CSE 403

Software Engineering
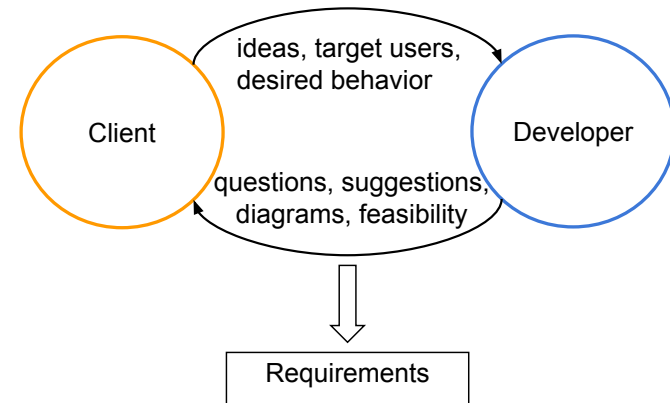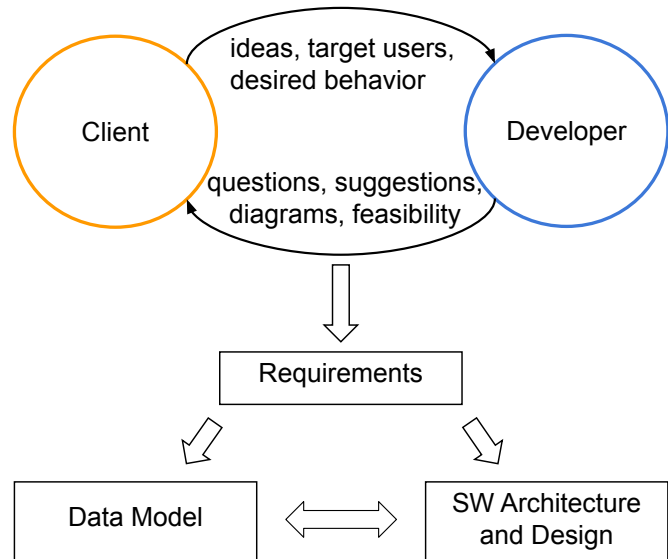
Data modelling

From Requirements to System Design
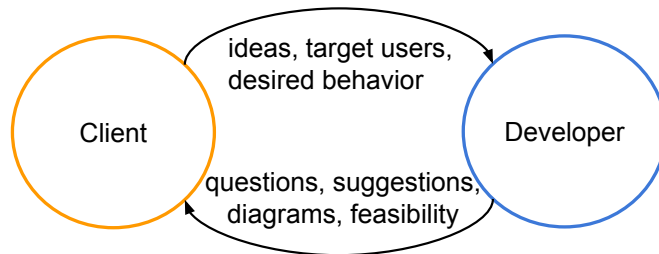
From Requirements to System Design

From Requirements to System Design

From Requirements to System Design

Client — ideas, target users, desired behavior → Developer

questions, suggestions, diagrams, feasibility

↓

Requirements — **What**

↓ ↓

Data Model ⟷ SW Architecture and Design — **How**

From Requirements to System Design

Client — ideas, target users, desired behavior → Developer

questions, suggestions, diagrams, feasibility

↓

Requirements

↓ ↓

Today → Data Model ⟷ SW Architecture and Design
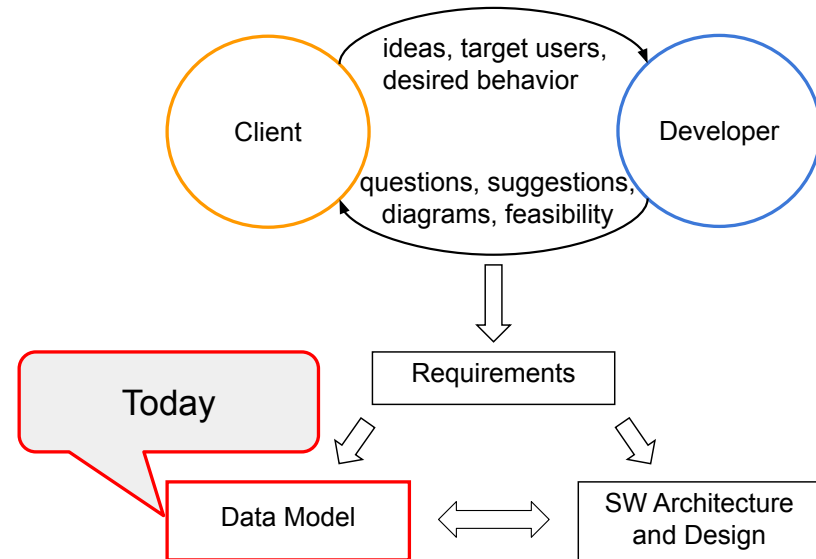
Data Modelling

Goals for today

- How to model data?
  - Identify Entities
  - Identify Attributes
  - Identify Relationships
  - Assign Keys
  - *(Normalization to reduce redundancy)*
  - *(Denormalization to improve performance)*

- Common "language" for data modelling
  - ER (Entity-Relationship) diagrams
  - Just one out of many possibilities (diagrams, tables, text)

- Develop a data model for a course-registration system

ER diagrams: overview

- An Entity Relationship (ER) diagram is a **graphical representation** of a **data model**.
- It shows the **relationship** between **entities** (e.g., people, objects, events, or concepts) within a system.
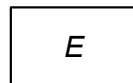- It **can be mapped** to a **relational** (database) **schema**.
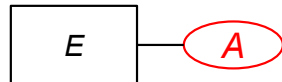
ER diagrams: graphical syntax

- An entity E
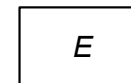
$$\boxed{E}$$

ER diagrams: graphical syntax

- An entity E

$$\boxed{E}$$

- An attribute A of entity E

$$\boxed{E} - (A)$$

ER diagrams: graphical syntax

- An entity E

$$\boxed{E}$$

- An attribute A of entity E

$$\boxed{E} - (A)$$

- A relationship R between two entities $E1$ and $E2$
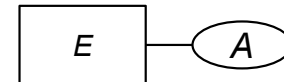
$$\boxed{E_1} - \diamond R \diamond - \boxed{E_2}$$
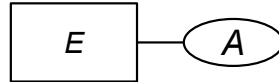
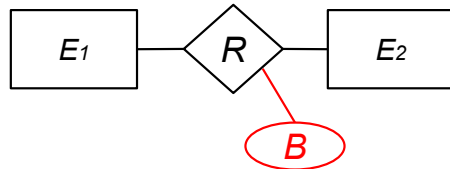## ER diagrams: graphical syntax

- An entity E

- An attribute A of entity E

- A relationship R between two entities $E1$ and $E2$

- An attribute B of relationship R
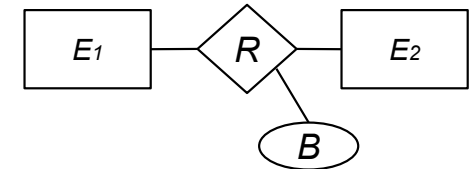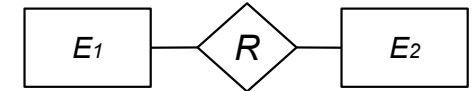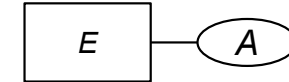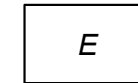
ER diagrams: rules

- An interconnecting line is only allowed between:
  - a box and a diamond,
  - a box and an oval,
  - a diamond and a oval.
- An oval must have exactly one connecting line.
- Names of boxes must be unique in the diagram.
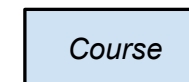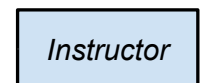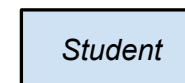- Names of ovals must be unique per box/diamond.

A first example

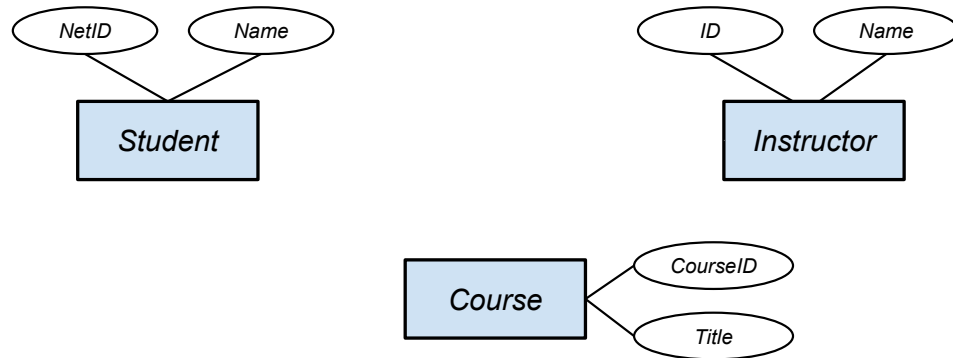Let's model a simple course registration system:
- **Students**
- **Instructors**
- **Courses**

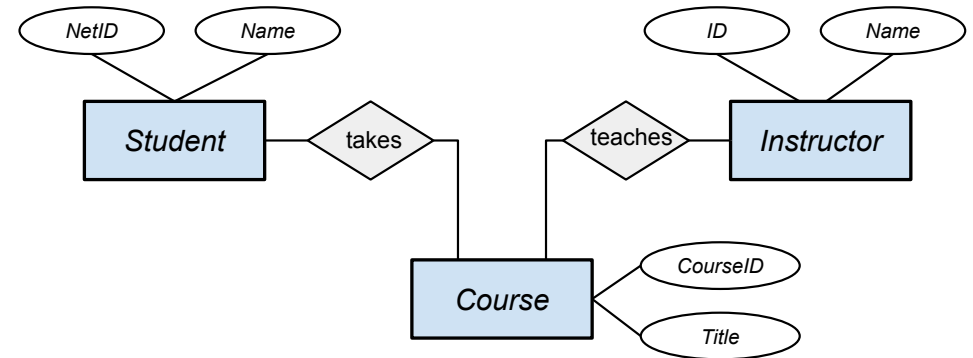## A first example: identify entities

Student

Instructor

Course

What attributes should we add?

A first example: identify attributes

NetID   Name

Student

ID   Name

Instructor

Course   CourseID   Title

What relationships should we add?

A first example: identify relationships

NetID   Name

Student — takes — teaches — **Instructor**

ID   Name

Course — CourseID   Title

ER diagrams: keys and cardinalities

- A **key** is an (underlined) attribute, or a set of attributes, which **uniquely identifies an entity**.

Student — *NetID*, FirstName, LastName

takes

Course — *CourseID*, Title

ER diagrams: keys and cardinalities

- A key is an (underlined) attribute, or a set of attributes, which uniquely identifies an entity.
- A key can be **artificial** or **natural**.

Student — *NetId*, *FirstName*, *LastName*

takes

Course — *CourseID*, Title
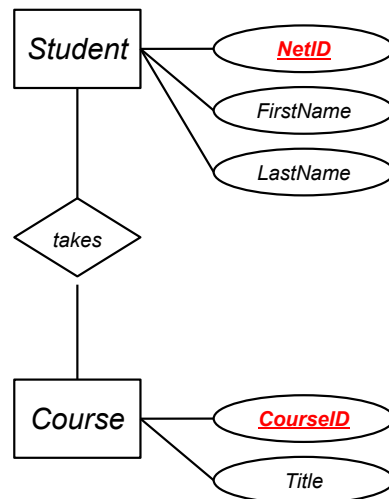
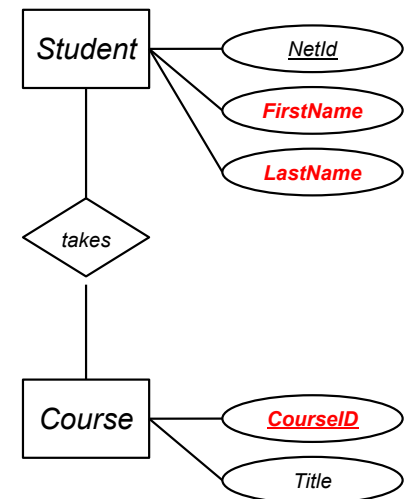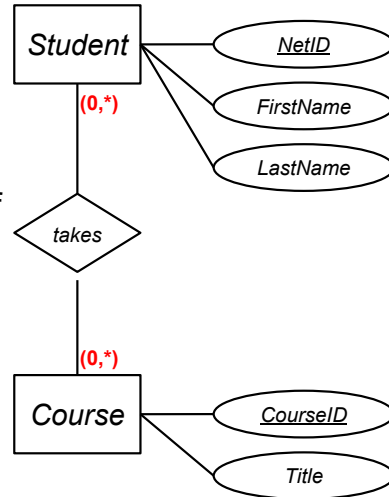## ER diagrams: keys and cardinalities

- A key is an (underlined) attribute, or a set of attributes, which uniquely identifies an entity.
- A key can be artificial or natural.
- The cardinalities define the kind of relationship (one-to-one, one-to-many, or many-to-many).
- There are different notations for cardinalities. For example:
 - 1 = (1,1)
  - c = (0,1)
  - m = (1,*)
  - mc = (0,*)

Student — *NetID*, *FirstName*, *LastName*
(0,*)
takes
(0,*)
Course — *CourseID*, *Title*

ER diagrams: weak entities

- A weak entity can't exist on its own (if a building is torn down, its rooms disappear).

Building — *BuildingNo*
has
Room — *RoomNo*
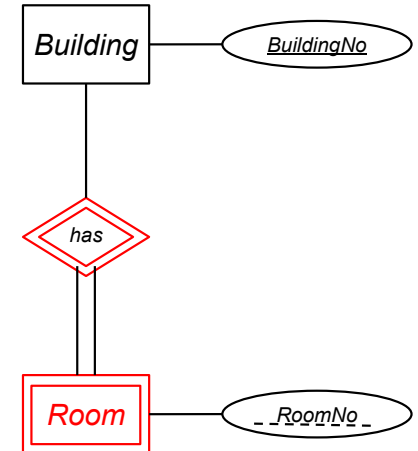
ER diagrams: weak entities

- A weak entity can't exist on its own (if a building is torn down, its rooms disappear).
- A weak entity is only uniquely identifiable in reference to another entity.

Building — *BuildingNo*
has
Room — *RoomNo*
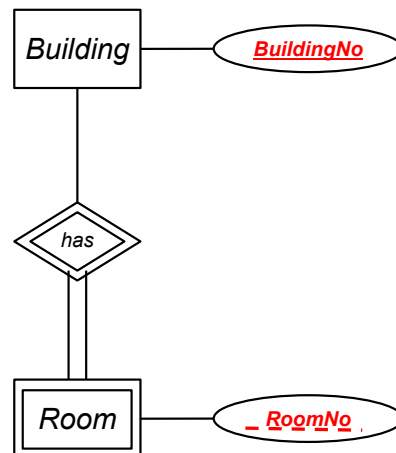
ER diagrams: self references and roles

- A self reference is usually explicitly annotated with roles to clarify the meaning of the self-referencing relationship.

Employee — manages / is managed by ◇
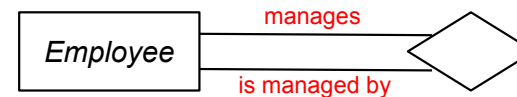
Think about (but never draw) the following:

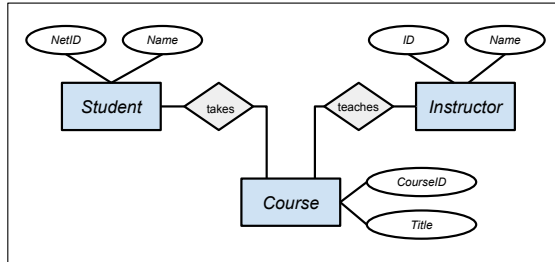**Employee** — manages ◇ is managed by — **Employee**
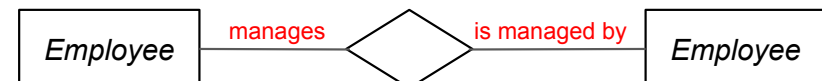
Putting it all together

Let's **augment** our **model** of a course registration system:
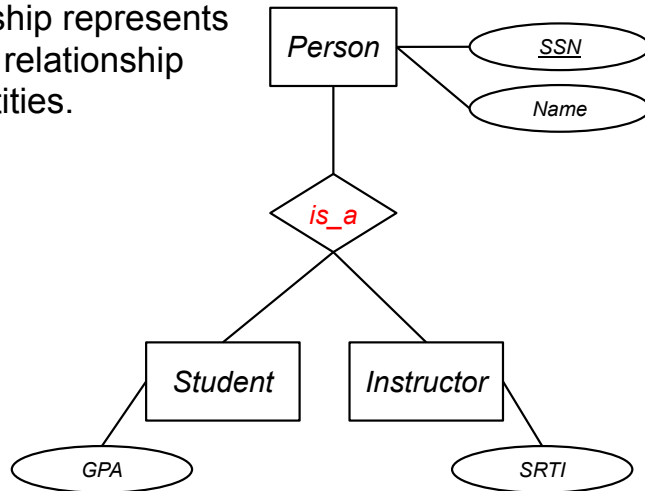
- Prerequisites
- Assignments
- Points/grades

Instructions

https://docs.google.com/presentation/d/1iUWnrOAwhrzkwPfp5AGXFdKUAFYSq-0oA-L9cbDyeCs/edit

---

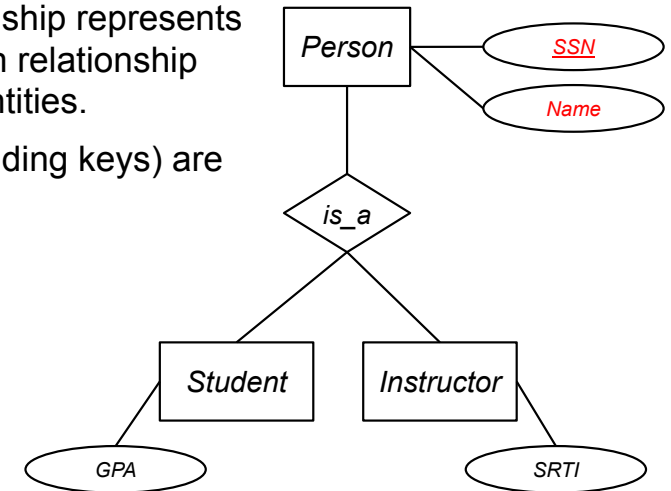**Additional material, not discussed in class**

ER diagrams: generalization

- An is_a relationship represents a generalization relationship between two entities.

ER diagrams: generalization

- An is_a relationship represents a generalization relationship between two entities.
- Attributes (including keys) are "inherited".

ER diagrams: generalization

- An is_a relationship represents a generalization relationship between two entities.

- Attributes (including keys) are "inherited".

- Additional attributes can be defined.