CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

Entity-Relationship Models

LECTURE 2

Dr. Philipp Leitner

✉ philipp.leitner@chalmers.se

@xLeitix

Some Admin Info …

Has everybody got access to GUL?

Please let me know if you still cannot access the online platform!

Please register for groups in GUL!

Registration is already possible.

Deadline:
2018-01-26 16:00

If you don’t have a partner, use the forum to find one.

We will randomly merge all groups with 1 student after the deadline.

Grading Scheme for Chalmers Students

… turns out there actually *is* a different grading scheme for Chalmers students …

<u>If you are a Chalmers student:</u>

Five:	**>= 90% of exam points**
Four:	**>= 70% of exam points**
Three:	**>= 50% of exam points**
Fail:	**< 50% of exam points**

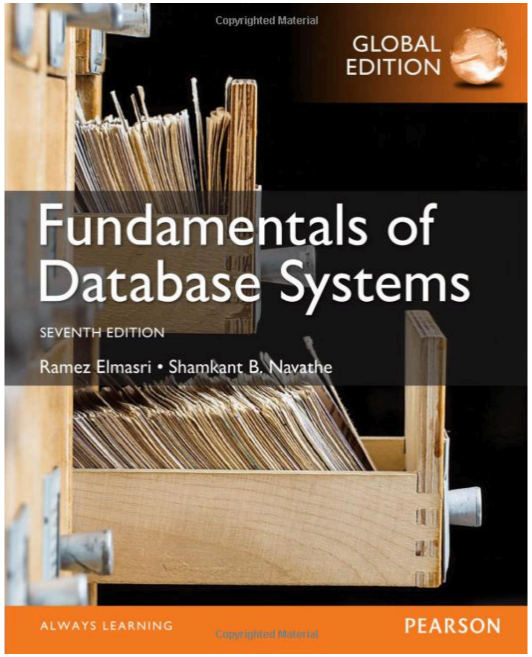
(I have also updated the slides from TUE to be consistent)

LECTURE 2

Covers …

Chapter 3

Please read this up until next lecture!



What we will be covering

Basics of ER modelling

Lots and lots of notation

Overview of Database Design Process

Two main activities:

Database design

Applications design

Today: focus on **conceptual database design**

To design the conceptual schema for a database application

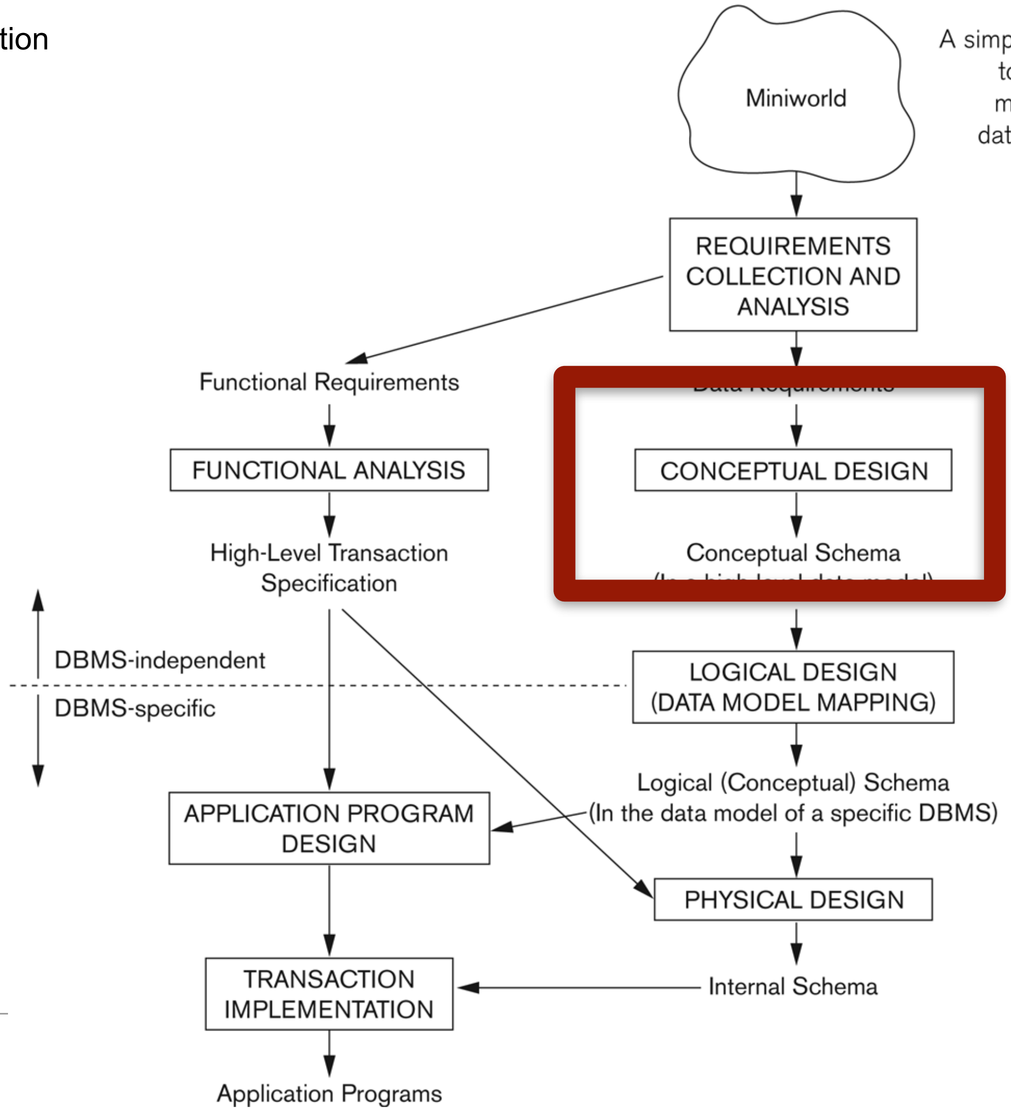

Miniworld
REQUIREMENTS COLLECTION AND ANALYSIS
Functional Requirements
Data Requirements
FUNCTIONAL ANALYSIS
CONCEPTUAL DESIGN
High-Level Transaction Specification
Conceptual Schema
(In a high-level data model)
DBMS-independent
DBMS-specific
LOGICAL DESIGN (DATA MODEL MAPPING)
Logical (Conceptual) Schema
(In the data model of a specific DBMS)
APPLICATION PROGRAM DESIGN
PHYSICAL DESIGN
TRANSACTION IMPLEMENTATION
Internal Schema
Application Programs

Figure 3.1
A simplified diagram to illustrate the main phases of database design.

Why ER Diagrams?

Simple

Or at least much simpler than UML

Maps **very closely** to relational algebra and the logical database design

(Somewhat) widely spread in **industry**

Notations for Conceptual Database Design

Entity Relationship (ER) Diagrams

(today)

Enhanced Entity Relationship (EER) Diagrams

(next week)

UML

(not part of this course)

Industrial tools (e.g., CASE tools)

Important to not get too hung up on notation (but important to know it!)

The very basics of ER Diagrams

table rows

Entities

Things that exist in the real world that you want to manage

Attributes

The data that makes up those things

table column values

Relationships

How those things relate to each other

foreign keys

Let’s start with an example

Consider the following functional requirements

The company is organized into departments. Each department has a name, number and an employee who manages the department. We keep track of the start date of the department manager. A department may have several locations.

Each department controls a number of projects. Each project has a unique name, unique number and is located at a single location.

Let’s start with an example

Consider the following functional requirements

The company is organized into **departments**. Each department has a name, number and an **employee** who manages the department. We keep track of the start date of the department manager. A department may have several **locations**.

Each department controls a number of **projects**. Each project has a unique name, unique number and is located at a single location.

More requirements

The database will store each **employee's** social security number, address, salary, sex, and birthdate. We will also need to know how many employees we have.

- Each employee works for one **department** but may work on several **projects**.
- The DB will keep track of the number of hours per week that an employee currently works on each **project**.
- It is required to keep track of the direct supervisor of each **employee**.

Each **employee** may have a number of **dependents**.

- For each **dependent**, the DB keeps a record of name, sex, birthdate, and relationship to the **employee**.

ER Model Concepts

Entity is a basic concept for the ER model. Entities are specific things or objects in the mini-world that are represented in the database.

Examples: EMPLOYEE John Smith, the Research DEPARTMENT, the ProductX PROJECT

Attributes are properties used to describe an entity.

Examples: an EMPLOYEE entity may have the attributes Name, SSN, Address, Sex, BirthDate

Entity Notation

Entity types are demarked as squares (boxes)

Attributes are ovals attached to the boxes with straight lines

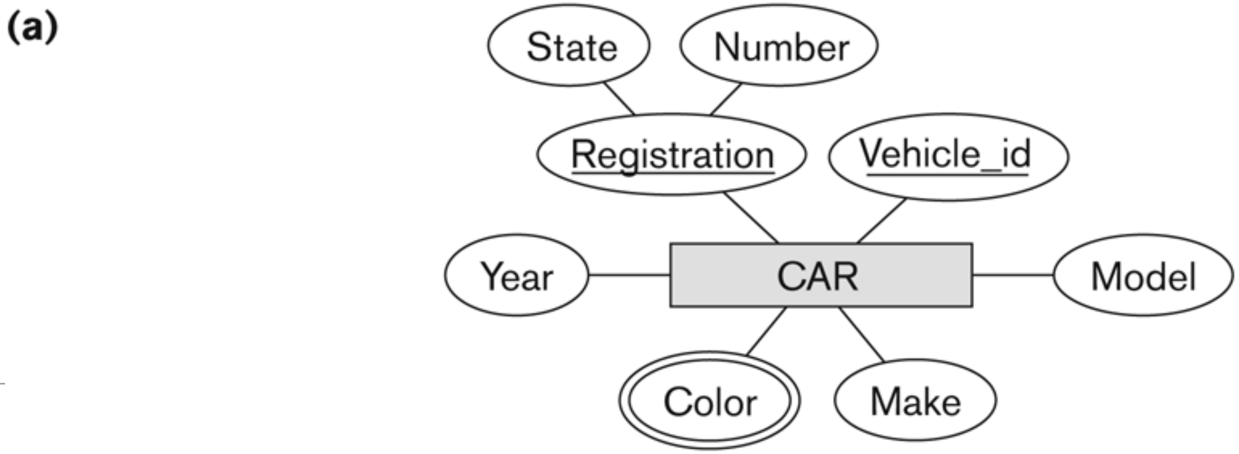


Figure 3.7
The CAR entity type with two key attributes, Registration and Vehicle_id. (a) ER diagram notation. (b) Entity set with three entities.

ER Model Concepts

A specific entity will have **a value for each of its attributes**

Example:

a specific employee entity may have Name='John Smith', SSN='123456789', Address ='731, Fondren, Houston, TX', Sex='M', BirthDate='09-JAN-55'

'NULL' may or may not be allowed

Each attribute has a value set (or data type)

e.g., integer, string, date, enumerated type, …

Types of Attributes

Simple

Every entity has **exactly one** atomic value

Example: `project_name`

Composite

Attribute consists of several components

Example: `Address(Street, City, State, ZipCode, Country)`

Multi-valued

Entity may have a list of values for that attribute.

Example: `{PreviousDegrees}`

Attribute Composition

Composites and multi-values may be hierarchically composed

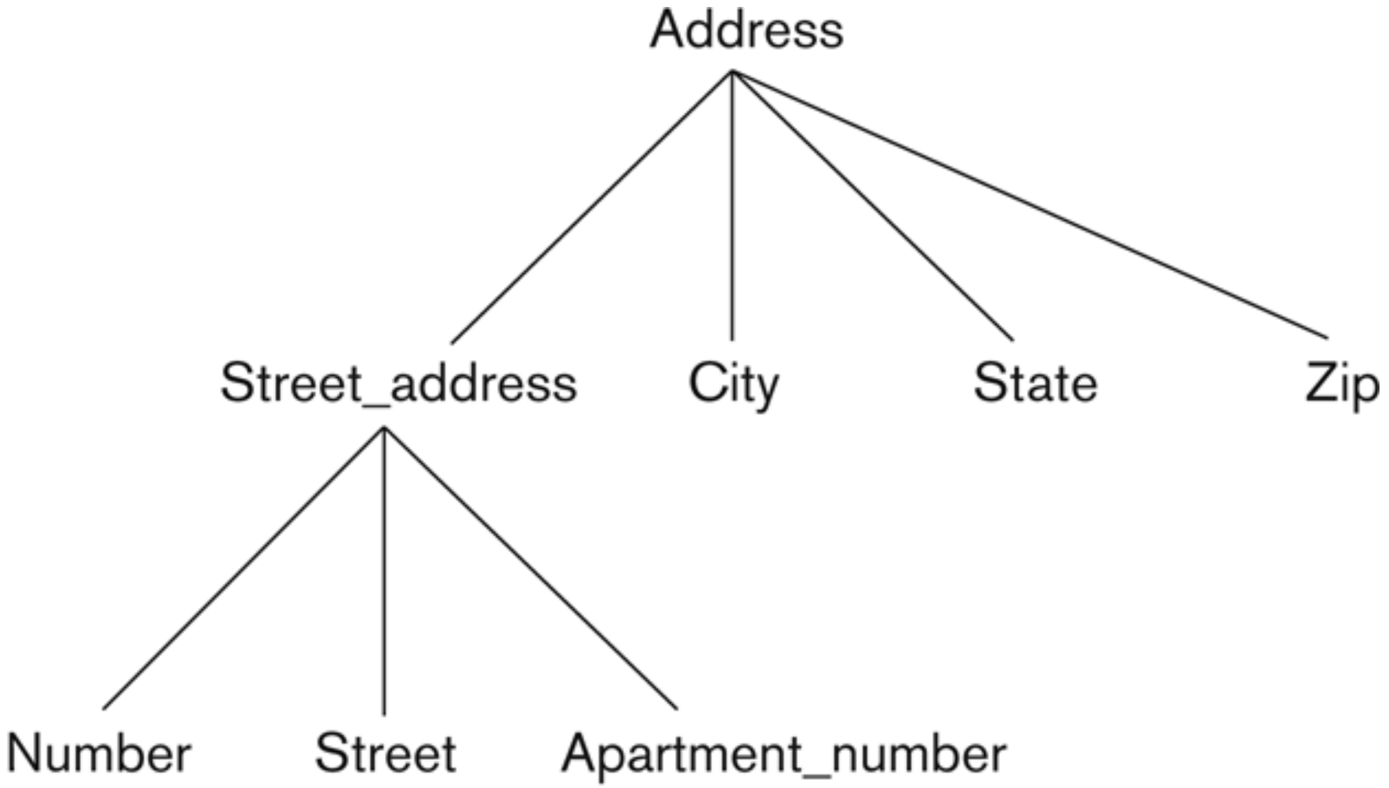


Figure 3.4
A hierarchy of composite attributes.

Entity Types and Keys

Entities of the same type (e.g., all employees) are grouped into an **entity type**

Example: the entity type EMPLOYEE and PROJECT

An attribute of an entity type for which each entity must have a **unique value** is called a **key attribute** of the entity type.

Example: Ssn of EMPLOYEE.

By convention we use ALL CAPS for entity type names (but not for attributes)

Entity Types and Keys

Keys may be **composite**

Multiple attributes together are unique

Entity types may have **more than one key**

Key notation:

Each key is underlined

Entity Sets

Each entity type will have a **collection of entities** stored in the database

- Called the **entity set** or sometimes entity collection

- At every time all entity sets together form the current database snapshot

Example in ER Notation

(a)

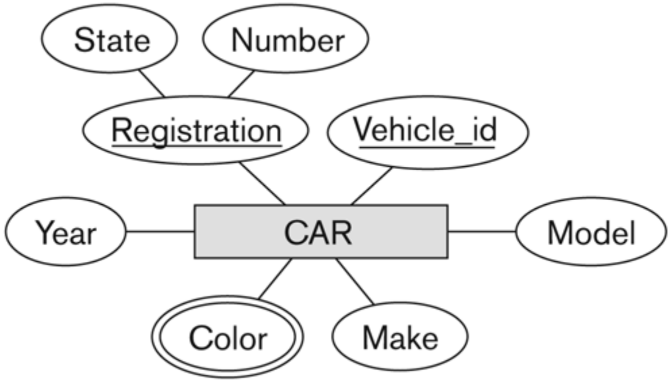


Figure 3.7
The CAR entity type with two key attributes, Registration and Vehicle_id. (a) ER diagram notation. (b) Entity set with three entities.

(b)

CAR
Registration (Number, State), Vehicle_id, Make, Model, Year, {Color}

CAR$_1$
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 2004 {red, black})

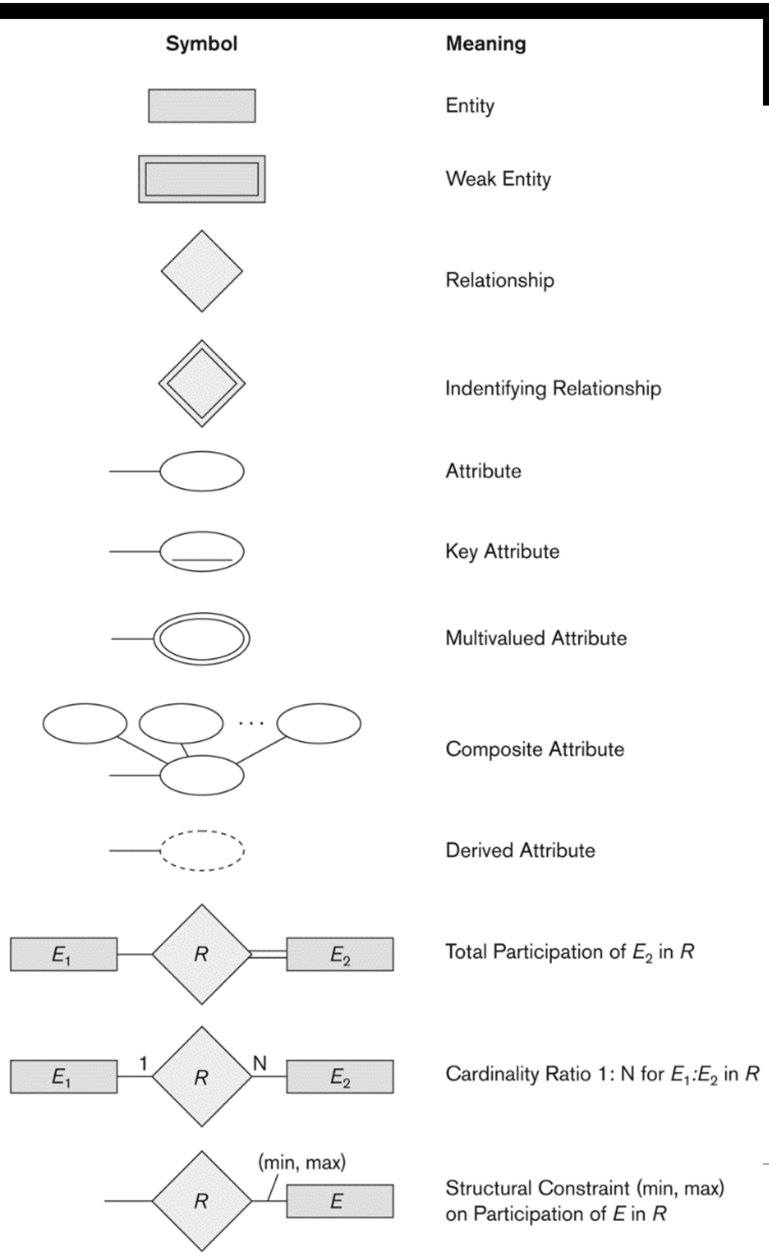
CAR$_2$
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})

CAR$_3$
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

⋮

ER notation cheat sheet

Figure 3.14
Summary of the notation for ER diagrams.



Entity

Weak Entity

Relationship

Indentifying Relationship

Attribute

Key Attribute

Multivalued Attribute

Composite Attribute

Derived Attribute

Total Participation of E_2 in R

Cardinality Ratio 1: N for E_1:E_2 in R

Structural Constraint (min, max) on Participation of E in R

Let's revisit our example

- The company is organized into **departments**.
  - Each department has a name, number and an **employee** who manages the department.
  - We keep track of the start date of the department manager. A department may have several **locations**.
  - Department name and number are unique.
- Each department controls a number of **projects**.
  - Each project has a unique name, unique number and is located at a single location.

Let’s revisit our example

The database will store each **employee’s** social security number, address, salary, sex, name, and birthdate.

- Names consist of first names, last names, and middle names.
- We will also need to know how many employees we have.
- Each employee works for one **department** but may work on several **projects**.
- It is required to keep track of the direct supervisor of each **employee**
- The DB will keep track of the number of hours per week that an employee currently works on each **project**.

Let’s revisit our example

- Each **employee** may have a number of **dependents**.
    - For each **dependent**, the DB keeps a record of name, sex, birthdate, and relationship to the **employee**.

Entity Types

Based on the requirements, we can identify **four initial entity types** in the COMPANY database:

DEPARTMENT
PROJECT
EMPLOYEE
DEPENDENT

Let’s model them!

Entity types and attributes

A lot is still missing here - namely how those entities relate

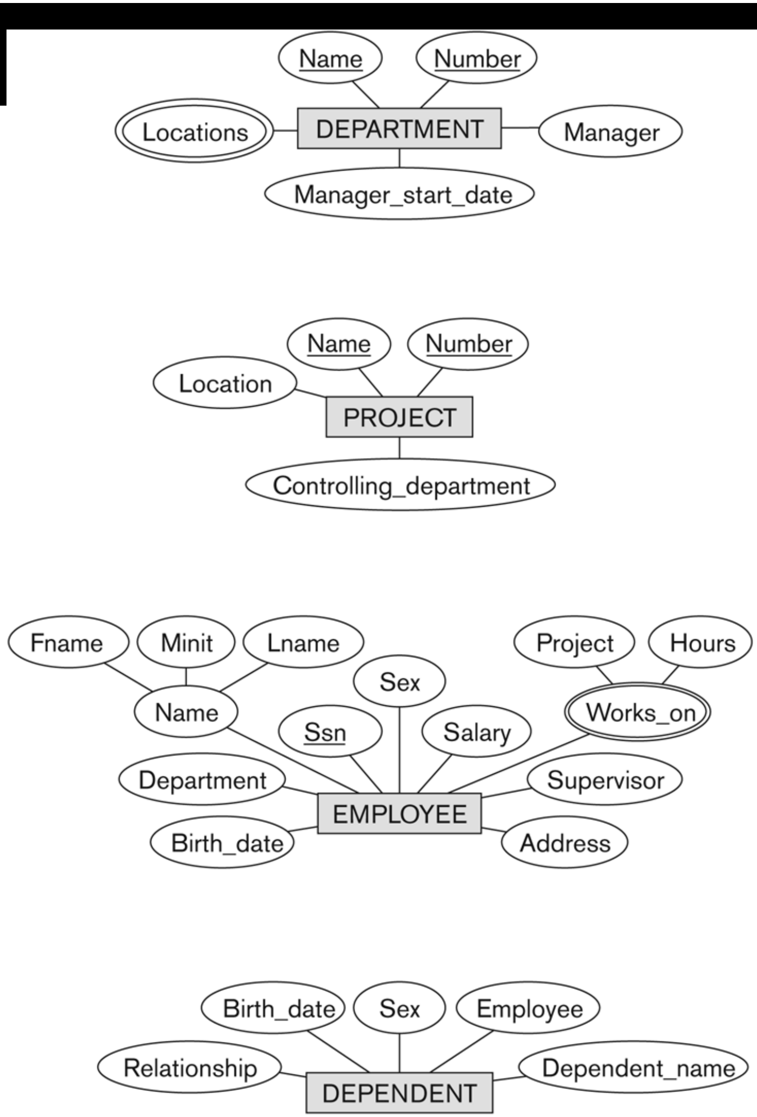


Figure 3.8
Preliminary design of entity types for the COMPANY database. Some of the shown attributes will be refined into relationships.

Relationships and Relationship Types

A relationship relates two or more distinct entities with a specific meaning.

Examples:

EMPLOYEE John Smith works on the ProductX PROJECT

EMPLOYEE Franklin Wong manages the Research DEPARTMENT.

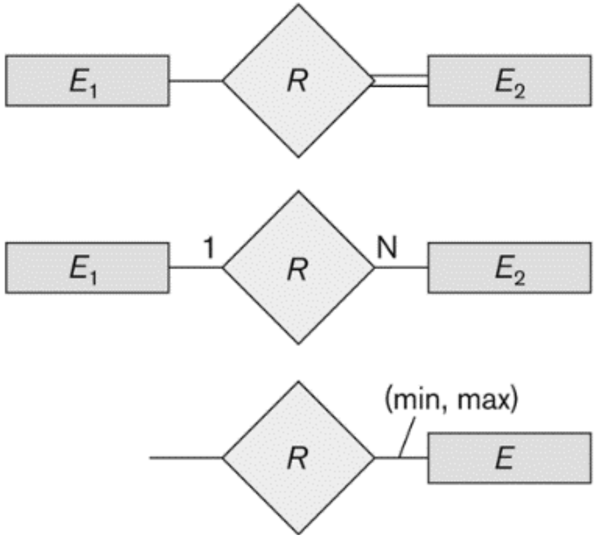
Relationships of the same type are grouped or typed into a relationship type.

Examples:

the WORKS_ON relationship type in which EMPLOYEEs and PROJECTs participate

the MANAGES relationship type in which EMPLOYEEs and DEPARTMENTs participate.

Relationship Notation



“Rotated square” connected by straight lines (no arrows)

Relationship Cardinalities

The cardinality of a relationship type is the number of participating entity types

1:1 **:** one entity on each side

1:N **:** “has many” relationship (N can also be 0!)

N:M **:** “many-to-many” relationship (N and M can be 0)

1:N

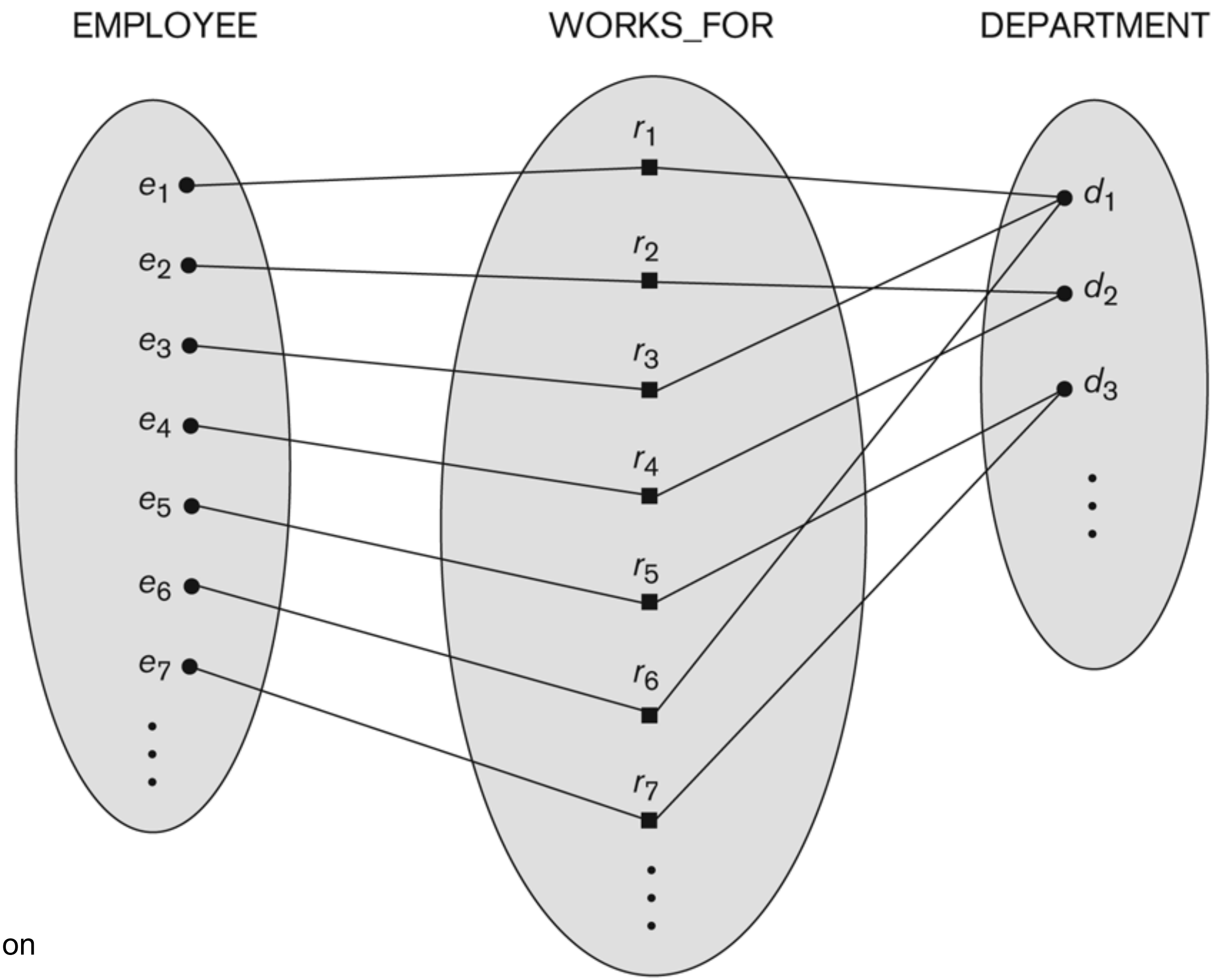


Figure 3.9
Some instances in the WORKS_FOR relationship set, which represents a relationship type WORKS_FOR between EMPLOYEE and DEPARTMENT.

N:M

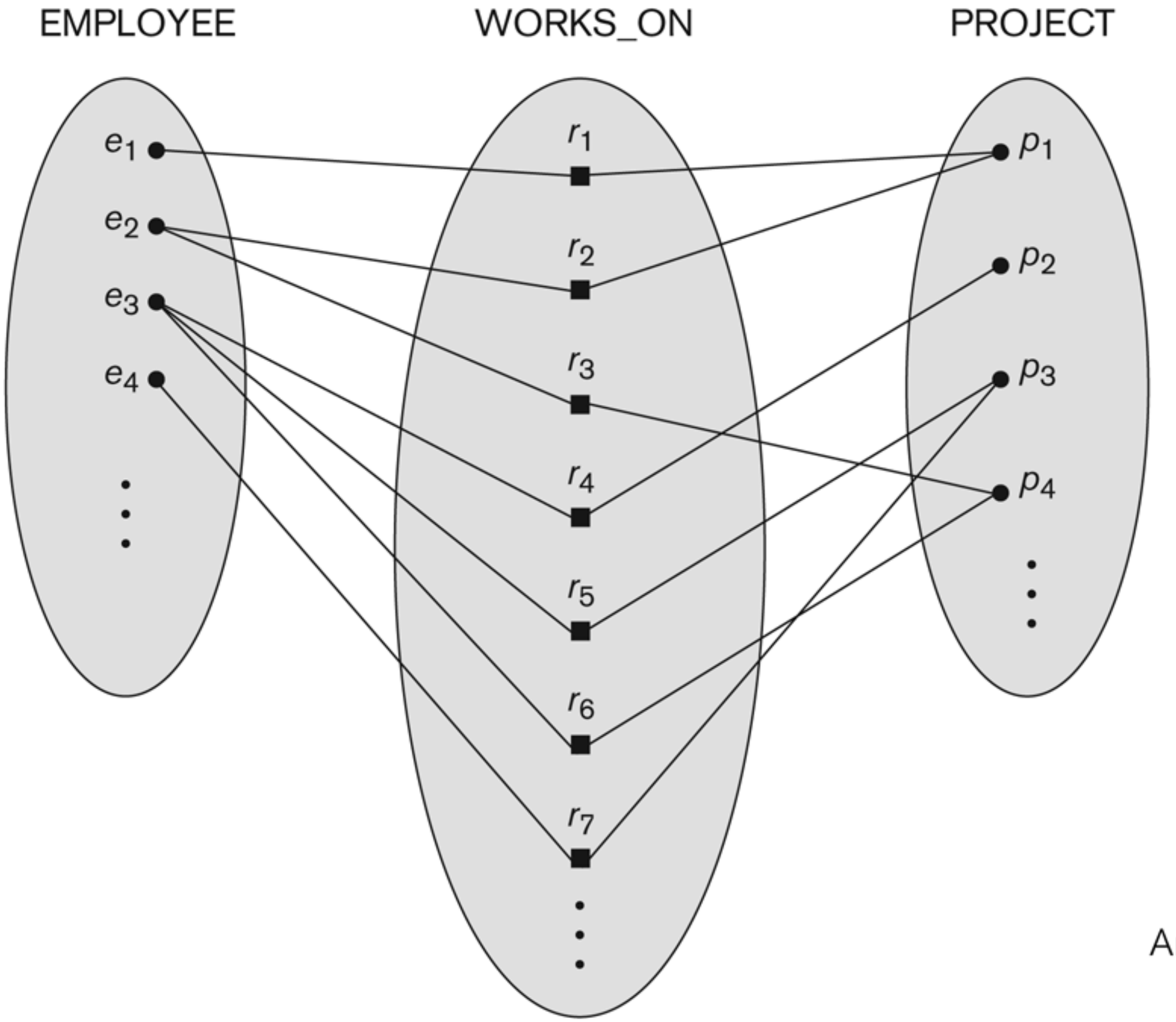


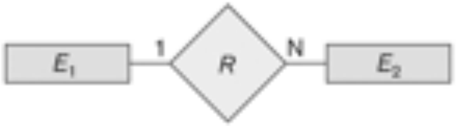
Figure 3.13
An M:N relationship, WORKS_ON.

Notation for Relationship Sets

Diamond-shaped box is used to display a relationship type

Connected to the participating entity types **via straight lines** (no arrow heads)

- By convention the name should be readable **from left to right and top to bottom**.



Constraints on Relationships

Cardinality (1:1, 1:N, N:1, M:N)

Shown by placing appropriate numbers on the relationship edges.

Participation constraint indicates whether relationship needs to exist for each concrete entity

Required (called total participation, double line)

Not Required (called partial participation, single line)

Let’s revisit our example yet again

Some of those attributes should probably be relationships

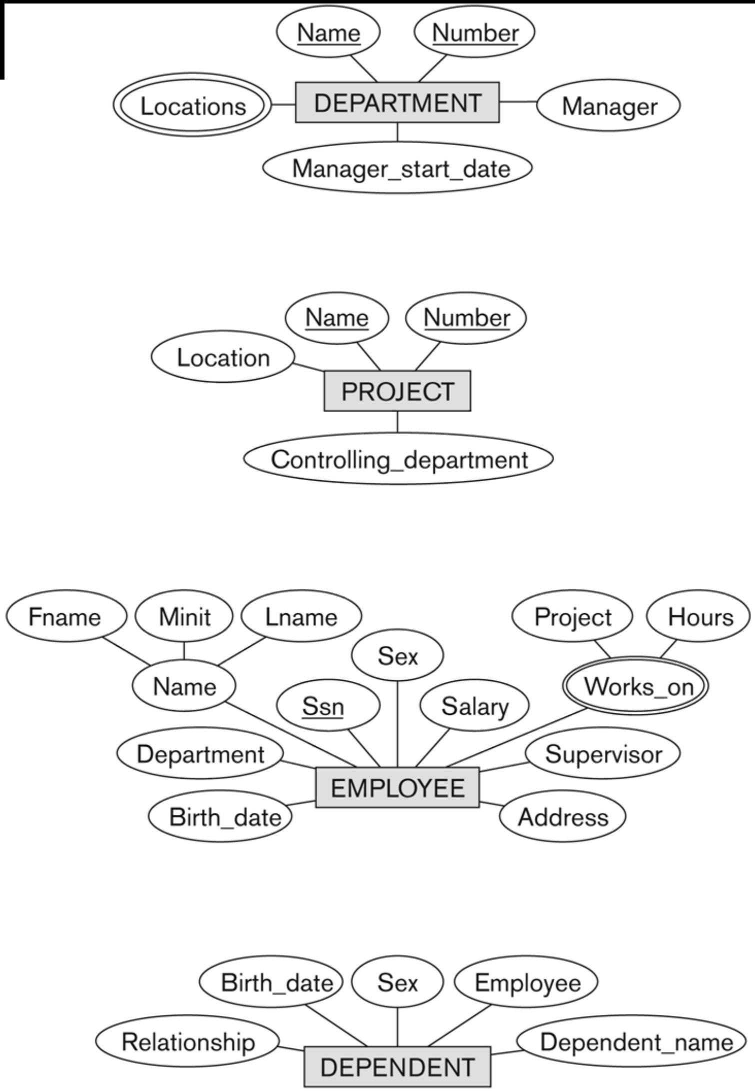


Figure 3.8
Preliminary design of entity types for the COMPANY database. Some of the shown attributes will be refined into relationships.

Manager of DEPARTMENT -> MANAGES

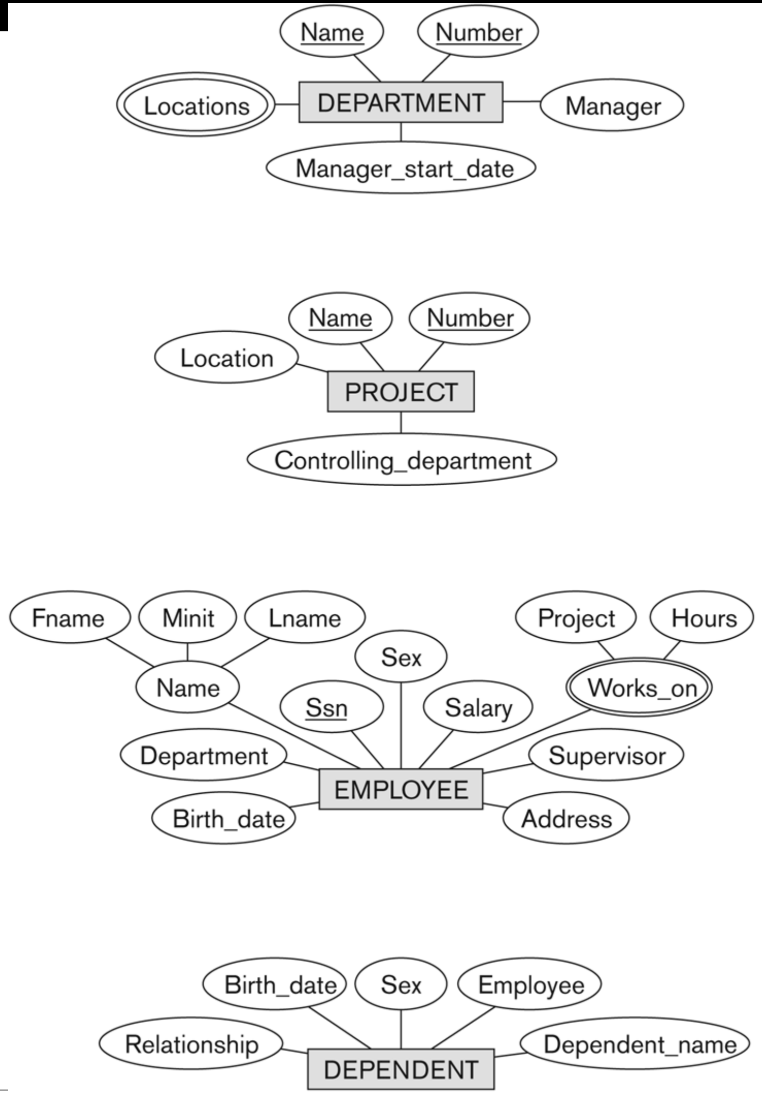
Department of EMPLOYEE -> WORKS_FOR

Works_on of EMPLOYEE -> WORKS_ON

Controlling_department of PROJECT -> CONTROLS

Employee of DEPENDENT -> DEPENDENTS_OF

Supervisor of EMPLOYEE -> SUPERVISION



Complete example

Don’t mind the few notations you don’t know yet

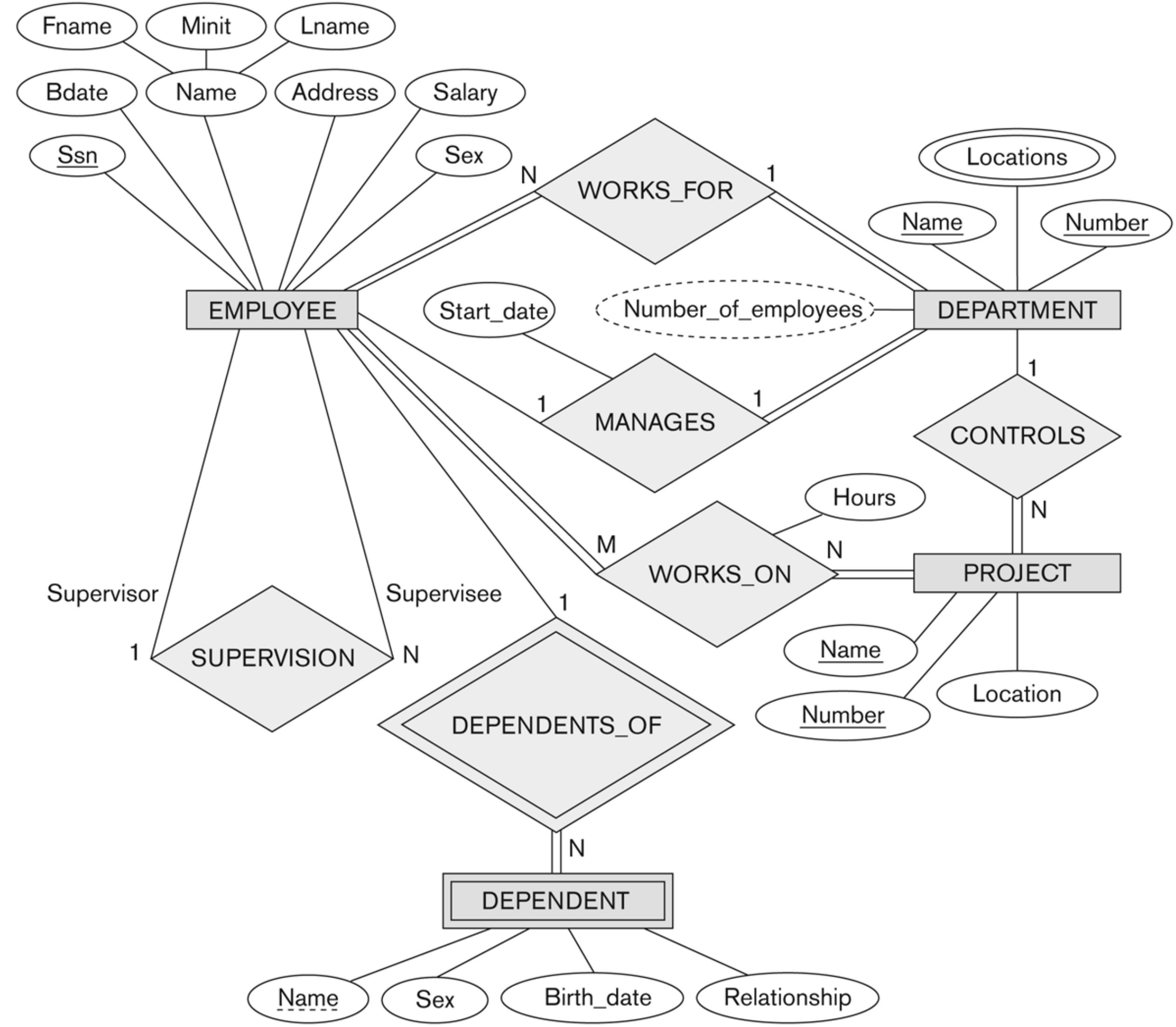


Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

Some Remarks

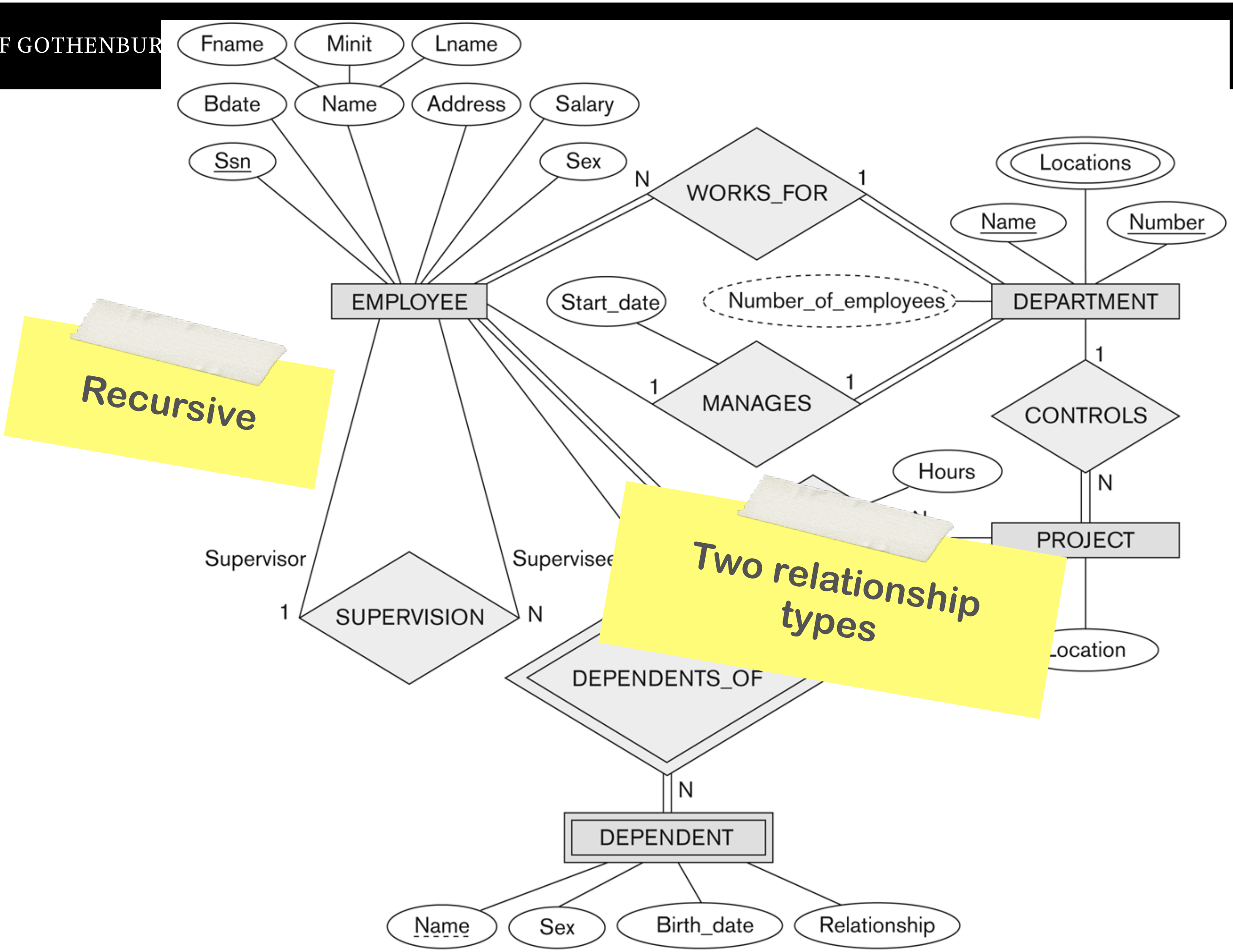
More than one relationship type can exist between two entity types.

MANAGES and WORKS_FOR are distinct relationship types!

Relationship types may be **recursive**

Both “ends” of the relationship are the same entity type

Requires a “role name” for both sides in the diagram



Recursive

Two relationship types

Copyright (c) 2011 Pearson Education

Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

Weak Entities

A **weak entity** does not have a key of it's own (it cannot be identified in the database)

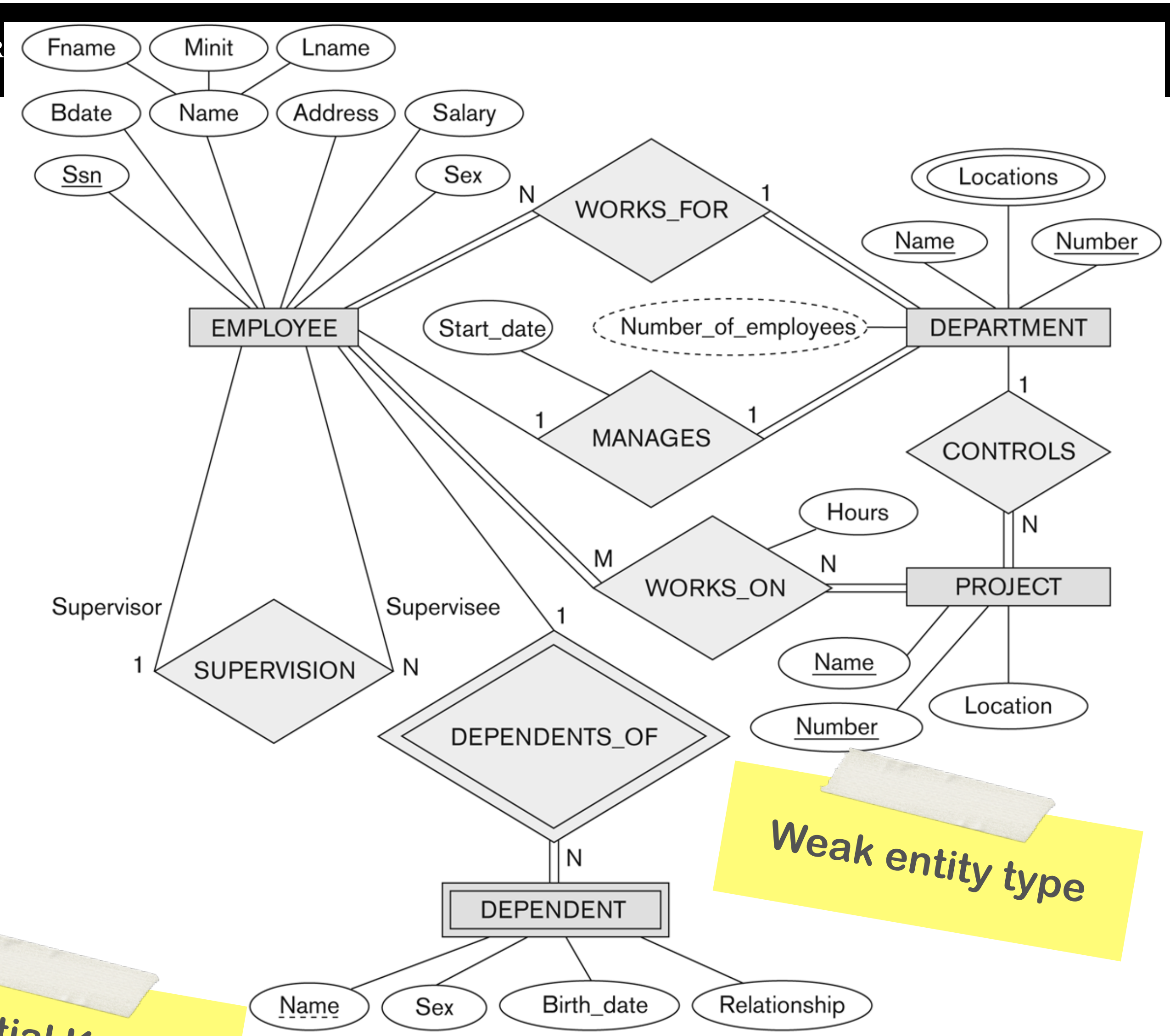
Instead it has an **owner** (a relationship with another entity that is not weak)

Weak entities are identified through a partial key and the owner

Notation:

Double-line entity symbol and association symbol

For partial key: **dashed underline**



Weak entity type

Partial Key

Copyright (c) 2011 Pearson Education

An E n for the COMPANY database. The diagrammatic notation is introduced gradua y throughout this chapter.

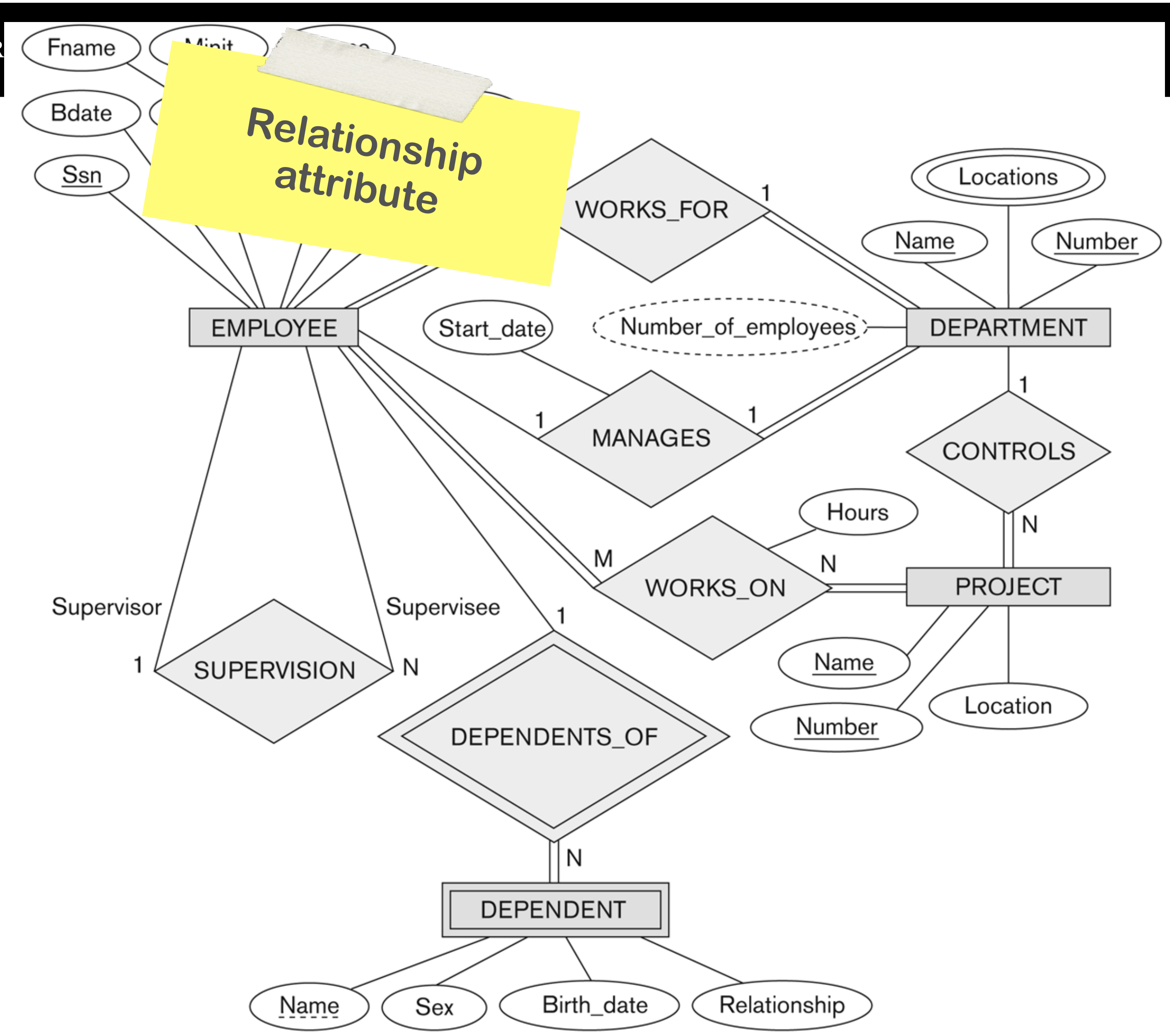
Attributes of Relationship Types

A relationship type can have attributes:

Example: Start_date of MANAGES

Value for each relationship instance describes the number of hours per week that an EMPLOYEE works on a PROJECT.

Each value depends on a particular (employee, project) **combination**



Relationship attribute

Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

Derived Attribute

Derived attributes are special in that they keep redundant information.

Their value can be **derived** (calculated from other information)

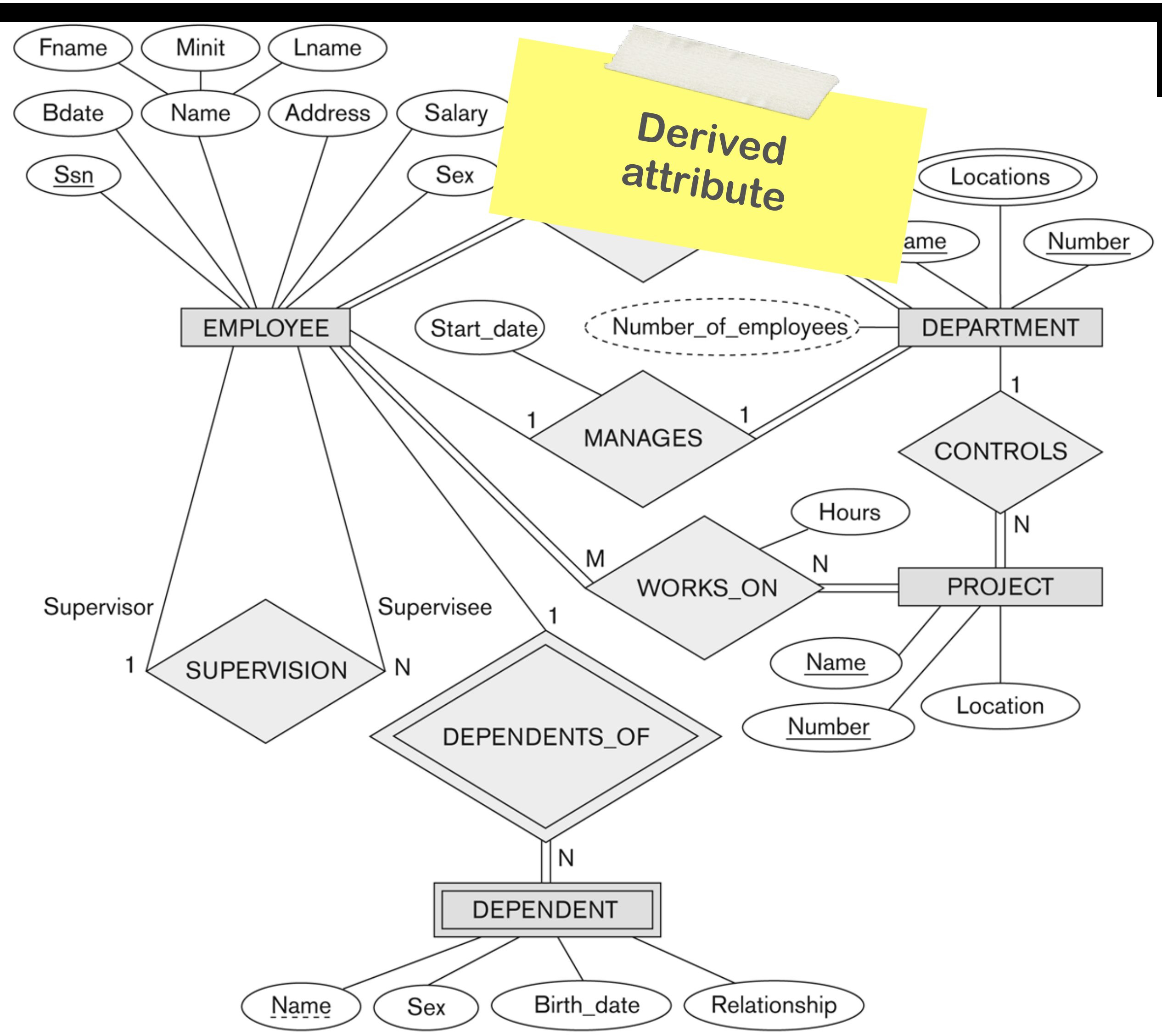
Example:

Nr_of_employees

Is just the number of EMPLOYEE entities in the entity set

Notation:

Attribute with **dashed line**



Derived attribute

Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

Ternary Relationships

In (rare) cases, you may want to model relationships between three (or more) entity types

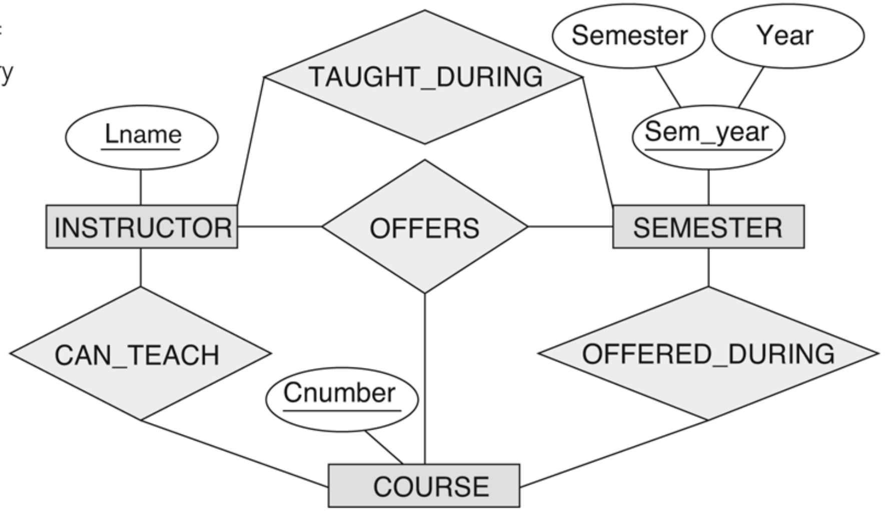
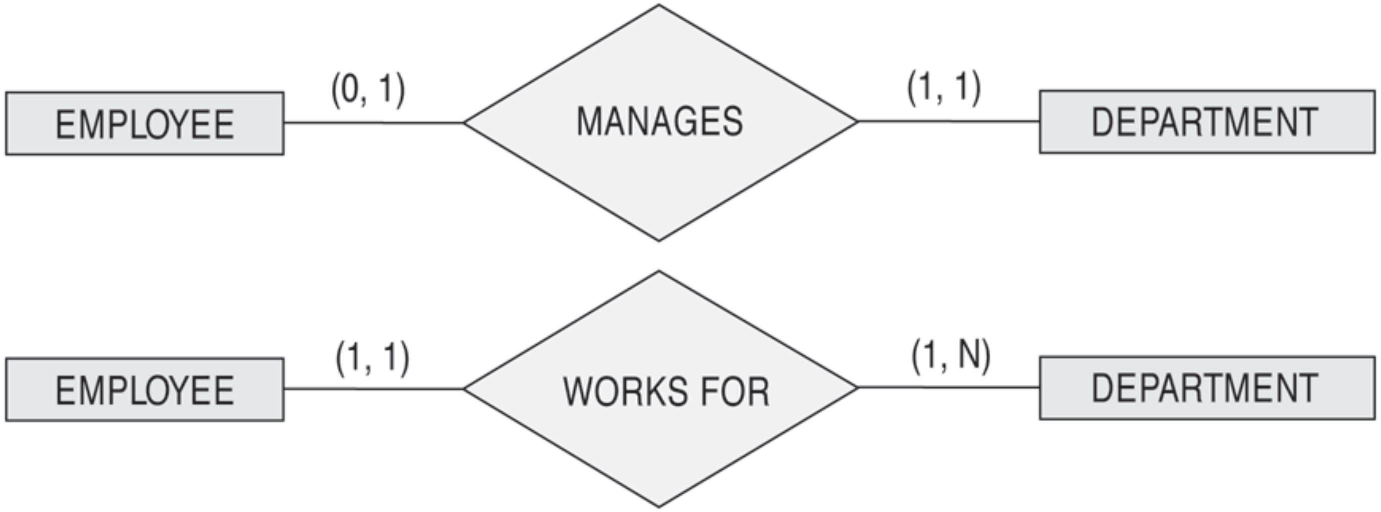


Figure 3.18
Another example of ternary versus binary relationship types.

(min,max) Notation: Alternative Notation for Cardinalities



Read the min,max numbers next to the entity type and looking **away from** the entity type

Example using (min,max) notation

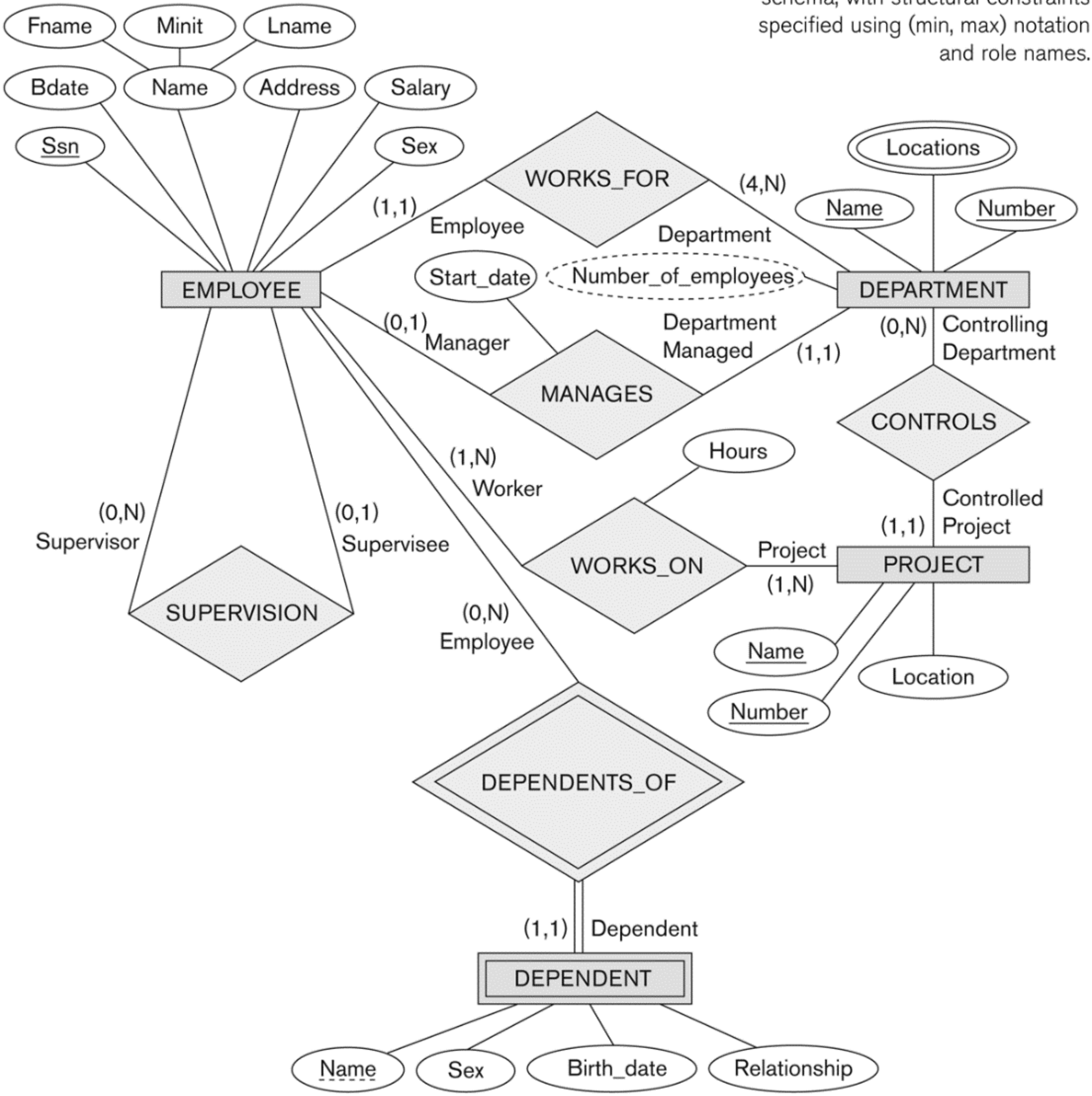


Figure 3.15
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.

Key Takeaways

Entities and Relationships

Be able to identify entity types and their basic relationships

Keys

This will be a fundamental concept that we fall back on a lot later

Cardinality

1:1, 1:N, N:M

Notation, notation, notation

Remember the basic notation of ER diagrams

Key Takeaways

More advanced concepts

- Composed keys and attributes
- Weak entity types
- Derived attributes
- Association attributes

Most of these concepts will re-surface at some point in your career as software engineers