

Hi.  
  
In this video we are going to discuss how to turn this information that we gathered so far, namely information about entity types and about their relationships into a diagram.  
  
It's called the so called entity relationship diagram or shortly, er, diagram.  
  
Why do we do this?  
  
Well, if you have a diagram, normally you just see immediately clear the overall structure and you might for example, also see redundancies and it's easier to forget something without the diagram.  
  
I should warn you, we are doing diagrams only on a fairly basic level because I do not want to spend a lot of time.  
  
Modern tools for databases allow you to really specify a diagram in a much more detailed way, such that once you have the diagram, you, you just hit a button and the diagram is automatically turned into the database.  
  
What we are going to do is we are going to draw a diagram that is more like a sketch and then you turn the sketch into a database yourself.  
  
You don't let the system do it, but you do it manually.  
  
So what are the elements of an entity relationship diagram?  
  
In our case it's just entity types which are represented as rectangles.  
  
Generally you could also have attributes associated to these entity types which are usually then represented as circles.  
  
But for the sake of this class, we are not going to put the attributes into the diagram because this can quickly overload the diagram and then the overall structure of the whole problem is rather obfuscated, rather than made more clear.  
  
Those rectangles representing entity types will be now connected with relationship, which are somehow lines of various types.  
  
So here we have an overview again.  
  
So we have entity types.  
  
So we have for example, the box containing the word employee representing the entity type employee.  
  
Then we have here a circle which says name, which would be the attribute name of the employee.  
  
But we will forget about this for the remainder of this class.  
  
But this would be if you were to put attributes into the diagram.  
  
And now we come to the most interesting bit, the relationship between those.  
  
So we saw already that we might have team and manager.  
  
And it's in a one to one relationship.  
  
So if you have a one to one relationship between two entities, you just draw a line between the two.  
  
So a line between two entity types means it's a one to one relationship.  
  
If you have a one to many relationship, what you do is you draw a line, but on the many side you draw two more lines which is kind of opening up like a fan.  
  
So that kind of expresses one to many relationship.  
  
And if you have a many to many relationship, you do this two more lines on both sides.  
  
This is to indicate the degree of the relationship.  
  
So here we have again listed match is one to many goals and student and class is many to many.  
  
This is how it is depicted.  
  
In addition, you can then also put the optionality.  
  
So we had match and goal.  
  
If you think about the optionality of this relationship, well, a match exists irrespectively whether there is a goal or not.  
  
So the relationship between match and goal is non obligatory on the match side, but clearly obligatory on the goal side.  
  
So if you have a non obligatory side in the relationship, instead of having a full line, you just write a dashed line on this side.  
  
So this line between match and goal is half dashed and half a proper line.  
  
The one between student and class we saw previously was non obligatory on both sides.  
  
That's why it's just a dashed line between the two.  
  
So dashed means non obligatory, otherwise it's obligatory.  
  
And we have the degree indicated with this two additional lines, sometimes referred to as crow feet because it looks a bit like a crow feed.  
  
So this is one.  
  
This is the easiest way I know to draw diagrams.  
  
With the dashed line you indicate the optionality and otherwise you have the degree also indicated very intuitively.  
  
So now let's go back to our bus example.  
  
We had this example of buses being allocated to routes.  
  
It was many buses might be allocated to one route.  
  
So it's a many to one relationship.  
  
And it's obligatory on both, I would say.  
  
So you just easily can draw this.  
  
And as depicted on the slide.  
  
Now, continuing with all of the relationships we had in the example, we end up with a bigger diagram.  
  
So we have it here on the slide again, where we have a diagram depicting all the relationships and their degree in the diagram.  
  
We do not specify yet the optionalities.  
  
Just not to do too much in one go.  
  
But I show you here all the different degrees of the relationships that we identified.  
  
For example, you can here see also two many to many relationship, one between town and root and one between town and stage.  
  
And in general, we will try to remove those many to many relationships because they're difficult to represent with tables.  
  
But this will be the topic of the following week for now.  
  
But in general this is an important thing to do.  
  
Once you have the first sketch of your diagram, try to think how to maybe remove many to many relationships.  
  
You also should now indeed, because we don't specify yet the optionality, think about the optionality of the relationships.  
  
Think about what are redundant relationships.  
  
It could be that one relationship between two entity types is represented in two different ways.  
  
And then we want to just Remove one of those ways and we will see indeed in this example that there's a redundancy.  
  
Can you spot the redundancy?  
  
So here the redundancy is that we have towns and stages related.  
  
And so basically stages passes through towns.  
  
And each state has one route.  
  
So if you want to know to which towns a route passes, you just have to look at the stages of the route and then cheque to which towns a stage passes.  
  
But here we have also a relationship between routes and towns directly which records through which towns a route passes.  
  
So this additional direct connection between routes and towns is redundant because if we want to have that information, we just look at the stages of the route and then cheque through which town it goes.  
  
You could say, so why not just leave the redundancy?  
  
But if you then later on store this in tables and you have the same information twice, you might change it at some point, you might change it at one spot and you don't change it in the other spot and then you are stuck with inconsistent data.  
  
So in general what we want to do is we always want to store information only in one point.  
  
So that's why we already start the diagram stage to remove redundancies whenever possible.  
  
In this case, we will remove one of the passes through relationships because it's represented by the other one.  
  
So this results in the following diagram now, where we just replaced one of the many to many relationships between two one to many relationships.  
  
We will see how to do this in more detail next week.  
  
And we removed one redundant relationship.  
  
So now we've seen the full diagram for the example, but without optionalities.  
  
But before I'm going to put the optionalities into the diagram, I want to briefly mention that there are many different notations for er diagrams.  
  
So the notation is just one of many options.  
  
Another popular option is called chance notation.  
  
And if you want to look it up, just you can consult Wikipedia for example.  
  
And there's crow's feet notation, which is very similar to what we do.  
  
And I'm going to describe it on the next slide.  
  
The difference there is that we are not using dashed lines to indicate optionality.  
  
So for example, here we have one relationship.  
  
It's a one to one relationship.  
  
And we are just indicating one to one by writing one which is like a vertical short line on both sides.  
  
This means one to one.  
  
If we have one too many, we put one on the one side and many on the that crow foot on the other side to indicate the many side of the relationship.  
  
And then many to many would be two Crow feet.  
  
And then finally, what happens if we want to have optionality?  
  
Well, if it's non obligatory on the one side, we put a zero on the other side next to the other entity, which means that this thing might be related to zero things on the other side.  
  
So it's a bit counterintuitive that we have to put that circle which indicates the zero on the other side.  
  
So if it's non obligatory, for example, on student we put a circle next to class.  
  
And if it's non obligatory in class, we put a circle next to student.  
  
So now that brings us to the final diagram.  
  
The example with optionalities where we have now all I use now this crossed feet notation where we have now all the relationships with their degree and they also indicated the relationship between town and town stop, where town stop is a new entity to resolve the many to many relationship.  
  
And there we have a zero next to town stop to indicate that the town might be just not related to anything.  
  
In some cases it's not completely clear what the optionality should be.  
  
As I said previously just then, write down what your reasoning was in the homework.  
  
Thanks for listening to this video and see you next time.