

Geographic Information Systems
Prof. Bharth H Aithal
Ranbir and Chitra Gupta School of Infrastructure Design and Management
Indian Institute of Technology - Kharagpur

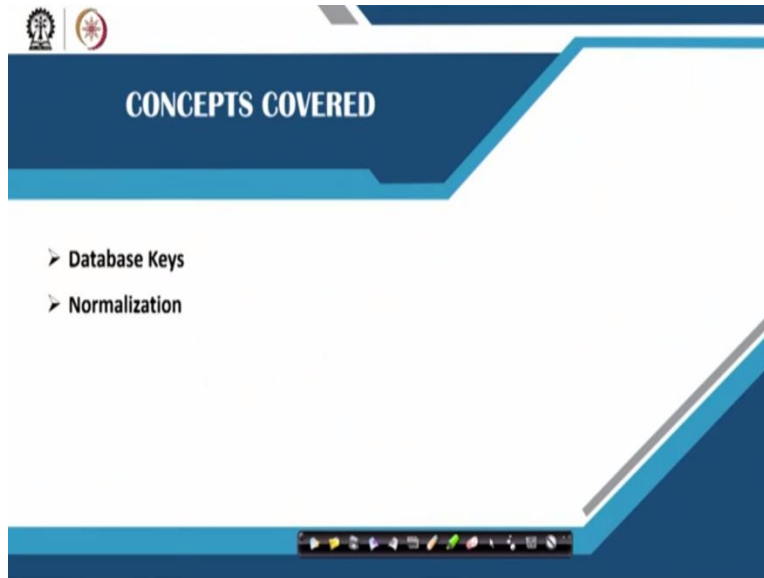
Module No # 07
Lecture No # 33
Database Management System – (continued)

Hello Namaste welcome back to the course on geographic information system. As we were speaking in the previous the modules on database and database management system and when we were looking at what are different database I mean when we looked at a database management system what is its architecture how GIS can be integrated with the database management system. We looked at what do you mean by an entity? What do you mean by relationship?

Then what do you mean by ER relationship model? Then we looked at how and different types of relationship exist? For example, 1 is to 1 relationship, 1 is to many relationship, many is to many relationship. So now after we have understood that we have also understood how we represent an attribute. So now the next set here where we would in this particular class probably will take it forward will understand what are different key that are that exist in the database.

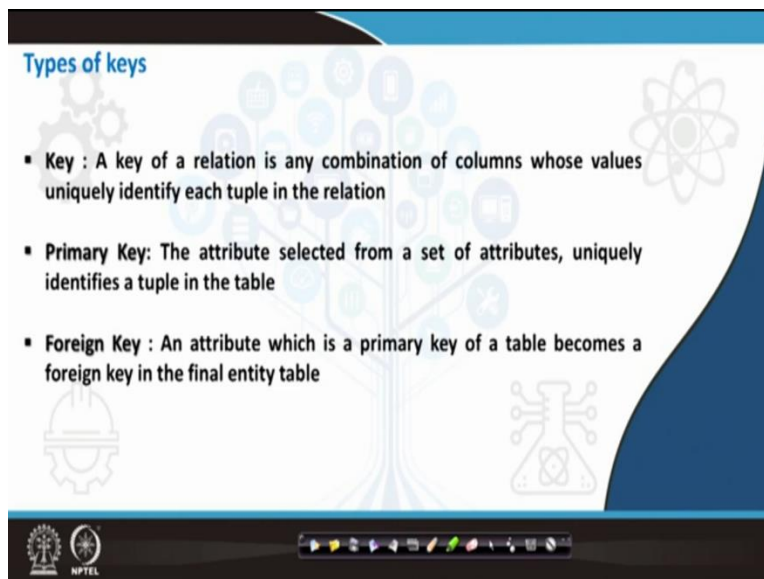
Then how a database can be normalize normalization is also one of the very important techniques in order if you are trying to develop any databases. So how do we normalize different database different forms of normalization of a database that is what we would look at in this particular class.

(Refer Slide Time 01:42)



And as I said we look at database keys and normalization.

(Refer Slide Time 01:46)



So when we look at different types of keys first of all when I say keys it is key can be defined as a relation in any combination of columns whose values uniquely identify each tuple in a relation. When I say tuple it is a row which has a data which has a certain value so that is nothing called as but a tuple. Now when you are looking at a key it is a relation that is a combination of many columns ok. When I say columns these are different values for each of these tuples.

So when you look at each of these tuples, tuples are nothing but rows and columns have values, rows are uniquely identified values. So when you look at it for example a student ID, a student

ID a student name is a tuple and a or student ID is a tuple it can be name etc., All of these are different values in different columns or his percentage of marks in tenth standard twelveth standard all these are different columns so that becomes a key.

This key becomes a primary key if the attributes selected from a set of attributes are unique and it uniquely defined in terms of a tuple. So as I said a student ID is a unique number and it defines only a particular student then it becomes a primary key ok. So let us say this primary key or a particular key is representing 2 different tables connecting 2 different tables and is actually acting as maybe I can say it as bridge between 2 different tables and this particular key will be then called as a foreign key ok.

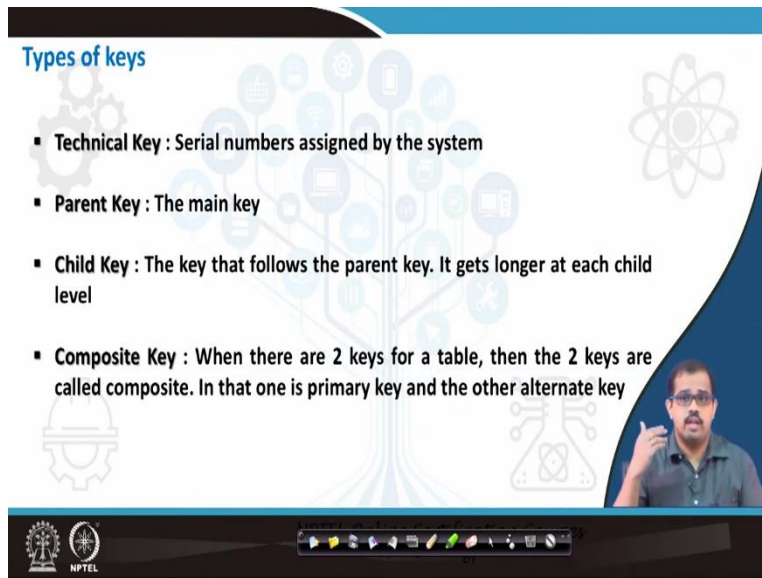
So when I say a foreign key it may it is a primary key of a table that becomes a foreign key in a final entity table. Which means when you are building up the entire database the first primary key that you would have denoted maybe even connecting 2 or 3 or 4 more tables that are actually required for your entire database then it becomes a foreign key for different table that are that exist in that particular database. So the key it is relationship or any combination of columns whose value are uniquely identified.

These uniquely identified values are in a form of rows ok in the form of row are called tuple and these tuples forms a primary key ok. So when a primary key is defining a different number of entity key a different number of entities in different table and these are connecting different table then it becomes a foreign key for such an for a certain tables. So this is different types of keys and there are huge number of list which we will see in the next slides.

(Refer Slide Time 04:50)

Types of keys

- **Technical Key** : Serial numbers assigned by the system
- **Parent Key** : The main key
- **Child Key** : The key that follows the parent key. It gets longer at each child level
- **Composite Key** : When there are 2 keys for a table, then the 2 keys are called composite. In that one is primary key and the other alternate key



So when we when we go into the next so if you have understood what is a primary key and a foreign key well that is what extremely important for us ok. So the rest this is just for your understanding so yea real user should know most of these you know as a part of the database to in order to understand how the database works. But if not at least if you can understand the primary key and a foreign key then the most of the operations is possible in terms of handling a database.

And when you look at the technical key these are the ones that are assigned by the system itself ok. If you have the database management system it provides it assign certain serial numbers which can be said as a technical key. Then you have a parent key is a main key or it can be even said that this parent keys are nothing but a primary keys ok. So when you look at this parent key if forms a very main key and if the rest certain keys that follow this parent key ok then it is called as a child key ok.

So it gets a child key can get as long as a different levels of childs that you have for example in entire module that you develop the first if you look at is a institution then you have let us say a faculties. Then with the faculties they will take 4 subjects let us say 4 subjects has so many students, each students have its own id he or she has number of different subjects all of these. So now you have a parent key you have a child key. That child key has another child key but for the second level of child key this becomes a parent key ok.

And similarly it goes on as a number of child key grows your entire system grows as it goes ok. Then you have a composite key where when there are 2 keys for a particular table 2 different keys for a particular table then the 2 keys are called as composite keys. This can be a primary key also or it can be any other alternate key. So it is not compulsory that both of these keys has to be a primary key ok.

So this is called as a composite key right. So we have looked at what is a primary key? What is a foreign key? What is a technical key? Technical key is where in the system. We looked at what is a parent key and child key? Parent key can be an primary key itself having a lot number of child keys. So that is that is about then you have a composite key ok.

(Refer Slide Time 07:46)

Attributes

Domain: a set of potential values

- Key: minimum nonempty subset of its attributes
 - uniquely identify each tuple (e.g., ss#)
- Super Key: the superset of a key (e.g. {ss# name}) -one or more attributes that are taken collectively and can identify all other attributes
- Candidate Keys: more than one, potential keys (e.g., id or ss#)
- Primary Key: selected one in candidate keys
- Primary attributes: attributes that make up key
- Degree: # of attributes
- Cardinality: # of tuples (a data structure consisting of multiple parts)

So just to summarize all of these we have keys this is a minimum nonempty a subset of an attributes for example let say this is one of the database that we have developed. SS is nothing but the serial number, name is the name of that particular student and you have an ID ok. So when we say key SS can be your key ok that is the serial number is uniquely defining if it is uniquely defining then it becomes a primary key ok depending on its feature it raises a different kind of a key.

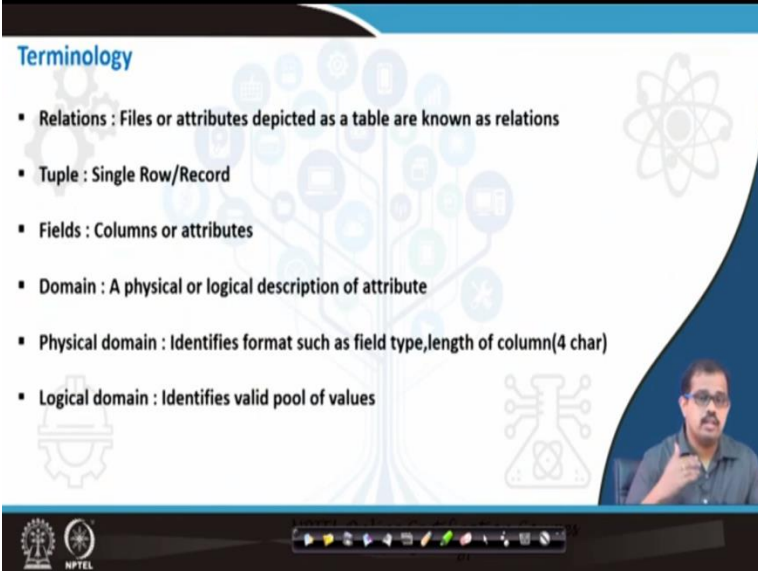
Then there is a concept called as super key. The super set of a key one or more attributes that are actually taken collectively and can identify all other attributes it is nothing but a super key ok.

For example the SS hash name so if you have a serial number with the name or your unique serial number with a name can actually define all other attributes in that particular table then it is called as a super key if both of them are together.

Then you have a candidate key it can be more than one or it can be many so then the ID can be one candidate key SS your serial number can be another candidate key. Then you have a primary key ok any of those candidate keys which has unique representation becomes a primary key ok. So then you have primary attributes. Attributes that make up a key is nothing but a primary key primary attribute. Then you have degree how many number of attributes do you have in that is representing that entire database is nothing but a degree ok.

So as I said cardinality is nothing but you have number of rows representing different entities ok. So when you are looking at cardinality you have how many number of tuples are there in that particular entire database becomes a cardinality of the database. So a data structure consisting of multiple parts or multiple rows is nothing but the cardinality. If there are 100 rows then it becomes its cardinality values 100 ok.

(Refer Slide Time 10:05)



The slide is titled "Terminology" and contains a list of six items:

- Relations : Files or attributes depicted as a table are known as relations
- Tuple : Single Row/Record
- Fields : Columns or attributes
- Domain : A physical or logical description of attribute
- Physical domain : Identifies format such as field type,length of column(4 char)
- Logical domain : Identifies valid pool of values

The slide also features a video inset of a man with glasses and a beard, wearing a dark shirt, speaking. The background of the slide is white with blue accents and various icons related to technology and databases. At the bottom, there is a navigation bar with icons for back, forward, and other presentation controls, along with the NPTEL logo.

Now similarly so we I am trying to give you the flavor of all understanding of why in what does what mean in the database. So this is very important in terms of in case someone is trying to look at the advanced database management systems. So these are some basics very basic

terminologies but there is if you are looking at the advanced part of it probably you will have to take up a database management course.

So if you have learned special analytics here and a database management course probably you can become a super user in terms of looking at the special data in a GIS package ok. So as far as now we have learned what is the key where different types of key that is primary key. We have looked at foreign key, we have looked at technical key, we have looked even a composite key. We have looked at parent key and a child key and once we have understood all of this we looked at what do you mean by a cardinality and a degree.

Degrees number of attributes that it has and if there are many number of tuples then it becomes a cardinality or how many of tuples are there in a particular system then it becomes a cardinality the cardinality of that particular system ok. And some more and whatever some of the terminologies that I defined before let me define it in true sense in terms of what it means relationships. When I say relation, it is the files or attributes depicted as a table are known as relations ok.

Files are attributes let us remove files attributes that are depicted as a table is known as a relation. Now there is student ID, student name, student age, class then you have all the subjects that that particular student has taken. So this all this attributes became become a relations in that particular database ok. Then tuple, tuple is a single row or a record in a database ok or it is also called as tuple. So tuple or tuple is nothing but the single row or a record in the entire database.

Then you have fields be very careful when you have columns that is representing attribute values each of these columns are called as fields ok. So different attributes, different field, tuples, fields ok. So then domain a physical or a logical distribution of an attribute. So this can be anywhere in the database ok. When you look at physical domain identifies format such as field type for a length of a column normally 4 characters etc.,

When you look at a logical domain it identifies a valid pool of values that is the range of values it may be yes or no values qualitative quantitative values whatever is there so that becomes a logical domain. Whereas physical domain gives you the field type, field length etcetera so this

becomes a physical domain ok. So these are different terminologies that are used when you are looking at the database.

(Refer Slide Time 13:23)

Data Integrity Constraints

- Provides a means of ensuring that changes made to the database do not result in loss of data consistency.
- They guard against accidental damage to the database
- **Referential Integrity**
It states that a given non null foreign key value must have a matching primary key value somewhere in the reference table.
- **Entity Integrity**
It states that no value in the primary key field can be null.
- **Domain Integrity**
It ensures that the value in the column falls within an accepted range

The slide features a blue header, a white background with faint technical icons, and a small video inset of a man in a blue shirt in the bottom right corner. At the bottom, there is a navigation bar with various icons and the NPTEL logo.

Now when we are looking at data integrity when you are looking at data usage the very important part of why do you use a database is data integrity. So when you are looking at this there are certain constraints in terms of maintain the data integrity. So what are those concerned let us see in this particular lecture it means says that it provides a means of ensuring the changes made to the database does not result in the in the loss of data consistency.

So whenever you are looking at the database the very important point is why we call as integrated part is that because it can maintain consistency of data. So whatever changes you make so your entire database has to be consistent data consistent that is what is called as a data integrity. So they normally guard against accidental damages to a database model probably sometimes you delete certain things by mistake or sometimes it is wiped out all of this. So when they are actually guarding against any accidental damages that can happen ok.

So without of knowledge you type in something or you enter something into a database when you are using it as a super user so all of these things has to be guarded in terms of that is why it is called it is all databases should have data integrity that is exactly for the reason. So when we look at data integrity there is something called as a referential integrity. When you are looking at the referential integrity it states that a giving a non-null foreign key value.

So foreign key is the one that that may be a primary key in the database that has been shattered but becomes a key that is connecting the other key and for that particular data set this becomes a foreign key. So foreign key value have a matching primary value somewhere in the reference table ok. If that is not there then the referential identity or the referential integrity is missing in that particular data set. Then you have entity integrity. Entity integrity is no value in the primary key field can be null ok please be careful.

So when I give an assignment for certain students to define to develop an entire database with certain the probably using the names of the students in the class. When they develop they keep null values certain primary keys that they would have developed in different tables. So when the primary key cannot be null if it is null then it gives it is wrong entity integrity in terms of data integrity and it cannot be null. So it has to have certain values that has to be maintain.

Then you have domain integrity this the value in the column that is the field ok falls within the accepted range or not. For example when you are actually measuring NDVI values are between -1 and +1. So if there is a value of +4 so that is a value that is out of range. So you have to maintain integrity of that saying that that is a wrong value. So it has to be it has to maintain the proper way that is what is called the domain integrity maintaining the domain knowledge in terms of what is the range of values that can be entered ok. It can be both qualitative or quantitative. I just giving an example of NDVI. So these are different data integrity constraints that are there.

(Refer Slide Time 17:09)

Normalization - Definitions

- A data analysis method used during the design stage of relational data modeling which allows to reduce storage space, data redundancy, inconsistent data and ease data maintenance
- It is the process of organizing data to minimize data redundancy
- Normalization usually involves dividing a database into two or more tables and defining a relationship between the tables
- The process of creating an efficient, reliable, flexible, and appropriate "relational" structure for storing information

The slide features a blue header and footer. The footer contains the NPTEL logo and a navigation bar with icons for back, forward, and search. A small video inset in the bottom right corner shows a man with glasses speaking.

The next concept that are the very important concept that we would understand in the today's class is normalization. When I say normalization that any data analysis method used during the design stage of a relational data. So relational data so data that has relationship entity that has relationship and modeling which allows to reduce storage space data redundancy, inconsistent data and ease the data maintenance this is nothing but nothing had called as a normalization of a data.

You are improving the way the data is that through a data model the way the data is stored the data redundancy are reducing and inconsistent data is being removed in terms of you are trying to improve the system of maintaining the data that is nothing but the normalization ok. So it is a process of organizing the data for minimum data redundancy. When is say redundancy let me see if I have a example here.

(Refer Slide Time 18:17)

EMP

ENO	ENAME	TITLE	SAL	PNO	RESP	DUR
E1	J. Doe	Elect. Eng.	40000	P1	Manager	12
E2	M. Smith	Analyst	34000	P1	Analyst	24
E2	M. Smith	Analyst	34000	P2	Analyst	6
E3	A. Lee	Mech. Eng.	27000	P3	Consultant	10
E3	A. Lee	Mech. Eng.	27000	P4	Engineer	48
E4	J. Miller	Programmer	24000	P2	Programmer	18
E5	B. Casey	Syst. Anal.	34000	P2	Manager	24
E6	L. Chu	Elect. Eng.	40000	P4	Manager	48
E7	R. Davis	Mech. Eng.	27000	P3	Engineer	36
E8	J. Jones	Syst. Anal.	34000	P3	Manager	40

So example here this is an a table which is actually defining what is a name of an employee how many hours that he has worked ok. Maybe here they have 2 cases there is a case P1, there is a case P2 where he has work 24 hours in the case P1 and 6 hours in the case P2. For example when we are actually may be defining here are the entities. So when we define this particular entity it has actually duplicated in 2 places. Just to mention how many hours has he worked in the case P2.

But incase this redundancy can be reduced for example M Smith analyst the 34000 is his salary. So P1, P2 analyst 24,6 are in 2 different fields so in the same row. So that would reduce the data redundancy. So when you are actually accessing the data in your entire database when you give it as Smith if you are searching Smith as a query so it would give you two results that is data redundancy.

So in case if the same thing can give a single result giving the entire value then it gives you then you are reducing the data redundancy in your entire database. That is a very good example even the Lee here is also having the same redundancy issue in terms of this database ok. So that is how you reduce your data redundancy. And a normalization usually involves a database into two or more tables and defining a relationship between the tables ok.

Always when you are looking at normalization you are speaking about relationships and the tables that are there. Entire database will be divided into table and each table is defining a

important part of normalization is efficiency because it actually stores your space also saves a time when you are actually looking at the entire database. For example if someone is trying to access a very big database even through mobile phone though mobile phones today are very efficient.

But let us say in some places where the network is an issue and you have a mobile phone that accessing the entire database with a network it should be extremely efficient in terms of access of that particular database. So a very good example you I have given it previously is your IRCTC website there may be a few years before and now. So when you look there when you looked at it and in 2016 17 are and now you can see a stock change in terms of how the access is there.

What is the speed and how you can easily book your tickets ok. I am not saying that you get a confirmed ticket. But you can look at what is availability etc., in a much easier way which you which never used to happen before. So that is a difference between the efficient database that is normalized and a non-efficient database that is not normalized previously ok.

(Refer Slide Time 23:28)

The slide is titled "Needs for normalization" and lists four types of anomalies with their descriptions:

- **Repetition anomaly**
Certain information may be repeated unnecessarily
- **Update anomaly**
All repeated data should be updated
- **Insertion anomaly**
Cannot insert a certain set of attributes
- **Delete anomaly**
Deleting some information cause to lose another information

The slide features a background with a tree-like structure of icons and a small video inset of a man in the bottom right corner. The NPTEL logo is visible in the bottom left corner.

So if someone ask me why do you need the normalization of the first thing as I said previously said you are repetition of a repetition or normally can be easily removed in terms of the database update anomaly where multiple task are involved we can remove it. Then insertion anomaly where you are looking at insertion of a data that anomaly can also be removed in in in the database.

So then this is because of the normalization then you have delete anomaly. When I say delete anomaly deleting some information cause the loss for other information. So that anomaly can be removed in with normalization. That is exactly why we need normalization ok. So this I have already spoke I would not again get into this.

(Refer Slide Time 24:22)

Without normalization

- Database systems can be inaccurate
- Data modifications may be slow
- Inefficient System
- It might not produce data you expect

So in case let us say that the your database you have not normalized at all. Then your database systems can be inaccurate ok inaccurate in terms of the data that it is representing, the data that is stored in different places and there the data that is actually even connected ok. So that can be inaccurate it can be a for a query it can give wrong results. And data modifications may be slow in without normalization and it can be inefficient system ok.

Inefficient in terms of both a processing a query modifying a particular system or handling a particular system. And it may it might not produce a particular data that you expect or it can I can say it has the information that you expect in terms of output. So without normalization you are actually going to fail in terms of producing the outputs ok. So that is why you need a normalization.

(Refer Slide Time 25:18)

Goals of normalization

- Arrange data into logical grouping such that each group describes a small part of the whole
- Minimize amount of duplicate data stored in database
- When data modifications occur, it happens in only one place
- Building a database which you can access and modify the data quickly and efficiently without compromising the integrity of the data

The slide features a blue and white color scheme with various icons representing data and technology. A small video inset in the bottom right corner shows a man with glasses speaking. The NPTEL logo is visible in the bottom left corner.

So I have spoke about why do you need a normalization but it is for any data scientist there are certain goals through normalization that a particular scientist has to reach which mean to say that how the normalization should actually be affected in your database. So when you look at this arrange data into logical groupings such that a group describes a small part of the whole ok. So the entire group should describe a small part of the whole database.

So for example if I am trying to understand how the what is the academic part of a particular student. So I should be represented with a small dataset than giving me what a particular student is interested in what is his hobbies? What is his age? What where he is coming from his parents his parents name etc., So these are unnecessary information. So this should be able to give me a subset of that particular dataset which is a part of information that I am interested in.

So minimize the amount of duplicate data or redundant data that is stored in the database that is a major goal of any database that you may be representing with. Then when with data modification occurs it happens only at one place ok. You should go into such a normalization in terms that whenever there is a one modification only whenever there is a data modification that has to be done it has to be done only in one place.

So that is what you have to remember then when you are building a database which you can access and modify data quickly efficiently without compromising the integrity of the data. So which means to say that it has to be quick it has to efficient. If it is quick and not efficient again

your database whatever the service that you are trying to provide to the user may fail. Even if it is efficient but not quick again a user may not be want to use the same service again in terms of using your database for application or for usage of a particular user.

So that will actually create a compromising value for your integrity of your data. So that is another goal of normalization where you have to maintain your both data integrity in terms of efficiency and easy access or quick access to your database. So rather than easy access it is a quick access to your database.

(Refer Slide Time 28:00)

The image shows a presentation slide with a blue header and footer. The main content area is white with a blue border on the right. The title 'Summary' is in blue. Below it is a bulleted list: 'Types of Keys', 'Data integrity constraints', 'Definitions of normalization', and 'Need of normalization'. The background has a light blue tree graphic with various icons. A small video inset of a man is in the bottom right. The NPTEL logo is in the bottom left.

So in summary we have learnt about different types of keys today. We started with the primary key, we looked at foreign key, we looked at what is a technical key, we looked at composite key ok. We then looked at what are the integrity constraints that are there in your entire in the entire system. Then what do we mean by normalization? We define normalization in the terms of maintaining the storing maintaining and also querying that particular entire database as why we need a normalization.

And most importantly to reduce redundancy of data and to help user to quickly and easily access a particular database. Then we looked at the normalization we looked at why we need a normalization and what are the different ways or goals of normalization for any person who is actually developing the particular database. So this is about why and how the normalization is important.

So in probably in the last class of this module I would also show you how what are the different normalization techniques from first normal form to fifth normal form. I would show you with examples how a particular database can be made more efficient and quick to handle. So in the next class probably we will look at different database management systems. What are the different database management system I would just give you a very brief on each of the management systems that are there whether it is hierarchical relational and object oriented database management system.

Just a brief overview of all of these systems and next we would look at different normal forms that is very important point important part of how your database is handled. So till then have a nice time thank you very much.