1st Database Hand in : Pets and Petshops Database

Mie Grønkjær Jørgensen

Cph-mj761

Hand-in Description:

- Make a table Pet with id, type (e.g. bird), breed (e.g. dove) and price.
- Make a table Petshop with id, name and address.
- Make the database so a certain kind of pet can be in many petshops and a petshop can have many different pets.
- A petshop should be able to have many of a certain kind of pet. Set up the necessary relations.
- Populate the tables with typical data.

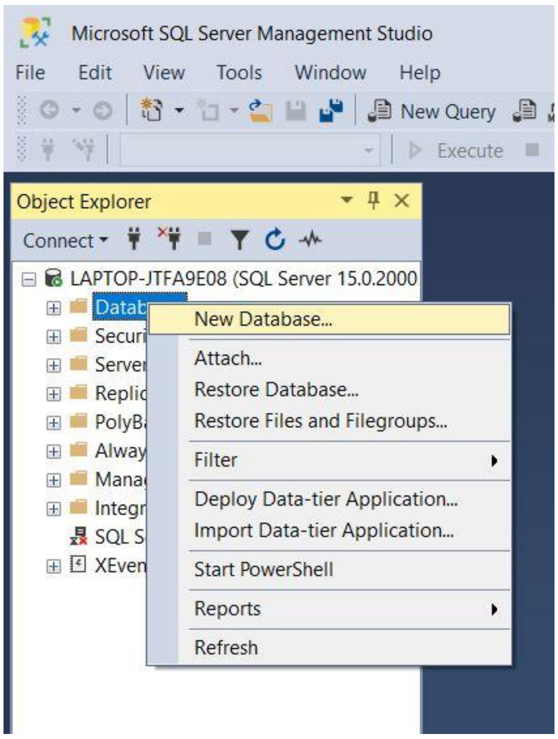
- Make a SQL query to show the pets of a given type.
- Make a SQL query to show where a given kind of pet can be found.

And try to :

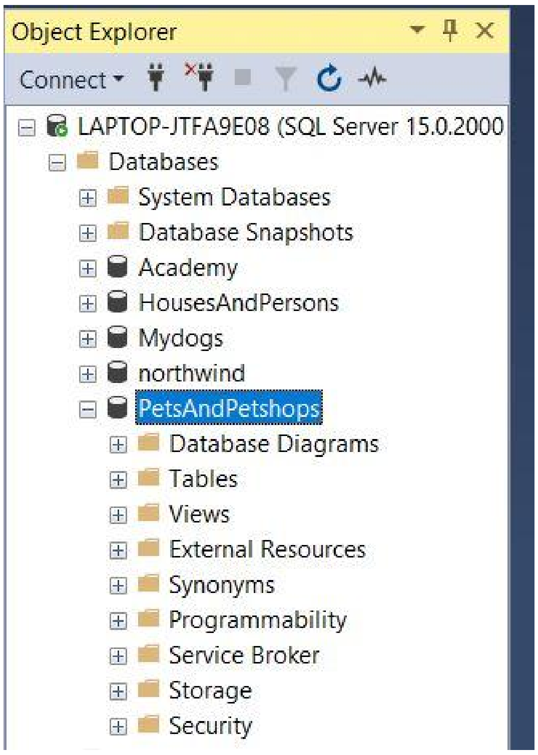
- Make a SQL query to show the number of different pets at a given petshop.
- Make a SQL query to show the number of pets at a given petshop.

Databases

I started off creating a new Database called PetsAndPetshops. To create my tables in.

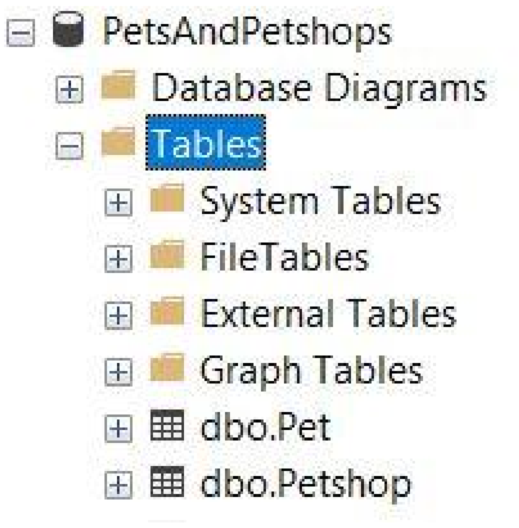


Microsoft SQL Server Management Studio
File Edit View Tools Window Help
New Query
Execute
Object Explorer
Connect
LAPTOP-JTFA9E08 (SQL Server 15.0.2000
Datab
Securi
Server
Replic
PolyB
Alway
Mana
Integr
SQL S
XEven
New Database...
Attach...
Restore Database...
Restore Files and Filegroups...
Filter
Deploy Data-tier Application...
Import Data-tier Application...
Start PowerShell
Reports
Refresh



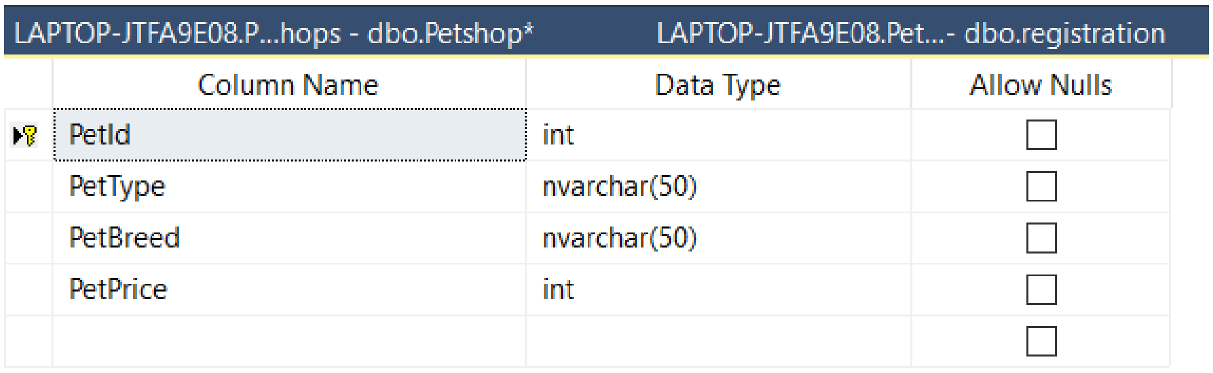
Tabels

I then created two new tabels, one called Pets and another called Petshop.



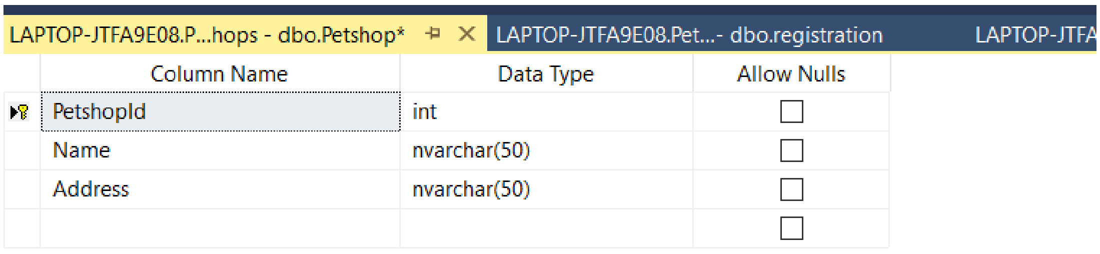
- PetsAndPetshops
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - Graph Tables
    - dbo.Pet
    - dbo.Petshop

Inside the table Pet I then put in the data which the assignment is required. I put in the ID and called it PetId, put the datatype as int, and didn't check 'allow null'.Then I made it a Primary key by pressing the little black key. I put in 3 other columns containing PetType, PetBreed and PetPrice, I gave Type and Breed the datatype nvarchar and PetPrice the int datatype since it's numbers only being used in that section. I also didn't check the 'allow null' either.
I then proceeded to do the same on Table Pets, but called the ID PetshopId, again with int as datatype. And put in Name and Address as the other columns, also with nvarchar and not checked on 'allow null'.



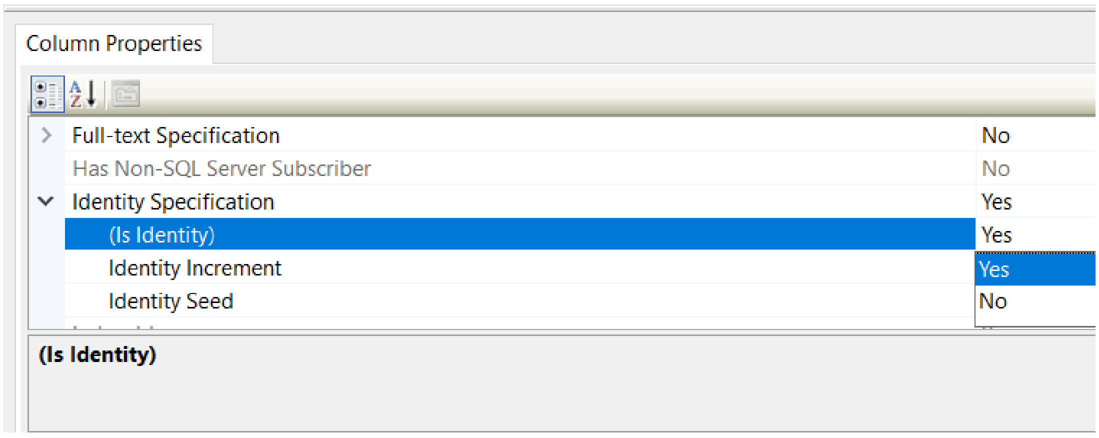
LAPTOP-JTFA9E08.P...hops - dbo.Petshop* LAPTOP-JTFA9E08.Pet...- dbo.registration

Column Name	Data Type	Allow Nulls
PetId	int	☐
PetType	nvarchar(50)	☐
PetBreed	nvarchar(50)	☐
PetPrice	int	☐
		☐



LAPTOP-JTFA9E08.P...hops - dbo.Petshop* | LAPTOP-JTFA9E08.Pet...- dbo.registration | LAPTOP-JTFA

Column Name	Data Type	Allow Nulls
PetshopId	int	☐
Name	nvarchar(50)	☐
Address	nvarchar(50)	☐
		☐

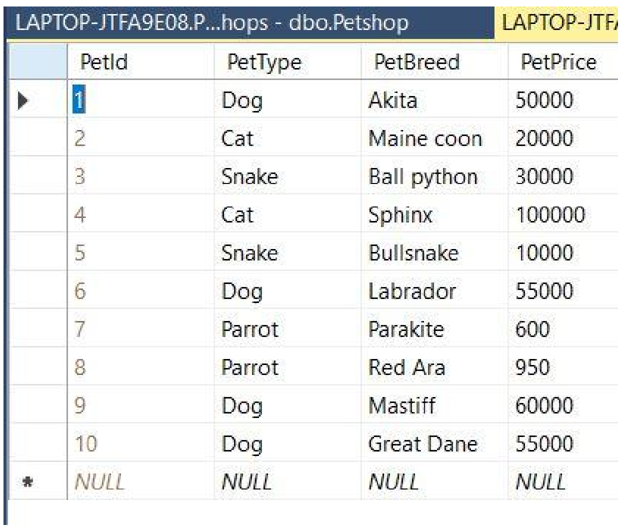


Column Properties

Full-text Specification	No
Has Non-SQL Server Subscriber	No
Identity Specification	Yes
(Is Identity)	Yes
Identity Increment	Yes
Identity Seed	No

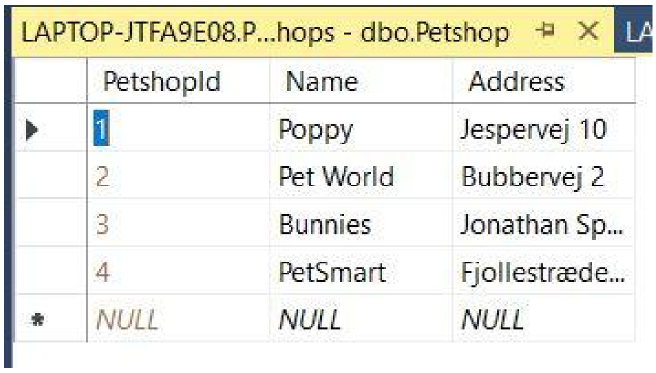
(Is Identity)

I chose in both the Pet and Petshop table to change the Identity specification from Is Identity ‘no’ to ‘yes’ so it will automatically put in the ID number when changing.
Inside both the Pet and Petshop table I then went into editing first Pet, by putting in data for the columns. Where I then proceeded to fill in random animal types/categories, their breeds and a price for them. In Petshop I then filled in names of Petshops and addresses for them. Since I changed the Identity specification to yes it will automatically count the next ID number.



LAPTOP-JTFA9E08.P...hops - dbo.Petshop | LAPTOP-JTF

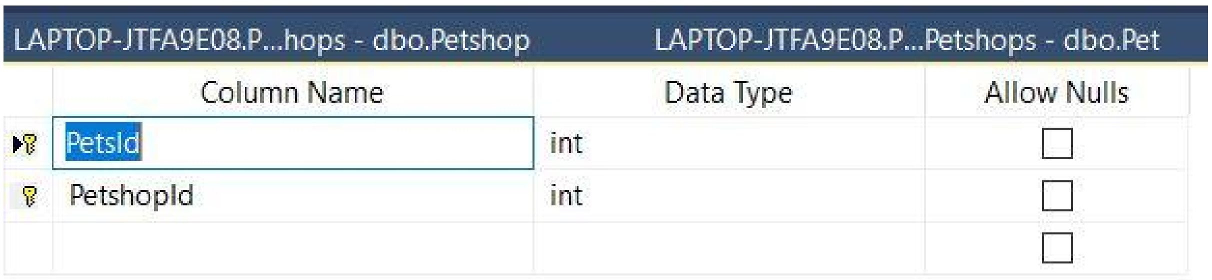
PetId	PetType	PetBreed	PetPrice
1	Dog	Akita	50000
2	Cat	Maine coon	20000
3	Snake	Ball python	30000
4	Cat	Sphinx	100000
5	Snake	Bullsnake	10000
6	Dog	Labrador	55000
7	Parrot	Parakite	600
8	Parrot	Red Ara	950
9	Dog	Mastiff	60000
10	Dog	Great Dane	55000
NULL	NULL	NULL	NULL



LAPTOP-JTFA9E08.P...hops - dbo.Petshop

PetshopId	Name	Address
1	Poppy	Jespervej 10
2	Pet World	Bubbervej 2
3	Bunnies	Jonathan Sp...
4	PetSmart	Fjollestræde...
NULL	NULL	NULL

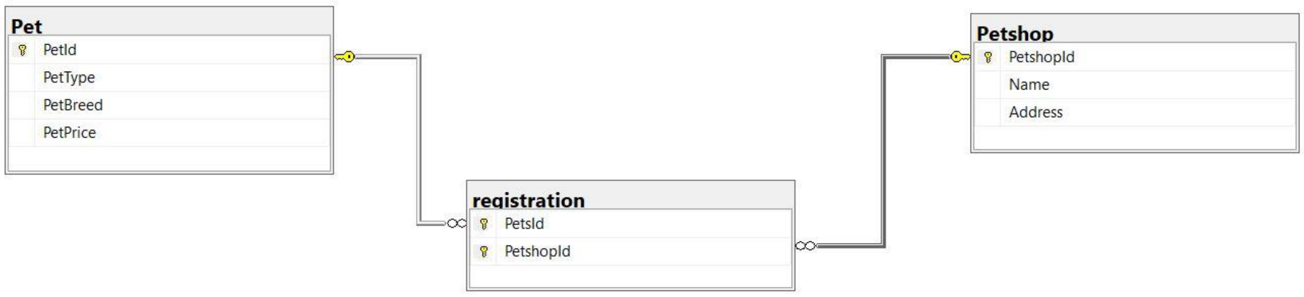
In order to make a Many to many relation I then chose to create a third table, a join table, called Registration. In that I put in two ID’s matching both Petshop and Pet’s ID’s and made them both primary keys, with int datatypes.



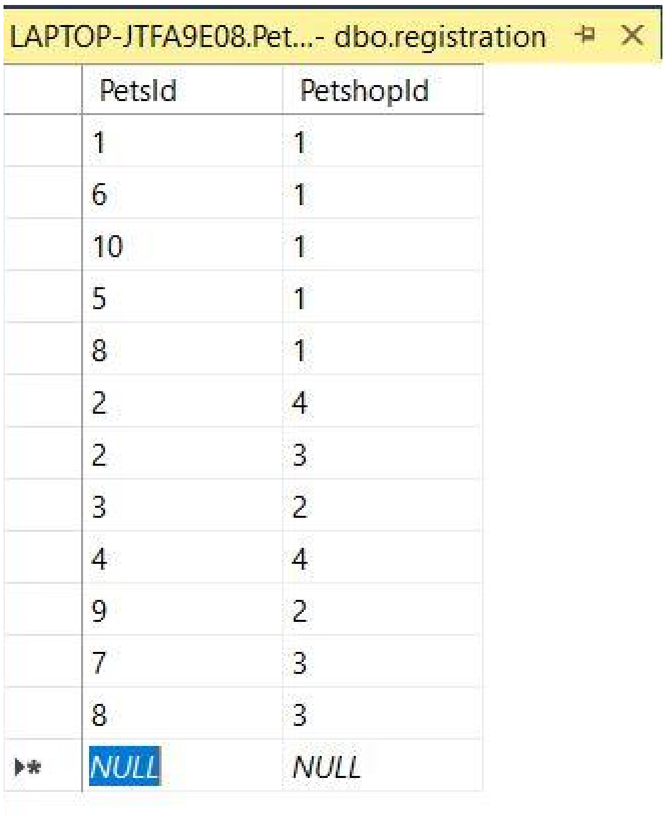
LAPTOP-JTFA9E08.P...hops - dbo.Petshop | LAPTOP-JTFA9E08.P...Petshops - dbo.Pet

Column Name	Data Type	Allow Nulls
PetsId	int	☐
PetshopId	int	☐
		☐

That way I could create a database diagram where I could put in the two other tables to make the multiple ID’s connection.
I would then drag PetId to the Registration table window where the ID matching PetId, then I made sure all the information were correct and that it said FK (foreign key), before pressing ok/accept.
That then creates the connection between those two tables. I then did the exact same thing with PetshopId and the ID matching that in the Registration table. This way all 3 tables are connected.



Then I filled in the typical data in the Registration table with the ID numbers from PetsopId and PetId. So now the animals data are assigned to a petshop or more petshops.

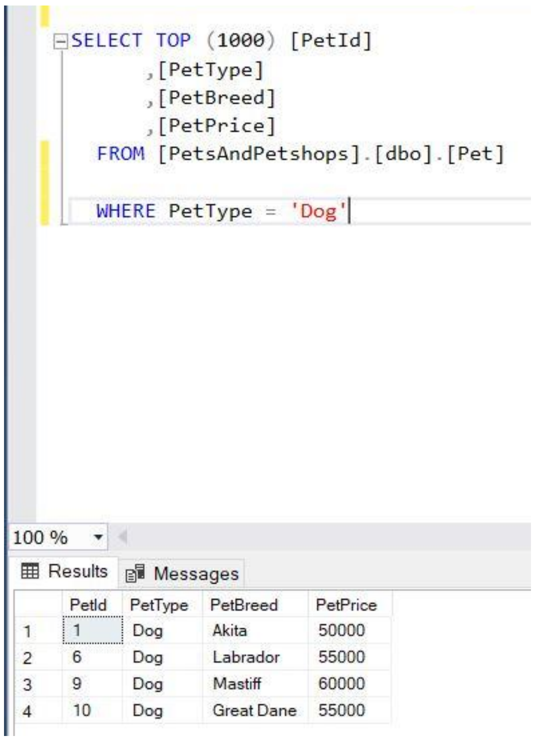

LAPTOP-JTFA9E08.Pet...- dbo.registration

	PetsId	PetshopId
	1	1
	6	1
	10	1
	5	1
	8	1
	2	4
	2	3
	3	2
	4	4
	9	2
	7	3
	8	3
▸*	*NULL*	*NULL*

SQL

Question 1.

Using the Pet tables Query with the pre-typed query code, it shows every column in that table when executed, so in order to show the pets of a given type, I then wrote WHERE PetType = ‘Dog’. This will focus on the PetType column where I then define what specific type of animal I want to show up when I print.

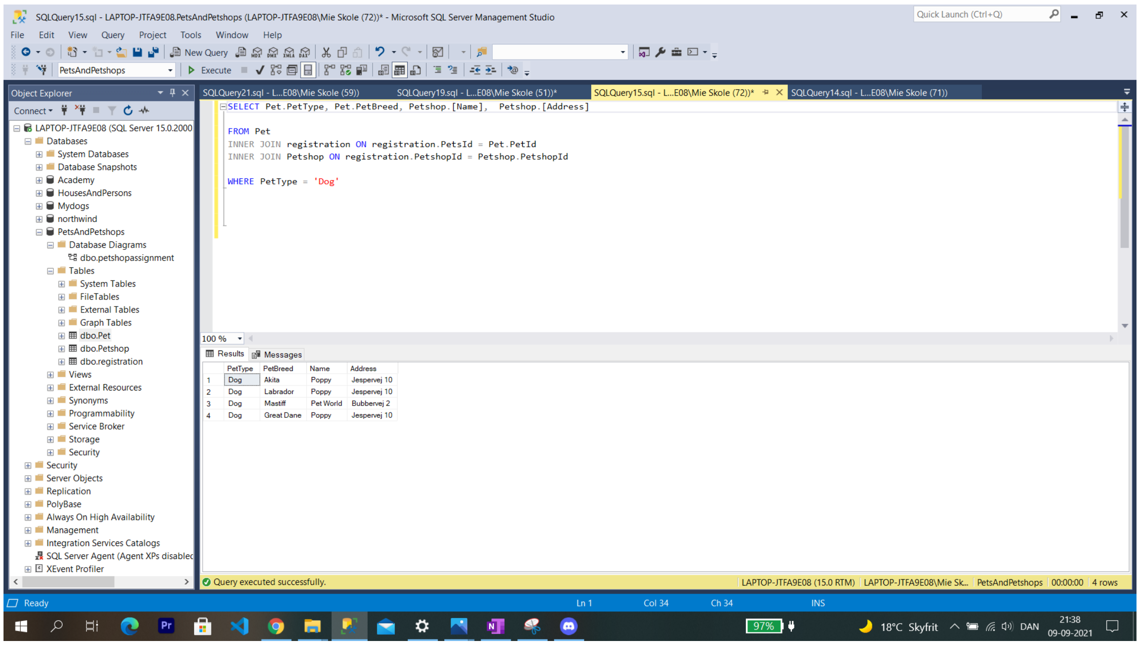


```sql
SELECT TOP (1000) [PetId]
      ,[PetType]
      ,[PetBreed]
      ,[PetPrice]
  FROM [PetsAndPetshops].[dbo].[Pet]

  WHERE PetType = 'Dog'
```

	PetId	PetType	PetBreed	PetPrice
1	1	Dog	Akita	50000
2	6	Dog	Labrador	55000
3	9	Dog	Mastiff	60000
4	10	Dog	Great Dane	55000

Question 2.

Since I made a Many to Many relation, I can then in the Query try to show where to find a specific kind of animal/ breed from the Pet table, and where to find them and in what shop from the Petshop table. In order to do that I use INNER JOIN to join the tables together by telling the program I wanted to INNER JOIN my Join table registration and it's ID's to the matching ID's from the two other tables. example: INNER JOIN registration (the join table) ON registration.PetsId (registration ID1) = Pet.PetId, meaning we want the registration PetsID to = PetId from the Pet table.

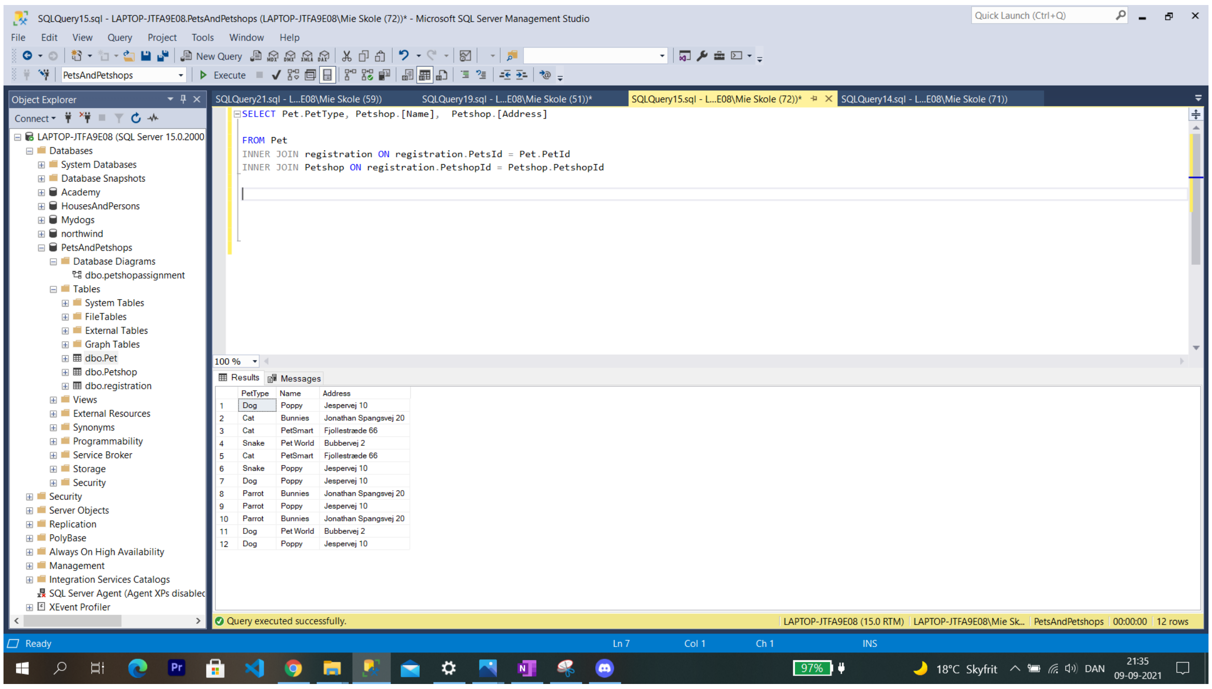


```sql
SELECT Pet.PetType, Pet.PetBreed, Petshop.[Name],  Petshop.[Address]

FROM Pet
INNER JOIN registration ON registration.PetsId = Pet.PetId
INNER JOIN Petshop ON registration.PetshopId = Petshop.PetshopId

WHERE PetType = 'Dog'
```

	PetType	PetBreed	Name	Address
1	Dog	Akita	Poppy	Jespervej 10
2	Dog	Labrador	Poppy	Jespervej 10
3	Dog	Mastiff	Pet World	Bubbervej 2
4	Dog	Great Dane	Poppy	Jespervej 10

Query executed successfully.

In this picture I use the same method as before using WHERE PetType = 'Dog', to show the one animal.
However I could also remove that part of the code to show all the types of animals and where to find them. As seen in the picture below.



```sql
SELECT Pet.PetType, Petshop.[Name],  Petshop.[Address]

FROM Pet
INNER JOIN registration ON registration.PetsId = Pet.PetId
INNER JOIN Petshop ON registration.PetshopId = Petshop.PetshopId
```

	PetType	Name	Address
1	Dog	Poppy	Jespervej 10
2	Cat	Bunnies	Jonathan Spangsvej 20
3	Cat	PetSmart	Fjollestræde 66
4	Snake	Pet World	Bubbervej 2
5	Cat	PetSmart	Fjollestræde 66
6	Snake	Poppy	Jespervej 10
7	Dog	Poppy	Jespervej 10
8	Parrot	Bunnies	Jonathan Spangsvej 20
9	Parrot	Poppy	Jespervej 10
10	Parrot	Bunnies	Jonathan Spangsvej 20
11	Dog	Pet World	Bubbervej 2
12	Dog	Poppy	Jespervej 10

Query executed successfully.