

데이터베이스

강사 : 강병준

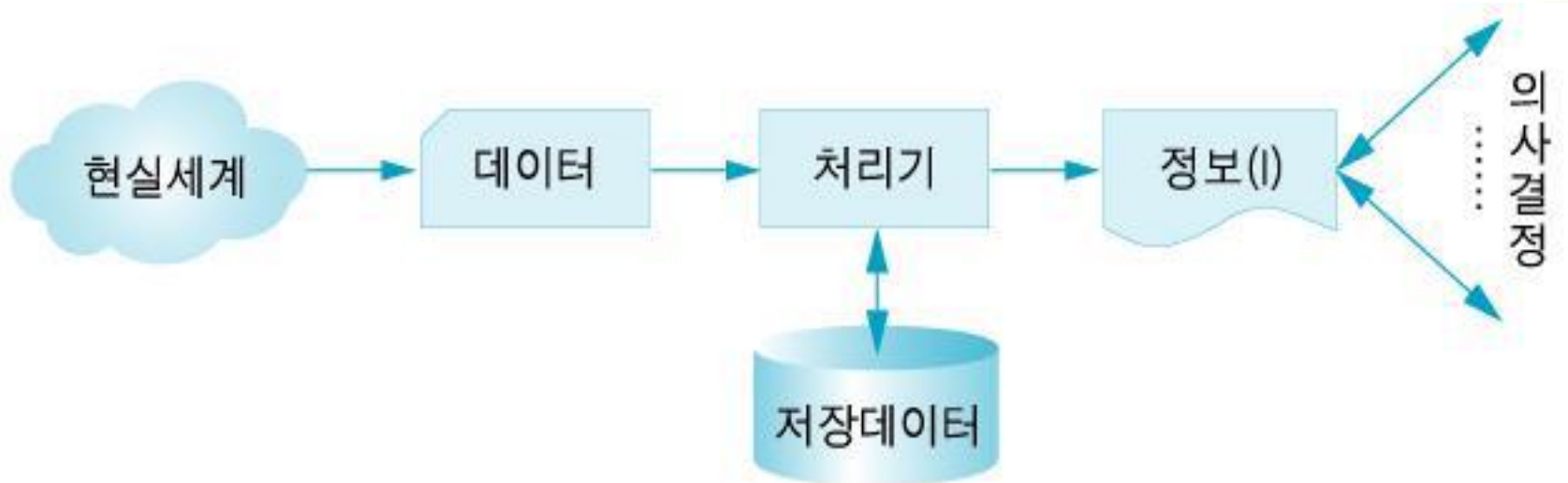
데이터란 ?

- 데이터 (**data**)
 - 관찰이나 측정을 통해 현실 세계에서 수집한 사실(**facts**)이나 값(**value**)
- 정보 (**information**)
 - 어떤 상황에서 의사 결정을 돕는
데이터의 유효한 해석이나 상호 관계
 - 데이터를 처리(**data processing**)해서 얻을 수 있는 경과



데이터란 ?

- 정보 시스템(**information system**)
 - 하나의 기관을 위해 데이터를 수집, 조직, 저장하고 정보를 생성, 분배하는 수단
 - 경영 정보 시스템, 군사 정보 시스템 등



파일시스템

데이터를 가공, 처리하여 유용한 정보를
얻기 위한 기본적인 데이터 저장 도구로
초기에 사용된 것

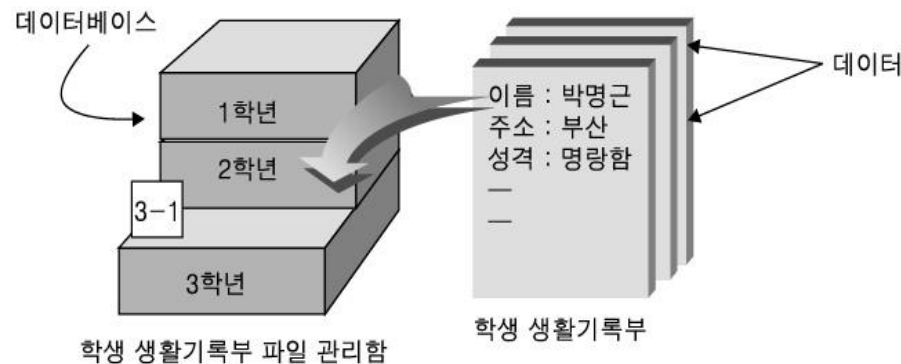
- SAM
- DAM
- ISAM
- VSAM

파일시스템의 문제점

- 데이터 종속
- 데이터 중복
 - 일관성 - 동일성을 유지하기 위해 데이터 중복을 피함
 - 보안성- 동일한 수준에서 보안이 유지
 - 경제성- 저장되는 공간에 대한 비용 절감
 - 무결성- 데이터가 정확성을 유지

데이터베이스 정의

- 데이터베이스란?
 - 논리적으로 연관된 하나 이상의 자료의 모음
 - 특정 조직의 응용 업무에 공동으로 사용하기 위하여 운영상 필요한 데이터를 중복을 최소화하여 컴퓨터 기억 장치 내에 모아 놓은 집합체
 - 데이터의 중복 없이 서로 관련되어 있어 관련된 모든 응용환경에서 사용될 수 있는 데이터들의 집합
 - 자료를 획득하여 체계적으로 분류하고 정리한 다음, 컴퓨터에서 처리가 가능하도록 전자적 형태로 저장한 것
 - 하나의 주제와 관련된 의미있는 데이터(Data)들의 모음



데이터베이스의 정의

- 통합된 데이터(integrated data)
 - 데이터베이스는 똑같은 데이터가 원칙적으로 중복되어 있지 않다는 것을 말하며, 데이터의 중복은 일반적으로 관리상의 복잡한 부작용을 초래합니다.
- 저장된 데이터(stored data)
 - 컴퓨터가 접근할 수 있는 기억장치에 저장된 데이터를 말합니다. 주로 하드디스크에 저장되어 관리됩니다.
- 운영 데이터(operational data)
 - 존재 목적이 명확하고 유용성을 지니고 있는 데이터를 말합니다. 즉 단순히 데이터를 모아둔 개념이 아닌 병원 관리를 위한 데이터 구축과 같은 목적이 분명한 데이터여야만 합니다.
- 공용 데이터(shared data)
 - 여러 사용자들이 서로 다른 목적으로 사용하는 공유 가능한 데이터를 말합니다.

데이터베이스의 특징

(1) 실시간 접근성(Real-time accessibility)

다수의 사용자의 요구에 대해서 처리 시간이 몇 초를 넘기지 말아야 한다는 의미입니다.

(2) 지속적인 변화(Continuos evolution)

데이터베이스에 저장된 데이터는 최신의 정보가 정확하게 저장되어 처리되어야 합니다.

(3) 동시 공유(Concurrent sharing)

동일 데이터를 동시에 서로 다른 목적으로 사용할 수 있어야 합니다.

(4) 내용에 대한 참조

데이터베이스 내에 있는 데이터 레코드들은 주소나 위치에 의해 참조되는 것이 아니라 가지고 있는 값에 따라 참조해야 합니다.

데이터베이스의 목적

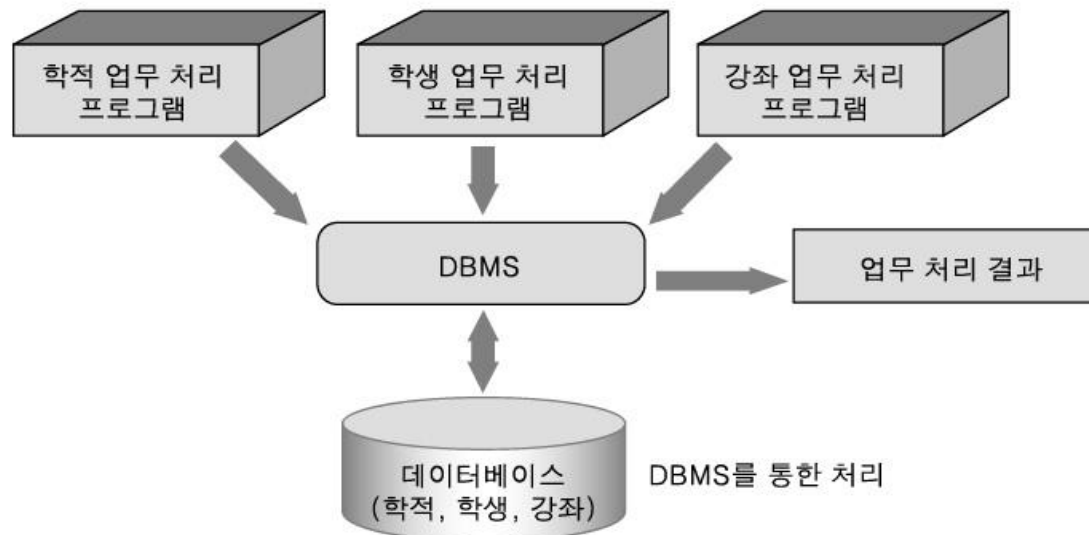
1. 데이터 독립 유지-데이터와 응용 프로그램의 독립
 - 1) 물리적 데이터 독립(physical data independence)
 - 2) 논리적 데이터 독립(logical data independence)
2. 데이터 중복의 최소화
3. 데이터의 공유
4. 데이터의 보안 유지
5. 데이터 무결성(Integrity)의 유지
6. 데이터베이스를 이용한 데이터베이스 관리 시스템은 파일 관리 시스템이 갖는 구조적인 한계를 극복하기 위해 고안
7. 데이터와 응용 프로그램의 종속성을 최소화
 - 1) 파일 관리 시스템의 가장 큰 문제인 데이터와 프로그램의 밀접한 관계를 제거하기 위해 고안
 - 2) 데이터를 저장하거나 검색하는 등의 기능을 데이터베이스 관리 시스템이 전담하도록 함으로써 파일 관리 시스템이 지닌 데이터와 응용 프로그램의 종속성을 최소화, 데이터 변경으로 인해 생기는 프로그램의 수정, 관리를 최소화.
8. 데이터의 중복으로 인한 비 일관성 최소화
 - 1) 공통된 데이터 저장 공간을 사용하기 때문에 데이터의 중복에서 발생하는 비 일관성을 최소화

Database의 역사

연도	역사
1963	데이터베이스라는 용어가 'Development and Management of Computer - Center Data Bases'라는 심포지움에서 처음 사용
1963	최초의 범용 DBMS인 설계 : GE에서 개발한 'Integrated DataStore
1970	E. F. Codd는 관계형 데이터베이스 모델 제안
1976	Chen이 개체 관계(ER) 모델 제안
1979	IBM의 System/R 프로젝트의 엔지니어들이 오라클의 전신인 Relational Software 설립함 이들이 최초의 상용 관계형 DBMS인 오라클을 출시함
1983	상용 관계 DBMS 등장(DB2, ORACLE, SYBASE 등)
1986	데이터베이스를 다루는 언어인 SQL이 관계형 데이터베이스 관리 시스템의 표준 언어로 채택
1990 ~	상용 객체 지향 DBMS 등장

DBMS

- 기업이 지속적으로 유지관리 해야 할 데이터의 집합(Database)
- 방대한 양의 데이터를 편리하게 관리하고 효율적으로 저장하고 검색할 수 있는 환경을 제공해 주는 시스템 소프트웨어(DBMS)
- 데이터베이스에 대한 사용자의 모든 요구를 수행할 수 있는 기능을 갖도록 하는 각 단계별 구조와 이들 사이의 인터페이스 및 데이터베이스 언어로 구성된 소프트웨어



데이터베이스 관리 시스템의 기능

정의 기능 (Definition)

1. 데이터의 형태, 구조, 데이터베이스의 저장에 관한 내용을 정의하며, 다양한 응용 프로그램과 데이터베이스가 서로 인터페이스 할 수 있는 방법 제공
2. 구현된 하나의 물리적 구조의 데이터베이스로 여러 사용자가 다양한 형태의 데이터를 요구해도 이를 지원할 수 있도록 가장 적절한 데이터베이스 구조를 정의할 수 있는 기능
3. 특징
 - 모든 응용 프로그램이 요구하는 데이터 구조를 지원할 수 있게끔 데이터베이스의 논리적 구조와 그 특성을 목표 DBMS가 지원하는 데이터 모델에 맞게 기술.
 - 데이터베이스를 물리적 저장장치에 저장하는 데 필요한 명세 포함.
 - 데이터의 논리적 구조와 물리적 구조 사이에 변환이 가능하도록 명세.

데이터베이스 관리 시스템의 기능

조작 기능 (Manipulation)

1. 사용자의 요구에 따라 검색, 갱신, 삽입, 삭제 등을 지원하는 기능
2. 사용자와 데이터베이스 사이의 인터페이스를 위한 수단 제공.
3. 특징
 - 1) 사용하기 쉽고 자연스러워야 한다.
 - 2) 명확하고 완전해야 한다.
 - 3) 효율적이어야 한다.

조작 기능 (Manipulation)

1. 데이터베이스의 내용에 대해 정확성과 안전성을 유지하는 기능(무결성 유지, 보안, 병행 수행 제어)
2. 특징
 - 1) 데이터베이스를 접근하는 갱신, 삽입, 삭제 작업의 정확하게 수행되게 하여 데이터의 무결성이 파괴되지 않도록 제어할 수 있어야 한다.
 - 2) 정당한 사용자가 허가된 데이터가 접근할 수 있게끔 보안을 유지하고 권한을 검사할 수 있어야 한다.
 - 3) 여러 사용자가 데이터베이스를 동시에 접근하여 데이터를 처리할 때 데이터베이스와 처리 결과가 항상 정확성을 유지하도록 병행 제어를 할 수 있어야 한다.

데이터베이스 관리 시스템의 기능

데이터베이스 관리 시스템의 장점

1. 데이터 중복의 최소화
2. 데이터 공유
3. 데이터의 일관성 유지
4. 데이터의 무결성 유지
5. 데이터의 보안 보장
6. 데이터 관리 표준화
7. 데이터 관리의 유연성
8. 자료에 대한 접근성 및 응답성 향상

데이터베이스 관리 시스템의 단점

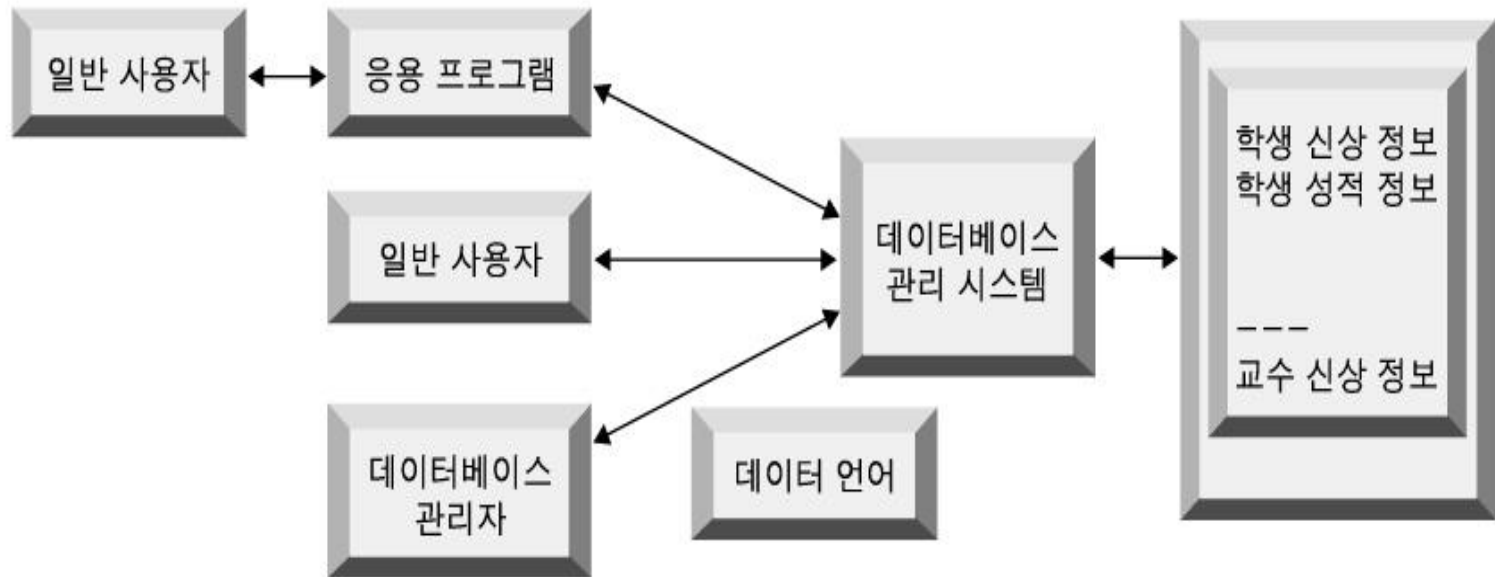
1. 추가적인 운영비 증대
2. 자료 처리의 복잡화
3. 복잡한 백업과 복구

데이터 독립성

1. 데이터베이스 구조 변화로 인한 영향을 응용 프로그램 (또는 논리적 구조)에 미치지 않도록 하는 것.
2. 종류
 - 1) 논리적 데이터 독립성
 - 2) 물리적 데이터 독립성
3. 논리적 데이터 독립성
 - 1) 응용 프로그램에 영향을 주지 않고 논리적 데이터 구조를 변경할 수 있게 하는 것
 - 2) 외부/개념 스키마간의 사상에 의해 제공
 - 3) 데이터베이스에 새로운 데이터 항목이나 레코드를 추가해도, 현재 정의된 사용자의 관점이나 사용되고 있는 응용 프로그램 중에서 직접 관련되지 않는 사용자 관점과 응용 프로그램은 영향을 받지 않는 성질
4. 물리적 데이터 독립성
 - 1) 응용 프로그램과 데이터베이스의 논리적 구조에 영향을 주지 않고 물리적 구조를 변경할 수 있게 하는 것
 - 2) 개념/내부 스키마 간의 사상에 의해 제공된다.

데이터베이스 관리 시스템 구성 요소

1. 데이터베이스
2. 데이터베이스 관리 시스템
3. 데이터 언어(Data Language)
4. 데이터베이스 관리자
5. 응용 프로그램(어)과 사용자



관계형 DBMS

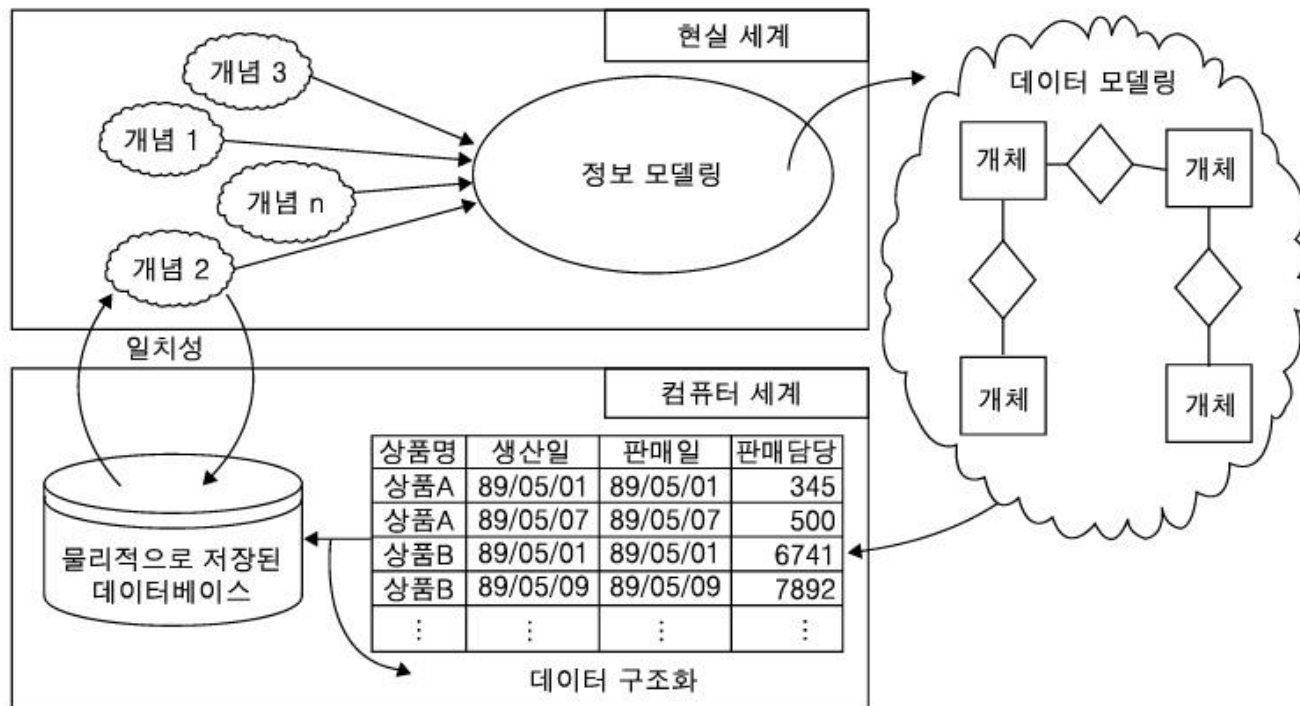
❖ 관계형 데이터베이스(RDBMS - Relation Data Base Management System)

- ✓ 데이터베이스를 테이블의 집합으로 설명하는 데이터베이스 관리 시스템
- ✓ 1970년 Codd에 의해 개발
- ✓ 상용 관계형 데이터베이스 관리 시스템으로는 Oracle, IBM의 DB2, 마이크로소프트의 MS-SQL Server, Sybase를 포함한 SAP의 HANA DB, TeraData Database, Tiberio 등이 있음
- ✓ 오픈 소스 DBMS는 MySQL, PostgreSQL, SQLite 등이 있으며 MySQL이 Oracle에 인수된 이후 가장 많이 사용되는 MySQL의 포크는 MariaDB

데이터 모델링

데이터 모델링(Data Modelling)

- 복잡한 실세계를 단순화하여 실세계에 존재하는 개체들을 식별하여 이들 객체와 객체 사이의 관계를 정의함으로써 컴퓨터상의 데이터베이스를 추상화된 개념으로 이해하기 쉽게 할 뿐만 아니라 사용자들 사이의 의사소통을 원활히 할 수 있도록 도와주는 도구



모델링

❖ 모델링의 이해

- ✓ 모델링의 정의: 복잡한 현실세계를 일정한 표기법에 의해 표현하는 것
 - Webster 사전
 - ◆ 가설적 일정 양식에 맞춘 표현
 - ◆ 어떤 것에 대한 예비 표현으로 그로부터 최종 대상이 구축되도록 하는 계획으로서 기여하는 것
 - 복잡한 현실 세계 를 단순화시켜 표현하는 것
 - 모델이란 사물 또는 사건에 관한 양상(Aspect)이나 관점(Perspective)을 연관된 사람이나 그룹을 위하여 명확하게 하는 것
 - 모델이란 현실 세계의 추상화된 반영
- ✓ 모델링의 관점
 - 프로세스 모델링
 - 데이터 모델링
 - 상관 관계 모델링

모델링

❖ 모델링의 이해

✓ 모델링의 3가지 관점

● 데이터 관점

- ◆ 업무가 어떤 데이터와 관련이 있는지?
- ◆ 데이터간의 관계는 무엇인지?
- ◆ 비즈니스 프로세스에서 사용하는 데이터 - Data
- ◆ What
- ◆ 정적 분석과 구조 분석

● 프로세스 관점

- ◆ 업무(비즈니스 프로세스에서 수행하는 작업)가 실제하고 있는 일이 무엇인지?
- ◆ 무엇을 모델링해야 하는지?
- ◆ How
- ◆ 시나리오 분석, 도메인 분석, 동적 분석

● 데이터와 프로세스의 상관 관점

- ◆ 업무가 처리하는 일의 방법에 따라 데이터는 어떻게 영향을 받고 있는지?
- ◆ Interaction(상호작용)
- ◆ CRUD

모델링

❖ 모델링의 이해

✓ 모델링의 특징

- 추상화(모형화, 가설적): 현실 세계를 일정한 형식에 맞추어 표현을 한다는 의미로 다양한 현상을 일정한 양식인 표기법에 의해 표기한다는 것
- 단순화: 복잡한 현실 세계를 약속된 규약에 의해 제한된 표기법이나 언어로 표현하여 쉽게 이해할 수 있도록 하는 개념
- 명확화: 누구나 이해하기 쉽게 하기 위해 대상에 대한 애매 모호함을 제거하고 정확하게 현상을 기술하는 것
- 모델링의 재정의: 현실 세계를 추상화, 단순화, 명확화 하기 위해 일정한 표기법에 의해 표현하는 기법
- 정보 시스템 구축에서의 모델링 활용
 - ◆계획/분석/설계 단계: 업무를 분석하고 설계하는데 이용
 - ◆구축/운영 단계: 변경과 관리의 목적으로 이용

데이터 모델링

개체-관계(Entity-Relationship) 모델

- 현실 세계의 많은 데이터 중에서 관심의 대상이 되는 데이터를 언어보다 좀더 형식화된 다이어그램을 사용하여 표현한 것
- 1976년 Chen에 의해 제안
 - 데이터에 대해 관리자, 사용자, 프로그래머들이 서로 다르게 인식되고 있는 뷰들을 하나로 통합할 수 있는 단일화된 설계안을 만들기 위해서 사용.
 - 서로 다른 뷰들을 충족시킬 수 있는 데이터 처리와 제약조건 등의 요구사항들을 정의하기 위해 사용.



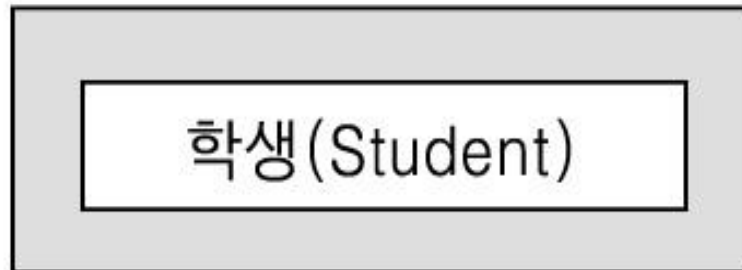
사용자의 요구로부터 개체-관계 모델의 설계

데이터 모델링

구성 요소

– 개체(엔티티, Entity)

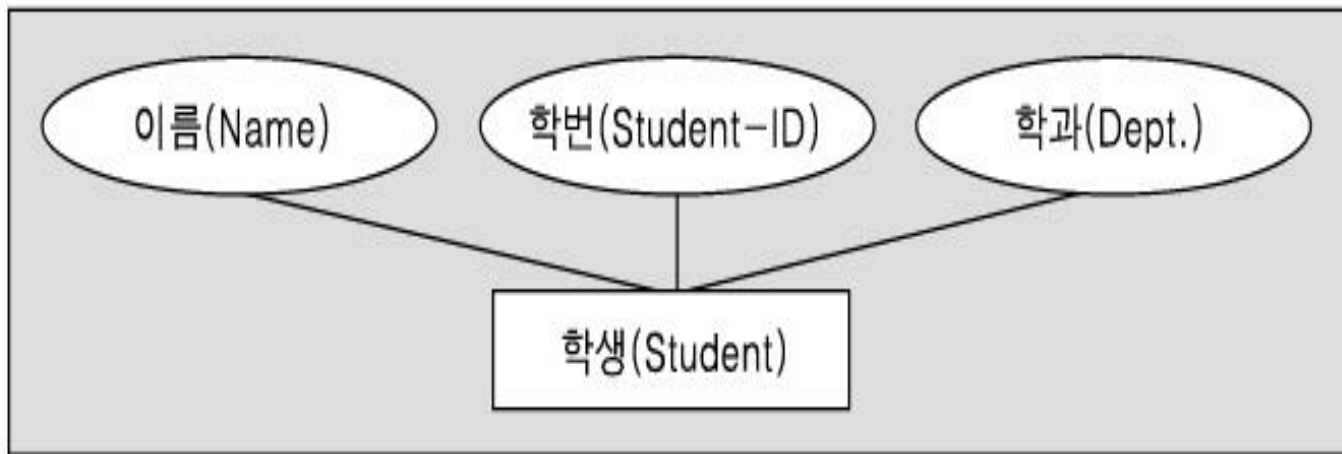
- 데이터 수집의 대상이 되는 정보 세계에 존재하는 사물(thing)
- 종류
 - 개념적 개체 : 장소, 사건 등과 같은 눈에 보이지 않는 것
 - 물리적 개체 : 물건 등과 같은 눈에 보이는 것. 즉 현실 세계에 존재하는 사물.
- 각 개체는 속성(attribute)으로 알려진 특성들로 정의.
- 개체 관계 다이어그램(ERD)에서 개체 집합은 직사각형으로 표시



데이터 모델링

속성(Attribute)

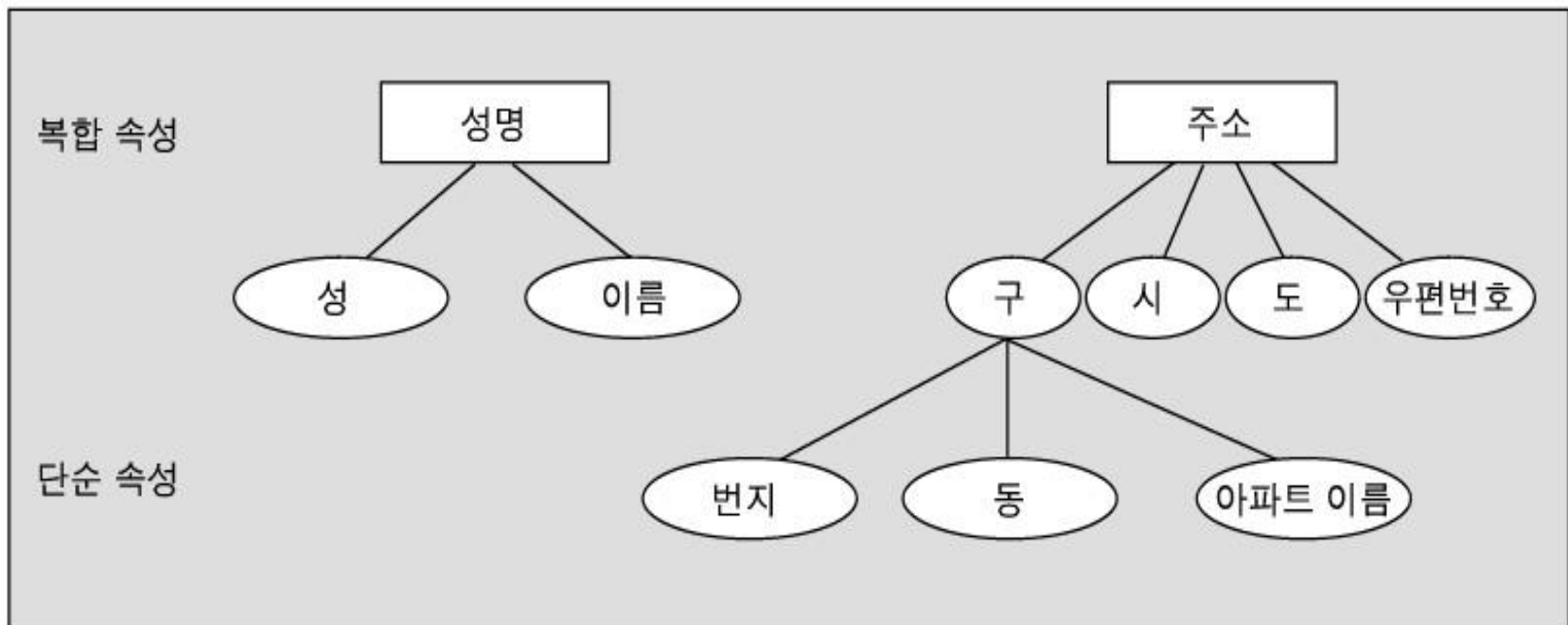
- 개체를 나타내는 특성
- ERD에서 속성은 개체 집합을 나타내는 직사각형에 실선으로 연결된 타원형으로 표현
- 도메인(domain)
 - 각 속성마다 가질 수 있는 값들의 범위



데이터 모델링

복합 속성(Composite Attribute)과 단순 속성(Simple Attribute)

- 단순 속성
더 이상 작은 구성 요소로 분해할 수 없는 속성
- 복합 속성
독립적인 의미를 좀더 기본적인 속성들로 분해할 수 있는 속성



데이터 모델링

- 단일치 속성(single-valued attribute)과 다중치 속성(multi-valued attribute)
 - 단일치 속성(single-valued attribute): 개체의 속성 중 주민등록번호, 또는 학번과 같이 반드시 하나의 값만 존재하는 속성
 - 다중치 속성(multi-valued attribute): 전화번호와 같이 집, 핸드폰, 회사 전화번호와 같이 여러 개의 값을 가질 수 있는 속성.
- 다중치 속성 변경 방법
 1. 개체형 내에서 다중치 속성을 여러 개의 새로운 속성들로 분리
 2. 다중치 속성을 구성하는 속성들로 구성된 새로운 개체형 생성하고 새로운 개체 형과는 다대일 관계 설정.

모델링

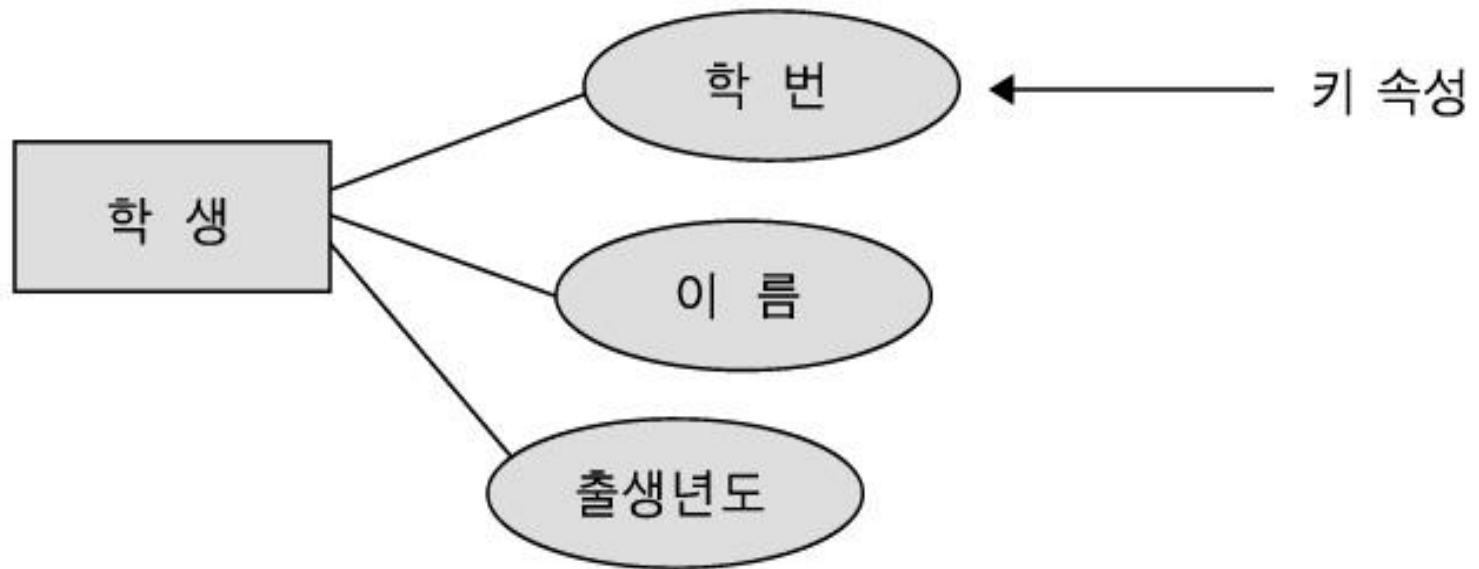
❖ ER Diagram

기호	의미
-----	<ul style="list-style-type: none">• 비식별자 관계(non-identifying relationship): 강한 개체 타입• 부모 개체의 키가 일반 속성으로 포함되는 관계
_____	<ul style="list-style-type: none">• 식별자 관계(identifying relationship): 약한 개체 타입• 부모 개체의 키가 주식별자로 포함되는 관계
———<	<ul style="list-style-type: none">• 일대다(1:N)의 관계: N 쪽에 새발을 표시
———○	<ul style="list-style-type: none">• 0(선택 참여, 최소 참여가 0일 경우
———+	<ul style="list-style-type: none">• 1(필수 참여, 최소 참여가 1일 경우

데이터 모델링

- 개체의 키(Key)

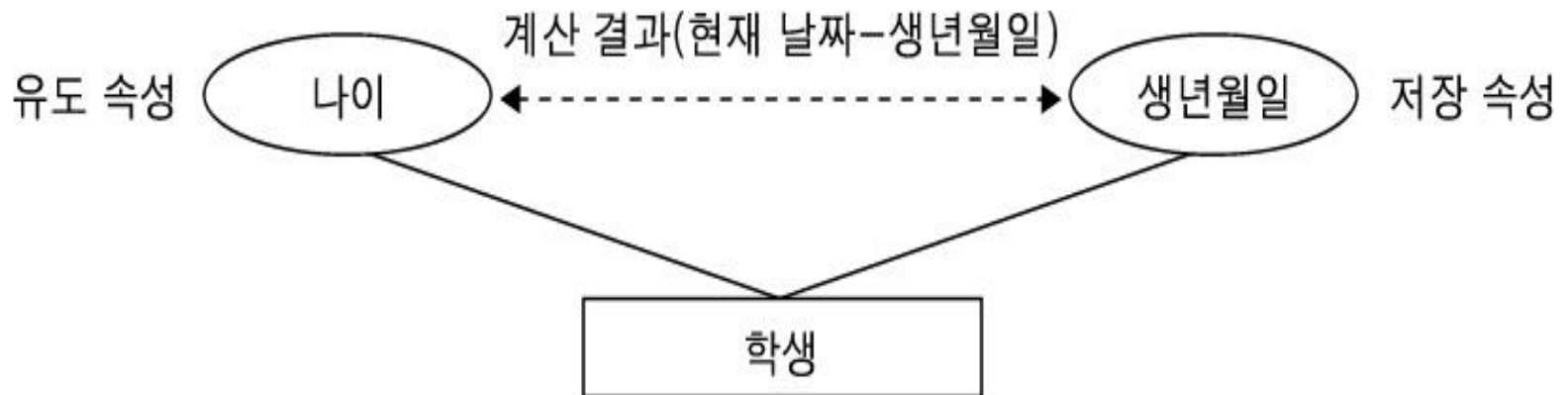
- 개체의 속성 중 하나 또는 그 이상의 속성이 개체를 다른 개체와 구별할 수 있는 속성



데이터 모델링

- 유도 속성(Drived Attribute)과 저장 속성(Stored Attribute)
 - 유도 속성: 속성의 값이 다른 관련된 속성이나 개체가 가지고 있는 값으로부터 유도되어 결정되는 속성
 - 저장 속성: 유도 속성을 결정하기 위해 사용된 속성

ERD에서 유도 속성은 점선으로 표시



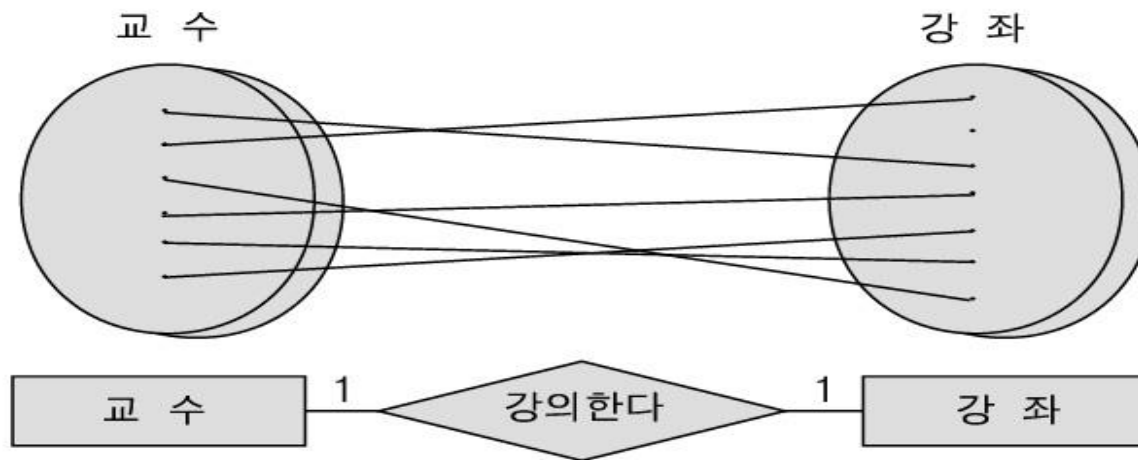
데이터 모델링

관계(Relationship)

- 개체-관계 모델에서 개체 사이의 연관성을 표현하는 개념
- ERD에서 개체들 사이의 관계는 마름모를 사용하여 표현

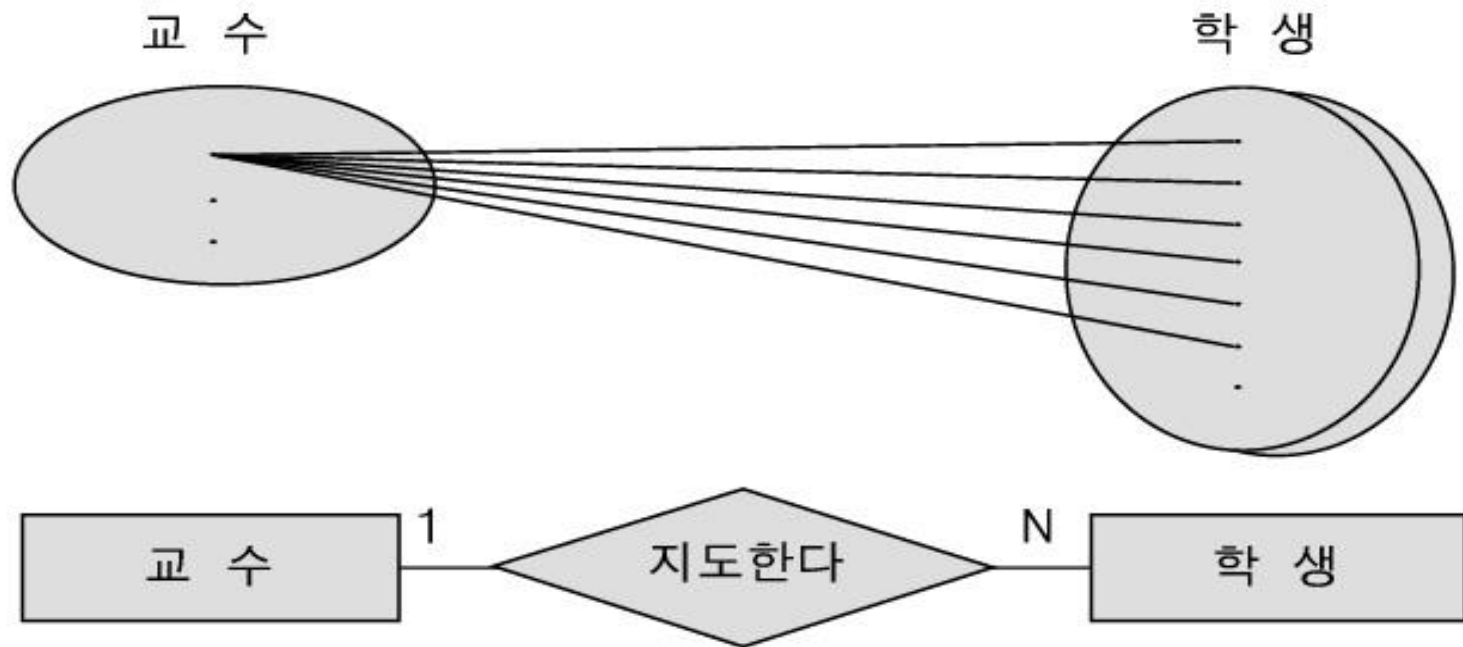


- 관계의 유형
 - 일대일 관계



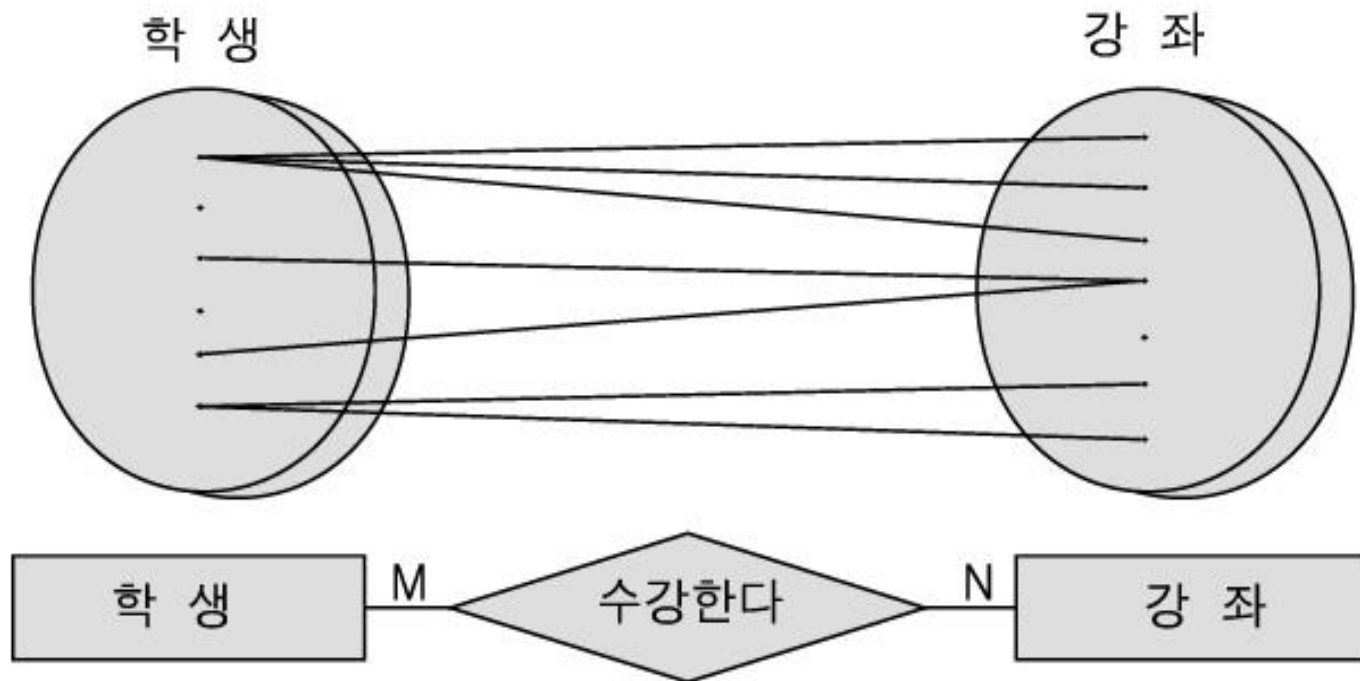
데이터 모델링

일대다 관계



데이터 모델링

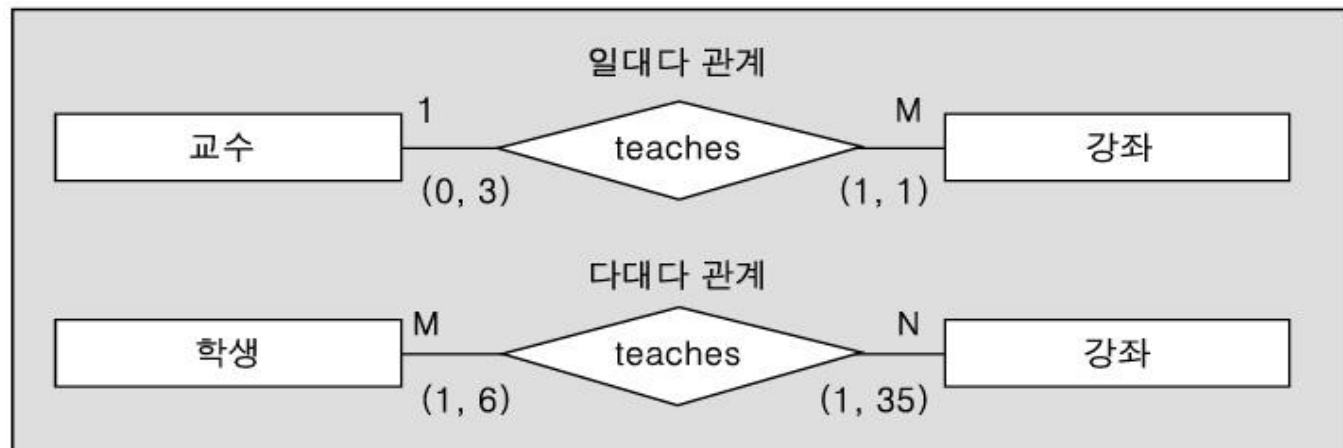
다대다 관계



데이터 모델링

관계의 대응 개체 수(Cardinality, 카디날리티)

- 관계에 참여하는 하나의 개체에 대해 다른 개체 형에서 몇 개의 개체가 참여하는지를 나타내는 것
- 예
 - 한 명의 학생이 1개 이상 6개 이하의 과목에 등록할 수 있다면 대응 개체 수는 (1, 6)
 - 대응 개체 수는 (min, max)의 한 쌍의 값으로 표현하는데 여기서 min은 관계에 참여하는 개체의 최소 개수, max는 관계에 참여하는 최대 개수 의미



데이터 모델링

존재 종속(existence-dependent)

- 한 개체의 존재가 다른 개체(들)의 존재에 영향을 받는 경우 이를 존재 종속 (existence-dependent)이라 한다.

- 예

보험 회사가 “이몽룡”이라는 사원과 그의 부양가족에게 보험 혜택을 준다고 할 때 “사원”, “부양_정보”라는 개체형들을 정의한다고 하자. “이몽룡”에게 “박하늘, 이그름, 이단비”라는 3명의 부양가족이 있다면 부양가족 3명은 “이몽룡”없이 보험 혜택을 받을 수 없다. 다시 말해 부양가족 3명의 정보는 “부양_정보”에 존재하지만 “사원”과 연관지어 지는 경우에만 존재하게 되는데, 이를 존재 종속이라 한다. 만약 “이몽룡”이 직장을 그만 두어 “사원” 테이블에서 삭제되면 부양가족 3명도 함께 “부양_정보”로부터 삭제된다.

사원		부양_정보		
사원_번호	사원_이름	사원_번호	부양가족	부양가족_이름
10111	홍길동	10111	1	최성실
10258	이몽룡	10258	1	박하늘
19658	성춘향	10258	2	이구름
		10258	3	이단비
		19685	1	김가은

데이터 모델링

관계 참여

- 선택적(optional) 관계
 - 개체형의 개체는 관계를 이루는 다른 개체형에서의 개체와 연관이 없어도 되는 관계.
 - 대응 개체 수(min, max)로도 표현하는 경우 min의 값이 0
- 의무적(mandatory) 관계
 - 그 개체형의 모든 개체는 반드시 관계를 이루는 다른 개체형에서의 개체와 연관이 있어야 하는 것 의미
 - 대응 개체 수(min, max)로도 표현하는 경우 min의 값이 1



데이터 모델링

관계 데이터 구조

- 1970년 Codd에 의해 개발
- 관계형 데이터 구조를 구성하는 용어
 - 엔티티(Entity) : 정보 저장의 기본 형태가 2차원 구조의 테이블
 - 속성(attribute) : 테이블의 각 열을 의미
 - 도메인(Domain) : 속성이 가질 수 있는 값들의 집합
 - 튜플(Tuple) : 테이블이 한 행을 구성하는 속성들의 집합



데이터 모델링

엔티티(Entity)

- 정의

- 엔티티는 행(row)과 열(column)로 구성되는 2차원 구조
- 엔티티의 각 행은 하나의 개체를 나타내고, 릴레이션의 각 열은 개체의 각 속성 의미
- 관계형 모델에서는 행은 튜플(tuple), 열은 속성(attribute)이라는 이름 사용

- 특징

- 하나의 엔티티에 있는 튜플들은 모두 상이(distinct)해야 한다.
- 하나의 엔티티에서 튜플들의 순서와 속성들의 순서는 아무런 의미가 없다.
- 하나의 엔티티에서 같은 이름을 가진 속성들이 있을 수 없다
- 각 속성이 가질 수 있는 값들의 범위(도메인,domain)를 벗어나는 값을 가진 튜플들이 존재할 수 없다.
- 행과 열이 교차되는 곳은 원자값(atomic value)으로만 표현된다. 원자값은 논리적으로 더 이상 쪼개질 수 없는 값으로서, 여러 개의 값을 갖는 속성을 직접 표현하는 것이 불가능함을 의미한다.
- 수치형 (Numeric Type), 문자형 (Character Type) , 날짜형 (Date Type), 논리형 (Logical Type)

데이터 모델링

키 (key)

- 엔티티를 구성하는 각 튜플들을 데이터 값들에 의해 유일하게 식별할 수 있는 속성
- 후보키(Candidate Key)
 - 엔티티의 한 속성 집합(K)이 릴레이션이 전체 속성 집합 A의 부분 집합이면서 유일성(uniqueness)과 최소성(minimality)을 만족하는 경우 속성 집합(K)
 - 학번 속성
 - {이름, 학과} 속성

학생

학번	이름	학년	학과
100	홍길동	4	컴퓨터
200	이하늘	3	전기
300	임꺽정	1	컴퓨터
400	송호준	4	컴퓨터
500	박현진	2	음악

데이터 모델링

- 기본 키(Pimary Key)
 - 후보 키 중 하나의 속성만으로 엔티티의 튜플들을 유일하게 식별할 수 있는 키
- 대체키(Alternate Key)
 - 기본 키를 제외한 나머지 후보키(Candidate Key)
- 외래키(Foreign Key)
 - 하나 이상의 테이블을 연결하여 사용하는 경우 필요한 키
 - 한 테이블의 속성들의 집합으로 그 값이 다른 테이블의 주키와 일치하거나 null 값인 키를 의미

데이터 모델링

엔티티의 연결 : 릴레이션

공통된 속성을 공유하여 관계형 데이터베이스 내의 엔티티들을 연결

수강과목

학번	이름	학점	학과	과목번호
100	홍길동	4	컴퓨터	C123
200	이하늘	3	전기	C234
300	임걱정	1	컴퓨터	C236
400	송호준	4	컴퓨터	C156
500	박현진	2	음악	C234

외래 키

과목

과목번호	과목이름	학점	학과	과목이름
C123	컴퓨터개론	3	컴퓨터	임사랑
C234	프로그래밍	3	전기	정파란
C236	자료구조	3	컴퓨터	최하늘
C156	파일처리	3	컴퓨터	최정상
C198	데이터베이스	3	컴퓨터	김홍두

데이터 모델링

도메인(Domain)과 속성(Attribute)

- 속성 값(Attribute Value)
 - 개개의 속성이 가지는 데이터 값
 - 관계형 모델에서 이러한 데이터 값들은 더 이상 분해할 수 없는 원자 값만을 허용
- 도메인 (Domain)
 - 하나의 속성이 가질 수 있는 같은 타입의 모든 원자 값의 각 속성은 하나의 도메인에 대해서만 값을 사용할 수 있다.

제약 조건

- 개체 무결성(Entity Integrity)
 - 엔티티 개체들을 식별할 기본 키로 사용되는 속성이 튜플들을 유일하게 식별할 수 있도록 널 값(null)을 가질 수 없는 성질
- 참조 무결성(Referential Integrity)
 - 엔티티는 참조할 수 없는 외래키의 값을 가져서는 안 된다는 것을 의미.
 - 참조할 수 없는 외래키 값이란 널이 아니면서 참조된 엔티티의 어떤 기본키의 값과도 일치하지 않는 값 의미.

데이터 모델링

카티션 프로덕트(CARTESIAN PRODUCT, X)

- 두 엔티티의 조합 가능한 모든 릴레이션

A
A#
A1
A2
A3

B
B#
B1
B4
B5

(A TIMES B)	
A#	B#
A1	B1
A1	B4
A1	B5
A2	B1
A2	B4
A#	B#
A2	B5
A3	B1
A3	B4
A3	B5

데이터 모델링

선택(SELECT, σ)

- 엔티티로부터 조건에 만족된 튜플들을 선택하는 연산자
- 선택 연산 결과 구성되는 엔티티의 수평 부분 집합(horizontal subset)으로 데이터베이스 조작어의 조건절에 지정된 조건식(predicate)을 만족하는 엔티티 내의 튜플의 집합이 된다.
- $\sigma_{\langle \text{선택조건} \rangle}$ (테이블이름)

학생

학번	이름	학번	학과	점수
100	홍길동	4	컴퓨터	80
200	이하늘	3	전기	90
300	임꺽정	1	컴퓨터	85
400	송호준	4	컴퓨터	70
500	박현진	2	음악	79

$\sigma_{\text{점수} \geq 80}$ (학생)

학번	이름	학번	학과	점수
100	홍길동	4	컴퓨터	80
200	이하늘	3	전기	90
300	임꺽정	1	컴퓨터	85

데이터 모델링

프로젝트(PROJECT, π)

- 프로젝트 연산은 엔티티의 특정 속성만으로 구성된 새로운 릴레이션을 구하기 위한 연산
- 결과 릴레이션은 엔티티를 수직으로 절단한 열(column)의 집합
- 프로젝트는 엔티티의 수직적 부분 집합(vertical subset)
- 프로젝트 연산 결과로 만들어진 엔티티에 똑같은 튜플이 중복되어 존재하는 경우 시스템은 그 중 하나만 제외하고 나머지는 모두 삭제
- π 속성리스트(테이블이름)

학생

학번	이름	학년	학과	점수
100	홍길동	4	컴퓨터	80
200	이하늘	3	전기	90
300	임க்க정	1	컴퓨터	85
400	송호준	4	컴퓨터	70
500	박현진	2	음악	79

π 이름, 학과 (학생)

이름	학과
홍길동	컴퓨터
이하늘	전기
임க்க정	컴퓨터
송호준	컴퓨터
박현진	음악

데이터 모델링

조인(JOIN, \bowtie)

- 두 엔티티와 관련된 튜플을 하나의 튜플로 결합하는 연산
- 카티션 프로덕트(cartisian product) 연산의 결과에서 얻어진 엔티티로부터 조건에 맞는 튜플의 집합을 구하기 위한 연산
- 조인 연산은 연산자를 θ 로 표현하여 일반화하므로 θ 로 표현될 수 있는 조인을 세타 조인(θ -join)이라 한다.
- θ 가 "="인 조인 : 동일 조인 또는 이퀴 조인(equijoin)

학생

학번	이름	학점	학과
100	홍길동	4	컴퓨터
200	이하늘	3	전기
300	임꺽정	1	컴퓨터

성적

학번	과목번호	등급
100	C123	B
100	C234	A
200	C236	B
300	C156	A
300	C234	A

학생 \bowtie 학번=학번 성적

학생 · 학번	학생 · 이름	학생 · 학년	학생 · 학과	수강과목 · 과목 번호	수강과목 · 등급
100	홍길동	4	컴퓨터	C123	B
100	홍길동	4	컴퓨터	C234	A
200	이하늘	3	전기	C236	B
300	임꺽정	1	컴퓨터	C156	A
300	임꺽정	1	컴퓨터	C234	A

데이터 모델링

– θ 가 "N"인 조인 : 자연 조인(Natural Join)

학생 \bowtie_N 등록

학생 · 학번	학생 · 이름	학생 · 학년	학생 · 학과	수강과목 · 과목 번호	수강과목 · 등급
100	홍길동	4	컴퓨터	C123	B
100	홍길동	4	컴퓨터	C234	A
200	이하늘	3	전기	C236	B
300	임꺽정	1	컴퓨터	C156	A
300	임꺽정	1	컴퓨터	C234	A

데이터 모델링

디비전(DIVISION, ÷)

- 두 엔티티에 대해 A, B에 대해 B 엔티티의 튜플에 관련된 모든 튜플들을 A 엔티티로부터 구하는 것

학과목(SC)

학번	과목 번호
100	C123
100	C234
200	C236
200	C234
200	C123
300	C156
300	C234
300	C236
300	C123
400	C156
400	C234

과목_1(C1)

과목 번호
C123

과목_2(C2)

과목 번호
C234

과목_31(C3)

과목 번호
C236
C156

(디비전의 결과)

SC ÷ C1

학번
100
200
300

SC ÷ C2

학번
100
200
400

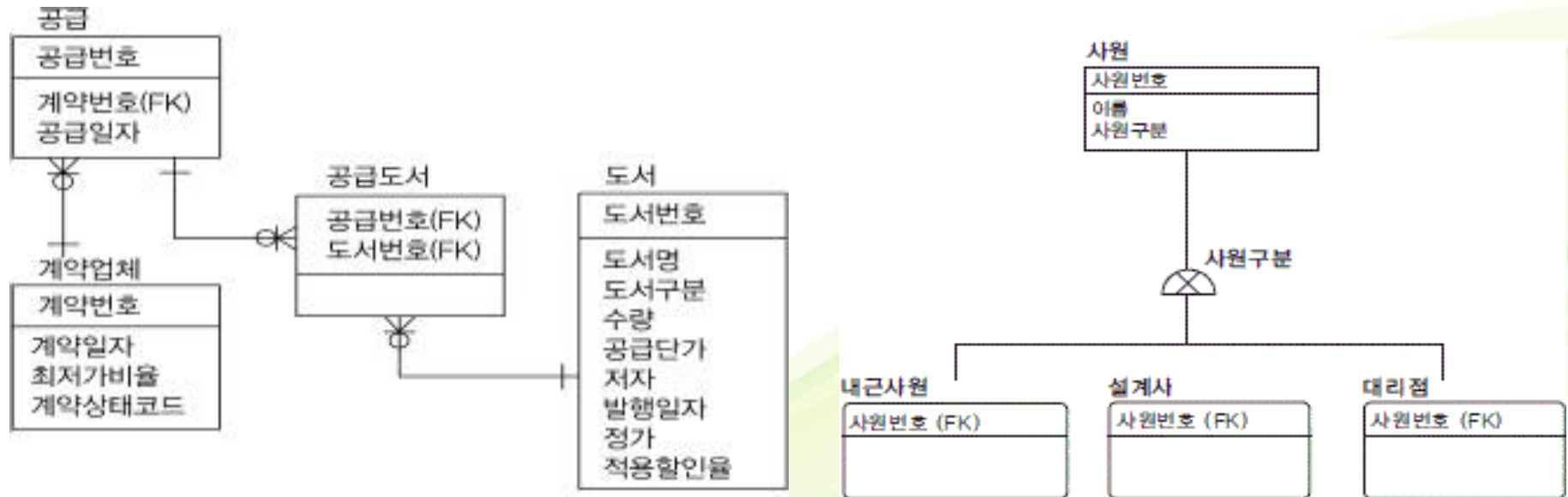
SC ÷ C3

학번
300

모델링

❖ ER Diagram: 정보 공학(IE – 까마귀 발)표기법

- ✓ 제임스 마틴(James Martin)이 주창한 Information Engineering에서 사용하는 Data 모델 표기법
- ✓ 까마귀발 모양과 같은 형태의 관계 표현 때문에 Crow's Foot Model로 불리기도 함
- ✓ 가장 널리 사용되는 표기법의 하나이지만 서브 타입과 같은 개체 집합의 상세 표현에 있어서 공간을 많이 차지하는 단점이 있음



모델링

❖ ER Diagram: 정보 공학(IE – 까마귀 발)표기법

Entities

An **entity** is represented by a rectangle, with its name on the top. The name is singular (entity) rather than plural (entities).



Attributes – describe the characteristics of a particular entity instance. The attribute(s) that uniquely distinguishes an instance of the entity is the identifier. Usually, this type of attribute is marked with an asterisk.



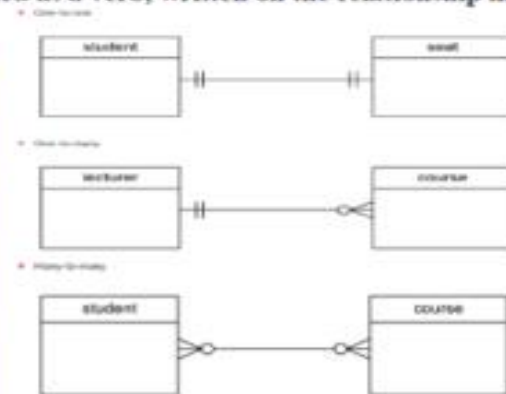
Relationships

They are presented as a straight line. Usually, each relationship has a name, expressed as a verb, written on the relationship line.

Cardinality of the Relationship in ERD of Crow's Foot Notation

Symbol		Meaning
		Mandatory & One (One and Only One)
		Optional & One (Zero or one)
		Mandatory & Many (One or many)
		Optional & Many (Zero or many)

2019.08
min
max
○ Zero
| One
< Many



❖ ER Diagram

✓ ERD 작업 순서

- Entity 를 도출하고 그림
- Entity 를 적절하게 배치
- Entity 간 관계를 설정: 연결
- 관계 이름을 기술
- 관계의 참여도를 기술
- 관계의 필수 여부를 기술

✓ Entity 배치

- 좌에서 우로, 위에서 아래로
- 가장 중요한 Entity(고객, 주문)을 좌측 상단에 배치
- 업무 흐름의 중심이 되는 Entity(출고, 주문 목록, 출고 목록)를 중앙에 배치
- 중심 Entity와 관계 있는 Entity(창고, 고객, 사원, 재고)를 주위에 배치

모델링

❖ ER Diagram









✓ ERD 관계의 연결

- 서로 관련있는 Entity 간의 관계를 설정
- 초기에는 모두 PK 로 속성이 상속되는 식별자 관계를 설정
- 중복 관계, Cycle 관계 등을 유의

✓ ERD 관계 이름의 표시

- 관계 이름은 현재형을 사용
- 지나치게 포괄적인 용어(예, 이다, 가진다 등)은 사용하지 않도록
- 실무에서는 생략해도 무방 - 관계 이름이 없어도 ERD의 흐름을 알 수 있음

✓ ERD 관계 관계 차수와 선택성 표시

관계	선택성	IE 표기법	Barker 표기법
1 : 1	필수		
1 : 1	선택		
1 : n	필수		
1 : n	선택		

모델링

❖ Entity: 실체, 객체 – 정보 시스템의 구조 분석 결과

✓ Entity

- 변별할 수 있는 사물 - Peter Chen (1976)
- 데이터베이스 내에서 변별 가능한 객체 - C.J Date (1986)
- 정보를 저장할 수 있는 어떤 것 - James Martin (1989)
- 정보가 저장될 수 있는 사람, 장소, 물건, 사건 그리고 개념 등 - Thomas Bruce (1992)
- Entity는 데이터의 집합
- Entity는 사람, 장소, 물건, 사건, 개념 등의 명사에 해당
- Entity는 업무 상 관리가 필요한 관심사에 해당
- Entity는 저장이 되기 위한 어떤 것

❖ Entity 와 Instance

- ✓ Entity(객체)가 실제 구현된 예가 Instance(사례, 경우)
- ✓ Entity 는 Instance 의 집합

모델링

❖ Entity 특징

- ✓ 업무에서 필요로 하는 정보: 반드시 해당 업무에서 필요하고 관리하고자 하는 정보

병원시스템	Entity	인사시스템
O	환자	X
X	토익점수	O

- ✓ 식별이 가능해야 함: 유일한 식별자에 의해 식별이 가능해야 함
- ✓ Instance 의 집합
 - 연속적으로 존재하는 Instance 의 집합
 - 2개 이상의 Instance 의 집합
 - 1개의 Instance 로 이루어진 집합은 Entity 가 아님
- ✓ 업무 프로세스에 의해 이용: 업무 프로세스가 반드시 그 Entity를 이용
- ✓ 속성을 포함
 - Entity 에는 반드시 속성(Attributes)이 포함되어야 함
 - 식별자만 존재하고 일반 속성이 전혀 없는 객체는 Entity 가 될 수 없음
 - 단 관계 Entity 의 경우엔 주 식별자 속성만으로도 Entity 로 인정

모델링

❖ Entity 특징

✓ 관계의 존재

- Entity 는 다른 Entity 와 최소 한 개 이상의 관계가 존재하여야 함
- 데이터 모델링에서 관계를 생략하여 표현하는 경우
 - ◆ 시스템 처리시 내부적으로 필요한 Entity: 로그 테이블
 - ◆ 통계를 위한 데이터: 통계 만을 위한 Read Only Table
 - ◆ 코드성 Entity
 - 너무 많은 Entity 들과의 관계로 데이터 모델이 복잡해 짐
 - 일반적으로 코드 테이블에 FK를 설정하지 않는 경우가 대부분

❖ Entity 분류

✓ 유무 형에 따른 분류

- 유형 Entity(Tangible Entity)

- ◆ 물리적 형태가 있고 안정적이며 지속적으로 활용되는 Entity
- ◆ 업무에서 도출되며 Entity를 구분하기가 가장 용이한 Entity
- ◆ 사원, 물품, 강사

- 개념 Entity(Conceptual Entity)

- ◆ 물리적 형태는 존재하지 않고 관리해야 할 개념적 정보로 구분이 되는 Entity
- ◆ 조직, 보험상품

- 사건 Entity(Event Entity)

- ◆ 업무(비즈니스 프로세스)를 수행함에 따라 발생하는 Entity
- ◆ 비교적 발생량이 많으며 각종 통계자료에 이용
- ◆ 주문, 청구, 미납

모델링

❖ Entity 분류

✓ 발생 시점에 따른 분류

- 기본 Entity(Basic Entity, Fundamental Entity, Key Entity)
 - ◆ 업무에 원래 존재하는 정보로서 다른 Entity와 관계에 의해 생성되지 않고 독립적으로 생성 가능
 - ◆ 다른 Entity로부터 주 식별자를 상속받지 않고 자신의 고유 식별자를 가짐
 - ◆ 사원, 부서, 고객, 상품, 자재
- 중심 Entity(Main Entity, Transaction Entity)
 - ◆ 기본 Entity로부터 발생되고 그 업무에 있어서 중요한 역할
 - ◆ 데이터 양이 많이 발생되고 다른 Entity와의 관계를 통해 행위 Entity를 생성
 - ◆ 계약, 사고, 청구, 주문, 매출
- 행위 Entity(Active Entity)
 - ◆ 두개 이상의 Entity로부터 발생되고 자주 내용이 바뀌거나 데이터 양이 증가
 - ◆ 주문 목록, 사원 변경 이력
- 종속 Entity(Dependent Entity): 정규화의 결과로 만들어지는 Entity
- 교차 Entity(Active Entity): 다 대 다 관계 해소 목적의 Entity

❖ Entity 의 명명

- ✓ 가능하면 현업 업무에서 사용하는 용어를 사용
- ✓ 가능하면 약어를 사용하지 않음
- ✓ 가능하면 단수 명사를 사용
- ✓ 모든 Entity 에서 유일하게 이름을 부여
- ✓ Entity 생성 의미대로 이름을 부여

모델링

❖ 속성(Attribute)

✓ 속성(Attribute) 의 사전적 의미

- 사물의 성질, 특징, 또는 본질적인 성질, 그것이 없다면 실체를 생각할 수 없는 것

✓ 데이터 모델링 관점에서 속성(Attribute) 의 정의

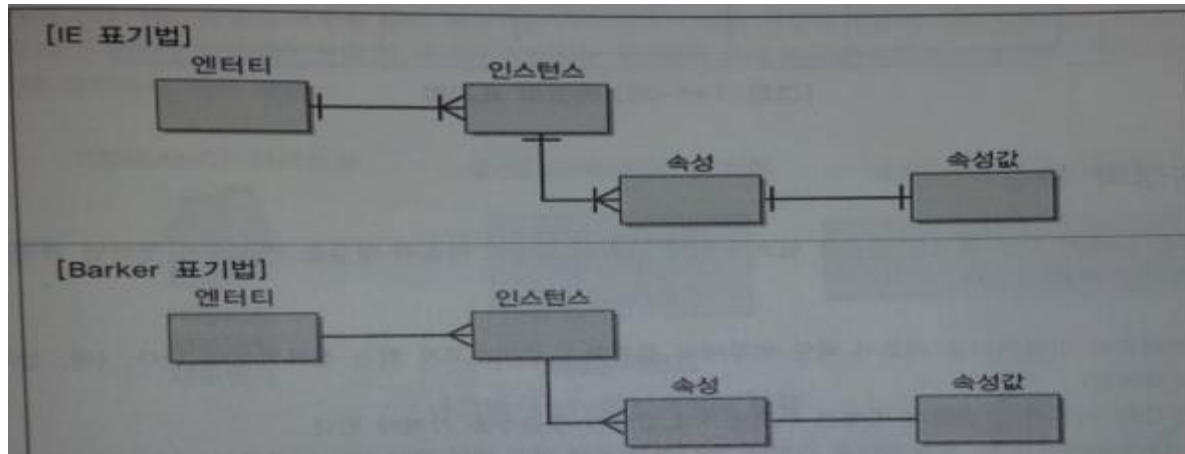
- 업무에서 필요로 하는 인스턴스로 관리하고자 하는 의미상 더이상 분리되지 않는 최소의 데이터 단위
- 업무 상 관리하기 위한 최소의 의미 단위
- Entity가 가지는 항목으로 속성은 Entity를 설명
- 속성은 인스턴스의 구성 요소
- 업무에서 관리되는 정보로 하나의 값만 가져야 하고 주식별자(기본키)에 함수적으로 종속됨

✓ 예시

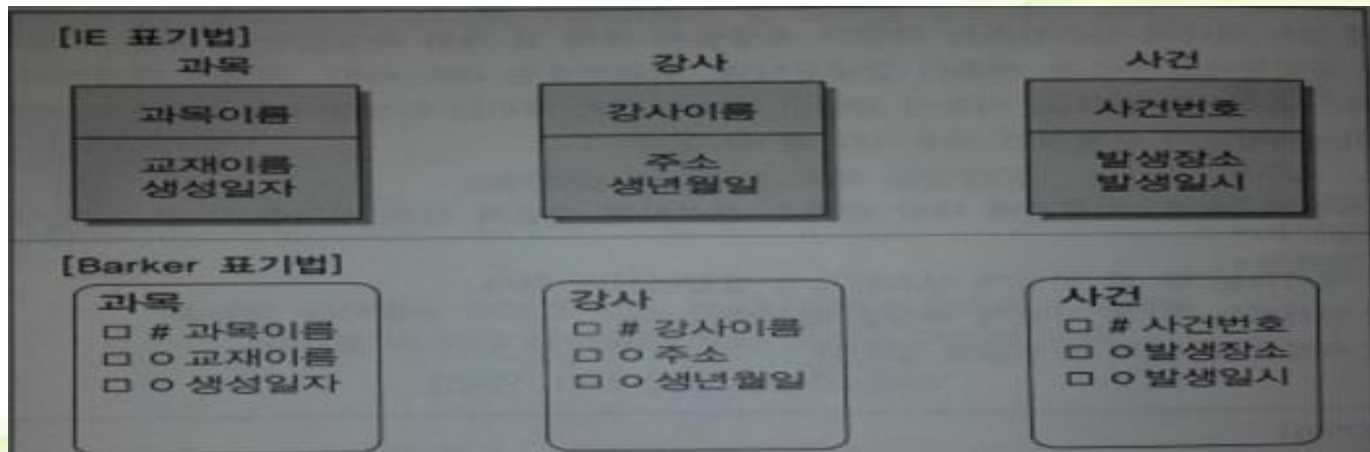
- 생년월일은 그 자체로 의미가 있으므로 속성이라 할 수 있음
- 생년, 생월, 생일 로 분리가 가능하지만 이는 하나의 속성을 관리 목적으로 분리한 것일 뿐 각각을 속성이라 할 수는 없음
- 이름과 주소는 각각 의미있는 속성이지만 이름주소 로 묶는다면 하나의 속성이 두 가지 의미를 가지므로 기본 속성이라 할 수 없음

모델링

- ❖ Entity, 인스턴스와 속성, 속성값에 대한 내용과 표기법
 - ✓ Entity, 인스턴스, 속성, 속성값의 관계



- ✓ 속성의 표기법



❖ 속성의 분류

✓ 속성의 분해 여부에 따른 분류

- 단일 속성: 하나의 의미로 구성된 것으로 아이디나 이름
- 복합 속성: 여러 개의 의미가 있는 것으로 주소 속성이 있음
- 다중 값 속성: 여러 개의 값을 가질 수 있는 것으로 리스트 형태 들인데 이 속성은 다른 Entity로 독립 시켜야 함

모델링

❖ 속성

✓ 속성의 특성에 따른 분류

- 기본 속성: 업무로부터 추출한 모든 속성
- 설계 속성: 코드성 데이터, Entity 식별 용 일련번호
- 파생 속성: 다른 속성에 영향을 받아 발생하는 속성, 계산된 값, 합계, 재고, 잔액
 - ◆ 파생 속성은 그 속성이 가지고 있는 계산 방법에 대해 반드시 어떤 Entity의 어떤 속성에 의해 영향을 받는지 정의가 되어야 함
 - ◆ 타 속성에 의해 지속적으로 영향을 받아 자신의 값이 변하는 성질을 가지고 있는 속성
 - ◆ 파생 속성은 꼭 필요한 경우에만 정의하여 업무 로직이 속성 내부로 스며들지 못하도록 주의해야 함
 - ◆ 파생 속성을 정의한 경우라면 그 값의 정합성을 유지할 수 있도록 해야 함
 - ◆ 통계 관련 Entity, 배치 작업 수행 관련

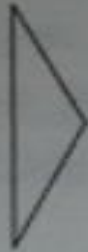
모델링

❖ 속성

✓ Entity 구성 방식에 따른 분류

[IE 표기법]
사원

사원번호(PK)
사원명 우편번호 주소 전화번호 부서코드(FK)



속성분류	속성명
PK속성	사원번호
FK속성	부서코드
일반속성	사원명, 우편번호, 주소, 전화번호

[Barker 표기법]

사원
□ # 사원번호
□ ○ 사원명
□ ○ 우편번호
□ ○ 주소
□ ○ 전화번호
□ ○ 부서코드(FK)



속성분류	속성명
PK속성	사원번호
FK속성	부서코드
일반속성	사원명, 우편번호, 주소, 전화번호

모델링

❖속성

✓도메인

- 속성이 가질 수 있는 값의 범위
 - ◆ 학점은 0.0~4.0 사이의 실수
 - ◆ 주소는 20자리 문자열

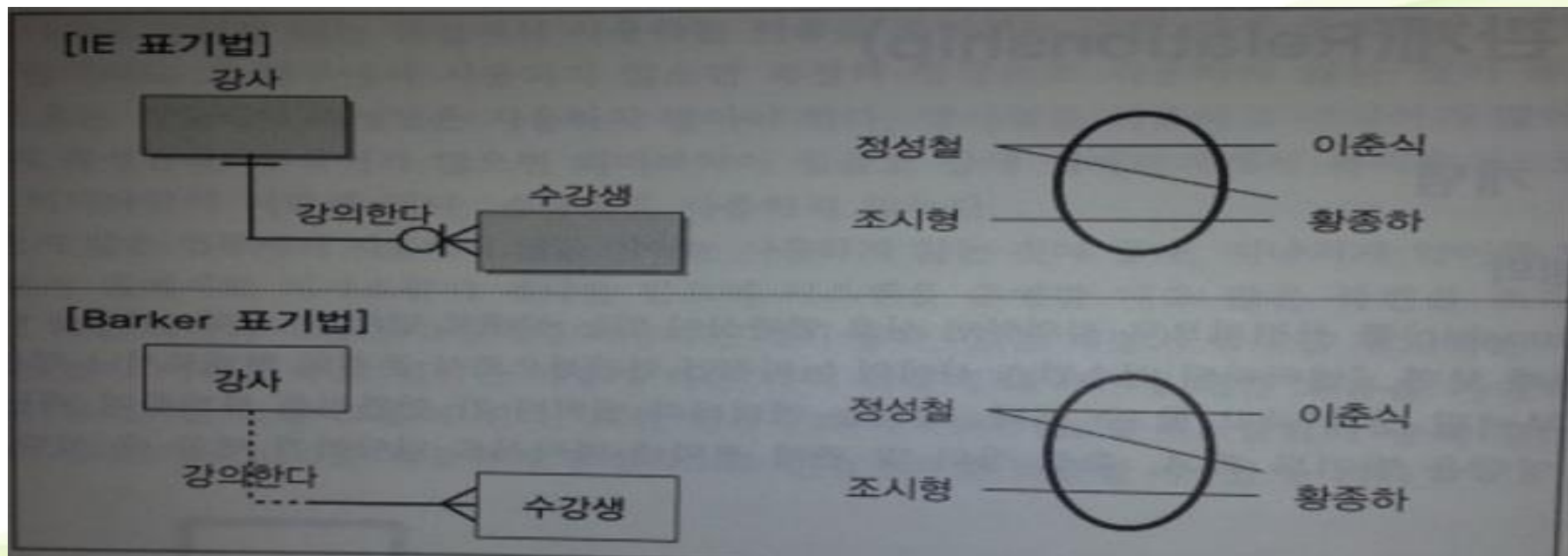
✓속성의 명명(Naming)

- 용어 사전: 속성 이름을 정확하게 부여하고, 용어의 혼란을 없애기 위함
- 도메인 정의: 각 속성이 가지는 값의 범위를 명확하게 하기 위해
- 속성 이름 부여 원칙
 - ◆ 해당 업무에서 사용하는 이름을 부여
 - ◆ 서술식 속성 명은 사용하지 않음
 - ◆ 약어 사용은 가급적 제한
 - ◆ 전체 데이터 모델에서 유일성 확보하는 것이 좋음

모델링

❖ 관계

- ✓ 관계의 정의: 인스턴스 사이의 논리적 연관성으로서 존재 또는 행위로서 서로에게 연관성이 부여된 상태
- ✓ 관계의 페어링(Relationship Paring)
 - Relationship 은 Entity 안의 Instance 가 개별적으로 관계를 가지는 것 (Paring)이고 이것의 집합을 관계로 표현한다는 것
 - 개별 인스턴스가 각각 다른 종류의 관계를 가지고 있다면 두 Entity 사이에 2개 이상의 관계가 형성 될 수 있음



모델링

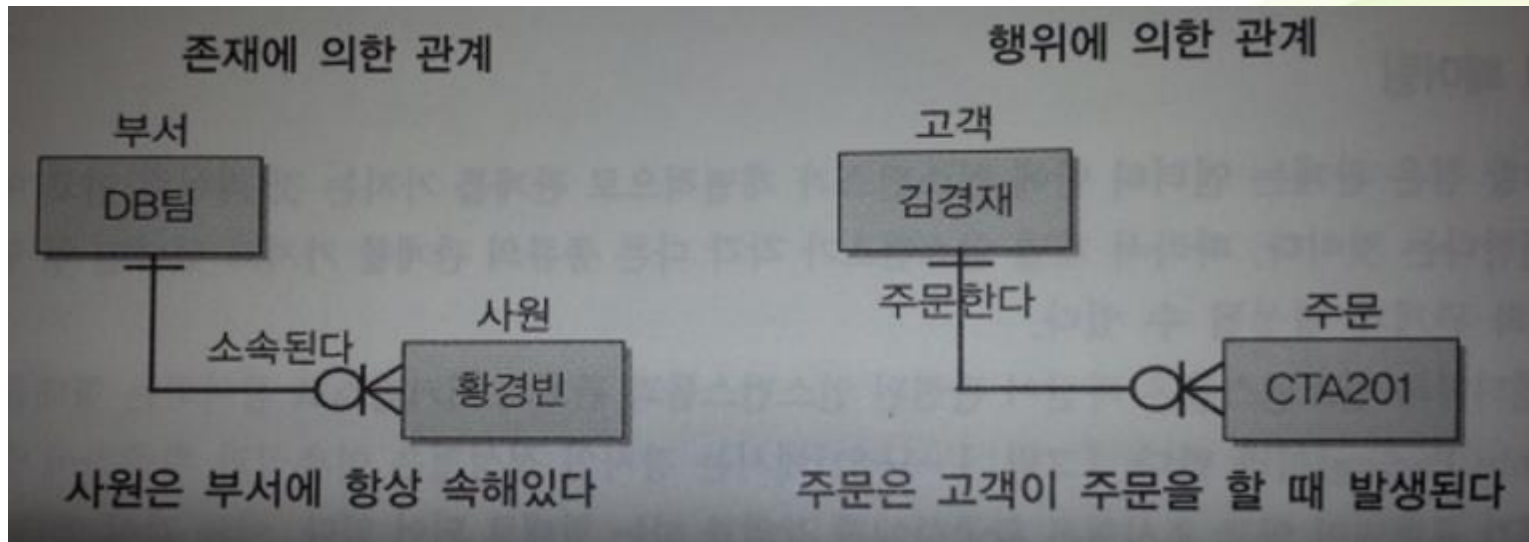
❖ 관계의 분류

✓ 존재 관계

- Entity 간의 상태를 의미
- 학과와 학생처럼 학생이 아무런 행위를 하지 않아도 특정 학과에 소속되는 경우

✓ 행위 관계

- Entity 간의 행위가 있는 것
- 특정 행위를 해야만 만들어지는 관계



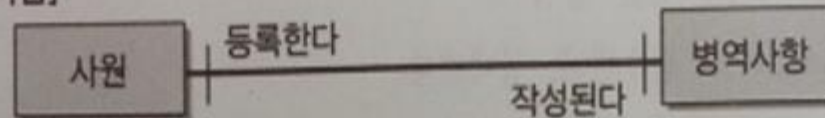
모델링

❖ 관계의 표기법

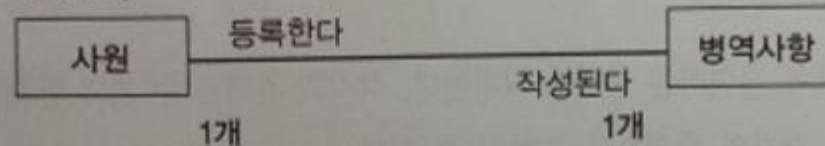
- ✓ 관계 이름 과 관계 차수 - 1:1 관계
 - 필수적 관계와 선택적 관계로 구분할 수 있음

1) 1:1(ONE TO ONE) 관계를 표시하는 방법

[IE 표기법]



[Barker 표기법]



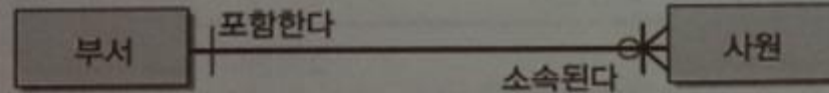
모델링

❖ 관계의 표기법

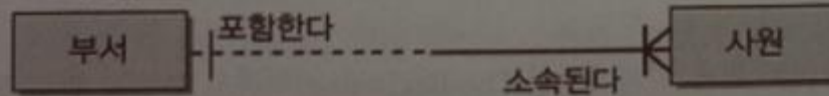
✓ 관계 이름 과 관계 차수 - 1:N 관계

2) 1:M(ONE TO MANY) 관계를 표시하는 방법

[IE 표기법]



[Barker 표기법]



한 명의 사원은 한 부서에 소속되고 한 부서에는 여러 사원을 포함한다

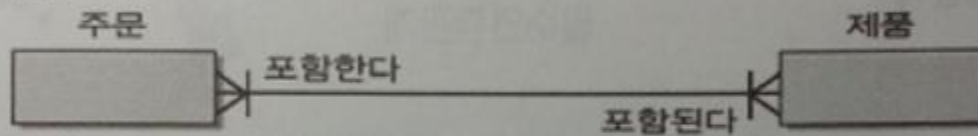
모델링

❖ 관계의 표기법

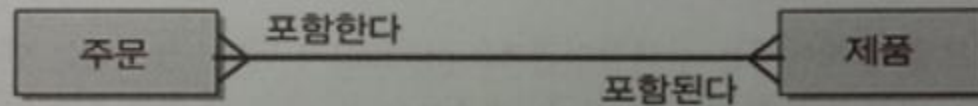
✓ 관계 이름 과 관계 차수

3) M:M(MANY TO MANY) 관계를 표시하는 방법

[IE 표기법]



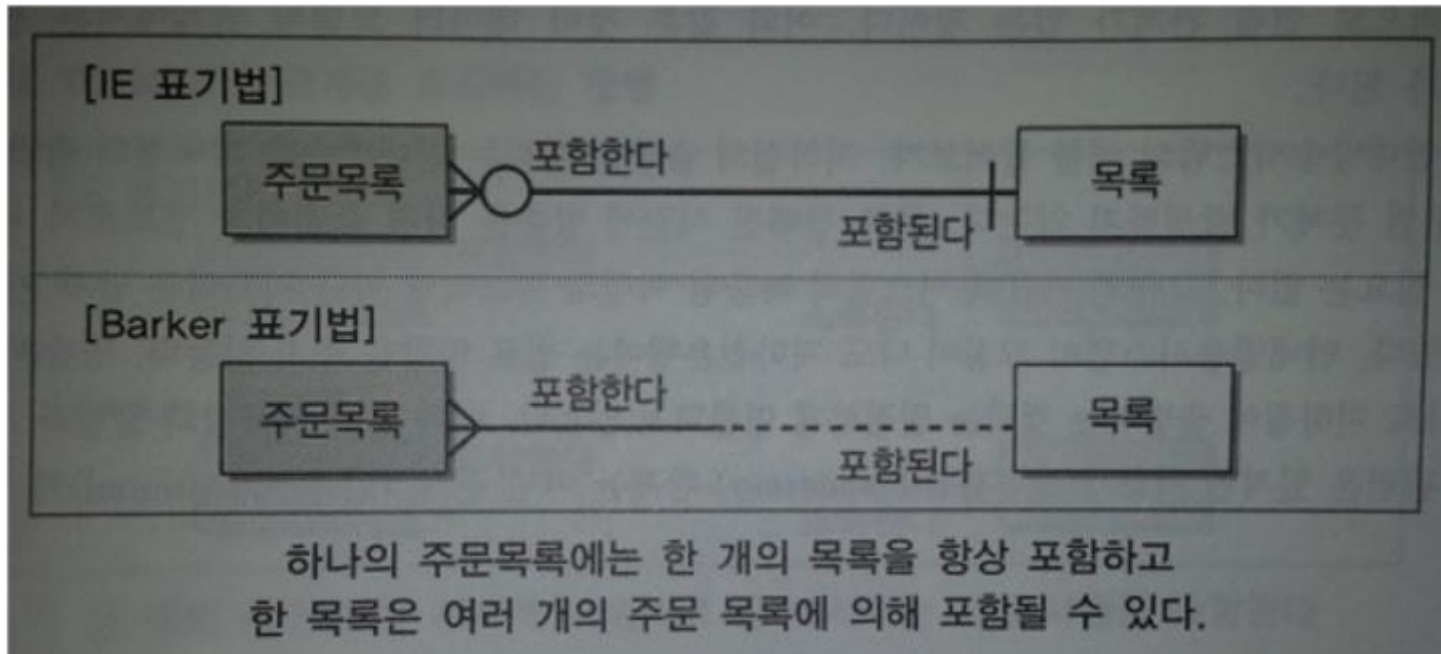
[Barker 표기법]



모델링

❖ 관계의 표기법

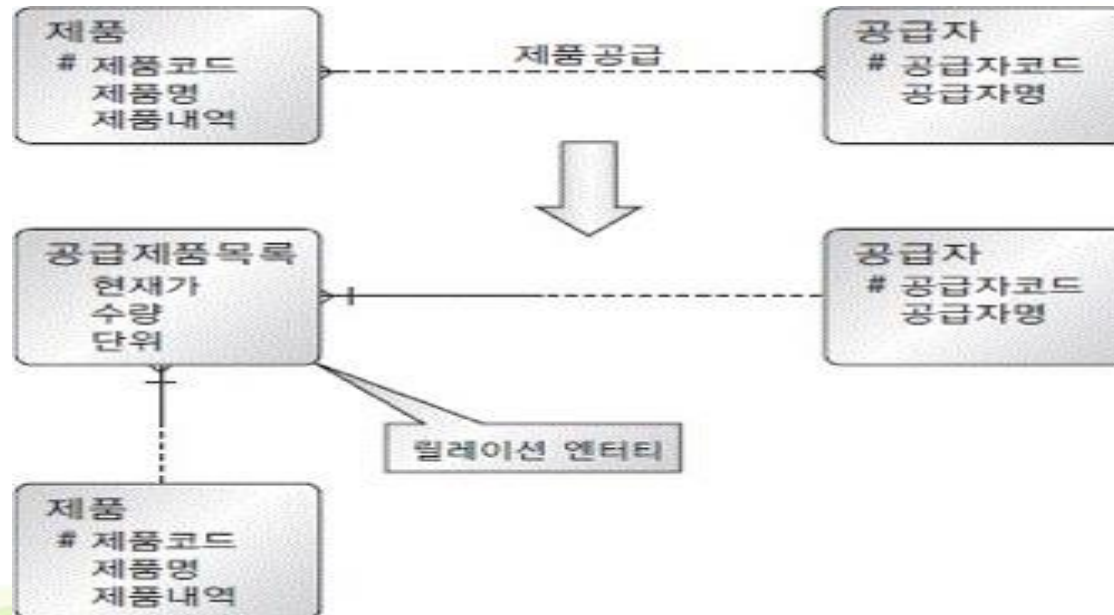
- ✓ 관계 선택 사양(Optionality): 필수 관계, 선택 관계



모델링

❖ M:M 관계

- ✓ M:M 관계는 논리 데이터 모델링 과정에서 많이 나타나며 최종적으로 완성된 데이터 모델에는 존재할 수 없는 형태라고 할 수 있음
- ✓ 제품 Entity와 공급자 Entity 간에는 M:M 관계가 존재하는데 이런 경우는 1:M 관계 2개로 분할해야 함
- ✓ 이 관계가 해소된 결과로 공급제품목록이라는 새로운 Entity가 만들어지고 또한 각각의 두 Entity 즉 제품, 공급자와 1:M의 관계를 부모로서 가지게 됨



모델링

❖ 함수적 종속(Functional Dependency)

- ✓ 함수적 종속이란 어떤 릴레이션 R이 있을때 X와 Y를 각각 속성의 부분집합이라고 가정하고 여기서 X의 값을 알면 Y의 값을 바로 식별할 수 있고 X의 값에 Y의 값이 달라질 때 Y는 X에 함수적 종속되었다고 하고 X를 결정자 그리고 Y를 종속자라고 함
- ✓ 기호로 표현하면 $X \rightarrow Y$
- ✓ 함수적 종속관계의 종류
 - 완전 함수적 종속
 - 부분 함수적 종속
 - 이행적 함수 종속

모델링

❖ 함수적 종속(Functional Dependency)

- ✓ 학번을 알면 이름, 나이, 성별 속성을 식별할 수 있으며 학번이 다르면 그에 따른 값도 다름
- ✓ 이름, 나이, 성별 속성은 학번에 함수적 종속 관계
- ✓ 전공 속성 또한 전공코드에 함수적 종속 관계
- ✓ 학번→이름, 학번→나이, 학번→성별 형태로 표현

학번	이름	나이	성별	전공코드	전공명
110011	박지현	26	여성	AAA1	국문학과
110011	박지현	26	여성	C0B7	컴퓨터공학과
131001	김민석	25	남성	C0A5	전기전자공학과
120006	홍현희	25	여성	B1027	무용과
150705	한태민	23	남성	C0A5	전기전자공학과
171024	설화영	22	여성	B01K2	공예과

모델링

❖ 함수적 종속(Functional Dependency)

- ✓ 완전 함수적 종속(Full Functional Dependency): 기본키가 여러 속성으로 구성되어 있을 경우 기본키를 구성하는 모든 속성이 포함된 기본키의 집합에 종속된 경우로 기본키가 하나의 속성으로 구성된 경우는 모든 속성이 기본키에 대하여 무조건 완전 함수적 종속
 - 이 릴레이션에서는 기본키가 회원번호 속성으로 구성되어 있는데 여기서 이름, 나이, 거주지역 속성은 기본키인 회원번호를 알아야 식별 가능하기 때문에 이름, 나이, 거주지역은 회원번호에 완전 함수 종속

<u>회원번호</u>	이름	나이	거주지역
A001	송민지	17	서울
A002	박아람	15	부산
A003	이예은	16	대전

모델링

❖ 함수적 종속(Functional Dependency)

- ✓ 완전 함수적 종속(Full Functional Dependency): 기본키가 여러 속성으로 구성되어 있을 경우 기본키를 구성하는 모든 속성이 포함된 기본키의 집합에 종속된 경우로 기본키가 하나의 속성으로 구성된 경우는 모든 속성이 기본키에 대하여 무조건 완전 함수적 종속
 - 해당 릴레이션의 기본키는 고객ID와 상품코드 속성으로 구성되어 있는데 수량 속성은 기본키를 구성하는 고객ID, 상품코드 속성을 모두 알아야 식별할 수 있기 때문에 수량은 기본키에 대하여 완전 함수 종속

<u>고객ID</u>	<u>상품코드</u>	주문상품	수량	가격
AAAA01	T001	티셔츠	2	12000
AAAA01	B110	청바지	1	11000
AAAA02	B110	청바지	2	22000
AAAA03	T091	와이셔츠	1	15000
AAAA03	O100	원피스	1	19000

모델링

❖ 함수적 종속(Functional Dependency)

- ✓ 부분 함수적 종속(Partial Functional Dependency): 기본키가 여러 속성으로 구성되어 있을 경우 기본키를 구성하는 속성 중 일부에 종속되는 경우
 - 기본키가 고객ID 와 상품코드 속성으로 구성된 위의 릴레이션에서 주문 상품 은 기본키 중 상품코드 만 알아도 식별할 수 있는데 주문상품 속성 은 기본키에 부분 함수 종속된 관계

<u>고객ID</u>	<u>제품코드</u>	주문상품	수량	가격
AAAA01	T001	티셔츠	2	12000
AAAA01	B110	청바지	1	11000
AAAA02	B110	청바지	2	22000
AAAA03	T091	와이셔츠	1	15000
AAAA03	O100	원피스	1	19000

모델링

❖ 함수적 종속(Functional Dependency)

- ✓ 이행적 함수 종속(Transitive Functional Dependency): 릴레이션에서 X, Y, Z 라는 3개의 속성이 있을 때 $X \rightarrow Y, Y \rightarrow Z$ 이란 종속 관계가 있을 경우, $X \rightarrow Z$ 가 성립될 때 이행적 함수 종속이라고 하는데 X 를 알면 Y 를 알고 그를 통해 Z 를 알 수 있는 경우
 - 회원번호를 알면 생년월일을 알 수 있고 생년월일을 알면 나이를 알 수 있으면 회원번호를 알면 나이를 알 수 있는데 이러한 관계가 이행적 함수 종속

<u>회원번호</u>	생년월일	나이	주소
A001	1995.1.3	26	군포 산본
A002	1996.1.16	25	강남 청담
A003	1997.2.11	24	오uckland
A004	1997.3.27	23	부리람

모델링

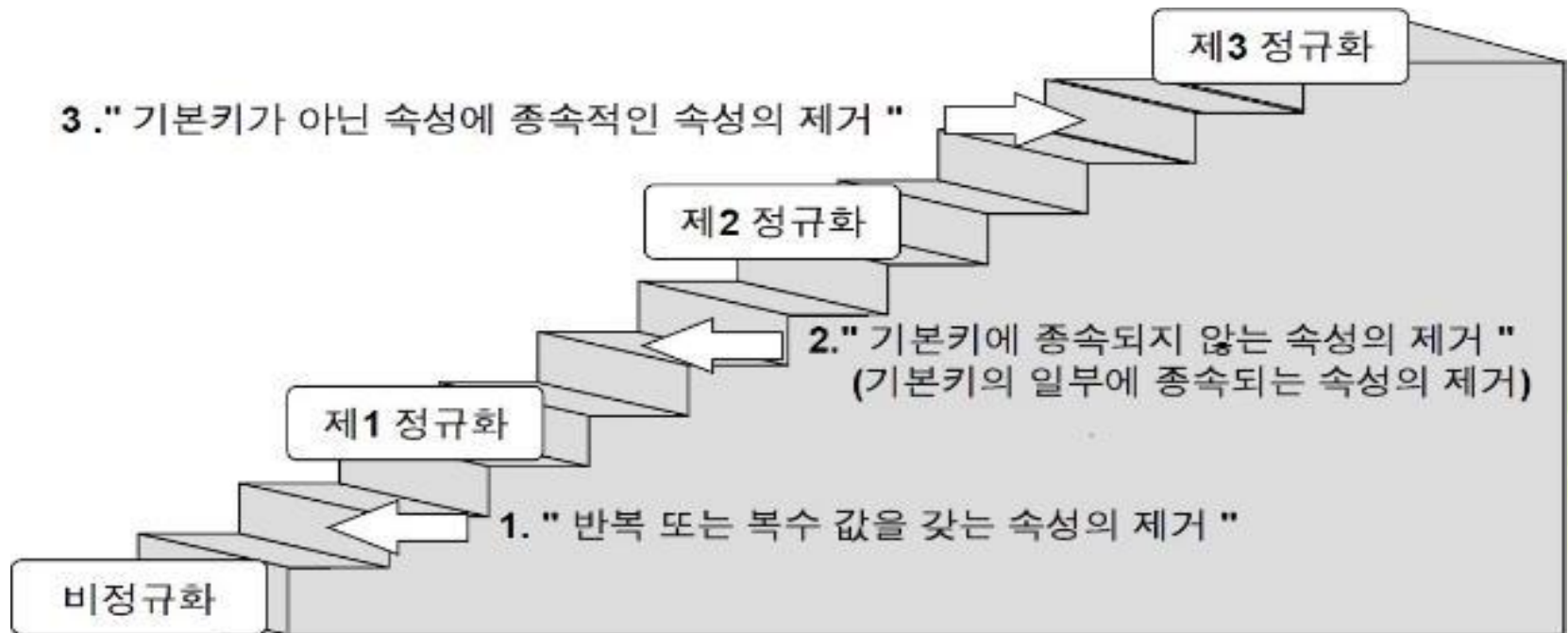
❖ 정규화(Normalization)

- ✓ 논리 데이터 모델을 일관성이 있고 안정성 있는 자료 구조로 만드는 단계
- ✓ 데이터의 일관성, 최소한의 데이터 중복, 최대한의 데이터 유연성을 위한 방법으로 데이터를 분해하는 과정
- ✓ 데이터 모델의 독립성을 확보하기 위한 방법
- ✓ 정규화를 수행하면 비즈니스의 변경에 유연하게 대처해서 데이터 모델의 변경을 최소화할 수 있음
- ✓ 정규화의 장점
 - 중복값이 줄어듦
 - NULL 값이 줄어듦
 - 복잡한 코드로 데이터 모델을 보완할 필요가 없음
 - 새로운 요구 사항의 발견 과정을 도움
 - 업무 규칙의 정밀한 포착을 보증
 - 데이터 구조의 안정성을 최대화

모델링

❖ 정규화

- ✓ 정규화는 1차 정규화부터 BCNF(Boyce-Codd Normal Form)을 포함한 5차 정규화 까지로 구성
- ✓ 일반적으로 중복이 최소로 발생하는 제 3 정규화 까지 진행



❖ 정규화

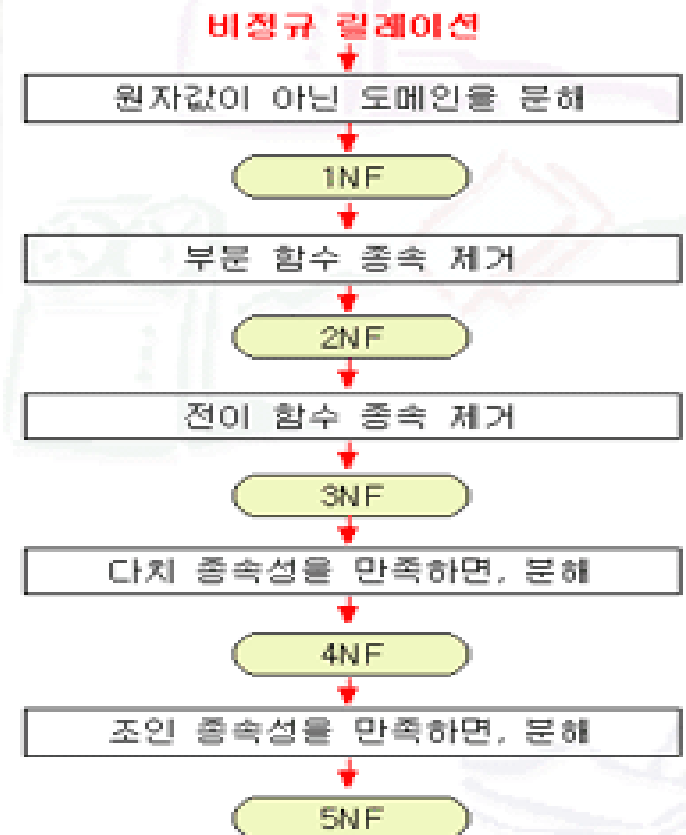
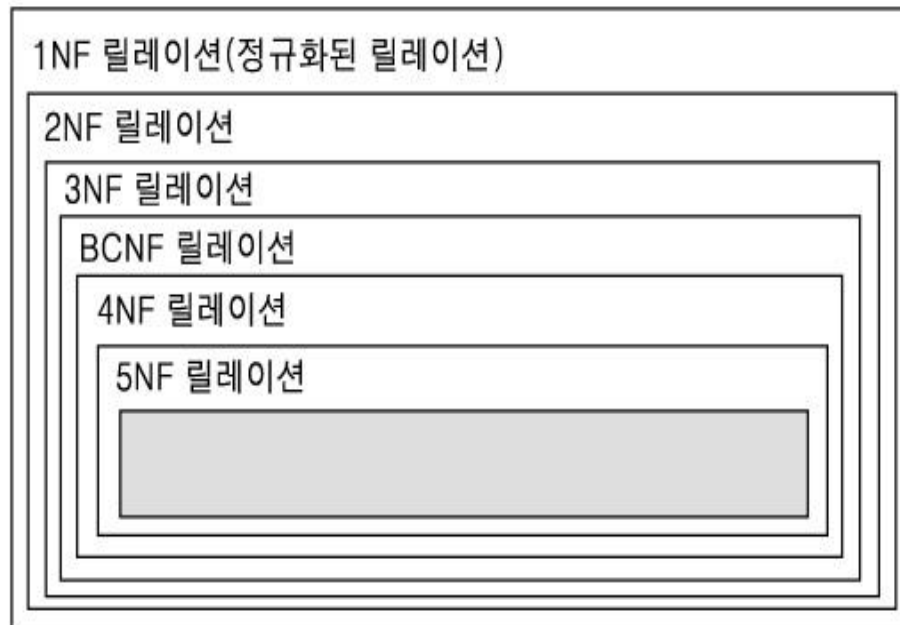
- ✓ 제 1 정규형(1NF, First Normal Form): 한 릴레이션을 구성하는 모든 도메인이 원자값으로 구성
- ✓ 제 2 정규형(2NF, Second Normal Form): 제 1 정규형을 만족하면서 릴레이션에 존재하는 부분 함수적 종속을 제거하여 모든 속성이 기본키에 완전 함수 종속이 되도록 만들어진 정규형
- ✓ 제 3 정규형(3NF, Third Normal Form): 제 2 정규형을 만족하면서 릴레이션을 구성하는 속성들 간의 이행적 종속관계를 분해하여 속성들이 비이행적 함수 종속관계를 만족하도록 만들어진 정규형
- ✓ 보이스-코드(BCNF, Boyce-Codd Normal Form): 제 3 정규형을 만족하면서 릴레이션의 모든 결정자가 후보키가 되도록 하는 정규형
- ✓ 제 4 정규형(4NF, Fourth Normal Form): BCNF를 만족하면서 릴레이션에서 다치 종속 관계를 제거한 정규형
- ✓ 제 5 정규형(5NF, Fifth Normal Form): 후보키를 통하지 않은 조인종속(Join Dependency)을 제거한 정규형

정규화

정규화 형태

- 엔티티의 정규화는 실제 데이터 값이 아니라 개념적인 측면에서 다루어져야 함
- 실제 정규화 과정은 정규형의 순서와 다를 수 있음

전체 릴레이션(정규화 또는 비정규화된)



정규화

제1정규형(1NF) : 반복그룹 제거

- 엔티티에서 속성의 값은 속성의 도메인에 속하는 단일 값(원자 값, atomic Value)이어야 한다는 제약

학번	이름	학과	동아리
20013426	박하늘	컴퓨터학과	{영어회화반, 검도부}
20025914	홍길동	영문학과	{수화반, 합창반}
20038540	홍길순	음악학과	미술반
99590264	이몽룡	사회복지학과	검도부
97456123	최푸름	국어국문학과	축구부

- 반복되는 값이나 여러 값을 갖는 다치 애트리뷰트를 제거해서 각 애트리뷰트가 반드시 하나의 값만을 갖도록 하는 과정



학번	이름	학과	동아리
20013426	박하늘	컴퓨터학과	영어회화반
20013426	박하늘	컴퓨터학과	검도부
20025914	홍길동	영문학과	수화반
20025914	홍길동	영문학과	합창반
20038540	홍길순	음악학과	미술반
99590264	이몽룡	사회복지학과	검도부
97456123	최푸름	국어국문학과	축구부

정규화

• 키가 되는 속성위주로 정리

학번	이름	학과	동아리
20013426	박하늘	컴퓨터학과	{영어회화반, 검도부}
20025914	홍길동	영문학과	{수화반, 합창반}
20038540	홍길순	음악학과	미술반
99590264	이몽룡	사회복지학과	검도부
97456123	최푸름	국어국문과	축구부



학번	동아리	이름	학과
20013426	영어회화반	박하늘	컴퓨터학과
20013426	검도부	박하늘	컴퓨터학과
20025914	수화반	홍길동	영문학과
20025914	합창반	홍길동	영문학과
20038540	미술반	홍길순	음악학과
99590264	검도부	이몽룡	사회복지학과
97456123	축구부	최푸름	국어국문과

모델링

❖ 정규화

✓ 제 1 정규형(1NF, First Normal Form)

다가속성(Multivalued Attributes)

- 같은 종류의 값을 여러 개 가지는 속성을 다가 속성이라 함

#고객ID	고객이름	주민등록번호	전화번호
Blues	홍길동	123456-7890123	123-4567, 234-5678, 345-6789
Pupils	김길동	234567-8901234	456-7890, 567-8901

- 위 그림의 모델은 하나의 속성에 여러 전화번호를 관리하고 있으므로 속성은 반드시 하나의 값을 가져야한다는 1정규형에 어긋남
- 위와 같이 다가 속성이 존재하면 새로운 릴레이션이 필요함
- 위와 같은 릴레이션을 1정규화하면 아래와 같음

[고객]

#고객ID	고객이름	주민등록번호
Blues	홍길동	123456-7890123
Pupils	김길동	234567-8901234

[고객전화번호]

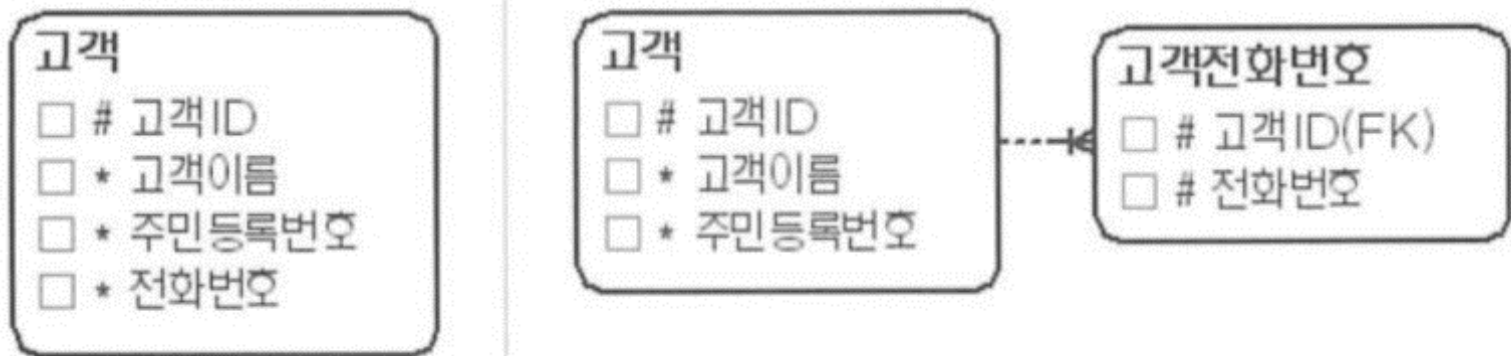
#고객ID	#전화번호
Blues	123-4567
Blues	234-5678
Blues	345-6789
Pupils	456-7890
Pupils	567-8901

모델링

❖ 정규화

✓ 제 1 정규형(1NF, First Normal Form)

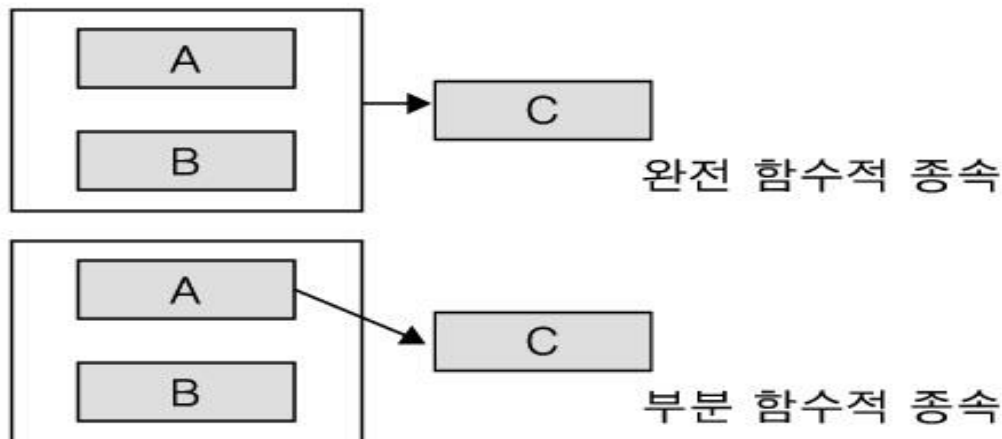
- 이를 모델로 표현하면 아래의 오른쪽 모델과 같음



정규화

제 2정규형(2NF) ; 완전 기능 종속

- 완전 함수적 종속성(full functional dependency)의 개념에 기반을 둔 것으로, 엔티티 R이 1NF이고 키가 아닌 모든 속성이 기본키에 완전 함수적 종속이면 엔티티 R은 2NF임.
- 완전 함수적 종속성
 - 두 속성 A와 B 사이에 $A \rightarrow B$ 의 함수적 종속성이 존재할 때, B가 A의 부분집합 A'에 함수적으로 종속되지 않는 것 의미
 - 즉, $A' \rightarrow B$ 가 성립되지 않아야 하며, 만약 $A' \rightarrow B$ 가 성립하면 부분종속(partial dependency)이라고 부른다.



정규화

공장	라인	생산품	전화		공장	전화		공장	라인	생산품
제1	A	라면	111-1111		제1	111-1111		제1	A	라면
제1	B	소주	111-1111		제2	222-2222		제1	B	소주
제1	C	맥주	111-1111					제1	C	맥주
제2	A	건포도	222-2222					제2	A	건포도
제2	B	오징어	222-2222					제2	B	오징어
키를 공장 + 라인										

정규화

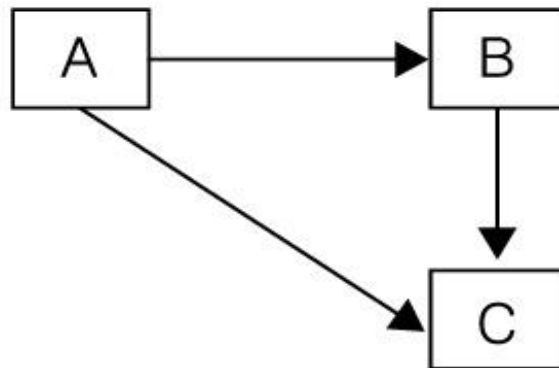
제 3정규형(3NF) : 전이 종속성

- 특징

- 추이적 종속성(transitive dependency)의 개념에 기반을 둔 정규형
- 키가 아닌 속성 값을 갱신하는 경우 불필요한 부작용(이상) 발생 없음
- 모든 이진 엔티티(2NF)는 3NF에 속함

- 추이적 종속성

- 세 속성 사이에 존재하는 함수적 종속성
- A, B, C가 한 테이블 내의 세 속성이고 $A \rightarrow B$ 와 $B \rightarrow C$ 의 함수적 종속성이 존재하면, 함수적 종속성 $A \rightarrow C$ 가 성립하는 것 의미



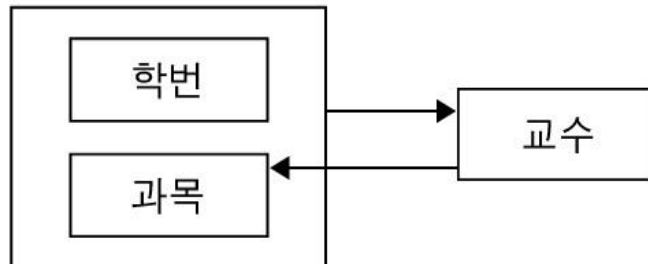
정규화

학번	학과명	전화번호		학번	학과명		컴공	1111-1111
111	컴공과	1111-1111		111	컴공과		수학	2222-2222
222	컴공과	1111-1111		222	컴공과			
333	수학과	2222-2222		333	수학과			
444	수학과	2222-2222		444	수학과			

정규화

BCNF

- 기본 키가 둘 이상의 속성으로 구성된 합성키이고, 합성키가 둘 이상 존재할 경우에 발생하는 이상현상을 방지하기 위해서 고안
- 모든 결정자가 후보키일 경우 엔티티는 BCNF형
- 엔티티 R이 BCNF에 속하면 R은 제1, 제2, 제3 정규형에 속함
- 강한 제3정규형(strong 3NF)
- 예(3NF) : 과목 엔티티
 - 과목 (학번,과목,교수)
 - 후보키 : (학번,과목), (학번,교수)
 - 기본키 : (학번,과목)
 - 함수종속 : (학번,과목) \rightarrow 교수 , 교수 \rightarrow 과목



학번	과목	교수
100	프로그래밍	P1
100	자료구조	P2
200	프로그래밍	P1
200	자료구조	P3
300	자료구조	P3
300	프로그래밍	P4

정규화

3NF(수강과목 릴레이션)에서의 이상

삽입이상: 교수 P5가 자료구조를 담당한다는 사실의 삽입은 수강 학생이 있어야 가능

삭제이상: 100번 학생이 자료구조를 취소하여 튜플을 삭제하면 P2가 담당 교수라는 정보도 삭제됨

갱신이상: P1이 프로그래밍 대신 자료구조를 담당하게 되면 P1이 나타난 모든 튜플을 변경하여야 함

⇒ 원인 : 교수가 결정자이나 후보키가 아님

⇒ 해결 : 과목 ⇒ 교수, 지도과목 엔티티로 분해

수강

학번	교수
100	P1
100	P2
200	P1
200	P3
300	P3
300	P4

지도과목

교수	과목
P1	프로그래밍
P2	자료구조
P3	자료구조
P4	프로그래밍

정규화

제 4정규형 : 조건 의존성

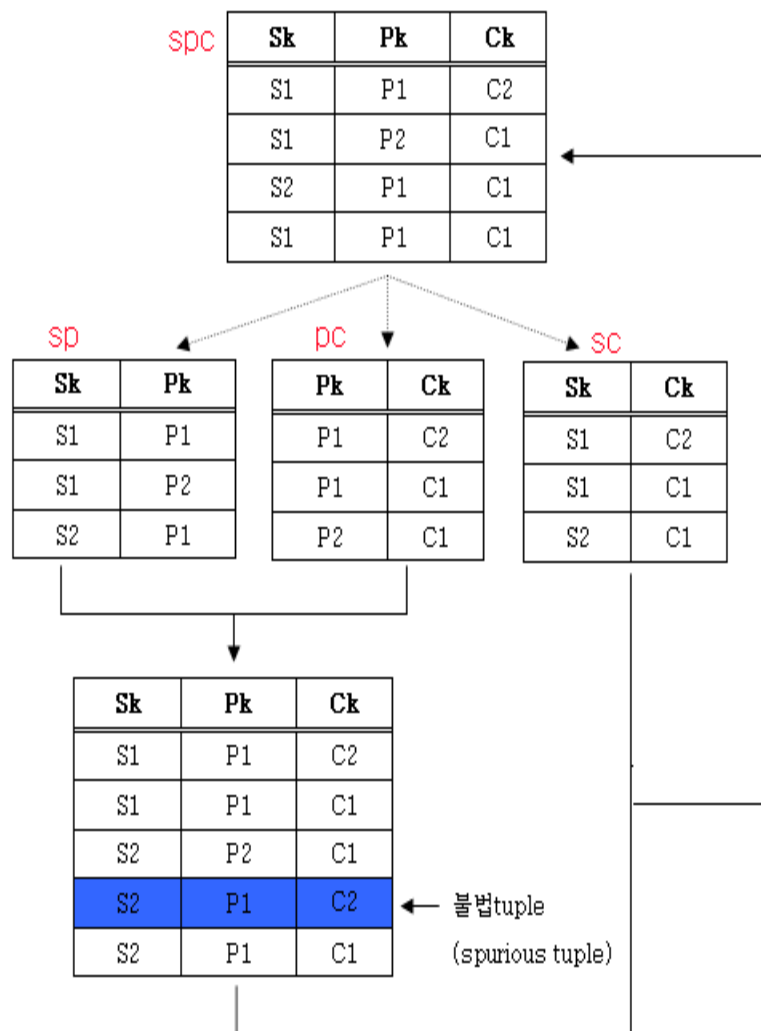
- 의미

- 엔티티 R에서 A B가 존재할 때 R의 모든 속성들이 A에 함수적 종속 (FD)이면 R은 4NF
(즉, R의 모든 애트리뷰트 X에 대해 $A \rightarrow X$ 이고 A가 후보키)
- 엔티티 R이 BCNF에 속하고 모든 MVD(다치종속성)가 FD이면 R은 4NF
- 엔티티 R이 4NF이라면 MVD가 없거나, MDV A B|C가 있을 경우 A에 대응되는 B와 C의 값은 하나씩 이어야 하며 이때 A는 후보키라는 것 의미

사번	이름	전화	자격증	사번	이름	전화	사번	자격증
1	김갑돌	111-1111		1	김갑돌	111-1111	2	정보처리가사
2	이갑돌	1111-222	정보처리가사	2	이갑돌	1111-222	4	운전면허
3	최갑돌	222-3333		3	최갑돌	222-3333		
3	전갑동	333-444	운전면허	3	전갑동	333-444		

제5정규형 : 조인 종속성

- 제 5 정규형은 주어진 테이블이 그보다 더 작은 테이블의 join으로 재구성할 수 없을 때, 즉 테이블을 분해하면 원래 가지고 있던 정보가 손실되어서, 더 이상 테이블을 분해할 수 없는 경우일 때, 그 테이블을 제 5 정규형이라고 합니다. 테이블을 칼럼들로 분해하는 것이 Projection, 다시 합치는 것이 Join, 그래서 이 둘을 합쳐 제 5 정규형을 PJNF라고도 합니다. 정보의 손실이 없이 더 작은 테이블로 분해할 수 있는가, 분해된 테이블들을 Join하면 원래의 테이블이 구성되는가에 있습니다. 원래의 테이블보다 더 작은 테이블로 분해될 수 있고, 이들을 다시 Join해서 원래의 테이블이 구성된다면, 이 테이블은 Join Dependency를 갖게 되는 것이지요.
- "릴레이션(테이블) R은 Join Dependency $*(R_1, R_2, \dots, R_n)$ 을 만족한다"
- \leftrightarrow "R의 부분집합인 각 R_i 에 대하여 R은 R_1, R_2, \dots, R_n 의 Join과 같다"



정규화

학생	수강과목	동아리
홍길동	심리학	타임반
홍길동	심리학	영화감상반
홍길동	컴퓨터	타임반
홍길동	컴퓨터	영화감상반
홍길동	교양영어	타임반
홍길동	교양영어	영화감상반
이몽룡	교양영어	영어회화반
이몽룡	교양영어	유도반
이몽룡	프로그래밍 기초	영어회화반
이몽룡	프로그래밍 기초	유도반

⇐ BCNF

∴ (키에 속하지 않는 결정자 속성이 없음)

기본키: (학생, 수강과목, 동아리)



학생등록

수강등록

동아리등록

학생	수강과목
홍길동	심리학
홍길동	컴퓨터
홍길동	교양영어
이몽룡	교양영어
이몽룡	프로그램의기초

학생	동아리
홍길동	타임반
홍길동	영화감상반
이몽룡	타임반
이몽룡	영화감상반

❖ 정규화

✓ 비 정규형과 정규형의 특징

● 비정규형(Non-Normal Form)

- ◆ 업무 요건의 변경에 매우 취약. 즉 모델의 확장성이 좋지 않음
- ◆ 인덱스 수가 증가하고 특정 요건 조회 시 SQL이 복잡해 짐
- ◆ Entity의 속성이 추가될 가능성이 없을 때 사용 가능
- ◆ 속성 레벨로 관리되는 데이터의 자식 Entity를 가질 수 없음(여러 데이터가 하나의 인스턴스에 묶여 있어 개별로 관리해야 할 하위 데이터가 있으면 관리 불가능)

● 정규형(Normal Form)

- ◆ 업무 요건 변경이 유연. 확장성 좋음
- ◆ 인덱스 수 감소, 특정 요건 조회 시 SQL 단순해 짐(복잡해질 수도 있음)
- ◆ Entity의 속성이 추가될 가능성이 존재할 때 사용
- ◆ 로우 레벨로 관리되는 데이터의 자식 Entity를 가질 수 있음

모델링

❖ 정규화의 문제점 과 해결 방안

✓ 정규화의 문제점

- 빈번한 Join 연산의 증가 : 시스템 성능 저하
- 부자연스러운 DB Semantic 초래
- 조회/검색 위주의 응용시스템에 부적합

✓ 문제점 해결 방안 및 업계 동향

- 정규화 완료 후 업무특성과 성능 향상을 위해 De-normalization 수행으로 유연성 확보
- 업무 특성 별 정규화 수준
 - ◆ 온라인 처리: 소규모 정규화
 - ◆ 단순 조회 용 데이터: 비정규화
 - ◆ 배치 처리: 비정규화
- 응답 시간이 요구되는 온라인, 실시간 시스템에는 제한적인 정규화
- 기억 장치가 대용량화 되고 저렴해지면서 정규화의 중요성이 저하되고 성능이 우선시 됨

모델링

❖ 반정규화를 통한 성능 향상 전략

✓ 반정규화

- 반정규화는 정규화된 Entity, 속성, 관계를 시스템의 성능 향상 및 개발과 운영의 단순화를 위해 중복, 통합, 분리 등을 수행하는 데이터 모델링 기법
- 디스크 I/O량이 많아서 조회 시 성능이 저하되거나 테이블끼리의 경로가 너무 멀어 조인으로 인한 성능저하가 예상되거나 컬럼을 계산하여 읽을 때 성능이 저하될 것이 예상되는 경우 반정규화를 수행
- 업무적으로 조회에 대한 처리 성능이 중요하다고 판단될 때 부분적으로 반정규화를 고려하는데 반정규화를 수행하게 되면 모델의 유연성은 떨어지게 됨
- 설계 단계에서 반정규화를 적용하게 되며 반정규화 미수행시에는 다음과 같은 현상이 발생할 수 있음
 - ◆ 성능이 저하된 데이터베이스가 생성될 수 있음
 - ◆ 구축 단계나 시험 단계에서 반정규화를 적용할 때 수정에 따른 노력 비용이 많이 소모

모델링

❖반정규화를 통한 성능 향상 전략

✓ 반정규화 적용 방법

- 반정규화에 대한 필요성이 결정되면 컬럼의 반정규화 뿐만 아니라, 테이블의 반정규화, 관계의 반정규화를 종합적으로 고려하여 적용
- 반정규화는 막연하게 중복을 유도하는 것만을 수행하기 보다는 성능을 향상시킬 수 있는 다른 방법을 고려하고 그 이후에 반정규화를 적용
- 반정규화의 대상을 조사
 - ◆ 자주 사용되는 테이블에 액세스하는 프로세스의 수가 가장 많고 항상 일정한 범위만을 조회하는 경우에 반정규화를 검토
 - ◆ 테이블에 대량데이터가 있고 대량의 범위를 자주 처리하는 경우 성능을 보장할 수 없는 경우에 반정규화를 검토
 - ◆ 통계성 프로세스에 의해 통계정보를 필요로 할 때 별도의 통계 테이블(반정규화)를 생성
 - ◆ 테이블에 지나치게 조인을 많이 하게 되어 데이터를 조회하는 것이 기술적으로 어려울 경우 반정규화를 검토

모델링

❖ 반정규화를 통한 성능 향상 전략

✓ 반정규화 적용 방법

- 반정규화의 대상에 대해 다른 방법으로 처리할 수 있는지 검토
 - ◆ 여러 테이블을 조인하여 데이터를 조회하는 것이 기술적으로 어려운 경우 View를 검토하는데 조회 성능을 향상 시키지는 않지만 SQL작성의 미숙함으로 인하여 생기는 성능 저하를 예방할 수 있음
 - ◆ 대량의 데이터 처리나 부분 처리에 의해 성능이 저하되는 경우 클러스터링을 적용하거나 인덱스 조정을 통해 성능을 향상 시킬 수 있음
 - ◆ 대량의 데이터는 PK의 성격에 따라 파티셔닝 기법을 적용하여 성능 저하를 방
 - ◆ 어플리케이션에서 로직을 구현하는 방법을 변경함으로써 성능을 향상 시킬수 있음
- 반정규화 적용
 - ◆ 반정규화 대상으로는 테이블, 속성, 관계에 대해 적용할 수 있으며, 중복을 통한 방법만이 반정규화가 아니고, 테이블,속성,관계를 추가/분할/제거할 수도 있다.

❖반정규화 기법

✓ 테이블 반정규화

● 테이블 추가

- ◆ 중복 테이블 추가: 다른 업무이거나 서버가 다른 경우 동일한 테이블구조를 중복하여 원격 조인을 제거하여 성능을 향상
- ◆ 통계 테이블 추가: SUM,AVG 등을 미리 수행하여 계산해 둬으로써 조회시 성능을 향상
- ◆ 이력 테이블 추가: 이력 테이블 중에서 마스터 테이블에 존재하는 레코드를 중복하여 이력 테이블에 존재시키는 방법
- ◆ 부분 테이블 추가: 하나의 테이블을 전체 컬럼 중 자주 이용하는 집중화된 컬럼이 있을 경우, 디스크I/O를 줄이기 위해 해당 컬럼들을 모아놓은 별도의 반정규화된 테이블을 생성

❖반정규화 기법

✓ 테이블 반정규화

● 테이블 병합

- ◆ 1:1 관계 테이블 병합 - 1:1 관계를 통합하여 성능 향상
- ◆ 1:M 관계 테이블 병합 - 1:M 관계를 통합하여 성능 향상
- ◆ 슈퍼/서브타입 테이블 병합 - 슈퍼/서브 관계를 통합하여 성능 향상
 - Super type: 여러 Sub Type이 공통으로 가지는 내용을 소유하고 있는 타입
 - Sub Type: 자신만의 가져야 하는 내용을 소유하고 있는 타입

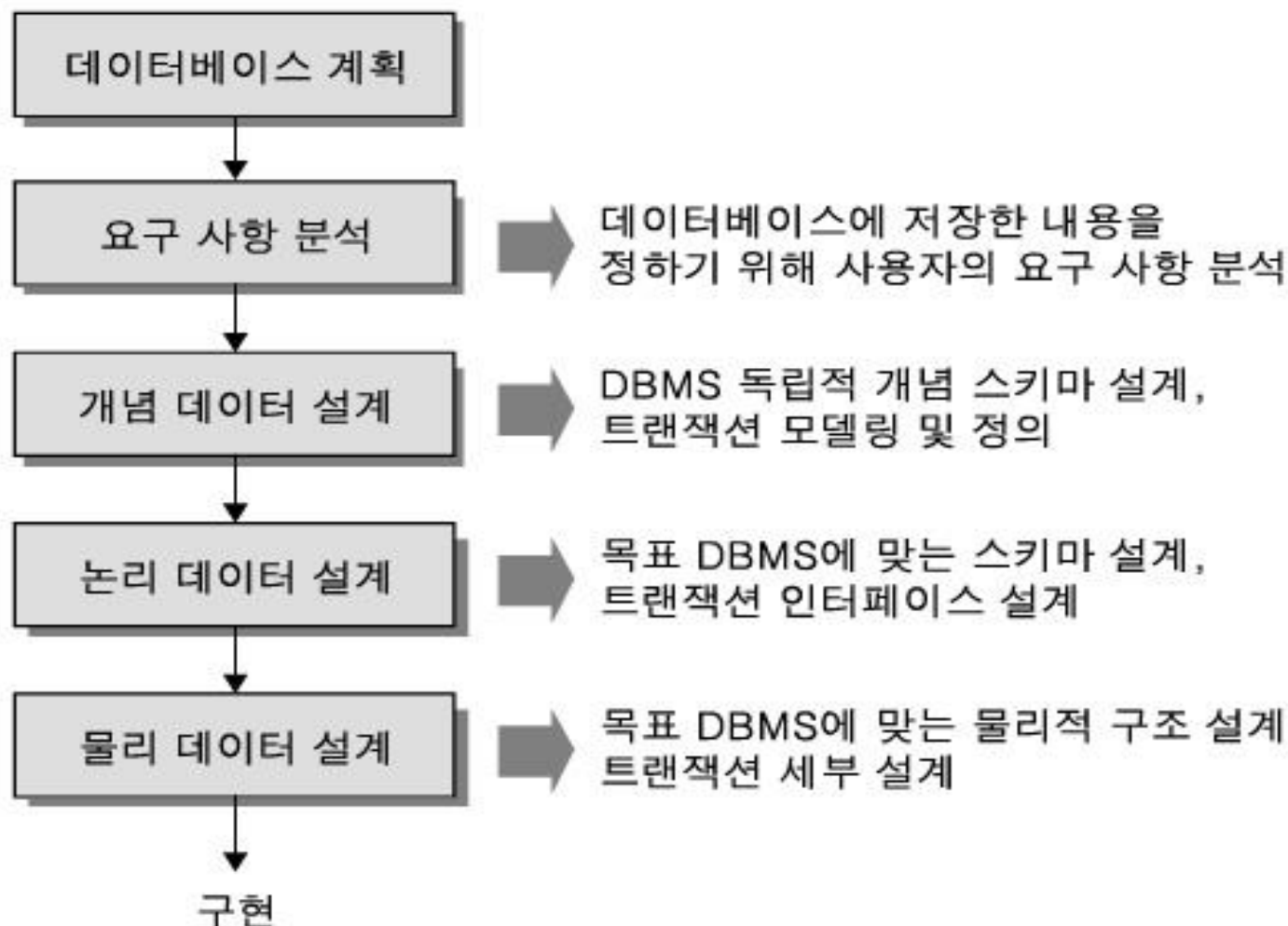
데이터베이스 설계

- 데이터베이스 설계 시 고려 사항

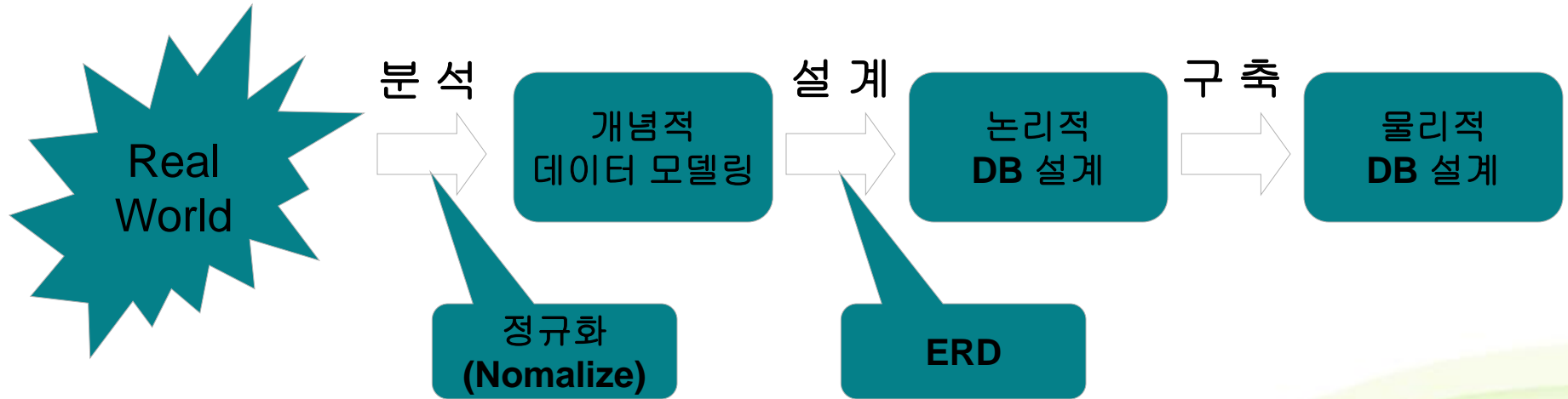
항목	설명
무결성	데이터가 만족해야 할 제약 조건 만족 (갱신, 삽입, 삭제 등의 연산 후에도 데이터 값이 정확)
일관성	저장된 데이터와 질의응답 일치하여 모순성이 없어야 함
회복	시스템에 장애 발생 시 장애 발생 전의 일관된 상태의 데이터베이스 상태로 복구 가능
보안	불법 접근(데이터의 변경, 손실, 노출)에 대한 보호 가능
효율성	응답시간 단축, 저장 공간 최적화, 시스템의 생산성(처리도) 등을 고 려
데이터베이스 확장	응용과 데이터의 확대(시스템에 영향을 주지 않고, 새로운 데이터 추 가 가능)

데이터베이스 설계

- 데이터베이스 설계 과정



데이터베이스 설계



- 1) 사람 : 사원, 회원
- 2) 사건 : 계약, 작업
- 3) 장소 : 창고, 지역
- 4) 개념 : 목표, 계획

- 1) Entity
 - 관리대상
- 2) Attribute
 - 관리대상의 구체적 내용
- 3) Identifier
 - Entity를 대표하는 Att.
- 4) Tuple
 - 여러 개의 Att.의 집합
- 5) Relation
 - Entity와 Entity의 관계

- 1) TABLE
- 2) COLUMN
- 3) PRIMARY-KEY
- 4) ROWS
- 5) FOREIGN-KEY

SQL

- SQL(Structured Query Language)
 - 1974년 IBM 연구소에서 발표된 SEQUEL(Structured English QUery Language)에서 유래
 - 특징
 - SQL은 비절차적인
 - 대화식 언어로 사용 가능
 - 다른 종류의 범용 프로그래밍 언어로 작성된 프로그램에 내장(embed)시킨 형태로도 사용 가능
 - 각각의 튜플 단위가 아니라 튜플들을 집합 단위로 처리

SQL의 특징

관계대수와 관계해석을 기초로 한 고급 데이터 언어
이해하기 쉬운 형태로 표현
대화식 질의어로 사용 가능
데이터 정의, 데이터 조작, 제어 기능 제공
COBOL, C, PASCAL 등이 언어에 삽입
레코드 집합 단위로 처리
비절차적 언어

SQL

- SQL의 명령어

- DDL (Data Definition Language) : 데이터베이스 및 테이블의 구조를 정의하거나 변경

SQL문	내 용
CREATE	데이터베이스 및 객체 생성
DROP	데이터베이스 및 객체 삭제
ALTER	기존에 존재하는 데이터베이스 객체를 변경

- DML (Data Manipulation Language) : 데이터의 삽입, 삭제, 검색과 수정 등을 처리

SQL문	내 용
INSERT	데이터베이스 객체에 데이터를 입력
DELETE	데이터베이스 객체에 데이터를 삭제
UPDATE	기존에 존재하는 데이터베이스 객체안의 데이터 수정
SELECT	데이터베이스 객체로부터 데이터를 검색

SQL

- **DCL (Data Control Language) : 데이터베이스 사용자의 권한을 제어**

SQL문	내 용
GRANT	데이터베이스 객체에 권한 부여
REVOKE	이미 부여된 데이터베이스 객체의 권한 취소

ERD작성 툴

Amateras ERD

<http://takezoe.github.io/amateras-update-site/>

eXERD

http://www.exerd.com/ko_KR/

er win

<https://erwin.com/>

데이터베이스 모델링 실습

쇼핑몰 관리 프로그램

❖ 쇼핑몰 운영 규칙

- ✓ 처음 가입시 준회원으로 등록하며 매출이 10000원을 넘을 경우 정회원이 되고 50000원이 되면 우수회원으로 등록
- ✓ 정회원 이상은 배송비가 무료이며 우수 회원은 10% 추가 할인
- ✓ 간편한 결제를 위해 예치금을 맡기고 예치금으로만 쇼핑
- ✓ 현금이나 신용카드로는 구매할 수 없음
- ✓ 가입 즉시 축하금 1000원을 예치금으로 지급하고 예치금 10000원 적립 시 12000원 입금으로 처리
- ✓ 상품은 몇 가지 유형으로 나누고 유형에 따라 할인율, 배송비, 반품여부 가 다름
- ✓ 유형은 고정이며 편집 기능은 제공하지 않음
- ✓ 미성년자(19세 미만)에게는 성인용품을 판매하지 않음

쇼핑몰 관리 프로그램

❖ Entity 추출

- ✓ 모델링의 첫 단계는 업무 분석 결과를 바탕으로 Entity를 추출하는 것
- ✓ 쇼핑을 하려면 사는 사람이 있어야 하고 팔 물건이 있어야 하니 회원과 상품 두 명사가 기본적인 Entity
- ✓ 회원은 상품을 구입하는 식으로 관계를 맺으므로 이 관계를 주문 Entity로 정의
- ✓ 주문이 없으면 회원과 상품이 관계를 맺을 수 없으므로 주문은 쇼핑몰의 주요 Entity
- ✓ 예치금과 고객의 나이, 주문 수량 등은 Entity의 속성일 뿐 Entity는 아님
- ✓ Entity 추출 결과 회원, 상품, 주문이라는 3개의 Entity가 필요함을 파악

쇼핑몰 관리 프로그램

❖ 회원 테이블

- ✓ 회원의 속성을 나열해 보고 이 중 쇼핑몰에 어떤 정보가 필요한지 생각
- ✓ 회원은 사람이므로 다음과 같은 속성을 가지는데 이 중 업무에 사용할 만 한 속성만 추려내야 함
 - 이름, 전화번호, 주소, 이메일, 나이, 키, 몸무게, 고향, 시력, 혈액형, 좋아하는 연예인,
 - 이름은 기본 정보라 당연히 필요하고 배송을 위한 주소가 있어야 함
 - 배송 안내를 위해 이메일이나 전화번호가 필요
 - 업무 규칙에 미성년자 어쩌고 하는 조항이 있어 나이는 필요하지만 키나 몸무게, 고향 등은 쇼핑몰 업무와는 하등의 상관이 없어 제외
- ✓ 여기까지 회원 자신이 가진 고유한 속성을 선정하였는데 이 외에 업무에 필요한 인위적인 속성을 추가하는데 업무 규칙에 회원의 등급을 관리하는 항목이 있어 각 회원은 등급을 가져야 하고 예치금은 회원별로 관리되므로 회원의 속성으로 표현하는 것이 논리상 합당
- ✓ 다음은 식별을 위한 기본키를 선정하는데 이름은 동명이인 문제로 기본키로 잘 쓰지 않으며 전화 번호나 주소는 한 집에 사는 두 회원(예를 들어 고시원이나 하숙생)이 따로 가입할 수 있어 검색 대상일 수는 있어도 기본키로는 부적합한데 이럴 때는 별도의 ID를 쓰는 것이 보편적
- ✓ 여기까지 모델링한 결과를 문서로 정리하여 구현 단계에서 실제 테이블로 생성

쇼핑몰 관리 프로그램

❖ 회원 테이블

-- 회원 테이블

CREATE TABLE tMember

```
(
    member VARCHAR(20) PRIMARY KEY,    -- 아이디
    age INT NOT NULL,                   -- 나이
    email VARCHAR(30) NOT NULL,         -- 이메일
    ADDr VARCHAR(50) NOT NULL,          -- 주소
    money INT DEFAULT 1000 NOT NULL,     -- 예치금
    grade INT DEFAULT 1 NOT NULL,        -- 고객등급. 1=준회원, 2=정회원, 3=우수회원
    remark VARCHAR(100) NULL            -- 메모 사항
);
```

-- 회원 데이터

```
INSERT INTO tMember VALUES ('춘향',16,'1004@naver.com','전남 남원시',20000, 2, '');
INSERT INTO tMember VALUES ('이도령',18,'wolf@gmail.com','서울 신사동',150000, 3, '');
INSERT INTO tMember VALUES ('향단',25,'candy@daum.net','전남 남원시',5000, 2, '');
INSERT INTO tMember VALUES ('방자',28,'devlin@ssang.co.kr','서울 개포동',1000, 1, '요주의 고객');
```


쇼핑몰 관리 프로그램

❖ 상품 테이블

- ✓ 상품명이 필요하고 제조사, 정가 등의 기본적인 정보도 있어야 함
- ✓ 재고 정보는 상품 자체의 속성은 아니지만 업무상 필요
- ✓ 잘 팔리는 제품은 재고를 넉넉히 확보해 놔야 하며 인기 없는 상품은 한 두개만 있어도 됨
- ✓ 개별 상품 별로 레코드를 정의할 필요는 없고 같은 상품은 레코드 하나로 정의하고 재고 수만 기록
- ✓ 고객 입장에서는 남은 재고 중 어떤 상품을 구입 하나 동일
- ✓ 상품의 또 다른 속성으로 할인율과 배송비, 반품 여부 등이 있음
- ✓ 식품은 포장비가 많이 들어 배송비가 비싸며 할인 여력이 없고 부패 위험이 있어 반품을 받지 않음
- ✓ 전자 제품은 마진이 높아 할인을 왕창 해 줘도 되고 반품도 받아 줌
- ✓ 이런 규칙은 개별 상품마다 다른 것이 아니라 유형 별로 결정
- ✓ 기본키인 상품에 종속적이지 않고 일반 필드인 유형에 종속적

쇼핑몰 관리 프로그램

❖ 상품 테이블

✓ 1차 모델링

상품	제조사	재고	정가	유형	할인율	배송비	반품여부
노트북	삼성	3	820000	가전	20	2500	y
모니터	알지	1	450000	가전	20	2500	y
사과	문경농원	24	16000	식품	0	3000	n
대추	보은농원	19	15000	식품	0	3000	n
전자담배	T&G	4	7000	성인	5	1000	n
청바지	방방	99	21000	패션	10	2000	y

쇼핑몰 관리 프로그램

❖ 상품 테이블

- ✓ 1차 모델링한 위 테이블은 기본키가 아닌 일반 필드끼리 독립적이어야 한다는 제 3 정규화 규칙을 위반
- ✓ 할인율, 배송비, 반품 여부가 계속 중복되어 노트북과 모니터, 사과와 대추의 뒷부분이 같음
- ✓ 정규화하여 상품과 유형 테이블로 분리
- ✓ 상품 Entity에 직접 종속되지 않는 할인율, 배송비, 반품 여부는 제외
- ✓ 유형과 관련된 정보는 별도의 유형 테이블로 분리
- ✓ 상품명에 중복되는 경우도 가끔 있어 ID 필드가 있으면 좋지만 논리를 간단히 하기 위해 상품 의 이름인 item을 PK로 지정
- ✓ 담배 회사는 T&G가 맞지만 오라클이 &를 특수 기호로 쓰고 있어 TNG로 작성
- ✓ 상품의 분류는 category 필드로 지정하며 유형별 상세 정보를 구할 수 있는 외래키

쇼핑몰 관리 프로그램

❖ 상품 테이블

상품	제조사	재고	정가	유형
노트북	삼성	3	820000	가전
모니터	알지	1	450000	가전
사과	문경농원	24	16000	식품
대추	보은농원	19	15000	식품
전자담배	T&G	4	7000	성인
청바지	방방	99	21000	패션

유형	할인율	배송비	반품여부
가전	20	2500	y
식품	0	3000	n
성인	5	1000	n
패션	10	2000	y

쇼핑몰 관리 프로그램

❖ 상품 테이블

-- 상품 분류 테이블

CREATE TABLE tCategory

(

category VARCHAR(10) PRIMARY KEY,

discount INT NOT NULL,

delivery INT NOT NULL,

takeback CHAR(1)

);

-- 분류명

-- 할인율

-- 배송비

-- 반품 가능성

-- 분류 데이터

INSERT INTO tCategory (category, discount, delivery, takeback) VALUES ('식품', 0, 3000, 'n');

INSERT INTO tCategory (category, discount, delivery, takeback) VALUES ('패션', 10, 2000, 'y');

INSERT INTO tCategory (category, discount, delivery, takeback) VALUES ('가전', 20, 2500, 'y');

INSERT INTO tCategory (category, discount, delivery, takeback) VALUES ('성인', 5, 1000, 'n');

쇼핑몰 관리 프로그램

❖ 상품 테이블

-- 상품 테이블

CREATE TABLE tItem

```
(
    item VARCHAR(20) PRIMARY KEY,          -- 상품명
    company VARCHAR(20) NULL,               -- 제조사
    num INT NOT NULL,                       -- 재고
    price INT NOT NULL,                     -- 정가
    category VARCHAR(10) NOT NULL,          -- 분류
    CONSTRAINT item_fk FOREIGN KEY(category) REFERENCES tCategory(category)
);
```

쇼핑몰 관리 프로그램

❖ 상품 테이블

-- 상품 데이터

```
INSERT INTO tItem (item,company,num,price,category) VALUES ('노트북', '삼성', 3, 820000, '가전');
```

```
INSERT INTO tItem (item,company,num,price,category) VALUES ('청바지', '방방', 80, 32000, '패션');
```

```
INSERT INTO tItem (item,company,num,price,category) VALUES ('사과', '문경농원', 24, 16000, '식품');
```

```
INSERT INTO tItem (item,company,num,price,category) VALUES ('대추', '보은농원', 19, 15000, '식품');
```

```
INSERT INTO tItem (item,company,num,price,category) VALUES ('전자담배', 'TNG', 4, 70000, '성인');
```

```
INSERT INTO tItem (item,company,num,price,category) VALUES ('마우스', '논리텍', 3, 90000, '가전');
```


쇼핑몰 관리 프로그램

❖ 주문 테이블

- ✓ 주문은 회원이 상품을 구입하는 동작인데 이 둘은 전형적인 다:다 관계
- ✓ 다대다 관계 이므로 상품, 회원 두 테이블만으로는 주문 관계를 표현 할 수 없으며 중간에 두 Entity의 관계를 1:다로 표현하는 주문 Entity가 필요
- ✓ 주문은 누가 언제 어떤 상품을 구입했는지 모두 포함해야 하며 제대로 배송했는지 상태를 기록하는 속성도 있어야 하고 메모 사항도 남길 수 있어야 함

쇼핑몰 관리 프로그램

❖주문 테이블

```
CREATE TABLE tOrder
```

```
(
```

```
  orderID INT AUTO_INCREMENT PRIMARY KEY, -- 주문 번호
```

```
  member VARCHAR(20) NOT NULL REFERENCES tMember(member),
```

```
  -- 주문자
```

```
  item VARCHAR(20) NOT NULL REFERENCES tItem(item),
```

```
  -- 상품
```

```
  orderDate DATE DEFAULT SYSDATE() NOT NULL,
```

```
-- 주문 날짜
```

```
  num INT NOT NULL,
```

```
-- 개수
```

```
  status INT DEFAULT 1 NOT NULL,
```

```
-- 1:주문, 2:배송중, 3:
```

```
  배송완료, 4:반품
```

```
  remark VARCHAR(1000) NULL
```

```
-- 메모 사항
```

```
);
```

쇼핑몰 관리 프로그램

❖ 주문 테이블

-- 주문 데이터

```
INSERT INTO tOrder (member,item,orderDate,num,status) VALUES ('춘향','청바지','2019-12-3',3,2);
```

```
INSERT INTO tOrder (member,item,orderDate,num,status) VALUES ('향단','대추','2019-12-4',10,1);
```

```
INSERT INTO tOrder (member,item,orderDate,num,status) VALUES ('방자','전자담배','2019-12-2',4,1);
```

```
INSERT INTO tOrder (member,item,orderDate,num,status) VALUES ('향단','사과','2019-12-5',5,2);
```

```
INSERT INTO tOrder (member,item,orderDate,num,status) VALUES ('이도령','노트북','2019-12-5',2,1);
```

```
INSERT INTO tOrder (member,item,orderDate,num,status) VALUES ('방자','노트북','2019-12-1',1,3);
```