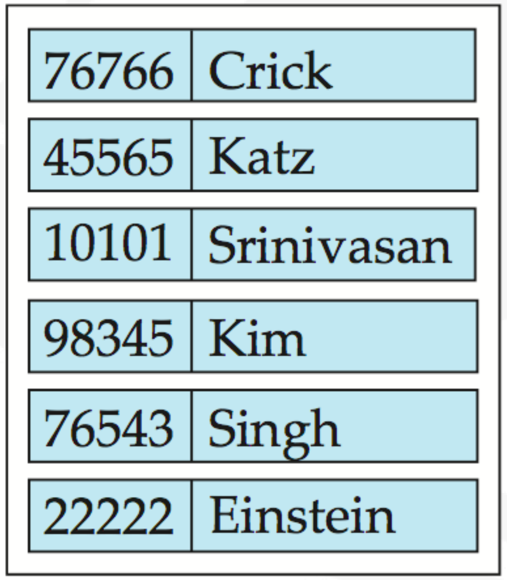
Entity-Relationship Model

Universidad de Alcalá

Modeling

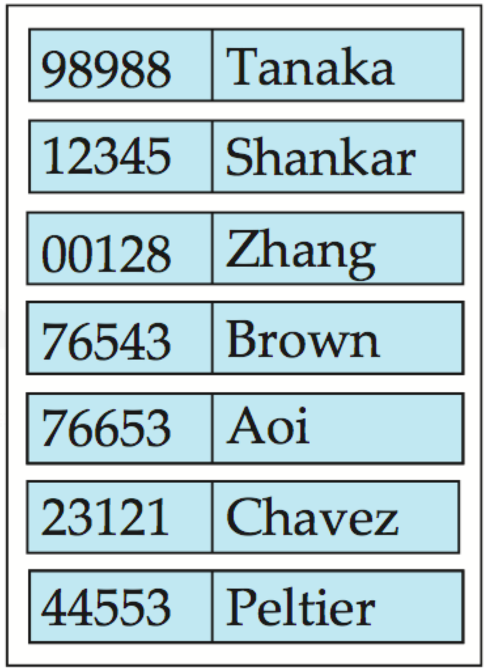
- A *database* can be modeled as:
  - a collection of entities,
  - relationships among entities.
- An **entity** is an object that exists and is distinguishable from other objects.
  - Example: specific person, company, event, plant
- Entities have **attributes**
  - Example: people have *names* and *addresses*

Entities *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

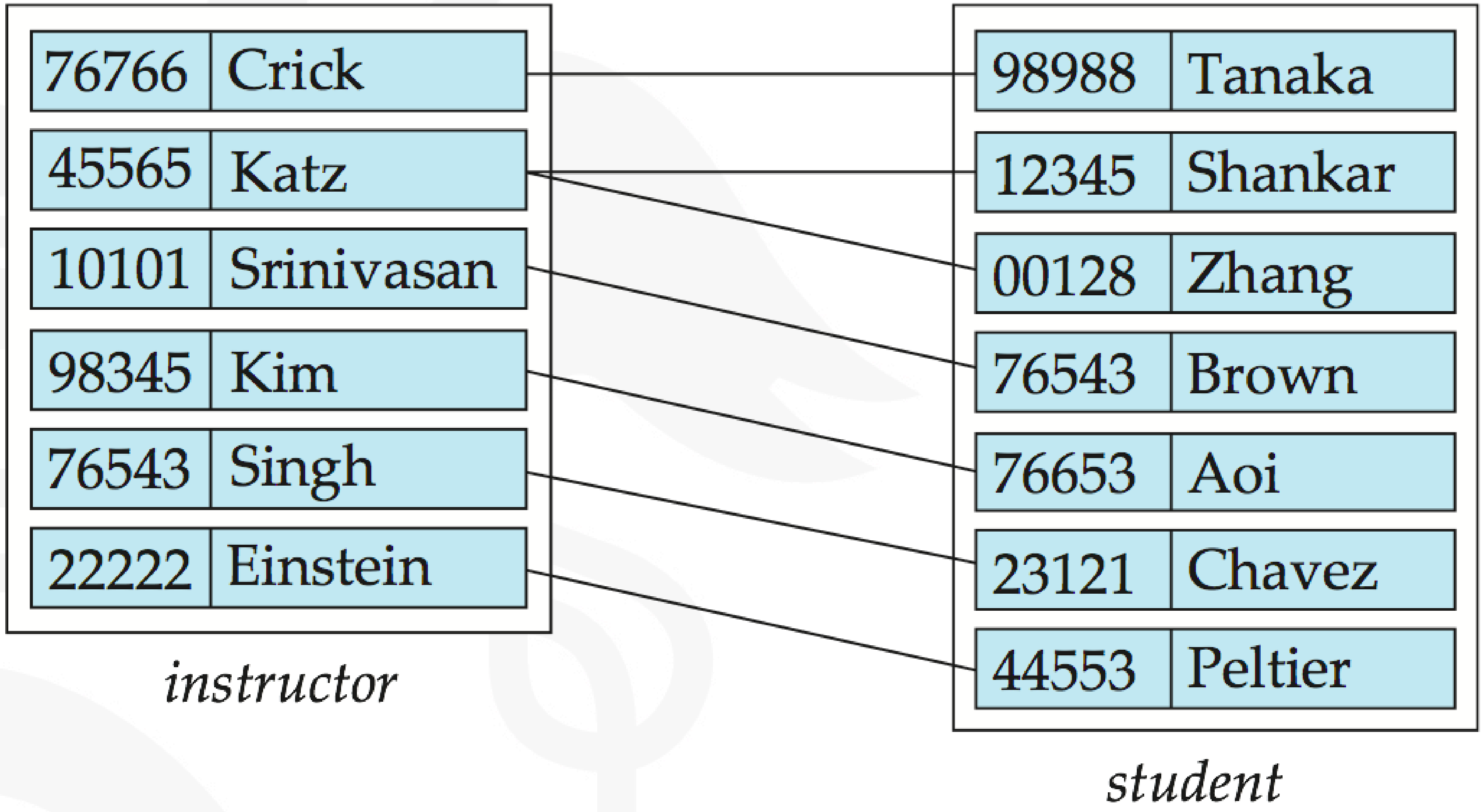
Relationships

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

Example:

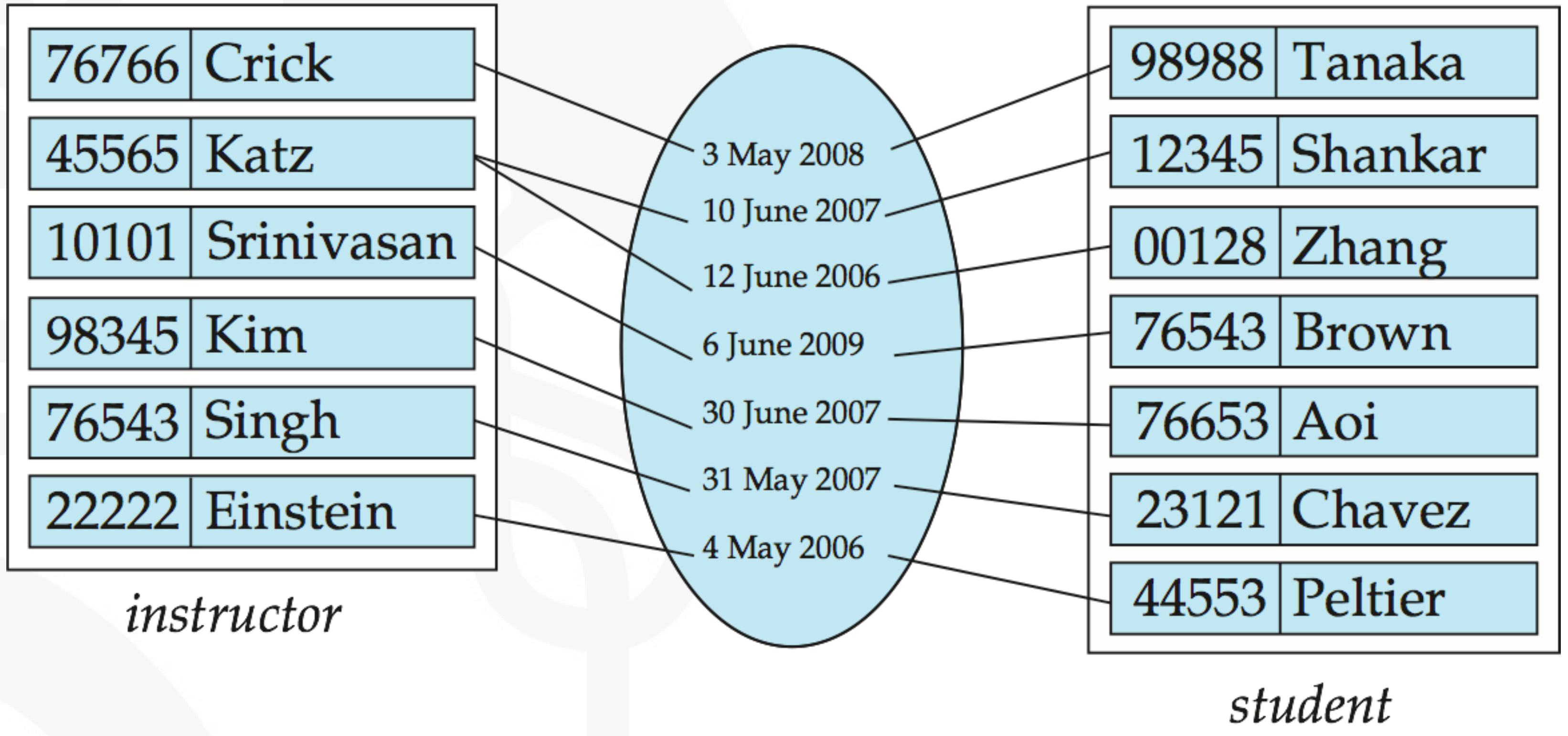
44553 (Peltier)	*teaches*	22222 (Einstein)
student entity	relationship set	*instructor* entity

Relationship *teach*



Relationships (Cont.)

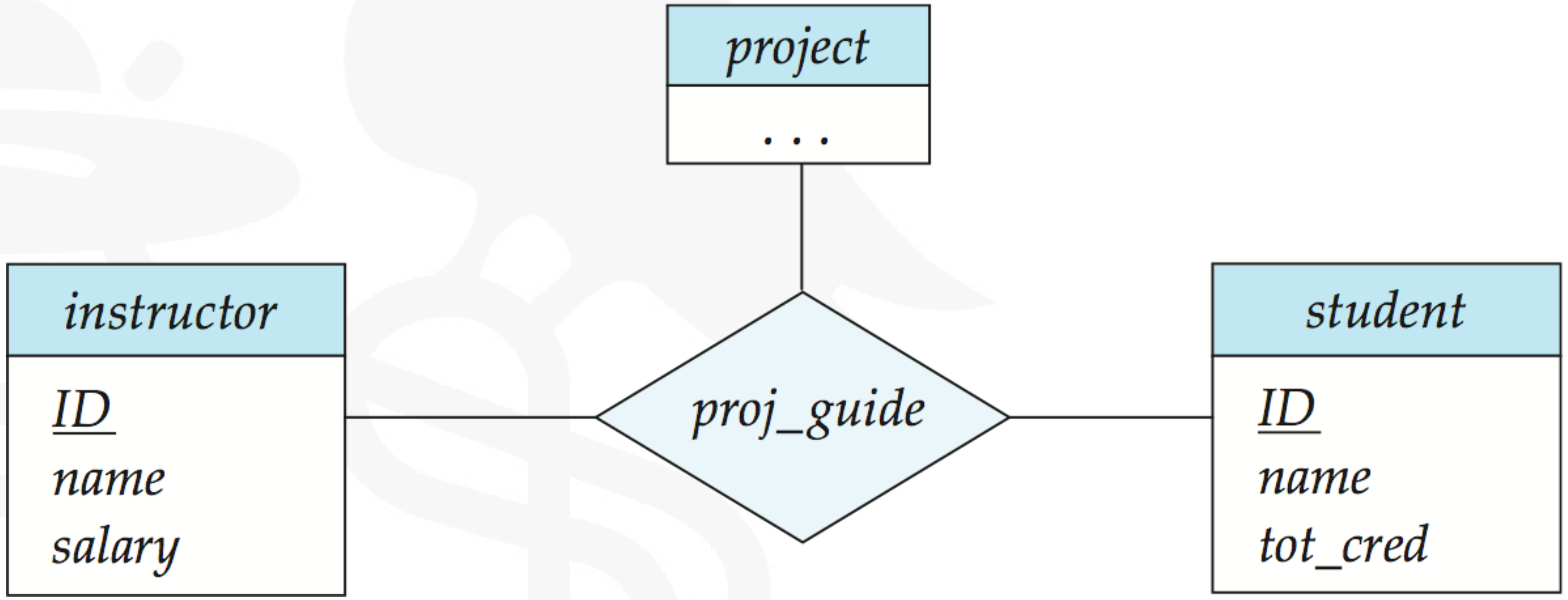
- An **attribute** can also be property of a relationship.
- For instance, the *advisor* relationship between entities *instructor* and *student* may have the attribute *date* which tracks when the student started being associated with the advisor



Degree of a Relationship

- **binary relationships**
  - involve two entities (or degree two).
  - most relationships in a database are binary.
- Relationships between more than two entities (ternary and quaternary) are rare.
  - Example: *students* work on research *projects* under the guidance of an *instructor*.
  - relationship *proj_guide* is a ternary relationship between *instructor, student,* and *project*

E-R Diagram with a Ternary Relationship



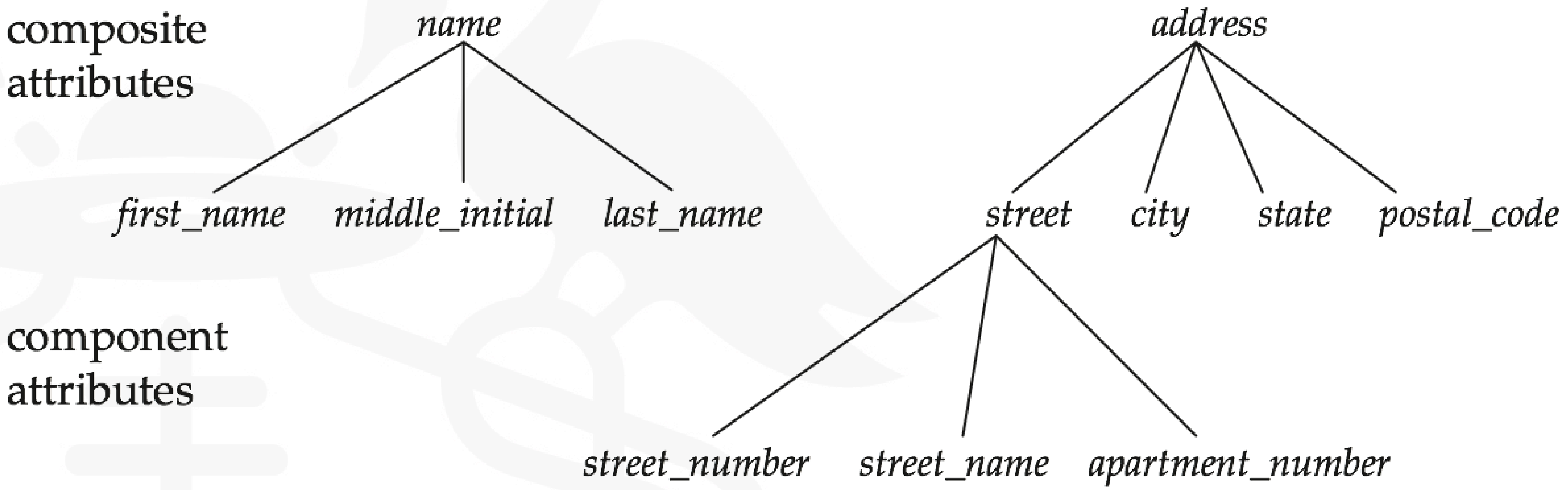
Attributes

- An entity is represented by a set of attributes, that is descriptive properties possessed by all members of an entity.
  - Example:

    instructor = (*ID, name, street, city, salary*)
    course= (*course_id, title, credits*)

- **Domain** – the set of allowed values for each attribute
- 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

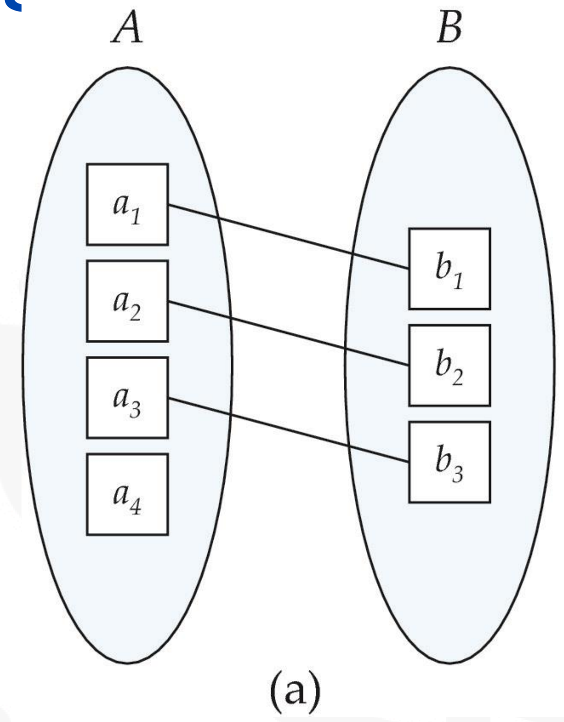
Composite Attributes

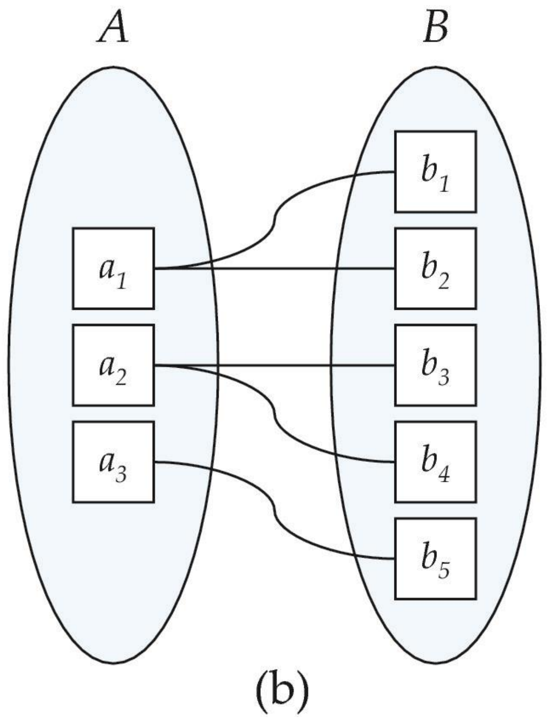


Mapping Cardinality Constraints

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

Mapping Cardinalities



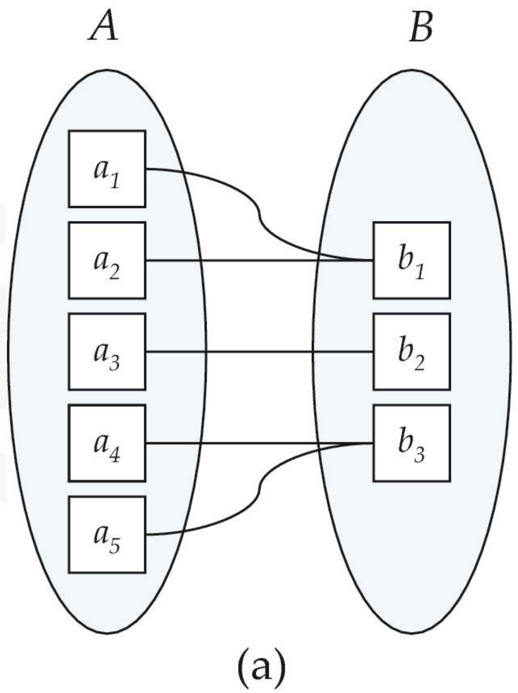


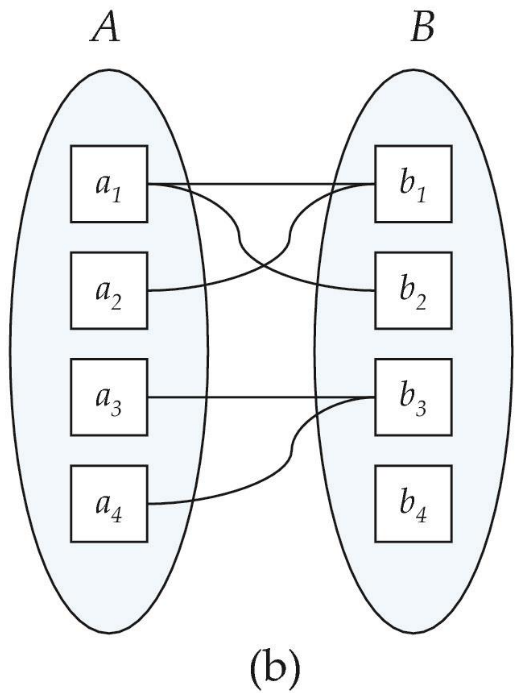
One to one

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

Keys

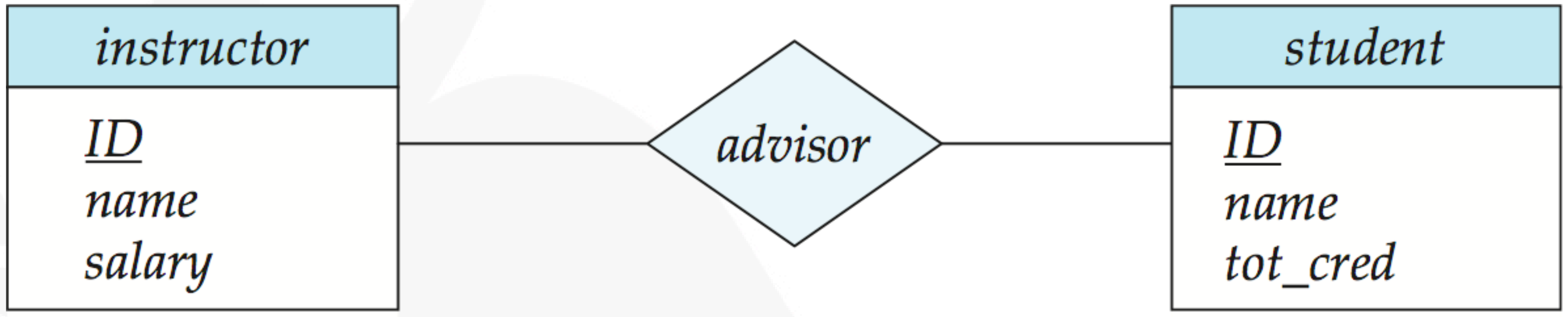
- A **super key** of an entity is a set of one or more attributes whose values uniquely determine each entity.
- A **candidate key** of an entity set is a minimal super key
  - *ID* is candidate key of *instructor*
  - *course_id* is candidate key of *course*
- Although several candidate keys may exist, one of the candidate keys is selected to be the **primary key**.

Redundant Attributes

- Suppose we have entities
  - *instructor*, with attributes including *dept_name*
  - *department*

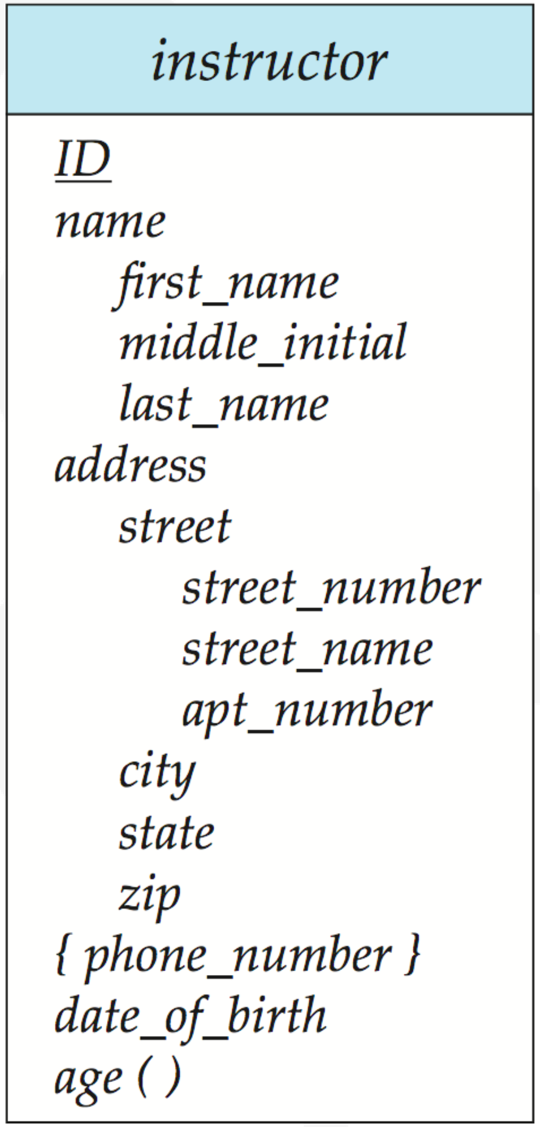
  and a relationship
  - *inst_dept* relating *instructor* and *department*
- Attribute *dept_name* in entity *instructor* is redundant since there is an explicit relationship *inst_dept* which relates instructors to departments
  - The attribute replicates information present in the relationship, and should be removed from *instructor*

E-R Diagrams

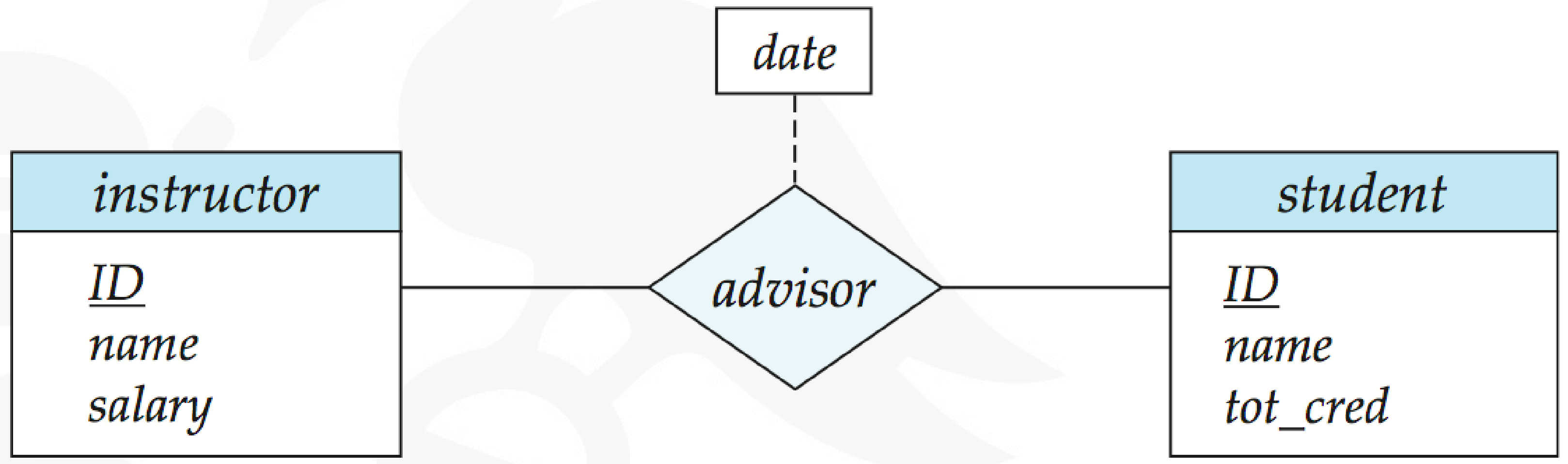


- Rectangles represent entities.
- Diamonds represent relationships.
- Attributes listed inside entity rectangle
- Underline indicates primary key attributes

Entity With Composite, Multivalued, and Derived Attributes

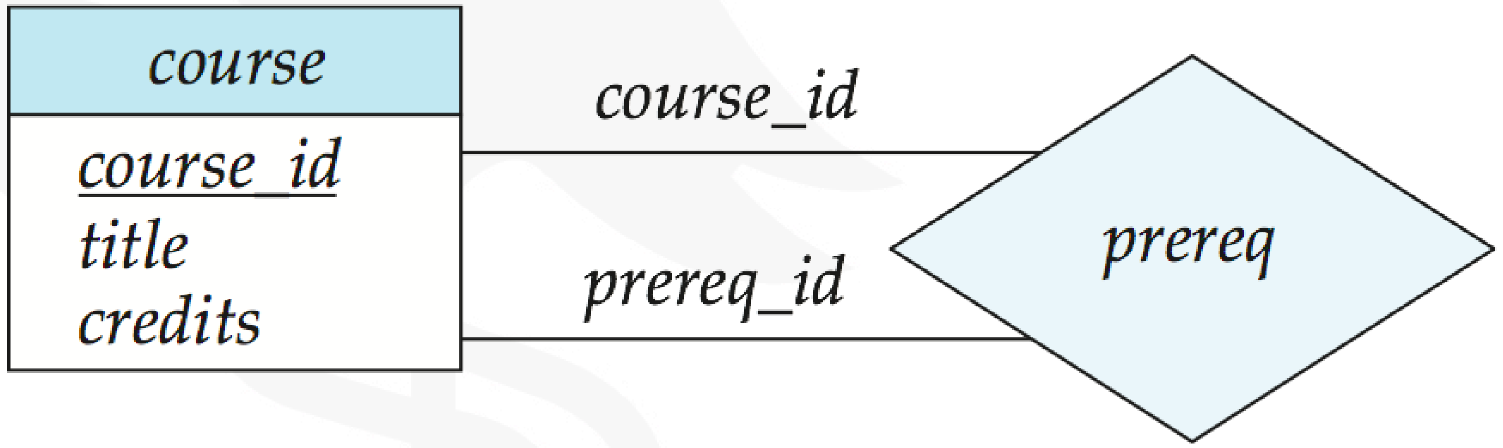


Relationships with Attributes



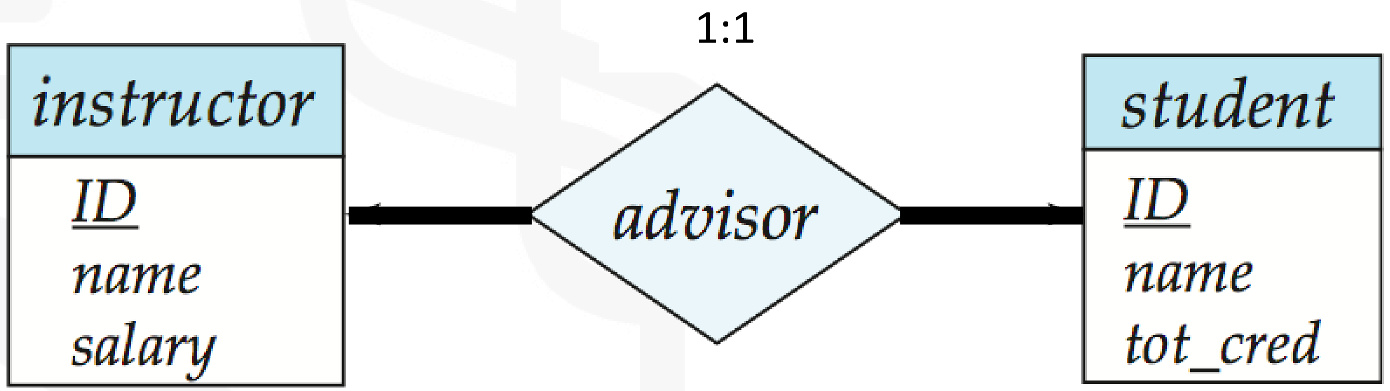
Roles

- In reflexive relationships, each occurrence of an entity plays a "role" in the relationship
- The labels "*course_id*" and "*prereq_id*" are called **roles**.



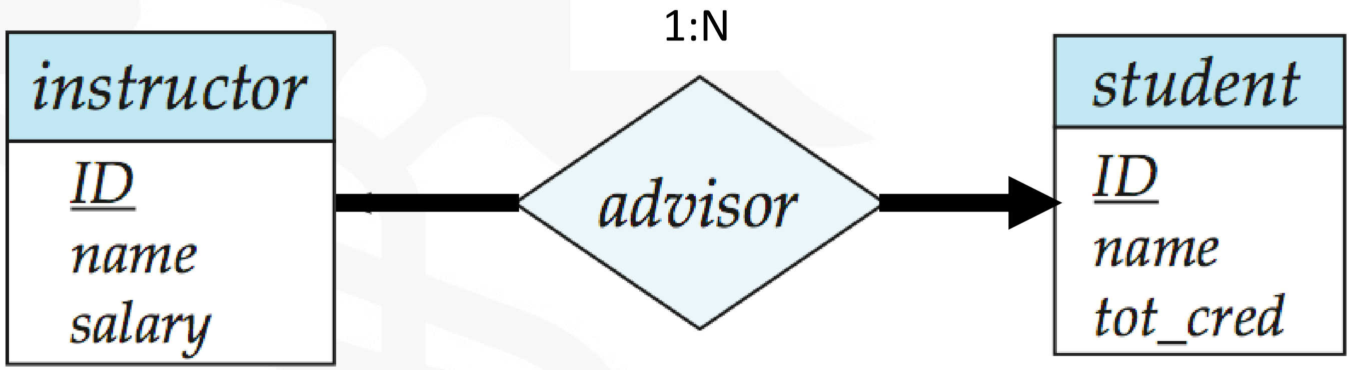
Cardinality Constraints

- We express cardinality constraints by drawing either a directed line (→), signifying "many," or an undirected line (—), signifying "one," between the relationship set and the entity set.
- One-to-one relationship:
  - an instructor is associated with at most one student via *advisor*
  - and a student is associated with at most one instructor via *advisor*



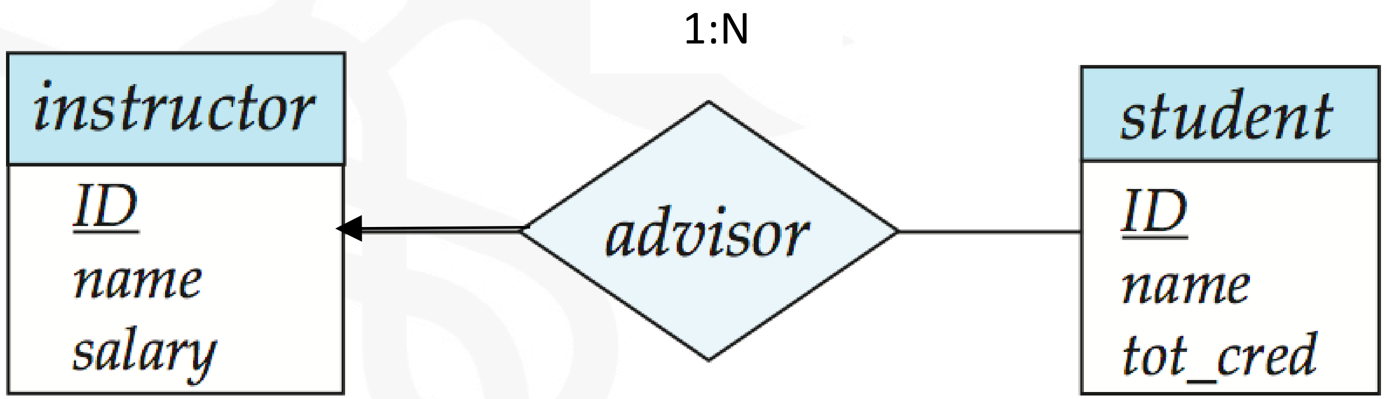
One-to-Many Relationship

- 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



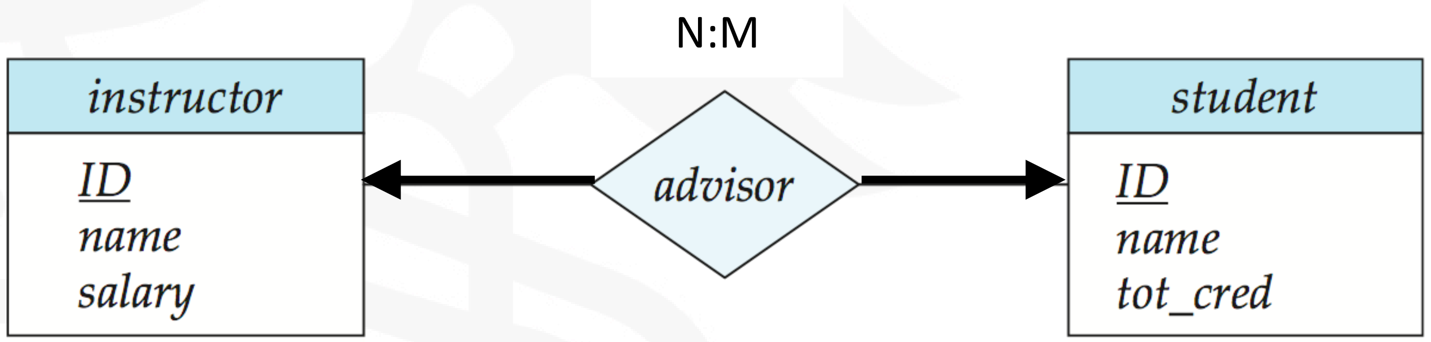
Many-to-One Relationships

- 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*



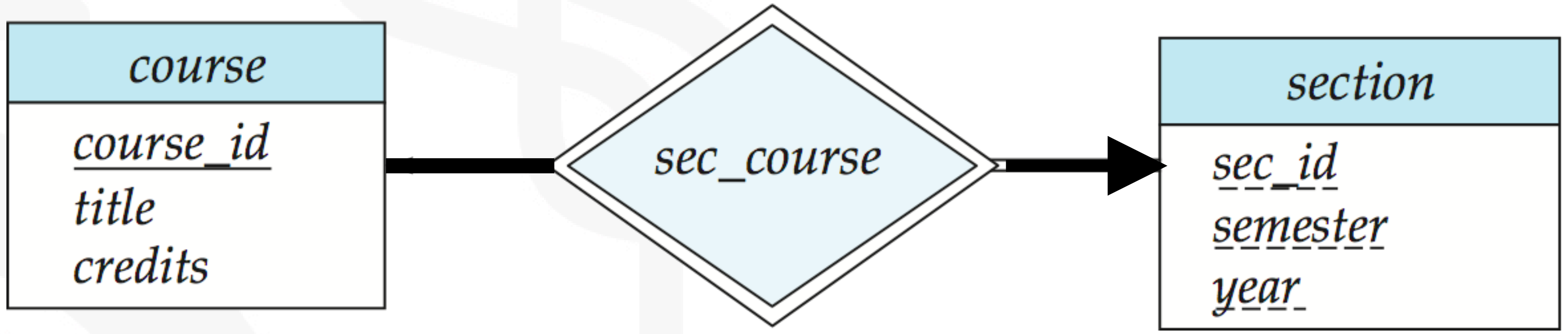
Many-to-Many Relationship

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



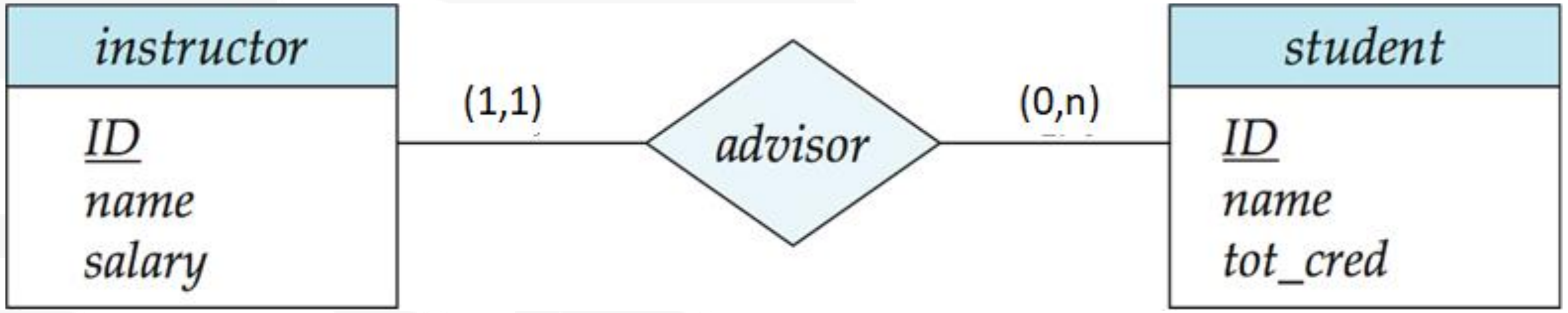
Participation of an Entity Set in a Relationship Set

- Total participation: every entity in the entity set participates in at least one relationship in the relationship set
  - E.g., participation of *section* in *sec_course* is total
    - every *section* must have an associated course
- Partial participation: some entities may not participate in any relationship in the relationship set
  - Example: participation of *instructor* in *advisor* is partial
    - Not every *instructor* must have an associated student



Alternative Notation for Cardinality Limits

- Cardinality limits can also express participation constraints

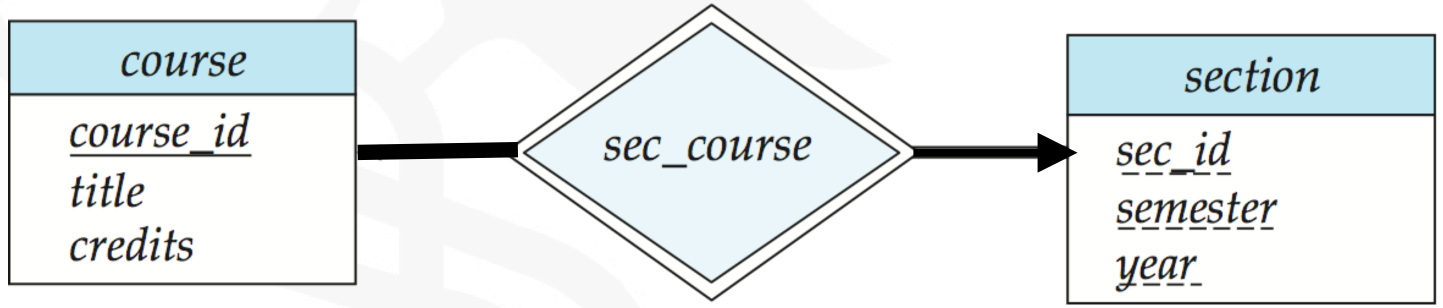


Weak Entities

- An entity that does not have a primary key is referred to as a **weak entity**.
- The existence of a weak entity depends on the existence of a **strong entity**
  - It must relate to the strong entity via a total, one-to-many relationship from the strong to the weak entity
  - **Identifying relationship** depicted using a double diamond
- The **discriminator** (*or partial key)* of a weak entity is the set of attributes that distinguishes among all the entities of a weak entity set.
- The primary key of a weak entity set is formed by the primary key of the strong entity on which the weak entity is existence dependent, plus the discriminator.

Weak Entities (Cont.)

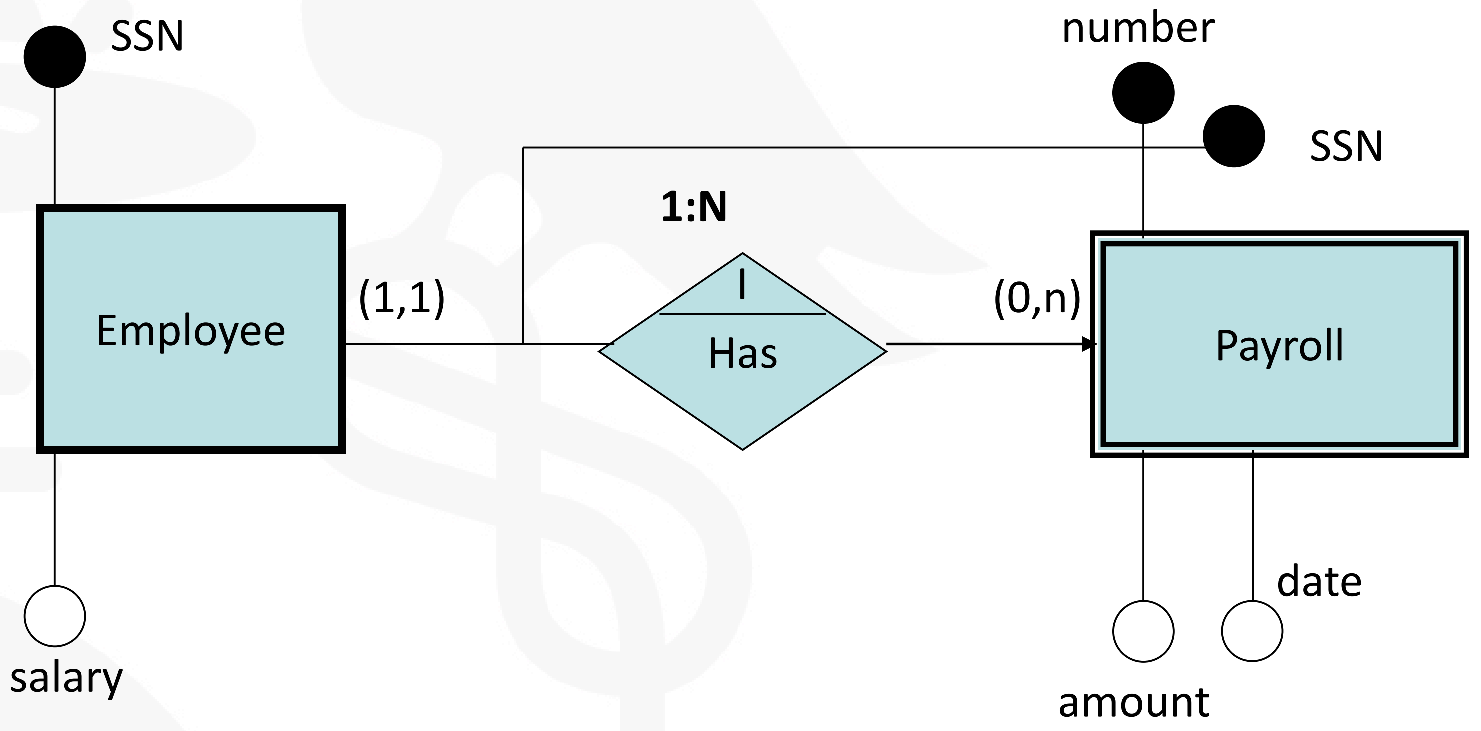
- We underline the discriminator of a weak entity with a dashed line.
- We put the identifying relationship of a weak entity in a double diamond.
- Primary key for *section* – (*course_id, sec_id, semester, year*)



Weak Entities (Cont.)

- **Identifying relationship**

  A company has 10 employees. An employee has payrolls, which are identified by a number that begins at 1 and continues according to the months of seniority in the company. To identify a payroll it is necessary to know the SSN of the employee.



Exercise 1

A company wants to build a database to manage the training of its employees. The semantic restrictions that must be collected are the following:

- The company organizes internal courses and the boss needs to know the code, the name, the description, the number of hours and the cost of the course.
- A course may have as a prerequisite to have previously completed another courses, and, in turn, the completion of one course may be a prerequisite for others. A course that is a prerequisite of another may be mandatory or optional.
- The same course has different editions, that is, it is taught in different places, dates and with different schedules (intensive, morning or afternoon). On the same start date, only one edition of a course can be taught.
- The courses are taught by the company's own staff.
- For each employee, we need to stored their code, name and surname, address, telephone number, NID, date of birth and salary, as well as what courses they are trained to teach.

Exercise 2

- Construct an E-R diagram for a car insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents. Each insurance policy covers one or more cars, and has one or more premium payments associated with it. Each payment is for a particular period of time, and has an associated due date, and the date when the payment was received.

Exercise 3

- Design an E-R diagram for keeping track of the exploits of your favorite sports team. You should store the matches played, the scores in each match, the players in each match, and individual player statistics for each match. Summary statistics should be modeled as derived attributes.