CPSC 304 – September 13/17, 2024 Administrative Notes

- Reminder: tutorials start the week of the 16th.
  - There will be nothing due from tutorial
  - Students who are registered in a tutorial have priority
  - You may ask tutorial TAs (in any section) questions as in office hours (subject to available time)
  - We will put a list of tutorial time/locations up on Canvas
- Reminder: syllabus quiz due 20nd @10pm
- Reminder: “In-class” exercise 1 due 18th @10pm
- Reminder: Project groups due September 20
  - Please look on Canvas (milestone 0 description)
  - Submit the survey to have a group created (see Canvas for link)

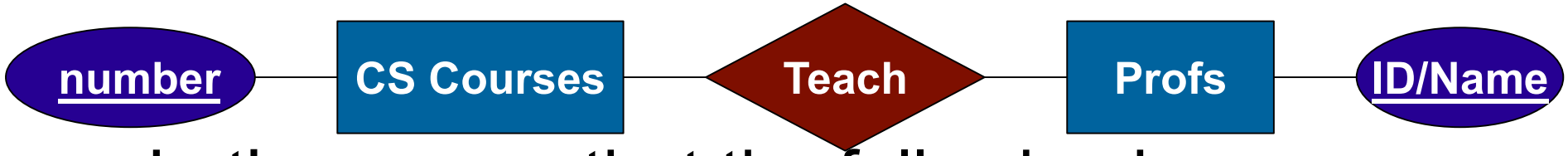
Now where were we…

- We'd been talking about ER diagrams
- We'd learned:
  - Entities
  - Attributes
  - Relationships
  - IsA (Inheritance)
  - Weak entities

Before we get to our last concept (aggregation), I want to make things clearer w.r.t. line types. Let's look at some examples…

Line annotation example

- Let's focus on a simplified version of "Profs teach courses":

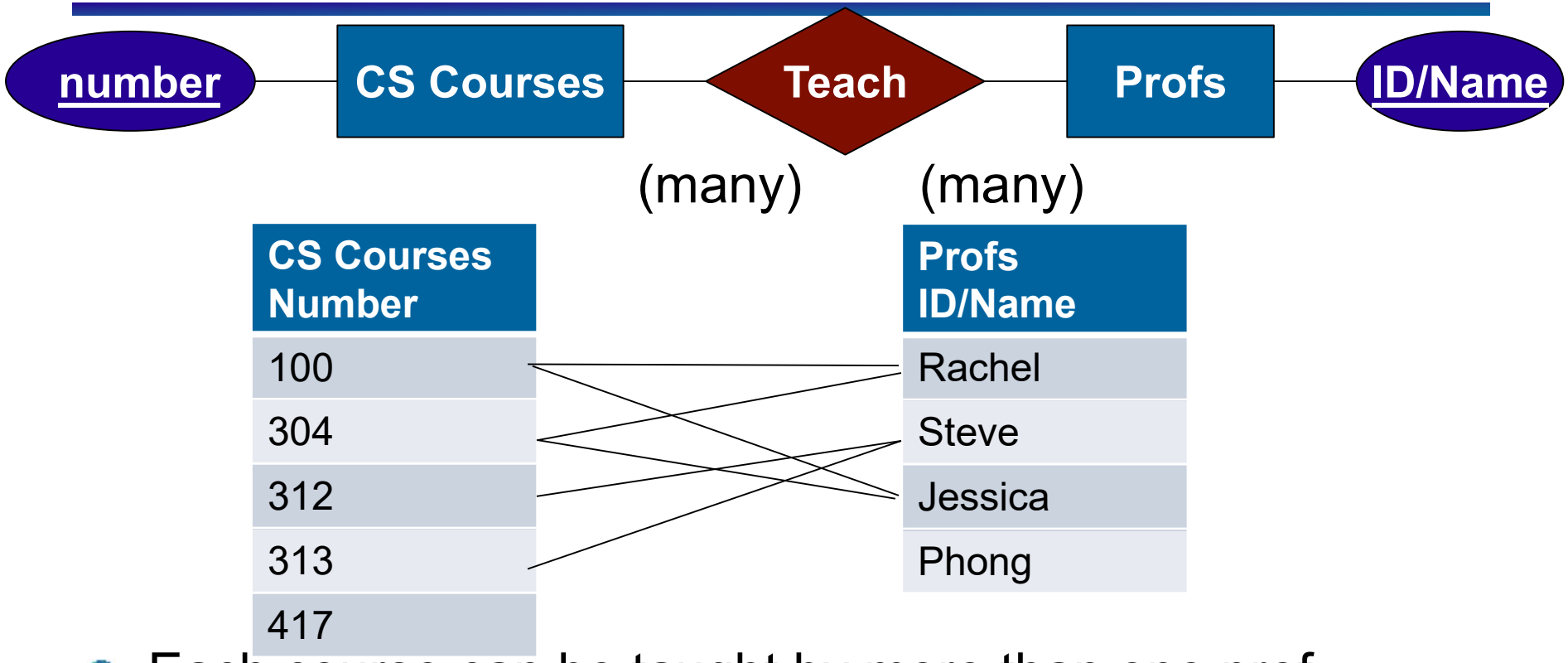


- Let's assume that the following is our entire set of entities:

CS Courses Number
100
304
312
313
417

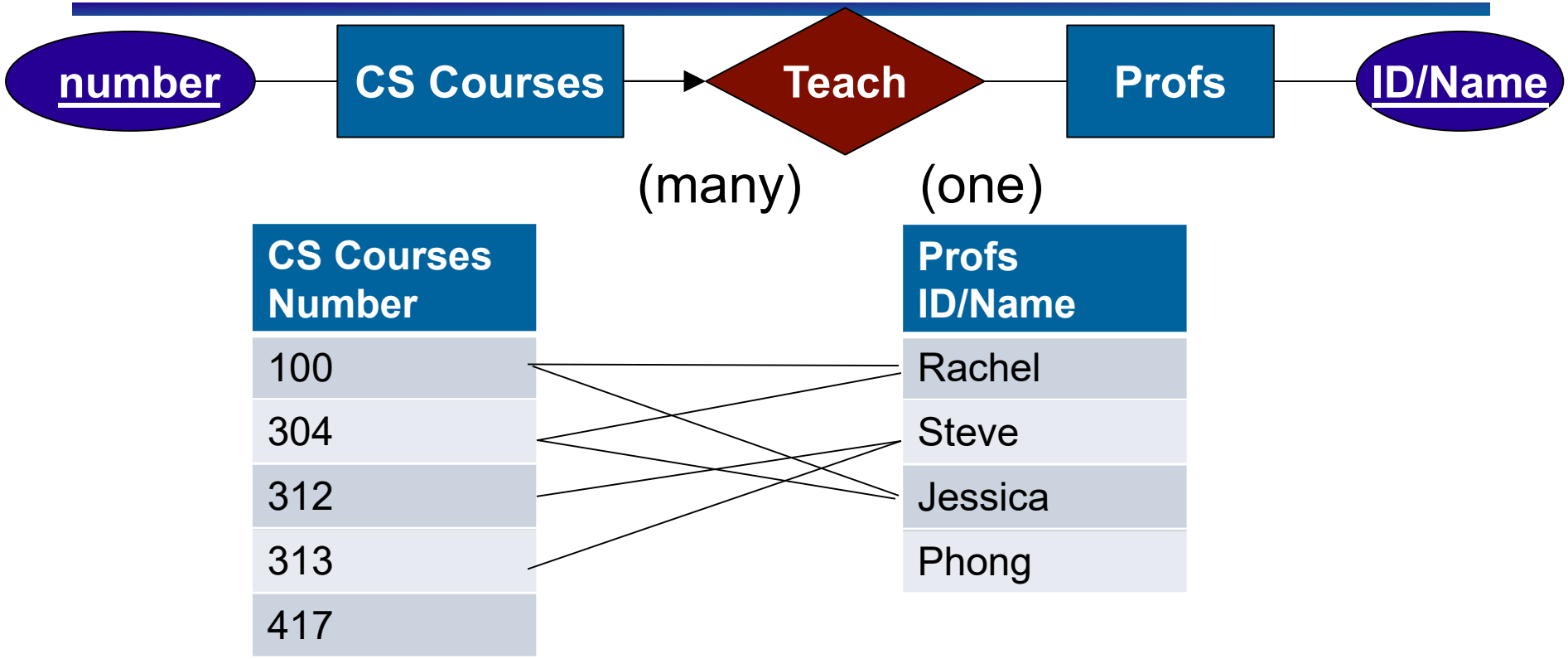
Profs ID/Name
Rachel
Steve
Jessica
Phong

Plain lines only



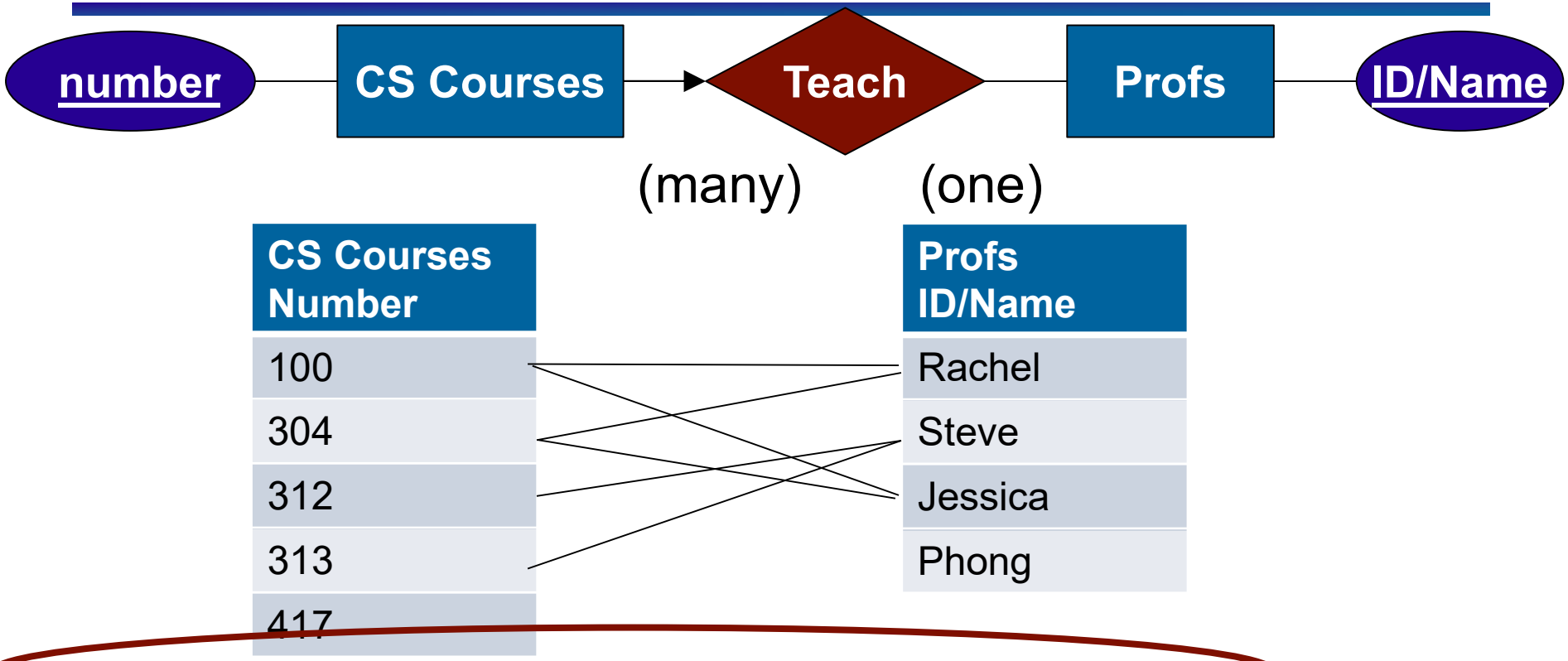
- Each course can be taught by more than one prof
- Each prof can teach more than one course
- Not all courses have to be taught by some prof
- Not all profs have to teach a course

Clicker question: Many-to-one. Which of the following would allow us to satisfy the given constraint?



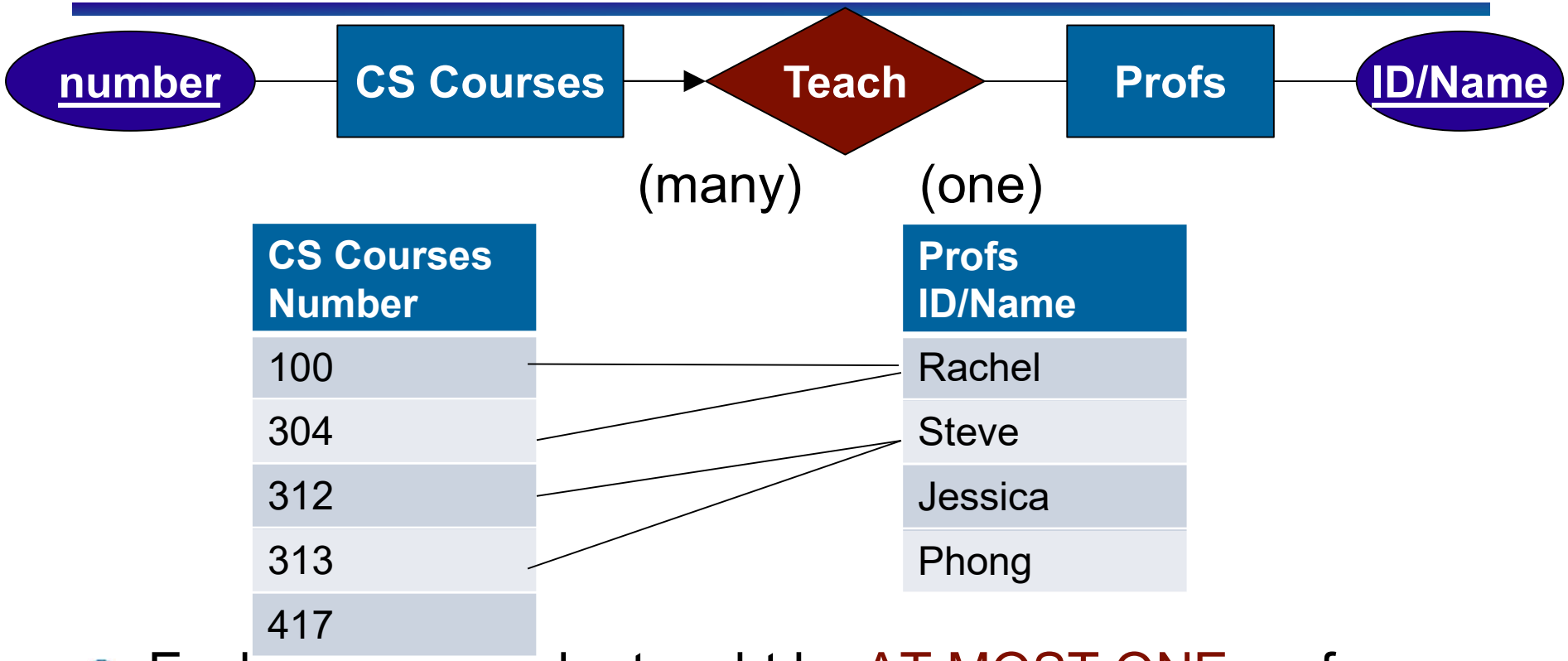
A. Removing {(100,Rachel), (304, Jessica)} from Teach
B. Removing {(304,Rachel), (304, Jessica), (312, Steve)} from Teach
C. Adding {(417, Phong)} to Teach
D. All of the above
E. None of the above

Clicker question: Many-to-one. Which of the following would allow us to satisfy the given constraint?



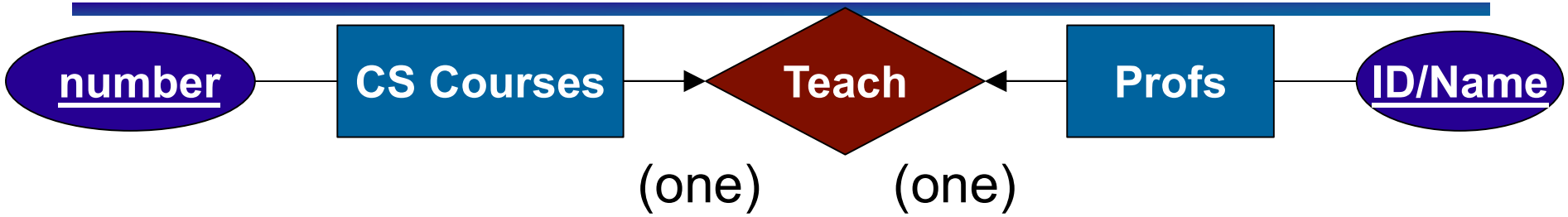
A. Removing {(100,Rachel), (304, Jessica)} from Teach
B. Removing {(304,Rachel), (304, Jessica), (312, Steve)} from Teach
C. Adding {(417, Phong)} to Teach
D. All of the above
E. None of the above

Many-to-one



- Each course can be taught by AT MOST ONE prof
- Each prof can teach more than one course
- Not all courses have to be taught by some prof
- Not all profs have to teach a course

One-to-one

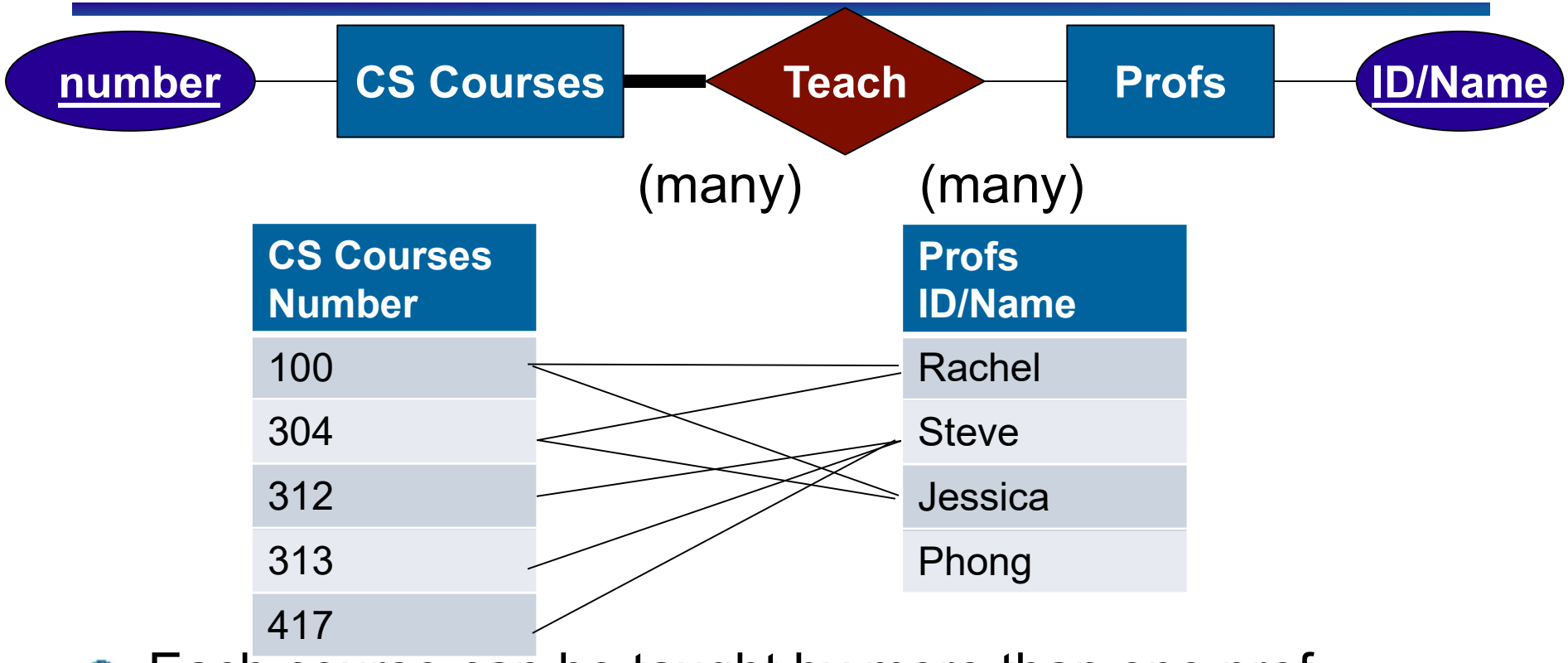


CS Courses Number
100
304
312
313
417

Profs ID/Name
Rachel
Steve
Jessica
Phong

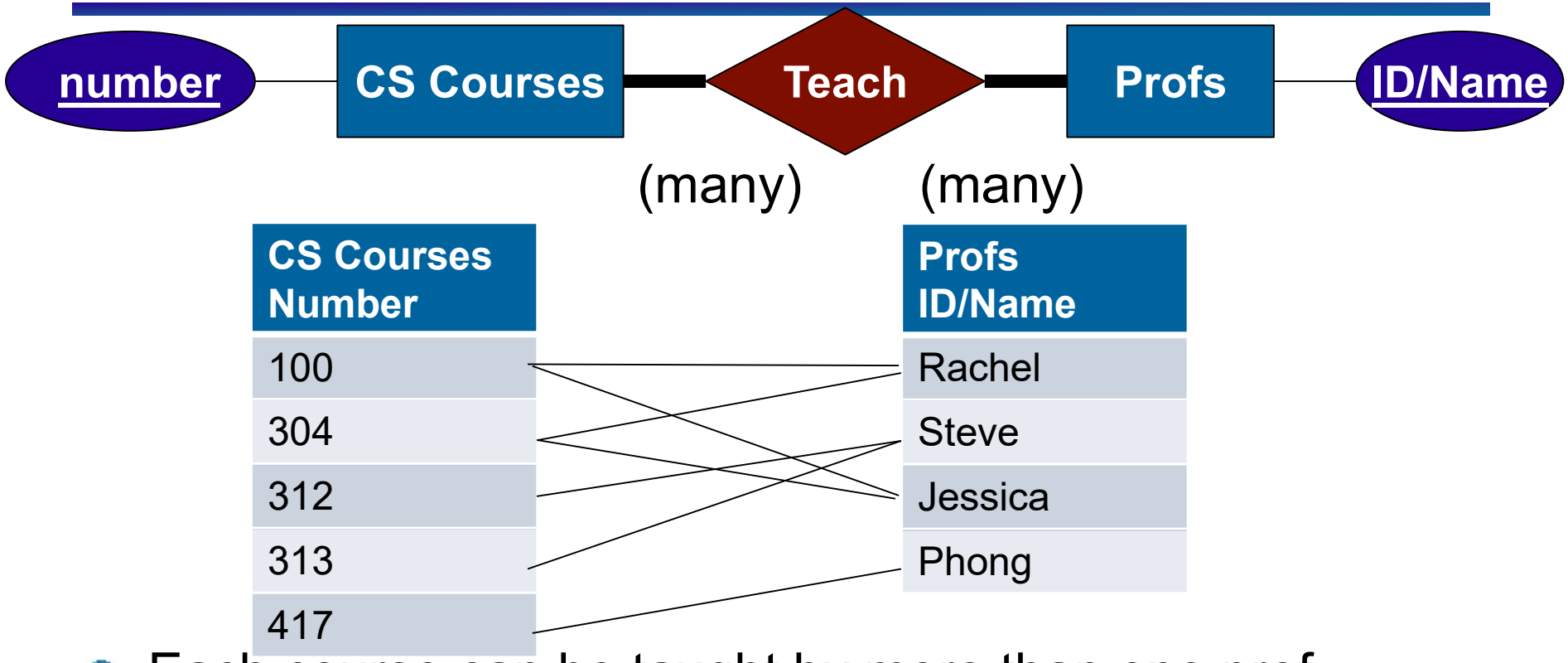
- Each course can be taught by AT MOST ONE prof
- Each prof can teach AT MOST ONE one course
- Not all courses have to be taught by some prof
- Not all profs have to teach a course

Thick lines – on one side



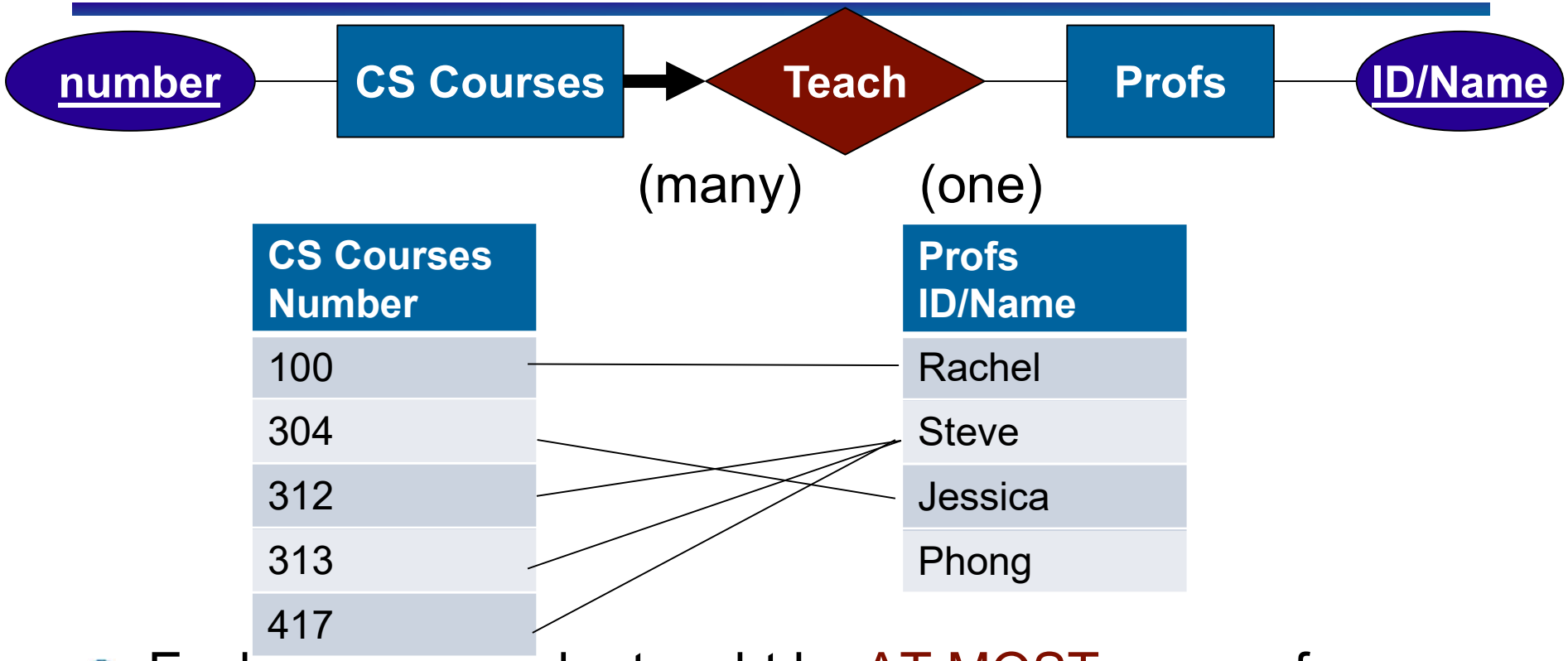
- Each course can be taught by more than one prof
- Each prof can teach more than one course
- All courses have to be taught by some prof
- Not all profs have to teach a course

Thick lines – on both sides



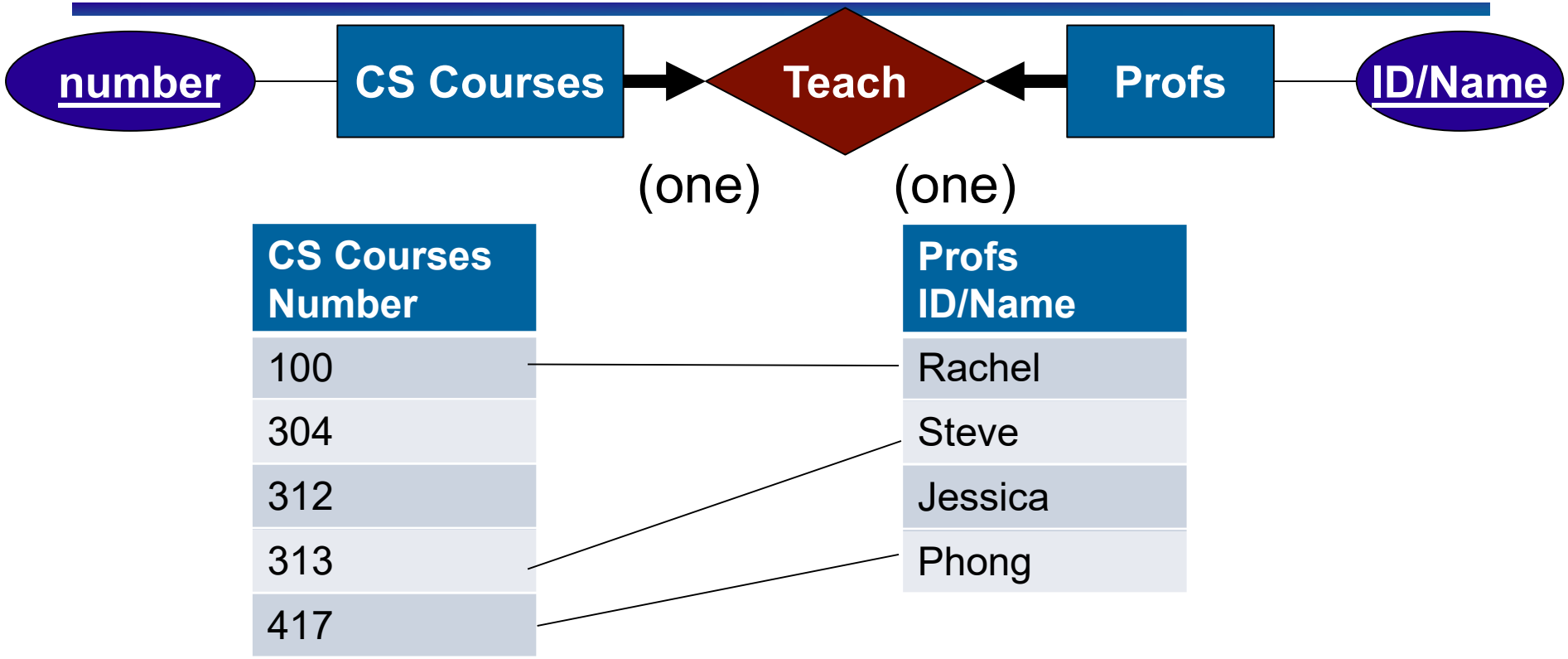
- Each course can be taught by more than one prof
- Each prof can teach more than one course
- ALL courses have to be taught by some prof
- ALL profs have to teach a course

Thick lines with arrows – on one side



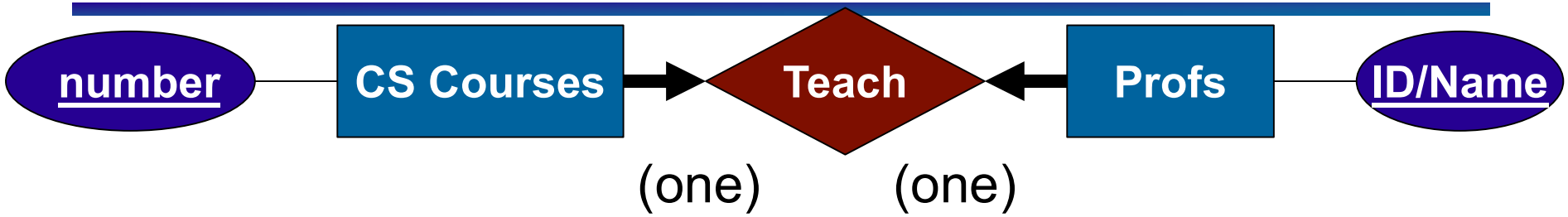
- Each course can be taught by AT MOST one prof
- Each prof can teach more than one course
- ALL courses have to be taught by some prof
- Not all profs have to teach a course

Clicker question: Thick lines with arrows – on both sides: Which of the following would allow us to satisfy the given constraint?



A. Adding {(304,Jessica)} to Teach
B. Adding {(304, Jessica), (312, Steve)} to Teach
C. Both of the above
D. None of the above

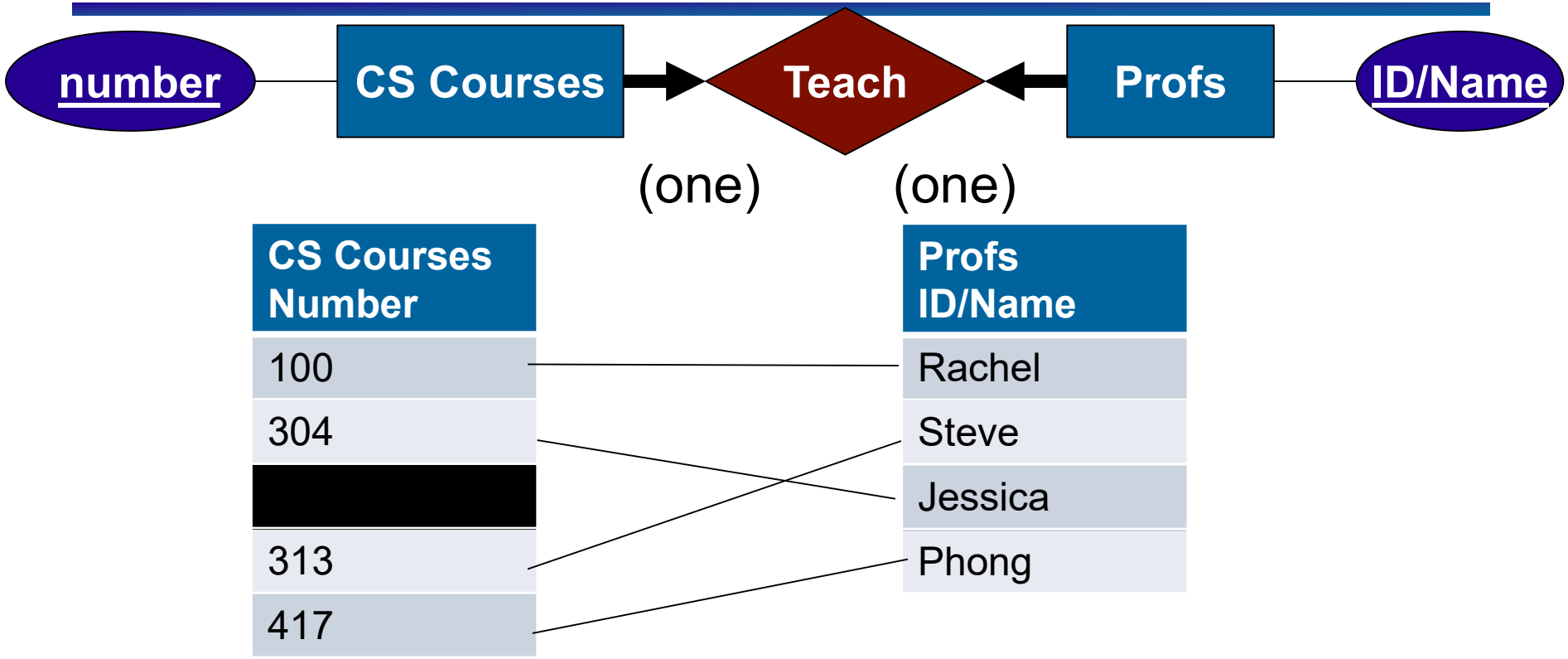
Clicker question: Thick lines with arrows – on both sides: Which of the following would allow us to satisfy the given constraint?



CS Courses Number	Profs ID/Name
100	Rachel
304	Steve
312	Jessica
313	Phong
417	

A. Adding {(304,Jessica)} to Teach
B. Adding {(304, Jessica), (312, Steve)} to Teach
C. Both of the above
D. None of the above

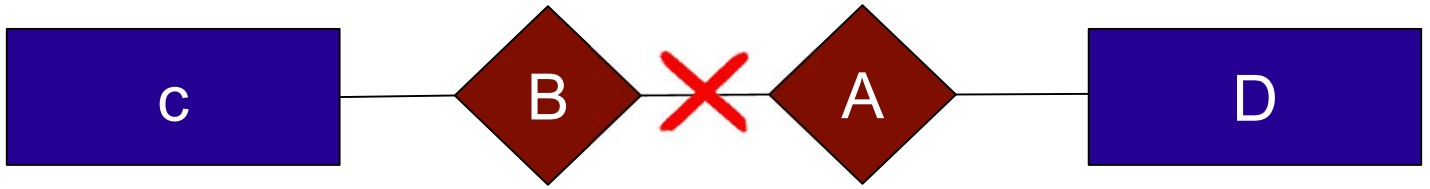
Thick lines with arrows – on both sides



- Each course can be taught by AT MOST one prof
- Each prof can teach AT MOST one course
- ALL courses have to be taught by some prof
- ALL profs have to teach a course
- **Some course has to go!**

Aggregation

- Having a relationship between relationships is forbidden.



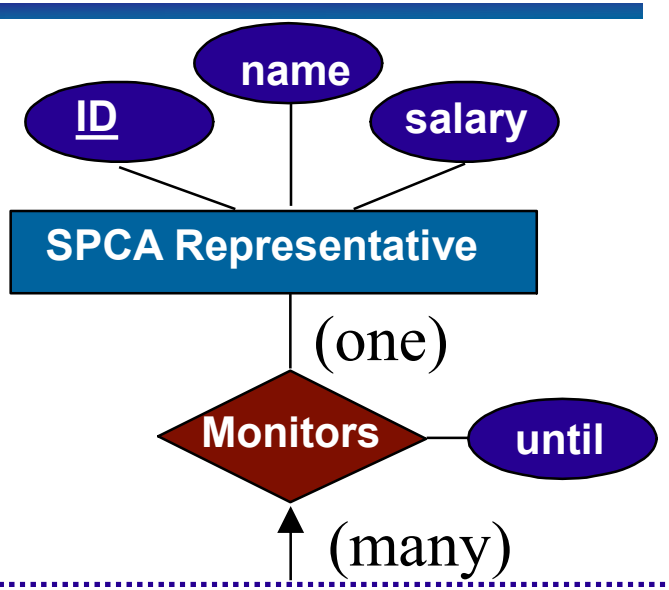
- *Aggregation* allows us to treat a relationship set as an entity set for purposes of participation in (other) relationships

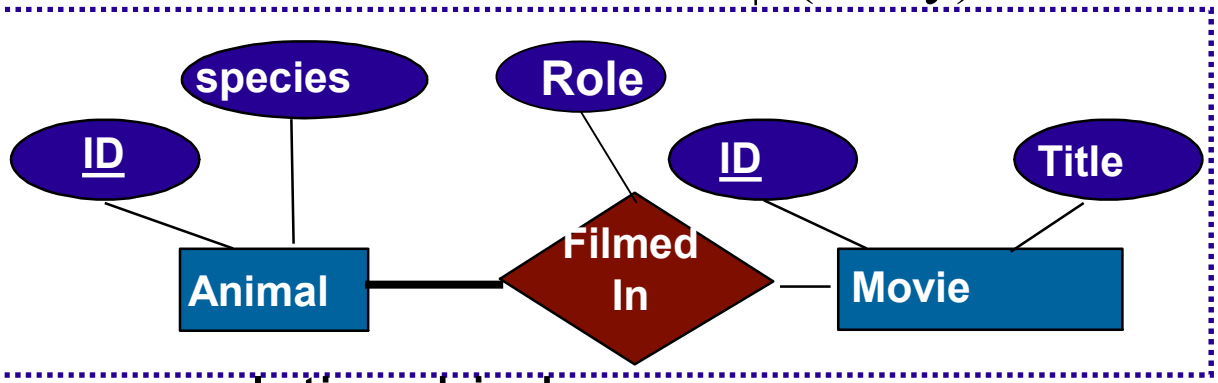
Aggregation: getting around relationships between relationships

- The Society for the Prevention of Cruelty to Animals (SPCA) monitors movies. Each sponsorship is monitored by at most one SPCA representative
- What is the key for FilmedIn?

  Animal ID, movie ID
- What is the key for Monitors?

  Animal ID, movie ID
- This differs from a ternary relationship because monitors is its own relationship with a descriptive attribute





Clicker Exercise

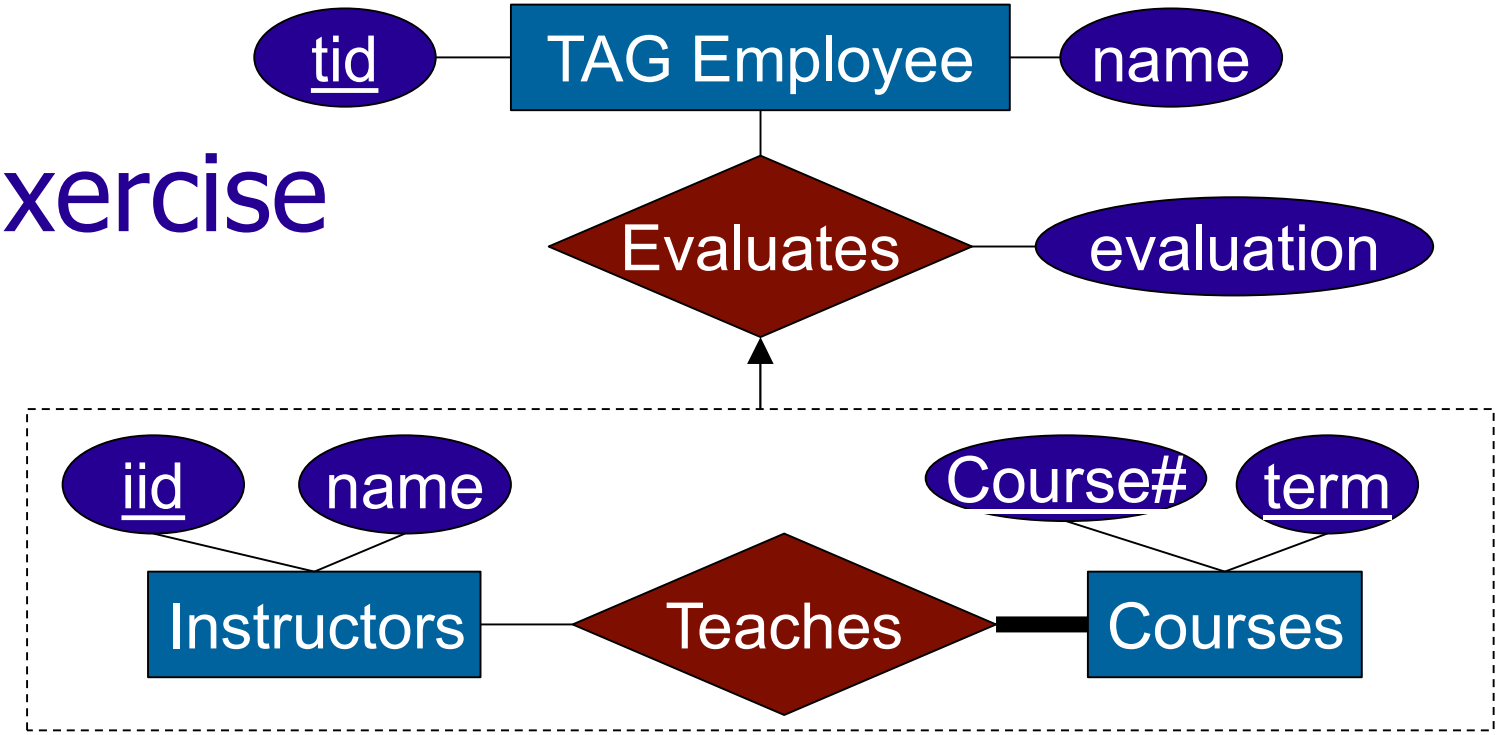


Figure out the (minimal) keys for each entity set and each relationship set in the above diagram.

Clicker Exercise

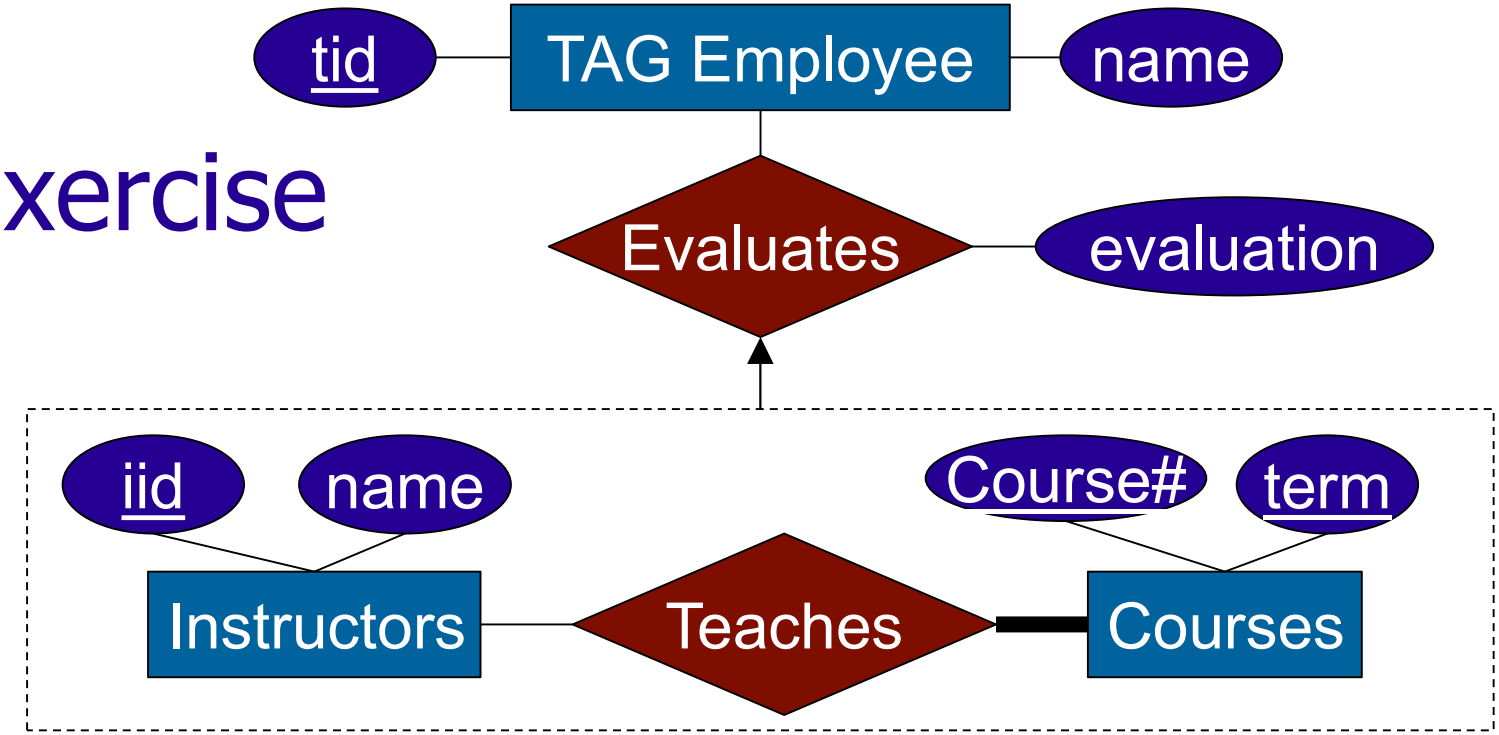
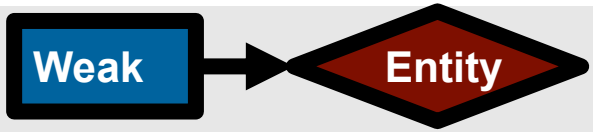
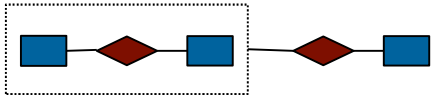
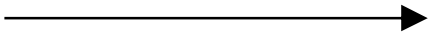


Figure out the (minimal) keys for each entity set and each relationship set in the above diagram.

Choose the correct choice of (minimal) key from the options below:

A. The (minimal) key of Evaluates is tid
B. The (minimal) key of Evaluates is iid + course# + term.
C. The (minimal) key of Evaluates is iid + course# + term + tid
D. The (minimal) key of Evaluates is iid+ course# + term + evaluation
E. None of the above

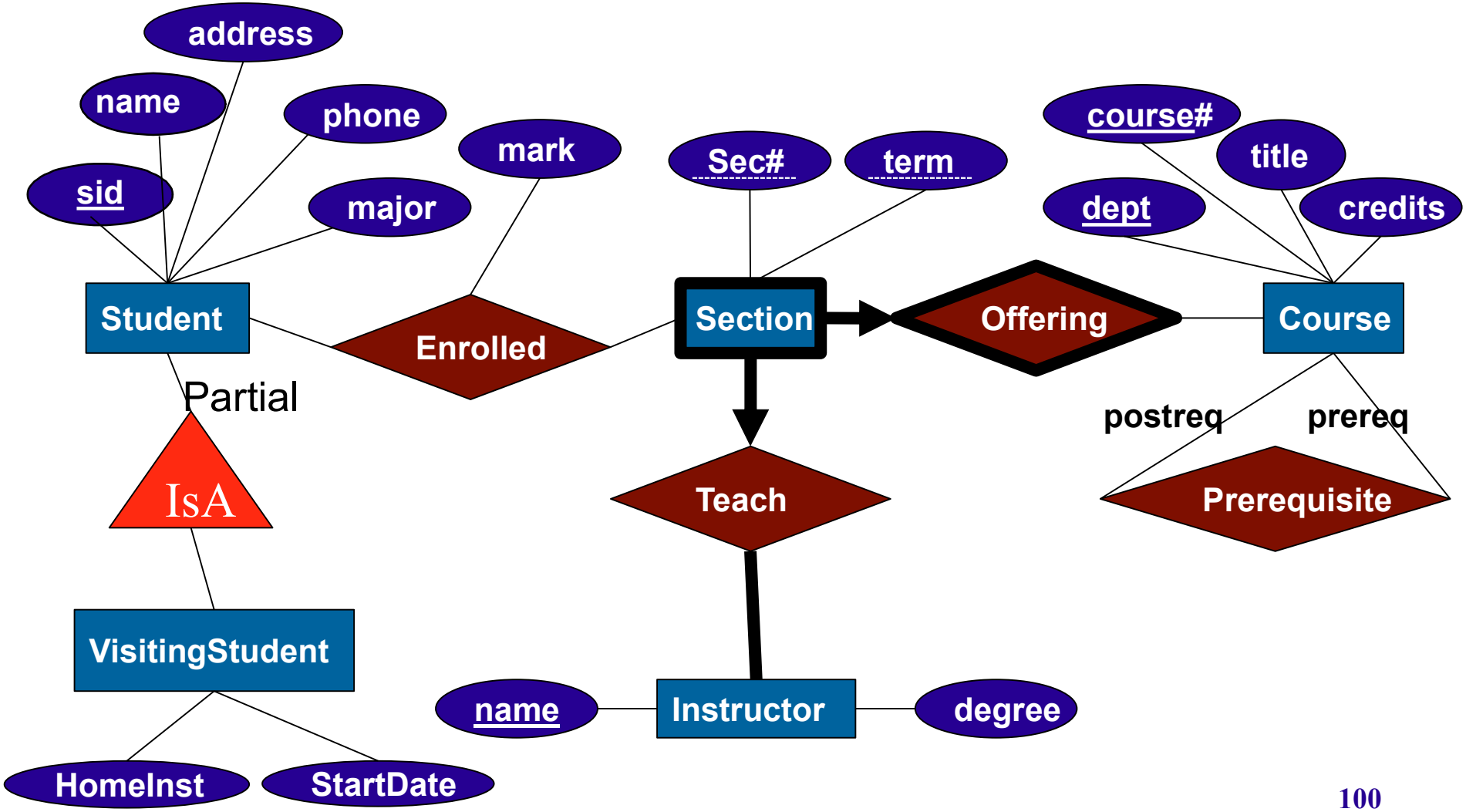
Summary

Name	Symbol
Entity	
Attribute	
Relationship	
Generalization/Specialization	
Weak Entity	
Aggregation	
Participation Constraints	
key constraints	

Exercise: Workday Student Minus (not to turn in)

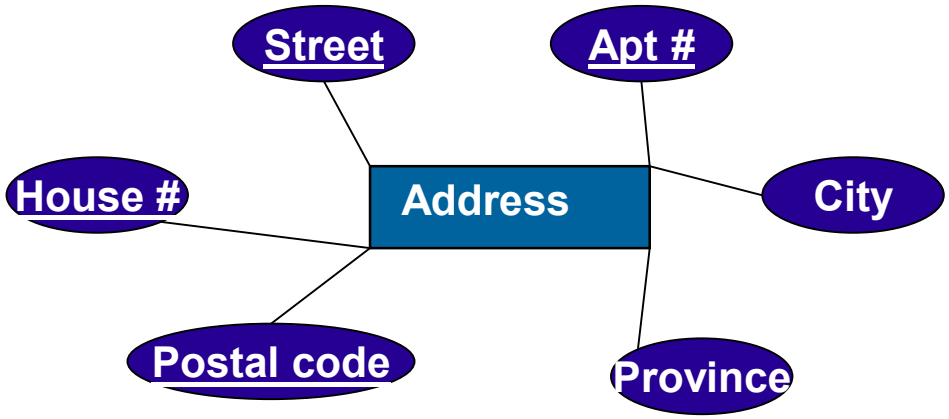
- The primary function of UBC is to offer courses to students.
- A student is identified by a unique student #, and has a name, address, and phone #. Each student is registered in a major at UBC.
- Visiting students stay at UBC for a year and come from a home institution.
- A course offered by UBC is identified by the department that offers the course and a course# which is unique within the department. We list our courses with their titles and the credits in our calendar.
- A course may be offered many times, even within the same term. Each offering is assigned a section # which is unique for a given course and term, and is taught by a single instructor.
- Each instructor is responsible for some section; there are no idle instructors. Instructors have unique names, and may teach a # of sections of different courses. For each instructor we like to keep info about their higher degree.
- A student register in a course section and gets a mark for the course.
- A course may have any number of other courses as prerequisites.

Sample solution



That's all there is to it

- Some ER models differ in expressiveness
- They model *most* concepts people want
- They don't model all of them, e.g.,
  - Functional dependencies – some attributes determine some other attributes, e.g., postal code determines (only) city and province



Conceptual Design Using the ER Model

- Design choices:
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?
  - Identifying relationships: Binary or ternary? Aggregation?
- Constraints in the ER Model:
  - A lot of data semantics can (and should) be captured.
  - But some constraints cannot be captured in ER diagrams.
    - i.e. domain constraints
    - dependencies