ECE30030/ITP30010 Database Systems

E-R Model

Reading: Chapter 6

Charmgil Hong

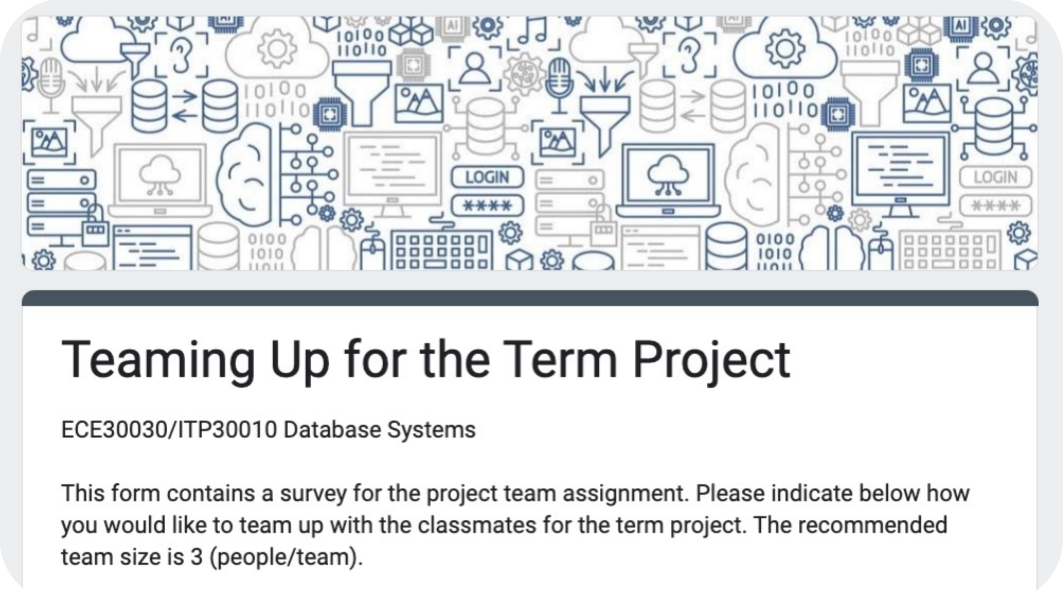
charmgil@handong.edu

Spring, 2025

Handong Global University

Announcements

- Make teams for the term project
  - https://forms.gle/T742G8LQBikzfrUv9 Reponse due: Thursday, April 17
  - Problem & data release: Week #8 (tentative)



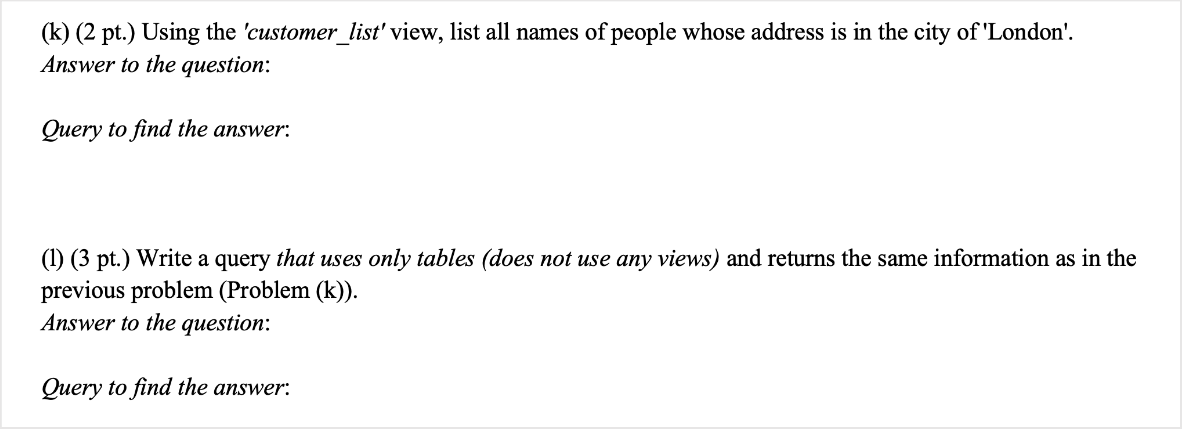
Teaming Up for the Term Project

ECE30030/ITP30010 Database Systems

This form contains a survey for the project team assignment. Please indicate below how you would like to team up with the classmates for the term project. The recommended team size is 3 (people/team).

Announcements

- HW#3 is released
  - Due: Thursday, April 24
  - For Problem 4 (k)-(l), treat *views* as pseudo-tables:
    - (k) Write a query that uses ‘custsomer_list’ as a table.
    - (l) Write an alternative query that does the same as (k) while not using ‘custsomer_list’.



(k) (2 pt.) Using the *'customer_list'* view, list all names of people whose address is in the city of 'London'.
Answer to the question:

Query to find the answer:

(l) (3 pt.) Write a query *that uses only tables (does not use any views)* and returns the same information as in the previous problem (Problem (k)).
Answer to the question:

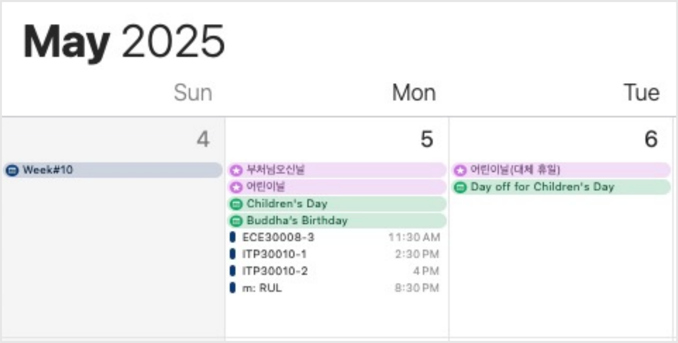
Query to find the answer:

- Heads-up: HW#4 will be released before the midterm exam
  - Check out the problems before the midterm exam

Announcements

- Midterm is scheduled on Thursday, May 1 (Week #9)
  - Coverage: ~ Advanced SQL

- No offline meeting on May 5 (National holiday)
  - Review on the midterm exam is on Thursday, May 8 (Week #10)



May 2025

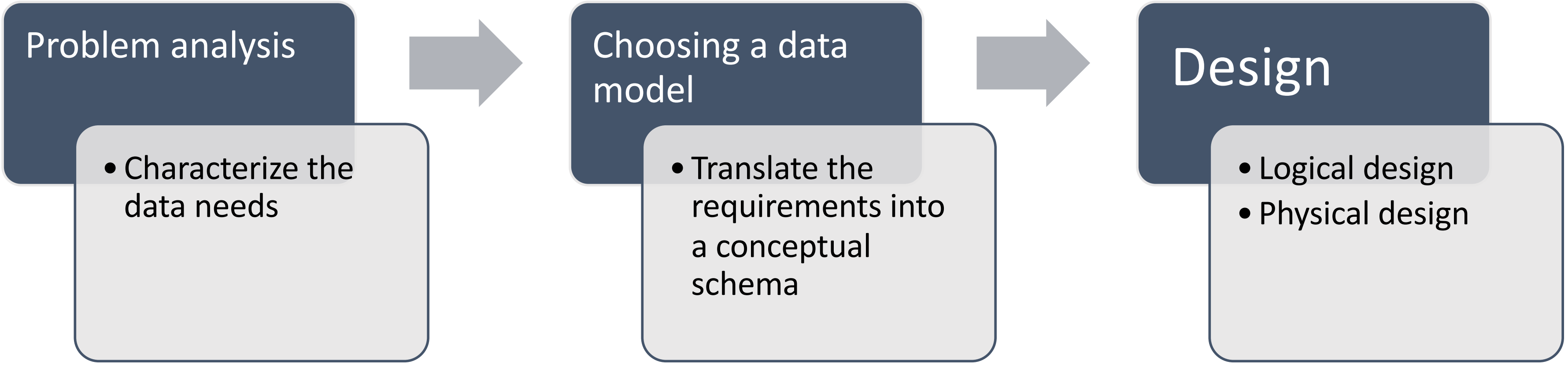
Sun	Mon	Tue
4	5	6
Week#10	부처님오신날 어린이날 Children's Day Buddha's Birthday ECE30008-3 11:30 AM ITP30010-1 2:30 PM ITP30010-2 4 PM m: RUL 8:30 PM	어린이날(대체 휴일) Day off for Children's Day

Agenda

- Designing a database
- E-R diagrams

Design Phases

- Initial phase: characterize fully the data needs of the prospective database users
- Second phase: choose a data model
  - Apply the concepts of the chosen data model
  - Translate the requirements into a conceptual schema of the database
  - A fully developed conceptual schema indicates the functional requirements of the enterprise
    - Describe the kinds of operations (or transactions) that will be performed on the data



Problem analysis

- Characterize the data needs

Choosing a data model

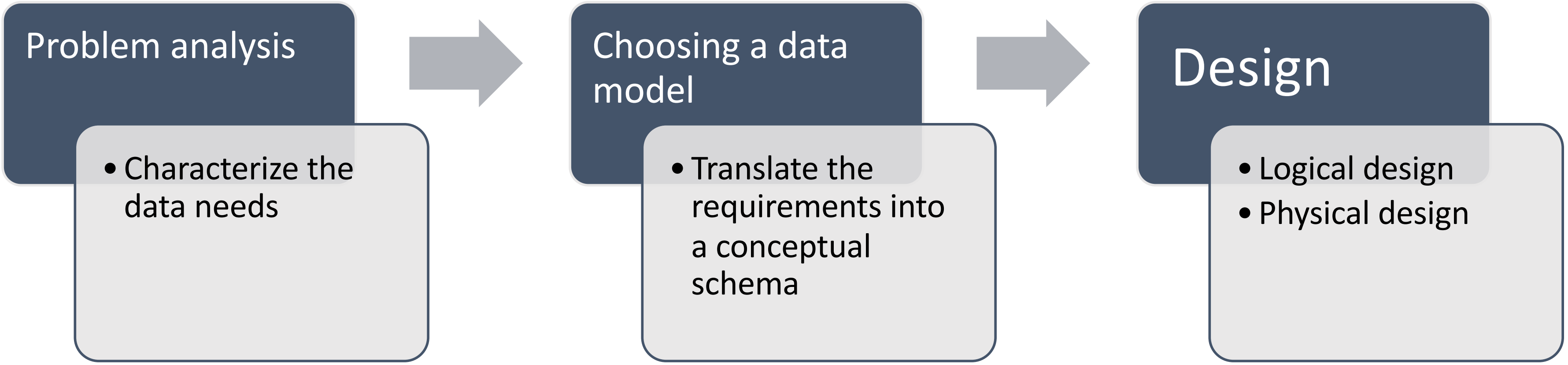
- Translate the requirements into a conceptual schema

Design

- Logical design
- Physical design

Design Phases

- Final Phase: Move from an abstract data model to the implementation of the database
  - Logical Design – Deciding on the database schema
    - Database design requires that 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 should the attributes be distributed among the various relation schemas?*
  - Physical Design – Deciding on the physical layout of the database



Problem analysis

- Characterize the data needs

Choosing a data model

- Translate the requirements into a conceptual schema

Design

- Logical design
- Physical design

Design Phases

- In designing a database schema, we must ensure that we avoid two major pitfalls:
  - Redundancy: a bad design may result in repeated information
    - Redundant representation of information may lead to data inconsistency among the various copies of information
  - Incompleteness: a bad design may make certain aspects of the enterprise difficult or impossible to model

- Avoiding bad designs is not enough. There may be a large number of good designs from which we must choose

Design Approaches

- Entity Relationship Model
  - 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* (E-R diagram)

- Normalization Theory
  - Formalize what designs are bad, and test for them

Agenda

- Designing a database
- **E-R diagrams**
  - Mapping cardinalities
  - Primary keys in E-R models
  - Weak entity sets
  - Reduction to relation schemas

E-R Model for Database Modeling

- The E-R data model was developed to facilitate database design by allowing specification of a database schema
  - Database schema represents the overall logical structure of a database

- The E-R data model employs three basic concepts:
  - Entity sets
  - Relationship sets
  - Attributes

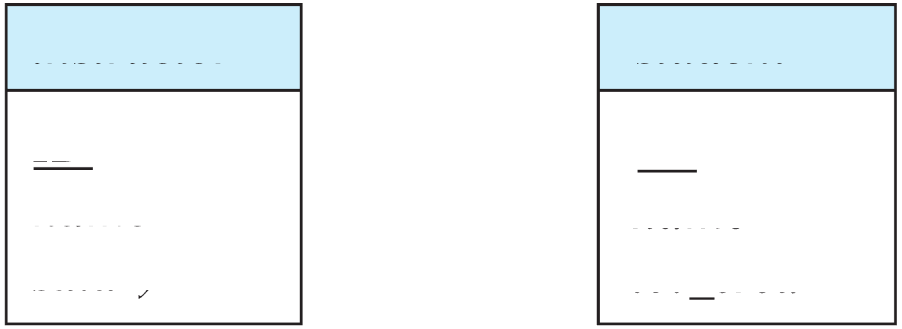
- The E-R model has an associated diagrammatic representation
  - E-R diagram can express the overall logical structure of a database graphically

Entity Sets

- An entity is an object that exists and is distinguishable from other objects
    - *E.g.*, specific person, company, event, plant
- An entity set is a set of entities of the same type that share the same properties
    - *E.g.*, 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
    - *E.g., instructor =* (*ID, name, salary*)
      course= (*course_id, title, credits*)
- A subset of the attributes form a primary key of the entity set; *i.e.*, uniquely identifying each member of the set

Representing Entity Sets in E-R Diagrams

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



Example: Entity and Relationship Sets

- Entity Sets – *instructor* and *student*

76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

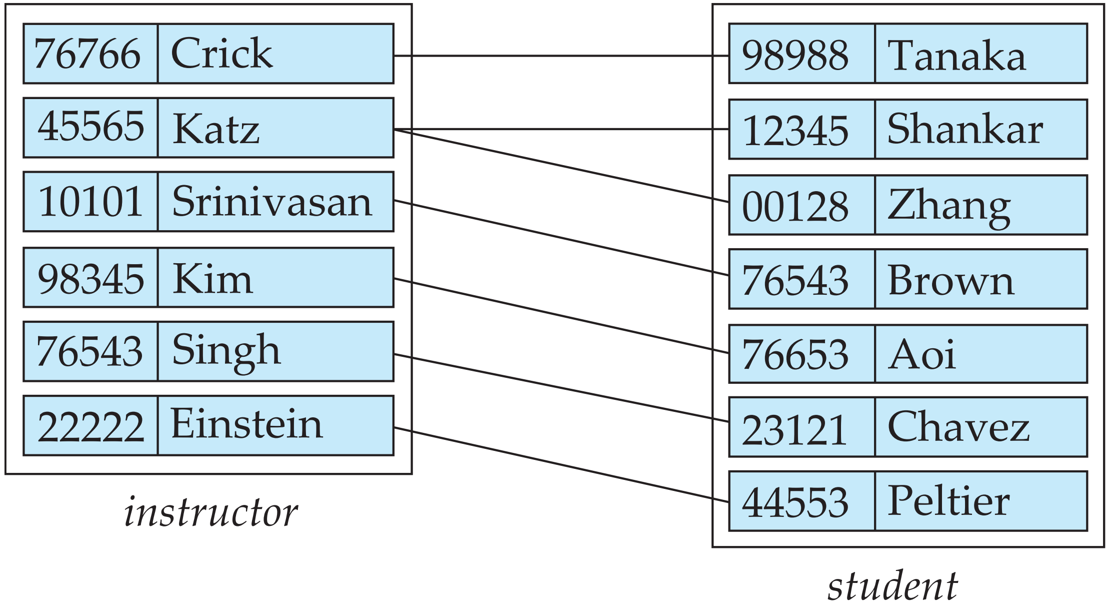
instructor

98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

Example: Entity and Relationship Sets

- Relationship Sets – define the relationship set *advisor* to denote the associations between students and the instructors who act as their advisors



76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

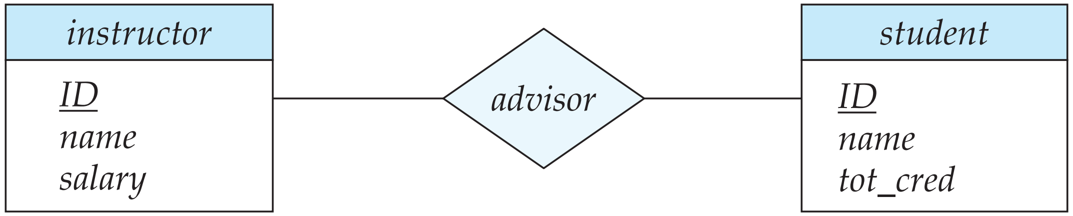
instructor

98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

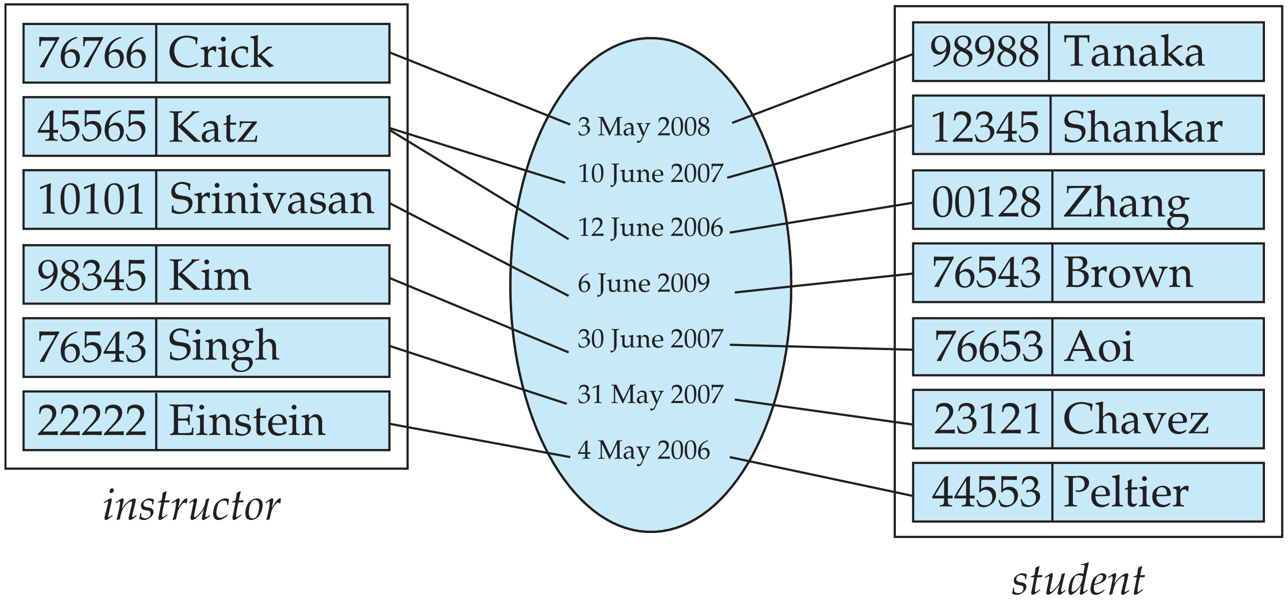
Representing Relationship Sets via E-R Diagrams

- Diamonds represent relationship sets



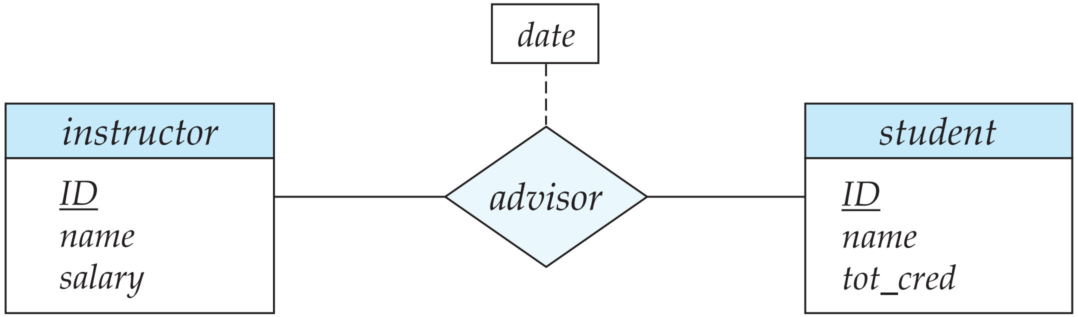
Example: Entity and Relationship Sets

- An attribute can also be associated with a relationship set
    - *E.g.*, the *advisor* relationship set between entity sets *instructor* and *student* may have the attribute *date* which tracks when the student started being associated with the advisor



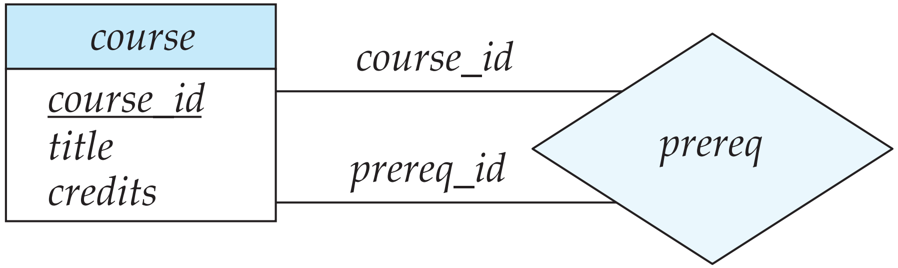
Relationship Sets with Attributes

- An attribute can also be associated with a relationship set



Roles

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



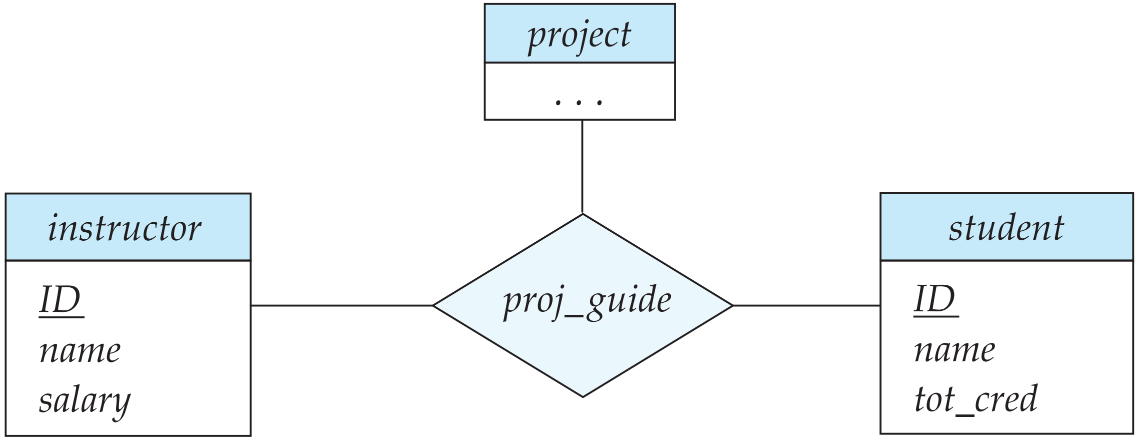
Degree of a Relationship Set

- Binary relationship
  - Involves two entity sets (or degree two)
  - Most relationship sets in a database system are binary

- Relationships between more than two entity sets are rare but possible
  - *E.g., students* work on research *projects* under the guidance of an *instructor*
  - Relationship *proj_guide* is a ternary relationship between *instructor, student,* and *project*

Non-binary Relationship Sets

- Most relationship sets are binary
- There are occasions when it is more convenient to represent relationships as non-binary
- E-R diagram with a ternary relationship:



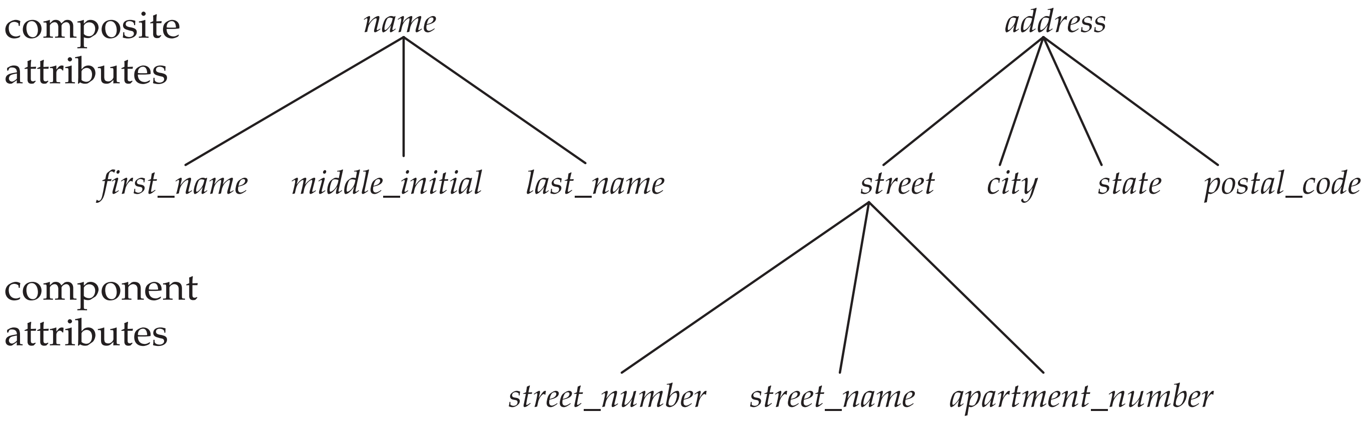
Complex Attributes

- Attribute types:
  - Simple and composite attributes
  - Single-valued and multivalued attributes
    - *E.g.*, multivalued attribute: *phone_numbers* – a person can have more than one phone numbers
  - Derived attributes: attributes that can be computed from other attributes
    - *E.g.*, age, given date_of_birth

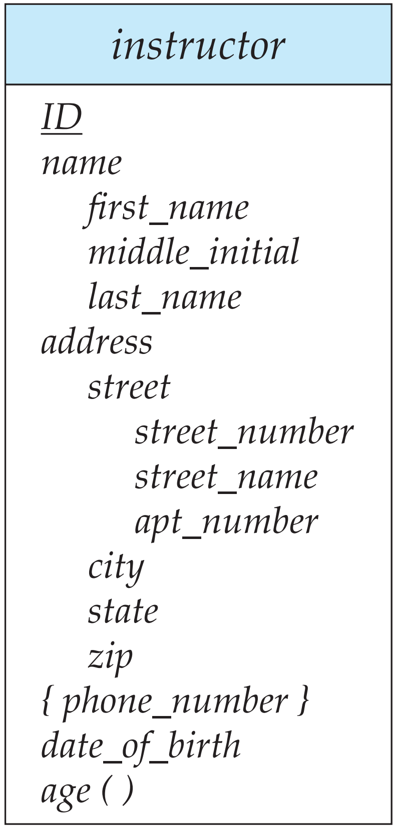
- Domain: the set of permitted values for each attribute

Composite Attributes

- Composite attributes allow us to divided attributes into subparts (other attributes)



Representing Complex Attributes in E-R Diagrams



```
instructor
ID
name
    first_name
    middle_initial
    last_name
address
    street
        street_number
        street_name
        apt_number
    city
    state
    zip
{ phone_number }
date_of_birth
age ( )
```

Agenda

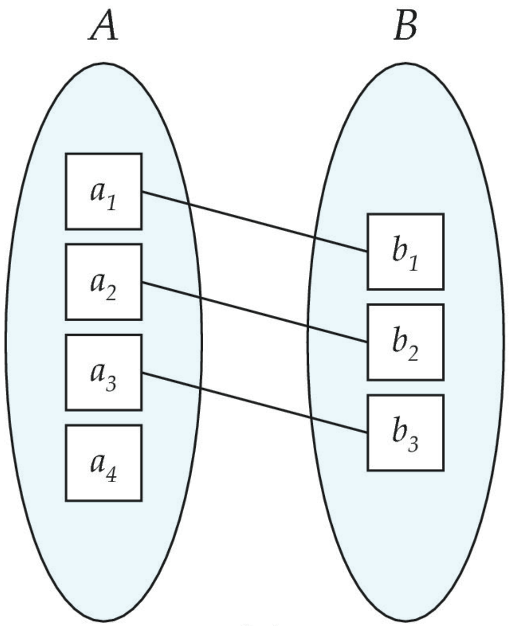
- Designing a database
- E-R diagrams
  - **Mapping cardinalities**
  - Primary keys in E-R models
  - Weak entity sets
  - Reduction to relation schemas

Mapping Cardinalities

- Express the number of entities to which another entity can be associated via a relationship set
  - Most useful in describing binary relationship sets

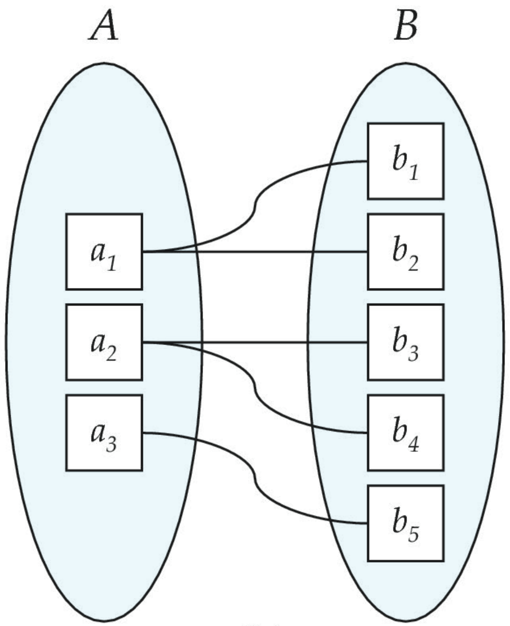
- For a binary relationship set the mapping cardinality must be one of the following types:
  - *One to one*
  - *One to many*
  - *Many to one*
  - *Many to many*

Mapping Cardinalities



(a)

One to one

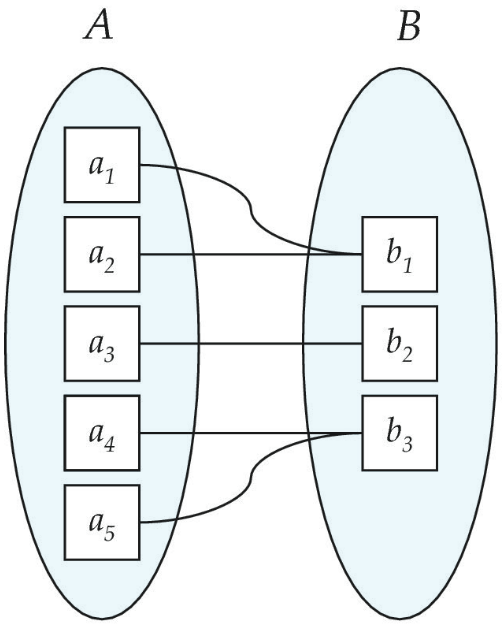


(b)

One to many

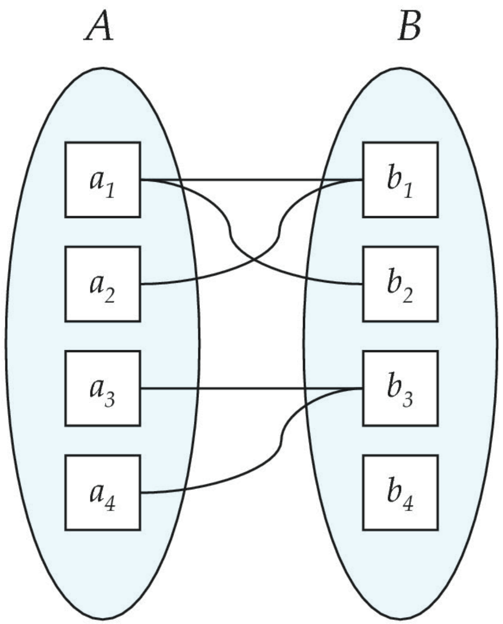
- Note: Some elements in *A* and *B* may not be mapped to any elements in the other set

Mapping Cardinalities



(a)

Many to one



(b)

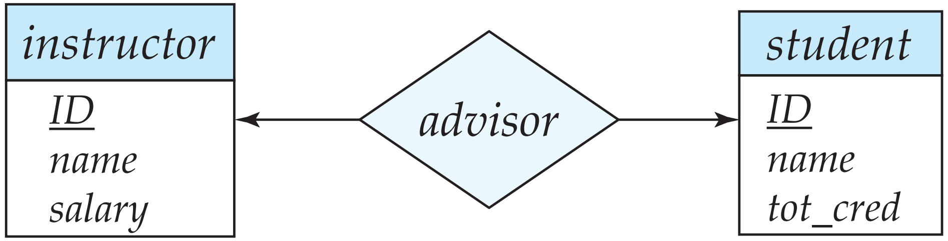
Many to many

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

Representing Cardinalities in E-R Diagrams

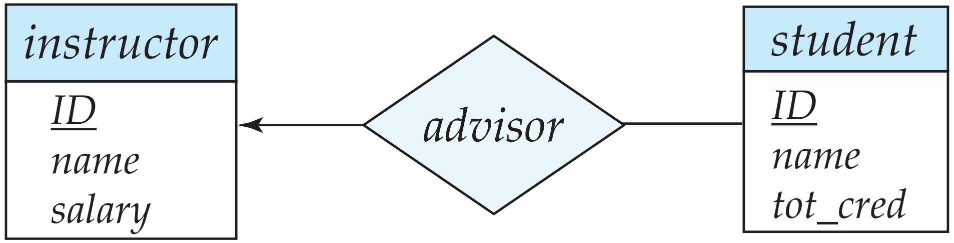
- Express cardinality constraints by drawing either a directed line (→), signifying "one," or an undirected line (—), signifying "many," between the relationship set and the entity set

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



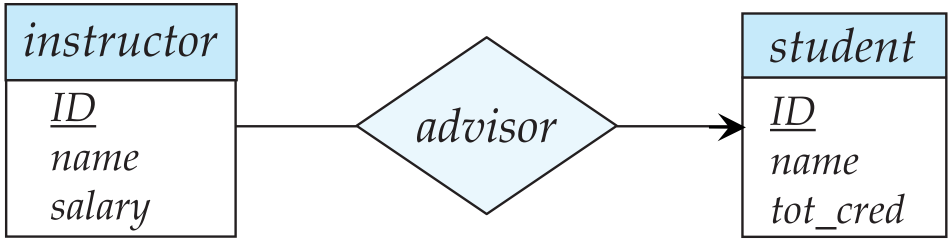
One-to-Many Relationship

- 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*



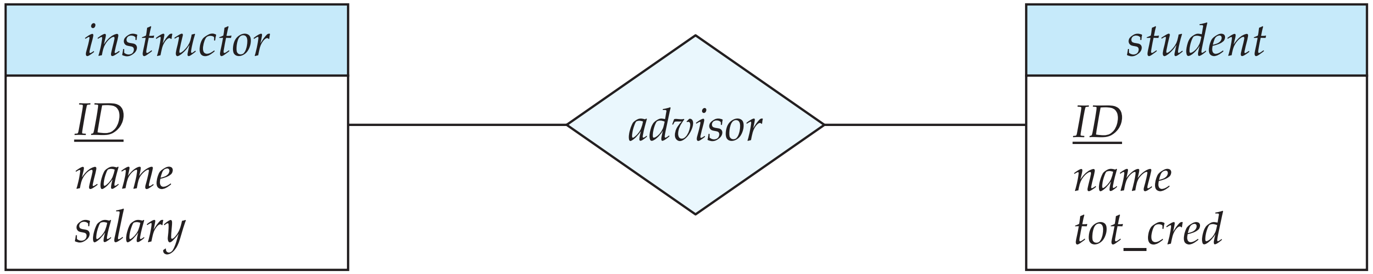
Many-to-One Relationship

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



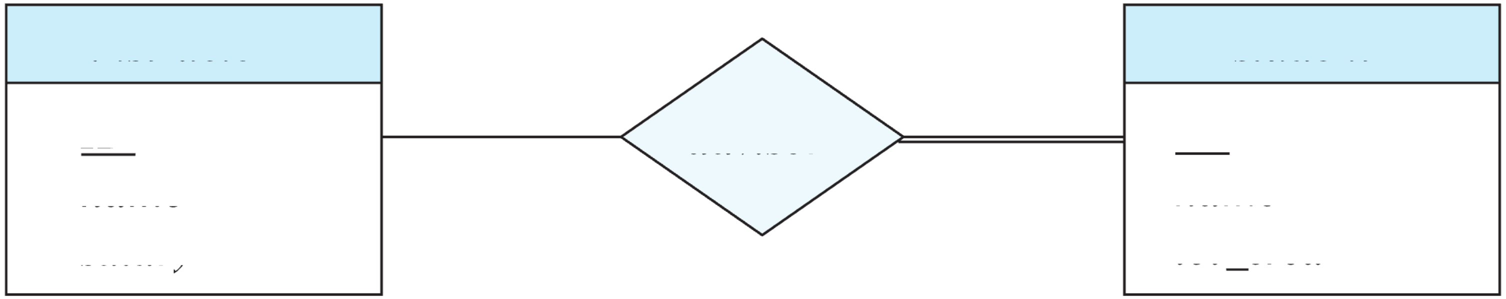
Many-to-Many Relationship

- Many-to-many relationship between an *instructor* and a *student*
  - 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 an entity set participates in at least one relationship in the relationship set



participation of *student* in *advisor r*elation is total

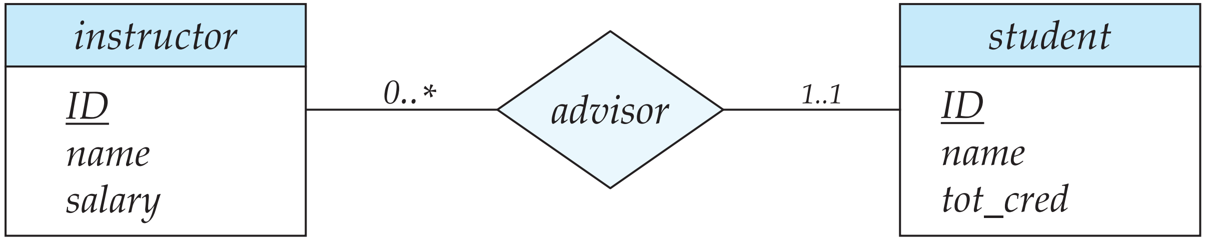
  - *E.g.*, Every *student* must have an associated *instructor*

- Partial participation: some entities may not participate in any relationship in the relationship set
  - *E.g.*, 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 the 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 relationship
  - A maximum value of * indicates no limit

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



Agenda

- Designing a database
- E-R diagrams
    - Mapping cardinalities
    - **Primary keys in E-R models**
    - Weak entity sets
    - Reduction to relation schemas

Primary Key

- Primary keys provide a way to specify how entities and relationships are distinguished
- We consider:
  - Entity sets
  - Relationship sets
  - Weak entity sets

Primary Key for Entity Sets

- By definition, individual entities are distinct
- From database perspective, the *differences among entities must be expressed in terms of their attributes*
  - The attribute values of an entity must be such that they can uniquely identify the entity
  - No two entities in an entity set are allowed to have exactly the same value for all attributes
- A key for an entity is a set of attributes that suffice to distinguish entities from each other

Primary Key for Relationship Sets

- To distinguish among the various relationships of a relationship set, use the individual primary keys of the entities in the relationship set
  - Let R be a relationship set involving entity sets E_1, E_2, …, E_n
  - The primary key for R is consists of the union of the primary keys of entity sets E_1, E_2, …, E_n
  - If the relationship set R has attributes a_1, a_2, ..., a_m associated with it, then the primary key of R also includes the attributes a_1, a_2, ..., a_m

- Example: relationship set "*advisor*"
  - The primary key consists of *inrsructor.ID* and *student.ID*

Choice of Primary Key for Binary Relationship

- The choice of the primary key for a relationship set depends on the mapping cardinality of the relationship set
  - Many-to-Many relationships: The preceding union of the primary keys is a minimal super key and is chosen as the primary key
  - One-to-Many relationships: The primary key of the "Many" side is a minimal super key and is used as the primary key
    - Many-to-one relationships: The primary key of the "Many" side is a minimal super key and is used as the primary key
  - One-to-one relationships: The primary key of either one of the participating entity sets forms a minimal super key, and either one can be chosen as the primary key

Agenda

- Designing a database
- E-R diagrams
  - Mapping cardinalities
  - Primary keys in E-R models
  - **Weak entity sets**
  - Reduction to relation schemas

Weak Entity Sets

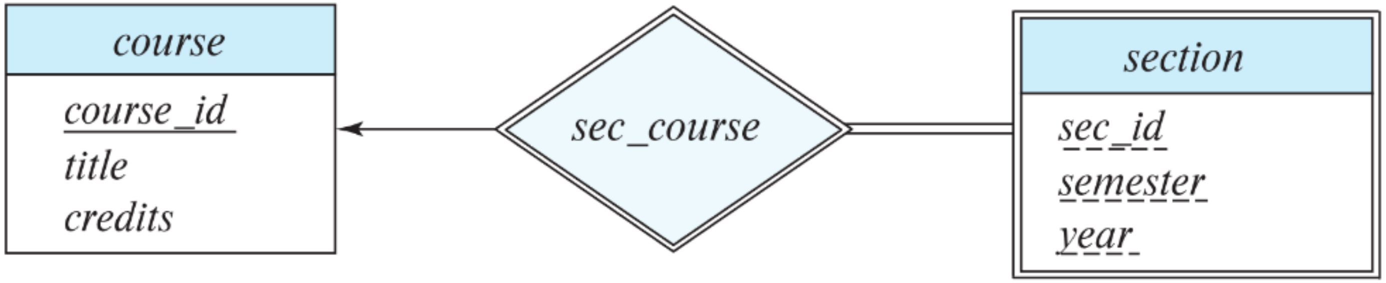
- A weak entity set is one whose existence is dependent on another entity, called its identifying entity
- Instead of associating a primary key with a weak entity, use the identifying entity, along with extra attributes called discriminator to uniquely identify a weak entity
  - A weak entity set does not have a primary key
  - We still need a *means of distinguishing* among an entity set
    - Discriminator of a weak entity: a set of attributes allowing such distinction
    - Primary key of a weak entity set

      = primary key of a strong entity set (which its existence depends)

      + its discriminator

Weak Entity Sets

- A weak entity set is one whose existence is dependent on another entity, called its identifying entity
- Instead of associating a primary key with a weak entity, use the identifying entity, along with extra attributes called discriminator to uniquely identify a weak entity
  - *E.g.*, Consider a *section* entity, which is uniquely identified by a *course_id*, *semester, year*, and *sec_id* → Section entities are related to course entities
    - Treat the relationship *sec_course* as a special relationship that provides extra information
    - In this case, the *course_id*, required to identify *section* entities uniquely



Weak Entity Sets

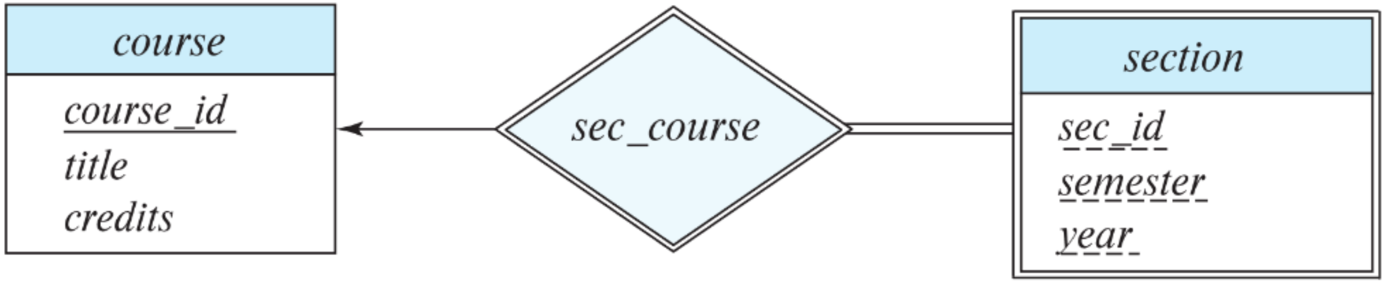
- Identifying entity
  - Every weak entity must be associated with an identifying entity;
    - That is, the weak entity set is said to be existence dependent on the identifying entity set

- The identifying entity set is said to own the weak entity set that it identifies
  - Identifying entity set: an entity set that has a primary key
  - Identifying entity set = strong entity set

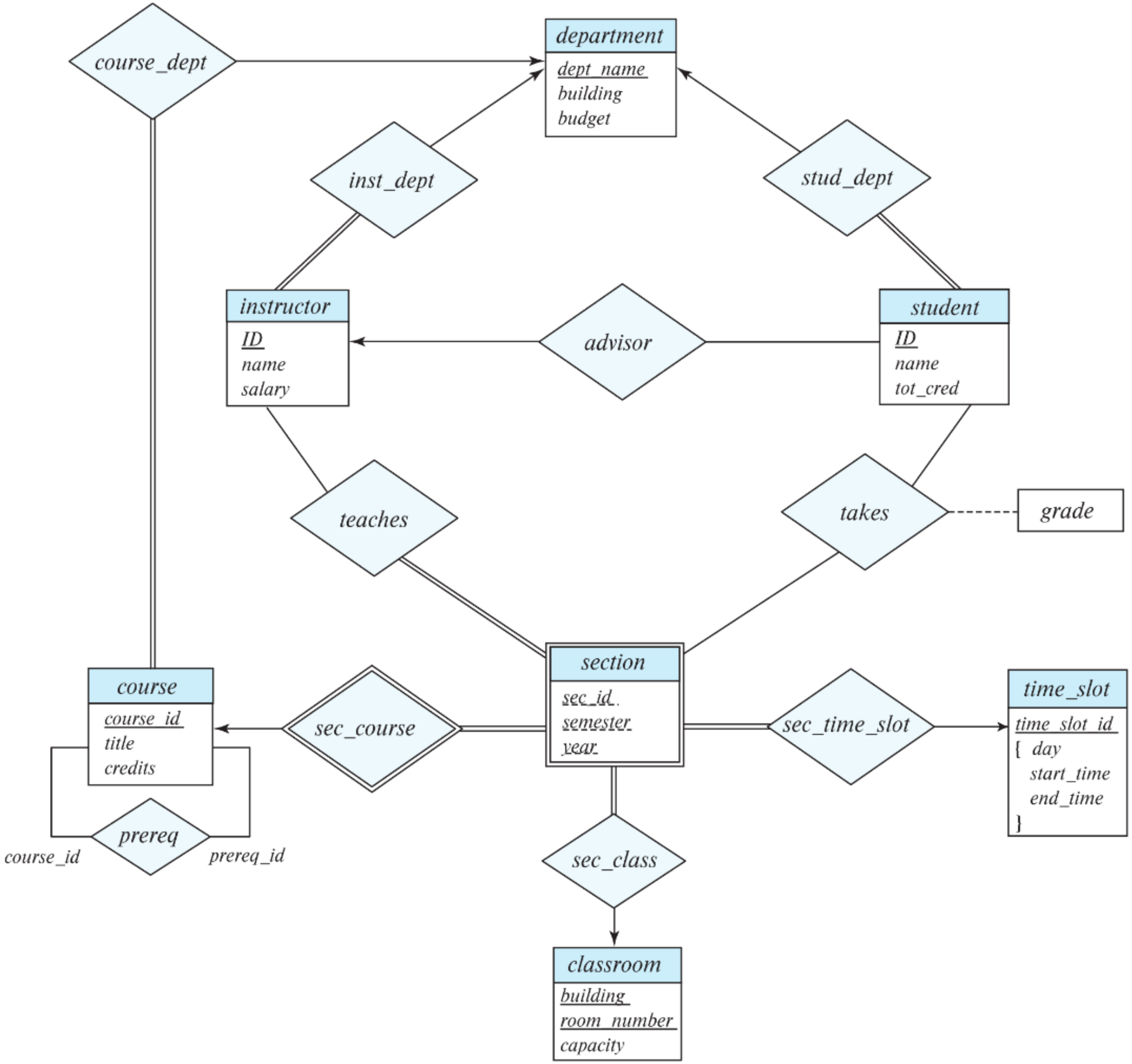
- Identifying relationship
  - Identifying relationship: The relationship associating the weak entity set with the identifying entity set

Expressing Weak Entity Sets

- A weak entity set is depicted via a double rectangle
- Underline the discriminator of a weak entity set with a dashed line
- The relationship set connecting the weak entity set to the identifying strong entity set is depicted by a double diamond
  - *E.g.*, Primary key for section – (*course_id, sec_id, semester, year*)



E-R Diagram for a *University* Database



Agenda

- Designing a database
- E-R diagrams
  - Mapping cardinalities
  - Primary keys in E-R models
  - Weak entity sets
  - **Reduction to relation schemas**

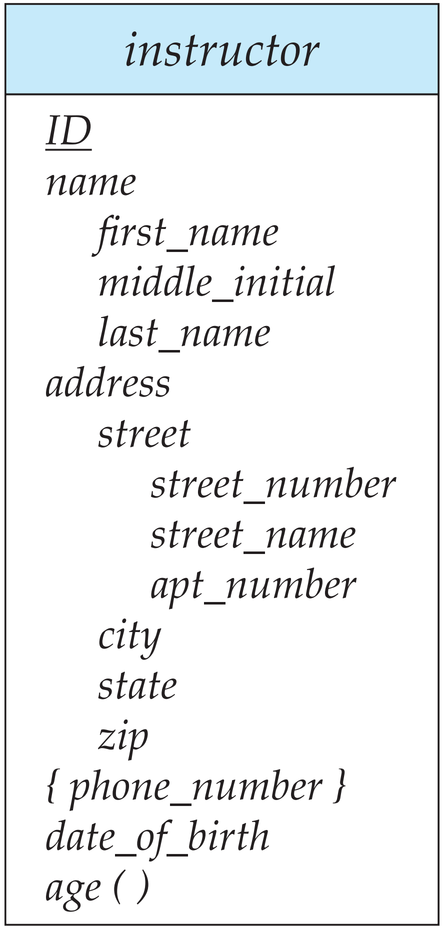
Reduction to Relation Schemas

- Entity sets and relationship sets can be expressed uniformly as relation 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
  - *E.g., 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
  - *E.g., section (course_id, sec_id, sem, year)*

Representation of Entity Sets with Composite Attributes



instructor

ID
name
first_name
middle_initial
last_name
address
street
street_number
street_name
apt_number
city
state
zip
{ phone_number }
date_of_birth
age ( )

- Composite attributes are flattened out by creating a separate attribute for each component attribute
  - *E.g., first_name* → *name_first_name*
    last_name → *name_last_name*
  - Prefixes can be omitted if there is no ambiguity
  - *E.g.*, Ignoring multivalued attributes (*phone_number*), a corresponding instructor schema is:
    - *instructor(ID,*
      first_name, middle_initial, last_name,
      street_number, street_name, apt_number,
      city, state, zip_code,
      date_of_birth)

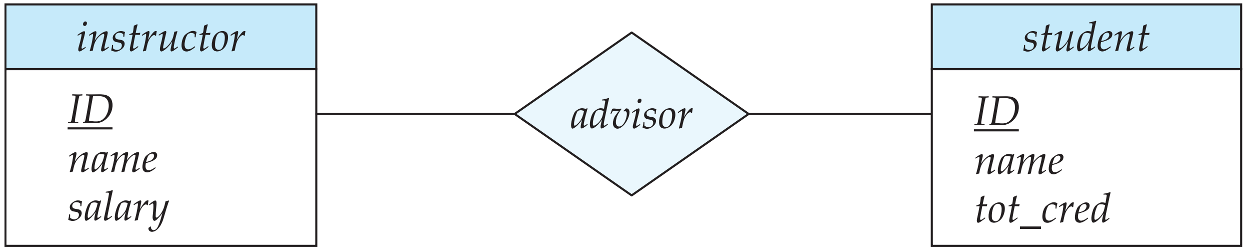
Representation of Entity Sets with Multivalued Attributes

- A multivalued attribute *M* of an entity *E* is represented by a separate schema *EM*
  - Schema *EM* has attributes corresponding to the primary key of *E* and an attribute corresponding to multivalued attribute *M*
  - *E.g.*, Multivalued attribute *phone_number* of *instructor*:
    - *inst_phone*(<u>*ID*</u>, <u>*phone_number*</u>)

- Each value of the multivalued attribute maps to a separate tuple of the relation on schema *EM*
  - *E.g.*, an *instructor* entity with primary key 22222
    and phone numbers 456-7890 and 123-4567
    → maps to two tuples

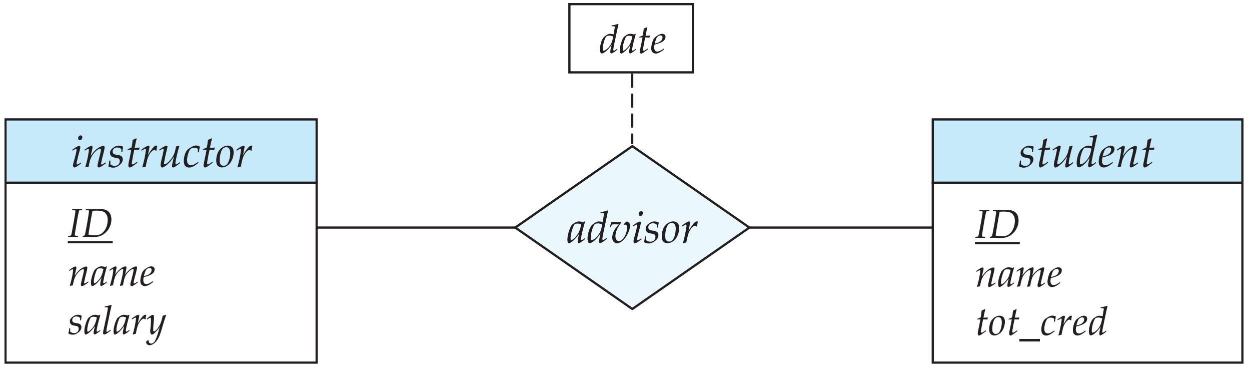
Representing Relationship Sets

- Any relationship set of strong entity sets can be represented as a schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set
  - *E.g.*, schema for relationship set *advisor*
    - *advisor* = (*s_id*, *i_id*)



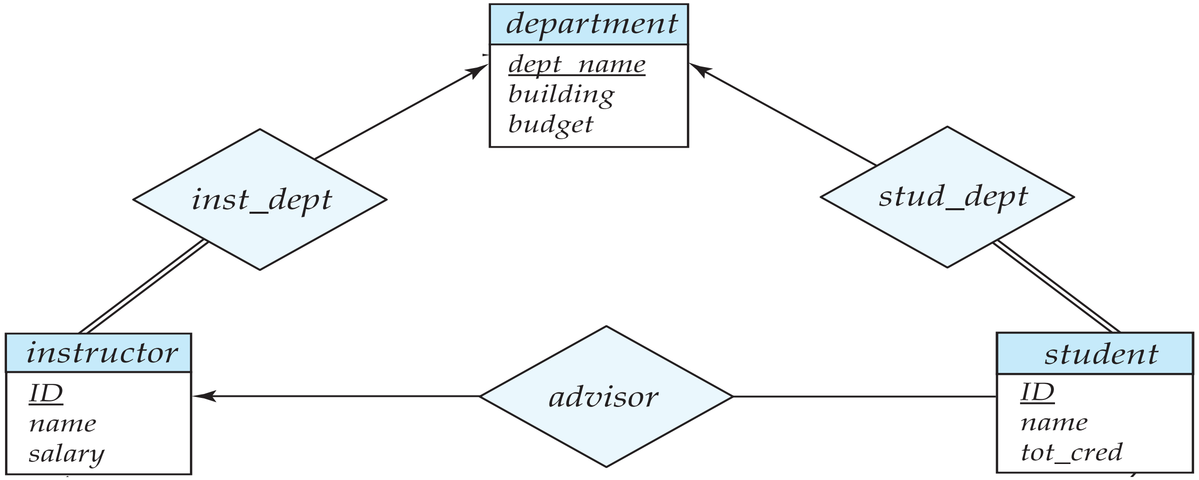
Representing Relationship Sets

- Any relationship set of strong entity sets can be represented as a schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set
  - *E.g.*, schema for relationship set *advisor*
    - *advisor* = (*s_id*, *i_id*)



Redundancy of Schemas

- Such "mapping tables" may be redundant
  - Many-to-one and one-to-many relationship sets that are total on the many-side
  - Can be represented by adding an extra attribute to the "many" side, containing the primary key of the "one" side
    - *E.g.*, Instead of creating a schema for relationship set *inst_dept*, add an attribute *dept_name* to the schema arising from entity set *instructor*



Redundancy of Schemas

- Such “mapping tables” may be redundant
  - Many-to-one and one-to-many relationship sets that are total on the many-side
  - Can be represented by adding an extra attribute to the “many” side, containing the primary key of the “one” side
    - *E.g.*, Instead of creating a schema for relationship set *inst_dept*, add an attribute *dept_name* to the schema arising from entity set *instructor*
  - When participation is partial on the “many” side, replacing a schema by an extra attribute in the schema corresponding to the “many” side could result in null values

Redundancy of Schemas

- Such “mapping tables” may be redundant
  - For one-to-one relationship sets, either side can be chosen to act as the “many” side
    - An extra attribute can be added to either of the tables corresponding to the two entity sets