데이터베이스

인공지능소프트웨어학과

담당교수: 김희숙

(jasmin11@hanmail.net)

데이터베이스

7주차

담당교수: 김희숙
(jasmin11@hanmail.net)

Quiz

담당교수: 김희숙
(jasmin11@hanmail.net)

[실습] 테이블 생성(테이블 3개)

[실습 1-2] (테이블 3개) 학생, 수강, 과목 테이블 생성(기본키, 외래키)

[3개 테이블 : 기본키, 외래키]

학생

학번	학생명	학년
1111	홍길동	1
2222	김윤식	3
3333	이정진	2
4444	홍진아	1

수강

학번	과목번호	성적
1111	CS100	98
1111	CS102	88
2222	CS102	90
3333	CS100	92

과목

과목번호	과목명
CS100	데이터베이스
CS101	운영체제
CS102	자료구조

❖ 학과, 학생

테이블 생성 순서

테이블 삭제 순서

데이터 입력 순서

[실습] 테이블 생성(테이블 3개)

[실습 1-2] (테이블 3개) 학생, 수강, 과목 테이블 생성(기본키, 외래키)

❖ 기본키가 복합키인 경우

```
drop table [redacted];
drop table [redacted];
drop table 학생;

--학생(학번, 학생명, 학년)
[redacted] 학생 (
  학번        char(4) NOT NULL ,
  학생명      varchar(12) ,
  학년        int ,
[redacted] KEY(학번)
);

--과목(과목번호, 과목명)
[redacted] 과목 (
  과목번호   char(5),
  과목명     varchar(30),
[redacted] KEY(과목번호)
);

--수강(학번, 과목번호, 성적)
[redacted] 수강 (
  학번        [redacted],
  과목번호   char(5),
  성적        int,
[redacted] KEY([redacted]),
[redacted] KEY (학번)      REFERENCES [redacted],
[redacted] KEY (과목번호) REFERENCES [redacted])
);
```

[3개 테이블 : 기본키, 외래키]

학생

학번	학생명	학년
1111	홍길동	1
2222	김윤식	3
3333	이정진	2
4444	홍진아	1

수강

학번	과목번호	성적
1111	CS100	98
1111	CS102	88
2222	CS102	90
3333	CS100	92

과목

과목번호	과목명
CS100	데이터베이스
CS101	운영체제
CS102	자료구조

[실습] 테이블 생성(테이블 3개)

동양미래대학교 DONGYANG MIRAE UNIVERSITY

[실습 1-3] (테이블 3개) 학생, 수강, 과목 테이블 생성(기본키, 외래키)

[실습]

-- 과목(과목번호, 이름, 강의실, 개설학과, 시수)

-- 학생(학번, 이름, 주소, 학년, 나이, 휴대폰번호, 소속학과)

-- 수강(학번, 과목번호, 신청날짜, 중간성적, 기말성적, 평가학점)

과목번호	이름	강의실	개설학과	시수
c001	데이터베이스	126	컴퓨터	3
c002	정보보호	137	정보통신	3
c003	모바일웹	128	컴퓨터	3
c004	철학개론	117	철학	2
c005	전공글쓰기	120	교양학부	1
NULL	NULL	NULL	NULL	NULL

학번	과목번호	신청날짜	중간성적	기말성적	평가학점
s001	c002	2019-09-03	93	98	A
s001	c004	2019-03-05	82	89	B
s001	c005	2020-09-03	74	79	C
s002	c001	2018-03-10	31	50	F
s003	c001	2019-03-03	81	82	B
s003	c002	2017-09-06	85	82	B
s004	c002	2018-03-05	92	95	A
s004	c003	2020-09-03	91	94	A
s004	c005	2019-03-03	72	78	C

학번	이름	주소	학년	나이	성별	휴대폰번호	소속학과
s001	김연아	서울 서초	4	23	여	010-1111-2222	컴퓨터
s002	홍길동	미정	1	26	남	NULL	통계
s003	이승엽	NULL	3	30	남	NULL	정보통신
s004	이영애	경기 분당	2	NULL	여	010-4444-5555	정보통신
s005	송윤아	경기 분당	4	23	여	010-6666-7777	컴퓨터
s006	홍길동	서울 종로	2	26	남	010-8888-9999	컴퓨터
s007	이은진	경기 과천	1	23	여	010-2222-3333	경영

[실습] 무결성 제약조건 (과목) (stu)

[실습 2-01] DDL (무결성 제약조건 없는 경우)

-- 과목(과목번호,이름,강의실,개설학과, 시수)

```
-- 1) 제약조건 없는 경우
drop table 과목;

CREATE TABLE 과목 (
      과목번호 char(4)          NOT NULL  ,
      이름       VARCHAR(20)    ,
      강의실     CHAR(3)         ,
      개설학과 VARCHAR(20)    ,
      시수       INT
) ;

-- 테이블 구조 확인
DESC 과목;
```

[실습 2-02] DDL (무결성 제약조건 있는 경우)

-- 과목(과목번호,이름,강의실,개설학과, 시수)

```
-- 2) 제약조건 있는 경우
drop table if exists 과목;
```

-- 다음 조건을 만족하는 데이터 정의어 작성

<조건>

1) 과목번호,이름,강의실,개설학과, 시수 로 구성된 과목 테이블을 생성

2) 기본키는 과목번호로 설정한다

[실습] 무결성 제약조건 (학생) (stu)

[실습 3-01] DDL (무결성 제약조건 없는 경우)

```
-- 학생(학번, 이름, 주소, 학년, 나이, 휴대폰번호, 소속학과)
-- 1) 제약조건 없는 경우
drop table 학생;

CREATE TABLE 학생 (
        학번        CHAR(4)          NOT NULL ,
        이름        VARCHAR(20)      NOT NULL ,
        주소        VARCHAR(50)      ,
        학년        INT
        나이        INT  ,
         성별        CHAR(1)
         휴대폰번호   CHAR(14) ,
         소속학과     VARCHAR(20)
) ;
-- 테이블 구조 확인
DESC 학생;
```

-- 다음 조건을 만족하는 데이터 정의어 작성

<조건>

1) 기본키는 학번으로 설정한다
2) 휴대폰번호를 대체키로 지정한다
3) 주소 에는 기본값을 설정한다 (기본값: 미정)

[실습 3-02] DDL (무결성 제약조건 있는 경우)

```
-- 학생(학번, 이름, 주소, 학년, 나이, 휴대폰번호, 소속학과)

-- 2) 제약조건 있는 경우
drop table if exists 학생;
```

[실습] 무결성 제약조건 (수강) (stu)

[실습 4-01] DDL (무결성 제약조건 없는 경우)

```
-- 수강(학번,과목번호,신청날짜,중간성적,기말성적,평가학점)
-- 1) 제약조건 없는 경우
drop table 수강;

CREATE TABLE 수강 (
        학번        char(6)          NOT NULL ,
        과목번호    CHAR(4)          NOT NULL ,
        신청날짜    DATE             ,
        중간성적    INT              ,
        기말성적    INT              ,
        평가학점    CHAR(1) ,
         PRIMARY KEY(학번, 과목번호)
) ;

-- 테이블 구조 확인
DESC 수강;
```

[실습 4-02] DDL (무결성 제약조건 있는 경우)

```
-- 수강(학번,과목번호,신청날짜,중간성적,기말성적,평가학점)

-- 2) 제약조건 있는 경우
drop table if exists 수강;
```

```
-- 다음 조건을 만족하는 데이터 정의어 작성
<조건>
1) 기본키는 (학번,과목번호) 로 설정한다
2) 학번 은 외래키로 학생 테이블의 학번을 참조한다
3) 과목번호는 외래키로 과목 테이블의 과목번호를 참조한다
4) 중간성적, 기말성적에는 기본값을 설정한다(기본값: 0)
```

Quiz

담당교수: 김희숙
(jasmin11@hanmail.net)

[Quiz] 테이블 생성(테이블 1개)

데이터베이스: testdb

테이블: 사원

[Quiz 1-1] (테이블 1개) 사원 테이블 생성(데이터 입력, 널 값)

```
-- (MySQL)
drop table 사원;


-- 사원(사원번호,사원명,연락처,생일)
CREATE TABLE 사원 (
    사원번호      char(4) ,
    사원명        varchar(20) ,
    연락처        char(13) ,
    생일          varchar(15)  ,
    PRIMARY KEY(사원번호)
);
```

사원

사원번호	사원명	연락처	생일
D001	정지영		NULL
D002	김선주	010-1111-1111	NULL
D003	정성호	NULL	10월04일

[실습] DML (사원)

[실습] 데이터 입력/수정/삭제/조회

[Quiz 1-02] DML: (사원)

-- 1. 정지영 연락처를 010-9999-9999, 생일을 10월11일 로 수정하라
-- 2. 김선주 생일을 10월11일로 수정하라
-- 3. 정성호 레코드를 삭제하라

사원

사원번호	사원명	연락처	생일
D001	정지영		NULL
D002	김선주	010-1111-1111	NULL
D003	정성호	NULL	10월04일

요약

담당교수: 김희숙
(jasmin11@hanmail.net)

데이터베이스

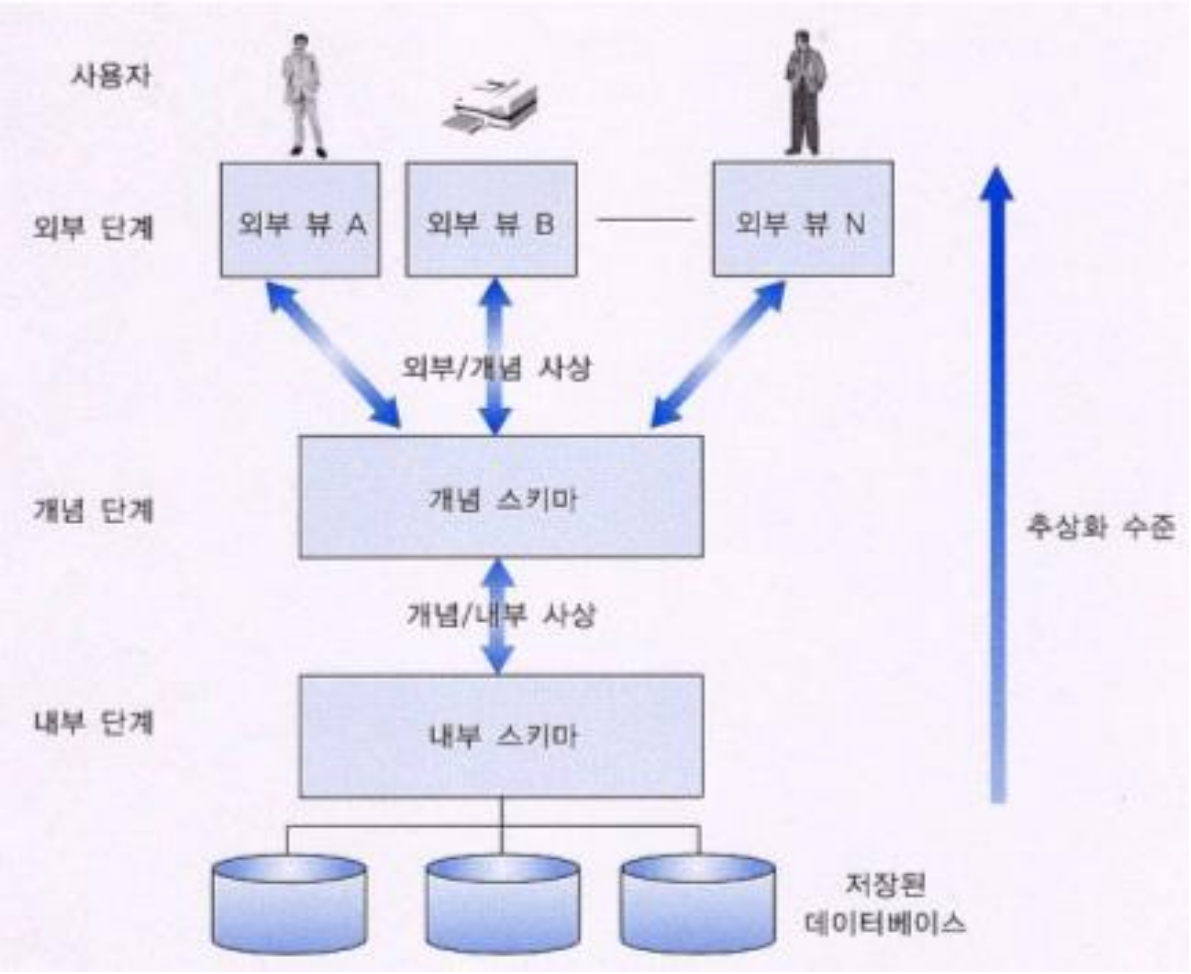
- **데이터베이스**:
  - 한 조직의 여러 응용시스템들이 데이터를 **공유**할 수 있도록 **통합**, **저장**된 **운영**데이터의 집합체

* **데이터베이스 정의**:
1) 공유 데이터(shared data)
2) 통합 데이터(integrated data)
3) 저장 데이터(stored data)
4) 운영 데이터(operational data)

* **데이터베이스 특성**:
1) 실시간 접근이 가능(real time accessibility)
2) 계속 변화(continuous evolution)
3) 동시 공유 가능(concurrent sharing)
4) 내용으로 참조 가능(contents reference)

[요약] 3단계 구조

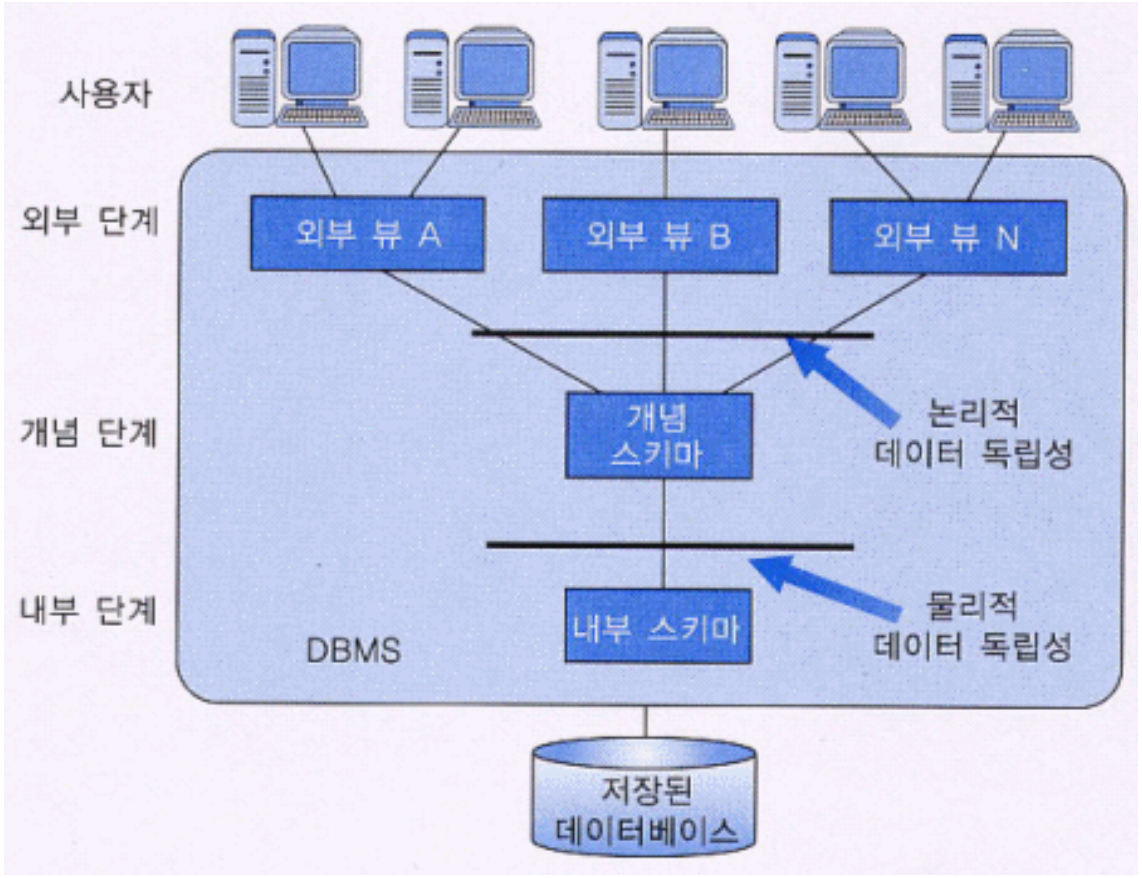
ANSI/SPARC 아키텍처 (스키마 3단계 구조)



(그림 출처: "데이터베이스배움터", 홍의경 저, 생능, 2012)

[요약] 데이터 독립성

* **3단계 스키마 구조:**
  1) 외부 스키마
  2) 개념 스키마
  3) 내부 스키마



(그림 출처: "데이터베이스배움터", 홍의경 저, 생능, 2012)

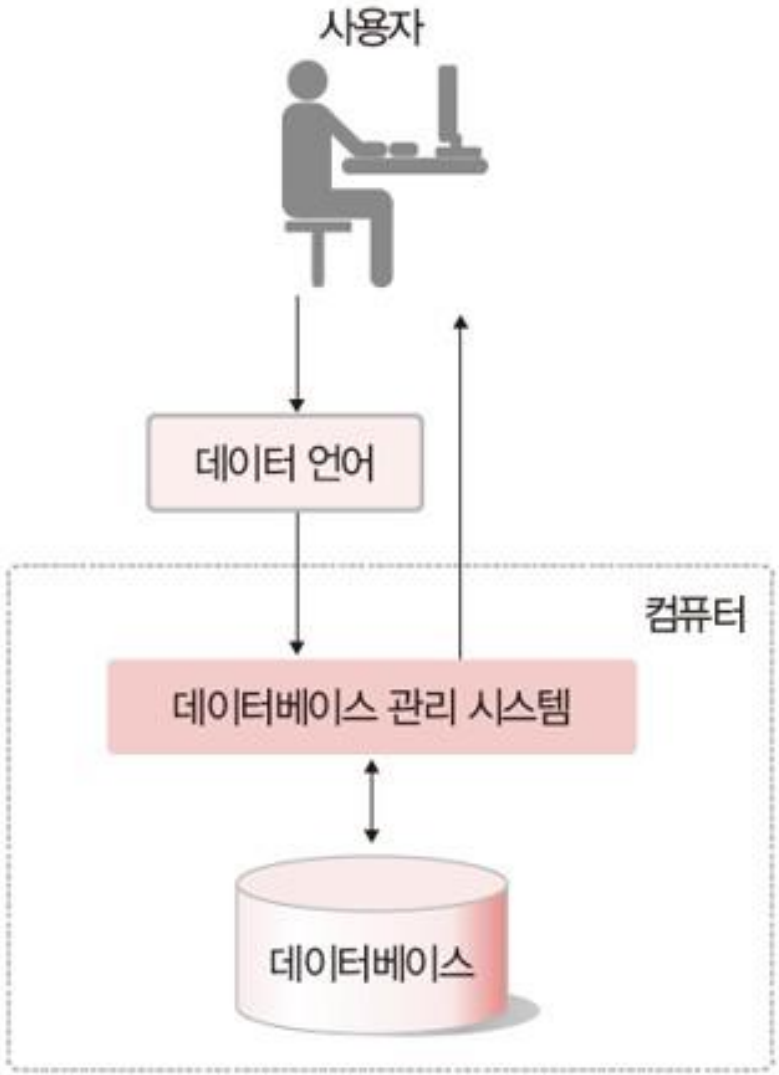
- **논리적 데이터 독립성:**
  개념스키마가 변화해도
  외부스키마는 영향 받지 않는다

- **물리적 데이터 독립성:**
  내부스키마가 변화해도
  개념스키마, 외부스키마는 영향 받지 않는다

데이터베이스 시스템: 구성요소

❑데이터베이스 시스템

- ✓데이터(데이터베이스)
- ✓하드웨어(**HW**)
- ✓**DBMS**
- ✓사용자
- ✓(데이터언어)



(그림 출처: "데이터베이스 개론 2판", 김연희 서, 한빛아카데미, 2019)

[요약] 데이터베이스 용어

* **데이터베이스 사용자**:
  데이터베이스 관리자(DBA), 응용프로그램 개발자, 최종 사용자

  데이터베이스 관리자(Database Administrator: DBA)
  (데이터 정의어와 데이터 제어어를 사용)

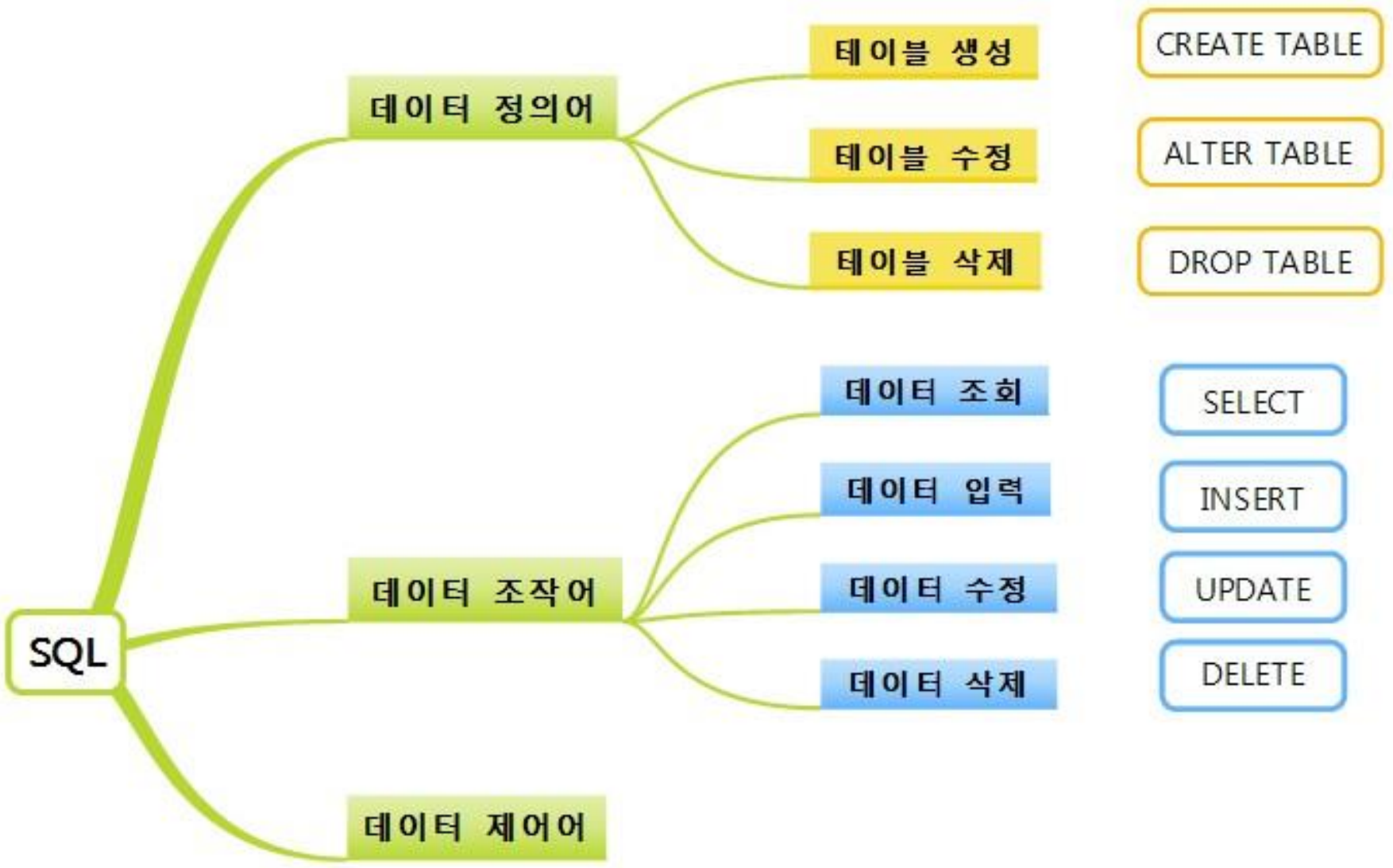
* **데이터베이스 언어(SQL)**:
  데이터 정의어(DDL)
  데이터 조작어(DML)
  데이터 제어어(DCL)

데이터베이스 언어(SQL)

- 데이터베이스 언어(SQL)
  - ✓데이터 정의어(DDL)
  - ✓데이터 조작어(DML)
  - ✓데이터 제어어(DCL)

SQL	명령어
DDL	CREATE ALTER DROP
DML	INSERT UPDATE DELETE SELECT
DCL	GRANT REVOKE

SQL


SQL
데이터 정의어
데이터 조작어
데이터 제어어
테이블 생성
테이블 수정
테이블 삭제
데이터 조회
데이터 입력
데이터 수정
데이터 삭제
CREATE TABLE
ALTER TABLE
DROP TABLE
SELECT
INSERT
UPDATE
DELETE

[요약] 관계형 데이터 모델

❑용어

- ✓ 릴레이션(Relation), 테이블
- ✓ 속성(attribute), 필드, 열(column)
- ✓ 투플(tuple), 레코드, 행(row)
- ✓ 도메인(domain)
- ✓ 차수(degree)
- ✓ 카디날리티(cardinality)

- ✓ 릴레이션 스키마: 내포(intension), 정적
- ✓ 릴레이션 인스턴스: 외연(extension), 동적

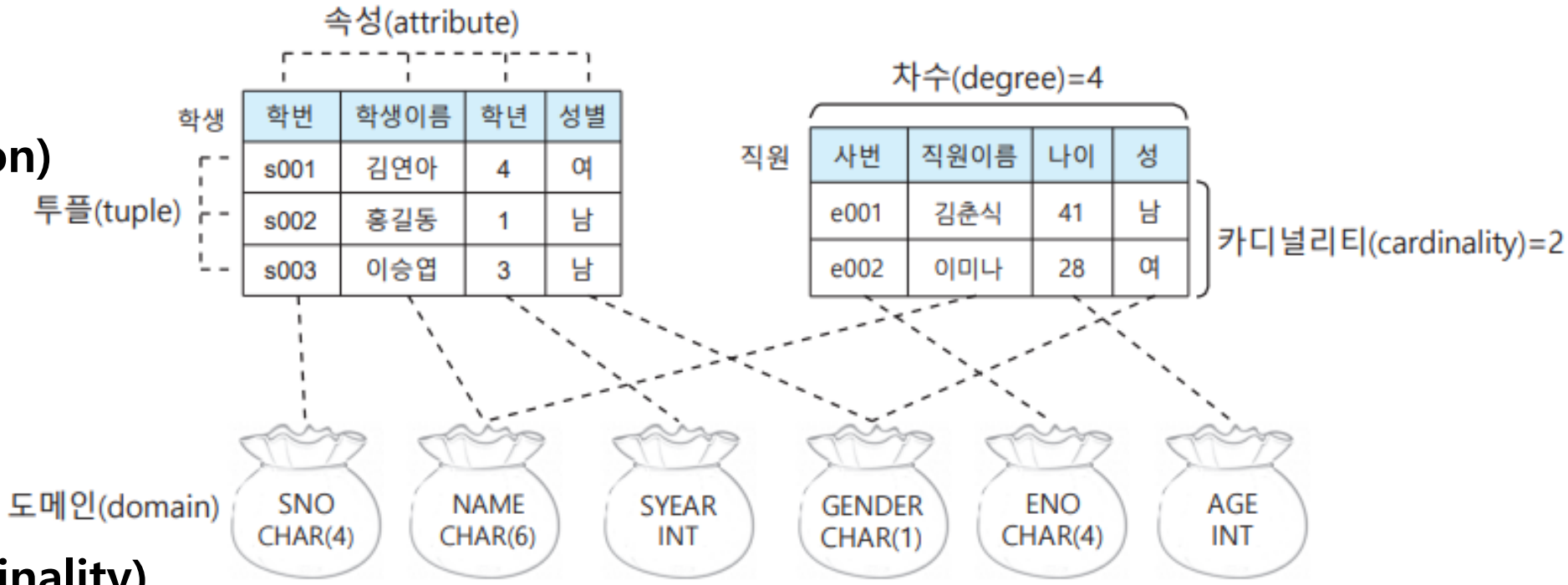
번호	이름	학년	분반	학과번호
1	한지혜	1	YB	1
2	이정우	1	YA	1
3	오지영	2	J1	2
4	강재미	1	YB	1
5	박철호	2	J1	2

사원

스키마	사원번호	사원명	연락처	생일
인스턴스	D001	정지영		NULL
	D002	김선주	010-1111-1111	NULL
	D003	정성호	NULL	10월04일

[요약] 관계형 데이터 모델

- 용어
  - ✓릴레이션(Relation)
  - ✓속성(attribute)
  - ✓투플(tuple)
  - ✓도메인(domain)
  - ✓차수(degree)
  - ✓카디날리티(cardinality)



(그림 출처: ""SQL과 NoSQL 기반의데이터베이스 입문", 박성진, 생능, 2023)

키(Key) / 제약조건

❑키

- 각 투플을 고유하게 식별할 수 있는 하나 이상의 애트리뷰트 들의 모임

✓ 수퍼키(**super key**)

✓ 후보키(**candidate key**)

✓ 기본키(**primary key**): **PRIMARY KEY**

✓ 대체키(**alternate key**): **UNIQUE**

✓ 외래키(**foreign key**): **FOREIGN KEY**

키	특성
수퍼키	유일성
후보키	유일성, 최소성
기본키	중복불가, 필수입력
대체키	
외래키	

[요약] 데이터 무결성(integrity)

◆ **무결성: 데이터의 정확성, 유효성**

◆ 데이터가 입력되거나 수정, 삭제되어 질 때 잘못된 데이터가 존재하게 되는 경우를 방지하기 위한 것

◆ **데이터 정의어**에서 기본키, 외래키를 정의하면 **DBMS**가 **일관성 조건 검사**

1. 무결성 제약조건

1) 개체 무결성 제약조건: 기본키는 널 값을 가질 수 없다(기본키) PRIMARY KEY

2) 참조 무결성 제약조건: (외래키) FOREIGN KEY

참조하는 테이블의 외래키 값은

참조되는 테이블의 기본키 값과 같다.

3) 키 제약조건: 키는 중복된 값을 갖지 않는다 UNIQUE

4) 도메인 제약조건: CHECK, DEFAULT 문법 CHECK, DEFAULT

[요약] 무결성 제약조건(integrity Constraint)

1) 개체 무결성 제약조건(엔티티 무결성 제약조건)

기본키는 널 값을 가질 수 없다

2) 참조 무결성 제약조건

참조하는 테이블의 외래키 값은 참조되는 테이블의 기본키 값에 반드시 존재해야 한다

제약조건	부모 테이블	자식 테이블
입력	**제약 없음**	부모테이블에 데이터가 존재하는지 검증
수정	수정하려는 데이터를 자식테이블에서 참조하고 있는지를 검증	부모테이블에 존재하는 다른 데이터로 변경가능
삭제	삭제하려는 데이터를 자식테이블에서 참조하고 있는지를 검증	**제약 없음**

실습

담당교수: 김희숙
(jasmin11@hanmail.net)

[실습 01] (주소록): 데이터 입력

데이터베이스: studydb

테이블: 주소록

```
-- 주소록(이름, 전화번호, 주소, 생일)
use studydb;

DROP TABLE 주소록;

-- 주소록 테이블 생성
CREATE TABLE 주소록(
    이름        char(10)          NOT NULL ,
    전화번호    char(13) ,
    주소        varchar(10) ,
    생일        varchar(11) ,
    PRIMARY KEY(이름)
);
```

자동증가 없는 경우,

주소록

이름	전화번호	주소	생일
홍길동	010-1234-5678	서울	3월 15일
이몽룡	010-3343-5643	부산	12월 14일
최용만	321-2346	대전	5월 8일
이건우	010-2132-2345	NULL	NULL

[실습 01] (주소록): 데이터 입력

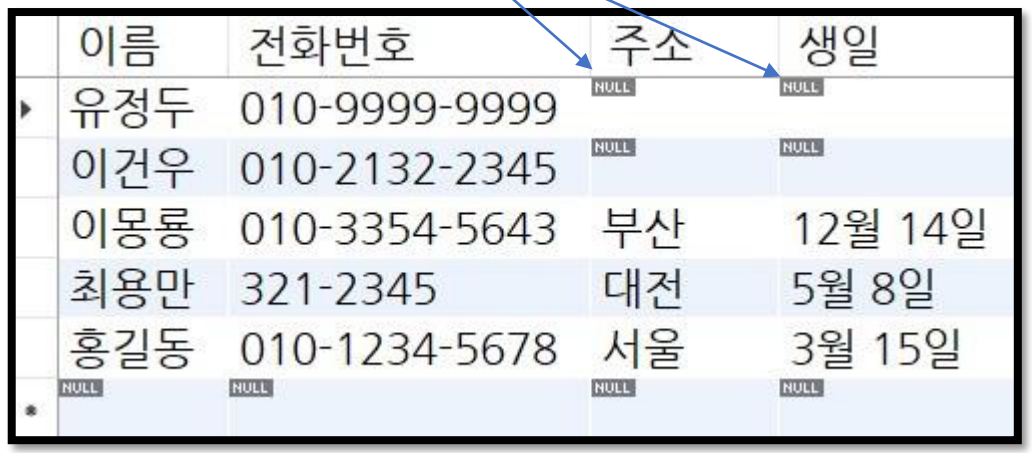
```
-- 데이터 입력
-- 주소록(이름, 전화번호, 주소, 생일)

INSERT INTO 주소록
VALUES('홍길동','010-1234-5678','서울','3월 15일');

INSERT INTO 주소록
VALUES('이몽룡','010-3354-5643','부산','12월 14일');

INSERT INTO 주소록
VALUES('최용만','321-2345','대전','5월 8일');
```

```
-- 다음과 같이 널 값 데이터 입력 하시오
```



이름	전화번호	주소	생일
유정두	010-9999-9999	NULL	NULL
이건우	010-2132-2345	NULL	NULL
이몽룡	010-3354-5643	부산	12월 14일
최용만	321-2345	대전	5월 8일
홍길동	010-1234-5678	서울	3월 15일
NULL	NULL	NULL	NULL

```
-- 데이터 조회
select * from 주소록;
```

[실습 01] (주소록): 널 값 입력

[실습] 널(NULL) 값 입력하는 방법

```
-- 데이터 입력 방법
INSERT INTO 테이블명 VALUES(데이터값, ...);                    -- 필드명 생략 한 경우
INSERT INTO 테이블명(필드명) VALUES(데이터값, ...);            -- 필드명 생략하지 않은 경우
```

이름	전화번호	주소	생일
홍길동	010-1234-5678	서울	3월 15일
이몽룡	010-3354-5643	부산	12월 14일
최용만	321-2345	대전	5월 8일
이건우	010-2132-2345	NULL	NULL

--널 값 데이터 입력하는 방법

```
INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('이건우','010-2132-2345', NULL, NULL);
```

```
-- --널 값 데이터 입력하는 방법
INSERT INTO 주소록(이름, 전화번호, 주소, 생일) VALUES('이건우','010-2132-2345', NULL, NULL);
INSERT INTO 주소록(이름, 전화번호) VALUES('유정두','010-9999-9999');
```

[실습 02] (주소록): 자동증가 AUTO_INCREMENT

```
-- 주소록(번호, 이름, 전화번호, 주소, 생일)
use studydb;

DROP TABLE if exists 주소록;

-- 주소록 테이블 생성
CREATE TABLE 주소록(
    번호        int          AUTO_INCREMENT ,
    이름        char(10)     NOT NULL ,
    전화번호    char(13) ,
    주소        varchar(10) ,
    생일        varchar(11) ,
    PRIMARY KEY(번호)
);
```

데이터베이스: testdb

테이블: 주소록

자동증가 있는 경우,

주소록

번호	이름	전화번호	주소	생일
1	홍길동	010-1234-5678	서울	3월 15일
2	이몽룡	010-3343-5643	부산	12월 14일
3	최용만	321-2346	대전	5월 8일
4	이건우	010-2132-2345	NULL	NULL

[실습 02] (주소록): 자동증가 AUTO_INCREMENT

```
-- 데이터 입력
-- 주소록(번호, 이름, 전화번호, 주소, 생일)

INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('홍길동','010-1234-5678','서울','3월 15일');

INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('이몽룡','010-3354-5643','부산','12월 14일');

INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('최용만','321-2345','대전','5월 8일');
```

번호	이름	전화번호	주소	생일
1	홍길동	010-1234-5678	서울	1990-03-15
2	이몽룡	010-3354-5643	부산	1994-12-14
3	최용만	321-2345	대전	1994-05-08
4	이건우	010-2132-2345	NULL	NULL
5	유정두	010-9999-9999	NULL	NULL
NULL	NULL	NULL	NULL	NULL

```
-- 다음과 같이 널 값 데이터 입력 하시오
```

```
-- 데이터 조회
select * from 주소록;
```

[실습 02] (주소록): 자동증가 AUTO_INCREMENT

[실습] 자동증가 필드가 있는 경우, 데이터 입력하는 방법

```
-- 주소록(번호, 이름, 전화번호, 주소, 생일)
INSERT INTO 테이블명 VALUES(데이터값, ...);          -- 오류: 필드명 생략 불가능

-- 예) 이 방법으로 데이터 입력한다(자동증가 필드 있는 경우)
INSERT INTO 주소록(이름, 전화번호, 주소, 생일) VALUES('홍길동','010-1234-5678','서울','3월 15일');
-- 자동증가 필드는 데이터 입력 불가능: 즉, 번호 필드는 자동생성 되므로 사용자가 입력하지 않는다
```

[실습] 널(NULL) 값 입력하는 방법

```
-- --널 값 데이터 입력하는 방법
INSERT INTO 주소록(이름, 전화번호, 주소, 생일) VALUES('이건우','010-2132-2345', NULL, NULL);
INSERT INTO 주소록(이름, 전화번호) VALUES('유정두','010-9999-9999');
```

[요약] (MySQL) 자동증가: auto_increment

[실습] 자동증가 없음

```
-- 주소록(이름, 전화번호, 주소, 생일)
CREATE TABLE 주소록(
  이름        char(10)     NOT NULL ,
  전화번호    char(13) ,
  주소        varchar(10) ,
  생일        varchar(11) ,
  PRIMARY KEY(이름)
);
```

[실습] 자동증가 있음

```
-- 주소록(번호, 이름, 전화번호, 주소, 생일)
CREATE TABLE 주소록(
  번호        int       AUTO_INCREMENT ,
  이름        char(10) NOT NULL ,
  전화번호    char(13) ,
  주소        varchar(10) ,
  생일        varchar(11) ,
  PRIMARY KEY(번호)
);
```

```
INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('홍길동','010-1234-5678','서울','3월 15일');
```

[실습] (MySQL) 자동증가: auto_increment

```
-- 주소록(이름, 전화번호, 주소, 생일)

-- 주소록 테이블 생성
CREATE TABLE 주소록(
  이름        char(10) NOT NULL ,
  전화번호     char(13) ,
  주소        varchar(10) ,
  생일        varchar(11) ,
  PRIMARY KEY(이름)
);

-- 데이터 입력
-- 주소록(이름, 전화번호, 주소, 생일)

INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('홍길동','010-1234-5678','서울','3월 15일');

INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('이몽룡','010-3354-5643','부산','12월 14일');

INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('최용만','321-2345','대전','5월 8일');

-- 데이터 조회
select * from 주소록;
```

```
-- MySQL 실습: 자동증가 필드 추가

use studydb;

-- 주소록(이름, 전화번호, 주소, 생일)

drop table 주소록;

-- 번호 필드 추가한다(자동증가: auto_increment)
-- 주소록(번호, 이름, 전화번호, 주소, 생일)

-- 주소록 테이블 생성
CREATE TABLE 주소록(
  번호        int       AUTO_INCREMENT,
  이름        char(10) NOT NULL ,
  전화번호     char(13) ,
  주소        varchar(10) ,
  생일        varchar(11) ,
  PRIMARY KEY(번호)
);

-- 데이터 입력
-- 주소록(번호, 이름, 전화번호, 주소, 생일)

-- 입력 가능
INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('홍길동','010-1234-5678','서울','1990-03-15');

INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('이몽룡','010-3354-5643','부산','1994-12-14');

INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('최용만','321-2345','대전','1994-05-08');

-- 데이터 조회
select * from 주소록;
```

[실습] (MySQL): 입력,수정,삭제

❖ (MySQL): Safe_mode 해제

1) MySQL Workbench

[Edit]-[Preferences]-SQL Editor-Safe Updates 체크 해제

2) 쿼리: safe_mode 체크 해제

```
SET SQL_SAFE_UPDATES = 0;
```

SET SQL_SAFE_UPDATES = 0;

[Quiz 1] (주소록): 데이터 수정, 삭제

데이터베이스: studydb

테이블: 주소록

```
-- 주소록(이름,전화번호,주소,생일)
-- 1. 홍길동의 전화번호를 '010-3245-4368' 로 수정하라
-- 2. 이건우의 주소는 '서울', 생일은 '8월 23일' 로 수정하라
-- 3. 최용만의 레코드를 삭제하라
```

```
-- (MySQL) Safe mode 해제
SET SQL_SAFE_UPDATES = 0;
```

-- 데이터 수정, 삭제 이전

수정, 삭제 이전

주소록

이름	전화번호	주소	생일
홍길동	010-1234-5678	서울	3월 15일
이몽룡	010-3343-5643	부산	12월 14일
최용만	321-2346	대전	5월 8일
이건우	010-2132-2345	NULL	NULL

-- 데이터 수정, 삭제 이후

수정, 삭제 이후

주소록

이름	전화번호	주소	생일
홍길동	010-3245-4368	서울	3월 15일
이몽룡	010-3343-5643	부산	12월 14일
이건우	010-2132-2345	서울	8월 23일

[실습 03] (주소록): 데이터 수정, 삭제

```
-- 주소록(이름, 전화번호, 주소, 생일)
use studydb;

DROP TABLE if exists 주소록;

-- 주소록 테이블 생성
CREATE TABLE 주소록(
    이름        char(10)          NOT NULL ,
    전화번호    char(13) ,
    주소        varchar(10) ,
    생일        varchar(11) ,
    PRIMARY KEY(이름)
);
```

```
INSERT INTO 주소록
VALUES('홍길동','010-1234-5678','서울','3월 15일');

INSERT INTO 주소록
VALUES('이몽룡','010-3354-5643','부산','12월 14일');

INSERT INTO 주소록
VALUES('최용만','321-2345','대전','5월 8일');

INSERT INTO 주소록(이름, 전화번호, 주소, 생일)
VALUES('이건우','010-2132-2345', NULL, NULL);

-- 데이터 조회
select * from 주소록;
```

[실습 03] (주소록): 데이터 수정

```
-- UPDATE: 데이터 수정

UPDATE   테이블명
SET      필드명 = 수정 후 값
WHERE    필드명 = 데이터값;
```

```
-- 1. 홍길동의 전화번호를 '010-3245-4368' 로 수정하라

UPDATE  주소록
SET     전화번호 =  '010-3245-4368'
WHERE   이름 = '홍길동' ;
```

```
-- 주소록(이름,전화번호,주소,생일)
-- 1. 홍길동의 전화번호를 '010-3245-4368' 로 수정하라

-- 2. 이건우의 주소는 '서울', 생일은 '8월 23일' 로 수정하라

-- 3. 최용만의 레코드를 삭제하라

-- 4. 이몽룡의 데이터 조회하라
```

[실습 03] (주소록): 데이터 수정

-- UPDATE: 데이터 수정

```
UPDATE   테이블명
SET      필드명 = 수정 후 값
WHERE    필드명 = 데이터값;
```

-- 2. 이건우의 주소는 '서울', 생일은 '8월 23일' 로 수정하라

```
UPDATE  주소록
SET     주소= '서울', 생일= '8월 23일'
WHERE   이름 = '이건우';
```

-- 주소록(이름,전화번호,주소,생일)

-- 1. 홍길동의 전화번호를 '010-3245-4368' 로 수정하라

-- 2. 이건우의 주소는 '서울', 생일은 '8월 23일' 로 수정하라

-- 3. 최용만의 레코드를 삭제하라

-- 4. 이몽룡의 데이터 조회하라

[실습 03] (주소록): 데이터 삭제

-- UPDATE: 데이터 수정

```
DELETE
FROM      테이블명
WHERE     필드명 = 데이터값;
```

-- 3. 최용만의 레코드를 삭제하라

```
DELETE
FROM      주소록
WHERE     이름 = '최용만';
```

-- 주소록(이름,전화번호,주소,생일)

-- 1. 홍길동의 전화번호를 '010-3245-4368' 로 수정하라

-- 2. 이건우의 주소는 '서울', 생일은 '8월 23일' 로 수정하라

-- 3. 최용만의 레코드를 삭제하라

-- 4. 이몽룡의 데이터 조회하라

[실습 03] (주소록): 데이터 조회

-- SELECT: 데이터 조회, 데이터 검색

```
SELETE    필드명
FROM      테이블명
WHERE     필드명 = 데이터값;
```

- 4. 이몽룡의 데이터 조회하라

```
SELETE    *
FROM      주소록
WHERE     이름 = '이몽룡';
```

-- 주소록(이름,전화번호,주소,생일)
-- 1. 홍길동의 전화번호를 '010-3245-4368' 로 수정하라

-- 2. 이건우의 주소는 '서울', 생일은 '8월 23일' 로 수정하라

-- 3. 최용만의 레코드를 삭제하라

-- 4. 이몽룡의 데이터 조회하라

무결성 제약조건

담당교수: 김희숙
(jasmin11@hanmail.net)

[실습] 무결성 제약조건 (신입생)

[실습 1-02] DDL (무결성 제약조건 있는 경우)

```
-- 다음 조건을 만족하는 데이터정의어를 작성하시오
-- (조건)
-- 1) 기본키는 학번으로 지정
-- 2) 주민등록번호는 중복된 값을 갖지 않는다
-- 3) 학년은 기본값 1을 갖는다
-- 4) 성별 입력값은 남, 여 로 값을 제한한다)
```

```
insert into 신입생(학번,이름,학년, 성별)
values('20211111','홍길동',3,'남');

insert into 신입생(학번,이름,성별)
values('20232222','조유나','여');

insert into 신입생(학번,이름,성별)
values('20233333','주민증','남자');
```

[실습 1-01] DDL (무결성 제약조건 없는 경우)

```
-- 신입생(학번,주민,이름,학년,성별,주소,학과명)


-- 1) 제약조건 없는 경우
drop table 신입생;

CREATE TABLE 신입생(
    학번            char(8) ,
    주민등록번호    char(14)        ,
    이름            varchar(20)     ,
    학년            int             ,
    성별            char(1)         ,
    주소            varchar(20) ,
    학과명          varchar(20) ,
    PRIMARY KEY(학번)
);

-- 테이블 구조 확인
DESC 신입생;
```

[실습] 무결성 제약조건 (신입생) (ans)

[실습 1-02] DDL (무결성 제약조건 있는 경우)

```
-- 다음 조건을 만족하는 데이터정의어를 작성하시오
-- (조건)
-- 1) 기본키는 학번으로 지정
-- 2) 주민등록번호는 중복된 값을 갖지 않는다
-- 3) 학년은 기본값 1을 갖는다
-- 4) 성별 입력값은 남, 여 로 값을 제한한다)
```

```
insert into 신입생(학번,이름,학년, 성별)
values('20211111','홍길동',3,'남');

insert into 신입생(학번,이름,성별)
values('20232222','조유나','여');

insert into 신입생(학번,이름,성별)
values('20233333','주민증','남자');
```

[실습 1-02] DDL (무결성 제약조건 있는 경우)

```
-- 신입생(학번,주민,이름,학년,성별,주소,학과명)


-- 2) 제약조건 있는 경우
drop table if exists 신입생;

CREATE TABLE 신입생(
    학번          char(8) ,
    주민등록번호  char(14)      UNIQUE  ,
    이름          varchar(20)    ,
    학년          int            DEFAULT 1 ,
    성별          char(1)        CHECK (성별 IN('남','여')) ,
    주소          varchar(20) ,
    학과명        varchar(20) ,
    PRIMARY KEY(학번)
);
```

[요약] 테이블 생성과 삭제

❖ 테이블 생성과 테이블 삭제

- ✓ 테이블 생성시, 부모 테이블 먼저 생성
- ✓ 테이블 삭제시, 자식 테이블 먼저 삭제

부모테이블

부서

부서코드	부서명
AA	총무부
BB	영업부
CC	기획부

자식테이블

사원

사원번호	이름	외래키 부서코드
1111	홍길동	AA
2222	임꺽정	AA
3333	박찬호	BB
4444	선동열	BB
5555	차두리	AA
6666	신동엽	BB

[실습] (MySQL) 무결성 제약조건

부서코드	부서명
AA	총무부
BB	영업부
CC	기획부

사원번호	이름	부서코드
1111	홍길동	AA
2222	임꺽정	AA
3333	박찬호	BB
4444	선동열	BB
5555	차두리	AA
6666	신동엽	BB

```
--[Quiz] 다음 문법은 실행하라.  실행 가능한가? 그 이유는?
-- 2-1) 다음 문법을 실행하라.  실행 가능한가?  그 이유는?
INSERT INTO 사원 VALUES('7777','이승엽','BB');

-- 2-2) 다음 문법의 실행결과는? 그 이유는?
UPDATE 부서 SET  부서코드='DD'    WHERE   부서코드='BB';

-- 2-3) 다음 문법의 실행결과는? 그 이유는?
INSERT INTO 부서 VALUES('DD','개발부');
```

❖ 참조 무결성 제약조건

제약조건	부모 테이블	자식 테이블
입력	제약 없음	부모테이블에 데이터가 존재하는지 검증
수정	수정하려는 데이터를 자식테이블에서 참조하고 있는지를 검증	부모테이블에 존재하는 다른 데이터로 변경가능
삭제	삭제하려는 데이터를 자식테이블에서 참조하고 있는지를 검증	제약 없음

[실습] (MySQL) 무결성 제약조건

부서코드	부서명
AA	총무부
BB	영업부
CC	기획부

사원번호	이름	부서코드
1111	홍길동	AA
2222	임꺽정	AA
3333	박찬호	BB
4444	선동열	BB
5555	차두리	AA
6666	신동엽	BB

--[Quiz] 다음 문법은 실행하라. 실행 가능한가? 그 이유는?

-- 2-4) 다음 문법의 실행결과는? 그 이유는?

```
DELETE  FROM 부서 WHERE 부서코드='DD';
```

-- 2-5) 다음 문법의 실행결과는? 그 이유는?

```
UPDATE 사원 SET  부서코드='AA'    WHERE   사원번호='7777';
```

-- 2-6) 다음 문법의 실행결과는? 그 이유는?

```
UPDATE 사원 SET  부서코드='FF'    WHERE   사원번호='7777';
```

-- 2-7) 다음 문법을 실행하라. 실행 가능한가? 그 이유는?

```
DELETE  FROM 사원 WHERE 사원번호='7777';
```

❖ 참조 무결성 제약조건

제약조건	부모 테이블	자식 테이블
입력	제약 없음	부모테이블에 데이터가 존재하는지 검증
수정	수정하려는 데이터를 자식테이블에서 참조하고 있는지를 검증	부모테이블에 존재하는 다른 데이터로 변경가능
삭제	삭제하려는 데이터를 자식테이블에서 참조하고 있는지를 검증	제약 없음