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

Introduction to databases

CSC343, Winter 2016

Based on slides by Manos Papagelis

Thanks to Ryan Johnson, John Mylopoulos, Arnold Rosenbloom and Renee Miller for material in these slides

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)

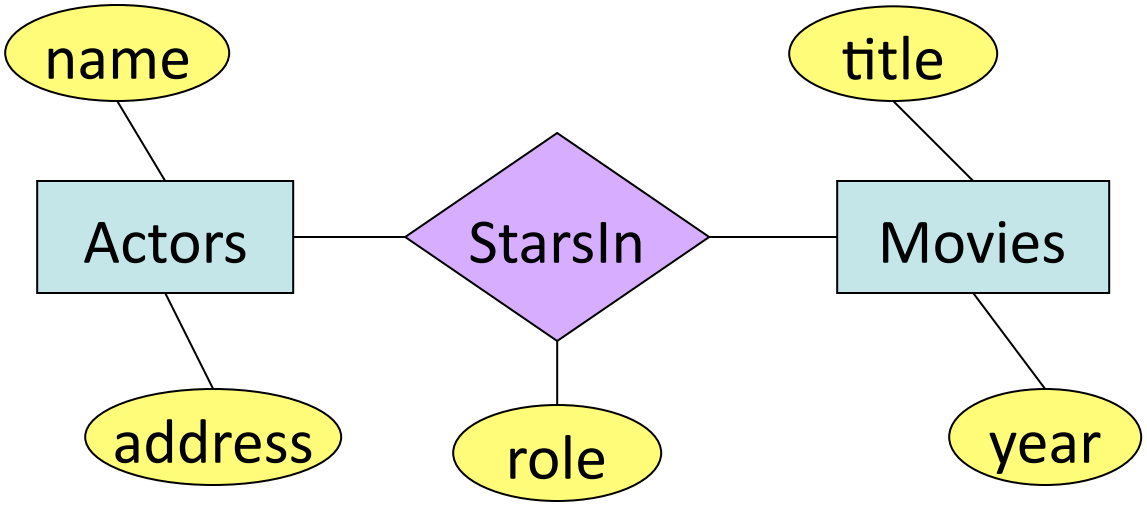
THE ENTITY/RELATIONSHIP (E/R) MODEL

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

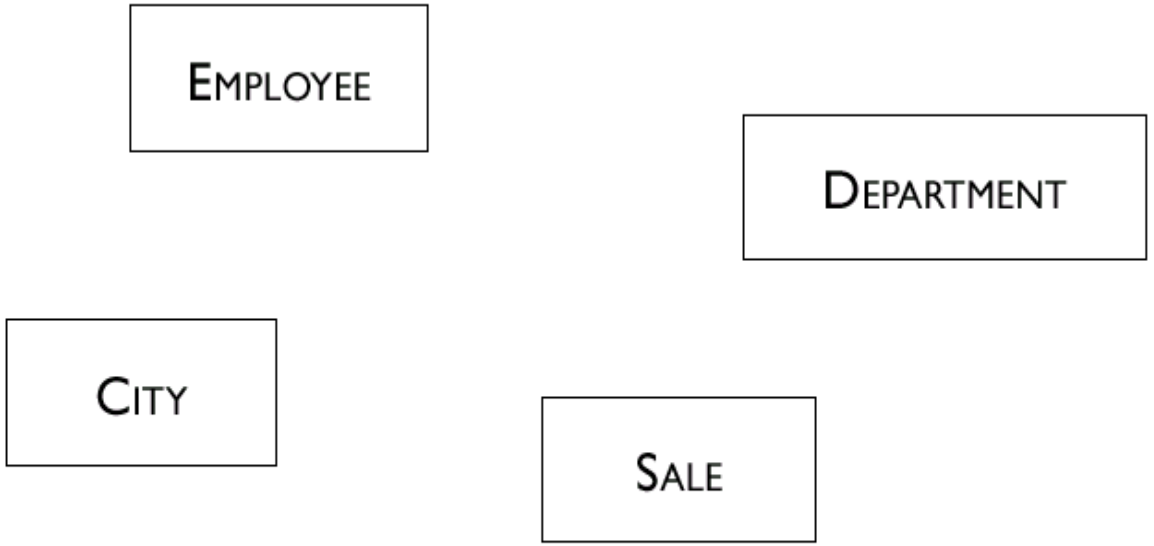
Entity/Relationship Model

- Visual data model (diagram-based)
  - Quickly "chart out" a database design
  - Easier to "see" big picture
  - Comparable to class diagrams in UML
- Basic concept: *entities* and their *relationships*, along with the *attributes* describing them



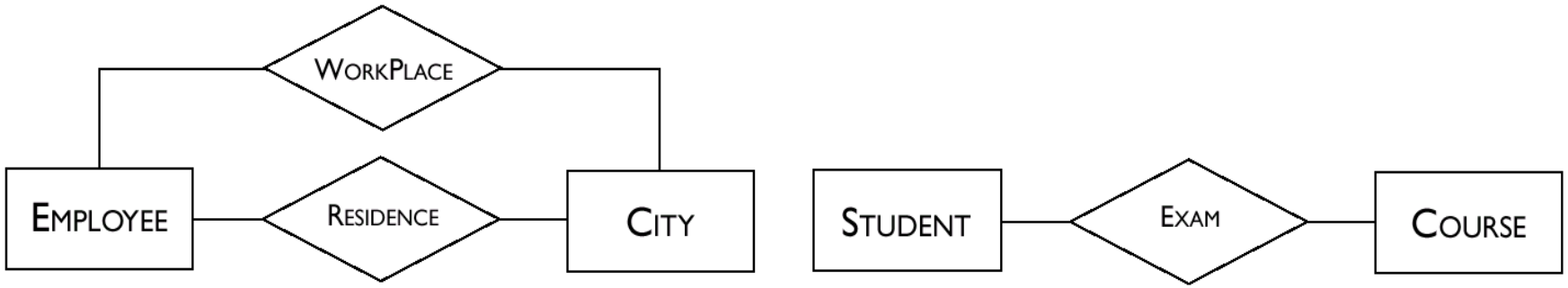
Entity Sets

- 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)

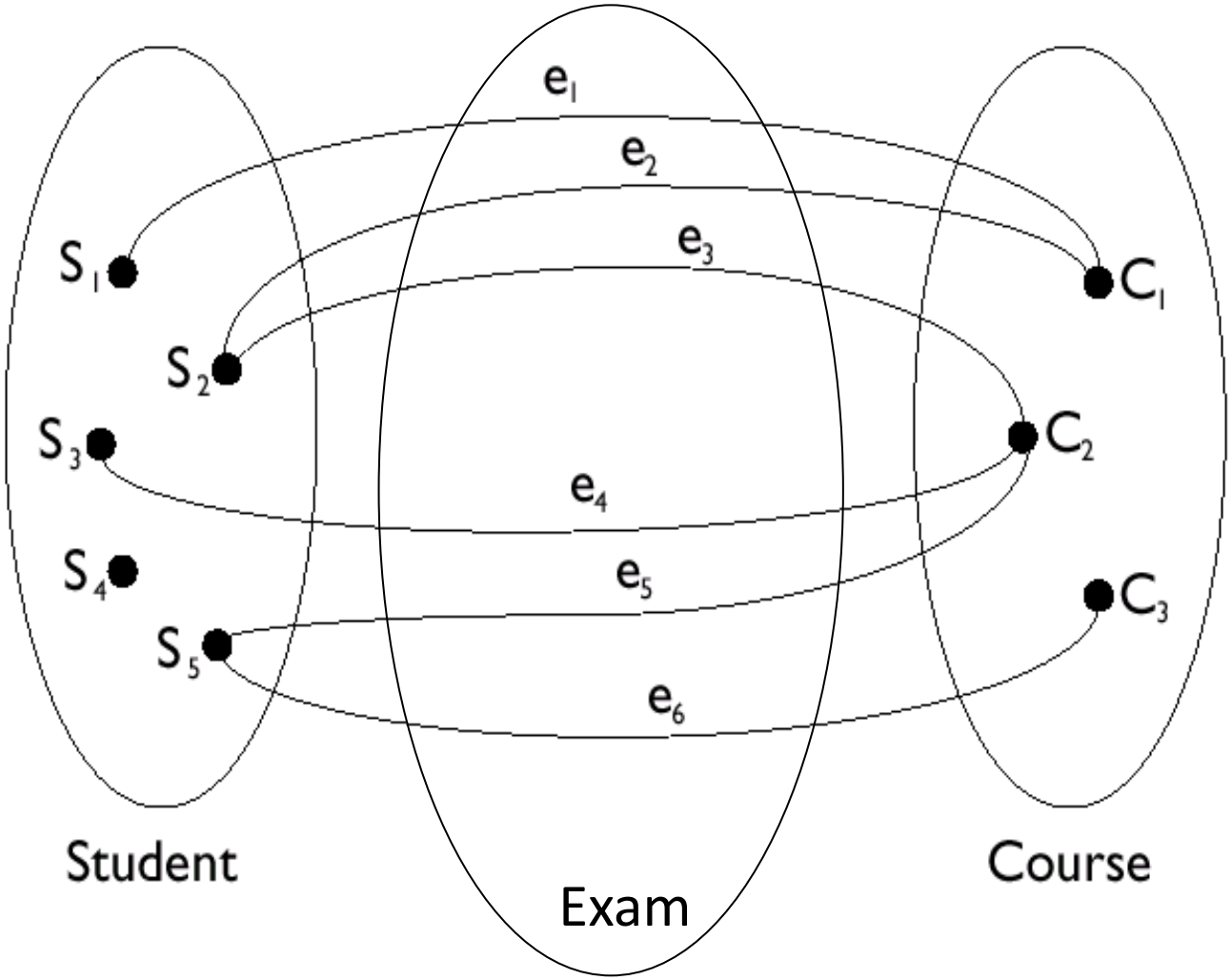


Relationship Sets

- 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)



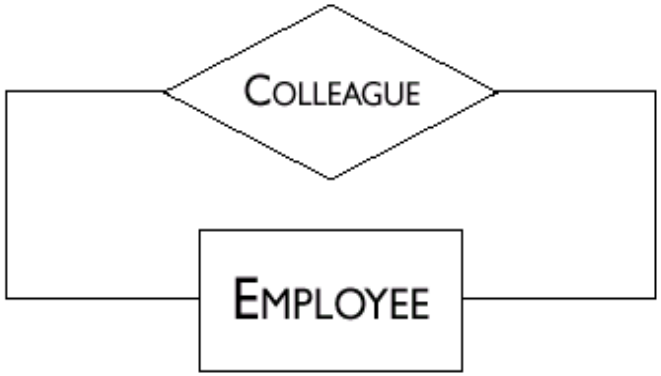
Example of Instances for Exam

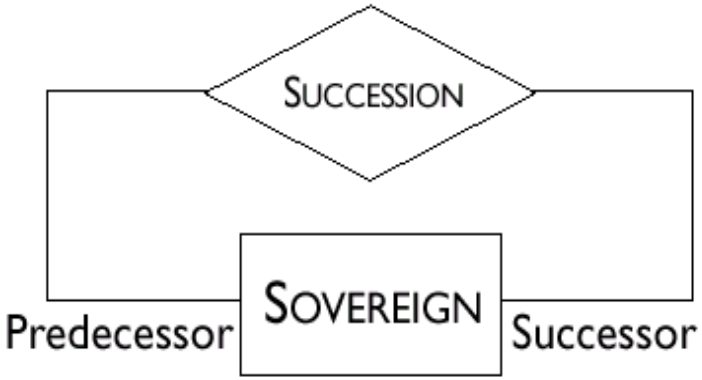


A student can't take more than one exam for a particular course

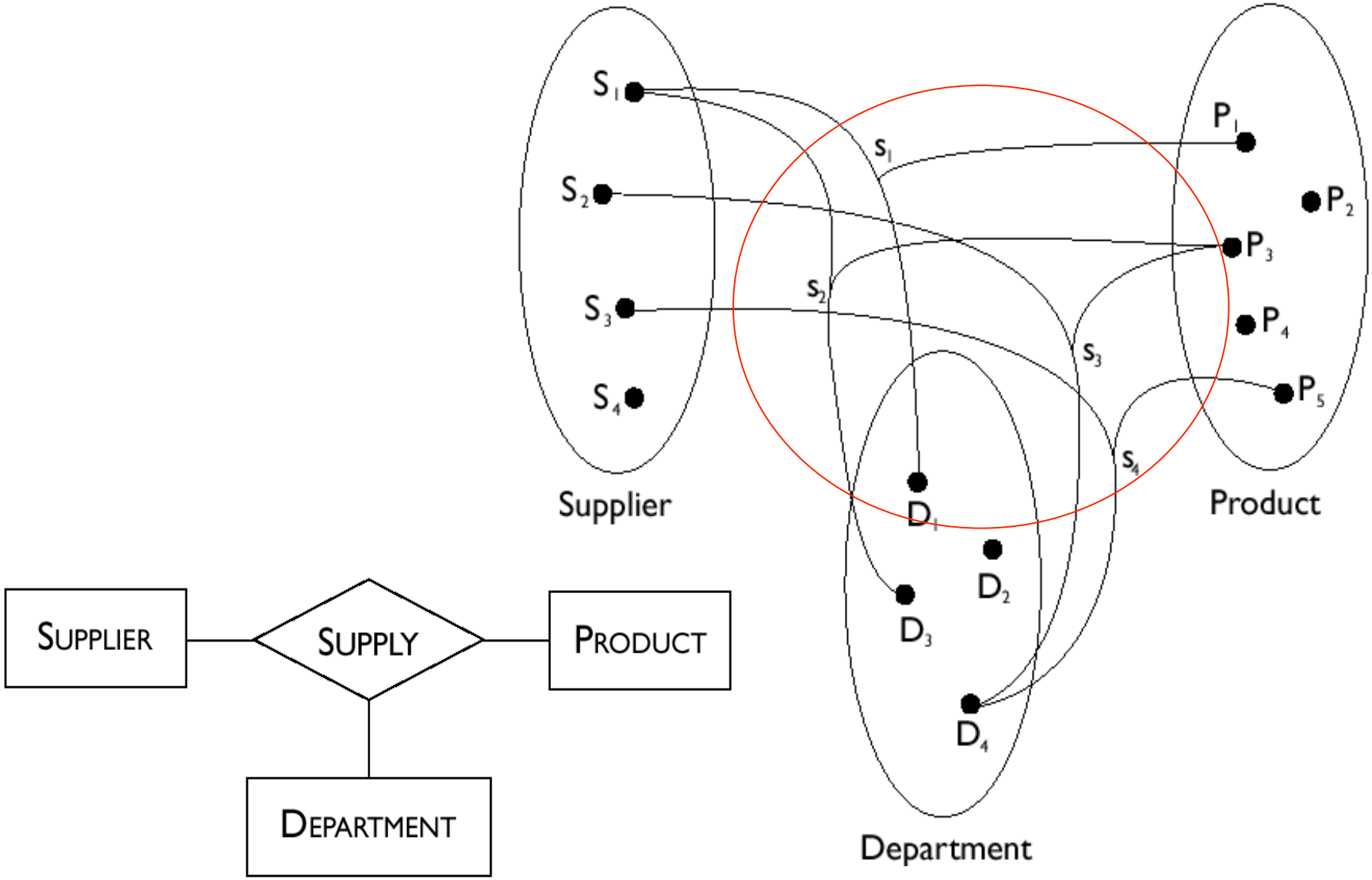
Recursive Relationships

- Recursive relationships relate an entity set to itself
- Note in the second example that the relationship is not symmetric
  - In this case, it is necessary to indicate the two roles that the entity plays in the relationship



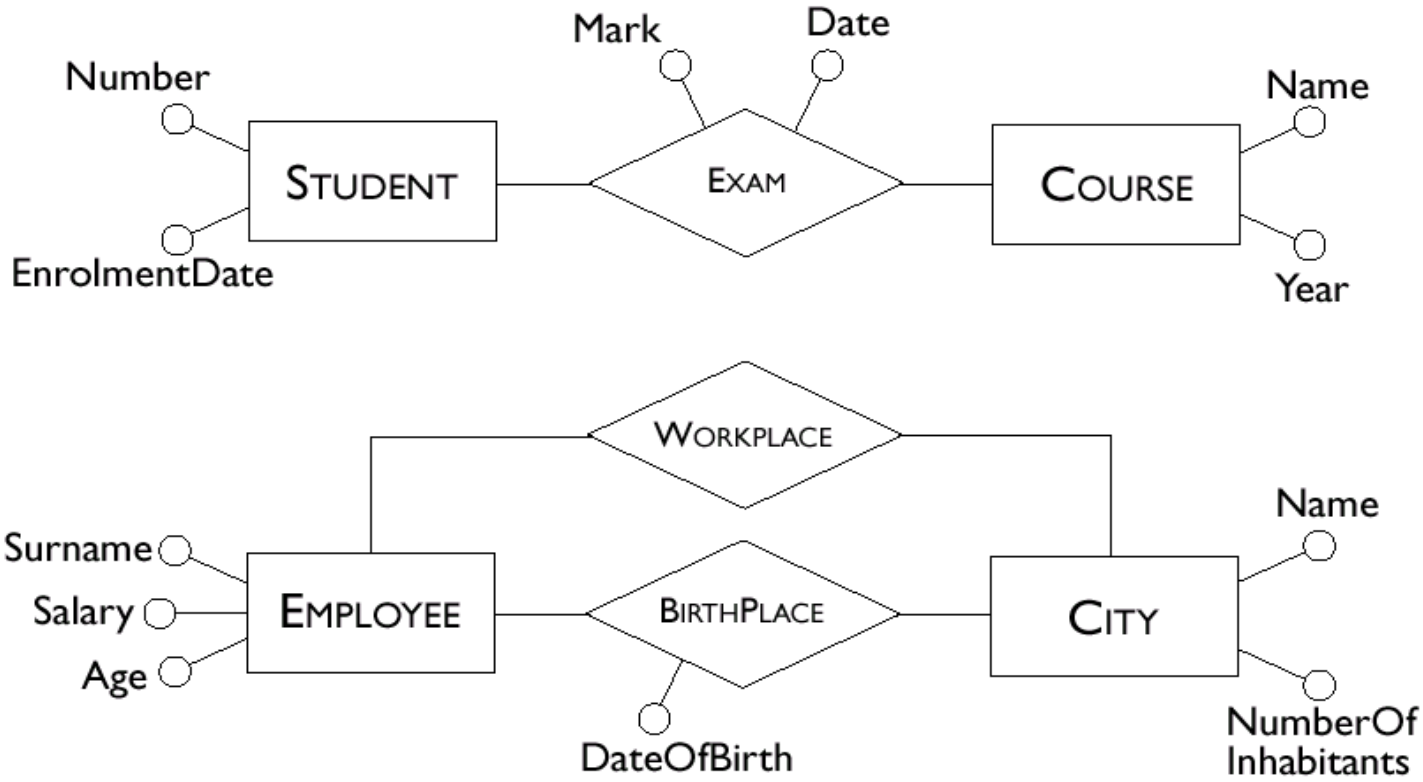


Ternary Relationships



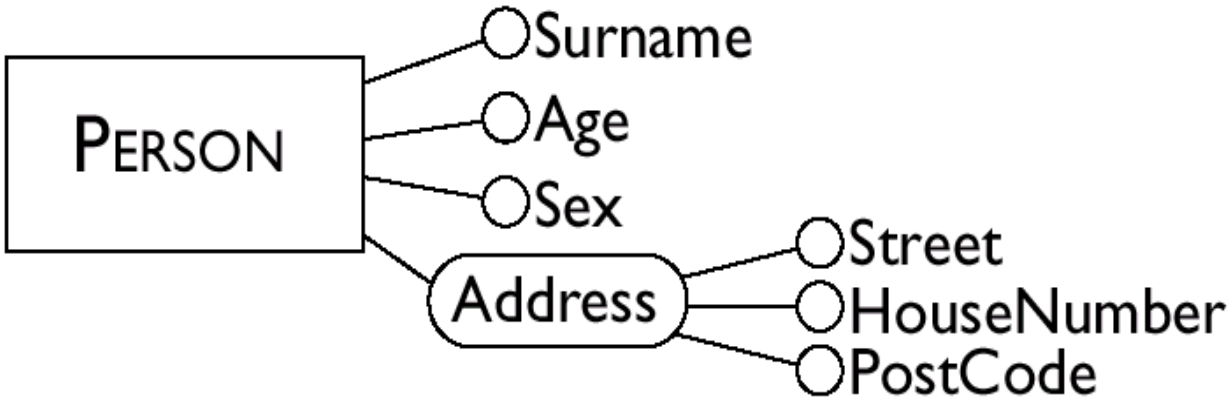
Attributes

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

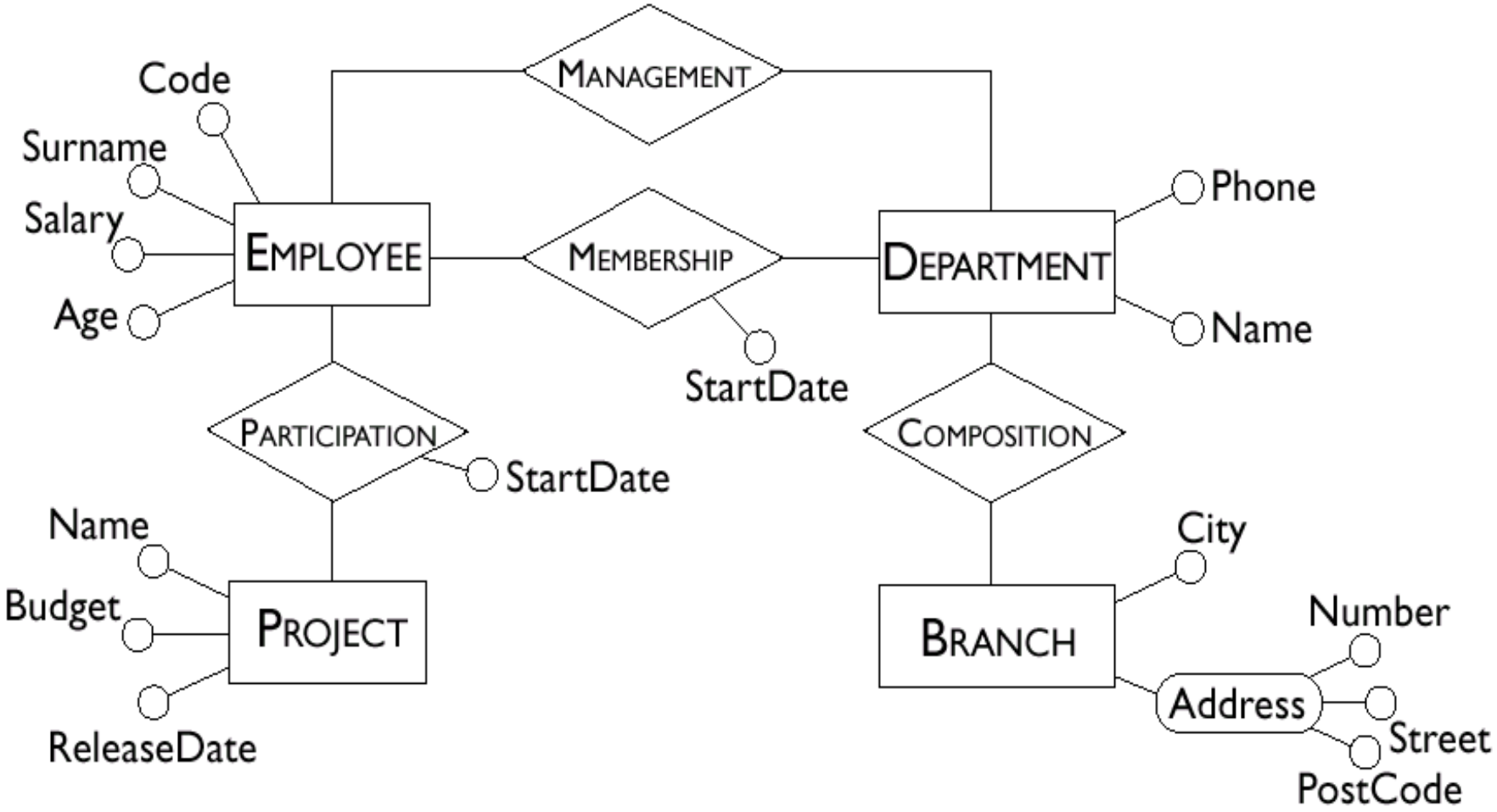


Composite Attributes

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



Example Schema with Attributes



Cardinalities

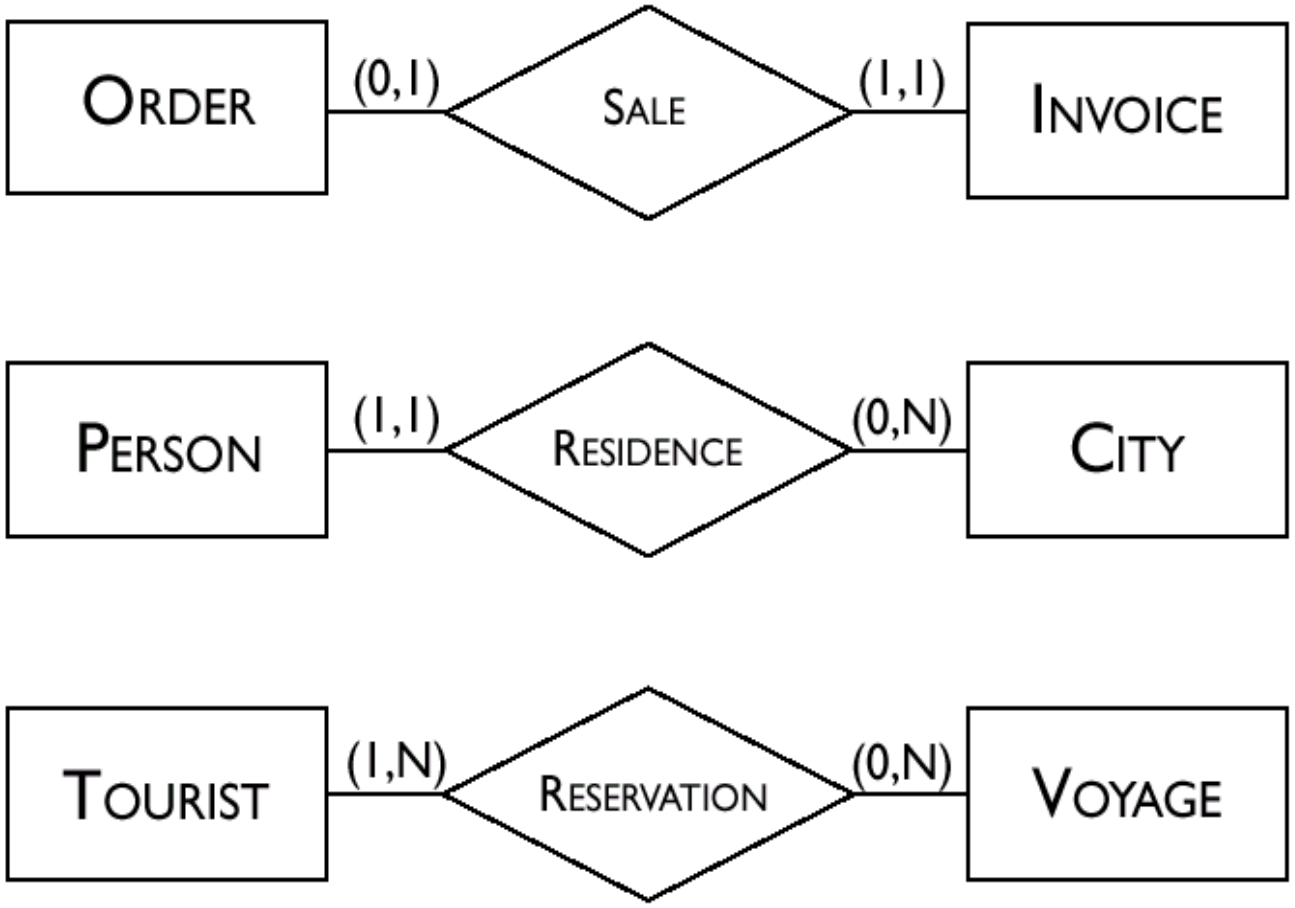
- Each entity set participates in a relationship set with a minimum (min) and a maximum (max) cardinality
- Cardinalities constrain how entity instances participate in relationship instances
- Graphical representation in E/R Diagrams: pairs of (min, max) values for each entity set

An entity might not participate in any relationship

Cardinalities (cont.)

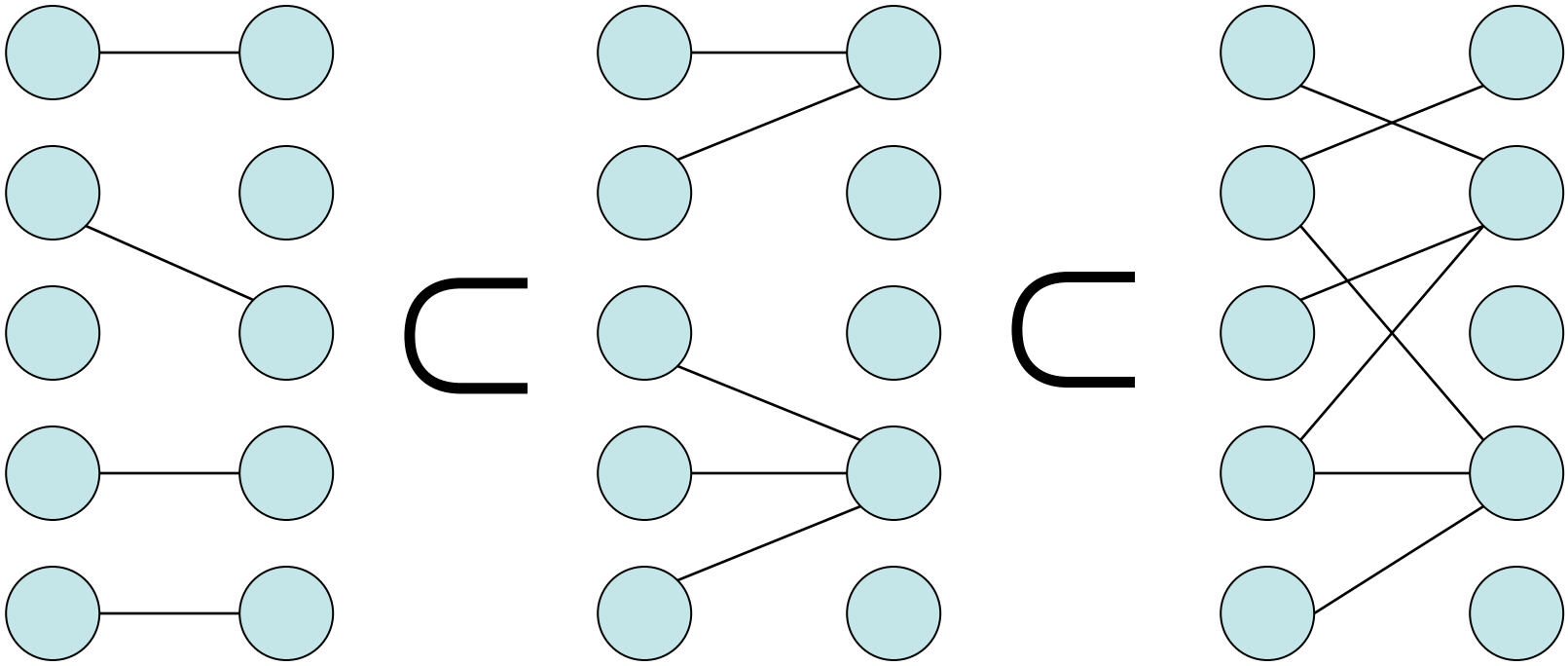
- In principle, cardinalities are pairs of non-negative integers (n, N) such that n ≤ N, where N means "any number"
- minimum cardinality n:
  - If 0, entity participation in a relationship is optional
  - If 1, entity participation in a relationship is mandatory
- maximum cardinality N:
  - If 1, each instance of the entity is associated at most with a single instance of the relationship
  - If N, then each instance of the entity is associated with many instances of the relationship

Cardinality Examples



Multiplicity of relationships

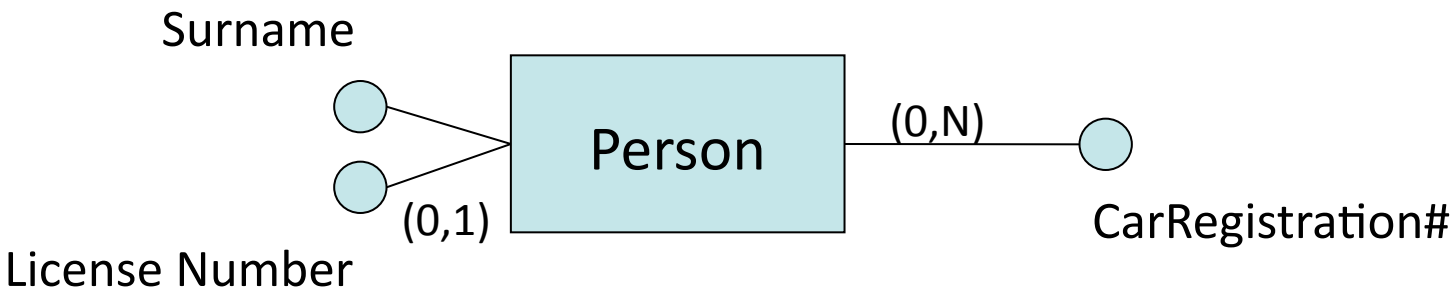
If entities E1 and E2 participate in relationship R with cardinalities (n1, N1) and (n2, N2) then the multiplicity of R is N1-to-N2 (which is the same as saying N2-to-N1)



1-to-1 ⊂ N-to-1 OR 1-to-N ⊂ N-to-N

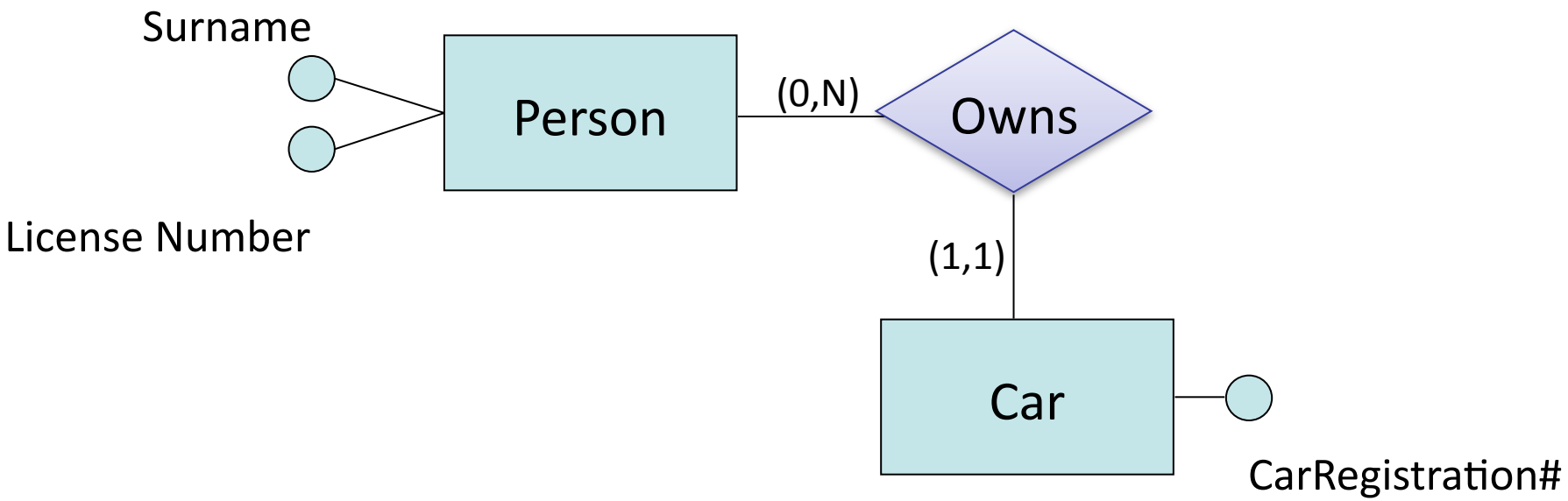
Cardinalities of Attributes

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



Cardinalities of Attributes (cont.)

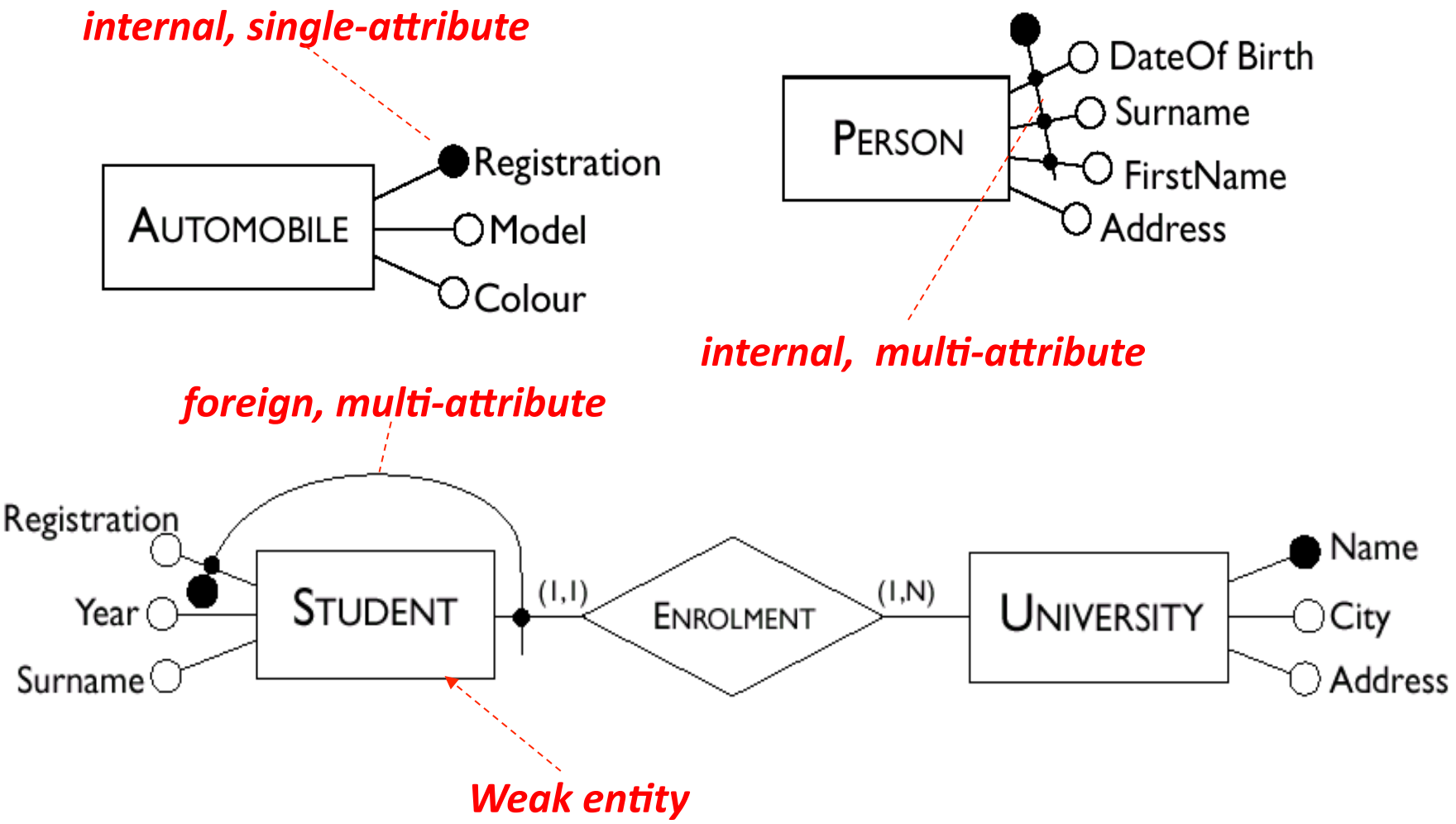
- Multi-valued attributes often represent situations that can be modeled with additional entities. E.g., the ER schema of the previous slide can be revised into:



Keys in E/R

- Keys consist of minimal sets of attributes which uniquely identify instances of an entity set
  - socialInsurance# may be a key for Person
  - firstName, middleName, lastName, address may be a key for Person
- In most cases, a key is formed by one or more attributes of the entity itself (*internal* keys)
- Sometimes, an entity doesn't have a key among its attributes. This is called a *weak entity*.
  Solution: the keys of related entities brought in to help with identification (becoming *foreign keys*).
- A key for a relationship consists of the keys of the entities it relates

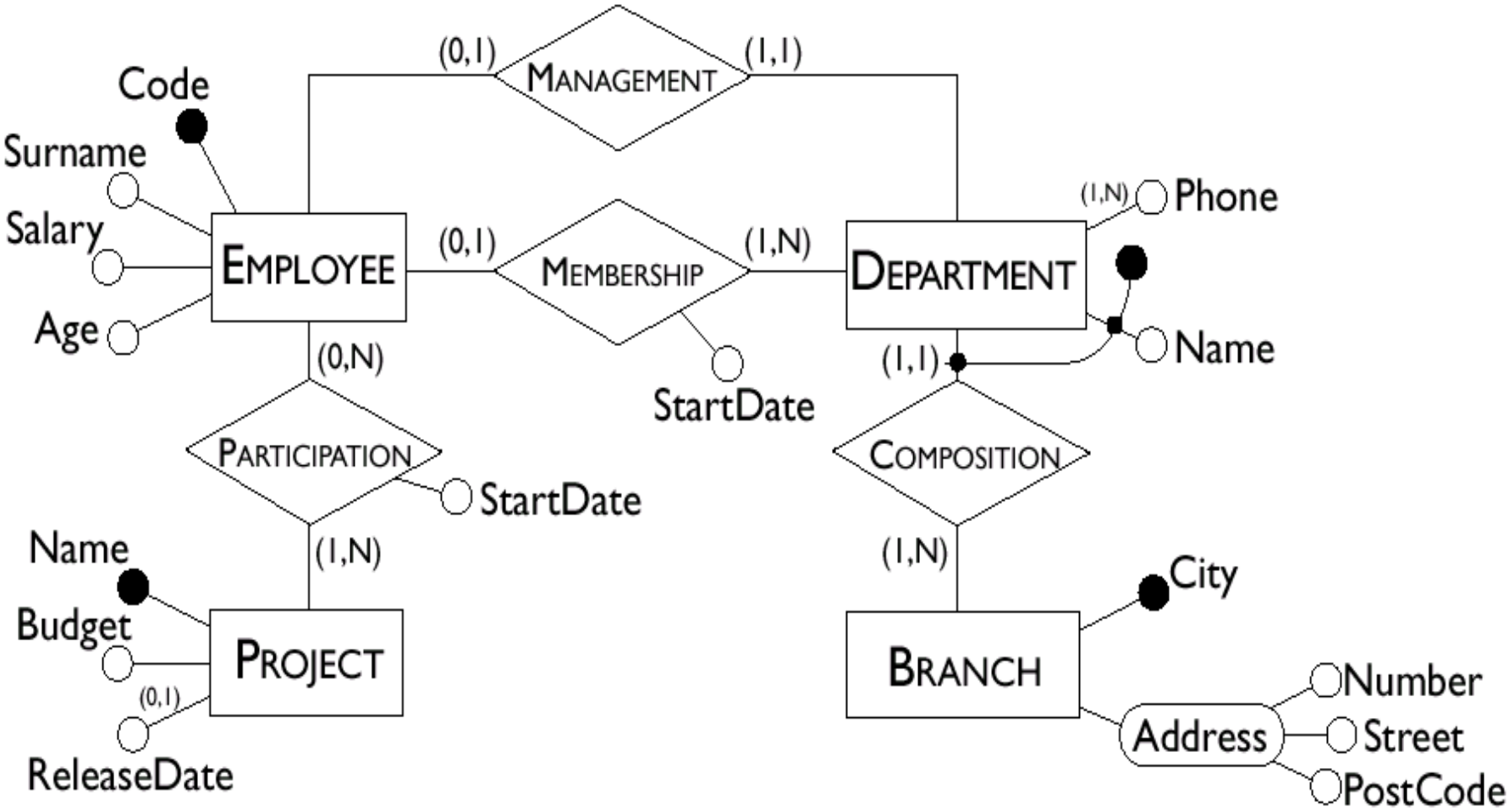
Examples of Keys in E/R



General Observations about Keys

- A key may consist of one or more attributes, provided that each of the attributes has (1,1) cardinality
- A foreign key can involve one or more entities, provided that each of them is member of a relationship to which the entity to be identified participates in the relationship with cardinality equal to (1,1)
- 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

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 ER 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