SqlKata를 이용한 Query 관리 기법

안현모

https://github.com/bluepope

https://github.com/bluepope/SqlKataSample

Agenda

1. SqlKata란?

2. 왜 SqlKata 를 선택했는가?

3. EF Core, Dapper 와의 차이

4. Sample Code

5. 응용사례

1. SqlKata 란

Open Source Query Builder

- Query String 을 생성합니다.
- SqlServer, MySQL, PostgreSql, Oracle, SQLite, Firebird를 지원합니다.
- Sql Injection 문제가 해소됩니다.
- 유연한 문법으로 Join, SubQuery 등을 지원합니다.
- 조건에 따른 Query 분기를 손쉽게 진행할 수 있습니다.
- 직관적인 문법으로 만들어지는 Query 문자열을 쉽게 예측할수 있습니다.

https://sqlkata.com/

https://github.com/sqlkata/querybuilder

1. SqlKata 란

```
var query = db.Query("Books").OrderByDesc("PublishingDate");

if(Request.Has("category.name"))
{
    var category = Request.Get("category.name");

    query.Join("Categories", "Categories.Id", "Books.CategoryId")
        .Where("Categories.Name", category);
}

var recentBooks = query.Limit(10).Get();
```

조건 분기

Select *
From Books
Order by PublishingDate Desc

Join, Where
추가

Limit 10 추가 후 Query 생성 및 실행
※ Dapper 로 실행됩니다.

1. SqlKata 란

Query 객체 생성

↓

내용 추가

↓

Query Compile

↓

Query 실행

```
Query query = new();
query.From("user");
query.Where("id", 1);

SqlResult sqlResult = new MySqlCompiler().Compile(query);

using (MySqlConnection conn = new(connectionString))
{
    conn.Open();

    resultList = await conn.QueryAsync<UserModel>(sqlResult.Sql, sqlResult.NamedBindings);
}
```

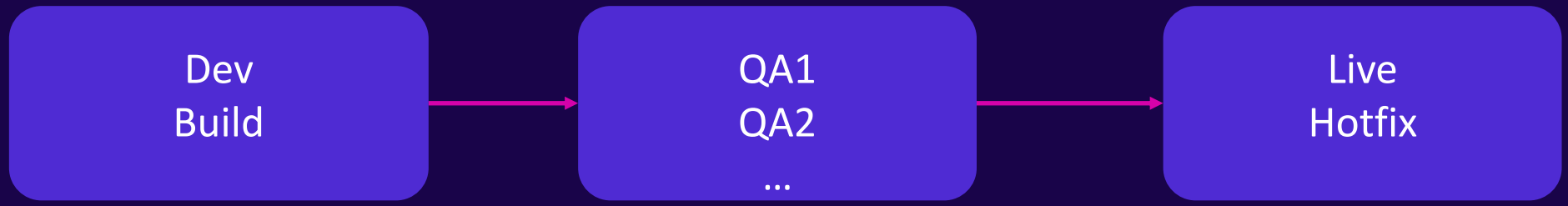
2. 왜 SqlKata 를 선택했는가?

유연한 설계 필요

- Table Column 의 존재 여부에 따른 버전 분기
- FK 가 없지만 Join 을 해야하는 경우가 존재
- Dev – QA – Live 브랜치 등으로 분기며, DB 스키마가 조금씩 다르지만 모든 DB 에 문제없이 접근가능 해야함

Query 작성의 단순화

- SELECT * 과 같은 반복적으로 생성되는 불필요한 Code 가 없어야함.
- Code는 가능한 단순하게 작성되어야하며, Model Mapping 되어야함
- 작성된 Code 에 따른 Query 가 예측이 쉬워야하며, 유지보수에 적합해야함



3. EF Core, Dapper 와의 차이

Dapper.SimpleCRUD와의 차이

- Dapper는 Query 실행 및 SqlMapper 에 사용되므로 Query 작성이 필요
- Dapper.SimpleCRUD 는 성능이 우수하나, 간단한 Query 에 적합
- 단순 CRUD 이외에는 Query 작성이 필요

EF Core 와의 차이

- Linq를 이용한 접근 방식이 매우 편리하나, 복잡한 Query 작성은 어려움
- DB 와 코드가 정확히 매치되어야하며, 그렇지 않다면 Runtime 오류가 발생
- ORM 에 적합한 DB 설계가 아니라면 오히려 Query Code 작성 난이도가 상승함
- 대량작업에는 부적합

4. Sample Code

일반적인 CRUD

- Select
- Insert
- Update
- Delete

고급 Query

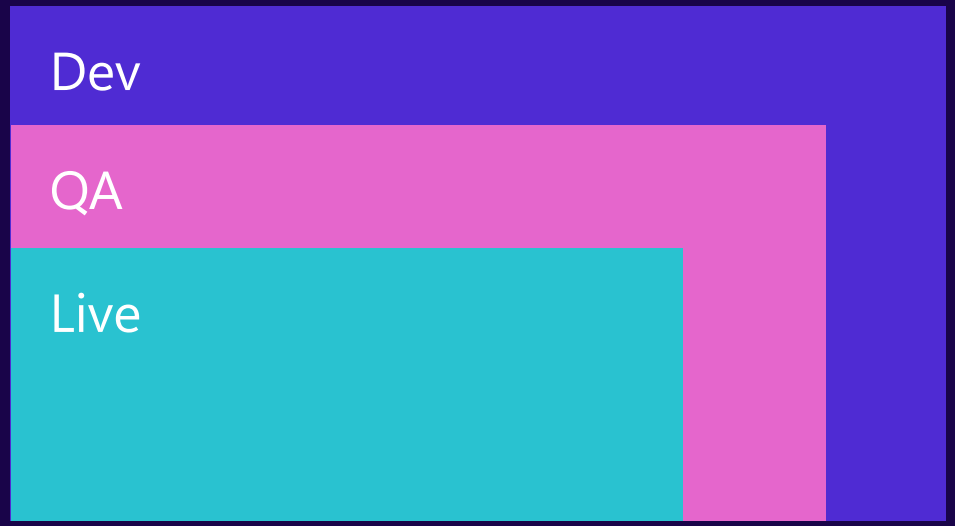
- Join
- SubQuery
- ...Raw 를 이용한 직접 입력

https://github.com/bluepope/SqlKataSample

5. 응용사례

목표 요구사항

- 부분적으로 다른 스키마를 가진 DB에 유연하게 접근할 수 있어야함
- EF의 DBContext 을 기본 연결로 하며 동일한 Transaction 을 가져야함
- Query 는 가능한 단순하게 작성할 수 있어야하며, 복잡한 Query 또한 작성이 편리해야함
- 연결된 ORM 객체 없이도 Join 이 쉬워야함
- 불필요하게 반복적인 코드를 줄일 수 있어야하며, 필요에 따라 유연하게 변경할 수 있어야함
- 생성된 Query 에 대한 예측이 쉬워야함



Thank you