E R Model

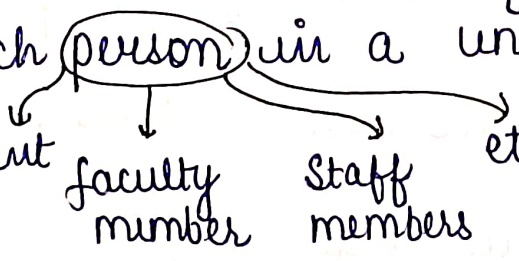
cc

- The entity-relationship model (ER-model) is a "blueprint for designing databases." It provides a conceptual representation of data and helps in understanding how data elements interact with each other.
- Widely used conceptual data model proposed by Peter P Chen in 1970s
- Data model to describe the database system at the requirements collection stage
  - High level description.
  - easy to understand for the enterprise manager.
  - rigorous enough to be used for system building.
- Concepts available in the model
  - entities and attributes of entities
  - relationships between entities
  - diagrammatic notation.

Basic components :-

- Entity - A thing (animate or inanimate) of independent physical or conceptual existence and distinguishable.
  [In OOP - object ≈ entity]
  - They have unique property.

eg Each person in a university is an entity.



Even a class room, a course are entities.

• Entity set or Entity type
Collection of entities all having the same properties.

Eg Student entity set - collection of all student entities.
Course entity set - collection of all course entities.

- Entity set do not need to be disjoint.
  Eg A ~~student~~ person entity may belong to both the Student entity set and the instructor entity set.

↑ columns in a table.

• Attributes
- Attributes define the properties of an entity.
- Each entity has a value for each of its attributes.
- chosen set of attributes - amount of detail in modeling.

Eg Student entity -
→ Student name
→ Roll no
→ Sex

- At least one attribute must be unique to differentiate each entity of a particular type (eg Student entity)

Types of Attributes

• Single-valued
  - having only one value rather than a set of values.
  - for instance, PlaceofBirth - single string value.

• Multi-valued
  - Having a set of values rather than a single values.
  - for instance, CoursesEnrolled, EmailAddress, PreviousDegree } attribute for Student.

- **Simple Attribute**
  Having atomic or indivisible values.
  eg Dept - a string
  Phoneno - a ten digit number.

- **Composite Attribute**
  Having several components in the value.
  eg Qualification with components
  (DegreeName, Year, UniversityName)

- **Derived Attribute**
  Attribute value is dependent on some other attribute.
  eg Age depends on DOB.
  Age = Current year - year of Birth

⇒ Attributes can be:
- simple - single valued , simple multi-valued ★
- composite - single valued , composite - multi valued.
  ↳ complex attribute

Domain of Attributes
Each attribute takes values from a set called its domain.
For instance, Students Age - {17, 18 ... 55}
HomeAddress - character strings of length 35.

★ Importance - ensure data consistency and prevents invalid values.

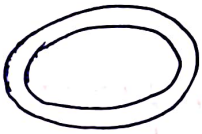
★ - Domain of composite attribute -
cross product of domains of component attributes.

★ - Domain of multi-valued attributes -
set of subset of values from the basic domain.

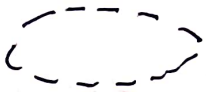
Diagrammatic Notation for entities

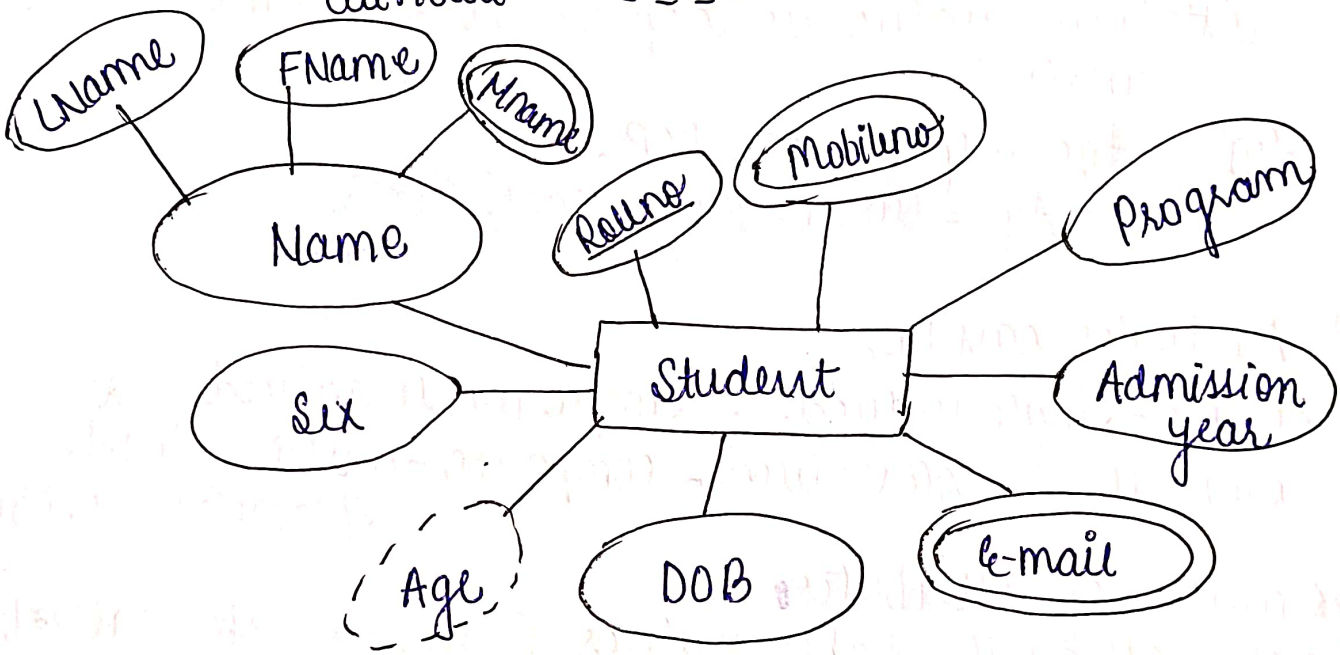
entity →  (rectangle)

attribute →  (ellipse connected to rectangle)

multi-valued attribute →  (double ellipse)

composite attribute → (ellipse connected to ellipse)

derived attribute →  (dashed ellipse)



- Entity sets and Key attribute

– Key – an attribute or a collection of attributes whose value(s) uniquely identify an entity in the entity set.

For instance
- RollNo – Key for Student entity set.
- HostelName, RoomNo – Key for Student entity set.

– A Key for an entity set may have more than one attribute.
– An entity set may have more than one Key.
– Key can be determined only from the meaning of the attributes in the entity type.
- Determined by the designer.

Relationship

When two or more entities are associated with each other, we have an instance of a relationship.

eg Student Himanshu enrolls in DBMS course.

- Relationship can have attributes to store additional information.

eg Instructor → Advises → Student (Relationship)

Advises → Date

- Formally, enrolls ⊆ Student × course.
- $(s, c) \in$ enrolls $\Leftrightarrow$ Student 's' has enrolled in course 'c'.
  - Tuples in enrolls - relationship instances.
  - enrolls is called a 'relationship Type/set.'

Degree of a relationship

- Number of entity set participating in a relationship.

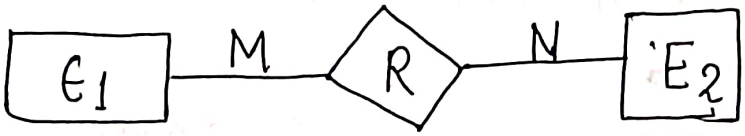
Degree 2 : Binary
Degree 3 : Ternary
Degree n : n-ary

Diagrammatic Notation

- Diamond shaped Box

- Rectangle of each participating entity is connected by a line to this diamond. Name of the relationship is written in the box.

• Binary relationship and cardinality Ratio



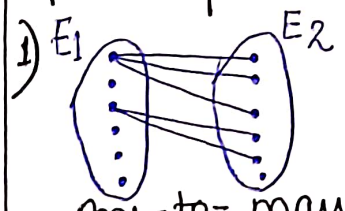
The maximum numbers of entities from E_2 that an entity from E_1 can possibly be associated thru R (and vice versa) determines the cardinality ratio of R.

Importance - In DBMS, mapping cardinality defines how entities in two different entity set relates to each other.

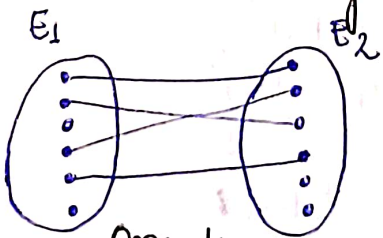
Need :-

1) Helps determine how many entities in one entity set can be associated with entities in another set.
2) Essential in ER diagram, as it determines the nature of the relationship between two entity sets.
3) Defines constraints on how relationships between entities are formed.

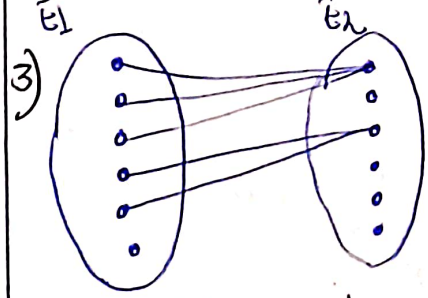
• Four possibilities are usually specified :

1) 

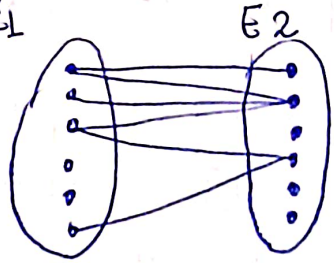
One-to-many (1:N)

2) 

One to one (1:1)

3) 

Many-to-one (M:1)

4) 

Many-to-Many (M:N)

Cardinality Ratio (Mapping cardinality)

1) **One to one** : An E_1 entity may be associated with at most one E_2 entity and similarly an E_2 entity may be associated with at most one E_1 entity.

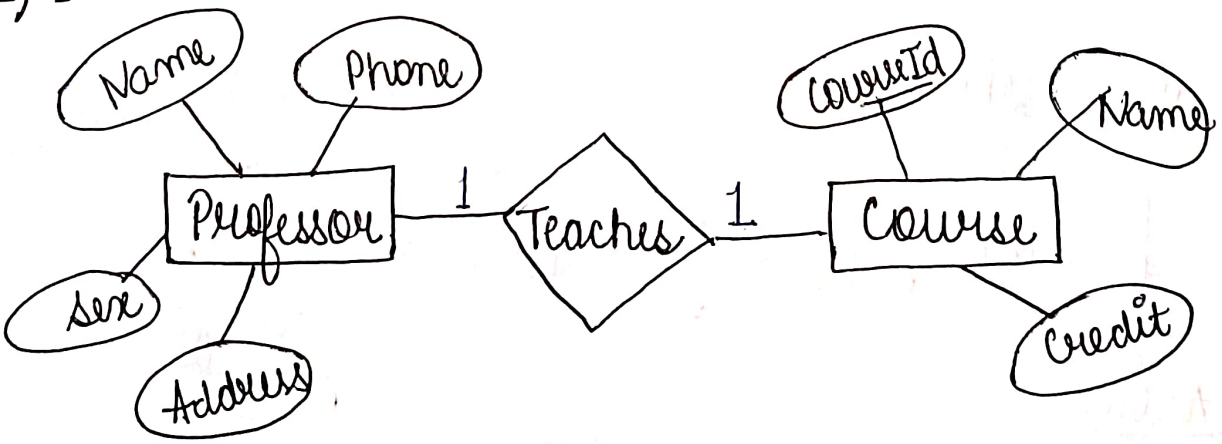
2) **One to many** : An E_1 entity may be associated with many E_2 entities whereas E_2 entity may be associated with at most one E_1 entity.

3) **Many to one** : An E_2 entity may be associated with many E_1 entities whereas E_1 entity may be associated with at most one E_1 entity.

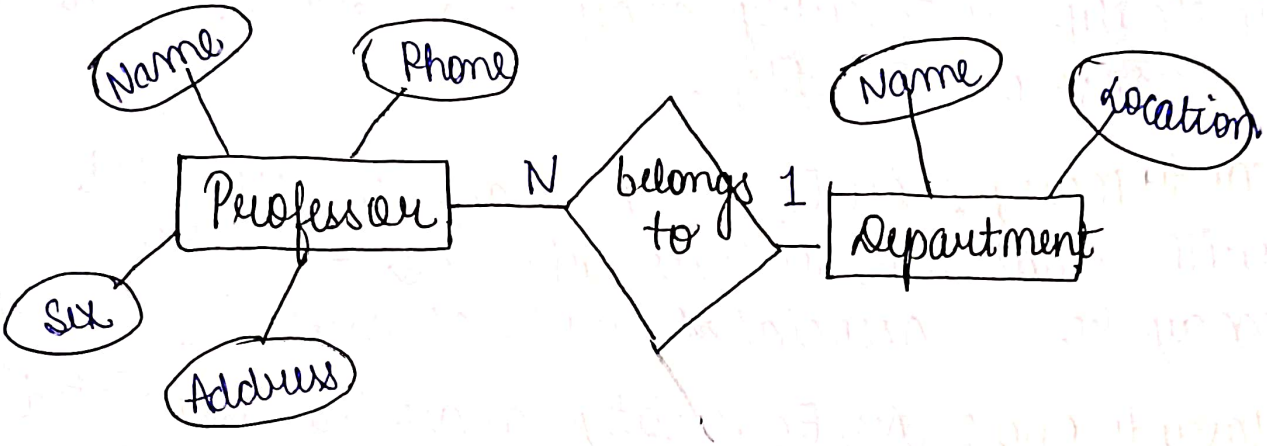
4) **Many to Many** : Many E_1 entities may be associated with a single E_2 entity and a single E_1 entity may be associated with many E_2 entities.

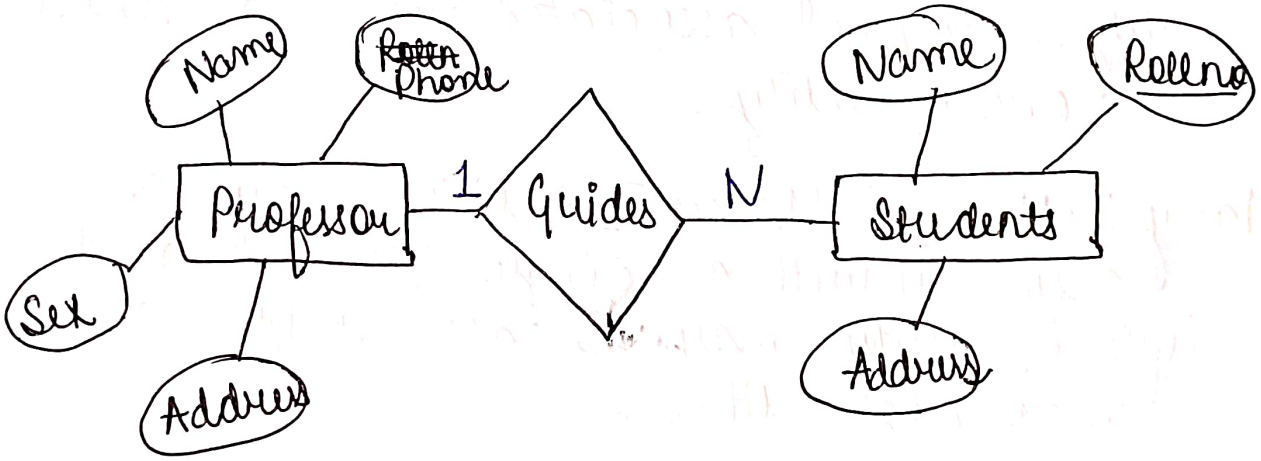
Example

1) 1 : 1

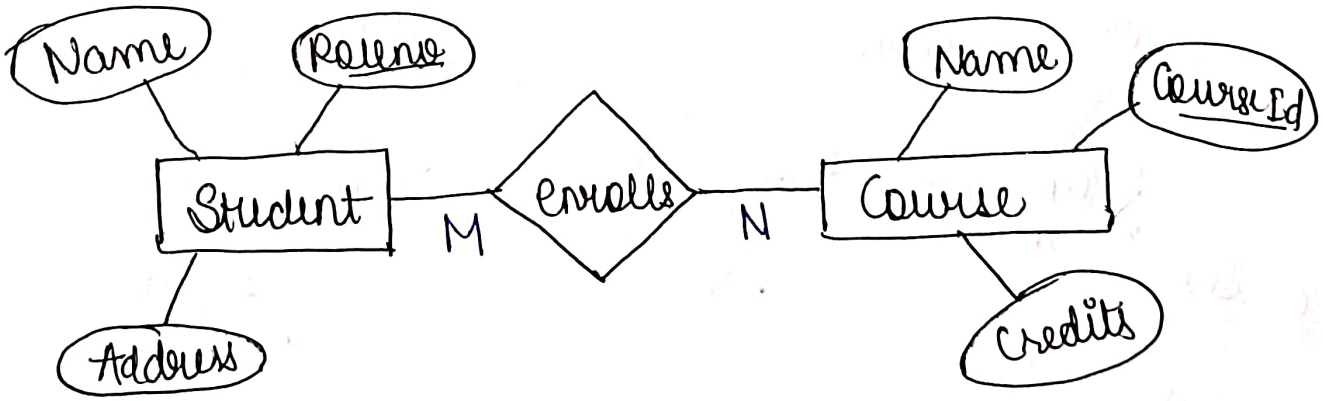


2) M:1 / 1:M


Name
Phone
Professor
N
belongs to
1
Name
Location
Department
Sex
Address


Name
Phone
Professor
1
Guides
N
Name
Rollno
Students
Sex
Address
Address

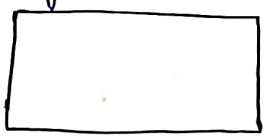
3) M:N


Name
Rollno
Student
M
enrolls
N
Name
CourseId
Course
Address
Credits

Entities

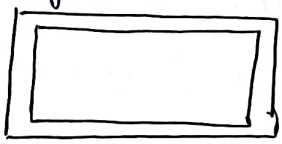
Strong Entity

An entity that has a primary key and does not depend on another entity.



Weak Entity

An entity that does not have a primary key and depends on a strong entity.

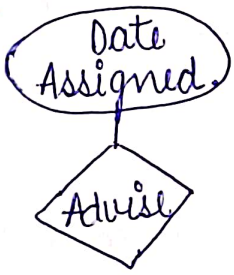


- **Relationship Attributes**

Attributes that belong to a relationship rather than an entity.

Notation → Oval attached to a relationship diamond.

Eg:



- **Alternative to Mapping Cardinality: Participation constraints**

→ Mapping cardinality does not always define exact numbers for entity relationship.

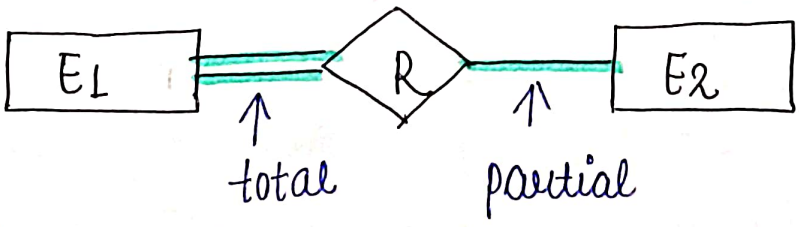
Solution – Participation constraints can specify additional rule on relationship.

- **Participation in Relationship**

Total participation - Every entity ... lved in some association (or ... ip.

Partial participation - Not all ... e involved in association (or tuples) of the relationship.

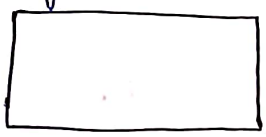
↳ Notation:



Entities

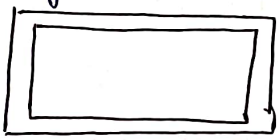
Strong Entity

An entity that has a primary key and does not depend on another entity.



Weak Entity

An entity that does not have a primary key and depends on a strong entity.

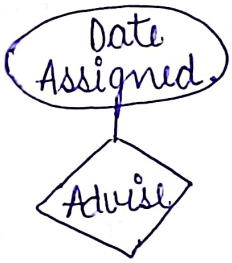


- **Relationship Attributes**

Attributes that belong to a relationship rather than an entity.

Notation → Oval attached to a relationship diamond.

Eg:



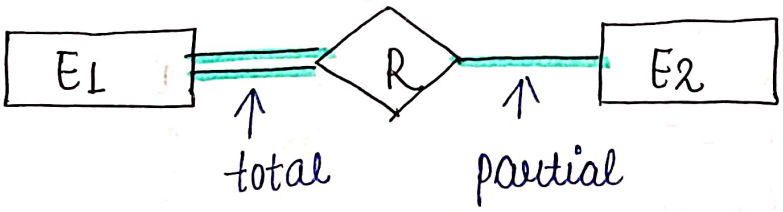
- **Alternative to Mapping Cardinality: Participation**

- **Participation in Relationship**

Total participation - Every entity in the set is involved in some association (or tuple) of the relationship.

Partial participation - Not all entities in the set are involved in association (or tuples) of the relationship.

↳ Notation:



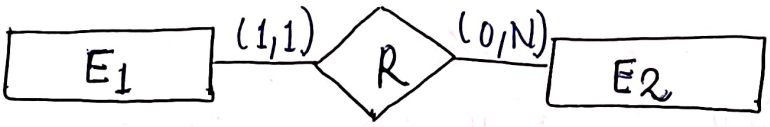
- It represents the minimum and maximum participation of entities in a relationship.

Eg → Total & partial
Employee - Department Relationship
→ Every employee must work in at least one Department.
→ Hence, employee has Total participation in the "Works in" relationship.
→ But a Department can exist without employees (eg a newly created department).
→ Hence, Department has partial participation in the "Works in" relationship.

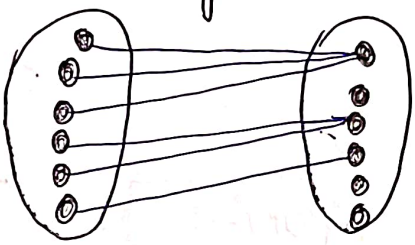
- Structural constraints
- Cardinality ratio and participation constraints are together called structural constraints.
- They are also called constraints as the data must satisfy them to be consistent with the requirement.
- Min-Max Notation : Pair of numbers (m, n) placed on the line connecting an entity to the relationship.
- m : the minimum numbers of times a particular entity must appear in the relationship tuples at any point of time.
  - 0 → partial participation
  - ≥ 1 → total participation
- n : similarly, the maximum number of times a particular entity can appear in the relationship tuples at any point of time.

comparing the Notation:

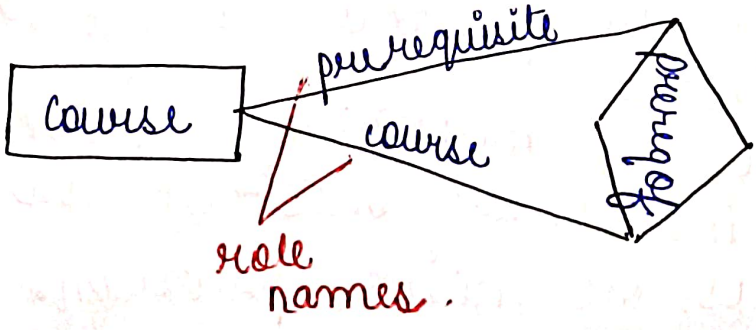
is equivalent to



Many to one.



- **Recursive relationship**
  An entity set relating to itself gives rise to a recursive relationship.
  eg. the relationship prereqOf is an example of a recursive relationship on the entity course.
- **Role Names**
  It is used to specify the exact role in which the entity participates in the relationship.
  - Essential in case of recursive relationship.
  - Optional in non-recursive case.

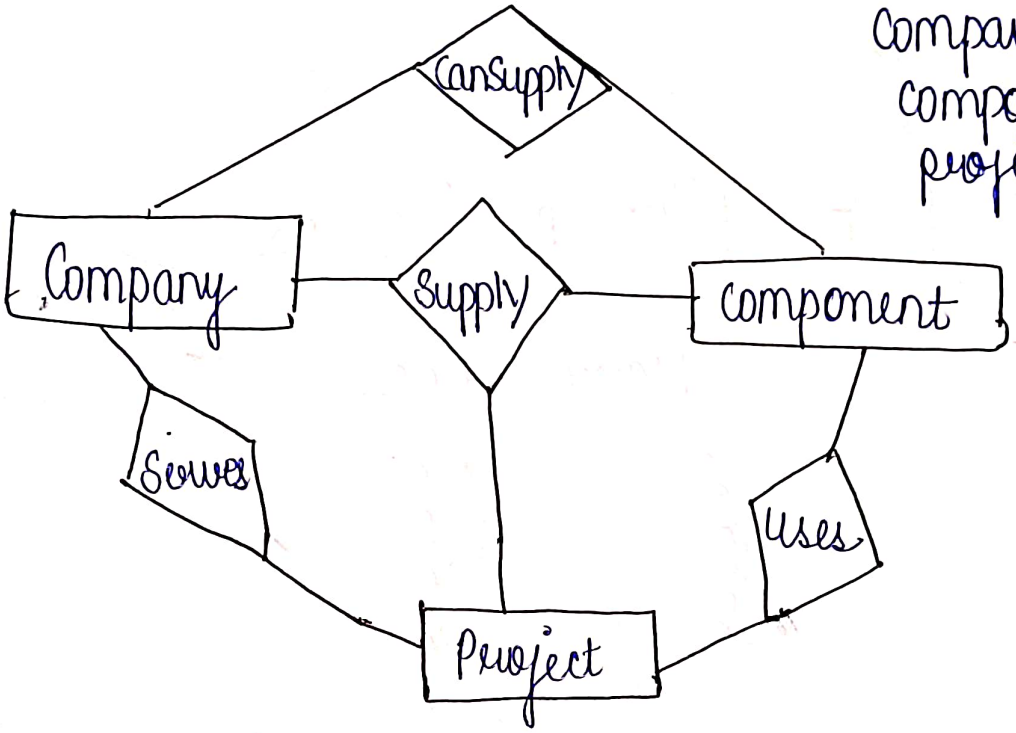


Ternary Relationship.

A relationship involving three entities.

Eg Instructor supervises students on a project.

Eg



Company(C)
Component(P)
project(j)

- Binary relationship
  - C(c, p) in canSupply
  - C(j|p) in uses
  - C(c, j) in serves

} The ternary C(c, p, j) is not derivable from any combinations of binary relationships.

"Binary relationship can't express the full meaning of the ternary relationship."

Even if -
- company canSupply Components
- Project uses component
- company serves Project.

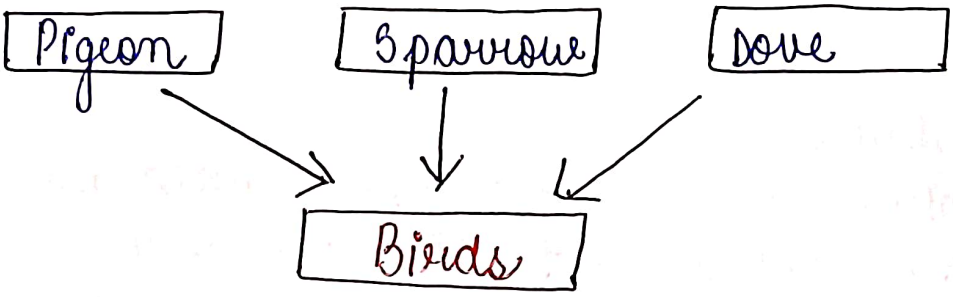
"It does not mean the company is supplying that component for that project."

Design implication=
Don't decompose a meaningful ternary relationship into binary ones - it can lead to data loss or misinterpretation.

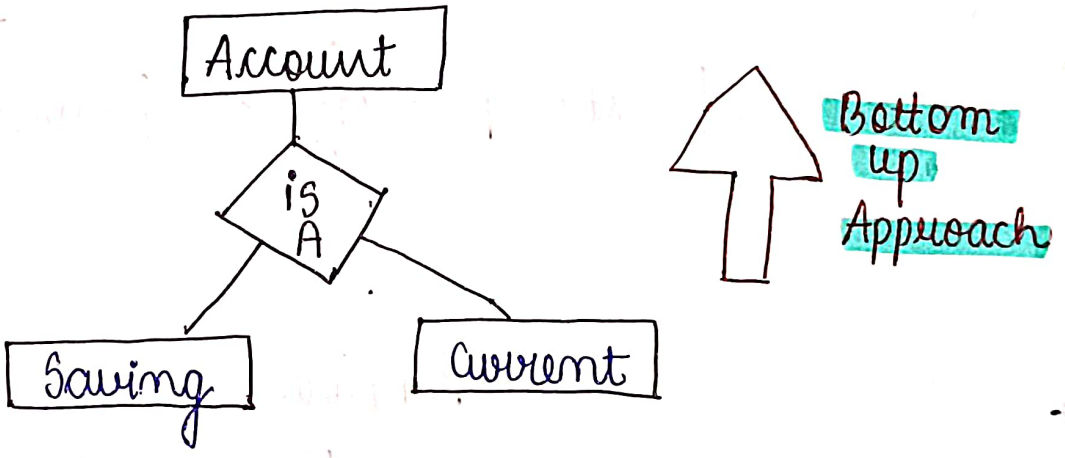
Generalization

The process of generalizing entities, where the generalized entities contain the properties of all the generalized entities, is called generalization. In generalization, a number of entities are brought together into one generalized entity based on their similar characteristics.

Eg 1)



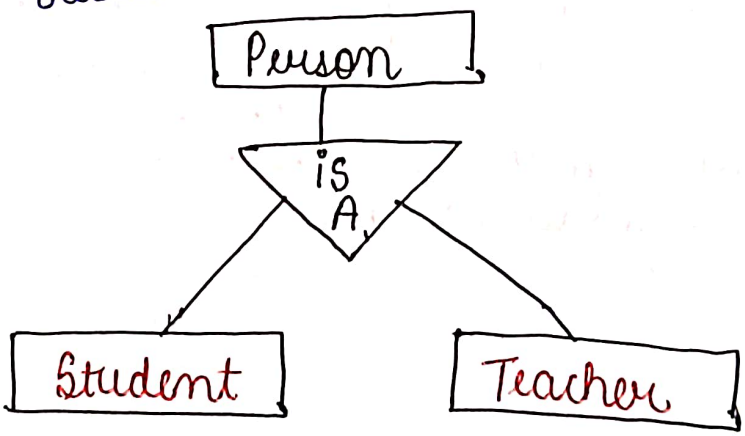
2)



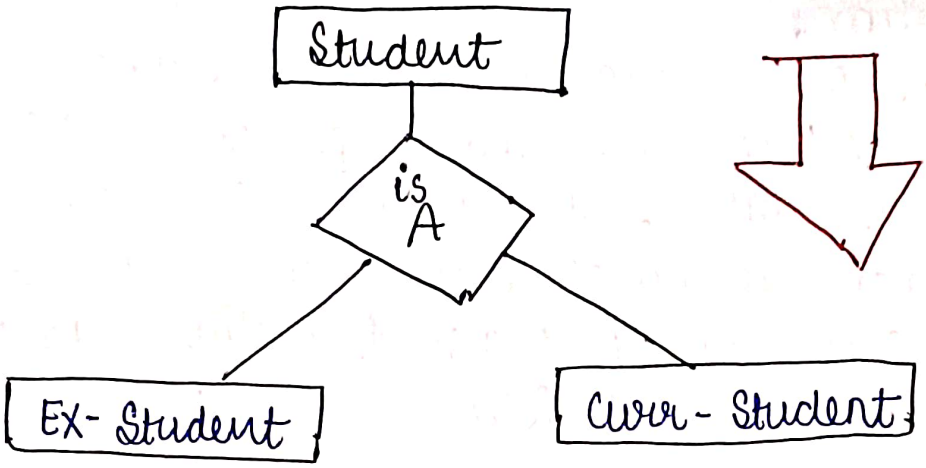
Specialization

Specialization is the opposite of generalization. In specialization, a group of entities is divided into sub-group based on their characteristics.

Eg 1)



2)

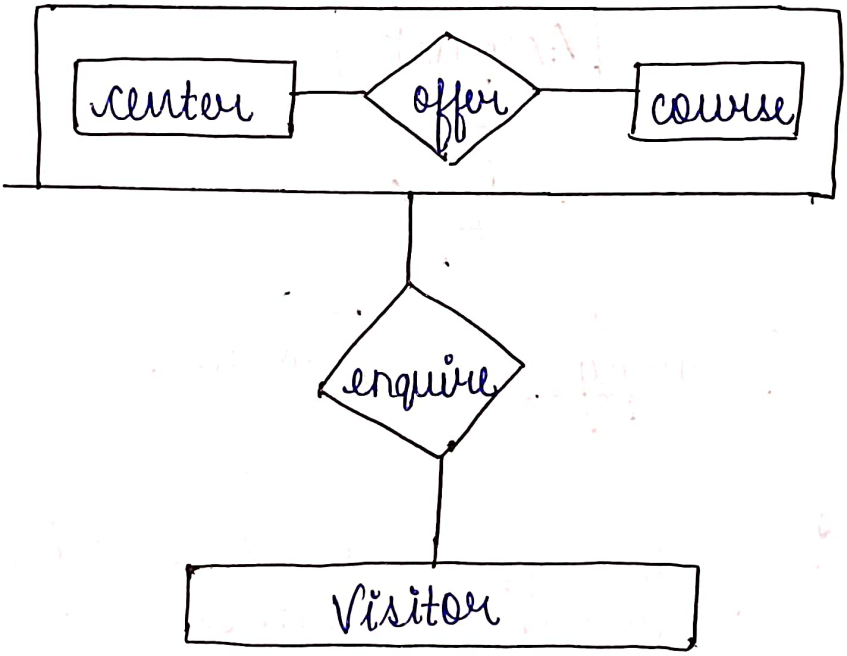


Top-Bottom/ Down Process

• Aggregation

Aggregation is a process when relation between two entity is treated as a single entity.

eg.



* → The E-R model cannot express relationships among relationships so we use aggregation. (Limitation of E-R.)

→ An abstraction through which relationships are treated as high-level entities.

Relationship → One photo
Relationship set → Album

(3) 1) FD & Normal form
(4) 2) Transaction & concurrency control
(1) (3) ER Model, FK, ER-RDMS
(2) 4) Query Language
(5) File Organization & Indexing

Unit - ER Model

Conceptual Model → Entity-Relationship Model

Object in the Real world.

- Entity set : Collection of similar entity which share same property.
  - Schema → Rectangle
  - eg STUDENT (RollNo. Name, gender)

1	Himanshu	M
2	Kapil	M
3	Dhaka	M

→ Entity

- Entities have (attributes) → ovals
  - → people have name & address

- Relationship – Association among the entity. → diamond ◇
- Relationship set – It is a mathematical relation among $n \geq 2$ entities, ~~each~~ each taken from entity set.

$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots e_n \in E_n\}$

where $(e_1, e_2 \ldots e_n)$ is a relationship

eg (4553, 2222) ∈ advisor

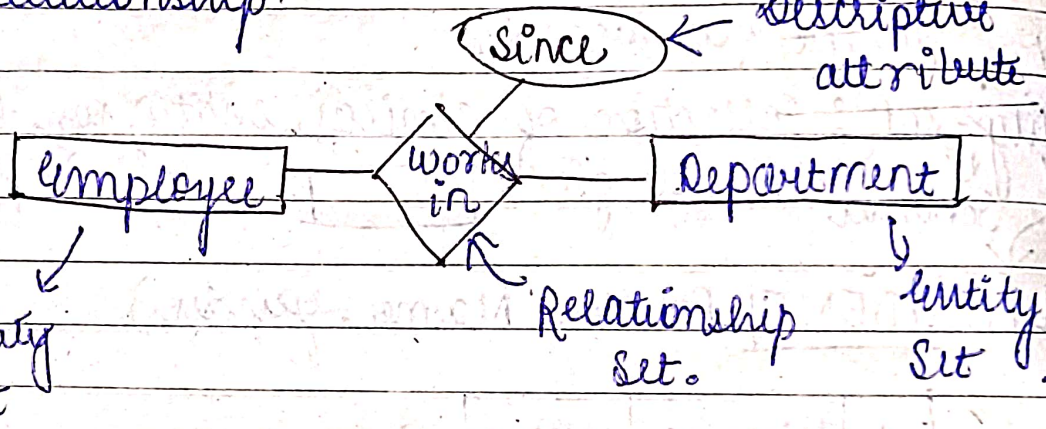
Teacher's Signature .....

eg

4553 (Peltier) — Student entity — advisor — relationship set — 2222 (Einstein) — Instructor entity

- Relationship set is collection of similar relationship.

eg



★ • An attribute can also be property of a relationship set.

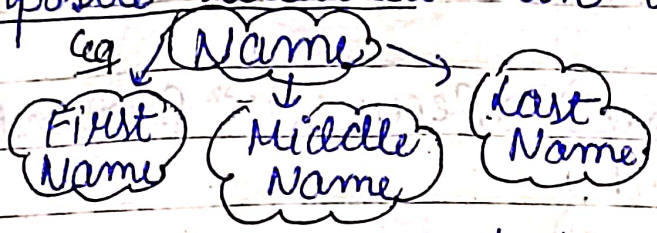
t • Attribute ⇒
Attribute are property used to describe an entity.

Types →

★ 1) Simple & Composite Attribute
2) Single valued & multi-valued attribute
3) Stored & Derived attribute
4) Key attribute
★ 5) Complex attribute
6) Descriptive attribute

Teacher's Signature.....

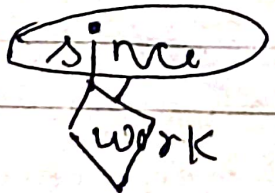
- Simple attribute can not be divided further eg Roll no
- Composite attribute can be divided further.

eg 

- Single valued attribute which takes only one value per entity eg gender, Round, Result
- Multi valued attribute which takes more than one value per entity eg MobileNo, E-mail, etc
- Stored attribute does not required any updation. eg DOB
- Derived attribute value can be derived from other attribute eg Age Year of Join.
- Complex attribute = Multi-valued attribute (MV) + composite attribute (CA)

eg Contact No
M.V.A → Sim 1, Sim 2
CA → STD + Tid / Country code + Tid

- Key attribute which uniquely identify an entity in the entity set eg Roll no
- Descriptive attribute gives information about the relationship set.

eg 

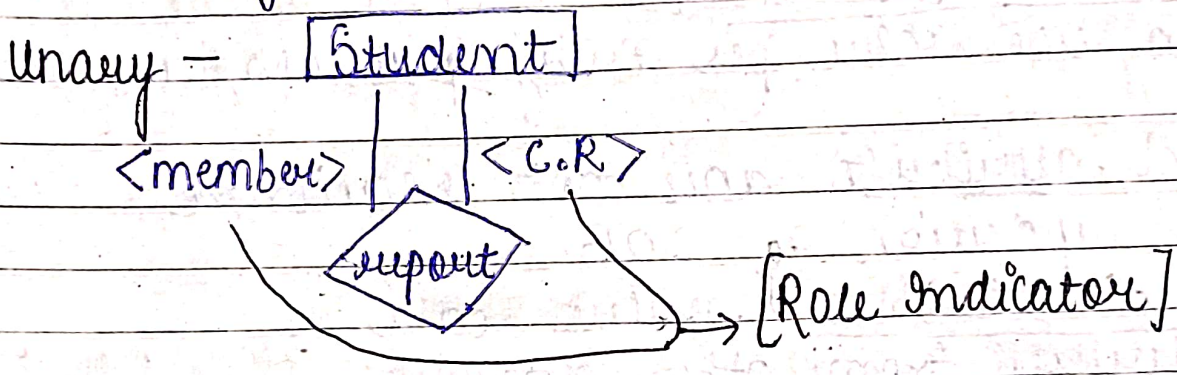
Teacher's Signature .....

→ No of entity set.

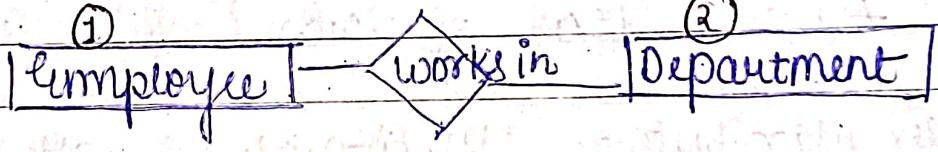
- Degree of Relationship set
  How many no. of entity set participate in relationship set!

→ Unary (1 entity set) ⇒ Recursive Relationship set
→ Binary (2 entity set)
→ Ternary (3 entity set)
→ n-ary (n entity set)

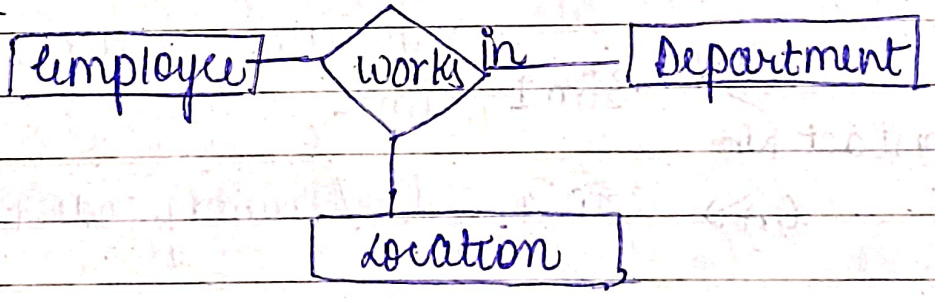
Unary –



Binary –

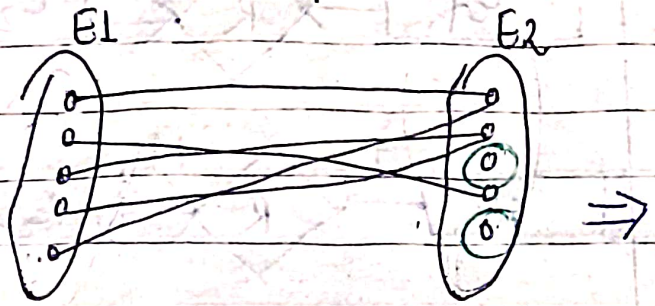


Ternary



- Participation
  1) Total participation (═)
     If every entity in the entity set participate in a relationship.

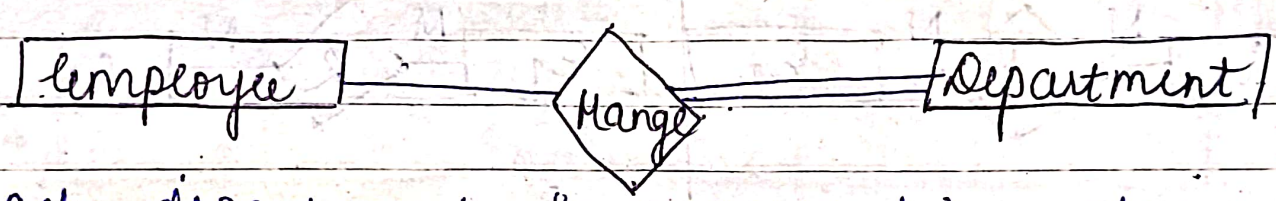
2) Partial Participation (—) participation (<100% participation)



Total / Full participation

Partial participation

eg



Each department is managed by at least one employee.

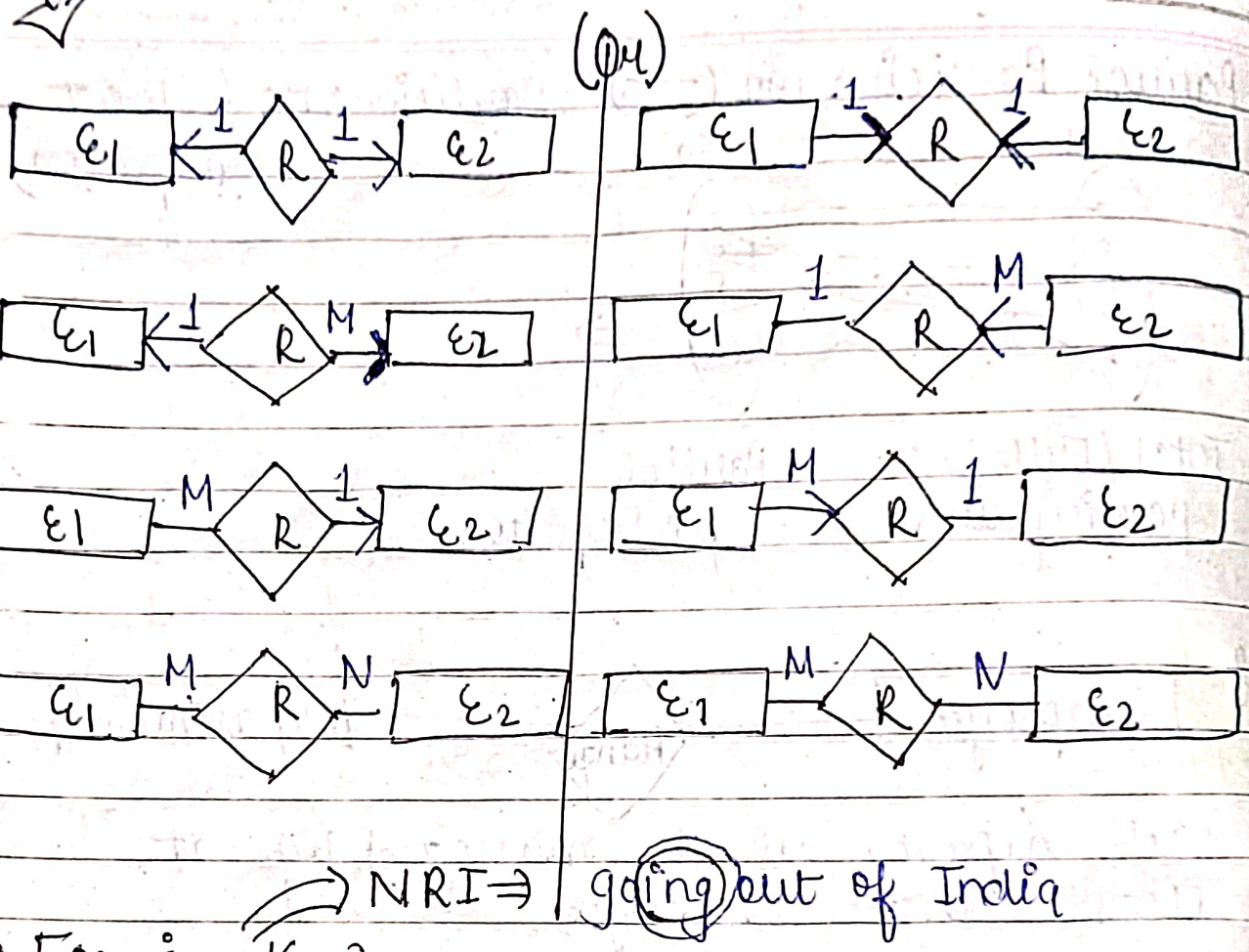
- Mapping cardinality constraints → Entity
  - Express the no. of entity to which another entity can be associated via relationship set.
  - Most useful in describing binary relationship set.

Types :-

1) One to One
2) One to Many
3) Many to one
4) Many to Many

- One Mapping : At most one. (0 or 1)
- Many Mapping : 0 or more (0 ... *)

Trick:- तीर जहा जा रहा है वो अकेला रहेगा।

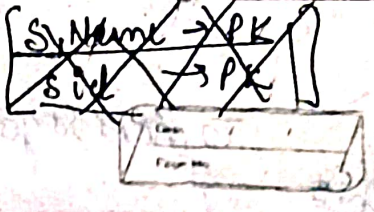


- Foreign Key
  Foreign Key is a set of attribute reference to the primary Key or alternative Key of same table or same other table.

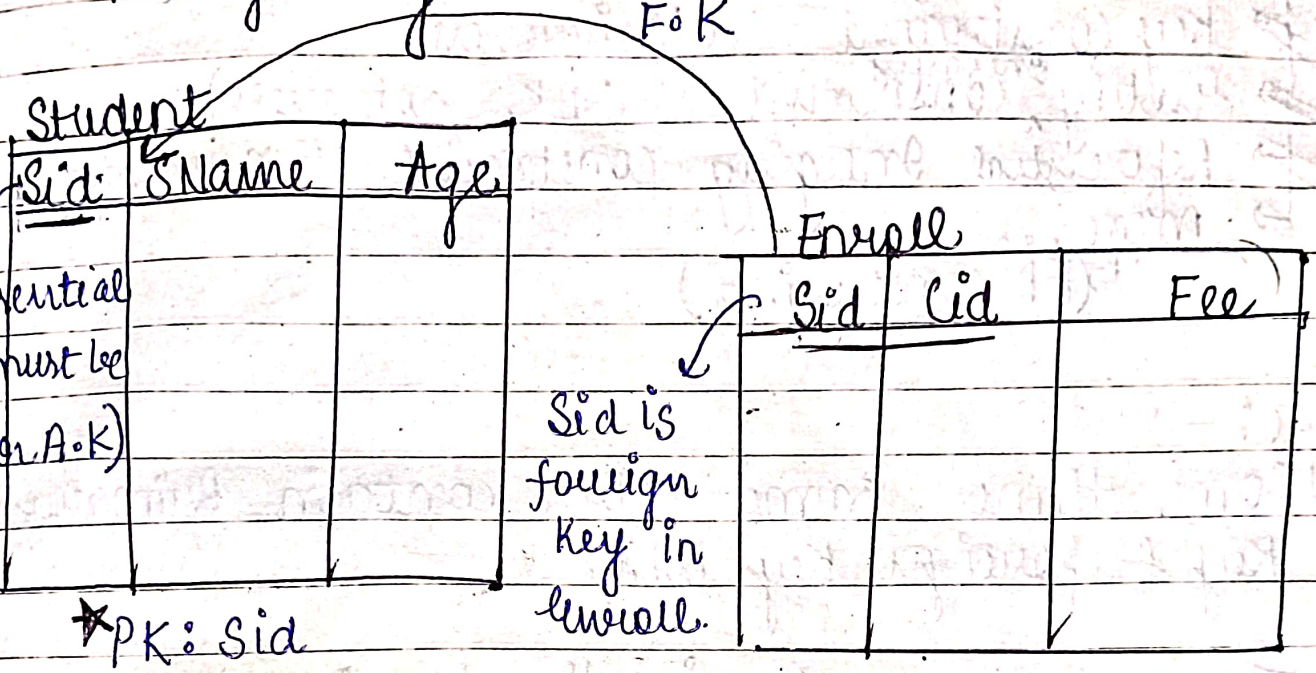
→ Referencing = Referencing relation is the table which contain the foreign Key. [CHILD Table / Relation].

→ Refrenced = The table which is referenced by foreign Key is known as Referenced relation. [Parent Table / Relation]

Teacher's Signature



• Referential Integrity Constraints
Foreign Key



⇒ By default F.K reference to to P.K of referenced Relation.

1) Note :- At most one Primary Key per relation (Table)
↳ (Unique + Not Null)

So Directly write Name of Table

Foreign Key (Sid) Reference Student

2) Note :- Assume Loginid is Primary Key Sid is Alternative Key of Student Table.

Foreign Key (Sid) Reference STUDENT (Sid) → when Sid is Alternative Key.

Teacher's Signature

To make legal Instance

RDBMS

- Constraints Set of rules applied on schema.
  - Domain Constraint (Atomic)
  - Key constraint (Uniqueness)
  - Entity Integrity constraint (P.K not null)
  - Referential Integrity constraint (F.K concept)
  - More...
    - (FD constraints)

Note:-

1) Some time same table contain Primary Key & foreign Key both.

2) The value present in the foreign key must be present in the primary key of referenced Relation.

3) Foreign key may contain Duplicate & NULL VALUES.

Parent Table	CHILD table
✓ Insert <4 D ECE>	✗ INSERT <105 DSA 67>
✗ Delete <1 A CSE>	✓ DELETE <103 CD 37>

- Deletion from the Referenced Relation & Insertion into referencing relation may violate FK constraint.

- A relation can act as Parent & CHILD ie Relation may contain PK & FK that refer to the same relation.

Teacher's Signature

1) Referencid Relation
   i) Insertion : No violation
   ii) Delete : May cause violation if primary key is used by referencing relation.

   I ON DELETE NO ACTION.
   II ON DELETE CASCADE.
   III ON DELETE SET NULL.

a) On delete on action - Means if it cause problem on delete then deletion is not allowed on table.
b) On delete cascade - If we want to delete primary key value from referenced table then it will delete that value from referencing table also.
c) On delete set null - If we want to delete primary key value from referenced table then it will try to set the null value in place of that value in referencing table.

* Note on :- On Delete Cascade
- When Primary key deletes then corresponding Tuple (value) from the referencing relation delete cascadely.
- for good DB design, sometimes on delete cascade not suggested because deletion of one Tuple ~~Table~~ (on delete cascade) result leads to deletion of complete table.

Eg If we want to delete (E_2, Null) with on delete cascade then how many no. of additional deleted to preserve Referential Integrity?

```
foreign key (mgr id) Reference emp ON
                DELETE  CASCADE
```

emp

eid	ename ...	manager id
E_1		E_2
~~E_2~~		~~NULL~~
E_3		E_2
E_4		E_3
E_5		E_4
E_6		E_5
E_7		E_6

- (E_2 Null) PK E_2 Deleted
- a) Due to PK $E_2 \Rightarrow [E_1\ E_2]$ $[E_3\ E_2]$ Deleted
- Now P.K E_1 & E_2 deleted
- b) Due to PK $E_3 \Rightarrow [E_4, E_3]$ deleted
- Now pK E_4 deleted
- c) Due to p.k $E_4 \Rightarrow [E_5, E_4]$ deleted
- Now pK E_5 deleted
- d) Due to pK $E_5 \Rightarrow [E_6, E_5)$ deleted
- Now pK Key E_6 deleted
- e) Due to p.K $E_6 \Rightarrow$ $[E_7, E_6]$ deleted.

p.k emp

eid	ename	manager id
~~E_1~~		~~E_2~~
~~E_2~~		~~Null~~
~~E_3~~		~~E_2~~
~~E_4~~		~~E_3~~
~~E_5~~		~~E_4~~
~~E_6~~		~~E_5~~
~~E_7~~		~~E_6~~

∴ All tuple deleted.

- ER to Relational Model conversion
ER diagram used to represent diagrammatic design [High-level diagram] of Database.

ER Model = Conceptual Design
↓
RDBMS

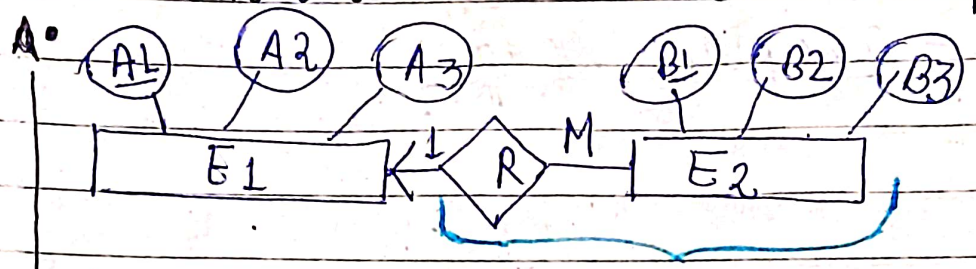
Db design
a) Requirement: what type of data stored & what operation required etc.
b) conceptual & logical: ER Diagram
ic) ER Diagram to RDBMS table design & apply Normalization.
d) Physical Db design (Indexing Design)
e) user interface design & Security design.

- In RDBMS,
No multivalued, no composite attribute or weak entity concept.

ER	→	RDBMS
(1) Entity set	→	Relation (Table)
(2) Key attribute	→	Primary Key
(3) Entity	→	Tuple.
(4) Composite Attribute	→	Set of simple component
(5) Multivalued attribute	→	(1) Key attribute + all other attribute (Table) (2) key attribute + all multivalued attribute

☆

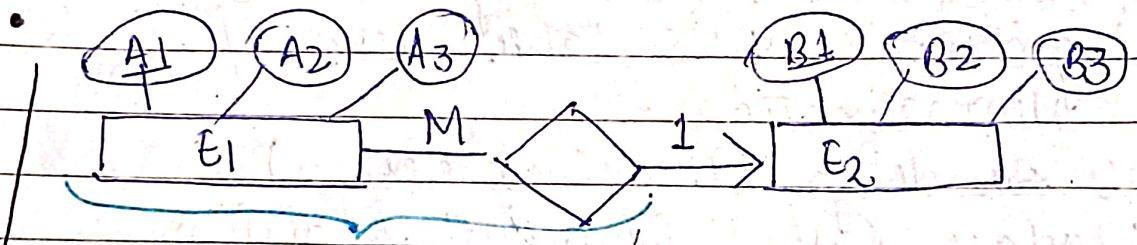
1 To Many	→	2 Table	(Relationship set merge with many side)
Many to one	→	2 Table	"
Many to Many	→	3 Table	(separate table for relationship set)
one to one	→	2 Table	(Relationship set merge any side)



→ 2 Table required & 1 Foreign key

F.K

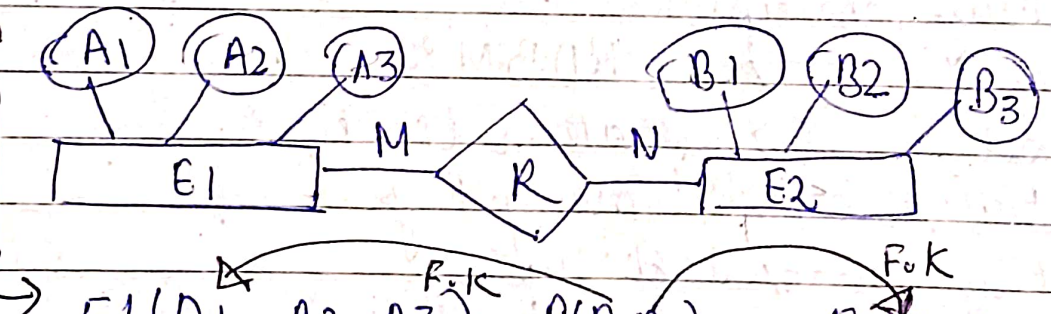
$E_1(\underline{A_1}, A_2, A_3)$ $RE_2(\underline{B_1}, B_2, B_3, A_1)$



2 Table required & 1 F.K

F.K

$RE_1(\underline{A_1}, A_2, A_3, B_1)$ $E_2(\underline{B_1}, B_2, B_3)$

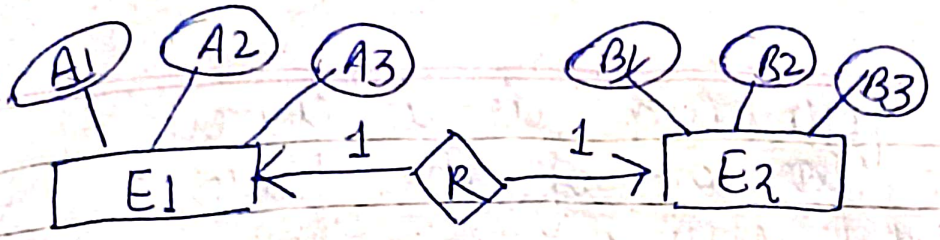


F.K F.K

$E1(\underline{A_1}, A_2, A_3)$ $R(A_1, B_1)$ $E2(\underline{B_1}, B2, B3)$

3 Tables required & 2 Foreign Key.

☆☆



↳ Table Required ↓ 1 F.K

F.K

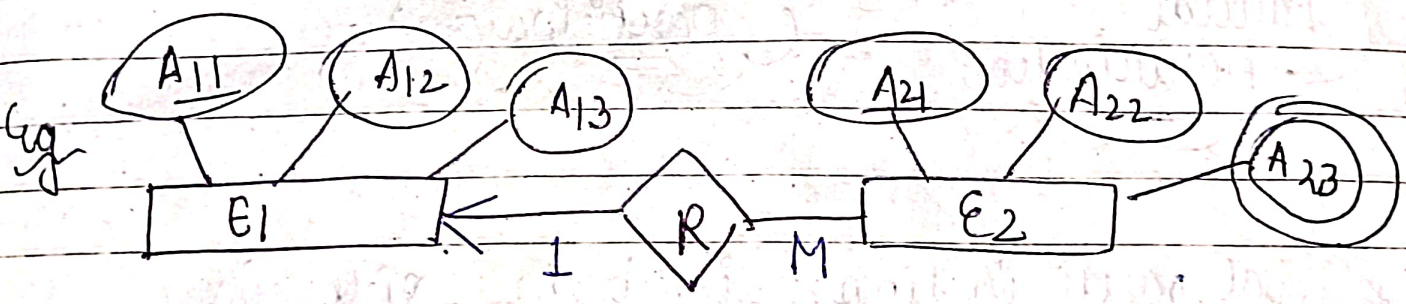
$E_1R(\underline{A_1}, A_2, A_3, B_1)$ ↓ $E_2(\underline{B_1}\ B_2\ B_3)$

(OR)

$E_1(\underline{A_1}, A_2, A_3)$ ↓ $RE_2(\underline{B_1}\ B_2\ B_3\ A_1)$

F.K

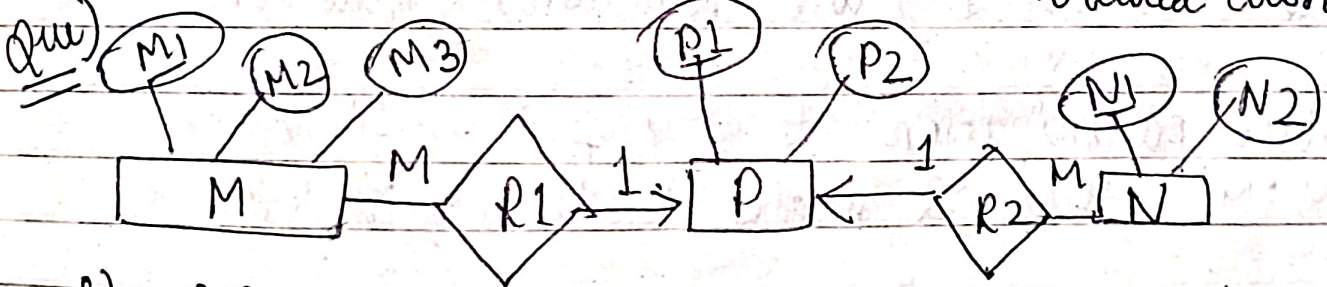
eg:



→ $E_1(\underline{A_{11}}, A_{12}, A_{13})$
$RE_2(\underline{A_{21}}, A_{22}, A_{11})$
$E_2'(A_{21}\ A_{23})$

- Relationship R does not have any attribute
- A23 is a multi valued attribute.

Qu)



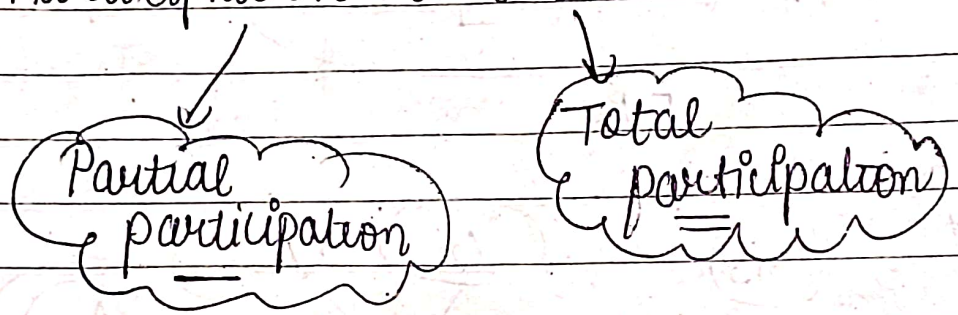
i) Minimum no. of tables required to represent M, P, R_2, N, R_1

→ $MR_1(\underline{M_1}\ M_2\ M_3\ P_1)$
$P(P_1, P_2)$
$NR_2(\underline{N_1}, N_2 \& P_1)$

ii) which of the following is a correct attribute set for one of the table for the correct answer to the above question?

⇒ {M1, M2, M3, P1}

- Participation constraints



1) If Total participation in both side in mapping cardinality then 1 table required.

2) If total participation at one (1) side (1:M / M:1) then only 1 Table (Relation) Required.

3) If total participation at any of one side (M:M) then 2 Tables are required.

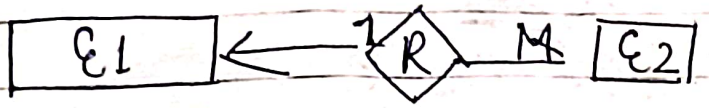
4) If all other are same as normal.

☆ Total participation cases:

= (tp) = total participation

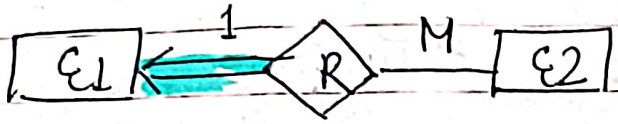
= (pp) = partial participation

1) 1 : M



2 Table

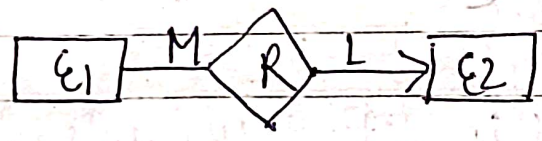
2 Table



1 Table

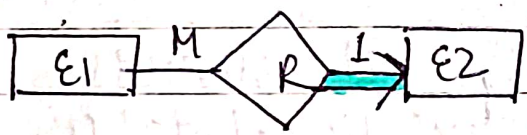
1 Table.

2) M : 1

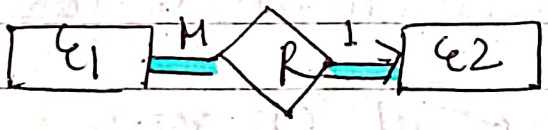


2 Table.

2 Table

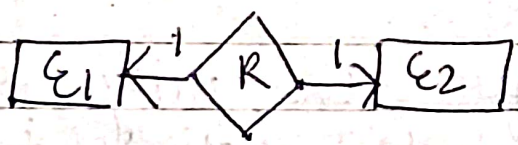


1 Table

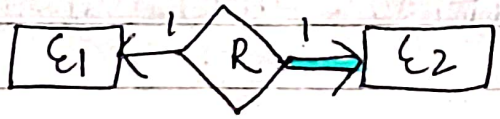


1 Table

3) one : one (1 : 1)



2 Table



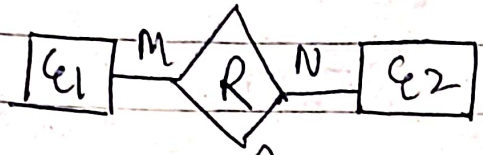
1 Table

1 Table

1 Table

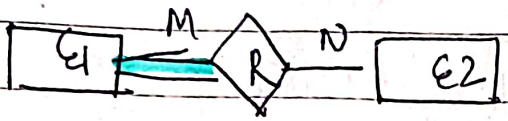
Teacher's Signature .....

4) M:N

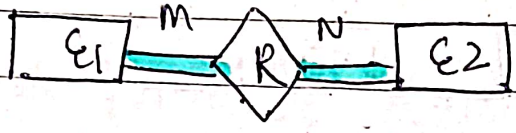


3 Table

2 Table



2 Table



1 Table.

★ Partial participation on both side of binary relationship.

1) 1:M - Merge relationship set towards many side So, 2 relational table.

2) M:1 - Merge relationship set towards many side So, 2 relational table.

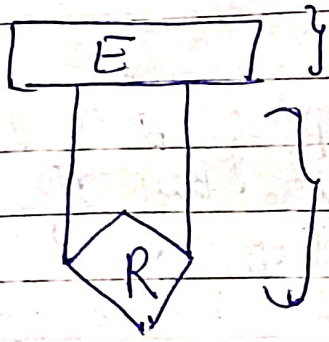
3) 1:1 - Merge relationship set at any side, So 2 relational table.

4) M:M - Separate table for each entity set & relationship set, so 3 relational table.

Teacher's Signature.....

• Self-Referential Relationship set

(Recursive entity set)
Entities of entity set (E) related to some other entity of same entity set (E)



} Recursive entity set

} Self-Referential Relationship set

- Binary relationship set.
- 1:M, M:1, 1:1, ~~M:M~~ can be any of the relationship.

↓

∴ Only one Table required.

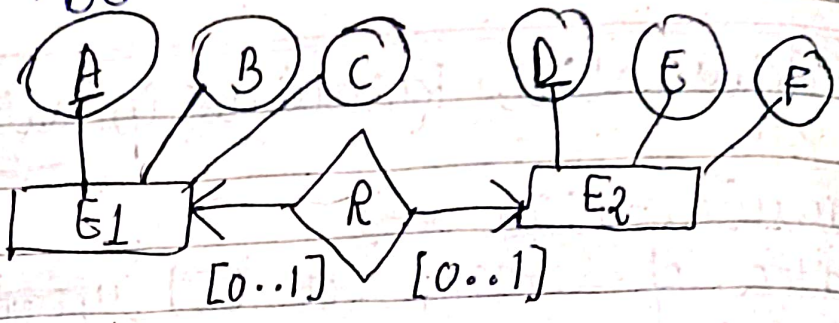
- M:M ⇒ 2 Table required.

---

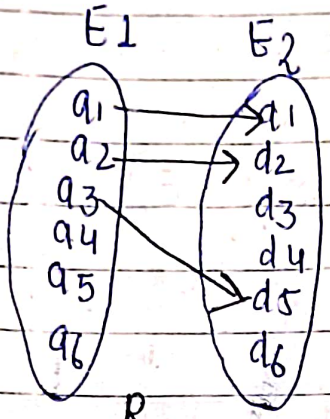
1) How to find Candidate Key of Relationship set?
2) Merge concept why?

One : [At Most one] [0...1]
Many : [0 or more [0...M]

$A \to BC$ $D \to EF$



★ 1:1



R

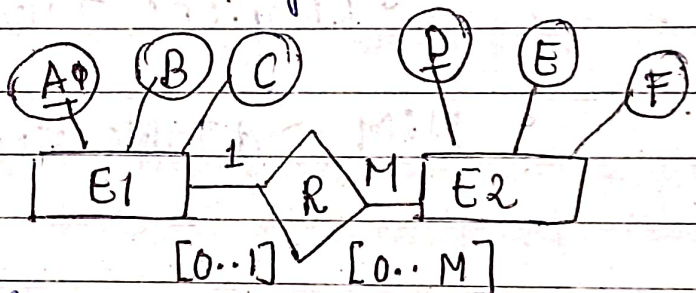
A	D
a_1	d_1
a_2	d_2
a_3	d_5

$R(\underline{A}, \underline{D}) = \{A, D\}$

a_1 d_1 1:1 mapping.
a_2 d_2 candidate key of
a_3 d_5 Relationship set.

- Candidate Key = A / D [either A or D]

★ 1:M



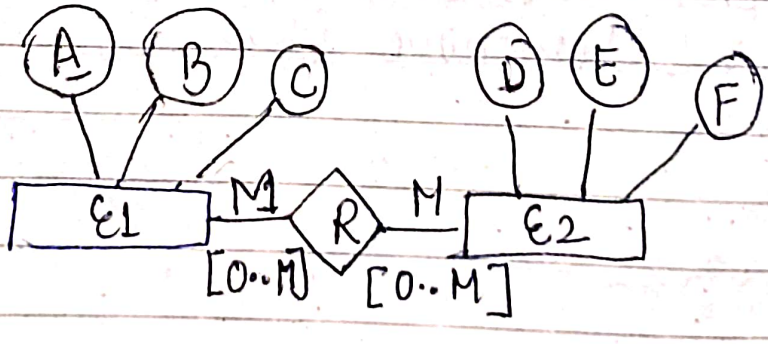
- Candidate Key of R(AD) = {D}

A	D
a_1	d_1
a_1	d_2
a_2	d_3

Each object of E_2 allowed to pair by atmost one entity of E_1

Each entity of E_1 can pair with many E_2 entity.

★ M:M



R(A,D) = {AD}

A	D
a_1	d_1
a_1	d_2
a_2	d_2
a_3	d_3

for Many to Many mapping,
- Candidate key for relationship set is {AD}

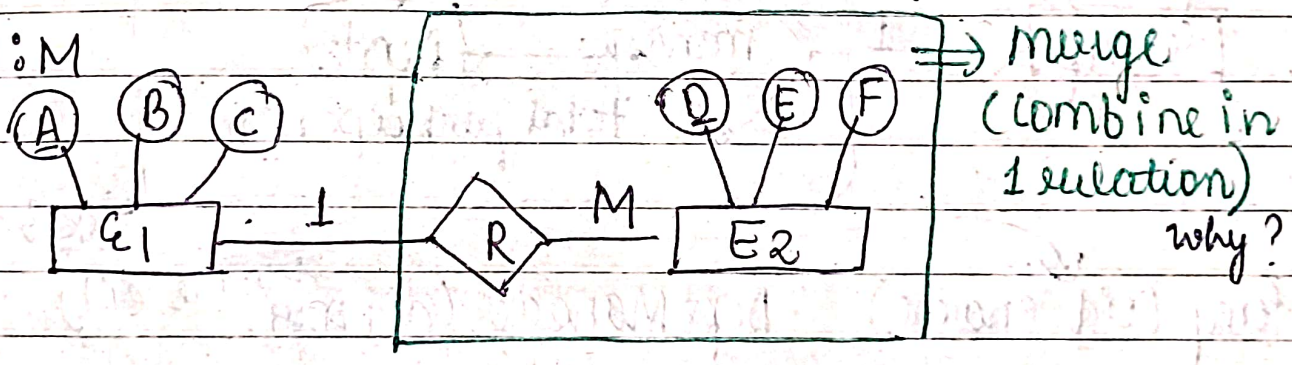
Summary

Candidate key (CK)

1:1 — A or D
1:M — D
M:1 — A
M:N — AD

- RDBMS Design of given ER diagram:-

a) 1:M



⇒ merge (combine in 1 relation) why?

E1 (ABC)	R(AD)	E2 (DEF)
a_1	a_1 d_1	d_1
a_2	a_1 d_2	d_2
a_3	a_2 d_3	d_3
a_4		d_4
$A \rightarrow BC$	$D \rightarrow A$	$D \rightarrow EF$
C.K {A}	{D}	{D}

{A → BC} E_1(ABC)

a_1 - -
a_2 - -
a_3 - -
a_4 - -

E_2R(DEFA)

d_1 - - a_1
d_2 - - a_1
d_3 - - a_2
d_4 - - NULL

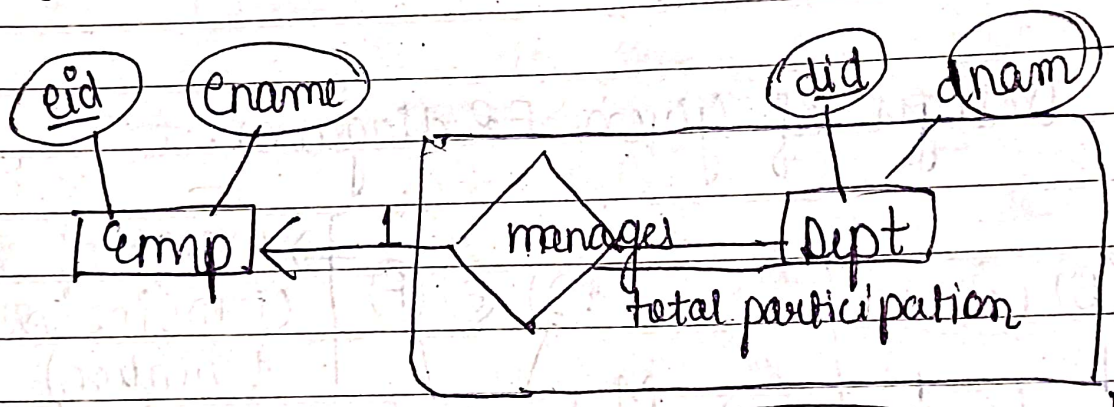
{D → AEF}

Partial participation b/w E_2 & R

due to partial participation (F.K can have null)

- Minimum 2 relational table & 1 F.K required for given data.

eg ERD



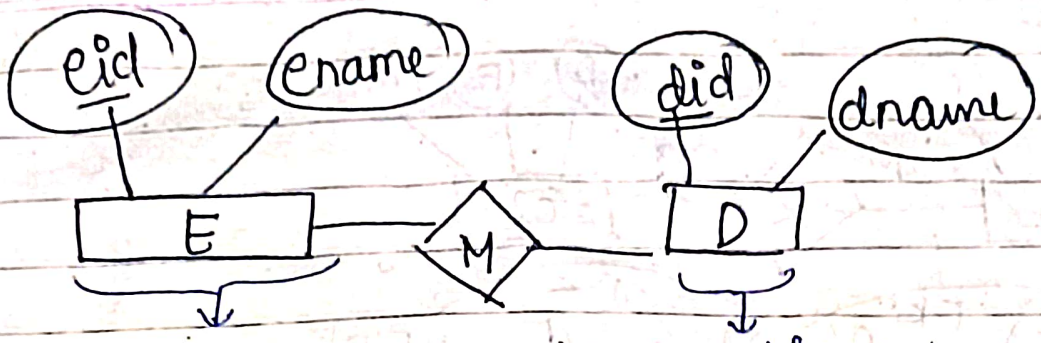
F.K 3

Emp(eid ename) Dept Manager(did dname eid)

d_1 - - e_1
d_2 - - e_2
d_3 - - e_3
d_4 - - e_4

NOT NULL & all as total participation at E_2 side

= Why not merge at 1 side in (1:M)?



eid → ename

did → dname

did → eid

E_M(eid ename did) F.K D(did dname)

Not Key	e_1	A	d_1
	e_1	A	d_2
	e_1	B	d_2
	e_1	C	d_3
	e_1	D	Null

Not valid as did is Key of E_M

(Partial Participation) [lost at E side]

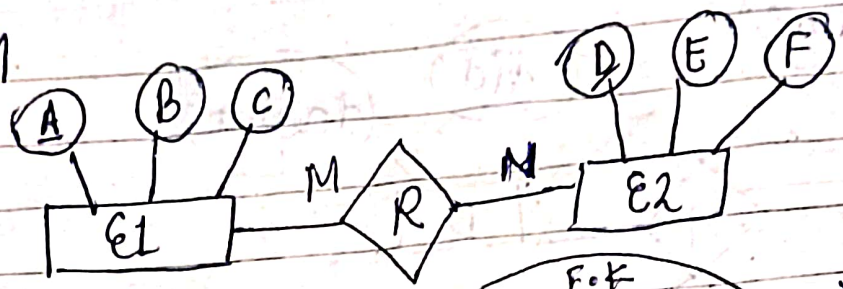
{If eid did Key} So, Not possible.

Disadvantage:

1) Data redundancy occur
   eid → ename
   Not Super Key (S.K)

2) Partial participation will be lost.
   [Not possible to insert employees who are not manager of dept.]

b) M:M



F.K F.K

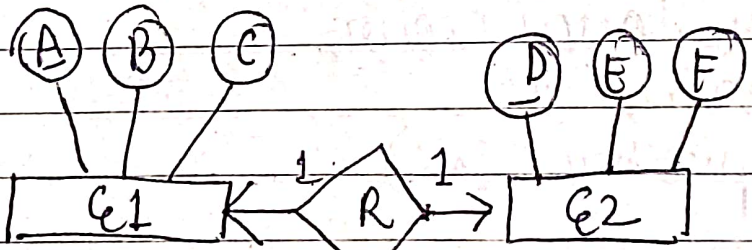
E1(A B C) R(A D) E_2(D E F)

E1	R	E_2
a_1	a_1 d_1	d_1
a_2	a_2 d_2	d_2
a_3	a_2 d_2	d_3
↳ Key	↳ Not key	↳ Key

Minimum 3 relational table and 2 foreign key required. M:N relationship set not allowed to merge with any entity set.

c) One: One

1:1 + partial participation



E1(A B C) R(A D) E_2(D E F)

E1	R	E_2
a_1	a_1 d_2	d_1
a_2	a_2 d_3	d_2
a_3	a_3 d_4	d_3
a_4	Key	d_4
Key		Key

[A → BC]

[A → D
D → A]

[D → EF]

⇒ $E_1(\underline{A}BC)$ & $RE_2(DEF\underline{A})$

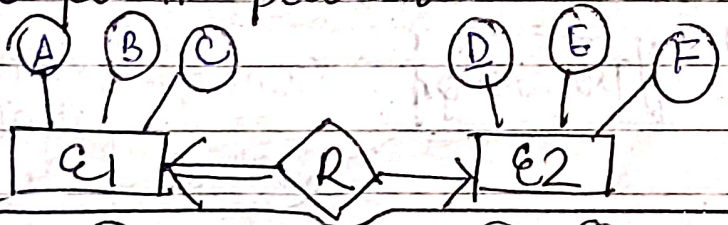
If we merge with singer table:

A	B	C	D	E	F
a_1	—	—	d_2	—	—
a_2	—	—	d_3	—	—
a_3	—	—	d_4	—	—
a_4	—	—	NULL	NULL	NULL
NULL	NULL		d_1	—	—

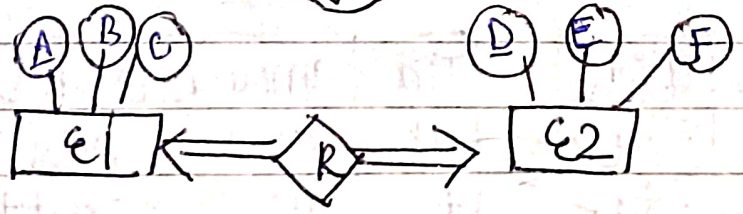
Candidate Key = {A, D}

NO candidate key is present, & no primary key. A relational table in which no attribute having 'NOT NULL' values are NOT allowed in RDBMS.
So, minimum 2 Relational Table required & 1 Foreign key.

Total participation :-



$E_1RE_2(\underline{A}\,B\,C\,\underline{D}\,E\,F)$



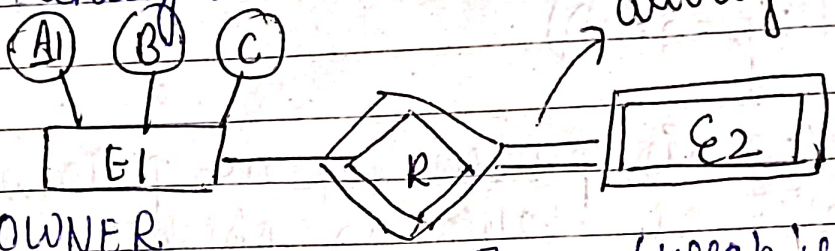
$E_1RE_2(\underline{A}BC\underline{D}EF)$

P.K ↙ ↘ unique & Not null (Alternative Key).

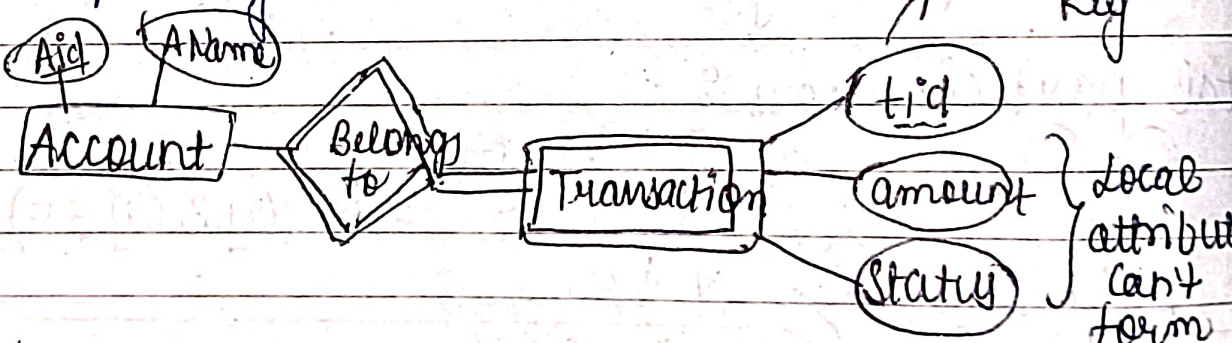
Teacher's Signature .....

- Weak Entity set
  - Entity set with no key. [attribute of weak entity set not sufficient to differentiate entities uniquely]

- For each weak entity set there must be OWNER entity set which is strong entity.



- Entities of weak entity set are depending entities.



Aid	AName
A1	
A2	
A3	
A4	

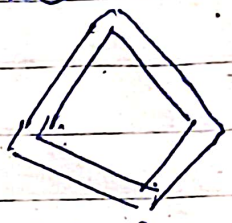
Aid	Tid
A1	t1
A2	t1
A3	t2
A3	t2

Tid	amount	Status
t1	5000	Debit
t1	4000	Credit
t2	5000	Debit
t2	5000	Credit

[1:M] Ambiguity

Teacher's Signature

- Relationship set between weak entity set & identifier entity is also a weak relationship set.



- Participation towards weak entity set ~~And~~ end must be Total participation.

- Mapping between identifier set must be one : many (1 : m)

RDBMS design :

Transaction belongs (Aid tid amount state)
Account (Aid AName)

NOTE : Weak entity set & multivalued attributes allowed to represent in ERD, but not allowed in RDBMS Table.