目录

[简介 1](#_Toc9616)

[什么是EJPA 1](#_Toc24387)

[GIT开源地址 2](#_Toc19448)

[入门 2](#_Toc30811)

[jar包引入 2](#_Toc7132)

[application.properties配置 3](#_Toc30412)

[Sql查询 3](#_Toc5967)

[1.使用JpaRepository接口查询 3](#_Toc23745)

[2.使用Class类提供接口查询 4](#_Toc9899)

[1. 创建ClassJpa查询文件 4](#_Toc3223)

[2. 2.ModelQuery参数说明 4](#_Toc6276)

[Update数据 5](#_Toc24987)

[使用方法 5](#_Toc3566)

[例子 5](#_Toc18336)

[与save的区别 6](#_Toc7735)

[分布式ID 6](#_Toc4192)

[使用方法1 7](#_Toc5451)

[使用方法2 8](#_Toc31941)

[1.引入生成ID的bean 8](#_Toc2648)

[2.利用映射类获取ID 8](#_Toc24967)

[分布式锁 8](#_Toc14447)

[1. 开启分布式锁的功能 8](#_Toc18893)

[2. 引入分布式锁的bean 8](#_Toc4392)

[3. 加锁方法 8](#_Toc20392)

[4. 解锁方法 9](#_Toc420)

[慢查下记录 9](#_Toc25000)

[1.慢查询配置 9](#_Toc25696)

[2.查下接口 9](#_Toc9694)

# 1.简介

### 什么是EJPA

EJPA是Spring data Jpa 一个增强工具，它解决了JPA无法自定义结果映射问题，没有动态sql，原生sql分页查询复杂，save会update Null字段等问题。它还提供了分布式ID生成策略，慢查询记录，分布式锁等功能。

### GIT开源地址

<https://gitee.com/hongzihao/SpringEnhanceJPA>

# 2.入门

### jar包引入

//由于目前还没有正式发布，还必须私用sonatype仓库

//或者在 《jar包下载》 ejpa-x.x.x.jar下载到本地引入项目

<repositories>

<repository>

    <id>snapshots</id>

    <name>snapshots</name>

    <url>https://oss.sonatype.org/content/repositories/snapshots/</url>

    <snapshots>

        <enabled>true</enabled>

    </snapshots>

</repository>

</repositories>

<dependency>

<groupId>io.gitee.hongzihao</groupId>

<artifactId>ejpa</artifactId>

<version>0.0.1-SNAPSHOT</version>

</dependency>

//插件依赖包，这插件有依赖到redis所以也要配置redis

<dependency>

<groupId>cn.hutool</groupId>

<artifactId>hutool-all</artifactId>

<version>4.1.15</version>

</dependency>

<dependency>

<groupId>commons-beanutils</groupId>

<artifactId>commons-beanutils</artifactId>

<version>1.9.3</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-redis</artifactId>

<version>1.5.4.RELEASE</version>

</dependency>

# 3.application.properties配置

#是否记录慢查询  
ejpa.log-slow-queries=true  
#记录最近几天的慢查询 默认5  
ejpa.slow-log-days=5  
#慢查询超过查询多少秒记录 默认3  
ejpa.log-slow-seconds=3  
#分布式ID本地JVM一次取多少条，默认1  
ejpa.identity-size=5  
#是否开启分布式锁,默认false  
ejpa.open-lock=false

# 4.Sql查询

### 1.使用JpaRepository接口查询

* 1.创建interface查询文件
* 2.使用JpaRepository可以不用设置retrunClass
* 3.没有注解ModelQuery的方法不会走定义对象查询

public interface VoteWorkJpa extends JpaRepository<VoteWork, String> {

@ModelQuery(value = " select v.\*,(select count(l.id) from vote\_log l where l.vote\_work\_id = v.id )as vote from vote\_work v where v.vote\_activitie\_id = :voteActivitieId and v.is\_delete is null ",nativeQuery = true)  
List<VoteWork> findByVoteActivitieIdAndIsDeleteIsNull(@Param("voteActivitieId") String voteActivitieId);

｝

这样VoteWork里面的

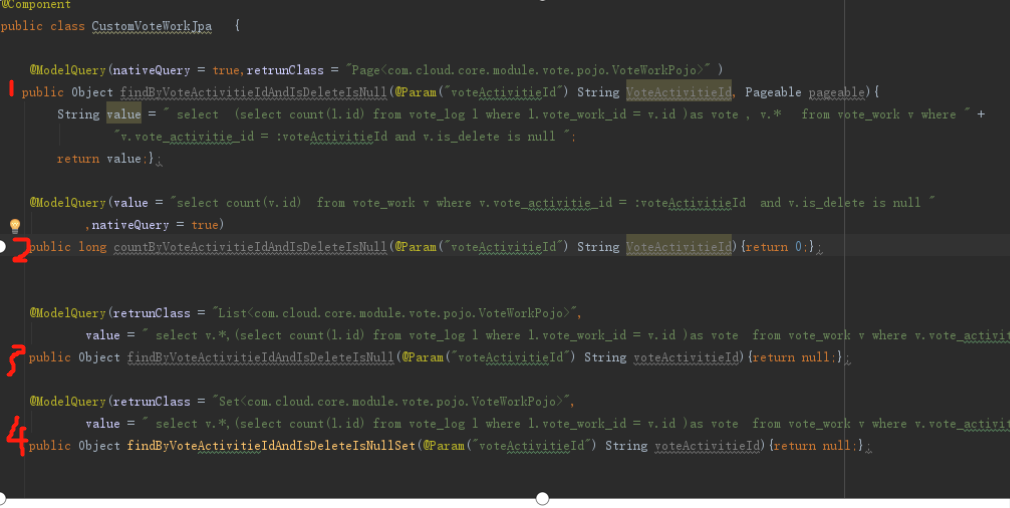
@Transient  
private long vote;

会被注入数据

### 2.使用Class类提供接口查询

1. 创建ClassJpa查询文件
2. 2.ModelQuery参数说明

* retrunClass = 添加返回值的类型，见下图findByVoteActivitieIdAndIsDeleteIsNull 第一个方法
* value = 查询语句，或者方法return的值2选一也是查询语句（如2个都写value 优先与return） 见3，4个方法
* countColumn = 分页指定count的字段，默认是\*号，如sql select id from xxx 默认会自动使用 id。只用使用Page返回值才会执行count
* countQuery = 自己写count的sql与 countColumn 2选一，countQuery 优先与countColumn
* nativeQuery =是否是sql =true 或者 jsql = false ，默认false
* （注：使用方法的return可以自定义平拼接sql）
* （注：使用long，Long等基本类型不需要指定retrunClass见下图方法2）



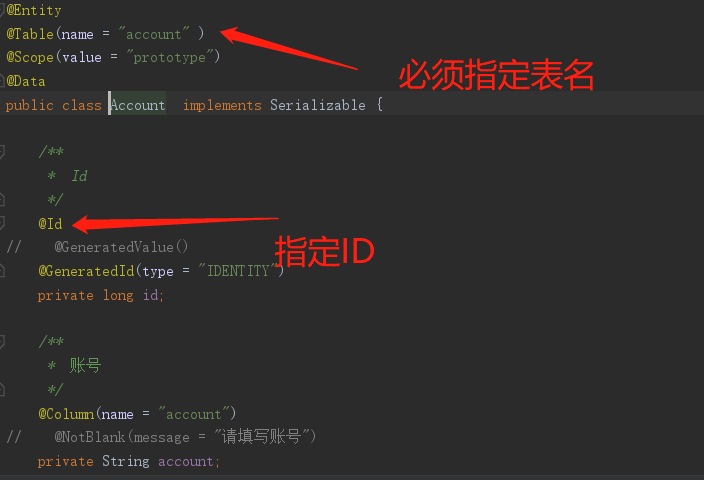
# 5.Update数据

### 使用方法

interface 集成 EjpaRepository

public interface AccountJpa extends JpaRepository<Account, String> , EjpaRepository<Account, String> {  
}

### 例子





updateRecord =1，因为第一条没有指定ID

### 与save的区别

1. null字段不会去更新数据库
2. 只做update 不会去insert数据

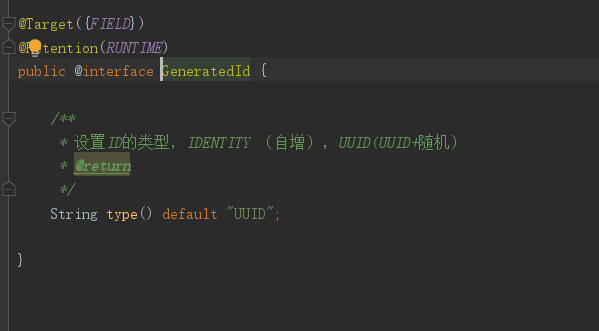
# 6.分布式ID

EJPA提供2种分布式ID，

一种是long类型利用redis的increment分配 id，

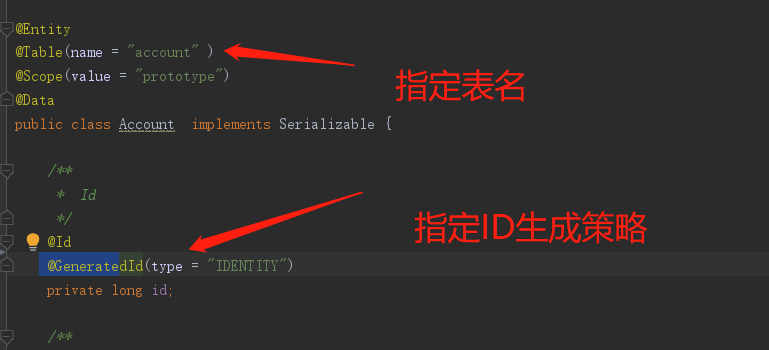
一种是Sring类型的uuid

利用注解GeneratedId，它提供了2种ID生成策略（注意不能与@GeneratedValue @GenericGenerato一起使用)



### 使用方法1

1.创建实体映射类



1. 然后直接调用JpaRepository 的save或者saveAll 方法保存数据就可以了。

如：

Account account2 =new Account();  
 account2.setAccount("update2");  
 account2.setPassworasdd(null);  
 account2.setDoubleTest(1.23);  
 account2.setIntTest(122212);  
 accounts.add(account2);  
accountJpa.save(account2);

### 使用方法2

1.引入生成ID的bean

@Autowired  
private GenerateIdUtil generateIdUtil;

2.利用映射类获取ID

String id = generateIdUtil.getId(Account.class));

然后自己再手动写insert语句

自增类型可以配每次从redis取到本地的Id数量

#分布式ID本地JVM一次取多少条，默认1  
ejpa.identity-size=5

# 分布式锁

系统第一次启动会创建distributed\_lock表，sql脚本文件放在项目resource/sql

### 开启分布式锁的功能

#是否开启分布式锁,默认false  
ejpa.open-lock=false

### 引入分布式锁的bean

@Autowired  
private MysqlService mysqlService;

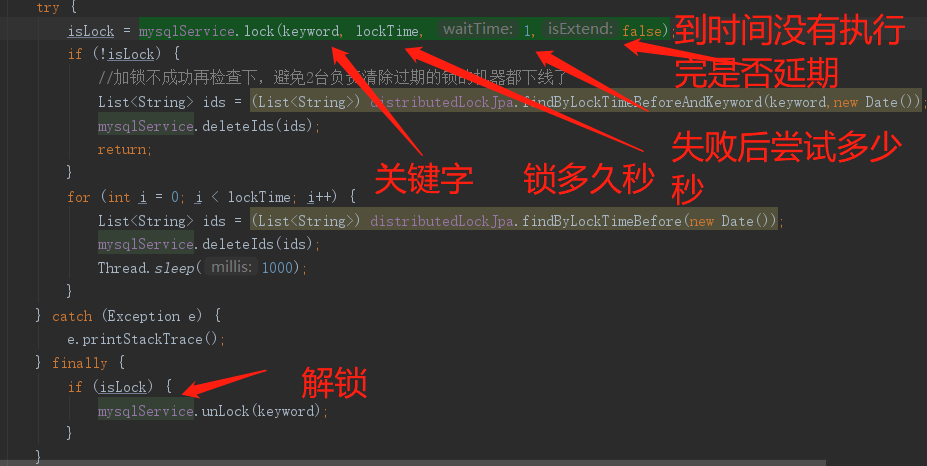
### 加锁方法

*/\*\*  
 \* mysql加锁  
 \** ***@param*** *key  
 \** ***@param*** *lockTime 锁时间 秒  
 \** ***@param*** *waitTime 等待时间 秒 0不等待  
 \** ***@param*** *isExtend 是否自动延期  
 \** ***@return*** *\*/*boolean lock(String key, int lockTime, int waitTime,boolean isExtend);

### 解锁方法

*/\*\*  
 \* 解锁  
 \** ***@param*** *key  
 \** ***@return*** *\*/*boolean unLock(String key);

例如



# 慢查下记录

### 1.慢查询配置

是否记录慢查询  
ejpa.log-slow-queries=true  
#记录最近几天的慢查询 默认5  
ejpa.slow-log-days=5  
#慢查询超过多少秒记录 默认3  
ejpa.log-slow-seconds=3

系统第一次启动会创建ejpa\_slow\_query

表，sql脚本文件放在项目resource/sql

### 2.查下接口

通过关键字查慢查询语句

<http://localhost:8080/ejpa/getSlowSql?keyword=select>

通过方法查询慢查询

<http://localhost:8080/ejpa/getSlowSqlByMethod?method=find>