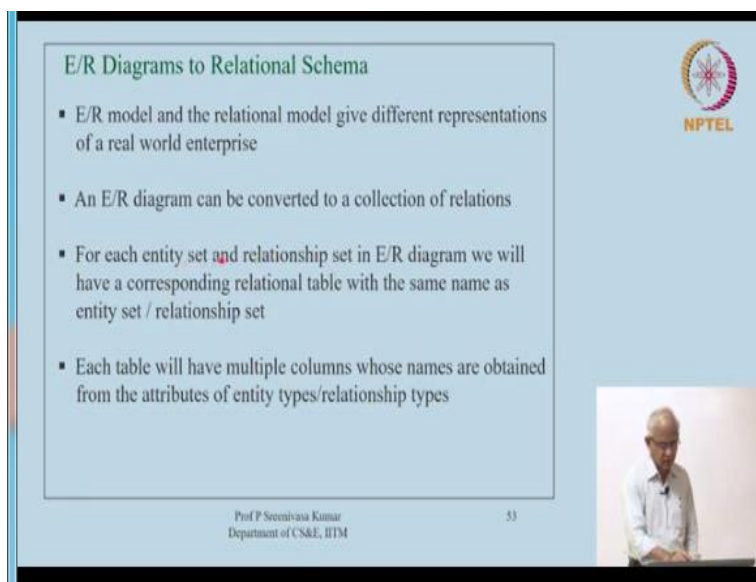


**Database Systems**  
**Prof. Sreenivasa Kumar**  
**Computer Science and Engineering Department**  
**Indian Institute of Technology-Madras**

**Lecture-12**  
**Convert ER-Model to a Relational Model**

So basically in this lecture I want focus on how the information that we have represented in the E/R model you know.

**(Refer Slide Time: 00:29)**



The slide is titled "E/R Diagrams to Relational Schema" and contains the following bullet points:

- E/R model and the relational model give different representations of a real world enterprise
- An E/R diagram can be converted to a collection of relations
- For each entity set and relationship set in E/R diagram we will have a corresponding relational table with the same name as entity set / relationship set
- Each table will have multiple columns whose names are obtained from the attributes of entity types/relationship types

The slide also features the NPTEL logo in the top right corner and a small video inset of Prof. P. Sreenivasa Kumar in the bottom right corner. At the bottom of the slide, it says "Prof P Sreenivasa Kumar, Department of CSE, IITM" and the number "53".


Information about an enterprise database that we are trying to design we will capture that in E/R model which is a conceptual level model into the relational model. We have just studied this relational data model and we know that you know it gives certain concepts like relations, keys, foreign keys and things like that. So now we focus on how to translate or map the information that we represented in the E/R model to the relational model.

So they give in some sense are different representations of the real world enterprise and now we were trying to get the relational scheme. Now, so the general approach that we will do we will use is that for both entities entity types right entity types as well as the relation types, we will basically map them into relations ok. So notice that we have in the E/R model context we have these 2 concepts, the concept of an entity type, the concept of a relationship type.

And of course there are variations of this entity type like strong, weak and all that ok, but when it comes to relational data model we have only one the concept of a relation. So basically for each entity set as well as the relationship set, relationship set or type right entity type or relationship type we interchangeably use this words. So for each entity set and relationship set in the E/R model we will have a corresponding relational table.

So let me use this word table in this set of slides to kind of make it very clear that we are talking about relations. Now the relation table with the same name as that relationship as the entity type or the relationship set. So that is the basic approach and then each of these tables will have multiple columns, so kind of we are calling attributes as columns here, whose names are obtained from the attributes of the entity types or the relationship types. So we will see the details as to how exactly we should obtain these names ok.

**(Refer Slide Time: 03:30)**




**Relational representation of strong entity sets**

- Create a table  $T_i$  for each strong entity set  $E_i$ .
- Include simple attributes and simple components of composite attributes of entity set  $E_i$  as attributes of  $T_i$ .
  - Multi-valued attributes of entities are dealt with separately.
- The primary key of  $E_i$  will also be the primary key of  $T_i$ .
- The primary key can be referred to by other tables via foreign keys in them to capture relationships as we see later

Prof P Sreenivasa Kumar  
Department of CS&E, IITM

54



So let us first focus on strong entity sets ok, so let us say there is a strong entity set  $E_i$ . So what we will do is to create a table  $T_i$  for that particular entity tab set the name of the table will be the same as in the name of the entity. But in this case we just representing it as you know symbols like  $E_i$ ,  $T_i$ . Now the entity set will have attributes right now it will have different kinds of attributes.

So we will take the simple attributes and then if there are composite attributes what we do basically here is to take the simple components of those composite attributes ok, each of these composite attribute has components right. So and that component itself can be again composite right and so we kind of go to the leaf level in some sense and pick up all the simple attributes simple components of all these composite attributes and those names alone will be the attribute names for the table T i ok.

One thing that we have to do is to handle this multi-valued attributes of these entities separately and I will talk about how to handle multi-valued right now we will ignore the multi-valued attributes of the entity set E i ok. We will take this simple attributes and composite attributes ok, for composite attributes what we will do is to pick up their simple components, components and then check whether they are simple or not and then pick up that ok.

So basically you might also realize here that we are in some sense losing some information like for example, while modeling a person we might say that the name has 2 components, the last name and the first name ok and these 2 things together are called name that is how the model hand as composite attribute. So we will now basically ignore that name thing and then only take the L name F name if that is the name those are the attributes the names that are being used.


Then we will simply pick up L name and N name as the attributes of the table we are creating and a little later we will see how exactly to handle multi-valued attributes of these entities. Now we also now decide the primary key for this table T i, so the primary key will be basically decided by picking up the primary key of the entity T i. So whatever was the primary key attribute of the entity and that would be in at attribute of the table we have created and so we will make that as the key here.

Now as we will see later basically the primary key can would be actually referred to by other tables via the foreign keys you know in order to capture relationships ok. We will see that say here relationship sets also are going to be translated into tables. And so we will see how exactly we will use foreign case in order to capture that associations ok


**(Refer Slide Time: 07:41)**

### Relational representation of weak entity sets

- Let  $E'$  be a weak entity owned by a strong/weak entity  $E$
- $E'$  is converted to a table, say  $R'$ , where...
- Attributes of  $R'$  will be
  - Attributes of the weak entity set  $E'$  and  
Primary key attributes of the identifying strong entity  $E$   
(Or, partial key of  $E$  + primary key of the owner of  $E$ ,  
if  $E$  is itself a weak entity)
  - These attributes will also be a foreign key in  $R'$  referring to the table corresponding to  $E$
- Key of  $R'$  : partial key of  $E'$  + Key of  $E$
- Multi-valued attributes of  $E'$  are dealt separately as described later



Prof P Sreenivasa Kumar  
Department of CS&E, IITM



So move on, let us look at weak entity sets, entity sets and weak entity sets, recall that weak entity sets are basically entities that do not have an independent existence and they kind of depend on some other entity for their relevance to the information system. So we have weak entities, so let us say some  $E'$  is a weak entity owned by some strong entity  $E$  it could even be owned by another weak entity but we will take up that case a little later.

So let us say it is owned by a strong entity, so this  $E'$  is going to be converted into a table say  $R'$ , where you know we will pick up we will have to make up their attributes for this  $E'$  the table. So these attributes of  $R'$  the table will be basically the attributes of the weak entity set  $E'$  and in addition the primary key attributes of the identifying strong entity are the owner entity there is an owner right, for because  $E'$  is a weak entity it has an owner.

So pick up the primary key attributes of the owner and include that as attributes of  $R'$  ok. Now you can see that the attribute of  $E'$  you know will have only what is called a partial key ok. And so in order to distinguish between the various entities in the weak entity set, we need the information about the owner entity. So the owner entity's information will come because we are including this primary key attributes of that ok.

Now let us look at this case we suppose the owner is itself is a weak entity in which case the partial key of that entity plus the primary key of the owner of that would be the attributes that we

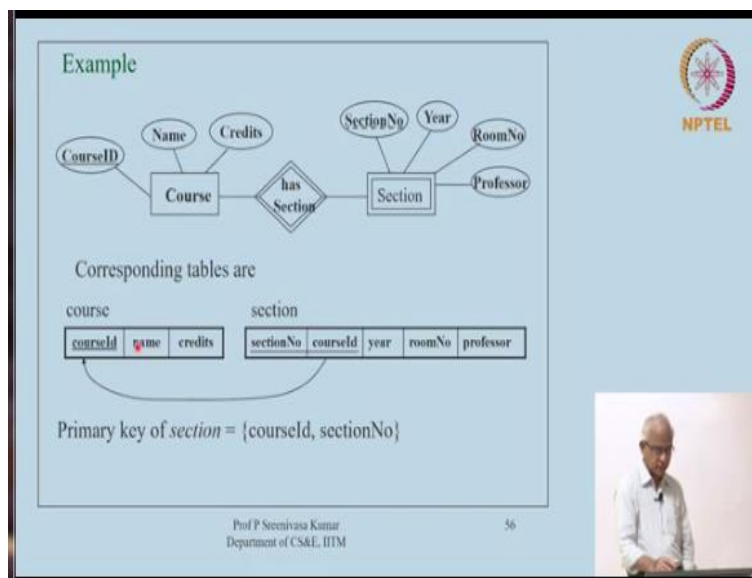
will include. Of course this chain will go on because it has to end with some strong entity ok. So basically you can see that in case a weak entity is owned by another weak entity. Then the approach we can follow is basically first map the owner range, owner weak entity into a table ok.

And then bother about mapping the weak entity ok, so that is how we will need to handle the weak entity type. Now these attributes these means these are the primary key attributes of the identifying relationship ok and the so will in fact also be a foreign key in R prime referring to the table corresponding to the owner entity. The owner entity we has been mapped to a table.

And so now these primary key attributes of that particular that we have included into the table of R prime will be the foreign key in R prime referring to the table corresponding to E. We will see some examples and then it will become clear and what should be the key of R prime, is the partial key of E prime plus the key of the owner entity. The key of owner entity has anyway being included into this table.

And so that set of attributes plus the partial key together would be the key for R prime the table R prime and then multi-valued attributes have to be dealt separately as we will see a little later ok.

**(Refer Slide Time: 12:37)**



And now let us look at an example to clarify this situation, we had this kind of an example in our earlier discussion. So section is a weak entity owned by course ok and the identifying relationship is this has section and partial key is section number ok. So let us assume that the course has been already mapped to a table like this course Id, name, credits ok. So it is a strong entity and so we will map it directly into a table with these attributes and then the whatever is the key for the entity will become the key attribute for the tables also.

Now this is a weak entity and it is owned by a strong entity and so all these attributes will come but then the primary key of the owner entity also is added as an attribute and in case that owner entity is itself a weak entity as I was pointing it out in the previous slide. Then we will first map that into a table in which case you know it will get the partial key of that plus the you know primary key of it is own identity as the key.

So those 2 things together will now come to the table here, so like that we will be able to capture the information. So section number course Id now together will be the key, so course Id because it is the primary key of the owner, section number it is because it is the partial key of the weak entity. So these 2 things together will be the key, so we underlined them and now the course Id that we have introduced here ok, borrowed from the owner entity will or obviously should be a foreign key referring to the courses.

In general the primary key attributes that we introduced from the owner will be the foreign key attributes that refer to the owner relation, whatever is the owner relation ok. So I hope this so let us go back to this slide, so attributes of R primary or attributes of the weak entity plus the primary key attributes of the identifying strong entity. And in case that is a weak entity the partial key of E plus the primary key of the owner of E will be the attributes.

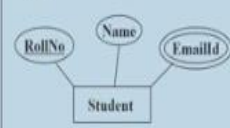
And these attributes will be a foreign key also in R prime referring to the table bar and so on. So this should be clear now, any questions in that.

**(Refer Slide Time: 15:47)**

### Relational representation of multi-valued attributes

- One separate table for each multi-valued attribute
- One column for this attribute and
- Column(s) for the primary key attribute(s) of the table that corresponds to the entity / relationship set for which this is an attribute.

e.g.,




student


rollNo	name
--------	------

mailds

emailId	rollNo
---------	--------

Prof P Sreenivasa Kumar  
Department of CS&E, IITM





Now let us talk about multi-valued attributes. Multi-valued attributes actually give rise to a special kind of a situation recall that in the relational data model. In the relational data model I have been emphasizing that a cell of a table in a particular row contains only atomic values. It cannot contain set of values or lists of values or anything else it contains only one value, so these multi-valued attributes as you know by its definition is a bunch of values right.

So we cannot directly store it as a value in a cell of our any relational table because of the constraints that the relational model has ok and so what happens here is that we need to kind of introduce a separate table for each of the multi-valued attributes that the entity has. So let us take this just one example here, so Email Id, so we will do this after the entities have been mapped into tables.

So right now student has already been mapped and it does not have the multi-valued attribute, so it is as we thought. So that is why we did not mention about the multi-valued attribute when we are mapping, we said they will be dealt with separately. So now take up this multi-valued-attribute create a separate relation for that whatever be the so you will not invent the name for the relation.

And then introduce this attribute name and then the primary key of the entity for which this is a multi-valued attribute and then make them together as the key. So you can see that why is this needed you can see that since roll number which kind of identifies the student you know has

association with multiple Email Ids we will have to repeat the roll number with all of those Email Ids that is the only way to capture a set of entity, a set of values who associated with a particular entity.

Because this relational model has a constraint that this particular column can only have exactly one value it cannot have list or set of values right. So roll number Email Id will obviously be the key together now unless both of them are kind of given you cannot identify the rho in a you a tuple in this relation ok. So it is also obvious that this roll number is referring to the student entity and so it will be a foreign key referring to the corresponding entity I mean the table corresponding to that entity ok.

So 1 column for this attribute what is that attribute is this multi valued attribute. So 1 column for that and then the columns for the primary key attributes, so in general this primary key there could be multiple attributes in the primary key. So 1 column for each of those multi print corresponding to the either the entity type or the relationship type for which we are we have this as the multi-valued attribute.



Recall that even relationships can have attributes and they can also be multi-valued in the case of E/R model and so when we deal with multi-valued attributes we would already map that relationship type into a table ok and so we take the key attributes of that table and then include them here ok. So that is how we have to deal with multi-valued attributes ok, so moving on let us see how we can so far we have focused on entities, let us see how to handle relationships in the E/R model.

**(Refer Slide Time: 20:39)**



### Handling Binary 1:1 Relationship

- Let S and T be entity sets in relationship R and S' and T' be the tables corresponding to these entity sets
- Choose an entity set which has total participation in R, if there is one (say, S)
- Include the primary key of T' as a foreign key in S' referring to relation T'
- Include all simple attributes (and simple components of composite attributes) of R as attributes of S'
- We can do the other way round too  
– lot of null values



Prof P Sreenivasa Kumar  
Department of CS&E, IITM

58

Let us first take up this binary 1 to 1 kind of relationship, so 1 to 1 relationship as you can recall is involves it is binary. So there are 2 entities, so S and T let them be the entities that are in relationship R and let us say we have already mapped S into S prime, T into T Prime right. So we did the mapping of the entities and so we have got 2 tables and now we have to handle this relationship.

One approach to handle this relationship is the you know a kind of a simple you know interesting approach, where we kind of introduce what is called a what can be probably called as a lookup table ok. So which will basically just have 2 attributes when whatever is the number of attributes basically contain the key of S key of S prime and key of T prime together that is all that is one approach, we will see that actually I will come back to that little later.

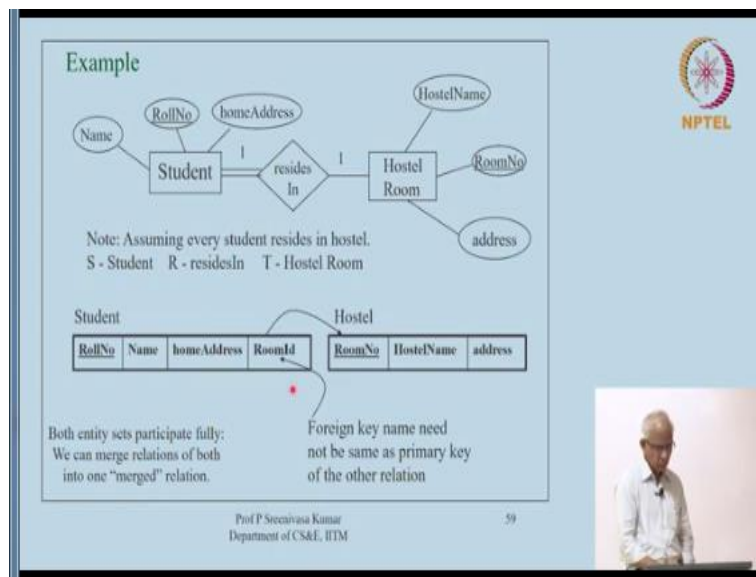
But let us see how we can handle the specific case of 1 to 1 relationships, let us say choose an entity which has a total participation in R. One of them is likely to have you know a total participation, let us say if there is 1, let us say S has a total participation. Then in this case what we can actually do is to include the primary key of T prime, the other relation as a foreign key in S prime referring to the relation or the table T prime.

And in case this relationship type itself has some attributes then map all those simple attributes or simple components of this composite attributes as attributes of S prime ok, so this is 1

approach, I will show you an example and then it will become clear as to what we are exactly doing. So we give reference to the one that has total participation and we basically change the relation scheme of that particular a thing by including the primary key of the other as a foreign key here ok.

In this case we can in some sense do the other way around also but I will show you with an example ok.

**(Refer Slide Time: 23:36)**



So let us look at this student entity , hostel room entity ok and student is identified by roll numbers and hostel room for a moment let us assume that hostel rooms are identified by room number, room number is the key for hostels let us say ok will make a specific assumption here. We want basically 2 entities with their keys, so we have taken student and hostel room and let us say it is a 1 to 1 relationship that means every student is given a hostel room ok and this is a hostel room can only have 1 person if at all.

So this is a total participation that means every student is given a room and this is partial that means some hostel rooms could be unallocated it is a luxury of course and to have a hostel rooms not allocated ok. So let us say the so S is the student R is that relationship T is this one. So what basically we have done here is that the student would have let us say student is actually mapped

to student table in which case we have roll number, name and home address as the 3 attributes only up to that will be the student entity.

And the hostel room will be room number, hostel name, address because these are the 3 attributes room number is the key, so we will write it here. Now in order to take care of this, what we basically have done is to augment this side which has a total participation by basically adding 1 foreign key to take care of this relationship one key sorry one attribute that is actually a foreign key referring to the other relation.

So basically we want to capture as to what is the room to which this particular student is associated with. So we augment the student and then introduced a new attribute room Id and then since it is value ok this values are going to be used to refer to the hostel rooms we make it as a foreign key not refers to the hostel table ok. Now you might say that why do not we do the other way round, why do not we include the roll number in the hostel table and saying that this particular hostel room is assigned to so and so student.

In this case we could even do that because it is a 1 to 1 relationship, so there is exactly 1 student to which it is been assigned and so you could actually augment the hostel table right hostel I mean the room this table with a roll number as with roll number or student Id or whatever you can call that attribute. And then make that as a foreign key that refers to the student just like we did here we can do it in the other way not also.

But only issue would be that since we have assumed that this is not say partial participation. So there is lots of there might be several rooms which are not allocated in which case they will the value of roll number or a student Id here would be null in case it is not allocated it would be null. If it is allocated it will be the student Id of the corresponding thing, so there will be lot of null values if we choose the other relation which has partial participation, you would also go to another extreme and then ok.

So this foreign key name in general the foreign key name of course need not be the same as a primary key name. But it could be some other name right, for example I said here room Id which

is basically refers to room number. Now we could go also to a little bit of a extreme approach in which we can merge these 2 relations together and then put them you know as one big relation we could do that in this case.

Because it is a one to one you know there are exactly as many if you know it is one to one and then let us say both of them. So both entities participate fully say this is an important thing if both entities participate fully that means every hostel room is also you know participates here saying that it is assigned to some student no room is ok. If that is the case, so if both entities participate fully then we can basically merge these 2 relations.


And then get one abroad relation a merge relation which is a little I do not advice that mainly because these entities have you know their own identity in some sense. So we would like to capture them as separate relations and then the association or the relationship is best captured using foreign keys rather you know in some sense merging them because they would not then logically make sense.

So as far as possible keep a relation to ensure that a relation is you know obtained from one entity. An entity is represented as one relation ok, so that is the approach that we can do , the other approach that I was referring to like lookup table kind of approach is that let this particular student relation be separate roll number, name, home address, let this be separate room number, hostel name address.

We will create a third table in which we just you know capture the association only what is the roll number, what is the room Id associated room Id that we can capture separately as I look up table in some sense right. And then of course in that relation you will have to make the room Id as a foreign key that refers to roll number and the student Id is a foreign key that refers to roll number and room Id as a foreign key that refers to the first one.

You can have a third relation leaving these 2 things as they are but we do not normally advice that in the case of these 1 to 1 relationships because you can manage the situation by augmenting one of these relation tables ok.

**(Refer Slide Time: 31:21)**




**Handling 1: N Relationship**

- Let S be the participating entity on the N-side and T the other entity. Let S' and T' be the corresponding tables.
- Include primary key of T' as foreign key in S'
- Include any simple attribute (and simple components of composite attributes) of 1:N relation type as attributes of S'

Prof P Sreenivasa Kumar  
Department of CS&E, IITM

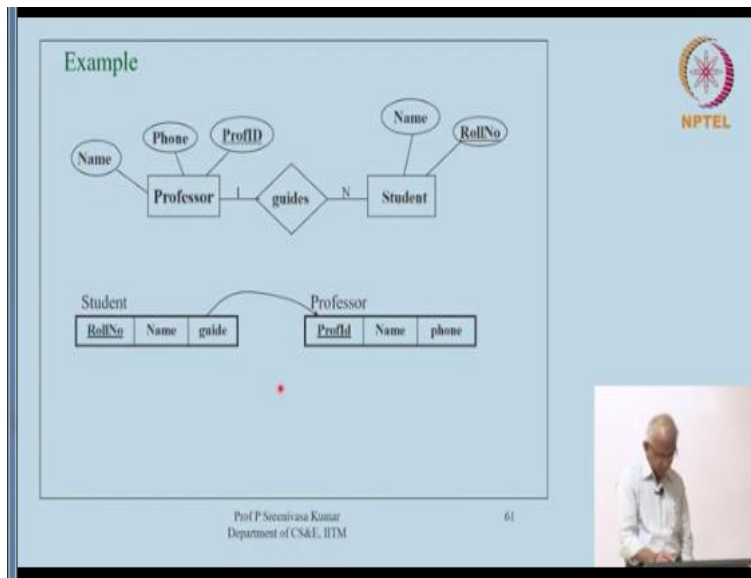
60



So now let us move on to how to handle a one to many relationship right, in this case what we do let us say S is a participating entity on the N side one to-many right. So there is a one side and N side and T is the other entity let us say and let us say we have already mapped S to S prime T to T prime these are the corresponding tables now what the approach here is in order to handle this 1 to many relationship is to kind of modify the N side table.

The table corresponding to the N side entity and include the primary key of the other as the foreign key here and then include any simple attributes or simple components of the composite attributes of that relationship itself as attributes of S prime. So basically we augment the scheme for S prime which is the table corresponding to the N side entity. So again this will become clear if we show an example.

**(Refer Slide Time: 32:42)**



Let us say professor, guides, many students say each professor guides, many students, but each student is guided by exactly 1 professor. Each student is guided by exactly 1 professor but a professor guides many students, let us say that is the one to many relationship and so what we are now saying is that take the N side entity that entity had the N entity has come this side, the N entities here ok.

The N side entity is student, so it has roll number and name I just simplified the things you mean it could have several other attributes but for the illustration purpose let us just have roll number and name. Now this is the N side entity which is so augment this, augment this by introducing a new attribute and make that as a foreign key that refers to the other relation, what is the other relation, the other relation is the 1 side entity which is professor.

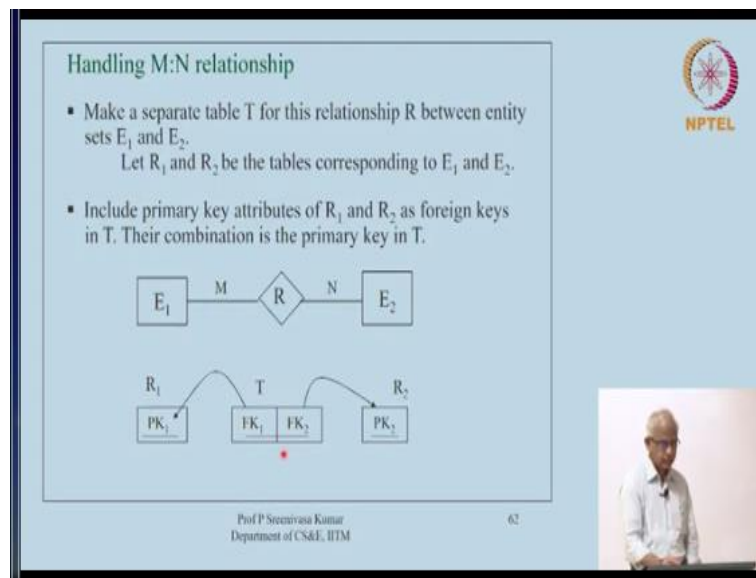
So it has name, phone, professor Id, so name, phone, prof Id, now basically what we have done is to for each student in each student rho we now introducing a new attribute called the guide of the student which will be an employee Id that refers to the professor who is the guide of the student. So notice that we cannot do the other way around here because a professor guides many students.

And so if you want to now augment this professor side table with a new attribute saying that you know whatever students guided then that will be multiple values and you cannot put multiple values into a cell and that causes trouble. So we actually solve the problem by augmenting of the

N side of the N side entity, I mean the table corresponding to the N side entity by appropriately introducing additional attributes ok.

So let us just reread this again if you want, so let S be the participating entity on the N side and T the other side so include the primary key of T prime as a foreign key in S prime that is what we are saying. And in case there are attributes for the relationship include them also as additional attributes in S prime. So in case there is an attribute called begin date or something like that, you basically make that as an attribute on the student side itself ok. So that is about the 2 specialized relationships 1 to 1 and one to many.

**(Refer Slide Time: 36:04)**



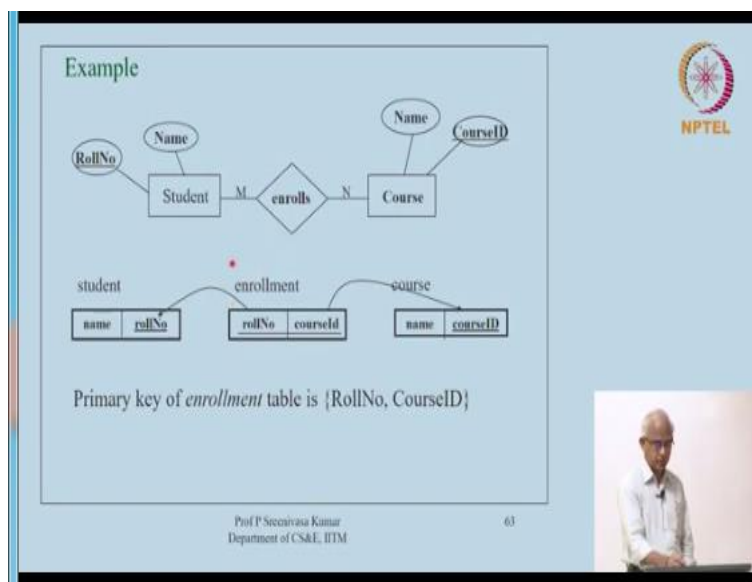
Let us now look at the many to many relationship, here there is actually no option to either augment the 1 side or the other side. So now we basically have to go for a third relation ok, so make a separate table T for this relationship R ok between entities even  $E_1$  and  $E_2$  ok let us say even in it were already mapped to  $R_1$  and  $R_2$ ,  $E_1$  is mapped to  $R_1$ ,  $E_2$  is mapped to  $R_2$ , so in order to handle this many to many relationship.

Now introduce the third table where you basically include the primary key attributes of  $R_1$  and the primary key attributes of  $R_2$  as foreign keys ok. Because this is what basically captures the association ok, here is 1 entity, here is another entity. These 2 entities are associated with each

other. So let us take that combination and then give a provision for expressing those combinations in some sense ok.

Now and of course once we do that these FK 1 will be the obviously the foreign key that refers to the primary key of R1 and FK 2 will be the foreign key that refers to the primary key of K2 and these 2 things together will be the key for the new table in some sense they lookup table ok.

**(Refer Slide Time: 38:01)**



Again an example will clarify the situation, student enrolls for courses, so course is an entity identified by the course Id. Student is an another entity roll number is the key and now this is a many to many relationship, student enrolls for many courses and each course is unrolled by many student right, that is why it is many to many relationship and so you cannot ok basically let us say student is mapped to roll number, name, course is mapped to a course Id name ok.

Now in order to handle this we have to necessarily go for a third relation which basically captures what roll numbers are associated, what course Ids and these things together for every course I say for a, if you from a student side if you see if he is doing 5 courses then that roll number will get associated with those 5 courses and will get listed in the as tuples here and if course Id one particular course is being done by 40 people.



Then all those 40 people along with that course Id will be there as tuples here ok. So you basically have to realize that we cannot augment the student table with a course Id because there are multiple courses that student is doing and we cannot augment the course with a student Id because there are multiple students doing the same course. So we introduced a new relation which is in some sense called the.

You know in some sense a cross reference kind of relation or it can also be called as a lookup table kind of relation which basically allows us to look capture the association and then in order to get more information about this we need to refer to the table. So we make this as a foreign key, in order to get more details about the course we have to refer to the course table and so we make this as a foreign key ok.

So this is how we capture many to many relationship, so you might, so since the other relationships like 1 to 1 and 1 to many or you know in some sense a special cases of many to many. We could take this approach in order to handle the other cases also but then know for those cases as we have argued earlier there are better solutions, where now we can avoid having a third relation instead we can simply go for a better solution where we either augment the one side or the another side ok.

**(Refer Slide Time: 41:12)**

### Handling Recursive relationships

- Make a table T for the participating entity set E (this might already be existing) and one table for recursive relationship R.

Example

Course

CourseID	Credits	Timing
----------	---------	--------

PreRequisites

preReqCourse	CourseID
--------------	----------

Prof P Seemivasa Kumar  
Department of CS&E, IITM

64

Now as far as finally we have to handle these so called recursive relationships, where if a course I mean a particular entity is participating more than once in a relationship in appropriate kind of rolls. So but then this is if you think about it is not a major issue because we basically can treat it as 2 different entities and then basically create foreign keys to refer to the same table more than once ok.

So let us look at this, so is prerequisite of is a recursive relationship involving course in 2 roles once as a course, once as a precursor. So course itself has course Id credits and some I do not know timing should not be here actually the department offering department take it as offering department course Id, credits, offering department let us say . So now in order to handle this a respect to whether it is many to many or otherwise we basically create a new table.

And then create attributes corresponding to the role names, prerequisite and course and then make both of them as foreign keys that refer to the entity ok. So this is how the all the major constructs in the relational model can be mapped and the information can be mapped into the relation all the major elements of the E/R model can be handled and then we can capture the information into the relational model.

So please practice this on those exercise problems that I have given after drawing the E/R model try and translate that into a relational model using this systematic some 6, 7 steps are there here. So using this approach you map it and then see how exactly it is coming out, so that you can also try and repair your relational model. Now finally one point is this approach of you know keeping a crosstab cross reference kind of a table generalizes to n-ary relationships.

So in case there is a n-ary relationship then basically we can have a this third table which captures this relationship. We will have one more attribute which again will be a foreign key that refers to the primary key of the other. So in general we can use this approach to capture the n-ary relationships in the E/R ok. So like that we can capture all the details that are mentioned in the E/R ok any questions.

So for those exercise problems please try out and map them into relational models and then try and then see how exactly they are coming of ok, so let us stop.