DBMS PROJECT OVERVIEW 2

1.UNIVERSAL TABLE:

(V\_id,V\_age,Price,Dealer\_id,D\_name,D\_Phn,D\_Email,Modify\_id,Colour,Fuel,Seats,Type\_id,Type\_name,Brand\_name,Model\_name,Year\_of\_release,Order\_id, Order\_Quatity,C\_id, C\_name,C\_phno,C\_Email,C\_Age)

Here we assumed that one customer can place many orders and one order may contain more than one vehicle but one vehicle can be repeated because two customers can buy same vehicle so the candidate key for the universal table is (V\_id,Order\_id )

2.

1NF

Here we can see that both D\_name and C\_name are both composite attributes so we divide name into fname and lname in both attributes now the universal table becomes

(V\_id,V\_age,Price,Dealer\_id,D\_fname,D\_lname,D\_Phn,D\_Email,Modify\_id,Colour,Fuel,Seats,Type\_id,Type\_name,Brand\_name,Model\_name,Year\_of\_release,Order\_id, Order\_Quatity,C\_id, C\_fname,C\_lname,C\_phno,C\_Email,C\_Age)

The functional dependencies we can identify from the universal table are

V\_id-> V\_age,Price ,Dealer\_id,Type\_id,Modify\_id

Dealer\_id-> D\_fname,D\_lname,D\_Phn,D\_Email

Type\_id ->Type\_name,Brand\_name,Model\_name, Year\_of\_release

Modify\_id -> Colour,Fuel,Seats

C\_id -> C\_fname, C\_lname, C\_phno, C\_Email, C\_Age

Order\_id -> C\_id

Order\_id,Vehicle\_id -> Order\_Quatity

2NF

Relation R become 2NF if R is in 1NF and R does not have any partial dependency

Which means a proper subset of candidate key should not determine non key attributes

But we can see V\_id is a proper subset of candidate key

Where candidate key is V\_id,Order\_id

V\_id determines many non key attributes so to make R into 2NF we will divide table into

**Vehicle**(V\_id,V\_age,Price,Dealer\_id,D\_fname,D\_lname,D\_Phn,D\_Email,Modify\_id,Colour,Fuel,Seats,Type\_id,Type\_name,Brand\_name,Model\_name,Year\_of\_release)

**Order**(Order\_id,Order\_Quantity,V\_id,C\_id,C\_fname,C\_lname, C\_phno, C\_Email, C\_Age)

Now we can observe in relation Order Order\_id can determine C\_id,C\_fname,C\_lname, C\_phno, C\_Email, C\_Age

So, by dividing the table

**Ordered\_Customer**(Order\_id, C\_id,C\_fname,C\_lname, C\_phno, C\_Email, C\_Age)

**Ordered\_Quantity**(Order\_id,Order\_Quantity,V\_id)

Now we got 3 Relations after doing the 2NF the relations are:

**Vehicle**(V\_id,V\_age,Price,Dealer\_id,D\_fname,D\_lname,D\_Phn,D\_Email,Modify\_id,Colour,Fuel,Seats,Type\_id,Type\_name,Brand\_name,Model\_name,Year\_of\_release)

**Ordered\_Customer**(Order\_id, C\_id,C\_fname,C\_lname, C\_phno, C\_Email, C\_Age)

**Ordered\_Quantity**(Order\_id,Order\_Quantity,V\_id)

We can observe in the three relations that no partial dependency in any of the table and all table are preserving every FD.

3NF

For a relation to be in 3NF first the table should be in 2NF and there should be no transitive dependency which means no non key attribute should not determine a non key attribute

In the first Vehicle relation Dealer\_id, Modify\_id, Type\_id are the non key attributes in the relation Vehicle which can determine non key attributes

So, by splitting the table into 4 relations

**Vehicle**(V\_id,V\_age,Price,Dealer\_id,Modify\_id, Type\_id)

**Dealer** (Dealer\_id,D\_fname,D\_lname,D\_Phn,D\_Email)

**Modify**(Modify\_id,Colour,Fuel,Seats)

**Type**(Type\_id,Type\_name,Brand\_name,Model\_name,Year\_of\_release)

We can see that in the above relations the 4 tables are in 3NF

Now lets look into the other two tables

Ordered\_Customers and Ordered\_Quantity

**Ordered\_Customer**(Order\_id, C\_id,C\_fname,C\_lname, C\_phno, C\_Email, C\_Age)

Here in this relation C\_id is an non key attribute and determines non key attributes so we need to make it into 3NF

**Ordered\_Customer**(Order\_id, C\_id)

**Customer**(C\_id,C\_fname,C\_lname, C\_phno, C\_Email, C\_Age)

And the both relations are in 3NF

Now lets normalize the last relation

**Ordered\_Quantity**(Order\_id,Order\_Quantity,V\_id)

Here no non-key attribute determine no non-key attribute so the relation is in 3NF

So the relations we got after normalizing universal table are

**Vehicle**(V\_id,V\_age,Price,Dealer\_id,Modify\_id, Type\_id)

**Dealer** (Dealer\_id,D\_fname,D\_lname,D\_Phn,D\_Email)

**Modify**(Modify\_id,Colour,Fuel,Seats)

**Type**(Type\_id,Type\_name,Brand\_name,Model\_name,Year\_of\_release)

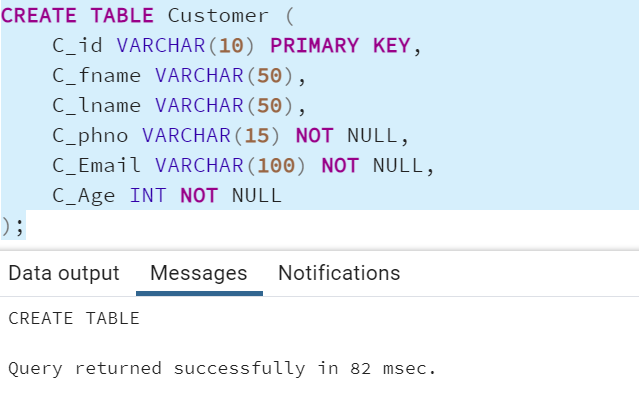
**Ordered\_Customer**(Order\_id, C\_id)

**Customer**(C\_id,C\_fname,C\_lname, C\_phno, C\_Email, C\_Age)

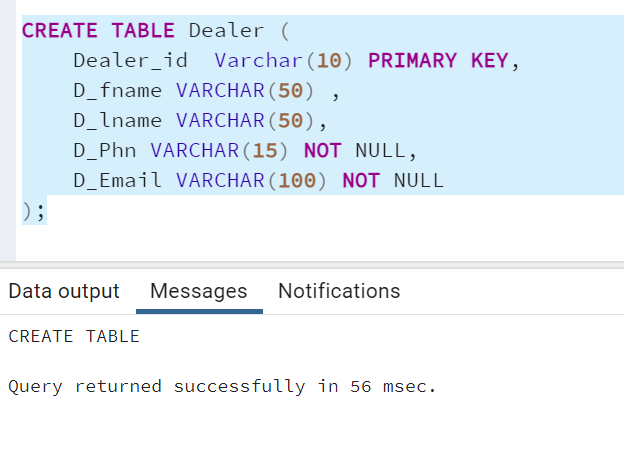
**Ordered\_Quantity**(Order\_id,Order\_Quantity,V\_id)

3.DDL Queries

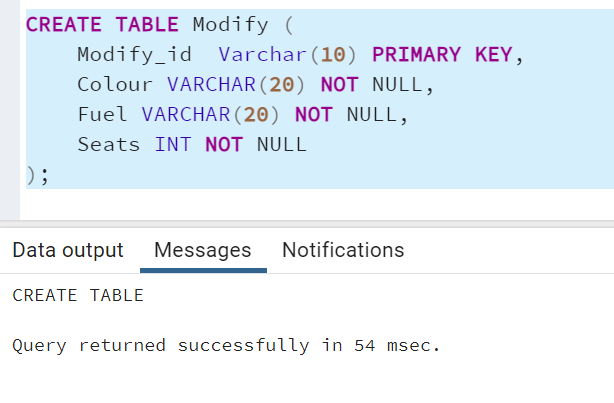
Customer table



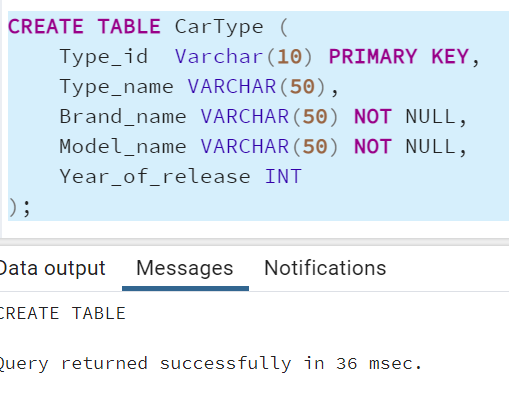
Dealer table



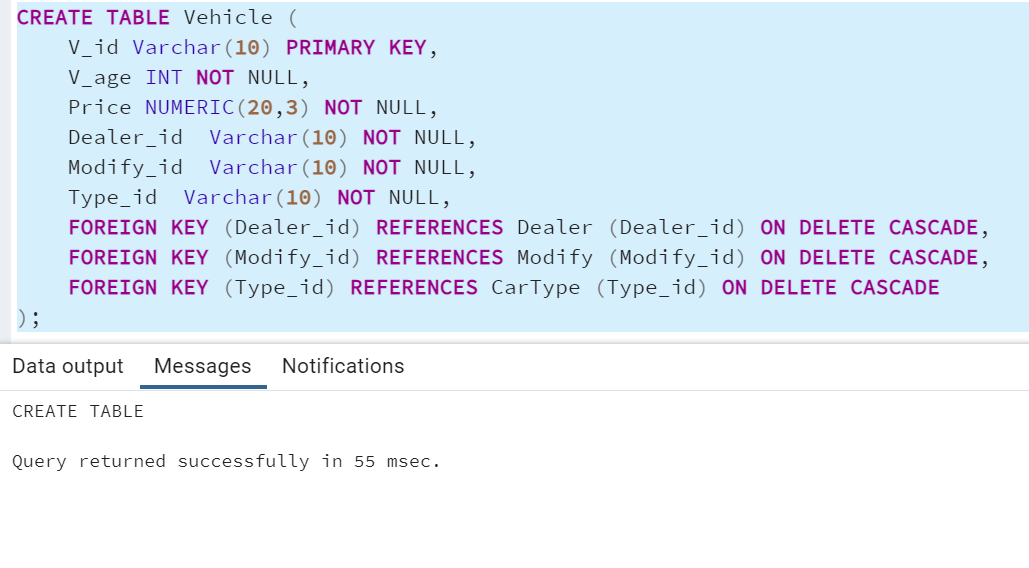
Modify table creation



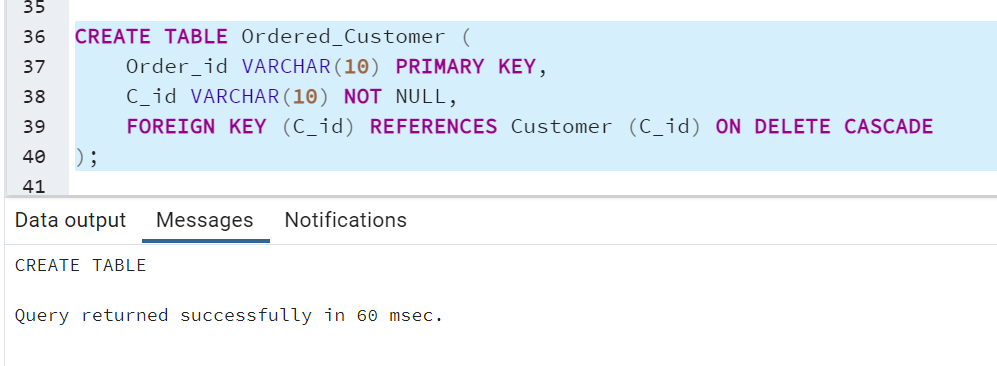
CarType table



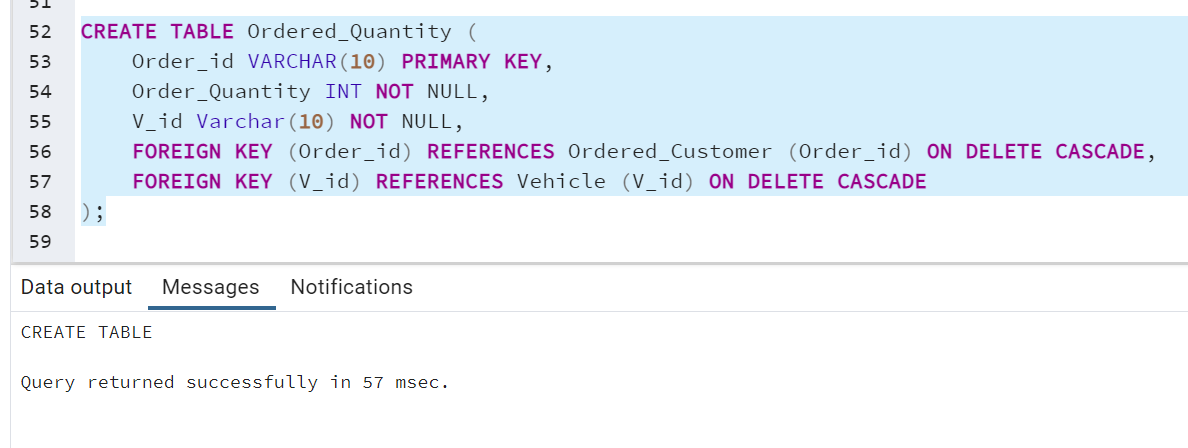
Vehicle table creation



Ordered\_Customer table



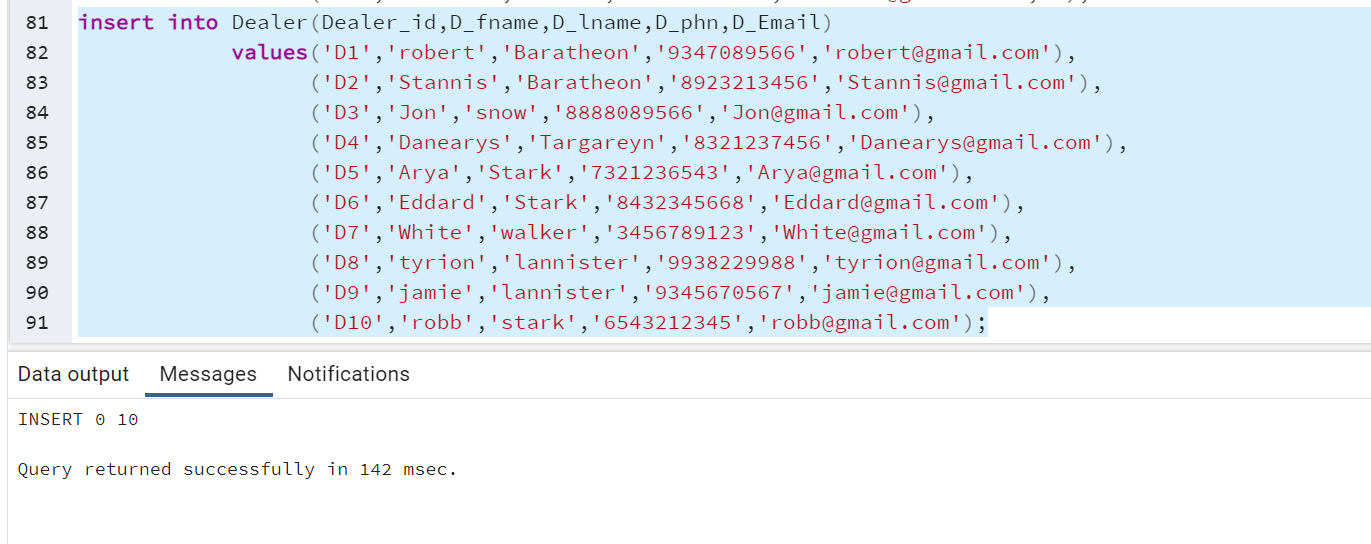
Ordered\_Quantity table



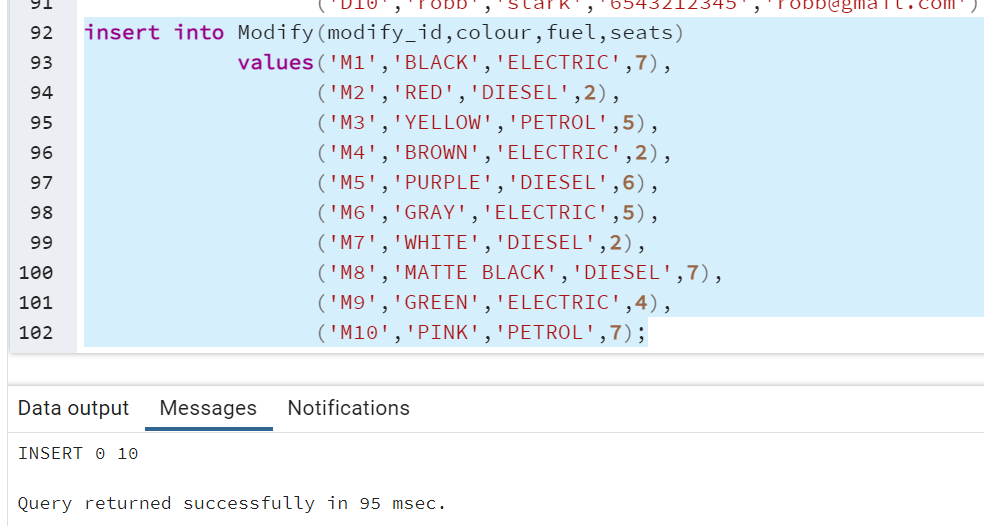
Customer table insertion



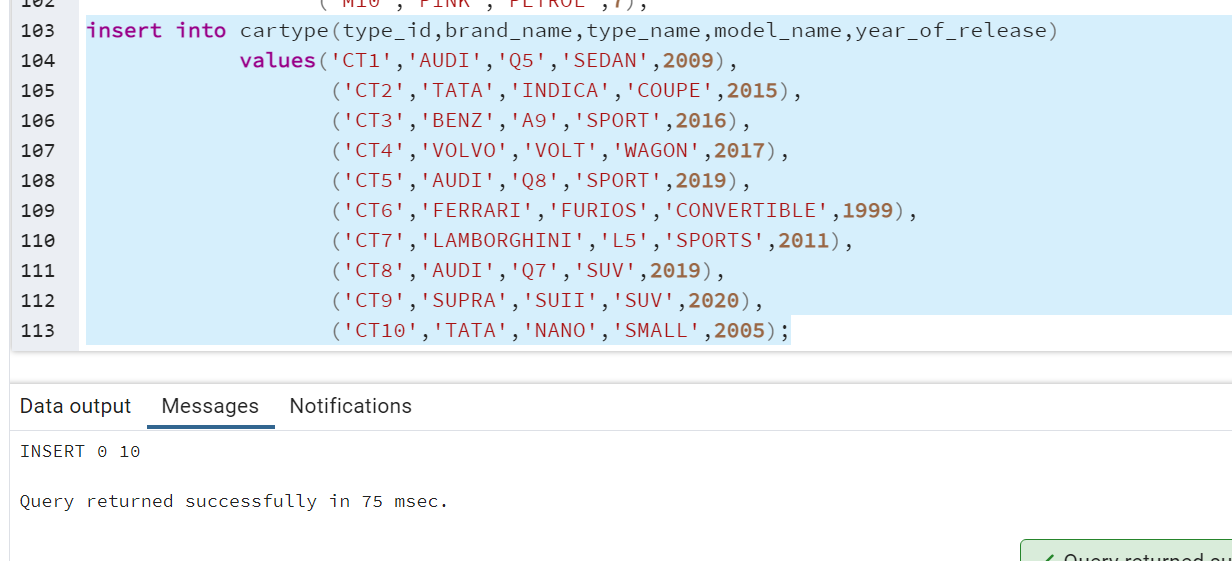
Dealer table insertion



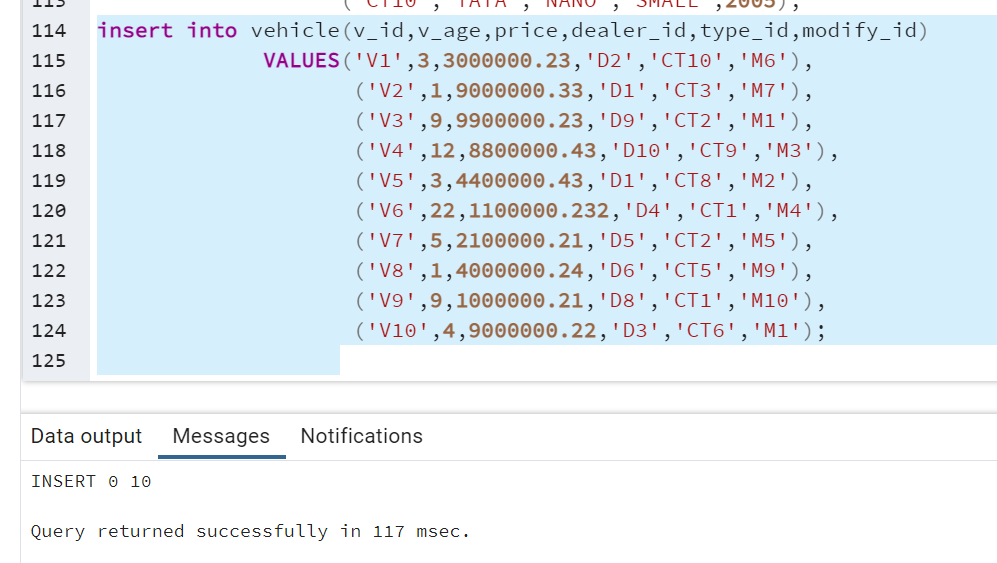
Modify table insertion



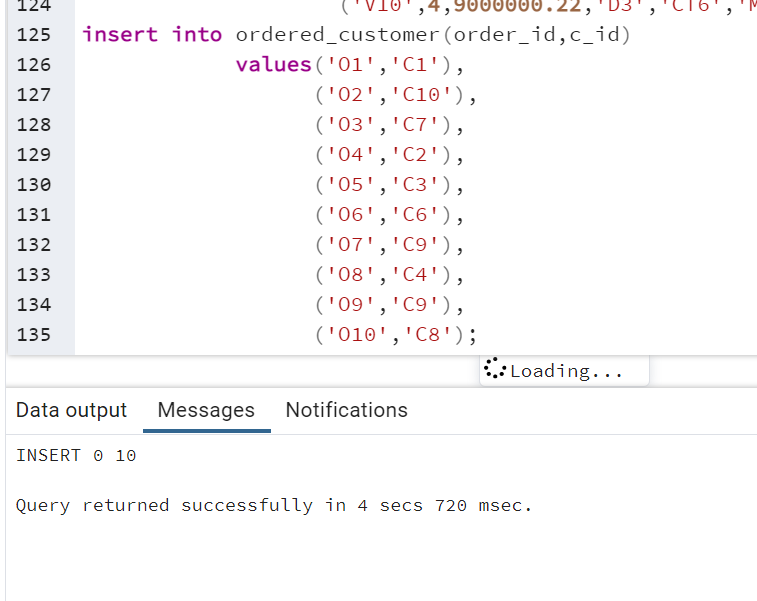
Car\_Type table insertion



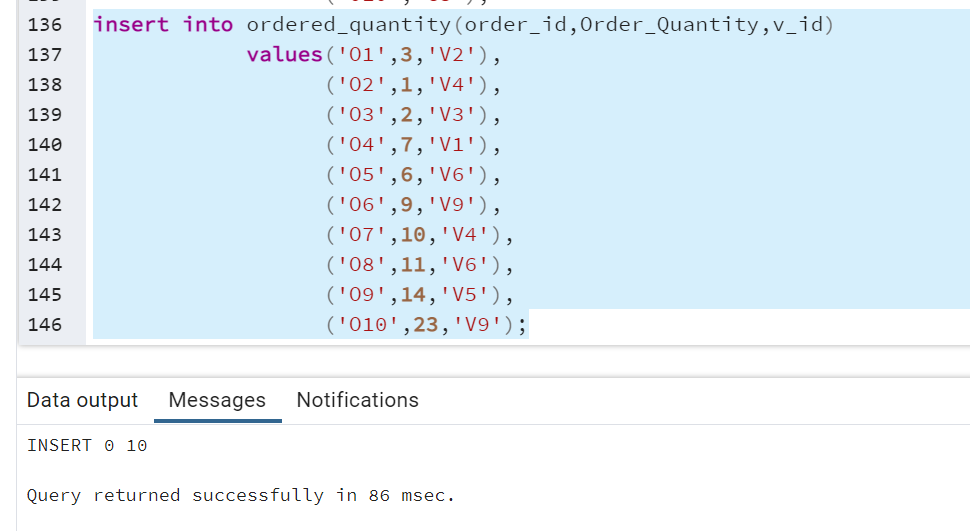
Vehicle insertion



Customer\_order insertion



Ordered\_quantity

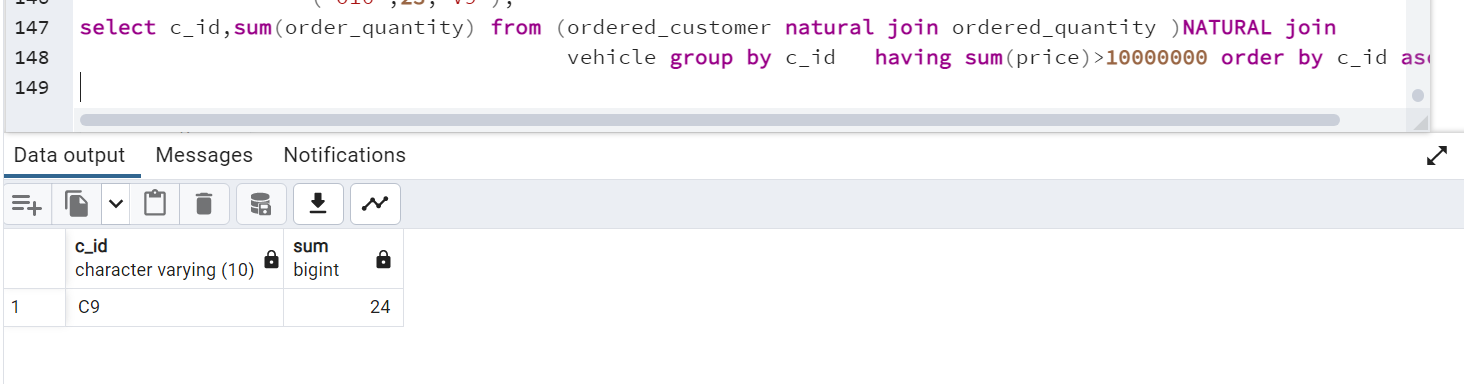


4.

1.

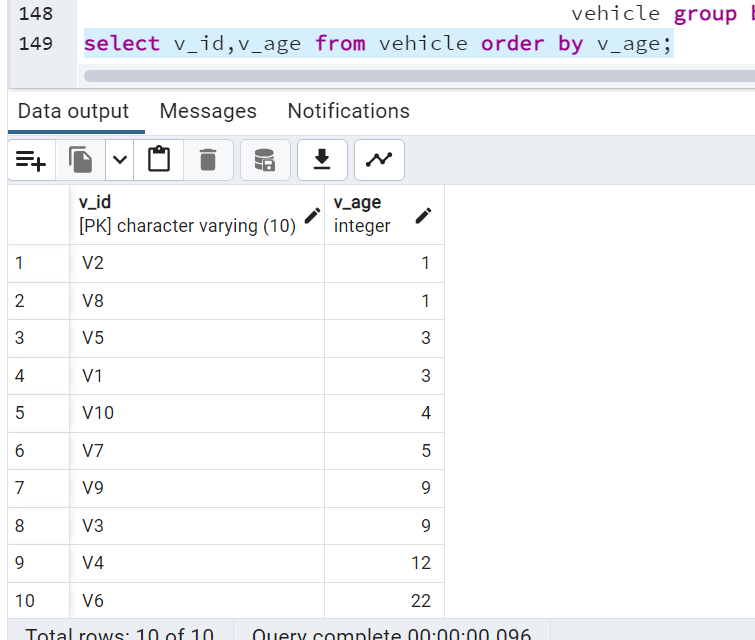
Query 1

Find the number of cars each customer ordered where sum of the prices of the individual cars is greater than 10000000

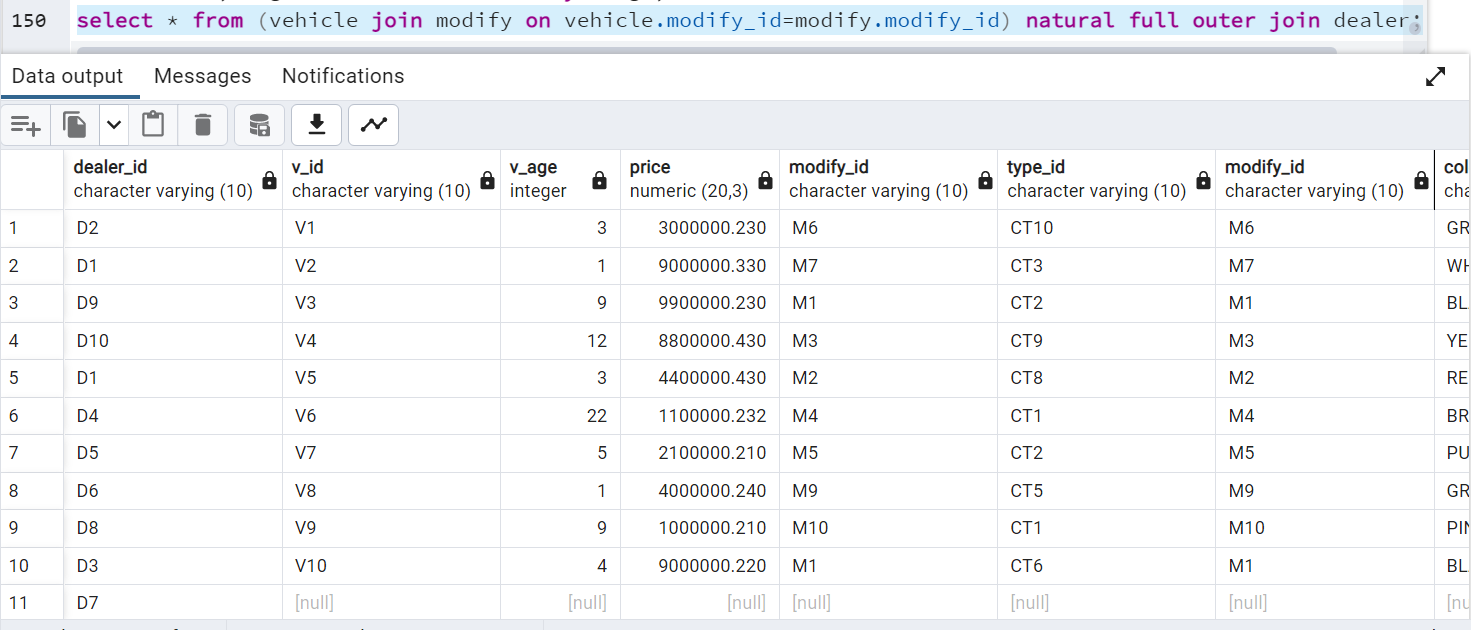


2.

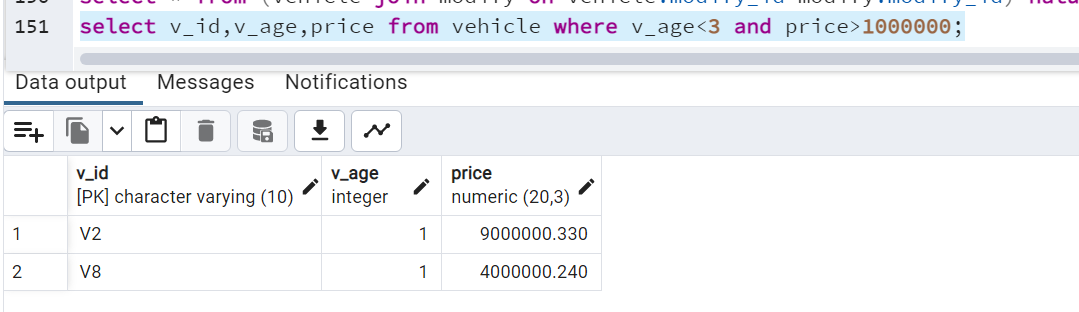
Display the vehicle id , age order by age



3.Mention all the dealer details irrespective of the vehicle they may have or not

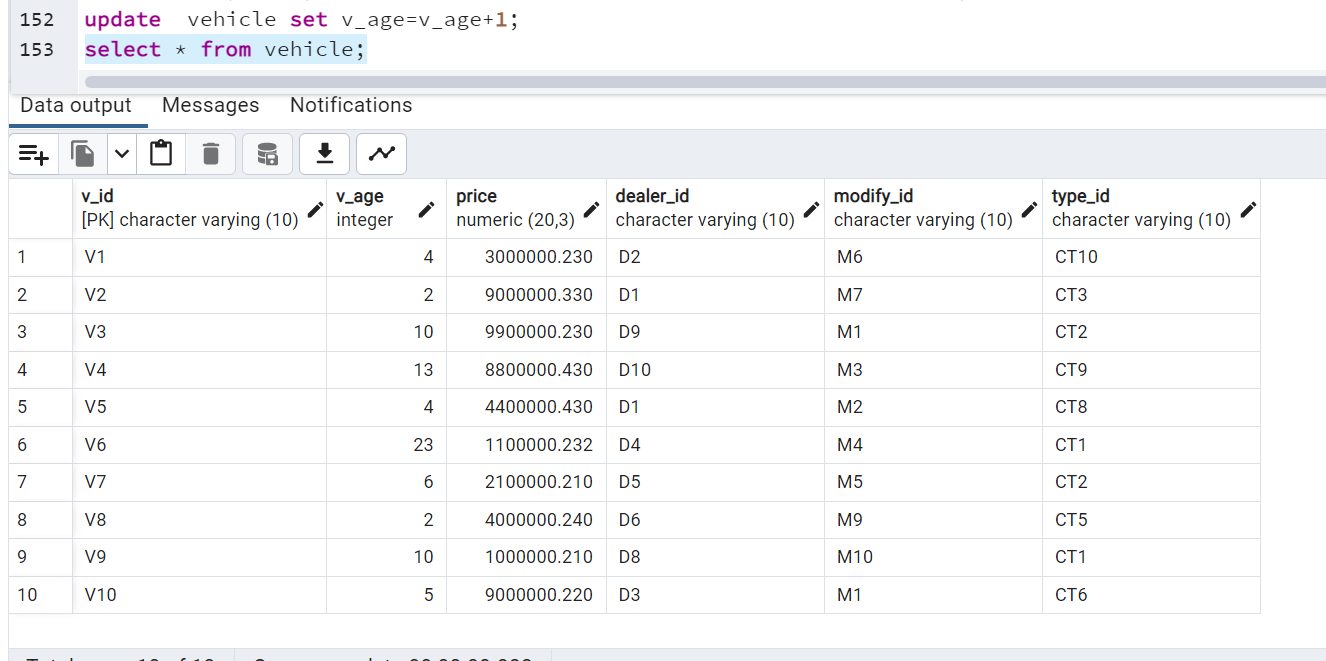


4.Display all the vehicles whose age is less than 3 and price is greater than 1000000.



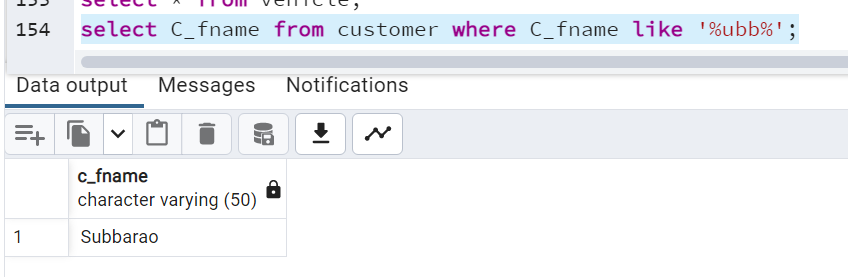
5.

Increase each vehicleage by 1 year



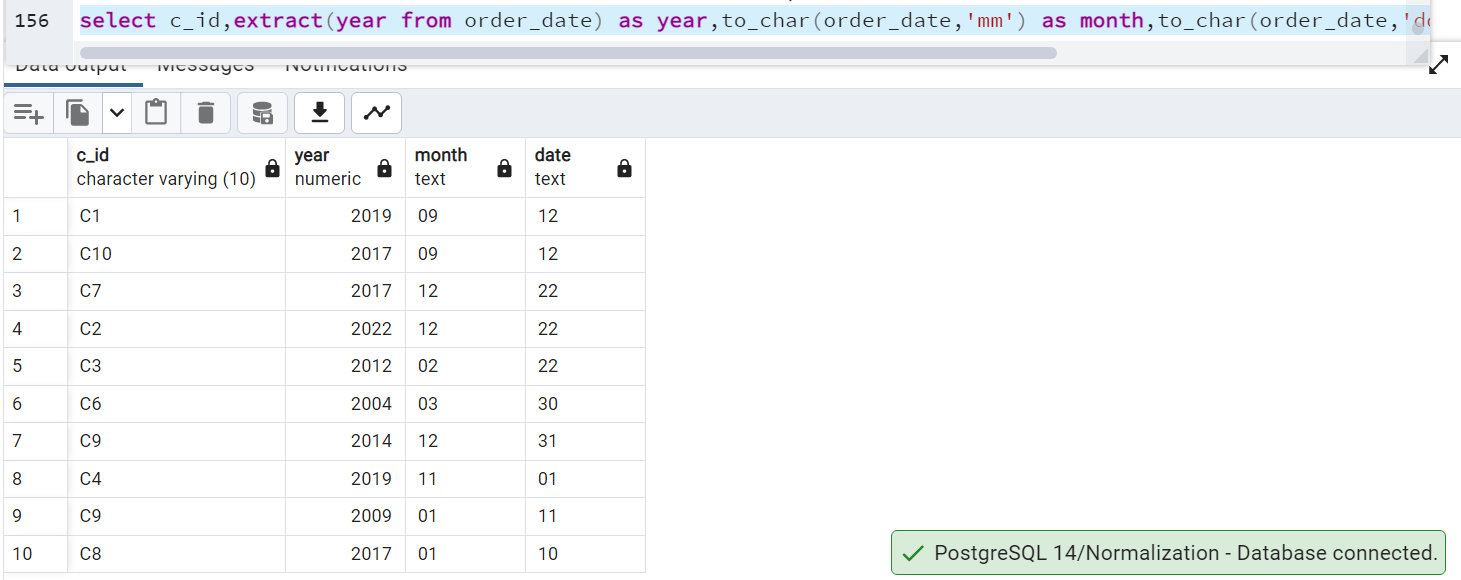
6.

Print the name of the customer whose fname contains the substring as ubb



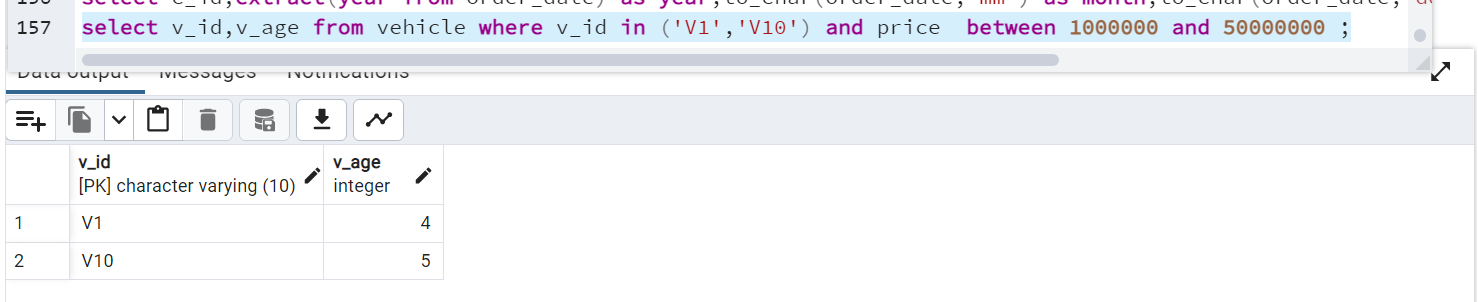
7.

Display the customer id and the date order placed as separate columns for year,month,date using to\_char and extract

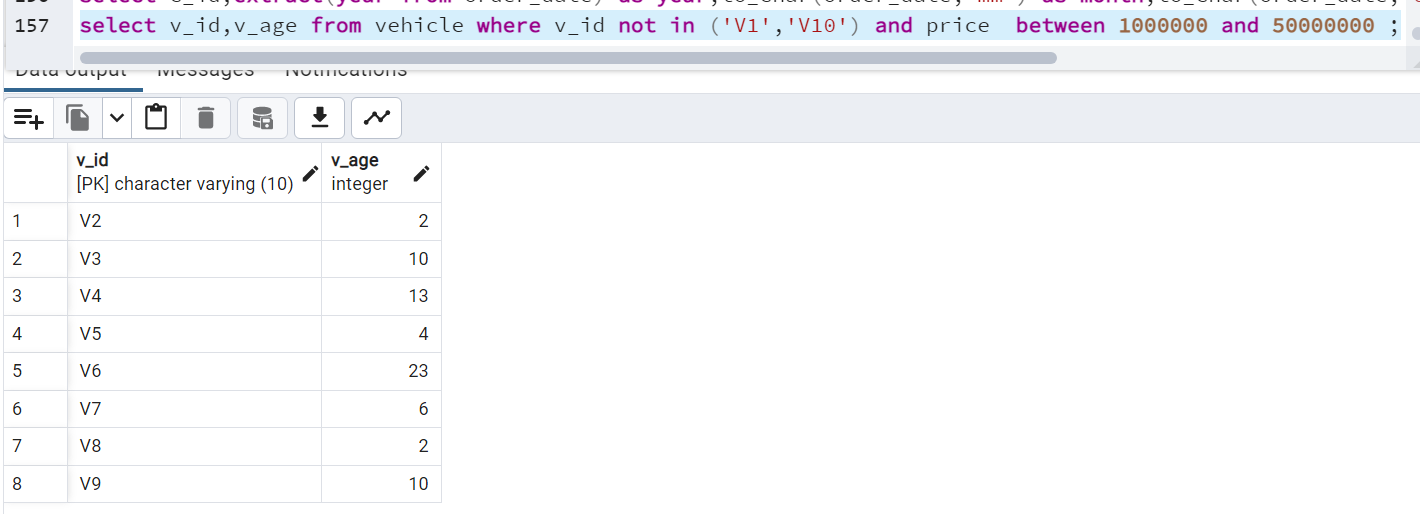


8.

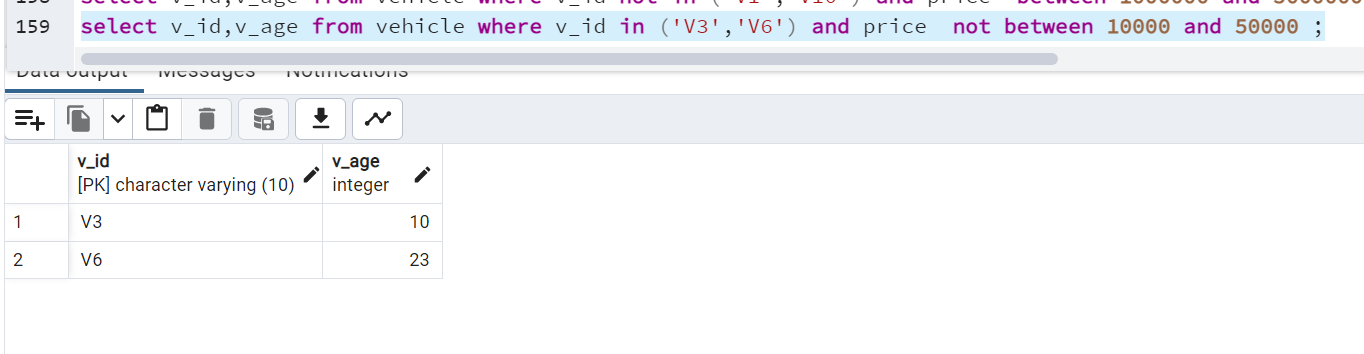
In and Between



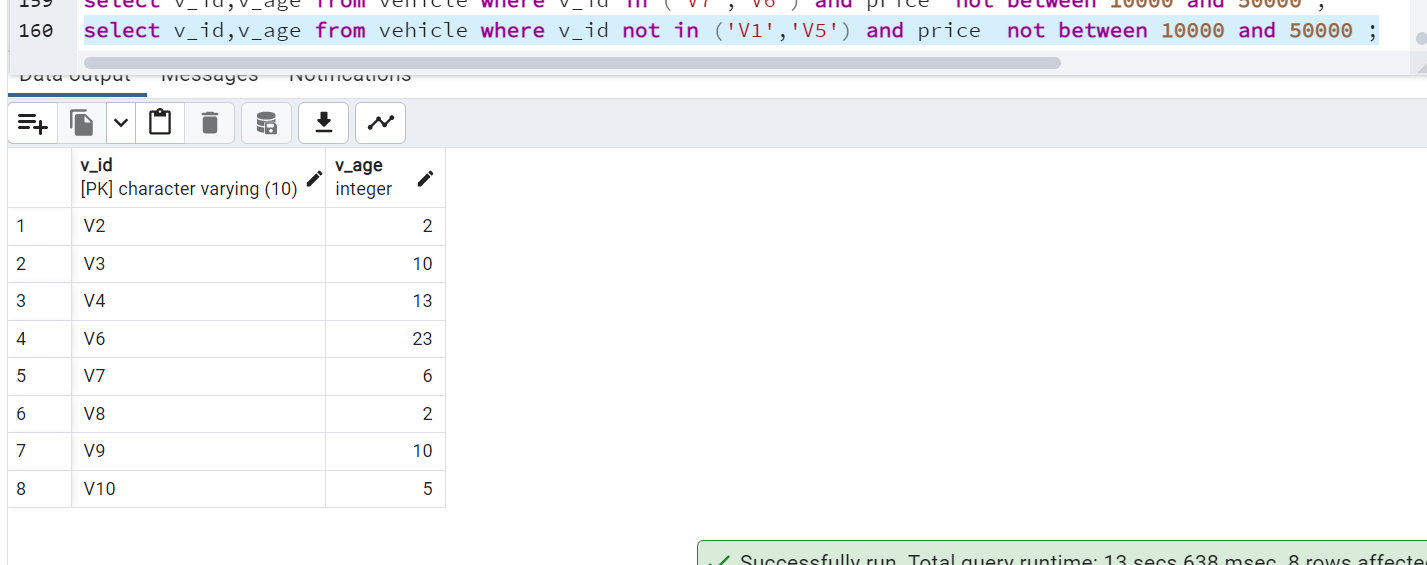
Not in and between



In and not between

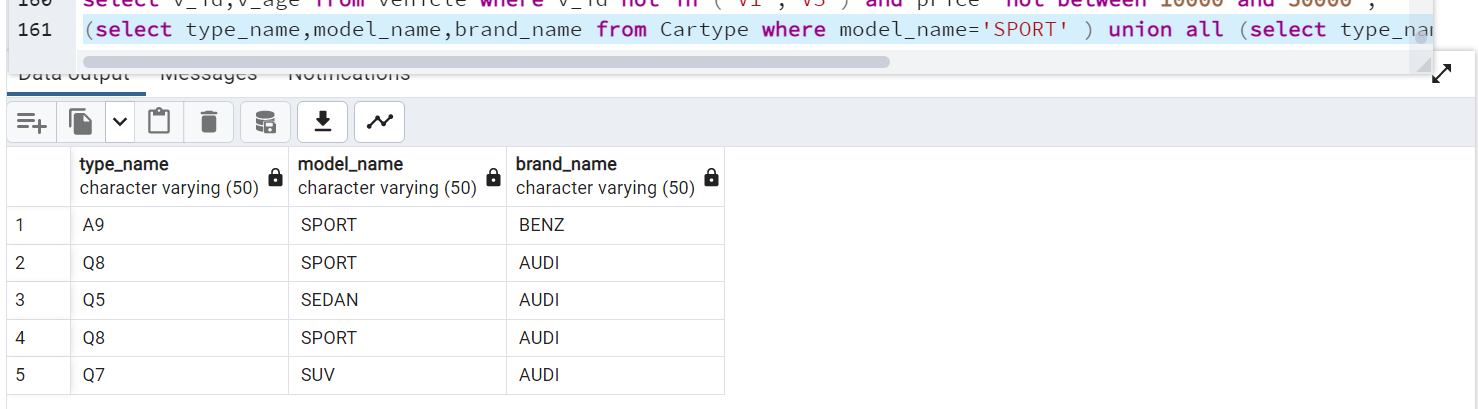


Not in and not between



9.

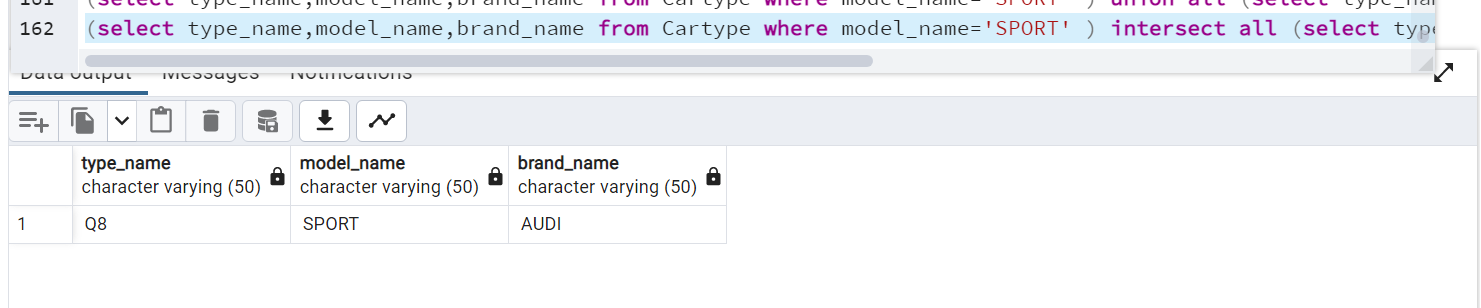
Cars having either brand name is audi or model is sports



union

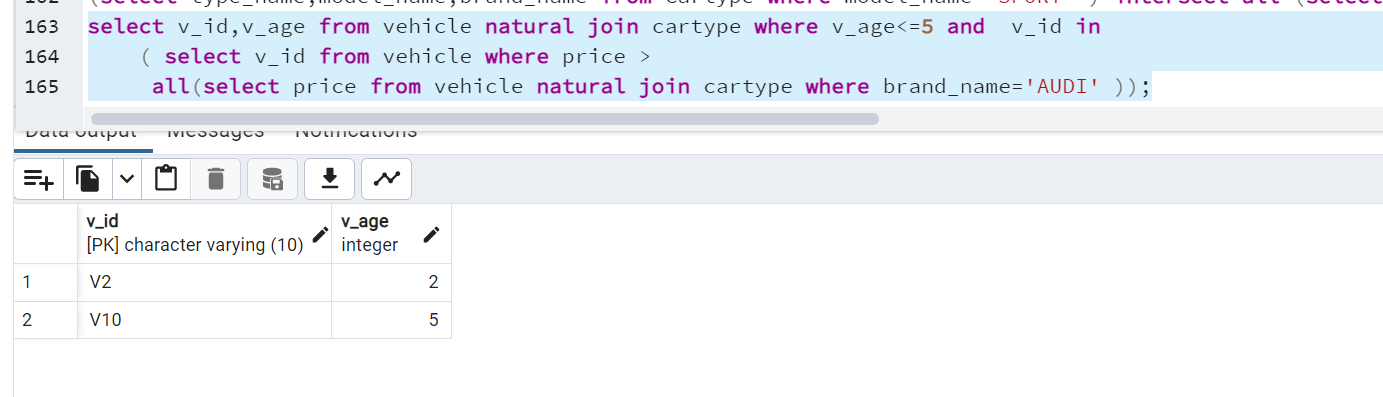
Cars having either brand name is audi and model is sports

intersect



10.

All



Any

