Entity Relationship Diagrams

Tanzim Hossain
Lecturer
Department of Software Engineering
Daffodil International University

Basic Symbols

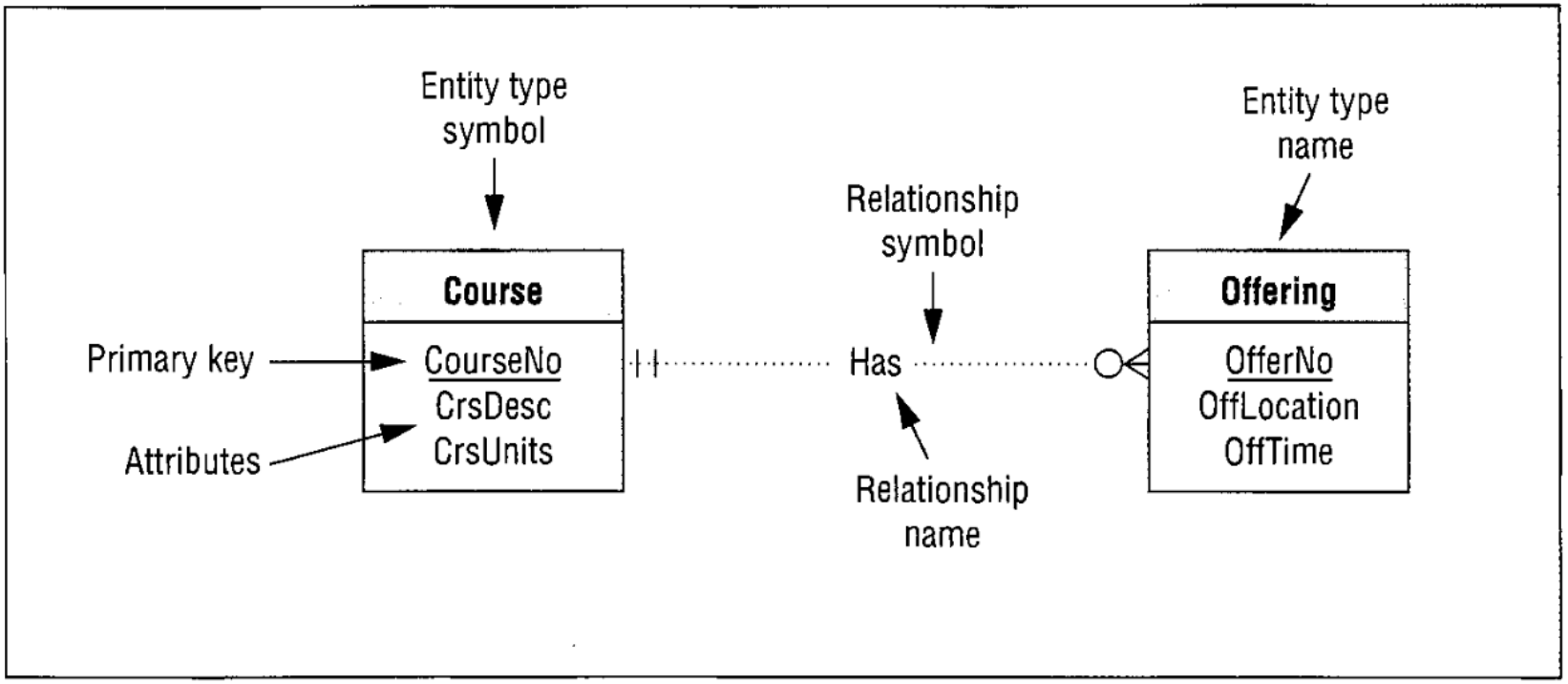
Entity Type & Attributes

- **Entity type:** a collection of entities (persons, places, events, or things) of interest represented by a rectangle in an entity relationship diagram.
- **Attribute:** A property of an entity type or relationship. Each attribute has a data type that defines the kind of values and permissible operations on the attribute.

Relationship

- **Relationship:** A named association among entity types.
- A relationship represents a two-way or bidirectional association among entities.
- Most relationships involve two distinct entity types.

Basic Symbols



Cardinality

Cardinality & Existence Dependency

- A constraint on the number of entities that participate in a relationship.
- In an ERD, the minimum and maximum cardinalities are specified for both directions of a relationship.
- **Existence dependency:** An entity that cannot exist unless another related entity exists.
- A mandatory relationship creates an existence dependency.

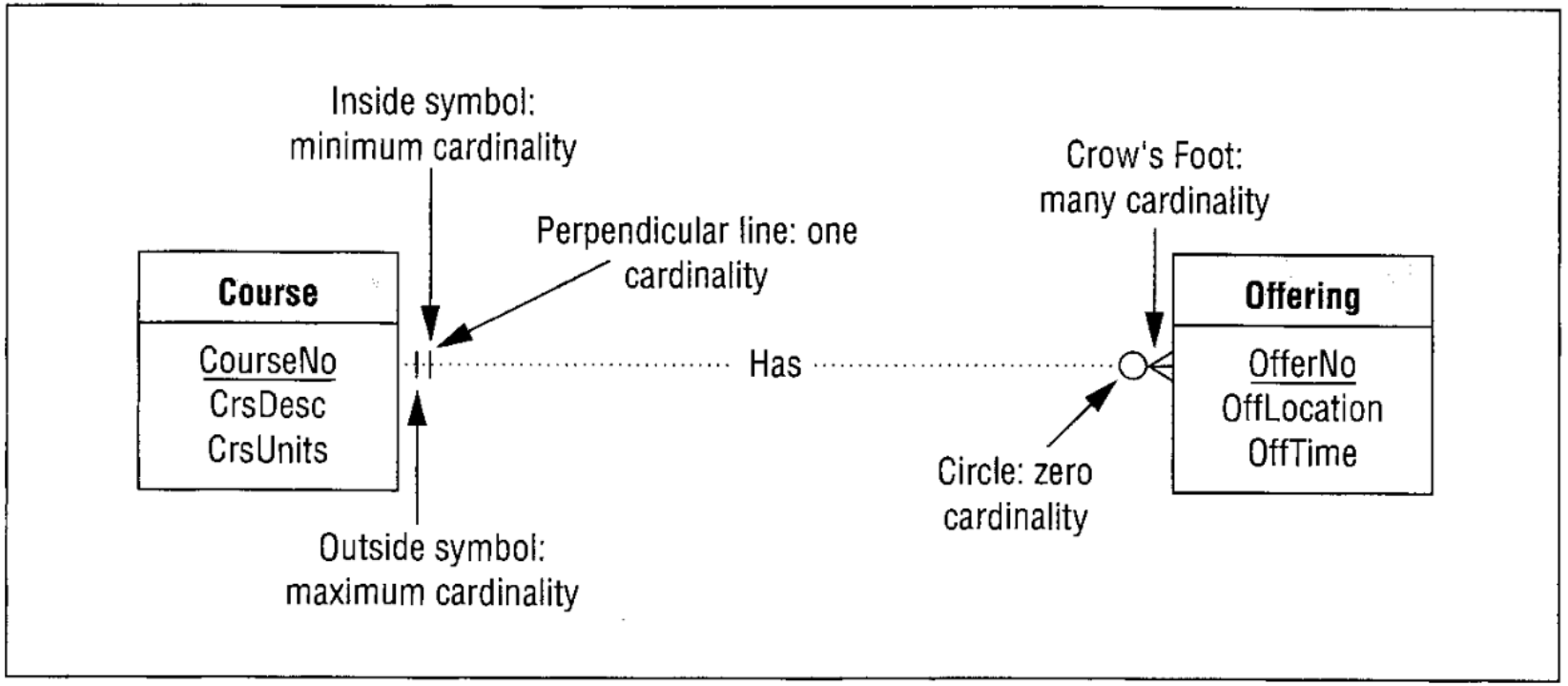
Crow's Foot Representation

- The Crow's Foot notation uses three symbols to represent cardinalities.
- The Crow's Foot symbol (two angled lines and one straight line) denotes many (zero or more) related entities. The Crow's Foot symbol near the Offering entity type means that a course can be related to many offerings.
- The circle means a cardinality of zero, while a line perpendicular to the relationship line means a cardinality of one.

Crow's Foot Representation

- To depict minimum and maximum cardinalities, the cardinality symbols are placed adjacent to each entity type in a relationship.
- The minimum cardinality symbol appears toward the relationship name while the maximum cardinality symbol appears toward the entity type.

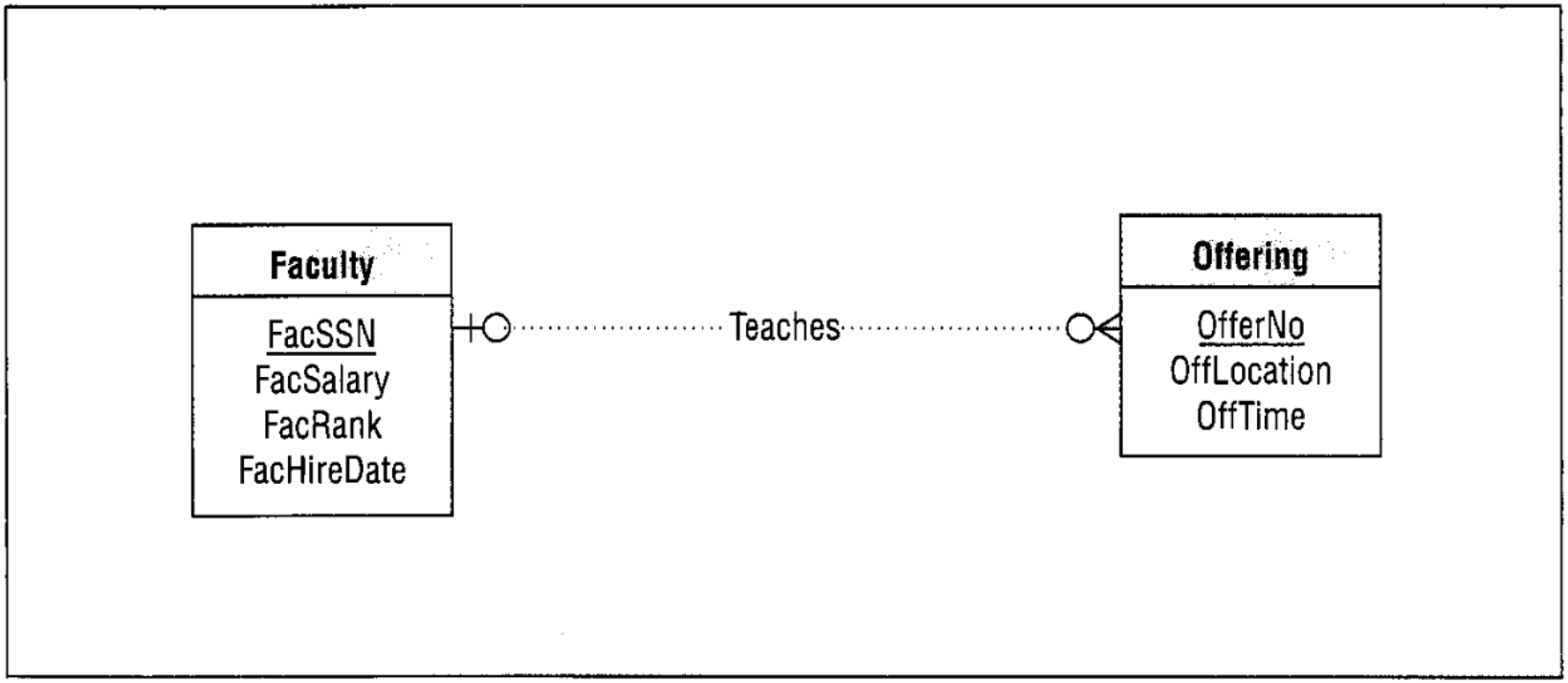
ERD With Crow's Foot Cardinality



Crow's Foot Representation

- A course is related to a minimum of zero offerings (circle in the inside position) and a maximum of many offerings (Crow's Foot in the outside position).
- Similarly, an offering is related to exactly one (one and only one) course as shown by the single vertical lines in both inside and outside positions.

Optional Relationship



A minimum cardinality of zero indicates an optional relationship.

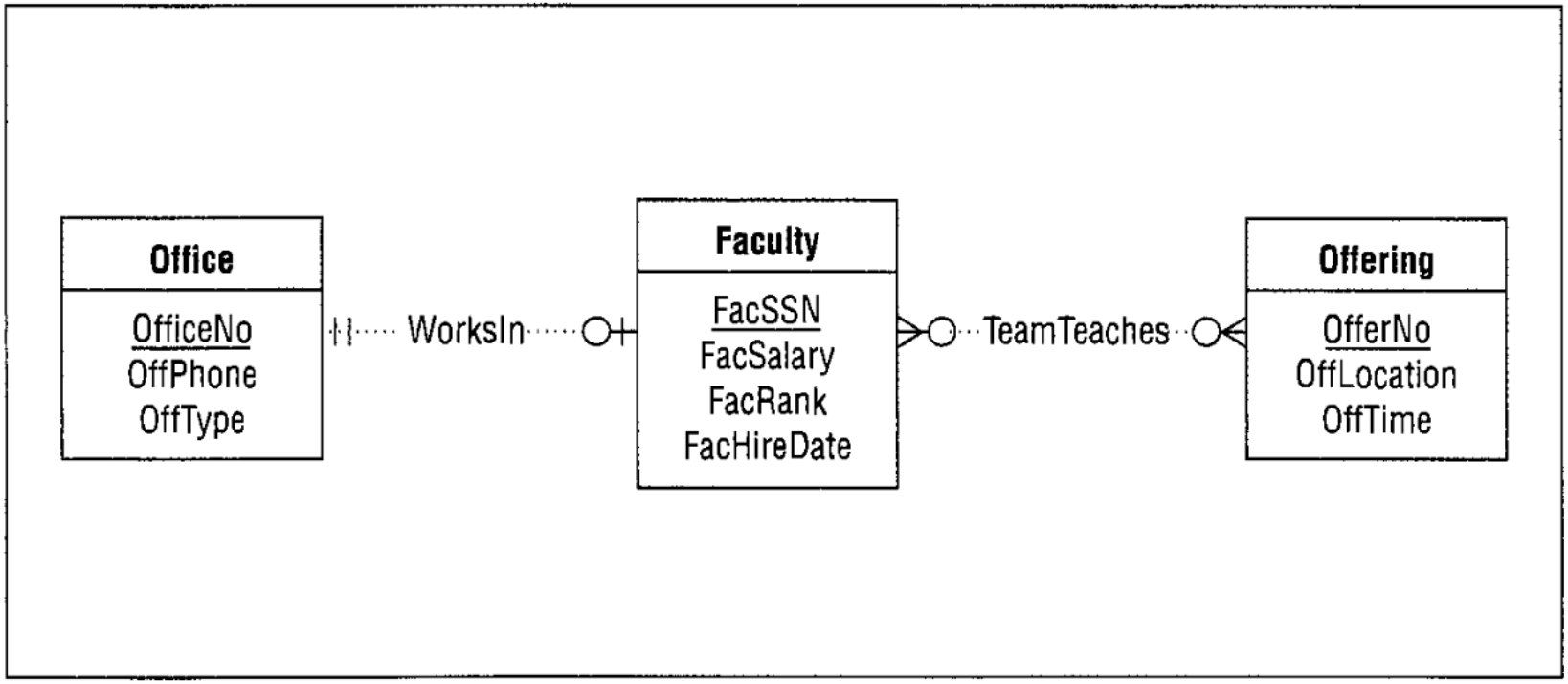
Functional Relationship

- A maximum cardinality of one means the relationship is **single-valued** or **functional**.
- For example, the *Has* and *Teaches* relationships are functional for Offering because an Offering entity can be related to a maximum of one Course and one Faculty entity.
- The word function comes from mathematics where a function gives one value.

1-M Relationship

- A relationship that has a maximum cardinality of one in one direction and more than one (many) in the other direction is called a 1-M (read one-to-many) relationship.
- Both the Has and Teaches relationships are 1-M.

M-N & 1-1 Relationship



M-N & 1-1 Relationship

- A relationship that has a maximum cardinality of more than one in both directions is known as an M-N (many-to-many) relationship.
- The *TeamTeaches* relationship allows multiple professors to jointly teach the same offering.
- Less common are 1-1 relationships in which the maximum cardinality equals one in both directions.
- For example, the WorksIn relationship in allows a faculty to be assigned to one office and an office to be occupied by at most one faculty.

Summary of Cardinality Classification

Classification	Cardinality Restrictions
Mandatory	Minimum cardinality ≥ 1
Optional	Minimum cardinality $= 0$
Functional or single-valued	Maximum cardinality $= 1$
1-M	Maximum cardinality $= 1$ in one direction and maximum cardinality > 1 in the other direction
M-N	Maximum cardinality is > 1 in both directions
1-1	Maximum cardinality $= 1$ in both directions

Understanding Relationships

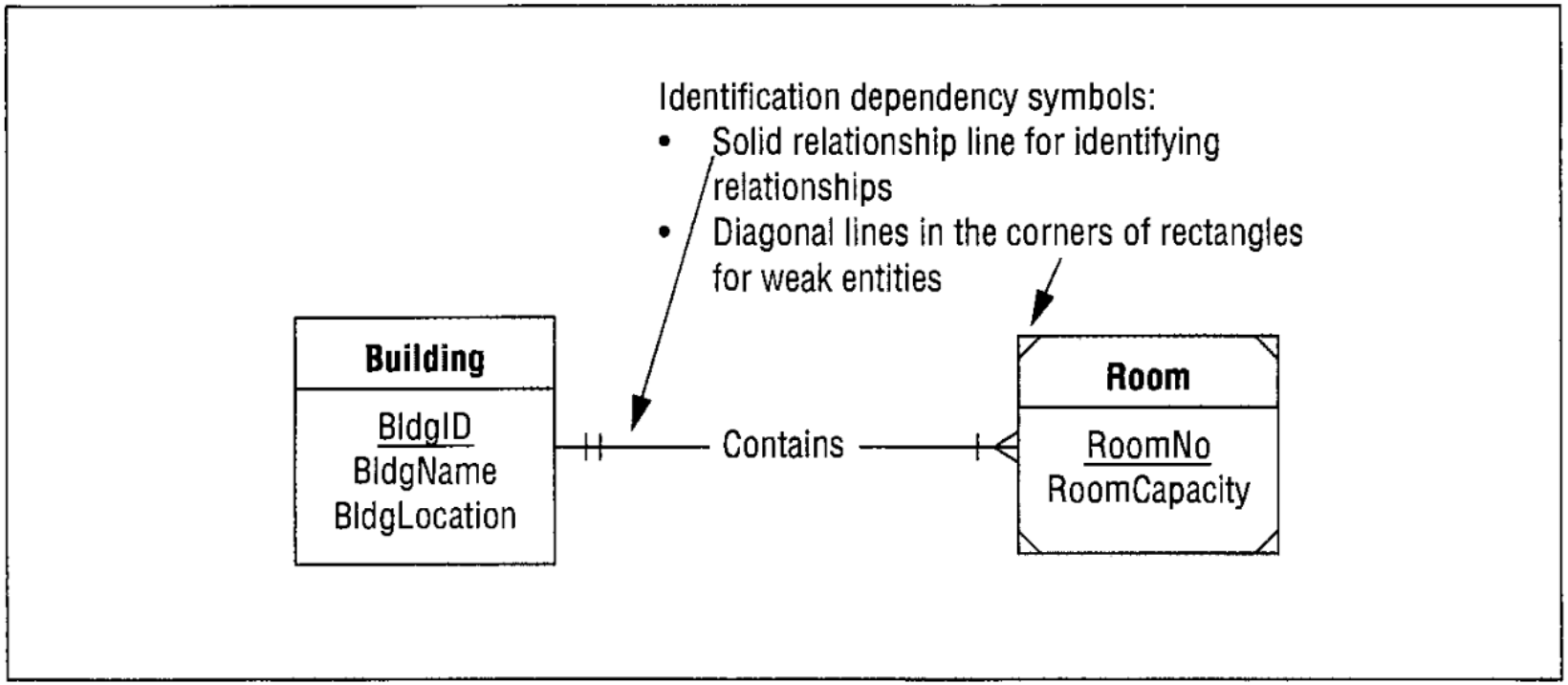
Identification Dependency

- In an ERD, some entity types may not have their own primary key.
- Entity types without their own primary key must borrow part (or all) of their primary key from other entity types.
- Entity types that borrow part or their entire primary key are known as **weak entities**.
- The relationship(s) that provides components of the primary key is known as an **identifying relationship**.

Identification Dependency

- Identification dependency occurs because some entities are closely associated with other entities.
- For example, a room does not have a separate identity from its building be cause a room is physically contained in a building.
- Thus, the primary key of Room is a combination of BldgID and RoomNo.

Identification Dependency



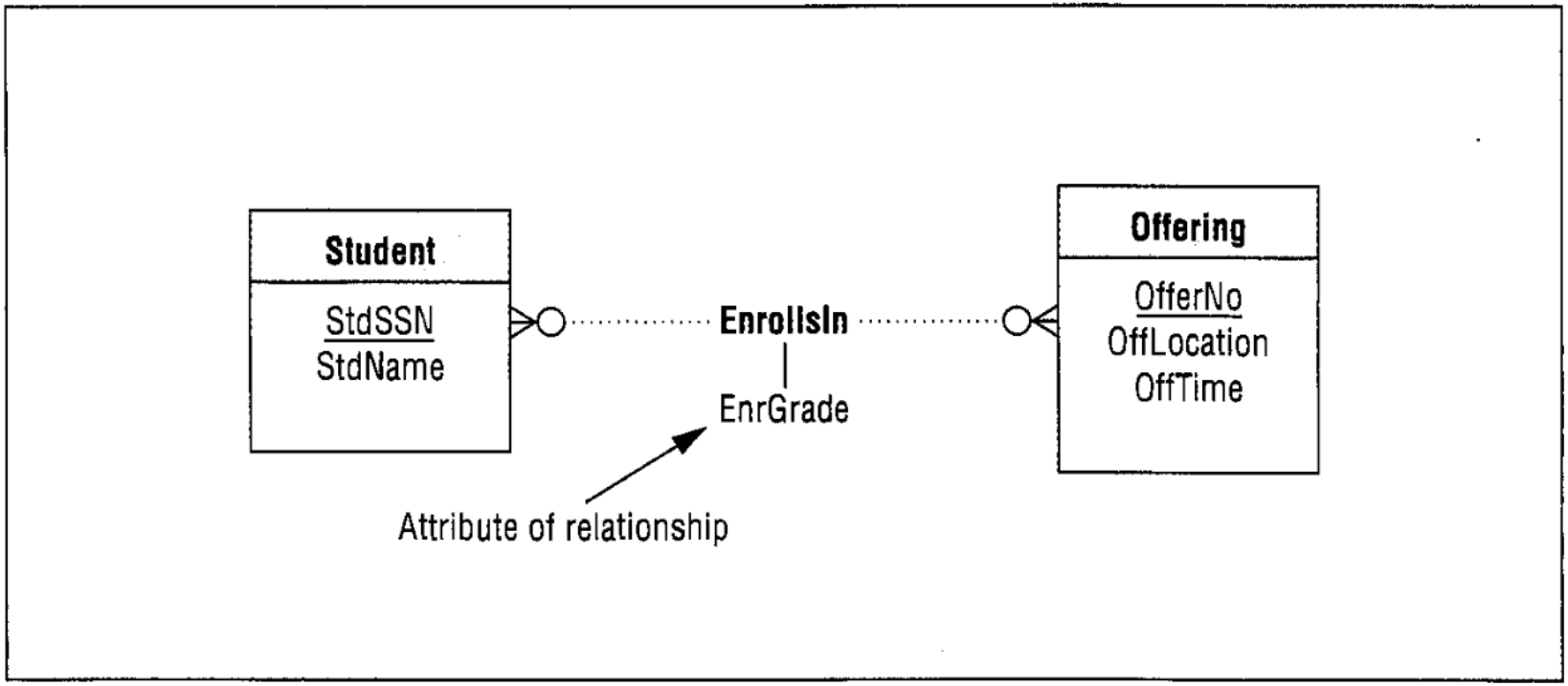
Identification Dependency

- Identification dependency is a specialized kind of existence dependency.
- Recall that an existent-dependent entity type has a mandatory relationship (minimum cardinality of one).
- Weak entities are existent dependent on the identifying relationships.
- Because of the existence dependency and the primary key borrowing, the minimum and maximum cardinalities of a weak entity are always 1.

M-N Relationship with Attribute

- Relationships can have attributes.
- This situation typically occurs with M - N relationships.
- In an M - N relationship, attributes are associated with the combination of entity types, not just one of the entity types.
- If an attribute is associated with only one entity type, then it should be part of that entity type, not the relationship.

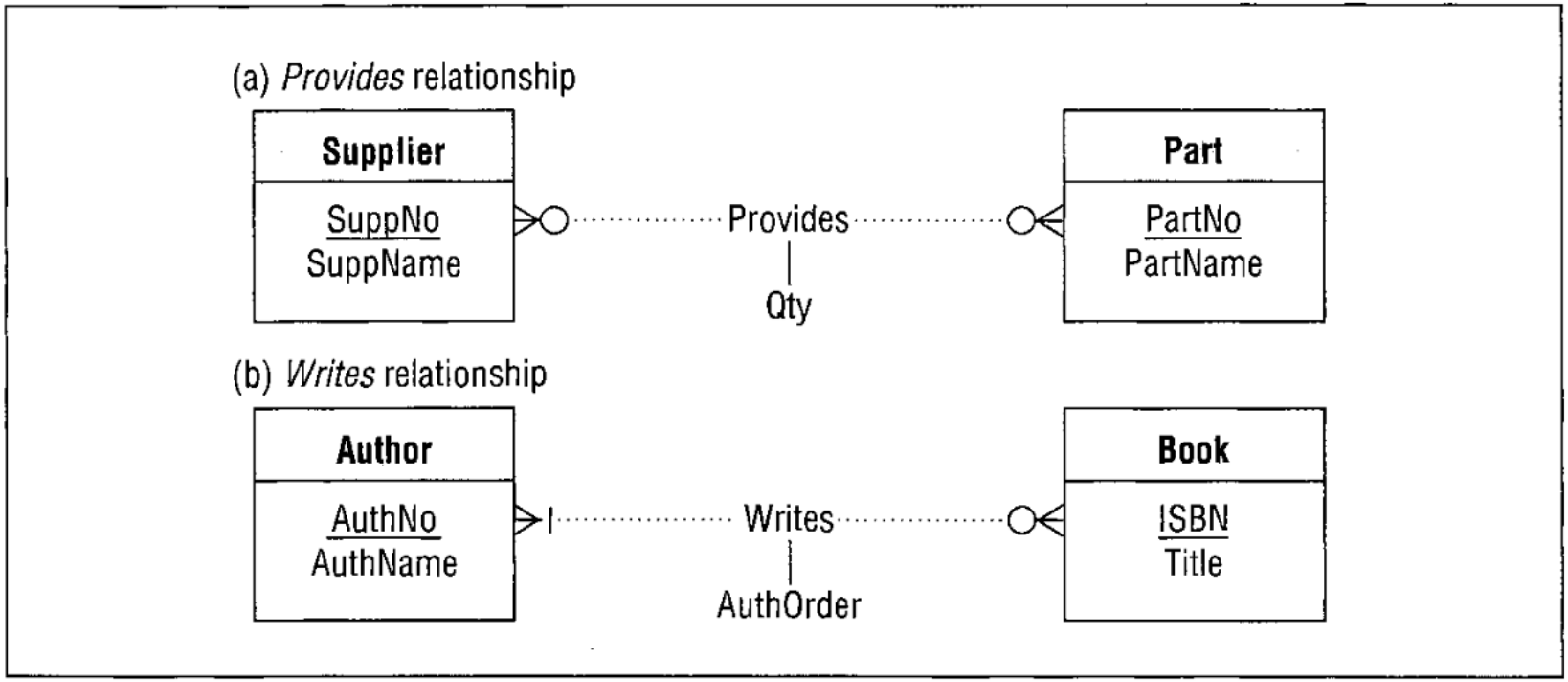
M-N Relationship with Attribute



M-N Relationship with Attribute

- Rhe attribute *EnrGrade* is associated with the combination of a student and offering, not either one alone.
- For example, the *EnrollsIn* relationship records the fact that the student with Social Security number 123-77-9993 has a grade o f 3.5 in the offering with offer number 1256.

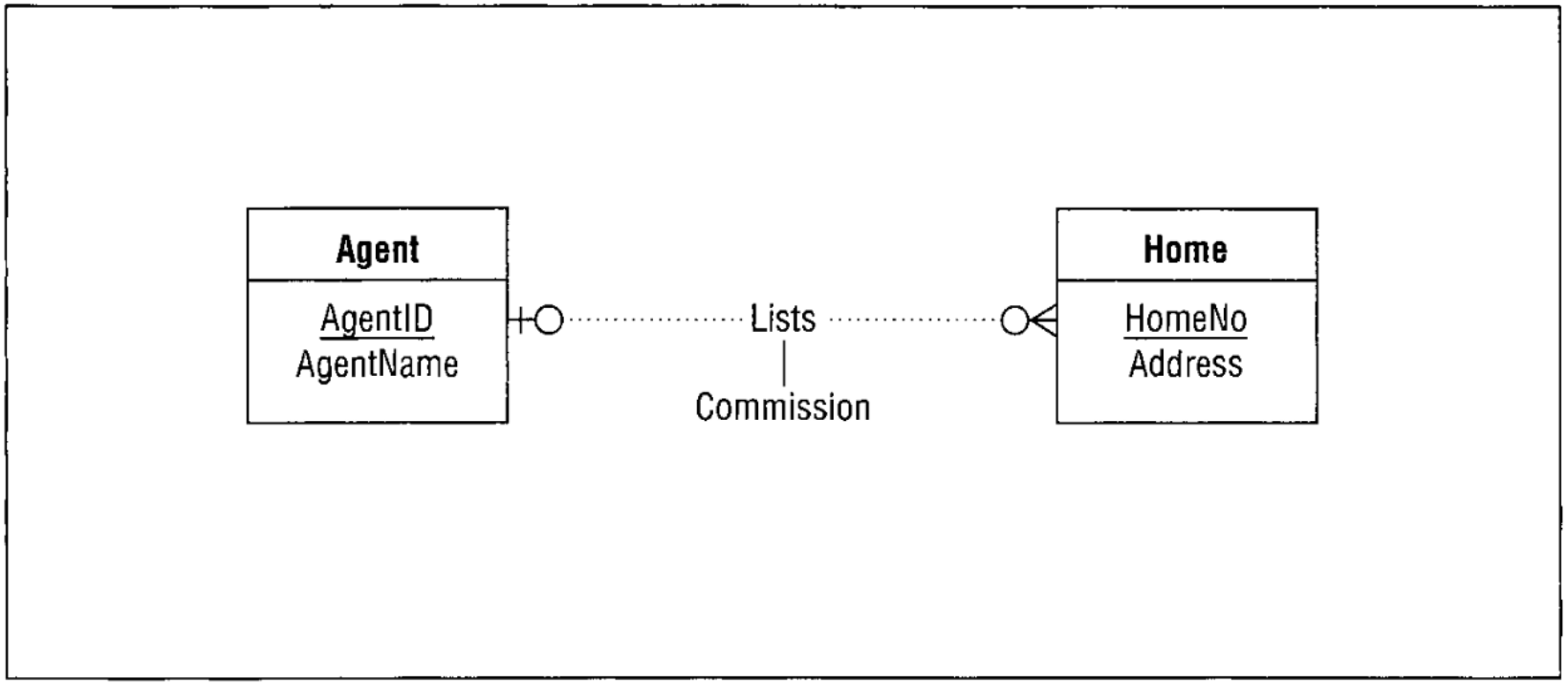
M-N Relationship with Attribute



M-N Relationship with Attribute

- The attribute *Qty* represents the quantity of a part supplied by a given supplier.
- The attribute *AuthOrder* represents the order in which the author's name appears in the title of a book.
- To reduce clutter on a large diagram, relationship attributes may not be shown.

1-M Relationship with Attribute



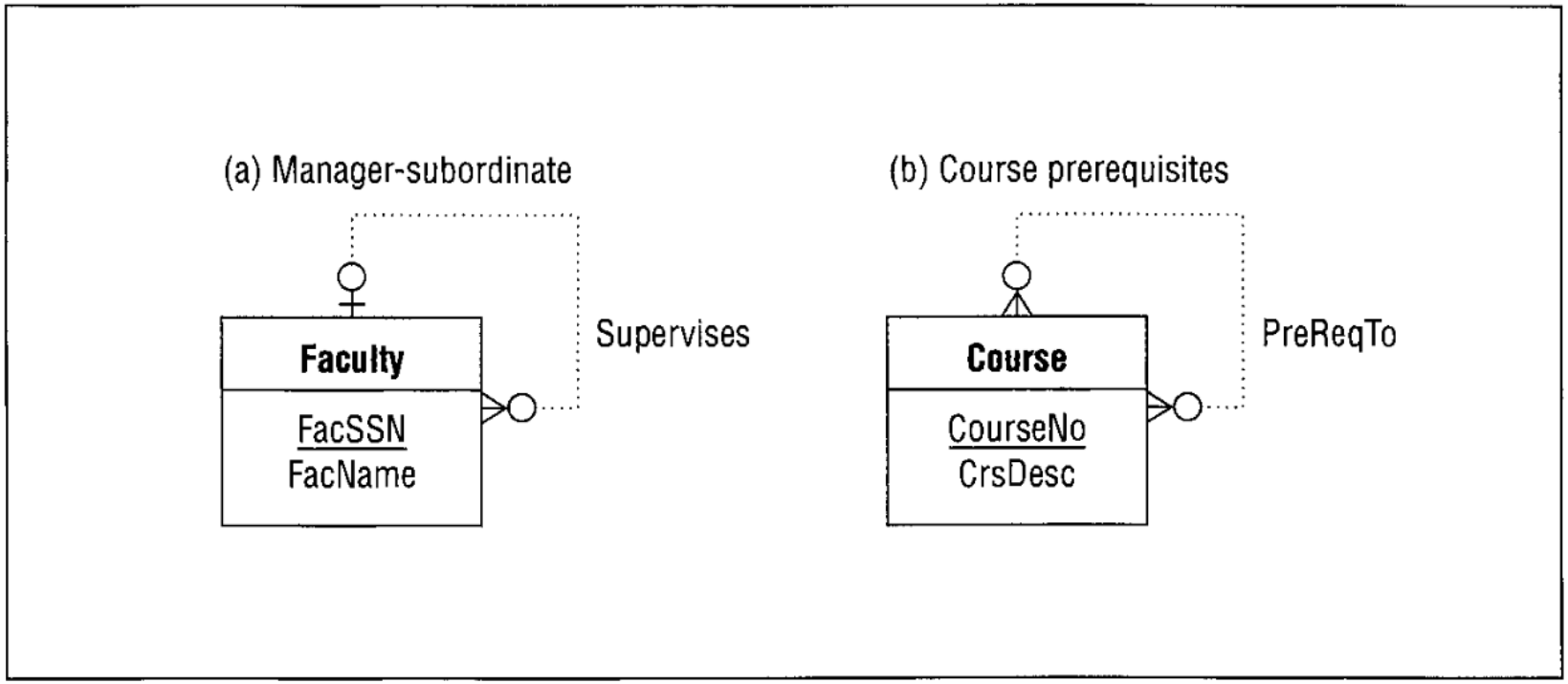
1-M Relationship with Attribute

- 1-M relationships also can have attributes, but 1-M relationships with attributes are much less common than M-N relationships with attributes.
- The *Commission* attribute is associated with the Lists relationship, not with either the Agent or the Home entity type. A home will only have a commission if an agent lists it.
- Typically, 1 -M relationships with attributes are optional for the child entity type. The Lists relationship is optional for the Home entity type.

Self-Referencing Relation

- A relationship involving the same entity type.
- Self-referencing relationships represent associations among members of the same set.
- Self-referencing relationships are sometimes called **reflexive** relationships because they are like a reflection in a mirror.

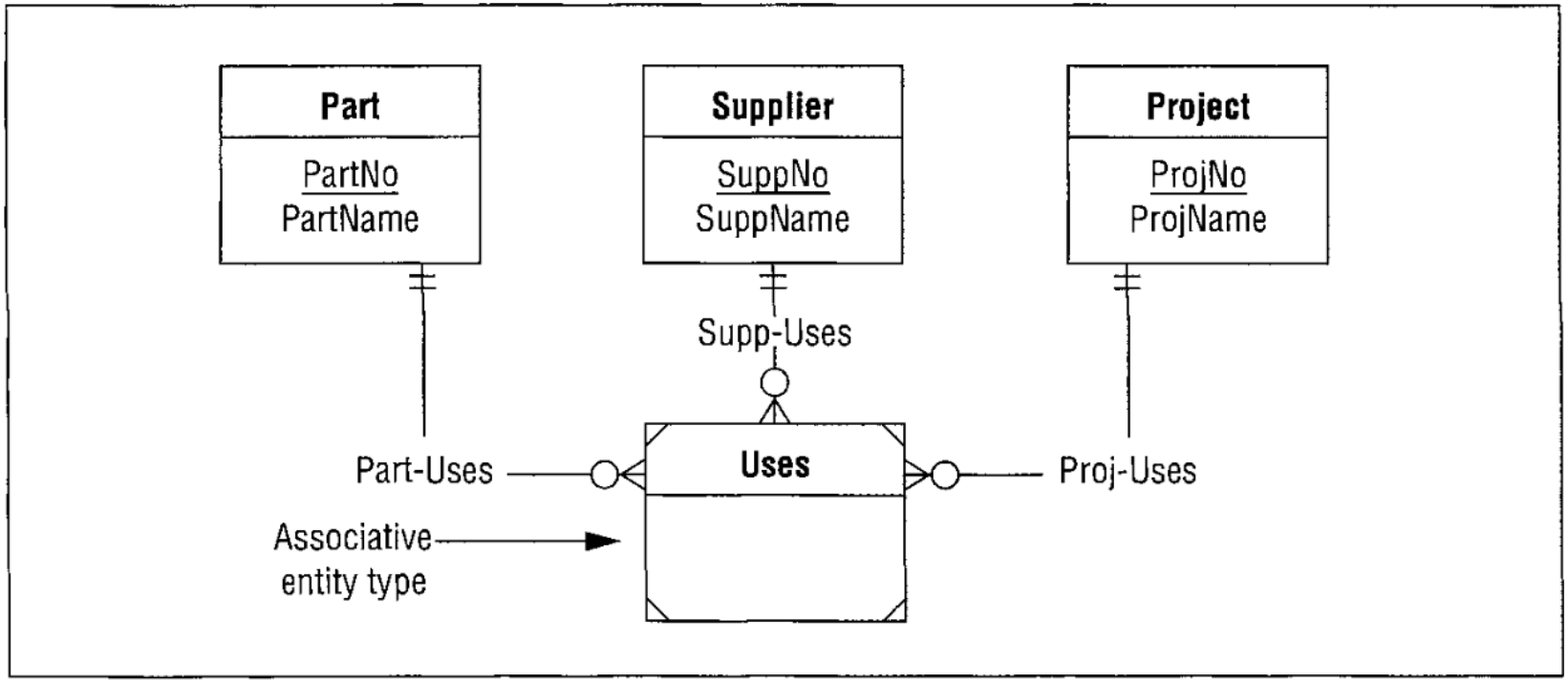
Self-Referencing Relation



Associative Entity Types

- A weak entity that depends on two or more entity types for its primary key.
- An associative entity type with more than two identifying relationships is known as an M-way associative entity type.
- Some ERD notations support relationships involving more than two entity types known as M-way (multiway) relationships where the M means more than two.

Associative Entity Types



Associative Entity Types

- 3 1-M relationships link the associative entity type, *Uses*, to the *Part*, the *Supplier*, and the *Project* entity types.
- The *Uses* entity type is associative because its role is to connect other entity types.
- Because associative entity types provide a connecting role, they are sometimes given names using active verbs.
- In addition, associative entity types are always weak as they must borrow the entire primary key.
- For example, the Uses entity type obtains its primary key through the three identifying relationships.

Relationship Equivalence

- A M-N relationship can be replaced by an associative entity type and two identifying 1-M relationships.
- In most cases the choice between a M-N relationship and the associative entity type is personal preference.

Relationship Equivalence

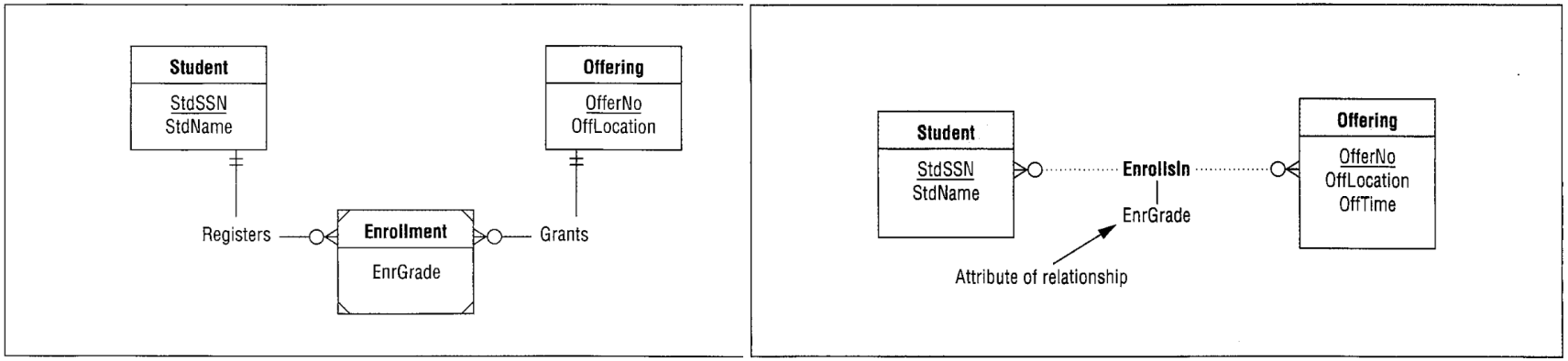
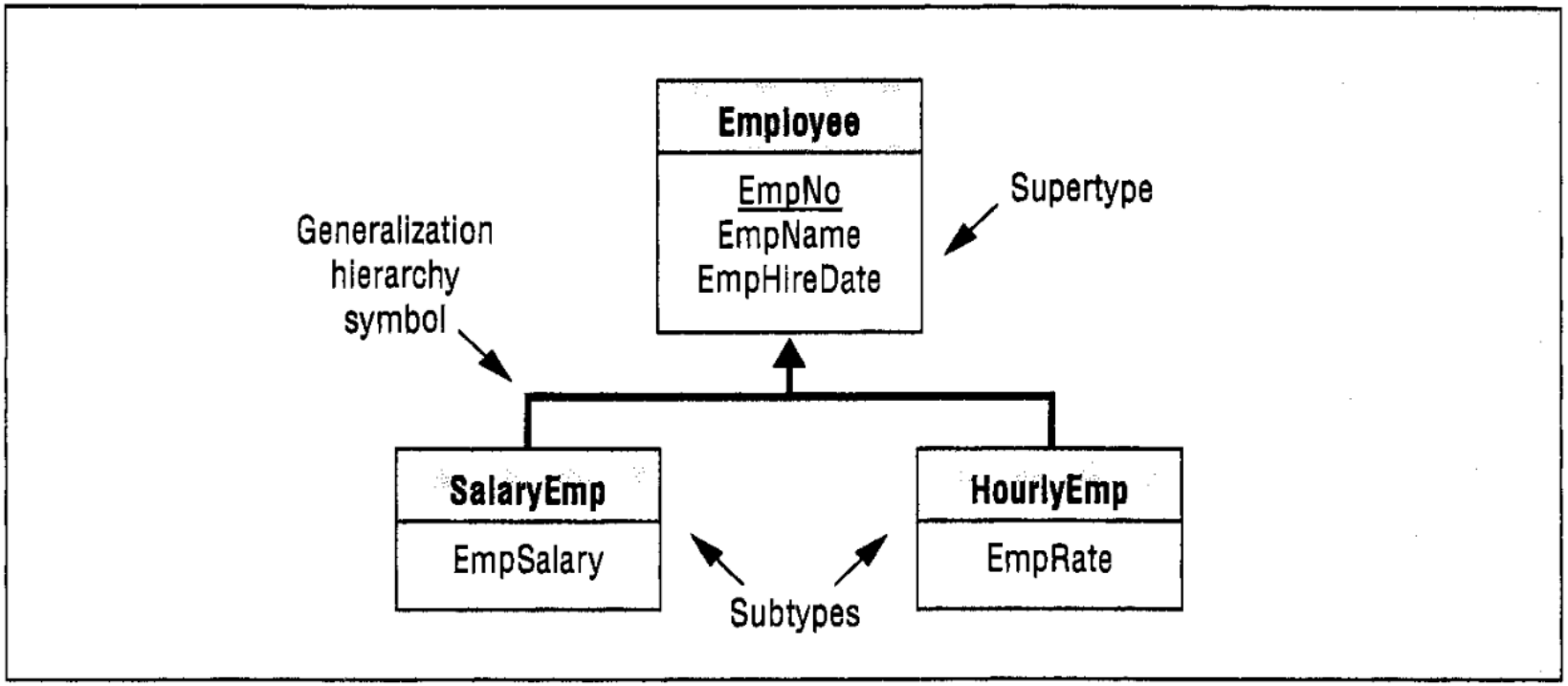


Figure shows the *Enrollsln* relationship converted to this 1-M style. In left, two identifying relationships and an associative entity type replace the *Enrollsln* relationship. The relationship name *(Enrollsln)* has been changed to a noun *(Enrollment)* to follow the convention of nouns for entity type names.

Generalization Hierarchy

- Generalization hierarchies allow entity types to be related by the level of specialization.
- A collection of entity types arranged in a hierarchical structure to show similarity in attributes.
- Each subtype or child entity type contains a subset of entities of its supertype or parent entity type.

Generalization Hierarchy



The *Employee* entity type is known as the supertype (or parent). The entity types *SalaryEmp* and *HourlyEmp* are known as the subtypes (or children). Because each subtype entity *is* a supertype entity, the relationship between a subtype and supertype is known as ISA. For example, a salaried employee is an employee.

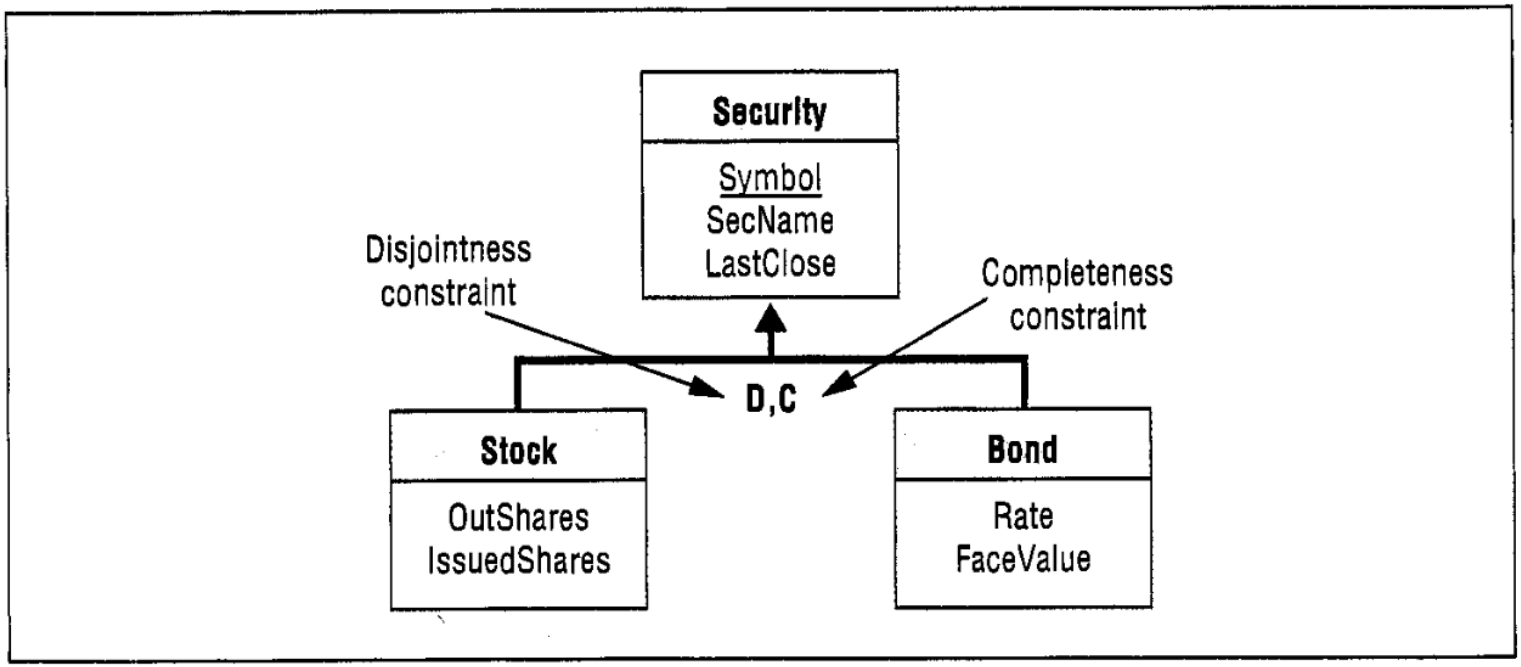
Inheritance

- A data modeling feature that supports sharing of attributes between a supertype and a sub type.
- Subtypes inherit attributes from their supertypes.
- For example, every entity of SalaryEmp has an employee number, name, and hiring date because it is also an entity of Employee.
- Inherited attributes are not shown in an ERD. Whenever you have a subtype, assume that it inherits the attributes from its supertype.

Disjointness & Completeness

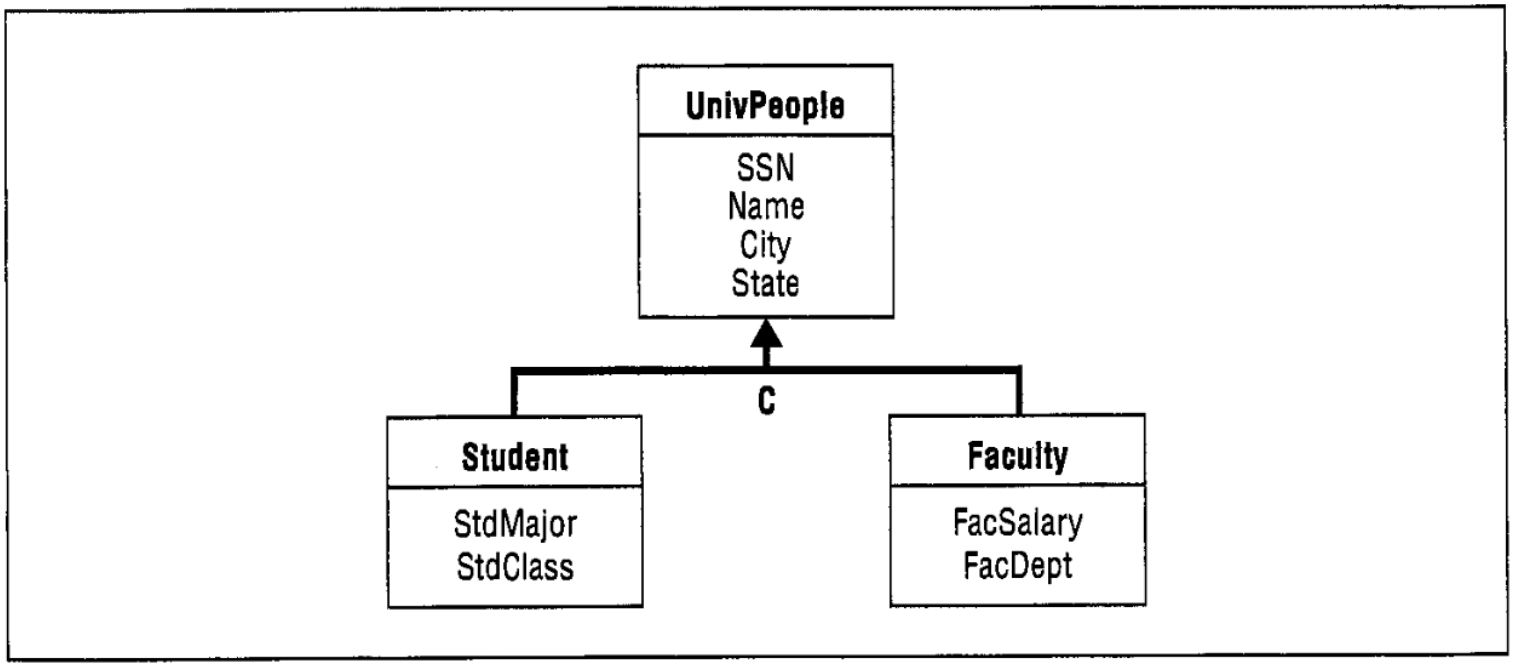
- Disjointness means that subtypes in a generalization hierarchy do not have any entities in common.
- Completeness means that every entity of a supertype must be an entity in one of the subtypes in the generalization hierarchy.

Disjointness & Completeness



The generalization hierarchy is disjoint because a security cannot be both a stock and a bond. The completeness constraint means that every security must be either a stock or a bond.

Disjointness & Completeness



This is not disjoint because teaching assistants can be considered both students and faculty. Thus, the set of students overlaps with the set of faculty.