4.엔티티 매핑

JPA를 사용하는데 가장 중요한 것은 엔티티와 테이블을 정확히 매핑하는 것이다

JPA는 다양한 매핑 어노테이션을 지원하는데 크게 4가지로 분류할 수 있다.

JPA 어노테이션

Aa Name	Column
객체와 테이블 매핑	@Entity , @Table
기본 키 매핑	@Id
필드와 컬럼 매핑	@Column
연관관계 매핑	@ManyToOne , JoinColumn

4.1 @Entity

JPA를 사용해서 테이블과 매핑할 클래스는 @Entity 어노테이션 필수로 붙여야 한다.

Entity 속성 저리

Aa 속성	기능	기본값
name	JPA에서 사용할 엔티티 이름을 지정한다. 보통 기본값인 클래스 이름을 사용한다. 이름을 지정해서 충돌하지 않도록 해야한다	설정하지 않으면 클래스 이름을 그대로 사용한다
Untitled		

@Entity 적용 시 주의 사항

1.기본 생성자는 필수다(파라미터가 없는 public 또는 protected 생성자)

2.final 클래스 , enum , intergace , inner 클래스에는 사용할 수 없다.

3.저장할 필드에 final을 사용하면 안된다

4.2 @Table

@Table은 엔티티와 매핑할 테이블을 지정한다.

생략하면 매핑한 엔티티 이름을 테이블 이름으로 사용한다.

@Table 속성절리

Aa 속성	기능	기본값
name	매핑할 테이블 이름	엔티티 이름을 사용한다.
catalog	catalog 기능이 있는 데이터베이스에서 catalog를 매핑한다	
schema	schema 기능이 있는 데이터베이스에서 schema를 매핑한다	
Untitled	DDL 생성 유니크 제약조건을 만든다. 2개 이상의 복합 유니크 제약조건도 만들 수 있다. 참고로 이 기능은 스키마 자동 생성 기능을 사용해서 DDL를 만들 때만 사용한다	

4.4 데이터베이스 스키마 자동 생성

JPA는 데이터베이스 스키마를 자동으로 생성하는 기능일 지원한다.

클래스의 매핑정보를 보면 어떤 테이블에 어떤 컬럼을 사용하는지 알 수 있다.

JPA는 이 매핑 정보와 데이터베이스 방언을 사용해서 데이터베이스 스키마를 생성한다.

스키마 자동 생성 기능을 사용하려면 persistence.xml에 속성을 추가해야한다.

<property name="hibernate.hbm2ddl.auto" value="create"/>

스키마 자동 생성 기능이 만든 DDL은 운영 환경에서 사용할 만큼 완벽하지 않으므로 개발 환경에서 사용하거나 매핑을 어떻게 해야 하는지 참고하는 정도로만 사용하느 것이 좋다

hibernate,.hbm2ddl.auto 속성

Aa 옵션	설명
create	기본 테이블을 삭제하고 생성한다. DROP + CREATE
create-drop	create 속성에 추가로 애플리케이션을 종료할 때 생성한 DDL을 제거한다.
update	데이터베이스 테이블과 엔티티 매핑정보를 비교해서 변경 사항만 수정한다
vaildate	데이터베이스 테이블과 엔티티 매핑정보를 비교해서 차이가 있으면 경고를 남기고 애플리케이션을 싱행하지 않는다 , ddl을 수정하지 않는다.
none	자동 생성 기능을 사용하지 않으려면 auto 속성 자체를 삭제하거나 유효하지 않는 옵션 값을 주면 된다

이름 매핑 전략 변경하기

단어와 단어를 구분할 때 자바 언어는 관례상 roleType 같이 카멜 표기법을 주로 사용하고

데이터베이스는 관례상 role_type과 같이 언더스코어(_)를 주로 사용한다.

hibername.ejb.naming_strategy 속성을 사용하면 매핑 전략을 변경할 수 있다.

하이버네이트는 org.hibername.cfgImprovedNamingStrategy 클래스를 제공한다.

이 클래스는 테이블 명이나 컬럼 명이 생략되면 자바의 카멜 표기법을 테이블의 언더스코어 표기법으로 매핑한다

4.5 DDL 생성 기능

조건 회원 이름은 필수로 입력해야하고 , 10자를 초과하면 안된다는 제약조건이 추가될시

AS_IS
@Column(name = "NAME")
private String username;

TO_BE
@Column(name = "NAME" , nullable = false, length = 10)
private String username;

AS_IS

id bigint not null,
NAME varchar(255),
primary key (id)

TO_BE

id bigint not null,
NAME varchar(10) not null,
primary key (id)

@Column의 속성 length , nullable 을 포함에서 이런 기능들은 단지 DDL 자동 생성할 때만 사용되고 JPA실행 로직에는 영향을 주지 않음

4.5 기본 키 매핑

▼ 직접 할당 : 기본 키를 애플리케이션에서 직접 할당한다.

▼ 자동 생성 : 대리 키 사용 방식

- IDENTITY : 기본 키 생성을 데이터베이스에 위임한다
- SEQUENCE : 데이터베이스 시퀀스를 사용해서 기본 키를 할당한다.
- TABLE : 키 생성 테이블을 사용한다.

자동생성이 전략이 다양한 이유는 지원하는 방식이 다르기 때문이다.

키 생략 전략을 사용하려면 persistence.xml

hibername_id.new_generator_mappings=ture 속성을 반드시 추가 해야한다.

-기본 키 직접 할당 전략

기본 키 직접 할당 전략은 em.persist()로 엔티티를 저장하기 전에 애플리케이션에서 기본 키를 직접 할당하는 방법이다.

-자동생성 IDENTITY 전략

IDENTITY 전략은 데이터베이스에 값을 저장하고 나서야 기본 키 값을 구할 수 있을 때 사용한다.

IDENTITY 전략을 사용하려면 @GeneratedValue의 strategy 속성 값을 GenerationType.IDENTITY 로 지정하면 된다. 이 전략을 사용하면 JPA는 기본 키 값을 얻어오기 위해 데이터베이스를 추가로 조회한다.

이 전략은 엔티티를 데이터베이스에 저장한 후에 식별자를 조회해서 인티티의 식별자에 할당한다.

@GeneratedValue(strategy = GenerationType.IDENTITY)

-자동생성 SEQUENCE 전략

데이터베이스 시퀀스는 유일한 값을 순서대로 생성하는 특별한 데이터베이스 오브젝트다

시퀀스 전략은 이 시퀀스를 사용해서 기본 키를 생성한다.

이 전략은 시퀀스를 지원하는 오라클,포스트그레 H2등 사용할 수 있다

이 전략은 em.persist()를 호출할 때 먼저 데이터베이스 시퀀스를 사용해서 식별자를 조회한다. 그리고 조회한 식별자를 엔티티에 할당한 후에 엔티티를 영속성 컨텍스트에 저장한다.

@GeneratedValue(strategy = GenerationType.SEQUENCE , generator = "BOARD_SEQ_GENERATOR")

@SequenceGenerator 속성 정리

속성	기능	기본값
name	식별자 생성기 이름	필수
sequceName	데이터베이스에 등록되어 있는 시퀀스 이름	hivernate_sequnce
initalVlue	DDL 생성 시에만 사용됨, 시퀀스 DDL을 생성할 때 처음 시작하는 수를 지정한다	1
allocationSize	시퀀스 한 번 호출에 증가하는 수 (성능 최적화에 사용됨)	50
catalog.schema	데이터베이스 catalog,schema 이름	

Sequnce 전략과 최적화

아래 링크 참조

https://dololak.tistory.com/479

-자동생성 Table 전략

이 전략은 키 생성 전용 테이블을 하나 만들고 이름과 값으로 사용할 컬럼을 만들어 데이터베이스 시퀀스를 흉내내는 전략이다. 이 전략은 테이블을 사용하므로 모든 데이터베이스에 적용할 수 있다.

Table전략은 시퀀스 대신에 테이블을 사용한다는 것만 제외하면 시퀀스 전략과 내부 동작방식이 같다.

@GeneratedValue(strategy = GenerationType.TABLE)

TableGenerator 속성 정리

Name	Column	Column 1
name	식별자 생성기 이름	필수
table	키 생성 테이블 명	hibernate_sequences
pkColumnName	시퀀스 컬럼명	sequence_name
valueColumnNa	시퀀스 값 컬럼명	next_val

Name	Column	Column 1
pkColumnValue	키로 사용할 값 이름	엔티티 이름
initialValue	초기 값, 마지막으로 생성된 값이 기준	0
allocationSize	시퀀스 한 번 호출에 증가하는 수 (성능 최적화에 사용)	50
catalog, schema	데이터베이스 catalog, schema 이름	
uniqueConstraints(DDL)	유니크 제약 조건을 지정할 수 있다.	

-Auto 전략

기본 설정 값
방언에 따라 위의 세 가지 전략을 자동으로 지정한다.

@GeneratedValue(strategy = GenerationType.AUTO)

기본 키 매핑 정리

- 직접 할당 : em.persist()를 호출하기 전에 애플리케이션에서 직접 식별자 값을 할당해야 한다.만약 식별자값이 없으면 예외가 발생한다
- SEQUNCE : 데이터베이스 시퀀스에서 식별자 값을 획득한 후 영속성 컨텍스트에 저장한다.
- TABLE : 데이터베이스 시퀀스 생성용 테이블에서 식별자 값을 획득한 후 영속성 컨텍스트에 저장한다.
- IDENTITY : 데이터베이스에 엔티티를 저장해서 식별자 값을 획득한 후 영속성 컨텍스트에 저장한다.

권장하는 식별자 선택전략

데이터베이스 기본 키는 다음 3가지 조건을 모두 만족해야한다

- NULL값은 허용하지 않는다
- 유일해야 한다
- 변해선 안된다

테이블의 기본 키를 선택하는 전략은 크게 2가지가 있다.

자연 키 (natural key)

- 비즈니스에 의미가 있는 키(주민등록번호, 이메일, 전화번호)

대리키 (surrogate key)

- 비즈니스와 관련 없는 임의로 만들어진 키 . 대체 키로도 불린다