Lecture 9: Database Design (E/R Model)

CS348 Spring 2025:
Introduction to Database Management

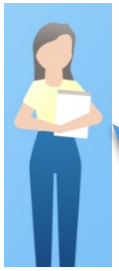
Instructor: **Xiao Hu**

Sections: 001, 002, 003

Announcements

- Assignment 2 released on Learn
  - Due on 11:59 PM Jun 24
  - Coverage: Lectures 4 – 12
  - Content: SQL and Database design
  - Crowdmark and Marmost are already open

Motivating Example



I want to have a registrar's database. Can you help build a database for managing courses, students, and faculty members ?

It has these requirements …

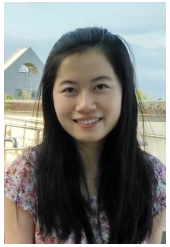
Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1. Most course sections are taught on-site, but a few are taught at off-site locations.

Students have student numbers and names. Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.

Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.

A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

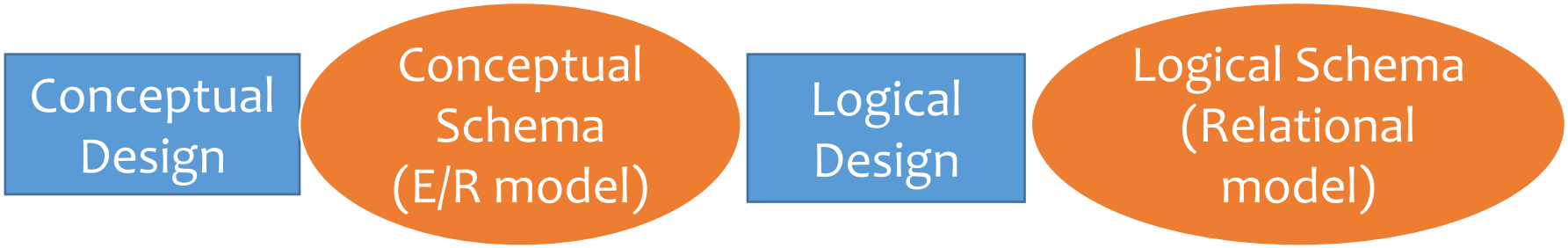
I know how to use SQL now!



What tables do you want me to create? What are the primary keys, constraints, queries, ……?

We still need to learn about database design ☺

Database Design



- Understand the real-world domain being modeled and constrained
- Entity-Relationship (E/R) model
- Translate E/R diagram to Relational data model
- Create DBMS schema (DDL in SQL)

Outline

Lectures 9 and 10

- Entity-Relationship (E/R) model
- Translating E/R diagram to relational schema

Lectures 11 and 12

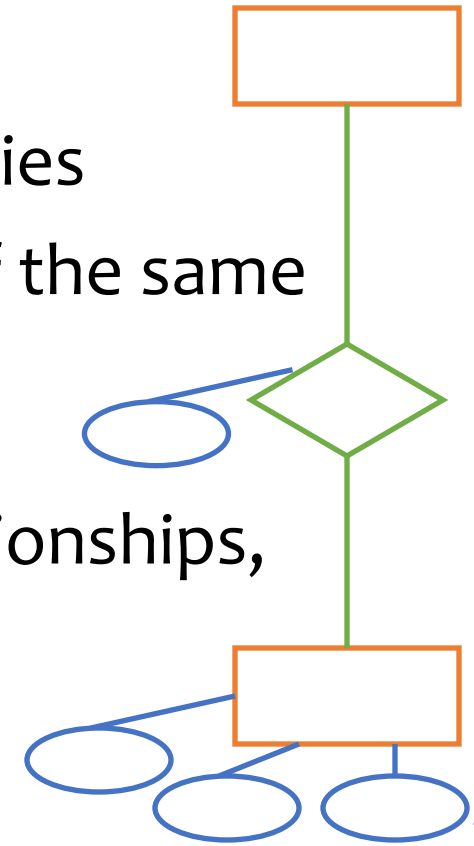
- Principles behind relational schema

Entity-relationship (E/R) model

- Historically and still very popular
- Primarily a design model -- not directly implemented by DBMS
- Designs represented by E/R diagrams
  - We use E/R diagram styles slightly different from the one covered by the textbook.
  - There are other styles/extensions

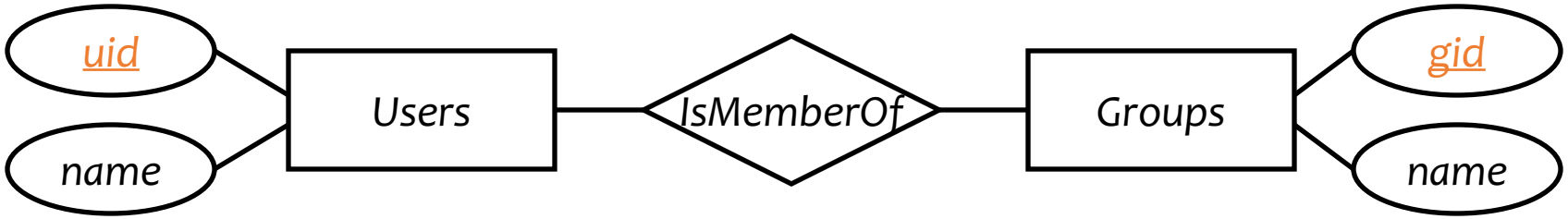
E/R basics

- 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
  - Represented as a rectangle
- Relationship: an association among entities
- Relationship set: a set of relationships of the same type (among same entity sets)
  - Represented as a diamond
- Attributes: properties of entities or relationships, like attributes of tuples or objects
  - Represented as ovals



An example E/R diagram

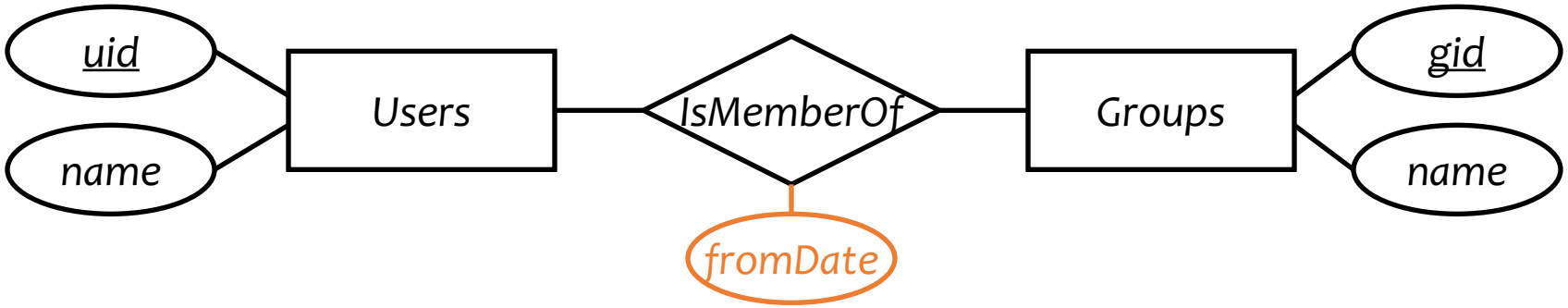
- Users are members of groups



- A primary 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
  - underlining all attributes in the primary key

Attributes of relationships

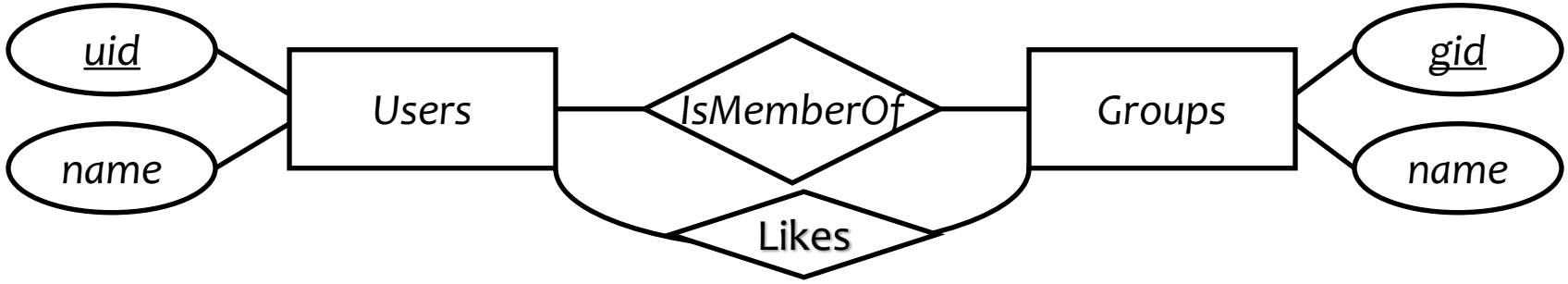
- Example: a user belongs to a group since a particular date



- 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*!

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



More on relationships

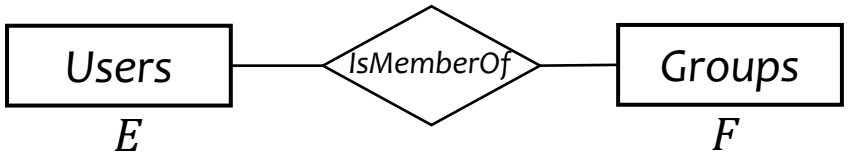
- There could be multiple relationship sets between the same entity sets
  - Example: *Users IsMemberOf Groups*; *Users Likes Groups*
- In a relationship set, each relationship is uniquely identified by the entities it connects
  - Example: Between "Bart" and "Dead Putting Society", there can be at most 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?

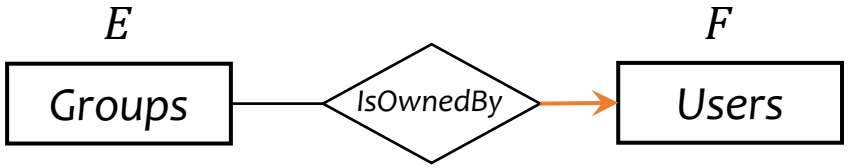
☞Make an entity set of *MembershipRecords*

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:
    Each group has many users;
    Each user belongs to many groups.

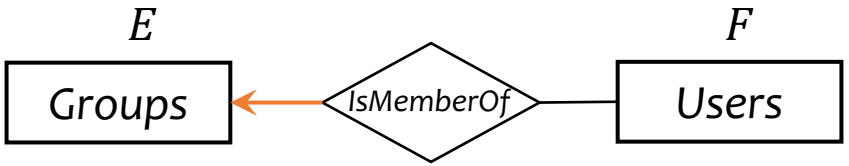


- Many-one: Each entity in E is related to 0 or 1 entity in F, but each entity in F is related to 0 or more in E
  - Example: *Each group is owned by at most 1 user; Each user can own many groups*

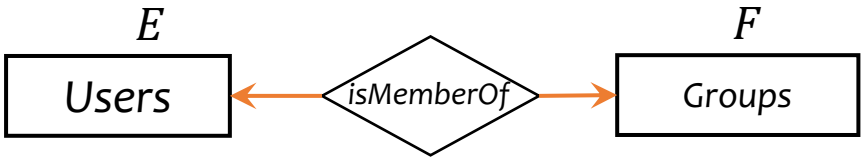


Multiplicity of relationships

- E and F: entity sets
- One-many: Each entity in E is related to 0 or more entities in F, but each entity in F is related to 0 or 1 in E
  - Example: *Each group has many users; Each user belongs to at most 1 group*

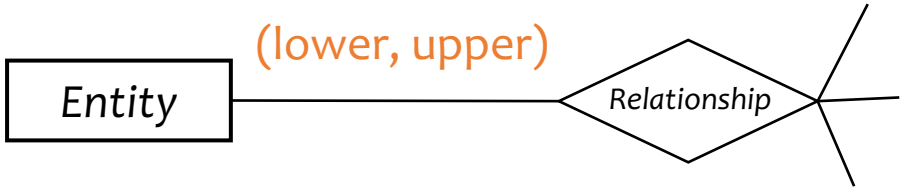


- One-one: Each entity in E is related to 0 or 1 entity in F and vice versa
  - Example: *Each group has at most 1 user; Each user belongs to at most 1 group*

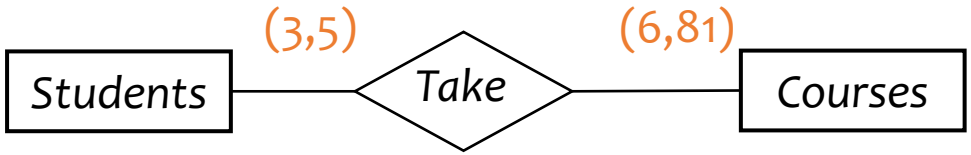


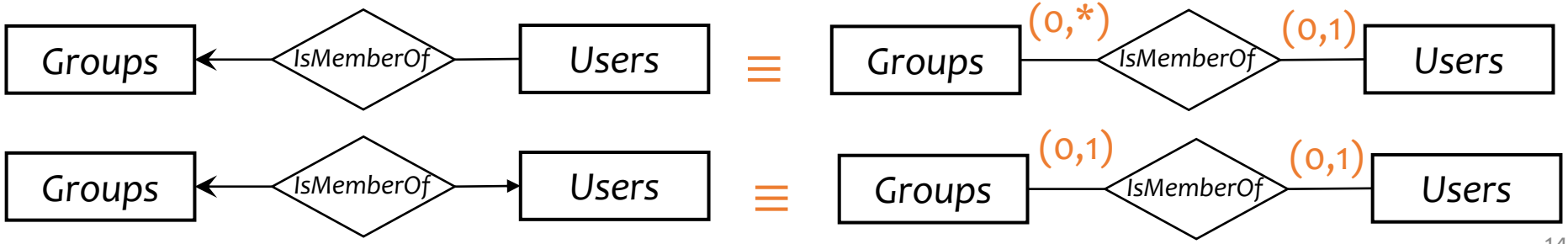
General cardinality constraints

- General cardinality constraints determine lower and upper bounds on the number of relationships of a given relationship set in which a component entity may participate



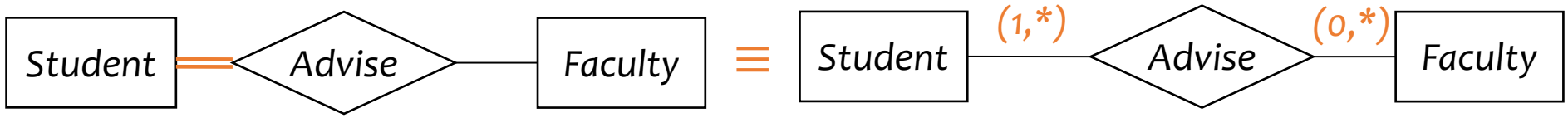
- Example:





Total vs. Partial participation

- Total participation (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
- Partial participation: some entities may not participate in any relationship in the relationship set
- Example:



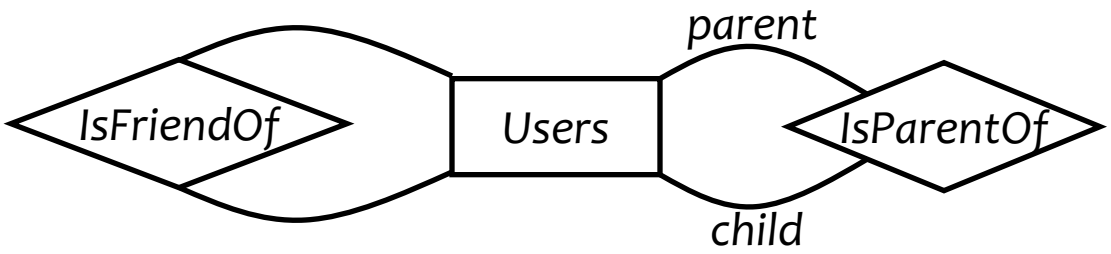
  - Every student must be advised by at least one faculty member
  - Some faculty member may not advise any student

Roles in relationships

- An entity set may participate more than once in a relationship set

☞ May need to label edges to distinguish roles

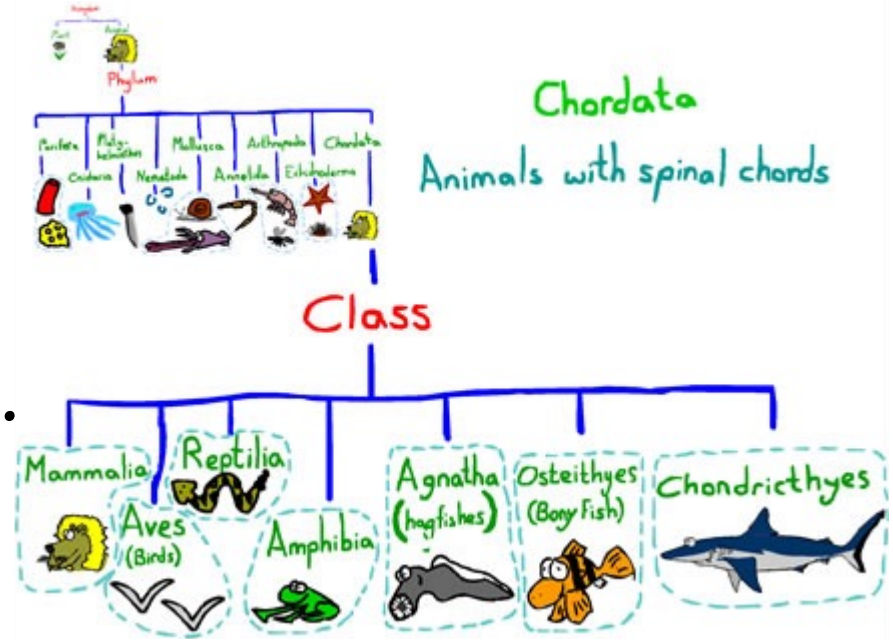
- Examples
  - Users may be parents of others; label needed
  - Users may be friends of each other; label not needed



Next: two special relationships

… is part of/belongs to …

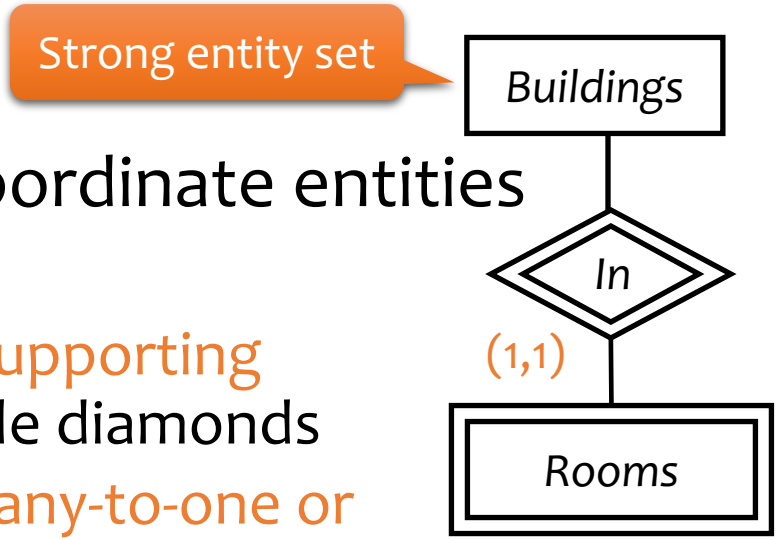
… is a kind of …



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

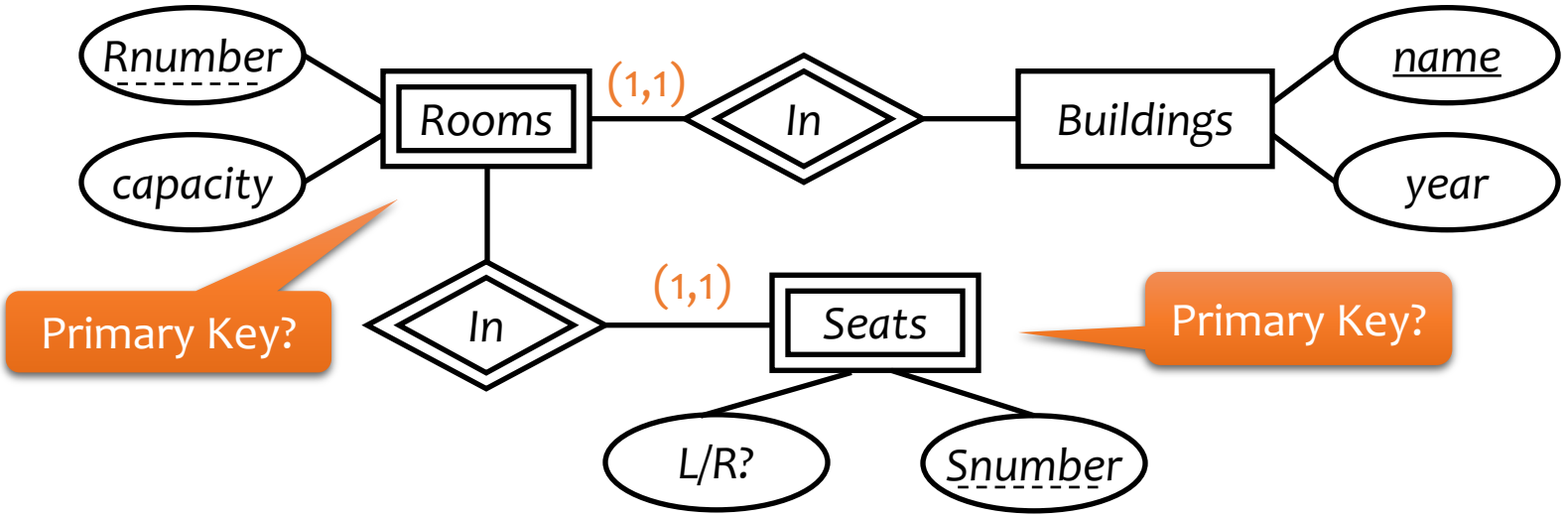
Weak entity sets

- If entity E's existence depends on entity F, then
  - F is a dominant entity
  - E is a subordinate entity
  - Example: *Rooms* inside *Buildings* are partly identified by *Buildings*' name



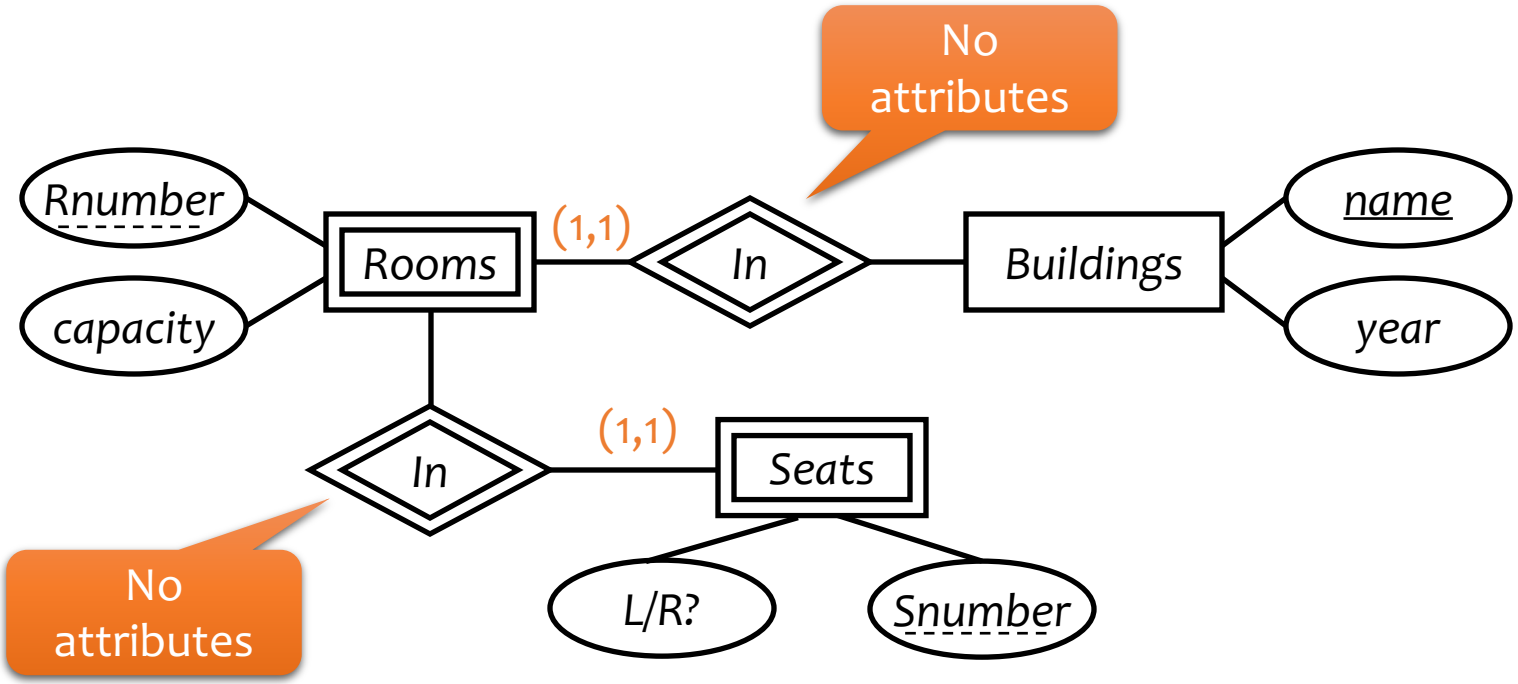
- Weak entity set: containing subordinate entities
  - Drawn as a double rectangle
  - The relationship sets are called supporting relationship sets, drawn as double diamonds
  - A weak entity set must have a many-to-one or one-to-one + total participation relationship to a distinct entity set

Weak entity set examples



- Discriminator of a weak entity set (dashed underline): a set of attributes that distinguish subordinate entities of the set, for a particular dominant entity
- Primary key of a weak entity set: discriminator + primary key of the dominant entities

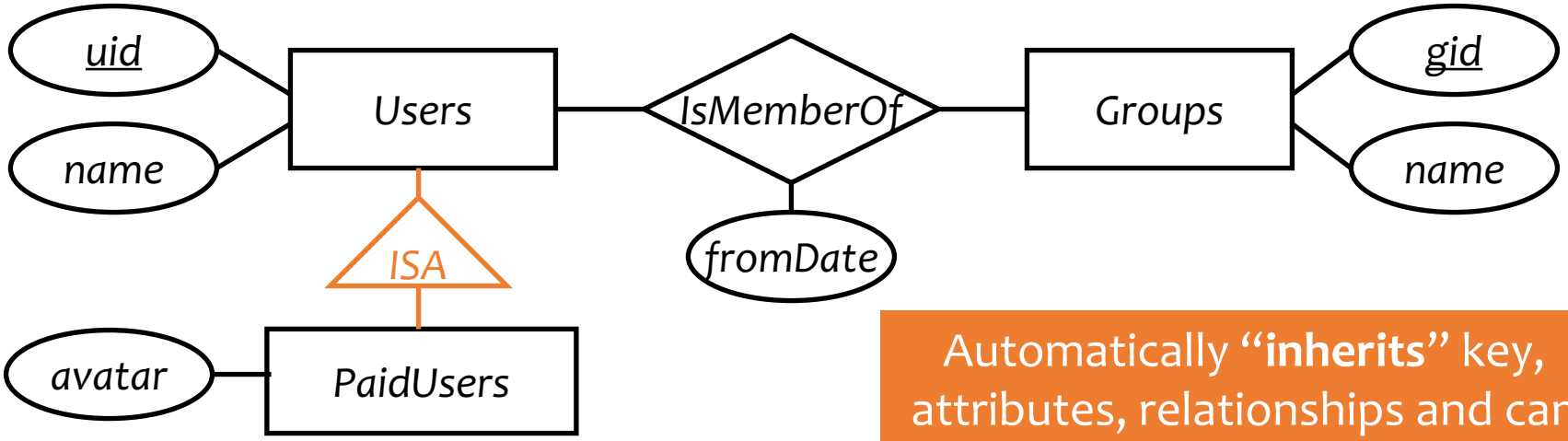
Weak entity set examples



- The supporting relationship should not have attributes
  - These can be associated with the weak entity set
  - Each weak entity participates in exactly one supporting relationship

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!)



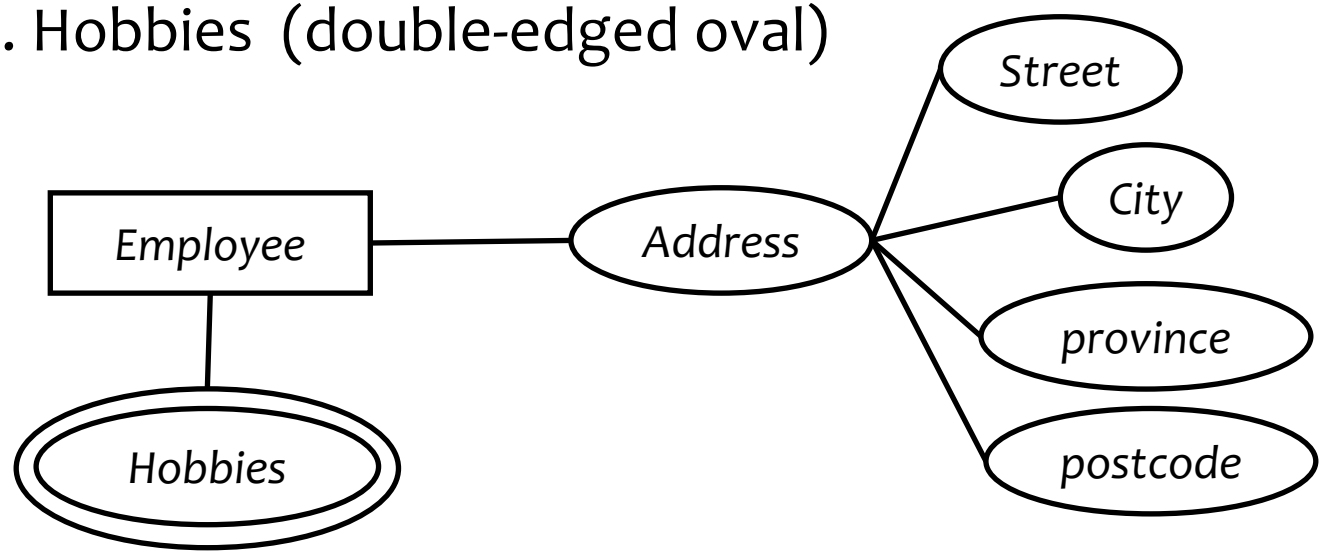
Automatically "**inherits**" key, attributes, relationships and can participate in other relationships

Summary of E/R concepts

- Entity sets
  - Keys
  - Weak entity sets
- Relationship sets
  - Attributes of relationships
  - Multiplicity
  - Roles
  - Supporting relationships (related to weak entity)
  - ISA relationships
- Other extensions:
  - Composite and Multi-valued attributes
  - Aggregation

Composite and multi-valued attributes

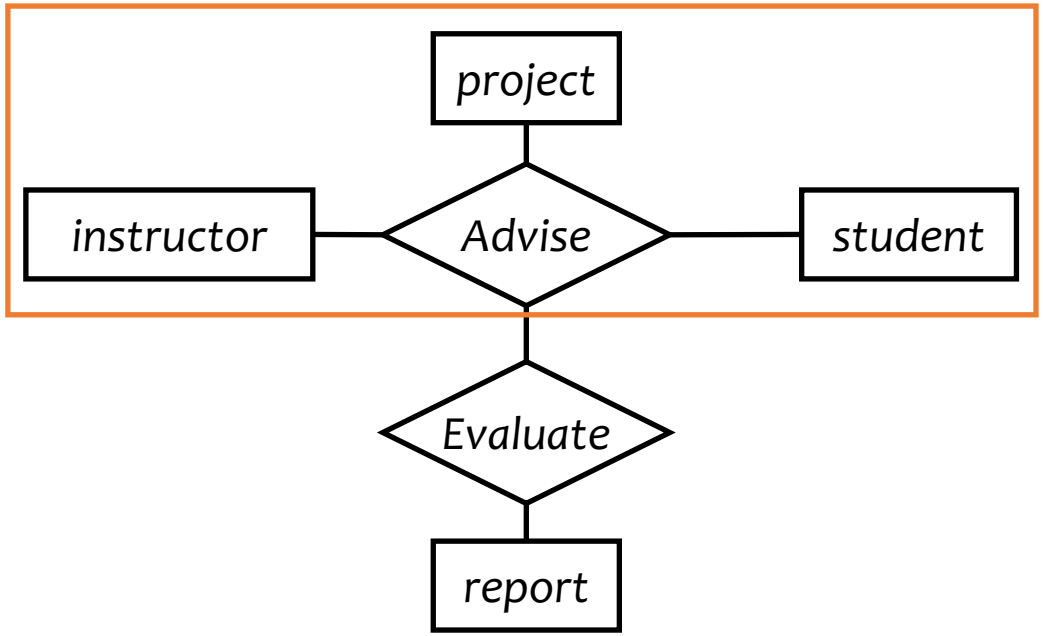
- Composite attributes: composed of fixed number of other attributes
  - E.g. Address is composed of street, city, province, and postcode
- Multi-valued attributes: attributes that are set-valued
  - e.g. Hobbies (double-edged oval)



Aggregation

- How to model the relationships between relationships?
  - Relationships can be viewed as high-level entities
- Example: Each instructor advising a student on a project is required to fill out a monthly evaluation report

Example of 3-ary relationship



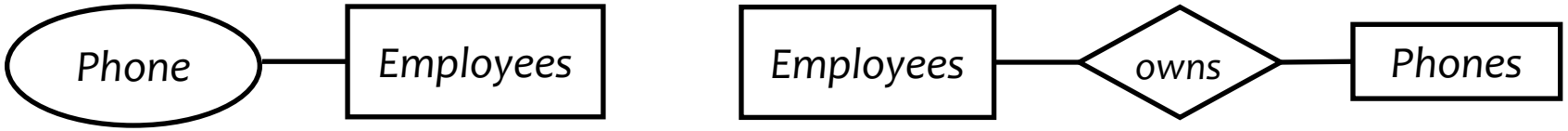
Designing an E/R schema

- Usually many ways to design an E/R schema

- Questions to consider:
  - use attribute or entity set?
  - use entity set or relationship set?
  - degrees of relationships?
  - extended features?

Attributes or Entity Sets?

- How to model employees' phones?

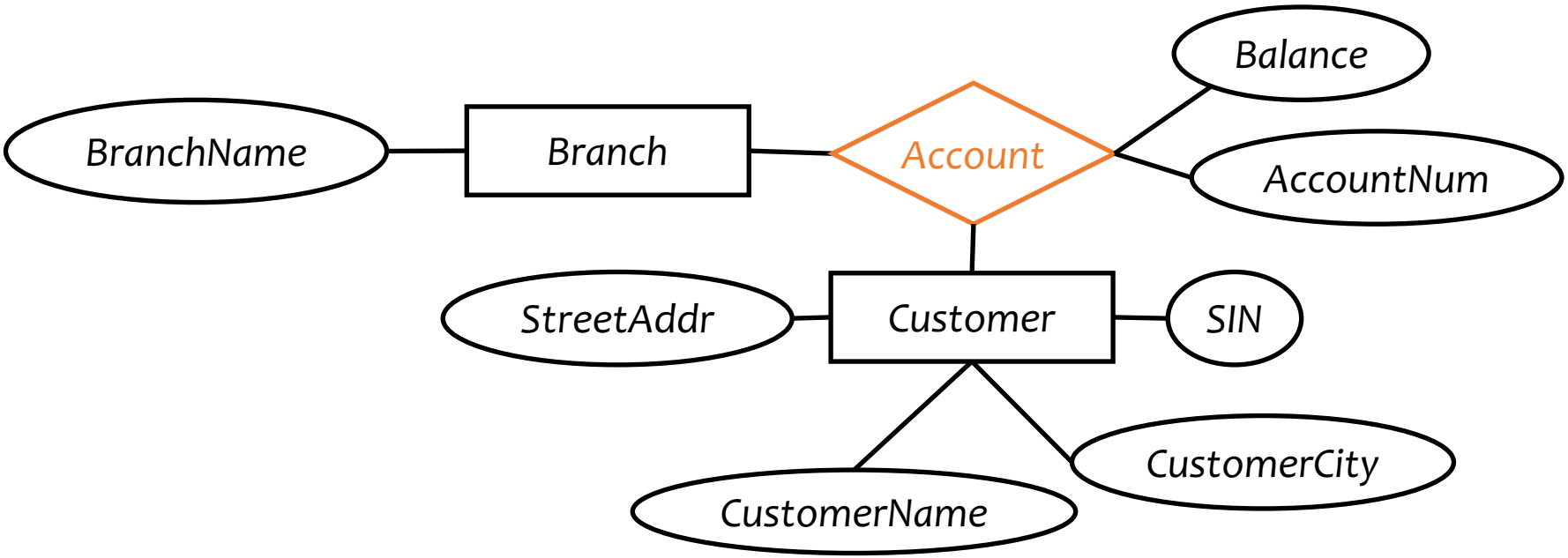


- Rules of thumb:
  - Do we need to maintain other information (phone number, model, start-date, price etc)?
  - Can several of its kind belong to a single entity?
  - Can it be missing from the employee set's entities?
  - Can it be shared by different entities?

- A "yes" to any of the above suggests a new entity set

Entity Sets or Relationships?

- Customers have a bank account in a bank branch
- Instead of representing accounts as entities, we could represent them as relationships



A simple methodology

Step 1: Identify entity sets

Step 2: Identify relationship sets and participating entity sets

Step 3: Identify attributes of entity and relationship sets

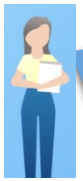
Step 4: Identify relationship types and existence dependencies

Step 5: Specify general cardinality constraints, keys, and discriminators

Step 6: Draw E/R diagram

- For each step, maintain a log of assumptions and of restrictions imposed by your choices

Case study 1



Design a database representing cities, counties, and states

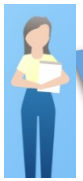
- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

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 located in a single county
- A county is always located in a single state

What are the entity sets, relationship sets, and their attributes? What are the types of relationships and cardinality constraints, keys, discriminators?

Case study 1



Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

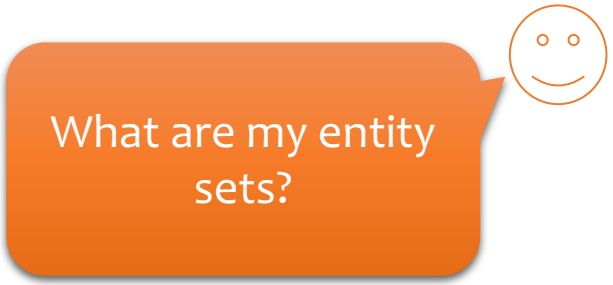
Assume the following:

- 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 located in a single county
- A county is always located in a single state

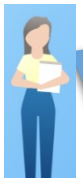
Cities

Counties

States



Case study 1

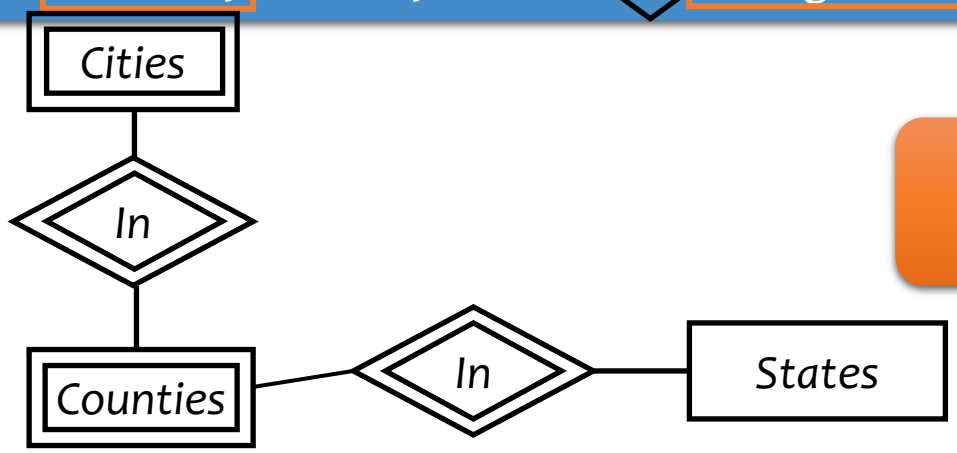


Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

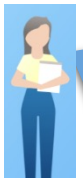
Assume the following:

- 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 located in a single county
- A county is always located in a single state



What are my relationship sets?

Case study 1



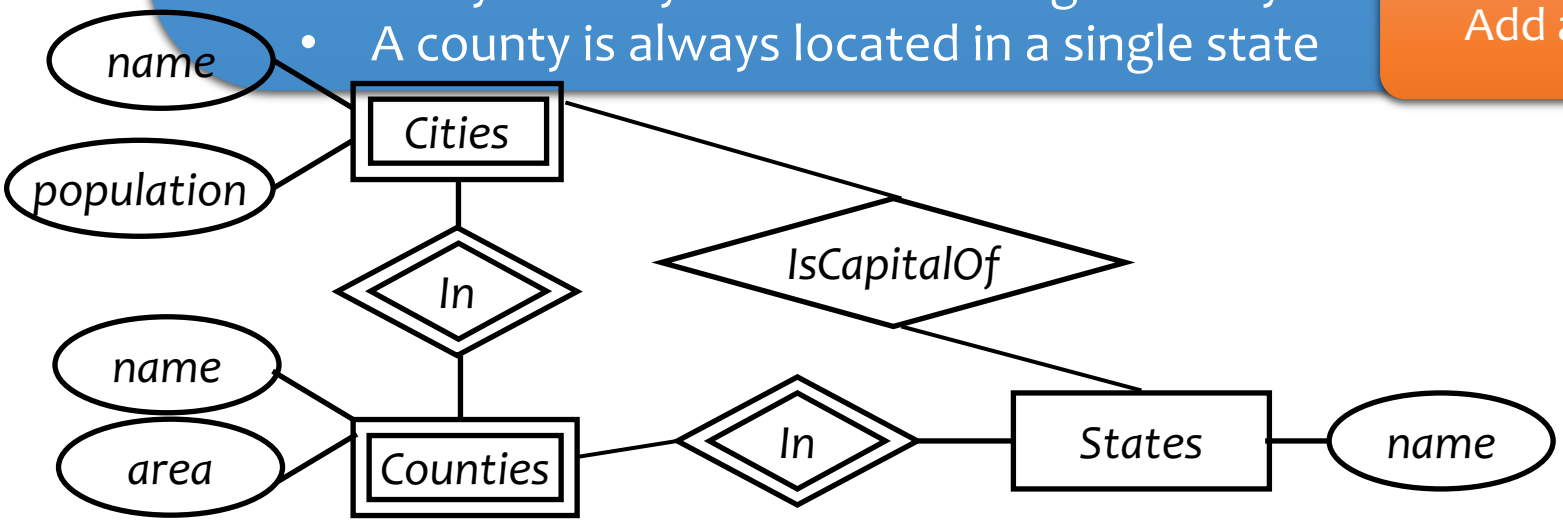
Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

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 located in a single county
- A county is always located in a single state

Add attributes!



Case study 1

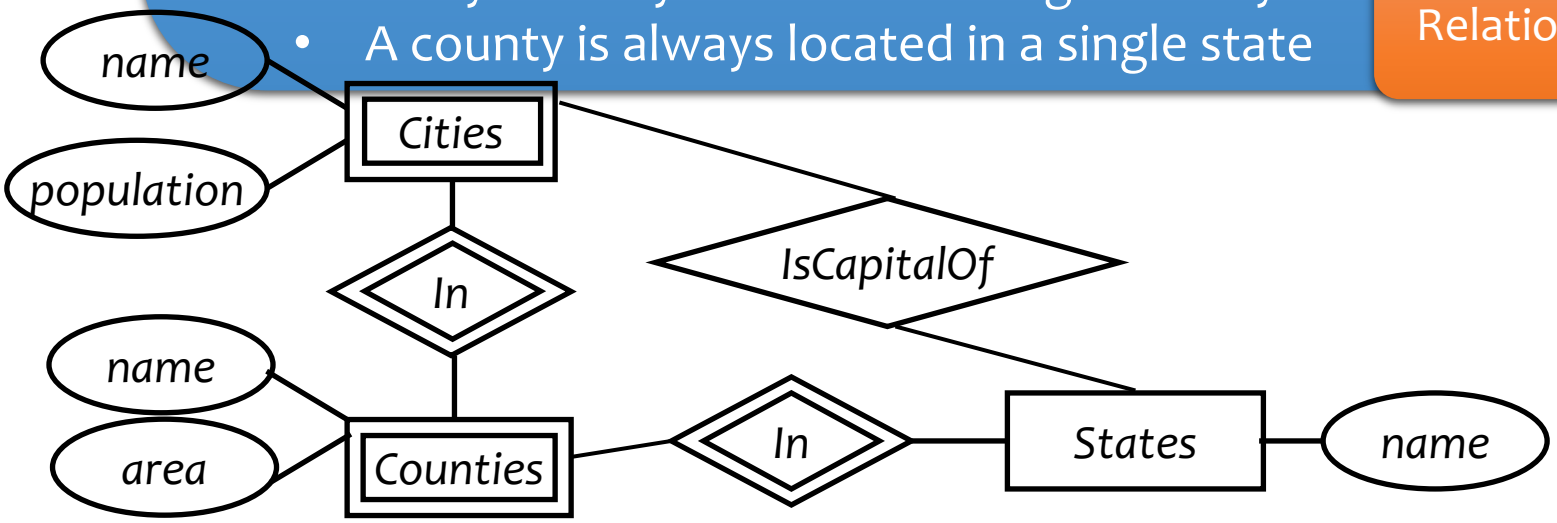
Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

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 located in a single county
- A county is always located in a single state

Relationship types?



Case study 1

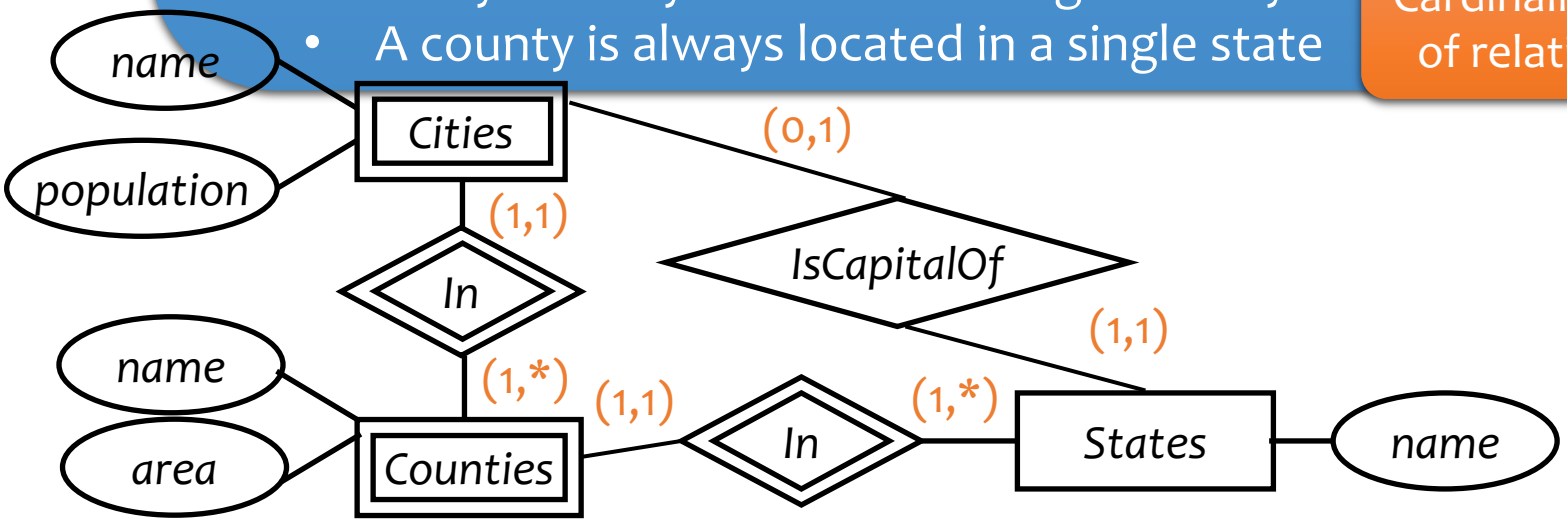
Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

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 located in a single county
- A county is always located in a single state

Cardinality constraints of relationship sets?



Case study 1

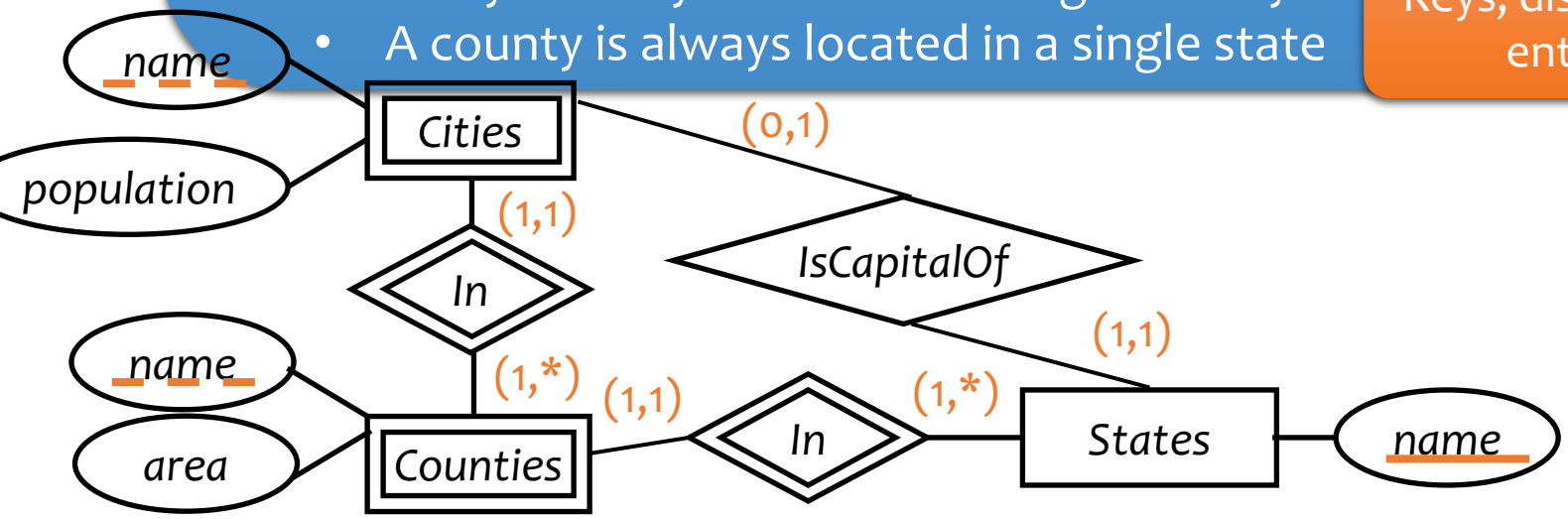
Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

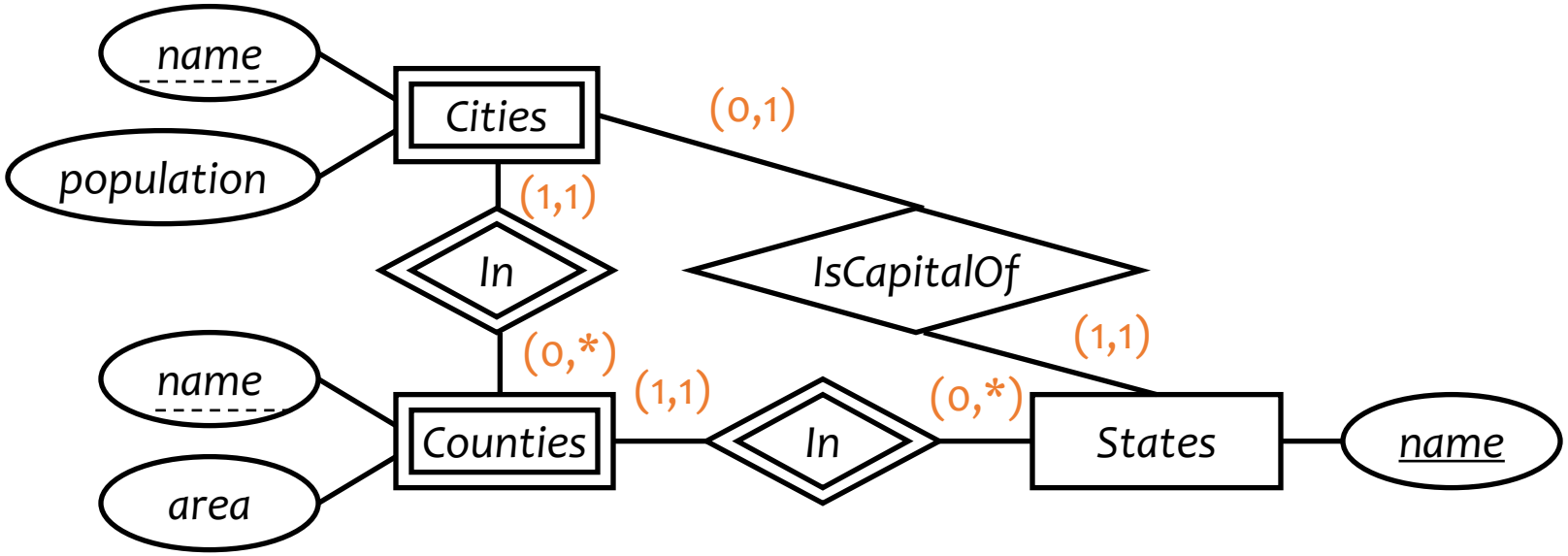
Assume the following:

- 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 located in a single county
- A county is always located in a single state

Keys, discriminator of entity sets?

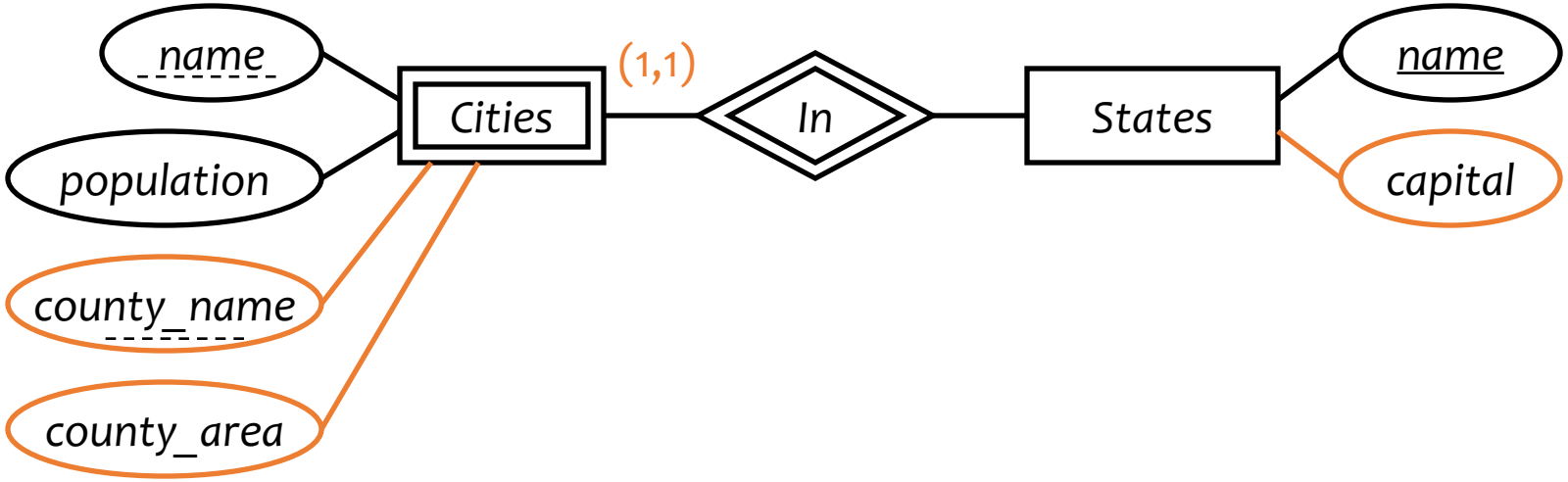


Case study 1: final design



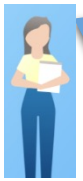
- Cities with the same name and population appearing in different counties are different entities
- Counties with the same name and area appearing in different states are different entities

Case study 1: why not good?



- County area 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

Take Home Exercise

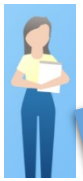


Design a database consistent with the following:

- 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 everyday

Can you draw a E/R diagram?

Take Home Exercise



A Registrar's Database:

Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.

Most course sections are taught on-site, but a few are taught at off-site locations.

Students have student numbers and names.

Each course section is taught by a professor. Professors have professor numbers and names. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.

Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.

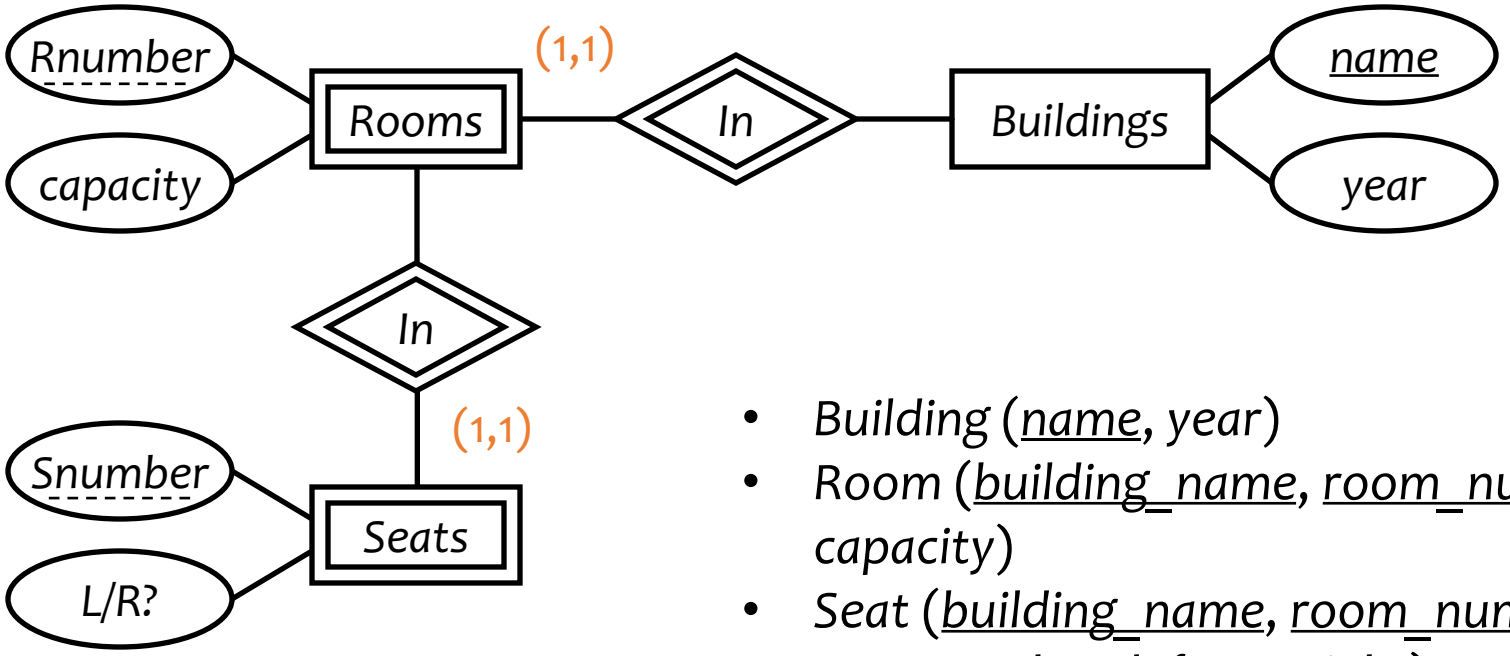
A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

Can you draw a E/R diagram?

What you have learned so far

- Entity-Relationship (E/R) model

- Next: Translating E/R to relational schema



- *Building* (<u>*name*</u>, *year*)
- *Room* (<u>*building_name*</u>, <u>*room_number*</u>, *capacity*)
- *Seat* (<u>*building_name*</u>, <u>*room_number*</u>, <u>*seat_number*</u>, *left_or_right*)