# Spring Boot核心-Spring Data JPA

JPA 郎Java Persistence API。JPA 是一个基于O/R映射的标准规范（目前最新版本是JPA 2.1 ）。所谓规范即只定义标准规则（如注解、接口），不提供实现，软件提供商可以按照标准规范来实现，而使用者只需按照规范中定义的方式来使用，而不用和软件提供商的实现打交道。JPA 的主要实现由Hibernate、EclipseLink 和OpenJPA 等，这也意味着我们只要使用JPA 来开发，无论是哪一个开发方式都是一样的。

Spring Data JPA 是Spring Data 的一个子项目，它通过提供基于JPA 的Repository 极大地减少了JPA 作为数据访问方案的代码量。

## 定义数据访问层

使用Spring Data JPA 建立数据访问层十分简单，只需定义一个继承JpaRepository 的接口即可，定义如下：

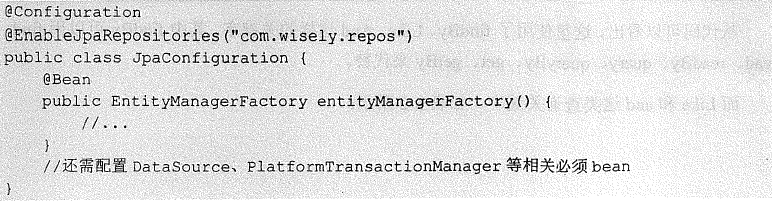
|  |
| --- |
| **public interface** PersonRepository **extends** JpaRepository<Person, Long>{} |

继承JpaRepository 接口意味着我们默认已经有了下面的数据访问操作方法：

|  |
| --- |
| @NoRepositoryBean **public interface** JpaRepository<T, ID **extends** Serializable>  **extends** PagingAndSortingRepository<T, ID>, QueryByExampleExecutor<T> { List<T> findAll(); List<T> findAll(Sort sort); List<T> findAll(Iterable<ID> ids); <S **extends** T> List<S> save(Iterable<S> entities); **void** flush(); <S **extends** T> S saveAndFlush(S entity); **void** deleteInBatch(Iterable<T> entities); **void** deleteAllInBatch(); T getOne(ID id); @Override  <S **extends** T> List<S> findAll(Example<S> example);@Override  <S **extends** T> List<S> findAll(Example<S> example, Sort sort); } |

## 配置使用Spring Data JPA

在Spring 环境中，使用Spring Data JPA 可通过＠EnableJpaRepositories 注解来开启SpringDataJPA 的支持，@EnableJpaRepositories 接收的value 参数用来扫描数据访问层所在包下的数据访问的接口定义。



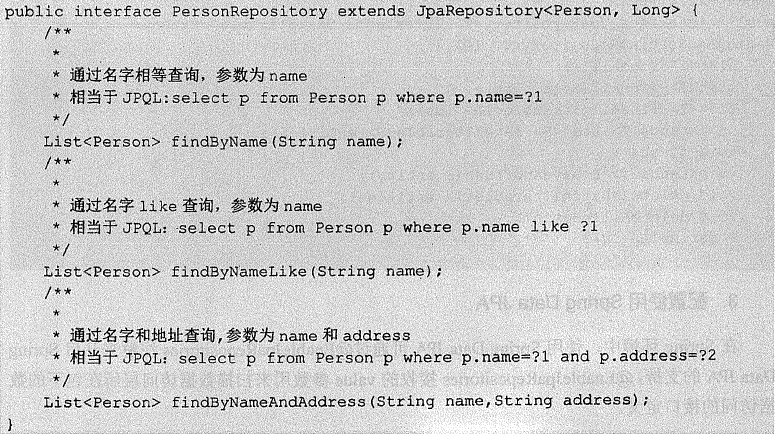
## 定义查询方法

在讲解查询方法前，假设我们有一张数据表名叫PERSON ，有ID (Number)、NAME(Varchar 2)、AGE (Number)、ADDRESS(Varchar2)几个字段；对应的实体类叫Person ，分别有id (Long)、name (String)、age (Integer)、address (String)。

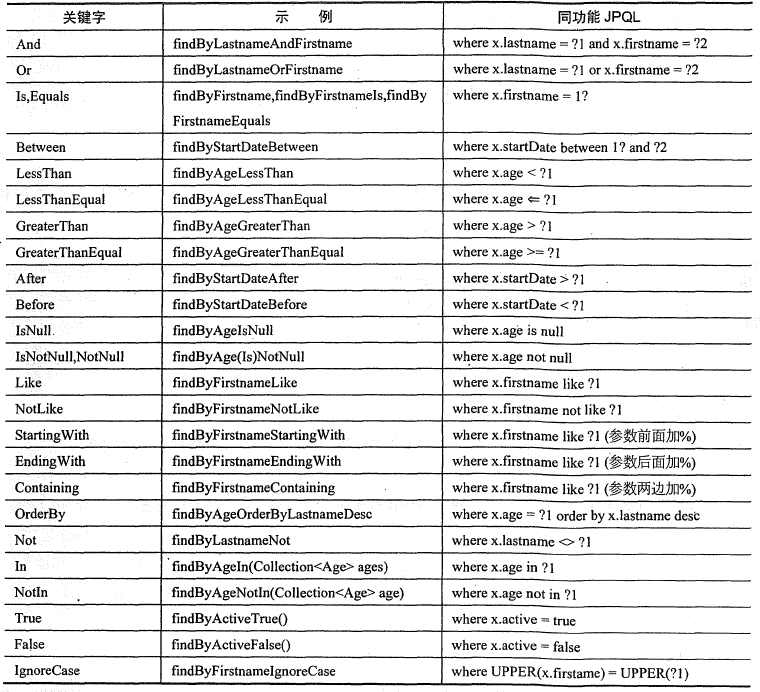
( 1 )根据属性名查询

Spring Data JPA 支持通过定义在Repository 接口中的方法名来定义查询，而方法名是根据实体类的属性名来确定的。

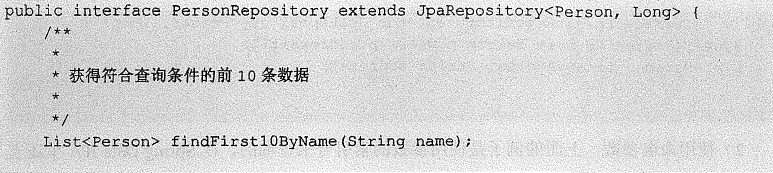
1. 常规查询。根据属性名来定义查询方法，示例如下：

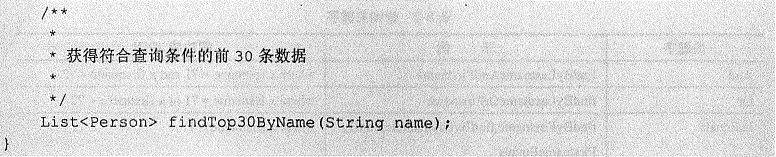


从代码可以看出，这里使用了findBy 、Like 、And 这样的关键字。其中findBy 可以用find 、read、readBy 、query 、queryBy 、get、getBy 来代替。而Like 和and 这类查询关键字，如表所示：



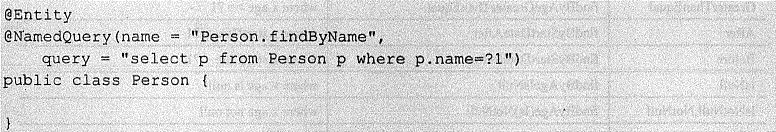
2 ）限制结果数量。结果数量是用top 和first 关键字来实现的。

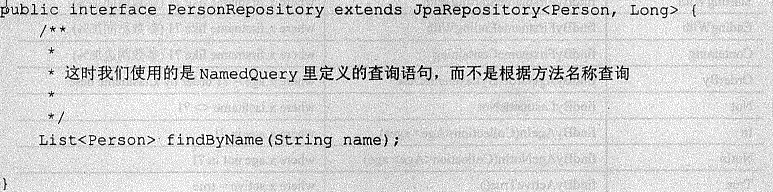




(2 ）使用JPA 的 NamedQuery 查询

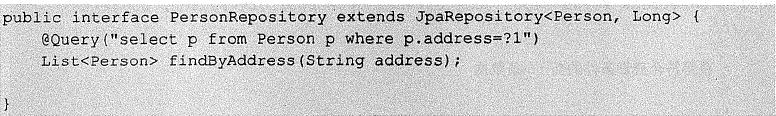
Spring Data JPA 支持用 JPA 的NameQuery 来定义查询方法，即一个名称映射一个查询语句。定义如下：



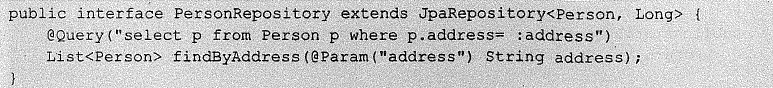


（3）使用@Query查询

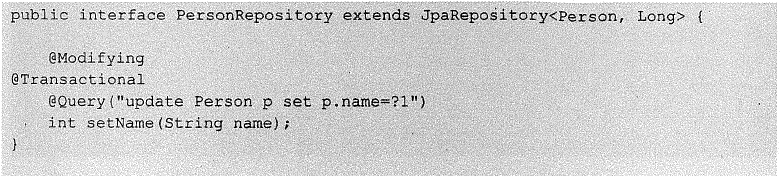
1 ）使用参数索引。Spring Data JPA 还支持用＠Query 注解在接口的方法上实现查询，例如：



2 ）使用命名参数。上面的例子是使用参数的索引号来查询的，在Spring Data JPA 里还支持在语句里用名称来匹配查询参数。



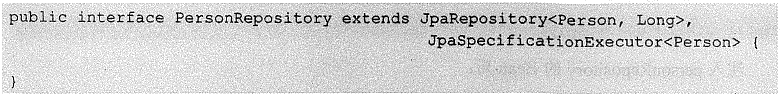
3 ）更新查询。Spring Data JPA 支持@Modifying 和@Query 注解组合来实现更新查询，其中返回值int 表示更新语句影响的行数



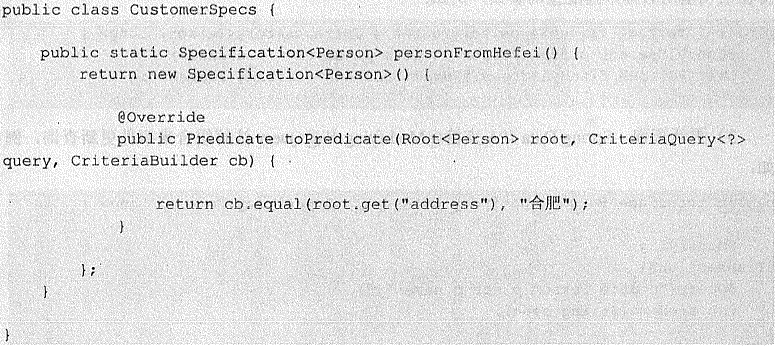
(4) Specification

JPA 提供了基于准则查询的方式，即Criteria 查询。而Spring Data JPA 提供了一个Specification （规范）接口让我们可以更方便地构造准则查询， Specification 接口定义了一个toPredicate 方法用来构造查询条件。

1. 定义。我们的接口类必需实现JpaSpecificationExecutor 接口，代码如下：



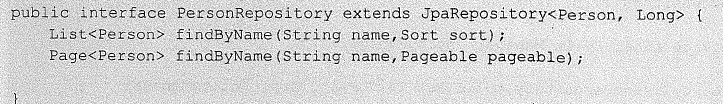
然后需要定义Criterial查询，代码如下



CriteriaBuilder 、CriteriaQuery ,. Predicate 、Root 都是来自JPA 的接口。CriteriaBuilder 包含的条件构造有： exists 、and 、or, not 、conjunction 、disjunction 、isTrue 、isFalse 、isNull 、isNotNull 、equal 、notEqual 、greaterThan 、greaterThanOrEqualTo 、lessThan 、lessThanOrEqualTo 、between 等，详细请查看CriteriaBuilder 的API。

(5)排序与分页

Spring Data JPA 充分考虑了在实际开发中所必需的排序和分页的场景，为我们提供了Sort类以及Page 接口和Pageable 接口。

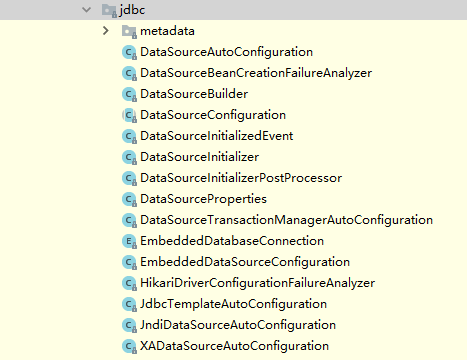


## 自定义Repository 的实现

Spring Data 提供了和CrudRepository 、PagingAndSortingRepository; Spring Data JPA 也提供了JpaRepository。我们也可以自定义Repository 的实现

## Spring Boot 的JPA支持

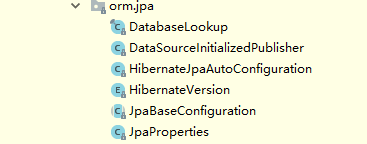
Spring boot-starter-data-jpa 依赖于spring-boot-starter－jdbc ，而Spring Boot 对JDBC做了一些白动配置。源码放置在org.springframework. boot.autoconfigure.jdbc。



从源码分析可以看出，我们通过spring.datasource 为前缀的属性自动配置dataSource。Spring Boot 自动开启了注解事务的支持（ @EnableTransactionManagement ）；还配置了一个jdbcTemplate 。Spring Boot 还提供了一个初始化数据的功能：放置在类路径下的schema.sql文件会自动用来初始化表结构；放置在类路径下的data.sql 文件会在动用来填充表数据。

## 对JPA 的自动配置

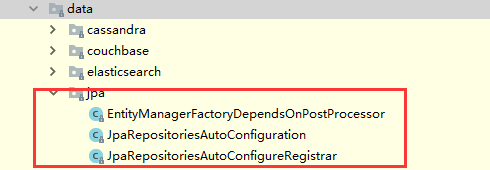
Spring Boot 对JPA 的自动配置放置在org.springframework. boot.autocon figure. orm.jpa 下



从HibemateJpaAutoConfiguration 可以看出， Spring Boot 默认JPA 的实现者是Hibernate，HibemateJpaAutoCorifiguration 依赖于DataSourceAutoConfiguration 。从JpaProperties 的源码可以看出，配置JPA 可以使用spring.jpa 为前缀的属性在. application.properties 中配置。从JpaBaseConfiguration 的源码中可以看出， Spring Boot 为我们配置了transactionManager、jpaVendorAdapter、entityManagerFactory 等Bean。JpaBaseConfiguration 还有一个getPackagesToScan方法，可以自动扫描注解有＠Entity 的实体类。在Web 项目中我们经常会遇到在控制器或者页面访问数据的时候出现会话连接己关闭的错误，这时候我们会配置一个Open Entity Manager (Session) In View 这个过滤器。令人惊喜的是， Spring Boot 为我们自动配置了OpenEntityManagerlnViewInterceptor 这个Bean ，并注册SpringMVC 的拦截器中。

## 对Spring Data JPA 的自动配置

Spring Boot 对Spring Data JPA 的自动配置放置在org.springframework.boot.autoconfigure.data.jpa下。



从JpaRepositoriesAutoConfiguration 和JpaRepositoriesAutoConfigureRegistrar 源码可以看出， JpaRepositoriesAutoConfiguration 是依赖于HibernateJpaAutoConfiguration 配置的，且Spring Boot 自动开启了对Spring Data JPA 的支持，即我们无须在配置类中显示声明@EnableJpaRepositories。

通过上面的分析可知，我们在Spring Boot 下使用Spring Data JPA ，在项目的Maven 依赖里添加spring-boot-stater-data-jpa，然后只需定义DataSource、实体类和数据访问烂，并在且在需要使用数据访问的地方注入数据访问层的Bean 即可，无须任何额外自己置。