Unit-3 E-R Model

→ E-R model helps us to identify entities of database and relation between the entities.

→ E-R model is a Pictorial representation of Entity and relation ship.

* Overview of Design Process :-

Creation of database application is complex and it includes tasks such as:

1) Design of database schema
2) Design of programs that access & update the data.
3) Design of security scheme to control access.

* Design Phases :-

→ Database desiner must interact with users of application to under stand the needs of aplication.

→ It specifies data requirement of database users and a database structure that fulfils these requirements.

Phase-1 : Characterize data need as per user perspective.

→ database designer needs to interact extensively with domain experts and users.

outcomes: specification of users requirement.

Phase-2: Choose data model

→ Choose data model & translate these requirement into conceptual design.

→ Generally E-R model is used to represent Conceptual design.

→ It provides clear Picture of enterprise.

Phase-3: Specifications
user describes kinds of operations or transactions that will be performed on data.

Phase-4: Implementation
process of moving from an abstract data model to implementation of database contains two final design phases:
i) Logical design phase
ii) Physical design phase

1) Logical design phase:-
→ It maps conceptual schema defined using E-R model into relation schema.

2) Physical design phase:-

Using system specific database schema, it will create physical database (actual implementation).

★ The Entity-Relationship Model:-

→ E-R model is very useful to convert real world problem into a conceptual schema.

→ It employs 3 main notions:
i) Entity set ii) Relationship set iii) Attributes

i) Entity set:

An entity is a thing or object in real world that is distinguishable from all other objects.

→ An entity set is a set of entities of same type that share same properties or attributes.

example: Each customer of a bank is an Entity. A set of all customers of bank can be called as Entity set - customer.

· Entity set - customer is given below

C_id	C_name	C_street	C_city
321-12	Jones	Main	Los Angeles
619-28	Smith	North	California
677-89	Adam	Dupont	New York.

2) Attributes :- attributes are descriptive properties possessed by each member of an entity set.

→ an entity is represented by set of attributes.

→ An entity has value for each of its attribute

domain:- For each attribute there is a set of permitted values, called domain or value set of that attribute. each entity can be written as (attribute, data value) pair.

→ Thus, a database includes a collection of entity sets, each of which contains any number of entities of same type.

example:- C_id i.e. customer_id is an attribute of entity set customer, such other attributes are C_name, C_street and c_city. Now, the domain of attribute customer_name might be set of all text strings of a certain length (let's say length of 8 character). In example of entity set given above, a perticular customer may have the value 321-12 for C_id, the value 'Jones' for C_name, value 'Main' for C_street attribute and for c_city attribute value is 'Los angeles'.

3) Relationship set:-

Simple attribute:- simple attributes ar intact, they can't be divided.

Composite attribute:- they can be divided into sub parts (components).

example:- 'Age' is an simple attribute. Where as, 'Name' is an composite attribute because it can be further divided into, first name, middle name, last name. 'address' is an composite attribute, it can be divided into - street name, area name, city name etc.

Another Category for attributes:

Single valued:- If an attribute has a single value for particular entity, it is called as single valued attribute.

Multivalued:- If an attribute has more than one value for particular entity, it is called as multivalued attribute.

Example:- C_id is a single valued attribute of Entity set customer, where as phone-number of a Entity set customer can be multi-valued attribute because one customer can have multiple phone numbers.

Derived attribute:- a value of one attribute can be derived from value of other related attribute is called as derived attribute.

example:- from attribute date_of_birth of customer, we can derive his age. So, age is an derived attribute.

→ An attribute takes NULL value when an entity does not have value for it. Null value indicates either missing or not known or Not applicable. Not applicable means value does not exist for that entity.

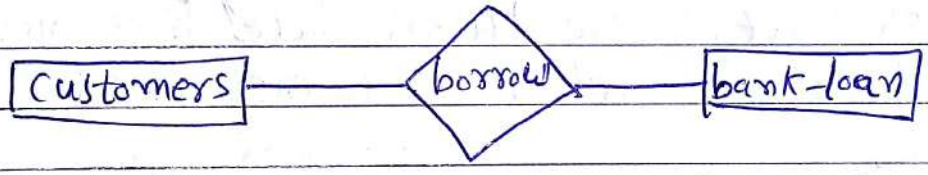
for example, if some customer does not have phone number, then contact-no attribute of that customer will have NULL value.

3) Relation ship Sets:-

It is an association among several entities.

→ A set of relationships of same type, is called as relationship.

Example: Two entity sets ⇒ customer & loan. We can define relation ship borrow to denote association between customers and bank-loans.



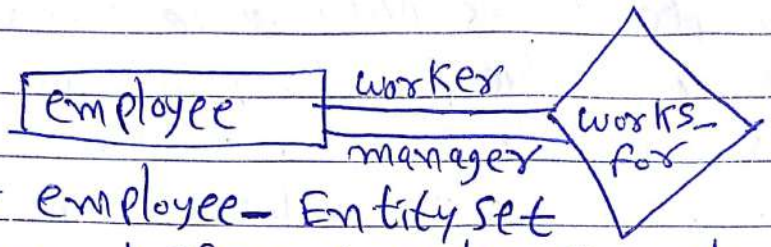
→ list of entity sets which are involved in relationship-set, is called as participation. The Entity set $E_1, E_2, \ldots, E_n$ Participate in relationship set R.

Entity's role:- It is a function that entity plays in a relationship, is called as entity's role.

→ Entity's role is useful when meaning of a relationship needs clarification.

Recursive Relationship set:- Same entity set participates in a relationship set more than once, in different role.

example:



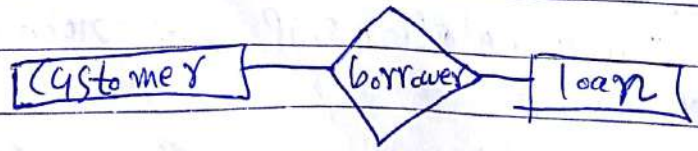
Here, for employee - Entity set there are different roles of employees - worker or manager. Here relationship set 'works-for' can be modeled by ordered pair (worker, manager) of employee entities. Here, pair (manage, worker) are excluded. These example says that some employees which are worker, works-for some employees which are manager!!

Descriptive attributes :- attributes of a relationship is called as descriptive attribute.

example:- Consider relationship set depositor with entity sets - customer and account. We can keep 'access_date' as an attribute of an relationship set; where 'access_date' is a most recent date on which a customer accessed his account.

★ Relationship instance:- One instance or value of 1st entity set is associated with ~~another~~ one instance or value of another entity set at any moment is called as relationship instance.

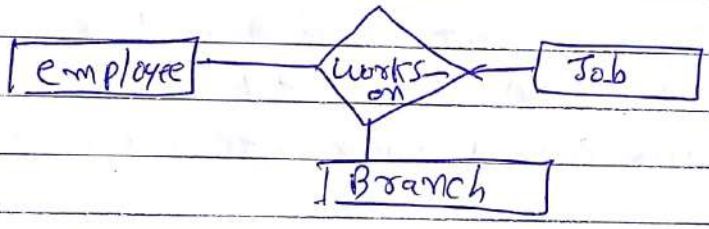
example → an individual customer entity 'Suresh' whose id is 677-89-9011 is associated with loan entity 'L-15' at any moment of time, then it is an relationship instance of borrower at that moment of time.



a binary relationship set involves 2-entity sets. Most of relationship sets in a database system are binary.

a ternary relationship set involves 3 entity sets.

Example-> entity sets employee, branch & Job are connected using relationship set works-on. Job entity set includes manager, teller, auditor etc. Job has attributes Job-title and level. Branch has attributes branch-id and branch name as well as location. now, let's say as a relationship instance - Jones acts as a manager at new york branch, Jones could also act as auditor at downtown branch, Smith works as a teller at new Jersey branch.



Degree of relationship set: No. of entity sets participate in a relationship set is called as a degree of relationship set. Binary relationship set of degree 2 and ternary relationship set is of degree 3.

* Constraints

ER enterprise schema defines certain constraints which contents of database must conform.

Two major contraints

1) mapping cardinalities 2) participation constraint

1) Mapping Cardinalities :- (cardinality ratios)

for relationship set R, mapping cardinalities between entity sets A & B is given as

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

Mapping cardinalities or cardinality ratios is the number of entities to which another entity can be associated via a relationship set.

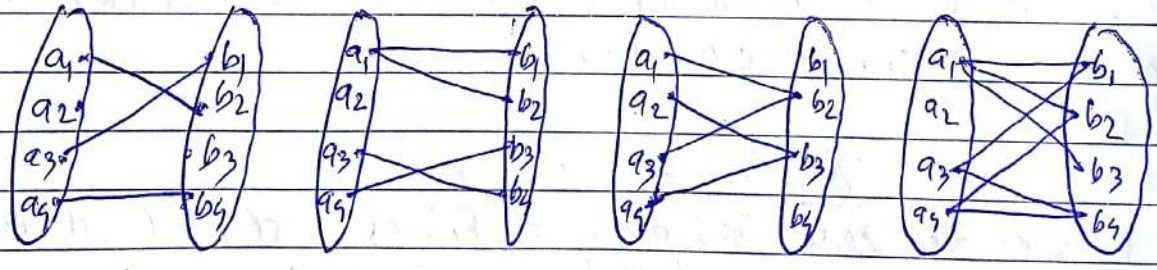
mapping cardinalities are more useful for binary relationship sets.

1) one to one :- An entity of A is associated with at most one entity in B, and an entity in B is connected with at most one entity of A.

2) one to many - An entity of A is associated with any number (0 or more) of entities of B, but entity in B is associated with at most one entity in A.

3) many to one :- An entity of A is associated with at most one entity in B, An entity in B is associated with any number (0 or more) of entities in A.

4) many to many :- An entity of A is associated with any number of entities in B and an entity of B is associated with any number of entities in A.



one to one one to many Many to one Many to Many

example → Customer & loan - Entity sets are associated using relationshipset borrower, in a particular bank one loan can be borrowed by ~~only~~ only one customer, and a customer can have several loans, thus relationship from customer to loan is one to many, if a loan can belong to several customers (a loan ~~by~~ jointly by several business partners) then relationship is many to many.

2) Participation Constraints:-

i) Total Participation :- Participation of an Entity set E in a relationship set R said to be total, if every entity in E participates in at least one relationship in R. ~~If only some entities in E participates~~

ii) Partial Participation:- If only some entities in E Participates in relationships in R, the participation of Entity set E in relationship R is said to be Partial.

example:- loan entity belongs to either one customer or more customer through borrow relationship. Hence, Participation of loan in relationship set borrower is total, but it is not necessary that every customer has taken loan, thus participation of customer in borrower relationship is partial.

* Keys :- conceptually individual entities are distinct from a database perspective, however the difference among them must be specified in terms of their attributes.

No two entities in an entity set are allowed to have exactly the same value for all attributes.

A key uniquely identifies entities of an entity set from each other.

Super-key :- It is a set of one or more attributes that, taken collectively, identifies each entity in an entity set uniquely.

example.→ customer-id attribute of entity set customer is sufficient to distinguish entity from another. So, customer_id is a superkey. Similarly customer-id + customer-name combinely work as a superkey for

see Unit-2 for foreign key.

entity set customer. customer_name alone can not work as super key because several customer might have the same name. Here, we can see that any attribute which is taken collectively with customer_id becomes super key.

Candidate-key :- Minimal superkeys are called as candidate keys.

example :- In customer relation several distinct set of attributes work as a candidate key. Suppose combination of customer_name & customer_street combinely sufficient to distinguish among members of customer entity set. So, both {customer_id} and {customer_name, customer_street} are candidate keys. {customer_id, customer_name} is also able to uniquely identify entities, but it is not candidate key, as it is not minimal because customer_id alone able to identify each Entity uniquely.

Primary key :- Out of all candidate key, one is chosen by database designer as principal means of identifying entities within an entity set.
→ Primary key should be chosen such that its attributes are changed never or very rarely. Means address field of a customer can never be an primary key because address fields are changed likely.

Primary key for relationship set :-
Let R be a relationship set involving entity sets $E_1, E_2, \ldots, E_n$. Let primary-key(E_i) denote set of attributes which forms primary key for entity set E_i.
The composition of Primary Key for a relationship set depends on set of attributes associated with relationship set R.

If relationship set R does not have its own attributes, then the set of attributes

Primary key (E_1) ∪ Primary key (E_2) ∪ ... ∪ Primary key (E_n)

describes an individual relationship in set R.

If relationship set R has attributes associated with it, then set of attributes

Primary key (E_1) ∪ Primary key (E_2) ∪ ... ∪ Primary key (E_n) ∪ $\{a_1, a_2, ..., a_m\}$

($a_1, a_2, ..., a_m$: attributes of relationship set)

describes an individual relationship in set R.

In both of above cases, set of attributes

Primary key (E_1) ∪ Primary key (E_2) ∪ ... ∪ Primary-key (E_n)

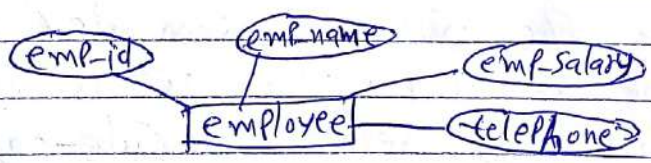
forms superkey for relationship set.

If attributes names of primary keys are not unique among entity sets, the attributes are renamed to distinguish them.

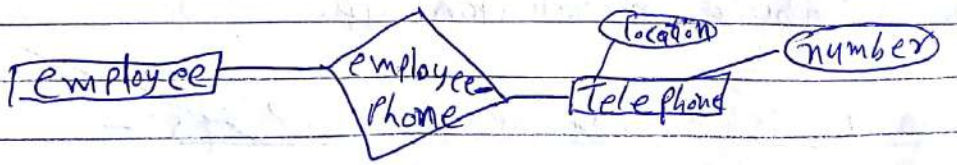
* Design Issues:-

1) Use of entity sets V/s attributes:-

Sometimes attribute of an Entity set has its own attributes or properties.



Case-1: telephone is an attribute.



Case-2 telephone is an Entity set.

There are 2 different point of views:-

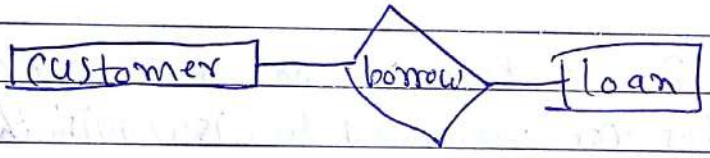
1) employee entity set with telephone attribute
2) employee entity set without telephone attribute and telephone is a new entity with location & number as attribute.

If telephone_no is an attribute of employee, that implies every employee has one telephone number precisely. Treating telephone as an entity permits employees to have several telephone number (including zero) associated with them. However, we could define telephone-number as a multivalued attribute to allow multiple telephones per employee.

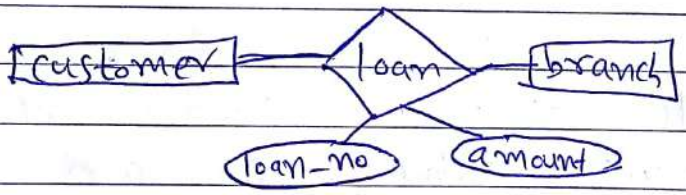
Ultimately, telephone_number should be an attribute or saperate entity set & It depends on structure of real-world enterprise being modeled, and on semantics associated with attribute in question.

2) Use of Entity Set V/s Relationship set:-

It is not always clear that it better to express an object as Entity set or as relationship set.



Case-1 loan is an Entity-set.



Case-2 loan is relationship set between customer and branch.

Whether to use an object as an entity set or relationship set, the answer is consider a relationship set to describe an action that occurs between entities.

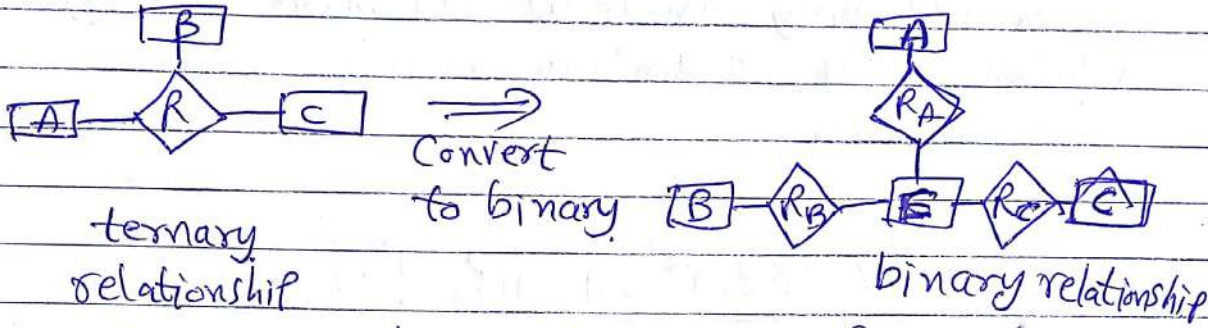
This approach is also useful to decide-object should be taken as attribute or relationship.

3) Binary V/s n-ary relationship Sets:-

Relationships in database are often binary. Non-binary relationships can be represented by several binary relationships.

Consider ternary relationship set R, relating entity sets A, B, and C. Replace relationship set R by an entity set E to convert ternary to binary, and create three relationship sets.

1) R_A - relating E & A 2) R_B - Relating E & B 3) R_C - Relating E & C. attribute of R are assigned to entity set E and one special identifying attribute is created for E.



we can generalize this process for n-ary relationship sets. Thus we can restrict E-R model to include only binary relationship sets. But this restriction is not always desirable.

4) Placement of Relationship attributes:

Mapping cardinality of a relationship can affect the placement of relationship attribute. for one to one and one to many relationships, attributes of relationship sets are associated with one of the participating entity sets, instead of relationship set.

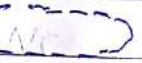
Here, relationship is one to many means one customer may have several accounts, but each account is held by at most one customer. Here, we can associate access-date attribute to account entity set. It will not change meaning.

Attributes of one to many relationship set are placed with the entity set which is one "many" side of relationship. For one to one relationship, attribute can be associated with any of participating entity sets.

placing of descriptive attribute with relationship or entity will reflect the characteristics of the enterprise being modeled. For many-to-many relationship set, attribute must be associated with relationship set.

✱ Entity-Relationship Diagram:

E-R diagram can express overall logical structure of a database graphically. E-R diagrams are simple and clear. Major Components of E-R diagram are:

object	shape	object	shape
1) entity-set		5) Multivalued attribute	
2) attributes		6) Derived attribute	
3) Relationship set		7) Total Participation -	
4) Link between attributes & Entity or Entity & relationship		8) weak entity set	

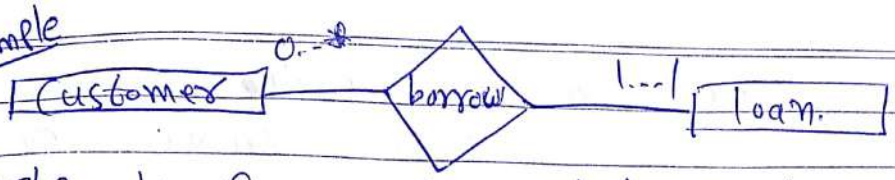
✱ Complex Constraints:

An edge between entity set and a binary relationship set can have minimum & maximum cardinality, shown in the form of l..h where l is the minimum and h is maximum Cardinality. A minimum value of 1 indicates total participation of entity set in relationship set. A maximum value of 1 indicates that entity participates in at most one relationship, maximum value * indicates no limit.

at most 1 i.e. 0...1	exactly once i.e. 1...1
at least 1 i.e. 1...*	

Complex constraints shows no. of times each entity participates in relationships in a relationship set.

example



→ The edge from customer to borrow has cardinality of 0..* indicates that a customer can have zero or more loans. Thus, relationship from customer to loan is one to many.

The edge from loan to borrow has cardinality of 1..1, that means each loan must have exactly one associated customer. And participation of loan in borrow is total.

If both edges from binary relationship have a maximum value of 1, the relationship is one to one.

>>> Weak Entity Set :-

An entity set may not have sufficient attributes to form a primary key. Such an entity set is termed as weak entity set.

For a weak entity set to be meaningful, it must be associated with another entity set called identifying or ower or strong entity set. weak entity set is existance dependent on strong entity set. The relationship between weak entity set and identifying entity set is called as identifying relationship

→ The identifying relationship from strong entity set to weak entity set is one to many. Participation of weak entity set in the relationship is total.

Discriminator (Partial key) :- It is an attribute of weak entity set that can uniquely identify entities that are related to same owner entity. It is also called as partial key of that entity set.

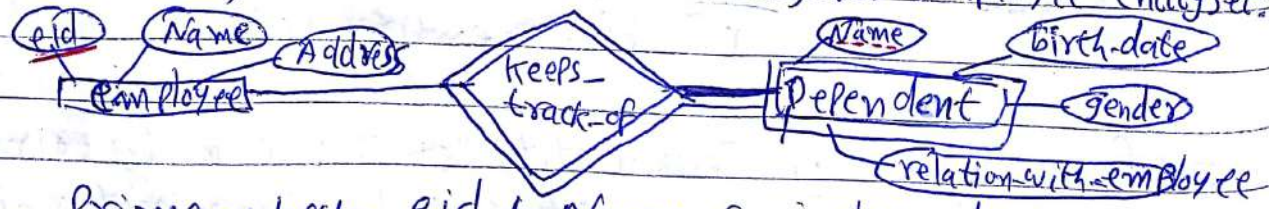
→ Primary key of weak entity set is formed by primary key of identifying entity set + discriminator of weak entity set.

A weak entityset can have more than one identifying entity set. That time, primary key of weak entity set is union of primary keys of all identifying entity sets and discriminator of weak entity set.

In E-R diagram, a doubly rectangle box is a weak entity set and doubly diamond box indicates corresponding identifying relationship.

In some cases, database designer may choose to express a weak entity set as a multivalued composite attribute of owner entity set. It is appropriate if weak entity set participates in only one relationship which is identifying relationship and it has few attributes. Weak entity set representation is more preferable when set participates in relationships other than identifying relationship and weak entity set has several attributes.

example :- Consider an entity set 'Dependent' related to 'Employee', which is used to keep track of dependents of each employee by one to many cardinality. attributes of dependent are Name, birth-date, Gender, relation-with-employee. Two dependents of two distinct employees may have same values for Name, birth-date, Gender & relation-with-employee. Each entity can be identified as distinct entity only after determining particular employee entity related to each dependent entity. Here, 'Name' is partial key and eid is primary key in employee entityset.



Primary key = eid + Name from dependent.

* Extended E-R features:-

Basic E-R model can represent most of features of database, but to express some aspects of database we need extension of E-R model. Features of extended E-R model are specialization, generalization, higher & lower level entity sets, attribute inheritance and aggregation.

1) Specialization:

→ An entity set may include subgroupings of entities that are distinct in some way from other entities in the set. Process of creating subgrouping with in an entity is called as specialization.

→ A subset of entities within an entity set may have some attributes that are not possessed by all the entities in the entity set.

→ Consider entity set person with attribute name, street and city. A person may be further classified as
- customer
- employee.

→ Each of above person types is described by a set of attributes that includes all the attributes of entity set person plus possibly additional attributes. For example, person has attributes P-id, Pname, address and contact no. Now, one ~~ets~~ Person type - customer entity can be further described by attribute - credit-rating and another person type - employee entity can be described with attribute - salary. The specialization of person allows us to distinguish among persons according to whether they are employees or customers.

→ Another example of specialization is: Animal ~~entity~~ set can be divided further in birds & mammals. Birds have some special attribute like no. of wings and mammals

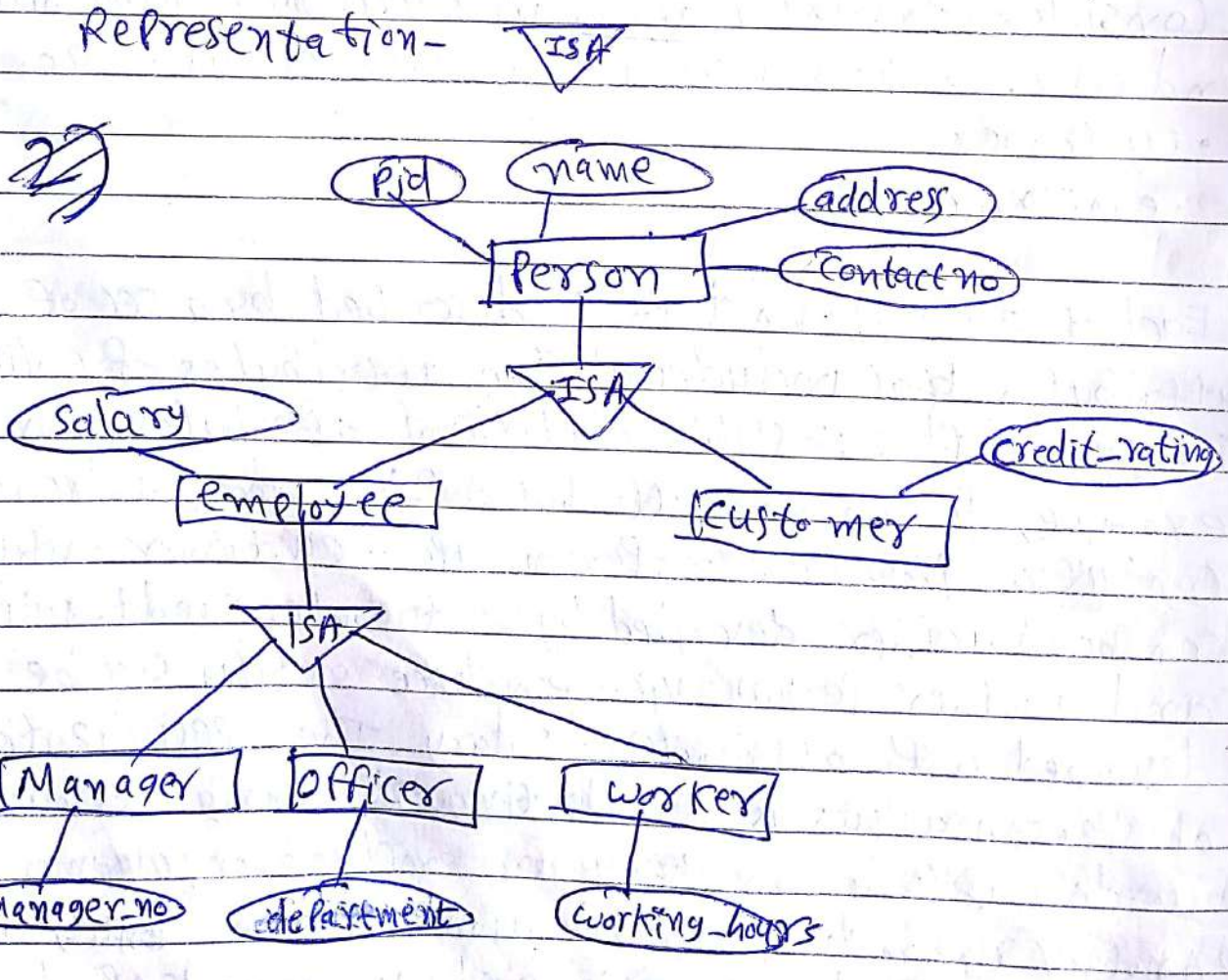
have special attribute like no. of limbs (common name for hand & leg).

→ We can apply specialization repeatedly to refine a design scheme. We had subgrouped person into employee and customer. Now, employee can further be divided as - Manager, officer and worker. These new entity sets will have their specialized attributes like - Manager_no, department or worker_id.

→ Specialization is depicted by a triangle component labeled ISA. The label ISA stands for "is a". for example, customer is a person.

→ ISA relationship may also be referred to as a superclass - subclass relationship. It is also refferred as higher level and lower level entity sets.

Representation -



2) Generalization:-

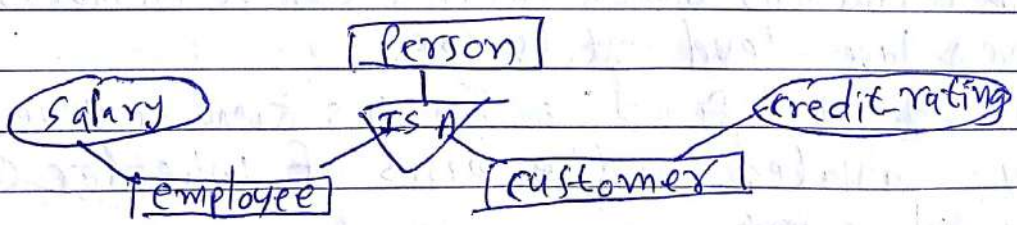
→ Specialization is an top-down approach because an entity is divided into subgrouppings.

→ Generalization is an bottom-up process. Thus, generalization is reverse process of specialization.

→ In it, Multiple entity sets are synthesized or merged into a higher-level Entity set on basis of common features.

For example → Designer first identifies customer entity set with attributes cid, name, address and Contact_no & credit_rating. Later on, employee entity set with attributes eid, name, address, Contact_no & salary. In customer & employee have most of attributes are common except one. This commonality can be expressed by Generalization. By taking common attributes, we can generalize customer & employee entity sets into a higher level or super class entity set—person. Customer & employee are lower-level entity set or subclass.

→ Generalization shows that a number of entity sets share some common features. It also helps to remove repeation of shared attributes.



3) Attribute Inheritance :-

→ In specialization or generalization, attributes of higher level entities are inherited by lower level entity sets. This is called as attribute inheritance. For example, customer and employee inherit the

attributes of person entity set.

→ lower level entity set also inherit participation in the relationship sets in which its higher-level entity participates.

→ Attribute inheritance applies to all the levels of lower-level entity sets. Similarly, participation constraint is also inherited to all the lower level entity sets.

→ An entity set may have single inheritance or multiple inheritance. If one entity set is a lower-level entity set in more than one ISA relationship, then entity has multiple inheritance and resulting structure is called as lattice.

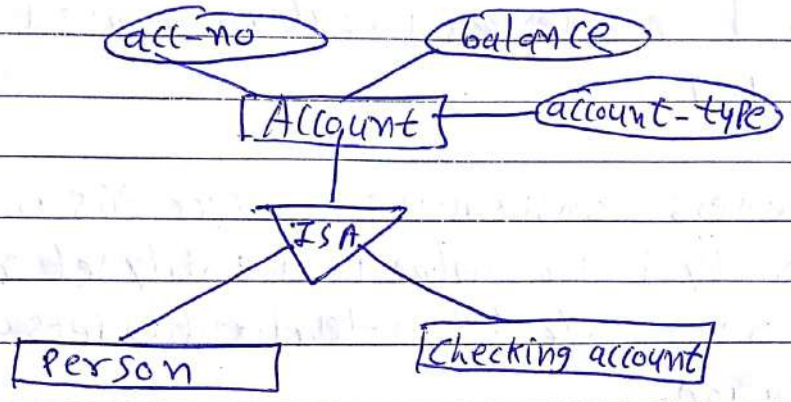
4) Constraints on Generalizations :-

→ Database designer may choose which kind of constraints should be placed on particular generalization
→ Constraints on generalization can be divided into 3 categories.

1) Determining which entities can be members of a given lower-level entity set.
   A) Condition defined :- In this kind of generalization entity evaluated on the basis of whether condition is satisfied or not.

For example:- higher level entity set has attribute account-type. All account entities are evaluated on the defining attribute account-type. Only those entities that satisfy the condition account-type = "Savings account" are allowed to belong to lower-level

entity person. All other entities that satisfies the condition account-type = "checking account" are included in checking account entity set. Here, all entities are evaluated on basis of same attribute, this type of generalization is also called as attribute-defined.



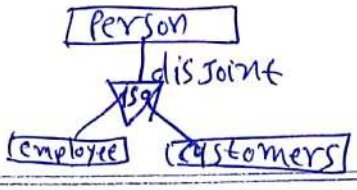
B) User-defined:- Instead of checking any condition, database user assigns entities to a given entity set.

for example:- After some training, coach assigns some specific position to a football player in a team. Here, football player is an higher level entity set and mid fielder, attacker and defender are lower level entity sets. Here, candidate is not assigned to any lower level entity set automatically, but coach makes decision on individual basis from their performance in training.

2) Whether or not entities may belong to more than one lower-level entity set within a single generalization.

A) Disjoint: A disjoint constraint says that an entity belongs not more than one lower level entity set. forexample:- In previous example, an account entity can satisfy only one condition for account_type attribute, an entity can be either a savings account or checking account, but not both.

Representation:



B) overlapping :- In it same entity may belong to more than one lower-level entity set within a single generalization. For example, Consider higher level entity set Person and its lower level entity sets are employee and customer. It is possible that some person is employee as well as customer too. This kind of generalization is called as overlapping.

3) Completeness Constraint:- It specifies whether or not an entity in the higher level entity set must belong to atleast one of the lower-level entity sets within generalization.

A) Total Generalization :- Each higher level entity must belong to one or more lower level entity set.

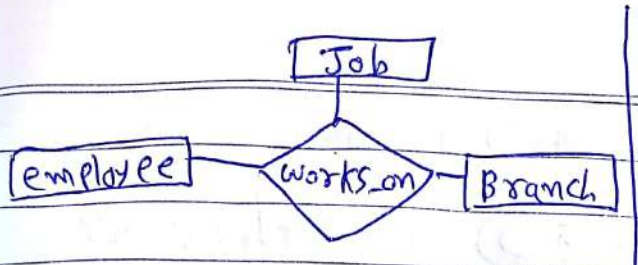
B) Partial Generalization:- Some higher level entities may not belong to any lower level entity set.

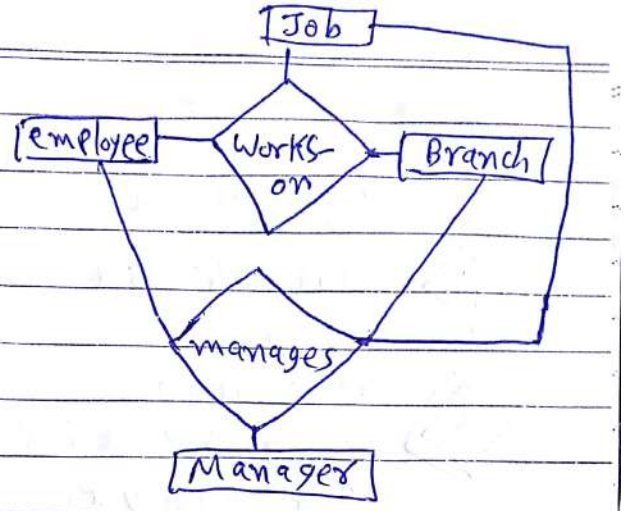
example → Account Participation is total, means all entities must be either a savings account or a checking account. But in Person entity set, It is possible that person may not be a customer or an employee. It is called as Partial generalization.

Aggregation:-

Limitation of ER model is that it can not represent relationship among relationships. We can overcome this limitation using aggregation.

Example:- Consider ternary relationship works_on between employee, branch and job. Now, we want managers to keep track of tasks performed by an employee at each branch for a particular job.

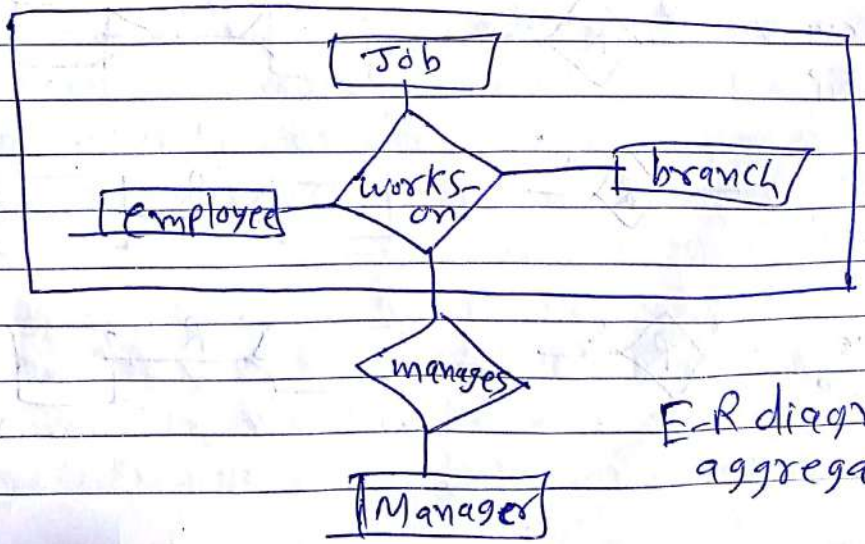




One alternative for representing this relationship is to create a quaternary relationship 'manages' between employee, branch, job and manager.

Another option is to combine 'works_on' and 'manages' relationship sets into single relationship set. But we should not do that because some employee, branch, job combinations may not have a manager.

The best way to model this situation is to use aggregation. Aggregation is an abstraction through which relationships are treated as higher level entities. In this way, we consider 'work_on' relationship set as a higher-level entityset called as 'Work_on'. This kind of entity set is treated in same manner like other entity set. We can create binary relationship between works_on & manager to represent who manages what tasks.



E-R diagram with aggregation.

entity set

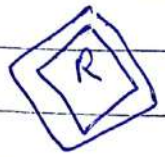
attribute

weak entity set

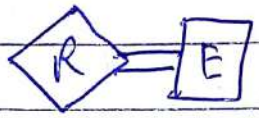
multivalued attribute

relationship set

derived attribute

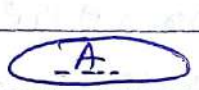


Identifying relationship set for weak entity set

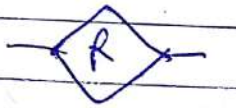


Total Participation of entity set in relationship

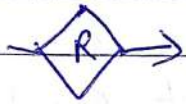
Primary key



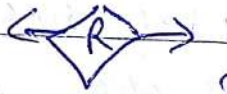
discriminating attribute of weak entity set



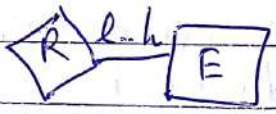
many-to-many relationship



many-to-one relationship



one-to-one relationship



cardinality limits

role indicator

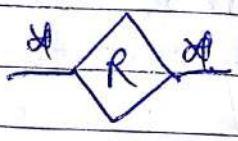
ISA relationship (Specialization or generalization)

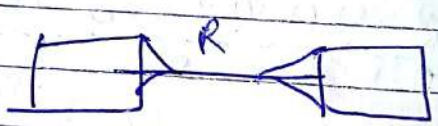
Total generalization

disjoint generalization

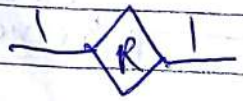
Entity set E with attributes A1, A2, A3 where Primary key A1

many-to-many relationship





one-to-one relationship



many-to-one relationship

* Reduction of E-R schema to Relational schema:-

For each entity set and for each relationship set, there is a unique table. We assign name of corresponding entity set or relationship set, to that table.

1) Representation of strong entity set:

Let E be a strong entity set with attributes $a_1, a_2, \ldots, a_n$. We represent this entity by schema called E with n distinct attributes.

example ⇒ Consider loan entity set with attributes: loan-number and amount. We represent this entity set by a schema called Loan-schema with these 2 attribute.

Loan-schema (loan-number, amount)

loan number is primary key of loan entity set, it is also the primary key of schema.

2) Representation of Weak Entity set:

Weak entity set depends on strong entity set. Primary key of schema of weak entity set is Union of primary key of strong entity set and discriminator of weak entity set.

ex: Weak entity set Payment has 3 attributes: payment-number, payment-date, and payment-amount. strong entity set for payment entity set is loan. Primary key of loan entity set is loan_number. Hence, schema for given as Payment is given as:

Payment_schema (loan-number, payment-number, payment-date, payment-amount)

where Primary key is Union of loan_number & payment_number.

3) Representation of Relationship set:

Let R be a relationship set with descriptive attributes $a_1, a_2, \ldots a_m$. Primary keys of each entity sets participating in R, are $b_1, b_2, \ldots b_n$. Schema of R has the attributes given as:

$$\{a_1, a_2, a_3 \ldots a_m\} \cup \{b_1, b_2, \ldots, b_n\}$$

where Primary key of schema of R is $\{b_1, b_2, \ldots b_n\}$

example:- customer entity set having Primary key cid and loan entity set having Primary key loan_no are associated with relationship borrower. Schema for borrower is given as:

Borrower-Schema (cid, loannumber)

Here, borrower has no descriptive attribute. Primary key for borrower is (cid, loannumber).

4) Representation of Composite attribute:-

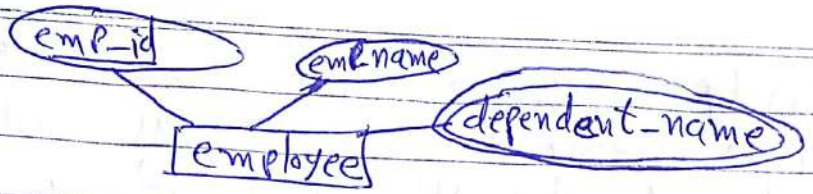
We create separate attribute for each component of composite attribute, We do not create a column for composite attribute itself.

example⇒ Address is a composite attribute of entity set Customer. Components of address are street, area, city, state and pincode. The schema generated from Customer would then contain columns address-street, address-area, address-city, address-state and address-pincode. There is no separate column for address.

5) Representation of Multivaled Attribute:-

Generally, one attribute of entity set becomes attribute of schema of that entity set too. But for multivalued attribute, new table (relation) and schema are created.

example.



employee entity set has Primary key attribute emp-id and multivalued attribute dependent_name. we create a schema dependent_name which contains column dname (means dependent name) referring to attribute of employee entity set as well as emp-id (Primary key of employee entity set). Each dependent of employee is represented as Unique row in the table (Relation).

schema ⇒ dependent_name (emp-id, dname)

6) Representation of Generalization :-
There are two methods to convert E-R diagram having generalization to schema.

A) create schema for higher level entity set. For each lower level entity set, create a schema that includes a column for each attribute of that entity set plus one column for each attribute of Primary key of higher level entity set.
Example ⇒ entity set - person has attributes personid, Pname, address and city where personid is a Primary key. Lower level entity sets are employee having additional attribute salary and customer having additional attribute credit-rating. Schema created for Person, employee and customer is given as:
Person = (personid, Pname, address, city)
employee = (personid, salary)
customer = (personid, credit_rating)

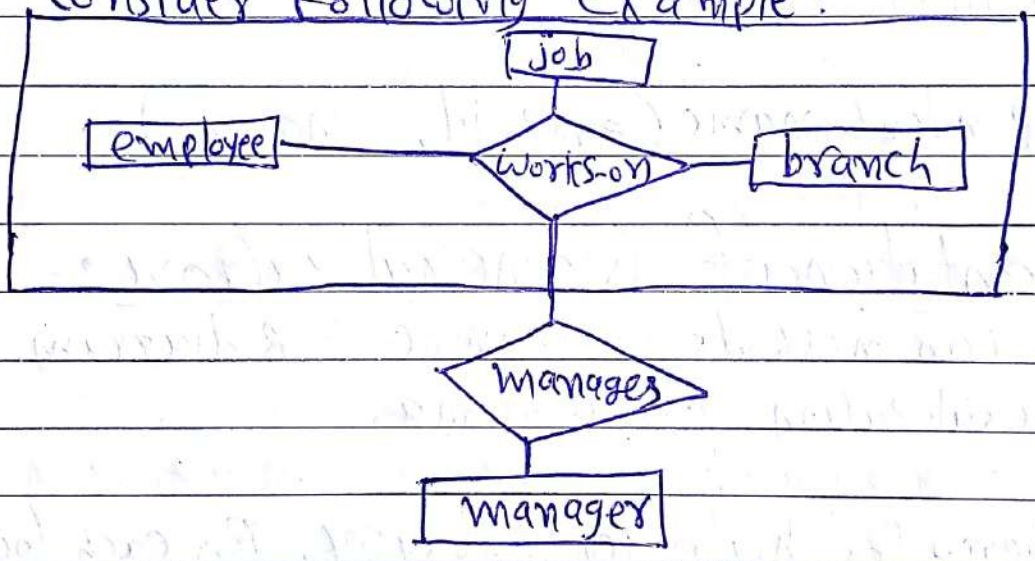
B) If the generalization is disjoint and complete (total participation). Here, do not create schema for higher-level

entity set. Instead foreach lower-level entity set, create schema that includes attributes of that entity set plus all attribute of high level entity set. Hence, schema for lower level entity sets are given as:

employee = (personid, fname, address, city, salary)
customer = (personid, fname, address, city, credit_rating)

7) Representation of Aggregation:

Consider following example:



Here, 'manages' relationship set includes the attributes of primary keys of entity set manager and relationship set works-on. 'manages' also may have any descriptive attributes. The schema for relationship set 'manages' contains columns of all the attributes of relationship set 'manages'.