

# CHAPTER 3

## 엔티티-관계(ER) 모델을 사용한 데이터 모델링

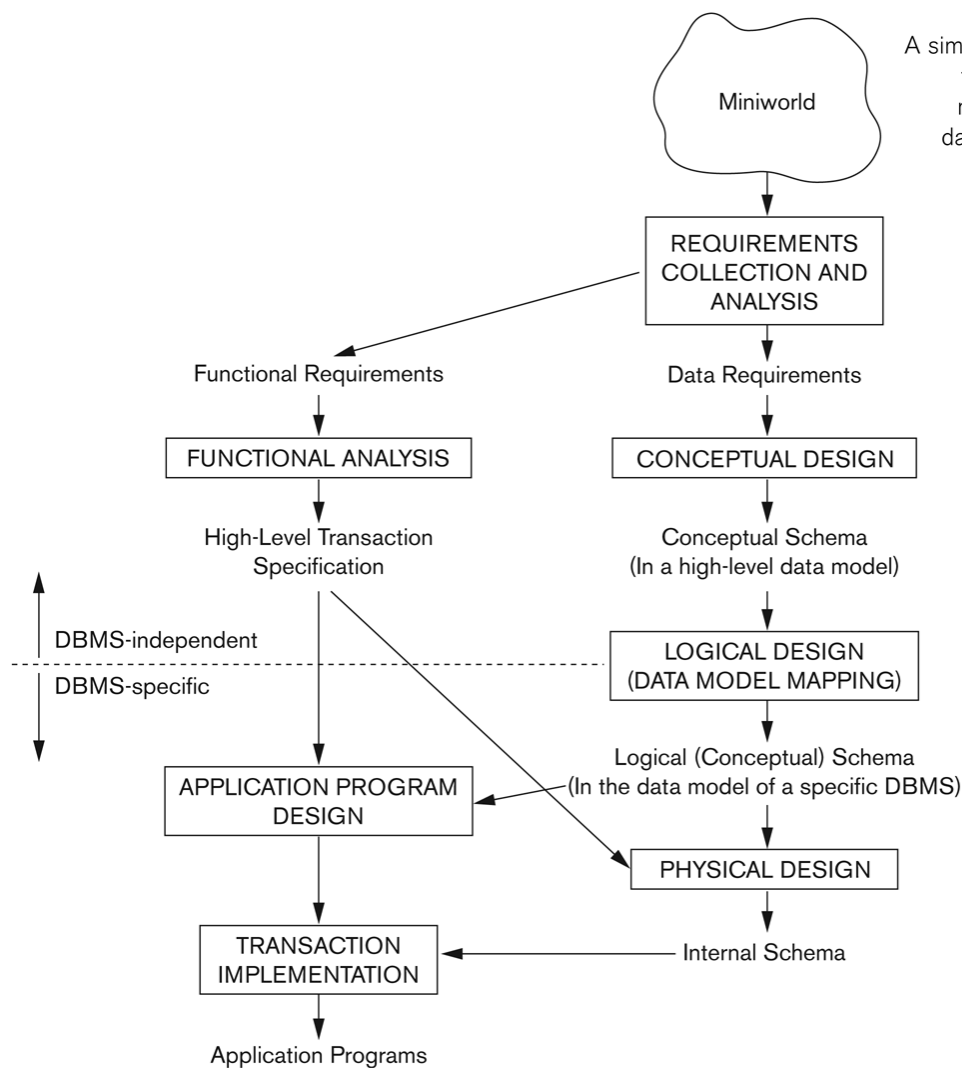
# Chapter Outline

- Overview of Database Design Process
- Example Database Application (COMPANY)
- 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
- ER Diagrams - Notation
- ER Diagram for COMPANY Schema
- Alternative Notations – UML class diagrams, others
- Relationships of Higher Degree

# 데이터베이스 설계 과정 개요

- 두가지 주요 작업:
  - 데이터베이스 설계
  - 응용 설계
- 이 장에서는 개념적 데이터베이스 설계 (conceptual database design)에 집중함
  - 데이터베이스 응용을 위한 개념적 스키마 (conceptual schema)를 설계하기 위한 단계
- 응용 설계는 데이터베이스를 접근하는 프로그램과 인터페이스에 집중함
  - 주로 소프트웨어 공학 측면에서 고려됨

# 데이터베이스 설계 과정 개요



**Figure 3.1**

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

# 개념적 설계 방법론

- 엔터티 관계(ER) 다이어그램 (이 장에서 다룸)
- 대규모 디자인의 설계 및 문서화를 위해 산업체에서 디자인 툴로 사용
- UML (Unified Modeling Language) 클래스 다이어그램은 개념 데이터베이스 설계를 문서화하는 데 실무에서 널리 사용됨

# 회사(COMPANY) 데이터베이스 예제

- 회사(COMPANY) 데이터베이스의 다음 (간략화된) 요구 사항을 기반으로 데이터베이스 스키마 설계를 만들과자 함 :
  - 회사는 부서로 구성되어 있습니다. 각 부서에는 이름, 번호 및 부서를 관리하는 직원이 있습니다. 부서 관리자의 시작 날짜를 추적합니다. 부서에 여러 위치가 있을 수 있습니다.
  - 각 부서는 여러 프로젝트를 관리합니다. 각 프로젝트는 고유한 이름과 고유 번호를 가지며 한 곳에만 위치합니다.

# 회사(COMPANY) 데이터베이스 예제 (Continued)

- 데이터베이스는 각 사원의 주민등록번호, 주소, 급여, 성별 및 생년월일을 저장합니다.
  - 각 사원은 한 부서에서 일하지만 여러 프로젝트에서 일할 수 있습니다.
  - 사원이 현재 각 프로젝트에서 작업하는 주당 시간 수를 추적합니다.
  - 각 직원의 직속 상사(supervisor)를 추적해야 합니다.
- 각 사원은 여러 명의 부양가족(DEPENDENT)를 가질 수 있습니다.
  - 각 부양 가족에 대해, 이름, 성별, 생년월일 및 사원과의 관계를 기록합니다.



# ER 모델 개념

- 엔티티(Entity)와 애트리뷰트(Attribute)
  - 엔티티는 ER 모델의 기본 개념임. 엔티티는 데이터베이스에 표현되는 작은 세계내의 특정 사물 또는 객체임.
    - 예: John Smith라는 사원, Research라는 부서, ProductX라는 프로젝트 등.
  - 애트리뷰트는 엔티티를 설명하는 속성(properties)들임.
    - 예: 사원 엔티티는 이름, SSN, 주소, 성별, 생일을 애트리뷰트로 가질 수 있음.
  - 특정 엔티티는 각 애트리뷰들에 대한 값을 가질 수 있음.
    - 예: 한 특정 사원 엔티티는 Name='John Smith', SSN='123456789', Address='731, Fondren, Houston, TX', Sex='M', BirthDate='09-JAN-55' 값을 가짐.
  - 각 애트리뷰트는 자신과 관련된 값의 집합 (또는 데이터 타입)을 가짐 – 예: 정수, 문자열, 날짜, 열거형 타입 등. ...

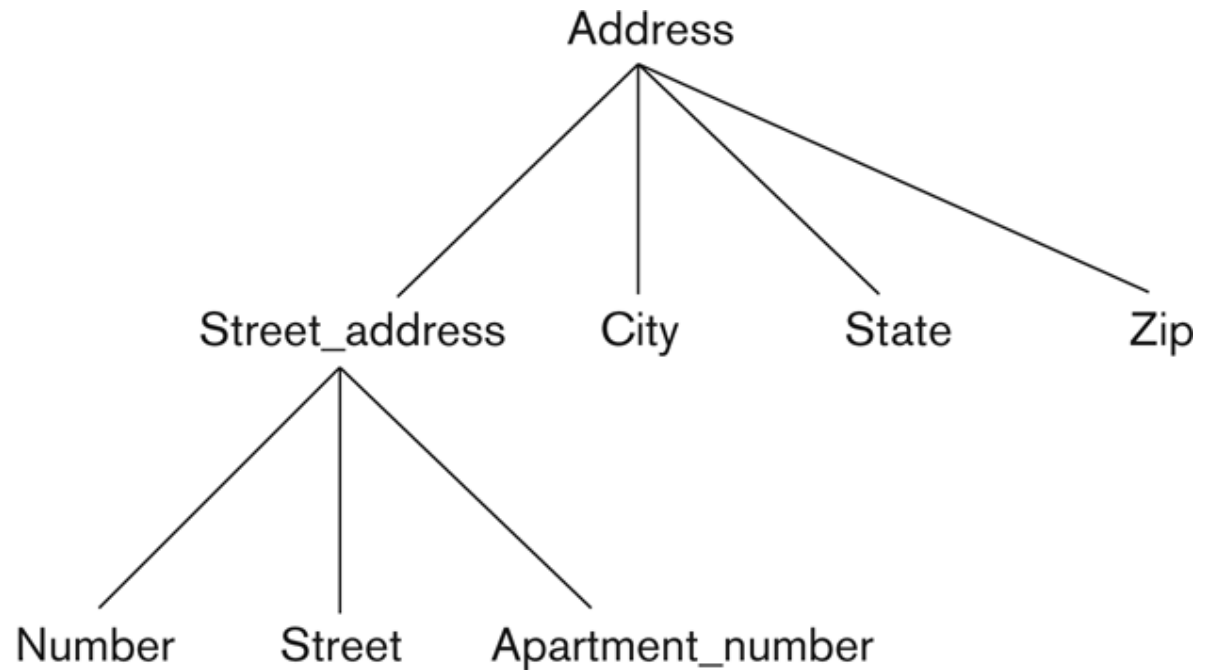
# 애트리뷰트 종류 (1)

- 단순(Simple)
  - 각 엔티티는 애트리뷰트 값으로 단일 원자 값을 가짐. 예: SSN 또는 성별.
- 복합(Composite)
  - 애트리뷰트가 여러 구성 요소로 구성됨. 예를 들면:
    - Address(Apt#, House#, Street, City, State, ZipCode, Country), 또는
    - Name(FirstName, MiddleName, LastName).
    - 복합 애트리뷰트는 일부 구성 요소가 또 다른 복합 애트리뷰트인 계층 구조 형태를 가질 수 있음.
- 다치(Multi-valued)
  - 애트리뷰트 값으로 여러 개의 값을 가질 수 있음. 예: 자동차의 색깔, 학생의 이전 학위(PreviousDegree).
    - {Color} 또는 {PreviousDegrees} 라고 표기함.

## 애트리뷰트 종류(2)

- 드물지만, 복합 및 다중 값 속성은 임의 개수의 단계들로 중첩 될 수 있음.
  - 예를 들어, 학생의 이전 학위(PreviousDegrees)는 {PreviousDegrees (College, Year, Degree, Field)}로 표시되는 복합 다중 값 애트리뷰트입니다.
  - 여러 개의 PreviousDegrees 값이 존재할 수 있으며, 각각 4 개의 하위 구성 요소 애트리뷰트를 갖습니다:
    - College, Year, Degree, Field

# 복합 애트리뷰트의 예제



**Figure 3.4**

A hierarchy of composite attributes.

# 엔터티 타입과 키 애트리뷰트 (1)

- 동일한 애트리뷰트들을 갖는 엔터티들은 한 엔터티 타입으로 그룹화되거나 타입화됨.
  - 예를들어, EMPLOYEE 엔터티 타입과 PROJECT 엔터티 타입.
- 각 엔터티가 고유한 값을 가져야 하는 애트리뷰트를 그 엔터티 타입의 키 애트리뷰트라고 함.
  - 예를 들어, EMPLOYEE의 SSN.

## 엔터티 타입과 키 애트리뷰트(2)

- 키 애트리뷰트는 복합 애트리뷰트일 수 있음.
  - VehicleTagNumber는 CAR 엔터티 타입의 키이며, (Number, State)로 구성된 복합 애트리뷰트이다.
- 한 엔터티 타입은 여러 개의 키를 가질 수 있음.
  - CAR 엔터티 타입은 두개의 키를 갖는다:
    - VehicleIdentificationNumber (주로 VIN라 불림)
    - VehicleTagNumber (Number, State), 자동차 번호판.
- 각 키에는 밑줄을 치. (참고 : ER 모델의 키는 하나의 "기본 키"에 밑줄이 있는 관계형 스키마와 다름).

# 엔터티 집합 (Entity Set)

- 각 엔터티 타입은 데이터베이스에 저장된 엔터티들의 모임(collection)을 가짐.
  - 엔터티 집합(**entity set**) 또는 가끔 **엔터티 모임(entity collection)** 이라고 부름
- 이전 슬라이드는 CAR 엔터티 집합 내에 있는 세개의 CAR 엔터티 인스턴스를 보여줌.
- 동일한 이름(CAR)이 엔터티 타입과 엔터티 집합 모두를 나타내는 데 사용됨.
- 그러나, 엔터티 타입과 엔터티 집합에 서로 다른 이름을 줄 수 있음.
- 엔터티 집합은 데이터베이스에 저장된 그 타입의 엔터티들의 현재 상태임

# 애트리뷰트의 값 집합(도메인)

- 각 단순 애트리뷰트는 하나의 값 집합에 연관됨.
  - 예: 성(Lastname)은 최대 15개 문자로 된 문자 열을 값으로 갖는다.
  - 날짜는 MM-DD-YYYY로 구성된 값을 갖는다  
(여기서 각 문자는 하나의 정수임)
- **값 집합(value set)**은 한 애트리뷰트에 연관된 값들의 집합을 말함



# 애트리뷰트와 값 집합

- 값 집합은 대부분의 프로그래밍 언어의 데이터 타입(예 : 정수, 문자 (n), 실수, 비트) 과 유사함.
- 엔터티 타입 E의 한 애트리뷰트 A가 값 집합 V를 가질 경우, 수학적으로 다음과 같은 함수로 정의됨:

$$A : E \rightarrow P(V)$$

여기서  $P(V)$ 는 V의 멱집합(power set) (모든 가능한 부분 집합을 의미함)를 나타냄.








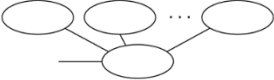

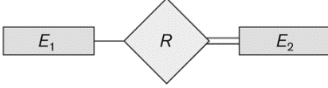
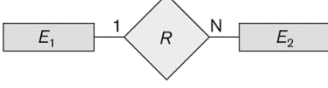
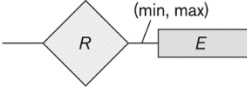
- 엔터티 e에 대한 애트리뷰트 A의 값을  $A(e)$ 로 표기함.

# 엔티티 타입의 표시

- ER 다이어그램에서 엔티티 타입은 사각형 상자에 표시됨.
- 애트리뷰트들은 타원으로 표시됨
  - 각 애트리뷰트는 해당 엔티티 타입에 연결됩니다
  - 복합 애트리뷰트의 구성 요소는 복합 애트리뷰트를 나타내는 타원에 연결됩니다
  - 각 키 애트리뷰트는 밑줄이 그어집니다.
  - 다치 애트리뷰트는 이중 타원으로 표시됩니다.
- 전체 ER 표기법은 다음 슬라이드를 참조 바람

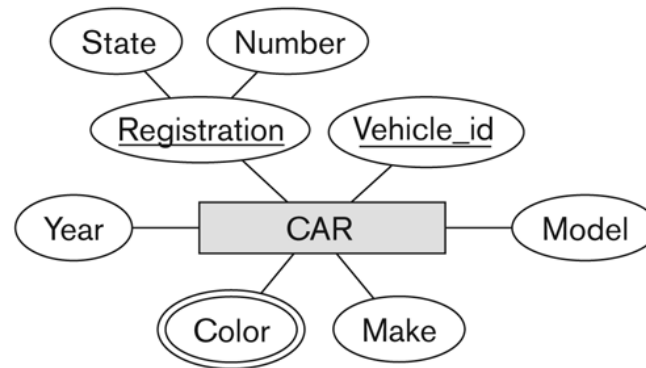
# ER 다이어그램을 위한 표기법

**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)<br>on Participation of $E$ in $R$ |

# 두개의 키를 갖는 CAR 엔터티 타입 및 관련된 엔터티 집합

(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})

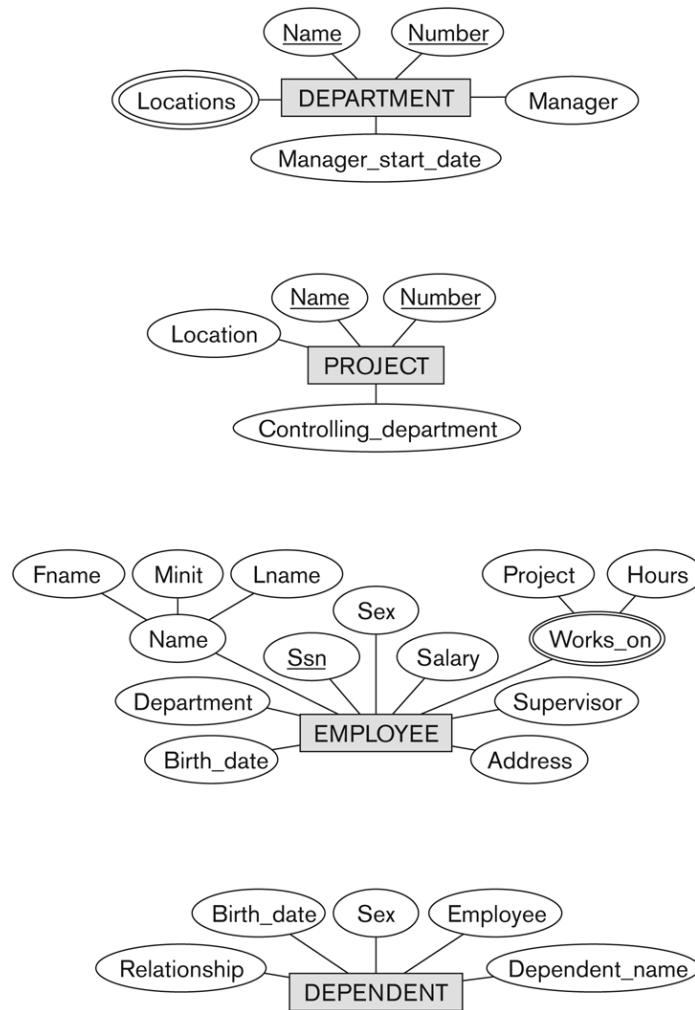
⋮

# 회사(COMPANY) 데이터베이스 스키마를 위한 엔터티 타입들의 초기 개념적 설계

- 요구 사항에 따라 COMPANY 데이터베이스에서 4가지 초기 엔터티 유형을 식별 할 수 있음:
  - DEPARTMENT
  - PROJECT
  - EMPLOYEE
  - DEPENDENT
- 초기 개념 설계는 다음 슬라이드에 나와 있음.
- 표시된 초기 애트리뷰트는 요구 사항 설명에서 유도됨.

# 엔터티 타입들의 초기 설계:

## 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.

# 관계를 도입하여 초기 설계를 개선

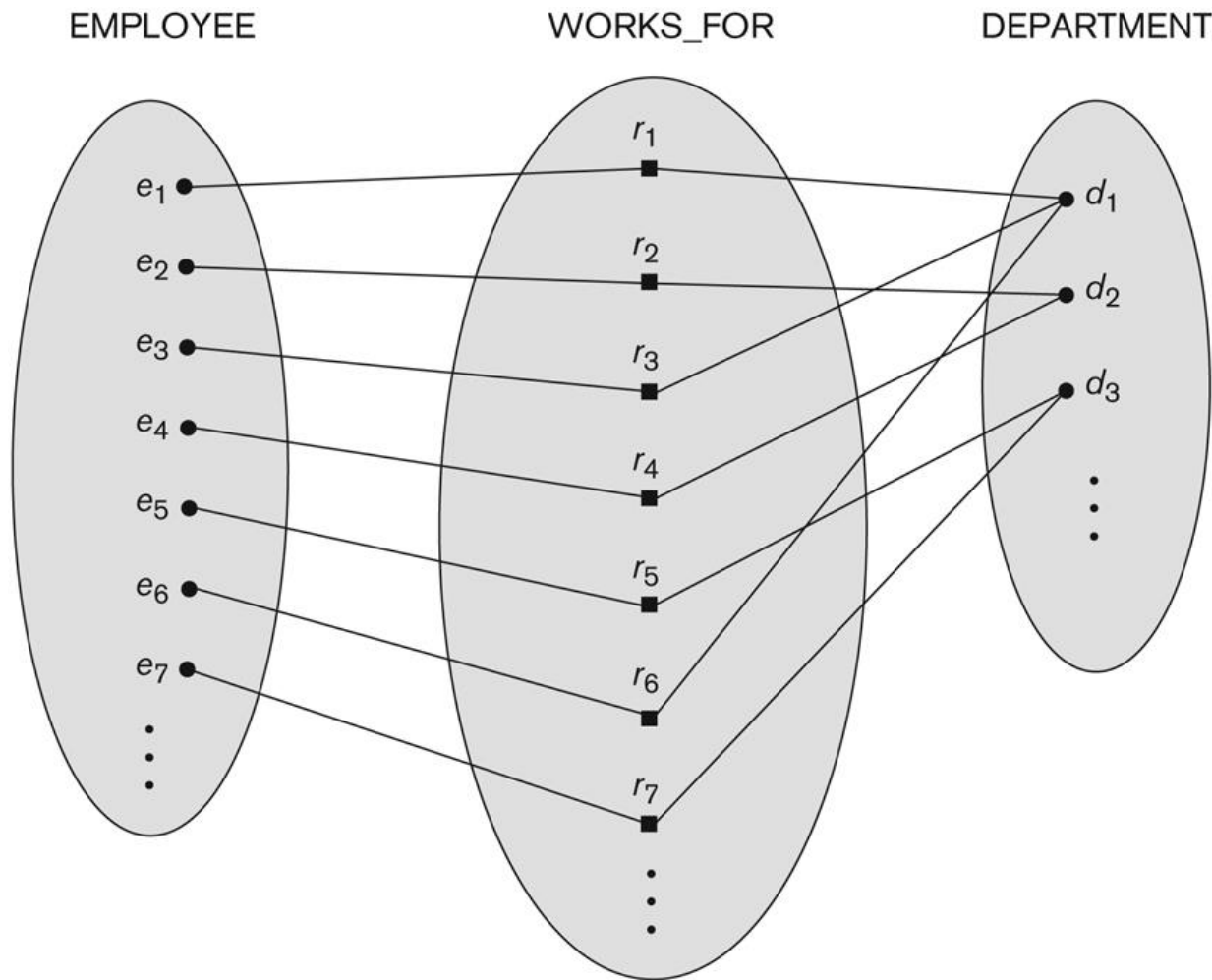
- 초기 디자인은 일반적으로 완전하지 않음
- 요구 사항의 일부 요소들은 **관계**로 표시됨
- ER 모델에는 세 가지 주요 개념이 있음:
  - 엔터티 (및 엔터티 타입과 엔터티 집합)
  - 애트리뷰트 (단순, 복합, 다치)
  - 관계 (및 관계 타입과 관계 집합)
- 관계 개념은 뒤에서 소개함

# 관계와 관계 타입 (1)

- 관계는 특정 의미를 가진 둘 이상의 별개의 엔티티와 관련됨.
  - 예를 들어, EMPLOYEE John Smith는 ProductX PROJECT에서 일하거나, EMPLOYEE Franklin Wong은 Research DEPARTMENT를 관리합니다.
- 동일한 타입의 관계는 관계 타입으로 그룹화되거나 타입화됨.
  - 예를 들어, EMPLOYEE 및 PROJECT가 참여하는 WORKS\_ON 관계 타입 또는 EMPLOYEE 및 DEPARTMENT가 참여하는 MANAGES 관계 타입이 있습니다.
- 관계 타입의 차수는 참여하는 엔티티 타입의 개수임.
  - MANAGES와 WORKS\_ON은 이진 관계입니다.



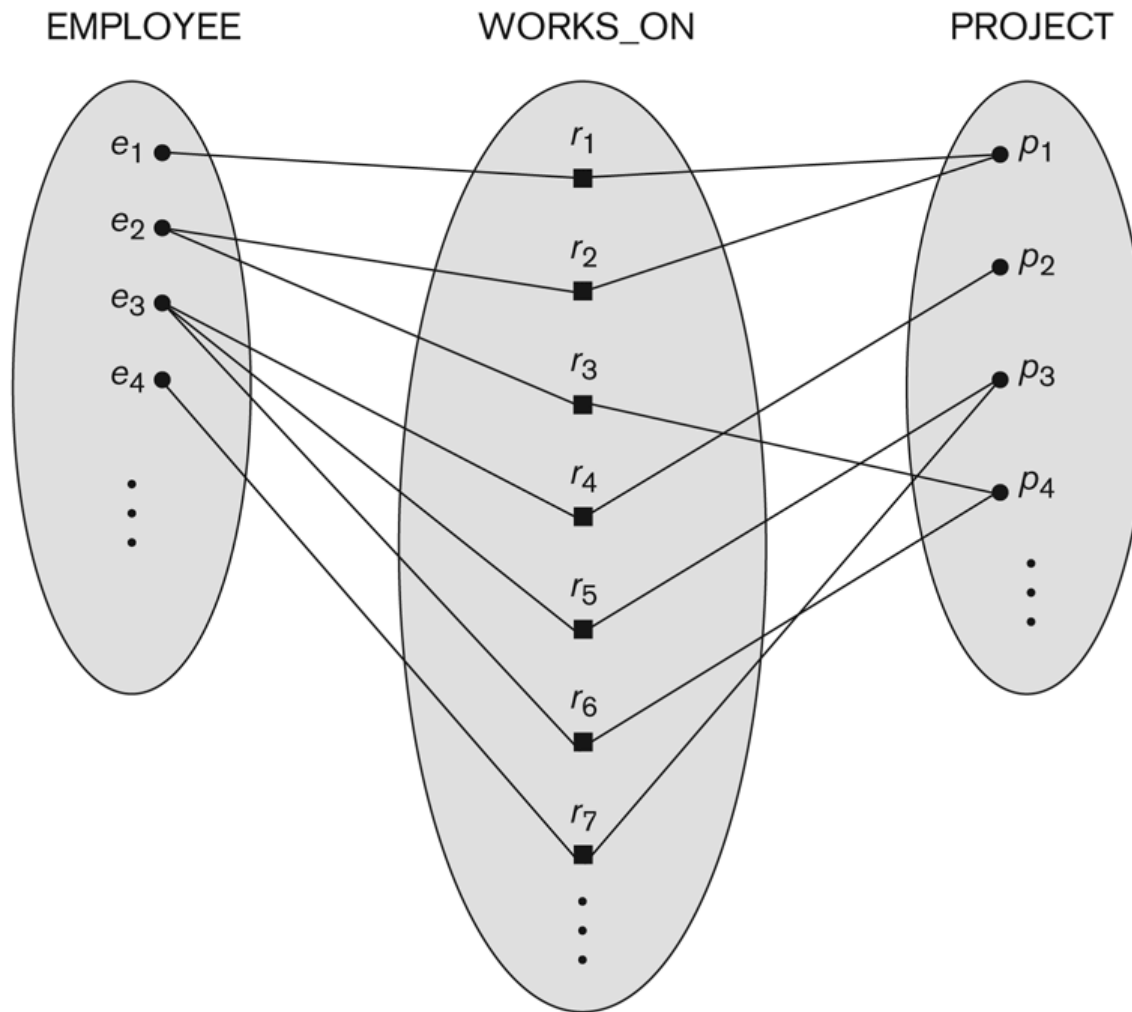
# EMPLOYEE와 DEPARTMENT 간의 WORKS\_FOR N:1 관계의 관계 인스턴스



**Figure 3.9**

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

# EMPLOYEE와 PROJECT 간의 M:N WORKS\_ON 관계의 관계 인스턴스



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

# 관계 타입 대 관계 집합 (1)

- 관계 타입:
  - 관계의 스키마 설명이다.
  - 관계 이름 및 참여 엔티티 유형을 식별합니다.
  - 특정 관계 제약 조건도 식별합니다.
- 관계 집합:
  - 데이터베이스에 표시된 현재 관계 인스턴스의 집합
  - 관계 타입의 현재 상태

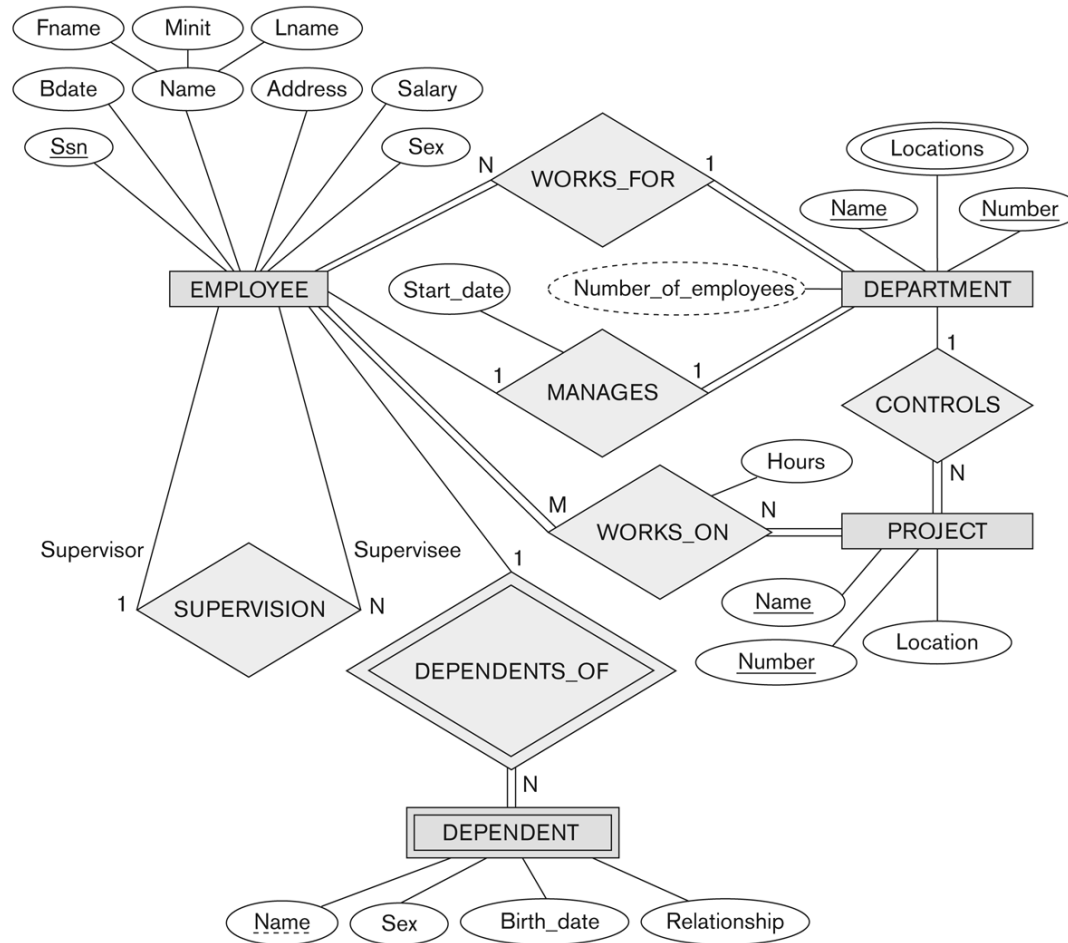
## 관계 타입 대 관계 집합 (2)

- 이전 그림에는 관계 세트가 표시되었음.
- 집합내의 각 인스턴스는 개별 참여 엔티티 (각 참여 엔티티 타입에서 하나씩)와 관련됨
- ER 다이어그램에서 관계 타입을 다음과 같이 나타냄 :
  - 다이아몬드 모양의 상자는 관계 타입을 표시하는 데 사용됨
  - 참여 엔티티 타입을 직선으로 연결함
  - 관계 타입은 화살표로 표시되지 않습니다. 그 이름은 일반적으로 왼쪽에서 오른쪽으로, 위에서 아래로 읽을 수 있어야 합니다.

# 관계를 도입하여 회사 데이터베이스 스키마 개선

- 요구 사항을 조사하여, 6 가지 관계 타입이 식별됨
- 모든 관계 타입은 (차수가 2인) 이진 관계임
- 참여 엔티티 타입들과 함께 아래에 나열 됨 :
  - WORKS\_FOR (EMPLOYEE와 DEPARTMENT간의 관계)
  - MANAGES (EMPLOYEE와 DEPARTMENT간의 관계)
  - CONTROLS (DEPARTMENT와 PROJECT간의 관계)
  - WORKS\_ON (EMPLOYEE와 PROJECT간의 관계)
  - SUPERVISION (EMPLOYEE (부하 역할)와 EMPLOYEE (감독자 역할)간의 관계)
  - DEPENDENTS\_OF (EMPLOYEE와 DEPENDENT간의 관계)

# ER 다이어그램 – 관계 타입들: WORKS\_FOR, MANAGES, WORKS\_ON, CONTROLS, SUPERVISION, DEPENDENTS\_OF



**Figure 3.2**

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

# 관계 타입에 대한 논의

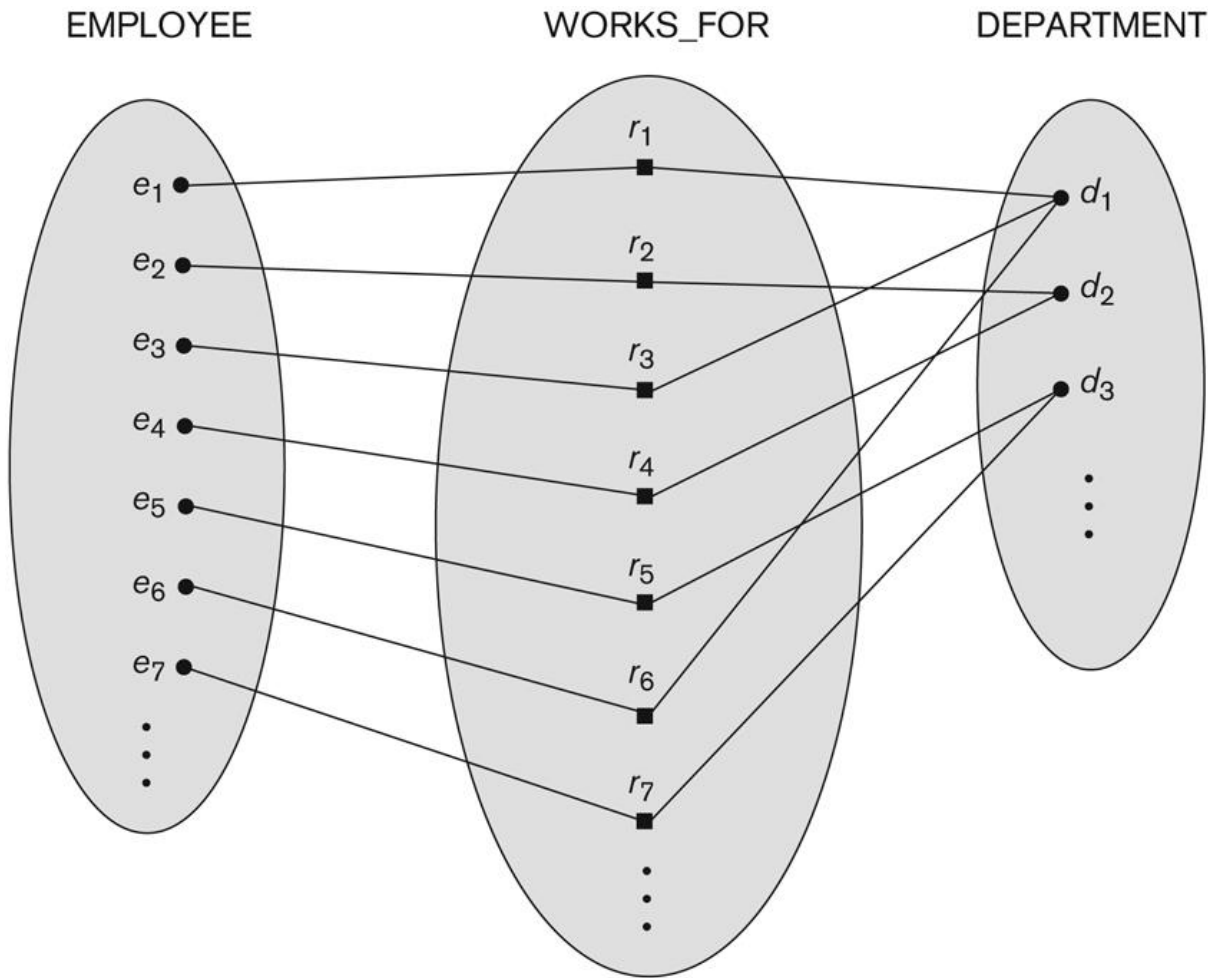
- 초기 엔티티 타입의 일부 애트리뷰트들은 개선된 설계에서 관계로 개선됨:
  - DEPARTMENT의 Manager -> MANAGES
  - EMPLOYEE의 Works\_on -> WORKS\_ON
  - EMPLOYEE의 Department -> WORKS\_FOR
  - 등등
- 일반적으로 동일한 참여 엔티티 타입들 간에 둘 이상의 관계 타입이 존재할 수 있음
  - MANAGES와 WORKS\_FOR은 EMPLOYEE와 DEPARTMENT 사이에 서로 다른 관계 타입들이다.
  - 서로 다른 의미와 서로 다른 관계 인스턴스들을 가짐

# 관계에 대한 제약조건

- 관계 타입들에 대한 제약조건들
  - (비율 제약조건이라고도 함)
  - 카디널리티 비율 (Cardinality Ratio) (최대 참여 수를 나타냄)
    - One-to-one (1:1)
    - One-to-many (1:N) 또는 Many-to-one (N:1)
    - Many-to-many (M:N)
- 존재 의존성 제약조건 (최소 참여 수를 나타냄) (참여 제약조건이라고도 함)
  - 영(zero) (선택적 참여, 존재-종속적이지 않음)
  - 하나 이상 (필수적 참여, 존재-종속적임)



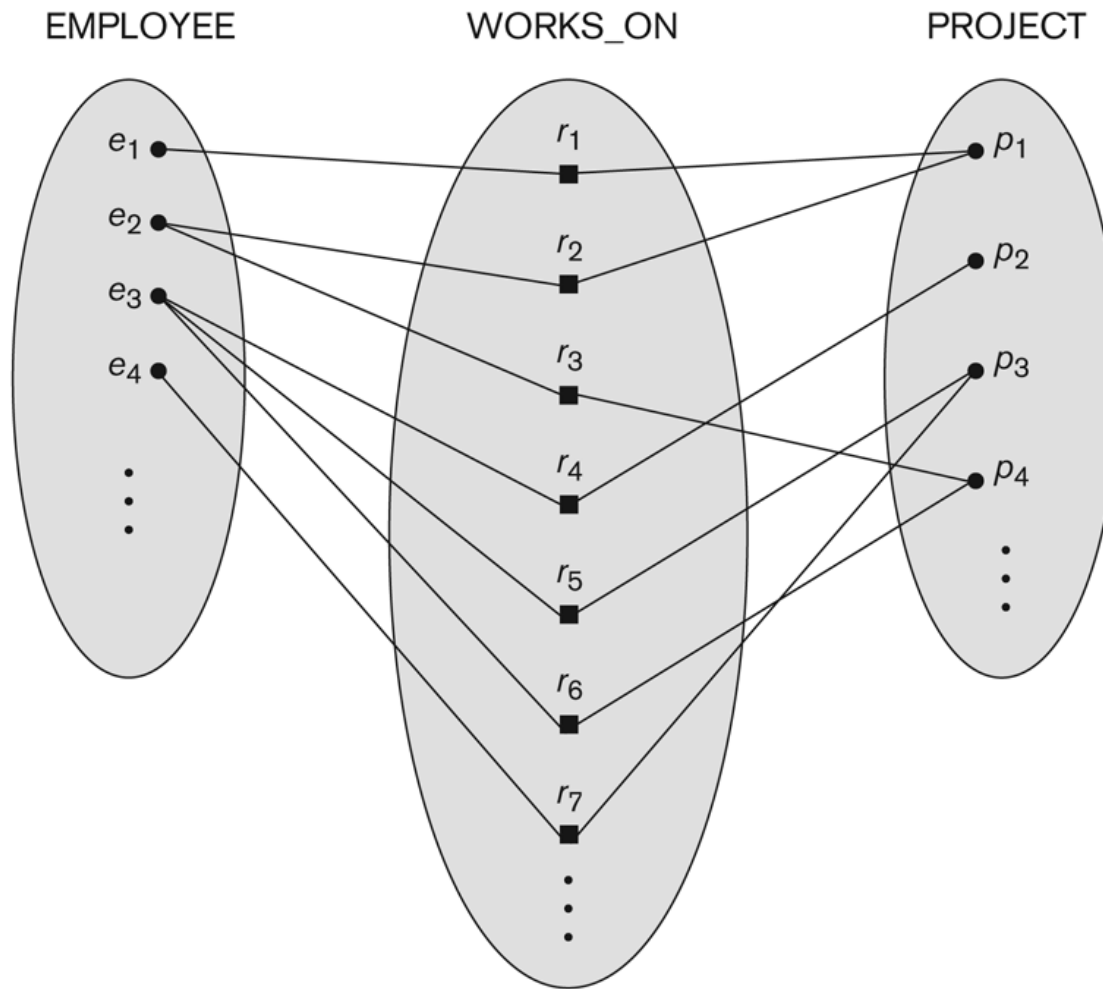
# Many-to-one (N:1) 관계



**Figure 3.9**

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

# Many-to-many (M:N) 관계



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

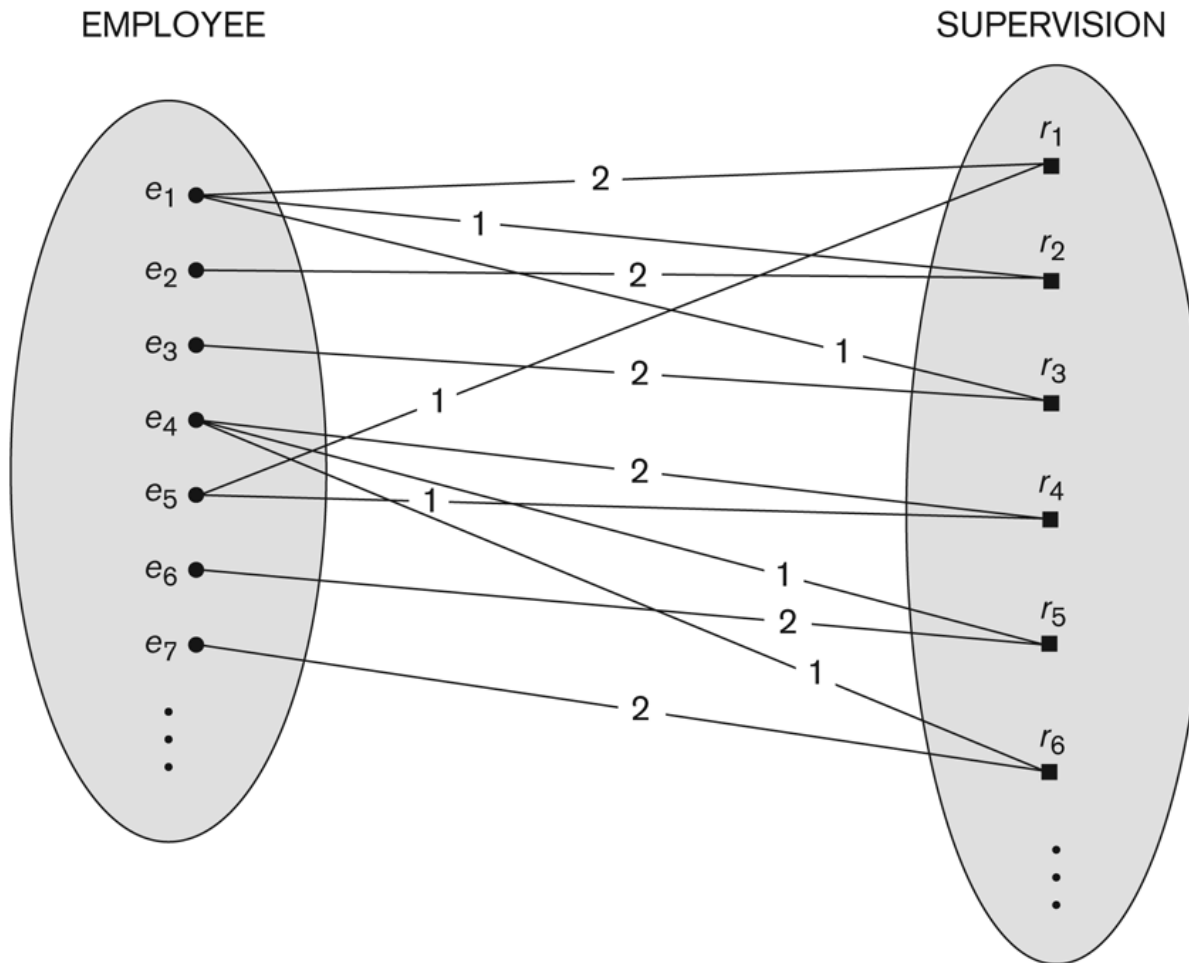
# 재귀 관계 타입

- 동일한 엔터티 타입이 서로 다른 역할로 참여하는 관계 타입
- 자체-참조 관계 유형이라고도 함.
- 예: SUPERVISION 관계
- EMPLOYEE가 서로 다른 역할로 두 번 참여함:
  - 감독자 (또는 상관/보스) 역할
  - 피감독자 (또는 부하) 역할
- 각 관계 인스턴스는 서로 다른 EMPLOYEE 엔터티들과 관련함:
  - 감독자 역할을 하는 한 사원(employee)
  - 피 감독자 역할을 하는 또 다른 사원(employee)

# 재귀 관계 표시

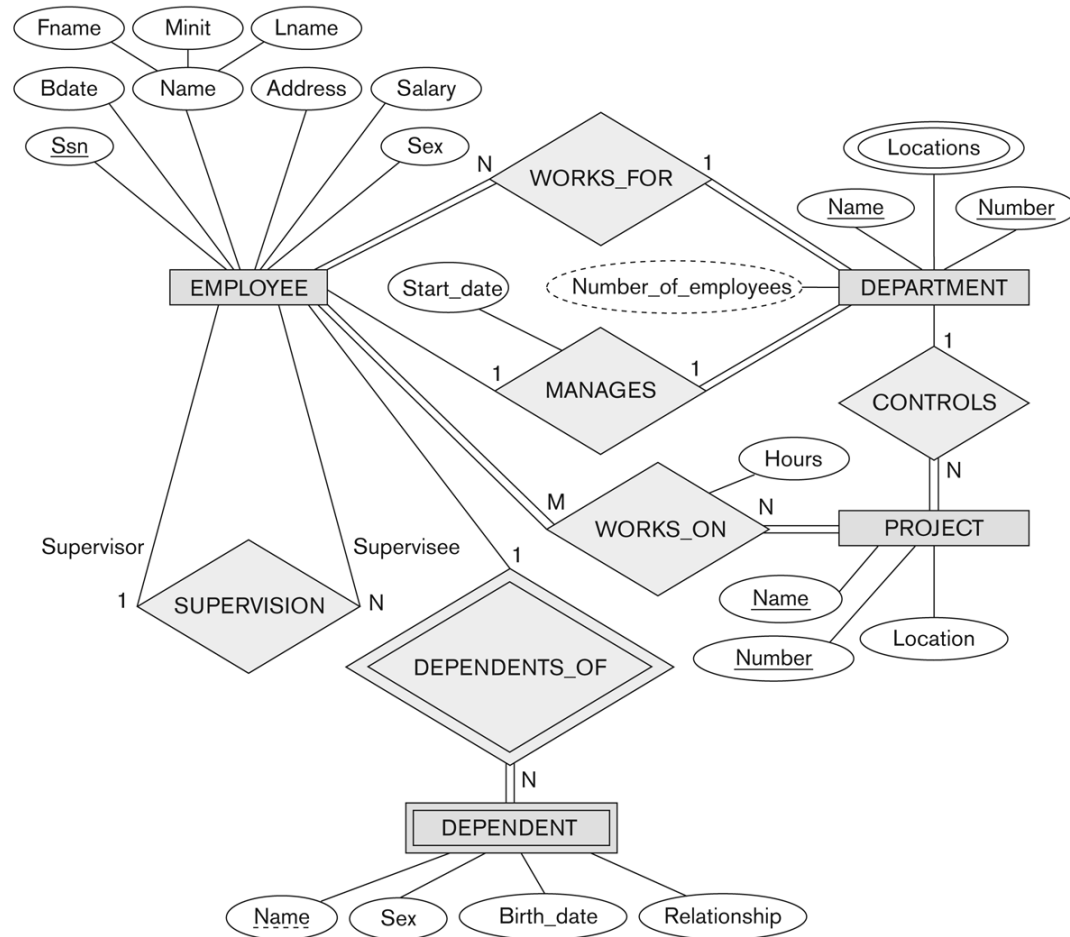
- 재귀 관계 타입에서
  - 두 참여자는 서로 다른 역할을 하는 동일한 엔터티 타입이다.
  - 예를 들어, EMPLOYEE (감독자 또는 상사 역할)와 (또 다른) EMPLOYEE (부하 또는 작업자 역할) 간의 SUPERVISION 관계.
- 다음 그림에서 첫 번째 역할 참여는 1로 표시되고 두 번째 역할 참여는 2로 표시됨.
- ER 다이어그램에서 참여를 구별하기 위해 역할 이름을 표시해야 함.

# 재귀 관계의 SUPERVISION`



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

# 재귀 관계 타입: SUPERVISION (참여 역할 이름들이 표시됨)



**Figure 3.2**

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

# 약한 엔터티 타입

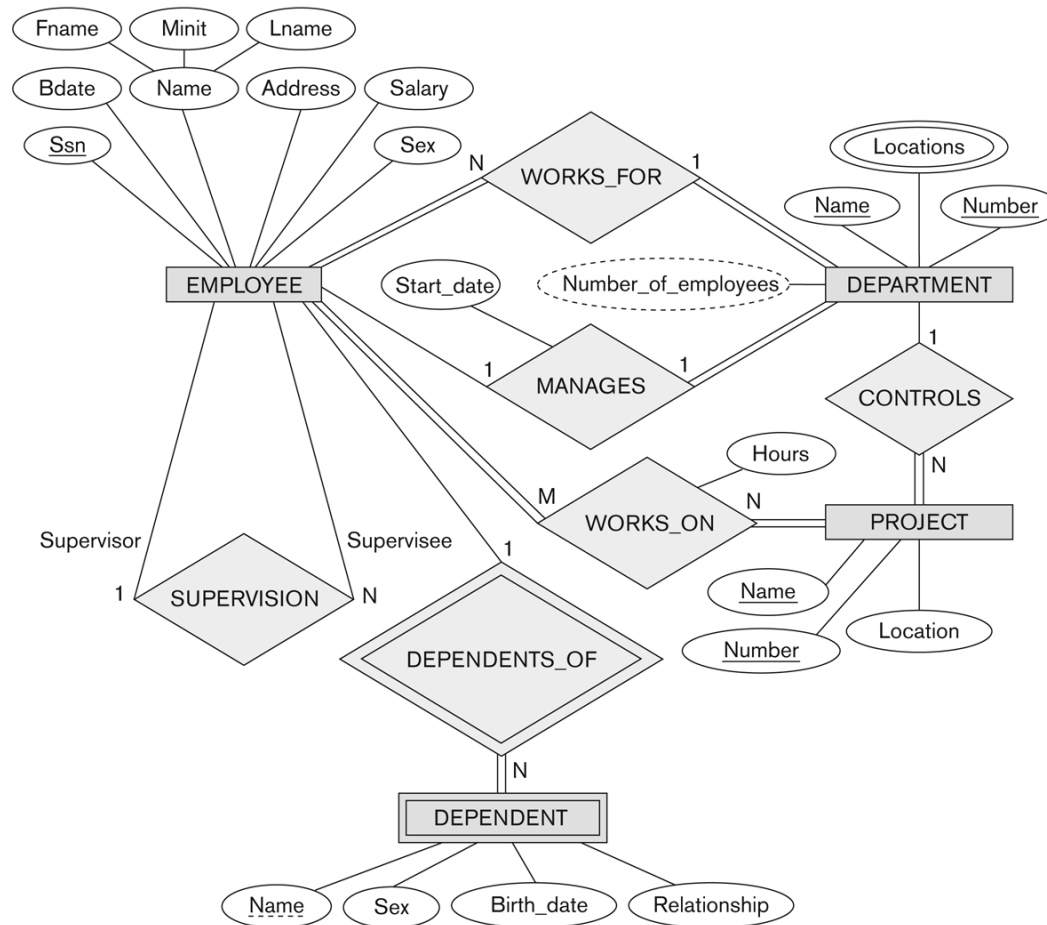
- 키 애트리뷰트가 없고 다른 엔터티 타입의 식별에 의존하는 엔터티.
- 약한 엔터티는 소유자 또는 식별 엔터티 타입과의 식별 관계 타입에 참여해야 함:
  - 약한 엔터티 타입의 부분 키(partial key)
  - 식별 관계 타입에서 관련된 특정 엔터티
- 예제:
  - DEPENDENT 엔터티는 부양 가족의 이름과, 그 부양 가족과 관련된 특정 사원(EMPLOYEE) 엔터티에 의해 식별됩니다.
  - DEPENDENT의 이름(Name)은 부분 키임
  - DEPENDENT는 약한 엔터티 타입임
  - EMPLOYEE는 식별 관계 타입 DEPENDENT\_OF를 통한 식별 엔터티 타입임

# 관계 타입의 애트리뷰트

- 관계 타입은 애트리뷰트를 가질 수 있음:
  - 예: WORKS\_ON의 HoursPerWeek
  - 각 관계 인스턴스의 값은 EMPLOYEE가 PROJECT에서 작업하는 주당 시간 수를 나타냅니다.
    - HoursPerWeek 값은 특정 (사원, 프로젝트) 조합에 따라 다릅니다.
  - 대부분의 관계 애트리뷰트는 M : N 관계와 함께 사용됩니다
    - 1 : N 관계에서는 관계의 N 측에 있는 엔티티 타입으로 옮길 수 있습니다



# 관계 타입의 예제 애트리뷰트: WORKS\_ON의 Hours



**Figure 3.2**

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

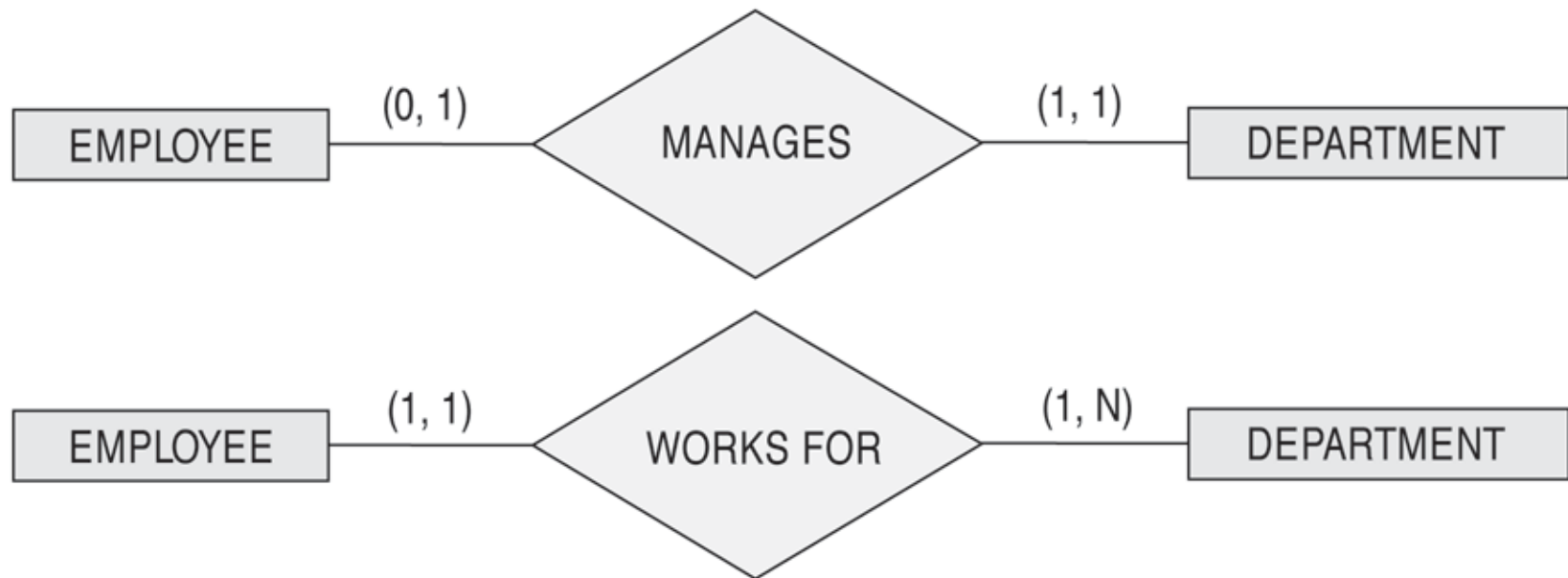
# 관계에 대한 제약조건의 표기

- (이진 관계의) 카디널리티 비율: 1:1, 1:N, N:1, 또는 M:N
  - 관계 간선(edge)에 적절한 숫자를 표시함.
- (각 참여 엔티티 타입에 대한) 참여 제약 조건: 전체 (존재 종속성이라고 함) 또는 부분.
  - 전체 참여는 두줄 선, 부분 참여는 한줄 선으로 표시.
- 주의: 이진 관계 타입에 대해 쉽게 지정할 수 있음.

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

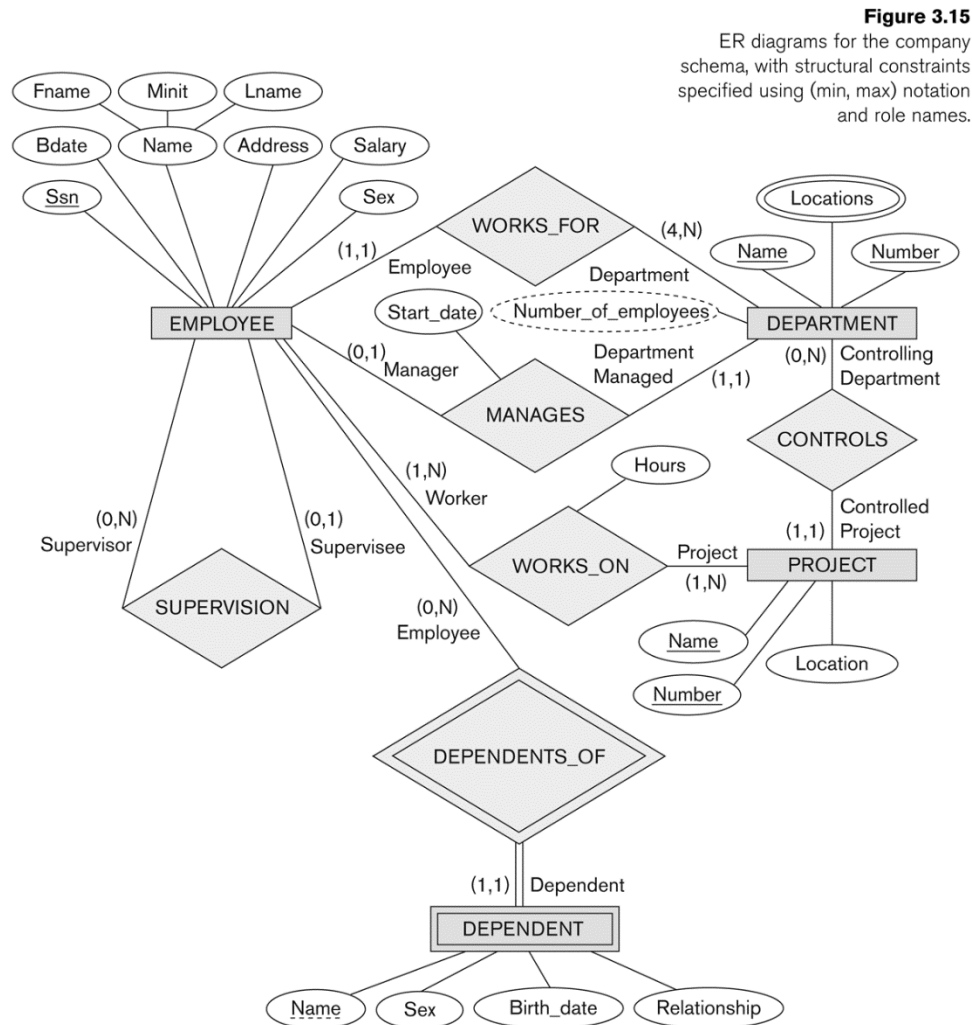
- Specified on each participation of an entity type E in a relationship type R
- Specifies that each entity e in E participates in at least *min* and at most *max* relationship instances in R
- Default(no constraint): min=0, max=n (signifying no limit)
- Must have  $\text{min} \leq \text{max}$ ,  $\text{min} \geq 0$ ,  $\text{max} \geq 1$
- 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 and looking **away from** the entity type

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








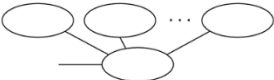

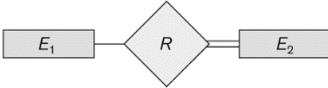

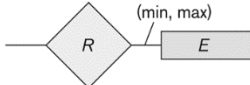


# Alternative diagrammatic notation

- 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
- Appendix A illustrates some of the alternative notations that have been used
- UML class diagrams is representative of another way of displaying ER concepts that is used in several commercial design tools

# ER 다이어그램을 위한 표기법 요약

**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)<br>on Participation of $E$ in $R$ |

# UML class diagrams

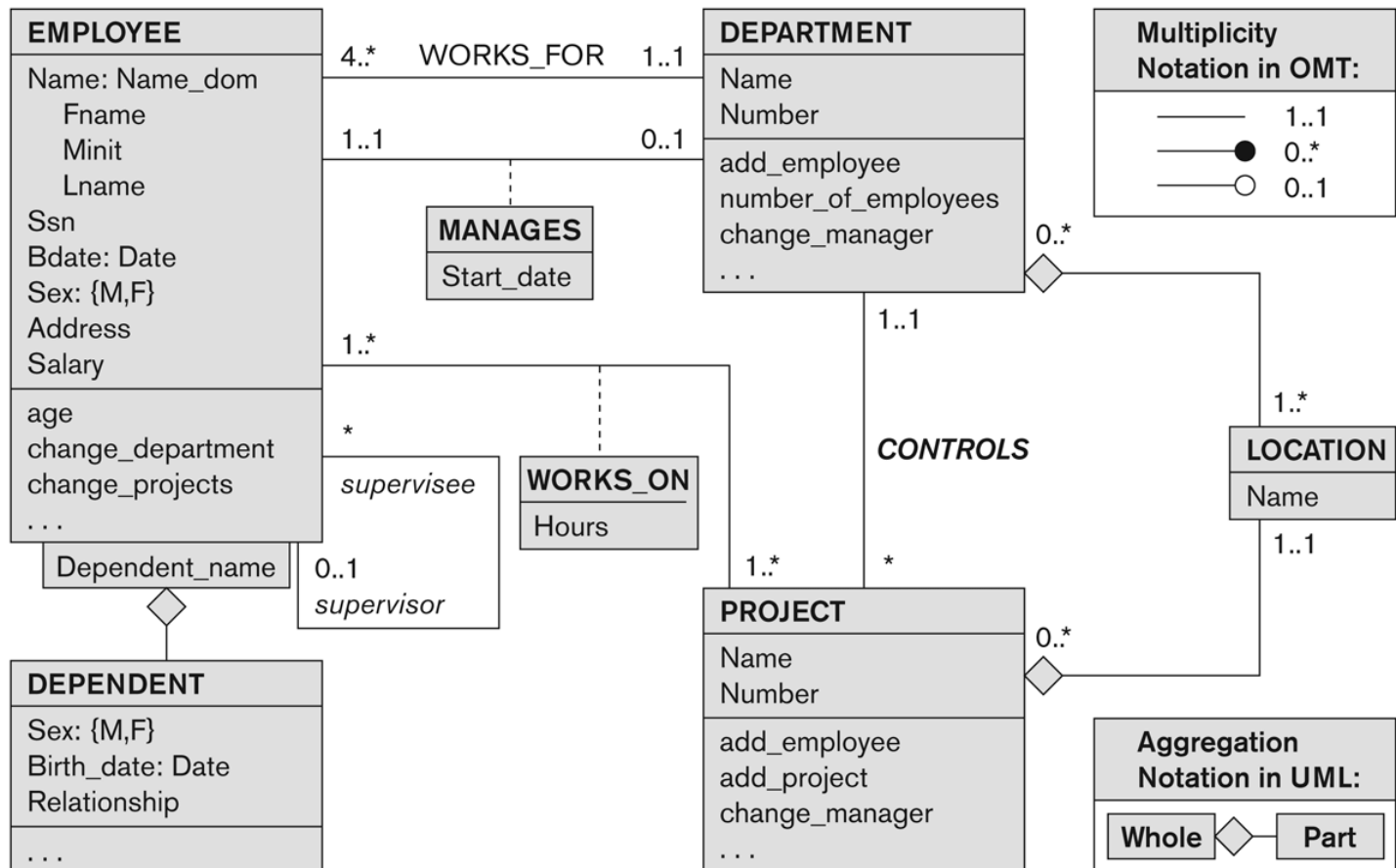
- Represent classes (similar to entity types) as large rounded boxes with three sections:
  - Top section includes entity type (class) name
  - Second section includes attributes
  - Third section includes class operations (operations are not in basic ER model)
- Relationships (called associations) represented as lines connecting the classes
  - Other UML terminology also differs from ER terminology
- Used in database design and object-oriented software design
- UML has many other types of diagrams for software design



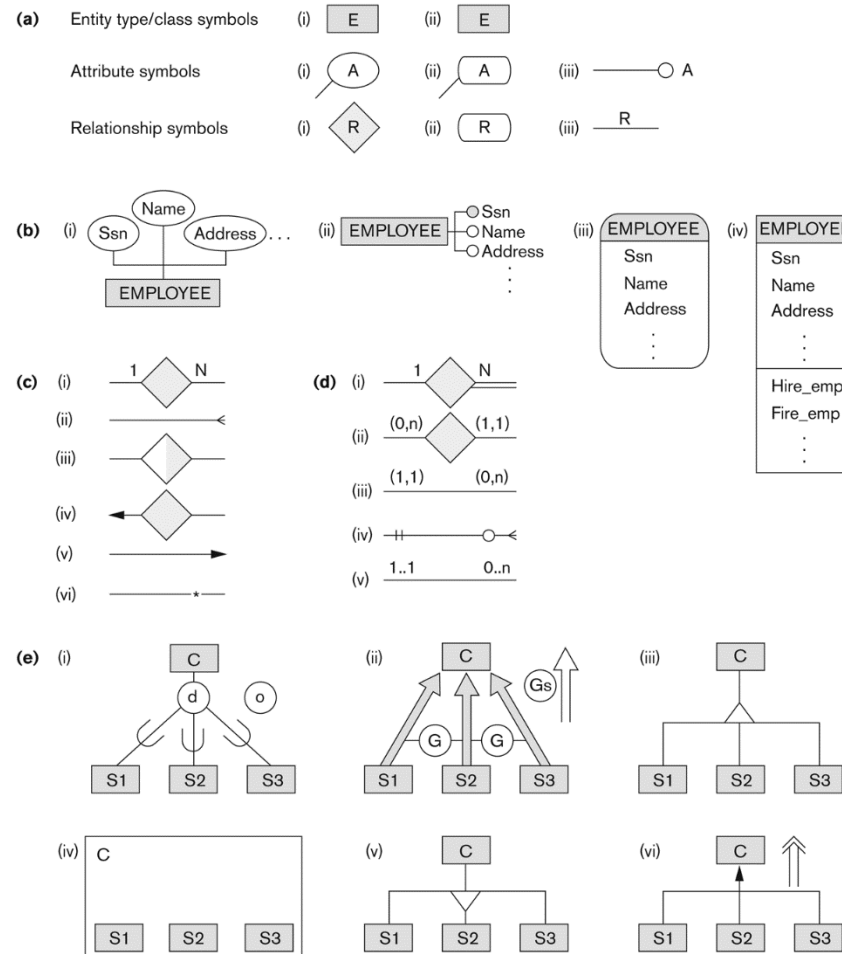
# UML class diagram for COMPANY database schema

**Figure 3.16**

The COMPANY conceptual schema in UML class diagram notation.



# Other alternative diagrammatic notations



**Figure A.1**

Alternative notations. (a) Symbols for entity type/class, attribute, and relationship. (b) Displaying attributes. (c) Displaying cardinality ratios. (d) Various (min, max) notations. (e) Notations for displaying specialization/generalization.

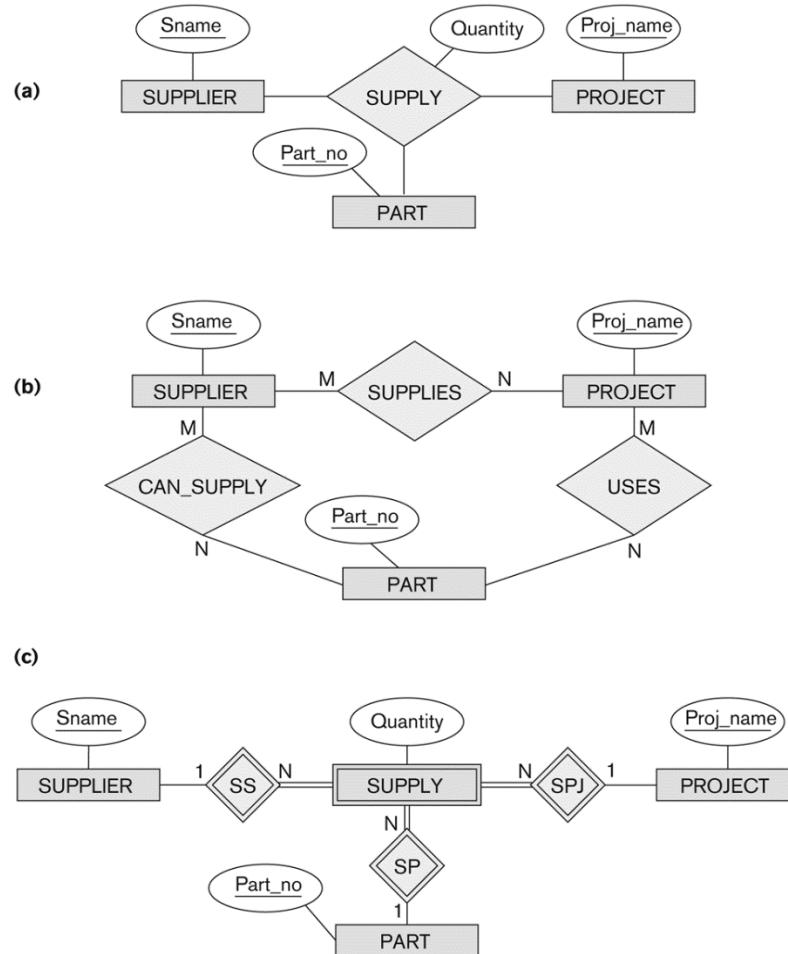
# 상위 차수의 관계

- 차수가 2인 관계 타입을 이진 관계 타입이라고 함
- 차수가 3인 관계 타입을 3차(ternary), 차수가  $n$ 은  $n$ -차( $n$ -ary) 관계 타입이라고 함
- 일반적으로,  $n$ -ary 관계는  $n$  이진 관계와 동일하지 않음
- 이진 관계보다 상위 관계 ( $n > 2$ )에 대한 제약 조건을 지정하기가 더 어려움

# n-차 ( $n > 2$ ) 관계에 대한 토의

- 일반적으로 3 개의 이진 관계는 단일 3차 관계와 다른 정보를 나타낼 수 있음 (다음 슬라이드의 그림 3.17a 및 b 참조).
- 필요한 경우 이진 관계와 n차 관계를 모두 스키마 설계에 포함할 수 있음 (모든 관계가 다른 의미를 전달하는 그림 3.17a 및 b 참조)
- 데이터 모델이 하나의 약한 엔티티 타입에 여러 식별 관계 (따라서 다중 소유자 엔티티 타입)를 갖도록 허용할 경우, 3차 관계는 약한 엔티티로 표시 될 수 있음 (그림 3.17c 참조).

# 3차 관계의 예제



**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.

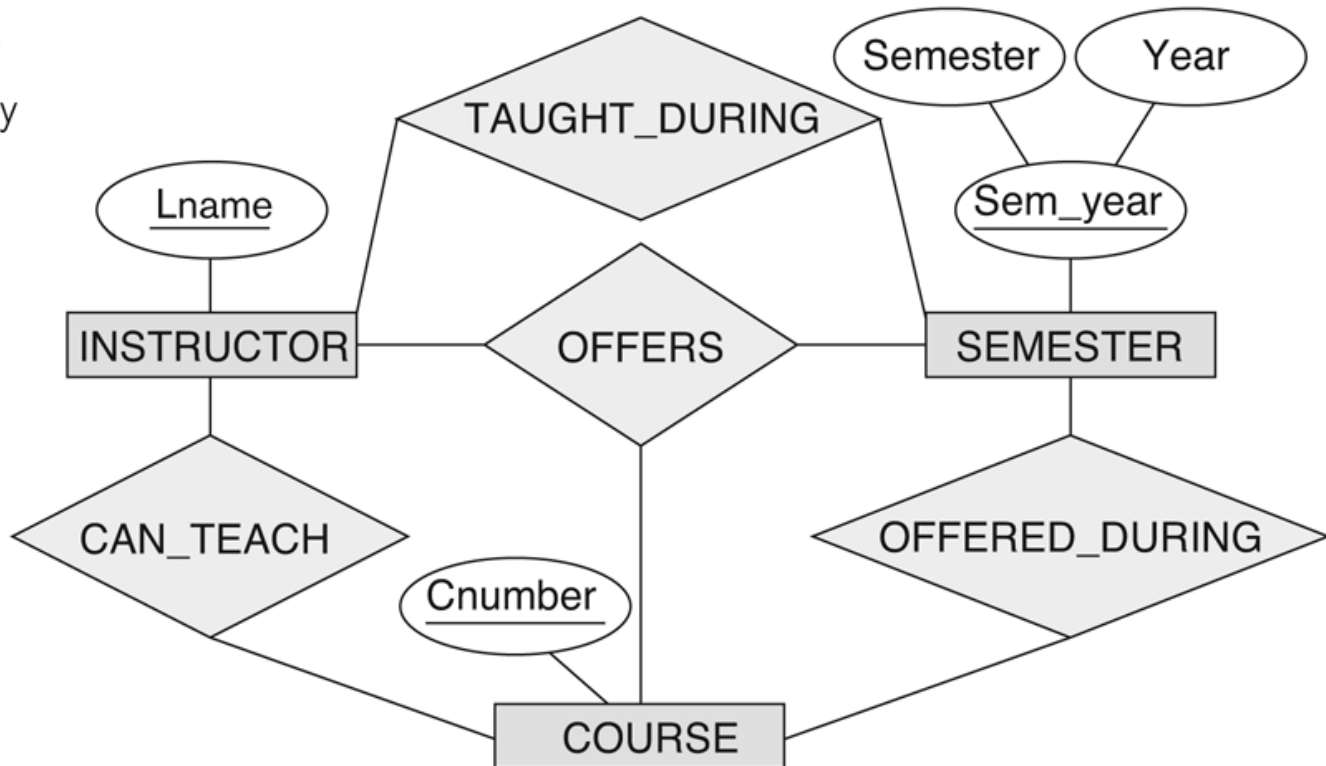
# n-차 ( $n > 2$ ) 관계에 대한 토의

- 특정 이진 관계가 항상 상위 차수의 관계에서 파생될 수 있으면. 이 이진 관계는 중복됨
- 예를 들어, 그림 3.18의 TAUGHT\_DURING 이진 관계 (다음 슬라이드 참조)는 3차 관계 OFFERS (관계의 의미에 따라)에서 파생될 수 있음

# 3차 관계의 또 다른 예제

**Figure 3.18**

Another example of ternary versus binary relationship types.



# Displaying constraints on higher-degree relationships

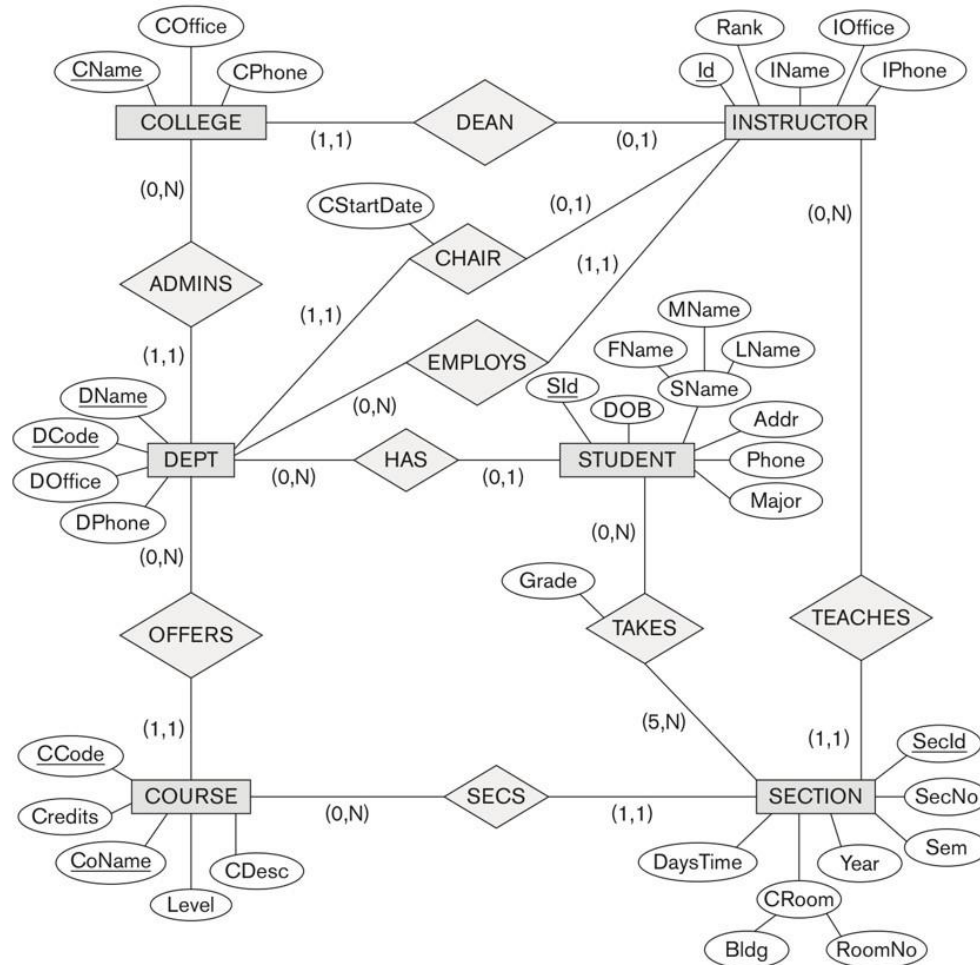
- The (min, max) constraints can be displayed on the edges – however, they do not fully describe the constraints
- Displaying a 1, M, or N indicates additional constraints
  - An M or N indicates no constraint
  - A 1 indicates that an entity can participate in at most one relationship instance *that has a particular combination of the other participating entities*
- In general, both (min, max) and 1, M, or N are needed to describe fully the constraints
- Overall, the constraint specification is difficult and possibly ambiguous when we consider relationships of a degree higher than two.



## 또 다른 예제: UNIVERSITY 데이터베이스

- 수강 신청 및 학생 성적을 관리하기 위한 다른 데이터베이스를 설계함
- 이 데이터베이스는 대학, 각 대학 내 부서, 부서가 제공하는 과목 및 과목의 강좌, 강좌를 가르치는 강사 등을 저장함
- 이러한 엔티티 타입들과 이들 간의 관계는 다음 슬라이드의 그림 3.20에 표시됨

# UNIVERSITY 데이터베이스 개념적 스키마



# Chapter Summary

- ER Model Concepts: Entities, attributes, relationships
- Constraints in the ER model
- Using ER in step-by-step mode conceptual schema design for the COMPANY database
- ER Diagrams - Notation
- Alternative Notations – UML class diagrams, others
- Binary Relationship types and those of higher degree.

# Data Modeling Tools (Additional Material )

- A number of popular tools that cover conceptual modeling and mapping into relational schema design.
  - Examples: ERWin, S- Designer (Enterprise Application Suite), ER- Studio, etc.
- POSITIVES:
  - Serves as documentation of application requirements, easy user interface - mostly graphics editor support
- NEGATIVES:
  - Most tools lack a proper distinct notation for relationships with relationship attributes
  - Mostly represent a relational design in a diagrammatic form rather than a conceptual ER-based design

# Some of the Automated Database Design Tools

(Note: Not all may be on the market now)

| COMPANY                        | TOOL   | FUNCTIONALITY  |
|--------------------------------|--|--|
| Embarcadero Technologies       | ER Studio  | Database Modeling in ER and IDEF1X   |
|                                | DB Artisan   | Database administration, space and security management                       |
| Oracle                         | Developer 2000/Designer 2000                           | Database modeling, application development                                   |
| Popkin Software                | System Architect 2001                                  | Data modeling, object modeling, process modeling, structured analysis/design |
| Platinum (Computer Associates) | Enterprise Modeling Suite: Erwin, BPWin, Paradigm Plus | Data, process, and business component modeling                               |
| Persistence Inc.               | Pwertier   | Mapping from O-O to relational model   |
| Rational (IBM)                 | Rational Rose  | UML Modeling & application generation in C++/JAVA                            |
| Resolution Ltd.                | Xcase  | Conceptual modeling up to code maintenance                                   |
| Sybase                         | Enterprise Application Suite                           | Data modeling, business logic modeling                                       |
| Visio                          | Visio Enterprise                                       | Data modeling, design/reengineering Visual Basic/C++                         |

# Extended Entity-Relationship (EER) Model (in the next chapter)

- The entity relationship model in its original form did not support the specialization and generalization abstractions
- Next chapter illustrates how the ER model can be extended with
  - Type-subtype and set-subset relationships
  - Specialization/Generalization Hierarchies
  - Notation to display them in EER diagrams