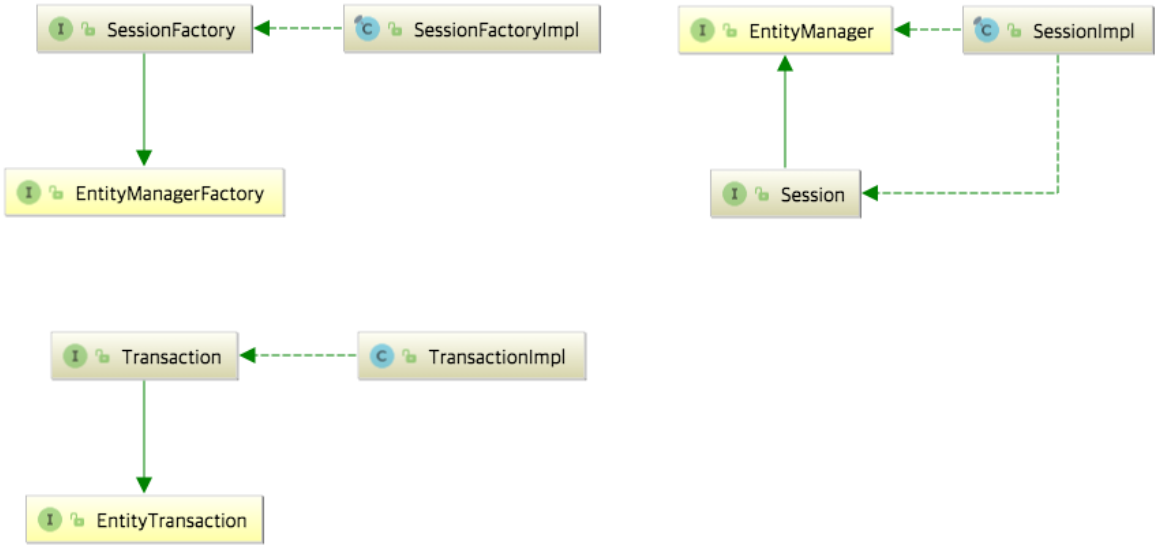
JPA, Hibernate, SpringDataJPA(Repository)

1. JPA는 기술 명세

- JPA는 Java Persistence API의 약자
- **자바 어플리케이션에서 관계형 데이터베이스를 사용하는 방식을 정의한 인터페이스**
- 특정 기능을 하는 **라이브러리가 아니라 인터페이스**
- JPA는 단순히 명세이기 때문에 구현이 없다. 예를 들어, JPA의 핵심이 되는 `EntityManager`는 아래와 같이 `javax.persistence.EntityManager` 라는 파일에 `interface`로 정의되어 있다.

2. Hibernate는 JPA의 구현체

- **JPA라는 명세의 구현체**
- `javax.persistence.EntityManager`와 같은 인터페이스를 직접 구현한 라이브러리
- **JPA와 Hibernate는 마치 자바의 interface와 해당 interface를 구현한 class와 같은 관계**이다.



위 사진은 JPA와 Hibernate의 상속 및 구현 관계를 나타낸 것이다. JPA의 핵심인 `EntityManagerFactory`, `EntityManager`, `EntityTransaction`을 Hibernate에서는 각

각 `SessionFactory`, `Session`, `Transaction`으로 상속받고 각각 `Impl`로 구현하고 있음을 확인할 수 있다.

- **JPA를 사용하기 위해서 반드시 Hibernate를 사용할 필요가 없다.**
- DataNucleus, EclipseLink 등 다른 JPA 구현체를 사용해도 되고, 심지어 본인이 직접 JPA를 구현해서 사용할 수도 있다.

3. Spring Data JPA는 JPA를 쓰기 편하게 만들어놓은 모듈이다

- Spring Data JPA는 Spring에서 제공하는 모듈 중 하나
- **JPA를 한 단계 추상화시킨 `Repository`라는 인터페이스를 제공함으로써 이루어진다**.
- 사용자가 `Repository` 인터페이스에 정해진 규칙대로 메소드를 입력하면, Spring이 알아서 해당 메소드 이름에 적합한 쿼리를 날리는 구현체를 만들어서 Bean으로 등록해준다.
- **Spring Data JPA의 `Repository`의 구현에서 JPA를 사용하고 있다**. 예를 들어, `Repository` 인터페이스의 기본 구현체인 `SimpleJpaRepository`에 내부적으로 `EntityManager`을 사용하고 있는 것을 볼 수 있다.

4. 요약