SFU

Database Systems I

CMPT 354 Summer 2024

Zhengjie Miao

22 May 2024

Assignment 1 Setup

- Due ~~May 26~~ May 29
  - Please start now, if not already
- Problem 1: either local radb or RATest
  - RATest is ready (with occasionally server deployment issues)
  - Post any issues you encounter on Piazza
  - But you are still able to work on this problem without the tools
  - Will have a more detailed document tonight or tomorrow

Outline

- Relational Model continued
- E/R Basics: Entities & Relationships
- Advanced E/R Concepts

Recap: Relational Model

- A database is a collection of relations (or tables)
- Each relation has a set of attributes (or columns)
- Each attribute has a name and a domain (or type)
- Each relation contains a set of tuples (or rows)

Keys

- A set of attributes K is a key for a relation R if
  - In no instance of R will two different tuples agree on all attributes of K
    - That is, K can serve as a "tuple identifier"
  - No proper subset of K satisfies the above condition
    - That is, K is minimal
- Example: *User* (*uid*, *name*, *age*, *pop*)
  - *uid* is a key of *User*
  - *age* is not a key (not an identifier)
  - {*uid*, *name*} is not a key (not minimal)

Since uid can serve as a unique identifier, adding name does not make the key minimal.

Schema vs. Instance

uid	*name*	*age*	*pop*
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3

- Is *name* a key of *User*?
  - Yes? Seems reasonable for this instance
  - No! User names are not unique in general
- Key declarations are part of the schema

Has to be true for all instances of this relation to be a key

More examples of keys

- *Member* (*uid*, *gid*)
  - {*uid*, *gid*}
  - A key can contain multiple attributes

More examples of keys

- *Member* (*uid*, *gid*)
  - {*uid*, *gid*}
  - A key can contain multiple attributes
- *Address* (*street_address*, *city*, *state*, *zip*)
  - {*street_address*, *city*, *state*}
  - {*street_address*, *zip*}
  - A relation can have multiple keys!
    - We typically pick one as the "primary" key, and underline all its attributes, e.g., *Address* (*street_address*, *city*, *state*, *zip*)

Use of keys

- More constraints on data, fewer mistakes
- Look up a row by its key value
  - Many selection conditions are “key = value”
- “Pointers” to other rows (often across tables)
  - Example: *Member* (*uid*, *gid*)
    - *uid* is a key of *User*
    - *gid* is a key of *Group*
    - A *Member* row “links” a *User* row with a *Group* row
  - Many join conditions are “key = key value stored in another table”

Outline

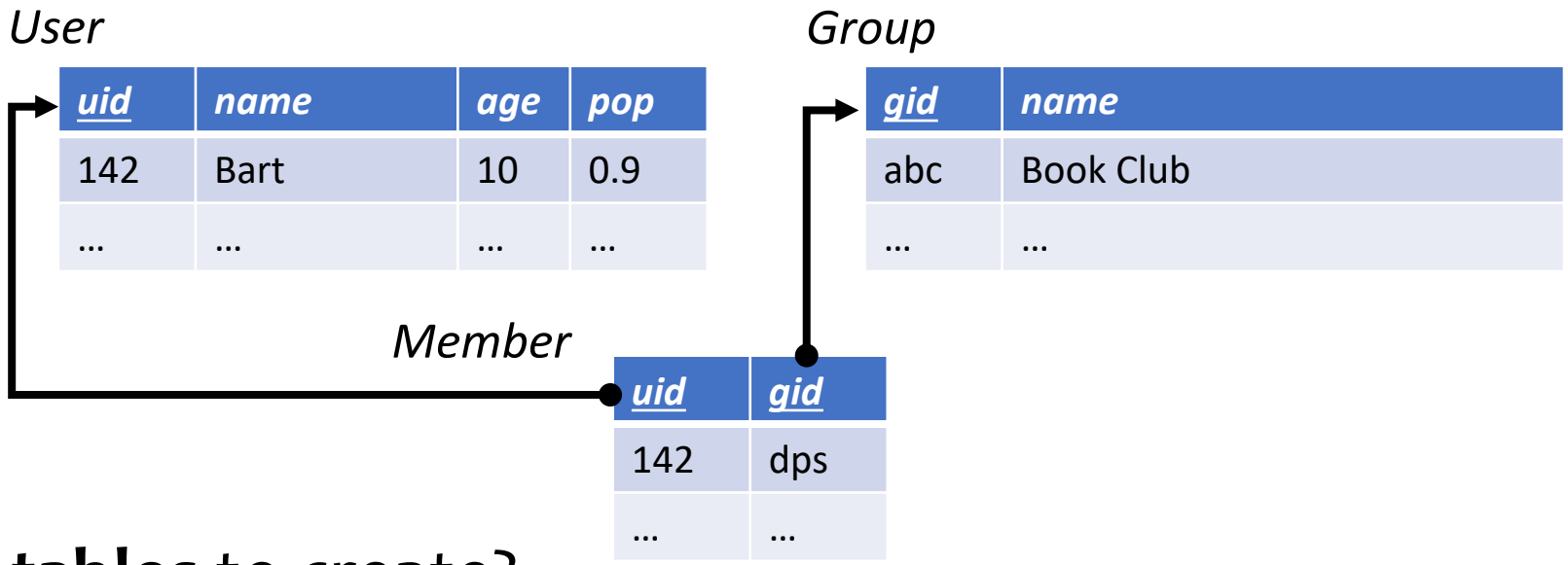
- Relational Model continued
- **E/R Basics: Entities & Relationships**
- Advanced E/R Concepts

Database design

- Understand the real-world domain being modeled
- Specify it using a database design model
  - More intuitive and convenient for schema design
  - But not necessarily implemented by DBMS
  - A few popular ones:
    - Entity/Relationship (E/R) model
    - Object Definition Language (ODL)
    - UML (Unified Modeling Language)
- Translate specification to the data model of DBMS
  - Relational, XML, object-oriented, etc.
- Create DBMS schema

Motivation

- How to figure out this **database design**?



- What **tables** to create?
- Which **attributes** should be added to each table?
- What are the **relationships** between the tables?

But what about ORM?

- Automatic object-relational mappers are made popular by rapid Web development frameworks
  - For example, with Python SQLAlchemy:
    - You declare Python classes and their relationships
    - It automatically converts them into database tables
    - If you want, you can just work with Python objects, and never need to be aware of the database schema or write SQL
- But you still need designer discretion in all but simple cases

Ask professor about this point

- Each language/library has its own syntax for creating schema and for querying/modifying data
  - Quirks and limitations cause portability problems
  - They are not necessarily easier to learn than SQL

History of E/R Model

- E/R Model (Entity-Relationship Modeling)
  - Codd wrote a long letter criticizing paper
  - Many people suggested him to give up this idea

Combine navigational model and relational model

The entity-relationship model—toward a unified view of data

PPS Chen - ACM Transactions on Database Systems (TODS), 1976 - dl.acm.org

A data model, called the entity-relationship model, is proposed. This model incorporates some of the important semantic information about the real world. A special diagrammatic technique is introduced as a tool for database design. An example of database design and description using the model and the diagrammatic technique is given. Some implications for data integrity, information retrieval, and data manipulation are discussed. The entity-relationship model can be used as a basis for unification of different views of data: the …

☆ 99 Cited by 11297 Related articles All 79 versions

Dr. Peter Chen

- Why not build RDBMS based on E/R Model?
  - No query language proposed
  - Relational DBMS in the 1970's

People find the E/R model is useful for designing the schema.

Entity-relationship (E/R) model

- Historically and still very popular
- Concepts applicable to other design models as well
- Can think of as a "watered-down" object-oriented design model
- Primarily a design model — not directly implemented by DBMS
- Designs represented by E/R diagrams
  - We use the style of E/R diagram covered by the GMUW book; there are other styles/extensions
  - Very similar to UML diagrams

Database Design Process

1. Requirements Analysis → 2. Conceptual Design → 3. Logical, Physical, Security, etc.

1. **Requirements analysis**
   - What data is going to be stored?
   - What are we going to do with the data?
   - Who should access the data?

What constraints are being added to the domain

Technical and non-technical people are involved

Database Design Process

1. Requirements Analysis → **2. Conceptual Design** → 3. Logical, Physical, Security, etc.

2. Conceptual Design

- A high-level description of the database
- Sufficiently precise that technical people can understand it
- But, not so precise that non-technical people can't participate

This is where E/R fits in

Database Design Process

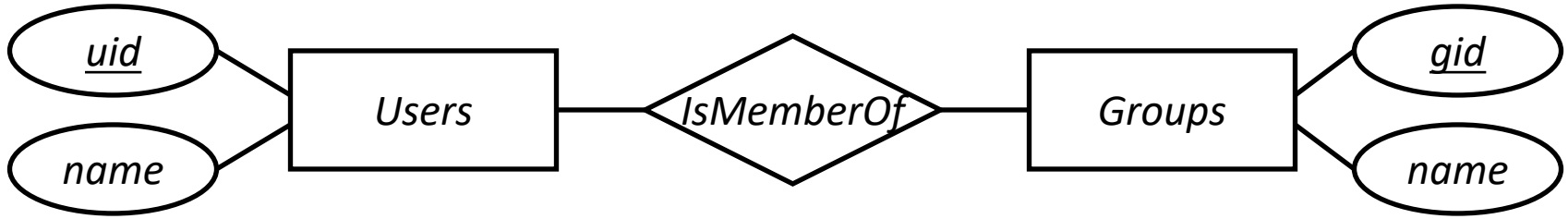
3. More:

- Logical Database Design
- Physical Database Design
- Security Design

What indexes you are using

Database Design Process

E/R Model & Diagrams used



E/R is a visual syntax for DB design, which is **precise enough** for technical points but **abstracted enough**

Entities and Entity Sets

- Entity: a "thing," like an object
- Entity set: a collection of things of the same type, like a relation of tuples or a class of objects
  - *These are what is shown in E/R diagrams - as rectangles*
  - Eg: A specific person or product

How do you display a sole entity?

Attributes

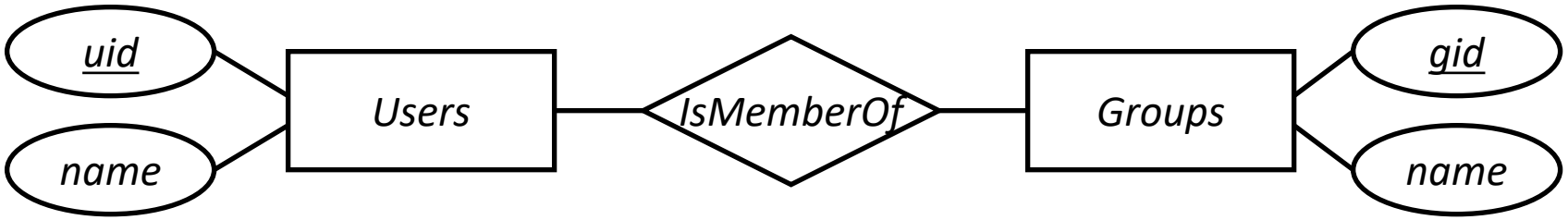
- Attributes: properties of entities or relationships, like attributes of tuples or objects
  - Represented by ovals attached to an entity set

Keys

- A key of an entity set is represented by underlining all attributes in the key
  - A key is a set of attributes whose values can belong to at most one entity in an entity set—like a key of a relation
  - Every entity set must have a key
  - Denote by underlining.

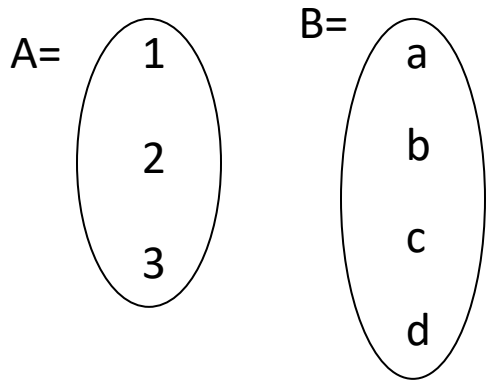
Relationships

- Relationship: an association among entities
- Relationship set: a set of relationships of the same type (among same entity sets)
  - Represented as a diamond



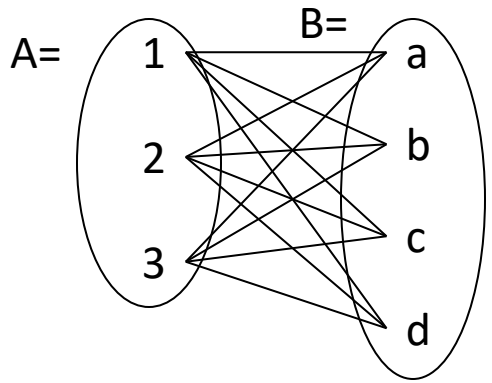
What is a Relationship exactly?

- Relationship: an association among entities
- A mathematical definition:
  - Let A, B be sets
    - *A={1,2,3}, B={a,b,c,d}*



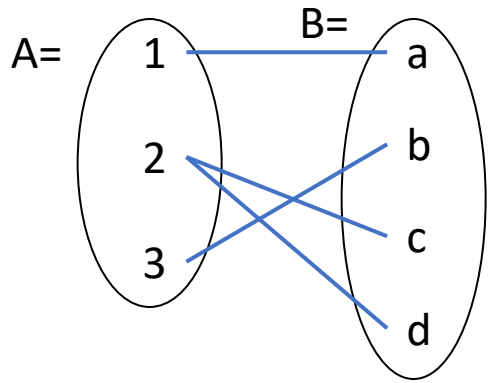
What is a Relationship exactly?

- Relationship: an association among entities
- A mathematical definition:
  - Let A, B be sets
    - *A={1,2,3}, B={a,b,c,d}*
  - A x B (the **cross-product**) is the set of all pairs (a,b)
    - *A × B = {(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)}*



What is a Relationship exactly?

- Relationship: an association among entities
- A mathematical definition:
  - Let A, B be sets
    - $A=\{1,2,3\}$, $B=\{a,b,c,d\}$
  - A x B (the **cross-product**) is the set of all pairs (a,b)
    - $A \times B = \{(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)\}$
  - We define a relationship to be a subset of A x B
    - $R = \{(1,a), (2,c), (2,d), (3,b)\}$



What is a Relationship exactly?

User

uid	name	age	pop
123	Milhouse	10	0.2
857	Lisa	8	0.7
…	…	…	…

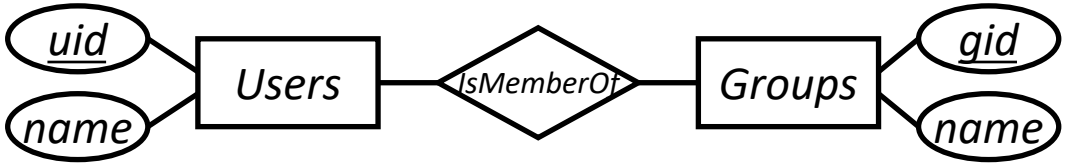
×

Group

gid	name
abc	Book Club
gov	Student Government
dps	Dead Putting Society
…	…

User × Group

uid	u.name	age	pop	gid	g.name
123	Milhouse	10	0.2	abc	Book Club
123	Milhouse	10	0.2	gov	Student Government
123	Milhouse	10	0.2	dps	Dead Putting Society
857	Lisa	8	0.7	abc	Book Club
857	Lisa	8	0.7	gov	Student Government
857	Lisa	8	0.7	dps	Dead Putting Society
…	…	…	…	…	…



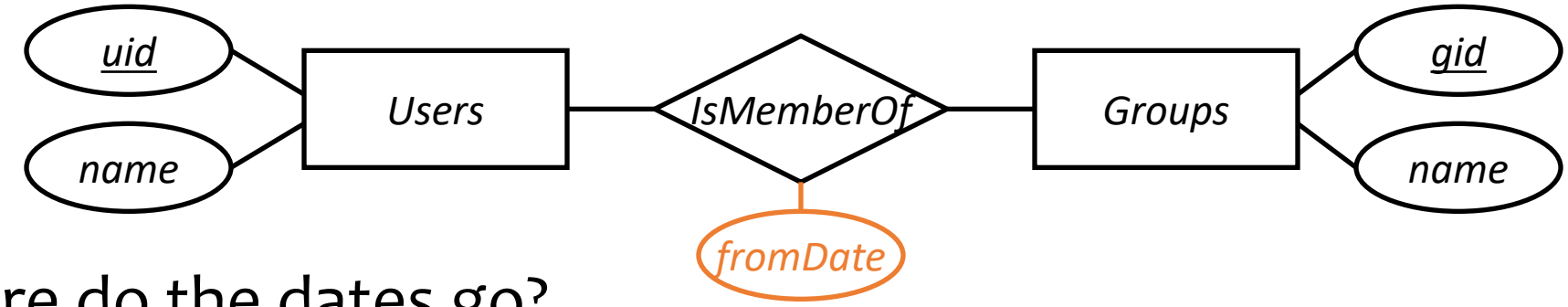
Member

uid	gid
123	gov
857	abc
857	gov
…	…

A relationship between entity sets User and Group
- A **subset of all possible combinations**
- with tuples uniquely identified by **keys**

Attributes of relationships

- Example: a user belongs to a group since a particular date



- Where do the dates go?
  - With *Users*?
    - But a user can join multiple groups on different dates
  - With *Groups*?
    - But different users can join the same group on different dates
  - With *IsMemberOf*!

Outline

- Relational Model continued
- E/R Basics: Entities & Relationships
- **Advanced E/R Concepts**

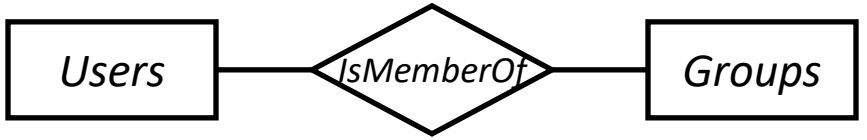
More on relationships

- There could be multiple relationship sets between the same entity sets
  - Example: *Users IsMemberOf Groups*; *Users Likes Groups*
- In a relationship set, each relationship is uniquely identified by the entities it connects
  - Example: Between Bart and "Dead Putting Society", there can be at most one *IsMemberOf* relationship and at most one *Likes* relationship
  - What if Bart joins DPS, leaves, and rejoins? How can we modify the design to capture historical membership information?
    - Make an entity set of *MembershipRecords*

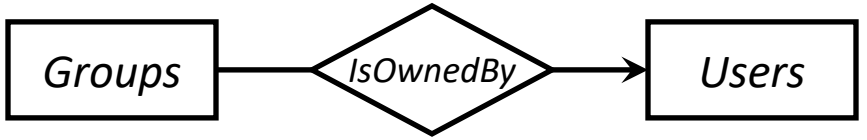
If a relationship could not be uniquely identified, does this mean that for any two entities, there could be various relationships and we are not sure to which the connection pertains?

Multiplicity of relationships

- E and F: entity sets
- Many-many: Each entity in E is related to 0 or more entities in F and vice versa
  - Example:



- Many-one: Each entity in E is related to 0 or 1 entity in F, but each entity in F is related to 0 or more in E
  - Example:

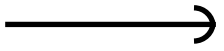


- One-one: Each entity in E is related to 0 or 1 entity in F and vice versa
  - Example:

Would we not have that the arrow from TwitterUsers to Users is a rounded arrow?

- "One" (0 or 1) is represented by an arrow

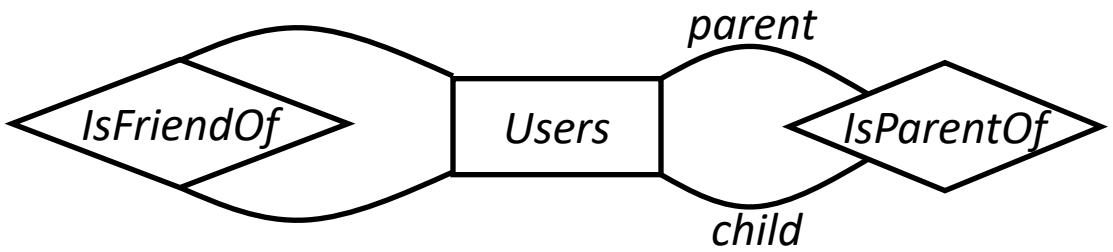
- "Exactly one" is represented by a rounded arrow



Roles in relationships

- An entity set may participate more than once in a relationship set
- May need to label edges to distinguish roles
- Examples
  - Users may be parents of others; label needed
  - Users may be friends of each other; label not needed

For the friendship relationship, there is only one role, friend.
Therefore, we do not need the labels

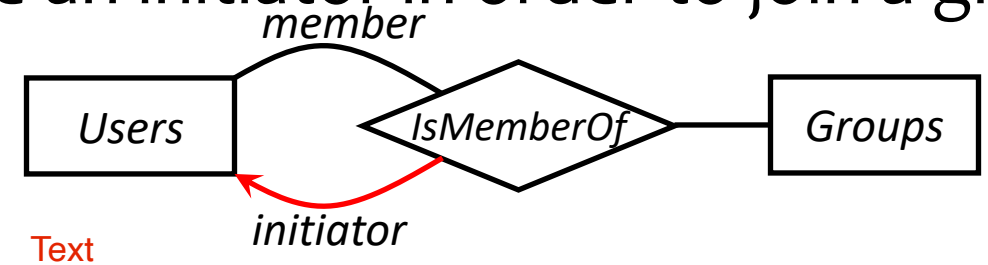


n-ary relationships

Initiator:
Someone who invites you to a group

- Example: a user must have an initiator in order to join a group

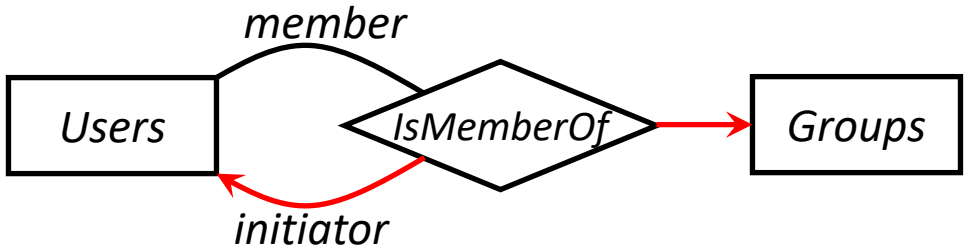
To say someone is a member of a group, you also need to mention the initiator



Text

Rule for interpreting an arrow into entity set E in an n-ary relationship:

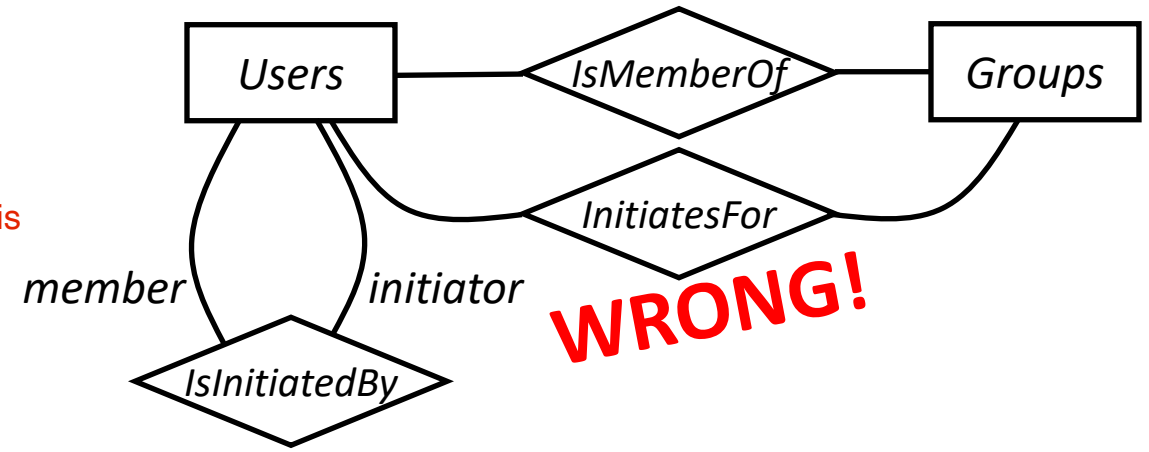
- Pick one entity from each other entity set (excluding only E); together they can relate to $\leq$ one entity in E
- Exercise: hypothetically, what do these two arrows imply?



You can only initialize a member for one group at most once.

n-ary versus binary relationships

- Can we model n-ary relationships using just binary relationships, like this?



Ask professor to go over this example and detaill

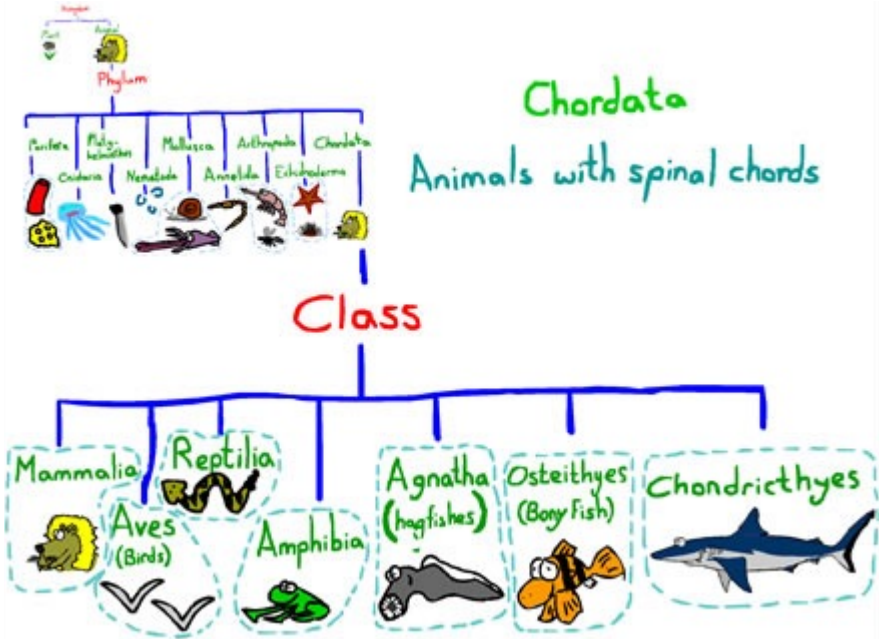
- No; for example:
  - Ralph is in both abc and gov
  - Lisa has served as initiator in both abc and gov
  - Ralph was initiated by Lisa in abc, but not by her in gov

The third bullet can not be captured in the diagram
If Lisa initiates Ralph, we do not know for which group it was done for.

Next: two special relationships

… is part of/belongs to …

… is a kind of …

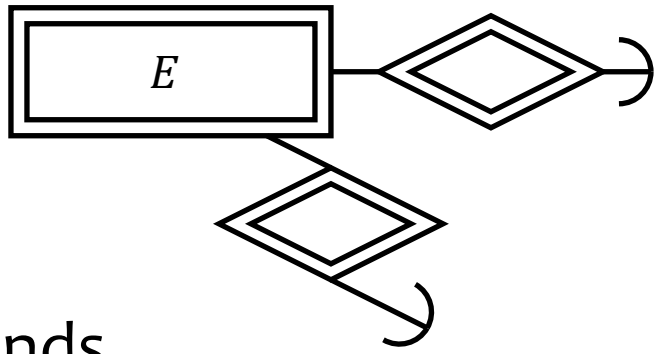


http://blogs.library.duke.edu/renovation/files/2012/08/Rubenstein-Library-First-Floor-Floorplan.jpg
http://www.sharky-jones.com/Sharkyjones/Artwork/taxonomy%20artwork/Class1.jpg

Weak entities & supporting relationships

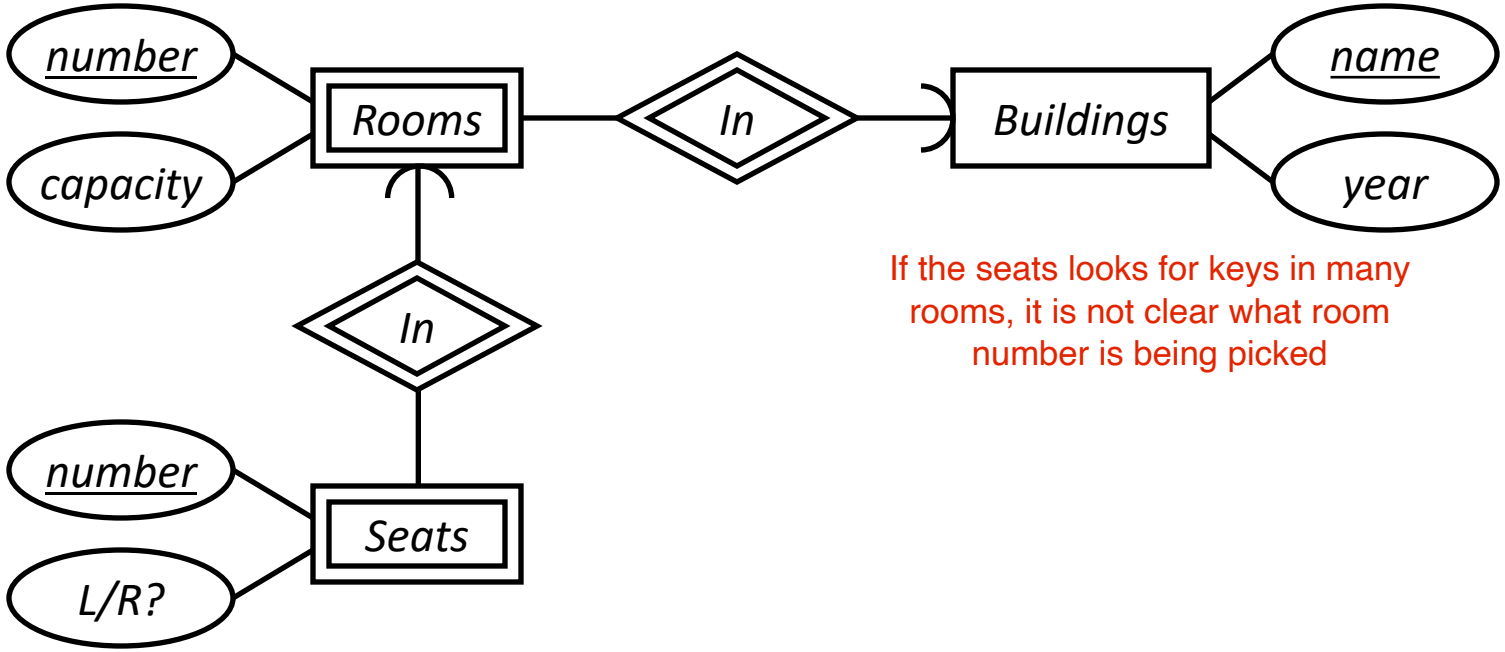
Sometimes, an entity's identity depends on some others'

- The key of a weak entity set E comes not completely from its own attributes, but from the keys of one or more other entity sets
  - E must link to them via many-one or one-one relationship sets
- Example: *Rooms* inside *Buildings* are partly identified by *Buildings*' name
- A weak entity set is drawn as a double rectangle
- The relationship sets through which it obtains its key are called supporting relationship sets, drawn as double diamonds



Weak entity set examples

- Seats in rooms in building

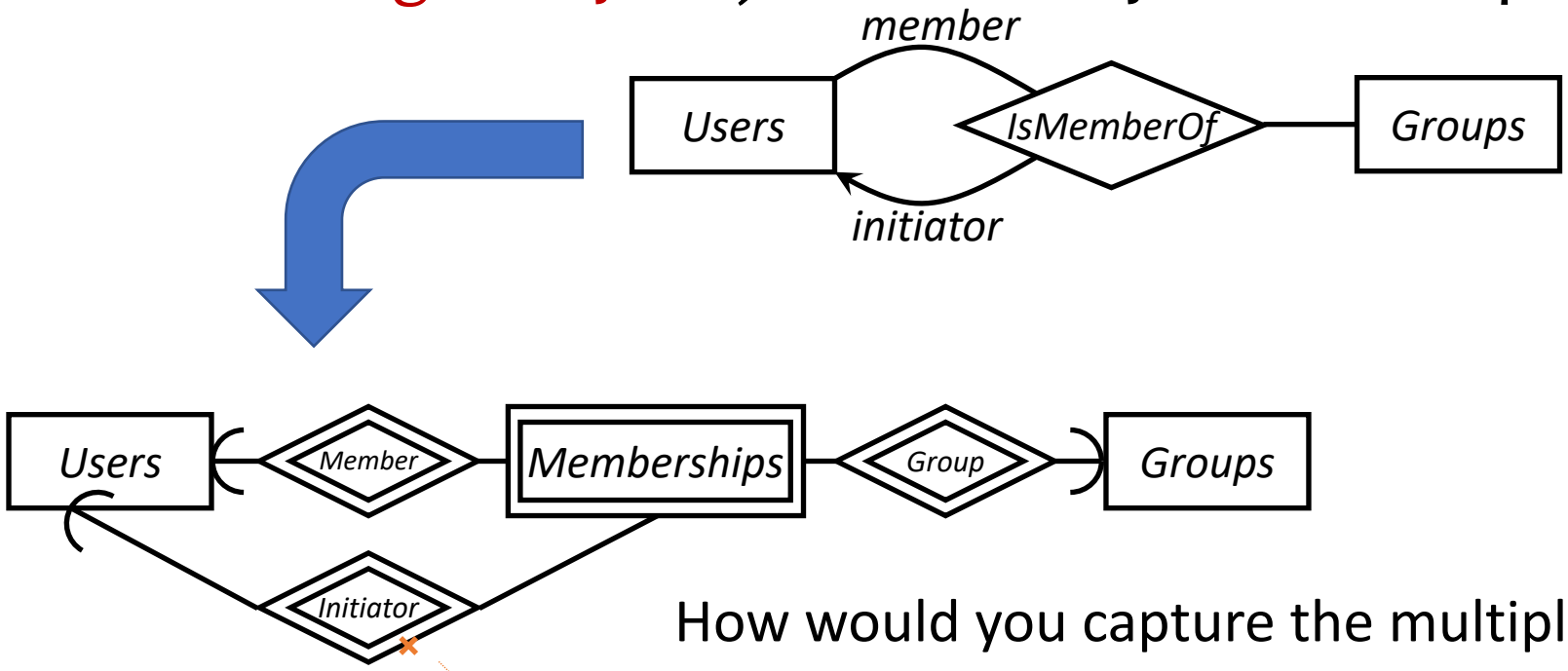


If the seats looks for keys in many rooms, it is not clear what room number is being picked

- Why must double diamonds be many-one/one-one?
  - With many-many, we would not know which entity provides the key value!

Remodeling n-ary relationships

- An n-ary relationship set can be replaced by a weak entity set (called a connecting entity set) and n binary relationship sets



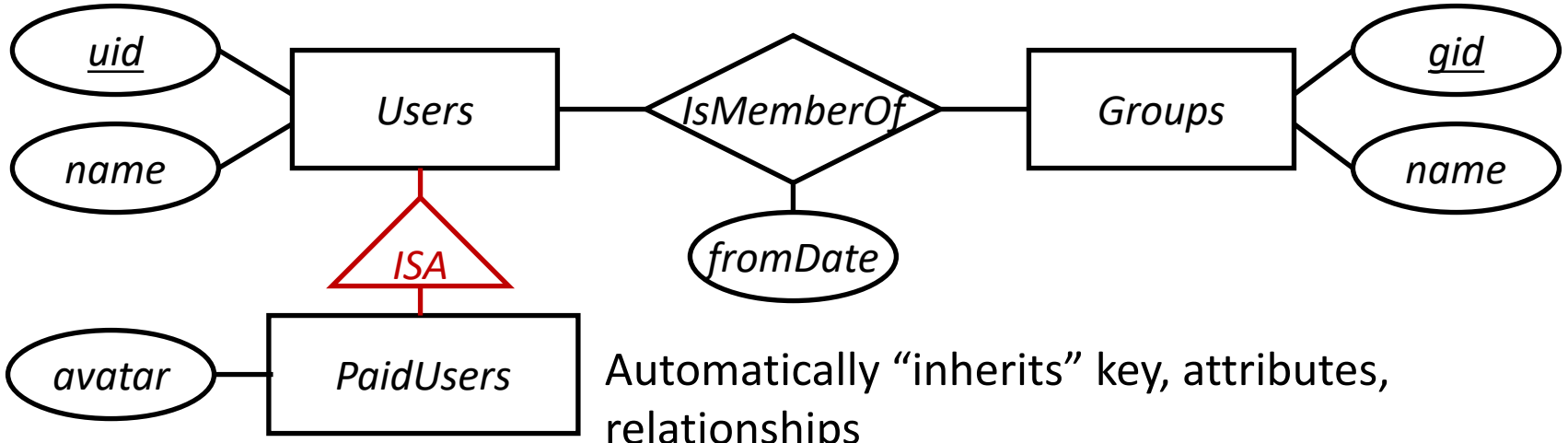
How would you capture the multiplicity constraint for *IsMemberOf* now?

Make Initiator regular instead of supporting

A membership is composed of exactly one member, initator, and group

ISA relationships

- Similar to the idea of subclasses in object-oriented programming: subclass = special case, fewer entities, and possibly more properties
  - Represented as a triangle (direction is important)
- Example: paid users are users, but they also get avatars (yay!)



Automatically "inherits" key, attributes, relationships

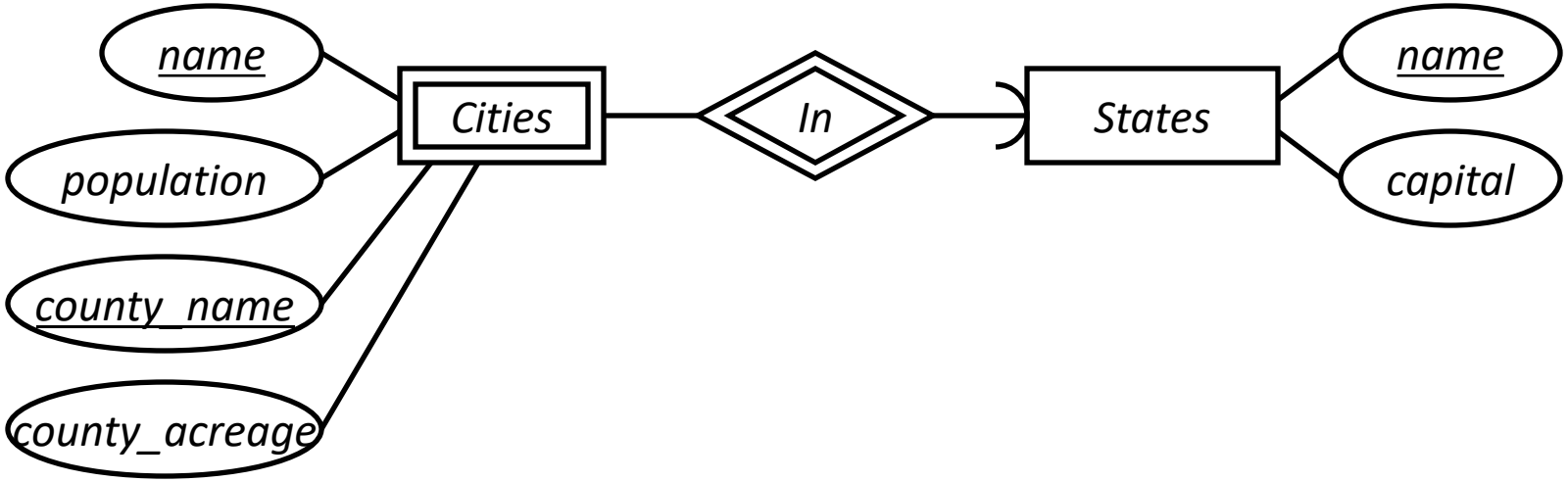
Summary of E/R concepts

- Entity sets
  - Keys
  - Weak entity sets
- Relationship sets
  - Attributes of relationships
  - Multiplicity
  - Roles
  - Binary versus n-ary relationships
    - Modeling n-ary relationships with weak entity sets and binary relationships
  - ISA relationships

Case study 1

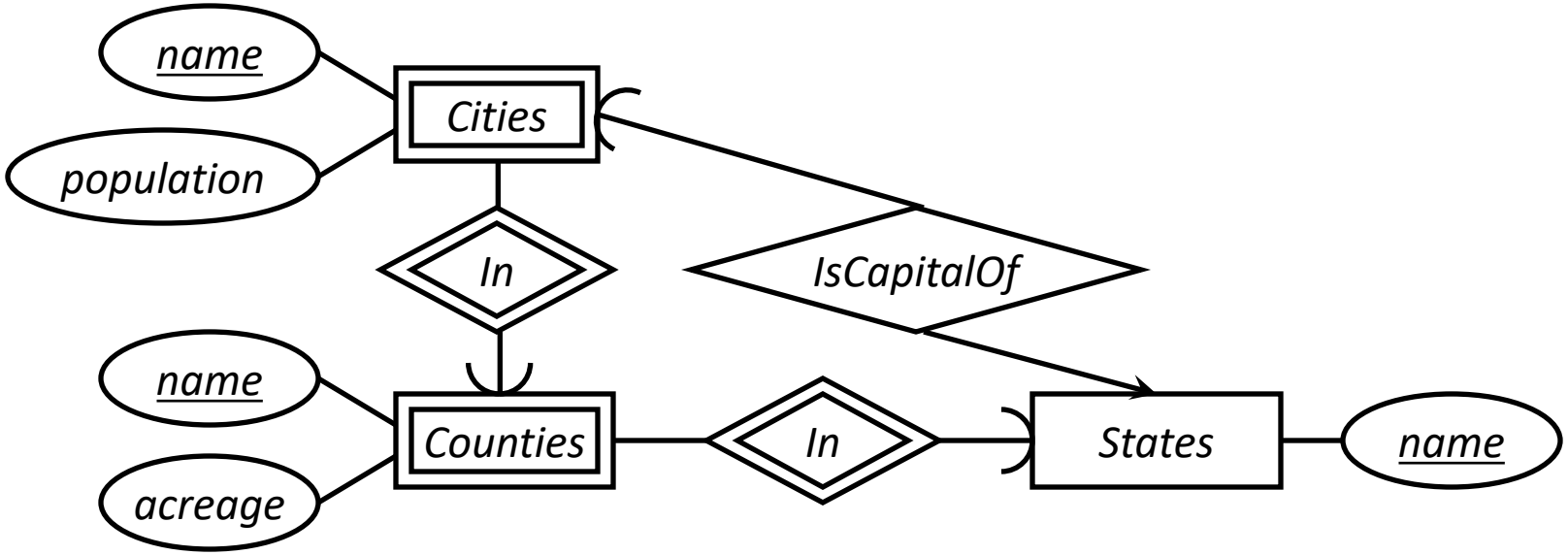
- Design a database representing cities, counties, and states
  - For each state, record name and capital (city)
  - For each county, record name, acreage, and state it's in
  - For each city, record name, population, and county/state it belongs to
- Assume the following:
  - Names of states are unique
  - Names of counties are only unique within a state
  - Names of cities are only unique within a county
  - A city is always in a single county
  - A county is always in a single state

Case study 1: first design



- County acreage information is repeated for every city in the county
  - ☞Redundancy is bad (why?)
- State capital should really be a city
  - ☞Should "reference" entities through explicit relationships

Case study 1: second design

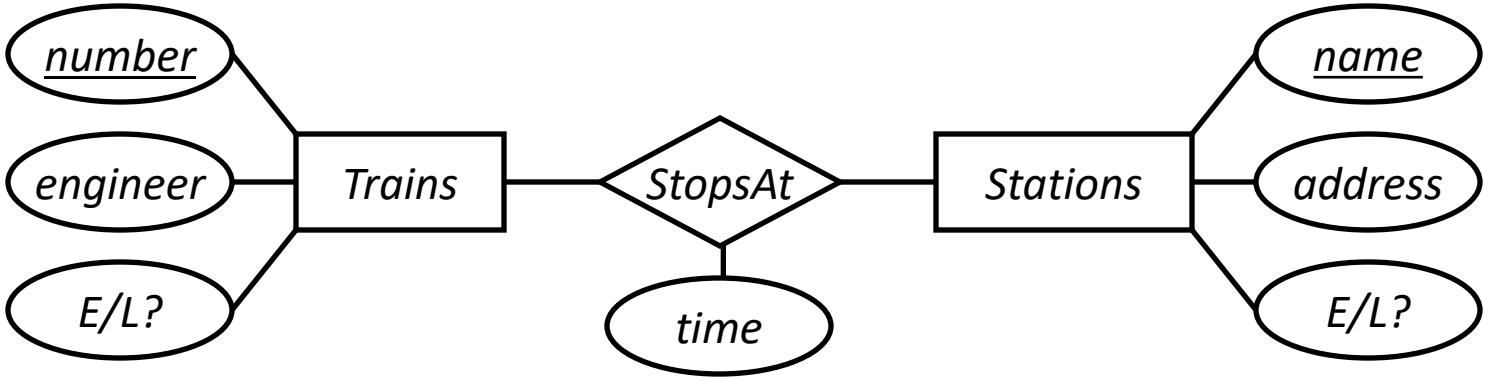


- Technically, nothing in this design prevents a city in state X from being the capital of another state Y, but oh well…

Case study 2

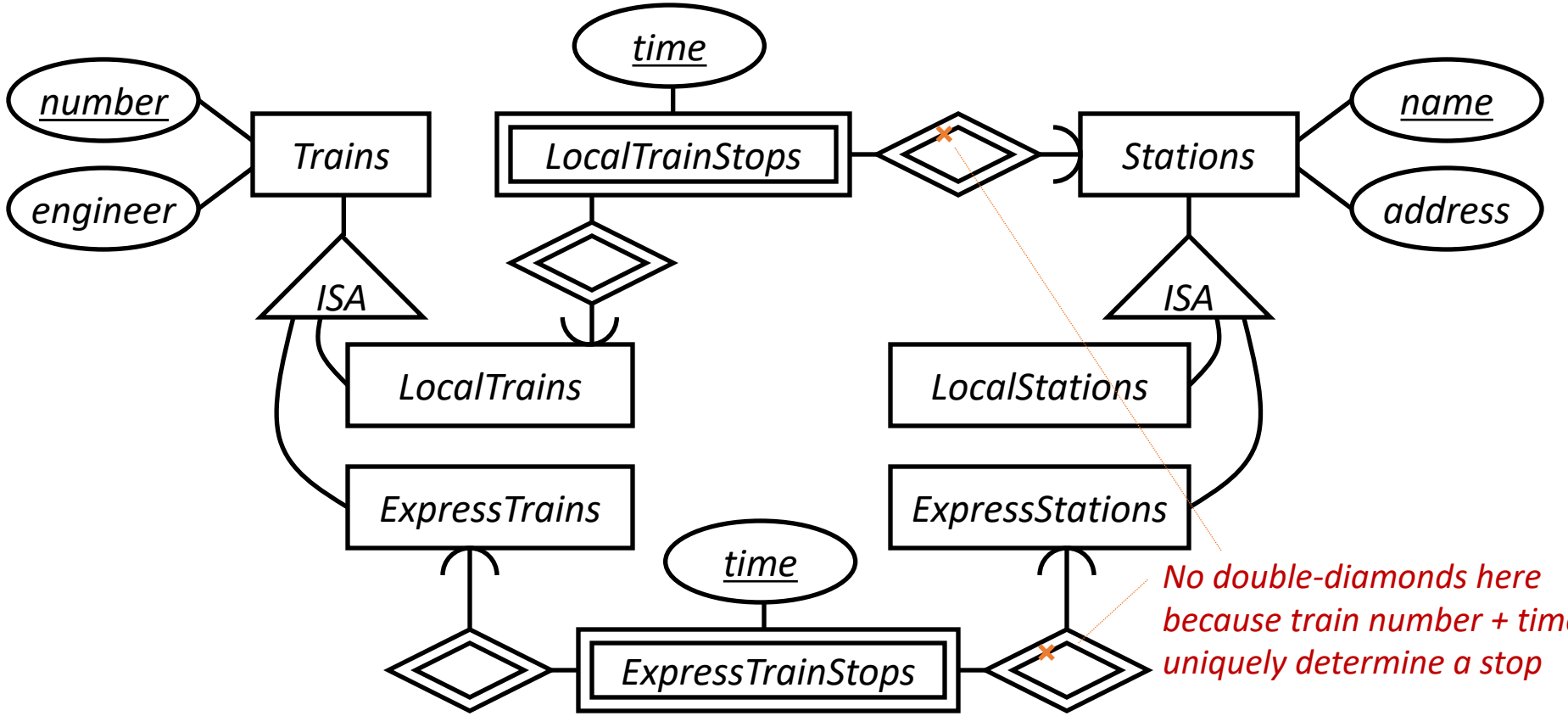
- Design a database consistent with the following:
  - A station has a unique name and an address, and is either an express station or a local station
  - A train has a unique number and an engineer, and is either an express train or a local train
  - A local train can stop at any station
  - An express train only stops at express stations
  - A train can stop at a station for any number of times during a day
  - Train schedules are the same every day

Case study 2: first design



- Nothing in this design prevents express trains from stopping at local stations
  - ☞We should capture as many constraints as possible
- A train can stop at a station only once during a day
  - ☞We should not introduce unintended constraints

Case study 2: second design



Is the extra complexity worth it?

What's Next

- Next class:
  - More about Database design
- Assignment 1 due 5/29