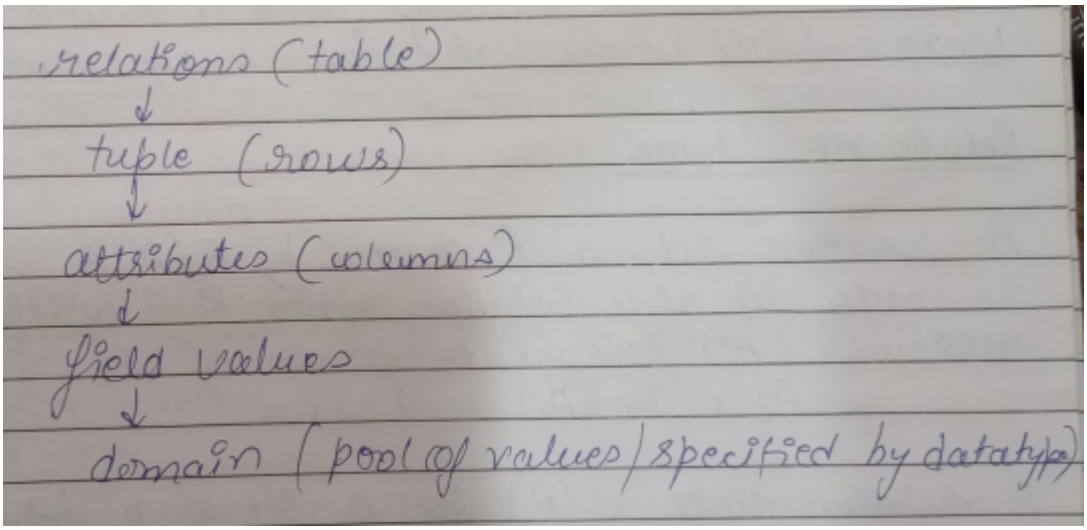
UNIT 2: Relational and some stuff

Relational Data Model



Relational Data Model Concepts

NOT WORTH LEARNING FOR MID

Relational Data Model Concepts

The relational model was first introduced by E.F. Codd in 1970. The model uses the concept of mathematical relation which looks somewhat like a table of values as it's

Spiral

Date....21/3/23.

basic building block.
The relational model represents the database as a collection of relation. Each relation resembles a table of values & is assigned a unique name. When a relation is thought of table, each row in the table (tuple) represents a collection of related values. In the formal terminology, a column header in the table is called attributes.

Domain → A domain is a set of atomic values
By atomic we mean that every value is indivisible. A common method of specifying a domain is to specify a datatype from which the datavalues can be taken. When we specify a datatype, we specify both the value & the format. A domain is also given a name.

Note:- Null is the value for every domain.

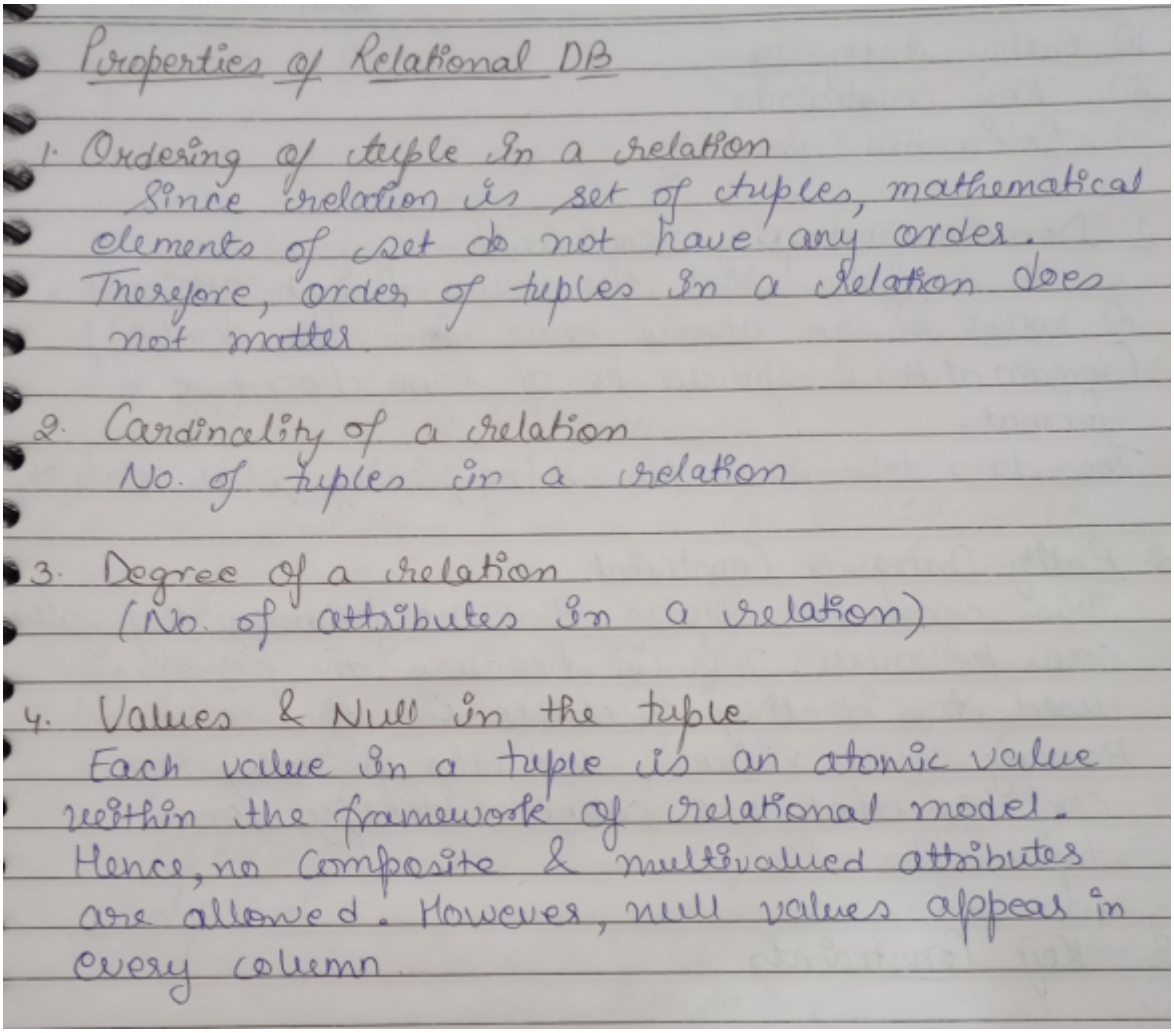
Relational Schema

A relation schema R denoted by $R(A_1, A_2, \ldots, A_n)$ is made up of a relation name R & a list of attributes (A_i) belongs to domain of A_i.

$A_i \in dom(A_i) \quad 1 \le i \le n$

A relation state $r(R)$ is a set of n tuples $r = \{t_1, t_2, \ldots, t_n\}$ where each tuple t is an ordered list of n values i.e. $t = \langle v_1, v_2, \ldots, v_n \rangle$ where each value $v_i \in dom(A_i) \quad 1 \le i \le n$.

Properties of Relational DB



Integrity Constraints

- way of implementing rules of DB
- restricts data that can be stored in relation
- helps us ensure INTEGRITY CORRECTNESS
- 4 types: Domain Integrity, Entity Integrity, Key, Referential

1. Domain Integrity Constraint
    - specify that value of each attribute A must be atomic value from domain(A) AND should be of same data type & format
    - data type includes String, Integer, Char, date, etc.
2. Entity Integrity Constraint
    - states that "no primary key value can be null".
    - because primary key used to id tuple in relation.

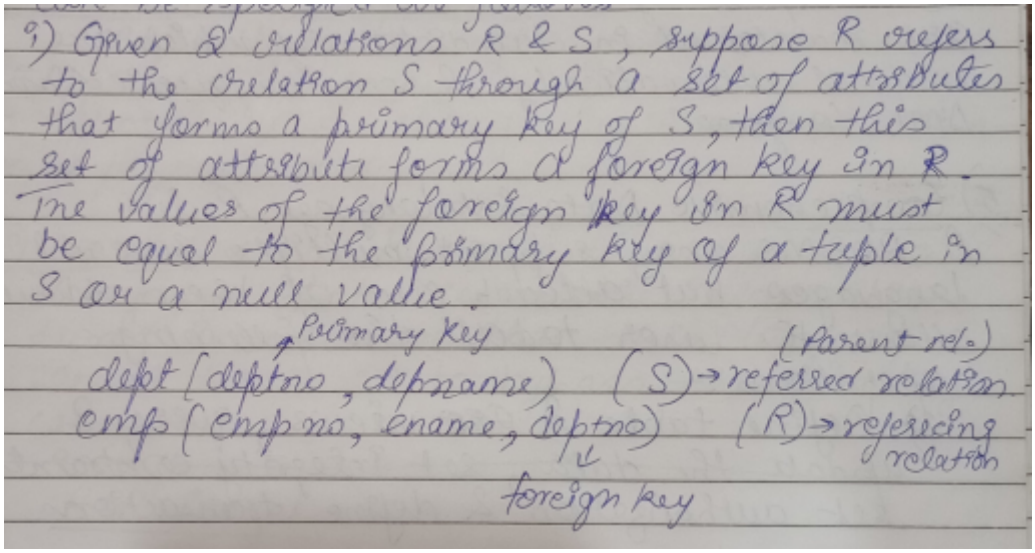
- having null as prim. key -> we cant id some tuples.

3. Key Constraints

- keys are entity set _ used to id an entiry within relation
- relation can have mulitple key, among them 1 must be primary and only 1, which have unique AND not null values.

4. Referential Integrity Constraint

- THIS specified b/w 2 relations.
- used to maintain consistency among tuples of 2 relations
- this rule concerned with FOREIGN key, i.e., attr. of relation having domain that is prim. key of another relation.
- rule can be specified as follows



Codd's Rule

1. Information Rule

- All info in relational DB including table name, col name are represented by values in table.

2. Guaranteed Access Rule

- Every piece of data in realtional DB can be accessed using COMBINATION of table name, col. name.

3. Systematic Treatment of Null Values

- RDBMS handles records having unknown values (NULL vals) in predefined manner.

4.

5.

6.

7.

5. Physical Data Independence

6. Logical Data Independence
7.
    11.
        12.

ER Model

- THIS perceives real world consisting of basic obj. called ENTITY, ATTR., and RELATIONSHIP among (us lul) those models.
- useful in mapping meanings & interaction of real world situation onto conceptual schema.
- use 3 basic notations: Entity set, Attr., and Relationship sets.

Entity

- obj. in real world that is distinguishable from other obj.
- each entity has attr., i.e., the prop. that describes it.

Entity Set

- set of entities of same type _ share same properties.

Attibutes

- prop. used to descibe obj / entity.
- For each attr., there is set of permited vals called DOMAIN of attr.
- can be of following types:

1. Composite vs. Atomic (Simple) Attributes
    - Composite attr. can be divided into sub parts which repre. moe basic attr. with independent meaning. EX- student's full name, etc.
    - can form a heirarchy
    - value of THIS is concatenation of values of its constituents simple attr.
    - used to model situation _ user sometimes refer to THIS as whole but other times refer specificallt to its components
    - Simple attr. can't be further divided.
    - EX- Phone number.
2. Multivalued vs. SingleValued Attr.
    - Single valued - have single value for particaular entity
    - Ex- Aadhar Card no.
    - But some attr., have set of vals for specific entity - Multivalued attr.
    - THIS may have LOWER and UPPER bound to contraint no. of values. allowed.

    - EX- email add. (1 person can have >1).

3. Derived attr.

    - value of THIS cna be derived from value of other related attr.
    - ITS value not stored, rather computed when required.
    - EX- age can be derived from data_of_birth.

4. Null val

    - attr. take THIS when entity doesnt have any value for it.
    - i.e., NULL val indicates not applicable / unknown / something missing.

realtion stuff

Relationship

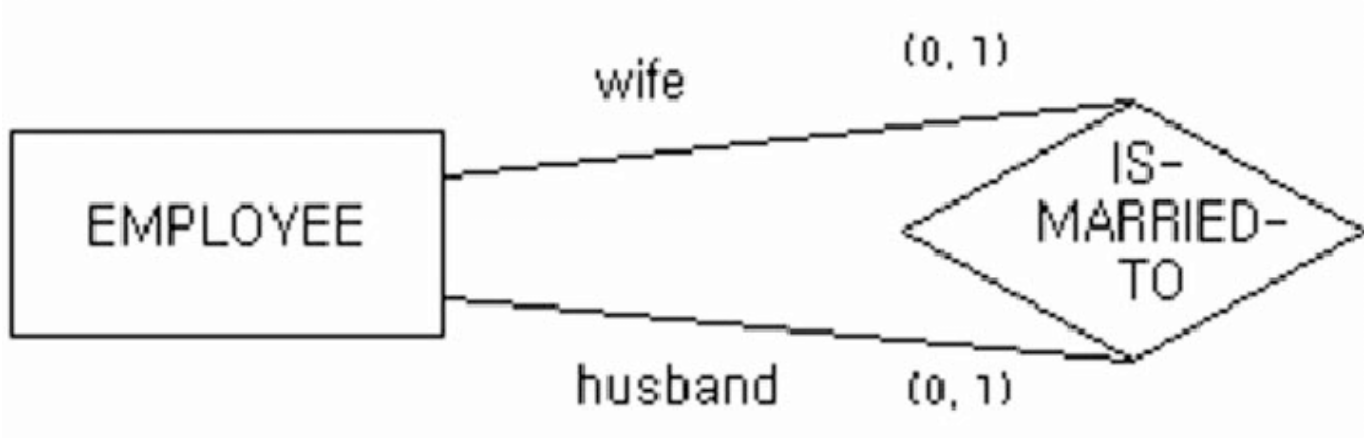
- association among several entities.
- relationsship set is set of relationship of same type.
- representeed by <>.

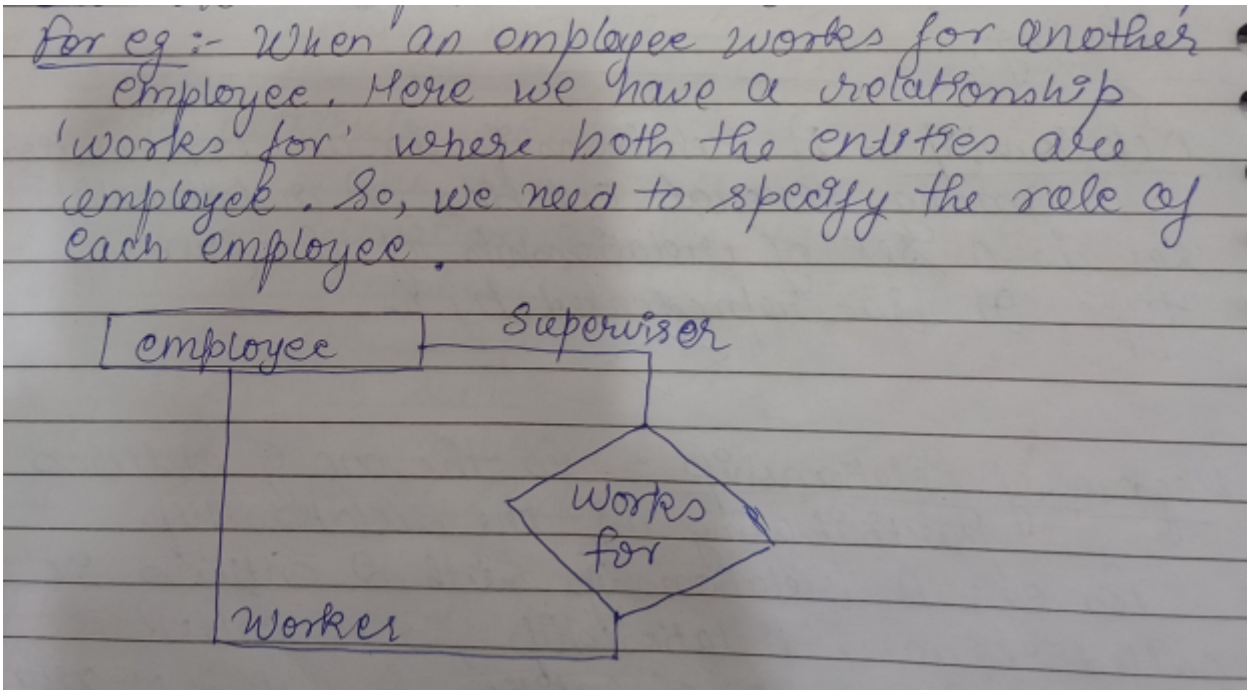
Degree of realtionship

- no. of entities participating in relationship.
- EX- relationship with 2 entities = binary relationship, 3 = ternary, n = n-ary

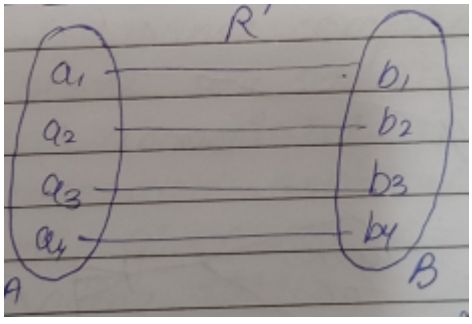
Role name

- each entity in relationship play particular role.
- THIS signifies what role it plays
- useful when entity participate in relationship >1 in different roles.
- here IT become vital.
- such relationships are called RECURSIVE RELATIONSHIP.

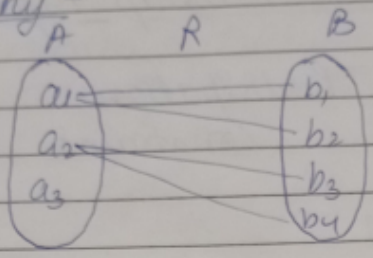




Constraints on relationship type

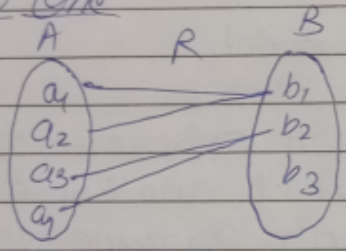
1. Cardinality ratio
    - expresses no. of entites to which another entity can be associated with.
    - ASSUME 2 entities
        1. One - ONE:
            1. ent. in A -> only 1 ent. in B and vice versa.
            2. 

ii) One to Many -



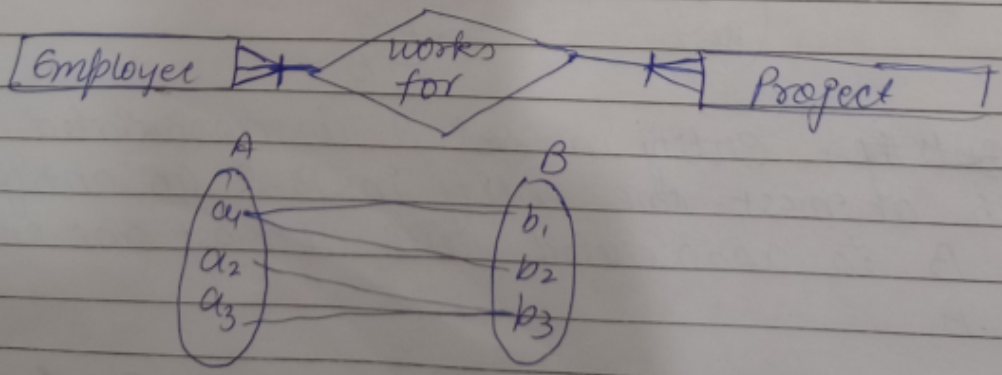
An entity in A is associated with any no. of (0 or more) entities in B. An entity in B can be associated with almost one entity in A.

iii) Many to One -



An entity in A is associated with almost one entity in B. An entity in B however can be associated with any no. of entities in A.

iv) Many to Many

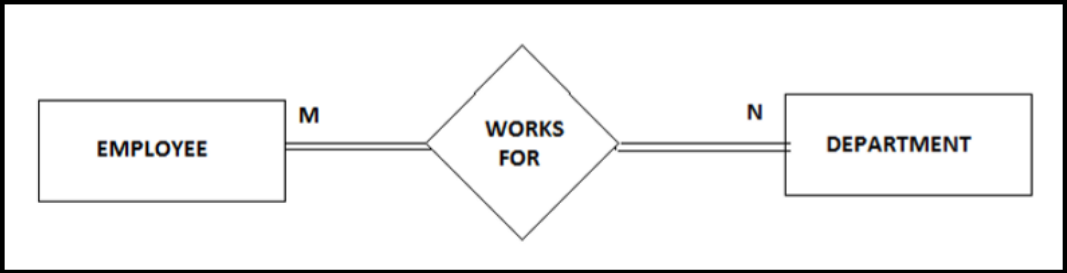


2. Paritcipation Constraints

- specifies existence of ent. when its related to another entity in relationship.

Total Participation

Each entity in the entity set is involved in at least one relationship in a relationship set i.e. the number of relationship in every entity is involved is greater than 0.

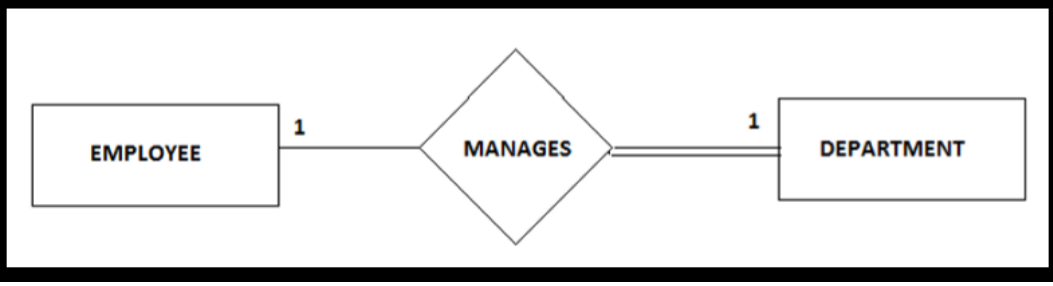


Consider two entities Employee and Department related via Works_For relationship. Now, every Employee works in at least one department therefore an Employee entity exist if it has at least one Works_For relationship with Department entity. Thus the participation of Employee in Works_For is total relationship.Total Participation is represented by double line in ER diagram.

Partial Participation

Each entity in entity set may or may not occur in at least one relationship in a relationship set.

For example: Consider two entities Employee and Department and they are related to each other via Manages relationship. An Employee must manage a Department, he or she could be the head of the department. But not every Employee in the company manages the department. So, participation of employee in the Manages relationship type is partial i.e. only a particular set of Employees will manage the Department but not all.

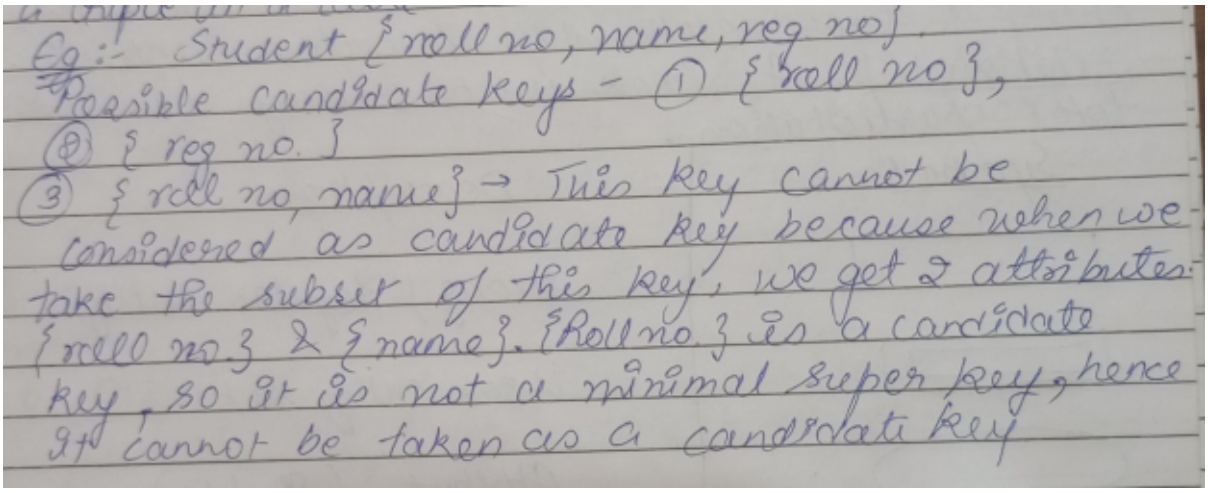


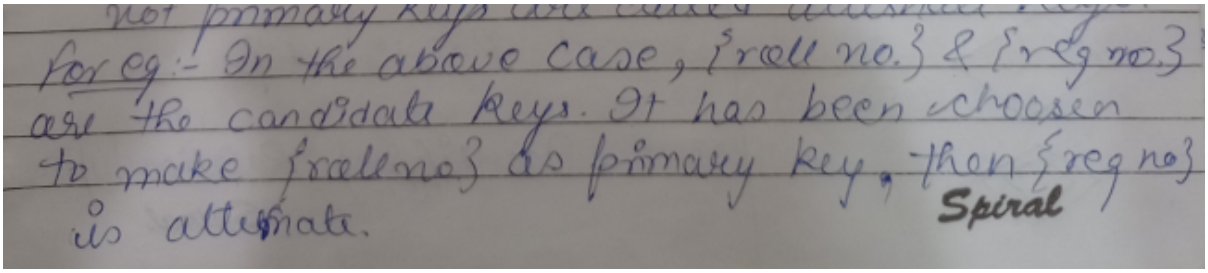
KEY

- attr. / set of attr. _ helps in uniquely id rows of table.
- can be types: SUPER, CANDIDATE, PRIMARY, ALTERNATE, FOREIGN

1. SUPER KEY
    - combination of all possible atttr. _ which can uniquely id rows in table.
    - can have EXTRA attr. not needed to uniquely id rows.
    - EX- FOR KEYS = Roll no., Name, reg. no.
    - so possible combinaations is power set of these. except null.

2. CANDIDATE KEY
    - minimal SUPER key with NO REDUNDANT attr,
    - called ↑ , because we select CAND. from set of SUPER, such that minimum attr. needed to id rows.

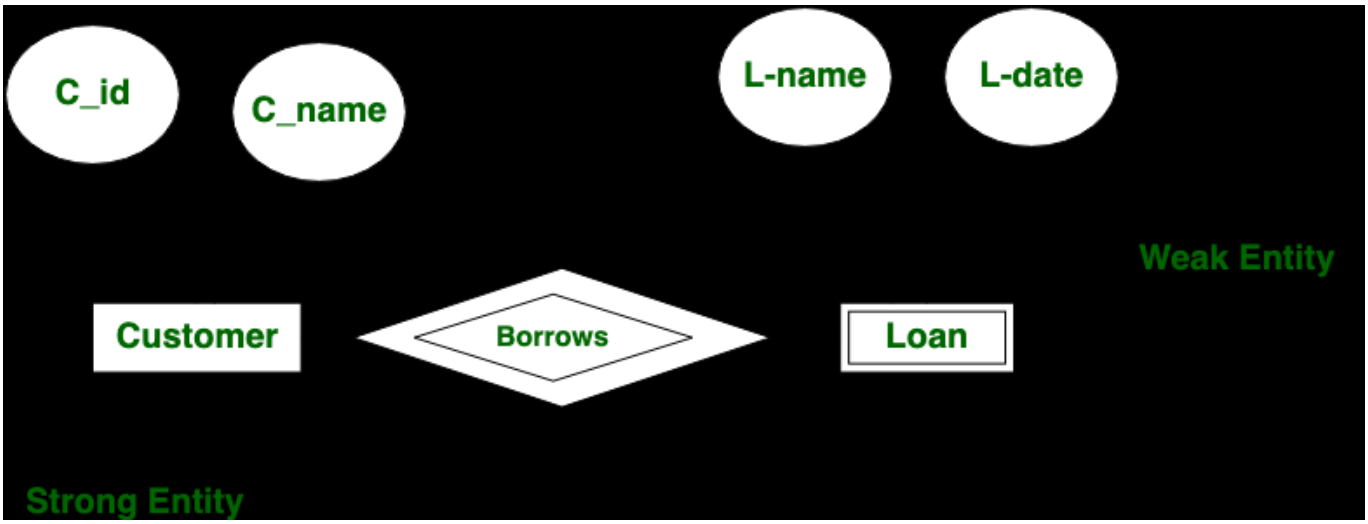
- 

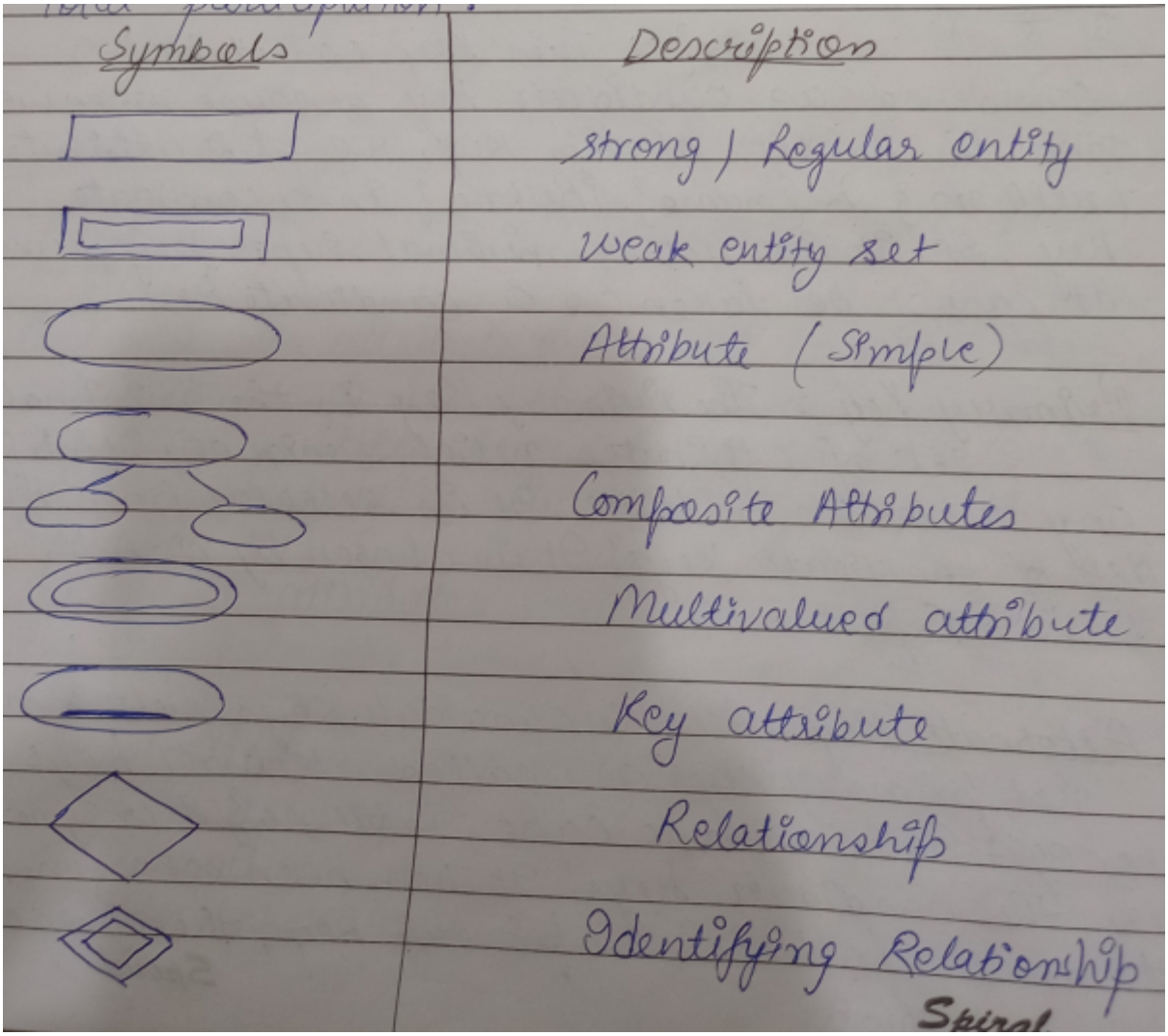
3. PRIMARY KEY
    - minimal set of attr. _ which uniquely id rows.
    - selected from CANDIDATE KEY by DBA.
4. ALTERNATE KEY
    - CANDIATE KEYS - PRIMARY KEY. (- is minus)
    - 

A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table.

Strong & Weak Entity

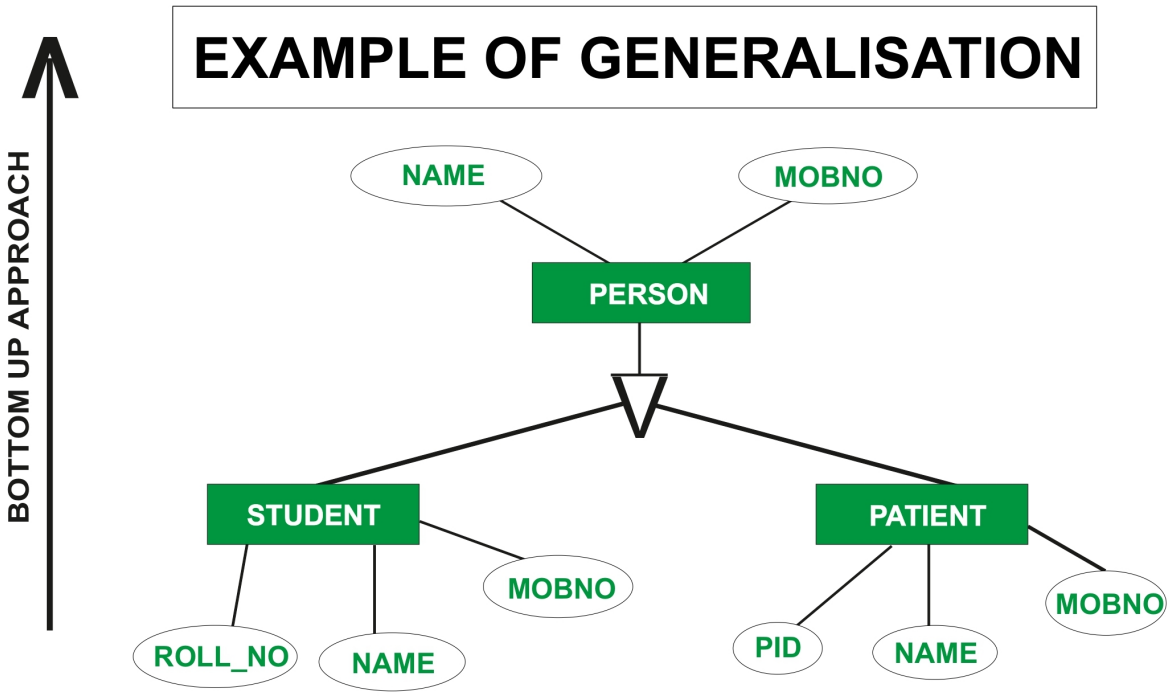
- Strong - ent. having sufficient attr. _ which can used as CANDIDATE KEY
- WEAK - opposite of ↑, and depends on another entity for their existence.
- ↑ id'ed with another entity set -> identifying entity set
- every weak MUST BE ASSOCIATED with an identifying entity and such relationship = identifying relationship





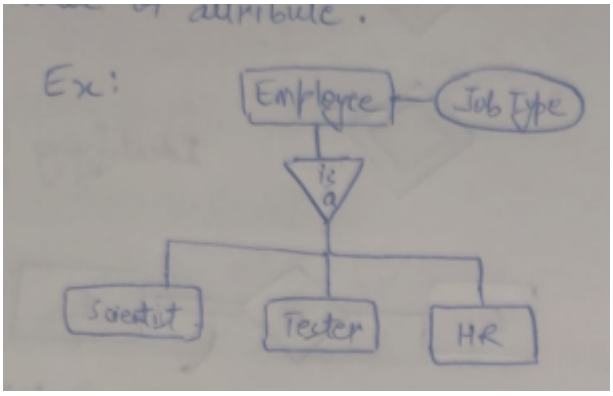
Extended ER

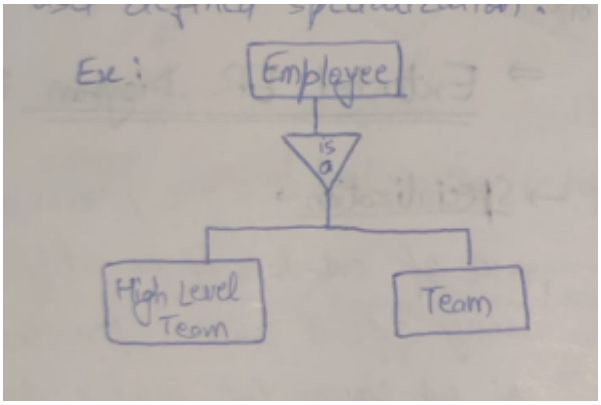
Specialization and Generalization



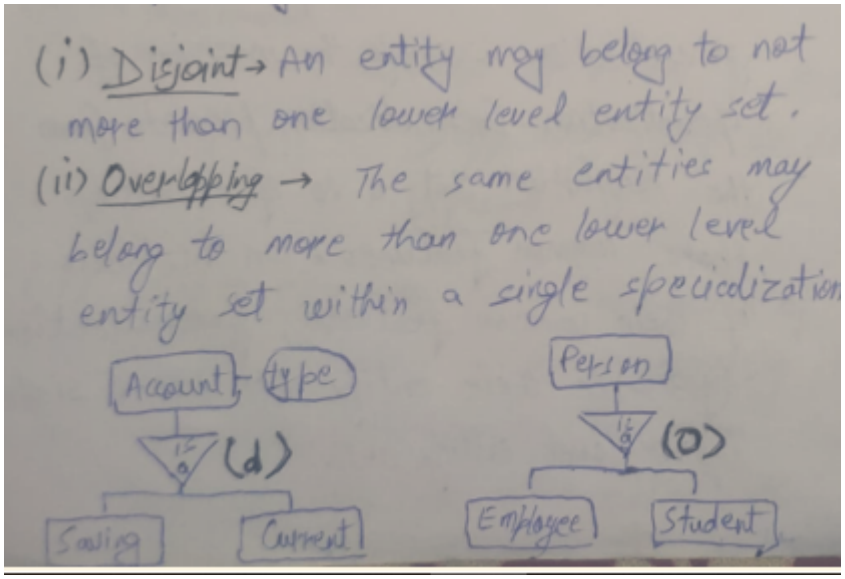
GENERALIZATION	SPECIALIZATION
Generalization works in Bottom-Up approach.	Specialization works in top-down approach.
In Generalization, size of schema gets reduced.	In Specialization, size of schema gets increased.
Generalization is normally applied to group of entities.	We can apply Specialization to a single entity.
Generalization can be defined as a process of creating groupings from various entity sets	Specialization can be defined as process of creating subgrouping within an entity set
In Generalization process, what actually happens is that it takes the union of two or more lower-level entity sets to produce a higher-level entity sets.	Specialization is reverse of Generalization. Specialization is a process of taking a subset of a higher level entity set to form a lower-level entity set.
Generalization process starts with the number of entity sets and it creates high-level entity with the help of some common features.	Specialization process starts from a single entity set and it creates a different entity set by using some different features.
In Generalization, the difference and similarities between lower entities are ignored to form a higher entity.	In Specialization, a higher entity is split to form lower entities.
There is no inheritance in Generalization.	There is inheritance in Specialization.

Constraints on specialization and genr

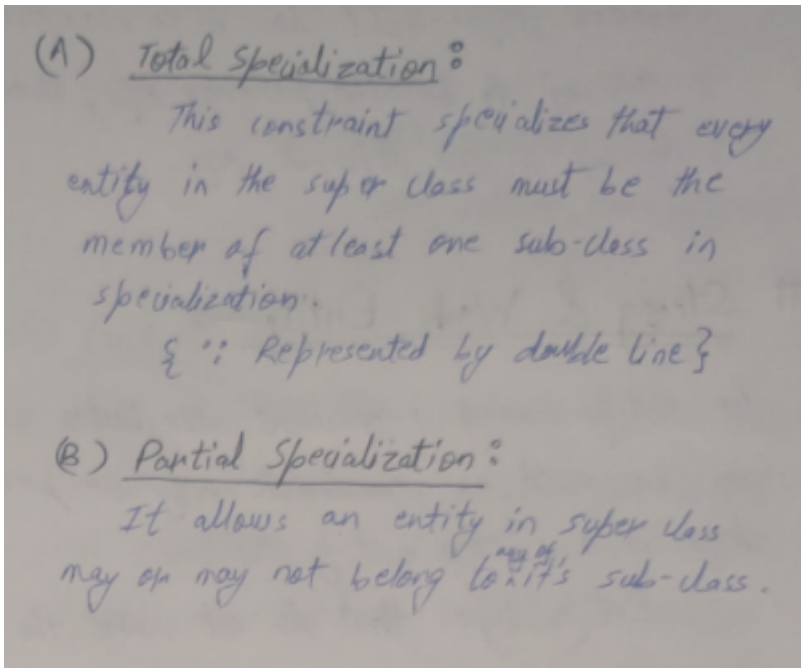
1. Condition defined OR Attr. defined
    - whether subclass / lower lvl entity satisfy EXPLICIT condition
    - 

2. User Defined
    - not constrained by membership condition
    - user defined constraint.
    - 

    - a second type of contraint... based on belonging.
    - DISJOINT AND OVERLAPPING



3. Completedness Constraint



AGGREGATION

- ER DIAG CANT EXPRESS RELATIONSHIP AMONG RELATIONSHIP
- USE AGGREGATION
- THIS is an abstraction where relationship are treated as higher lvl entity.
- can establish BINARY relation.