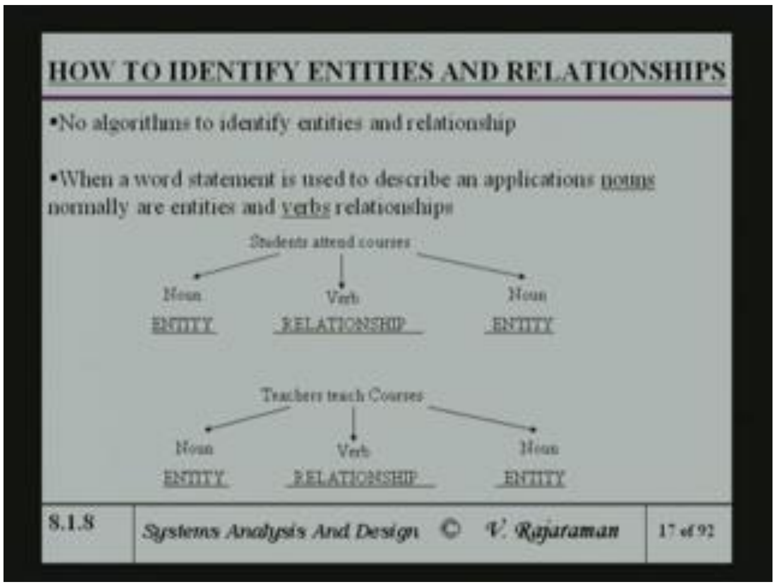
System Analysis and Design
Prof. V. Rajaraman
Department of Super Computer Education & Research
Indian Institute of Science, Bangalore

Lecture - 23

Last time, we were starting to look at an entity relationship model. We said, that the entity relationship model is the start of coming up with a, what is known as a relation database. So, the entity relationship model is required as a beginning at the beginning to be able to identify what are the entities you take part in the relation. And what relationships we have with other entities. So, we said that there is no real algorithm, to detect the identities identify entities or the relationships.

So, given an requirement facilitation or a word statement, one thumb rule is to pick up the nouns as entities and the verbs, connecting verbs as the relationship. So, in the case of student attend courses, students is a name there is an entity. And attend, which connects students with courses is a relationship. And courses is another entity.

(Refer Slide Time: 02:42)



HOW TO IDENTIFY ENTITIES AND RELATIONSHIPS

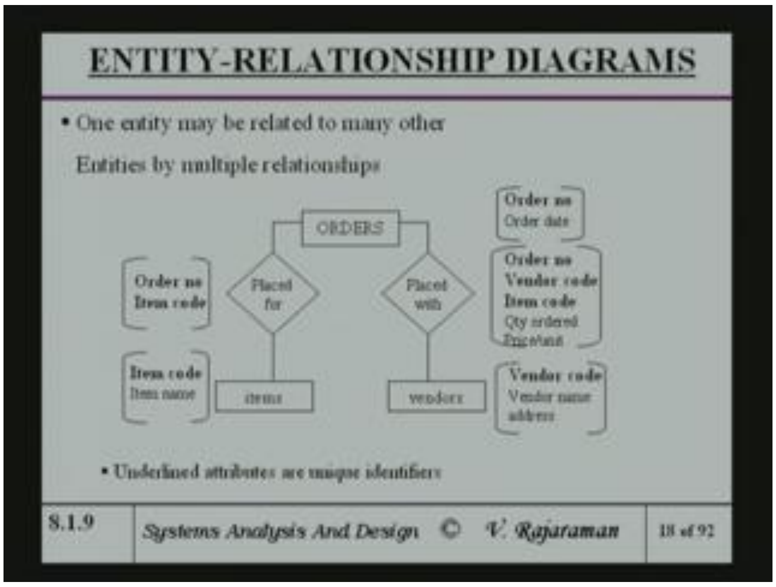
•No algorithms to identify entities and relationship

•When a word statement is used to describe an applications nouns normally are entities and verbs relationships

8.1.8 | Systems Analysis And Design © V. Rajaraman | 17 of 92

Teacher, if teacher teach courses, then teachers as one entity. And courses as other entity. They are related by relationship called teach. And is shown in a diagram, where squares rectangles are used for entities. And rhombus kind of this diamond shaped box is used to identify relationships.

(Refer Slide Time: 03:23)



ENTITY-RELATIONSHIP DIAGRAMS

- One entity may be related to many other Entities by multiple relationships

ORDERS
Placed for
Placed with
items
vendors
Order no
Item code
Item code
Item name
Order no
Order date
Order no
Vendor code
Item code
Qty ordered
Price/unit
Vendor code
Vendor name
address

- Underlined attributes are unique identifiers

8.1.9 Systems Analysis And Design © V. Rajaraman 18 of 92

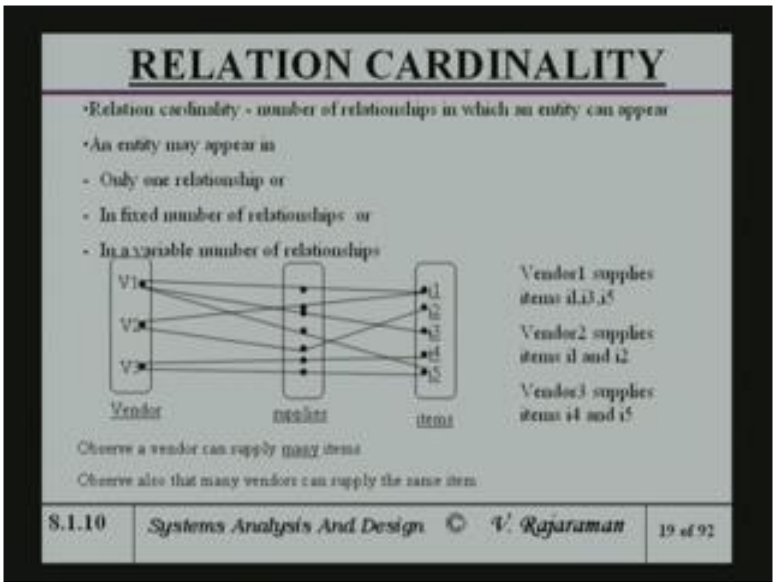
Now, look at another example, where if you look at orders. Orders are placed for items and they are placed with vendors. So, point of this example is, so that the same two entities may be connected by more and one relationship. In this case, place falls on relationship and placed with is another relationship. So, the order, the order is what certain attributes. The attributes is order number and how you did. Order number is an attribute which identifies. It is uniquely identifies the order. It is a key attribute, so it is got to be unique.

And if you look at placed for it connects with items. So, the item itself has got attributes item code and item name. And the, placed for has got a composite key namely order number and item code. The reason is, they an order can be placed for many items. And the items many and many orders. And similarly, an order placed may be placed with many vendors. A vendor may have many orders. So in the case, of the relationship, which is placed for, the order number and the vendor code and item code. All three appear as a composite key.

I will explain this in a greater length with more examples as we proceed. But, the general idea is that before I really define. What is borne as the cardinality as a relationship. I am just giving this example, but cardinality. Or the so called is it a 1 is to 1 relationship or a 1 is to 1 relationship all those things will be the ones determines. Whether more than one key appears in a relationship or just a single key appears in the relationship. This determinacy and the determinacy diagram will so on; will be later on will explain that.

And when we kind of understand this reason why more than one key is put it in the relationship ((Refer Time: 06:11)). Because, it is a composite, what is called a composite key. This is what, meant by relation cardinality.

(Refer Slide Time: 06:21)



RELATION CARDINALITY

•Relation cardinality - number of relationships in which an entity can appear

•An entity may appear in

- Only one relationship or
- In fixed number of relationships or
- In a variable number of relationships

V1
V2
V3
i1
i2
i3
i4
i5
Vendor
supplies
items

Vendor1 supplies items i1,i3,i5

Vendor2 supplies items i1 and i2

Vendor3 supplies items i4 and i5

Observe a vendor can supply many items

Observe also that many vendors can supply the same item

8.1.10	Systems Analysis And Design © V. Rajaraman	19 of 92

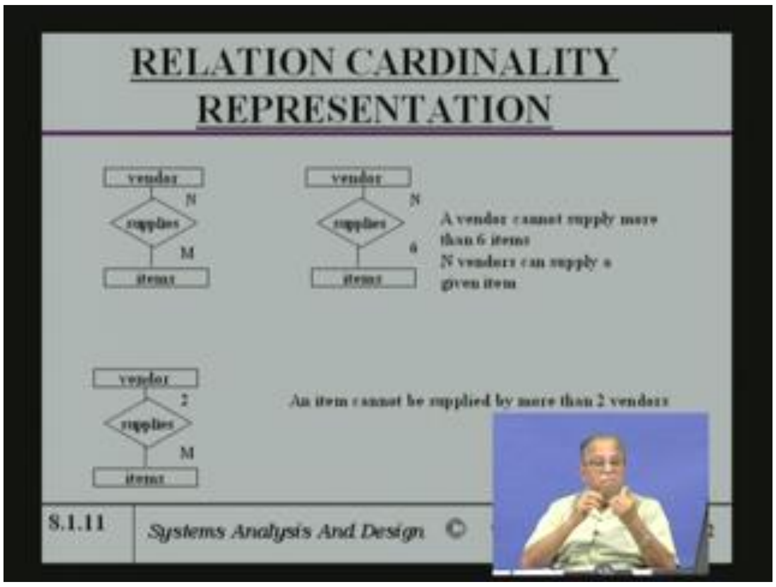
We said that, vendor can supply many items. For look at the vendor and items are the entities and supplies a relationship which connects vendors and items. So, a 1 vendor, vendor number 1 can supply item 1 as well as item 3 in this case. So, two lines are shown and connecting v 1 i 1 and v 1 and i 3. And when they pass that supply relationship thick dot is shown. So, in that supplies connects these two entities. We will look at vendor 2, vendor 2 supplies item i 1, as well as item you can see supplies item i 2.

So, the item 1 is supplied by both vendor 1 and vendor 2. That means, an item can be supplied by more than 1 vendor. And a vendor can supply more than 1 item. This is called a multiple relationship that is n is to 1 relationship. And this can be found out by just looking at this diagram. And see, how many entities are connected to how many other entities through the relationship. And from this diagram itself evident, whether is a, what is a multiplicity of the relationship.

So, an entity may appear in only one relationship or in fixed member relationship or in the variable member relationship. In this case, the simple example of just 5 items at the maximum vendor 1 can supply is 3 items. In that maximum any vendor can supply is 3 items. Because, vendor 2 supplies 2 items and vendor 3 supplies 3 items. But the point is in general there are N vendors and M items. So, if do not have any prior knowledge of how many items a particular vendor can supply.

It is indefinite then it is called a N to N, that is although a vendor can supply many items. And of course, the, that many vendors can also apply, supply the same item.

(Refer Slide Time: 09:00)


RELATION CARDINALITY REPRESENTATION

A vendor cannot supply more than 6 items
N vendors can supply a given item

An item cannot be supplied by more than 2 vendors

8.1.11 Systems Analysis And Design ©

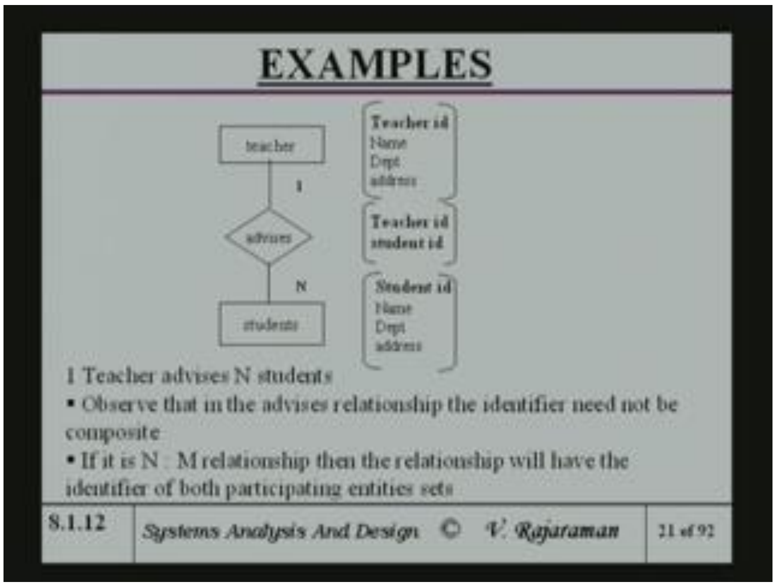
So, this essentially, the fact so if the vendor supplies item, that can be represented by the entity relationship diagram, where the as shown N and M are in the links between vendor and supplies and supplies an item. What really means is that, n and m variable. So, the number of vendors who are, who are supplying the number of items can be variable. Similarly, an item can be supplied by many vendors. And that many is not defined by an actual number. It is variable, it is arbitrary. It can vary from situation to situation.

In our example vendor 8, where vendor that is vendor supplies that that arc, I will put a value n. And between supplies and items in that arc, I will put the number 6. It means, the vendor cannot supply more than 6 items. But, any vendors can supply a given item. So, when n is shown in that vendor that means, number of items can be supplied, which a number of vendors, who can supply a given item is indefinite. But, as far as a vendor is concerned, he is not allowed to supply more than 6 items.

The third example here is that items can be I have put two against vendor and supplies in that arc. And 9 in the arc correcting supplies and items. What basically, means is that item cannot be supplied by more than 2 vendors? But, a vendor can supply any number of items M items. That is given an item any vendor can supply not more than 2 vendors can supply that specific item. But, as far as a vendor is concerned he can supply any number of items.

It can be item cannot be supplied by more than 2 vendors to give an item. But, vendor say this is the in other words many items, the specification really is in terms of the cardinality connecting vendor supplies and supplies items.

(Refer Slide Time: 11:35)



EXAMPLES

teacher
1
advises
N
students
Teacher id
Name
Dept
address
Teacher id
student id
Student id
Name
Dept
address

1 Teacher advises N students

- Observe that in the advises relationship the identifier need not be composite
- If it is N : M relationship then the relationship will have the identifier of both participating entities sets

8.1.12 | *Systems Analysis And Design © V. Rajaraman* | 21 of 92

In this case, another example I put 1 and N that means, a teacher can advice N students. 1 teacher can advice N students. But, a student cannot have more than 1 advisor. So, this is what this relationship is 1 teacher advises N students, observed that advises is relationship ((Refer Time: 12:02)) look at the teacher. Teacher is described by a teacher id, name, department, address. So, a student is student id, name, department, address.

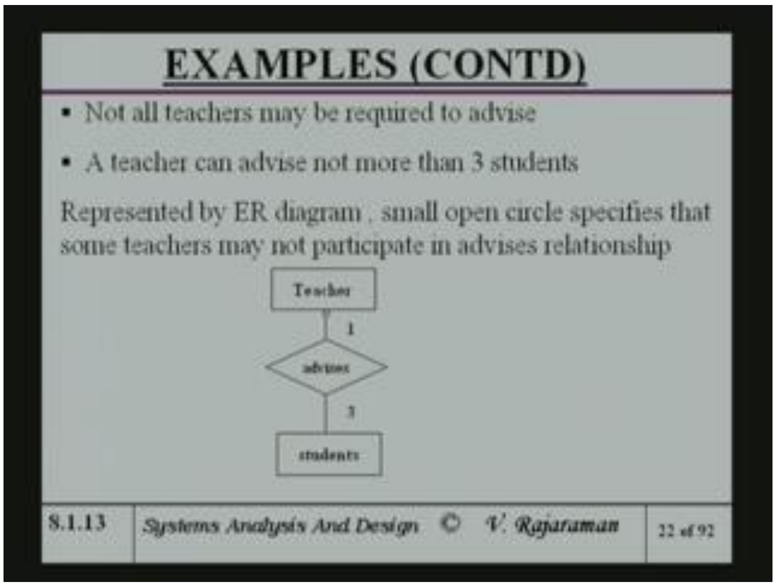
The advises has a key normally, if it is a 1 term relationship. Then the student is not part of the composite key. Because, given the teacher number the student is known. Because, a teacher can in other words for any given student. Given a student, the teacher is known. But, given a teacher a student teacher can actually advise many students. So, in that relationship teacher and student must both appear. But, as far as the key is concerned it is teacher only, not student.

In fact, in this bold face it is shown here, is not really relevant. I have shown that just to indicate the possibility. Suppose, there are situated a stage. And we have something like 2 instead of 1. And having instead of N down below that means, the student can have 2 advisors. If a student can have 2 advisors, but the teacher can advise many students. Then there is a requirement of a composite relationship. Because, given a student I cannot identify who is the teacher who advises, because more than 1 teacher advises a student.

So, teacher student they are given together as a composite key. Then given the teacher number of student number I can essentially find out. The fact that the student has 2 different teachers, who can advise him. They advise him, because depends upon the situation. There are situations, where student may have only 1 advisor in a certain year, say first year. But, he may have multiple advisor in later years depending upon the courses he takes or some other work he does.

So, the point is a composite key becomes relevant, when there is a multiple relationship multi order relationship or cardinality is multiple.

(Refer Slide Time: 14:31)



EXAMPLES (CONTD)

- Not all teachers may be required to advise
- A teacher can advise not more than 3 students

Represented by ER diagram , small open circle specifies that some teachers may not participate in advises relationship

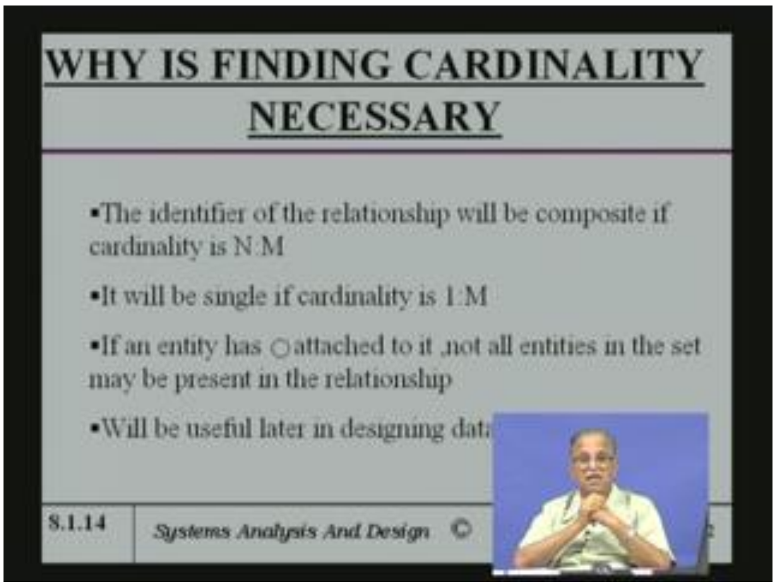
Teacher
1
advises
3
students

8.1.13 Systems Analysis And Design © V. Rajaraman 22 of 92

If in the previous case, we have assumed that every teacher is expected to teach advise one or more students, any number of students. Suppose, there are certain teachers, who do not advise at all. And every given teacher cannot advise more than 3 students. Then I will put a cylindrical open circle below a teacher. It indicates that a teacher, they not be required to teach to advise at all. That means, that 1 is not compulsory. It does not have to ((Refer Time: 15:17)) a teacher can be absent in the certain advises relationships.

And there is student every student must have an advisor. That is ((Refer Time: 15:38)) clear from this diagram. But, then it also indicates that not more than 3 students can be advised by a given teacher. So, that is this is cardinality in this case.

(Refer Slide Time: 15:52)



WHY IS FINDING CARDINALITY NECESSARY

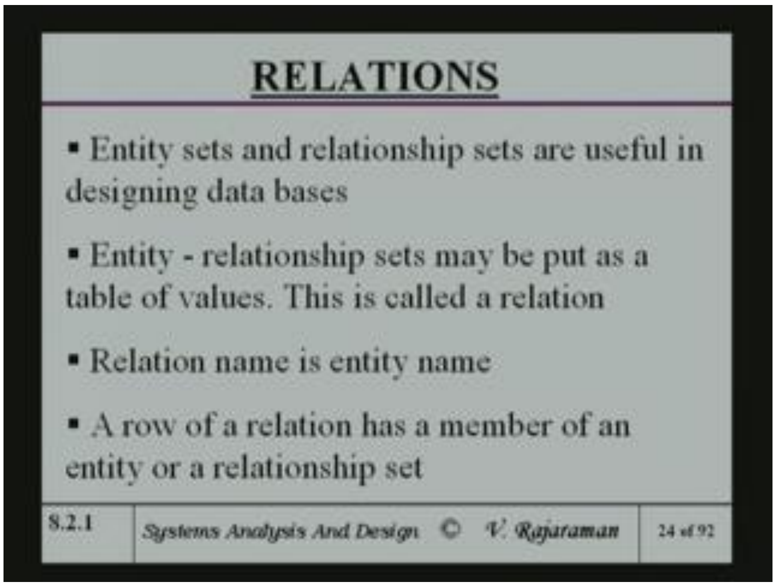
- The identifier of the relationship will be composite if cardinality is N:M
- It will be single if cardinality is 1:M
- If an entity has ○ attached to it ,not all entities in the set may be present in the relationship
- Will be useful later in designing data

8.1.14 Systems Analysis And Design ©

The identifier our relationship to be composite is cardinalities N is to M. It will be simply the cardinality 1 is to M. When a entity a circle attached to it, not all entities in the set be present in the relationship. Because, some teachers in the previous case, ((Refer Time: 16:12)) some teachers do not appear in the advisors relationship. It will be ((Refer Time: 1620)) this idea of cardinality, they will be useful later on in what is known as normalizing database.

At this time of course, it is a little bit we might say, looks more theory than necessary. I think this theory becomes a ground work later on when you talk about the idea of normalization. Because, idea of normalization comes based on the cardinality of relationship.

(Refer Slide Time: 16:54)



RELATIONS

- Entity sets and relationship sets are useful in designing data bases
- Entity - relationship sets may be put as a table of values. This is called a relation
- Relation name is entity name
- A row of a relation has a member of an entity or a relationship set

8.2.1 | Systems Analysis And Design © V. Rajaraman | 24 of 92

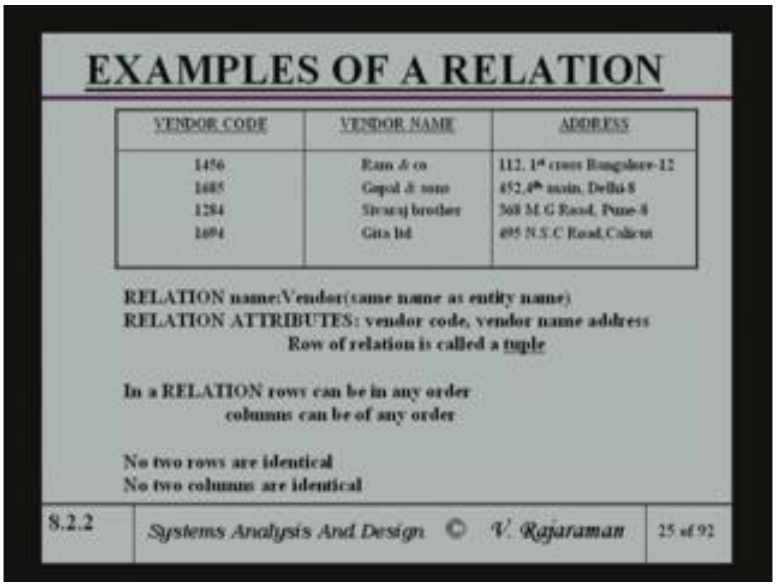
Entity sets and relationship sets are useful in designing databases. That is the reason we talked about entities and relationships. Entities and relationships sets may be put in a table of values. This is called a relation. A relation see there is a lot of mathematical definition of relation and so on. And pure theory of relations as proposed by card the mathematical definition of relations have been used, because the subject called relational calculus, which he proposed which has a number of operators, like say, in differential calculus operators like differentiation integration and so on.

So, similarly relational calculus has certain operations project join and so on. We will not really look at that, in this particular sort of short 3 lectures. It is to give you a broad overview of what relationship, what is the relation. And what is how is the relation important in a database and how is a database management system created using relations. Because, today almost all database management systems are relational database management systems.

As I told that, there are entire courses devoted to database management systems, where the relational calculus at all ground work we will discuss in detail. But, in this course system analysis and design we extend, we are expected to as a system analyst understand, whether relationship is what relation is and what relation database. I think we are giving a bird’s eye view or the theory of relational database.

So, this will give you a very important kind of a grounding and also the motivation, to go out to the later course on database management systems. Because, it is a important course in the computer science curriculum. Entity relationship set may be put in a table of values, as we said this is called a relation. Relation name is an entity name. So, the name of the relation is one of the entities. A row of a relation has a member of an entity or a relationship set.

(Refer Slide Time: 19:30)



EXAMPLES OF A RELATION

VENDOR CODE	VENDOR NAME	ADDRESS
1456	Ram & co	112, 1st cross Bangalore-12
1685	Gopal & sons	452, 4th main, Delhi-8
1284	Sivaraj brother	368 M G Road, Pune-8
1694	Gita ltd	495 N.S.C Road, Calicut

RELATION name:Vendor(same name as entity name)
RELATION ATTRIBUTES: vendor code, vendor name address
Row of relation is called a tuple

In a RELATION rows can be in any order
columns can be of any order

No two rows are identical
No two columns are identical

8.2.2 *Systems Analysis And Design © V. Rajaraman* 25 of 92

Let us, take an example. If I just take a vendor, which is a entity. Vendor has got vendor is the entity name. And the entity name has many attributes. So, the relation is a relation representing vendor. And it has got many attributes. In this case attributes, which are shown are the vendor code. Because, we need a unique identifier for a vendor. And the unique identifier is a vendor code and there is a vendor name, which may not be unique need not be unique and the address.

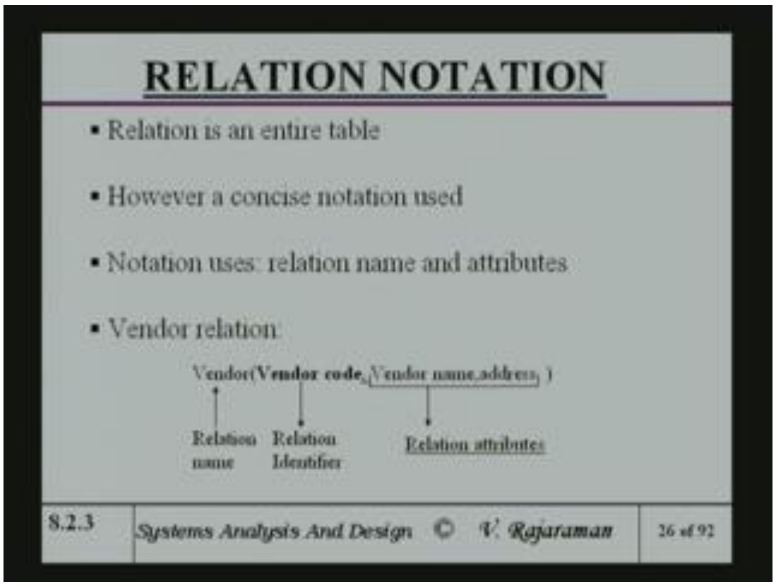
The address normally is unique, but need not be. May be more than one ((Refer Time: 20:21)) same address. That is so as far as we are concerned irrelevant. So, the address is another attributes of the vendor. So, given the vendor code, I will be able to find out the vendor name and the vendor address. So, the relation is shown in a short hand form as vendor as the relation name. And within parenthesis the attributes namely, this case attributes are vendor code vendor name and vendor address.

And this row of a relation is called tuple. Because, it cut tuple end generally, when you say end tuple that means, there is a you might say let it as a vector return component. So, if you look at a row as a vector, in a data structure point of view. It is it has got 3 components. In this case, 3 attributes, so row relation is called a tuple. In a relational a row can be of any order. There are rows, there is no particular ordering of the rows, which is imposed by the definition of relation.

Similarly, columns can be of any order. So, this will also, this is also allowed. But normally, the unique identification of the relation namely in this case, vendor code is normally put as the first attribute. And the others follow, so normally we first attribute is always the unique identifier. The other attributes can be of any order. That is normally a better way of doing it practically. Even though in theory they can be in any order. Not to be arrays can be identical. Because, that means it is duplicated data, obvious.

Now, 2 tuples can be identical. Then it is wrong net. No 2 columns can be identical, because there are different attributes, which describe the same entity. So, you cannot have attributes actually, duplicated. See, same attribute cannot be more than one column. These are ((Refer Time: 22:56)) simply common sense reasons.

(Refer Slide Time: 23:00)



RELATION NOTATION

- Relation is an entire table
- However a concise notation used
- Notation uses: relation name and attributes
- Vendor relation:

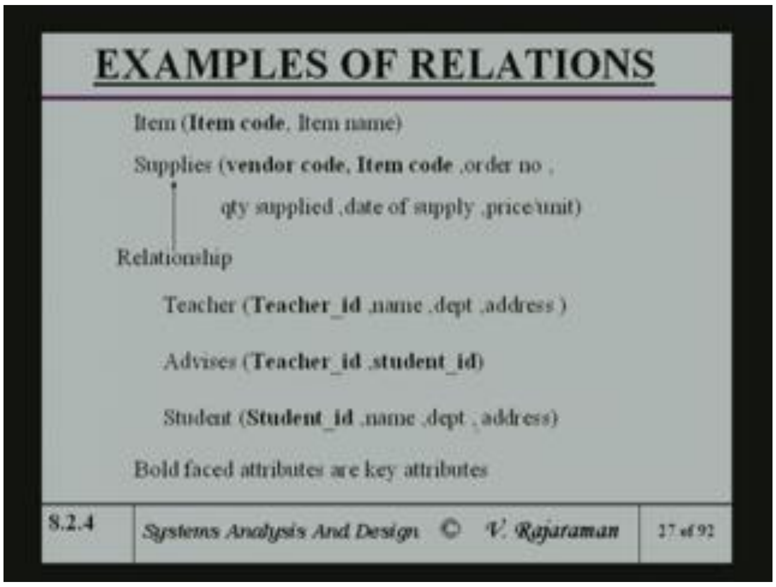
Vendor(**Vendor code**, Vendor name,address,)

Relation name
Relation Identifier
Relation attributes

8.2.3 | *Systems Analysis And Design © V. Rajaraman* | 26 of 92

So, relation is an entire table. But, we use a concise notation as I said, relation name attributes. So, relation name is vendor. And relation identifier is vendor code that is the identifier is simply a key. Identifier is a term key, an identifier has synonymous. But normally, the term identifier relation identifier is more accurate terminology to use and relation attributes.

(Refer Slide Time: 23:36)



EXAMPLES OF RELATIONS

Item (**Item code**, Item name)

Supplies (**vendor code, Item code** ,order no ,
qty supplied ,date of supply ,price/unit)

Relationship

Teacher (**Teacher_id** ,name ,dept ,address)

Advises (**Teacher_id** ,**student_id**)

Student (**Student_id** ,name ,dept , address)

Bold faced attributes are key attributes

8.2.4 | *Systems Analysis And Design* © *V. Rajaraman* | 27 of 92

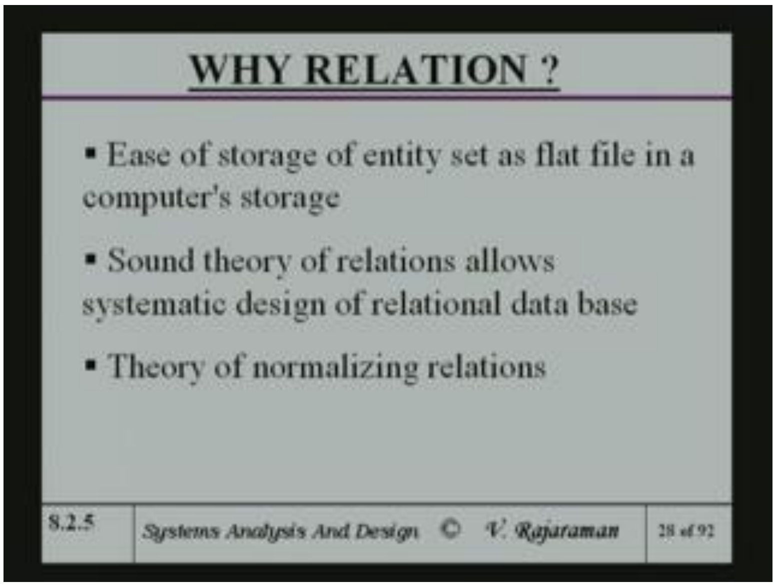
Some more example, if you take item, item code and item name is an entity. And relationship is supplies. Supplies, because the item code is connected to vendor code. And there is a multiple that is cardinality is N is to 1. In this supplies relationship, where vendor code and item code appear in the composite identifier. And in the order number, quantity supplied date of supply and price per unit. There is a suppliers relationship. Because, the vendor the reason, why the composite term appears also is that the vendor and item together can be determined.

The order number and also the quantity supplied date of supply and price per unit. Because, by itself vendor or item they will not have any of this, see if you look at the vendor relation, the vendor relation will have vendor code, vendor name and vendor address. Whereas, the quantity supplied order number and so on. Is not there in either the item relation or the vendor relation. Supply is the one, this connects these two and other attributes, which are important as far as supply is concerned you split that.

Another reason may be that because these vendor code and item code appear together. Because, price per unit may depend, may be depended on the vendor. It may be in different prices, different vendors for whatever reason. May be one vendors quality is better. And obviously, the quantity which is supplied will depend upon the vendor and item together. So, that is the reason why composite thing is required. Whereas, in the case of a teacher advises ((Refer Time: 25:59)) if multiple teachers are there for advising multiple students then teacher id and student id will occur together in advises.

Otherwise, student id is only an attribute. And teacher id is the identifier. Because, given the student id, you can always find out, which teacher advises that student. So, the bold faced are the key attributes.

(Refer Slide Time: 26:28)



WHY RELATION ?

- Ease of storage of entity set as flat file in a computer's storage
- Sound theory of relations allows systematic design of relational data base
- Theory of normalizing relations

8.2.5 | Systems Analysis And Design © V. Rajaraman | 28 of 92

So, why do we use relationships, relation in general. So, relation as we said is nothing but, a table. I would call as a flat file in the computer storage. Anyway, in the computer storage, see as far as the theory of database is concerned. When, you look at the physical storage of or that logical database. So, now we are looking at the logical database design. The physical storage is based on, whatever device you are going to use for storage. And the attributes are the physical storage, change from time to time to time.

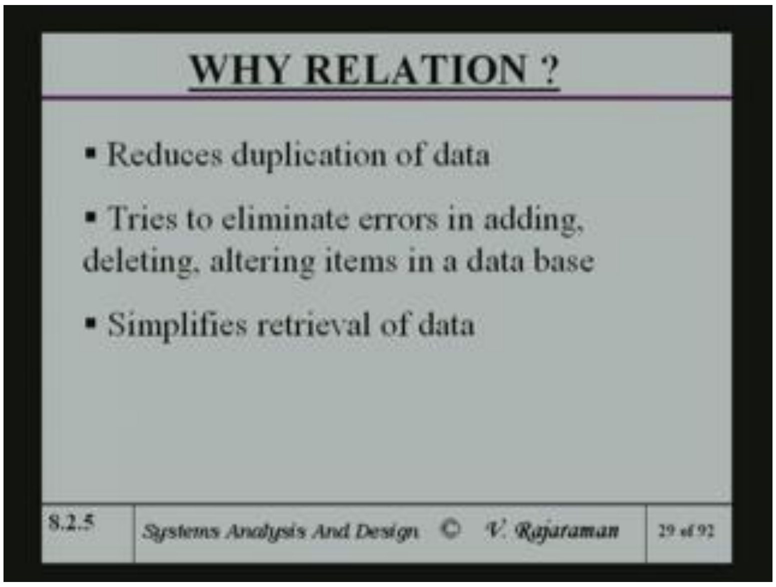
Like for instance today, this I was having characteristics. Few days from today, the characteristics of disk by change. So, mapping from the logical database to the physical device. This data is dependent and it will change whereas, in the design the logical part is going to become invariant. So, the logical part is important. It is once you come up with a logical design, then later on DBMS is more like a mapping method of mapping you might say. And that mapping may change with time.

I will look at that later on in the in this course this mapping idea. But, this time we are more or less concerned about the logic of relationships. Some theory of relations, allow systematic design of relational database at the logical level. So, there is a theory of normalizing relations. That is the question is the normalizing relations is an important corner stone or a foundation of theory of relations. And that is very important point in logical design of databases.

Why do we need a relation? When you use a relation it reduces duplication of data. See the actually, when you talk about relation in general. We do the in a flat file and flat file is something where, when you talk about a flat file. What it really means is that every card in the file has a same property in terms of length. So, the records all are of equal length and they all are stored one after the other. If it is a sequential file, it will be in some sequential order of the key.

If it is any kind of a direct cuts file, it may be get scattered all over the physical device. But, the identifier of course, when you have as the key. And if it is scattered all over the disk, that is a key to address transformation, you one can use and given the key can always find the address of direct cuts sort of a device like disk and access it. So, once you access, you are accessing a fixed card size, that is what you mean by a flat file.

(Refer Slide Time: 30:10)



WHY RELATION ?

- Reduces duplication of data
- Tries to eliminate errors in adding, deleting, altering items in a data base
- Simplifies retrieval of data

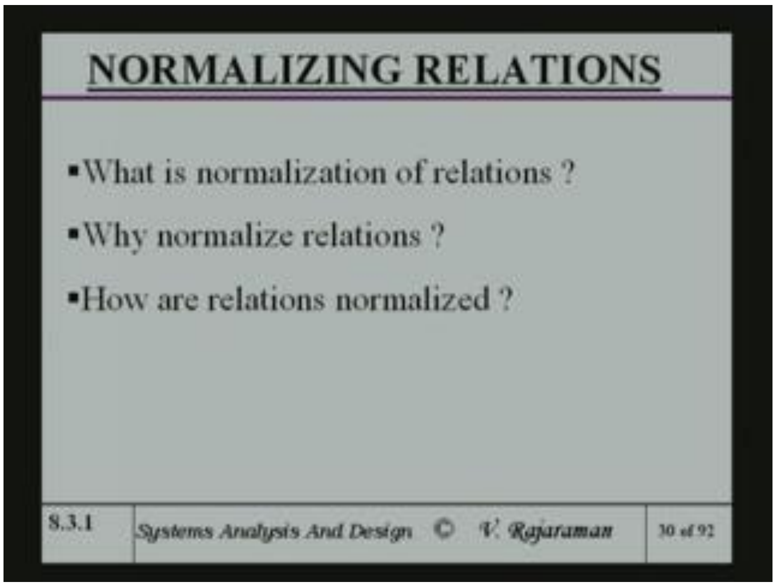
8.2.5 | Systems Analysis And Design © V. Rajaraman | 29 of 92

And the use of relation database and this normalization, which we will talk about, reduces duplication of data. Tries to eliminate errors in adding, deleting, altering items in a database. Because, one of the most important thing in a database is that no database is static. Because, suppose you are talking looking at a certain data of students in a college. Students may leave the college in the middle of the term or students may be joining the college year after year. Or students may be actually changing their address.

So, the addition and deletion are important. Altering that is in this I have changed the address. I have to change the address at every one place not at multiple places. Because, for changing the address in multiple places, we have always a possibility of error. So, eliminate errors in adding, deleting and altering database. In fact, one of the most difficult or common problems encountered earlier was that when there is not sound theory of databases. This inconsistency always arose in a file based system.

Because, there are multiple files and we update some file and do not update some other file and you get great difficulty. Because, the same student in different files may have different addresses. So, that is kind of thing which, you want to completely eliminate in terms of a relational database. Simplifies retrieval of data.

(Refer Slide Time: 32:00)



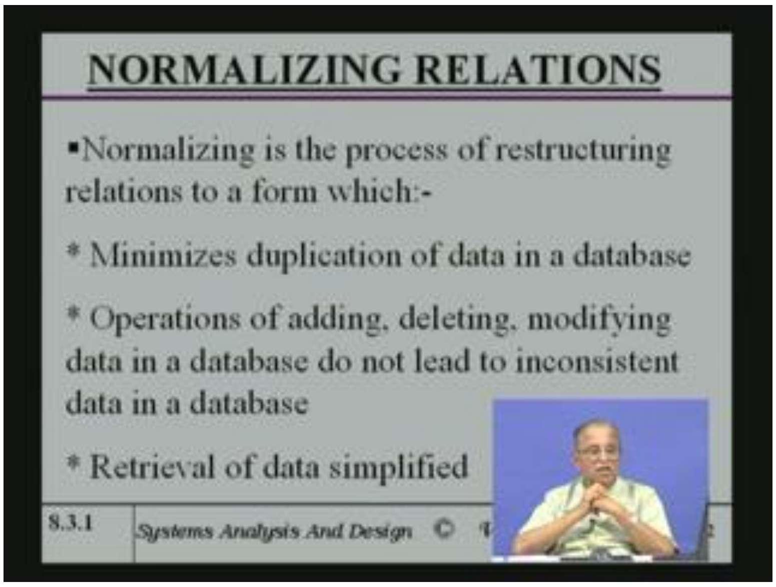
NORMALIZING RELATIONS

- What is normalization of relations ?
- Why normalize relations ?
- How are relations normalized ?

8.3.1 | Systems Analysis And Design © V. Rajaraman | 30 of 92

So, we will all this is done by what is known as normalizations. And we have to first define, what is normalization of relations. Why do we normalize relations? And how are we going to normalize relations. Because, we start with what is normalization that will describe why it is important. Because, very often students know how to normalize. We have to understanding, why you should normalize. And what is normalization is straight forward.

(Refer Slide Time: 32:41)



NORMALIZING RELATIONS

▪Normalizing is the process of restructuring relations to a form which:-

* Minimizes duplication of data in a database

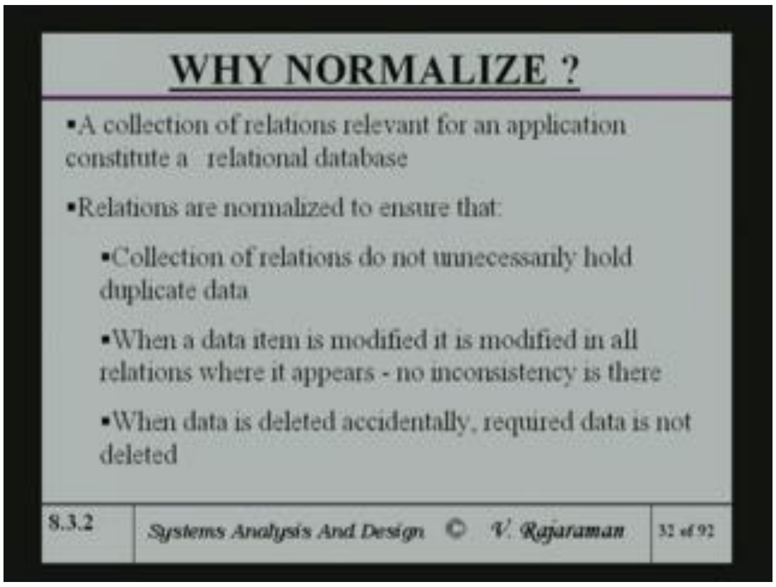
* Operations of adding, deleting, modifying data in a database do not lead to inconsistent data in a database

* Retrieval of data simplified

8.3.1 *Systems Analysis And Design* ©

Normalizing is a process of restructuring relation to a form, which minimizes duplication of data in a database. The reason why we normalize actually, the normalization is the process of restructuring, which minimize with the aim of minimizing duplication of data in a database. Operation of adding, deleting, modifying data in a database do not lead to inconsistent data in the database and retrieval of data is simplified. This ((Refer Time: 33:19)) point I made last time also.

(Refer Slide Time: 33:20)



WHY NORMALIZE ?

- A collection of relations relevant for an application constitute a relational database
- Relations are normalized to ensure that:
  - Collection of relations do not unnecessarily hold duplicate data
  - When a data item is modified it is modified in all relations where it appears - no inconsistency is there
  - When data is deleted accidentally, required data is not deleted

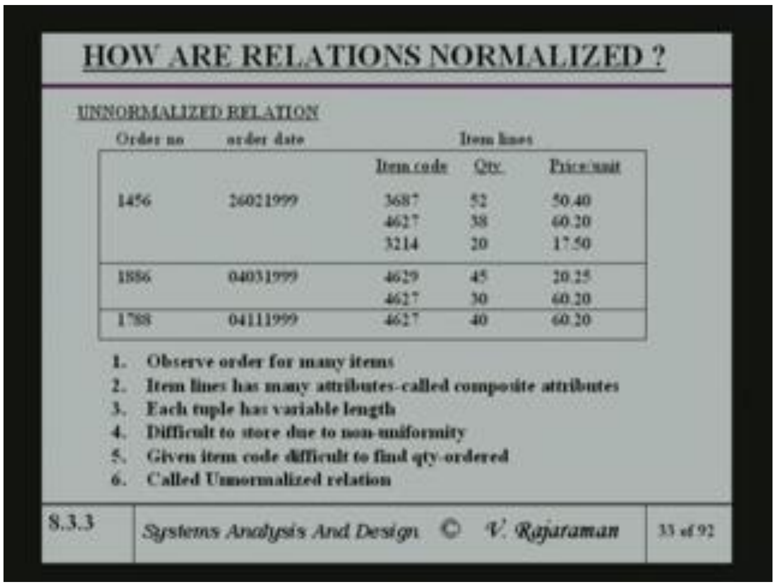
8.3.2 | Systems Analysis And Design © V. Rajaraman | 32 of 92

A collection of relations relevant for an application constitute a relational database. And relations are normalized to ensure that collection of relations do not unnecessarily hold duplicate data. Because, duplicate data can be there are multiple relations. And same data's replicate in multiple relations. Then, you will have an update problem. Whenever, you update one relation, you should remember to update all the other relations, where this data appears. Otherwise, you will have inconsistency.

When a data item is modified it is modified in all relations, where it appears. No inconsistency is there. When data is deleted accidentally, required data is not deleted. That is a in fact, I should say that sometimes data is not deleted accidentally. When, data is deleted in fact, I should specify it much more correctly. When data is deleted, required data is not deleted accidentally. In other words, the point is when a particular data is deleted. Some other data is not accidentally deleted. This is what is, what is implied by this sentence.

In fact, this sentence is not properly formed. When data is deleted required data is not deleted accidentally, somewhere, somewhere else ((Refer Time: 34:46)). So, that is what we want to ensure. I will start with a process of normalization starts with an un normalized relation.

(Refer Slide Time: 35:00)



HOW ARE RELATIONS NORMALIZED ?

UNNORMALIZED RELATION

Order no	order date	Item lines		
		Item code	Qty	Price/unit
1456	26021999	3687	52	50.40
		4627	38	60.20
		3214	20	17.50
1886	04031999	4629	45	20.25
		4627	30	60.20
1788	04111999	4627	40	60.20

1. Observe order for many items
2. Item lines has many attributes-called composite attributes
3. Each tuple has variable length
4. Difficult to store due to non-uniformity
5. Given item code difficult to find qty-ordered
6. Called Unnormalized relation

8.3.3 Systems Analysis And Design © V. Rajaraman 33 of 92

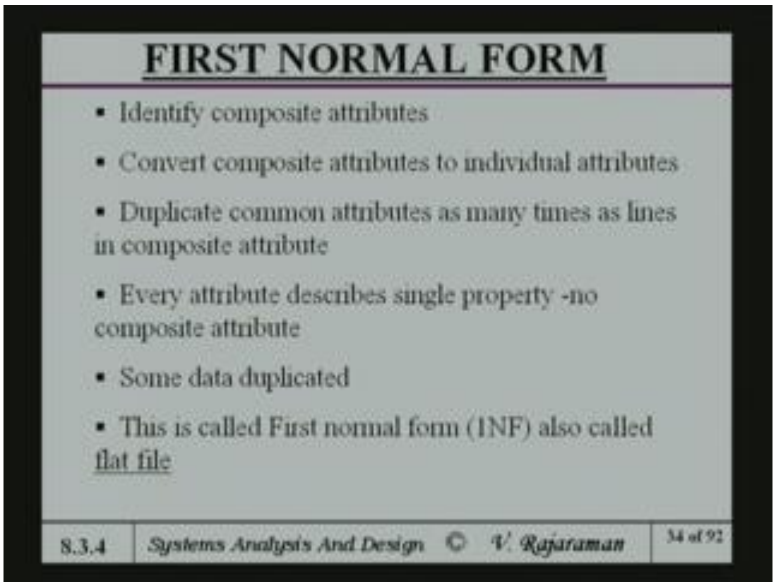
Like in this case, an order number, an order date is there. And the number of order is placed for number of items. And so multiple items the order may contain more than one item. In this case, I will look at the, who is going to supply the item. I will just look at a un normalized relation of order, what is an order. And in this case, order has got a date and certain number of items for which the order has been placed. And observe, that an order can have many items.

So, by looking at the file each record and file has got variable lengths. Because, in some records, because order number and order date as common. And item lines is variable. So, items lines is variable. Some record may be long, some records may be short. So, it is the variable length file, which is always very difficult to handle in any particular situation. That is one problem. Items lines has many attributes called composite attributes. Each tuple has variable length. Difficult to store date in non uniformity, as we said record is a variable length record.

And given item code is difficult to find quantity ordered. That is given the item code like in this case, you have to just go through the entire relation from top to bottom like suppose, 467 I want to find out, how many items of 467 are ordered. Then, first second line it is 38, then the fifth line it is 30. Sixth line it is 40 and add 40 plus 30 plus 38. So, essentially now it is first I have to all the entire, set of all the lines. And effectively sets with the entire file you might say.

If you are really going to find out, how many items of a particular type is been ordered. So, this is not a very satisfactory kind of a way of representing logically the data. So, it is called un normalized relation. So, I cannot first in fact, I should really start with something which is not normalized. In other words it is not really a relation at all. This ends up the definition of relation. Because, relation in a flat file. So, I should change it identify composite attributes.

(Refer Slide Time: 37:52)



FIRST NORMAL FORM

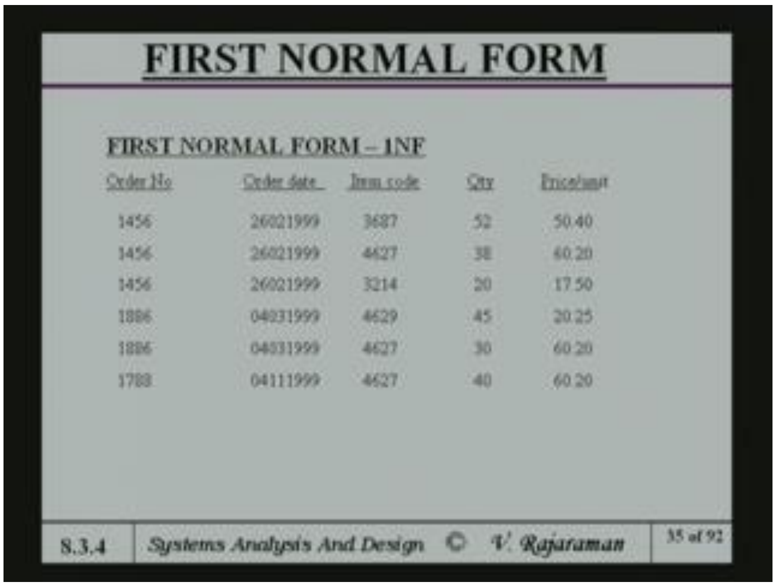
- Identify composite attributes
- Convert composite attributes to individual attributes
- Duplicate common attributes as many times as lines in composite attribute
- Every attribute describes single property -no composite attribute
- Some data duplicated
- This is called First normal form (1NF) also called <u>flat file</u>

8.3.4 | *Systems Analysis And Design* © *V. Rajaraman* | 34 of 92

Convert all composite attributes to individual attributes. Duplicate common attributes as many times as lines in composite attribute. Every attribute describes a single property a no composite attribute representation. In other words, the point is if you look at this ((Refer Time: 38:13)) there are variable length. I should make all of the lengths, equal ((Refer Time: 38:18)) duplicating if necessary. Every attribute describes single property no composite attributes. Some data may be duplicated. This is called first normal form or called flat file.

Nowadays, any relation I start with the first normal form. First normal form is the you might say 0 state or beginning state.

(Refer Slide Time: 38:42)



FIRST NORMAL FORM

FIRST NORMAL FORM – 1NF

Order No	Order date	Item code	Qty	Price/unit
1456	26021999	3687	52	50.40
1456	26021999	4627	38	60.20
1456	26021999	3214	20	17.50
1886	04031999	4629	45	20.25
1886	04031999	4627	30	60.20
1788	04111999	4627	40	60.20

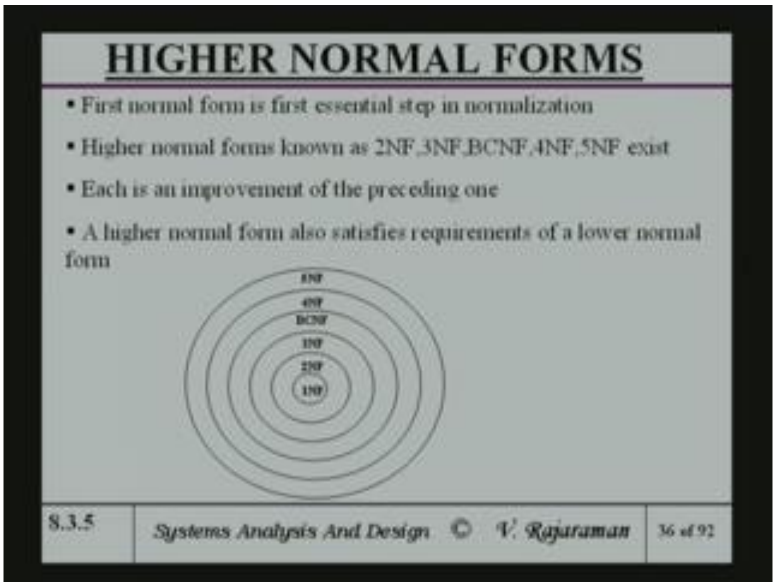
8.3.4 Systems Analysis And Design © V. Rajaraman 35 of 92

So, I have taken the same 1 and if I take this ((Refer Time: 38:51)) 1456 there are 3 items of which are ordered in this 1456. So, I duplicate that 1456 order date, item code, quantity price per unit. I am duplicating. And so, I got a set of lines, where each tuple has got a length of 1 2 3 4 5. So, there are 5 tuple. Suppose, ending each line in the relation and there are in this case, set of tuples. There are 3 plus 3 6, 6 tuples to represent this first normal form relation.

The first normal form or you can see a lots of duplication is there. And duplication creates a lot of problems. And so, we normally for instance in this case suppose order number 1456 is cancelled for item 3687. That is I do not want to cancel 1456 completely. But, only for the item 3687. Then I have to go through the entire file to match 1456 and 3687 and then you need to kind of delete that row. Otherwise, if I delete 1456 all 3 first lines will be deleted. Right other items, which may I still like to have in that order will be gone.

So, this is not a good idea or if suppose the price is changed for an item. I got to go through all the item codes. And like in the 467 the price changes 60 20 to 58 20 reduces. Then I got to go through the entire end tuples last end tuples and then change wherever it appears.

(Refer Slide Time: 41:12)


5NF
4NF
BCNF
3NF
2NF
1NF

HIGHER NORMAL FORMS

- First normal form is first essential step in normalization
- Higher normal forms known as 2NF,3NF,BCNF,4NF,5NF exist
- Each is an improvement of the preceding one
- A higher normal form also satisfies requirements of a lower normal form

8.3.5 | Systems Analysis And Design © V. Rajaraman | 36 of 92

It will not be a very nice thing to do. Very it is not a convenient thing to do. I would like to have straight forward convenient words ways of doing it. So, that I do not have to do a lot of searching. While doing any deletion and not only that I should not delete accidentally. So normally, we start with first normal form. And we go to higher normal forms. Second normal form, third normal form and there is something called Backus Boyce codd not Backus. Backus is a personal actually invented photran, this is Boyce. Similar person called Boyce.

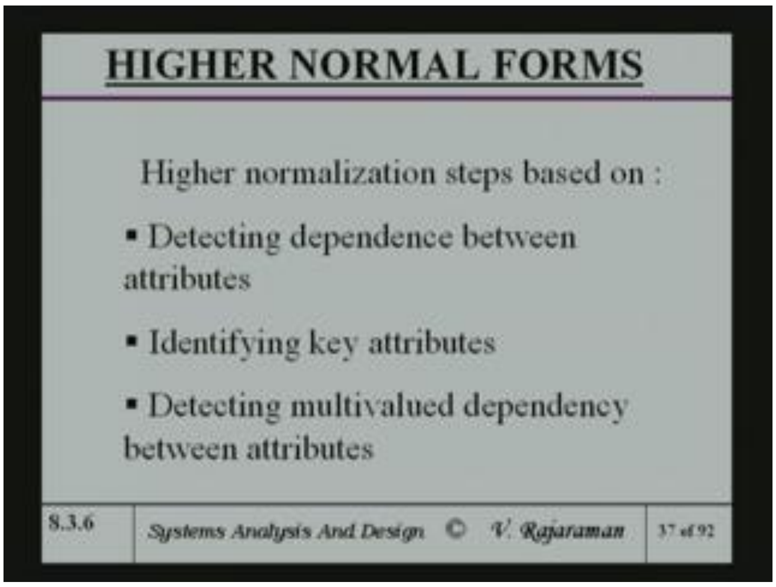
Boyce codd normal form and there is a fourth normal form and fifth normal form. There are normal forms seem to be coming out all the time. Some person finds out little problem with the some normal form. And then tries to come up with a straightly better situation in certain pathological cases. And what we mean by pathological cases or cases which are rare they are an, occur all the time. But still, there is a theory which exists. And by and large, I think in practice up to 5NF but satisfactory for both situations.

But of course, I should not generalize, because after all researchers have gone to higher normal forms for good reasons, this is not trivial reasons. Because, they found some multiple problems, when dealing with relational databases. So. In fact, you may find, that 5 NF is not sufficient. Only, I have to go to 6NF and almost got enough. But, I use to start with 5NF. And any case I am not going to concentrate on the at the base NF. First normal form is first essential step.

And each improvement, each improvement is a preceding one. And higher normal form also satisfies the requirement for lower normal form. Unless something like in first normal form, I cannot start in fact, and go on to second normal form. You might in fact, this figure may somewhat be misleading, because 1NF is a much bigger set. And 5NF is subset of 1NF. And 3NF is another subset of that and 4NF is a further subset. And in other words, in fact, I want it in a set theoretic fashion of contained in.

Then I should really put first NF and make everything else contained in that 1NF. But, here I am talking about requirement specially. That is 5NF has to be 4NF, has to be BCNF and has to be 1NF to begin with.

(Refer Slide Time: 44:34)



HIGHER NORMAL FORMS

Higher normalization steps based on :

- Detecting dependence between attributes
- Identifying key attributes
- Detecting multivalued dependency between attributes

8.3.6 | Systems Analysis And Design © V. Rajaraman | 37 of 92

Higher normalization steps are based on detecting dependence between attributes. Dependency is a very, very important point, when it comes to normalization. So, dependency between any, any attributes any two attributes. There is no algorithm to find out dependency. Dependence which we found out by common sense or the sense of the problem, the statement of the problem what the dependency really means. And so the process of normalization is not entirely algorithmic. What I mean by that is, it is not possible given the requirement facilitation in a word statement.

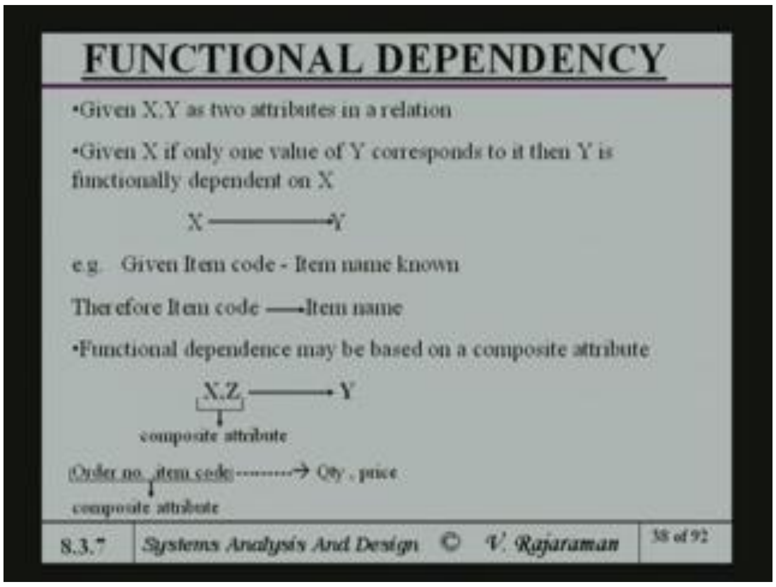
You have a computer program. Create a normalized database for you, without any human intervention. That will be very nice if it happens. But, then if it happens it will put a lot of business see, after all when you learn all these things human brain. And use of our of our language ideas and idea of common relationships and so on. Are the ones which are useful, in describing the normalization. And so, the entire normalization process is more of a normal process. And that is the reason why, it is both challenging and also error prone.

Challenging because you have to make in such a way or you normalize in such a way, that no one else creep in. And you have to understand the problem fairly well. So, good system analyst is able to normalize a relation of database in a good way. So, that no problem of what is the inconsistency in updation and incorrect deletion or incorrect additions take place in the database. So, normalization is based on detecting the dependence between attributes.

Identifying what are the key attributes. Detecting multivalued dependency between attributes. Now, In fact we find out, that there is something called a relational cardinality. So, between relations there are multiple multivalued connections like, we said the multiple teachers can be teaching multiple courses. That is one teacher may teach many courses. And one course may be taught by many teachers. In some places there are multiple sections of a course, the same course may be taught by a one more teacher.

So, there is a multivalued dependency. If on the other hand if, one course can be taught by only one teacher then it is a single valued dependency. where given the value of teacher, I know given the course, I know which teacher teaches that. And a teacher may teach many courses see. So, this is a kind of what is meant by dependency.

(Refer Slide Time: 48:16)



FUNCTIONAL DEPENDENCY

•Given X,Y as two attributes in a relation

•Given X if only one value of Y corresponds to it then Y is functionally dependent on X

$$X \longrightarrow Y$$

e.g. Given Item code - Item name known

Therefore Item code ⟶ Item name

•Functional dependence may be based on a composite attribute

$$\underbrace{X,Z}_{\text{composite attribute}} \longrightarrow Y$$

$\underbrace{\text{(Order no. , item code)}}_{\text{composite attribute}} \dashrightarrow$ Qty , price

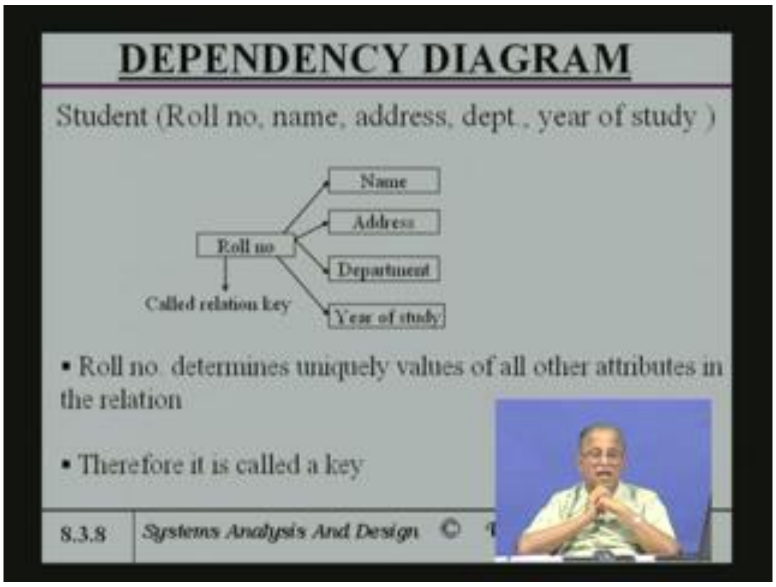
8.3.7 | Systems Analysis And Design © V. Rajaraman | 38 of 92

Given X and Y are two attributes in a relation. And for each value of X, there is only one value of Y to the corresponding value. Then we say it is Y is functional dependent on X. Now, it is given X, you can find out Y. It is like saying Y equivalent of X. What is Y equal to f X mean, given the value of X and given the function, you can calculate X and find out Y. So, Y is dependent on X. So, it is somewhat similar to that. Given the item code, item name is known. And so this is called a dependency.

If the name depends upon the code. The item code is completely dependent on item. In other words, given X the that value of Y correspond to it. Y is functionally dependent. So, item name and item code are functionally dependent. And functionally dependent based on composite attributes or in other item code together very composite attribute. Had to be given to find out the quantity ordered and the price. So, in case of dependency, there if dependency based on composite attribute or a single attribute.

Based on single key there is one type of relationship. Based on composite key there is a different type of relationship. This distinction becomes important in the normalization as we look in future.

(Refer Slide Time: 50:13)



DEPENDENCY DIAGRAM

Student (Roll no, name, address, dept., year of study)

$$\begin{tikzcd}
 & \text{Name} \\
 & \text{Address} \\
\text{Roll no} \arrow[ur] \arrow[uur] \arrow[r] \arrow[dr] \arrow[d] & \text{Department} \\
\text{Called relation key} & \text{Year of study}
\end{tikzcd}$$

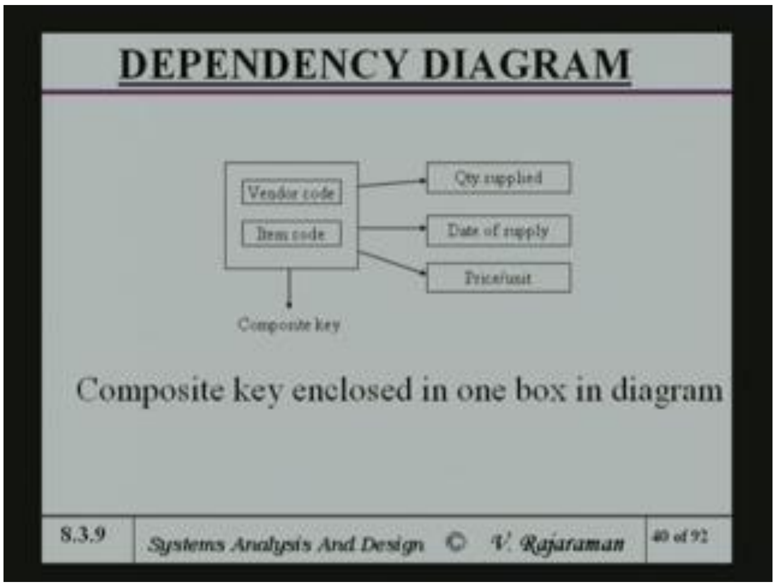
- Roll no. determines uniquely values of all other attributes in the relation
- Therefore it is called a key

8.3.8 Systems Analysis And Design ©

So, normally dependence is shown on what is known as a dependency diagram. So, I have got a in fact, diagrams are very nice to have. Because, as picture gives you lot more intuitive understanding. So, in this case, I have given a student relation and the student relation has got a identifier or a key, which is roll number. And a roll number given the roll number, there are a number of attributes like name, address, department, year of study. They are functionally dependent on the roll number. So, roll number is there for a key.

In other words, given the roll number I have to always find out the name, address, department, year of study. So, roll number is all the others, they are functionally dependent. Roll number and name are functionally dependent. And given the roll number I can always find out the name.

(Refer Slide Time: 51:32)



DEPENDENCY DIAGRAM

Vendor code
Item code
Qty supplied
Date of supply
Price/unit
Composite key

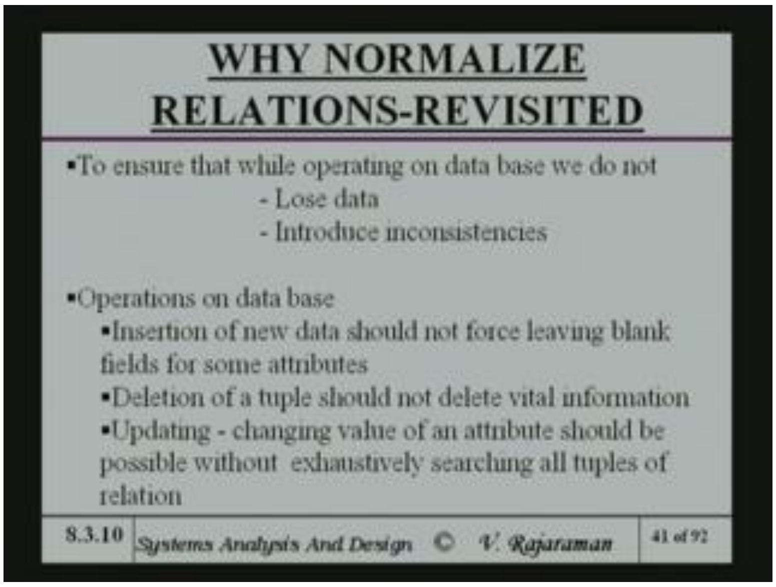
Composite key enclosed in one box in diagram

8.3.9 Systems Analysis And Design © V. Rajaraman 40 of 92

Composite key is enclosed in a in one box in the dependency diagram. So, in this case, I have got vendor code and item code together it is a composite attribute. And the diagram shows the quantity supplied the date of supply and price per unit depend upon the both vendor code, item code taken together. So, composite attribute is always enclosed in a box. And I am in this by implication that means, that the none of the quantity supplied date of supply or price per unit is dependent directly, on either by item or vendor code alone.

And both the vendor code and item code are given together as a composite key in order to find out the other the quantity supplied date of supply and price per unit are dependent on the composite attribute vendor code and item code.

(Refer Slide Time: 52:42)



WHY NORMALIZE RELATIONS-REVISITED

- To ensure that while operating on data base we do not
  - Lose data
  - Introduce inconsistencies

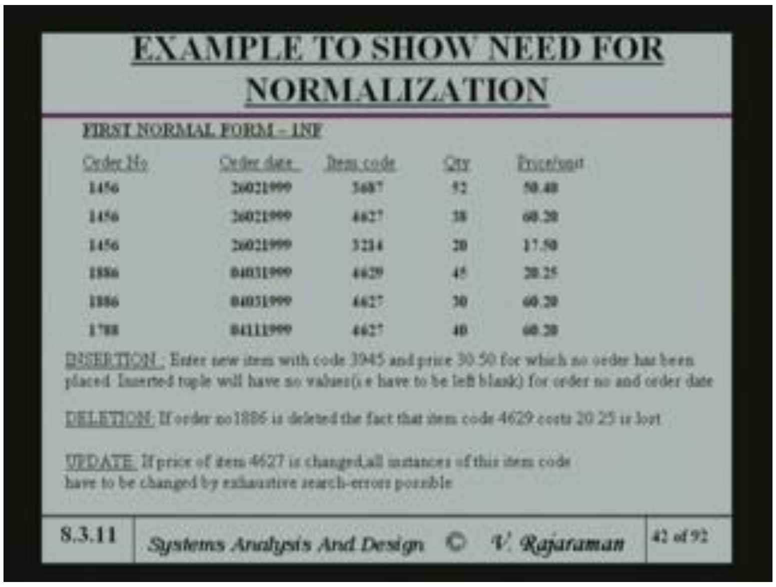
- Operations on data base
  - Insertion of new data should not force leaving blank fields for some attributes
  - Deletion of a tuple should not delete vital information
  - Updating - changing value of an attribute should be possible without exhaustively searching all tuples of relation

8.3.10 | Systems Analysis And Design © V. Rajaraman | 41 of 92

Next, why normalize relations revisited in other words. I am repeating, what I said earlier. To ensure that while operating on database, we do not lose data, we introduce inconsistencies. While doing any kind of a operation on database. Operation on database, are insertion of new data should not force leaving black fields for some attributes. In other words, when you enter a data item, see by definition a relation either attributes. Every attribute has got a value.

So, I should not get to a situation where, some attribute values are blank. Deletion of a tuple should not delete vital information. Obviously, I mean some vital information will not be lost, while deleting some item. Updating or changing value of an attribute should be possible without exhaustively searching all tuples.

(Refer Slide Time: 53:54)



EXAMPLE TO SHOW NEED FOR NORMALIZATION

FIRST NORMAL FORM – 1NF

Order No	Order date	Item code	Qty	Price/unit
1456	26021999	3687	52	50.40
1456	26021999	4627	38	60.20
1456	26021999	3214	20	17.50
1886	04031999	4629	45	20.25
1886	04031999	4627	30	60.20
1788	04111999	4627	40	60.20

INSERTION : Enter new item with code 3945 and price 30.50 for which no order has been placed. Inserted tuple will have no values(i.e have to be left blank) for order no and order date

DELETION: If order no1886 is deleted the fact that item code 4629 costs 20.25 is lost

UPDATE: If price of item 4627 is changed,all instances of this item code have to be changed by exhaustive search-errors possible

8.3.11 | Systems Analysis And Design © V. Rajaraman | 42 of 92

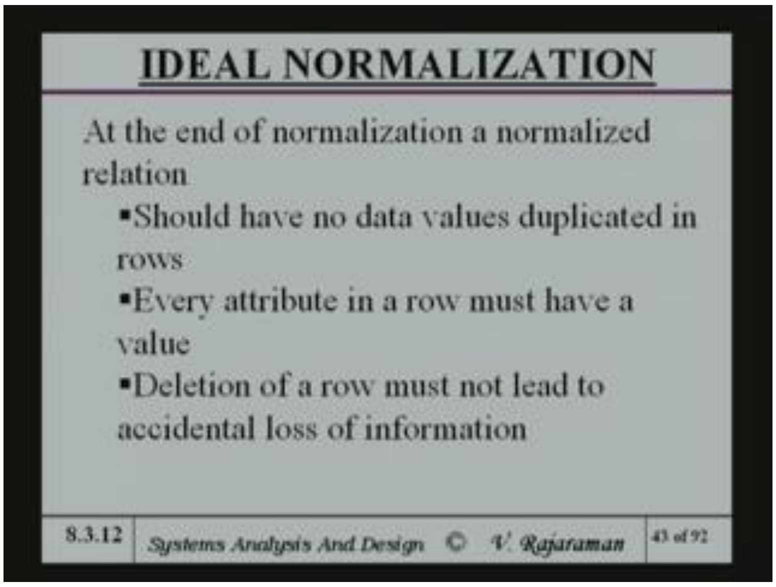
We saw that earlier, that is that if I do not in the first normal form I have a problem. See, insertion vendor and item code. Suppose, we want to in the first normal form enter or insert a new item code with a price of 30 rupees 50 paisa, for which no order has been placed. Then you may have negotiated certain price and item with a vendor ((Refer Time: 54:22)) that we have not yet placed the order. When we insert that item with code 3945 and price 30 50, there will not be any entry for quantity and there will not be any entry for order date and there will not be any entry for order number.

So, it is obviously, a situation which is not acceptable. And if we have to delete order number 1886 to be deleted. The fact that item code 4629 costs 20 25 is lost. In other words, if you look at 1886 here, 1886 now delete. 1886 occurs in two places. So, repeat take out 1886 the fact that 4629 the item code 4629. See, I can just look at that. 4629 see, costs 20 25 will be lost, because I am deleting 1886. Also, I am deleting this.

Unfortunately, 4627 occurs somewhere else. And so this is not deleted. But, this 20 25 is deleted. And that information is not found anywhere else ((Refer Time: 55:49)). Price item 4627 is changed. In fact, I told you earlier. Look at all instances of item code, have to be changed. So, I will look at all 4627 is occurring to search the entire 1NF relation to be able to find out all the ((Refer Time: 56:12)) what you have to change. And without out any by accident I should essentially change all of them. Otherwise there will be inconsistency.

Somewhere, the price will be 60 rupees 20 paisa still old price. And somewhere the price will be newer price. So, it will not, will be inconsistent database.

(Refer Slide Time: 56:36)



IDEAL NORMALIZATION

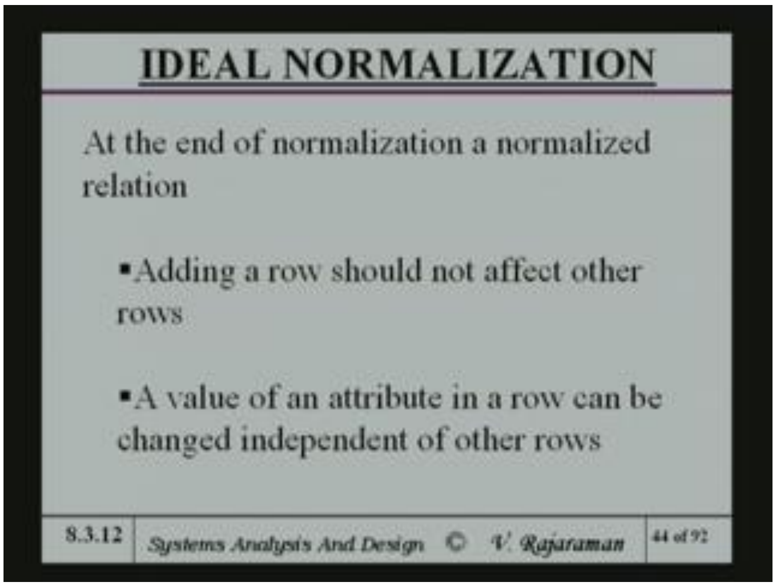
At the end of normalization a normalized relation

- Should have no data values duplicated in rows
- Every attribute in a row must have a value
- Deletion of a row must not lead to accidental loss of information

8.3.12 | Systems Analysis And Design © V. Rajaraman | 43 of 92

At the end of normalization a ideal normalization is 1. At the end of the normalization should have no data values duplicated in rows. In this case of, ((Refer Time: 56:52)) I have got duplicated values like 1456 this duplicated, so many times. Every attribute in a row must have a value. No more blanks are allowed. Deletion of a row must not lead to accidental loss of information. See, deletion in this case, I have showed with example, of deleting one particular order then the price is lost. So, deletion must not lead to accidental loss.

(Refer Slide Time: 57:23)



IDEAL NORMALIZATION

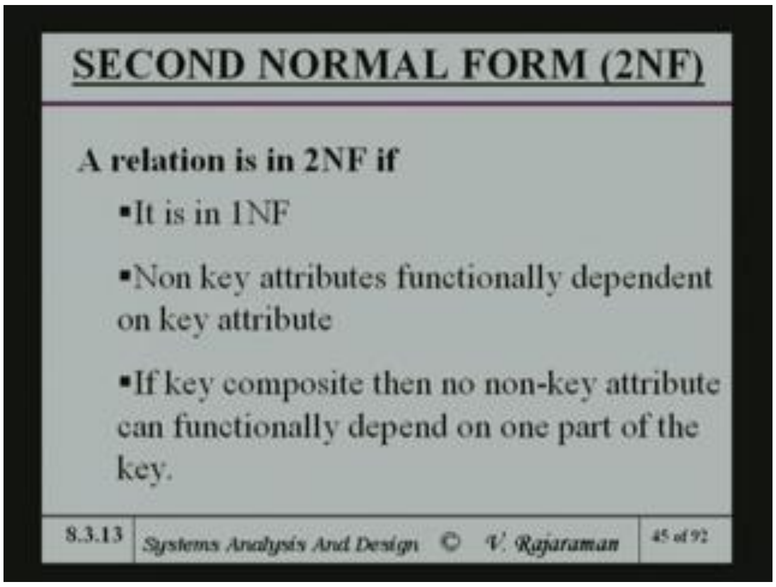
At the end of normalization a normalized relation

- Adding a row should not affect other rows
- A value of an attribute in a row can be changed independent of other rows

8.3.12 | Systems Analysis And Design © V. Rajaraman | 44 of 92

At the end of normalization, a normalized relation adding a new row should not affect other rows. I should be able to add new rows. And the value of an attribute in a row can be changed independent of other rows. And so that is normalized. In other words, the point really is we have to eliminate unnecessary searching. And you have to be clear or you have to be certain, if I change something only that gets changed. Something else does not get changed.

(Refer Slide Time: 58:02)



SECOND NORMAL FORM (2NF)

A relation is in 2NF if

- It is in 1NF
- Non key attributes functionally dependent on key attribute
- If key composite then no non-key attribute can functionally depend on one part of the key.

8.3.13 | Systems Analysis And Design © V. Rajaraman | 45 of 92

So, the next step is to go from what is known as a 1NF to 2NF. So, start with 1NF and non key attributes are functionally dependent on key attribute. All non key attributes must be functionally dependent on key attribute. If key composite then no non key attribute can functionally depend on one part of the key. So, I will explain this in very detail next time. Because, 2NF is a starting point for further details of 3NF and so on.

So, I should you should understand 2NF little bit more clearly. I do not want to rush through it. And so I will stop at this point. And continue from this next time.