Carbon Accounting and Sustainable Designs in Product Lifecycle Management

Prof. Deepu Philip
Department of Management Sciences

Dr. Amandeep Singh Oberoi
Imagineering Laboratory

Dr. Prabal Pratap Singh
Department of Management Sciences

Indian Institute of Technology, Kanpur

Week 10

Lecture 46

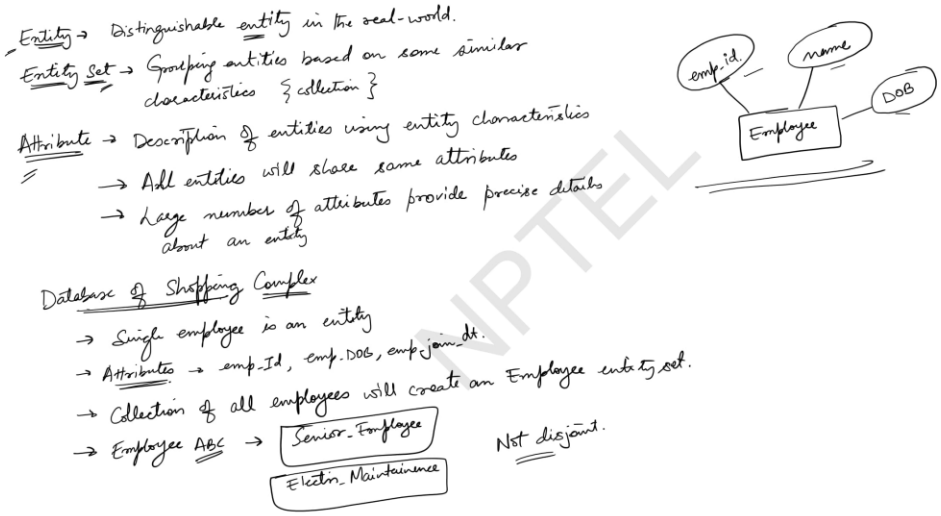
Terminologies in Database Design (Part-1)

Hello everyone, welcome again to the course on Carbon Accounting and Sustainable Designs in Product Lifecycle Management. I am Dr. Prabal Pratap Singh and we are co-teaching this course with Professor Deepu Philip and Dr. Amandeep Singh.

Till now we have seen the database design process which has six different steps and how they are iterative. We have also discussed about the data models in the database design. And now we will try to understand the diagrammatic representation of the databases and how can we read those diagrammatic representations.

How we can create our own and how these diagrammatic representations will be used further to study the real world problems and how developers try to convert these problems into a database.

Fundamental Concepts



So let us start by understanding the fundamental concepts or fundamental terminologies of these representations. So the most basic thing in these kinds of representation is an entity. So as the name says it is an entity relationship model, so we try to identify the entities in our database and using the requirement analysis. And then we will try to create a conceptual design of the database using these er diagrams.

So let us try to use a particular example of the organization where we are trying to develop a database for the employee management system. If you are trying to develop this kind of database, the basic entity in this organization is an employee. So, another basic entity could be departments in the organization, right. So, these kinds of entities needs to be defined. Now, the next thing is the collection of these entities.

So it is known as the another term we can use is entity set. So a single employee is an entity but a collection of employees in an entity set. So let us try to write down these things. So entity is it is a distinguishable in the real world entity set. So when we start grouping entities grouping entities based on some similar characteristics.

Then we will call it as an entity set, so it is nothing but a collection so let me try to show you how this employee management systems basic year diagram look like very basic diagram for a particular entity set employee So let us create it. So this rectangle is representing something. We will talk about all these representations but currently this is

an employee entity set where there will be n number of employees that can be shown diagrammatically here. Now each employee will have some kinds of characteristics.

So this could be something like employee ID or the name of the employee or the date of birth of the employee. Right now I am trying to create another diagrammatic representation by creating these ovals. So we will see why we are using this different representation. We could have used rectangle here but maybe there are some rules that is why we are not using rectangle here. So this is an entity set employee and these all are attributes or characteristics of this.

So the other basic concept is attributes. Now it is nothing but a description of entities using entity characteristics. So while creating an entity set like employee we need to identify first what are the similar characteristics of all the entities in a particular entity set. So all the employees in this entity set will have an employee id a name and the date of birth, right. So that is why all entities will share same attributes in an entity set.

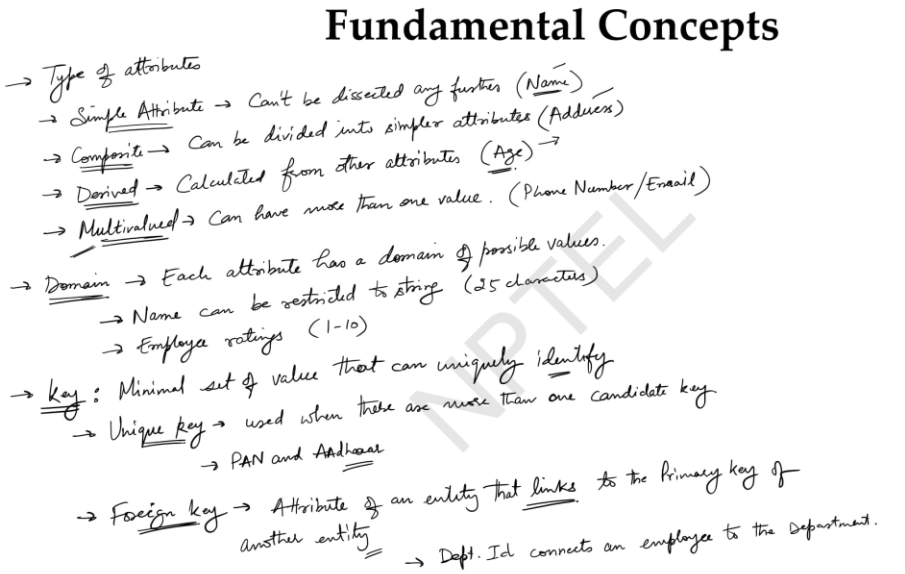
So this is a feature of this entity set now if the more the number of attributes we try to define for a particular entity set. We will have a precise definition of that particular entity set. So, that is why large number of attributes provide precise details about an entity. So if we start adding other attributes like the address of the employee the blood group of the employee. So all these attributes will try to give a precise information of each employee in the organization ,right.

So let us take an example of a shopping complex the database of a shopping complex. So here a single employee is an entity which you define here, right. It is a distinguishable entity in the real world of the shopping complex. Next, the attributes. So each employee will have some characteristics.

So it can be an employee ID. Or the employee date of birth employee join date collection of all employees will create an employee entity set Now other distinguishable thing is we may think that these entity sets are disjoint but they are actually not. Why? Because let us say employee ABC in the shopping complex employee ABC is associated with another entity set that is a senior employee entity set.

And another entity set that is electric maintenance employees. So this ABC could be a part of this entity set and this entity set both. So that is why the entities may not be disjoint. So they are not disjoint in the database, right. So let us now see what are the different kinds of attributes because we have seen that employee can have an employee

ID or the name or the date of birth. So what are the different kinds of attributes that a particular entity set can have?



So there are different types of attributes. The first one is a simple attribute. It is nothing but anything that cannot be divided any further. So can't be dissected any further, so this could be the first name of the entity that could be an employee.

So an example is the name the second type of attribute is a composite attribute this can be divided into simpler attributes. So let's say the database developers designed the database in a way so that the address of an employee is a complete single complete string now.

This could be a simple attribute but what if the designers think that we can further divide the database address attribute of in the database into simpler things like the town this city the state in which the particular address or the particular employee resides in. So these kinds of further dissection is possible in the composite attributes. So, the example is an address attribute of the database.

Next kind is derived attributes. So there are two ways to store the age of the employee. The first is to exactly store the integer that is the current age of the employee. The next better way is to store the date of birth of the employee. So with date of birth, if we want to calculate the actual age, then it is a very easy task.

But if we have only stored the current age of the employee. Then what happens is this variable is a dynamic is a dynamic variable and it will keep on changing but the database has stored a particular number only. So that is why we usually try to store the date of birth of the employee so developer will then calculate anything that they want from this date of birth. So that the dynamic nature of the data will be removed from this attribute. So these kinds of data that can be derived using the store data will come under the category of derived attributes.

So these are calculated from other attributes example is age the last kind of attribute is multivalued. So these attributes can have more than one value. So, example could be phone number like the employee may have a landline and a mobile or multiple email IDs of the employee. So, when we need to store these kinds of data in the database, we will use multi-valued attributes. Now, if there are different kinds of attributes, then their representation should also be different, right.

So, that is why when I was drawing this simple scenario, we are using different elements of the diagrammatic representation. So, we will try to see how these things vary for different kinds of attributes also. Concept is the domain. So it says that each attribute has a domain of possible values. So, for example, now whatever be the attributes you are using in your database for defining an entity or entity set, these attributes will have a particular kind of data in that.

Name could not be a numeric variable. Name will always be alphabetical, right. Similarly, address could be alphanumerical and age could be numerical. However, the age will not go beyond 100 or 200. So this will have a range that will be defined.

So that is why this domain is important while defining your attributes. So an example could be name can be restricted to string data type only. And further it can be restricted for let's say 25 characters these things vary with the developers of the database another example could be the employee ratings in the database. So, ratings could only go between 1 to 10. So, I have already shown you an example of the schema here.

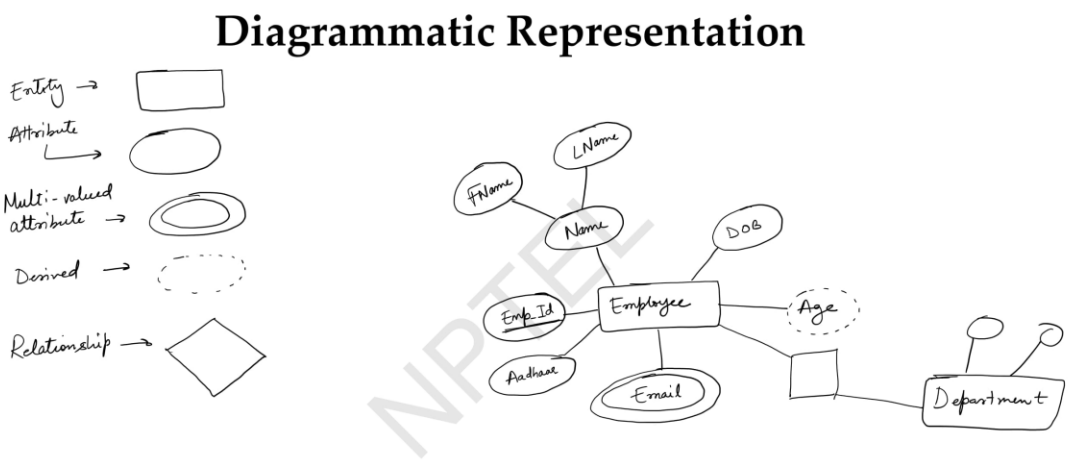
Now, you can see here that the student ID here will only be a string. Similarly, name will be a string, age will only be an integer and GPA will be real valued. So, these kinds of constraints over your particular attribute, this is a part of the domain of the database. Now the next important thing is a key in the database, so key is nothing but minimal set of values that can uniquely identify identify what identify an entity. So we can define a

unique key during our database designing and let's say the identifying information for a particular employee that can uniquely identify the employee are more than one.

So maybe the organization is providing a unique employment id the government is also providing a unique aadhar id. So these kinds of uniquely identifying attributes of a particular entity could be multiple. So the developers need to think about the future requirements of the database and try to define a particular unique key in the database. So that each entity will stay unique in the complete database table, right. So this is the role of the unique key unique key will this key will be used when there are more than one candidate key for identifying an entity, right.

So the example could be let's say pan and adhar. These IDs will uniquely identify an employee, but we will only use one of them as a unique key. The next important key is a foreign key. So since we are talking about the employee database management system of an organization, so in this DBMS system, we may have different entity sets like employees and the departments. Now, each department will have multiple employees, but an employee may be working in different kinds of departments as well.

So a particular employee can be identified with its employment ID or its Aadhaar number. But how it will be represented to get linked with a particular department. So the department ID will become a foreign key and it will be stored with each employment ID in the entity set of the table, right. So attribute. of an entity that links to the primary key or the identifying information of another entity. So, an example could be department ID connects an employee to the department.



Now, let us understand the different kinds of diagrammatic representation. So the most basic thing in the database is an entity. So let us start with it entity we usually define an entity with a rectangle the next thing is attribute. So attribute is usually done with an oval or sometimes called an ellipse. Now there are different kinds of attributes as we have already seen so a multi-value attribute, this could be phone number or email when we are trying to store these kinds of data in the database we will use double over.

The next important attribute was derived attribute and we gave an example of calculating the age of the employee in the database. So this can be represented by a dashed oval. It represents that it got calculated by using the information from the database. Another important representation is the representation of a relationship. Relationship between what?

Relationship between entity or entity sets. So this is represented by a diamond. Now let us try to develop a basic employee entity set and create its diagram. So since employee contains the collection of employee entity set. So we will use an entity representation rectangle and call it as an employee now it will have different kinds of attributes the first attribute could be named.

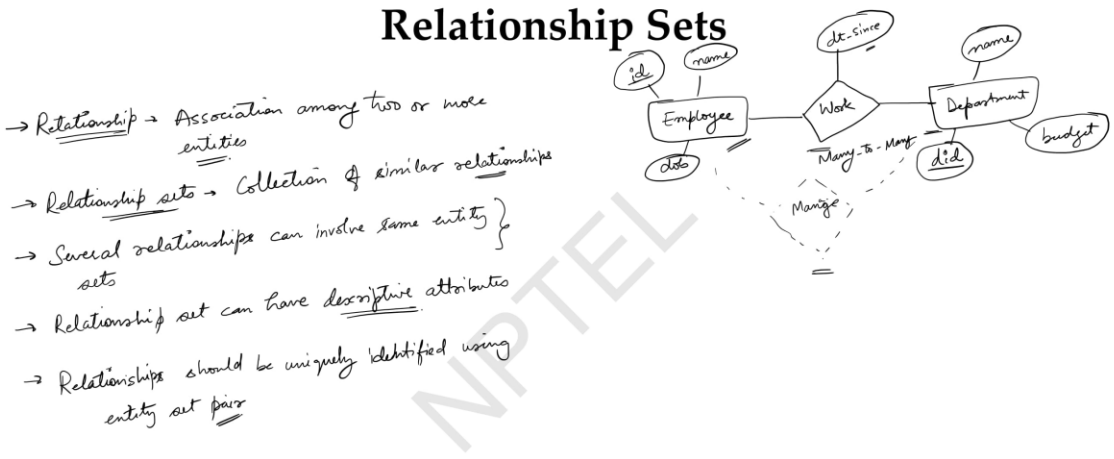
Now name is a composite attribute which may have more basic attributes in it. So this could be the first name or the last name, since these are simple attributes we will try to

create this using simple ovals. Now an employee can have an date of birth also this is also a simple attribute. So we can draw it like this the age so by using the date of birth we can calculate the age of the employee and this age will be shown as a dashed oval, right. Further let's say the employee will have an email id.

Now the organization will provide an email id and the employee will have its his or her own personal email id and developers decided to store both of these email ids. So they will show this representation by using this multi-valued attribute because an email id will have multiple values. So we can write email and we will draw two concentric ovals. Now this representation is telling you that when we are trying to store email in the data in the actual database tables. This email attribute will have more than one email ids that can store more than one email ids, right.

And we can have another identifying information that could be an employee id that is also a simple attribute and a government authorized id that could be aadhar. So, this is a simple representation of an employee using the ER diagrams right and we have not shown here any relationship with other entities. So, other entity could be departments in the complete organizations database. So, department could be another entity set. It could have its other different kinds of attributes with it.

And then we will have some kind of relationship with this entity set, with the department entity set, right. So this is a complete diagrammatic representation of a particular employee management system in an organization.



Now let us understand we have already talked about the entity sets that is the collection of entities. Now there could be a collection of relationships as well so to understand that let us first create a simple diagrammatic representation of to understand this relationship sets. So in an organization again we are creating an employee entity set which will have an attribute let's say id and.

This ID is the identifying information so the primary key will always gets underlined in the ER diagram. So even while creating this we can say out of employee ID or other ID we are saying that this employee ID is the actual unique key of this database schema. Similarly here the ID is the unique key. Another thing could be named as an attribute. And another thing is date of birth right, similarly there are departments and department will have a department id.

And it is a unique identifying information department will also have a name and department. Will let's say have a budget right now what kind of relationships could be possible between an employee and a department employee can work in a particular department. So let's say we define this relationship as work. And this work relationship can have its own attributes like date since. So since when a particular employee from the employee entity set is working in a particular department from the department entity set.

So now let us understand these relationship sets. Relationship is nothing but an association among two or more entities. All right. So analogous to entity and having an entity set, we have relationship and relationship sets. So it is nothing but a collection of similar relationships and we are drawing it using a diamond. So several relationships can involve same entity sets.

So what this feature of relationship sets is saying is that let's say we are using the employee entity set and the department entity set to define the work relationship. But it is not a criteria that once these entity sets are used, they cannot be reused while defining the database schema.

These both employee and department can have more than one kind of relationships and these relationships can be shown with other relationship entity relationship sets. So as an example the other kind of relationship could be a manager so using the entities from the employees person could be a manager or not of the department. So other relationship could also exist that shows how we are managing these departments or who is managing from the employee entity set for a particular department.

So we can write manage right so other interesting thing about this is a relationship set can have descriptive attributes So these attributes will try to define how these relationships are working between two entity sets. So here we are using a descriptive attribute called dateSense. So what it is storing is it is storing that a particular entity is working since when in a particular department, right. So these kinds of descriptive attributes are also possible by using relationship sets.

Another important thing is these relationships should be uniquely identified using entity set pair. So, while defining a particular relation like work or manage, we are not using the attributes of this relation. We are only trying to first identify the relation between two entity sets, not their attributes.

Thank you.