JPA 기초

# **▣** JPA란 무엇인가?

* 자바 애플리케이션에서 객체와 RDB(관계형 데이터베이스) 간의 데이터를 쉽게 처리할 수 있도록 도와주는 표준 API
* 개발자가 SQL을 직접 작성하지 않고도 데이터베이스에 저장, 조회, 수정, 삭제 등의 작업을 할 때 처리할 수 있게해줌
* JPA는 자바 EE의 일부로, 다양한 구현체(예: Hibernate, EclipseLink)를 통해 동작

# ▣ JPA 와 Hibernate

## JPA

* + JAVA에서 제공하는 표준 ORM(Object-Relational Mapping) API
  + 객체와 데이터베이스 간의 데이터를 매핑하는 방법을 정의한 인터페이스의 모음
  + 하지만 구현체가 아니므로 실제 동작은 할 수 없음

## Hibernate

* + JPA의 구현체 중 하나로, ORM 프레임워크
  + JPA에서 정의한 인터페이스와 메서드를 실제로 구현하여 JPA를 사용할 수 있게 함
  + 즉 JPA를 사용하면, 그 내부적으로 Hibernate와 같은 구현체를 통해 기능을 구현

# **▣** 주요 용어

## 1. 엔티티(Entity)

* 테이블에 해당하는 자바 객체로 각 행(row)은 엔티티 객체의 인스턴스로 표현
* 레코드를 자바 객체로 매핑하여, 데이터베이스에서 읽어오거나 저장할 수 있음
* @Entity @Id 와 같은 어노테이션 사용

## 2. 엔티티 매니저(EntityManager)

* JPA에서 가장 핵심적인 인터페이스
* 엔티티에 대한 CRUD(Create, Read, Update, Delete) 작업을 처리하는 객체
* 주요 메서드
  + persist(entity) : 엔티티를 데이터베이스에 저장
  + find(Class<T> entityClass, Object primaryKey) : 기본 키를 이용해 엔티티 조회
  + remove(entity) : 엔티티 삭제
  + merge(entity) : 엔티티 갱신

## 3. 영속성 컨텍스트(Persistence Context)

* 엔티티 매니저가 관리하는 엔티티 객체들의 집합
* 특정 시점에서 데이터베이스와 동기화되지 않은 상태의 엔티티 객체들이 포함된 컨텍스트
* 엔티티를 저장하거나 조회할 때, 영속성 컨텍스트 내에서 관리됨
* 자동으로 데이터베이스와 동기화

## 4. 영속성 유닛(Persistence Unit)

* JPA 설정의 논리적인 단위
* 데이터베이스와 상호작용할 때 필요한 엔티티 클래스 목록, 데이터베이스 설정 등을 포함
* 다양한 영속성 컨텍스트를 필요로 하는 경우 영속성 유닛을 사용하여 분리할 수 있음

## 5. JPQL (Java Persistence Query Language)

* JPA에서 제공하는 객체 지향 쿼리 언어
* SQL과 유사한 문법으로 작성
* 테이블 대신 엔티티 객체를 대상으로 질의를 수행

# 프로젝트명 : B21aJPA\_Basic

## 준비사항

* 의존설정 : Oracle Driver, Spring Data JPA
* JSP 사용을 위한 설정을 하지 않아도된다.
* Refresh Gradle Project 를 눌러 적용한다.

## 테이블

* JpaMember1 ~ JpaMember4 까지 자동생성된다.
* 직접 생성하지 않는다.

### 설정파일 : resources/META-INF/persistence.xml

* JPA의 설정 파일로, 영속성 유닛(persistence unit)을 정의하는데 사용
* 데이터베이스와의 연결, 엔티티 매핑, 트랜잭션 처리 등을 설정

#### 형식

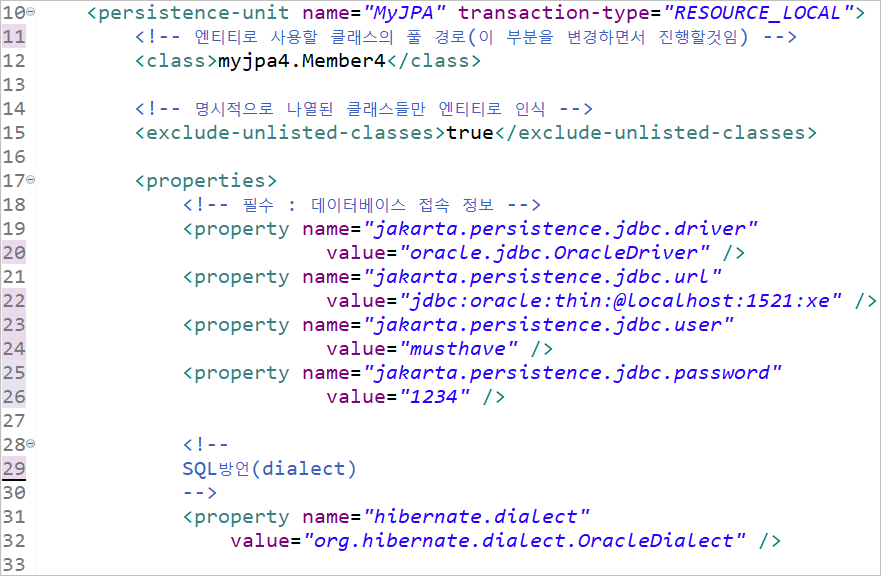
|  | <?xml version="1.0" encoding="utf-8" ?>  <persistence xmlns="네임스페이스 URL ">  <persistence-unit>  <class>JPA가 관리할 엔티티 클래스</class>  <exclude-unlisted-classes>true</exclude-unlisted-classes>  <properties>  <property name=DB접속정보 value=값 />  <property name=여러가지옵션 value=값 />  </properties>  </persistence-unit>  </persistence> |
| --- | --- |

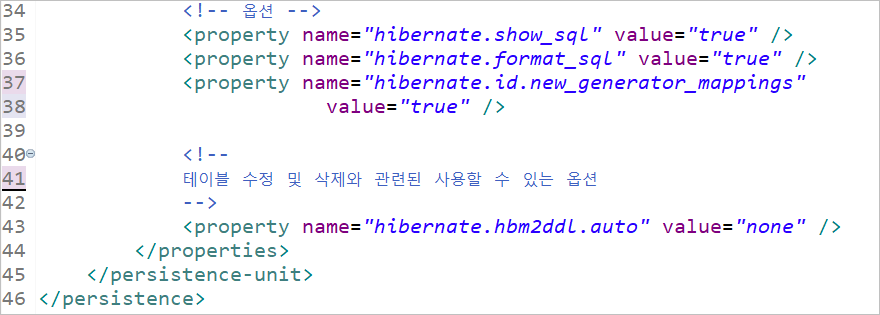
1. 영속성 유닛 정의
   1. <persistence-unit> 항목
   2. 애플리케이션 내에서 JPA가 사용할 데이터베이스와 엔티티를 설정
2. 데이터베이스 연결 설정
   1. <property> 항목
   2. 데이터소스, 계정명, 패스워드 등을 설정
3. 엔티티 클래스 목록
   1. <class> 항목
   2. JPA가 관리해야 할 엔티티 클래스를 등록
   3. <exclude-unlisted-classes>를 true로 설정하여, 명시적으로 나열된 클래스들만 엔티티로 인식
4. 추가 속성 설정
   1. 캐싱 전략, 데이터베이스 생성 정책, SQL 로그 출력 여부 등의 추가 설정 가능

| <?xml version="1.0" encoding="utf-8" ?>  <persistence xmlns="https://jakarta.ee/xml/ns/persistence"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="https://jakarta.ee/xml/ns/persistence  https://jakarta.ee/xml/ns/persistence/persistence\_3\_0.xsd"  version="3.0"> |
| --- |

<class>myjpa4.Member4</class> ⇒ 이 부분을 myjpa1.Member1 로 수정 후 테스트

hibernate.hbm2ddl.auto ⇒ create 로 수정 후 테스트





#### 최근 버전의 Dialect에서는 버전별로 각각의 Dialect를 사용하는 대신 통합된 Dialect를 사용한다.

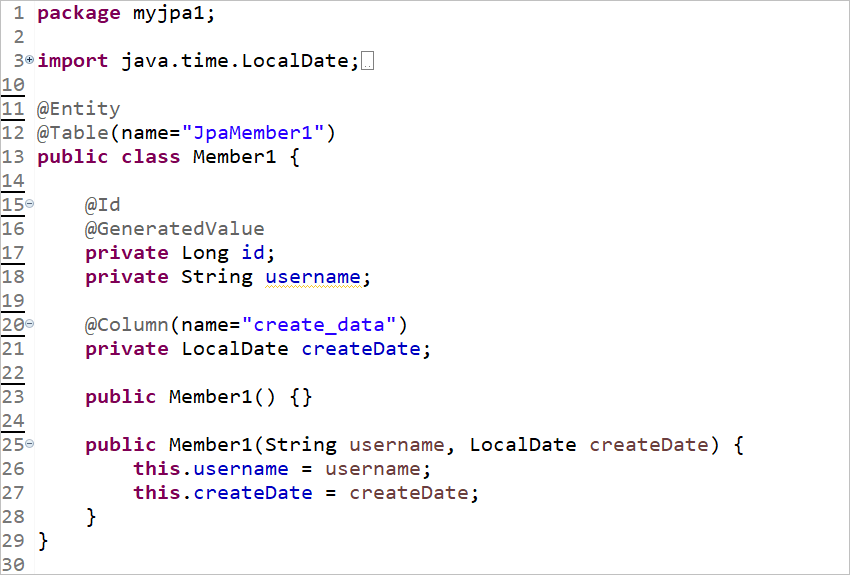
|  | <property name="hibernate.dialect" value="org.hibernate.dialect.OracleDialect" /> |
| --- | --- |

#### hibernate.hbm2ddl.auto

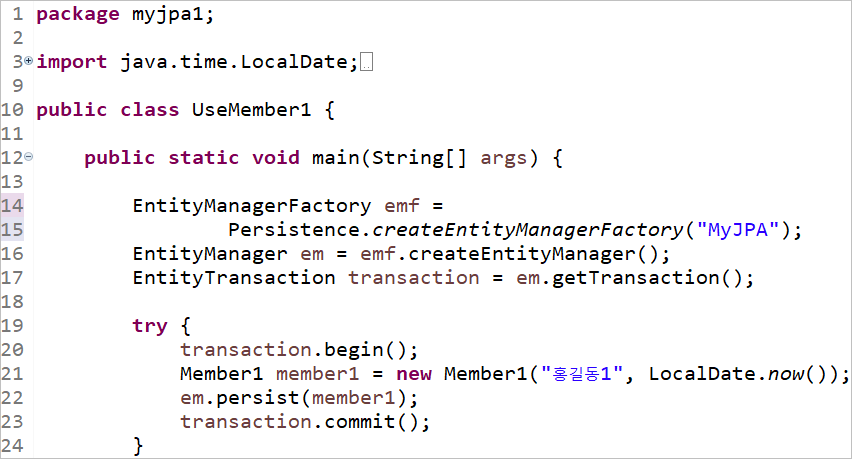
| create | 시작시 기존 테이블 삭제 후 다시 생성 drop + create |
| --- | --- |
| create-drop | create와 같지만 종료시점에 테이블을 drop |
| update | 변경분만 반영. 추가만 되고, 지워지는건 안됨. |
| validate | 엔티티와 테이블이 정상 매핑되었는지만 확인 |
| none | 사용하지 않음 |

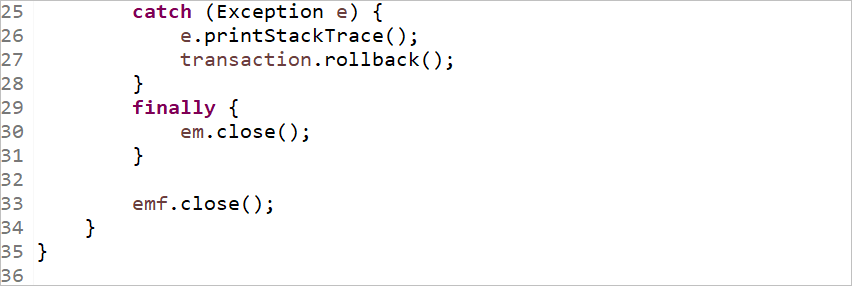
## 첫번째실습

### /B21aJPA\_Basic/src/main/java/myjpa1/Member1.java



### /B21aJPA\_Basic/src/main/java/myjpa1/UseMember1.java





실행은 우클릭 ⇒ Run As ⇒ Java Application 클릭한다.

SQL Developer 에서 jpamember1 테이블이 생성되었는지 확인한다.