orphanRemoval = true vs DDL의 On Delete Cascade vs CascadeType.REMOVE

먼저 예문을 보자.

```
@Entity
class Emp {
    @OneToOne(orphanRemoval=true)
     private Addr addr;
}
```

Emp 엔티티가 삭제될 때 참조가 끊어진 연관된 Addr 엔티티도 삭제하라는 의미이며 DB에서도 삭제되는데, 참조(연결)가 끊어진 Addr 객체는 DB에서도 삭제된다는 뜻이다.

```
@Entity
class Emp {
    @OneToOne(cascade=CascadeType.REMOVE)
     private Addr addr;
}
```

Emp 엔티티가 삭제될 때 연관된 Addr 엔티티도 삭제하라는 의미이며 DB에서도 삭제된다.

```
@Entity
class Emp {
    @OneToMany(orphanRemoval=true)
     private List<Addr> addr;
}
```

addr 컬렉션에서 Addr 객체가 제거되는 경우 DB에서도 삭제하라는 의미.

orphanRemoval은 JPA2.0 이상에서 지원하는 것으로 ORM 스펙, JPA 레벨에서의 정의이고 On Delete Cascade는 DBMS 레벨에서 작동되며 하는 일은 같다. orphanRemoval은 @OneToMany 연관에서 부모 엔티티의 컬렉션 등에서 자식 엔티티가 삭제될 때 참조가 끊어지므로 DB 레벨에서도 삭제되고, @OneToOne 연관에서는 엔티티가 삭제될 때 연관된 엔티티가 참조가 끊어지므로 DB에서 삭제된다. 즉 참조, 연결이 끊어진(Disconnected 된) 엔티티를 같이 삭제하라는 의미로 Owner 객체와 참조가 끊어진 객체들을 정리할 때 유용하다.

cascade=CascadeType.REMOVE는 연결이 끊어진다고 해서 자동 삭제되는 것은 아니고 명시적으로 연관 엔티티가 삭제될 때 같이 삭제하라는 영속성 전이와 관련된 옵션이다.

반면 **On Delete Cascade**는 DB레벨에서 부모 테이블의 레코드가 삭제될 때 자식레코드

도 같이 삭제하라는 의미이다.