JPA

> Java Persistence API

```
개발자가 직접 SQL을 작성하지 않고, JPA API를 활용해 DB를 저장하고 관리할 수 있다.
```

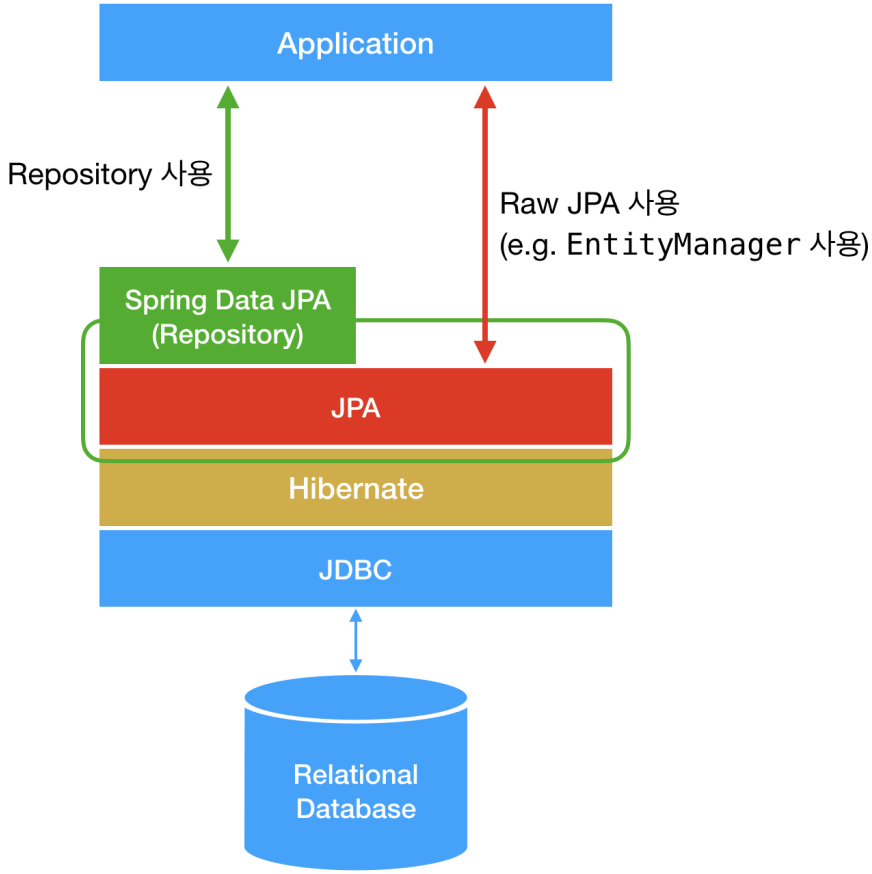
JPA는 오늘날 스프링에서 많이 활용되고 있지만, 스프링이 제공하는 API가 아닌 **자바가 제공하는 API다.**

자바 ORM 기술에 대한 표준 명세로, 자바 어플리케이션에서 관계형 데이터베이스를 사용하는 방식을 정의한 인터페이스다.

ORM(Object Relational Mapping)

ORM 프레임워크는 자바 객체와 관계형 DB를 매핑한다. 즉, 객체가 DB 테이블이 되도록 만들어주는 것이다. ORM을 사용하면, SQL을 작성하지 않아도 직관적인 메소드로 데이터를 조작할 수 있다는 장점이 있다. (개발자에게 생산성을 향상시켜줄 수 있음)

종류로는 Hibernate, EclipseLink, DataNucleus 등이 있다.



스프링 부트에서는 `spring-boot-starter-data-jpa`로 패키지를 가져와 사용하며, 이는 Hibernate 프레임워크를 활용한다.

JPA는 애플리케이션과 JDBC 사이에서 동작하며, 개발자가 JPA를 활용했을 때 JDBC API를 통해 SQL을 호출하여 데이터베이스와 호출하는 전개가 이루어진다.

즉, 개발자는 JPA의 활용법만 익히면 DB 쿼리 구현없이 데이터베이스를 관리할 수 있다.

JPA 특징

1. **객체 중심 개발 가능**

   SQL 중심 개발이 이루어진다면, CRUD 작업이 반복해서 이루어져야한다.

   하나의 테이블을 생성해야할 때 이에 해당하는 CRUD를 전부 만들어야 하며, 추후에 컬럼이 생성되면 관련 SQL을 모두 수정해야 하는 번거로움이 있다. 또한 개발 과정에서 실수할 가능성도 높아진다.

2. **생산성 증가**

   SQL 쿼리를 직접 생성하지 않고, 만들어진 객체에 JPA 메소드를 활용해 데이터베이스를 다루기 때문에 개발자에게 매우 편리성을 제공해준다.

3. **유지보수 용이**

   쿼리 수정이 필요할 때, 이를 담아야 할 DTO 필드도 모두 변경해야 하는 작업이 필요하지만 JPA에서는 엔티티 클래스 정보만 변경하면 되므로 유지보수에 용이하다.

4. **성능 증가**

   사람이 직접 SQL을 짜는 것과 비교해서 JPA는 동일한 쿼리에 대한 캐시 기능을 지원해주기 때문에 비교적 높은 성능 효율을 경험할 수 있다.

제약사항

JPA는 복잡한 쿼리보다는 실시간 쿼리에 최적화되어있다. 예를 들어 통계 처리와 같은 복잡한 작업이 필요한 경우에는 기존의 Mybatis와 같은 Mapper 방식이 더 효율적일 수 있다.

> Spring에서는 JPA와 Mybatis를 같이 사용할 수 있기 때문에, 상황에 맞는 방식을 택하여 개발하면 된다.

[참고 사항]

- 링크
- 링크