JPA

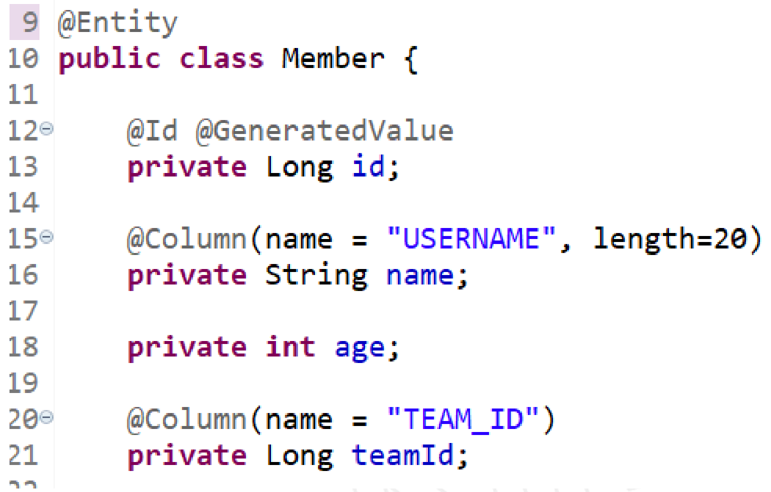
Kim Hye Kyung

topickim@naver.com

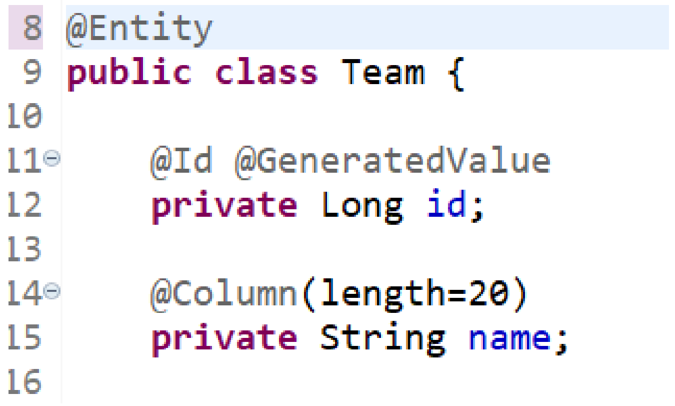
실습

step02_mapping 실습 단계

step01

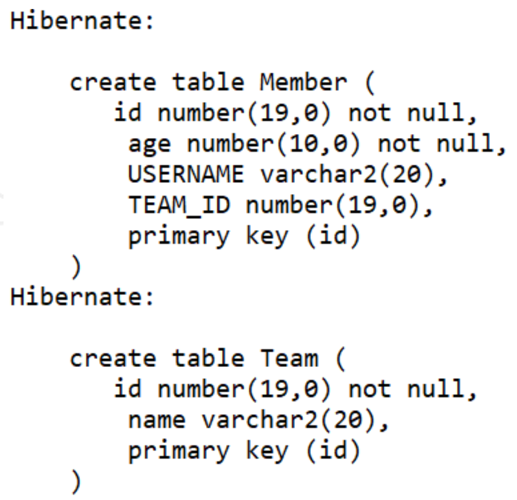


```java
@Entity
public class Member {

    @Id @GeneratedValue
    private Long id;

    @Column(name = "USERNAME", length=20)
    private String name;

    private int age;

    @Column(name = "TEAM_ID")
    private Long teamId;
```



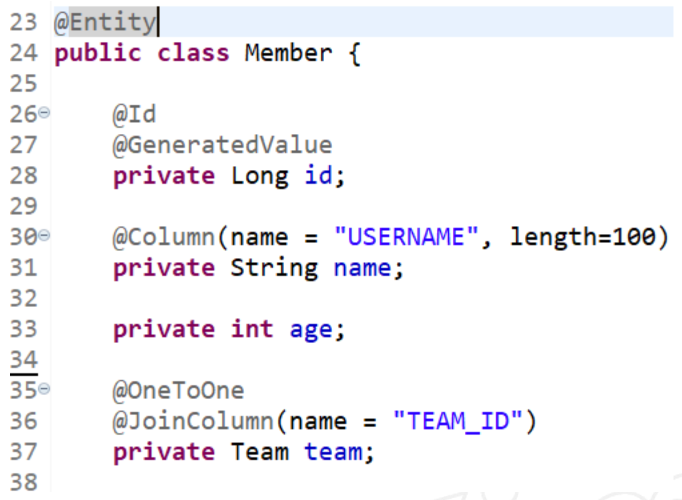
```java
@Entity
public class Team {

    @Id @GeneratedValue
    private Long id;

    @Column(length=20)
    private String name;

```

실행시 table 생성 sql

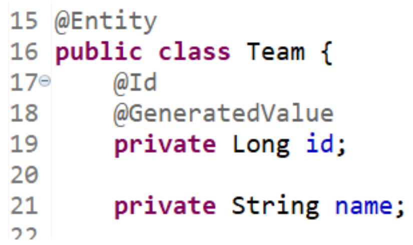


```
Hibernate: 

    create table Member (
       id number(19,0) not null,
        age number(10,0) not null,
        USERNAME varchar2(20),
        TEAM_ID number(19,0),
        primary key (id)
    )
Hibernate: 

    create table Team (
       id number(19,0) not null,
        name varchar2(20),
        primary key (id)
    )
```

step02

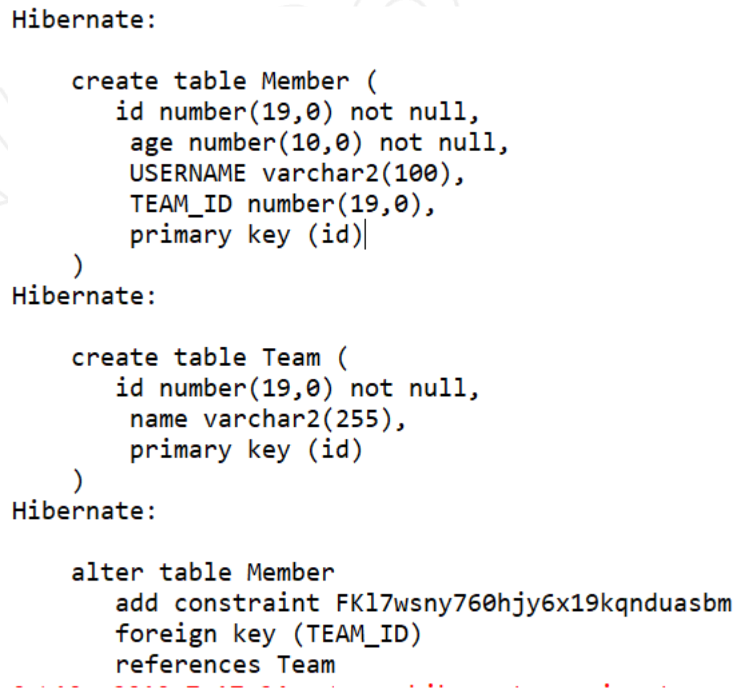


```java
@Entity
public class Member {

    @Id
    @GeneratedValue
    private Long id;

    @Column(name = "USERNAME", length=100)
    private String name;

    private int age;

    @OneToOne
    @JoinColumn(name = "TEAM_ID")
    private Team team;

```



```java
@Entity
public class Team {
    @Id
    @GeneratedValue
    private Long id;

    private String name;
```

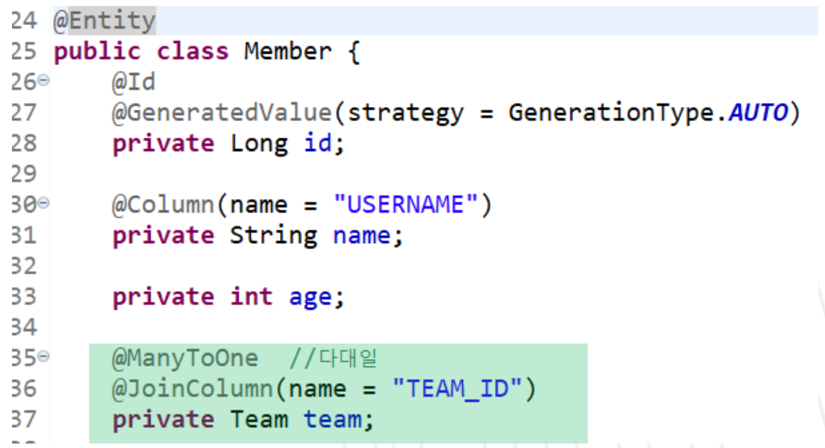
실행시 table 생성 sql



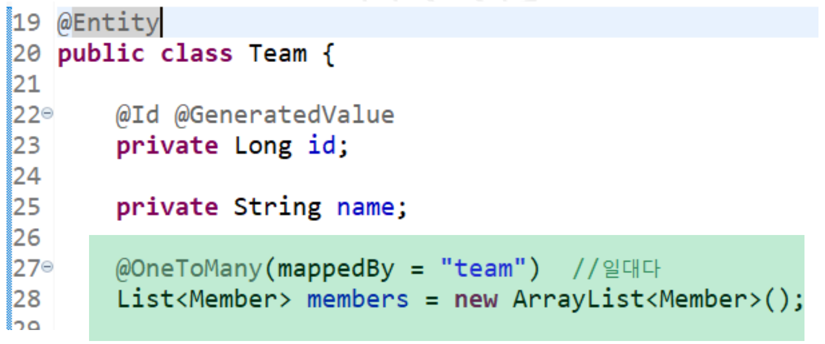
```
Hibernate: 

    create table Member (
       id number(19,0) not null,
        age number(10,0) not null,
        USERNAME varchar2(100),
        TEAM_ID number(19,0),
        primary key (id)
    )
Hibernate: 

    create table Team (
       id number(19,0) not null,
        name varchar2(255),
        primary key (id)
    )
Hibernate: 

    alter table Member 
       add constraint FKl7wsny760hjy6x19kqnduasbm 
       foreign key (TEAM_ID) 
       references Team
```

step03

- 한 team에 다수의 member 보유

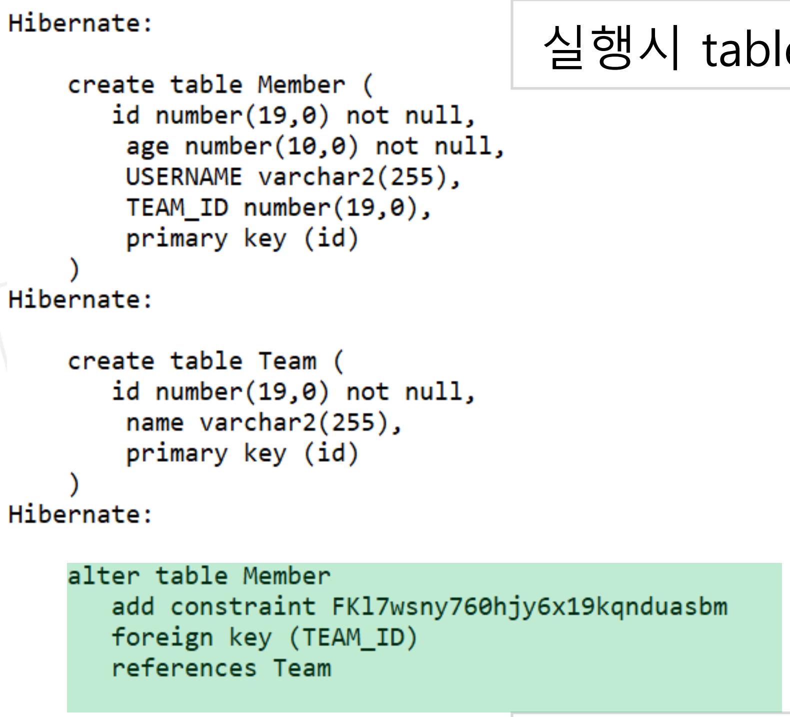


```java
@Entity
public class Member {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;

    @Column(name = "USERNAME")
    private String name;

    private int age;

    @ManyToOne  //다대일
    @JoinColumn(name = "TEAM_ID")
    private Team team;
```



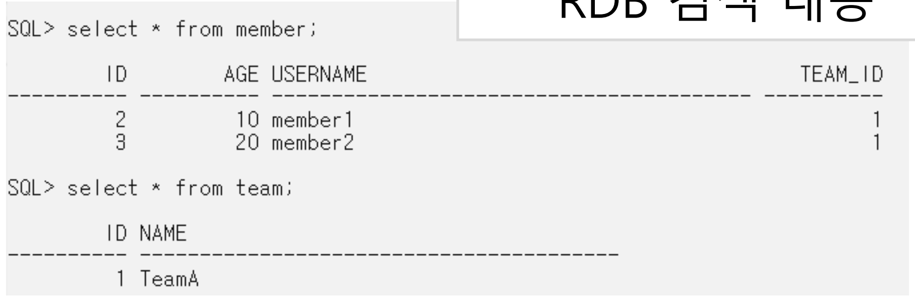
```java
@Entity
public class Team {

    @Id @GeneratedValue
    private Long id;

    private String name;

    @OneToMany(mappedBy = "team")  //일대다
    List<Member> members = new ArrayList<Member>();
```

실행시 table 생성 sql



```
Hibernate: 

    create table Member (
       id number(19,0) not null,
        age number(10,0) not null,
        USERNAME varchar2(255),
        TEAM_ID number(19,0),
        primary key (id)
    )
Hibernate: 

    create table Team (
       id number(19,0) not null,
        name varchar2(255),
        primary key (id)
    )
Hibernate: 

    alter table Member
       add constraint FK17wsny760hjy6x19kqnduasbm
       foreign key (TEAM_ID)
       references Team
```

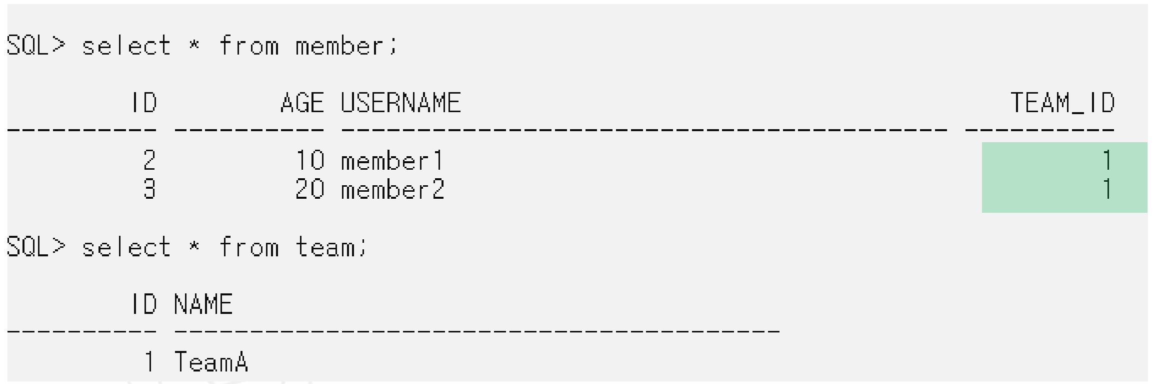
RDB 검색 내용



```
SQL> select * from member;

        ID        AGE USERNAME                                    TEAM_ID
---------- ---------- ---------------------------------------- ----------
         2         10 member1                                           1
         3         20 member2                                           1

SQL> select * from team;

        ID NAME
---------- ----------------------------------------
         1 TeamA
```

step03

```java
Team team = new Team();
team.setName("TeamA");
em.persist(team);

Member member = new Member();
member.setName("member1");
member.setAge(10);

// 양방향 연관관계 설정
// Team의 List에 Member 저장 & Member의 team 변수에 Team 저장
team.getMembers().add(member);
member.setTeam(team);  //생략시 member table의 team_id 컬럼은 null 보유

em.persist(member);

Member member2 = new Member();
member2.setName("member2");
member2.setAge(20);

// 양방향 연관관계 설정
team.getMembers().add(member2);
member2.setTeam(team);

em.persist(member2);
```

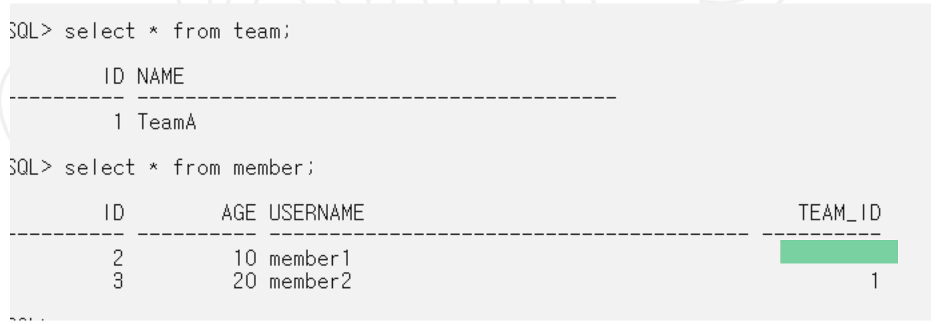


```
SQL> select * from member;

        ID        AGE USERNAME                                    TEAM_ID
---------- ---------- ---------------------------------------- ----------
         2         10 member1                                           1
         3         20 member2                                           1

SQL> select * from team;

        ID NAME
---------- ----------------------------------------
         1 TeamA
```

step03

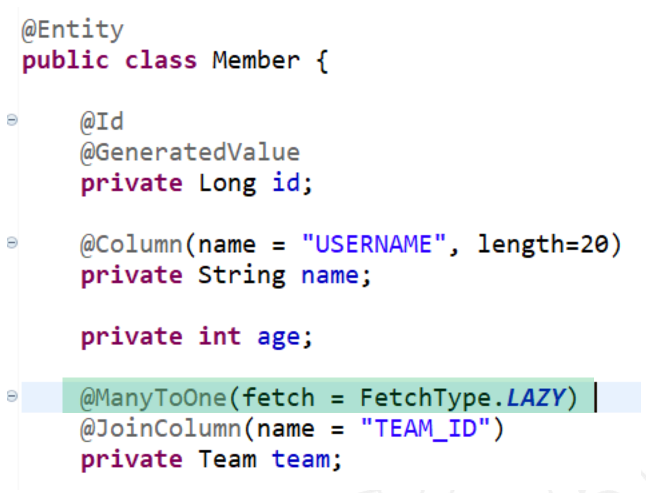
- 매핑 코드 생략시

```
Team team = new Team();
team.setName("TeamA");
em.persist(team);

Member member = new Member();
member.setName("member1");
member.setAge(10);

// 양방향 연관관계 설정
// Team의 List에 Member 저장 & Member의 team 변수에 Team 저장
team.getMembers().add(member);
//member.setTeam(team);   //생략시 member table의 team_id 컬럼은 null 보유

em.persist(member);

Member member2 = new Member();
member2.setName("member2");
member2.setAge(20);

// 양방향 연관관계 설정
team.getMembers().add(member2);
member2.setTeam(team);

em.persist(member2);
```

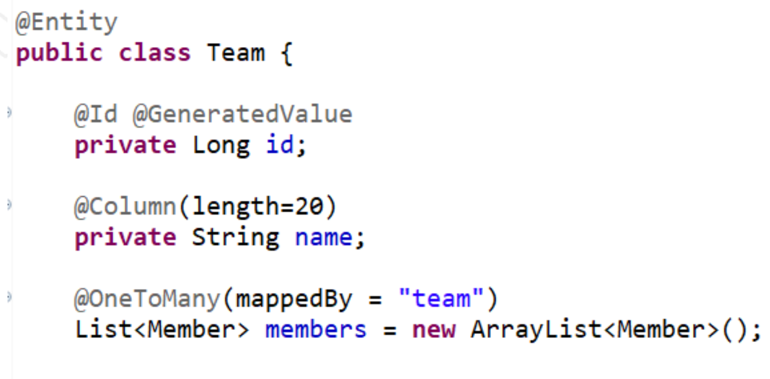


```
SQL> select * from team;

        ID NAME
---------- ----------------------------------------
         1 TeamA

SQL> select * from member;

        ID        AGE USERNAME                                    TEAM_ID
---------- ---------- ---------------------------------------- ----------
         2         10 member1
         3         20 member2                                           1
```

step04

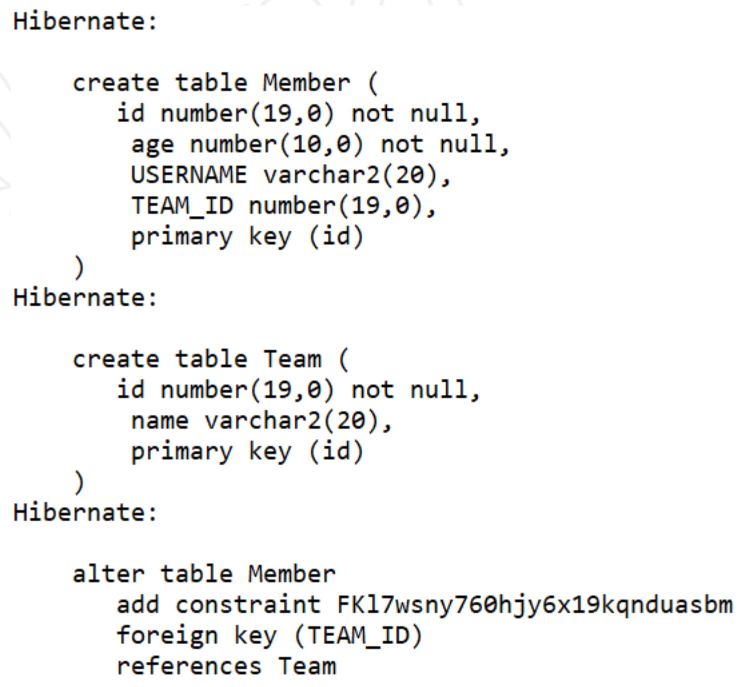


```
@Entity
public class Member {

    @Id
    @GeneratedValue
    private Long id;

    @Column(name = "USERNAME", length=20)
    private String name;

    private int age;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "TEAM_ID")
    private Team team;
```



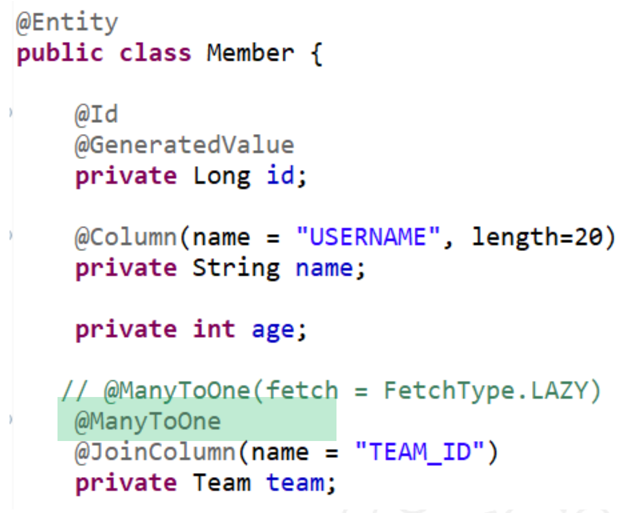
```
@Entity
public class Team {

    @Id @GeneratedValue
    private Long id;

    @Column(length=20)
    private String name;

    @OneToMany(mappedBy = "team")
    List<Member> members = new ArrayList<Member>();
```

실행시 table 생성 sql

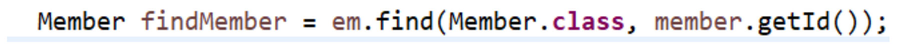


```
Hibernate: 

    create table Member (
       id number(19,0) not null,
        age number(10,0) not null,
        USERNAME varchar2(20),
        TEAM_ID number(19,0),
        primary key (id)
    )
Hibernate: 

    create table Team (
       id number(19,0) not null,
        name varchar2(20),
        primary key (id)
    )
Hibernate: 

    alter table Member
       add constraint FK17wsny760hjy6x19kqnduasbm
       foreign key (TEAM_ID)
       references Team
```

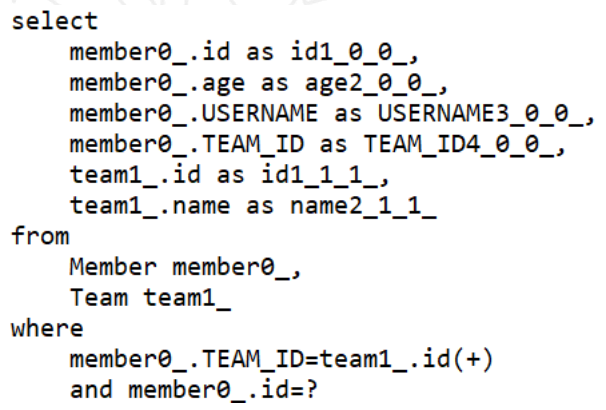
step04 – LAZY 미 적용시



```java
@Entity
public class Member {

    @Id
    @GeneratedValue
    private Long id;

    @Column(name = "USERNAME", length=20)
    private String name;

    private int age;

//  @ManyToOne(fetch = FetchType.LAZY)
    @ManyToOne
    @JoinColumn(name = "TEAM_ID")
    private Team team;
```



```java
Member findMember = em.find(Member.class, member.getId());
```



```sql
select
    member0_.id as id1_0_0_,
    member0_.age as age2_0_0_,
    member0_.USERNAME as USERNAME3_0_0_,
    member0_.TEAM_ID as TEAM_ID4_0_0_,
    team1_.id as id1_1_1_,
    team1_.name as name2_1_1_
from
    Member member0_,
    Team team1_
where
    member0_.TEAM_ID=team1_.id(+)
    and member0_.id=?
```

step04 – 지연 로딩과 즉시 로딩

지연 로딩	즉시 로딩
객체가 실제 사용될 때 로딩됨	JOIN SQL로 한번에 연관된 객체까지 미리 조회

step04 – 지연 로딩과 즉시 로딩

지연 로딩

```
Member member = memberDAO.find(memberId);
Team team = member.getTeam();
String teamName = team.getName();
```

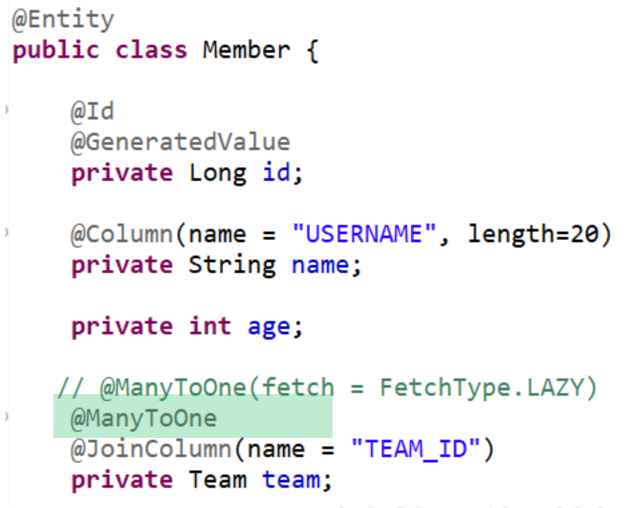
```
SELECT * FROM MEMBER

SELECT * FROM TEAM
```

즉시 로딩

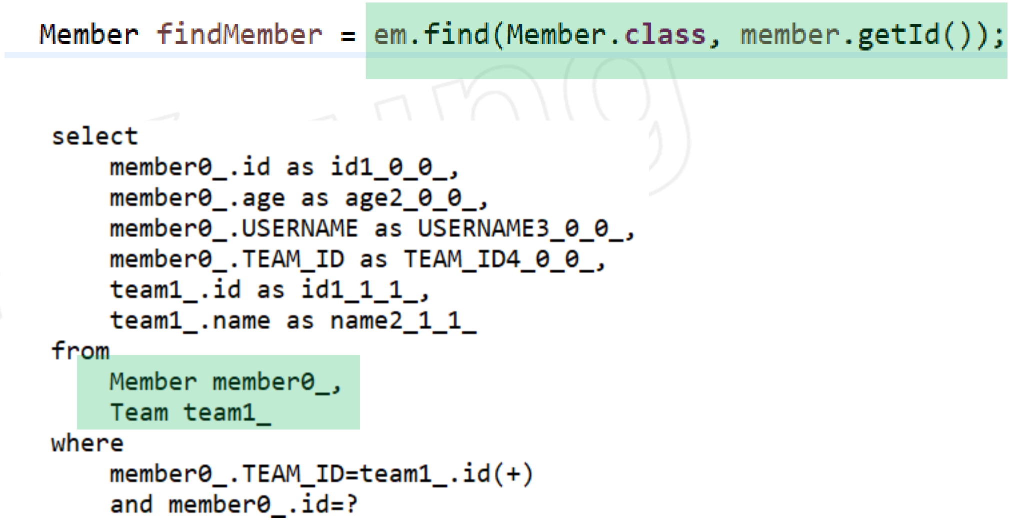
```
Member member = memberDAO.find(memberId);
Team team = member.getTeam();
String teamName = team.getName();
```

```
SELECT M.*, T.*
FROM MEMBER
JOIN TEAM …
```

step04 – LAZY 미 적용시



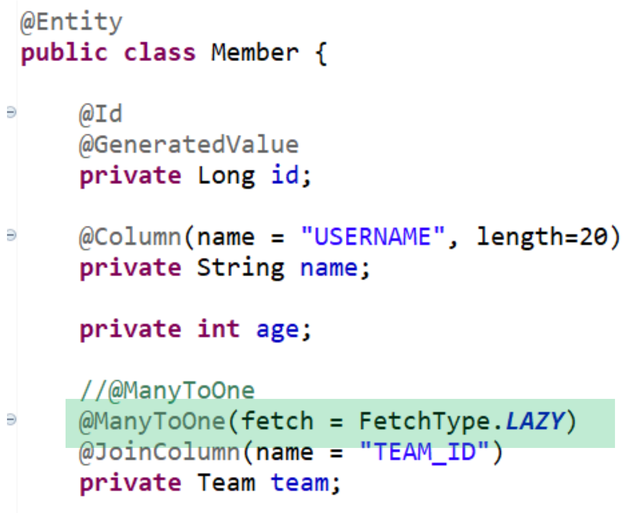
```java
@Entity
public class Member {

    @Id
    @GeneratedValue
    private Long id;

    @Column(name = "USERNAME", length=20)
    private String name;

    private int age;

//  @ManyToOne(fetch = FetchType.LAZY)
    @ManyToOne
    @JoinColumn(name = "TEAM_ID")
    private Team team;
```



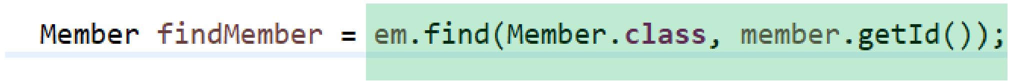
```java
Member findMember = em.find(Member.class, member.getId());
```

```sql
select
    member0_.id as id1_0_0_,
    member0_.age as age2_0_0_,
    member0_.USERNAME as USERNAME3_0_0_,
    member0_.TEAM_ID as TEAM_ID4_0_0_,
    team1_.id as id1_1_1_,
    team1_.name as name2_1_1_
from
    Member member0_,
    Team team1_
where
    member0_.TEAM_ID=team1_.id(+)
    and member0_.id=?
```

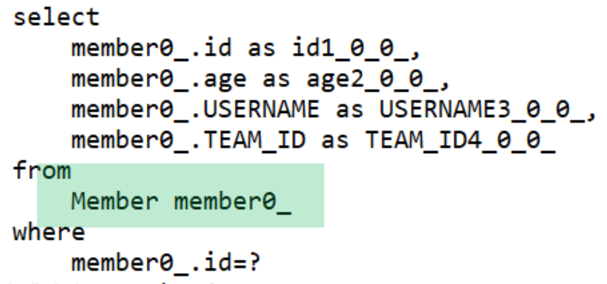
step04 – LAZY 적용시



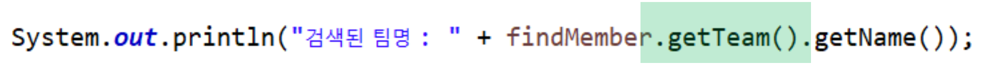
```
@Entity
public class Member {

    @Id
    @GeneratedValue
    private Long id;

    @Column(name = "USERNAME", length=20)
    private String name;

    private int age;

    //@ManyToOne
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "TEAM_ID")
    private Team team;
```



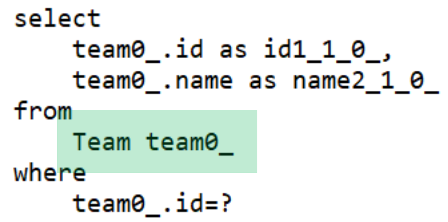
```
Member findMember = em.find(Member.class, member.getId());
```



```
select
    member0_.id as id1_0_0_,
    member0_.age as age2_0_0_,
    member0_.USERNAME as USERNAME3_0_0_,
    member0_.TEAM_ID as TEAM_ID4_0_0_
from
    Member member0_
where
    member0_.id=?
```



```
System.out.println("검색된 팀명 : " + findMember.getTeam().getName());
```



```
select
    team0_.id as id1_1_0_,
    team0_.name as name2_1_0_
from
    Team team0_
where
    team0_.id=?
```