😇

자바 ORM 표준 JPA 프로그래밍

섹션 0

JPA : Java Persistence API

객체를 DB에 저장하려면 JDBC API와 SQL을 적었어야 했음

→ JdbcTemplate… → JPA의 등장, SQL 작성할 필요가 없음!

섹션 1

객체 지향 프로그래밍은 추상화, 캡슐화, 정보은닉, 상속, 다형성 등 시스템의 복잡성을 제어할 수 있는 다양한 장치들을 제공한다.

객체를 영구 보관하는 다양한 저장소: RDB, NoSQL, File, OODB

객체 → SQL 변환 → RDB

객체 vs 관계형 데이터베이스

상속, 연관관계(참조 vs 외래키), 데이터 타입, 데이터 식별 방법

객체를 자바 컬렉션에 저장 하듯이 DB에 저장할 수는 없을까?

JPA? Java Persistence API (자바 진영의 ORM 기술 표준)

ORM? Object-relational mapping (객체 관계 매핑)

객체는 객체대로, 관계형 DB는 관계형 DB대로 설계하고 ORM 프레임워크가 중간에서 매핑

JPA는 애플리케이션과 JDBC 사이에서 동작

인터페이스의 모음

JPA를 왜 사용해야 하는가?

SQL 중심적인 개발에서 객체 중심으로 개발해야

생산성, 유지보수, 패러다임의 불일치 해결, 성능, 데이터 접근 추상화, 표준

JPA와 패러다임의 불일치 해결

상속, 연관관계, 객체 그래프 탐색, 비교하기

섹션 2

데이터베이스 방언

- JPA는 특정 DB에 종속 x
- 각각의 데이터베이스가 제공하는 SQL 문법과 함수는 다름
    - 가변문자, 문자열을 자르는 함수, 페이징
- 방언: SQL 표준을 지키지 않는 특정 데이터베이스만의 고유한 기능
- hibernate.dialect 속성에 지정
- 하이버네이트는 40가지 이상의 데이터베이스 방언 지원

JPA 구동방식

설정 정보 조회 → 생성 → 생성

엔티티 매니저 팩토리는 하나만 생성해서 애플리케이션 전체에서 공유

엔티티 매니저는 쓰레드 간에 공유하면 안됨 → 사용하고 버려야 함

JPA의 모든 데이터 변경은 트랜잭션 안에서 실행

→ JPQL

섹션 3

JPA에서 가장 중요한

→ 객체와 관계형 데이터베이스 맵핑

→ 영속성 컨텍스트

영속성 컨텍스트란?

JPA를 이해하는데 가장 중요한 용어

엔티티를 영구 저장하는 환경

EntityManager.persist(entity)

논리적인 개념으로 눈에 보이지 않으며 엔티티 매니저로 접근 가능

엔티티의 생명주기

비영속, 영속, 준영속, 삭제

영속성 컨텍스트의 이점

1차 캐시, 동일성 보장, 트랜잭션을 지원하는 쓰기 지연, 변경 감지, 지연 로딩

영속 엔티티의 동일성 보장

1차 캐시로 반복 가능한 읽기 등급의 트랜잭션 격리 수준을 데이터베이스가 아닌 애플리케이션 차원에서 제공!

플러시

영속성 컨텍스트의 변경 내용을 데이터베이스에 반영

플러시발생

변경 감지

수정된 엔티티 쓰기 지연 SQL 저장소에 등록

쓰기 지연 SQL 저장소의 쿼리를 데이터베이스에 전송

플러시는,

영속성 컨텍스트를 비우지 않음

영속성 컨텍스트의 변경 내용을 데이터베이스에 동기화

트랜잭션이라는 작업 단위가 중요 → 커밋 직전에만 동기화 하면 됨!

영속 → 준영속

영속 상태의 엔티티가 영속성 컨텍스트에서 detached

how to 준영속

em.detach(entity), em.clear(), em.close()