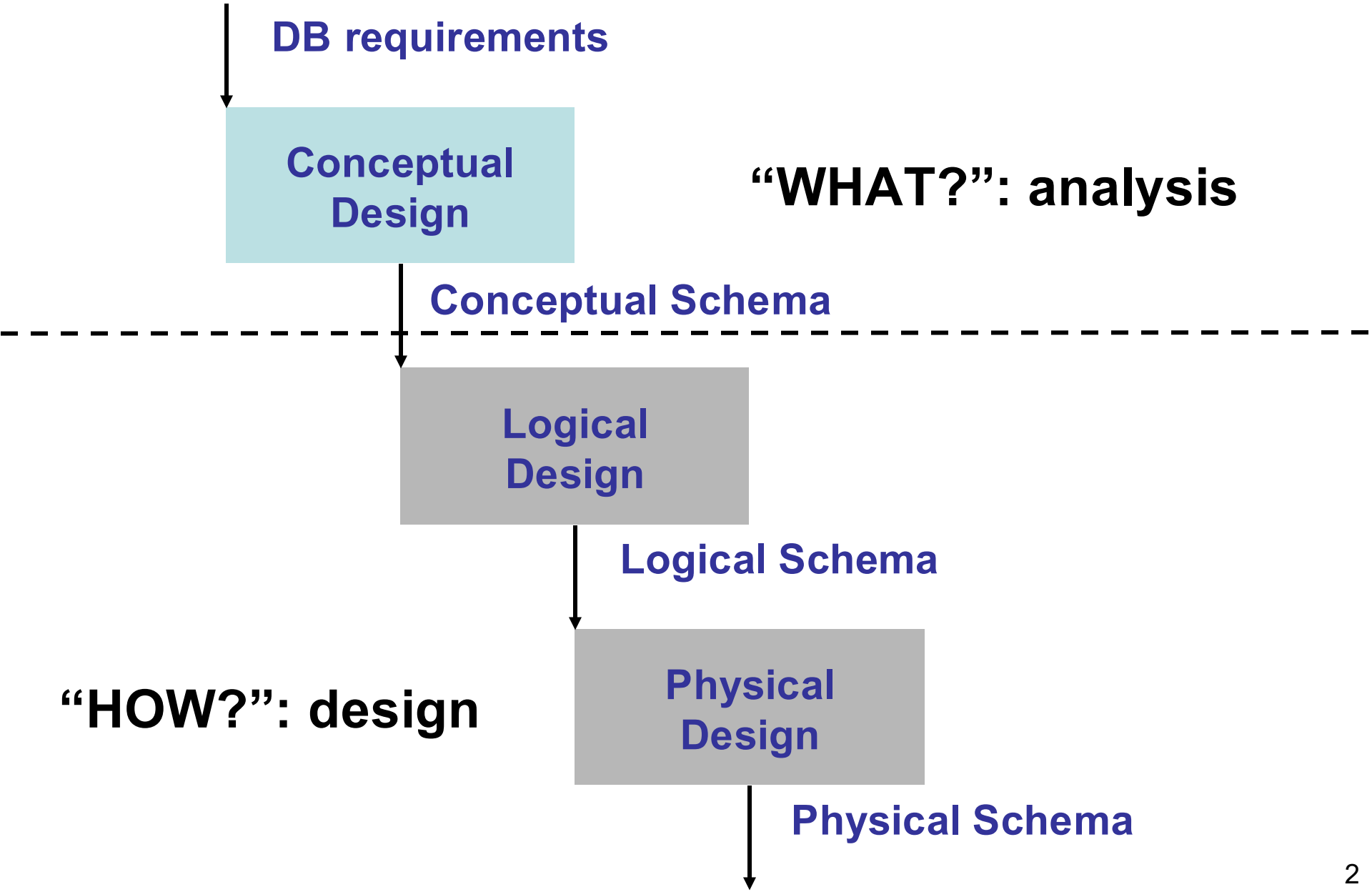
Databases

Conceptual Design

Danilo Montesi

danilo.montesi@unibo.it

Requirements & Design



Requirements & Conceptual Data Modelling

- Covers the following (interconnected) tasks:
  - Requirements' elicitation
  - Requirements' analysis
  - Building the conceptual schema
  - Building the glossary

Requirements

- Possible sources:
  - Users and customers, through:
    - interview
    - specific documentation (*ad hoc*)
  - (Already existing) documentation:
    - regulations (laws, industry regulations)
    - internal regulations, business processes
    - pre-existing solutions
  - Forms

Requirements' Acquisition and Analysis

- Acquisition of requirements is a difficult and non-standardized activity
- It starts by collecting the first requirements. Such requirements often need to be refined (through other acquisitions)

Acquisition by Interviews

- Different informations are provided by different kind of users
- Users belonging to a higher level in the management structure often have a wider but less detailed view
- Interviews often leads to a requirement acquisition by further refinements (e.g., through other customer interviews)

Interacting with Users

- Tips:
  - Comprehension (and consistency) checks should be carried out frequently
  - Use cases examples are very useful (e.g., generic and limit cases)
  - Ask for definitions and classifications
  - Ask to remark which aspects are essential and which are peripheral

Requirements: Descriptive Documentation

- Rules of thumb:
  - Choose the right abstraction level
  - Keep sentences' structure as standard as possible
  - Split the most verbose sentences
  - Distinguish "data" sentences from "function" sentences

Requirements: Terms and Concepts

- General rules:
  - build a glossary for the terms
  - homonyms and synonyms should be unified with a single term
  - Clarify explicitly the relations between terms
  - Sort the sentences by concepts

Requirements: an Example (a)

Bibliography Database:

- We want to automate the data for bibliographical references
- We want to handle the data of interest concerning the bibliographical references. For each reference we want an ID code formed by 7 characters, formed by the authors' initial letters, the year of publication, and a disambiguation character

Requirements: an Example (b)

Bibliography Database:

- A reference could be either a *monograph* (we are interested on editor, date and place of editing) or *journal papers* (paper name, #volume, number, pages and year of publication); for both papers we are interested on the authors' names

Requirements: another Example (a)

Training company:

- A “training company” requires a database providing some courses, for which we want to handle both lectures and teachers
- There are ~5000 students, which have an ID, their tax code (Tax), a surname, age, sex, place of birth, their employers’ name, where they have worked previously and when, address and mobile phone, the attended courses (~200) and the final grade

Requirements: another Example (b)

Training company:

- We want to store the workshops attended by the students and, for each day of the year, where and when the lectures are hold
- The courses have a code, a title and could have different editions, have a beginning and an end date and have a number of participants
- If a student is *freelancer*, we store the area of interest and, if they have it, an honorific

Requirements: another Example (c)

Training company:

- Regarding the teachers (~300), we store their surname, their age, the place of birth, the name of the taught course, the set of the courses that they taught in the past and the set of the courses that they could teach in the future. We want to keep all their phone records. Teachers could be either house employees or independent contractors

Glossary

Term	Description	Synonym	Related To
Participant	*Who takes part to the courses*	Student	Course, Society
Lecturer	*The courses' lecturer. He could be a house employee*	Teacher	Course
Course	*Internal course. It could have several editions.*	Workshop	Lecturer
Company	*Current (or past) place of work*	Place	Participant

Structuring Requirements in Homogeneous Groups of Sentences

Structuring: an Example (d)

Training company (general sentences):

- A “training company” requires a database providing some courses, for which we want to handle both lectures and teachers data

Structuring: an Example (e)

Training company (participants' sentences):

- There are ~5000 **participants**, which have an ID, their tax code (Tax), a surname, age, sex, place of birth, their employers' name, where they have worked previously and when (**starting and end date**), address and mobile phone, the attended courses (**current and past courses**) with their final grade

Structuring: an Example (f)

Training company (specific participants' sentences):

- If a student is *freelancer*, we store the area of interest and, if they have it, an honorific
- **If a student belongs to the same organization, we want to know their level within the management structure and their working position**

Structuring: an Example (g)

Training company (employer's statement):

- We store the participants' employers, both current and from the past. In particular we represent their name, address and phone number

Structuring: an Example (h)

Training company (courses' statement):

- The courses (~200) have a code, a title and could have different editions, have a beginning and an end date. For each edition we keep the number of participants, the day of the week, the rooms and when the courses are held

Structuring: an Example (i)

Training company (lecturers' statement):

- Regarding the **lecturers** (~300), we store their surname, their age, the place of birth, the name of the taught course, the set of the courses that they taught in the past and the set of the courses that they could teach in the future. We want to keep all their phone records. **Lecturers** could be either house employees or independent contractors

Requirements and Conceptual Schemas 1/2

- Which ER constructor shall we use for each specific “term” within the requirements?
  - Let’s look back at the E-R constructors definitions

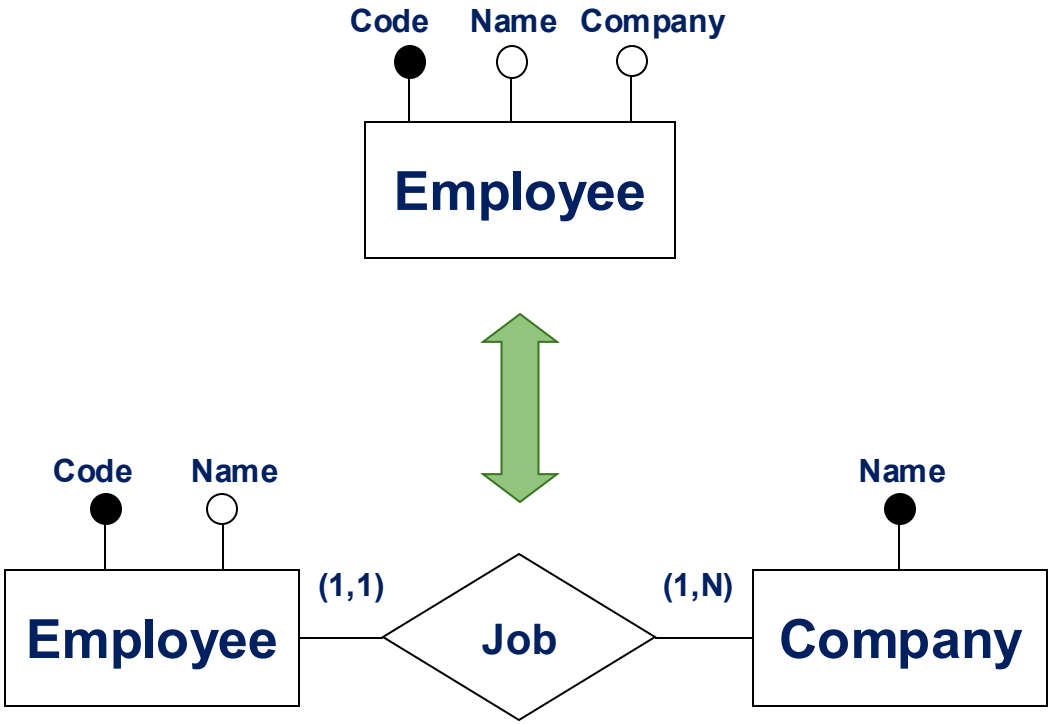
Requirements and Conceptual Schemas 2/2

- **Entity**: if the term has relevant properties and describes stand-alone objects
- **Attribute**: if it is a simple term with no further specifications
- **Relationship**: when a term relates to other terms
- **Generalization**: one terms is a more general case of another

Design Pattern

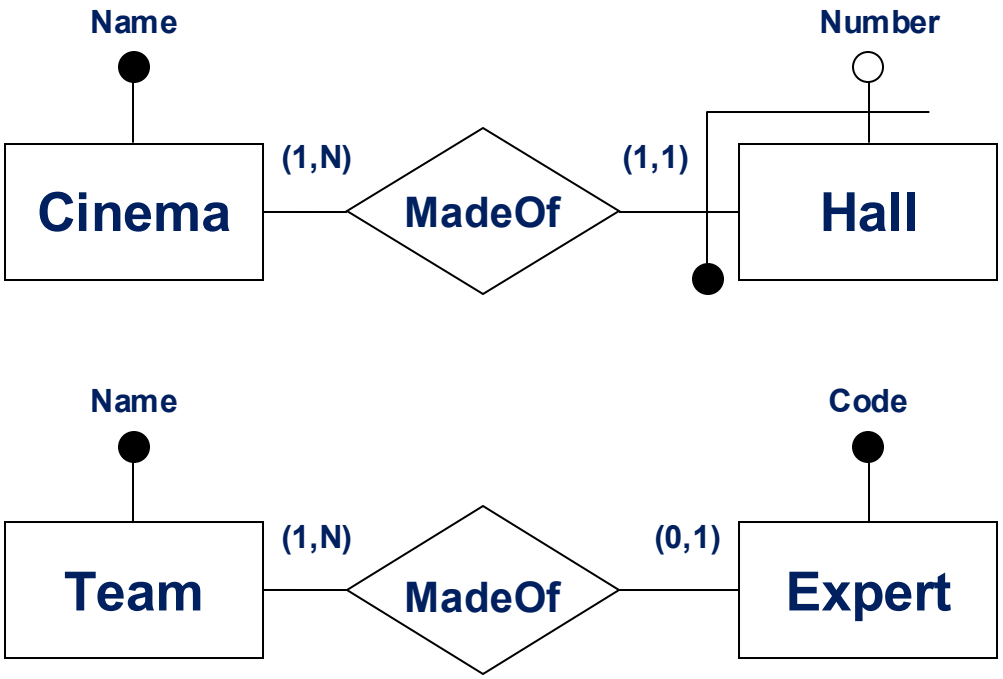
- Common best practices in software design for common problems
- Software Engineers use them on a daily basis
- Let's take a look on some ER design patterns

Attributes' Reification (Entities)



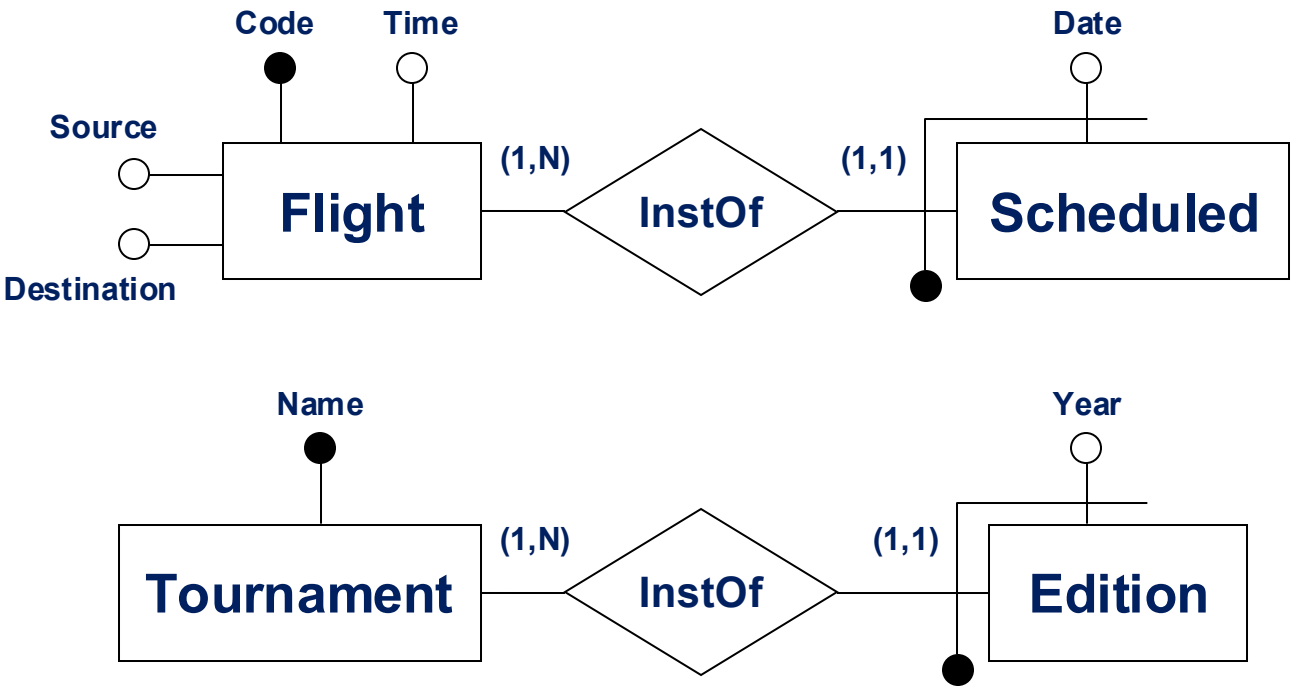
Company is (usually) a different concept from the employee (e.g., when companies are involved in other relations)

Part-of



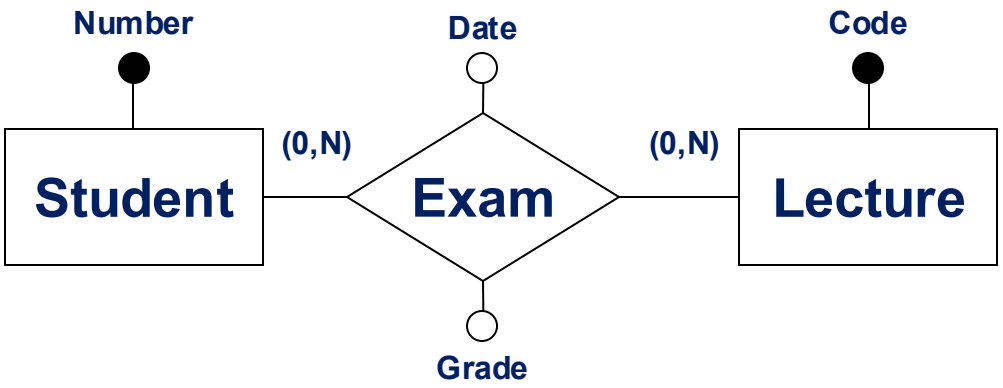
(1,N) relations could represent **part-of relations**. They could represent **compositions** (a cinema is made of halls, each hall belongs to a cinema) or **aggregations** (a team is made of experts)

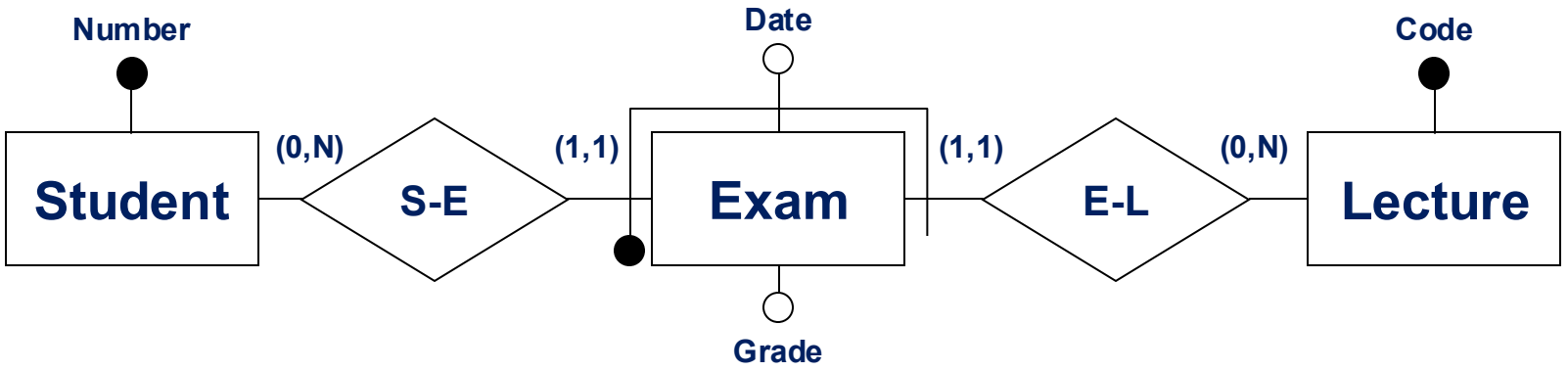
Instance-of

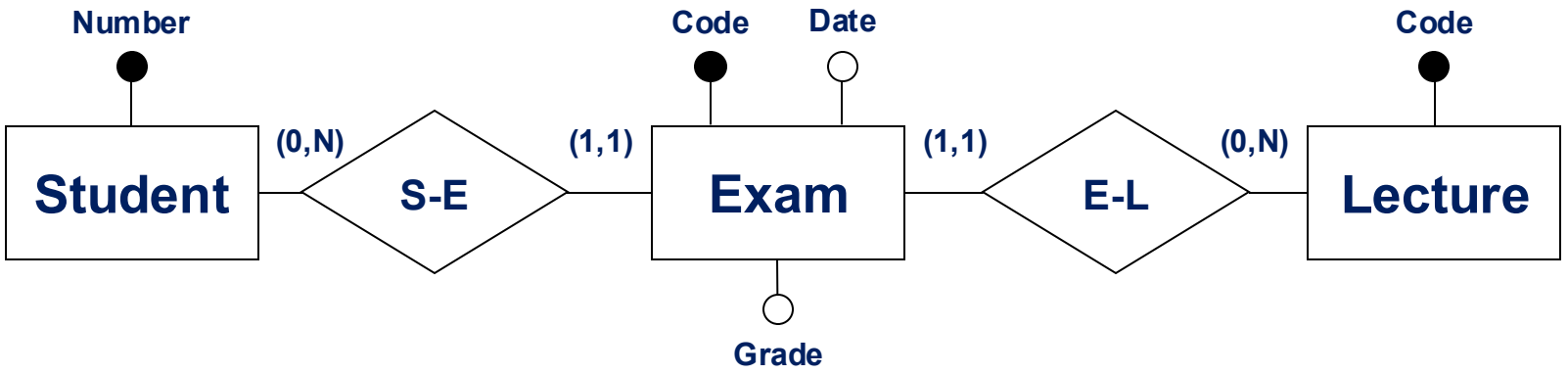


Sometimes two distinct entities are needed, one involving an **abstract representation**, and the other one storing the **pieces of informations** needed by our requirements

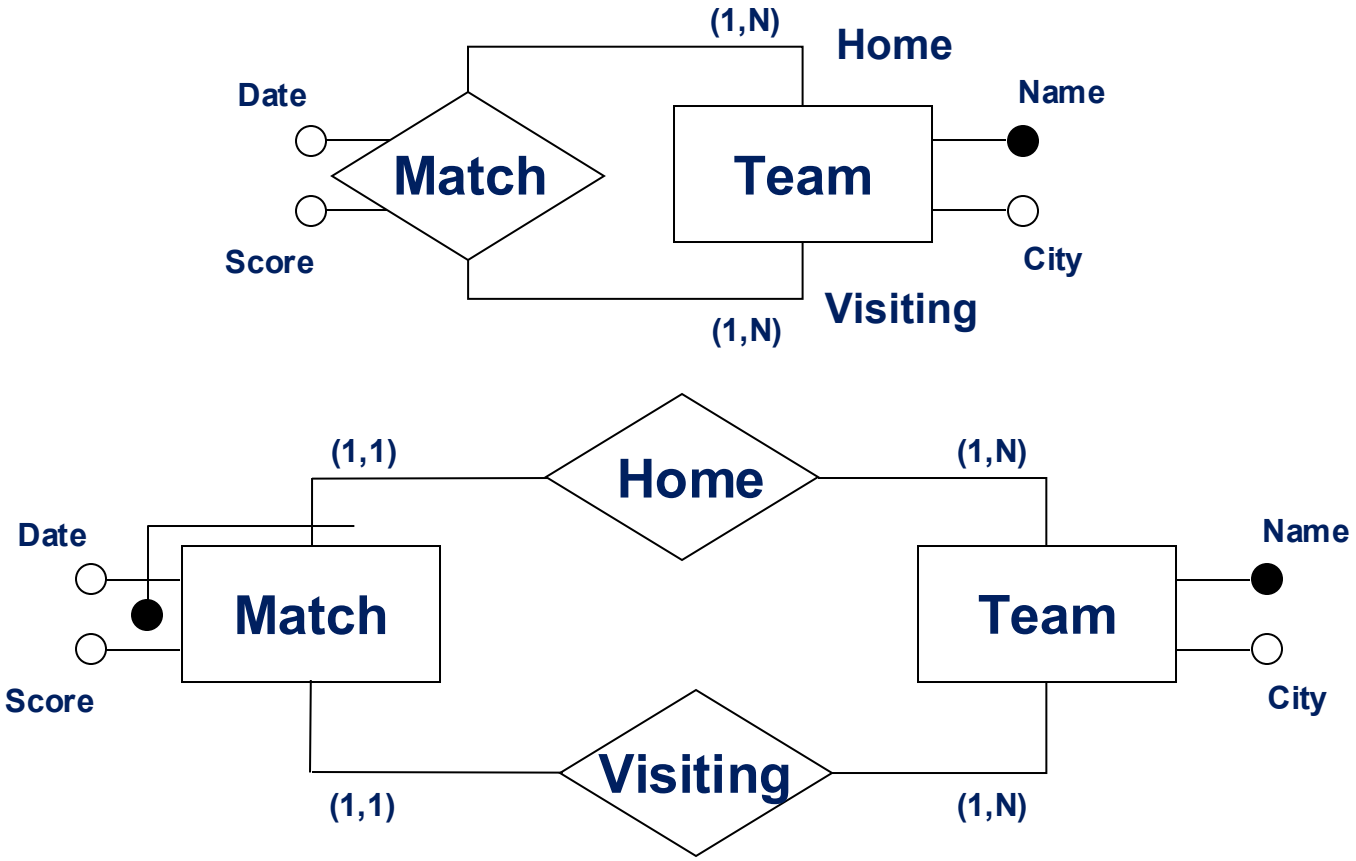
Reification: Binary Relations





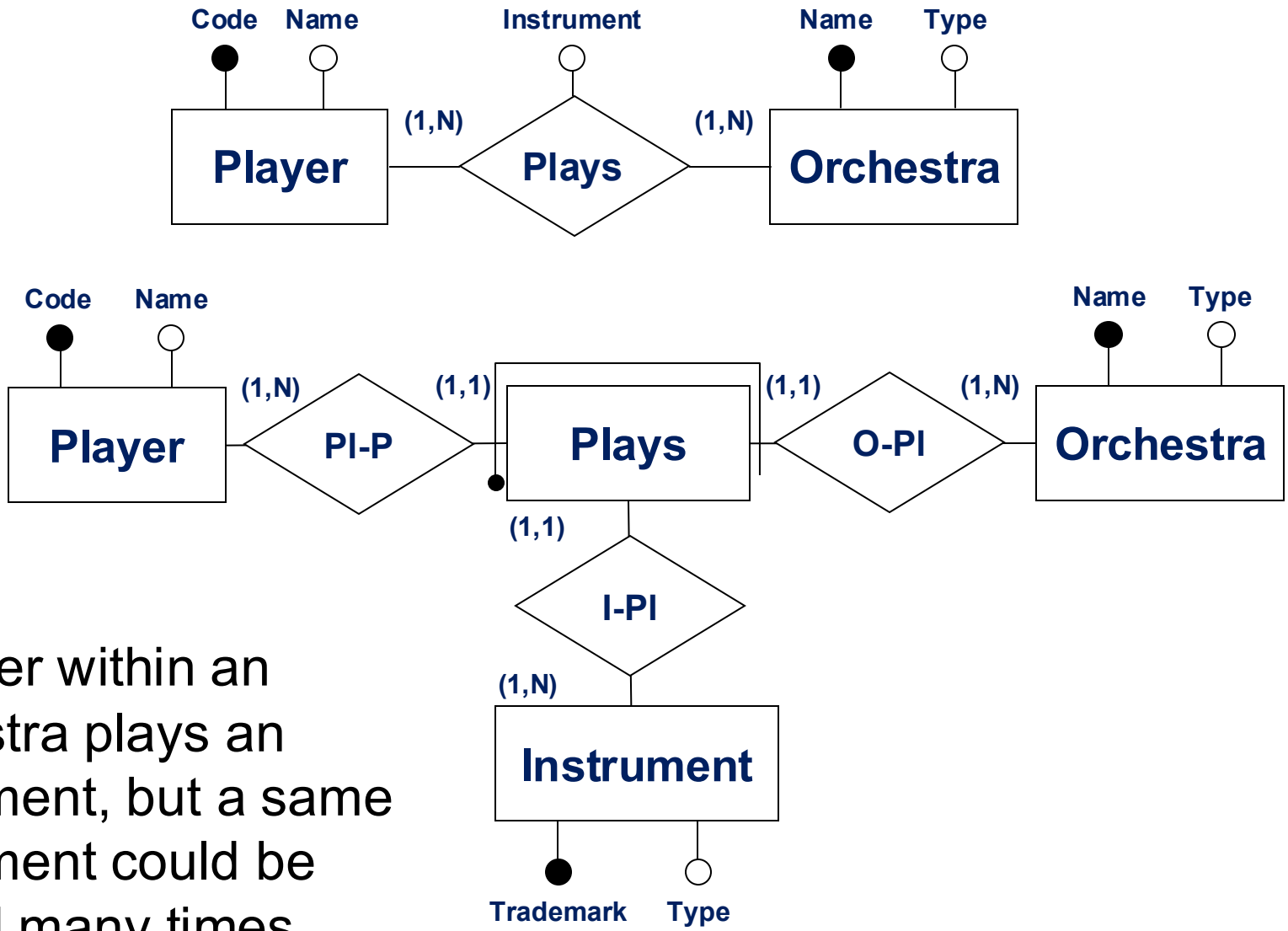


Reification: Recursive Relation



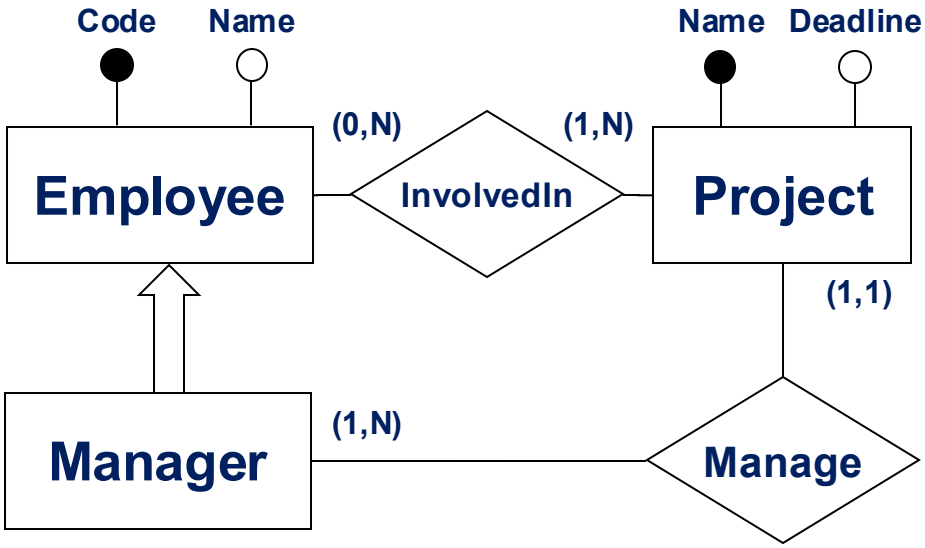
A match could be expressed as a binary relation between two teams. Since any team could play more times a match, we could reify a match as in the given ER schema (hence providing the closest representation)

Attributes' Reification (Relations)



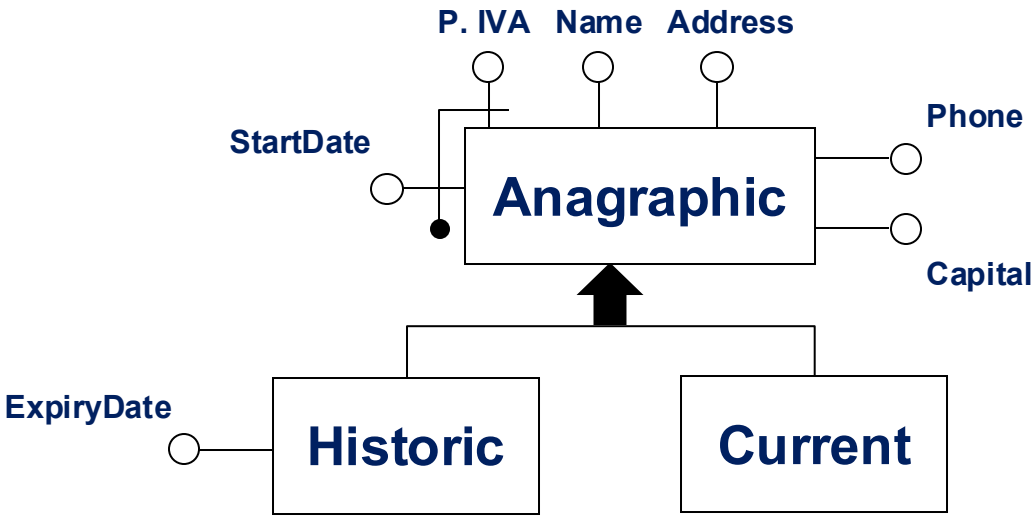
A player within an orchestra plays an instrument, but a same instrument could be played many times

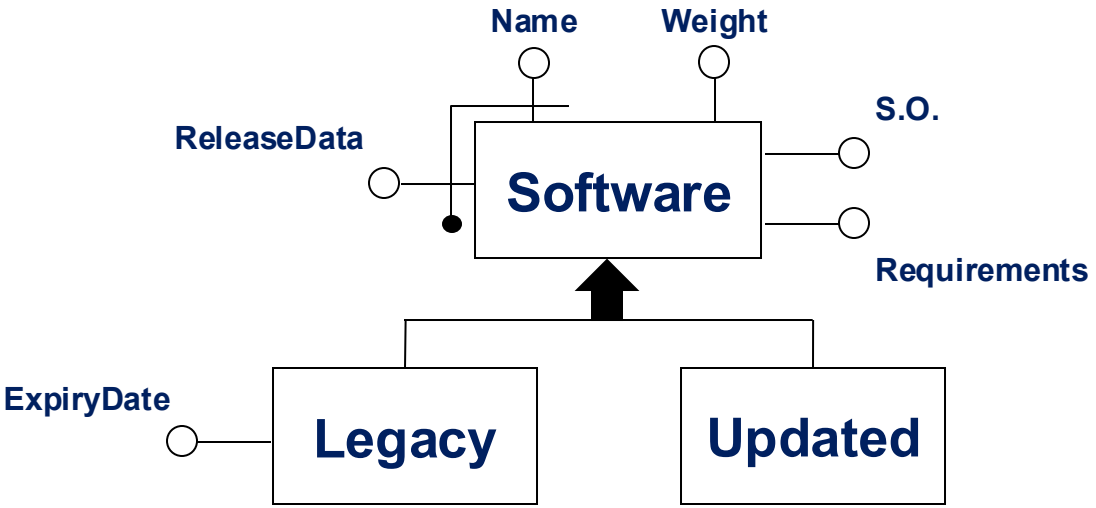
Specific Case



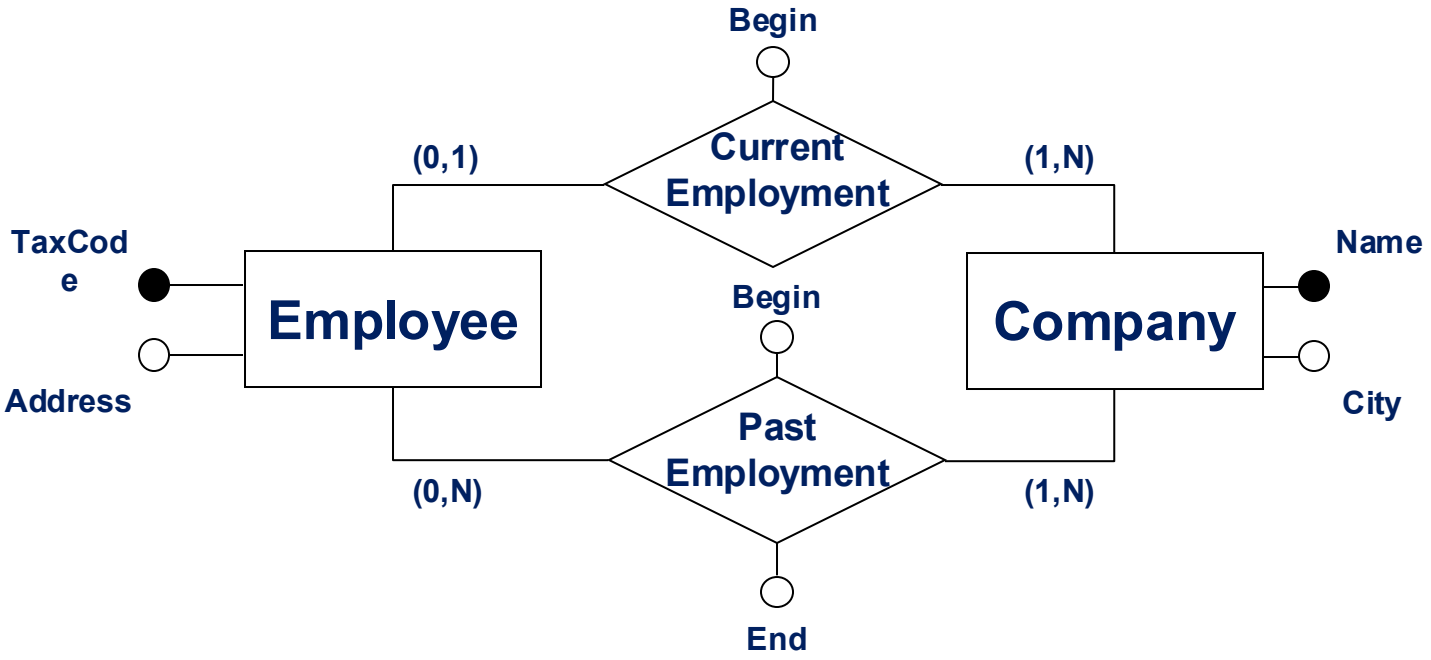
Generalizations are used to define a specific case (Manager) of a given entity (Employee). In this case not all the employees manage a project

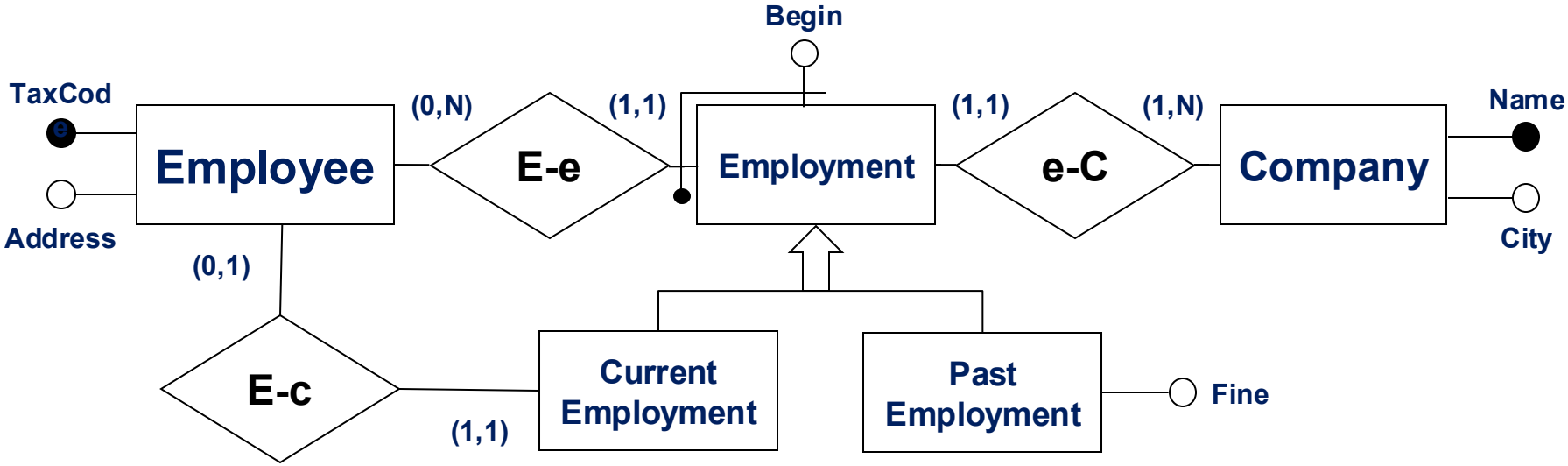
Historicizing of a Concept



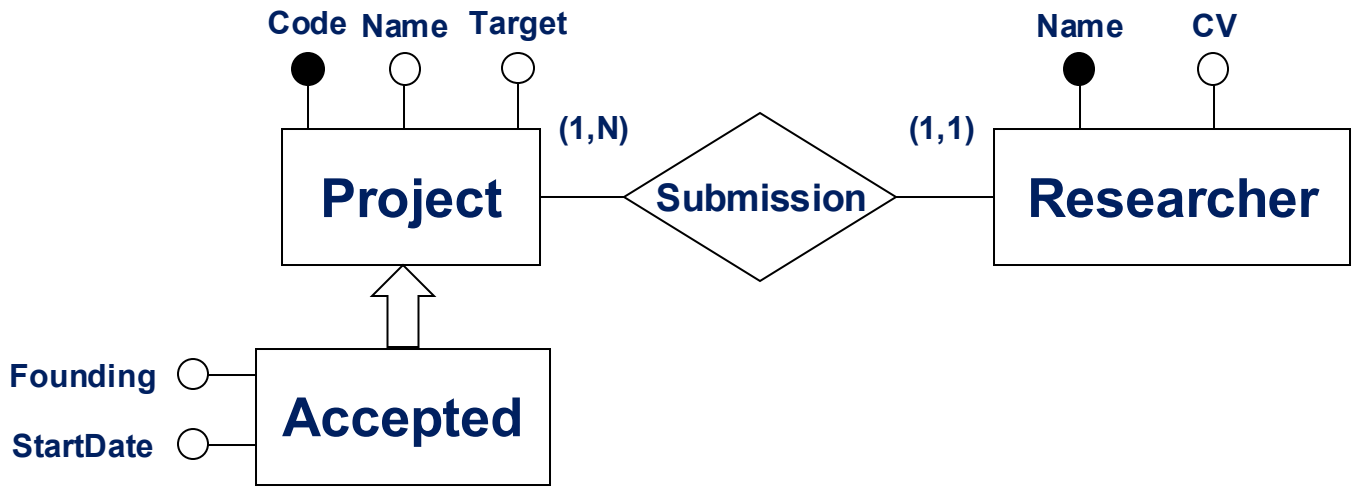


Historicizing: Further Examples



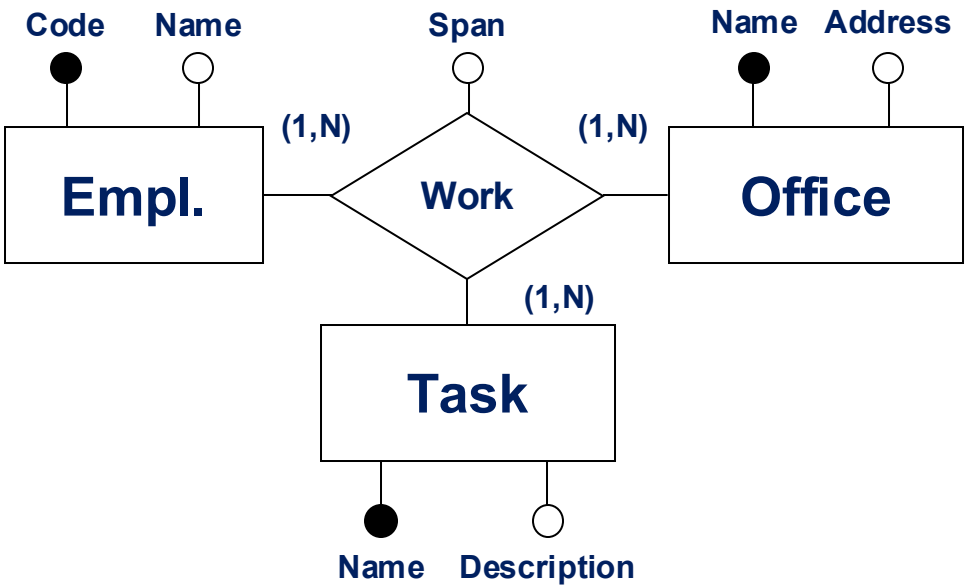


Extending a Concept



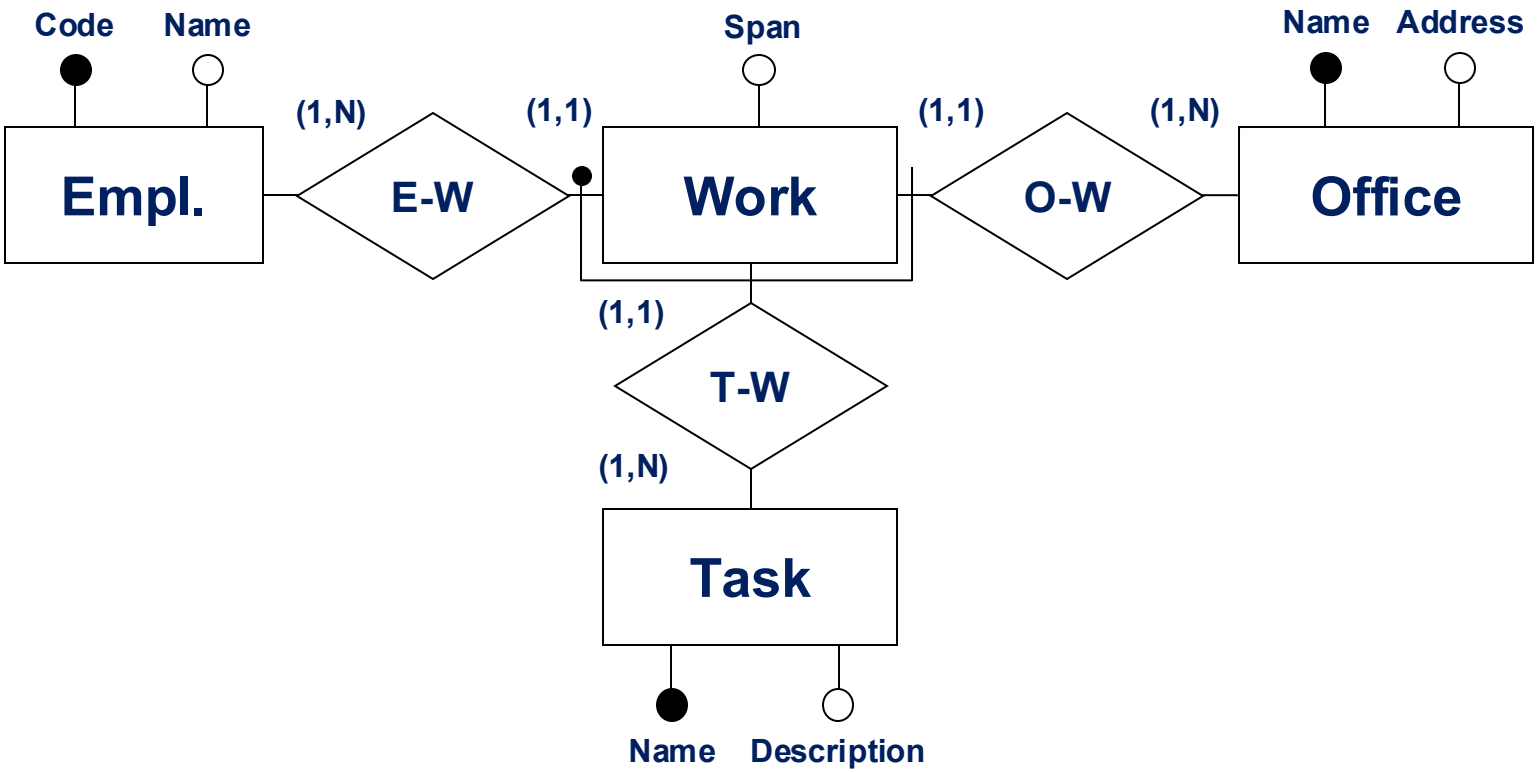
Generalizations can be used to extend current implementations when the project is already in operation. In this case the accepted project requires more information than other projects (either idle or rejected projects)

Ternary Relationship



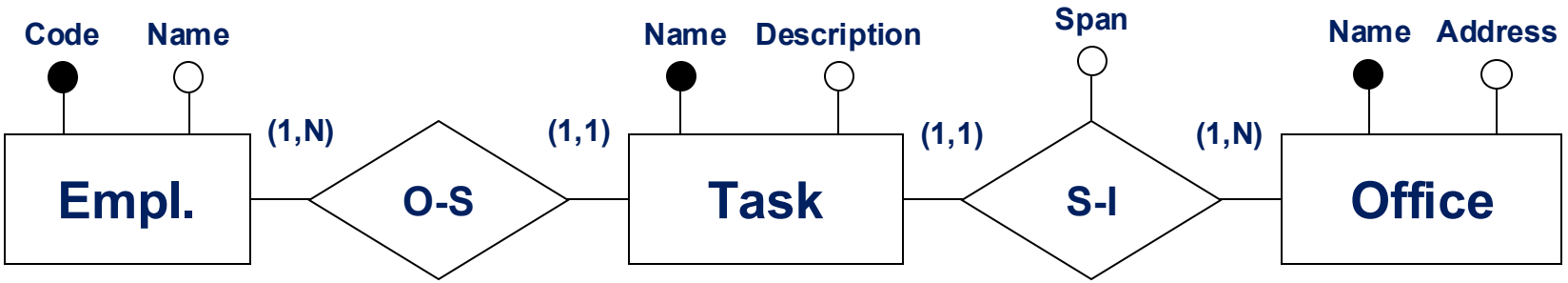
- Employees can work on several tasks in different offices
- Offices host different employees working on several tasks
- Tasks could be performed by different employees and in different offices

Reification: Ternary Relationship (1)



Each Work is defined by an Employee (E-W) working in a given Office (O-W) for a given Task (T-W)

Reification: Ternary Relationship (2)

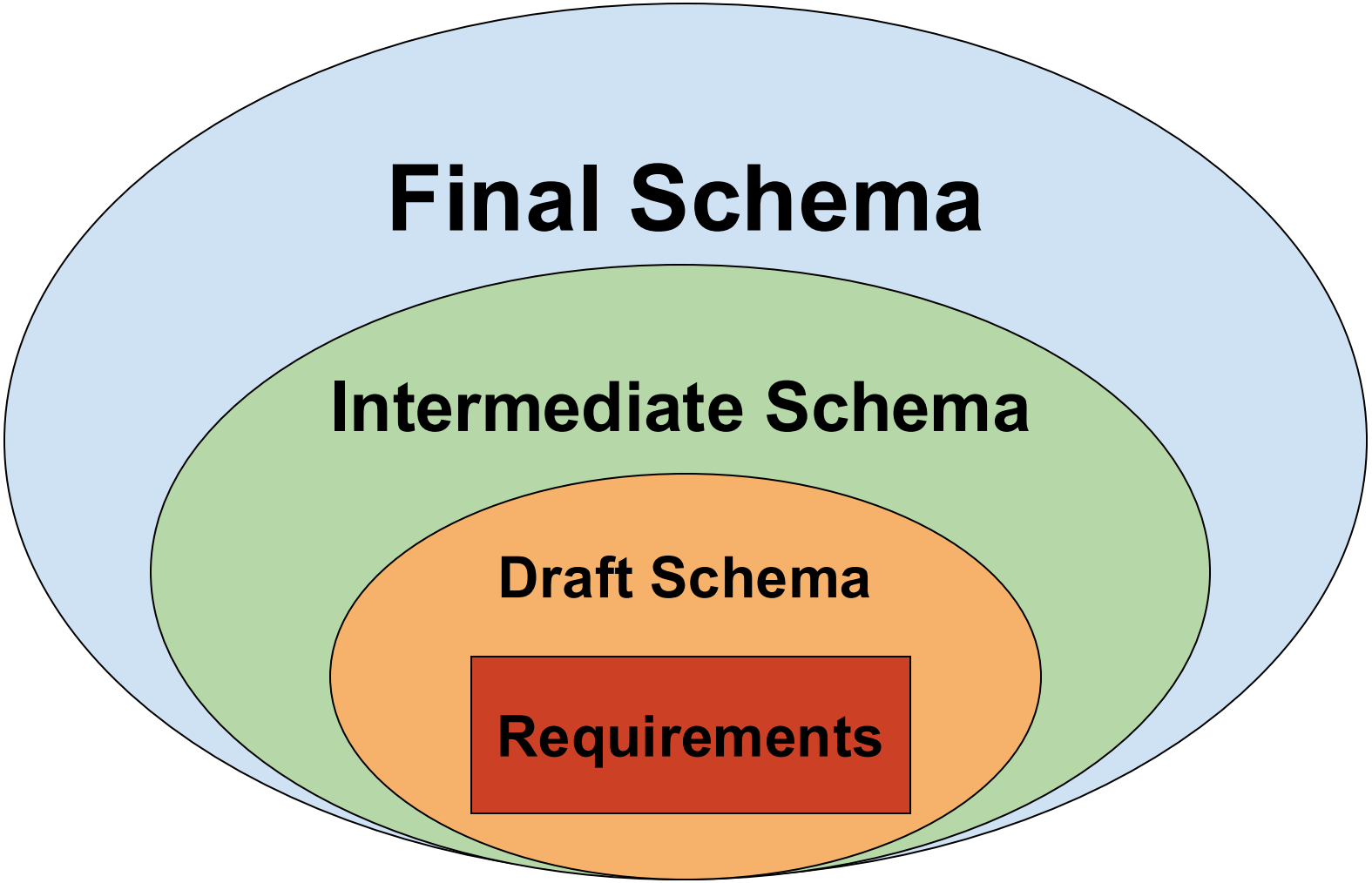


If a task can be performed by one operator and only in one office, then the reification could be simplified as in this ER schema

Strategies

- How could we map our requirements into instances of ER schemas?
- Strategies:
  - **Top-Down**
  - **Bottom-Up**
  - **Inside-Out**

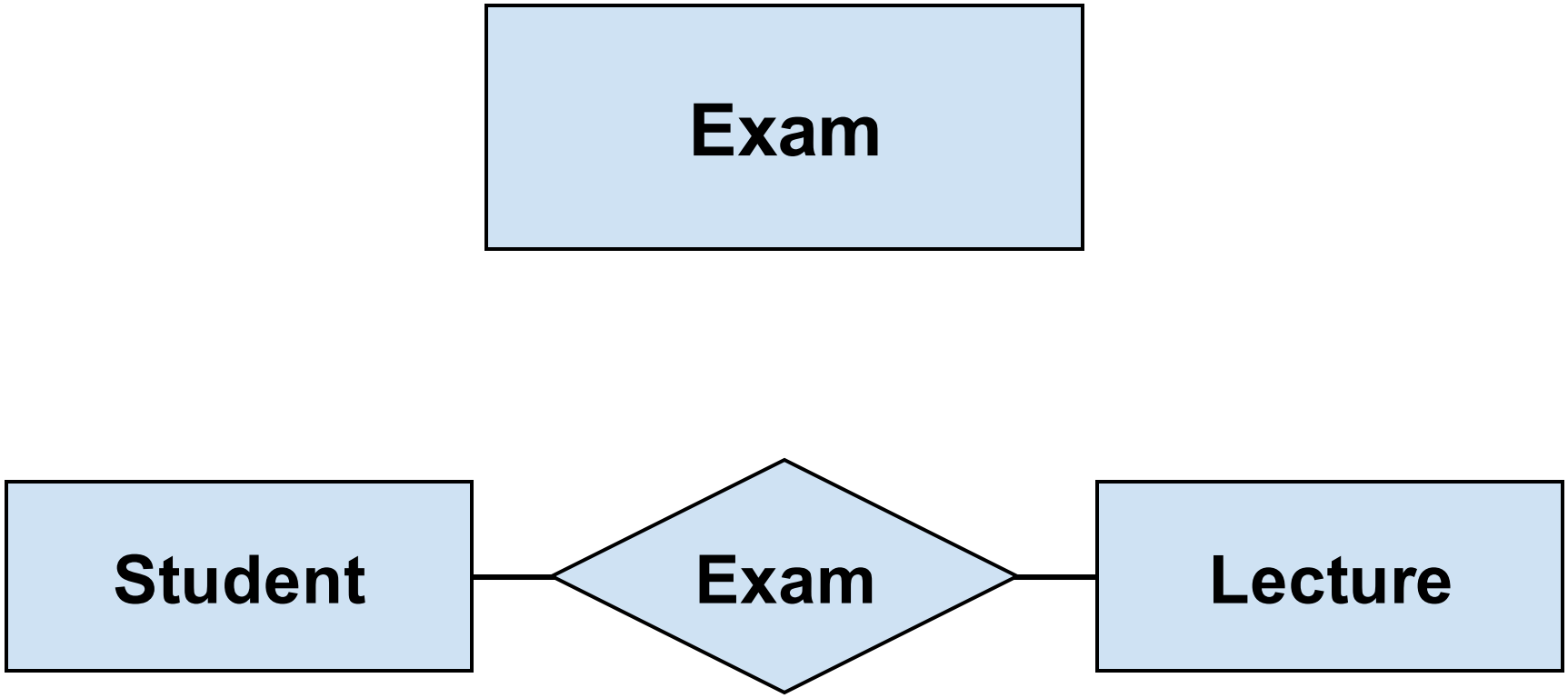
Strategy: Top-Down



Top-Down Refinements (1)

Exam

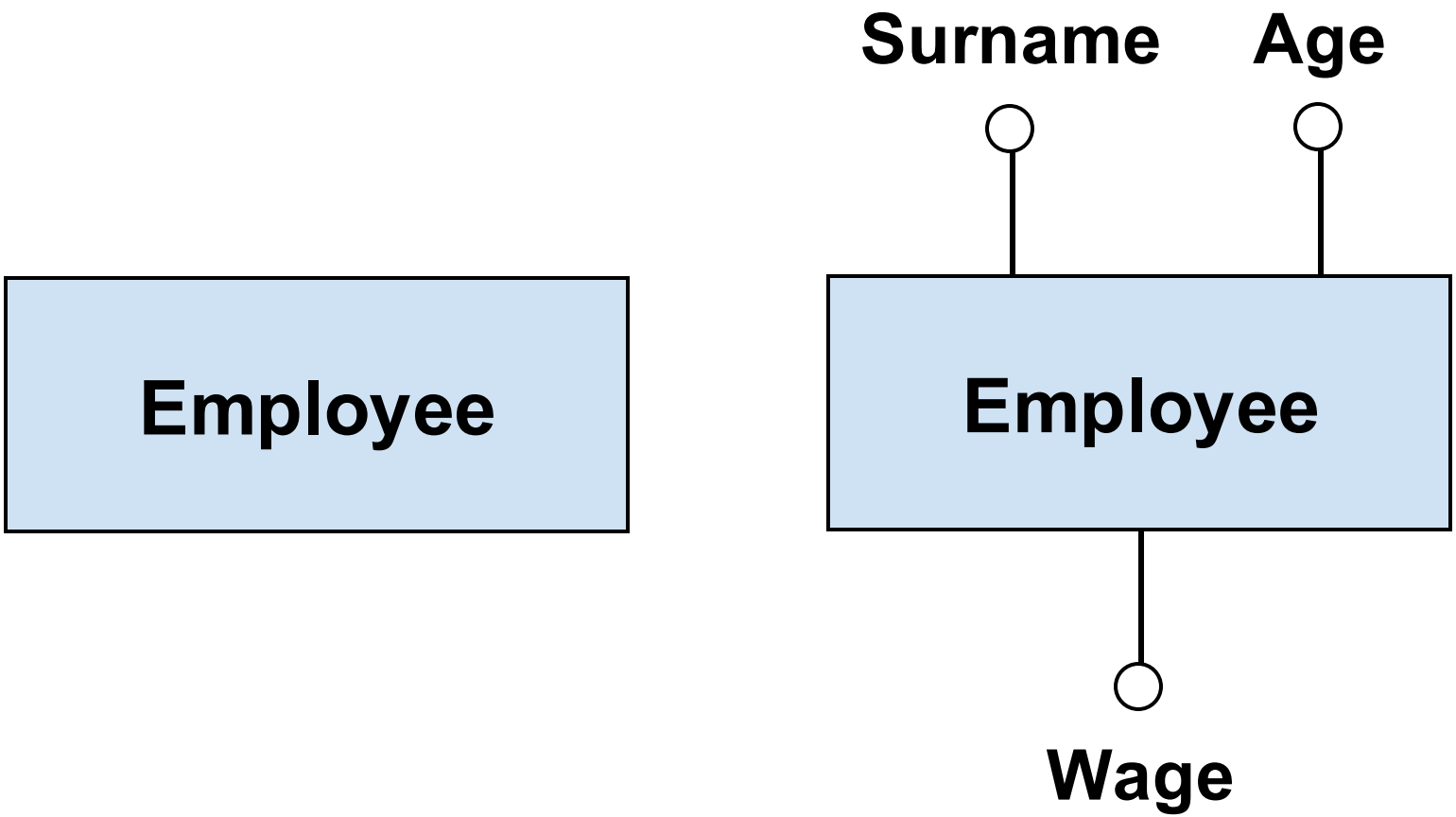
Top-Down Refinements (1)



Top-Down Refinements (2)

Employee

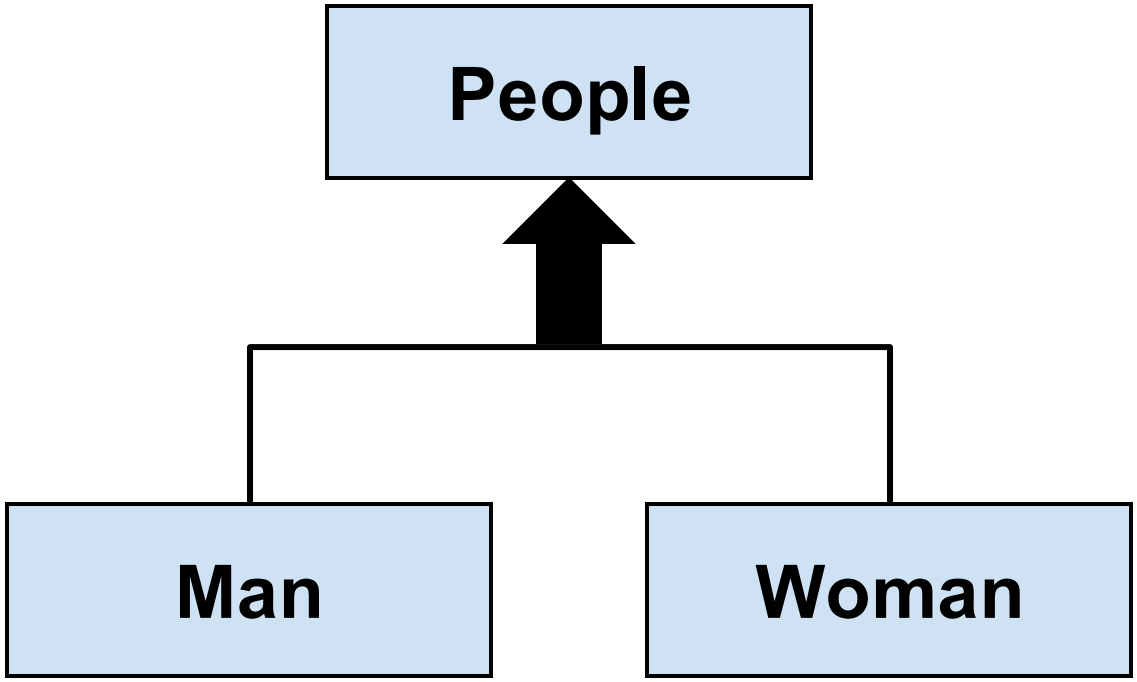
Top-Down Refinements (2)



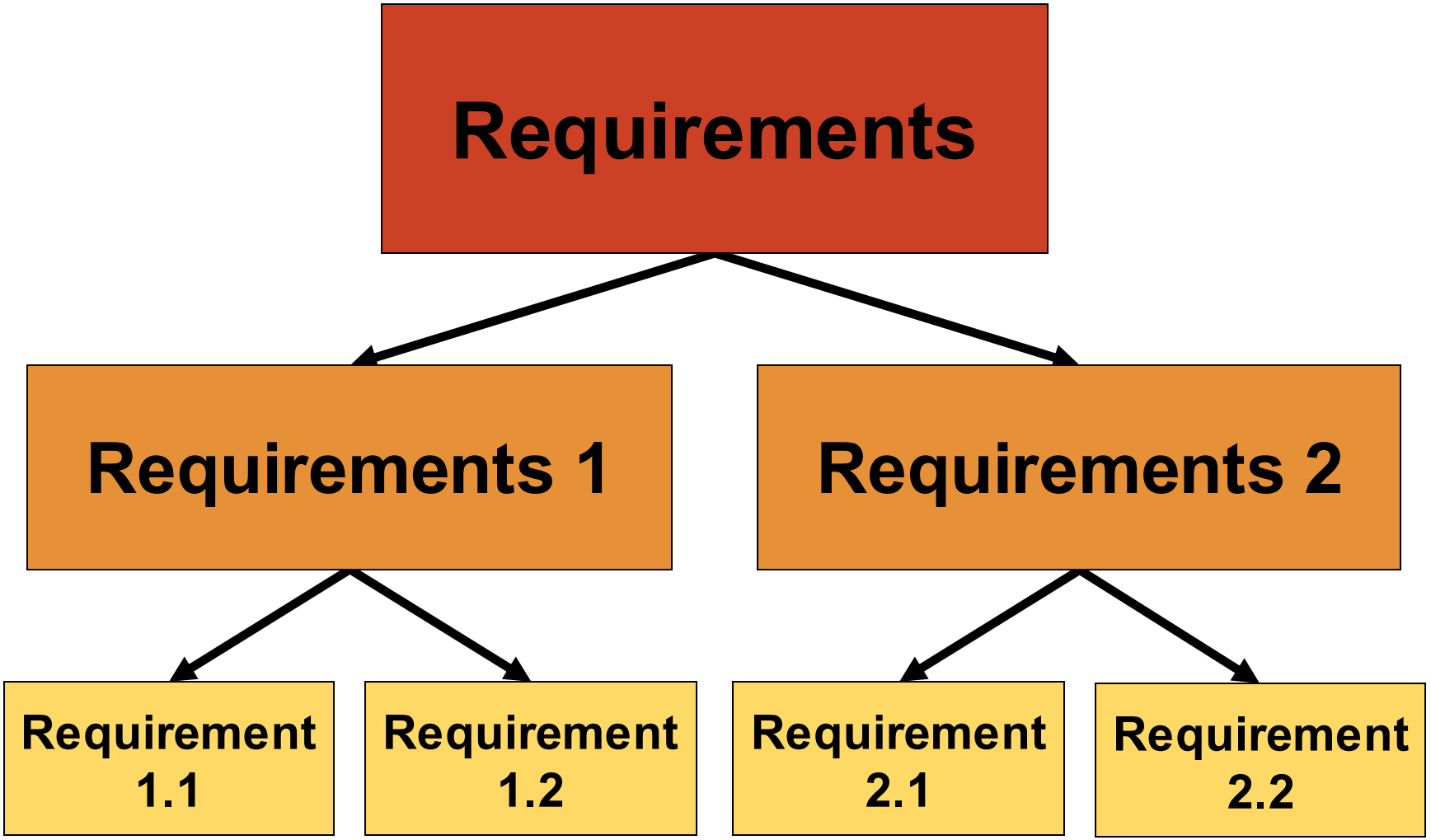
Top-Down Refinements (3)

People

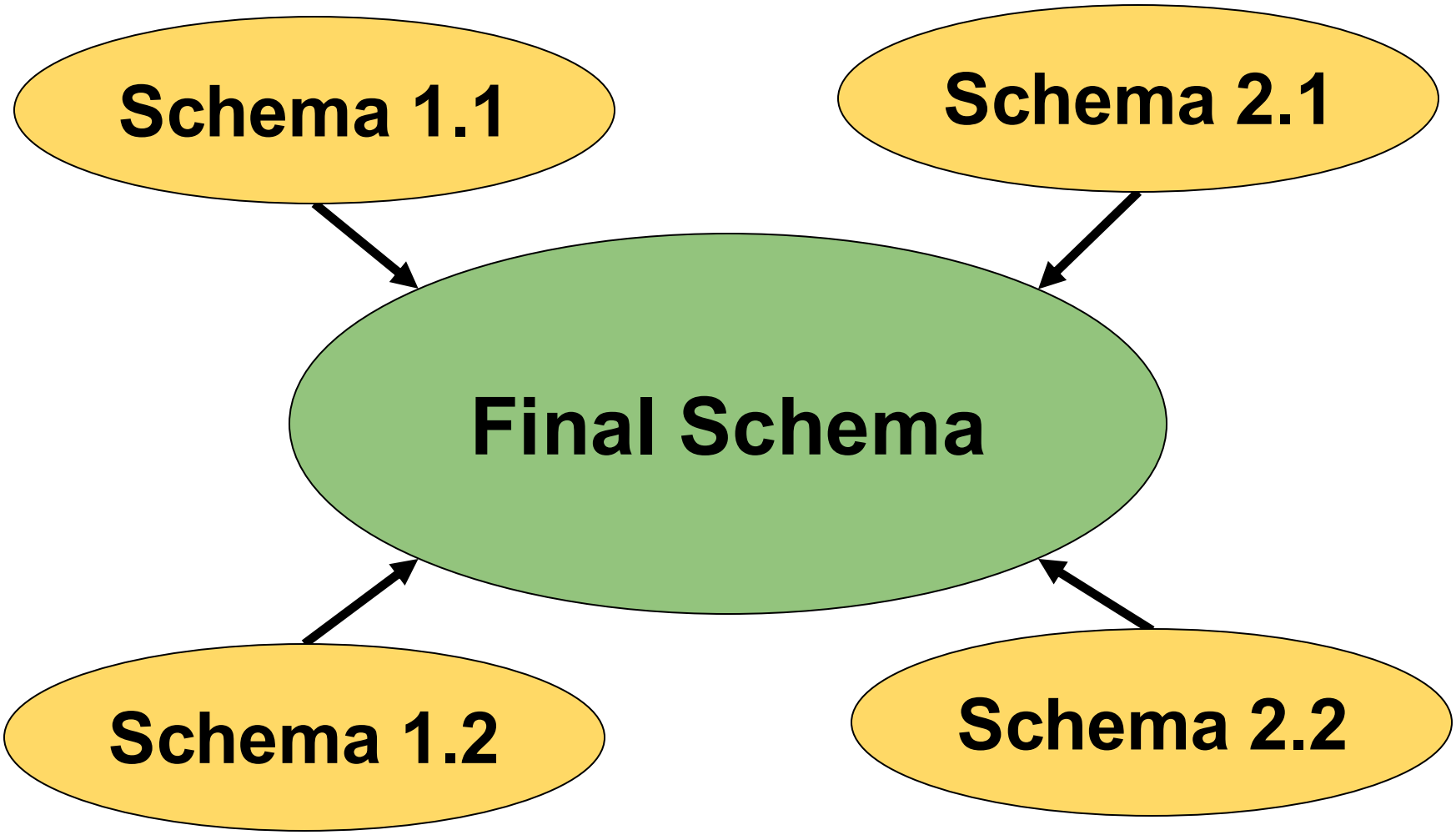
Top-Down Refinements (3)



Strategy: Bottom-Up (1)

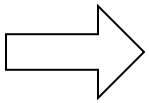


Strategy: Bottom-Up (2)



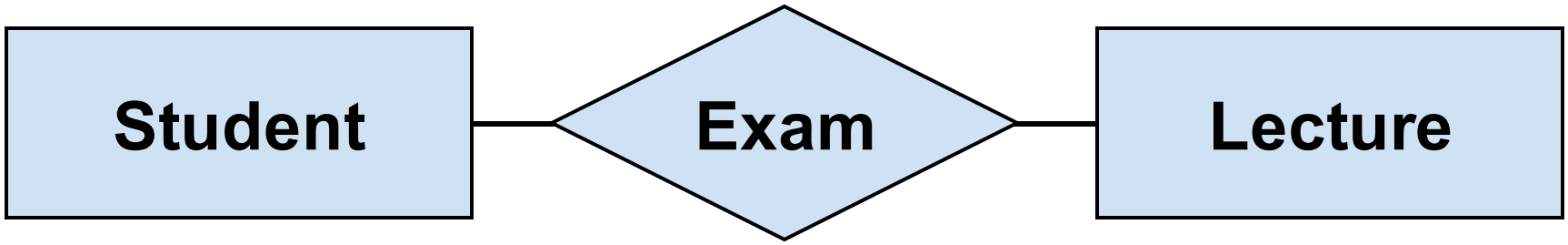
Bottom-Up Refinements (1)

Requirement on Employee

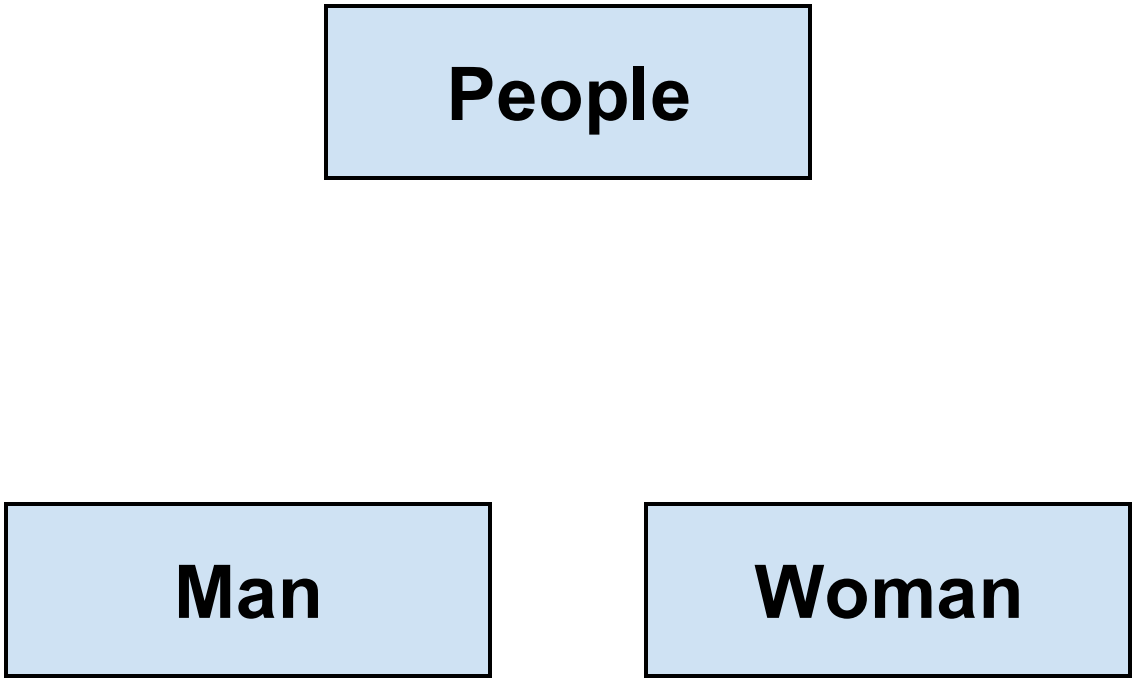


Employee

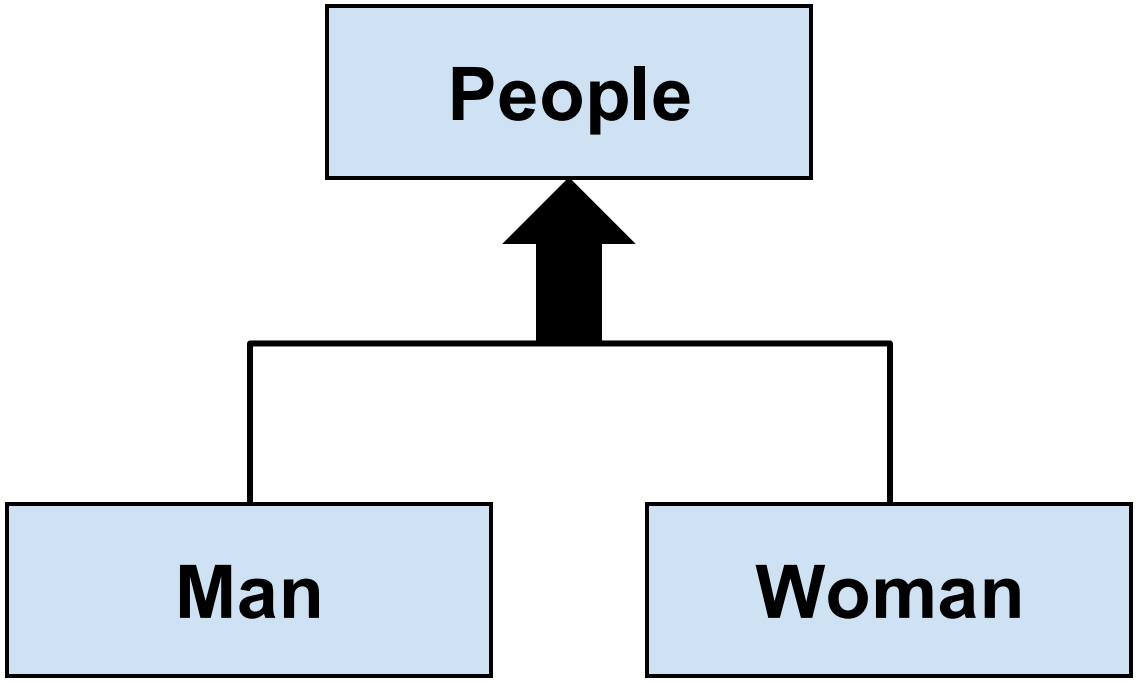
Bottom-Up Refinements (2)



Bottom-Up Refinements (3)



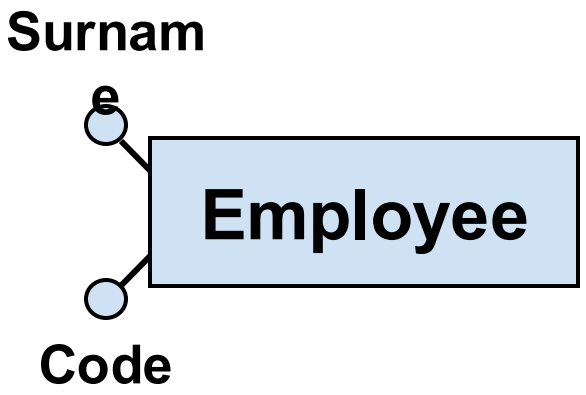
Bottom-Up Refinements (3)



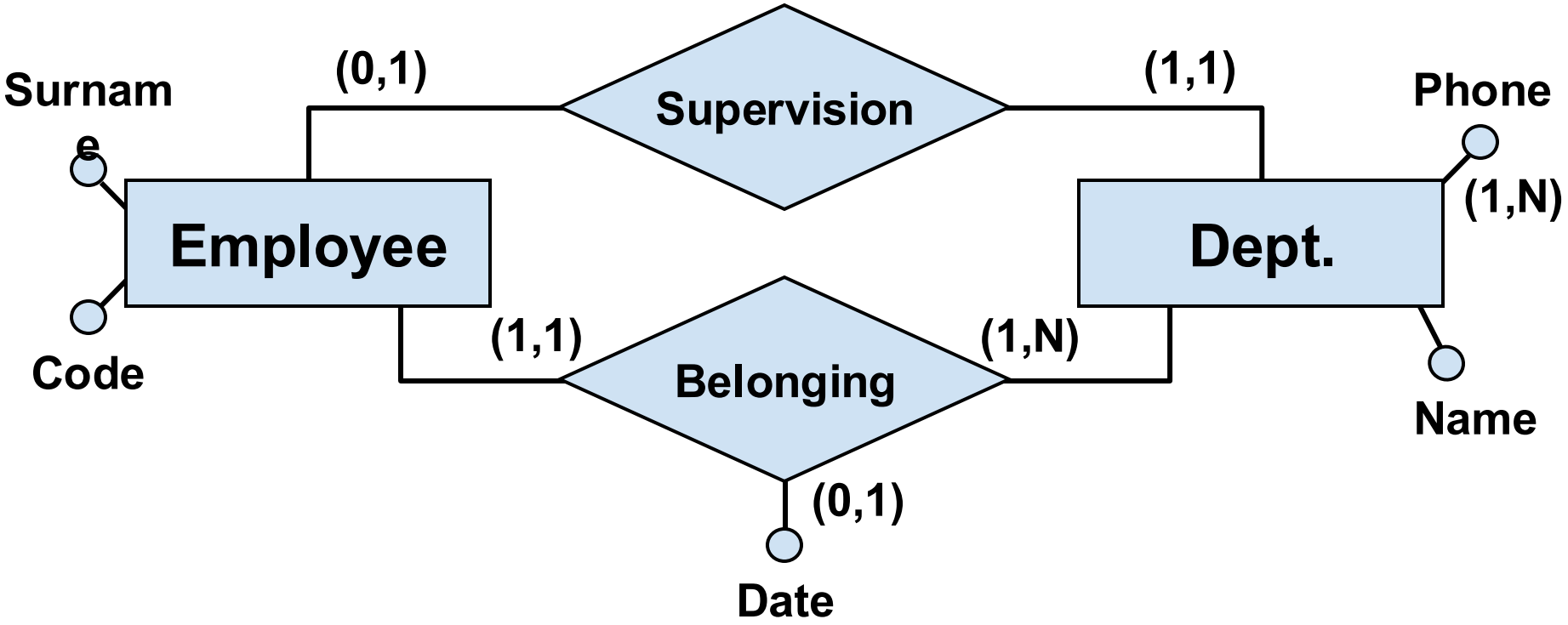
Strategy: Inside-Out (1)

Employee

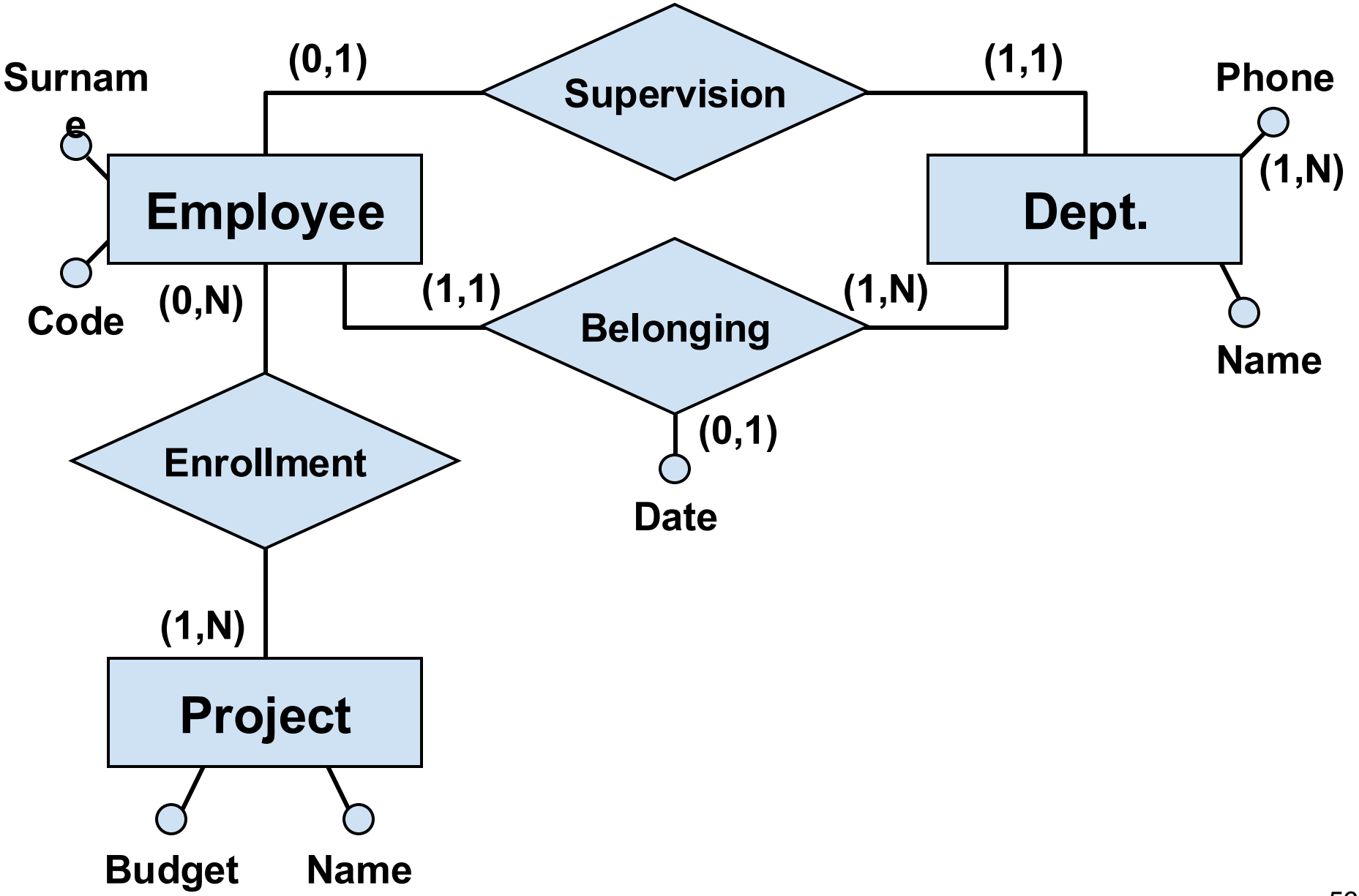
Strategy: Inside-Out (2)



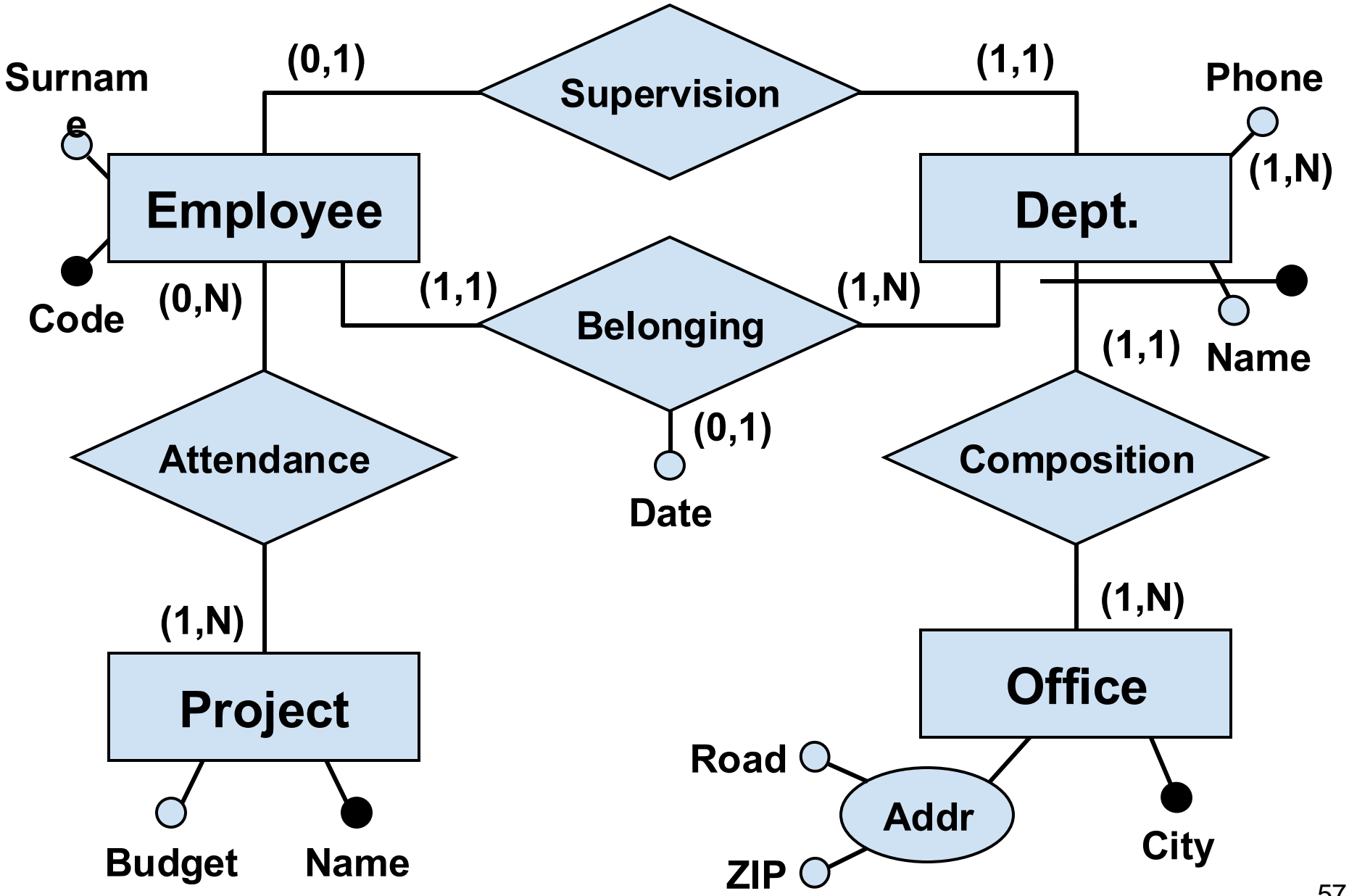
Strategy: Inside-Out (3)



Strategy: Inside-Out (4)



Strategy: Inside-Out (5)



Rule of Thumb

- Always use a mixed style:
  - First, you create a “sketch” by using the most relevant entities
  - Then, you decompose the schema
  - Then refine (top-down), integrate (bottom-up), expand (inside-out)

Sketching the ER Schema

- Start from the most relevant entities, either because are the most cited or because it is explicitly stated that they are relevant, and do a first basic ER schema

Best Practice Methodology

- **Requirements Analysis**
  - Analyse the requirements, resolve ambiguities
  - Create a glossary
  - Collect the requirements in similar sentences
- **Base case**
  - Define a first sketched schema with most relevant concepts
- **Iterative case** (repeat … until it's fine)
  - Refine the basic concepts using the requirements
  - Add concepts to describe requirements not described
- **Quality analysis** (repeat it throughout the schema)
  - Check the schema's quality and modify

ER Schema Quality: Some Measures

- Correctness
- Completeness
- Clarity
- Minimality

Best Practice & Schema Integration (1)

- **Requirements Analysis**
- **Base case**
- **Decomposition**
  - Decompose complex requirements according to the skeleton schema
- **Iterative case** for each sub-schema
- **Integrate**
  - Integrate the several sub-schemas in a total schema, using the skeleton schema as a reference
- **Quality analysis**

Best Practice & Schema Integration (2)

- **Requirements Analysis**
- **Decomposition**
  - Identification of the areas of interest and partition of requirements (or even separate acquisition)
- **For each area**
  - **Base case**
  - **Iterative case**
- **Integrate**
- **Quality analysis**

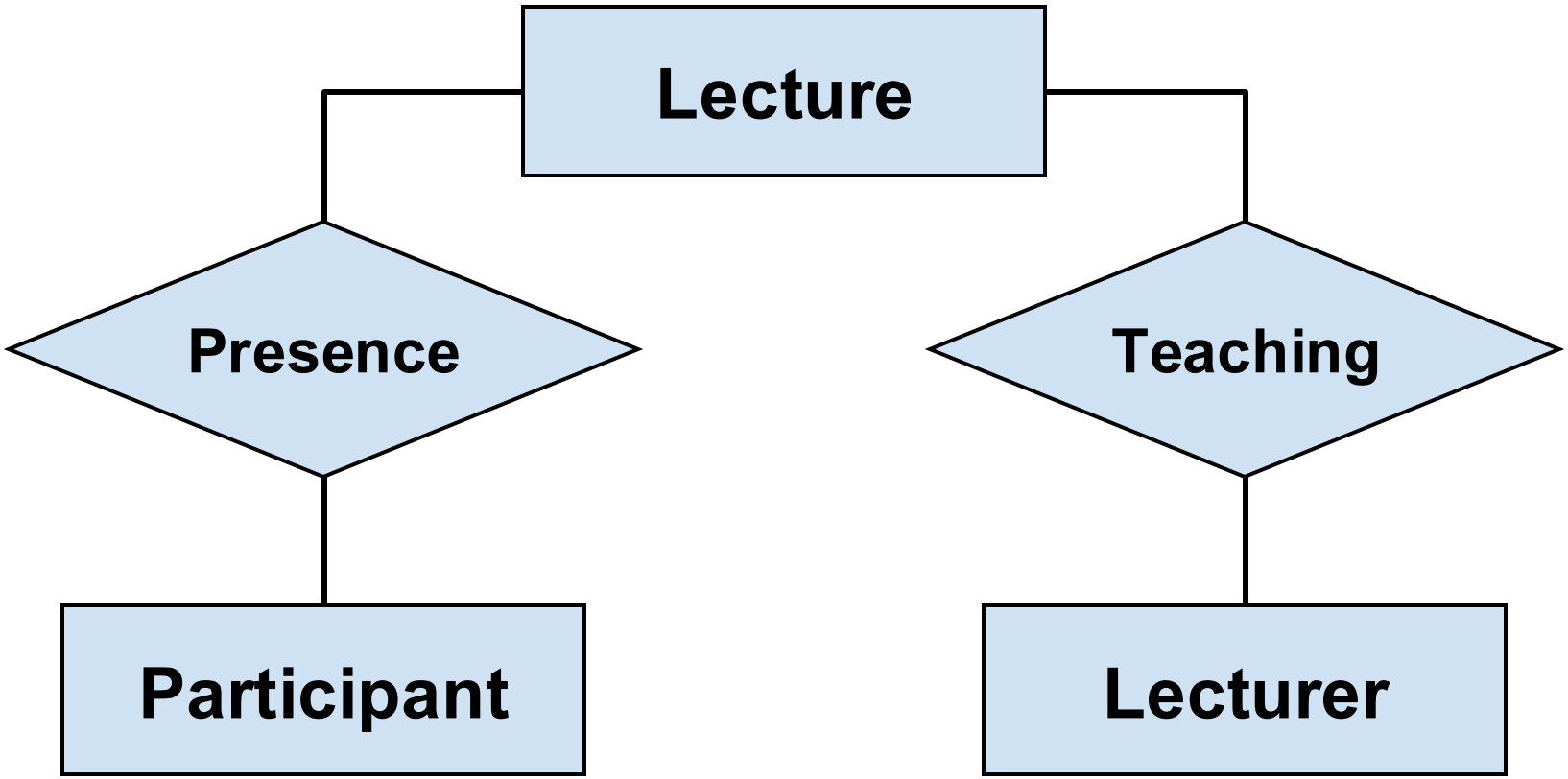
Final Example of Conceptual Design

The Training Company

Example: General Statement

- A “training company” requires a database providing some courses, for which we want to handle both lectures and teachers

Sketched Schema



Refining: Participants (1)

- There are ~5000 **participants**, which have an ID, their tax code (Tax), a surname, age, sex, place of birth, their employers' name, where they have worked previously and when (**starting and end date**), address and mobile phone, the attended courses (**current and past courses**) with their final grade

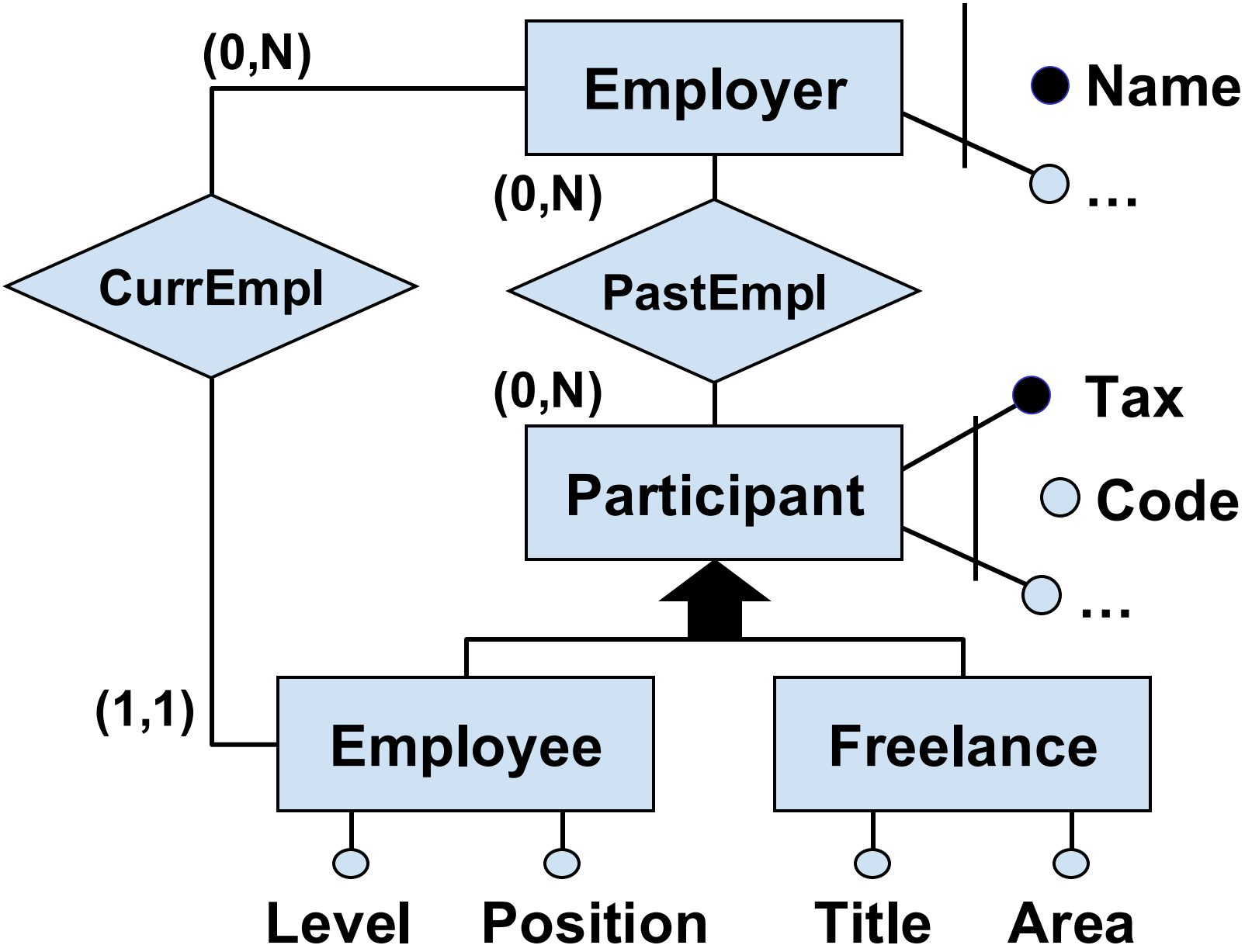
Refining: Participants (2)

- If a student is *freelancer*, we store the area of interest and, if they have it, an honorific
- **If a student belongs to the same organization, we want to know their level within the management structure and their working position**

Refining: Employer

- We store the participants' employers, both current and from the past. In particular we represent their name, address and phone number

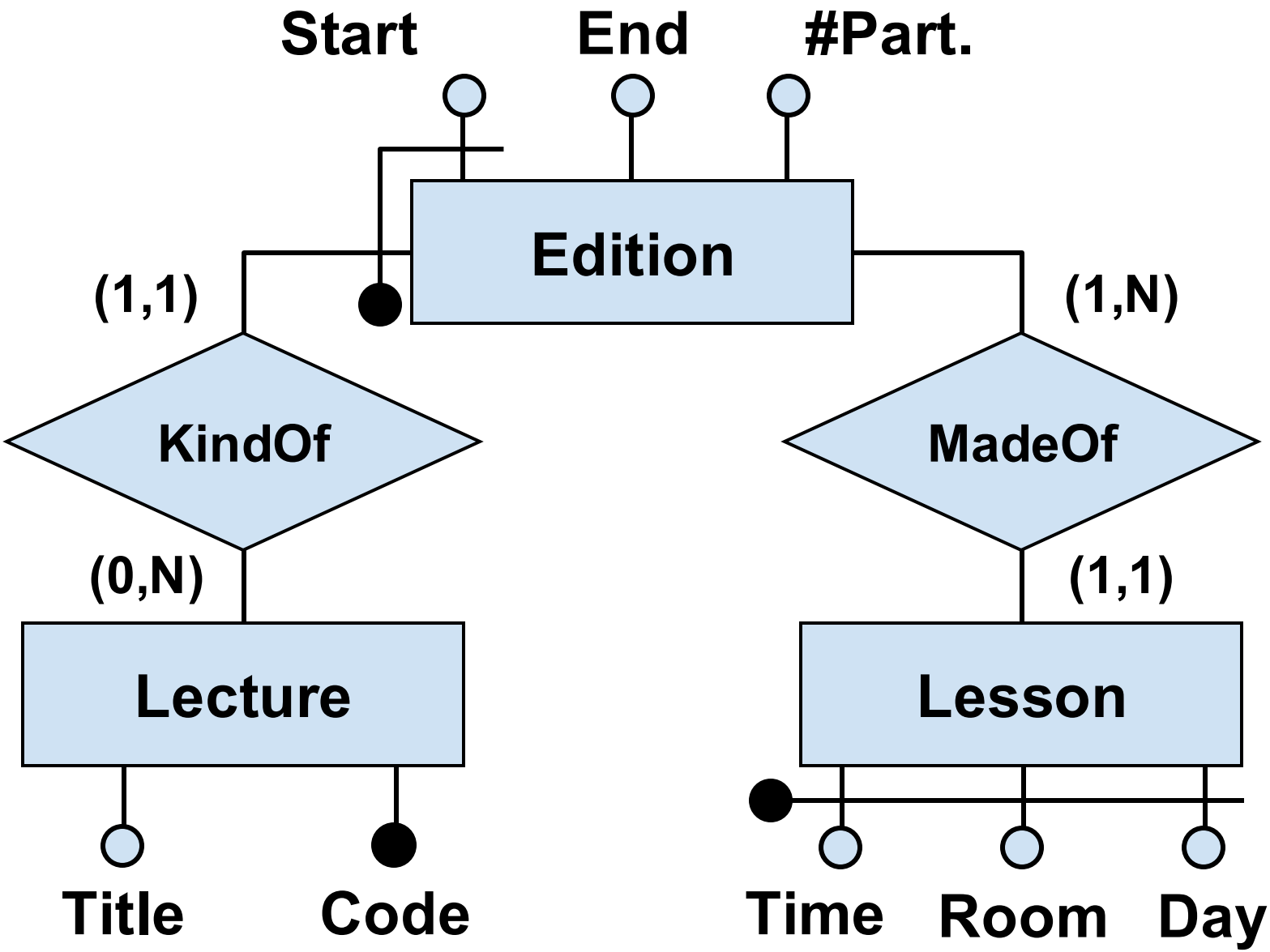
Partial ER Schema (1)



Refining: Course

- The courses (~200) have a code, a title and could have different editions, have a beginning and an end date. For each edition we keep the number of participants, the day of the week, the rooms and when the courses are held

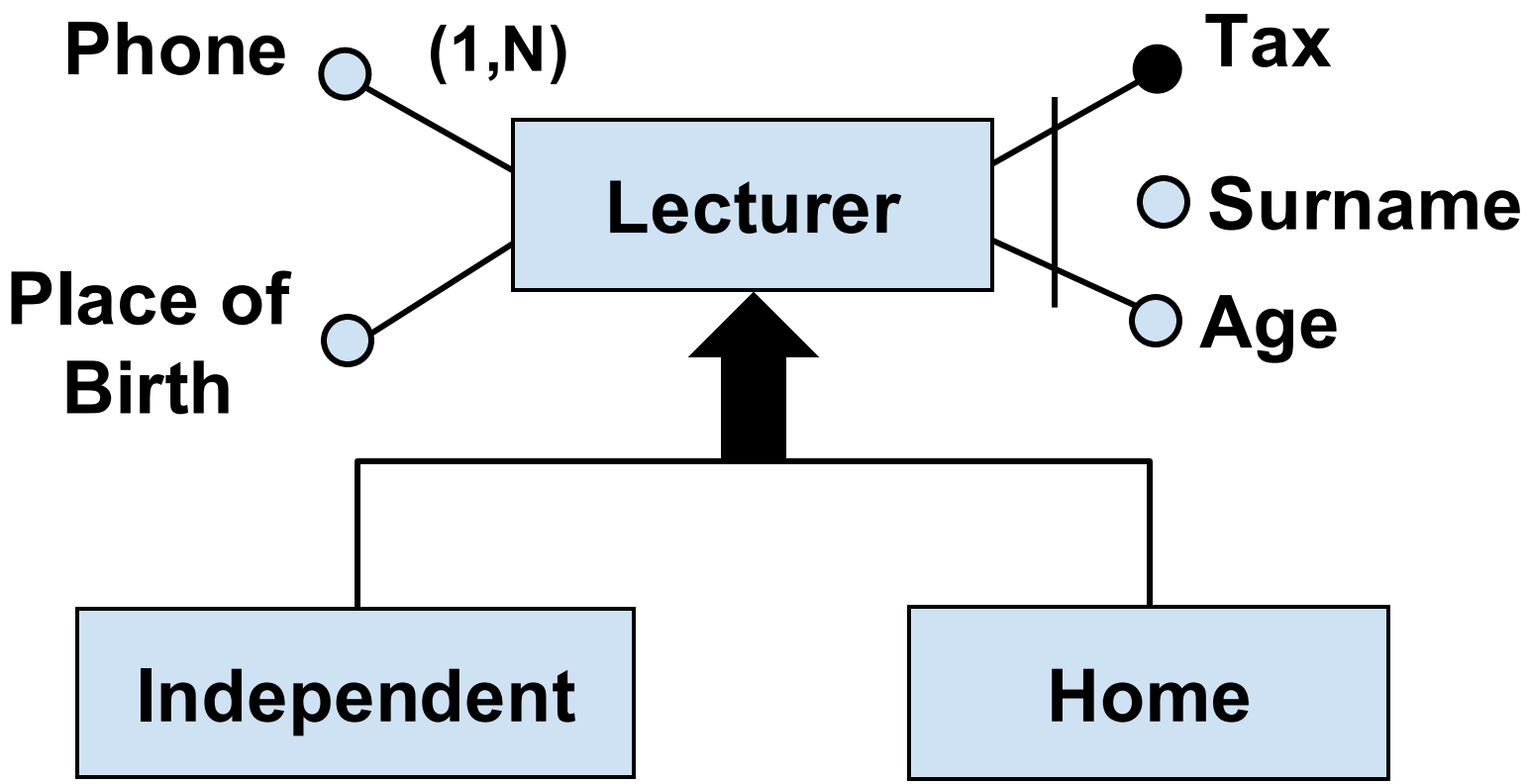
Partial ER Schema (2)



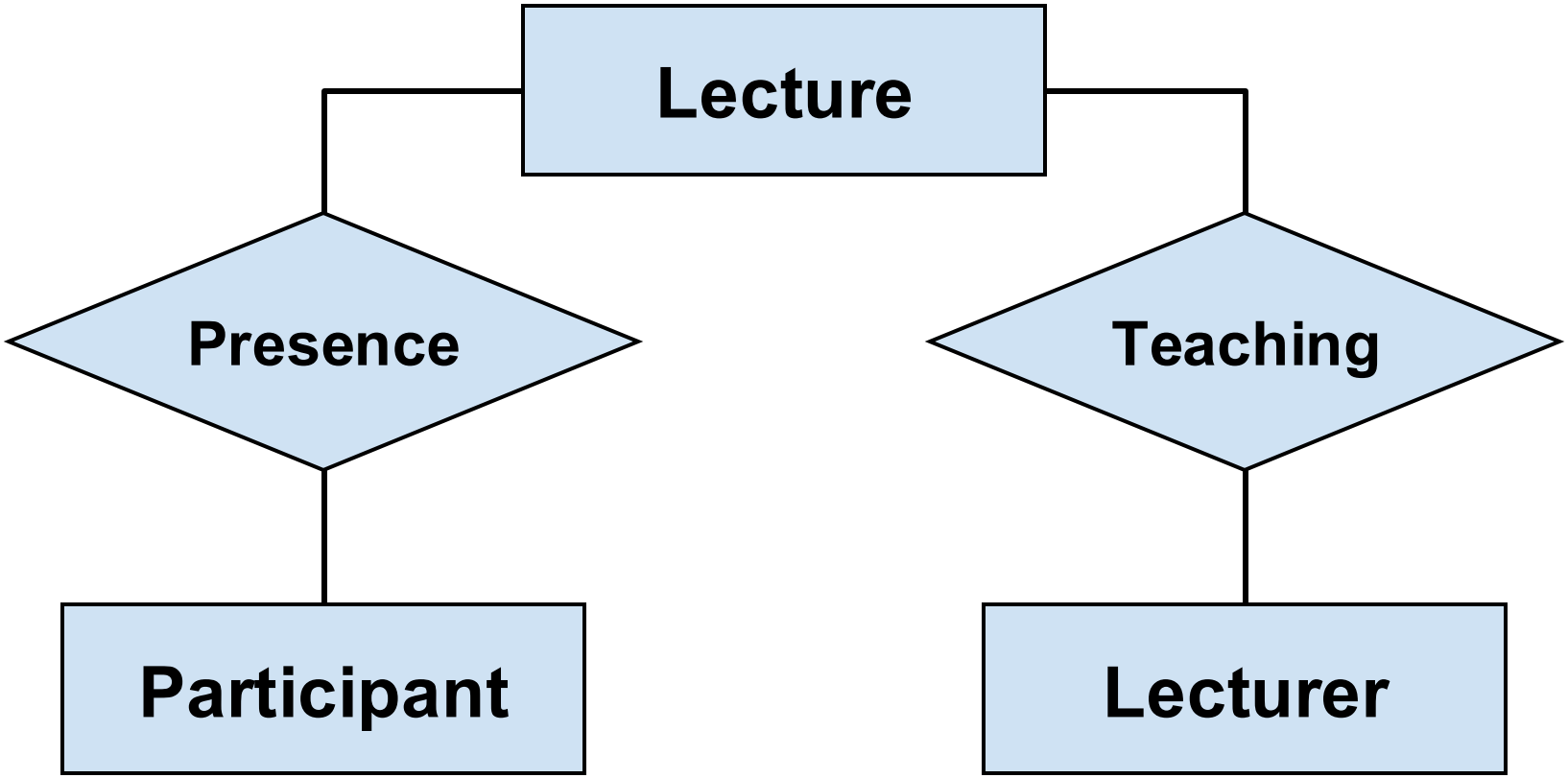
Refining: Lecturers

- Regarding the **lecturers** (~300), we store their surname, their age, the place of birth, the name of the taught course, the set of the courses that they taught in the past and the set of the courses that they could teach in the future. We want to keep all their phone records. **Lecturers** could be either house employees or independent contractors

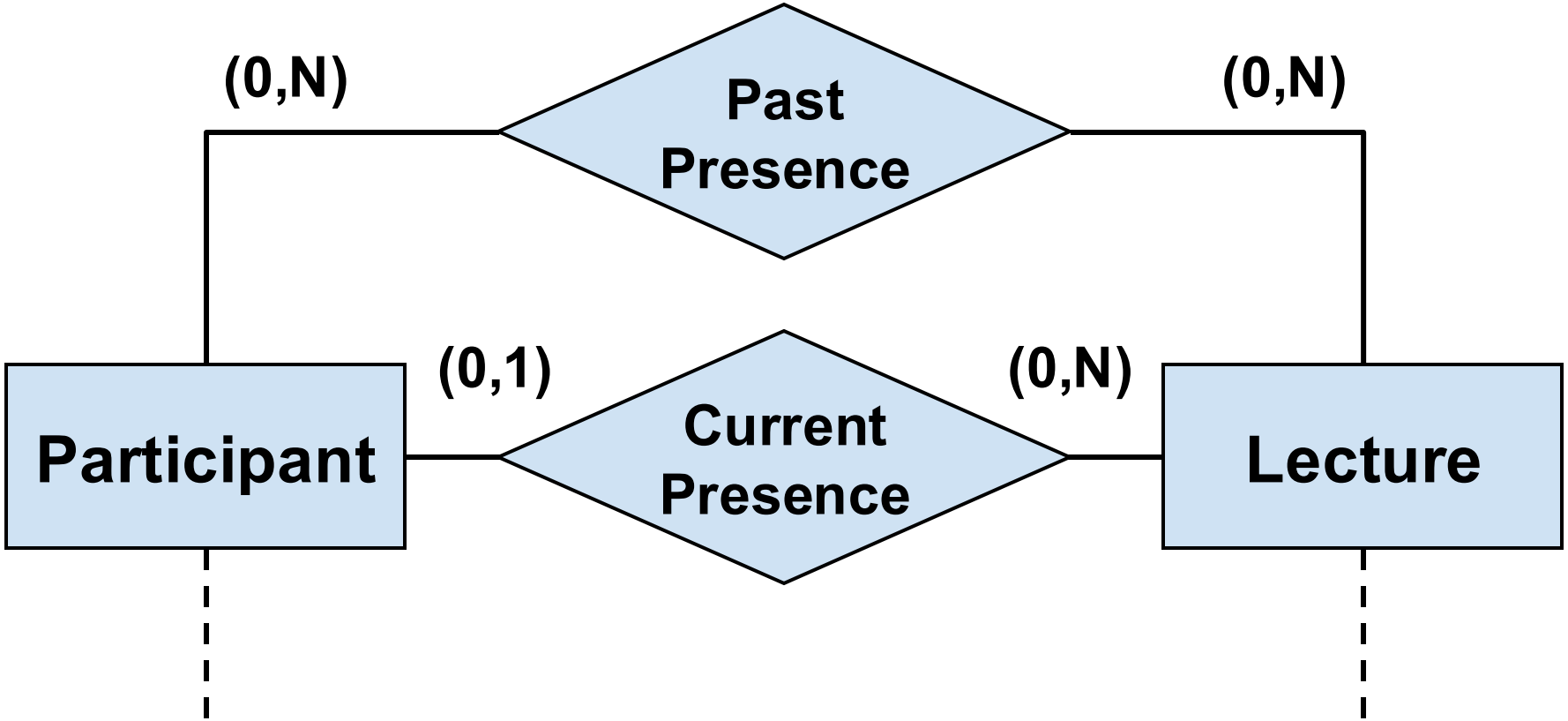
Partial ER Schema (3)



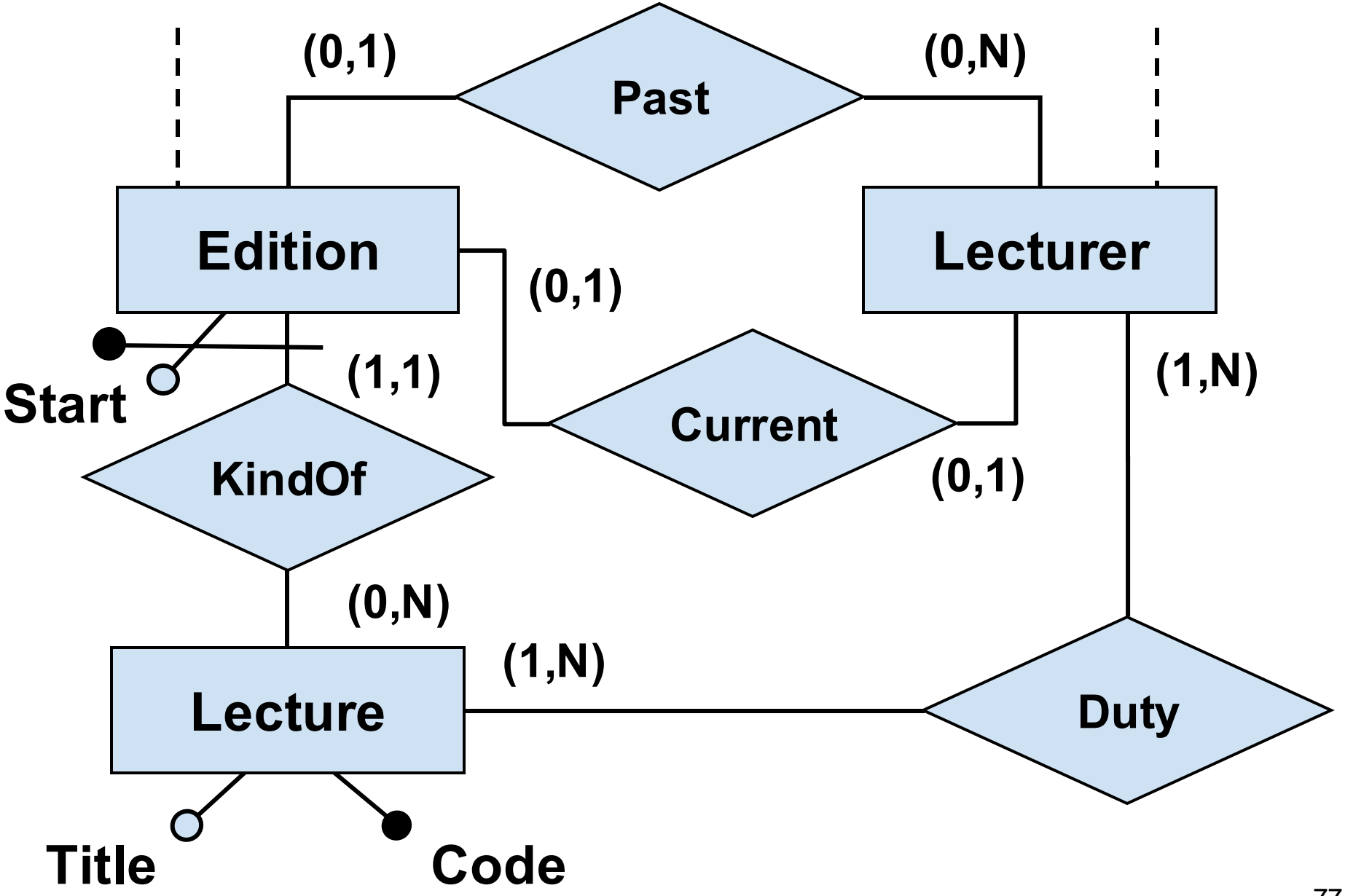
Sketched Schema: Integration



Intermediate Schema (1)



Intermediate Schema (2)



Final Schema: Just Entities & Relations