DBMS

Highlighting patterns:

Common highlighting of important points

Common highlight but in different color to make for color mixing

Doubtful highlighting (need to get highlighted part verified)

I tend to forget this so need to remember it better.

Note-taking patterns.

Headings

Normal text

Diagrams and visual information

Note / info panel

Doubts

Other (PYQs, Questions)

DBMS (+8 marks)

ER model, Relational model - 1 problem

Relational algebra, Tuple calculus - 1 problem

SQL Integrity constraint - 1 problem

Transactional & concurrency control - 1 problem

File organization, indexing, B, B^+ trees - 1 problem

→ Time consuming

→ Multi-level indexing can also be problematic

Data

↓

Related data

for ex: s(Ram, 101, 21) ⇒ This is a record

↓

A collection of such records forms a relation / table

Eg: (Ram, 101, 21)
(Shyam, 102, 21)
(Gita, 103, 20)

↓

R_1 (Name, Rollno, Age) ⇒ This representation is called relational schema

Name of the table

↓

A collection of such relations forms a database

Eg: I can have another table with schema R_2(student, courses)

↓

The above examples look like a database for a college. We'll call it college.db

Note: In the above example,

These are known as attributes/columns

Name	Roll no.	Age
Ram	101	21

This is known as a tuple/row.

Also, in a relation schema, the unique attribute is underlined. In the above case, it is roll.no.

A row differs from a tuple in such a way that a row always means the entire record while a tuple can be a subset of the record as well.

The relation is a set:

$R_1 \{ (\text{Ram}, 101, 21),$
$(\text{Shyam}, 102, 21),$
$\vdots$
$\}$

Order cannot be changed

The number of elements in this set is called the quality of the set?

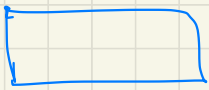
probably meant "cardinality"

ER-Model

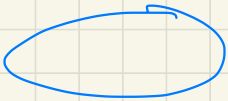
→ Diagrammatic representation of collection of objects (called entities) and relationship among the objects and their entities. → Verify if this is "other entities".

→ ER Model Components

- Entity Set
- Entity attribute
- Relationship b/w the entity



(An entity set is represented by a rectangle)



→ We represent entity sets and not single entities (although the terms are used interchangeably.)

Entity

→ Any real world object that is distinguishable from other objects based on their value (attributes) is an entity.

→ The diff. b/w an entity and an object is the distinguishability part.

→ Every entity needs to have atleast one attribute so it can be distinguished from other entities.

→ Entity is of 2 types

Entity

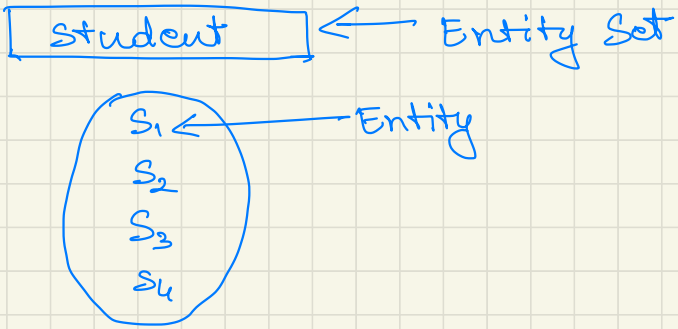
Tangible
- → Can touch
- → Physically exists
- → Eg: chalk

Intangible
- → Cannot touch
- → Logically exists i.e. logical entity.
- → Eg: Account number 1413279763

Entity Set

→ A collection of entities of the same type is called an entity set.

→ Entity sets are represented in the ER Model as:



Entity Attribute

→ Characteristics or properties of an entity which distinguish one entity from another.

→ Eg: Students have attributes:
- ⇒ Name
- ⇒ Roll no.
- ⇒ Mobile no.

→ Attributes are called columns in Table

→ student (Name, Roll no, Mobile no)

→ In relational schema, we call them attributes and not columns.

→ In the foll. section, we define the diff. types of attribute.

→ An attribute can be of multiple types.

For ex: An attribute can be simple, single-valued and stored. Like a roll. no.

→ But an attribute cannot belong to conflicting types.

Eg: An attribute cannot be simple and composite.

Attribute types

Simple attribute

→ It cannot be broken down further

→ It is also called an atomic attr.

Eg: Gender.

Represented by oval

Gender

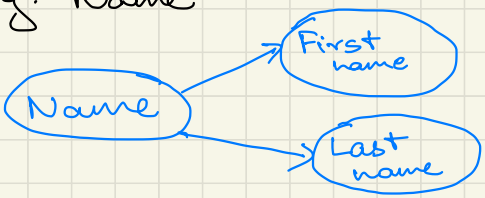
Composite attribute

→ Which can be broken down further

→ Also called inatomic

→ Represented by oval connecting with more ovals.

→ Eg: Name



Single value

→ Have only one value.

→ Represented by oval

→ Eg:

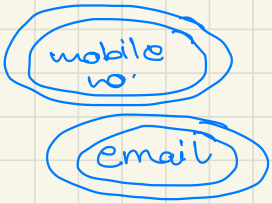
Aadhaar no.

Multivalued

→ Can have more than one value in the attribute.

→ Eg: mobile no.s, email ids

→ Represented by double ovals.



Stored value

→ Eg: DOB

→ This is value that needs to be stored.

→ Represented by oval

→ Eg: DOB

Derived value

→ Can be derived

→ Represented by dashed oval

→ Eg: Age

Age

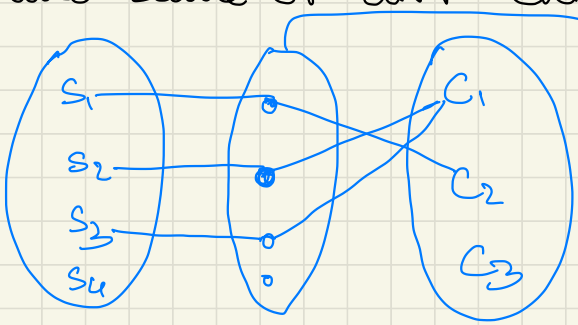
Note: Whether an attribute is simple or composite is dependant on our view of it.

Eg: Roll no. of the form 16H61A0505 can be considered atomic if we want to but if we want, we can also consider it to be composite.

Relationship b/w entities

Relation or association b/w 2 or more entities of the same or diff entity set.

Eg:



This entire thing is your relationship set while its constituent connections are your relations

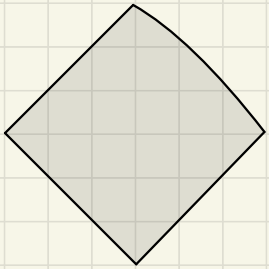
Relationship set: A collection of similar type of relationships.

Note: A relationship represents the relation b/w 2 entities while a relationship set represents the relation b/w 2 entity sets. So, in ER model, we use relationship set.

is this actually "function"?

Unlike in a function, in a relation it is not necessary that every element should participate in the relation. Eg: S_4 in the above example

In E-R model, relationship set is represented by diamond/rhombus.



Example ER-model

Let's construct the example ER Model for "students enrollment in courses"

A student entity set will probably have the foll:
- → Name (composite)
- → DOB
- → Age (derived)
- → Roll no.
- → Mobile no. (multivalued)

A course entity set will probably have the foll:
- → Name
- → Course Id

The relationship b/w the 2 will be through enrollment.

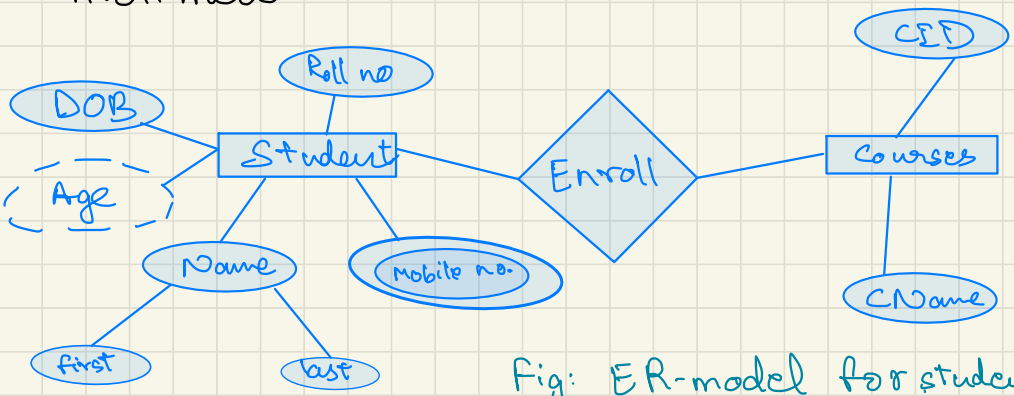
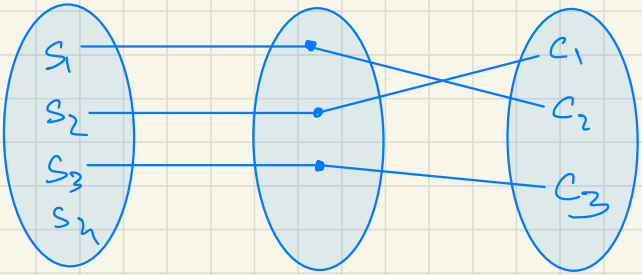


Fig: ER-model for student enrollment in courses

We can also create a set diagram for the above ER-model:



From the ER-model, a couple of things are not clear:

1) Is every entity in the "student" entity set supposed to enroll?

This issue is tackled by the participation constraint which is of 2 types:

a.) Total participation (all entities need to participate.)

b.) Partial participation (not all entities need to participate.)

2.) Can a single "student" entity enroll in multiple course entities?

This issue is addressed by cardinality ratios which have the foll. types:

a.) One to one
b.) One to Many
c.) Many to One
d.) Many to Many

Relationship set components

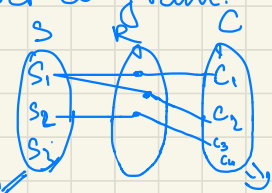
1) **Degree of relationship**: The no. of entity sets that the relation is connected to.

In the above example, the relation is connected to 2 entities. So, it's binary.

We can have unary (self join), binary, ternary .... n degree for a relationship.

2) **Cardinality ratio**: It show how many times (atmost) an entity can participate in a relationship.

set diagram:



ER model

atmost participation = 2 (many)

at most participation (1)

In ER model, we only deal with 2 values:

a) 1
b) Many

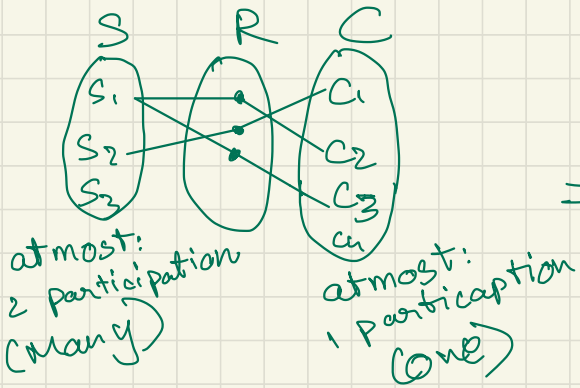
These values represent the atmost no. of relationships that an entity set's members can have.

Note that the keyword here is "atmost". It represents the maximum

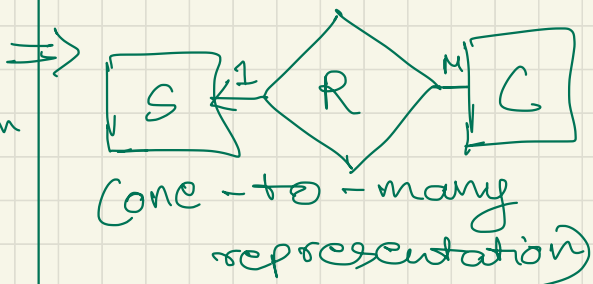
no. of relationships an entity can have. But the minimum no. of relationships is not defined. So, an entity can have 0 relationships as well.

Eg: In the above set diagram, S_3 has no relationships. Similarly, C_4 also does not have any relationships.

Note: When converting from a set diagram to an ER model, note that the representation of 1 and Many changes.



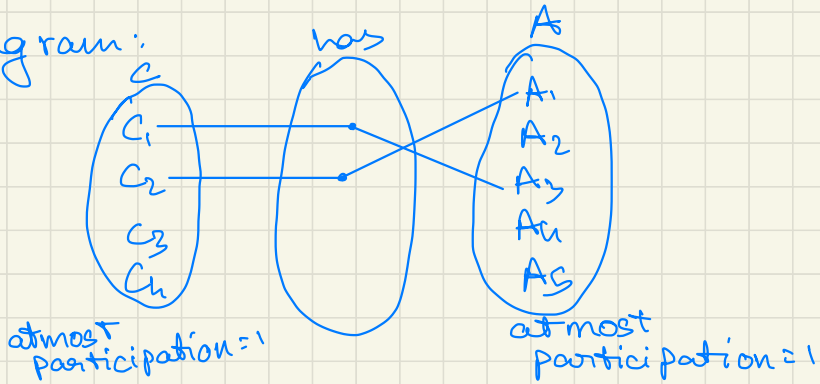
Convert to ER:



1:1 cardinality ratio:

Eg of this is a citizen-aadhar relationship

Set diagram:



Here, the atmost participation of every citizen is 1. Why? Since once a citizen has received an aadhar, they cannot have a 2nd aadhar.

It is valid that citizens like c_3, c_4 don't have aadhars (haven't applied for it) but cardinality only cares about atmost values.

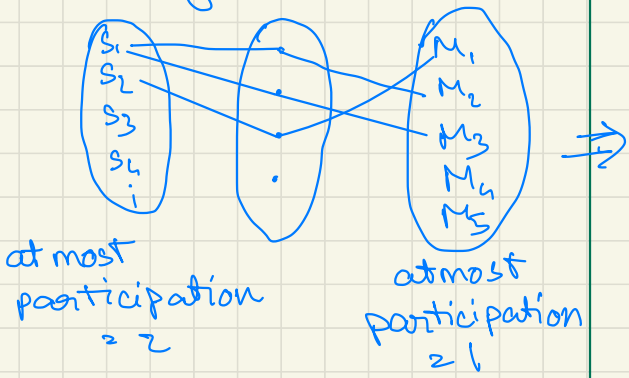
Similarly, not every aadhar that is generated needs to be assigned. But once assigned (participated), it cannot be assigned to another person as well.

ER model:

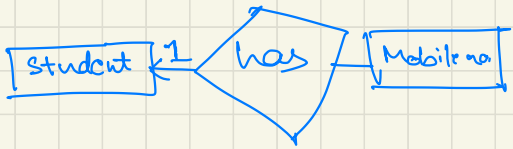
1-Many cardinality ratio:

Eg: Student have multiple mobile no.s:

Set diagram:



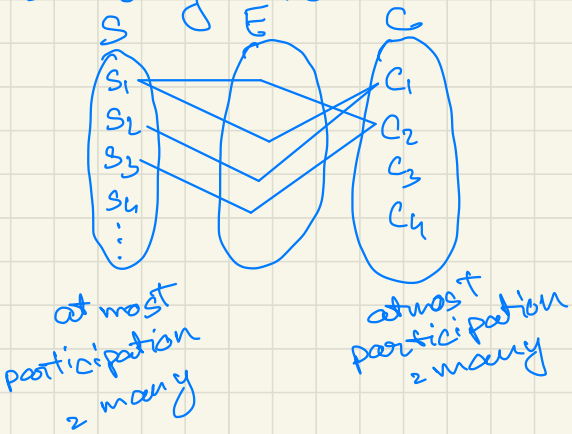
Convert to ER-model:



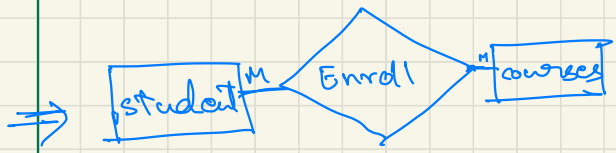
Many - Many cardinality ratio:

Eg. Student enrolled in courses

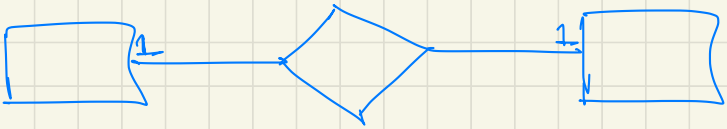
Set diagram:



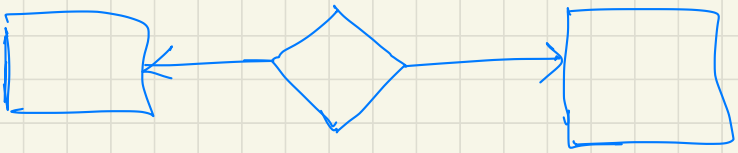
Convert to ER model:



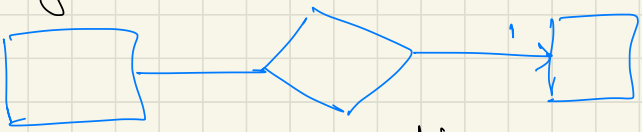
You can denote 1—1 relationship using an arrow or a 1



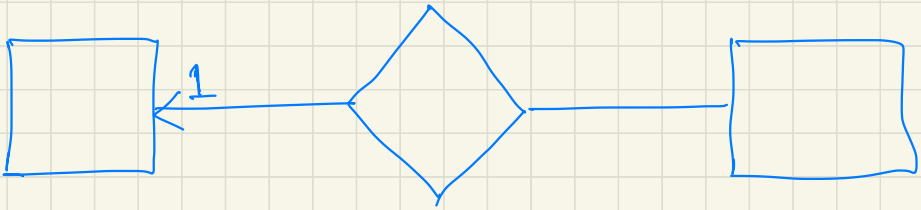
(or)



You can denote M-1 (or) 1-Many by not adding an arrow/1 on the many entity side

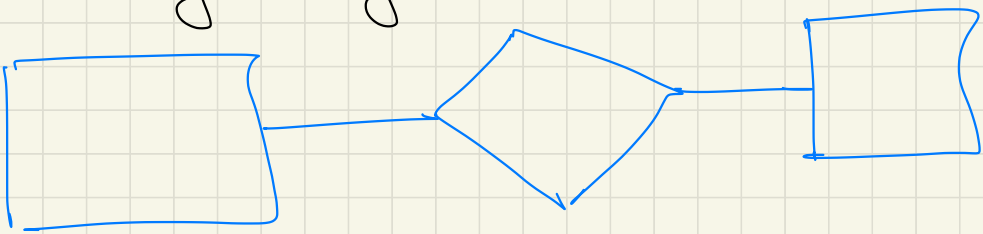


Many-1 ratio



1 - Many ratio

Many - Many can be denoted by not adding any arrows



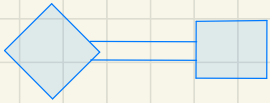
The student - courses ER model was depicting a Many - Many ratio.

Remember the notation as follows:

Many - no arrow/no.
One - add arrow/no.

3.) Participant Constraint

Total participation is denoted by double-line



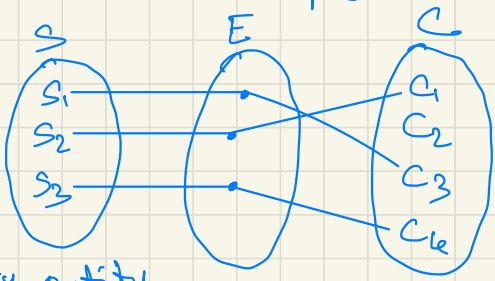
Partial participation is denoted by single line

Participation constraint tells us if every entity of entity set participates in relationship set.

If each entity has participated, it is total participation

If some entity is not participating then it is partial participation

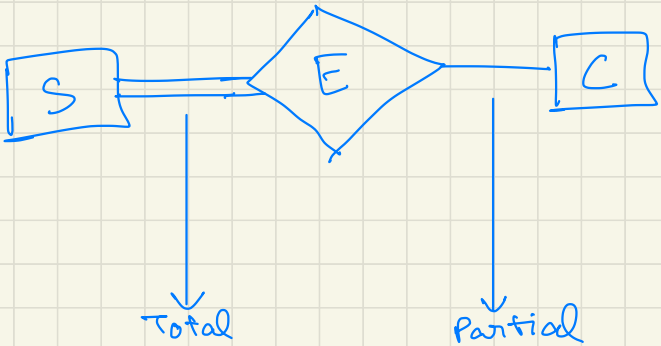
Student - course example



Every entity in S participated

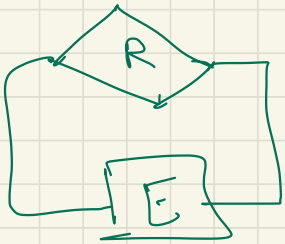
Some entity in C did not participate

Convert to ER model



Note: The degree of a relationship set is different from the degree of a relation. The degree of a relation denotes the no. of columns in it.

Unary relationship set:



Also called recursive relationship

Note: A relationship set cannot exist without an entity but an entity can exist without a relationship set.

Note: All the elements in the above ER models will be converted to tables.

Eg:

In the above ex, each entity set and the relationship set will be a table.

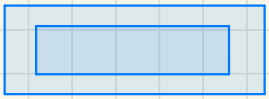
Since the relationship set is a table, it can also have its own attributes.
For ex: Date of enrollment.

Such an attribute on a relationship is called a descriptive attribute.

Weak Entity → Reread/Rewatch

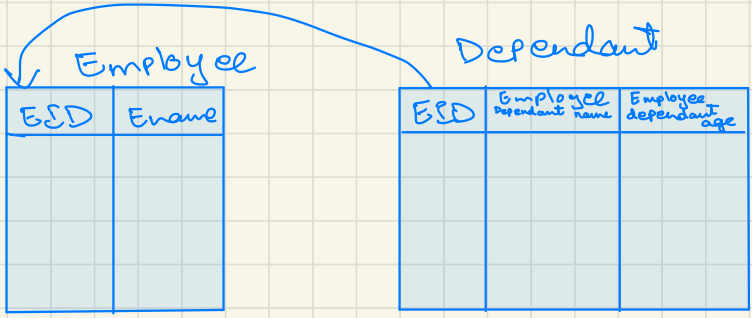
Lecture 2
21st July, 2023

→ It is dependant on strong entity.
→ It is represented by a double rectangle.



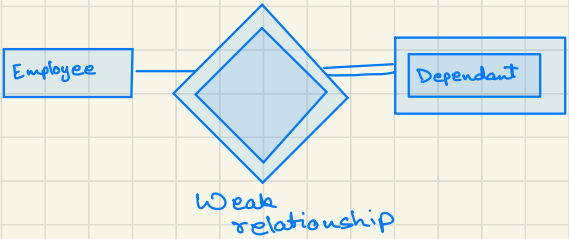
→ Primary key of strong entity is part of keys of weak entity. The key can be stored in atomic or composite form by the weak/dependant entity set but it will be stored.

Eg:



→ A strong entity can exist without a weak entity but a weak entity cannot exist without a strong entity.

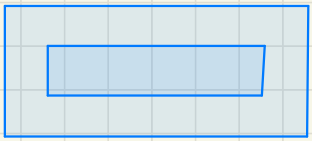
→ A weak entity set always has total participation because of it's dependancy.



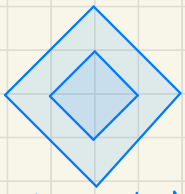
→ A relationship between a strong entity and a weak entity is called a weak relationship.

→ While we might or might not make a table for strong relationships (b/w strong entities), we DO NOT make a separate table for weak relationships since the weak table already stores all the necessary info.

→ A weak relationship is represented by a double diamond/rhombus.



Weak entity



Weak relationship

2005 - IT

Store rent payment information in Lodging.

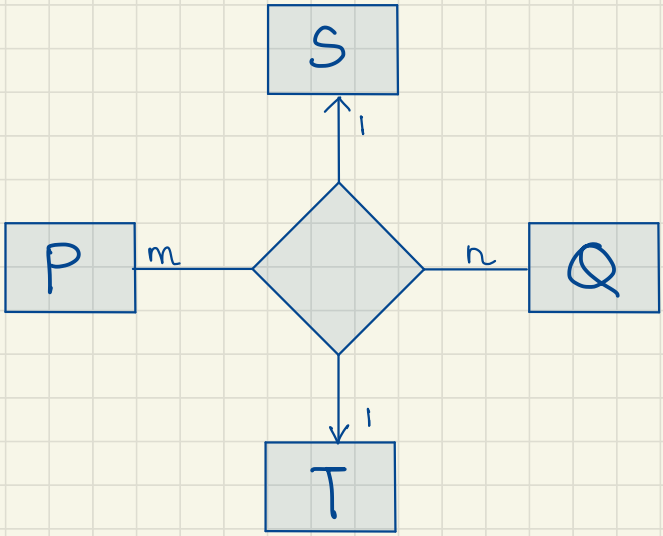
2012 - CS

Q) A.) Attribute can have more than one value.
B.) Attribute of entity can be composite.
C.) In relational table, attribute can have multiple values.
D.) In relational table, attribute can have exactly one value or a NULL value.

2020-CS-2

Q.) How to represent the supporting many-one relationships of a weak entity set in ER diagram?
→ Diamond with double border.

Q.)



Which is not correct relation (table)?

I

P	Q	S	T
P_1	Q_1	S_1	T_1
P_1	Q_1	S_1	T_2

II

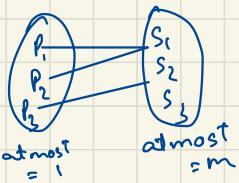
P	Q	S	T
P_1	Q_1	S_1	T_1
P_1	Q_1	S_2	T_2

III

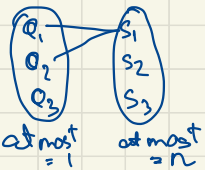
P	Q	S	T
P_1	Q_1	S_1	T_1
P_1	Q_2	S_1	T_1

Lecture 2 - 1:20:00 for explanation

P - S ⇒ m - 1 ⇒

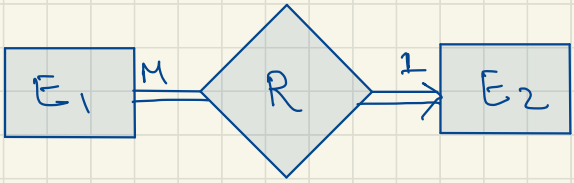


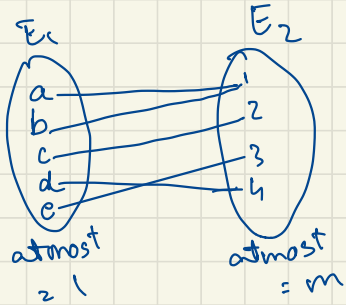
Q - S ⇒ m:1



2018 - CS - 1

Q.)





$|E_1| > |E_2|$

(Cardinality of E_1 is greater than cardinality of E_2. So, no. of elements in E_1 > no. of elements in E_2)

Functional Dependancy (FD):

→ FD is a method which describes the relationship b/w attributes i.e. it describes attribute dependancy.

→ It is represented as:

$$A \longrightarrow B$$

This is read as

A functionally determines B

(or)

B functionally depends on A.

Valid / Invalid FD:

→ A functional dependancy $A \longrightarrow B$ is valid if:

(i) every element in A is distinct

A	B
1	x
2	x
3	x
4	y

(ii) for every repeated value of A, value of B is also repeated.

A	B
1	x
2	y
3	y
1	x

An example of (ii) is DOB ⟶ Age i.e. Age is functionally dependant on Date Of Birth. So, when Age is calculated for two people with the same DOB, their age will also be the same.

Note : If in an FD $A \longrightarrow B$, A has all distinct values, then it is valid.

If a functional dependancy does not satisfy the above 2 conditions, then it is invalid.

Shortcut for FD:

① $A \longrightarrow B$

All values are distinct

Valid

② $A \longrightarrow B$

All values are the same

Valid

If ① & ② are not satisfied, we check for invalidity.

$A \longrightarrow B$

Check a repeated value

B returns distinct result

$\Rightarrow$ Invalid

2000 - 3

X	Y	Z
1	4	2
1	5	3
1	6	3
3	2	2

A.) $XY \rightarrow Z$ and $Z \rightarrow Y$

Let's take $Z \rightarrow Y$

(i) Does Z have all unique values? No (2, 3 repeated)

(ii) Does Z functionally determine the same value for Y for every repeated value in Z? No.

$Z(2) \longrightarrow Y(4)$

$Z(2) \longrightarrow Y(2)$

B.) $YZ \rightarrow X$ and $Y \longrightarrow Z$

$Y \longrightarrow Z$

(i) Y have all distinct values? Yes So, valid.

$YZ \longrightarrow X$

(i) YZ have all distinct values? Yes. So, valid.

C.) $YZ \longrightarrow X$ and $X \longrightarrow Z$

Take $X \longrightarrow Z$

(i) X all distinct? No.

(ii) X repeated give same Z value? No

$X(1) \longrightarrow Z(2)$

$X(1) \longrightarrow Z(3)$

D.) $XZ \rightarrow Y$ and $Y \longrightarrow X$

$Y \longrightarrow X$

(i) Y have all distinct values? Yes. Valid.

$XZ \rightarrow Y$

(i) XZ have distinct values? No. X(1) Z(3) repeated.

(ii) For repeated values, does XZ functionally determine same Y value? No. So, invalid.

$X(1)\,Z(3) \longrightarrow 5$

$X(1)\,Z(3) \longrightarrow 6$

Q.)

A	B	C
1	2	4
3	5	4
3	7	2
1	4	2

1.) $A \rightarrow B$, $BC \rightarrow A$
2.) $C \rightarrow B$, $CA \rightarrow B$
3.) $B \rightarrow C$, $AB \rightarrow C$
4.) $A \rightarrow C$, $BC \rightarrow A$

As a first step, just check if the attribute with distinct values is on left side. Since that will definitely be valid.

Q.)

A	B	C	D	E
a	2	3	4	5
2	a	3	4	5
a	2	3	6	5
2	2	3	6	6

1. $A \rightarrow BC$
2. $C \rightarrow DE$
3. $DE \rightarrow C$
4. $BC \rightarrow A$

C has all same values. So, if it's functionally dependant on any value (i.e. it is on the right side) it can be valid (if it is the only dependant, it will be definitely valid.)

Trivial and Non-trivial FD:

FD is of 2 types

FD

Trivial (always valid)

FD $X \rightarrow Y$ is called trivial if Y is a subset of X ($Y \subseteq X$)

Eg: $AB \rightarrow B$

$A \rightarrow A$

$B \subseteq AB$

$A \subseteq A$

Non-trivial (may or may not be valid)

An FD $X \rightarrow Y$ is non-trivial if $Y \not\subseteq X$.

Eg: $AB \rightarrow C$

$AB \rightarrow BC$

We deal with non-trivial functional dependancies.

Armstrong's Axioms of FD

Lecture-3
3rd Aug, 2023

An axiom is a self-evident principle that is accepted as true without proof.

Primary Axioms

There are 3 primary axioms, abbreviated as RAT:-

① **Reflexivity:**

If $B \subseteq A$, then $A \longrightarrow B$ is always valid.

Reflexivity is just a diff. name for a trivial FD.

② **Augmentation:**

If $A \longrightarrow B$, then $AC \longrightarrow BC$

Eg: Roll no $\longrightarrow$ name, then Roll no, Age $\rightarrow$ Name, Age

The term "augmentation" means increasing in size.

Augmentation here, basically just adds a trivial FD spin on a FD.

$A \rightarrow B$ is the normal FD $\Rightarrow$ $AC \rightarrow BC$ is the FD with a trivial spin

③ Transitivity:

If $A \rightarrow B$ & $B \rightarrow C$, then $A \rightarrow C$

Secondary Axioms:

① Union

If $X \rightarrow Y$, $X \rightarrow Z$ then $X \rightarrow YZ$

② Decomposition

If $X \rightarrow YZ$, then $X \rightarrow Y$ & $X \rightarrow Z$

③ Pseudo Transitivity

If $X \rightarrow Y$ & $WY \rightarrow Z$, then $WX \rightarrow Z$

This can be derived from:

$X \rightarrow Y$

Using Augmentation:

$WX \rightarrow WY$

Now, we have

$WX \rightarrow WY$ & $WY \rightarrow Z$

Using transitivity:

$WX \rightarrow Z$

④ Composition

If $X \longrightarrow Y$ & $Z \longrightarrow W$ then, $XZ \longrightarrow YW$

This can be derived from:

$X \longrightarrow Y$ & $Z \longrightarrow W$

Using Augmentation

$XZ \longrightarrow YZ$ & $Z \rightarrow W$

Using Augmentation again:

$XZ \longrightarrow YZ$ & $YZ \longrightarrow YW$

Using transitivity:

$XZ \longrightarrow YW$

Note: Using the rules marked with *, we can determine pseudo-transitivity and composition. So, it is better to remember these 4 for exam.

Closure of Attribute(s)

The closure of an attribute A is the collection of attributes that can be determined by A either directly or indirectly.

Denotion:

$$\text{Closure}(A) = A^+ \quad \text{or} \quad (A)^+$$

A closure is never empty. Even if no other attributes functionally depend on A, A^+ will atleast contain A itself (reflexivity).

Given the relation schema

$$R(ABCDEF)$$

and FD

$$FD = \{ A \rightarrow B, \; C \rightarrow DE, \; AC \rightarrow F, \; D \rightarrow AF, \; E \rightarrow CF \}$$

where A, B, C, D, E & F are attributes, let's find their closures

$B^+ = \{B\}$

$B \subseteq B$

reflexivity

$A^+ = \{A, B\}$

$A \rightarrow B$

$A \subseteq A$

We cannot find any more attributes using only A & B

So, closure(A) = $\{A, B\}$

$C^+ = \{C, D, E, A, F, B\} = R$

$A \rightarrow B$

$C \rightarrow DE$ (using decomposition)

$C \rightarrow D$

$C \rightarrow E$

$D \rightarrow AF$ (decomposition)

$D \rightarrow A$

$D \rightarrow F$

Because C is able to determine all attributes, C is a candidate key.

$E^+ = \{E, C, F, A, B, D\} = R$

reflexivity

$E \rightarrow CF$ (decomposition)

from closure (C) i.e. C^+ because $E \rightarrow C$

Because E determines C which in turn is able to determine all other attributes, E is also a candidate key.

$D^+ = \{D, A, F, B\}$

$D \rightarrow AF$ decomposition

$A \rightarrow B$

$F^+ = \{F\}$

Now, let's check the closure of multiple attributes:

$$(BD)^+ = \{\underbrace{B, D}, \underbrace{A, F}\}$$

$D \rightarrow AF$

$BD \rightarrow BD$
(reflexivity)

$BD \rightarrow B$
(ref.)

$BD \rightarrow D$
(ref.)

$$(AC)^+ = \{A, C, B, D, E, F\} = R$$

AC also looks like it would be a candidate key. But it is not. Let's see why.

Key:

A key is an attribute or a collection of attributes whose value is unique & is/are capable of determining all the remaining attributes of R.

In short:

$$key^+ = R$$

i.e.

$$closure(K) = R$$

Keys are of 3 types.

Key

- Super key (SK)
- Candidate key (CK)
- Primary key (PK)

Super key (SK)

A superkey is an attribute or a collection of attributes whose closure is R.

Eg: from the relation and FD in the previous example,

C, E and AC are super keys. There could be more as well.

$SK = \{C, E, AC, \ldots\}$

Candidate key (CK)

A candidate key is a superkey whose proper subset is not again a super key.

Eg: from the previous relation and FD, let's take the super key AC.

$A^+ \neq R$ ✓

$C^+ = R$

i.e. AC's proper subset C is a superkey by itself. So, AC is not a candidate key.

So,

$CK = \{C, E\}$

Candidate key is also called minimal super key.

Primary key (PK)

A candidate key selected for the database by the DB admin is called primary key.

Let's take a new example:

$R(ABCDEF)$

and

$A^+ = R$

$C^+ = R$

$(DE)^+ = R$

$(DC)^+ = R$

$(ADC)^+ = R$

$(BDF)^+ = R$

$(BDEF)^+ = R$

Here, the superkeys are:

$SK = \{ A, C, DE, DC, ADC, BDF, BDEF \ldots \}$

There will be many more SKs since every attribute that is grouped with an SK will form another SK.

Eg: A is already an SK.
So, AB, AC, AD, AE, AF, ABC ...
etc will all be SKs.

From the above list of SKs we can filter the candidate keys.

A $\longrightarrow$ candidate key since this is an atomic attrib. and we can't get a proper subset whose closure is R.

C $\longrightarrow$ same as above.

$DE \longrightarrow D^+ \neq R,\ E^+ \neq R$

So, DE is also a candidate key.

$DC \longrightarrow D^+ \neq R,\ C^+ = R$

Not a candidate key.

$ADC \longrightarrow$ Not a candidate key

$BDF \longrightarrow B^+ \neq R,\ D^+ \neq R,\ F^+ \neq R$

So, BDF is a candidate key.

$BDEF \longrightarrow (DE)^+ = R$

So, BDEF is not a candidate key.