The Entity/Relationship (E/R) Model & DB Design

Slides by Manos Papagelis, Ryan Johnson, John Mylopoulos, Arnold Rosenbloom, Renee Miller and Diane Horton

Overview

- Using the Entity/Relationship (ER) Model to model the real world
- From there, designing a database schema
  - Restructuring of an E/R model
  - Translating an E/R model into a logical model (DB Schema)

THE ENTITY/RELATIONSHIP (E/R) MODEL

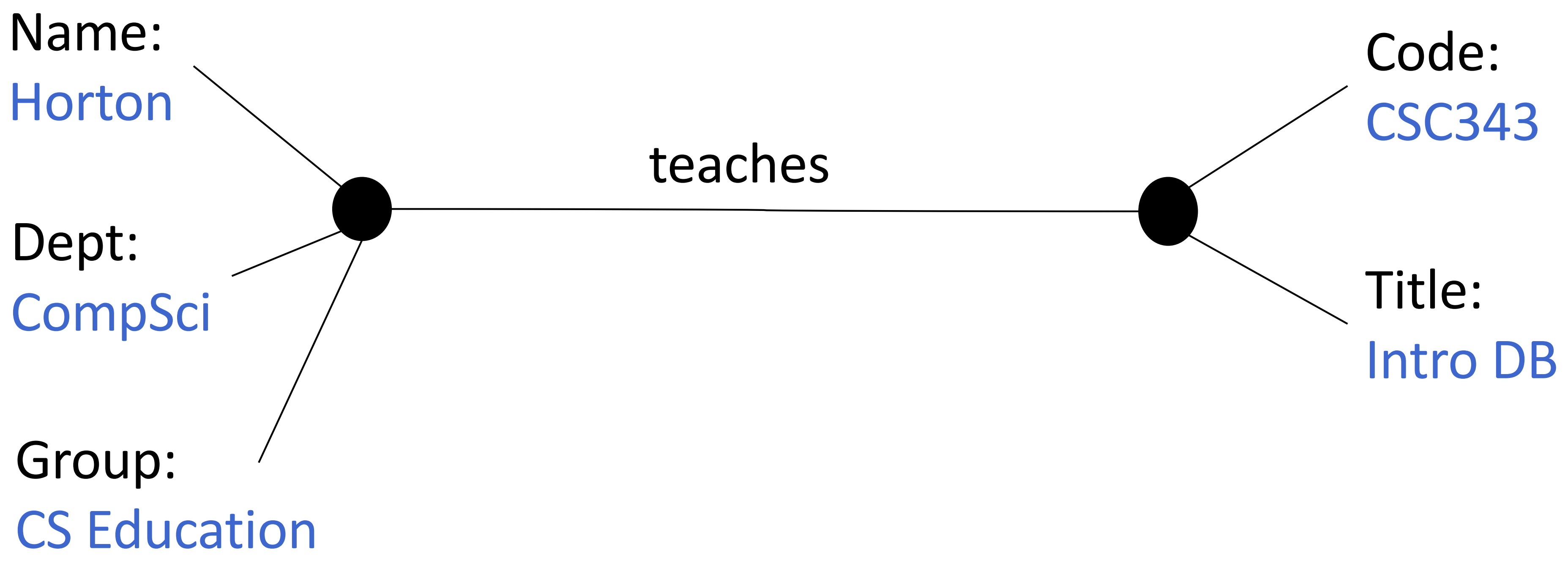
Conceptualizing the real-world

- DB design begins with a boss or client who wants a database.
- We must map the entities and relationships of the world into the concepts of a database. This is called *modeling*.
- Sketching the key components is an efficient way to develop a design.
  - Sketch out (and debug) schema designs
  - Express as many constraints as possible
  - Convert to relational DB once the client is happy

Entity/Relationship Model

- Visual data model (diagram-based)
  - Quickly “chart out” a database design
  - Easier to “see” big picture
  - Comparable to class diagrams in UML
- Basic concepts:
  - *entities*
  - *relationships* among them
  - *attributes* describing the entities and the relationships

Example: 2 entities with a relationship



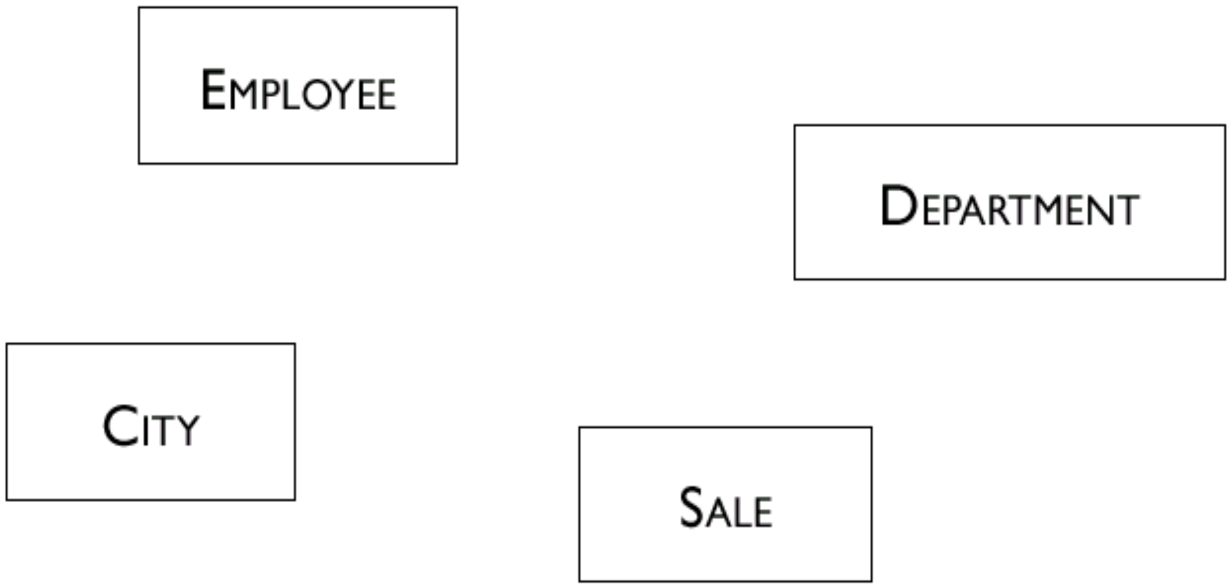
Defining a Schema

- E/R allows us to specify what these structures can and must look like.
- We generalize from specific entities & relationships to the sets they are drawn from.

Instance	Schema
Entity [with attributes]	Entity Set [with attributes]
Relationship [with attributes]	Relationship Set [with attributes]

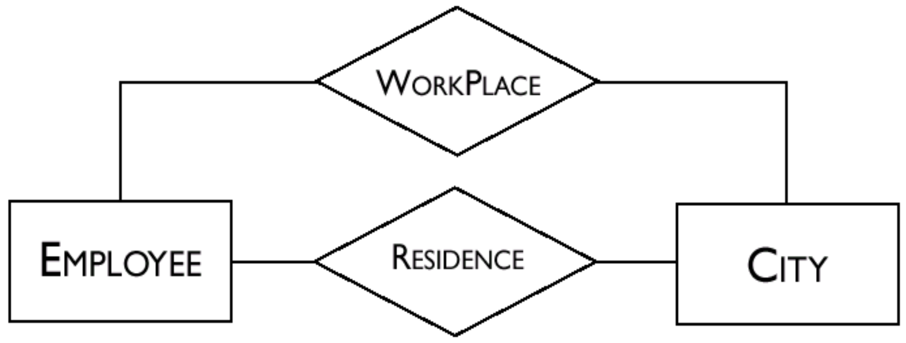
Entity Sets

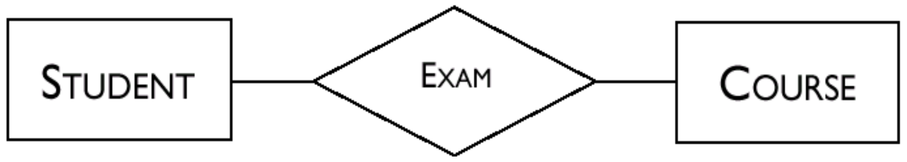
- An entity set represents a *category* of objects that have properties in common and an autonomous existence (e.g., City, Department, Employee, Sale)
- An entity is an *instance* of an entity set (e.g., Stockholm is a City; Peterson is an Employee)



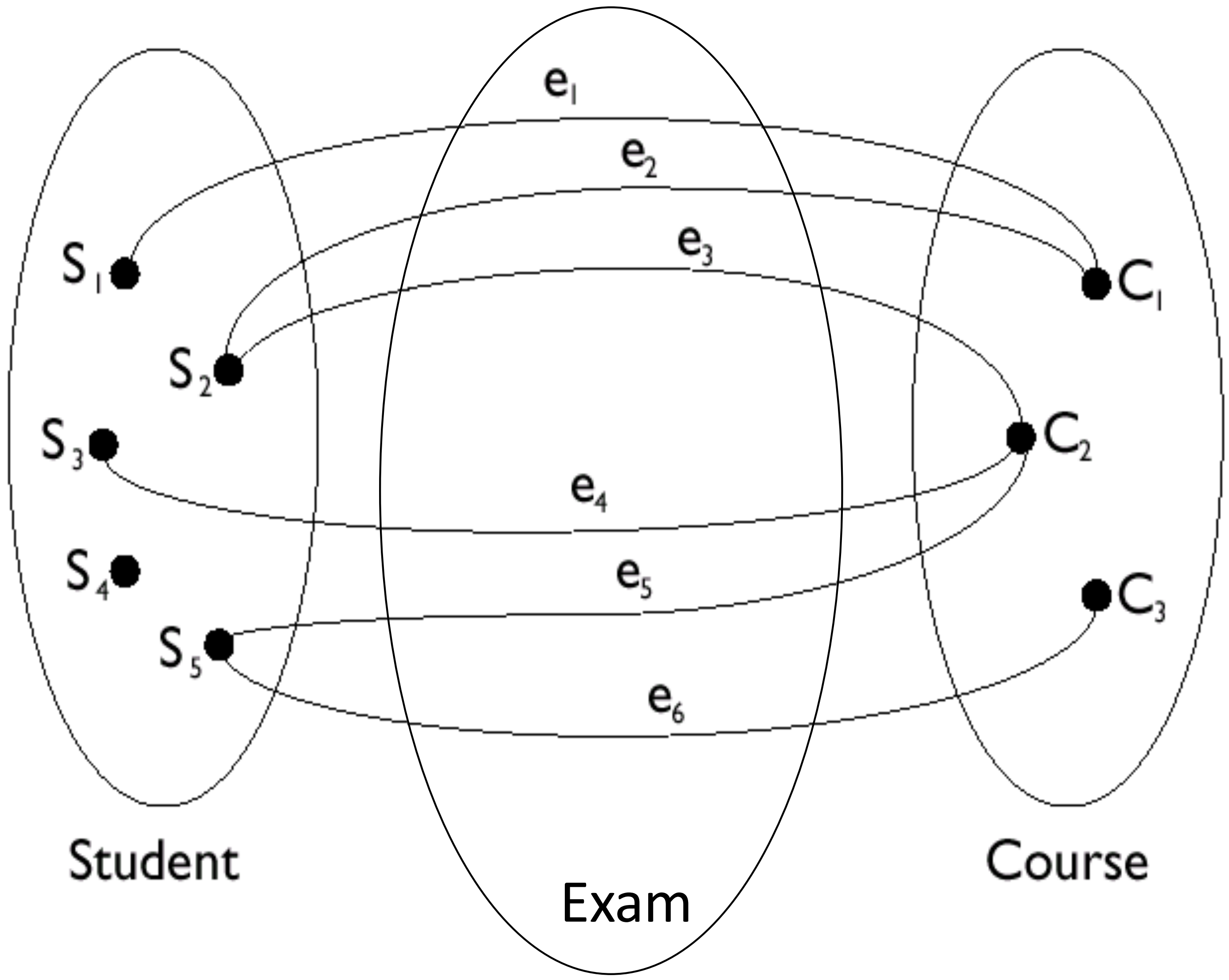
Relationship Sets

- A relationship set is an association between 2+ entity sets (e.g., Residence is a relationship set between entity sets City and Employee)
- A relationship is an instance of a n-ary relationship set (e.g., the pair <Johanssen, Stockholm> is an instance of relationship Residence)



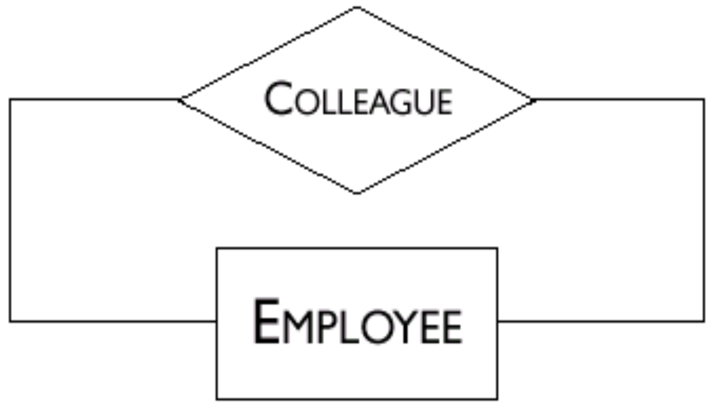


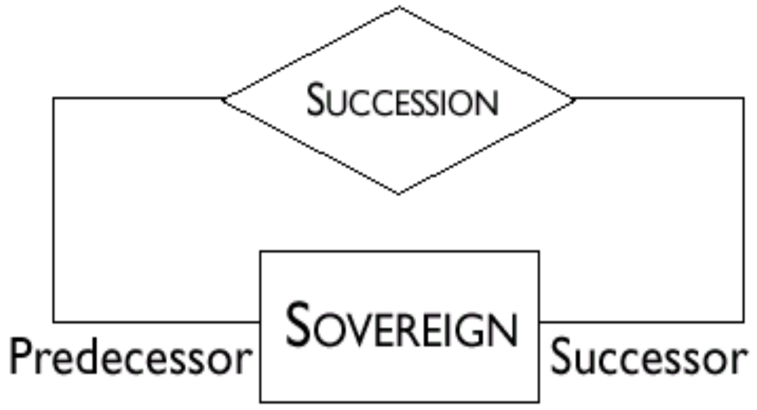
Example Instance



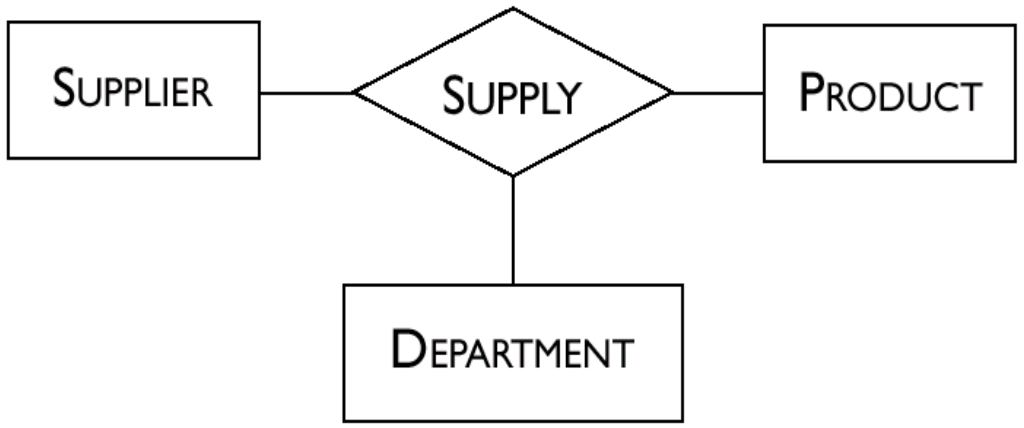
Recursive Relationships

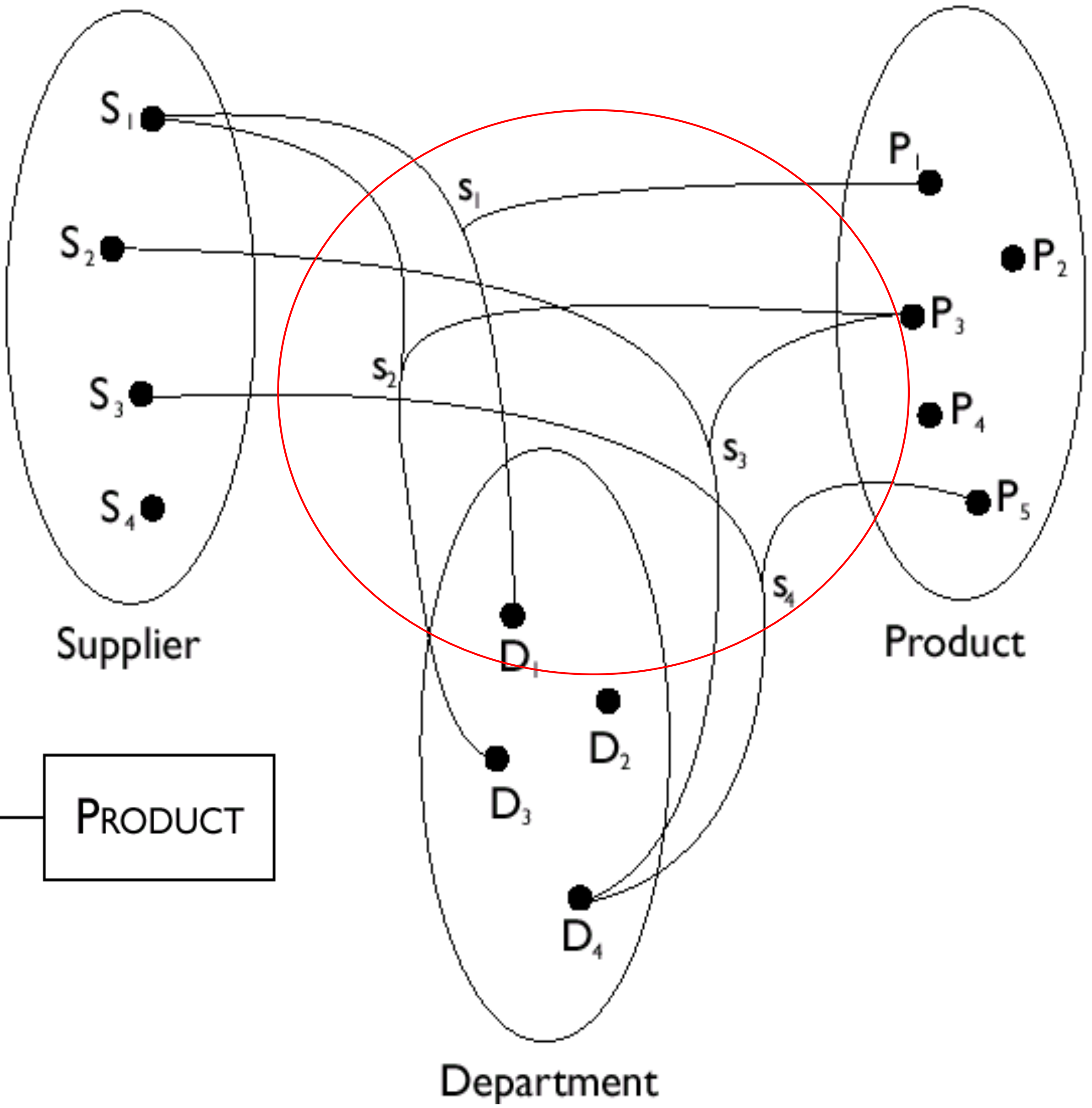
- Recursive relationships relate an entity set to itself
- The relationship may be asymmetric
  - If so, we indicate the two roles that the entity plays in the relationship





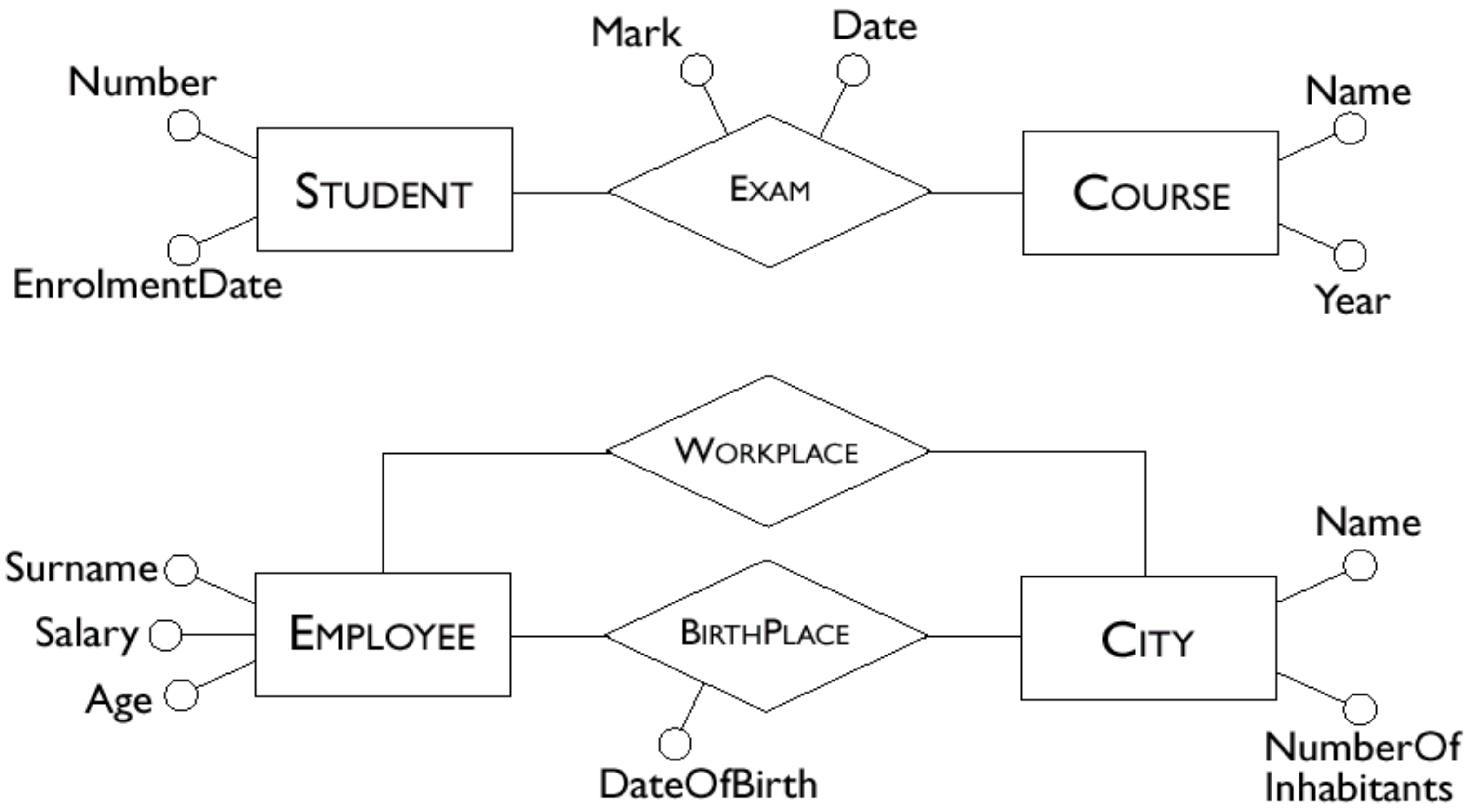
Ternary Relationships





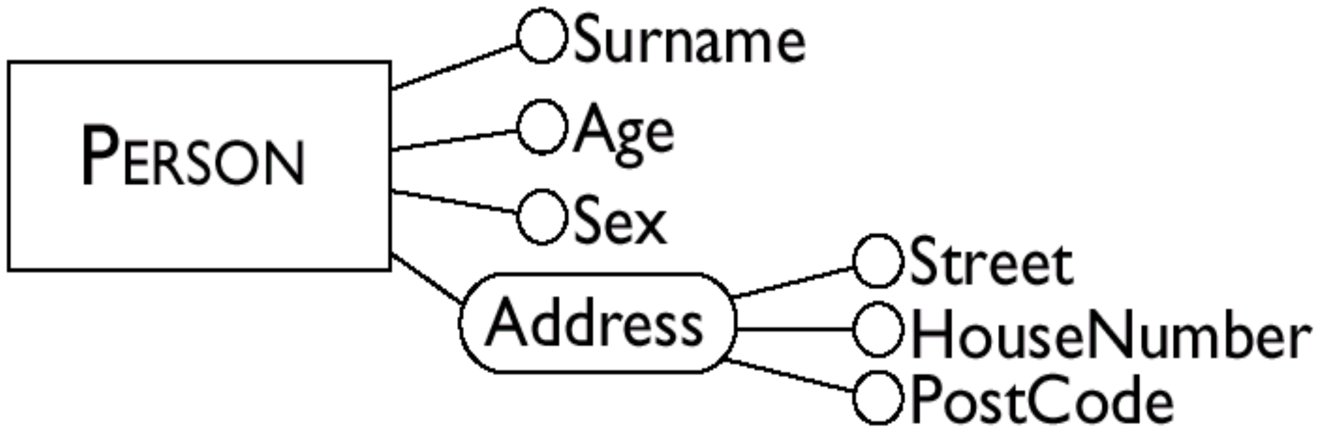
Attributes

- Describe elementary properties of entities or relationships (e.g., Surname, Salary and Age are attributes of Employee)
- May be single-valued, or multi-valued

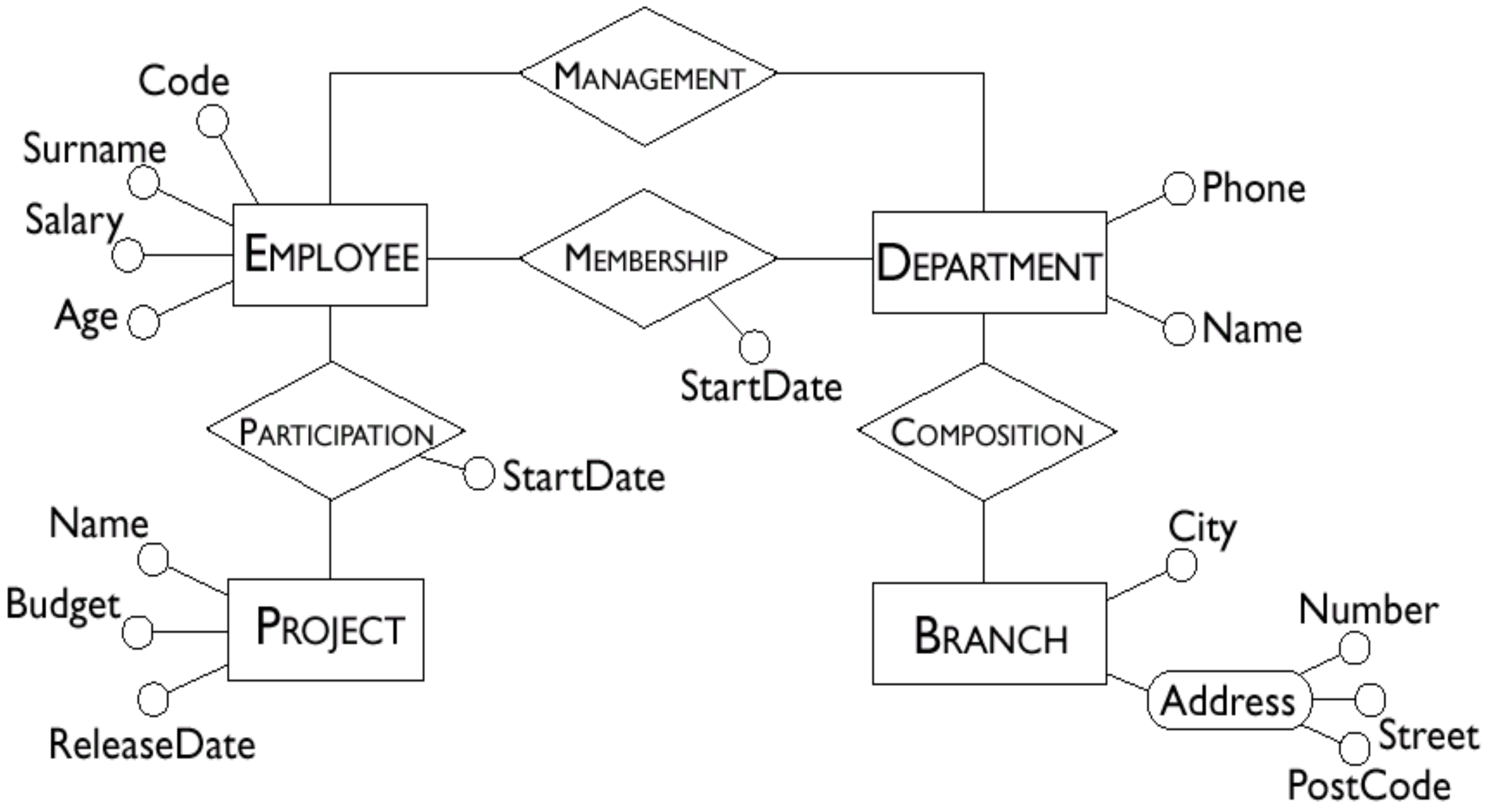


Composite Attributes

- composite attributes are grouped attributes of the same entity or relationship that have closely connected meaning or uses

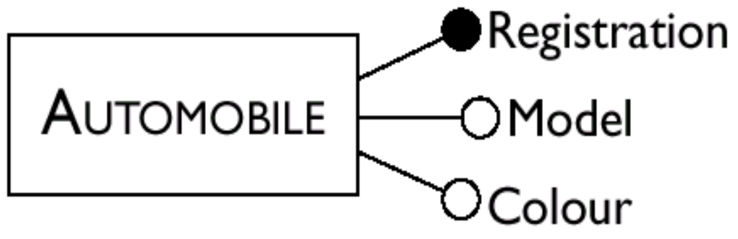


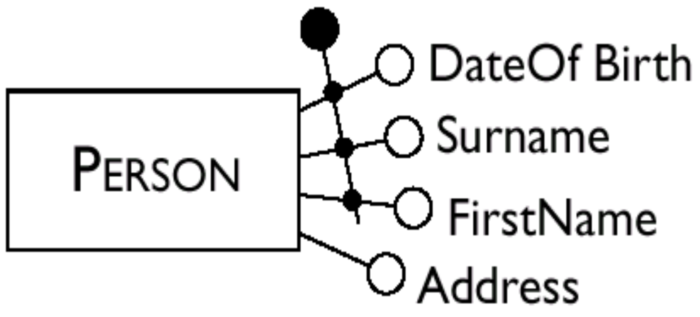
Example Schema with Attributes



Keys in E/R

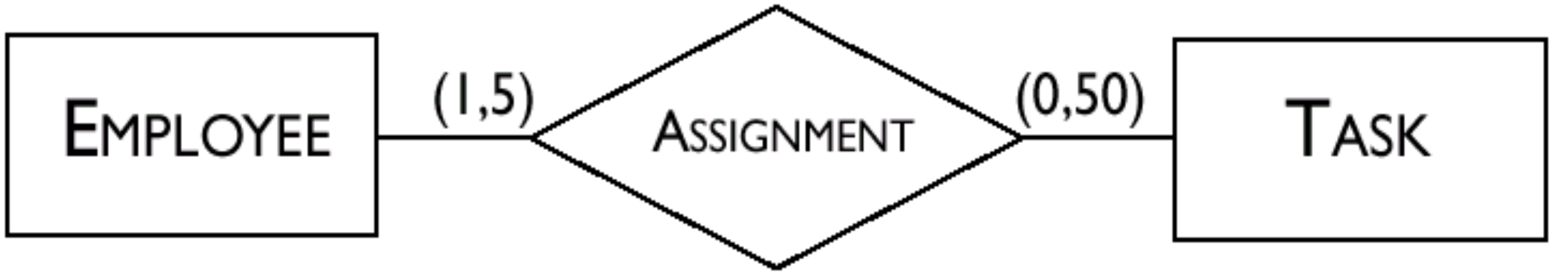
- Notation: solid circle
- If multi-attribute, connect with a line and a “knob”





Cardinalities

- Each entity set participates in a relationship set with a minimum (min) and a maximum (max) cardinality
- Cardinalities constrain how entity instances participate in relationship instances
- Notation: pairs of (min, max) values for each entity set



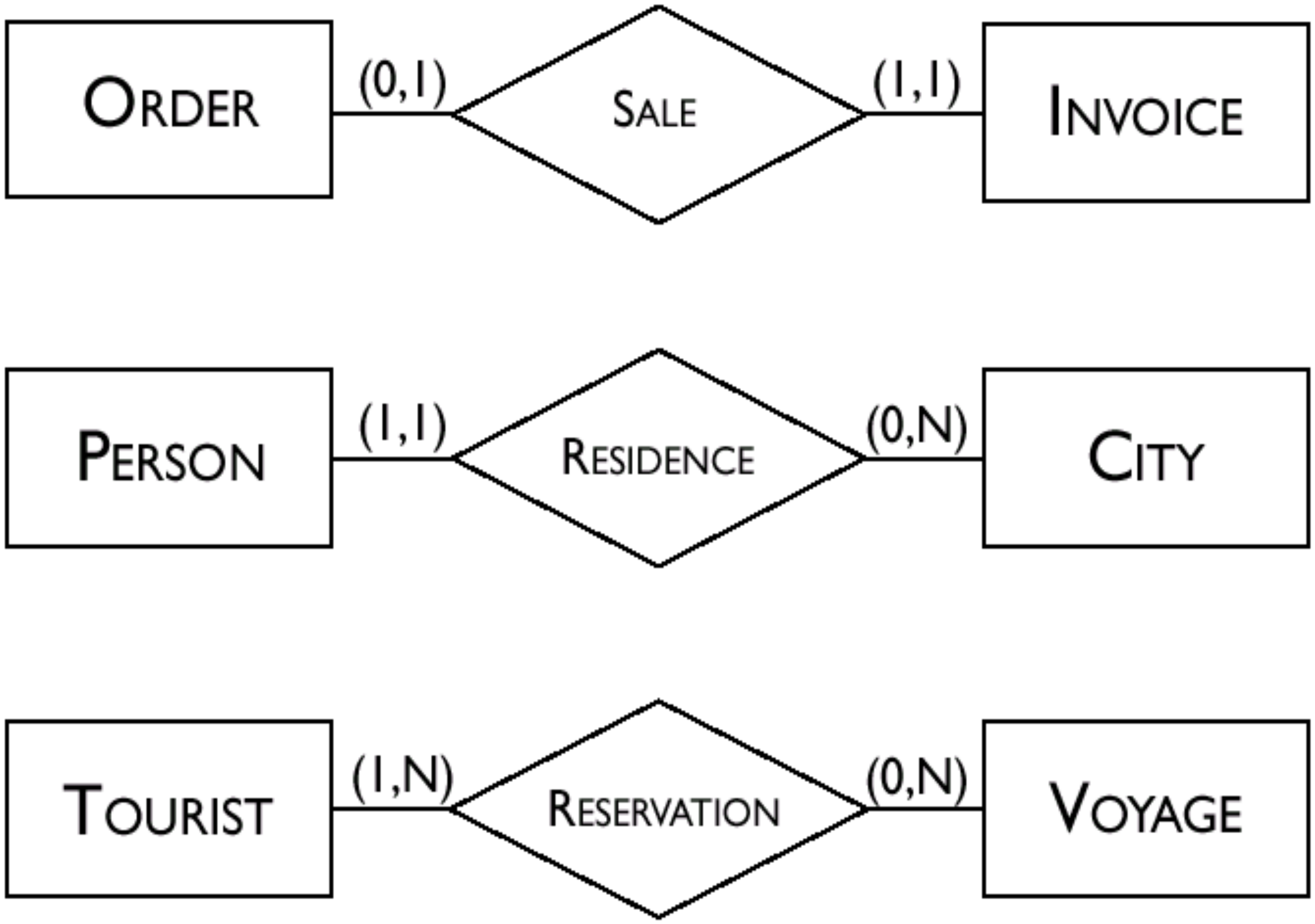
How this constrains instances

How this constrains instances

Cardinalities

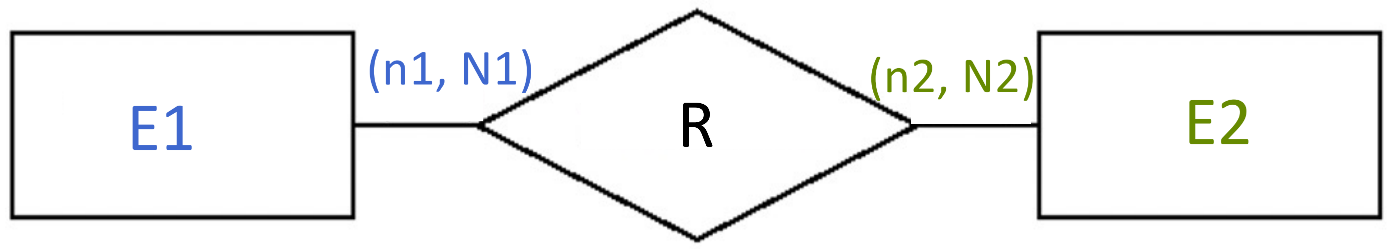
- Cardinalities are pairs of non-negative integers (min, max) such that min ≤ max.
- minimum cardinality min:
  - If 0, entity participation in the relationship is optional
  - If 1, entity participation in the relationship is mandatory
  - Other values are possible
- maximum cardinality max:
  - If 1, each instance of the entity is associated at most with a single instance of the relationship
  - If > 1, then each instance of the entity can be associated multiple instances of the relationship
  - We write N to indicate no upper limit
  - Other values are possible

Cardinality Examples



Multiplicity of relationships

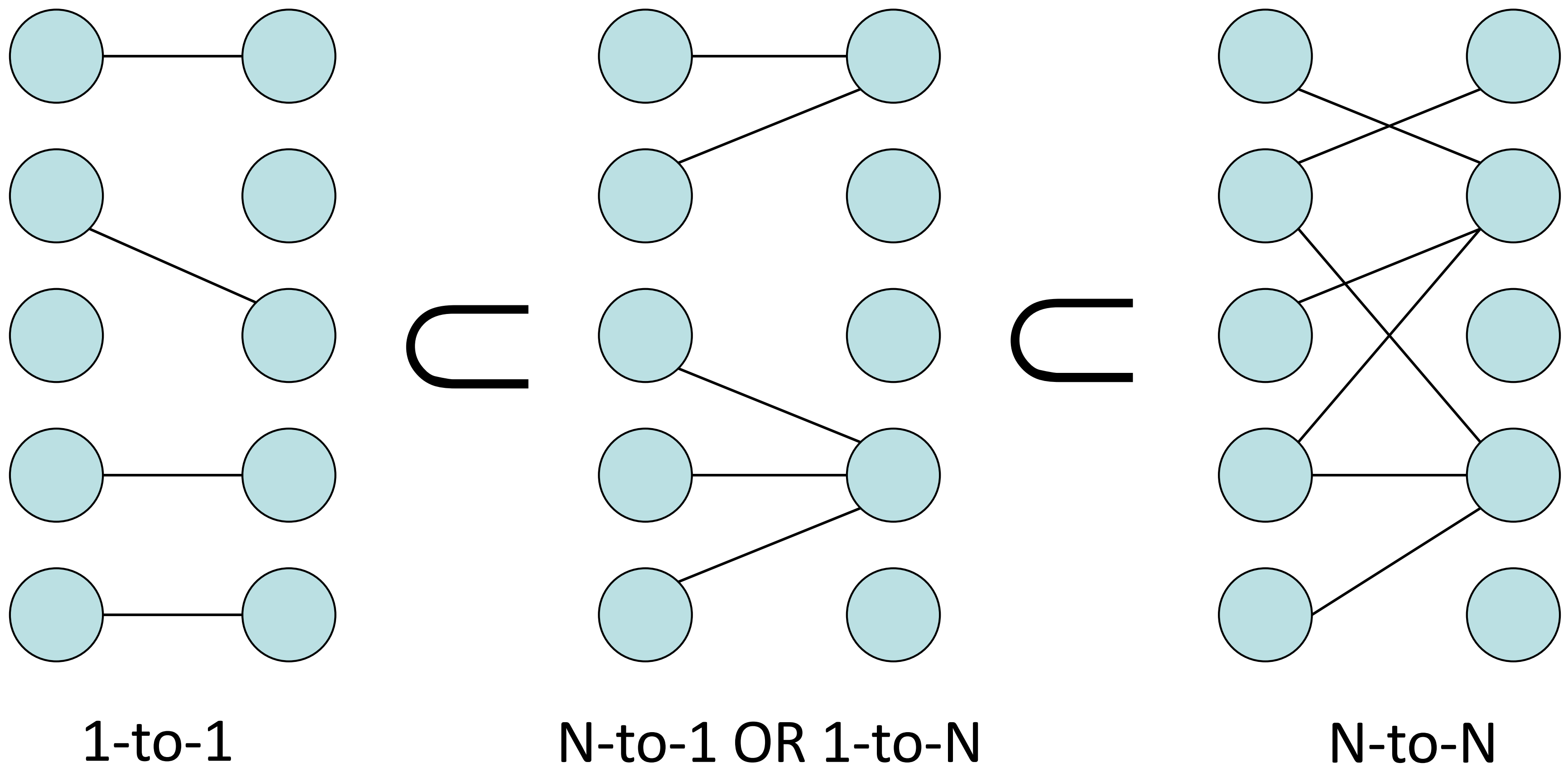
Suppose entity sets E1 and E2 participate in relationship R with cardinalities (n1, N1) and (n2, N2):



We say that the multiplicity of R is N1-to-N2,
which is the same as saying it is N2-to-N1.

Multiplicity of relationships

These particular multiplicities are probably familiar:



They convey much less information than our (min, max) notation.

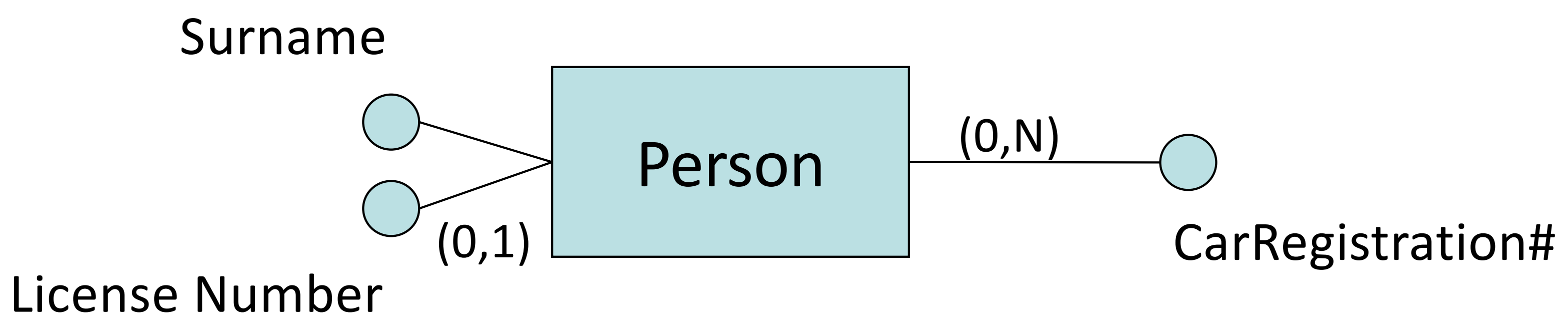
Multiplicity of relationships

- 1-to-1 means no ”branching” is allowed on either side.
- 1-to-N means branching is allowed on one side.
  - You must examine the ER diagram to find out which side.
  - The wording (1-to-N vs N-to-1) does not tell you.
- N-to-N means branching is allowed on either side.

- Conveys less information than our (min, max) notation.
  - It only says what the maxs are, not the mins.
  - For example, a 1-to-1 relationship could be:
    - (0, 1) and (0, 1)
    - (1, 1) and (1, 1)
    - (0, 1) and (1, 1)
    - (1, 1) and (0, 1)

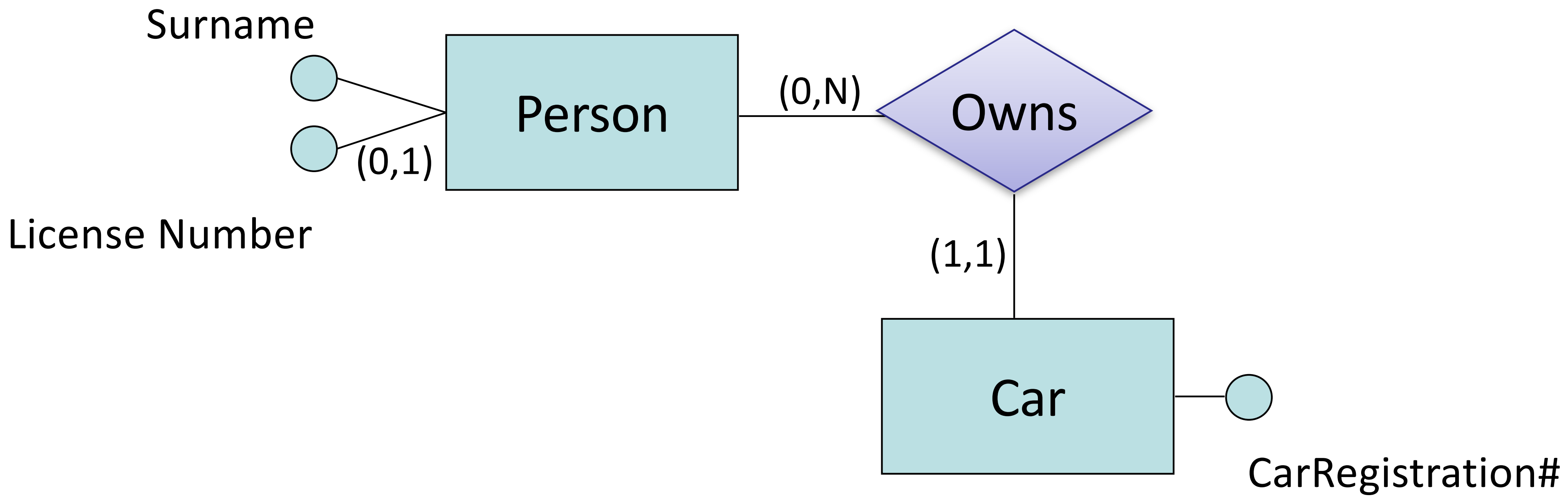
Cardinalities of Attributes

- Describe min/max number of values an attribute can have
- When the cardinality of an attribute is (1, 1) it can be omitted from the diagram. This is a single-valued attribute.
- The value of an attribute may also be null, or have several values (that is a multi-valued attribute).



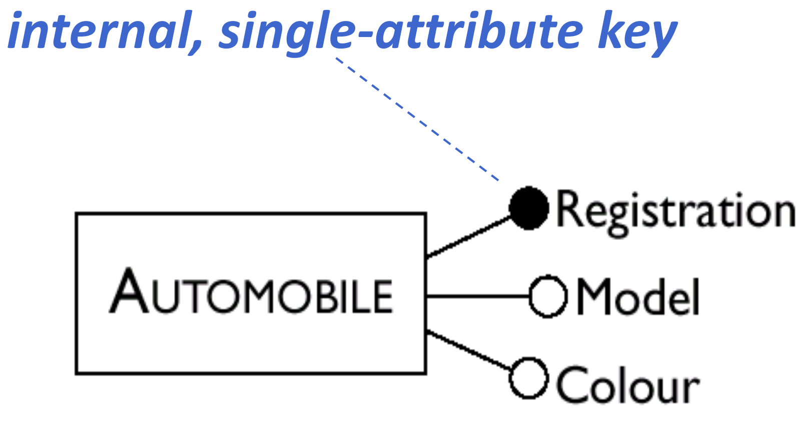
Cardinalities of Attributes (cont.)

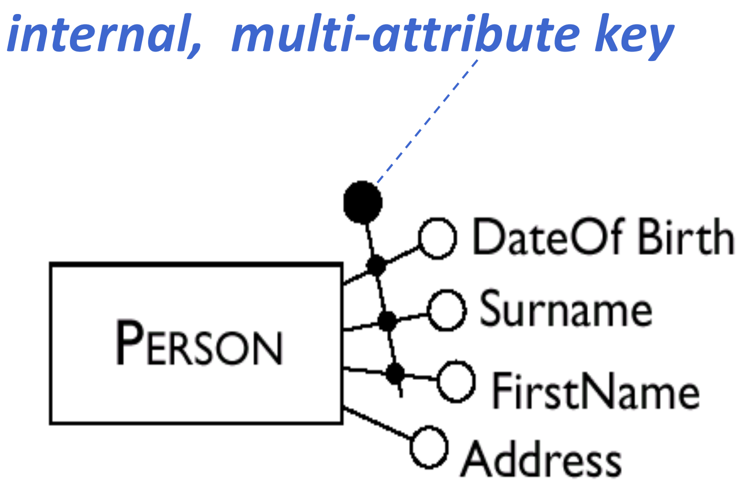
- Multi-valued attributes often represent situations that can be modeled with additional entities. E.g., the ER schema of the previous slide can be revised into:



Recall keys in E/R

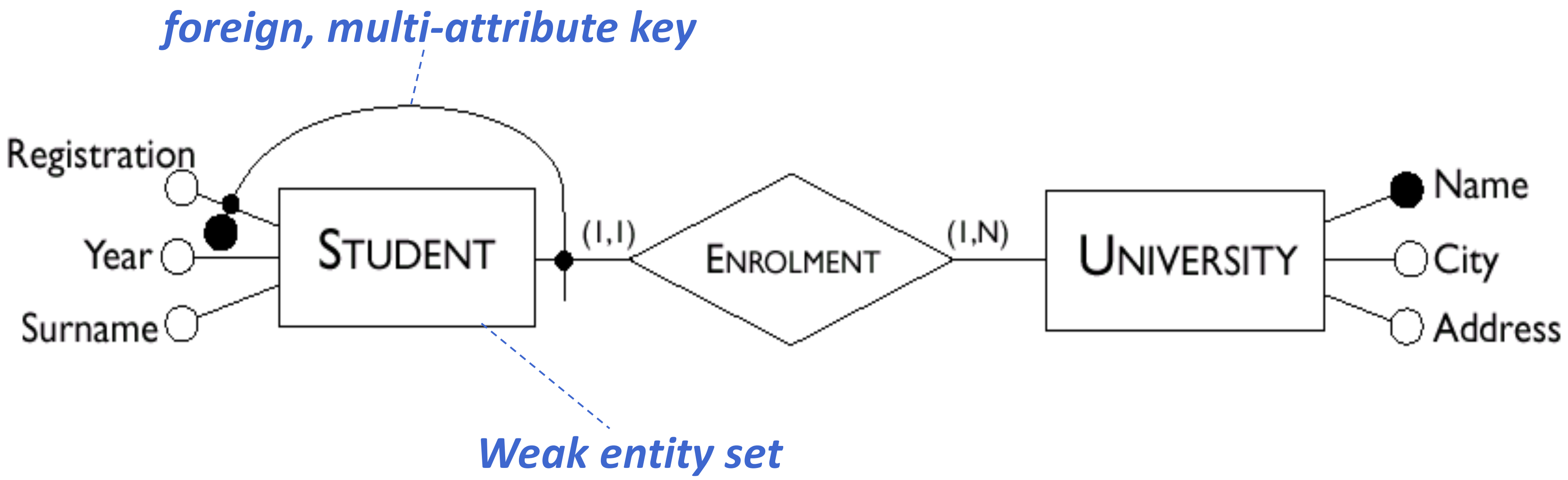
- Usually, a key is formed by one or more attributes of the entity itself. This is an *internal* key.





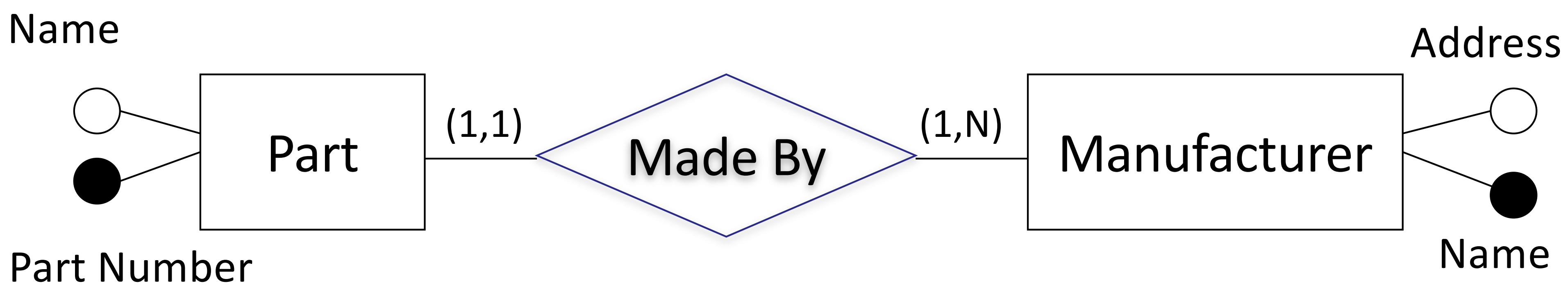
Weak entity sets

- Sometimes, an entity set doesn't have a key among its attributes. This is called a *weak entity set*.
- Solution: the keys of related entities are brought in to help with identification (becoming *foreign keys*).



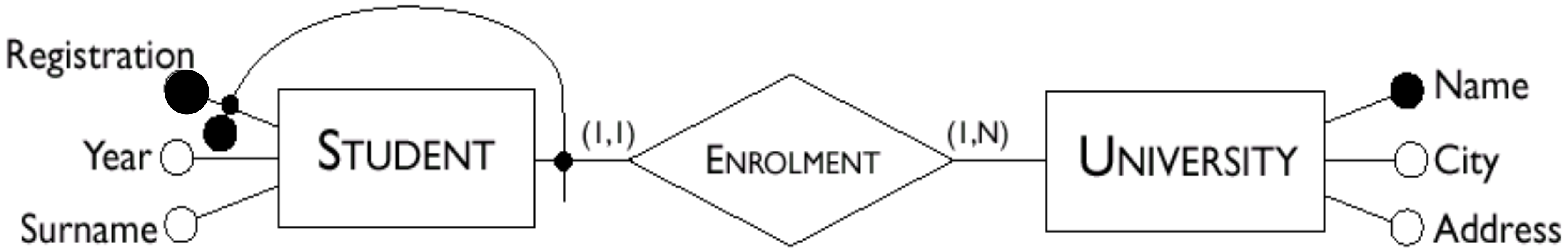
Keys of relationship sets

- The key for a relationship set consists of the keys of the entity sets that it relates.
- In this example, the key of the Made By relationship set is Part Number and Name.



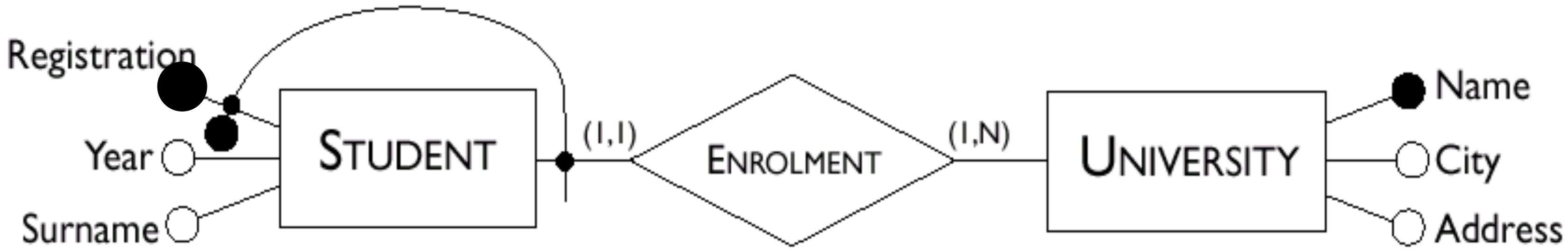
Requirements for Keys

- Each attribute in a key must have (1,1) cardinality.
- A foreign key for a weak entity set must come through a relationship which the entity set participates in with cardinality (1,1).
- E.g., what if a student could enrol at > 1 university?

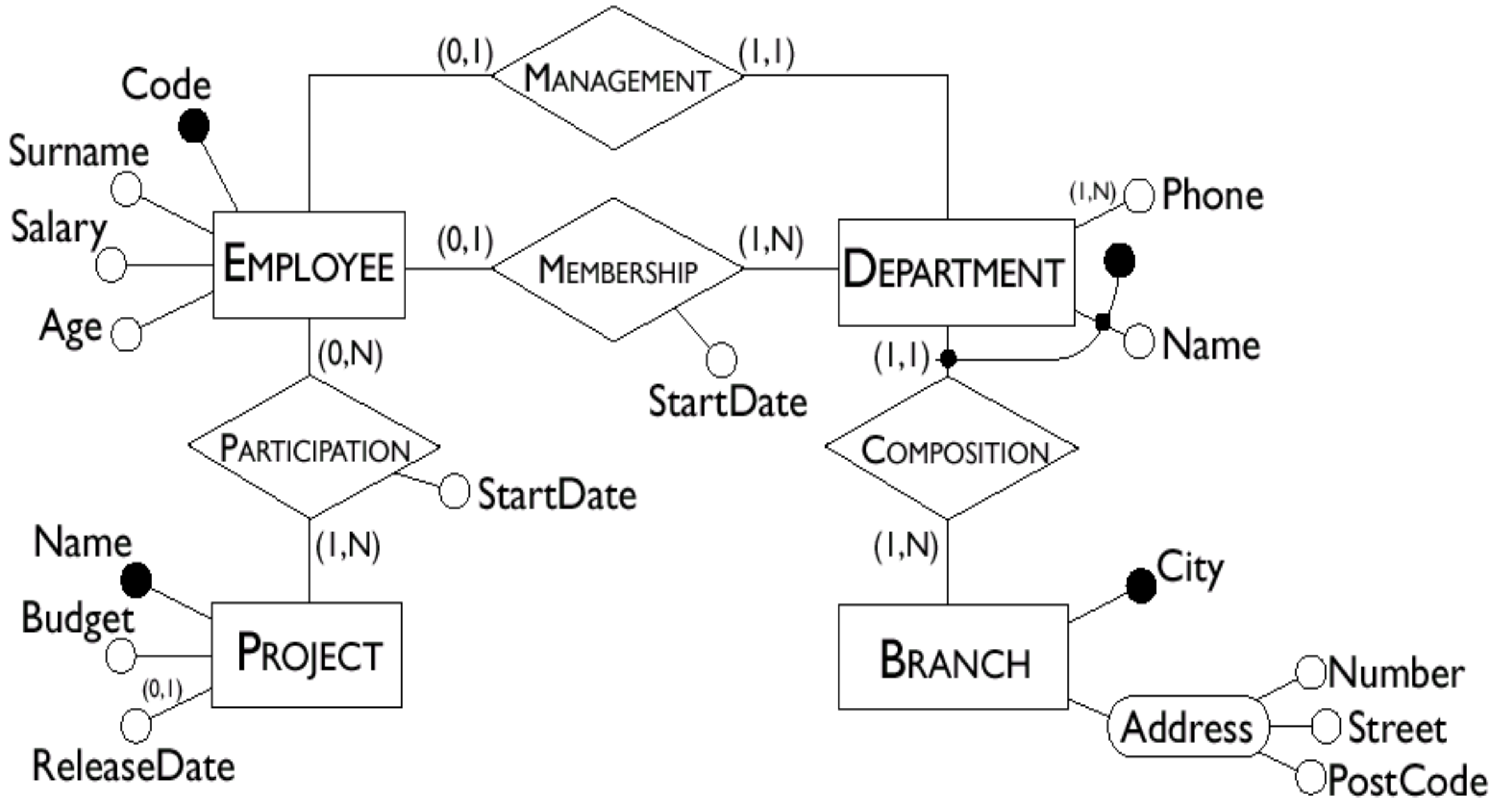


Requirements for Keys

- A foreign key may involve an entity that has itself a foreign key, as long as cycles are not generated.
- Each entity set must have at least one (internal or foreign) key.
- E.g., What if universities can have the same name?



A larger schema with keys



Challenge: modeling the "real world"

- Life is arbitrarily complex
  - Directors who are also actors? Actors who play multiple roles in one movie? Animal actors?
- Design choices: Should a concept be modeled as an entity, an attribute, or a relationship?
- Limitations of the E/R Model: A lot of data semantics can be captured but some cannot
- Key to successful model: parsimony
  - As complex as necessary, but no more
  - Choose to represent only "relevant" things

EXAMPLE

From real world to E/R Model

We wish to create a database for a company that runs training courses. For this, we must store data about trainees and instructors. For each course participant (about 5,000 in all), identified by a code, we want to store her social security number, surname, age, sex, place of birth, employer's name, address and telephone number, previous employers (and periods employed), the courses attended (there are about 200 courses) and the final assessment for each course. We need also to represent the seminars that each participant is attending at present and, for each day, the places and times the classes are held.

Each course has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an 'edition' of the course. For each edition, we represent the start date, the end date, and the number of participants. If a trainee is self-employed, we need to know her area of expertise, and, if appropriate, her title. For somebody who works for a company, we store the level and position held. For each instructor (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the tutor is qualified to teach. All the instructors' telephone numbers are also stored. An instructor can be permanently employed by the training company or freelance.

From real world to E/R Model

We wish to create a database for a company that runs training courses. For this, we must store data about the *trainees* and the *instructors*. For each *course participant* (about 5,000), identified by a code, we want to store her social security number, surname, age, sex, place of birth, employer's name, address and telephone number, previous employers (and periods employed), the courses attended (there are about 200 courses) and the final assessment for each course. We need also to represent the *seminars* that each participant is attending at present and, for each day, the places and times the classes are held.

Each *course* has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an 'edition' of the course. For each edition, we represent the start date, the end date, and the number of participants. If a trainee is self-employed, we need to know her area of expertise, and, if appropriate, her title. For somebody who works for a company, we store the level and position held. For each *instructor* (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the *tutor* is qualified to teach. All the instructors' telephone numbers are also stored. An instructor can be permanently employed by the training company or freelance.

Glossary

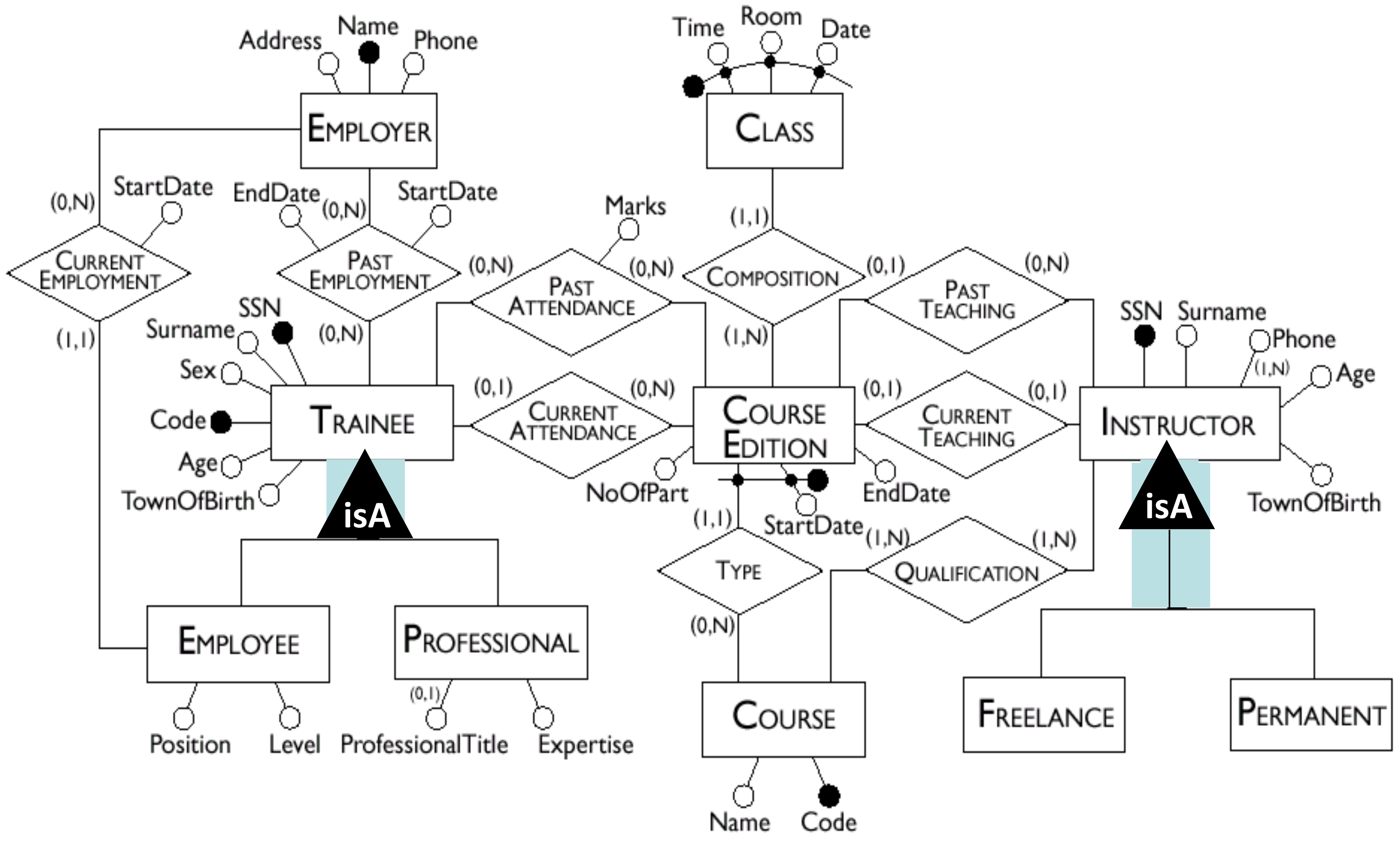
Term	Description	Synonym	Links
Trainee	Participant in a course. Can be an employee or self-employed.	Participant	Course, Company
Instructor	Course tutor. Can be freelance.	Tutor	Course
Course	Course offered. Can have various editions.	Seminar	Instructor, Trainee
Company	Company by which a trainee is employed or has been employed.		Trainee

More Annotations

We wish to create a database for a company that runs training courses. For this, we must store data about *trainees* and *instructors*. For each *course participant* (about 5,000), identified by a code, we want to store her *social security number, surname, age, sex, place of birth, employer's name, address and telephone number, previous employers (and periods employed)*, courses attended (there are about 200 courses) and the final assessment for each course. We need also to represent *seminars* that each participant is attending at present and, *for each day, the places and times the classes are held.*

Each course has a code and a title and any course can be given any number of times. Each time a particular course is given, we will call it an 'edition' of the course. For *each edition, we represent the start date, the end date, and the number of participants.* If a trainee is self-employed, we need to know her area of expertise, and, if appropriate, her title. For somebody who works for a company, we store the level and position held. For each *instructor* (about 300), we will show the surname, age, place of birth, the edition of the course taught, those taught in the past and the courses that the *tutor* is qualified to teach. All the instructors' telephone numbers are also stored. An instructor can be permanently employed by the training company or freelance.

… the E/R model result


Address
Name
Phone
EMPLOYER
Time
Room
Date
CLASS
(0,N)
StartDate
EndDate
(0,N)
StartDate
Marks
(1,1)
CURRENT EMPLOYMENT
PAST EMPLOYMENT
(0,N)
PAST ATTENDANCE
(0,N)
COMPOSITION
(0,1)
PAST TEACHING
(0,N)
(1,1)
SSN
Surname
(0,N)
(1,N)
SSN
Surname
Phone
(1,N)
Age
Sex
Code
TRAINEE
(0,1)
CURRENT ATTENDANCE
(0,N)
COURSE EDITION
(0,1)
CURRENT TEACHING
(0,1)
INSTRUCTOR
Age
TownOfBirth
isA
NoOfPart
EndDate
StartDate
isA
TownOfBirth
(1,1)
(1,N)
TYPE
QUALIFICATION
(1,N)
(0,N)
EMPLOYEE
PROFESSIONAL
COURSE
FREELANCE
PERMANENT
(0,1)
Position
Level
ProfessionalTitle
Expertise
Name
Code