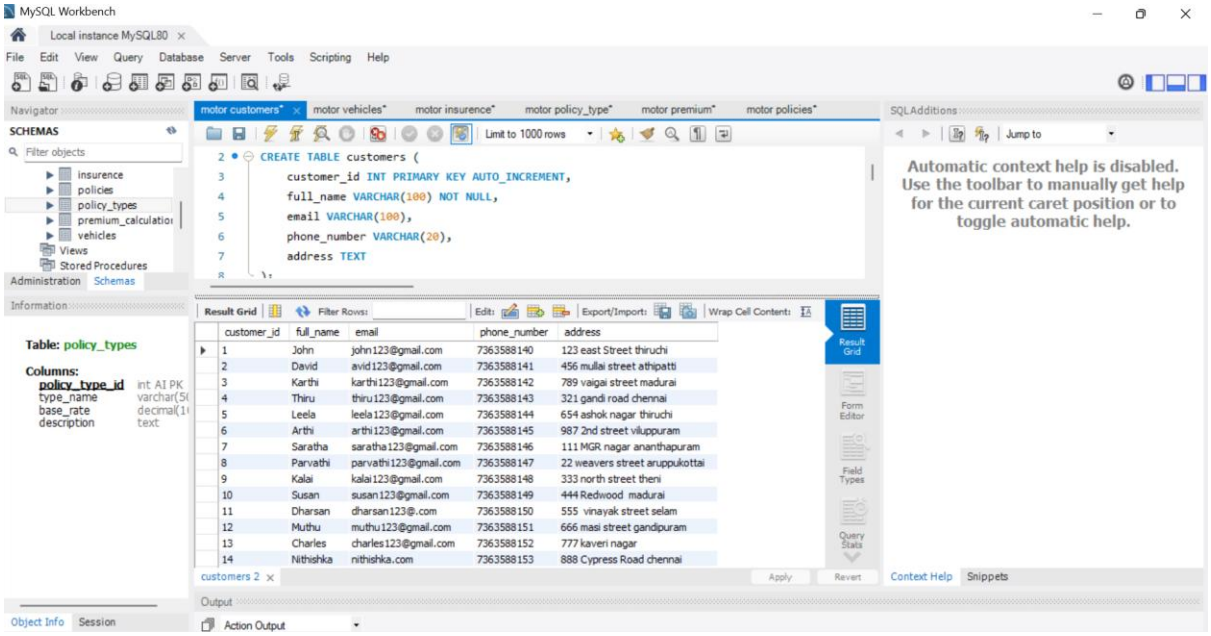
MOTOR INSURANCE PROJECT

TABLE CREATION:-

1.CUSTOMERS TABLE:

QUERY WITH OUTPUT:

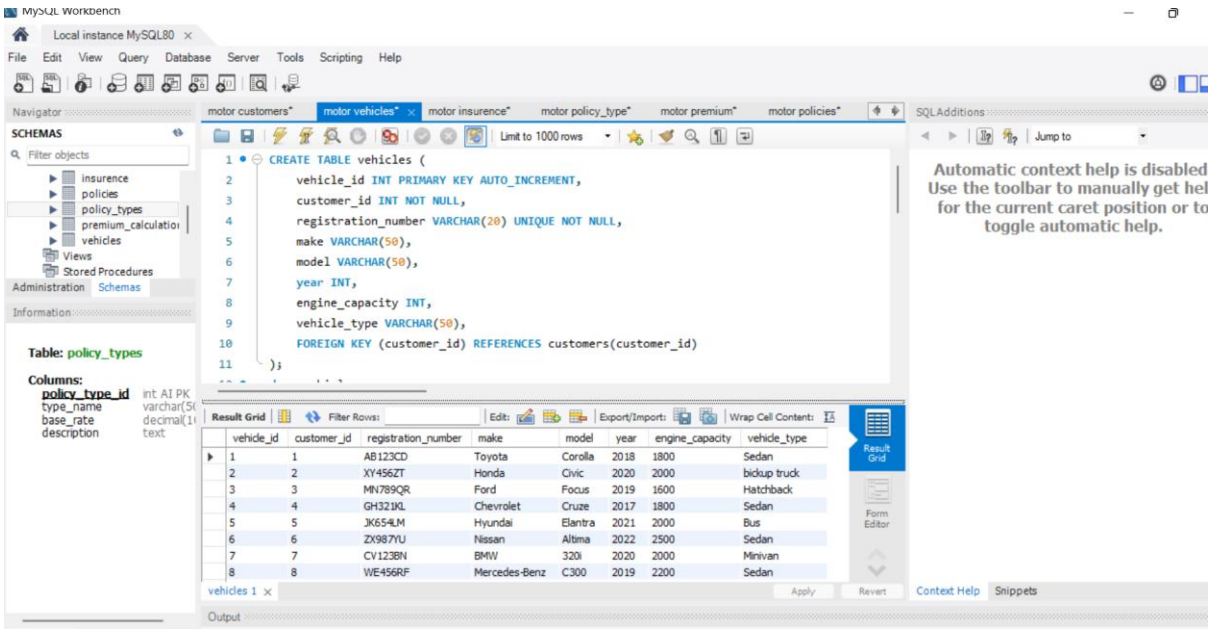
```
CREATE TABLE customers (
   customer_id INT PRIMARY KEY AUTO_INCREMENT,
   full_name VARCHAR(100) NOT NULL,
   email VARCHAR(100),
   phone_number VARCHAR(20),
   address TEXT
);
```



2.VEHICLES TABLE:

QUERY WITH OUTPUT:

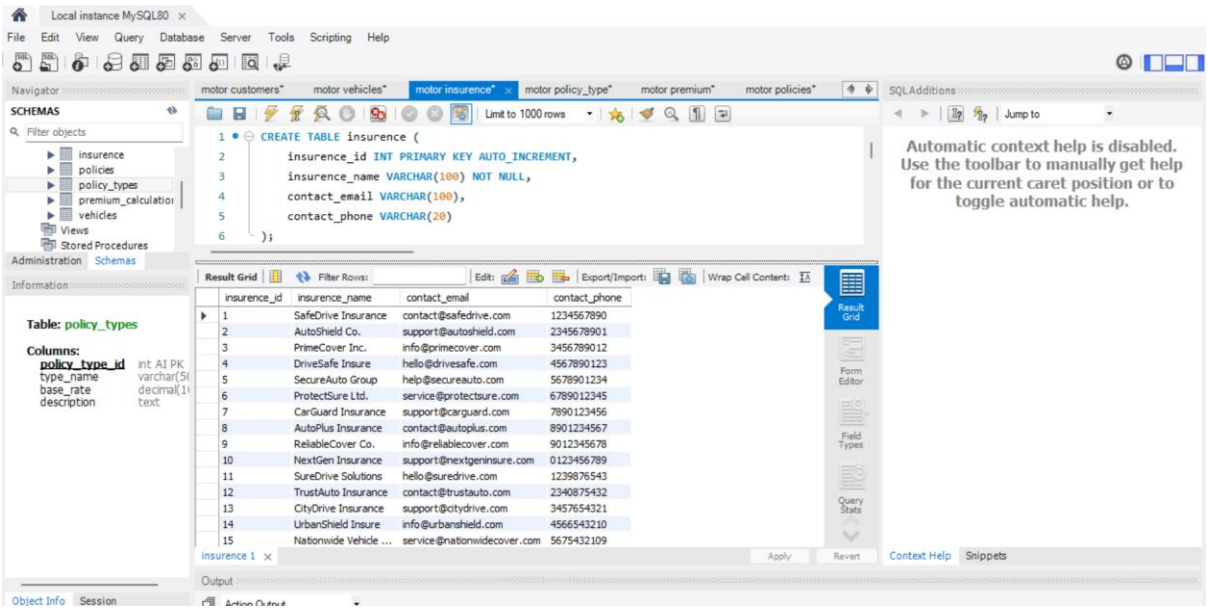
```
CREATE TABLE vehicles (
    vehicle_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT NOT NULL,
    registration_number VARCHAR(20) UNIQUE NOT NULL,
    make VARCHAR(50),
    model VARCHAR(50),
    year INT,
    engine_capacity INT,
    vehicle_type VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);
```



3.INSURENCE TABLE:

QUERY WITH OUTPUT:

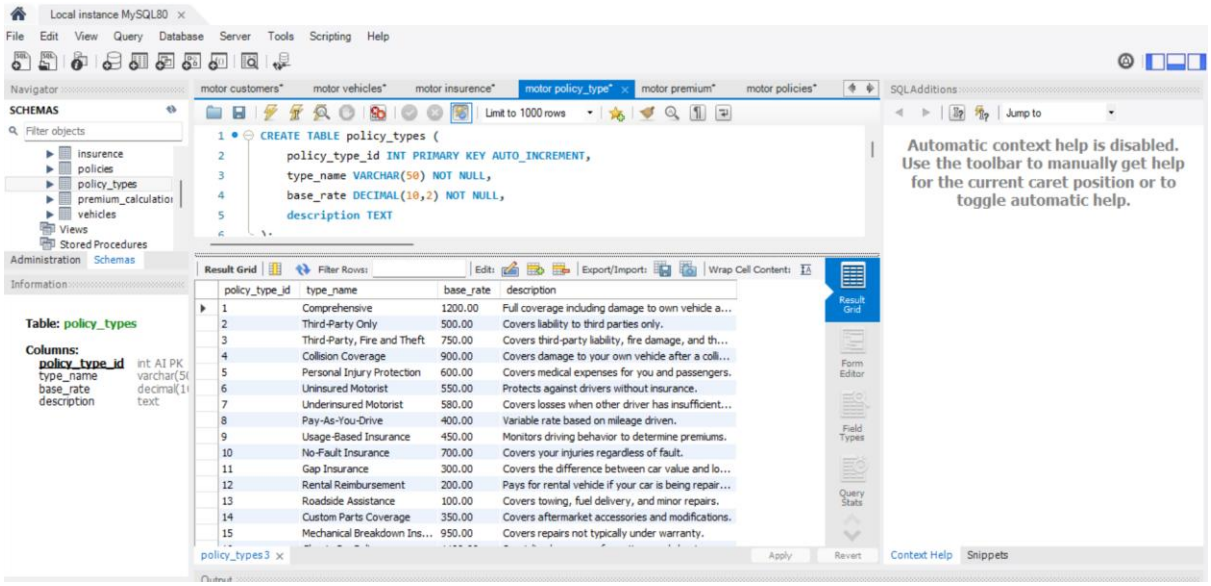
```
CREATE TABLE insurence (
    insurence_id INT PRIMARY KEY AUTO_INCREMENT,
    insurence_name VARCHAR(100) NOT NULL,
    contact_email VARCHAR(100),
    contact_phone VARCHAR(20)
);
```



4.POLICY_TYPE TABLE:

QUERY WITH OUTPUT:

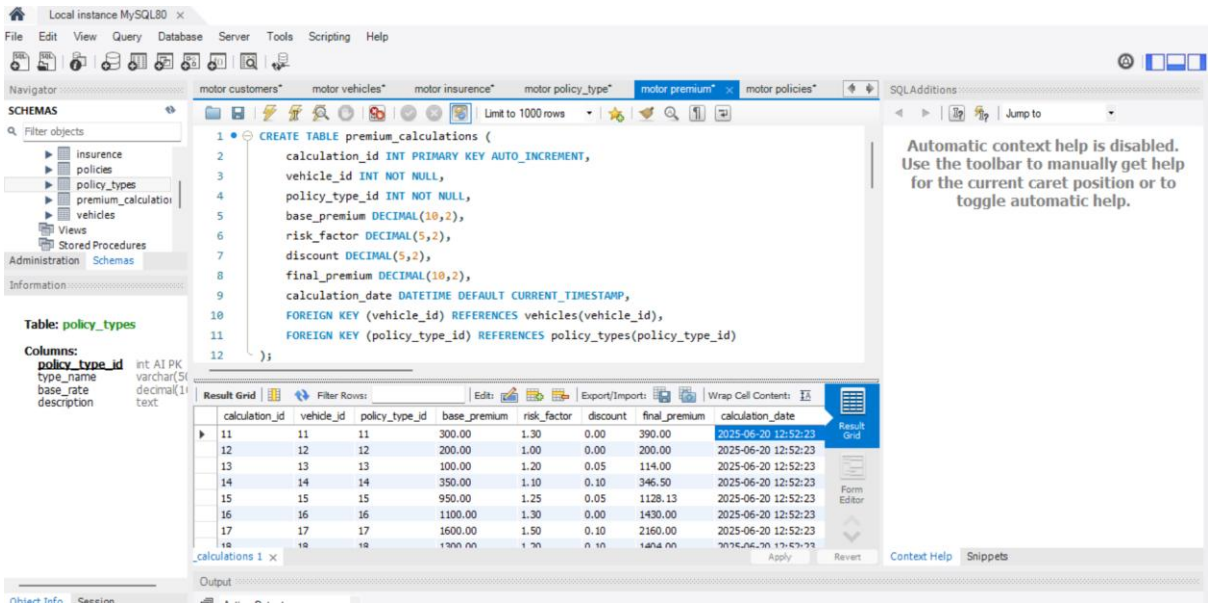
```
CREATE TABLE policy_types (
    policy_type_id INT PRIMARY KEY AUTO_INCREMENT,
    type_name VARCHAR(50) NOT NULL,
    base_rate DECIMAL(10,2) NOT NULL,
    description TEXT
);
```



5.PREMIUM_CALCULATION TABLE:

QUERY WITH OUTPUT:

```
CREATE TABLE premium_calculations (
    calculation_id INT PRIMARY KEY AUTO_INCREMENT,
    vehicle_id INT NOT NULL,
    policy_type_id INT NOT NULL,
    base_premium DECIMAL(10,2),
    risk_factor DECIMAL(5,2),
    discount DECIMAL(5,2),
    final_premium DECIMAL(10,2),
    calculation_date DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (vehicle_id) REFERENCES vehicles(vehicle_id),
    FOREIGN KEY (policy_type_id) REFERENCES
policy_types(policy_type_id)
);
```



6.POLICIES TABLE:

QUERY WITH OUTPUT:

CREATE TABLE policies (

policy_id INT PRIMARY KEY AUTO_INCREMENT,

policy_number VARCHAR(50) UNIQUE NOT NULL,

customer_id INT NOT NULL,

vehicle_id INT NOT NULL,

insurence_id INT NOT NULL,

policy_type_id INT NOT NULL,

premium_id INT NOT NULL,

start_date DATE NOT NULL,

end_date DATE NOT NULL,

policy_status VARCHAR(20) DEFAULT 'Issued',

FOREIGN KEY (customer_id) REFERENCES customers(customer_id),

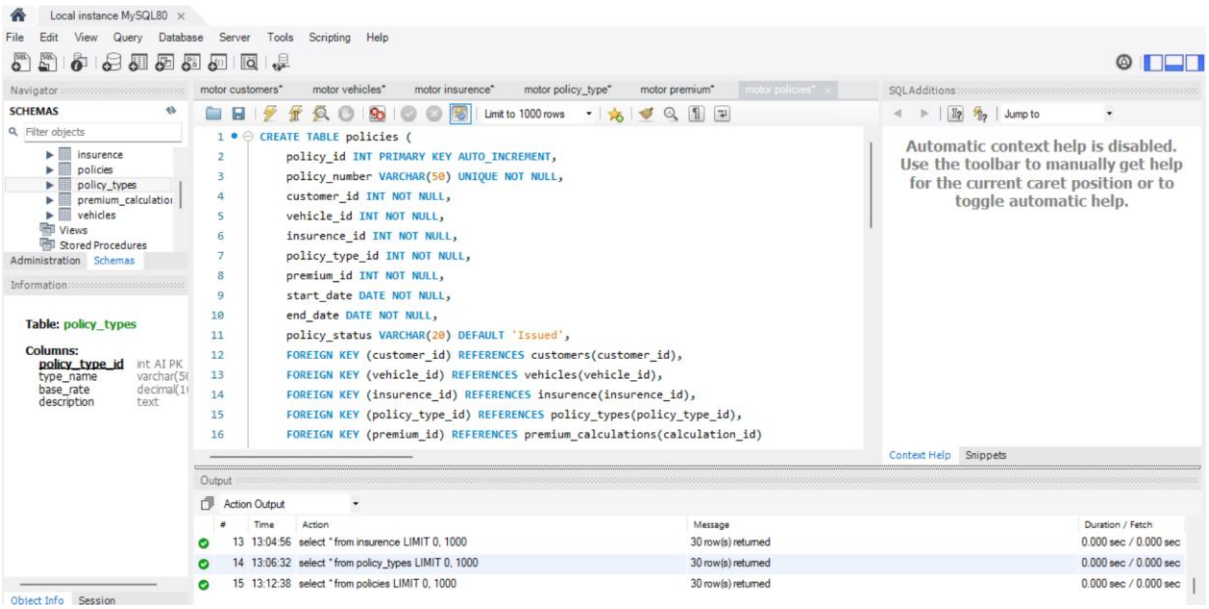
FOREIGN KEY (vehicle_id) REFERENCES vehicles(vehicle_id),

FOREIGN KEY (insurence_id) REFERENCES insurence(insurence_id),

FOREIGN KEY (policy_type_id) REFERENCES policy_types(policy_type_id),

FOREIGN KEY (premium_id) REFERENCES premium_calculations(calculation_id)

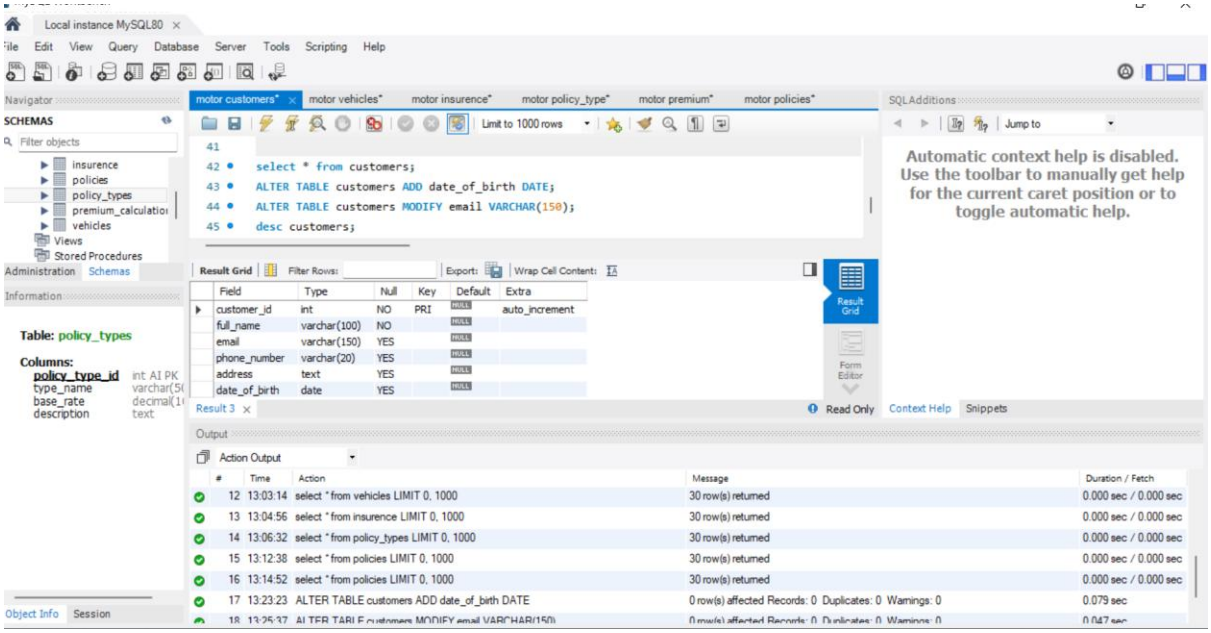
);



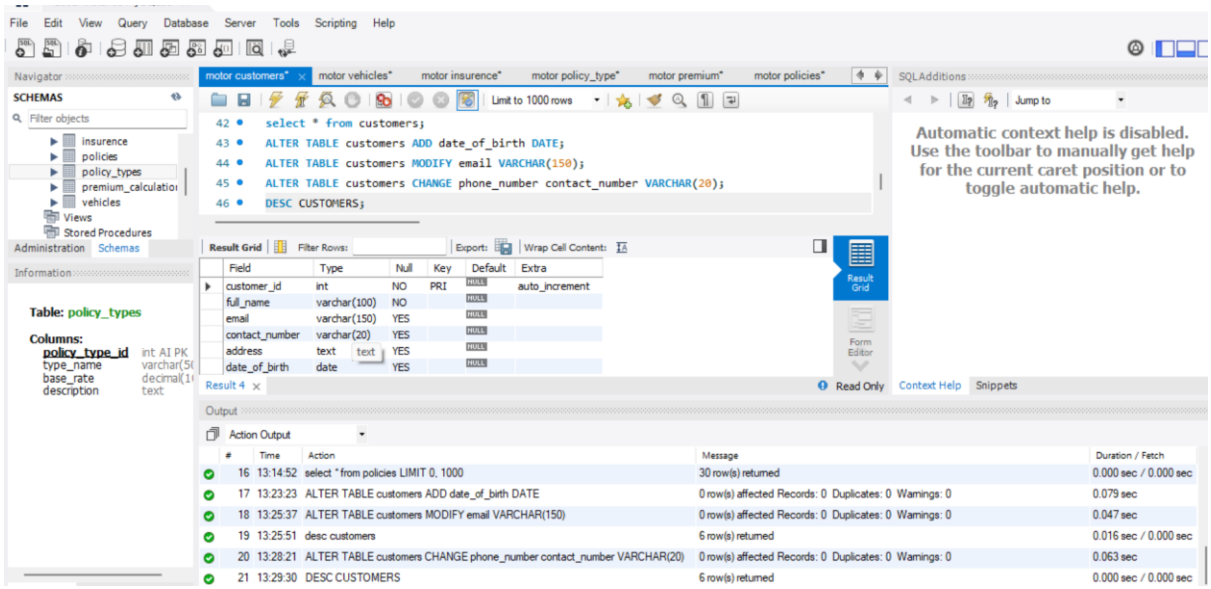
ALTER TABLE:

QUERY WITH OUTPUT:

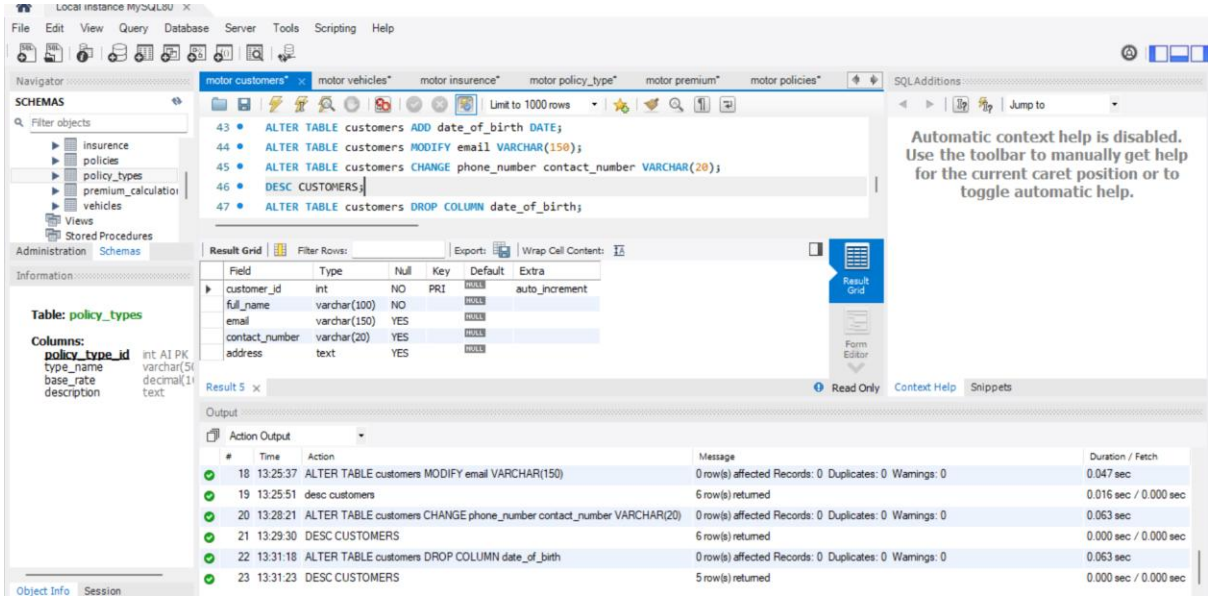
ADD,MODIFY COLUMN:



CHANGE COLUMN:

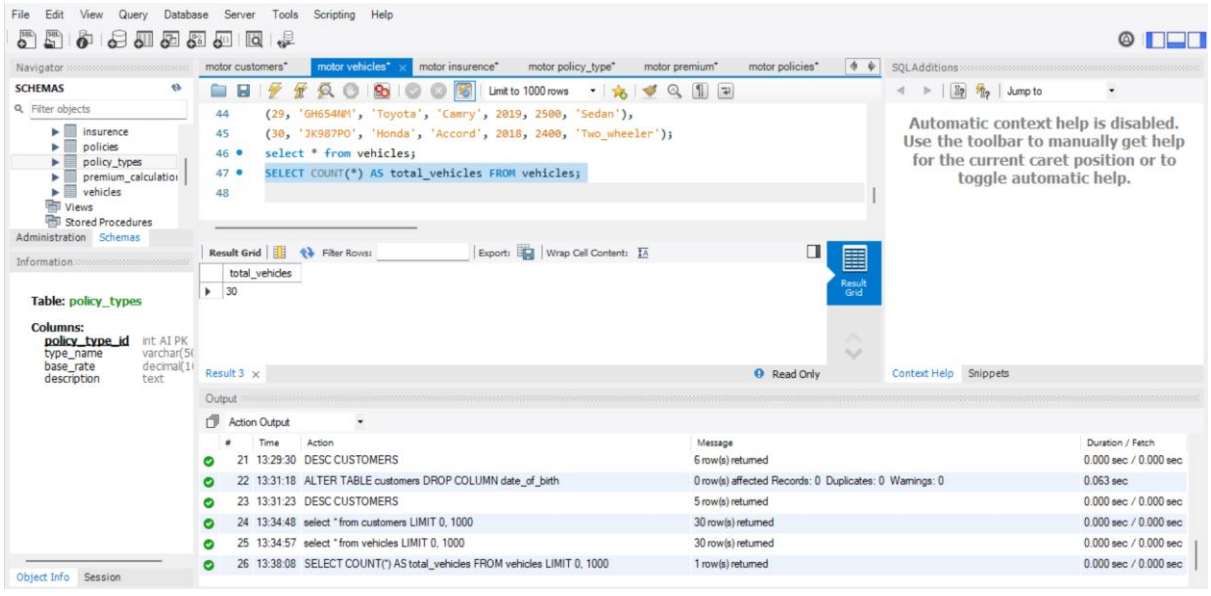


DROP COLUMN

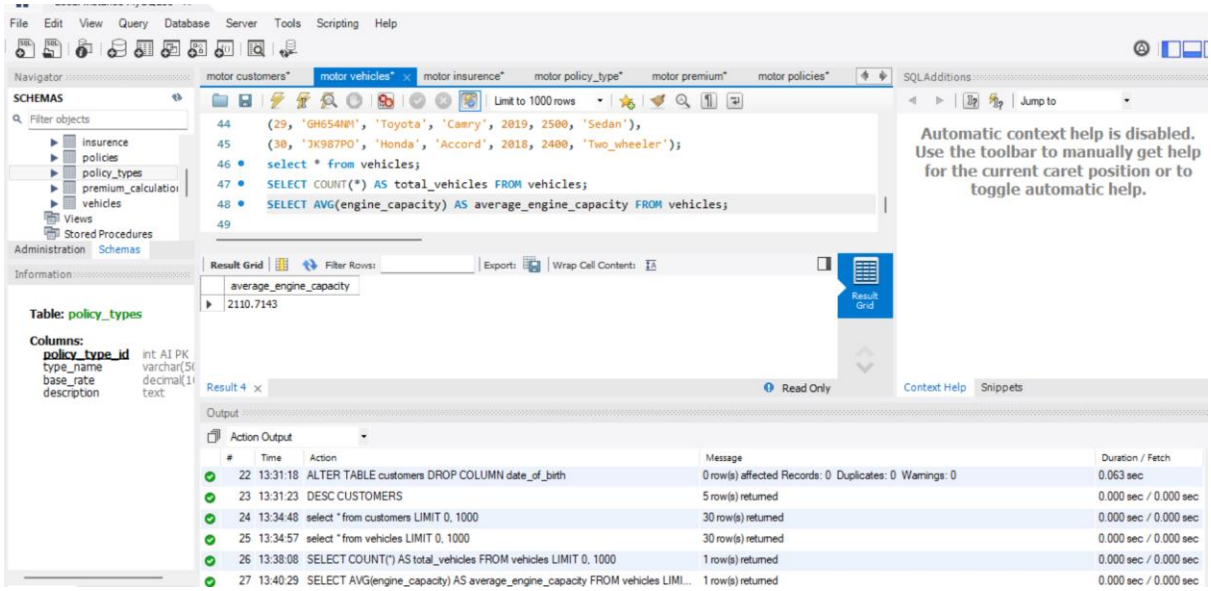


AGGREGATE FUNCTION:

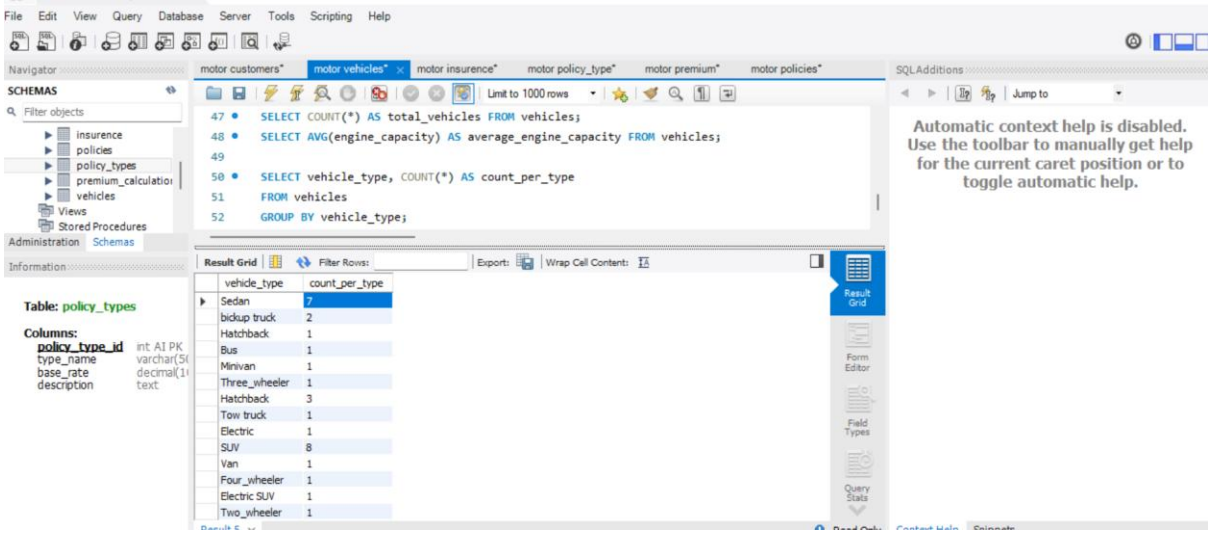
1.COUNT TOTAL NUMBER OF VEHICLES:



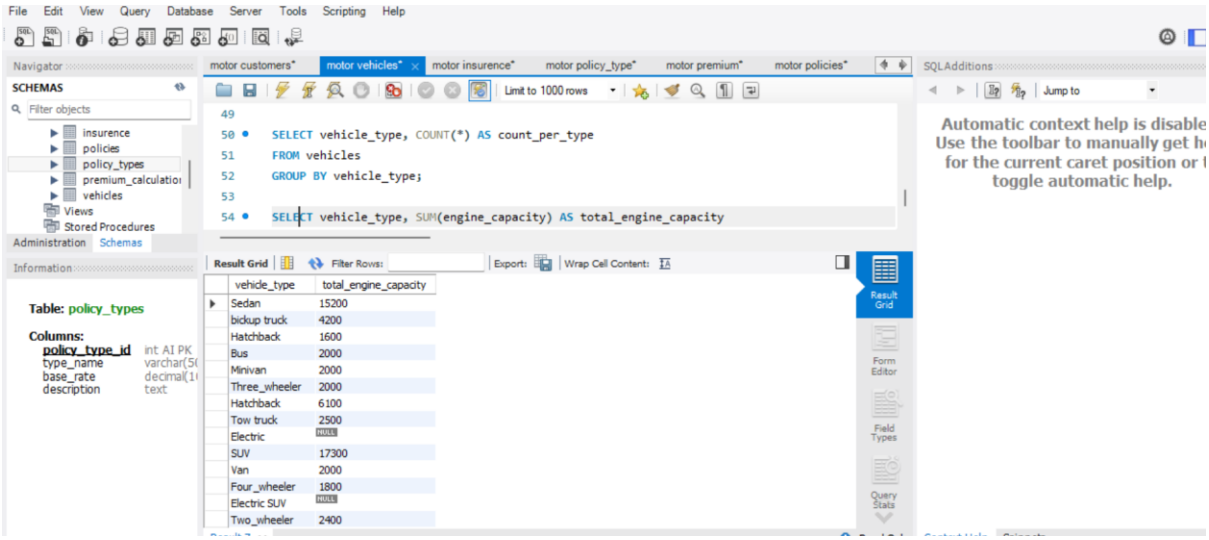
2.AVG ENGINE CAPACITY:



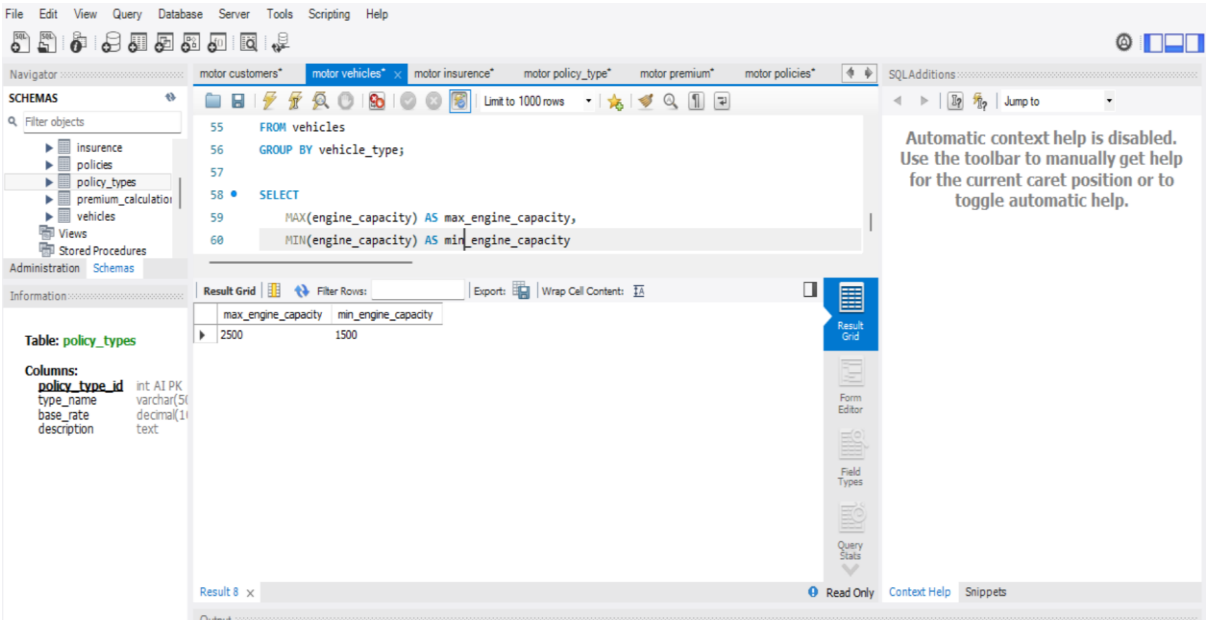
3.TOTAL NUMBER OF VEHICLES PER VEHICLE TYPE:



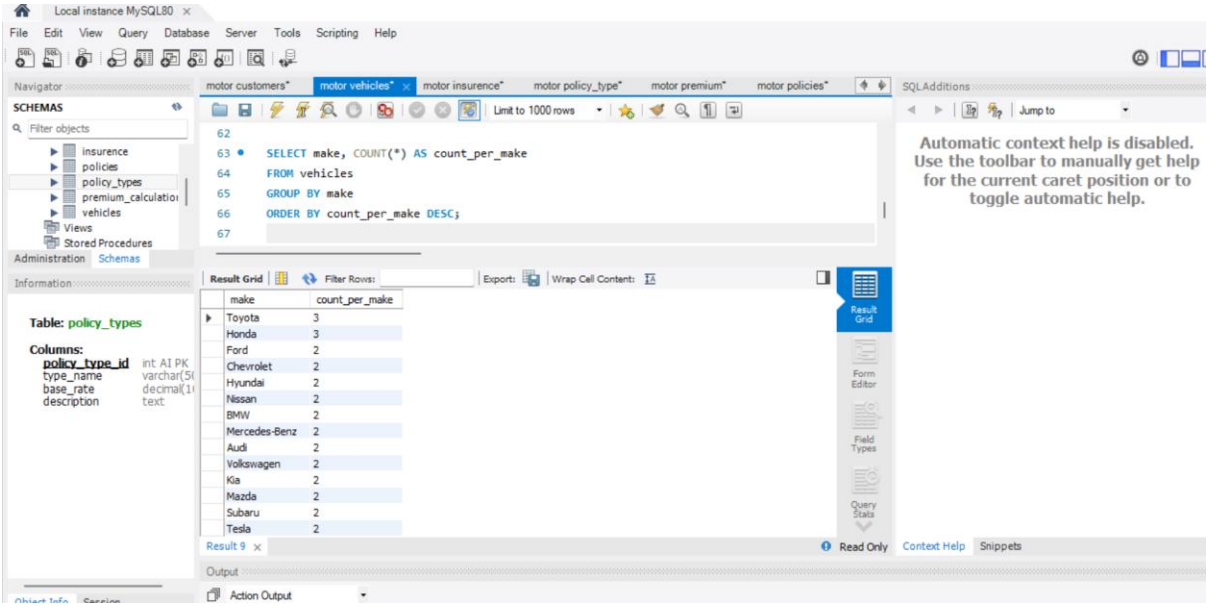
4.TOTAL ENGINE CAPACITY PER VEHICLES TYPE:



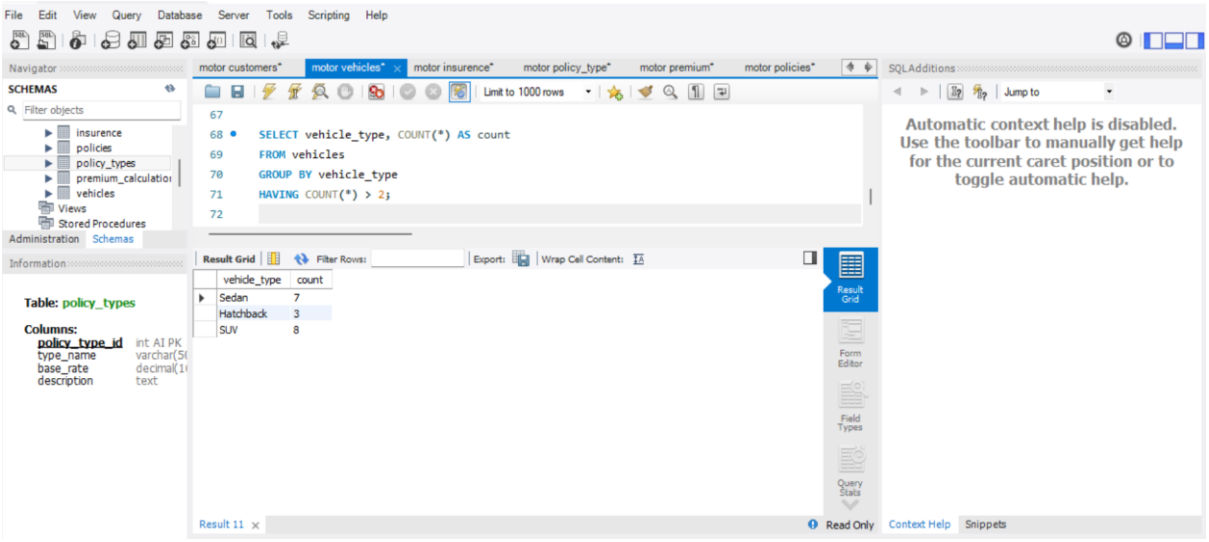
5. MAXIMUM AND MINIMUM ENGINE CAPACITY:



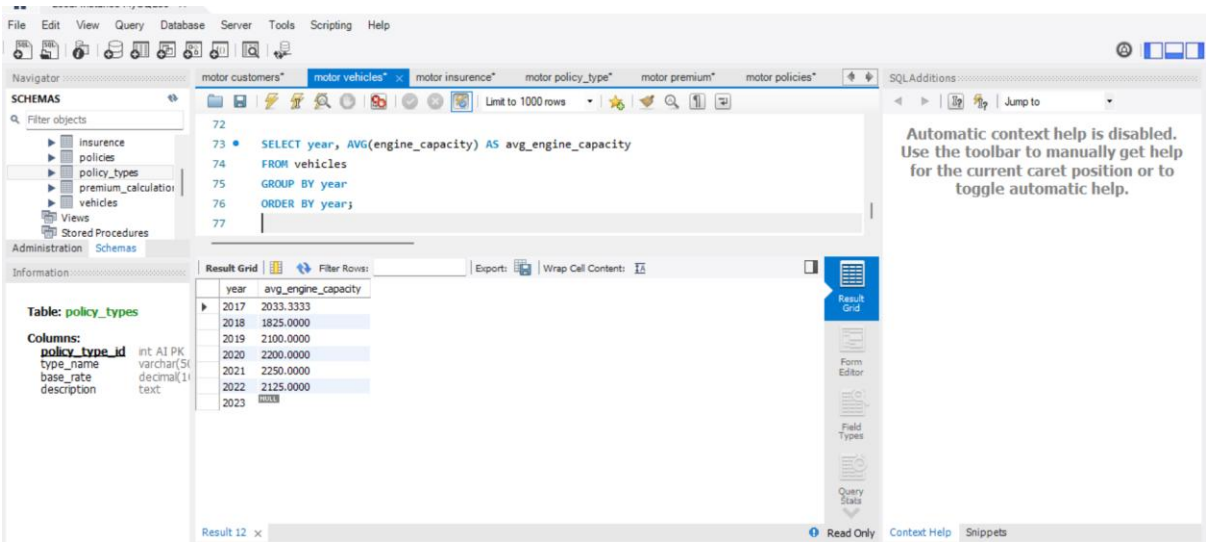
6.COUNT VEHICLES PER MAKE



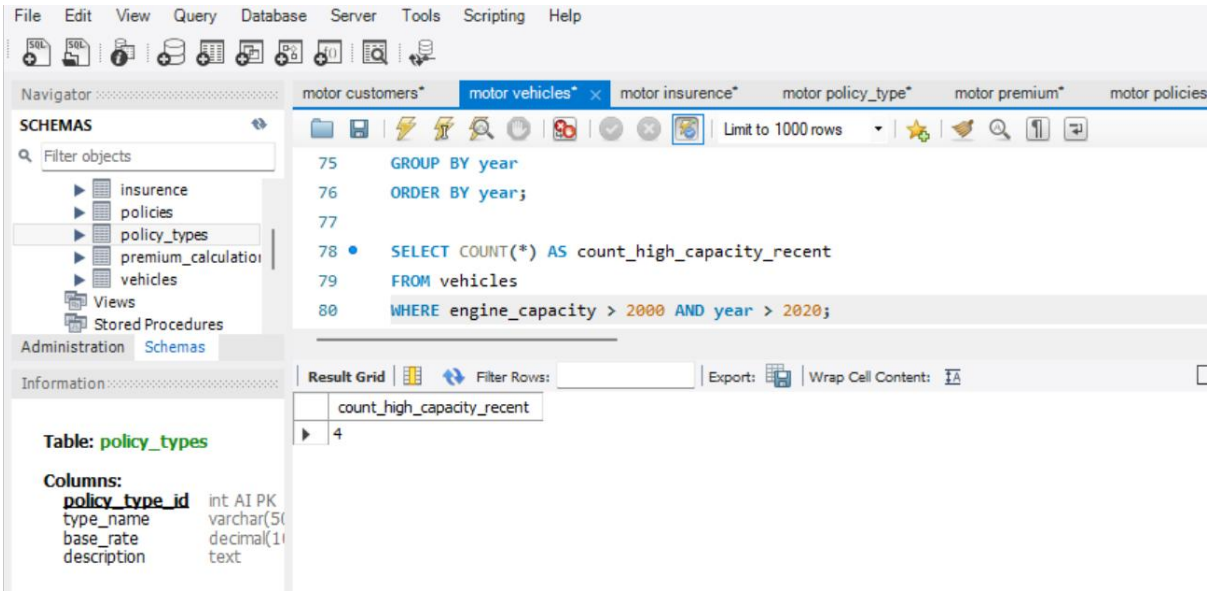
7.FIND VEHICLES TYPE WITH MORE THAN 2 ENTIRES

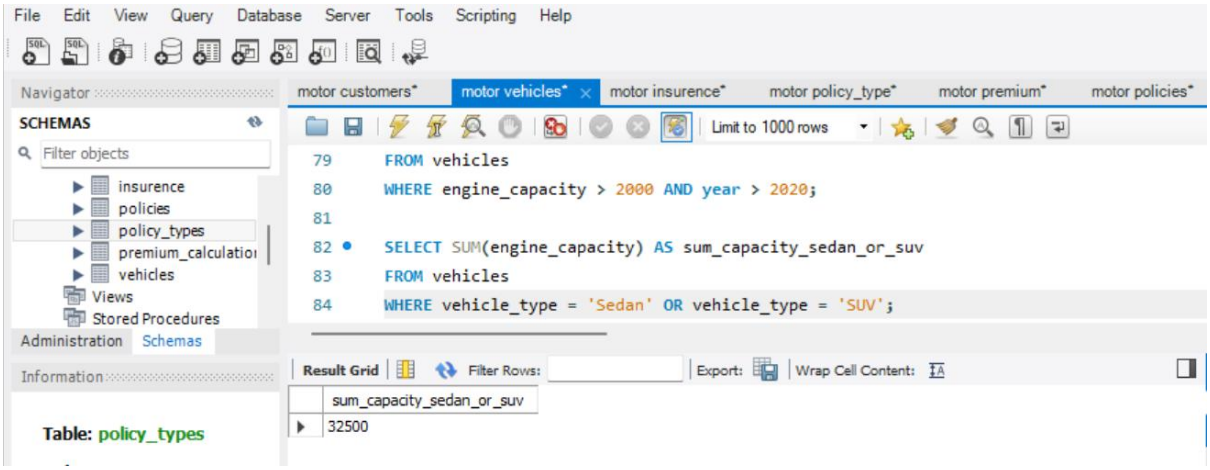


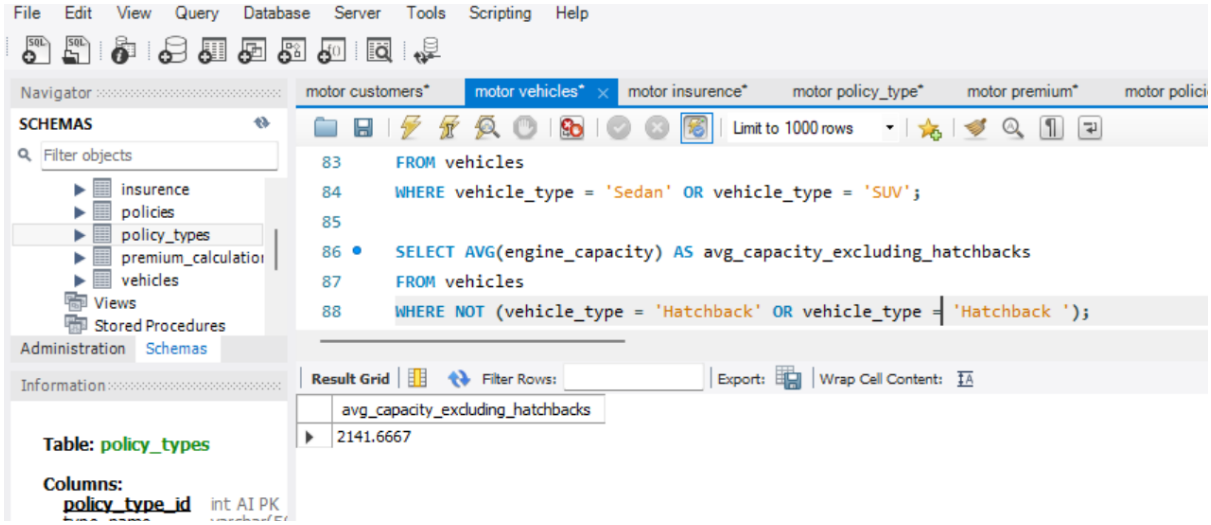
8. AVERAGE ENGINE CAPACITY PER YEAR

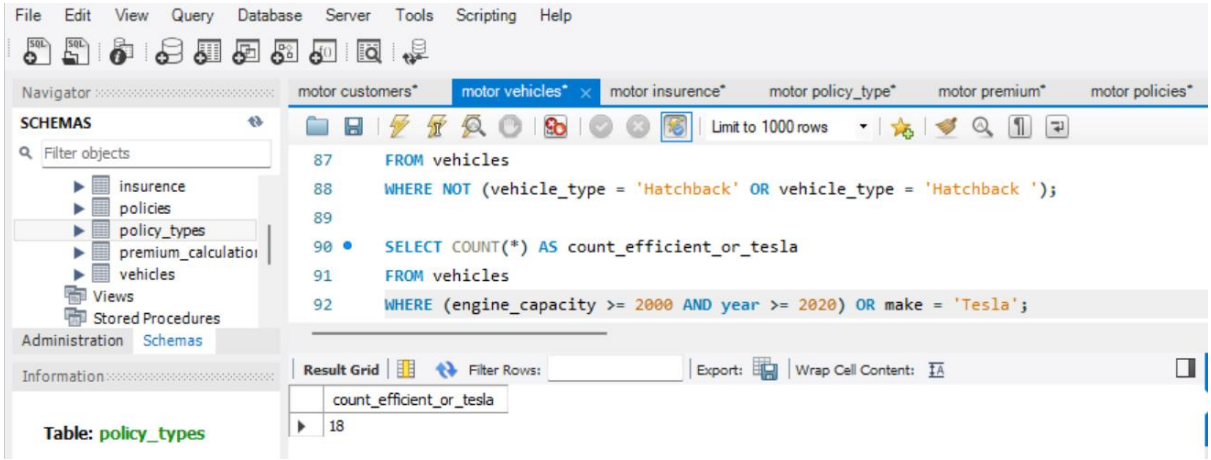


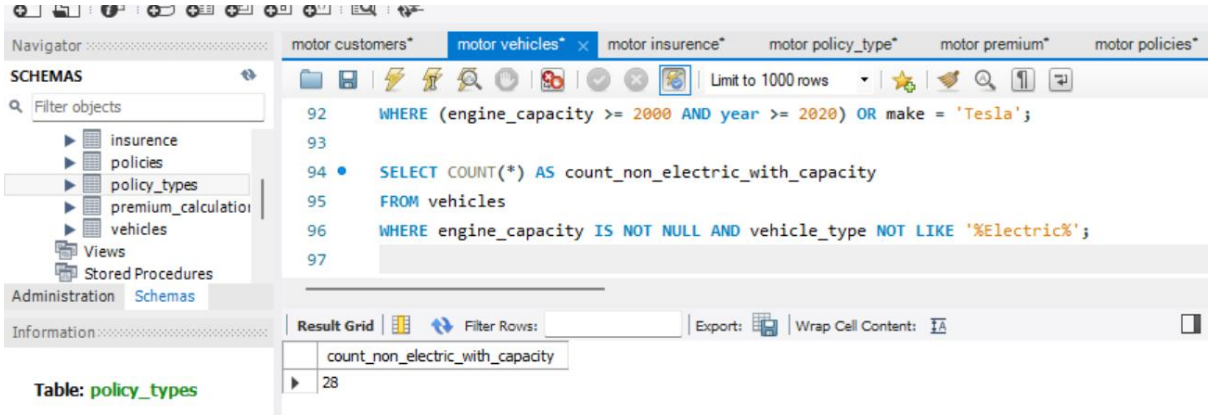
USING OPERATORS:

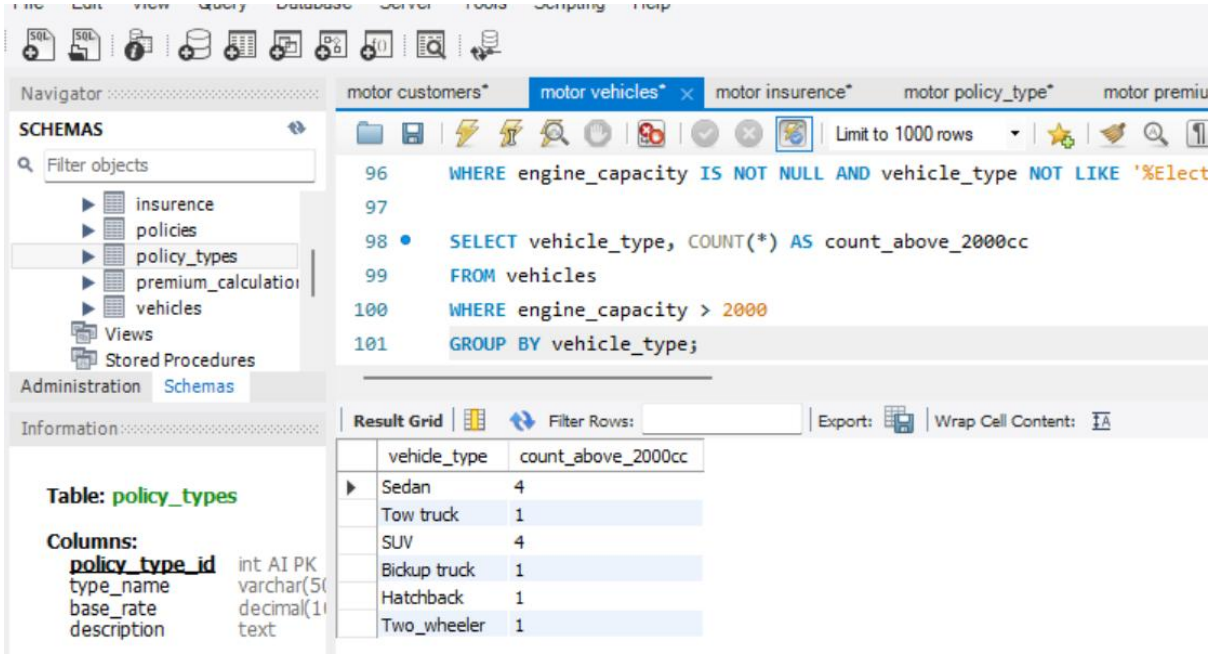




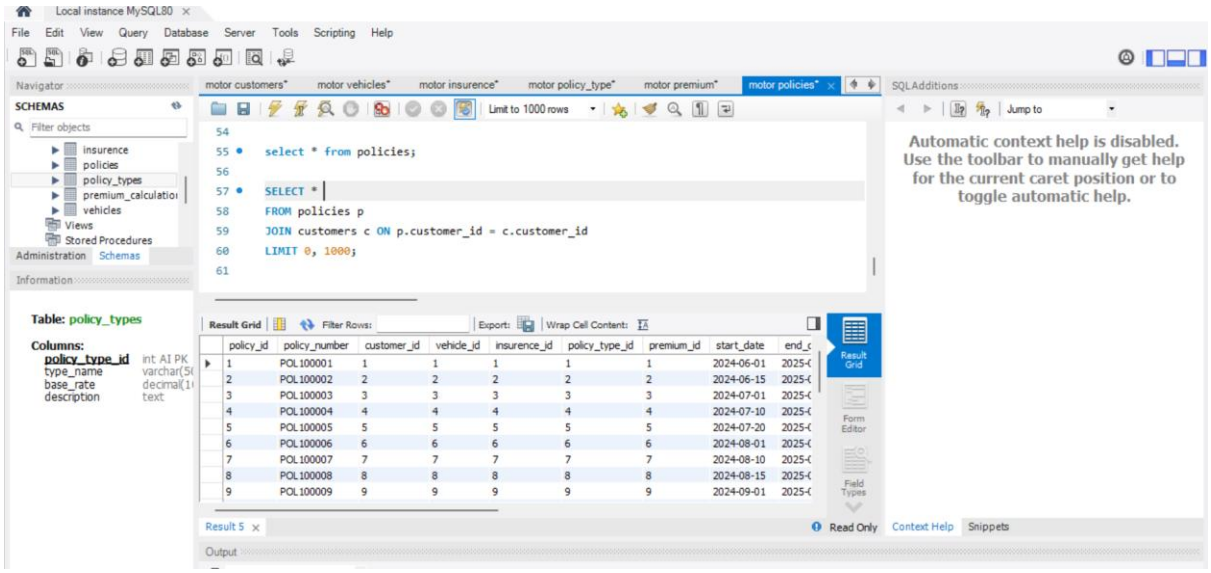

File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
insurance
policies
policy_types
premium_calculatio
vehicles
Views
Stored Procedures
Administration Schemas
Information
Table: policy_types
Columns:
policy_type_id int AI PK
motor customers* motor vehicles* motor insurance* motor policy_type* motor premium* motor polici
Limit to 1000 rows
83 FROM vehicles
84 WHERE vehicle_type = 'Sedan' OR vehicle_type = 'SUV';
85
86 SELECT AVG(engine_capacity) AS avg_capacity_excluding_hatchbacks
87 FROM vehicles
88 WHERE NOT (vehicle_type = 'Hatchback' OR vehicle_type = 'Hatchback ');
Result Grid Filter Rows: Export: Wrap Cell Content:
avg_capacity_excluding_hatchbacks
2141.6667


File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
insurance
policies
policy_types
premium_calculatio
vehicles
Views
Stored Procedures
Administration Schemas
Information
Table: policy_types
motor customers* motor vehicles* motor insurance* motor policy_type* motor premium* motor policies*
Limit to 1000 rows
87 FROM vehicles
88 WHERE NOT (vehicle_type = 'Hatchback' OR vehicle_type = 'Hatchback ');
89
90 SELECT COUNT(*) AS count_efficient_or_tesla
91 FROM vehicles
92 WHERE (engine_capacity >= 2000 AND year >= 2020) OR make = 'Tesla';
Result Grid Filter Rows: Export: Wrap Cell Content:
count_efficient_or_tesla
18

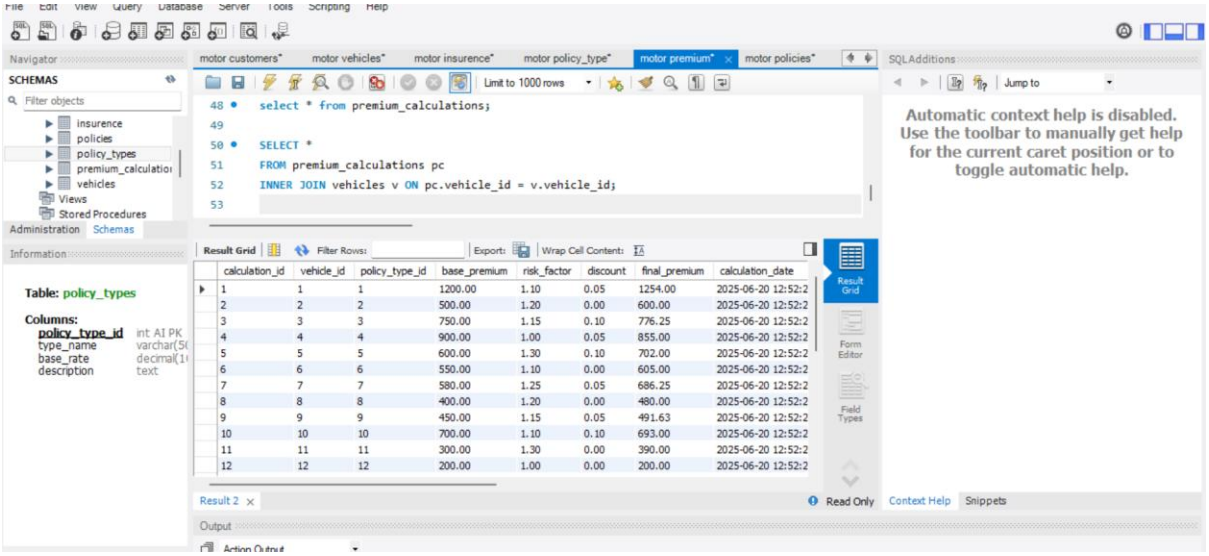

Navigator
SCHEMAS
Filter objects
insurance
policies
policy_types
premium_calculatio
vehicles
Views
Stored Procedures
Administration Schemas
Information
Table: policy_types
motor customers* motor vehicles* motor insurance* motor policy_type* motor premium* motor policies*
Limit to 1000 rows
92 WHERE (engine_capacity >= 2000 AND year >= 2020) OR make = 'Tesla';
93
94 SELECT COUNT(*) AS count_non_electric_with_capacity
95 FROM vehicles
96 WHERE engine_capacity IS NOT NULL AND vehicle_type NOT LIKE '%Electric%';
97
Result Grid Filter Rows: Export: Wrap Cell Content:
count_non_electric_with_capacity
28

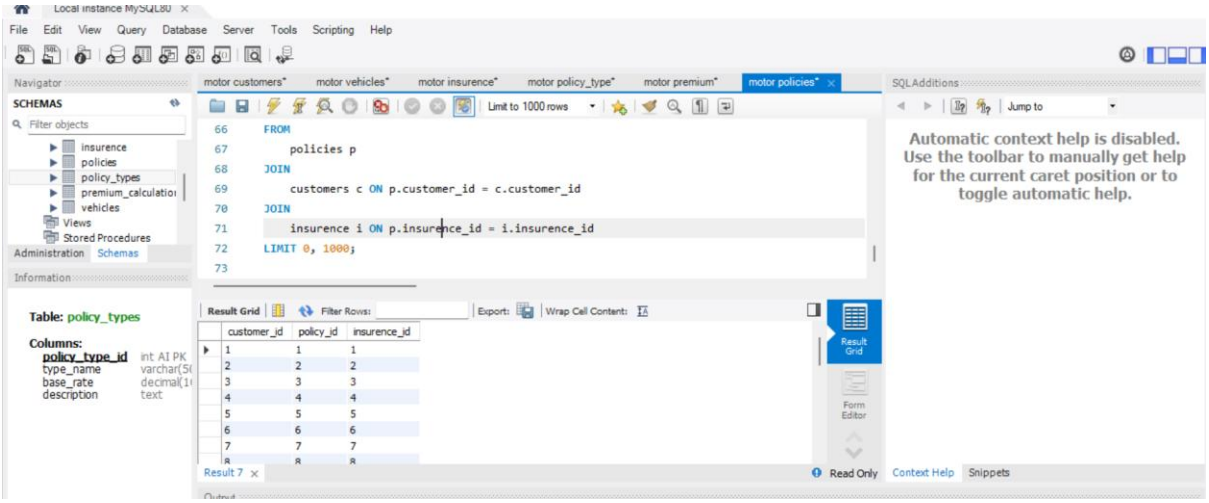

motor customers*
motor vehicles*
motor insurence*
motor policy_type*
Limit to 1000 rows
96 WHERE engine_capacity IS NOT NULL AND vehicle_type NOT LIKE '%Elect
97
98 SELECT vehicle_type, COUNT(*) AS count_above_2000cc
99 FROM vehicles
100 WHERE engine_capacity > 2000
101 GROUP BY vehicle_type;
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
vehicle_type
count_above_2000cc
Sedan 4
Tow truck 1
SUV 4
Bickup truck 1
Hatchback 1
Two_wheeler 1
Table: policy_types
Columns:
policy_type_id int AI PK
type_name varchar(5
base_rate decimal(1
description text

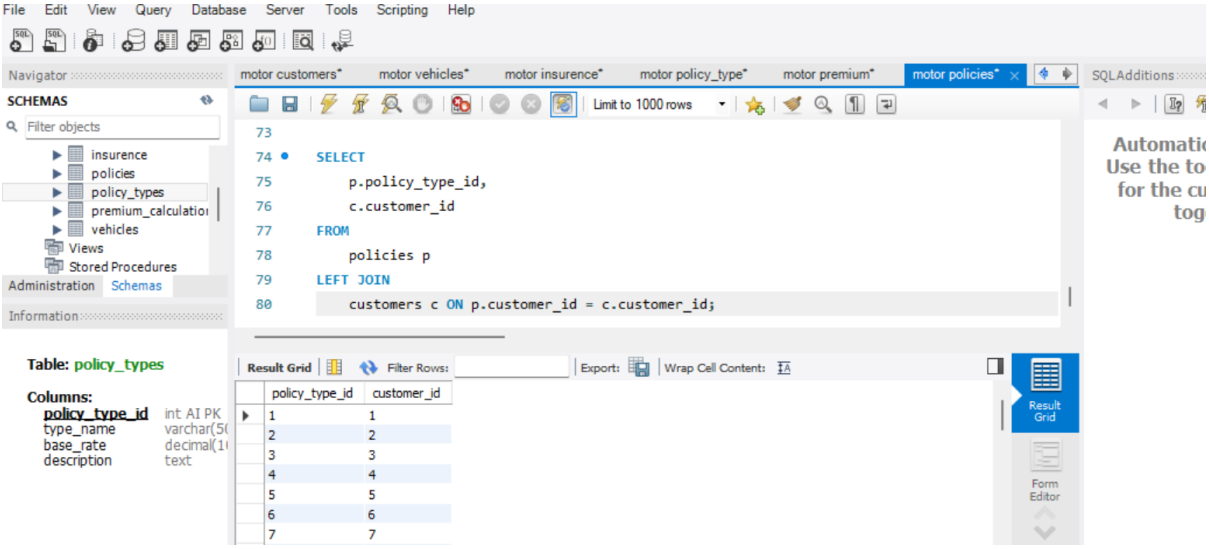
JOINS

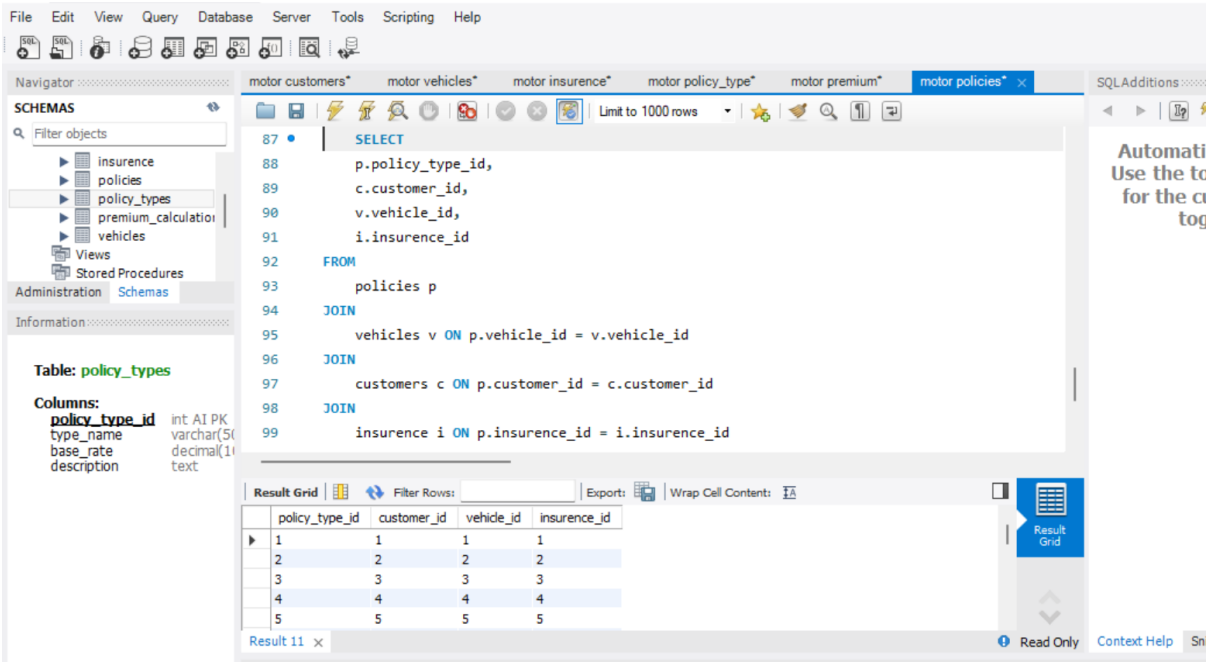

Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
motor customers*
motor vehicles*
motor insurence*
motor policy_type*
motor premium*
motor policies*
Limit to 1000 rows
54
55 select * from policies;
56
57 SELECT *
58 FROM policies p
59 JOIN customers c ON p.customer_id = c.customer_id
60 LIMIT 0, 1000;
61
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
policy_id policy_number customer_id vehicle_id insurance_id policy_type_id premium_id start_date
1 POL100001 1 1 1 1 1 2024-06-01
2 POL100002 2 2 2 2 2 2024-06-15
3 POL100003 3 3 3 3 3 2024-07-01
4 POL100004 4 4 4 4 4 2024-07-10
5 POL100005 5 5 5 5 5 2024-07-20
6 POL100006 6 6 6 6 6 2024-08-01
7 POL100007 7 7 7 7 7 2024-08-10
8 POL100008 8 8 8 8 8 2024-08-15
9 POL100009 9 9 9 9 9 2024-09-01
Automatic context help is disabled.
Use the toolbar to manually get help
for the current caret position or to
toggle automatic help.
Table: policy_types
Columns:
policy_type_id int AI PK
type_name varchar(5
base_rate decimal(1
description text

INNER JOINS:


select * from premium_calculations;
SELECT *
FROM premium_calculations pc
INNER JOIN vehicles v ON pc.vehicle_id = v.vehicle_id;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

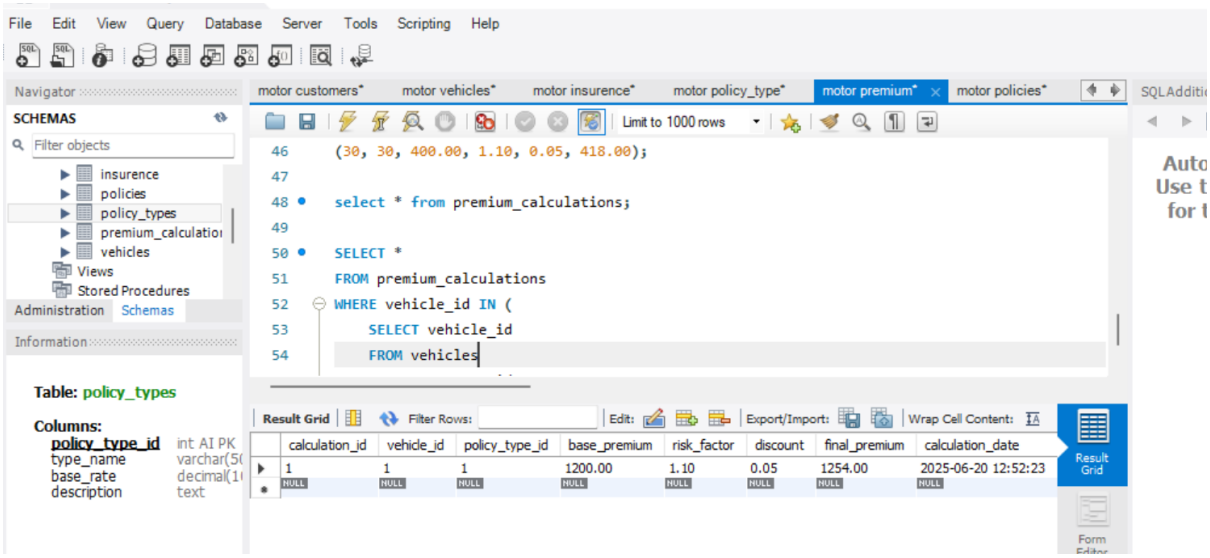

FROM
policies p
JOIN
customers c ON p.customer_id = c.customer_id
JOIN
insurence i ON p.insurence_id = i.insurence_id
LIMIT 0, 1000;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.


SELECT
p.policy_type_id,
c.customer_id
FROM
policies p
LEFT JOIN
customers c ON p.customer_id = c.customer_id;

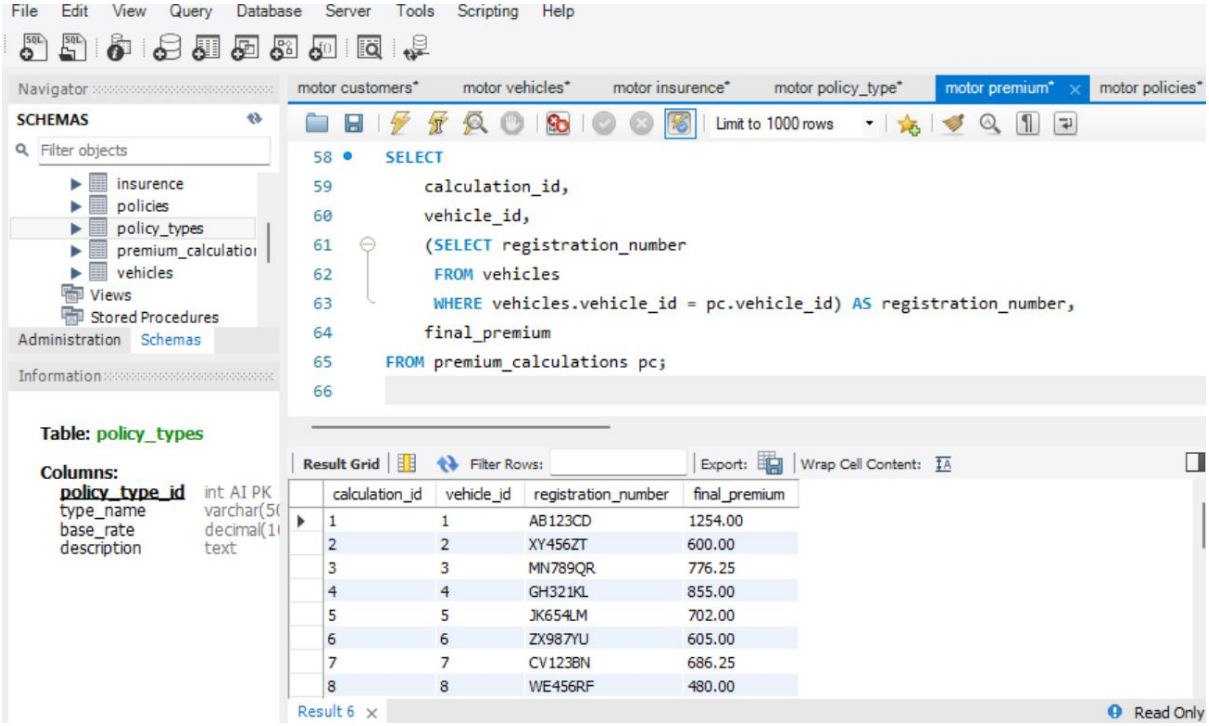


SUBQUERY:

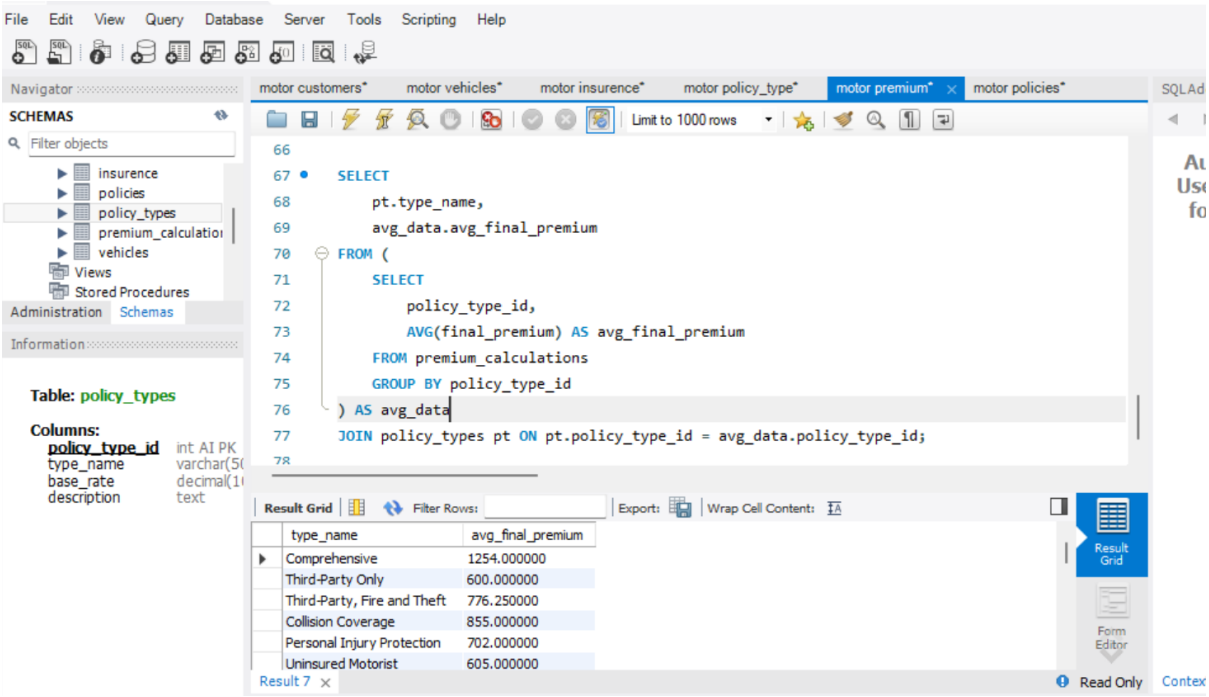
1.SUBQUERY IN WHERE CLAUSE:



2.SUBQUERY IN SELECT CLAUSE:

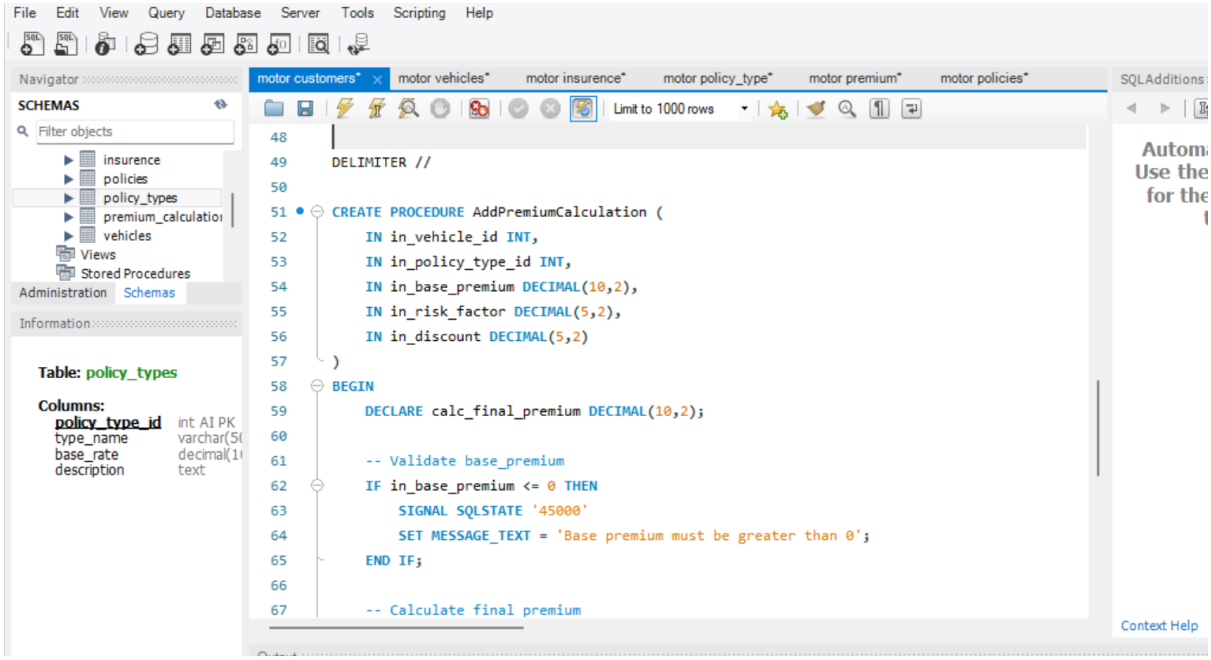


3.SUBQUERY IN FROM CLAUSE (INLINE VIEW)

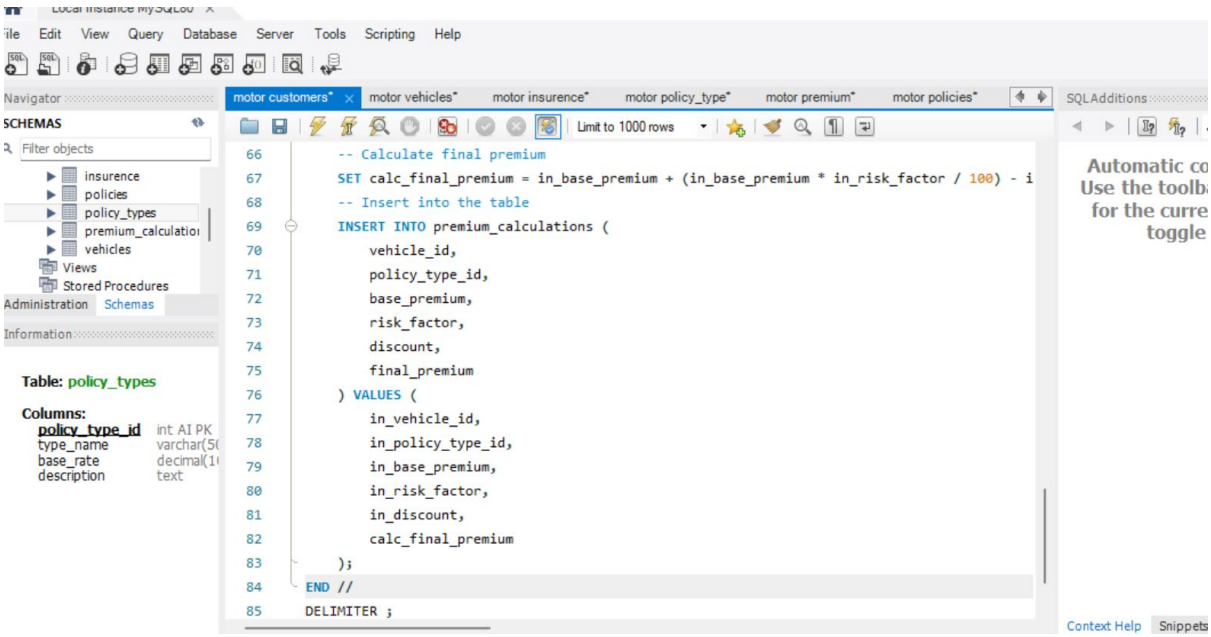


STORED PROCEDURE:

QUERY:

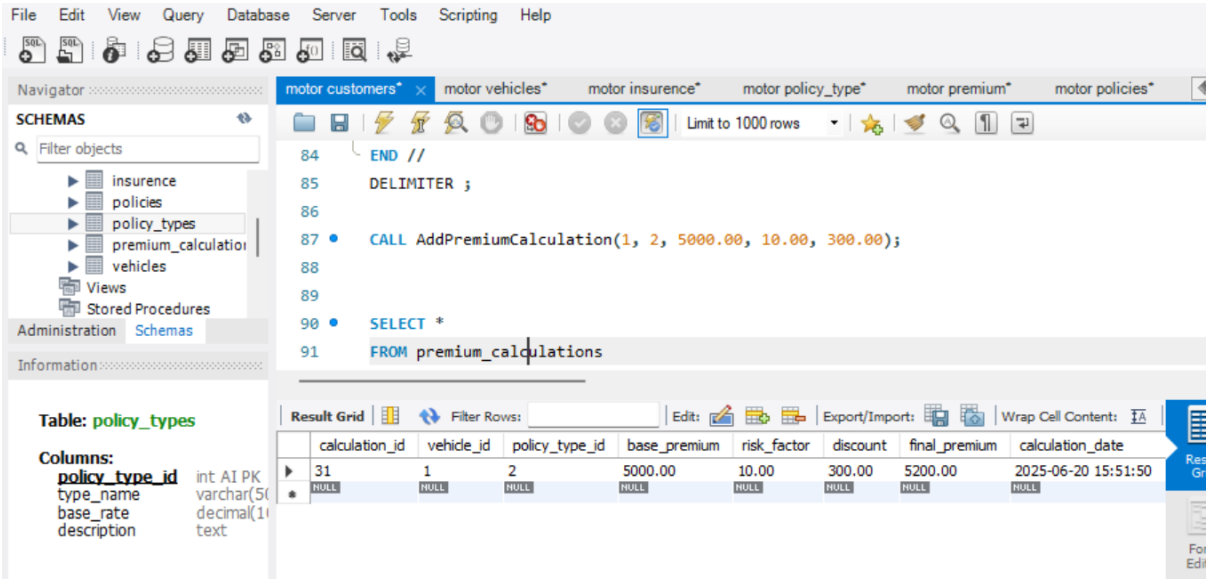


```
48
49  DELIMITER //
50
51  CREATE PROCEDURE AddPremiumCalculation (
52      IN in_vehicle_id INT,
53      IN in_policy_type_id INT,
54      IN in_base_premium DECIMAL(10,2),
55      IN in_risk_factor DECIMAL(5,2),
56      IN in_discount DECIMAL(5,2)
57  )
58  BEGIN
59      DECLARE calc_final_premium DECIMAL(10,2);
60
61      -- Validate base_premium
62      IF in_base_premium <= 0 THEN
63          SIGNAL SQLSTATE '45000'
64          SET MESSAGE_TEXT = 'Base premium must be greater than 0';
65      END IF;
66
67      -- Calculate final premium
```



```
66      -- Calculate final premium
67      SET calc_final_premium = in_base_premium + (in_base_premium * in_risk_factor / 100) - i
68      -- Insert into the table
69      INSERT INTO premium_calculations (
70          vehicle_id,
71          policy_type_id,
72          base_premium,
73          risk_factor,
74          discount,
75          final_premium
76      ) VALUES (
77          in_vehicle_id,
78          in_policy_type_id,
79          in_base_premium,
80          in_risk_factor,
81          in_discount,
82          calc_final_premium
83      );
84  END //
85  DELIMITER ;
```

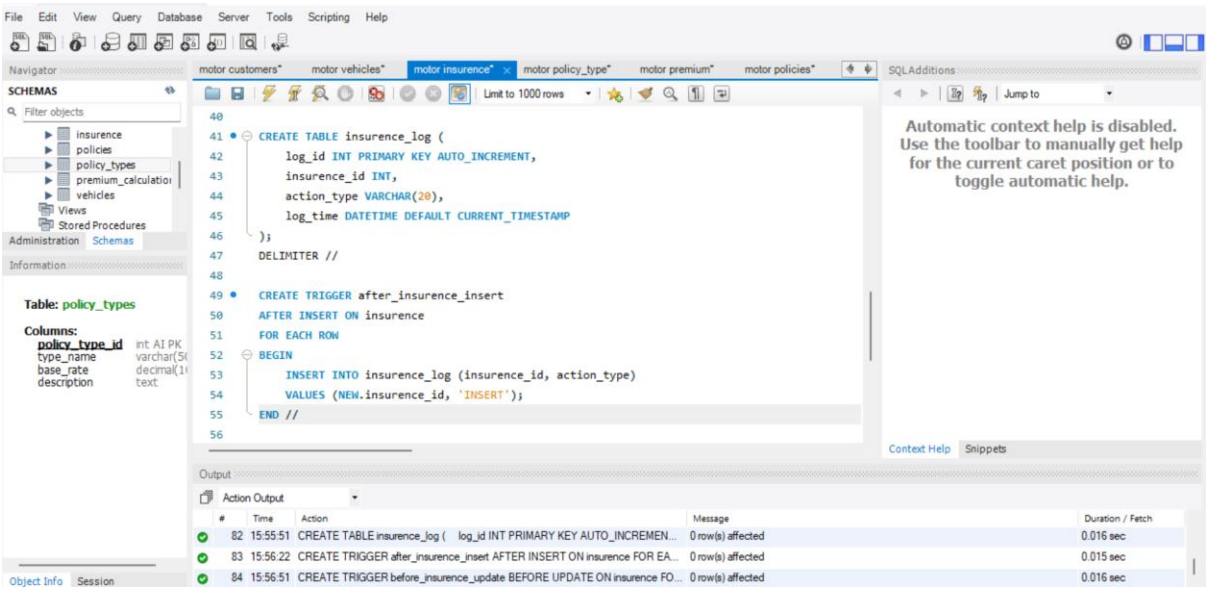
OUTPUT



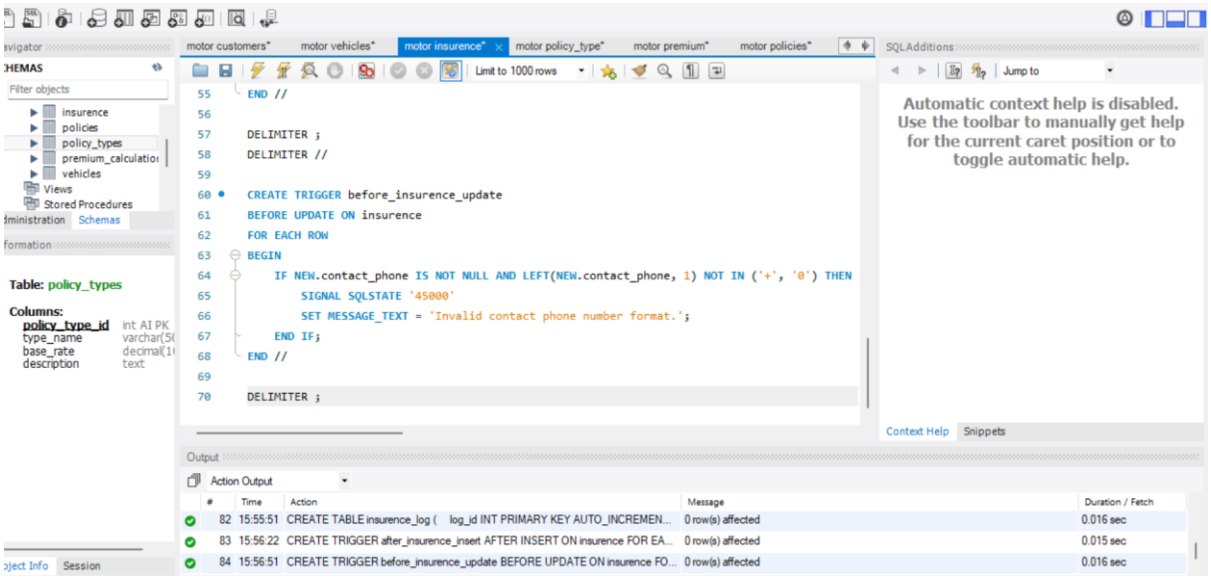
TRIGGERS

SQL TRIGGERS FOR THE INSURANCE TABLE:

AFTER INSERT TRIGGERS:



BEFORE UPDATE TRIGGER



AFTER DELETE TRIGGERS:

DELIMITER //

CREATE TRIGGER after_insurence_delete

AFTER DELETE ON insurence

FOR EACH ROW

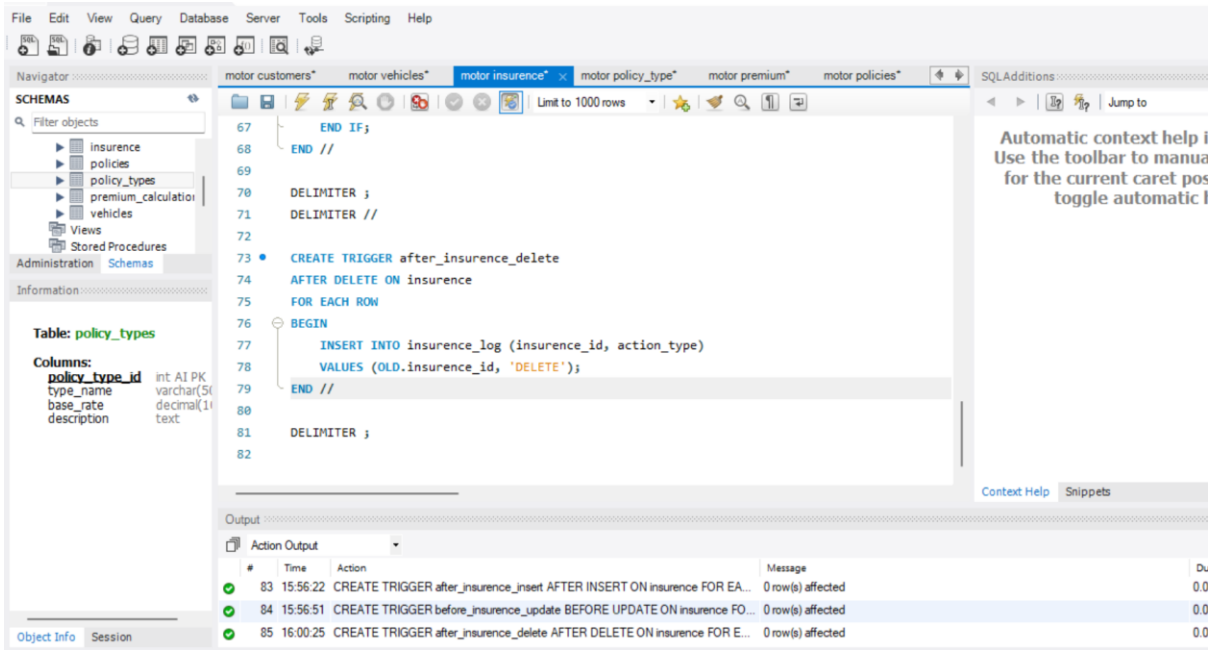
BEGIN

INSERT INTO insurence_log (insurence_id, action_type)

VALUES (OLD.insurence_id, 'DELETE');

END //

DELIMITER ;


File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
insurance
policies
policy_types
premium_calculation
vehicles
Views
Stored Procedures
Administration Schemas
Information
Table: policy_types
Columns:
policy_type_id int AI PK
type_name varchar(5
base_rate decimal(1
description text
Object Info Session
motor customers* motor vehicles* motor insurence* motor policy_type* motor premium* motor policies*
Limit to 1000 rows
67 END IF;
68 END //
69
70 DELIMITER ;
71 DELIMITER //
72
73 CREATE TRIGGER after_insurence_delete
74 AFTER DELETE ON insurence
75 FOR EACH ROW
76 BEGIN
77 INSERT INTO insurence_log (insurence_id, action_type)
78 VALUES (OLD.insurence_id, 'DELETE');
79 END //
80
81 DELIMITER ;
82
SQLAdditions
Jump to
Automatic context help i
Use the toolbar to manua
for the current caret pos
toggle automatic h
Context Help Snippets
Output
Action Output
Time Action Message Du
83 15:56:22 CREATE TRIGGER after_insurence_insert AFTER INSERT ON insurence FOR EA... 0 row(s) affected 0.0
84 15:56:51 CREATE TRIGGER before_insurence_update BEFORE UPDATE ON insurence FO... 0 row(s) affected 0.0
85 16:00:25 CREATE TRIGGER after_insurence_delete AFTER DELETE ON insurence FOR E... 0 row(s) affected 0.0