Field-Column Mapping

- **@Column** 컬럼 매핑
- **@Temporal** 날짜 타입 매핑
- **@Enumerated** enum 타입 매핑
- **@Lob** BLOB, CLOB 매핑
- **@Transient** 매핑하지 않음

@Column

속성	설명	기본값
name	필드와 매핑할 테이블의 컬럼 이름	객체의 필드 이름
insertable, updatable	변경 가능 여부	TRUE
nullable	null 값의 허용 여부를 설정 false로 설정하면 DDL 생성 시에 not null 제약조건이 붙는다.	
unique(DDL)	유니크 제약조건을 걸 때 사용한다.	
columnDefinition (DDL)	데이터베이스 컬럼 정보를 직접 줄 수 있다. ex) varchar(100) default 'EMPTY'	필드의 타입과 디비의 방언 정보를 적용한 적절한 컬럼 타입
length(DDL)	문자 길이 제약조건, String 타입에만 사용한다.	255
precision, scale(DDL)	BigDecimal 타입에서 사용한다(BigInteger도 사용할 수 있다). precision: 소수점을 포함한 전체 자릿수 scale: 소수의 자릿수 참고로 double, float 타입에는 적용되지 않는다. 아주 큰 숫자나 정 밀한 소수를 다루어야 할 때만 사용한다.	precision=19, scale=2

@Enumerated

자바의 Enum 타입과 매핑할 때 사용

```
public enum RoleType {
    USER, ADMIN, VENDOR
}
```

```
@Entity
public class User {
    ....

    @Enumerated
    RoleType type;
}
```

속성	설명	기본값
value	EnumType.ORDINAL: enum 순서를 데이터베이스에 저장 EnumType.STRING: enum 이름을 데이터베이스에 저장 **ORDINAL은 쓰지말자!**	EnumType.ORDINAL

ORDINAL 과 STRING 의 차이

MiniUser 클래스 (ORDINAL 을 사용했을 때)

```
package domain;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import javax.persistence.*;

@Entity
@Setter @Getter @ToString
public class MiniUser {

    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;

    @Enumerated(EnumType.ORDINAL)
    private RoleType roleType;


}
```

RoleType 이넘

```
package domain;

public enum RoleType {
    USER, ADMIN
}
```

Main 클래스

```
package domain;


import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.EntityTransaction;
import javax.persistence.Persistence;

public class Main {
    public static void main(String[] args) {

        EntityManagerFactory factory =
                Persistence.createEntityManagerFactory("myjpa");

        EntityManager em = factory.createEntityManager();
        EntityTransaction tx = em.getTransaction();
        tx.begin();
        try {
            MiniUser user = new MiniUser();
            user.setName("admin");
            user.setRoleType(RoleType.ADMIN);

            em.persist(user);

            tx.commit();
        } catch (Exception e){
            tx.rollback();
        } finally {
            em.close();
        }


    }
}
```

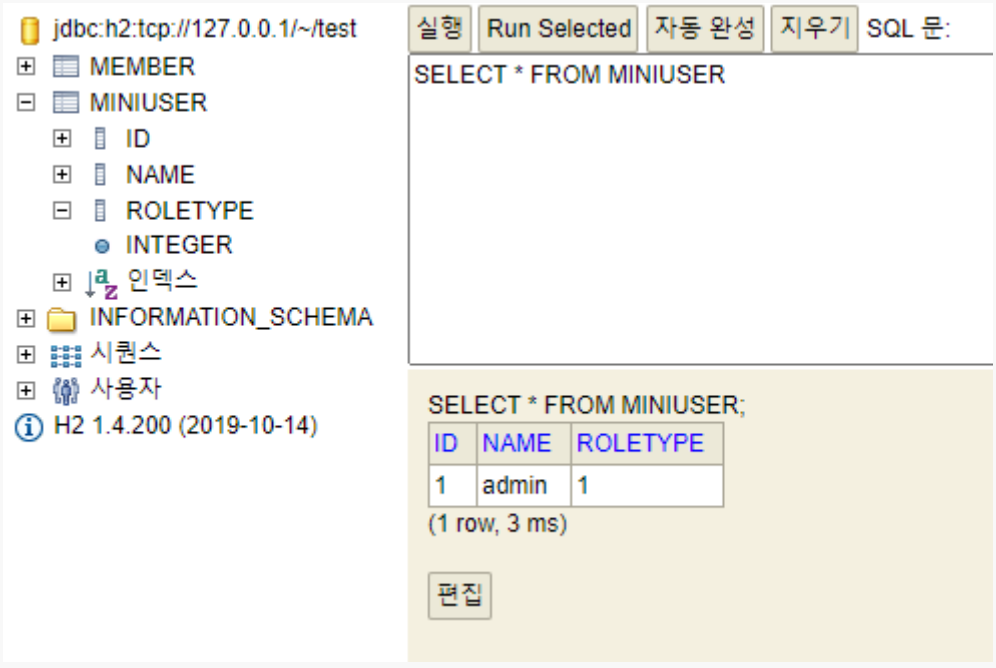
META-INF/persistence.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.2"
             xmlns="http://xmlns.jcp.org/xml/ns/persistence"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_2.xsd">
    <persistence-unit name="myjpa">
        <properties>
            <!-- 필수 속성 -->
            <property name="javax.persistence.jdbc.driver"
                      value="org.h2.Driver"/>
            <property name="javax.persistence.jdbc.user" value="sa"/>
            <property name="javax.persistence.jdbc.password" value=""/>
```

```
            <property name="javax.persistence.jdbc.url"
                      value="jdbc:h2:tcp://localhost/~/test"/>
            <property name="hibernate.dialect"
                      value="org.hibernate.dialect.H2Dialect"/>
            <!-- 옵션 -->
            <property name="hibernate.show_sql" value="true"/>
            <property name="hibernate.format_sql" value="true"/>
            <property name="hibernate.use_sql_comments" value="true"/>
            <property name="hibernate.hbm2ddl.auto" value="create" /> <!-- 이 부분
주석 해제 -->
        </properties>
    </persistence-unit>
</persistence>
```

결과

```
Hibernate:

    drop table MiniUser if exists
Hibernate:

    create table MiniUser (
       id bigint generated by default as identity,
        name varchar(255),
       roleType integer,
        primary key (id)
    )
Jan 07, 2021 3:08:51 AM org.hibernate.resource.trans
INFO: HHH10001501: Connection obtained from JdbcConn
Jan 07, 2021 3:08:51 AM org.hibernate.resource.trans
INFO: HHH10001501: Connection obtained from JdbcConn
Jan 07, 2021 3:08:51 AM org.hibernate.tool.schema.in
INFO: HHH000476: Executing import script 'org.hiberr
Hibernate:
    /* insert domain.MiniUser
        */ insert
        into
            MiniUser
            (id, name, roleType)
        values
            (null, ?, ?)
```

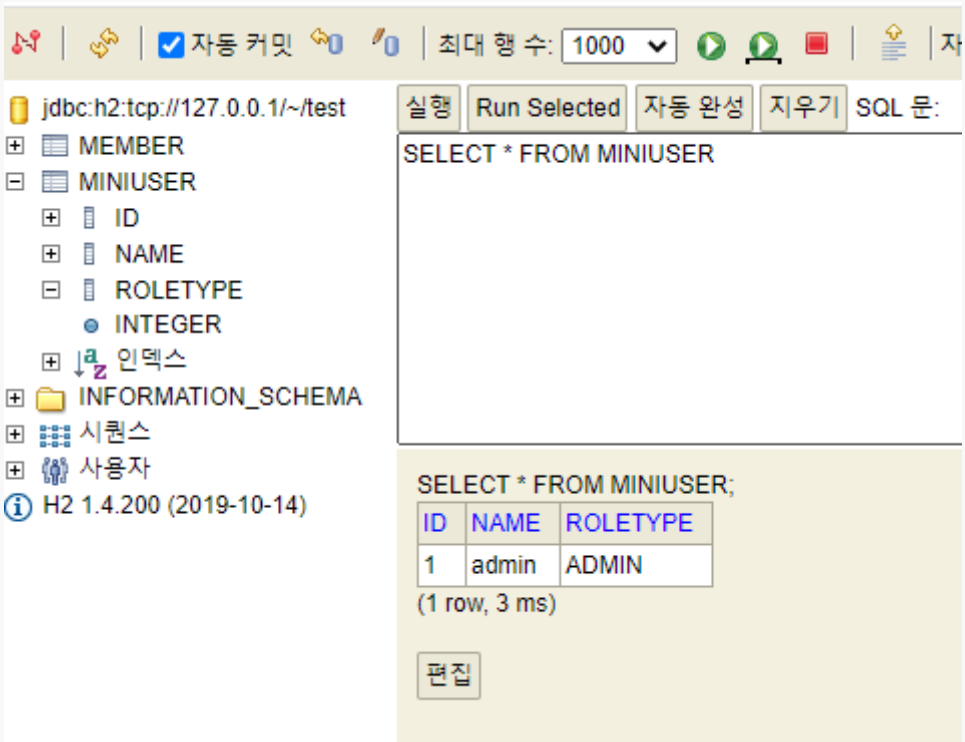


MiniUser 클래스 (STRING 을 사용했을 때)

```
package domain;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import javax.persistence.*;

@Entity
@Setter @Getter @ToString
public class MiniUser {

    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;

    @Enumerated(EnumType.STRING)
    private RoleType roleType;


}
```

결과

```
Hibernate:

    drop table MiniUser if exists
Hibernate:

    create table MiniUser (
       id bigint generated by default as identity,
        name varchar(255),
        roleType varchar(255),
        primary key (id)
    )
Jan 07, 2021 3:11:36 AM org.hibernate.resource.trans
INFO: HHH10001501: Connection obtained from JdbcConn
Jan 07, 2021 3:11:36 AM org.hibernate.resource.trans
INFO: HHH10001501: Connection obtained from JdbcConn
Jan 07, 2021 3:11:36 AM org.hibernate.tool.schema.in
INFO: HHH000476: Executing import script 'org.hibern
Hibernate:
    /* insert domain.MiniUser
        */ insert
        into
            MiniUser
            (id, name, roleType)
        values
            (null, ?, ?)
```



@Temporal

날짜 타입(java.util.Date, java.util.Calendar)을 매핑할 때 사용

참고: LocalDate, LocalDateTime을 사용할 때는 생략 가능(최신 하이버네이트 지원)

속성	설명	기본값
value	• `TemporalType.DATE`: 날짜, 데이터베이스 date 타입과 매핑 (예: 2013–10–11) • `TemporalType.TIME`: 시간, 데이터베이스 time 타입과 매핑 (예: 11:11:11) • `TemporalType.TIMESTAMP`: 날짜와 시간, 데이터베이스 timestamp 타입과 매핑(예: 2013–10–11 11:11:11)	

@Lob

데이터베이스 BLOB, CLOB 타입과 매핑

- @Lob에는 지정할 수 있는 속성이 없다.
- 매핑하는 필드 타입이 문자면 CLOB 매핑, 나머지는 BLOB 매핑
- CLOB: `String`, `char[]`, `java.sql.CLOB`
- BLOB: `byte[]`, `java.sql. BLOB`

@Transient

필드 매핑X