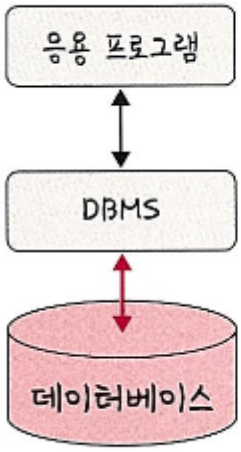
4.1 데이터베이스의 기본

- 데이터 베이스는 일정한 규칙, 혹은 규악을 통해 구조화되어 저장되는 데이터의 모음이다.
- 해당 데이터베이스를 제어 관리하는 통합 시스템을 DBMS라고 하며 데이터베이스 안에 있는 데이터들은 특정 시스템마다 정의된 쿼리 언어를 통해 삽입 삭제 수정 조회등을 수행할 수 있다.
- 데이터베이스는 실시간 접근과 동시 공유가 가능하다.

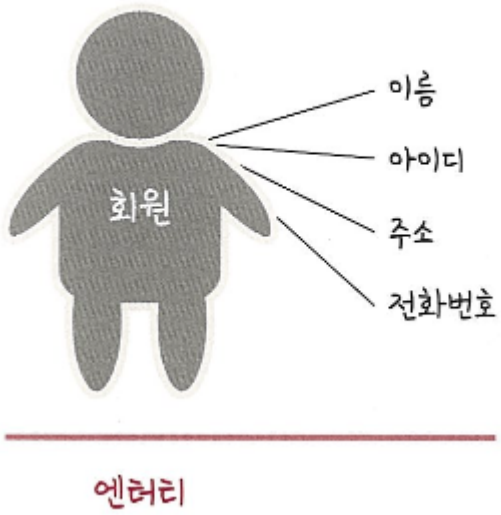
▼ 그림 4-1 데이터베이스와 DBMS



4.1.1 엔터티

- 엔터티는 사람,장소,물건,사건,개념 등 여러 개의 속성을 지닌 명사를 의미한다.

▼ 그림 4-2 엔터티



- 서비스 요구사항에 맞춰 속성이 정해진다.

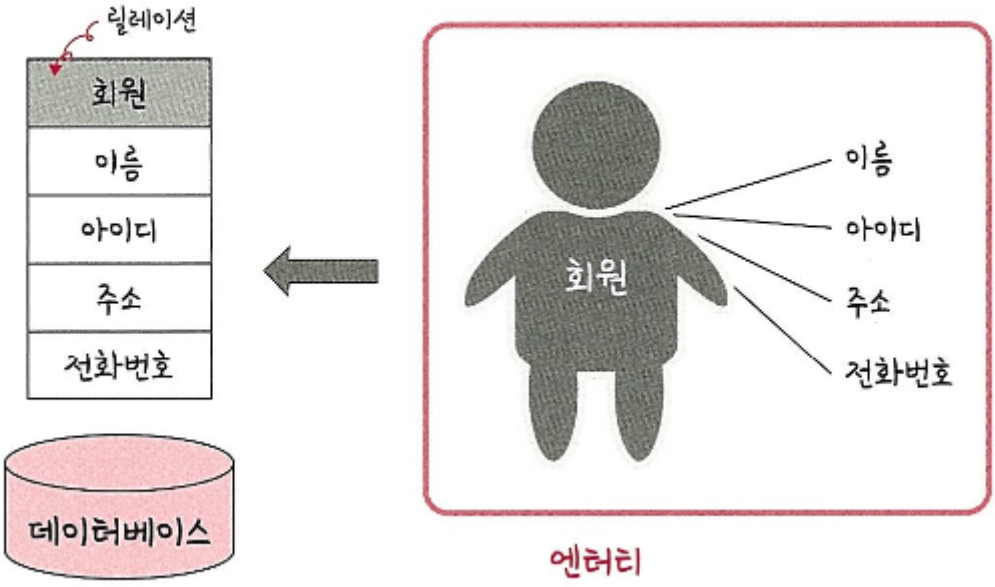
약한 엔터티와 강한 엔터티

- a가 혼자서는 존재하지 못하고 b의 존재 여부에 따라 종속적이면 a는 약한 엔터티이고 b는 강한 엔터티이다.

4.1.2 릴레이션

- 릴레이션은 데이터베이스에서 정보를 구분하여 저장하는 기본단위이다.
- 엔터티에 관한 데이터를 데이터베이스는 릴레이션 하나에 담아서 관리한다.

▼그림 4-3 릴레이션

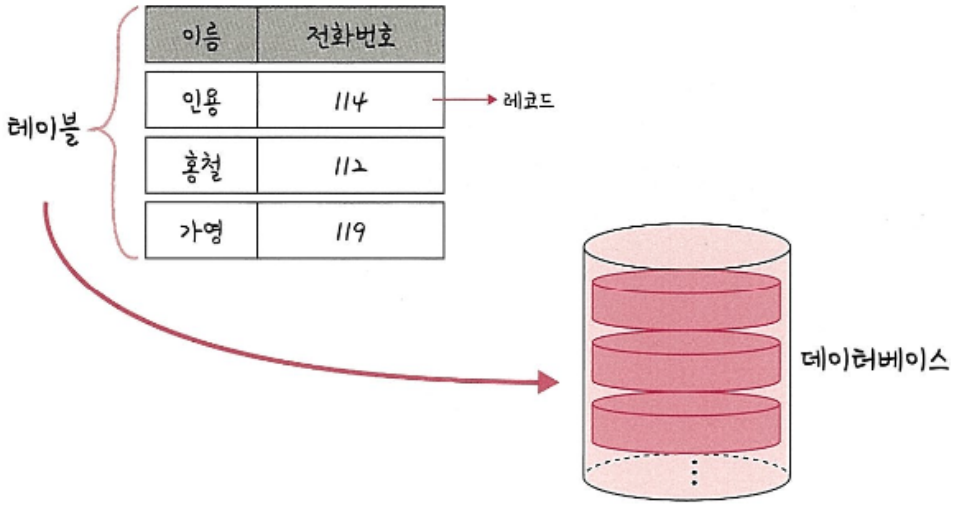


- 회원이라는 엔터티가 데이터베이스에서 관리될때 릴레이션으로 변화된것을 볼 수 있다.
- 릴레이션은 관계형 데이터베이스에선 테이블이라 하며 NoSQL 디비에선 컬렉션이라 한다.

테이블과 컬렉션

- 데이터베이스의 종류는 크게 관계형 데이터베이스와 NoSQL 데이터베이스로 나눌 수 있다.
- 이중 대표적인 관계형 db인 MySQL과 대표적인 NoSQL **데이터베이스인 MongoDB를 예로 들면**
- MySQL의 구조는 레코드-테이블-db로 이루어져 있고 NoSQL 데이터베이스의 구조는 도큐먼트- 컬렉션 - db로 이루어져 있다.

▼그림 4-4 레코드-테이블-데이터베이스의 구조



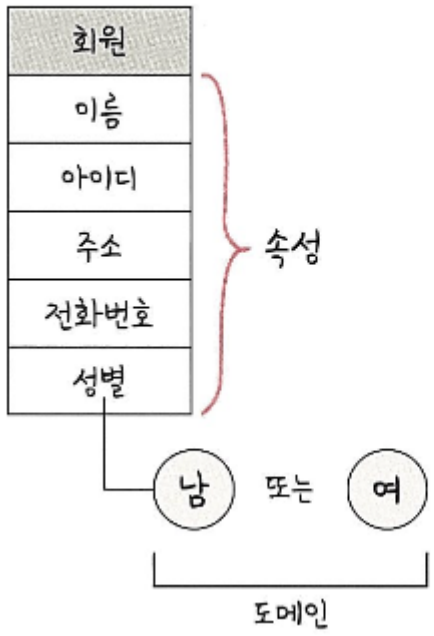
4.1.3 속성

- 속성은 릴레이션에 관리하는 구체적이며 고유한 이름을 갖는 정보이다.
- 차 라는 엔터티의 속성으로는 차 넘버, 바퀴수 차 색, 차종이 있다.
- 서비스의 요구사항을 기반으로 관리해야 할 필요가 있는 속성들만 엔터티의 속성이 된다.

4.1.4 도메인

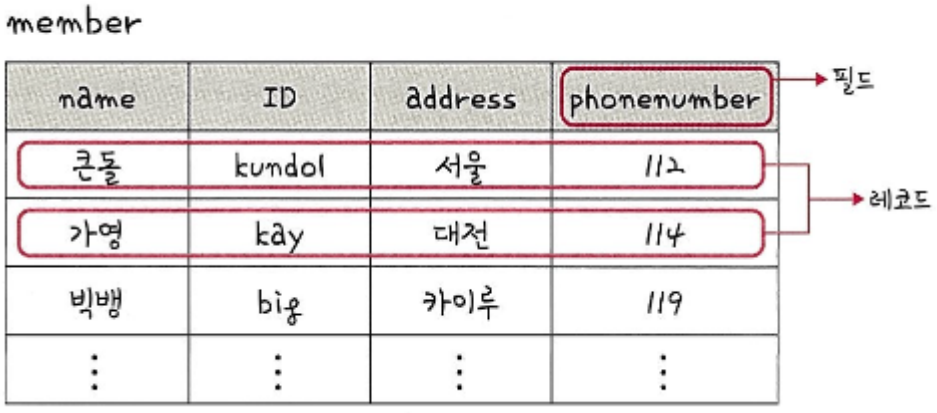
- 도메인이란 릴레이션에 포함된 각각의 속성들이 가질 수 있는 값의 집합을 말한다.
- 예를 들어 성별이라는 속성이 있다면 이 속성이 가질 수 있는 값은 남,여 라는 집합이 된다.

▼ 그림 4-5 속성과 도메인



4.1.5 필드와 레코드

▼ 그림 4-6 필드와 레코드



- 회원이라는 엔터티는 member라는 테이블로 속성인 이름, 아이디 등을 가지고 있으며
- name, id, address등의 필드를 가진다. 그리고 이 테이블에 쌓이는 행 단위의 데이터를 레코드라 한다.

필드 타입

- 필드는 타입을 갖는다. 예를 들어 이름은 문자열이고 전화보호는 숫자이다.
- 이러한 타입들은 DBMS마다 다르며 MySQL 기준으로 설명하면

숫자 타입

숫자 타입으로는 TINYINT, SMALLINT, MEDIUMINT, INT, BIGINT 등이 있습니다.

▼ 표 4-1 MySQL 숫자 타입

타입	용량(바이트)	최솟값(부호 있음)	최솟값(부호 없음)	최댓값(부호 없음)	최댓값(부호 있음)
TINYINT	1	-128	0	127	255
SMALLINT	2	-32768	0	32767	65535
MEDIUMINT	3	-8388608	0	8388607	16777215
INT	4	-2147483648	0	2147483647	4294967295
BIGINT	8	-263	0	263-1	264-1

날짜 타입

날짜 타입으로는 DATE, DATETIME, TIMESTAMP등이 있다.

DATE

날짜 부분은 있지만 시간 부분은 없는 값에 사용됩니다. 지원되는 범위는 1000-01-01~9999-12-31입니다. 3바이트의 용량을 가집니다.

DATETIME

날짜 및 시간 부분을 모두 포함하는 값에 사용됩니다. 지원되는 범위는 1000-01-01 00:00:00에서 9999-12-31 23:59:59입니다. 8바이트의 용량을 가집니다.

TIMESTAMP

날짜 및 시간 부분을 모두 포함하는 값에 사용됩니다. 1970-01-01 00:00:01에서 2038-01-19 03:14:07까지 지원합니다. 4바이트의 용량을 가집니다.

문자 타입

문자 타입으로는 CHAR, VARCHAR, TEXT, BLOB, ENUM, SET이 있습니다.

CHAR와 VARCHAR

CHAR 또는 VARCHAR 모두 그 안에 수를 입력해서 몇 자까지 입력할지 정합니다. 예를 들어 CHAR(30)이라면 최대 30글자까지 입력할 수 있습니다.

CHAR는 테이블을 생성할 때 선언한 길이로 고정되며 길이는 0에서 255 사이의 값을 가집니다. 레코드를 저장할 때 무조건 선언한 길이 값으로 '고정'해서 저장됩니다.

VARCHAR는 가변 길이 문자열입니다. 길이는 0에서 65,535 사이의 값으로 지정할 수 있으며, 입력된 데이터에 따라 용량을 가변시켜 저장합니다. 예를 들어 10글자의 이메일을 저장할 경우 10글자에 해당하는 바이트 + 길이기록용 1바이트로 저장하게 됩니다. VARCHAR(10000)으로 선언했음에도 말이죠.

그렇기 때문에 지정된 형태에 따라 저장된 CHAR의 경우 검색에 유리하며, 검색을 별로 하지 않고 유동적인 길이를 가진 데이터는 VARCHAR로 저장하는 것이 좋습니다.

TEXT와 BLOB

- 두 개의 타입 모두 큰 데이터를 저장할때 쓰는 타입이다.
- TEXT는 큰 문자열 저장에 쓰며 주로 게시판의 본문을 저장할때 쓴다.
- BLOB은 이미지, 동영상 등 큰데이터 저장에 쓴다.
- 그러나 보통은 아마존의 이미지 호스팅 서비스인 S3를 이용하는 등 서버에 파일을 올리고 파일에 관한 경로를 VARVHAR로 저장한다.

enum과 set

ENUM은 ENUM('x-small', 'small', 'medium', 'large', 'x-large') 형태로 쓰이며, 이 중에서 하나만 선택하는 단일 선택만 가능하고 ENUM 리스트에 없는 잘못된 값을 삽입하면 빈 문자열이 대신 삽입됩니다. ENUM을 이용하면 x-small 등이 0, 1 등으로 매핑되어 메모리를 적게 사용하는 이점을 얻습니다. ENUM은 최대 65,535개의 요소들을 넣을 수 있습니다.

SET은 ENUM과 비슷하지만 여러 개의 데이터를 선택할 수 있고 비트 단위의 연산을 할 수 있으며 최대 64개의 요소를 집어넣을 수 있다는 점이 다릅니다.

참고로 ENUM이나 SET을 쓸 경우 공간적으로 이점을 볼 수 있지만 애플리케이션의 수정에 따라 데이터베이스의 ENUM이나 SET에서 정의한 목록을 수정해야 한다는 단점이 있습니다.

4.1.6 관계

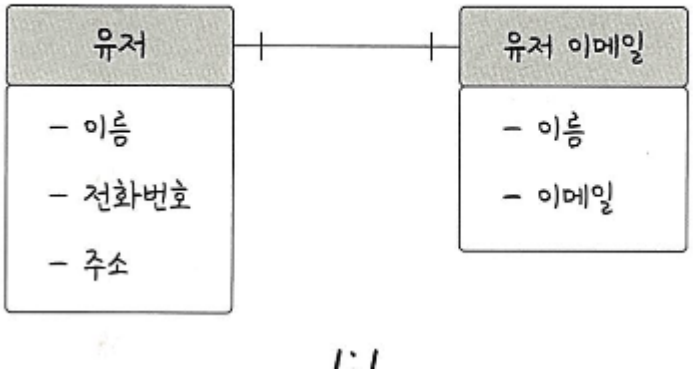
- 데이터베이스에 테이블은 하나만 있는 것이 아니다. 여러 개의 테이블이 있고 이러한 테이블은 서로의 관계가 정의되어 있다.
- 이러한 관계를 관계 화살표로 나타낸다.

▼ 그림 4-9 관계화살표

1:1 관계

- 예를 들어 유저당 유저 이메일은 한개씩 있다 → 1:1관계

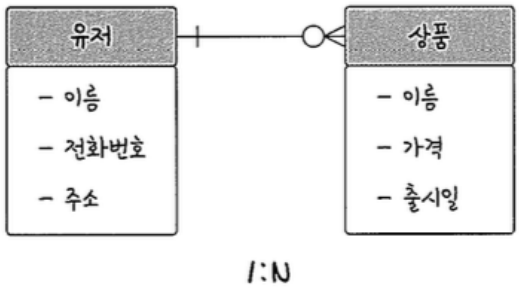
▼ 그림 4-10 1:1 관계



1:N관계

예를 들어 쇼핑몰을 운영한다고 해봅시다. 한 유저당 여러 개의 상품을 장바구니에 넣을 수 있겠죠? 이 경우 1:N 관계가 됩니다. 물론 하나도 넣지 않는 0개의 경우도 있으니 0도 포함되는 화살표를 통해 표현해야 합니다.

▼ 그림 4-11 1:N 관계

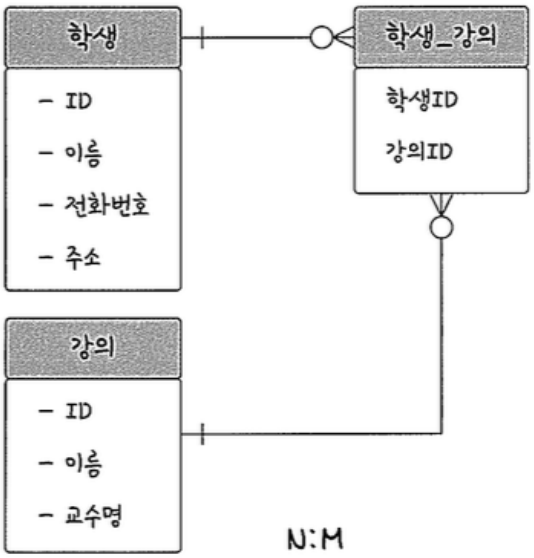


이렇게 한 개체가 다른 많은 개체를 포함하는 관계를 말합니다.

N:M관계

학생과 강의의 관계를 정의하면 어떻게 될까요? 학생도 강의를 많이 들을 수 있고 강의도 여러 명의 학생을 포함할 수 있습니다. 이 경우 N:M이 됩니다.

▼그림 4-12 N:M 관계

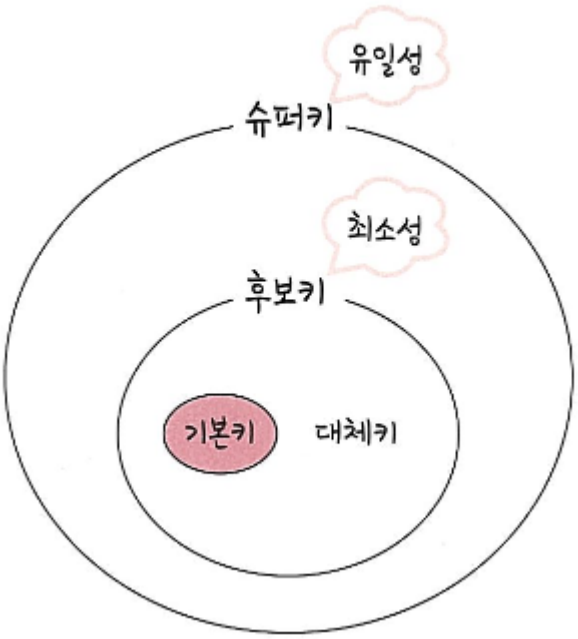


- 중간에 학생_강의 라는 테이블이 끼어 있다. N:M테이블은 두개를 직접적으로 연결해서 구축하지 않고, 1:N, 1:M이라는 관계를 갖는 테이블 두개로 나눠서 설정한다.

4.1.7 키

- 테이블 간의 관계를 조금 더 명확하게 하고 테이블 자체의 인덱스를 위해 설정된 장치로 기본키, 외래키, 후보키, 슈퍼키, 대체키가 있다.
-

▼ 그림 4-13 키 간의 관계



- 슈퍼키는 유일성이 있다.
- 후보키는 최소성까지 갖춘 키 이다. 후보키 중에서 기본키로 선택되지 못한 키는 대체키가 된다.
- 유일성은 중복되는 값은 없으며 최소성은 필드를 조합하지 않고 최소 필드만 써서 키를 형성할 수 있는 것을 말한다.

기본키

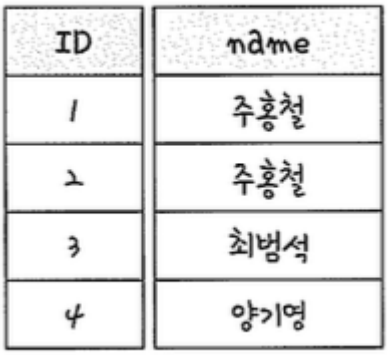
- 기본키는 줄여 PK또는 프라이머리 키라고 부르며 유일성과 최소성을 만족하는 키이다.

▼ 그림 4-14 기본키가 안 되는 키

ID	name
PDT-0001	홍철이의 따스한 점퍼
PDT-0002	제호의 BMW
PDT-0002	제호의 BMW
PDT-0003	종선이의 벤츠

- 이는 테이블의 데이터 중 고유하게 존재하는 속성이며 기본키에 해당하는 데이터는 앞의 그림처럼 ID처럼 중복되어서는 안된다.

▼ 그림 4-15 기본키가 되는 키



ID	name
1	주홍철
2	주홍철
3	최범석
4	양기영

- ID는 기본키로 설정할 수 있다. 물론 iD,name이라는 복합키를 기본키로 설정할 수 있지만 그렇게 되면 최소성을 만족하지 않는다.
- 기본키는 자연니 또는 인조키 중에 골라 설정한다.

자연키

- 예를들어 유저 테이블을 만든다고 가정하면 주민번호, 이름, 성별들의 속성이 있다.
- 이중 이름 성별등은 중복된 값이 들어올 수 있으므로 부적절하고 남은것은 주민번호이다.
- 이런식으로 중복된 값들을 제외하면 중복되지 않는 것을 자연스럽게뽑다가 나오는 키를 자연키라고 한다.
- 자연키는 언제가는 변하는 속성을 가진다.

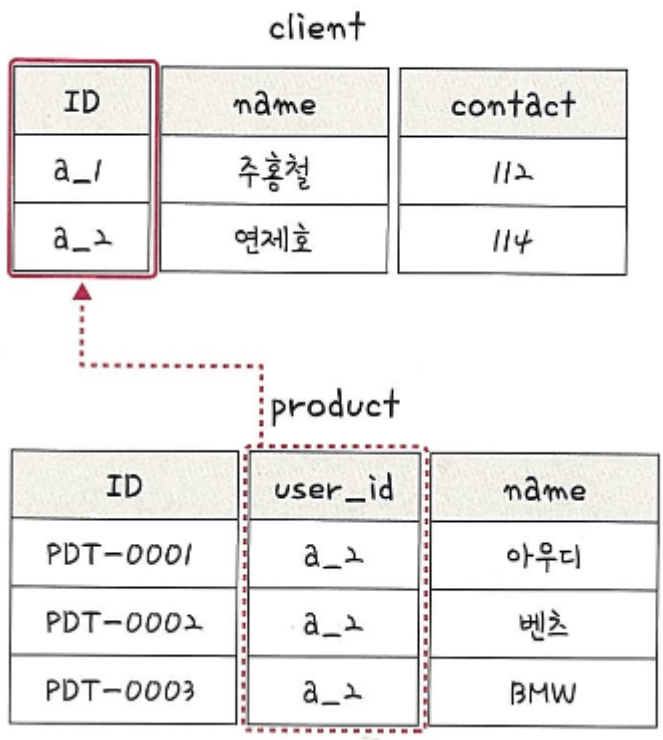
인조키

- 유저테이블을 만들때 인위적으로 유저 아이디를 부여한다.
- 이를 통해 고유 식별자가 생겨난다.
- 오라클은 sequence, MySQL은 auto increment등으로설정한다.
- 이렇게 인위적으로 생성한 키를 인조키라 하며 자연키와는 대조적으로 변하지 않는다.
- 보통 기본키는 인조키로 설정한다.

외래키

- 외래키는 FK라고 하며, 다른 테이블의 기본키를 그대로 참조하는 값으로 개체와의 관계를 식별하는데 사용한다.

▼ 그림 4-16 외래키



- 외래키는 중복되어 괜찮다. 앞의 그림을 보면 client라는 테이블의 기본키인 id가 product라는 테이블의 user_id라는 외래키로 설정될 수 있음을 보여주며, 중복되는것을 볼 수 있다.

후보키

후보키(candidate key)는 기본키가 될 수 있는 후보들이며 유일성과 최소성을 동시에 만족하는 키입니다.

대체키

대체키(alternate key)는 후보키가 두 개 이상일 경우 어느 하나를 기본키로 지정하고 남은 후보키들을 말합니다.

슈퍼키

슈퍼키(super key)는 각 레코드를 유일하게 식별할 수 있는 유일성을 갖춘 키입니다.