# Lecture 2
# Entity-Relationship Model

Eugene Wu

# Steps for a New Application

Requirements

what are you going to build?

Conceptual Database Design

pen-and-pencil description

Logical Design

formal database schema

Schema Refinement:

fix potential problems, normalization

Physical Database Design

use sample of queries to optimize for speed/storage

App/Security Design

prevent security problems

# Steps for a New Application

Requirements
- what are you going to build?

Conceptual Database Design
- pen-and-pencil description

ER Modeling

Logical Design
- formal database schema

Schema Refinement:
- fix potential problems, normalization

Physical Database Design
- use sample of queries to optimize for speed/storage

App/Security Design
- prevent security problems

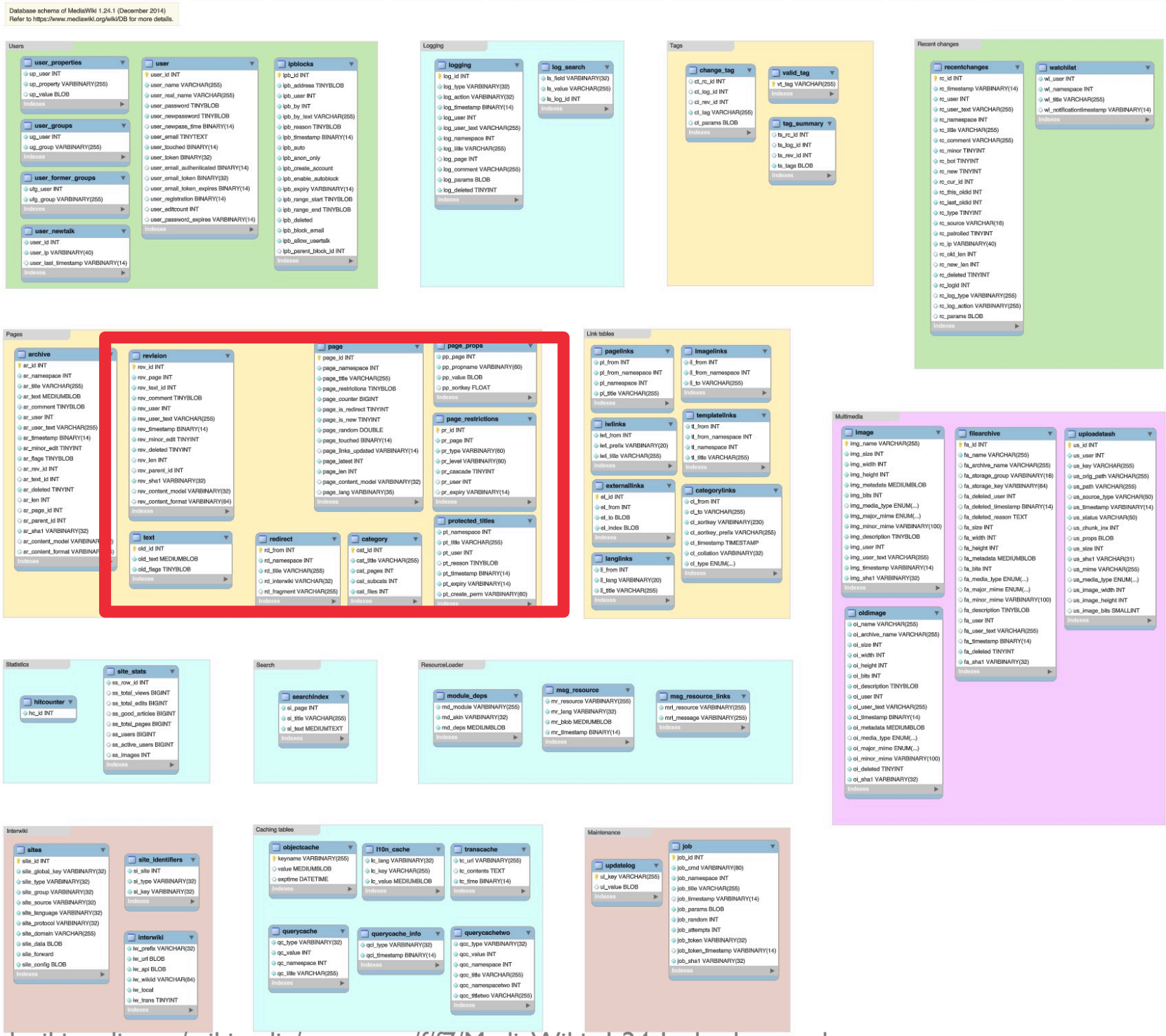
# Database Apps Are Complicated

Typical Fortune 100 Company

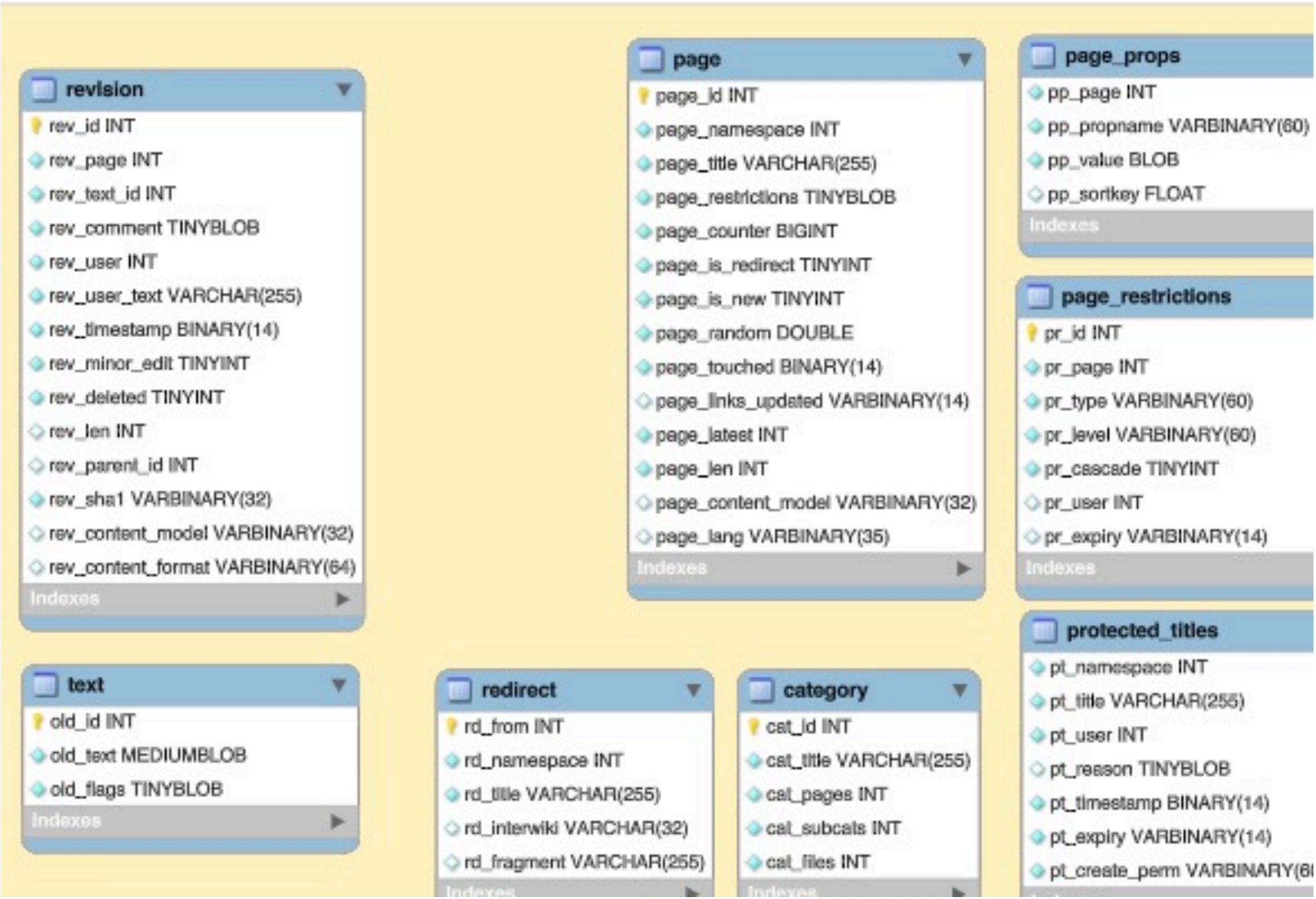
- ~10k different information (data) systems
- 90% relational databases (DBMSes)
- Typical database has >100 tables
- Typical table has 50 – 200 attributes

```
test=# \d election
                                    Table "public.election"
      Column       |  Type   | Collation | Nullable |                Default
-------------------+---------+-----------+----------+-----------------------------
 year              | integer |           |          |
 state             | text    |           |          |
 state_po          | text    |           |          |
 state_fips        | integer |           |          |
 state_cen         | integer |           |          |
 state_ic          | integer |           |          |
 office            | text    |           |          |
 candidate         | text    |           |          |
 party_detailed    | text    |           |          |
 writein           | text    |           |          |
 candidatevotes    | integer |           |          |
 totalvotes        | integer |           |          |
 version           | integer |           |          |
 notes             | text    |           |          |
 party_simplified  | text    |           |          |
 id                | integer |           | not null | nextval('election_id_seq'::
Indexes:
    "election_id_key" UNIQUE CONSTRAINT, btree (id)

test=# \d food
                     Table "public.food"
      Column       |  Type   | Collation | Nullable | Default
-------------------+---------+-----------+----------+---------
 camis             | integer |           |          |
 dba               | text    |           |          |
 boro              | text    |           |          |
 building          | integer |           |          |
 street            | text    |           |          |
 zipcode           | integer |           |          |
 phone             | bigint  |           |          |
 inspection_date   | date    |           |          |
 action            | text    |           |          |
 score             | integer |           |          |
 grade             | text    |           |          |
 inspection_type   | text    |           |          |
 census_tract      | integer |           |          |
 year              | integer |           |          |
 month             | integer |           |          |
 day               | integer |           |          |
```

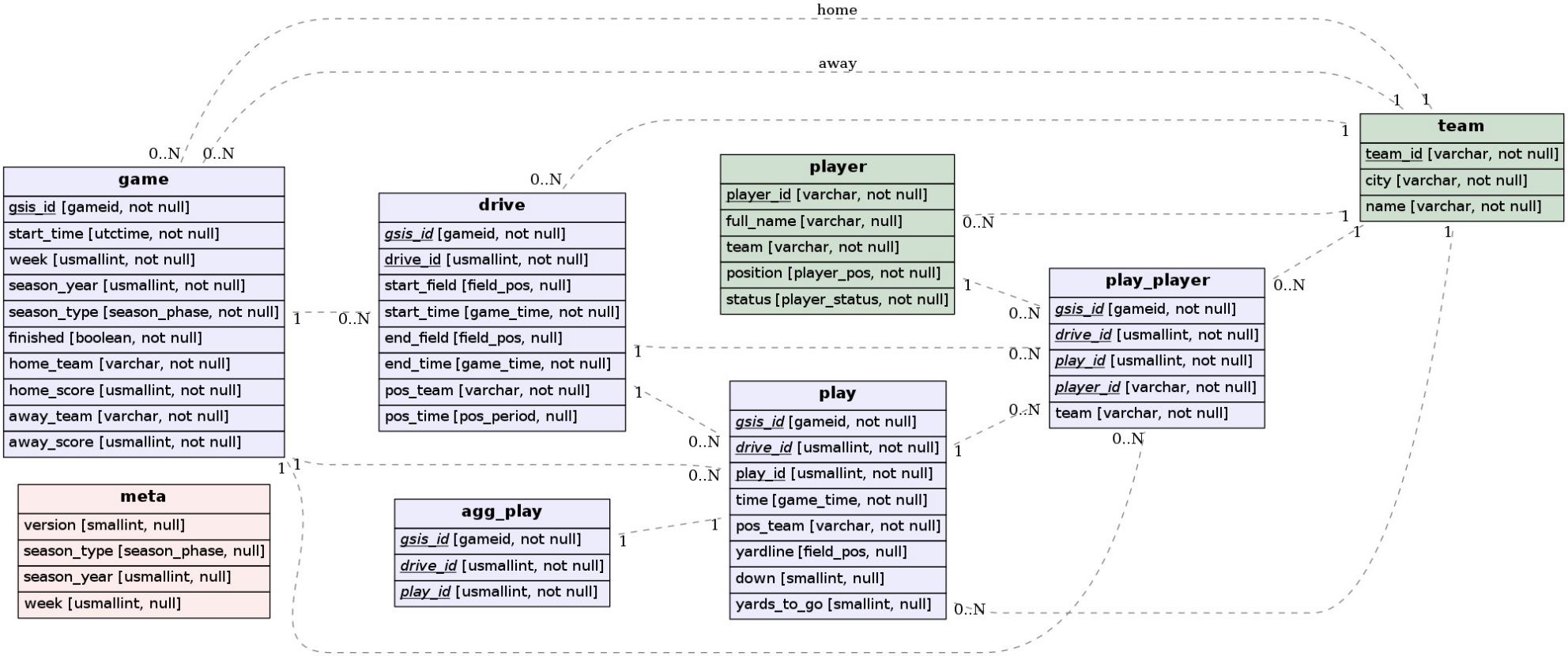



https://upload.wikimedia.org/wikipedia/commons/f/f7/MediaWiki_1.24.1_database_schema.svg

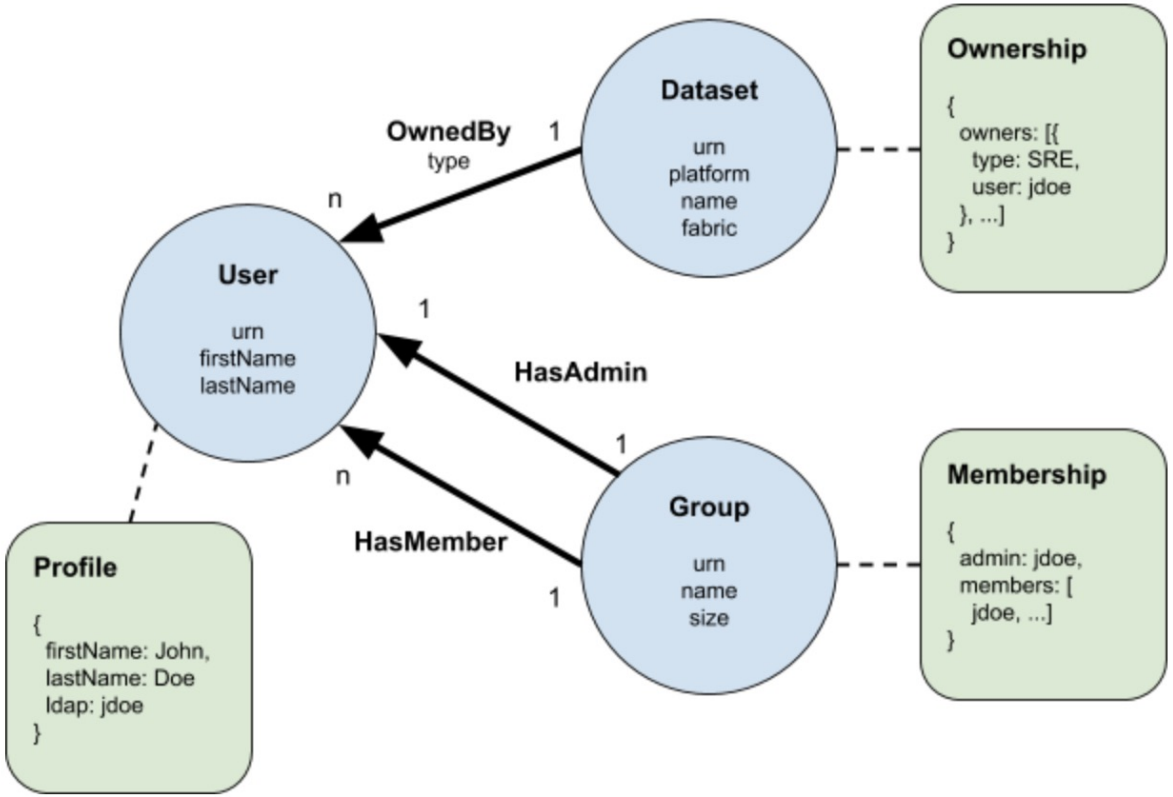



https://upload.wikimedia.org/wikipedia/commons/f/f7/MediaWiki_1.24.1_database_schema.svg

nfldb Entity-Relationship diagram (condensed)




https://github.com/BurntSushi/nfldb/wiki/The-data-model#er-diagrams



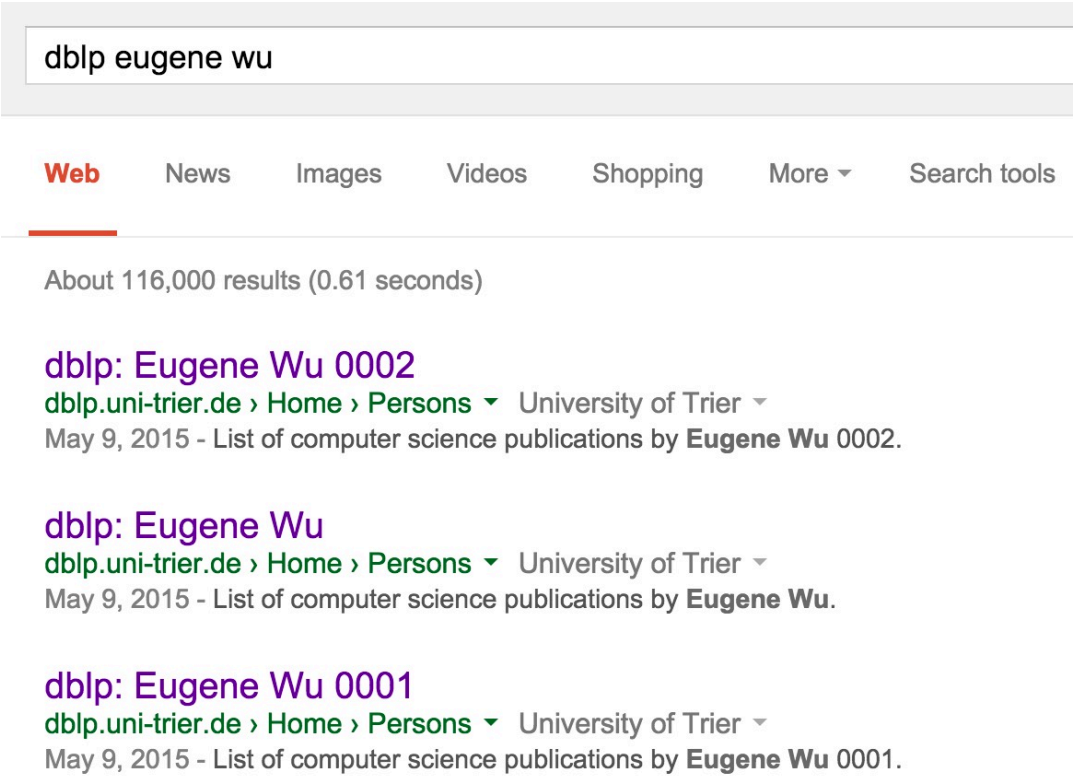

https://engineering.linkedin.com/blog/2019/data-hub

# Inconsistencies/Constraint Violations

*Huge* amount of effort to avoid inconsistencies

Can data model help us avoid automatically?

DBLP is *the* site for computer science publications



# Inconsistencies/Constraint Violations

[–] **2010 – today** [+] **Refine list**

**2014**

[j8] Eugene Wu, Leilani Battle, Samuel R. Madden:
**The Case for Data Visualization Management Systems.** PVLDB 7(10): 903-906 (2014)

[j7] Alekh Jindal, Praynaa Rawlani, Eugene Wu, Samuel Madden, Amol Deshpande, Mike Stonebraker:
**VERTEXICA: Your Relational Friend for Graph Analytics!** PVLDB 7(13): 1669-1672 (2014)

$\neq$

[–] **1990 – 1999** [+] **Refine list**

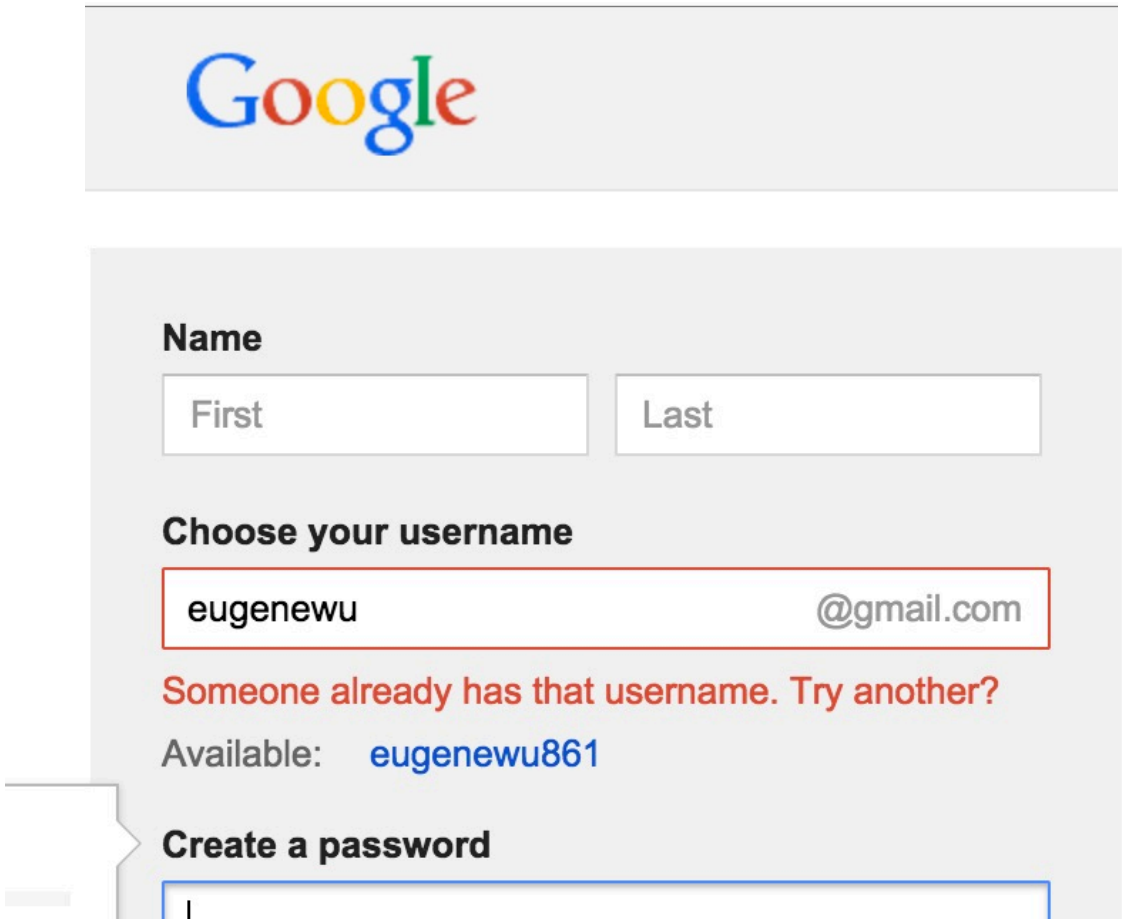
**1994**

[c2] James Hwang, Eugene Wu, Alan Bell, Andy Cordell, LeBarian Stokes, Scott Hankins:
**Design of a SPDM-Like Robotic Manipulator System for Space Station on Orbit Replaceable Unit Ground Testing - An Overview of the System Architecture.** ICRA 1994: 1286-1291

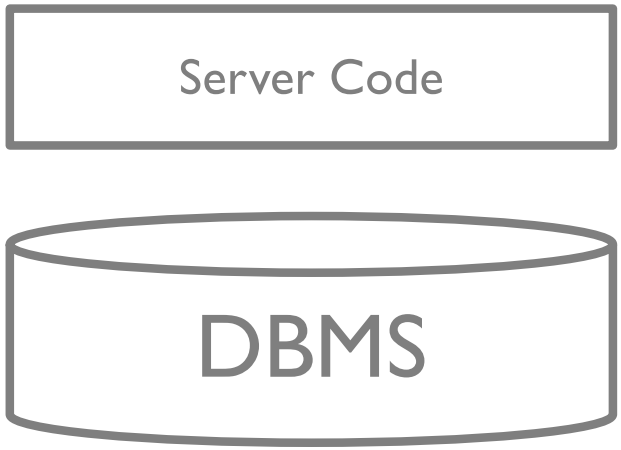
[c1] Eugene Wu, James Hwang, Scott Hankins:
**Design of the Control System for a Robotic Manipulator for Space Station On-Orbit Replaceable Unit Ground Testing.** ICRA 1994: 1415-1420

# Inconsistencies/Constraint Violations

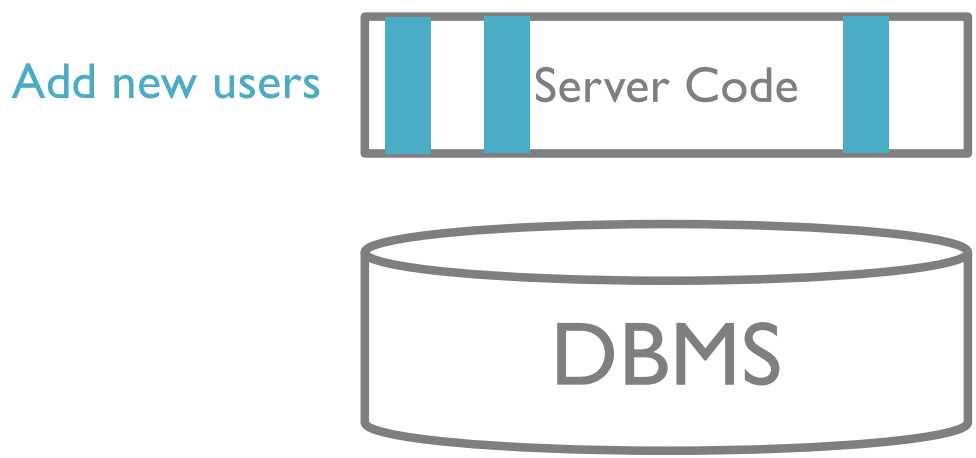
Check in application code!



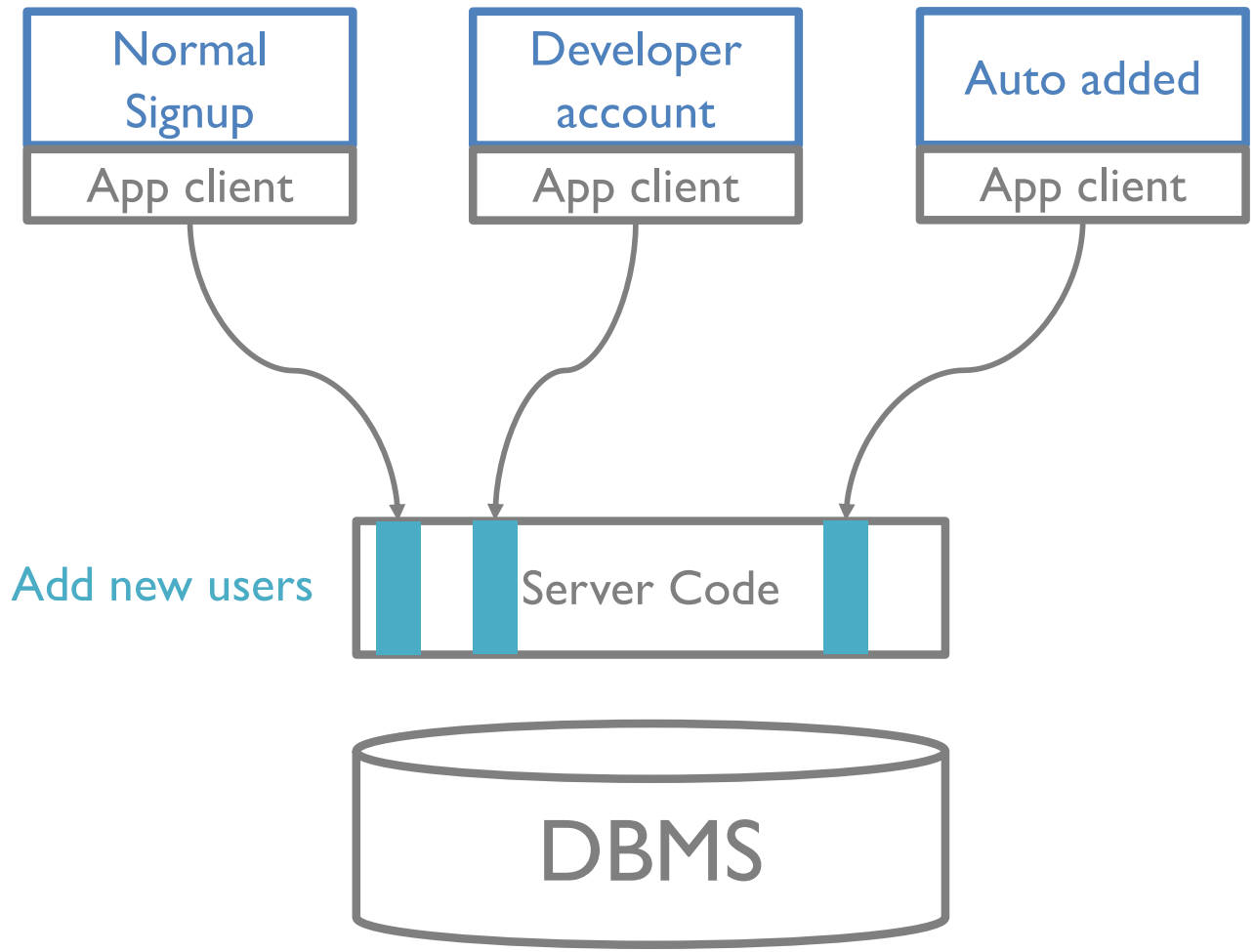
# It is Hard to Design Applications



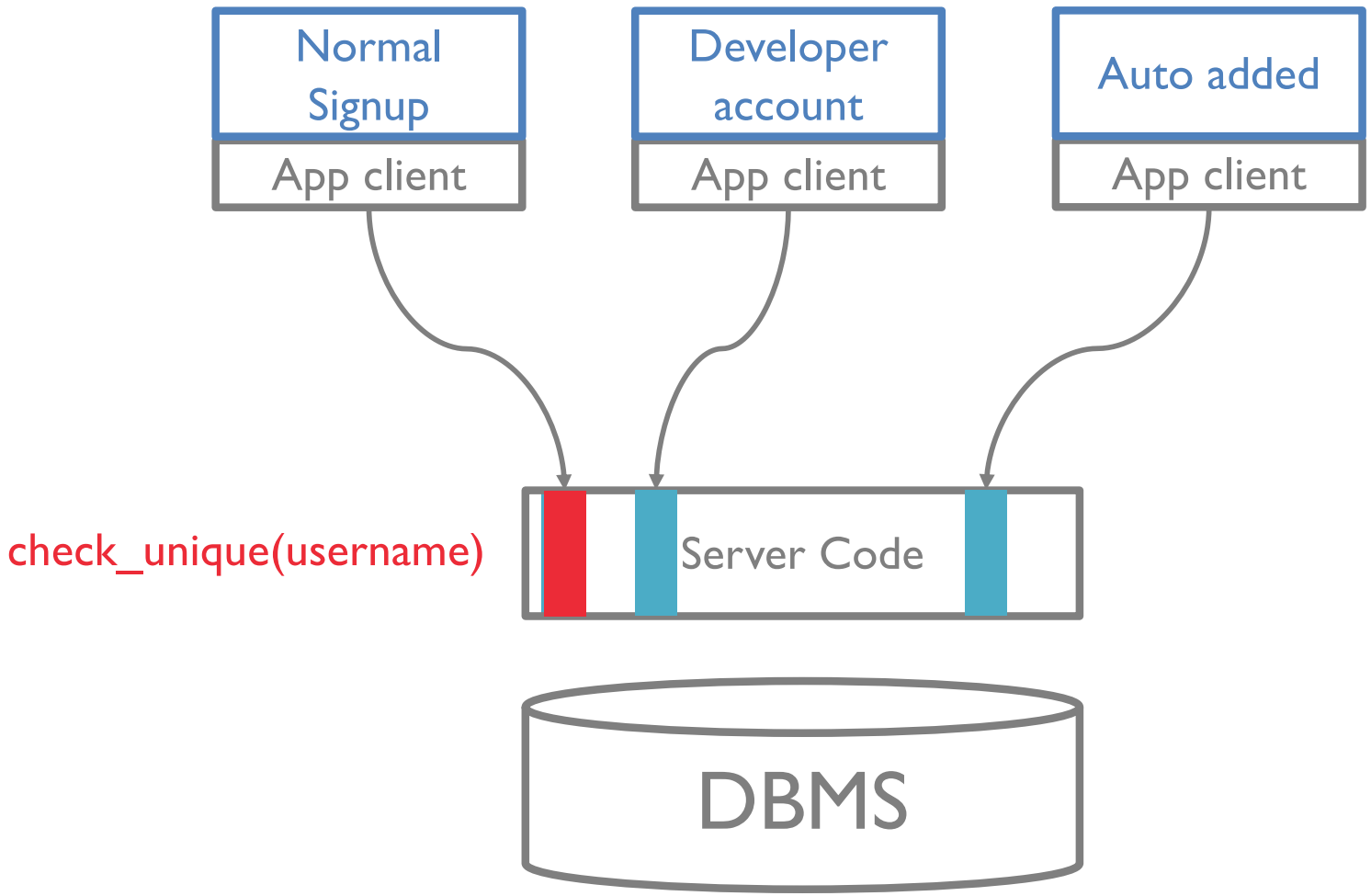
# It is Hard to Design Applications



# It is Hard to Design Applications



# It is Hard to Design Applications



Quick Links | Eugene Wu (Session will expire at 11:50 AM)

Support | My Courses | Viewable Courses | Logout

COURSEWORKS@COLUMBIA

COMSW4111_001_2015_3: INTRODUCTION TO DATABASES (Fall 2015)

View Site As ✓ – Select Role –
Student
Teaching Assistant

- Home
- Files & Resources
- Syllabus
- Mailtool
- Gradebook
- Site Settings
- Library Reserves
- Research Guides
- Roster
- Textbooks
- Piazza
- Help

## INTRODUCTION TO DATABASES

Edit | Permissions

**CourseNo:** COMSW4111_001_2015_3
**Meeting Time:** MW 02:40P-03:55P **Meeting Location:** SEELEY W. MU 833

**Instructor Information:**
Eugene Wu

COMSW4111_001_2015_3

# Entity-Relationship Modeling

Entities (objects) to store and their attributes

Relationships between entities and their attrs.

Integrity constraints & business rules

Visually modeled, easy to turn into DB schema

NEXT SEMESTER COURSES

Fall 2015 – Spring 2016 Courses

| Course Number | Course Title |
|---|---|
| COMSE6910_024_2015_3 | FIELDWORK |
| COMSW4111_001_2015_3 | INTRODUCTION TO DATABASES |

Reflects Registrar changes through Mar-06-2015 2:02:13AM

Courses

- Course Number
- Course Title
- Year
- Semester

**Eugene Wu** test test again just then Clear

Say something | Say it

Profile | Wall

**Basic Information**

Nickname

Birthday

Personal summary

B I U ABC $x_2$ $x^2$ HTML

Save changes | Cancel

**Contact Information**

Email: ew2493@columbia.edu

Home page

Work phone

Home phone

Mobile phone

Facsimile

Save changes | Cancel

# Users

- Nickname
- Name
- Birthday
- Summary
- Email
- …

# Basics: Entities

Entity e.g., intro to databases

- real-world object distinguishable from other objects
- described as set of attributes & the values
- (think one record)

Entity Set e.g., all courses

- collection of similar entities
- all entities have same attributes (unless Is-A)
- must have one or more keys
- attributes have domains
- ≈ table

# Example: Entity

Keys (cid, uid) are underlined

- Values must be unique
- (can use as hashtable key to lookup in table)

| Course |
| --- |
| <u>cid</u> |
| name |
| loc |
| schedule |

| Users |
| --- |
| <u>uid</u> |
| name |
| age |
| summary |

# Basics: Relationships

Relationship: association between 2 or more entities

e.g., alice is taking Introduction to DBs

Relationship Set: collection of similar relationships

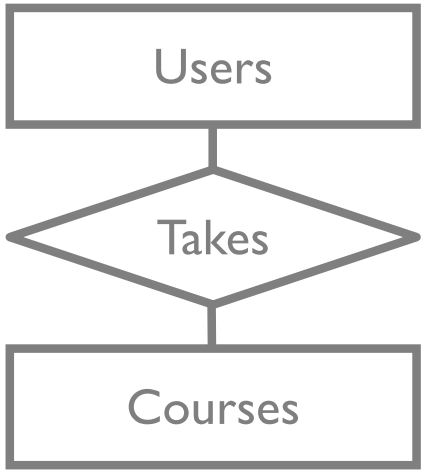
N-ary relationship set R relates N entity sets $E_1 \ldots E_n$

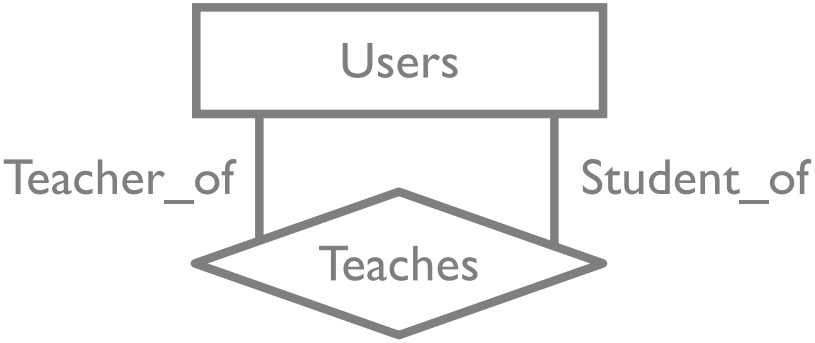
Each $r \in R$ involves entities $e_1 \ldots e_n$

An $E_i$ can be part of diff. relationship sets or diff. roles in same set

# Basics: Relationships

Users can have different roles in same relationship set



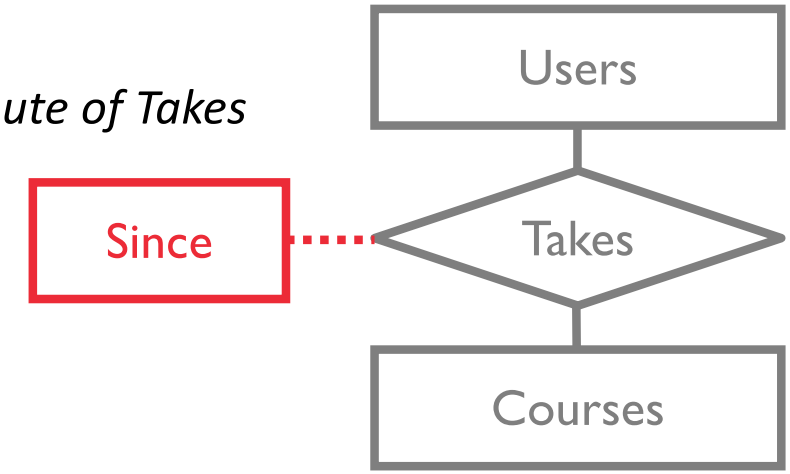



# Basics: Relationships

Relationships sets can have descriptive attributes

Denoted with dotted line from diamond to box

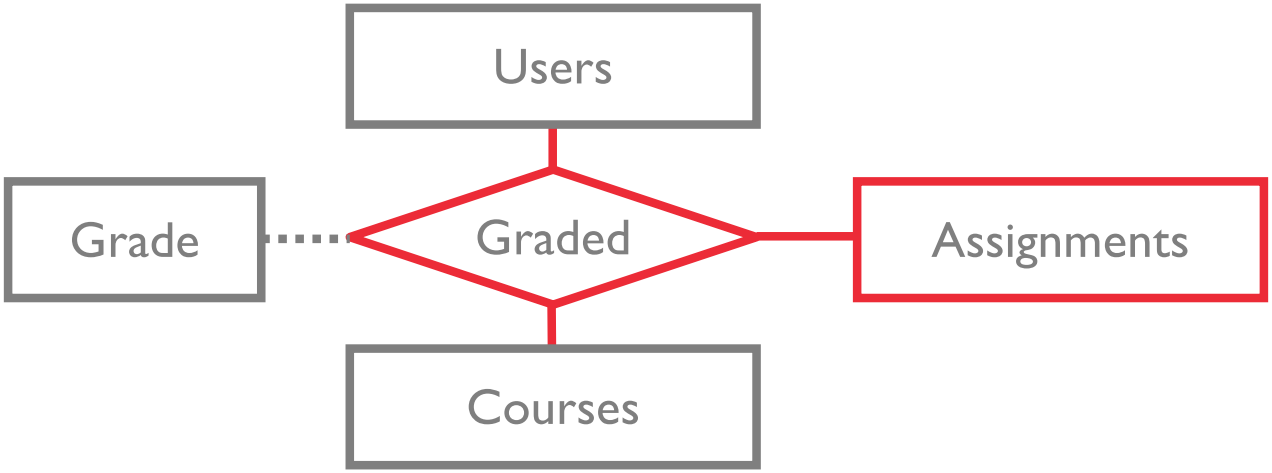
*since is attribute of Takes*



# Basics: Ternary Relationships

Connects three entities

N-ary relationships possible too.




*Assignments, Courses, and Users participate in the Graded relationship set*

# Constraints

Help avoid corruption, inconsistencies

- Key constraints
- Participation constraints
- Weak entities
- Overlap and covering constraints

# Key Constraints

Defines cardinality requirements on relationships

Many to many e.g., *Takes*

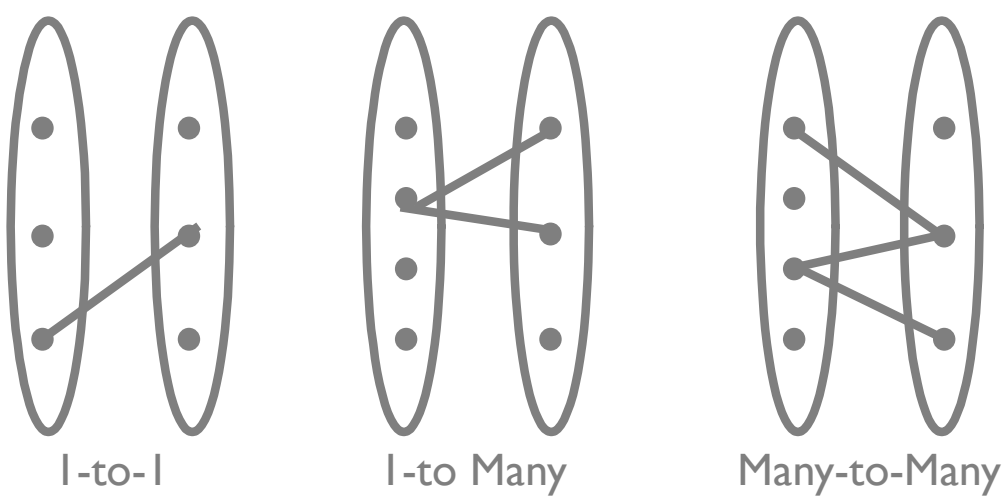
a user can take many courses

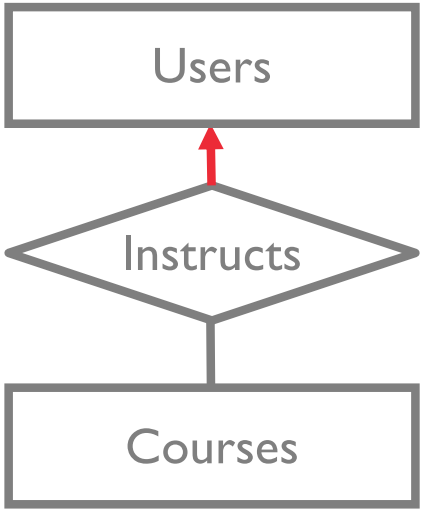
a course can have many users that take the course

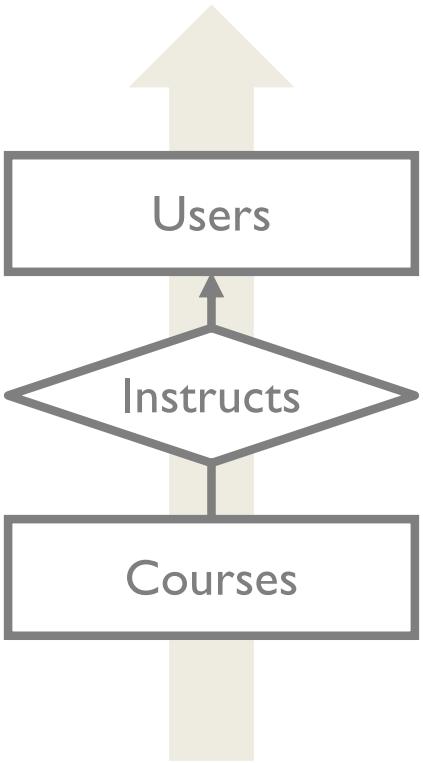
One to Many e.g., *Instructs*

a course has at most one instructor

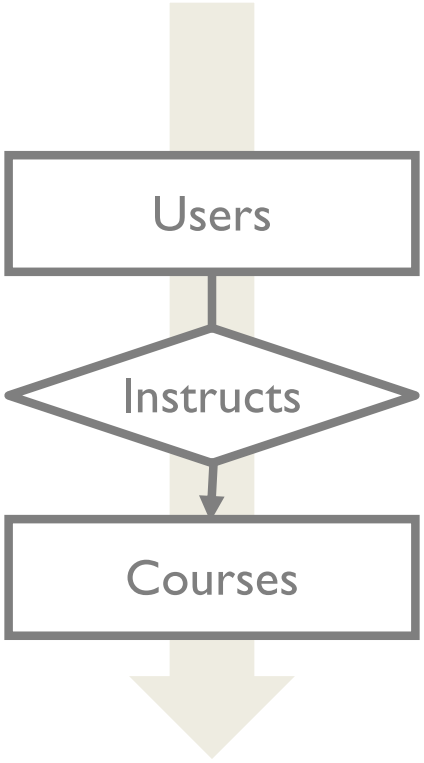
*Draw arrow from diamond to box*



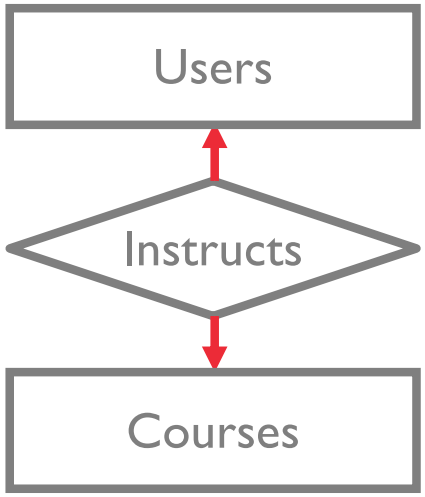






*A course is instructed by ≤1 user*
*(read along the beige arrow)*




*A user instructs ≤1 course*




*A course is instructed by ≤1 user*
*AND*
*A user instructs ≤1 course*

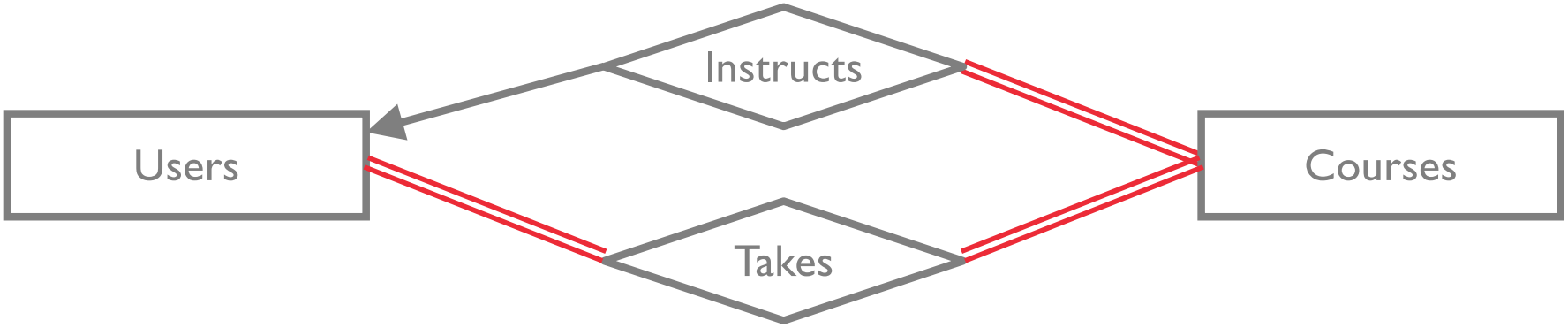
# Participation Constraints

Does every course need an instructor?

- If yes, it's a participation constraint
- Otherwise, *partial* participation constraint

Denoted by double line between entity set and relationship set

*Each course MUST have an instructor*
*(participation of Courses in instructs is Total)*



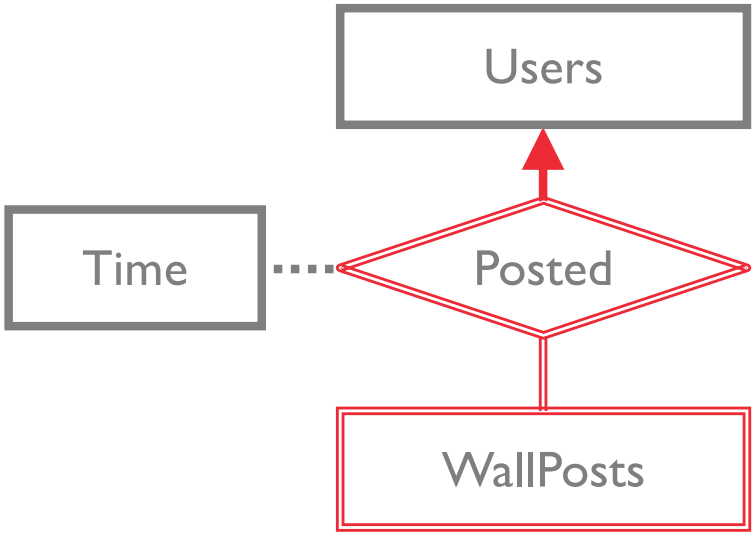

*Each user must take at least one course and*
*Each course must have at least one user (student)*

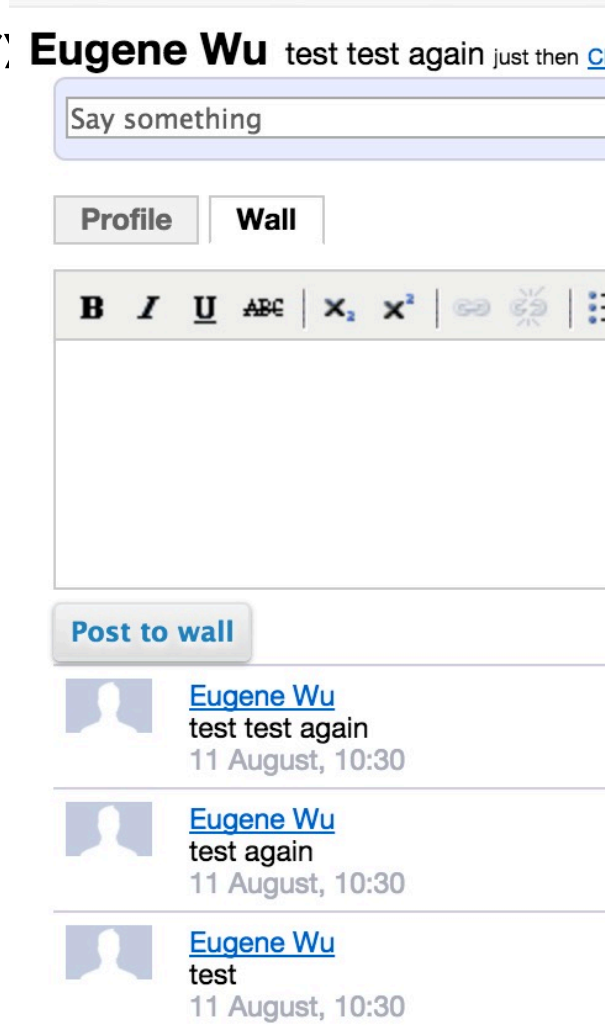
# Weak Entities

A *weak entity* can only be uniquely identified by using the primary key of its owner entity

- Owner and weak entity sets must have 1-to-N relationship
- Weak entity set must have total participation in this *identifying* relationships set

Denoted as double line around weak entity, set relationship set, and the edge between them; an arrow to owner entity



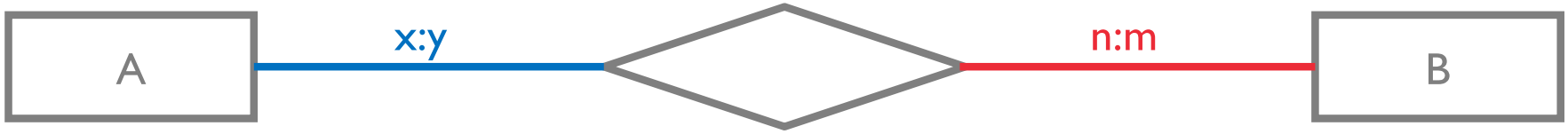



# General Cardinality Constraints


same as


A user instructs 0 to ∞ courses

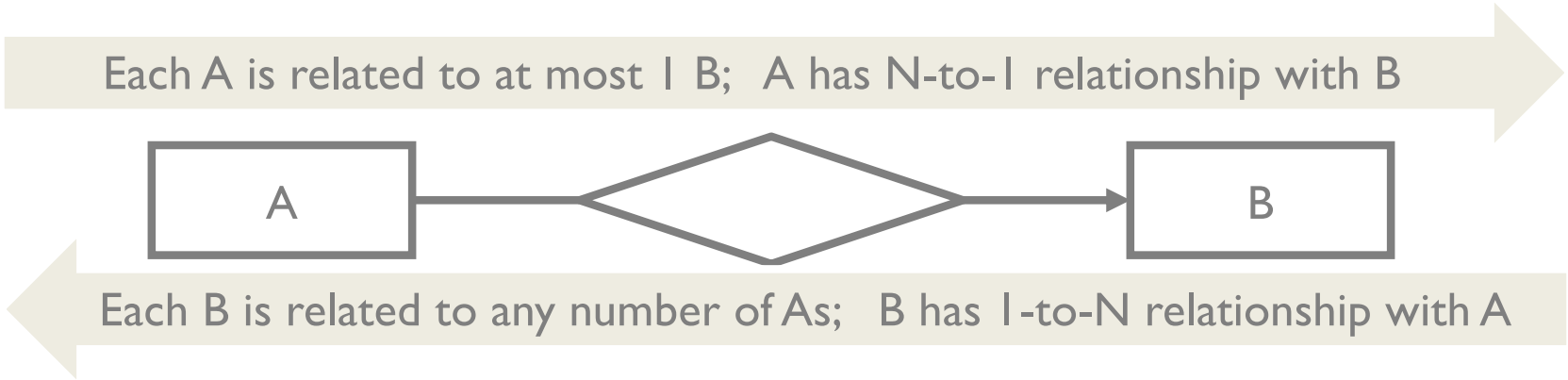
A course has 0 to 1 instructors

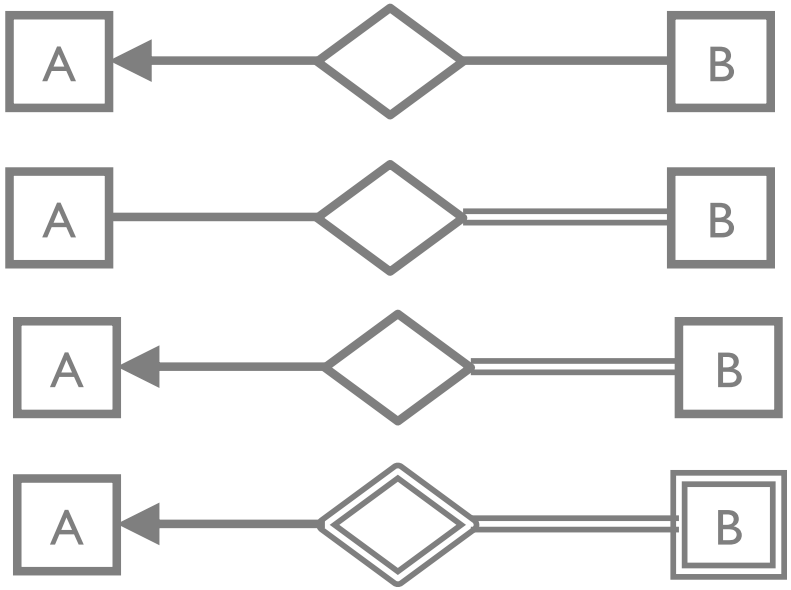



Each A entity has a relationship with between x to y different B entities

Each B entity has a relationship with between n to m different A entities

# Read arrows pointing in the direction from start to end




B has at most one A

B has at least one A

B has exactly one A

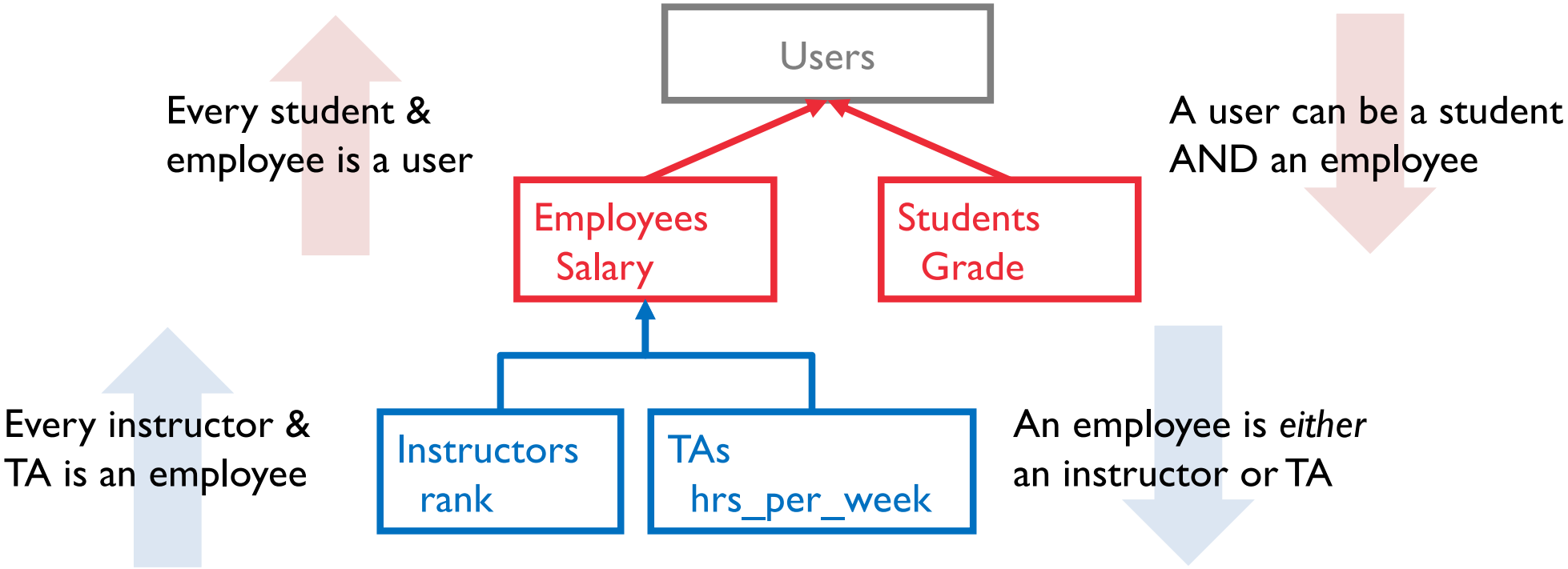
B is a weak entity

# Specialization Hierarchies

Inheritance rules similar to programming languages

- add descriptive attributes specific to a subclass e.g., grade
- identify entity set that participate in a relationship

Denoted with arrow from subclass to superclass without a diamond

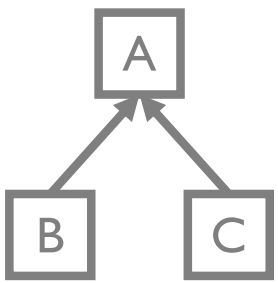


# Specialization Hierarchies

## Overlap Constraint

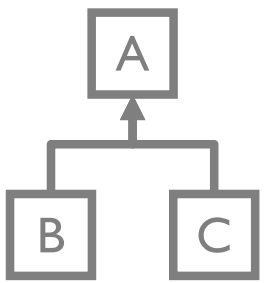
can A be a B *and* a C?

YES




*separate arrows*

NO



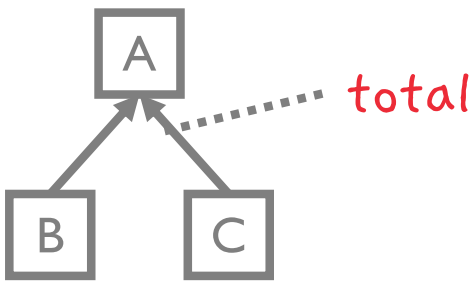

*merged into 1 arrow*

## Total Specialization Constraint

*must* A be a B or C?

specify as the comment "total"

with dashed link to arrows

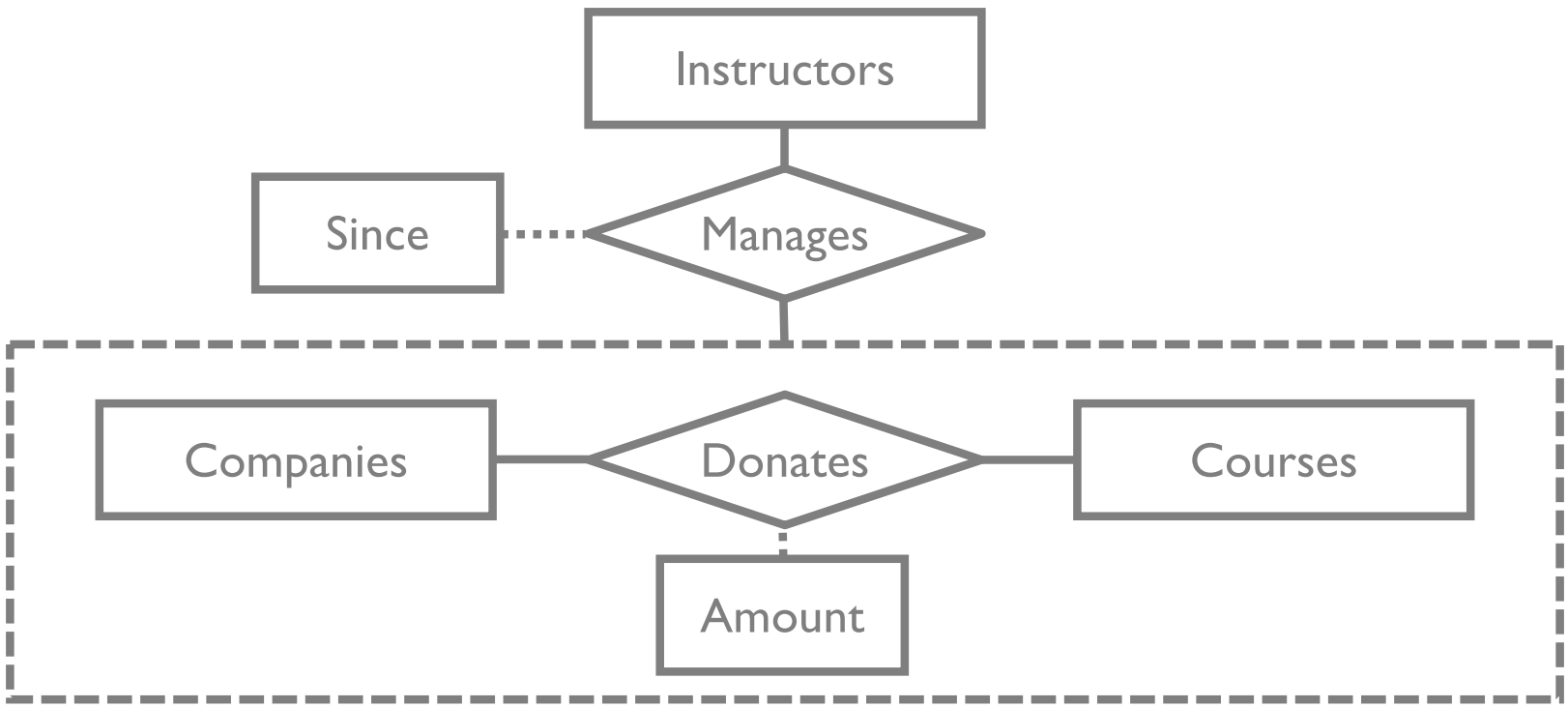


# Aggregation

Relationships between (entities – relationships)

Treat Relationship Set like an Entity Set to participate in other relationships

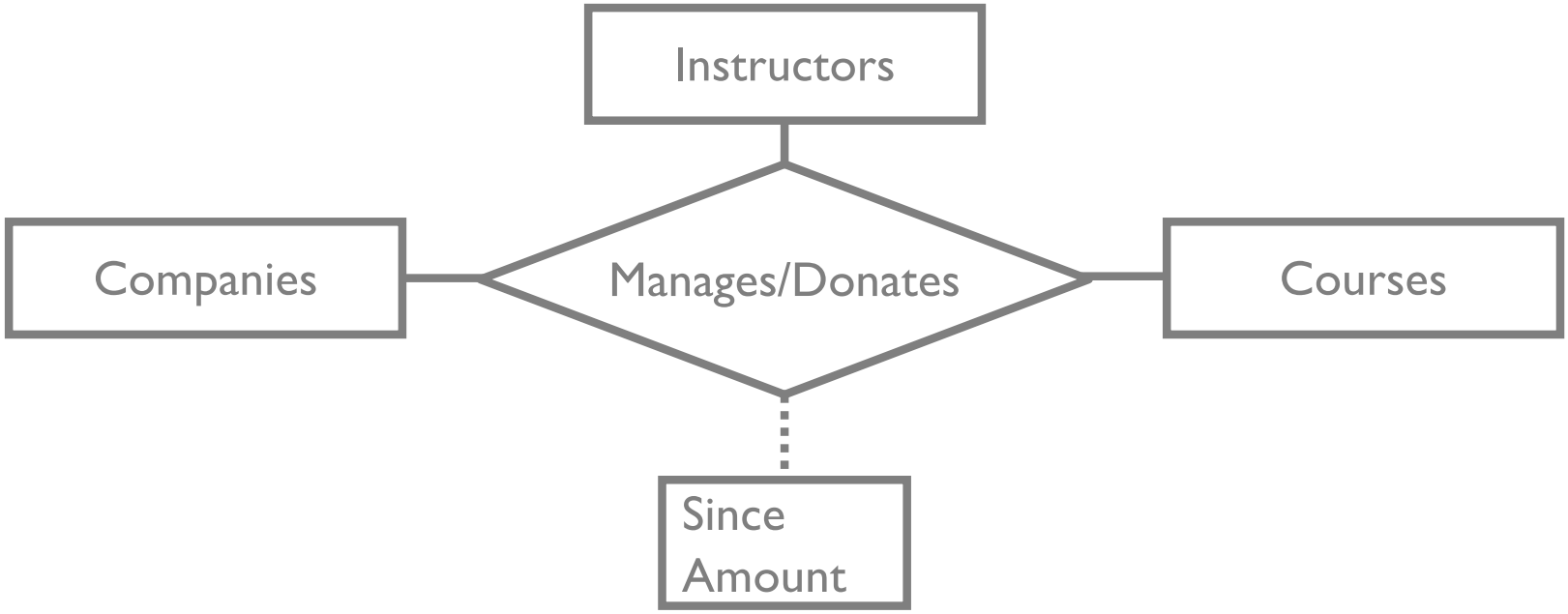
Denoted as dashed line around the relationship set



# Aggregation vs Ternary Relationships

Why use aggregation?

- Manages and Donates are distinct relationships with own attrs
- Can define constraints on relationship sets

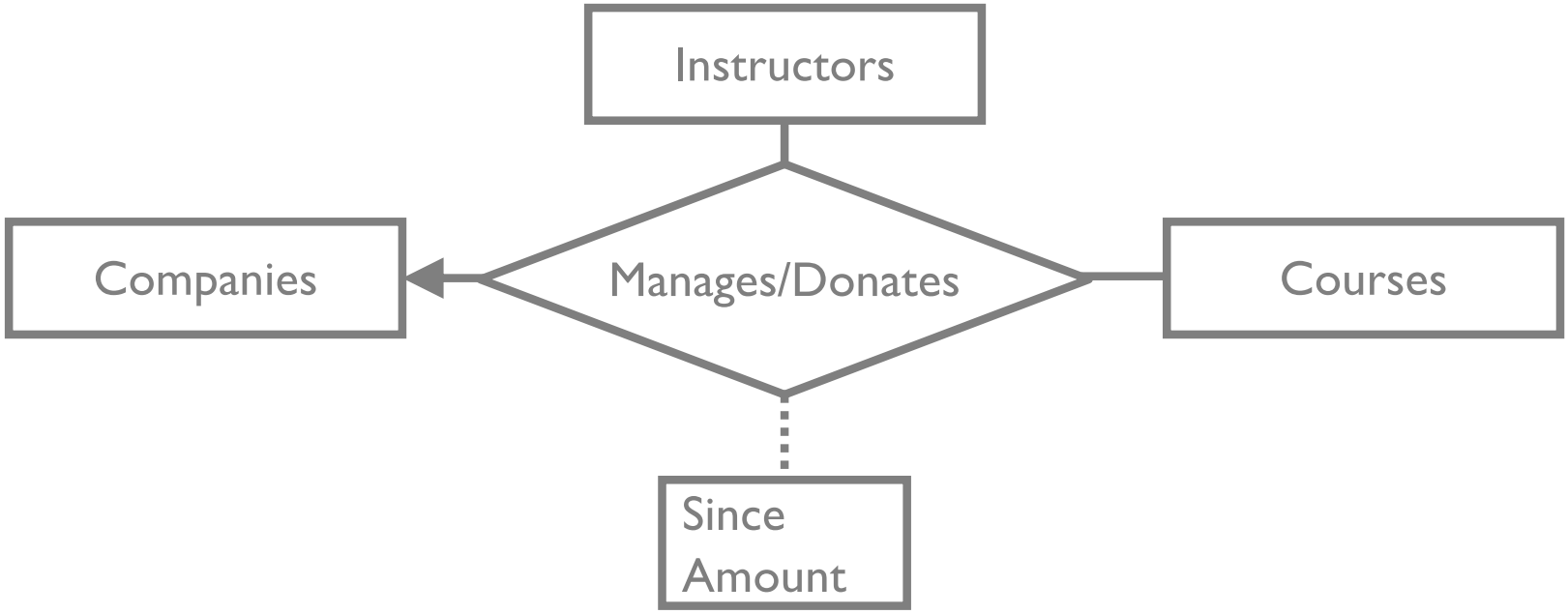


# Aggregation vs Ternary Relationships

Constraints apply to all connected entity sets

*A donation can be managed by at most one instructor*

But also enforces: *A course can have at most one donation*



# Using the ER Model

OK, we've seen the *syntax*.

How to use it involves design choices

Design Choices for a concept

- Entity or Attribute?
- Entity or Relationship?
- Binary or Ternary relationship?
- Aggregation or Ternary relationship?

# Entity or Attribute?

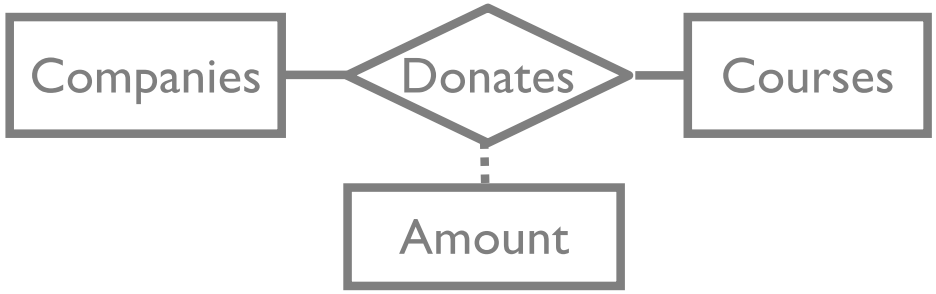
Is users.address an attribute of Users or an entity connected to Users by a relationship?

Depends (and may change over time!)

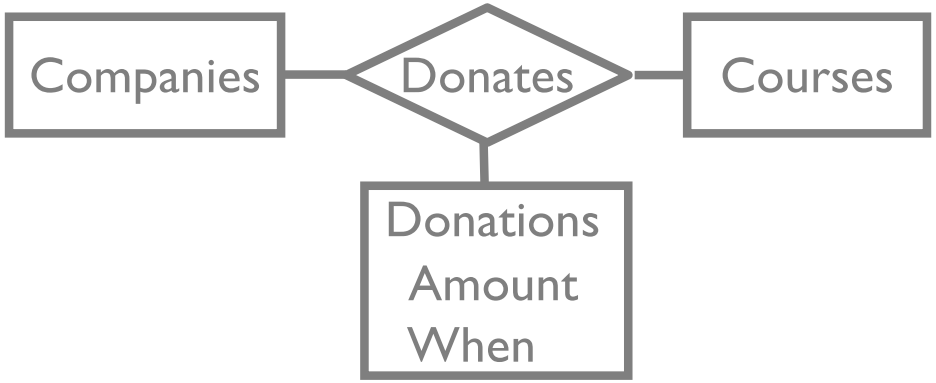
- If a user has >1 addresses, must be an entity
- If an address has attrs (structure), must be entity
  - e.g., want to search for users by city, state, or zip

# Entity or Attribute?

A company can't donate multiple amounts



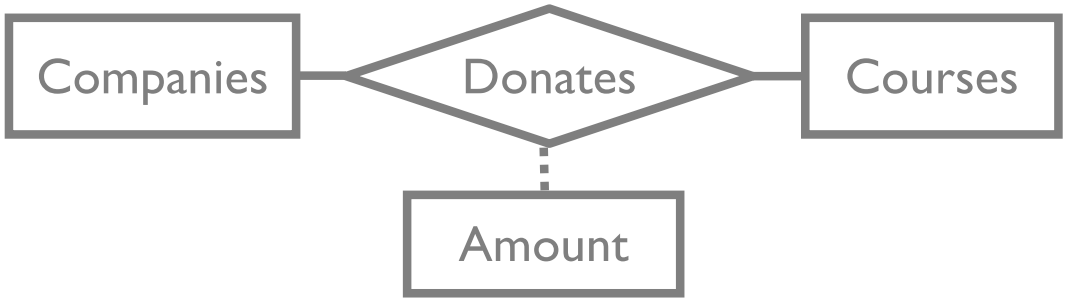

Company can make multiple donations



# Entity or Relationship?

But what if company donates to school for all data-related courses?

- Redundancy of *amount*, need to remember to update every one
- Misleading implies *amount* tied to *each* donation individually



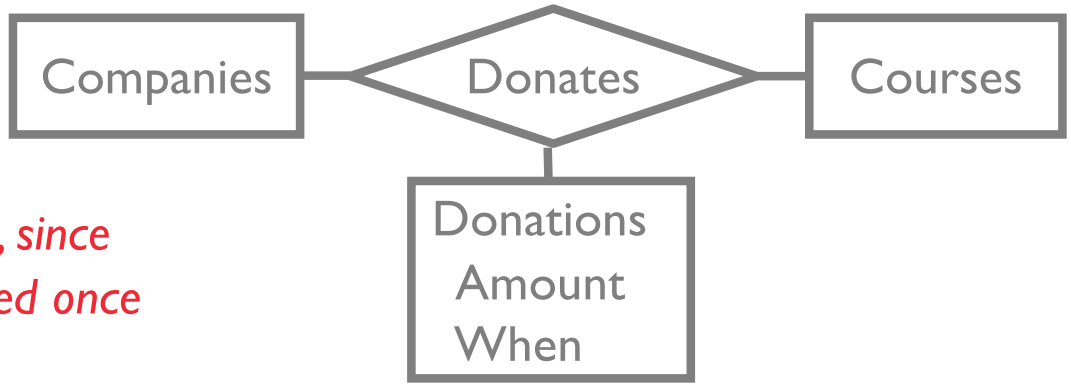

| Company | Course | Amount |
|---|---|---|
| Amazon | 4111 | 2000 |
| Amazon | 4112 | 2000 |
| Amazon | 5111 | 2000 |

*These amounts are logically the same (redundant)!*

# Entity or Relationship?

If company donates once to school for data related courses.

Refactor amount into an entity




*Company redundant, since company only donated once*

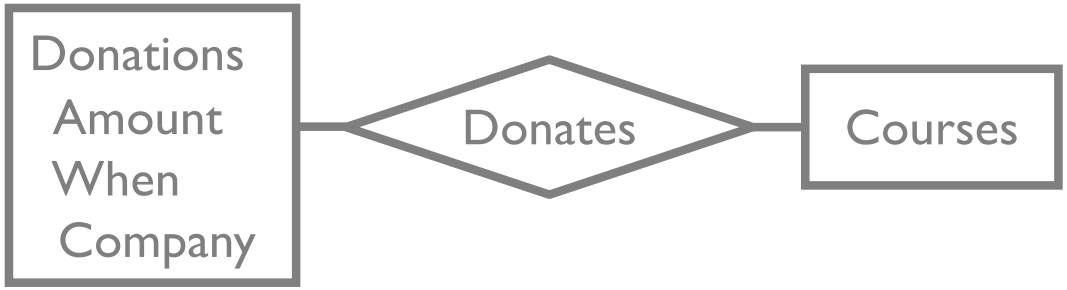
| Company | Course | Donation |
|---|---|---|
| Amazon | 4111 | 1 |
| Amazon | 4112 | 1 |
| Amazon | 5111 | 1 |

| Donation | When | Amount |
|---|---|---|
| 1 | Today | 2000 |

# Entity or Relationship?

If company donates once to school for data related courses.

Refactor amount into an entity

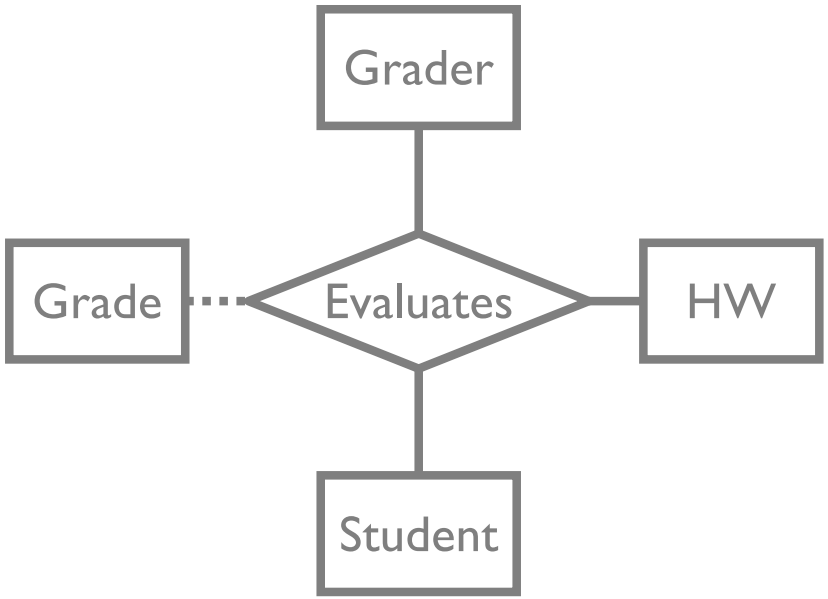



| Course | Donation |
|---|---|
| 4111 | 1 |
| 4112 | 1 |
| 5111 | 1 |

| Donation | When | Amount | Company |
|---|---|---|---|
| 1 | Today | 2000 | Amazon |

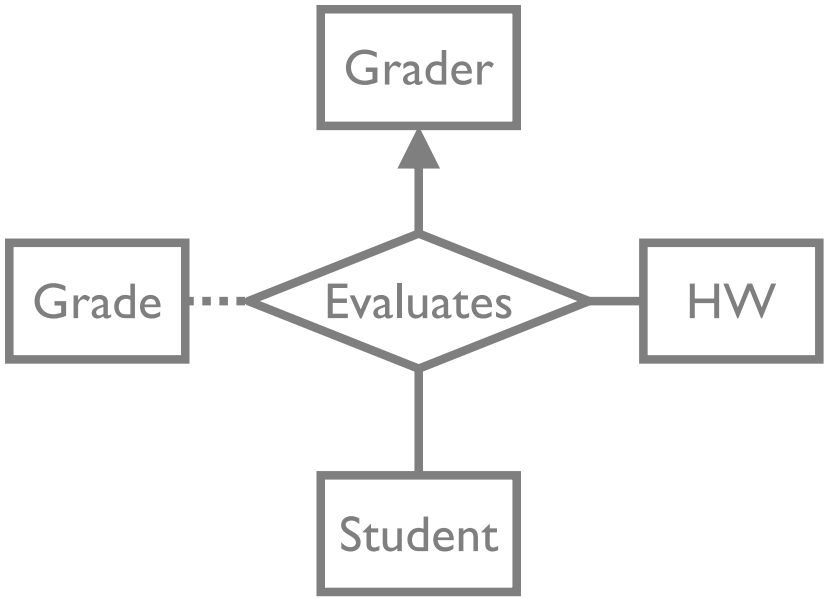
# Binary or Ternary Relationship?

What if each HW has at most one grader?

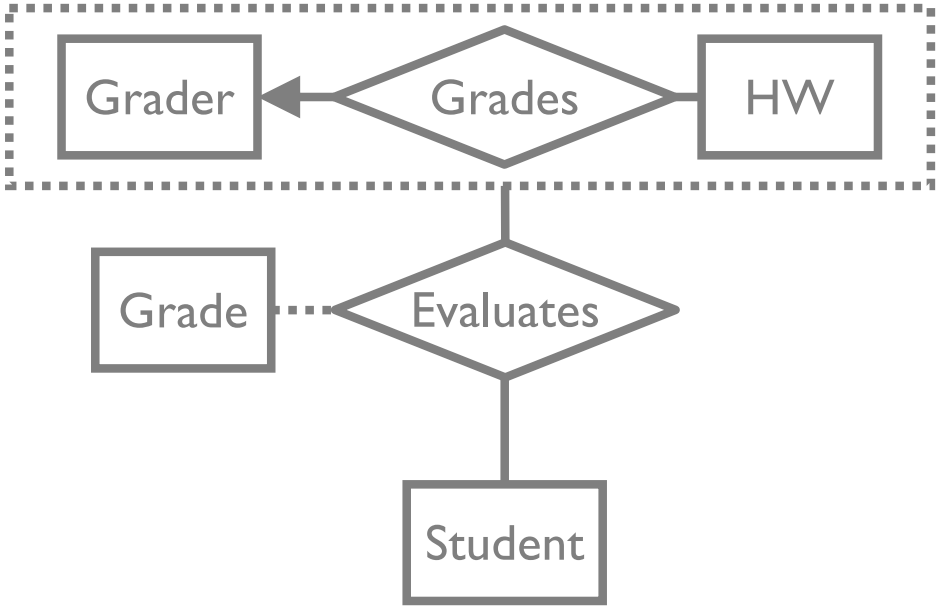


# Binary or Ternary Relationship?

What if each HW has at most one grader?




*Actually says that each student's HW submission has at most one grader*

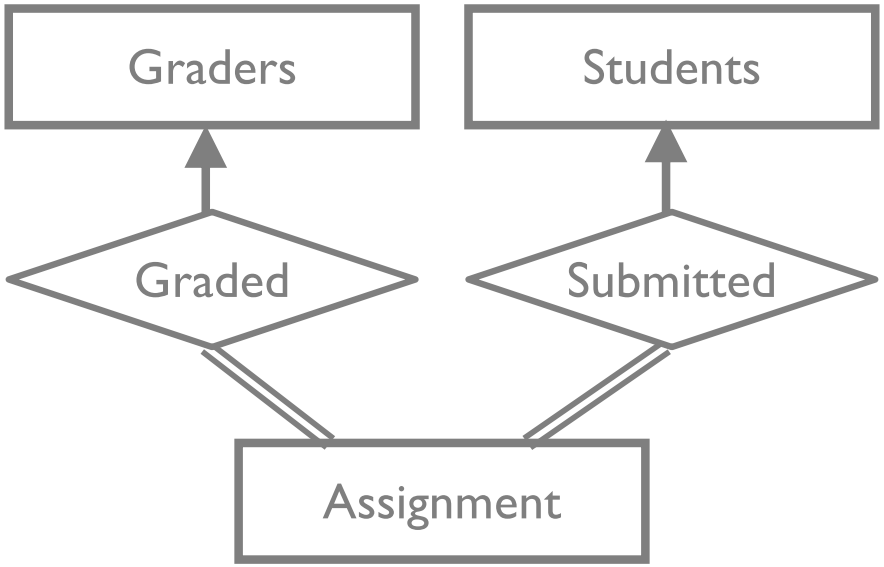



*Each HW has at most 1 grader and the grader evaluates student submissions*

# Binary or Ternary Relationship?

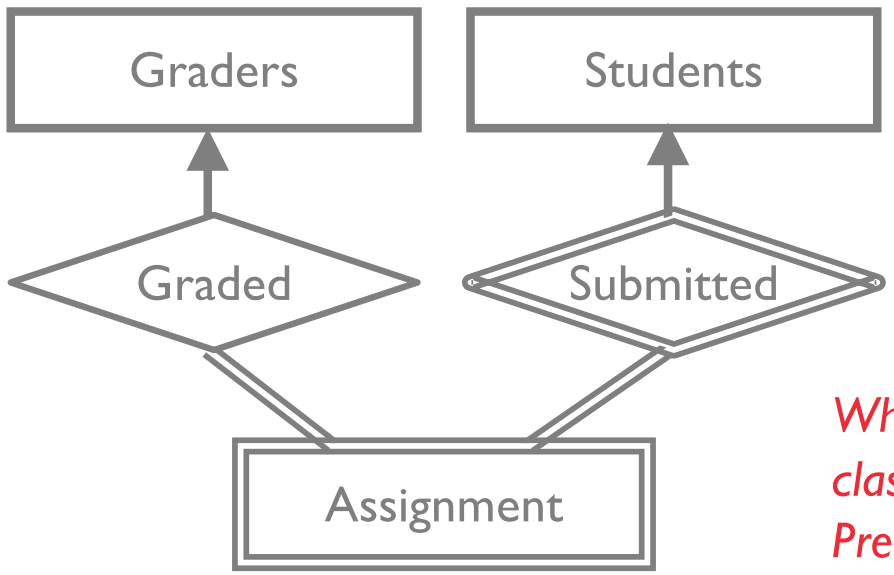
## Binary relationships allows additional constraints

What should happen if a student drops the class? (see next slide)



# Binary or Ternary Relationship?

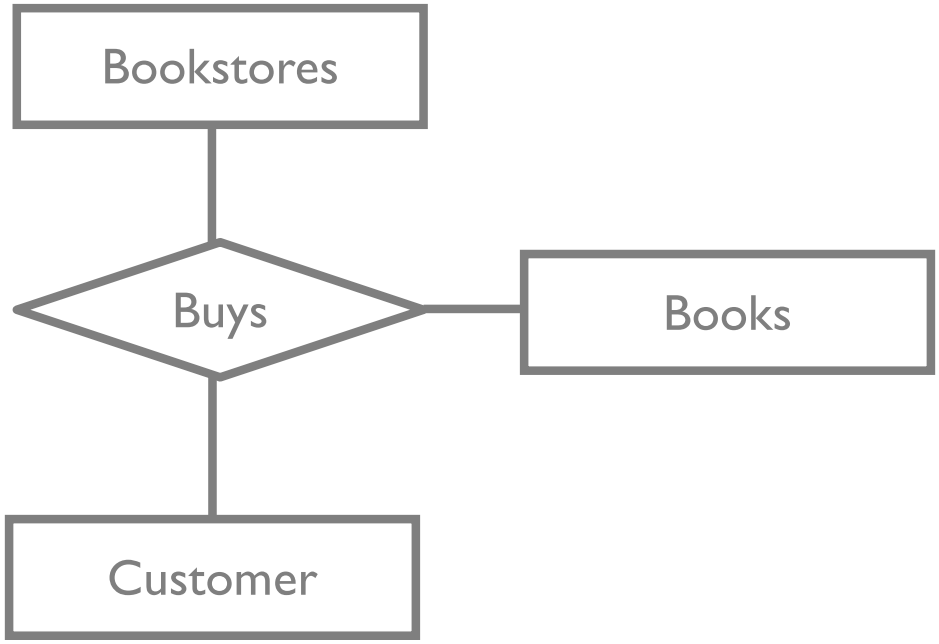
Binary relationships allows additional constraints




*When student drops the class, HW0 also disappears! Previous slide was correct*

# Binary or Ternary Relationship?

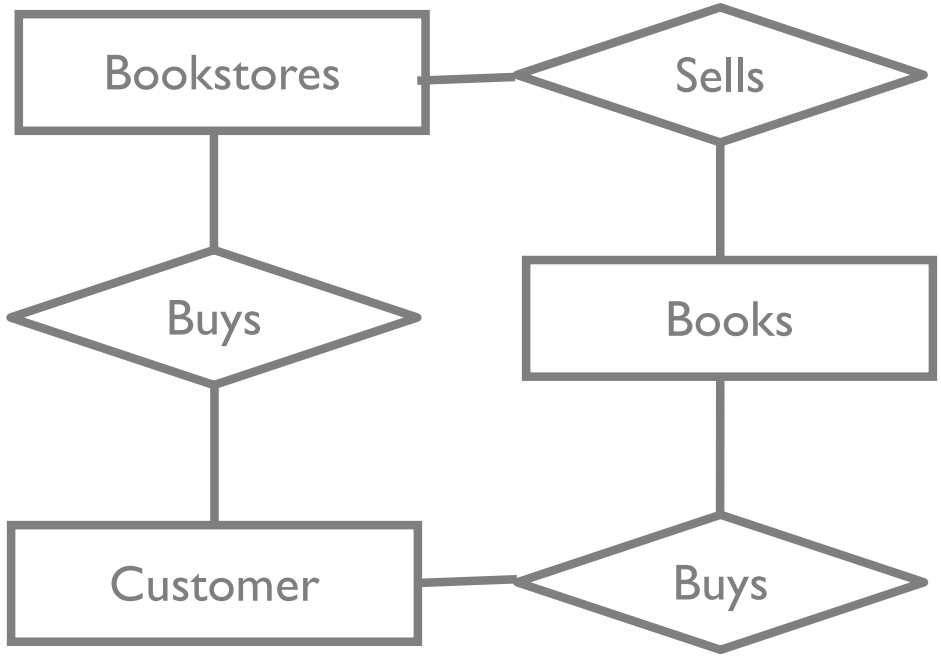
Sometimes have true ternary relationship that is defined by all three entities.



# Binary or Ternary Relationship?

Sometimes have true ternary relationship that is defined by all three entities.

*Doesn't Really Work*



# Using ER Modeling

## Constraints in ER Modeling

- Many types of data semantics can be captured using ER
- Some constraints not captured (discuss limitations later)

## Need further schema refinement

- ER Model is still subjective, need further refinement after translated into relational schema

# Summary

Requirements

what are you going to build?

Conceptual Database Design

pen-and-pencil description

(Today) ER Modeling

Logical Design

formal database schema

Schema Refinement:

fix potential problems, normalization

Physical Database Design

use sample of queries to optimize for speed/storage

App/Security Design

prevent security problems

# Summary

Conceptual design follows *requirements analysis*

ER model helpful for conceptual design

- constraints are expressive
- matches how we often think about applications

Core constructs

- entity, relationship, attribute
- weak entities, ISA, aggregation

Many variations beyond today's discussion

# Summary

ER design is subjective based on usage+needs

- Today we saw multiple ways to model same idea

ER design is not complete/perfect

- Developed in an enterprise-oriented world (ER First)
- Doesn't capture semantics (what does "instructor" *mean?)*
- Doesn't capture e.g., processes/state machines
- How to combine multiple ER models automatically?
- Limitation of imagination when designing application
- Open problems!

ER design is a useful way to think

# Next Time

Relational Model: de-facto DBMS standard

Set up for ER diagrams → Relational models