8. 스프링 부트 3.x(2.6 이상), Querydsl 5.0 지원 방법

#1.인강/JPA활용편/querydsl/강의

- /build.gradle 설정 방법
- /PageableExecutionUtils Deprecated(향후 미지원) 패키지 변경
- /Querydsl fetchResults(), fetchCount() Deprecated(향후 미지원)

> 참고: 해당 내용은 강의 이후에 추가된 내용입니다.

최신 스프링 부트 3.x(2.6부터)은 Querydsl 5.0을 사용한다.

스프링 부트 3.x(2.6 이상) 사용시 다음과 같은 부분을 확인해야 한다.

1. `build.gradle` 설정 변경
2. `PageableExecutionUtils` Deprecated(향후 미지원) 패키지 변경
3. Querydsl `fetchResults()`, `fetchCount()` Deprecated(향후 미지원)

build.gradle 설정 방법

스프링 부트 3.x(2.6도 포함)관련 Querydsl 설정에 대한 자세한 내용은 다음 링크를 확인해주세요: https://bit.ly/springboot3

PageableExecutionUtils Deprecated(향후 미지원) 패키지 변경

PageableExecutionUtils 클래스 사용 패키지 변경

기능이 Deprecated 된 것은 아니고, 사용 패키지 위치가 변경되었습니다. 기존 위치를 신규 위치로 변경해주시면 문제 없이 사용할 수 있습니다.

- 기존: `org.springframework.data.repository.support.PageableExecutionUtils`
- 신규: `org.springframework.data.support.PageableExecutionUtils`

Querydsl fetchResults(), fetchCount() Deprecated(향후 미지원)

Querydsl의 `fetchCount()`, `fetchResult()`는 개발자가 작성한 select 쿼리를 기반으로 count용 쿼리를 내부에서 만들어서 실행합니다.
그런데 이 기능은 강의에서 설명드린 것 처럼 select 구문을 단순히 count 처리하는 용도로 바꾸는 정도입니다. 따라서 단순한 쿼리에서는 잘 동작하지만, 복잡한 쿼리에서는 제대로 동작하지 않습니다.
Querydsl은 향후 `fetchCount()`, `fetchResult()`를 지원하지 않기로 결정했습니다.
참고로 Querydsl의 변화가 빠르지는 않기 때문에 당장 해당 기능을 제거하지는 않을 것입니다.

따라서 count 쿼리가 필요하면 다음과 같이 별도로 작성해야 합니다.

count 쿼리는 예제

```
@Test
public void count() {
    Long totalCount = queryFactory
            //.select(Wildcard.count) //select count(*)
            .select(member.count()) //select count(member.id)
            .from(member)
            .fetchOne();

    System.out.println("totalCount = " + totalCount);
}
```

- `count(*)`을 사용하고 싶으면 예제의 주석처럼 `Wildcard.count`를 사용하시면 됩니다.
- `member.count()`를 사용하면 `count(member.id)`로 처리됩니다.
- 응답 결과는 숫자 하나이므로 `fetchOne()`을 사용합니다.

`MemberRepositoryImpl.searchPageComplex()` 예제에서 보여드린 것 처럼 select 쿼리와는 별도로 count 쿼리를 작성하고 `fetch()`를 사용해야 합니다. 다음은 최신 버전에 맞추어 수정된 예제입니다.

수정된 searchPageComplex 예제

```
import org.springframework.data.support.PageableExecutionUtils; //패키지 변경

public Page<MemberTeamDto> searchPageComplex(MemberSearchCondition condition,
Pageable pageable) {
    List<MemberTeamDto> content = queryFactory
            .select(new QMemberTeamDto(
                    member.id.as("memberId"),
                    member.username,
                    member.age,
                    team.id.as("teamId"),
```

```
                        team.name.as("teamName")))
                .from(member)
                .leftJoin(member.team, team)
                .where(
                        usernameEq(condition.getUsername()),
                        teamNameEq(condition.getTeamName()),
                        ageGoe(condition.getAgeGoe()),
                        ageLoe(condition.getAgeLoe())
                )
                .offset(pageable.getOffset())
                .limit(pageable.getPageSize())
                .fetch();

        JPAQuery<Long> countQuery = queryFactory
                .select(member.count())
                .from(member)
                .leftJoin(member.team, team)
                .where(
                        usernameEq(condition.getUsername()),
                        teamNameEq(condition.getTeamName()),
                        ageGoe(condition.getAgeGoe()),
                        ageLoe(condition.getAgeLoe())
                );

        return PageableExecutionUtils.getPage(content, pageable,
    countQuery::fetchOne);
    }
```