Database Management System

Database :- • A database system is basically a computer based record keeping system.

Database → Collection of data
→ collection of interrelated data stored together to serve multiple applications

- Intention of database is that the same Collection of data should serve as many application as possible.

Limitation of File-Processing system :-

i) Data duplication (Data Redundancy)
ii) Data Inconsistency
iii) Unsharable data
iv) unstandarized data
v) Insecure data
vi) Incorrect data

DBMS :-
- store data
- visualize data
- Access (query) data
- update data

Advantages of DBMS :-

i) Redundancy and Inconsistency

- Redundancy reduces data duplication.
- Data inconsistency is a condition that occurs between files when similar data is kept in different formats in two different files, or when matching of data must be done between Files. As a result of data inconsistency, these files duplicate some data such as addresses & names compromising data integrity.

ii) Data Isolation
It defines how/when the changes made by one operation become visible to other.

iii) Data Integrity
This can be indicated by the absence of alteration between two updates of data record, meaning data is unchanged.

iv) Atomicity of operations

An atomic transaction is an indivisible & irreducible series of database operations such that either all occur, or nothing occurs.

v) Concurrency

It is the ability of a database to allow multiple users to affect multiple transactions. ex- spreadsheets

vi) Security

Disadvantage of DBMS

i) Cost of Hardware & software of a DBMS is quite high which increases the budget of your organizat^n.

ii) Most DBMS are often complex systems, so the training for users to use the DBMS is required.

iii) Use of same program at a time by many users sometimes lead to the loss of some data.

iv) DBMS can't perform sophisticated calculations.

Application of DBMS

i) Banking — For customer information, account activities, Payments, deposits, loans etc.

ii) Airlines — For reservations and schedule informat^n.

iii) Universities — For student informat^n, course registrat^n, colleges & grades.

iv) Telecommunication — It helps to keep call records, monthly bills, maintaining balances etc.

v) Finance

vi) Sales

vii) Manufacturing

viii) HR Management

Database System Vs File System

DBMS	File System
Multi - user access	It does not support multi - user access.
Design to fulfill the need for small & large business.	It is only limited to smaller DBMS system.
Remove redundancy & Integrity.	Redundancy & Integrity issues.
Expensive. But in the long term total cost of ownership is cheap.	It is cheaper.
Easy to implement complicated transactions.	No support for complicated transactions.

Popular DBMS Software

- MySQL
- Microsoft Access
- Oracle
- PostgreSQL
- dBASE
- FoxPro
- SQLite
- IBM DB2
- Microsoft SQL Server

Types of DBMS

i) Hierarchical DBMS :- In this DB, model data is organized in a tree-like structure. Data is stored hierarchically (top down or bottom up) format.

ii) Network Model :- This model allows each child to have multiple parents. It helps you to address the need to model more complex relationship like as the orders / parts many to many relationship.

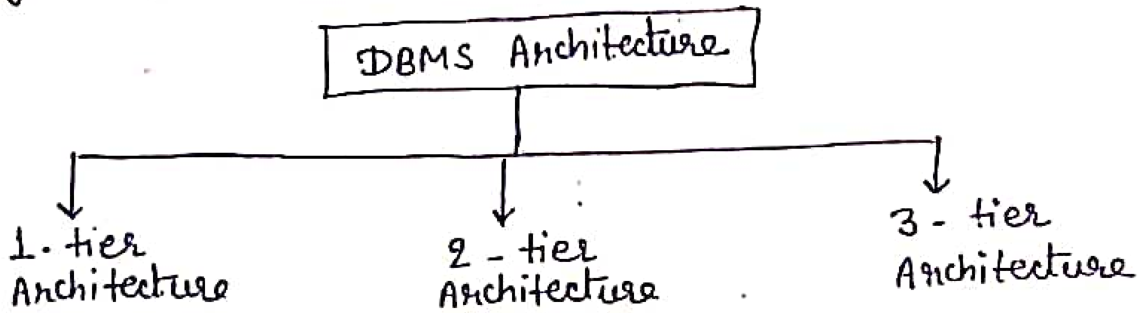
iii) Relational Model :- It is the most widely used DBMS model because it is one of the easiest. This model is based on normalizing data in the rows & columns of the tables.

iv) Object-oriented Model :- In this model, data stored in the form of objects. The structure which is called classes which display data within it. It defines a database as a collection of objects which stores both data members values & operations.

DBMS Architecture

- DBMS architecture helps in design, development, implementation and maintenance of a database. A database stores critical information for a business. Selecting the correct database architecture helps in quick + secure access to their data.
- DBMS architecture depends upon how users are connected to the database to get their request done.

- Types of DBMS Architecture



1 - tier Architecture

- In their architecture, the database is directly available to the user. It means the user can directly sit on the DBMS + uses it.
- Any changes done here will directly be done on the database itself. It does not provide a handy tool for end users.
- It is used for development of the local application, where programmers can directly communicate with the database for the quick response.

2 - tier Architecture

- This is same as basic client - server. In their architecture, applications on the client end can directly communicate with the database at the server side. For their interaction, API's like ODBC, JDBC are used.

- The user interfaces + application programs are run on the client side.
- The server side is responsible to provide the functionalities like query processing + transaction management.

- To communicate with the DBMS, client side application establishes a connection with the server side.

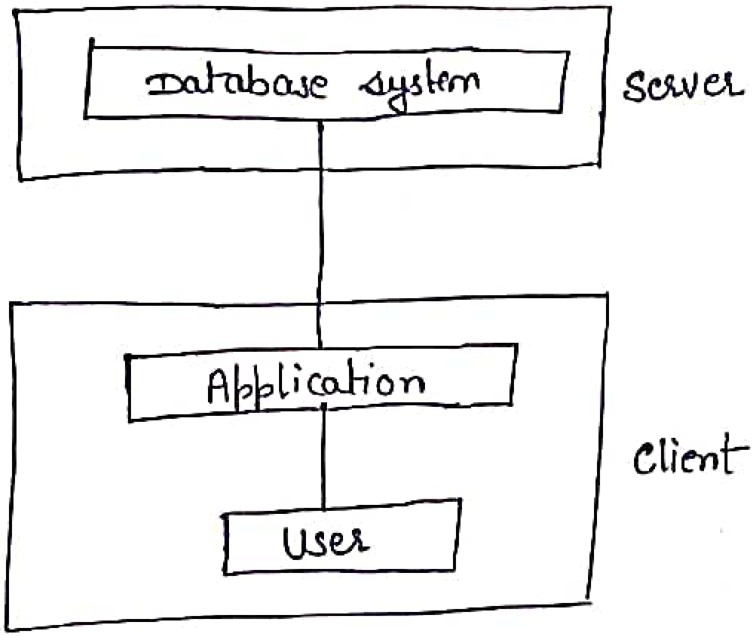


Fig - 2-tier architecture

3-tier Architecture

- It contains another layer between the client + server. In this architecture, client can't directly communicate with the server.
- The application on the client end interacts with an application server which further communicates with the database system.
- End user has no idea about the existance of the database beyond the application server. The DB has no idea about any other user beyond the application.
- It is used in case of large web application.

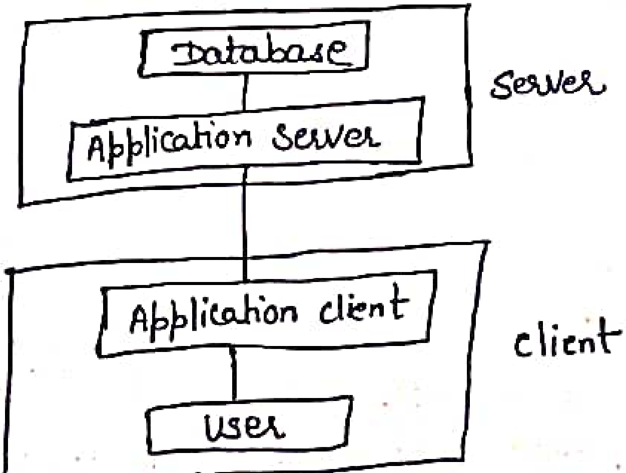


Fig - 3-tier Architecture

Data model Schema and Instance

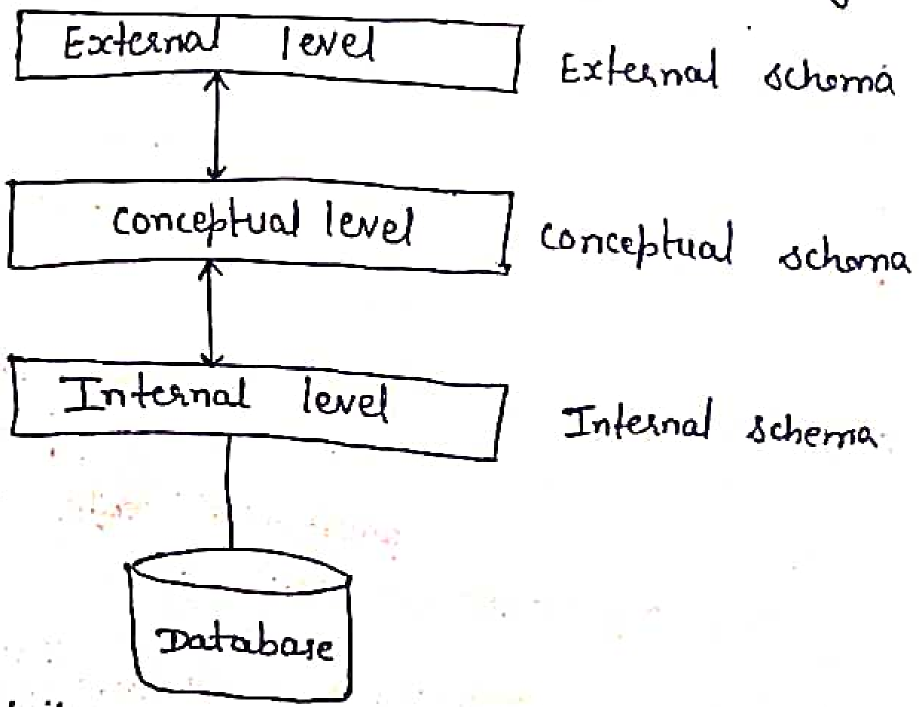
Schema :- The overall design of a database is called schema.

- A database schema is the skeleton structure of the database. It represents the logical view of the entire database.
- It contains schema objects like table, Foreign key, primary key, views, columns, data types, stored procedure, etc.
- It is designed by the database designers to help the programmers whose software will interact with the DB. The process of DB creation is called data modeling.

Instance :- The data which is stored in the database at a particular moment of time is called an instance of the database.

imp Three Schema Architecture :-

- It is also called ANSI / SPARC architecture or three-level architecture.
- This arch. is used to seperate the user applications and physical database.
- It breaks the DB down into three different categories.



Internal schema

- It is lowest level of data abstraction
- It defines the physical storage structure of the DB
- It helps you to keeps information about the actual representation of the entire database.
- The internal view tells us what data is stored in the database and how.
- It never deals with the physical devices, Instead, internal schema views a physical device as a collection of physical pages.

Conceptual schema

- It describes the database structure of the whole database for the community of users.
- This schema hides information about the physical storage structures & focuses on describing data types, entities, relationships, etc.
- Security & integrity information.

External schema

- It describes the part of the database which specific user is interested in.
- It is nearest to the user.
- It is only related to the data which is viewed by specific end users.
- An external view is just the content of the DB as it seen by some specific particular user.
  For ex- a user from the sales department will see only sales related data.

Data Independence

- It refers characteristics of being able to modify the schema at one level of the database system without altering the schema at the next higher level.
- There are two types of data independence.

i) Logical data Independence

- It refers characteristic of being able to change the conceptual schema without having to change the external schema.
- It is used to seperate the external level from the conceptual view.
- It occurs at the user interface level.

ii) Physical Data Independence

- It can be defined as the capacity to change the internal schema without having to change the conceptual schema.
- It is used to seperate conceptual levels from the internal levels.
- It occurs at the logical interface level.

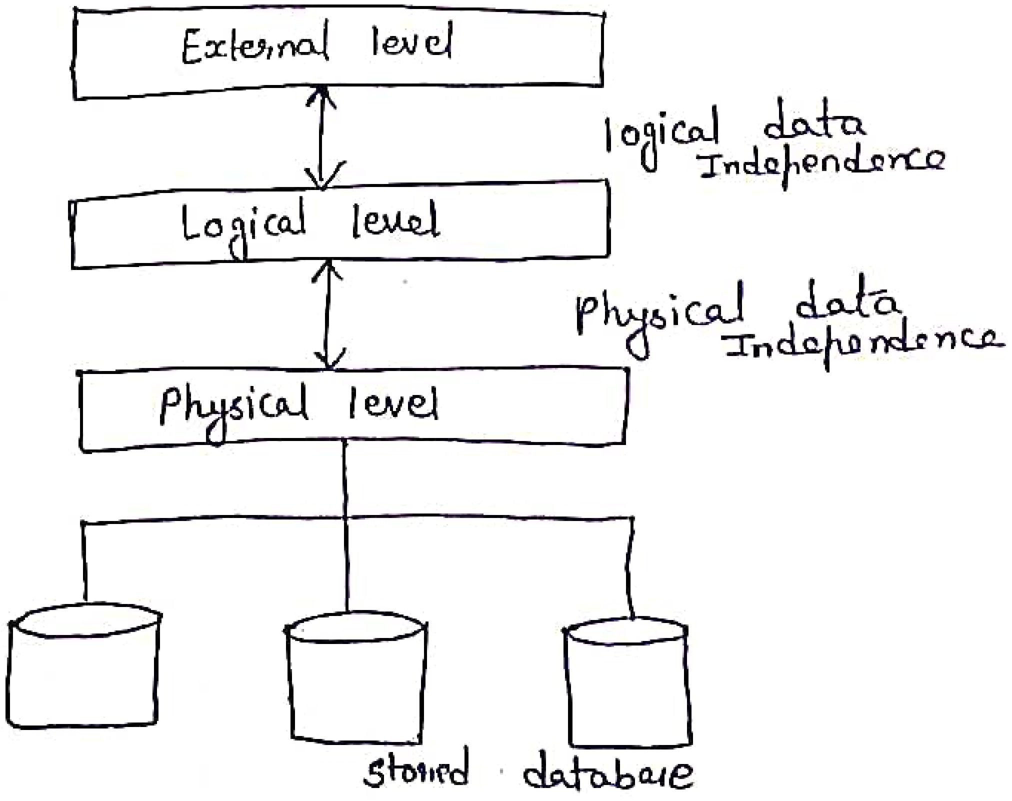


Fig- Data Independence

Database Language

- Database languages can be used to read, store and update the data in the database.

- Types of Database language :-
  i) DDL (data definition language)
  ii) DCL (data control language)
  iii) DML (data Manipulation language)
  iv) TCL (Transaction control language)

DDL :- • It is used to define database structure or pattern.

- It is used to store the information of metadata like the no. of tables and schemas, their names, etc.
- Some tasks that come under DDL -

i) Create - It is used to create objects in the DB.
ii) Alter - It is used to alter the structure of DB.
iii) Drop - It is used to delete objects from the DB.
iv) Truncate - It is used to remove all records from a table.
v) Rename - It is used to rename the object.
vi) Comment - It is used to Comment on the data dictionary.

DML :- • It is used for accessing & manipulating data in DB.
• It handles user request.

- Some tasks :-

i) Select - It is used to retrieve data from a DB.
ii) Insert
iii) Update
iv) Delete - It is used to delete all records from a table
v) Merge
vi) Call
vii) lock table - It controls concurrency.

DCL :- • It is used to retrieve the stored or saved data.

- The DCL execution is transactional. It also has rollback parameters.
- Some tasks –

i) Grant – It is used to give user access privileges to a DB.

ii) Revoke – It is used to take back permissions from the user.

There are the following operations which have the authorization of Revoke :

Connect, Insert, Usage, Execute, Delete, Update & Select.

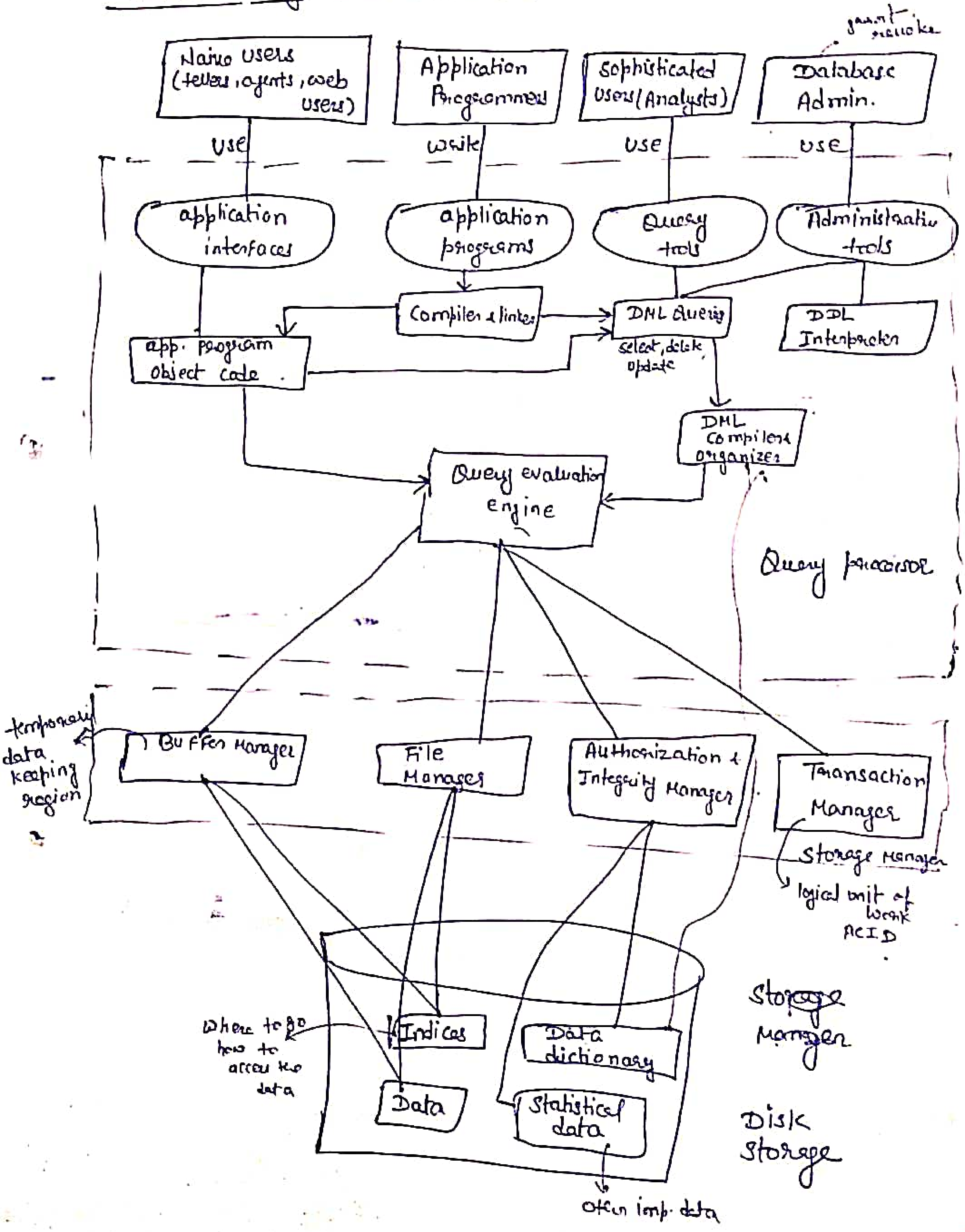
TCL :- • It is used to run the changes made by the DML statement.

- Some tasks –

i) Commit – It is used to save the transaction on the database.

ii) Rollback – It is used to restore the database to original since the last commit.

Database System Architecture



Database Schemas & Instances

schema — It is the overall description of the database.

- 3-level arch. one schema at each level
- Does not specify relationship among Files.

Instance - collection of info. stored in the database at a particular moment is called as an instance.

ex- emp 1/7/1980 {4 records} 1 instance

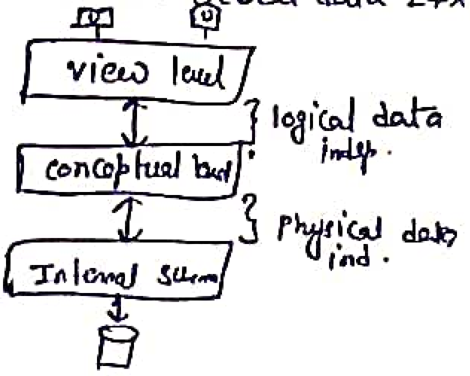
emp 4/7/1984 {400 records}

subschema - It is an application programmer or users view of the data item types & record types which he or she uses.

EMP → Prog. [all columns access]
→ User [2 columns]

Data Independence

- data independent from user
- We can access data 24x7



physical data Ind.

if we change anything in Physical schema, then conceptual level will not change.

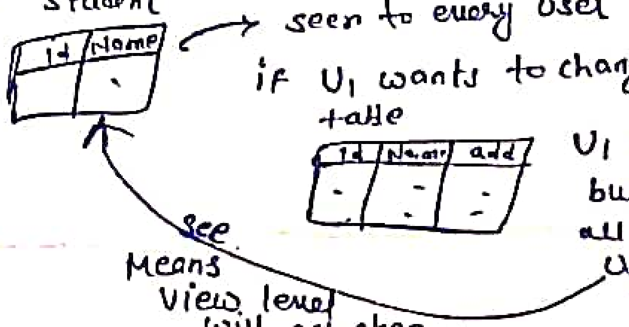
Suppose DB1 is transfered to HD2, then it does not means table name will be change.

Storage str.
Data str.
Index
} will change but conceptual level will not change.

ex- google

logical data Ind.

- implemented by views (virtual table)
- Student



seen to every user

if U1 wants to change the table

U1 see but all others user see

Means view level will not change.

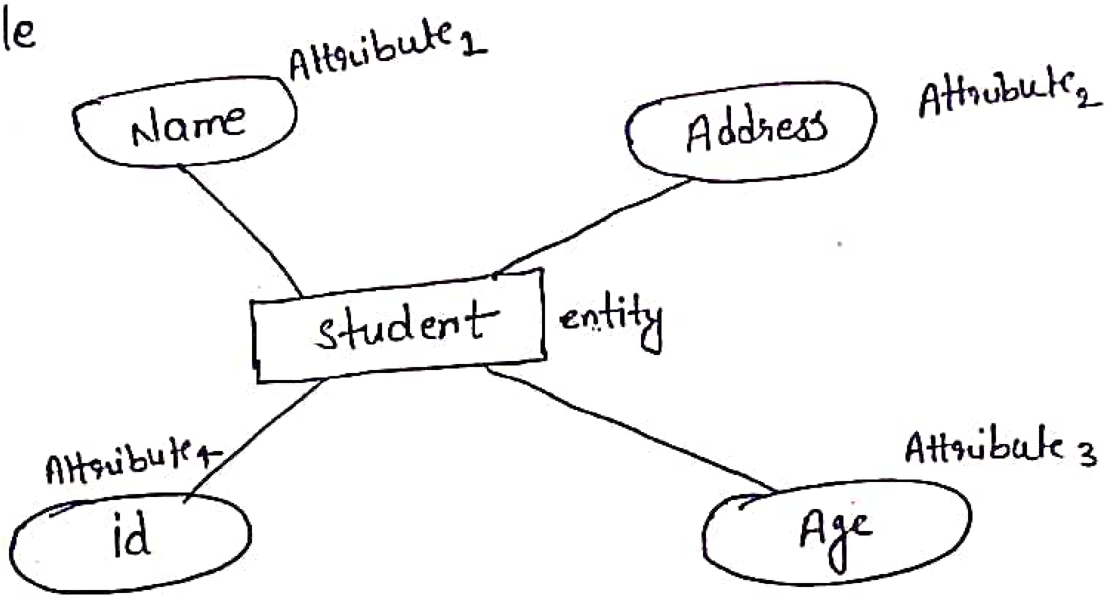
- web app. remains same. whether any user changes, delete, insert the table,

Data ind. provides transparency.

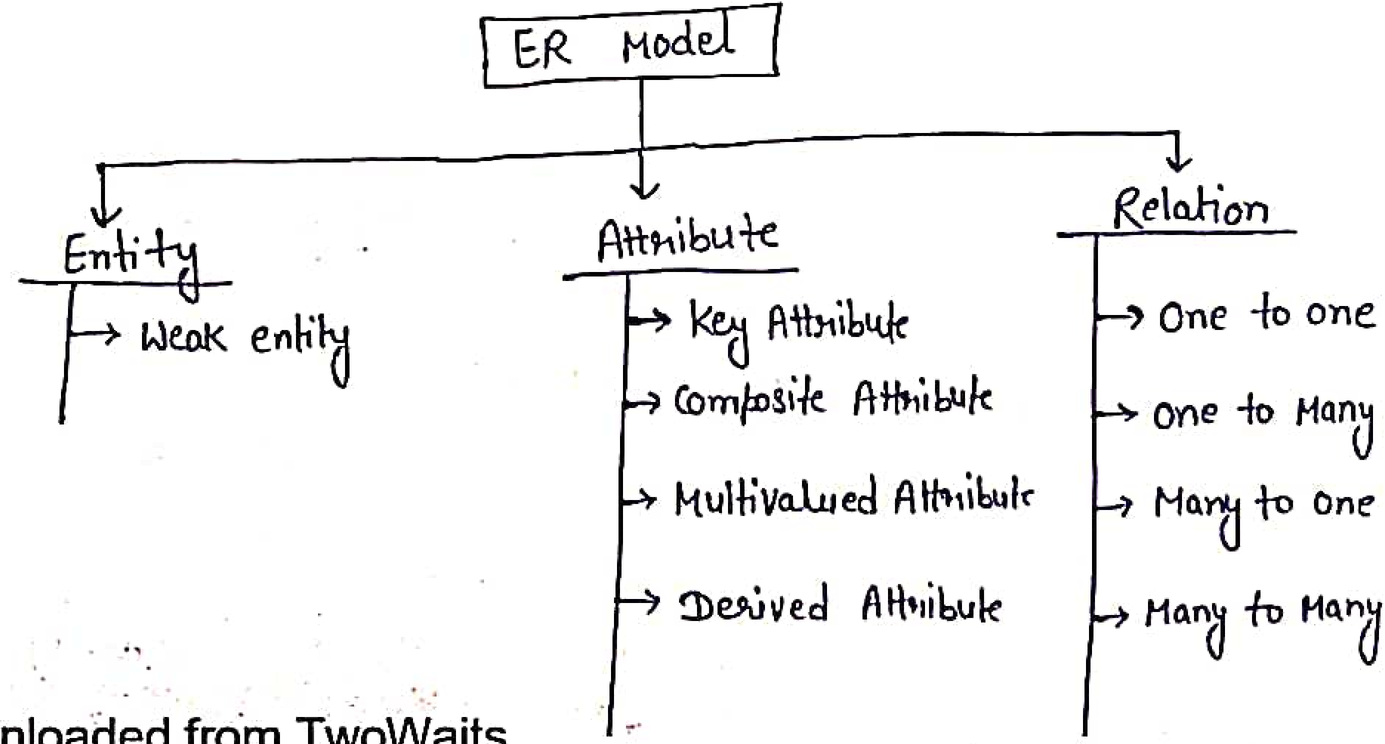
ex- gmail.

ER Model

- It stands for an Entity-Relationship model.
- It is high level data model. This model is used to define the data elements & relationship for a specified System.
- It develops a conceptual design for the database.
- In ER modeling, the database structure is portrayed as a diagram called an Entity-relationship diagram.
- Example



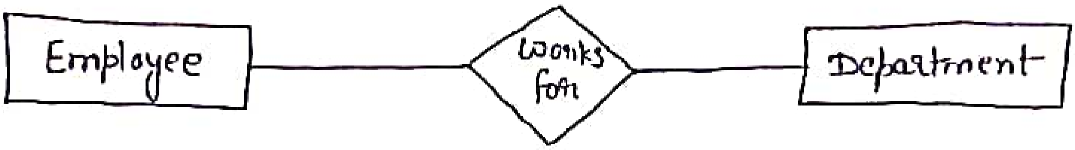
Component of ER Diagram



Entity :- • An entity may be any object, class, person or place.

• In ER diagram, an entity can be represented as (▭) rectangles.

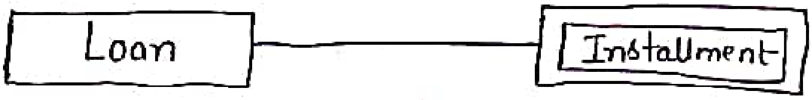
ex-



Weak Entity :- • An entity that depends on another entity Called a weak entity. The weak entity does not contain any key attribute of its own.

• It is represented by a double rectangle (⧈).

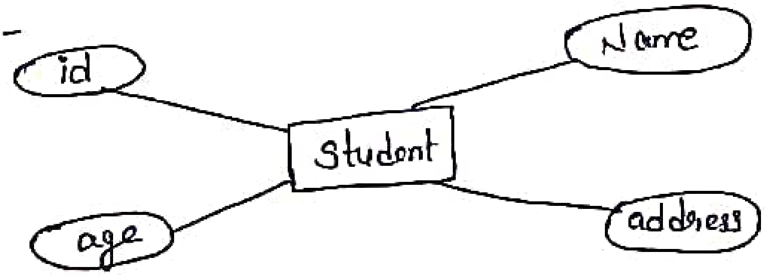
ex-



Attribute :- • It is used to describe the property of an entity.

• It is represented by an Eclipse (⬭).

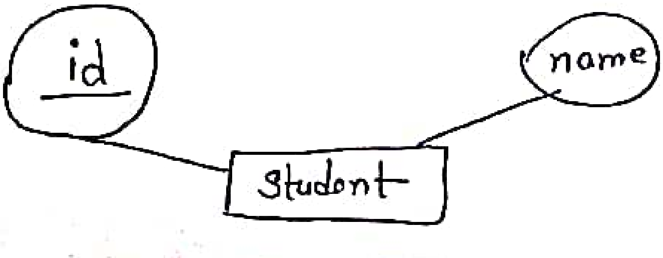
ex-



Key Attribute :- • It is used to represent the main characteristics of an entity.

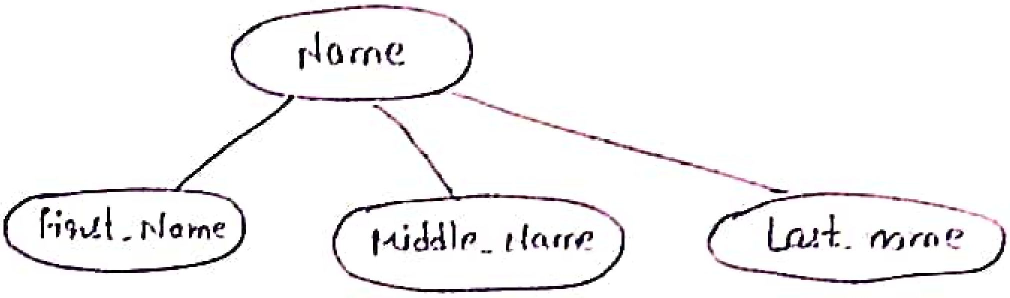
• It represents a primary key.

• It is represented by an ellipse with teo text underlined.



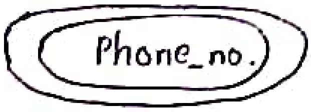
Composite Attribute :— • An attribute that composed of many other attributes is known as composite attribute.

• It is represented by an ellipse, & those ellipses are connected with an ellipse



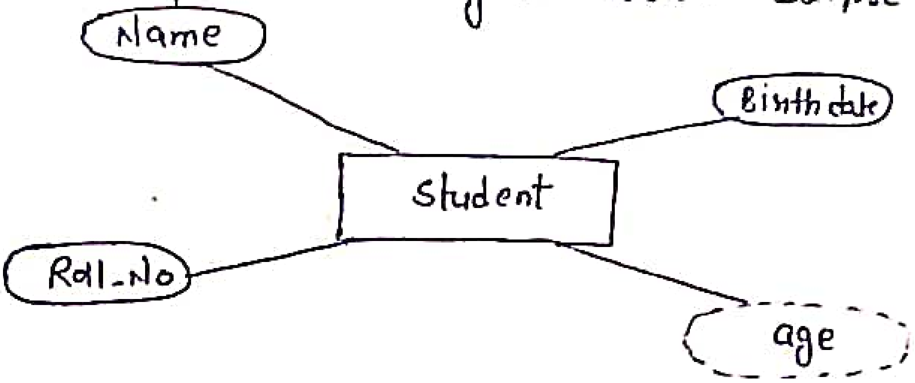
Multivalued Attribute :— • An attribute can have more than one value.

• It is represented by double oval.



Derived Attribute :— • An attribute that can be derived from other attribute.

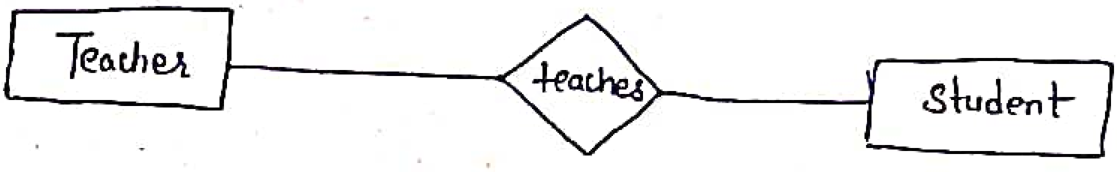
• It is represented by a dashed ellipse.



Relationship :—

• It is used to describe the relation between entities.

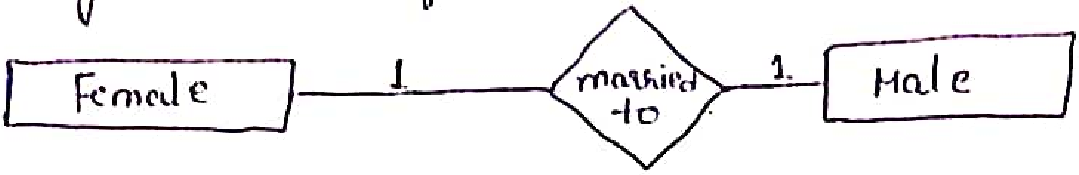
• Diamond or rhombus is used to represent the relationship.



Types of Relationship

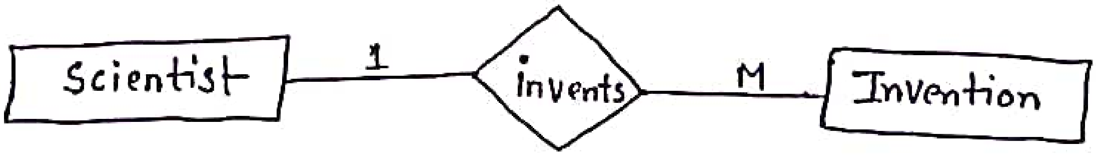
a) **one-to one** :- • when only one instance of an entity is associated with the relationship.

- Ex- A female can marry to one male & a male can marry to one female.



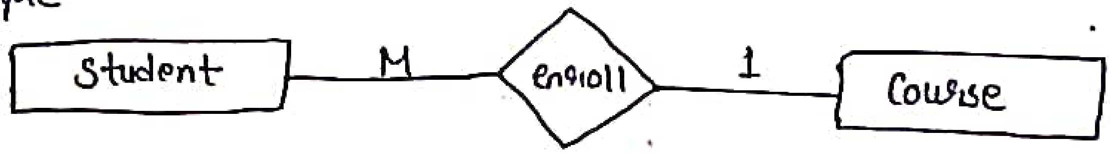
b) **One-to-many** :- • when only one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship.

- Ex-



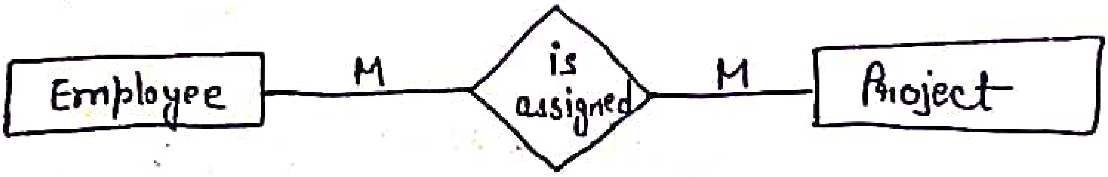
c) **Many-to-one** :- • when more than one instance of the entity on the left, & only one instance of an entity on the right associates with the relationship.

- Example



d) **Many-to-Many** :- • when more than one instance of the entity on the left & more than one instance of an entity on the right associates with the relationship.

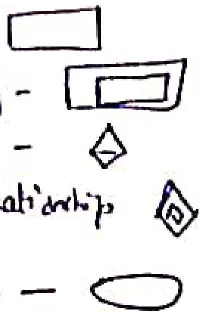
- Example



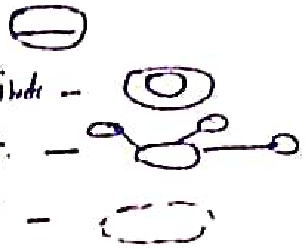
ER Model → Represents database with ER diagram.

Notations used –

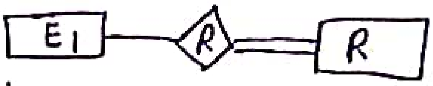
Entity –
Weak entity –
Relationship –
Identifying relationship
Attribute –



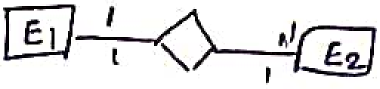
Primary key –
Multivalued attribute –
Composite att. –
Derived att. –



total participation of E_2 in R



Cardinality



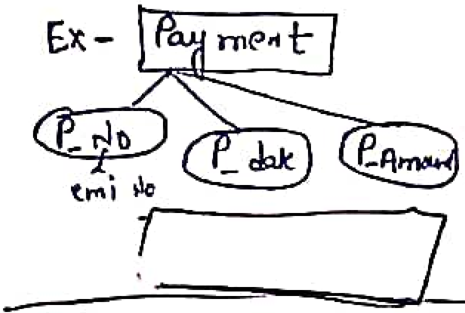
imp

Strong & weak entity sets

Strong: is an entity set that has primary key.

Weak: ↳ entity set which does not have sufficient attributes to form primary key

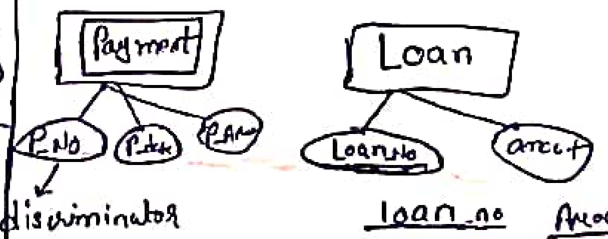
Ex –



Payment

	P_No	P_date	P_Am
u_1	P_1	d_1	10
	P_2	d_2	20
u_2	P_1	d_1	30
	P_2	d_3	40

* How to form primary key of a weak Entity set?
→ Primary key of strong entity set on which weak entity is dependent plus the weak entity set discriminator.



loan_no	Amount
L_1	30

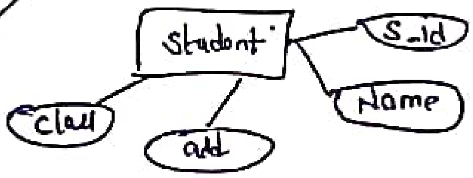
Primary key – { Loan_No, Payment_No }

Entity – An entity is a thing or object in the real world that is distinguishable from all other objects.
ex – student of college, Employee in company

Entity set – It is set of entities of the same type that share their same properties or attributes.

Bank – Employee — {e_id}
Bank – Customer – {AccountNo}

Attribute – An entity is represented by a set of attributes.



For each attribute, there is a set of permitted values called the domain.

S_id → (0 – 1000) ↳ abcd ✗
name → [a–z] [A–Z].

Types of attribute

1. simple / composite

simple ↙ are not divided into sub parts.
ex – Unique id.
1234 ✓
12 ✗ 34 ✗

composite ↳ can be divided into sub parts.
Name → F_N, L_N
address → city, Pin, Street No

2. Single ~~valued~~ / Multivalued

Single ↙ has single value at a instance.
eg – order. id, age, DOB, RollNo
ex – 1234,

Multivalued ↳ can have multiple values.
ex – phone No, degree

3. Derived att. / store att.

Derived att.: that is calculated from stored att. ex – Age

store att. ↳ value that is directly stored.
ex – DOB

Notation of ER Diagram

In ER diagram, many notations are used to express the cardinality.

i) one to one

ii) one to Many (mandatory)

iii) Many

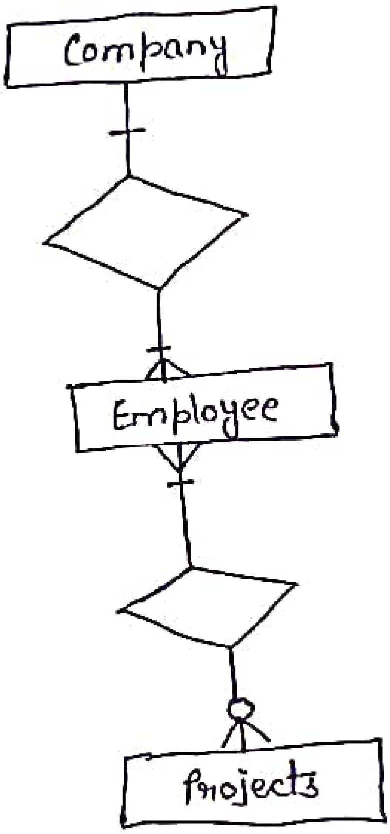
iv) one or more (mandatory)

v) one and only one (mandatory)

vi) zero or one (optional)

vii) zero or many (optional)

Example-



Mapping Constraints

- It is data constraint that expresses the no. of entities to which another entity can be related via a relationship set.
- It is most useful in describing the relationship sets that involve more than two entity sets.
- For binary relationship set R on an entity set A & B, there are four mapping cardinalities.

i) one to one (1:1)
ii) One to many (1:M)
iii) Many to one (M:1)
iv) Many to Many (M:M)

Relationship - a relationship is an association among several entities.

Father — Son

Relationship set - It is a set of relationships of the same type.

Customer	loan	
c1	L1	} Borrow
c2	L2	
cn	Ln	

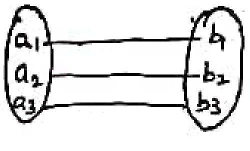
Connectivity/cardinality -

It describes the mapping of associated entity instances in the relationship. OR

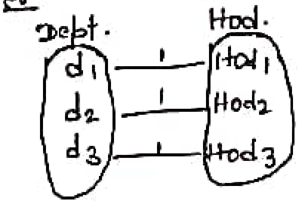
It express the no of entities to which an entity can be associated via a relationship set.

Mapping Cardinality

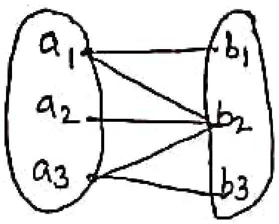
1) one to one



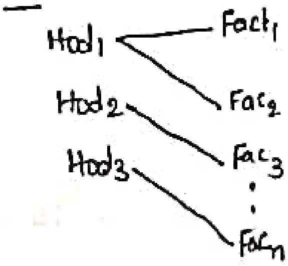
ex-



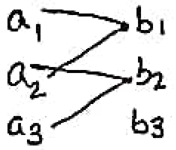
2) one to many

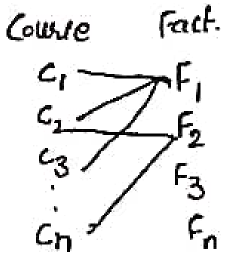


ex-

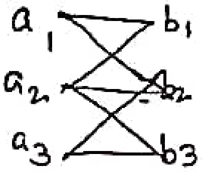


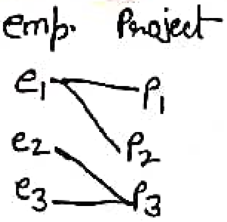
3) Many to one





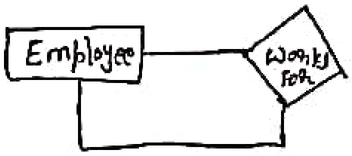
4) Many to Many



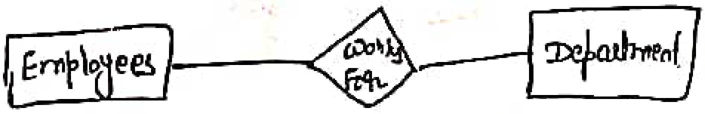


Types of relationship -

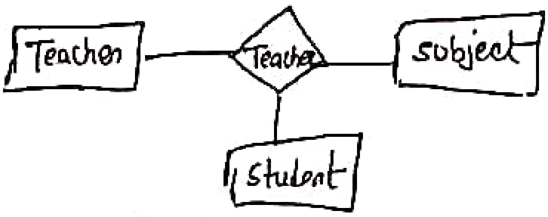
1) Unary -



2) Binary



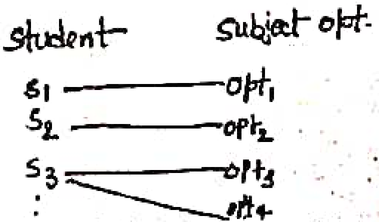
3) Ternary



Participation constraints - How an entity participates in a relationship.

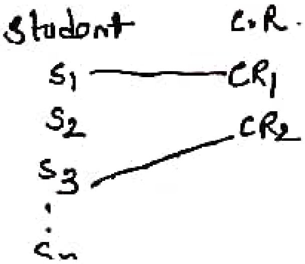
i) total participation

If every entity in E participates in atleast one relationship in R.



(ii) Partial participation

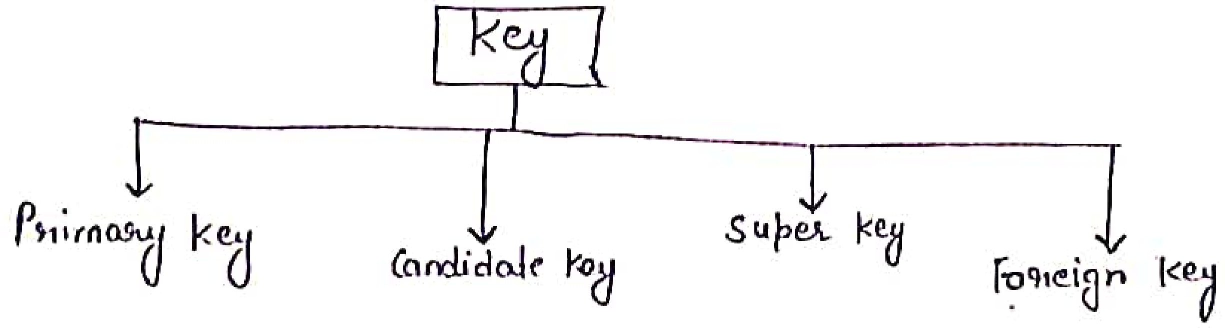
Some entities in E participates in the relation R.



KEYS

- It is used to uniquely identify any record or row of data from the table. It is also used to establish & identify relationships between tables.

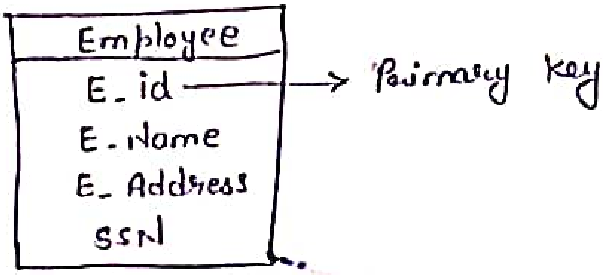
- Ex. In student table, ID is used as a key because it is unique for each student.



Primary key :-

- It is the first key which is used to identify one and only one instance of an entity uniquely.

ex-

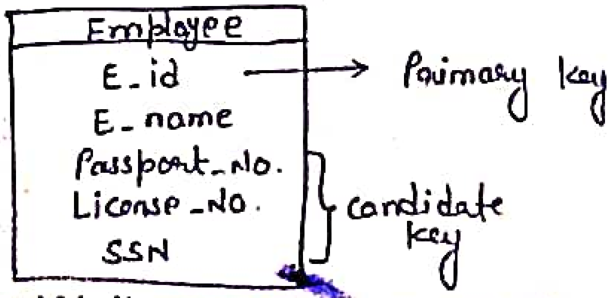


- one of the candidate key is chosen as the primary key with constraint that it can never have null values & duplicates.

Candidate key :-

- It is an attribute or set of an attribute which can uniquely identify a tuple.

- The remaining attributes except for primary key are considered as a candidate key.



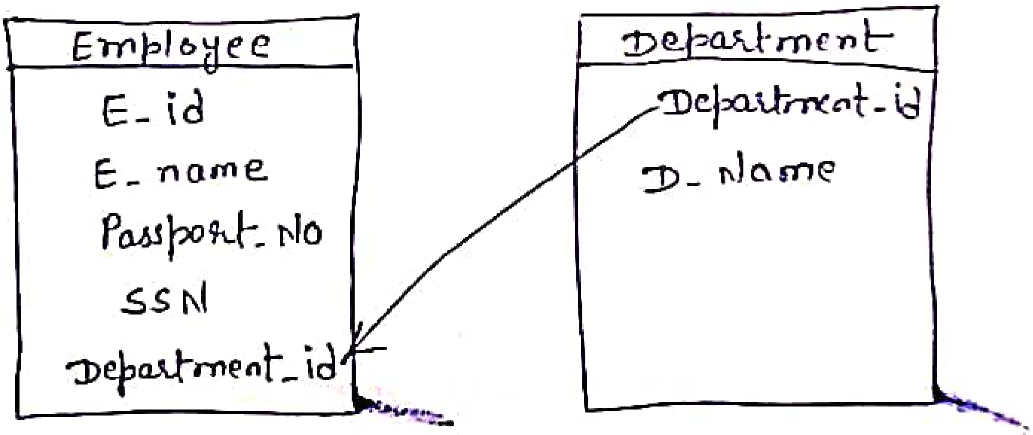
Super Key :-

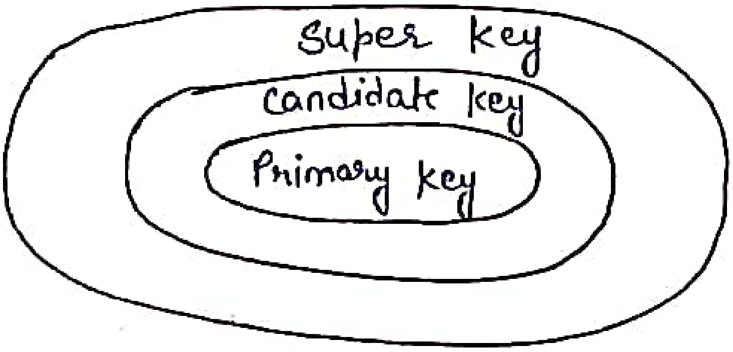
- It is set of an attribute which can uniquely identify a tuple.
- It is superset of a candidate key.

ex- In the above Employee table, (E-id, E-name) the name of employees can be the same, but their E-id can't be same. Hence their combination can also be a key.

Foreign key :-

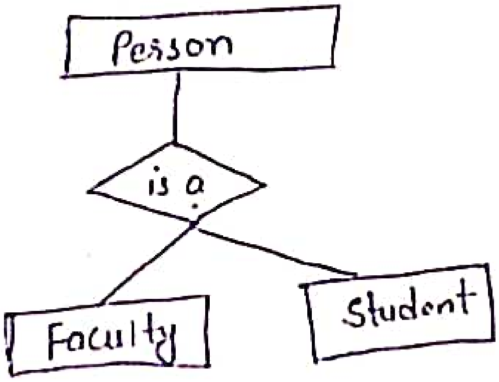
- Foreign keys are the column of the table which is used to point to the primary key of another table.





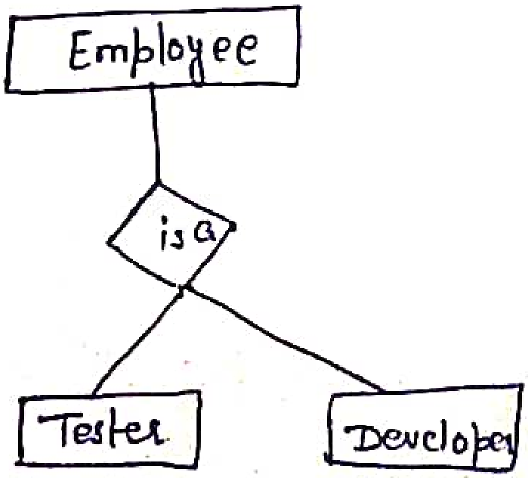
Generalization

- It is like a bottom-up approach in which two or more entities of lower level combine to form a higher level entity if they have some attributes in common.
- In this, entities are combined to form a more generalized entity. i.e; subclasses are combined to make a superclass.
- Ex-



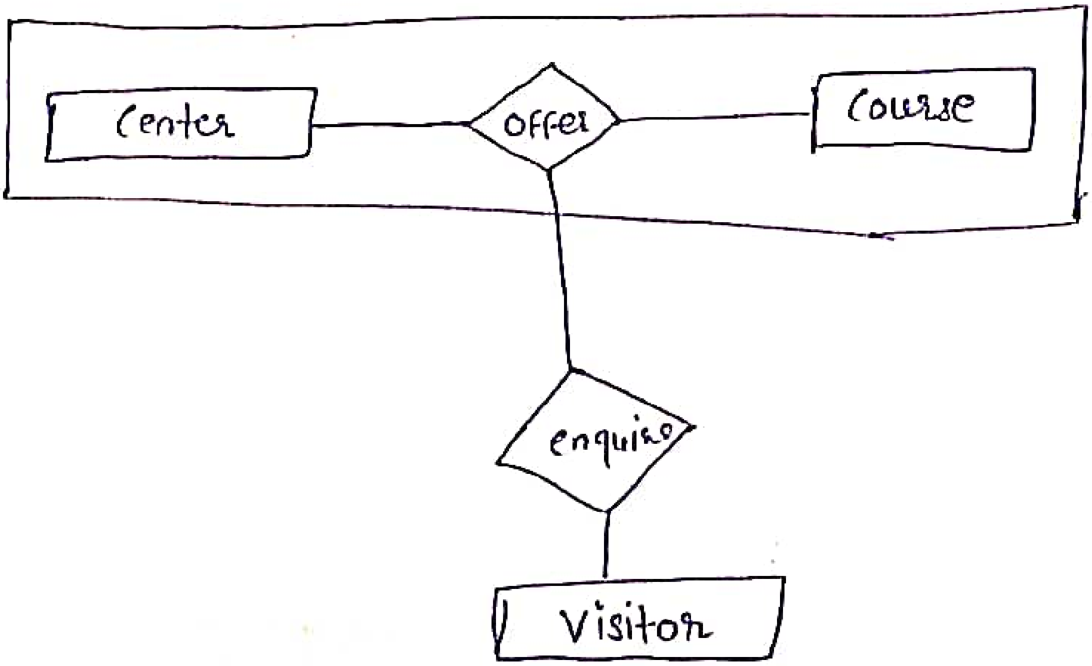
Specialization

- It is a top-down approach, and it is opposite to generalization.
- In this, one higher level entity can be broken down into two lower level entities.
- It is used to identify the subset of an entity set that shares some distinguishing characteristics.

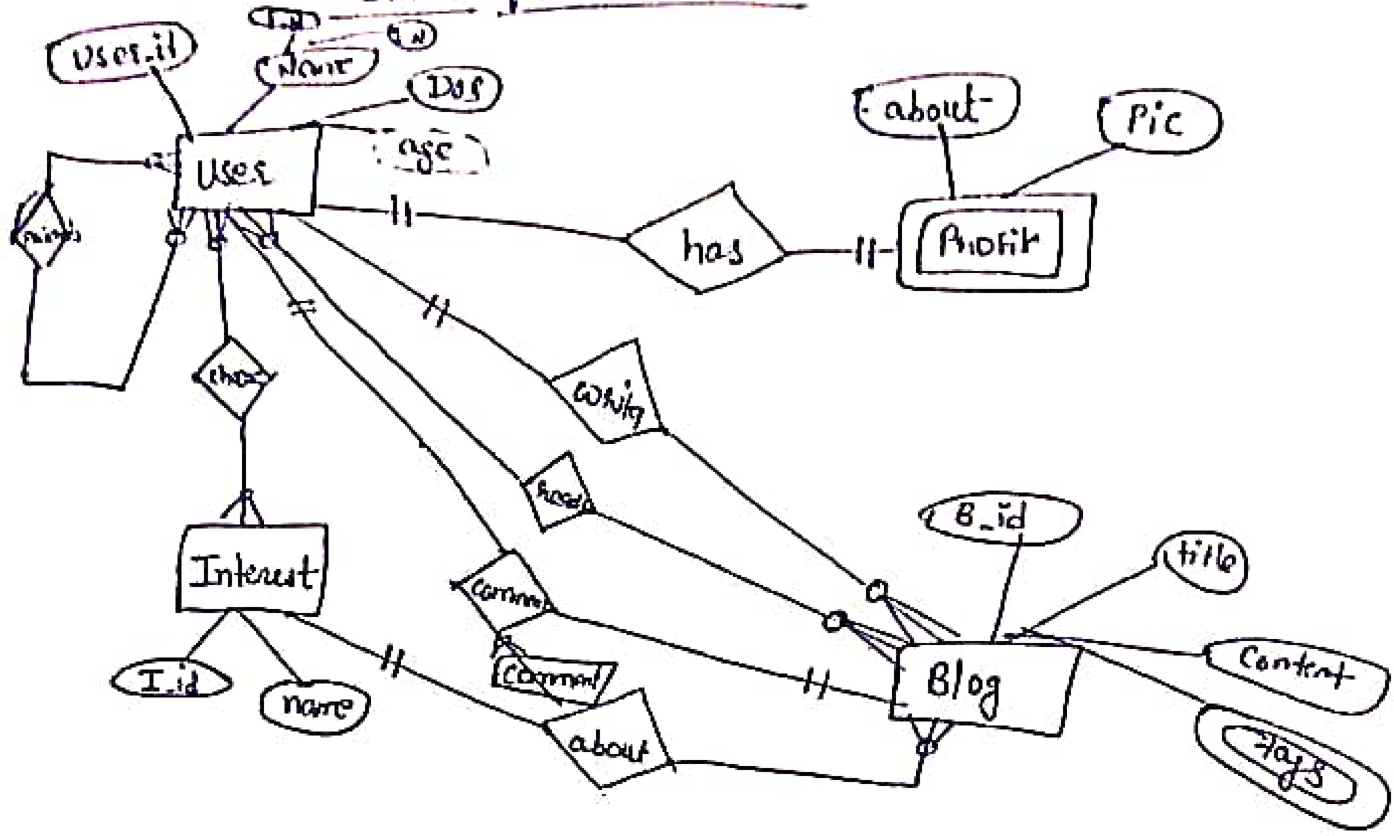


Aggregation

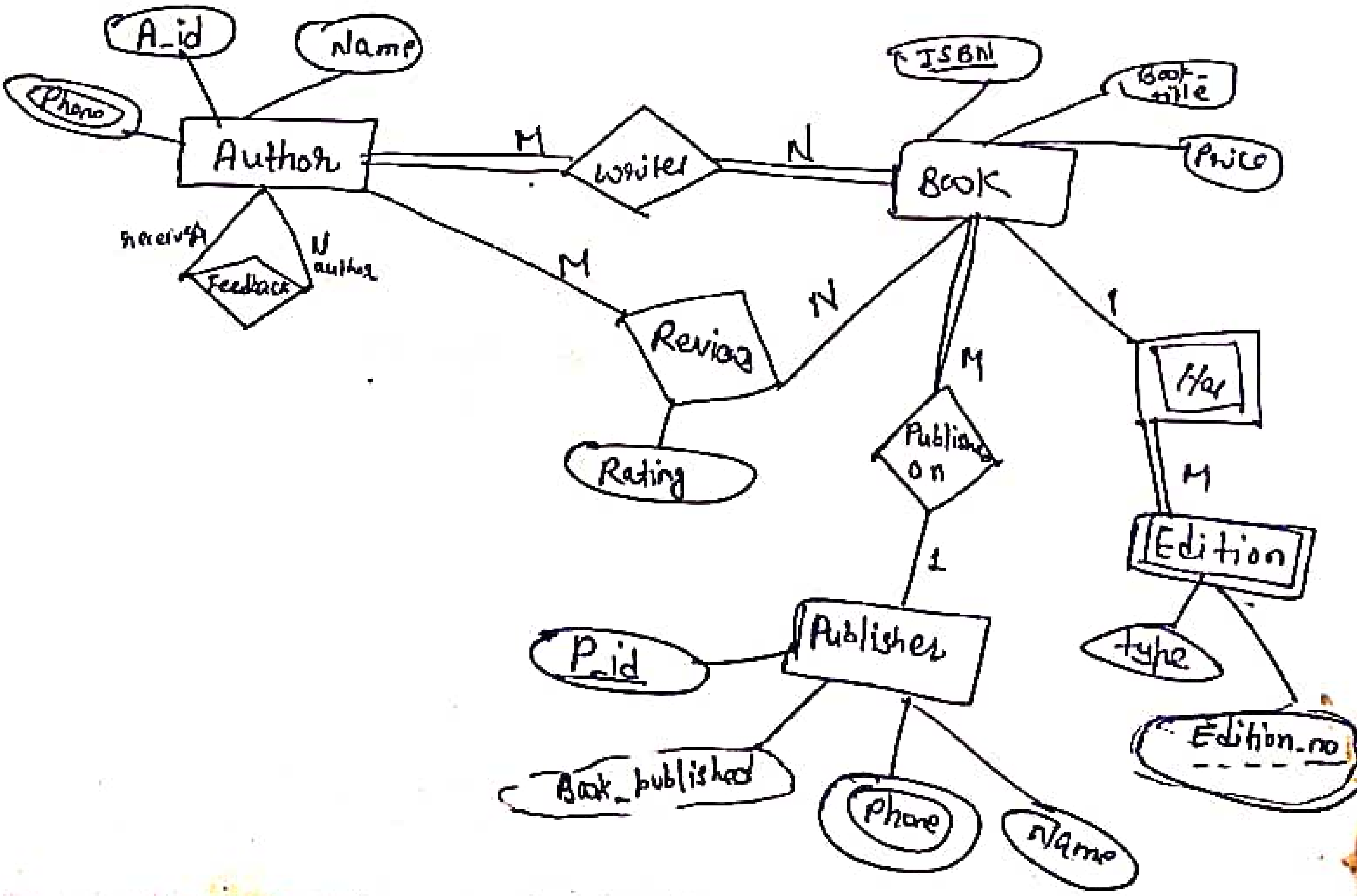
- In aggregation, the relation between two entities is treated as a single entity.
- In Aggregation, relationship with its corresponding entities is aggregated into a higher level entity.



ERD for Bbarlers

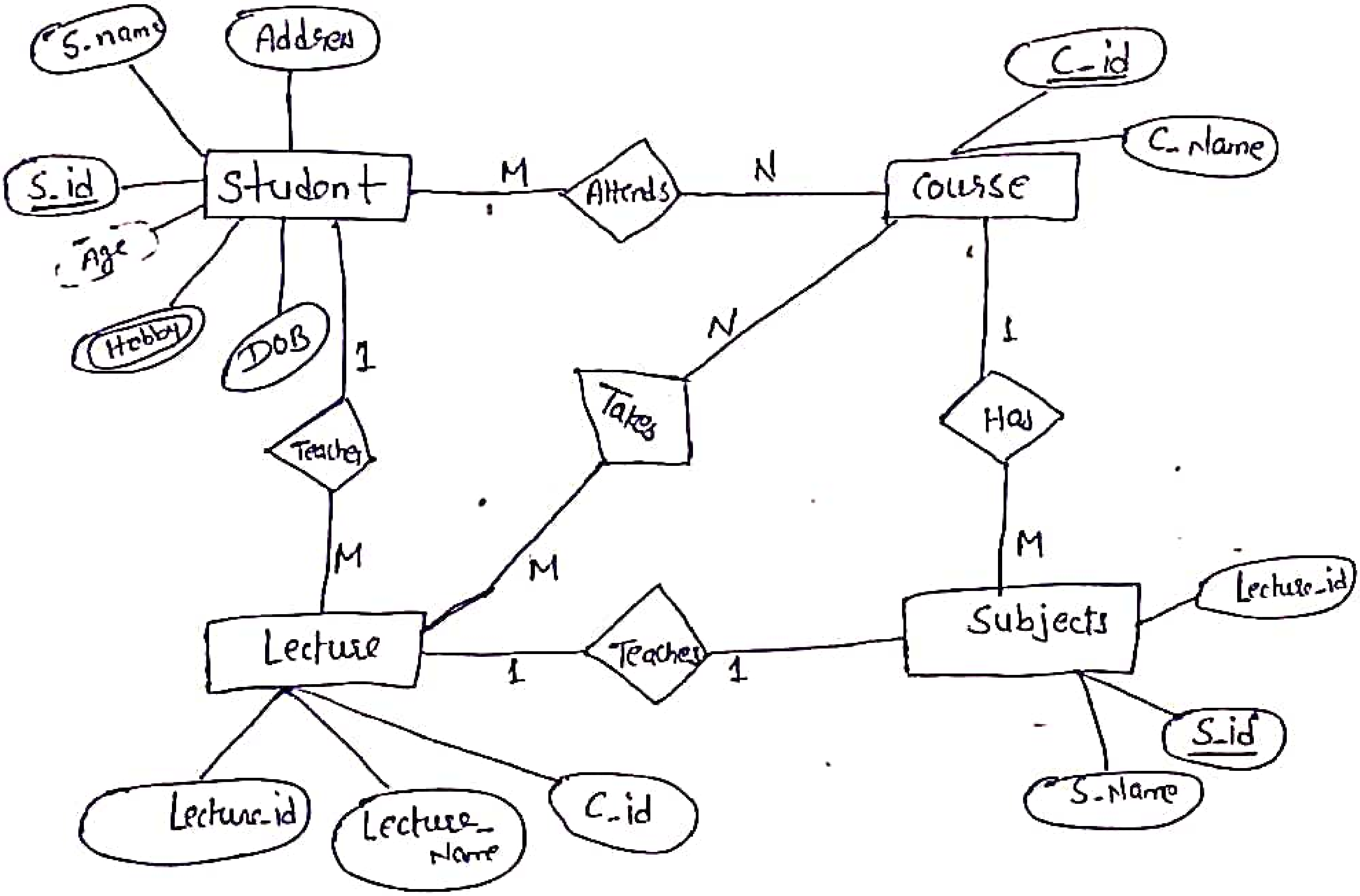


Online Book database



Reduction of ER diagram to Table

The database can be represented using the notations, and these notations can be reduced to a collection of tables.



- There are some points for converting the ER diagram to the table.

i) Entity type becomes a table.

ii) All single-valued attribute becomes a column for table.

iii) Key attribute of the entity type represented by the primary key.

iv) The Multivalued attribute is represented by a seperate table.

v) Composite attribute represented by components.

vi) Derived attributes are not considered in the table.

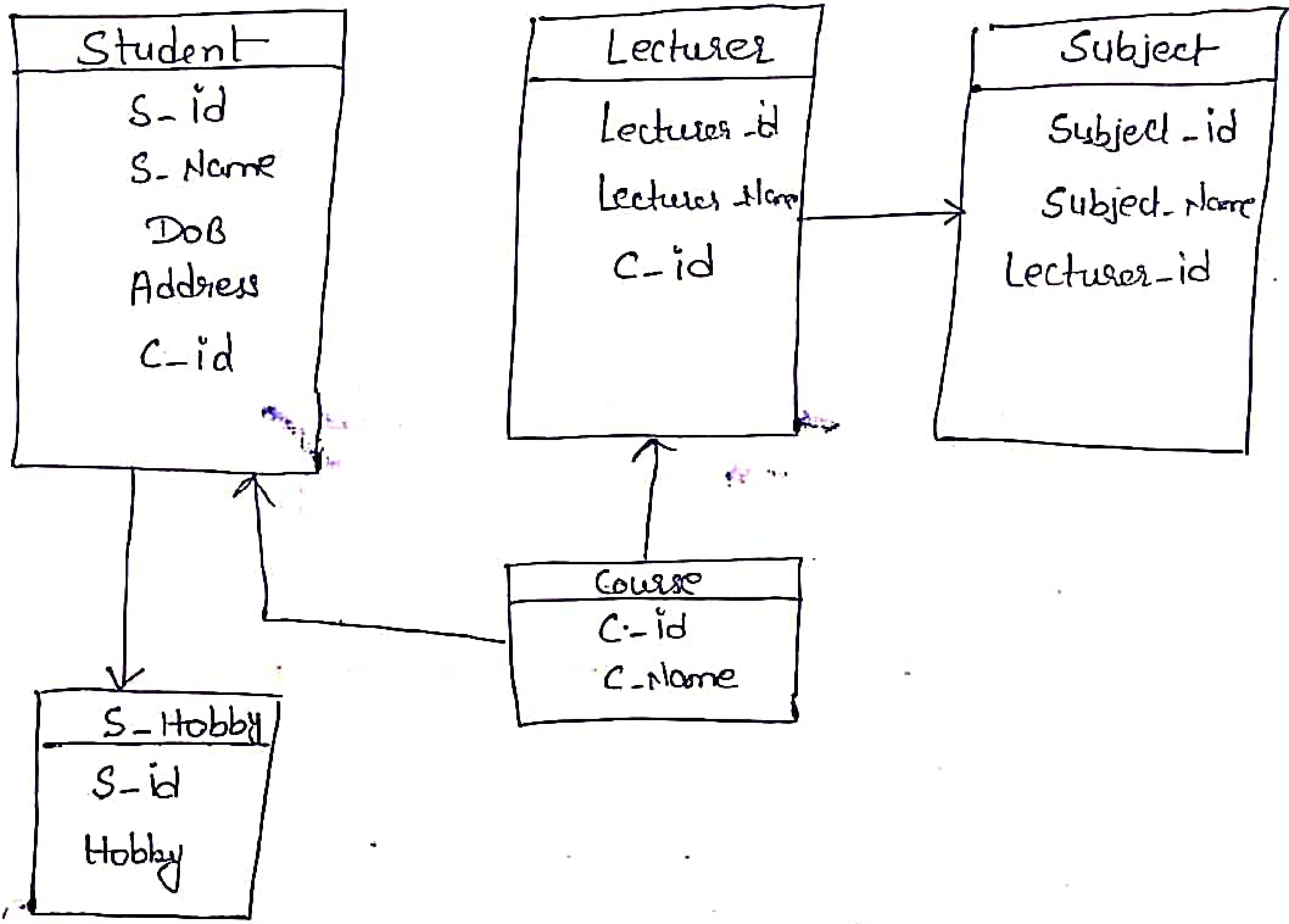
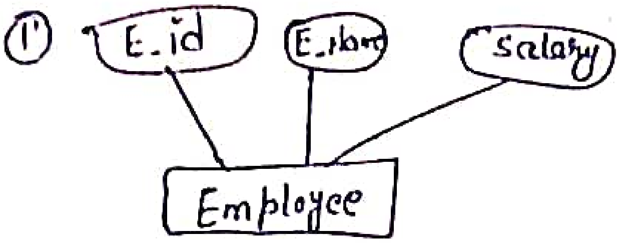


Fig - Table structure

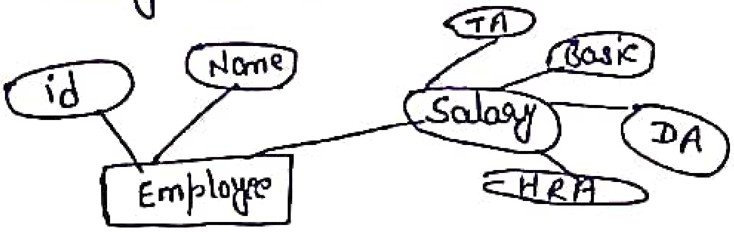
① Entity set



Employee

E.id	E_Name	Salary

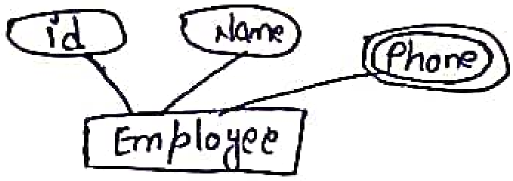
2) Entity set with composite attribute



Employee

id	Name	Basic	TA	DA	HRA

3) Entity set with Multivalued attribute



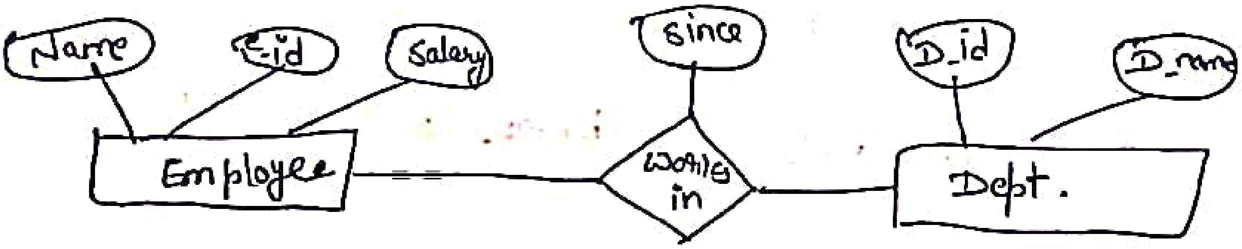
id	Name	Phone
1	Ram	954
1	Ram	854
1	Ram	964

i)

id	Name

id	Phone

4) translating relationship set into a table



Employee

Name	eid	salary

Works in

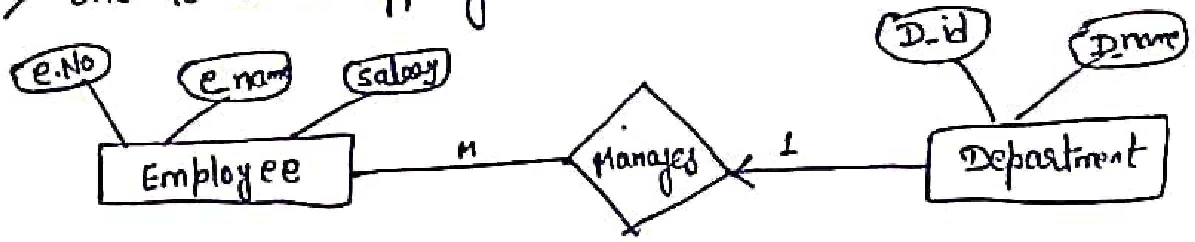
e_id	D_id	since

Dept.

D_id	D_name

- Relationship set with key constraint —

1) one to ~~one~~ Many Mapping



Employee

e_No	e_Name	salary
1	A	5K
2	B	10K

✓

Manages

e_no	D_Id
1	101
1	102
2	103

Department

D_id	D_name
101	CSE
102	IT
103	ECE

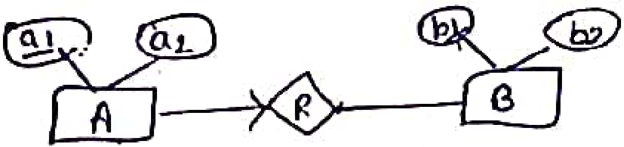
Merge these tables

✓

Did	D_name	e_no

→ only 2 table

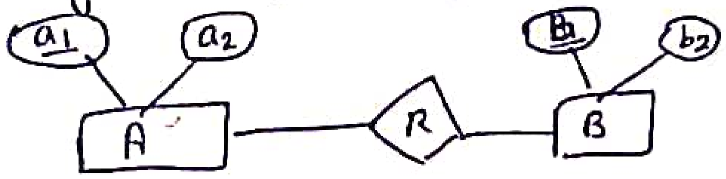
② Many to one Mapping



→ same as one to many Mapping

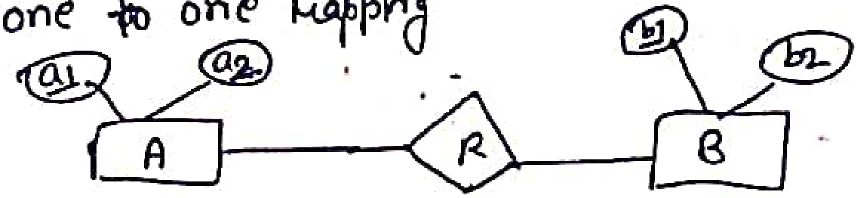
AR($\underline{a_1}$, a_2, b_1) , B($\underline{b_1}$, b_2)

③ Many to Many Mapping



A($\underline{a_1}$, a_2) , B($\underline{b_1}$, b_2) , R($\underline{a_1}$, $\underline{b_1}$)

④ one to one Mapping



AR($\underline{a_1}$, a_2, b_1) , B($\underline{b_1}$, b_2)

or

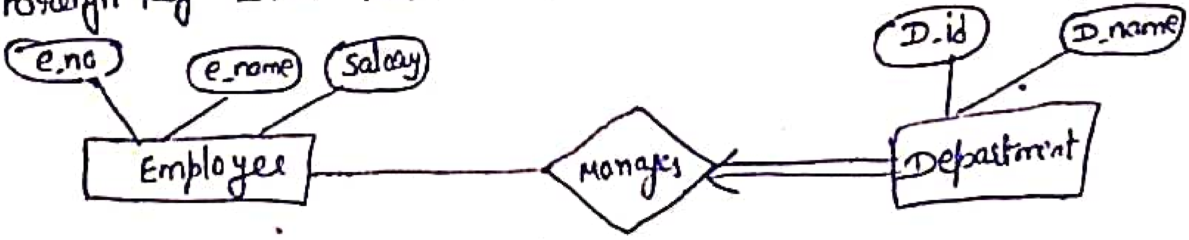
A($\underline{a_1}$, a_2) , BR($\underline{b_1}$, b_2, $\underline{a_1}$)

* Relationship set with key constraint + participation constraint.

- Each dept. is required to have exactly one employee as a manager.

- if there is a key constraint, merge the relationship set table with an entity set table.

- If the entity set totally participating with relationship set then Foreign key with Not Null constraints.



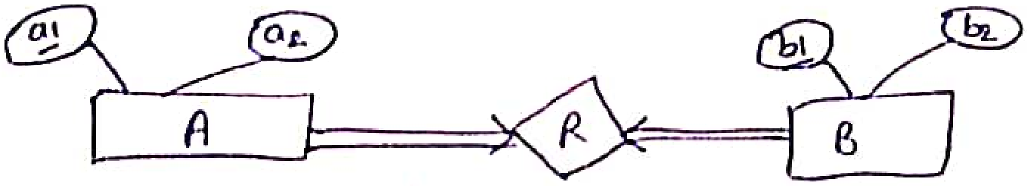
Employee

e.no	e.name	salary

Dept. Manager

D.id	D.name	e.no.

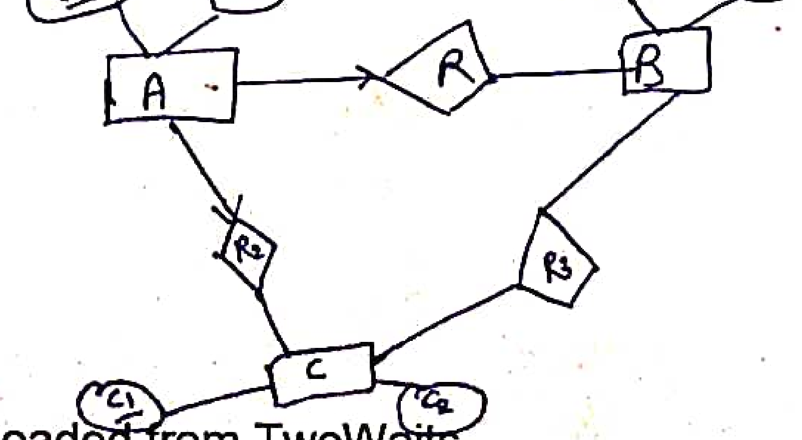
Q.



→ If there is a key constraint from both the sides of an entity set with total participation then we represent that binary relationship using single table.

⇒ A + B ($\underline{a_1}$ a_2, $\underline{b_1}$, b_2)

Q. Find the min. no. of tables that are possible when you translate the above E-R diagram into Relational Model.



⇒ 4 tables

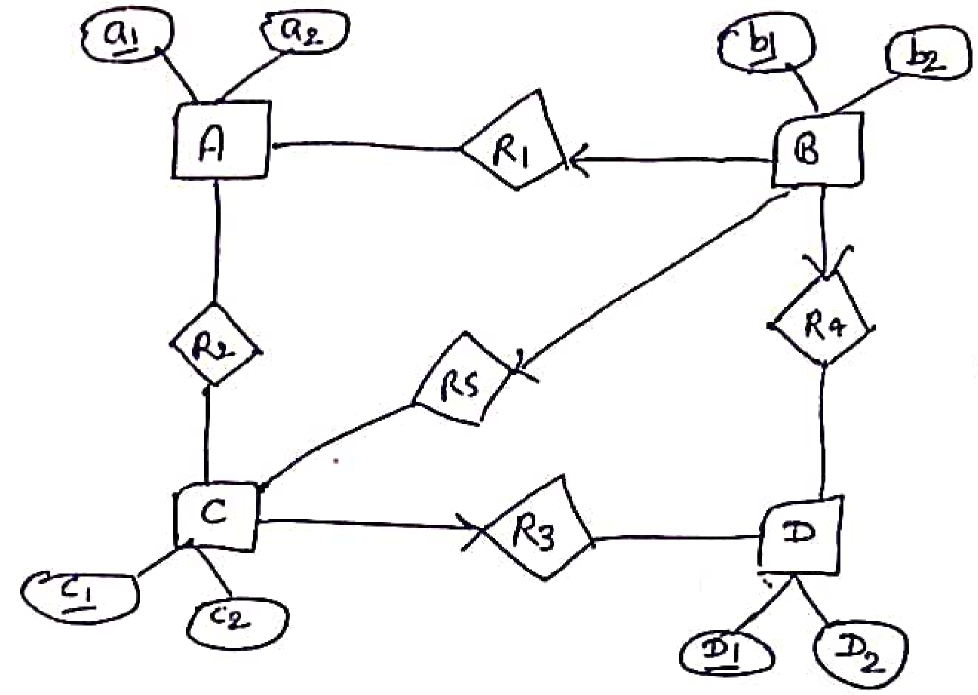
B ($\underline{b_1}$, b_2)

C ($\underline{c_1}$, c_2)

R_3 ($\underline{b_1}$, c_1)

ARR_2 ($\underline{a_1}$, a_2, $\underline{c_1}$, b_1)

Q



Min. No of tables = ?

=> $A(\underline{a_1}, a_2)$

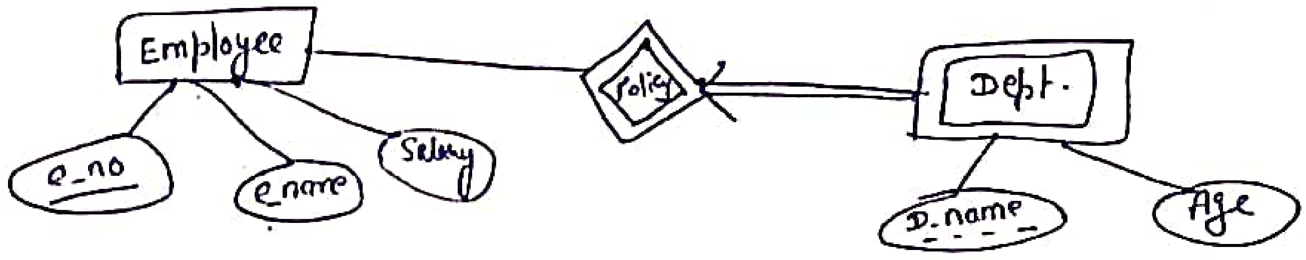
$D(\underline{D_1}, D_2)$

$BR_1R_4R_5(\overset{F.K}{\underline{b_1}}, b_2, \underline{D_1}, \overset{F.K}{\underline{c_1}}, a_1)$

$R_2(\underline{a_1}, \underline{c_1})$

$CR_3(\underline{c_1}, c_2, \underline{D_1})$

Q.

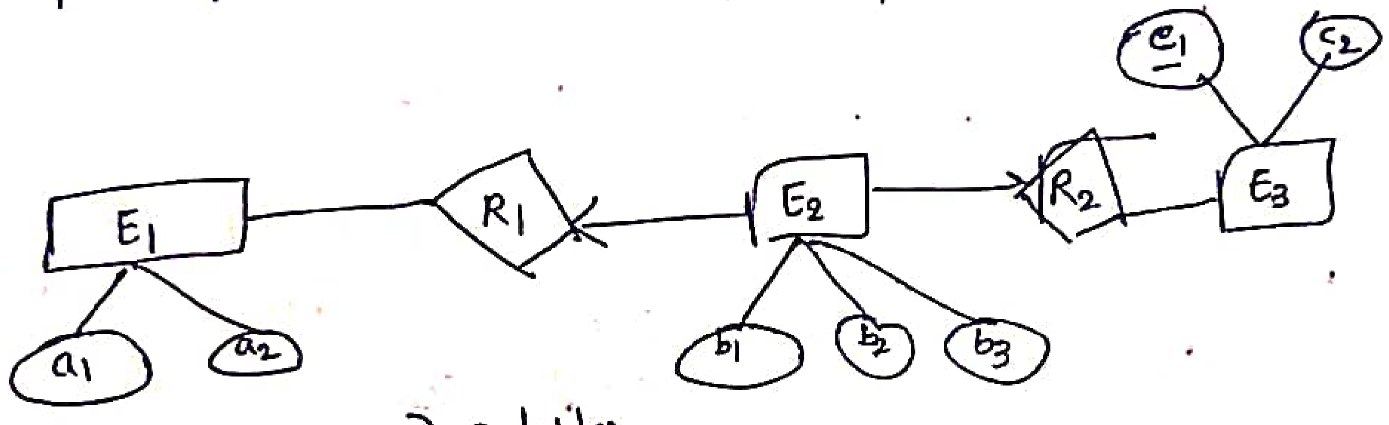


Employee

e_no	e_name	salary

e_no	D_name	age

Q.



=> 3 tables

$E_1(\underline{a_1}, a_2)$ $E_2(b_1, b_2, b_3, a_1, c_1)$ $E_3(\underline{c_1}, c_2)$

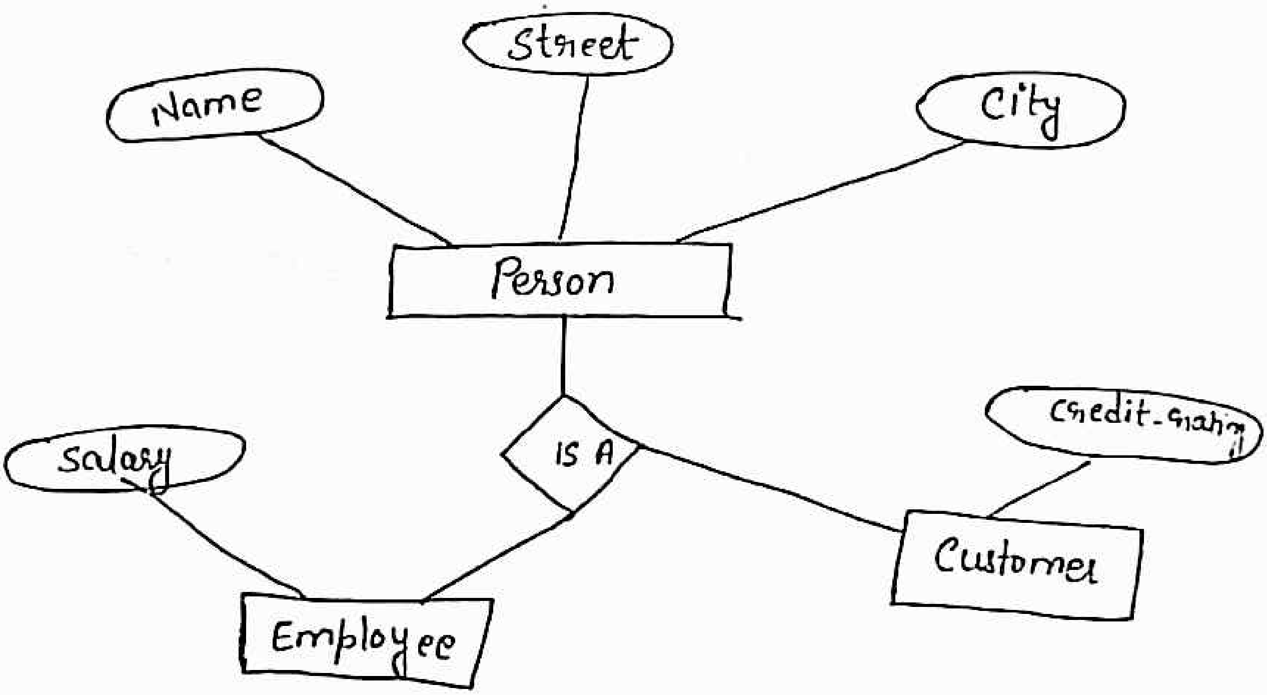
Extended E-R Model

- The E-R Model that is supported with the additional semantic concepts is called extended entity relationship model or EER model. The EER model includes concepts of the original E-R model together with the following additional concepts.

  i) specialization
  ii) Generalization
  iii) Aggregation

Generalization :- • It is the process of defining a more general entity type from a set more specialized entity types.

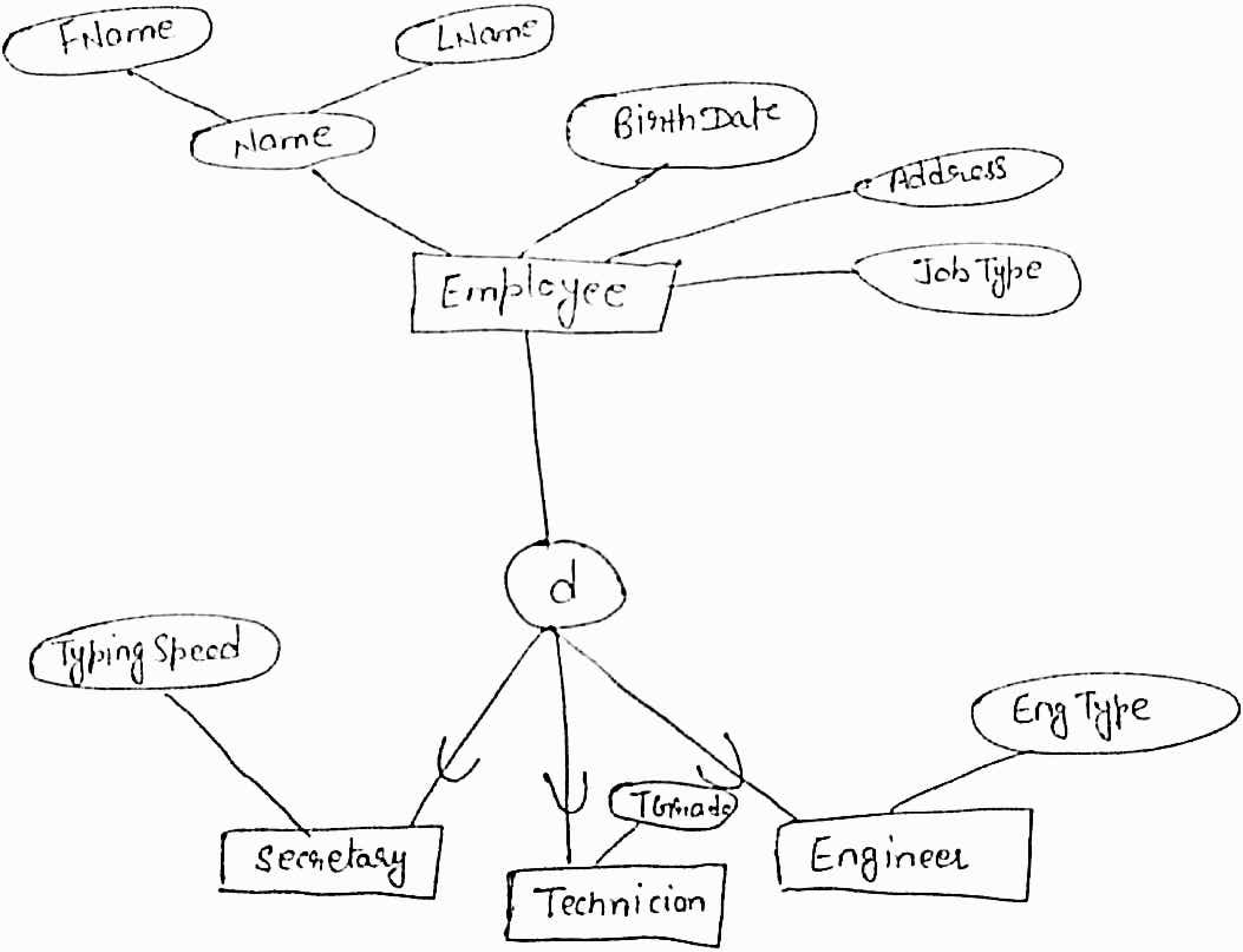
- It is a bottom-up approach.
- Ex -

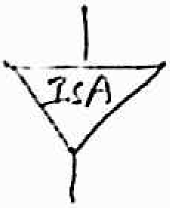


Person is a superclass, if Customer and Employee are subclass. Person as Higher Entity level and Customer and Employee as lower Entity model.

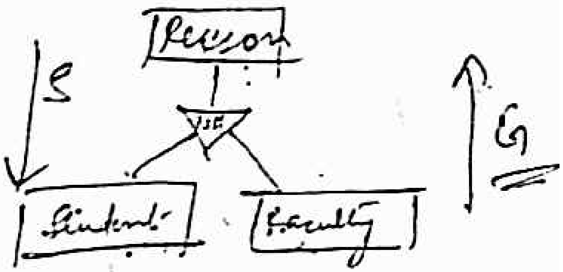
Specialization :– • It is the process of designing, subgrouping within an entity set.

• It is a top-down process.



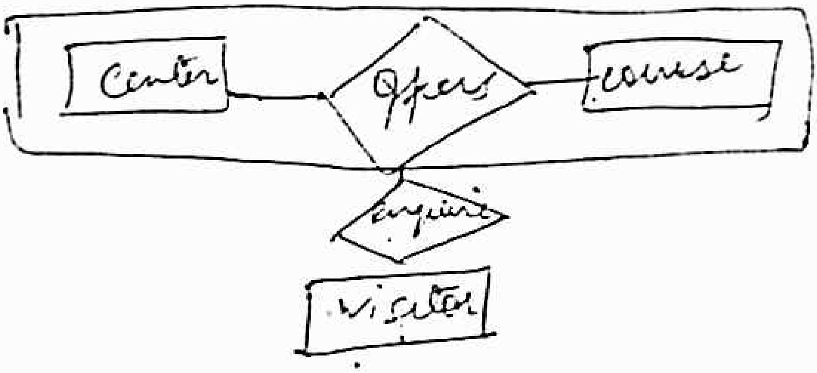


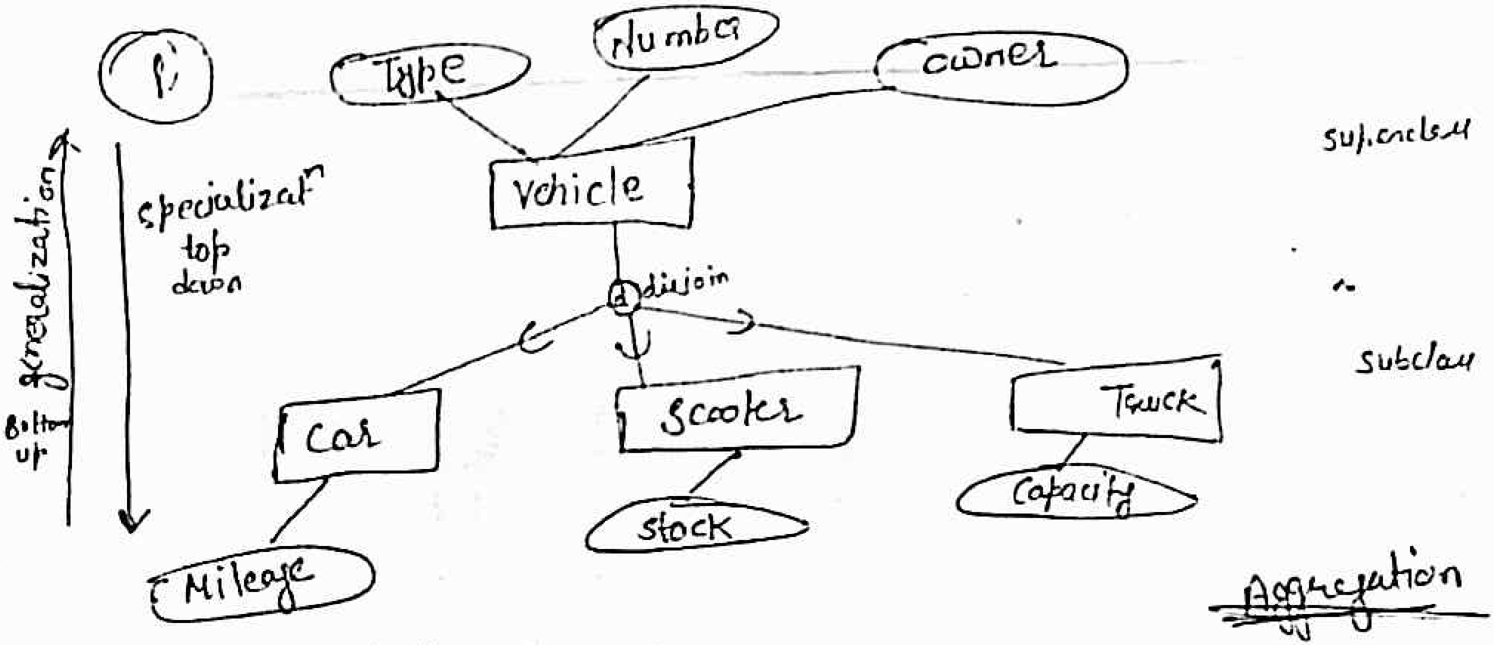
1) Generalization
(Bottom-up approach)



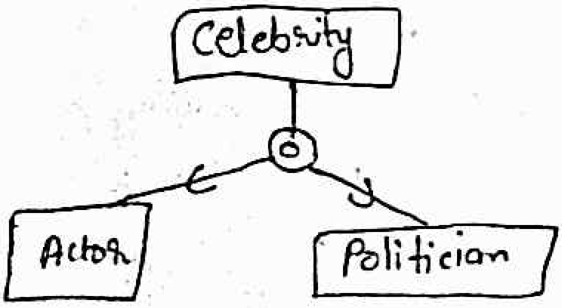
ii) Specialization → (Top-down approach)

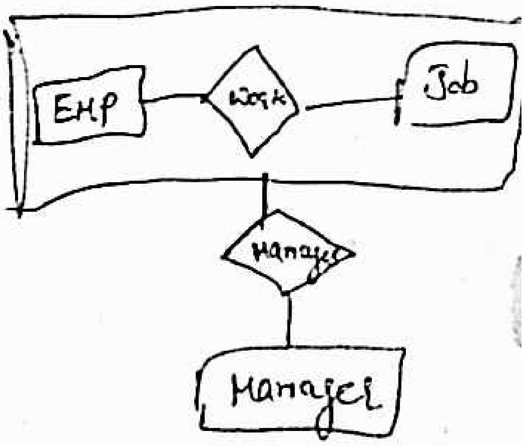
iii) Aggregation



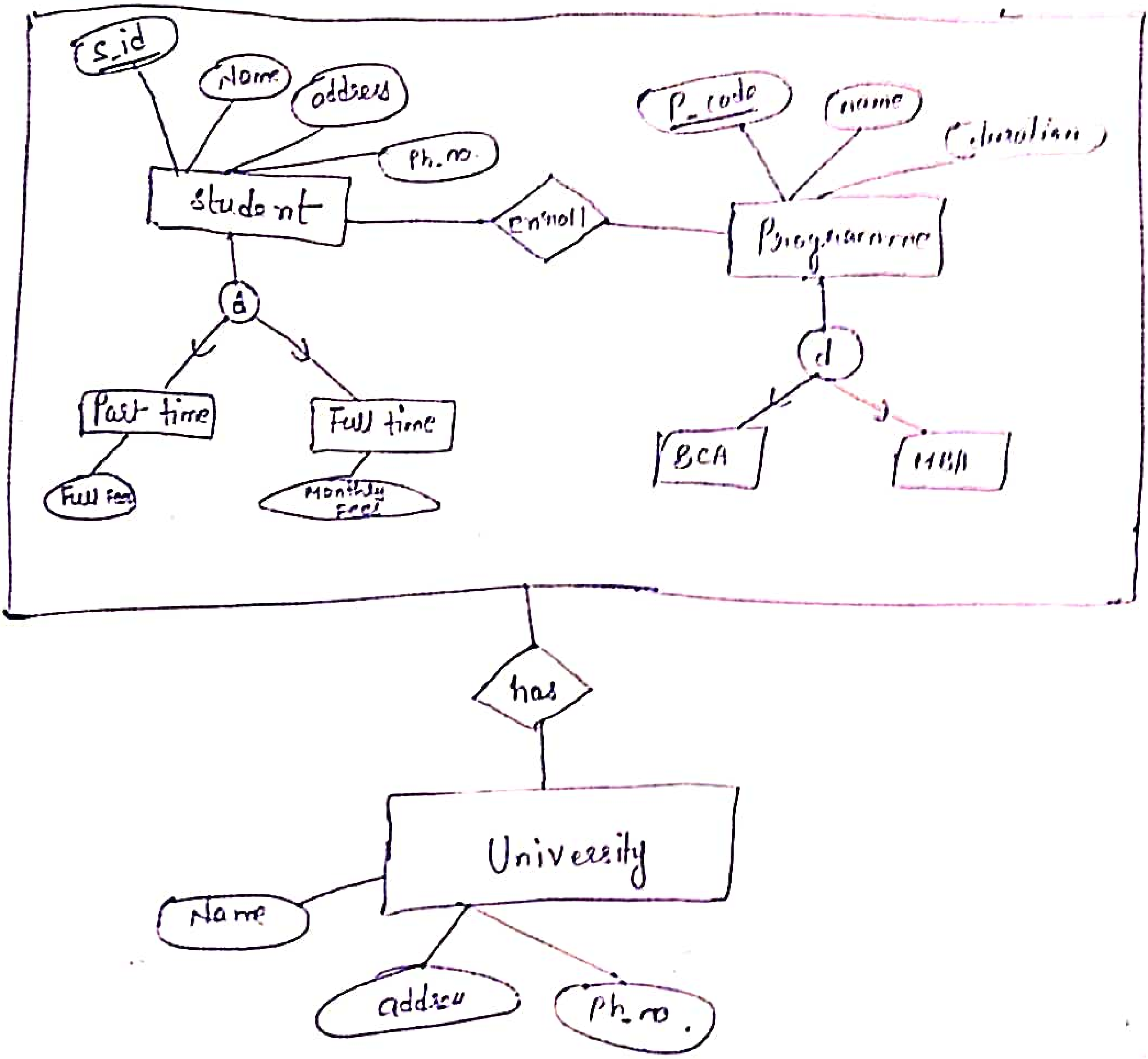


Aggregation





* Construct an EER diagram for the following description.



Q A University maintains records of its students & programmes in which they have enrolled. It stores student i.d, name, address & Ph_no. of student. and Program contains P_code, P_name & duration. A student is either a part time or a full time student (only one of the types). A student can register for many programme and a prog. can have many students