

CHAPTER

# 10

## 데이터베이스 설계

1. 데이터베이스 설계
2. 요구사항 분석
3. 개념적 설계
4. 논리적 설계
5. ERwin 실습

### 학습목표

- 데이터베이스 설계 과정을 이해한다.
- 요구사항 분석, 개념적 설계, 논리적 설계 과정을 이해하고 적용 방법을 알아본다.
- ERwin 활용 방법을 학습하고 IE 표기법의 적용 방법을 알아본다.

# 1. 데이터베이스 설계

## ● 데이터베이스의 역할

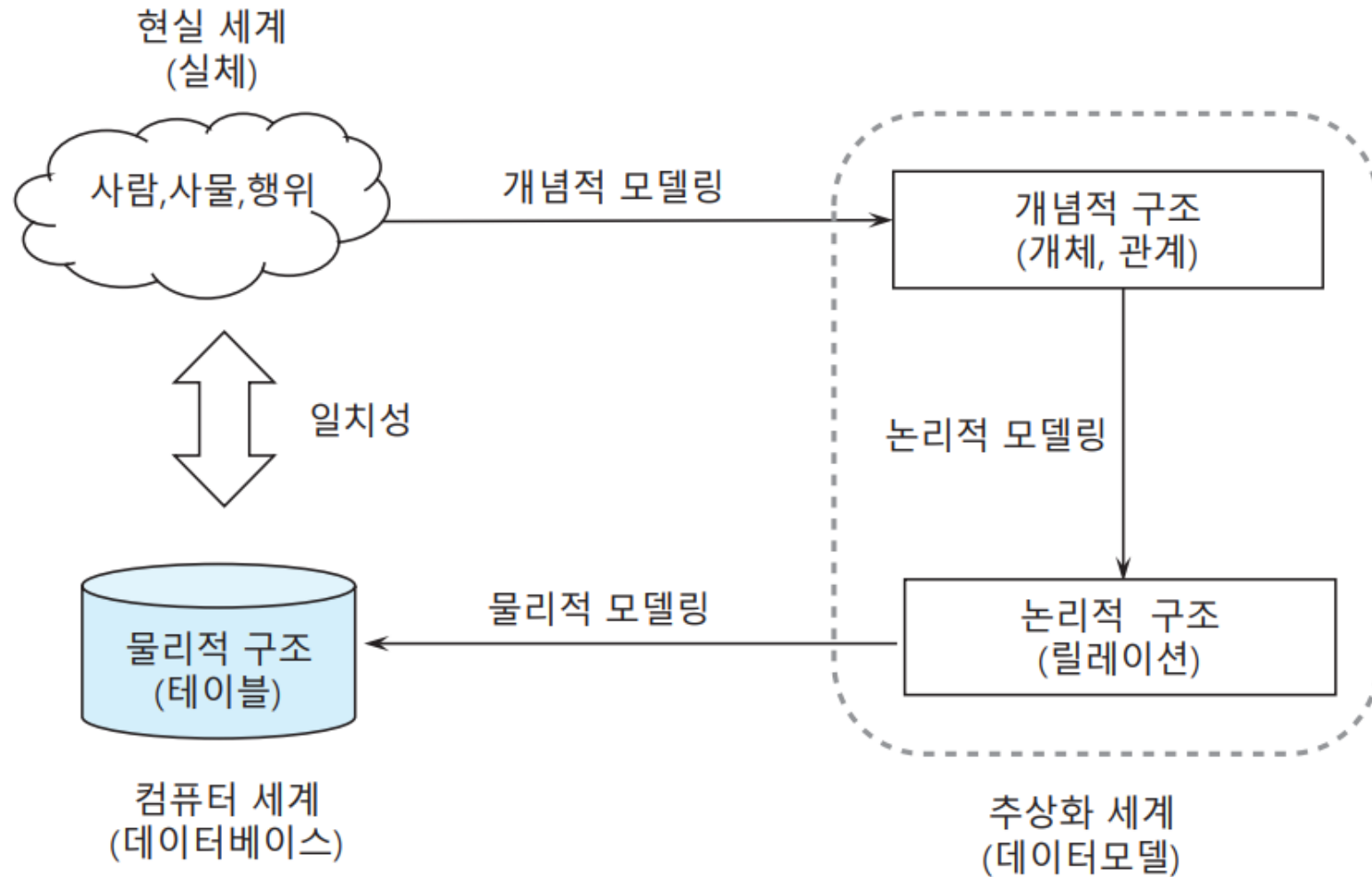


## ● 데이터 모델링(data modeling)

- 데이터베이스 구조를 생성하는 절차적 과정
- 실세계를 개념화하고 논리적 구조로 추상화한 다음 물리적 테이블 구조를 컴퓨터 안에 생성하는 일련의 단계를 거쳐 데이터베이스 구조를 완성

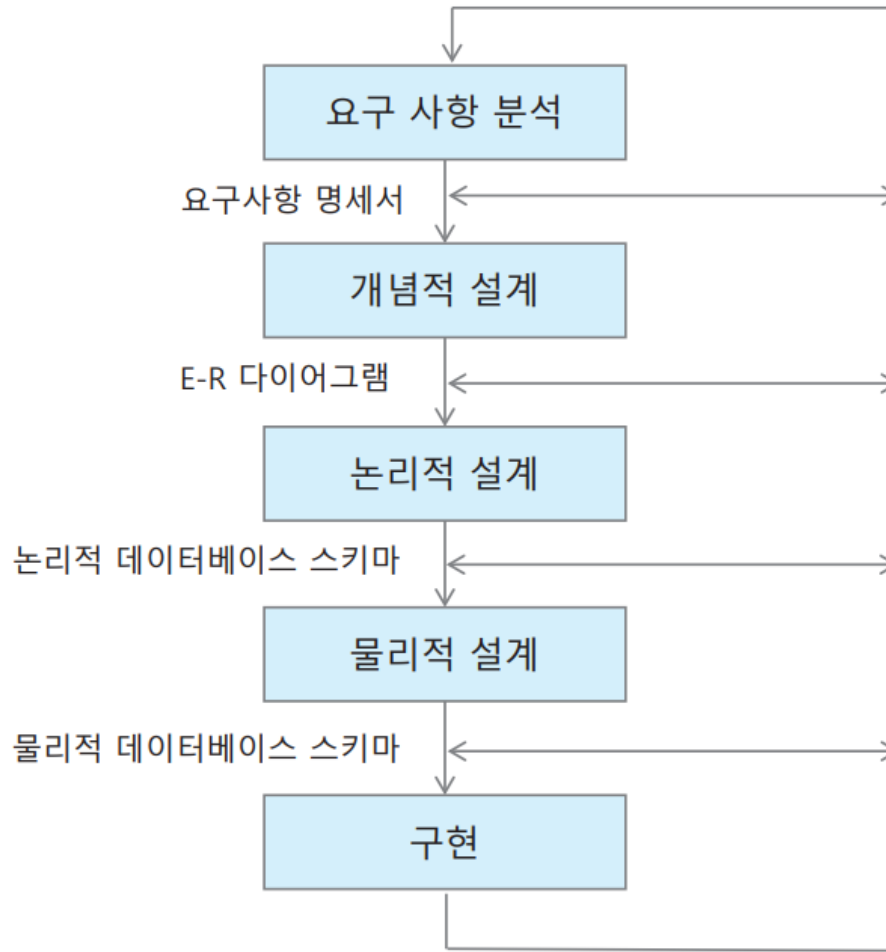
# 데이터베이스 모델링

- 데이터 모델링 과정



## 1.2 데이터베이스의 설계 과정

- 반복적인 설계 과정을 통해 정제된 최종 설계 결과가 생성



## 2. 요구사항 분석

### 2.1 요구사항 명세

- 데이터베이스 설계 과정의 첫 번째 단계
  - 구축하고자 하는 데이터베이스의 구현 범위와 사용자의 범주를 결정
  - 예비 사용자들로부터 요구 사항을 수집하고 업무 처리를 위해 필요한 데이터를 분석
  - 사용자와의 인터뷰, 설문지 조사, 업무 관련 문서의 검토와 기존 시스템 또는 유사 시스템에 대한 분석 등을 포함
- 분석 결과는 요구사항 명세서로 문서화
  - 분석가는 담당자의 요구를 정확하게 파악하도록 노력하고 용어 등이 모호성이 없도록 최대한 명확하게 명세서를 정리
  - 명세서는 분석가의 자의적 해석이나 짐작이 아닌 사용자의 요구 사항만을 충실히 반영하여 구체적 문장 형태로 서술
  - 최종적인 요구사항 명세서는 사용자에게 최종 확인을 받는 과정이 필요

## 2.2 병원 DB 요구사항 명세서

### ● 요구사항 분석 적용 예(병원 DB)

- 병원에 근무하는 모든 의사들은 특정 진료과에 소속되어 많은 환자들을 진료한다. 의사는 전문의와 전공의(인턴, 레지던트)로 구분되며 전공의들을 포함한 일부 의사들은 각각 특정 의사로부터 지도를 받는다.
- 환자는 여러 명의 의사로부터 진료를 받으며 진료날짜, 진료의사, 진료내용 등의 진료 기록들이 유지된다.
- 진료 결과, 수술이 필요한 환자는 병실을 배정받아 입원하게 되며 수술이 끝나 퇴원한 후에도 입원날짜와 퇴원날짜가 보관된다.
- 입원한 수술 환자는 환자 1인당 1명의 간병인을 둘 수 있으며 간병인에 대한 이름과 전화번호, 경력연수 등이 관리된다. 간병인은 많아야 1명의 환자만 간병할 수 있다.
- 모든 병실은 여러 개의 침대를 보유하고 있으며 병실에 대해서는 병실번호와 건물명, 층, 수용인원 등의 정보가 관리되어야 한다. 침대에 대해서도 침대일련번호와 침대종류, 청결상태 등의 정보가 추가로 관리되어야 한다.
- 모든 의사에 대해서는 고유한 의사번호와 이름, 진료과목 등의 정보가 유지되고 진료과에 배정된 날짜도 관리한다. 모든 환자에 대해서는 고유한 주민등록번호와 전화번호, 이름, 나이 등의 신상 정보를 보관한다.
- 의사 중에서 전문의에 대해서는 근무기간 정보가, 전공의(인턴, 레지던트)에 대해서는 수련기간 정보가 유지된다.

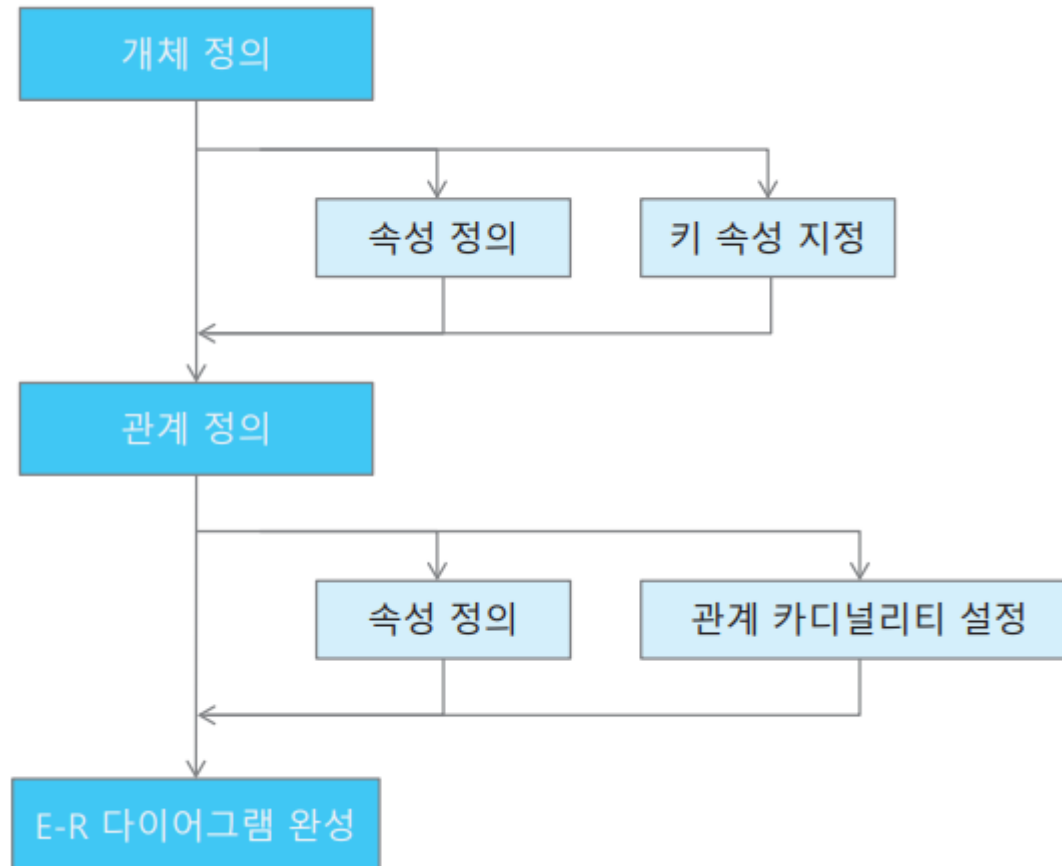
## [참고] 데이터베이스 설계를 단계별로 표로 작성

단계	내용
요구사항 분석	<ul style="list-style-type: none"><li>• 병원 내 의사와 환자 간의 관계 파악</li><li>• 의사 구분 및 소속 진료과 파악</li><li>• 의사 지도 관계 파악</li><li>• 환자의 진료와 입원, 수술 과정 이해</li><li>• 간병인과 환자 간의 관계 이해</li><li>• 병실 및 침대 정보 파악</li><li>• 의사 및 의료진의 세부 정보 수집 및 파악</li></ul>
개념적 설계	<ul style="list-style-type: none"><li>• 의사, 환자, 간병인, 병실, 침대 등의 엔터티 식별</li><li>• 엔터티 간의 관계 도출</li><li>• 속성 및 기본키 식별</li><li>• ER 다이어그램 작성</li><li>• 데이터 모델링 도구를 사용하여 개념적 모델링 수행</li></ul>
논리적 설계	<ul style="list-style-type: none"><li>• 개념적 모델을 논리적 구조로 변환</li><li>• 테이블 정의 및 관계 정의</li><li>• 정규화 및 데이터 무결성 고려</li><li>• 외래키 및 기본키 설정</li><li>• 데이터 모델 최적화를 위한 검토 및 검증</li></ul>
물리적 설계	<ul style="list-style-type: none"><li>• 데이터베이스 시스템 선택 (예: MySQL, PostgreSQL, 등)</li><li>• 테이블 간의 물리적 구조 디자인 (인덱스, 파티셔닝 등)</li><li>• 데이터베이스 인스턴스 생성 및 설정</li><li>• 보안 및 백업 전략 수립</li><li>• 데이터 마이그레이션 계획 수립</li><li>• 테스트 수행 및 최적화</li></ul>

### 3. 개념적 설계

- 개념적 설계(E-R 다이어그램 작성) 과정

- 데이터베이스 설계의 전체 골격을 결정하는 과정
- 요구사항 명세서의 내용을 기반으로 핵심적인 데이터 요소들을 추출하여 E-R 다이어그램을 작성





## 3.1 개체 정의

### ■ 개체 도출

#### 【개체 식별 방법】

- 개체는 저장할 가치가 있는 핵심 데이터를 가진 사람이나 사물, 장소 또는 무형적 개념을 의미한다. 보통 요구 사항 명세서 문장 중에서 주어나 목적어로 표현되는 명사들을 찾는다.
- 명사들은 서로 의미를 명확히 표현하거나 꾸며주는 연관성을 가지고 있다. 속성에 해당하는 하위 개념의 명사들로부터 꾸밈을 받는 상위 개념의 명사가 주로 개체가 된다.
- 개체는 실세계에서 독립적 존재이므로 꾸밈을 받는 하위 개념 명사 중에 고유한 명칭(번호나 코드, 이름 등)을 가진다.
- 개체는 관계와 비교하면 상대적으로 오랜 시간 지속되는 특성이 있다.
- 개체는 보통 또 다른 여러 개체들이 공유하는 대상이다.

### ■ 개체 정의 예



## 3.2 관계 정의

### ■ 관계 도출

#### 【관계 식별 방법】

- 관계는 저장 가치가 있는 데이터를 발생시키는 사건이나 행위와 같은 개체간의 연관성을 의미한다. 요구사항 명세서 문장 중에서 서술어로 표현되는 동사들을 찾는다.
- 관계는 반드시 연관성을 갖는 둘 이상의 개체가 필요하다.
- 대부분의 관계는 의미를 명확히 표현하거나 꾸며주는 하위 개념의 속성을 갖는다.
- 관계는 실세계에서 종속적 존재이므로 보통 하위 개념인 관계 속성 중에 고유한 명칭(번호, 코드, 이름 등)을 갖지 않는다.
- 관계는 개체와 비교하면 일시적이며 상대적으로 짧은 시간만 지속되는 특성이 있다.

### ■ 관계 정의 예



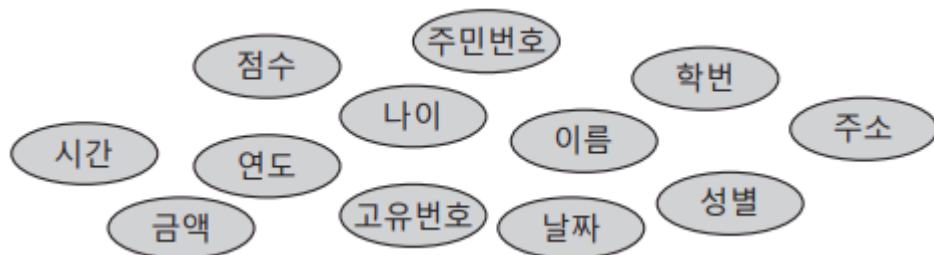
## 3.3 속성 정의

### ■ 속성 도출

#### 【속성 식별 방법】

- 속성은 도출된 개체나 관계의 존재 혹은 상태를 나타내는 특성을 의미한다. 요구사항 명세서 문장 중에서 주어 나 목적어, 서술어를 수식하거나 꾸며주는 하위 명사들을 찾는다.
- 속성은 보통 또 다른 속성들과 함께 공통의 특정 대상을 명확히 표현하기 위해 연관성을 가진다. 개체에 해당하는 상위 개념의 명사나 관계에 해당하는 상위 개념의 동사를 꾸며주는 하위 개념의 명사가 주로 속성이 된다. 속성은 실제 값을 저장한다.
- 속성은 실세계에서 종속적 존재이므로 상위 개념이 반드시 필요하다.

### ■ 속성 정의 예



# 요구사항 명세서의 E-R 다이어그램 변환 규칙

- 개체, 관계, 속성 구별 방법

개체	관계	속성
독립적 존재	종속적 존재	종속적 존재
명사(주어,목적어)로 표현	동사(서술어)로 표현	명사(수식어)로 표현
상위 개념	상위 개념	하위 개념
고유한 명칭(이름,번호) 보유	인위적 명칭(이름,번호) 부여	해당사항 없음
지속적	일시적	지속적/일시적

## [참고] 개체(Entity), 관계(Relationship), 속성(Attribute)

구분	예시	설명
개체	학생, 교수, 강의실	현실 세계에서 독립적으로 식별 가능한 유형이나 객체를 나타냅니다. 각각의 개체는 데이터베이스에서 테이블로 표현될 수 있습니다.
속성	학생의 이름, 교수의 전화번호	개체에 대한 특성이나 속성을 나타냅니다. 각 속성은 개체의 특정 정보를 기술하며, 데이터베이스의 열로 표현될 수 있습니다.
관계	학생이 수강하는 강의, 교수의 지도학생	두 개체 사이의 연결이나 상호 작용을 나타냅니다. 관계는 데이터베이스에서 테이블 간의 관계로 표현될 수 있습니다.

위의 표에서, "학생", "교수", "강의실"은 개체(Entity)를 나타내며, 각각의 개체는 데이터베이스의 테이블로 매핑될 수 있습니다.

"학생의 이름", "교수의 전화번호"와 같은 정보는 속성(Attribute)입니다.

이들은 개체의 특징을 기술하며, 데이터베이스의 열(Column)로 표현됩니다.

"학생이 수강하는 강의", "교수의 지도학생"과 같은 연결은 관계(Relationship)를 나타냅니다. 이러한 관계는 데이터베이스에서 테이블 간의 연결 또는 외래 키(Foreign Key)로 표현됩니다.

# 요구사항 명세서 유의사항

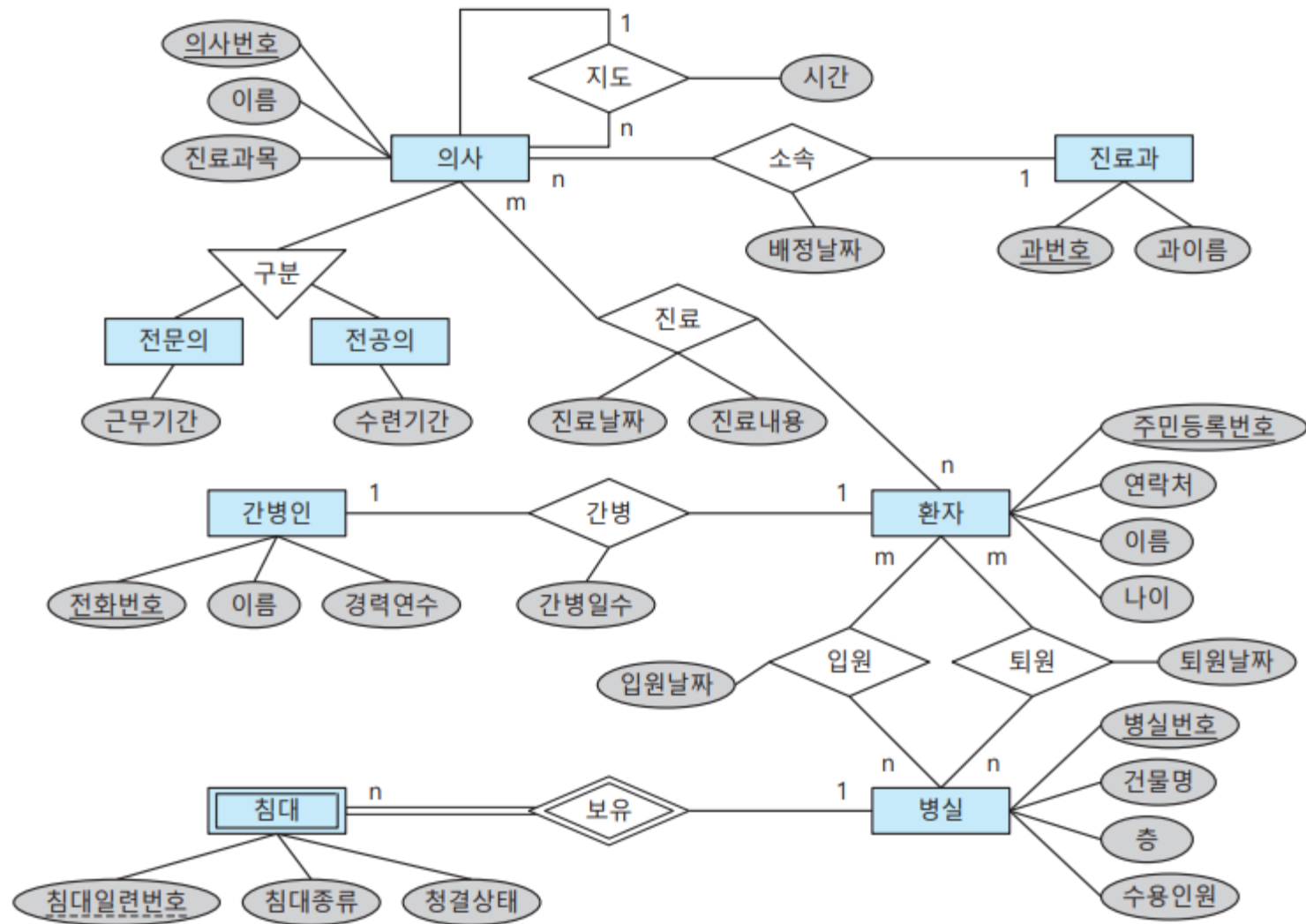
## ● 유의 사항

- 명세서 문장 안에서 개체를 찾아 사각형, 관계는 마름모, 속성은 타원형으로 표시
- 의미가 중복되거나 반복되는 대상은 하나의 표준 용어로 통일하여 표시
- 데이터와 직접적인 연관성이 낮거나 혹은 너무 일반적인 표현은 삭제(가운데줄) 표시
  - 특히, 데이터 관점에서 불필요하거나 기능 관련 표현들은 무시

## ● E-R 다이어그램 작성 시 유의 사항

불필요한 표현	무시해도 좋은 이유
정보, 데이터	모든 특성에 관련되는 공통 표현에 해당되어 속성으로 부적절함
유지, 저장, 관리, 기록, 보관	데이터베이스의 일반 기능에 해당되어 관계로 부적절함
입력, 수정, 삭제, 검색	데이터베이스 시스템의 일반 기능으로 해당되어 관계로 부적절함
발급, 부여, 선택, 확인, 취소	관리자나 사용자의 소프트웨어 기능에 해당되어 관계로 부적절함

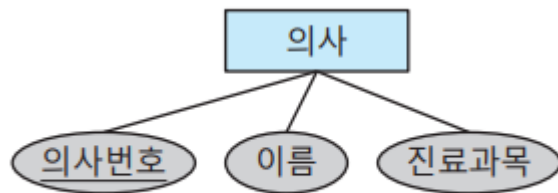
# 병원 DB E-R 다이어그램



## 4. 논리적 설계

### 4.1 개체 변환

- 개체는 기본적으로 하나의 릴레이션으로 변환
  - ✓ 새로 생성된 릴레이션을 E-R 다이어그램의 개체를 표현한다고 해서 개체 릴레이션(entity relation)이라고 함.
  - ✓ 생성된 개체 릴레이션은 개체 이름이 릴레이션의 이름이 됨
- 개체의 키 속성은 릴레이션의 기본키 속성으로, 일반 속성은 릴레이션의 속성으로 변환
  - ✓ 개체의 속성 이름이 릴레이션의 속성 이름이 됨



의사(의사번호, 이름, 진료과목)

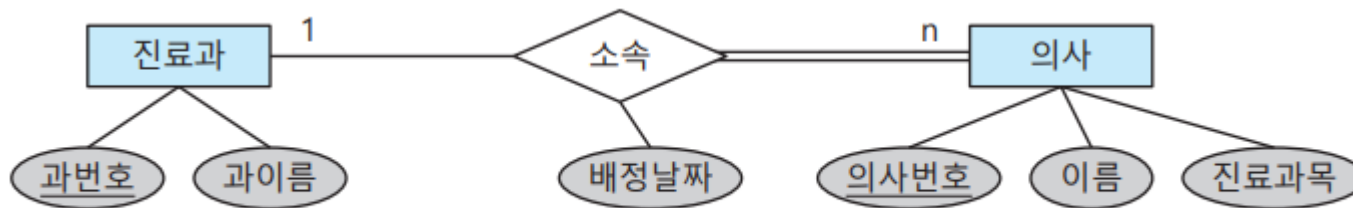
[그림 10-9] 개체(의사)의 예



## 4.2 관계 변환

### ● 일대다(1:n) 관계 변환

- 일대다 관계 자체는 하나의 외래키 속성으로 변환한다.
- 일대다 관계를 갖는 두 릴레이션(개체에서 변환됨)에서 ‘일’(1)측 개체 릴레이션의 기본키 속성을 가져와 ‘다’(n)측 개체 릴레이션에 외래키 속성으로 추가하여 포함시킨다. 추가하는 외래키 속성의 이름에 관계 이름을 포함하도록 변경(예를 들면 ‘과번호’를 ‘소속진료과번호’처럼 변경)하면 그 의미를 명확히 할 수 있다.
- 일대다 관계가 가지고 있던 모든 하위 속성들도 외래키를 추가한 ‘다’(n)측 개체 릴레이션의 속성으로 함께 변환한다.



[그림 10-10] 일대다 관계(소속)의 예

진료과(과번호,과이름)

의사(의사번호,이름,진료과목,소속진료과번호,배정날짜)

# [참고] 일대다 관계

일대다 관계를 하나의 외래키 속성으로 변환하는 것은 간단하게 말하면, 관계를 나타내는 테이블에서 다층 구조를 간소화하여 하나의 외래키로 표현하는 것입니다.

예를 들어, 학교(School)와 학생(Student)이라는 두 테이블이 있을 때, 하나의 학교는 여러 학생을 가질 수 있습니다 (일대다 관계). 이때, 학생 테이블에는 학교를 나타내는 별도의 컬럼(외래키)을 추가하여 일대다 관계를 표현할 수 있습니다.

일대다 관계 변환 전:

학교(School) 테이블	학생(Student) 테이블
학교 ID (School ID)	학생 ID (Student ID)
학교 이름 (School Name)	학생 이름 (Student Name)
	학교 ID (Foreign Key)

일대다 관계 변환 후:

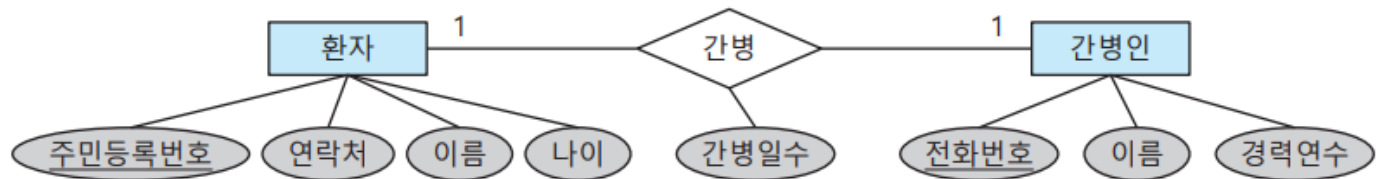
학생(Student) 테이블
학생 ID (Student ID)
학생 이름 (Student Name)
학교 ID (Foreign Key)

여기서 학교(School) 테이블은 제거되었고, 학생(Student) 테이블에 있는 '학교 ID' 컬럼이 일대다 관계를 표현하는 외래키 역할을 하게 됩니다. 이제 하나의 학생은 특정 학교의 ID를 가지고 있어 해당 학교에 속해 있다는 것을 나타내는 방식으로 일대다 관계를 표현합니다.

# 관계 변환

## ● 일대일(1:1) 관계 변환

- 일대일 관계 자체도 일대다 관계의 경우처럼 하나의 외래키 속성으로 변환한다.
- 한쪽 개체 릴레이션의 기본키 속성을 가져와 다른 쪽 개체 릴레이션에 외래키 속성으로 포함시킨다. 어느 쪽 릴레이션에 외래키를 추가하더라도 상관없다.
- 일대일 관계가 갖고 있던 모든 속성들도 외래키를 추가한 릴레이션의 속성으로 함께 변환한다.



[그림 10-11] 일대일 관계(간병)의 예

### <방법1>

환자(주민등록번호,전화번호,이름,나이,간병인전화번호,간병일수)

간병인(전화번호,이름,경력연수,간병환자주민등록번호)

### <방법2>

환자(주민등록번호,전화번호,이름,나이,간병인전화번호,간병일수)

간병인(전화번호,이름,경력연수)

### <방법3>

환자(주민등록번호,전화번호,이름,나이)

간병인(전화번호,이름,경력연수,간병환자주민등록번호,간병일수)

# [참고] 일대일 관계를 하나의 외래키 속성으로 변환

일대일 관계를 하나의 외래키 속성으로 변환하는 방법을 표로 설명해 보겠습니다.  
일대일 관계 변환:

부모(Parent) 테이블	부모(Parent) 테이블
부모 ID (Parent ID – Primary Key)	부모 ID (Parent ID – Primary Key)
다른 속성들 (Other Attributes)	다른 속성들 (Other Attributes)
	자식 ID (Child ID – Foreign Key, Unique Constraint)

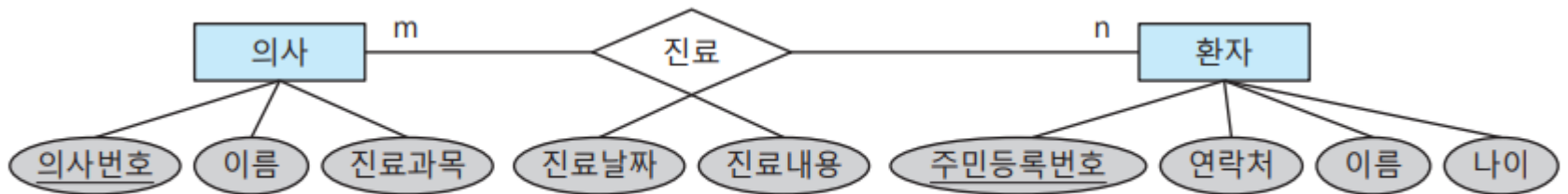
위의 표에서, 일대일 관계 변환 전의 경우 하나의 부모(Parent) 테이블이 있고, 여기에는 부모 ID와 다른 속성들이 포함되어 있습니다.  
일대일 관계를 변환하여 하나의 외래키 속성으로 표현한 후에는 부모(Parent) 테이블에 자식 ID라는 컬럼이 추가되었습니다.  
이 자식 ID 컬럼은 외래키(Foreign Key)로 설정되었고, 부모 테이블과 연결된 일대일 관계를 나타내는 유니크 제약 조건이 적용되었습니다. 이를 통해 일대일 관계를 하나의 외래키 속성으로 효과적으로 변환할 수 있습니다.

\* 유니크 제약조건(Unique Constraint)은 데이터베이스에서 특정 열(컬럼)에 중복된 값을 허용하지 않는 제약 조건입니다. 이 조건을 설정하면 해당 열의 값들이 모두 고유해야 하며, 중복된 값이 들어가는 것을 방지합니다.

# 관계 변환

## ● 다대다(m:n) 관계 변환

- 다대다 관계는 하나의 독립된 릴레이션으로 변환된다. 관계 이름이 새로운 릴레이션의 이름이 된다.
- 관계의 속성은 릴레이션의 속성으로 변환된다. 관계 속성 이름이 새로운 릴레이션의 속성 이름이 된다.
- 새로 생성된 릴레이션을 E-R 다이어그램의 관계를 표현한다는 의미에서 **관계 릴레이션**(relationship relation)이라고 한다. 관계 릴레이션은 보통 키 속성이 없으므로 관계를 맺고 있던 양쪽 개체 릴레이션의 기본키 속성을 외래키 속성으로 포함시키고 양쪽 기본키의 조합을 새로운 기본키로 지정한다. 양쪽 기본키 속성의 이름이 같을 경우, 이름을 다르게 변경한다.



[그림 10-12] 다대다 관계(진료)의 예

의사(의사번호, 이름, 진료과목)

진료(의사번호, 환자주민등록번호, 진료날짜, 진료내용)

환자(주민등록번호, 전화번호, 이름, 나이)

# [참고] 다대다(m:n) 관계 변환

다대다 관계를 효율적으로 표현하기 위해 매개 테이블(Intermediate Table)을 사용하여 다대다 관계를 두 개의 일대다 관계로 변환하는 방법을 예를 들어 설명해보겠습니다.

예를 들어, 학생과 강의 사이에 다대다 관계가 있다고 가정해봅시다. 각 학생은 여러 강의를 수강할 수 있고, 한 강의는 여러 학생들에게 수업을 제공할 수 있습니다.

다대다 관계 변환 전:

학생(Student) 테이블	강의(Class) 테이블
학생 ID (Student ID)	강의 ID (Class ID)
학생 이름 (Student Name)	강의 제목 (Class Title)

다대다 관계 변환 후:이 다대다 관계를 해소하기 위해 매개 테이블을 추가하여 두 개의 일대다 관계로 변환합니다.

학생(Student) 테이블	수강(Course Enrollment) 매개 테이블	강의(Class) 테이블
학생 ID (Student ID)	학생 ID (Student ID, Foreign Key)	강의 ID (Class ID)
학생 이름 (Student Name)	강의 ID (Class ID, Foreign Key)	강의 제목 (Class Title)

위 예시에서 매개 테이블인 '수강(Course Enrollment)' 테이블은 학생과 강의 사이의 관계를 나타냅니다. 이 매개 테이블은 두 테이블 간의 연결 역할을 하며, 각각의 학생과 강의에 대한 관계를 표현합니다. 이렇게 변환함으로써, 다대다 관계를 두 개의 일대다 관계로 분해하여 효과적으로 데이터를 관리할 수 있습니다.

## 4.3 병원DB 논리적 스키마 작성

### ● 병원DB의 논리적 데이터베이스 스키마

진료과(과번호,과이름)

의사(의사번호,이름,진료과목,소속진료과번호,배정날짜,지도의사번호,구분)

전문의(의사번호,근무기간)

전공의(의사번호,수련기간)

진료(의사번호,환자주민등록번호,진료날짜,진료내용)

환자(주민등록번호,전화번호,이름,나이,간병인전화번호,간병일수)

입원(환자주민등록번호,병실번호,입원날짜)

퇴원(환자주민등록번호,병실번호,퇴원날짜)

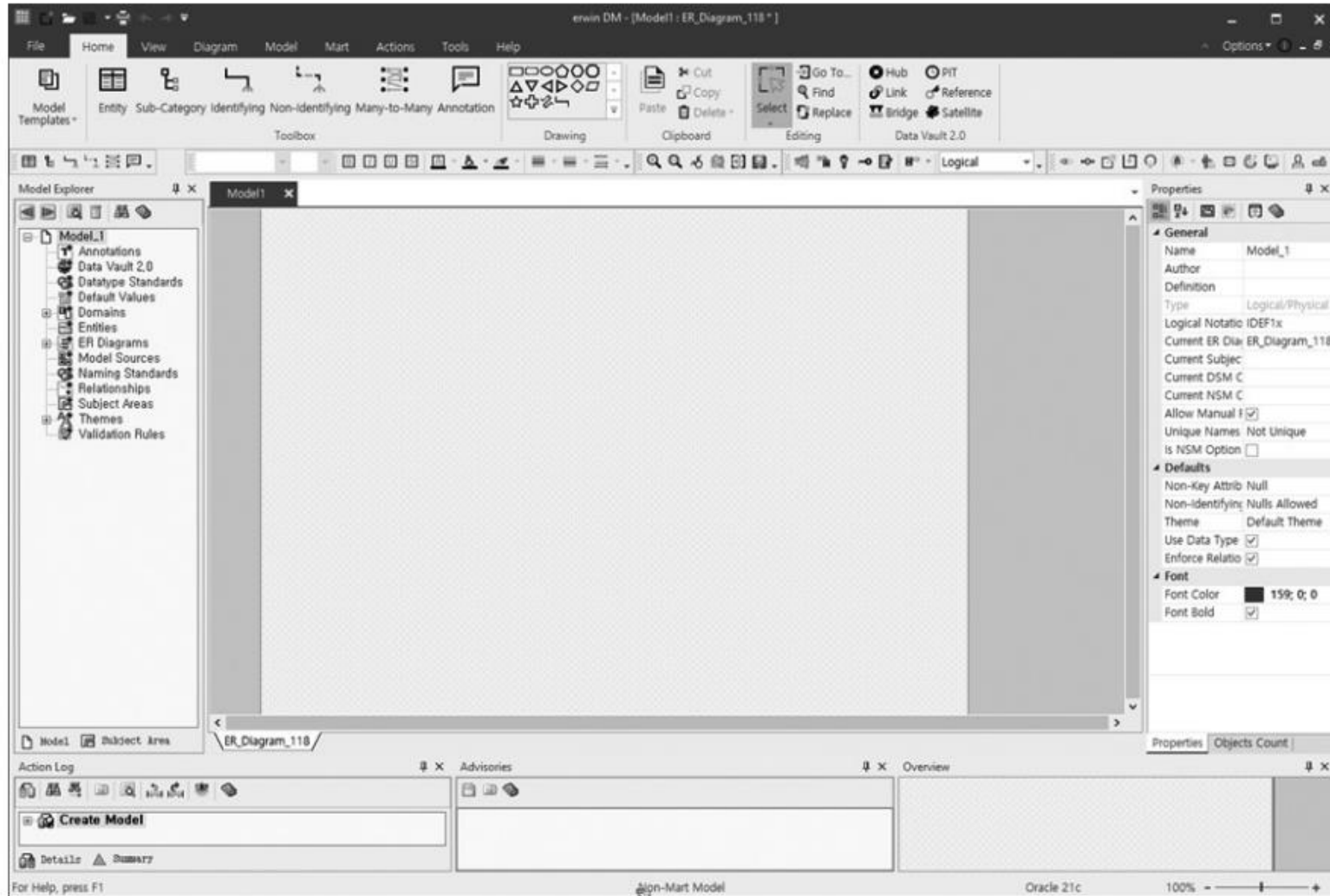
병실(병실번호,건물명,층,수용인원)

침대(보유병실번호,침대일련번호,침대종류,청결상태)

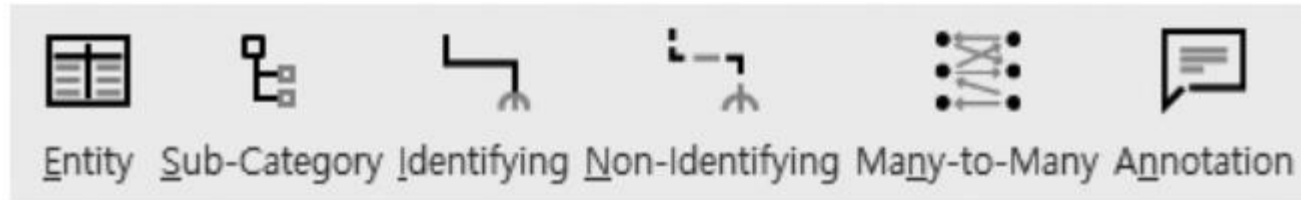


# 5.1 ERwin 화면 구성

## ● 기본 화면







- 개체(Entity) : 개체 유형의 이름, 식별자, 속성을 표현
- 서브 타입(Sub-Category) : IS-A 관계를 갖는 슈퍼(상위) 개체와 서브(하위) 개체를 개체 간의 부모와 자식 관계로 표현
- 일대다 식별(Identifying) 관계 : 두 개체 사이의 1:n 식별 관계를 표현
- 일대다 비식별(Non-Identifying) 관계 : 두 개체 사이의 1:n 비식별 관계를 표현
- 다대다 식별(Many-to-Many) : 두 개체 사이의 m:n 식별 관계를 표현

# ERwin 기본 환경 설정

## ● 모델 유형과 DBMS, 표기법 선택

설정 항목		설정 값
모델 유형 (Type)	Logical Physical Logical/Physical	Logical/Physical
DBMS (Target Server)	ORACLE SQL Server MySQL	ORACLE
표기법 (Notation)	IDEF1x(Integration DEFinition for Information Modeling) IE(Information Engineering)	IE 표기법

IE(Information Engineering) 표기법은 데이터 모델링과 시스템 분석을 위한 표기법 중 하나로, 데이터베이스 설계나 정보 시스템 개발 과정에서 사용됩니다. 이 표기법은 데이터베이스의 구조와 관련된 다양한 개체(Entity), 속성(Attribute), 관계(Relationship)를 나타내기 위해 사용됩니다.

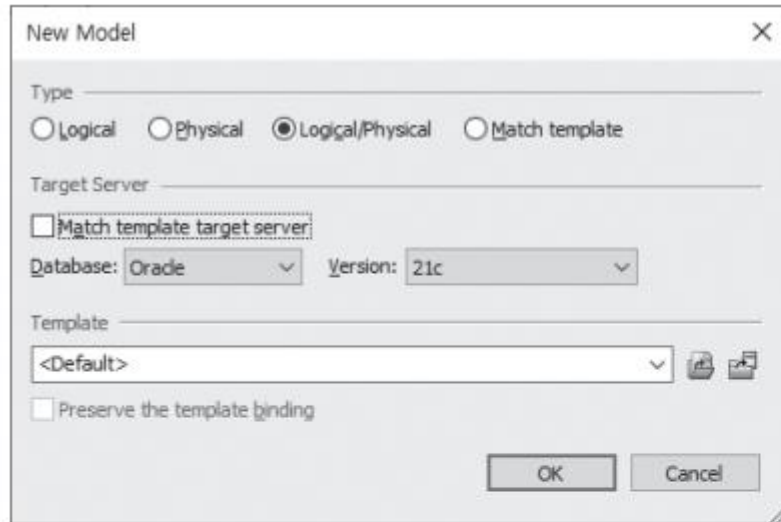
IE 표기법은 주로 그래픽적으로 표현되며, 데이터베이스 구성 요소를 간단하게 나타내기 위해 다양한 기호와 다이어그램을 사용합니다. 주요 구성 요소로는 개체(Entity), 속성(Attribute), 관계(Relationship) 등이 있으며, 이를 사용하여 현실 세계의 데이터를 추상화하고 표현합니다.

개체(Entity)는 현실 세계에서 독립적으로 식별 가능한 것들을 나타내며, 속성(Attribute)은 개체의 특성을 설명합니다. 관계(Relationship)는 데이터베이스 개체들 간의 연결을 나타냅니다.

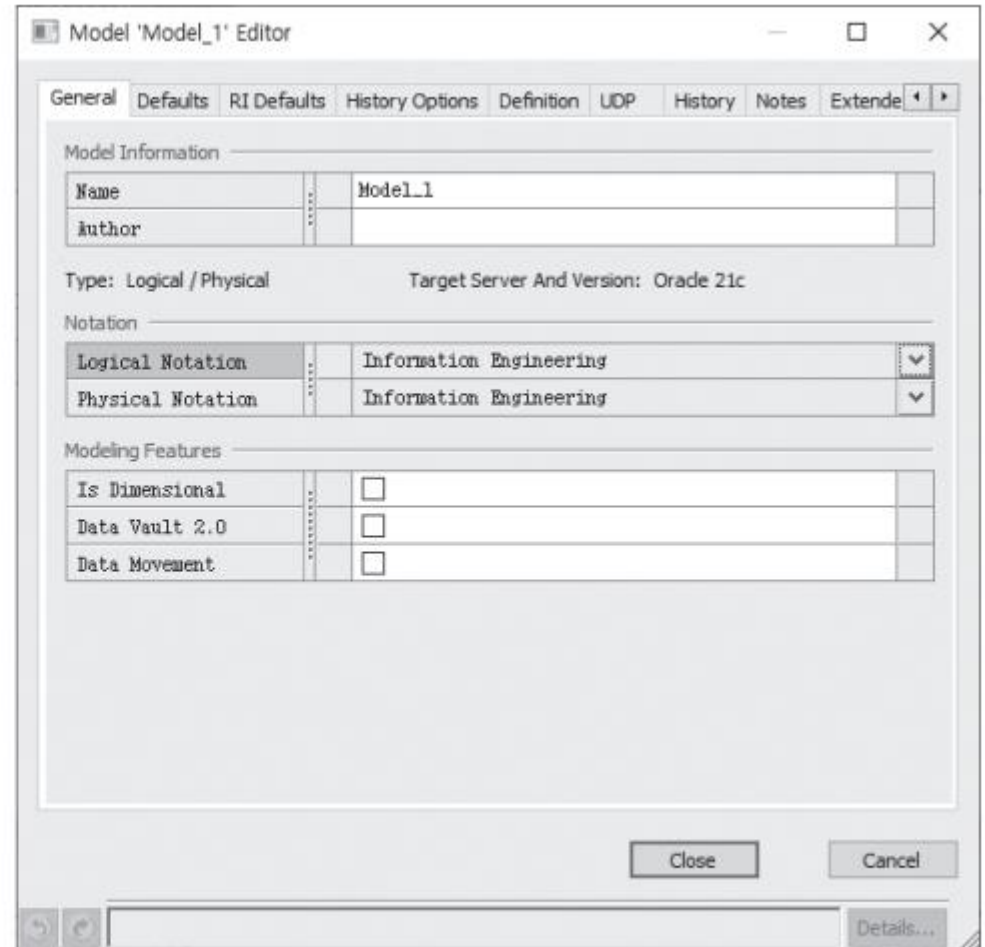
IE 표기법은 다른 표기법과는 다르게 그래픽적인 요소를 많이 사용하여 데이터베이스의 구조를 시각적으로 이해하기 쉽게 설명합니다. 데이터 모델을 개발하고 분석할 때 이 표기법을 활용하여 시스템을 분석하고 설계하는 데 도움을 줍니다.

# ERwin 기본 환경 설정

- 모델 속성 설정 및 IE 표기법 지정



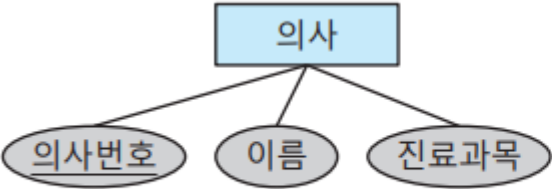
(a) 모델 속성 설정



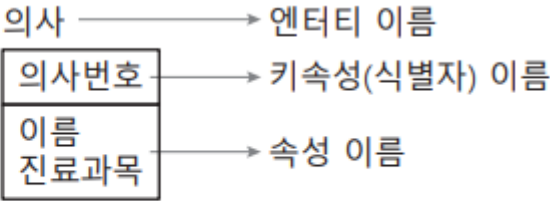
(b) IE 표기법으로 변경

# 5.3 IE 표기법

- 엔터티 표현

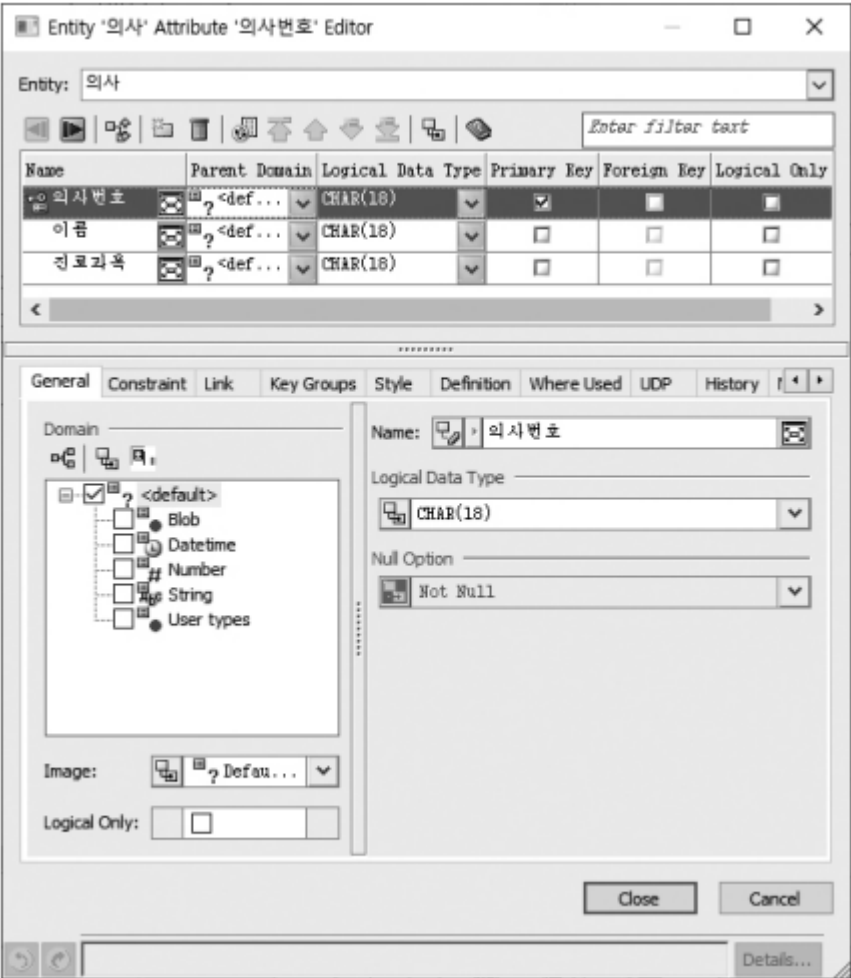


E-R 모델 표기법



ERwin IE 표기법

- 엔터티 속성 입력



## 5.3 IE 표기법

- 관계선 표현



## IE 표기법\_관계선 유형(type)

- 'Non-identifying'

- 비식별 관계, 점선으로, 자식 엔터티를 강 엔터티(직사각형)로 표시
- 툴박스에서 'Non-identifying' 관계선을 선택한 후 부모 엔터티에서 자식 엔터티로 관계선을 연결 -> 부모 엔터티의 키 속성이 사각형 안의 아래쪽 영역에 외래키 속성으로 자동 추가됨

- 'Identifying'

- 식별 관계, 실선으로, 자식 엔터티를 약 엔터티(둥근 사각형)로 표시
- 부모 엔터티에서 자식 엔터티로 Identifying 관계선을 연결 -> 사각형 안의 위쪽 영역에 키 속성으로 외래키 속성이 자동 추가됨
- 대부분의 엔터티는 강 엔터티이므로 보통 관계선은 비식별 관계인 점선으로 표시


# [참고] 비/식별 관계

이러한 차이점들로 인해 식별 관계는 자식 엔터티가 부모 엔터티의 주식별자를 포함하는 반면, 비식별 관계에서는 자식 엔터티의 주식별자는 부모 엔터티의 키를 참조하는 외래 키로만 구성됩니다.

## 식별 관계 (Identifying Relationship)

특징	설명
주요 특징	자식 엔터티가 부모 엔터티의 주 식별자에 종속적이다.
관계 표기 방법	실선과 굵은 화살표로 표시되며, 자식 엔터티와 부모 엔터티를 연결한다.
외래 키	자식 엔터티의 주식별자는 부모 엔터티의 기본 키(주 식별자)를 포함한다.
종속성	자식 엔터티는 부모 엔터티 없이 존재할 수 없다.
예시	주문상세(Order Details)가 주문(Order)에 종속되는 경우

SCSS

 Copy code

## 비식별 관계 (Non-Identifying Relationship)

특징	설명
주요 특징	자식 엔터티가 부모 엔터티의 주 식별자에 직접적으로 종속되지 않는다.
관계 표기 방법	점선과 굵은 화살표로 표시되며, 자식 엔터티와 부모 엔터티를 연결한다.
외래 키	자식 엔터티의 주식별자에 부모 엔터티의 기본 키를 참조하는 외래 키가 생성된다.
종속성	자식 엔터티는 부모 엔터티와 관계 없이도 존재할 수 있다.
예시	주문상세(Order Details)가 고객(Customer)에게 연결된 경우

# IE 표기법\_자식 엔터티의 널 옵션(null option)

- ‘Nulls Allowed’




- 자동 추가되는 외래키 속성의 널 값을 허용함
- 부모 엔터티 쪽의 연결선 끝에 ‘o’ 기호가 표시
  - > 관계선 반대쪽 자식 엔터티의 관계 참여가 선택(optional)임을 의미

- ‘Nulls Not Allowed’

- 자동 추가되는 외래키 속성의 널 값을 허용하지 않음
- 부모 엔터티 쪽의 연결선 끝에 ‘|’ 기호가 표시
  - > 반대쪽 자식 엔터티의 관계 참여가 필수(mandatory)임을 의미

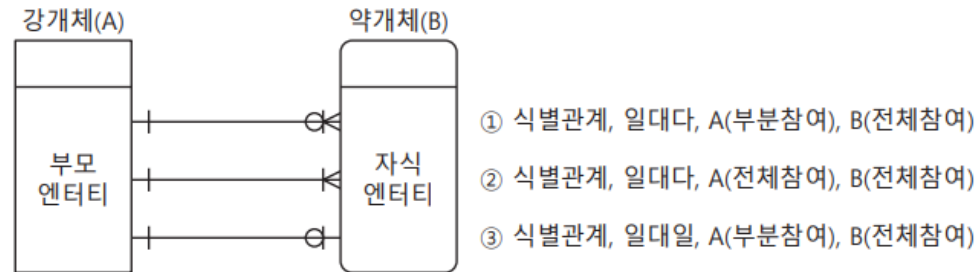


# IE 표기법\_자식 엔터티의 관계 카디널리티(cardinality)

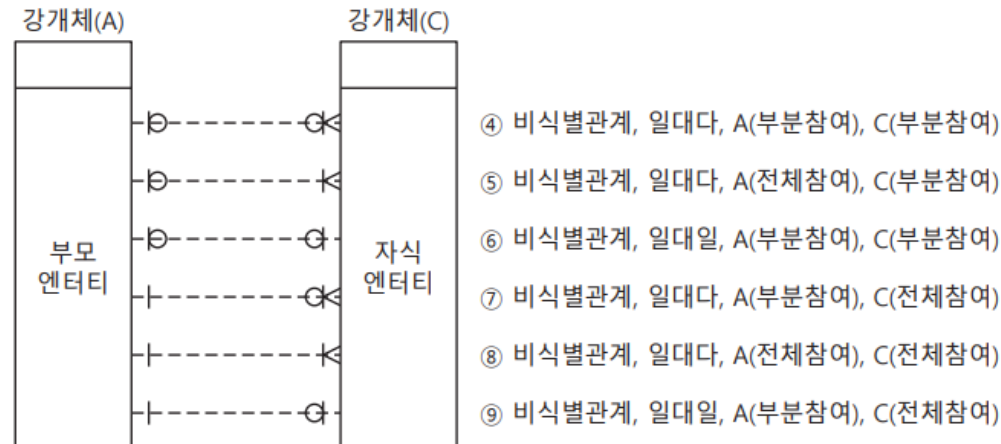
- 관계 카디널리티(cardinality) : 관계에 참여하는 자식 엔터티 인스턴스의 수
- ‘Zero, One or More’ 특성
  - 자식 엔터티의 관계 대응수는 0 이상이고 부모 엔터티가 선택 참여임을 의미
  - ‘’ 기호로 표시
  - 자동 추가된 외래키 속성의 값으로 각 부모 엔터티의 키 속성값이 한 번도 참조되지 않거나 여러 번 참조될 수 있음을 의미
- ‘One or More’ 특성
  - 자식 엔터티의 관계 대응수는 1 이상이고 부모 엔터티가 필수 참여임을 의미
  - ‘’ 기호로 표시
  - 자동 추가된 외래키 속성의 값으로 각 부모 엔터티의 키 속성값이 적어도 한 번 이상 여러 번 참조될 수 있음을 의미
- ‘Zero or One’ 특성
  - 자식 엔터티의 관계 대응수는 0 또는 1이고 부모 엔터티가 선택 참여임을 의미
  - ‘’ 기호로 표시
  - 자동 추가된 외래키 속성의 값으로 각 부모 엔터티의 키 속성값이 한 번도 참조되지 않거나 많아야 한 번만 참조될 수 있음을 의미

# 관계선의 유형

- A는 부모 엔터티, B와 C는 자식 엔터티로 부모 엔터티로부터 자식 엔터티로의 일대일 또는 일대다 관계선을 연결한다.
- ‘|’ 기호는 일대일의 ‘일’(1)을, ‘≡’ 기호는 일대다의 ‘다’(n)를 표시
- 부모 엔터티 쪽의 ‘o’ 기호 -> 자식 엔터티의 부분 참여를 표현
- 자식 엔터티 쪽의 ‘o’ 기호 -> 부모 엔터티의 부분 참여를, ‘o’ 기호가 없으면 전체 참여를 의미



(a) 식별 관계선



(b) 비식별 관계선

# 관계선의 유형



Tip

## 새발(crow-feet) 표기법

IE 표기법은 관계 카디널리티를 새의 발 모양 기호로 표현한다. 자식 엔터티의 관계선 연결부에 표현하는 '1' 기호와 '≤' 기호는 자식 엔터티의 최대사상수가 각각 '1'과 'n'임을 나타낸다. 자식 엔터티의 관계선 연결부에 표현하는 'o'는 부모 엔터티의 최소 사상수가 '부분참여'임을 'o' 기호가 없으면 '전체참여'임을 나타낸다.

### 최소 사상수 (Minimum Cardinality)

종류	설명
부분 참여	관계 중 하나의 엔터티 인스턴스가 다른 엔터티와 연결되지 않을 수 있음을 나타냄.
전체 참여	관계 중 하나의 엔터티 인스턴스가 다른 엔터티와 반드시 연결되어야 함을 나타냄.

예를 들어, '부서(Department)'와 '직원(Employee)' 사이에 관계가 있다고 가정해봅시다.

관계: 부서(Department) - 직원(Employee)

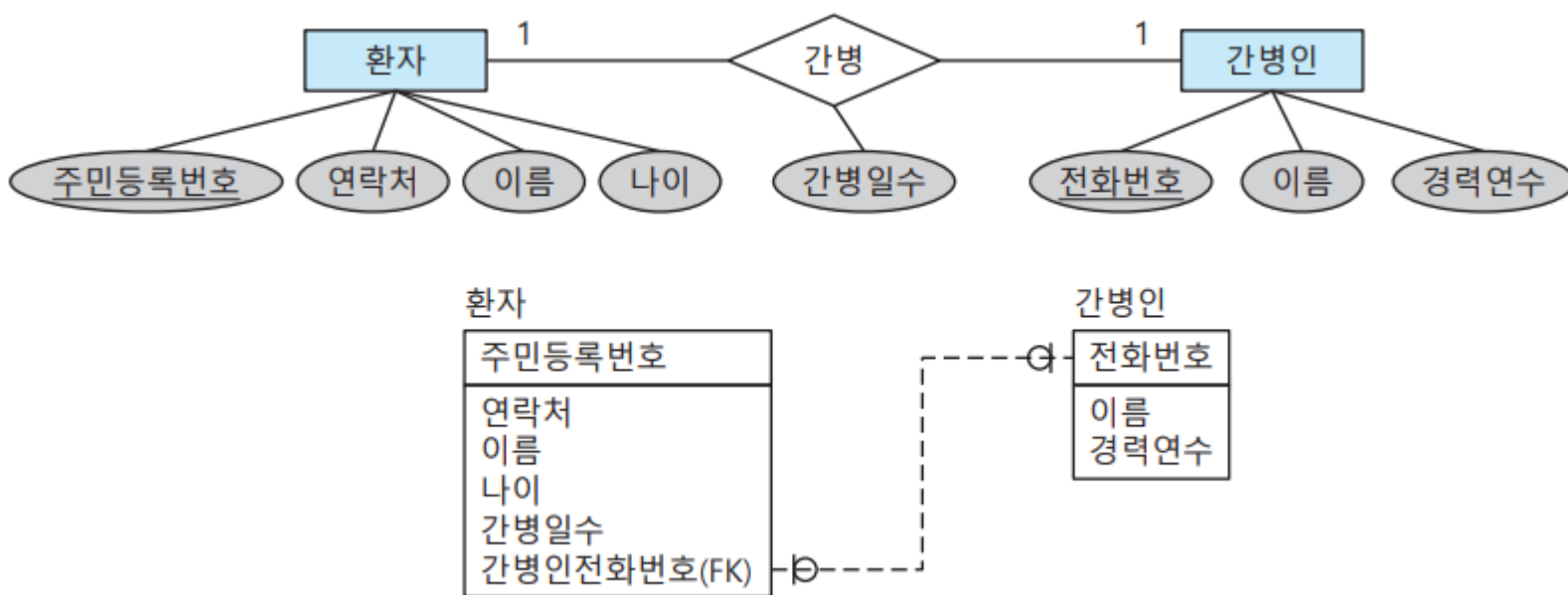
최소 사상수의 유형	설명
부분 참여	모든 부서가 반드시 직원을 가질 필요는 없음. 어떤 부서는 직원을 가지지 않을 수 있음.
전체 참여	모든 부서는 적어도 한 명 이상의 직원을 반드시 가지고 있어야 함.

이를 통해 부분 참여는 어떤 엔터티 인스턴스가 다른 엔터티와 연결되지 않을 수 있지만, 전체 참여는 반드시 연결되어야 함을 나타냅니다.

## 5.4 IE 표기법의 적용 예

### ● 일대일(1:1) 관계 예

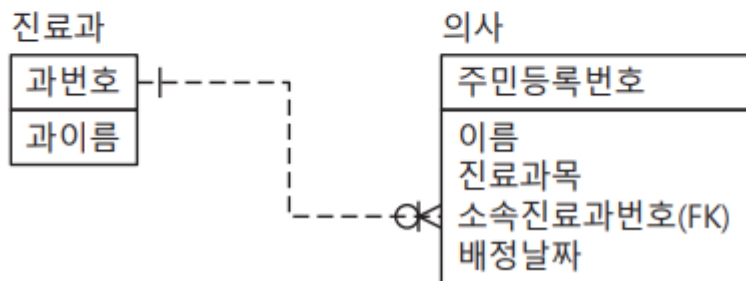
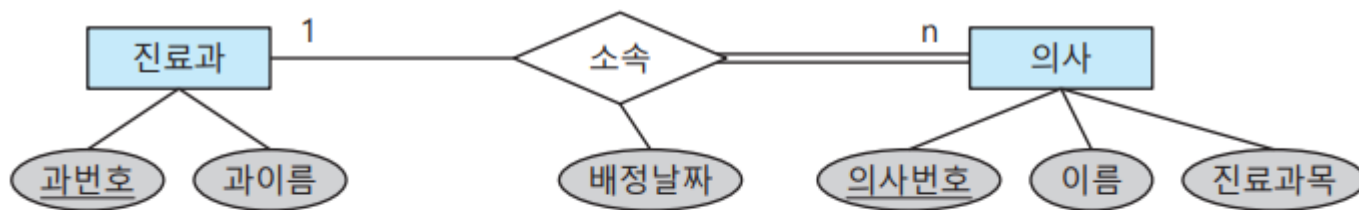
[그림 10-18]에서 ‘간병’ 관계는 강 엔터티 간의 관계이므로 IE 표기법에서 비식별(non-identifying) 관계선인 점선으로 연결한다. ‘간병인’ 엔터티 쪽의 ‘ $\oplus$ ’ 기호를 통해 환자가 ‘간병을 원하지 않을 경우 간병인이 없을 수도 있다(선택)’은 것을 표현한다. ‘환자’ 엔터티 쪽의 ‘ $\oplus$ ’ 기호를 통해 ‘환자는 많아야 한 명의 간병인으로부터 간병 받을 수 있으며 일시적으로 간병하는 환자가 없는 간병인도 있을 수 있다(선택)’는 것을 표현한다.



## 5.4 IE 표기법의 적용 예

### ● 일대다(1:n) 관계 예

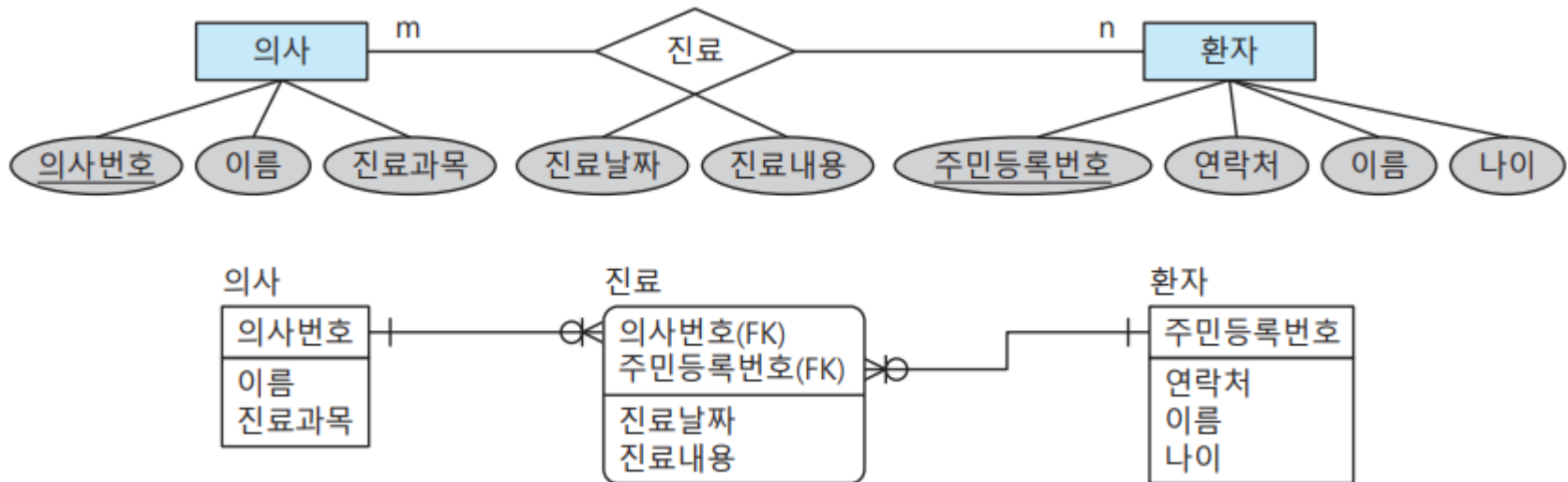
[그림 10-19]에서 '소속' 관계도 강 엔터티 간의 관계이므로 점선으로 연결한다. '진료과' 엔터티 쪽의 '+' 기호를 통해 '의사는 진료과에 반드시 소속되어야 한다(필수)'는 것을 표현한다. '의사' 엔터티 쪽의 '0' 기호를 통해 '진료과는 의사를 0명 이상 포함한다'라는 것을 표현한다. 이는 '의사가 없는 진료과(예를 들면 신설 진료과인 경우)는 있을 수 있다(선택)'는 의미이다. '의사' 엔터티에 자동 추가된 '과번호' 외래키 속성의 이름을 '소속' 관계의 의미를 살려 '소속진료과번호'로 변경한다.



# IE 표기법의 적용 예

## ● 다대다(m:n) 관계 예

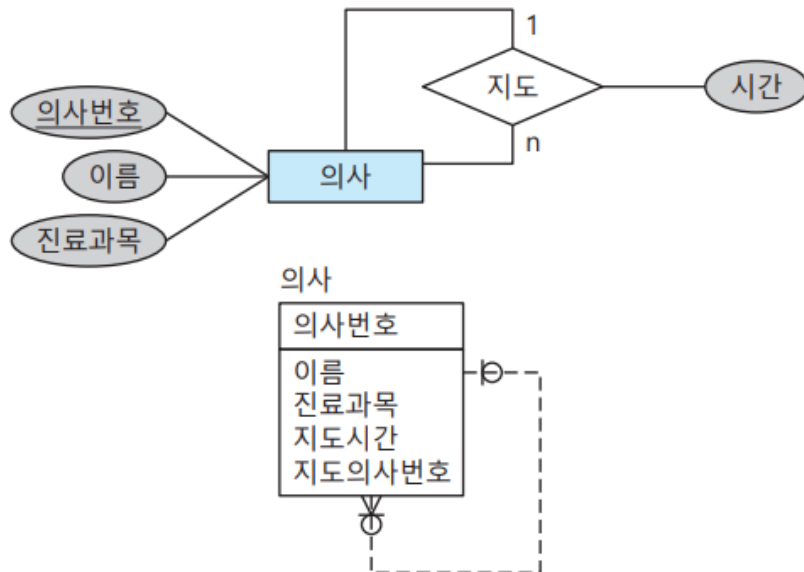
[그림 10-20]에서 '진료' 관계는 강 엔터티 간의 다대다 관계로 2개의 식별(identifying) 관계선으로 분리하여 실선으로 연결한다. 이를 위해 2개의 식별 관계선을 연결하는 **중간 엔터티**를 생성한다. 관계 이름이었던 '진료'가 새로운 중간 엔터티의 이름이 된다. m:n '진료' 관계는 '의사' 엔터티와 '진료' 중간 엔터티, '환자' 엔터티와 '진료' 중간 엔터티 간의 두 개의 1:n 관계로 변환하여 표현한다. '의사'와 '환자' 엔터티에서 '진료' 엔터티로 관계선을 연결하면 자동적으로 '의사'와 '환자' 엔터티의 기본키가 '진료' 엔터티의 외래키로 사각형 안의 위쪽 영역에 키 속성으로 자동 추가된다.



# IE 표기법의 적용 예

## ● 순환 관계 예

엔터티가 자신과 관계를 맺는 순환 관계를 IE 표기법으로 표현하면 [그림 10-21]과 같다. '지도' 관계는 강 엔터티인 '의사' 자신과의 관계이므로 비식별(non-identifying) 관계, 실선으로 연결한다. 지도를 받는(예를 들면 인턴, 레지던트) '의사' 엔터티는 지도교수 역할을 하는 '의사' 엔터티 쪽에 표현된 '⌘' 기호를 통해 '지도교수 의사로부터 지도를 받지 않을 수도 있다(선택)'는 것을 표현한다. 지도교수 역할을 하는 '의사' 엔터티는 지도학생에 해당하는 '의사' 엔터티 쪽에 표현된 '⌘' 기호를 통해 '의사(지도교수)는 의사(인턴, 레지던트)를 0명 이상 여러 명까지 지도한다(선택)'라는 것을 표현한다. 결과적으로 지도교수가 없는(있더라도 최대 1명뿐임) 의사와 어느 누구도 지도하지 않는(지도교수가 아닌) 의사도 가능하다는 의미이다.



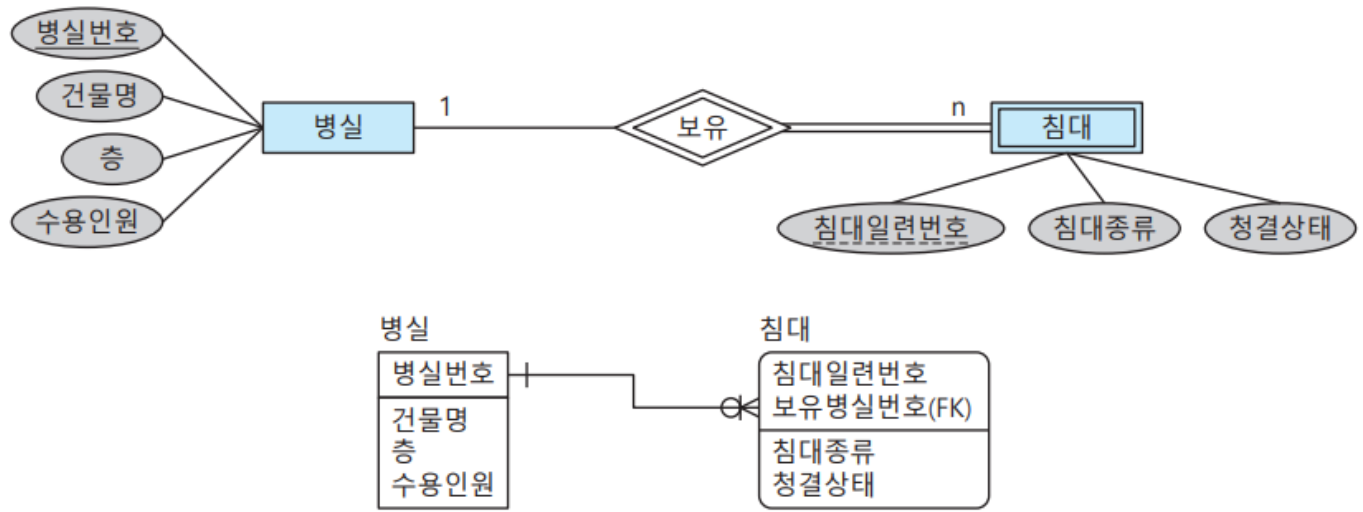


# IE 표기법의 적용 예

## ● 약 엔터티의 식별 관계 예

[그림 10-22]는 약 엔터티의 IE 표기법 예를 보여준다. ‘보유’ 관계는 약 엔터티를 포함하는 식별 관계이므로 약 엔터티인 ‘침대’ 엔터티의 키 속성으로 부모 엔터티인 ‘병실’ 엔터티의 식별자인 ‘병실번호’ 속성을 포함시킨다. 병실마다 같은 번호를 가진 침대가 있을 수 있으므로 침대 일련번호만으로는 어느 병실의 침대인지 알 수가 없기 때문이다. 식별 관계는 관계선을 실선으로 표시하며 약 엔터티는 둥근 사각형으로 구별된다.

‘병실’ 엔터티에서 ‘침대’ 엔터티로 식별(identifying) 관계선을 연결하면 자동적으로 ‘병실’ 엔터티의 기본키가 ‘침대’ 엔터티의 외래키로, 사각형 안의 위쪽 영역에 키 속성으로 자동 추가된다. 추가된 외래키 속성은 ‘보유병실번호’로 이름을 변경한다.





# [참고] 병실과 침실 사이의 관계를 통해 약 엔터티와 식별 관계의 예

약 엔터티 (Weak Entity)	식별 관계 (Identifying Relationship)
병실 (Ward)	침실 (Bed)
병실 번호 (Ward Number)	병실 번호 (Ward Number - 외래 키 및 식별자)
침대 ID (Bed ID)	침대 정보 (Bed Information)

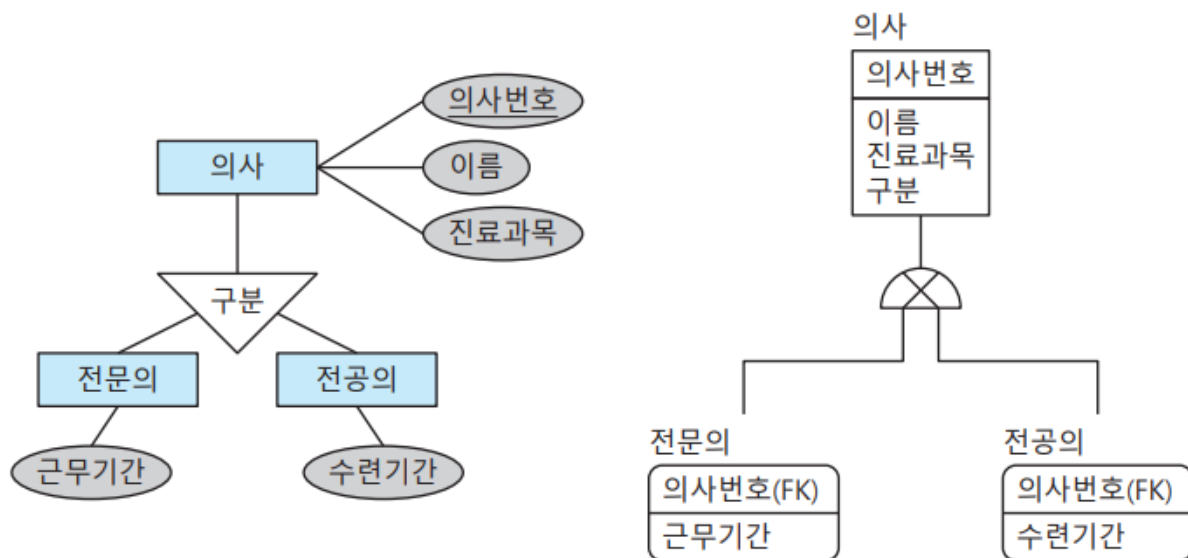
이 예시에서, '병실(Ward)'은 약 엔터티입니다. 각 병실은 여러 개의 침대('침실(Bed)')를 가지고 있지만, 병실 자체만으로는 고유하게 식별되기 어렵습니다.

- '침실(Bed)'은 '병실(Ward)'에 의해 식별됩니다. '병실 번호(Ward Number)'는 '침실' 엔터티의 외래 키(Foreign Key)이자 동시에 식별자(Identifier) 역할을 수행하여 침실을 고유하게 식별합니다.
- '침실(Bed)' 엔터티에는 침대 ID와 침대에 관한 정보를 담는 속성들이 있습니다.
- 각 '병실(Ward)'은 여러 '침실(Bed)'을 가질 수 있지만, 각 '침실'은 특정 '병실'에 속해 있어야 합니다. 이를 통해 '병실(Ward)'이 약 엔터티이며, '침실(Bed)'과의 관계를 통해 식별 관계가 형성되고 있음을 확인할 수 있습니다. '침실'은 '병실'에 종속되며, '병실'을 식별하는데 필요한 주요한 구성 요소가 됩니다.

# IE 표기법의 적용 예

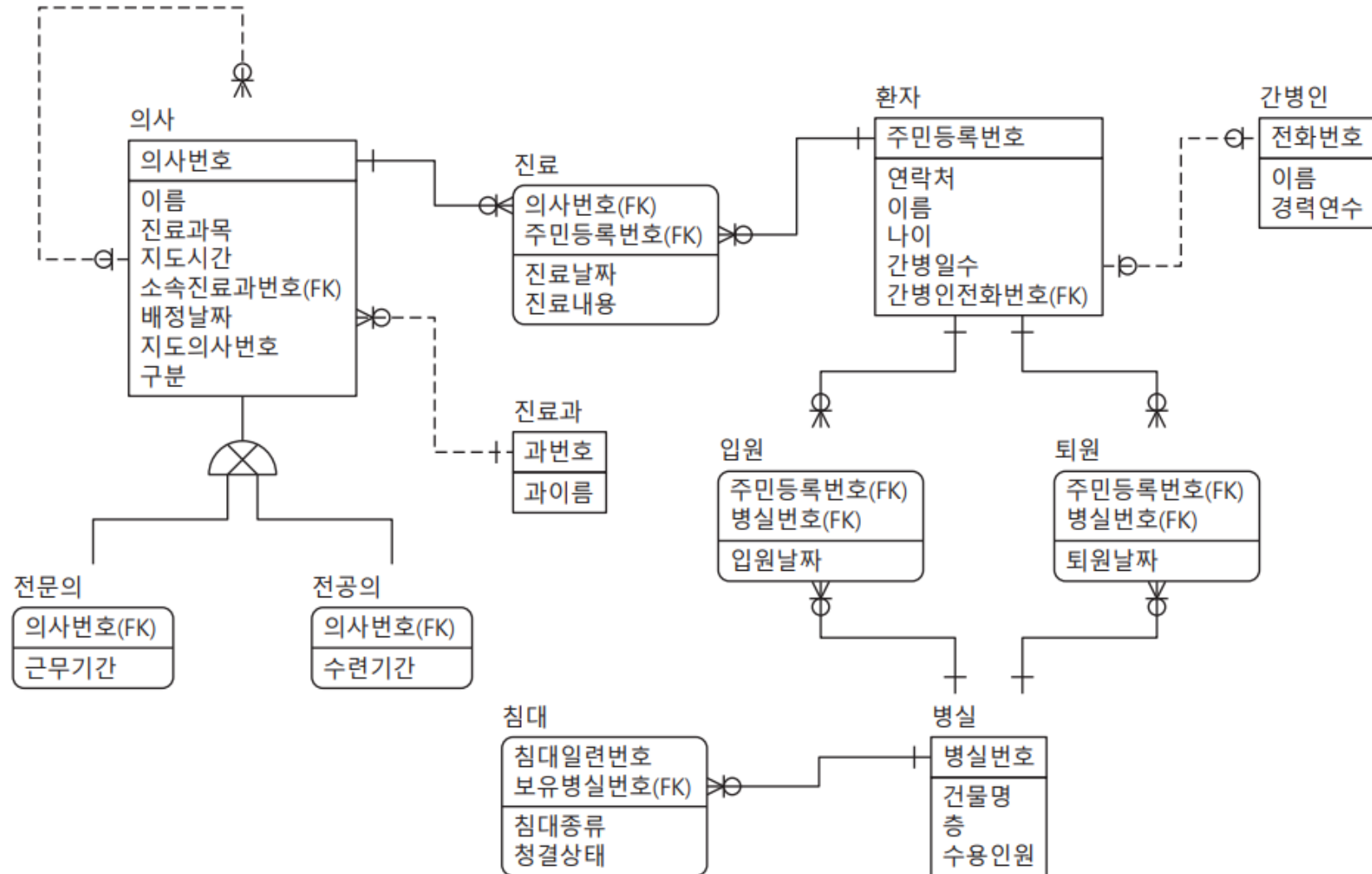
## ● 일반화 관계 예

[그림 10-23]은 상위 엔터티와 하위 엔터티들 사이의 일반화 관계를 IE 표기법으로 표현한 예이다. 가장 간단한 방법은 상위 엔터티와 하위 엔터티들을 각각의 독립된 엔터티로 표현하는 것이다. 먼저 상위 엔터티에는 ‘구분’ 일반화 관계를 표현하기 위한 ‘구분’ 일반 속성이 추가된다. 상위 엔터티의 키 속성인 ‘의사번호’는 하위 엔터티의 키 속성으로 식별자 역할을 하기 위해 상속되어 추가된다. 일반화 관계선도 실선으로 표시된다.



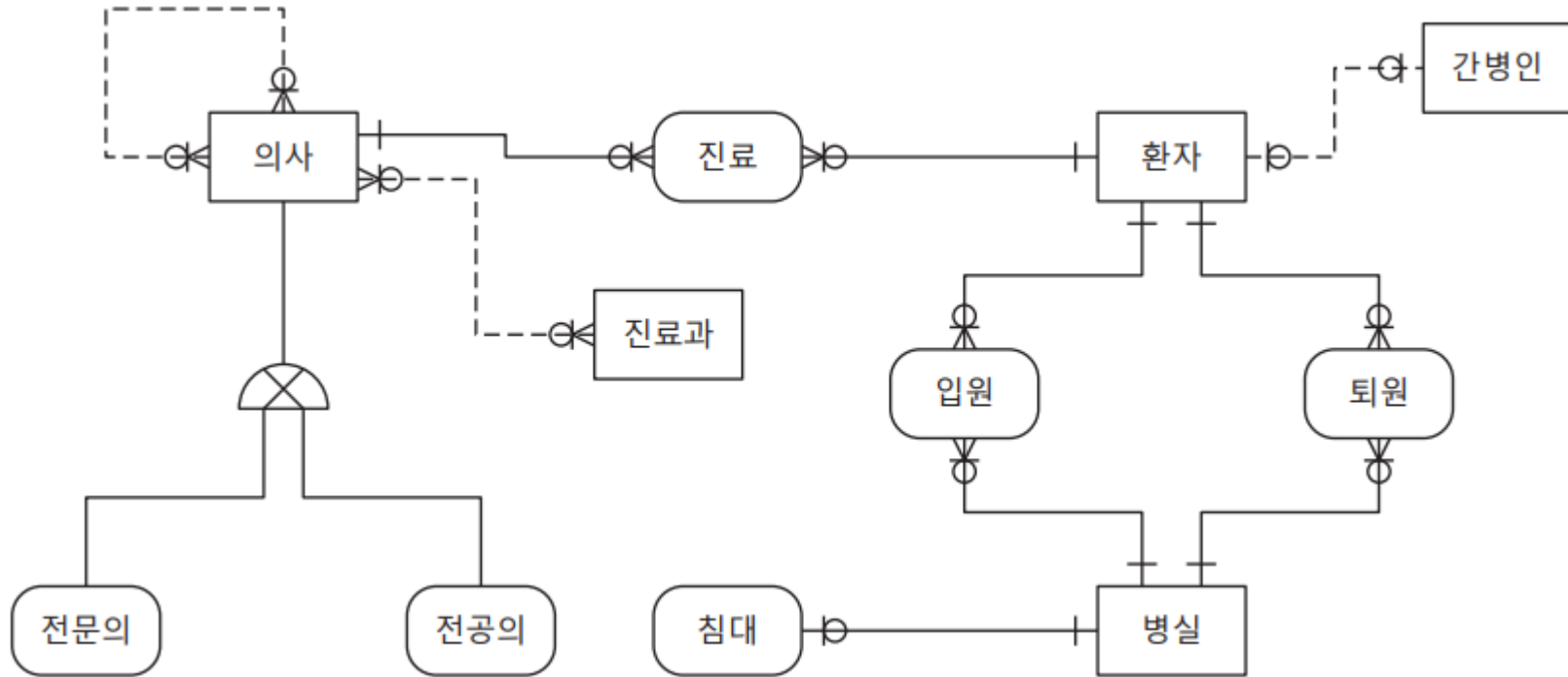
# ERwin을 이용한 IE 표기법(logical model) 변환

- 병원 DB의 IE 논리적 모델 변환 표현



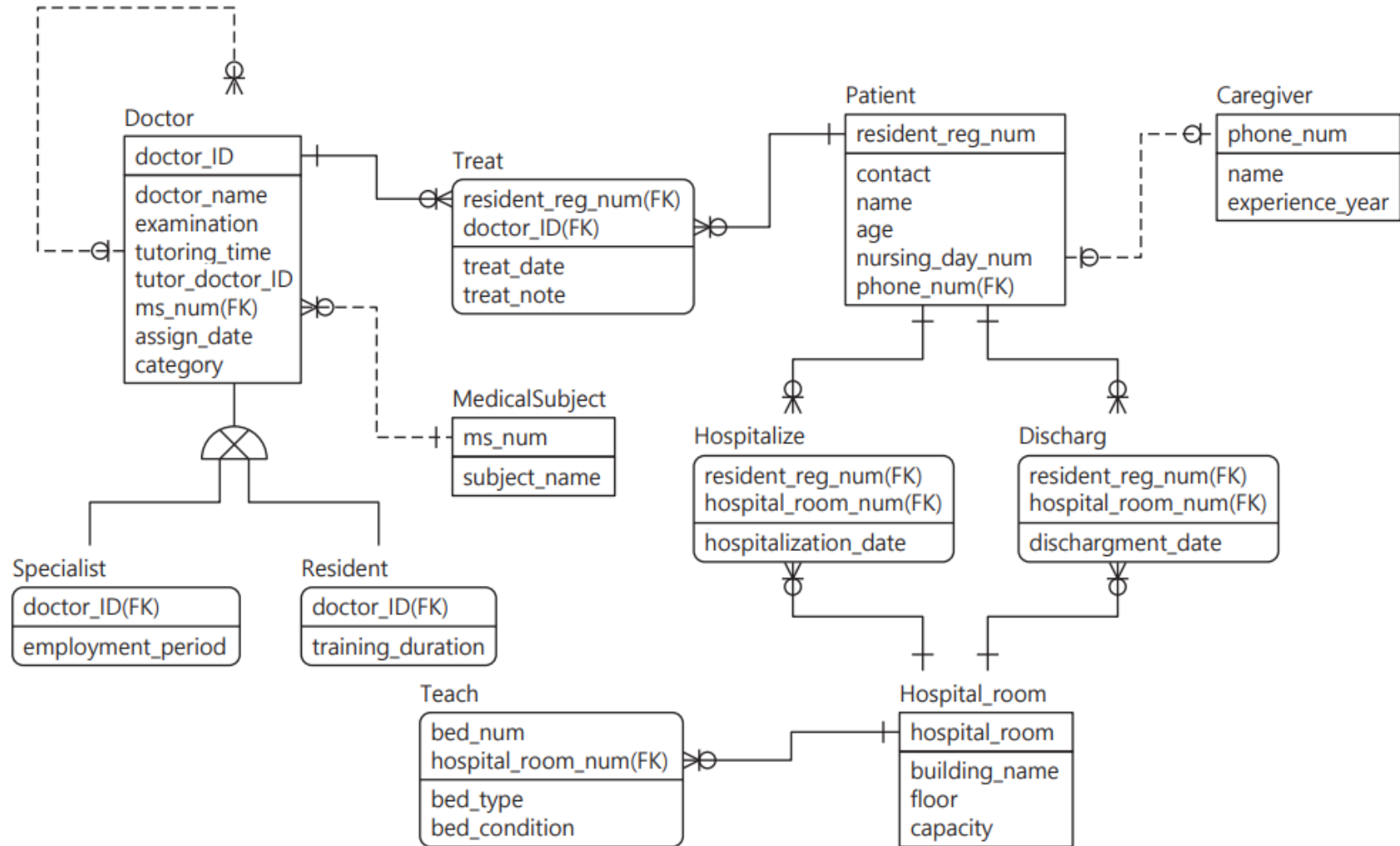
# ERwin을 이용한 IE 표기법(logical model) 변환

- ERwin IE 논리적 모델(엔터티 수준 요약) 표현



# ERwin을 이용한 IE 표기법(physical model) 변환

- 병원 DB의 IE 물리적 모델 변환 표현



# ERwin을 이용한 SQL 자동 생성

- ERwin의 자동 SQL(DDL 스크립트) 생성

