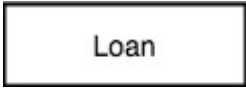
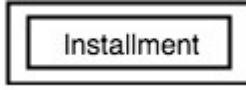
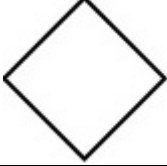
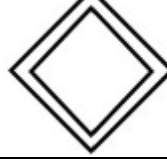







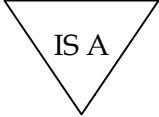
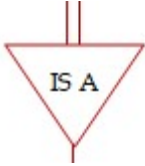

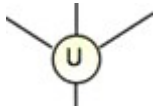
ER Diagrams

ER Diagrams

- An ER diagram shows **the relationship among entity sets**
- ER Diagram is a **visual representation of data** that describes how data is related to each other.
- Entity relationship diagram displays the relationships of entity set stored in a database.
- In other words, we can say that ER diagrams help you to **explain the logical structure of databases**.

ER diagram symbols

Symbol	Name
	Entity
	Weak Entity
	Relationship
	Weak Relationship
	Attribute / Simple Composite / Single valued Attributes
	Key Attribute
	Multi valued Attributes
	Derived Attributes

	Partial Key Attributes
	Generalization / Specialization
	Total Generalization
	Super class / Sub class
	Union / Category

Components of ER Diagram

- Entity, Attributes, Relationships etc form the components of ER Diagram and there are defined symbols and shapes to represent each one of them.

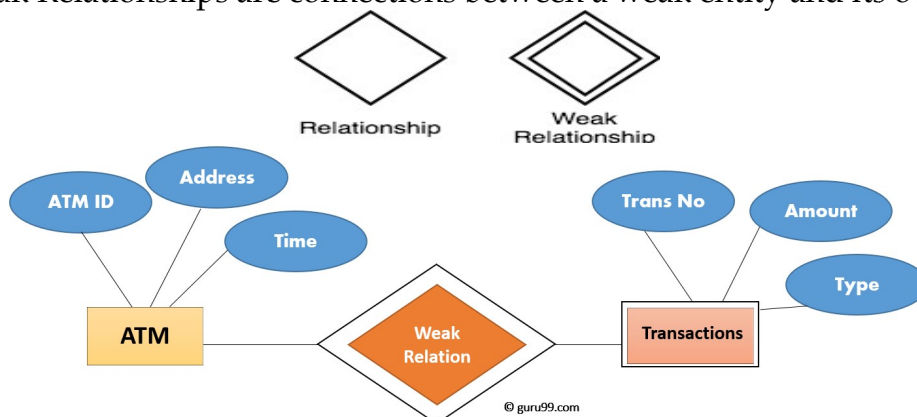
Entity

- Simple rectangular box represents an Entity.



Relationships between Entities - Weak and Strong

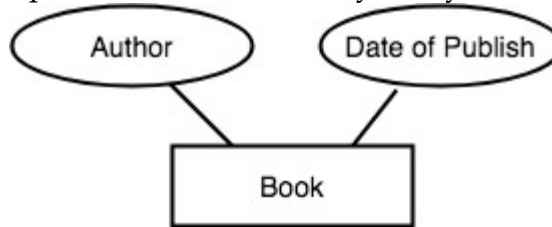
- Rhombus is used to setup relationships between two or more entities.
- Weak Relationships are connections between a weak entity and its owner.



- In above example, "Trans No" is a discriminator within a group of transactions in an ATM.

Attributes for any Entity

- Ellipse is used to represent attributes of any entity. It is connected to the entity.



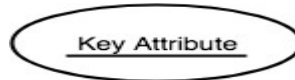
Weak Entity

- 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.
- They don't have primary keys, and have no meaning in the diagram without their parent entity.
- A weak Entity is represented using double rectangular boxes. It is generally connected to another entity.



Key Attribute for any Entity

- To represent a Key attribute, the attribute name inside the Ellipse is underlined. (Primary key)



Derived Attribute for any Entity

- Derived attributes are those which are derived based on other attributes, for example, age can be derived from date of birth.
- To represent a derived attribute, another dotted ellipse is created inside the main ellipse.



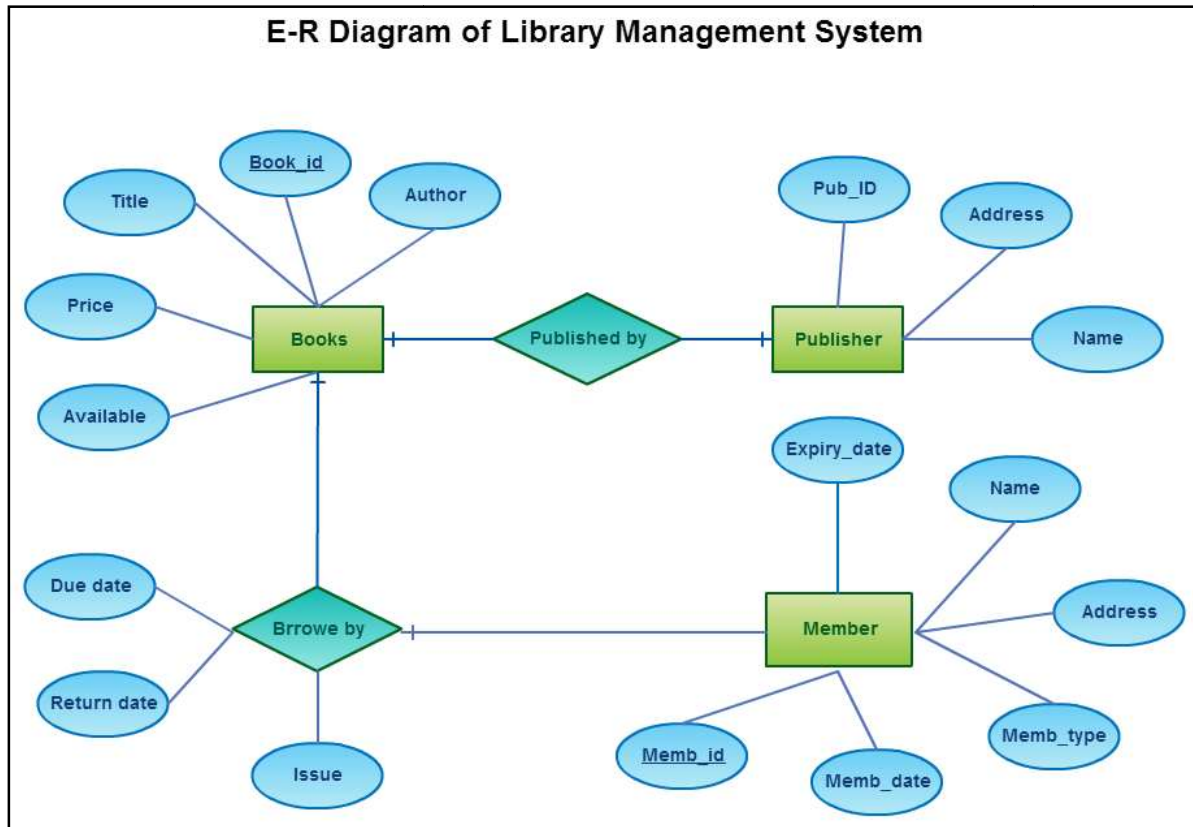
Multivalued Attribute for any Entity

- Multivalued attributes are those that are can take on more than one value.
- Double Ellipse, one inside another, represents the attribute which can have multiple values.

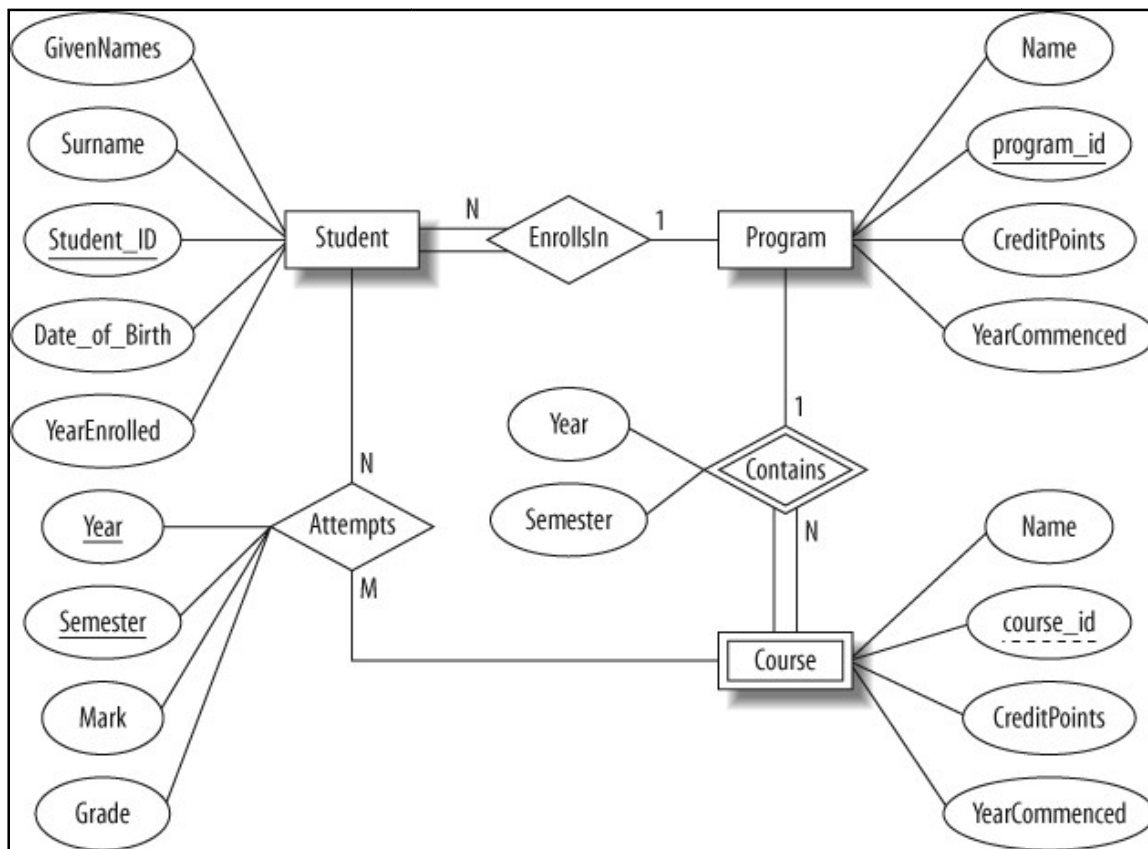


Sample ER diagrams

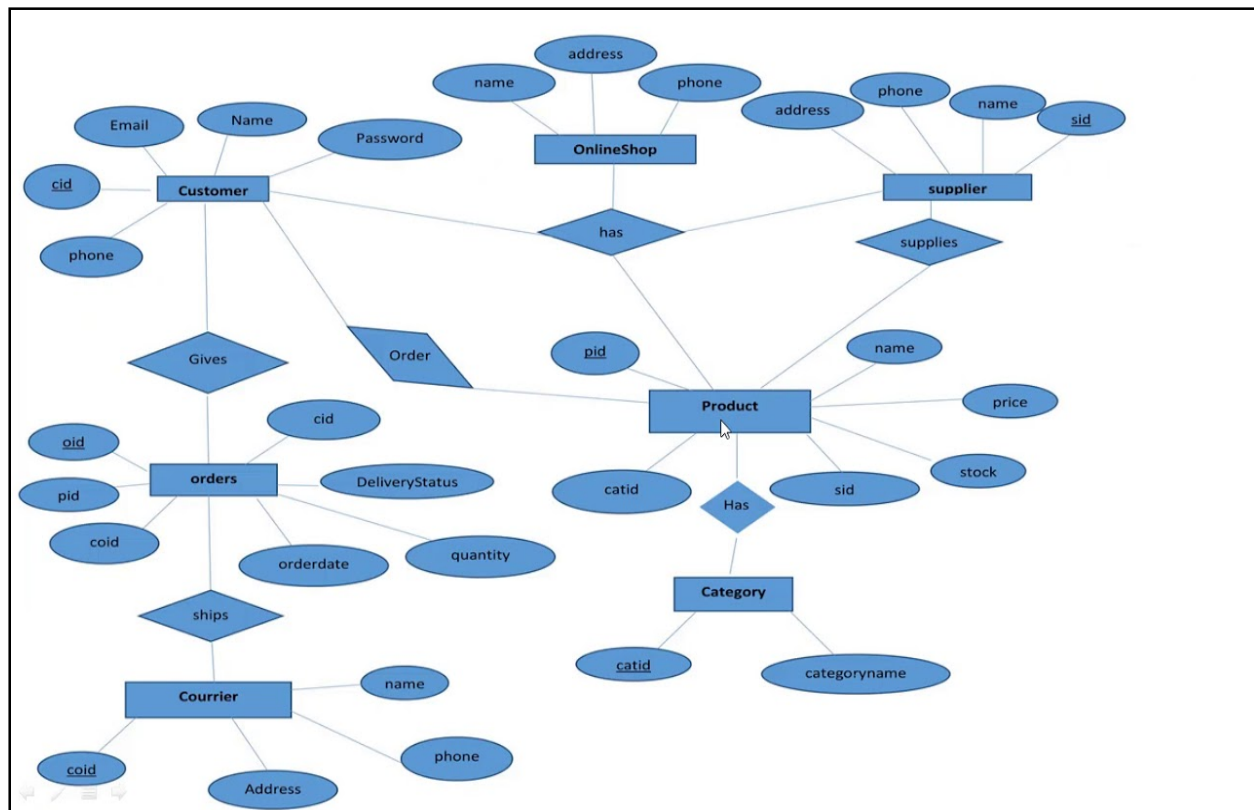
1) Library Management System



2) Student management System



3) Hospital Management System



Assignments on ER diagrams

- Employee database system
- Banking system
- Exam database system
- Hotel Management system
- Anna university database system
- Airline Reservation system
- Railway reservation system etc.,

Enhanced ER model/ Extended ER model

- ER model is supported with the additional semantic concepts is called as Extended / Enhanced Entity Relationship Model
- EER model includes following concepts
 - Specialization
 - Generalization
 - Aggregation
 - Sub class and Super Class

- Union or Category
- Enhanced entity-relationship diagrams are advanced database diagrams very similar to regular ER diagrams which represent requirements and complexities of complex databases.
- As the complexity of data increased in the late 1980s, it became more and more difficult to use the traditional ER Model for database modelling. Hence some improvements or enhancements were made to the existing ER Model to make it able to handle the complex applications better.

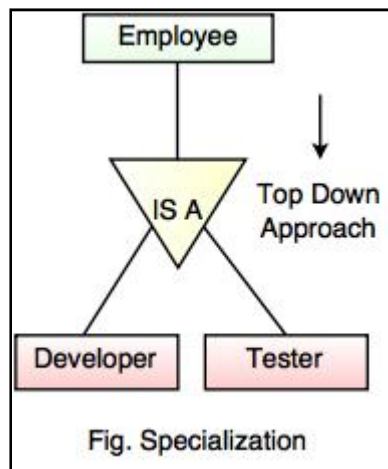
Features of EER Model

- EER creates a design more accurate to database schemas.
- It reflects the data properties and constraints more precisely.
- It includes all modeling concepts of the ER model.
- Diagrammatic technique helps for displaying the EER schema.
- It includes the concept of specialization and generalization.
- It is used to represent a collection of objects that is union of objects of different of different entity types.

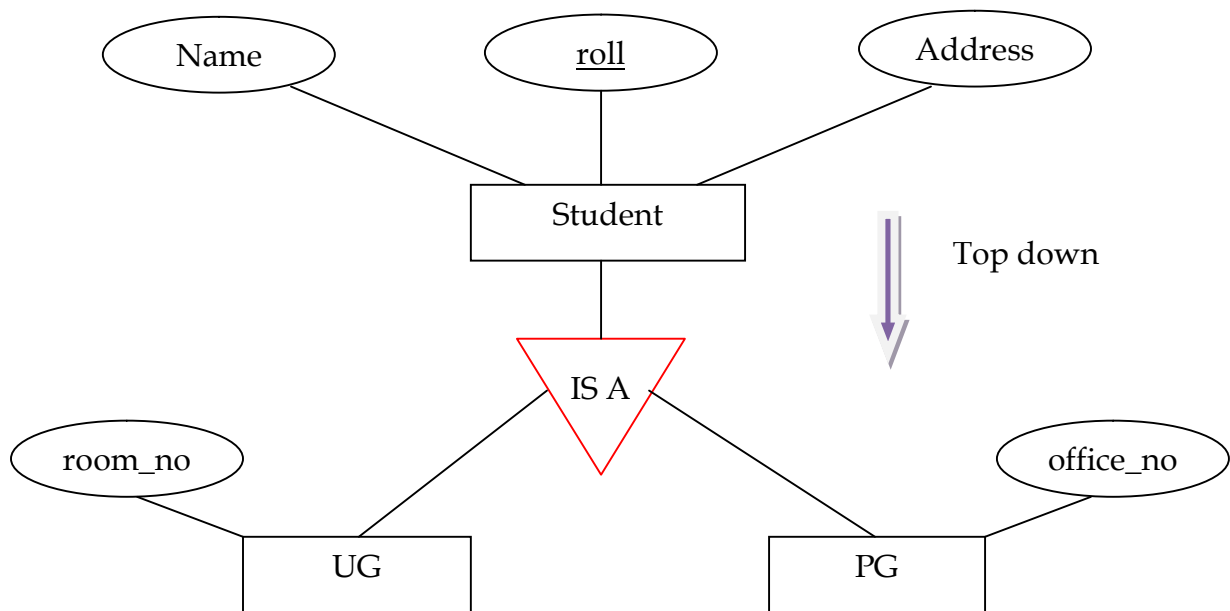
1. Specialization

- Specialization is the process of defining a set of subclasses of an entity
- Specialization is a process that defines a group entities which is divided into sub groups based on their characteristic.
- It is a top down approach, in which one higher entity can be broken down into two lower level entity.
- It maximizes the difference between the members of an entity by identifying the unique characteristic or attributes of each member.
- It defines one or more sub class for the super class and also forms the superclass/subclass relationship.

- **For example1**

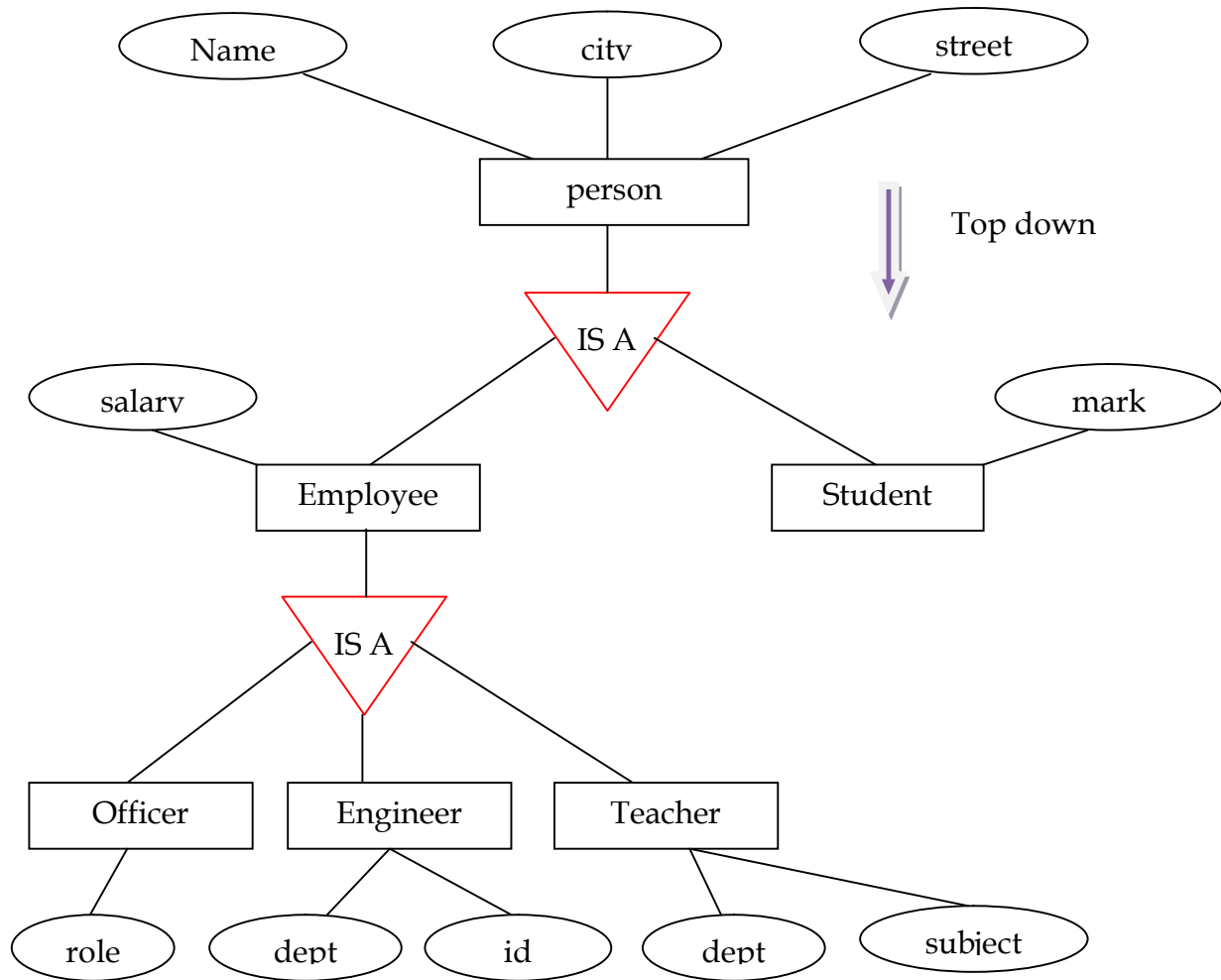


- In the above example, Employee can be specialized as Developer or Tester, based on what role they play in an Organization.
- Inverted triangle stands for “is a” relationship for both specialization and generalization.
- **Example2**
 - In university, student belongs to 2 categories “Undergraduate” & “Postgraduate”. Both categories of students may have common attributes as well as unique attributes(UG students in Hostel, PG students in Office).



UG Entity Attributes	PG entity Attributes
roll, Name, Address, room_no	roll, Name, Address, office_no

Example 3:



2. Generalization

- Generalization is the process of generalizing the entities which contain the properties of all the generalized entities.
- It is a bottom up approach, in which two lower level entities combine to form a higher level entity.
- Generalization is the reverse process of Specialization.
- It defines a general entity type from a set of specialized entity type.
- It minimizes the difference between the entities by identifying the common features.

For example:

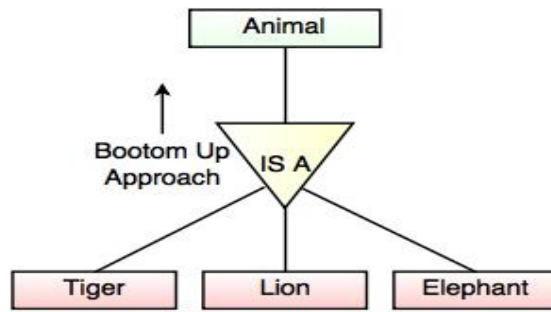
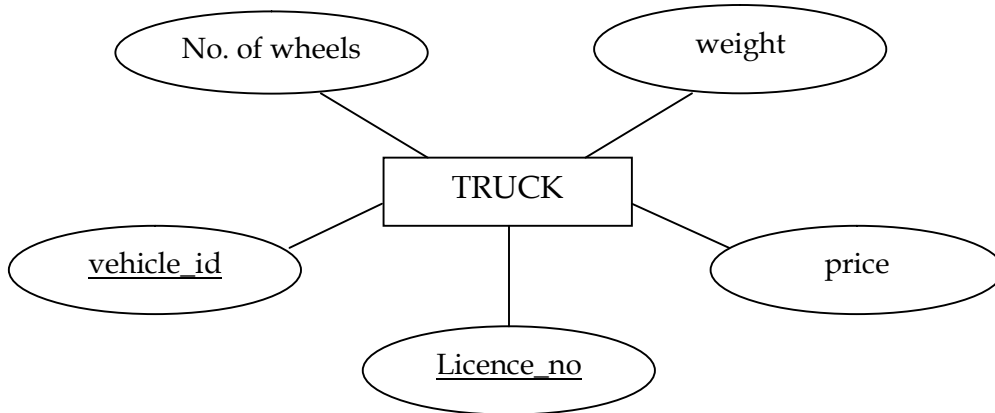
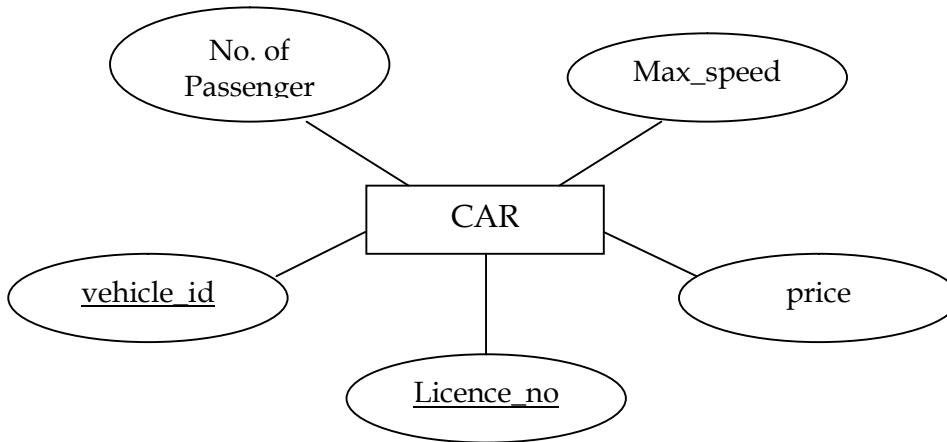


Fig. Generalization

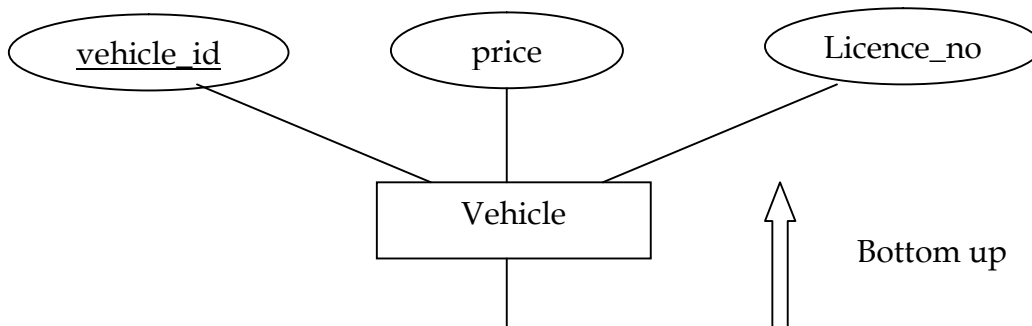
In the above example, Tiger, Lion, Elephant can all be generalized as Animals.

Example

- Consider 2-entity sets CAR & TRUCK



- In above 2 entity set, vehicle_id, price & Licence_no are common attributes.
- Hence generalize them into a single super class.



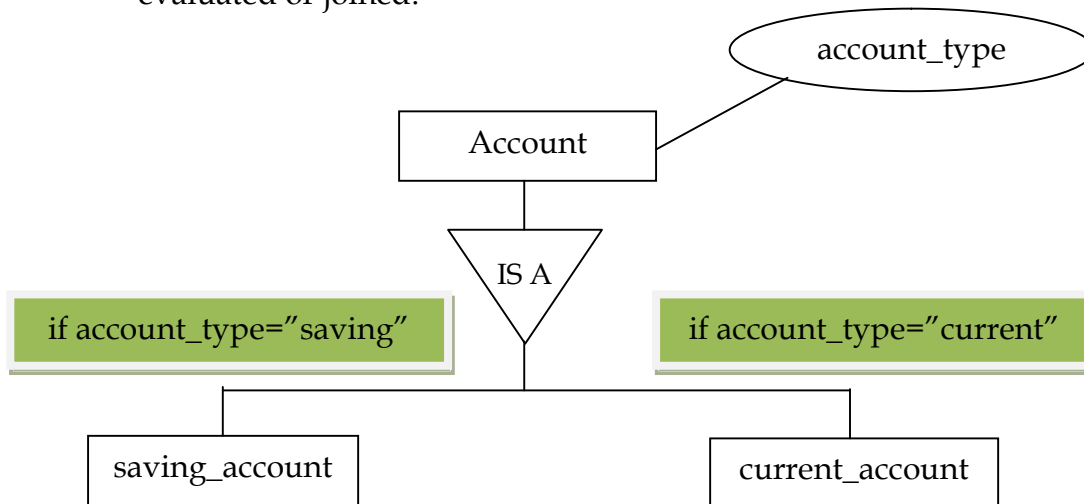
Generalization Constraints

- Constraints are application for generalization and specialization.
- There are three categories of constraints
 - Determine which entities can be member of a given lower -level entity set
 - Conditional defined
 - User Defined
 - Determine entities belong to more than one lower level entity set
 - Disjoint
 - Overlapping
 - Determine participation/belonging of higher level entity
 - Total generalization/specialization
 - Partial generalization /specialization

a) Determine which entities can be member of a given lower -level entity set

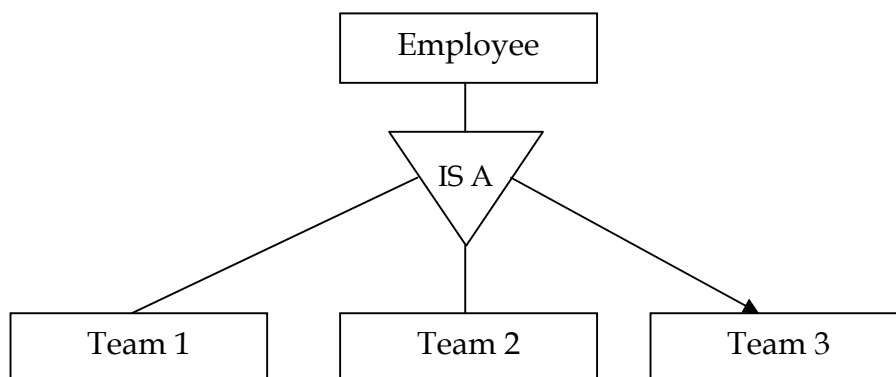
i. Conditional defined

- Condition is defined, based on that members of lower-level entities is evaluated or joined.



ii. User Defined

- Type of the constraints / members of entities are defined by the user

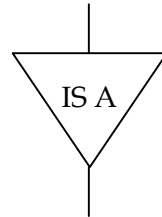


- If an employee joins an entity “employee”, **Team Incharge** will assign to any one of the 3 teams.
- It is defined by the user.

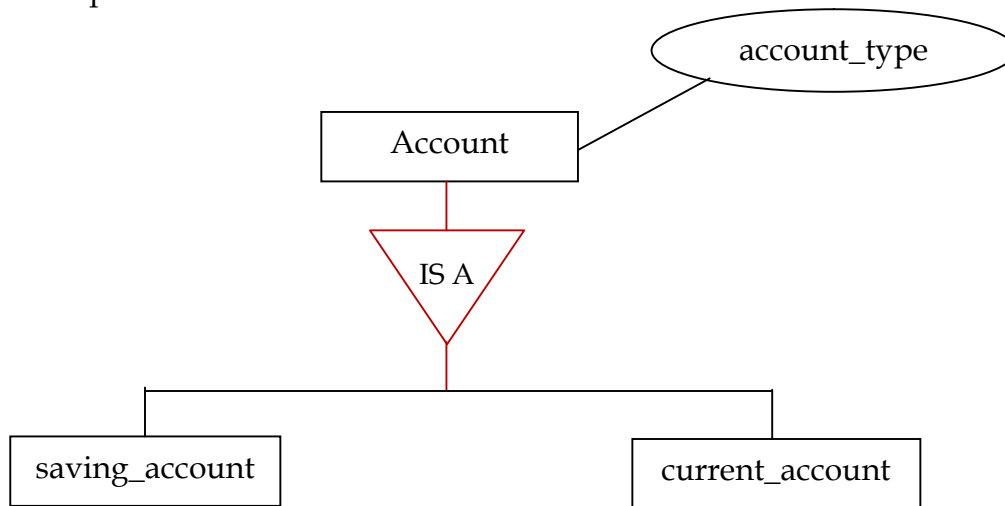
b) Determine entities belong to more than one lower level entity set

i. Disjoint

- Disjoint constraints specifies that an entity belong to only one lower-level entity set.
- It is represented user the ER notation



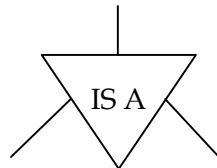
- Example



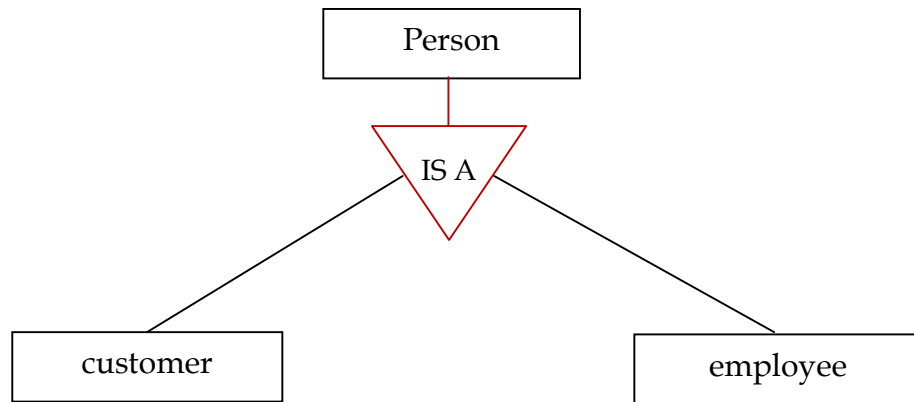
- User can **join only one** of the lower-level entity

ii. Overlapping

- In overlapping generalization, same **entity belongs to more than one lower-level entity set.**
- It is represented user the ER notation



Example

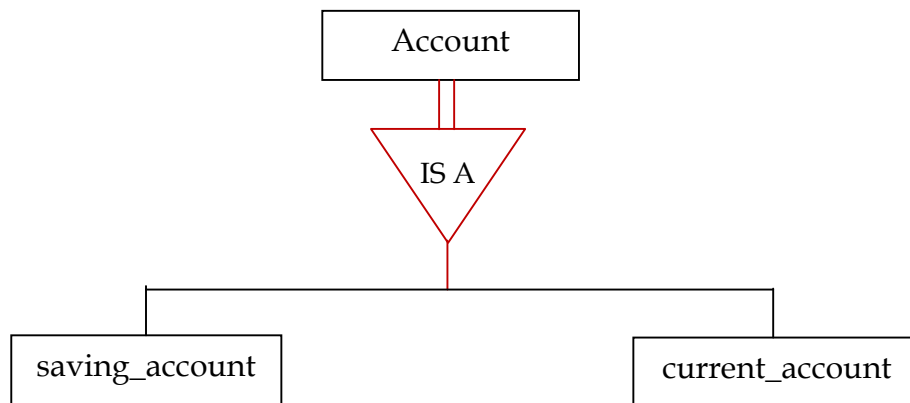


- A person can be customer & employee.
- Can participate in more than one entity of lower level.

c) Determine participation/belonging of higher level entity

i. Total generalization/specialization

- Each entity at higher level must belong to a lower-level entity set.
- Total generalization is represented using double line
- Example: Each account must belong to any one of the lower level entity.



ii. Partial generalization/specialization

- Some entities of higher level may not belong to any lower level entity set.
- Partial generalization is default
- The work team entity set illustrates a partial specialization. Since employees are assigned to a team only after three months on the job, some employee entities may not be members on any of the lower level team entity sets.

3. Aggregation

- Aggregation is an abstraction through which relationship are treated as higher-level entities.
- Aggregation is used to avoid redundant information.

- Aggregation is a process that represent a relationship between a whole object and its component parts.
- It abstracts a relationship between objects and viewing the relationship as an object.
- It is a process when two entity is treated as a single entity.

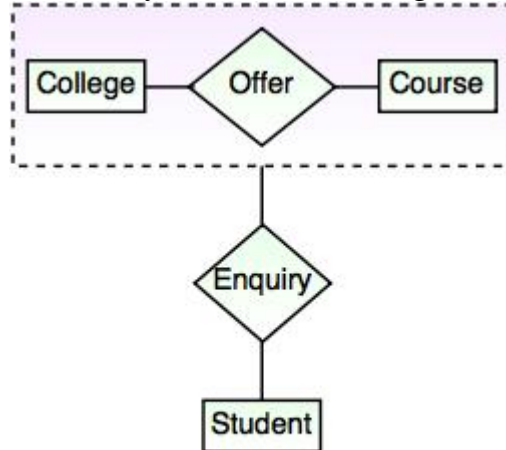
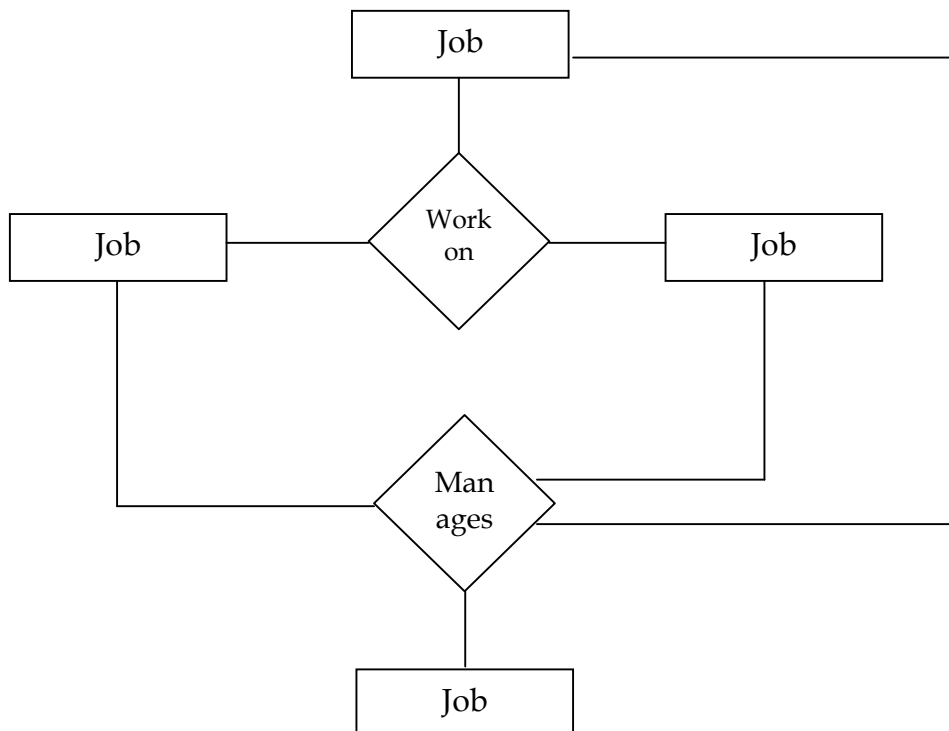


Fig. Aggregation

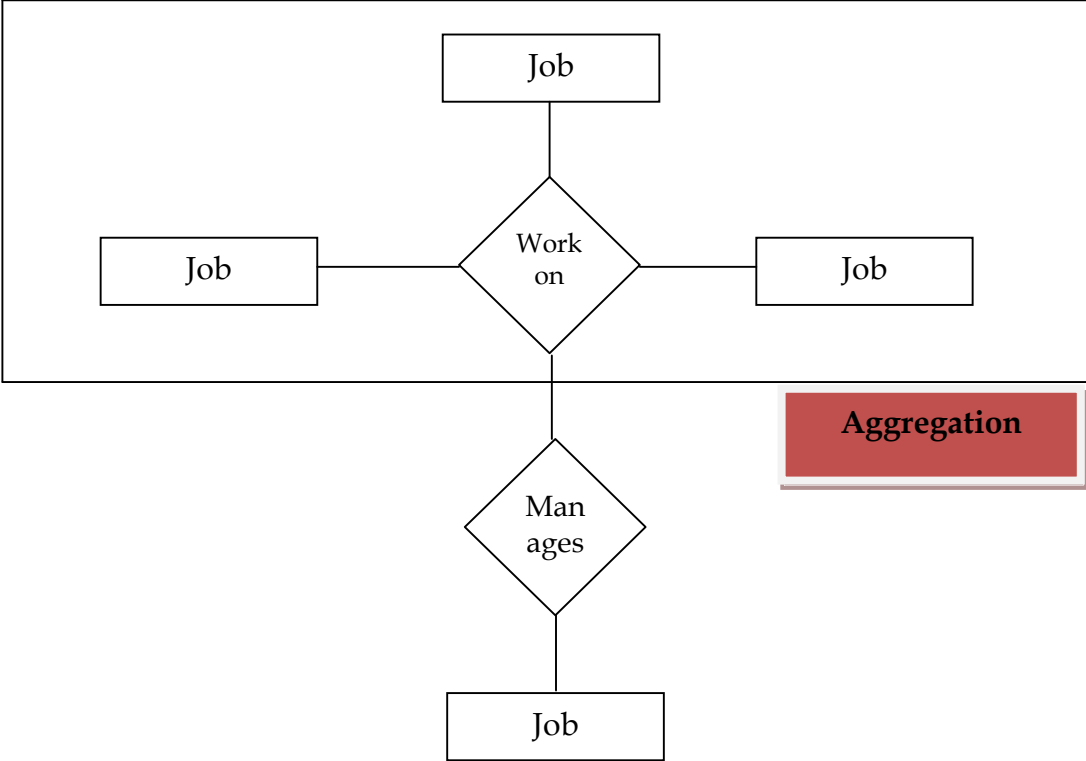
In the above example, the relation between College and Course is acting as an Entity in Relation with Student.

Example:

ER diagram with redundant relationship



ER Diagram with Aggregation



3. Sub Class and Super Class

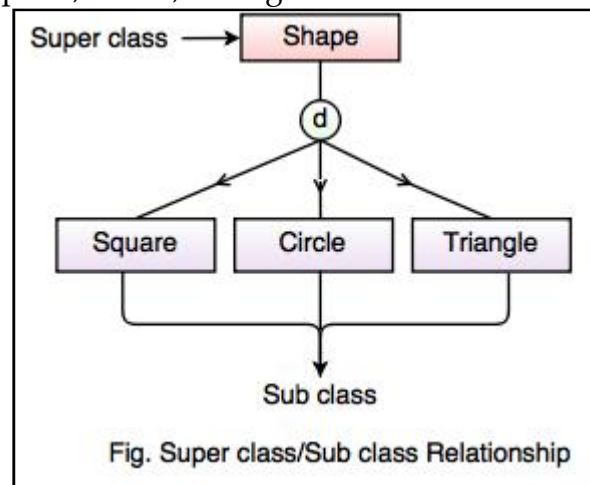
- Sub class and Super class relationship leads the concept of Inheritance.
- The relationship between sub class and super class is denoted with **(d)** symbol.

1. Super Class

- Super class is an entity type that has a relationship with one or more subtypes.
- An entity cannot exist in database merely by being member of any super class.
For example: Shape super class is having sub groups as Square, Circle, Triangle.

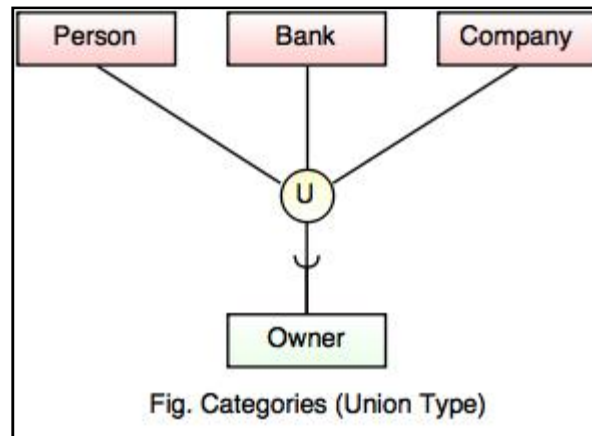
2. Sub Class

- Sub class is a group of entities with unique attributes.
- Sub class inherits properties and attributes from its super class.
For example: Square, Circle, Triangle are the sub class of Shape super class.



5. Category or Union

- Category represents a single super class or sub class relationship with more than one super class.
- It can be a total or partial participation.
For example Car booking, Car owner can be a person, a bank (holds a possession on a Car) or a company. Category (sub class) → Owner is a subset of the union of the three super classes → Company, Bank, and Person. A Category member must exist in at least one of its super classes.

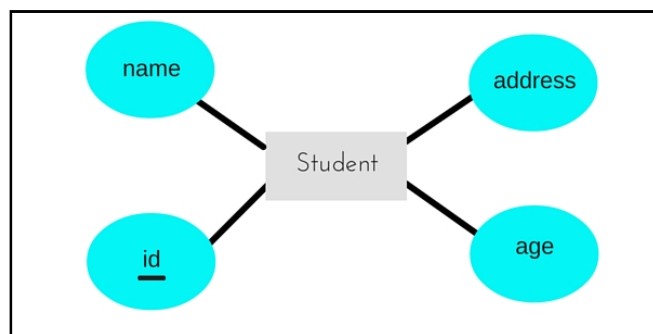


ER to Relational Mapping

- ER Model can be represented using ER Diagrams which is a great way of designing and representing the database design in more of a flow chart form.
- It is very convenient to design the database using the ER Model by creating an ER diagram and later on converting it into relational model to design your tables.
- Examples of ER diagrams and convert it into relational model schema, hence creating tables in RDBMS.

1. Mapping Entity

- Entity in ER Model is changed into tables, for every Entity in ER model, a table is created in Relational Model. (Entity → Table)
- The **attributes** of the Entity gets converted to columns of the table. (Attributes → Columns of the Table)
- The **primary key** specified for the entity in the ER model, will become the **primary key for the table** in relational model.
- For example, for the below ER Diagram in ER Model,

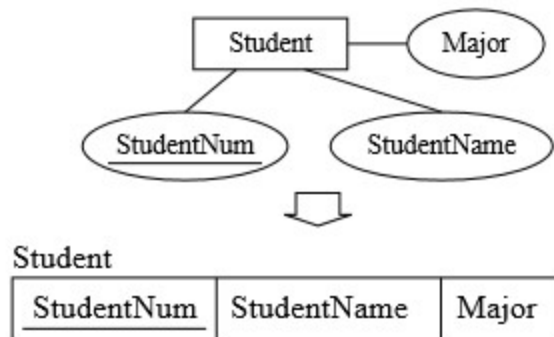


- A table with name **Student** will be created in relational model, which will have 4 columns, **id**, **name**, **age**, **address** and **id** will be the primary key for this table.

Table Name: Student

<u>id</u>	name	age	address

Example 2



Entity Mapping Process (Algorithm)

- Create table for each entity.
- Entity's attributes should become fields of tables with their respective data types.
- Declare primary key.

2. Mapping Relationship

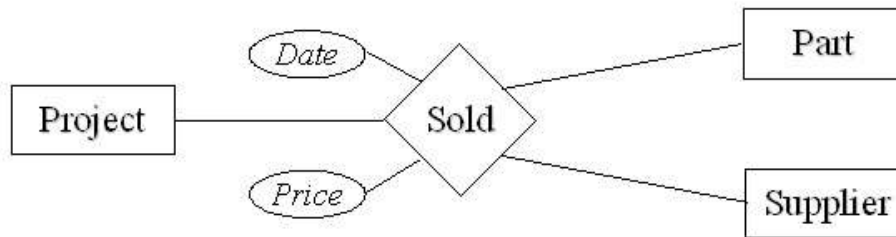
- In ER diagram, diamond/rhombus used to represent a relationship between two entities.
- In the ER diagram below, have two entities **Teacher** and **Student** with a relationship between them.

Example 1:



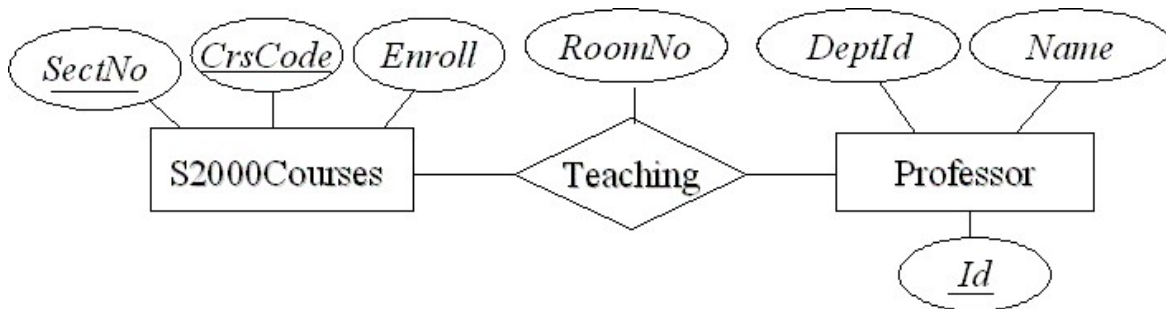
- Here, Entity gets mapped to table,
 - hence we will create table for **Teacher** and a table for **Student** with all the attributes converted into columns.
- Now, an additional table will be created for the relationship,
 - for example **StudentTeacher** or give it any name you like.
 - This table will hold the primary key for both Student and Teacher, in a tuple to describe the relationship, which teacher teaches which student.
- If there are additional attributes related to this relationship, then they become the columns for this table, like subject name.
- Also proper foreign key constraints must be set for all the tables.

Example 2



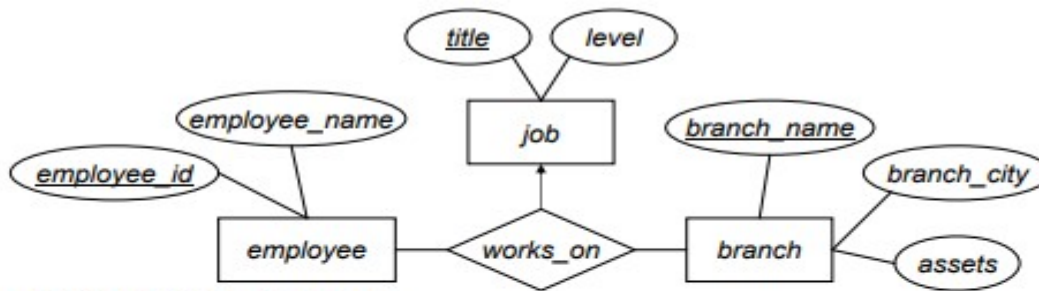
- Project(ProjId, Name, TotalCost, StartDate)
- Parts(UPC, PartName, Weight, WSPPrice)
- Suppliers(SupId, Name, Address)
- **Sold(ProjId, UPC, SupId, Date, Price)**

Example 3:



- S2000Courses (CrsCode, SectNo, Enroll)
- Professor (Id, DeptId, Name)
- **Teaching (CrsCode, SecNo, Id, RoomNo)**

4:Example



- Entity-set schemas:
job(title, level)
employee(employee_id, employee_name)
branch(branch_name, branch_city, assets)
- Relationship-set schema:
 - Primary key includes entity-sets on non-arrow links
works_on(employee_id, branch_name, title)

Relationship Mapping Process (Algorithm)

- Create table for a relationship.
- Add the primary keys of all participating Entities as fields of table with their respective data types.
- If relationship has any attribute, add each attribute as field of table.
- Declare a primary key composing all the primary keys of participating entities.
- Declare all foreign key constraints.

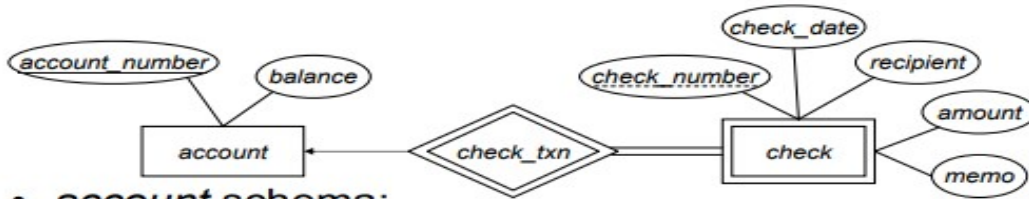
3. Mapping Weak Entity

- Weak entity-sets depend on at least one strong entity-set
- Identifying entity-set, or owner entity-set
- Relationship between the two called the identifying relationship
- A weak entity set is one which does not have any primary key associated with it.

Mapping Process

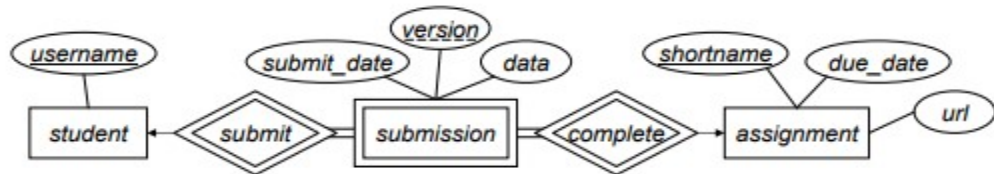
- Create table for weak entity set.
- Add all its attributes to table as field.
- Add the primary key of identifying entity set.
- Declare all foreign key constraints.

Example 1



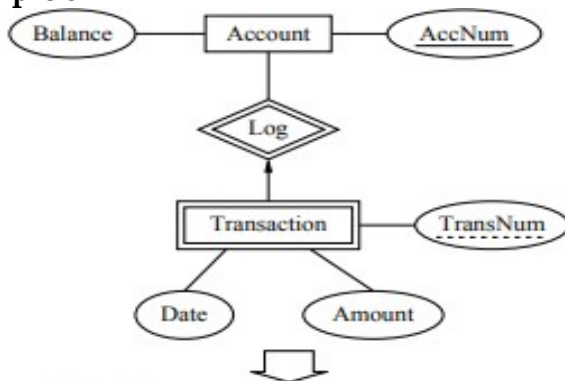
- **account** schema:
account(account_number, balance)
- **check** schema:
 - Discriminator is *check_number*
 - Primary key for *check* is:
 (account_number, *check_number*)*check*(account_number, *check_number*, *check_date*, recipient, amount, memo)

Example 2



- Schemas for strong entity-sets:
student(username)
assignment(shortname, due_date, url)
- Schema for *submission* weak entity-set:
 - Discriminator is *version*
 - Both *student* and *assignment* are owners!*submission*(username, *shortname*, *version*, submit_date, data)

Example 3



Account

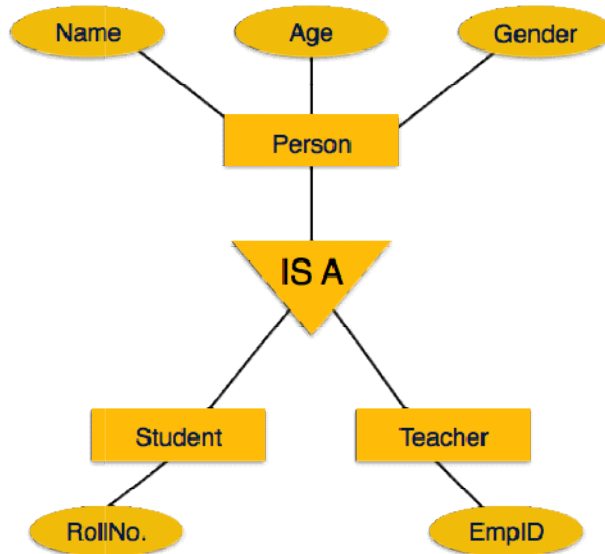
<u>AccNum</u>	Balance
---------------	---------

Transaction

<u>TransNum</u>	<u>AccNum</u>	Date	Amount
-----------------	---------------	------	--------

Mapping Hierarchical Entities

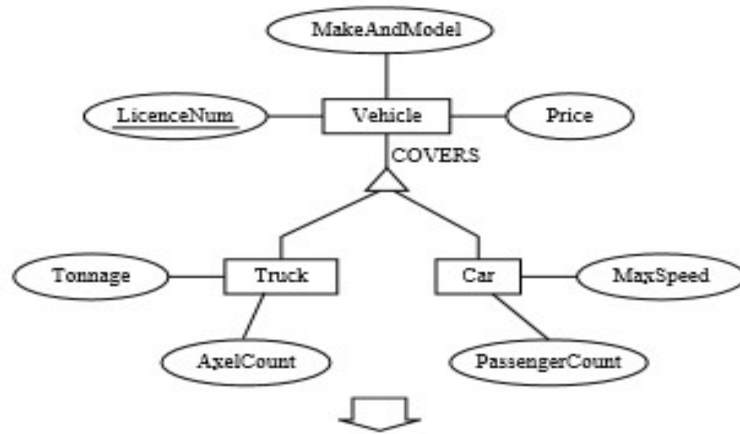
ER specialization or generalization comes in the form of hierarchical entity sets.



Mapping Process

- Create tables for all higher-level entities.
- Create tables for lower-level entities.
- Add primary keys of higher-level entities in the table of lower-level entities.
- In lower-level tables, add all other attributes of lower-level entities.
- Declare primary key of higher-level table and the primary key for lower-level table.
- Declare foreign key constraints.

le:



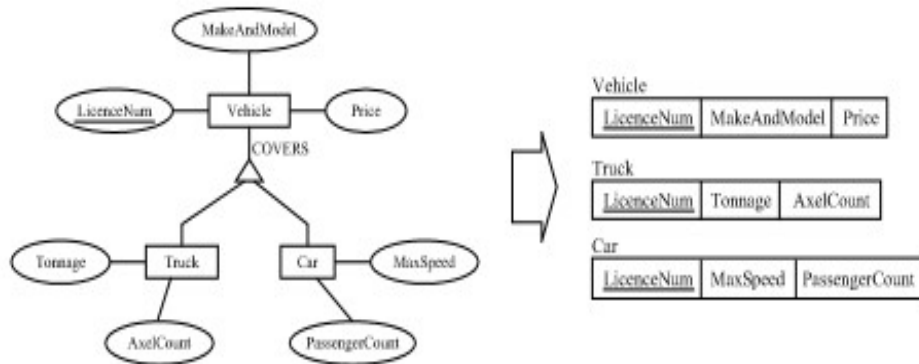
Truck

<u>LicenceNum</u>	MakeAndModel	Price	Tonnage	AxelCount
-------------------	--------------	-------	---------	-----------

Car

<u>LicenceNum</u>	MakeAndModel	Price	MaxSpeed	PassengerCount
-------------------	--------------	-------	----------	----------------

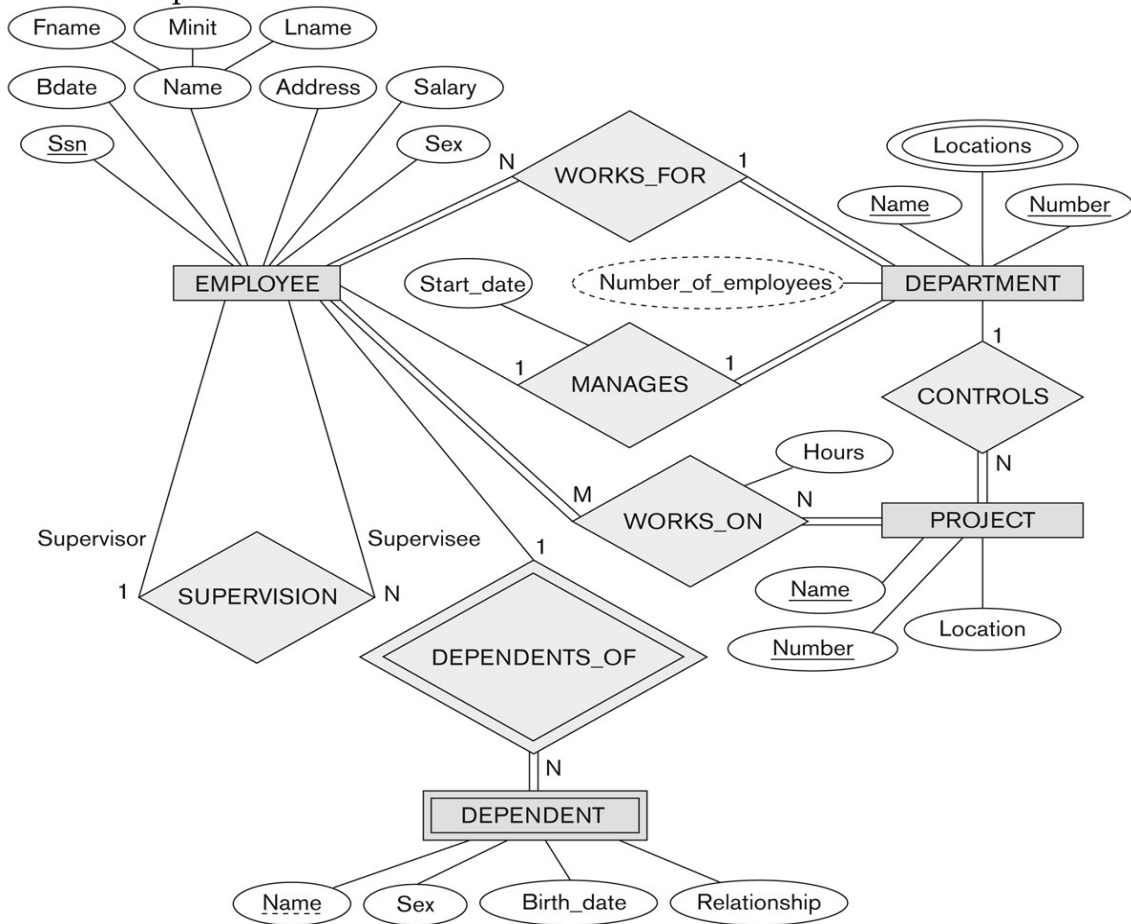
(or)



Recall

- Entity gets converted into Table, with all the attributes becoming fields(columns) in the table.
- Relationship between entities is also converted into table with primary keys of the related entities also stored in it as foreign keys.
- Primary Keys should be properly set.
- For any relationship of Weak Entity, if primary key of any other entity is included in a table, foreign key constraint must be defined.

Text book Example:



Result

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	-----	-------	---------	-----	--------	-----------	-----

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

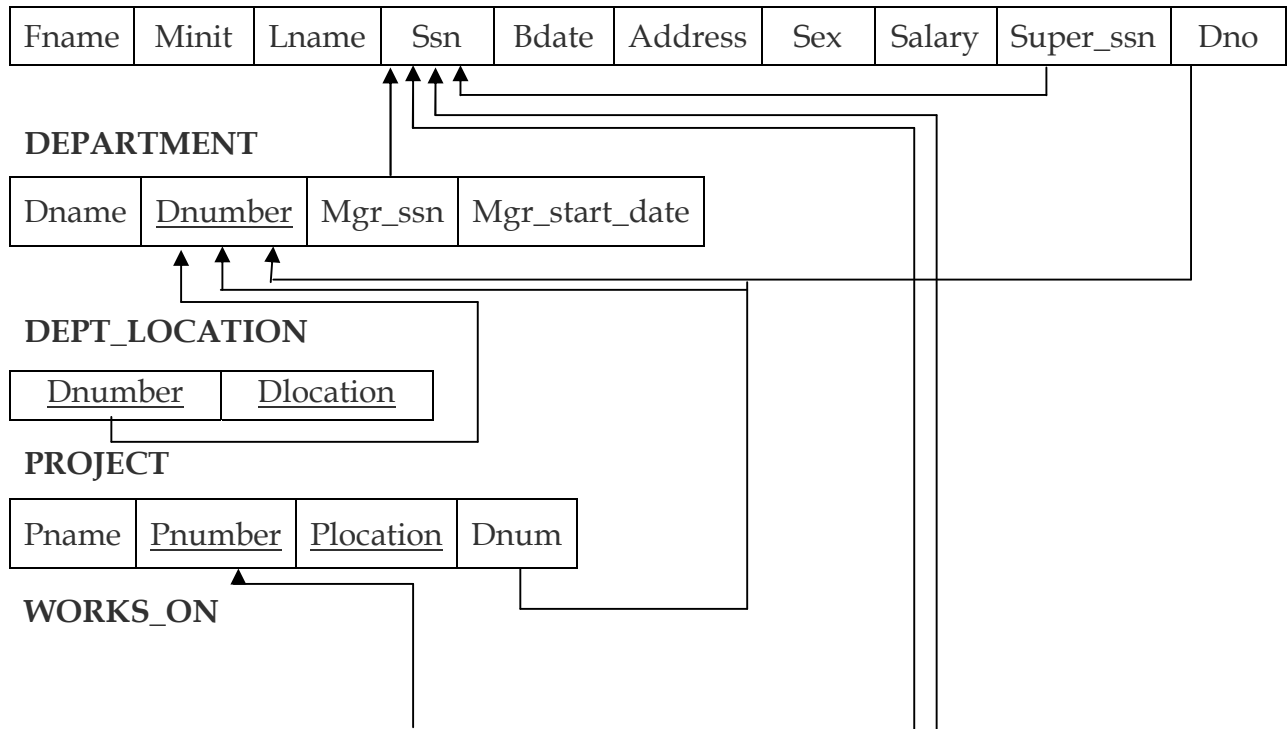
DEPT_LOCATION

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

PROJECT

Pname	<u>Pnumber</u>	<u>Plocation</u>	Dnum
-------	----------------	------------------	------

WORKS_ON



<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship
-------------	-----------------------	-----	-------	--------------

