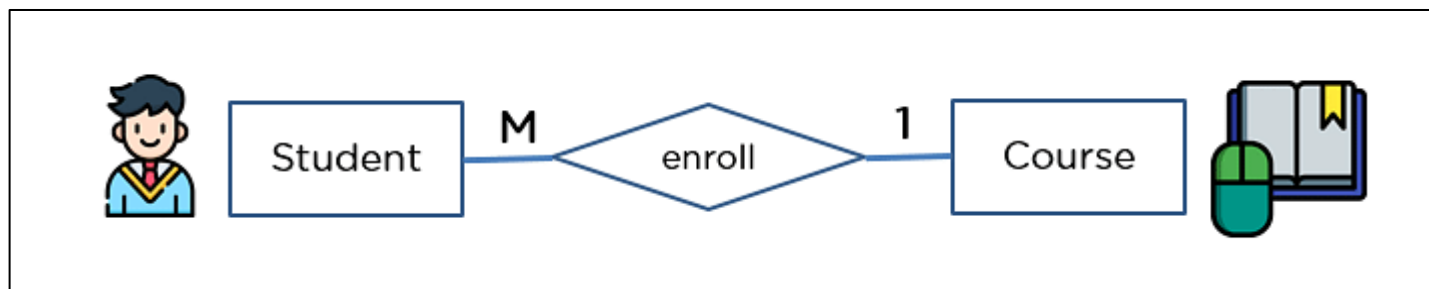


Database Management Systems (BCSC-1003)

Topic: Entity-Relationship Model



Nishtha Parashar

Assistant Professor, Dept. of CEA, GLA University Mathura

E-R Model

- Introduction
- Entity
- Types of Entity
- Attributes
- Types of Attributes
- Relationship
- Degree of a Relationship
- Cardinality

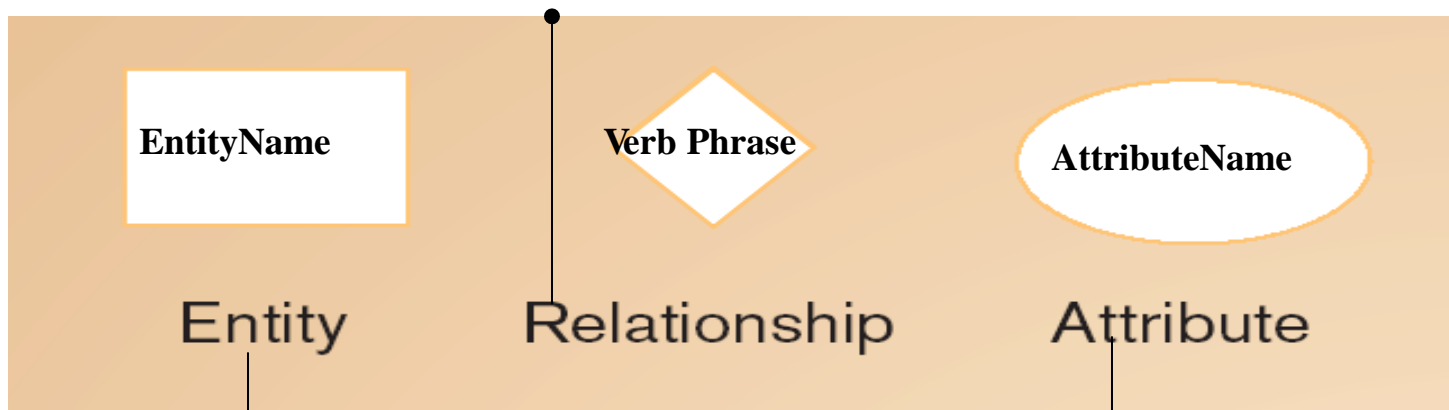
E-R Model

- An E-R model is the logical representation of data as objects and relationships among them.
- These objects are known as entities, and relationship is an association among these entities.
- This model was designed by Peter Chen in 1976.
- This model is widely used in database designing.

ER Modeling

- **ER Modeling:** A **graphical technique** for *understanding* and organizing the data independent of the actual database implementation.
- **Entity:** Any thing that may have an independent existence and about which we intend to collect data.
Also known as **Entity type**. E.g.: **Trainee**
- **Relationships:** Associations between entities. E.g.: Trainee belongs to a Batch
- **Attributes:** Properties/characteristics that describe entities.eg: Trainee Name, BatchName, DOB, Address, etc.

Association between the instances of one or more entity types

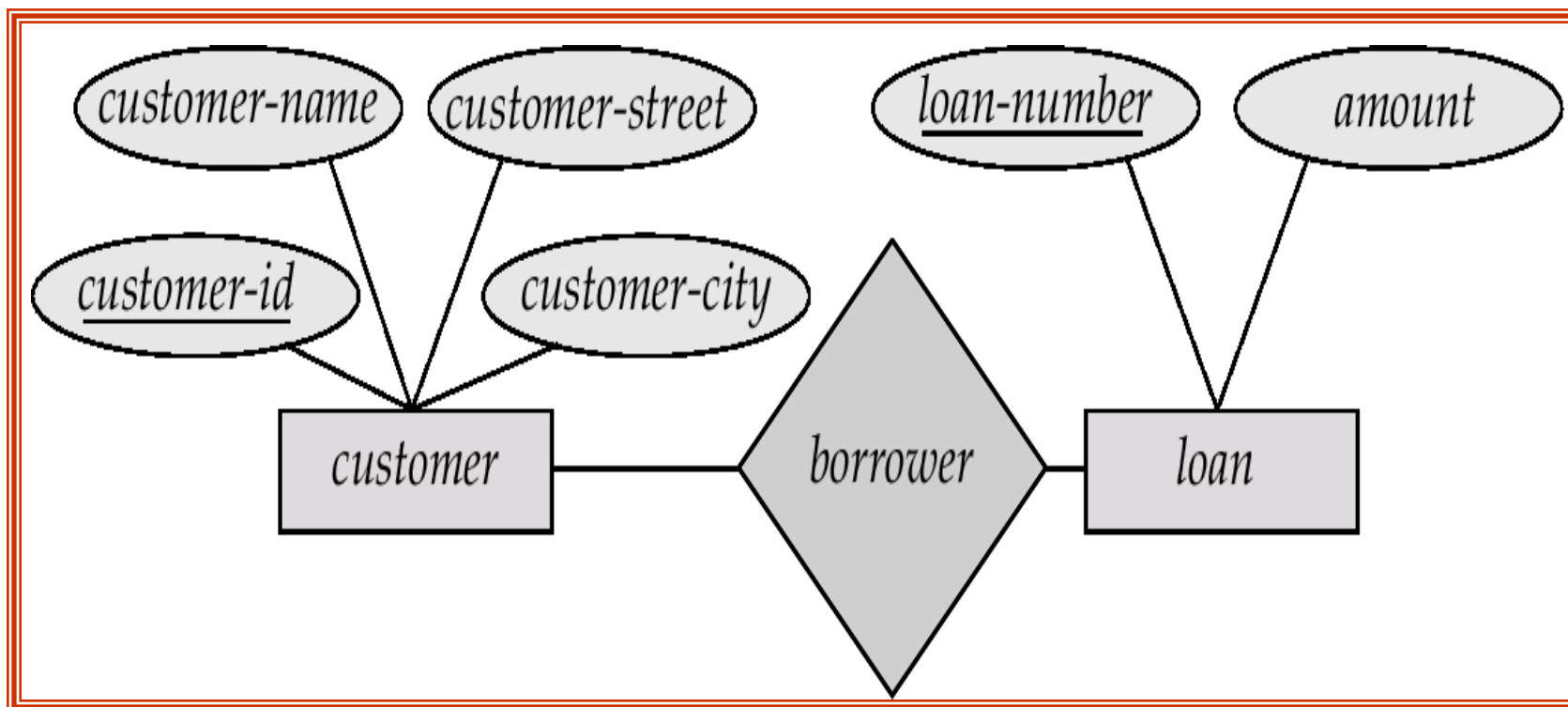


Person, place, object, event or concept about which data is to be maintained

named property or characteristic of an entity

Represents a set or collection of objects in the real world that share the same properties

An Example



Entities

- **Examples of entities:**

- Person: EMPLOYEE, STUDENT, PATIENT
- Place: STORE, WAREHOUSE
- Object: MACHINE, PRODUCT, CAR
- Event: SALE, REGISTRATION, RENEWAL
- Concept: ACCOUNT, COURSE



- **Guidelines for naming and defining entity types:**

- An entity type name is a singular noun
- An entity type should be descriptive and specific
- An entity name should be concise
- Event entity types should be named for the result of the event, not the activity or process of the event.

Entity Types

1. Regular Entity/Strong Entity: Entity that has its own key attribute (s). Its existence is not dependent on any other entity.

E.g.: Employee, student, customer, policy holder etc.

-A *TIRE* might be considered as a strong entity because it also can exist without being attached to a *CAR*.

2. Weak Entity: Entity that depends on other entity for its existence and doesn't have key attribute (s) of its own.

E.g.: A *ROOM* can only exist in a *BUILDING*.

Attributes

- **Example of entity types and associated attributes:**

STUDENT: *Student_ID, Student_Name, Home_Address, Phone_Number, Major*

- **Guidelines for naming attributes:**

- An attribute name is a noun.
- An attribute name should be unique
- To make an attribute name unique and clear, each attribute name should follow a standard format

- **Attribute Types:**

- Simple and composite attributes
- Single-valued and multi-valued attributes
- Derived attribute

Attribute Type

Types of Attributes	Definition	Example
Simple Attribute	Cannot be divided into simpler components	Gender of the employee
Composite Attribute	Can be split into components	Name of the employee (First Name, Last Name)
Single-Valued	Can take on only a single value for each entity instance	Age of the employee
Multi-Valued	Can take up many values	Phone No. of the employee (Mb. No., Landline No.)
Stored Attribute	Attribute that need to be stored permanently	Date of joining of the employee
Derived Attribute	Attribute that can be calculated based on other attributes.	Years of service of the employee

Attributes

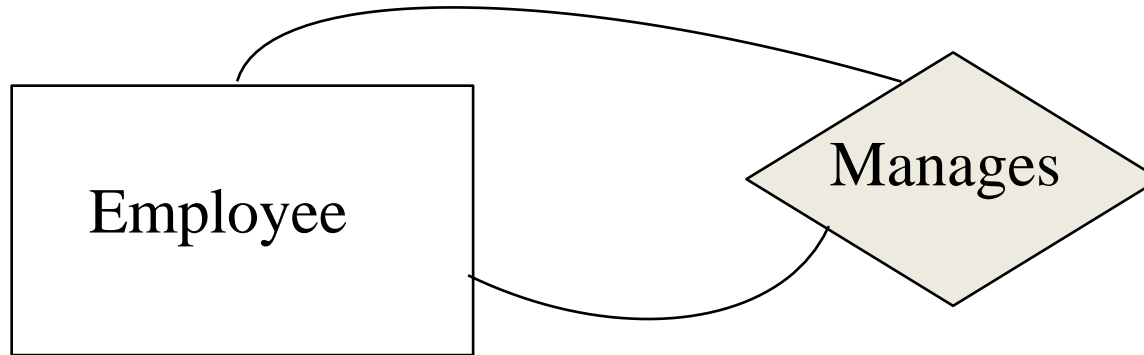
- The set of possible values for an attribute is called the **domain of the attribute**.
 - *Example: The domain of the attribute month is having twelve values ranging from January to December.*
- **Key attribute**: The attribute (or combination of attributes) that is unique for every entity instance.
 - *Example: the account number of an account, the employee id of an employee etc.*

DEGREE OF A RELATIONSHIP

- **Degree:** the number of entity types involved
 - One *Unary*
 - Two *Binary*
 - Three *Ternary*

*E.g.: 1. employee **manager-of** employee is unary
2. employee **works-for** department is binary
3. customer **purchase** item, shop keeper is a ternary relationship*

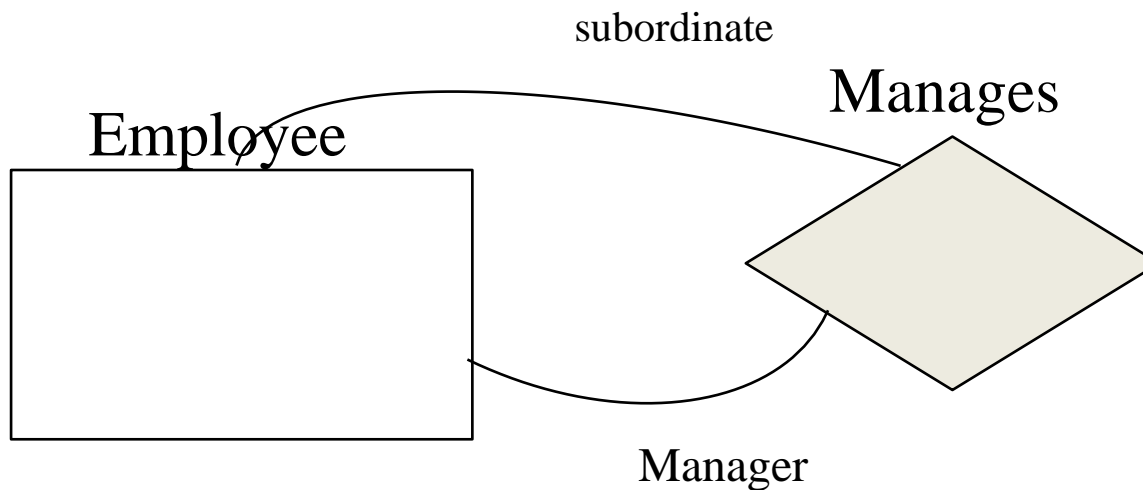
Unary Relationship



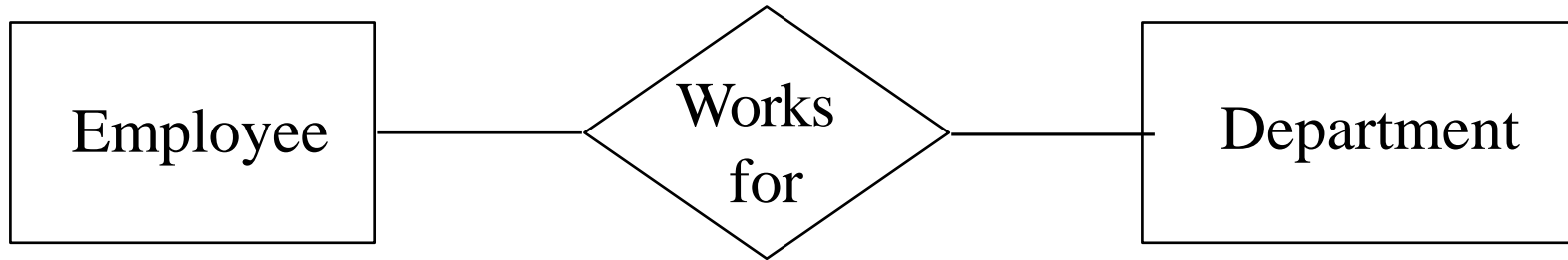
- A unary relationship is represented as a diamond which connects one entity to itself as a loop.
- The relationship above means, some instances of employee manage other instances of Employee.

Role names

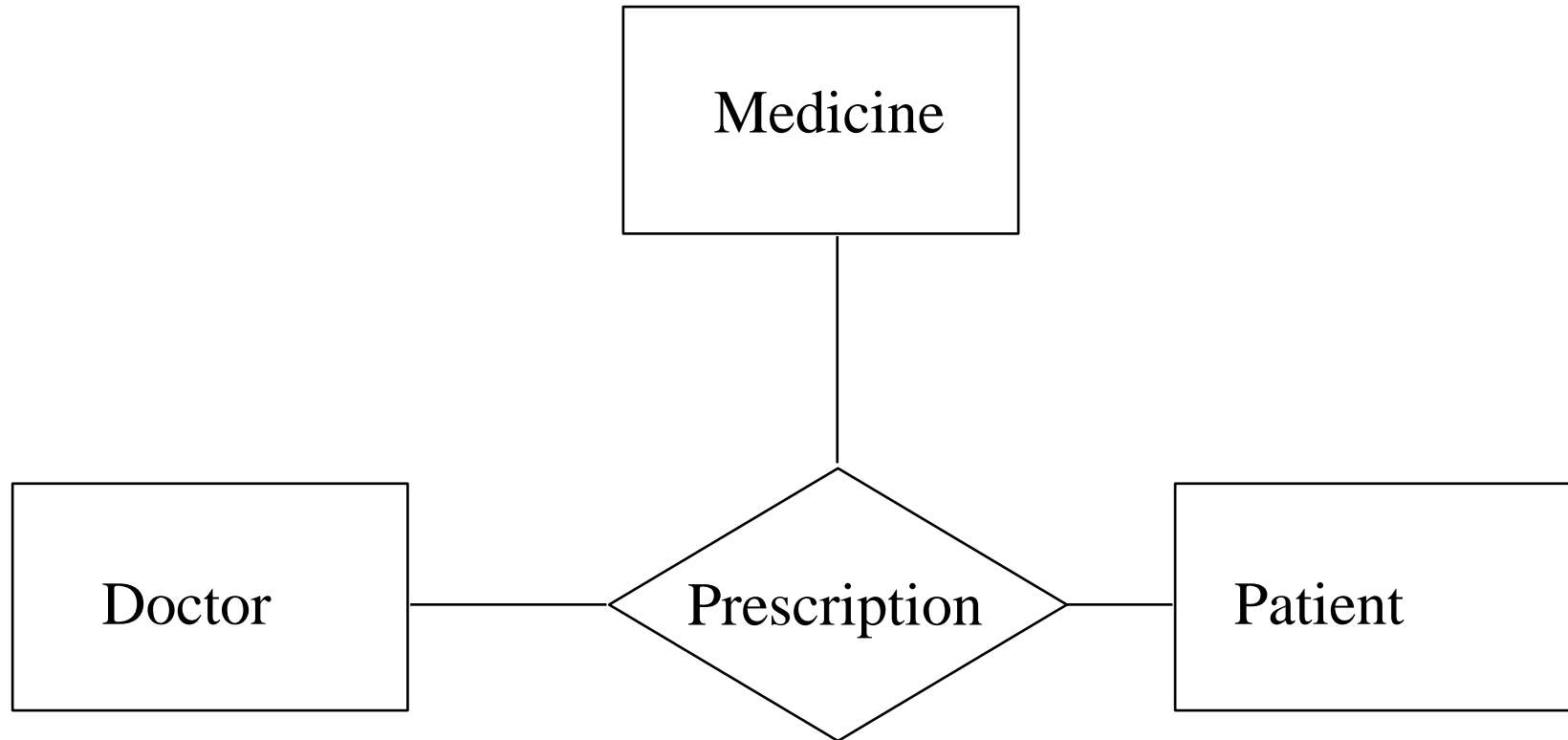
- **Role names** may be added to make the meaning more explicit



Binary Relationship



Ternary Relationship



Cardinality

- Relationships can have different *connectivity*
 - **one-to-one** (1:1)
 - **one-to-many** (1:N)
 - **many-to-One** (M:1)
 - **many-to-many** (M:N)

E.g.:

Employee **head-of** department (1:1)

Lecturer **offers** course (1:N) assuming a course is taught by a single lecturer

Students **chooses** subjects (M:N)

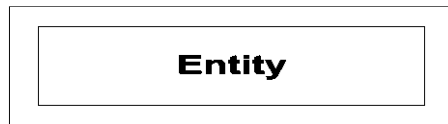
The minimum and maximum values of this connectivity is called the **cardinality of the relationship**

ER Modeling - Notations

ER Modeling -Notations



An **Entity** is an object or concept about which business user wants to store information.



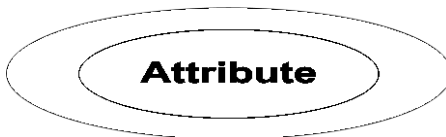
A **Weak Entity** is dependent on another Entity to exist. Example Order Item depends upon Order Number for its existence. Without Order Number it is impossible to identify Order Item uniquely.



Attributes are the properties or characteristics of an Entity



A **Key Attribute** is the unique, distinguishing characteristic of the Entity

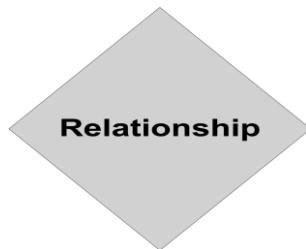


A **Multi-valued Attribute** can have more than one value. For example, an employee Entity can have multiple skill values.

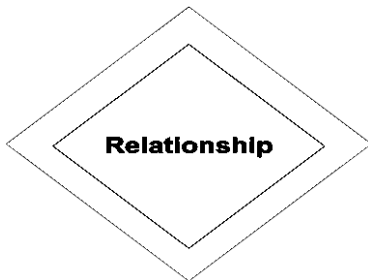
ER Modeling -Notations



A **derived attribute** is based on another attribute. For example, an employee's monthly salary is based on the employee's basic salary and House rent allowance.

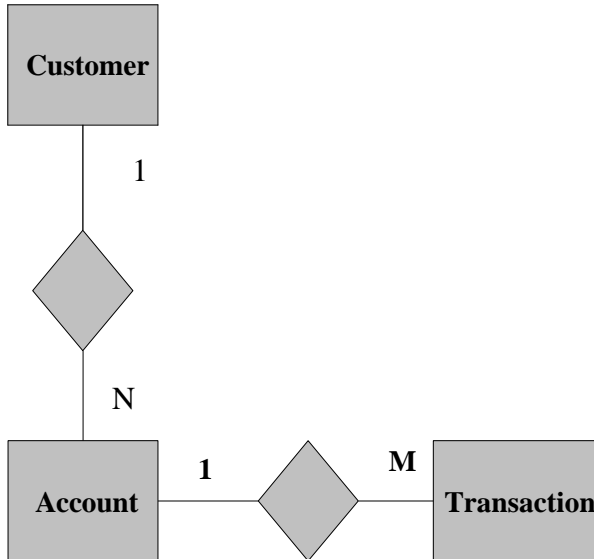


Relationship illustrate how two entities share information in the database structure.

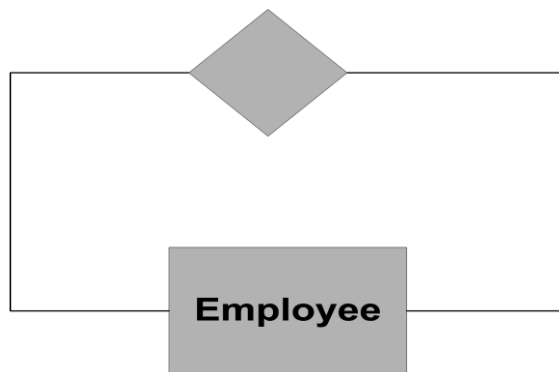


To connect a weak Entity with others, you should use a **weak relationship** notation.

ER Modeling -Notations

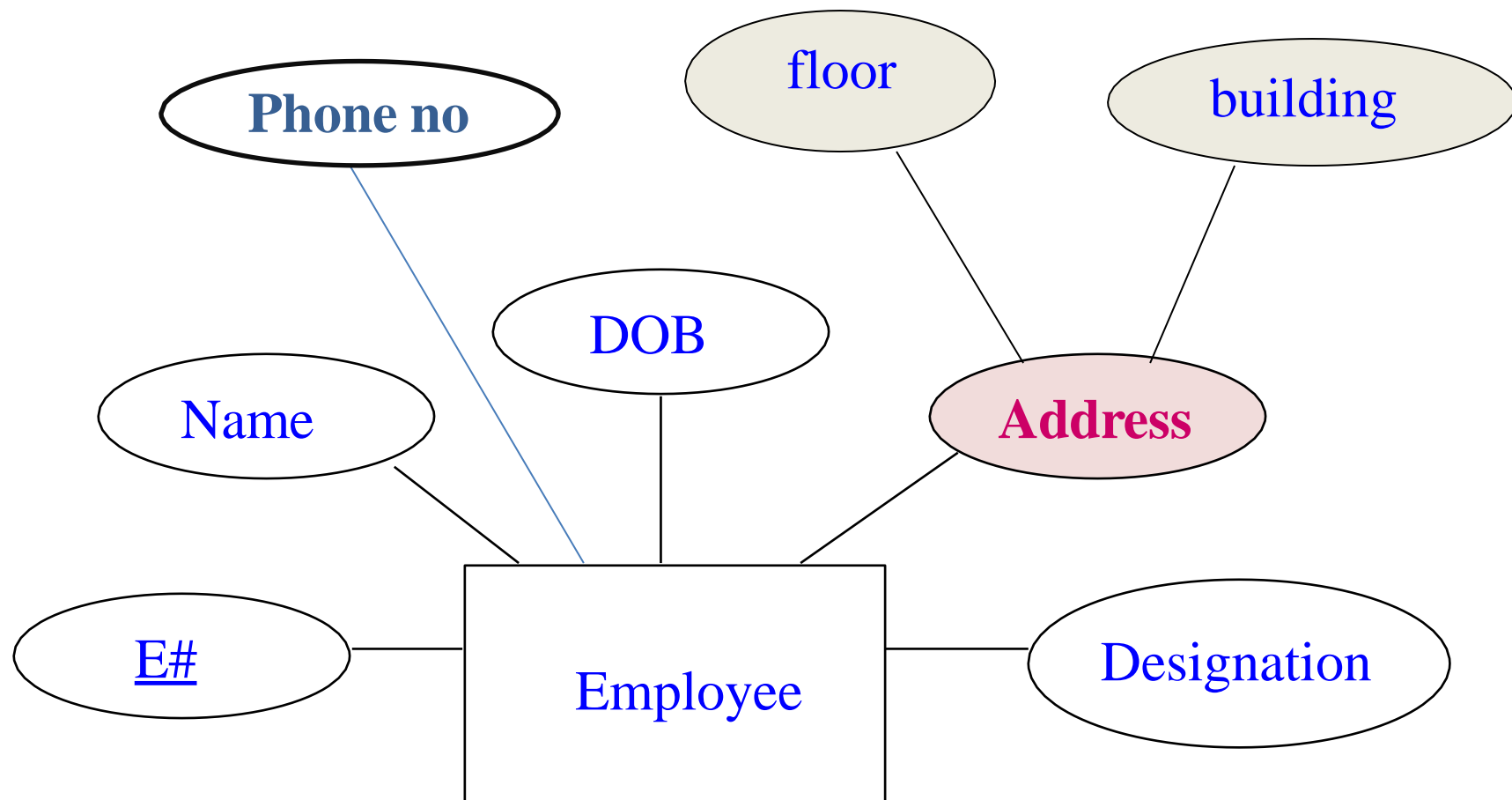


Cardinality specifies how many instances of an Entity relate to one instance of another Entity. M,N both represent ‘MANY’ and 1 represents ‘ONE’ Cardinality



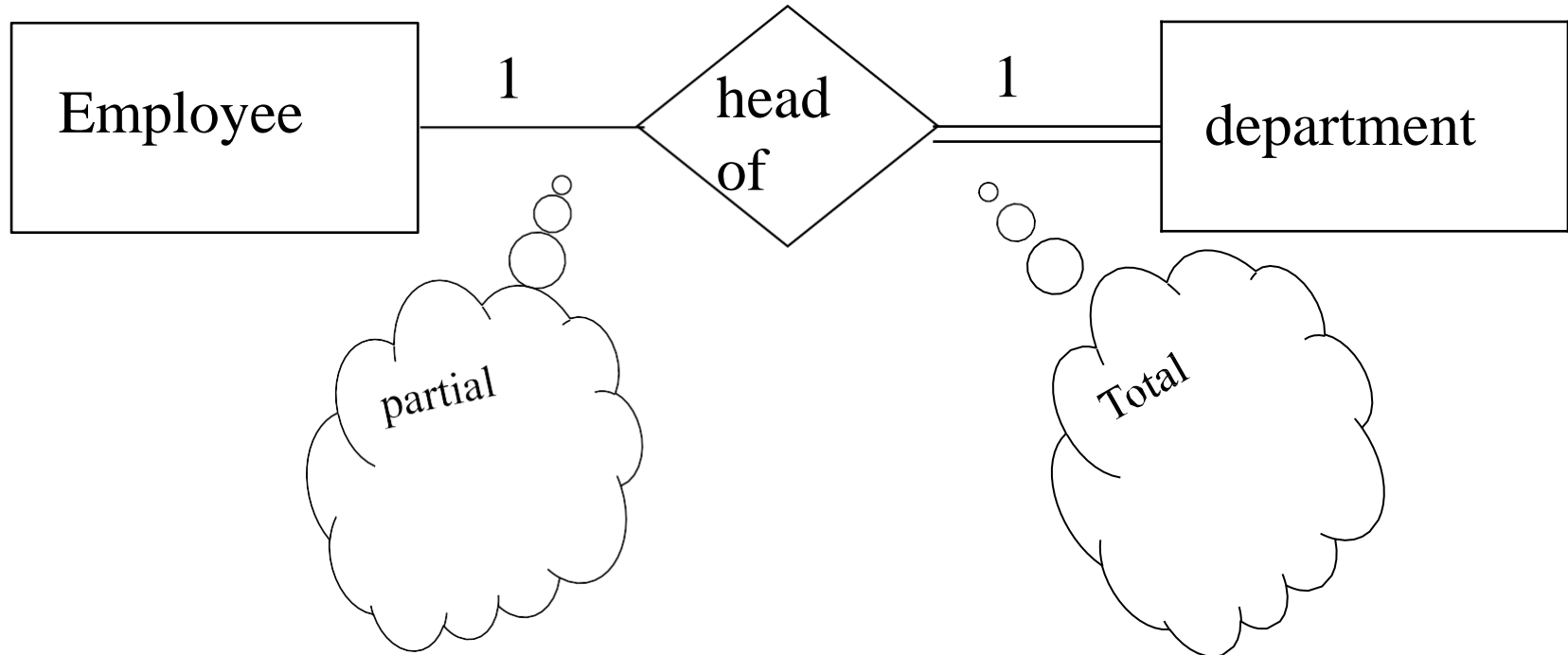
In some cases, entities can be **self-linked**. For example, employees can supervise other employees

Composite attribute

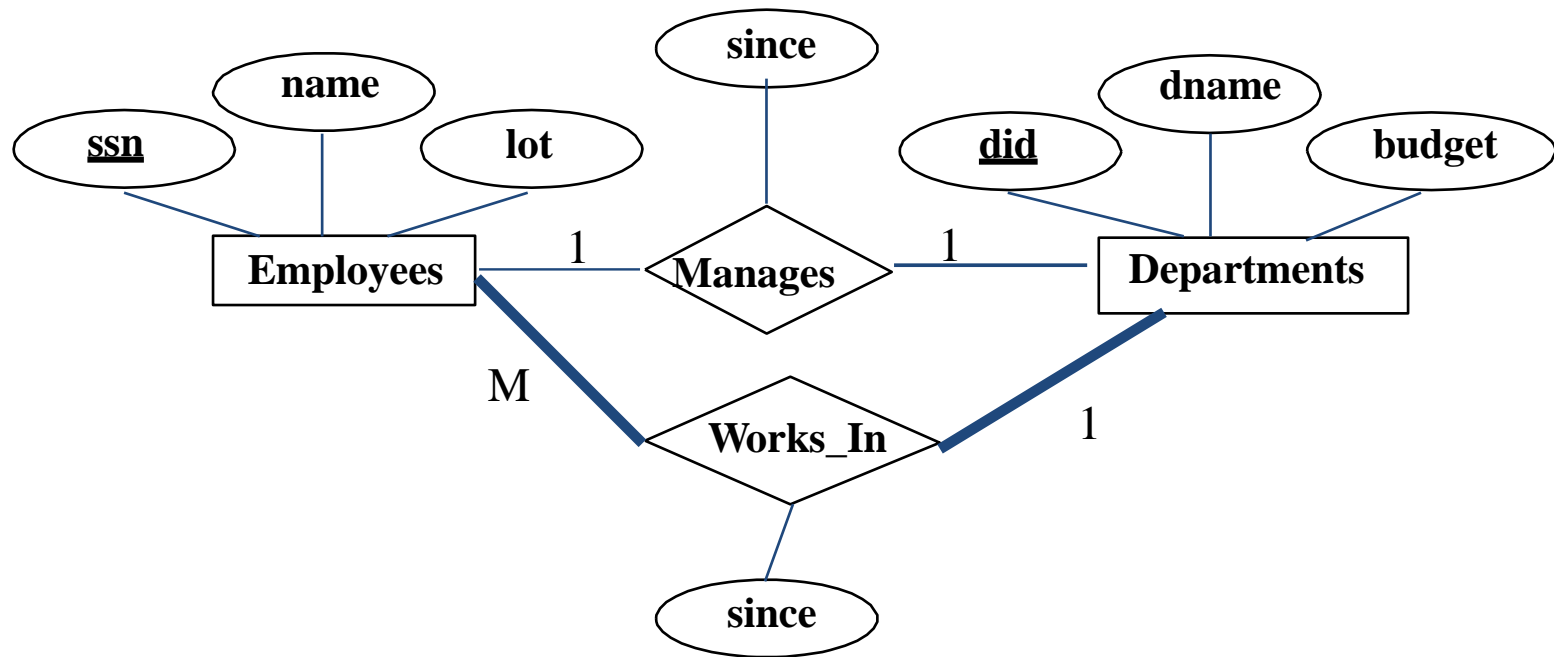


Represented by an ellipse from which other ellipses emanate and represent the component attributes. E.g Address

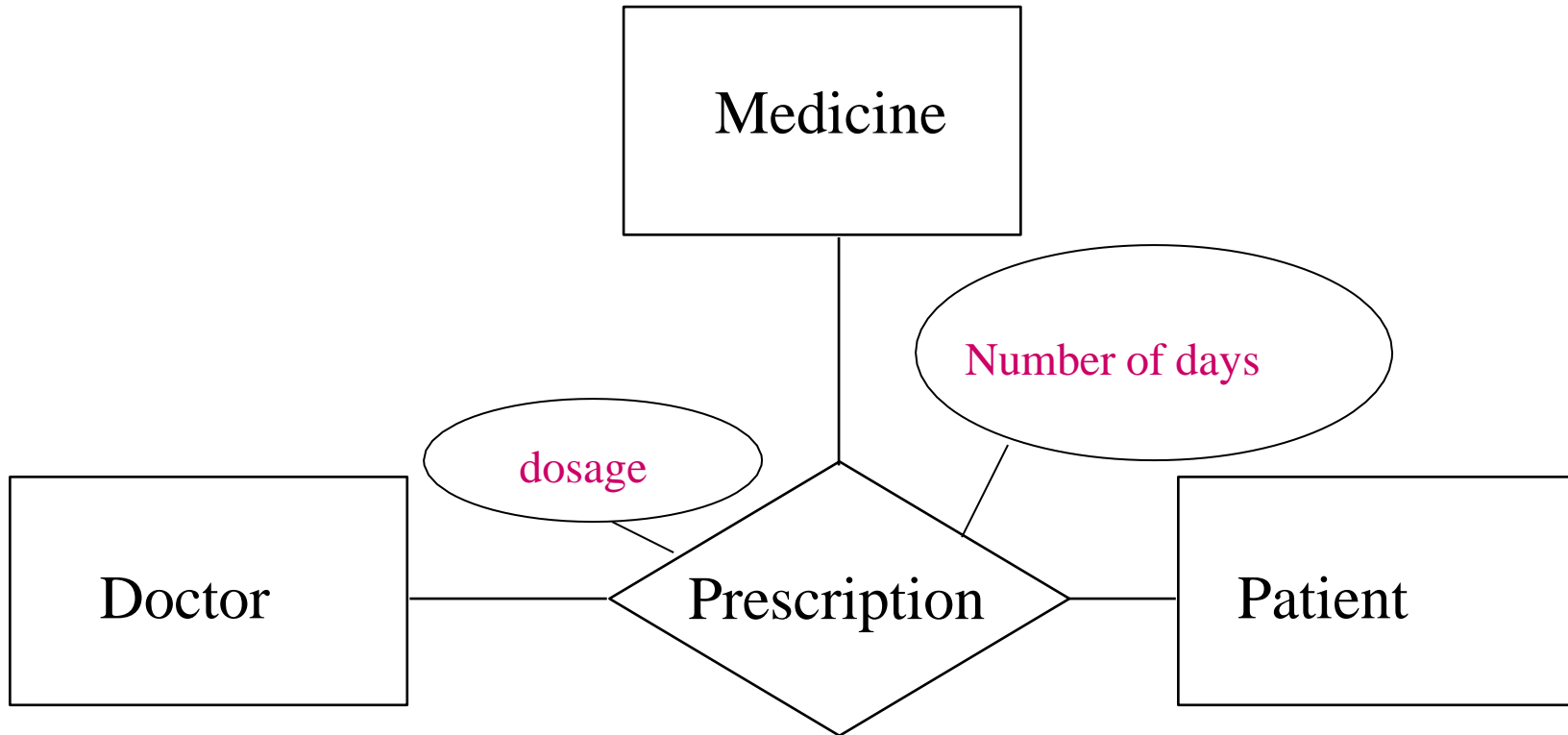
Relationship participation



- All instances of the entity type Employee don't participate in the relationship, Head-of.
- Every employee doesn't head a department. So, employee entity type is said to partially participate in the relationship.
- But, every department would be headed by some employee.
- So, all instances of the entity type Department participate in this relationship. So, we say that it is total participation from the department side.



Attributes of a Relationship



These attributes best describe the relationship prescription rather than any individual entity Doctor, Patient or Medicine. Such relationships are also known as associative entity.

Case Study – ER Model For a college DB

Assumptions :

- A college contains many departments
- Each department can offer any number of courses
- Many instructors can work in a department
- An instructor can work only in one department
- For each department there is a Head
- An instructor can be head of only one department
- Each instructor can take any number of courses
- A course can be taken by only one instructor
- A student can enroll for any number of courses
- Each course can have any number of students

Steps in ER Modeling

1. Identify the Entities
2. Find relationships
3. Identify the key attributes for every Entity
4. Identify other relevant attributes
5. Draw complete E-R diagram with all attributes including Primary Key
6. Review your results with your Business users

Step 1: Identify the Entities

- DEPARTMENT
- STUDENT
- COURSE
- INSTRUCTOR

Steps in ER Modeling

Step 2: Find the relationships

- One course is enrolled by multiple students and one student enrolls for multiple courses, hence the cardinality between course and student is Many to Many.
- The department offers many courses and each course belongs to only one department, hence the cardinality between department and course is One to Many.
- One department has multiple instructors and one instructor belongs to one and only one department, hence the cardinality between department and instructor is one to Many.
- Each department there is a “Head of department” and one instructor is “Head of department”, hence the cardinality is one to one.
- One course is taught by only one instructor, but the instructor teaches many courses, hence the cardinality between course and instructor is many to one.

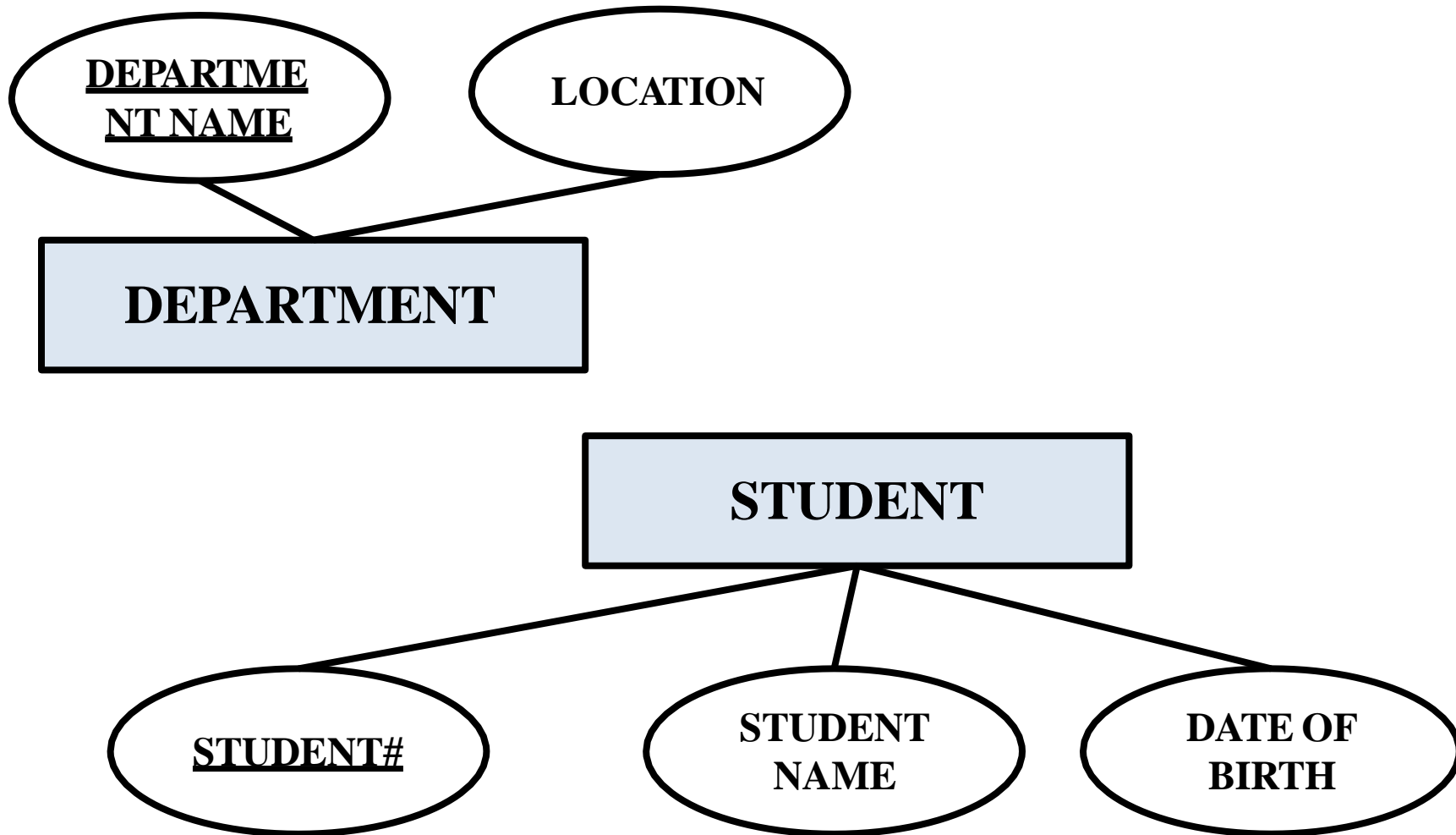
Step 3: Identify the key attributes

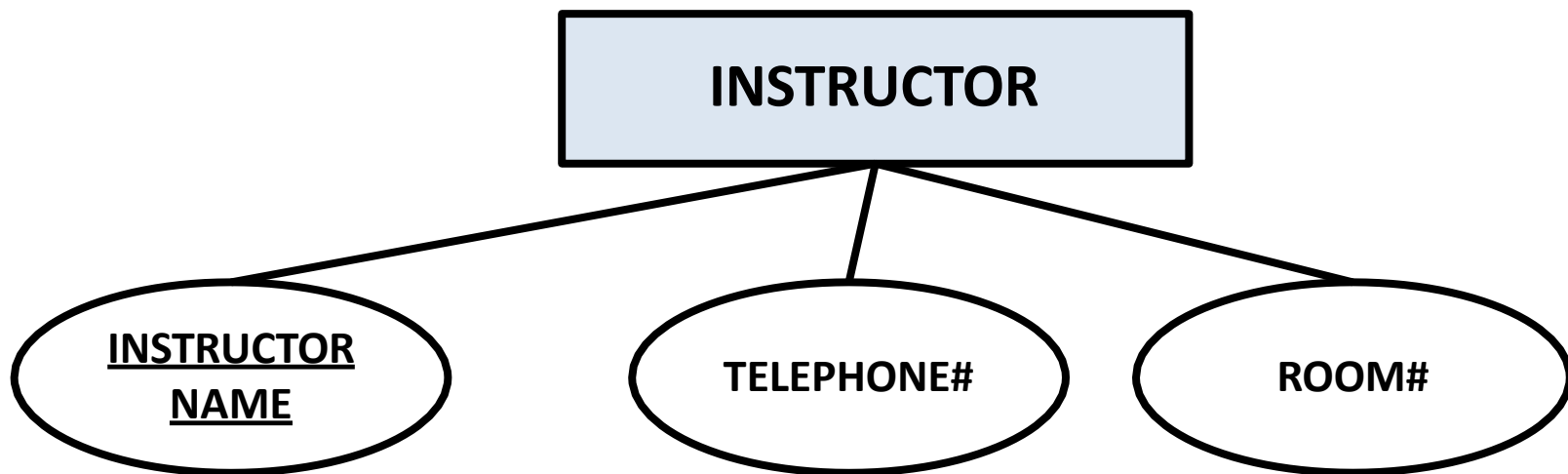
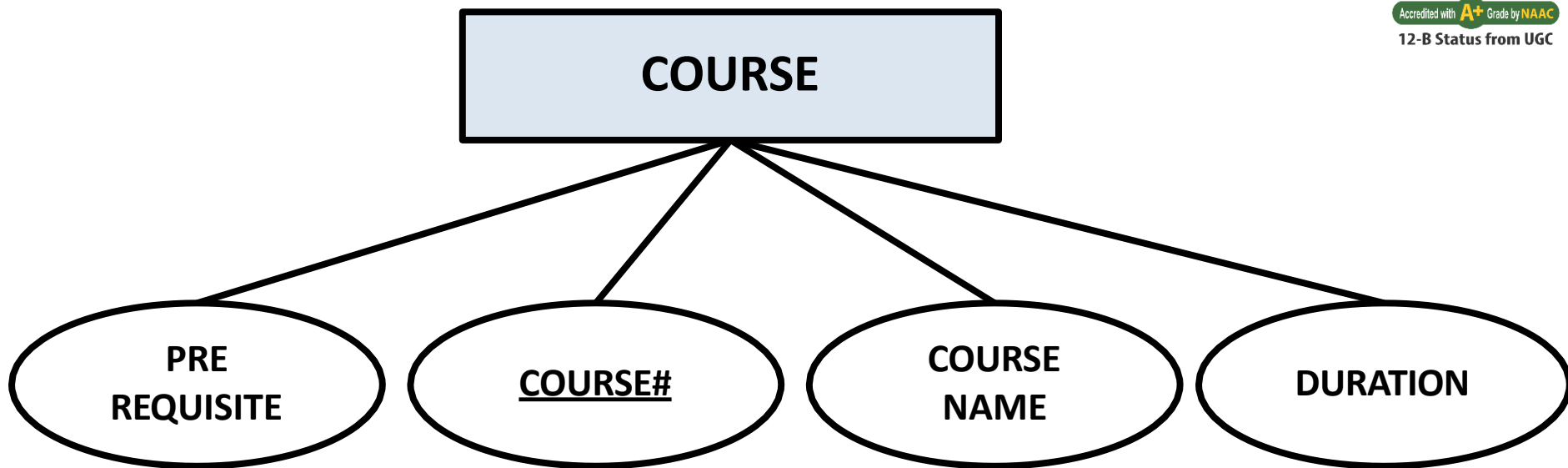
- Deptname is the key attribute for the Entity “Department”, as it identifies the Department uniquely.
- Course# (CourseId) is the key attribute for “Course” Entity.
- Student# (Student Number) is the key attribute for “Student” Entity.
- Instructor Name is the key attribute for “Instructor” Entity.

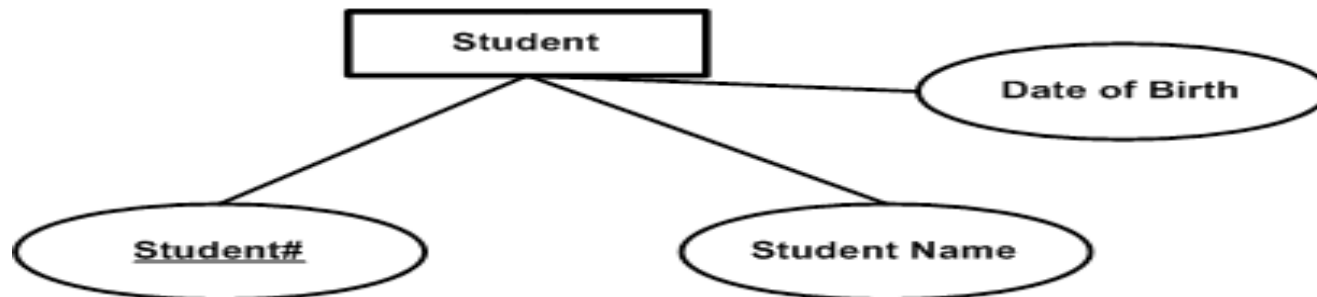
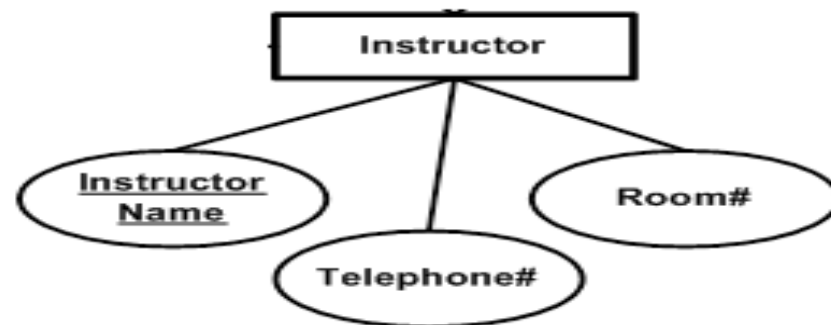
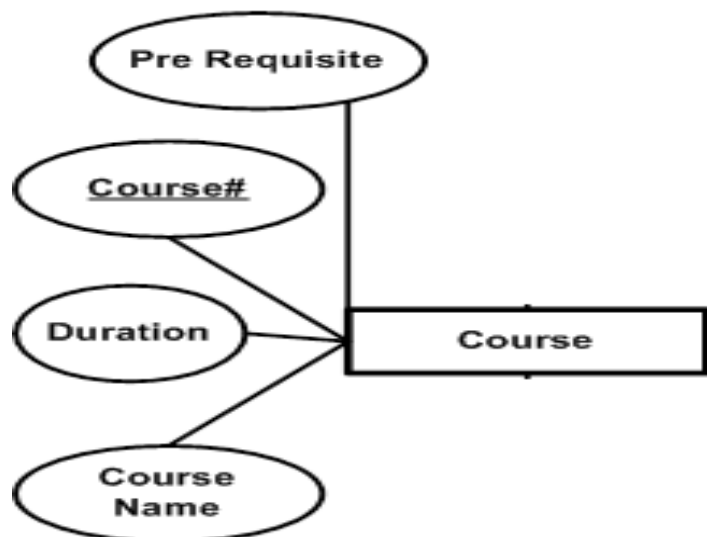
Step 4: Identify other relevant attributes

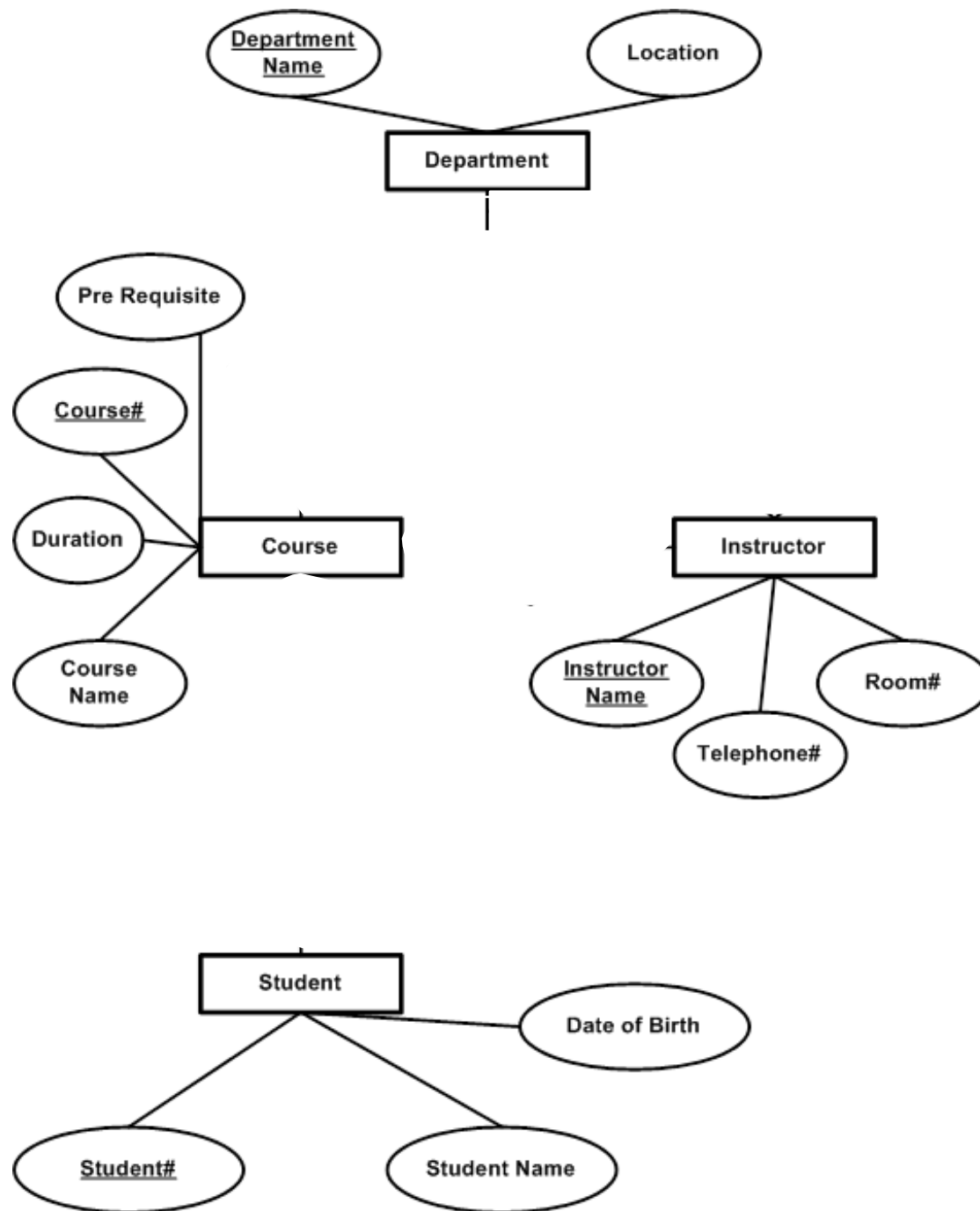
- For the department entity, the relevant attribute is location
- For course entity, course name, duration, prerequisite
- For instructor entity, room#, telephone#
- For student entity, student name, date of birth

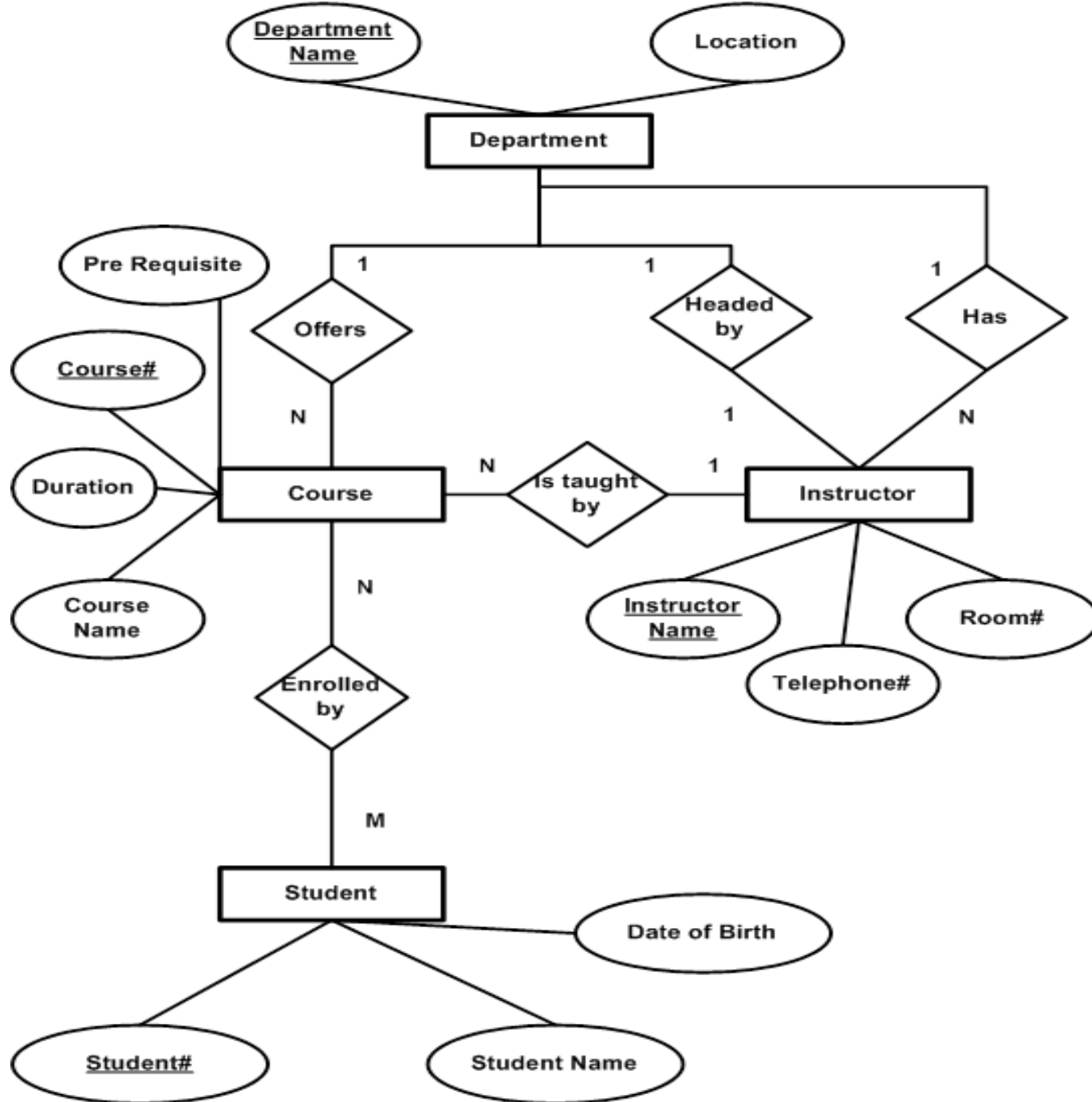
Entities



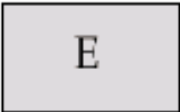

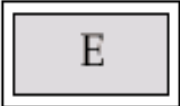
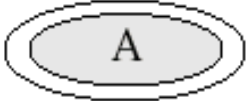



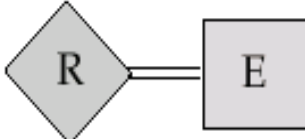

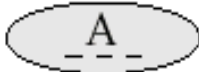




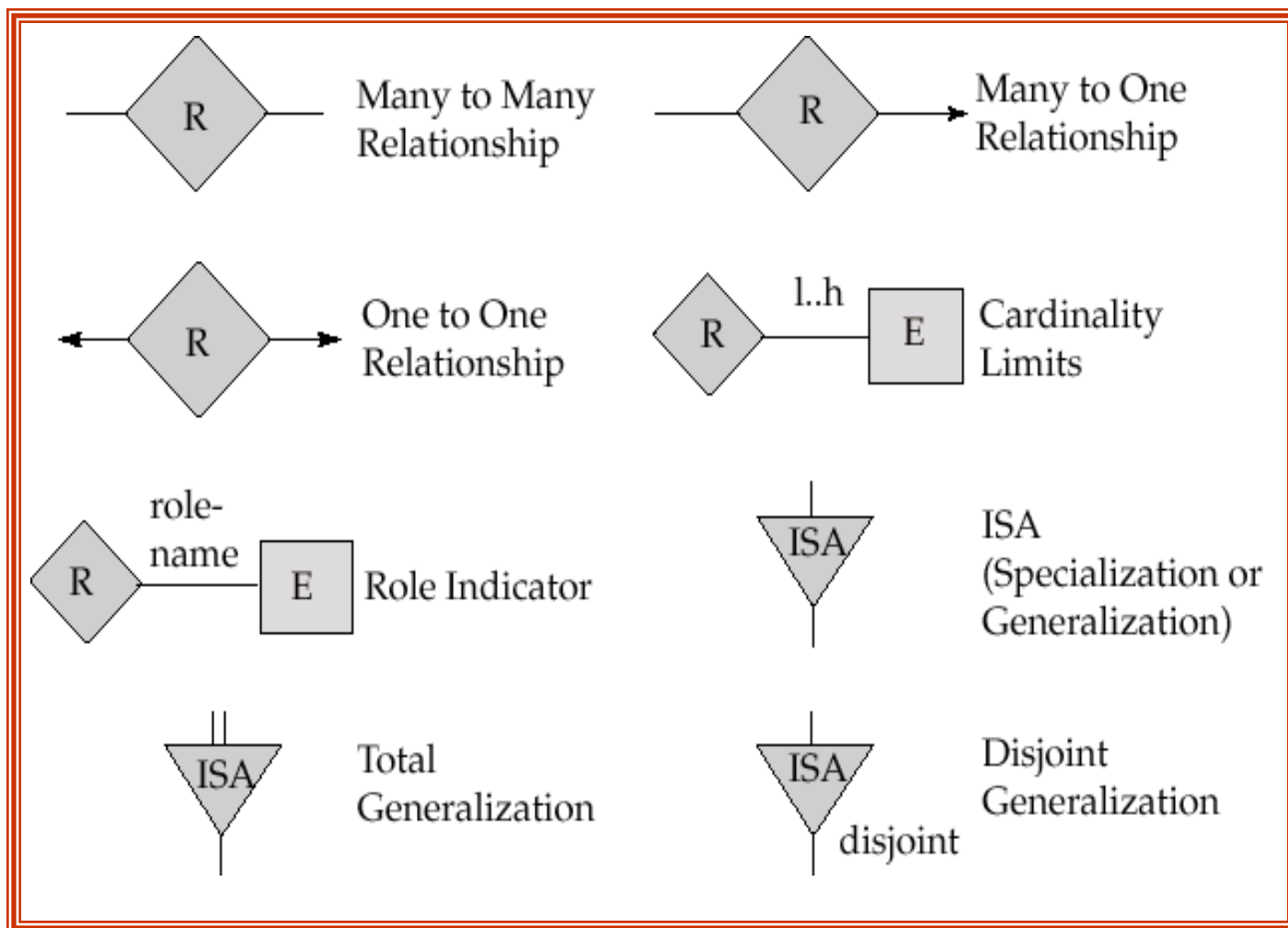




Notations

	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

Notations



University Roll No.....

First Term Examination, Even Semester 2019-20

Programme: B.Tech Year: II Sem:IV

Database Management Systems: BCSC 0003

Time: 1 Hour

Maximum Marks:15

Section – A

3 x 2 = 6 Marks

- I. List four significant differences between a file-processing system and a DBMS.
- II. What is the importance of data independence in the context of DBMS? Explain with suitable examples.
- III. Enlist various categories of End Users in DBMS.

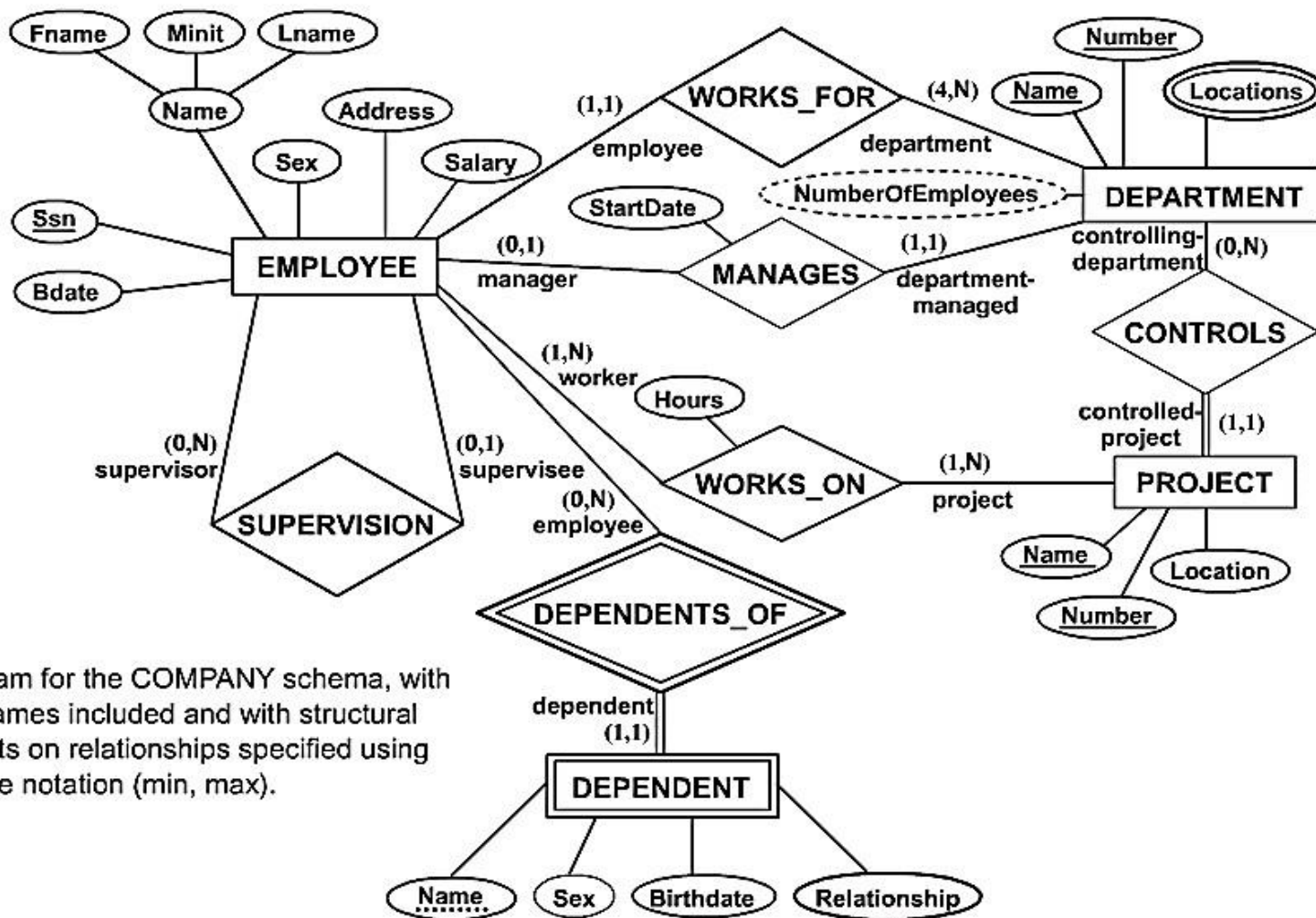
Section – B

3 x 3 = 9 Marks

- I. Construct an ER Diagram for Company having following details:
 - Company organized into DEPARTMENT. Each department has unique name and a particular employee who manages the department. Start date for the manager is recorded. Department may have several locations.
 - A department controls a number of PROJECT. Projects have a unique name, number and a single location.
 - Company's EMPLOYEE name, ssno, address, salary, sex and birth date are recorded. An employee is assigned to one department, but may work for several projects (not necessarily controlled by her dept). Number of hours/week an employee works on each project is recorded; The immediate supervisor for the employee.
 - Employee's DEPENDENT are tracked for health insurance purposes (dependent name, birthdate, relationship to employee).
- II. What are the different levels of abstraction in the DBMS? Explain each level in detail.

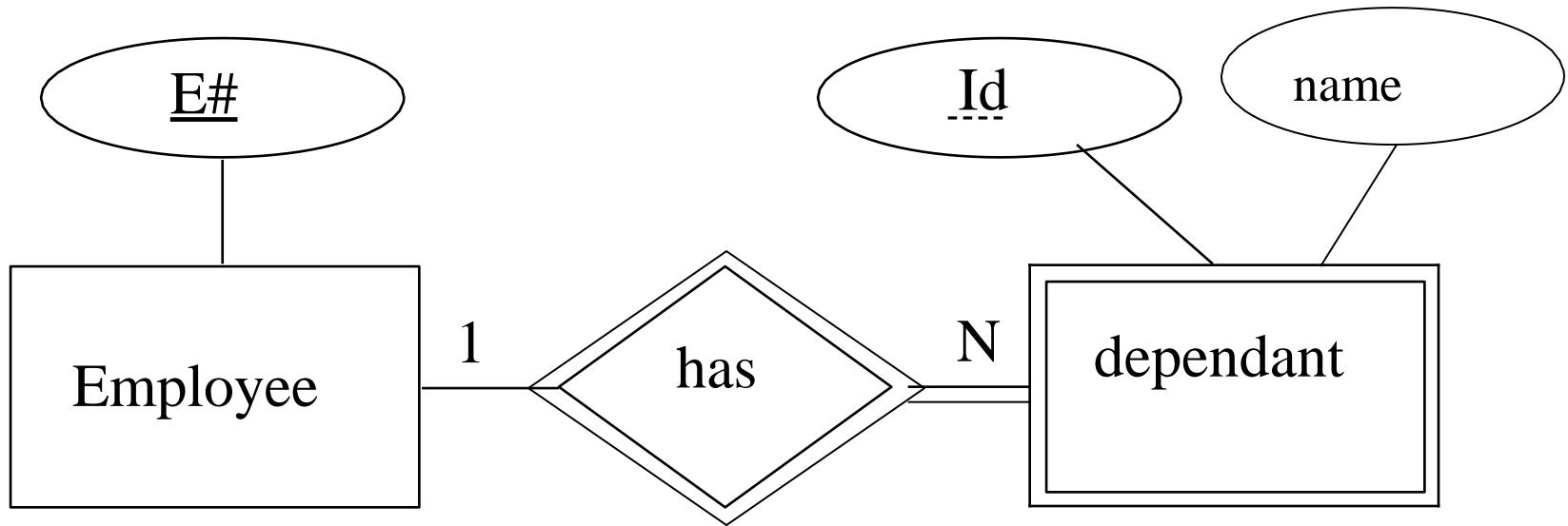
COMPANY ER Schema Diagram using (min, max) notation

Alternative ER Notations



ER diagram for the COMPANY schema, with all role names included and with structural constraints on relationships specified using alternative notation (min, max).

Weak Entity



The dependant entity is represented by a double lined rectangle and the identifying relationship by a double lined diamond

- Cardinality:- Number of tuples in a relation (table).
- Degree:- Number of attributes in a relation (table).



ENHANCED ENTITY-RELATIONSHIP (EER) MODEL



- Introduction to EER Model

- Since the late 1970 there has been an increase in the emergence of new database applications with more demanding requirements.
- Basic concepts of the ER model are not sufficient to represent the requirements of the newer, more complex applications.
- To represent these requirements additional 'semantic' modeling concepts are needed.
- These semantic concepts are incorporated into the original ER model and became **Enhanced Entity-Relationship** (EER) model.
- Additional concepts of EER model include:
 - Specialization
 - Generalization
 - Categorization
- EER diagrams are similar to the **class diagrams** used in the OO model.



- Concepts of EER Model ...

- The EER model includes all the concepts used in the ER model:
 - Entity types
 - Attributes
 - Relationship types

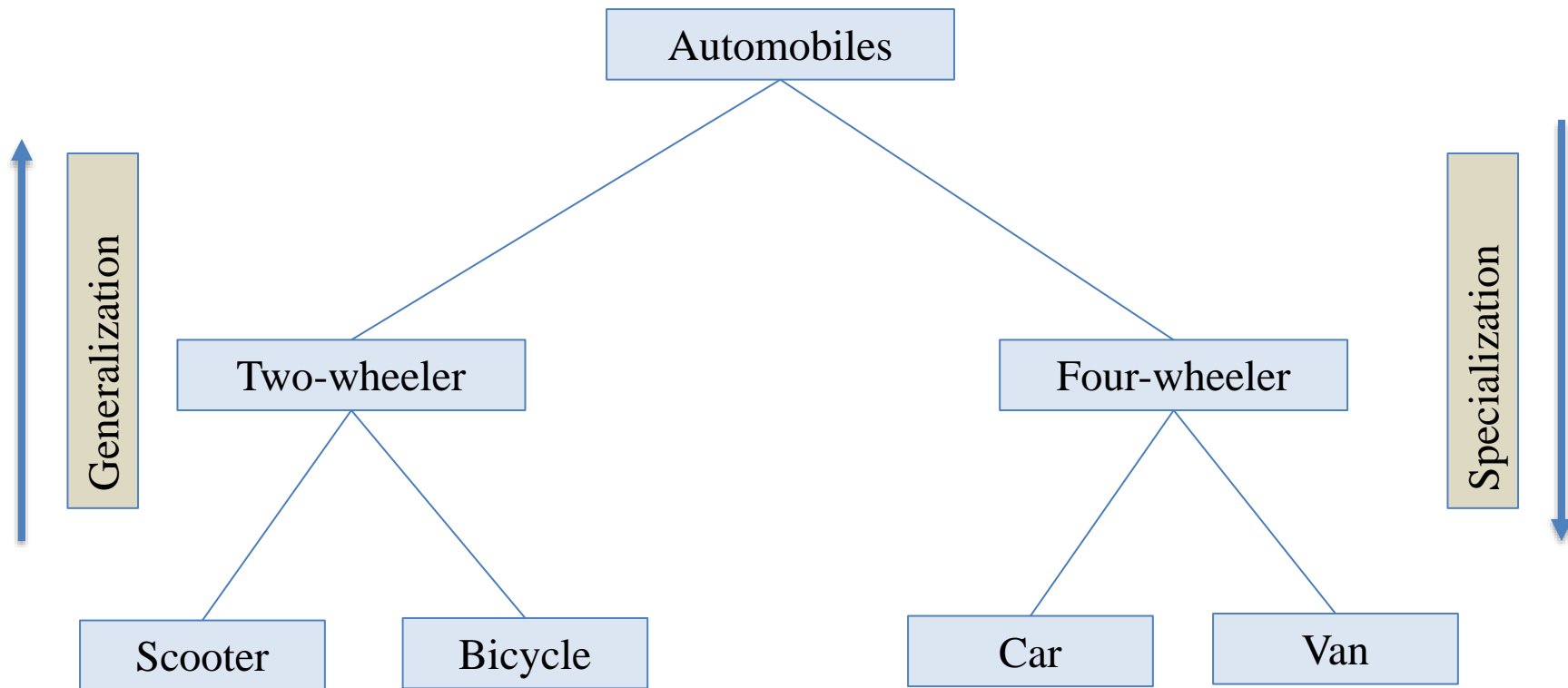
- In addition, the EER model includes the following concepts, which come from the OO model:
 - Supreclass/subclass relationships
 - Attribute inheritance
 - Specialization
 - Generalization
 - Categorization



... - Concepts of EER Model

- **Subclass**: is an entity type that has a distinct role and is also a member of the superclass.
- **Superclass**: is an entity type that includes distinct subclasses that require to be represented in a data model.
- **Attribute inheritance**: is an entity in a subclass may possess subclass specific attributes, as well as those associated with the superclass.
- The relationship between a superclass and any of its subclasses is called the **superclass/subclass relationship**.

Generalization and Specialization



Generalization: In generalization, a number of entities are brought together into one generalized entity based on their similar characteristics. For e.g. 2-wheeler and 4-wheeler can all be generalized as Automobiles.

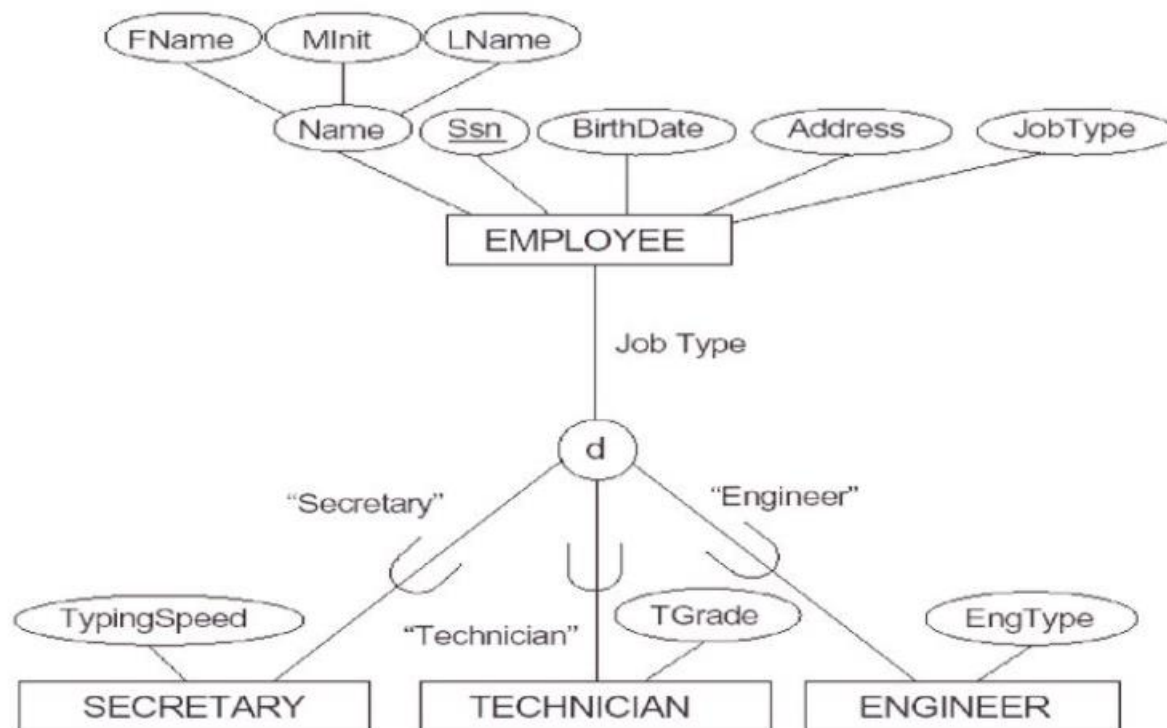
Specialization: A group of entities is divided into sub-groups based on their characteristics. Specialization is just opposite of generalization.



- Specialization

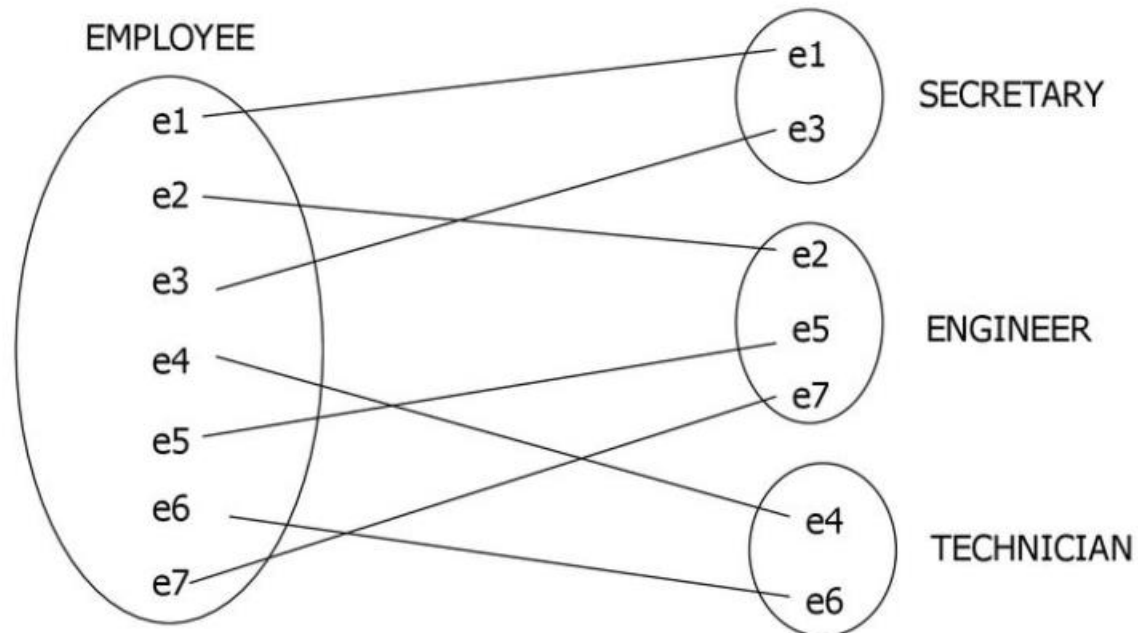
- A **Specialization** is the process of defining a set of subclasses of an entity type.
- The specialization of an entity type allows us to do the following:
 - Define a set of subclasses of an entity type.
 - Establish additional attributes with subclasses
 - Establish additional relationship types between some subclasses and other entity types or other subclasses.
- The figure in the next slide shows an example of a superclass/subclass relationship between the entity type EMPLOYEE and some of its classes.
 - In this figure, local attributes are attached to subclasses
 - Common attributes are attached to the superclass
 - Some relationships between subclasses are shown.

-- Example: Specialization



... - Specialization

- The following Figure shows an example of some instances of the specialization of the superclass EMPLOYEE into the set of subclasses {SECRETARY, ENGINEER, TECHNICIAN}

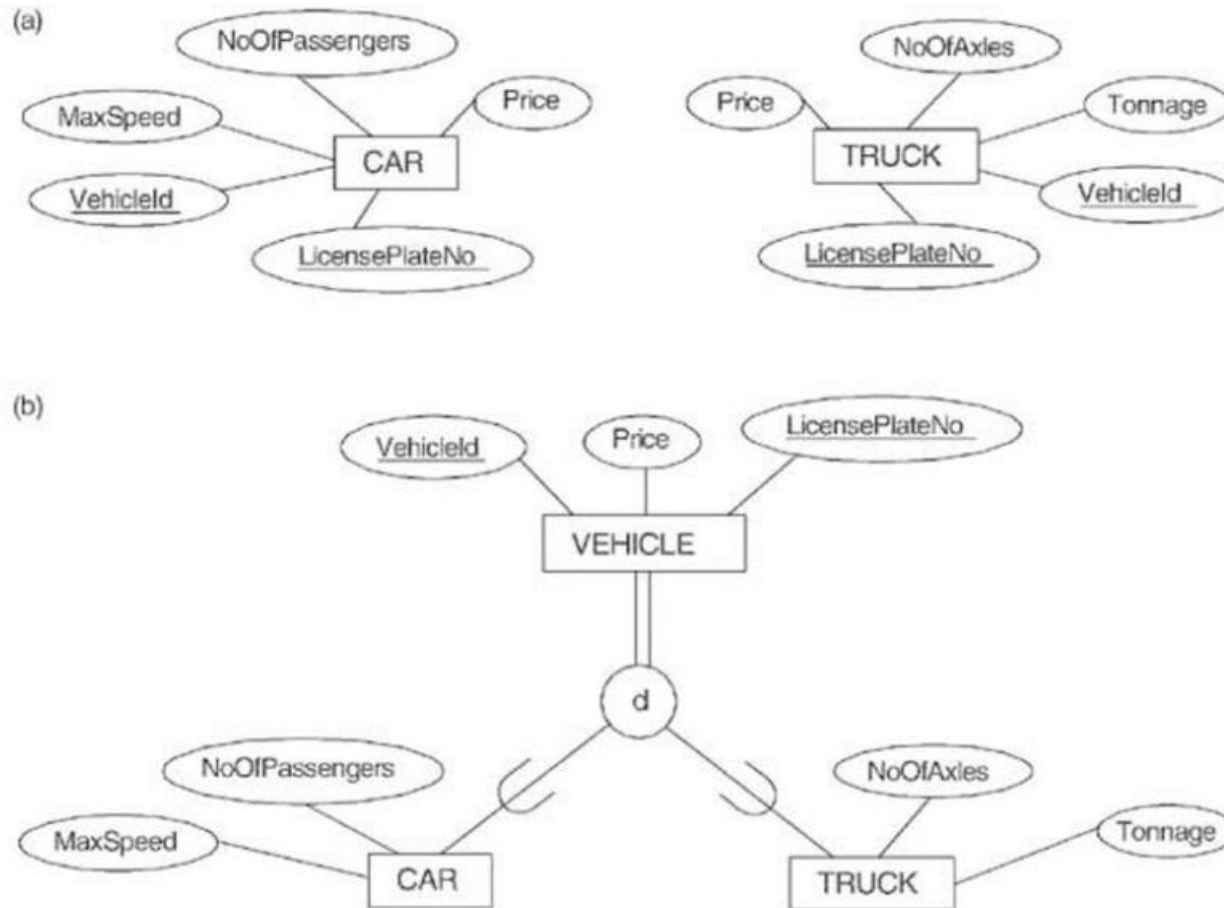




- Generalization

- **Generalization** is the process of defining a superclass of many entity types.
- Generalization can be considered as the reverse of specialization.
 - Example: We can view EMPLOYEE as a generalization of SECRETARY, TECHNICIAN, and ENGINEER.
- The **figure** in the next slide shows an example of generalizing the two entity types (subclasses) CAR and TRUCK into the VEHICLE entity type (superclass).

Figure 4.3 Examples of generalization. (a) Two entity types CAR and TRUCK. (b) Generalizing car and TRUCK into VEHICLE.



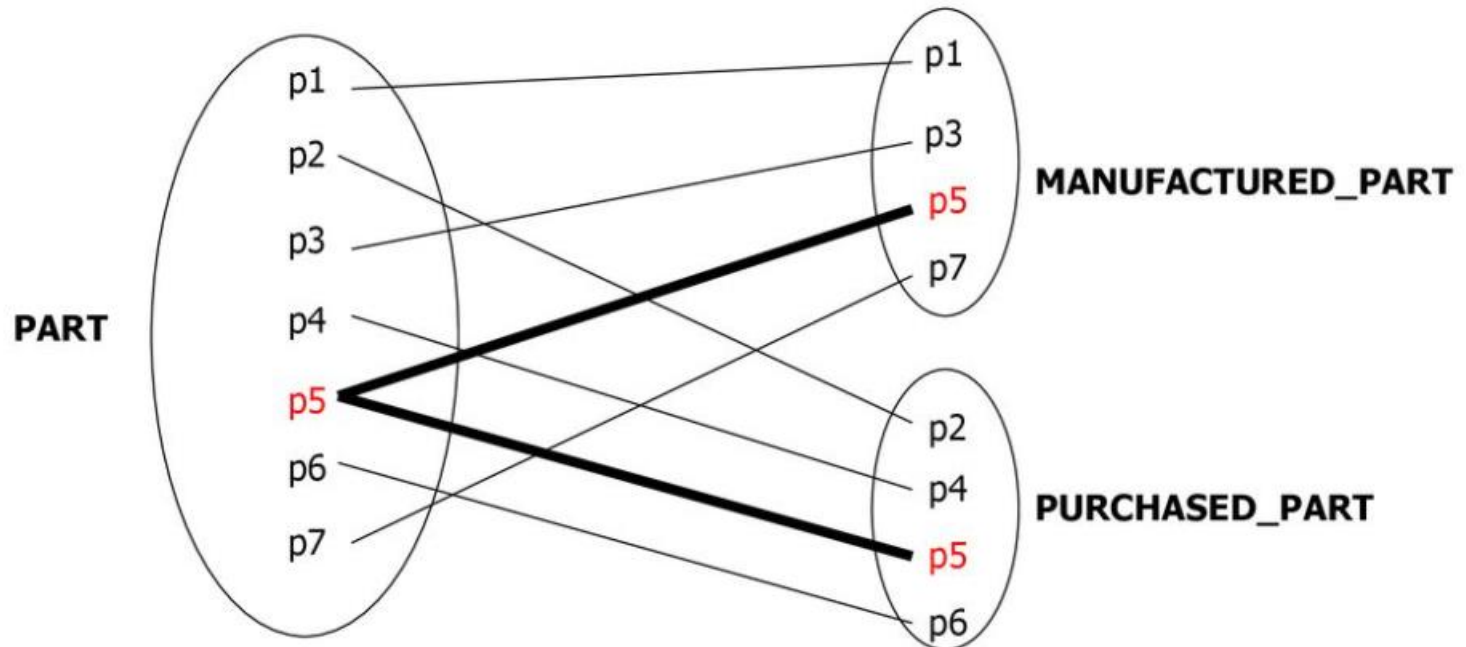


-- Disjointness constraint

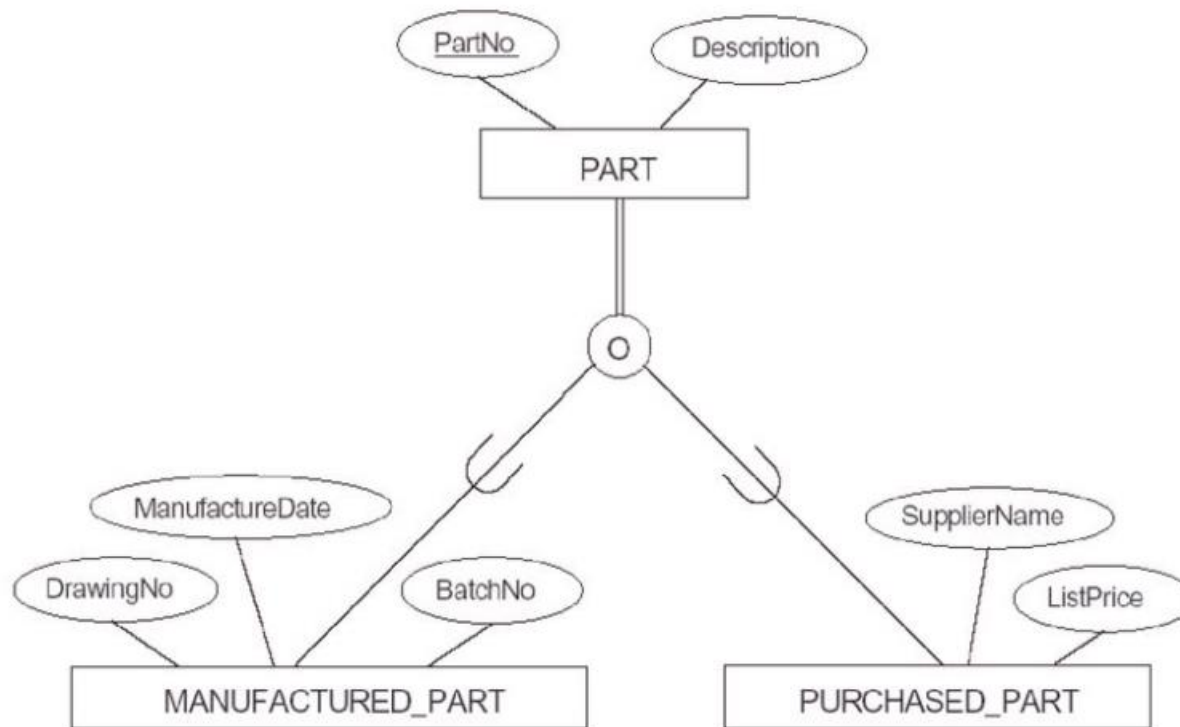
- Look to the superclass/subclass relationship shown in this [figure](#). We can see that the subclasses are disjoint, i.e. every employee entity will belong to only one of the subclasses, SECRETARY or TECHNICIAN or ENGINEER.
- This is represented in [this diagram](#) by the symbol **d** which stands for disjoint.
- However, there are some superclasses that can be specialized into subclasses that may overlap as the example shown the following [figure](#). Here a PART entity can belong to one or more of the entities MANUFACTURED or PURCHASED.
- This is represented in the [diagram](#) by the symbol **o** which stands for overlapping.



... - Overlapping Subclasses ...



--- Overlapping subclasses



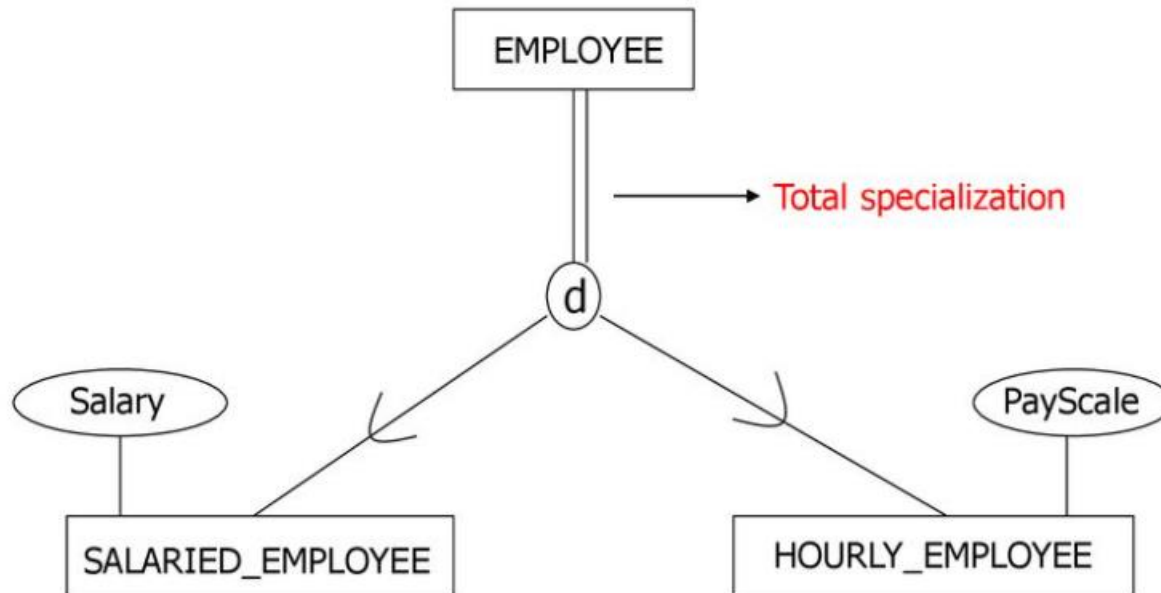


-- Completeness constraints

- A **total specialization** constraint specifies that every entity in the superclass must be a member of some subclass.
- For example, if every employee must be either SALARIED_EMPLOYEE or HOURLY_EMPLOYEE then the specialization {SALARIED_EMPLOYEE, HOURLY_EMPLOYEE} is total specialization.
- This constraint is shown in the figure which is in the next slide by using **double lines** to connect the superclass as shown in the figure.



--- Example of Total Specialization



Thank
you!

