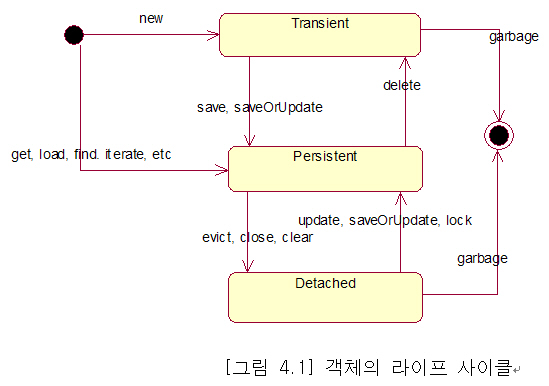
**[Database ID 기반 Equals, HashCode 구현](http://blog.paran.com/dce/4886814" \t "_dummy)**

Hibernate 객체의 lifecycle은 아래 그림과 같다.



|  |  |
| --- | --- |
| 상태 | 의미 |
| transient | 메모리에서 생성된 객체가 아직 데이터베이스의 특정 ROW로 관계를 갖기 전의 상태 |
| persistent | 데이터베이스의 특정 ROW와 관계를 갖은 상태 |
| detached | Hibernate의 세션이 종료되어 persistent 상태의 객체가 Persistence Manager와 관계를 잃게 된 상태 |

어플리케이션에서 하나의 유즈케이스에서 데이터베이스에서 읽어 들인 객체를 프리젠테이션 계층으로 전송하여 사용자에게 보여주고 변경하도록 하고 다른 유즈케이스에서 이 객체를 받아서 데이터베이스에 변경을 반영하고자 할 때가 있다.

이때 성능상의 이유로 객체 그래프 전체를 두번째 유즈케이스에서 reattach하지 않으려 할 것이다. Hibernate는 **selective reaassociation of detached instance**를 지원한다.

이 말은 어플리케이션이 detached된 객체 그래프의 서브 그래프를 효율적으로 두번째 유즈케이스의 Persistence Manager(Hibernate Session)에 reattach한다는 것을 의미하다.

Reattachment는 데이터베이스에 새로운 ROW를 생성할 수도 있다. 예를 들어 프리젠테이션 계층으로 전송된 detached Item에 새로운 Bid가 추가된 경우를 생각할 수 있다. Hibernate는 이 Bid가 새로이 생성되었다는 것을 감지하고 데이터베이스에 insert 해야 한다는 것을 감지해 낸다. 이러한 작업을 위해 Hibernate는 새로운 transient 인스턴스와 기존의 detached 인스턴스를 구별해야만 한다.

detached 상태의 인스턴스를 persistent 상태로 전이 시키는 주요한 개념이 Equality이다. 자바에서 Equality는 equals()와 hashCode()에 의해 구현된다.

equals, hasCode를 구현하는 방법에는 다음과 같은 방법이 있다.

- 데이터베이스 ID Equality를 이용하는 방법

- 객체의 모든 속성을 비교하는 방법

- 객체의 비즈니스 키 Equality를 이용하는 방법

이 방법 중 객체의 비즈니스 키 Equality를 이용하는 방법 만 모든 경우에 안전하게 동작한다.

User 객체의 userName이 비즈니스 키일 경우 아래와 같이 구현할 수 있다.

    public boolean equals(Object other) {  
        if (this == other)  
            return true;  
        if (getId() == null)  
            return false;  
        if (!(other instanceof User))  
            return false;  
        final User that = (User) other;  
        return this.getUserName().equals(that.getUserName());  
    }  
      
    public int hashCode() {  
        return getUserName().hashCode();  
    }

**. Cascading Persistence with Hibernate**

cascade 속성으로 지정할 수 있는 값과 의미는 아래 표와 같다.

|  |  |
| --- | --- |
| cascade="none" | default  hibernate가 association을 무시하도록 한다. |
| cascade="save-update" | save(), update() 메소드에 대해 association을 따라 navigate하면서 save, update가 발생하도록 한다. |
| cascade="delete" | delete() 메소드가 호출되었을 때 association을 따라 navigate하면서 delete가 발생하도록 한다. |
| cascade="all" | save(), update(), delete(), evict(), lock() 메소드가 호출되었을 때 association을 따라  navigate하면서 해당 오퍼레이션이 발생하도록 한다. |
| cascade="all-delete-orphan" | cascade="all"과 동일.  추가적으로 association에서 제거된 객체가 delete되도록 한다. |
| cascade="delete-orphan" | association에서 제거된 객체가 delete되도록 한다. |

**. Hibernate가 인스턴스를 transient로 간주하는 경우**

- identifier property가 null인 경우

- version property가 null인 경우

- 맵핑에 정의된 unsaved-value와 identifier property의 값이 동일한 경우

- 맵핑에 정의된 unsaved-value와 version property의 값이 동일한 경우

- Interceptor를 정의하고 Interceptor.isUnsaved() 메소드에서 Boolean.TRUE를 반환하는 경우