## Minute 1

How to create a one-to-many relationship in
a database design.
Today, you're going to learn what a one-to-many
relationship is in a database design.
How to represent it in a diagram, and how
to know if you need to use
a one-to-many relationship.
I use this method and this type of relationship
in every database that I've designed over
the last 15 years of my career.
By the end of this video, you'll see the technique
that I use to determine if a one-to-many is
needed in the database.
How do we design a one-to-many relationship
in a database?
There are seven steps that you can follow.
I'll explain each of them in this video.
Step one is to understand the definition of
a one-to-many relationship.
What is a one-to-many relationship?
In the database, tables are created to store
data.
Each table usually represents an object you
want to store data about, such as a person,
a car, a product, or business.
For a database to work efficiently you need
to be able to relate these different tables
together.
This is how you an store data where it needs
to be stored, and also find data that relates

## Minute 2

to each other when needed.
For example, having tables related to each other will let you find out how many orders a customer's made, how many cars are currently in each car showroom, and how many students are enrolled in each course.
How does a one-to-many relationship fit into this?
A one-to-many relationship is a way that two tables relate to each other.
It means that one record in one table can be related to many records in another table.
One example of this is a traditional relationship between a customer table and an order table.
A customer can have many orders that they place with the business over time, but an order can only belong to one customer.
Therefore, the customer to order relationship is said to be one-to-many.
That's what a one-to-many relationship is.
One record in one table can be led to many records in another table.
How do we implement that for our database?
We move on to step two.

## Minute 3

Step two is where where we write down what we're trying to design in a sentence.
Writing it down help us determine if relationship between to tables is a one-to-many relationship.
Other types of relationships are one-to-one and one-to-many, which I'll cover in separate videos.
For now, write down what you want to store in a sentence.
Some examples of this are: Store the customer information and the order that they've made.
Store the cars and the showroom that they are located in.
Or, store the students and the courses they are enrolled in at a specific date.
Write down what you want to store.
Try to keep it limited to just your two tables or two objects for now.
If you need to write down more, you can, but it's easy to just start with the two.
Let's move on to step three.
Step three is to write down the objects from this sentence.
This will help you work out what tables are going to be needed.
Take a look at the sentence you just wrote down.

## Minute 4

Identify the words that describe objects in
the sentence.
These are nouns or words that refer to things.
Let's use the examples we looked at earlier.
In the first example, store
the customer information
and the orders that they've made.
In this example,
customer, or customer information,
and the orders are the objects.
Store the cars in the showrooms that they're
located in.
In this sentence, we have cars and showroom.
Store the students and the courses they're
enrolled in at a specific date.
In this example, we have students, courses,
and date.
We now have our objects.
This will help us identify the one-to-many
relationships in our database, and help us
draw them in a diagram, and get them working
correctly in our database.
Let's move on to step four.
Step four is where we determine if it's a
one-to-many relationship.
This is an important step.
Not all of the relationships we define will
be one-to-many.
This step will let you identify if they are
or not.

## Minute 5

To work out if a relationship is a one-to-many relationship, ask yourself this question.
Using the two objects that you have identified from the previous step, does object one have many object twos, or does object two have many object ones?
This will help you work out which way your relationship goes, and if it's a one-to-many relationship.
If you substitute the word object with your objects from the previous step, your questions may look like this: Does a customer have many orders, or does than order have many customers?
Does a car have many showrooms, or does a showroom have many cars?
Does a student have many courses, or does a course have many students?
In most cases, only one part of this question will make sense to you.
Look at the question and determine which side of it makes sense, based on your understanding of the system and the business you're capturing this for.
In the first example, a customer can have many order, but an order cannot have many customers.
An order can only have one customer, so the customer order relationship here is one-to-many,

## Minute 6

or the customer to teddy bear relationship.
In the second example, a car does not have many showrooms.
It can't be in multiple showrooms.
It can only be in one.
A showroom has many cars, so the showroom to car relationship is a one-to-many relationship.
In the third example, a student can have many courses.
In this example, they can be enrolled in courses in different dates.
Also, a course can have many students.
This is a situation where both sides of the question make sense.
It means it's a many-to-many relationship.
I'll explain this in the separate video, as it requires a different process to get it working.
Now, we have our question answered.
Let's move on to step five.
Step five is to create the diagram with these two tables.
We're just going to draw them.
We won't link them yet.
You can do this with pen and paper or use a program to do it.
I've done many with pen and paper, and it's easy to do.
In this video, I'll use a diagramming tool called Lucidchart, which is my diagramming

## Minute 7

tool of choice.
I'll have a separate video on how to use this tool.
For now, all you need to know is the concept that's being drawn.
This is called an entity relationship diagram, or an ERD.
It's a common way to draw database tables and how they relate to each other.
We'll need to draw two tables.
I'll do this for each of my examples.
Let's space them out a little bit.
I've drawn the first table on the left and the second table on the right.
Heres the car table and here is the showroom table.
Further down, we have the customer table and the order table, for the second example.
In the third example, we have the student and the course table.
You can do the same thing on paper or in a program with your tables.
You don't have to use Lucidchart.
You can use Microsoft Visio, or Word, or PowerPoint, or MySQL Workbench, or any other tool that supports this kind of diagramming.
Now, you have your two tables here.
They don't need columns for now.
That's a later stage.
Let's move on to step six of the seven steps.

## Minute 8

Step six is to draw a line between the two tables.
We're just going to draw a single line.
This lets us know that there's a relationship between the two tables.
You can draw it with pen, if you're doing this on paper, or use your software tool.
In Lucidchart, I'll just need to drag a line from one table to the other, and join them like this.
We can just ignore the arrow for now.
Now, I've drawn a line between the two tables in each of my examples here.
Let's move on to step seven, which is the final step of diagramming a one-to-many relationship.
Step seven is to add to our diagram to describe the one-to-many relationship.
Our ERD will show which tables are one-to-many and which tables follow different relationships.
We add a symbol to the line between the two tables to indicate that it's a one-to-many relationship.
There are few different symbols or methods that can be used.
My preference is something called crow's foot notation, because the symbol kind of looks like a crow's foot.
This means we add some more angled lines to the end of one of the lines here.
It goes on one side of the line in the one-to-many relationship.

## Minute 9

It means the side with this symbol is on the many side of the one-to-many relationship.
Let's do that with our examples.
On the car and showroom example, the line will look like this.
This is because a showroom has many cars.
On the many side, we have this line ending here.
This will let us know that one showroom has many cars.
On the customer and order table, we know that a customer has many orders.
So, on the many side, which is on the order side, we change the icon to use this.
This is what it looks like in Lucidchart.
In your tool, it many look similar.
You can just draw the lines like this, if you're using pen and paper.
In the third example, we identified earlier that a student has many courses and a course has many students.
This means a line goes on both sides.
I'll have another video that explains why this is an issue and how to resolve it, but that's okay for now.
Now, you're done.
You've successfully gone from having some idea of what you want to store in your database for two objects, to defining the relationship between two of them and drawing it on a diagram.
That's how you model a one-to-many relationship in a database.

## Minute 10

If you learned something new from this video, make sure to subscribe to my channel, If you want to learn more about database design and development, visit databasestar.com.
That's where I share my best database related content.
Which step from this tutorial was the most helpful for you?
Was it step four with the sentence that our wrote, or step seven about adding the foreign key, or something else?
Thanks for watching.