УДК 004

Пагинация Keyset с помощью Spring

Еровлева Регина Викторовна
Приамурский государственный университет имени Шолом-Алейхема
Студент

Еровлев Павел Андреевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье будут представлены примеры реализации пагинации Keyset с помощью Spring boot. Работа будет происходить в среде программирования IntelijIdea с использованием языка программирования Java.
Ключевые слова: Java, Пагинация, Keyset

Keyset pagination with Spring

Erovleva Regina Viktorovna
Sholom-Aleichem Priamursky State University
Student

Erovlev Pavel Andreevich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article will provide examples of implementing Keyset pagination using Spring boot. The work will take place in the IntelijIdea programming environment using the Java programming language.
Keywords: Java, Pagination, Keyset

Системы реляционных баз данных уже давно предоставляют определенные способы ограничения набора результатов запроса, но они не обеспечивают хорошую производительность в большом объеме значений.

Цель работы – реализовать пагинацию с использованием Spring boot.

Исследованиями в данной теме занимались следующие авторы.

А.А.Симаков реализовал удобную трассировку JVM программ для обратного проектирования и анализа ПО [1]. В.П.Великов, К.С.Добрева обсудили в своей работе некоторые проблемы автоматизированной генерации ПО [2]. С.В.Мельников в своей статье описал принцип работы с отладочным интерфейсом Java [3]. В.А.Дашук, Д.Е.Намиот рассмотрели современные подходы работы с базами данных в Java проектах [4].

М.К.Ермаков и С.П.Вартанов рассмотрели при помощи динамического анализа подход к поиску состояний гонки в многопоточных программах на языке Java. [5].

При использовании Spring логика доступа к данным реализуется с помощью «Spring Data Repositories». Поэтому базовые методы доступа к данным определяются классом «JpaRepository», а пользовательская логика может быть абстрагирована в один или несколько пользовательских классов «Spring Data Repository» (рис.1).

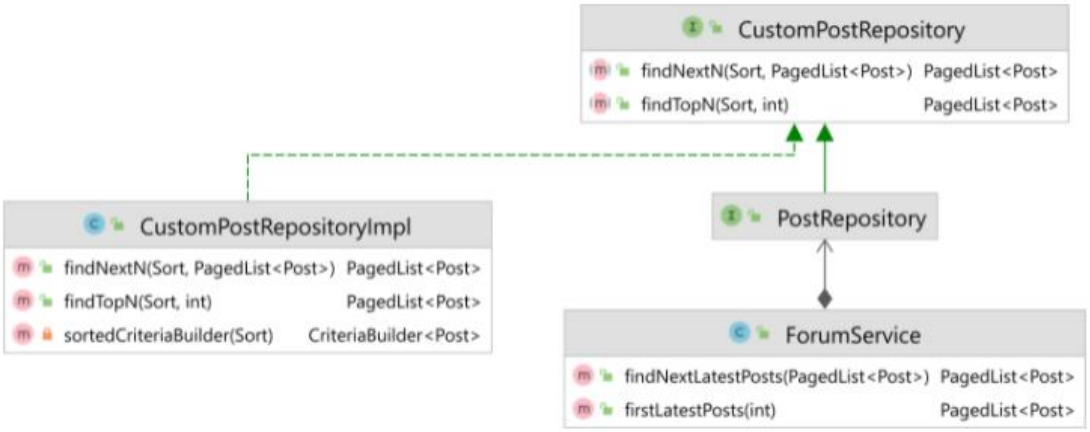

Рисунок 1 – Модель классов JPA

Создадим объект доступа к данным (рис.2).

```
@Repository
public interface PostRepository
        extends JpaRepository<Post, Long>, CustomPostRepository {
}
```

Рисунок 2 – Объект доступа

Если необходимо предоставить дополнительные методы доступа к данным, то можно сделать расширение «PostRepository». «CustomPostRepository» - здесь будет определена пользовательская логика доступа к данным (рис.3).

```
public interface CustomPostRepository {

    PagedList<Post> findTopN(
        Sort sortBy,
        int pageSize
    );

    PagedList<Post> findNextN(
        Sort orderBy,
        PagedList<Post> previousPage
    );
}
```

Рисунок 3 – Собственный интерфейс досупа

«CustomPostRepositoryImpl» класс, реализует «CustomPostRepository» интерфейс (рис.4).

```
public class CustomPostRepositoryImpl
        implements CustomPostRepository {

    @PersistenceContext
    private EntityManager entityManager;

    @Autowired
    private CriteriaBuilderFactory criteriaBuilderFactory;

    @Override
    public PagedList<Post> findTopN(
            Sort sortBy,
            int pageSize) {
        return sortedCriteriaBuilder(sortBy)
            .page(0, pageSize)
            .withKeysetExtraction(true)
            .getResultList();
    }

    @Override
    public PagedList<Post> findNextN(
            Sort sortBy,
            PagedList<Post> previousPage) {
        return sortedCriteriaBuilder(sortBy)
            .page(
                previousPage.getKeysetPage(),
                previousPage.getPage() * previousPage.getMaxResults(),
                previousPage.getMaxResults()
            )
            .getResultList();
    }

    private CriteriaBuilder<Post> sortedCriteriaBuilder(
            Sort sortBy) {
        CriteriaBuilder<Post> criteriaBuilder = criteriaBuilderFactory
            .create(entityManager, Post.class);

        sortBy.forEach(order -> {
            criteriaBuilder.orderBy(
                order.getProperty(),
                order.isAscending()
            );
        });

        return criteriaBuilder;
    }
```

Рисунок 4 – Реализация класса доступа

Теперь реализуем метод «ForumServiceKeySet» с использованием «PostRepository» (рис.5).

```
@Service
@Transactional(readOnly = true)
public class ForumService {

    @Autowired
    private PostRepository postRepository;

    public PagedList<Post> firstLatestPosts(
            int pageSize) {
        return postRepository.findTopN(
            Sort.by(
                Post_.CREATED_ON
            ).descending().and(
                Sort.by(
                    Post_.ID
                ).descending()
            ),
            pageSize
        );
    }

    public PagedList<Post> findNextLatestPosts(
            PagedList<Post> previousPage) {
        return postRepository.findNextN(
            Sort.by(
                Post_.CREATED_ON
            ).descending().and(
                Sort.by(
                    Post_.ID
                ).descending()
            ),
            previousPage
        );
    }
}
```

Рисунок 5 – Реализация метода KeySet

Теперь для того, чтобы вывести на первую страницу 25 записей создадим передачу данных на html форму (рис.6).

```
PagedList<Post> topPage = forumService.firstLatestPosts(PAGE_SIZE);

assertEquals(POST_COUNT, topPage.getTotalSize());

assertEquals(POST_COUNT / PAGE_SIZE, topPage.getTotalPages());

assertEquals(1, topPage.getPage());

List<Long> topIds = topPage.stream().map(Post::getId).toList();

assertEquals(Long.valueOf(50), topIds.get(0));
assertEquals(Long.valueOf(49), topIds.get(1));
```

Рисунок 6 – Вывод первых 25 значений

Данный код преобразуется в SQL запрос и выглядит он как показано на рисунке 7.

```
SELECT
    p.id AS col_0_0_,
    p.created_on AS col_1_0_,
    p.id AS col_2_0_,
    (
        SELECT count(*)
        FROM post post1_
    ) AS col_3_0_,
    p.id AS id1_0_,
    p.created_on AS created_2_0_,
    p.title AS title3_0_
FROM
    post p
ORDER BY
    p.created_on DESC,
    p.id DESC
LIMIT 25
```

Рисунок 7 – SQL запрос

Далее напишем код для вывода на вторую и последующие страницы (рис.8).

```
PagedList<Post> nextPage = forumService.findNextLatestPosts(topPage);

assertEquals(2, nextPage.getPage());

List<Long> nextIds = nextPage.stream().map(Post::getId).toList();

assertEquals(Long.valueOf(25), nextIds.get(0));
assertEquals(Long.valueOf(24), nextIds.get(1));
```

Рисунок 8 – Вывод последующих страниц

Данный код так же будет переведен на SQL запрос (рис.9).

```
SELECT
    p.id AS col_0_0_,
    p.created_on AS col_1_0_,
    p.id AS col_2_0_,
    (
        SELECT count(*)
        FROM post post1_
    ) AS col_3_0_,
    p.id AS id1_0_,
    p.created_on AS created_2_0_,
    p.title AS title3_0_
FROM
    post p
WHERE
    (p.created_on, p.id) <
    ('2021-12-30 12:26:00.0', 26) AND 0=0
ORDER BY
    p.created_on DESC,
    p.id DESC
LIMIT 25
```

Рисунок 9 – SQL запрос

Пагинация «Keyset» очень полезна при реализации решения с большим объемом данных и хотя в «Spring Data» нет встроенной поддержки для него, то в данной статье был рассмотрен пример как реализовать его

самостоятельно, используя «Blaze Persistence» и пользовательские репозитории данных «Spring».

Библиографический список

1. Симаков А.А. Java tracer. Программное средство для трассировки java программ // Заметки по информатике и математике. 2019. №3. С. 51-57.
2. Великов В.П., Добрева К.С. Генератор из диаграмм классов java в исходный код java // Информационные системы и технологии: управление и безопасность. 2014. №3. С. 14-23.
3. Мельников С.В. Обзор и применение отладочного интерфейса java (jdi) для обратимой модификации программных продуктов // Современные проблемы науки и образования. 2018. №8. С. 8-19.
4. Дашук В.А., Намиот Д.Е. Сравнительный анализ моделей работы с данными в java-приложениях // International journal of open information technologies. 2020. №7-1(51). С. 95-108.
5. Ермаков М.К. и Вартанов С.П. Подход к проведению динамического анализа java-программ методом модификации виртуальной машины java // Труды института системного программирования РАН. 2015. №2. С. 39-52.