**06. 데이터베이스 연동**

**1. ORM (Object Relational Mapping)**

**①** ORM이란?

: 객체 지향 언어에서 의미하는 객체와 RDB(Relational Database)의 테이블을 자동으로 매핑하는 방법

∵ 클래스가 테이블과 매핑하기 위해 만들어진 게 아니기 때문에 불일치 발생

② 장점

• DB 쿼리를 객체지향적으로 조작 가능 -> 비용 감소, 코드 가독성 증가

• 재사용 및 유지보수 편리

• DB에 대한 종속성 감소

③ 단점

• ORM 만으로는 온전한 서비스 제공 불가 -> 복잡한 서비스의 경우 쿼리 작성 필요

• 객체 관점과 DB의 관계 관점 사이 불일치 발생

-> 세분성: DB의 테이블 수와 어플의 Entity 클래스 수가 다를 경우

-> 상속성: RDBMS에는 상속 x

-> 식별성: RDBMS는 기존 키로 동일성 정의, 자바는 두 값이 같아도 다르게 판단 가능

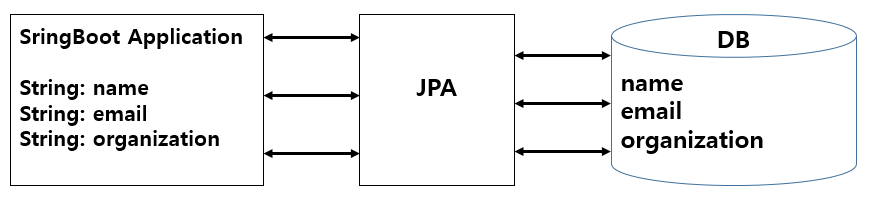
-> 연관성: 자바는 객체 참조 이용, RDBMS는 외래 키로 연관성 표현

-> 탐색: 자바는 객체 참조를 활용해 연결, RDBMS는 조인을 통해 로드하여 조회

**2. JPA (Java Persistence API)**

**①** JPA란?

: 자바 진영의 ORM 기술 표준으로 채택된 인터페이스의 모음



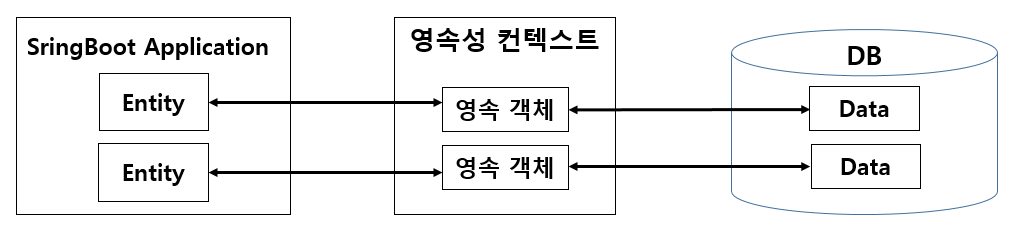
Ex) 하이버네이트, 이클립스 링크, 데이터 뉴클리어스

② 하이버네이트

• spring data JPA: JPA를 편리하게 사용하기 위해 지원하는 스프링 하위 프로젝트

• CRUD 처리

**2. 영속성 컨텍스트**

****

**①** 정의

: 어플과 DB 사이에서 entity와 레코드의 괴리를 해소하는 기능과 개체를 보관하는 기능 수행

**•**영속 객체: entity 객체가 영속성 컨텍스트에 들어와 매핑 정보를 DB에 반영하고 JPA의 관리 대상이 되는 시점의 해당 객체

**②** Entity 매니저

: entity를 관리하는 객체로 DB에 접근해서 CRUD 작업 수행

**3. DAO 설계**

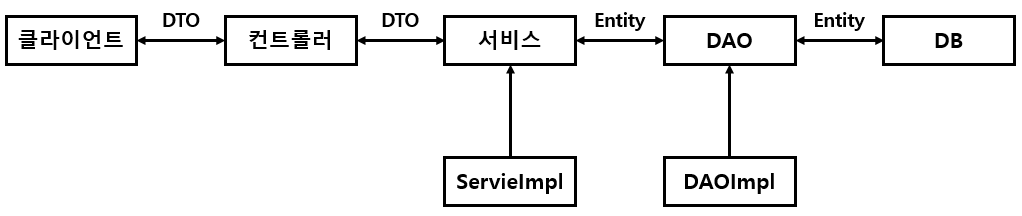
**①** 정의

: Data Access Object의 약어로, DB에 접근하기 위한 로직을 관리하는 객체

• 비즈니스 레이어에 존재

• 리포지토리의 비슷하지만 다름 -> 자세한 건 기본지식 폴더에서…

**4. 구조**

****

**①** 클라이언트: 애플리케이션의 이용자.

**②** 컨트롤러: 클라이언트의 요청과 비즈니스 로직을 연결.

**③** 서비스: 애플리케이션에서 수행하는 핵심 기능을 제공

**④** DAO: DB와 서비스 사이를 연결.

**⑤** DB: 영속성 데이터를 저장.