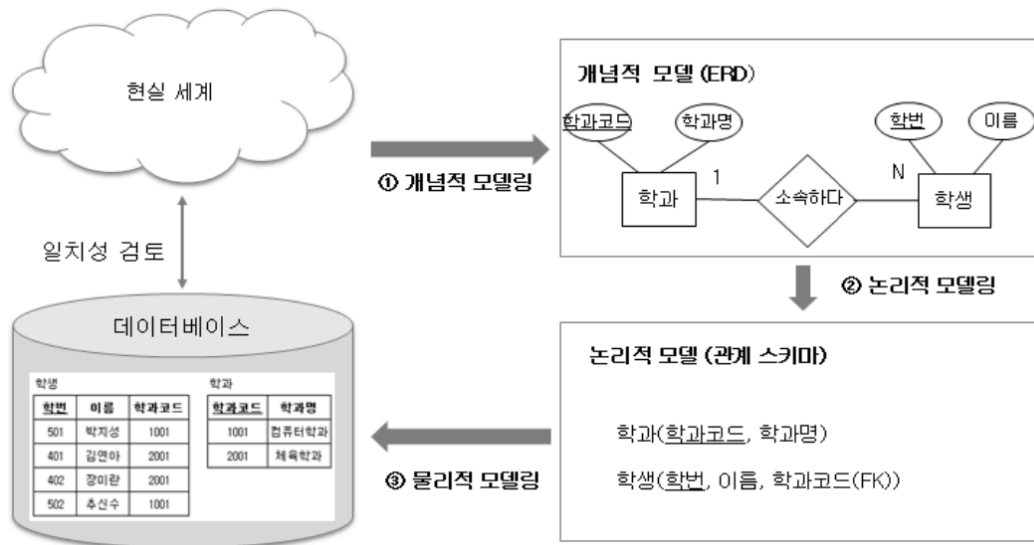




6주차(04/14)

데이터 모델링



1. 개념적 설계(개념적 모델링)
2. 논리적 설계(논리적 모델링)
3. 물리적 설계(물리적 모델링)

매핑 규칙(Mapping Rule) - 7단계

- 개념적 설계에서 작성된 ERD를 데이터베이스 관리시스템에 매핑(Mapping 사상) 하는 것
- 개념적 설계(ERD → 논리적 설계(관계 스키마))
- 마름모(관계) 사라짐

ERD		매핑 규칙(관계 스키마 생성)
엔티티 타입 단일값 속성	정규엔티티	릴레이션 복합속성 → 단일값 속성 (2가지 방법) 유도속성 → 사용자의 선택(관계 스키마에 포함여부)
	약엔티티	소유엔티티의 기본키와 약엔티티의 부분키를 결합하여 기본키를 설정한다
2진 관계	일대일 관계	외래키는 전체참여가 있는 쪽의 엔티티에 포함하고, 참여가 불분명할 때는 어느 쪽에 외래키를 위치시켜도 된다
	일대다 관계	N쪽의 엔티티로 외래키를 위치시킨다
	다대다 관계	별도의 관계 엔티티를 생성하여 각기 2개의 일대다 관계를 설정한다(2개 외래키)
3진 관계		별도의 관계 엔티티와 N개의 외래키
다중값 속성		별도의 새로운 엔티티를 생성하여 새로운 관계를 설정한다(외래키)

엔티티 타입과 속성

- 정규 엔티티 : 릴레이션(테이블) 생성
- 복합 속성 : 단일값 속성으로 바꿈(2가지 방법)
- 유도 속성 : 사용자의 선택(관계 스키마에 포함할지 여부 결정)
- 약 엔티티 : 소유 엔티티의 기본키 + 약 엔티티의 부분키를 결합하여 기본키 설정

2진 관계(두 엔티티 사이 관계)

- 일대일(1 : 1)
 - 외래키를 한 쪽 엔티티에 포함
 - 전체 참여면 반드시 포함
 - 선택 참여면 어느쪽이든 가능
- 일대다(1 : N) : N쪽(다수 엔티티)에 외래키 추가
 - 일대다 === 다대일
- 다대다(N : M) : 별도 관계 엔티티 생성 + 두 개의 일대다 관계로 변환(두 개의 외래키 필요)

3진 관계

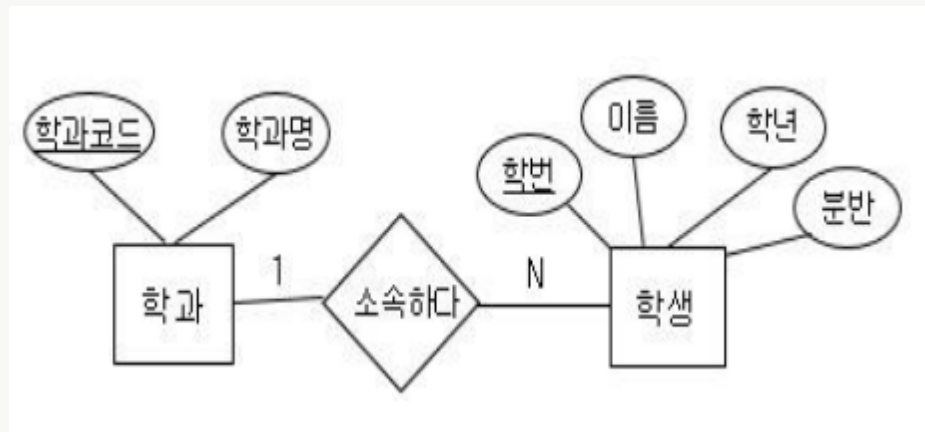
- 별도 엔티티 생성 후 N개의 외래키 설정

다중값 속성

- 별도 새로운 엔티티 생성 + 새로운 관계 설정(외래키 포함)



- 1단계 - 엔티티 타입의 변환
 - 엔티티 → 테이블
- 2단계 - 약한 엔티티 타입의 변환
 - 식별 관계로 연결
- 3단계 - 관계 타입(1 : 1)의 변환
 - 하나로 합칠수도 있고 나눌수도 있음
- 4단계 - 관계 타입(1 : N)의 변환
 - N쪽으로 복사
- 5단계 - 관계 타입(M : N)의 변환
 - 별도의 테이블로 분리
- 6단계 - 다치 애트리뷰트(다중값 속성)의 변환
 - 별도의 테이블로 분리
 - 식별 관계로 연결
- 7단계 - N차 관계 타입의 변환
 - 별도의 테이블로 분리



1. 엔티티 타입의 변환

- 학과(학과코드pk, 학과명)
- 학생(학번pk, 이름, 학년, 분반)

2. 약한 엔티티 타입의 변환(X)

3. 관계 타입(1 : 1)의 변환(X)

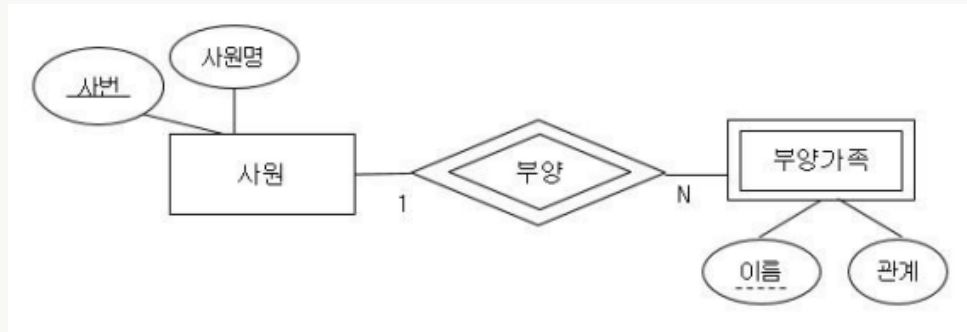
4. 관계 타입(1:N)의 변환

- 학생(학번pk, 이름, 학년, 분반, 학과코드fk)

5. 관계 타입(M : N)의 변환(X)

6. 다치 애트리뷰트(다중값 속성)의 변환(X)

7. N차 관계 타입의 변환(X)



1. 엔티티 타입의 변환

- 직원(사번pk, 사원명)

2. 약한 엔티티 타입의 변환

- 부양가족(사번pk fk, 이름 pk, 관계) ⇒ 부모의 기본키 가져옴

3. 관계 타입(1 : 1)의 변환(X)

4. 관계 타입(1:N)의 변환

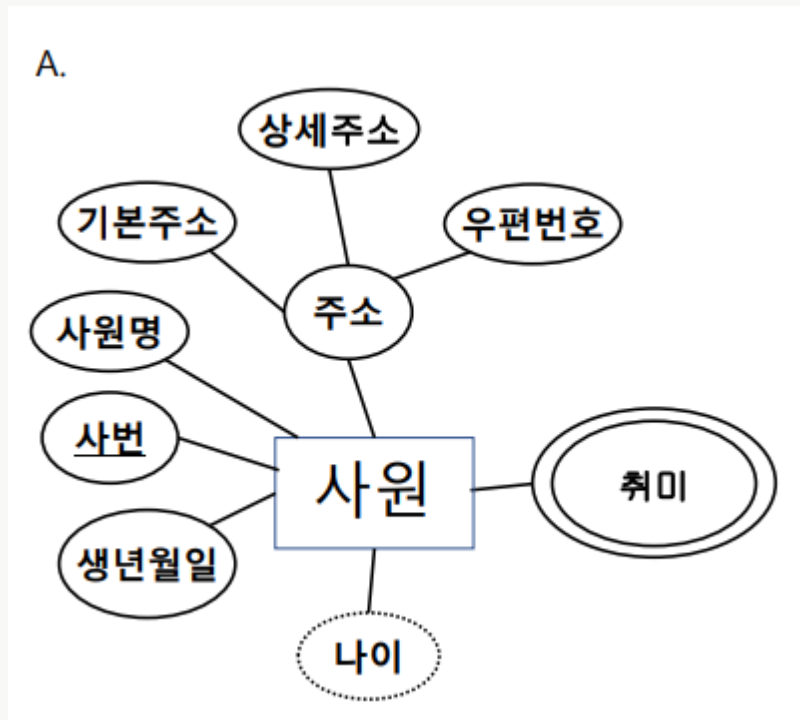
- 2번에서 한 번에 해결

5. 관계 타입(M : N)의 변환(X)

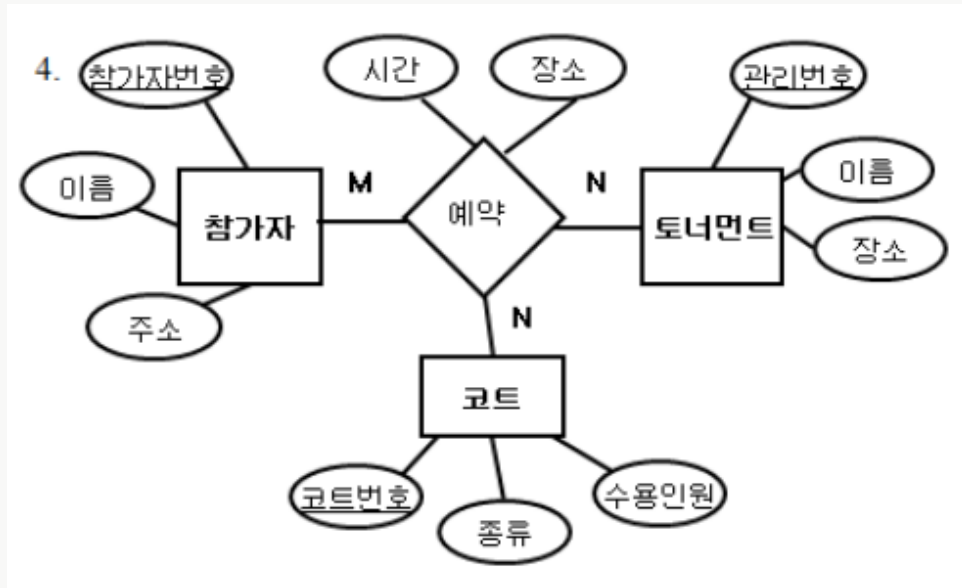
6. 다치 애트리뷰트(다중값 속성)의 변환(X)

7. N차 관계 타입의 변환(X)

- 이중 마름모 : 약한 엔티티 + 부모 엔티티



1. 엔티티 타입의 변환 ⇒ 나이 속성은 계산으로 얻어질 수 있는 값(유도 속성)이므로 생략 가능
 - 사원(사번pk, 사원명, 생년월일, 주소)
 - 사원(사번pk, 사원명, 생년월일, 우편번호, 기본주소, 상세주소)
2. 약한 엔티티 타입의 변환(X)
3. 관계 타입(1 : 1)의 변환(X)
4. 관계 타입(1:N)의 변환(X)
5. 관계 타입(M : N)의 변환(X)
6. 다치 애트리뷰트(다중값 속성)의 변환
 - 사원_취미(사번pk fk, 취미pk) ⇒ 부모 기본키 먼저 작성
 - 약한 엔티티와 비슷하지만 아님.
7. N차 관계 타입의 변환(X)



관계와 바로 연결되는 속성은 관계의 속성

1. 엔티티 타입의 변환

- 참가자(참가자번호pk, 이름, 주소)
- 코트(코트번호pk, 종류, 수용인원)
- 토너먼트(관리번호pk, 이름, 장소)

2. 약한 엔티티 타입의 변환(X)

3. 관계 타입(1 : 1)의 변환(X)

4. 관계 타입(1:N)의 변환(X)

5. 관계 타입(M : N)의 변환(X)

6. 다치 애트리뷰트(다중값 속성)의 변환(X)

7. N차 관계 타입의 변환

- 예약(참가자번호pk fk, 코트번호 pk fk, 관리번호pk fk, 시간, 장소)

[실습]

```
create schema W06 default character set utf8mb4;
use W06;
```

```
-- (MySQL) safe mode 해제
set SQL_SAFE_UPDATES = 0;
```

```
-- 테이블 목록 조회
show tables;
```

```
-- 테이블 삭제
drop table if exists 환자;
drop table if exists 질병;
drop table if exists 신체정보;
drop table if exists 학생;
```

```
-- ***** 질병, 환자 table *****
```

```
-- 질병(질병코드, 질병명, 증상)
```

```
create table 질병 (
```

```
    질병코드 char(3),
```

```
    질병명 varchar(15),
```

```
    증상 varchar(20)
```

```
);
```

```
alter table 질병
```

```
    add constraint pk_질병
```

```
        primary key(질병코드);
```

```
insert into 질병 values('A01', '뇌졸중', '어지럼증');
```

```
insert into 질병 values('A02', '콜레라', '설사');
```

```
insert into 질병 values('A03', '기관지염', '발열');
```

```
insert into 질병 values('A04', '장티푸스', '발열');
```

```
select * from 질병;
```

```
-- 환자(환자번호, 이름, 질병코드, 나이)
```

```
create table 환자 (
```

```
    환자번호 char(5),
```



```

이름 varchar(10),
질병코드 char(3),
나이 int
);

alter table 환자
add constraint pk_환자
primary key(환자번호);

alter table 환자
add constraint fk_질병_TO_환자
foreign key(질병코드)
references 질병(질병코드);

insert into 환자 values('P1001', '김철수', 'A01', 30);
insert into 환자 values('P1002', '양길현', 'A03', 29);
insert into 환자 values('P1003', '임영수', 'A01', 50);
insert into 환자 values('Q1001', '박한나', 'A04', 40);

select * from 환자;

```



```

-- ***** 학생, 신체정보 table *****
-- 학생(학번, 이름, 주소, 전공)
create table 학생 (
    학번 char(5),
    이름 varchar(10),
    주소 varchar(15),
    전공 varchar(10)
);

alter table 학생

```

```

add constraint pk_학생
primary key(학번);

insert into 학생 values('21001', '김철수', '서울', '영문학');
insert into 학생 values('21002', '양길현', '인천', '컴퓨터');
insert into 학생 values('21003', '임영수', '광주', '화학');
insert into 학생 values('21004', '박한나', '부산', '수학');

select * from 학생;

-- 신체정보(학번, 키, 몸무게, 혈액형)
create table 신체정보 (
    학번 char(5),
    키 int,
    몸무게 int,
    혈액형 varchar(5)
);


alter table 신체정보
add constraint pk_신체정보
primary key(학번);

alter table 신체정보
add constraint fk_학생_TO_신체정보 foreign key(학번)
references 학생(학번);

insert into 신체정보 values('21001', 175, 70, 'A');
insert into 신체정보 values('21002', 169, 65, 'B');
insert into 신체정보 values('21003', 180, 60, 'O');
insert into 신체정보 values('21004', 170, 85, 'B');


select * from 신체정보;

```

학생		학생	
	학번	학번	Domain
	이름	이름	Type
	주소	주소	Domain
	전공	전공	Type

신체정보		신체정보	
	키	키	Domain
	몸무게	몸무게	Type
	혈액형	Field3	Domain

관계 설정 이전

학생		학생	
	학번	학번	Domain
	이름	이름	Type
	주소	주소	Domain
	전공	전공	Type

신체정보		신체정보	
	학번	학번	Domain
	키	키	Type
	몸무게	몸무게	Domain
	혈액형	Field3	Type

관계 설정 이후