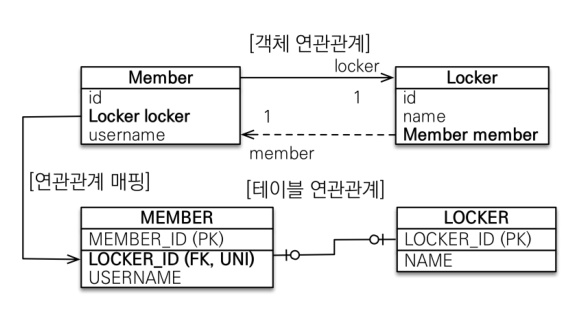
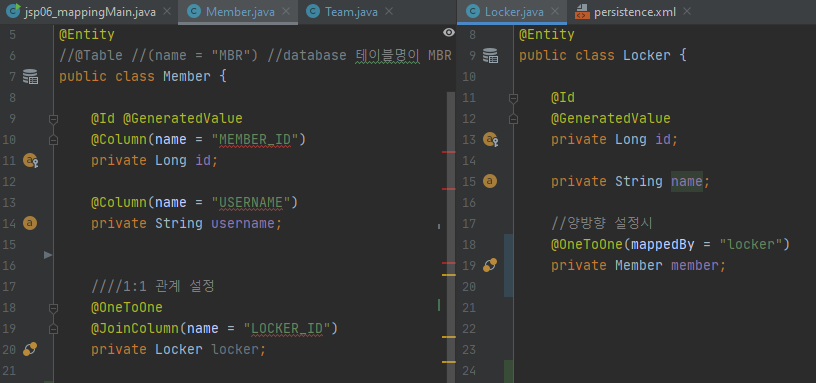
1:1 양방향 1.(Member연관관계 매핑)



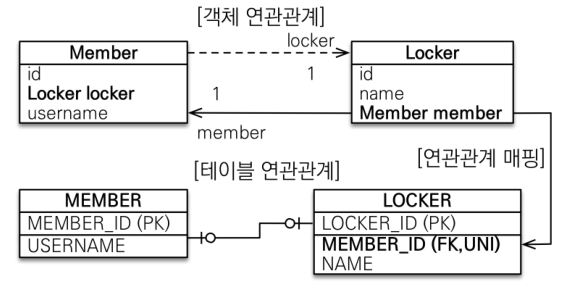
* 주 테이블(Member 테이블) 에 외래키 양방향을 설정한다.



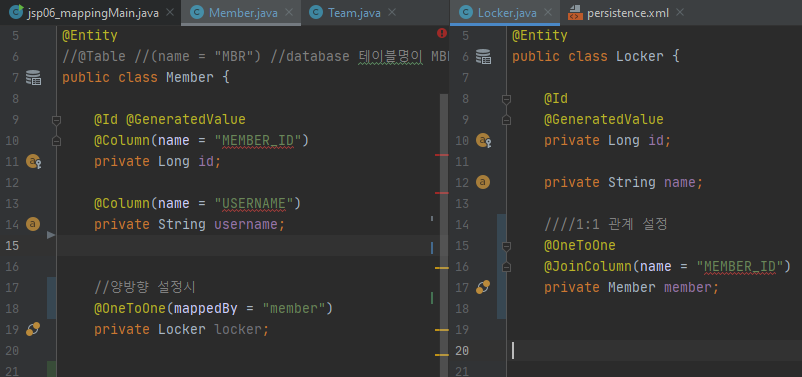
양방향이므로 연관관계의 주인을 정해야한다. MEMBER테이블에 외래키를 가지고있으니, member.locker가 연관관계의 주인. 따라서 반대 매핑인 사물함의 Locker테이블의 locker.member는 mappedBy를 사용해서 연관관계의 주인이 아니라고 설정한다. LOCKER 필드는 읽기전용이 된다.

1:1 양방향 2.(Locker연관관계 매핑)

* 주 테이블(Locker 테이블) 에 외래키 양방향을 설정한다.

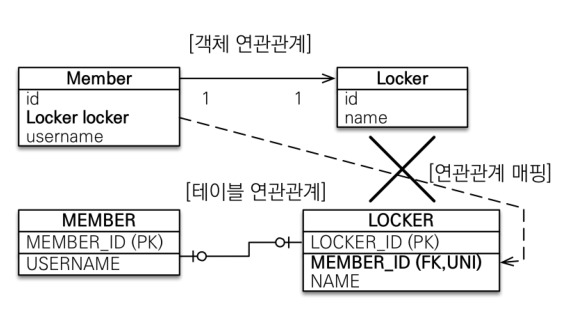


* 주 테이블(Locker 테이블) 에 외래키 양방향을 설정한다.



직전 상황과 반대로 Member엔티티 대신에 대상 엔티티의 LOCKER 를 연관관계의 주인으로 만들어서 LOCKER 테이블의 외래키를 관리하도록함.

1:1에서 대싱테이블에 왜래키



일대일 대상 테이블에 외래키 단방향의 경우

1. 단방향 관계는 JPA 에서 지원하지 않는다.
2. JPA2.0 부터 일대다 단방향 관계는 지원하나, 일대일 단방향의 이러한 매핑은 지원하지 않음으로 주의.

---- 정리

1. 주 테이블에 왜래키가 있으면

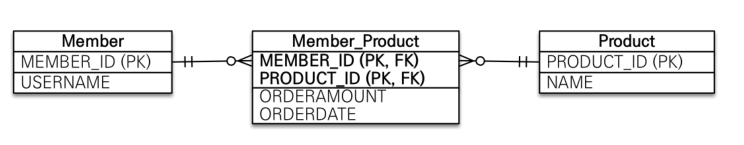
* 테이블 연관관계를 설정할 때 주 객체를 설정하고, 외래키를 설정하여 대상 테이블을 찾는 방식으로 데이터를 조회해야 한다.
* 데이터를 INSERT 시 값이 없으면 왜래키에 NULL 이 발생할수 있으므로 테이블 생성시 해당 왜래키에는 NULL 을 허용해야한다. (DB의 입장에서는 NULL이 임의로 들어갈수있어서 치명적일수 있다. )

1. 대상 테이블에 왜래키가 있으면

* 주테이블에 왜래키를 설정하는 방식이 아닌 반대방식. (대상테이블에 왜래키를 설정)
* 기존DB개발자가 선호하는 방식이며 NULL을 허용할 필요도 없다.
* 주테이블과 대상테이블을 1;1에서 1;N으로 변경시 테이블 구조 유지가 가능하다.

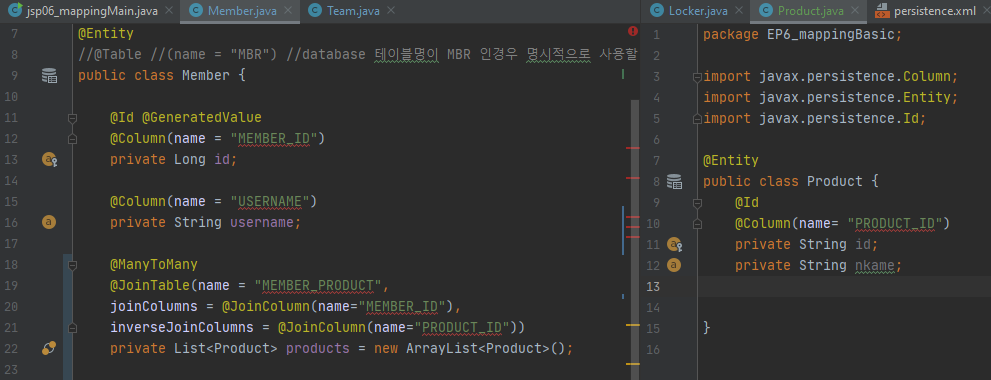
다대다 N:M

정규화된 테이블 2개로 다대다관계를 표현할 수 없기 때문에 제3의 테이블을 생성해서 두 테이블간의 관계를 FK 로 나타내는 방식이다.



구현방법은 @ManyToMany를 사용하며 동시에 다른테이블은 @JoinTable 로 연결하여 테이블을 지정한다.

* 다대다 단방향의경우



@Jointable.name=연결 테이블 지정. (MEMBER\_PRODUCT테이블 선택

@Jointable.joinColumns=현재 방향인 회원과 매핑할 조인칼럼 정보를 지정(MEMBER\_ID)

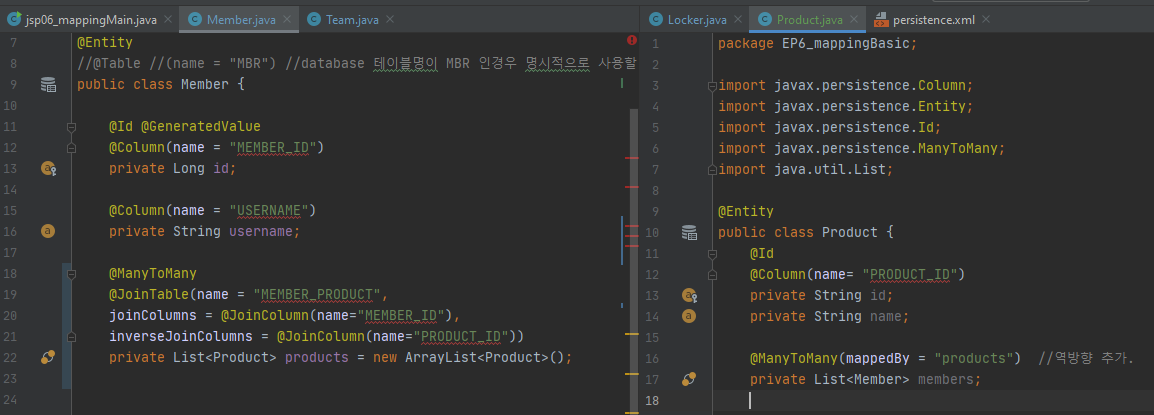
@joinTable.inverseJoinColumns=반대방향인 상품과 매핑한 조인칼럼 정보를 지정.(PRODUCT\_ID)

* Insert : Product테이블에 상품 persist.
* member테이블에 회원 정보 넣고 상품 테이블에 있는 name을 .add 해주면
* product, member, member\_product 테이블 순으로 insert 진행된다.

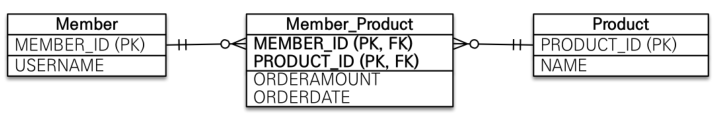
테이블 조회시 member\_product 테이블에 저장해 두었던 상품이 저장되어있다.

다대다 양방향

@ManyToMany 로 매핑되있는 곳을 연관관계의 주인으로 설정해 두었다면, 양방향으로 설정할때는 반대 관계에 있는 테이블에 @MappedBy로 연관관계 주인을 지정할 수 있다.



그런데 여기서 Member\_product에 칼럼이 추가될 경우에는 다른방법으로 진행을 해야한다.

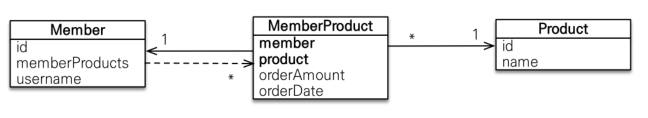
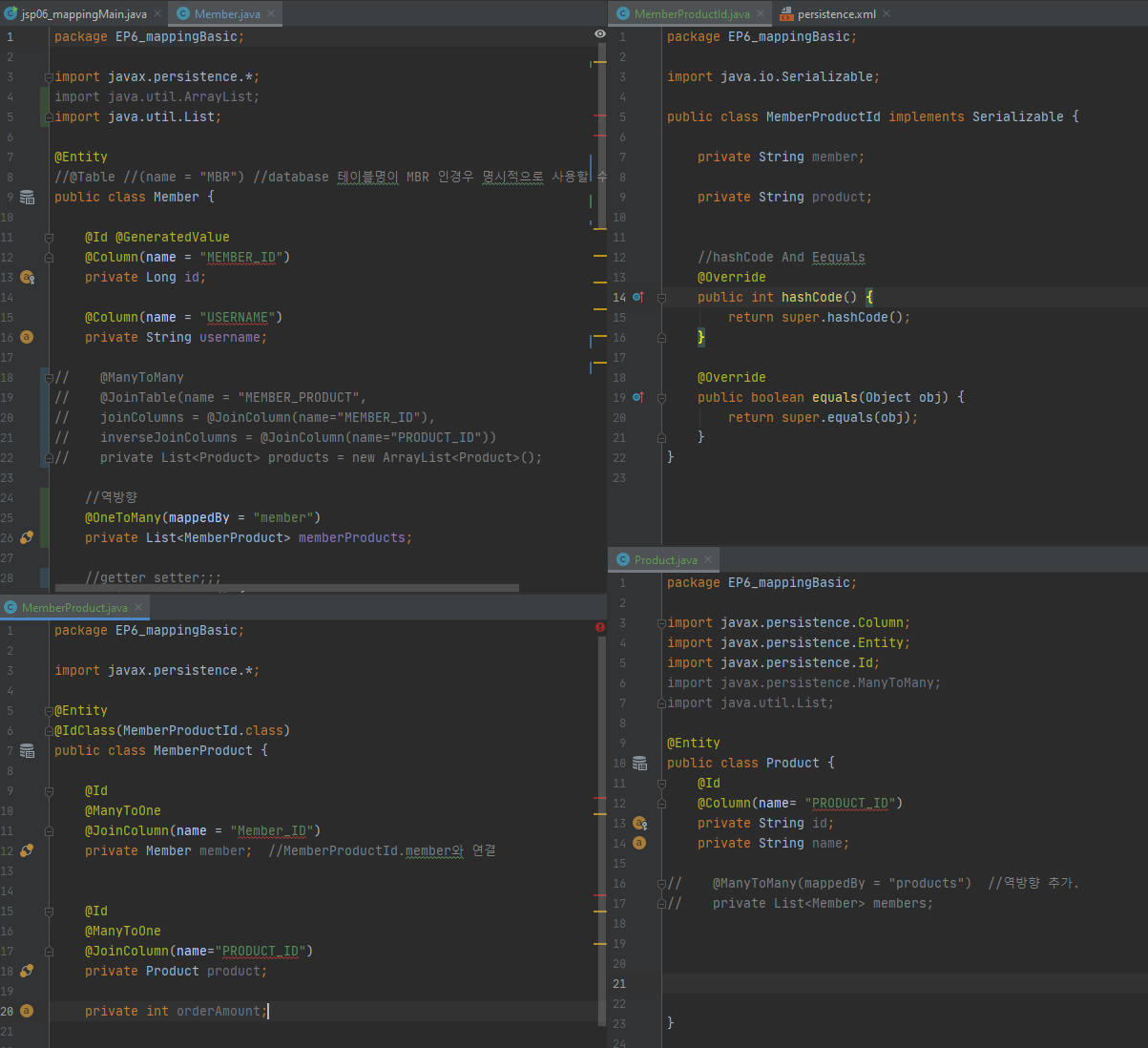


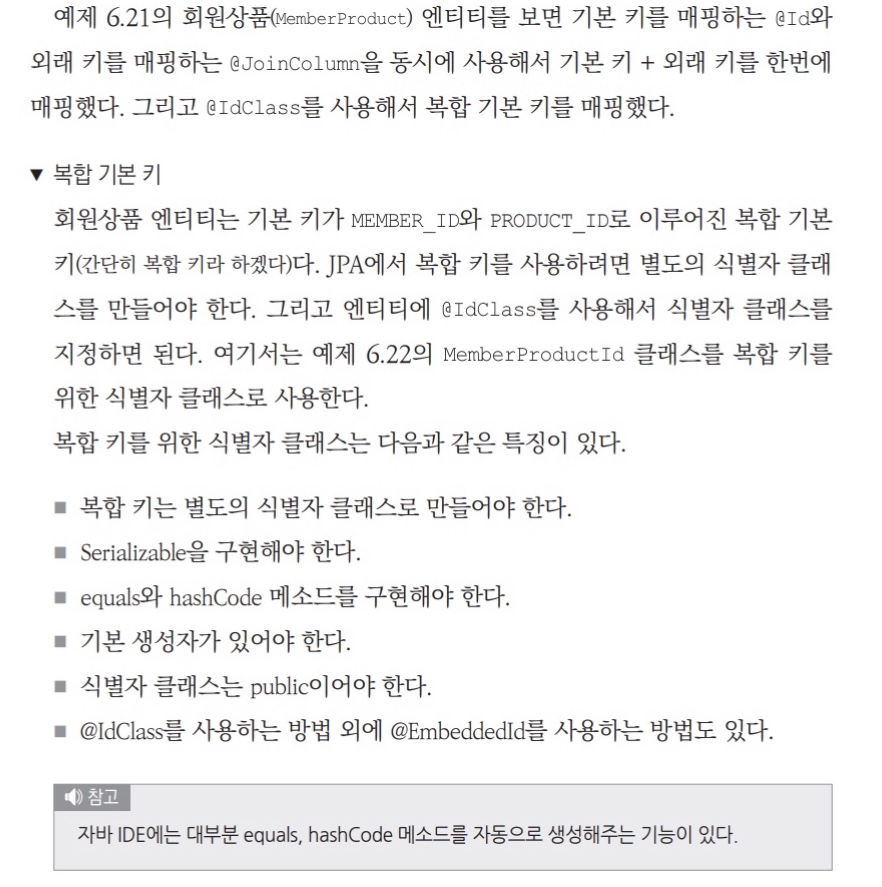
위와같이 주문일자, OrderMount 와 같은 경우에는 양쪽에서의 @ManyToMany를 사용할 수 없다.

왜냐하면 Member\_Product 엔티티에서 추가한 컬럼들을 매핑할 수 없기때문.

* 다대다 ManyToMany 의 형태에서 칼럼이 추가되게 될 경우 방법은 2가지가 있다.

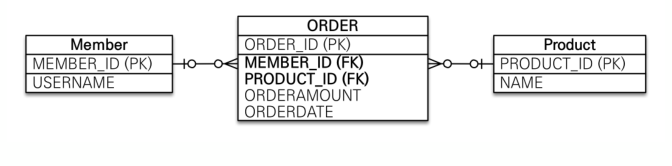
1. 다대다 (N:M)테이블을 구성하는 각각의 객체 이외에 **복합기본키 클래스**를 추가로 생성하여 복합키를 생성 해주는 방법
2. 다대다 (N:M) 에서 새로운 기본키 사용하는 방법.

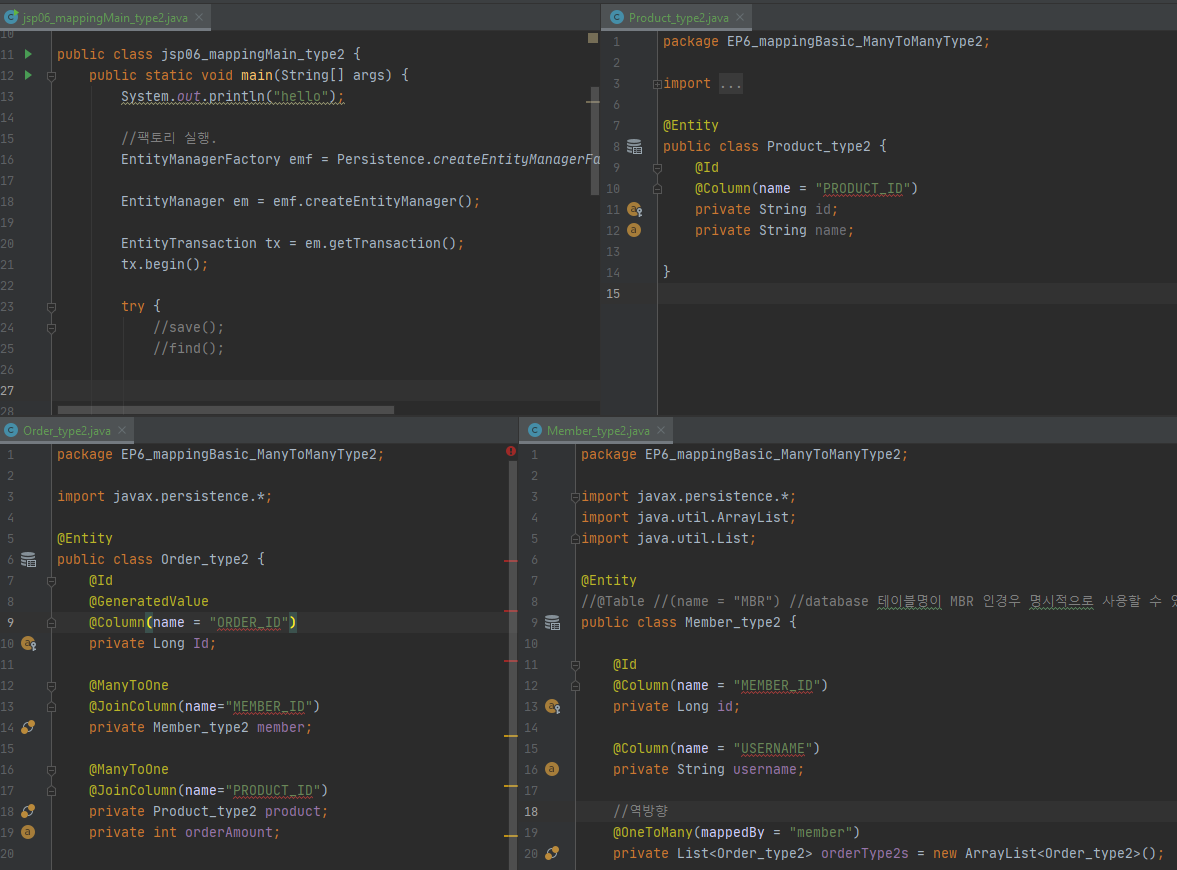
@ManyToMany 를 @ManyToOne과 @OneToMany로 변경한 상태에서 복합기본키 을래스 추가생성.



2번째 방법 : 새로운 기본키 사용

두번째 방법으로 구성할 테이블 구성도는 다음과 같습니다.





1:다 , 다:1 에 해당하는 테이블을 order 테이블에 구성하여 다대다 연관관계를 구성하였다.

Savecode는 책 232페이지 예제6.27에 구성되어 있으며 find코드는 6.28에 구성되어 있다

방법1에 비해서 방법2가 식별자클래스를 생성하지 않아서(방법1의경우 기본키+외래키 테이블 이용) 다대다 관계를 풀어냄에 있어서 더 좋은 방법이라고 생각이 된다.

보기에 편리해 보이지만 실무에서는 사용하지 않는 방식이다.

연결 테이블이 단순히 연결만 하고 끝나지 않는다.

회원 주문을 예로 들자면 주문시간, 수량 같은 별도의 데이터가 들어올 수 있다.