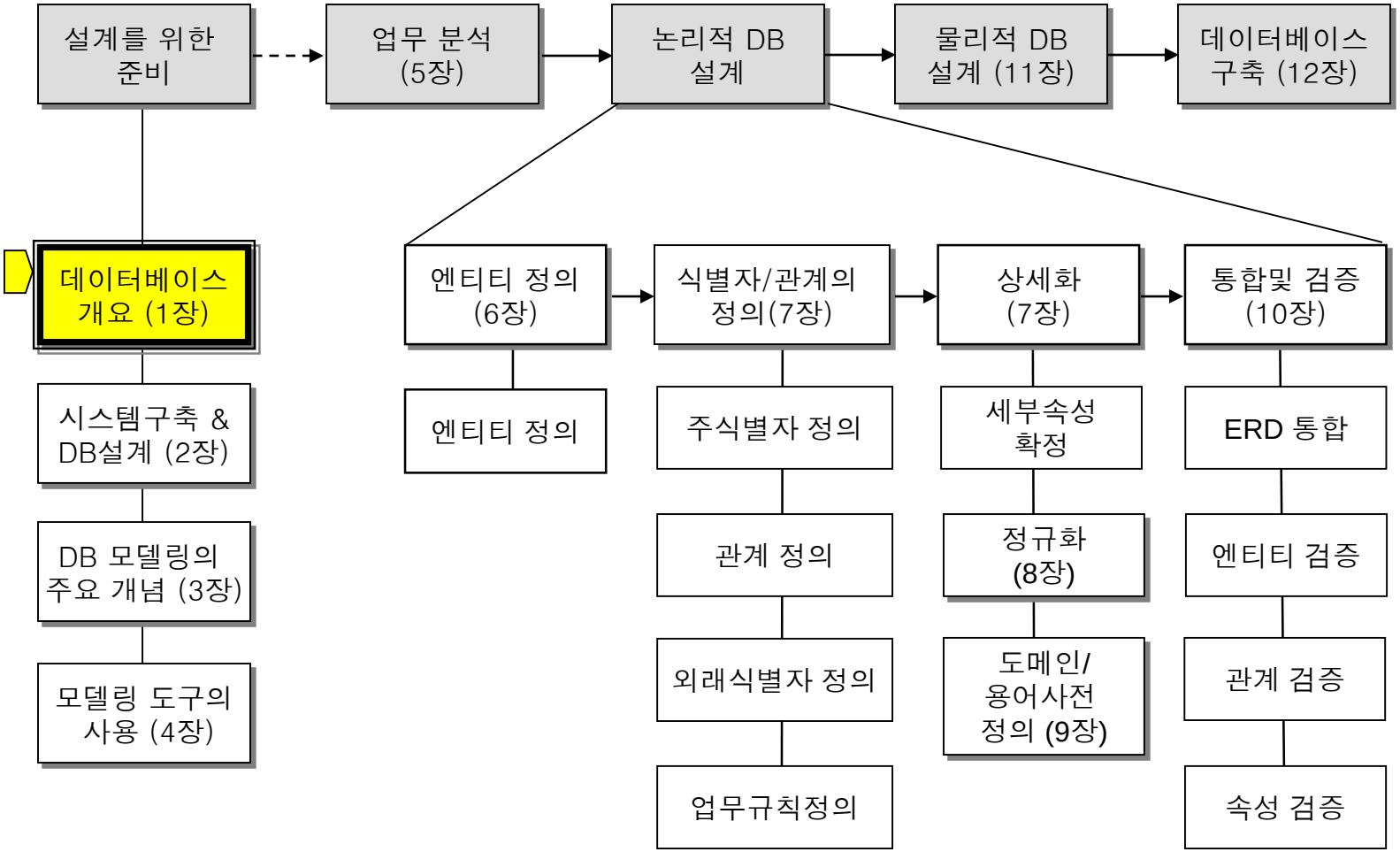
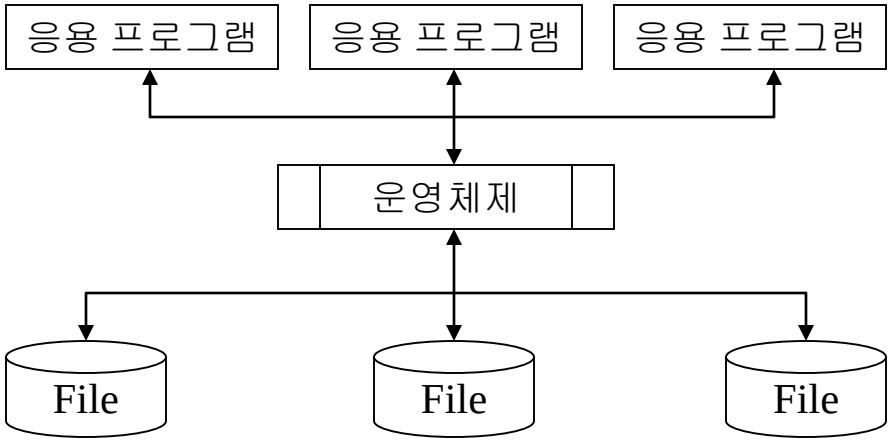
1장. 관계형 데이터베이스의 주요 개념

- ❑ 데이터베이스의 역사
- ❑ 관계형 데이터베이스 용어
- ❑ 기본키와 외래키


설계를 위한 준비
업무 분석 (5장)
논리적 DB 설계
물리적 DB 설계 (11장)
데이터베이스 구축 (12장)
데이터베이스 개요 (1장)
시스템구축 & DB설계 (2장)
DB 모델링의 주요 개념 (3장)
모델링 도구의 사용 (4장)
엔티티 정의 (6장)
엔티티 정의
식별자/관계의 정의(7장)
주식별자 정의
관계 정의
외래식별자 정의
업무규칙정의
상세화 (7장)
세부속성 확정
정규화 (8장)
도메인/ 용어사전 정의 (9장)
통합및 검증 (10장)
ERD 통합
엔티티 검증
관계 검증
속성 검증

1.1 데이터베이스의 역사

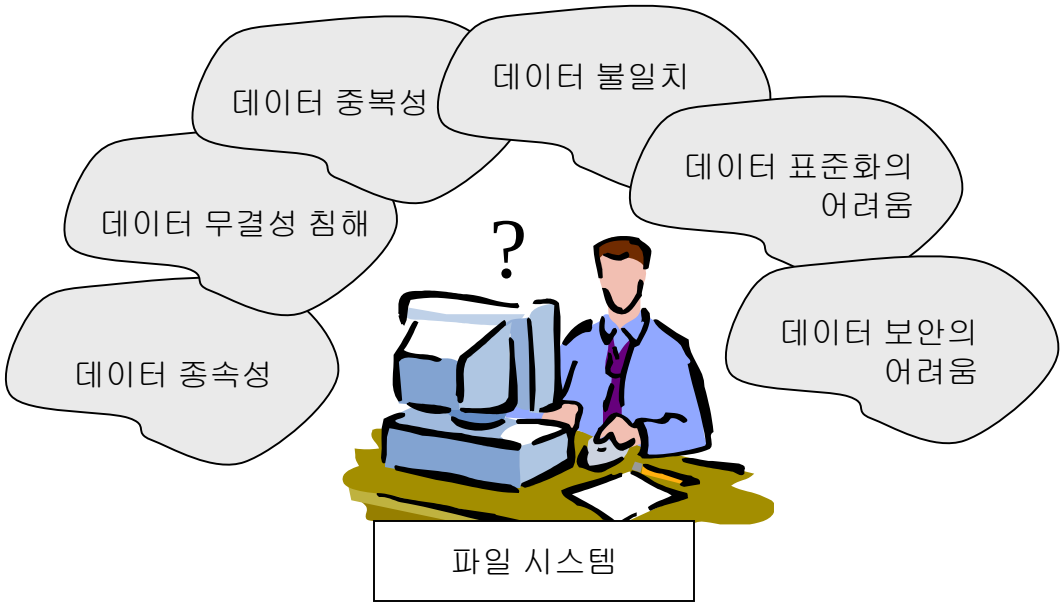
❑ 파일 시스템의 위기



<그림 1.1> 파일 시스템에 기초한 자료처리 모델

1.1 데이터베이스의 역사

❑ 파일 시스템의 위기



<그림 1.3> 파일 시스템의 위기

1.1 데이터베이스의 역사

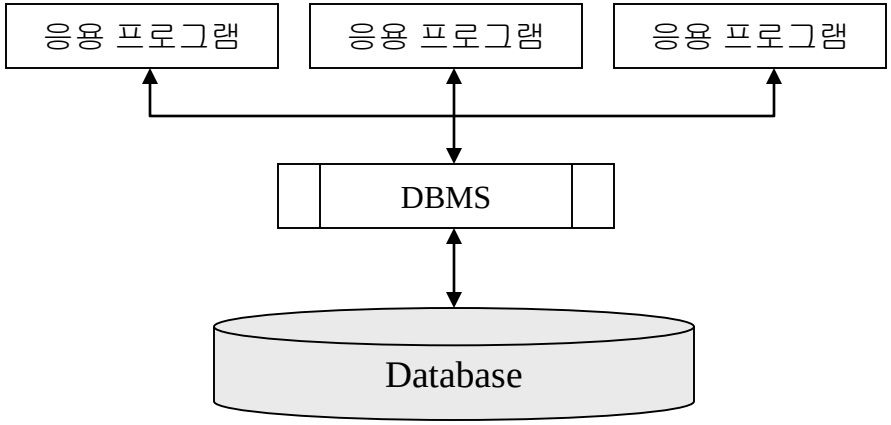
❑ 데이터베이스의 등장

- 파일 시스템의 단점을 극복하면서도 다수의 사용자들이 정보를 공유할 수 있어야 한다는 요구에 따라 제안됨

- 데이터베이스의 철학
  - 파일 형태로 여기저기에 흩어져 있는 데이터, 정보들을 하나로 모아 관리
  - 모아놓은 데이터들을 관리하고 사용자(응용 프로그램)와 데이터 사이에 인터페이스 역할을 할 수 있는 S/W를 제공

1.1 데이터베이스의 역사

❑ 데이터베이스의 등장

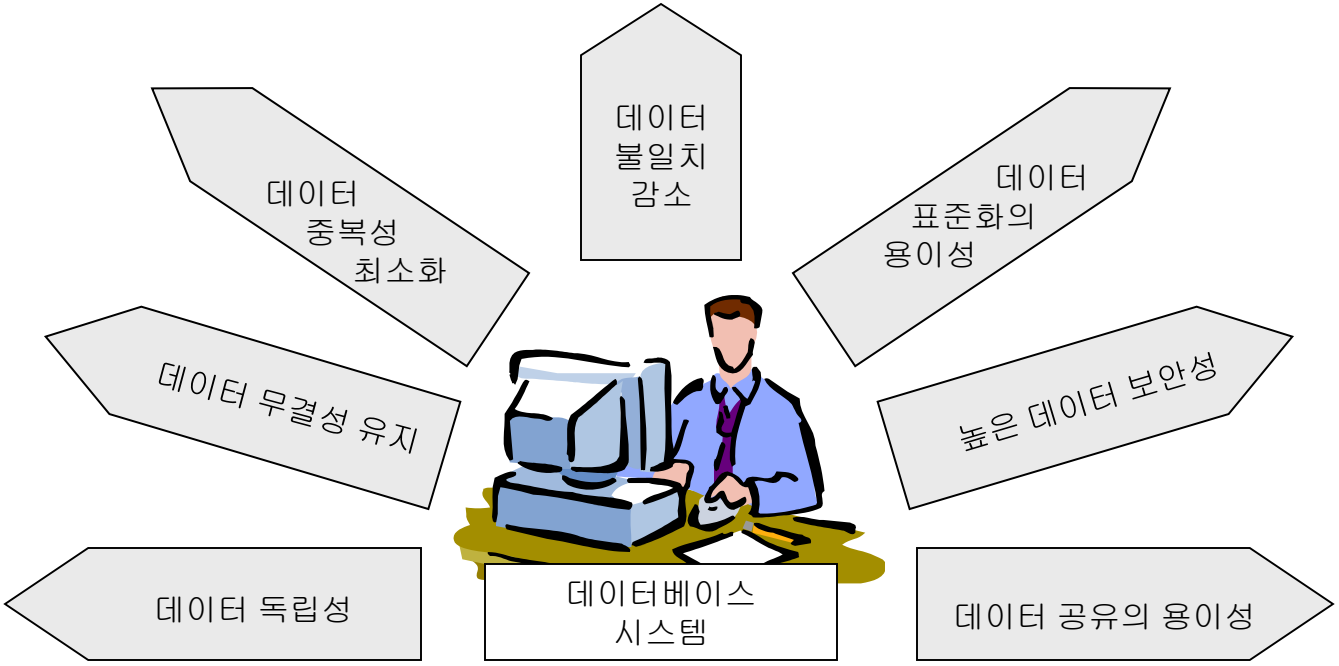


<그림 1.4> 데이터베이스 시스템의 개요

✓ 모아놓은 데이터의 집합 : 데이터베이스(database)
✓ 데이터를 관리하는 S/W : 데이터베이스 관리 시스템
(DBMS; Database Management System)

1.1 데이터베이스의 역사

❑ 데이터베이스의 등장



<그림 1.5> 데이터베이스 시스템의 장점

1.1 데이터베이스의 역사

❑ 관계형 데이터베이스 모델

- 현재 가장 많이 사용되는 데이터베이스 모델
- 데이터가 테이블 형태로 표현
- 질의어(SQL)를 제공
- 테이블 형태로 표현된 데이터는 단순해서 누구나 쉽게 이해할 수 있음
- SQL 명령어나 문법은 표준화 되어 있기 때문에 대부분의 명령어는 모든 관계형 데이터베이스 제품에서 공통적으로 사용 가능

1.1 데이터베이스의 역사

❑ 관계형 데이터베이스 모델

EMPLOYER

empno	ename	dept	tel	salary
100	김기훈	영업	1241	200
101	홍성범	기획	5621	200
102	이만수	영업	5251	250
103	강나미	생산	1231	300

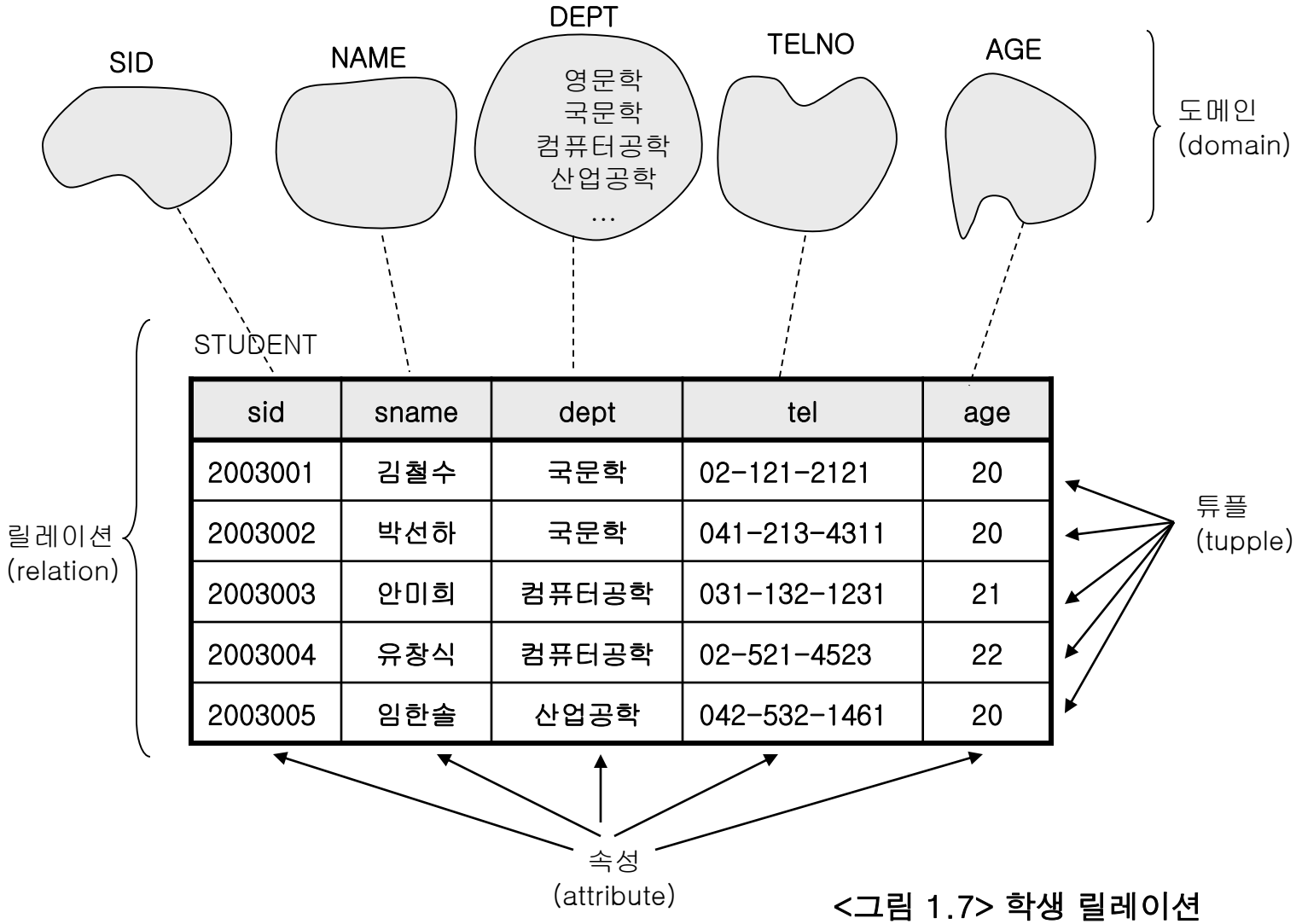
"영업부에 속한 모든 사원의 이름과 전화 번호를 보이시오"

⇒

```
SELECT ename, tel
FROM   employer
WHERE  dept = '영업'
```

<그림 1.6> 사원 테이블과 질의의 예

1.2 관계형 데이터베이스 용어



sid	sname	dept	tel	age
2003001	김철수	국문학	02-121-2121	20
2003002	박선하	국문학	041-213-4311	20
2003003	안미희	컴퓨터공학	031-132-1231	21
2003004	유창식	컴퓨터공학	02-521-4523	22
2003005	임한솔	산업공학	042-532-1461	20

<그림 1.7> 학생 릴레이션

1.2 관계형 데이터베이스 용어

※ <그림 1.7>에서 사용한 용어들은 E.F.Codd가 정의한 것으로 오늘날 일반적으로 사용하는 용어와는 차이가 있음

- 릴레이션(relation)
  - 테이블이라는 용어로 더 많이 사용
  - 관계형 데이테베이스에서 정보를 구분하여 저장하는 기본 단위

1.2 관계형 데이터베이스 용어

- 속성(attribute)
  - 릴레이션에서 관리하는 구체적인 정보 항목에 해당하는 것이 속성
  - 현실세계의 개체(예: 학생, 교수, 과목,...)들은 많은 속성들을 갖는데 그중에서 관리해야할 필요가 있는 속성들만을 선택하여 릴레이션에 포함시킴
  - 속성 역시 고유한 이름을 갖으며 동일 릴레이션 내에서는 같은 이름의 속성이 존재할 수 없음

- 튜플(tuple)
  - 릴레이션이 현실세계의 어떤 개체를 표현한다면 튜플은 그 개체에 속한 구성원들 개개의 정보를 표현
  - 한 릴레이션에 포함된 튜플의 개수는 시간에 따라 변할 수 있으며, 한 릴레이션은 적게는 수십 개 많게는 수십만 개의 튜플을 포함할 수 있음

1.2 관계형 데이터베이스 용어

- 도메인(domain)
  - 도메인이란 릴레이션에 포함된 각각의 속성들이 갖을 수 있는 값들의 집합
  - 예)'성별'이라는 속성이 있다면 이 속성이 가질 수 있는 값은 {남,여}.
  - 현실적으로 도메인을 구현하는 것은 어렵기 때문에 대부분의 DBMS 제품에서는 사용자 정의 데이터타입으로 사용

1.2 관계형 데이터베이스 용어

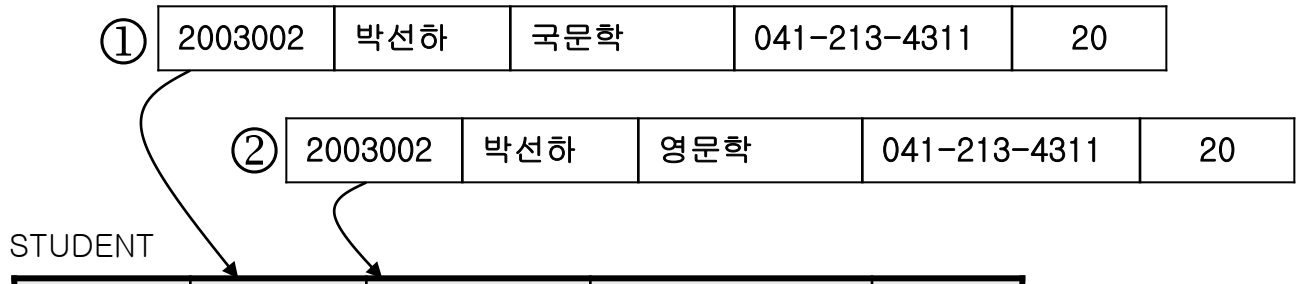
※ 현재는 여러 용어가 혼용되고 있는 상황임

E.F.Codd 의 용어	File 시스템의 용어	자주 사용되는 용어
릴레이션(relation)	파일(file)	테이블(table)
속성(attribute)	필드(field)	열(column), 컬럼
튜플(tuple)	레코드(record)	행(row)

<표 1.2> 용어 대비표

1.3 기본키와 외래키

❑ 키의 필요성



STUDENT

sid	sname	dept	tel	age
2003001	김철수	국문학	02-121-2121	20
2003002	박선하	국문학	041-213-4311	20
2003003	안미희	컴퓨터공학	031-132-1231	21
2003004	유창식	컴퓨터공학	02-521-4523	22
2003005	임한솔	산업공학	042-532-1461	20

<그림 1.8> 중복된 튜플의 삽입

1.3 기본키와 외래키

- 키의 필요성
  - 레코드 순서는 의미가 없으므로 레코드가 저장된 위치가 아닌 레코드 값으로 식별해야함
  - 중복 여부를 효과적으로 알 수 있도록 하는 수단이 '키(key)' 임

1.3 기본키와 외래키

- 후보키(candidate key)
  - 후보키(candidate key)란 테이블에서 각 튜플을 구별하는데 기준이 되는 하나 혹은 그 이상의 컬럼들의 집합이다. (후보키는 테이블에 있는 각 튜플을 고유하게 식별할 수 있어야 한다).
  - <그림 1.8>의 STUDENT 테이블의 경우 학번(sid)이 후보키
  - 튜플의 중복 여부 확인시 기존 튜플의 모든 컬럼값을 비교하는 대신 후보키 컬럼의 값만 비교

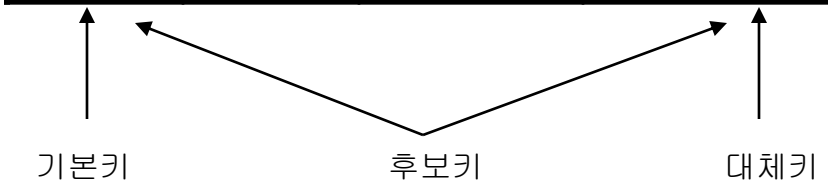
1.3 기본키와 외래키

❑ 후보키(candidate key)

➢ 후보키, 기본키, 대체키

STUDENT

sid	sname	dept	pid	age
2003001	김철수	국문학	831212-1213112	20
2003002	박선하	국문학	830823-2130121	20
2003003	안미희	컴퓨터공학	820224-2013112	21
2003004	유창식	컴퓨터공학	810509-1934142	22
2003005	임한솔	산업공학	831227-1324123	20



<그림 1.9> 후보키, 기본키, 대체키

1.3 기본키와 외래키

- 후보키(candidate key)
  - 기본키(primary key)
    - 후보키 중 튜플을 식별하는데 기준으로 사용할 키
  - 대체키(alternate key)
    - 후보키중 기본키로 선택되지 않은 나머지 키

1.3 기본키와 외래키

- 후보키(candidate key)
  - 복합키(composite key)
    - 하나의 컬럼이 후보키의 역할을 하지 못하고 두개 이상의 컬럼이 합쳐져야 후보키의 역할을 하는 경우

STUDENT_CLUB

sid	club	club_president
2003001	영어회화반	강우혁
2003001	낚시회	김민우
2003002	영어회화반	강우혁
2003003	한문강독	박문길
2003004	해커스	안홍섭

<그림 1.10> 복합키의 예

1.3 기본키와 외래키

- 외래키(Foreign key)
  - 상호 관련이 있는 테이블들 사이에서 데이터의 일관성을 보장해 주는 수단이 외래키 이다.
  - 다음 슬라이드에서 사원 테이블의 부서번호(deptid)는 부서정보 테이블의 부서번호(deptid)를 참조하고 있음.
    - 부서 테입블의 첫번째 튜플이 삭제된다면?
    - 부서테이블의 100번 부서 번호가 110 으로 변경된다면?
    - 사원 테이블에 부서 번호 500인 사원이 삽입된다면?

⇒ 사원테이블과 부서 테이블에 있는 데이터 사이에 불일치 발생!

1.3 기본키와 외래키

EMP

empid	ename	deptid	hire_date	job	manager	salary
1001	홍성길	100	2001.2.1	특수영업	1002	350
1002	곽희준	100	1999.1.1	영업관리	1004	400
1003	김동준	200	2000.9.1	품질관리	1005	300
1004	성재규	300	1997.2.1	급여	1009	450
1005	박성범	200	2000.2.1	수입자재	1004	320

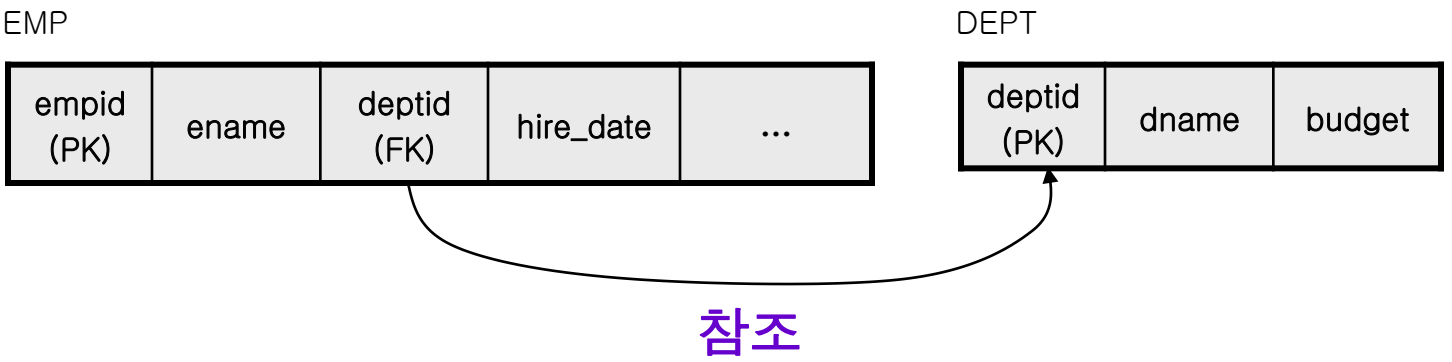
DEPT

deptid	dname	budget
100	영업부	100k
200	관리부	300k
300	구매부	220k
400	생산부	500k

<그림 1.13> 사원과 부서정보 테이블

1.3 기본키와 외래키

❑ 외래키(Foreign key)



1.3 기본키와 외래키

❑ 외래키(Foreign key)

부서 테이블의 첫번째 튜플을 삭제하려 할 때

- 제한(restrict)
  - 삭제하려는 튜플의 부서번호 값을 사원 테이블에서 가지고 있는 튜플이 있으므로 삭제 연산을 거절
- 연쇄(cascade)
  - 삭제된 부서번호 값을 갖는 사원 테이블의 튜플도 함께 삭제
- 널값으로 대체(nullify)
  - 삭제연산을 수행한 뒤 삭제된 부서번호 값을 갖는 사원 테이블의 튜플에서 부서번호를 null 값으로 대체

* 외래키를 통해 두 테이블간의 데이터 무결성을 유지하는 것을 '참조 무결성 제약조건'이라고 한다.

1.3 기본키와 외래키

❑ 외래키(Foreign key)

예제

```
CREATE TABLE Test2
(
    ID INT,
    ParentID INT,
    FOREIGN KEY (ParentID)
    REFERENCES Test1(ID) ON UPDATE CASCADE ON DELETE RESTRICT
);
```