# Database Systems
# Lecture #2 E/R Model

Guifeng Zheng

School of Software, SYSU

# Agenda

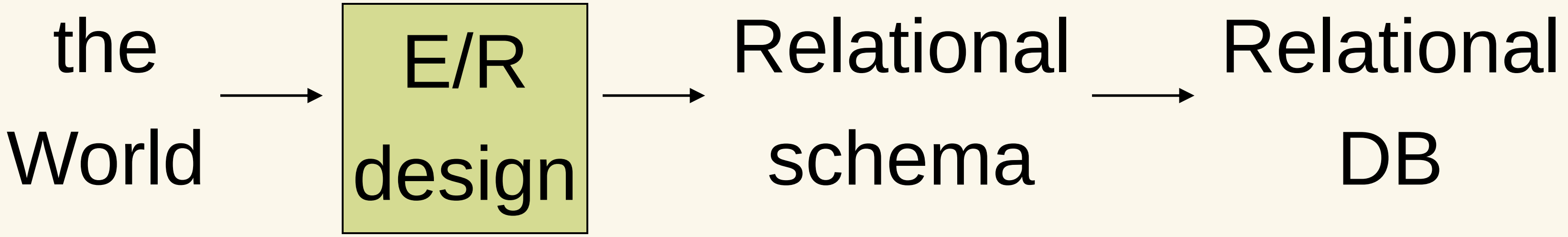
- Last time: intro
- This time: E/R model

1. Identify entity sets, relations and attributes
2. One-one, one-many, many-many relations
3. Simple ER diagrams to model a situation
4. 3-way relationships, multiple roles, subclasses

- Design issues

1. Simplicity
2. Redundancy
3. Replacing a relationships with entity sets

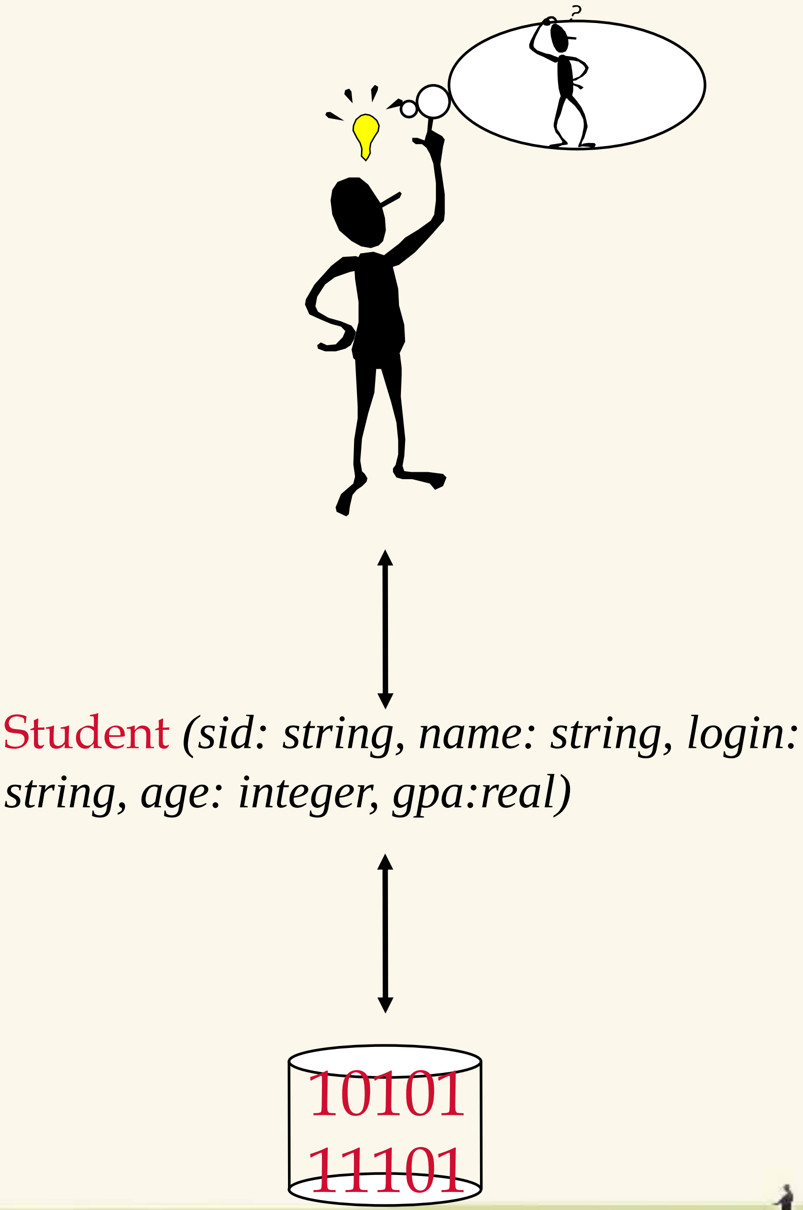
# DB development path



the World → E/R design → Relational schema → Relational DB

# Data Models

- DBMS models real world
- *Data Model* is link between user's view of the world and bits stored in computer
- Many models exist
- We will ground ourselves in the Relational Model
  - clean and common
- But use the Entity-Relationship model as a middle ground for design



Student *(sid: string, name: string, login: string, age: integer, gpa:real)*

# Entity Relationship (E/R) Model

- A popular data model – useful to database designers
    - Graphical representation of miniworld

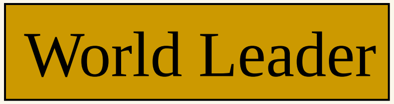
- E/R design translated to a relational design
    - then implemented in an RDBMS

- Elements of model
    - Entities
    - Entity Sets
    - Attributes
    - Relationships (!= *relations*!)

    /* 注意：联系 (!=关系) */

# E/R Model: Entity Sets

- Entity: like an object
  - Particular instance of a concept
- *Entity set*: set of one sort of entities or a concept
  - All with same attributes
- Represented by a rectangle:


World Leader

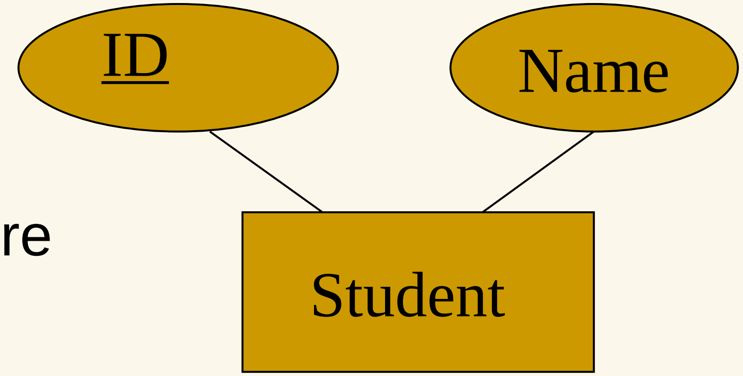
- A “good” entity set
  - Common properties
  - Correspond to class of phys. or bus. objects
    - E.g., Employees, products, accounts, grades, campaigns, etc.

# E/R Model: Attributes

```
struct student
{
  int   id;
  char*   name;
};
```

- Properties of entities in entity set
  - Like fields in a struct
  - Like columns in a table/spreadsheet
  - Like data members in an object
- Values in some domain (e.g., ints, strings)

- Represented by ovals:
- Assumed atomic
  - But could have limited structure
  - ints, strings, etc.



- Each entity set has a *key (underlined attribute)*.

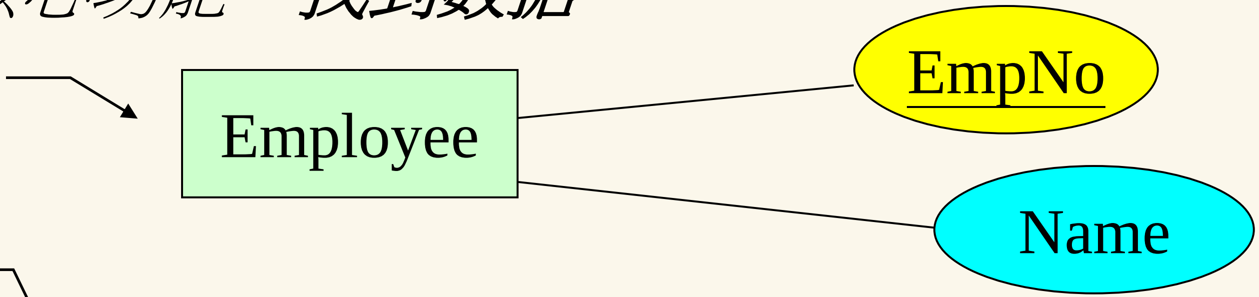
# Key Attributes

Super Key

- A set of attributes that can uniquely identify an entity (唯一标识一个实体)

*DB的核心功能：**找到数据***

ERD

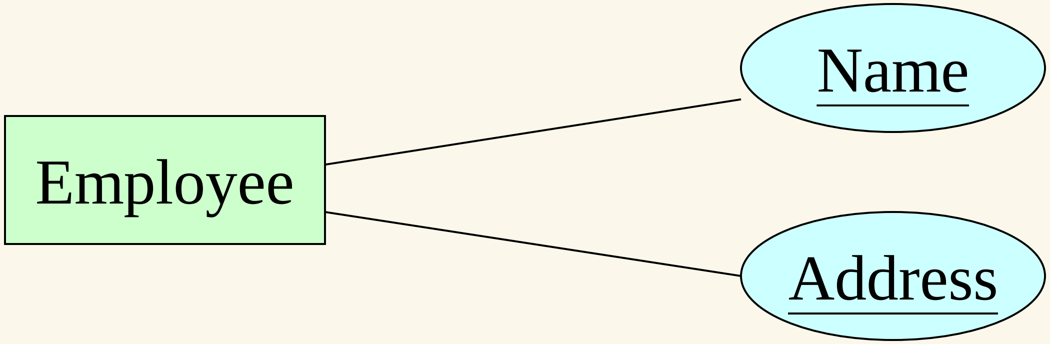


tabular

| EmpNo | Name | . . . |
|---|---|---|
| 123456 | John Wong | . . . |
| 456789 | Mary Cheung | . . . |
| 146777 | John Wong | . . . |

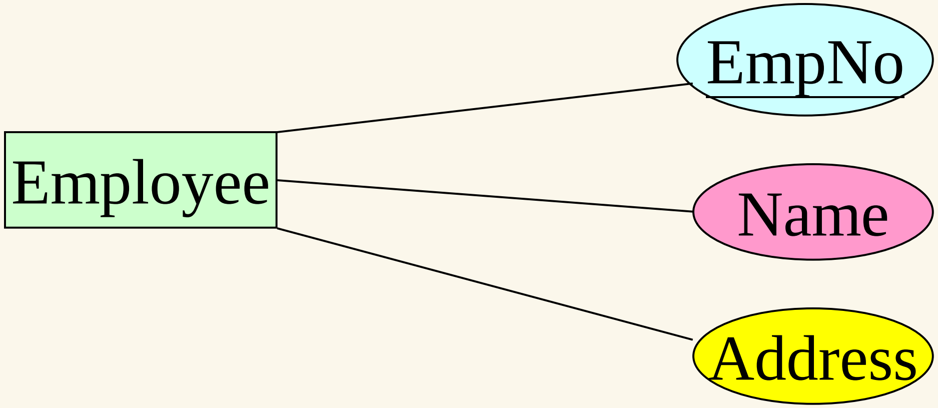
# Key Attributes

- Composite key: *Name* or *Address* alone cannot uniquely identify an employee, but together they can!



# Key Attributes

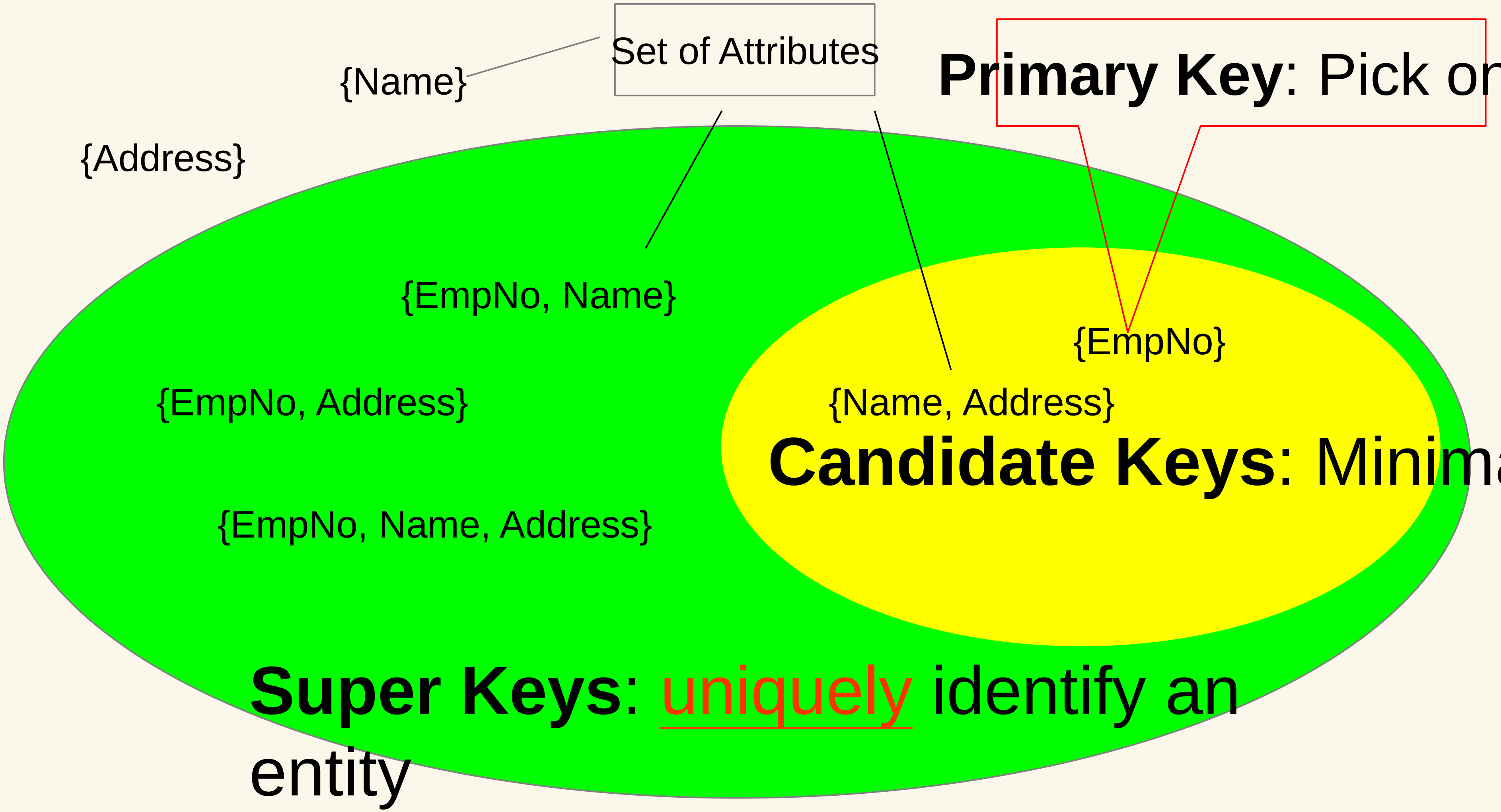
- An entity may have more than one key
  - e.g., EmpNo, (Name, Address)
  - only one is selected as the key. (sometimes called the Primary key)



In many cases, a key is artificially introduced (e.g., EmpNo) to make applications more efficient.

Question: does a desk has a key?

# Overview of Various Kinds of Keys



Set of Attributes

{Name}

**Primary Key**: Pick or

{Address}

{EmpNo, Name}

{EmpNo}

{EmpNo, Address}

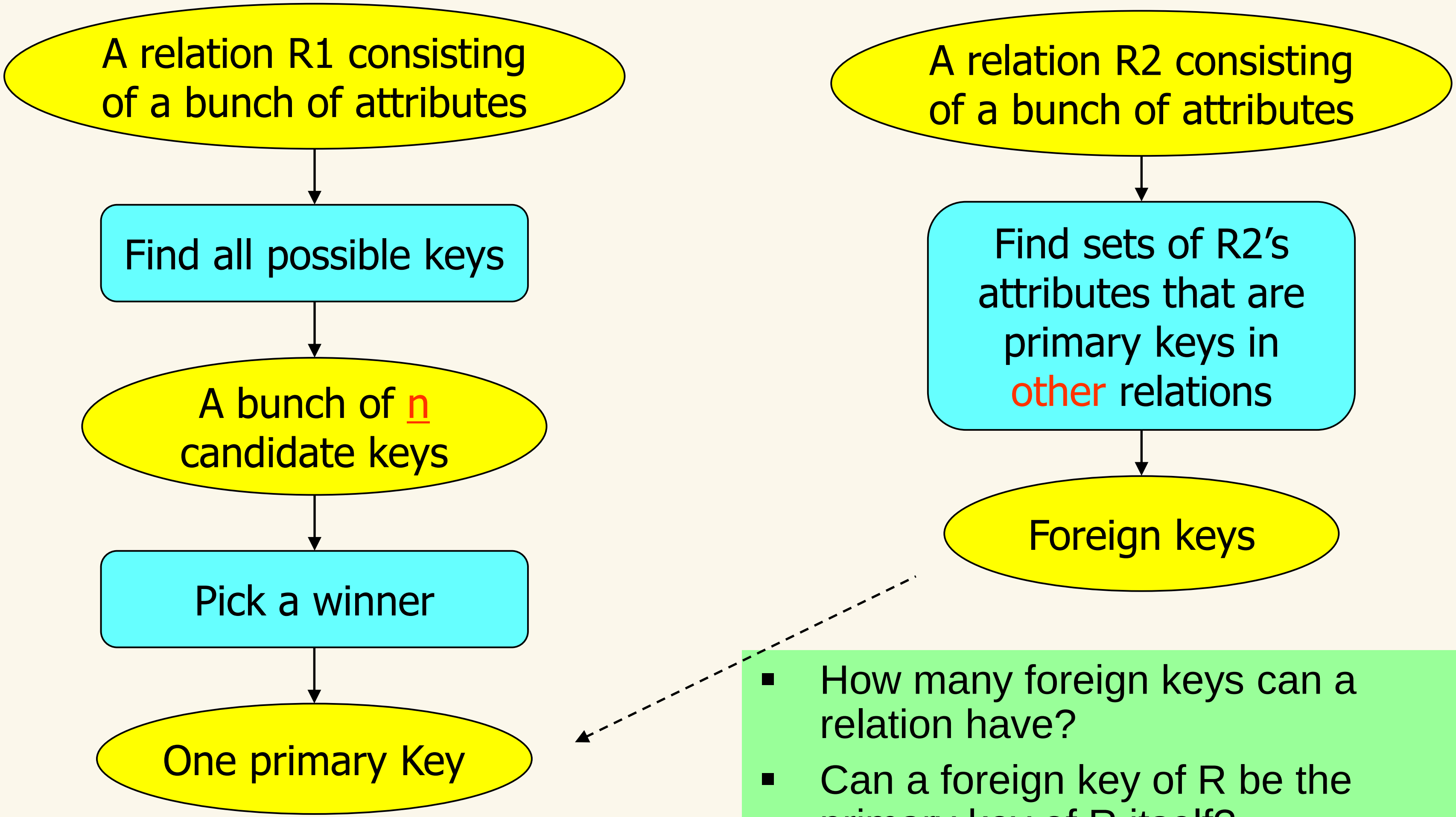
{Name, Address}

**Candidate Keys**: Minima

{EmpNo, Name, Address}

**Super Keys**: uniquely identify an entity

# Overview of Various Kinds of Keys



A relation R1 consisting of a bunch of attributes

Find all possible keys

A bunch of n candidate keys

Pick a winner

One primary Key

A relation R2 consisting of a bunch of attributes

Find sets of R2's attributes that are primary keys in other relations

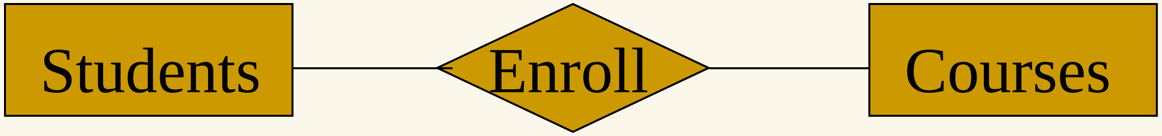
Foreign keys

- How many foreign keys can a relation have?
- Can a foreign key of R be the primary key of R itself?
- Is a primary key still a candidate key?

# E/R Model: Relationships

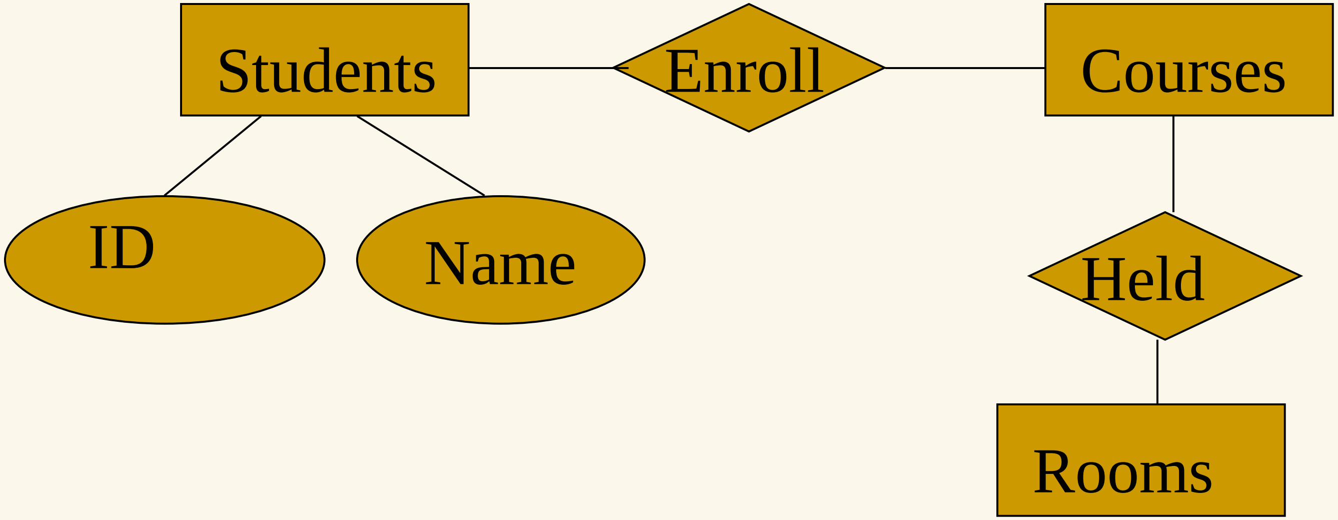
- Connect two or more entity sets
  - e.g. students *enroll* in courses
  - Binary relationships: connect two entity sets
    - most common
  - Multiway relationships: connect several ESs

- Represented by diamonds:



# E/R Model: Relationships

- Students *Enroll* in courses
- Courses are *Held* in rooms
- The E/R data model:



# Set Theory

- Invented by Georg Cantor
  - Great 19th-C German mathematician
- Big set theory results in 1870s-1890s
- Controversial at the time
  - Kronecker: “humbug”
  - First rigorous math of the “actual infinite”
- we’ll mostly deal with *finite* sets

# A little set theory

- A mathematical *set* is a collection of *members*
- A set is defined by its members
  - "Are you in or are you out?"
  - No other structure, no order, no duplicates allowed

- Sets specified by listing:
  - {1, 2, 3, …} = **N**
  - {1, 2, George Bush} (tho usually homogeneous sets in DBMS…)

- Or by "set-builder" notation:
  - { x in **N**: 2 divides x} = ?
  - { x in Presidents | reelected(x)} = ?
  - {2x: x in **N**} = ?

# A little set theory

- One set can be a *subset* of another (which is a *superset* of it)
  - ReelectedPresidents is a subset of Presidents
  - Also, RP is a *proper subset* (真子集) of Pres – some lost reelection

- Given two sets X and Y, the *cross product* or *Cartesian product* is

  X x Y = {(x,y): x in X, y in Y}
  = the set of all *ordered pairs*

- Important: (x,y) != {x,y}
- In an order pair or tuple
  - Order matters; duplicates are allowed

# A little set theory

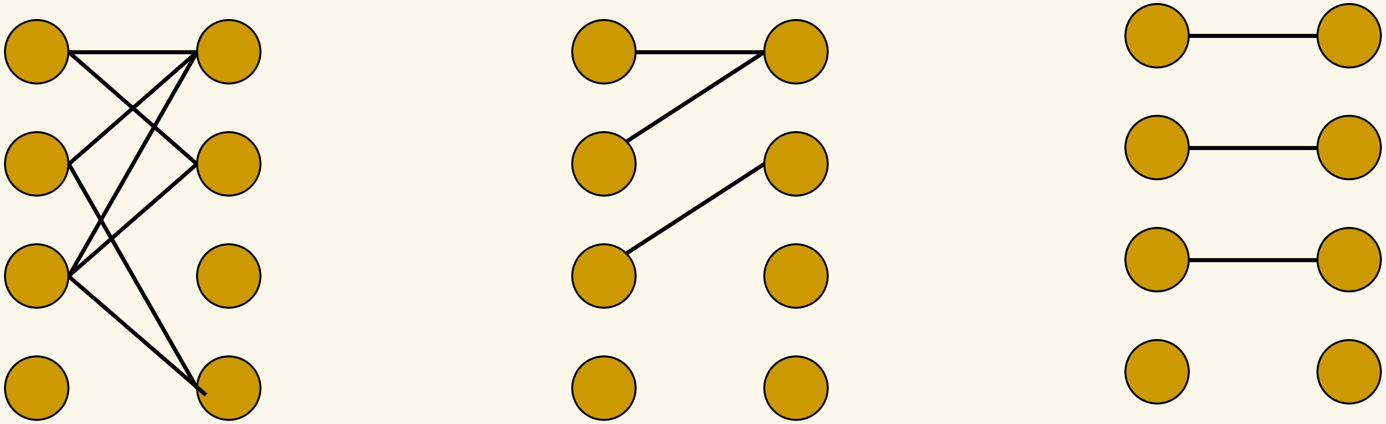
- Mathematically, a *relation* between X and Y is just a *subset of X x Y* = all those pairs (x,y) s.t. x is related to y

- Example: owner-of O on People, Cats
  - O(MPJ, Gödel the Cat) holds

- The equals relation E on N, N:
  - E(3,3) holds because 3 = 3
  - E(3,4) does not hold
  - E is still a set: E = {(1,1), (2,2), (3,3), …}

- Father-of relation F on People, People:
  - F(GHWB, GWB) holds
  - F(GWB, GHWB) does not hold
  - → Relations aren't necessarily *symmetric*

# Multiplicity of Relationships



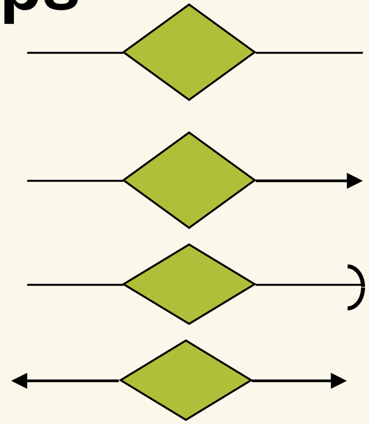
Many-many

Many-one

One-one

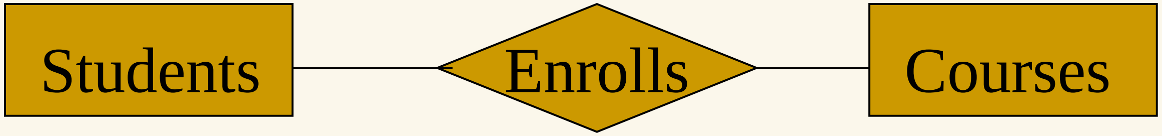
**Representation of relationships**

- No arrow: many-to-many
- Sharp arrow: many-to-one
- Rounded arrow: “exactly one”
  - “key constraint”
- One-one:

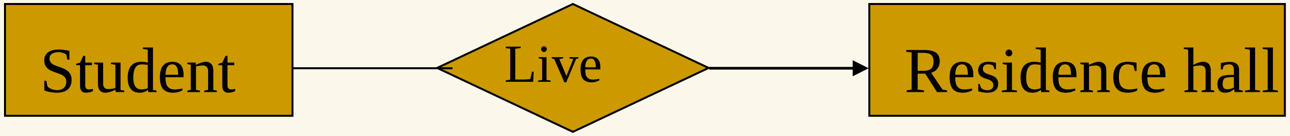


# Multiplicity of Relationships

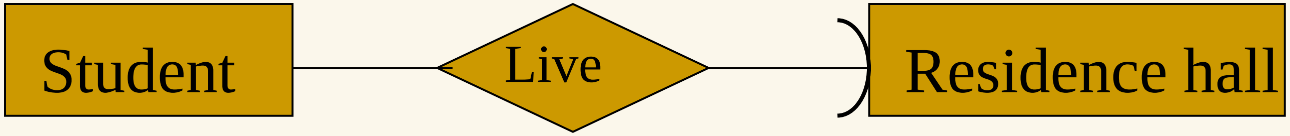
Many-to-many:



Many-to-one: a student living in a residence hall



Many-to-exactly-one: a student must live in a residence hall



# Multiplicity, set-theoretically

- Assume *no vars below are equal*

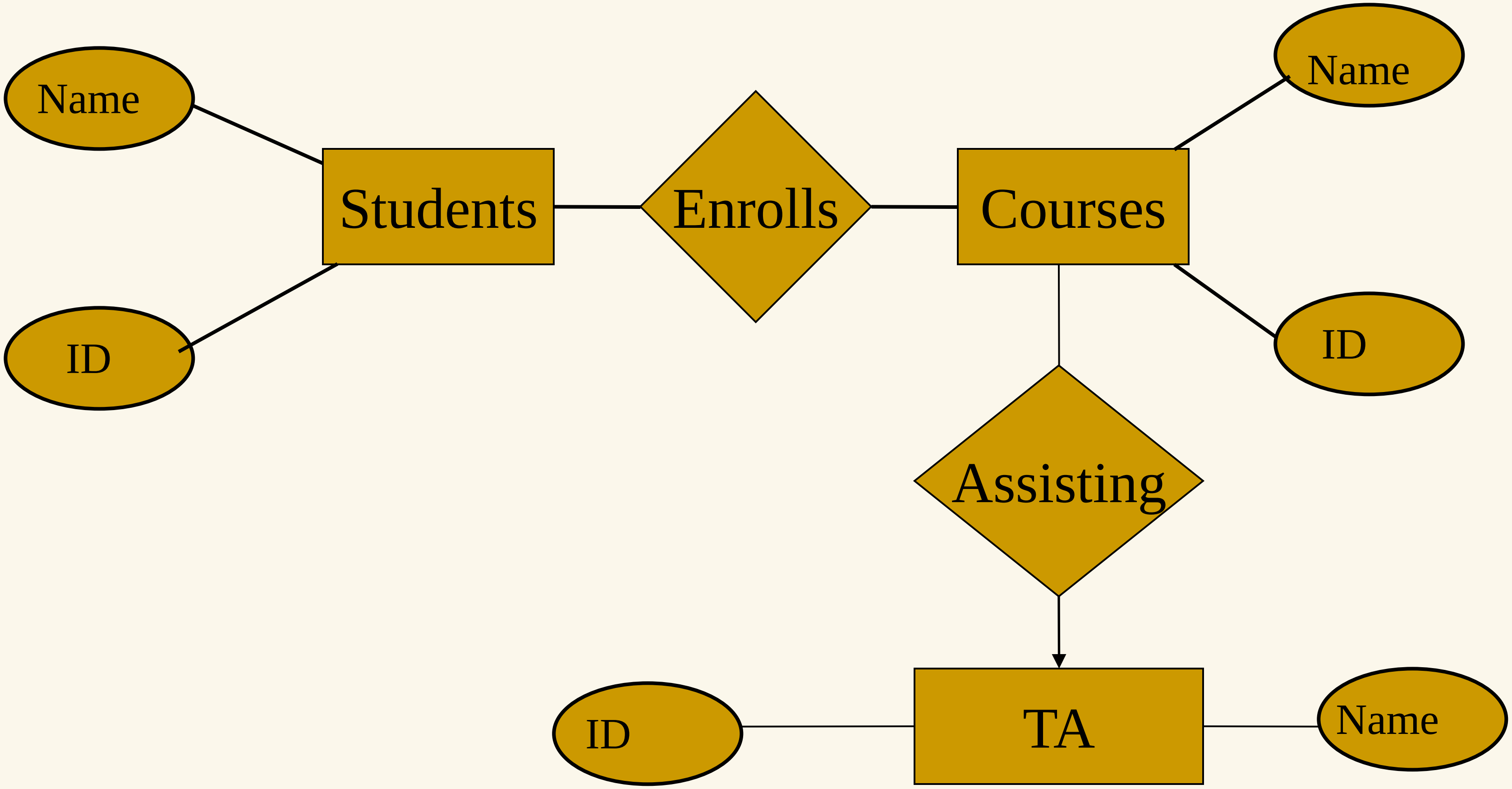
- Many-one means:
  - if (x1,y1) in R then (x1,y2) cannot be in R
- One-many means:
  - (Y,X) is many-one
- One-one means:
  - if (x1,y1) in R, then *neither* (x2,y1) nor (x1,y2) can be in R

- Notice: one-one is stronger than many-one
- One-one *implies* both many-one and one-many

# Review

- Multiplicity review:
  - Square-of? (e.g.?)
  - Cube-of? (e.g.?)

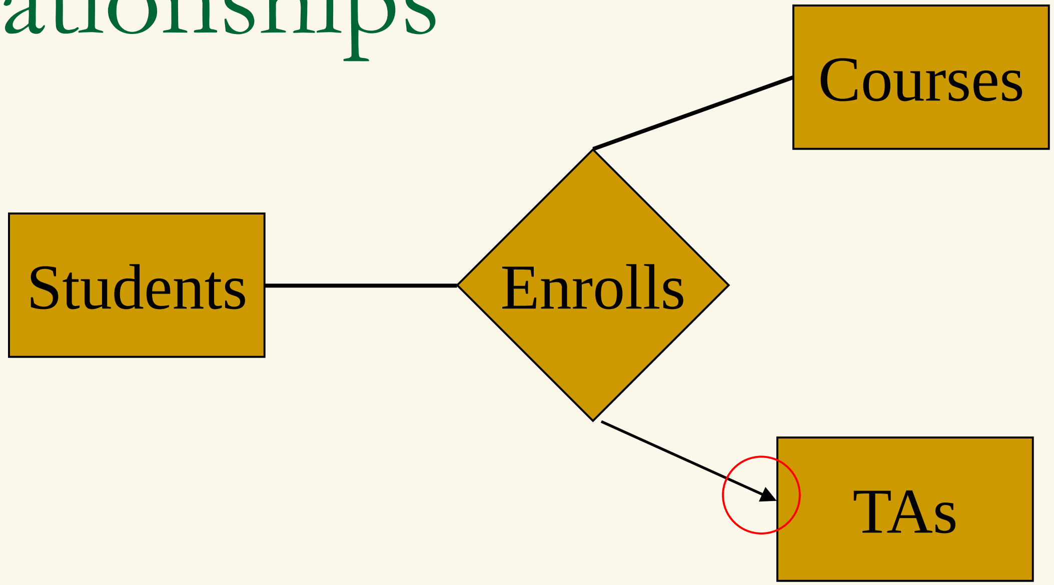
# E/R Diagram


Name
Students
ID
Enrolls
Courses
Name
ID
Assisting
ID
TA
Name

# E/R Diagrams

- OK if each TA is a TA of all students
  - Student and TA connected only through Course
- But what if students were divided among multiple TAs?
  - Then a student in SE-304 would be related to only one of the TA's for SE-304—which one?
  - Schema doesn’t store enough info
- 3-way relationship is helpful here

# Multiway Relationships


Courses

Students

Enrolls

TAs

Enrolls entries:

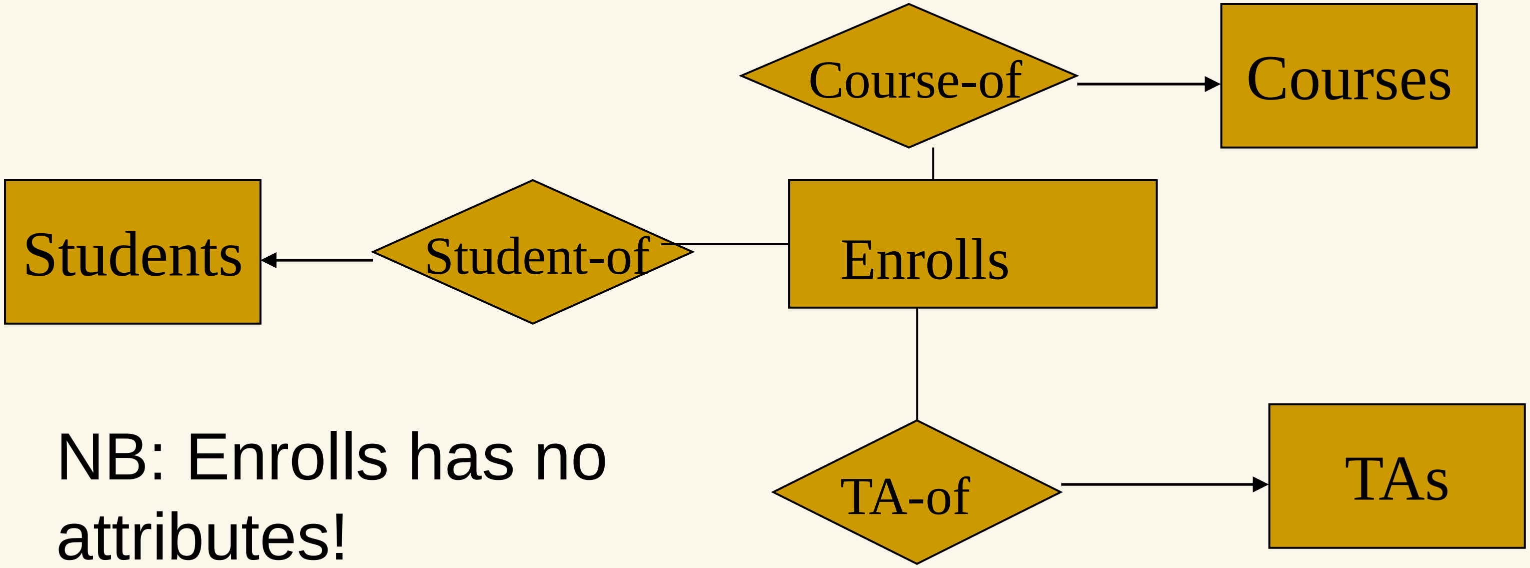
| Student | Course | TA |
|---|---|---|
| John | SE-304 | Chen |
| Mary | SE-304 | Li |
| Alice | SE-304 | Zhang |
| Mary | SE-304 | Wang |
| ... | ... | ... |

NB: *Enrolls* determines *TA*:

(student, course) → at most one TA

# Converting multiway relships to binary

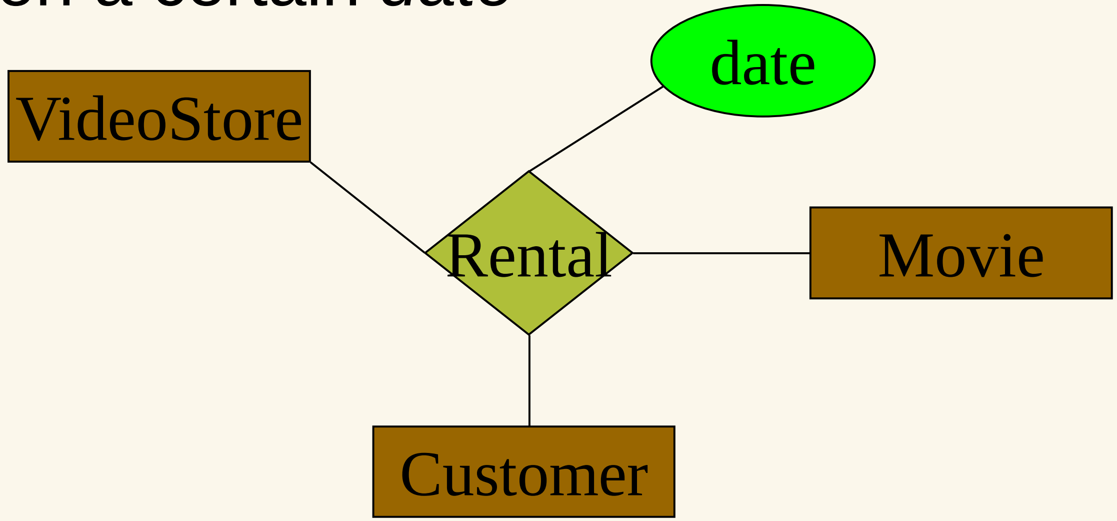
- Binary relationships are as strong as multiway
- Replace relationship with *connecting entity set* and multiple binary relationships



NB: Enrolls has no attributes!

# Second multiway e.g.: renting movies

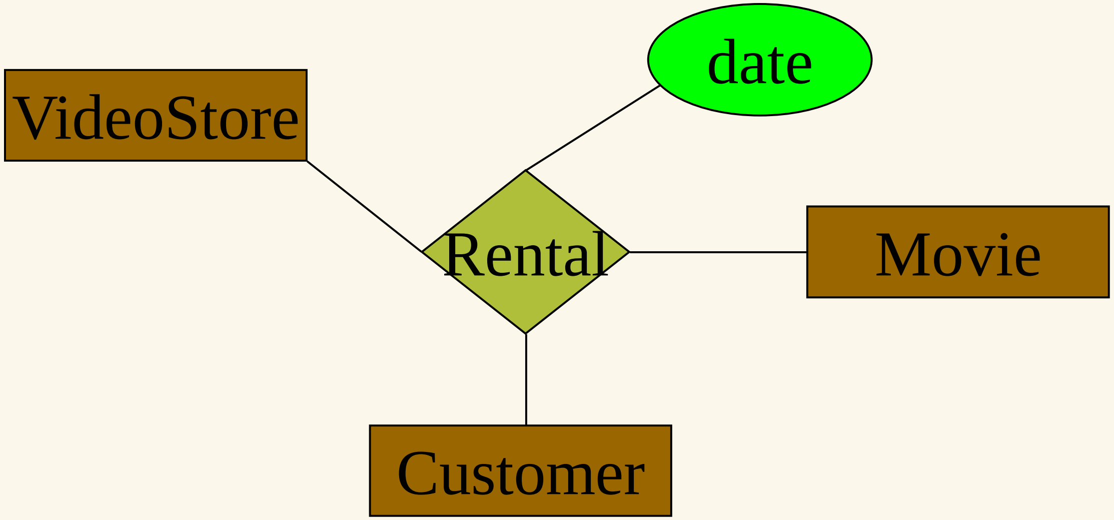
- Scenario: a *Customer Rents* a *Movie* from a *VideoStore* on a certain *date*



- date should belong to the *fact* of the renting
  - Relationship attribute

# Second multiway e.g.: renting movies

- Where can we draw arrows?



- (store, video, customer) → date ?
- (store, video, date) → customer ?
- (store, date, customer) → video ?
- (video, date, customer) → store ?

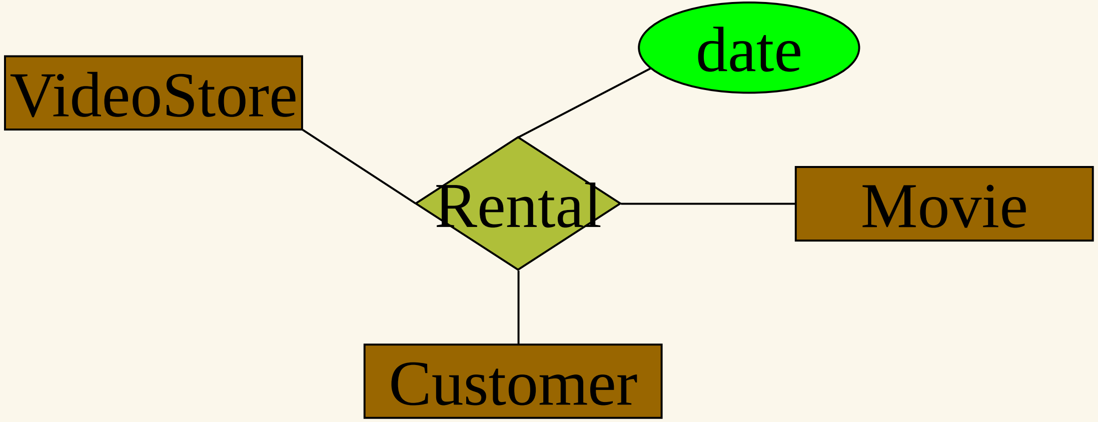
# Q: Why does it matter?

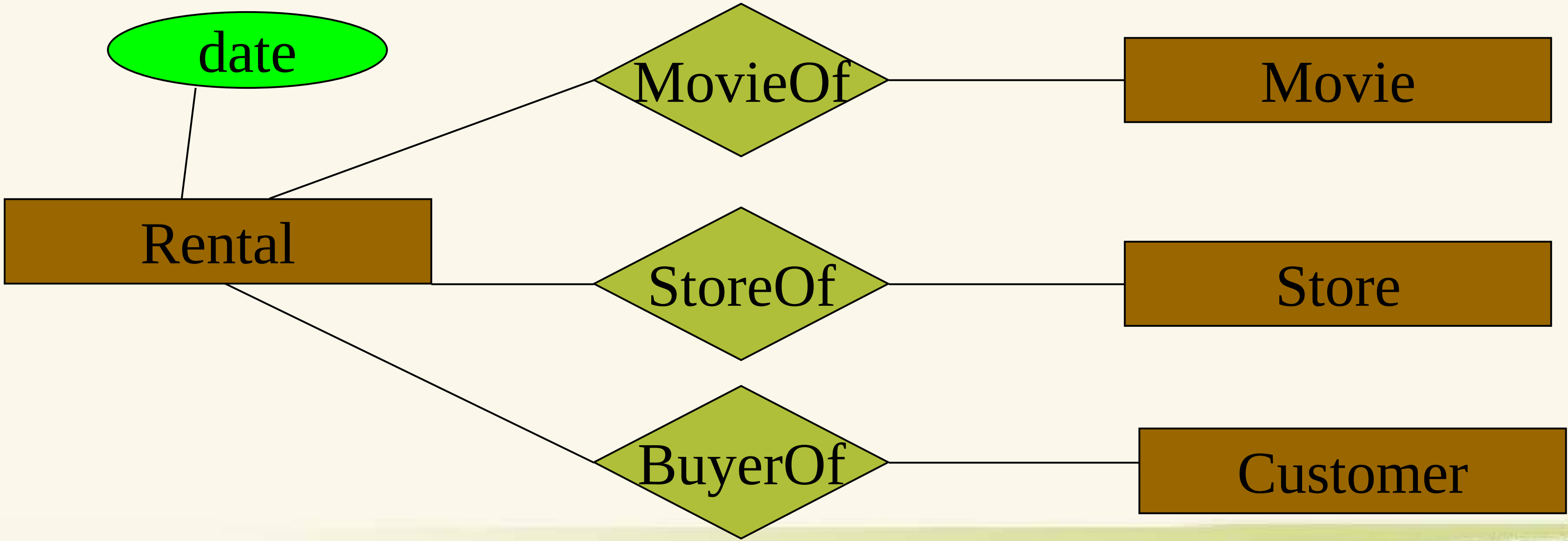
- Round arrow benefit:
  - Obvious: one item takes less space than many
  - Less obvious: easier to access one item x than set of one item {x}
    - In programming: an int v. a linked list with just one int

- Regular arrow benefit:
  - Mapping to a set of either one elm or none seems bad
  - But not implemented this way
  - *Always* one element, which may be *NULL*

# Second multiway e.g.: renting movies

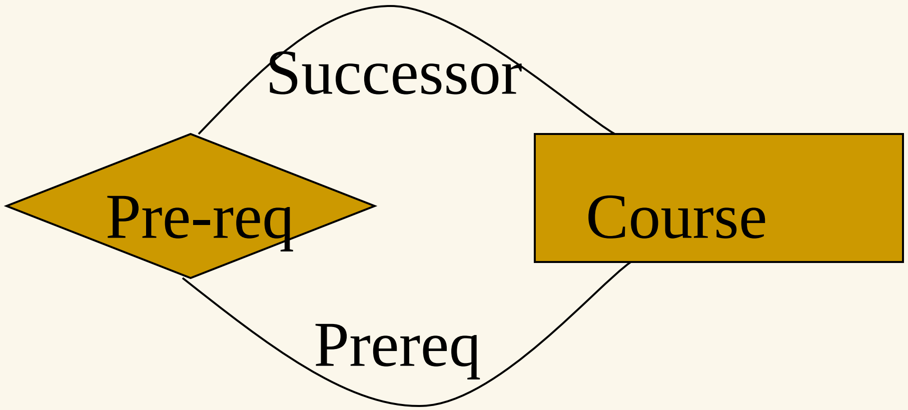
- Convert to binary?





# Roles in relationships

- Entity set appears more than once in a relship
  - Generally distinct *entities*
- Each appearance is in a different *role*
- Edges labeled by roles



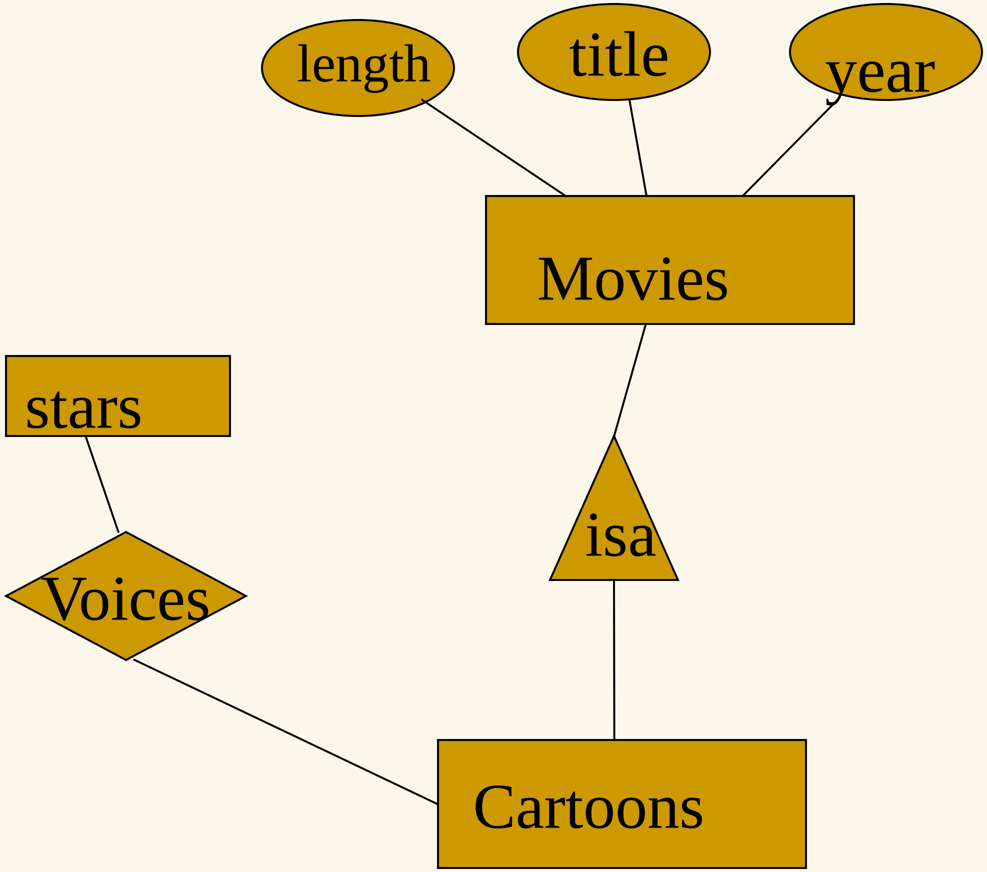
| Course (Pre-req) | Course (Successor) |
|---|---|
| Accounting | Finance-I |
| Finance-I | Derivatives |
| Finance-I | Finance-II |
| Calculus | Derivatives |

# Subclasses in the E/R model

- Some ESs are special cases of others
- Conversely: some are generalizations
  - Mammals, humans, students, grad students
  - NB: These aren’t *members* but subclasses

- Subclass A isa B
  - Represented by a triangle
  - Root is more general

# Subclasses



# New topic: Design Issues

- Faithfulness (如实、正确)
- Avoiding redundancy (避免冗余)
- Simplicity (简单性)
- Choice of relationships
- Picking elements

# Faithfulness

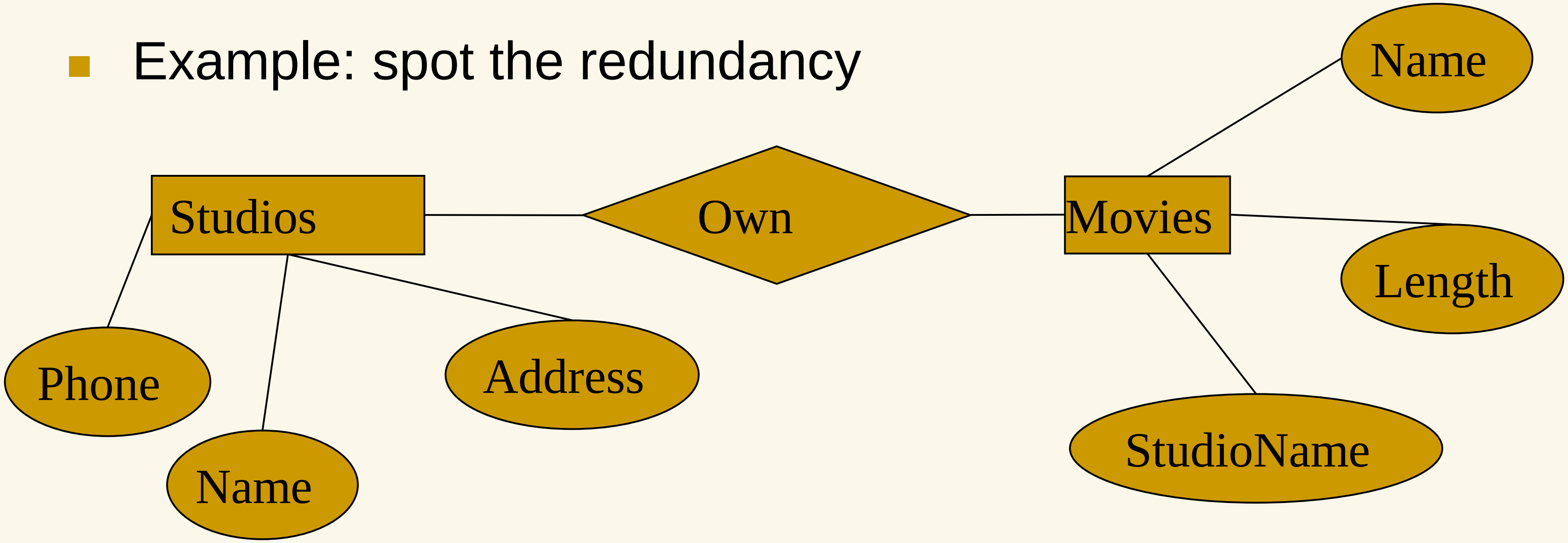
- Is the relationship many-many or many-one?
- Are the attributes appropriate?
- Are the relationships applicable to the entities?

- Examples:
  - ❑ Courses & instructors
    - maybe many-one, maybe many-many
  - ❑ Bosses & subordinates
    - maybe one-many, maybe many-many

# Avoiding redundancy

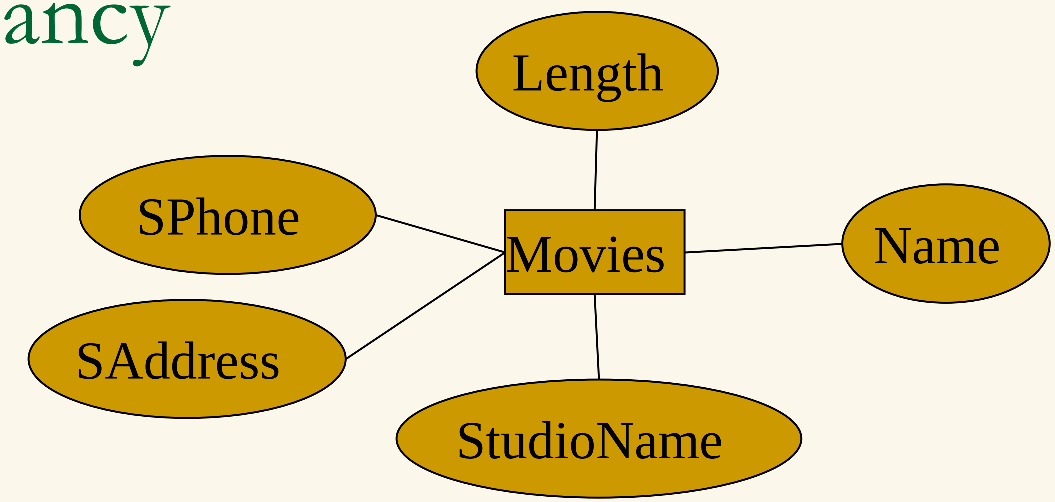
- Say everything once and only once
  - Minimize database storage requirements
  - More important: prevent possible update errors
    - One danger: modifying data one place but not the other

- Example: spot the redundancy



Redundancy: Movies “knows” the studio two ways

# Spot more redundancy



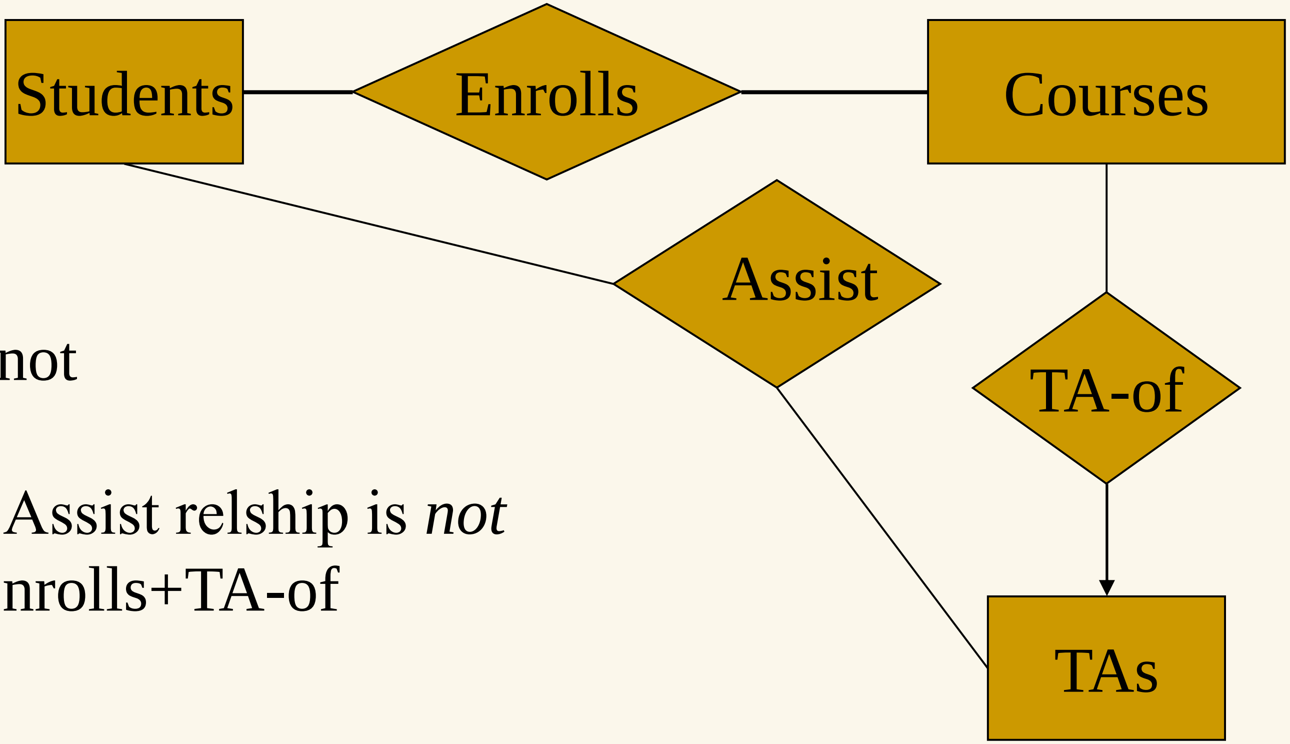
| Name | Length | Studio | SAddress | SPhone |
|---|---|---|---|---|
| Pulp Fiction | … | Miramax | NYC | 212-… |
| Sylvia | … | Miramax | NYC | 212-… |
| Jay & Sil. Bob | … | Miramax | NYC | 212-… |
| … | | | | |

Different redundancy: studio info listed for every movie!

# Don’t add relships that are implied

Suppose each course again has <=1 TA

Q: Is this good design?

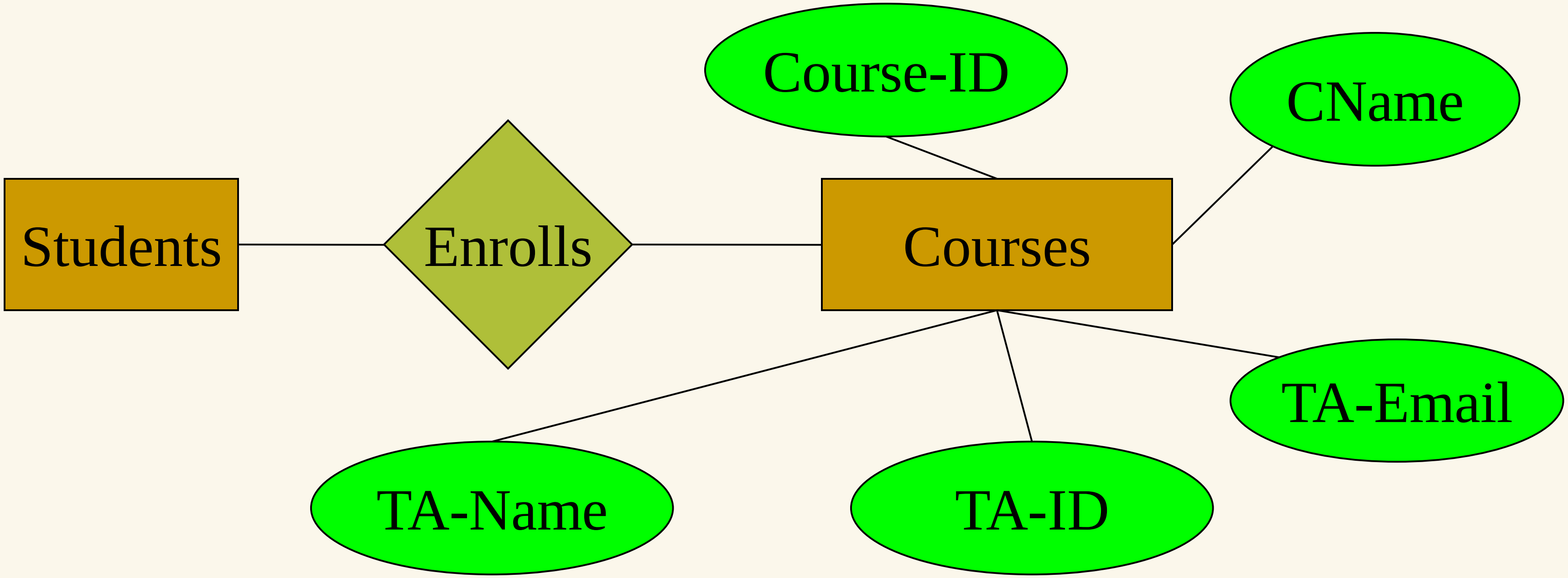


A: probably not

…unless the Assist relship is *not* implied by Enrolls+TA-of

# Still more redundancy

Q: What’s wrong with this design?



A:
- Repeating TA names & IDs – redundant
- TA is not TAing any course now ➔ lose TA’s data!
- TA should get its own ES

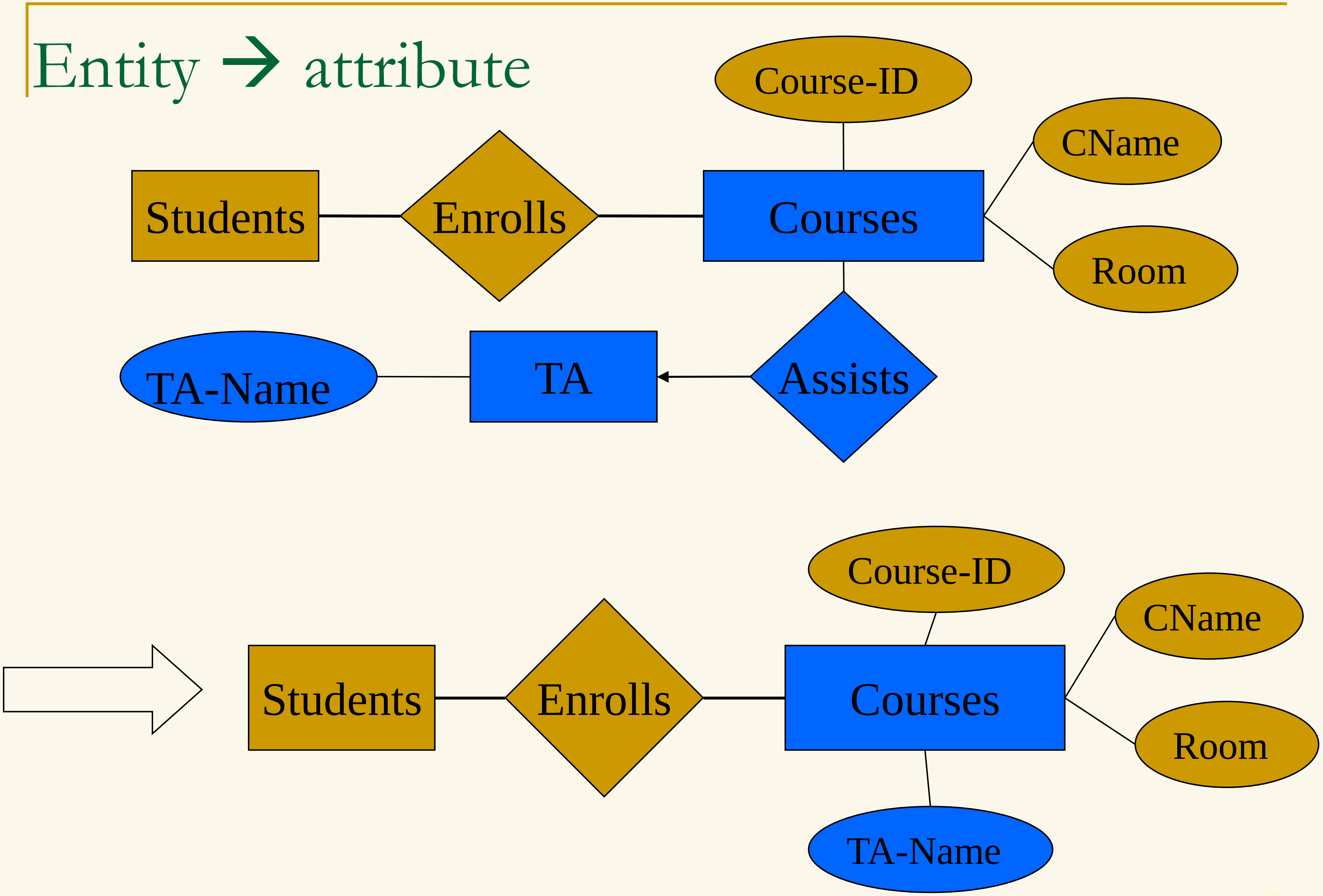
# Related issue: entity or attribute?

- Some E/Rs improved by removing entities

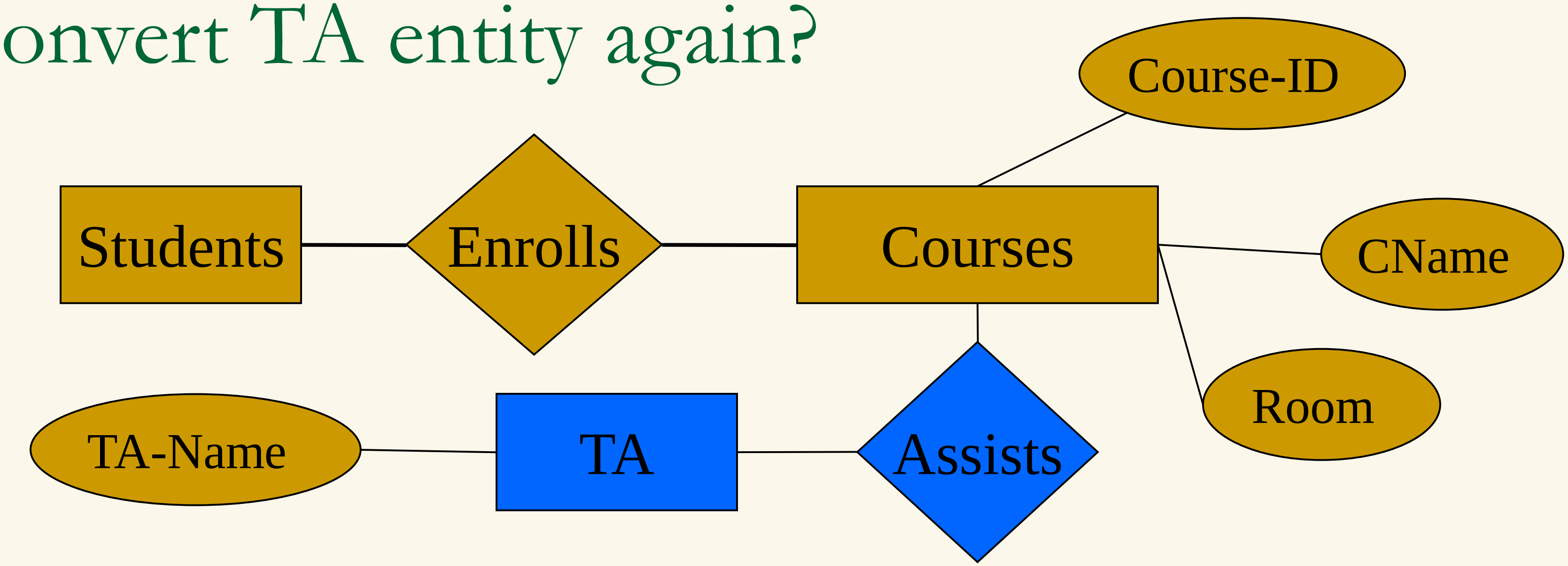
- Can convert Entity E into attributes of F if
  1. R:F→E is many-one (or 1-1)
  2. Attributes for E are *mutually independent*
     - knowing one att val doesn't tell us another att val

- Then
  - remove E
  - add all attributes of E to F

# Entity → attribute



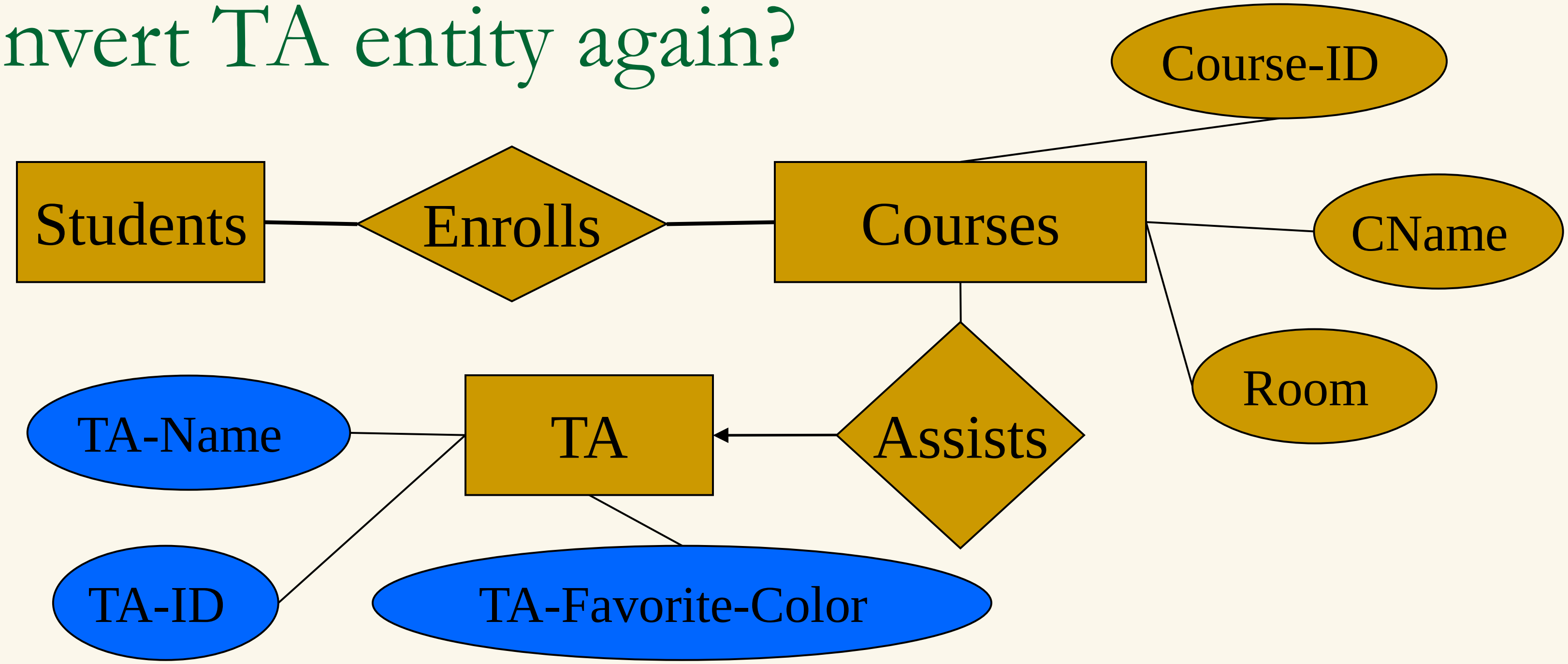
# Convert TA entity again?



- No! Multiple TAs allowed → redundant course data
- Violates *condition (1)*

| CName | CID | Room | TA-Name |
|---|---|---|---|
| DBMS | 46 | 123 | Howard |
| DBMS | 46 | 123 | Wesley |
| … | | | |

# Convert TA entity again?



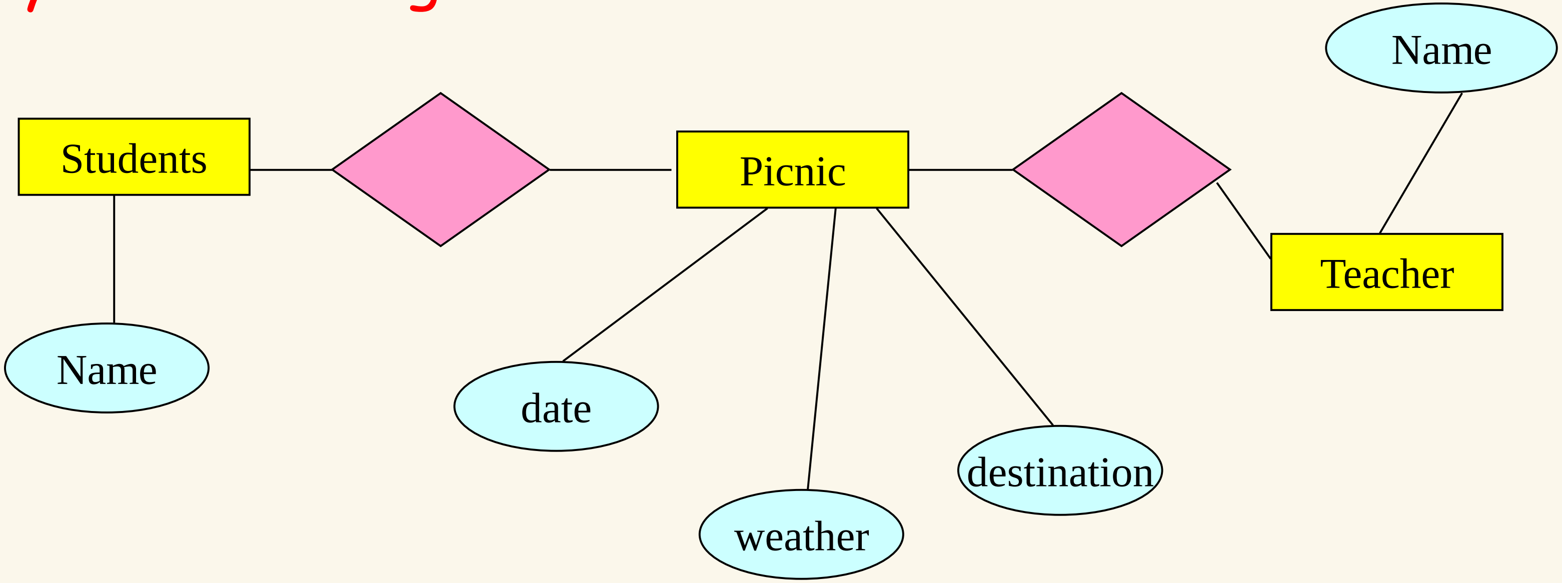
- No! TA has *dependent fields* → redundant TA data
- Violates *condition (2)*
  - How can it tell?

| CName | TA-Name | TA-ID | TA-Color |
|---|---|---|---|
| DBMS | Ralph | 678 | Green |
| A.Soft. | Ralph | 678 | Green |
| … | | | |

# A case Study

A primary school student writes a composition about a picnic:
Today is Sep 9, the weather is fine.
My classmates, John, Mary and I go to a picnic in Sai Kung.
Our teacher is Ms Wong

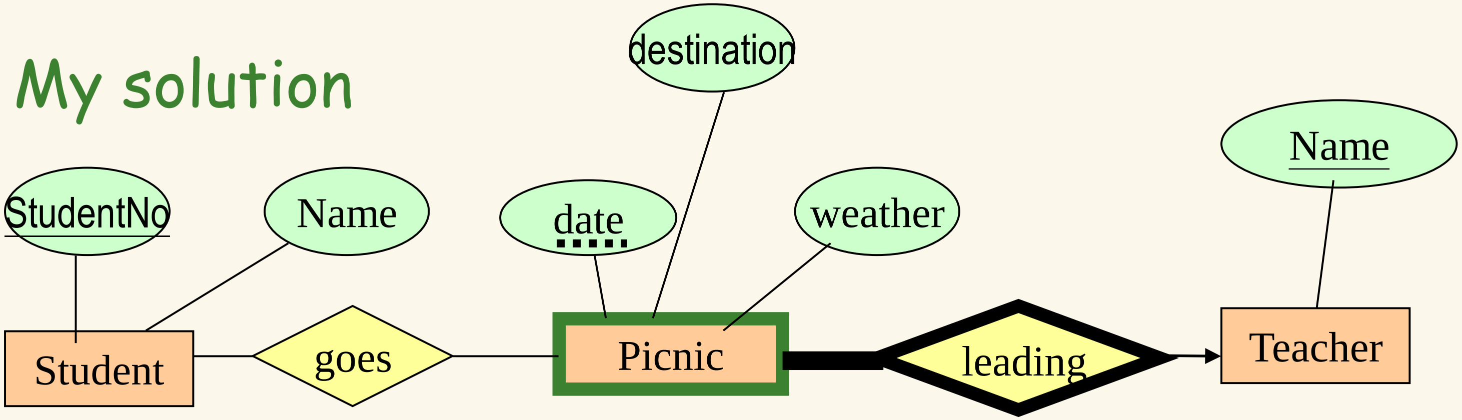
## My Initial Design:



# Questions ?

- Why “John”, “Mary”, “Miss Wong” are not in the ER diagram ?
- What do these names tell us ?
- What are the keys of Student, Picnic & Teacher ?
- What are the cardinalities of the relationships ?

# My solution

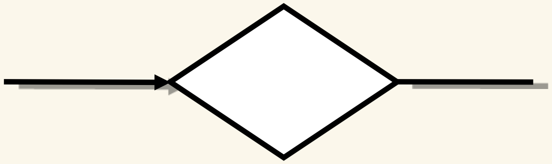


- Every student has an ID number, it is better to keep it in the database and use it as a key
- I bet that there won't be teachers with the same name; otherwise, I'll add employee number and use it as a key
- goes is N:M, why ? *A picnic has more than one student participating; also, a student can go to more than 1 picnic. However, this N:M relationship allows a student to go to more than one picnic on the same date*
- leading is N:1 , why? *Depends on your assumptions*
  - I assume a teacher can only lead 1 picnic on a certain date, so given the teacher name and the date, I can identify a picnic
- Picnic is made a weak entity. I *could* have added a PicnicNo, but it would be very awkward.

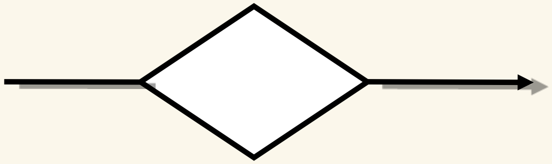
Question:
How to record number of students in a picnic?

# Textbook vs. PPT

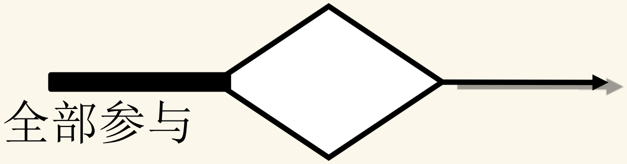
多对一



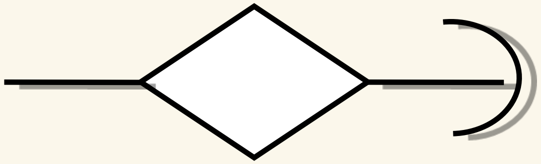
=



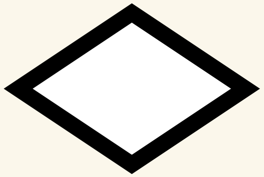
多对唯一



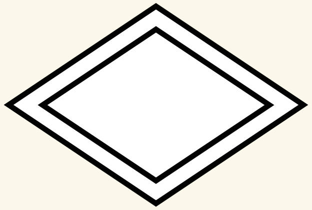
=



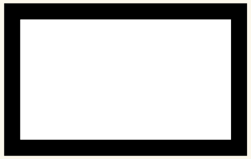
支撑联系、弱联系



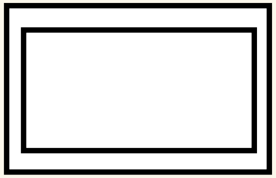
=



弱实体



=



# The End

Check course homepage for homework if any.