연관관계 매핑

객체는 참조를 사용하여 연관관계를 맺고 / 컴포넌트는 카메라를 통해 / DB는 외래키(FK)를 사용한다.
하지만 참조와 외래키는 다른 특징을 갖는다.

① 방향 : 단방향 vs 양방향 (객체참조)

- FK를 통해 양쪽 모두 JOIN 가능.

⇒ 참조필드가 있는 객체만 객체를 참조하는것이 가능.

한쪽만 있는경우 : 단방향 / 양쪽 있는 경우 : 양방향.

⇒ 무조건 양방향으로 해야 하나? NO!

ex) USER 엔티티 (회원관계)

⇒ 많은 다른 데이블과 연관관계 맺음 ⇒ 복잡해짐 ⇒ 잘정해야 한다.

(기본적으로 단방향 매핑후 ⇒ 나중에 역방향으로 객체탐색이 필요한 경우 추가.)

② 연관관계의 주인.

양방향 관계 시 주인을 정해 줘야함.

- 주인의 역할: 두객체 사이에서 "조회, 수정, 삭제" 가 가능
- 주인 x 역할: 조회만 가능.

-주인 설정 : 주인이 x 객체에서 "mappedBy" 속성을 사용해 주인을 지정 해주면 된다.

(* 외래키 있는곳 : 주인으로 간주)?

Why 지정 해야함?

A ⟷ B (양방향) : 수정 ⇒ A or B? 어디일까?

: 두개 다 맞다 하.지.만! JPA 입장에서 혼란: FK를 수정? PK를 수정?

⇒ PK 수정할 때만 FK를 수정해 줬다 라고 정해야한다.

주인만 제어 해야함?

Yes 하.지.만! 객체 입장에서는 두개다 변경이 좋음.

Why? 두개의 참조를 사용하는 순수 객체는 데이터 동기화를 해줘야함.

③ 다중성.

DB 기준으로 다중성을 결정한다.

- 연관관계는 대칭성을 가진다

1:N↔N:1 / N:1↔1:N / N:N↔N:N

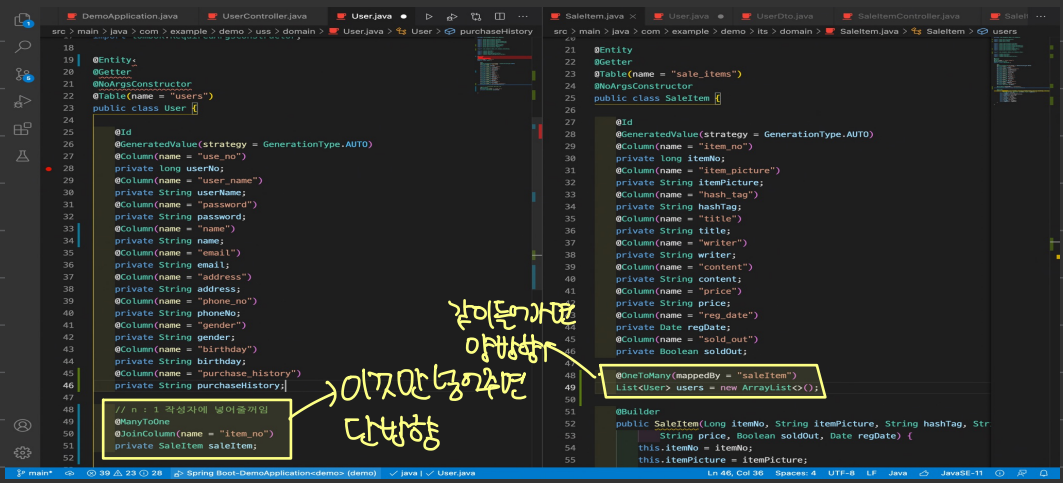
① N:1 @ManyToOne / @JoinCulumn : @OneToMany(mappedby="")

판매상품 (1) 에는 여러개의 유저 속성 (N)을 넣을 수 있다.

하나의 유저는 1개의 판매 상품에만 작성 할 수 있다.

⇒ N (유저) : 1 (판매상품) 관계이다.

(외래키를 N쪽에 두는것이 일반적)



② 1:N @OneToMany, @JoinColumn (양방향은 존재 X)

1:N 단방향은 주로 사용 X. : FK는 대부분 N쪽에서관리 / 1:N은 1쪽에서 객체를 가짐

⇒ PK 수정/초기/삭제가 FK 쪽도 같이 변경된다.

→ Debug가 힘들어짐

⇒ N:1 로 양방향 관계 만들어서 쓰자.

유용한곳 : JPA 같은 타입을 사용하는것은 대신해서 사용할때 유용.

③ 1 : 1 @OneToOne, @JoinColumn : @OneToOne(mappedby "")

주테이블에 FK 또는 대상 테이블 FK 넣을 수 있다.

⇒ A 테이블 (PK, FK) (주) : (대상) B테이블 (PK)

그렇다면 어디에서 FK를 관리?

: 일반적으로 변경이 빈번 (1 → N) 이될 가능성↑ 곳에 두는것이 좋음.

1 쪽에 두면 이미 객체 참조를 갖고 있기 때문에 성능상 이득

④ N : N

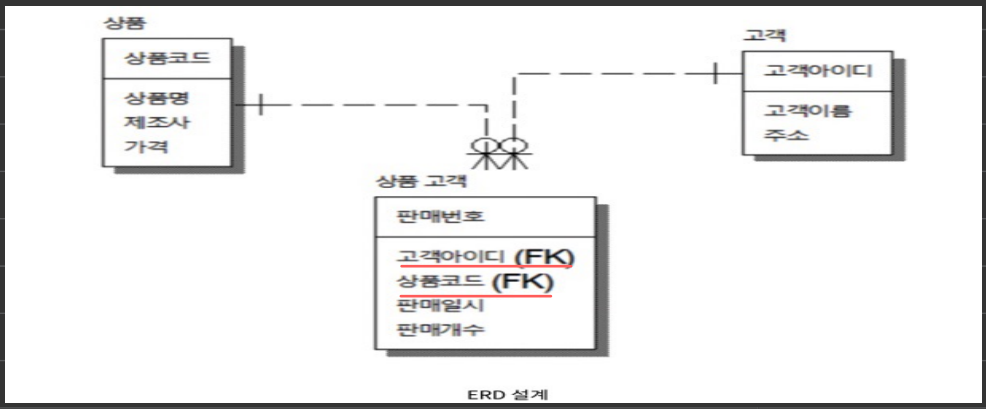
⇒ 주로 사용 X

자기도 모르는 복잡한 쿼리 발생 → 두객체의 외래키만 저장되어 문제 발생↑

⇒ N:N ⇒ N:1 or 1:N 로 풀어서 만들어야 한다.

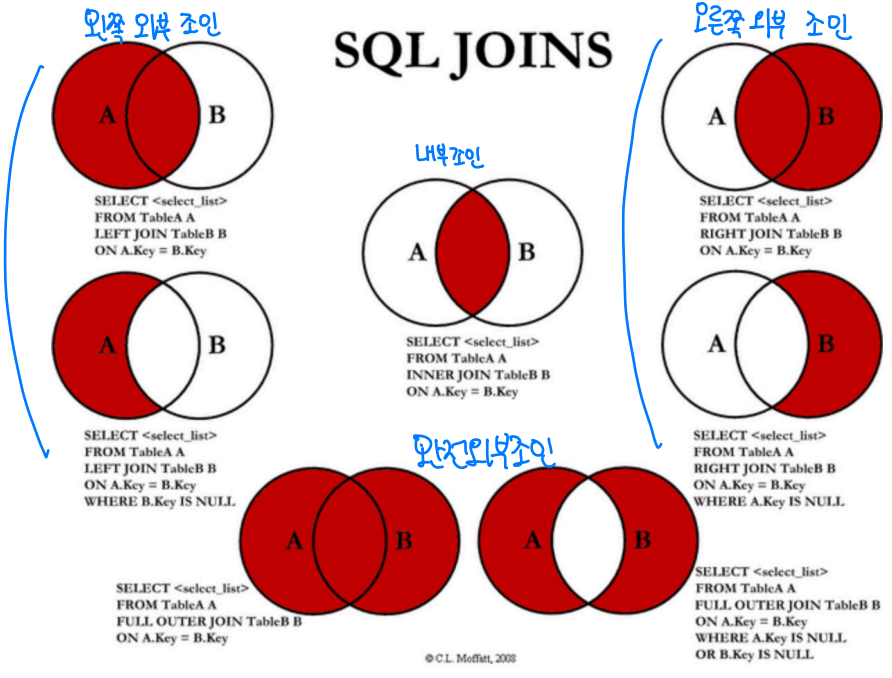
유지보수, Debug 시 대처↑

↳ Association Entity (교차 엔티티) 를 생성하여 해결한다.



ERD 설계

SQL 조인 쉽게 이해하기 위한 다이어그램입니다.



내부조인

같은 이름의 컬럼이 여러테이블에 있을경우 '테이블/컬럼명' 형태로 명시

INNER 키워드 생략가능

[where 검색조건] 추가하면 조인된 값에서 해당 조건에 맞는 결과만 출력

외부조인

① 왼쪽외부 조인 (LEFT[OUTER] JOIN)

- OUTER 생략 가능
- 검색조건 추가시 조건에 맞는 결과만 출력

② 오른쪽외부조인 (RIGHT [OUTER] JOIN)

- OUTER 생략가능 / 검색 조건 추가가능.

③ 완전 외부조인

- ''

교차조인 (CROSS JOIN)

조인 조건이 없는 모든 데이터 조합을 추출 → (ON X)

셀프조인

SELECT A.컬럼1, A.컬럼2..., B.컬럼1, B.컬럼2

FROM 테이블 A [INNER] JOIN 테이블 B ON 조인조건

[WHERE 검색조건] ;

- 같은 테이블 명을 쓰고 별칭만 A. B 로 같이 다르게함
- 검색 조건 추가 → 조건에 맞는 결과만 출력