# Database Design:
# The Entity-Relationship Approach

# Chapter 7: Entity-Relationship Model

- Design Process
- Modeling
- Constraints
- E-R Diagram
- Design Issues
- Weak Entity Sets
- Extended E-R Features
- Design of the Bank Database
- Reduction to Relation Schemas
- Database Design
- UML

# Design Phases

- The initial phase of database design is to characterize fully the data needs of the prospective database users.
- Next, the designer chooses a data model and, by applying the concepts of the ch osen data model, translates these requirements into a conceptual schema of the database.
- A fully developed conceptual schema also indicates the functional requirements of the enterprise. In a “specification of functional requirements”, users describe the kinds of operations (or transactions) that will be performed on the data.

# Design Phases (Cont.)

- The process of moving from an abstract data model to the implementation of th e database proceeds in two final design phases.

- Logical Design – Deciding on the database schema. Database design requires t hat we find a “good” collection of relation schemas.
  - Business decision – What attributes should we record in the database?
  - Computer Science decision – What relation schemas should we have and how shoul d the attributes be distributed among the various relation schemas?
- Physical Design – Deciding on the physical layout of the database

# Design Approaches

- Entity Relationship Model (covered in this chapter)
  - Models an enterprise as a collection of *entities* and *relationships*
    - Entity: a “thing” or “object” in the enterprise that is distinguishable from other objects
      - Described by a set of *attributes*
    - Relationship: an association among several entities
  - Represented diagrammatically by an *entity-relationship diagram:*
- Normalization Theory (Chapter 8)
  - Formalize what designs are bad, and test for them

# ER model -- Database Modeling

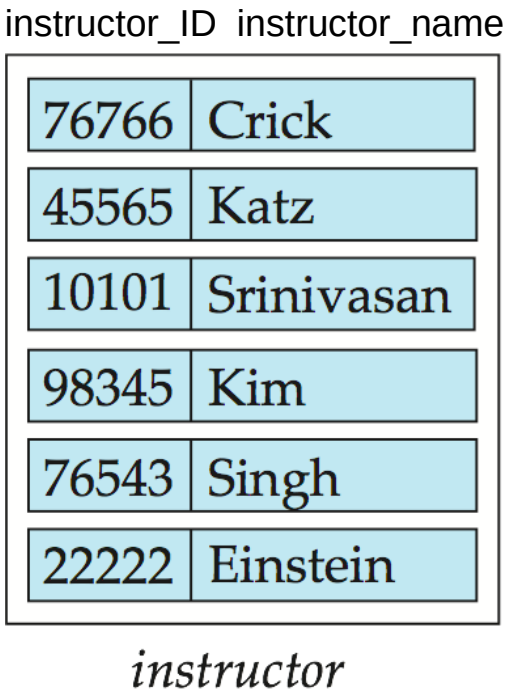
- The ER data model was developed to facilitate database design by allowing spe cification of an enterprise schema that represents the overall logical structure of a database.
- The ER model is very useful in mapping the meanings and interactions of real-world enterprises onto a conceptual schema. Because of this usefulness, many database-design tools draw on concepts from the ER model.
- The ER data model employs three basic concepts:
  - entity sets,
  - relationship sets,
  - attributes.
- The ER model also has an associated diagrammatic representation, the ER diagr am, which can express the overall logical structure of a database graphically.

# Entity Sets

- An **entity** is an object that exists and is distinguishable from other objects.
  - Example: specific person, company, event, plant
- An **entity set** is a set of entities of the same type that share the same properties.
  - Example: set of all persons, companies, trees, holidays
- An entity is represented by a set of attributes; i.e., descriptive properties possessed by all members of an entity set.
  - Example:

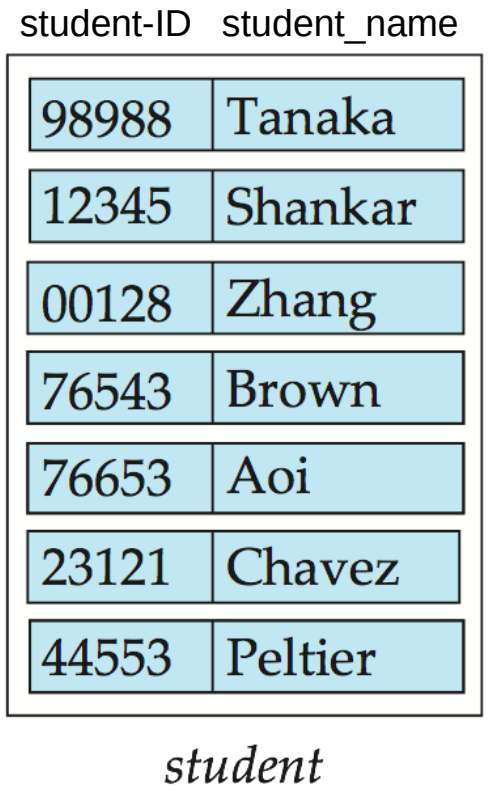
    *instructor* = (*ID*, *name*, *street*, *city*, *salary* )
    *course*= (*course_id*, *title*, *credits*)
- A subset of the attributes form a **primary key** of the entity set; i.e., uniquely identifiying each member of the set.

# Entity Sets -- *instructor* and *student*



| instructor_ID | instructor_name |
|---|---|
| 76766 | Crick |
| 45565 | Katz |
| 10101 | Srinivasan |
| 98345 | Kim |
| 76543 | Singh |
| 22222 | Einstein |

*instructor*



| student-ID | student_name |
|---|---|
| 98988 | Tanaka |
| 12345 | Shankar |
| 00128 | Zhang |
| 76543 | Brown |
| 76653 | Aoi |
| 23121 | Chavez |
| 44553 | Peltier |

*student*

# Relationship Sets

- A **relationship** is an association among several entities

  Example:

| 44553 (Peltier) | *advisor* | 22222 (Einstein) |
|---|---|---|
| *student* entity | relationship set | *instructor* entity |

- A **relationship set** is a mathematical relation among $n \geq 2$ entities, each taken f rom entity sets

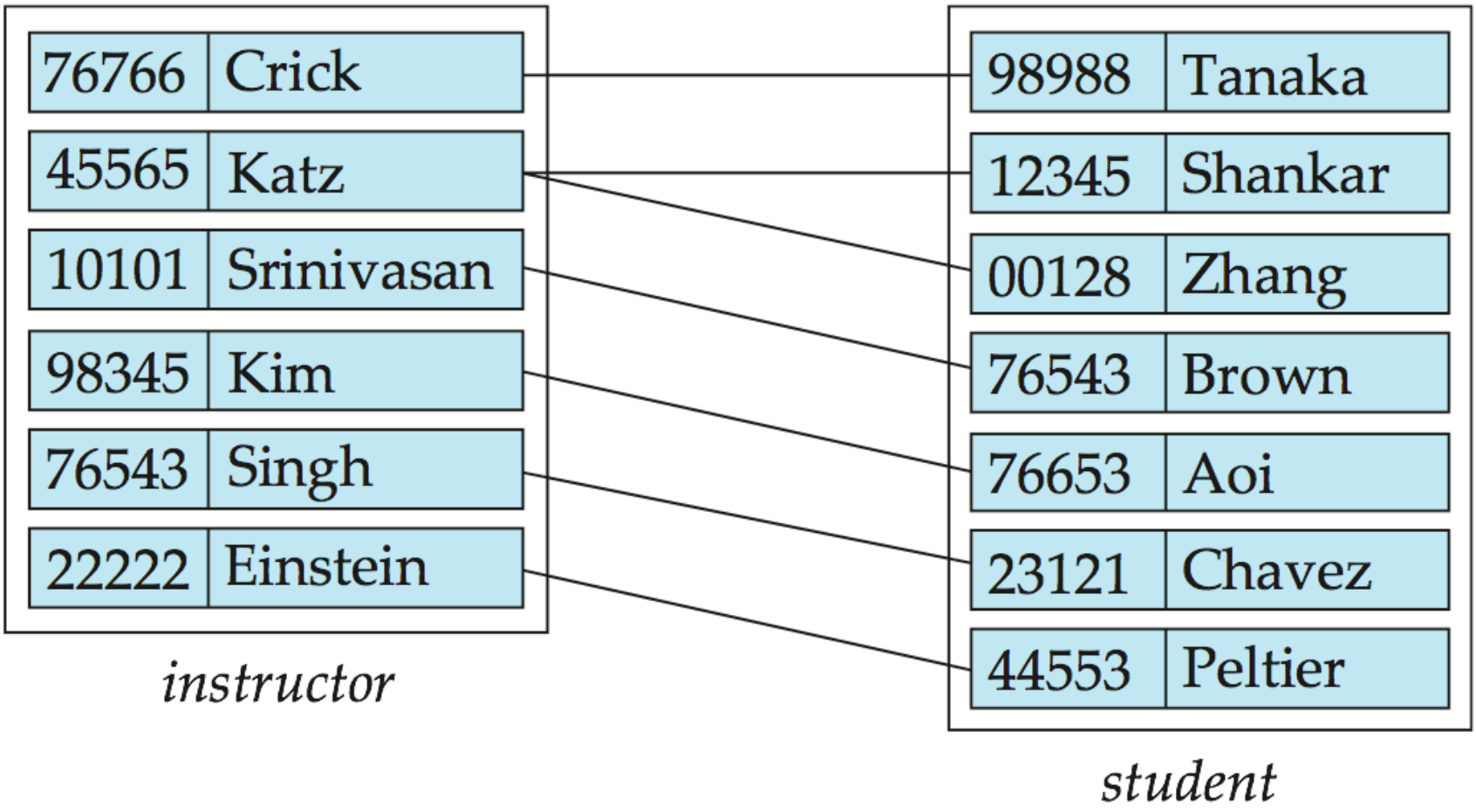
$$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$$

  where $(e_1, e_2, \ldots, e_n)$ is a relationship

  - Example:

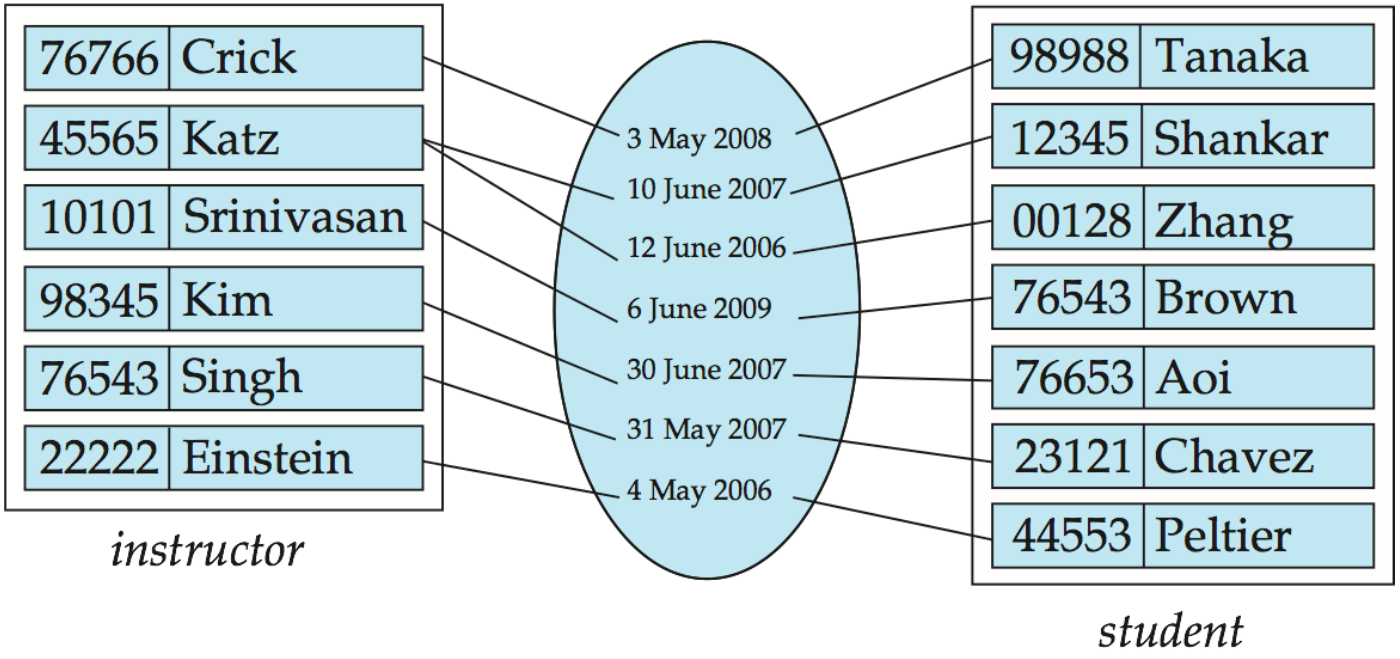
    (44553,22222) $\in$ *advisor*

# Relationship Set *advisor*



# Relationship Sets (Cont.)

- An attribute can also be associated with a relationship set.
- For instance, the *advisor* relationship set between entity sets *instructor* and *stud ent* may have the attribute *date* which tracks when the student started being asso ciated with the advisor



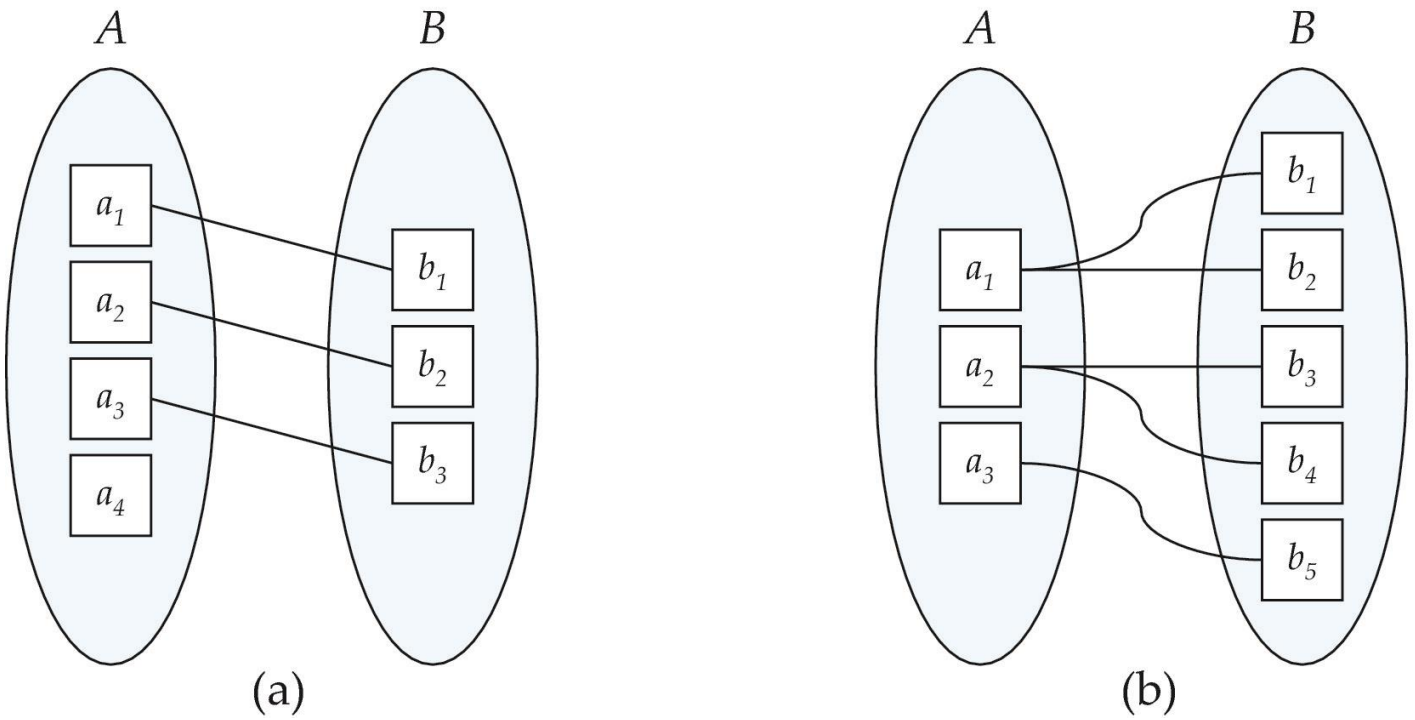
# Degree of a Relationship Set

- binary relationship
  - involve two entity sets (or degree two).
  - most relationship sets in a database system are binary.
- Relationships between more than two entity sets are rare. Most relationships ar e binary. (More on this later.)
  - Example: *students* work on research *projects* under the guidance of an *instructor*.
  - relationship *proj_guide* is a ternary relationship between *instructor*, *student*, and *proj ect*

# Mapping Cardinality Constraints

- Express the number of entities to which another entity can be associated via a re lationship set.
- Most useful in describing binary relationship sets.
- For a binary relationship set the mapping cardinality must be one of the followi ng types:
  - One to one
  - One to many
  - Many to one
  - Many to many

# Mapping Cardinalities

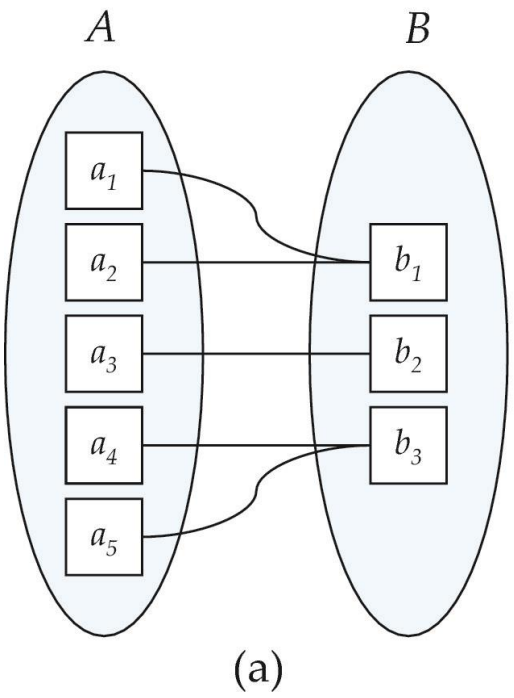



One to one

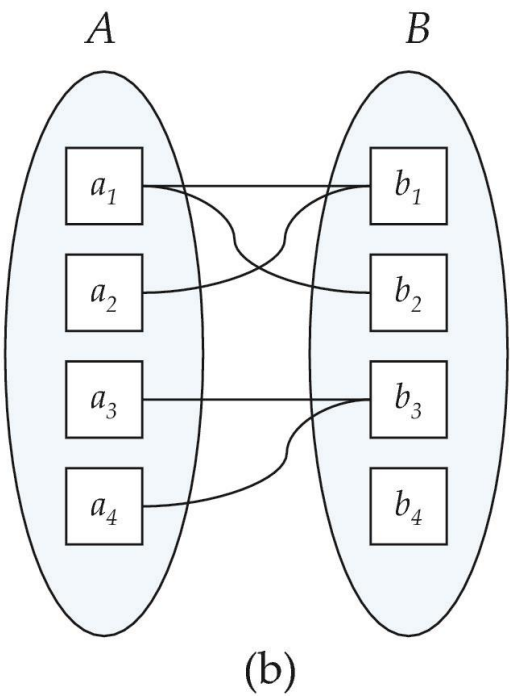
One to many

Note: Some elements in *A* and *B* may not be mapped to any elements in the other set

# Mapping Cardinalities




Many to one



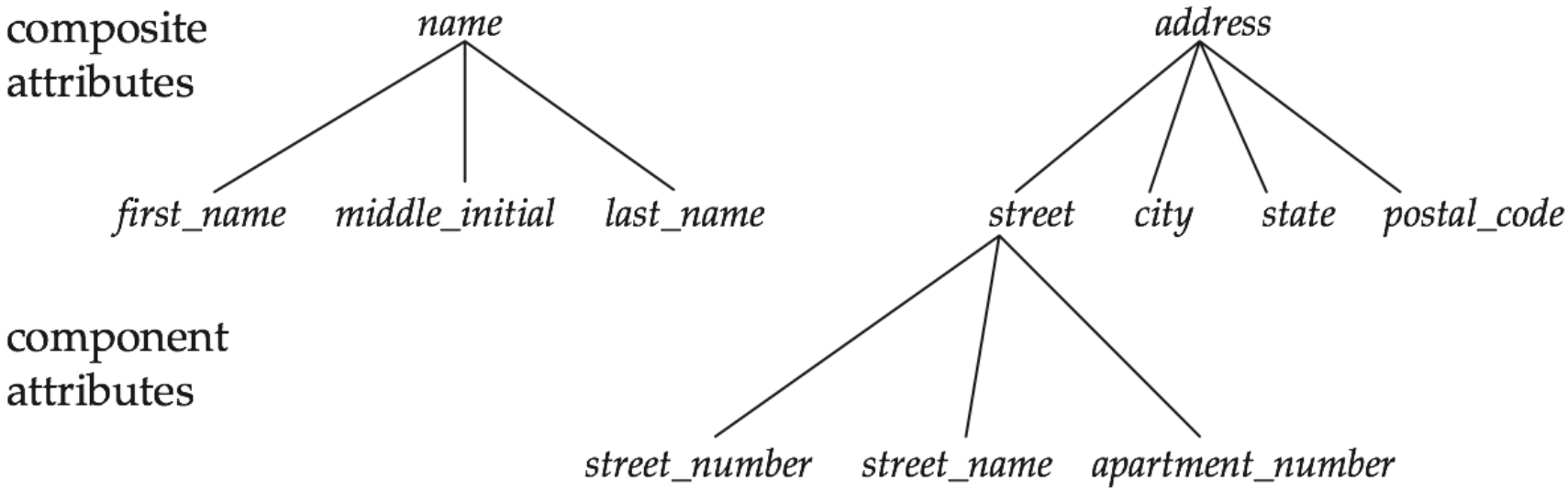

Many to many

Note: Some elements in A and B may not be mapped to any elements in the other set

# Complex Attributes

- Attribute types:
  - **Simple** and **composite** attributes.
  - **Single-valued** and **multivalued** attributes
    - Example: multivalued attribute: *phone_numbers*
  - **Derived** attributes
    - Can be computed from other attributes
    - Example:  age, given date_of_birth
- **Domain** – the set of permitted values for each attribute

# Composite Attributes

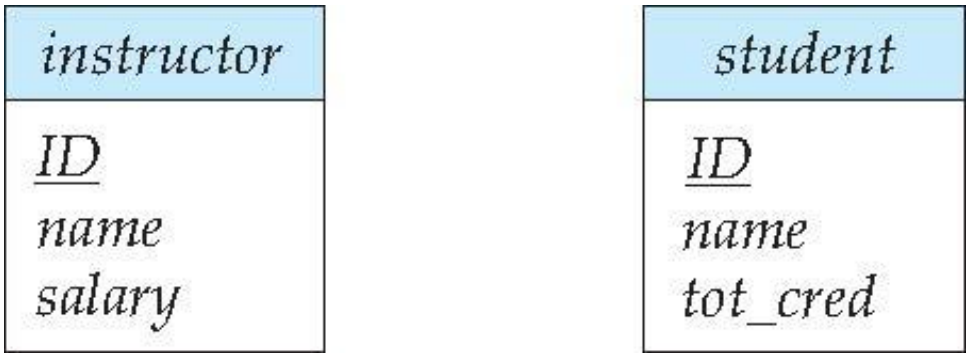


# Redundant Attributes

- Suppose we have entity sets:
  - *instructor*, with attributes: *ID*, *name*, *dept_name*, *salary*
  - *department*, with attributes: *dept_name*, *building*, *budget*
- We model the fact that each instructor has an associated department using a relationship set *inst_dept*
- The attribute *dept_name* appears in both entity sets. Since it is the primary key for the entity set *department*, it replicates information present in the relationship and is therefore redundant in the entity set *instructor* and needs to be removed.
- BUT: when converting back to tables, in some cases the attribute gets reintroduced, as we will see later.

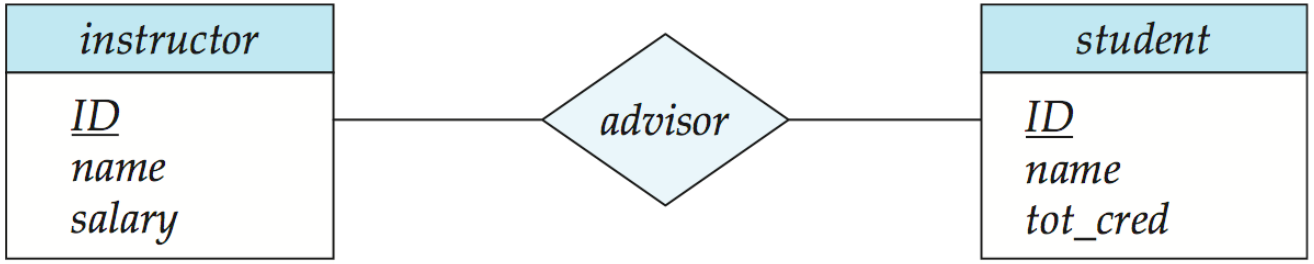
# Entity Sets

- Entities can be represented graphically as follows:
  - Rectangles represent entity sets.
  - Attributes listed inside entity rectangle
  - Underline indicates primary key attributes

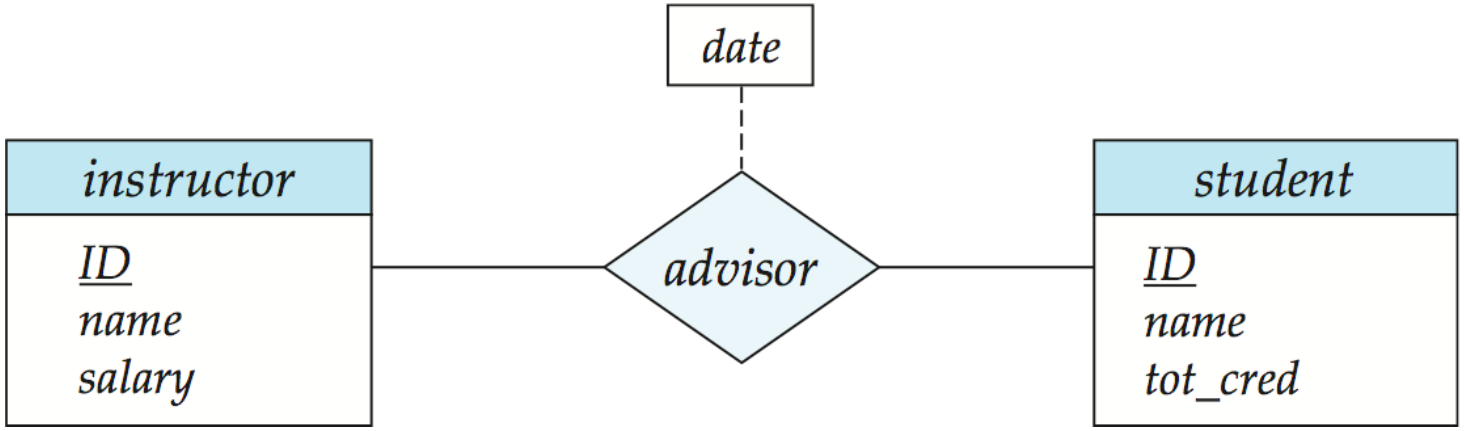


# Relationship Sets

- Diamonds represent relationship sets.



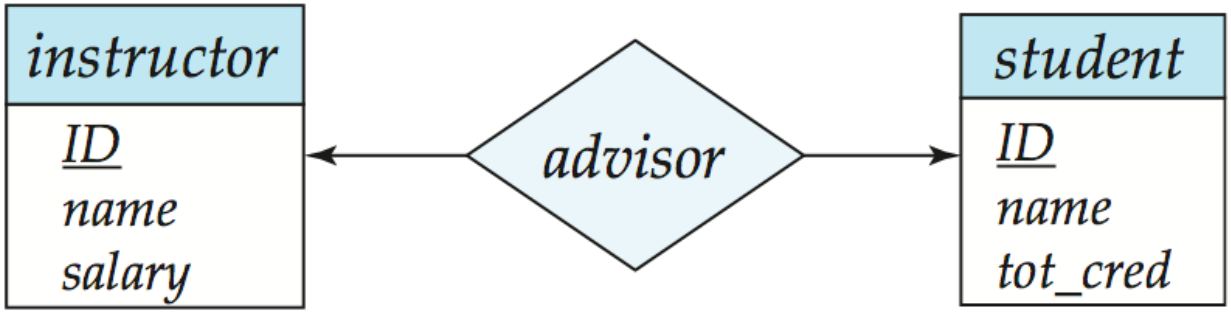
# Relationship Sets with Attributes



# Cardinality Constraints

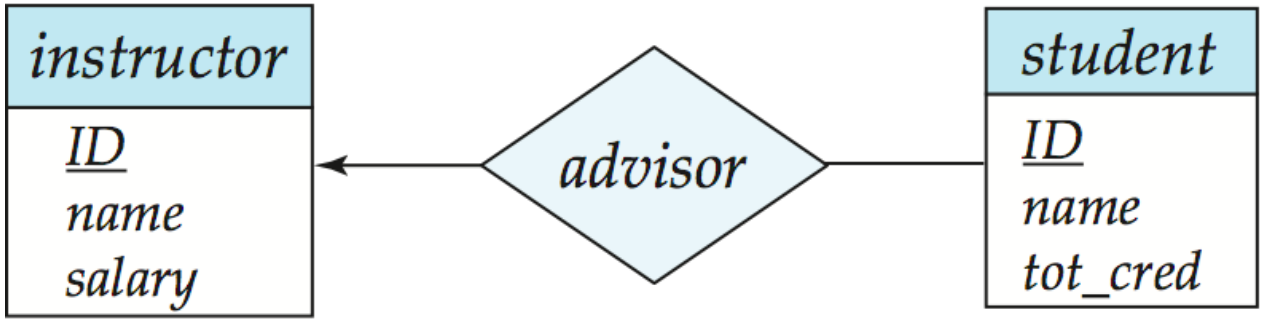
- We express cardinality constraints by drawing either a directed line (→), signif ying "one," or an undirected line (—), signifying "many," between the relations hip set and the entity set.

- One-to-one relationship between an *instructor* and a *student* :
  - A *student* is associated with at most one *instructor* via the relationship *advisor*
  - A *instructor* is associated with at most one *student* via the relationship *advisor*



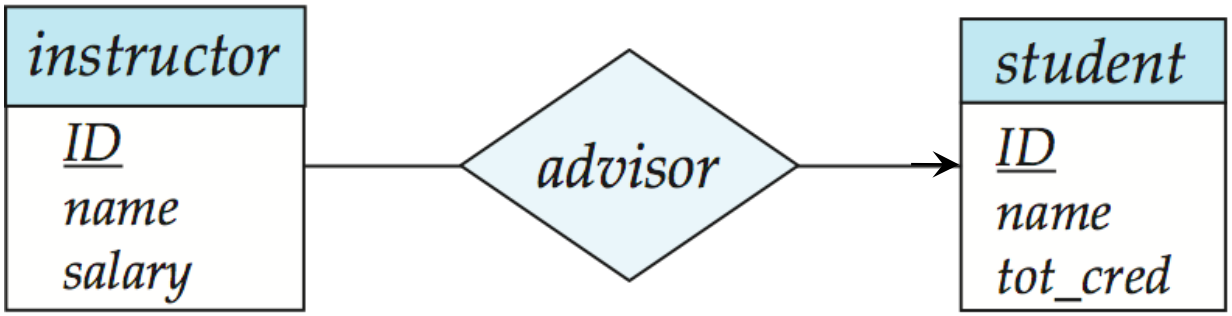
# One-to-Many Relationship

- one-to-many relationship between an *instructor* and a *student*
  - an instructor is associated with several (including 0) students via *advisor*
  - a student is associated with at most one instructor via advisor,



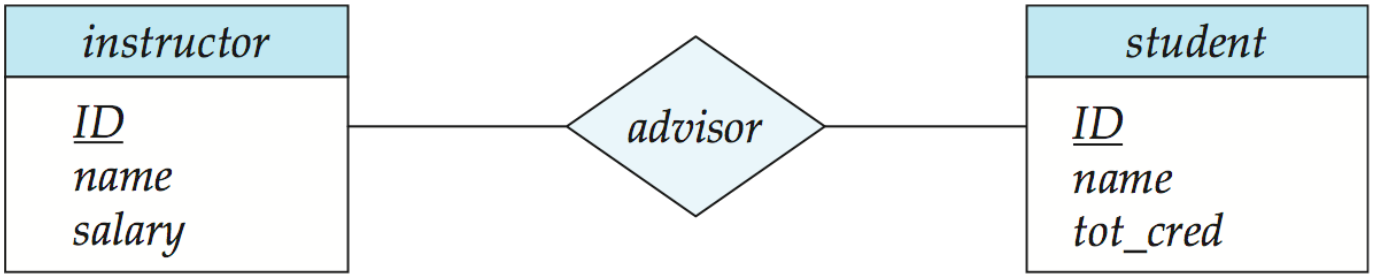
# Many-to-One Relationships

- In a many-to-one relationship between an *instructor* and a *student*,
  - an instructor is associated with at most one student via *advisor*,
  - and a student is associated with several (including 0) instructors via *advisor*



# Many-to-Many Relationship

- An instructor is associated with several (possibly 0) students via *advisor*
- A student is associated with several (possibly 0) instructors via *advisor*



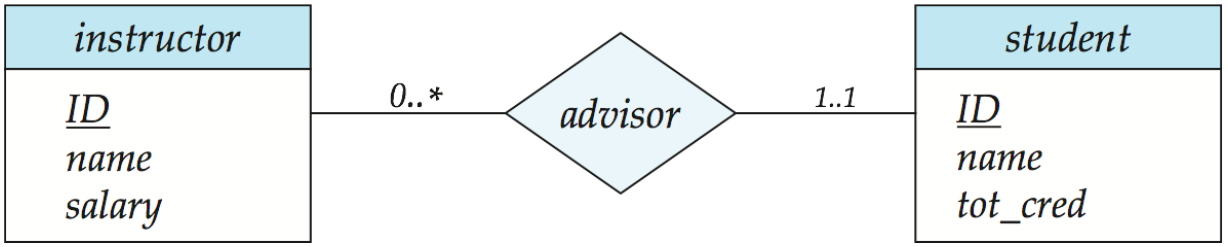
# Total and Partial Participation

- Total participation (indicated by double line): every entity in the entity set parti cipates in at least one relationship in the relationship set


  - participation of *student* in *advisor* relation is total
    - every *student* must have an associated instructor
- Partial participation: some entities may not participate in any relationship in the relationship set
  - Example: participation of *instructor* in *advisor* is partial

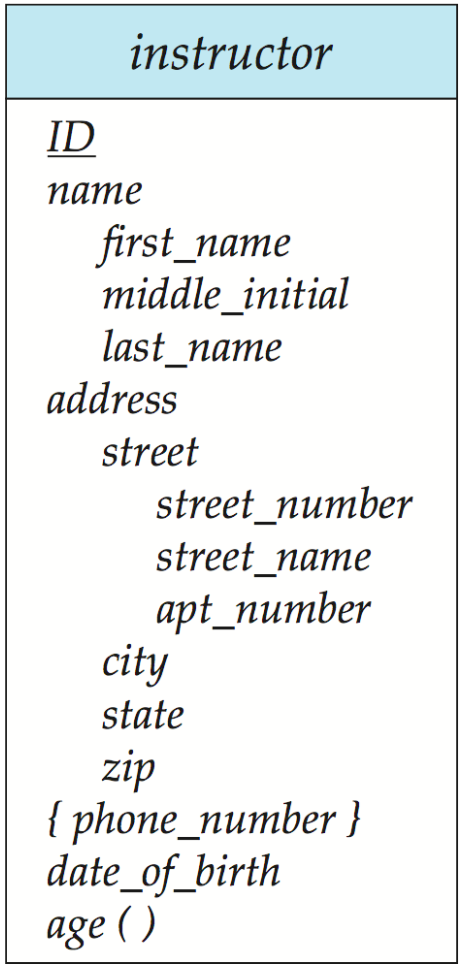
# Notation for Expressing More Complex Constraints

- A line may have an associated minimum and maximum cardinality, shown in th e form *l..h*, where *l* is the minimum and *h* the maximum cardinality
  - A minimum value of 1 indicates total participation.
  - A maximum value of 1 indicates that the entity participates in at most one r elationship
  - A maximum value of * indicates no limit.



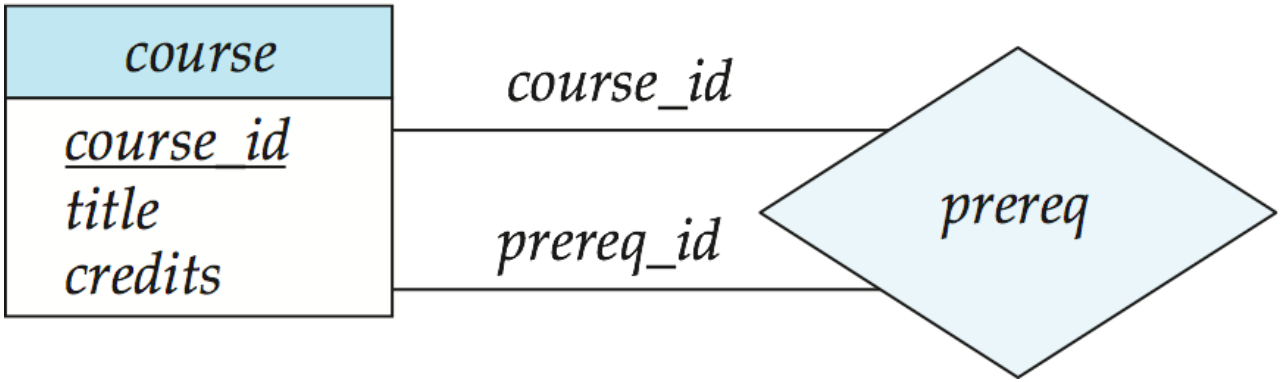

Instructor can advise 0 or more students. A student must have 1 advisor; cannot have multiple advisors

# Notation to Express Entity with Complex Attributes



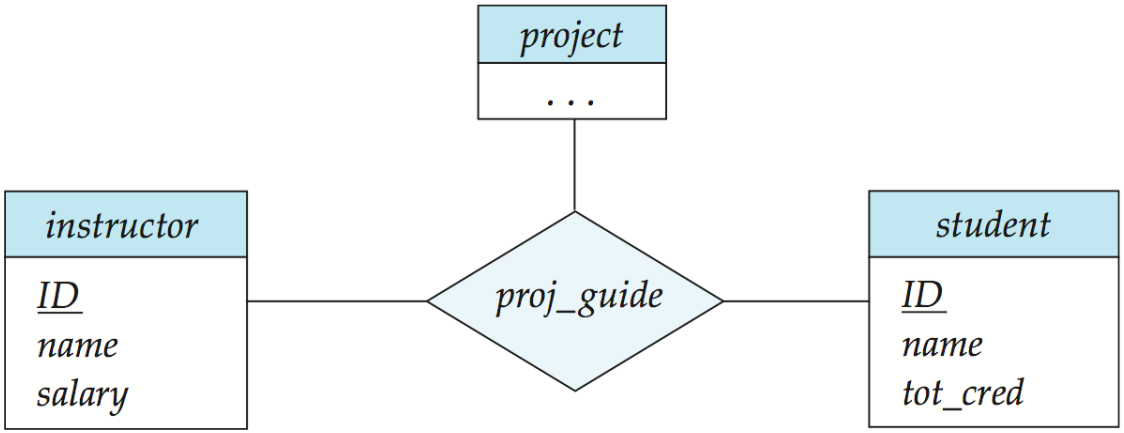
# Roles

- Entity sets of a relationship need not be distinct
  - Each occurrence of an entity set plays a "role" in the relationship
- The labels "*course_id*" and "*prereq_id*" are called **roles**.



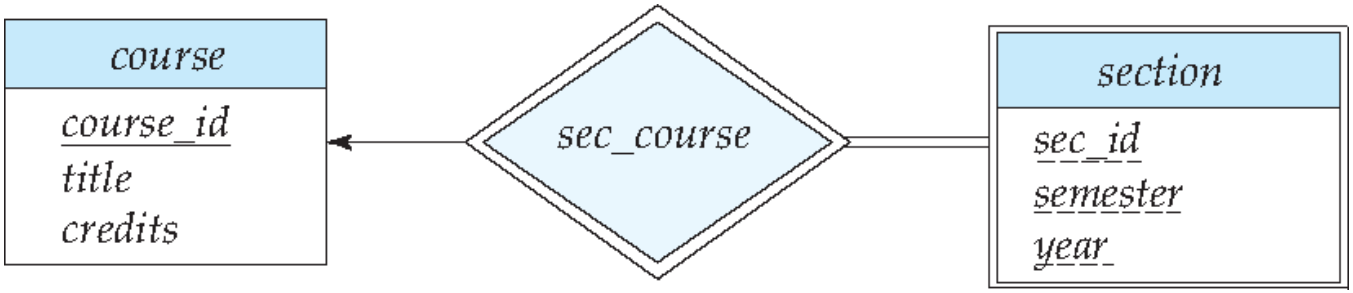
# Non-binary Relationship Sets

- Most relationship sets are binary
- There are occasions when it is more convenient to represent relationships as no n-binary.
- E-R Diagram with a Ternary Relationship



# Expressing Weak Entity Sets

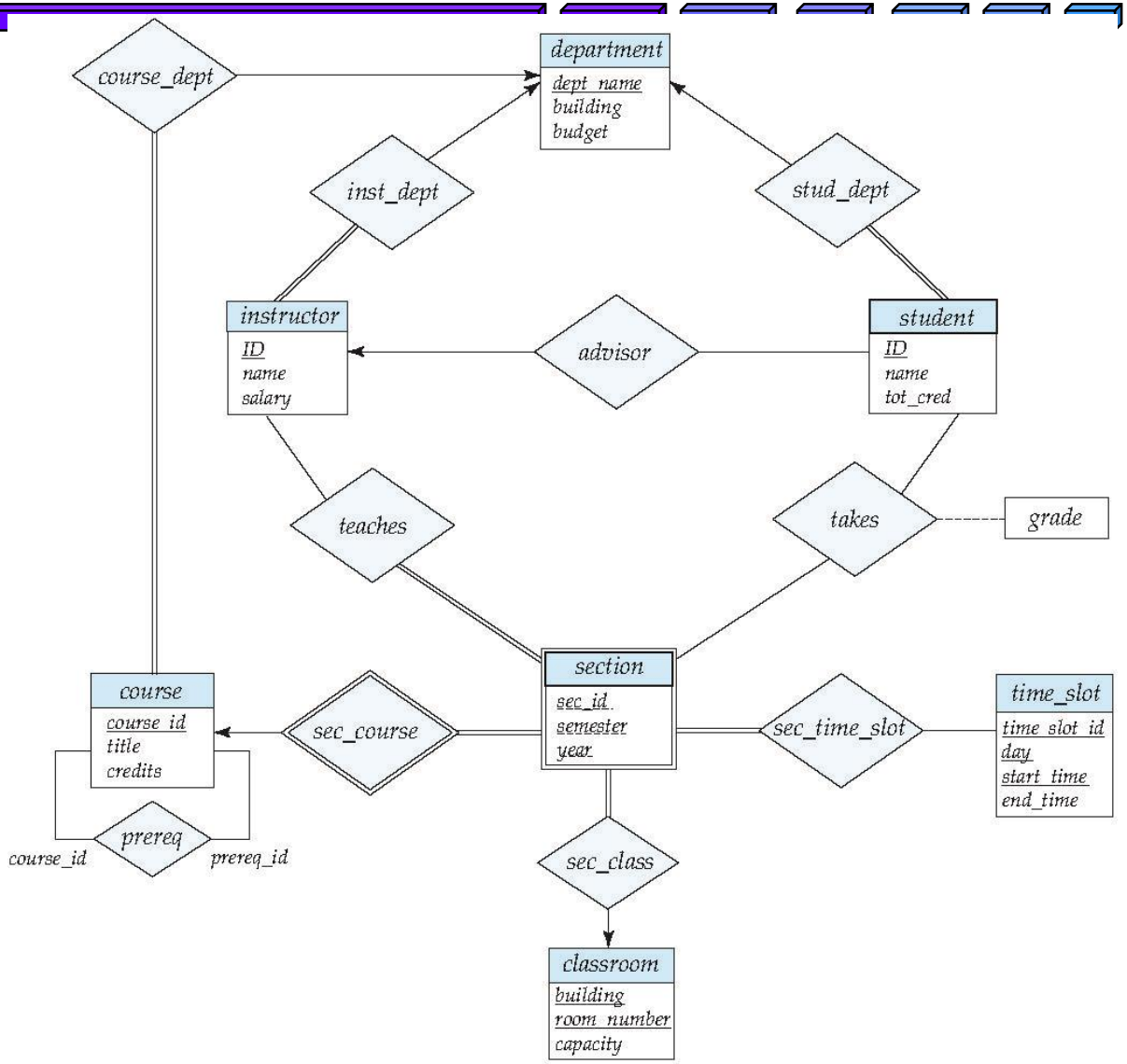
- In E-R diagrams, a weak entity set is depicted via a double rectangle.
- We underline the discriminator of a weak entity set with a dashed line.
- The relationship set connecting the weak entity set to the identifying strong enti ty set is depicted by a double diamond.
- Primary key for *section* – (*course_id*, *sec_id*, *semester*, *year*)



# Weak Entity Sets

- Consider a *section* entity, which is uniquely identified by a *course_id*, *semester*, *year*, and *sec_id*.
- Clearly, section entities are related to course entities. Suppose we create a relati onship set *sec_course* between entity sets *section* and *course*.
- Note that the information in *sec_course* is redundant, since *section* already has an attribute *course_id*, which identifies the course with which the section is rela ted.
- One option to deal with this redundancy is to get rid of the relationship s*ec_cou rse*; however, by doing so the relationship between *section* and *course* becomes implicit in an attribute, which is not desirable.

# E-R Diagram for a University Enterprise



# Reduction to Relation Schemas

- Entity sets and relationship sets can be expressed uniformly as *relation schemas* that represent the contents of the database.
- A database which conforms to an E-R diagram can be represented by a collectio n of schemas.
- For each entity set and relationship set there is a unique schema that is assigned the name of the corresponding entity set or relationship set.
- Each schema has a number of columns (generally corresponding to attributes), which have unique names.

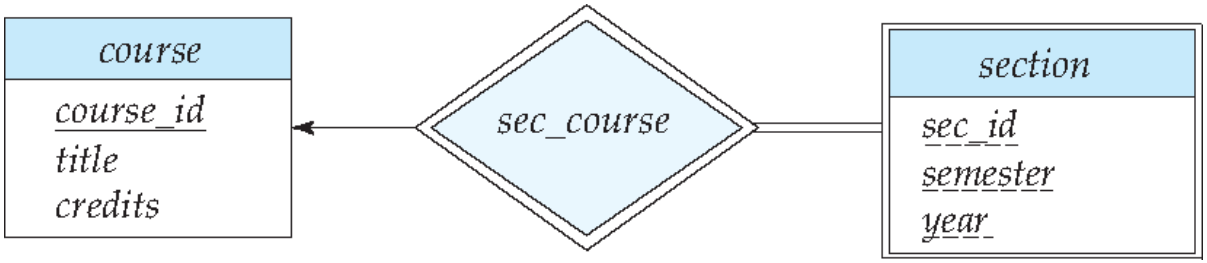
# Representing Entity Sets

- A strong entity set reduces to a schema with the same attributes

  *student(ID, name, tot_cred)*

- A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set

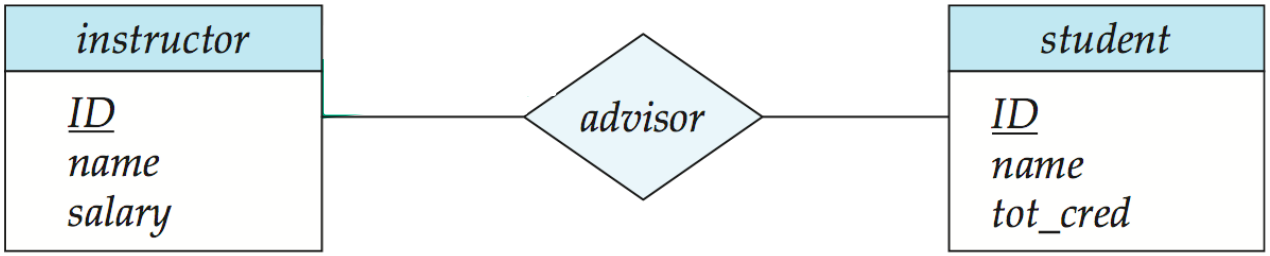
  *section ( course_id, sec_id, sem, year )*



# Representing Relationship Sets

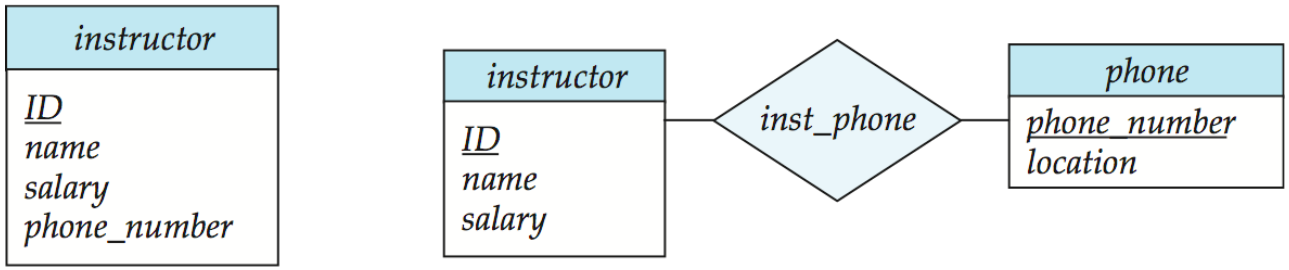
- A many-to-many relationship set is represented as a schema with attributes for t he primary keys of the two participating entity sets, and any descriptive attribut es of the relationship set.
- Example: schema for relationship set *advisor*

*advisor* = (*s_id*, *i_id*)



# Entities vs. Attributes

- Use of entity sets vs. attributes

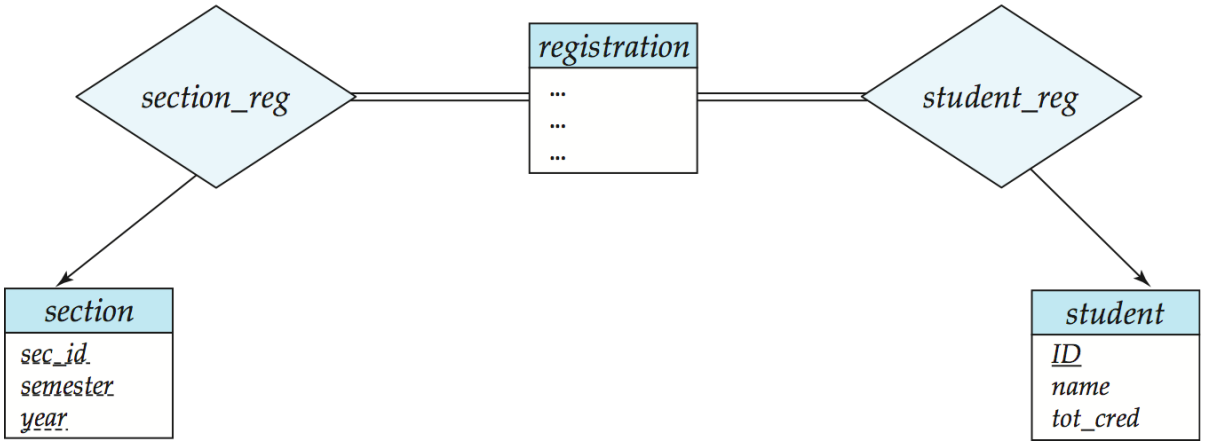



- Use of phone as an entity allows extra information about phone numbers (plus multiple phone numbers)

# Entities vs. Relationship sets

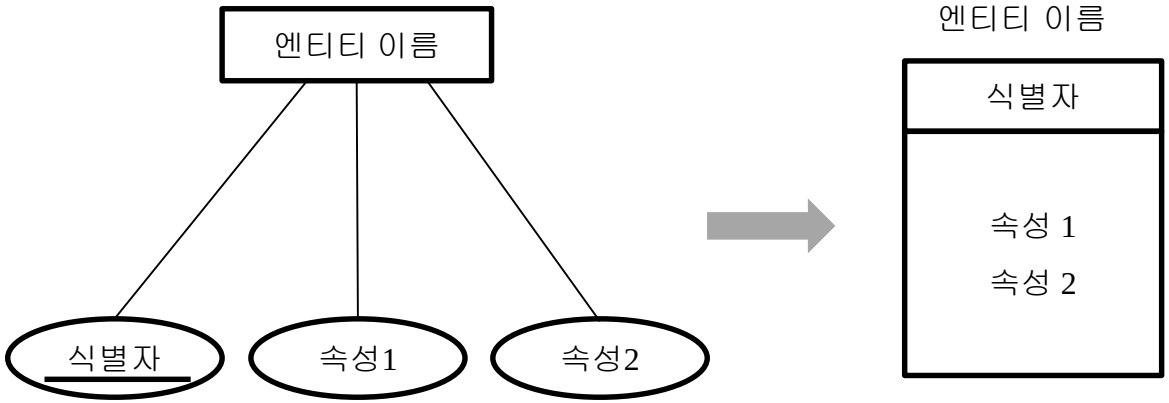
- **Use of entity sets vs. relationship sets**

  Possible guideline is to designate a relationship set to describe an action that oc curs between entities




- **Placement of relationship attributes**
  - For example, attribute date as attribute of advisor or as attribute of student

# Information Engineering Notations

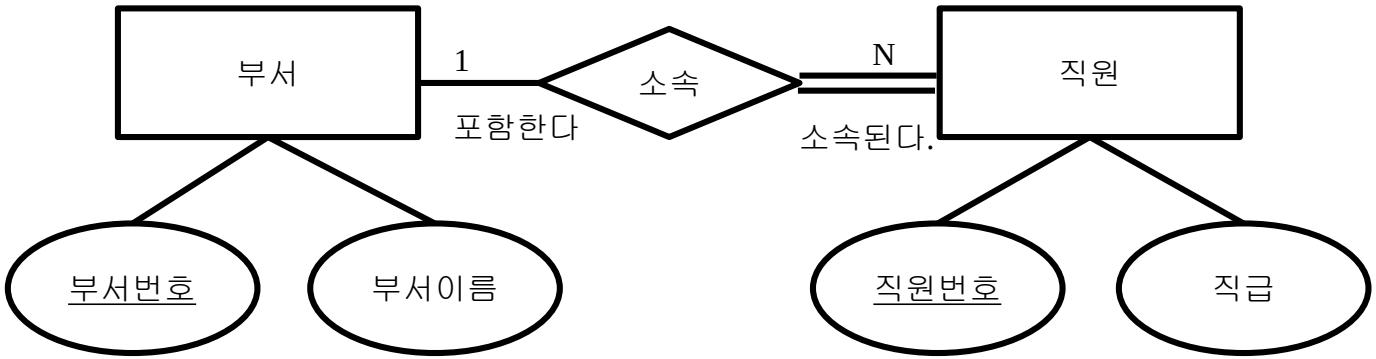


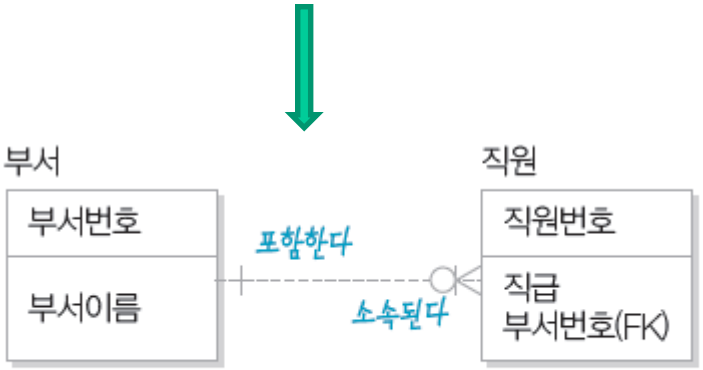

**Peter Chen 표기법** **IE 표기법**

# Information Engineering Notations

| 기호 | 의미 |
| --- | --- |
|  | • 비식별자 관계(non-identifying relationship) : 강한 개체 타입<br>• 부모 개체의 키가 일반 속성으로 포함되는 관계 |
|  | • 식별자 관계(identifying relationship) : 약한 개체 타입<br>• 부모 개체의 키가 주식별자로 포함되는 관계 |
|  | • 일대다(1:N)의 관계 : N 쪽에 새발을 표시 |
|  | • 0(선택 참여), 최소 참여가 0일 경우 |
|  | • 1(필수 참여), 최소 참여가 1일 경우 |

# Information Engineering Notations




(b) IE 표기법으로 작성한 직원–부서 관계