Database System
(SW5)

5. ER Model (Part 1)

Tiantian Liu
Department of Computer Science
Aalborg University

Fall 2025

Motivation

- ER diagrams are widely used
- ER model is easy to learn--Much simpler than UML
- An ER diagram is a good communication tool--Talking the same language

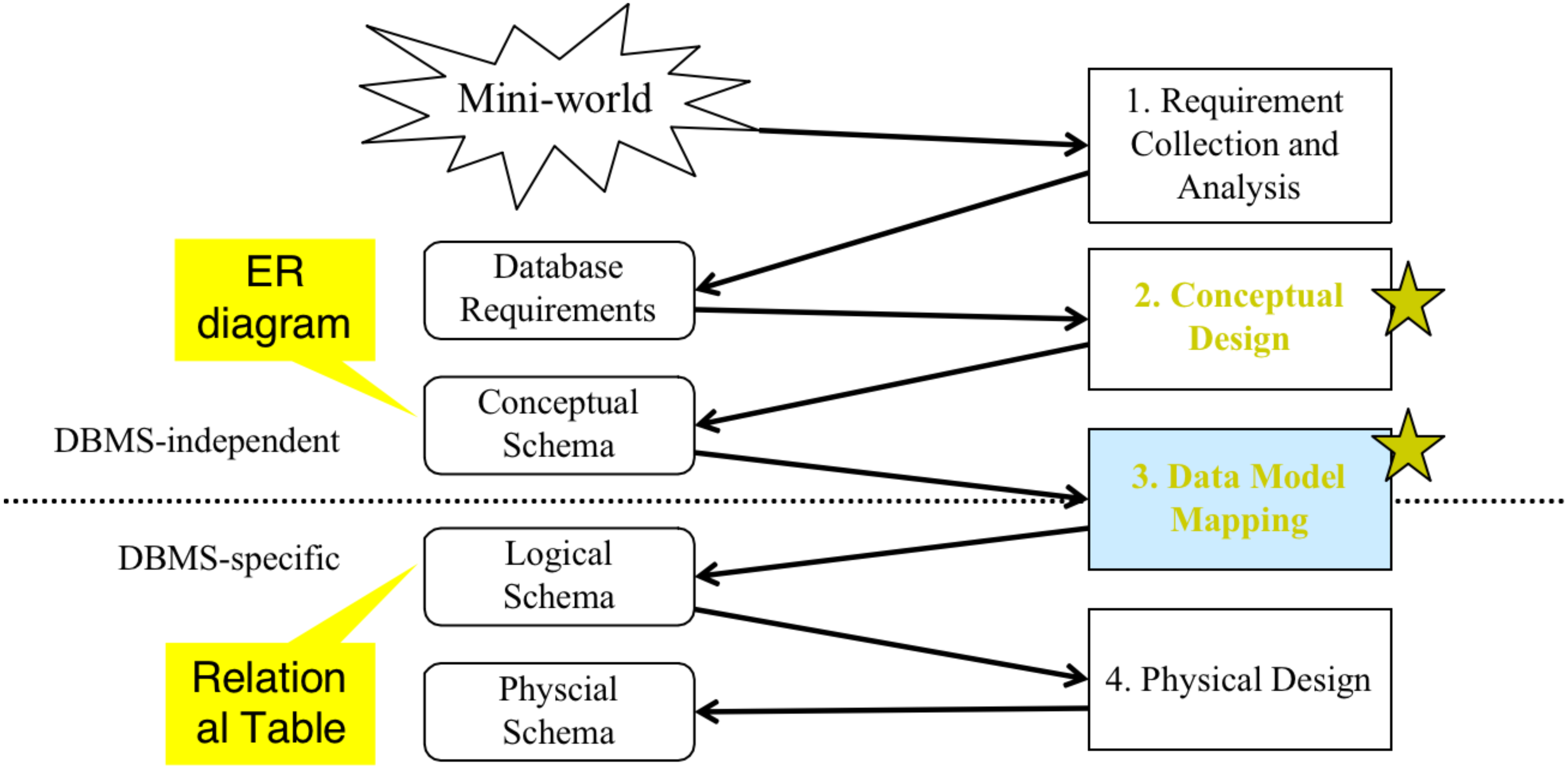
Learning Goals

- Create non-trivial ER diagrams
- Analyze if an ER diagram is good or bad
- Create and explain the mapping of ER diagrams to relations
- Use a particular ER notation properly

Agenda

- Data Modelling Process
- ER Model
- Mapping Cardinality Constraints
- Keys
- Weak Entity Types
- Specialization

Data Modelling Process



Mini-world

1. Requirement Collection and Analysis

Database Requirements

2. Conceptual Design

Conceptual Schema

3. Data Model Mapping

Logical Schema

4. Physical Design

Physcial Schema

DBMS-independent

DBMS-specific

ER diagram

Relation al Table

Step 1: Requirement Collection and Analysis

- For a university database
  - “The instructor ID unambiguously identifies an instructor”
  - “Instructors are in different departments”
  - “Instructors teach courses”
  - “Students take courses”

Step 1: Requirement Collection and Analysis

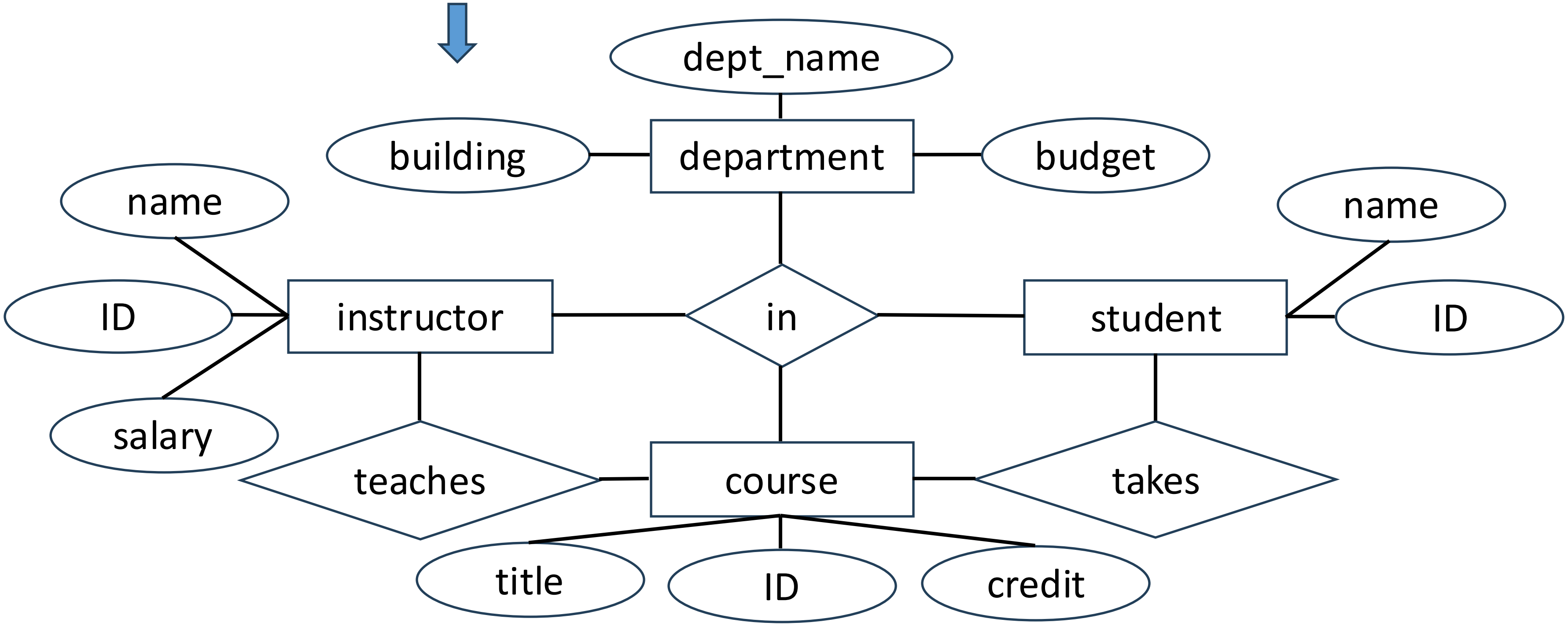
- Object specification
- Instructor
  - ID: char, not null, unique
  - name: varchar
  - dept_name: varchar
  - salary: numeric(8, 2)
- Course
  - course_id: char, not null, unique
  - title: varchar
  - dept_name: varchar
  - credit: numeric(2, 0)

Step 1: Requirement Collection and Analysis

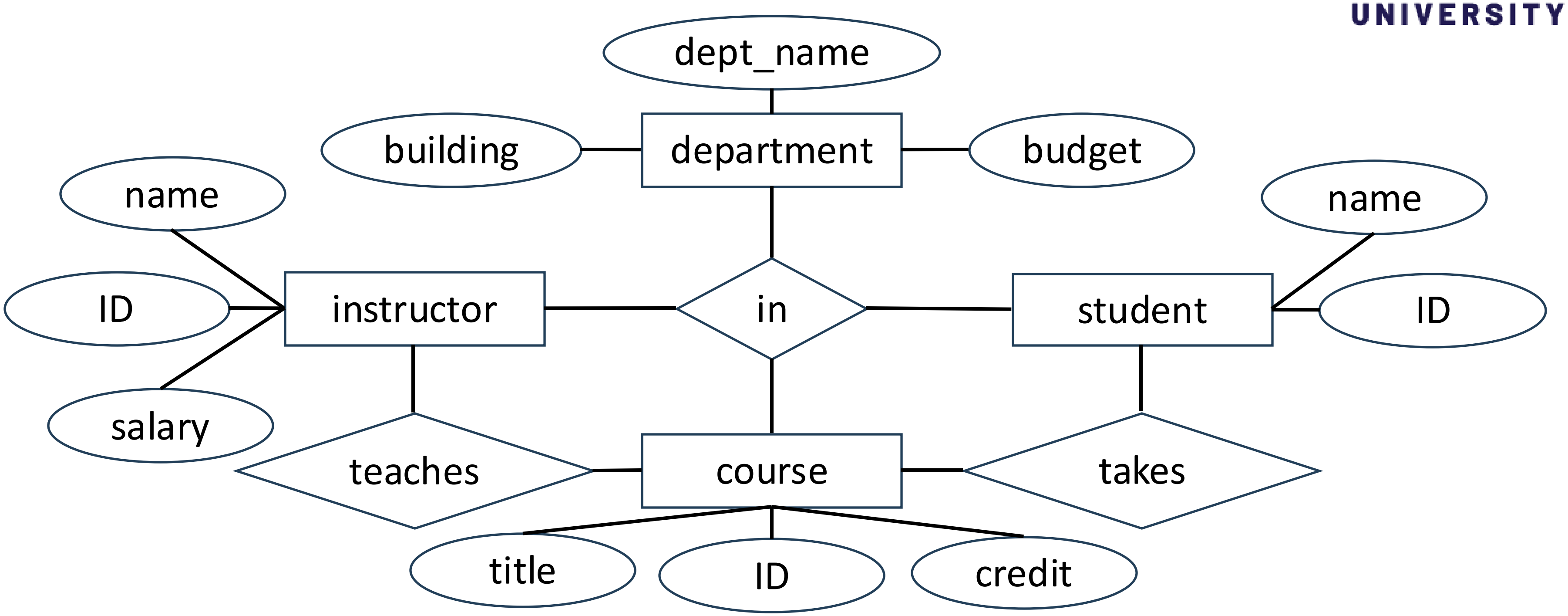
- Relationship specification
- Teaches
  - Participate objects: instructor, course
  - Attributes: semester, year…
- Takes
  - Participate objects: student, course
  - Attributes: semester, year, grades…

Step 2: Conceptual Design

- Requirements
  - “The instructor ID unambiguously identifies an instructor”
  - “Instructors are in different departments”
  - “Instructors teach courses”
  - “Students take courses”



Step 3: Data Model Mapping

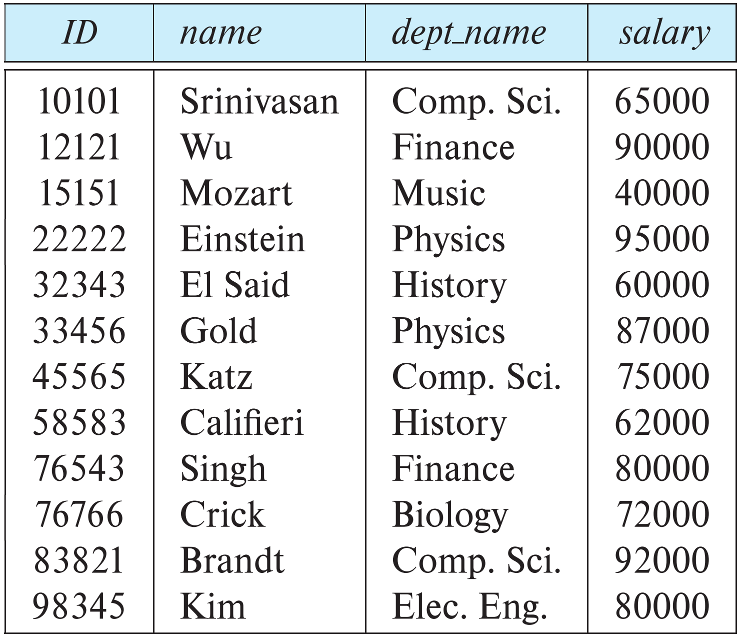


- *instructor* (*ID, name, dept_name, salary*)
- *student* (*ID, name, dept_name, tot_cred*)
- *course* (*course_id, title, dept_name, credits*)
- *department* (*dept_name, building, budget*)

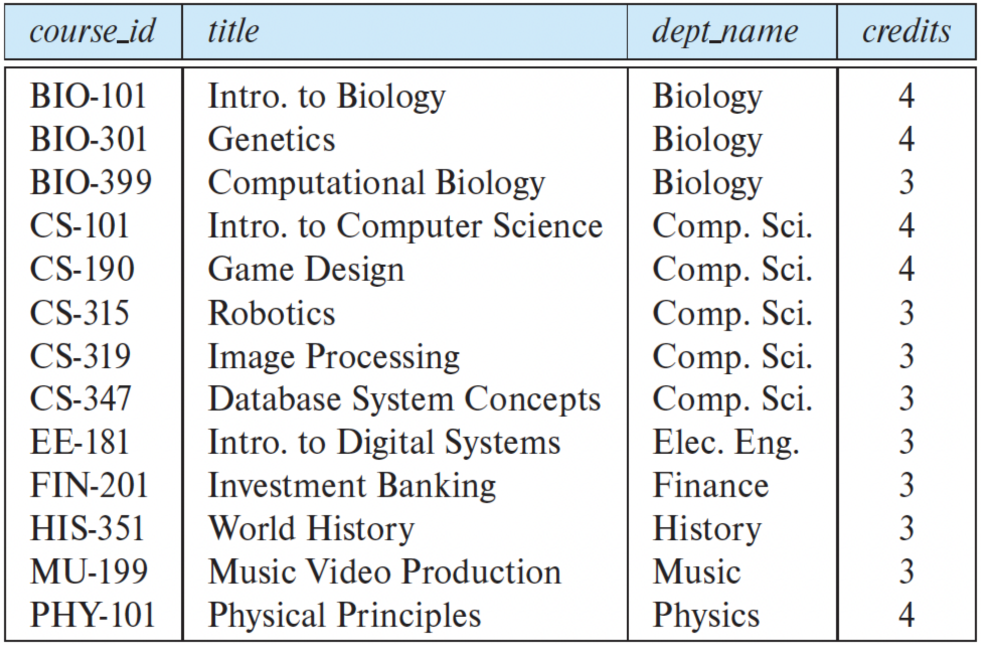
Step 4: Physical Design

- *instructor* (*ID, name, dept_name, salary*)
- *student* (*ID, name, dept_name, tot_cred*)
- *course* (*course_id, title, dept_name, credits*)
- *department* (*dept_name, building, budget*)

Table in DB

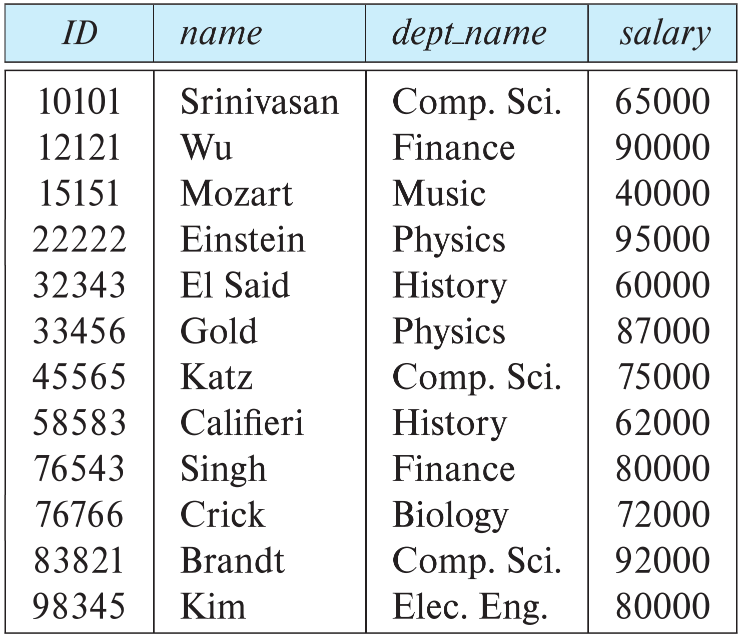


ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

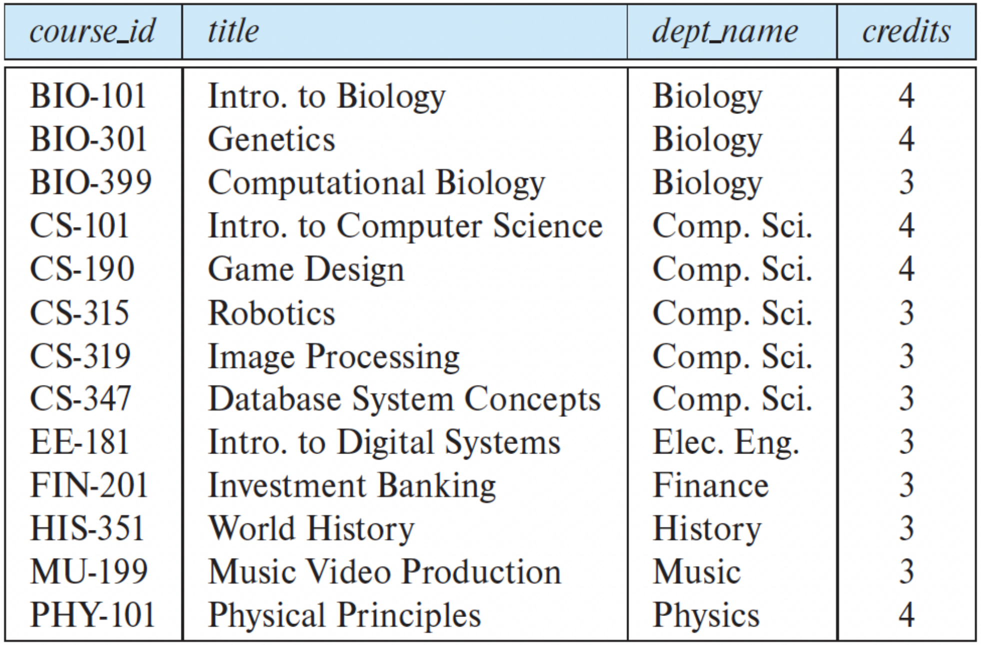


course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

Step 4: Physical Design



ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000



course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

⇩

- Memory, pages, data structures, indexes, devices

Steps of database design

- Requirement collection and analysis
  - What are we dealing with?
- Conceptual design: mapping to a conceptual model
  - What data and relationships have to be captured?
- Data model mapping
  - How to structure data in a specific model (here: the relational model)?
- Physical design: realization and implementation
  - Which adaptations and optimizations does a specific DBMS require?

A good design avoids redundancy and incompleteness.

Design Alternatives

- In designing a database schema, we must ensure that we avoid two major pitfalls:
  - Redundancy: a bad design may result in repeat 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:
- Normalization Theory
  - Formalize what designs are bad, and test for them

Agenda

- Data Modelling Process
- ER Model
- Mapping Cardinality Constraints
- Keys
- Weak Entity Types
- Specialization

ER model -- Database Modeling

- The ER model is used in conceptual design
  - The second step of the data modelling process
- The ER model employs three basic concepts:
  - Entity sets
  - Relationship sets
  - Attributes
- The ER model also has an associated diagrammatic representation, the ER diagram, which can express the overall logical structure of a database graphically.

Entity Types

- 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, salary*); *course=* (*course_id, title, credits*)
- An **entity type** describes the entity set.
  - *instructor* is an entity type with attributes: ID, name, salary

Entity Types--Example

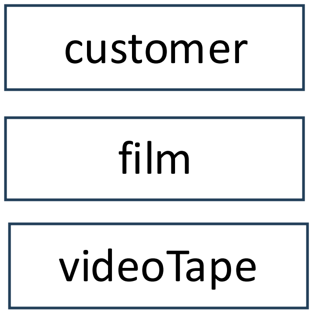
- Video Store Database

  Customer

  Film

  Video tape

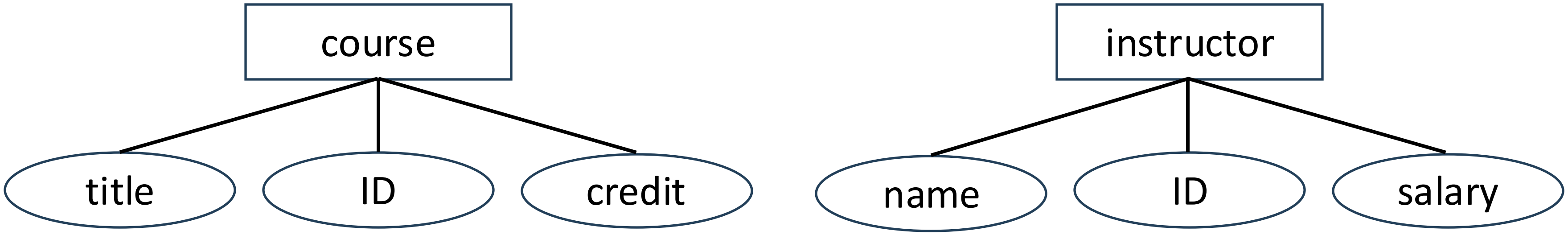
    - For each film, the store maintains a number of tapes



- Entity types are represented in **rectangles**
- How to find entities when modelling?
  - Look for nouns in the database requirements.

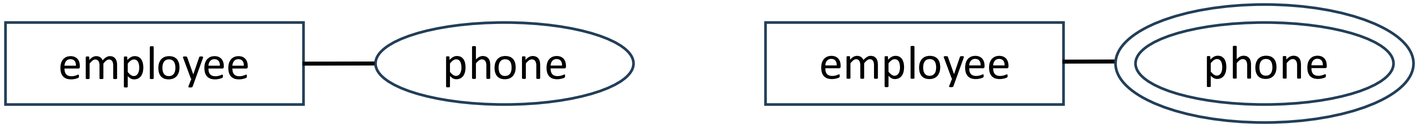
Attributes

- **Attributes** model characteristics of entities or relationships.
  - All entities of an entity type have the same characteristics
  - Attributes are declared for entity types
  - Attributes have a domain or value set

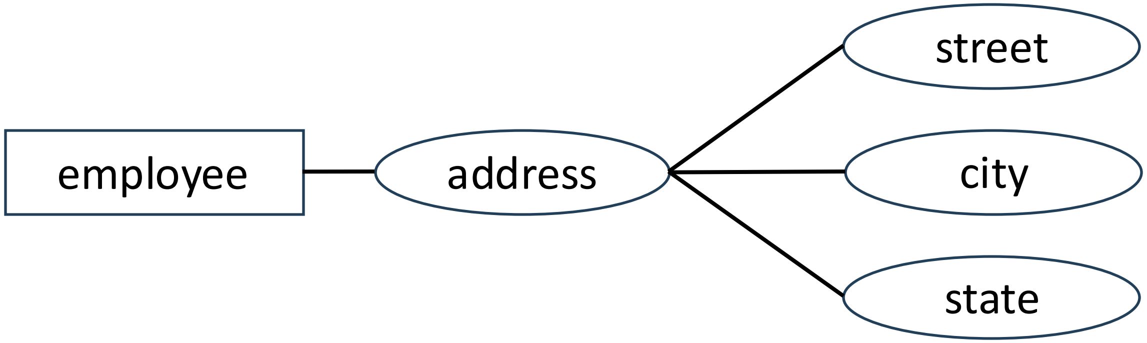


Attributes

- **Single-valued** vs. **multi-valued** attributes
  - An employee might have multiple phone numbers (or a single one)

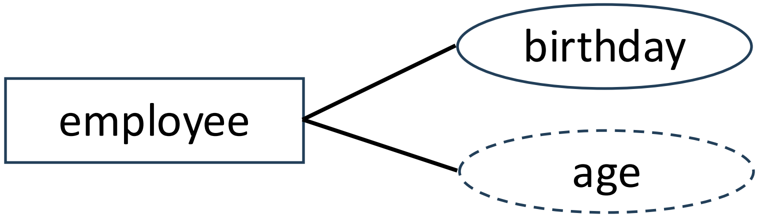


- **Simple attributes** vs. **composite attributes**
  - An address can be modelled as a string or composed of street, city and state

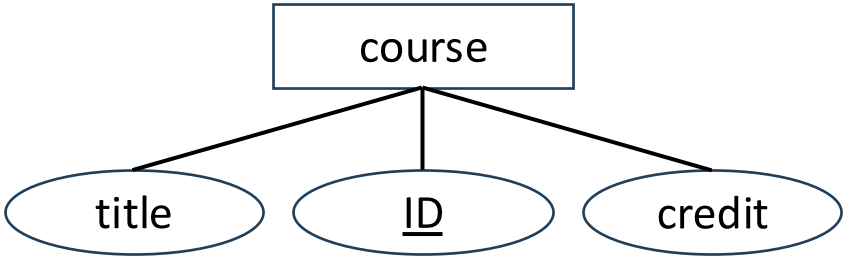


Attributes

- **Stored** attributes vs. **derived** attributes
  - Age can be derived from a stored birth-date

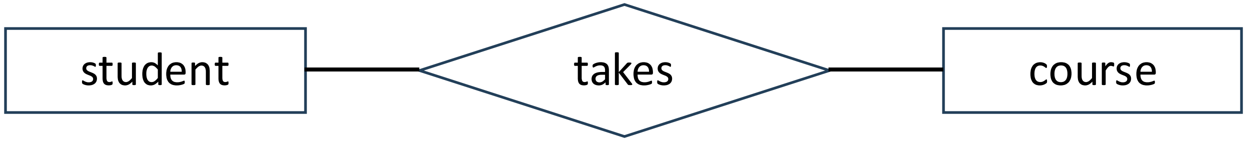


- **Primary keys**
  - Uniquely identifies an entity within an entity set



Relationships

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



- A **relationship set** is a mathematical relation among n ≥ 2 entities, each taken from 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

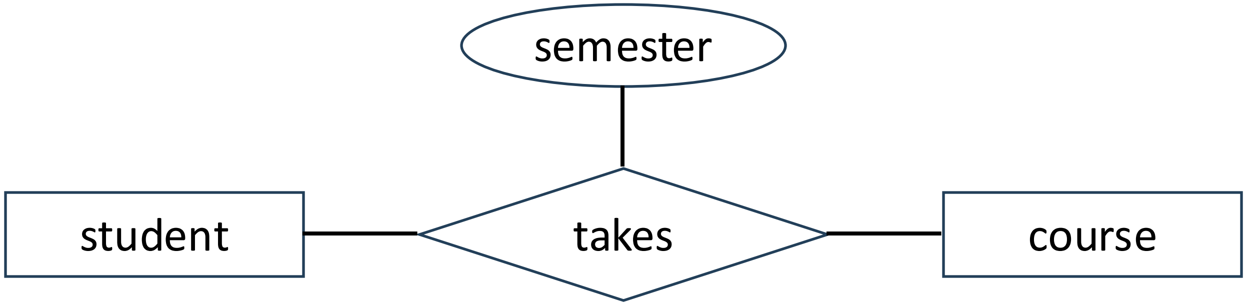
- Example:
  - (student1, Computer Science) $\in$ takes
- A **relationship type** describes a relationship set

Relationships

- We say each entity set E_i **participates** in this relationship set

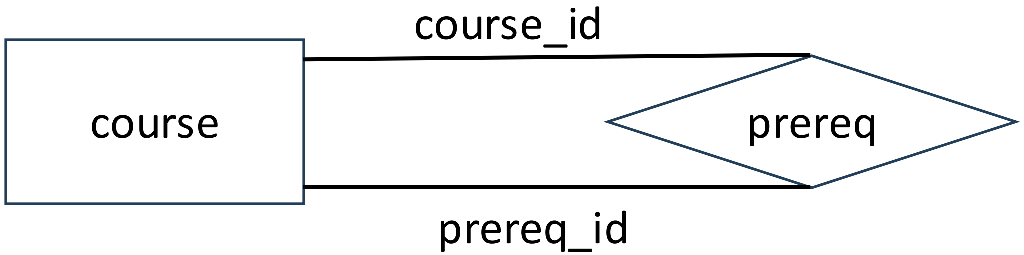
  $$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$$

  - n-ary relationship set in general
  - Binary relationship set if n=2
- A **diamond** represents a relationship set
- A relationship type can also have attributes.



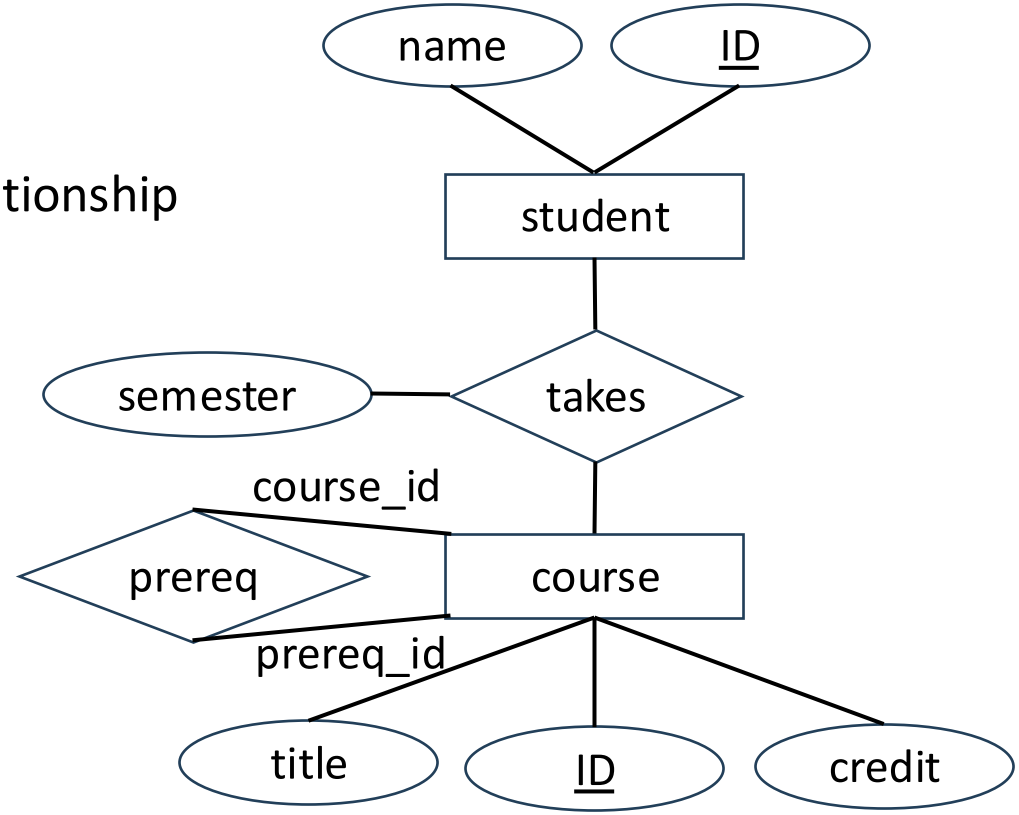
Roles

- Entity sets of a relationship need not be distinct
  - Each occurrence of an entity set plays a “role” in the relationship
- Role names are optional and used to characterize a relationship type.
  - Especially useful for recursive relationship types, i.e., an entity type is participating multiple times in a relationship type.
- The labels “*course_id*” and “*prereq_id*” are called roles.



Example

- Students take courses
  - 1: Entity→Entity type
  - 2: Relationship → Relationship type
  - 3: Attribute
  - 4: Primary key
  - 5: Role

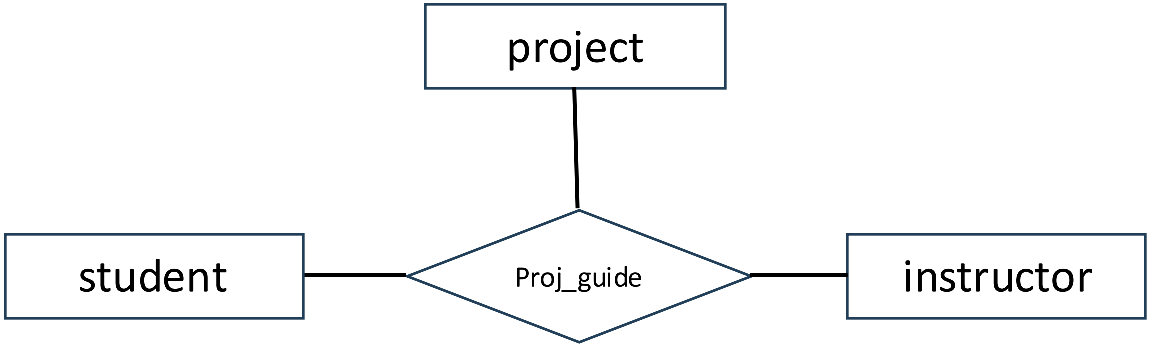


Degree of a Relationship Set

- Degree: Number of participating entity types
- 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 are binary.
  - Example: 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



Exercise 1

- Library database:
  - It contains several branches. Each branch has a name, location, and a unique branch ID.
  - A book has a title, author, and ISBN (unique for each book).
  - A member has a unique member ID, a name, and a phone number.
  - Each branch holds many books
  - Members of the library can borrow books.
- Consider the requirement of Library Database:
  - Identify the entities, their attributes, and relationships for the Library Database.
  - Create an ER diagram that models the database.

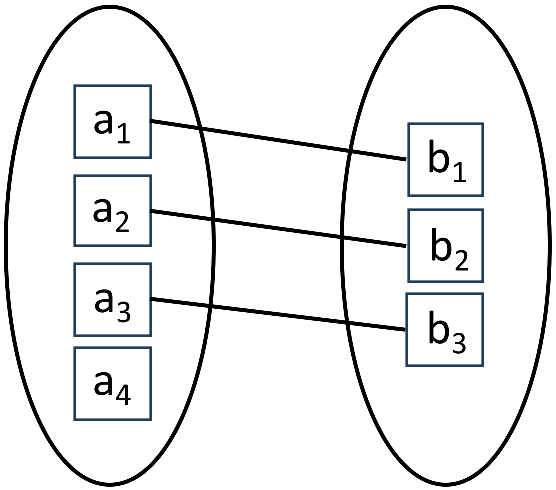
Agenda

- Data Modelling Process
- ER Model
- Mapping Cardinality Constraints
- Keys
- Weak Entity Types
- Specialization

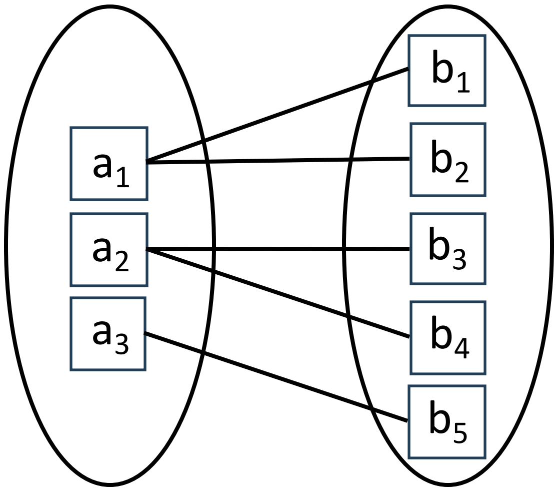
Mapping Cardinality Constraints

- 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



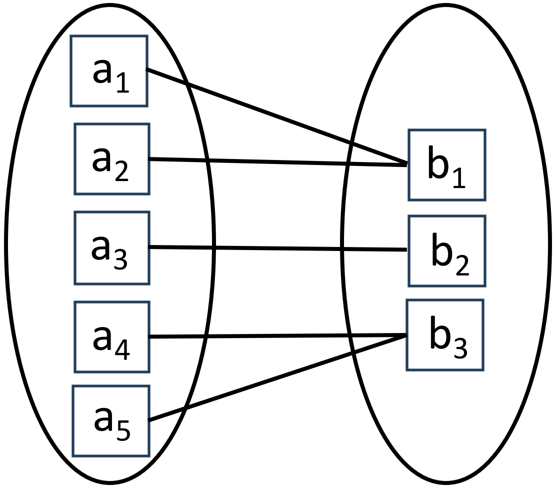
One to one



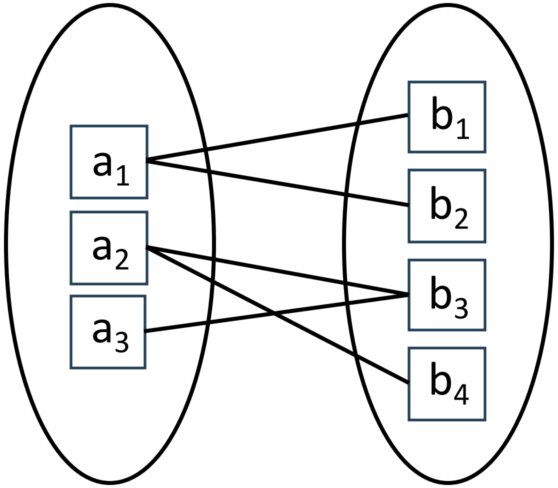
One to many

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

Mapping Cardinalities



many to one

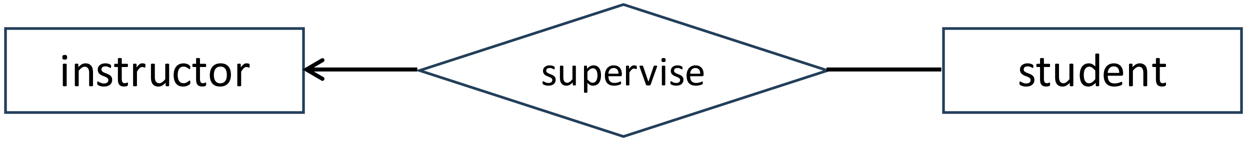


many to many

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

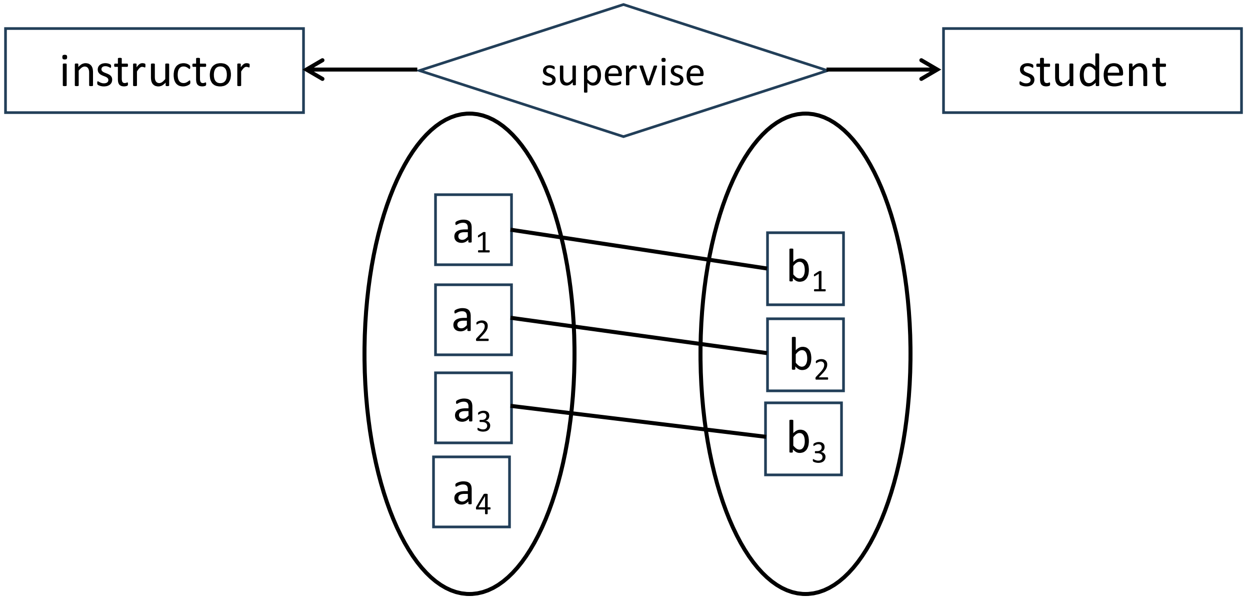
Cardinality Constraints--Graphical notation

- We 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.



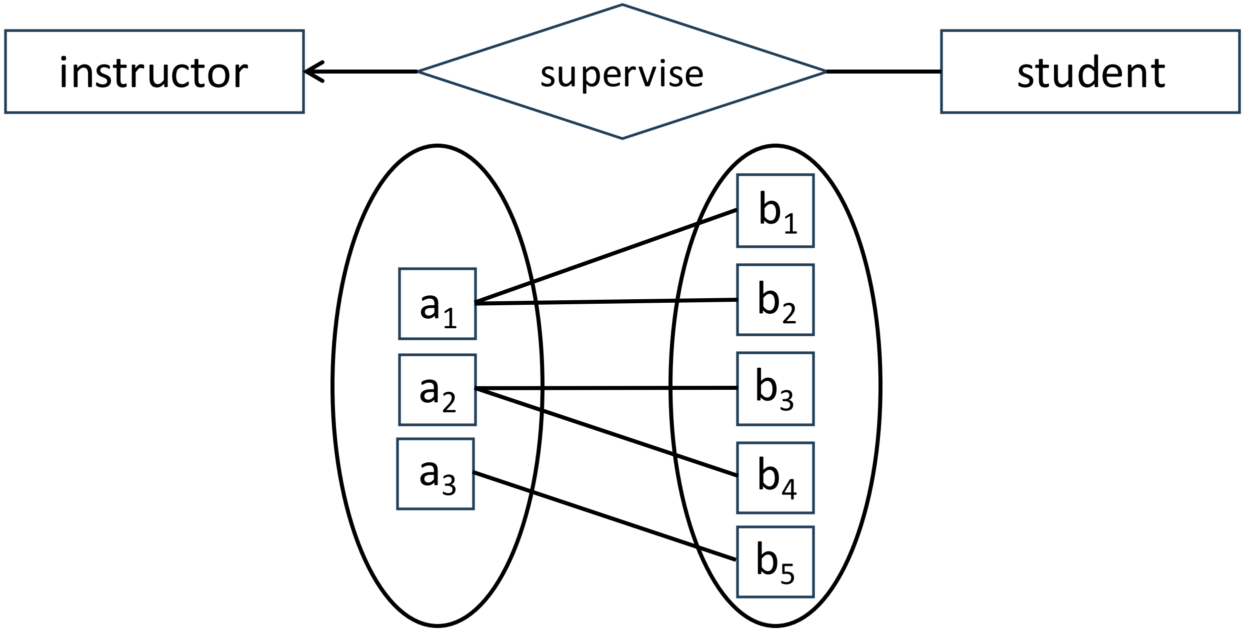
One-to-one relationship

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



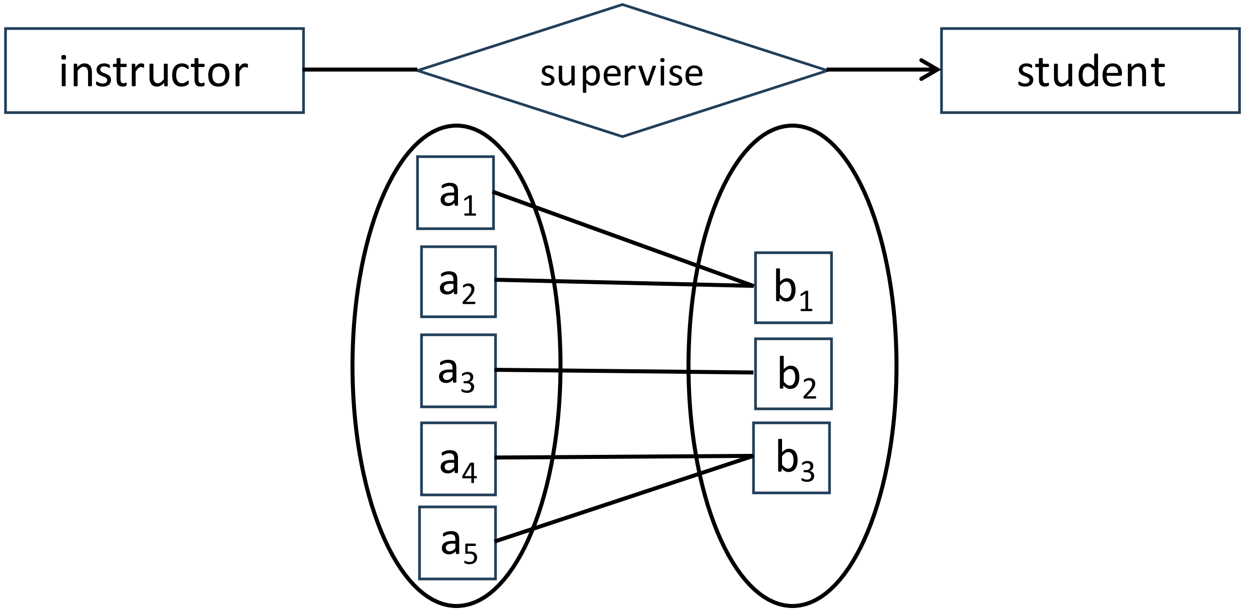
One-to-many relationship

- One-to-many relationship between an *instructor* and a *student* :
  - An *instructor* is associated with several (including 0) *student* via the relationship *supervise*
  - A *student* is associated with at most one *instructor* via the relationship *supervise*



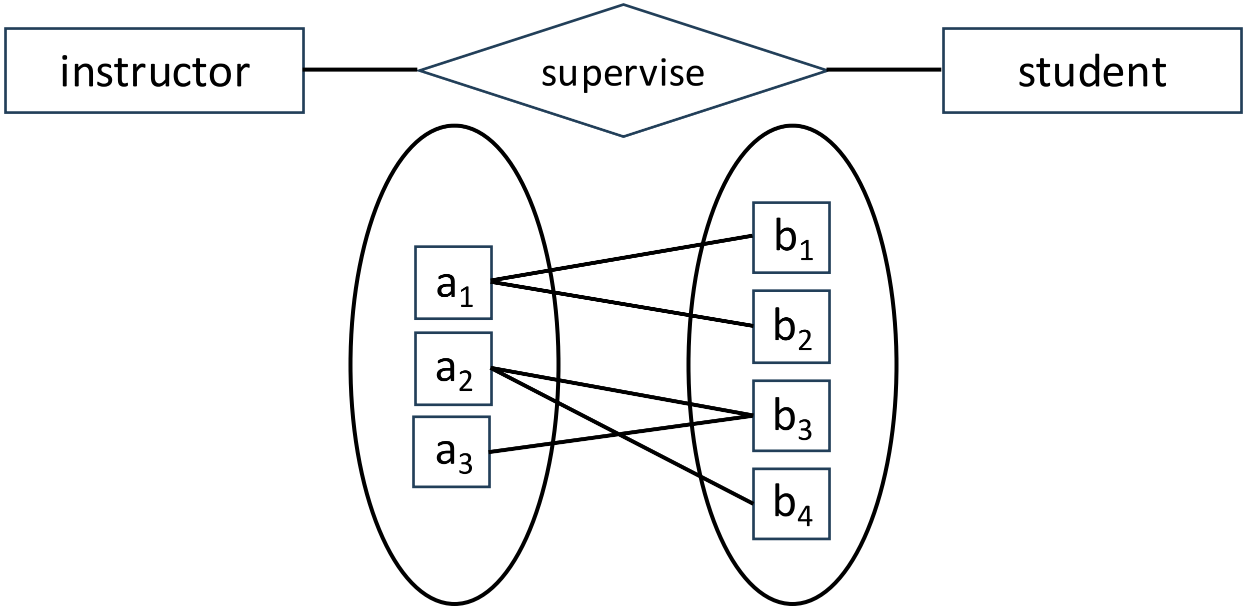
Many-to-one relationship

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

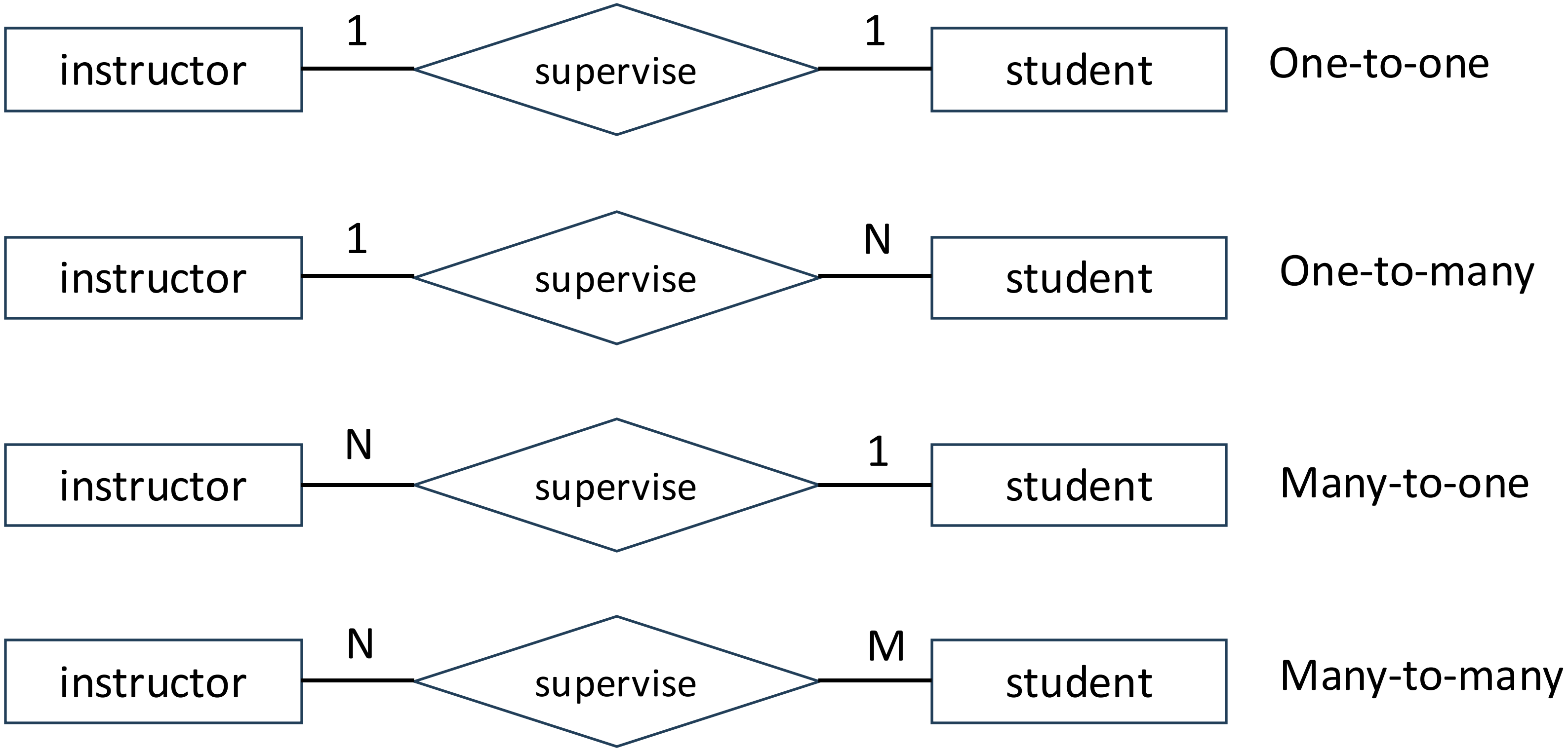


Many-to-many relationship

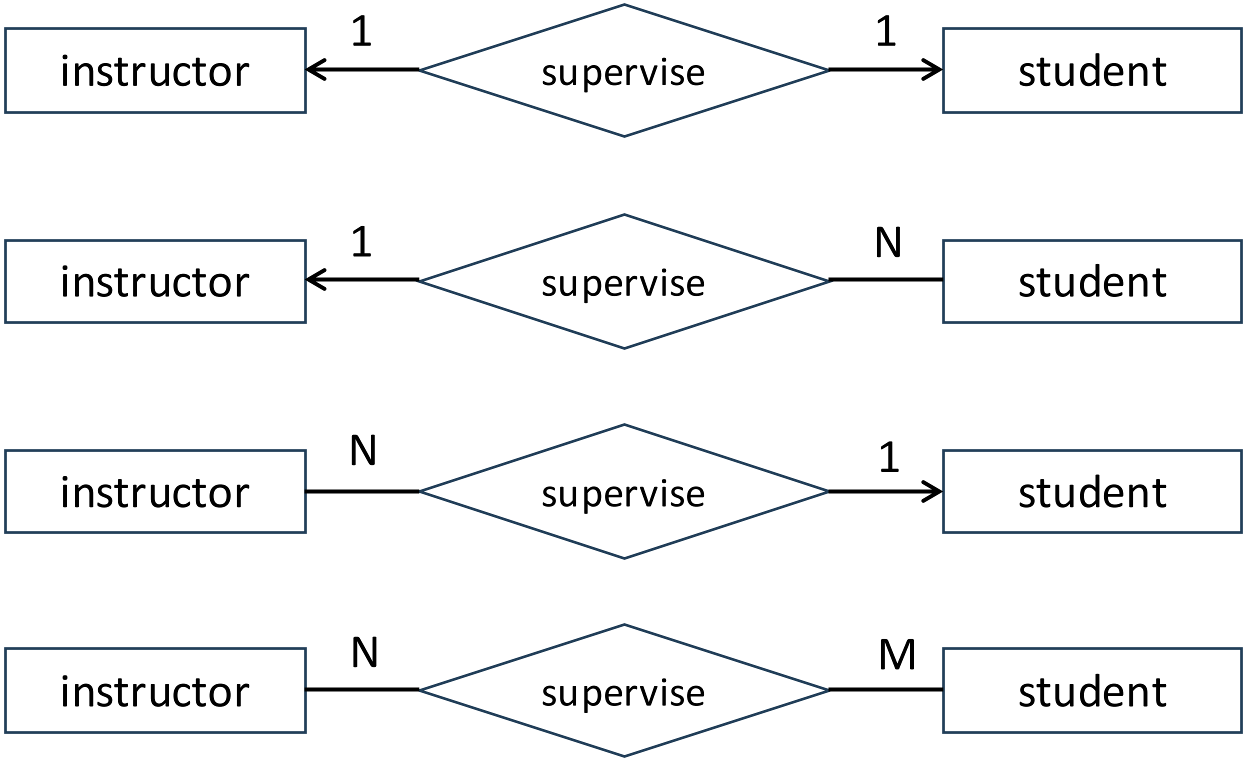
- Many-to-many relationship between an *instructor* and a *student* :
  - An *instructor* is associated with several (including 0) *student* via the relationship *supervise*
  - A *student* is associated with several (including 0) *instructor* via the relationship *supervise*



Cardinality Constraints--Chen Notation



Cardinality Constraints



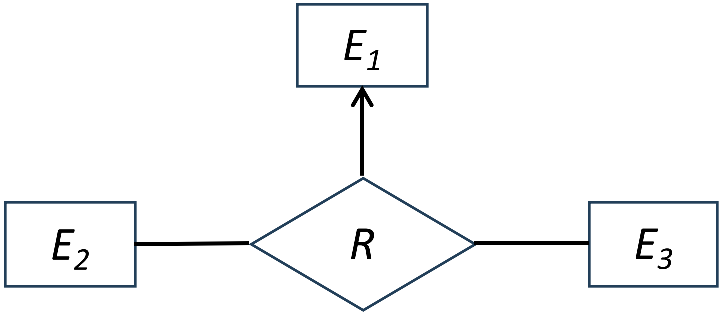
One-to-one

One-to-many

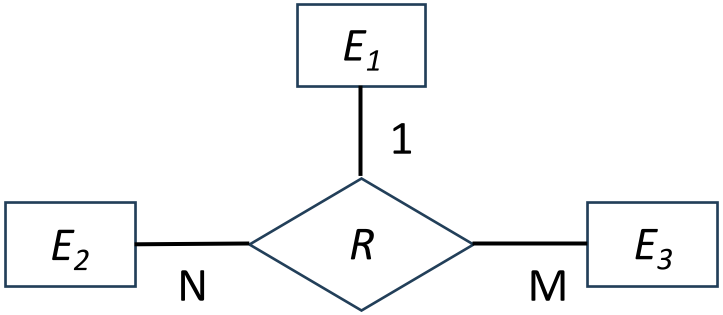
Many-to-one

Many-to-many

Cardinality Constraints for Ternary Relationship



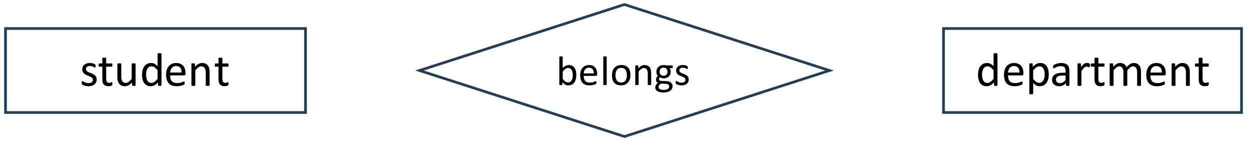
Graphical notation



Chen notation

Using arrows or annotating lines with 1, N, M, etc. is equivalent. Having both is not necessary but sometimes useful for clarification.

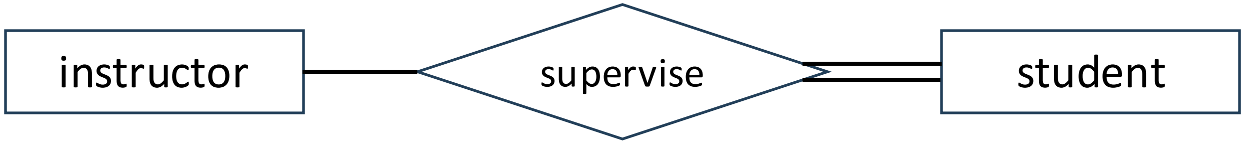
Exercise 2



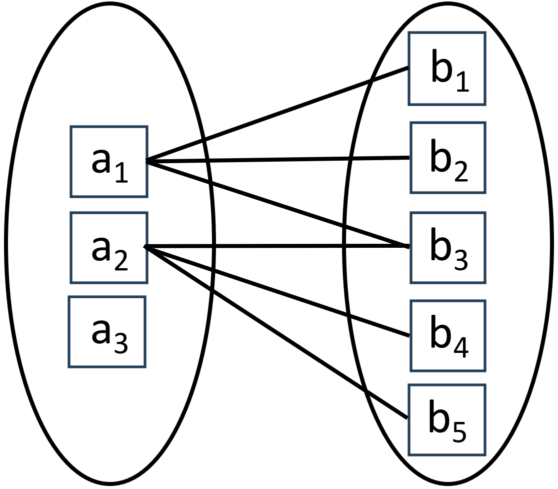
- Use Graphical Notation and Chen Notation to represent the relationship
  - One student belongs to only one department
  - One department can have many students

Total and Partial Participation

- **Total participation** (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
- **Partial participation**: some entities may not participate in any relationship in the relationship set



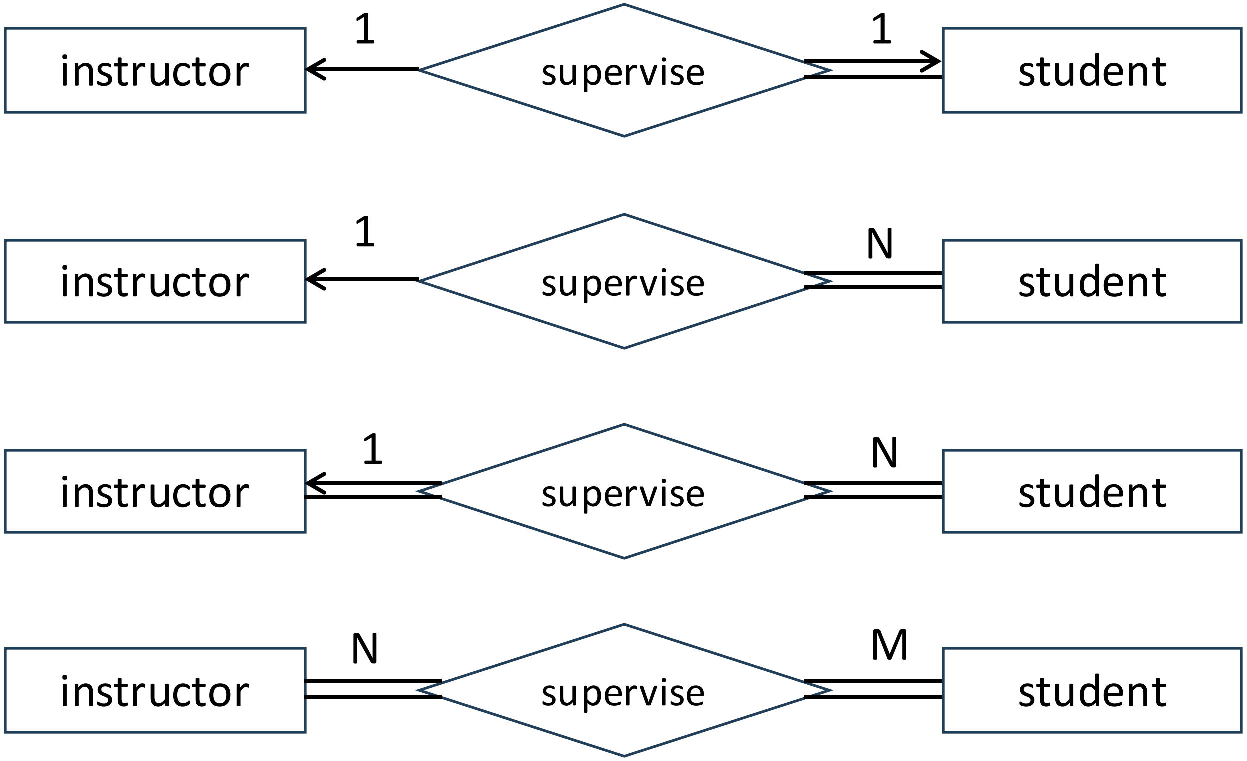
Participation of *instructor* in *supervise* is **partial.**



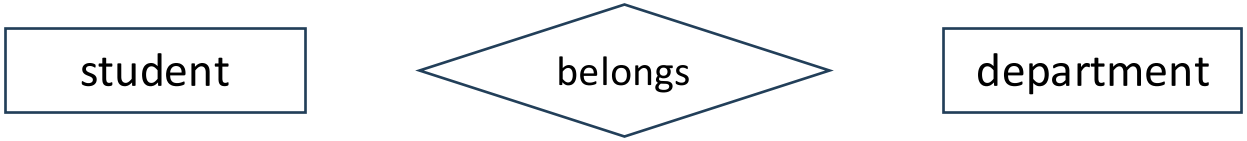
Participation of *student* in *supervise* is **total.**

Every *student* must have an associated instructor

Participation Constraints--Example



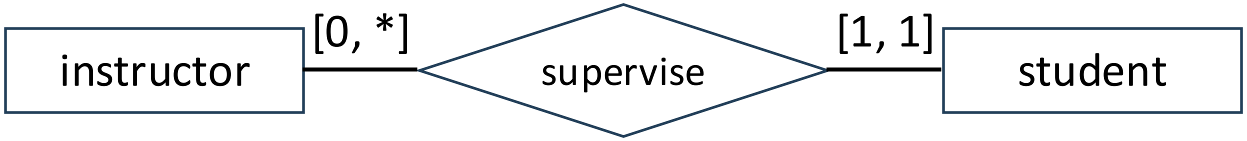
Exercise 3



- Use Graphical Notation and Chen Notation to represent the relationship
  - Each student must belong to one department
  - Each department can have many students or have no students

Cardinality Constraints--[min, max] Notation

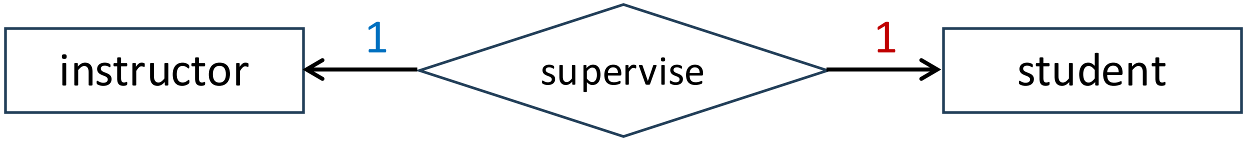
- A line may have an associated minimum and maximum cardinality, shown in the form [*min, max*], where *min* is the minimum and *max* the maximum cardinality
  - A minimum value of 1 indicates total participation
  - A maximum value of 1 indicates that the entity participates in at most one relationship
  - A maximum value of * indicates no limit



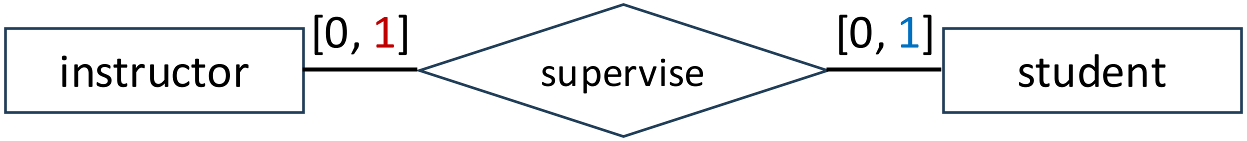
An instructor can advise 0 or more students.

A student must have 1 instructor; cannot have multiple instructors.

Cardinality Constraints--Example



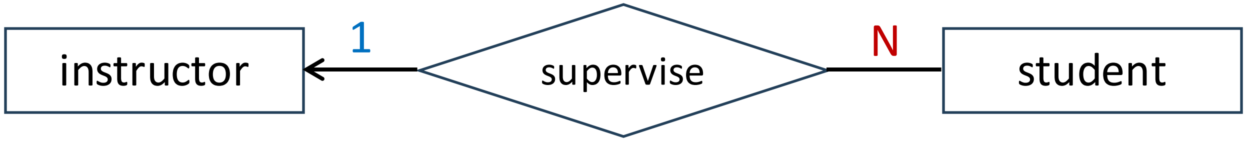
An instructor **can** supervise at most 1 (0 or 1) student. (Partial)
A student **can** have at most 1 (0 or 1) instructor. (Partial)



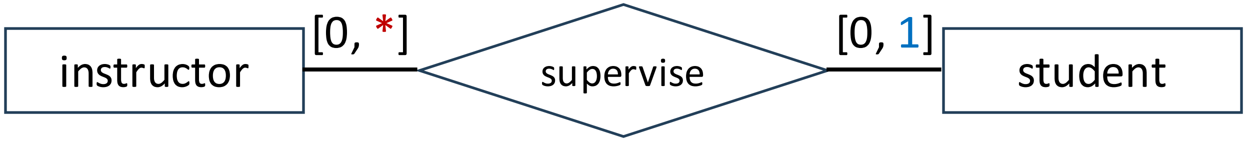
[min, max] Notation

An instructor **can** supervise at most 1 (0 or 1) student. (Partial)
A student **can** have at most 1 (0 or 1) instructor. (Partial)

Cardinality Constraints--Example



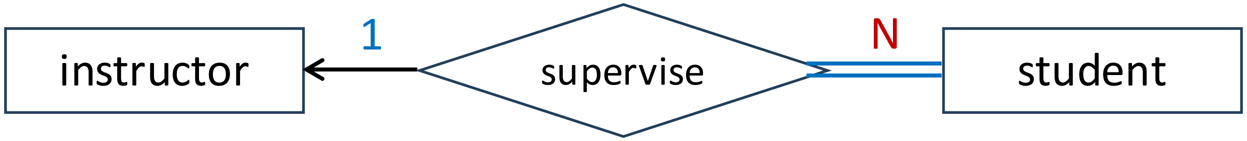
An instructor **can** supervise 0 or more students. (Partial)
A student **can** have at most 1 (0 or 1) instructor. (Partial)



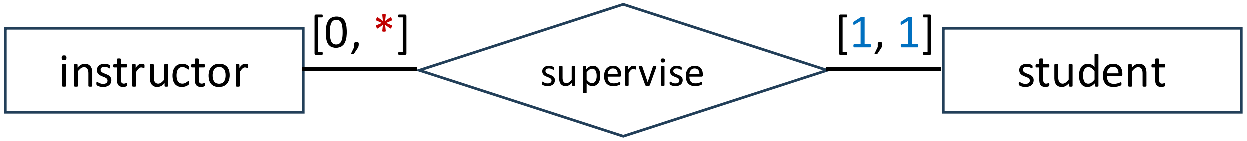
[min, max] Notation

An instructor **can** supervise 0 or more students. (Partial)
A student **can** have at most 1 (0 or 1) instructor. (Partial)

Cardinality Constraints--Example



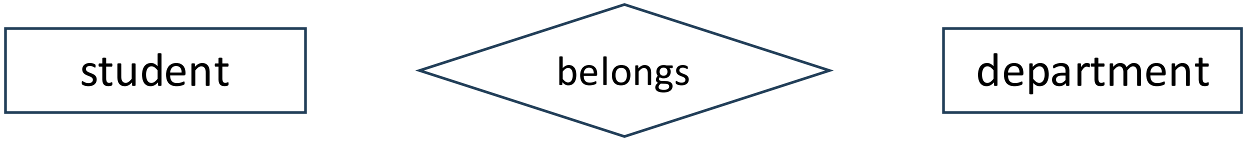
An instructor **can** supervise 0 or more students. (Partial)
A student **must** have 1 instructor. (Total)



[min, max] Notation

An instructor **can** supervise 0 or more students. (Partial)
A student **must** have 1 instructor. (Total)

Exercise 4



- Use [min, max] Notation represent the relationship
  - Each student must belong to one department
  - Each department can have many students or have no students

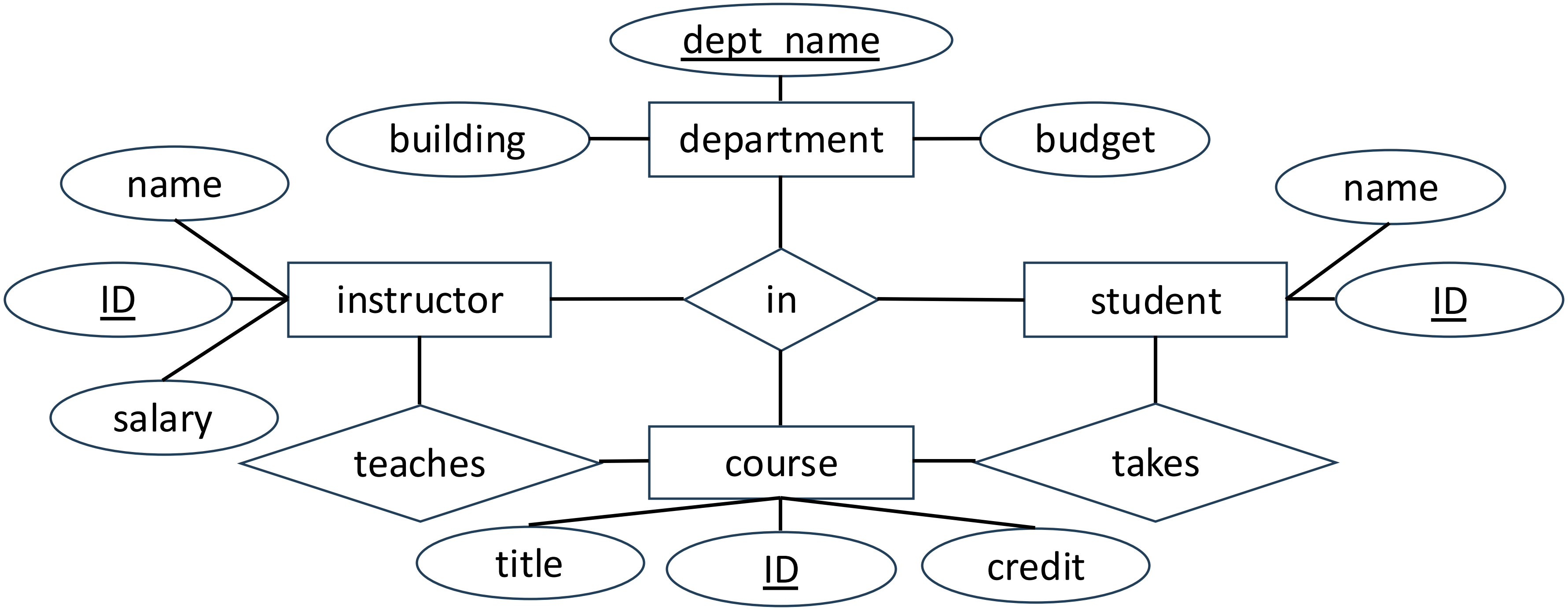
Agenda

- Data Modelling Process
- ER Model
- Mapping Cardinality Constraints
- Keys
- Weak Entity Types
- Specialization

Primary key for Entity Sets

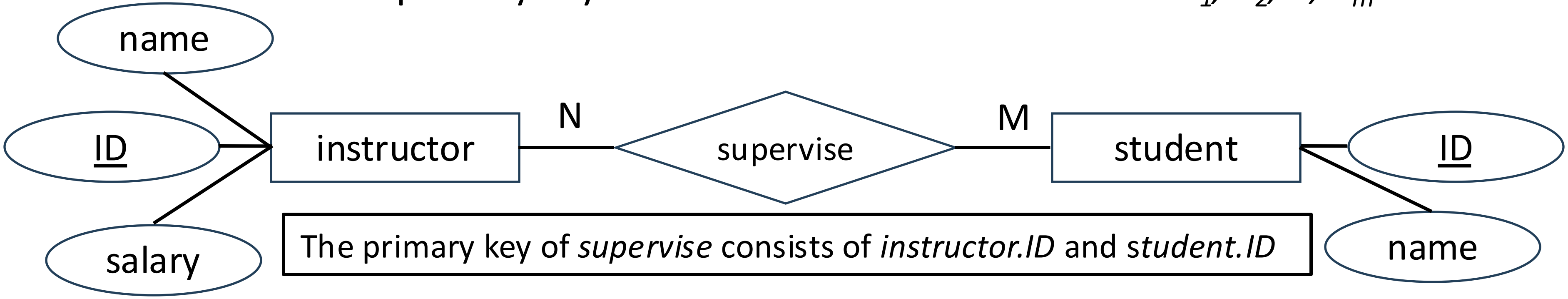
- By definition, individual entities are distinct.
- From database perspective, the differences among them must be expressed in terms of their attributes.
- The values of the attribute values of an entity must be such that they can uniquely identify the entity.
  - No two entities in an entity set are 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 Keys



Primary key for Relationship Sets

- To distinguish among the various relationships of a relationship set we use the individual primary keys of the entities in the relationship set.
    - Let *R* be a relationship set involving entity sets $E_1, E_2, \ldots, E_n$
    - The primary key for R is consists of the union of the primary keys of entity sets $E_1, E_2, \ldots, E_n$
    - 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 $a_1, a_2, .., a_m$



The primary key of *supervise* consists of *instructor.ID* and *student.ID*

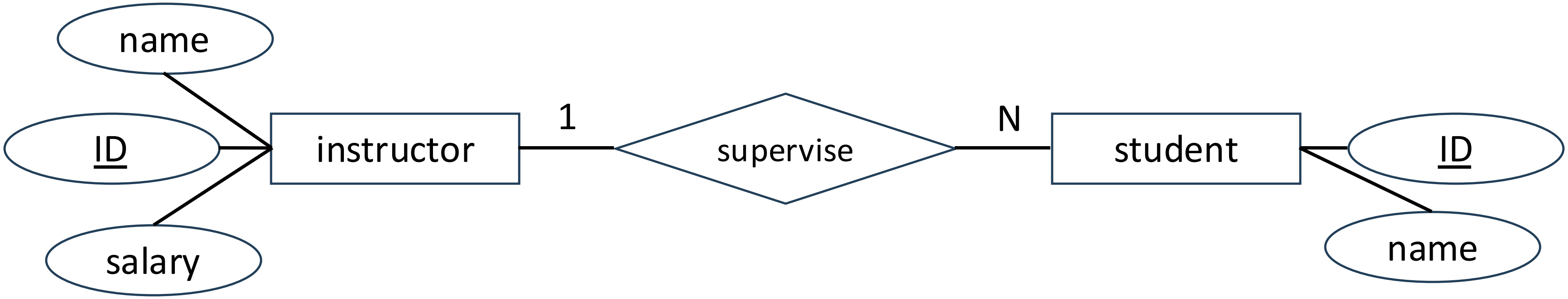
The choice of the primary key for a relationship set depends on the mapping cardinality of the relationship set.

Choice of Primary key for Binary Relationship

- Many-to-Many relationships.
  - The the union of the primary keys of entity sets is chosen as the primary key.
- One-to-Many relationships .
  - The primary key of the "Many" side is used as the primary key.
- Many-to-one relationships.
  - The primary key of the "Many" side is used as the primary key.
- One-to-one relationships.
  - The primary key of either one of the participating entity sets can be chosen as the primary key.

Choice of Primary key for Binary Relationship

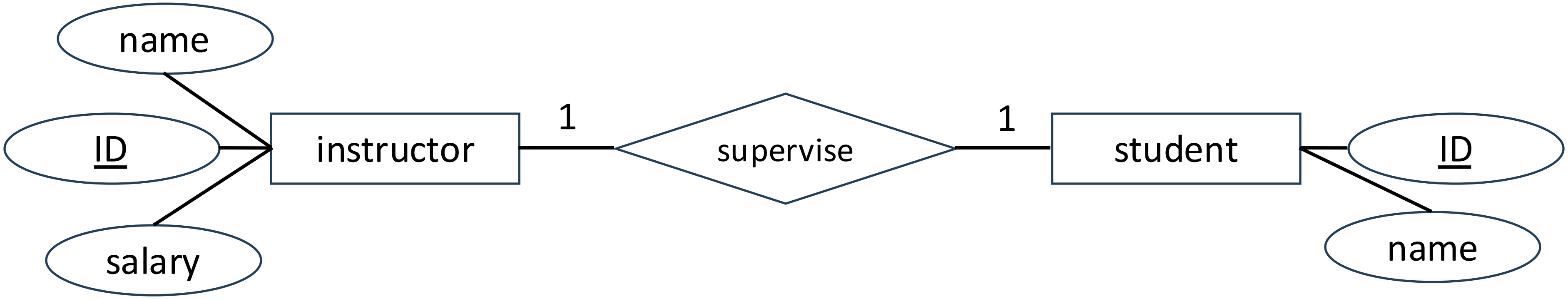
- One-to-Many relationships .
  - The primary key of the “Many” side is used as the primary key.



The primary key of *supervise* is *student.ID*

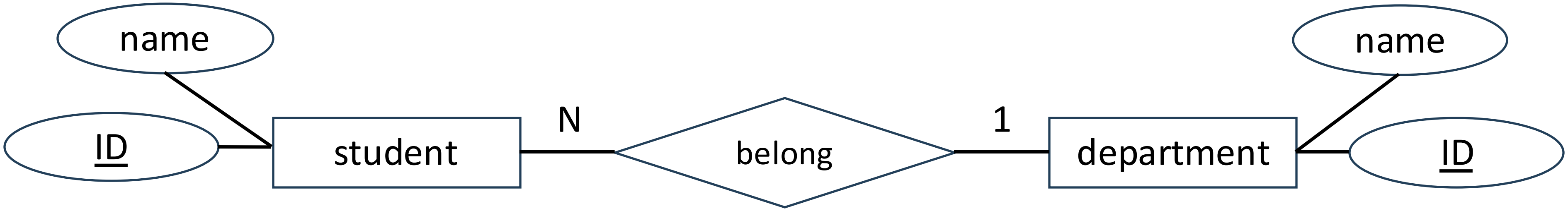
Choice of Primary key for Binary Relationship

- One-to-one relationships.
  - The primary key of either one of the participating entity sets can be chosen as the primary key.



The primary key of *supervise* can be either *instructor.ID* or *student.ID*

Exercise 5

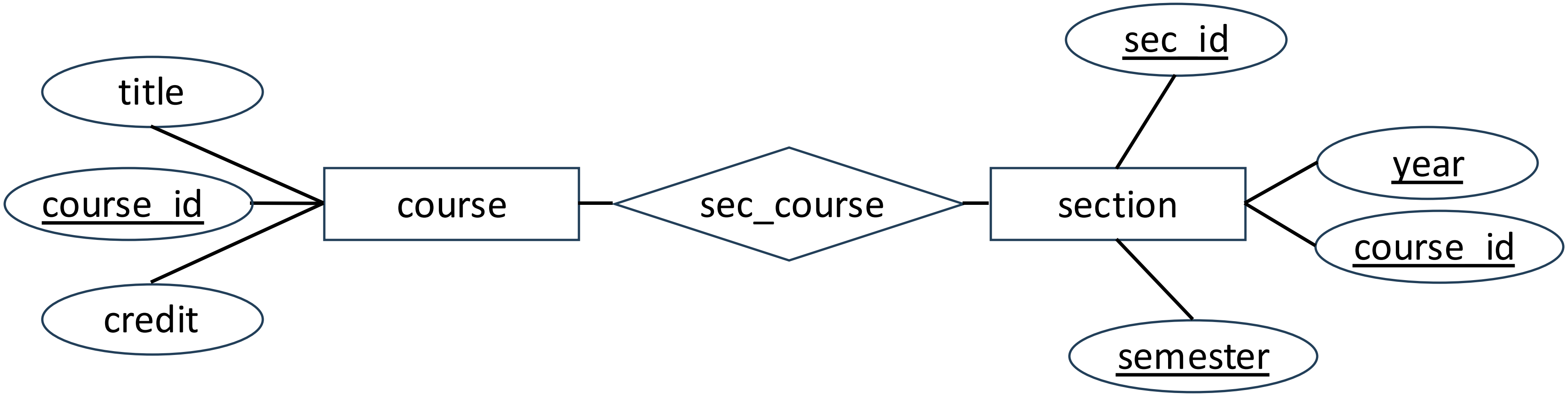


- Which attribute should be the primary key of *belong*
  - student.ID
  - department.ID
  - student.name
  - department.name

Agenda

- Data Modelling Process
- ER Model
- Mapping Cardinality Constraints
- Keys
- Weak Entity Types
- Specialization

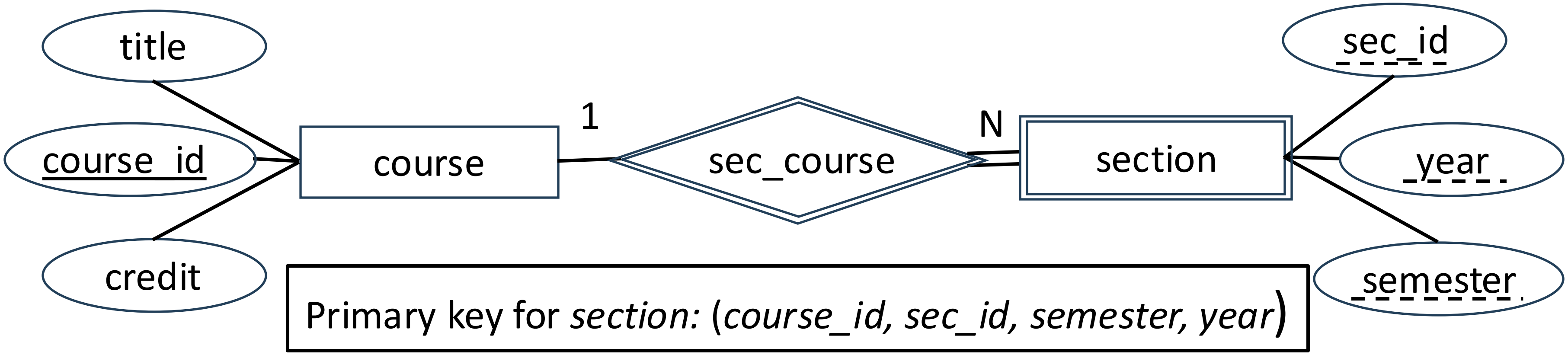
Weak Entity Types



- A **weak entity type** is one whose existence is dependent on another entity, called its **identifying entity**

Expressing Weak Entity Types

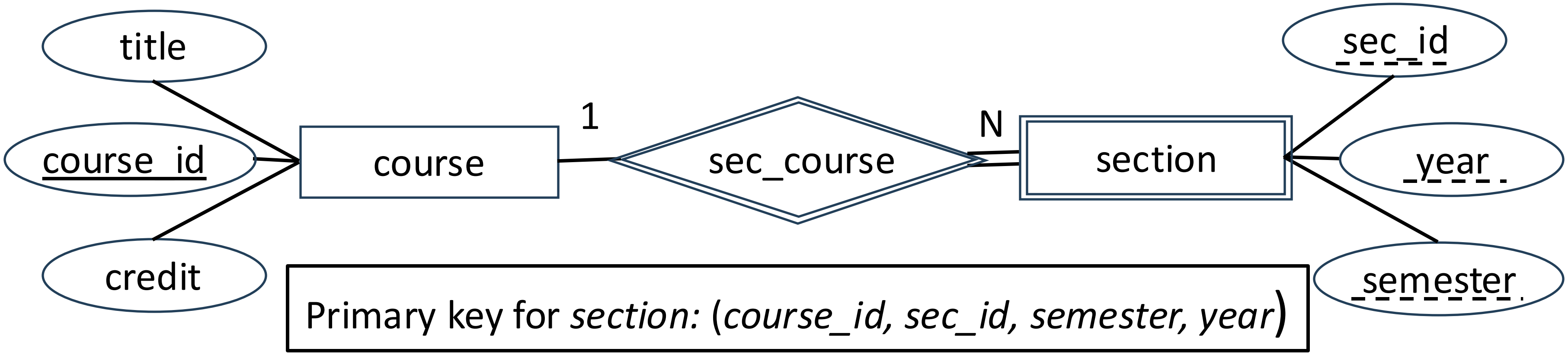
- In E-R diagrams, a weak entity set is depicted via a double rectangle.
- We underline the discriminator of a weak entity set with a dashed line.
- The relationship set connecting the weak entity set to the identifying strong entity set is depicted by a double diamond.



Primary key for *section:* (*course_id, sec_id, semester, year*)

Expressing Weak Entity Types

- Total participation of the weak entity type.
- Only in combination with 1:N (N:1) (or rarely also 1:1) relationship types
- The strong entity type is always on the "1"-side



Primary key for *section:* (*course_id, sec_id, semester, year*)

Agenda

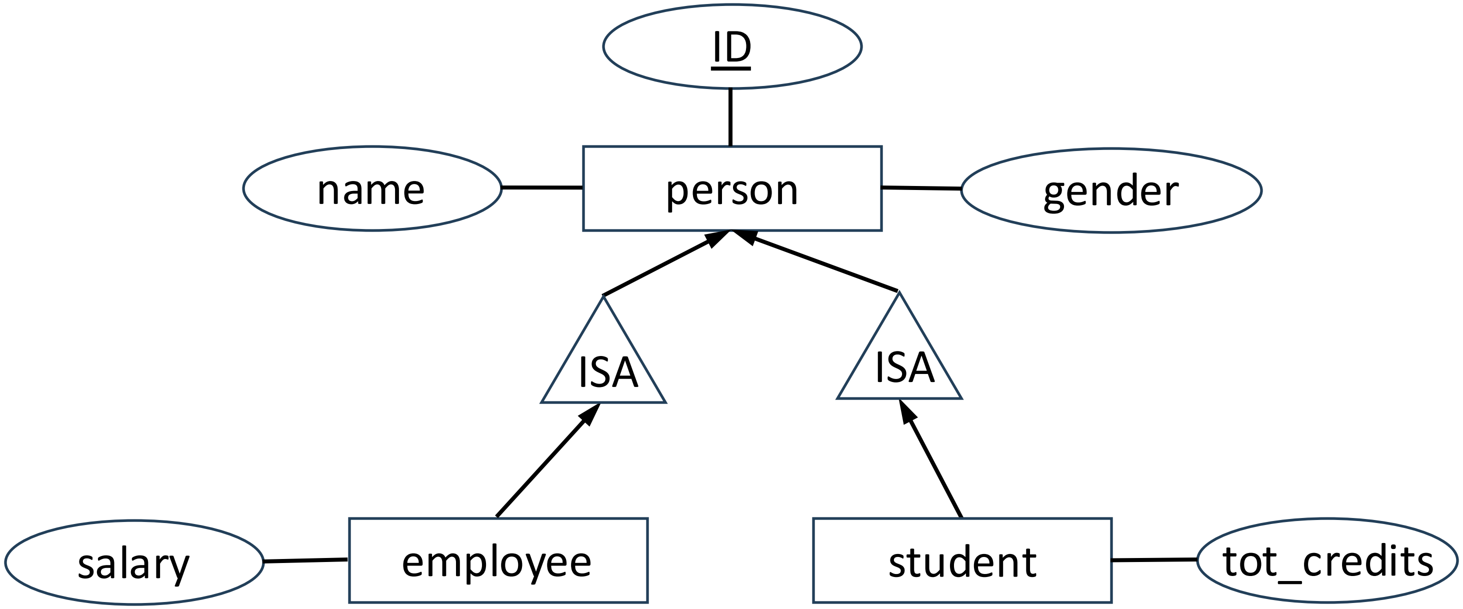
- Data Modelling Process
- ER Model
- Mapping Cardinality Constraints
- Keys
- Weak Entity Types
- Specialization

Specialization

- **Top-down design process:** we designate sub-groupings within an entity set that are distinctive from other entities in the set.
- These sub-groupings become lower-level entity sets that have attributes or participate in relationships that do not apply to the higher-level entity set.
- Depicted by a *triangle* component labeled ISA (e.g., *instructor* "is a" *person*).

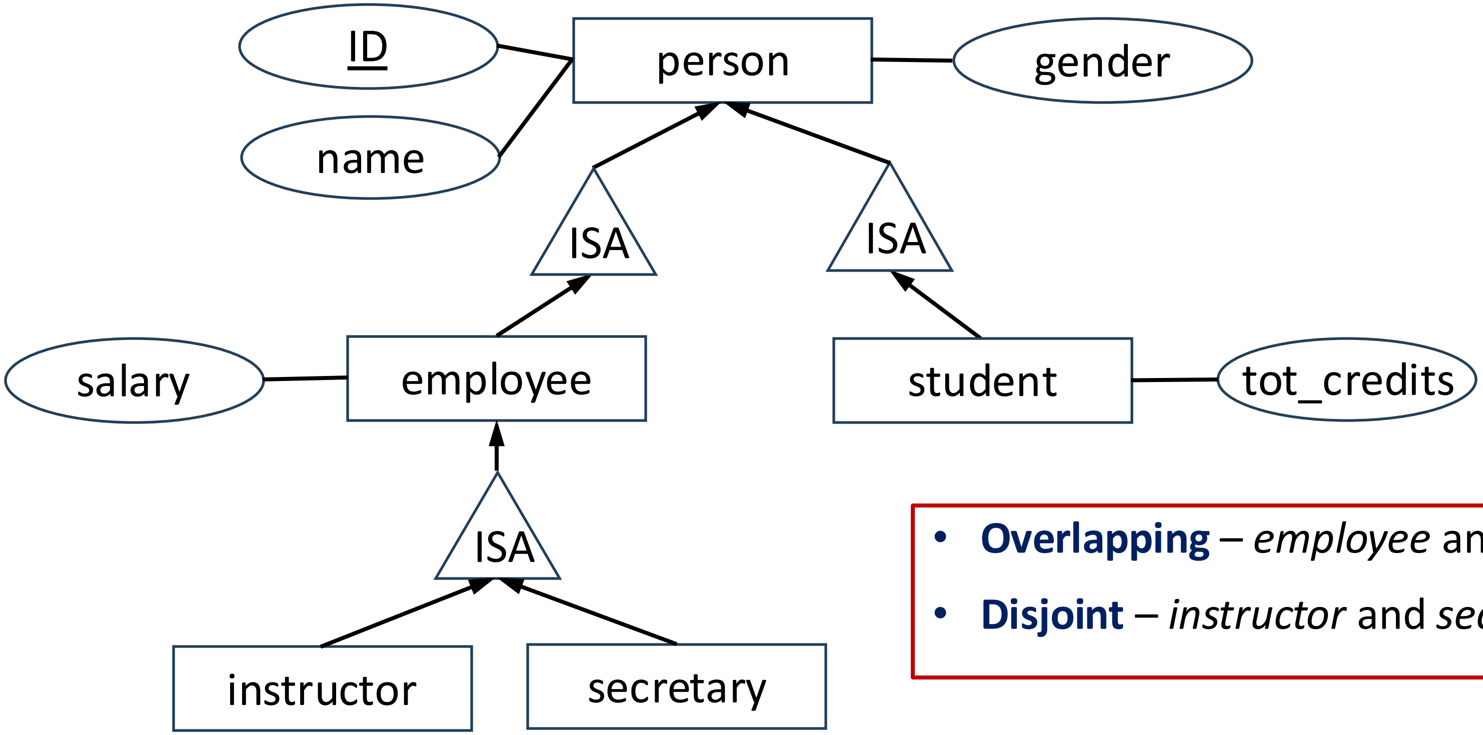
Specialization--Example

- **Attribute inheritance** – a lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked.



Specialization--Example

- **Overlapping specialization--** An entity may belong to multiple specialized entity sets.
  - separate ISA symbols are used
- **Disjoint specialization--** An entity may belong to at most one specialized entity set.
  - arrows to a shared ISA symbol in the diagram



- **Overlapping** – *employee* and *student*
- **Disjoint** – *instructor* and *secretary*

Design notes

- Entities correspond to nouns, relationships to verbs.
- Each statement in the requirement specification should be reflected somewhere in the ER schema.
- Each ER diagram (ERD) should be located somewhere in the requirement specification.
- Conceptual design often reveals inconsistencies and ambiguities in the requirement specification, which must be first resolved.

Summary

- Data Modelling Process
- ER Model
- Mapping Cardinality Constraints
- Keys
- Weak Entity Types
- Specialization

Next Lecture

- The Entity-Relationship Model
  - Create non-trivial ER diagrams
  - Analyze if an ER diagram is good or bad
  - Create and explain the mapping of ER diagrams to relations
  - Use a particular ER notation properly