course: Data Management
part: Entity-Relationship modeling

Jan Van den Bussche
UHasselt

Intro: Designing a database

- We want to design a database about a well-defined enterprise
  - company, university, hospital, …
  - e-government, research project, social network, …
- Two main steps:
  1. Conceptual design:
     - Define precisely **which** information you want; we cannot and do not want to store everything!
  2. Logical design:
     - **How** will this information be represented in the actual database system?
- Interestingly, we will see that once we have a good conceptual design, it can be **automatically** converted in a good logical design.

Data models

- A data model (not to be confused with “models” in statistics, e.g., Gaussian, Bayesian) is a logical formalism to describe information, data
- Logical formalisms allow unambiguous descriptions
- To do conceptual design we use a conceptual data model
  - we will work here with the **Entity-Relationship (ER)** model
  - many more are used, e.g., UML , ORM
- To do logical design we use a logical data model
  - the data model used by an actual database system
  - we will work here with the **relational** data model, used by so-called relational database systems
  - many more are used, e.g., XML, JSON, RDF, property graphs

ER modeling at a glance

- Which types of **entities** do we want to "live" in our database?
  - what are the **attributes** of an entity type?
  - which attributes form the **key**?
- How are entities **related**?
  - **relationship** types
  - relationships may have **functionality** restrictions
  - relationships also can have attributes
- Describing all entity types + attributes + keys, and relationship types + functionality + attributes, comprises the ER design, called an ER **schema**.
- ER schemas have a visual, graphical notation

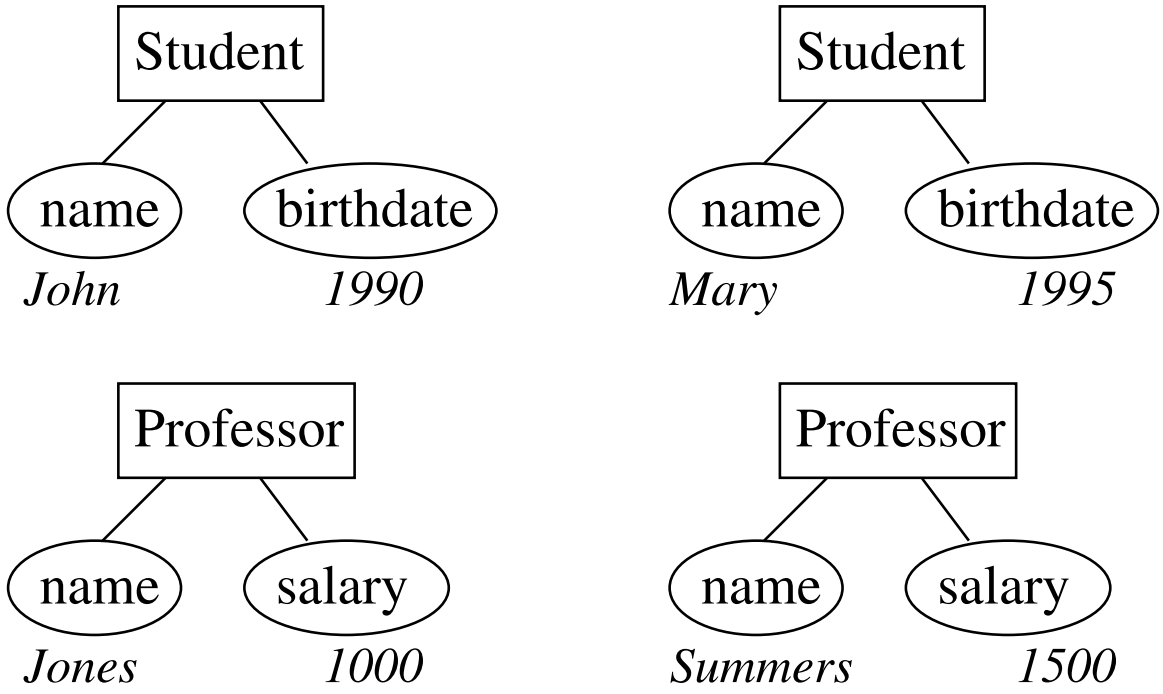
The ER instance

- A graph structure that represents the content of the database at a particular instant
- Purely **conceptual** way to think about what the database contains
- In the logical design phase, we will figure out a way to really store it in the computer
- A “world” of **entities** that are **related** to each other

Entity and attribute nodes

- Rectangular shape
- Have a **type** (a label written in the rectangle)
- Are linked to **attribute** nodes
  - Oval shape
  - represent actual values
  - also have a type

Example: university database



A very small ER instance consisting of two students and two professors

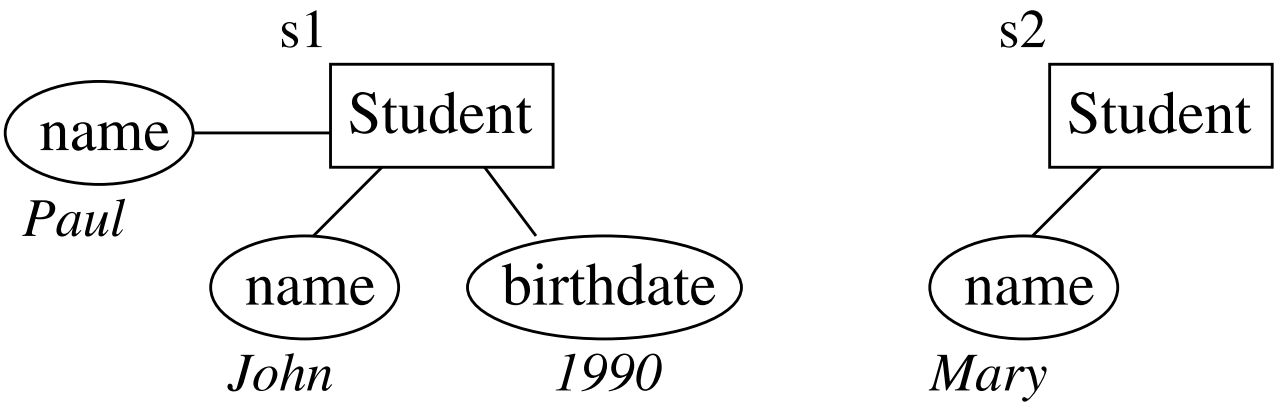
Well-formedness requirements

- We expect every instance to be well-formed, i.e., satisfy the following constraints:

Single attribute value: each entity node never has more than one attribute value of a given type

Uniform attributes: entity nodes of the same type, have the same types of attributes

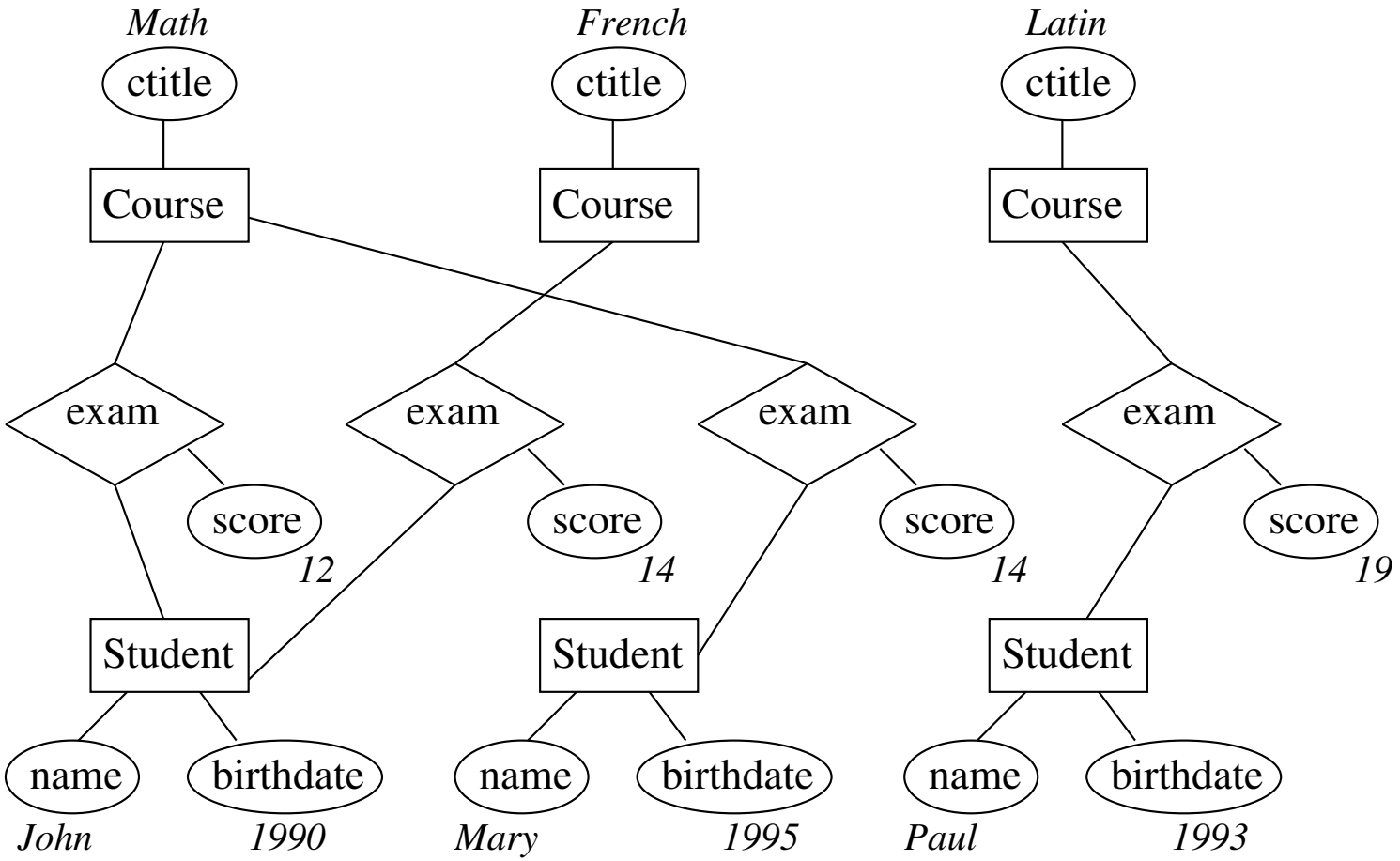
Example: not well-formed!



s1 has two name values. Moreover,
s1 has birthdate but s2 does not.

Relationships

- Entities are linked through relationships
- Relationship nodes
  - diamond-shaped
  - have a type
  - linked to several entities
  - can have attributes just like entities



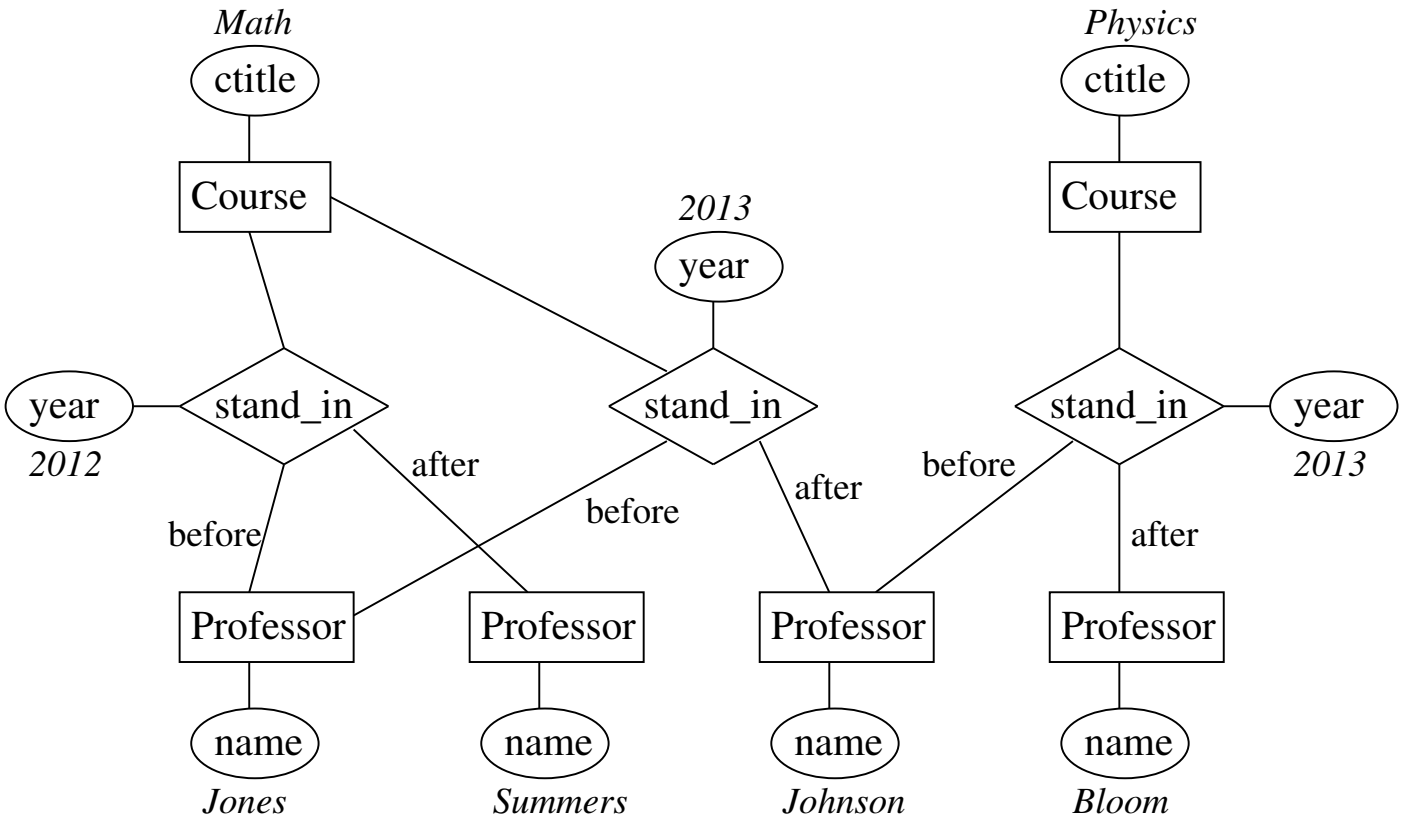
This ER instance contains some student and course entities, and **exam** relationships that indicate how students did for the courses.

Non-binary relationships

- Often relationships are binary, i.e., linked to two entities
- But other arities are possible
- E.g.:
  - assume entities professors, courses, rooms
  - relationships of type “teaches”
    - link a professor, a course, and a room

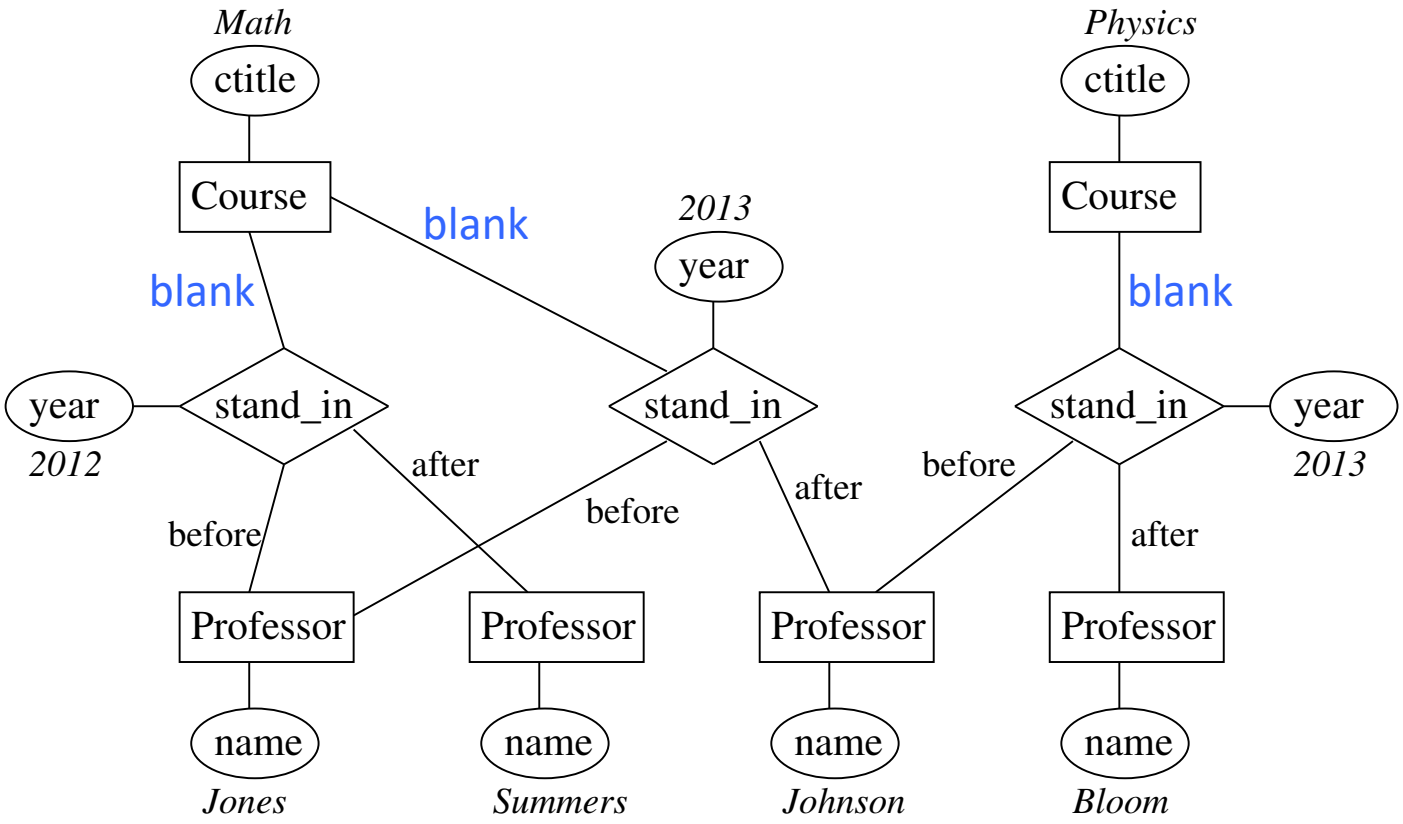
Relationship roles

- When a relationship has more than one link to entities of the same type
- Use explicit role indicators on the links, for example:



Relationship roles

- When a relationship has more than one link to entities of the same type
- Use explicit role indicators on the links, for example:



More well-formedness requirements

Single attribute value, uniform attributes:

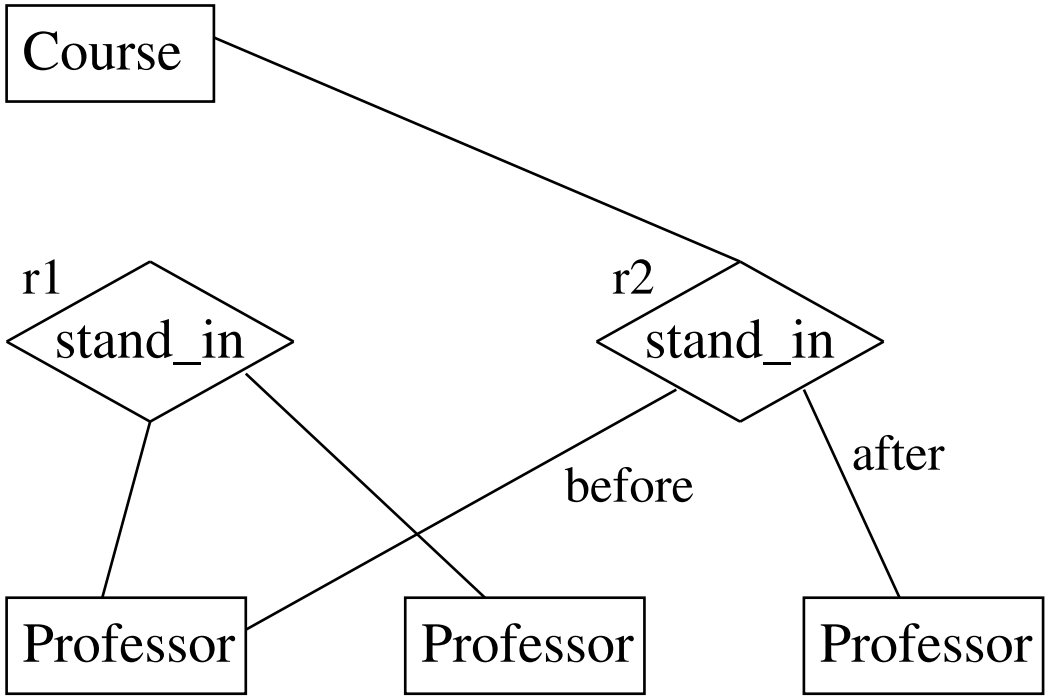
must hold for relationships as well as for entities

Distinguishing roles: two links from one relationship to entities with the same type, must have different roles

Uniform roles: relationships of the same type must have similar links to similar entity types

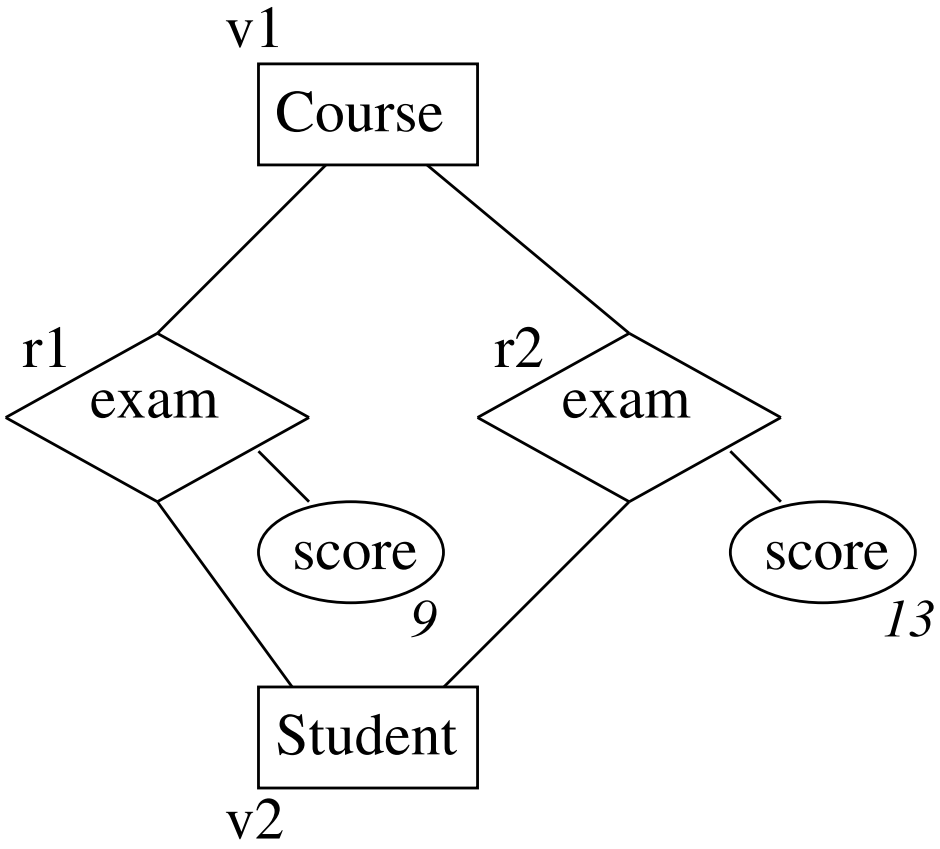
Unique relationship: two relationship nodes must not represent exactly the same relationship

Example not well-formed



- r1 violates “Distinguishing roles” (two links should have different roles)
- r2 violates “Uniform roles” (r2 links to a Course, but r1 does not)

Example Unique Relationship

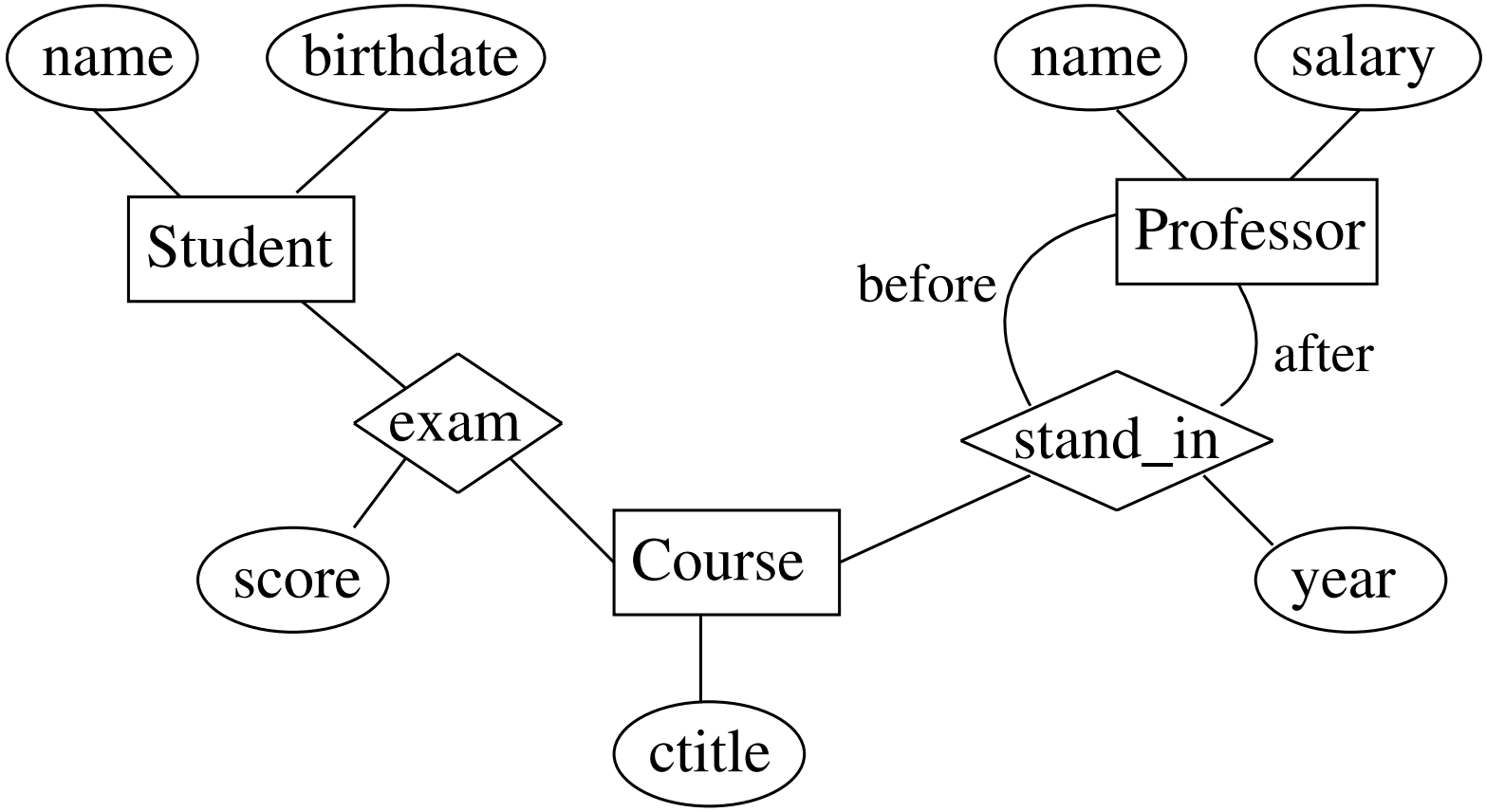


Violation: r1 and r2 represent exactly the same relationship (same course, same student)

Entity-Relationship schemas

- So far we have looked at **instances** as conceptual descriptions of database contents
- Of course the instance may change all the time (a database can be updated)
- A **schema** neatly summarizes the instances we want to allow for our database:
  - List allowed entity types, attributes
  - List allowed relationship types, participating entity types, roles, and attributes
  - Specify a **key** for each entity types
  - Optionally specify **functionality** constraints on relationship types
- Like instances, schemas can be shown as graphs

Example of an ER schema

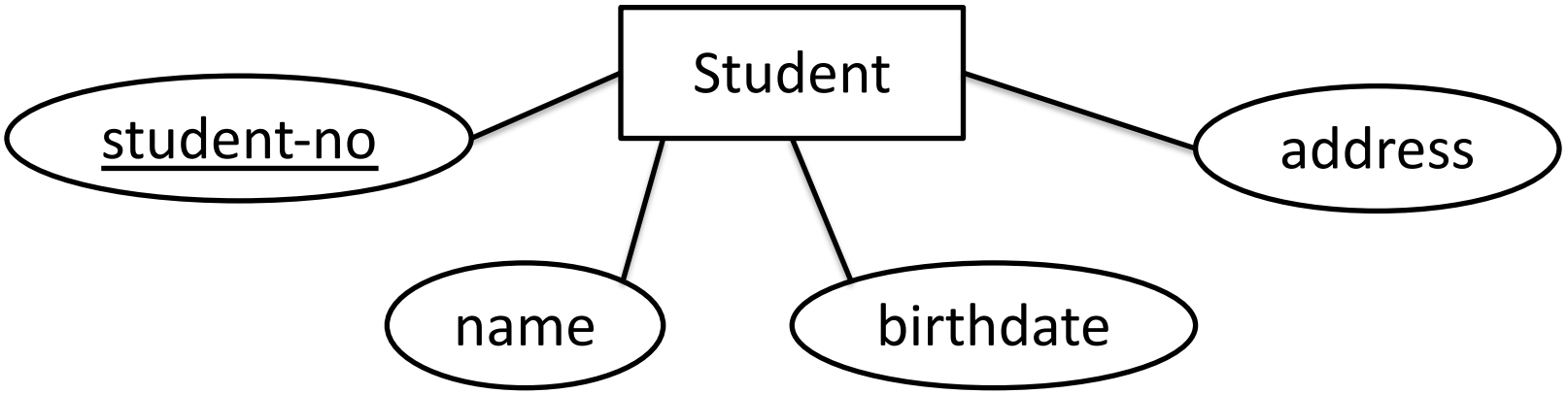


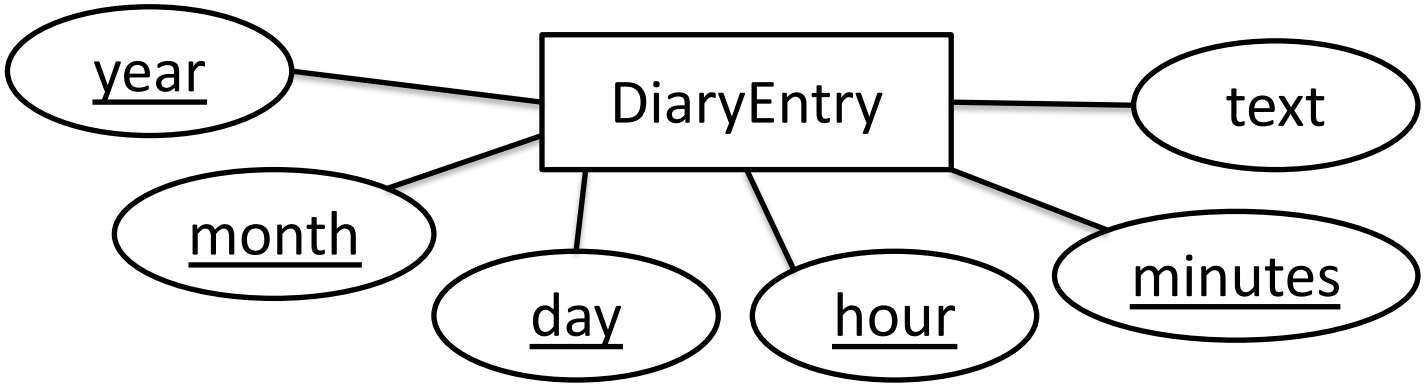
(keys are still missing)

Keys

- For each entity type in the schema, we must indicate its **key**
  - a subset of the attributes
  - indicated by <u>underlining</u>
- This means we do **not** allow instances where there are two entities that agree on the values of all key attributes

Keys: examples



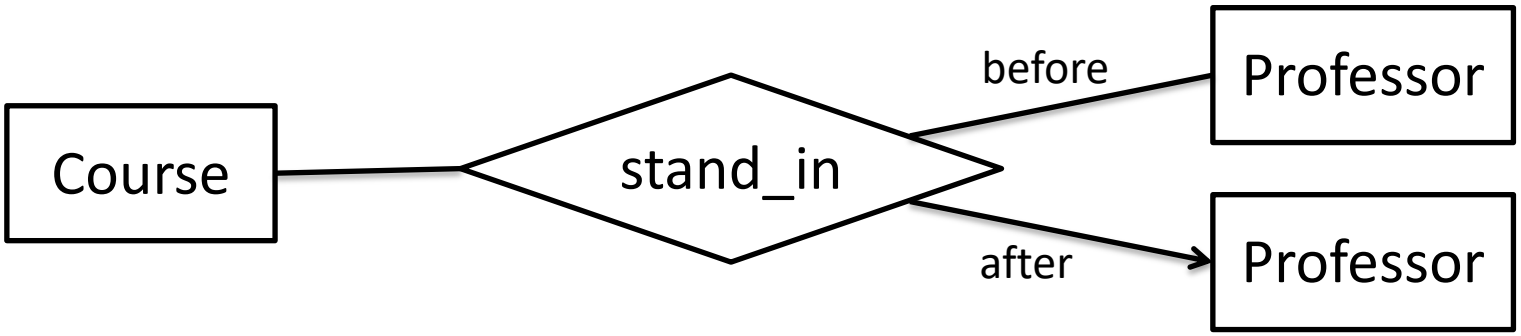


Note: it may be reasonable to designate the set of all attributes as the key, e.g., DateTime: year, month, day, hour, minutes

Functional relationship roles

- We may indicate some of the roles of a relationship type to be **functional**
  - indicated with an **arrow-head**
- Indicating a role to be functional means we **disallow** two different relationships (of that type) in the instance that agree on all other roles (so only differ in the functional role).

Example functional role



Means we don't allow that, for the same course, a professor has more than one stand-in.

- What would the following mean?