02 기본 CRUD 실습

출처
https://vladmihalcea.com/tutorials/hibernate/
https://vladmihalcea.com/books/high-performance-java-persistence/

환경설정

학습환경 세팅

- **Java**
  - java 11
- **IDE**
  - 인텔리제이
- **데이터베이스**
  - MySQL
- **빌드툴**
  - maven

간단 예제 만들기 순서

- MySQL데이터베이스 및 테스트 테이블 생성
- maven 프로젝트 생성
- 필요 라이브러리 의존 설정
- META-INF폴더 생성 및 persistence.xml 생성
- Main, Movie class 작성
- Movie class(엔티티) 어노테이션 설정

MySQL데이터베이스 및 테스트 테이블 생성

- 새로운 데이터베이스 생성: jpa_playground
- MOVIE 테이블 생성

```
CREATE TABLE `jpa_playground`.`movie` (
  `MOVIE_ID` INT NOT NULL,
  `NAME` VARCHAR(45) NULL,
  PRIMARY KEY (`MOVIE_ID`)
);
```

- IntelliJ에서 maven프로젝트 생성

프로젝트 생성

- **pom.xml에 JPA, MySQL 라이브러리 설치**
- **의존 설정은 구글에 maven repository에서 검색**
  - Hibernate EntityManager Relocation
  - MySQL Connector/J
- **오픈소스나 소스 라이브러리 release 시 GA, SNAPSHOT의 의미**
  - GA (General Availability) - 테스트가 완료된 정식 릴리즈 버전으로 안정적으로 운영되어야 하는 프로젝트에서 사용.
  - RC (Release Candidate) - 베타 버전. 정식 릴리즈 버전은 아니므로 기능은 확정 되었으나, 안정적 동작은 보장할 수 없음
  - M (Milestone) - 테스트 버전
  - SNAPSHOT - 스냅샷이 붙으면 아직 개발단계라는 의미이며, 일종의 백업시점

의존 설정

- 의존 설정 참고

```
<dependencies>
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.16</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-entitymanager -->
    <dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-entitymanager</artifactId>
        <version>5.4.22.Final</version>
    </dependency>
</dependencies>
```

META-INF폴더 생성 및 persistence.xml 생성

- **예제 프로젝트 구조**

```
src/main
        ->java
                ->study
                        -> Main.java
                        -> Movie.java
         -> resources
                -> META-INF
                        -> persistence.xml
```

  - META-INF가 없다면 폴더 생성하기(언더스코어가 아니라 하이픈임을 주의)

META-INF폴더 생성 및 persistence.xml 생성

```
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd"
        version="2.0">

  <persistence-unit name="playground" transaction-type="RESOURCE_LOCAL">
    <description>
      Persistence unit for the JPA tutorial of the Hibernate Getting Started Guide
    </description>
    <provider>org.hibernate.ejb.HibernatePersistence</provider>

    <properties>
      <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver" />
      <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/jpa_playground?serverTimezone=UTC" />
      <property name="javax.persistence.jdbc.user" value="kim" />
      <property name="javax.persistence.jdbc.password" value="sungryulKim12" />

      <property name="hibernate.dialect" value="org.hibernate.dialect.MySQL8Dialect " />
      <property name="hibernate.show_sql" value="true" />
      <!--<property name="hibernate.hbm2ddl.auto" value="create" /> -->
    </properties>

  </persistence-unit>
</persistence>
```

참고: https://gist.github.com/halyph/2990769

예제 작성

Movie class 작성

- 의존 설정 참고

```
@Entity
@Table(name="MOVIE")
public class Movie {
    @Id
    @Column(name="ID")
    private Long id;
    @Column(name="NAME")
    private String movieName;
```

Main class

```
public static void Main(String[] args) {
    EntityManagerFactory emf = Persistence.createEntityManagerFactory(" playground ");
    EntityManager em = emf.createEntityManager();

    EntityTransaction tx = em.getTransaction();

    try {
        tx.begin();
        Movie movie = new Movie();
        movie.setMovieId(1L);
        movie.setMovieName("타이타닉");
        em.persist(movie);
        tx.commit();
    } catch (Exception e) {
        tx.rollback();
    } finally {
        em.close();
    }
    emf.close();
}
```

JPA관련 코드는 트랜잭션 안에서 수행

트랜잭션의 중요성

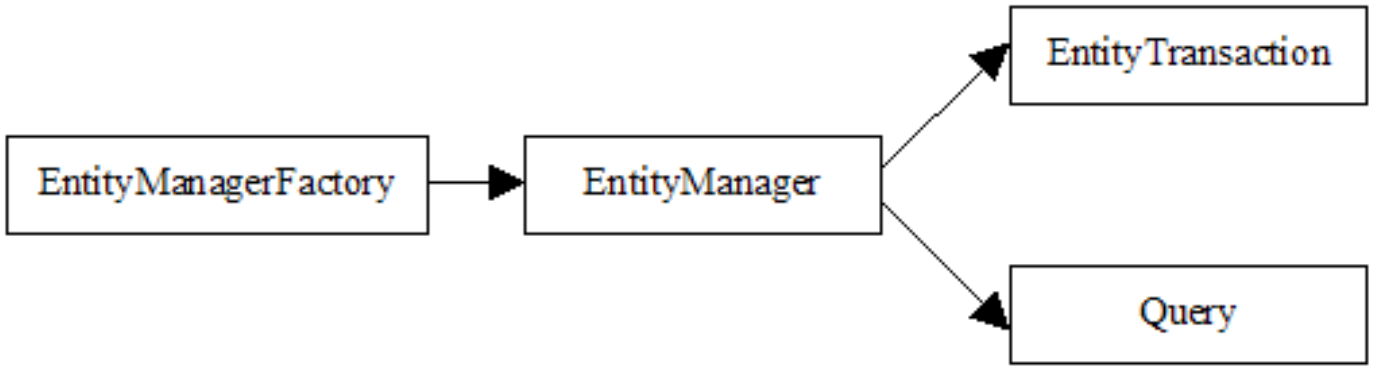
- JPA의 수행 로직은 반드시 트랜잭션 내에서 수행해야 함
- 트랜잭션 밖에서 수행할 경우 에러 발생

```
try {

    tx.begin();

    JPA 관련 코드

    tx.commit();

} catch (Exception e) {
    tx.rollback();
}
```

- 주의: persist 메소드는 엔티티를 영속성 컨텍스트라는 곳에 저장하는 것이지 엔티티의 내용을 DB에 저장하는 것은 아님
- 저장은 언제? → 트랙잭션이 종료되는 시점에 DB에 전달할 모든 SQL을 모아서 한 번에 처리

엔티티 매니저 설정

- **엔티티 매니저 팩토리와 엔티티 매니저**
  - 엔티티 매니저가 DB 연결을 유지하고 있음
  - 엔티티 매니저는 JPA의 대부분의 기능(DB와 연동되는 부분)을 제공
  - 엔티티 매니저 팩토리가 엔티티 매니저 인스턴스를 제공
  - 서로 다른 DB에 접근할 경우, 복수 개의 엔티티 매니저 팩토리 생성
  - 엔티티 매니저 팩토리는 생성 오버헤드가 있으므로 싱글톤 형태로 관리



[출처] https://letslearnjava.quora.com/Entity-Manager-in-JPA

기본 CRUD

- **등록**

```
em.persist(movie);
```

- **조회**

```
Movie movie = em.find(Movie.class, 1L);
```

- **수정**

```
Movie movie = em.find(Moive.class, 1L);
member.setMovieName("돈룩업")
```

- **삭제**

```
em.remove(movie);
```