JPA教程（基础API、Spring整合、Springboot整合）

JPA springdata 整合jpa 根据类创建表 无则建有则更 接口增删改查操作 类比springdata整合mongdb和es等

实例代码和资料

一、常见注解

注解 只有在spring或springboot环境下才可以在属性上加注解，否则强制getXX方法上加

```
@Entity          标志实体
@Table           别名，默认为实体名
@Id              必不可少 否则报错 指定主键
@GeneratedValue（strategy，generator） mysql支持identity、oracle支持sequence、auto默认自动选择、table通过表生成
@Column 标注的 columnDefinition 属性：表示该字段在数据库中的实际类型 Date属性无法自动，String默认对应varchar
@Transient       忽略映射      自定义的getXX方法加上此注解  Spring环境下如果找不到对应属性会自动忽略
@Basic       表示一个简单的属性到数据库表的字段的映射,对于没有任何标注的 getXxxx() 方法,默认即为@Basic
@Basic(fetch = FetchType.LAZY,optional = false) 懒加载、不允许为空
@Temporal(TemporalType.TIMESTAMP) 指定日期在数据库的类型 如date--年月日，timestamp年月日时分秒
```

映射关系相关注解 注意区分单向和双向映射

```
@JoinColumn(name="CUSTOMER_ID") 外键方
@ManyToOne(fetch=FetchType.LAZY) mappedby不和上面注解一起用  默认EAGER左外连接，懒加载获取时才查两次
@ManyToMany
@OneToOne
```

二、实体的状态:

- 新建状态：新创建的对象，尚未拥有持久性主键。 new没ID
- 游离状态：拥有持久化主键，但是没有与持久化建立上下文环境new 有ID
- 持久化状态：已经拥有持久性主键并和持久化建立了上下文环境
- 删除状态：拥有持久化主键，已经和持久化建立上下文环境，但是从数据库中删除

三、使用JPA持久化对象的步骤

1,创建 persistence.xml, 在这个文件中配置持久化单元

```
需要指定跟哪个数据库进行交互;
需要指定 JPA 使用哪个持久化的框架以及配置该框架的基本属性
```

2,创建实体类, 使用 annotation 来描述实体类跟数据库表之间的映射关系.

```
使用 JPA API 完成数据增加、删除、修改和查询操作
```

3,创建 EntityManagerFactory可指定属性如数据表替换或更新、sql的打印; 4,创建 EntityManager (对应 Hibernate 中的Session)，也可指定属性 5,开启事务，持久化操作，关闭事务

四、JPA常用API

- 1，创建EntityManagerFactory，两个重载构造
- 2，创建管理器EntityManage，两个重载构造
- 3，事务EntityTransaction：begin commit rollback isActive
- 4，EntityManager的基本操作

```
find/getReference    前者不存在null，后者存在ok，不存在报错    用到对象才调用select语句查询
persist不能有id，更新可先查最后提交事务 或 再次调用方法
remove 只能操作持久化对象，游离对象不可
merge 临时对象没ID创建--复制属性新对象--insert          3条语句（主键查更+insert）
      游离对象有ID创建--缓存存在--复制属性新对象--update   2条sql   1条查数据库，缓存了，1条更
      游离对象有ID创建--缓存不存在--数据库存在select--update      2条（先select数据库，再
update）
      游离对象有ID创建--缓存不存在--数据库不存在select--复制属性新对象--insert  4条（先select数
    据库，主键查更+insert）
flush    同步上下文环境，将未保存实体同步数据库，两种模式 auto、commit
refresh  更新数据库实体
clear 清除上下文环境
contains 判断实例是否被上下文环境管理
isOpen  管理器是否打开
getTran 事务
close

createQuery (String qlString)：创建一个查询对象。
createNamedQuery (String name)：根据命名的查询语句块创建查询对象。参数为命名的查询语句。
createNativeQuery (String sqlString)：使用标准 SQL语句创建查询对象。参数为标准SQL语句字符串。
createNativeQuery (String sqls, String resultSetMapping)：使用标准SQL语句创建查询对象，并指定
返回结果集 Map的 名称。
```

五、一对一、一对多、多对多：单向、双向

注意：@JoinColumn可选，会默认设置映射对象xx_id,若默认命名和该对象属性重复，则直接以映射对象名生成外键。只是可取别名

单向

单向使用一边注解即可，无需mappby，多的一方获取少的一方默认是左外连接，区别于SpringData默认接口查询方法是懒加载

双向必须通过mappedBy指定关系维护端

双向

```
##双向一对多:    （顾客、订单为例）
    1=被维护端@OneToMany(fetch=FetchType.LAZY,cascade={CascadeType.REMOVE},mappedBy="customer")
    n=关系维护端@JoinColumn(name="CUSTOMER_ID")       //可选 因为必须通过mapperby指定关系，否则生成
表数错误 如果两者都没，就会在两边都建立外键造成删除不了表，执行以下sql才可删除
    set @@foreign_key_checks=OFF;临时关闭约束

    SET FOREIGN_KEY_CHECKS=0;关闭约束
```

```
     @ManyToOne(fetch=FetchType.LAZY)
    *保存
    1）建议先保存1的一方 顾客表采用表主键策略，订单id自增    5条语句：查更+3条insert
    2）先保存n的一方  7条sql：2条oder insert+2条主键查更+1条customer insert+2条order update
    *不能先删除1的一端，有外键约束，设置级联可连带删除n的一端
    *在1的一方设置懒加载策略，即两次查询，否则默认左外连接
    *默认查1的一方即是会左外连接连带查出n的一方的，只是可以设置加载策略为懒加载
##双向一对一：（经理、部门为例）
    1=@OneToOne(fetch=FetchType.LAZY)
    1=@OneToOne(mappedBy="mgr") 必须有mapperby 否则两边都生成外键无法删除
    1）先保存没有外键的一方,2条sql
    2）先保存外键一方，3条其中最后一条更新
    *加载策略可查出来优化 如部门类设置懒加载，经理默认，查部门，3条sql：先查部门，再经理，懒加载部门第二条
    *先查经理默认左外连接 1条sql

##双向多对多：（商品类目、商品为例）
    n=@JoinTable(name="ITEM_CATEGORY",
        joinColumns={@JoinColumn(name="ITEM_ID", referencedColumnName="ID")},
        inverseJoinColumns={@JoinColumn(name="CATEGORY_ID", referencedColumnName="ID")})
    @ManyToMany
    n=@ManyToMany(mappedBy="categories")
    *一定要有维护关系且只能通过mapperby指定，不能通过@JoinColumn指定，使用一个或多个@JoinColumn都将生成表个数错误 都是4张
    *多对多保存数据8条sql：2+2+4条桥表
    *多对多查数据，两边查都一样的sql语句，默认就是懒加载，区别于上两种默认左外连接
```

六、二级缓存

ALL：所有的实体类都被缓存

NONE所有的实体类都不被缓存.

ENABLE_SELECTIVE：标识 @Cacheable(true) 注解的实体类将被缓存

DISABLE_SELECTIVE：缓存除标识 @Cacheable(false) 以外的所有实体类

UNSPECIFIED：默认值，JPA 产品默认值将被使用

```
默认一级缓存同一个管理器内 相同sql只查一次 类比mybatis缓存同一session
开启二级缓存 不同管理器相同sql也只查一次   类比mybatis缓存不同session   前提是配置合理
```

七、Query接口的方法

getSingleResult getResultList setHint缓存

1,createQuery 全部属性、部分属性、where、orderby、groupby、外连接、子查询、多表、内建函数等 对象
2,createNamedQuery query定义在实体中 无需select 对象 3,createNativeQuery setHint 标准sql
4,createNativeQuery 带结果集参数 标准sql

八、JPA操作数据的几种方式

1. 继承接口默认的方式，不够通过通过Query接口自定义JPQL
2. 使用JPA API 管理器操作

3. 自定义JPQL语言sql的编写查删改 JPQL对实体操作不支持insert，对本地sql可以
4. 自定义repository@PersistenceContext+管理器操作

   注意：增删改必须加@Modify注解，修改删除只能返回int或void，且必须在调用方法处声明事务（SpringData提供默认接口方式不用，Spring环境中调用客户端管理器操作也要事务）

SpringData相关

一、Repository接口

无任何方法，需按照规定自定义方法 使用方式：1，继承接口 2，注解 @RepositoryDefinition(domainClass=Person.class,idClass=Integer.class)+自定义方法

二、@NoRepositoryBean注解

过滤创建bean 无法注入 注解标识作为中间接口，不创建代理，如JpaRepo、CrudRepo等

三、CurdRepository、PagingAndSortingRepository、JpaRepository、JpaSpecificationExecutor（分页+筛选条件）等的使用

自定义JPQL

- ?1　第一个参数
- :name+@Param(不加默认相同名字)第一个参数
- ?#{[0]}第一个参数
- :#{#productInfo.productId} 参数为封装对象
- *#{#entityName}取实体

```
@Query("select t from #{#entityName} t where t.attribute = ?1")
List<T> findAllByAttribute(String attribute);

@Query("select u from User u where u.firstname = ?1 and u.firstname=?#{[0]} and u.emailAddress =
?#{principal.emailAddress}")
List<User> findByFirstnameAndCurrentUserWithCustomQuery(String firstname);

@Query("select u from User u where u.lastname like %:#{[0]}% and u.lastname like %:lastname%")
List<User> findByLastnameWithSpelExpression(@Param("lastname") String lastname);

@Query("select u from User u where u.firstname like %?#{escape([0])}% escape ?#
{escapeCharacter()}")
List<User> findContainingEscaped(String namePart);
```

5.3.7使用SpEL表达式

官方文档https://docs.spring.io/spring-data/jpa/docs/2.1.10.RELEASE/reference/html/#jpa.modifying-queries
对象参数https://spring.io/blog/2014/07/15/spel-support-in-spring-data-jpa-query-definitions

```
@Query("select u from User u where u.age = ?#{[0]}")
List<User> findUsersByAge(int age);

@Query("select u from User u where u.firstname = :#{#customer.firstname}")
List<User> findUsersByCustomersFirstname(@Param("customer") Customer customer);
```

由于JPA、lombok引起的CleanUp问题 https://blog.csdn.net/qq_22327273/article/details/88578187?tdsourcetag=s_pctim_aiomsg