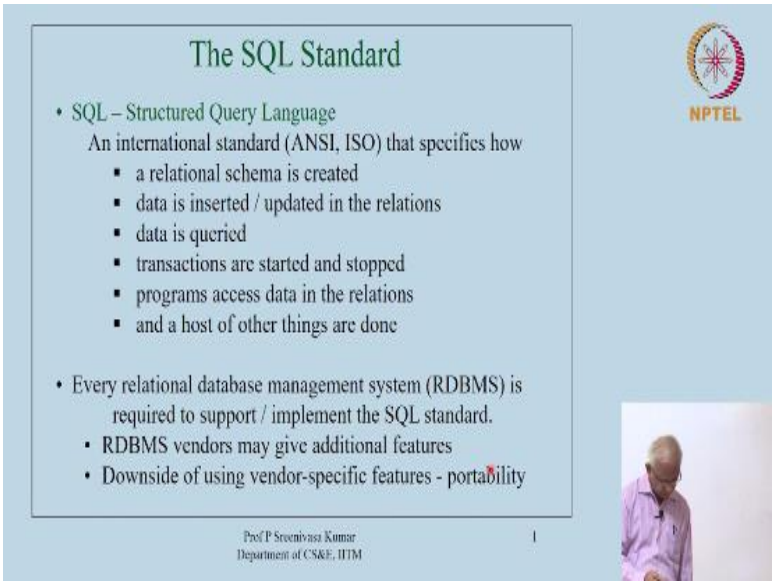
Database Systems
Prof. Sreenivasa Kumar
Department of Computer Science and Engineering
Indian Institute of Technology – Madras

Lecture - 13
Data Definition Using SQL

We want to talk about the new module which is the SQL Standard.

(Refer Slide Time: 00:28)



I have uploaded the slides for TRC. So please go through the remaining. There is one important aspect about the Tuple Relational Calculus which I could not discuss in class because of power failure last class, but please go through that. It is something to do very important aspect. It is something to do about safe query; some Tuple Relational Calculus queries are called safe Tuple Relational Calculus queries.

So information is there in the slides. Okay let us get started off with this; so sequel it is also called sequel SQL it originally expanded as Structured Query Language, but it is much more than a query language. So now we think about it more as a standard so that is why the title for module is the SQL Standard. So it is an international standard that specifies how a relational database is to be setup and so the various details about that.

How do we create the schema and then how data is to be updated and inserted into this relations, how data is to be queries one uniform way of querying the data that happens to be the principle you know function of this particular whole standard and that is why the name

originally was called as a query language and then how you know it also has constructs for specifying when to start this transaction, how to stop it.

And then setting certain parameters related to the transaction so how these transactions if the collection of transactions how should they collectively behave, what is there is something called isolation levels can be setup. So should this transaction should isolate from each other these are the various things that you can specify using SQL and then when data is there in relation database a program have to access the data and then make use of data.

And then also update the data. So how do programs access data so how is this programmatic access to data which is there in the database, how is that to be performed all these things are standardized and host of other things are also kind of standardized. So the advantage of a standard is basically that you get what is called interoperability that means if you setup your database and define it using standard SQL features.

Then in principle you will be able to operate with that you know from other applications develop in any language and if you want to and you develop it in one system and you want to port it in other system it should be easy to do that because you know the standard is followed. Okay so the database management RDBMS vendors so all the RDBMS management systems various RDBMS management system Oracle, Db2 Microsoft Sequel server, MySql so many RDBMS systems.

So all of them are required to support and implement this SQL standard okay that is when it can standards are for enforcing consensus, standards are essentially for enforcing consensus where all of them should agree that yes these are the features, these are the way we support the features etcetera and that is when interoperability will be possible okay. So of course due to certain commercial interest the RDBMS vendors may give some features which are over and above the SQL standard.

See SQL standard says that these are the SQL commands, these are the various features, these has to be implemented, but nobody prevents me from doing a little bit more right I can always add frills to my product and then make it more appeal so to say. So RDBMS vendors might will it is that to their commercial advantage to kind of actually give a little bit more

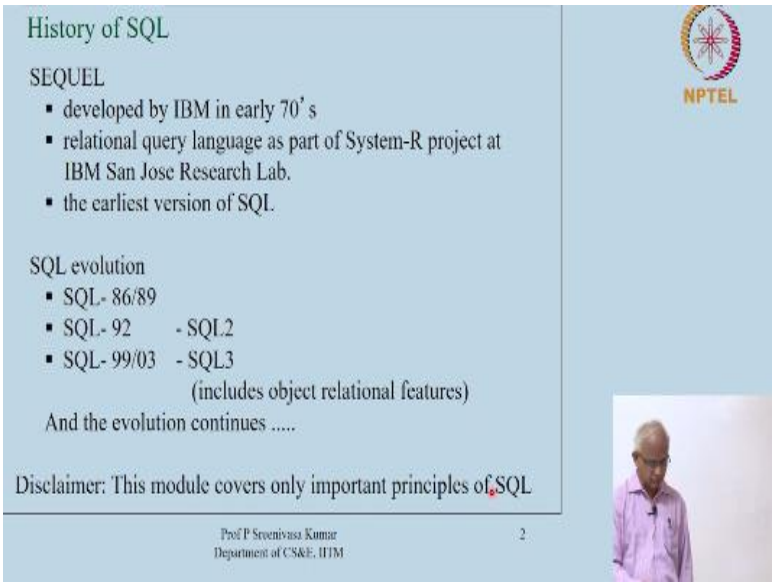
additional features so that you may get attracted to use those features, but of course the downside of using these vendor specific features is again portability.

So today you are using oracle and then you have some issue with the oracle service or the company or whatever and then now you want to move your entire application to another RDBMS system like Db2 or some other Db2 is supported by IBM like that then what you would expect is that your application you know it should be easy for you to put your application into the new RDBMS system that is what the meaning of interoperability is right.

So when both these vendors you know they implement SQL, but if over and top of that SQL if they have given some features and if you have used those features then you are likely to get difficulty putting your application from one vendor to another vendor, shifting your application from one specific system to another specific system. So the pragmatic advice here is that stick to the standard.

The vendors have they may their own reasons why they may give you additional features, but it is always advisable to check whether is this feature part of the SQL standard before you employ it in your application in the interest of portability of the application.

(Refer Slide Time: 08:08)



Okay so let us move on going a little bit of into the history of SQL if it is developed in by IBM in the early 70s and then it was actually implemented as part of the system or project at IBM San Jose Research Lab and that is actually the earliest version of SQL. Later on of

course it got standardized during 1986-89 one SQL standard version came then around 92 it was SQL 2 has been standardized.

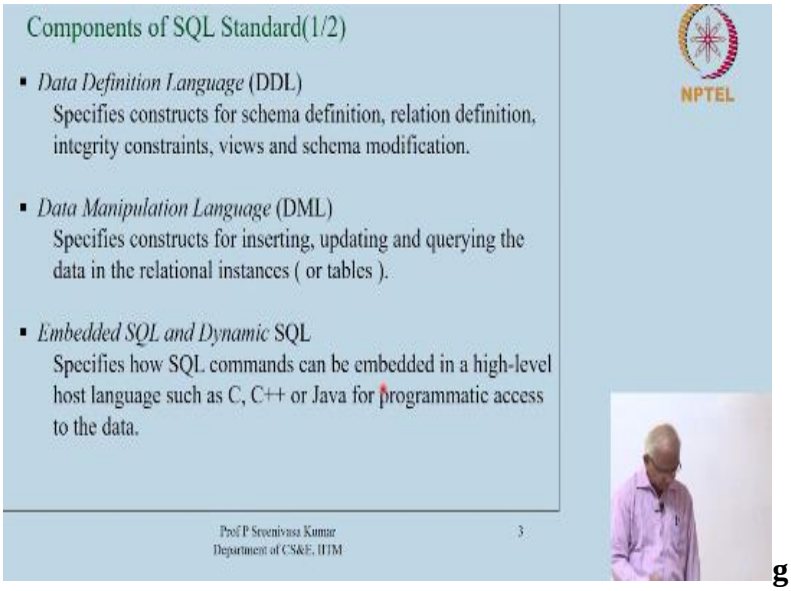
And around 1999 to 2003 there is new standard that came which is this SQL 3 that includes also object relational features and the kind of evolution continuous. It is as and when various development take place in the computing science general domain the SQL standard also kind of tries to evolve to cater to those additional needs. For example, when lot of developments for taking place in the XML world.

So RDBMS standards have reacted and then kind of included additional features as to how you can treat an entire XML chunk as a data item for example. Now you also have if you have to deal with large binary objects then how do you handle them multi as multimedia storage has increased how do we handle multimedia objects as part of relation database system. So many other you know additional details will you know need some standardization.

And these standards are continuously kind of evolving. So one has to keep up to date if you are working in the database world then it is kind of essential for you to keep looking as what is the development in the SQL standard and this module I just want to put a disclaimer here that this module covers important principles of SQL. It is not you know I am not going to give a complete syntactical and not even all the features of SQL, SQL is vast standard.

So we will touch upon the important features that are relevant for a introductory course like this course and then so I urge you to pick up lot more information about SQL from the SQL reference manuals.

(Refer Slide Time: 11:27)

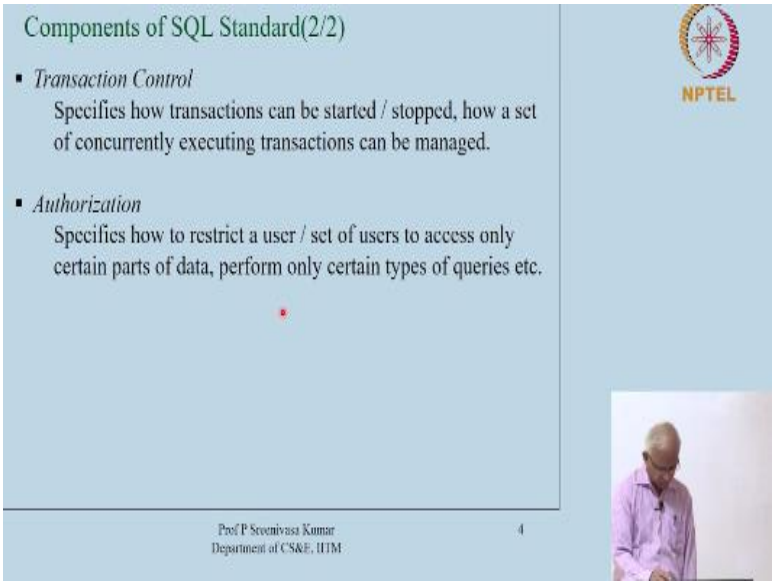


g

So various components of SQL standard okay we can yeah so there are various features that can be kind of grouped into some functional groups. So there is a data definitions language so called. So this basically has constructs for schema definition and the relation integrity constraints definitions and use and schema modification all these constructs that carry out these tasks are all can be thought of as the portion called data definition language.

And then how do we specify querying of data in relation instances and how do we insert data, how do we update data all these things is in the data manipulation sub language so to say and then what are the SQL commands or features that are required for programmatic access to this relation database systems. So from a high level host language like C, C++, Java programs how do we access data features too. So we will look at embedded SQL as part of this module and these are dynamic SQL and embedded SQL are involved in this process of setting up problematic access to databases.

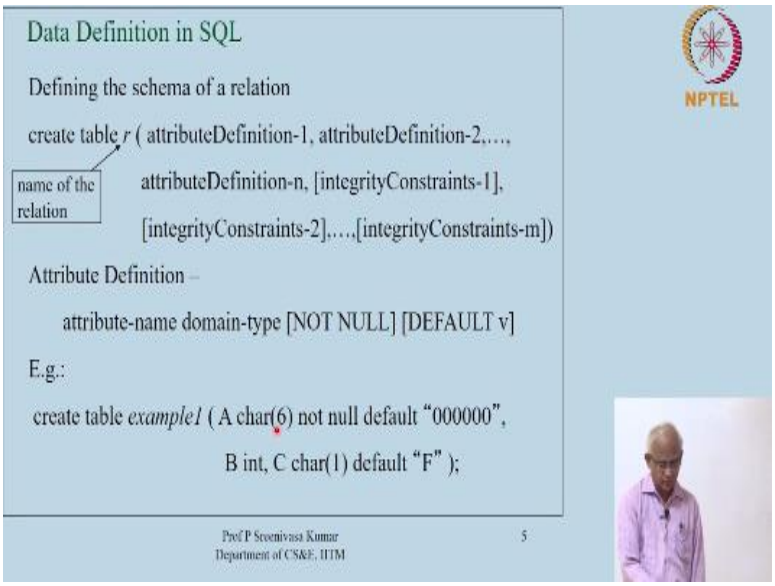
(Refer Slide Time: 13:16)



Okay now moving on we also have features for transaction control, commands for transaction control, how transactions have to started, stopped etcetera and one major module within SQL is the access control data access control. Data is actually sometimes data is very sensitive data you know you do not want the certain pieces of data to be seen by all people. You want to put some restrictions and who are the people who can look at the data, who can update the data.

Who are authorized to change the data etcetera these are access restrictions. So how do we specify these restrictions? So there are features in SQL in order to do that it is called authorization features. We will not actually study much of these things.

(Refer Slide Time: 14:21)



Okay let us start off with data definition. So far we have been saying there is a relation schema so how do we actually create a table. So it is the SQL command to do that. So here is

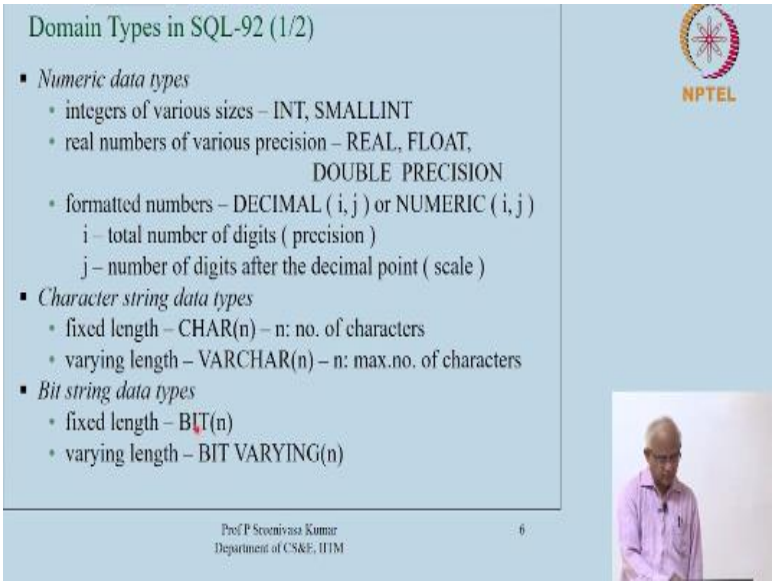
the create table command and then you give the name of the relation and then within braces you give various attribute definitions. So attribute definition, attribute definition some n number of attribute definitions followed by.

The square bracket indicate optional thing so this is integrity constraints various integrity constraints we can give these are also called the table constraints and we will go into the details about what are the integrity constraints we have already seen them actually these are called structural constraints right so how do we specify them etcetera. So this is the create table command.

And it has 2 things attribute definition and the integrity constraints. Attribute definition it basically has the name of the attribute or attribute name and the domain type followed by optionally followed by whether it is allowed to be null or not null. So if you give not null then it is never allowed to be null if you do not give this then it is allowed to be null and then you can also specify what is the default value for it.

So there is a you can by using this word default and then supplying a value in quotes you can give. For example, create table example 1 it has A and B and C these are the 3 attributes. So A is 6 character length value it is not allowed to be null and it is default is this all zeros and B is an integer and C is a single character whose default value is F.

(Refer Slide Time: 16:27)

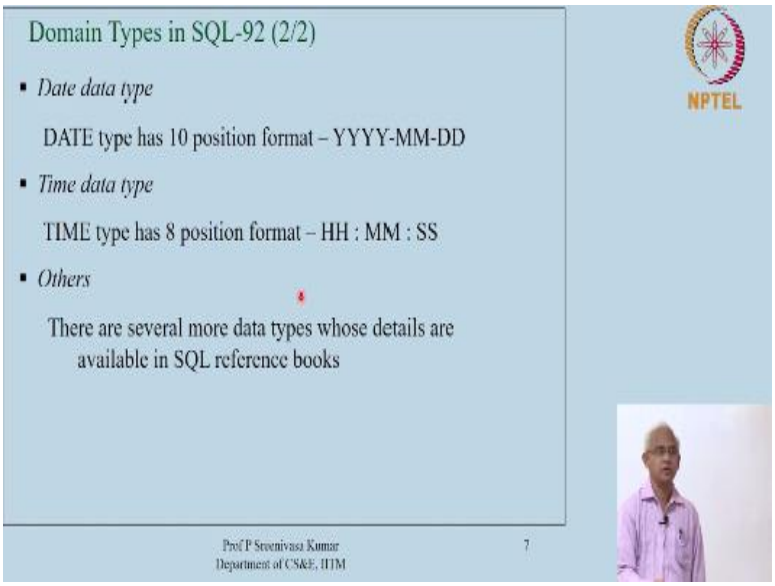


And there are lots of these domain types I will not go into full details about this, but most of the types that you are used to in programming languages have their kind of counterparts in

SQL data types. So there are numeric data types, integers of various sizes like integers small int, real numbers, real, float, double precision and then formatted numbers all of them are kind of available.

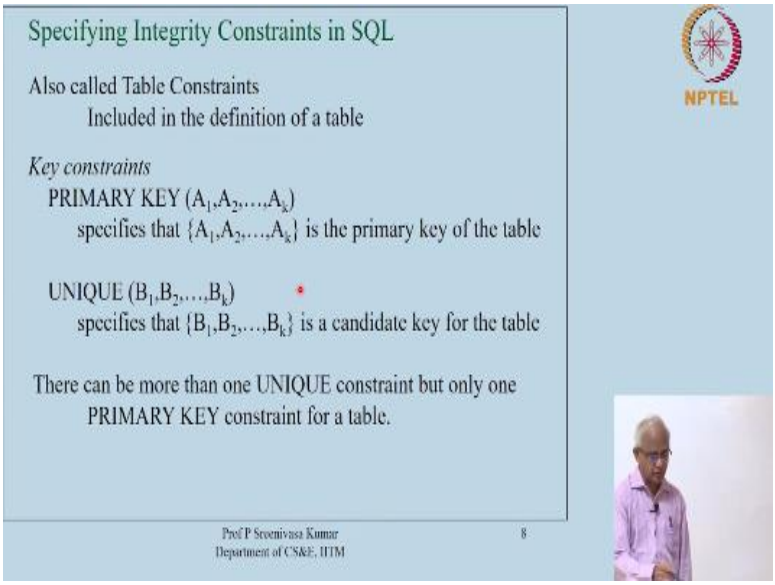
So and then character strings of 2 types the fixed length characteristics and the varying length characteristics. So varying length means VARCHAR so the value can have maximum of n characters you can specify that and if it is character n then it is fixed length it has exactly n characters and things like that and in the and there are bit string data types also and that means it is a string of bits and you have again 2 variations bit varying and just fixed length bit.

(Refer Slide Time: 17:41)



So and then date data type, time data type and lot of other data types are there please look up the SQL reference books when you come to developing the programs. Date is a 10 position format so year and then month and day time has hours, minutes and seconds etcetera. So, most of the data types that you are familiar within programming languages will be available here.

(Refer Slide Time: 18:19)

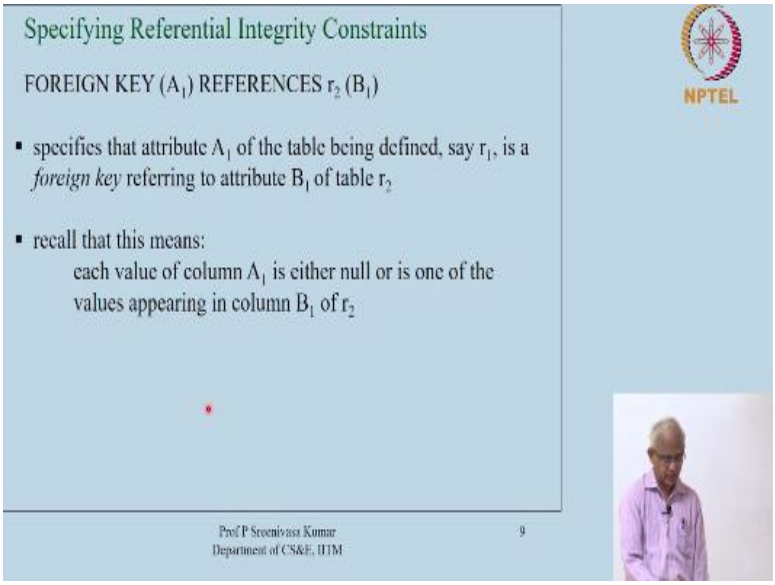


Now let us come to the integrity constraints part. These are also called the table constraints and what are the table constraints that we are familiar with. Domain constraints, key constraints and referential integrity constraints, so domain constraints are in some sense indirectly they are already specified here because you are giving the data type right for every attribute we are giving the data type, domain type okay.

So that specifies the domain constraints and then coming to the key constraints here is how you can specify the key constraints. So you use the word prime key word primary key and then give a list of attributes. So what it basically specifies here is that the set A1 through Ak is the primary key for the table and then a table can have a relation can have multiple keys and so you want to pick one of them and then say that it is a primary key.

And then the remaining things you can specify using this word called unique so this key word unique you can use and then say that so this B1 through Bk also is a key. So other candidates keys you will use this word unique to specify that these are also keys. So that kind of completely specifies as to what are all the keys that the relation has. So there can obviously be more than one unique constraint, but only one primary key constraint for a table right.

(Refer Slide Time: 20:01)

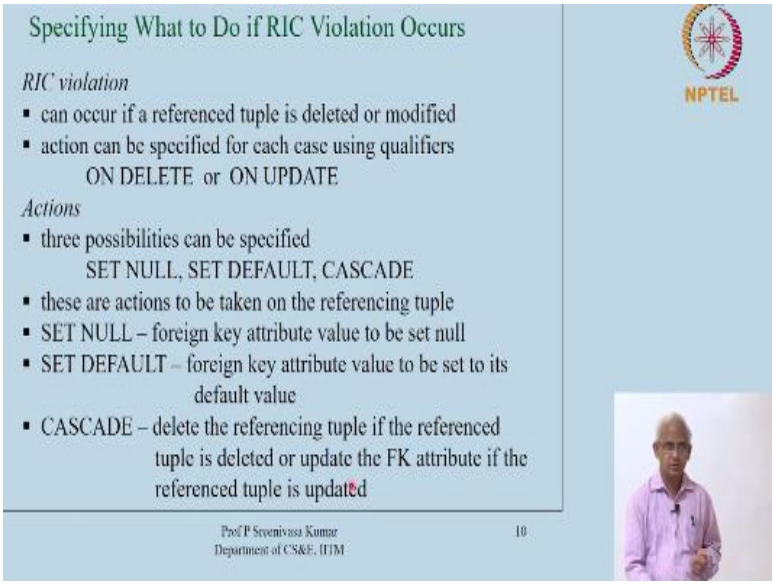


And then here is how you specify referential integrity constraints. So the key word here is to use foreign key and then specify what exactly is the keys sorry attribute or attributes that constitutes the foreign key and the key word references and the relation that it this that we refer to and what is the specific attribute in that relation that we refer to okay. So this phrase specifies that attribute A1 of the table that is being defined let us say r1 is a foreign key.

And it refers to attribute B1 of the table r2 okay. So we use the word table and relation interchangeably in this module onwards okay. So though there is some technical difference between the results of SQL queries we will see it later. Okay so what does this mean specifying this foreign key A1 references r2. You can recall that what this means is that each value of this column A1 in the relation r1 in the relation instance of r1 the relation instance r1 you know is either null is allowed to be null.

Or is one of the values that are appearing in the column B1 of r2 that is what referential integrity constraint is right recall that. So you could specify so this is how the foreign key gets specified.

(Refer Slide Time: 22:09)



So let us now look at the scenario as to what should happen if a referential integrity constraint gets actually violated okay. So we can actually specify while we are defining the table itself we can specify as to what are the actions that can be taken or should be taken actually if the referential integrity constraint violation occurs. Okay when does a violation occur? A tuple in one relation instance is trying to refer to refer to tuple in other relation okay.

A violation can occur if the refer tuple does not exist there okay. It should exist that is what our referential integrity constraint means supposing you are operating your database and there is a reference from one relation one tuple from one relation to a tuple in another relation and for some reason the second tuple the referenced tuple gets deleted. If it gets deleted, then there is a referential integrity constraint violation okay.

Or let us say it gets updated for some reason you know the actual column value B that this is being used for referencing has changed, the value has changed then also see while referring you are using department number 3 and for some reason in the department number 3 the department table 3 has got updated to 6 for some reason let us say. So then there is also a constraint violation because 3 got updated to something.

So this can occur the violation can occur if the reference tuple is either deleted or is modified under these 2 circumstances a violation can occur and what we will be doing as part of defining the relation is that what is the action that is to be taken on the relation that is being defined or the instance relation is data when such a thing happens okay. So there are 3 possibilities so these actions to be specified have these qualifiers.

Because only upon deletion or the modification the violation might occur so if occurs due to deletion or if it occurs due to update what should be the action. So there are 3 possibilities can actually be specified. So one of them is set null the other one is set default and the other one is cascade so let us look at those things. So, these actions that can be taken on the referencing tuple.

The tuple is referencing referring to null a tuple we will call it referencing tuple okay. So these actions will be taken on that referencing tuple. So what it says one of this first things is set null. Set null means you know this is the safest option like so if foreign key has certain value and due to update or delete that particular value is now not existing in the other table and you detected this.

If you detect, this the simplest thing that you can do is in the referencing tuple you can simply that value to null, simply convert the value to null. If you convert the value to null, then you are kind of breaking that link there is no more referencing at all there is no more referencing at all. So that means in some situations you may favor this. In other situations, you might say that set default.

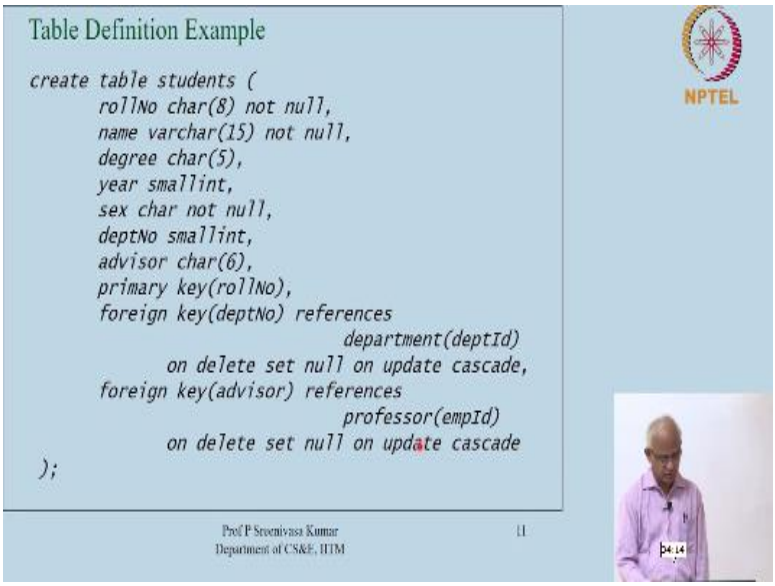
So the foreign key attribute value will be set to its default value. Let us say you know all employees who join are first by default assigned to some department and then later changed to other departments. So there is a default department for all employees let us say so there is a default value for the department ID or the section (()) (27:08) department ID. So you can choose to set up that.

So in case he loses the information about see the employee tuple for example is referencing to some department and then that department vanishes for some reason and so you can set this guide to work at the default department. Let us see later we will see how to handle the situation kind of right. So that is called set default I will take an example to illustrate all this 3 things.

Then there is a more one more option called cascade what this is it is a bit of a very drastic action what this says is that if the reference tuple is deleted then delete the referencing tuple

or update the foreign key attribute if the reference tuple is updated propagate that update okay. So this is the option. So I will show.

(Refer Slide Time: 28:08)



Let us talk about these things again in the context of an example let us create this student create table students. Now we have various attributes here roll number I would say fixed 8 characters' value which is not allowed to be null, name is a 15 character I am sure some of your names do not fit in 15 characters, but so 15 character up to 15 characters are not null, degree is 5 characters, year is small int, sex is 1 character not null.

And department number is small integer, advisor is a 6 character field then the primary key is roll number and then we have table constraints so these are all comma separated just you can see. So within the same thing there is no space separated and then comma separated. The foreign key department number okay so this department number is declared also as a foreign key and it references another relation called department and the attribute references is department ID okay.

So the student's department is a foreign key that references the department ID attribute in the department relation for further details about the department. So now here are the additional things that we specify in order to take care of the referential integrity constraint violations. What this says is on delete set null on delete set null and on update cascade okay. So what did he saying here is that supposing there is one (()) (30:19) whose department is 9.

And for some reason department 9 has been closed okay (()) (30:31). So what we do here is we set the students department as null value this fellow is now without any department okay and the other thing is on update cascade his department was actually 9 for some reason it got changed into 10 because somebody decided that will give department numbers according to alphabetical order of the department names or something like that happened.

So his department has now changed to department ID 10. So what this says is cascade it that means whoever is was in earlier department number 9 you update all those fellows department ID to 10 now which will solve the problem right. So that is what on update cascade that means basically propagate the change to the referencing tuples those tuples which are referencing this value.

Okay so this is how you specify the actions to be taken on this table the data of this table which is being defined when things happened on the tables that this table is referencing it is referring to some other tables and if something happens in those tables what should be done here okay that is what we are specifying here. So in case something happens in the department table what should be done for the department number attribute in the tuples that exist in the student relation that is what we are specifying.

Okay the other is foreign key other foreign key is advisor here. Advisor references a professor table and the specifically the employee ID attribute there and again same actions are found suitable here so for some reason somebody is advisor leaves the institute he may set it as null advisor as of now, somebody should figure out that this guy does not have an advisor.

And then Dean office will wake up Monday and then try to fix in advisor for this particular person or if the advisor for example changed from one department to another department okay and then his employee ID for some reason changed and then propagate that employee ID, propagate update cascade. So that is what we are specifying in as to. So these are very important and interesting actions that we can seek.

Because these are kind of structural constraints we have this ability to kind of tell as to what should happen when these constraints are violated and that is given as part of the definition of the relation itself the schema definition we set up this actions also okay. So that is the

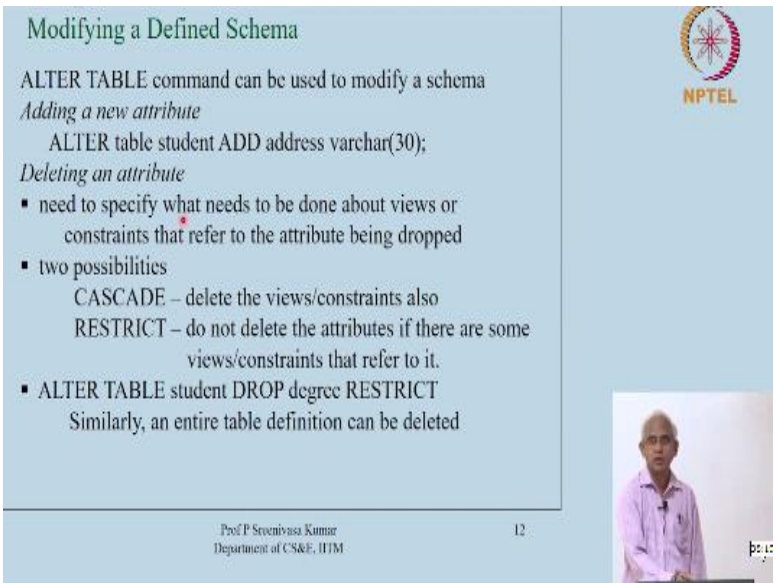
example so if you want to now go back to these things. So these are the 3 actions that can be so these of course these actions are specified for each of this foreign keys that are there independently.

For each of the foreign keys that are there in the relation that is being defined you have to set up all these referentially triggered actions okay any questions there. So you can set up on delete cascade it is a very drastic option on delete cascade. So your student is referring to some advisor when advisor leaves then you say that drop the student also it is a very drastic action.

So you should be careful while specifying on delete cascade that if the delete cascades to other tuples then we will lose lot of data. It might be appropriate in some cases for example let us say we are keeping track of the information about the dependents of an employee in a company and the employee leaves the company, if the employee leaves the company I do not have any incentive to keep track of the information about the dependents of that particular employee anymore in my company database.

And so I might use this option of on delete cascade while I am specifying the dependents while I am specifying the dependents in the dependents table while the dependent each dependent tuple refers to employee tuple and in case employee leaves drop the dependent also so that is an appropriate. Okay so these are the various actions that you can specify.

(Refer Slide Time: 36:37)



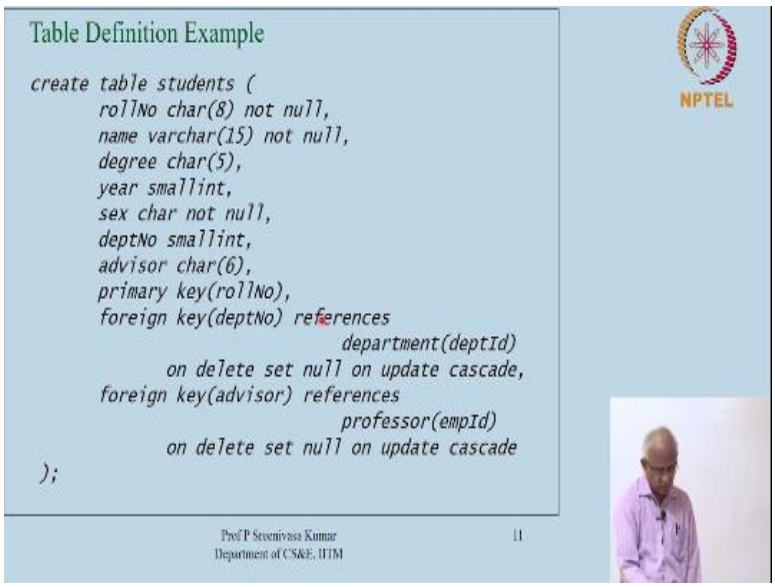
Now how do you modify a defined schema. So there is a command, there is a feature which allows you to kind of change the table. So alter table command can be used to kind of modify the schema by adding a new attribute. So far we did not have address in the students. So now we can say alter table student relation and add is a key word the address attributes with its data types.

Now you might wonder what should actually what happens to the data. So far we have some data and in which the address attribute was not there. So now we are trying to add this address attribute so it is a issue as to how to pick up address value for all the existing tuples and then etcetera. So schema modifications are pretty messy actually. So you should avoid them to the extent possible.

But in some cases that you may want to really record additional information which was list out while the design phase was going through you may want to do this. So there are comments for doing it, but it is going to be pretty messy to handle the (()) (38:04) during this schema modifications okay. So alter table add a new attribute and deleting an attribute is also possible.

You no longer think that it is appropriate to keep some information about certain entities. So if you delete an attribute you know what you need to also specify as to what should be done about the various views or constraints that refer to the attribute being dropped okay.

(Refer Slide Time: 38:41)



Now in though I did not talk about it in the table definition in the SQL standard it is also possible for you to kind of give a name for this constraint. See this is a table constraint right so you can give a numeric name for that particular constraint okay. So then you can later refer to that constraint and then say that drop that constraint okay. So when you are deleting an attribute it is possible that attribute is being used in some views or it is being used in some constraints etcetera.

And you need to kind of specify as to what should be done for those constraints or views. So again there are 2 possibilities. The cascade is a drastic option delete the views or the constraints also. In case you are dropping an attribute if there is a any constraint or a view that is referring to this particular attribute simply drop that also that is this option. This another option called restrict what it says is that even though the delete request has been made do not delete the attribute.

If there are some views or some constraints which are referring to it. They are using it some constraints and some views is using this attribute then go back to the person who has given that command and then say that this is being used somewhere and you still want to delete it, drop it. So this is safer option restrict then you will the system will prompt you back saying that you know it is being used somewhere and you really want to drop it.

So the command is alter table student drop degree restrict you want to some attribute I am dropping the important attribute of course similarly in an entire table definition can also be deleted and all that. I have anyway in some sense which the logical point where we can take up this discussion about the important very nice features SQL. So we will start doing it in the next class.