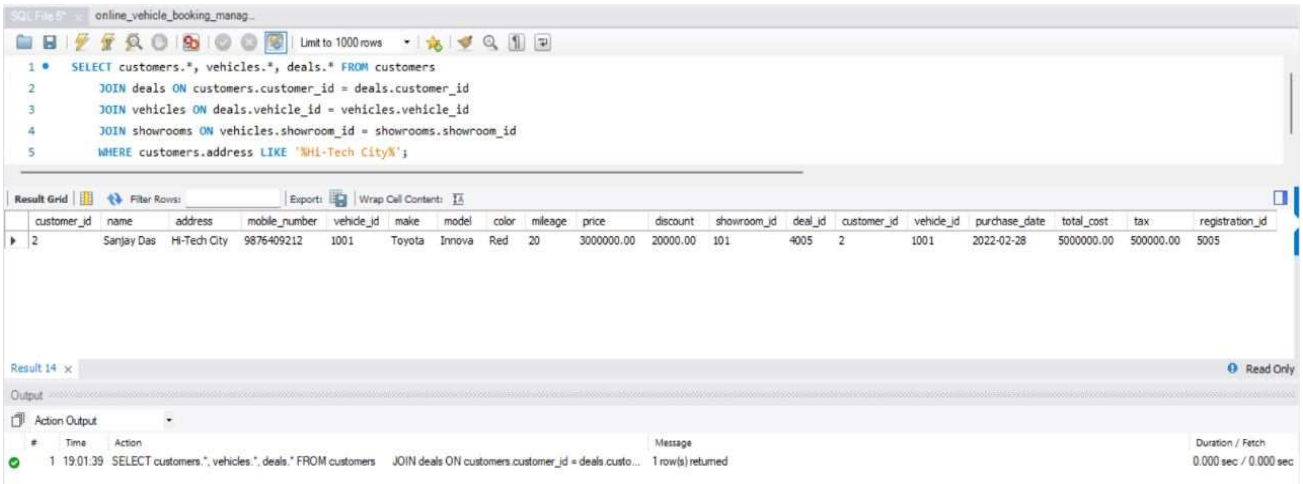
#	Time	Action	Message
1	18:59:55	SELECT customers.*, vehicles.* FROM customers JOIN deals ON customers.customer_id = deals.custo...	5 row(s) returned

➢ RA QUERY: πcustomers.customer_id, customers.customer_name, customers.customer_address, customers.customer_phone, vehicles.vehicle_id,

vehicles.make, vehicles.model, vehicles.year, vehicles.mileage (customers ⋈ deals) ⋈ vehicles

12. Find all deals made by customers who live in Hi-Tech City, along with the details of the vehicle they purchased:

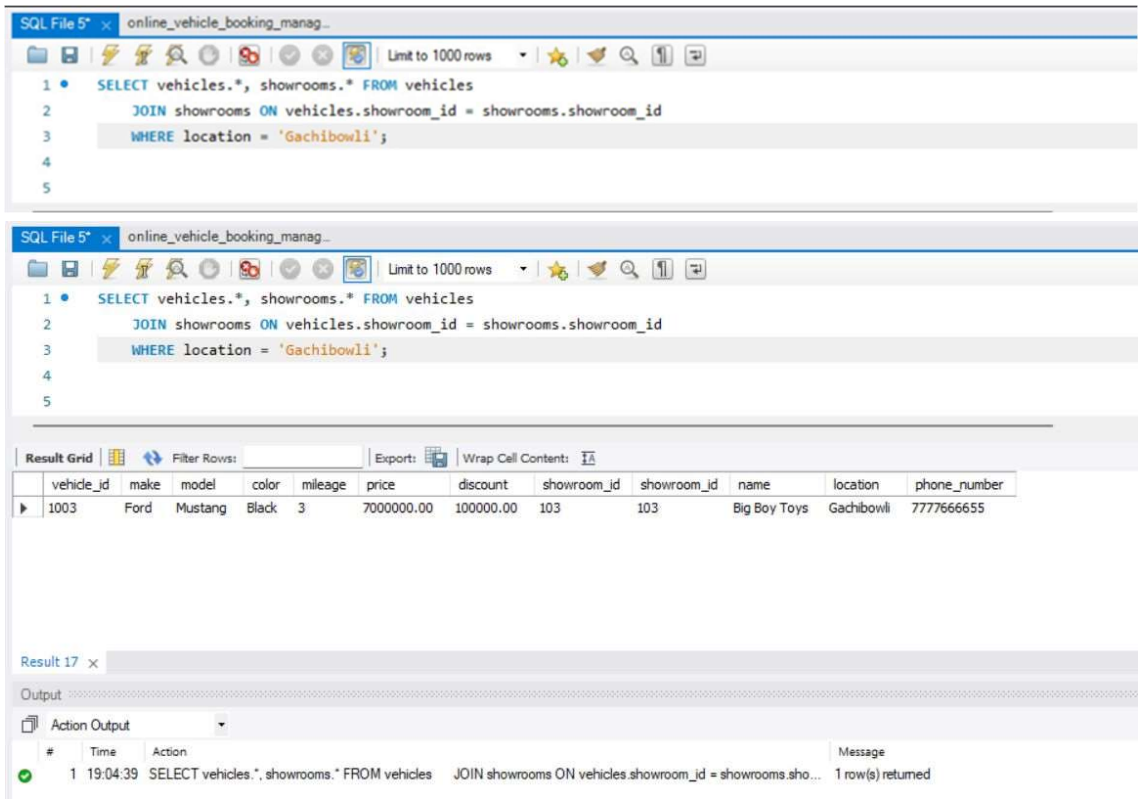
- SELECT customers.*, vehicles.*, deals.* FROM customers
  JOIN deals ON customers.customer_id = deals.customer_id
  JOIN vehicles ON deals.vehicle_id = vehicles.vehicle_id
  JOIN showrooms ON vehicles.showroom_id = showrooms.showroom_id
  WHERE customers.address LIKE '%Hi-Tech City%';



- RA QUERY: πcustomers.customer_id, customers.customer_name, customers.customer_address, customers.customer_phone, vehicles.vehicle_id, vehicles.make, vehicles.model, vehicles.year, vehicles.mileage, deals.deal_id, deals.customer_id, deals.vehicle_id, deals.deal_date, deals.price, deals.payment_method, deals.payment_status (customers ⋈ deals) ⋈ vehicles ⋈ showrooms | customers.address LIKE '%Hi-Tech City%'

13. Find all vehicles and their showroom details for showrooms located in Gachibowli:

- SELECT customers.*, vehicles.*, deals.* FROM customers
  JOIN deals ON customers.customer_id = deals.customer_id
  JOIN vehicles ON deals.vehicle_id = vehicles.vehicle_id
  JOIN showrooms ON vehicles.showroom_id = showrooms.showroom_id
  WHERE customers.address LIKE '%Gachibowli%';



- RA QUERY: π_customers., vehicles., deals.* (σ_customers.address LIKE '%Gachibowli%' (customers ⋈_customers.customer_id = deals.customer_id deals ⋈_deals.vehicle_id = vehicles.vehicle_id vehicles ⋈_vehicles.showroom_id = showrooms.showroom_id showrooms))

14. Find all vehicles with a price greater than Rs 50,000, along with the name and location of the showroom selling the vehicle:

- SELECT vehicles.*, showrooms.name, showrooms.location FROM vehicles JOIN showrooms ON vehicles.showroom_id = showrooms.showroom_id WHERE price > 50000;

- RA QUERY: π_vehicles.*, showrooms.name, showrooms.location (σ_price > 50000 (vehicles ⋈_vehicles.showroom_id = showrooms.showroom_id showrooms))

SQL File 5* | online_vehicle_booking_manag...

Limit to 1000 rows

```
SELECT vehicles.*, showrooms.name, showrooms.location FROM vehicles
    JOIN showrooms ON vehicles.showroom_id = showrooms.showroom_id
    WHERE price > 50000;
```

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

vehicle_id	make	model	color	mileage	price	discount	showroom_id	name	location
1001	Toyota	Innova	Red	20	3000000.00	20000.00	101	Silver Motors	Film Nagar
1002	Honda	City	White	15	2500000.00	50000.00	102	Apachi Autos	Jubliee Hills
1003	Ford	Mustang	Black	3	7000000.00	100000.00	103	Big Boy Toys	Gachibowli
1004	Nissan	Sunny	Blue	16	3000000.00	40000.00	104	Kun Exclusive	Banjara Hills
1005	Mercedes	Benz-C-Class	Grey	7	5000000.00	70000.00	105	Sky Motors	Kukatpally

Result 18

Output

Action Output

#	Time	Action	Message
1	19:06:08	SELECT vehicles.*, showrooms.name, showrooms.location FROM vehicles JOIN showrooms ON vehicles.showr...	5 row(s) returned

15. Find the List of customer names, vehicle makes, and vehicle models for all the deals in the database, where each deal corresponds to a customer purchasing a vehicle:

- SELECT customers.customer_name, vehicles.vehicle_make, vehicles.vehicle_model
  FROM customers
  JOIN deals ON customers.customer_id = deals.customer_id
  JOIN vehicles ON deals.vehicle_id = vehicles.vehicle_id;

SQL File 5*

Limit to 1000 rows

```
SELECT customers.name, vehicles.make, vehicles.model
FROM customers
JOIN deals ON customers.customer_id = deals.customer_id
JOIN vehicles ON deals.vehicle_id = vehicles.vehicle_id;
```

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

name	make	model
Abdul Aziz	Ford	Mustang
Sanjay Das	Toyota	Innova
Badrinath	Honda	City
Sohan Reddy	Nissan	Sunny
Hemanth Reddy	Mercedes	Benz-C-Class

Result 2

Output

Action Output

#	Time	Action	Message
1	19:11:27	SELECT customers.name, vehicles.make, vehicles.model FROM customers JOIN deals ON customers.customer_id ...	5 row(s) returned

- RA QUERY: π_customers.customer_name, vehicles.vehicle_make, vehicles.vehicle_model (customers ⋈_customers.customer_id = deals.customer_id deals ⋈_deals.vehicle_id = vehicles.vehicle_id vehicles)

FUNCTIONAL DEPENDENCY & NORMALIZATION:

Here are the functional dependencies for each table:

Table: customers

customer_id -> name, address, mobile_number
mobile_number -> customer_id

This table is already in 3NF.

Table: vehicles

vehicle_number -> make, model_name, color, mileage, price, discount, showroom_id

This table is already in 3NF.

Table: showrooms

showroom_id -> name, location, phone_number

This table is already in 3NF.

Table: deals

deal_id -> customer_id, vehicle_number, purchase_date, total_cost

This table is already in 3NF.

Table: maintenance

maintenance_id -> vehicle_number, date, mileage, maintenance_type, cost
This table is already in 3NF.

Table: insurance

insurance_id -> vehicle_number, coverage

This table is already in 3NF.

Table: taxes

tax_id -> vehicle_number, tax_amount

This table is already in 3NF.

Table: registrations

registration_id -> vehicle_number, customer_id, registration_date, driver_license_number

This table is already in 3NF.

Table: resales

resale_id -> customer_id, vehicle_number, condition

This table is already in 3NF.

All of the tables are already in 3NF, so there is no need to further normalize them.