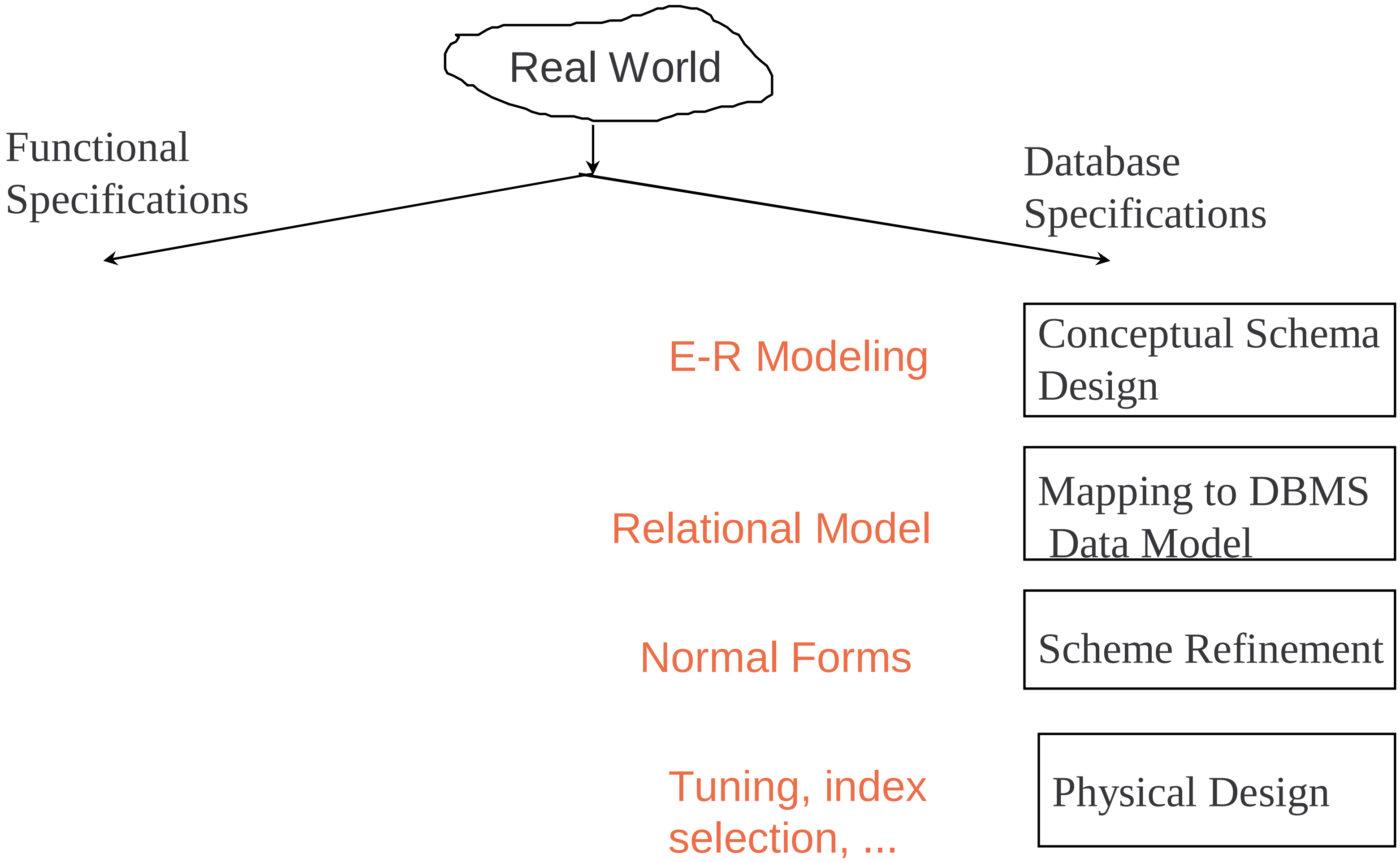
The Entity-Relationship Model

Jörg Sander

*Disclaimer: The slides used in the course may contain some of the slides provided by the authors of the adopted textbook (present and past) and those used in previous editions of this course by other instructors.

Database Design Process



Real World

Functional Specifications

Database Specifications

E-R Modeling

Conceptual Schema Design

Relational Model

Mapping to DBMS Data Model

Normal Forms

Scheme Refinement

Tuning, index selection, ...

Physical Design

ER Model Overview

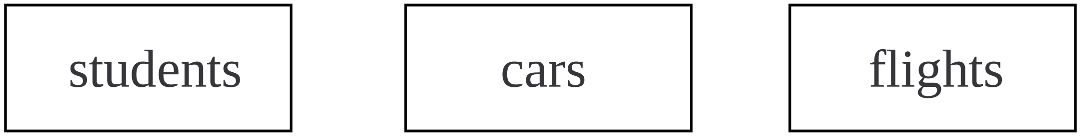
- Developed by Peter Chen in the mid 70’s
- Used for the design of conceptual schema.

- The “world” is described in terms of
  - entities
  - relationships
  - attributes

- The model is visualized by creating an ER diagram.

ER Model Basics

- **Entity:** a distinguishable object
  - e.g. person, thing, concept
- **Entity set:** a set of entities of the same type.
- Examples of entity sets:
  - students registered at UofA
  - cars currently registered in Alberta
  - flights offered by Air Canada
- Graphical representation:



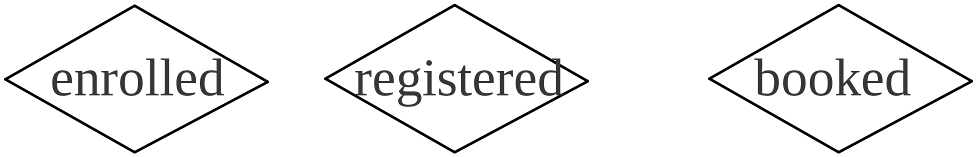
students

cars

flights

ER Model Basics

- **Relationship:** represents the fact that certain entities are related to each other.
  - e.g. John has taken CMPUT 291.
- **Relationship set:** set of relationships of the same type.
- Examples of relationship sets:
  - students enrolled in courses
  - cars registered to owners
  - passengers booked on flights
- Graphical representation:



enrolled registered booked

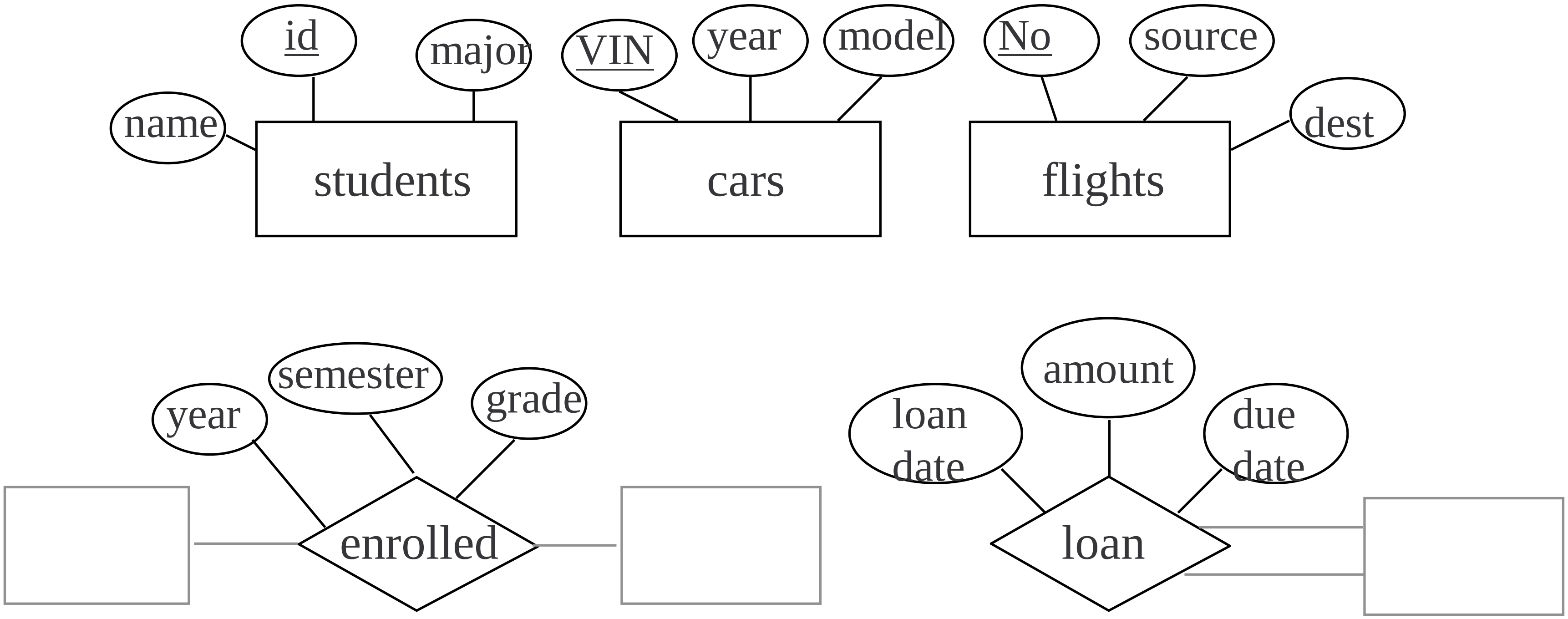
ER Model Basics

- **Attribute:** describes a property of an entity or a relationship.
- Attributes of entities and relationships - examples
  - student: id, name, major, …
  - flight: No, source, destination, …
  - loan: when (was it loaned), until when (is it loaned), …

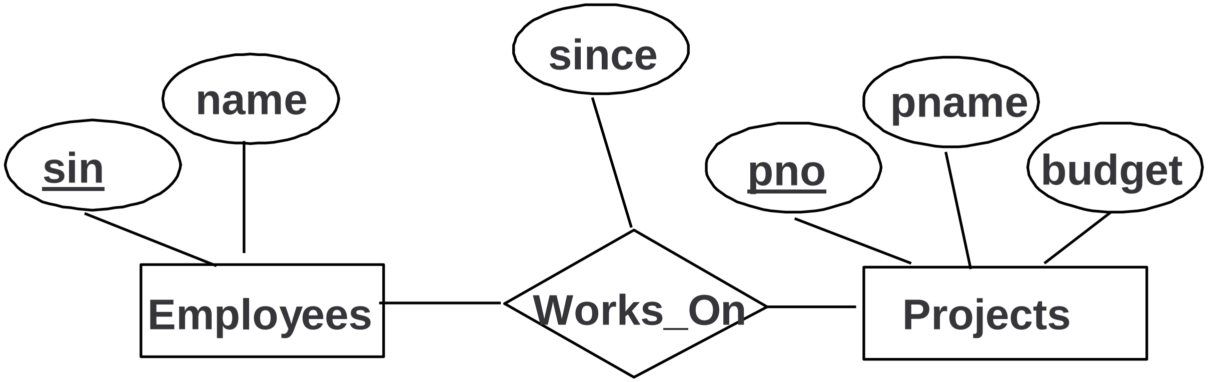
- **Key**: a minimal set of attributes that uniquely identifies each entity in an entity set.
  - Relationships cannot have keys

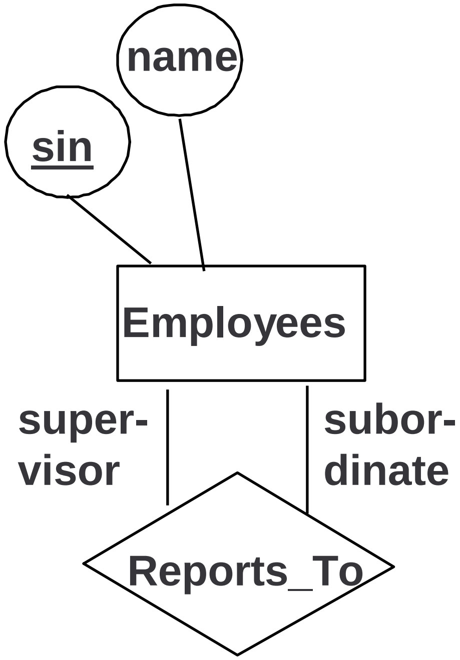
ER Model Basics

- Graphical representation:



Examples

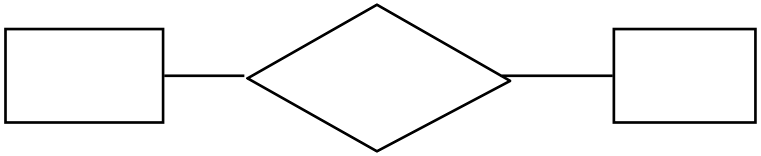




- Role: the function of an entity set in a relationship set.
- Role labels are needed whenever an entity set has multiple functions in a relationship set.

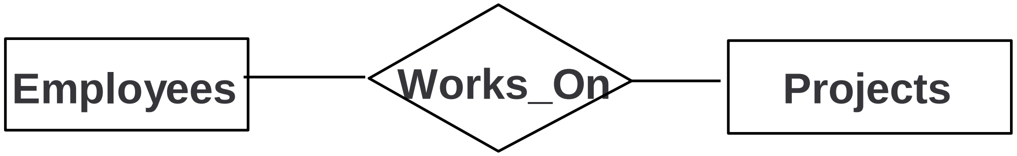
Constraints and Complications

- Key constraints
  - in binary relationships: binary relationship types
  - in general relationships
- Participation constraints
- Set-valued attributes
- Weak entities
- ISA hierarchies



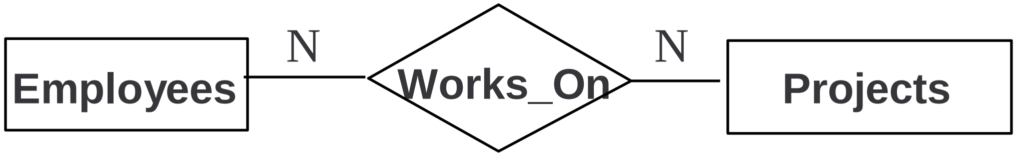
Binary Relationship Types: Many-to-Many

- Constraint: none.



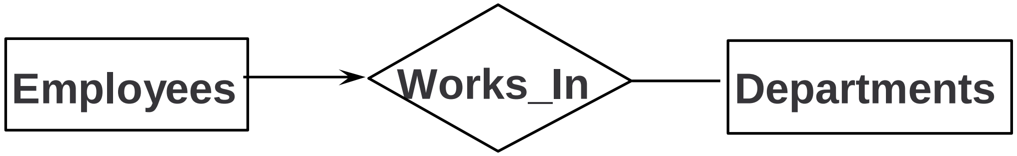
- Each employee can be in relationships with many projects and vice versa.

- Alternative representation (just FYI!)



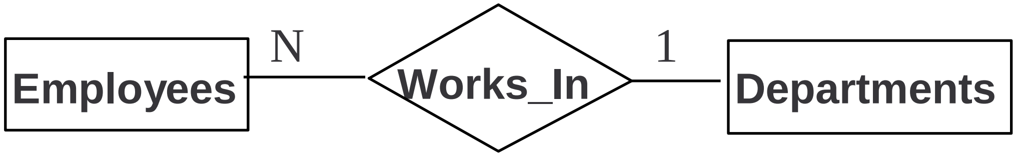
Binary Relationship Types: Many-to-One

- Constraint: each employee works in at most one department.



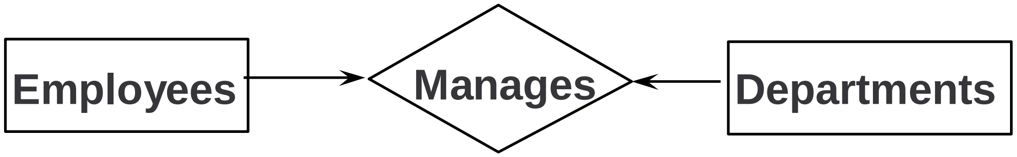
  - Given an employee, we can uniquely identify the department he/she works in.

  - Alternative representation:

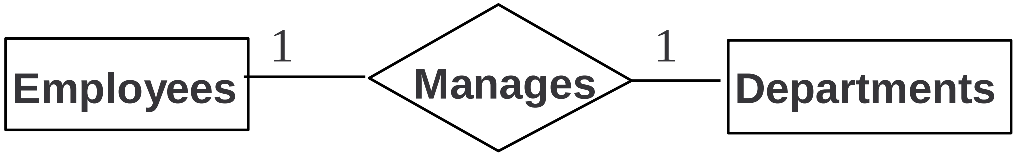


Binary Relationship Types: One-to-One

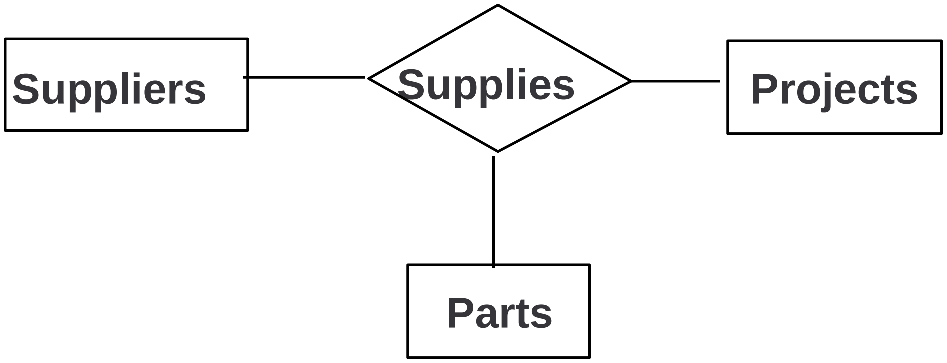
- Constraint: each employee can manage at most one department and each department is managed by at most one employee.



- Each employee can be in relationship with at most one department and vice versa.
- Alternative representation:



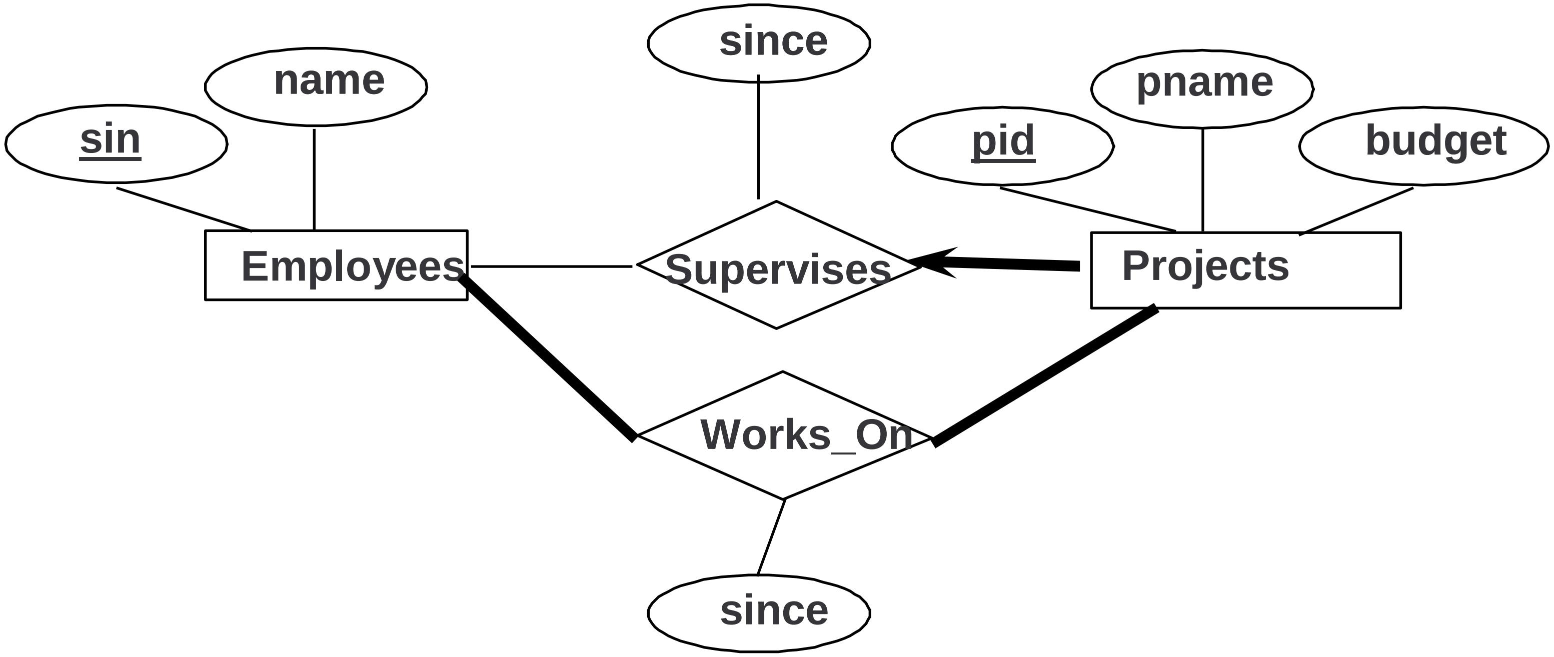
Ternary Relationships



- Meaning: Supplier **s** supplies part **p** for project **r**.
- Complication: add the Constraint “each part is supplied by a unique supplier for a unique project,”
  - i.e. each part is in relationship with at most one supplier and one project.

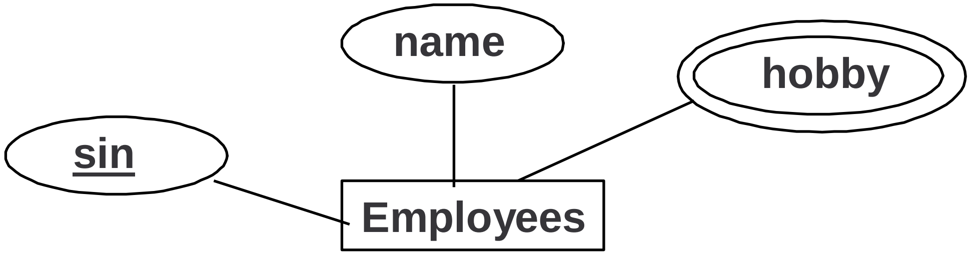
Participation Constraints

- Does every project have a supervisor?
  - If so, this is a *participation constraint*: the participation of Projects in Supervises is said to be *total* (vs. *partial*).
    - ✓ Every *pid* value in Projects table must appear in a row of the Supervises table (with a non-null *sin* value!)



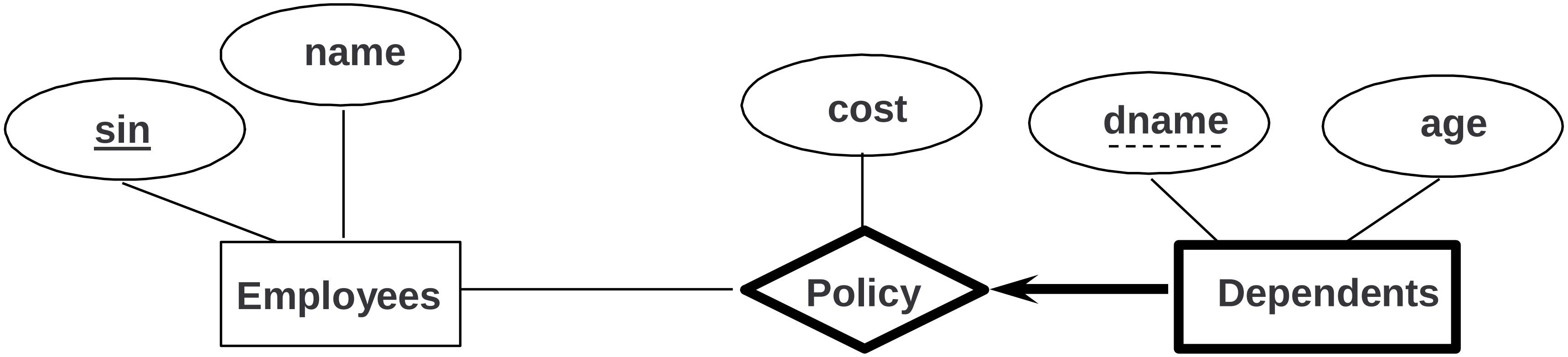
Set-Valued Attributes

- Each employee can have one or more hobby.
  - Attribute value can be a set (in contrast to the relational model as we shall see later)
  - E.g. (111111, John, (stamps, coins))



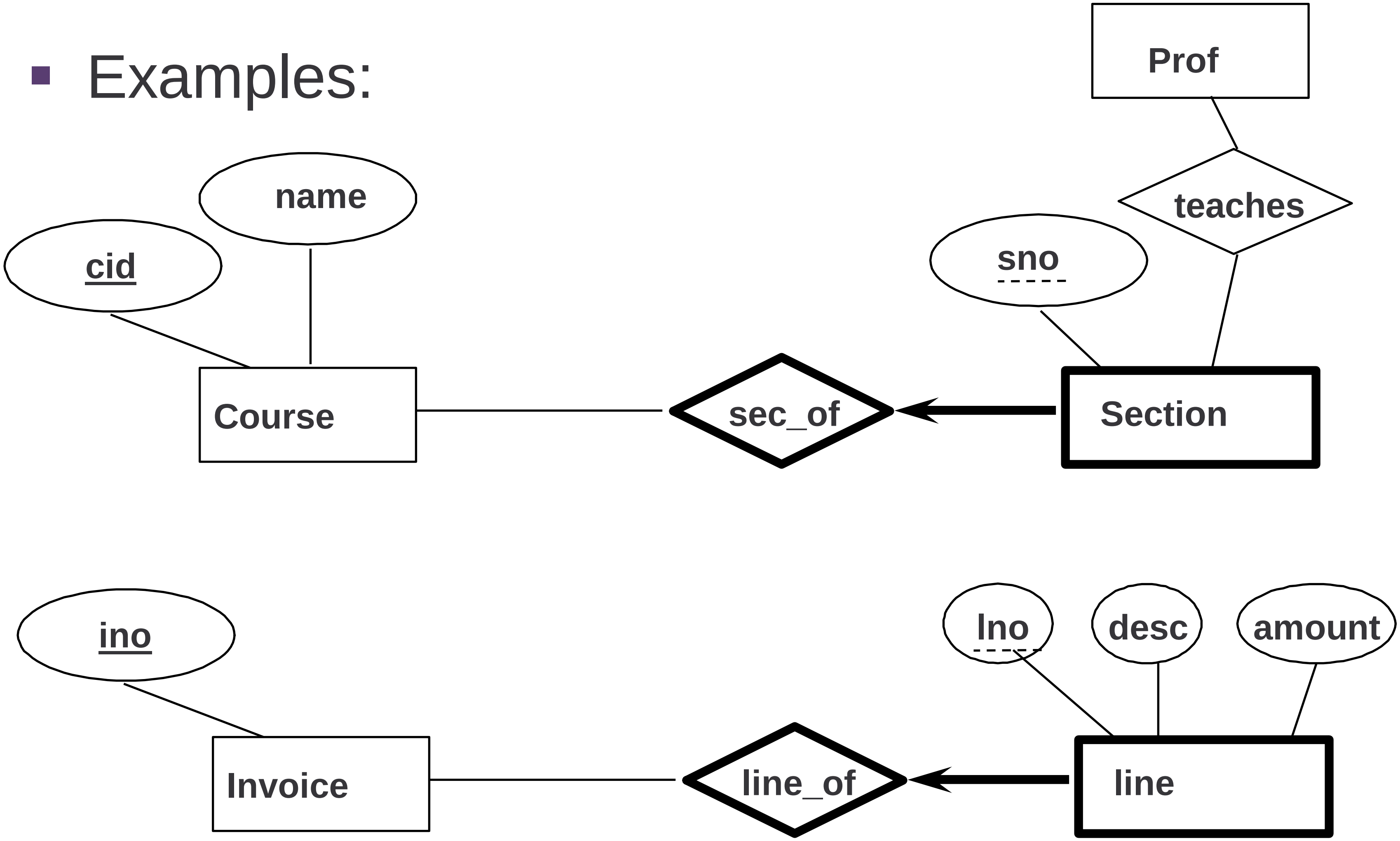
Weak Entities

- A *weak entity* can be identified uniquely only by considering the primary key of another (*owner*) entity.
  - Owner entity set and weak entity set must participate in a one-to-many relationship set (one owner, many weak entities).
  - Weak entity set must have total participation in this *identifying* relationship set.
  - Weak entities have a weak key (denoted by a dashed underline)



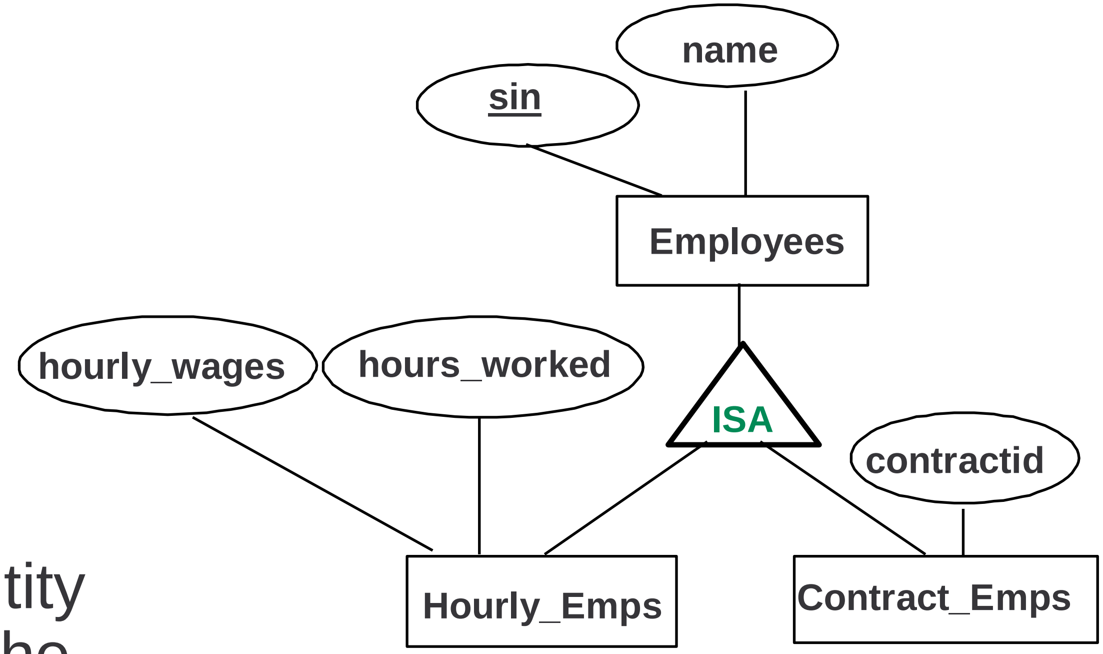
Weak Entities

- Examples:



Generalization

- forming a new entity set as the union of two or more entity sets.

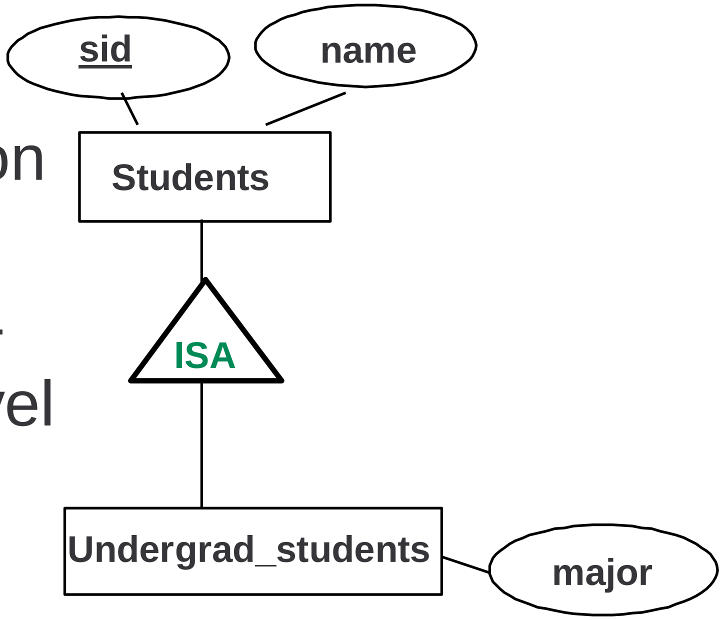


Contract_Emps and Hourly_Emps cover Employees.

- attributes common to all lower-level entity sets are moved to the higher-level entity set.

Specialization

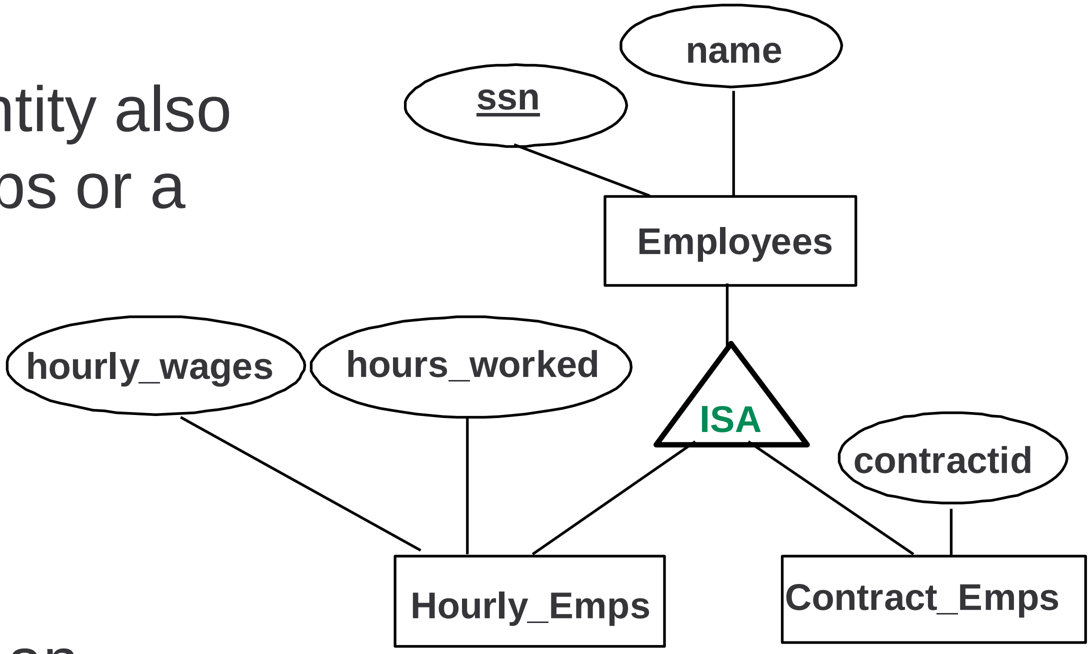
- forming a derived entity set by taking a subset of a given entity set.



- Differences between generalization and specialization:
  - in generalization, every higher-level entity must be a lower-level entity too.
  - this is not the case in specialization.

ISA Constraints

- *Covering constraints*: Does every Employees entity also have to be an Hourly_Emps or a Contract_Emps entity? *(default:no)*
- *Overlap constraints*: Can an employee be an Hourly_Emps as well as a Contract_Emps entity? (default:*disallowed*)



Contract_Emps and Hourly_Emps cover Employees.

Contract_Emps overlap Hourly_Emps.

ISA Hierarchies (Summary)

- Reasons for using ISA:
  - Makes ER diagram more concise and readable.
  - Common attributes/relationships need not be repeated.
- *Properties: inheritance, transitivity*
- *Constraints:*
  - *Covering constraints*: generalization vs specialization.
  - *Overlap constraints*

Conceptual Design Using the ER Model

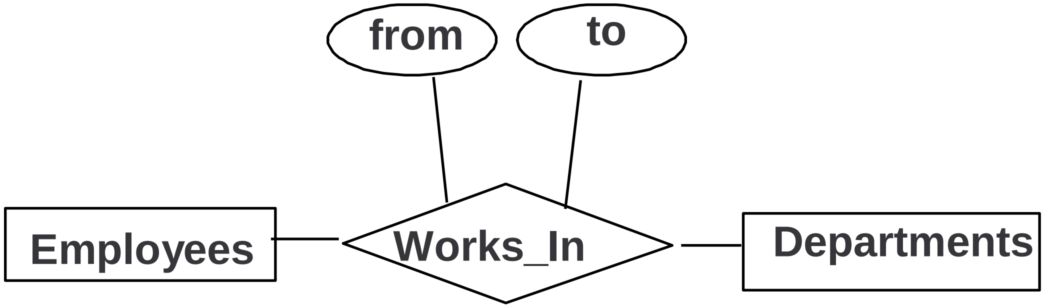
- Design choices:
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?
  - Identifying relationships: Binary or ternary?

Entity vs. Attribute

- Should *address* be an attribute of Employees or an entity (connected to Employees by a relationship)?
- Depends on the use we want to make of address information and the semantics of the data:
  - is it an object that we want to keep information about (independent from employees)?
  - does it participate in a relationship with an entity other than employees?
  - does it make sense to have employees with no addresses?
  - can several employees share the same address?
- A positive answer to one or more of those questions implies address better be modeled as an entity.

Entity vs. Attribute (Contd.)

- Can an employee work in the same dept for two or more periods?
  - first diagram: No
  - second diagram: Yes



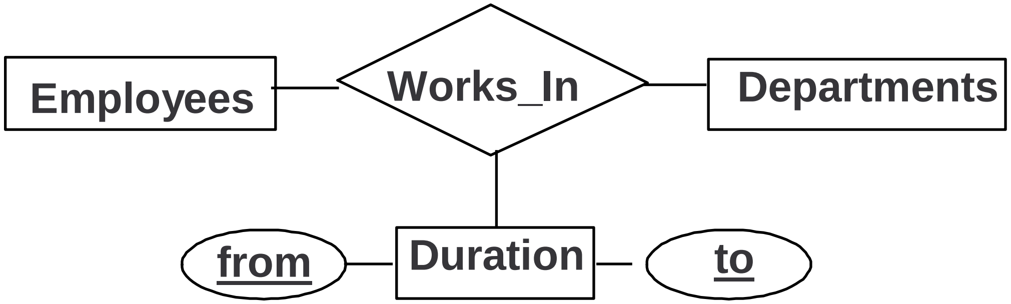
from

to

Employees

Works_In

Departments



Employees

Works_In

Departments

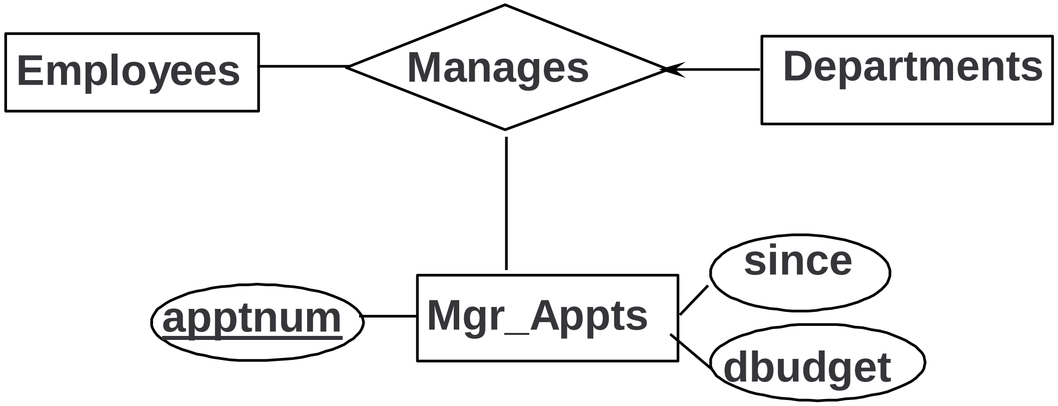
from

Duration

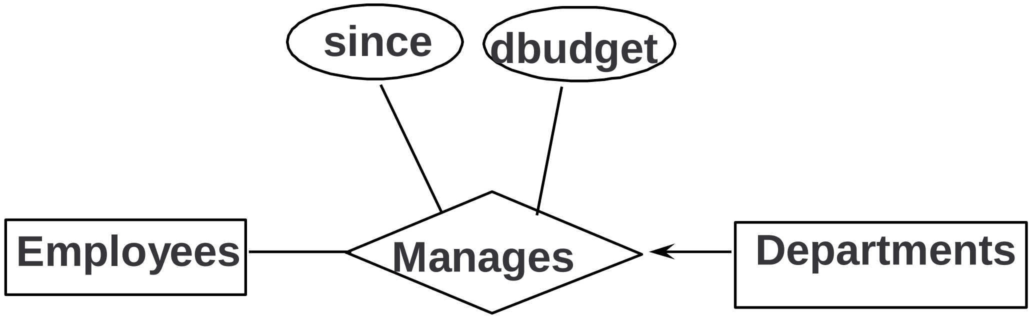
to

Entity vs. Relationship

- First ER diagram:
  - The appointment can be for more than one dept.
  - The budget can be also for more than one dept.

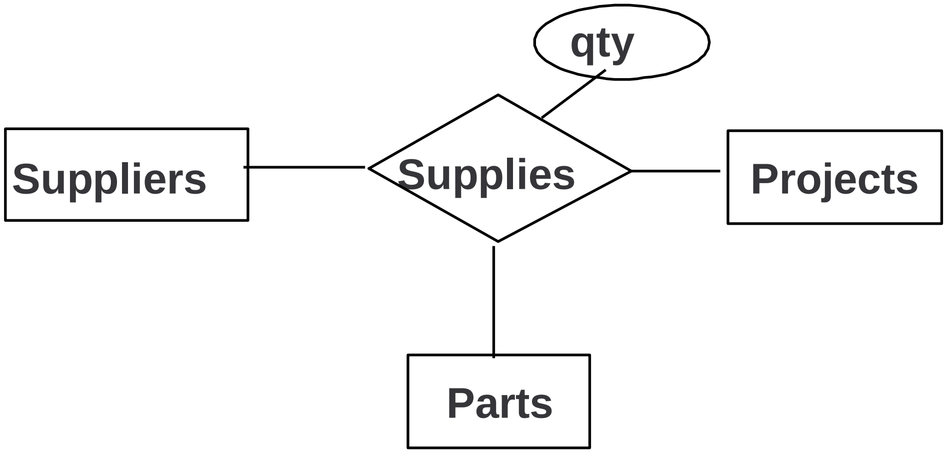


- Second ER diagram:
  - Ok if the dbudget is allocated to one specific employee for one particular dept.
  - Otherwise, dbudget is both redundant and misleading.

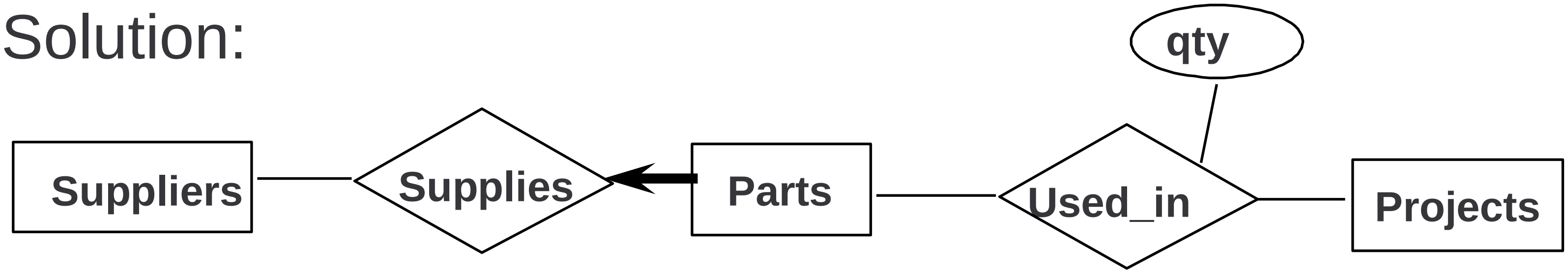


Binary vs. Ternary Relationships

- Add the constraint: each part is supplied by a unique supplier.
  - not possible!
  - a key constraint on Parts would also mean each part can only be used in a single project!



- Solution:



What is the additional constraint here?

Binary vs. Ternary Relationships (Contd.)

- Previous example: two binary relationships were better than one ternary relationship.

- Consider the same example without the constraint:
  - No combination of binary relationships is an adequate substitute:
    - ✓ supplier s “supplies” part p,
    - ✓ part p “used_in” project r, and
    - ✓ Supplier s “supplies_to” project r
  - They do not imply that part p supplied by supplier s is used in project r.
  - How do we record *qty*?

Summary of Conceptual Design

- *Conceptual design* follows *requirements analysis*,
  - Yields a high-level description of data to be stored
- ER model popular for conceptual design
  - Constructs are expressive, close to the way people think about their applications.
- Basic constructs: *entities*, *relationships*, and *attributes* (of entities and relationships).
- Some additional constructs: *weak entities* and *ISA hierarchies*.

Summary of ER (Contd.)

- Constraints play an important role in determining the best database design for an enterprise.
  - Several kinds of integrity constraints can be expressed in the ER model.
  - Some constraints (notably, *functional dependencies*) cannot be expressed in the ER model.
- ER design is *subjective*. There are often many ways to model a given scenario!

ER Exercise

- Professors have a SIN, a name, an age, a rank, and a research specialty.
- Projects have a project number, a sponsor name (e.g. NSERC), a starting date, an ending date, and a budget.
- Graduate students have a SIN, a name, an age, and a degree program (e.g. MS or PhD)
- Each project is managed by one professor (principal investigator).
- Each project is worked on by one or more professors (co-investigators).
- Professors can manage and/or work on multiple projects.
- Each project is worked on by one or more graduate students (research assistants).
- When graduate students work on a project, a professor must supervise their work on the project. Graduate students can work on multiple projects, in which case they will have a (potentially different) supervisor for each one.
- Departments have a department number, a department name, and a main office.
- Departments have a professor (chairman) who runs the department.
- Professors work in one or more departments, and for each department that they work in, a time percentage is associated with their job.

Design an ER diagram ...