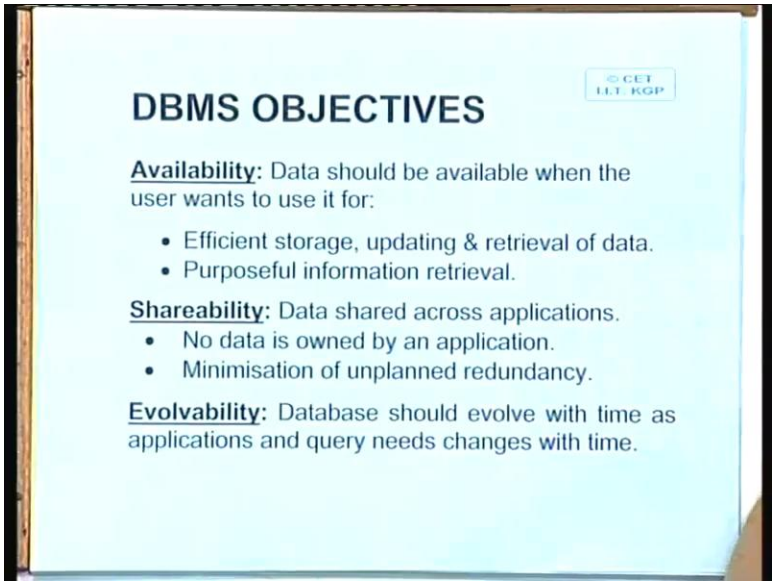
Management Information System
Prof. Biswajit Mahanty
Department of Industrial Engineering & Management
Indian Institute of Technology, Kharagpur

Lecture No. # 23
System Design – II
DBMS – II

(Refer Slide Time: 00:50)

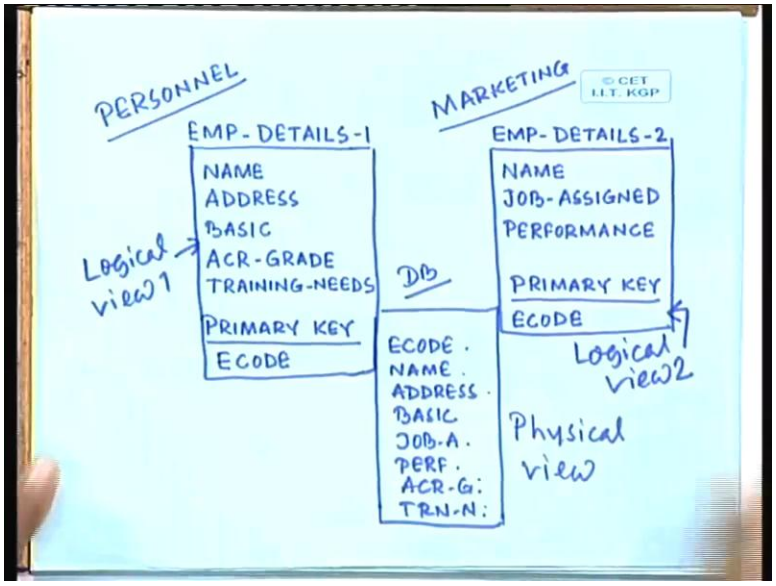
So in the previous class we were discussing about the objectives of database management systems.

(Refer Slide Time: 01:02)



Now you have seen that basically you have identified there are 5 broad objectives. What are them? The first one was data availability, shareability, evolvability, data integrity and data independence.

(Refer Slide Time: 01:20)

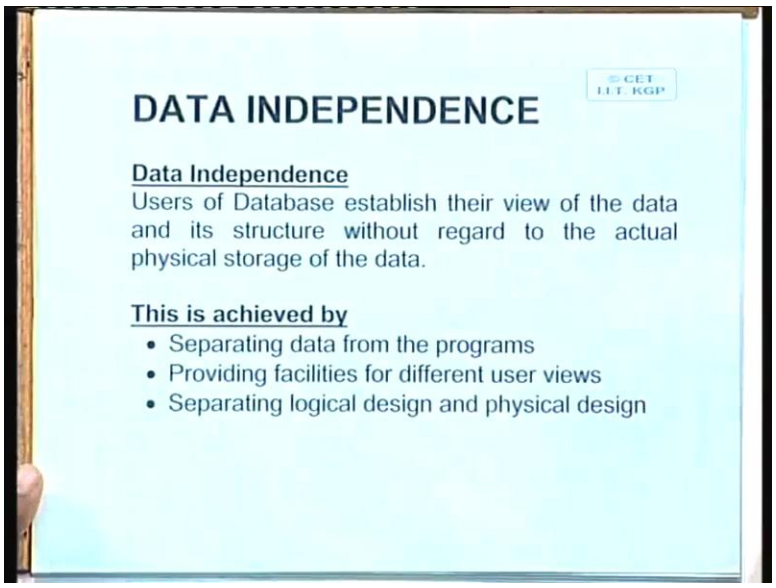


Now we have taken a simple example to explain the data availability and shareability how those things are actually possible. So the personnel department has got name, address, basic, ACR grade, training needs and marketing department also have name job assigned and performance of the employees. Now we could see that if we make a combined database by taking all the fields then the data is definitely shared and data is definitely available. But it still does not serve the purpose right. So why it does not serve the purpose because there are certain data like your ACR grade which the personnel department does not want to be available to others. Similarly the performance is not. You know the marketing is not interested to be disclosed to other departments. So what is possible that can we create that although the database has got all these fields that we may call its physical view? If we could create other logical views of the database that means the same database has got 2 logical views. If I look at it from this logical view definitely by entering the database through some password system right, then we shall have these many fields.

Whereas the same table database table if I view it from another logical view through another set of password system then it should appear this. In fact as I said earlier probably we can give something else. For example address may be address or even basic pay some other information which is not that difficult can be given. Now this is only access to the field. It could be even more than this. That is who has got the rights or permissions to upgrade or update a given data field. For example if an address changes who will make the address changes is it the personnel department or the marketing department. So what may happen we may give address and access to the marketing department, they can see the address. But they cannot change it this is probably possible for the personnel to do it all right.

So different levels of permissions can be granted to different users and although the tables are same the in the physical view, in logical view they may appear different all right. So this is also you know a type of data shareability you are sharing your data but you are still keeping your identity. In fact what is happening, suppose there is an address change the address change is done by only 1 place and everybody else gets that address fine. Now if you would have been keeping 2 different files what would have happened you would have kept 2 sets of addresses right. And any change in address in 1 file would not have changed it in the other files. So you can still have the old or wrong addresses in other file so that was the difficulty.

(Refer Slide Time: 04:55)



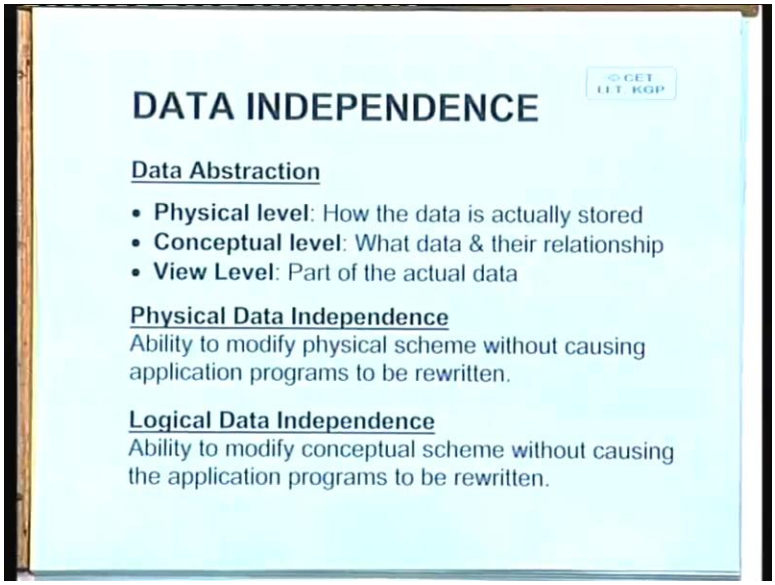
Now the very idea that we can create the logical views of the database is known as data independence. So although you are sharing your data but you are still having data independence right. So users of database establish their view of the data and its structure without regard to the actual physical storage of the data fine. So this is achieved by separating data from the programs and providing facilities for different user views separating logical design and the physical design right. So physically the data may be stored in any way it does not matter. But logically you can still design your data base from different points of view. Now this between data and programs that is a very big achievement.

In the old days particularly even if you look at the COBOL programming or even Basic or to some extent FORTRAN programming. You can see that you are defining your data is it not. Suppose you are writing a program in Basic or FORTRAN or even in COBOL the data that you are going to use you are defining the structure of that data clear. The difficulty that arises if the programmer is given the chance or choice to change the data then every programmer would define his own set of data someone may take name address etcetera, etcetera. In another program instead of calling it a name you may call it identity you see so many things.

The same thing you may give a different field structure instead of 25 you may like to give 30. So all kinds of things will come in if this is not there all right. So hum what is required if we could

take out the data from the programs the programmer has got nothing to do with data. The data is managed separately in the database all right. When you are writing programs it is the database administrator who looks after the data and its integrity the validity part of the data. Whereas the programmer only knows the data resides in a database. So whatever updates or things that is supposed to be done that is the responsibility of the programmer.

(Refer Slide Time: 07:27)



So on another angle, this is what we see data abstraction. We have the physical level how the data is actually stored at the conceptual level what data and their relationships at the view level part of the actual data. In fact see we were actually talking about the view level and the conceptual level. Whereas actually the physical level is beyond these codes all right. See there are many angles of DBMS you have your data. We shall mainly discuss how to define data structure. Basically you know that how to store data into the database and how to establish different logical views of the data base.

So we shall discuss at the conceptual level and the view level. But there is also a physical level of how the data is actually stored. See how the data is actually stored there are lots of things about the hardware. What kind of scheme you are using Oracle has got 1 kind of thing even other Sybase has got something else. So what DBMS you are using? So it is a different sort of subject.

So we are not discussing about the physical level. But what is important is the ability to modify the physical scheme without causing application programs to be rewritten all right.

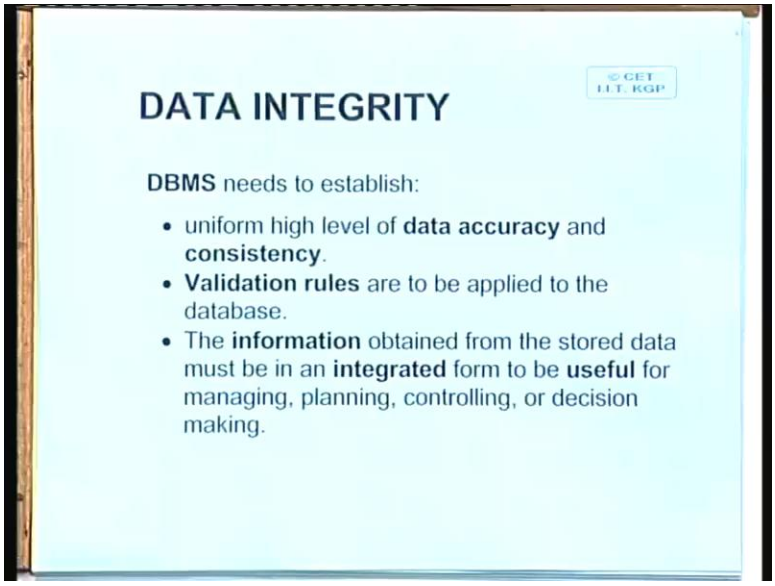
See if you are changing the conceptual scheme that is logical data independence. But suppose you decide that okay. There a new technology would be used right. Instead of oracle running under a different given platform, that is may be Sun Solaris a given version you want to run it in a Sun Solaris higher version. Now sun Solaris higher version is a different operating system all right it has got different way of looking at the Oracle implementation. So for that Oracle may require certain drivers those drivers may have to be reloaded. So what has happened there is a change in the database physical levels of data how it is stored. But these changes should not be in any way effect the change in applications programs the application programs should be still running all right.

In fact earlier what used to happen suppose you have a database right? Now you are making some changes to the database what kind of changes. That is you are adding a new table that means a new file right you have not considered earlier you have employee table and all those things. But a new kind of deduction has come that club membership is being given through salary. So these club memberships, a new file you are creating. So you have to make a change to the database right. Now what used to happen since it is a new change in the database all the application programs are to be recompiled is it not. But this is what should not be there that means the database was not having logical data independence. You may be changing to the table. But as long as the application program is program is not affected. If it is affected definitely something there will be something wrong.

For example if you are using employee data and this program is actually updating address field and you have dropped the address field from the database. Then definitely the application program will be showing errors that such a field is not existing, you have to recompile it. But that is necessitated because there is a definite change with regard to that program. But many a time it is not like that and a database may have many application programs right. You have an employee database which may be used by somewhere around 20 to 30 or even more application programs. Now when you make a database change it may affect only a few of them. So why should I have

to recompile all the programs right. So that is not required in the set of logical data independence.

(Refer Slide Time: 12:01)



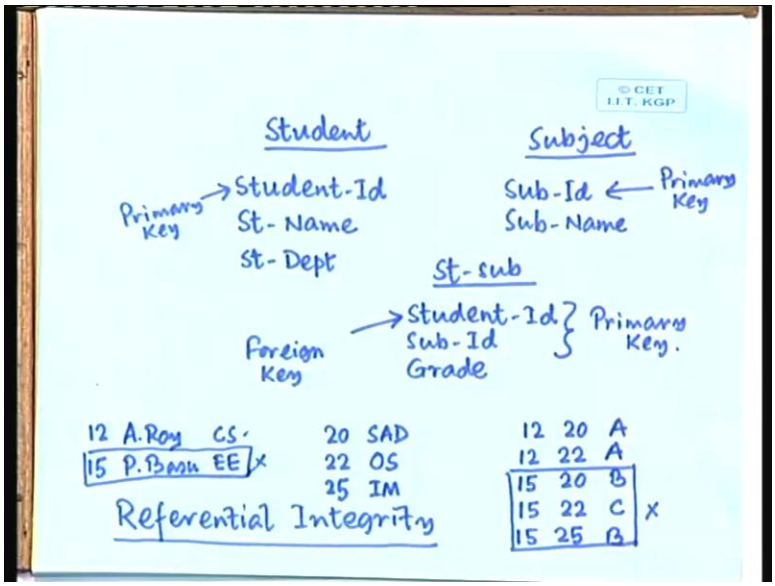
On the other hand the concept of data integrity the basic idea is that see whenever you are keeping the data in a database you have to see that the data is accurate and the data is consistent. How to do this by different kind of integrity rules data integration the rules of data integration. Say for example as I said about the range check for example your roll numbers your roll numbers have to begin with the year when you have joined. Now if the year has to be somewhere near about 2003, we cannot say that you have joined in 1990 right. You know student who has joined in 1990 should be here now. So 90 is an invalid set. So it is said that every data has got a domain value a domain value. If the data is not within that domain value that means there is something wrong about this, this can be done.

Then other kinds of integrity checks are possible. Say for example range of values. See whenever you are given a data item it may have a range of values all right. For example salary right. You can specify what could be the salary of your employee may be you, may say the minimum salary 5000 rupees, maximum salary 30000 rupees. Suppose that means anybody whose basic pay is above these 30000 or below 5000 is definitely a wrong entry. Now you can take it further. You may decide different scale codes. Say all your attendance is in 1 scale right. All assistant

professors in 1 scale all professors in another scale. So attendance salary will be within a certain range it would be not so high as 5000 to 30000, it will be may be in a smaller range all right. So you can specify, so if the scale code is attendant and the salary is 10000. Although 10000 is a valid salary all right.

But it will be wrong it will be wrong because may be the salary range of attendant if probably is 5000 to 8000. Suppose arbitrarily right so 10000 will be a wrong entry. So this is another kind of integrity. There is a very important type of integrity which is known as referential integrity. I will discuss referential integrity in detail some time later. But as on today let me just tell you this much. Say sometime what happens that you will have different files or different tables and there will be different sort of information right. Say for example a student. A student takes certain subjects all right. Now he has got some grade. Now see the grade is not an attribute which purely depend on the student. It also depends on the subject. Because the student has got many grades in different subjects all right. So you have to define another table I will discuss these in detail later that is student subject student id subject id and grade. So student id subject id grade this will be 1 table. And there is another table let me just put it so that you can understand what I am trying to say.

(Refer Slide Time: 17:12)



Now see this is a structure of the table in the relational design. I will discuss as I said later. But at this point let us understand what exactly I am saying. Student subject and this you may call as a st-sub this is a relational design. Now every table must have a primary key must have a primary key. What is the need for a primary key so that we can uniquely identify every record. Now the primary key should be not only unique it should be precise concise comprehensible. So there is lot of property of a key. You cannot say that student name is a key because student name is usually not precise is rather large. So there will be chances of making errors. So you see this is a primary key here.

Similarly we have this is our primary key so what will be the primary key here. Student and subject together because both determines grade. So this is our primary key. However individually this student id usually is also called a foreign key. So you can see that subject id is also a foreign key why it is a foreign key. Because this is a primary key of a separate table see the student id is a primary key of another table. So we say that this is a foreign key fine clear. Now what happens? say I have some records I have some records of let us say A Roy. Suppose see these are some records these are some actual records which are available. So we have the records of A Roy and P Basu 12 is employee student id. Now this is about the subject details and this is about some grade details all right.

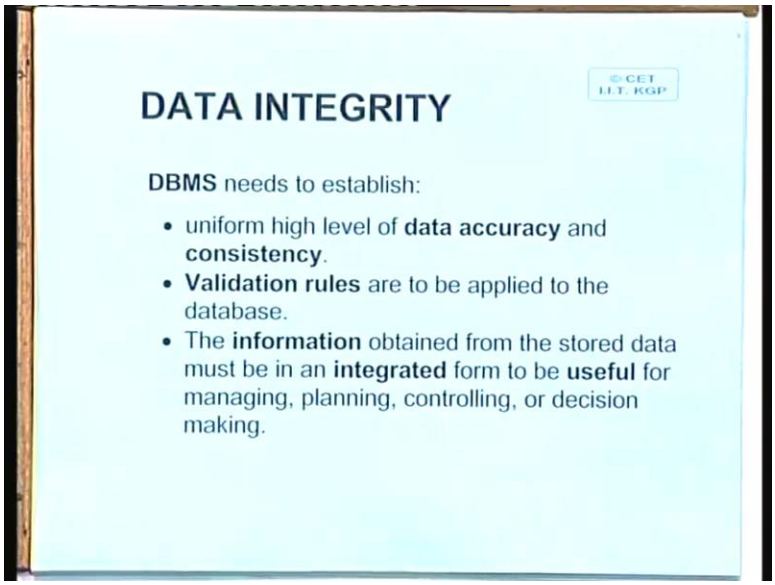
Now what is happening? Suppose due to whatever reason this P Basu has left the course. P Basu has left the course. Then if you simply delete this record from the data base because mind you, this is an active data base. Any record here means he is a current student. If you want to archive his record that is a separate issue. But we have to delete it from the current record. So what is happening this we deleted. This does not exist. But even then his records will remain in this file can you see that. So this basically means that if we want to completely delete the records of P Basu not only we should delete it where it is a primary key but also should delete it where it is a foreign key fine. So we must also delete all these records. It is not that we can delete only record which is in the primary key right. So this is known as referential integrity.

I will just write it referential integrity that means that if a particular record is deleted on the basis of its primary key with that record should also be deleted also be deleted in the where it is foreign key. (( )) (23:20). Yes that is another way see not only delete it could be modify. Suppose

see that is another issue that we will come up later. Say we due to whatever reason P Basu's 15 roll number has to be changed all right. We want to give him a different roll number may be he has changed his branch. So his roll number may change to something else say 35.

So not only changing it here will be enough, it has to be also changed everywhere where it is a foreign key. But fortunately if he changes his name tomorrow he calls himself instead of P Basu some other Basu, may be B Basu then we need not change it everywhere. If we have to change it in many other places then it is not a true relational design. But these are the aspects which we shall discuss later fine. So this is about referential integrity there are many ways the referential integrity can be established.

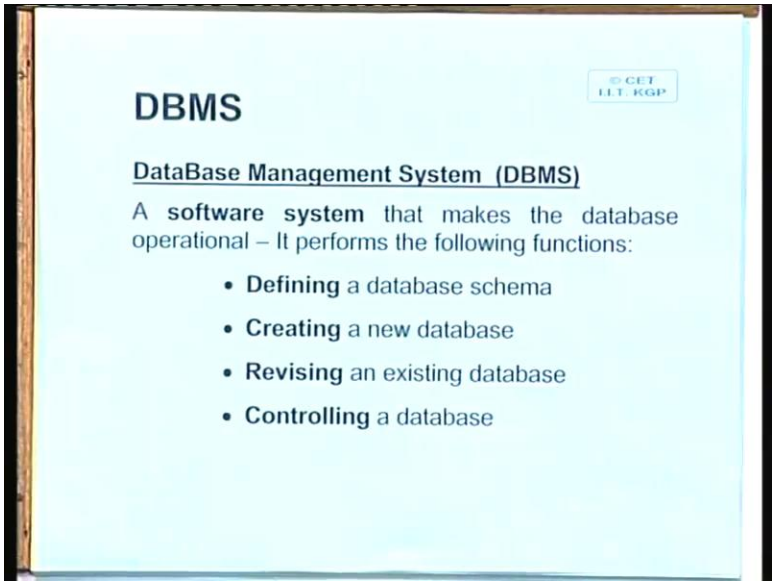
(Refer Slide Time: 24:32)



So there are various those kind of validation rules and integration rules which has to be available so that the DBMS has got and accuracy and consistency. The information obtained from the stored data must be in an integrated form to be useful for managing planning controlling or decision making. In fact the relational design is important to have integrity like I was saying that will come back to this slide see no not this. See what I was trying to say is that look at this P Basu. If this P Basu name also occurs in other tables or in other records right.

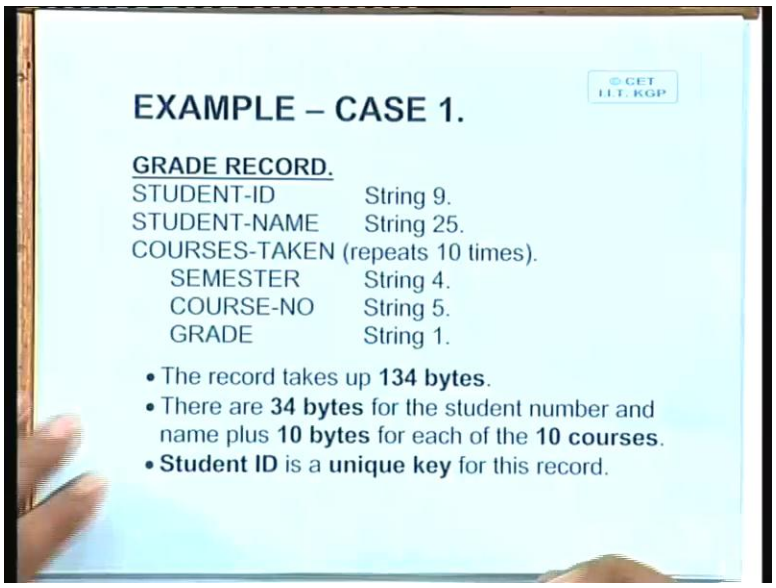
That means what? That means any change in this record has to be changed in all the other records. If it is not changed then we have what is known an anomaly known as anomaly. So we should see that these anomaly is minimized and this is 1 of the principle using which we must design our databases. We will come back to these discussions later. But before that let us try to see certain other aspects or certain other things about DBMS.

(Refer Slide Time: 26:19)



Now a data base management system is a software system that makes a data base operational, I have told you that earlier it performs the following functions. What it does? Defining a data base schema creating a new data base revising an existing database and controlling a data base. You see defining creating revising controlling defining revising we may say modifying, restructuring. Restructuring is different. Basically whenever we are talking about database we must be able to differentiate between its schema and its data there are 2 different portions about a data base. What is a schema? Schema is a definition of the database. For example what are the tables? What are the tables? Each table what I may have a little bit not much but at least something.

(Refer Slide Time: 27:39)



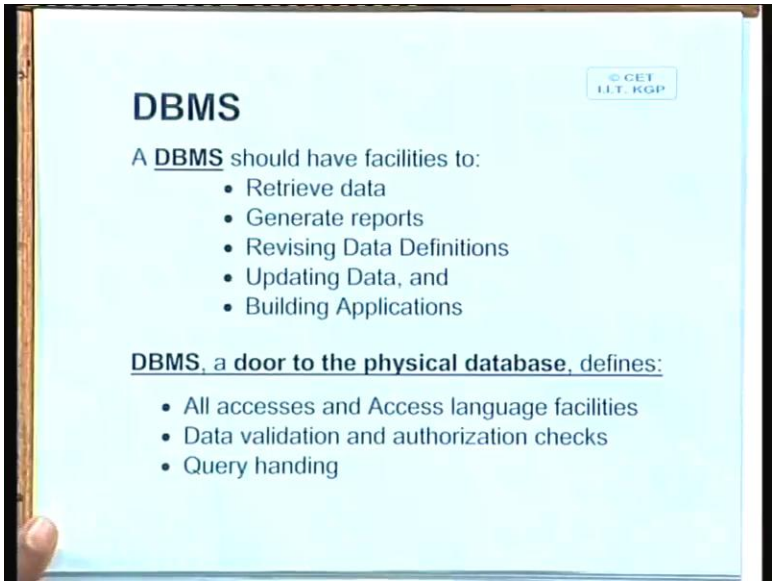
Say this one. Take this small example we have. Let us say suppose we define like this suppose it cannot be done. But that is another issue. Suppose we define student id string 9 student name string 25 courses taken, repeat 10 times. It cannot be done semester string 4 course numbers string 5 grades string 1 this this is not possible. But nevertheless suppose we define right. These kinds of definitions are known as schema I will come back to this example later right. This is called a schema all right. So whereas the actual data that is stored actual data that is stored is called the actual the data part.

So in this example you can see that these are like schema these are student schema. Everything is the database schema. This is the definition of student table subject table sub table and these are the actual data that is stored in the database all right. So any change in the schema when you create that schema first time that is known as defining the database. Then using the schema when you create the empty tables without any data in them is known as creating a database all right. Now any changes that you make in these keep on updating data putting in new data which is happening all the time in any running database is known as modifying a database.

And finally if you have to make any change in the schema you make a change in the schema and accordingly redefine the database that is known as redefining the database or restructuring the database. I think you have understood the difference between modifying and creating and

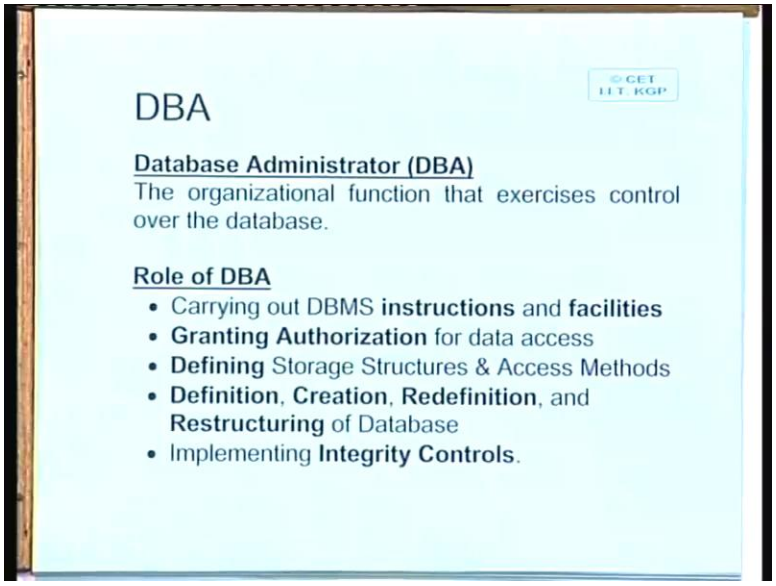
restructuring and redefining right. So anything to do with schema is defining creating redefining restructuring anything to do with data is modifying add delete modify. So these are some of and obviously we have to do controlling the database we should see that it is not wrongly accessed and proper permissions are available for the access of a database.

(Refer Slide Time: 30:10)



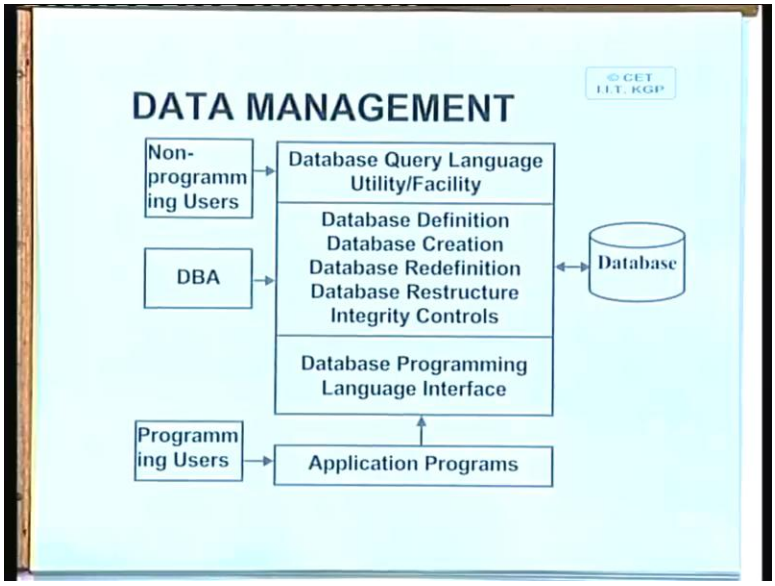
So these are the different facilities that the DBMS should have. It should retrieve data if some data is there. It should be possible to get it generate reports right. Revising data definitions data definitions data definition means schema revising schema updating data nothing to do with definitions and building applications programs writing programs. So DBMS is called a door to the physical database. Why because, it defines all accesses and access language facilities data validation and authorization checks query handling. So you cannot bypass DBMS if you want to use database you have to go through the database. So it is called a door to the physical database right? Is it clear? That means it is like you know all accesses and access languages facilities etcetera. That is I will come back to this do not worry. There is another slide you can note from there.

(Refer Slide Time: 31:21)



That on the other hand the database administrator I have briefly told it is an organizational function. It is a person actually who exercises control over the data base. So what exactly the DBA does carrying out DBMS instructions and facilities granting authorization for data access okay. So suppose you want a logical view right DBMS has the facilities but someone has to do it. So you give your request to the database administrator and the database administrator will actually grant you the authorization. Defining storage structure and access methods. Definition creation redefinition and restructuring of database right. So definition means just defining the schema writing down in a DDL data definition language creation compiling the DDL and actually you using it. Redefinition redefining the data definition language DDL and rewriting the schema and restructuring recreating the database. Implementing integrity controls. All the integration like referential integrity, then integrity of the domain values etcetera, etcetera. All of them are the role of the database administrator to implement okay.

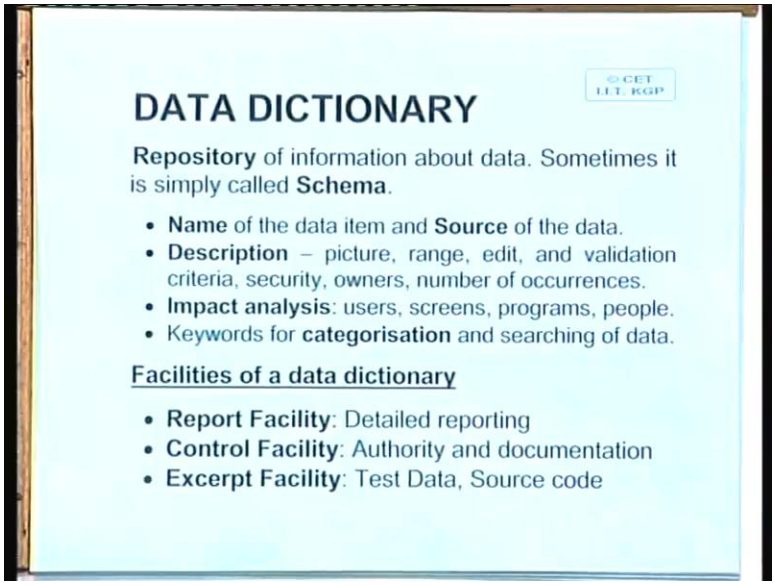
(Refer Slide Time: 32:53)



So all these things I have put it in the form of a diagram you can see that. This is the part of the DBMS the DBMS actually you know accesses the database. Now what are the things it has to do the database definitions, creation, redefinition, restructure, integrity controls, all this is you know done by the DBA database administrator all right. Now users again are of 2 types. 1 is the programming users the programming users will interface the DBMS through a database programming language interface. They will write the application programs and these application programs through this interface will interact with the database fine. The non-programming users will have to do in 2 ways. What are the 2 ways? 1 is through this application program it is not separately shown.

So you can write run these application programs and the non-programming users uses this application programs and through the application programming interface it interacts to the database. Or they can directly use the database through query language utility or facility right. Database query languages like structured query languages SQL fine. So you can 2 ways you can basically talk to a database through an application program or through database query languages for the users. But the DBA can actually use DDL data definition language and DML data manipulation language in a sense SQL is also a DML right is a data manipulation language.

(Refer Slide Time: 35:03)



Again that data dictionary it is the same thing I have already discussed earlier. There should like your data dictionary in your data flow diagrams. You should also have a data dictionary is a repository of information about data it is simply called a schema sometimes. So you have to have the name of data item source of the data. Then description. What are the descriptions? It should have a picture. What is a picture? Picture does not mean a picture or what is a picture.

Student: They will have outline.

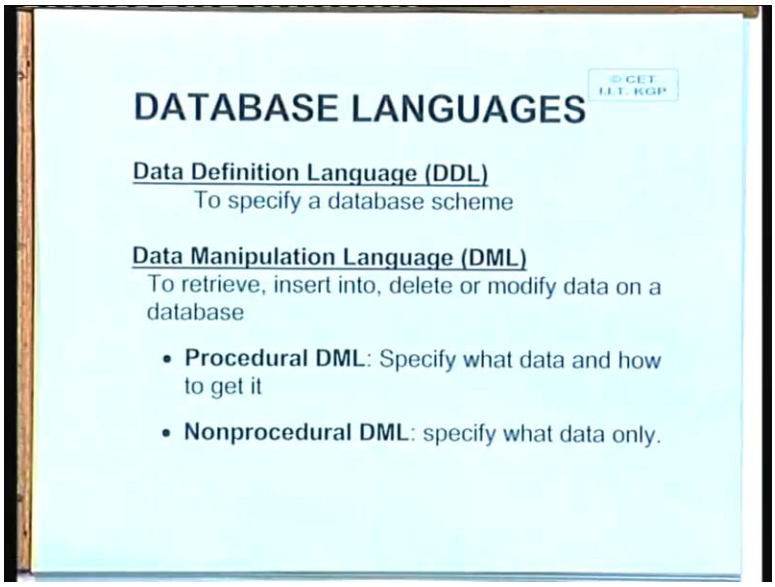
Outline of what? What is that? Suppose your name what will be its picture?

Student: String of length of 25.

Yeah. So you will say that the name is string 25. That means, that is the picture of the name right name string 25. That means it is a going to be a string and it is a length 25. That is called a picture. The range what is the range of values what are the possible values I have discussed. Then edit and validation criteria if there are some specific edit or validation criteria available that we can put it here. Security who is the owner what are the number of occurrences how frequently this data item is incurred. Then impact analysis the user's impact of user's screens programs people how often this is being used.

Then keywords for categorization and searching of data. Then if you needs to be categorized what are the keywords through which it can be done and searching of data. Then through from it usually a data dictionary provides report facility control facility and excerpt facility. That means you can excerpt out of the data dictionary the test data or the source code. Source code mind you which is like a program source is code is nothing but a program. What is the use of test data when your application program is actually developed you have to develop a test data to test the validity of the programs is a part of testing.

(Refer Slide Time: 37:30)

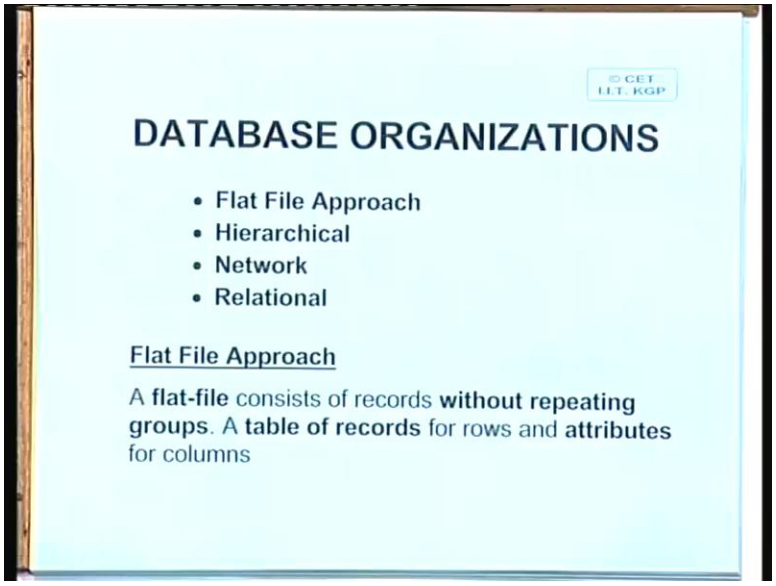


Now there are different database languages. For example the DDL. I was talking about DDL and DML data manipulation language. Data definition usually you see DDL, DML whatever you may call finally it is only 1 language. For example SQL. SQL has the facility of both being it a DDL as well as DML right. So using SQL you can write the data definition you can also write the codes for data manipulation so both are possible right. Usually there are some 4 GL's that is fourth generation languages which are little bigger than the SQL right. So SQL is embedded some SQL is a standard thing using SQL. You can do lot of things. But not everything not everything says like using SQL.

What you can do is you can define many simplified things you know very simplified kind of reports are possible. But if you want a very organized report with bold something colorful this that then you have to have something else those can be called 4 GL's. That is certain extra facilities which are also embedded along with the SQL. So in many situations there is no need to write any application programs. In fact SQL's are also sometimes very, very powerful. Very, very powerful. Will see to some extent what are the advantages of SQL. So again DML is of 2 types procedural and non-procedural.

See what is the difference between procedural and non-procedural. Procedural language like Fortran basic C to some extent C plus plus and java because object oriented programs are not so much procedural right. But not strictly non-procedural as such say for example SQL structured query language is a non-procedural DML. Because what is happening the output that you are getting you are not doing it in a step by step process fine. So that is the difference between procedural and non-procedural. These will understand when we discuss SQL.

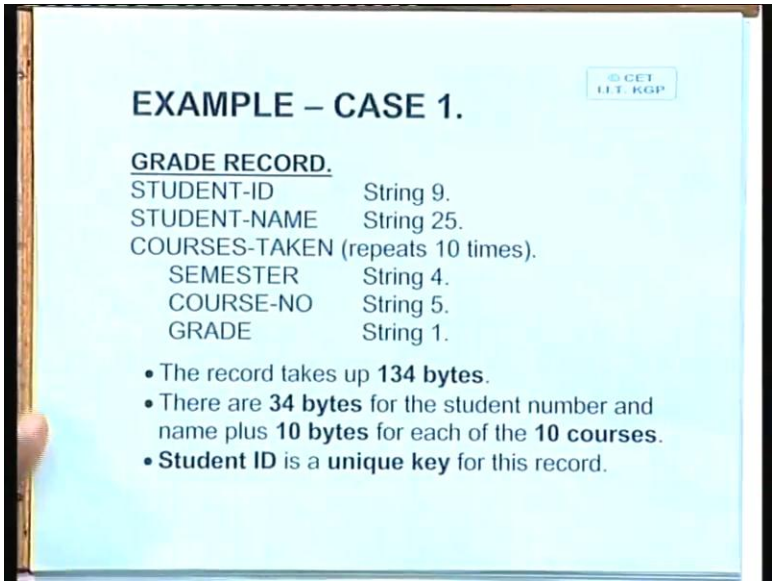
(Refer Slide Time: 40:03)



Now today let us see, the another concept that is known as the database organization. Now obviously this discussion may be of not much importance because today the hierarchical and network database organizations are not that popular anymore right. So we are basically having relational database organizations everywhere. But nevertheless we should understand the others

also right. The first 1 is a flat file approach a flat file approach is no approach. Actually it is consist of records without repeating groups a table of records for rows and attributes for columns. Now what is a repeating group I will I will explain repeating groups.

(Refer Slide Time: 40:53)



See this example you have seen that we are saying the course taken repeats 10 times. That means the semester course number and grade these are array elements these are array elements. That means they have some dimension all right. So semester 1 you can refer to semester n, semester 0, semester 1 semester 2 semester 3 so 10 time means up to 10 semester data can be stored. And we are giving the course number and grade for each of them. For each of them can you tell me what is the difficulty if you are have a repeating group? What is the difficulty if there are repeating groups?

Student: It might be difficult to modify the database.

Modify, modify what? Say course number.

You see the first and foremost thing is there is lot of data redundancy what is the data redundancy. You see here look at this. This record is what is the primary key is the student id but not just student id you must also have to take the semester and course number. Now what

happens for every student for every student we have to keep repeating the semester and course numbers. Suppose every student has got 10 semesters and in 10 semesters he takes nearly 50 subjects. For every student we have to repeat the 50 course numbers all right. Some students may not have taken all these courses but I have to keep provision all right. So for every student the semester and course number has to be repeated whereas grade is his own all right. This is 1 problem.

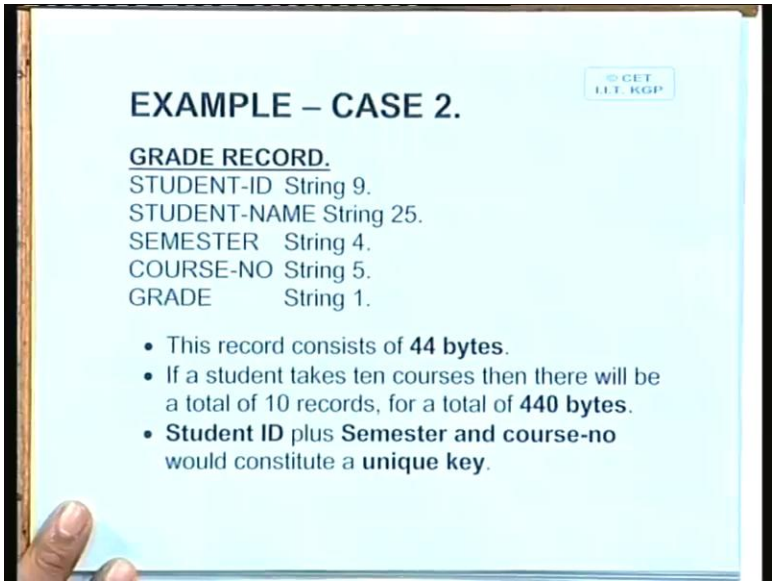
But more importantly much more than this problem the problem is that suppose due to whatever reason the number of semester change suppose instead of 10, now we have 12 semesters and not all students are going to have 12 semesters all right. See actually not 10 in a 4 year course there are 8 semesters and some dual degree students and MSE and architecture students have got 10 semesters okay. So what is happening that usually we have 8 semesters for most of the B Tech students but for some students we have 10. So for those students some records will not be filled up, so there will be vacant spaces some kind of redundancy right. And secondly and much more importantly if we have to locate a particular grade of a student for a particular course number what we have to do?

We have to locate a student we have to locate a student and then search for the course number all the way. Suppose I need a report just try to understand this that who are the students who have got b grade in a particular course. How to answer this query very difficult very difficult. What you have to do you have to go for every student because there is no way to know who has taken this course all right. So what you have to do you have to go to the first student, see whether he has taken that course and if he has got a b grade their if yes note it down, then go to the second student. If there are very large numbers of students then we have to go to the all the students and the query will take a lot of time all right.

So these are the major difficulty that any kind of query other than the normal channel suppose I say that a given student what are the subjects he has taken what are his grades or print out his grade card very easy very simple. All I have to do go to that student record and print everything very simple. For those kinds of standard reports repeating groups may be fine all right. Whereas if the report is not standard it is a non-standard kind of a report, it is a horrible in fact it should be

avoided. So because of this a much better approach is this where we have the student id, student name, semester, course number, grade all right.

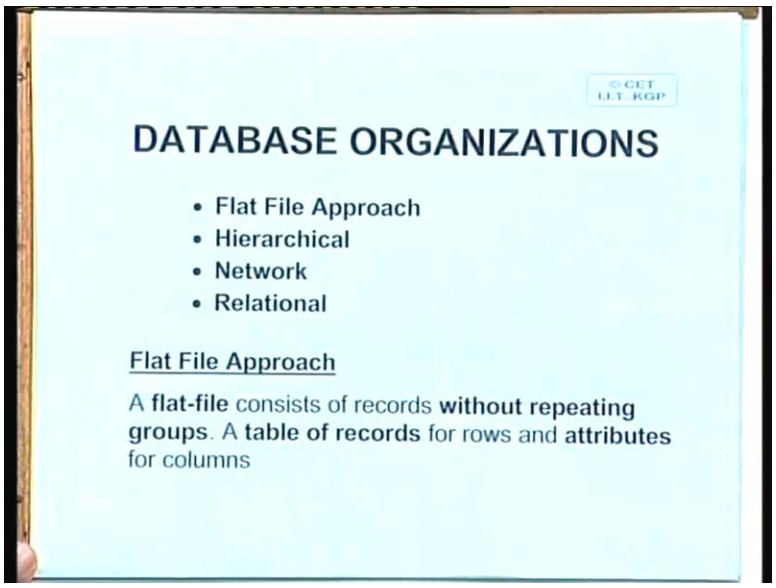
(Refer Slide Time: 45:56)



So what is happening here, we have a student id student name semester course number see what is this key? Key is student id semester course number together. In fact we may keep student number later fine. So what we do look at this here a little bit of redundancy has been removed in the sense that no repeating groups and here I can sort the file in many different ways. So I can sort it course number and grade wise. So if I sort it course number and grade wise all I need to do is find the particular course for which I know who have got b grade. And from the course within the course again it is sorted grade wise. So just pick up those records where we have b grades and print the student id student names.

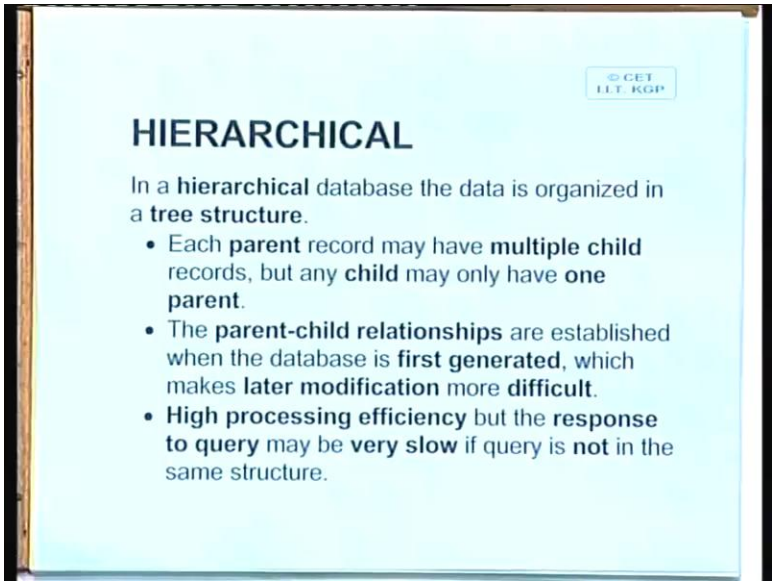
So these are the students who have got b grade in this course right. Suppose I have a particular query who are the students I mean in a given semester, in a given semester who are the students who have got say f grades. What we need to do I have to sort it semester and grade wise. Instead of course and grade wise semester and grade wise. So then what will happen I find the given semester find the given grade pick up the students. You can see that it becomes much easier. So that is why the flat file approach is considered better than the repeating groups or so called design where we have the arrays.

(Refer Slide Time: 47:40)



We are so interested in arrays particularly in your database systems so it should not be done right. A flat file consists of record without repeating groups a table of records for rows and attributes for columns.

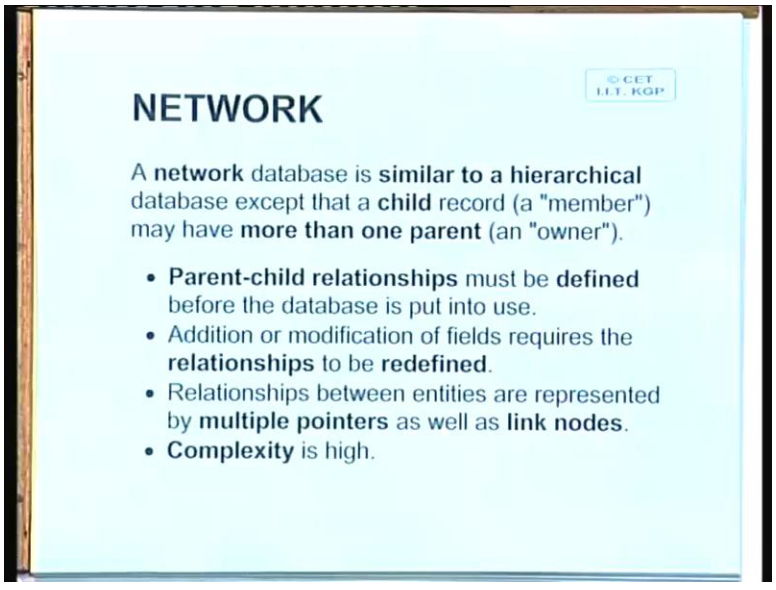
(Refer Slide Time: 47:56)



So other kind of design could be a hierarchical design what happens in a hierarchical database design. The data is organized in a tree structure each parent record may have multiple child records. But any child may only have 1 parent the parent child relationships are established when the database is first generated which makes later modification more difficult so high processing efficiency. But the response to query may be very slow if query is not in the same structure. See again what is happening in hierarchical structure. It is like the, suppose we make the subjects subject records as hierarchically below the student records so what happens. Any query which originates from student who are the student who have taken these subjects easy to answer I am.

Sorry this student what are the subjects that he has taken easy to answer? But any query that is subject wise that is from below will be difficult to answer. So what is the tree structure we have made student on top subject on bottom or below all right? So any query which is from low to up or bottom to top will be difficult or any query which is from top to bottom will be easy right. So the good point about hierarchical structure is that any structure where I have. Let us say pre-defined queries that sort of a thing will be easy. Pre-defined queries will be easy to answer but any query which is not pre-defined will be difficult.

(Refer Slide Time: 49:55)



So our last slide today that is the network database structure the network database structure is similar to an hierarchical database except that a child record or a member may have more than 1 parent. So it is it is like a mix structure. So what is happening we have the student records in 1 side subjects records on 1 side. So whenever a student takes certain subjects, we create access links to those subjects is it clear. Similarly from subject point of view all the students of a particular subject they are connected through some access paths. Now you may say where do I keep the grade. So to keep the grade we defined something else called the link nodes. So it is like a student records the subject records and the link nodes. Is it clear? So the link nodes will contain the grades. So one student is connected to the link nodes and the link node is connected to the subject all right and the grade is put on the link node.

But the point is suppose the student has dropped the subject, suppose a student has taken the subject then we have to define first a link node, then we have to provide a pointer or access path through which student and subject is connected. See all these are being done physically in the database you have to define a pointer and this pointer will connect student to the link node link node to the subject when grade comes grade has to be put in the link node. So once it is done, query can be very fast. But all add delete modify will be difficult right it will take lot of time lot of time. So we will discuss further I will stop here today and will bring from here relational design and we will see what are the advantages of relational design and how to carry out relational design. Thank you.