Entity Relationship diagram.

conceptually model data

Class - user defined datatype.

↓

describing what are the properties of objects and methods in it

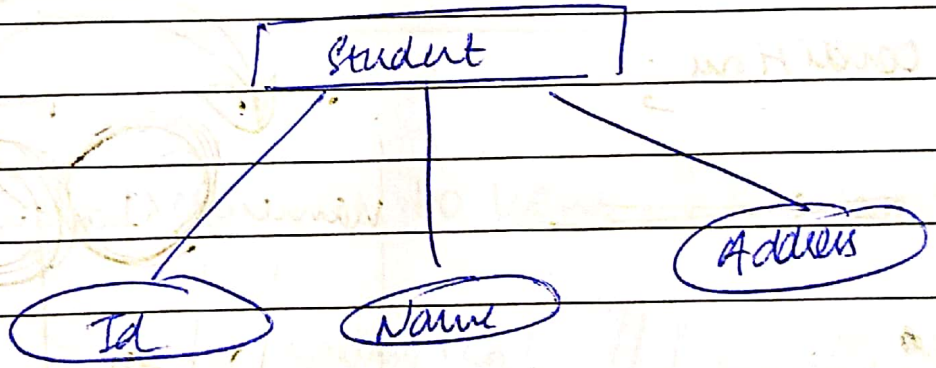
Object with independent existence : entity

entity set / type : denoted by ▭

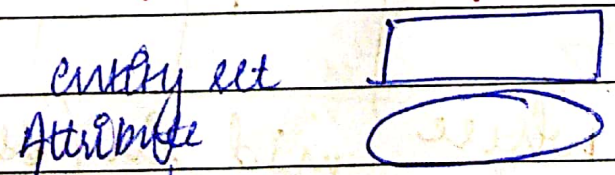
Attributes → represented by ◯

they share

~~If att~~ Attributes are connected by entity set.



* Sometimes entity and entity set are said interchangably



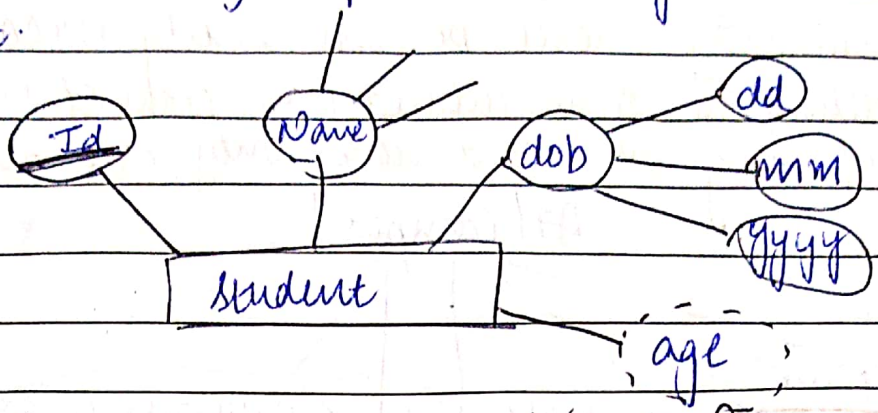
Key attribute ←

→ simple vs composite

→ Single valued vs multivalued

→ stored vs derived

Composite attribute: If simple attribute of it has other attributes in it.



Q) Mobile no is simple / composite attribute?

⇒ Simple

Multivalued: If an attribute has more than one values.
eg: if we have 2 mobile nos. Mob No 1 [ ]
2 [ ]

stored / derived: suppose we are storing dob so should we store age? (as it can be derived by performing mathematical operation on dob)
represented by dashed oval

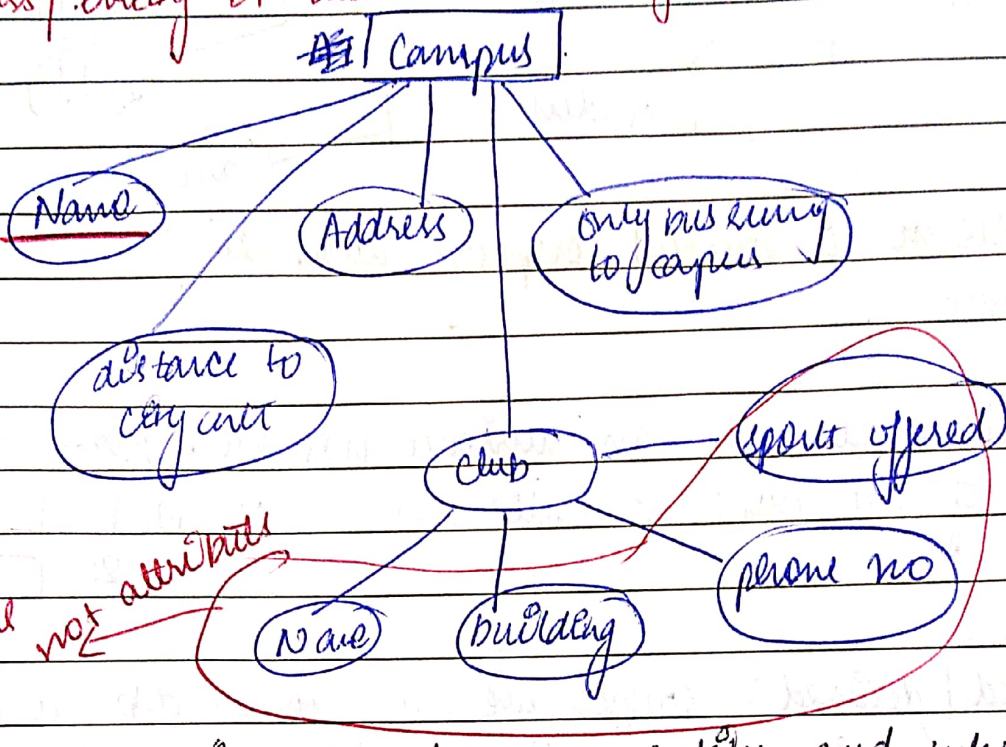
key attribute: an attribute that's unique. (underlined)

Generally NOUNS are attributes / entity set.

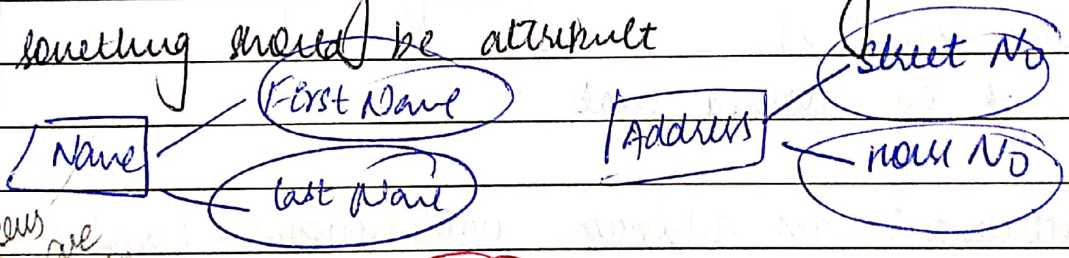
Q: professors VID, ~~Name~~, address, designation, ~~gender~~, ~~Reso~~
dob
Projects ~~project~~ no, spon

Q. University System

1° ✱ University would be an entity because we are modelling it. If we would have created university a class/entity it would have only one attribute ABC.



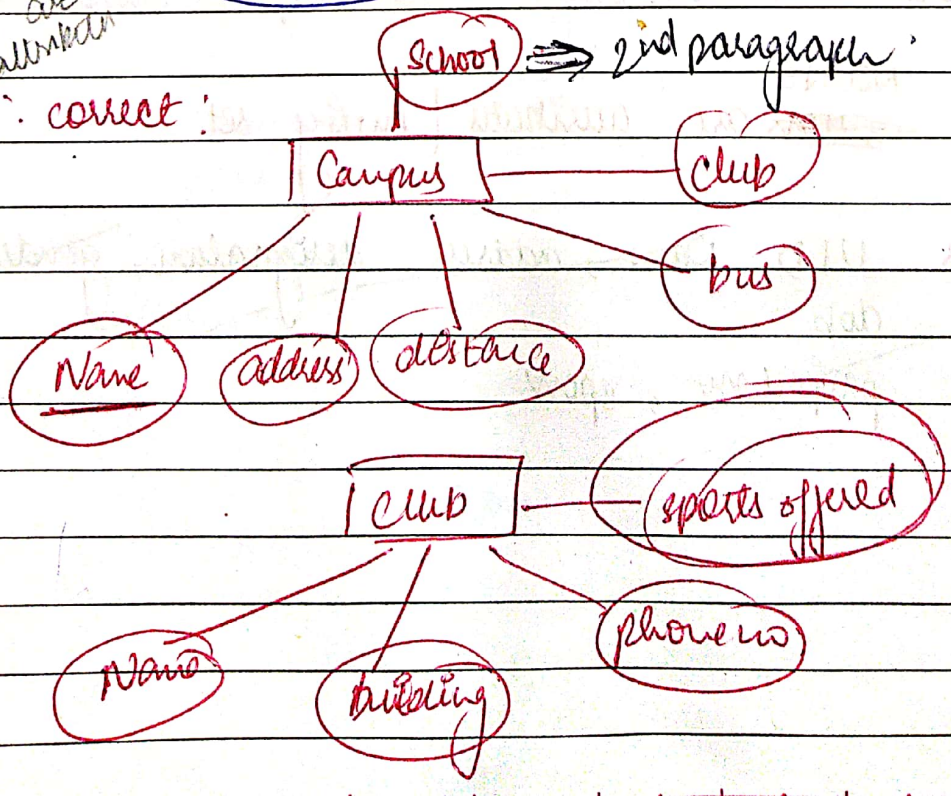
when something should be an entity and when something should be attribute



address parts are attributes

∴ correct!

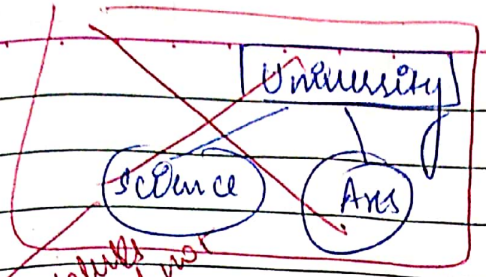
School ⇒ 2nd paragraph.

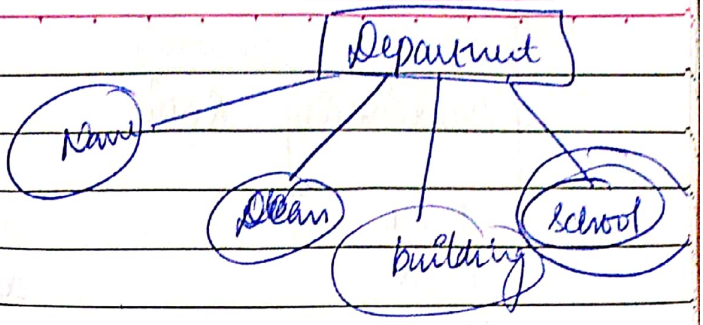


Class
attribute
↓
value

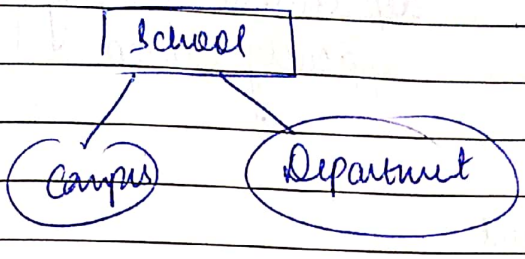
Camlin | Page
Date / /

2.

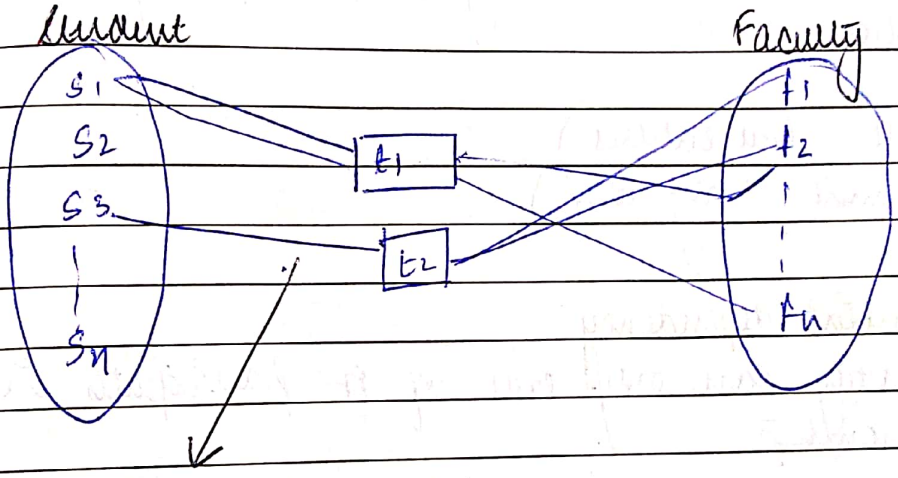




Science and Arts are values that can be stored in next attribute

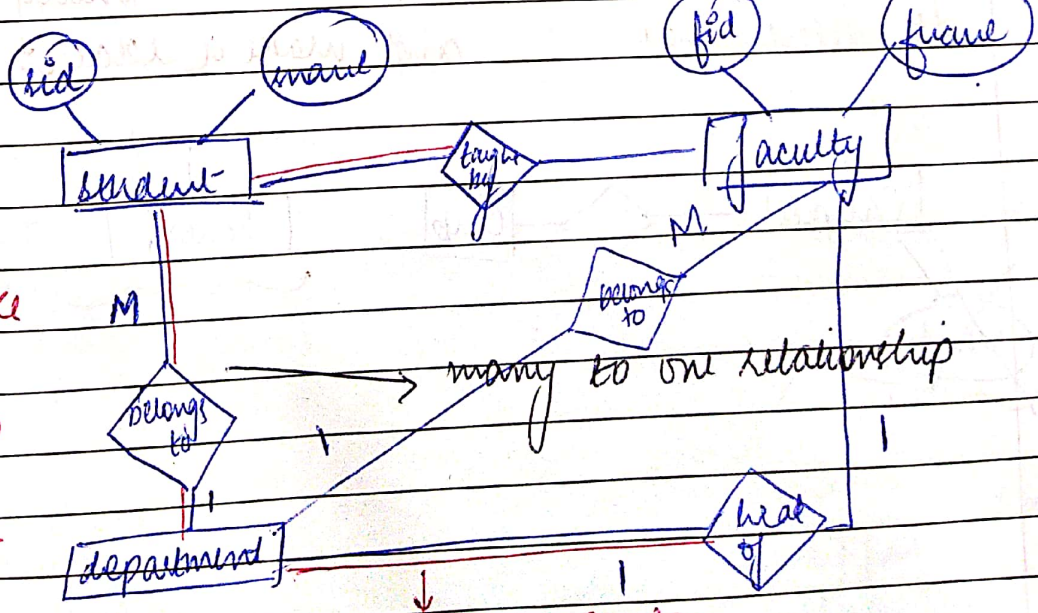


but one campus maybe the location of many schools.



Relationship is one to many.

The words we use 'belong to' 'taught by' should make sense when read top to bottom or left to right.



Cardinality Ratio

1 : 1
1 : M
M : 1
M : N

department : faculty (1 : M / ~~M : 1~~)
faculty : dept (M : 1)

Participation constraint

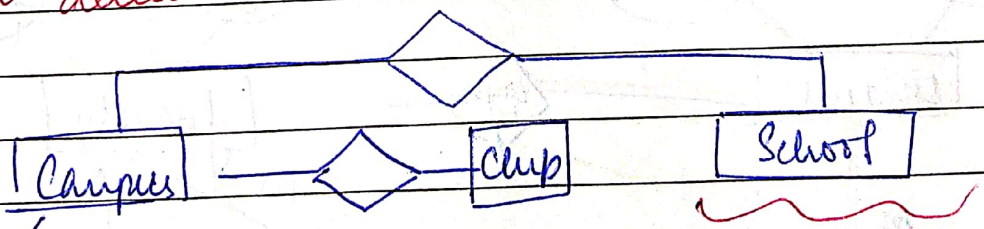
no of entities of an entity set participating in a relation

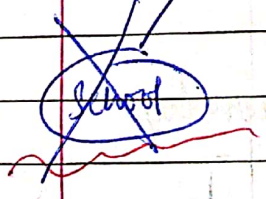
→ full (all entities)
→ partial (only some)

Existential dependency

An entity can only exist if it participate in a relationship

If there is an attribute which is an entity it delete that and make a relationship





Camlin | Page
Date / /

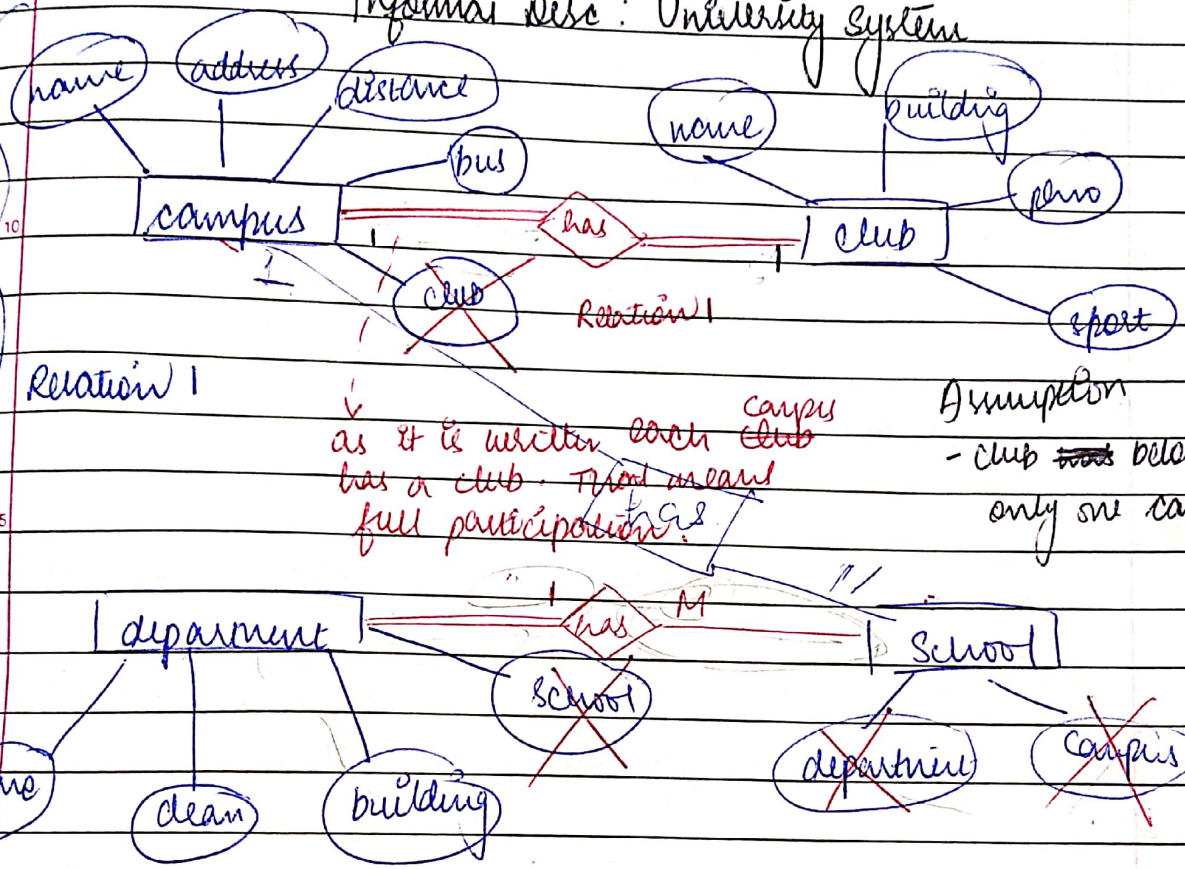
Student — teaches — Faculty

BINARY RELATIONSHIP

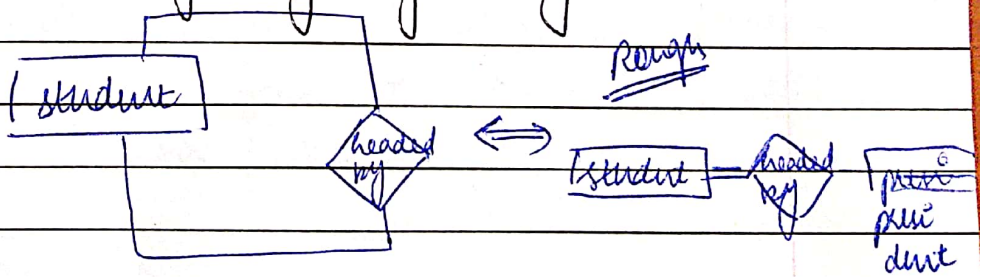
degree 2.

degree of relationship : how many entities are participating

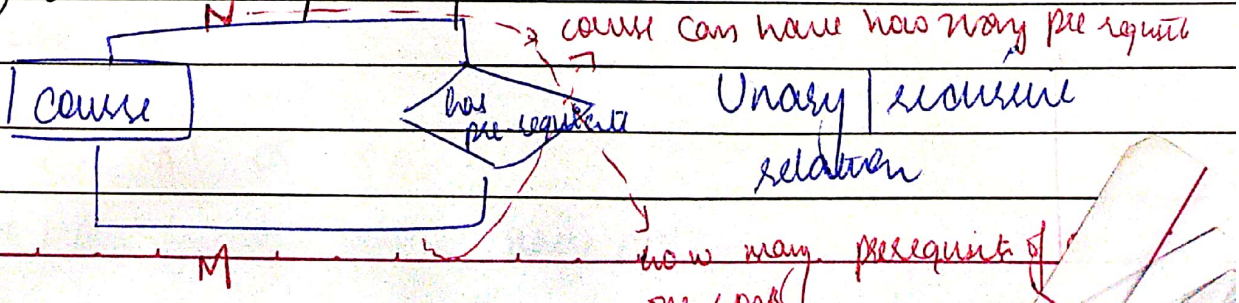
Informal Desc : University System



Understanding degree of unary relation



A course can pre-requisite course.



Multivalued

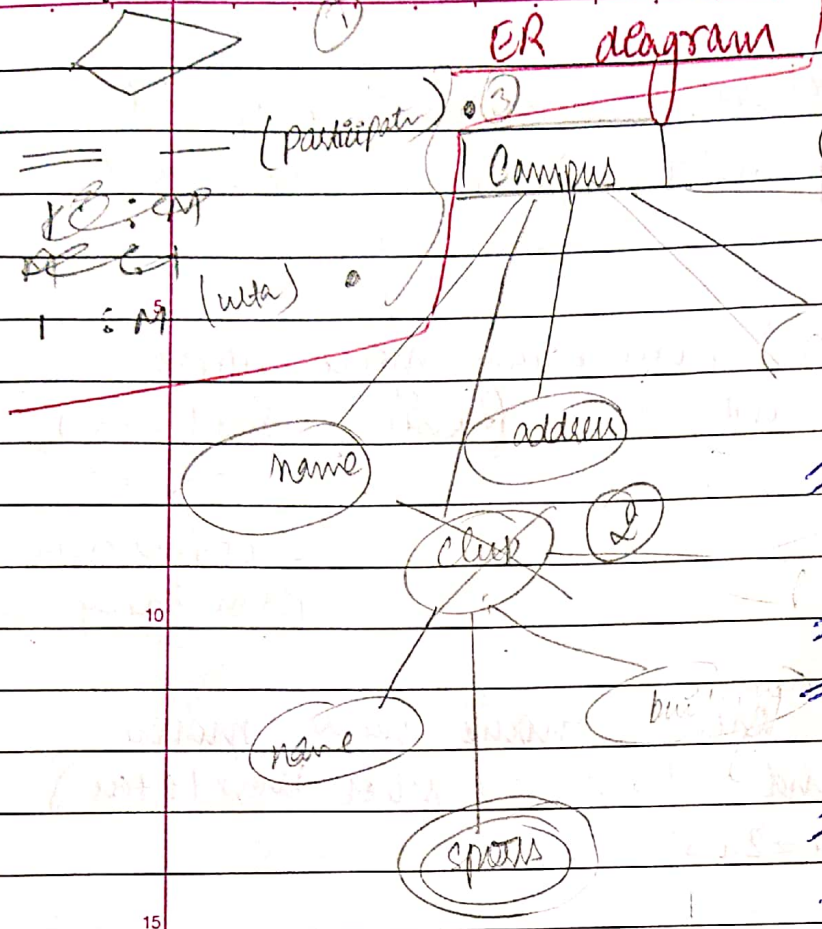
Key attribute

Weak entity

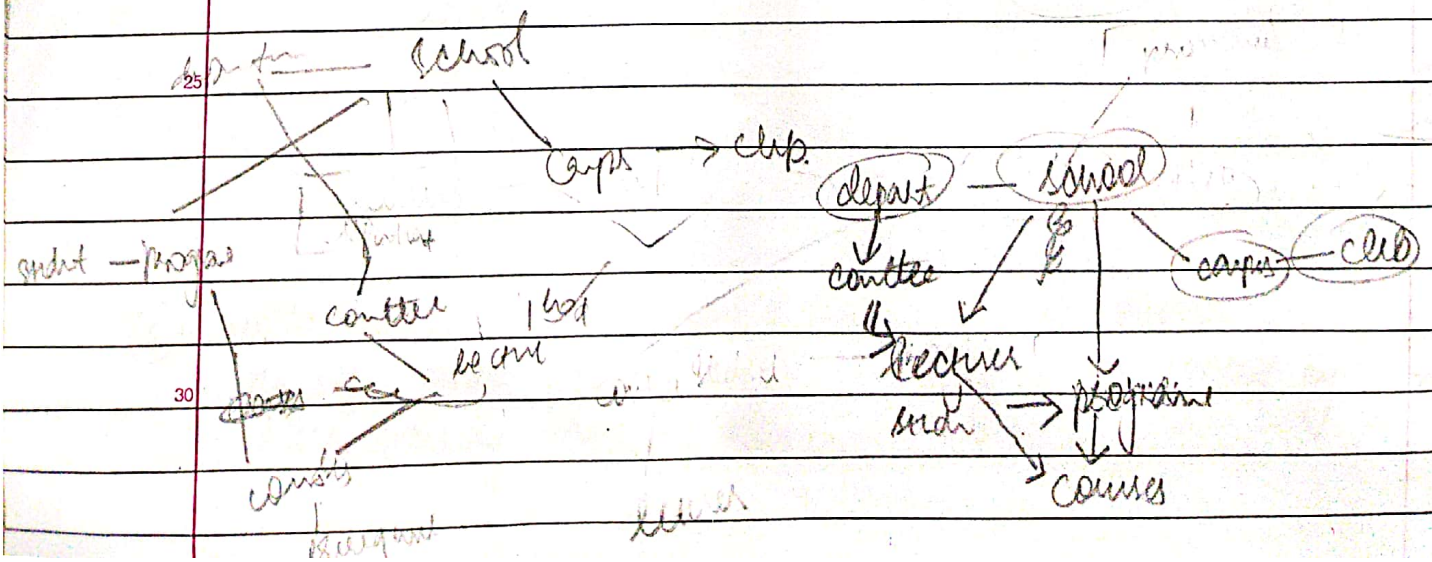
Partial key

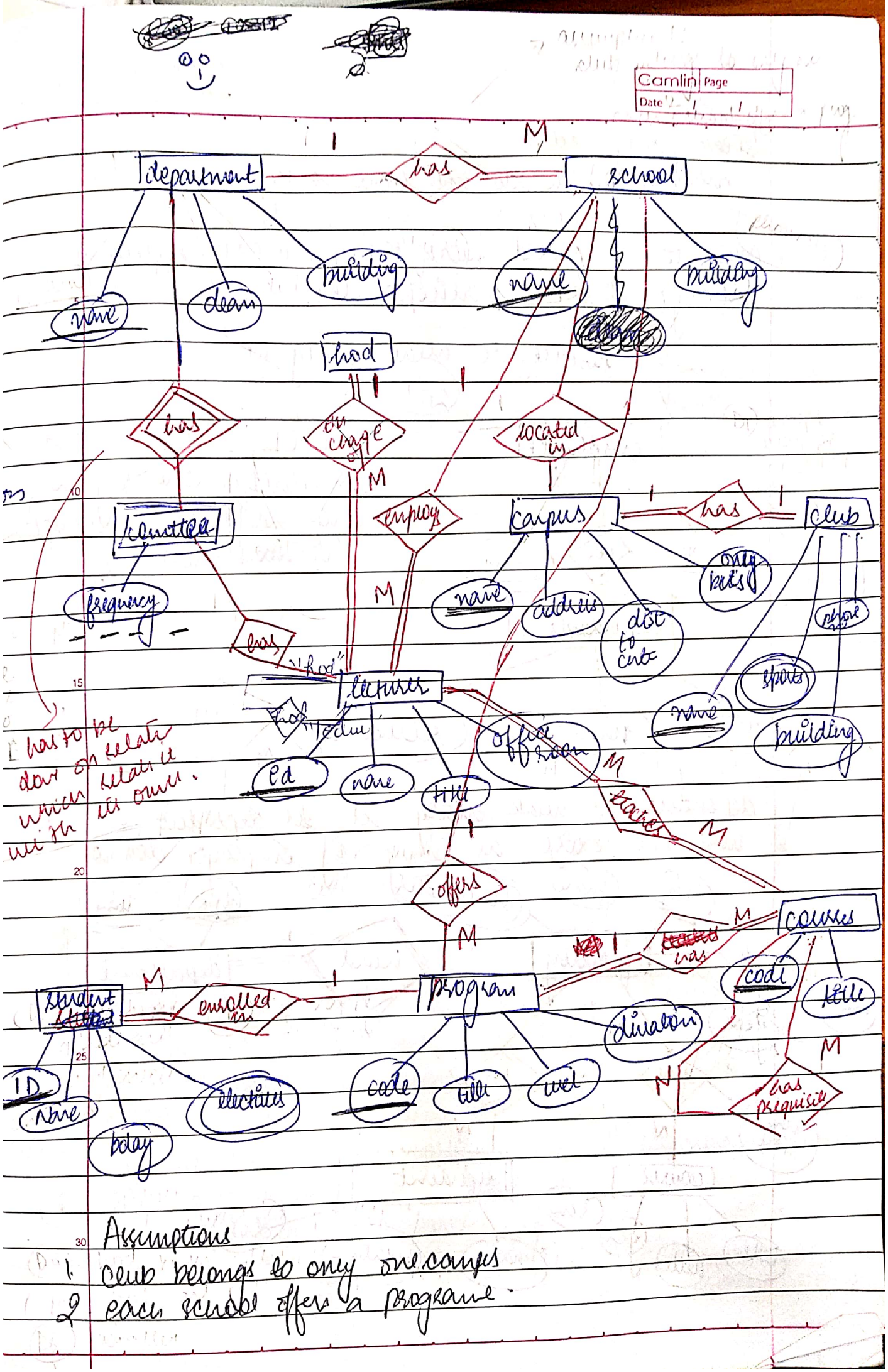
Camlin Page Date / /

ER diagram



1. a dept may be allocated then why full participation in class diagram
2. where to put grade, and year of enrollment and year of taking course.
3. lecturer & courses ka participation
4. hod entity has no attribute
5. hod and lect ka participation
6. computer
7. has prerequisite ka participation





has to be done on relation which relation it will go with its own.

Assumptions

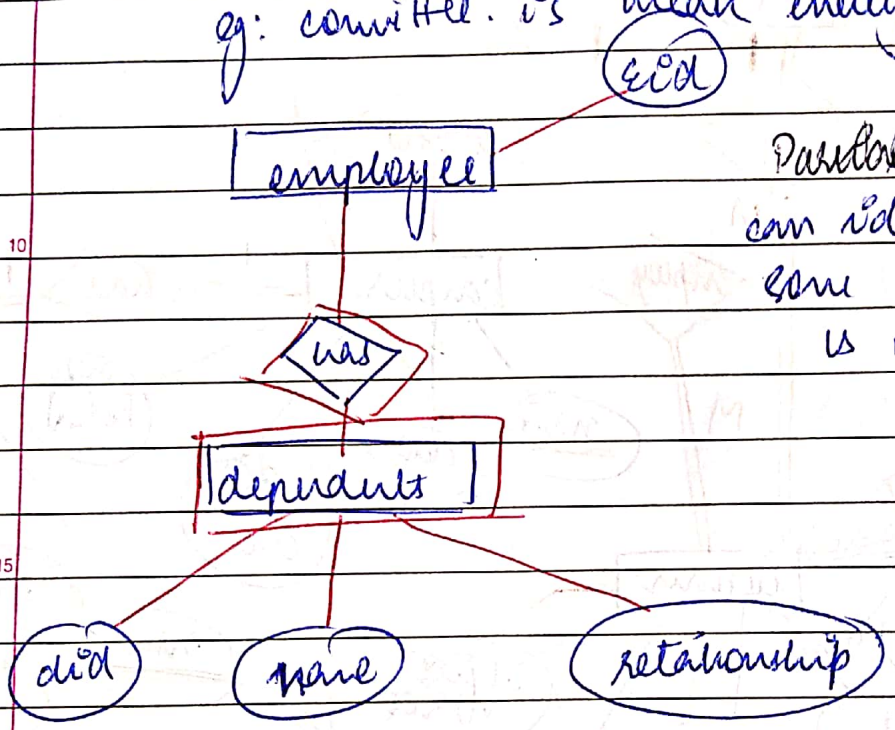
1. club belongs to only one campus
2. each school offers a programe.

Date / /

Weak entity set →
entities which do not have a key.

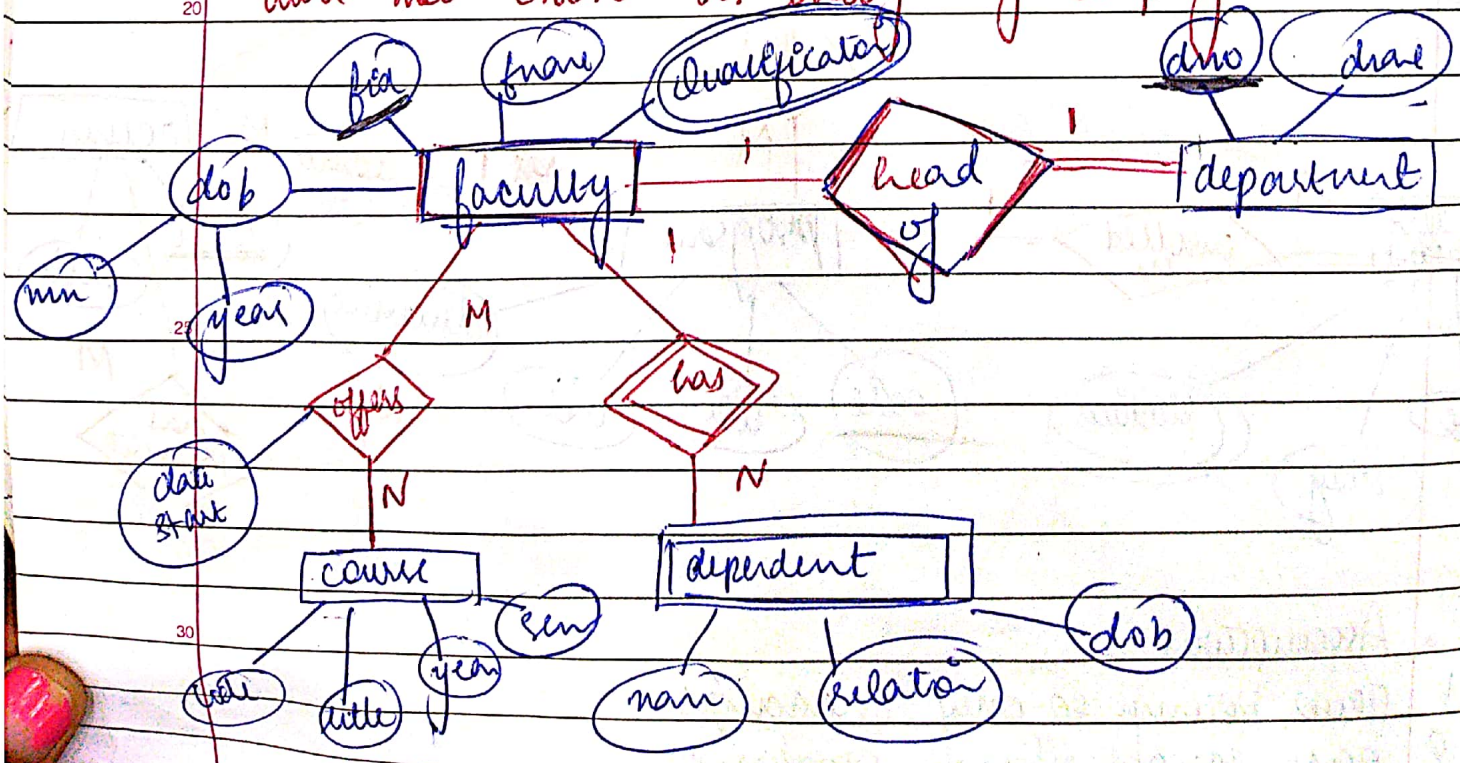
- Attribute is called weak entity, weak entity is always in full participation.

eg: committee. it's weak entity set



Partial key - attribute that can identify the key in some way if an assumption is made.

dependent is weak entity set as dependent will not exist as entity if employee does not -



Unconscious mind, I might awake
Wanna feel one last time take my pain away
If this night is not forever, atleast we are together
I know I am not Alone.

– ALAN WALKER

★ If we have 1 to 1 then ~~partial~~ key primary key on partial side becomes foreign key on total participate side.

Faculty (fid, fname, dd, mm, yy, dno)

Department (dno, dname, fid)

course (code, title, year, sem)

Q. Now which one to choose

Ans fid should be used as foreign key in department. as if dno is used we will have many null values coz not all faculty are HOD. so we choose where we get reduced NULL values.

★ We don't take multivalued variables in table like

Faculty (fid, fname, dd, mm, yy, dno)
Department (dno, dname, ~~hod~~ fid)
course (code, title, year, sem)

If we put fid as foreign key in Department then we have something like -

dno	dname	fid
01	CSE	f1, f2, f3 → multivalued problem

Case of multivalued program

Now relationship b/w faculty & course is many to many which means any word we use foreign key we will have multiple value of attribute so in such a case we create a seperate table of relation b/w them.

Faculty (fid, fname, dd, mm, yy, dno, doj)

Department (dno, dname, hodid)

course (code, title, year, sem)

offers (fid, code, date-start)

* here combination of fid, code is a primary key.

Now for weak entity, there is no primary key

Faculty (fid, fname, dd, mm, yy, dno, doj)

Department (dno, dname, hodid)

course ———

offers ———

dependent (fid, dependent no, relation dob)

so how could we add one?

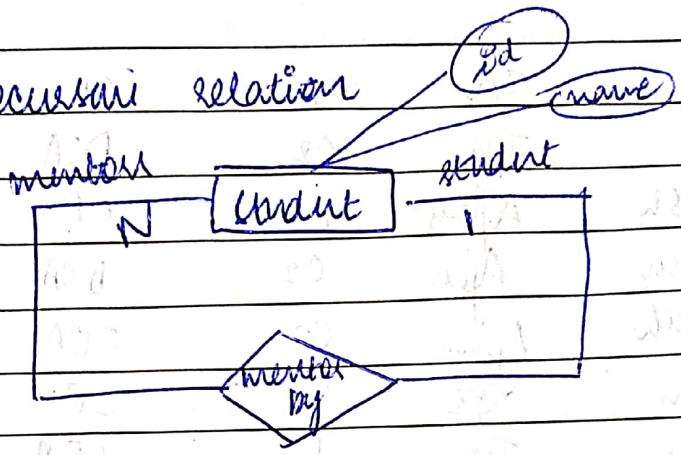
Sol Just as above we added an attribute dependent no, so combination of fid and dependent no will be primary key in dependent.

* for multi valued attributes

for - Qualification (fid, Qualification)

We create seperate table for multi valued attributes.

* case of recursive relation



so we have as follows :-

Student (id, name, mentorid)

Student

id	name	mentor id
1	a	2
2	b	null
3	c	1
4	c	1

Student

id	name	mentorid
1	a	2
2	b	null
3	c	1
4	c	1

↓

we create a copy of ~~table~~ for queries

self join

```
select S1.name, S2.name
from Student S1, Student S2
where S1.mentorid = S2.id
```

Normalization

Functional dependency (FD)

Sid	Sname	City	Cid	Cname	Credits
S1	Ram	Jpr	C1	CP	3
S1	Ram	Jpr	C2	DSA	4
S2	Ramesh	Ajmer	C1	CP	3
S2	Ramesh	Ajmer	C2	DSA	4
S2	Ramesh	Ajmer	C3	COA	4
S3	Mohan	Jpr	C1	CP	3
S4	Sohan	Jpr	C3	COA	4

Student (Sid, Sname, City)
Course (Cid, Cname, credits)
StuCourse (Sid, Cid)
Primary key (Sid, Cid)

Problems.

1. Redundancy : duplicate values,
   eg: credits for a subject is repeated and ~~same~~ ~~name~~ name.

Redancy → wastes space
→ problem in insertion
→ deletion

Insertion If we want to enter a new course, we can't until a student registers for it.
deletion If we want to delete a course we lose info about student also.

Camlin | Page
Date / /

Functional Dependency (FD)

Registration (Sid, Sname, city, Cid, Cname, Credits)

$$X \longrightarrow Y$$

determinant dependent

→ given a value of X, we are able to determine exactly one value for Y. so then we can say that X determines Y.

→ if 2 rows agree on their X value, they should also agree on their Y value.

→ as a many to one relationship btw X and Y.

→ It defines relationship b/w attributes of a relation

Sid → Cid

$S_1 \rightarrow \{3\}$

Functional dependency does not hold for Cid & Sid.

Sid → Sname hold.

Given a value of name, it may be possible there can be multiple value of Sid. as there can be many people with same name.

Sid → {name, city}

Armstrong Axioms

1. Reflexivity

   if $B \subseteq A \implies A \rightarrow B$

2. Augmentation

   if $A \rightarrow B \implies AC \rightarrow BC$

3. Transitivity

$A \to B$, $B \to C$ $\Rightarrow$ $A \to C$

4. Self determination

$A \to A$

5. decomposition

$A \to BC \Rightarrow A \to B$, $A \to C$

6. union

$A \to B$ $A \to C$ $\Rightarrow$ $A \to BC$

7. composition

$A \to B$ $C \to D$ $\Rightarrow$ $AC \to BD$

Q: $R(A, B, C, D, E, F)$

$F = \{ A \to BC$
$B \to E$
$CD \to EF \}$

Prove $AD \to F$

$A \to BC$

$A \to B$
$B \to E$

(transitivity) $A \to E$
$AD \to ED$

$A \to C$ (decomposition)

$CD \to EF$

$AD \to CD$ (augmentation)

$AD \to EF$

$AD \to E$, $AD \to F$

Camlin | Page
Date / /

- Closure of functional dependencies (deriving new dependencies)

closure of attributes

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

N.W

check for more composite key

left

$(A^+) = \{A, B, C, E\}$

$(D^+) = \{D\}$

$(AB)^+ = \{A, B, C, E\}$

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

Q. $A \to BC$
$B \to C$
$A \to B$
$AB \to C$
$AC \to D$

Convert them into a irreducible set of FDs.

(1) right hand side of all FDs should contain only one attribute.

~~$A \to B$~~
~~$A \to C$~~ → can be derived from (1) and (2)
$B \to C$ —(1)
$A \to B$ —(2)
$AB \to C$
$AC \to D$

Repeated

$A \to B$
$B \to C$
$AB \to C$
$AC \to D$

(2) Now, $(AB)^+ = \{C, A, B, C, \}$
$(A)^+ = \{B, C, A. \}$
$(B)^+ = \{B, C\}$

Date / /

Since A can determine all attributes no need of keeping B.

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

$(C)^+ = \{C\}$

$(A)^+ = \{A, B, C, D\}$

$A \longrightarrow B$

$B \longrightarrow C$

$A \longrightarrow C$

$C \longrightarrow D$

$\Longrightarrow$

$A \rightarrow B$

$B \rightarrow C$

$A \rightarrow D$

This is the irreducible set

Camlin Page
Date / /

20/9/19

for a specific staff number, we can determine specific name, position and branch no of that staff. The branch no: determines the address of the branch. The staff's salary is determined by the position held and branch.

Staff no Name position branch No address Salary.
SNo N P BNo A S

* does not give true short names

SNo → N, P, BNo
BNo → A

staff no → {name, position, branch no}
branch → address
{position, branch No} → salary

Q: Each supplier is identified by supplier no & has a name. Each supplier is located in precisely one city and has a unique by his city. A part is identified by its part no. A supplier supplies parts in a particular quantity
→ status. The status of the supplier is determined by his city.

R (Supplier No, Name, City, Status, Part, Part No., Quantity)

(Sno, pno) → Qty
Supplier No → Name, City, Status
Part No → Part
City → Status

* here don't use supplier as an attribute. Because suppose a student has an Id. So we use student id as an attribute. Functional dependency is b/w attributes not table & attribute.

Q. why are we doing this?
we want to convert a bad designed table into a good designed table. This is called Normalization.

Bad ⟶ Good

- Characteristics of Good table

1. Insertion
2. Deletion
3. Redundancy

Q. When is a table good?

If the relation is of 1NF / 2NF = bad designed table
3NF = good

- A relation is in 3NF iff :-

(1.) The non-key attributes (if any) are mutually independent
(2.) and irreducibly dependent on primary key.

Q. What are non key attributes?

for this we need to know key attributes are of R

(Sno & Pno) ⟶ we have no other candidate key ∴ this combination becomes primary key

* here we use closure property.

1. Mutually Independent

No FD should have both non-key attribute. In our R ~~&~~ city ⟶ ~~Status~~ status violates this rule as both city and status are non-key attributes.

2.

Only Qty is fully functionally dependent on primary key. All other non key attr are dependent on a part of primary key. All "~~non~~" " should be dependent on whole

∴ Our relation R is not 3NF.

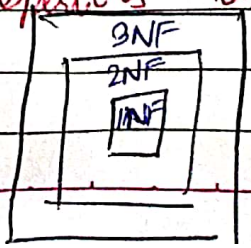
Sno	City	Status	Pno	Qty
S_1	Jaipur	10	P_1	20
S_1	Jaipur	10	P_2	30
S_2	Ajmer	20	P_1	20
S_2	Ajmer	20	P_2	20
S_3	Kota	40	P_3	20
S_2	Ajmer	20	P_3	30

→ deletion problem exists
Insertion " "
Redundancy " "

A relation is in 1NF iff :-
In every instance of that relation, every tuple contains exactly one value for each attribute

* If a relation is in 1NF and satisfies certain properties it can be converted in 2NF and if a relation is in 2NF and satisfies certain properties it can be converted in 3NF

How? Decomposition



- Decomposition concepts.

Eg:

Sno	Status	City
S_1	10	Jpr
S_2	10	Ajmer

R_1 (Sno Status) R_2 (status, city)

OR

R_3 (Sno, Status) R_4 (Sno, city

R_1 →

S_1	10
S_2	10

R_2

Status	City
10	Jpr
10	Ajmer

R_3 →

S_1	10
S_2	10

R_4

S_1	Jpr
S_2	Ajmer

→ If we join R_1 and R_2 can we get back original relation? NO

S_1	10	Jaipur
S_1	10	Ajmer
S_2	10	Ajmer
S_2	10	Jaipur

* The decomposition should be such that we get back original relation. This is called ~~non decomposition~~ lossless join decomposition. Also FD should be preserved.

If we join R_3, R_4, we get back origin relation

FD preserving → means is comfortable with what key attribute keep with them in separate table.

No functional dependency should be lost.

∴ We break the relation R as

R1 (Sno, name, city, status) → 2NF

R2 (S#, P#, Qty ( )) → 3NF

(1) (2) (3) (4) (5)

Now keys → ——

will have multually dependency ∴ not 3NF but 2NF.

A relation is in 2NF iff —

(1) it is in 1NF

(2) every non key attribute should be dependent on whole primary key.

So we again decompose :-

(or make them comfortable)

R11 (Sno, Name, City) → 3NF

(1) (2) FD apply here

but FD isn't preserved as ~~(2)~~ FD no (3)

(Sno → status) is lost.

But if we Sno → Status is derivable from Sno → city & city → status.

∴ We should have converted this into irreducible Canonical form

R12 (city, Status) → 3NF

(4) FD applies

(1) Sno → Name
(2) Sno → city
(3) Sno → status
(4) city → status
(5) Sno, Pno → Qty

Q. Relation between FD and relation ?

FD Primary keys are basically Primary keys.