Database

컴퓨터개론

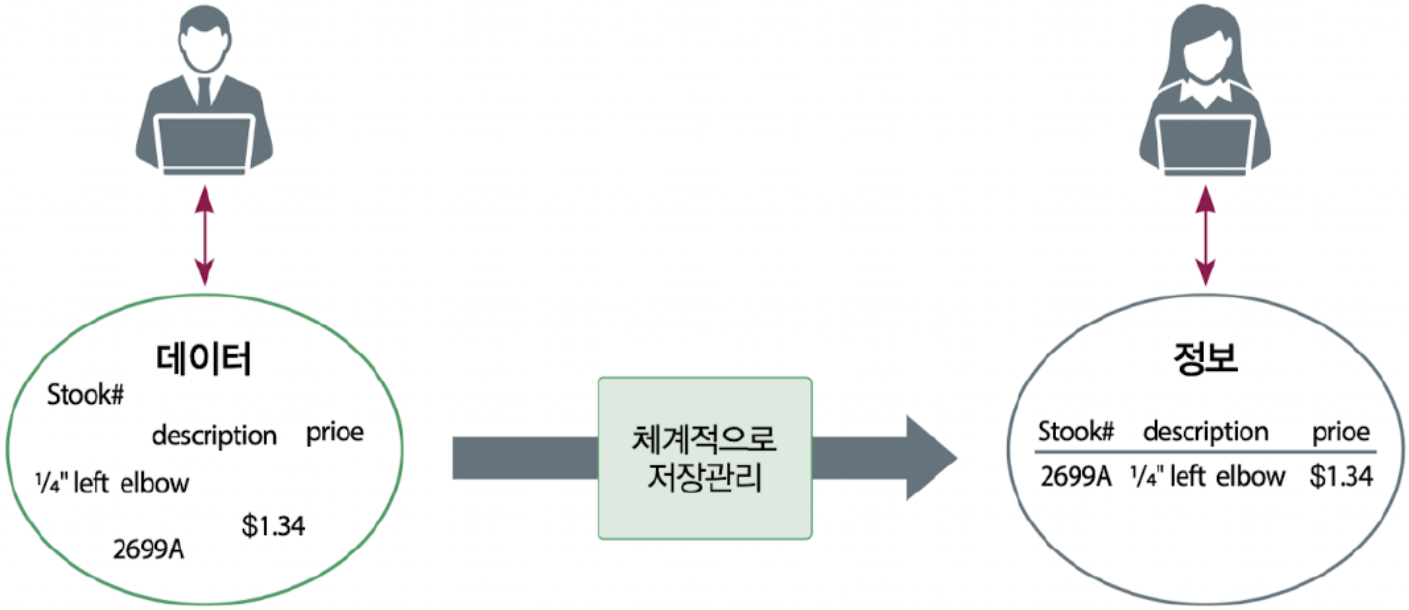
(Introduction to Computer Systems)

GEN1030

Database 개요

데이터와 정보

- 데이터(Data)
  - 단순한 사실에 불과한 아직 처리 되지 않은 값
- 정보(Information)
  - 데이터가 사람에게 유용한 의미로 쓰여질 수 있도록 처리된 값



데이터베이스

- Database
  - 관련 있는 데이터의 저장소
  - "여러 사람이나 응용시스템이 참조 가능하도록 서로 논리적으로 연관되어 통합 관리되는 데이터의 모임"
  - 데이터 추가, 공유, 찾기, 정렬, 분류, 요약, 출력 등의 조작 제공

- 특징
  - 통합되고 관련 있는 데이터 집합
  - 중복을 최소화하여 보조기억 장치에 저장
  - 무결성, 동시 접근, 보안 유지, 장애 회복 등 제공

데이터베이스 특징

- 빠른 접근성
  - 질의에 대해 빠르게 접근하고 처리할 수 있음
- 계속적 변화
  - 새로운 데이터의 삽입, 기존 데이터의 삭제 및 갱신 등을 통해 최신의 상태 유지
- 동시 접근 및 공유
  - 여러 사용자가 동시에 자료에 접근하더라도 문제없이 작업 수행
  - 요구에 맞게 여러가지 구조로 지원 가능
- 내용에 의한 참조
  - 수록되어 있는 데이터의 주소(위치)에 의해서가 아니라 데이터의 내용(데이터가 가지고 있는 값)에 따라 참조 됨

데이터베이스 특징

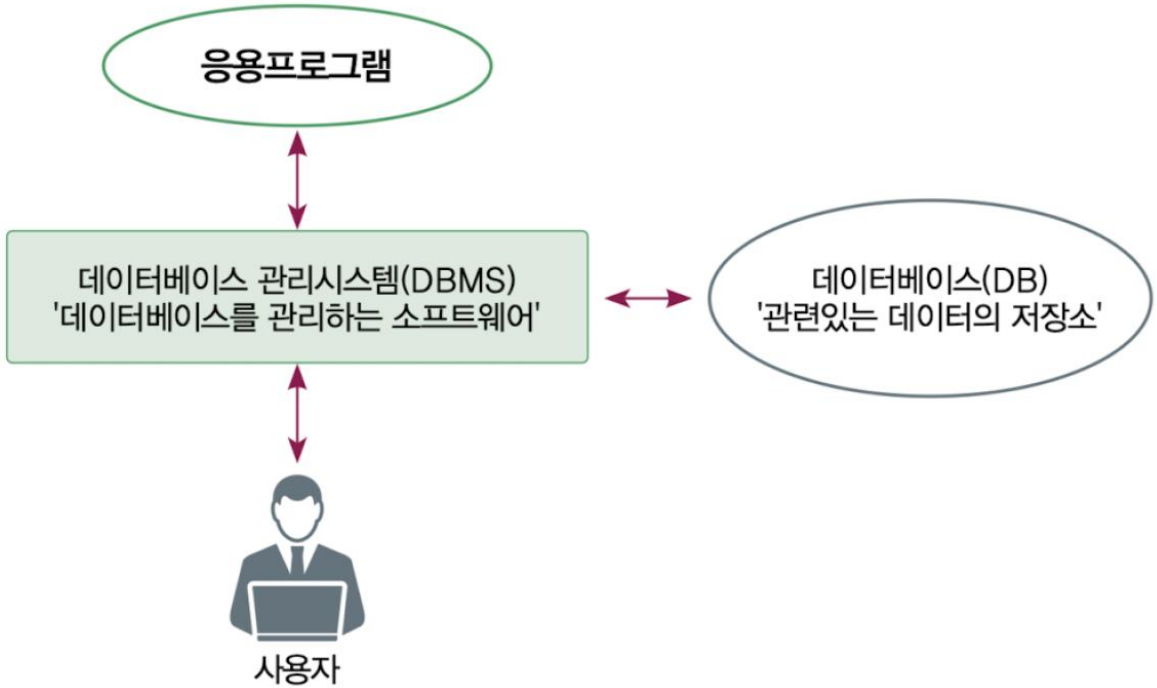
- 중복의 최소화
  - 동일한 내용의 데이터가 중복되지 않도록 함
  - 검색 및 갱신이 효율적으로 이루어질 수 있도록 중복 최소화
- 무결성(Integrity)
  - 저장된 데이터의 정확성을 유지
  - Ex) 허용하는 범위의 값만이 들어갈 수 있음, 데이터를 식별하는 기본키를 항상 가짐
- 보안 유지
  - 데이터베이스 접근을 관리 – 권한이 있는 사용자, 허용된 데이터의 연산만 허용

데이터베이스 특징

- 데이터 독립성
  - 응용프로그램과 데이터가 서로 종속되지 않음
  - 데이터베이스 구조가 바뀌어도 이를 사용하는 프로그램 수정 불필요
- 표준화
  - 정해진 규칙에 따라 데이터 관리하여 효율적인 공유 및 사용 가능
- 장애 회복
  - 문제가 발생하면 이전 상태로 복구 가능

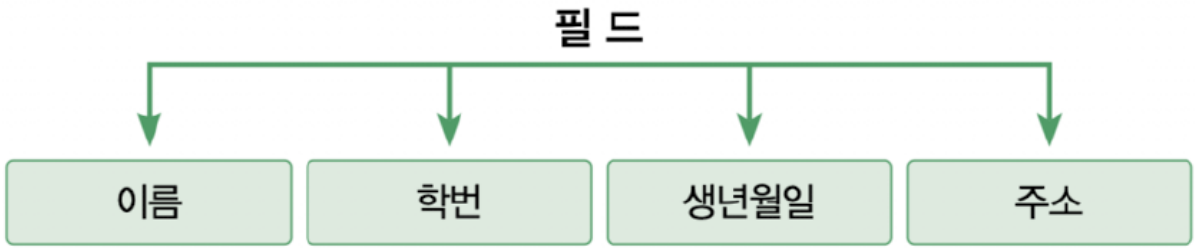
데이터베이스 관리시스템

- DataBase Management System (DBMS)
  - 사용자가 데이터베이스를 만들고 유지, 관리할 수 있도록 돕는 프로그램
  - 데이터와 응용 프로그램 사이 중재자 역할: 프로그램들이 데이터베이스를 유용하게 활용할 수 있도록 관리



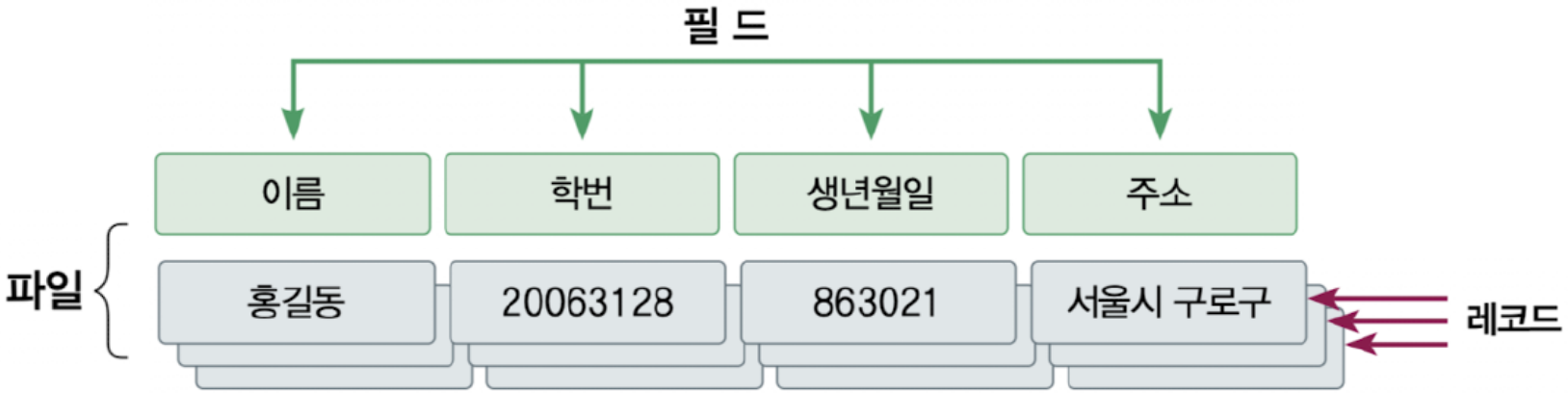
데이터베이스 구조

- 필드(Field)
  - 특정한 종류의 데이터를 저장하기 위한 영역
    - 특정한 종류? 필드에 저장될 수 있는 데이터 유형(Data type)
  - 논리적인 의미 있는 자료의 단위



데이터베이스 구조

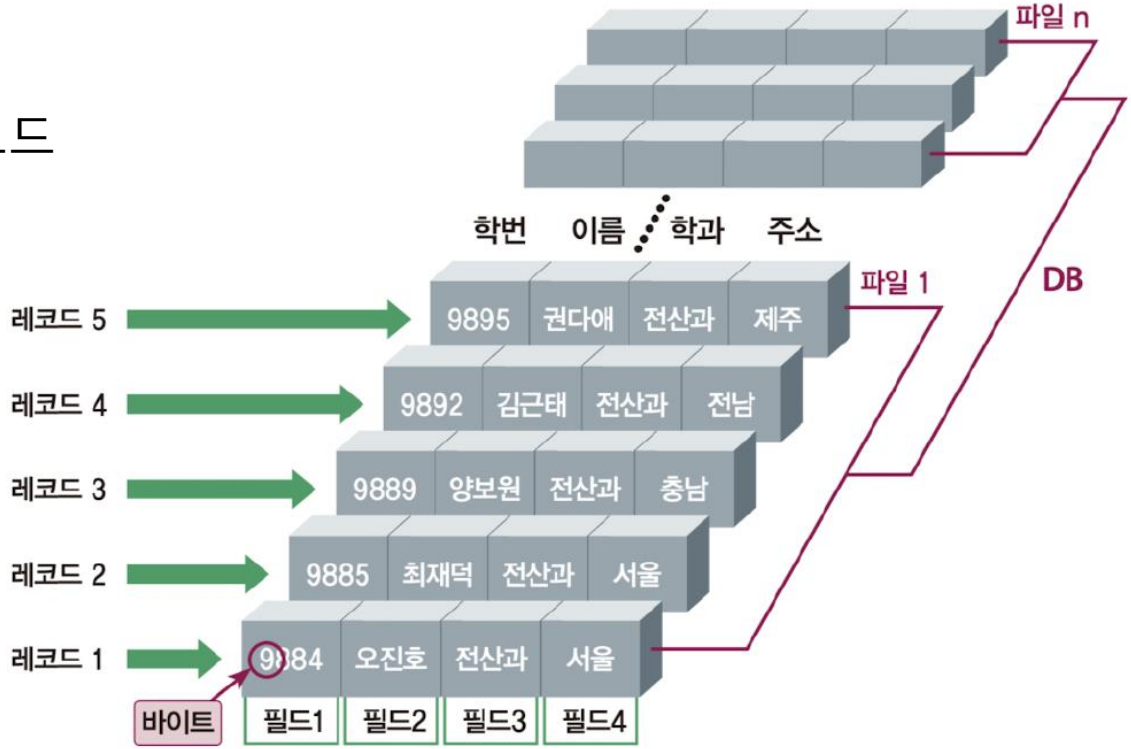
- 필드(Field)
  - 특정한 종류의 데이터를 저장하기 위한 영역
    - 특정한 종류? 필드에 저장될 수 있는 데이터 유형(Data type)
  - 논리적인 의미 있는 자료의 단위
- 레코드(Record)
  - 필드가 여러 개 모인 단위
- 파일(File)
  - 레코드가 여러 개 모인 단위



데이터베이스 구조

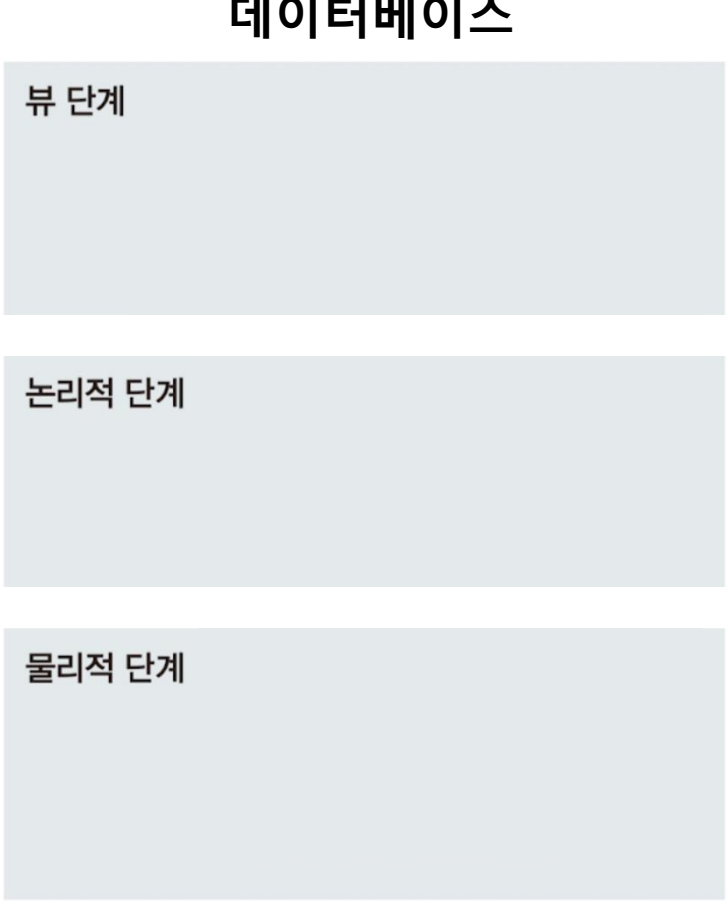
- 데이터베이스
  - 서로 관련 있는 데이터들로 통합된 파일의 집합
  - 파일이 여러 개 모여 데이터베이스 구성
    - 각 파일은 서로 다른 레코드 형태를 가질 수 있음

파일 1 각 레코드
- 학번
- 이름
- 학과
- 지역



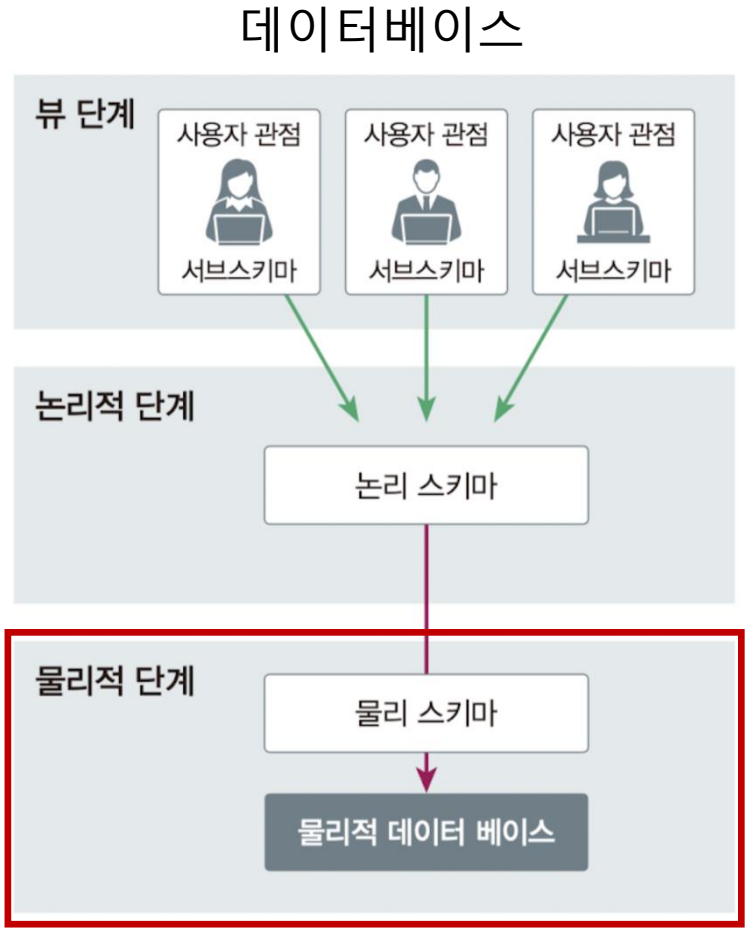
데이터베이스 추상화

- 데이터베이스 내부 구조는 상당히 복잡
- 3단계 추상화(Abstraction)
  - 뷰 단계(View level)
  - 논리적 단계(Logical level)
  - 물리적 단계(Physical level)
- 스키마(Schema)
  - 데이터베이스 전체적인 설계
  - 데이터베이스 구성하는 정보의 종류, 구조, 이들의 관계를 정의하는 구체적인 기술 및 명세
  - 데이터베이스 추상화의 세 단계에 각각 대응



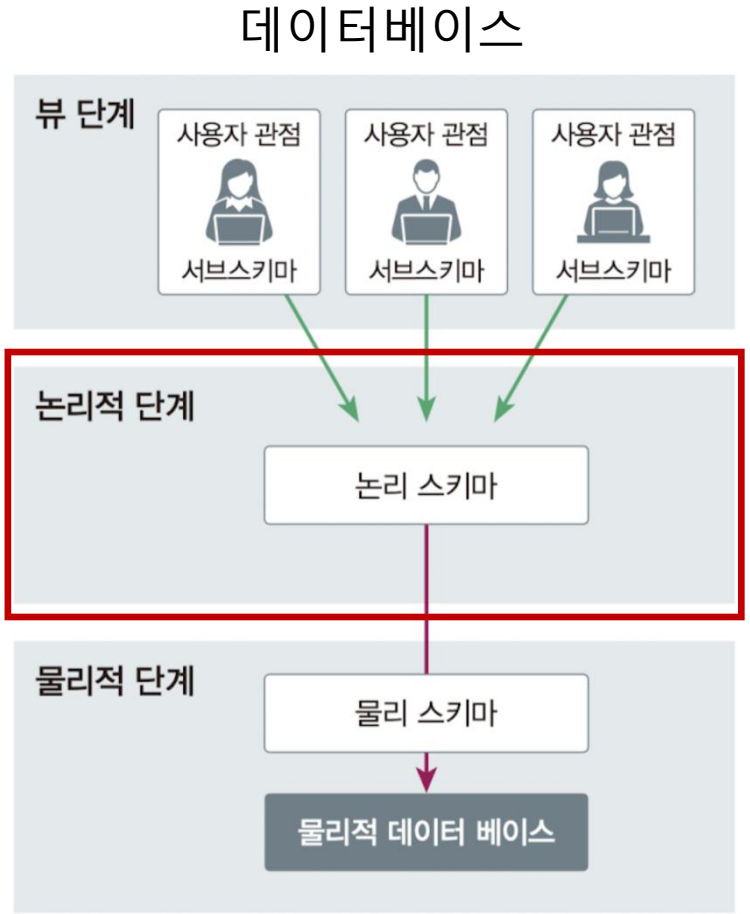
추상화

- 물리적 단계
  - 내부 단계(internal level)
  - 저장장치(e.g., HDD) 내부에 실질적으로 데이터가 저장될 구조와 위치 결정
  - 디스크 어느 위치에 저장되는지, 어떻게 배치할지, 압축 하는지 등
    - 하드웨어와 직접적인 상호 작용
  - 물리 스키마
    - 내부 스키마(internal schema)
    - 데이터베이스의 물리적 구조를 정의(기술)한 것



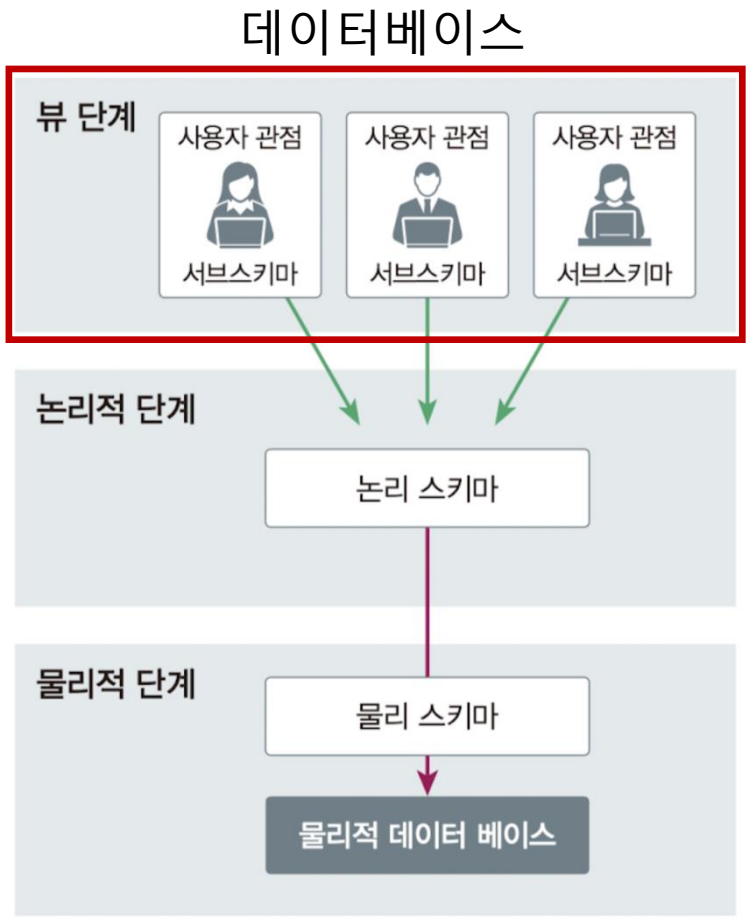
추상화

- 논리적 단계
  - 개념 단계(conceptual level)
  - 데이터베이스에 저장될 데이터 종류와 데이터 간의 관계를 기술
  - 데이터베이스 전체 구조를 사용자가 이해할 수 있는 형태로 설명하는 단계
  - 논리스키마
    - 개념 스키마(conceptual schema)
    - 데이터베이스 전체 논리적 구조를 정의한 것
    - Ex) '학생' 테이블에는 이름, 학번, 전공이 있음



추상화

- 뷰 단계
  - 외부 단계(external level)
  - 추상화 최상위 단계 → 사용자와 직접적인 상호작용
  - 사용자가 관심을 가지는 데이터만 제공
    - 같은 데이터베이스라도 사용자에 따라 보는 내용이 다를 수 있음
  - 서브스키마
    - 외부 스키마(external schema)
    - 개별 사용자나 응용프로그램이 볼 수 있는 데이터 구조 정의한 것
    - 하나의 데이터베이스에 여러 개의 서브스키마 존재 가능

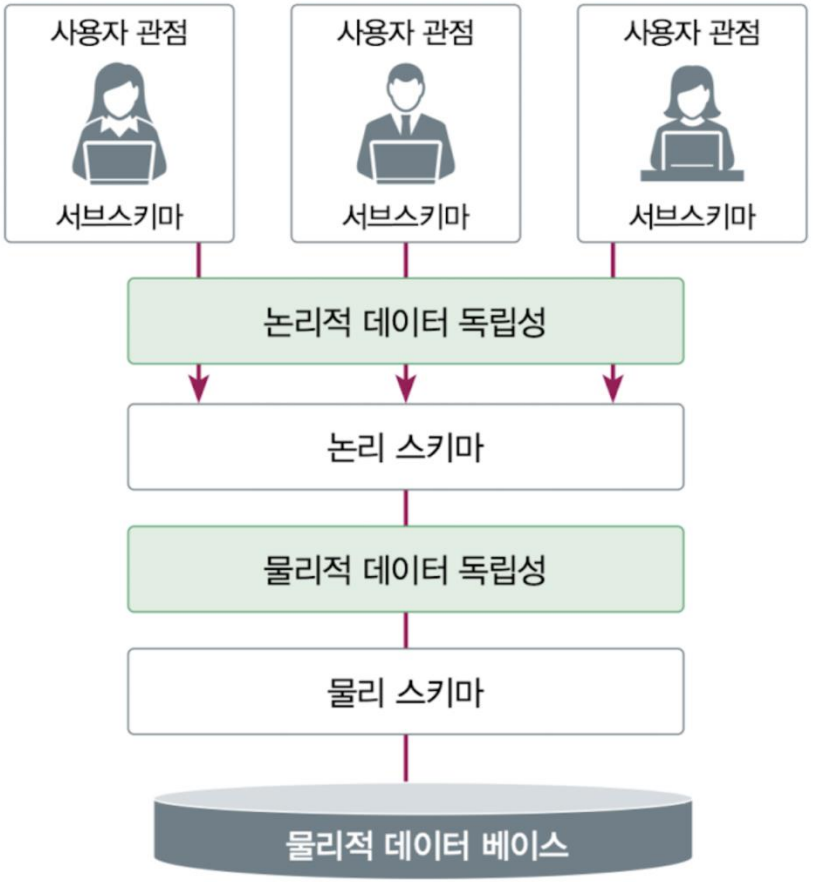


데이터 독립성(Data Independence)

- 추상화 과정에서 상위 수준의 스키마 정의에 영향을 주지 않고 해당 스키마 정의를 수정할 수 있는 능력

- 논리적 데이터 독립성
  - Logical data independence
  - 사용자의 응용프로그램에 영향 주지 않고 논리적 단계의 논리 스키마 수정 가능

- 물리적 데이터 독립성
  - Physical data independence
  - 응용프로그램이나 논리 스키마에 영향 주지 않고 물리적 스키마 수정 가능

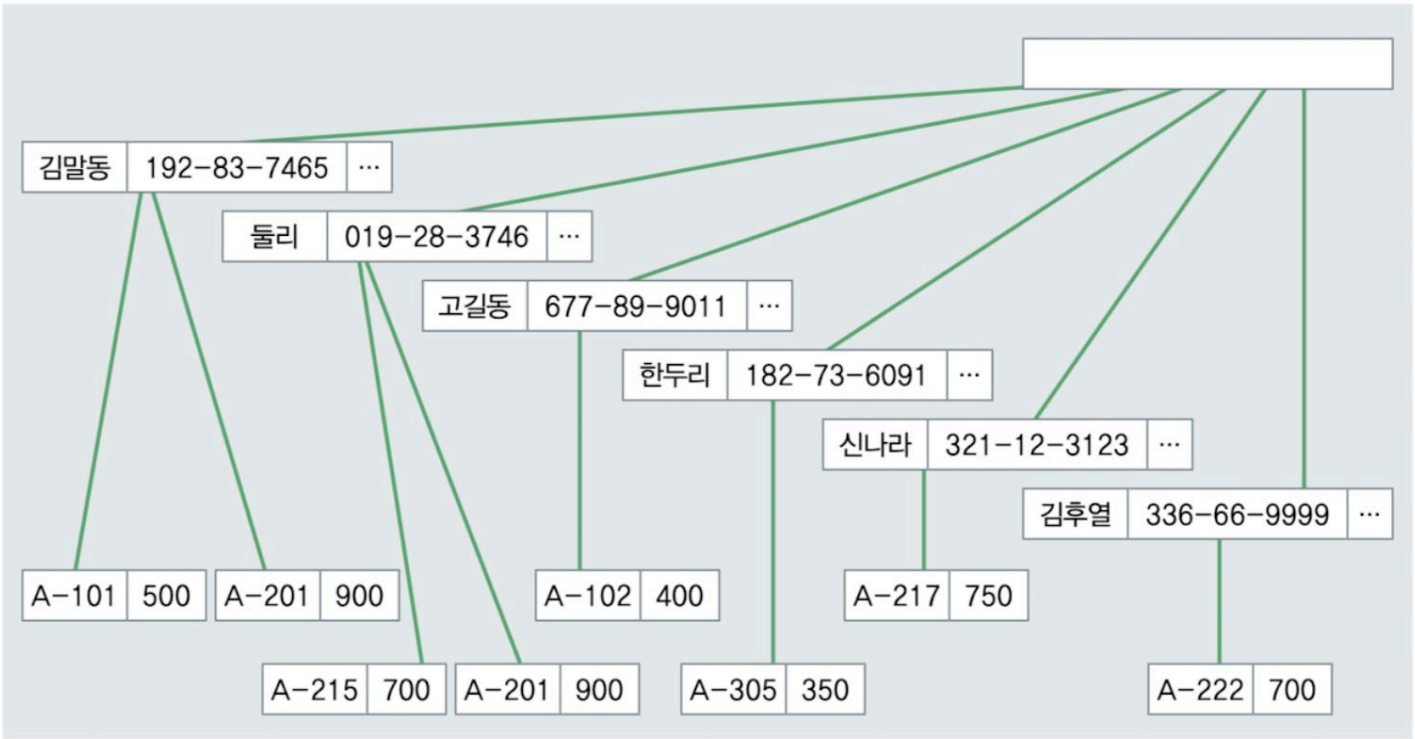


데이터베이스 모델

- 데이터의 논리적 설계와 그들 간의 관계를 표현
- 종류
  - 계층적 모델(Hierarchical model)
  - 네트워크 모델(Network model)
  - 관계형 모델(Relational model)

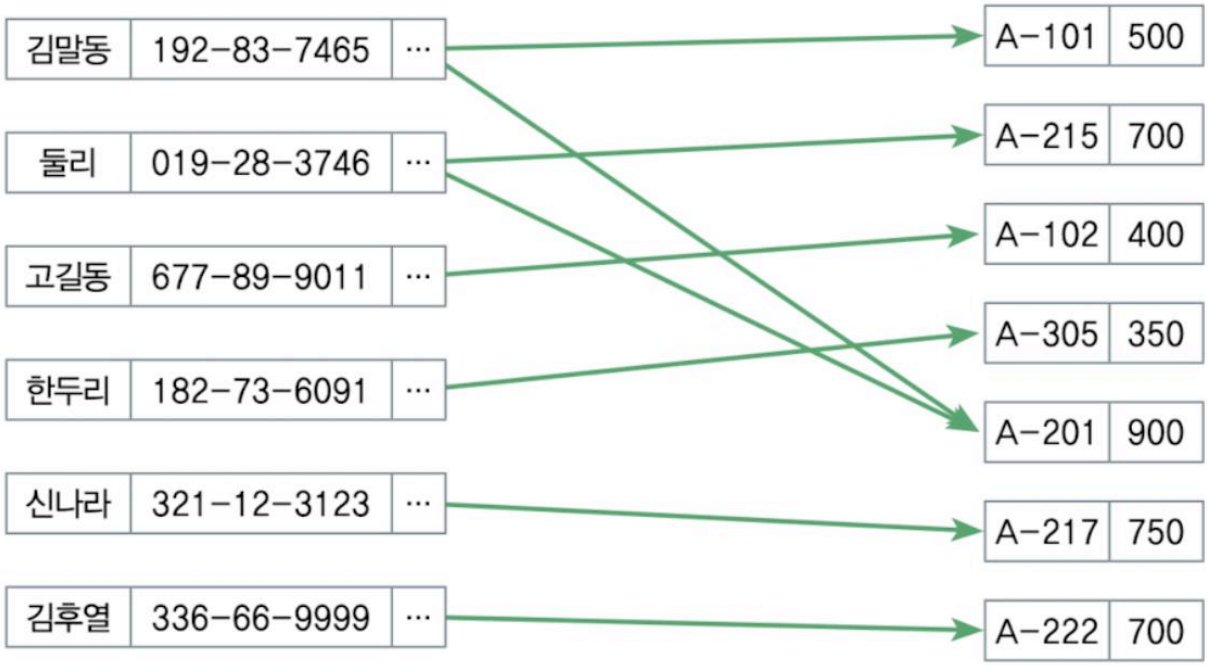
데이터베이스 모델

- 계층적 모델(Hierarchical model)
  - 데이터는 위에서 아래로 트리(Tree) 형태로 구성
  - 각 엔티티(entity)는 하나의 부모만 가짐
    - 부모 entity는 여러 자식 가질 수 있음



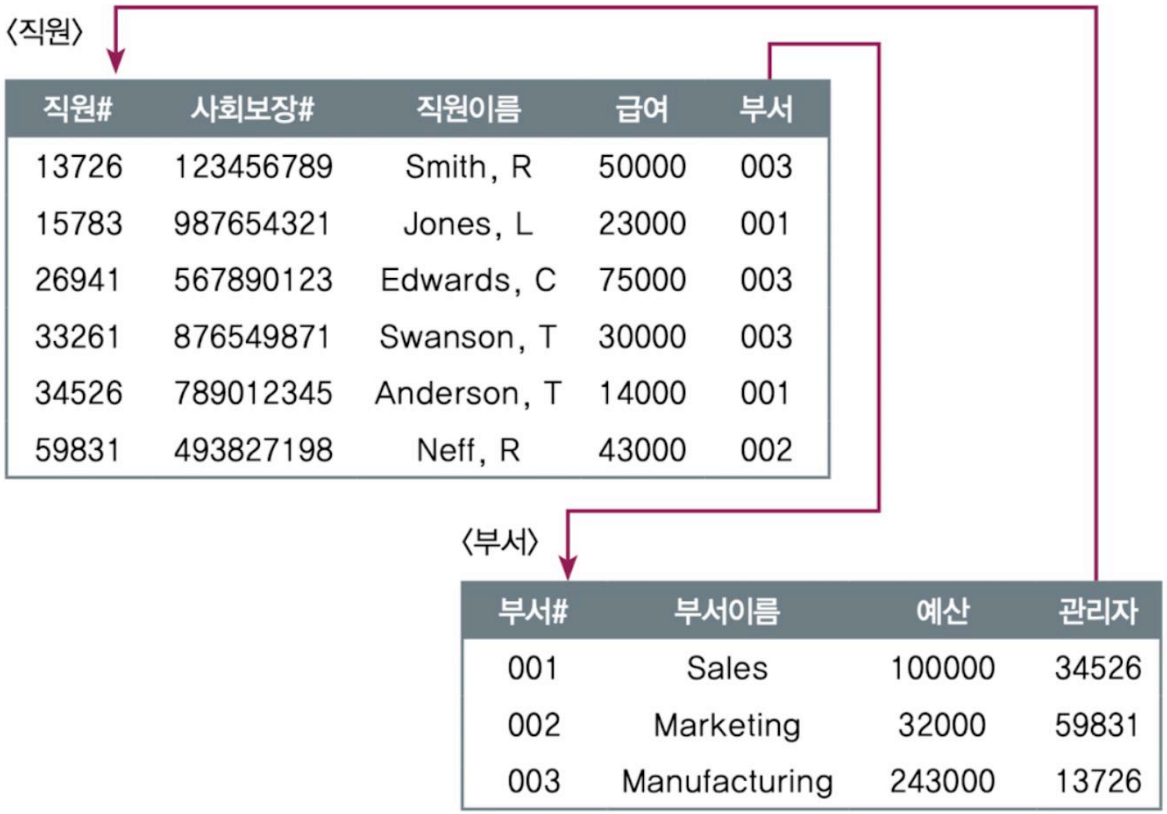
데이터베이스 모델

- 네트워크 모델(Network model)
  - 레코드(record)와 레코드 간의 관계를 서로 연결되는 그래프를 사용
  - 데이터 간의 관계는 링크(link)로 표현



데이터베이스 모델

- 관계형 모델(Relational model)
  - 데이터를 행과 열로 구성된 이차원 테이블의 집합으로 표현
  - 널리 활용되는 관계형 데이터베이스(Relational database)의 데이터 모델로 사용



〈직원〉

직원#	사회보장#	직원이름	급여	부서
13726	123456789	Smith, R	50000	003
15783	987654321	Jones, L	23000	001
26941	567890123	Edwards, C	75000	003
33261	876549871	Swanson, T	30000	003
34526	789012345	Anderson, T	14000	001
59831	493827198	Neff, R	43000	002

〈부서〉

부서#	부서이름	예산	관리자
001	Sales	100000	34526
002	Marketing	32000	59831
003	Manufacturing	243000	13726

데이터베이스 모델

관계형 모델(Relational Model)

- 모든 데이터를 2차원의 테이블(Table)로 표현
  - 2차원? 행(Row)과 열(Column)
  - 테이블을 관계(Relation)라고 함
- 관계 스키마(Relation Schema)
  - 관계(테이블) 구조 정의
  - 정적인 특성

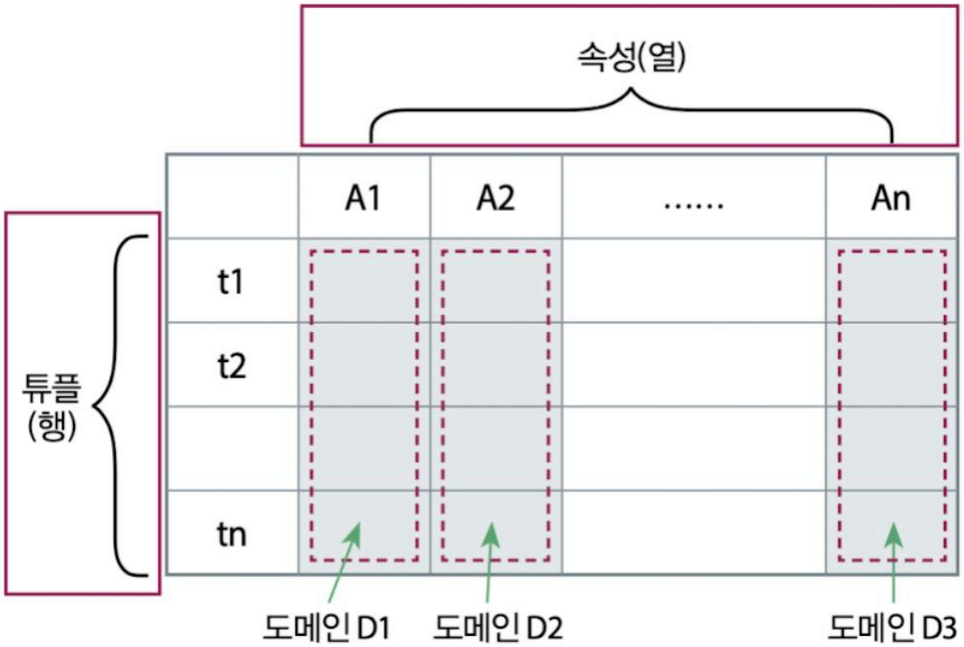
    관계/속성 이름 처음에 결정 후 자주 바뀌지 않음
- 관계 사례(Relation instance)
  - 관계 스키마에 들어있는 실제 데이터
  - 동적인 특성
    - 데이터 값 계속 변할 수 있음

학생

학번	이름	학과	주소	지도교수
2000001	오진호	001	서울	0001
2000002	권다애	002	경기도	0015
2000003	김근태	001	인천	0002
2000004	양보원	003	대전	0022
2000005	김태수	001	서울	0003

관계 구성 요소 – 속성, 튜플

- 속성(Attribute)
  - 관계에서 각 열
  - 한 테이블에서 속성 이름은 유일해야 함
  - 도메인(Domain)
    - 하나의 속성이 취할 수 있는 값의 범위
  - 차수(Degree)
    - 총 속성의 수
  - 데이터베이스에서 필드
- 튜플(Tuple)
  - 관계에서 각 행
  - 속성 값들의 집합
  - 데이터베이스에서 레코드



관계 구성 요소 – 속성, 튜플

- 속성 이름의 유일성
  - 한 관계에서 속성 이름은 유일해야 함
- 원자 값(Atomic value)
  - 튜플 내의 모든 값은 더 이상 나눌 수 없는 값이어야 함

학번	이름	학과
2026001	김OO	A
2026002	박OO	A, B

학번	이름	학과
2026001	김OO	A
2026002	박OO	A
2026002	박OO	B

관계 구성 요소 – 속성, 튜플

- 속성 간의 무순서
  - 한 관계에서 속성 간 순서는 무의미
- 튜플 간은 무순서
  - 관계에서 튜플 간 순서는 무의미

학번	이름	학과
2026001	김OO	A
2026002	박OO	B

학번	학과	이름
2026001	A	김OO
2026002	B	박OO

학번	학과	이름
2026002	B	박OO
2026001	A	김OO

관계 구성 요소 – 키

- 속성 간의 무순서
  - 한 관계에서 속성 간 순서는 무의미
- 튜플 간은 무순서
  - 관계에서 튜플 간 순서는 무의미
- 중복 불허
  - 한 관계에서 두 튜플의 속성값이 모두 같은 것 허용하지 않음

학번	이름	학과
2026001	김OO	A
2026002	박OO	B
2026002	박OO	B

관계 구성 요소 – 키

- 키(Key)
  - 튜플들을 유일하게(uniqueness)하게 구별할 수 있는 하나 이상의 속성의 집합
  - 한 테이블에 삽입될 수 있는 튜플은 반드시 키 값이 달라야 함

- 종류
  - Candidate key (후보키)
  - Primary key (주키)
  - Foreign key (외래키)

관계 구성 요소 – 키

- **Candidate key (후보키)**
  - 한 행(row) 유일하게 구별할 수 있는 속성 또는 속성들의 조합
  - 하나의 관계에서 유일성과 최소성(minimality) 만족하는 키
  - Primary key가 될 수 있는 후보들
- Primary key (주키)
- Foreign key (외래키)

학생

학번	이름	학과	주소	지도교수
2000001	오진호	001	서울	0001
2000002	권다애	002	경기도	0015
2000003	김근태	001	인천	0002
2000004	양보원	003	대전	0022
2000005	김태수	001	서울	0003

관계 구성 요소 – 키

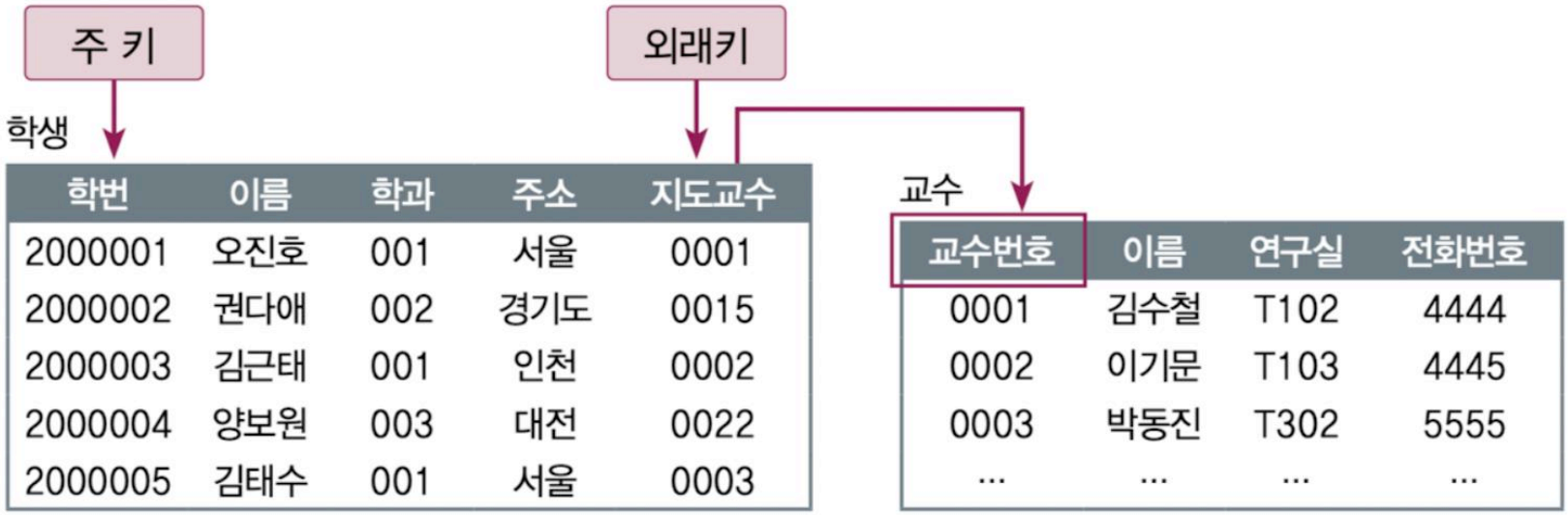
- Candidate key (후보키)
- **Primary key (주키)**
  - 관계에서 여러 튜플 중에서 하나의 튜플을 식별하는 역할 수행
  - Candidate key 중 대표 식별자로 선택된 키
  - 중복될 수 없고 비어있을 수 없음(NULL 불가능)
- Foreign key (외래키)

학생

학번	이름	학과	주소	지도교수
2000001	오진호	001	서울	0001
2000002	권다애	002	경기도	0015
2000003	김근태	001	인천	0002
2000004	양보원	003	대전	0022
2000005	김태수	001	서울	0003

관계 구성 요소 – 키

- Candidate key (후보키)
- Primary key (주키)
- **Foreign key (외래키)**
  - 다른 관계(테이블)의 primary key를 참조하는속성
  - 관계(테이블)와 관계(테이블) 서로 연결할 수 있음



관계 연산

- 2차원 테이블에 대한 연산
  - 삽입(Insert)
  - 삭제(Delete)
  - 수정(Update)
  - 조회(Select)

관계 연산

- 삽입(INSERT)
  - 튜플을 테이블에 삽입
  - 튜플의 순서 의미 없으므로 삽입되는 튜플 순서 상관하지 않음

(20153007,
김 남훈,
19961018,
010-5948-1234)
삽입

학생 관계

학번	이름	생년월일	핸드폰번호
20153001	이 종만	19970427	011-7384-0000
20153002	오 상조	19960717	010-6594-0001
20153003	남 승현	19960506	016-7285-0002
20153004	조 진형	19961110	011-3454-0003

삽입
(insert)

학생 관계

학번	이름	생년월일	핸드폰번호
20153001	이 종만	19970427	011-7384-0000
20153002	오 상조	19960717	010-6594-0001
20153003	남 승현	19960506	016-7285-0002
20153004	조 진형	19961110	011-3454-0003
20153007	김 남훈	19961018	010-5948-1234

관계 연산

- 삭제(DELETE)
  - 테이블에서 관련된 튜플을 삭제

핸드폰번호
011로 시작하는
튜플 삭제

학생 관계

학번	이름	생년월일	핸드폰번호
20153001	이 종만	19970427	011-7384-0000
20153002	오 상조	19960717	010-6594-0001
20153003	남 승현	19960506	016-7285-0002
20153004	조 진형	19961110	011-3454-0003

삭제
(delete)

학생 관계

학번	이름	생년월일	핸드폰번호
20153002	오 상조	19960717	010-6594-0001
20153003	남 승현	19960506	016-7285-0002

관계 연산

- 수정(UPDATE)
  - 테이블에서 관련된 속성 값을 수정

학번이 20153004 학생 생년월일을 19891010으로 수정

학생 관계

학번	이름	생년월일	핸드폰번호
20153001	이 종만	19970427	011-7384-0000
20153002	오 상조	19960717	010-6594-0001
20153003	남 승현	19960506	016-7285-0002
20153004	조 진형	19961110	011-3454-0003

수정 (update)

학생 관계

학번	이름	생년월일	핸드폰번호
20153001	이 종만	19970427	011-7384-0000
20153002	오 상조	19960717	010-6594-0001
20153003	남 승현	19960506	016-7285-0002
20153004	조 진형	19891010	011-3454-0003

관계 연산

- 조회(SELECT)
  - 테이블의 부분 집합 튜플/속성으로 구성된 새로운 테이블 생성
  - 테이블의 특정 속성만 지정 가능

태어난 연도가 1996인 학생의 **학번**, **이름**, **생년월일** 조회

학생 관계

학번	이름	생년월일	핸드폰번호
20153001	이 종만	19970427	011-7384-0000
20153002	오 상조	19960717	010-6594-0001
20153003	남 승현	19960506	016-7285-0002
20153004	조 진형	19961110	011-3454-0003

조회
(select)

조회된 새로운 관계

학번	이름	생년월일
20153002	오 상조	19960717
20153003	남 승현	19960506
20153004	조 진형	19961110

객체-관계형 모델

- Object-Relational Model
  - 관계형 모델의 확장 - 관계형 모델에 객체지향 개념을 추가
  - 기본은 관계형 데이터베이스이며, 여기에 객체지향 프로그래밍에서 쓰는 개념들을 지원

  - 사용자 정의 타입(User-defined type)
    - 정수, 문자 같은 기본 타입 외에도 사용자가 직접 복잡한 데이터 타입 정의 가능 (ex. 주소, 좌표)
  - 상속(Inheritance)
    - 한 테이블이 다른 테이블의 특성을 물려 받을 수 있음
  - 메소드(Methods)
    - 데이터베이스 안에 특정 데이터를 처리하는 함수를 포함할 수 있음

비관계형 데이터베이스

- NoSQL (Not Only SQL)
  - 고정된 표(Table)라는 틀에 얽매이지 않는 방식으로 데이터를 저장
  - 모든 데이터가 표 형태로 깔끔하게 표현되지 않음
    - 이미지, 영상 등 다양하고 용량이 큰 비정형 데이터의 증가
- 종류
  - Key-Value 데이터베이스
  - Document 데이터베이스
  - Column-family 데이터베이스
  - Graph 데이터베이스

비관계형 데이터베이스

- Key-Value 데이터베이스
  - 데이터를 key와 value 쌍으로 저장
  - 가장 단순한 형태로 읽기/쓰기 빠름

    Key: “user1”
    Value: “name”: Park, “age”: 20, …

- Document 데이터베이스
  - JSON 등의 document 형태로 데이터를 저장

```
{
   "id": 1,
   "name": "Kim",
   "major": "CS",
   "email": "kim@univ.edu",
   "clubs": ["soccer", "music"]
}
```

비관계형 데이터베이스

- Column-family 데이터베이스
  - Row마다 가지고 있는 column 내용이 다를 수 있음

User A name: Kim email: kim@univ.com job: Student

User B name: Park mbti: enfp mobile: 010-0000-0000

- Graph 데이터베이스
  - 데이터를 노드(node)와 엣지(edge)로 표현

Summary

- 데이터베이스
  - 관련 있는 데이터의 저장소
  - 구조
    - 필드, 레코드, 파일
  - 추상화 3단계
    - 뷰
    - 논리적
    - 물리적
  - 관계형 데이터베이스
    - 특징
    - 구성 요소