

第四篇 分页查询

一、让Spring Boot-MybatisPlus支持分页

从Mybatis Plus 3.4.0版本开始，不再使用旧版本的
`PaginationInterceptor`，而是使用`MybatisPlusInterceptor`。

`MybatisPlusInterceptor`是一系列的实现`InnerInterceptor`的拦截器链，也可以理解为一个集合。可以包括如下的一些拦截器

- 自动分页: `PaginationInnerInterceptor`（最常用）
- 多租户: `TenantLineInnerInterceptor`
- 动态表名: `DynamicTableNameInnerInterceptor`
- 乐观锁: `OptimisticLockerInnerInterceptor`
- sql性能规范: `IllegalSQLInnerInterceptor`
- 防止全表更新与删除: `BlockAttackInnerInterceptor`

在已经集成了Mybatis Plus的SpringBoot项目中，在config包加入如下分页拦截器的配置，让MybatisPlus支持分页

```
@Configuration
@MapperScan(basePackages = {"top.mqxu.**.mapper"})
public class MybatisPlusConfig {

    /**
     * 新的分页插件，一缓和二缓遵循mybatis的规则，需要设置
     * MybatisConfiguration#useDeprecatedExecutor = false 避免缓存
     * 出现问题(该属性会在旧插件移除后一同移除)
     */
    @Bean
    public MybatisPlusInterceptor mybatisPlusInterceptor()
    {
        MybatisPlusInterceptor interceptor = new
        MybatisPlusInterceptor();
        //向Mybatis过滤器链中添加分页拦截器
```

```

        interceptor.addInnerInterceptor(new
        PaginationInnerInterceptor(DbType.MYSQL));
        //还可以添加其他的拦截器
        return interceptor;
    }

    @Bean
    public ConfigurationCustomizer
    configurationCustomizer() {
        return configuration ->
        configuration.setUseDeprecatedExecutor(false);
    }
}

```

二、单表查询分页-表格分页

```

@Test
public void testSelectByPage() {
    LambdaQueryWrapper<User> query = new
    LambdaQueryWrapper<>();
    query.ge(User::getAge, 10) //查询条件: 年龄大于10
        .orderByDesc(User::getAge); //按照年龄的倒序排序
    Page<User> page = new Page<>(1, 5); //查询第1页, 每页5条
    数据
    userMapper.selectPage(page, query); //page分页信息,
    query查询条件
    System.out.println("总页数: " + page.getPages());
    System.out.println("总记录数: " + page.getTotal());
    // 分页返回的对象与传入的对象是同一个
    List<User> list = page.getRecords();
    list.forEach(System.out::println);
}

```

查询输出结果如下:

```

⇒ Preparing: SELECT COUNT(*) FROM user WHERE (age ≥ ?)
⇒ Parameters: 10(Integer)
⇐ Columns: COUNT(*)
⇐ Row: 8
⇐ Total: 1
⇒ Preparing: SELECT id,name,age,email FROM user WHERE (age ≥ ?) ORDER BY age DESC LIMIT ?
⇒ Parameters: 10(Integer), 5(Long)
⇐ Columns: id, name, age, email
⇐ Row: 3, Tom, 28, test3@baomidou.com
⇐ Row: 5, Billie, 24, test5@baomidou.com
⇐ Row: 4, Sandy, 21, test4@baomidou.com
⇐ Row: 2, Jack, 20, test2@baomidou.com
⇐ Row: 1376805307556085762, test, 19, test@baomidou.com
⇐ Total: 5
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@5939e24]
总页数: 2
总记录数: 8
User(id=3, name=Tom, age=28, email=test3@baomidou.com)
User(id=5, name=Billie, age=24, email=test5@baomidou.com)
User(id=4, name=Sandy, age=21, email=test4@baomidou.com)
User(id=2, name=Jack, age=20, email=test2@baomidou.com)
User(id=1376805307556085762, name=test, age=19, email=test@baomidou.com)

```

在分页查询过程中，一共执行了两条SQL

```

# 第一条SQL用于查询在query条件下的总条数
SELECT COUNT(*) FROM user WHERE age >= ?

# 第二条SQL用于查询具体的数据
SELECT id,name,age,email
FROM user
WHERE age >= ?
ORDER BY age DESC
LIMIT ?

```

这种分页方式比较适合于传统应用中: 表格分页的开发（需要给出总条数，以及每页多少条）

用户名称	联系电话	用户邮箱	所在组织	用户状态	创建时间	操作
yanfa1		111@qq.com	上海分公司	已激活	2020-02-28 23:16:30	重置密码 分配角色
admin	13756823456	Xx@163.com	DongTech	已激活	2020-02-28 08:37:47	重置密码 分配角色

共 2 条 20条/页 < 1 > 前往 1 页

三、不查询总记录数的分页-下拉分页

在一些现代的网站或者APP，通常不会给出数据的总条数，而是通过鼠标或者手势每次下拉都加载n条数据。



这种情况下的分页通常就不需要查询总条数了，如果查询总条数浪费数据库的计算资源，使响应时间变长。

所以我们应该只做分页数据查询，不查询总条数。

设置page分页的第三个参数为false。

```

@Test
public void testSelectByPage() {
    LambdaQueryWrapper<User> query = new
LambdaQueryWrapper<>();
    query.ge(User::getAge, 10)           //查询条件: 年龄大于10
        .orderByDesc(User::getAge);     //按照年龄的倒序排序
    Page<User> page = new Page<>(1, 5, false); //查询第1页,
每页5条数据
    userMapper.selectPage(page, query);    //page分页信息,
query查询条件
    System.out.println("总页数: " + page.getPages());
    System.out.println("总记录数: " + page.getTotal());
    // 分页返回的对象与传入的对象是同一个
    List<User> list = page.getRecords();
    list.forEach(System.out::println);
}

```

输出结果总页数和总条数都是0，但是分页数据正常查询到了。

```

=> Preparing: SELECT id,name,age,email FROM user WHERE (age >= ?) ORDER BY age DESC LIMIT ?
=> Parameters: 10(Integer), 5(Long)
<= Columns: id, name, age, email
<= Row: 3, Tom, 28, test3@baomidou.com
<= Row: 5, Billie, 24, test5@baomidou.com
<= Row: 4, Sandy, 21, test4@baomidou.com
<= Row: 2, Jack, 20, test2@baomidou.com
<= Row: 1376805307556085762, test, 19, test@baomidou.com
<= Total: 5
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@34a6d9db]
总页数: 0
总记录数: 0
User(id=3, name=Tom, age=28, email=test3@baomidou.com)
User(id=5, name=Billie, age=24, email=test5@baomidou.com)
User(id=4, name=Sandy, age=21, email=test4@baomidou.com)
User(id=2, name=Jack, age=20, email=test2@baomidou.com)
User(id=1376805307556085762, name=test, age=19, email=test@baomidou.com)

```

只执行了这样一个SQL

```

SELECT id,name,age,email
FROM user
WHERE age >= ?
ORDER BY age DESC
LIMIT ?

```