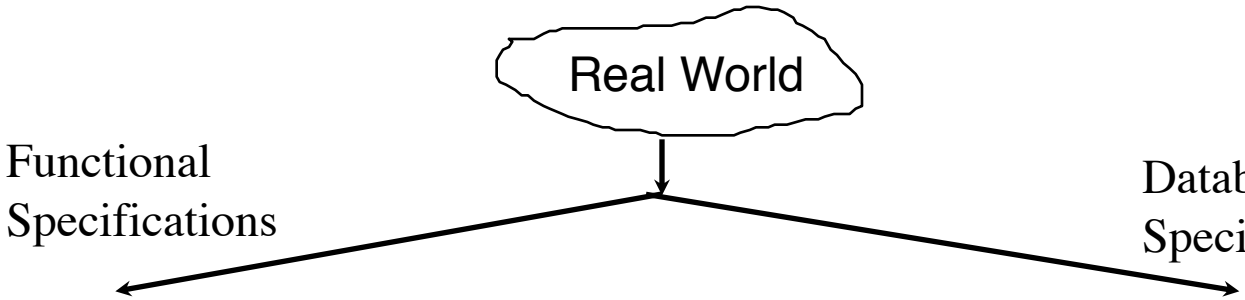
# The Entity-Relationship Model

Davood Rafiei



# Database Design Process




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

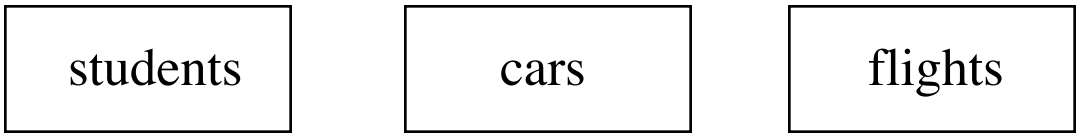
- Developed by Peter Chen in the mid 70's
- Used for the design of conceptual schema.

- 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:

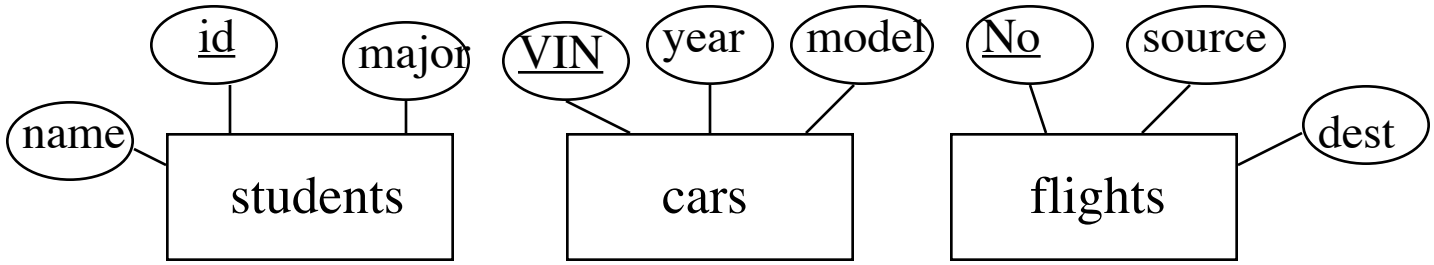



# ER Model Basics

- **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:

# ER Model Basics

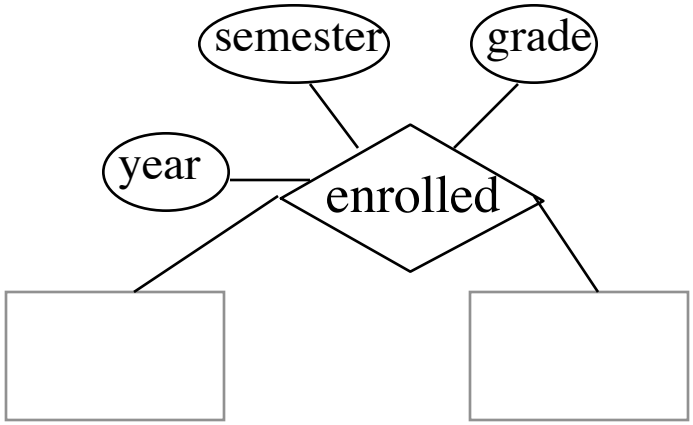
- **Attribute**: describes a property of an entity or a relationship.
- Attributes of entities - examples
  - student: id, name, major, …
  - car: VIN, year, model, …
  - flight: No, source, destination, …
- Graphical representation:

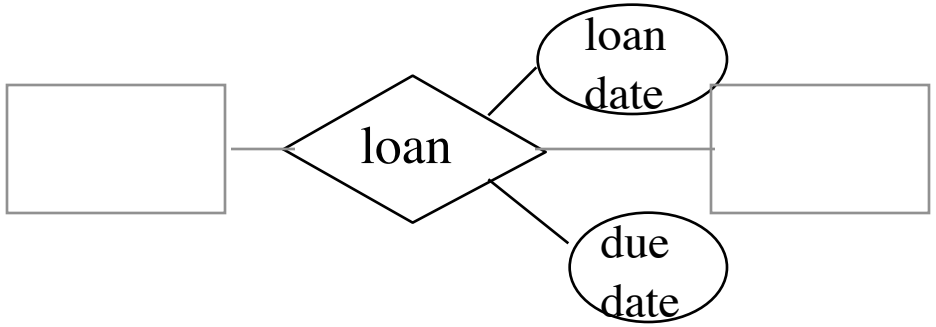


# ER Model Basics

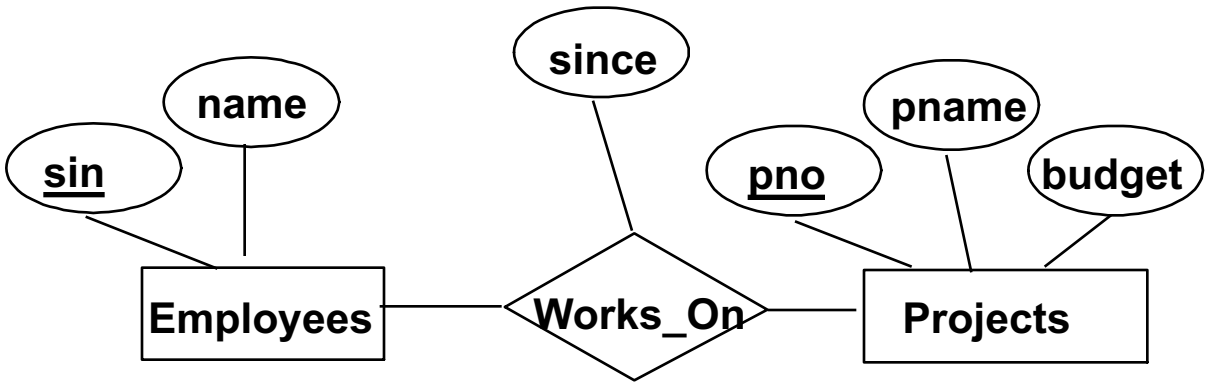
- **Key**: a minimal set of attributes that uniquely identifies each entity in an entity set.

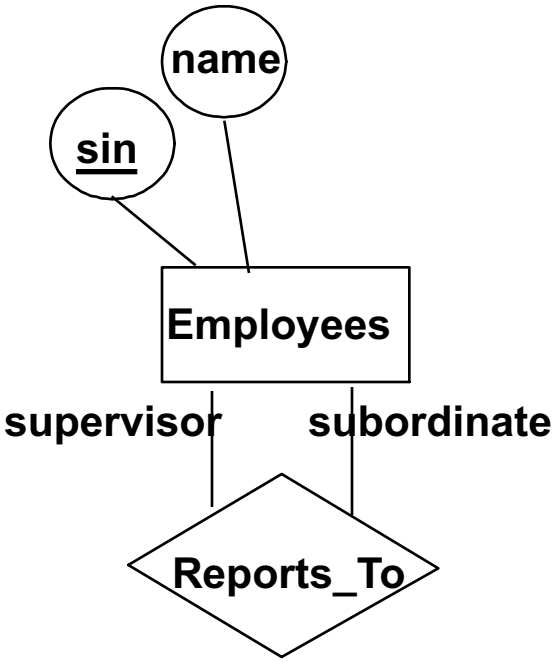
- Attributes of relationships - examples
  - student enrolled in a course: year, semester, grade
  - book on loan: loan date, due date



# Examples

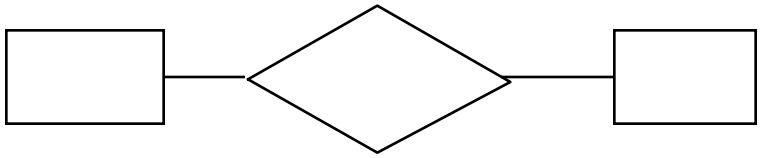



- Role: the function of an entity set in a relationship set.
- Role labels are needed whenever an entity set has multiple functions in a relationship set.

# Constraints and Complications

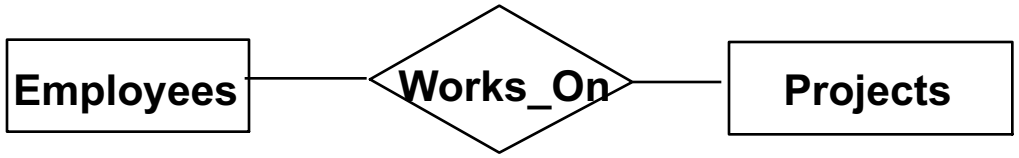
- Key constraints
  - in binary relationships: binary relationship types



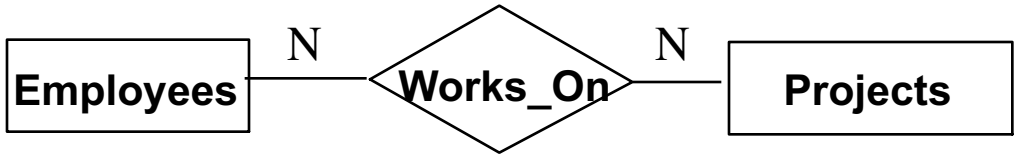
  - in general relationships
- Participation constraints
- Set-valued attributes
- Weak entities
- ISA hierarchies

# Binary Relationship Types: Many-to-Many

- Constraint: none.

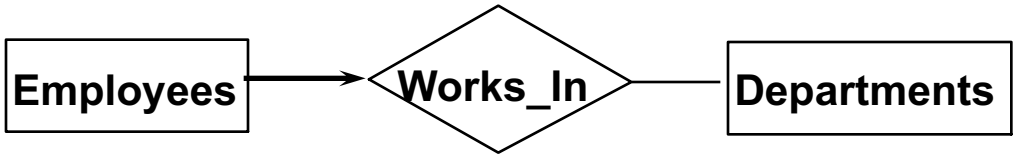



  - Each employee can be in relationships with many projects and vice versa.
  - Alternative representation



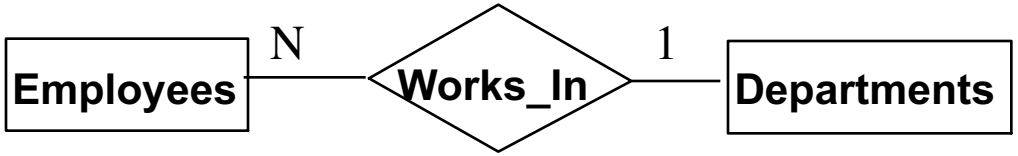
# Binary Relationship Types: Many-to-One

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




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

  - Alternative representation:



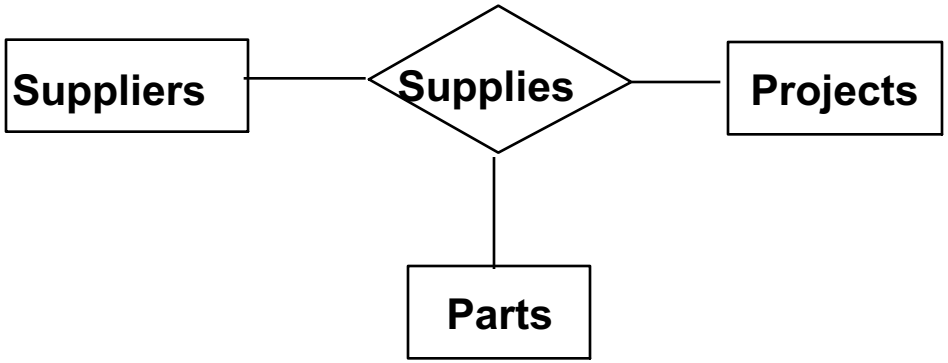
# Binary Relationship Types: One-to-One

- 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:

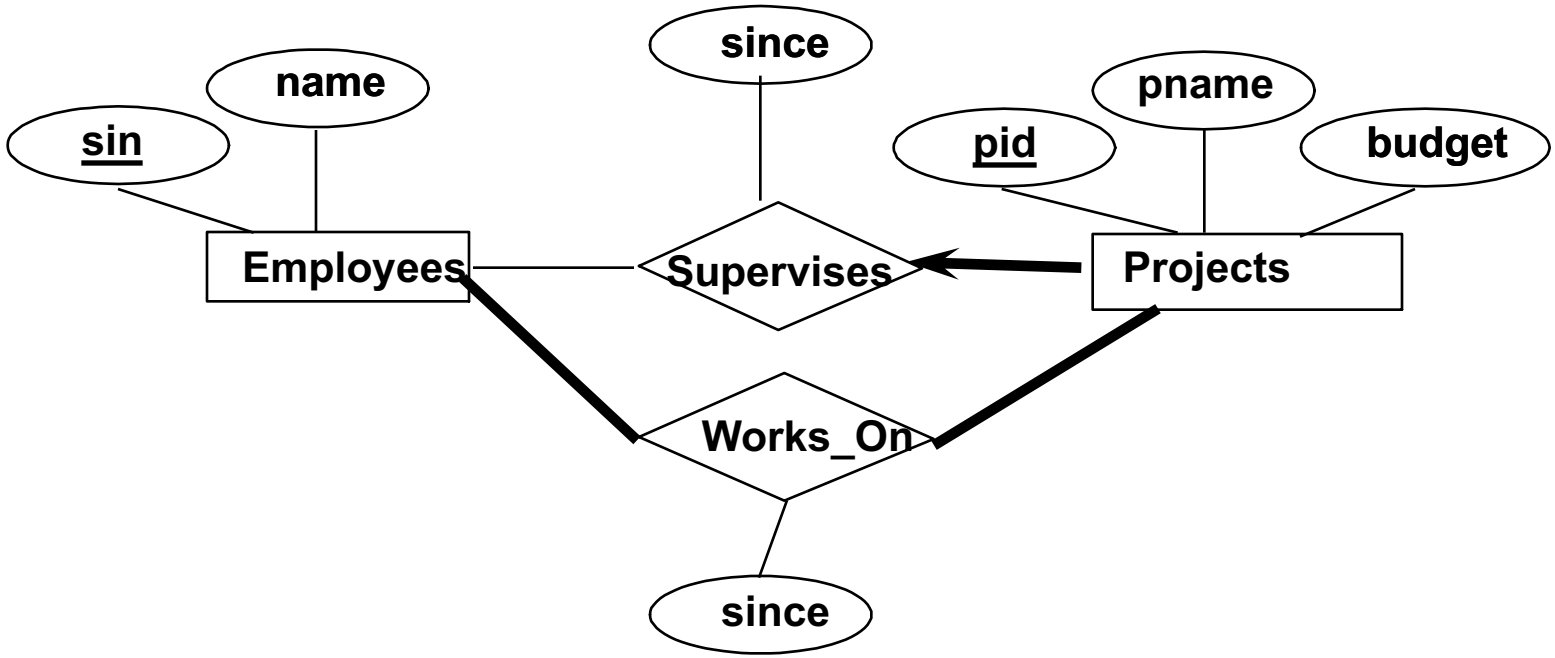
# Ternary Relationships




- Meaning: Supplier **s** supplies part **p** for project **r**
- Can we represent this using binary relationships?
- 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.

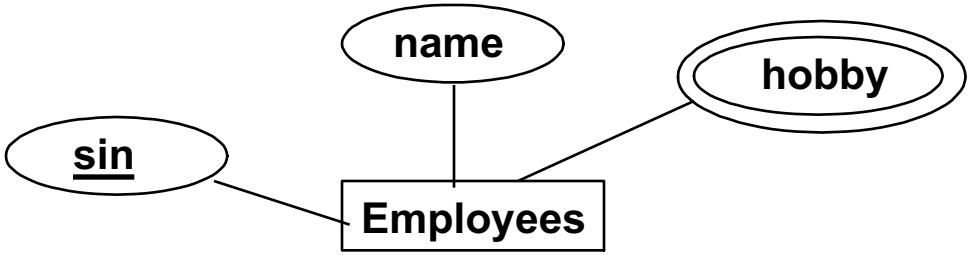
# Participation Constraints

- Let's force each project to have a supervisor.
  - This is a *participation constraint*: the participation of Projects in Supervises is said to be *total* (vs. *partial*).
    - ✓ Every *pid* value in Projects must be in a "supervises" relationship with a *sin* of an employee (i.e., *sin* cannot be null)



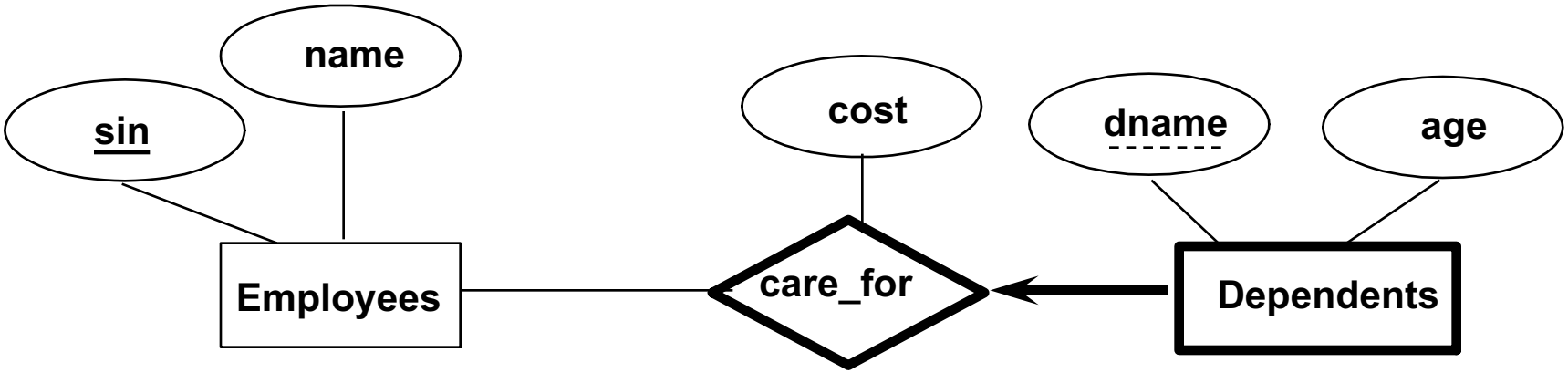
# Set-Valued Attributes

- Each employee can have one or more hobby.
  - Attribute value can be a set (in contrast to relational model)
  - E.g. (12345, Joe, (hockey, music))



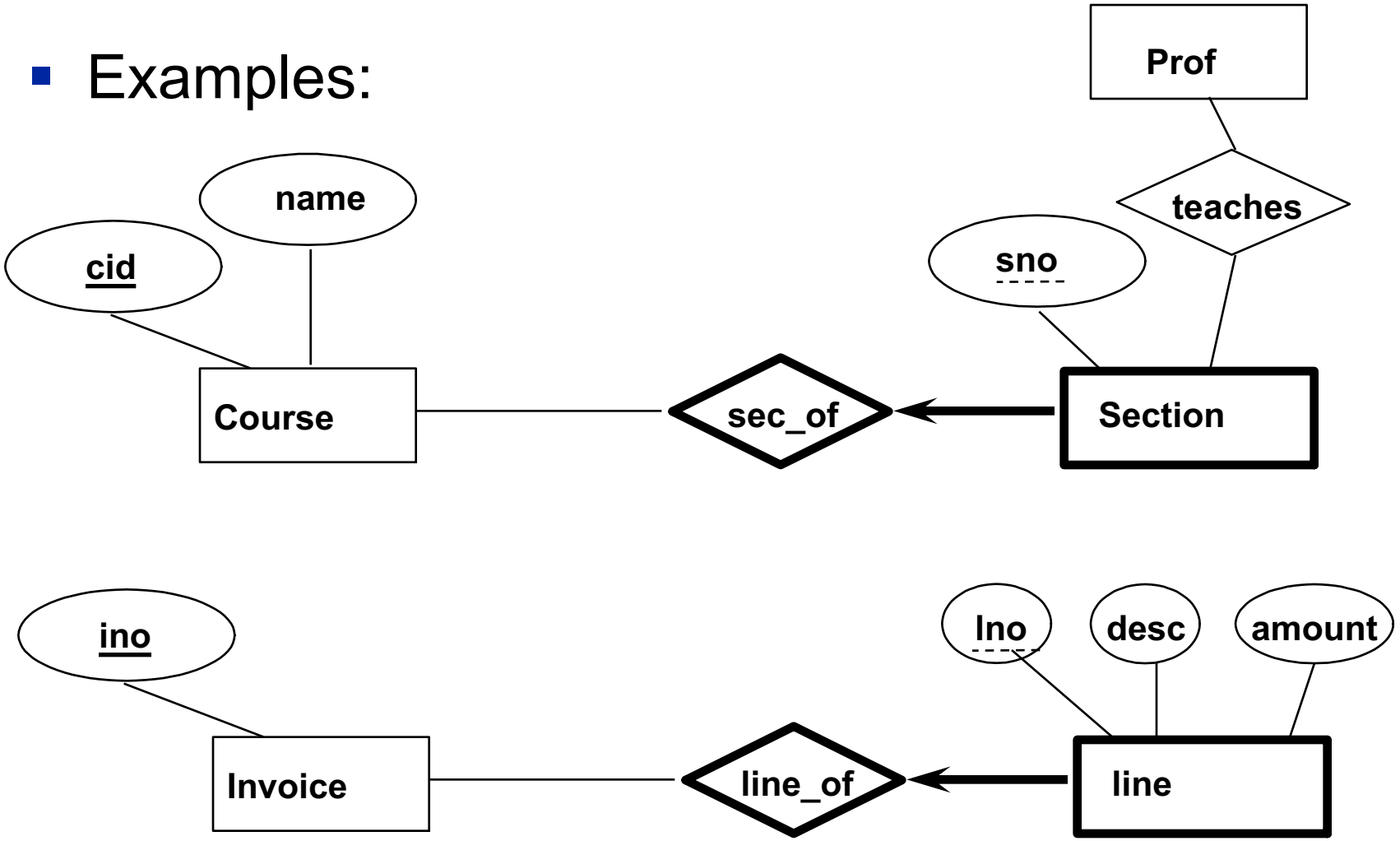
# Weak Entities

- A *weak entity* models an entity which is "part-of" an *owner entity*, and it cannot be uniquely identified without the primary key of the owner entity.
  - Relationship is one-to-many (one owner, many weak entities).
  - Weak entity set has total participation in the *identifying* relationship set.



# Weak Entities

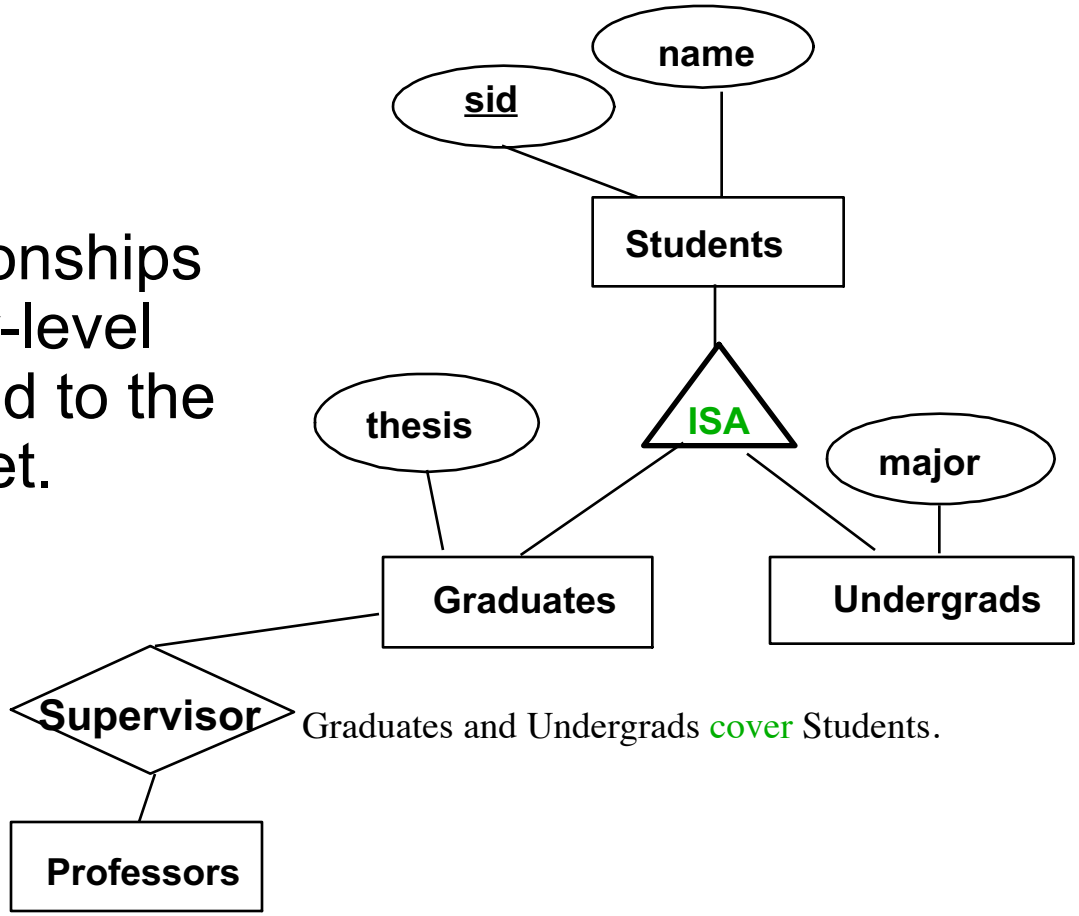
- Examples:



# ISA Hierarchy

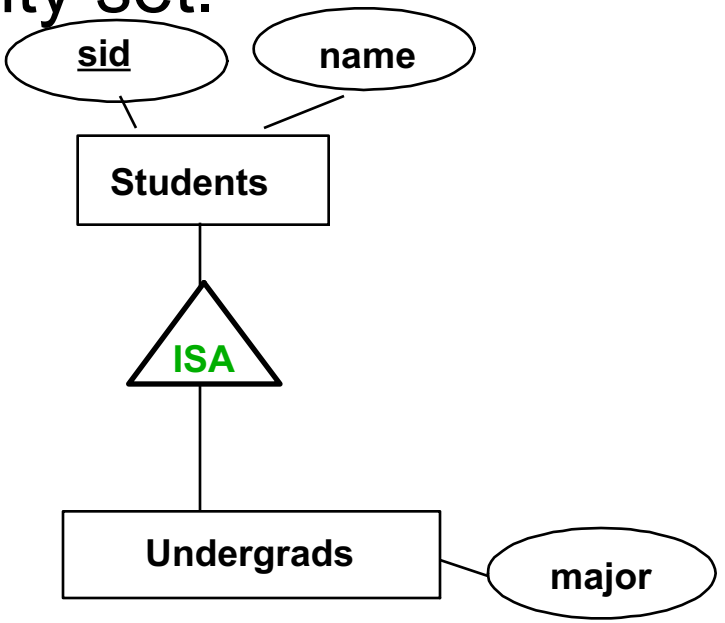
- Consider forming a new entity set as the union of two or more entity sets.

- Attributes and relationships common to all lower-level entity sets are moved to the higher-level entity set.



# ISA Hierarchy (cont.)

- Also consider forming a derived entity set by taking a subset of a given entity set.
  - We did a specialization



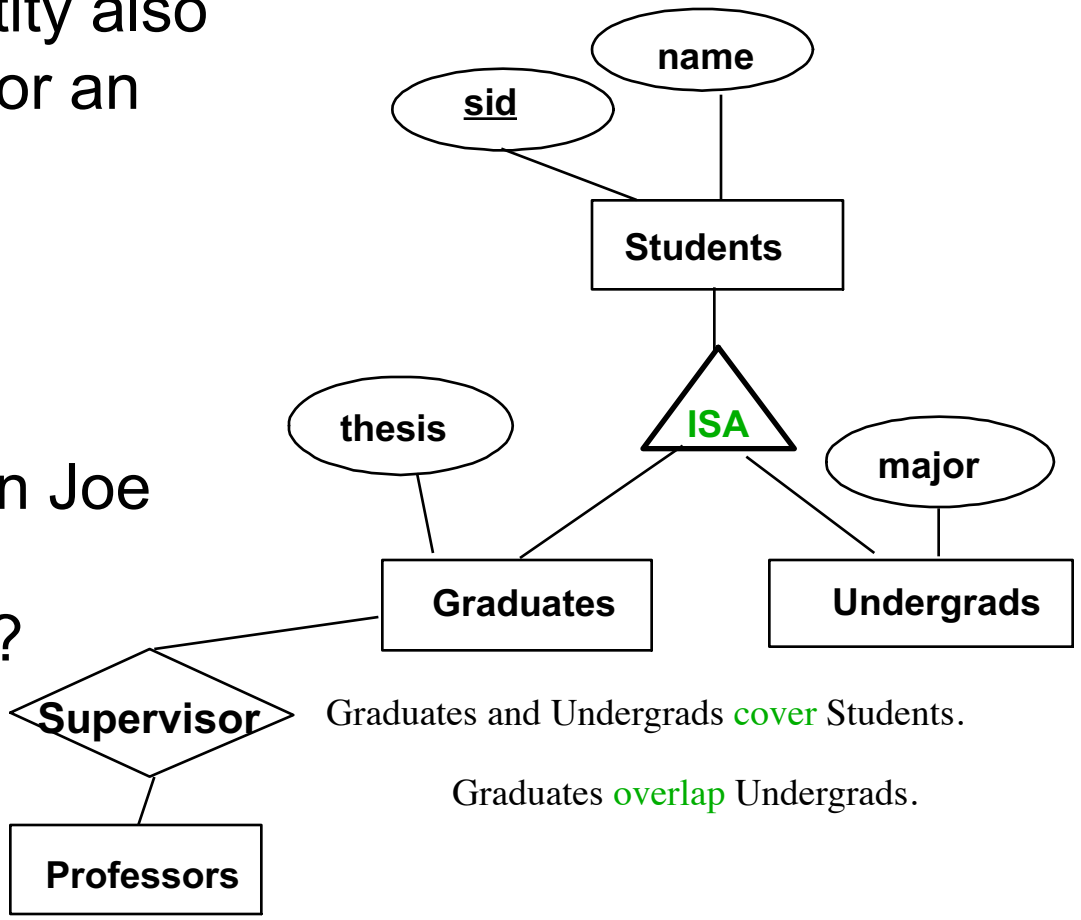
# Properties of ISA

- Inheritance
  - Attributes of supertype are attributes of subtype
  - Key of supertype is key of subtype
  - Relationships of supertype are relationships of subtype
- Transitivity - Hierarchy of IsA
  - Undergard student is subtype of Student, Student is subtype of Person, so Undergard student is also a subtype of Person
- Reasons for using ISA:
  - Makes ER diagram more concise and readable.
  - Common attributes/relationships need not be repeated.

# ISA Constraints

- *Covering constraints*:
  Does every *Students* entity also have to be a *Graduates* or an *Undergrads* entity?
  *(default:no)*

- *Overlap constraints*: Can Joe be a *Graduates* as well as an *Undergrads* entity?
  (default:*disallowed*)



# Conceptual Design Using the ER Model

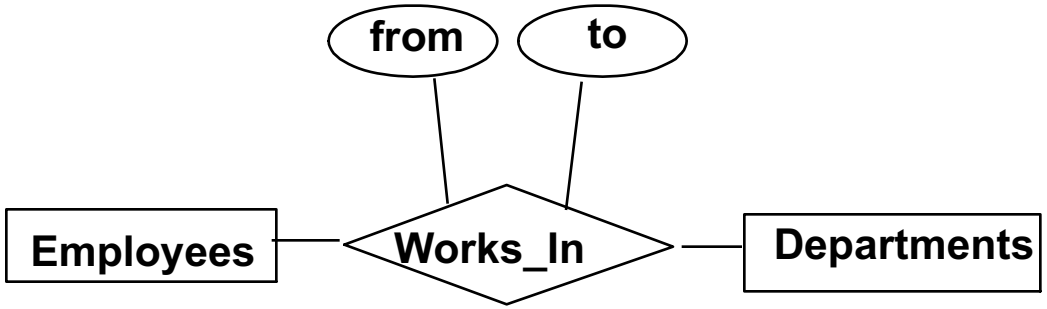
- Design choices:
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?
  - Identifying relationships: binary or ternary?

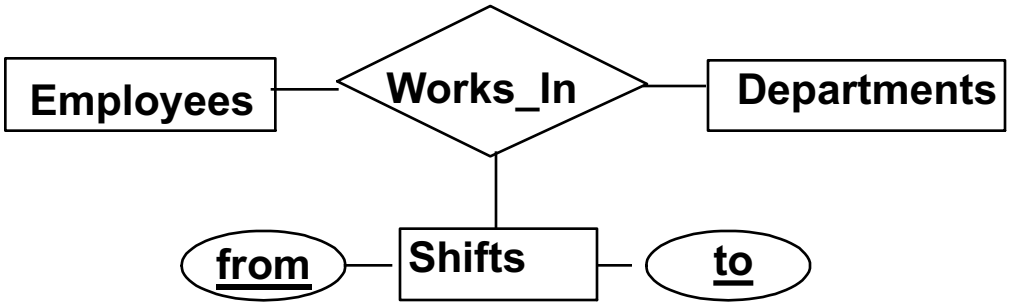
# Entity vs. Attribute

- Should *address* be an attribute (of Students) or an entity (connected to Students by a relationship)?
- Depends on (a) its use and (b) its relationships with other entities.
  - is it an object that we want to keep information about (independent of Students)?
  - does it participate in a relationship with an entity other than students?
  - are there many students with no addresses?
  - can several students 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.)

- Compare & contrast
- Can an employee work in two or more shifts in the same department?
  - first diagram: No
  - second diagram: Yes

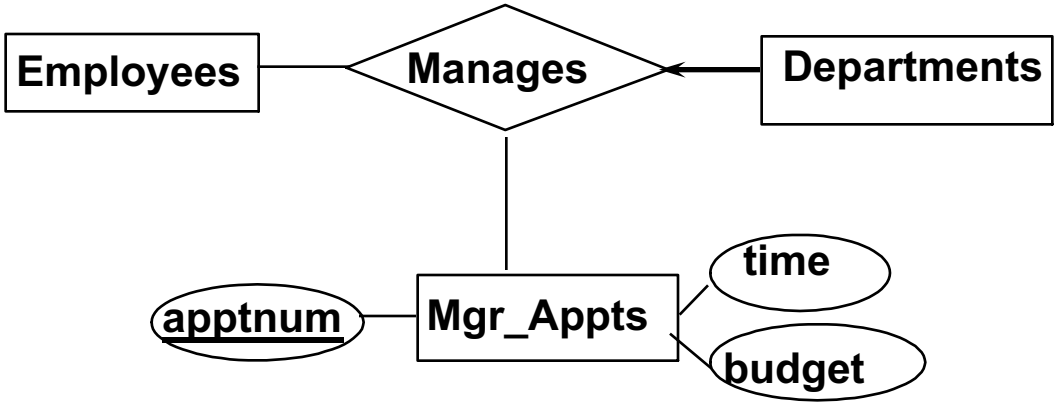


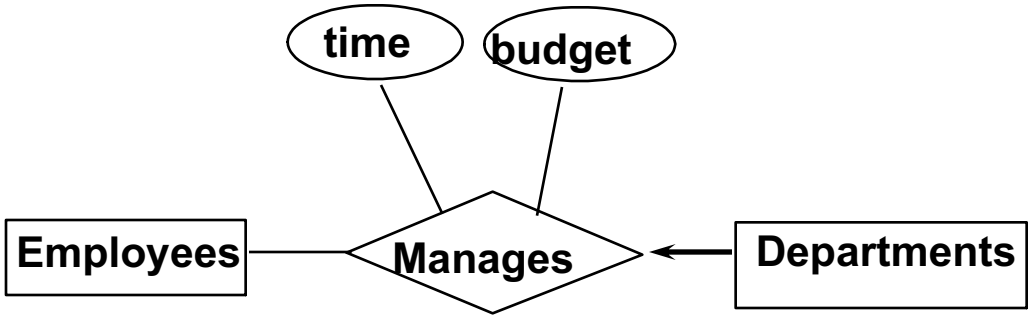




We have made up an entity called Shifts!

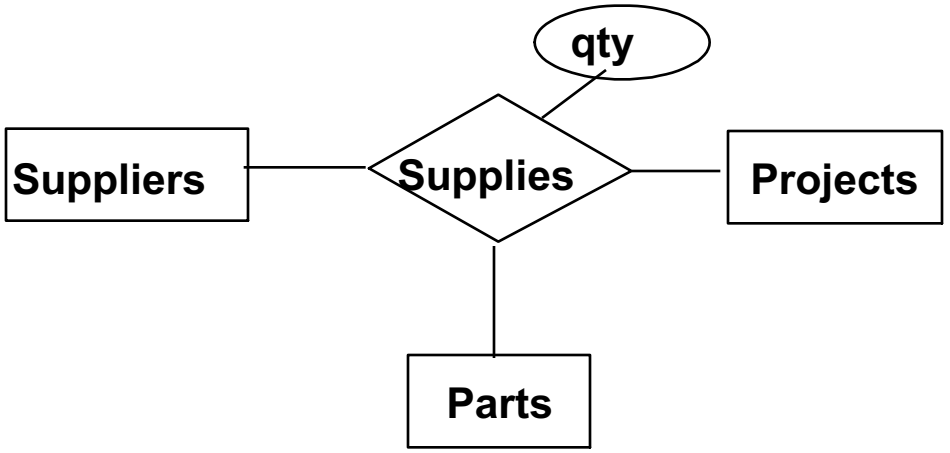
# Entity vs. Relationship

- Compare & contrast



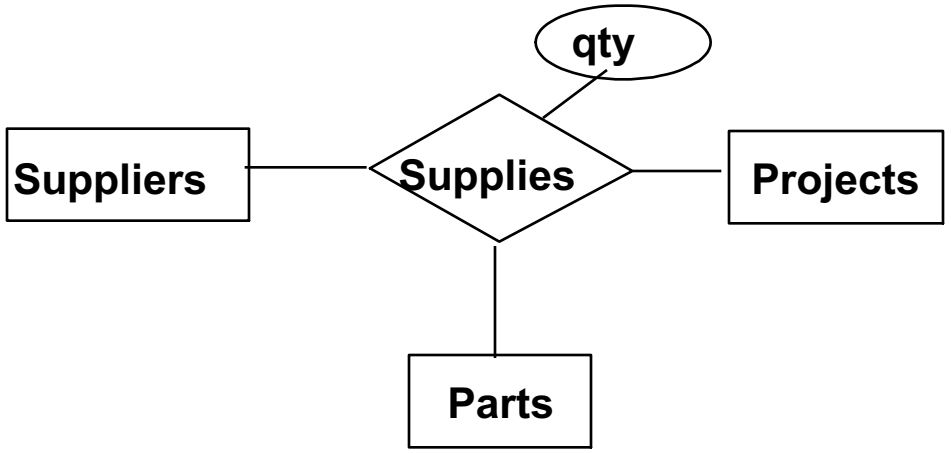
# Binary vs. Ternary Relationships



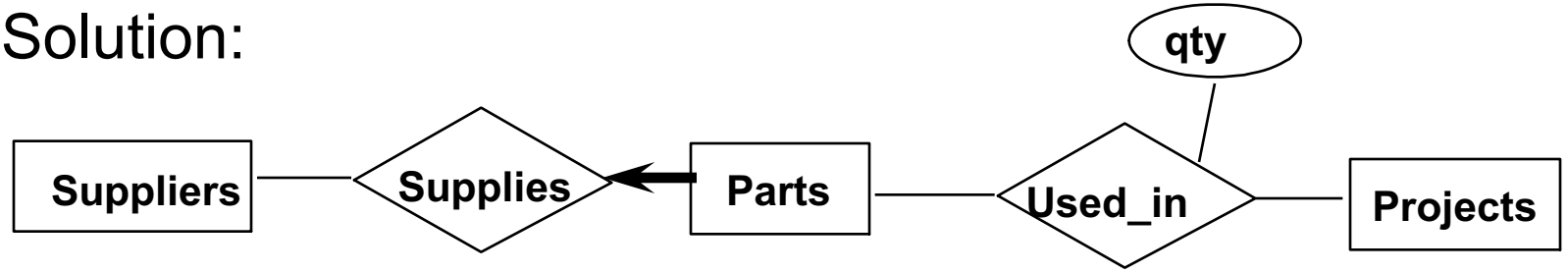

- Can we represent this using binary relationships?
  - ✓ supplier s "supplies" part p,
  - ✓ part p "used_in" project r, and
  - ✓ Supplier s "supplies_to" project r
- No combination of binary relationships implies that part p supplied by supplier s is used in project r.
- Not clear how to record *qty*?

# Binary vs. Ternary Relationships

- Now 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:



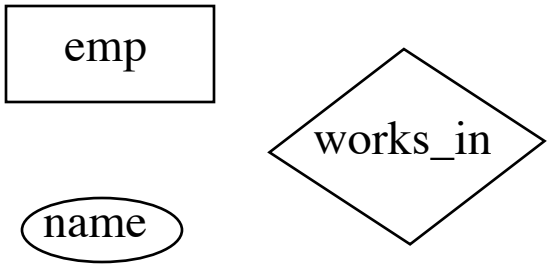

What is the additional constraint here?

# Work Out!

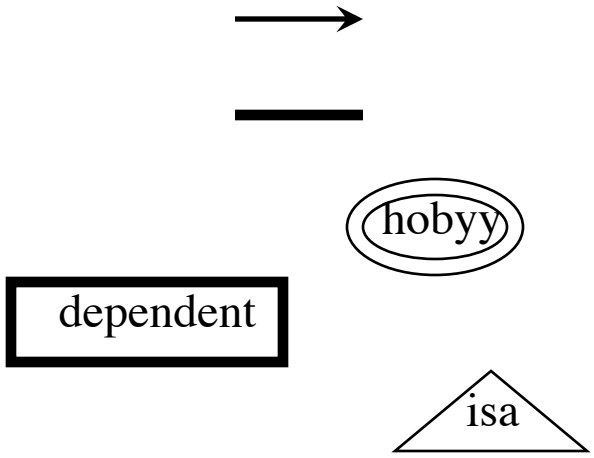
- Model the following in ER:
  - Canada Posts has for each employee a name, a phone number, and an employee id.
  - For each delivery employee, a cell number is also kept.
  - Each postal code in the city has a community name and a designated delivery employee.

# ER Review

- Basics:
  - entities,
  - relationships,
  - attributes




- Additions:
  - key constraint
  - participation constraint
  - Set-valued attributes
  - weak entities
  - isa hierarchy



# 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 ...