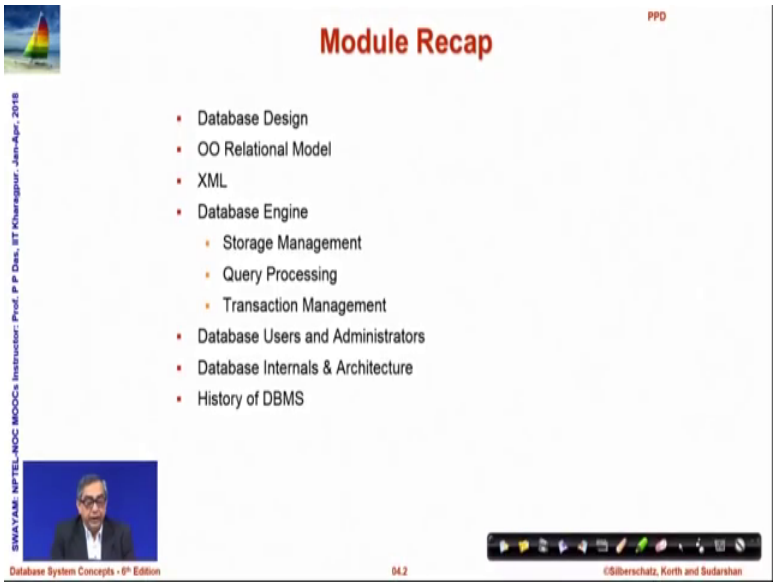
Database Management System
Prof. Partha Pratim Das
Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur

Lecture - 04
Introduction to Relational Model/1

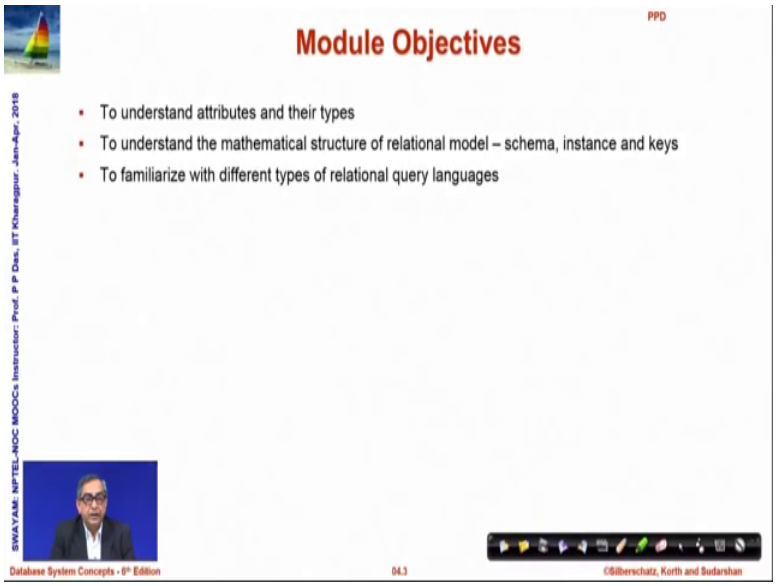
Welcome to module 4 of database management systems, in the last two modules, we have introduced the basic notions of DBMS in this module and the next; we make an introduction to the relational model, which we said is a major data model that we are going to use.

(Refer Slide Time: 00:46)



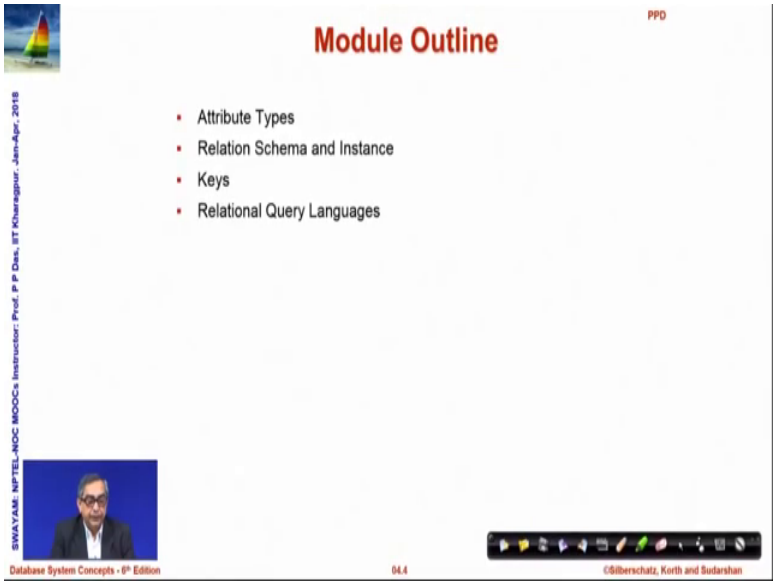
So, this is what we did in the last module.

(Refer Slide Time: 00:52)

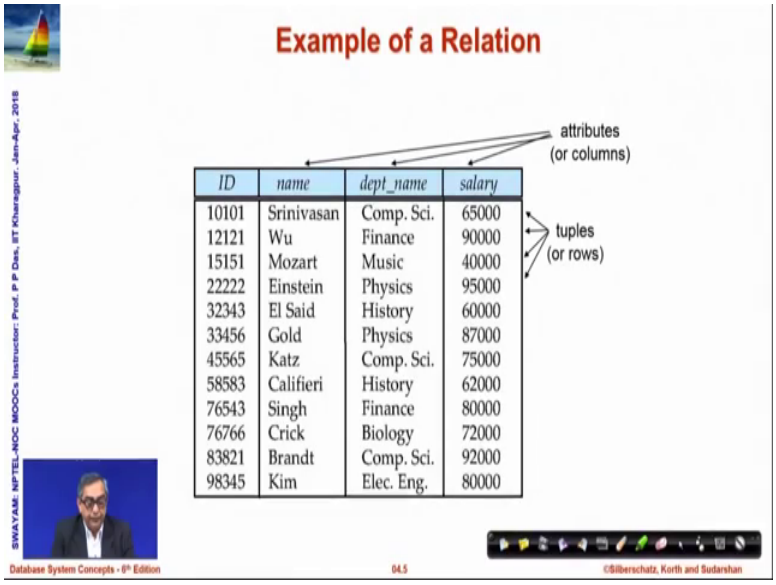


And in the current one, our objective will be to understand key concepts of relational model that is attributes and their types, the basic mathematical structure of instance schema and what is known as keys and to familiarize with different types of relational query languages. This is a module outline that we will follow.

(Refer Slide Time: 01:17)



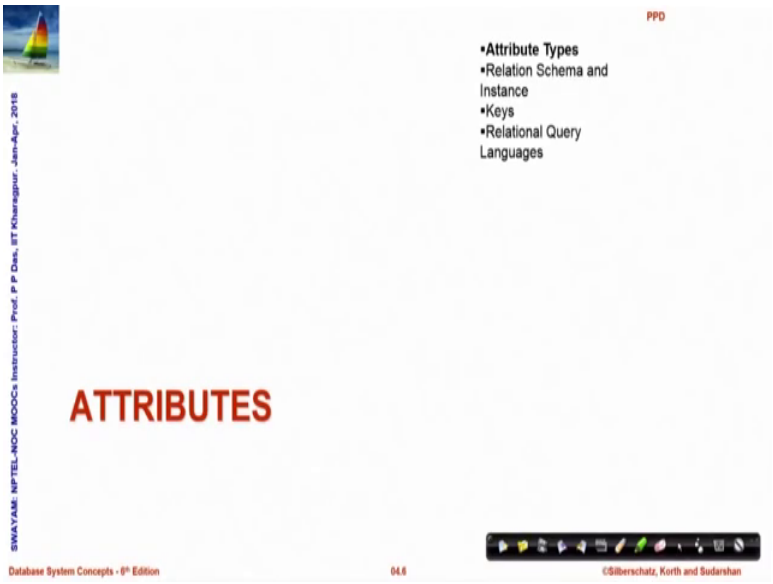
(Refer Slide Time: 01:22)



So, we again repeat this example from past, this is an example of instructors, in a university a table of instructors given by attributes or columns I D name, department name and salary. These are the four columns, the four I Ds and multiple rows, which are specific rows or we often refer to them as tuples. So, you can say that since there are 4 attributes, that is every row has 4 columns.

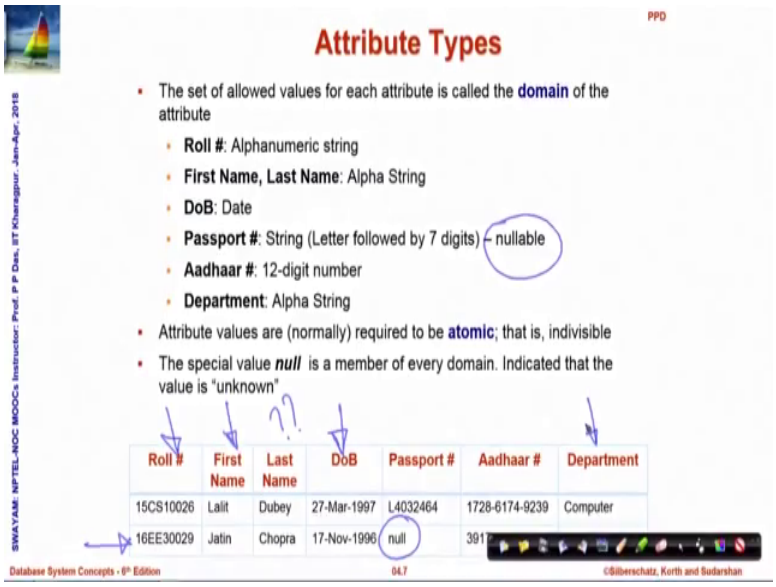
So, this is a 4 tuple that we have and such a table is called a relation is as simple as that. So, this is it, whenever we talk about a relation, we have a number of fields number of attributes number of columns, whatever way, we said of a table and that table according to those columns, it has multiple 0 1 or any number of rows of values, filled in and that is what is a relation.

(Refer Slide Time: 02:34)



Now; so, let us look at attributes more specifically.

(Refer Slide Time: 02:37)



So, attributes each column is an attribute as you said, this every attribute has a domain. The domain is a set of possible values that attribute can take. So, if you just look into the example here, so, I am trying to define a table having different students. So, there is a roll number for a student, there is a first name last name, the date of birth; DOB, the passport number, the Aadhaar card number, the department to which the student belongs and so on. So, let us say these 1 2 3 4 5 6 7 are the different attributes.

Now, if we look into every attribute, then every attribute has a set of possible values of which some value is entered in a particular row. For example, the roll number is an alphanumeric string, as you can see, it has numeric as well as it has letters whereas, the first name or the last name are simple alpha strings. In fact, we can also say that the roll number actually is not only alphanumeric, it has a fixed length, here is it has length of 9. So, you can say alphanumeric strings of length 9 are eligible for being values of this domain.

There could be more restrictions, but that the domain will be certain collection of values which are possible as values of that attribute, when you talk about D o B that certainly has to be a date. So, it is written in the form of d d m m m y y y y that is two digit date, three letter month codes and four digit year, the passport number is a string, a letter followed by seven digits. The other number is a twelve digit number, the department is alpha string and so on. So, the domain is a set corresponding to an attribute, which define that all possible values that attribute can take ok.

Now, these attribute values if you look at they are atomic in nature that is you cannot divide them into smaller parts. So, what I mean is say when we are talking about date of birth the whole date of birth type the date type is one atomic value. For example, if you were to code this in c what you could do you could possibly create a structure with three fields; one is date, one is a month, one is a year and we will say that this composite record composite structure is actually my date.

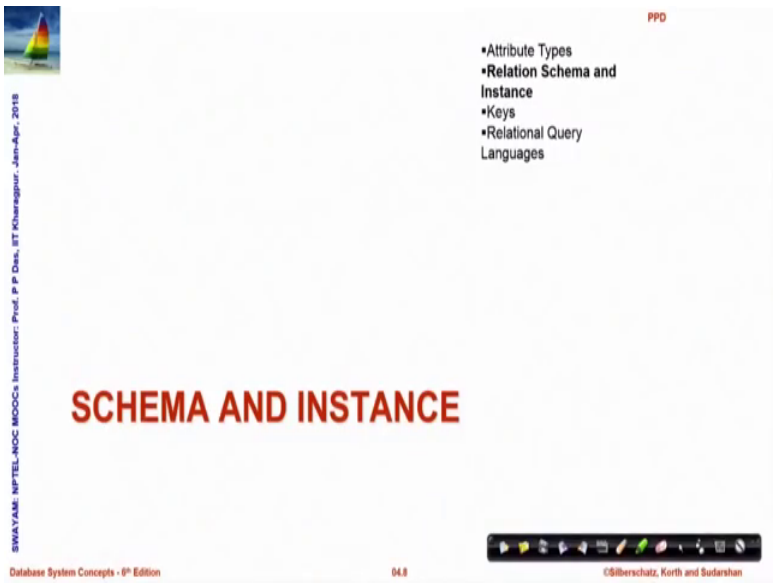
They can do a l s type def, you could do if you are working in C plus plus, you will define a class called date, which has these components and as well as operations with them, but that kind of types are not allowed in a relational database, it has to be an atomic type. So, in a relational database will give you atomic type called date, but all of these are pre specified and has to be taken as one unit, other atomic types are integer like we do not have an integer field here. There are strings; there are numerical values, which are kind of floating point values and so on.

Now, some attribute may have a special value called the null value, which is the member. It is domain; actually every attribute of any domain can have this special value. The null value is not actually a value; it is actually an absence of a value. So, it says that this value is not known. So, if you look into the example above, then you will see that for passport

we have said that the passport is a string letter followed by seven digits and it is null able, which means that in the passport field, I may have a value, may have this null value which means that it is not that, the passport is null, what it is saying is this passport number for this particular student, the row number 2 is not known, is unknown.

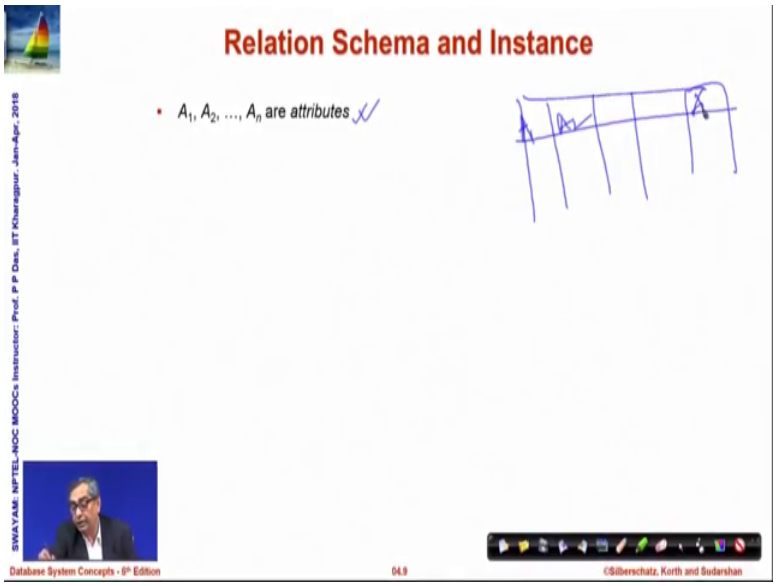
Now, all fields may or may not be null able. For example, will not allow D O B to be null able, date of birth has to be there, will not allow roll number to be null able, will not allow first name to be null able, but we may allow last name to be null able. It is been a style, let not to use your last name, many people just use one name. So, you could allow that, it is not known, it is not there whereas, department may not be null able, it must be there. So, null is a very critical concept and what it actually does? It actually creates a lot of issues and complications in terms of defining many operations. So, understanding null as a value in terms of an attribute is a critical requirement for the design.

(Refer Slide Time: 08:03)



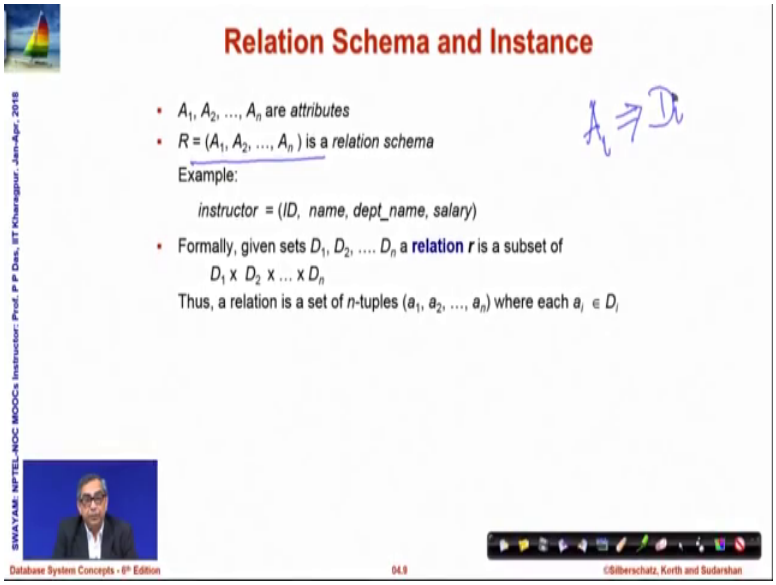
Now, coming to the schema and instance, we have discussed about the basic understanding of schema and instance. So, understanding them formally now, we say that if we have a schema. So, it is like a table having multiple columns say, there are n columns, having names A 1 to A n, then this A 1 to A n are the attributes.

(Refer Slide Time: 08:26)



So, these are the different attributes. So, if I have this, then it basically means that I have a table, where these are the columns A 1 A 2 A n like this.

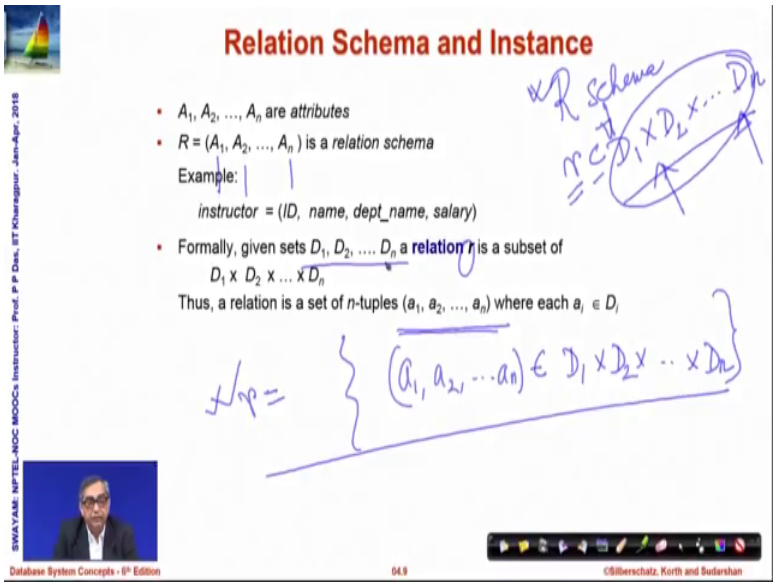
(Refer Slide Time: 08:50)



So, then a relational schema is a collection of these attributes. So, it is a collection of all these attributes. So, we said R is a relational schema, which has attributes A 1 to A n. Now, every attribute A i has a domain D i. So, for every attribute, I have a set of values that are possible. So, if you, if you recall then here we had different, these are the different attributes and these are their different domain. So, D o B is an attribute and the

domain is date. So, any possible date, other is an attribute and this is the domain, which is A. So, all attributes, each attribute will need to have certain domain and those are marked by the D Sets. So, we will say that a particular relation a particular relation R.

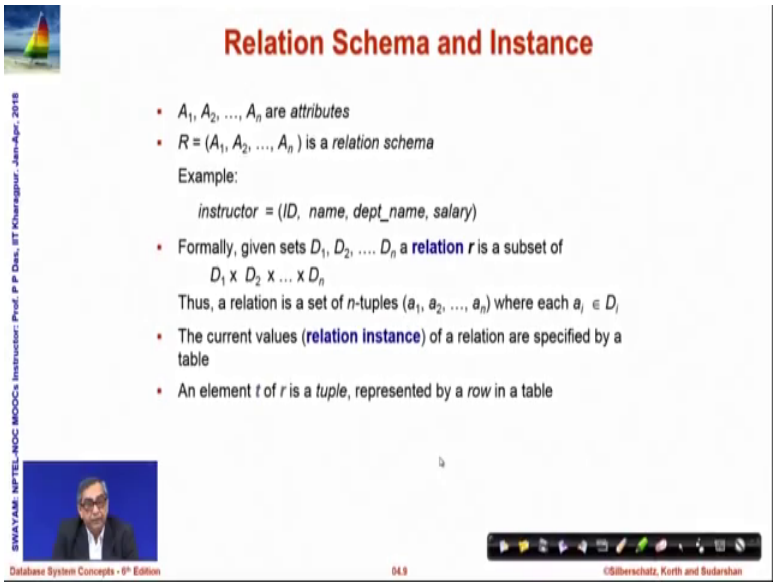
(Refer Slide Time: 10:01)



So, R is a schema. So, a particular relation R is a subset of D 1 plus D 2 cross dot dot dot D l. So, recall the mathematical notion of relation which says that a relation is basically a subset of a set. So, these are the possible values. So, the first attribute can take values from D 1, second attribute can take values from D 2 and so on and the nth attribute can take values from D n. So, any specific row, any specific record is a set of values for A 1 A 2 A n and therefore, is a member of this Cartesian product and the relation is a subset of that. So, this is a D value is an n tuple, which is a subset of this A 1; A 2 A n.

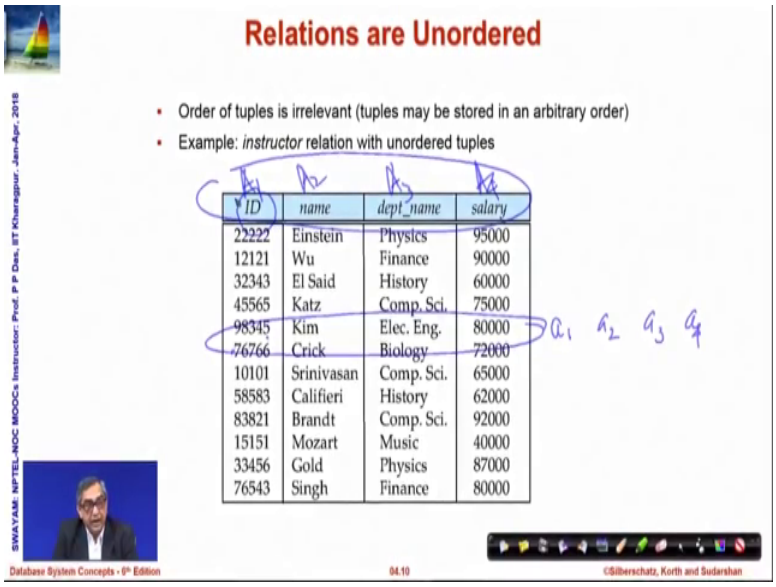
This particular record is an element of this Cartesian product set and R necessarily is a set of such tuples that is a mathematical view of the schema and the instance. So, this is the schema and this is the instance corresponding to that schema based on the different domains of the different attributes and this is the notion that we will continue using. So, please try to follow this carefully.

(Refer Slide Time: 11:46)



Now, whenever we have an instance, we mark that as a table and every such table.

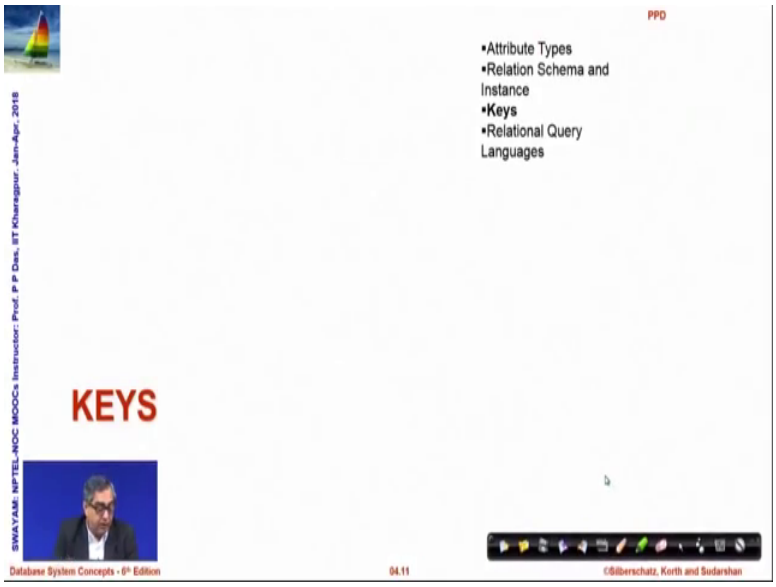
(Refer Slide Time: 11:56)



So, here you have now understood it very well. So, these are my attributes. So, this is A 1, this is A 2, this is A 3, this is A 4 and any one i name is at the different values a 2 a 3 a 1 a 2 a 3 a 4 98345 is a 1 Kim is a 2 and so on. Now, naturally this, it is not visible from the instance, because we are taking an instance view, we are not being able to see, what that domain is that will be visible? If we look at the corresponding D D l, the definition language description of the schema, which must have specified I D as a numeric value,

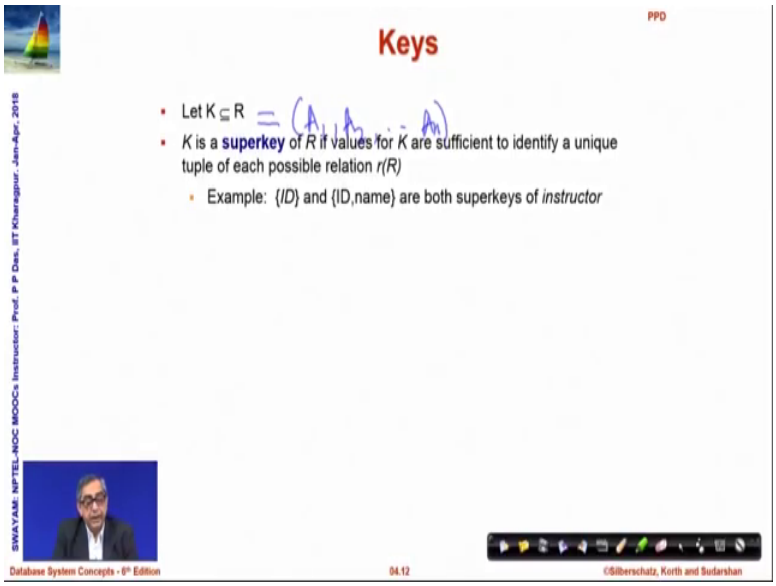
the name as a string value the department name as another string value whereas, salary as a numeric value and so on. Now, what is important to note here is a relation necessarily is a set as we said is a set, which is the as the relation R is a set, this is a set, which is a subset of this set.

(Refer Slide Time: 13:04)



So, we know the elements in a set are do not have any ordering, they are unordered. So, a relation is necessarily unordered. So, it does not really matter that in terms of this collection of rows, which row is at what position, if I reorder them, the relation does not change it is just that they are a collection of this set of rows. So, that lack of ordering is critical information that we will have to remember in mind next concept is key.

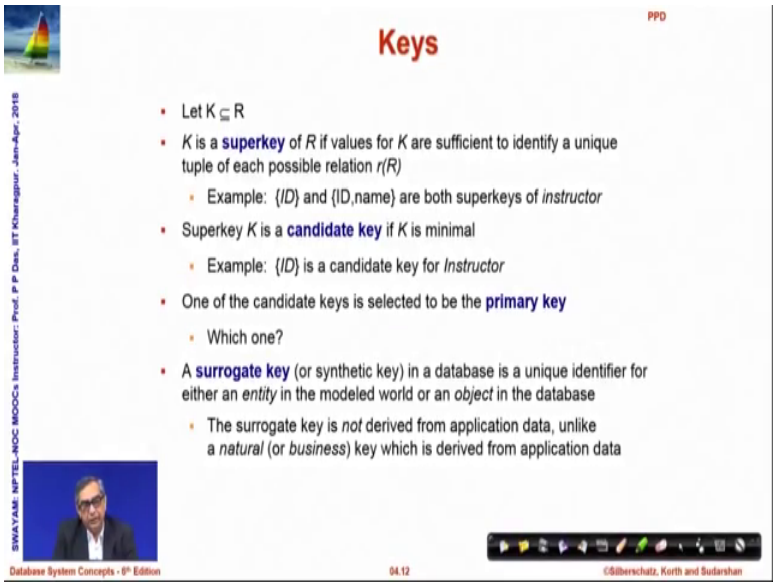
(Refer Slide Time: 13:49)



So, R as we have seen is a relational schema, which is a collection of attributes A 1 A 2 A n. Now, K let K be a subset of R. So, it is one or more attributes, it has to be a non-empty subset. Now, we will say that K is a super key of R, if we consider the values of different tuples in the attributes of K and we find that there cannot be two tuples, which are different, but match on these attributes, which mean that the values of the attributes of K, uniquely identify each row of the relation, then we will say that K is a super key of R.

So, the instructor table that we have seen, I D is a super key similarly. So, K can be taken as a singleton set of attribute I D or K can be thought of as the set comprising I D and name both of them are super keys of instructor.

(Refer Slide Time: 15:32)

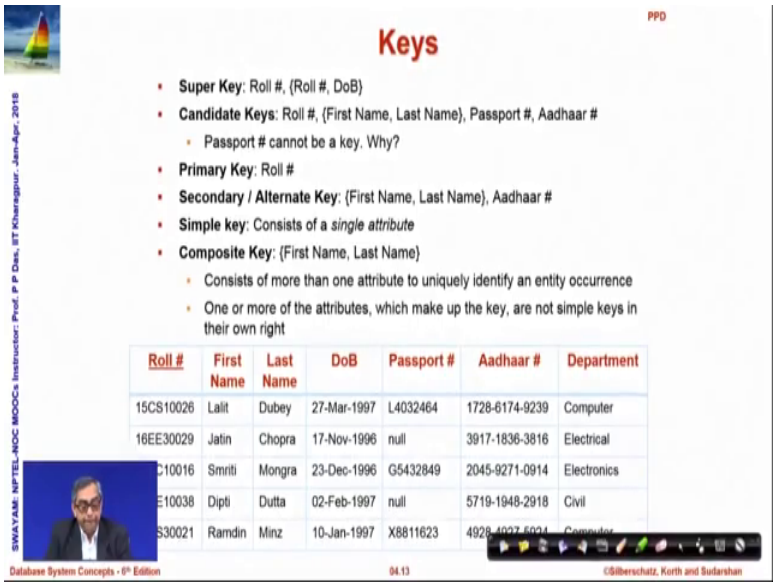


Now, we say a super key K is a candidate key, if K is minimal. So, the idea is like this that this is a key, super key, this is also a super key, but certainly this is a subset of this. This is smaller than this. So, we will say this is a candidate key, but this is not a candidate key, because it does not satisfy the minimality condition.

There could be multiple candidate key in a relation, if there are multiple candidates key then we select one to be the primary key. Now; obviously, there is a question of which one we select, but anyone can be selected as a primary key, which is the key of the relation and we will see that in some cases, there is concept of surrogate keys.

So, if I have a relation where there is no attribute, whose value can uniquely identify each and every row of the table then I might synthetically generate a value for example, like a serial number, I can generate a serial number and say that this is my value. So, that serial number or that computer generated field value has no business implication, the real world did not have this value, it is not like a Adhaar card number or like a passport number, but it is a value which is purely generated to identify every row uniquely. So, such keys are known as surrogate keys or synthetic keys.

(Refer Slide Time: 17:40)



Now, let us look at some examples, this is again the same student database, I just shown a while ago the same set of columns, but I have added few more rows. Now, if we look at what could be a super key there are several candidates, but I have just written a few roll number is certainly a key, because I am assuming that the university assigns roll numbers to uniquely identify every student.

So, there cannot be two rows in this table, which match in the value of the roll number and does not match in the values of the other fields. So, roll number can uniquely identify, if it can then any super set of attributes, which continual number will also be a super key. So, roll number and date of birth together is a super key that can also unique to identify every row trivial. What are the candidate keys? Now, there are of course, that could be several other super keys that has to be kept in mind, the candidate keys are roll number is a candidate key, the first name last name together, we can say is a candidate key.

So, we are saying that not only the first name, but if we take this pair, you remember the key, the set of attributes forming a super key is a set. It is not an individual field. So, I say the first name last name together, from say, key well. This does make some assumption, because if I say the first name last name together from say key; that means, that there cannot be two records in this student table, where the first name and last name match, but the records are different. So, which mean that no, two students having the

same first name and last name can be enrolled in the university. This is a restrictive assumption right, but I am just making that assumption to illustrate the different possibilities; then what is the other possibility passport number? Everybody has a unique passport number. So, passport number could also be a key, could be a candidate key. Adhaar number; everybody has a unique Adhaar number. So, that can be a key and so on.

So, these are called the candidate keys. Now of course, we can observe that given the data it is clear and it was also mentioned when the schema was designed this passport number cannot be a key. Why can it not be a key? Can two students have same passport number? Of course, not every student has a unique passport number, but it is possible that some student does not have a passport. So, if some student does not have a passport then the passport number field of that student is a null, the passport number is a null able field, if the passport number is null then it is possible that multiple students may not have passports.

So, as we can see here that this student Jatin Chopra does not have a passport. So, similarly, Dipti Dutta does not have a passport either. So, certainly if this were to be the key then for all records, for which passport number is nil, this value would not be able to distinguish them in terms of the rows of the table. So, we have to say that passport number cannot be a key or in other words, we can say that no key can be a null able field.

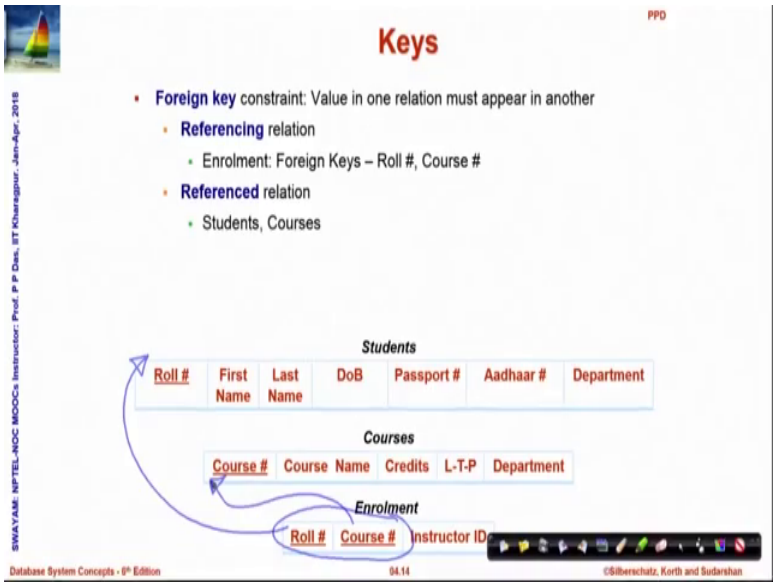
No key attribute or a participant to a key attribute could be a null able field right. So, this is one observation here. And, so that clearly also implies that, if we say that Adhaar number is a valid candidate key that will mean that for admission to that university having Adhaar number, would be mandatory, if somebody does not have a Adhaar number that will have to be null, which is not allowed.

So, let us move on. So, one of these candidate keys have to be made the primary keys. Let us say; we make roll number, the primary key and since, we make roll number the primary key in the schema. We underlined the roll number attribute; this would be a common way to show that roll number is a primary key. So, the others that are not taken as a primary key are called the secondary or alternate key. So, first name last name pair could be an alternate key Adhaar number could be an alternate key and so on. A key is said to be simple, if it consists of a single attribute.

So, roll number is a simple key, Adhaar numbered is a simple key, if it were taken to be primary, but first name last name pair, if we take that to be a primary that will not be considered assemble simple key, because it has more than one attribute naturally the other, if you have a simple key.

They have other side is a composite key is one, which has more than one field such that none of those fields individually can act as a key, but together they can act as a key. So, first name itself cannot be a key last name itself cannot be a key, but together. They can be a key of course, under the assumption that no two students with the same first name last name are given admission. So, these are the different types of keys that can happen.

(Refer Slide Time: 23:32)



Let us have some more views with the keys, we extend the schema and besides a student I introduce two more schema; one is called the courses, which is given by course number, course name credits L T P. L T P is number of hours of lectures tutorials and practical's and the department. So, these are the different fields and from the convention already stated you can figure out that course number is the key primary key of this relation. I use another schema, which is enrolment, which describes which student is attending which course. So, it has a roll number and the course number.

So, roll number of the student attending the particular course number and it also has an instructor I D as to who is teaching that course given this. You can see that in the enrolment relationship, I have this pair roll number and course number, which will

certainly be the key for enrolment, because if I have two rows in enrolment. How they will be distinguished, they cannot be distinguished by roll number, because a particular student may take multiple courses.

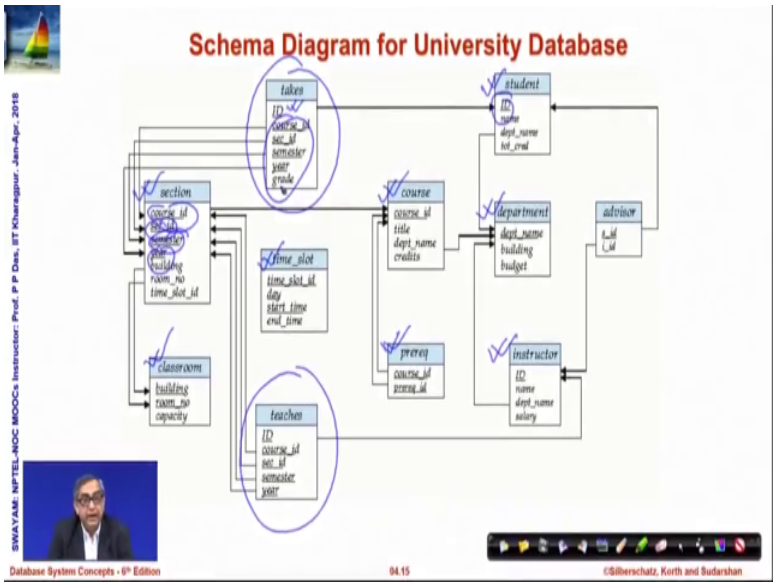
So, there will be multiple records having the same roll number, but different course number, the course number by itself cannot be the key, because every course will have multiple students. So, there will be multiple rows having the same course number, but all different roll numbers, but if we take this together, roll number and course number together then that forms a key.

Now, such a key such a key having roll number the roll number itself is a key of another relation the course number itself is a key of another relation. So, when we take the keys of other relations to form the key of a relation then we say that these are foreign keys. So, roll number and course number are foreign keys in student and course and since from enrolment the student and courses are being referenced are being referred.

So, we say enrolment is a referencing relation and students and courses other reference relation and we will often like to also mention as to what is a foreign key of a relational schema, because that will help us understand how the different schemas are interrelated and we will see that, this will come out directly from the notion of entities and relationships of a year model of a year diagram, a key is called, to be said to be compound, if it consists of more than one attribute to uniquely identify an entity occurrence.

So, each attribute which makes up the key is a simple key, in it's own right, mind you there is a subtle, it sounds very similar. We talked about composite key; earlier, we talked up; we are talking about compound key here, the subtlety of the difference is in a composite key, every component attribute is not a simple key by itself, but and the components come from the same table in a compound key. The components are simple key in their own right, in some other table and are put together as a compound key in the given table. So, the rule number, course number in the enrolment table is a compound key.

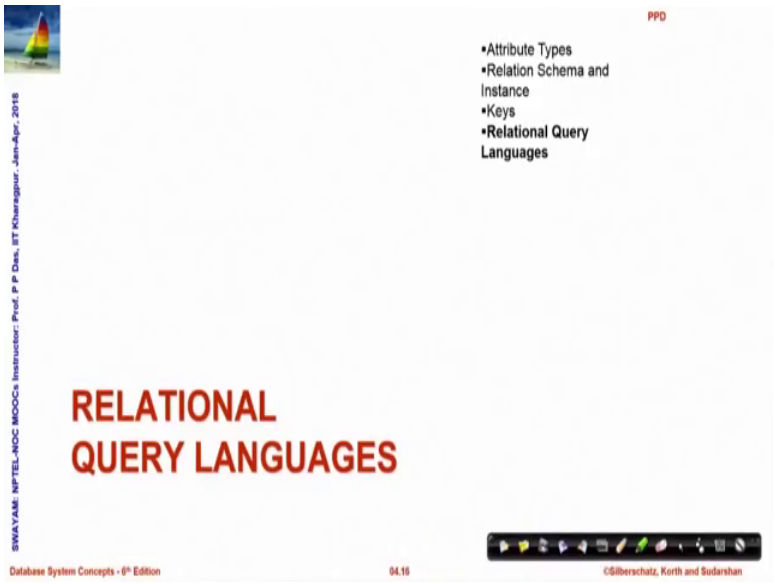
(Refer Slide Time: 27:31)



So, with this I would request you to spend some time with this relatively elaborated schema compared to what we have done already of the university database. So, every rectangular box shows a relational schema on top of each in blue, is written the name of that relation relational schema. So, it has a relational schema like courses, the students the instructors, the departments, the prerequisites, the time slots, the classrooms and so on.

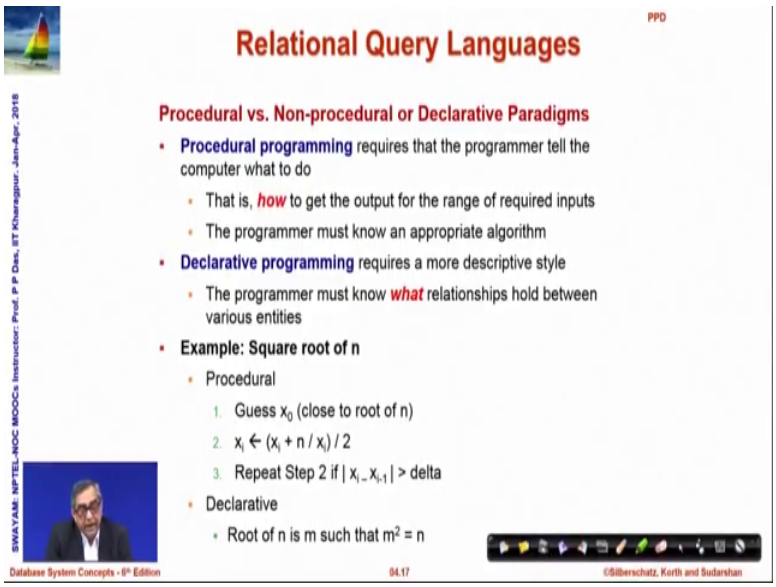
The sections and the relationships between them for example, the relationship is takes is a relationship, which relates students with different sections, with courses, teachers is another relationship, which relates to instructors with sections. So, it is showing you directly as to how the keys of this, what are the attributes? What are the key attributes primary key attributes and also what are the foreign keys that we have in this for example, intakes this is a foreign key, which is featured here, course I D section, I D semester here are the foreign key part of the takes that exists here. So, please study this schema. We will keep on regularly referring to this schema in future as well. So, this is what we have here.

(Refer Slide Time: 29:29)



Now, we move on to the relational query language, we briefly talk about the relational query language.

(Refer Slide Time: 29:33)



Now, we will have to in this the key thing that we need to understand is the relational query language is somewhat, different from the programming languages that you have studied so far which are procedural in nature, in contrast the relational query language is non-procedural or declarative in nature, a procedural programming language requires that the programmer tell the computer how to get the output given, the input a pro

program is about finding output, for a given input and you write a procedure, the sequence of steps that need to be done. So, that given the input, you can compute the output.

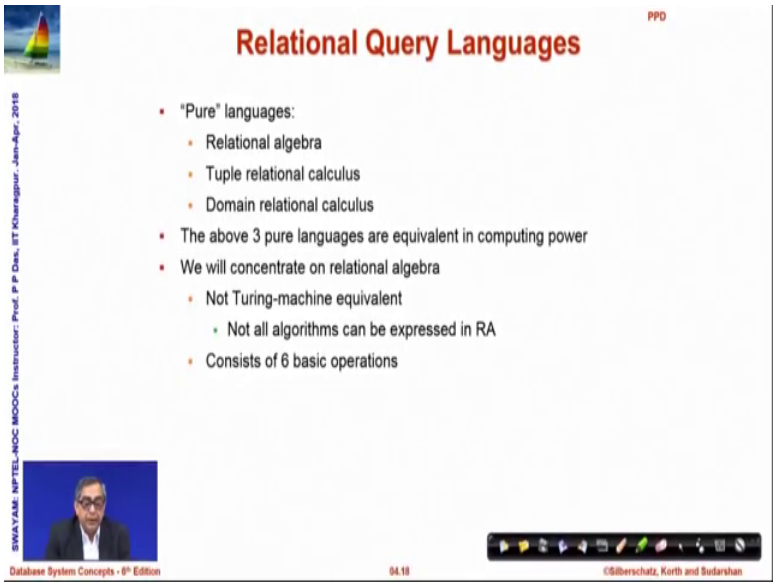
So, you say how that computation has to happen and the programmer must know that algorithm in contrast, in declarative programming, you say what you want? You will do not say how that needs to be computed? How that will be computed? You may not even know that, you may not even know a single algorithm to compute the output, but you specify what output you need. So, this distinction between how and what of programming differentiates procedural and declarative programming.

So, all that you have studied so far in terms of C C plus plus, java python and all that are procedural programming, where you necessarily have to specify, how you will have necessarily; have to specify what the algorithm is, but in declarative you just say what you need. So, just a simple you know pathological example to understand this difference. Suppose, we were interested in computing the square root of a number n assuming n is a positive number, the procedural step would be something like; this is an algorithm that you guess a X naught, which is a square root, which is close to the root of n.

I mean some guess you make and then you repeatedly refine this estimate by taking the arithmetic mean of the estimate and the quotient of the division of n by this estimate. So, you take an arithmetic mean and find the new estimate and repeat the steps, I mean as long as the difference between the two conservative estimates is more than a certain value delta, this is a procedural algorithm, you are giving an algorithm. So, given and following this algorithm, we will find the square root declaratively, you can just say that what is the result? I want to result m such that m square equals n.

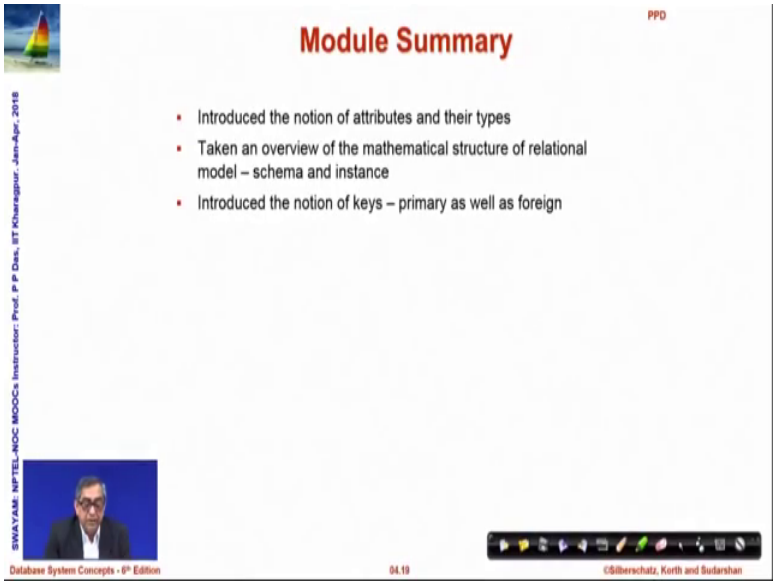
So, you are again asking for the same feel, you are expecting the same output, but the way you are saying is not an algorithm, you are rather specifying a predicate, which must be true in your output. You are saying that the predicate is m square must be n. So, whatever m is that square of it must equal n. So, this style is known as declarative whereas, the earlier style is known as procedure.

(Refer Slide Time: 32:38)



All query languages, relational query languages are declarative in nature. We have talked about the pure languages, are they are all equivalent? We mentioned that earlier also and also again to remember that none of them are actually Turing equivalent; that means, that not all algorithms can be expressed in them or specifically relational algebra, which we will look at in more depth and the relational algebra will consist of six basic operations, which we will discuss in the next module.

(Refer Slide Time: 33:11)



So, to sum up we have introduced the notion of attributes and their types, we have taken an overview of the mathematical structure of the relational model schema and instance, we would say mathematically they are relations, mathematically they made a mapping and we have introduced the very important concept of keys and in that very specifically, what is a primary key as well as what is a foreign key? In the next module, we will discuss about the different operations of relational model relational logic.