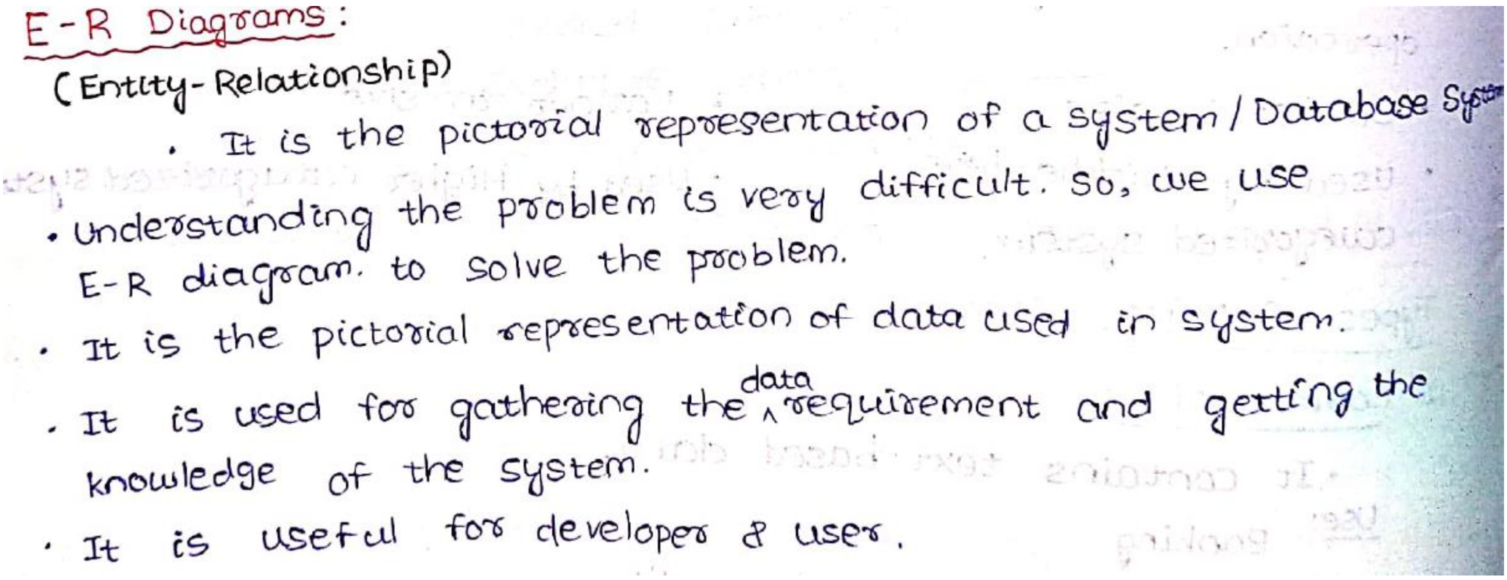


E-R Diagrams:

(Entity-Relationship)

- It is the pictorial representation of a system / Database System
- understanding the problem is very difficult. So, we use E-R diagram. to solve the problem.
- It is the pictorial representation of data used in system.
- It is used for gathering the data requirement and getting the knowledge of the system.
- It is useful for developer & user.

- ER modelling helps you to analyse data requirements systematically to produce a well-designed database. So, it is considered a best practice to complete ER modelling before implementing your database.

- It was proposed by Peter Chen in 1971 to create a uniform convention which can be used for relational database and network. He aimed to use an ER model as a conceptual modelling approach.

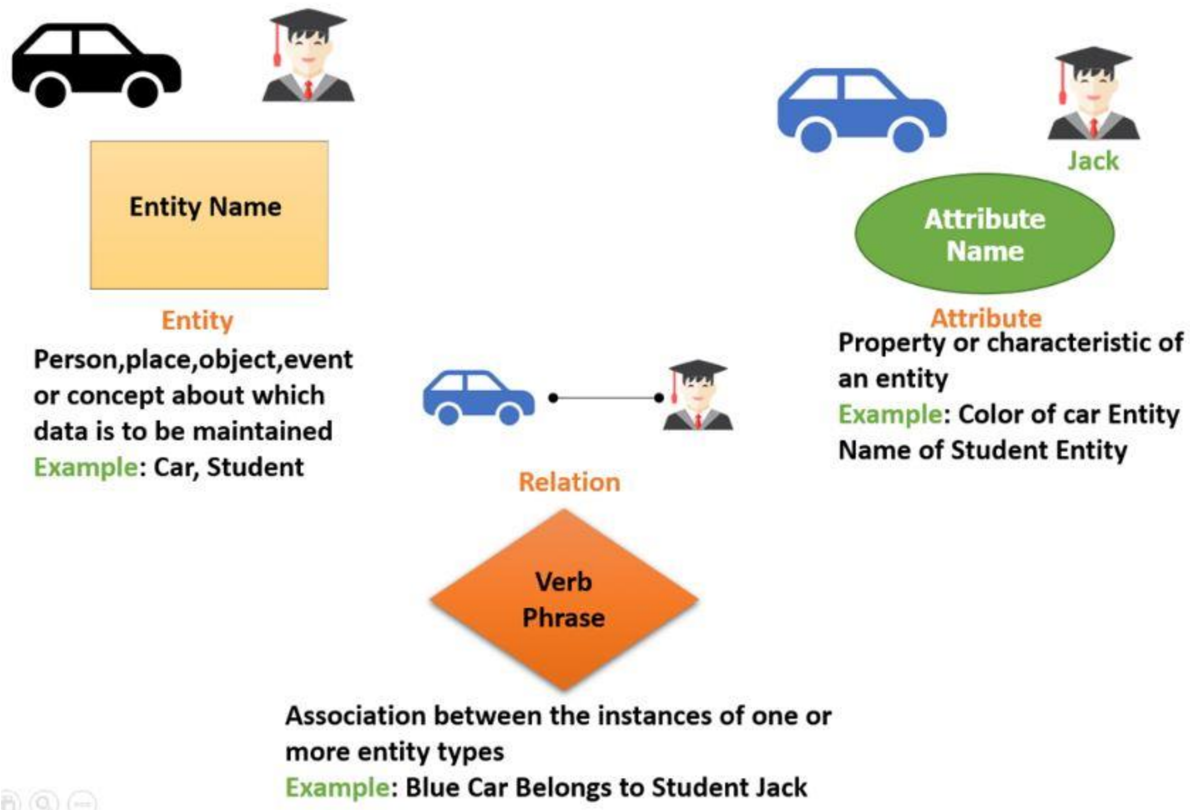
Why use ER Diagrams?

Here, are prime reasons for using the ER Diagram

1. Helps you to define terms related to entity relationship modeling
2. Provide a preview of how all your tables should connect, what fields are going to be on each table
3. Helps to describe entities, attributes, relationships
4. ER diagrams are translatable into relational tables which allows you to build databases quickly
5. ER diagrams can be used by database designers as a blueprint for implementing data in specific software applications
6. The database designer gains a better understanding of the information to be contained in the database with the help of ERP diagram
7. ERD is allowed you to communicate with the logical structure of the database to users

ER model is based on three basic concepts:

- Entities
- Attributes
- Relationships



Entity

Person,place,object,event or concept about which data is to be maintained

Example: Car, Student

Attribute

Property or characteristic of an entity

Example: Color of car Entity Name of Student Entity

Relation

Association between the instances of one or more entity types

Example: Blue Car Belongs to Student Jack

Entity

- Anything which is uniquely identified. (passive?
- It is passive.

Object

- It is active.
- It is a real-time entity

Entity	Object
1. Data(properities)	1. Properties 2. Functions
used in RDBMS.	used in ODBMS.

Presently, we are using ODBMS

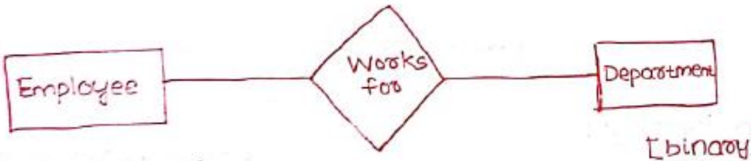
Entity < Physical {tangible - which actually existing}
Logical / Conceptual (intangible)

Ex. of Entity
Item, Product, Car, House, Employees, Person information.
↑ [tangible entity]

Logical entity is intangible.
Ex: Transaction, order of a product, account, load (can't be seen/touched)

Physical entity is tangible
Ex: Employees, Employer

Relationship between Entities:

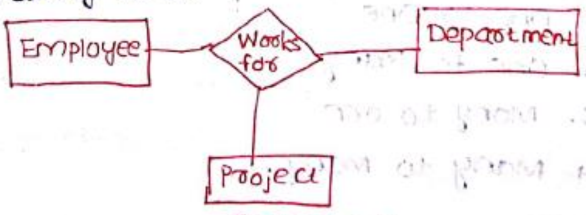


[binary]

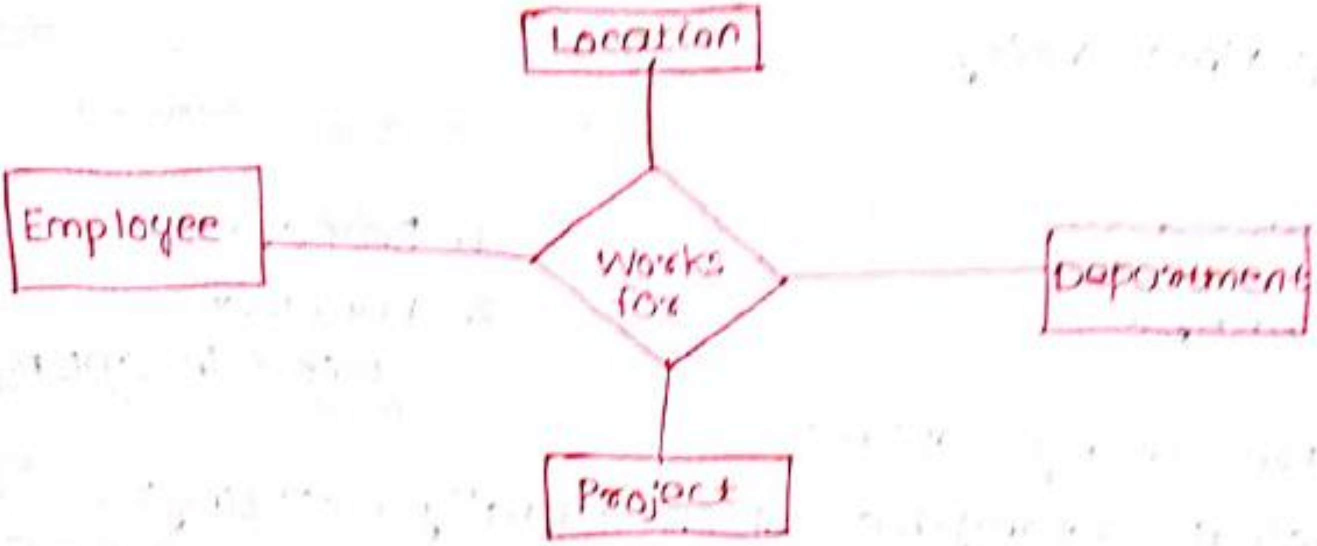
Relationship Diamond

Entity Rectangle

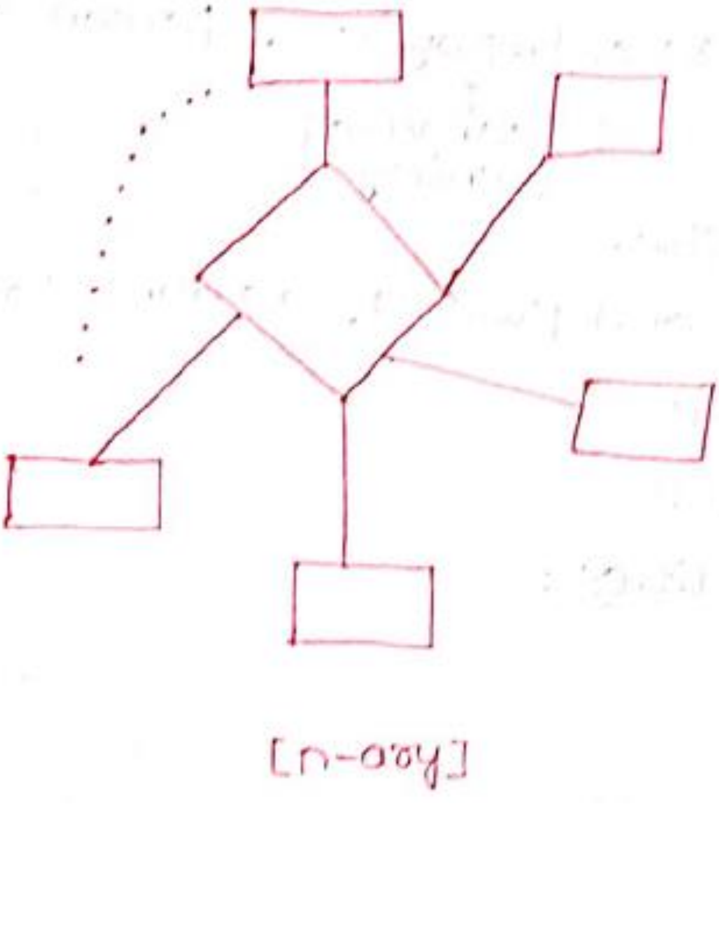
Relationship is the association between 2 or more entity.
It is of following types:
TYPES OF Relationship:
1. Binary : No. of entities participating is 2.
2. Ternary
3. Quadainary
4. n-ary



[Ternary]



[Quadarnary]



[n-ary]

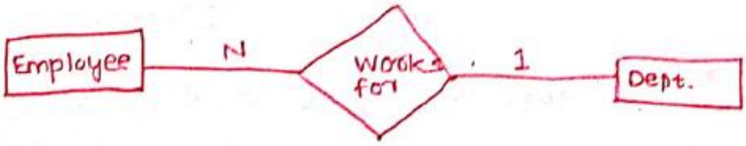
IMP Degree of Relationship:

No. of entity participated in a relationship is called Degree of Relationship. (ary of relationship)

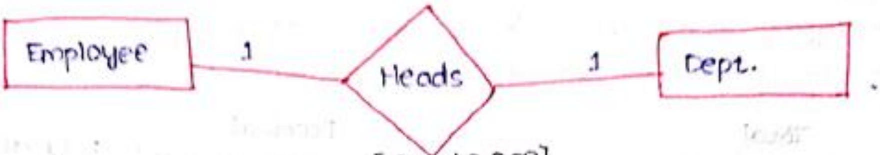
Mapping Cardinalities:

cardinalities: number

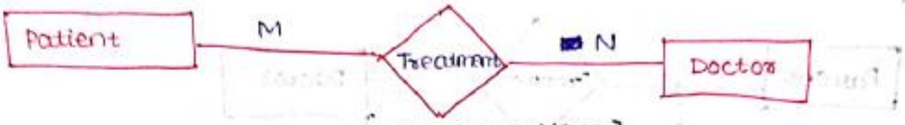
1. One to One
2. One to Many
3. Many to one
4. Many to Many



[Many to one]

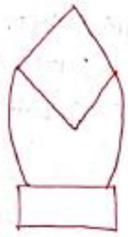


[one to one]

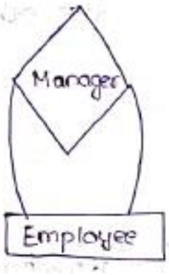


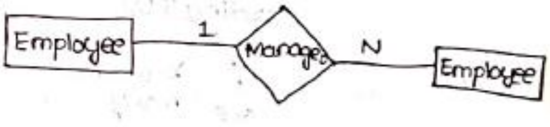
[Many to Many]

IMP • Same entity is participating in a Relationship: Recursive Relationship



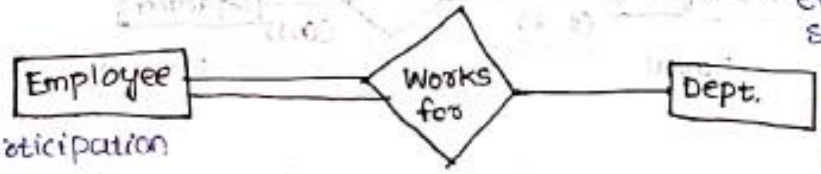
Example:





Participation constraint:

For a set of ass entity is associated with other set of entity.



Each Employee should belong to a dept. ∴ total participation

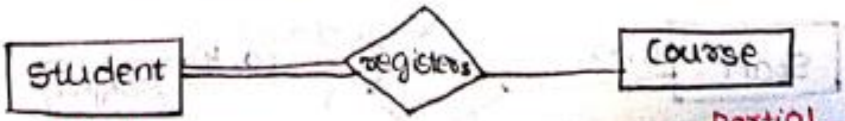
every dept. will have some employee. Now, Suppose no emp. is there in a dept & borrowed from other dept ∴ Partial

Participation is the involvement of entities in the relationship.

There are two types of participation

(i) Total participation

(ii) Partial "

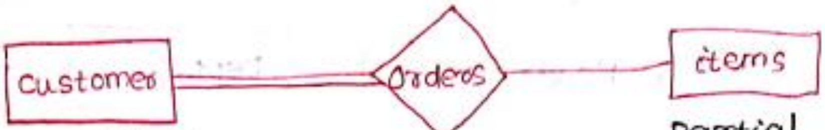


Total participation

Partial participation

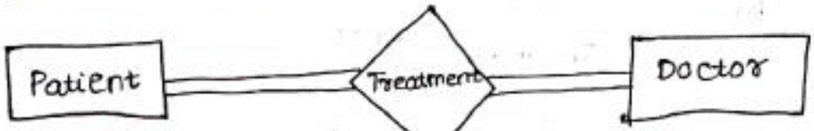
He is a student if he is in a course.

Course without student may exist.



Total
only when buys item

Partial
customer won't buy all items.



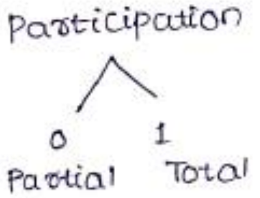
Total

Total

Cardinality:

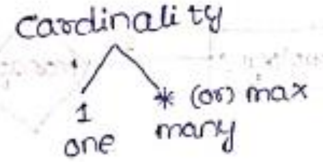
There are 2 constraints in a relationship.

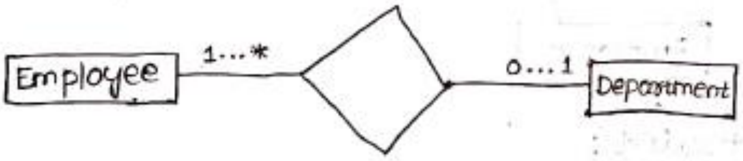
1. Participation constraint
2. Cardinality



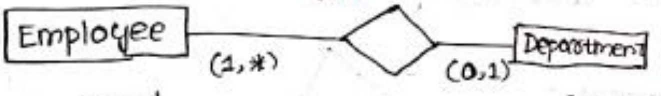
Full Participation:

Each of the entity containg pertains to atleast one of the other related entity.



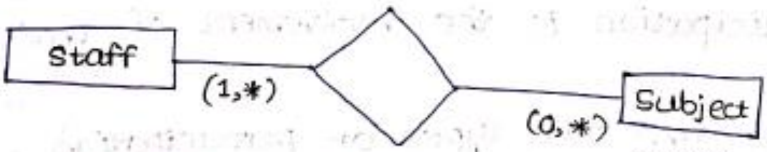


(or)



Total

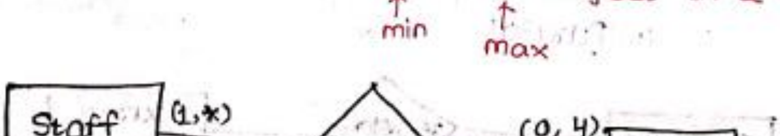
Partial



Partial

A staff can take 2 (or) 4 subject in a college.

↑ min ↑ max



1 - Total

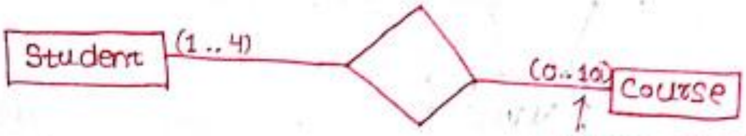
0 - Partial

only max. value can be represented by the E-R diagram.

Object

Q. Each student can register for at most 4 courses and each course can have at most 10 students. Draw the E-R diagram.

A:



10 times the course is appear.

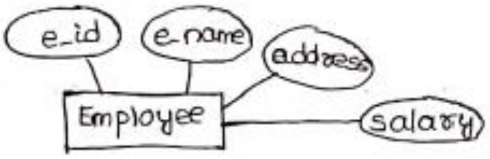
Attributes:

- It defines the property of entity.

Example:

Attribute

Ellipse



- Every attribute has its domain (set of valid values)
- Domain means set of valid value.

Salary: Domain (amoun

Name : Domain (Human Names)

In a pop-up menu, there are no. of choices (Suppose country: India, Pakistan, ...). It is called Domain.

Each country name is unique. So, it is the attribute. It is given to avoid mistakes (or) ambiguities.

- Domain is the constraint on attribute value.

Name: only alphabets

Salary : only numbers (not characters)

Types of attributes:

1. Simple & Composite

↓

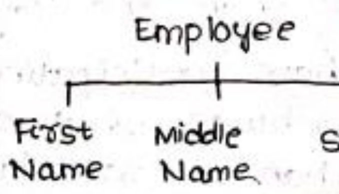
value of variable cann't be divided again.

Ex: emp-id, basic salary

2. Multi valued
3. Stored (or) Derived

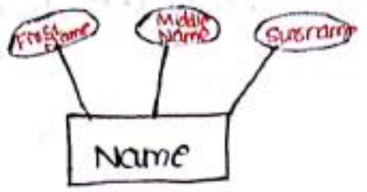
Composite:

Attribute may have collection of values. i.e. divided.



← Composite attribute.

- Address



Name may be an attribute.

The composite attribute can also be converted into entities, but it depends on designer (or) developer.

Multivalued Attribute:

Example:

Phone No.

An employee can have more than one phone No.

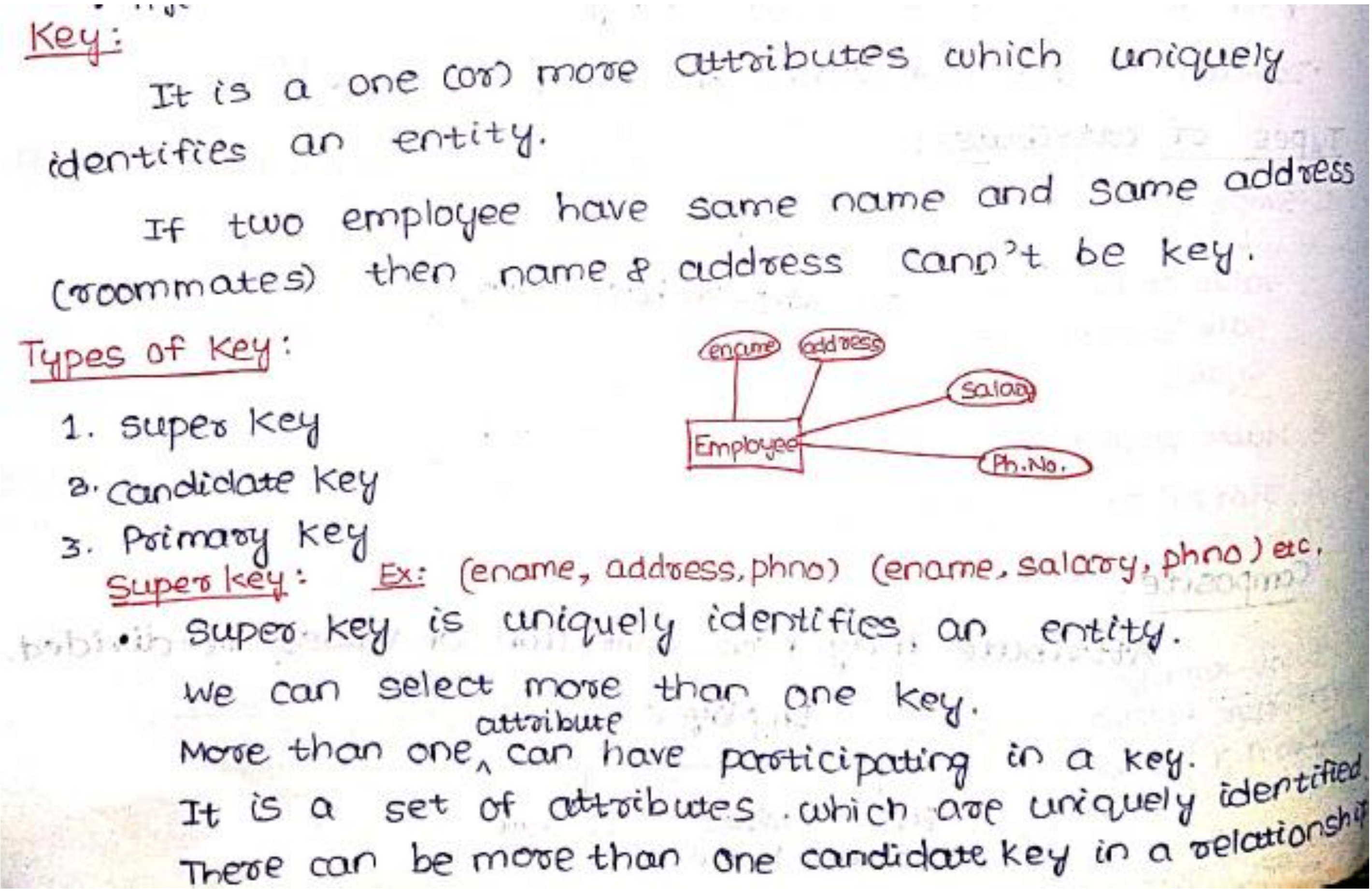
Stored (or) Derived Attribute:

- Age is a processed data. It varies day to day.
- Net salary is a derived one.
- In Derive attribute, the value can obtained by processing. (or) computing.
- D-O-B is a stored attribute. It is n't varying.
- Age is a derived attribute.

Why we need a Key?

Here, are reasons for using Keys in the DBMS system.

- Keys help you to identify any row of data in a table. In a real-world application, a table could contain thousands of records. Moreover, the records could be duplicated. Keys ensure that you can uniquely identify a table record despite these challenges.
- Allows you to establish a relationship between and identify the relation between tables
- Help you to enforce identity and integrity in the relationship.



Key:

It is a one (or) more attributes which uniquely identifies an entity.

If two employee have same name and same address (roommates) then name & address cann't be key.

Types of Key:

1. super key
2. candidate key
3. Primary key

ename
address
Salary
Employee
Ph.No.

Super key: Ex: (ename, address, phno) (ename, salary, phno) etc.

- Super key is uniquely identifies an entity.

We can select more than one key.

More than one attribute can have participating in a key.

It is a set of attributes which are uniquely identified

There can be more than one candidate key in a relationship

Candidate key:

- If we remove the unnecessary attribute from super key then it is a candidate key.
- It is a subset of super key whose subset cann't be a key again.
- It is a minimal subset of super key which is also a key and candidate keys subset is n't a key.

Example:

ename & address

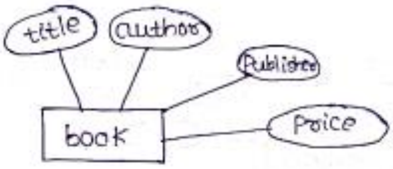
ename & Ph. No.

- Entity may have more than one candidate keys

$\boxed{\text{candidate key} \subset \text{Super key}}$

Primary key:

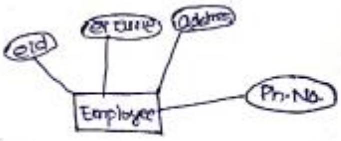
- Primary key is a candidate key but it is selected by the designer.



super key : (title, author, publisher)

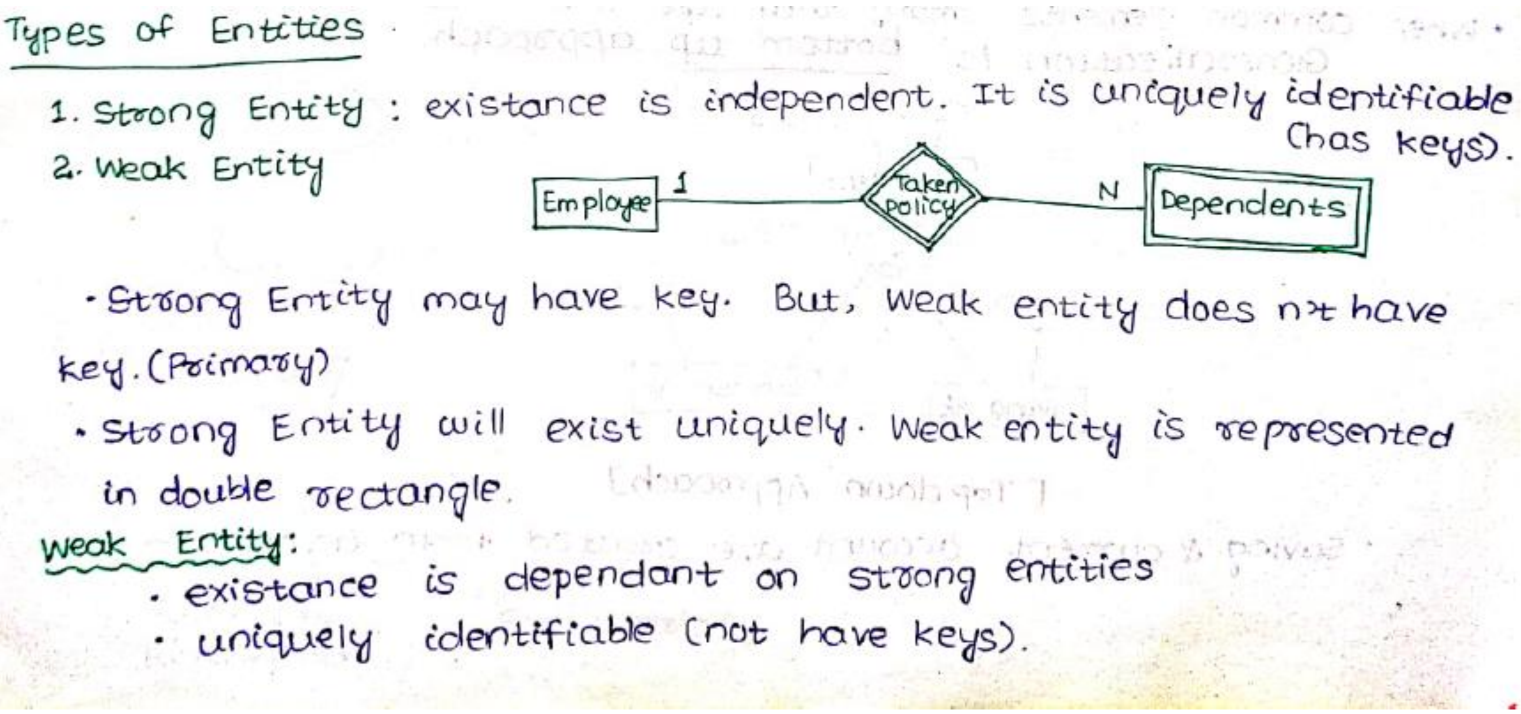
candidate key : (title, author)

Primary key :



Primary key : eid

- The value of primary key should n't be updated.



Types of Entities

1. Strong Entity : existance is independent. It is uniquely identifiable (has keys).
2. Weak Entity

Employee
1
Taken Policy
N
Dependents

- Strong Entity may have key. But, weak entity does not have key. (Primary)
- Strong Entity will exist uniquely. Weak entity is represented in double rectangle.

Weak Entity:

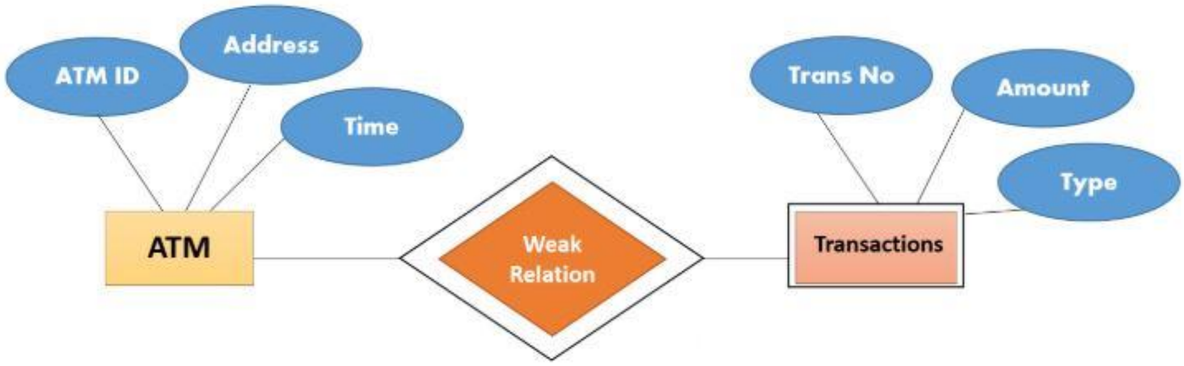
- existance is dependant on strong entities
- uniquely identifiable (not have keys).

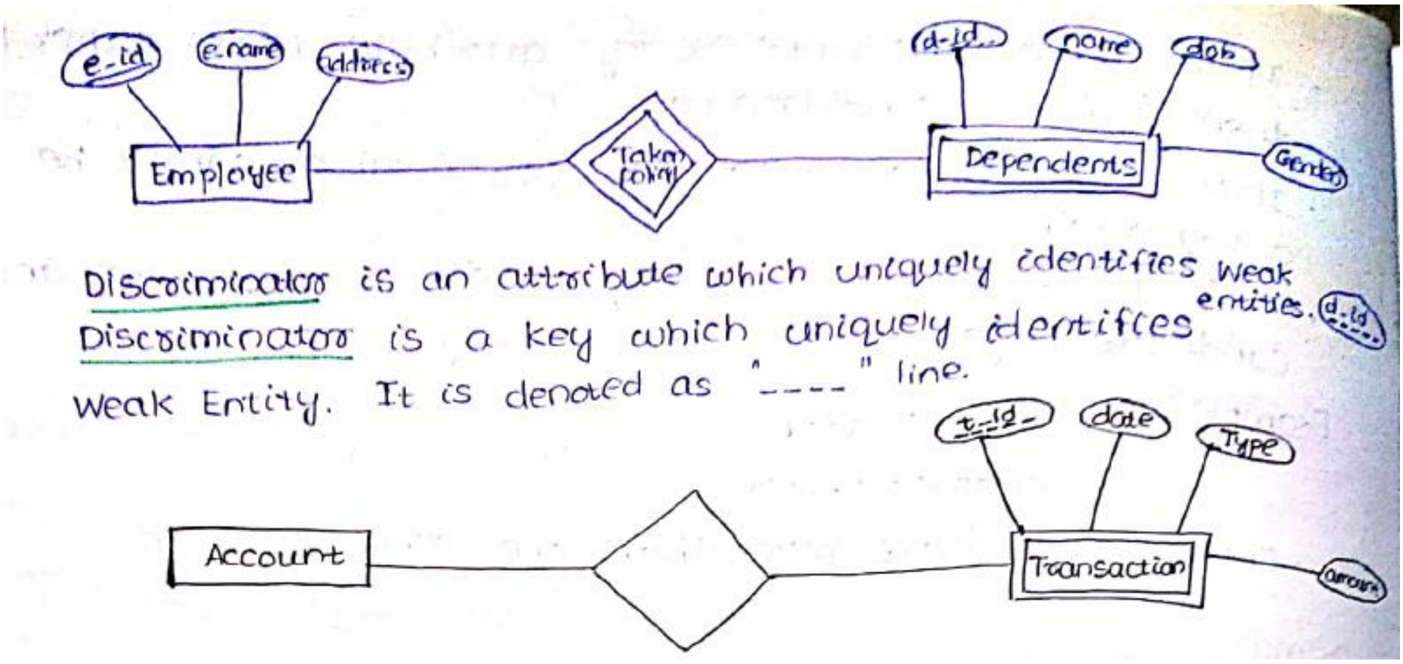
Strong Entity Set	**Weak Entity Set**
Strong entity set always has a primary key.	It does not have enough attributes to build a primary key.
It is represented by a rectangle symbol.	It is represented by a double rectangle symbol.
It contains a Primary key represented by the underline symbol.	It contains a Partial Key which is represented by a dashed underline symbol.
The member of a strong entity set is called as dominant entity set.	The member of a weak entity set called as a subordinate entity set.
Primary Key is one of its attributes which helps to identify its member.	In a weak entity set, it is a combination of primary key and partial key of the strong entity set.
In the ER diagram the relationship between two strong entity set shown by using a diamond symbol.	The relationship between one strong and a weak entity set shown by using the double diamond symbol.

The connecting line of the strong entity set with the relationship is single.	The line connecting the weak entity set for identifying relationship is double.

A weak entity is a type of entity which doesn't have its key attribute. It can be identified uniquely by considering the primary key of another entity. For that, weak entity sets need to have participation.

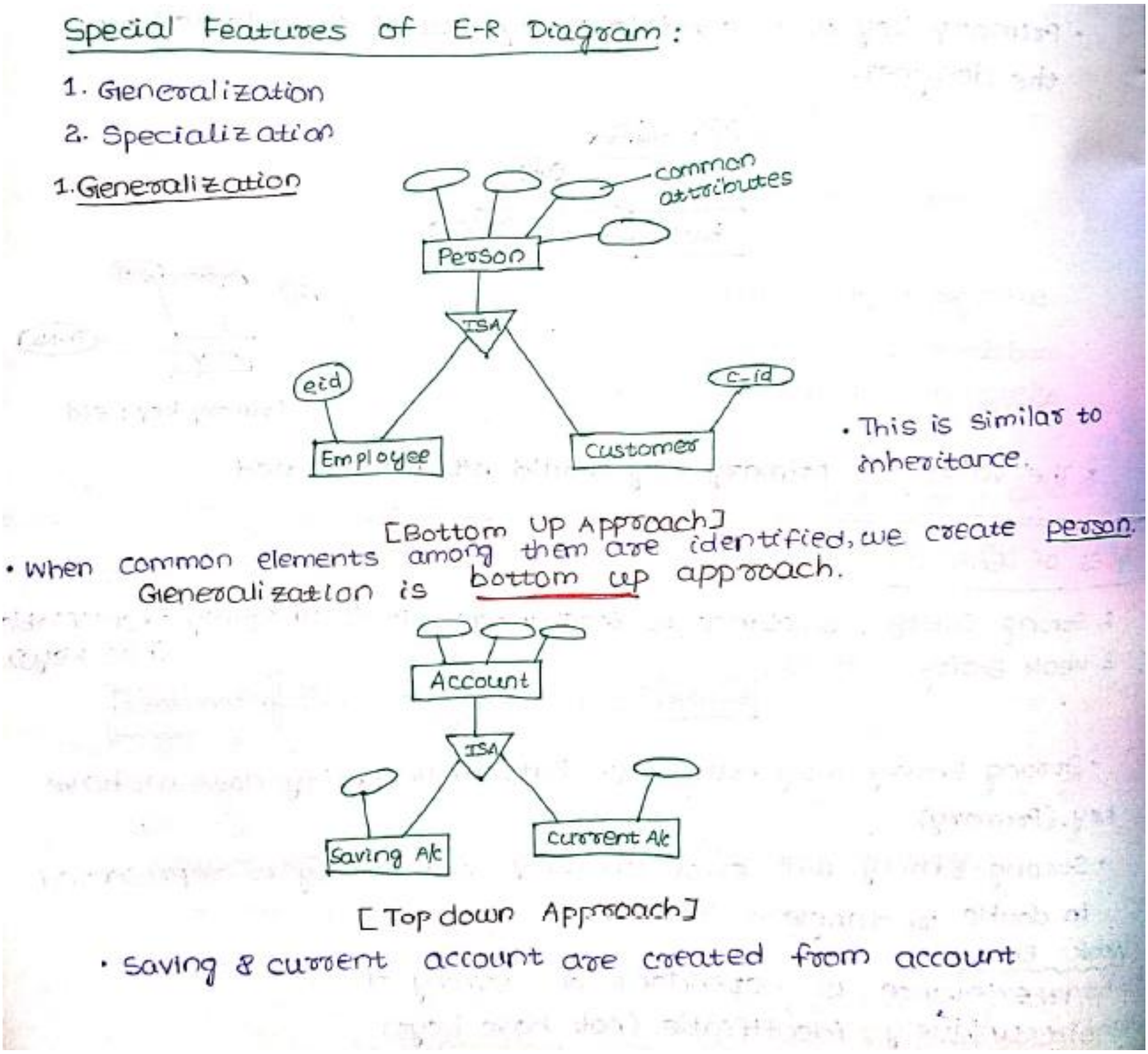
Examples:





Discriminator is an attribute which uniquely identifies weak entities.

Discriminator is a key which uniquely identifies Weak Entity. It is denoted as "----" line.



Special Features of E-R Diagram:

1. Generalization
2. Specialization

1. Generalization

• This is similar to inheritance.

[Bottom up Approach]

• When common elements among them are identified, we create person. Generalization is bottom up approach.

[Top down Approach]

• Saving & current account are created from account

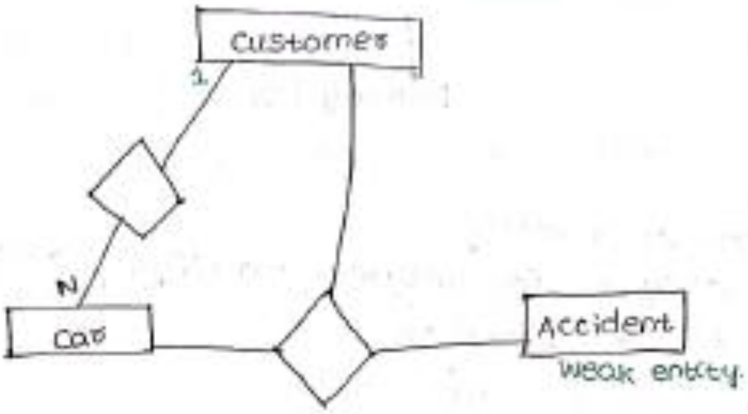
Generalization is a bottom-up approach in which two lower level entities combine to form a higher level entity. In generalization, the higher level entity can also combine with other lower level entities to make further higher level entity. For example, **Saving** and **Current** account types entities can be generalised and an entity with name **Account** can be created, which covers both.

Specialization is opposite to Generalization. It is a top-down approach in which one higher level entity can be broken down into two lower level entity. In specialization, a higher level entity may not have any lower-level entity sets, it's possible.

Q. Construct a E-R diagram for car Insurance Company whose customer own's one or more cars.

Buth car has zero or more recorded accidents.

A:



Employee

E_id	ENAME	Address	dept1	dept2	dept3	

- Weak Entity is used to avoid redundancy.
- If there is redundancy we can make them as weak entity. Weak entities are created to avoid redundancy.

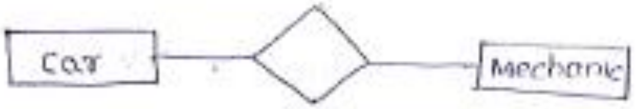
Relational Model :

- Table is called as a relation.
- Columns in a table is called Fields
- No. of Columns in a relation is the degree of relation (or) Entity of a relation.
- Rows in a table is called Tuples.
- Each value in a tuple has its own domain.
- If there is no value in the field the NULL value is used
- Key is used to retrieving the relation information from Relations identifies each Tuple.
- Key identifying each relation uniquely (a tuple is uniquely determined).

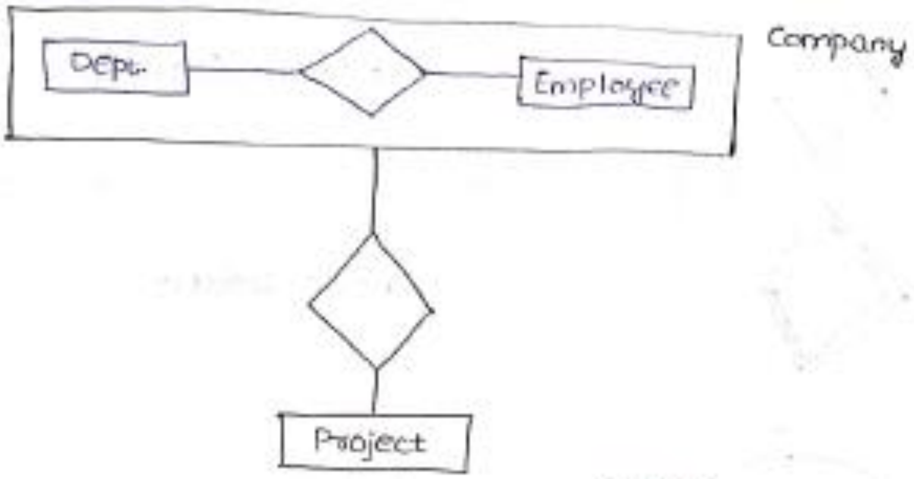
Aggregation:

In aggregation relationship itself becomes entity.

- We can represent containtment.

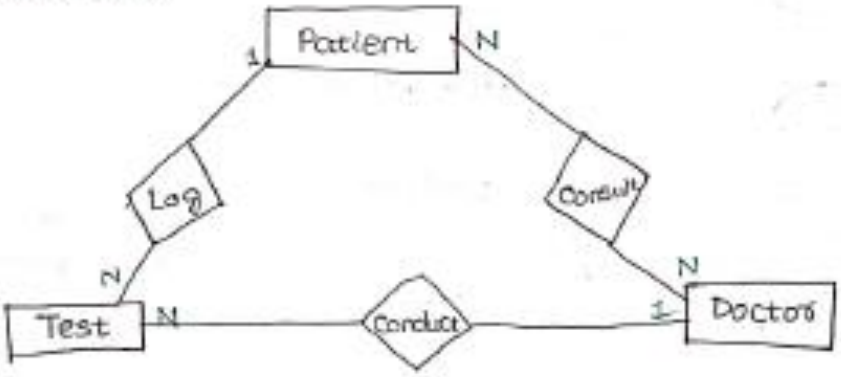


car consists of many entities: engine, tyre et

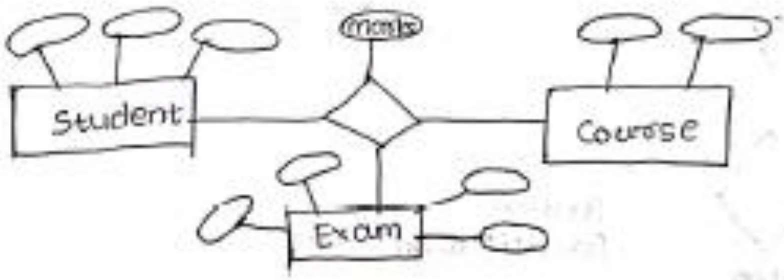


Q. Prepare a E-R diagram for a hospital with a set of patients and doctors associate with each patient, log of various test and examination conducted. Identify the entities & the attributes & show relationship.

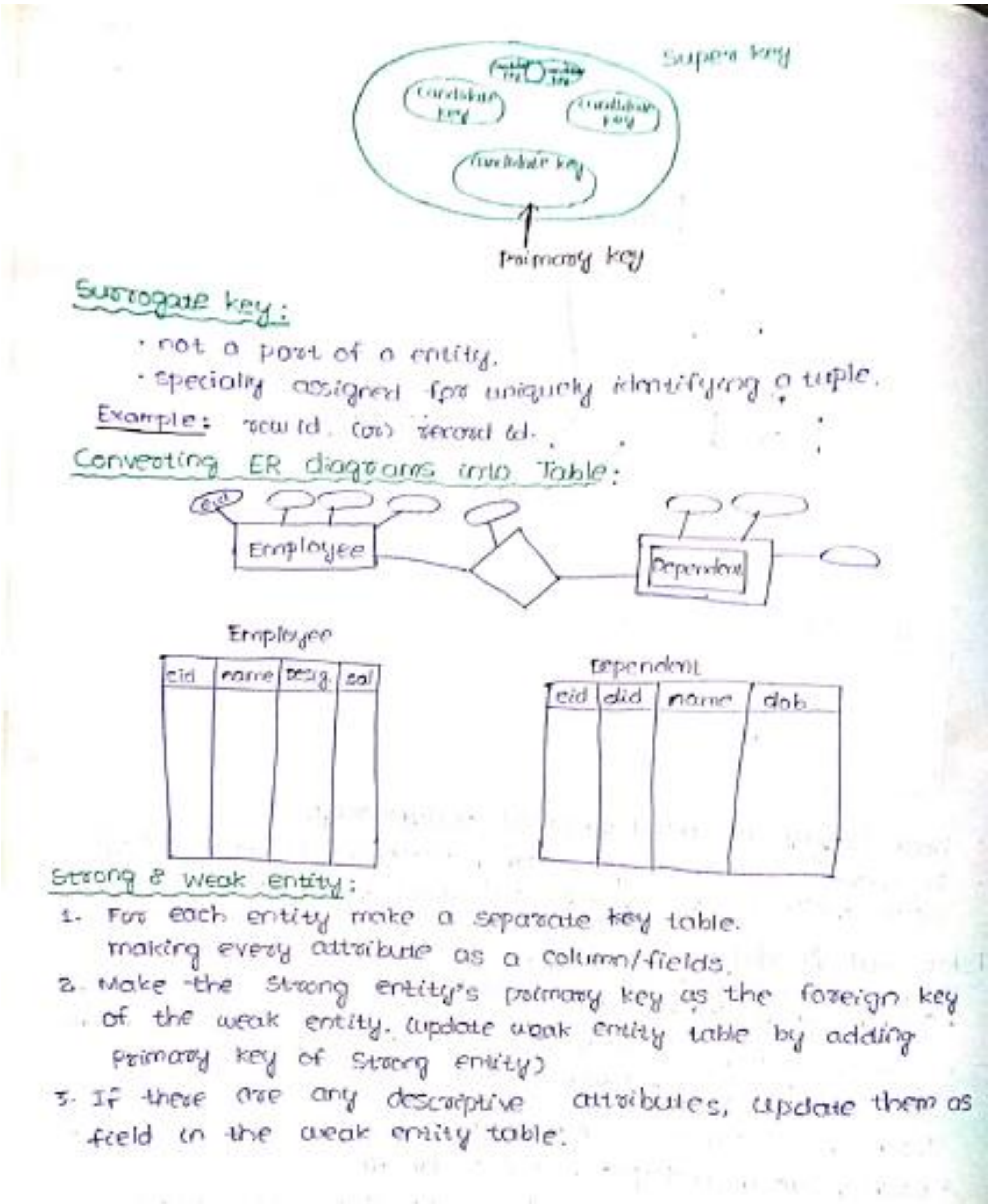
A:



Q. Construct E-R diagram for the student appearing Exam for the course.



A **surrogate key** is any column or set of columns that can be declared as the primary **key** instead of a "real" or natural **key**. Sometimes there can be several natural **keys** that could be declared as the primary **key**, and these are all called candidate **keys**.


Super key
candidate key
candidate key
candidate key
candidate key
candidate key
Primary key

Surrogate key:

- not a part of a entity.
- specially assigned for uniquely identifying a tuple.

Example: row id (or) record id.

Converting ER diagrams into Table:

eid
Employee
Dependent

Employee

eid	name	desig.	sal

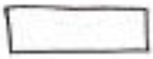
Dependent

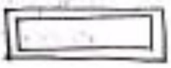
eid	did	name	dob

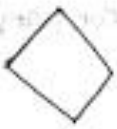
Strong & weak entity:

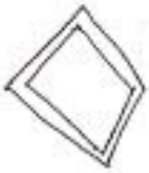
1. For each entity make a separate key table. making every attribute as a column/fields.
2. Make the Strong entity's primary key as the foreign key of the weak entity. (update weak entity table by adding primary key of Strong entity)
3. If there are any descriptive attributes, update them as field in the weak entity table.

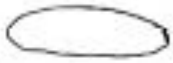
Symbols used in E-R diagram :

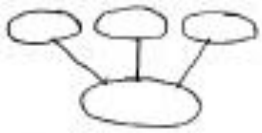

Strong Entity

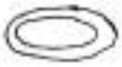

Weak Entity

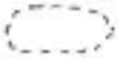

Relation (Strong)

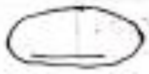

Weak Relation

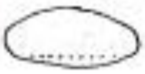

Attribute

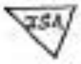

Composite Attribute


Multivalued

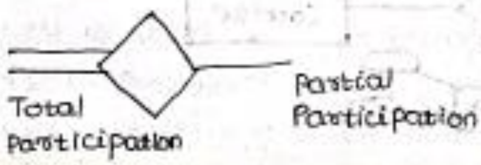

Derived


Key


Discriminator

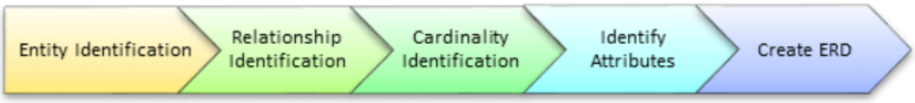


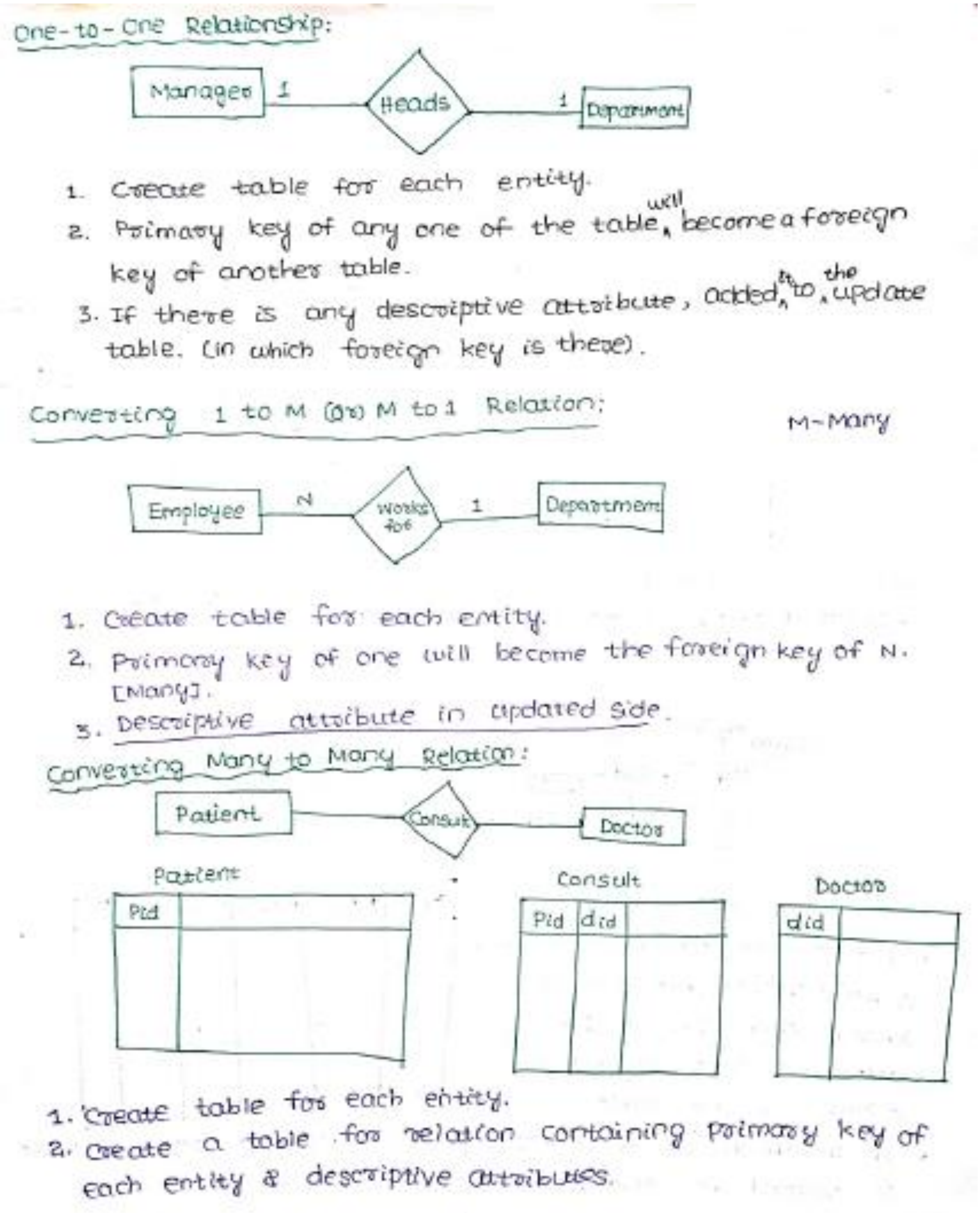
IS A



Steps to Create an ERD

Following are the steps to create an ERD.





One-to-One Relationship:

Manager
1
Heads
1
Department

1. Create table for each entity.
2. Primary key of any one of the table will become a foreign key of another table.
3. If there is any descriptive attribute, added it to the update table. (in which foreign key is there).

Converting 1 to M (or) M to 1 Relation:

M-Many

Employee
N
Works for
1
Department

1. Create table for each entity.
2. Primary key of one will become the foreign key of N. [Many].
3. Descriptive attribute in updated side.

Converting Many to Many Relation:

Patient

Pid	

Consult

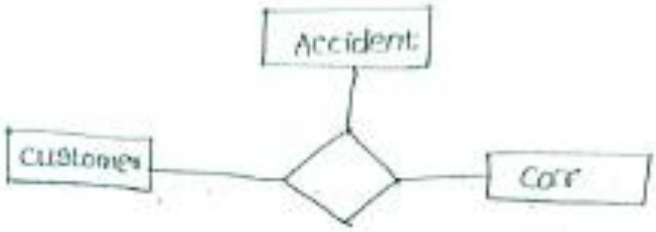
Pid	did	

Doctor

did	

1. Create table for each entity.
2. Create a table for relation containing primary key of each entity & descriptive attributes.

Converting Ternary (or) n-ary.

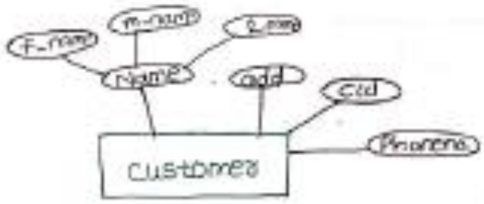


- same as many to many
1. For every entity create seperate table.
2. For one relation create a table containing primary key of every entity.

aid	cuid	cid	

Composite Attribute:

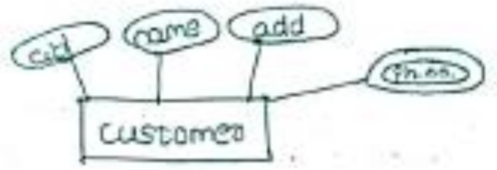
- Break a composite attribute into columns of a table.



cid	f.name	m.name	f.name	add	Ph. no.

Multi-valued Attribute:

- If maximum no. of values of a multi-valued attribute is known then break the attribute into as many columns of the table.
- If maximum no. of values are not known a seperate table is created for the multivalued attribute.



cid	name	add	Ph. no 1	Ph. no 2	Ph. no 3

Aggregation :

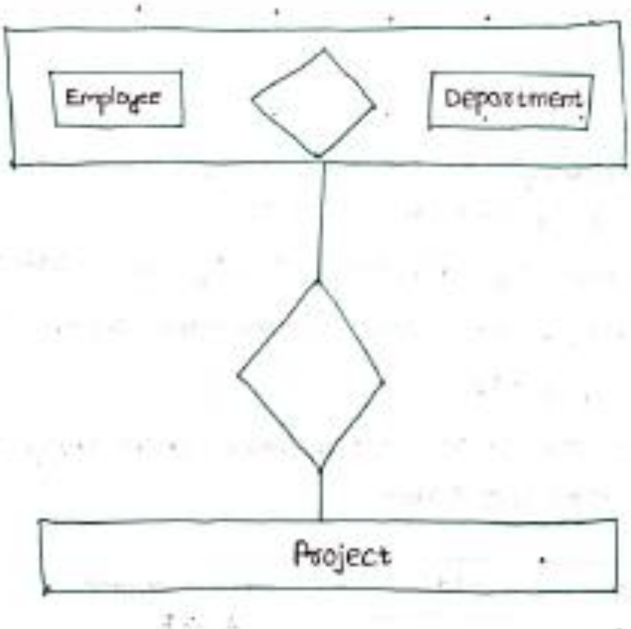
- It can be treated as a ternary relationship.
- If we are not doing so we have to create more tables
- For every entity create separate tables.

Customer

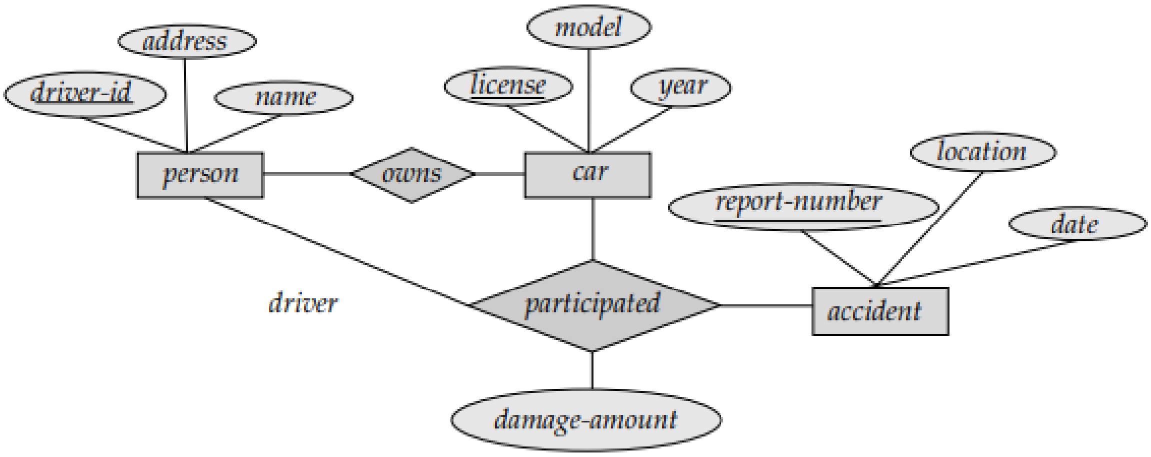
cid	name	add

Ph. no. table

cid	Ph. no.



An EXAMPLES of ERD


address
driver-id
name
person
owns
model
license
year
car
location
report-number
date
driver
participated
accident
damage-amount