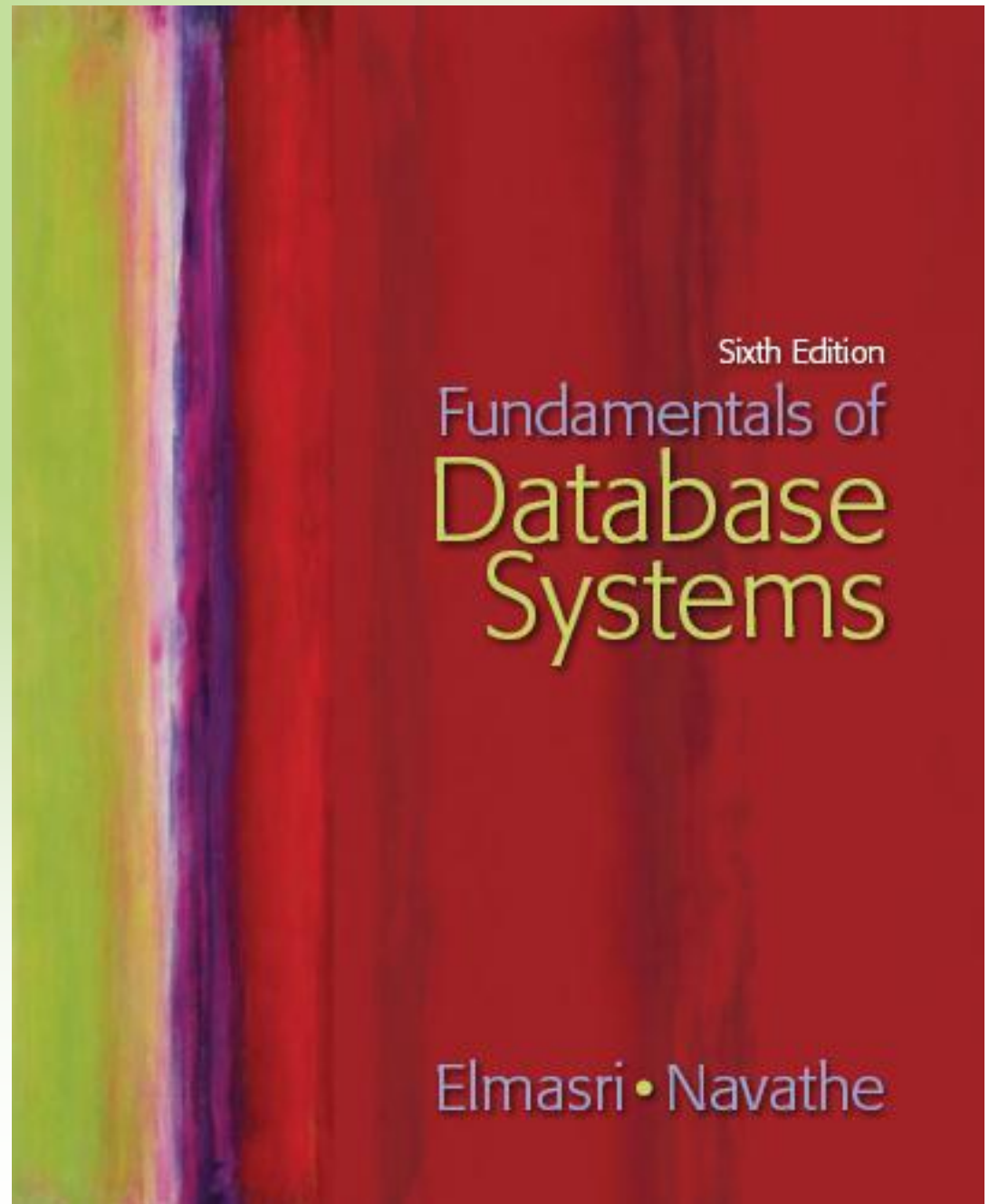


# Chapter 7

## Data Modeling Using the Entity- Relationship (ER) Model



Addison-Wesley  
is an imprint of

PEARSON

# Chapter 7 Outline

- Using High-Level Conceptual Data Models for Database Design
- A Sample Database Application
- Entity Types, Entity Sets, Attributes, and Keys
- Relationship Types, Relationship Sets, Roles, and Structural Constraints
- Weak Entity Types

# Chapter 7 Outline (cont'd.)

- Refining the ER Design for the COMPANY Database
- ER Diagrams, Naming Conventions, and Design Issues
- Example of Other Notation: UML Class Diagrams
- Relationship Types of Degree Higher than Two

# Data Modeling Using the Entity-Relationship (ER) Model

- **Entity-Relationship (ER) model**
  - Popular high-level conceptual data model
- **ER diagrams**
  - Diagrammatic notation associated with the ER model
- **Unified Modeling Language (UML)**

# Using High-Level Conceptual Data Models for Database Design

- **Requirements collection and analysis**
  - Database designers interview prospective database users to understand and document data requirements
  - Result: **data requirements**
  - **Functional requirements** of the application

# Using High-Level Conceptual Data Models (cont'd.)

- **Conceptual schema**
  - Conceptual design
  - Description of data requirements
  - Includes detailed descriptions of the entity types, relationships, and constraints
  - Transformed from high-level data model into implementation data model

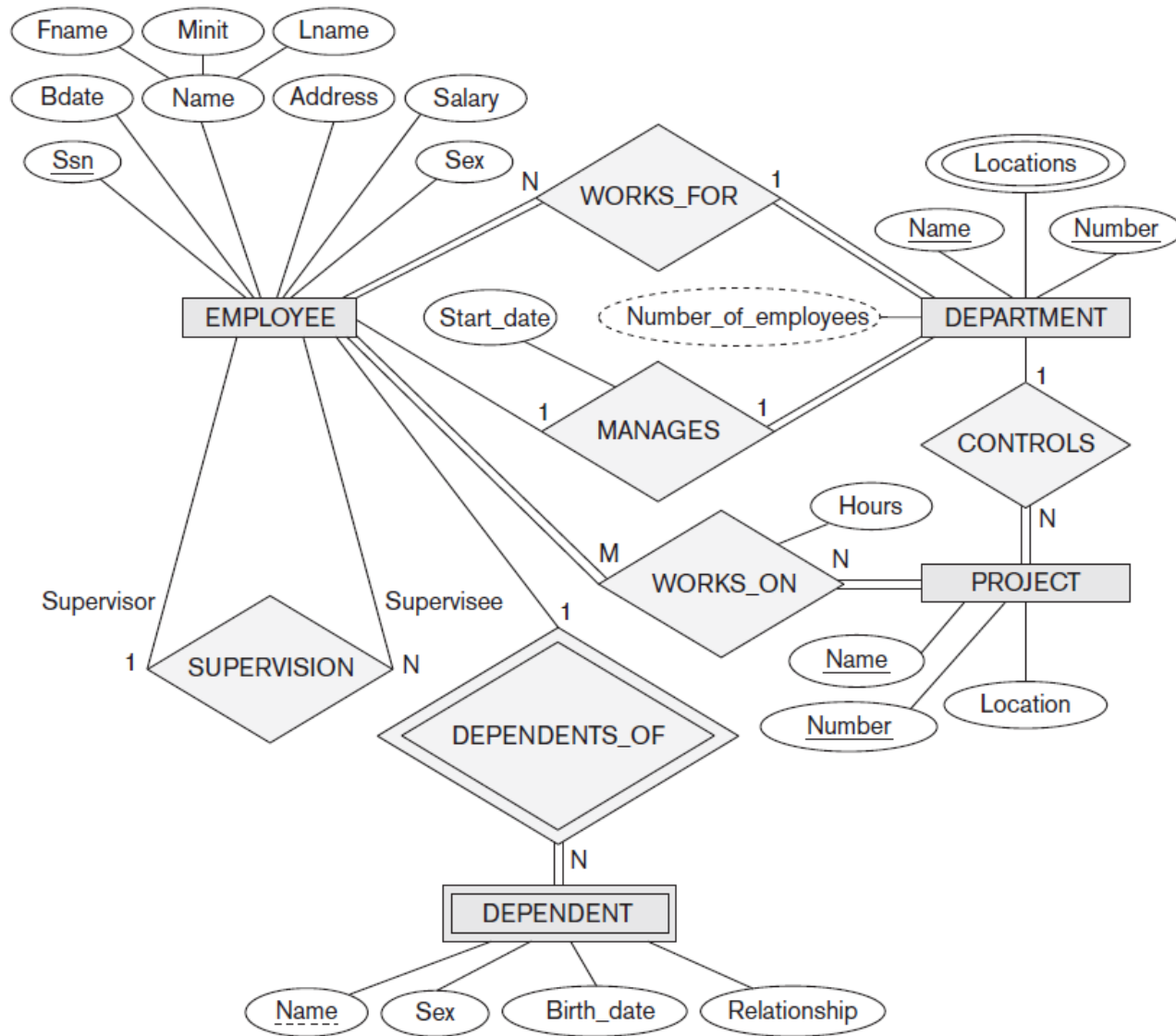
# Using High-Level Conceptual Data Models (cont'd.)

- **Logical design or data model mapping**
  - Result is a database schema in implementation data model of DBMS
- **Physical design phase**
  - Internal storage structures, file organizations, indexes, access paths, and physical design parameters for the database files specified

# A Sample Database Application

## ■ COMPANY

- Employees, departments, and projects
- Company is organized into departments
- Department controls a number of projects
- Employee: store each employee's name, Social Security number, address, salary, sex (gender), and birth date
- Keep track of the dependents of each employee



**Figure 7.2**

An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.

# Entity Types, Entity Sets, Attributes, and Keys

- ER model describes data as:
  - Entities
  - Relationships
  - Attributes

# Entities and Attributes

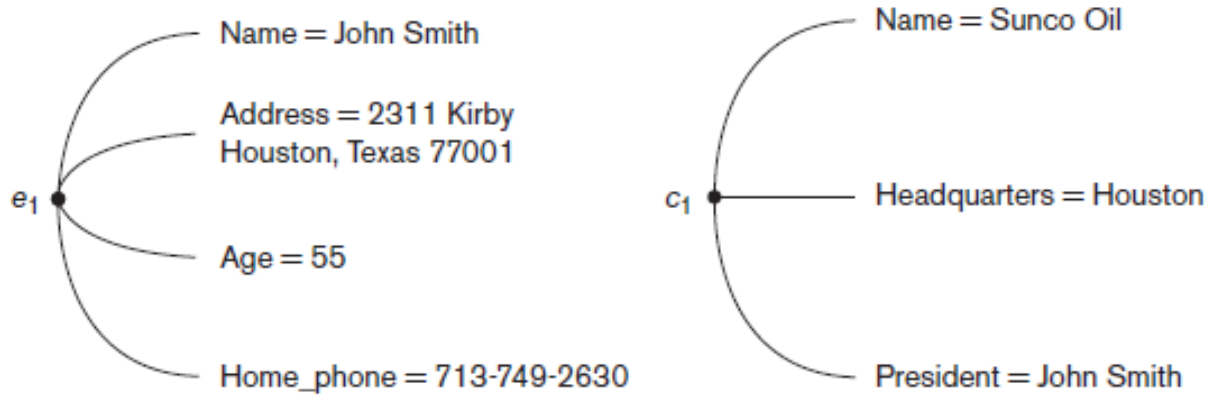
## ■ Entity

- Thing in real world with independent existence

## ■ Attributes

- Particular properties that describe entity
- Types of attributes:
  - *Composite versus simple (atomic) attributes*
  - **Single-valued** versus **multivalued** attributes
  - **Stored** versus **derived** attributes
  - **NULL** values
  - **Complex** attributes

# Entities and Attributes (cont'd.)

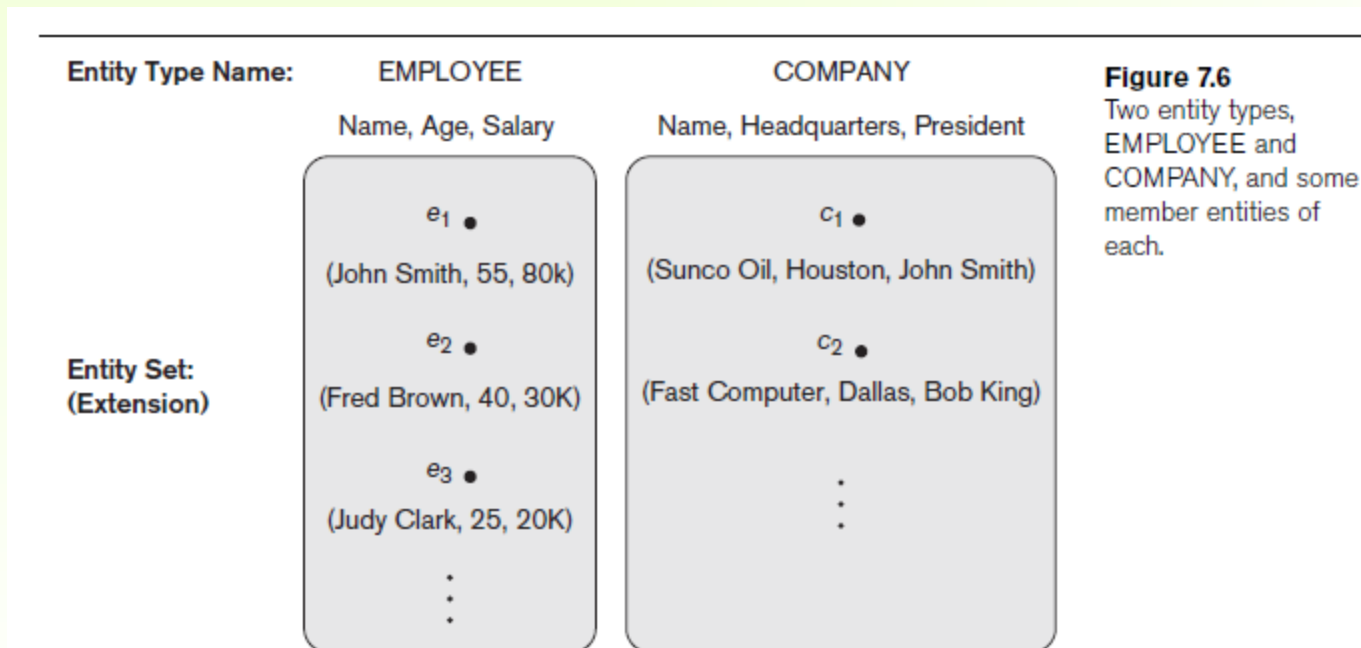


**Figure 7.3**  
Two entities,  
EMPLOYEE  $e_1$ , and  
COMPANY  $c_1$ , and  
their attributes.

# Entity Types, Entity Sets, Keys, and Value Sets

## ■ Entity type

- Collection (or set) of entities that have the same attributes



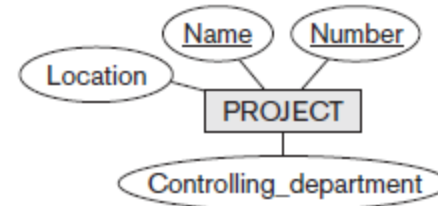
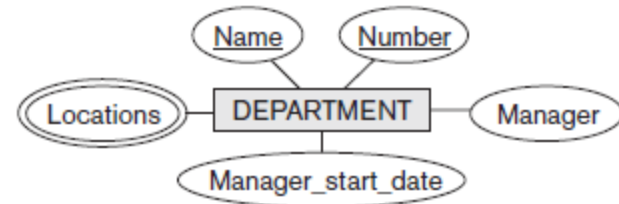
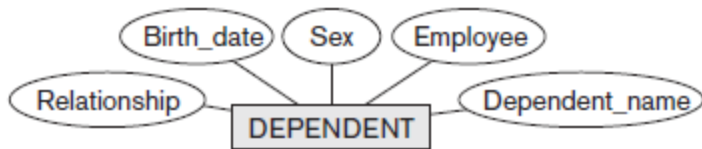
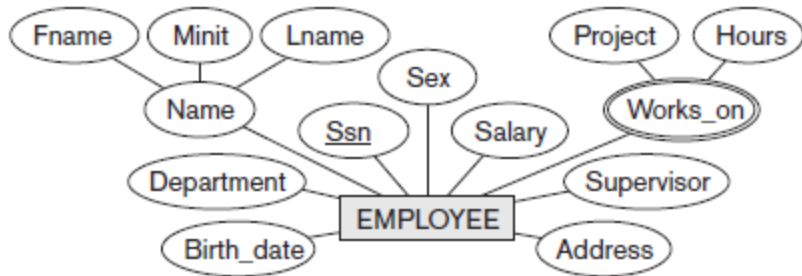
**Figure 7.6**

Two entity types, EMPLOYEE and COMPANY, and some member entities of each.

# Entity Types, Entity Sets, Keys, and Value Sets (cont'd.)

- **Key or uniqueness constraint**
  - Attributes whose values are distinct for each individual entity in entity set
  - **Key attribute**
    - Uniqueness property must hold for every entity set of the entity type
- **Value sets (or domain of values)**
  - Specifies set of values that may be assigned to that attribute for each individual entity

# Initial Conceptual Design of the COMPANY Database



**Figure 7.8**

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

# Relationship Types, Relationship Sets, Roles, and Structural Constraints

## ■ Relationship

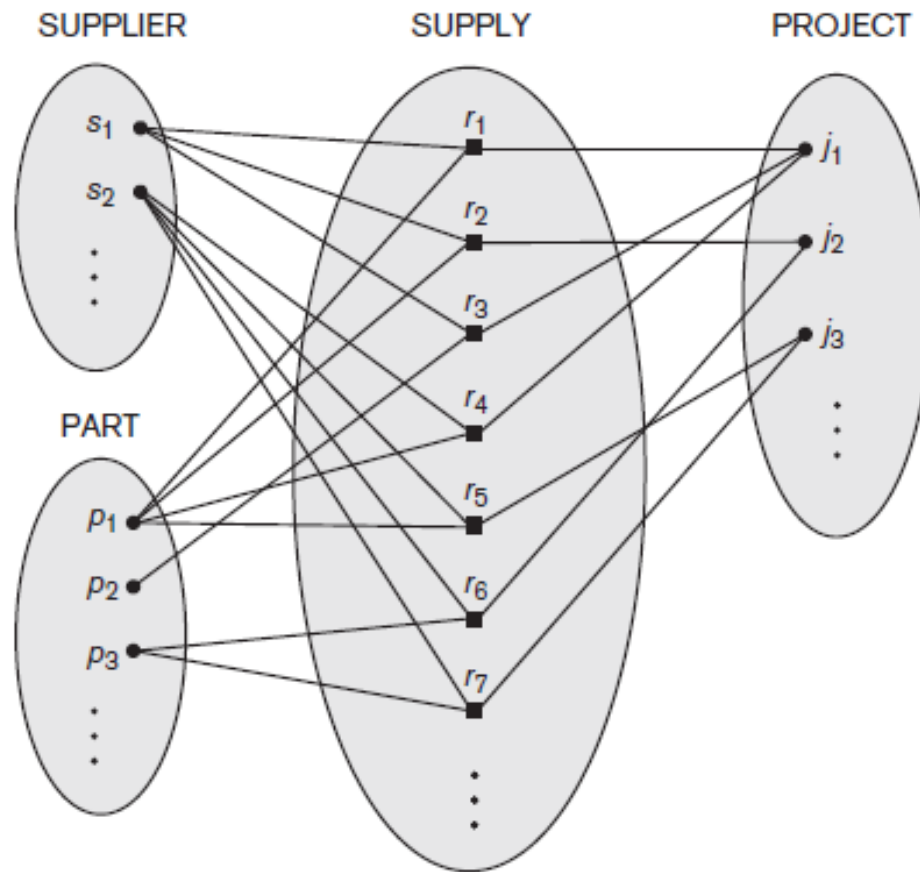
- When an attribute of one entity type refers to another entity type
- Represent references as relationships not attributes

# Relationship Types, Sets, and Instances

- **Relationship type**  $R$  among  $n$  entity types  $E_1, E_2, \dots, E_n$ 
  - Defines a set of associations among entities from these entity types
- **Relationship instances**  $r_i$ 
  - Each  $r_i$  associates  $n$  individual entities ( $e_1, e_2, \dots, e_n$ )
  - Each entity  $e_j$  in  $r_i$  is a member of entity set  $E_j$

# Relationship Degree

- **Degree** of a relationship type
  - Number of participating entity types
  - **Binary, ternary**
- Relationships as attributes
  - Think of a binary relationship type in terms of attributes

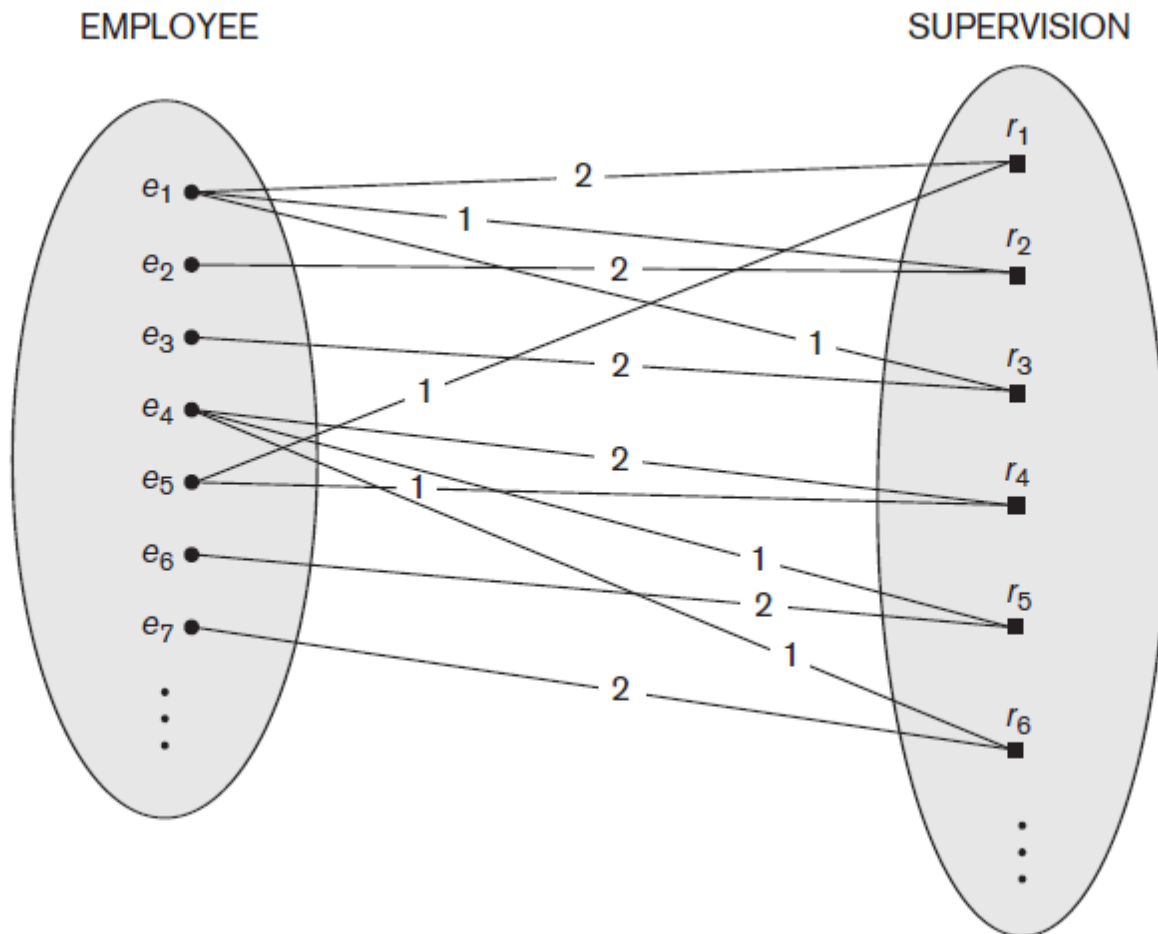


**Figure 7.10**

Some relationship instances in the SUPPLY ternary relationship set.

# Role Names and Recursive Relationships

- **Role names** and recursive relationships
  - Role name signifies role that a participating entity plays in each relationship instance
- **Recursive** relationships
  - Same entity type participates more than once in a relationship type in different roles
  - Must specify role name



**Figure 7.11**

A recursive relationship SUPERVISION between EMPLOYEE in the *supervisor* role (1) and EMPLOYEE in the *subordinate* role (2).

# Constraints on Binary Relationship Types

- **Cardinality ratio** for a binary relationship
  - Specifies maximum number of relationship instances that entity can participate in
- **Participation constraint**
  - Specifies whether existence of entity depends on its being related to another entity
  - Types: **total** and **partial**

# Attributes of Relationship Types

- Attributes of 1:1 or 1:N relationship types can be migrated to one entity type
- For a 1:N relationship type
  - Relationship attribute can be migrated only to entity type on N-side of relationship
- For M:N relationship types
  - Some attributes may be determined by combination of participating entities
  - Must be specified as relationship attributes

# Weak Entity Types

- Do not have key attributes of their own
  - Identified by being related to specific entities from another entity type
- **Identifying relationship**
  - Relates a weak entity type to its owner
- Always has a total participation constraint

# Refining the ER Design for the COMPANY Database

- Change attributes that represent relationships into relationship types
- Determine cardinality ratio and participation constraint of each relationship type

# ER Diagrams, Naming Conventions, and Design Issues

Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identifying 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 $R$

**Figure 7.14**  
Summary of the notation for ER diagrams.

# Proper Naming of Schema Constructs

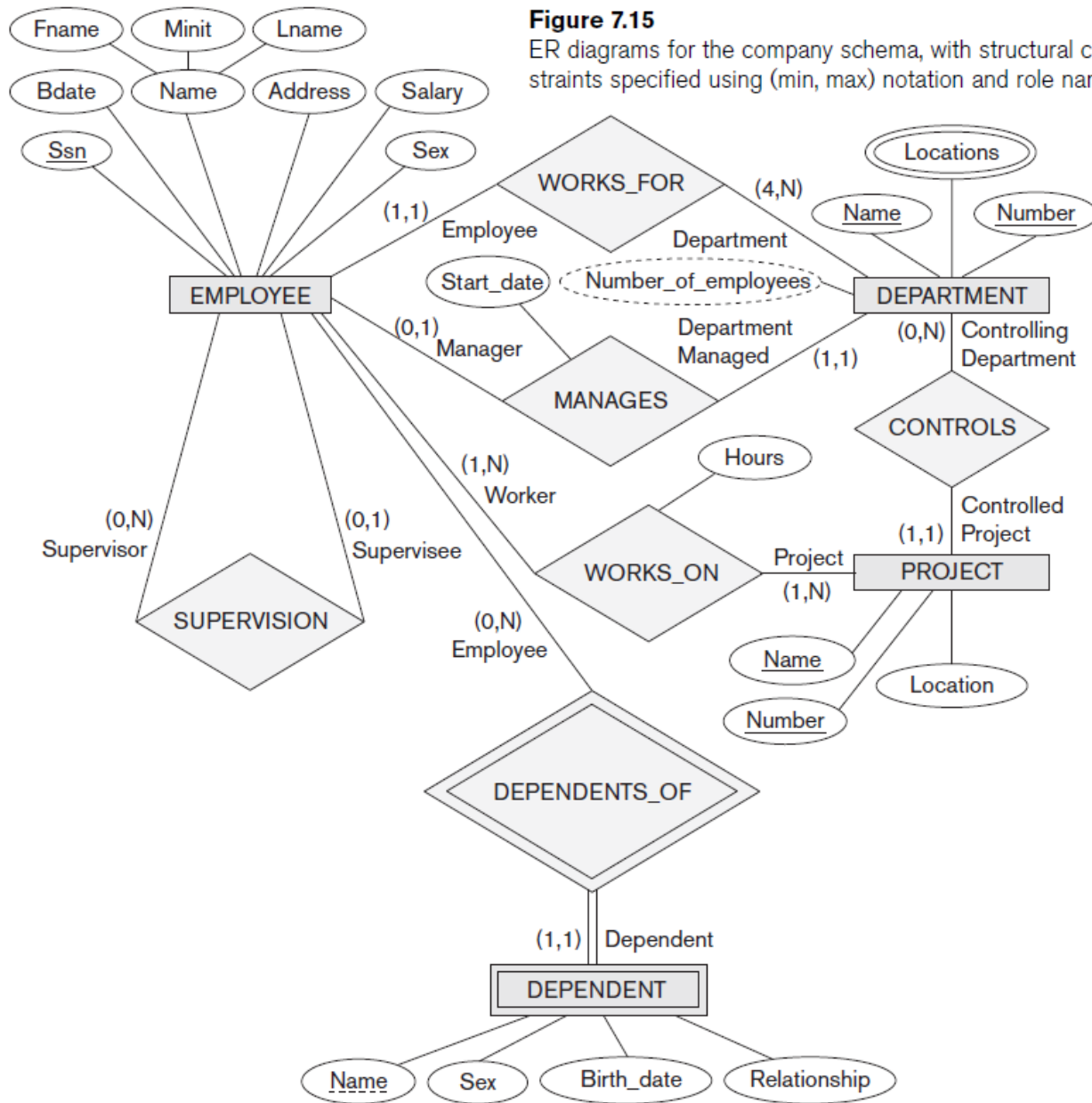
- Choose names that convey meanings attached to different constructs in schema
- Nouns give rise to entity type names
- Verbs indicate names of relationship types
- Choose binary relationship names to make ER diagram readable from left to right and from top to bottom

# Design Choices for ER Conceptual Design

- Model concept first as an attribute
  - Refined into a relationship if attribute is a reference to another entity type
- Attribute that exists in several entity types may be elevated to an independent entity type
  - Can also be applied in the inverse

# Alternative Notations for ER Diagrams

- Specify structural constraints on relationships
  - Replaces cardinality ratio (1:1, 1:N, M:N) and single/double line notation for participation constraints
  - Associate a pair of integer numbers (min, max) with each participation of an entity type  $E$  in a relationship type  $R$ , where  $0 \leq \min \leq \max$  and  $\max \geq 1$



**Figure 7.15**

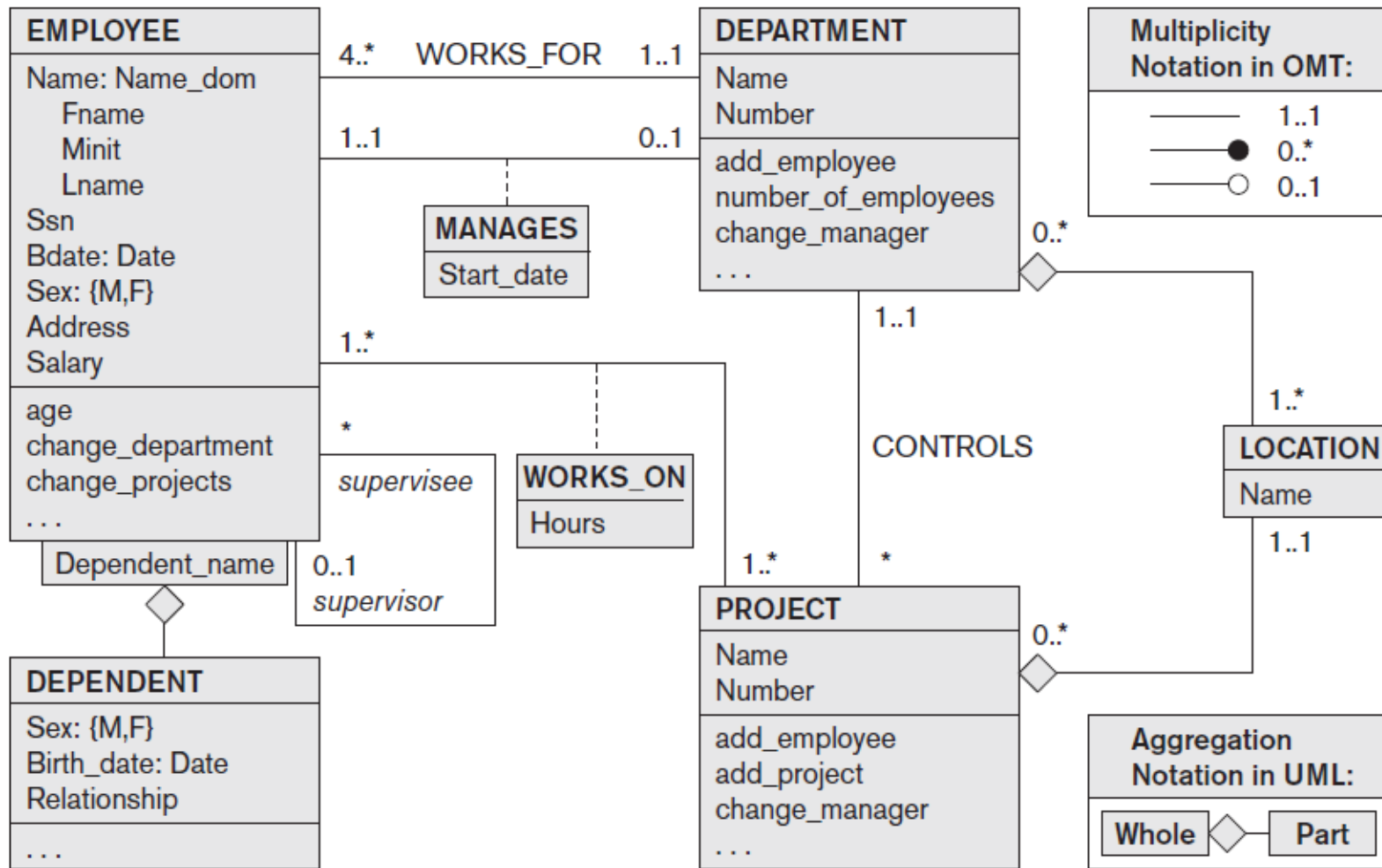
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.

# Example of Other Notation: UML Class Diagrams

- UML methodology
  - Used extensively in software design
  - Many types of diagrams for various software design purposes
- UML class diagrams
  - Entity in ER corresponds to an object in UML

**Figure 7.16**

The COMPANY conceptual schema in UML class diagram notation.



# Example of Other Notation: UML Class Diagrams (cont'd.)

- **Class** includes three sections:
  - Top section gives the class name
  - Middle section includes the attributes;
  - Last section includes operations that can be applied to individual objects

# Example of Other Notation: UML Class Diagrams (cont'd.)

- **Associations:** relationship types
- **Relationship instances:** links
- Binary association
  - Represented as a line connecting participating classes
  - May optionally have a name
- Link attribute
  - Placed in a box connected to the association's line by a dashed line

# Example of Other Notation: UML Class Diagrams (cont'd.)

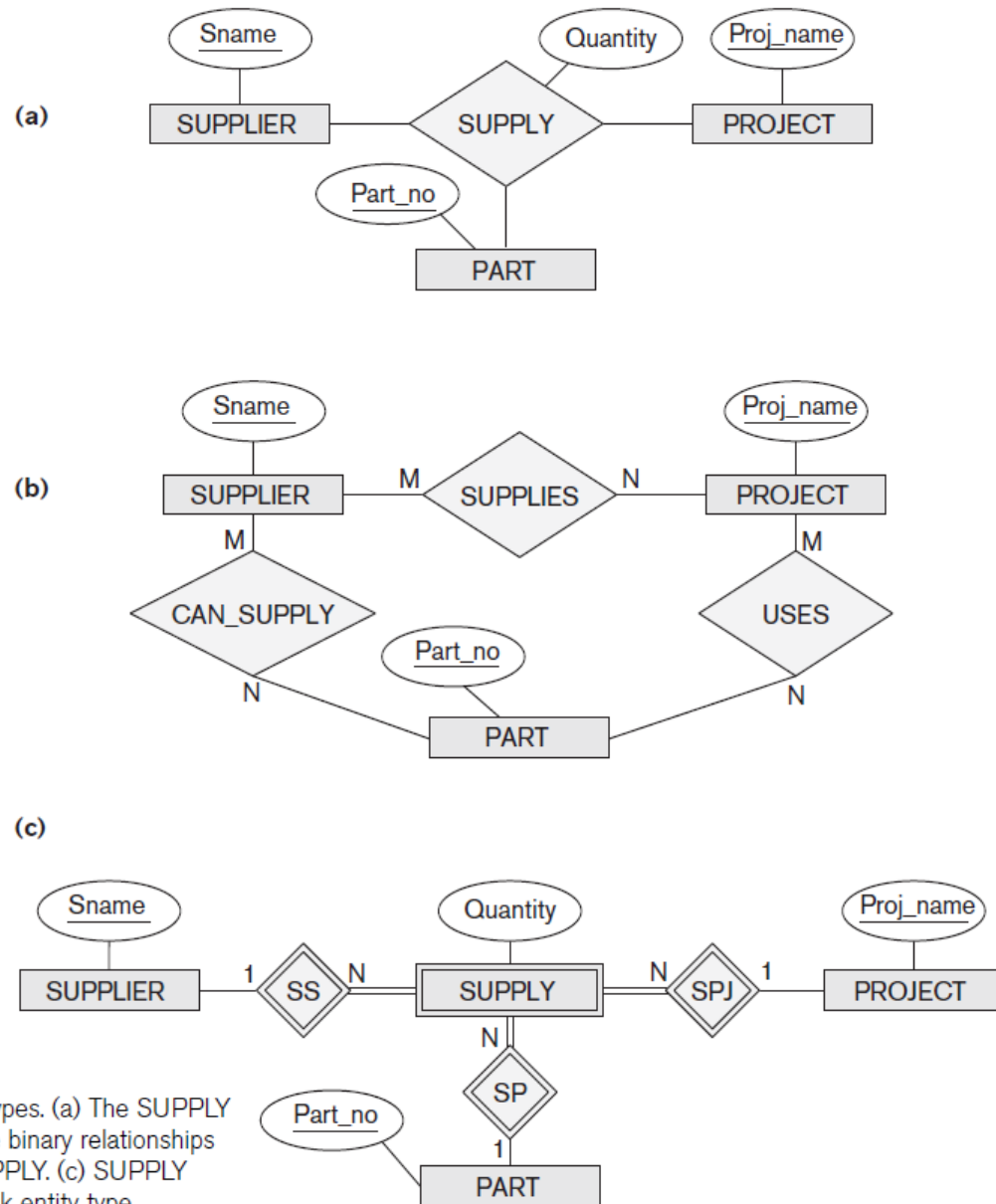
- **Multiplicities:** min..max, asterisk (\*) indicates no maximum limit on participation
- Types of relationships: **association** and **aggregation**
- Distinguish between **unidirectional** and **bidirectional** associations
- Model weak entities using **qualified association**

# Relationship Types of Degree Higher than Two

- **Degree** of a relationship type
  - Number of participating entity types
- *Binary*
  - Relationship type of degree two
- *Ternary*
  - Relationship type of degree three

# Choosing between Binary and Ternary (or Higher-Degree) Relationships

- Some database design tools permit only binary relationships
  - Ternary relationship must be represented as a weak entity type
  - No partial key and three identifying relationships
- Represent ternary relationship as a regular entity type
  - By introducing an artificial or surrogate key



**Figure 7.17**

Ternary relationship types. (a) The SUPPLY relationship. (b) Three binary relationships not equivalent to SUPPLY. (c) SUPPLY represented as a weak entity type.

# Constraints on Ternary (or Higher-Degree) Relationships

- Notations for specifying structural constraints on  $n$ -ary relationships
  - Should both be used if it is important to fully specify structural constraints

# Summary

- Basic ER model concepts of entities and their attributes
  - Different types of attributes
  - Structural constraints on relationships
- ER diagrams represent E-R schemas
- UML class diagrams relate to ER modeling concepts