DBMS v/s File System

→ Data Redundancy & inconsistency

DBMS is a centralised system where same data will be accessed by different nodes. Which means reducdancy of data will not be there in this system.

→ Difficulty in ~~f~~ accessing data

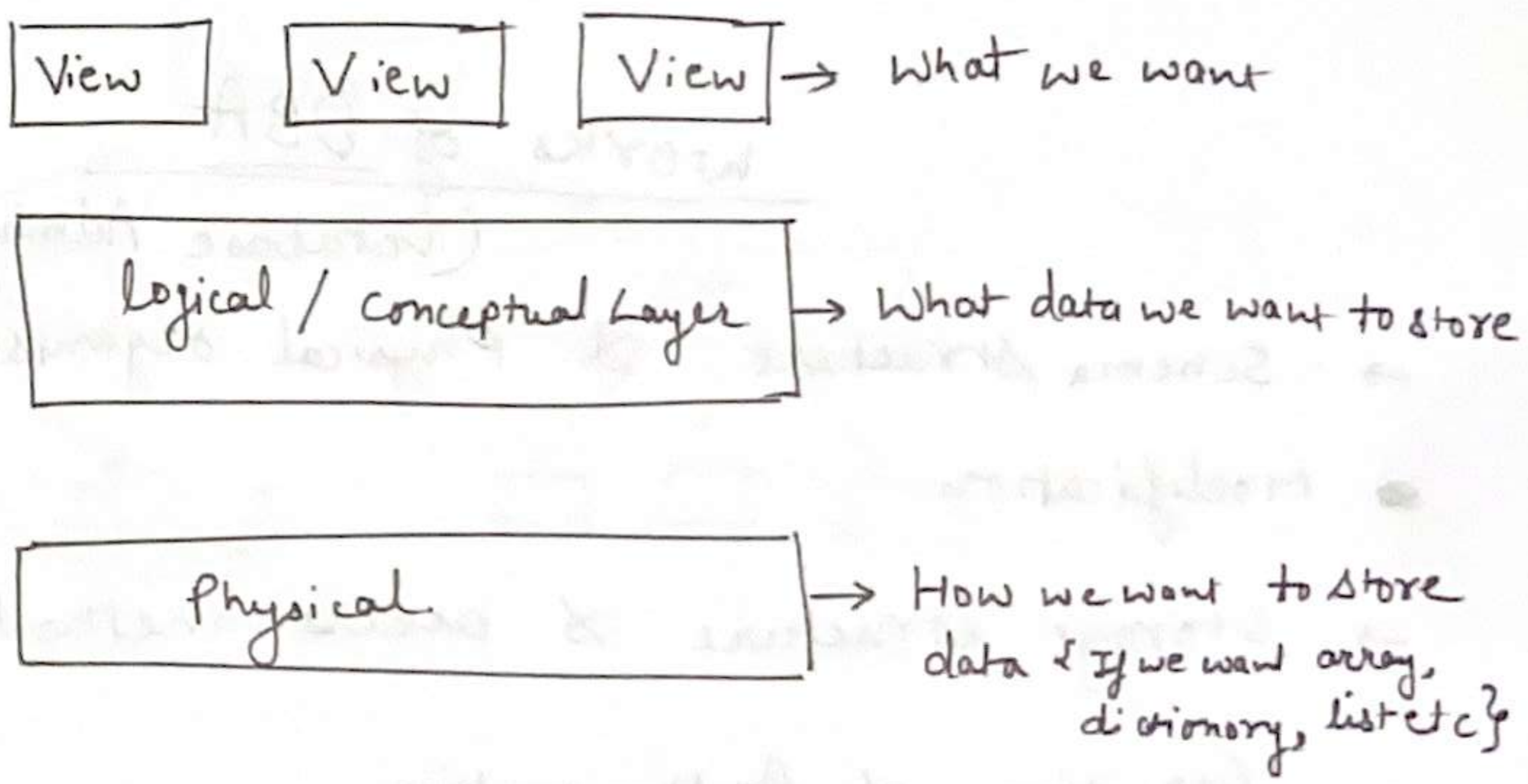
→ Integrity Problem

→ Security

→ Concurrent access anomalies

2|2|2024

3-layer architecture ⇒


View
View
View
→ What we want
Logical / conceptual Layer
→ What data we want to store
Physical
→ How we want to store data {If we want array, dictionary, list etc}

Data Independency

① Physical data Independence →

Changes done on physical layer in a way that no Changes are required on next higher layer.

② logical data Independence →

Changes in logical layer in a way that ~~so~~ no changes are required on next higher layer

Data users :→

①→ Native users { No idea of DBMS }

(No access to DBMS directly)

②→ Application programmers { have limited access of ~~DBMS~~ Data } but have no idea of DBMS itself

③→ Sphosticated users { have idea of DBMS, but have limited rights }

④→ Specialised users {DBA} – has all the rights

Works of DBA
(Database Administrator)

→ Schema structure & Physical organisation.

• Modification

→ Storage structure & access method modification.

→ Granting of Authorisation

→ Routine Maintenance.

Schema ⇒

Student

Roll. No.	Name	F. Name	DOB	Mob
1	A	B	–	–
2	.	–	–	–
3	–	.	–	–

Schema (structure) ← {

Instance {

{ data ~~to~~ in the DB at a particular time }

Database Models.

→ Hierarchical database

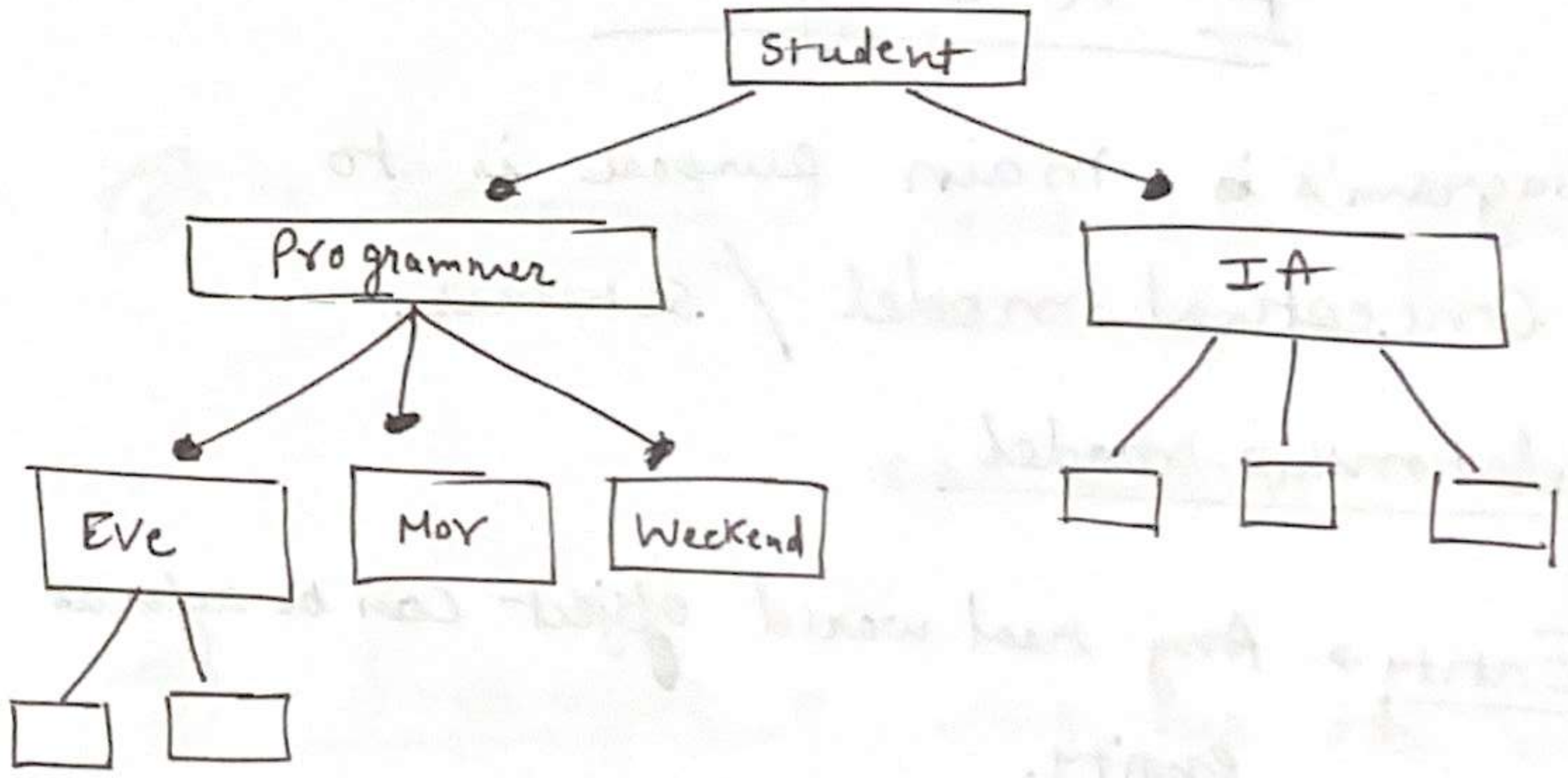
→ Network database

→ Relational database

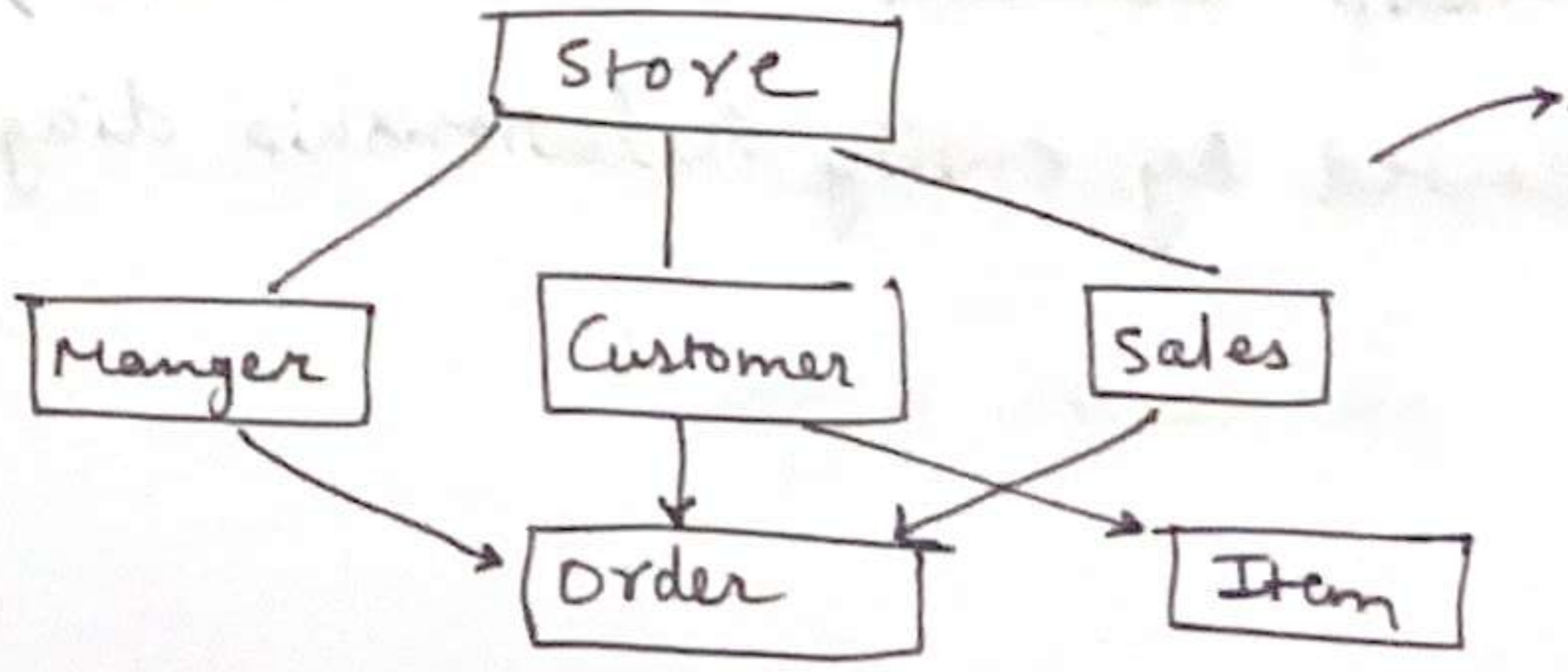
→ Object oriented database

→ Distributed database

① Hierarchical database →



② Network database →



{ When hierarchy closes or forms a closed space it becomes network database }

③ Relational DBMS →

where data is kept in table form.

Student

Roll no	Name	f. name	- - -
-	-	-	—
-	-	—	—
		—	—

④ Object oriented DBMS → data is keapt in form of Objects.

⑤ Distributed DBMS ⇒ Databases that are distributed over multiple machines.

5/2/2024

E-R diagram

→ E-R diagram's ~~is~~ main purpose is to draw conceptual model / schemas.

→ Entity - Relationship model

→ Entity → Any real world object can be refer as entity.

Tohe relationship between various entities, ~~caste~~ is represented by entity relationship diagrams.

Entity Type →

Entities are instance of ~~E~~ Entity Type

Set of entities are "Entity Set."

entity → 1

entity → 2

entity → 3

Student → Attributes

Roll no.	Name	fname	dob
1	A	-	-
2	-	-	-
3	-	-	-

& in this example entity type is "Student".

Domain – Value allowed for attribute

example → age ⇒ 18 to 60

↳ domain of age

Types of attributes ⇒

① → Simple (atomic) → which cannot be subdivided

→ eg → city - delhi

② → Composite → Combination of two or more attributes.

eg → address → house no., street, Pin etc.

③ → Single value → which has one value

eg → dob.

④ → Multivalue → which can have multiple values

eg → mob. no.

⑤ Stored

→ Which is stored in ~~databgs~~ database
↳ eg→ dob, name.etc.

⑥ derived →

→ which is derived from some other attribute
↳ eg→ age, salary etc.

⑦ Null value →.

→ unknown

→ Missing

→ Not Known

⑧ Complex value → Combination of ~~all ab~~ values above.

Keys

① → Super key
② → Candidate key
③ → Primary key
④ → foreign key
⑤ → Secondary key

①→ Super key →

Attribute or set of Attribute by a unique record can be retrieved.

eg→

(emp.id, name)

(name, mob)

(name, email)

Employee database

emp.id	name	f.name	dob.	mob	email
-	-	-	-	-	-

② Candidate key →

Minimum super key by which a unique record can be retrieved.

eg→ (emp.id), (mob), (email) } in our employee database above.

{ All candidate keys are super keys but not all super keys are candidate keys }

{ Candidate ~~co~~ key, can also have multiple attributes. }

③ Primary key →

One of the candidate key might be used in Software for retrieving unique record, that candidate key is ~~Kno~~ known as primary key.

{We can have multiple Candidate Keys but Key which is used by system to identify record is primary Key}

eg → emp.id in our employee database

06/02/24

④ **Secondary key** ⇒ one of the ~~pr~~ candidate key is primary key and rest of them are secondary key.

{Candidate key can have Null value but primary key cannot have null value}

⑤ **foreign key** ⇒

Emp

eid	ename	did	mob
1	A	1	
2	B	2	
3	C	1	
4	D	1	
5	E	3	

→ foreignkey

Designation

did	dname
1	TD
2	ACP
3	JD
4	AD

→ our primary key of designation

The key which is refering to primary key of some other table is our foreign key.

* {foreign Key can refer to primary Key of other table or of its own table}

eg →

Emp

eid	ename	manager_id
1	A	—
2	B	—
3	C	1
4	D	2

Emp → {Primary Key → eid}

{foreign key → manager-id}

which refers to eid of Emp

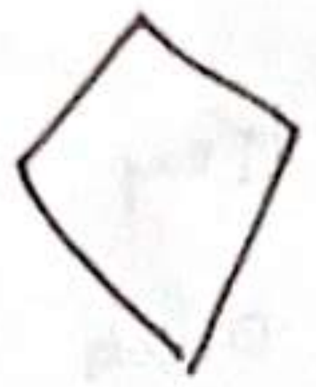
{foreign Key can have null values}

E-R diagram Notation ⇒

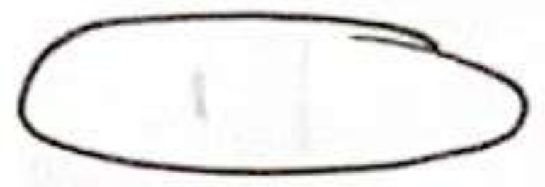
① Rectangle → Entity type

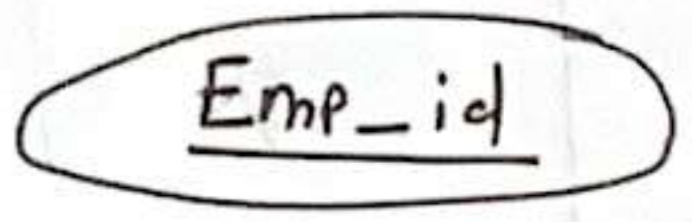
② Double Rectangle → weak Entity type

which donot have a primary key or candidate key.

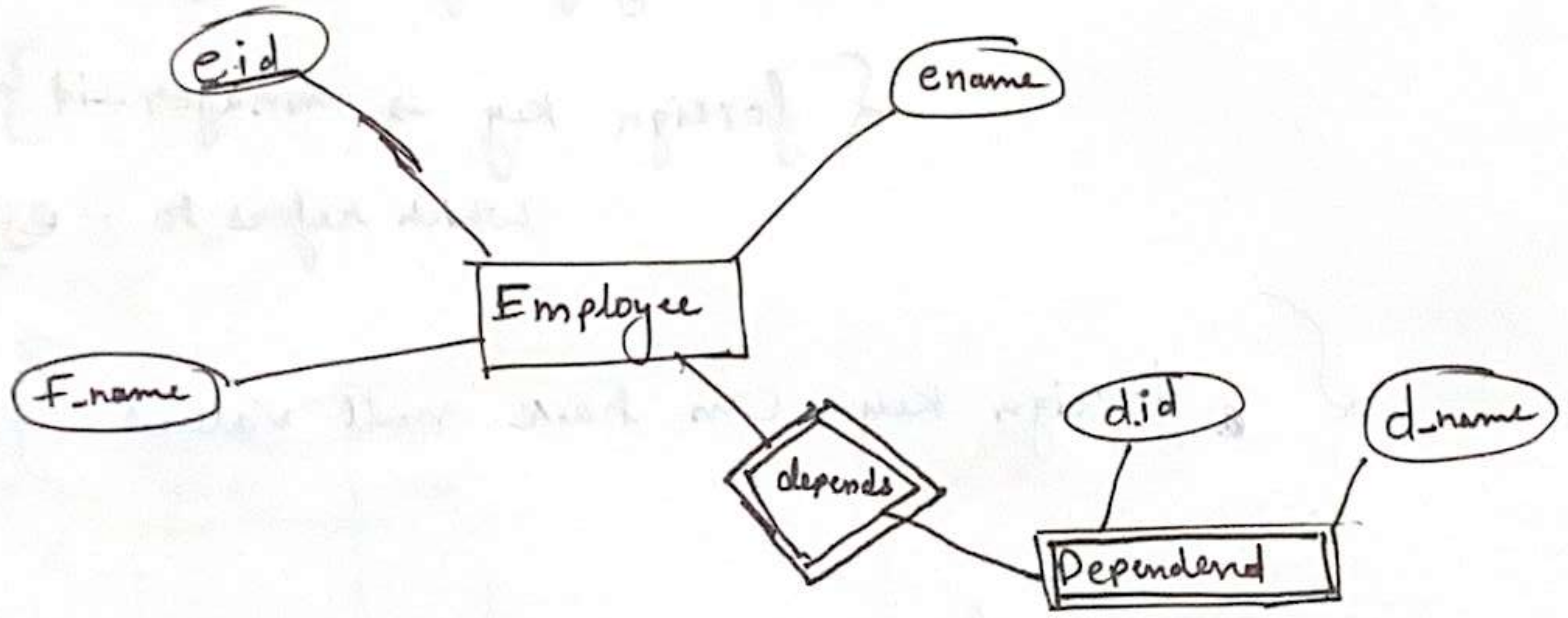
③ Diamond → Relationship 

④ double diamond → Identifying Relationship
{Relationship between Strong entity & weak entity}

⑤ Ellipse ⇒ Attributes 

ⓐ Ellipse with underline attribute → To represent Key attribute 

Eg →



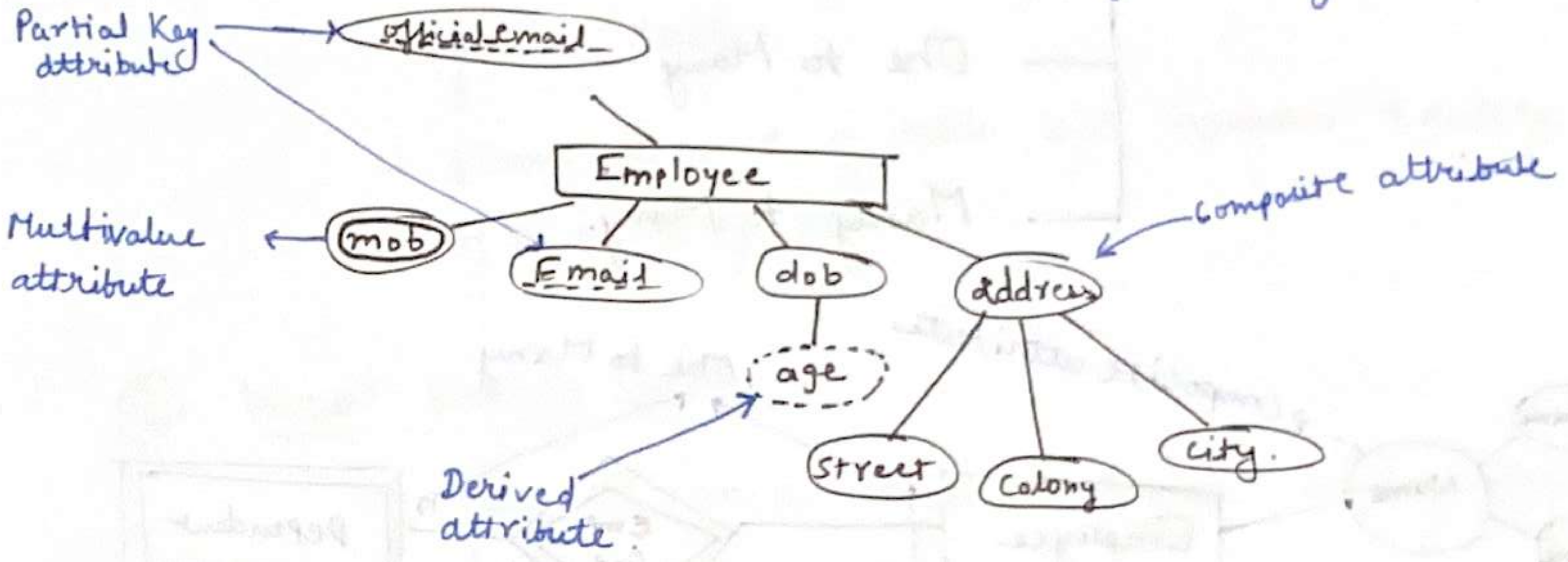
ⓑ Ellipse with dashed underline → Partial key attribute.

ⓒ Double ellipse → Multi value attribute.

ⓓ Ellipse attached with ellipse → Composite attribute.

ⓔ Dashed ellipse → Derived attributes.

{Partial key attribute ⇒ we have use more than 1 attribute to form a key}



8/2/24

⑥ Lines → Link attributes / entities & relations.

Double Lines → Total Participation

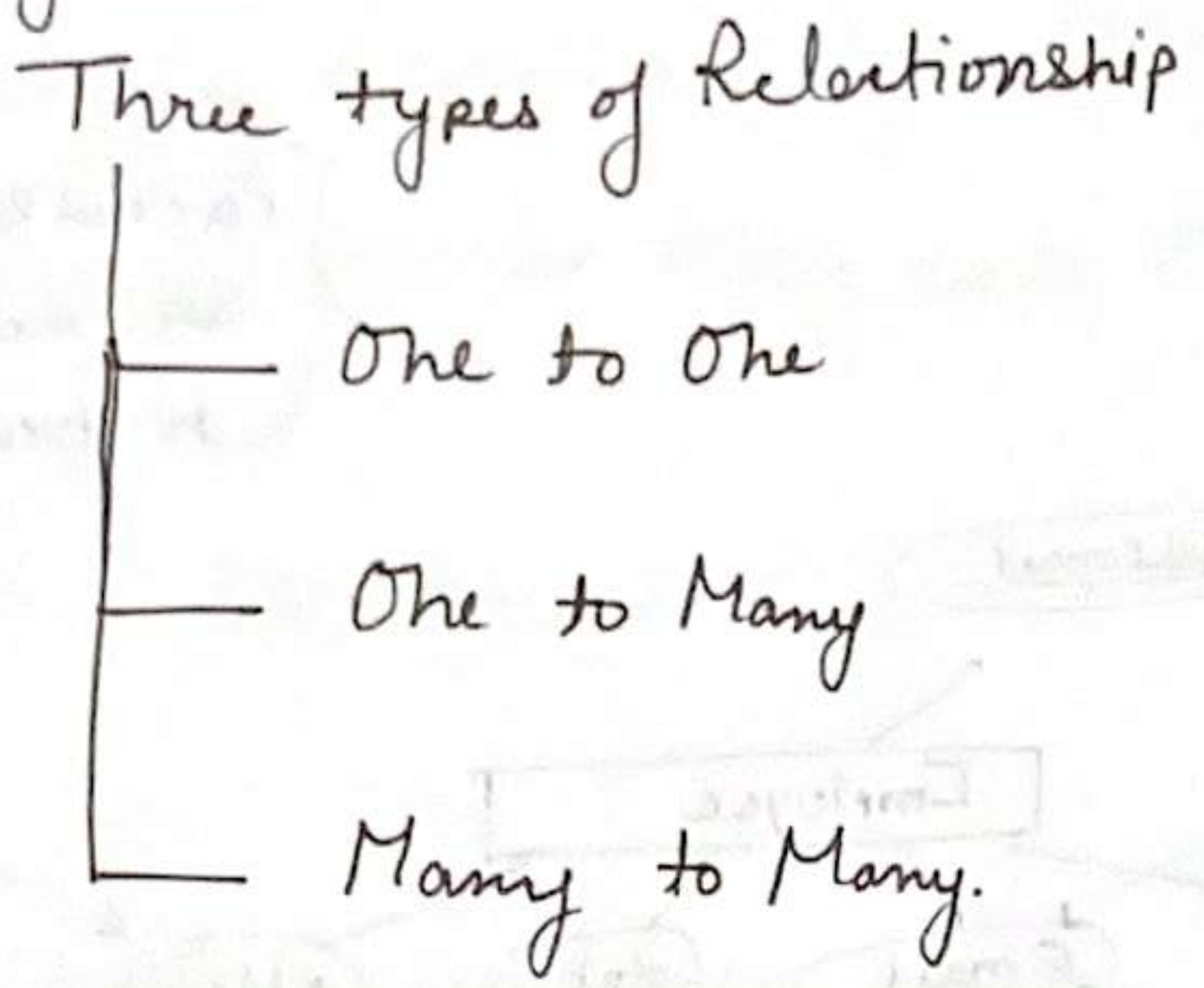
Partial
Total.
↓
Min. 1 Entity is participating in Relation.

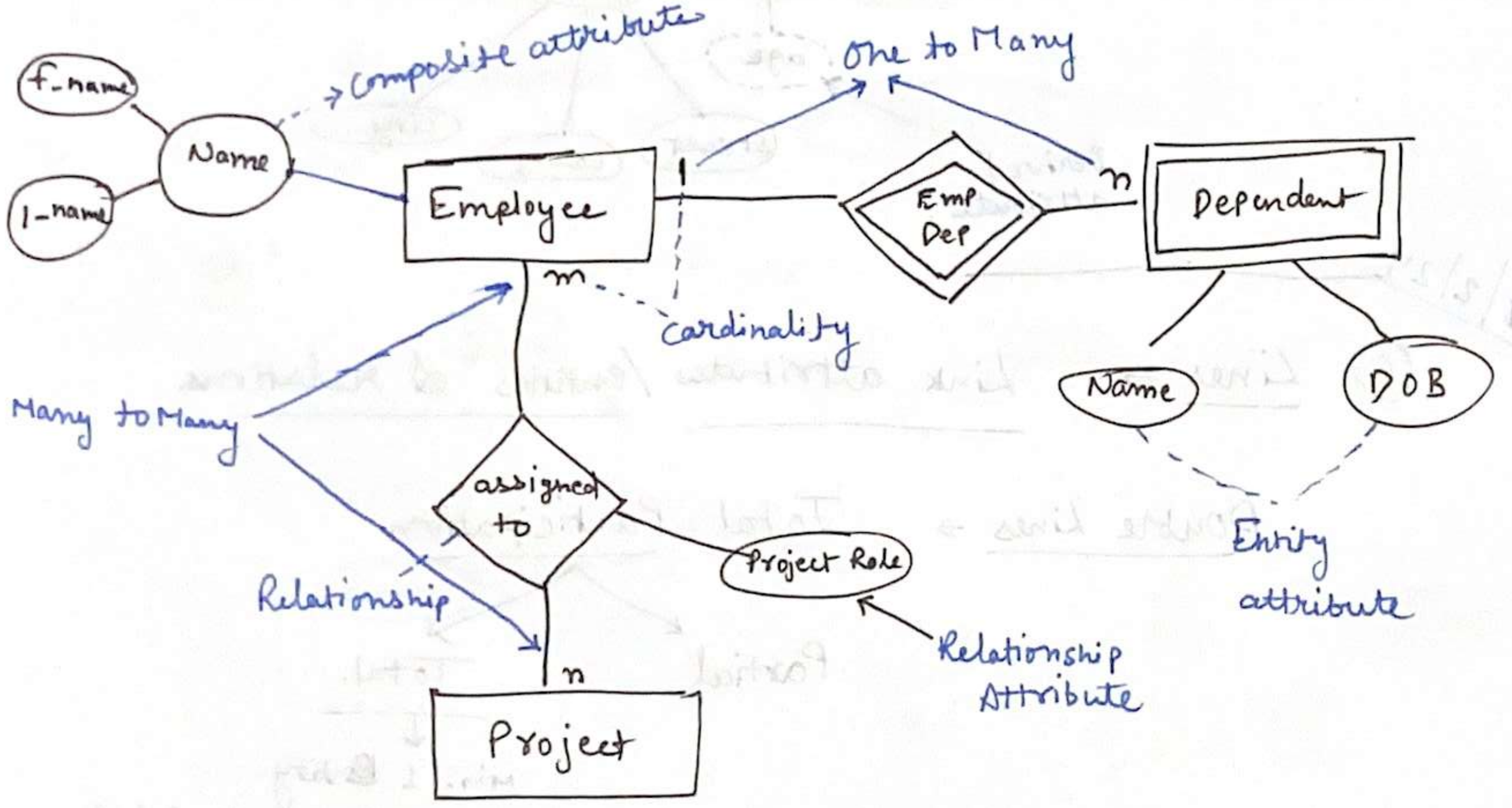
Relationship Constraints →
There are two constraints which we call Relationship Constraints :->

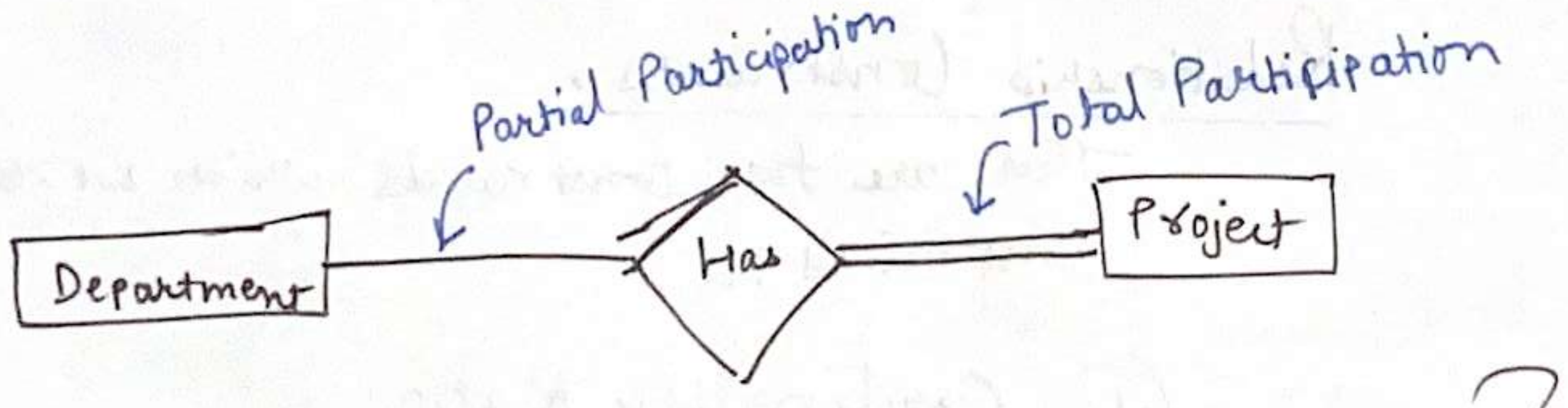
① Cardinality Ratio

② Participation

* Cardinality Ratio







100% Projects will have some departments,
Not all department will have projects

Transformation of ER diagrams to Table =>

~~Rule~~ ① → Strong Entity =>

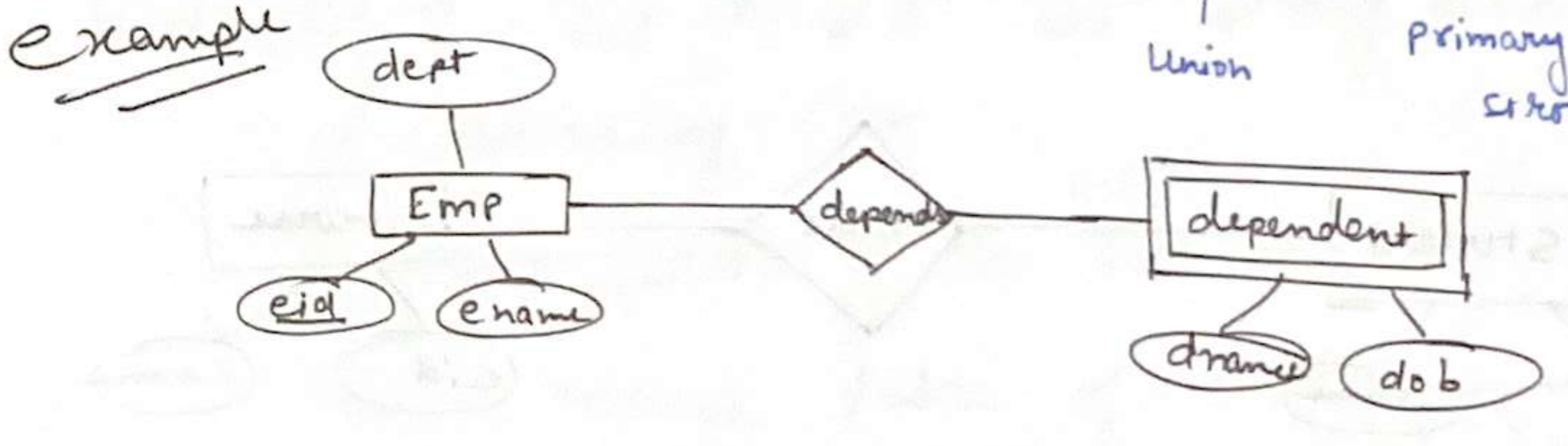
- There will be a table for each strong entity.
- All attributes of strong entity will be Columns of that table.
- Every record of table will represent a entity of that entity set.

② Weak entity set →

- Let A be a weak entity set having attributes (a1, a2, a3 .... a_n)
- Suppose B is a strong entity on which A depends. ~~B~~ Let's say B have primary key $\{B_1, B_2 \ldots B_n\}$
- There will be a table for weak entity ~~Aset~~ A with attribute.

$$\{a_1, a_2, a_3 \ldots a_n\} \underset{\uparrow\ \text{Union}}{\cup} \underbrace{\{b_1, b_2 \ldots b_n\}}_{\text{Primary key of strong entity.}}$$

example



Emp ← Strong entity

eid	ename	dept

dependent ← weak entity

dname	dob	eid

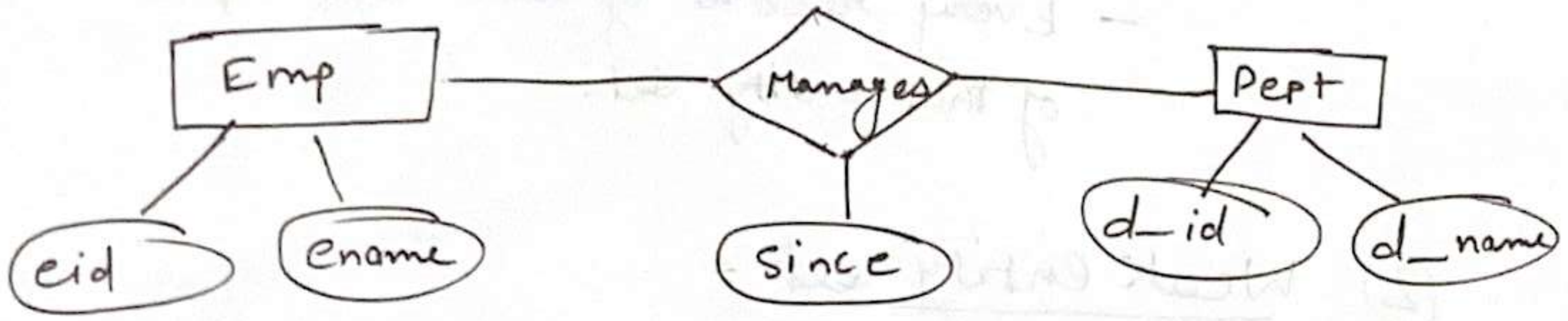
③ Relationship →

Each relationship will have a table

with → attributes of relationship

→ Primary key of both tables.

ex →



Till now what we have read there will be 3 tables for above E-R.

EMP

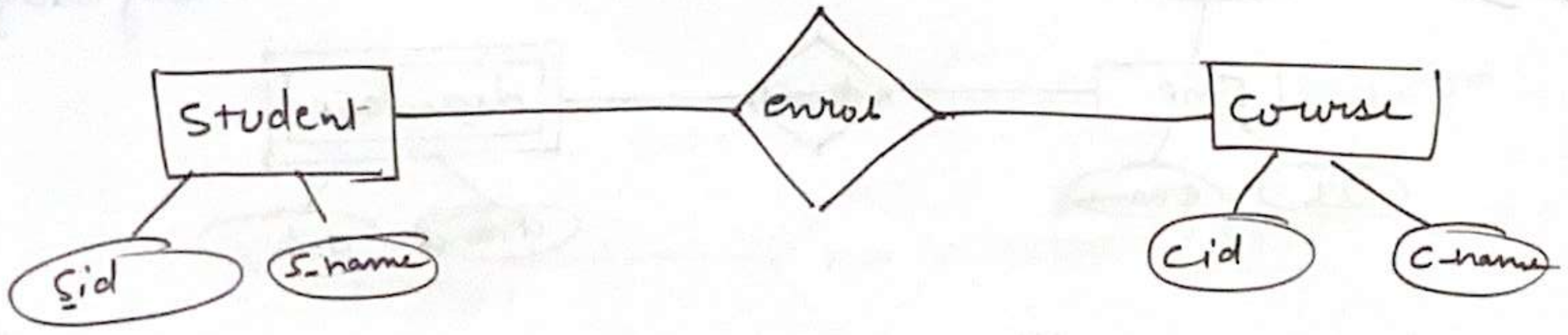
eid	ename

Manages

eid	did	since

Dept

d_id	d_name



Student

S_id	S_name

Enroll

Sid	C_id

Course

C_id	C_name

Rule 3.① Let A be a strong entity set.

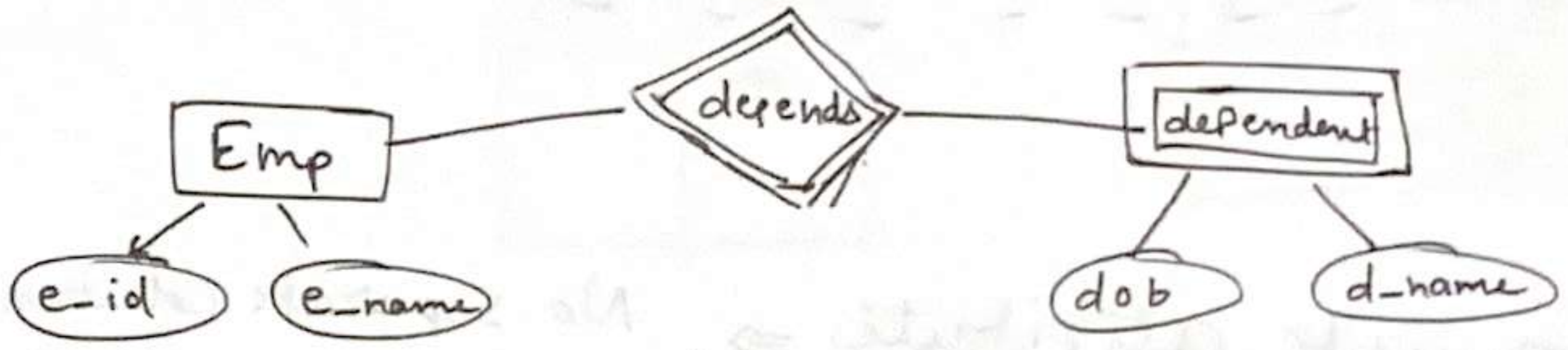
Let B be a weak entity set which is dependent on A.

Suppose relation b/w A & B is AB

Tables ⇒ ① Strong entity set

② B U AB will be a table

Here if AB table is created then it will be Redundant.



Emp

e_id	e_name

depends

~~e-id~~	~~dname~~

dependent

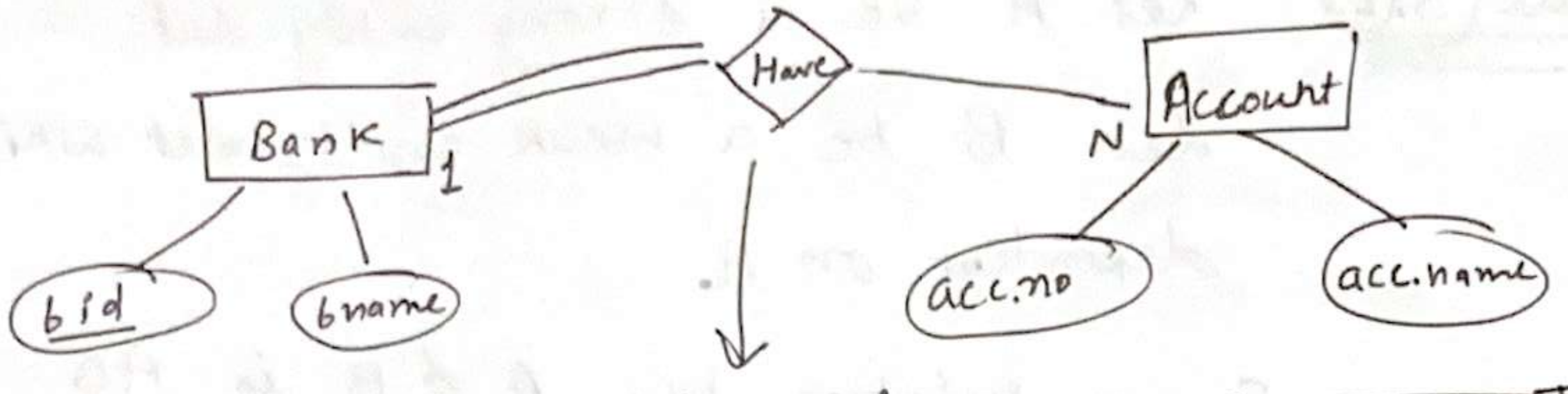
dob	d-name	eid

Rule 3.2 ⇒ Let A & B are strong entity set.

& Cardinality 1 : n

& Participation of one side is total (1-side)

Then merge table of relation & of table of N side.



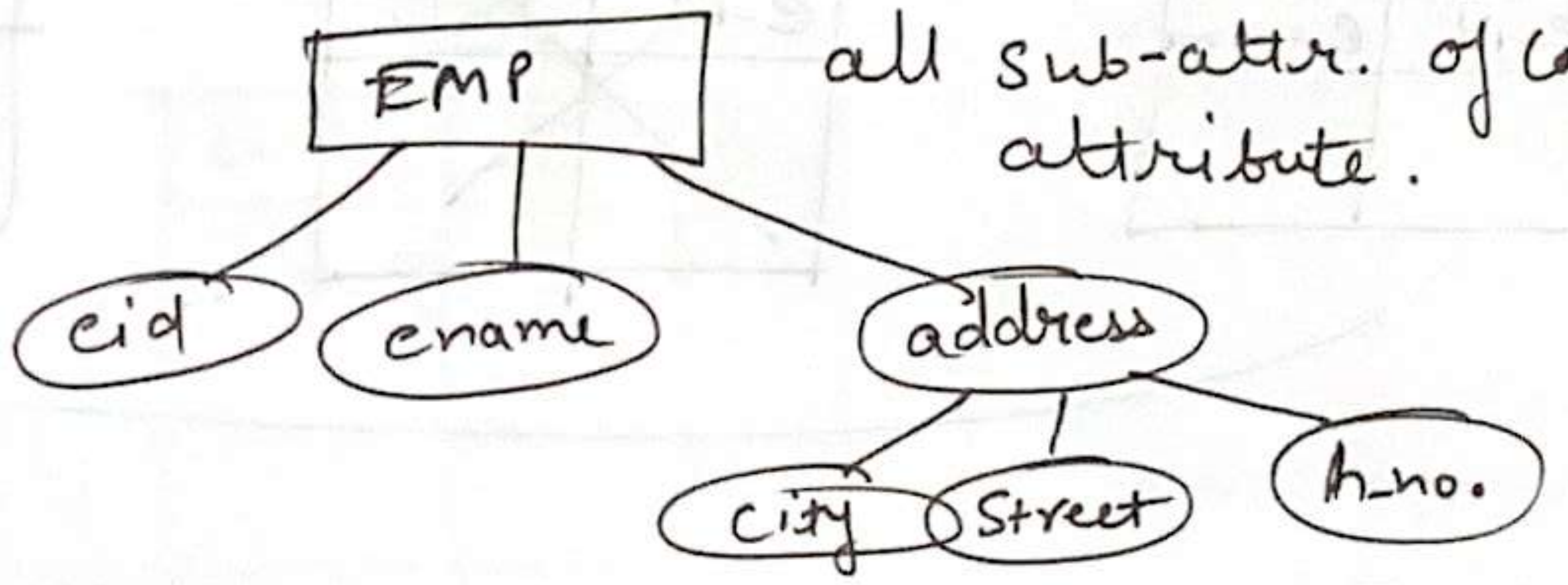
b.id	bname

~~bid~~	~~acc.no~~

acc.no.	acc.name	b.id

④ Composite attribute ⇒ No separate column for composite attribute but there will be columns for all sub-attr. of composite attribute.

example ⇒

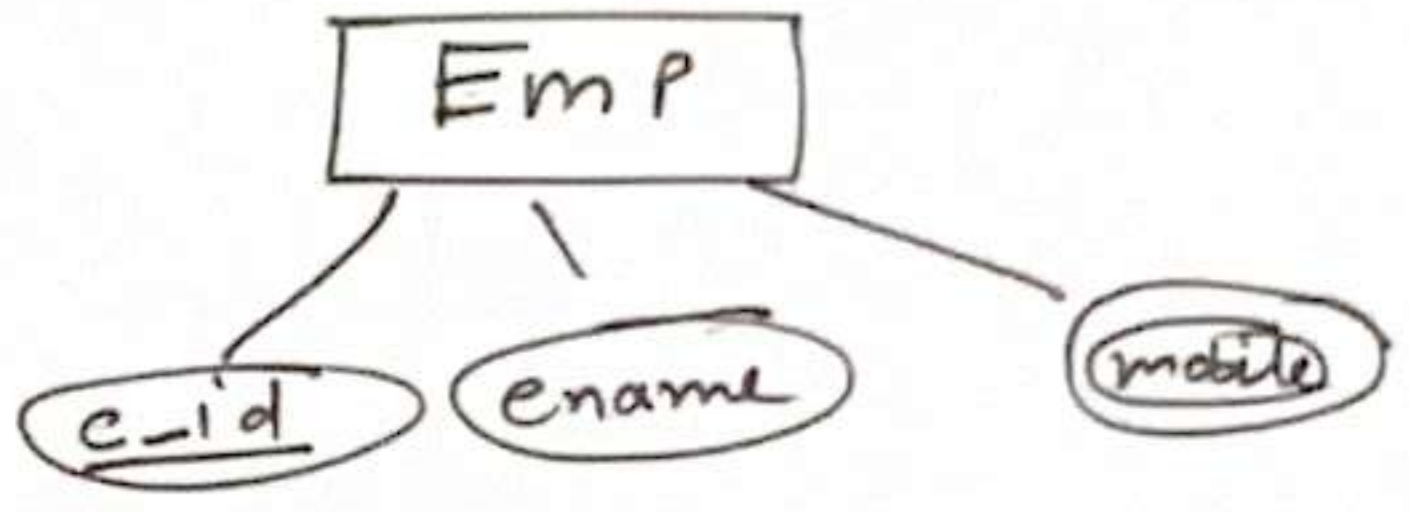


Emp

eid	ename	h_no.	street	city

⑤ Multivalued attributes ⇒

There will be a separate tables for each multivalue attribute and that table will keep primary key of main table.



Emp

e_id	ename	moble
1	A	123, 457, 145

Emp

eid	cname
1	A

mobile

eid	mobile
1	123
1	457
1	145