Relational Database Design using E/R

CS348 Spring 2024

Instructor: Sujaya Maiyya

Sections: **002 & 003 only**

Announcements

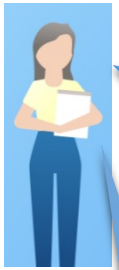
- June 11th lecture by Chao Zhang on converting ER diagrams to SQL scheme

- Next Thursday's June 13th lecture: ONLINE!
- Video link will be posted on Piazza.
  Must watch before Tue June 18th's class (follow up of online lecture)

- No office hours next weeks

- Assignment 2 will be released soon

Motivating Example



I want to have a registrar's database. Can you help?

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

Step 1: Understand the real-world domain being modeled

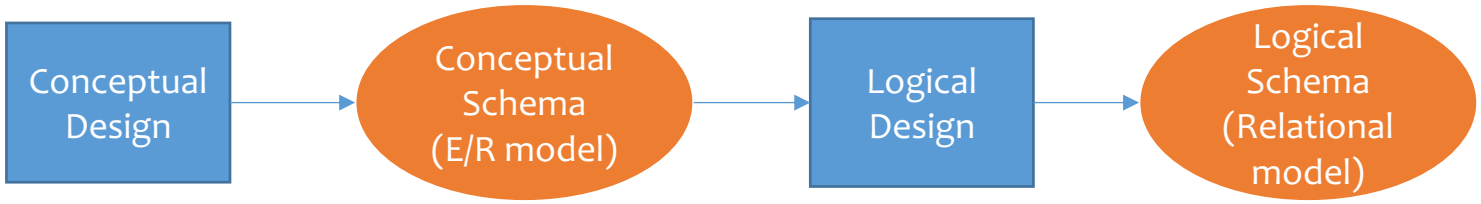
→Specify it using a database design model

- Entity/Relationship (E/R) model

Step 2: Translate specification to the data model of DBMS

- Relational

→ Create DBMS schema



Database Design

- Entity-Relationship (E/R) model
- Translating E/R to relational schema
- Relational design principles

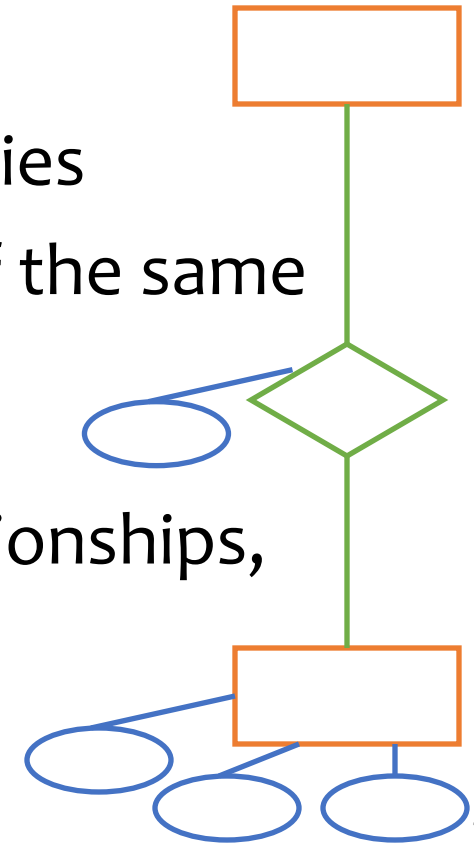
Lectures 9 & 10

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 book
  - There are other styles/extensions

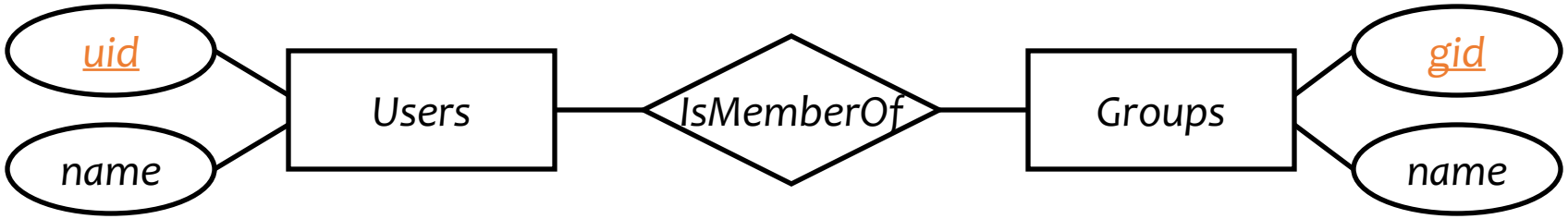
E/R basics

- 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

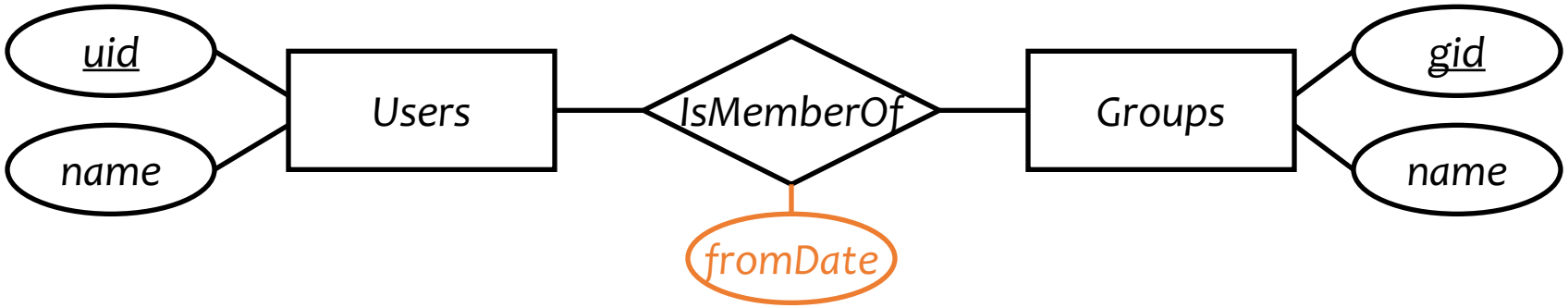
- Users are members of groups



- A 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

Attributes of relationships

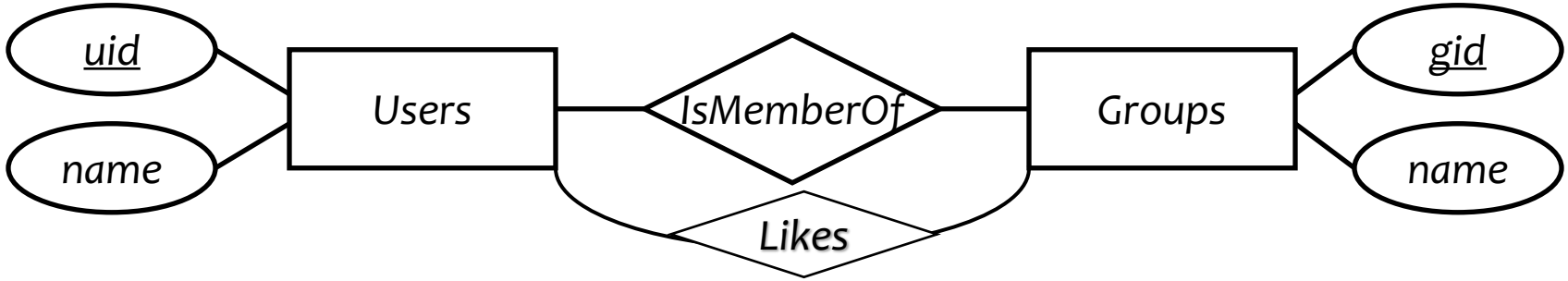
- Example: a user belongs to a group since a particular date



- 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

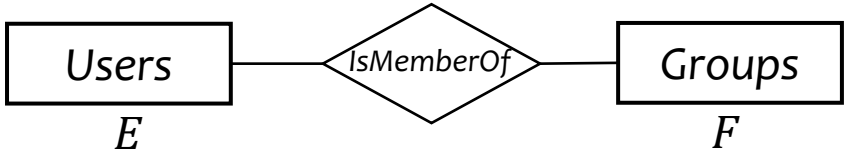
- There could be multiple relationship sets between the same entity sets
  - Example: *Users IsMemberOf Groups*; *Users Likes Groups*
- In a relationship set, each relationship is uniquely identified by the entities it connects
  - Example: Between Bart and "Dead Putting Society", there can be at most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

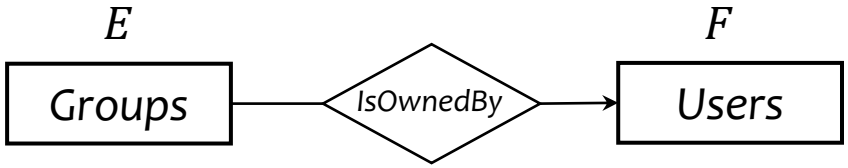
    ☞Make an entity set of *MembershipRecords*

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.

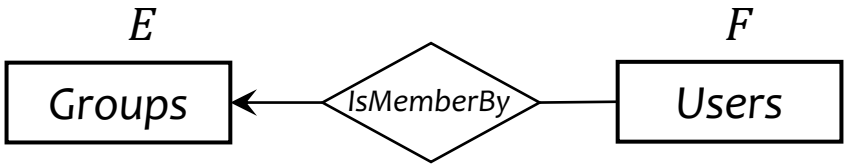


- Many-one: Each entity in E is related to 0 or 1 entity in F, but each entity in F is related to 0 or more in E
  - Example: *Each group is owned by at most 1 user; Each user can own many groups*

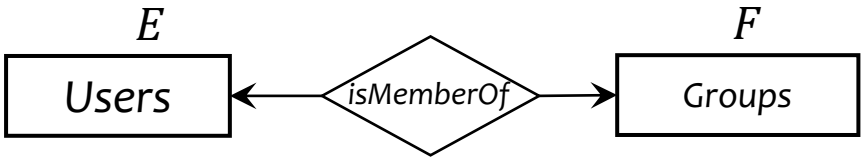


Multiplicity of relationships

- 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*



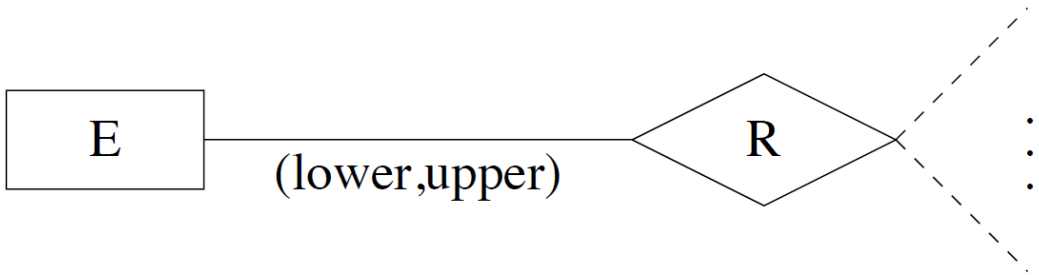
- One-one: Each entity in E is related to 0 or 1 entity in F and vice versa
  - Example: *Each group has at most 1 user; Each user belongs to at most 1 group*



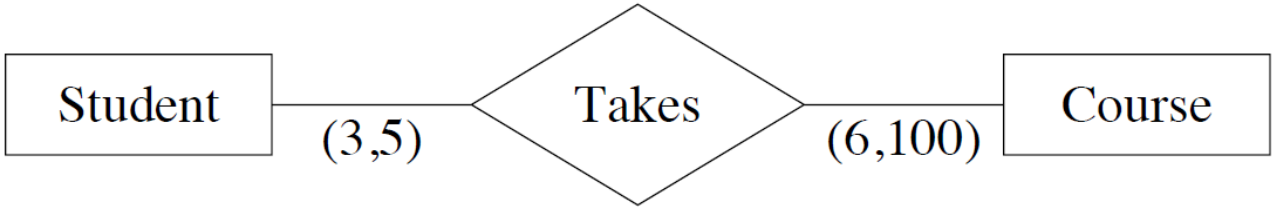
- "One" (0 or 1) is represented by an arrow ⟶

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

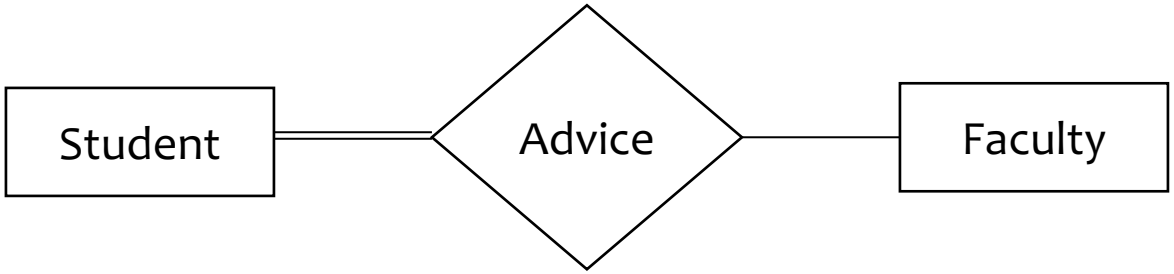


- Example:

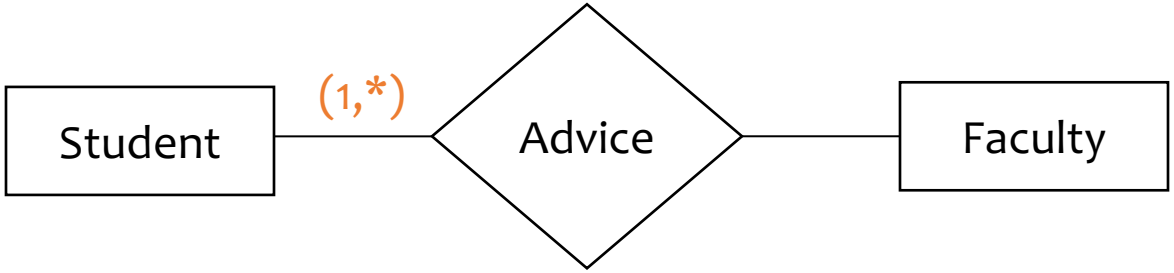


Total vs. partial participation

- 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



Every *Student* must participate in *Advice* (i.e., is advised by a faulty)
Some faculty may not advice any students

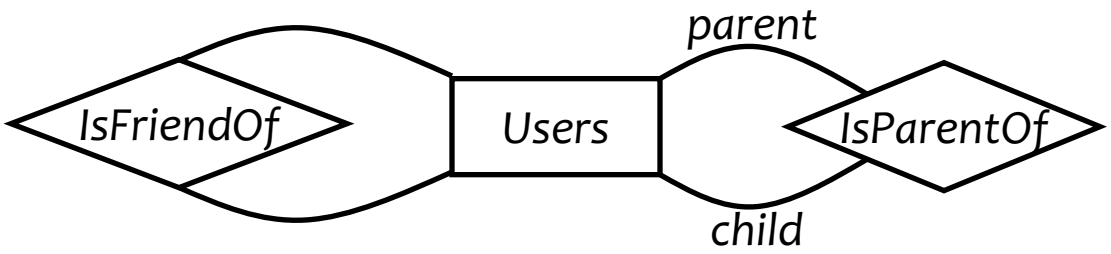


Roles in relationships

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

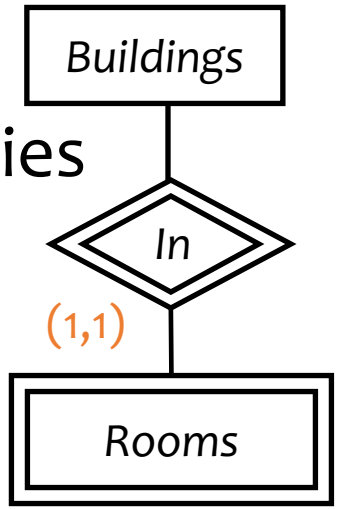
☞ May need to label edges to distinguish roles

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



Weak entity sets

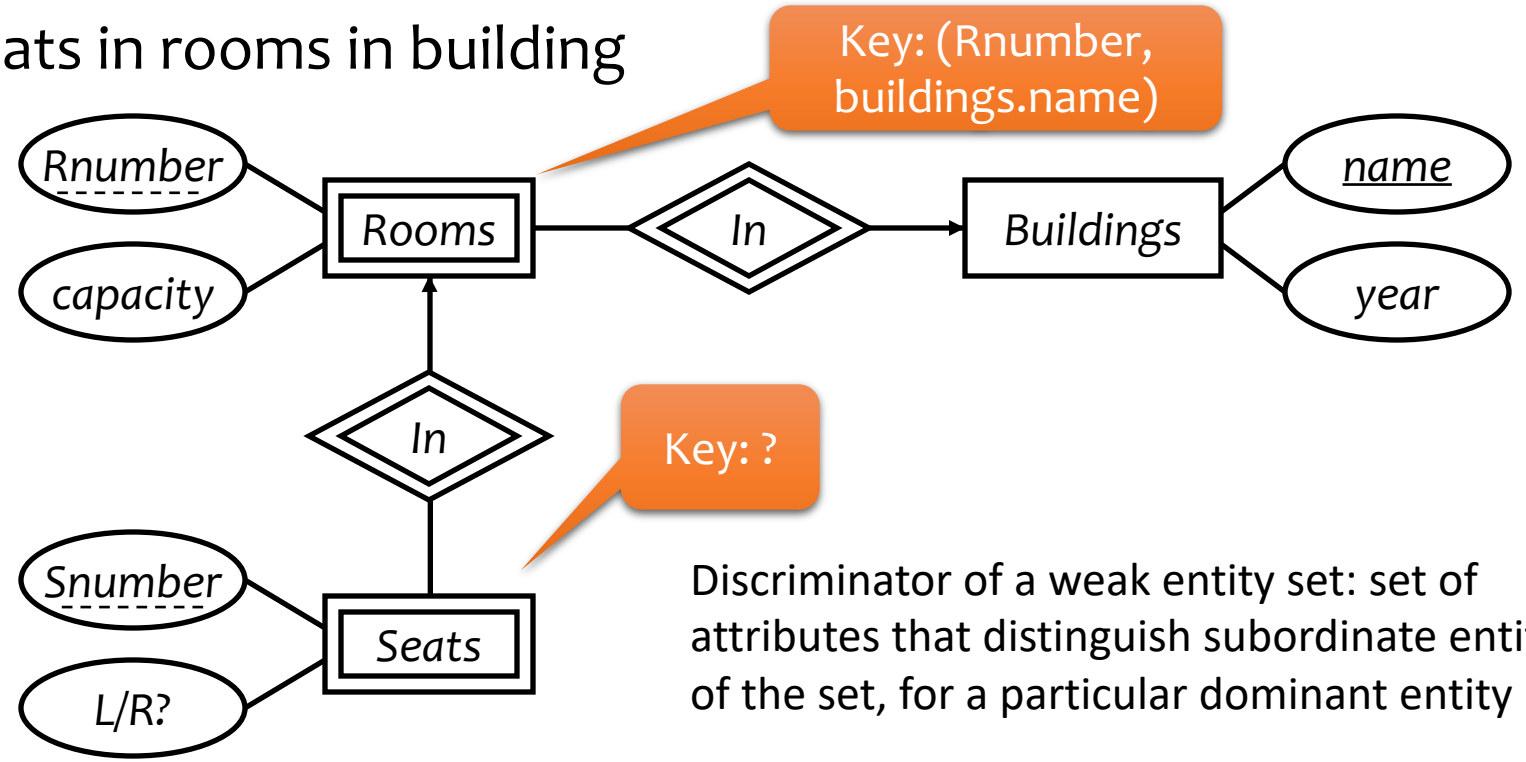
- If entity E's existence depends on entity F, then
  - F is a dominant entity
  - E is a subordinate entity
  - Example: *Rooms* inside *Buildings* are partly identified by *Buildings*' name



- 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

- Seats in rooms in building



Discriminator of a weak entity set: set of attributes that distinguish subordinate entities of the set, for a particular dominant entity

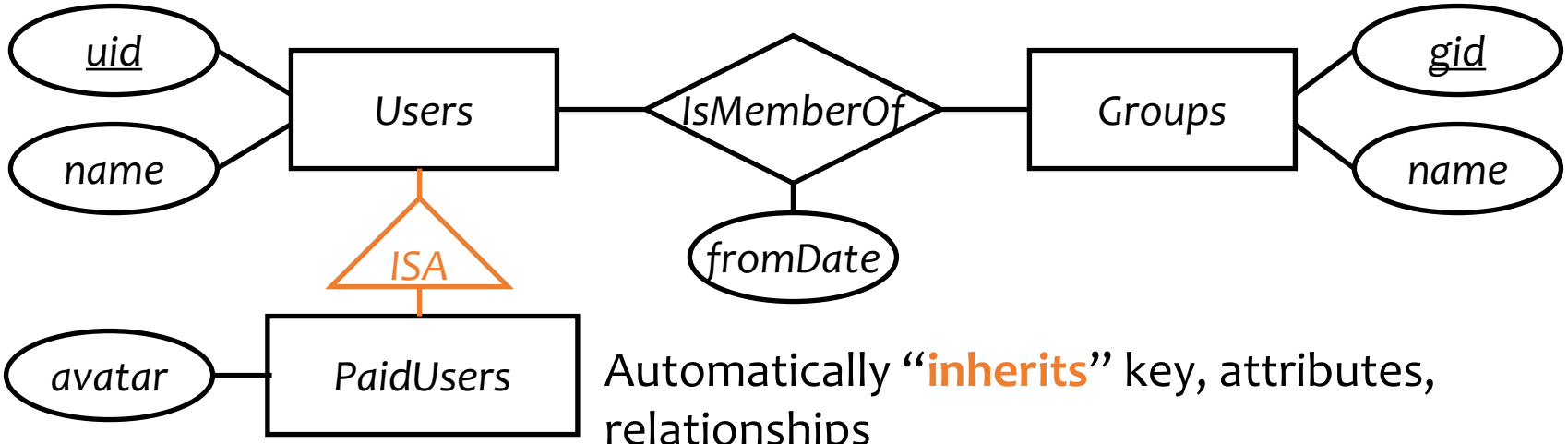
- Attributes of weak entity sets only form key relative to a given dominant entity → discriminator (dotted underline)
- Primary key of a weak entity set: discriminator + primary key of entity set for dominant entities

Extended E-R features

- Generalization vs. Specialization
- Aggregation (different from SQL aggregation!)

Specialization or 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!)

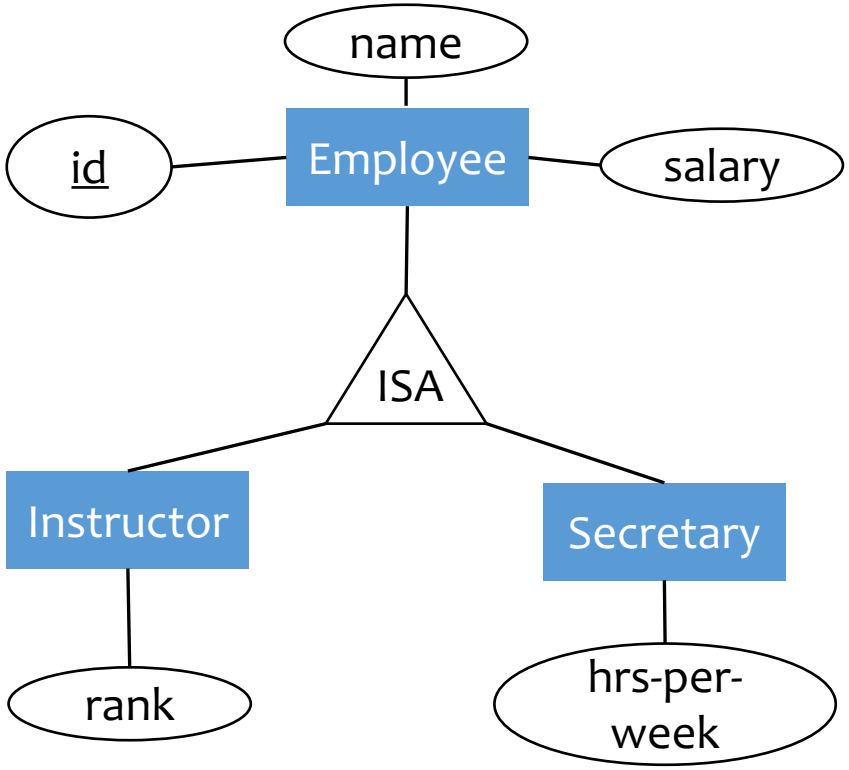


Automatically "**inherits**" key, attributes, relationships
Can participate in other relationships

Generalization

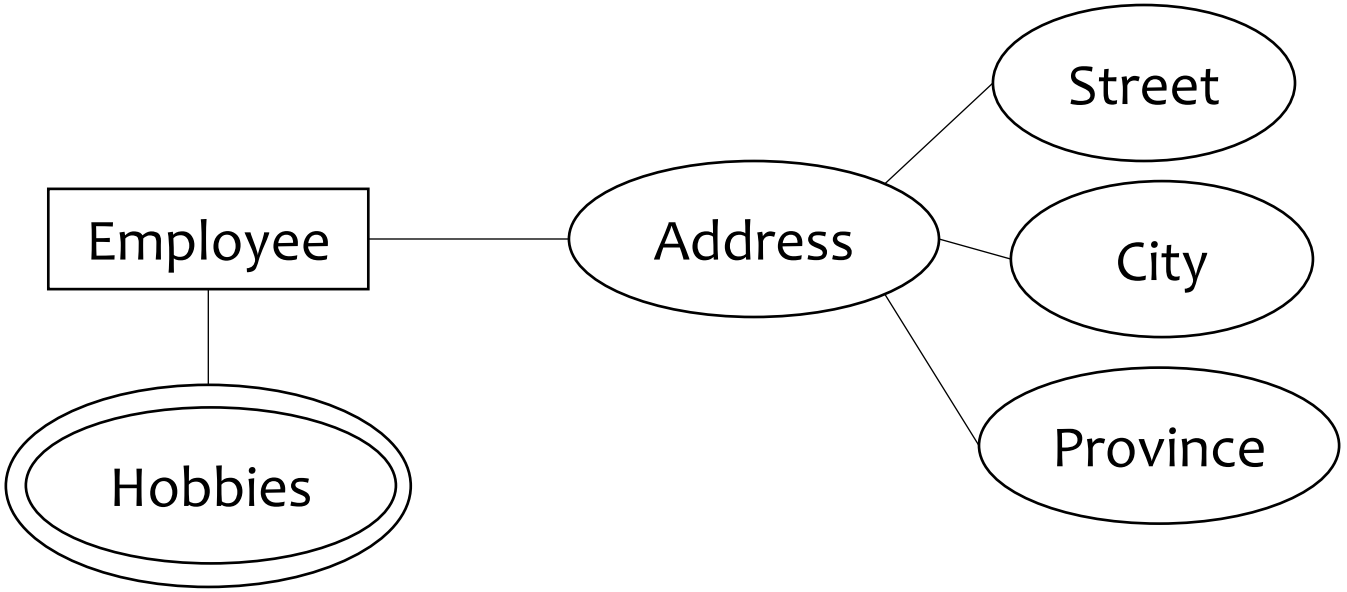
- Several entity sets can be abstracted by a more general entity set
  - Example: "An Employee can represent both an instructor and a secretary"

- Generalization: bottom-up
- Specialization: top-down



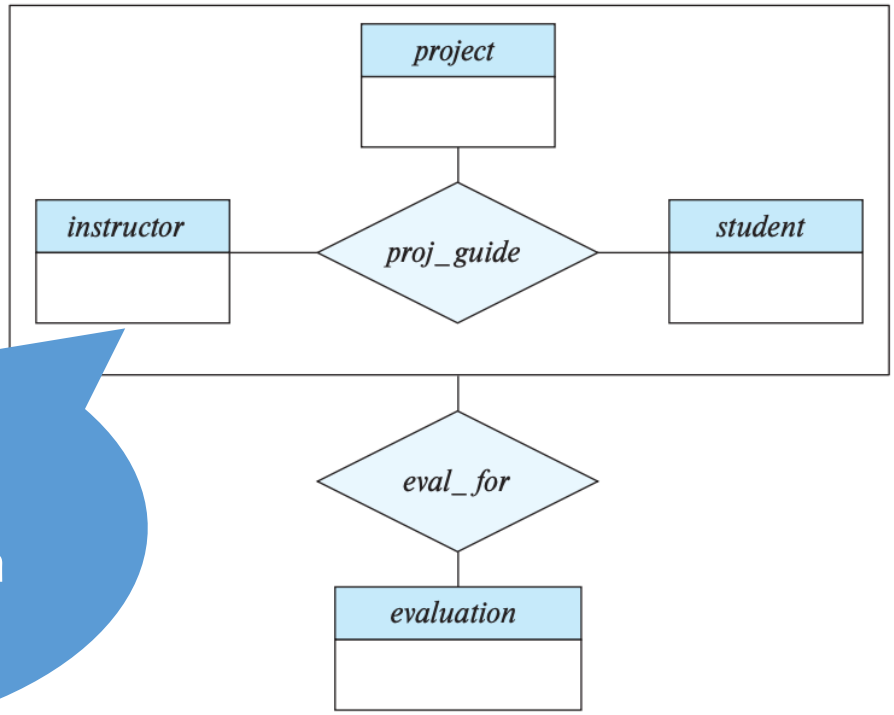
Composite and multi-valued attributes

- Composite attributes: composed of fixed number of other attributes
  - E.g. Address
- Multi-valued attributes: attributes that are set-valued
  - e.g. Hobbies (double edges)



Aggregation

- Aggregation: relationships can be viewed as high-level entities
- Example: "each instructor guiding a student on a project is required to fill a monthly evaluation report "



Example of 3-ary relationship! Relationships with n entities: n-ary

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:
  - Generalization / Specialization
  - Structured attributes
  - Aggregation

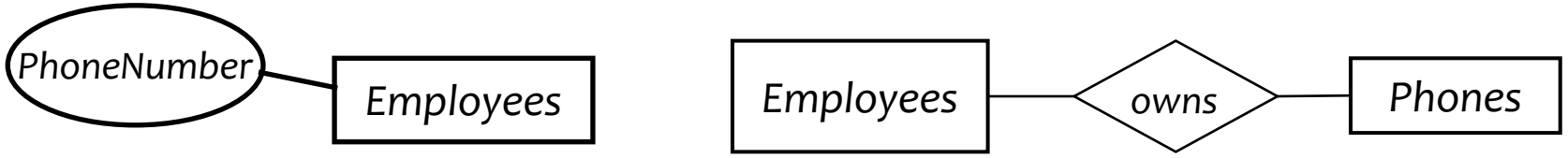
Designing an E/R schema

- Usually many ways to design an E-R schema

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

Attributes or Entity Sets?

- Example: How to model employees' phones?

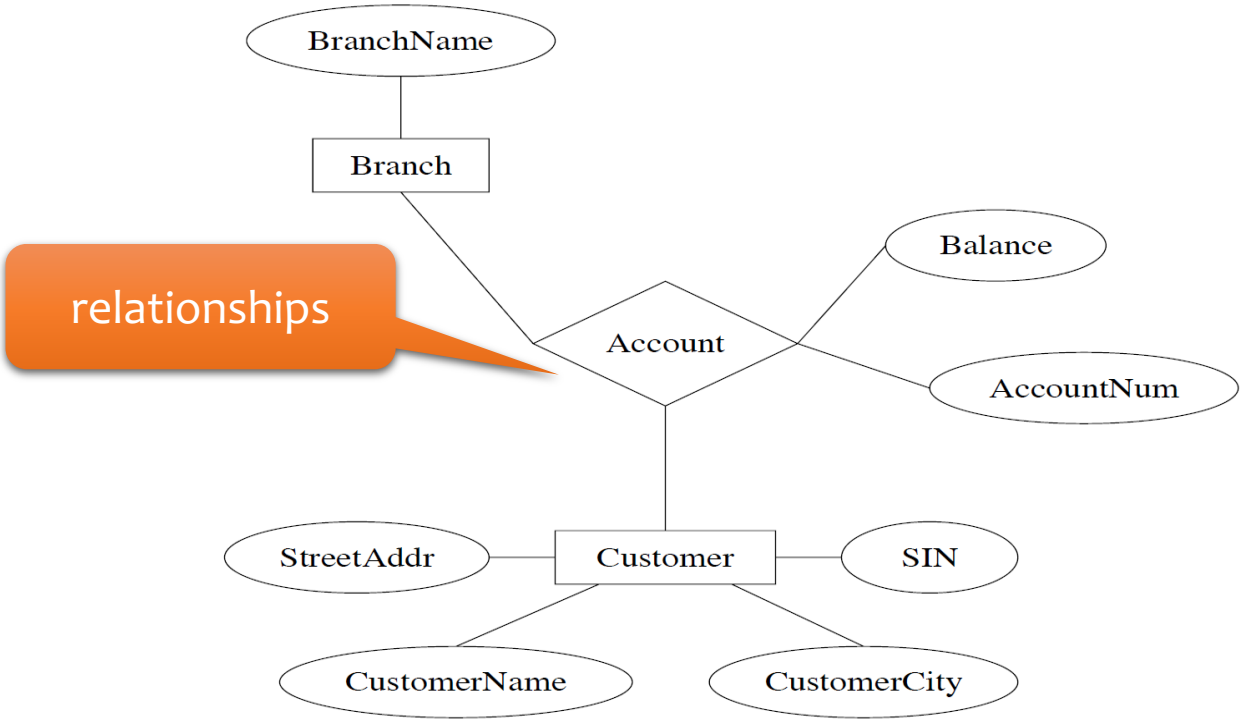


- Rules of thumb:
  - Do we maintain information about it? E.g., model, make
  - Can several of its kind belong to a single entity? E.g., home, office
  - Can it be missing from some of the entity set's entities?
  - Can it be shared by different entities? E.g., 2 employees share office phone

→ An affirmative answer to any of the above suggests a new entity set.

Entity Sets or Relationships?

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

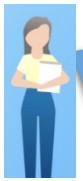


A simple methodology

1. Recognize entity sets
2. Recognize relationship sets and participating entity sets
3. Recognize attributes of entity and relationship sets
4. Define relationship types and existence dependencies
5. Define general cardinality constraints, keys and discriminators
6. Draw diagram

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

Case study 1



Design a database representing cities, counties, and states

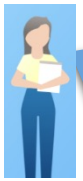
- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

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)

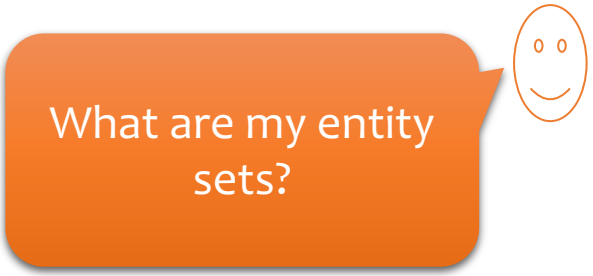
Assume the following:

- 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

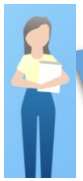
Cities

Counties

States



Case study 1

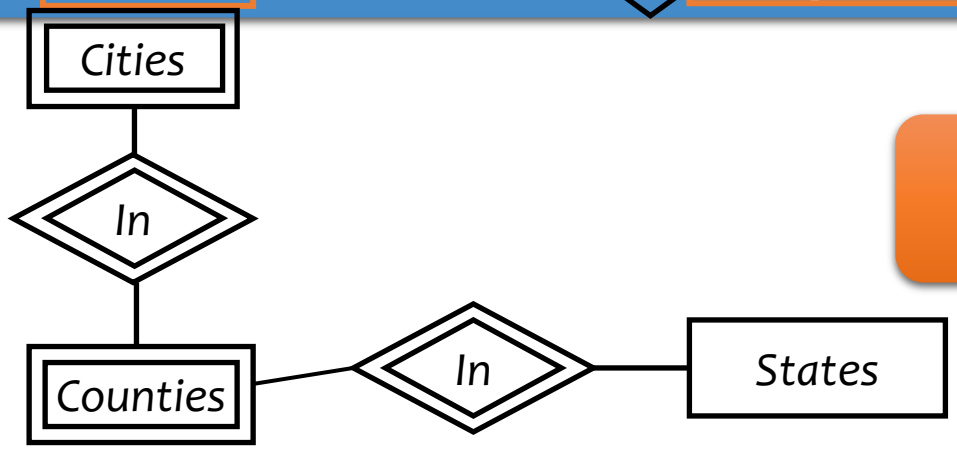


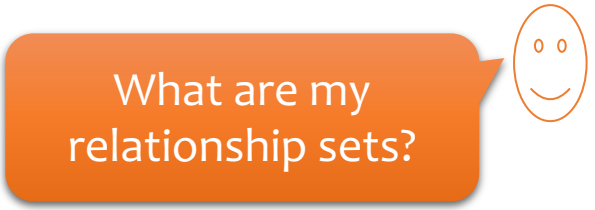
Design a database representing cities, counties, and states

- 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)

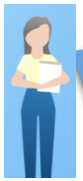
Assume the following:

- 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





Case study 1



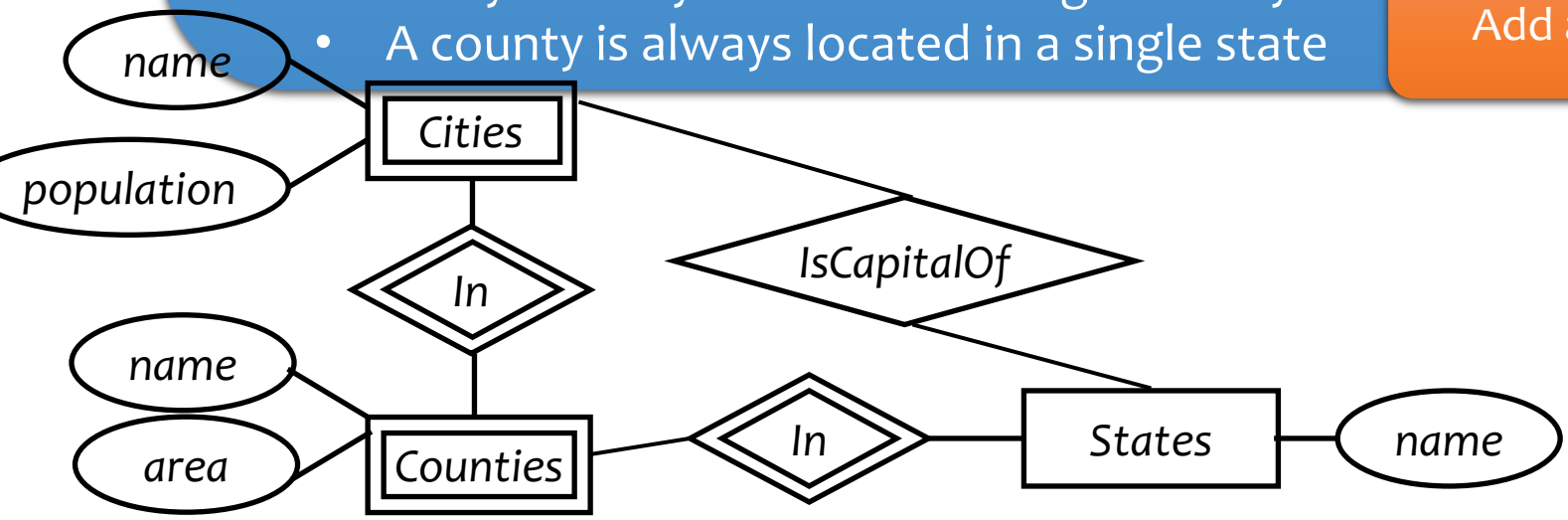
Design a database representing cities, counties, and states

- 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

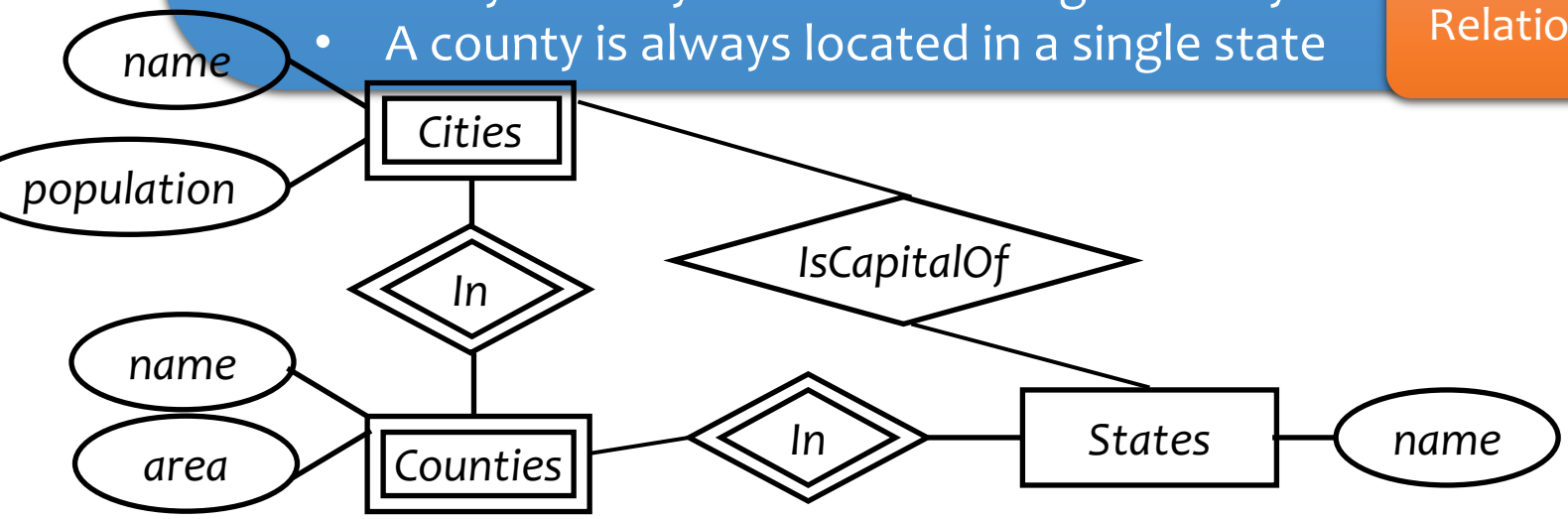
Design a database representing cities, counties, and states

- 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

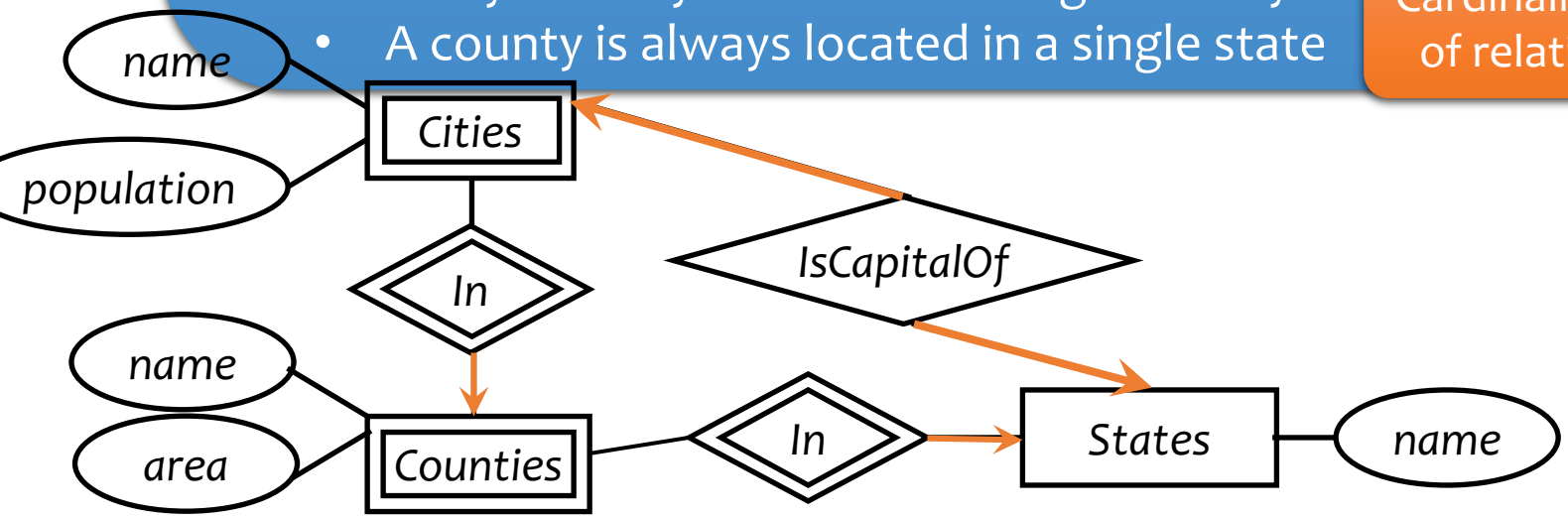
Design a database representing cities, counties, and states

- 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

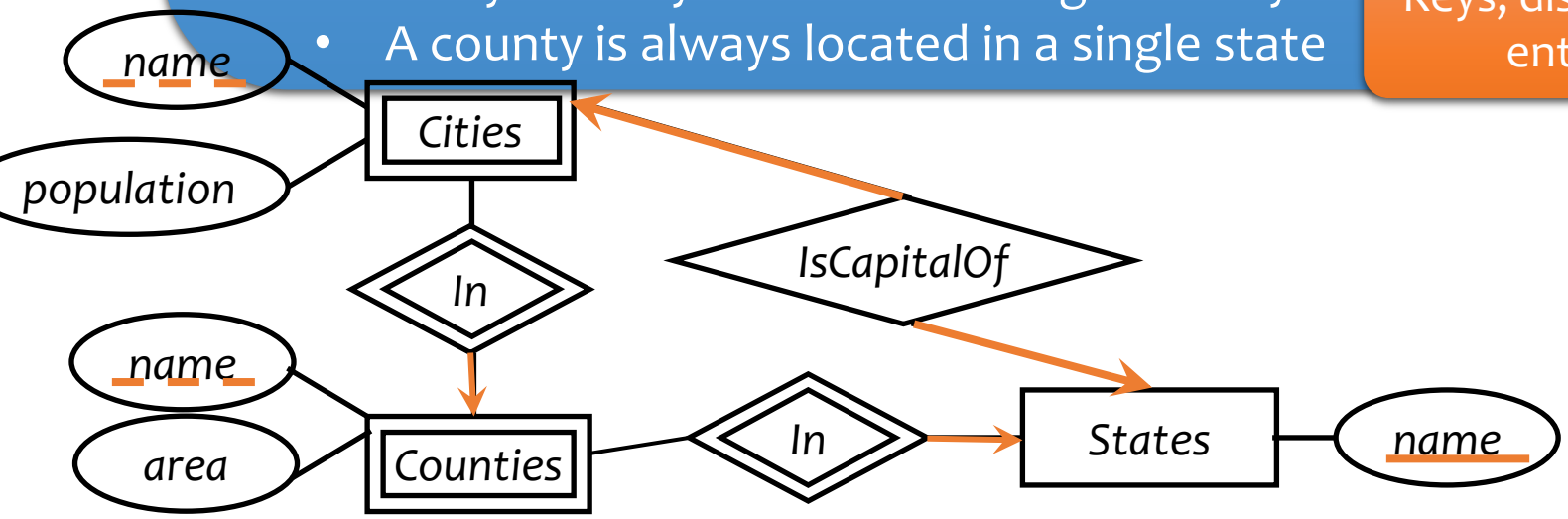
Design a database representing cities, counties, and states

- 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)

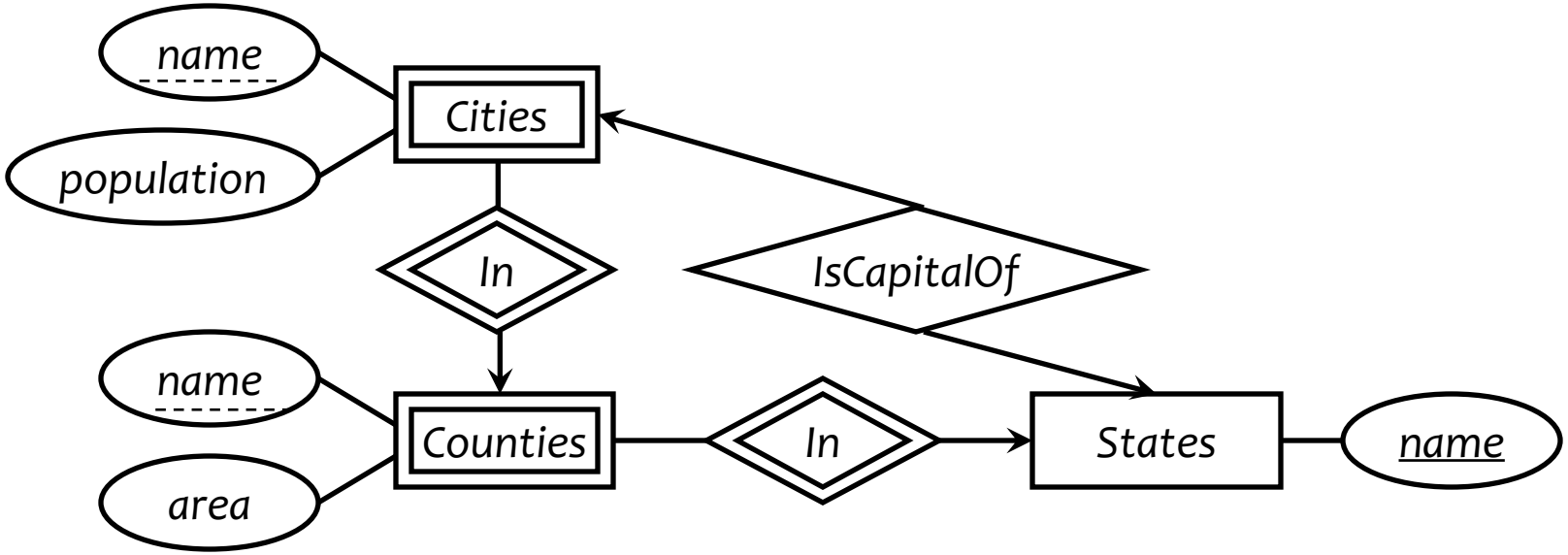
Assume the following:

- 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?

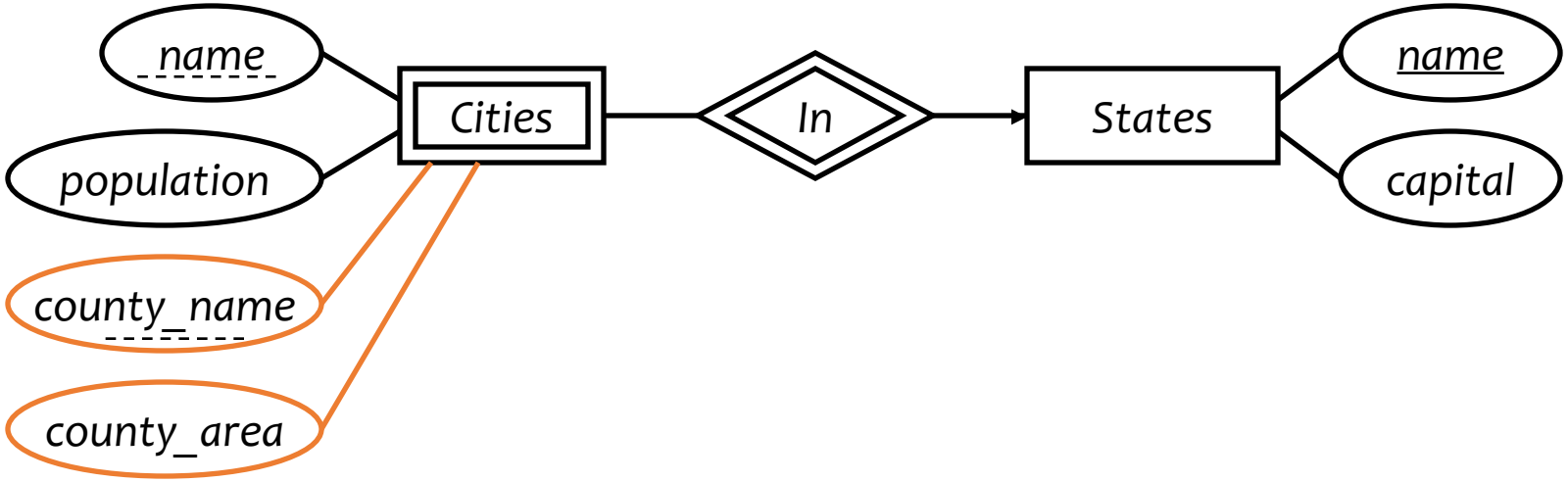


Case study 1: final design



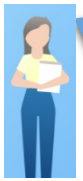
- Technically, nothing in this design prevents a city in state X from being the capital of another state Y, but oh well…

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

Case study 2 (Exercise)



Design a database consistent with the following:

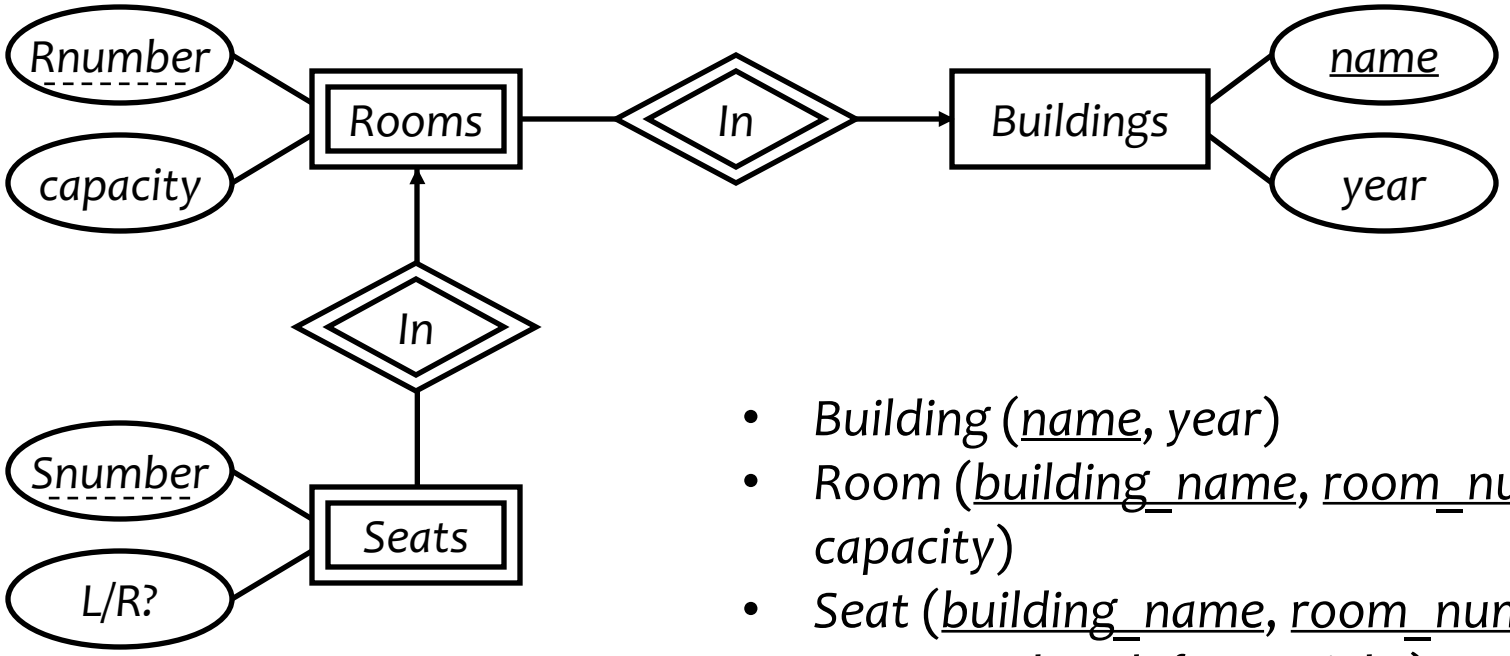
- A station has a unique name and an address, and is either an express station or a local station
- A train has a unique number and an engineer, and is either an express train or a local train
- A local train can stop at any station
- An express train only stops at express stations
- A train can stop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

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

What you have learned so far

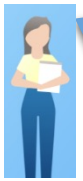
- Entity-Relationship (E/R) model

- 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*)

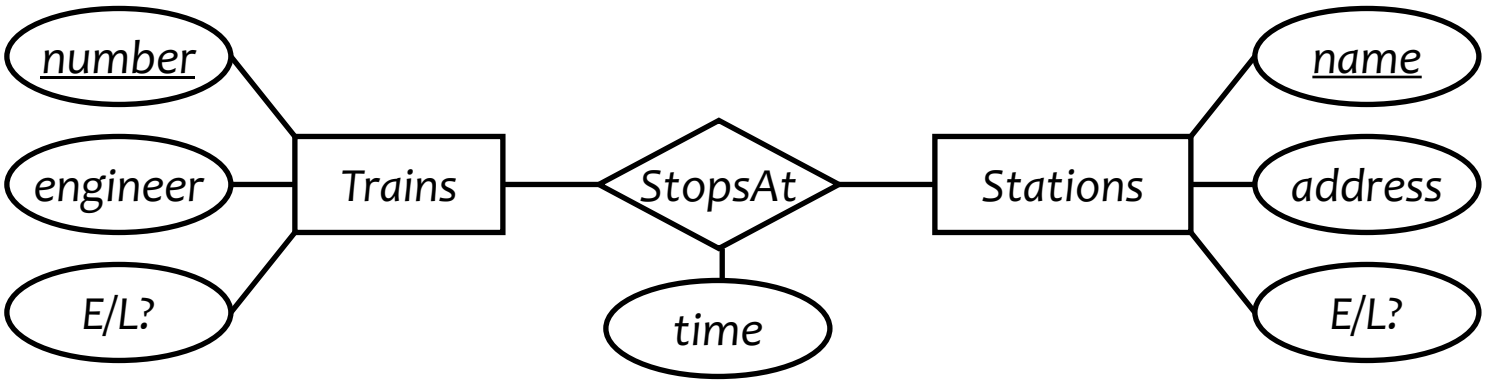
Case study 2: first design



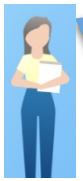
Design a database consistent with the following:

- 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

Why not good?



Case study 2: first design

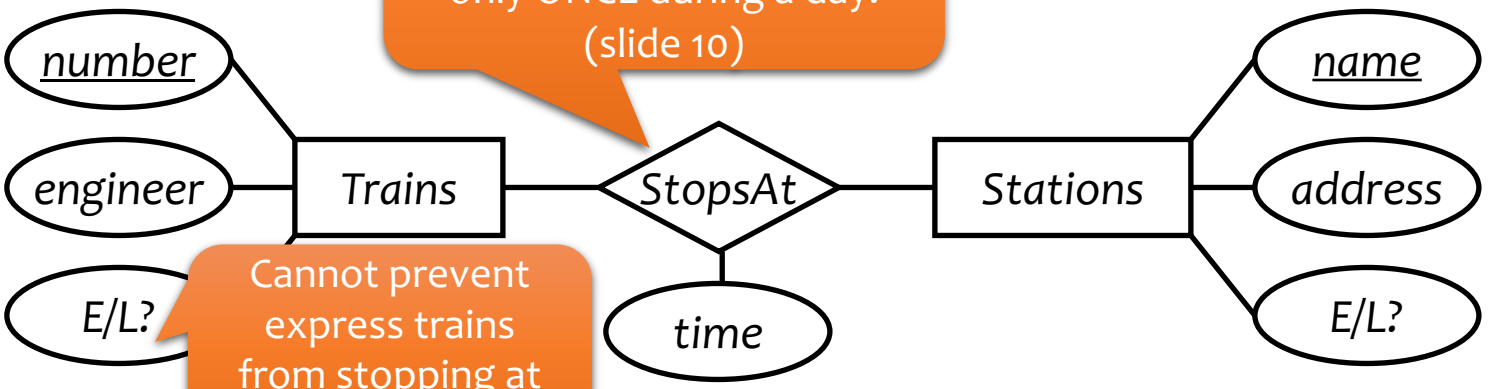


Design a database consistent with the following:

- 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

Why not good?

Unintended constraint: A train can stop at a station only ONCE during a day! (slide 10)

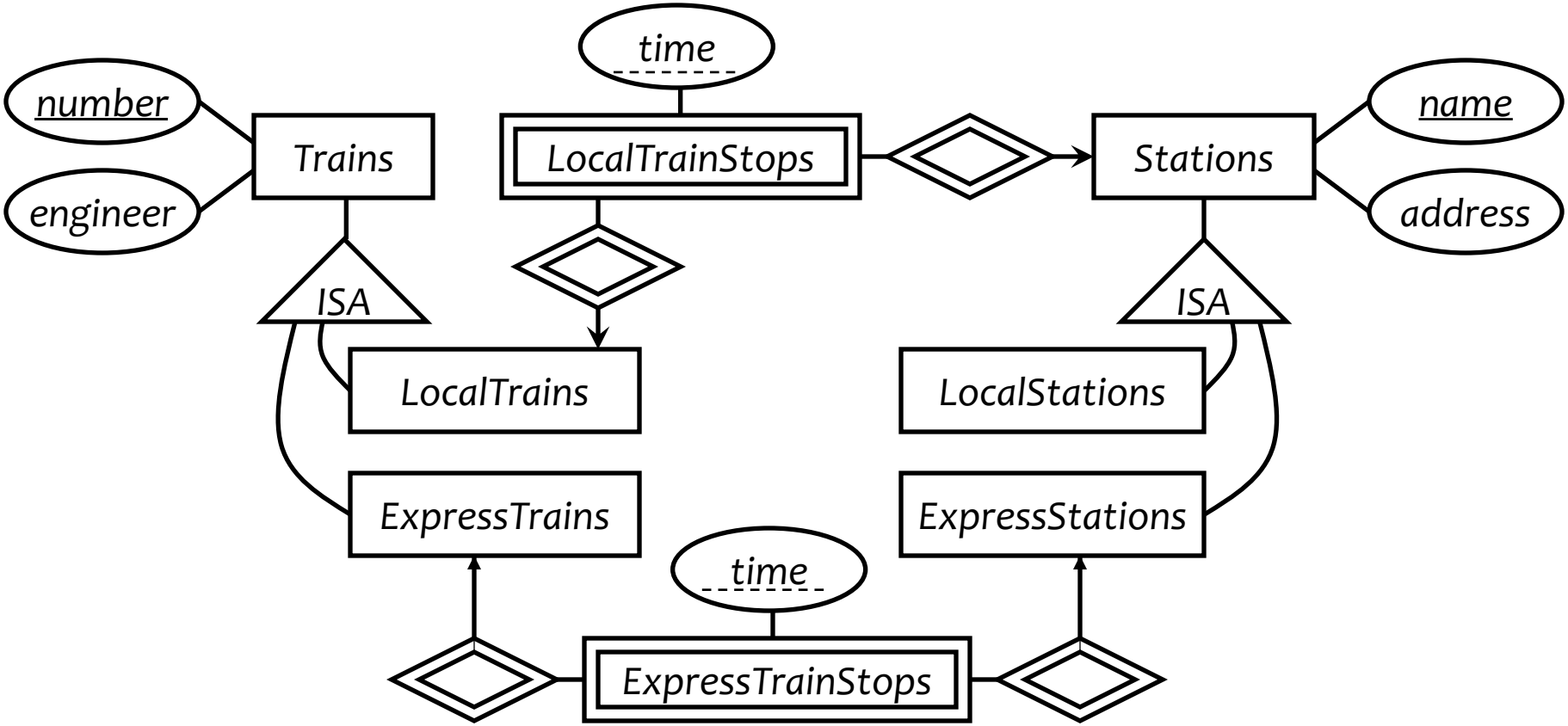


Cannot prevent express trains from stopping at any local stations

Case study 2: second design

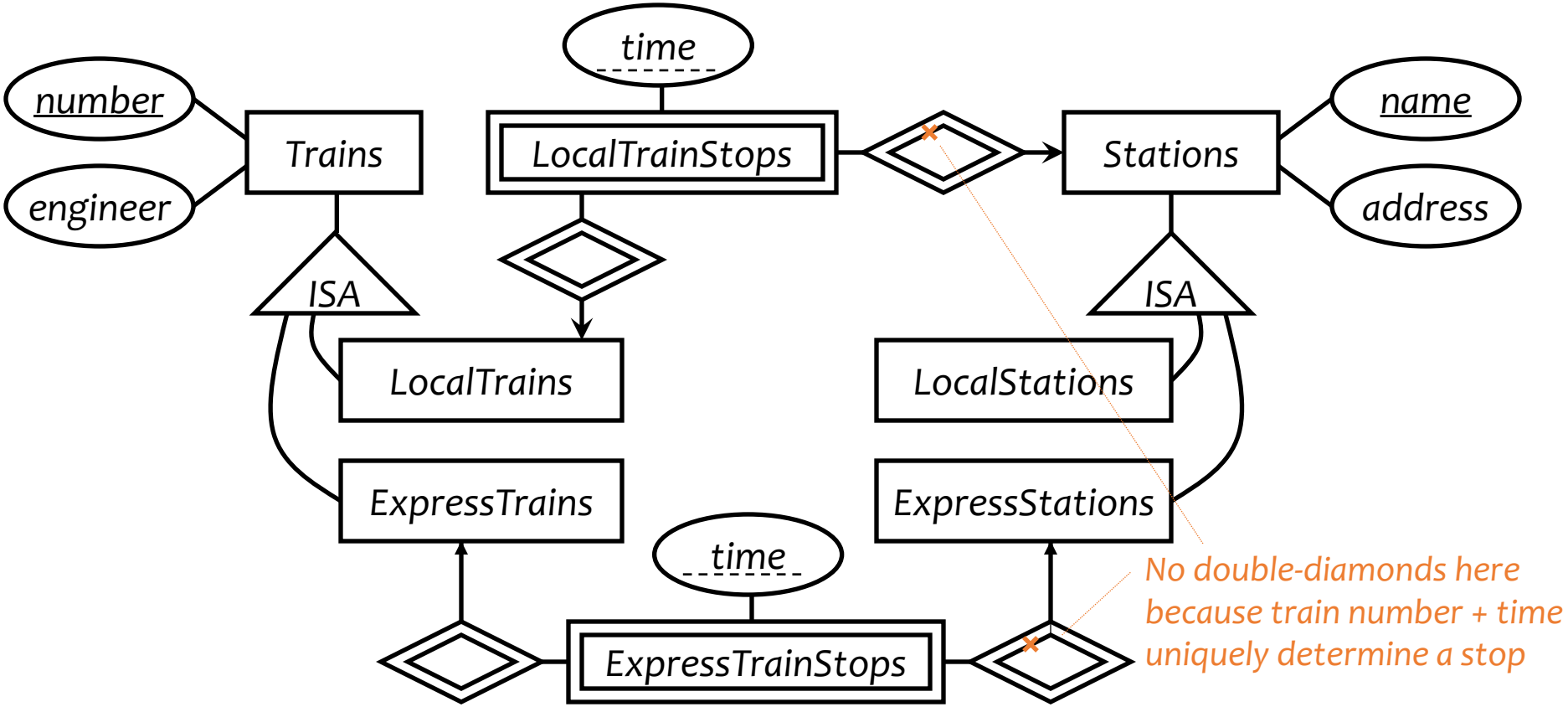
- A station has a unique name and an address, and is either an express station or a local station
- A train has a unique number and an engineer, and is either an express train or a local train
- …..

Case study 2: second design



- …
- 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
- …

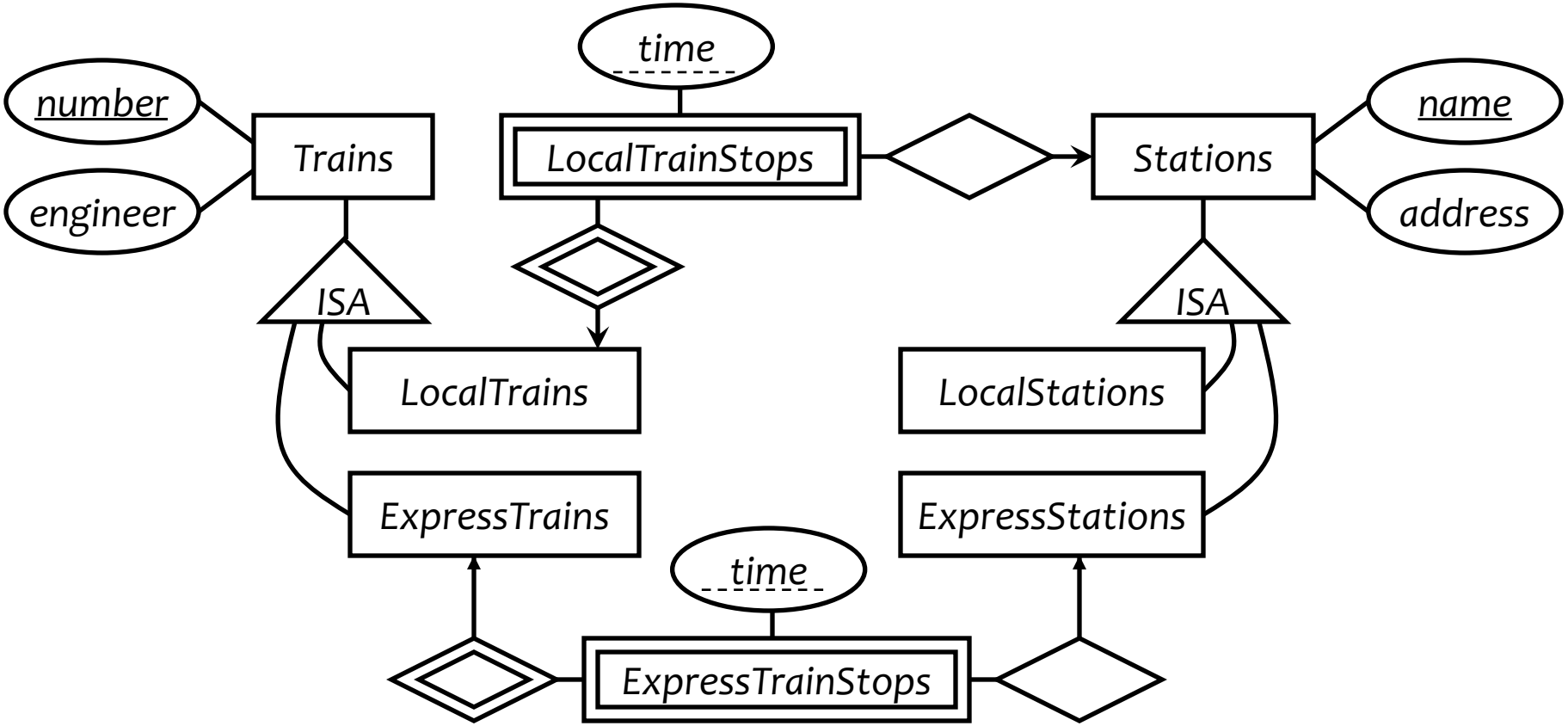
Case study 2: second design



Is the extra complexity worth it?

Yes! Captures more constraints and avoids unintended info

Case study 2: final solution looks like..



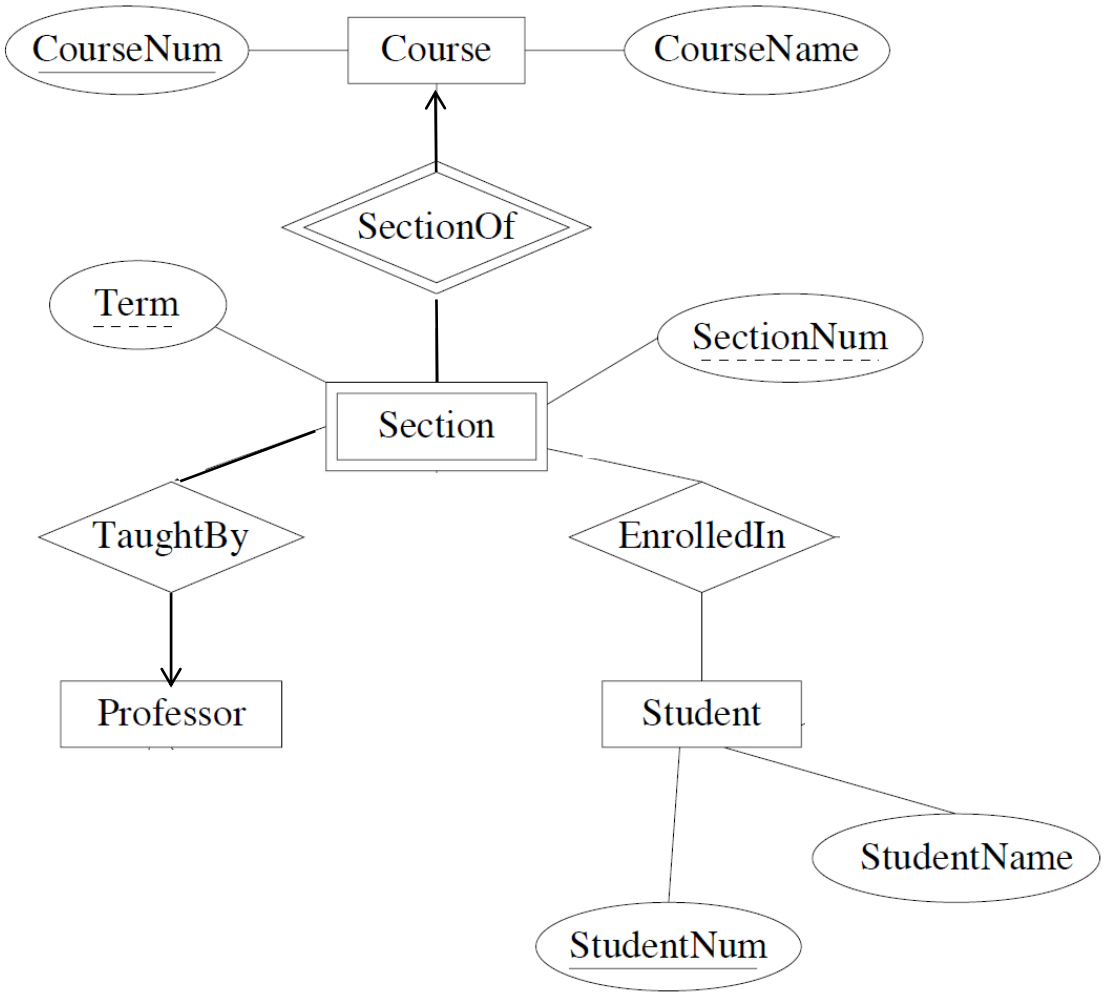
Case study 3 (Exercise)

- 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. 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.

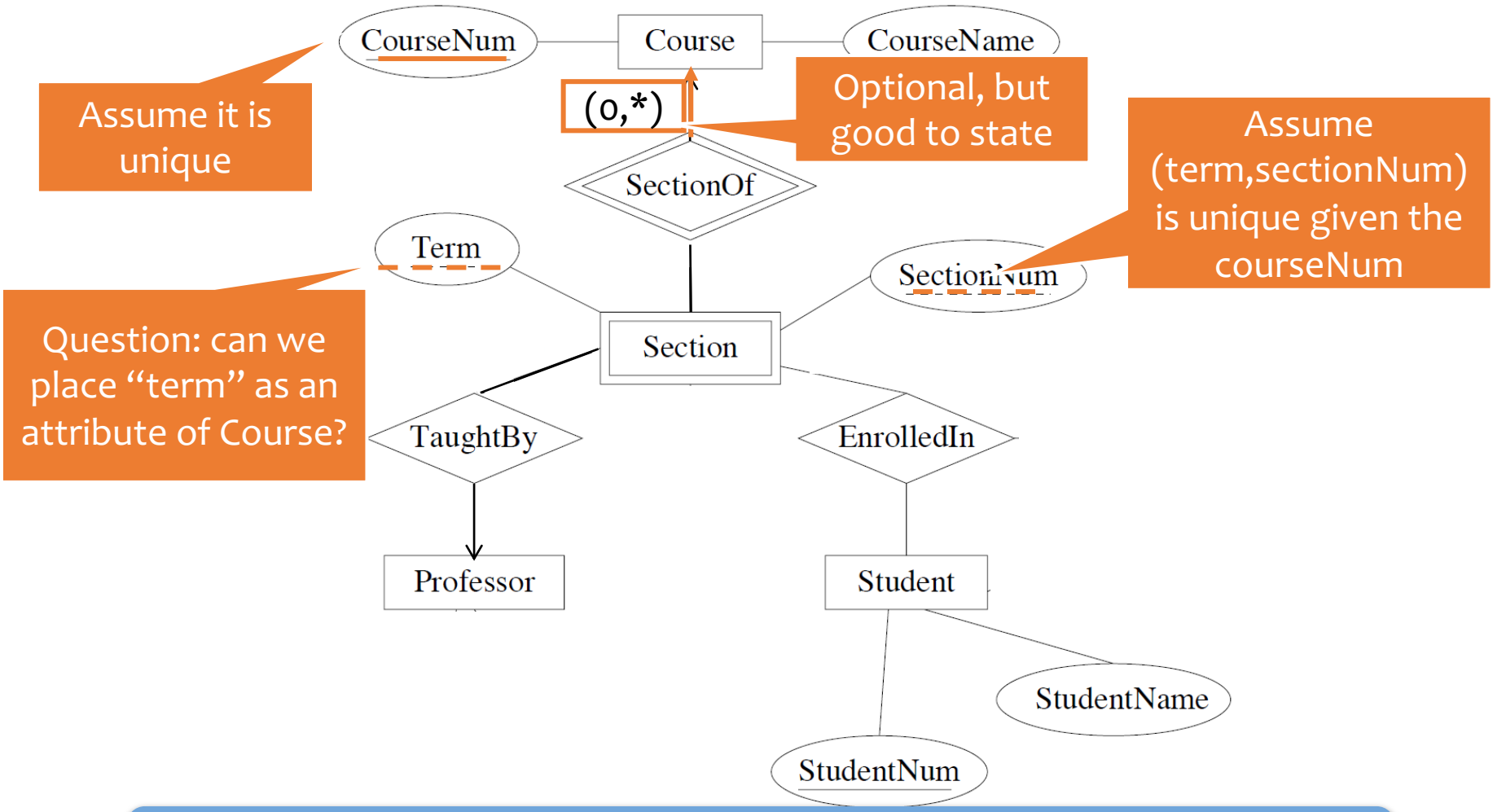
Case study 3 (Exercise)

- 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. 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.

Case study 3 (Exercise) cont.

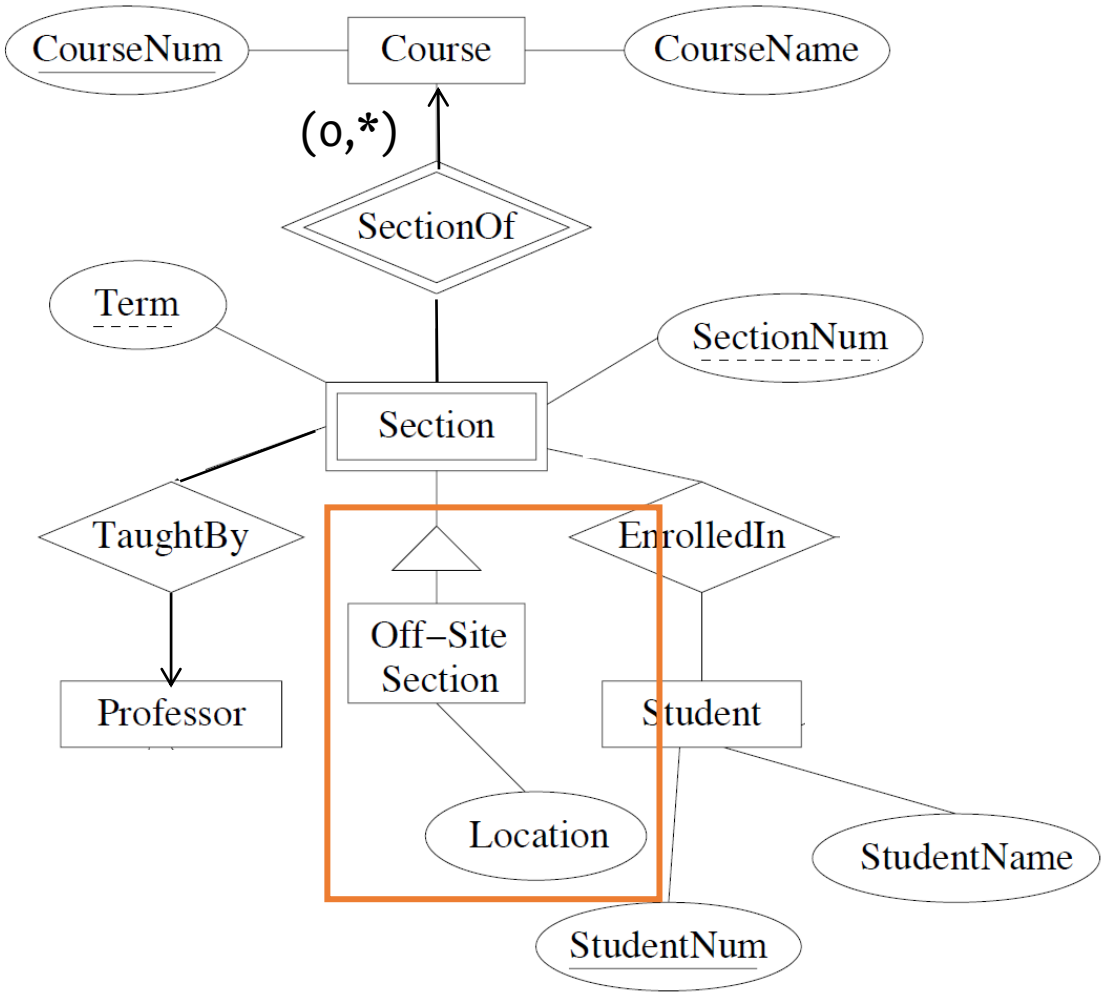


Case study 3 (Exercise) cont.



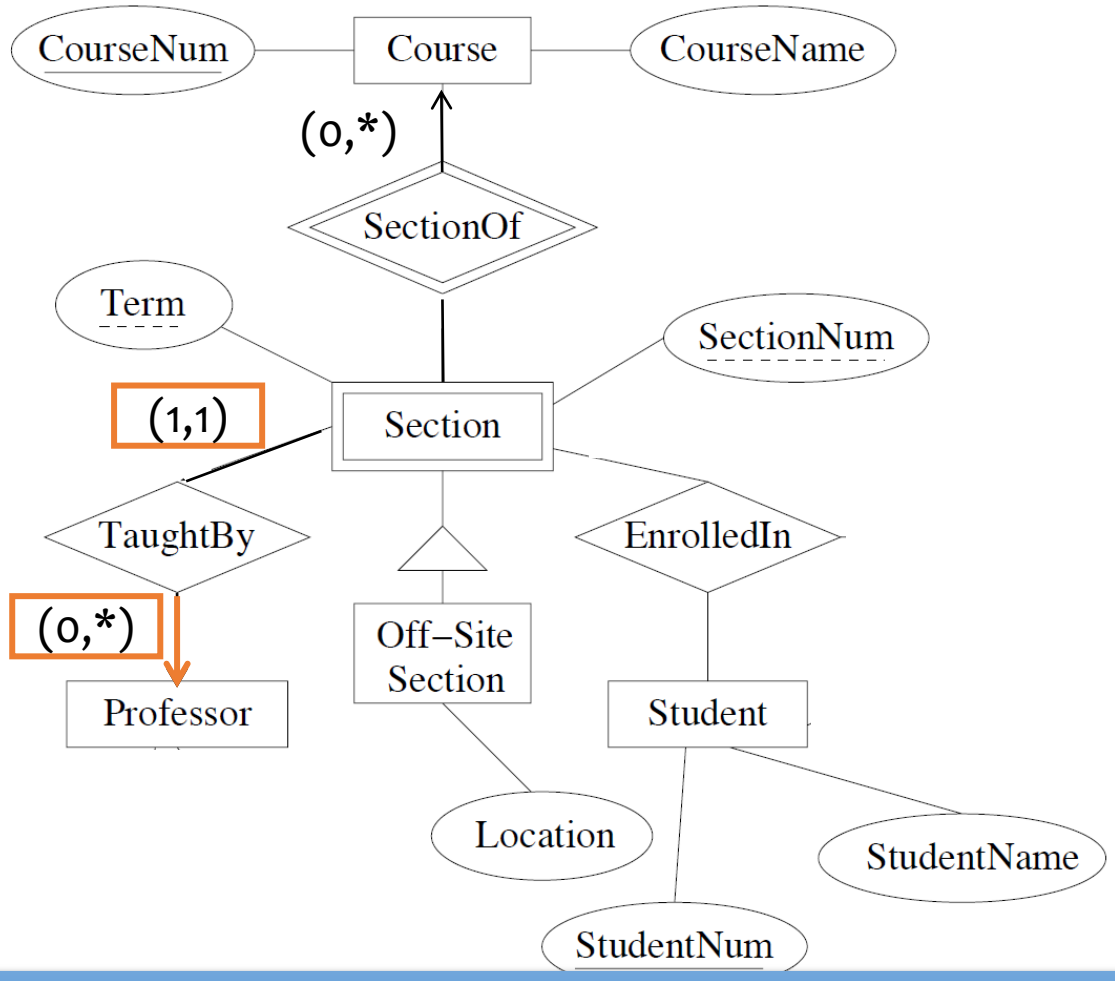
Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.

Case study 3 (Exercise) cont.



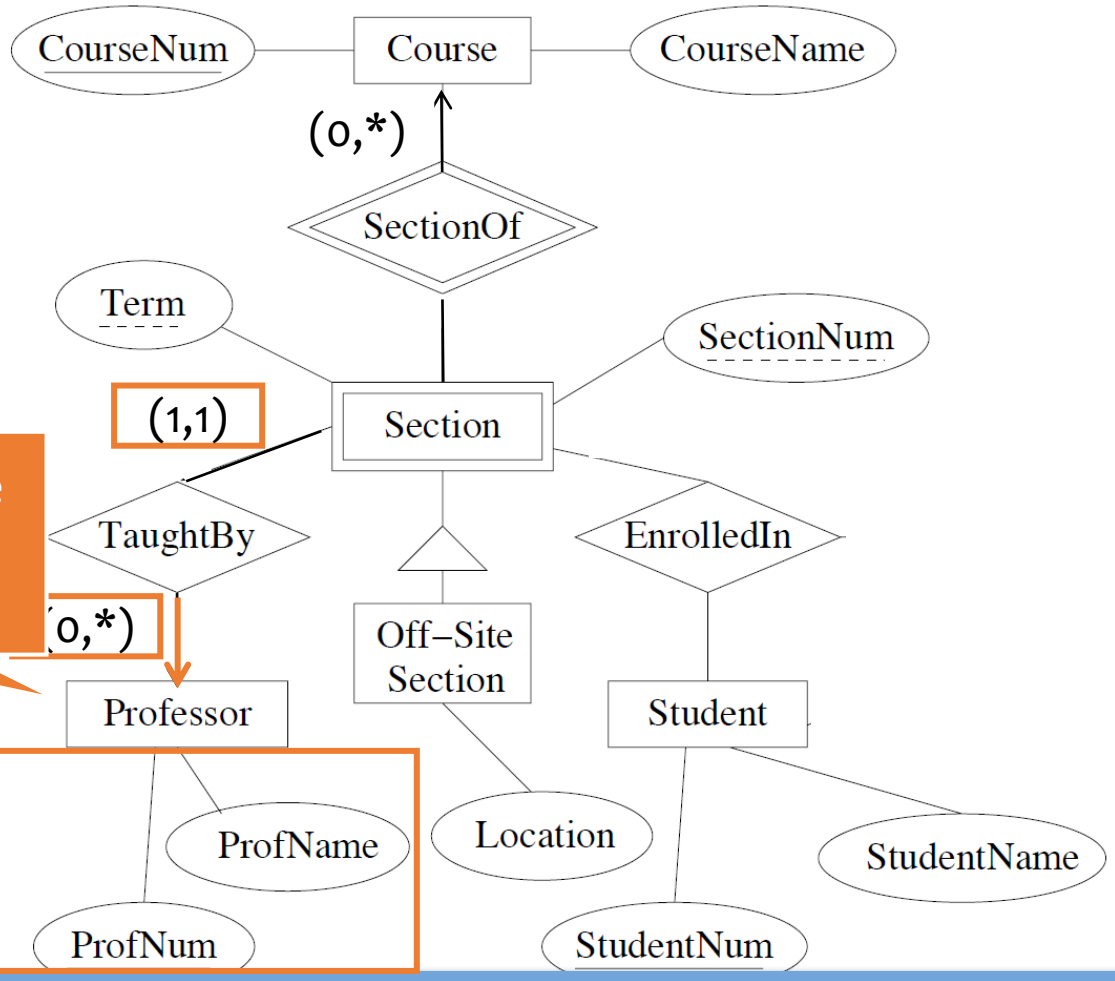
Most course sections are taught on-site, but a few are taught at off-site locations.

Case study 3 (Exercise) cont.



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.

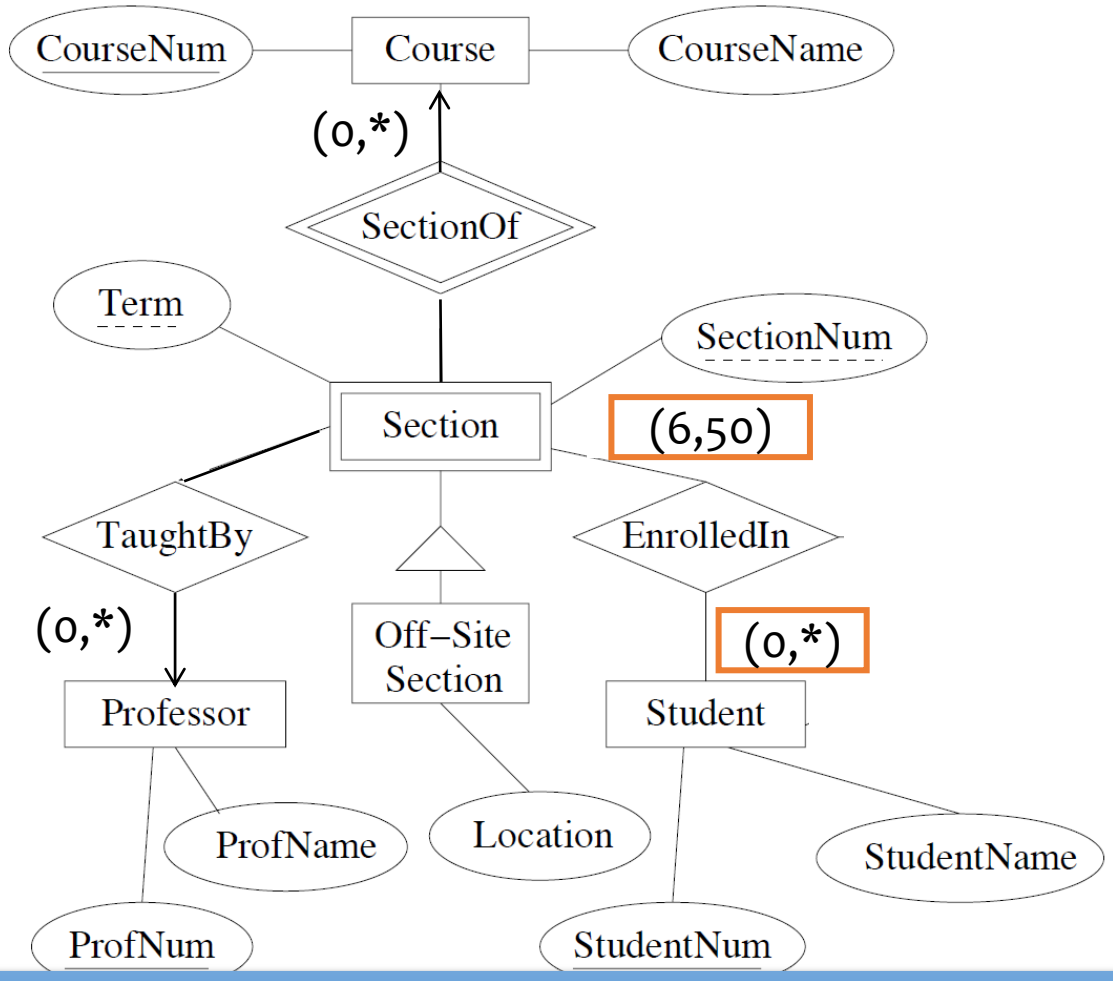
Case study 3 (Exercise) cont.



Can "Professor" be an attribute of Section?

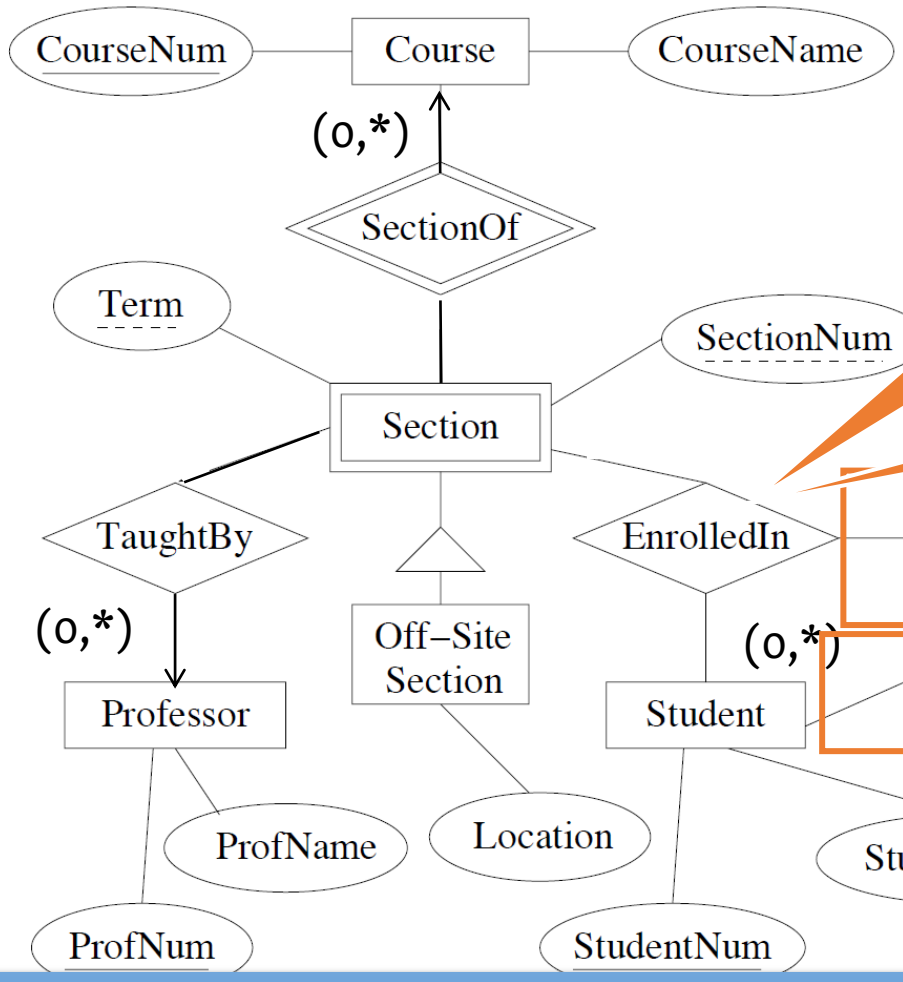
Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.

Case study 3 (Exercise) cont.



Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled. Students can enroll in 0 or more sections (implicitly derived).

Case study 3 (Exercise) cont.



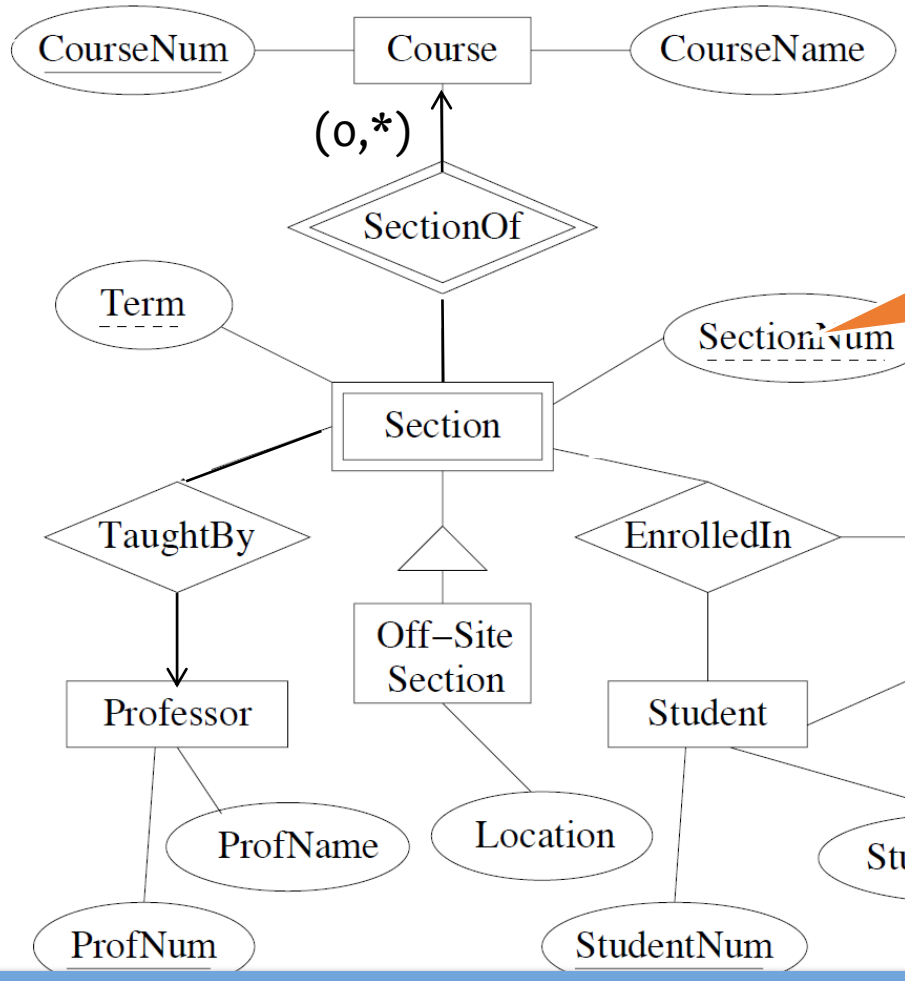
Can "EnrolledIn" be a relationship set between Course and Student instead?

No, we may not be able to specify "(6,50)"

We did not capture this requirement

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.

Case study 3: possible solution



Assume it is unique

Assume (term,sectionNum) is unique given the courseNum

We did not capture this requirement

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.