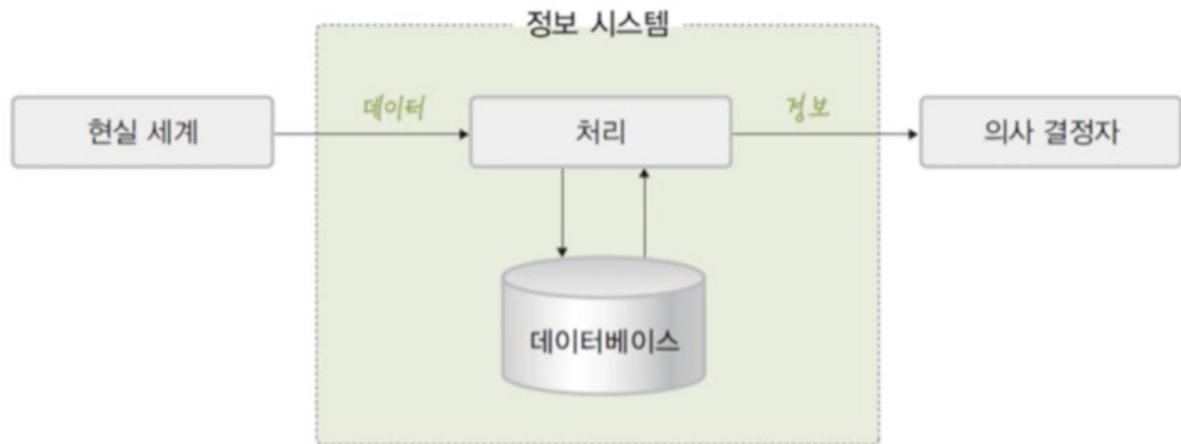


데이터베이스

데이터들의 집합.

- 데이터 vs 정보
 - 데이터
 - 현식 세계에서 관찰된 값
 - 의미부여 X
 - 정보
 - 의사 결정에 유용하게 쓰려고 데이터를 처리한 결과물
 - 의미 부여 O
- 정보 처리(information processing)
 - 데이터에서 정보를 추출하는 과정
 - Ex. 주문 내역 -정보 처리-> 제품별 총 판매액
- 정보 시스템 (information system)
 - 조직 운영에 필요한 데이터를 수집하여 저장해 두었다가 필요할 때 유용한 정보를 만들어 주는 수단
 - 일명 DBMS
- 데이터 베이스
 - 정보 시스템 안에서 데이터를 저장하고 있다가 필요할 때 제공하는 역할을 담당



- 데이터베이스
 - 특정 조직의 여러 사용자가 공유하여 사용할 수 있도록 통합해서 저장한 운영데이터의 집합
 - 공유 데이터
 - 특정 조직의 여러 사용자가 함께 소유하고 이용할 수 있는 공용 데이터
 - 통합 데이터
 - 최소의 중복과 통제 가능한 중복만 허용하는 데이터
 - 저장 데이터
 - 컴퓨터가 접근할 수 있는 매체에 저장된 데이터
 - 운영 데이터
 - 조직의 주요 기능을 수행하기 위해 지속적으로 유지해야 하는 데이터

- 특성

- 실시간 접근

- 사용자의 데이터 요구에 실시간으로 응답

- 계속 변화

- 데이터의 계속적인 삽입, 삭제, 수정을 통해 현재의 정확한 데이터를 유지

- Ex. 사원의 입사 퇴사에 따른 변화를 반영

- 동시 공유

- 서로 다른 데이터의 동시 사용 뿐 아니라 같은 데이터의 동시 사용도 지원

- 내용 기반 참조

- 데이터가 저장된 주소나 위치가 아닌 내용으로 참조

- Ex. 연봉이 500 이상이고 과장 이상인 사원을 선택해라.

데이터베이스 관리 시스템

- 파일 시스템

- 데이터를 파일로 관리하기 위해 파일을 생성, 삭제, 수정, 검색 기능을 제공하는 SW

- 응용 프로그램마다 필요한 데이터를 별도의 파일로 관리한다.

- 문제점

- 같은 내용의 데이터가 여러 파일에 중복 저장된다. → 데이터 중복성

- 저장 공간의 낭비

- 데이터의 일관성 깨짐

- 데이터의 무결성 깨짐

- 응용 프로그램이 데이터 파일에 종속적이다. → 데이터 종속성

- 사용하는 파일의 구조를 변경하면 응용 프로그램도 함께 변경해야 함.

- 데이터 파일에 대한 동시 공유, 보안, 회복 기능이 부족하다.

- 응용 프로그램 개발이 쉽지 않다.

- 파일 시스템의 주요 문제점

- 같은 내용의 데이터가 여러 파일에 중복되어 존재한다.

- 데이터베이스 관리 시스템 (DBMS)

- 파일 시스템의 문제 (데이터 종속성, 데이터 중복성)를 해결하기 위해 제시된 소프트웨어

- 조직에 필요한 데이터를 데이터베이스에 통합하여 저장하고 관리한다.

- 기능

- 정의 기능

- 데이터베이스의 구조를 정의하거나 수정할 수 있다.

- 조작 기능

- 데이터를 삽입, 삭제, 수정, 검색하는 연산 가능하다.

- 제어 기능

- 데이터를 정확하고 안전하게 유지할 수 있다.

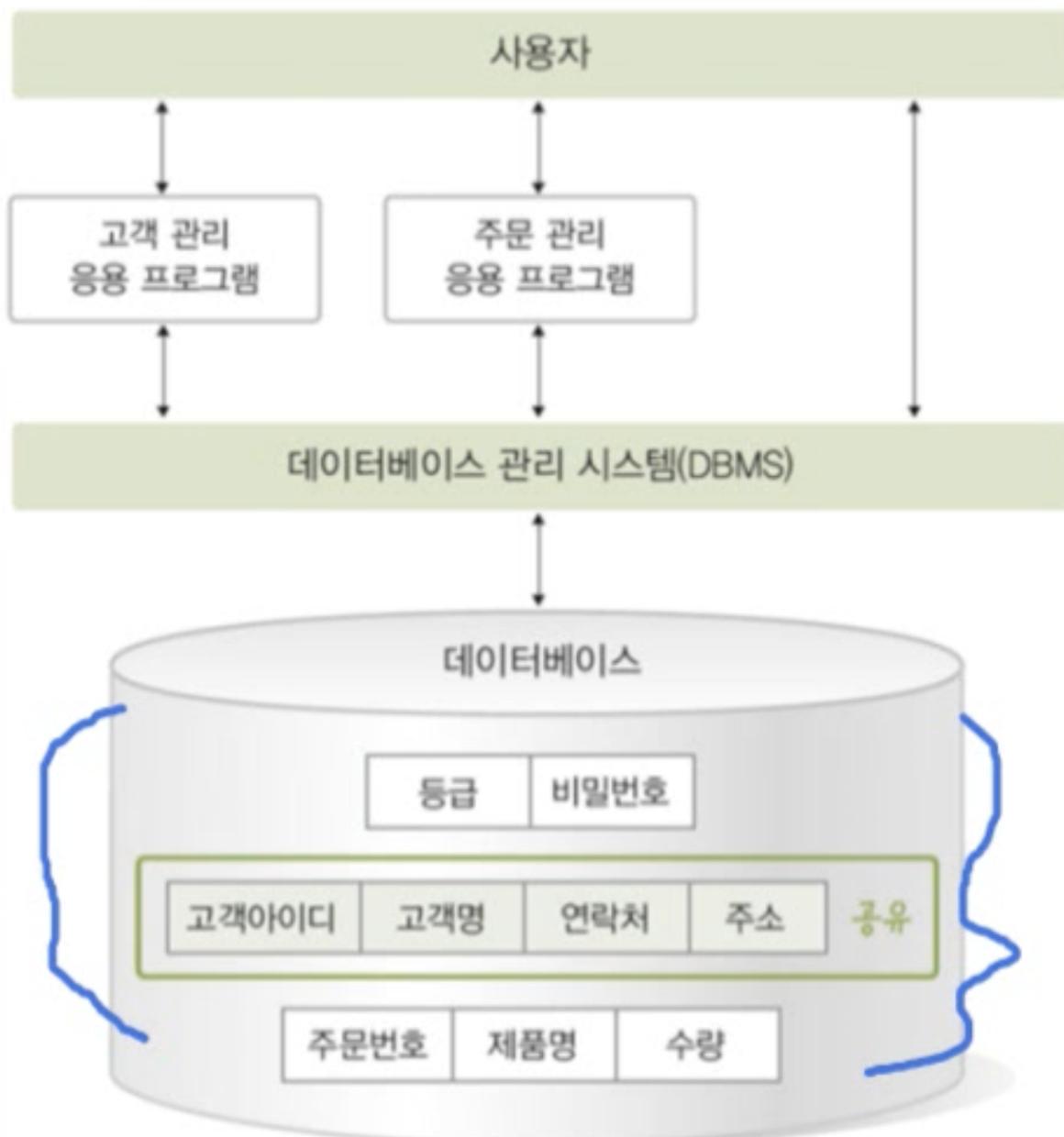
- 장단점

■ 장점

- 데이터 중복 통제 가능
- 데이터 독립성 확보
- 데이터를 동시에 공유 가능
- 데이터 보안이 향상
- 데이터 무결성 유지 가능
- 표준화 가능
- 장애 발생 시 회복이 가능
- 응용 프로그램 개발 비용이 줄어든다.

■ 단점

- 비싸다
- 백업과 회복 방법이 복잡
- 중앙 집중 관리로 인한 취약점



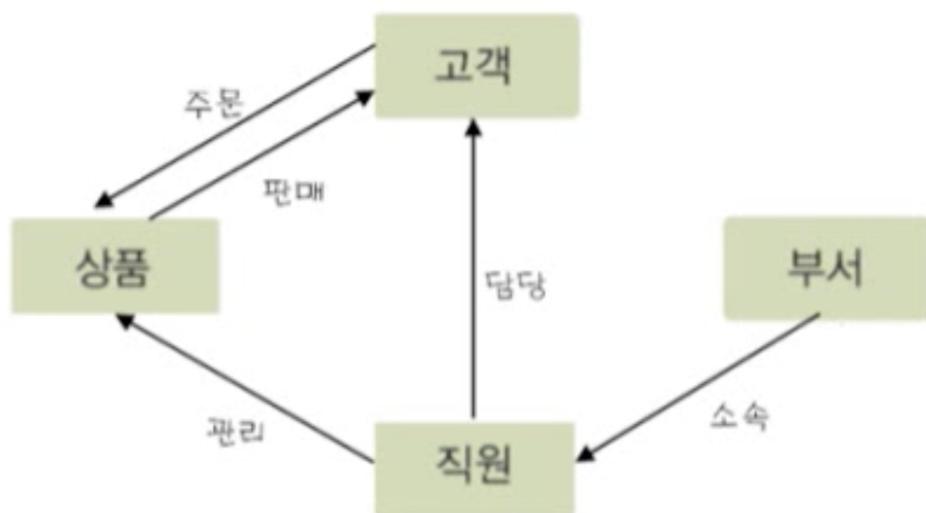
- DBMS 발전 과정

- 1세대

- 네트워크 DBMS

- 구조가 복잡하고 변경이 어렵다.

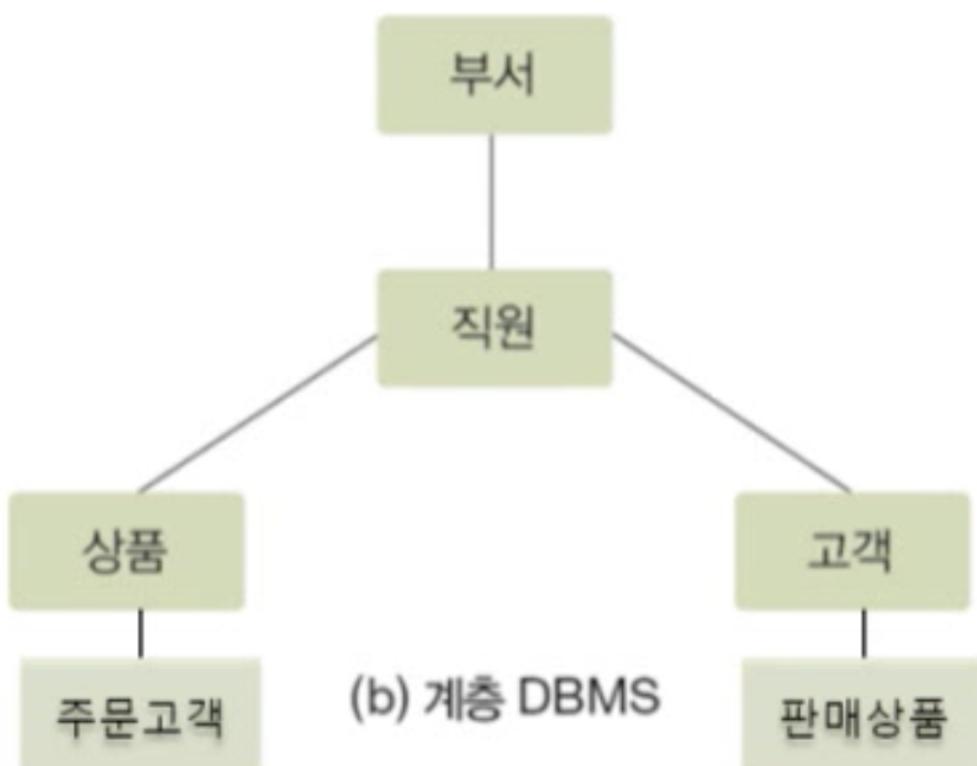
- Ex. IDS



- 계층 DBMS

- 구조 변경이 어렵다.

- Ex. IMS



- 2세대

- 관계 DBMS

- 테이블 형태

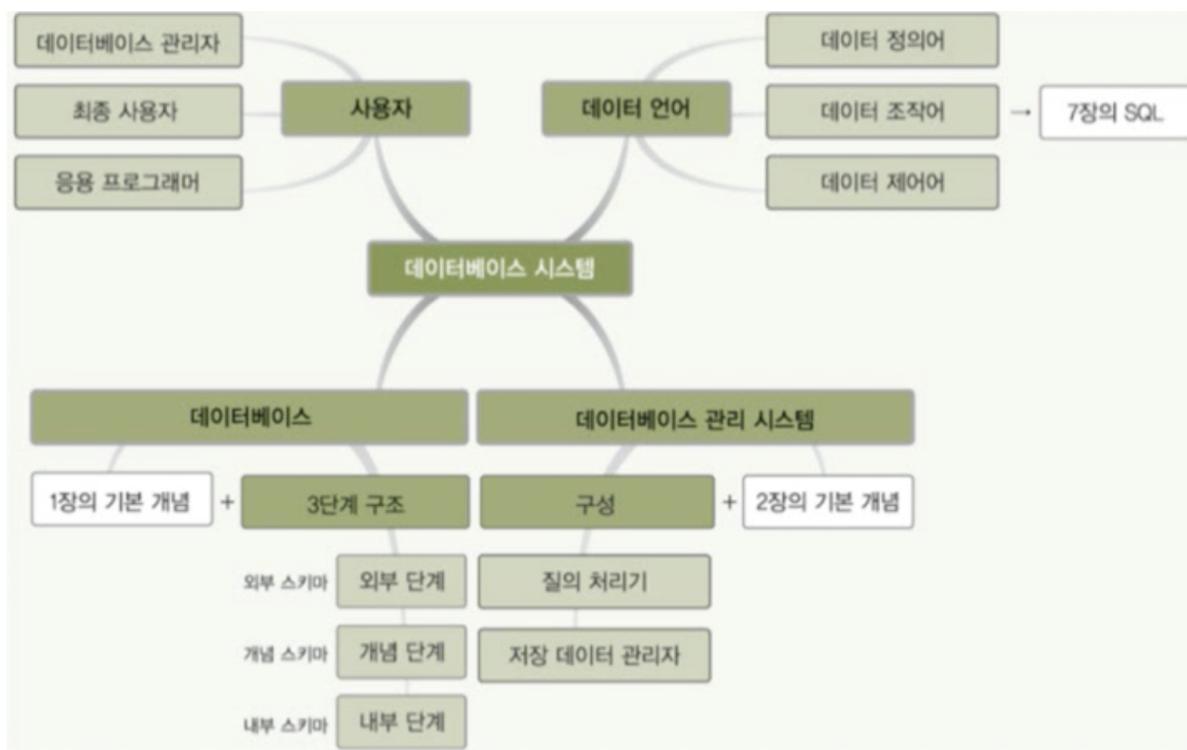
- Ex. Oracle, MySQL Server

- 3세대

- 객체지향 DBMS

- Ex. O2, ONTOS

데이터베이스 시스템



- 데이터베이스 시스템(DBS)
 - 데이터베이스에 데이터를 저장하고, 이를 관리하여 조직에 필요한 정보를 생성 해주는 시스템
 - 구조
 - 스키마(schema)
 - 데이터베이스에 저장되는 데이터 구조와 제약 조건을 정의한 것
 - 빠대
 - 인스턴스(instance)
 - 스키마에 따라 데이터베이스에 실제로 저장된 값
 - 살
 - 3단계 데이터베이스 구조
 - 데이터베이스를 바라보는 관점 3 가지
 - 외부 단계
 - 개별 사용자 관점
 - 데이터베이스 하나에 외부 스키마가 여러 개 존재할 수 있다.
 - 외부 스키마 (서브 스키마)
 - 외부 단계에서 사용자에게 필요한 데이터베이스를 정의한 것
 - 각 사용자가 생각하는 데이터베이스의 모습은 사용자마다 다르다.
 - 개념 단계
 - 조직 전체의 관점
 - 데이터베이스 하나에 개념 스키마가 하나만 존재한다.

개념 스키마

- 개념 단계에서 데이터베이스 전체의 논리적 구조를 정의한 것
- 전체 데이터베이스에 어떤 데이터가 저장되는지, 데이터들 간에는 어떤 관계가 존재하고 어떤 제약조건이 존재하는지 정의 뿐 아니라, 데이터에 대한 보안 정책이나 접근 권한에 대한 정의도 포함한다.

■ 내부 단계

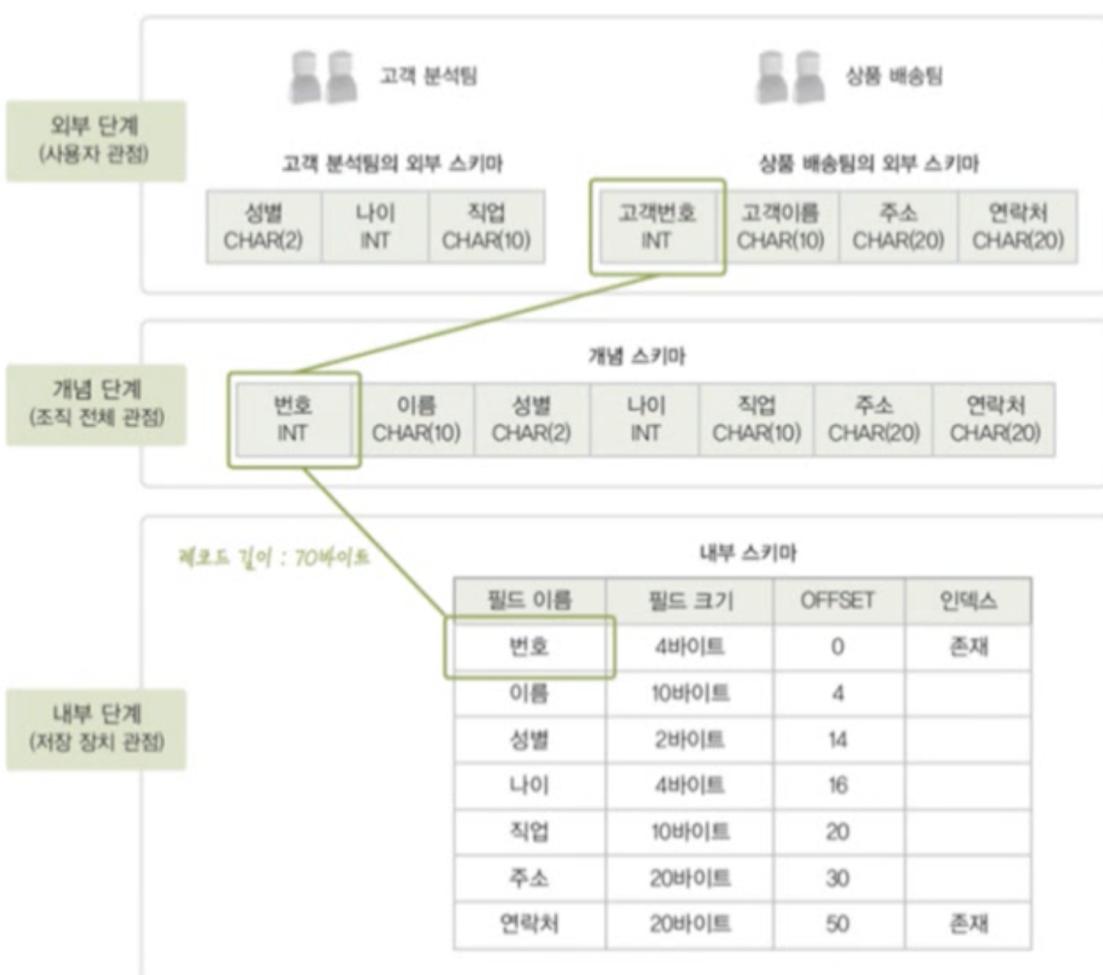
- 물리적인 저장 장치의 관점
- 데이터베이스 하나에 내부 스키마가 하나만 존재한다.

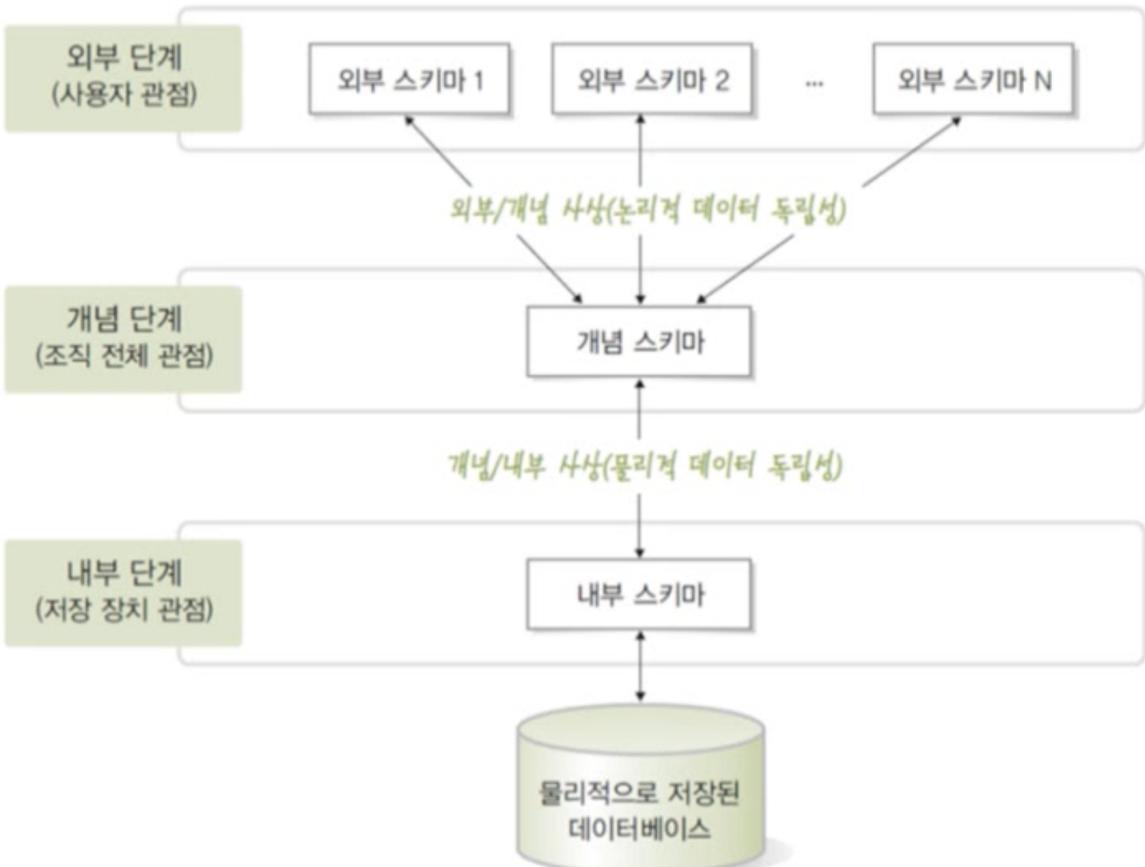
내부 스키마

- 전체 데이터베이스가 저장 장치에 실제로 저장되는 방법을 정의한 것
- 레코드 구조, 필드 크기, 레코드 접근 경로 등 물리적인 저장구조를 정의

○ 각 단계 별로 다른 abstraction 제공

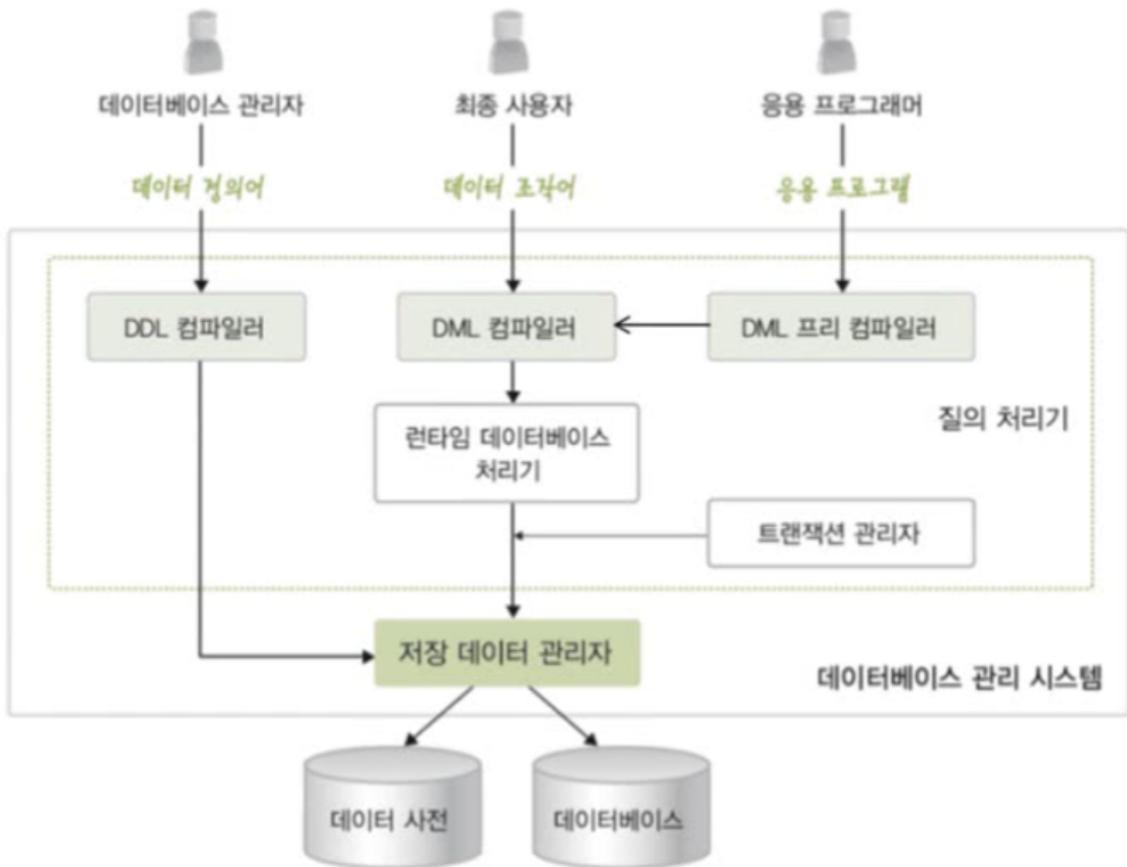
- 내부 단계에서 외부 단계로 갈수록 더 abstract됨





- 외부/개념 사상 (application interface) → 논리적 데이터 독립성
- 개념/내부 사상 (storage interface) → 물리적 데이터 돋립성
- 데이터 독립성
 - 하위 스키마를 변경하더라도 상위 스키마는 영향을 받지 않는 특성
 - 논리적 데이터 독립성
 - 개념 스키마가 변경되어도 외부 스키마는 영향을 받지 않는 특성
 - 개념 스키마가 변경되면 관련 외부/개념 사상만 정확하게 수정해주면 됨.
 - 물리적 데이터 돋립성
 - 내부 스키마가 변경되어도 개념 스키마는 영향을 받지 않는 특성
 - 내부 스키마가 변경되면 관련된 개념/내부 사상만 정확하게 수정해주면 됨
- 데이터 사전
 - system catalog
 - 메타 데이터를 유지하는 시스템 데이터베이스
 - 메타 데이터: 데이터에 대한 데이터
 - 스키마, 사상 정보, 다양한 제약 조건 등을 저장
 - 데이터베이스 관리 시스템이 스스로 생성하고 유지
 - 일반 사용자도 접근이 가능하지만 저장된 내용을 검색만 할 수 있다.
- 데이터 딕토리
 - 데이터 사전에 있는 데이터에 실제로 접근하는데 필요한 위치 정보를 저장하는 시스템 데이터 베이스
 - 일반 사용자의 접근은 불가능

- 사용자 데이터베이스
 - 사용자가 실제로 이용하는 데이터가 저장되어 있는 일반 데이터베이스
- 데이터베이스 사용자
 - 데이터베이스 관리자(DBA)
 - 데이터 베이스 시스템을 운영하고 관리
 - 주로 데이터 정의어, 제어어 사용
 - 최종 사용자
 - 일반 사용자
 - 데이터베이스에 접근하여 데이터를 조작(crud)
 - 주로 데이터 조작어 사용
 - 응용 프로그래머
 - 데이터 언어를 삽입하여 응용 프로그램을 작성
 - 주로 데이터 조작어 사용
- 데이터 언어
 - 데이터 정의어 (DDL)
 - 스키마 정의, 수정, 삭제
 - 데이터 조작어 (DML)
 - 데이터 CRUD 처리
 - 데이터 제어어 (DCL)
 - 내부적으로 필요한 규칙이나 기법을 정의하기 위해 사용
 - 목적
 - 무결성 : 정확하고 유효한 데이터만 유지
 - 보안 : 허가 받지 않은 사용자의 데이터 접근 차단, 허가된 사용자에 대한 부여
 - 회복: 장애가 발생해도 데이터 일관성 유지
 - 동시성 제어: 동시에 공유 지원
- 데이터베이스 관리 시스템
 - 데이터베이스 관리와 사용자의 데이터 처리 요구 수행
 - 구성요소
 - query processor
 - 사용자의 데이터 처리 요구를 해석하여 처리
 - Ex.DDL compiler, DML pre compiler, DML compiler, runtime database processor, transaction manager
 - stored data manager
 - 디스크에 저장된 사용자 데이터베이스와 데이터 사전을 관리하고 여기에 실제로 접근하는 역할을 담당



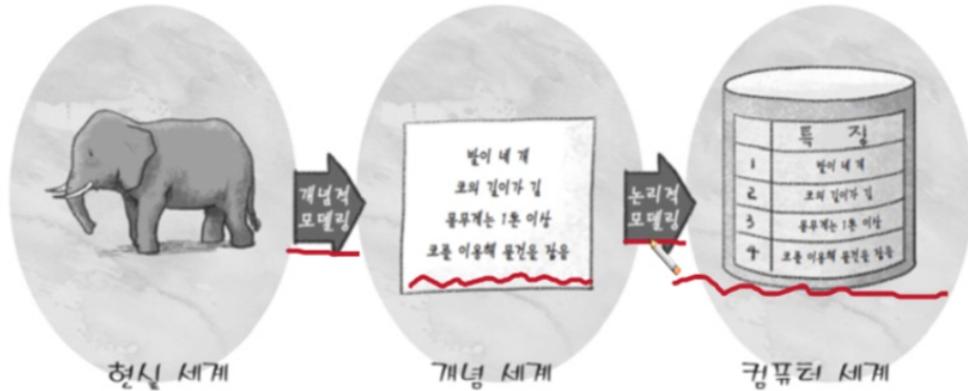
데이터 모델링

- 데이터 베이스 설계



- 데이터 모델링
 - 현실 세계에 존재하는 데이터를 컴퓨터 세계의 데이터베이스로 옮기는 변환 과정
 - 종류
 - 개념적 데이터 모델링
 - 현실 세계의 중요 데이터를 추출하여 개념 세계로 옮기는 과정
 - 논리적 데이터 모델링

- 개념 세계의 데이터를 데이터베이스에 저장하는 구조로 표현하는 작업



- 데이터 모델
 - 데이터 모델링의 결과물
 - 종류
 - 개념적 데이터 모델

- 현실 세계를 데이터베이스의 개념적 구조로 표현

- 개체 관계 모델

- 논리적 데이터 모델
 - 개념적 구조를 논리적 구조로 표현
 - 관계 데이터 모델

- 개체 관계 모델

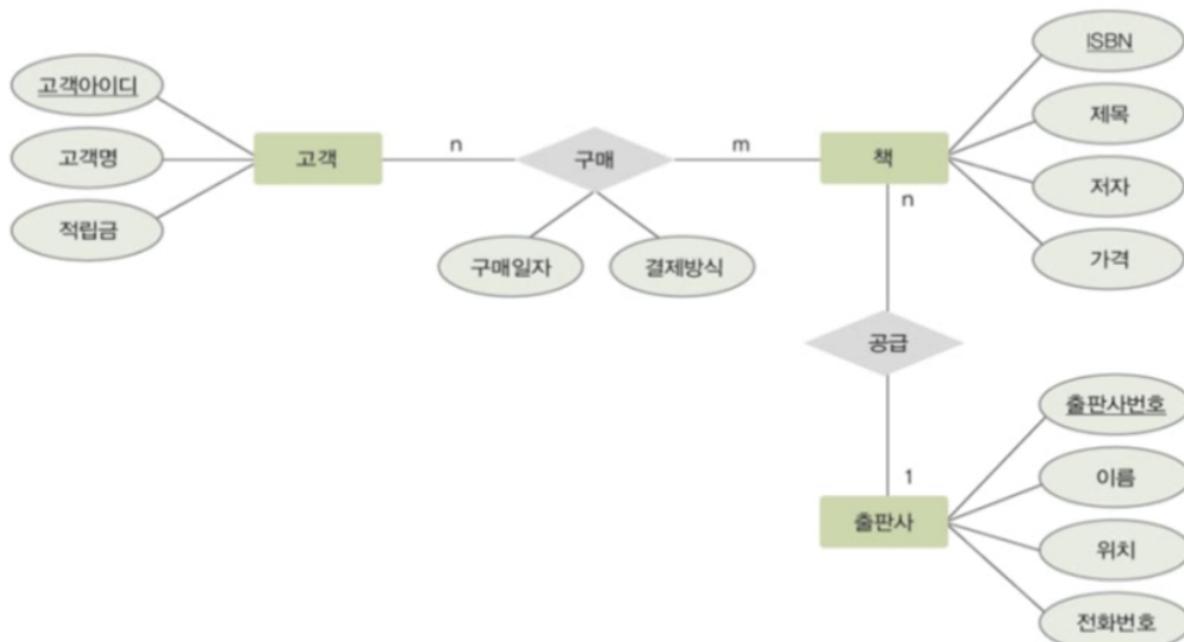
- E-R Model or Entity-Relationship model

- 개체와 개체 간의 관계를 이용해 현실 세계를 개념적 구조로 표현

- 개체 + 속성 + 관계

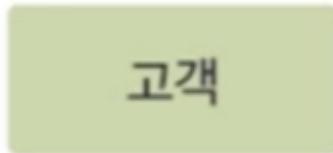
- 개체 관계 다이어그램 (ERD)

- 개체 관계 모델의 결과물

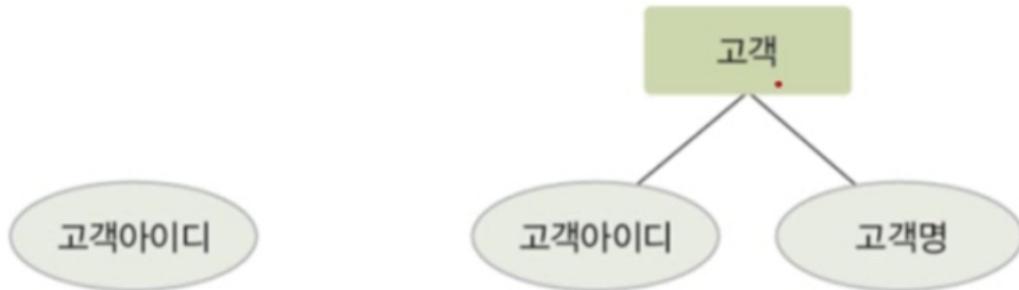


- 개체
 - 현실 세계에서 조직을 운영하는데 꼭 필요한

- 구별되는
- 저장할 가치가 있는 중요 데이터를 가지고 있는
- 다른 개체와 구별되는 이름을 가지고 있고, 각 개체만의 고유한 특성이나 상태(속성)을 하나 이상 가지고 있는
- 모든 것
- Ex. 학교에 필요한 개체: 학과, 과목
- ERD에서 사각형으로 표현하고 사각형 안에 이름을 표기
-



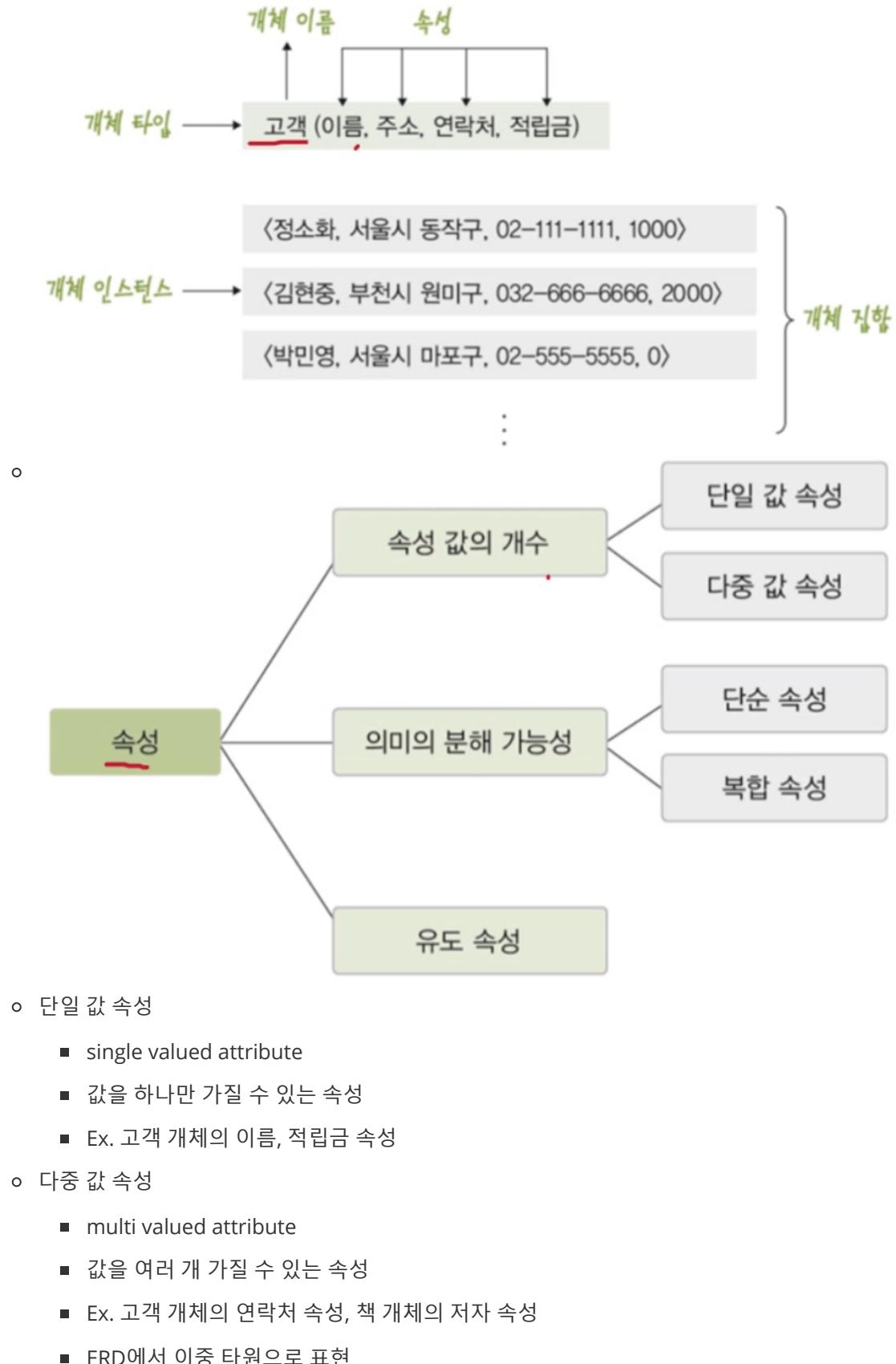
- 속성
 - 개체나 관계가 가지고 있는 고유의 특성
 - 의미 있는 데이터의 가장 작은 논리적 단위
 - ERD에서 타원으로 표현하고 타원 안에 이름을 표기
 -



(a) 고객아이디 속성

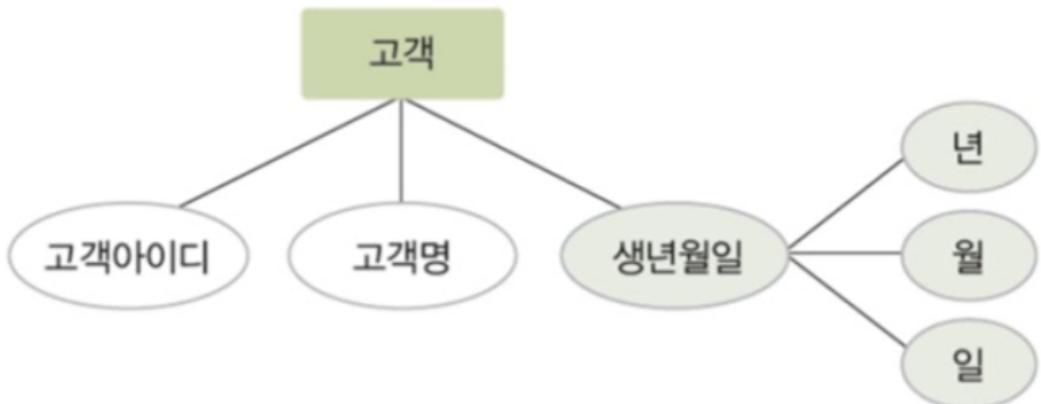
(b) 고객 개체의 속성

- 개체 타입
 - entity type
 - 개체 고유의 이름과 속성들로 정의한 것
- 개체 인스턴스
 - entity instance
 - 개체를 구성하고 있는 속성이 실제 값을 가지게 돼 하나의 어엿한 개체가 된 것
- 개체 집합
 - entity set
 - 특정 개체 타입에 대한 개체 인스턴스들을 모아놓은 것





- 단순 속성
 - simple attribute
 - 의미를 더는 분해할 수 없는 속성
 - Ex. 책 개체의 이름, 가격, ISBN
- 복합 속성
 - composite attribute
 - 의미를 분해할 수 있는 속성
 - Ex. 고객 개체의 주소, 생년월일 속성

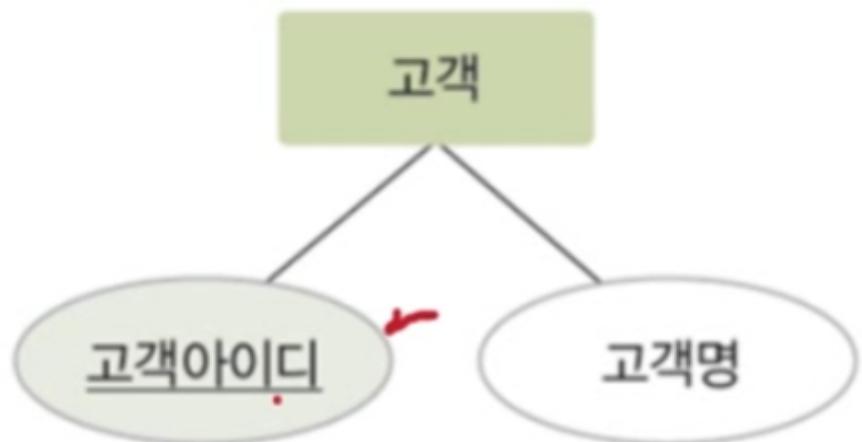


- 유도 속성
 - derived attribute
 - 기존의 다른 속성의 값에서 유도되어 결정되는 속성
 - 값이 별도로 저장되지 않음.
 - Ex. 책 개체의 가격과 할인율 속성으로 계산되는 판매 가격 속성

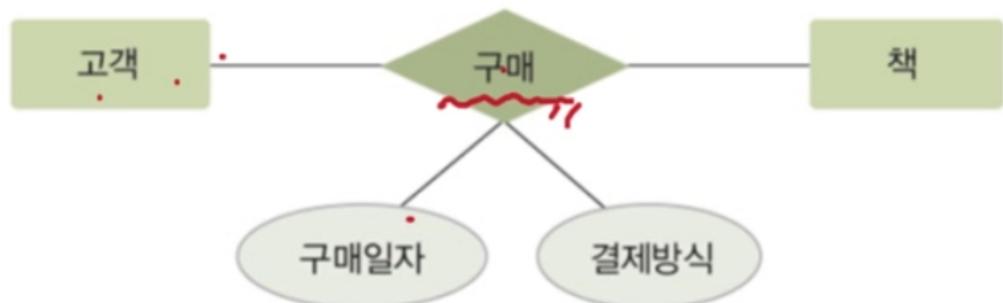


- ERD에서 점선 타원으로 표현
- 널 속성
 - null attribute
 - 널 값이 허용되는 속성
- 널 값
 - null
 - 아직 결정되지 않거나 모르는 값. 또는 존재하지 않는 값.
 - 공백이나 0과는 의미가 다름
 - Ex. 등급 속성이 null => 등급이 아직 결정되지 않음

- key attribute
 - 각 개체 인스턴스를 식별하는 데 사용되는 속성
 - 모든 개체 인스턴스의 키 속성 값이 다름
 - 둘 이상의 속성들로 구성되기도 함.
 - Ex. 고객 개체의 고객 아이디 속성
 - ERD에서 밑줄로 표현



- 관계
 - 개체와 개체가 맺고 있는 의미 있는 연관성
 - 개체 집합들 사이의 대응 관계, 즉 mapping
 - Ex. 고객 개체와 책 개체 간의 구매 관계
 - ERD에서 마름모로 표현



- 유형
 - 관계에 참여하는 개체 태입의 수 기준
 - 이항 관계

- 개체 탑 두 개가 맺는 관계
 - 삼항 관계
 - 개체 탑 세 개가 맺는 관계
 - 순환 관계
 - 개체 탑 하나가 자기 자신과 맺는 관계
 - mapping cardinality기준
 - 1:1
 - 1:n
 - n:m
- mapping cardinality
- 관계를 맺는 두 개체 집합에서, 각 개체 인스턴스가 연관성을 맺고 있는 상대 개체 집합의 인스턴스 개수
- 관계의 참여 특성
 - 전체 참여
 - 모든 개체 인스턴스가 관계에 반드시 참여
 - Ex. 고객 개체가 음료 개체와의 구매 관계에 필수적으로 참여
 - 모든 고객은 반드시 음료를 구매해야 함
 - ERD에서 이중 선으로 표현
 - 선택적 참여
 - 개체 인스턴스 중 일부만 관계에 참여해도 되는 것을 의미
 - Ex. 고객 개체가 음료 개체와의 구매 관계에 필수적으로 참여
 - 고객이 구매하지 않은 음료가 존재할 수 있음.



- 관계의 종속성
 - weak entity
 - 다른 개체의 존재 여부에 목숨이 달린 개체
 - owner entity
 - 다른 개체의 존재 여부를 결정하는 개체
 - 일반적으로 owner entity : weak entity = 1 : n
 - weak entity는 owner entity와의 관계에 필수적으로 참여
 - weak entity는 owner entity의 key를 포함하여 키를 구성하는 특징이 있음.
 - ERD에서 weak entity - 이중 사각형, owner entity 와 owner entity의 관계 - 이중 마름모
 - Ex.



관계 데이터 모델링
