What is a Database Management System?

1. Manages very large amounts of data.

2. Supports efficient access to very large amounts of data.

3. Supports concurrent access to v.l.a.d.

   - Example: bank and its ATM machines.

4. Supports secure, atomic access to v.l.a.d.

   - Contrast two people editing the same UNIX file — last to write "wins" — with the problem if two people deduct money from the same account via ATM machines at the same time — new balance is wrong whichever writes last.

Relational Model

- Based on tables, as:

acct#	name	balance
12345	Sally	1000.21
34567	Sue	285.48
...	...	...

- Today used in *most* DBMS's.
- Amazing success story of theoretical ideas taken to $20B industry.
- "Big three" DBMS companies — Oracle, Informix, Sybase — are among the largest software companies in the world.
  - IBM also plays, with its DB2 and other offerings.
  - Microsoft challenges in this area too — cheap DBMS on the desktop?

- Relational companies also challenged by startup “object-oriented DB” companies.
  - ❖ Conjecture: OODB is overblown; relational companies will prevail because they are moving rapidly to introduce OO concepts as needed via “object-relational” versions of their products and the SQL3 emerging standard.

Three Aspects to Studying DBMS's

1. Modeling and design of databases.
2. Programming: queries and DB operations like update.
   - SQL = "intergalactic dataspeak."
3. DBMS implementation.

CS145 = (1) + (2), while (3) is covered in CS245AB, CS347.

Why Study Design?

Sketch of structure for database needed, before implementation (say as relations in a relational DBMS).

- Allows discussion of issues in simplified terms before commiting to implementation.
- Example: registrar maintains a DB of courses, students, instructors, etc. What goes into the DB? What assumptions can we make, e.g., unique instructor for a course?

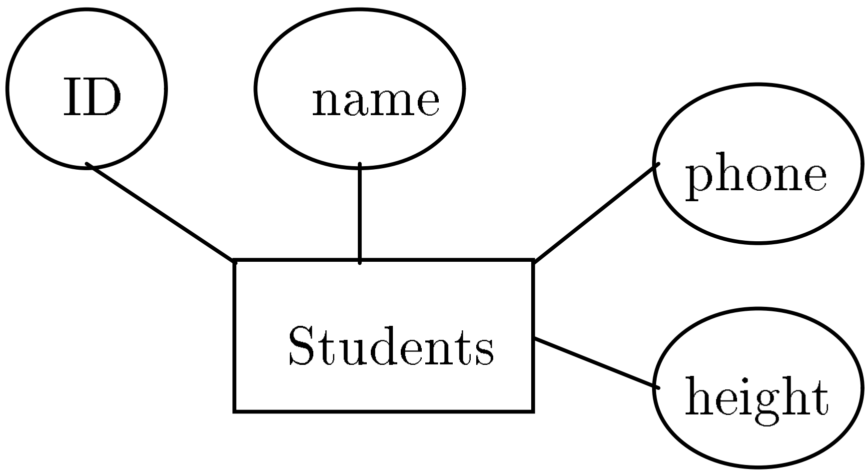
Design Notations

Entity/relationship (E/R) and Object Description Language (ODL) will be covered in CS145, and are in common use.

- For each there is a semimechanical way to convert to relational designs (= sets of tables).

Entity/Relationship Model

- Diagrams similar in spirit to OO models.
- *Entity* like object, = "thing."
- *Entity set* like class = set of "similar" entities/objects.
- *Attribute* = property of entities in an entity set, similar to fields or "instance variables."
- In diagrams, entity set → rectangle; attribute → oval.



Relationships

- Connect two or more entity sets.
- Represented by diamonds.

Relationship Set

Think of the "value" of a relationship set as a table.

- One column for each of the connected entity sets.
- One row for each list of entities, one from each set, that are connected by the relationship.

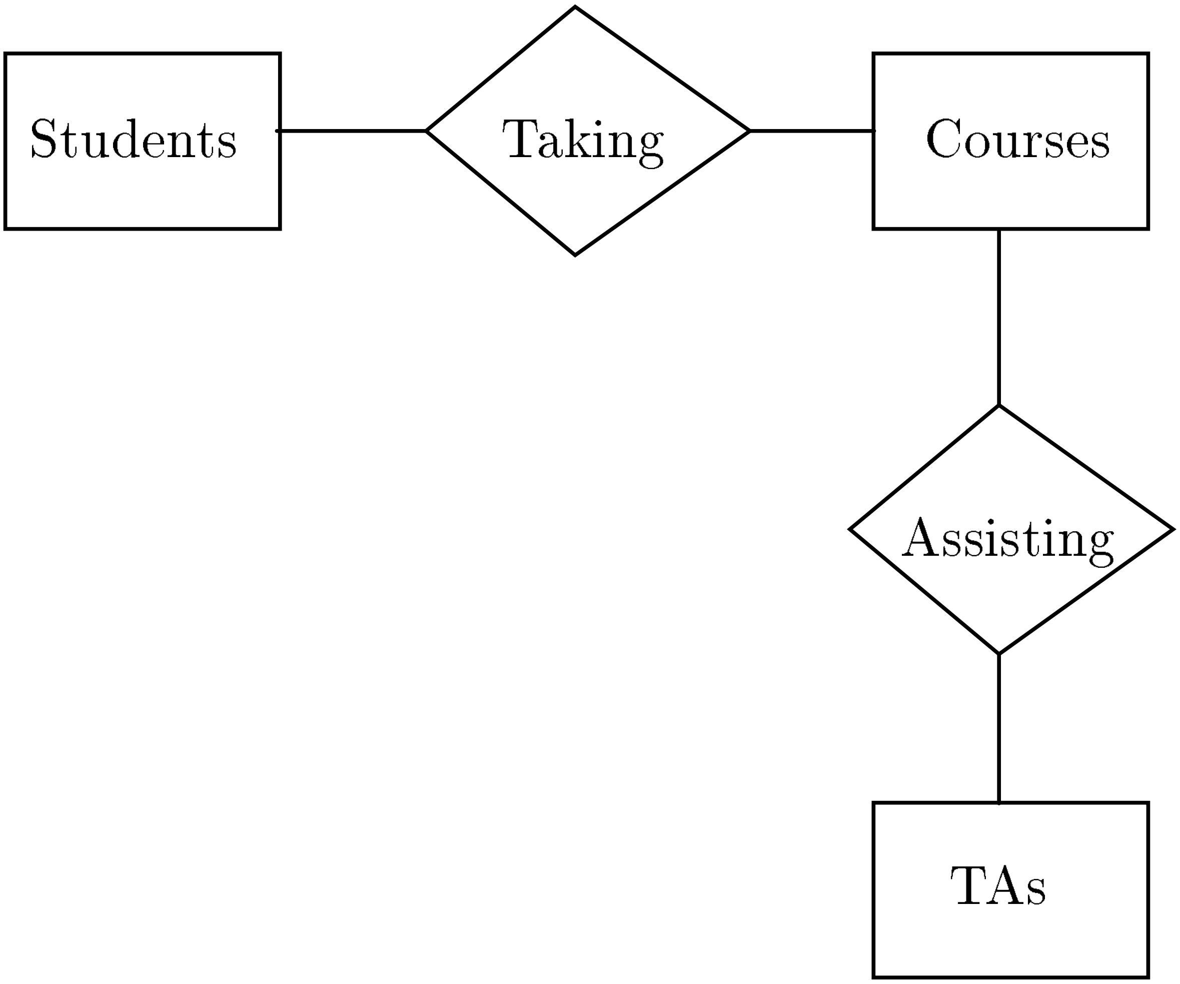
Students	Courses
Sally	CS145
Sally	CS244
Joe	CS145
...	...

Multiway Relationships

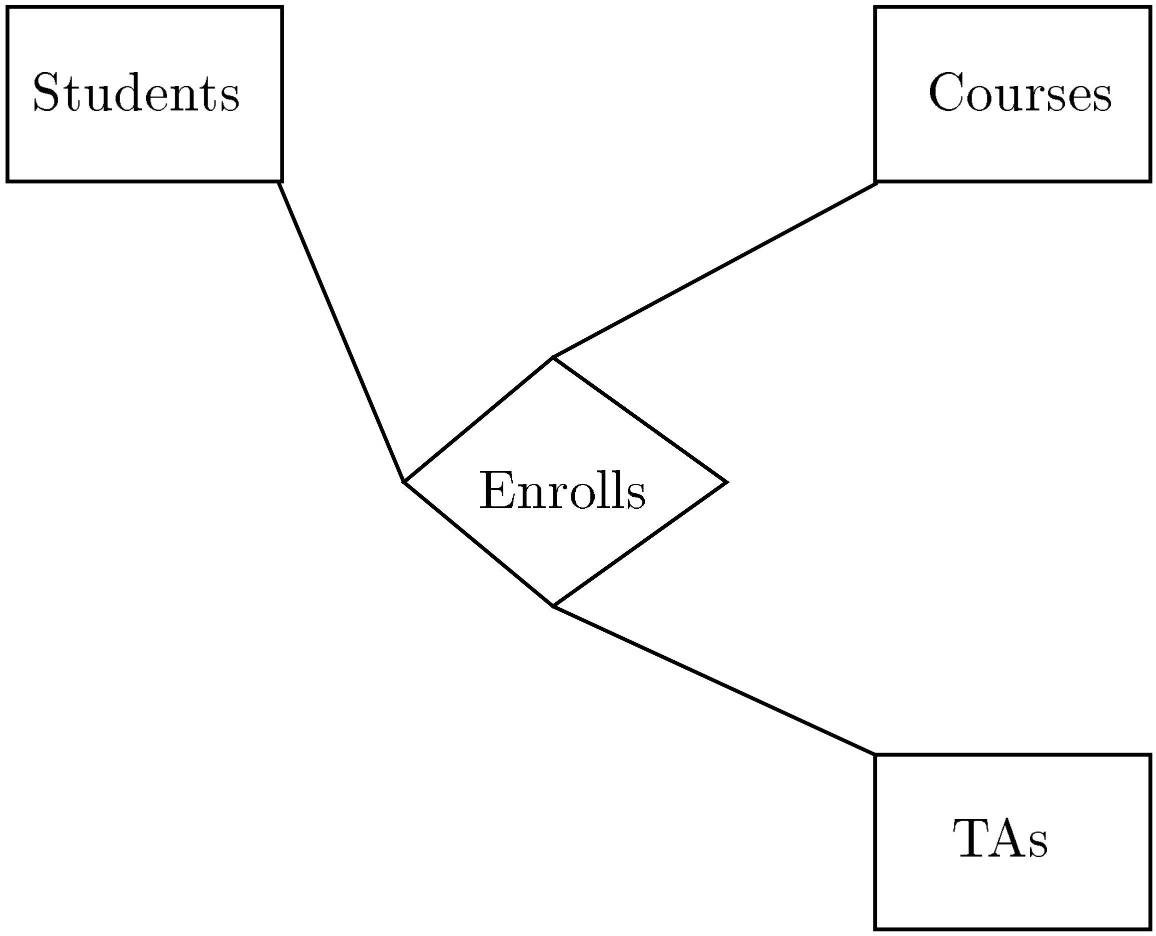
Usually *binary* relationships (connecting two E.S.) suffice.

- However, there are some cases where three or more E.S. must be connected by one relationship.

- Example: relationship among students, courses, TA's. Possibly, this E/R diagram is OK:



- Works in CS145, because each TA is a TA of all students. Connection student-TA is *only* via the course.
- But what if students were divided into sections, each headed by a TA?
  - Then, a student in CS145 would be related to only one of the TA's for CS145. Which one?
- Need a 3-way relationship to tell.

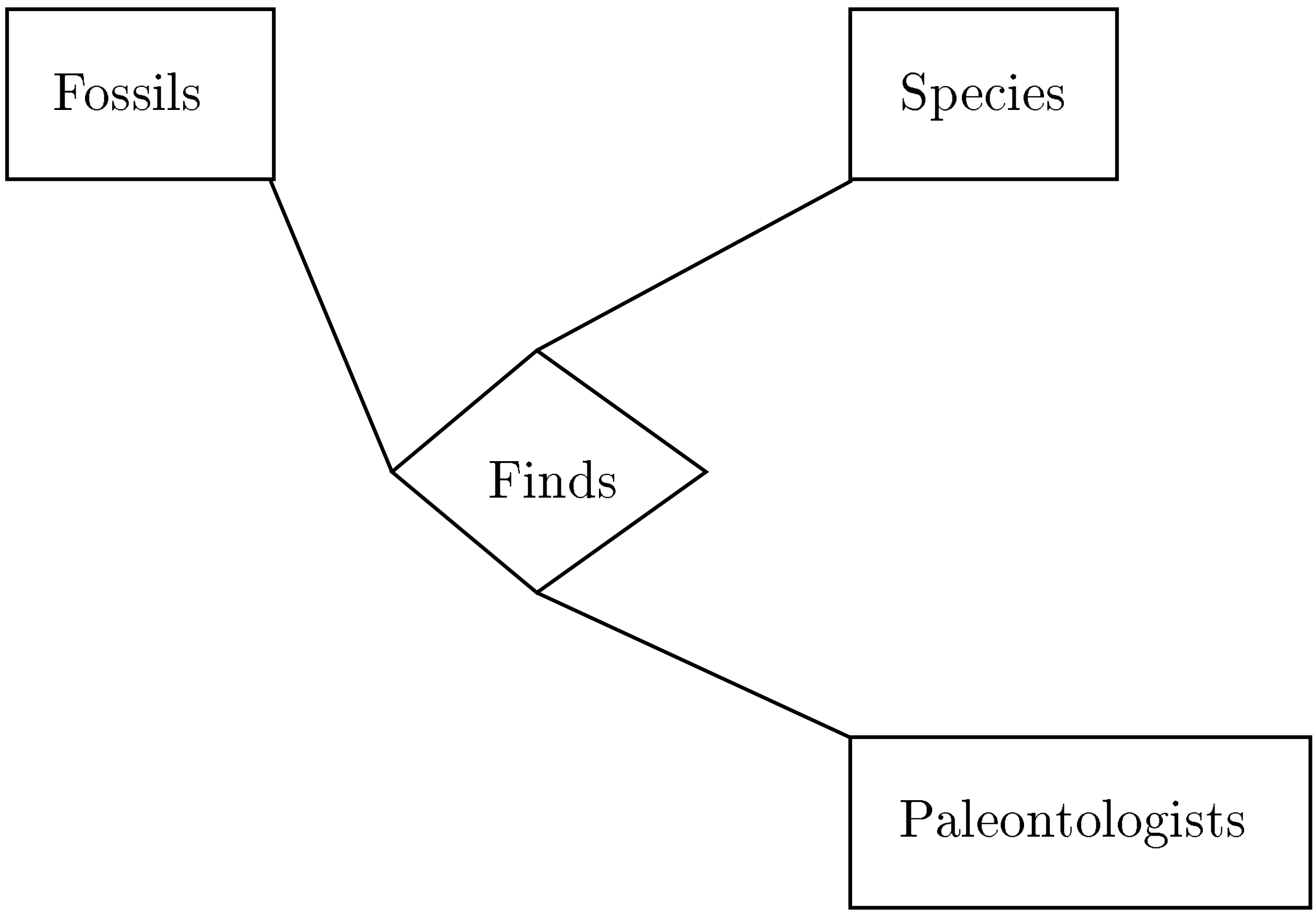


Students	Courses	TAs
Sally	CS145	Claire
Sue	CS145	Ankur
Joe	CS145	Claire
...	...	...

A Design Issue: Should Multiway Relationships be Split?

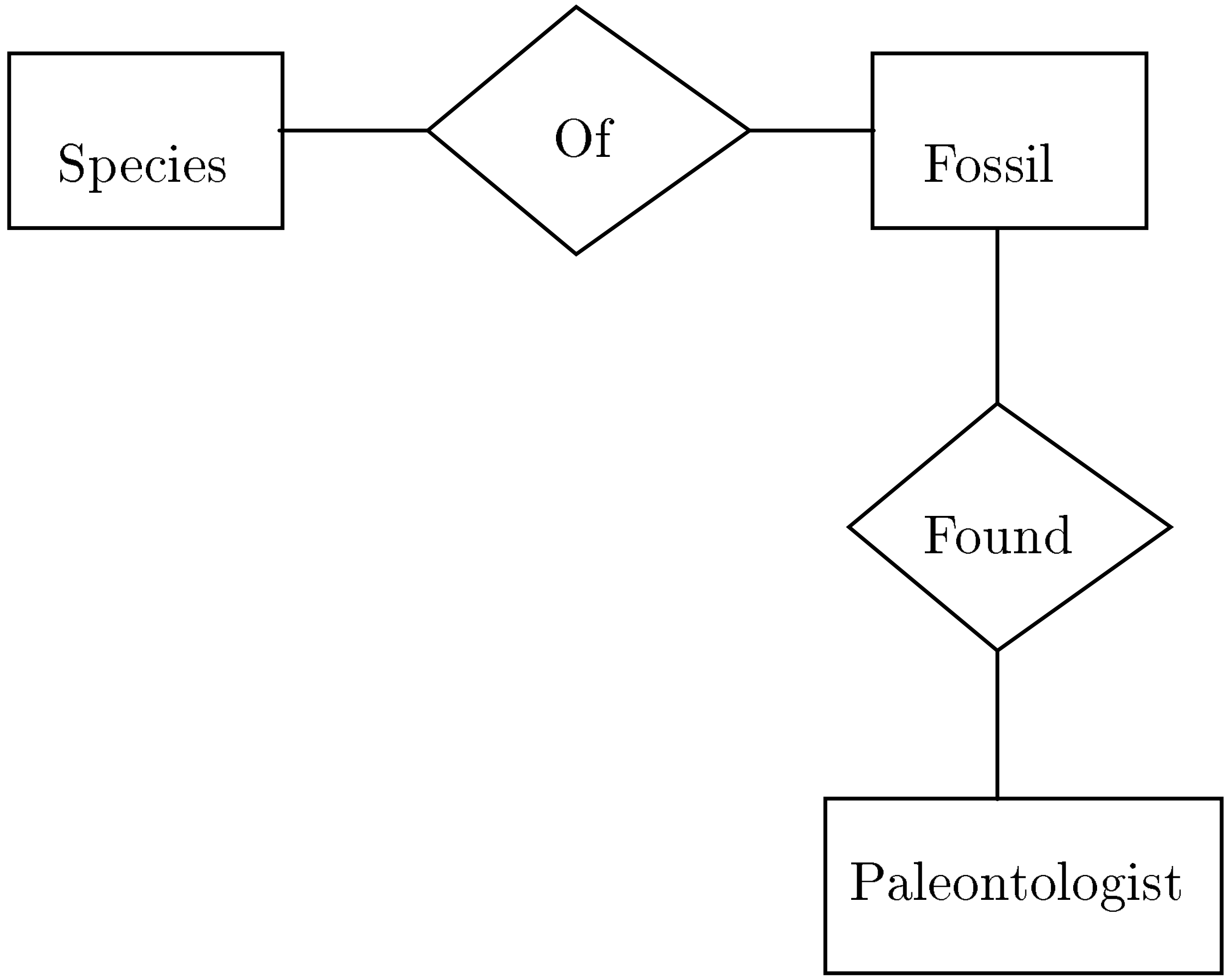
- Depends on whether it can be expressed as *independent* binary relationships.

- Example:



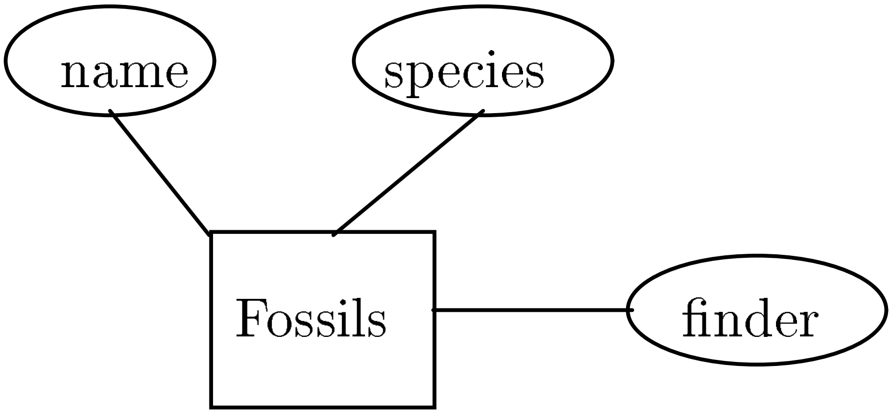
Fossils	Species	Paleontologists
Lucy	A. Afarensis	Johansson
...	...	...

- Here, OK to split, because species related only to fossil, not to paleontologist.



A Design Issue: Attributes Vs. E.S. + Relationships

What is wrong with:



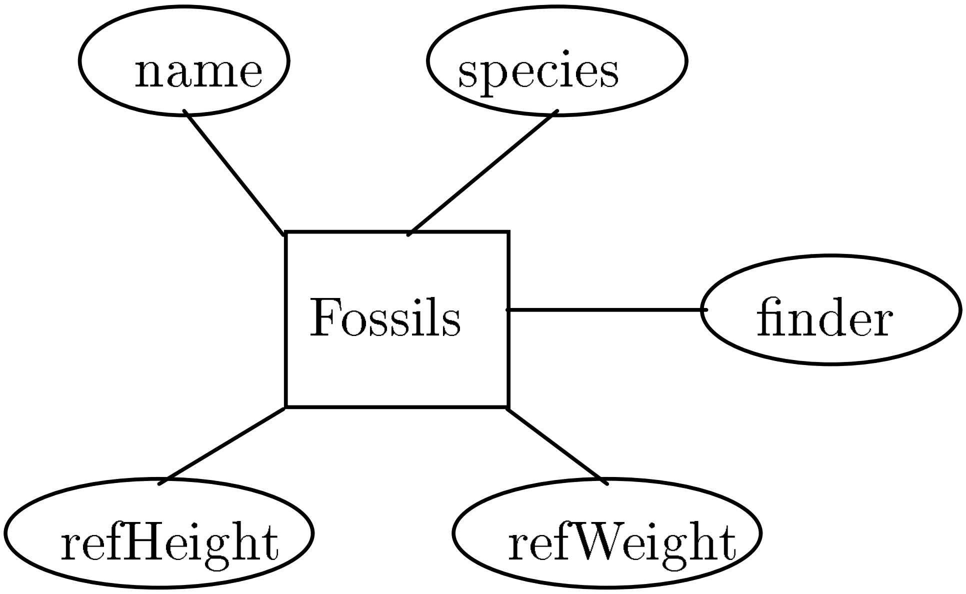
1. Makes it impossible to represent properties of species (e.g., height and weight of reference individual) or finders.

2. Difficult to represent a *set* of finders for a single fossil.

   ❖ Technically, E/R allows sets as the type of attributes, but it makes life difficult when we convert E/R to relational DB's, so avoid sets as attribute types.

- But if (1) and (2) are not a problem, e.g., we don't want to represent attributes of species, then E.S. + attributes is simplest and best.

Can We Solve (1) by Adding Attributes?



- Introduces the great problem of **Redundancy**.
  - ❖ Now for every fossil of a species, we repeat the reference height and weight.
  - ❖ Wastes space (not so important these days).
  - ❖ Offers opportunity to scr** up, e.g., change height in only one fossil among several of the same species (will remain important for the foreseeable future).