Entity-Relationship Model
Database Design – Part 2

Dr. Jeevani Goonetillake

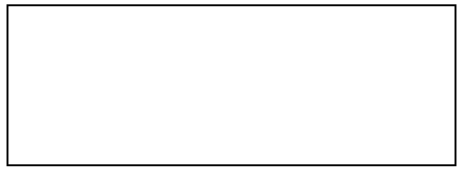
UCSC

ER Modeling

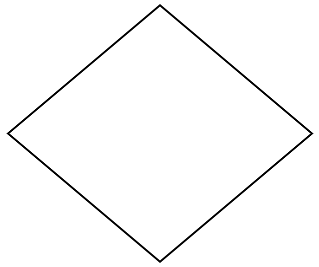
Many notations for ER diagrams are in use; We use the notation proposed by (Chen, 1976).

Notations

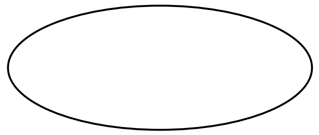
Entity



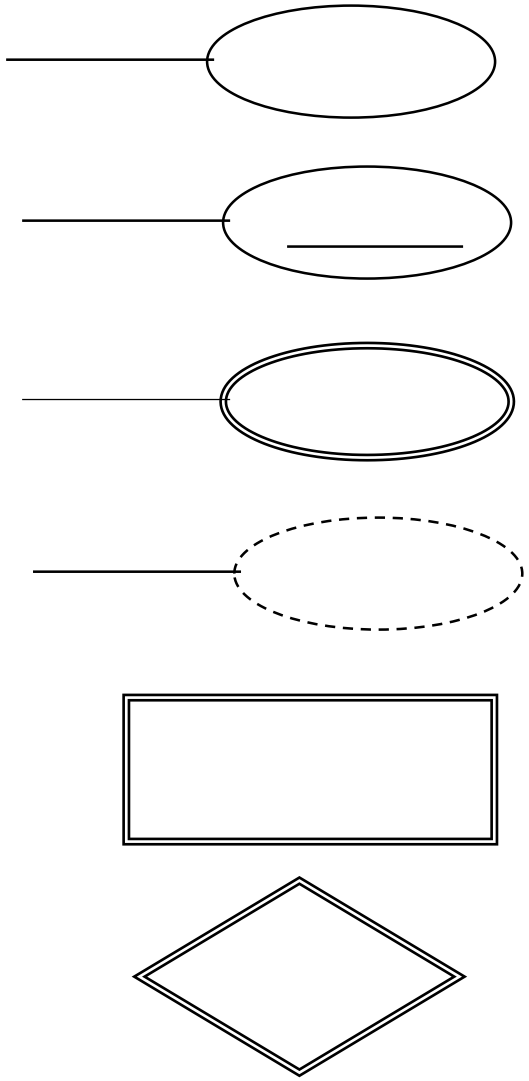
Relationship



Attribute



ER Modeling



Attribute

Key Attribute

Multivalued attributes

Derived Attribute

Weak Entity

Identifying Relationship

Initial Conceptual Design of the Company Database

- **Department**

  Name, Number,{Locations}, Manager, ManagerStartDate
- **Project**

  Name, Number , Location , ControllingDepartment
- **Employee**

  Name(Fname,Lname),Emp_id, Gender, Address, Salary,

  BirthDate, Department, Supervisor,{WorksOn(Project,Hours)}

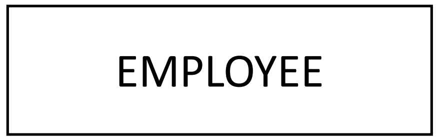
Initial Conceptual Design of the Company Database

- **Dependent**

  Employee, dependent Name, Gender,   BirthDate, Relationship

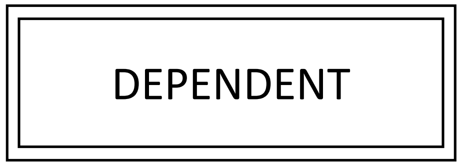
Strong Entity Types

- Regular entity types that do have a key attribute—are called **strong entity types**.
- Strong entity type exists independently of other entity types.



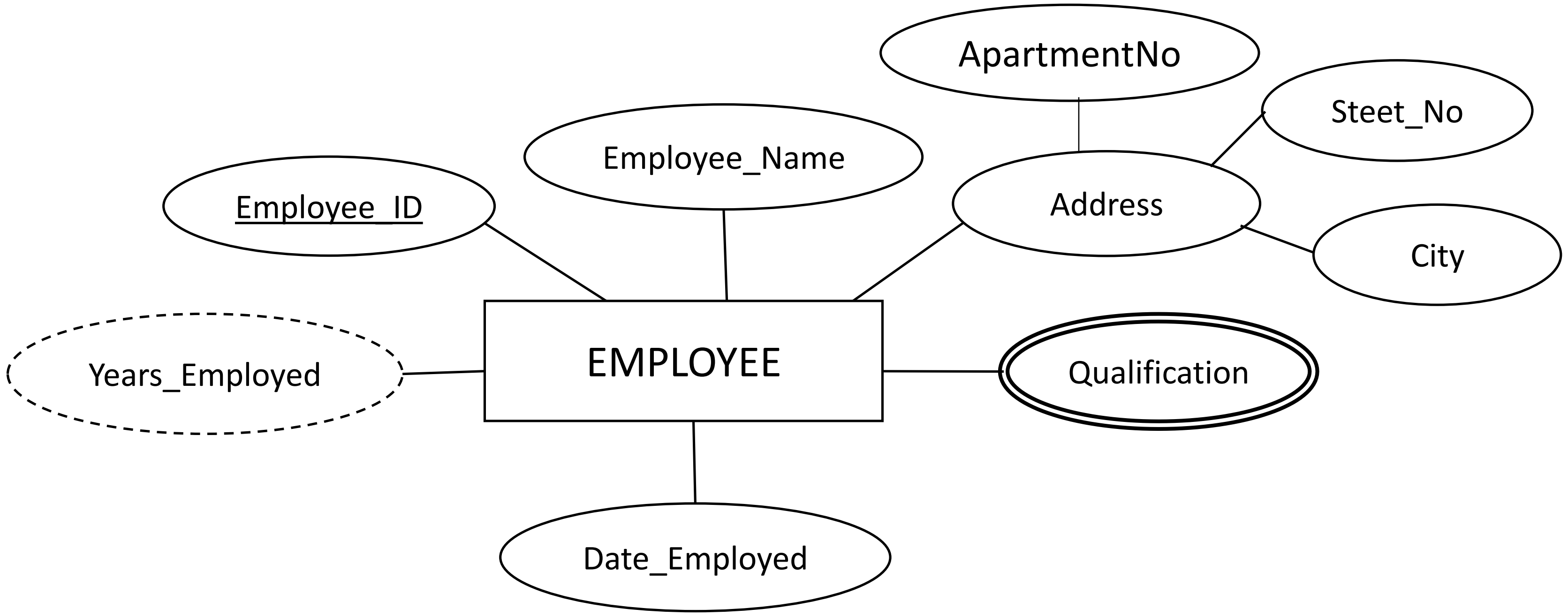
Weak Entity Types

- Entity types that do not have key attributes of their own are called **weak entity types**.
- Entities belonging to a weak entity type are identified by being related to specific entities from another entity type in combination with one of their attribute values.
- This other entity type is called the **identifying** or **owner entity type**.



Attribute Representation

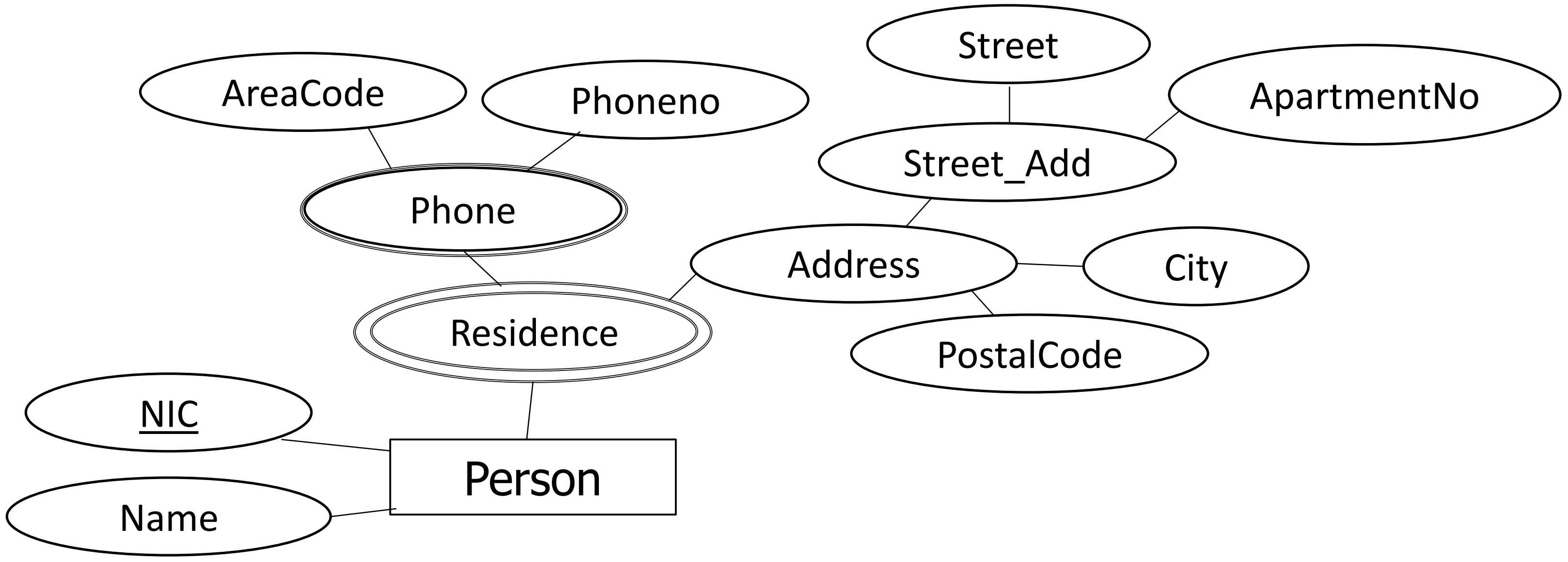
Employee entity with a key attribute (*Employee_ID*), multivalued attribute (*Qualification*), derived attribute (*Years_Employed*) and composite attribute (*Address*).



Attribute Representation

If a person can have more than one residence and each residence can have a single address and multiple phones, an attribute Address_phone for a person can be specified. Both Phone and Address are themselves composite attributes.

Residence { Address_phone ({Phone(Area_code, Phone_number)}, Address(Street_address (Number, Street, Apartment_number),City, Postal_Code))}

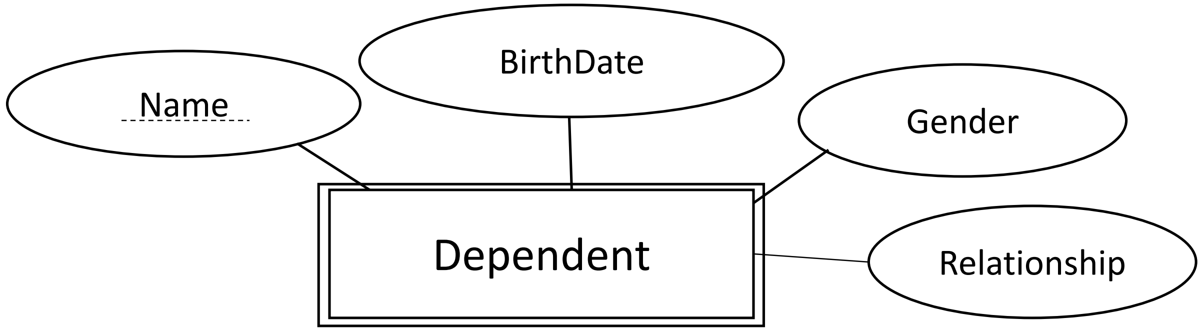


Weak Entity Types

- Weak entity types can sometimes be represented as complex (composite, multivalued) attributes.
- In the given COMPANY scenario, it would be possible to specify a multivalued attribute *Dependents* for EMPLOYEE, which is a multivalued composite attribute with the component attributes *Name, Birth_date, Gender*, and *Relationship*. The choice of which representation to use is made by the database designer.
- One criterion that may be used is to choose the weak entity type representation if the weak entity type participates independently in relationship types other than its identifying relationship type.

Weak Entity Types

- A weak entity type normally has a partial key, which is the attribute that can uniquely identify weak entities that are related to the same owner entity.
- In the given example, if we assume that no two dependents of the same employee ever have the same name (may be the first name), the Name of DEPENDENT is the partial key.

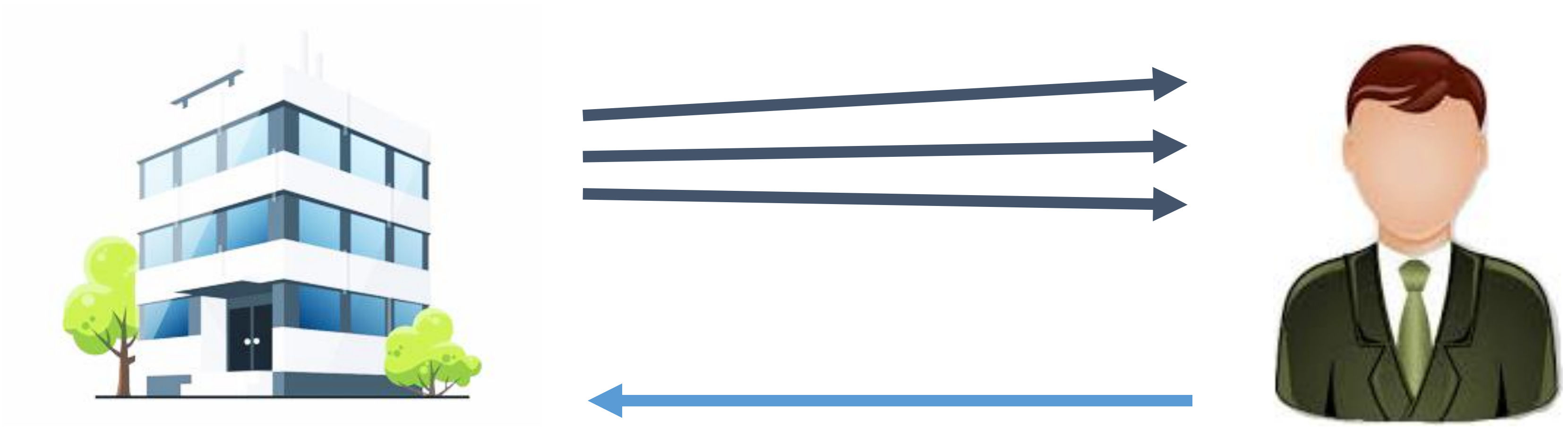


Relationships

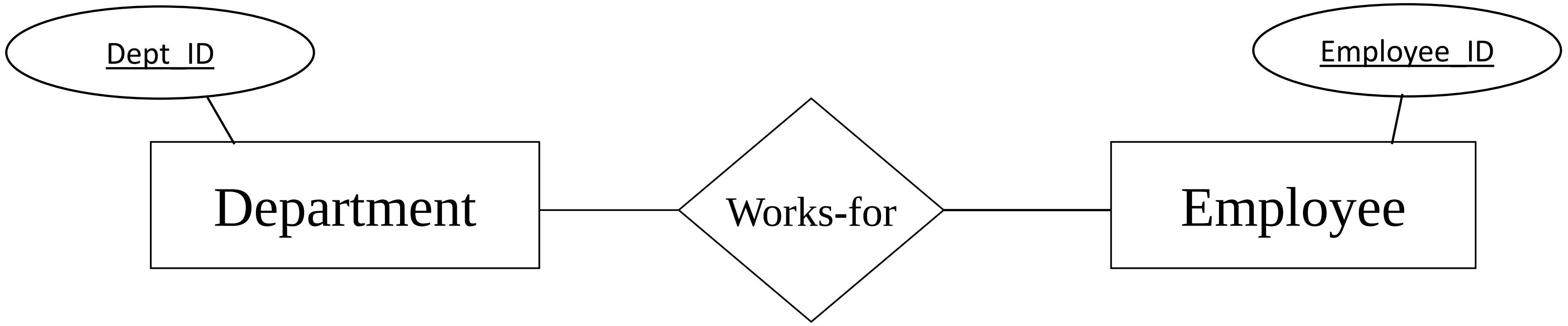
- Relationship exists, whenever an attribute of one entity type refers to another entity type.
- In the ER model these references should not be represented as attributes but as **relationships**.

Relationships

A **Department** has **Employees**



An **Employee** works for A **Department**



Degree of Relationship Type

The **degree** of a relationship type is defined as the Number of participating entity types.

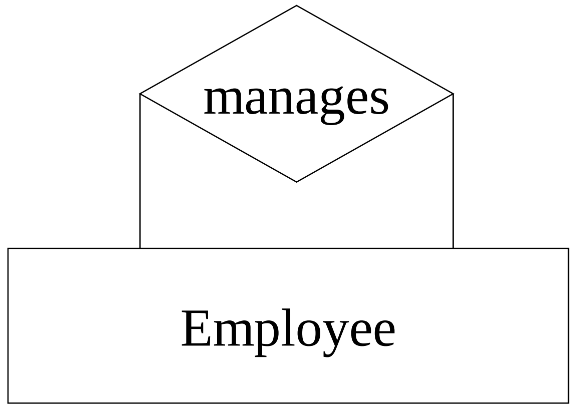
- Unary,
- Binary,
- Ternary,
- Quaternary,
- N-ary

Unary Relationship

- A relationship between the instances of a single entity type.
- Also known as **recursive relationships** or **self-referencing** relationships.

  e.g. Person is married to a Person

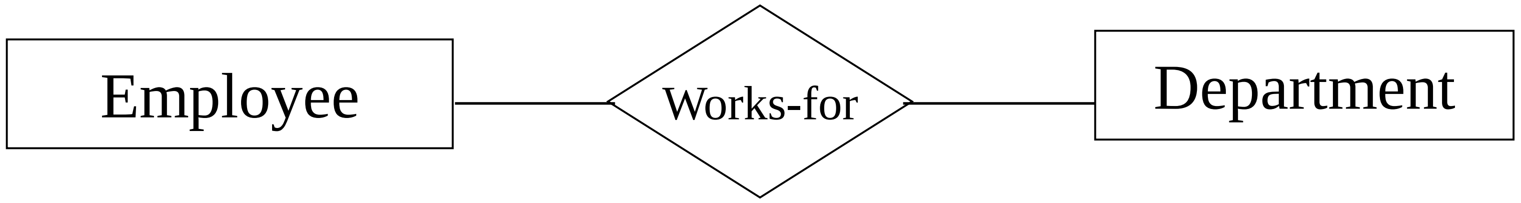
  Employee manages Employees



Binary Relationship

- A relationship between the instances of two entity types.

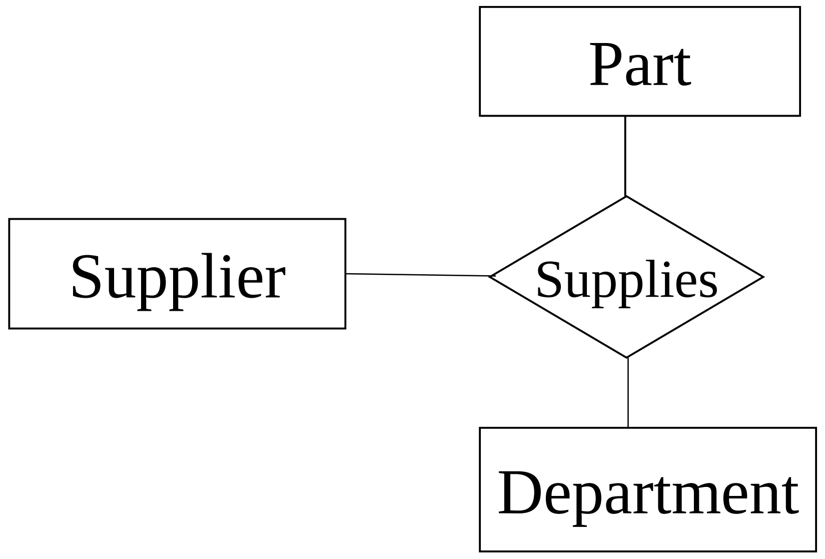
  e.g. An employee works for a department



Ternary Relationship

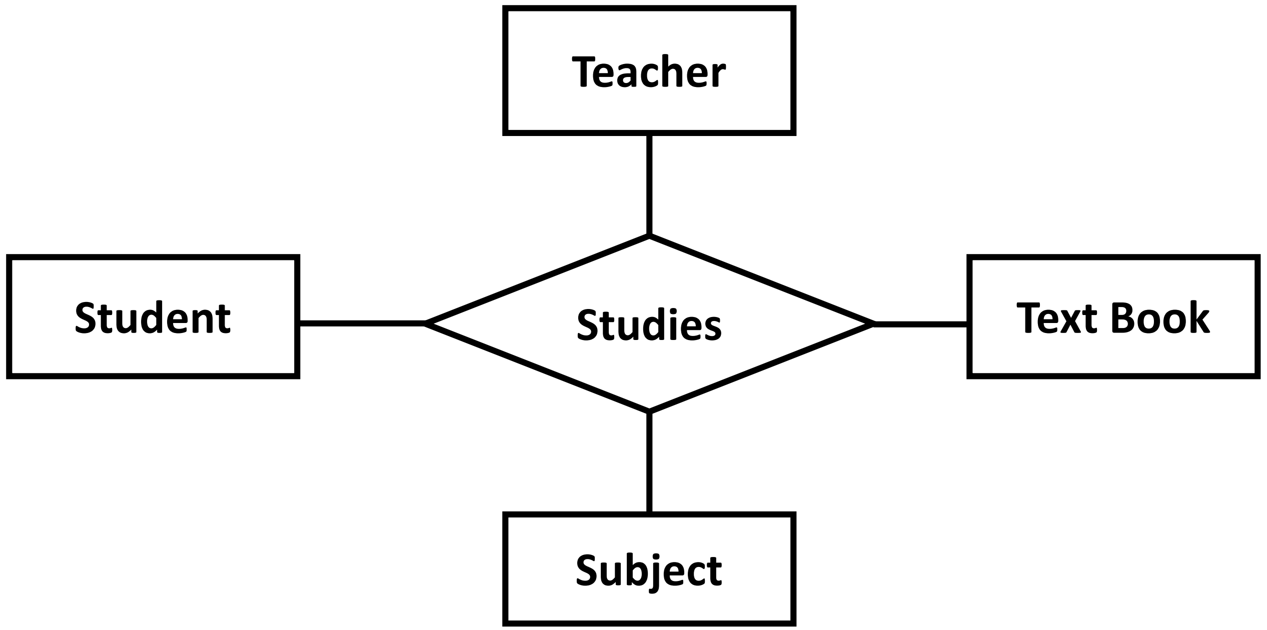
- A simultaneous relationship among the instances of three entity types.

  e.g. Supplier s supplies Part p to Department d



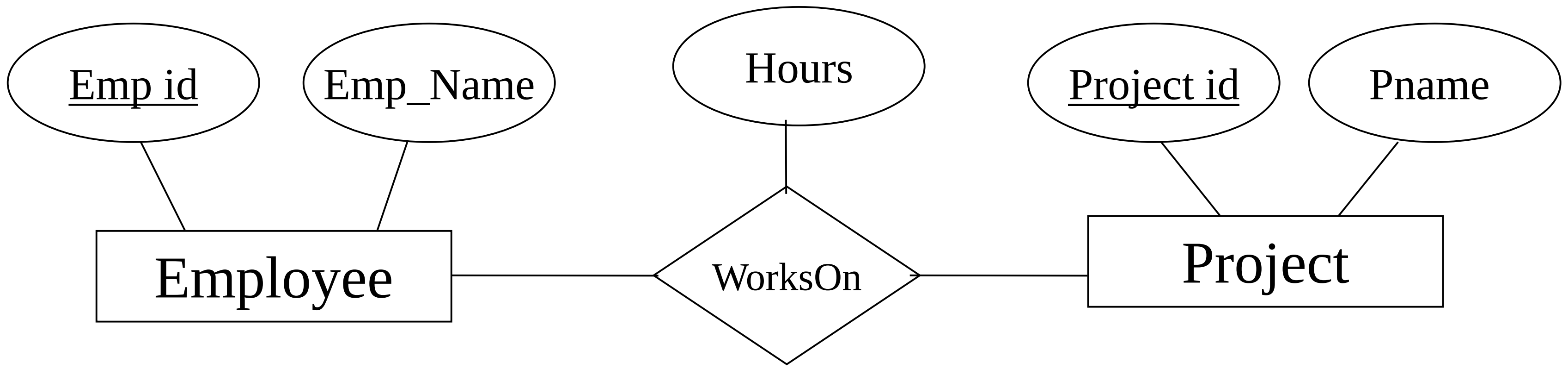
Quaternary Relationship

- A relationship between the instances of four entity types.



Relationship Attribute

- In some cases, a relationship type can have attributes.
- Usually in these cases the attribute does not belong to any of the participating entities.
- Due to that reason the attribute is added on the relationship.



Constraints on Relationship Type

- There are constraints that could be reflected in ER diagram.
- These constraints are known as **Structural Constraints**.

  For example, each employee should work for only one department.
- Types of structural constraints

  - Cardinality ratio

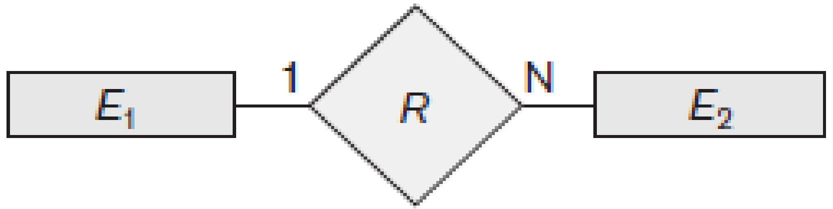
  - Participation

Cardinality Ratio

Specify the maximum number of instances of one entity that can (or must) be associated with each instance of another entity.

The possible cardinality ratios are

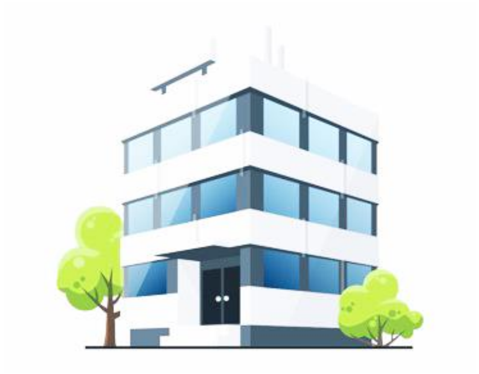
- 1:1
- 1:N
- M:N



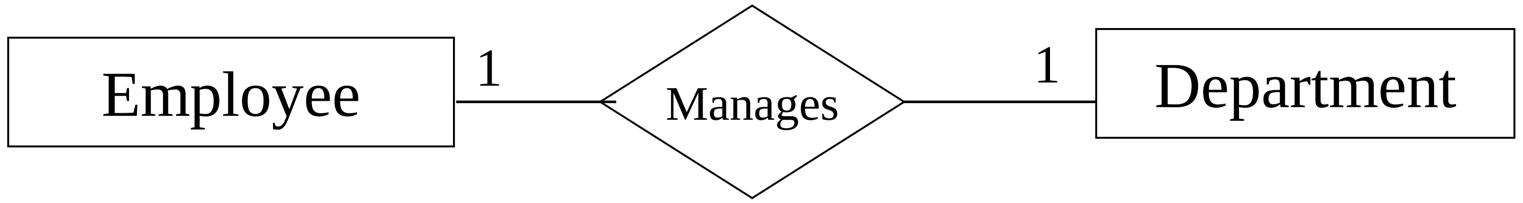
Cardinality Ratio 1: N for $E_1 : E_2$ in R

1:1

A **Department** has A Manager (**Employee**)

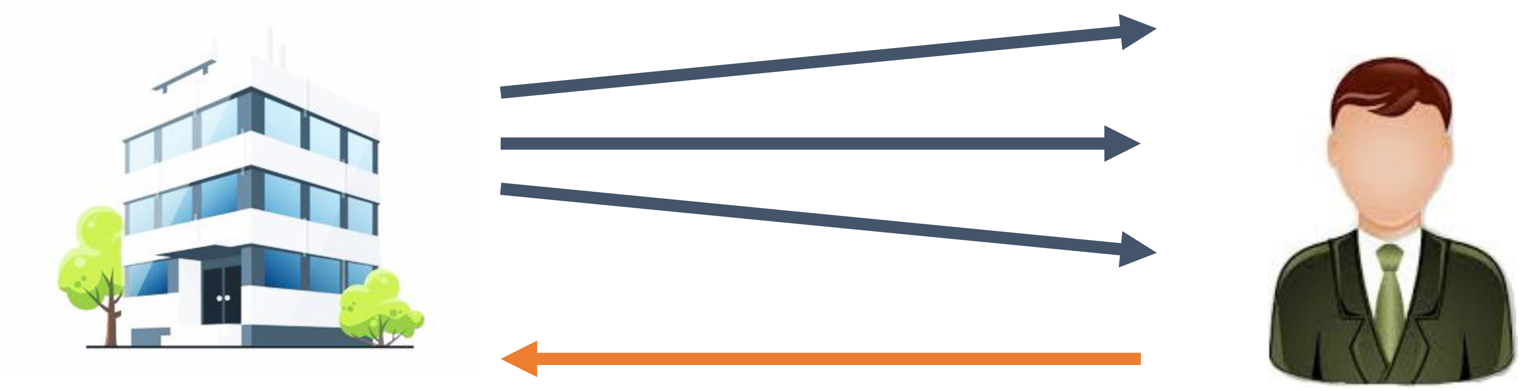


An **Employee** manages A **Department**

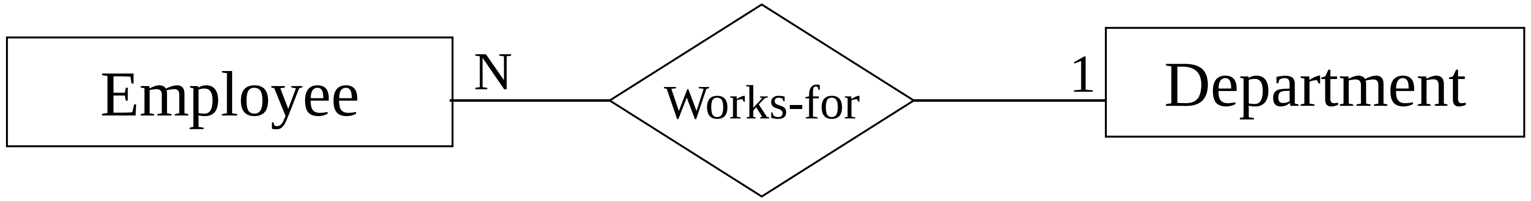


1:N

A **Department** has Many **Employees**

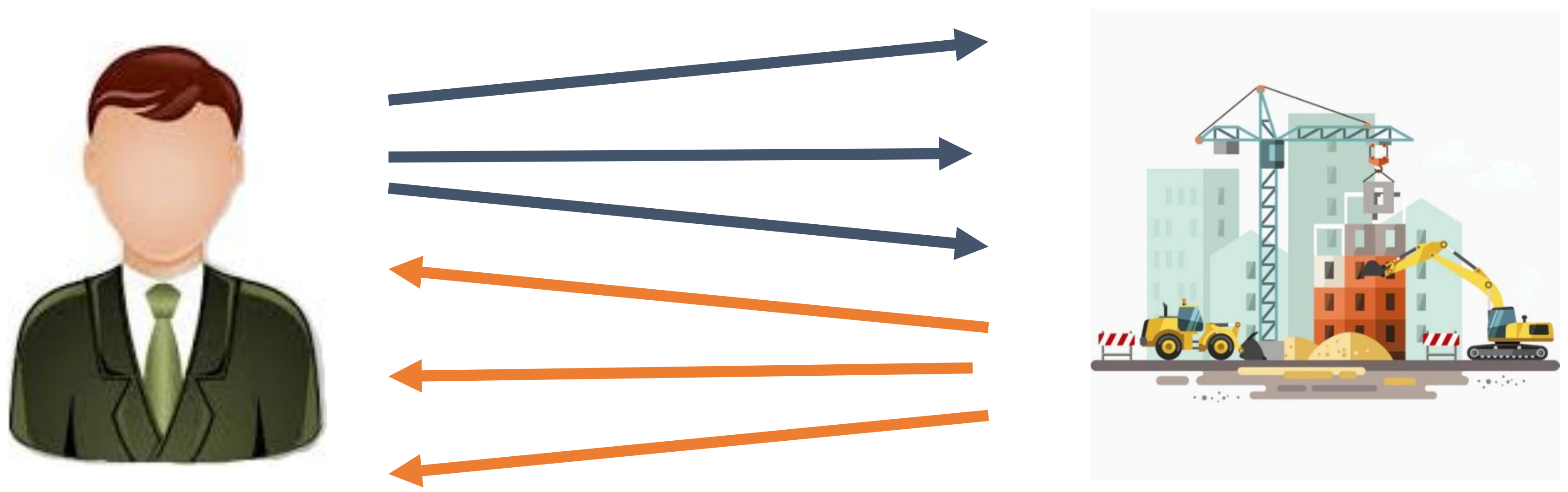


An **Employee** works for A **Department**

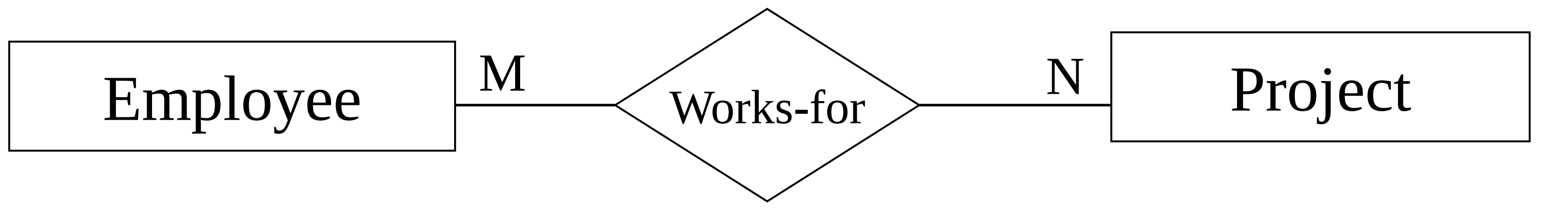


M:N

An **Employee** works on Many **Projects**

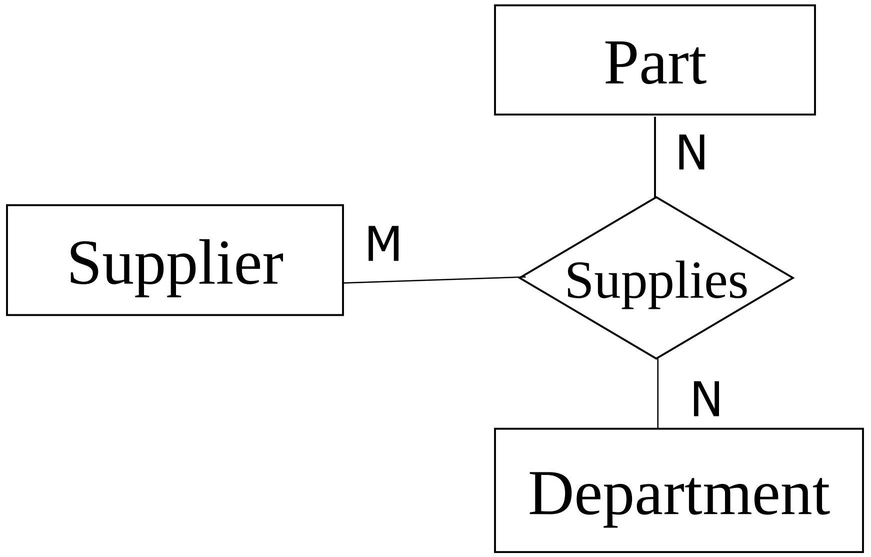


A **Project** has Many **Employees**



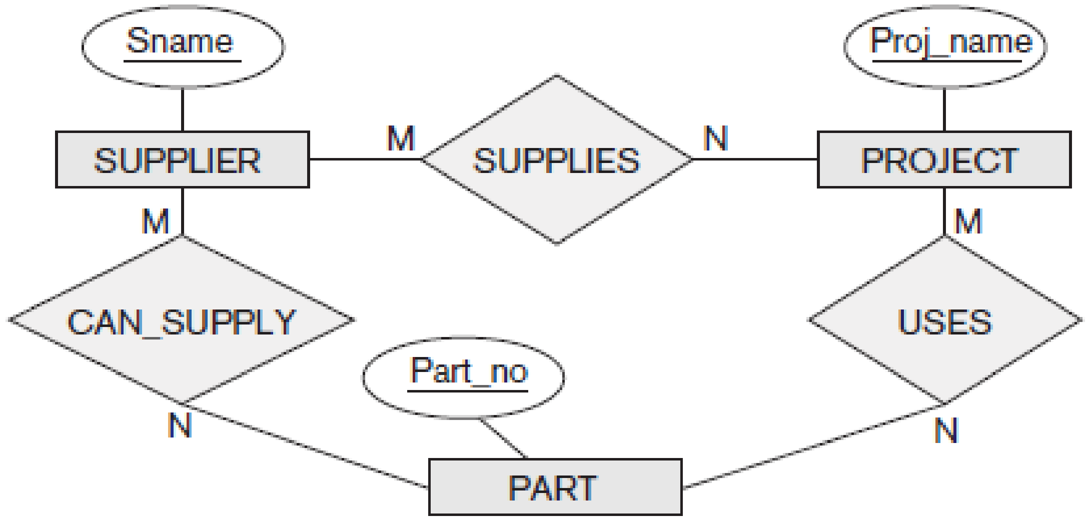
Ternary Relationship

- Ternary Cardinality
    - 1:1:1
    - 1:1:N
    - 1:M:N
    - M:N:N



Choosing between Binary and Ternary (or Higher-Degree) Relationships

- Ternary relationship type represents different information than do three binary relationship types.
- Consider the three binary relationship types *CAN_SUPPLY, USES*, and *SUPPLIES*.



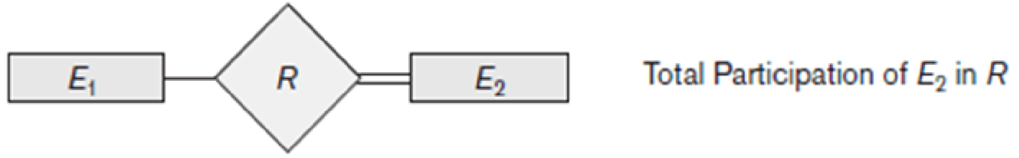
Choosing between Binary and Ternary (or Higher-Degree) Relationships

- CAN_SUPPLY - between SUPPLIER and PART, includes an instance (*s*, *p*) whenever supplier *s can supply* part *p* (to any project);
- USES - between PROJECT and PART, includes an instance (*j*, *p*) whenever project *j* uses part *p*; and
- SUPPLIES - between SUPPLIER and PROJECT, includes an instance (*s*, *j*) whenever *s* is a supplier for project *j*.

The existence of three relationship instances (*s*, *p*), (*j*, *p*), and (*s*, *j*) in *CAN_SUPPLY, USES*, and *SUPPLIES*, respectively, does not necessarily imply that an instance (*s*, *j*, *p*) exists in the ternary relationship SUPPLY, because the *meaning is different*.

Participation Constraints

This constraint specifies **the minimum number** of relationship instances that each entity can participate in.



There are two types

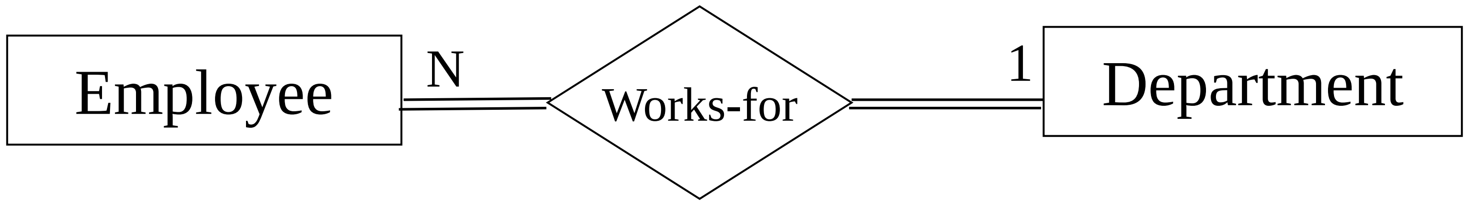
- **Total Participation** (existence dependency):

  e.g. every employee must work for a department.

  Every department must have employees working for it.

Therefore, participation of Employee in works-for relationship type is total.

Participation of department in works-for relationship type is total.



Participation Constraints

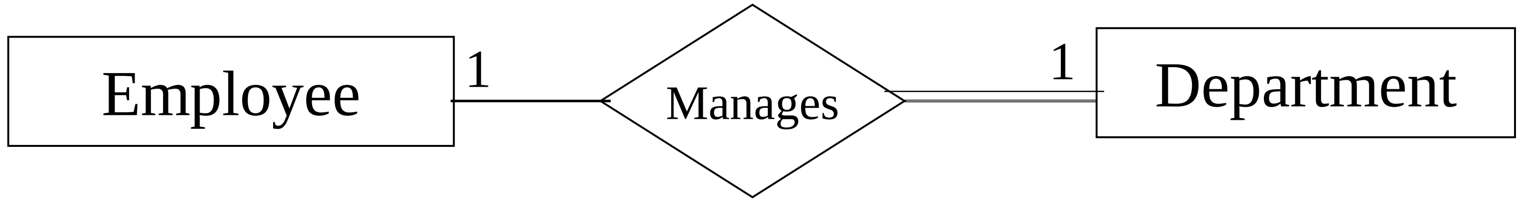
- **Partial Participation**

  - Should all employees manage departments?

    If the answer is yes : Total Participation

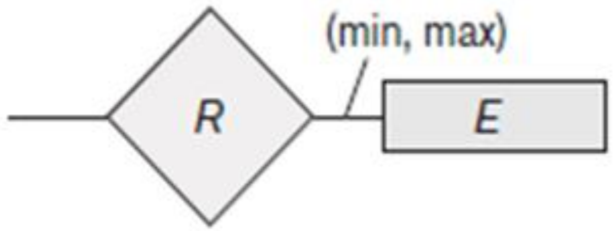
    If the answer is no : Partial Participation

  The participation of Employee in manages relationship type is partial.



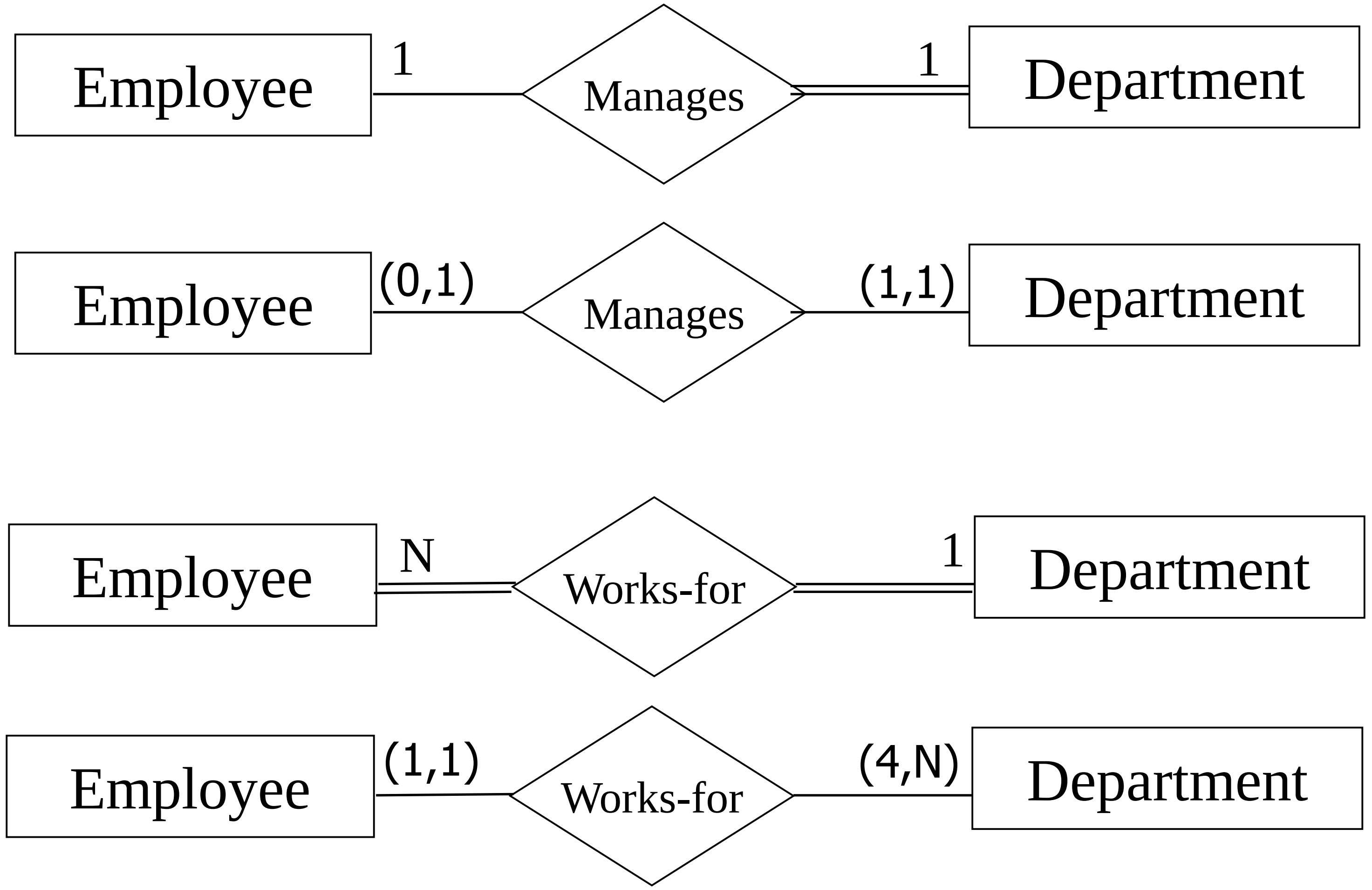
Constraints on Relationship Type - Notation

- It is possible to replace the cardinality ratio (1:1, 1:N, M:N) and single/double-line notation for participation constraints using an alternative notation. This notation involves associating a pair of integer numbers (min, max) with each participation of an entity type E in a relationship type R.
- The numbers mean that for each entity e in E, e must participate in at least min and at most max relationship instances in R at any point in time. In this method, min = 0 implies partial participation, whereas min > 0 implies total participation.

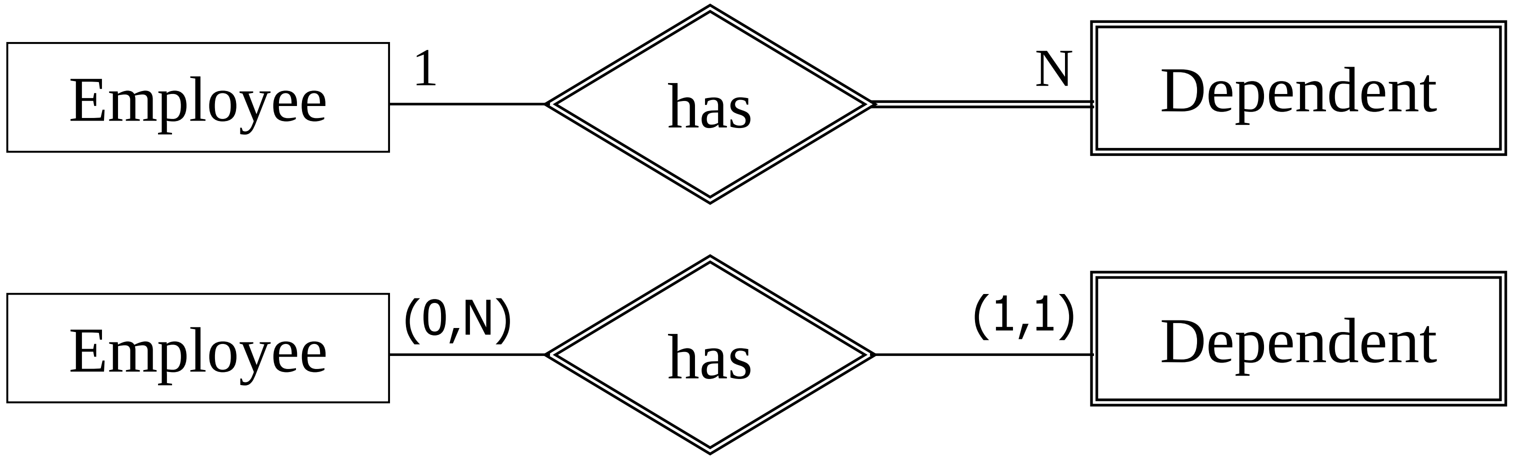


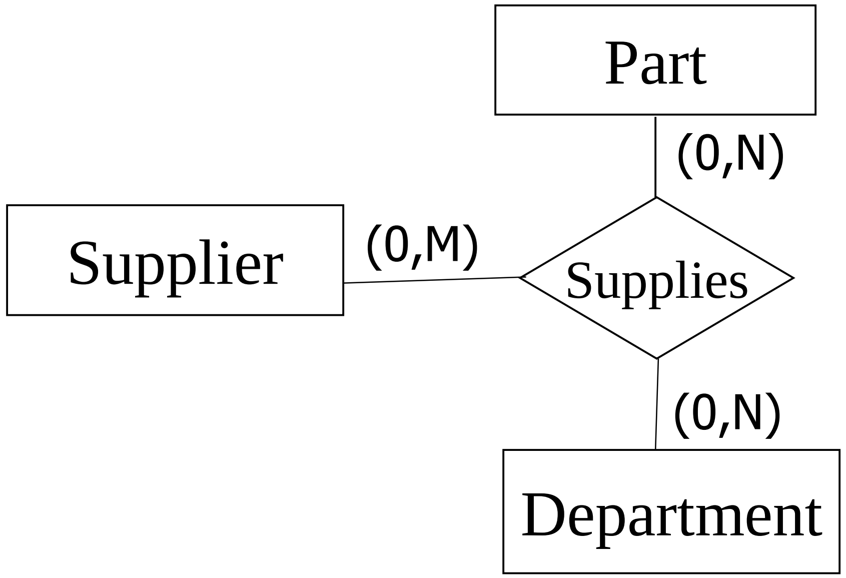
Structural Constraint (min, max) on Participation of *E* in *R*

Constraints on Relationship Type - Notation



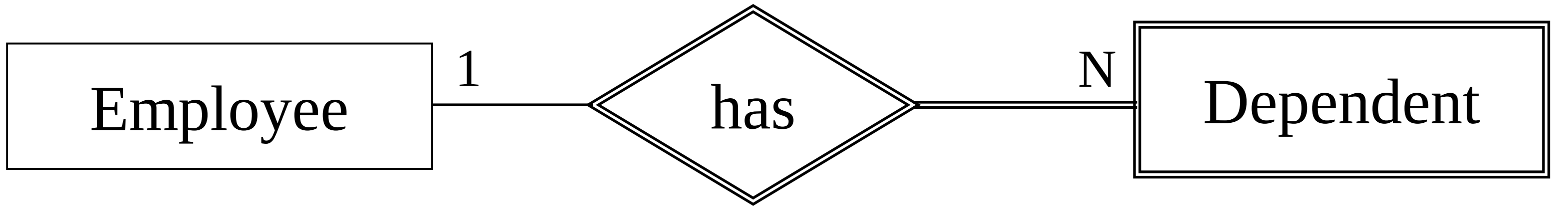
Constraints on Relationship Type - Notation



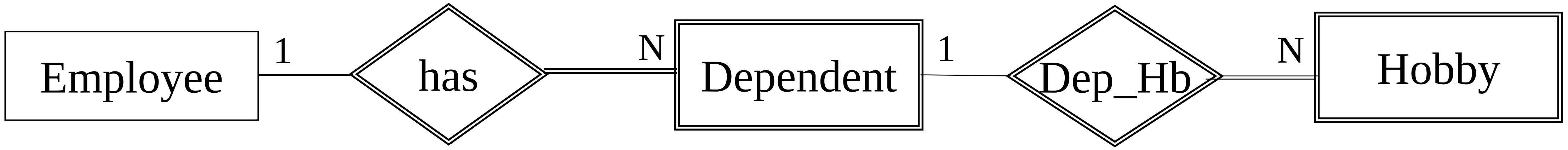


Relationships of Weak Entity Types

- The relationship type that relates a weak entity type to its owner the **identifying relationship** of the weak entity type.
- A weak entity type always has a *total participation constraint* (existence dependency) with respect to its identifying relationship because a weak entity cannot be identified without an owner entity.
- A relationship between a weak entity type and its owner is represented as given below.

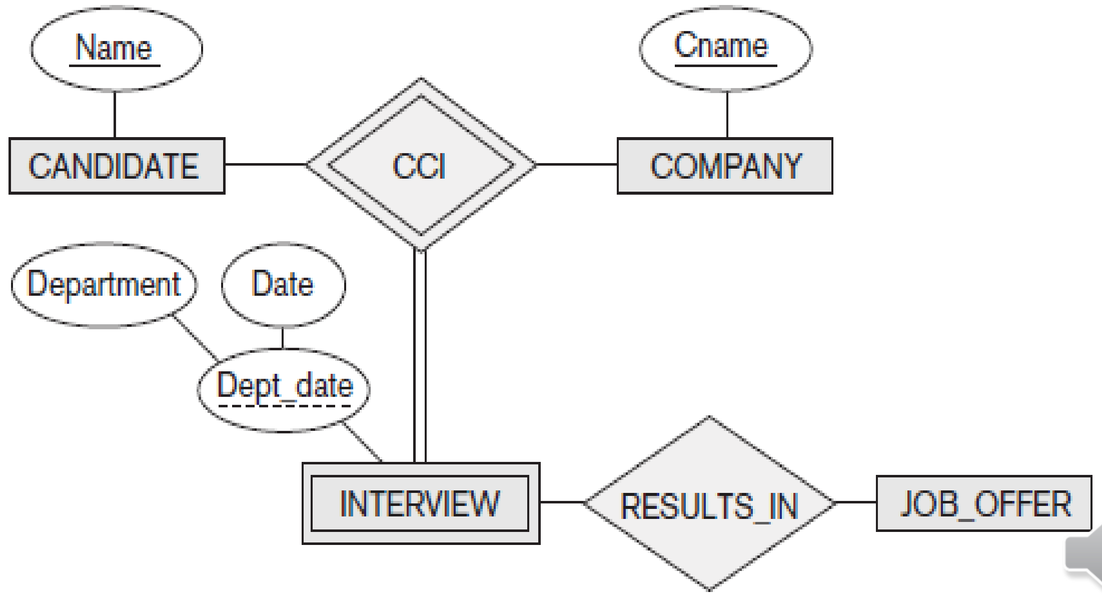


Relationships of Weak Entity Types



In general, any number of levels of weak entity types can be defined; an owner entity type may itself be a weak entity type.

In addition, a weak entity type may have more than one identifying entity type and an identifying relationship type of degree higher than two.

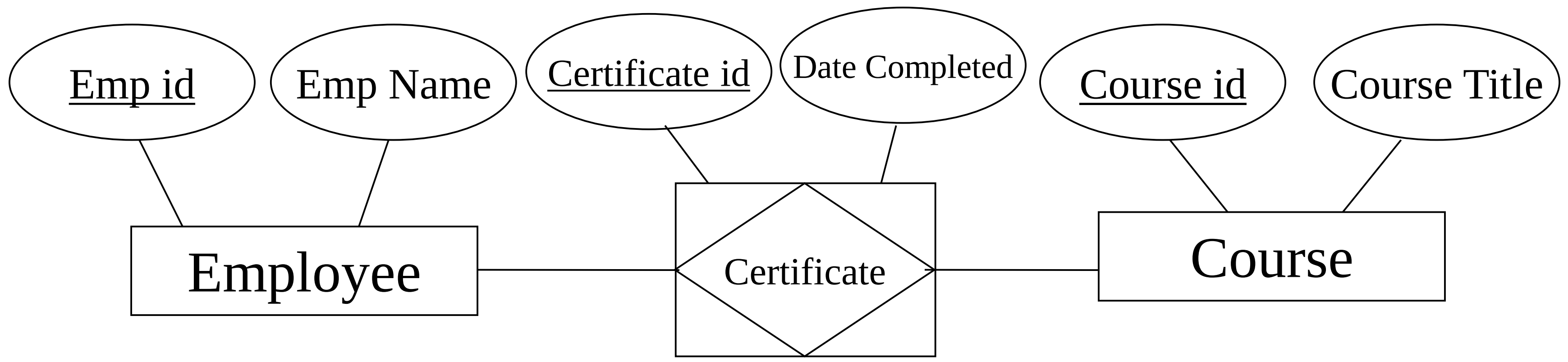


Associative Entities

- A **many**-to-**many** **relationship** can be modeled as an **associative entity**.
- The presence of one or more attributes on a relationship suggests that the relationship may be represented as an entity type.
- An associative entity is an entity type that associates the instances of one or more entity types and contains attributes that are relevant to the relationship between those entity instances.

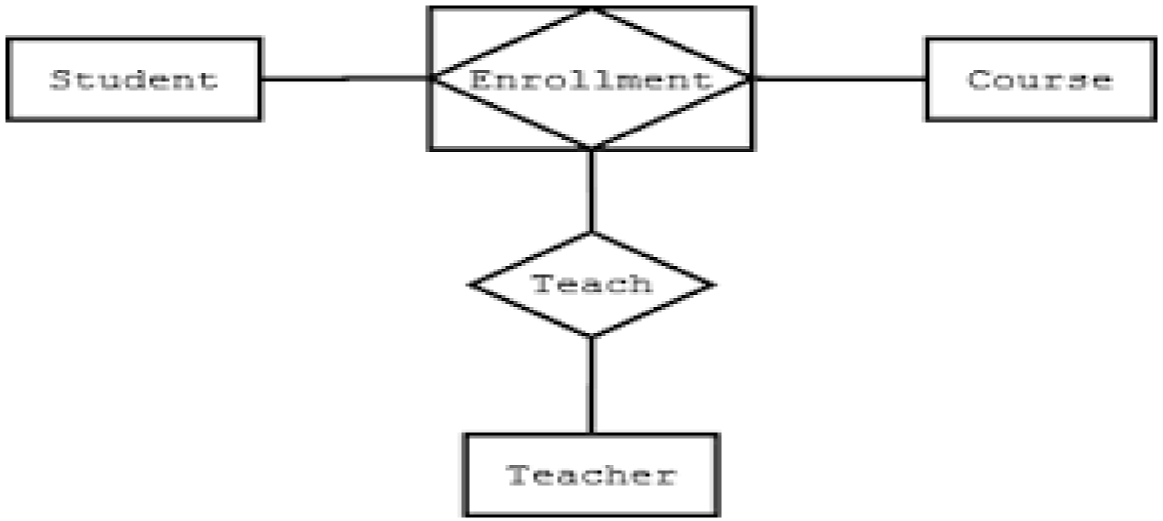
Associative Entities

- The resulting associative entity type has independent meaning to end-users, and can preferably be identified with a single-attribute identifier.
- The associative entity may have one or more attributes in addition to the identifier.
- E.g. Associative entity type CERTIFICATE is represented with the diamond relationship symbol enclosed within the entity rectangle.



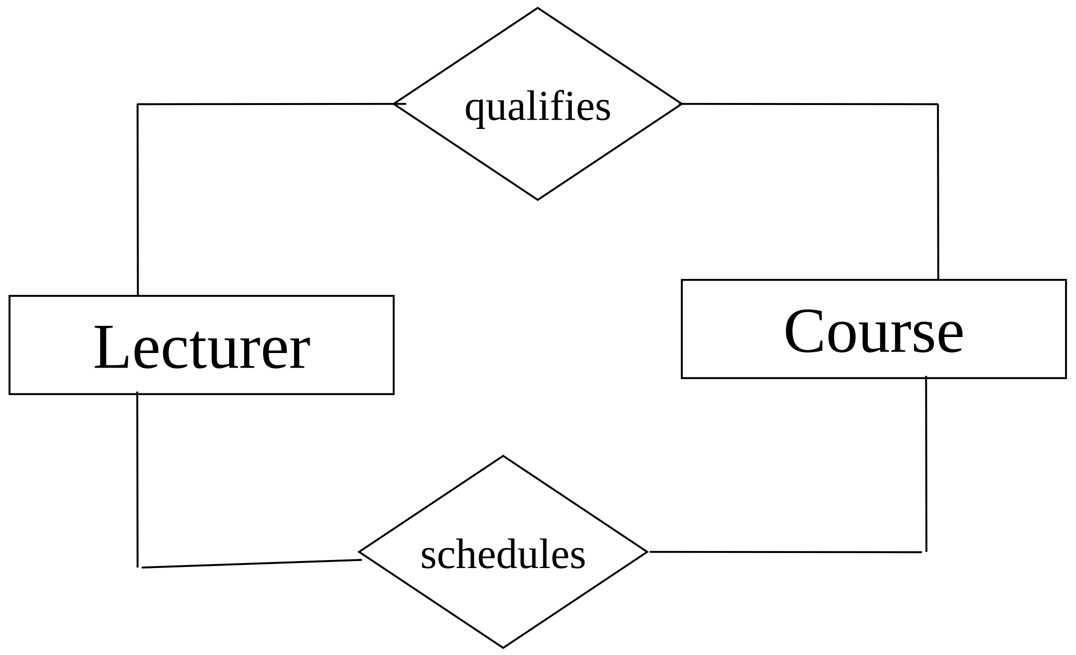
Associative Entities

The associative entity can participate in one or more relationships independent of the entities related in the associated relationship.

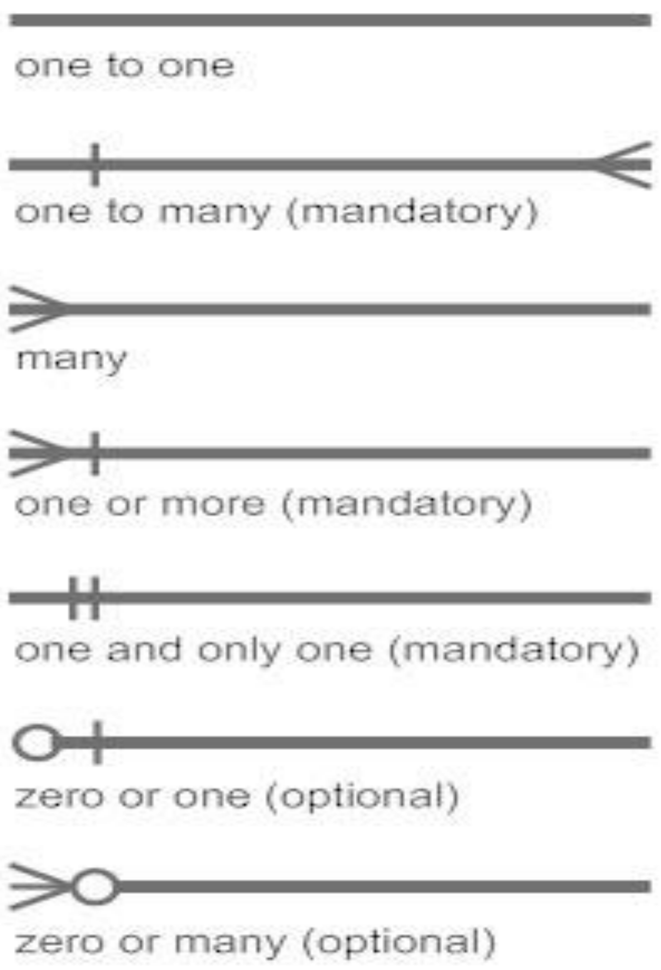


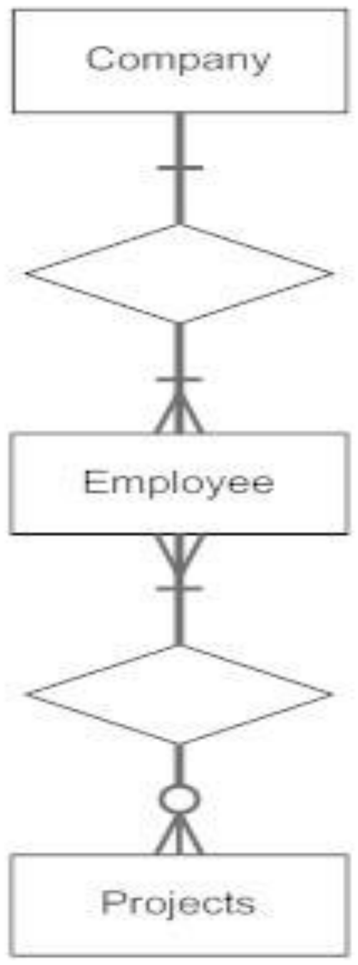
Multiple Relationships

In some situations an organisation may want to model more than one relationship between the same entity types.

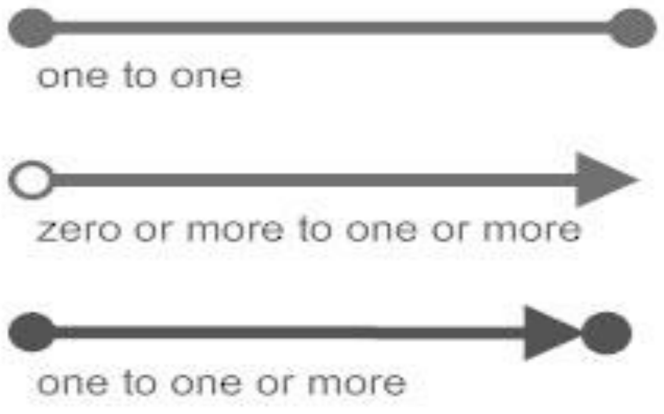


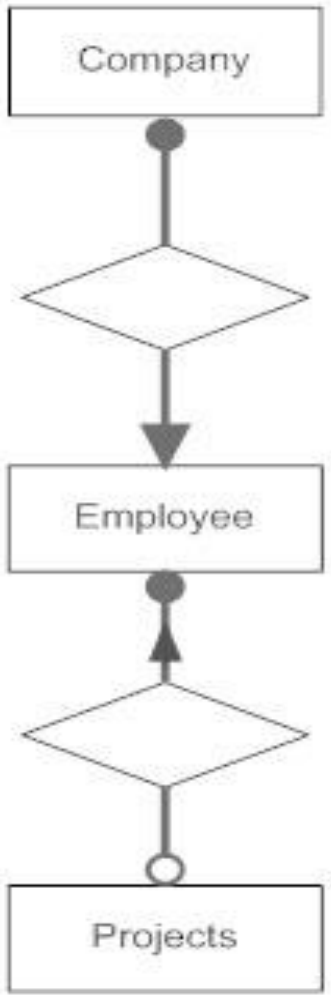
Information Engineering Style





Bachman Style





Martin Style

1 - one, and only one (mandatory)

* - many (zero or more - optional)

1...* - one or more (mandatory)

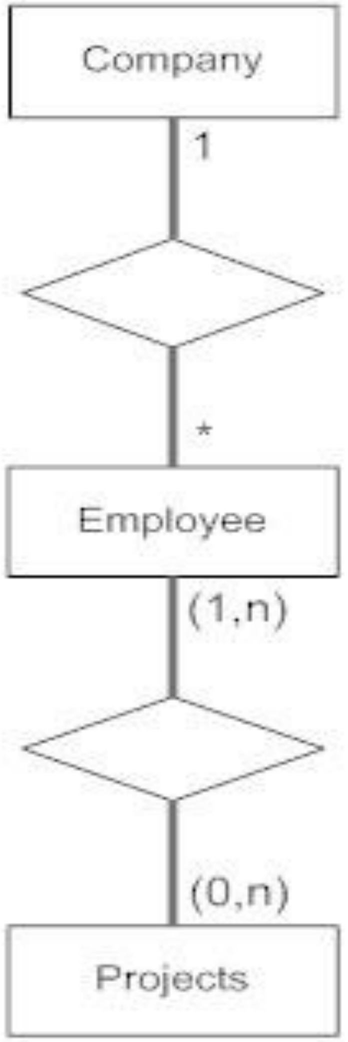
0...1 - zero or one (optional)

(0,1) - zero or one (optional)

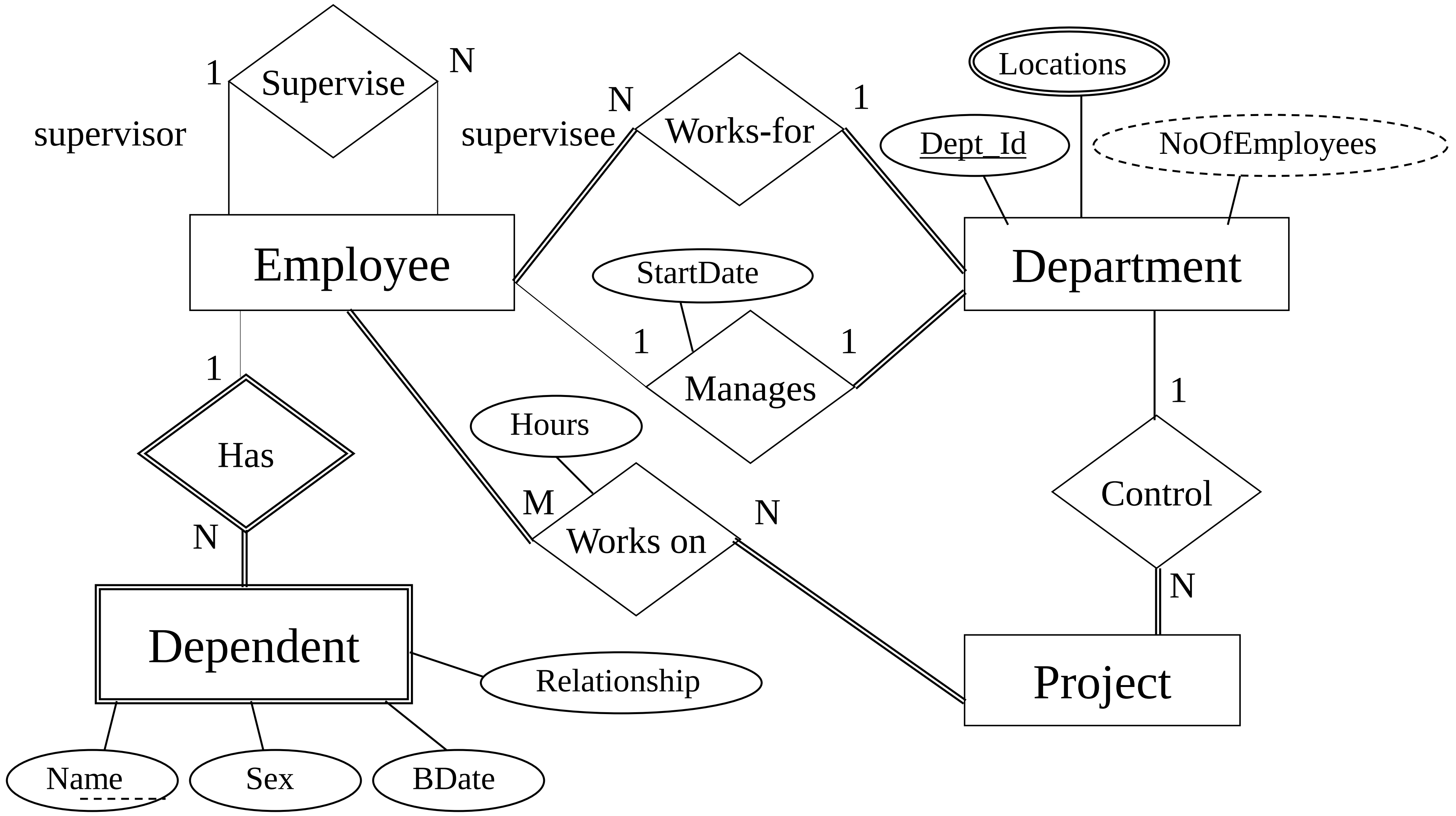
(1,n) - one or more (mandatory)

(0,n) - zero or more (optional)

(1,1) - one and only one (mandatory)



ER Diagram - COMPANY


Supervise
1
N
supervisor
supervisee
Works-for
N
1
Locations
Dept_Id
NoOfEmployees
Employee
Department
StartDate
1
1
Manages
1
Has
N
Hours
M
Works on
N
Control
1
N
Dependent
Relationship
Project
Name
Sex
BDate