2020-12-14

SQL 중심적인 개발의 문제점

객체지향언어 관계형 DB -> SQL 중심적인 개발 (무한 반복 ,지루한 코드 CRUD)

필드 추가나 수정시 모든 SQL문을 수정을 해야한다.

SQL에 의존적인 개발을 피하기 어렵다 .

패러다임의 불일치

**\*객체를 자바 컬렉션에 저장 하듯이 DB에 저장할 수는 없을까?**

JPA –Java Persistence API

ORM?

Object-relational mapping(객체 관계 매핑)

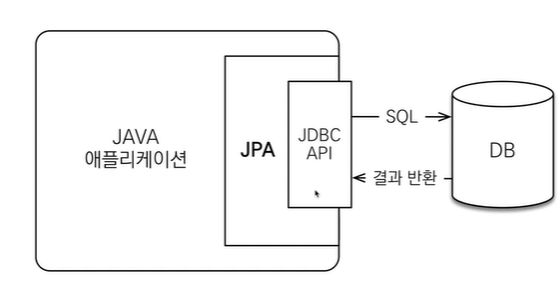
객체는 객체대로 설계

관계형 데이터베이스는 관계형 데이터 베이스대로 설계

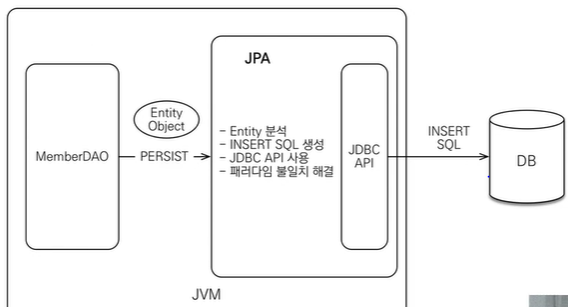
ORM 프레임워크가 중간에서 매핑

대중적인 언어에는 대부분 ORM 기술이 존재

JPA는 애플리케이션과 JDBC 사이에서 동작



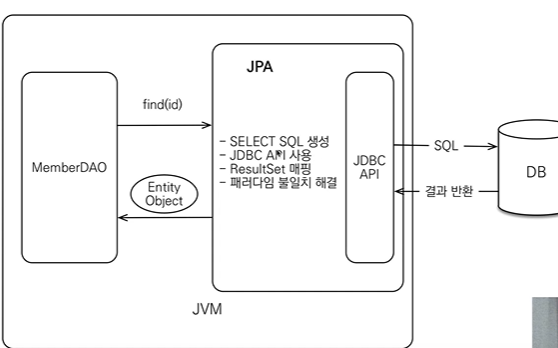
데이터 생성



\*패러다임 불일치 해결이 중요하다 .!

SQL을 JPA가 만들어주는것이다 .

데이터조회

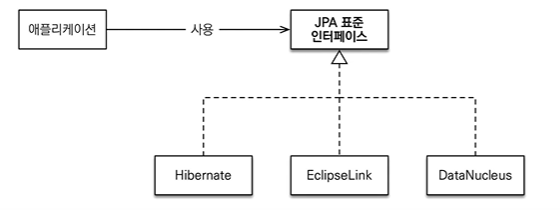


JPA 소개

EJB – 엔티티빈 (자바 표준) -> 하이버네이트(오픈 소스) -> JPA(자바 표준)

성능도 좋지않다 등등 -> 오픈소스를 통해서 많이 발전함 -> 하이버네이트정리해서 만듬

JPA는 인터페이스의 모음



JPA를 왜 사용해야 하는가 ?

SQL 중심적인 개발에서 객체 중심으로 개발-> 객체 상속관계 ,

생산성, 유지보수 -> CRUD가 만들어져있다 .,필드 추가시 모든 SQL이 알아서 수정된다.

패러다임의 불일치 해결

성능

데이터 접근 추상화와 벤더 독립성 //자유로운 객체 그래프탐색 가능

표준

JPA의 성능 최적화 기능

1차캐시와 동일설 보장 -> 약간의 조회 성능 향상

트랜잭션을 지원하는 쓰기 지연 em.persist // // transaction .commit

지연 로딩 : 객체가 실제 사용될 때 로딩 != ( 즉시 로딩 JOIN SQL로 한번에 연관된 객체까지 미리조회)

ORM은 객체와 RDB 두 기둥위에 있는 기술

2020-12-17

데이터 베이스 방언

JPA는 특정 데이터베이스에 종속 되어있지 않는다.

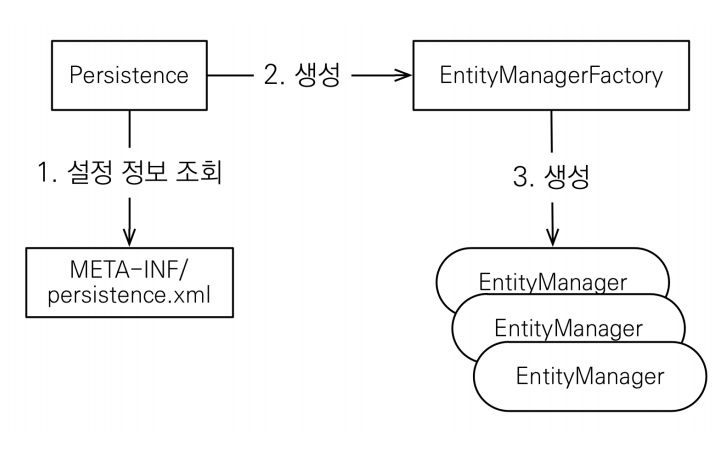
각각의 데이터베이스가 제공하는 sql 문법과 함수는 조금씩 다름

가변문자 :MySQL은 VARCHAR ,Oracle 은 VARCHAR2

문자열을 자르는 함수 :SQL 표준은 SUBSTRING(), Oracle은 SUBSTR

페이징 : MySQL은 LIMIT oracle은 ROWNUM

JPA 구동방식



엔티티 매니저 팩토리는 하나만 생성해서 애플리 케이션 전체에서 공유

엔티티 매니저는 쓰레드간에 공유 X (사용하고 버려야 한다.)

JPA의 모든 데이터 변경은 트랜잭션 안에서 실행

JPQL

JPA를 사용하면 에티티 객체를 중심으로 개발

문제는 검색 쿼리

검색을 할 때도 테이블이 아니 엔티티 객체를 대상으로 검색

모든 DB데이터를 객체로 변환해서 검색하는 것은 불가능

애플리케이션이 필요한 데이터만 DB에서 불러오려면 결국 검색 조건이 포함된 SQL 이 필요

Jpa는 SQL을 추상화한 JPQL이라는 객체 지향 쿼리 언어 제공

SQL과 문법 유사, SELECT,FROM,WHERE,GROUP BY ,HAVING ,JOIN 지원

차이점

JPQL은 엔티티 객체를 대상으로 쿼리

SQL은 데이터 베이스 테이블을 대상으로 쿼리