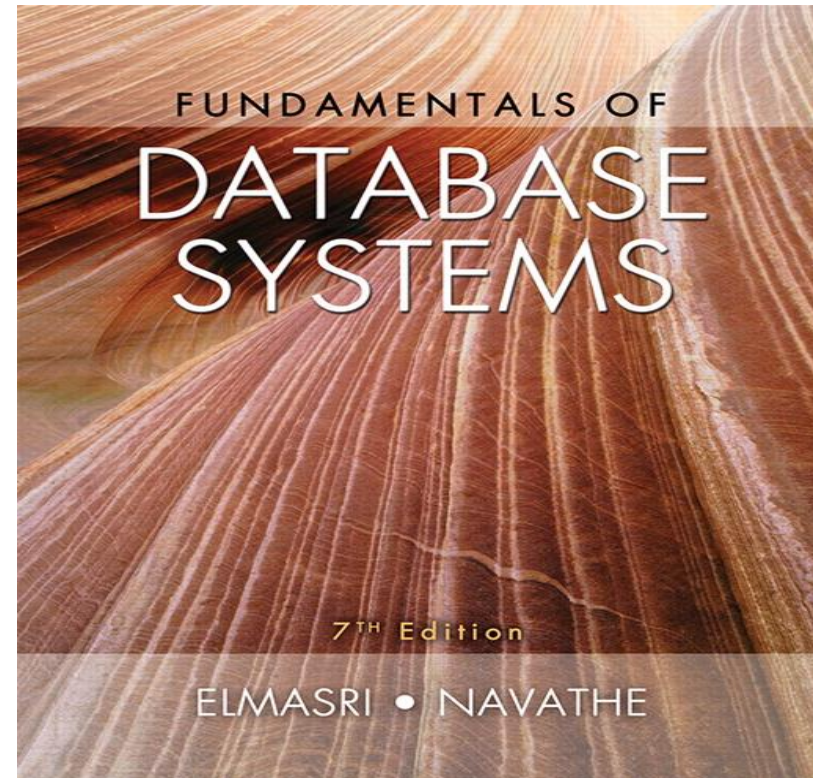


# Comp-3150: Database Management Systems

- Ramez Elmasri , Shamkant B. Navathe(2016) Fundamentals of Database Systems (7th Edition), Pearson, isbn 10: 0-13-397077-9; isbn-13:978-0-13-397077-7.
- Chapter 3:
- Data Modeling Using the Entity-Relationship (ER) Model



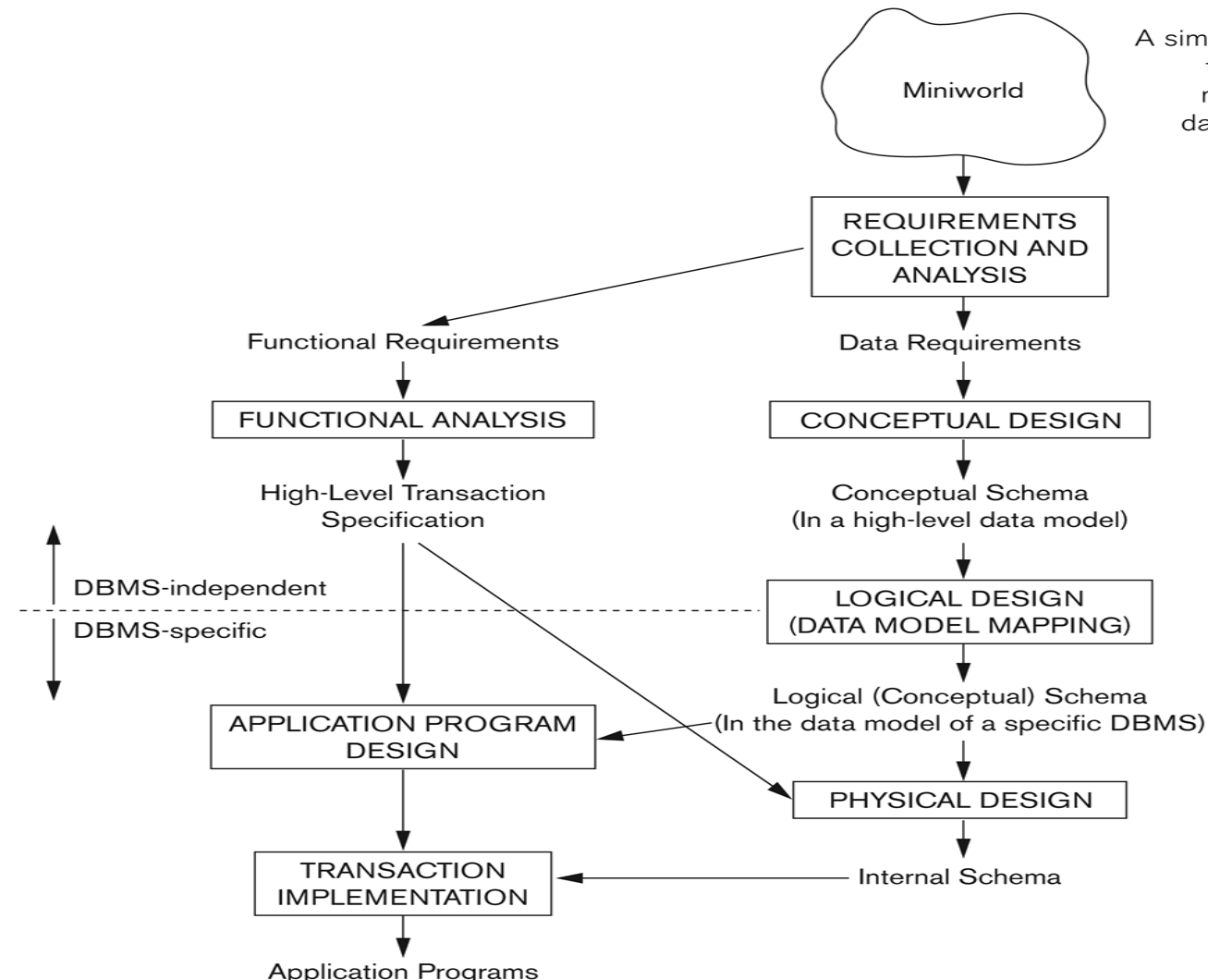
# Chapter 3: Data Modeling Using the Entity-Relationship (ER) Model

## Outline

- 1. Overview of Database Design Process
- 2. Example Database Application (COMPANY)
- 3. ER Model Concepts
  - Entities and Attributes
  - Entity Types, Value Sets, and Key Attributes
  - Relationships and Relationship Types
  - Weak Entity Types
  - Roles and Attributes in Relationship Types
- 4. ER Diagrams - Notation
- 5. ER Diagram for COMPANY Schema
- 6. Alternative Notations – UML class diagrams, others
- 7. Relationships of Higher Degree

# 1. Overview of Database Design Process

- Fig 3.1 shows an overview of DB design process



**Figure 3.1**

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

# 1. Overview of Database Design Process

- Two main activities in the design process (as in Fig. 3.1):
  - Database design (right side of Fig. 3.1)
  - Applications design (left side of Fig. 3.1)
- Database design goes through the 4 phases of:
  - (1) requirements analysis
  - (2) conceptual database schema design
  - (3) implementation with a commercial DBMS
  - (4) physical database design

# 1. Overview of Database Design Process

- Focus in this chapter is on conceptual database design
  - To design the conceptual schema for a database application
- Applications design focuses on the programs and interfaces that access the database (not the focus in this chapter)
  - Generally considered part of software engineering

# 1. Overview of Database Design Process

- Methodologies for Conceptual Design include:
  - Entity Relationship (ER) Diagrams (This Chapter)
  - Enhanced Entity Relationship (EER) Diagrams (Chapter 4)
  - Use of Design Tools in industry for designing and documenting large scale designs
  - The UML (Unified Modeling Language) Class Diagrams are popular in industry to document conceptual database designs

## 2. Example Database Application (COMPANY)

- We need to create a database schema design based on the following (simplified) **requirements** of the COMPANY Database:
  - 1. 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.
  - 2. Each department *controls* a number of PROJECTs. Each project has a unique name, unique number and is located at a single location.

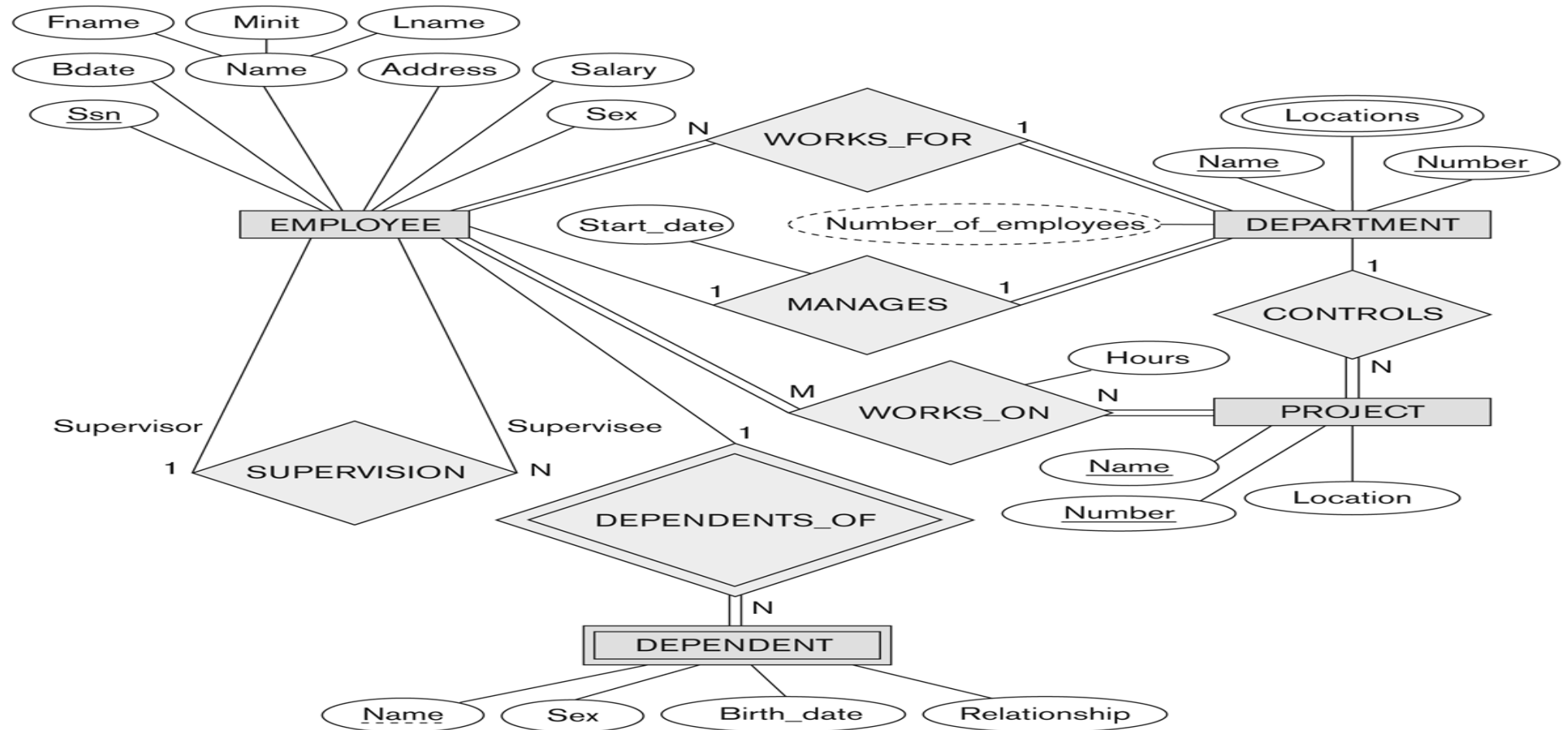
## 2. Example Database Application (COMPANY)

- 3. The database will store each EMPLOYEE's name, social security number, address, salary, sex, birthdate, department and supervisor.
  - 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.
- 4. 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.



## 2. Example Database Application (COMPANY)

- Fig. 3.2 shows the conceptual schema of the COMPANY db using the ER diagram.



**Figure 3.2**

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

# 3. ER Model Concepts

- ER Model Concepts Used to design schema of Fig. 3.2
- **Entities and Attributes**
  - 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.
    - For example the EMPLOYEE John Smith, the Research DEPARTMENT, the ProductX PROJECT
  - Attributes are properties used to describe an entity.
    - For example an EMPLOYEE entity may have the attributes Name, SSN, Address, Sex, BirthDate
  - A specific entity will have a value for each of its attributes.
    - For example a specific employee entity may have Name='John Smith', SSN='123456789', Address ='731, Fondren, Houston, TX', Sex='M', BirthDate='09-JAN-55'
  - Each attribute has a *value set* (or data type) associated with it – e.g. integer, string, date, enumerated type, ...

# 3. ER Model Concepts

- Types of Attributes (1)

- Simple

- Each entity has a single atomic value for the attribute. For example, SSN or Sex.

- Composite

- The attribute may be composed of several components. For example:
  - Address(Apt#, House#, Street, City, State, ZipCode, Country), or
  - Name(FirstName, MiddleName, LastName).
  - Composition may form a hierarchy where some components are themselves composite.

- Multi-valued

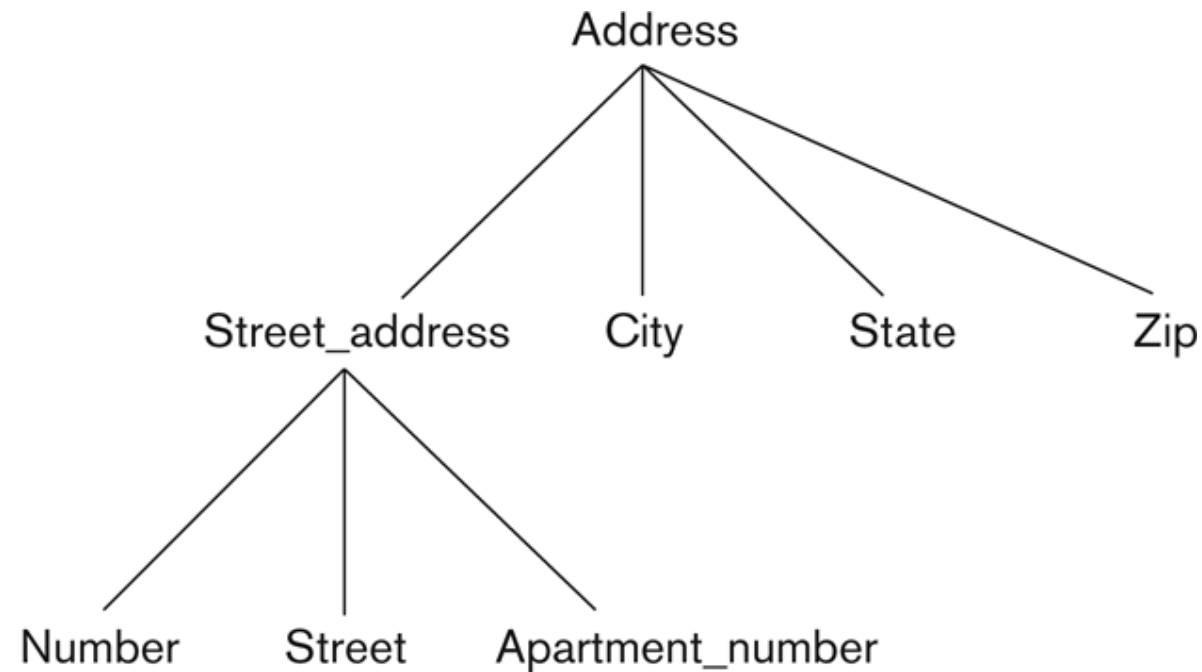
- An entity may have multiple values for that attribute. For example, Color of a CAR or PreviousDegrees of a STUDENT.
  - Denoted as {Color} or {PreviousDegrees}.

### 3. ER Model Concepts

- Types of Attributes (2)
  - In general, composite and multi-valued attributes may be nested arbitrarily to any number of levels, although this is rare.
    - For example, PreviousDegrees of a STUDENT is a composite multi-valued attribute denoted by {PreviousDegrees (College, Year, Degree, Field)}
    - Multiple PreviousDegrees values can exist
    - Each has four subcomponent attributes:
      - College, Year, Degree, Field

# 3. ER Model Concepts

An Example of Composite Attributes (Address and Street\_address)



**Figure 3.4**

A hierarchy of composite attributes.

### 3. ER Model Concepts

- Entity Types and Key Attributes (1)
  - Entities with the same basic attributes are grouped or typed into an entity type.
    - For 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.
    - For example, SSN of EMPLOYEE.

### 3. ER Model Concepts

- Entity Types and Key Attributes (2)
  - A key attribute may be composite.
    - VehicleTagNumber is a key of the CAR entity type with components (Number, State).
  - An entity type may have more than one key.
    - The CAR entity type may have two keys:
      - VehicleIdentificationNumber (popularly called VIN)
      - VehicleTagNumber (Number, State), aka license plate number.
  - Each key is underlined (Note: this is different from the relational schema where only one “primary key is underlined”).

### 3. ER Model Concepts

- Entity Set
- Each entity type will have a collection of entities stored in the database
  - Called the **entity set** or sometimes **entity collection**
- Entity set is the current *state* of the entities of that type that are stored in the database



### 3. ER Model Concepts



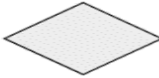
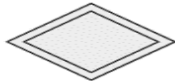







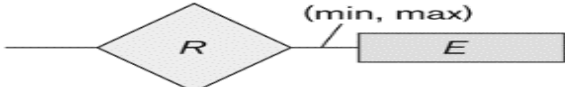
- Value Sets (Domains) of Attributes
  - Each simple attribute is associated with a value set
    - E.g., Lastname has a value which is a character string of up to 15 characters, say
    - Date has a value consisting of MM-DD-YYYY where each letter is an integer
  - A **value set** specifies the set of values that an attribute of each individual entity of this type may be assigned. For example, the domain of age can be 16 to 70 for each EMPLOYEE.

### 3. ER Model Concepts

- Displaying an Entity type
- In ER diagrams, an entity type is displayed in a rectangular box
- Attributes are displayed in ovals
  - Each attribute is connected to its entity type
  - Components of a composite attribute are connected to the oval representing the composite attribute
  - Each key attribute is underlined
  - Multivalued attributes displayed in double ovals
- See the full ER notation in advance on the next slide (Fig 3.14)

# 4. ER Diagrams - Notation

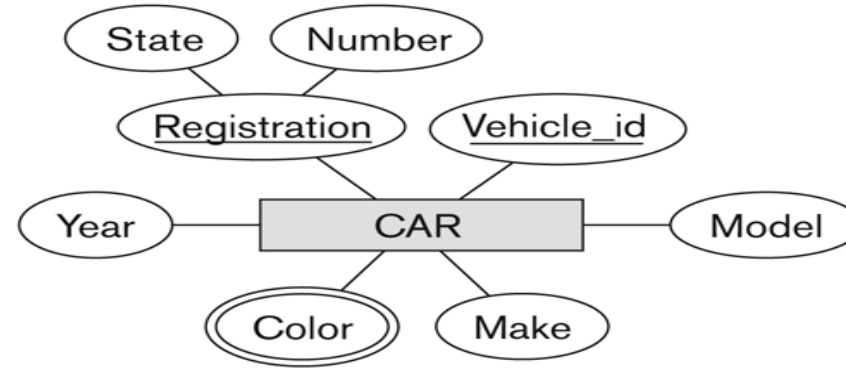
**Figure 3.14**  
Summary of the  
notation for ER  
diagrams.

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 $R$

1. Each E2 record must relate to An E1 record for total participation
2. Each E1 record can relate to up to N records but each E2 record must relate to only 1 E1 record.
3. Each E1 record can relate to a minimum of 0 and maximum of N E2 records. Each E2 record is related to 0 to 1 E1 records.

# Entity Type CAR with two keys and a corresponding Entity Set

(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<sub>1</sub>  
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 2004 {red, black})

CAR<sub>2</sub>  
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})

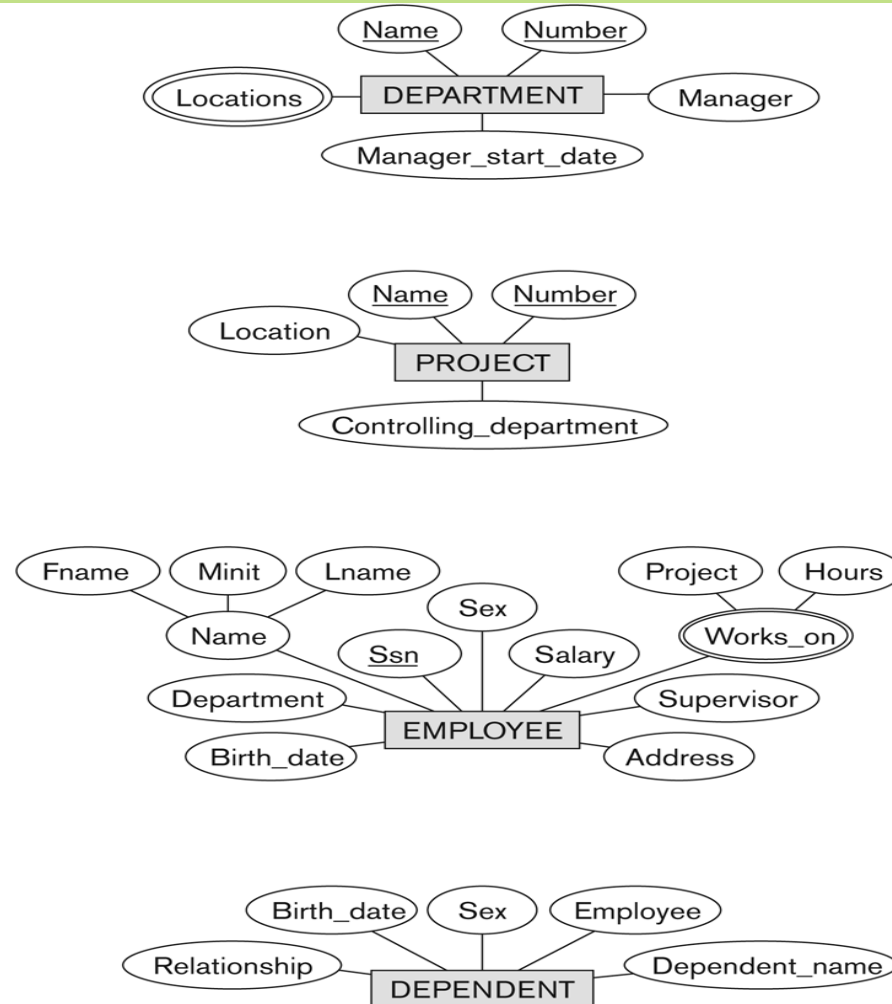
CAR<sub>3</sub>  
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

⋮

# Initial Conceptual Design of Entity Types for the COMPANY Database Schema

- Based on the requirements, we can identify four initial entity types in the COMPANY database:
  - DEPARTMENT
  - PROJECT
  - EMPLOYEE
  - DEPENDENT
- Their initial conceptual design is on the next slide as Fig 3.8
- The initial attributes shown are derived from the requirements description

# Initial Design of Entity Types: EMPLOYEE, DEPARTMENT, PROJECT, DEPENDENT



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

# Refining the initial design by introducing relationships

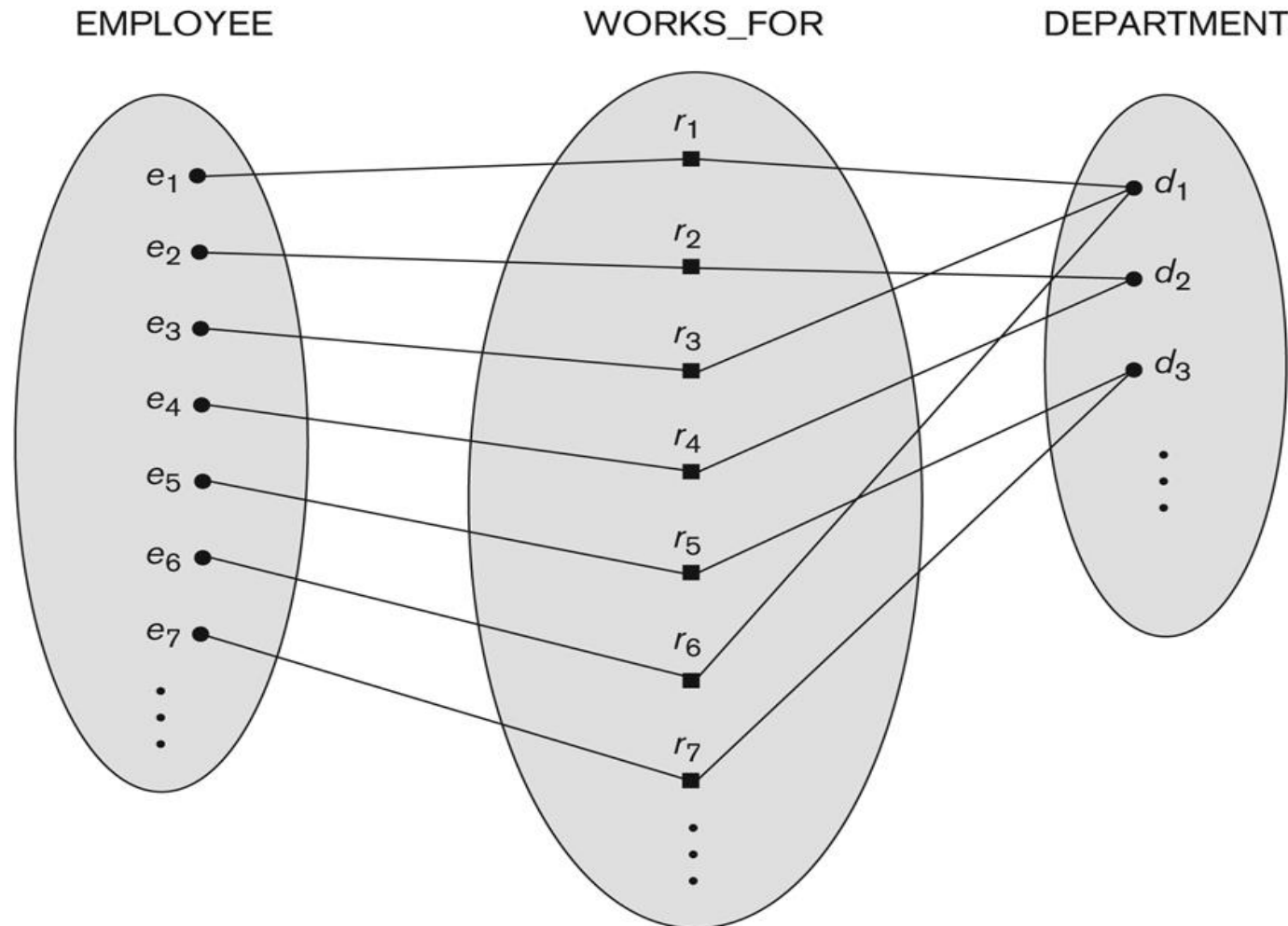
- The initial design is typically not complete
- Some aspects in the requirements will be represented as **relationships**
- ER model has three main concepts:
  - Entities (and their entity types and entity sets)
  - Attributes (simple, composite, multivalued)
  - Relationships (and their relationship types and relationship sets)
- We introduce relationship concepts next

# Relationships and Relationship Types (1)

- A **relationship** relates two or more distinct entities with a specific meaning.
  - For example, EMPLOYEE John Smith *works on* the ProductX PROJECT, or EMPLOYEE Franklin Wong *manages* the Research DEPARTMENT.
- Relationships of the same type are grouped or typed into a **relationship type**.
  - For example, the WORKS\_ON relationship type in which EMPLOYEEs and PROJECTs participate, or the MANAGES relationship type in which EMPLOYEEs and DEPARTMENTs participate.
- The degree of a relationship type is the number of participating entity types.
  - Both MANAGES and WORKS\_ON are *binary* relationships



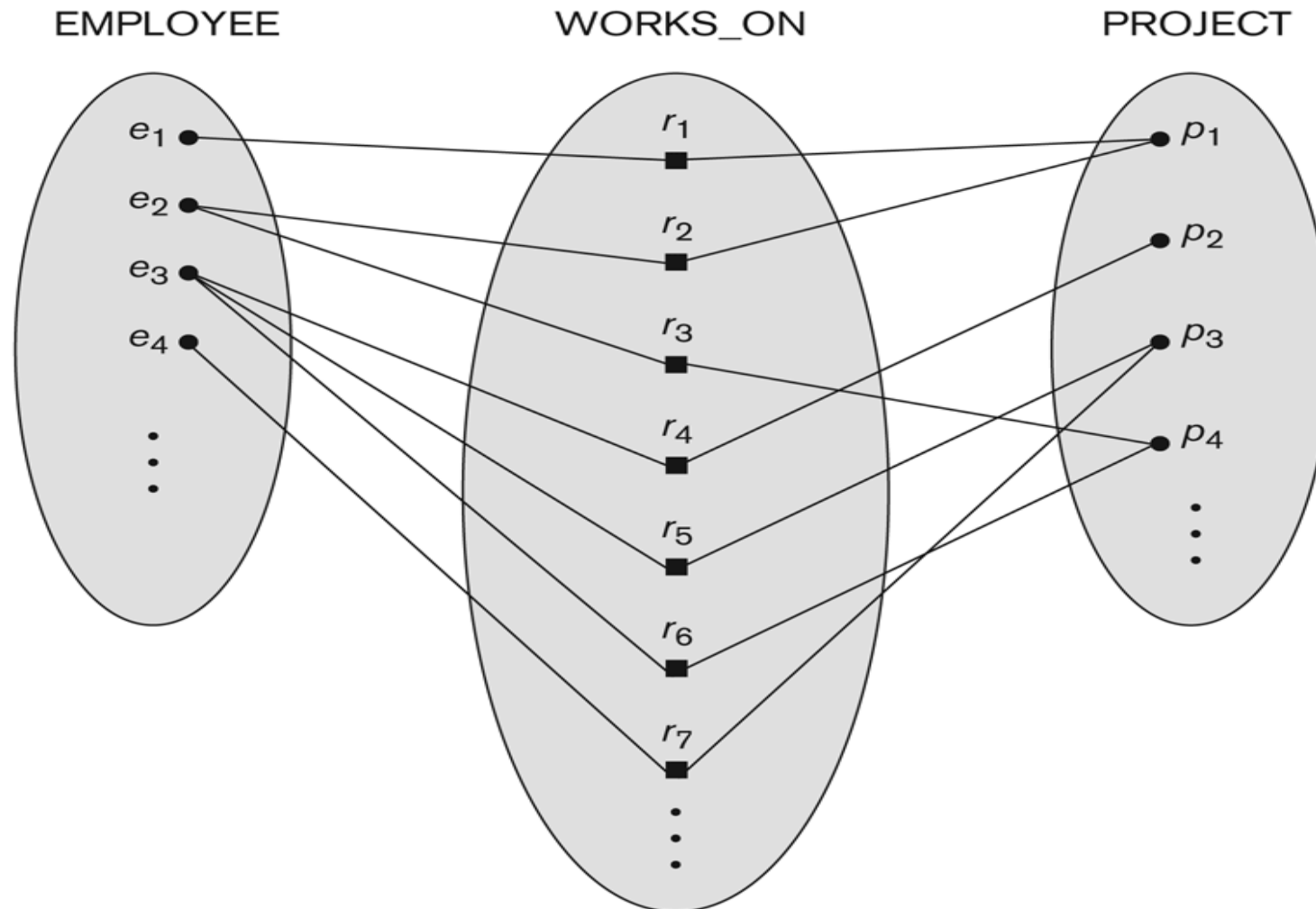
# Relationship instances of the WORKS\_FOR N:1 relationship between EMPLOYEE and DEPARTMENT



**Figure 3.9**

Some instances in the WORKS\_FOR relationship set, which represents a relationship type WORKS\_FOR between EMPLOYEE and DEPARTMENT.

# Relationship instances of the M:N WORKS\_ON relationship between EMPLOYEE and PROJECT



**Figure 3.13**  
An M:N relationship,  
WORKS\_ON.

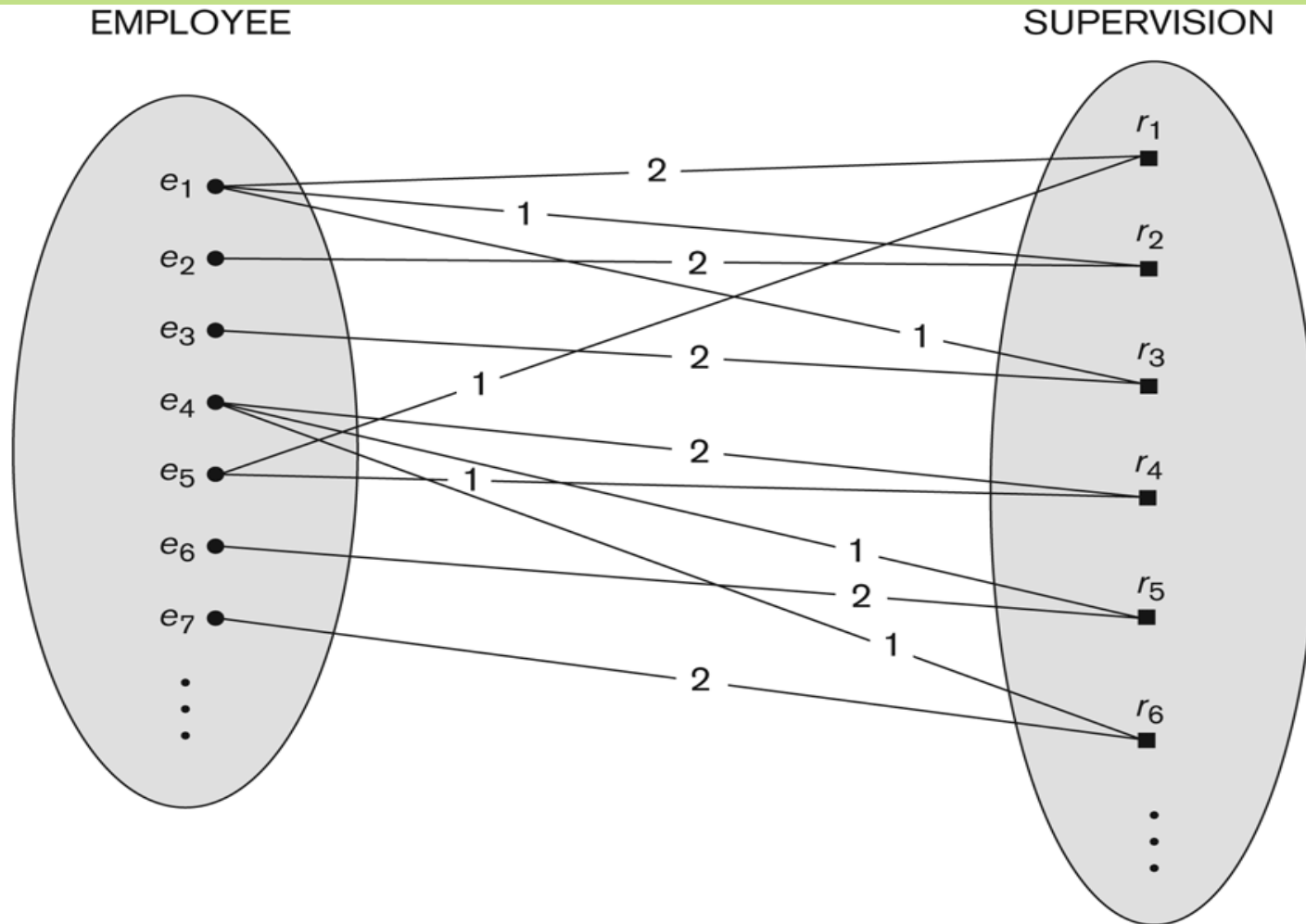
# Recursive Relationship Type

- A relationship type between the same participating entity type in **distinct roles**
- Also called a **self-referencing** relationship type.
- Example: the SUPERVISION relationship
- EMPLOYEE participates twice in two distinct roles:
  - supervisor (or boss) role
  - supervisee (or subordinate) role
- Each relationship instance relates two distinct EMPLOYEE entities:
  - One employee in *supervisor* role
  - One employee in *supervisee* role

# Displaying a recursive relationship

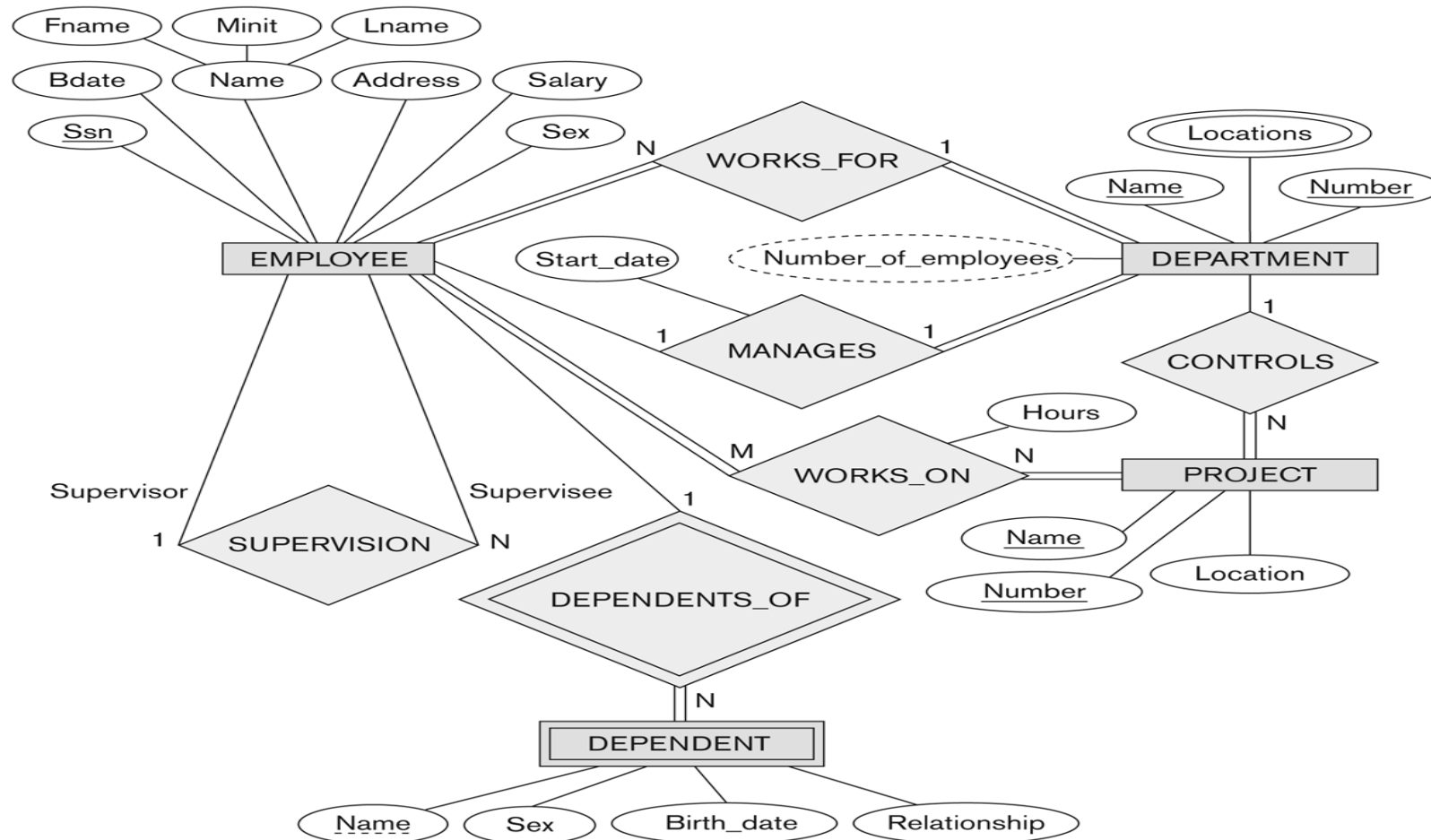
- In a recursive relationship type.
  - Both participations are same entity type in different roles.
  - For example, SUPERVISION relationships between EMPLOYEE (in role of supervisor or boss) and (another) EMPLOYEE (in role of subordinate or worker).
- In the following figure, first role participation is labeled with 1 and second role participation is labeled with 2.
- In ER diagram, there is need to display role names to distinguish participations.

# A Recursive Relationship Supervision`



**Figure 3.11**  
A recursive relationship SUPERVISION between EMPLOYEE in the *supervisor* role (1) and EMPLOYEE in the *subordinate* role (2).

# Recursive Relationship Type is: SUPERVISION (participation role names are shown)



**Figure 3.2**

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

# Weak Entity Types

- An entity that does not have a key attribute and that is identification-dependent on another entity type.
- A weak entity must participate in an identifying relationship type with an owner or identifying entity type
- Entities are identified by the combination of:
  - A partial key of the weak entity type
  - The particular entity they are related to in the identifying relationship type
- **Example:**
  - A DEPENDENT entity is identified by the dependent's first name, *and* the specific EMPLOYEE with whom the dependent is related
  - Name of DEPENDENT is the *partial key*
  - DEPENDENT is a *weak entity type*
  - EMPLOYEE is its identifying entity type via the identifying relationship type DEPENDENT\_OF

# Attributes of Relationship types

- A relationship type can have attributes:
  - For example, HoursPerWeek of WORKS\_ON
  - Its value for each relationship instance describes the number of hours per week that an EMPLOYEE works on a PROJECT.
    - A value of HoursPerWeek depends on a particular (employee, project) combination
- Most relationship type attributes are used with M:N relationships between two entity types since they cannot be migrated to one of the participating entity type.
  - In 1:N relationships, they can be transferred to the entity type on the N-side of the relationship
  - In 1:1 relationships, they can be transferred to one of the participating entity types. Eg., start\_date attribute for relationship type MANAGES can be migrated to EMPLOYEE or DEPARTMENT entity type which are 1:1.



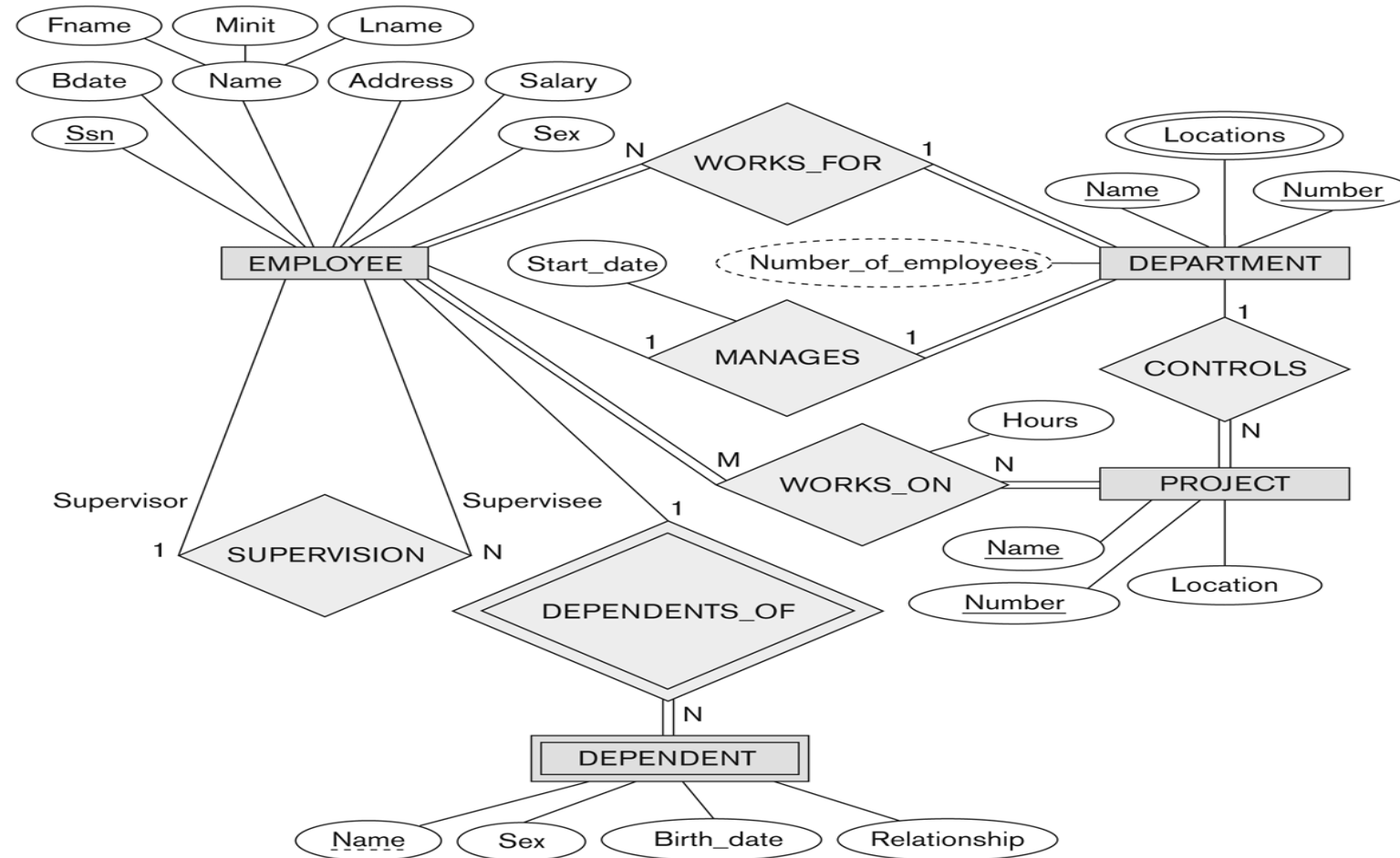
# Refining the ER Design for the COMPANY Database

- a. The initial DB design of Fig 3.8 is refined by changing the attributes that represent relationships into their relationship types.
- b. The cardinality ratio and participation constraint of each relationship type are determined from the requirements.
- c. To refine, examine each of the four existing entity type attributes and convert attributes that link this entity to another entity type through some action (relation) into a newly created relationship.
- d. For our example, in Fig 3.8, the attributes manager and manager\_start\_date of DEPARTMENT entity type are removed from department to create a relationship type MANAGES which is a 1:1 relationship between EMPLOYEE and DEPARTMENT entity types.

# Refining the ER Design for the COMPANY Database

- e. The remaining 5 newly created relationship types from all entity types are:
  - i. WORKS\_FOR, a 1:N relationship between DEPARTMENT and EMPLOYEE.
  - ii. CONTROLS, a 1:N relationship type between DEPARTMENT and PROJECT.
  - iii. SUPERVISION, a 1:N relationship type between EMPLOYEE ( in the supervisor role) and EMPLOYEE (in the supervisee role).
  - iv. WORKS\_ON, an M:N relationship type with attribute hours.
  - v. DEPENDENTS\_OF, a 1:N relationship type between EMPLOYEE and DEPENDENT.
    - 1. After creating the relationship types, all attributes in the entity types that are refined into relationships are removed.
    - 2. Removed attributes include controlling\_department from PROJECT, department, supervisor, and works\_on from EMPLOYEE; and employee from DEPENDENT.

# 5. ER Diagram for COMPANY Schema



**Figure 3.2**

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

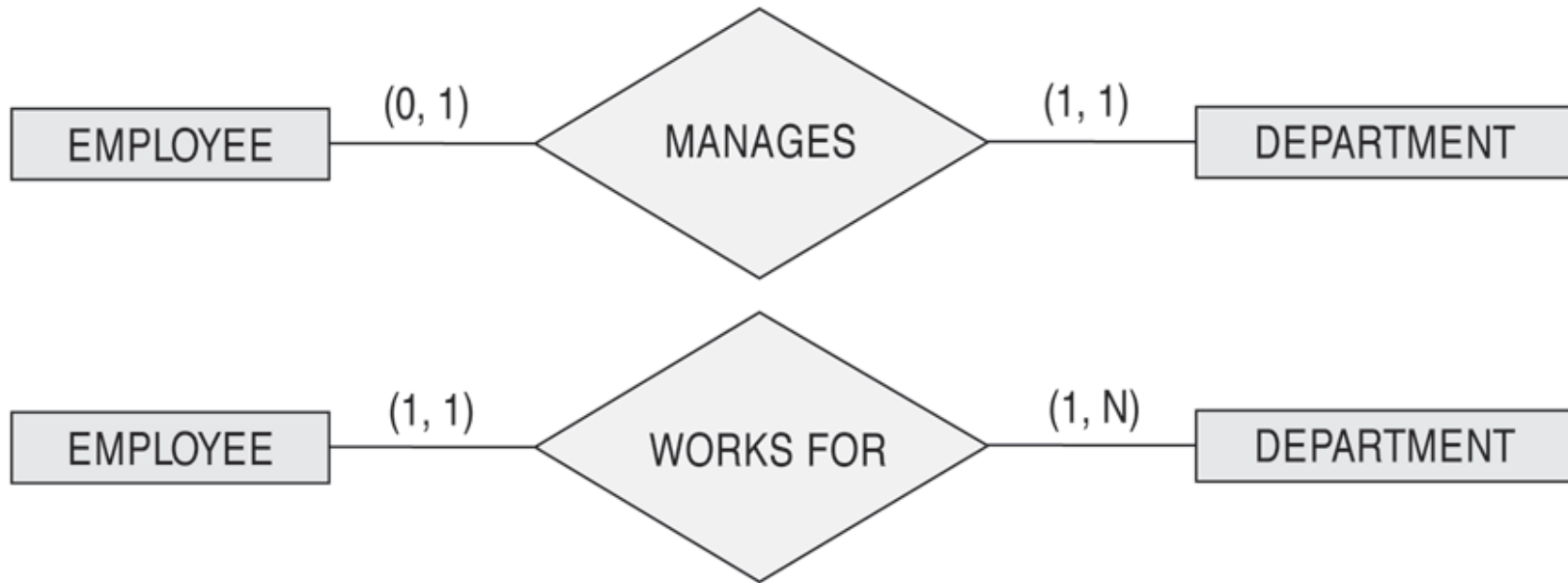
# Notation for Constraints on Relationships

- Cardinality ratio (of a binary relationship): 1:1, 1:N, N:1, or M:N
  - Shown by placing appropriate numbers on the relationship edges.
- Participation constraint (on each participating entity type): total (called existence dependency) or partial.
  - Total participation shown by double line, partial by single line.
- NOTE: These are easy to specify for Binary Relationship Types.

# Alternative (min, max) notation for relationship structural constraints:

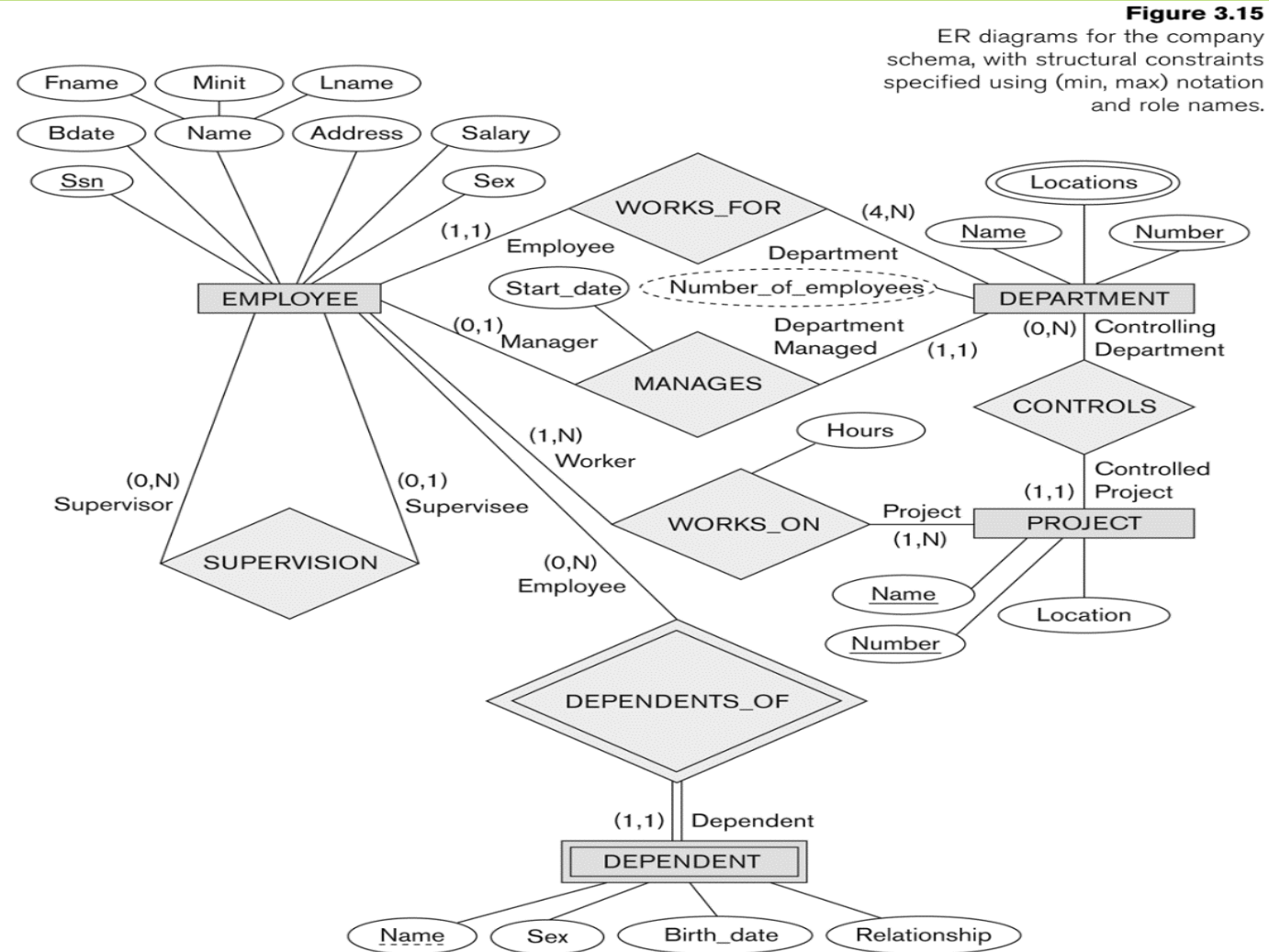
- Specified on each participation of an entity type E in a relationship type R
- Specifies that each entity e in E participates in at least *min* and at most *max* relationship instances in R
- Default(no constraint): min=0, max=n (signifying no limit)
- Must have  $\text{min} \leq \text{max}$ ,  $\text{min} \geq 0$ ,  $\text{max} \geq 1$
- Derived from the knowledge of mini-world constraints
- Examples:
  - A department has exactly one manager and an employee can manage at most one department.
    - Specify (0,1) for participation of EMPLOYEE in MANAGES
    - Specify (1,1) for participation of DEPARTMENT in MANAGES
  - An employee can work for exactly one department but a department can have any number of employees.
    - Specify (1,1) for participation of EMPLOYEE in WORKS\_FOR
    - Specify (0,n) for participation of DEPARTMENT in WORKS\_FOR

# The (min,max) notation for relationship constraints



Read the min,max numbers next to the entity type that applies to the entity type on the other side of the relationship. Eg, each employee works for exactly one department, but each department has at least 1 or at most N employees working for it.

# COMPANY ER Schema Diagram using (min, max) notation



## 6. Alternative diagrammatic notation

- ER diagrams is one popular example for displaying database schemas
- Many other notations exist in the literature and in various database design and modeling tools
- UML class diagrams is representative of another way of displaying ER concepts that is used in several commercial design tools (not discussed in this class)

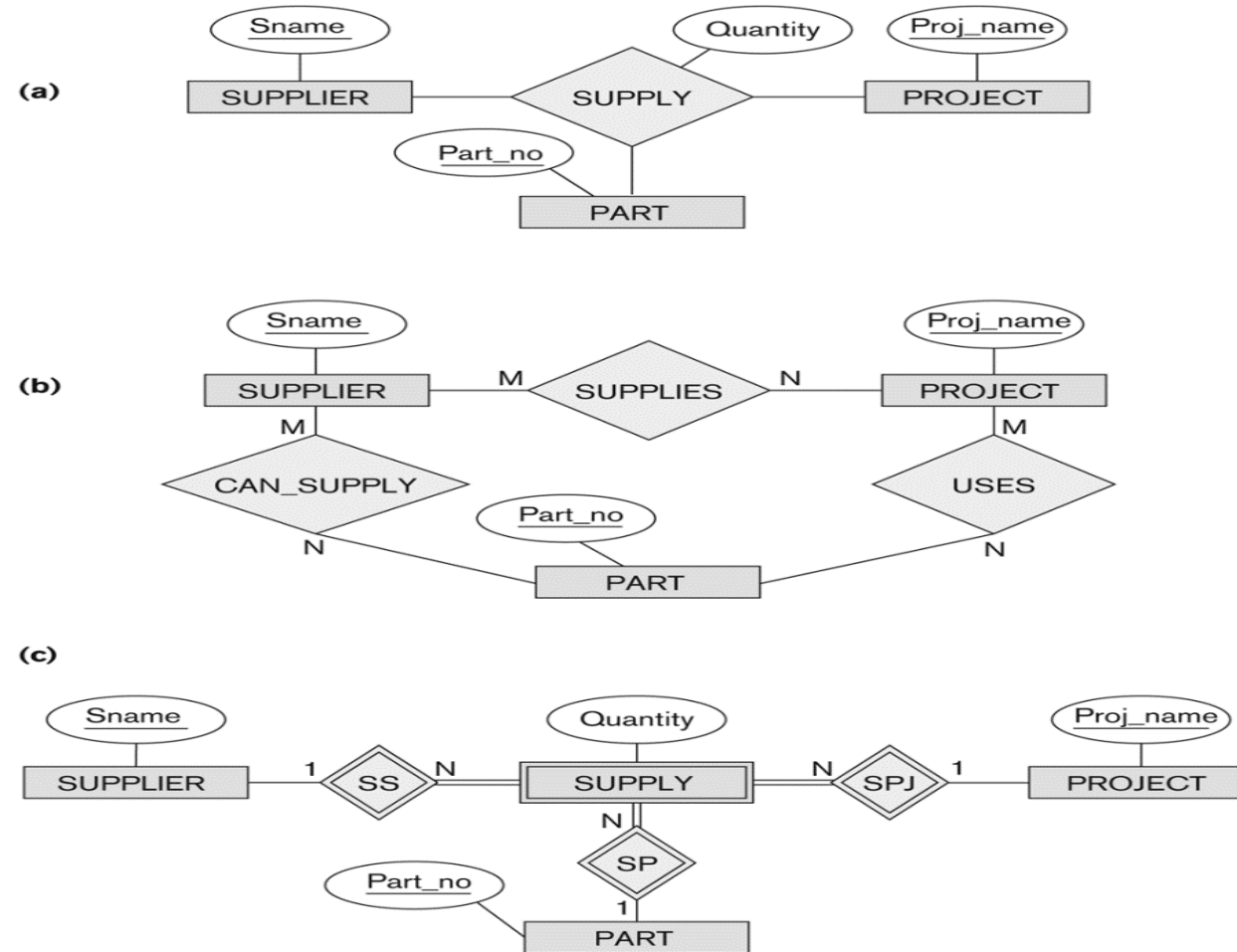


## 7. Relationships of Higher Degree:

### Discussion of n-ary relationships ( $n > 2$ )

- In general, 3 binary relationships can represent different information than a single ternary relationship (see Figure 3.17a and b on next slide)
- If needed, the binary and n-ary relationships can all be included in the schema design (see Figure 3.17a and b, where all relationships convey different meanings)
- In some cases, a ternary relationship can be represented as a weak entity if the data model allows a weak entity type to have multiple identifying relationships (and hence multiple owner entity types) (see Figure 3.17c)

# Example of a ternary relationship



**Figure 3.17**

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