JPA에서querydsl적용하기

데이터엔지니어링팀 서정은

LINE GAMES

목차

- QueryDsl
- 도입 이유
- 사용 예제
- QueryDsl의 장점
- QueryDsl의 단점
- Reference

QueryDsl

QueryDsl?

정적 타입을 이용해서 SQL과 같은 쿼리를 생성할 수 있도록 해주는 프레임워크

도입 이유

- AS-IS

Repository

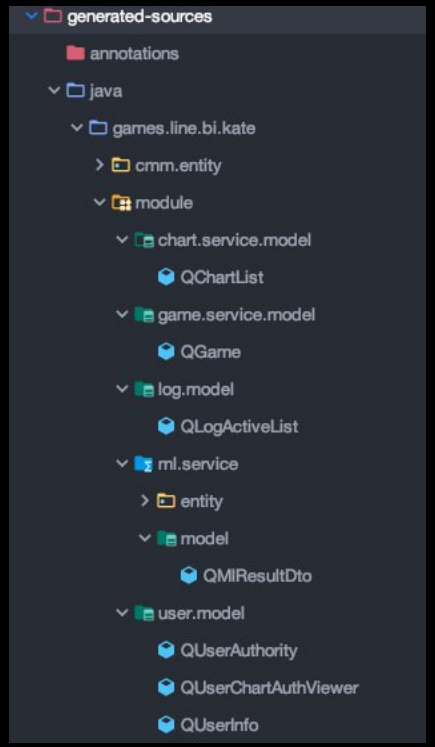
```
private HibernateQuery getUserListCommon(UserVo userVo) {
    HibernateQuery query = new HibernateQuery(UserInfo.class, hasOrder: true, hasOrParam: true);
    UserInfo userInfo = CommonUtil.getCurrentUser();
    boolean isProjectNull = (userVo.getProjectSeq()==null||"".equals(userVo.getProjectSeq()));
    List<Long> userSeqs = null;
    if(userInfo.getRole().equals(UserRole.PROJECT_MANAGER)) {
        String subQuery = "SELECT DISTINCT(i.userSeq) FROM UserInfo i LEFT JOIN UserAuthority a ON i.userSeq=a.userSeq "
            + "WHERE i.status <> :status AND i.role NOT IN :notRole AND i.allView=0 "
            + "AND ((i.role <> 'PROJECT_MANAGER' OR (i.role = 'PROJECT_MANAGER' AND a.projectSeq IN (SELECT projectSeq FROM UserAuthority WHERE userSeq =: userSeq))) "
            + (isProjectNull?"OR a.projectSeq IS NULL":"AND a.projectSeq = " + userVo.getProjectSeq())+")";
        userSeqs = entityManager.createQuery(subQuery)
            .setParameter( s: "notRole", Arrays.asList(UserRole.SUPER_ADMIN, UserRole.ADMIN))
            .setParameter( s: "status", UserStatus.DELETED)
            .setParameter( s: "userSeq", userInfo.getUserSeq())
            .getResultList();
    }else {
        String subQuery = "SELECT DISTINCT(i.userSeq) FROM UserInfo i LEFT JOIN UserAuthority a ON i.userSeq=a.userSeq "
            + "WHERE i.status <> " + String.format("'%s'", userInfo.getRole().equals(UserRole.ADMIN)?UserStatus.DELETED.name():"")
            + (isProjectNull?"":" AND a.projectSeq = " + userVo.getProjectSeq() + " OR i.role IN ('ADMIN','SUPER_ADMIN')");
        userSeqs = entityManager.createQuery(subQuery)
            .getResultList();
    }
    query.addCondition(ConditionType.IN, column: "userSeq", userSeqs);
    query.addCondition(ConditionType.EQUAL, column: "role", userVo.getRole());
    if(userVo.getSearch() != null &&!"".equals(userVo.getSearch())) {
        query.addOrCondition(ConditionType.LIKE, column: "name", userVo.getSearch());
        query.addOrCondition(ConditionType.LIKE, column: "userId", userVo.getSearch());
    }

    return query;
}
```

사용 예제

Pom.xml

```xml
<!-- dependency -->
<!-- https://mvnrepository.com/artifact/com.querydsl/querydsl-jpa -->
<dependency>
    <groupId>com.querydsl</groupId>
    <artifactId>querydsl-jpa</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/com.querydsl/querydsl-apt -->
<dependency>
    <groupId>com.querydsl</groupId>
    <artifactId>querydsl-apt</artifactId>
</dependency>
```

```xml
<plugin>
    <groupId>com.mysema.maven</groupId>
    <artifactId>apt-maven-plugin</artifactId>
    <version>1.1.3</version>
    <executions>
        <execution>
            <goals>
                <goal>process</goal>
            </goals>
            <configuration>
                <outputDirectory>target/generated-sources/java</outputDirectory>
                <processor>com.querydsl.apt.jpa.JPAAnnotationProcessor</processor>
                <options>
                    <querydsl.entityAccessors>true</querydsl.entityAccessors>
                </options>
            </configuration>
        </execution>
    </executions>
</plugin>
```



사용 예제

QuerydslCofig

```
@Configuration
public class QuerydslConfig {
    @PersistenceContext
    private EntityManager entityManager;

    @Bean
    public JPAQueryFactory jpaQueryFactory() { return new JPAQueryFactory(entityManager); }
}
```

사용 예제

1. Repository 생성

```
 */
public interface UserRepository extends JpaRepository<UserInfo, Long>, QuerydslPredicateExecutor<UserInfo>, UserRepositoryCustom {
    List<UserInfo> findUserInfoByUserName(String userName);
}
```

2. RepositoryCustom생성

```
 */
public interface UserRepositoryCustom {
    List<UserInfo> findUserInfoByUserName(String userName);
}
```

사용 예제

3. RepositoryCustomImpl 생성 및 QueryDsl 작성

```java
public class UserRepositoryCustomImpl implements UserRepositoryCustom {

    private final JPAQueryFactory queryFactory;

    public UserRepositoryCustomImpl(JPAQueryFactory queryFactory) {
        this.queryFactory = queryFactory;
    }

    @Override
    public List<UserInfo> findUserInfoByUserName (String userName) {
        QueryResults<UserInfo> results = queryFactory
                .selectFrom(QUserInfo.userInfo)
                .where(searchByLike(userName))
                .orderBy(QUserInfo.userInfo.rgsde.desc())
                .fetchResults();
        return results.getResults();
    }

    private BooleanExpression searchByLike(String searchQuery){

        return QUserInfo.userInfo.name.like( str: "%" + searchQuery + "%");

    }
}
```

사용 예제

4. Results


```
▼ [{userSeq: 48, userId: "test1111", email: "seoje1594@line.games", name: "정은1", role: "USER",…},…]
  ▼ 0: {userSeq: 48, userId: "test1111", email: "seoje1594@line.games", name: "정은1", role: "USER",…}
      allView: false
      authList: []
      authMap: {}
      company: "LINE GAMES"
      email: "seoje1594@line.games"
      name: "정은1"
      role: "USER"
      status: "NORMAL"
      userId: "test1111"
      userSeq: 48
  ▼ 1: {userSeq: 41, userId: "th1594", email: "seoje1594@line.games", name: "정은테스트", role: "PROJECT_MANAGER",…}
      allView: true
    ▶ authList: [,…]
    ▶ authMap: {,…}
      company: "LINE GAMES"
      email: "seoje1594@line.games"
      name: "정은테스트"
      role: "PROJECT_MANAGER"
      status: "NORMAL"
      userId: "th1594"
      userSeq: 41
  ▶ 2: {userSeq: 40, userId: "seoje1594", email: "seoje1594@line.games", name: "서정은", role: "SUPER_ADMIN",…}
```

QueryDsl 의 장점

- 컴파일 시점에서 디버깅 가능 및 가독성 향상
- 자동 완성 등 IDE의 보조기능
- 동적인 쿼리 작성이 편리
- 메서드 추출을 통한 재사용성 (쿼리 작성 시 제약 조건 등)

QueryDsl 의 단점

- From 절의 서브 쿼리 지원하지 않음
- 복잡한 쿼리에는 적합하지 않음

참고 링크

- https://tecoble.techcourse.co.kr/post/2021-08-08-basic-querydsl/
- https://blog.kuberix.co.kr/2020/02/19/tutorial-querydsl.html
- https://velog.io/@jkijki12/Spring-QueryDSL-%EC%99%84%EB%B2%BD-%EC%9D%B4%ED%95%B4%ED%95%98%EA%B8%B0
- https://studio108.tistory.com/26
- https://dico.me/java/articles/290/ko