Entity-Relationship Model

E/R Diagrams

Converting E/R Diagrams to Relational Schema

Jeff Ullman

(edited by Muazzam Siddiqui)

Purpose of E/R Model

- The E/R model allows us to sketch database schema designs.
  - Includes some constraints, but not operations.
- Designs are pictures called *entity-relationship diagrams*.
- Later: convert E/R designs and instance to graphs

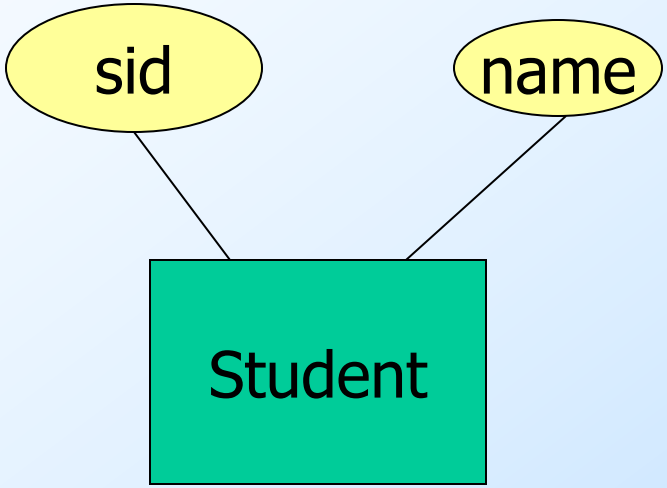
Entities and Entity Sets

- *Entity* = "thing" or object.
- *Entity set* = collection of similar entities.
  - Similar to a class in a class diagram.
- *Attribute* = property of an entity

E/R Diagrams

- In an entity-relationship diagram:
  - Entity = rectangle.
  - Attribute = oval, with a line to the rectangle representing the entity

Example:

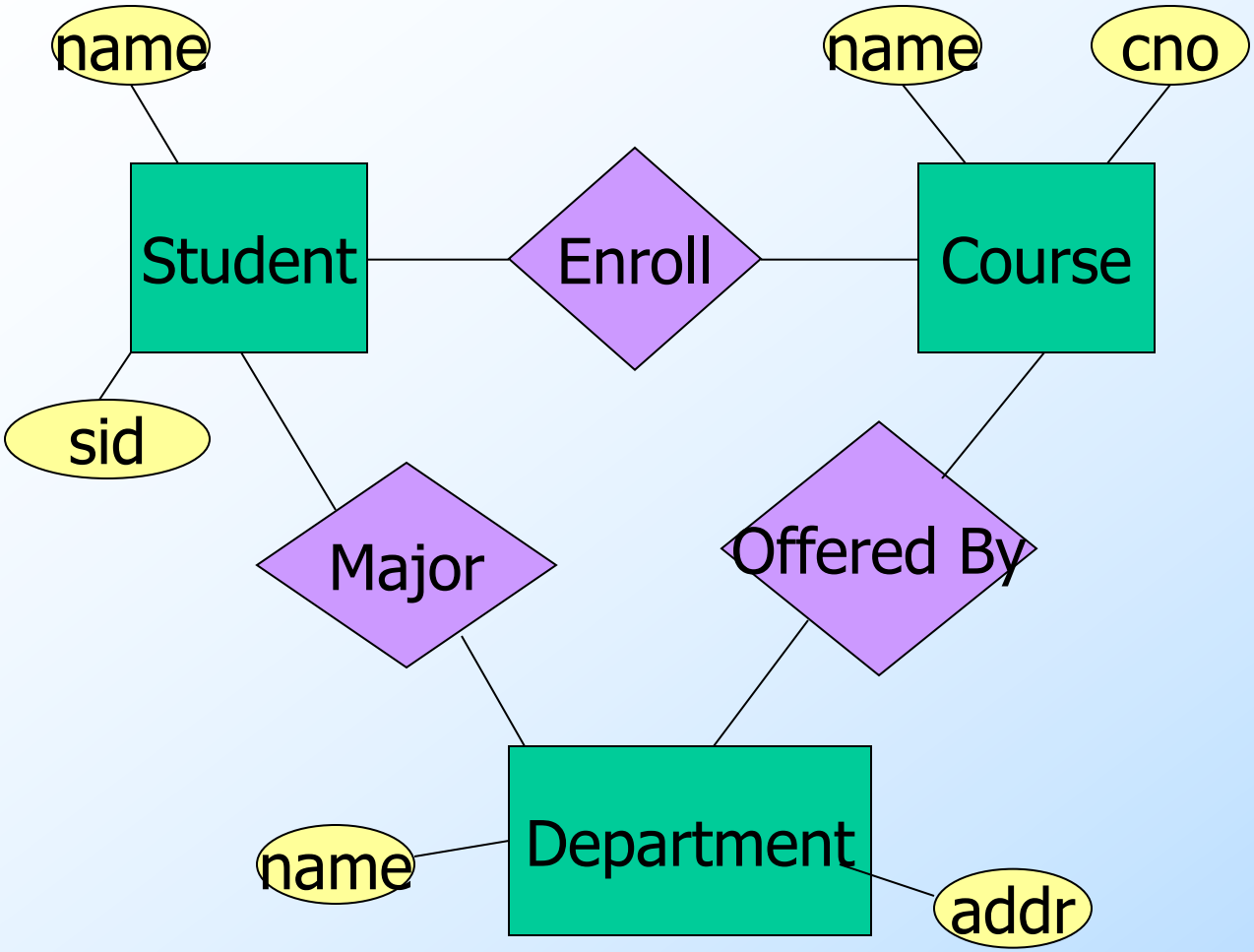


- Entity Student has two attributes, sid and name
- Each Student entity has values for these two attributes, e.g. (s1, Anna)

Relationships

- A relationship connects two or more entities
- It is represented by a diamond, with lines to each of the entities involved

Example: Relationships



Relationship Set

- The current "value" of an entity set is the set of entities that belong to it
  - Example: the set of all students in our database
- The "value" of a relationship is a *relationship set*, a set of tuples with one component for each related entity set

Example: Relationship Set

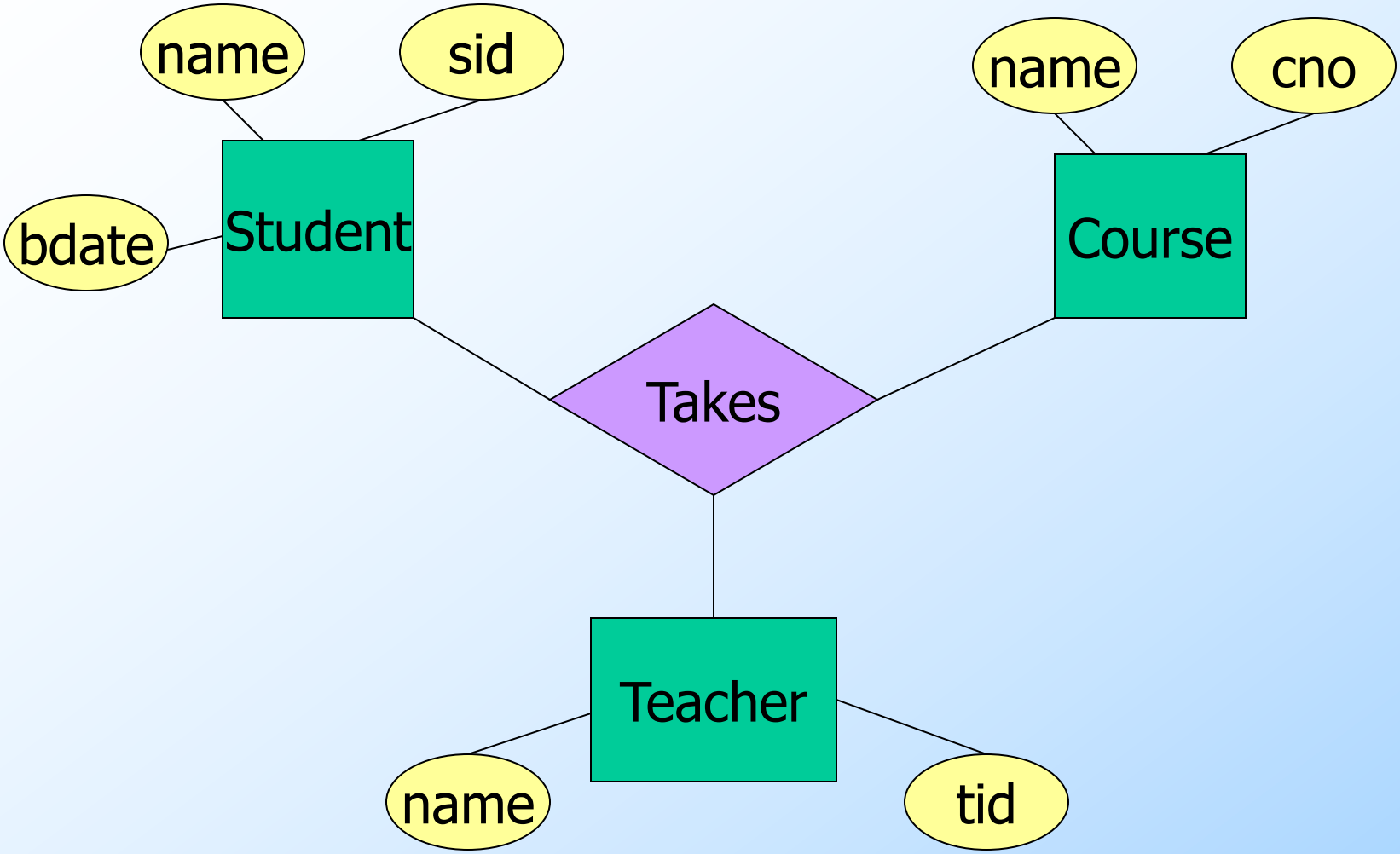
- For the relationship Major, we might have a relationship set like:

Student	Department
s1	CS
s1	Math
s2	Chemistry
s3	Math
s3	Physics

Multiway Relationships

- Sometimes, we need a relationship that connects more than two entities
- Suppose that student takes a course taught by a teacher (s,c,t)
- Three binary relationships do not allow us to precisely capture this relationship between a student, course, and teacher.
  - But a 3-way relationship would.

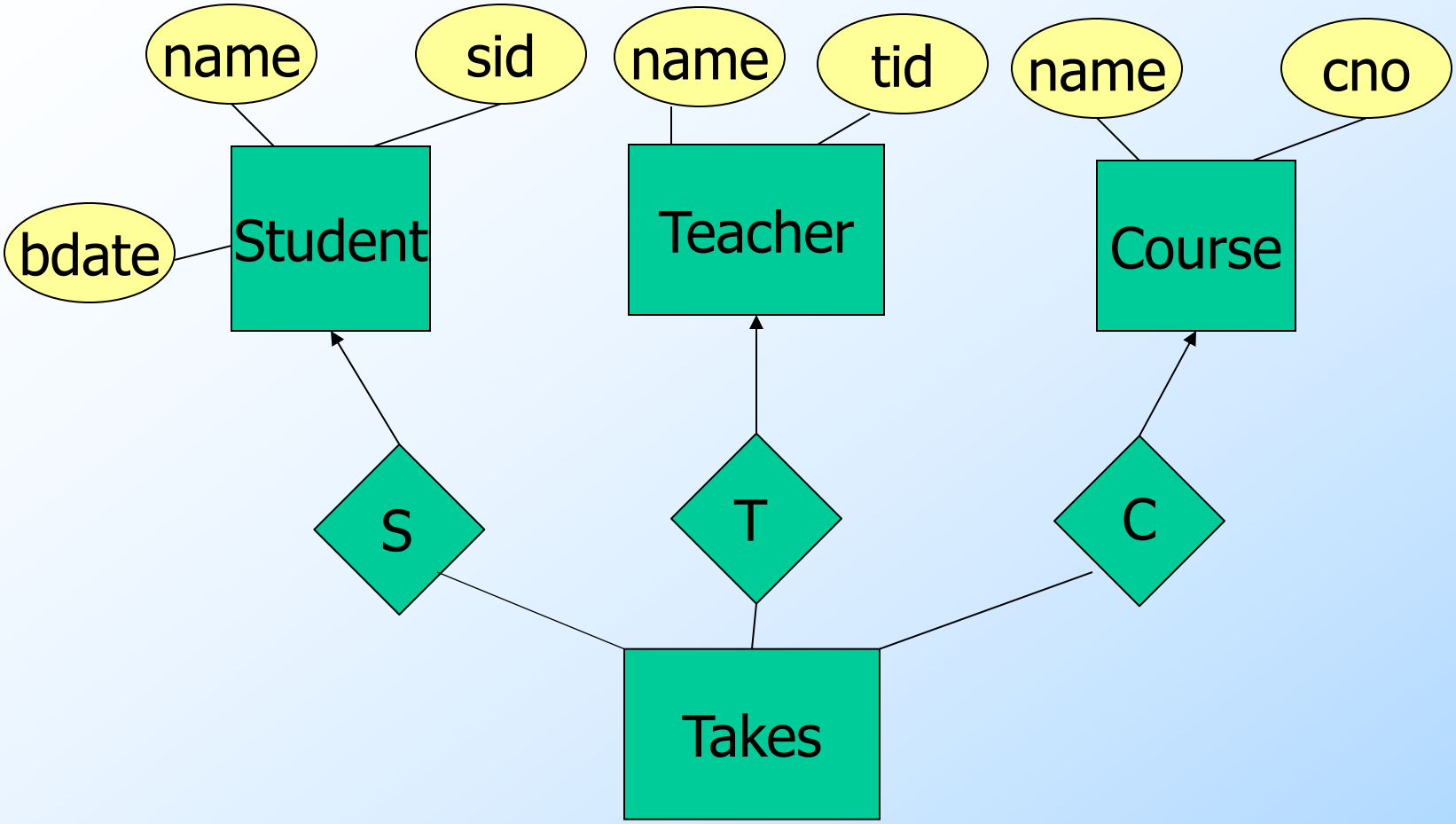
Example: 3-Way Relationship



A Ternary Relationship Set

Student	Course	Teacher
s1	c1	t1
s1	c2	t2
s2	c2	t1

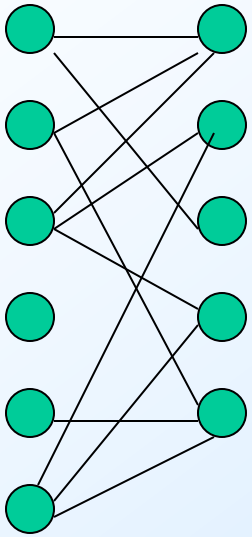
Example: Representation with 3 many-to-one relationships



Many-Many Relationships

- Focus: binary relationships, such as Enroll between Students and Courses
- In a *many-many* relationship, an entity of either set can be connected to many entities of the other set
  - E.g., a student may enroll in many courses; a course may enroll many students.

In Pictures:

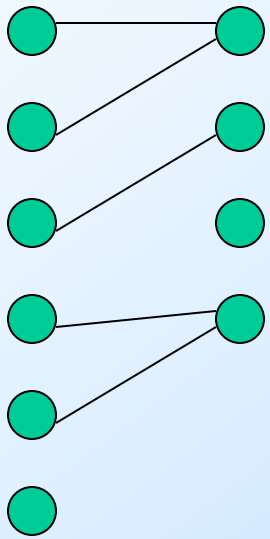


many-many

Many-One Relationships

- Some binary relationships are *many - one* from one entity set to another
- Each entity of the first set is connected to at most one entity of the second set
- But an entity of the second set can be connected to zero, one, or many entities of the first set
- A many-one relationship is a function

In Pictures:



many-one

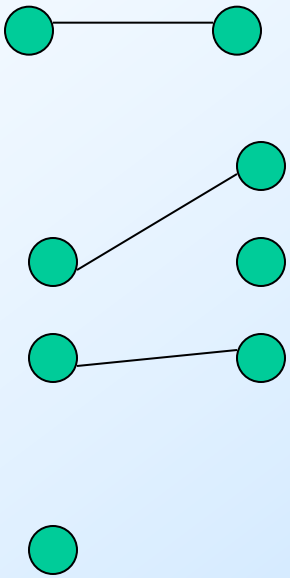
Example: Many-One Relationship

- Offered By, from Courses to Departments is many-one
- A course is offered by at most one department
- But a department can offer any number of students, including zero

One-One Relationships

- Some binary relationships are *one-one* from one entity set to another
- Each entity of the first set is connected to at most one entity of the second set
- Likewise, each entity of the second set is connected to at most one entity of the first set

In Pictures:

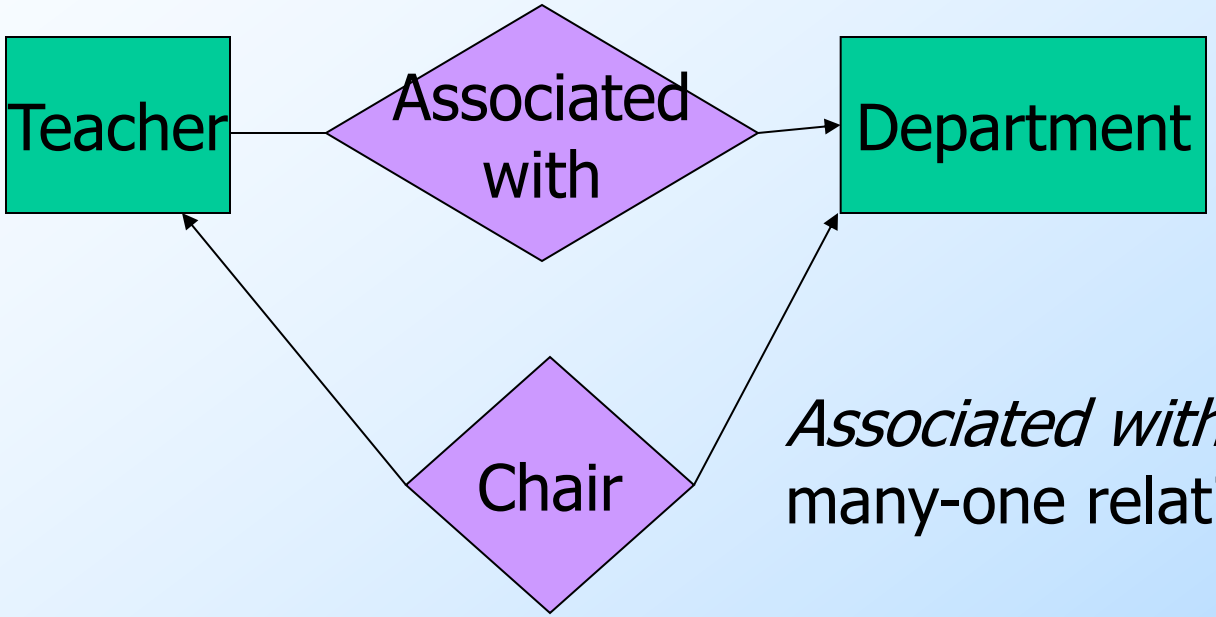


one-one

Representing Relationships

- Show a many-many relationship by a line
- Show a many-one relationship by an arrow entering the "one" side
- Show a one-one relationship by arrows entering each one sides

Example: Many-One & One-One Relationships



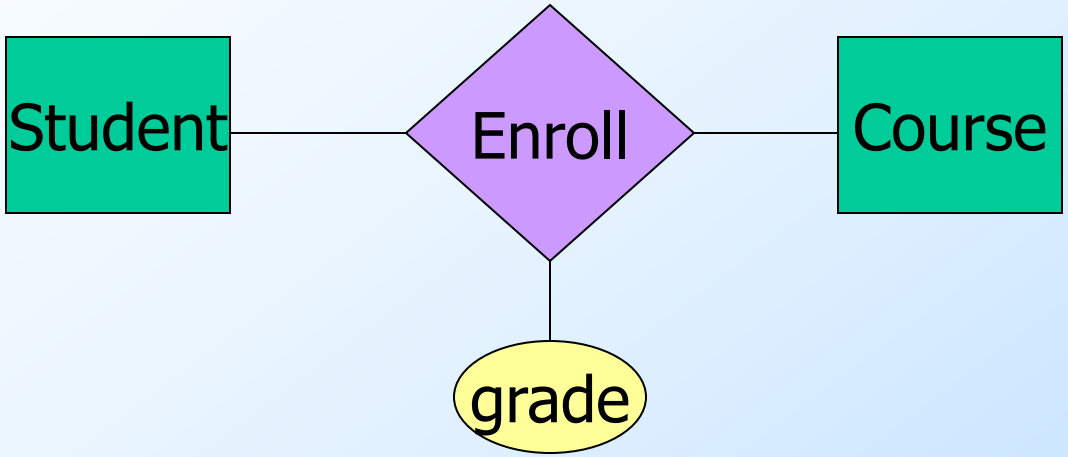
Associated with is a many-one relationship

Chair is a one-one relationship

Attributes on Relationships

- Sometimes it is useful to attach an attribute to a relationship
- Think of this attribute as a property of each tuple (relationship) in the relationship set

Example: Attribute on Relationship



Grade is a function of both the student and the course, not of one alone

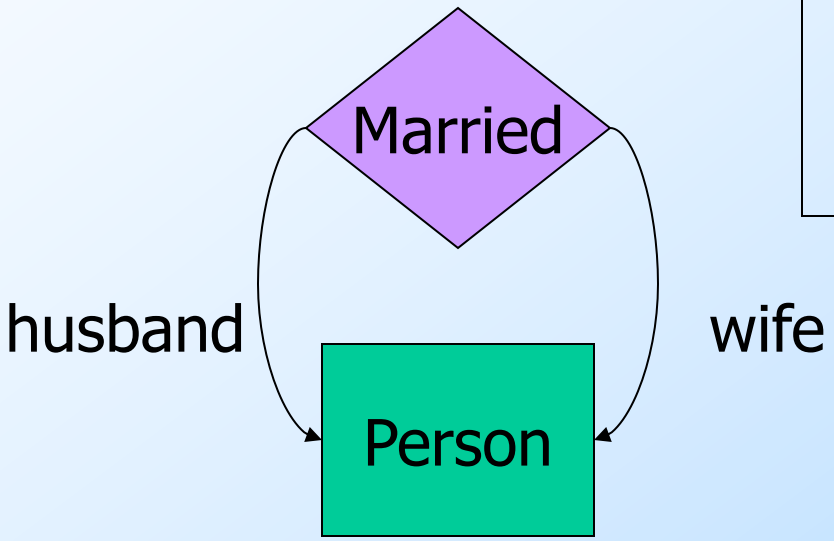
Roles

- Sometimes an entity set appears more than once in a relationship
- Label the edges between the relationship and the entity set with names called *roles*

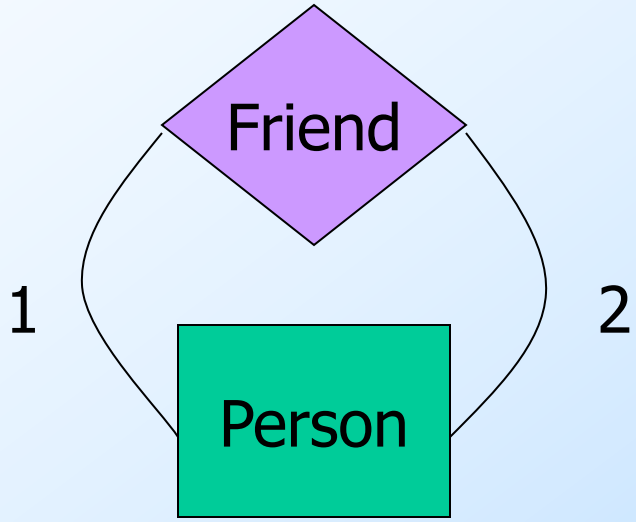
Example: Roles

Relationship Set

Husband	Wife
Bob	Ann
Joe	Sue
…	…



Example: Roles



Relationship Set

Friend1	Friend2
Bob	Ann
Joe	Sue
Ann	Bob
Joe	Moe
…	…

Keys

- A *key* is a set of attributes for one entity set such that no two entities in this set agree on all the attributes of the key
  - It is allowed for two entities to agree on some, but not all, of the key attributes.
- We must designate a key for every entity set

Keys in E/R Diagrams

- Underline the key attribute(s)
- In an Isa hierarchy, only the root entity set has a key, and it must serve as the key for all entities in the hierarchy

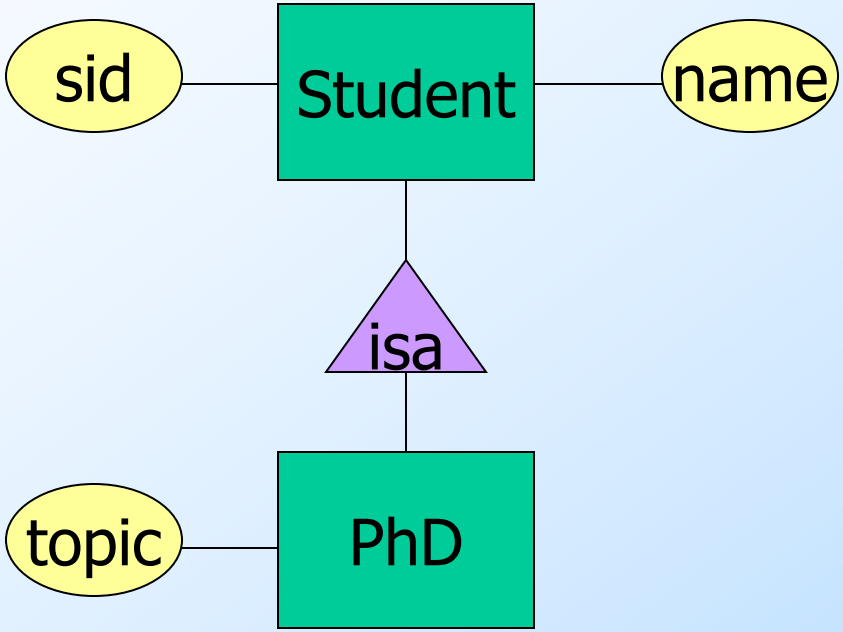
Subclasses

- *Subclass* = special case = fewer entities = more properties = more relationships
- Example: PhD students are a kind of student.
  - Not every student is a PhD student, but some are.
  - Let us suppose that in addition to all these *properties* (attributes and relationships), a PhD student also has the attribute topic

Subclasses in E/R Diagrams

- Assume subclasses form a tree.
  - I.e., no multiple inheritance.
- Isa triangles indicate the subclass relationship.
  - Point to the superclass.

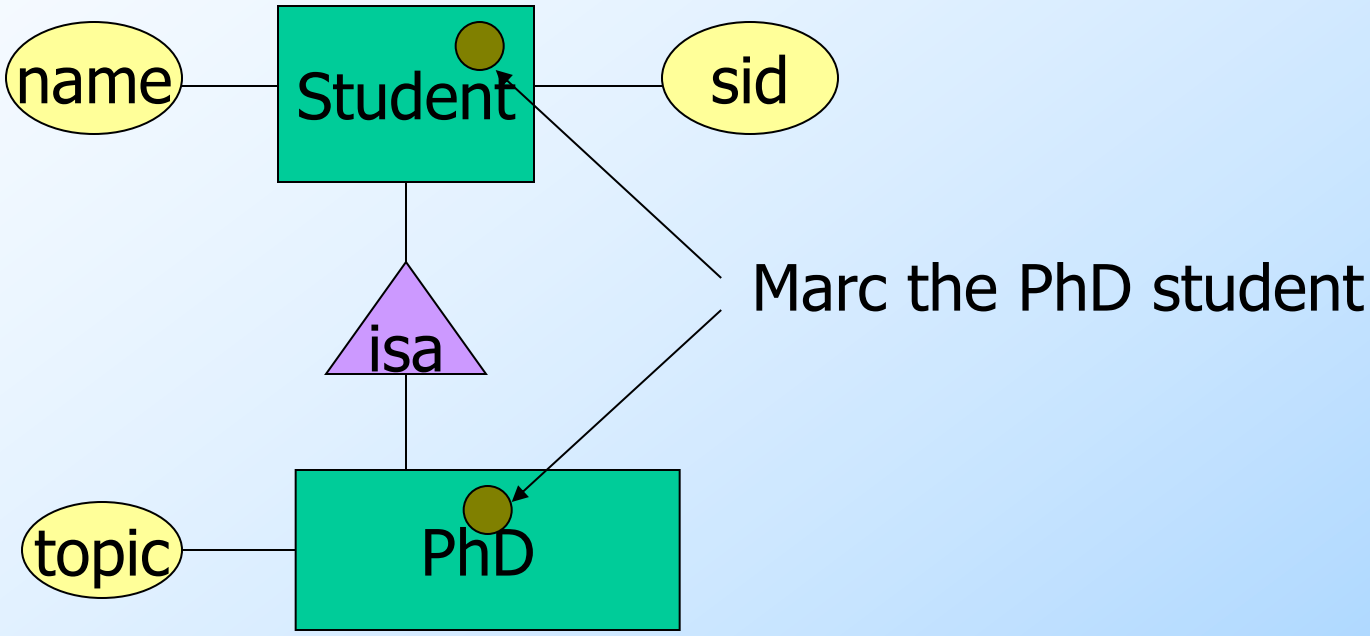
Example: Subclasses



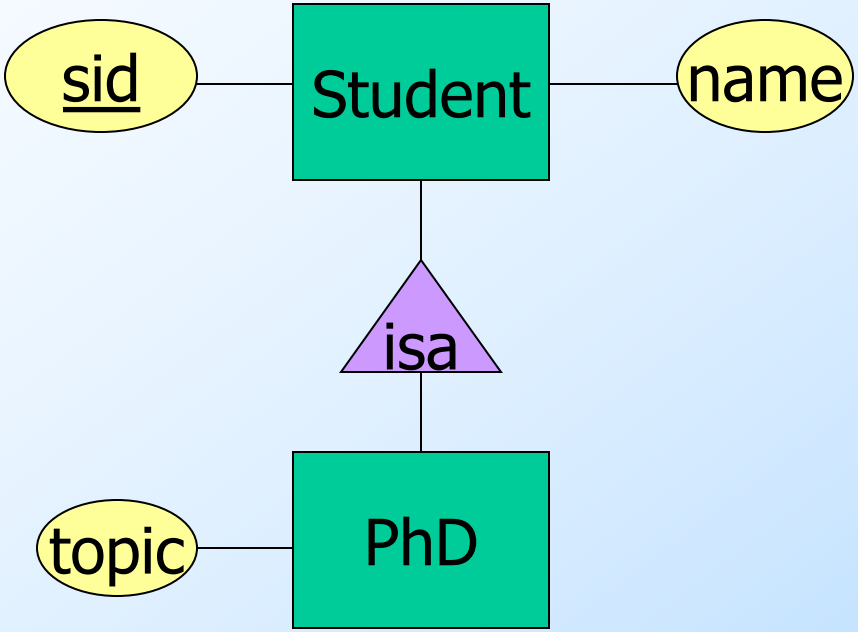
E/R Vs. Object-Oriented Subclasses

- In OO, objects are in one class only.
  - Subclasses inherit from superclasses.
- In contrast, E/R entities have *representatives* in all subclasses to which they belong.
  - Rule: if entity *e* is represented in a subclass, then *e* is represented in the superclass (and recursively up the tree).

Example: Representatives of Entities



Example: name is Key for Student



From E/R Diagrams to Relations

- Entity set -> relation.
  - Attributes -> attributes.
- Relationships -> relations whose attributes are only:
  - The keys of the connected entity sets.
  - Attributes of the relationship itself.

From E/R Diagrams to Relations

- Mapping of Entity Sets
  - For each entity set
    - create a relation
- Mapping of Binary 1:1 Relationships
  - For each 1:1 relationship
    - Foreign key approach
      - Add the primary key of one relation as the foreign key of the other
    - Relationship relation approach
      - Create a new relation and add the primary keys of participating relations

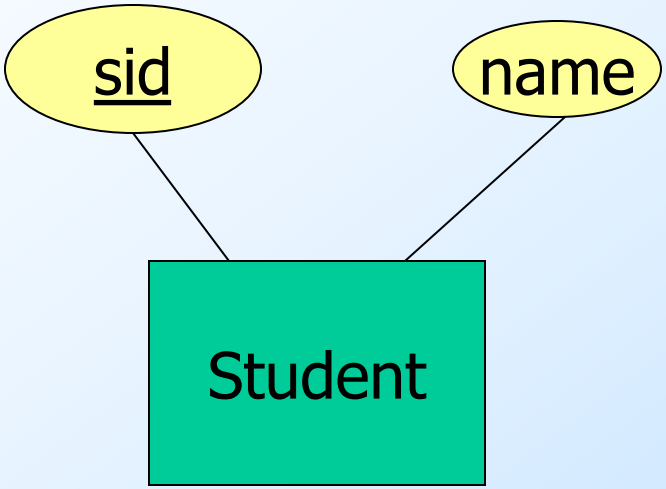
From E/R Diagrams to Relations

- Mapping of Binary 1:N Relationships
  - For each 1:N relationship
    - Foreign key approach
      - Identify relation, say S, that represents participating entity type at N-side of relationship type
      - Include primary key of other entity type as foreign key in S
    - Relationship relation approach
      - Create a new relation and add the primary keys of participating relations

From E/R Diagrams to Relations

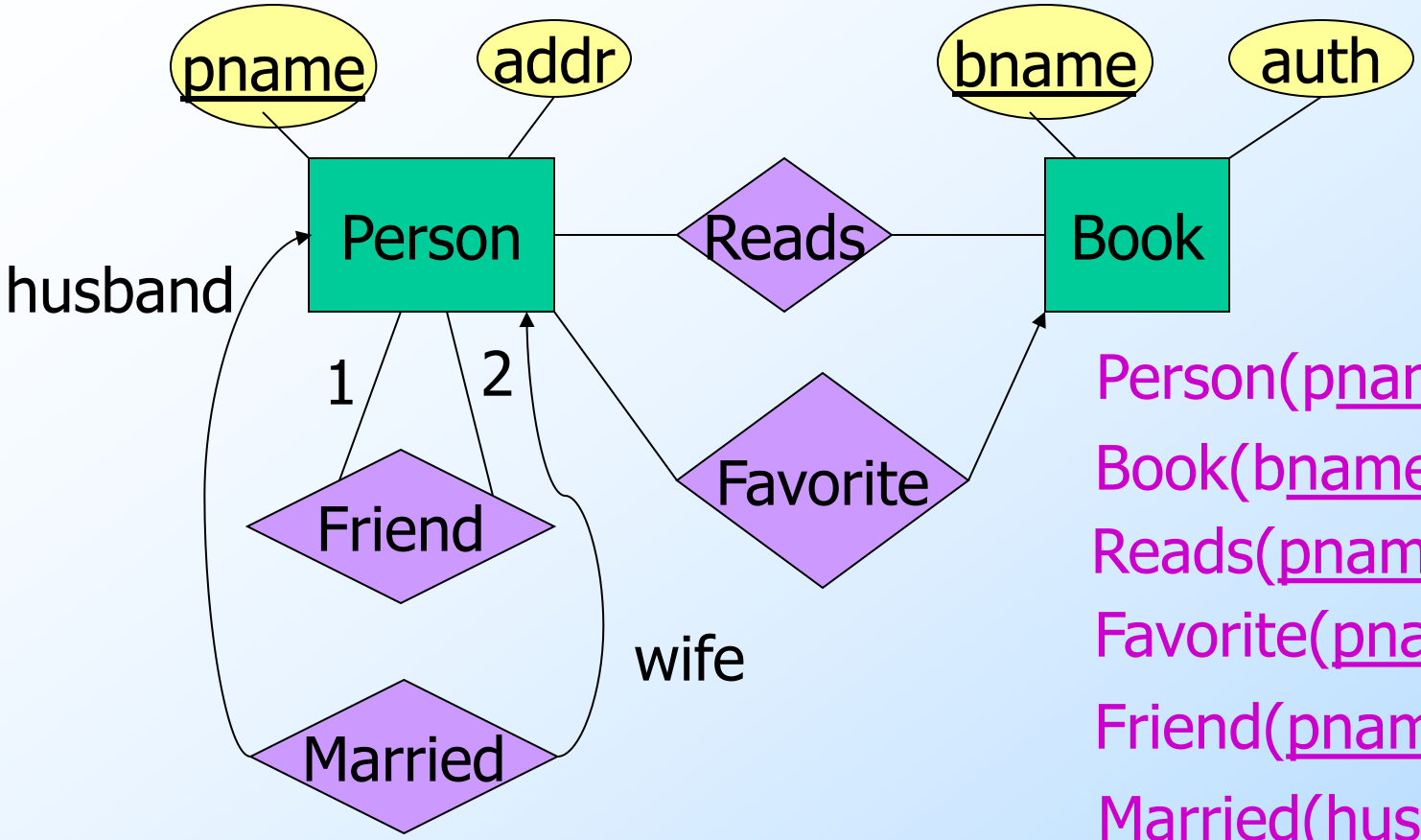
- Mapping of Binary M:N Relationships
  - For each M:N relationship
    - Use the relationship relation approach
      - Create a new relation and add the primary keys of participating relations
- Mapping of N-ary Relationships
  - For each N-ary relationship
    - Use the relationship relation approach
      - Create a new relation and add the primary keys of participating relations

Entity Set -> Relation



Relation: Student(sid, name)

Relationship -> Relation



Person(pname, addr)

Book(bname, auth)

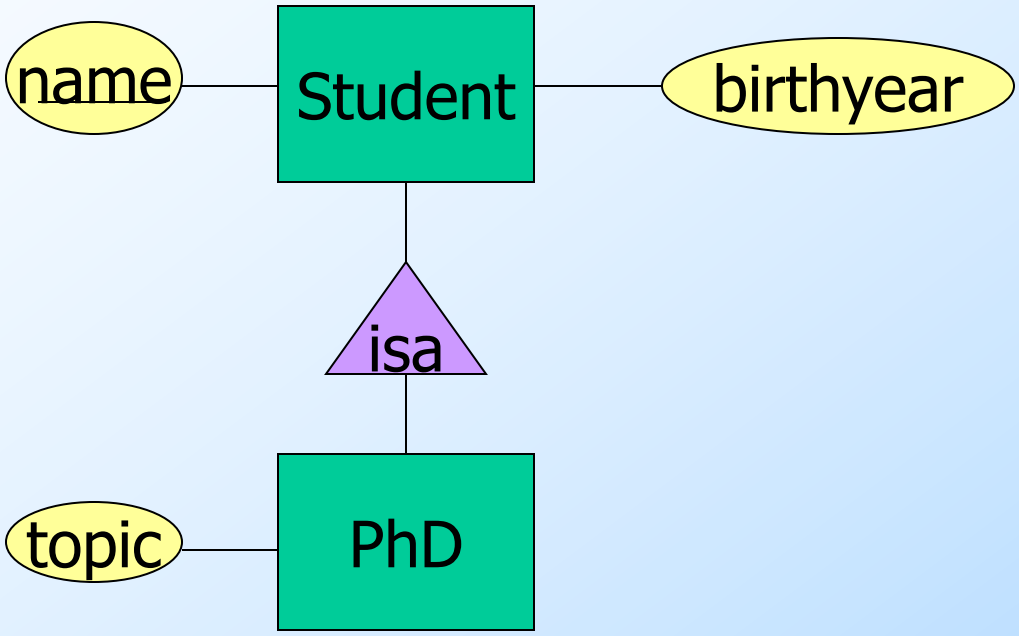
Reads(pname, bname)

Favorite(pname, bname)

Friend(pname1, pname2)

Married(husband, wife)

Example: Subclass -> Relations



E/R Style

name	birthyear
Ann	1999
Ellen	1995

Student

name	topic
Ellen	PL

PhD

Good for queries like "find all student (including PhDs) born in 1995."

In SQL (relationships)

- Consider student, course, enroll
- Maintain referential integrity

```
create table student (sid integer PRIMARY KEY,
                      name text);
create table course (cno integer PRIMARY KEY,
                     name text);
create table enroll (sid integer  REFERENCES student(sid),
                     cno integer REFERENCES course(cno),
                     grade VARCHAR(2),
                     primary key(sid,cno))
```

In SQL (subclasses) E/R style

- Consider student, PhD

```
create table Student (sid integer PRIMARY KEY,
                      name text,
                      birthyear integer);

create table PhD (sid integer REFERENCES Student(sid),
                  topic text);
```