

# 데이터베이스 모델링

# 1. 데이터 모델링이란?

- ▶ 현실세계에서 데이터베이스를 만들고자 하는 대상 (현실의 업무적인 프로세스)이 되는 작은 세계 (Mini World)를 추출하여 개념적 세계를 뽑아내고, 이를 물리적으로 데이터베이스화 시키는 과정을 의미한다.

# 1. 데이터 모델링

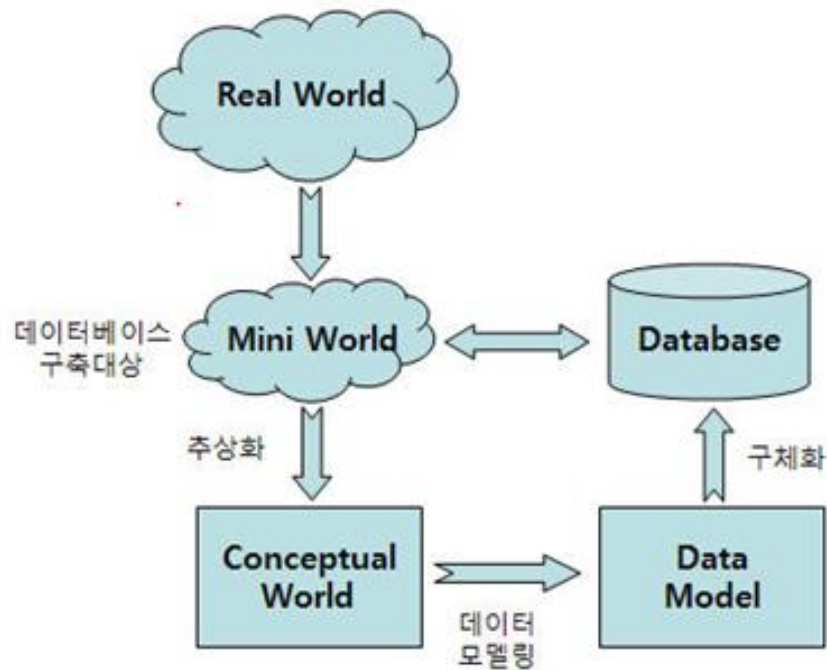
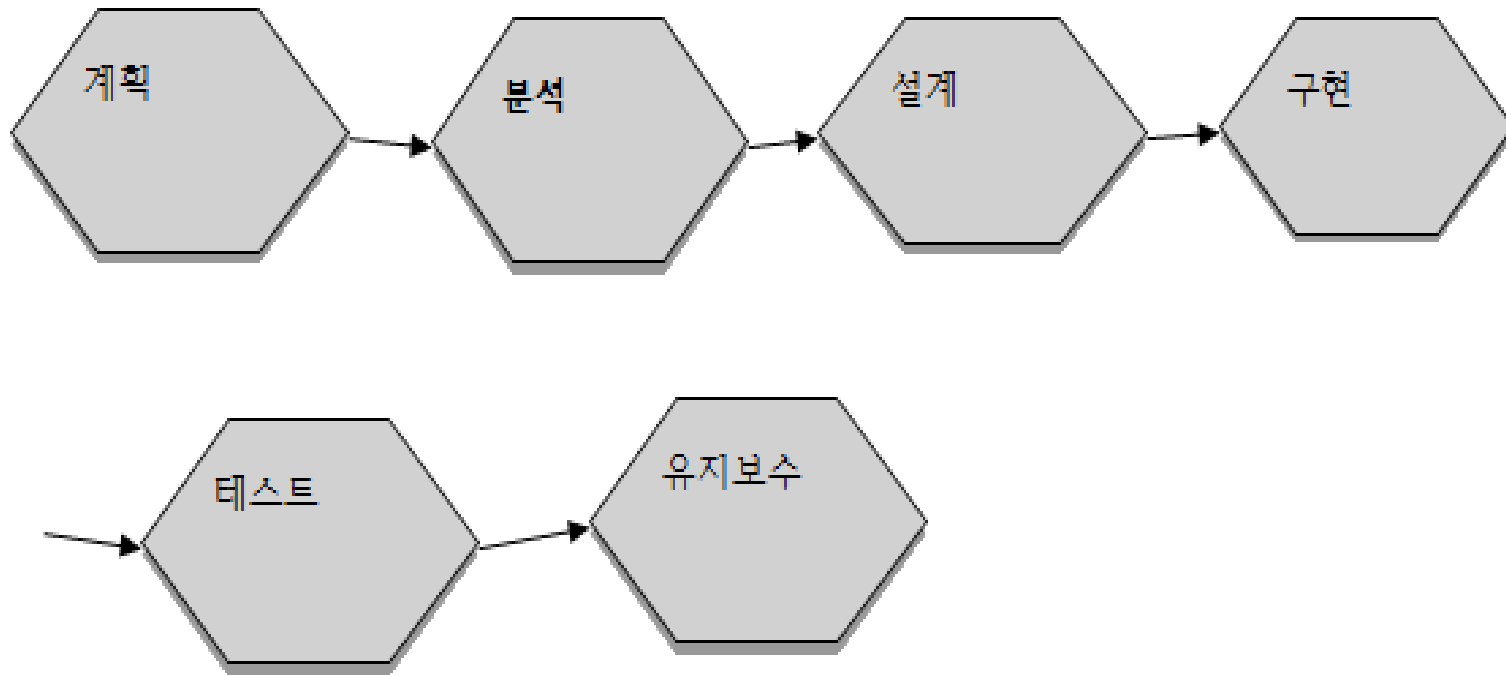


그림 2-1  
데이터 모델링

## 2. 프로젝트 진행과정

일반적인 프로젝트의 진행과정은 다음과 같다.



프로젝트를 수행함에 있어서 **업무분석과 데이터베이스 모델링**은 상당히 중요한 부분.

정확한 데이터베이스 모델링을 하려면 업무에 대한 이해와 체계적인 이론, 데이터베이스 시스템에 대한 깊은 이해가 있어야 한다.

### 3. 업무분석

- ▶ 우선 업무를 제대로 분석하기 위해서는 관련 분야에 대한 기본 지식과 상식을 가지고 있어야 한다.
- ▶ 업무를 분석할 때 주의할 사항은 관계형 데이터베이스 이론에 입각한 데이터베이스 스키마를 염두에 둘 필요가 없다는 점이다. 이는 논리적 데이터베이스 모델링 단계에서 정의할 사항이다.
- ▶ 업무 자체와 업무 프로세스 파악에 초점을 두고 분석해야만 한다.

## 4. 업무 분석 요령

- ▶ [1] 우선 **문서(서류,장표,보고서)**를 이용하여
- ▶ 데이터로 관리되어지는 항목들을 정확히 파악해야 한다.
- ▶ 문서에는 실제 업무에서 사용되어지고 데이터로 관리되어져야 하는 중요 항목들이 모두 정의되어 있기 때문이다.
- ▶ [2] 현업 **실무자와의 인터뷰**를 통해 업무를 분석하고 발생 가능한 경우 수에 대해 반드시 현업 실무자에게 내용확인을 해야 한다.
- ▶ **업무 현장 방문 및 설문지 작성, 기존 시스템 조사** 등의 작업도 병행한다.

## 4.업무 분석 요령

- ▶ [3] 또한 **사용자들의 요구분석**도 이루어져야 한다. 사용자의 요구분석은 현재 업무에 대한 분석이 마무리 된 후 진행하는 것이 좋다.
- ▶ 현업의 업무도 모르면서 사용자 요구까지 정리를 한꺼번에 할 수 없으므로 단계적으로 진행하는 것이 좋다.
- ▶ 데이터베이스 모델링에서 가장 중요한 것은 **업무**임을 명심하자.

## 4\_2. 요구사항 수집 및 분석

- **조사 방법**

- 기초 조사
- 자료 조사
- 사용자 면담
- 설문지 조사
- 현장 조사

- **조사 결과의 문서화**

- 무질서하게 추출된 정보를 데이터베이스 요구사항으로 분석하기 위해 필요한 작업

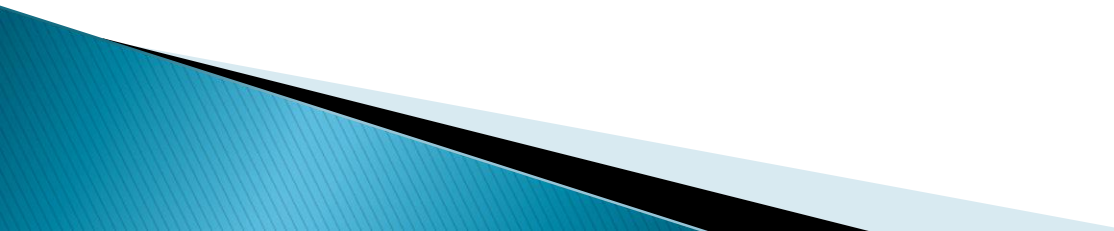


## 4\_2. 요구사항 수집 및 분석

- **요구 사항 분석**

- 데이터베이스를 구축하고자 하는 대상 기관에 대해 조사한 결과를 분석하여 데이터베이스 설계를 위한 제약 조건들로 재정리 하는 것

- **요구 사항 분석에 포함될 내용**

- 시스템의 목적 설명
  - 제약 조건 설정
  - 기존 시스템에 대한 이해
  - 요구 사항 명세서
- 

# 4\_3 자료흐름도(DFD) 작성하기

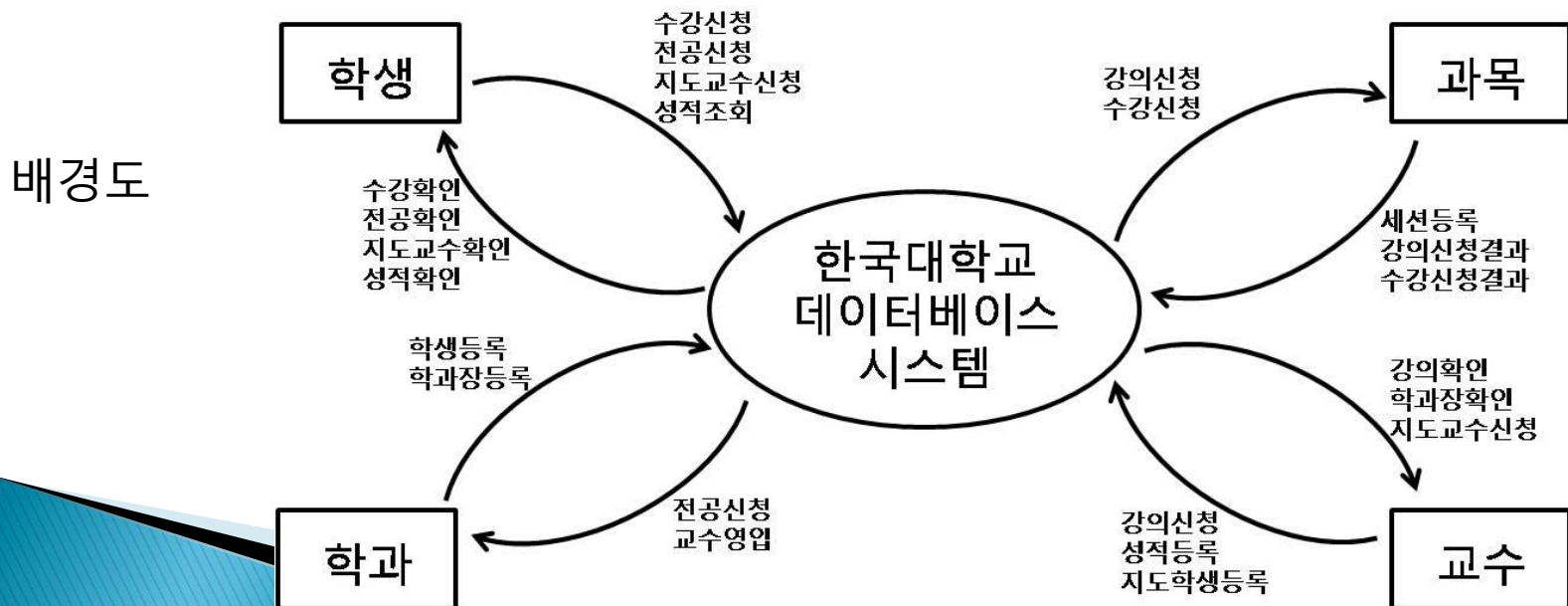
- 자료 흐름도의 작성법

- 배경도 : 시스템의 최상위 프로세스를 표기한 것
  - 시스템의 외부에서 바라본 시스템의 모습
- 배경도를 작성한 후에 하위 레벨의 자료 흐름도를 작성하면서 최종 자료 흐름도를 완성

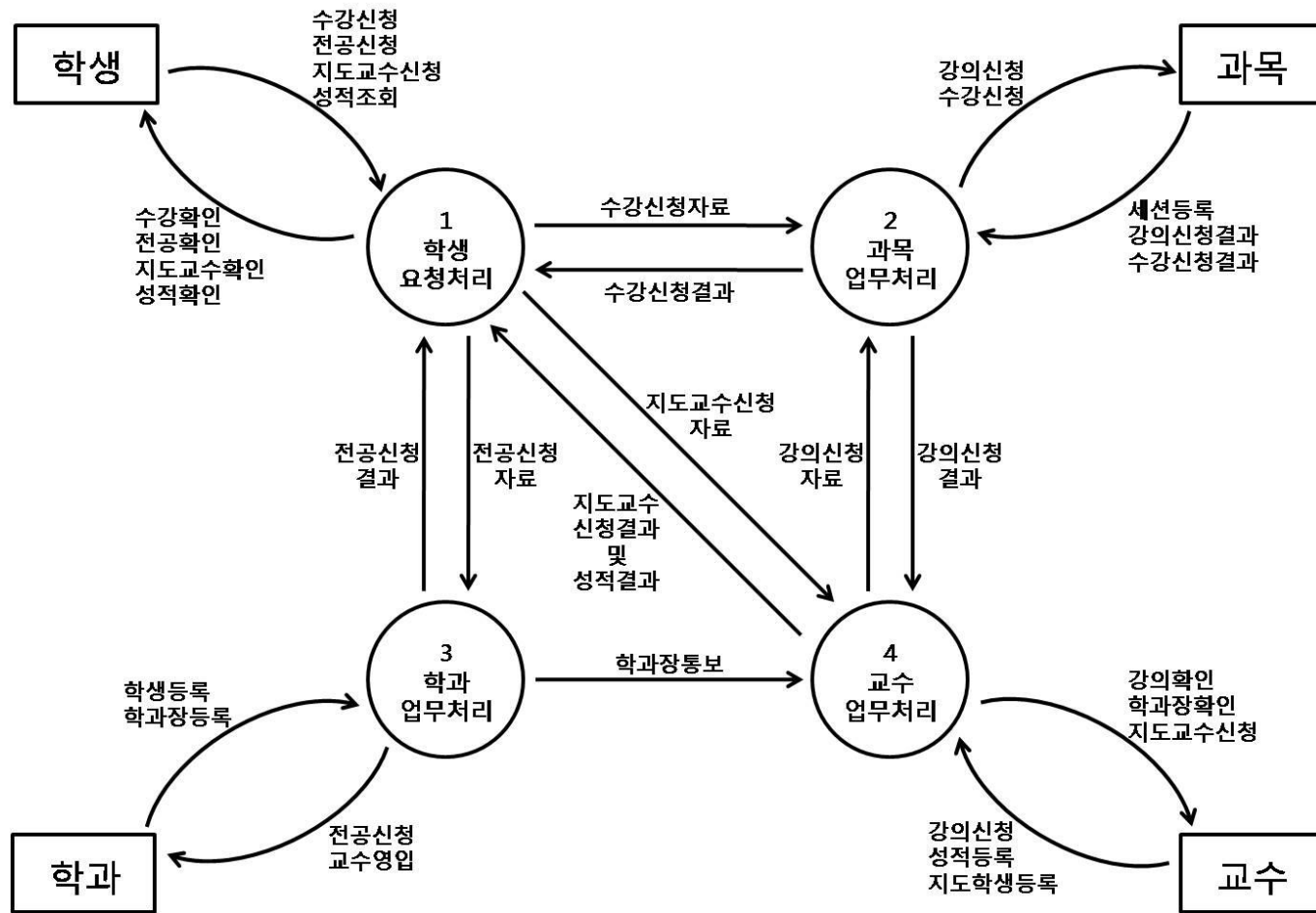
## 4\_3 자료흐름도(DFD) 작성하기

### • 자료 흐름도 작성시 유의점

- 순서도와 구분한다.
- 입출력 흐름을 명확하게 한다.
- 쉽고, 의미 있는 이름을 사용한다.



## 4\_3 자료흐름도(DFD) 작성 예제

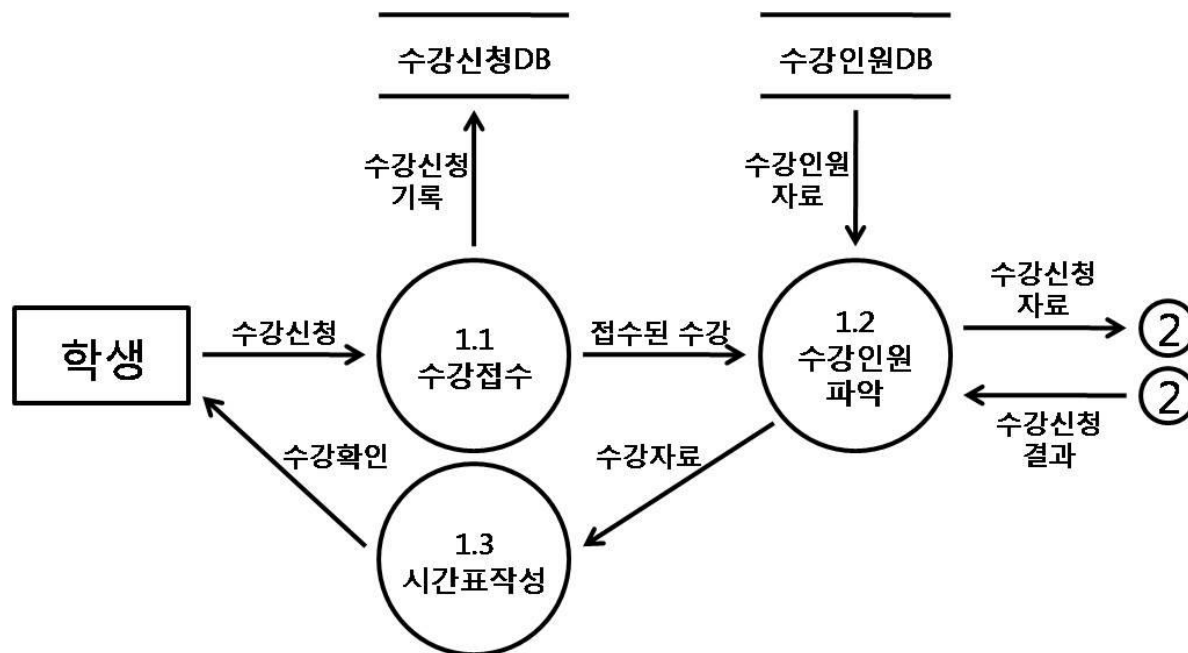


레벨0의 DFD

# 4\_3 자료흐름도(DFD) 작성 예제

- 자료 흐름도 작성예제

- 레벨 1~3 DFD



# 5. 데이터 모델의 종류

- ▶ 데이터 모델은 개발 공정 단계에 따라
  - ▶ [1]개념적 데이터 모델,
  - ▶ [2]논리적 데이터 모델,
  - ▶ [3]물리적 데이터 모델로
- ▶ 분류할 수 있다.

## [5\_1] 개념적 데이터 모델

- ▶ 업무분석 단계에서 얻어진 내용을 토대로 우선 엔티티(Entity)를 추출하고 엔티티내의 속성(Attribute)을 구성하여 엔티티간의 관계를 정의해서 ER-Diagram을 정의하는 단계
- ▶ 전체 시스템에 대한 개념적인 정보를 나타내는 데 사용
- ▶ 개체-관계 모델(ER Model, Entity-Relationship Model)이 대표적인 개념적 데이터 모델이다.

## [5\_2] 논리적 데이터 모델

- ▶ 개념적 데이터베이스 모델링 단계에서 정의된
- ▶ ER Diagram을 매핑 룰을 적용해 관계형 데이터베이스 이론에 입각한 스키마를 설계하는 단계와
- ▶ 완벽한 정규화 과정을 수행하는 정규화 단계로 구분할 수 있다.



## [5\_3] 물리적 데이터 모델

- ▶ 컴퓨터 내부에서 데이터들이 실제로 어떻게 저장되는가를 표현
- ▶ 이 단계에서는 우선 개발하고자 하는 DBMS종류를 결정하고 논리적 모델링 단계에서 얻어진 정규화된 모델에 컬럼의 데이터타입과 사이즈 등을 정의해야 한다.
- ▶ 또한 각종 제약조건 등을 정의하고 인덱스 정의, 정규화를 위배하는 역정규화 과정 등을 포함한다
- ▶ 물리적 모델링을 마치면 실제 데이터베이스 스키마를 생성하게 되며 이후부터는 본격적인 프로그램 개발 작업에 들어간다.

# 복습문제

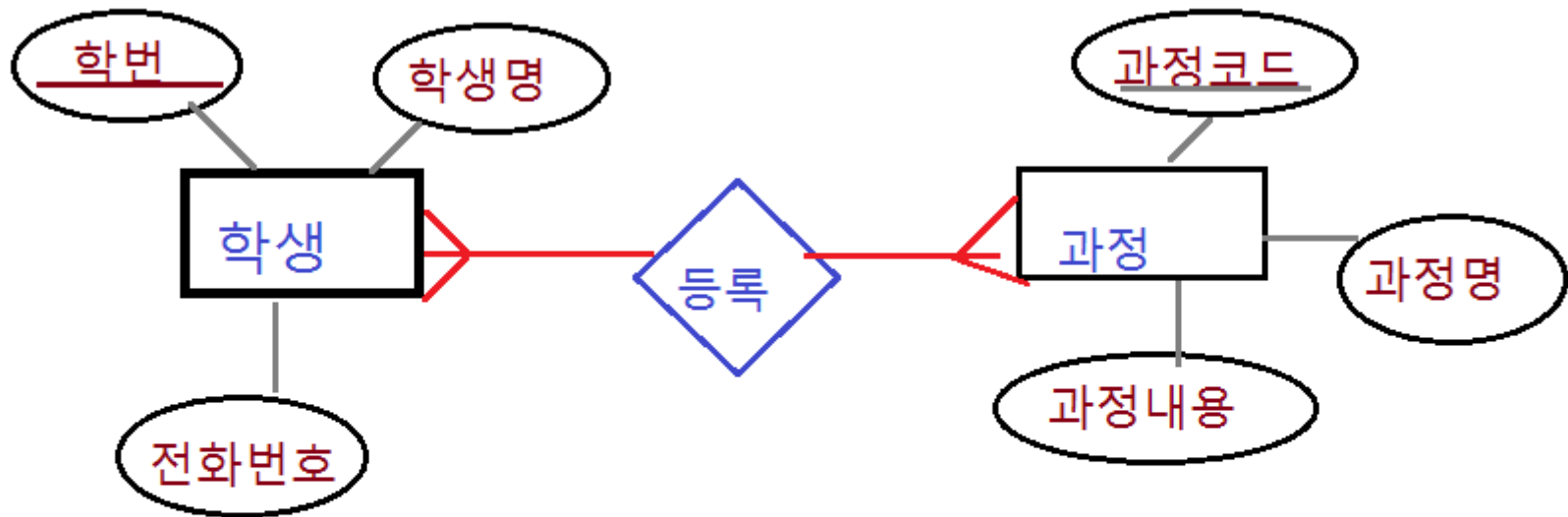
- ▶ 데이터베이스 모델링에 관해 정의하시오
- ▶ 업무를 분석하는 요령에 관해 기술하시오
- ▶ 데이터베이스 모델링의 각 단계별 특징에 관해 정의하시오.

# 6. E-R MODEL

## (Entity Relation Model)

- ▶ 개념적 데이터 모델
- ▶ 엔티티(Entity), 애트리뷰트(Attribute), 관계(Relationship)를 이용해서 실세계를 개념적으로 표현하는 기법
- **ER 모델의 구성 요소**
  - 엔티티(Entity)
  - 애트리뷰트(Attribute)
  - 관계(Relationship)




## 6. E-R MODEL



학생과 과정 관계를 표현한 ER-Diagram

## 6. E-R MODEL

- ▶ ER 모델은 1976년 P.Chen이 제안한 것으로 개체 타입, 관계 타입을 기본 개념으로 현실 시계를 개념적으로 표현하는 방법이다.

	개체를 표현할 때는 직사각형으로
	개체의 속성을 나타낼 때는 타원형으로
	개체들간의 관계는 마름모로 나타내고 이들을 연결하는 링크로 구성한다.

## 6. E-R MODEL – 엔티티

- 엔티티 (Entity)
  - 모델의 관리 대상
  - 사람과 물건, 장소 같은 실체가 있는 것이나 개념을 엔티티로 선택
  - 시스템 구축 단계까지 진행된다면 파일이나 데이터베이스의 테이블로 구현
  - ER 다이어그램에서는 사각형으로 표현

# 6. E-R MODEL – 엔티티

- 엔티티

- 실제로 존재하는 대상들
- ER 모델에서 가장 기본이 되며, 고유하게 식별이 되어야만 한다.

엔티티의 예

분야별↕	가능한 엔티티↕
사람↕	학생, 교수, 사원↕
장소↕	학교, 강의실↕
사물↕	교과서, 생산 제품↕
사건↕	강의, 면담↕
개념↕	강좌, 과목, 프로젝트↕

## 6. E-R MODEL -어트리뷰트

- 어트리뷰트 (Attribute)
  - 엔티티의 구성 요소
  - 엔티티 또는 관계가 갖는 성질이나 특성
  - 엔티티는 반드시 하나 이상의 **키 어트리뷰트**를 갖고 있어서 나머지 어트리뷰트를 유일하게 정의할 수 있다.
  - ER 다이어그램에서 어트리뷰트는 타원으로 표현



## 6. E-R MODEL -관계

- **관계 (Relationship)**

- 엔티티 간의 관계를 나타내는 것으로 1:1, 1:N, M:N 관계를 표현
- 관계는 관계형 데이터베이스로 매핑(Mapping, 사상) 되는데, ER 다이어그램에서는 마름모로 표현

## 7. 엔티티 정의

- ▶ 업무 수행을 위해 데이터로 관리되어야 하는 사람,사물,장소,사건 등을 엔티티라고 한다.
- ▶ 실체를 파악하는데 있어서 가장 중요한 점은 관련 업무에 대한 지식이 있어야 한다.
- ▶ 업무적 내용을 말로 풀어보고 그 내용 중 명사 위주로 실체들을 추출하면 가장 알기 쉽다.
- ▶ 다음 페이지의 업무 분석 내용을 보고 엔티티를 추출해보자.

# 7. 엔티티 정의

- ▶ 업무분석
  - ▶ - 모 대학에서는 학생들이 어떤 과목들을 수강하는지 전산화하고 싶다.
  - ▶ - 학생들의 학번과 이름과 연락처 조회와 어떤 과목이 얼마의 학점인가도 업무에서 사용된다.
  - ▶ - 해당 날짜에 누가 어떤 과목을 수강했는지와 어떤 과목들이 많은 수강을 했는지도 전산화하여 조회하고 싶다.

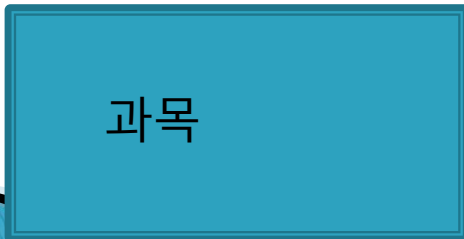
## 7. 엔티티 정의

- ▶ 그렇다면 위 내용 중 어떤 명사를 엔티티로 파악할 수 있을까?
- ▶ 관련된 업무의 단어들을 나열해보자.
- ▶ 대학, 학생, 과목, 수강, 학번, 이름, 연락처, 학점, 날짜=> 이 안에는 엔티티와 어트리뷰트가 다 포함되어 있음
- ▶ 엔티티를 정해야 한다.
- ▶ \* 학생 (학번, 이름, 연락처, 수강날짜)
- ▶ \* 과목 (과목코드, 과목명, 학점, 수강날짜)

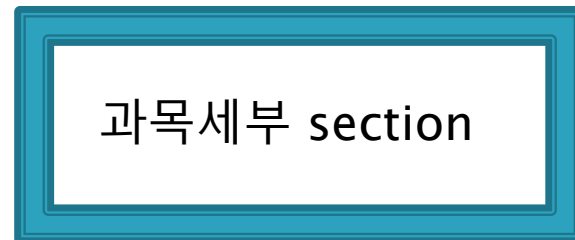
# 8. 엔티티 타입

- 엔티티 타입(Entity Type)

- 여러 엔티티가 모여서 하나의 집단을 이룬 형태
- ER 다이어그램에서 엔티티 타입은 사각형으로 표현
- 강한 엔티티 타입-보통의 일반 엔티티는 강한 타입
- 약한 엔티티 타입 : 자신의 키 애트리뷰트가 없는 엔티티 타입. 다른 엔티티에 종속되어 해당 엔티티가 없다면 존재하지 않는 종속성을 갖는다.

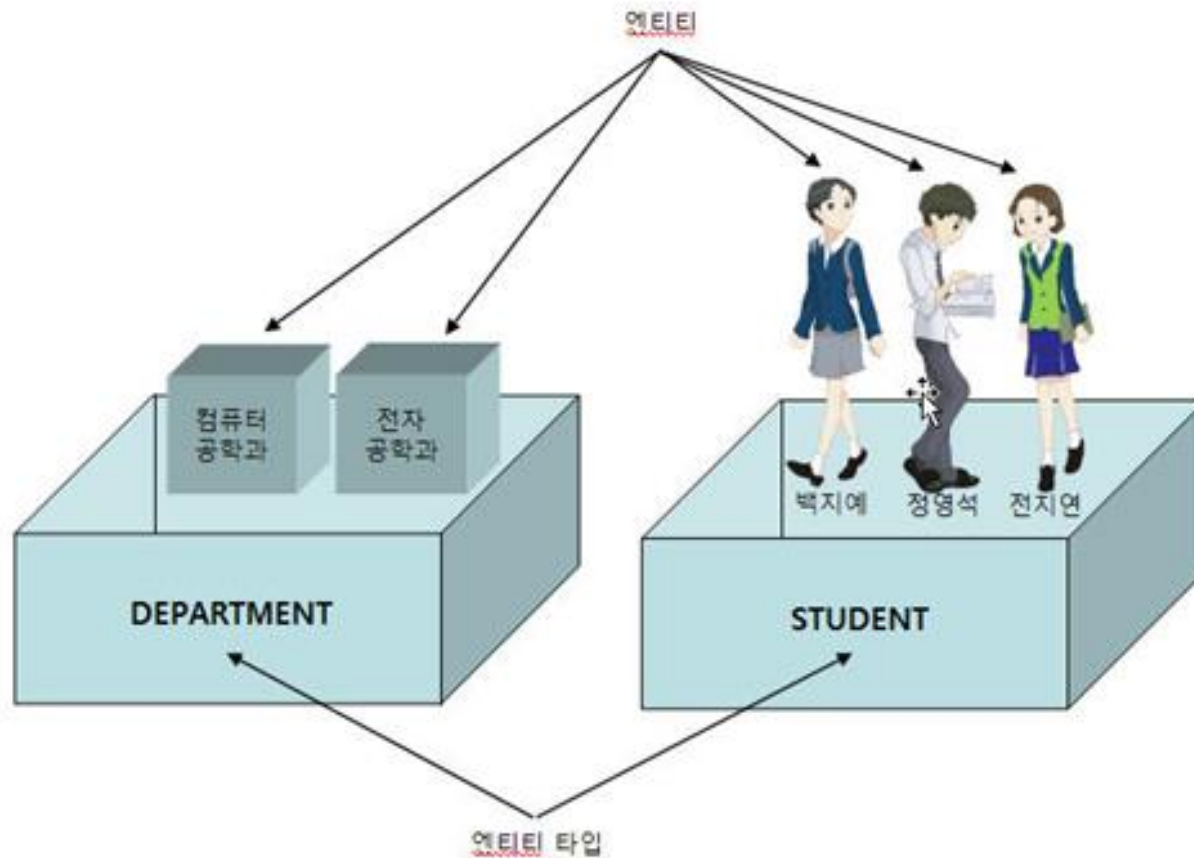


강한엔티티



약한엔티티(이중선으로 표시)

# 엔티티와 엔티티타입



## 9. 애트리뷰트 (Attribute)

- 단순 애트리뷰트(Simple Attribute)
- 아래 그림에 밑줄 그어진 학번이 키 애트리뷰트가 된다.



## 9. 애트리뷰트 (Attribute)

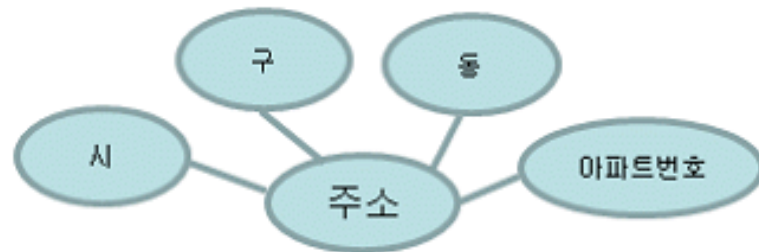
- **키 애트리뷰트(Key Attribute)**

- 엔티티들을 식별할 수 있는 유일한 제약조건을 갖는 애트리뷰트

- **복합 애트리뷰트(Composite Attribute)**

- 두 개 이상의 애트리뷰트로 이루어진다.
- 각각의 애트리뷰트는 그 자체로도 독립적인 의미가 있다.

- ▶ **복한 애트리뷰트 예**





# 9. 애트리뷰트 (Attribute)

- 다치 애트리뷰트(Multivalued Attribute)

- 애트리뷰트 하나에 여러 값이 들어갈 수 있는 애트리뷰트

- ▶ 학과 사무실 전화가 여러 개 있다면 전화번호

- ▶ 애트리뷰트에 여러 값이 들어가고,

- ▶ 교수 엔티티의 보유기술은 다양한

- ▶ 기술이 들어갈 수 있다.

- ▶ 다치 애트리뷰트는 두선으로 타원을 그려 표현



보유기술

## 9. 애트리뷰트 (Attribute)

- 유도된 애트리뷰트(Derived Attribute)
  - 애트리뷰트에 실제 값이 저장되어 있는 것이 아니라 저장된 값으로부터 계산해서 얻은 값을 사용하는 애트리뷰트
  - 나이는 실제 나이를 저장하지 않고 생년월일과 오늘날짜를 계산해서 얻는다.
  - 유도 속성은 점선으로 그려 표시



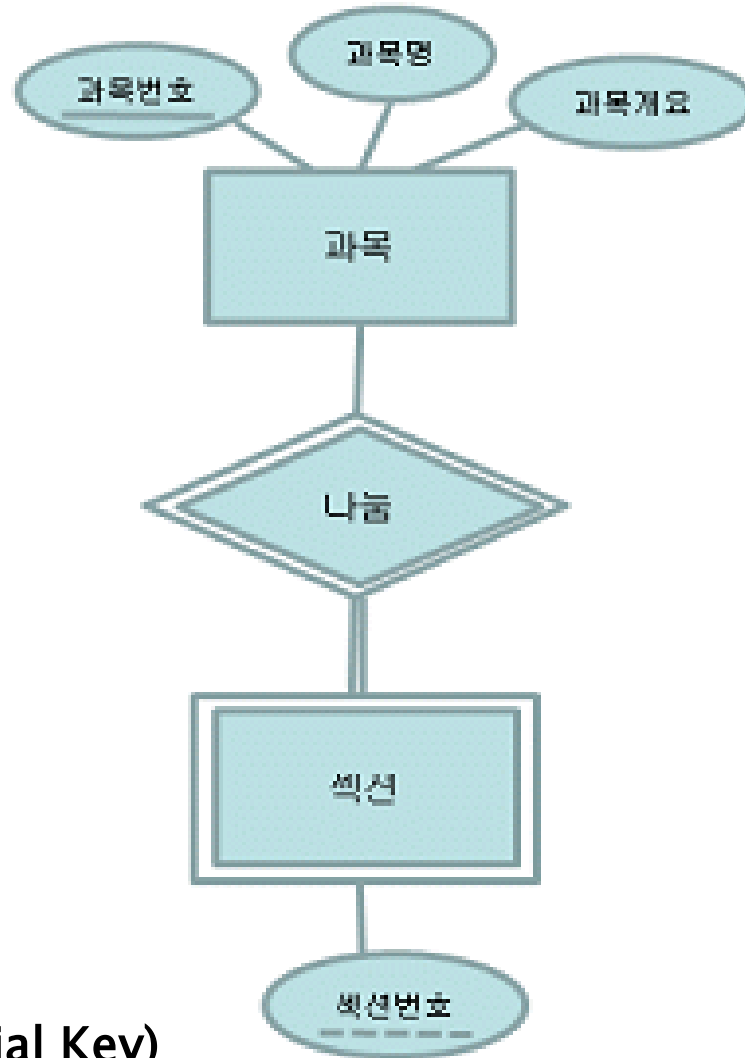
## 9. 애트리뷰트 (Attribute)

- **부분키(Partial Key)**

- 키와 비슷하지만 완벽하게 키라고는 할 수 없고 약한 엔티티에서만 사용되는데, 키 애트리뷰트에 반해서 부분키(Partial Key)라고 한다.
- 부분키는 ER 다이어그램에서 점선으로 밑줄을 그어서 표현한다.

# 9. 애트리뷰트 (Attribute)

자바 과목에는 약한 엔티티인 섹션이 있고 각각 section1, section2, section3이라면 자바 과목 내에서 섹션번호에 의해 유일하게 식별된다. 하지만 다른 과목에서도 동일한 섹션 번호를 사용할 수 있으므로 완벽하게 키라고 할 수는 없고 부분키라고 한다. 약한 엔티티에서만 사용된다. 점선으로 밑줄을 그어 표시한다.

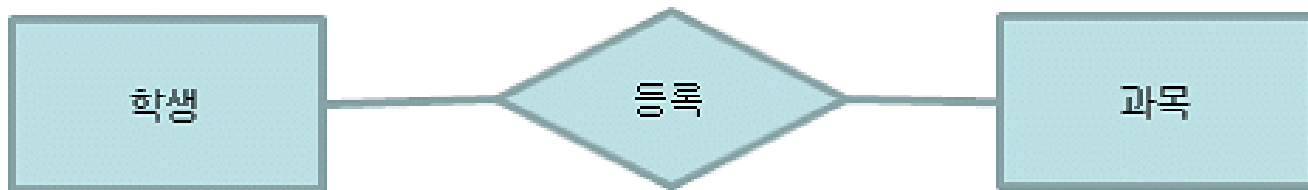


- 부분키(Partial Key)

# 10. 관계 타입

- 관계 타입

- 엔티티 타입 간의 관계를 표현할 때 사용
- 엔티티 간에 존재하는 수학적 관계를 말한다.
- ER 다이어그램에서 마름모를 사용하여 표현



# 10. 관계 타입

- ▶ 관계란 업무적인 연관성(Association)을 의미.
- ▶ 회원은 비디오를 대여한다.
- ▶ 비디오는 회원에게 대여되어진다.
- ▶ 교수는 학생을 가르친다.
- ▶ 학생은 교수로부터 배운다.
- ▶ 이렇듯 두 실체간에 관련 방식(동사)을 중심으로 관계를 정의한다.

# 10. 관계 타입

- ▶ 관계를 설정하는 순서
  - ▶ [1단계] 관계가 있는 두 실체를 실선으로 연결하고 관계를 부여한다.
  - ▶ [2단계] 관계 차수를 표혀한다.
  - ▶ [3단계] 선택성을 표시한다.
- 
- ▶ 차수성이란 하나의 인스턴스가 다른 실체의 몇 개의 인스턴스와 관련될 수 있는가를 정의하는 것.
  - ▶ 일대 일, 일대 다, 다대 다 등 3가지 경우가 있다.

# 10. 관계 타입

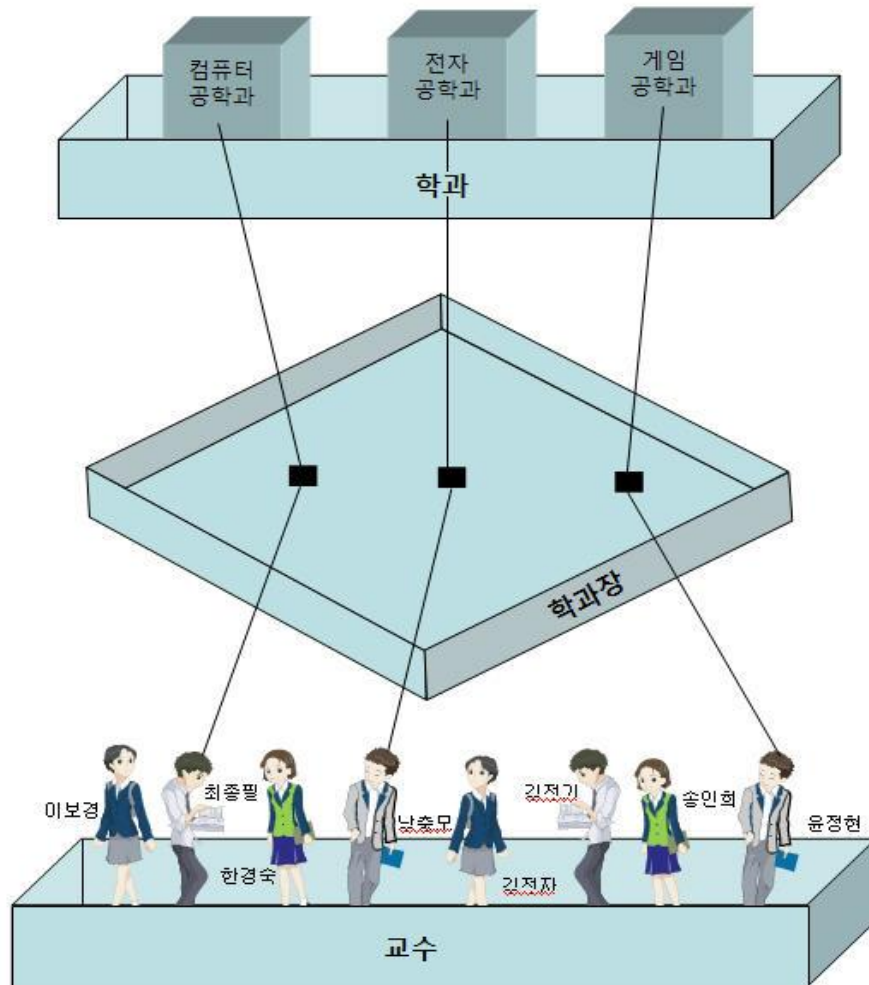
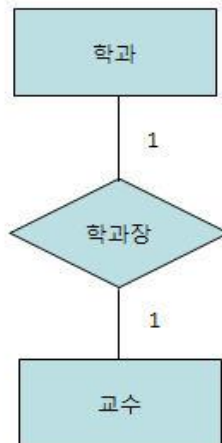
- ▶ 카디널리티 비율 - 일대다(1:1) 관계
- ▶ 관계 맺고 있는 두 실체의 레코드가 서로 하나씩 대응되는 관계
- ▶ 교수와 학과 사이에 학과장이라는 관계에서 교수 한 명은 한 학과의 학과장이 될 수 있다. 이처럼 하나의 엔티티에 대해 하나의 엔티티만이 관계를 맺는 경우를 1:1 관계라고 한다.



# 10. 관계 타입

카디널리티 비율 - 일대다(1:1) 관계

1대1의 관계를 표현한다.

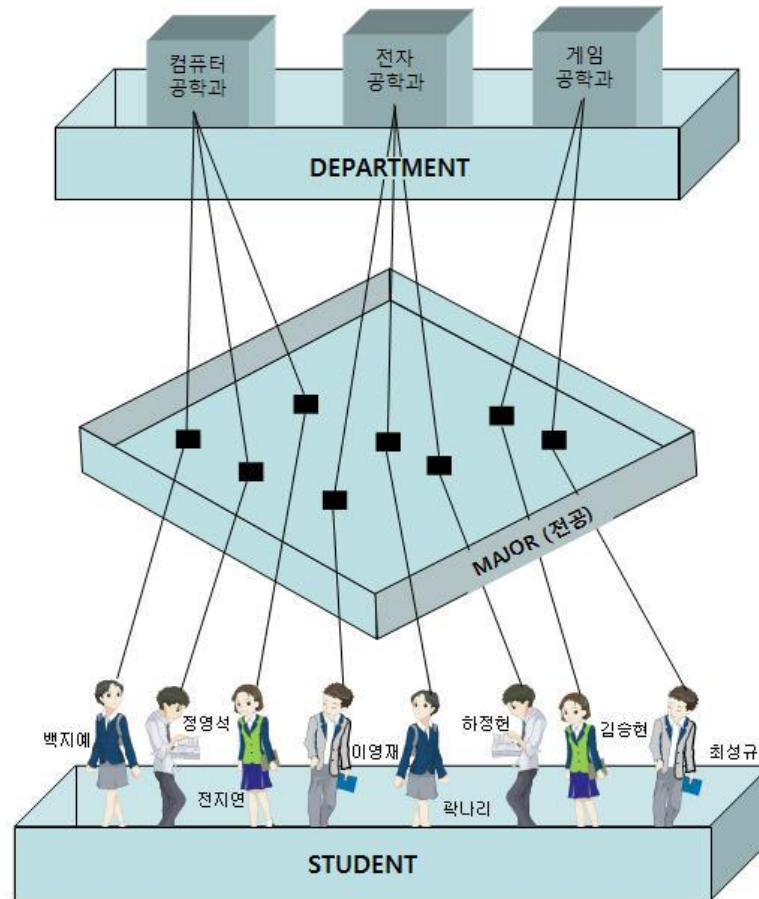
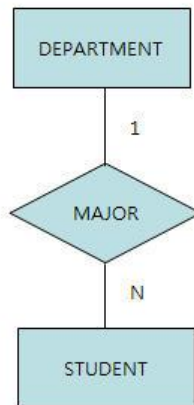


# 10. 관계 타입

## ▶ 카디널리티 비율 - 일대다(1:N) 관계

학과와 학생의 일대다 관계

1대다의 관계를 표현한다.

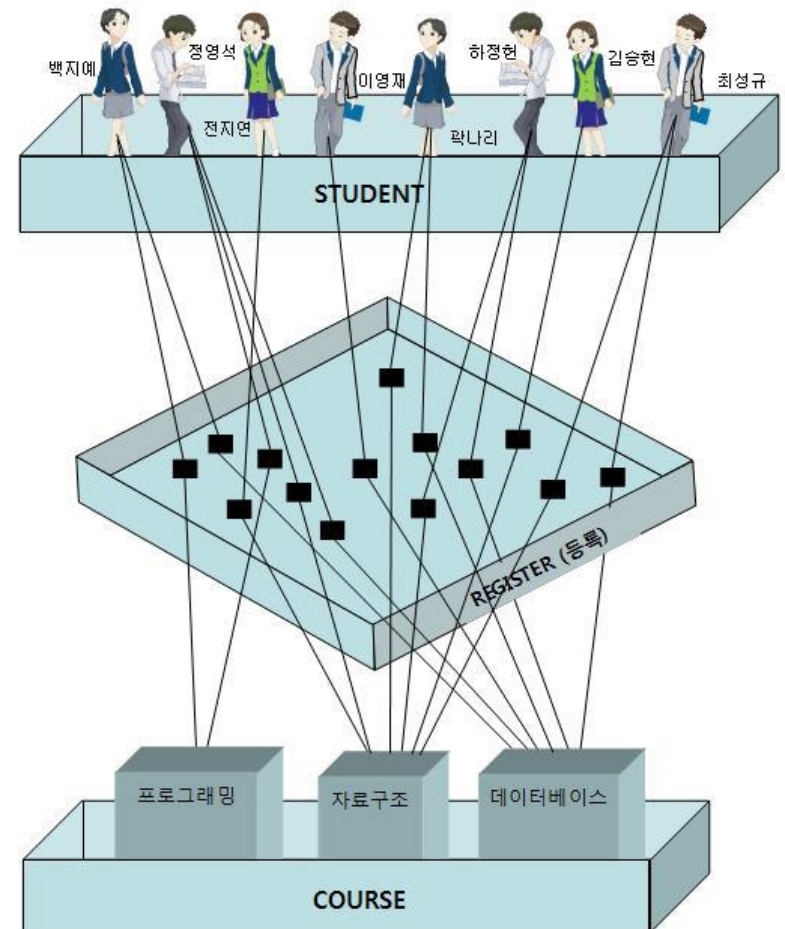
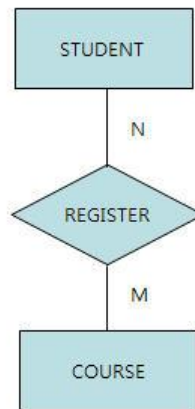


# 10. 관계 타입

## ▶ 카디널리티 비율 - 다대다(M:N) 관계

학생과 과목  
의 다대다 관  
계

다대다의 관계를 표현한다.



# 10. 관계 타입

- ▶ **카디널리티 비율 - 다대다(M:N) 관계**
- ▶ 학생은 여러 과목을 수강 신청할 수 있으며, 과목에 대해서도 여러 학생이 수강 신청할 수 있다. 이런 경우를 다대 다 관계라고 함.
- ▶ 뒤에서 자세히 설명하겠지만  $n:m$  관계는 이론상으로 존재하지만 실제 데이터베이스에서는  $1:1$ 과  $1:n$ 만 존재하게 된다.
- ▶ (다:다 관계일 경우 반드시 교차 테이블이 있어야 함)

# 11. 개념적 설계

- ▶ 명사 정제하기
- ▶ : 데이터베이스 요구사항으로부터 얻어낸 명사들 중 엔티티라고 생각되는 명사들을 추출, 구분
- ▶ 학생, 학번, 이름, 주소, 생년월일, 전공학과, 수강과목, 지도교수
- ▶ 과목, 과목번호, 과목명, 과목개요, 섹션
- ▶ 교수, 이름, 전공분야, 보유기술, 지도학생
- ▶ 학과, 학과명, 사무실위치, 전화번호, 학과장
- ▶ 굵은 글씨로 표기한 것이 엔티티일 가능성이 많다.

# 11. 개념적 설계

## ▶ 명사 그룹 짓기

: 대표하는 명사로 그룹의 이름을 붙이고,  
다른 그룹에 속하는 단어들 제거

학생, 학번, 이름, 주소, 생년월일, 전공학과, 수강과목, 지도교수

과목, 과목번호, 과목명, 과목개요, 섹션

교수, 이름, 전공분야, 보유기술, 지도학생

학과, 학과명, 사무실 위치, 전화번호, 학과장

# 11. 개념적 설계

## ▶ 엔티티 추출하기

학과

교수

학생

과목

섹션

# 11. 개념적 설계

## ▶ 엔티티에 대해서 정의 내리기

- ▶ EX) 학생: 학교에 입학하여 한 개의 학과에 속하며 교수로부터 강의 받는 사람이다.
- ▶ 과목: 학생이 배워야 할 내용을 세분화한 것이다.
- ▶ 섹션: 과목을 다시 세분화한 것이다.
- ▶ 교수: 학교와 계약을 맺고 학생에게 강의하는 사람이다.
- ▶ 학과: 전공 교육의 편의를 위해 구분한 학술의 분과이다.

## ▶ 엔티티 표기하기

- ▶ : ER 다이어그램에 엔티티 먼저 표기



# 11. 개념적 설계

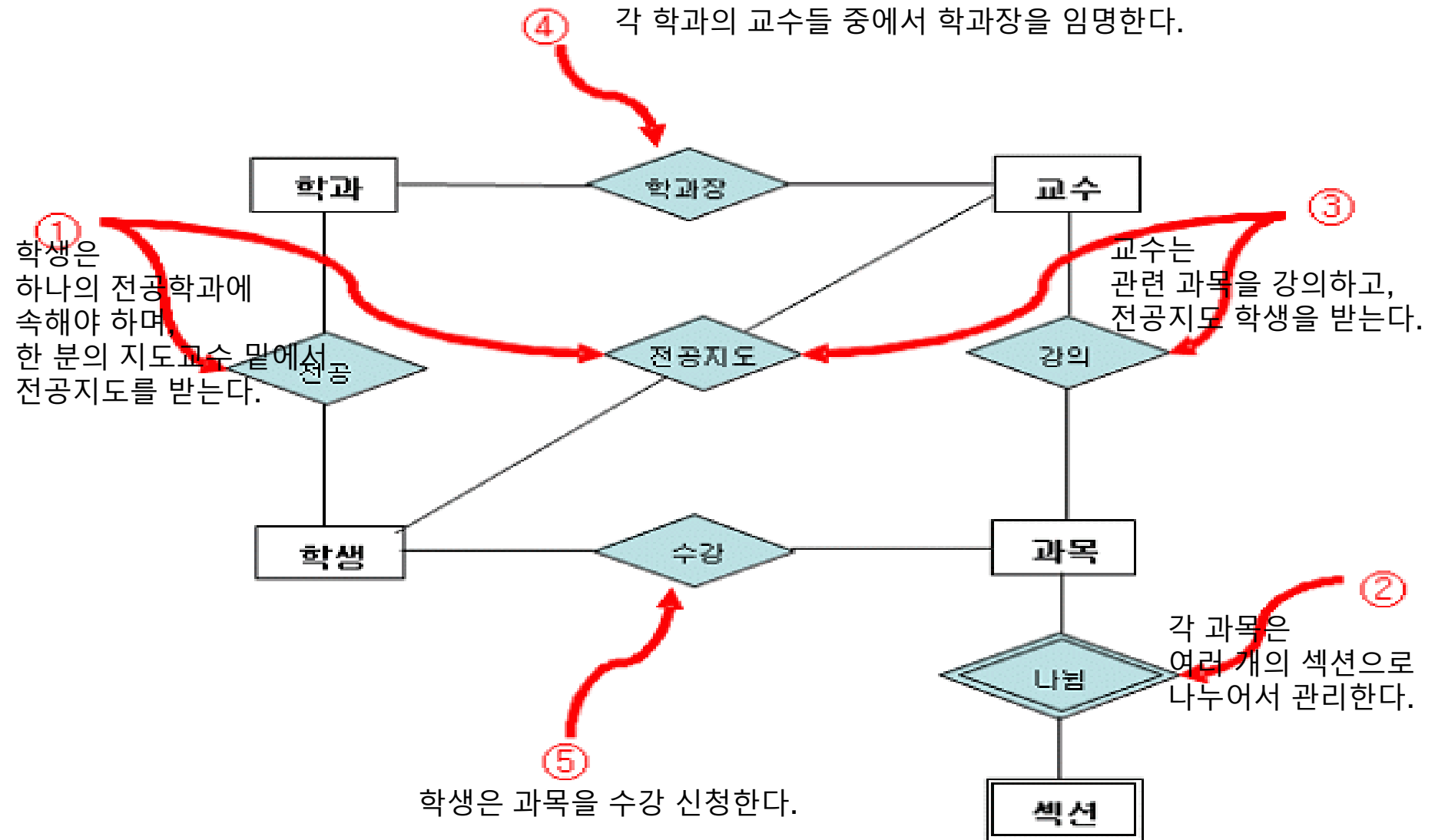
- 관계 설정하기

- 관계 정의

- : 동사를 포함하는 문장 속에서 관계 추출

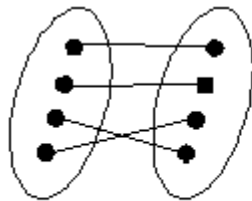
- : 엔티티 간의 관계를 설명하는 동사나 이벤트를 나타내는 동사로부터 찾음

# 11. 개념적 설계

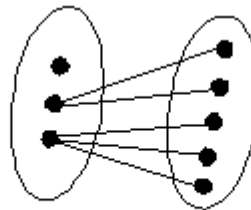


# 11. 개념적 설계

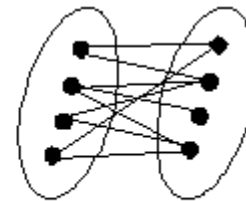
- 카디날리티
  - : 대응수 또는 원소수
  - : 관계의 유형을 정의
- 카디날리티의 종류
  - : 일대일(1:1)
  - : 일대다(1:N)
  - : 다대다(N:N)



1 : 1 관계



1 : 다 관계



다 : 다 관계

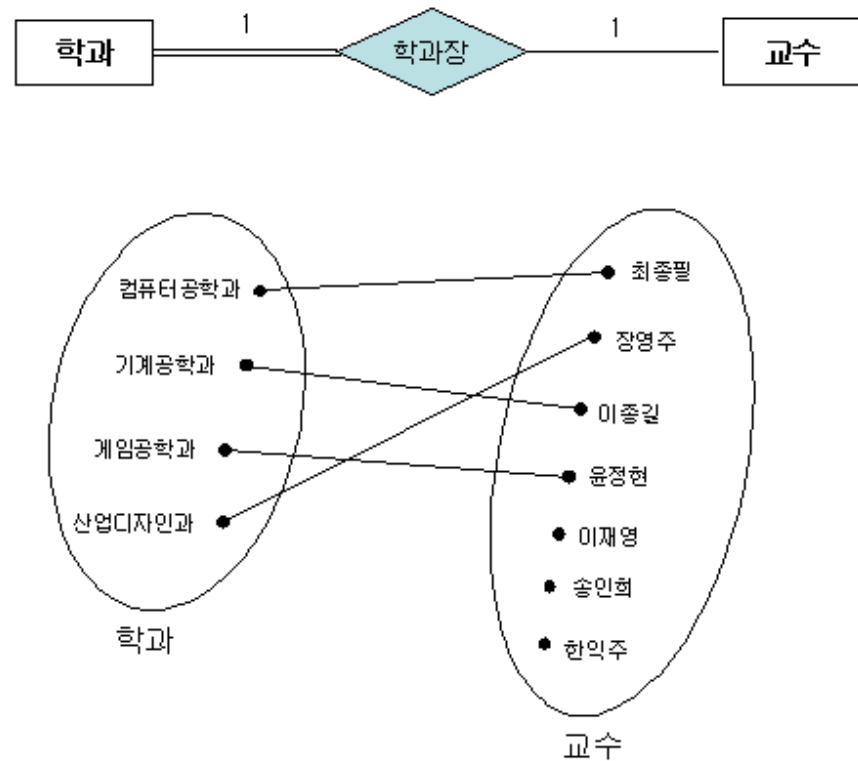
# 11. 개념적 설계

## ▶ 일대일(1:1) - 학과와 교수 사이

각 학과의 교수들 중 학과장을 임명한다. 이 때 여러 교수가 한 학과의 학과장이 될 수 없다. 또한 한 교수가 여러 학과의 학과장이 될 수도 없다.

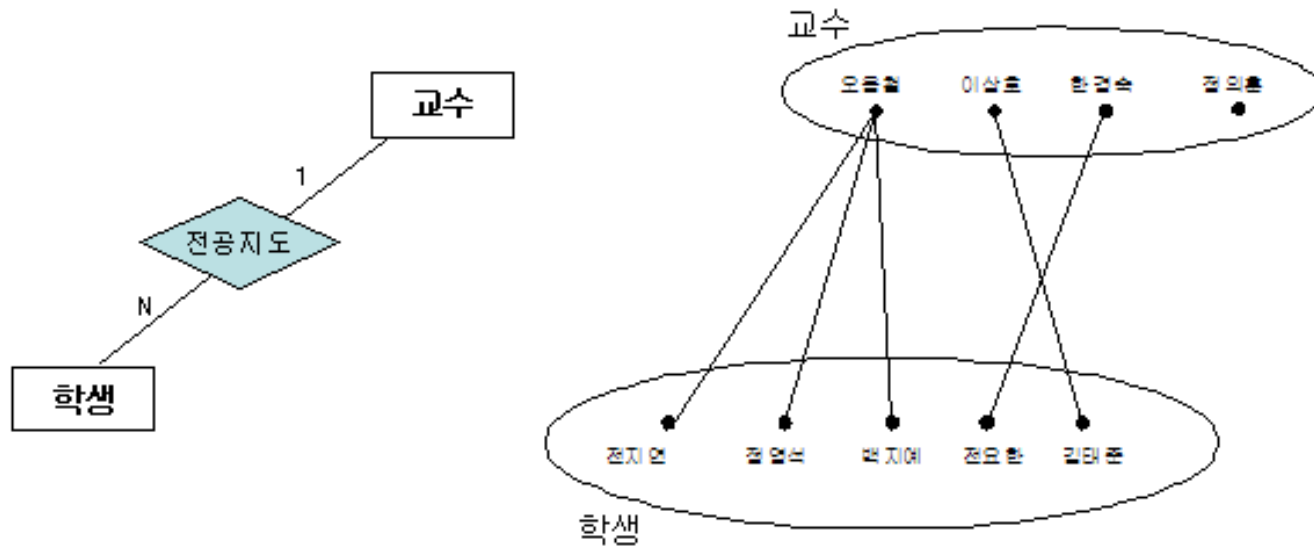
모든 학과에는 학과장이 있기 때문에 **학과와 학과장은 전체 참여 관계**라고 할 수 있으며 전체 참여는 두 줄로 표시한다.

반면 모든 교수가 학과장을 하는 것은 아니고 일부 교수만 학과장을 하므로 **학과장과 교수의 관계는 부분 참여관계**가 된다.



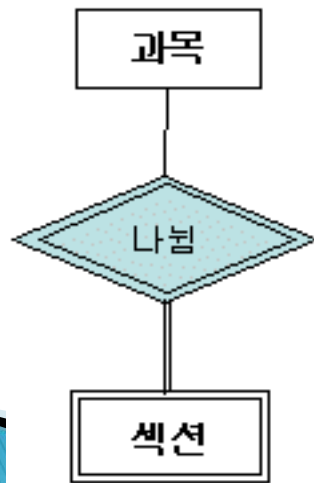
# 11. 개념적 설계

- ▶ 일대다(1:N) - 교수와 학생 사이



# 11. 개념적 설계

- ▶ 일대다(1:N) - 과목과 섹션 사이
- ▶ - 한 과목은 여러 섹션으로 나눌 수 있다.
- ▶ 이 때 섹션은 과목이 있어야 존재하고 모든 섹션은 과목에 속해 있으므로 섹션이 나눔이라는 관계에 있어 전체 참여가 된다.

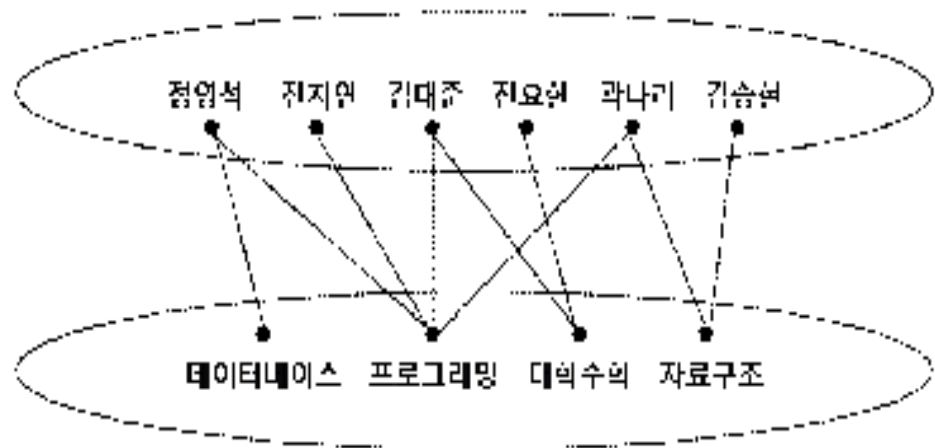


# 11. 개념적 설계

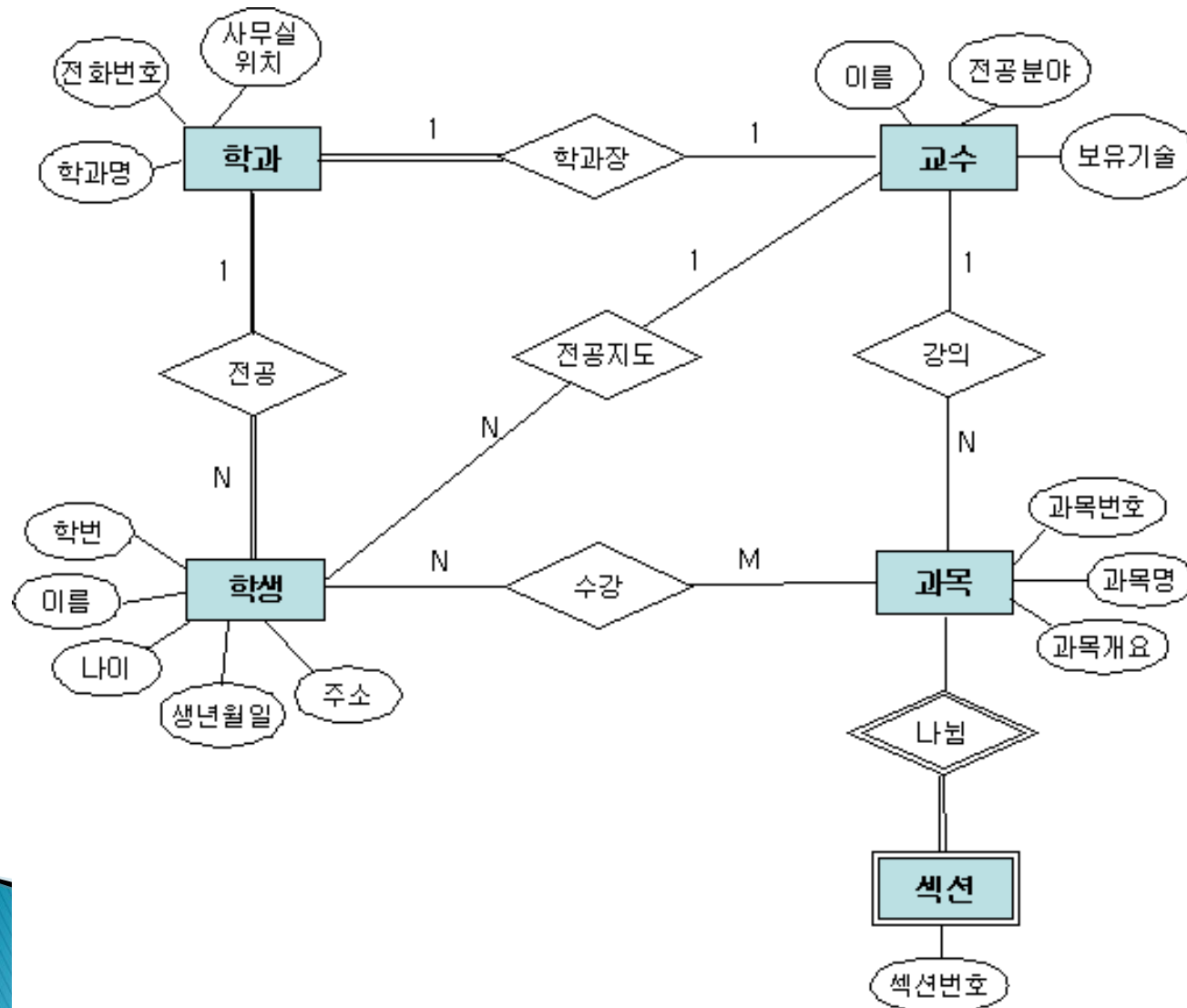
## ▶ 다대다(N:N) - 학생과 과목 사이



각 학생은 여러 과목을 들을 수 있고, 각 과목은 여러 학생이 수강할 수 있다.



# 11. 개념적 설계





# 12. 논리적 설계

- 논리적 설계

- 논리적 설계는 ER 다이어그램을 데이터베이스 관리 시스템에 매핑(Mapping, 사상)하는 것이다

# 12. 논리적 설계

- **관계**(테이블 또는 릴레이션)
  - 흔히 부르는 테이블이라는 이름으로 사용되고 있다.  
관계의 열은 애트리뷰트라고 하고, 행은 튜플이라 하며 실제 데이터의 값이 들어감
- **튜플**(레코드 또는 행)
  - 관계를 구성하는 각각의 행을 의미하며, 애트리뷰트의 모임으로 구성

# 12. 논리적 설계

- **애트리뷰트(속성 또는 열)**
  - 데이터베이스를 구성하는 가장 작은 논리적 단위이며, 개체의 특성을 기술
- **도메인**
  - 애트리뷰트가 취할 수 있는 같은 타입의 원자 값들의 집합을 의미
  - 실제 애트리뷰트 값이 나타날 때, 그 값의 합법 여부를 시스템이 검사하는 데에도 이용

# 12. 논리적 설계

## • 슈퍼키

- 관계에서 같은 튜플이 발생하지 않는 키를 구성할 때, 애트리뷰트의 집합으로 구성하는 것
- 모든 튜플에 대해 유일성은 만족하지만 최소성은 만족하지 못한다. 즉 슈퍼키의 최소 형태가 기본키다.
- 가령 학생{학번, 이름, 주소, 소속학과}에서 슈퍼키는 {학번, 이름}, {학번, 주소}, {학번, 소속학과}, {학번, 이름, 주소}, {학번, 이름, 소속학과}, {학번, 이름, 주소, 소속학과}, {학번}이다.
- 기본키는 슈퍼키의 최소 형태인 {학번}이 된다.

# 12. 논리적 설계

- 후보키

- 관계를 구성하는 애트리뷰트들 중에서 튜플을 유일하게 식별하려고 사용하는 애트리뷰트들의 부분집합, 즉 기본키로 사용할 수 있는 애트리뷰트들을 말한다
- 관계에 있는 모든 튜플에 대해 유일성과 최소성을 만족시켜야 한다.

# 12. 논리적 설계

- **기본키-Primary Key**

- 후보키 중에서 선택한 주 키
- 널(Null)을 값으로 가질 수 없다
- 동일한 값이 중복해서 저장될 수 없다

- **외래키-Foreign Key**

- 관계를 맺는 두 릴레이션에서 참조하는 릴레이션에 애트리뷰트로 지정되는 키 값을 말한다

# 12. 논리적 설계

- 참조 무결성 제약조건

- 참조 무결성 제약조건의 정의는 한 릴레이션에 있는 튜플이 다른 릴레이션에 있는 튜플을 참조하려면 반드시 참조되는 튜플이 해당 릴레이션 내에 있어야 한다는 것

- 키 제약조건

- 키 애트리뷰트의 값은 릴레이션 내의 각 튜플을 유일하게 식별해야 한다

# 12. 논리적 설계

- **도메인 제약조건**

- 각 애트리뷰트의 값은 반드시 도메인에 속하는 원자 값이어야 한다

- **엔티티 무결성 제약조건**

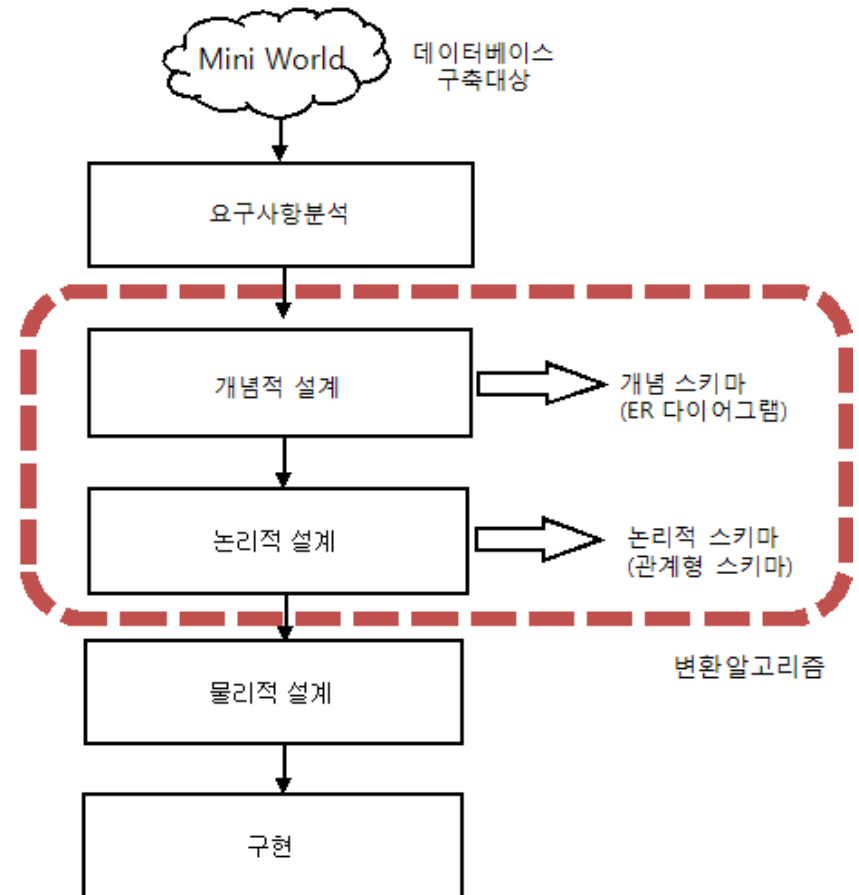
- 어떠한 기본키 값도 널을 가질 수가 없다



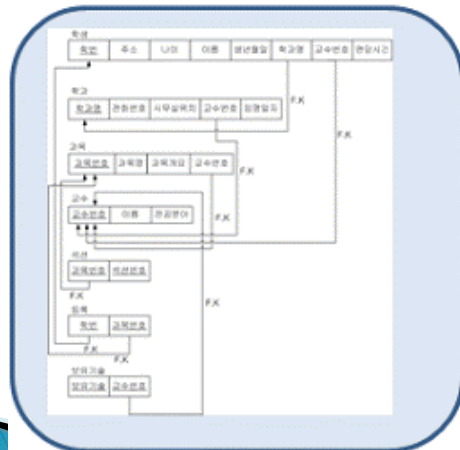
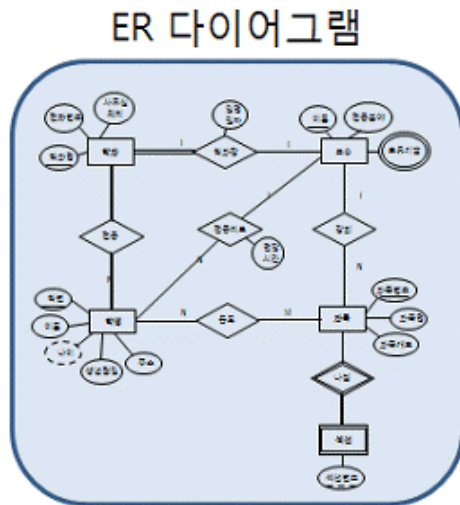
# 12. 논리적 설계

## ▶ 관계형 스키마 작성

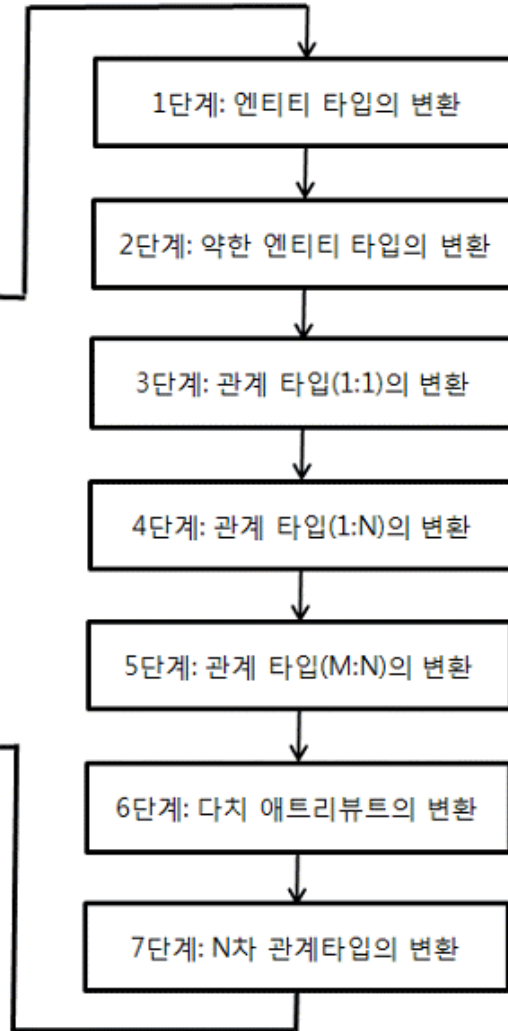
ER 다이어그램을 관계형 스키마로 변환하는 과정



# 12. 논리적 설계



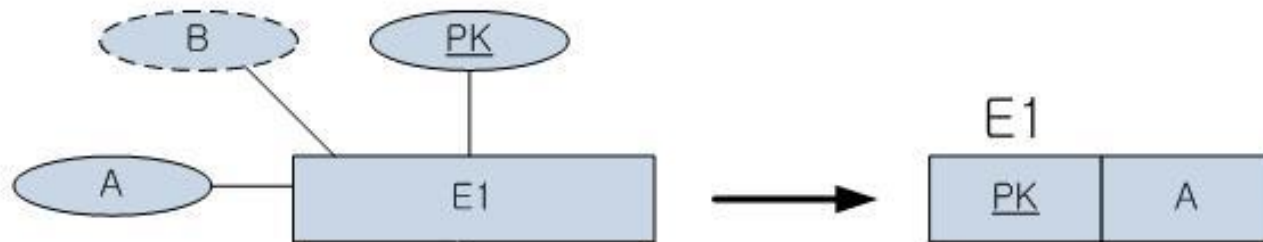
관계형스키마



ER 다이어그램을  
관계형 스키마로  
매핑하는 변환  
과정

# 12. 논리적 설계

- ▶ **엔티티 타입을 관계형 스키마로 매핑**
- ▶ 첫 번째 단계에서는 엔티티 타입을 릴레이션으로 변환한다. 약한 엔티티는 이 때 변환하지 않으며, 약한 엔티티 타입과 구별하기 위해 정규 엔티티 타입을 릴레이션으로 변환한다.
- ▶ 만약 엔티티 애트리뷰트 중 다치 애트리뷰트가 있다면 현재 단계에서는 포함시키지 않는다.
- ▶ 아래 그림은 엔티티 E1의 기본키를 포함한 릴레이션 E1을 생성한 것이다. B는 유도된 애트리뷰트로 이를 릴레이션에 포함할 지 여부는 설계자 선택에 달려 있다.

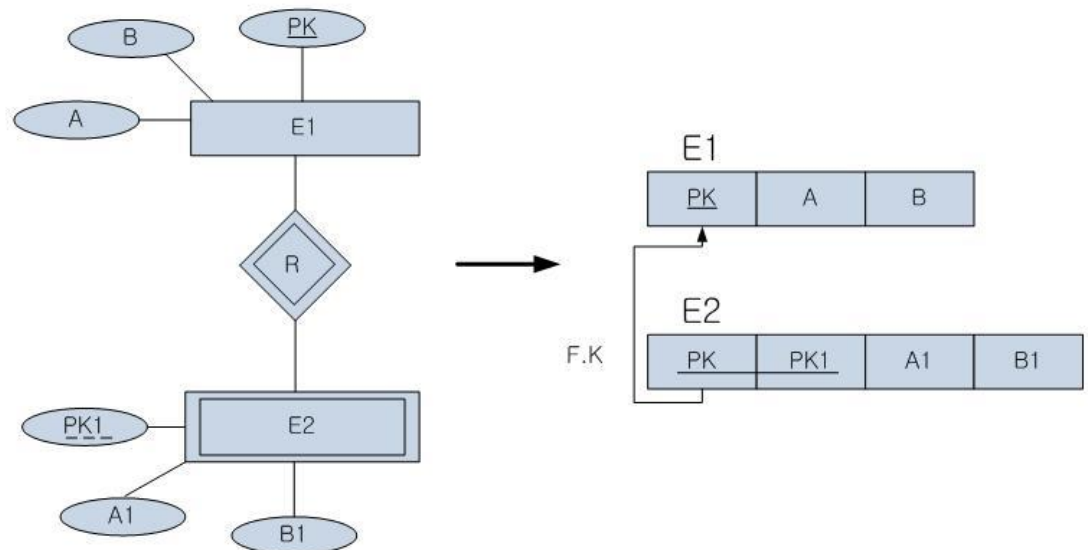


# 12. 논리적 설계-관계형 스키마 작성

## • 약한 엔티티 타입을 관계형 스키마로 매핑

- ▶ 두번째 단계는 약한 엔티티 타입의 변환이다. 소유 엔티티의 기본키와 약한 엔티티의 부분키를 합쳐서 변환된 릴레이션의 기본키로 지정한다.
- ▶ 그리고 약한 엔티티 타입에 있던 모든 애트리뷰트를 포함하는 릴레이션을 생성한다. 그림은 소유 엔티티 E1을 갖는 약한 엔티티 E2에 대해 모든 단순 애트리뷰트를 포함시키고, E1의 기본키를 외래키로 하는 릴레이션 E2를 생성한 것.

- ▶ E1의 기본키를 FK로
- ▶ 하고, 약한 엔티티 E2의
- ▶ 부분키와 합쳐 E2의
- ▶ 기본키로 했다.

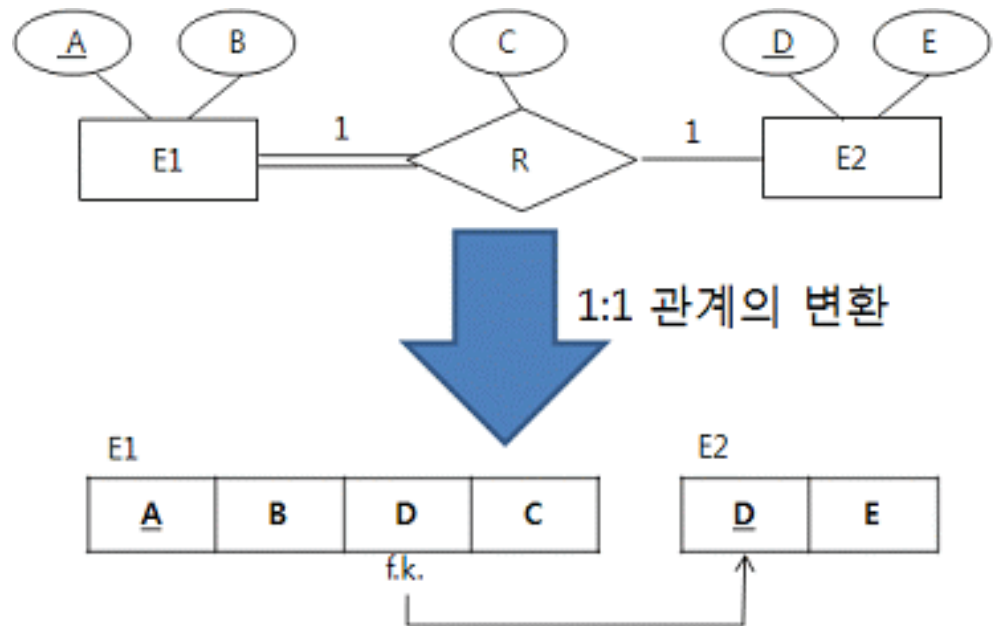


# 12. 논리적 설계-관계형 스키마 작성

## ▶ 1:1 관계 타입의 변환

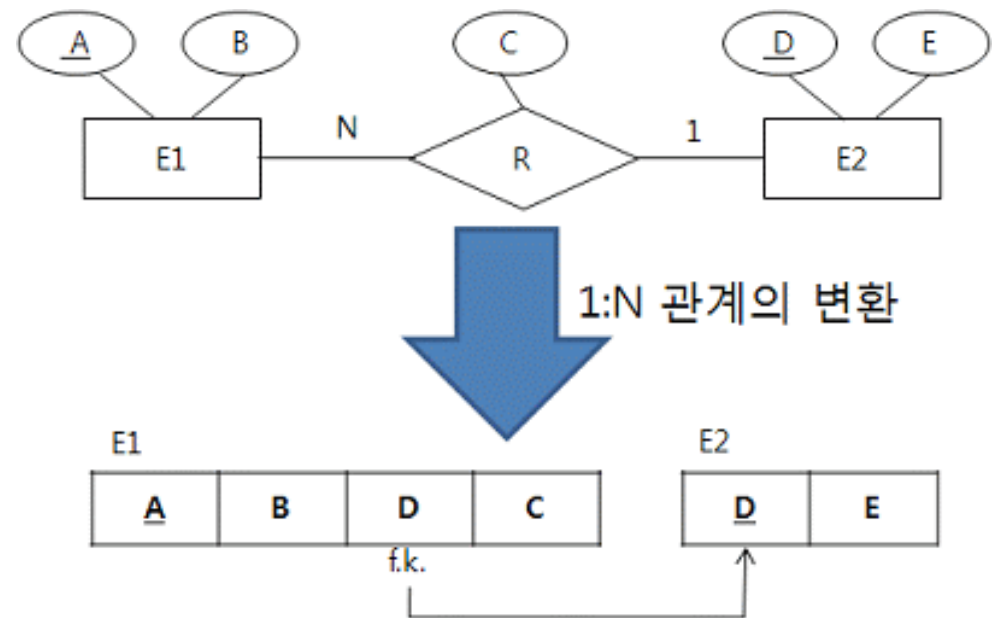
- ▶ 외래키 접근 방식으로 관계 타입 R에 참여하는 엔티티 타입 E1과 E2 중 기본키 하나를 다른 엔티티의 FK로 참여시키고,
- ▶ 만약 R에 단순 애트리뷰트가 있다면 모두 포함시키도록 한다.
- ▶ 만약 E1과 E2 중 한쪽이 관계 타입 R에 대해 전체 참여를 하고 있다면 전체 참여를 하고 있지 않은 엔티티의 기본키를 전체 참여를 하고 있는 엔티티의 외래키로 참여시키는 것이 바람직하다.

- ▶ 교수와 학과 사이의
- ▶ 학과장 관계를 대입해
- ▶ 보라.



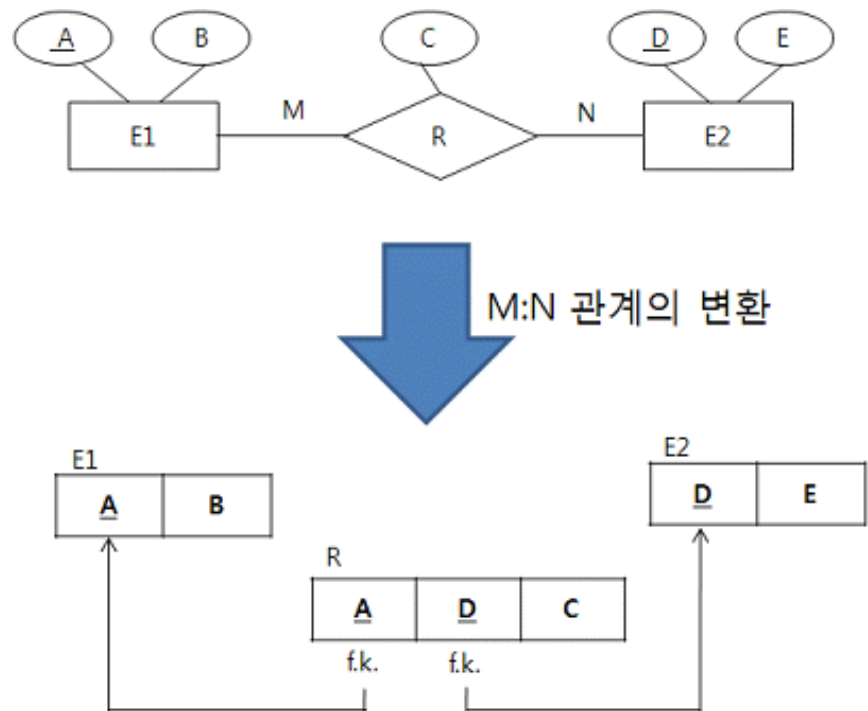
# 12. 논리적 설계-관계형 스키마 작성

- 1:N 관계 타입의 변환
- 1:N의 관계를 찾아 N측 릴레이션에 외래키(FK)를 포함시킨다.
- 관계 타입 R에 단순 애트리뷰트가 있다면 그것도 모두 N측에 포함시킨다.



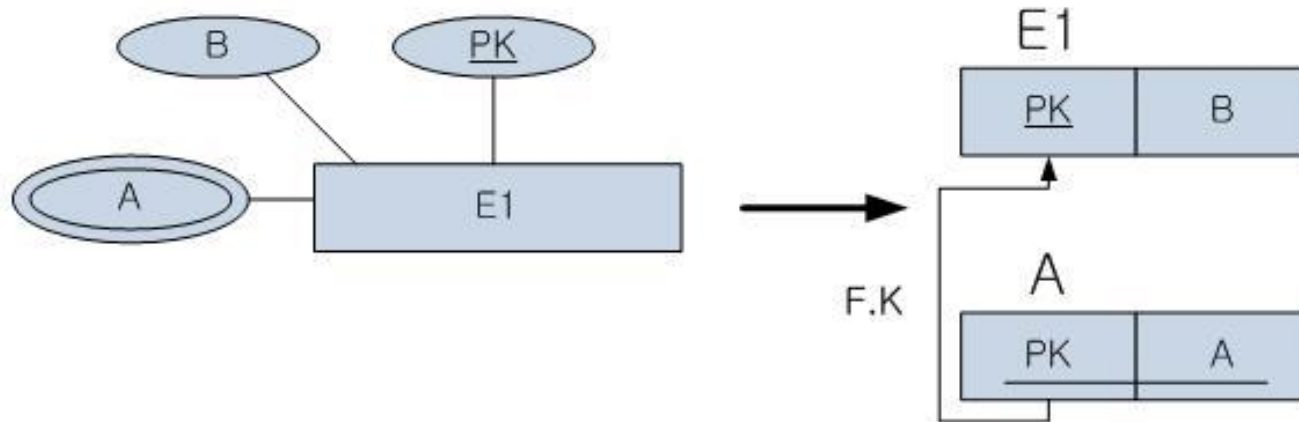
# 12. 논리적 설계-관계형 스키마 작성

- **M:N 관계 타입의 변환**
- M:N 관계 타입 R에 대해 R을 표현하는 새로운 릴레이션을 생성한다.
- 다음으로 관계에 참여하는 모든 엔티티의 기본키를 외래키로 참여시킨다.
- 새로 생성된 R의 기본키는
- 참여한 외래키의 조합이 된다.



# 12. 논리적 설계-관계형 스키마 작성

- 다치 애트리뷰트를 관계형 스키마로 매핑
- 애트리뷰트 하나에 여러 값이 들어갈 수 있는 다치 애트리뷰트의 경우 이를 새로운 릴레이션으로 만든다.
- 다치 애트리뷰트가 속해 있던 엔티티 타입의 기본키를 FK로 참여시킨다.
- 이 둘을 조합한 것이 새로 생성된 릴레이션의 기본키가 된다.

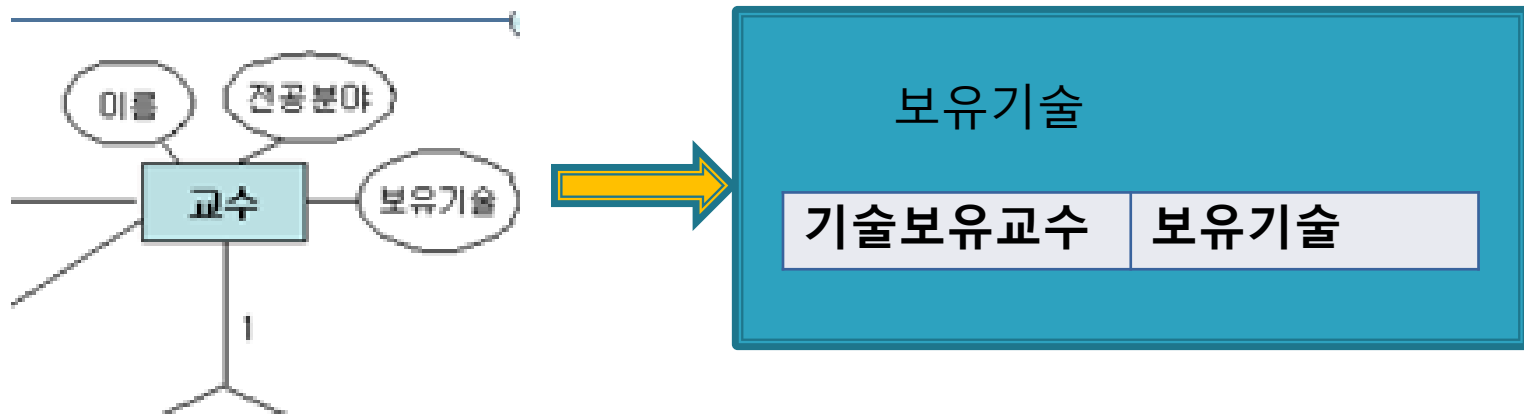




# 12. 논리적 설계-관계형 스키마 작성

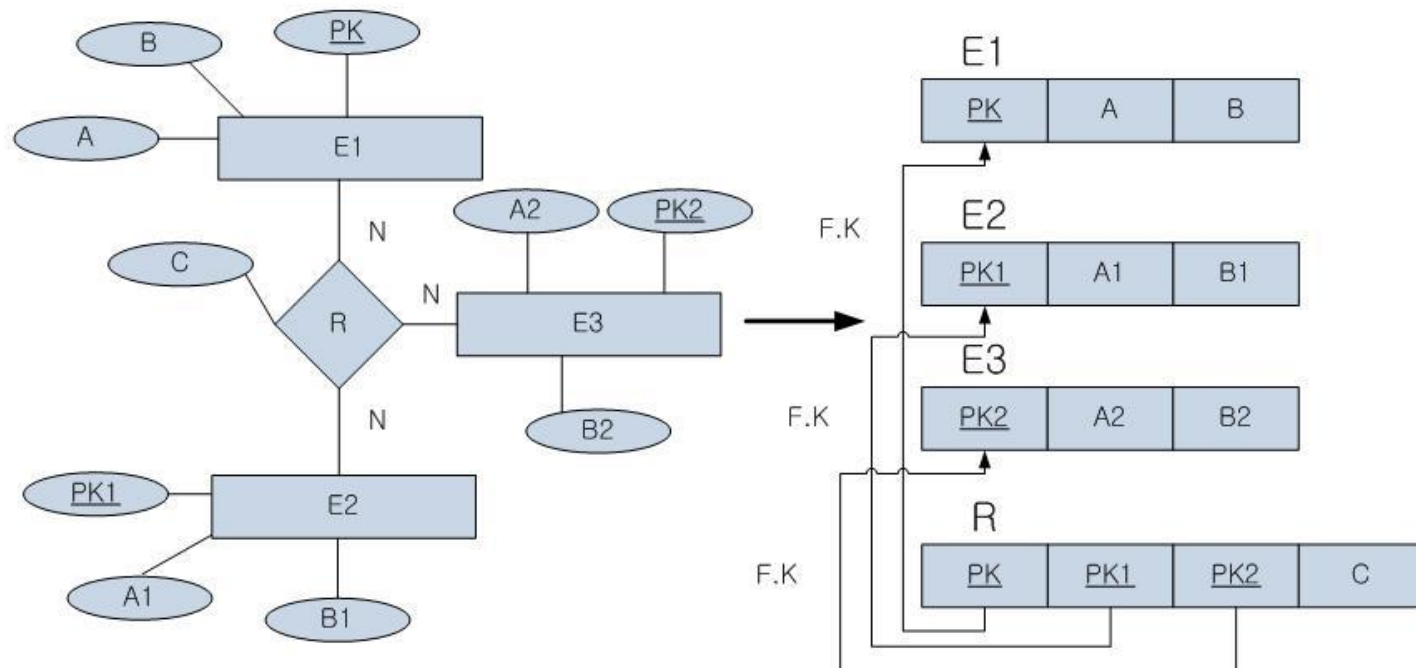
## 다치 에트리뷰트를 관계형 스키마로 매핑

- ▶ 보유 기술 같은 다치 애트리뷰트가 있을 경우,
- ▶ 먼저 보유기술이라는 새로운 릴레이션을 생성하고 다치 속성인 보유 기술을 포함시킨다. 보유기술 속성은 교수 엔티티에 속해있었다. 따라서 교수 엔티티 타입의 기본키인 교수번호를 보유기술 릴레이션의 외래티(FK)로 참여시킨다.
- ▶ 그리고 보유기술과 교수번호를 조합해서 기본키를 구성한다.



# 12. 논리적 설계-관계형 스키마 작성

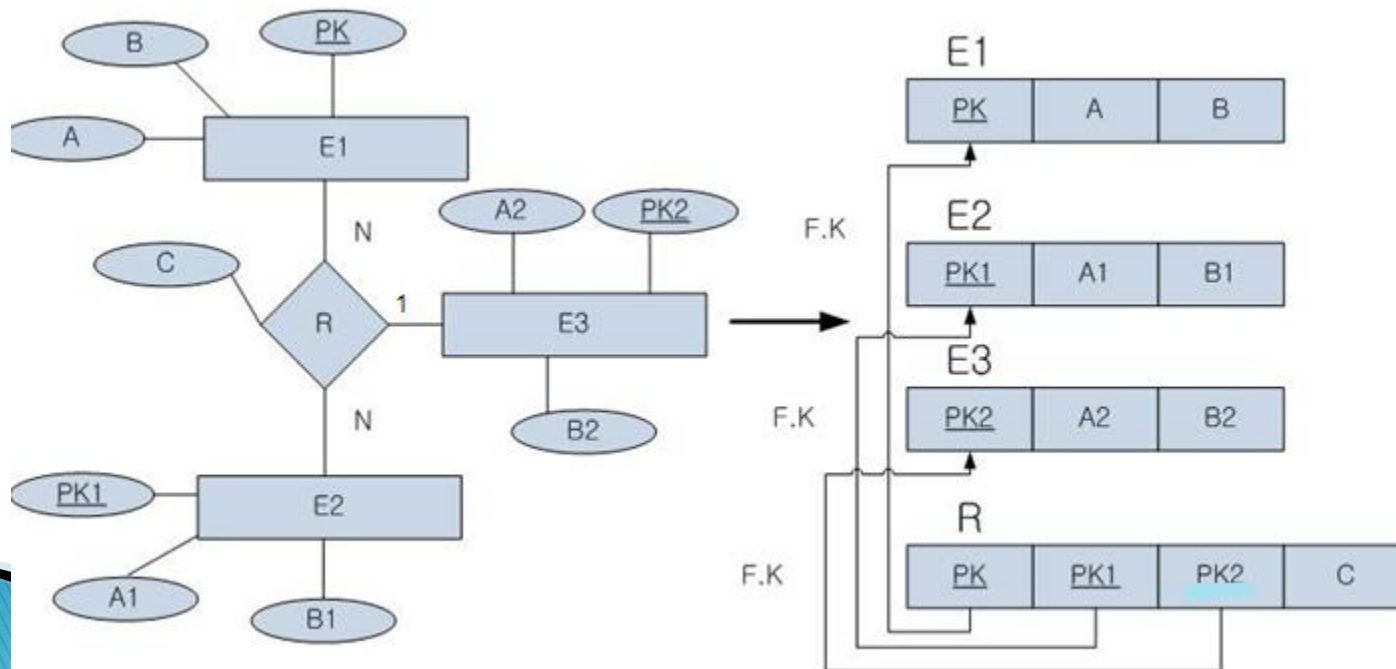
- ▶ 3차 관계 타입을 관계형 스키마로 매핑
- ▶ N차 관계 타입 R에 대응하는 새로운 릴레이션 R을 생성하고, 관계 타입의 모든 속성을 포함시킨 후 관계 타입 R에 참여하는 각각의 엔티티 타입의 기본키를 외래키로 참여시킨다.
- ▶ 새로 생성된 R의 기본키는 FK로 참여시킨 각각의 엔티티 타입의 PK를 조합한 것이다.



# 12. 논리적 설계-관계형 스키마 작성

## ▶ 3차 관계 타입을 관계형 스키마로 매핑

- ▶ 이 때 주의 사항이 있다. 엔티티 타입 중 관계 타입의 카디널리티 제약조건이 1이면 여기에 속하는 엔티티의 기본키는 R에 FK로 참여는 하지만, 기본키에는 참여할 수 없다.



# 복습 문제

- ▶ 데이터베이스를 설계하는 과정을 4단계로 요약하세요.
- ▶ -요구사항 분석-개념적 설계-논리적설계-물리적 설계
- ▶ 개념적 설계에서 사용하는 대표적인 데이터 모델은 무엇입니까? -ER 모델
- ▶ 논리적 설계에서 수행하는 작업 두 가지는 무엇입니까?- DBMS선정, 논리적 스키마로의 매핑
- ▶ 물리적 설계에서 결정해야 하는 중요한 사항은 무엇입니까?-저장 구조, 접근 경로

# 13. 정규화

- ▶ 지금까지 데이터베이스 설계과정을 살펴보았다.
- ▶ 이런 과정을 따라 설계한다 해도 효율적인 데이터베이스를 구축하기 위해서는 **논리적 설계에서도 출된 관계형 스키마를 정제할 필요**가 있다.
- ▶ 정제과정에서 고려해야 할 점이 정규화다.
- ▶ **정규화**란 함수적 종속성이라는 수학적 개념을 분석해서 관계형 스키마를 **단순하고 효율적 구조로 정제해 나가는 과정**을 말한다.

# 13. 정규화(Normalization)란?

- ▶ 데이터베이스 내의 애트리뷰트간의 종속성을 분석해서 하나의 종속성이
- ▶ 하나의 릴레이션으로 표현되도록 분리하는 과정
- ▶ 정규화를 하는 이유?
  - ▶ [1] 현실 세계를 정확하게 표현하는 좋은 데이터베이스 스키마를 생성하고 불필요한 자료의 중복성을 없애서 정보 검색을 쉽게 하기 위해.
  - ▶ [2] 데이터베이스 구조상에서 삽입, 삭제, 수정의 결과로 생기는 이상 현상을 제거하기 위해.

# 13. 정규화

## ▶ 정규화 효과

- ▶ [1] 데이터 중복을 제거하고 일관성을 유지
- ▶ [2] 데이터 모형을 단순화
- ▶ [3] 식별자와 속성과의 종속성 여부 판단 가능
- ▶ [4] DB설계가 용이하며 엔티티와 관계의 누락을 방지

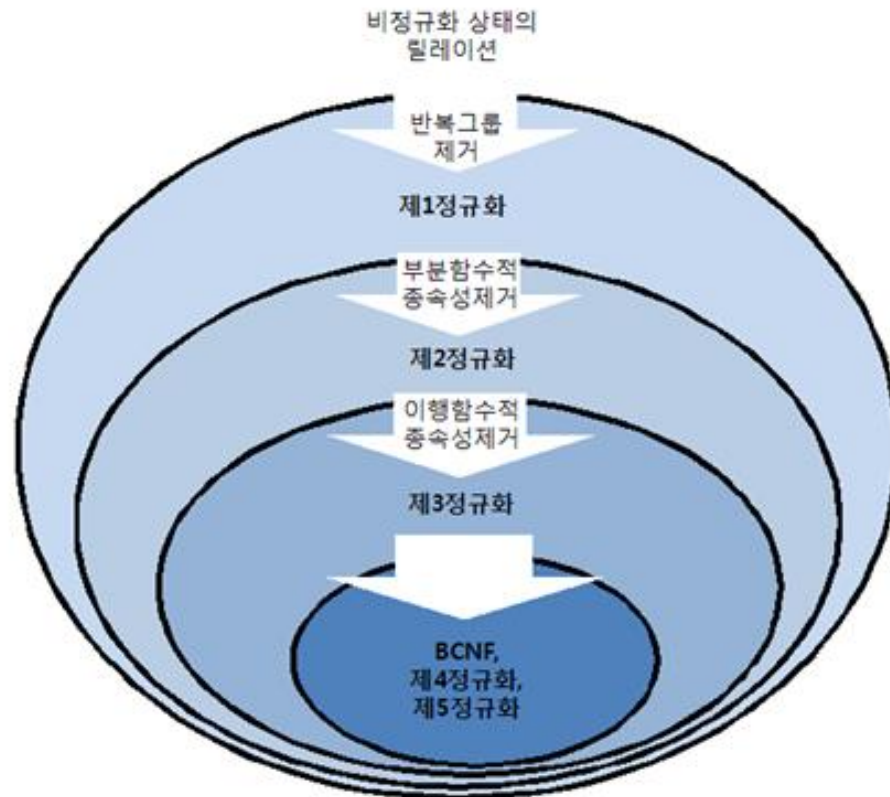
## ▶ 정규화 문제점

- ▶ [1] 테이블 수 증가로 JOIN이 많이 발생하여 응답 속도 지연이 있을 수 있다.
- ▶ [2] 데이터 공간의 비효율적 활용 발생 가능
- ▶ (3정규화 이상의 정규화를 수행하는 경우)

# 13. 정규화

- ▶ 제1, 2, 3, 4, 5, BCNF 정규형으로 나뉜다

## 정규화





# 13. 정규화

정규화	정규화 내용
1차 정규화	<ul style="list-style-type: none"> <li>-복수의 속성 값을 갖는 속성을 분리, 기본 키 식별</li> <li>-<u>반복되는 속성 및 그룹 속성 제거</u></li> <li>-모든 Attribute는 반드시 하나의 값을 가져야 함</li> </ul>
2차 정규화	<ul style="list-style-type: none"> <li>주식별자에 종속적이지 않는 속성의 분리</li> <li>-<u>부분 함수 종속성 제거</u></li> <li>모든 Attribute는 반드시 UID 전체에 종속되어야 함</li> </ul>
3차 정규화	<ul style="list-style-type: none"> <li>속성에 종속적인 속성의 분리</li> <li>-<u>이행함수 종속성 제거</u></li> <li>UID가 아닌 모든 Attribute 간에는 서로 종속 될 수 없음</li> </ul>
BCNF	<ul style="list-style-type: none"> <li>보이스 코드 정규화, 다수의 주 식별자 분리</li> <li>-<u>결정자가 부분키가 아닌 함수 종속성 제거</u></li> <li>-BCNF는 주식별자 안에서 이행함수가 발생하는 것으로 생각하면 됨</li> </ul>
4차 정규화	<ul style="list-style-type: none"> <li>키들간 종속성 제거</li> <li>함수 종속성이 아닌 <u>다중값 종속성 제거</u>(함수적 종속성이 아님)</li> </ul>
5차 정규화	<ul style="list-style-type: none"> <li><u>결합 종속일 경우는 두 개 이상의 N개로 분리</u></li> <li>후보 키를 통하지 않는 결합 종속성 제거(<u>join 종속성 제거</u>)</li> </ul>

# 13 정규화-제1 정규화

## • 제1 정규형(1NF)

- 반복되는 값이나 여러 값을 갖는 다치 애트리뷰트를 제거해서 각 애트리뷰트가 반드시 하나의 값을 갖도록 하는 과정
- 여러 값을 가진 컬럼이 존재할 수 없다. 즉 반복되는 그룹이 존재해서는 안 된다. 각 행과 열에는 하나의 값만이 올 수 있다.
- 반복되는 그룹 속성을 제거한 뒤 기본 테이블의 기본키를 추가해 새로운 테이블을 생성하고 기존 테이블과 1:N의 관계를 형성한다.
- 제1정규화(1NF)는 만족하게 되어도 각각 삽입, 수정, 삭제를 수행하는 과정에서 이상이 발생할 수 있다

# 13 정규화-제1 정규화

학생수강1

학번(pk)	학과명	학과연락처	과목번호	학점
111	컴퓨터	333-3333	A001	A
222	컴퓨터	333-3333	B011,C003	A, B

다치애트리뷰트를 제거

학생수강2

학번(pk)	학과명	학과연락처	과목번호	학점
111	컴퓨터	333-3333	A001	A
222	컴퓨터	333-3333	B011	A
222	컴퓨터	333-3333	C003	B

# 13 정규화-제1 정규화

- ▶ 위와 같은 과정을 거치면 제1정규화(1NF)는 만족한다. 하지만 삽입, 수정, 삭제를 수행하는 과정에서 이상이 발생할 수 있다.
- ▶ **삽입이상**
- ▶ 학생수강에서 학번을 제외한 새로운 정보("디자인", 555-5555, D001)만을 넣으려고 하면 엔티티 무결성 제약에 의해 삽입이상이 발생한다.
- ▶ 기본키값으로 넣을 가질 수 없기 때문에

# 13 정규화-제1 정규화

- ▶ 삭제 이상
- ▶ 테이블 내 하나만 있는 튜플의 일부 정보를 삭제할 때 삭제하지 말아야 할 정보까지 삭제되면서 발생한다.
- ▶ 학번이111인 학생의 수강과목을 삭제하고자 한다.
- ▶ 이 경우 학생의 수강과목을 삭제하면 과목번호 등 삭제되지 말아야 할 해당 학생의 다른 정보까지 삭제된다.

# 13 정규화-제1 정규화

## ▶ 수정이상

중복되는 값 일부만 수정되어 불일치성이 생길 때 발생

222학번 학생의 학과명을 컴퓨터에서 경영학과로 바꾸려면 테이블 내 모든 값을 찾아 바꿔야 하지만 그렇지 못하면 데이터의 일관성을 잃게 된다.

따라서 제1정규화에 이어 추가적인 정규화를 거쳐 이 문제들을 해결해야 함

# 13. 정규화-제2정규화

## • 제2정규형(2NF)

- 기본키에 종속적이지 않은 애트리뷰트를 분리하는 과정

학번+과목번호가 기본키 일 때 성적은 기본키에 완전 종속된다.

(학번+과목번호)→(성적)

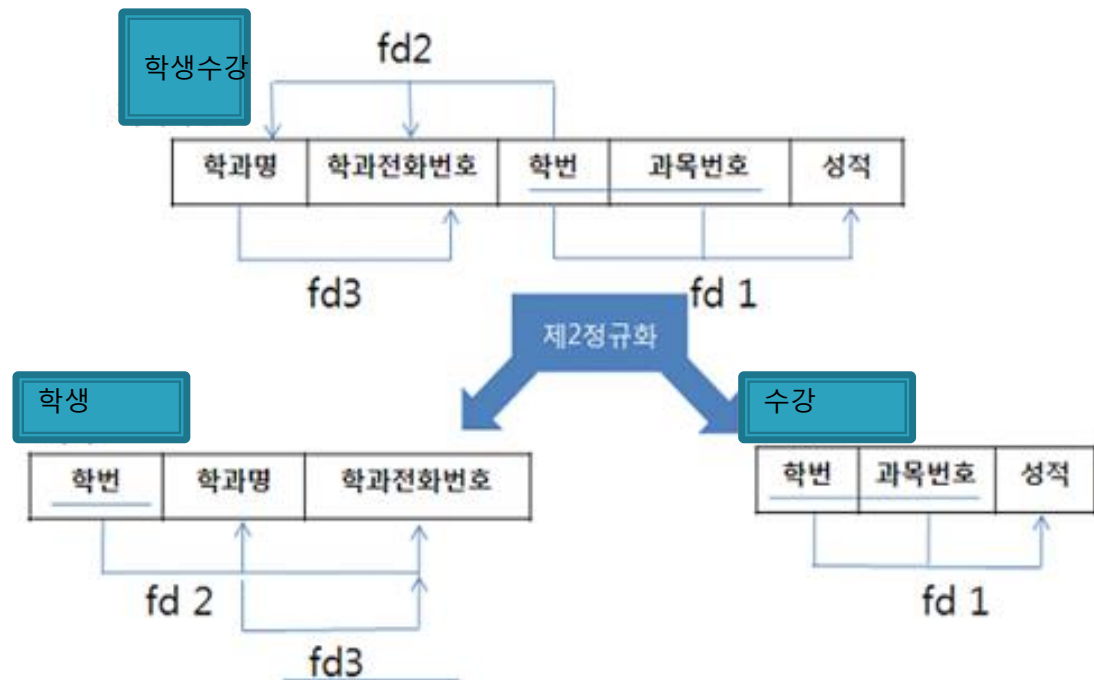
하지만 학과명, 학과 전화번호는

(학번+과목번호)

→(학과명, 학과전화번호)

식으로 완전한 함수적 종속은 아니다.

왜냐하면 학번,과목번호의 부분집합인 학번에 부분적으로 종속되기 때문에



# 13. 정규화-제2정규화

- ▶ 이러한 문제 해결을 위해 학생수강 릴레이션을 학생과 수강 두 개의 릴레이션으로 나눈다.
- ▶ 학생 (학번-> 학과명,학과전화번호)
- ▶ : 여기서 기본키는 학번이다.
- ▶ 수강(학번+과목번호-> 성적)
- ▶ : 여기서 기본키는 과목번호이며 외래키는 학생 릴레이션을 참조하는 학번이다.)



# 13. 정규화-제2정규화

- ▶ 제2정규화 대상 테이블은 키가 여러 컬럼으로 구성된 경우, 즉 복합키로 구성된 경우이다.
- ▶ 복합키로 구성되어 있는 테이블의 모든 컬럼들은 복합키 전체에 의존해야 하며 그렇지 않을 경우 제2 정규화의 대상이 된다.
- ▶ 복합키의 일부분에 의해 종속되는 것을 부분적 함수 종속관계라고 하며 이를 제거하는 것이다.
- ▶ 복합키 전체에 의존하지 않는 컬럼이 존재하는 경우 기본 테이블에서 제거시켜 새로운 테이블을 생성해 주어야 한다.

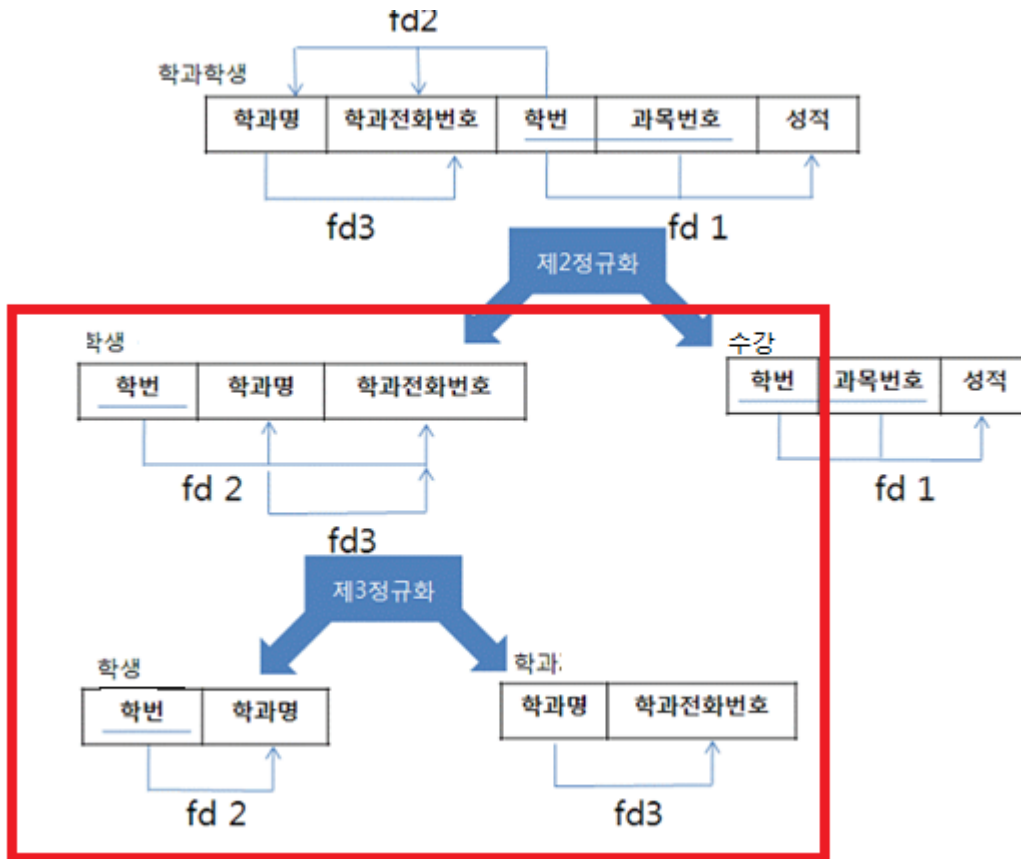
# 13. 정규화-제3정규화

- 제3정규형(3NF)

- 이행함수적 종속성을 갖는 애트리뷰트를 제거하는 과정
- 이행함수적 종속성 : 기본키가 아닌 애트리뷰트가 다른 애트리뷰트의 값과 관계가 있는 경우를 뜻한다

▶ 즉 기본키가 아닌 애트리뷰트 간에는 종속관계를 가질 수 없도록 하는 것.

# 13. 정규화-제3정규화



학생 릴레이션을 보면  
학과전화번호는 학번에 의해  
완전하게 함수적 종속이지만,  
학과는 다시 학과전화번호를  
결정한다. 이럴 때 갱신 이상  
이 발생할 수 있다.

이러한 이행적 종속성을 제  
거하기 위해 다음과 같이 학  
생 릴레이션을 다시 학생과  
학과 두 개의 릴레이션으로  
분리하게 된다.

# 13. 정규화

- ▶ 연습문제
- ▶ 아래 테이블을 보고 1,2,3 정규화를 진행해보세요.

제 품

제품코드	제품명	재고수량		주문번호	고객코드	고객명	고객신용도	주문일자	주문수량	
175	전화기	35,042		96071	A011	신명상사	상	19960614	5,000	
				96110	C024	정명물산	중	19960622	3,500	
329	냉장고	20,375		96089	D035	(주)한신	상	19960617	1,200	
				96071	A011	신명상사	상	19960614	800	
				96174	B047	진영통상	중	19960629	500	
854	컴퓨터	13,468		96089	D035	(주)한신	상	19960617	700	
				96036	H007	미래정보	상	19960608	200	
				96193	H007	미래정보	상	19960704	300	

반복그룹

# 13. 정규화-제3정규화

- ▶ 1 정규화 후
  - ▶ 제품: 제품코드->제품명,재고수량
  - ▶ 주문: 주문번호+고객코드,->고객명,고객신용도,주문일자,주문수량
- ▶ 2 정규화후 (부분적 함수 종속성 제거)
  - ▶ 제품: 제품코드->제품명,재고수량
  - ▶ 주문: 주문번호+고객코드-> 고객명,고객신용도, 주문일자
  - ▶ 주문상세: 주문번호+제품코드-> 주문수량
- ▶ 3 정규화후 (이행적 함수 종속성 제거)
  - ▶ 제품: 제품코드->제품명,재고수량
  - ▶ 주문번호->주문일자 주문수량
  - ▶ 고객코드->고객명,고객신용도

# 14. 역정규화

- ▶ 역정규화란 시스템 성능을 고려해서 기존 설계를 재구성하는 것을 말한다.
- ▶ 과도한 정규화는 시스템 성능을 저해할 수 있다.
- ▶ 과도한 정규화로 릴레이션을 최소 단위로 분리하는데 이렇게 분리된 릴레이션들에서 데이터를 추출하려면 여러 테이블을 조인해야 원하는 데이터를 얻을 수 있다. 여러 테이블을 조인하는 경우 SQL문이 복잡할 뿐 아니라 시스템 성능이 저하됨.
- ▶ 역정규화는 이러한 성능 저하를 해결하고자 정규화에 위배되는 행위를 하는 것을 말한다.