Spring Data MongoDB

Spring Data MongoDB 是 Spring 框架访问 MongoDB 数据库的组件，使用它可以非常方便地操作 MongoDB 数据库，Spring Data 是 Spring 的一个子项目，旨在为关系型数据库，非关系型数据库提供统一的数据访问 API。

Spring Data 提供了一套基于 Repository 的统一接口 CrudRepository 完成对象的 CRUD 操作。

- 新建 Maven 工程，pom.xml

```xml
<dependencies>
    <dependency>
        <groupId>org.springframework.data</groupId>
        <artifactId>spring-data-mongodb</artifactId>
        <version>2.0.8.RELEASE</version>
    </dependency>

    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <version>1.18.6</version>
        <scope>provided</scope>
    </dependency>
</dependencies>
```

- 配置 spring.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:p="http://www.springframework.org/schema/p"
xmlns:mongo="http://www.springframework.org/schema/data/mongo"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.2.xsd
    http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.3.xsd
http://www.springframework.org/schema/data/mongo
http://www.springframework.org/schema/data/mongo/spring-mongo.xsd">

    <!-- Spring 连接 MongoDB 客户端配置 -->
    <mongo:mongo-client host="127.0.0.1" port="12345" id="mongo">
</mongo:mongo-client>

    <!-- 配置 MongoDB 目标数据库 -->
    <mongo:db-factory dbname="testdb" mongo-ref="mongo"></mongo:db-factory>
    <!-- 配置 MongoTemplate -->
    <bean id="mongoTemplate"
class="org.springframework.data.mongodb.core.MongoTemplate">
        <constructor-arg name="mongoDbFactory" ref="mongoDbFactory">
</constructor-arg>
    </bean>

</beans>
```

- 创建实体类 Student

```
package com.southwind.entity;

import lombok.Data;
import org.springframework.data.annotation.Id;
import org.springframework.data.mongodb.core.mapping.Document;
import org.springframework.data.mongodb.core.mapping.Field;

import java.util.Date;

@Data
@Document(collection = "student_info")
public class Student {
    @Id
    private String id;
    @Field
    private String name;
    @Field
    private int age;
    @Field(value = "a_time")
    private Date addTime;
}
```

- 测试类中调用 MongoTemplate 接口完成相关的业务逻辑操作。

```
package com.southwind.test;

import com.southwind.entity.Student;
import org.bson.types.ObjectId;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.data.mongodb.core.MongoTemplate;
import org.springframework.data.mongodb.core.query.Criteria;
import org.springframework.data.mongodb.core.query.Query;
import org.springframework.data.mongodb.core.query.Update;

import java.util.ArrayList;
import java.util.Date;
```

```java
import java.util.List;

public class Test {

    private static MongoTemplate mongoTemplate;
    static {
        ApplicationContext applicationContext = new
ClassPathXmlApplicationContext("spring.xml");
        mongoTemplate = (MongoTemplate)
applicationContext.getBean("mongoTemplate");
    }

    public static void main(String[] args) {
//        insert();
//        deleteByCondition();
//        deleteReturn();
//        deleteList();
//        deleteCollection();
//        dropDB();
//        update();
//        upsert();
//        findAll();
//        findByAge();
//        count();
//        findOne();
//        findByPage();
//        findByPage(3,5);
//        findById();
//        findIn();
//        findOr();
        findAnd();
    }

    /**
     * 添加
     */
    public static void insert(){
        List<Student> list = new ArrayList<Student>();
        for (int i = 0;i < 10; i++){
            Student student = new Student();
            student.setName("李四"+i);
            student.setAge(18);
            student.setAddTime(new Date());
            list.add(student);
        }
        mongoTemplate.insert(list,Student.class);
    }

    /**
```

```java
     * 条件删除
     */
    public static void deleteByCondition(){
        mongoTemplate.remove(Query.query(new Criteria("name").is("张三
0")),Student.class);
    }

    /**
     * 删除并返回结果
     */
    public static void deleteReturn(){
        Student student = mongoTemplate.findAndRemove(Query.query(new
Criteria("name").is("张三1")),Student.class);
        System.out.println(student);
    }

    /**
     * 删除多条并返回集合
     */
    public static void deleteList(){
        List<Student> list = mongoTemplate.findAllAndRemove(Query.query(new
Criteria("age").is(18)),Student.class);
        for(Student student:list){
            System.out.println(student);
        }
    }

    /**
     * 删除集合
     */
    public static void deleteCollection(){
//        mongoTemplate.dropCollection(Student.class);
        mongoTemplate.dropCollection("student");
    }

    /**
     * 删除数据库
     */
    public static void dropDB(){
        mongoTemplate.getDb().drop();
    }

    /**
     * 修改
     */
    public static void update(){
//        mongoTemplate.updateFirst(Query.query(new Criteria("name").is("张三
0")), Update.update("age",33),Student.class);
```

```
        mongoTemplate.updateFirst(Query.query(new Criteria("name").is("张三
1")),Update.update("a_time",new Date()).update("age",22),Student.class);
    }

    /**
     * upsert
     */
    public static void upsert(){
        mongoTemplate.upsert(Query.query(new Criteria("name").is("李
四")),Update.update("age",26),Student.class);
    }

    /**
     * 查询
     */
    public static void findAll(){
        List<Student> list = mongoTemplate.findAll(Student.class);
        for(Student student:list){
            System.out.println(student);
        }
    }

    /**
     * 按条件查询
     */
    public static void findByAge(){
        List<Student> list = mongoTemplate.find(Query.query(new
Criteria("age").is(18)),Student.class);
        for(Student student:list){
            System.out.println(student);
        }
    }

    /**
     * 查询满足条件的记录数
     */
    public static void count(){
        long count = mongoTemplate.count(Query.query(new
Criteria("age").is(18)),Student.class);
        System.out.println(count);
    }

    /**
     * 查询满足条件的第一条记录
     */
    public static void findOne(){
        Student student = mongoTemplate.findOne(Query.query(new
Criteria("age").is(18)),Student.class);
        System.out.println(student);
    }

    /**
     * 分页查询
     */
    public static void findByPage(){
        List<Student> list = mongoTemplate.find(Query.query(new
Criteria("age").is(18)).limit(2),Student.class);
        for(Student student:list){
            System.out.println(student);
        }
    }

    public static void findByPage(int page,int limit){
        List<Student> list = mongoTemplate.find(Query.query(new
Criteria("age").is(18)).skip((page-1)*limit).limit(limit),Student.class);
        for (Student student:list){
            System.out.println(student);
        }
    }

    /**
     * 根据ID查询
     */
    public static void findById(){
        Student student = mongoTemplate.findById(new
ObjectId("5c7d294669beb402bb3df852"),Student.class);
        System.out.println(student);
    }

    /**
     * in查询
     */
    public static void findIn(){
        List<Student> list = mongoTemplate.find(Query.query(new
Criteria("age").in(18,22)),Student.class);
        for(Student student:list){
            System.out.println(student);
        }
    }

    /**
     * or查询
     */
    public static void findOr(){
        List<Student> list = mongoTemplate.find(Query.query(new
Criteria().orOperator(new Criteria("age").is(18),new
Criteria("age").is(22))),Student.class);
        for (Student student:list){
                System.out.println(student);
            }
        }

        /**
         * and查询
         */
        public static void findAnd(){
            Student student = mongoTemplate.findOne(Query.query(new
Criteria().andOperator(new Criteria("age").is(18),new Criteria("name").is("张
三3"))),Student.class);
            System.out.println(student);
        }
}
```

使用 Repository

- 自定义 Repository，继承 CrudRepository。

```
package com.southwind.repository;

import com.southwind.entity.Student;
import org.springframework.data.repository.CrudRepository;

public interface StudentRepository extends CrudRepository<Student,String> {
    public Student queryById(String id);
}
```

- spring.xml 配置自动扫描，让 IoC 容器管理 Repository 接口。

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:p="http://www.springframework.org/schema/p"
xmlns:mongo="http://www.springframework.org/schema/data/mongo"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.2.xsd
    http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.3.xsd
http://www.springframework.org/schema/data/mongo
http://www.springframework.org/schema/data/mongo/spring-mongo.xsd">

    <!-- Spring 连接 MongoDB 客户端配置 -->
    <mongo:mongo-client host="127.0.0.1" port="12345" id="mongo">
</mongo:mongo-client>
```

```
    <!-- 配置 MongoDB 目标数据库 -->
    <mongo:db-factory dbname="testdb" mongo-ref="mongo"></mongo:db-factory>

    <!-- 配置 MongoTemplate -->
    <bean id="mongoTemplate"
class="org.springframework.data.mongodb.core.MongoTemplate">
        <constructor-arg name="mongoDbFactory" ref="mongoDbFactory">
</constructor-arg>
    </bean>

    <!-- 扫描 Repository 接口 -->
    <mongo:repositories base-package="com.southwind.repository">
</mongo:repositories>

</beans>
```

- 从 IoC 容器中获取 StudentRepository 实例，调用其方法完成相关的业务。

```
package com.southwind.test;

import com.southwind.entity.Student;
import com.southwind.repository.StudentRepository;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class RepsotoryTest {
    private static StudentRepository studentRepository;
    static {
        ApplicationContext applicationContext = new
ClassPathXmlApplicationContext("spring.xml");
        studentRepository =
(StudentRepository)applicationContext.getBean("studentRepository");
    }

    public static void main(String[] args) {
        Student student =
studentRepository.queryById("5c7d255469beb40295197f29");
        System.out.println(student);
    }
}
```