# SpringData– day03

### 0.1学习目标

* 掌握 specification动态多条件查询
* 理解 关系型数据库多表之间关系
* 掌握 一对多关联操作
* 掌握 多对多关联操作
* 掌握 对象导航查询

### 0.2学习指南

* 动态查询概述
* Specification 项目搭建、模糊、分页、排序查询
* 多表关系回顾
* 一对多关联操作 项目搭建、级联、映射、外间维护 操作
* 多对多关联操作 项目搭建、级联、映射、外间维护 操作
* 对象导航查询 一对多 延迟加载

### 第一堂课

**本节知识点**

回顾与今日内容介绍

动态查询的概述

JpaSpecificationExecutor方法介绍

搭建测试环境

构造查询条件的步骤分析

使用动态查询完成查询单个对象

**本节目标**

理解动态查询概述

实现测试环境的搭建

掌握查询条件的步骤分析

完成动态查询单个对象

#### 回顾与今日内容介绍

##### 1.1.1知识概述

springData回顾第二天内容及今天课程内容 ：

回顾

i.springDatajpa，jpa规范，hibernate三者之间的关系

code -- > springDatajpa --> jpa规范的API --> hibernate

ii.符合springDataJpa规范的dao层接口的编写规则

1.需要实现两个接口（JpaRepository，JapSpecificationExecutor）

2.提供响应的泛型

iii.运行过程

\* 动态代理的方式：动态代理对象

iiii.查询

今天内容

第一 Specifications动态查询

第二 多表之间的关系和操作多表的操作步骤

第三 完成多表操作

##### 1.1.2视频详情



##### 1.1.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 1.1.4课堂提问与练习

无

##### 1.1.5习题答案

无

#### 动态查询的概述

##### 1.2.1知识概述

由于企业开发时，搜索的条件内容个数不确定，使用Specifications动态查询解决此问题

##### 1.2.2视频详情



##### 1.2.3总结与补充

###### 简单:

之前学习mybatis逆向工程，和动态sql语句可以轻松做到条件不固定

###### 中等:

###### 复杂:

##### 1.2.4课堂提问与练习

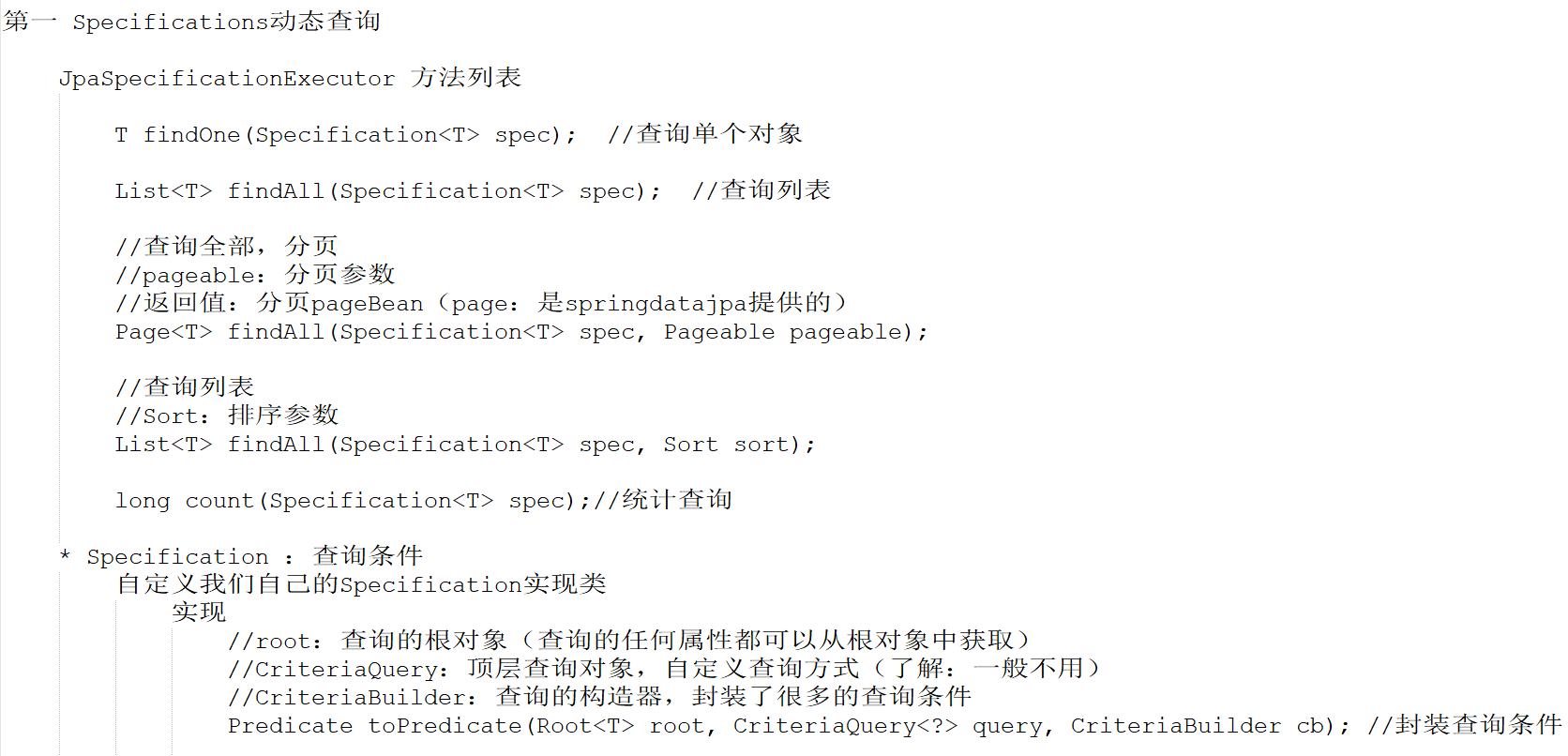
无

##### 1.2.5习题答案

无

#### 1.3 JpaSpecificationExecutor方法介绍

##### 1.3.1 知识概述



##### 1.3.2 视频详情

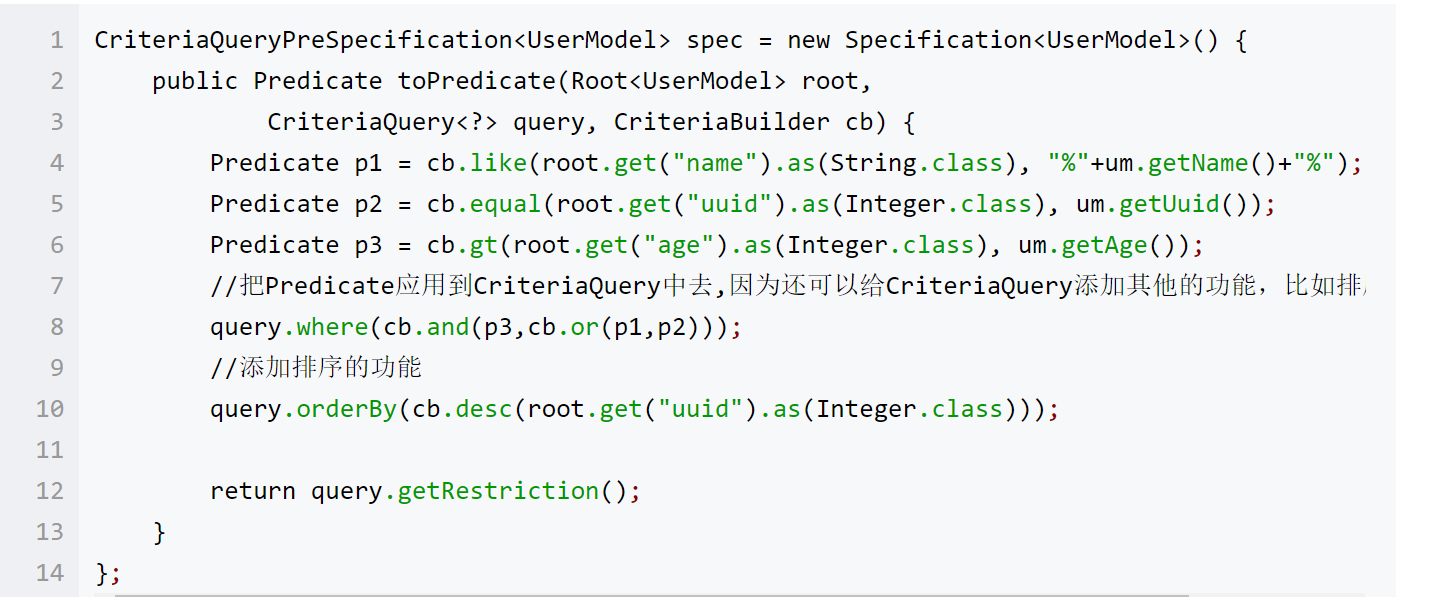


##### 1.3.3 总结与补充

###### 简单:

###### 中等:

CriteriaQuery：顶层查询对象，自定义查询方式



###### 复杂:

##### 1.3.4 课堂提问与练习

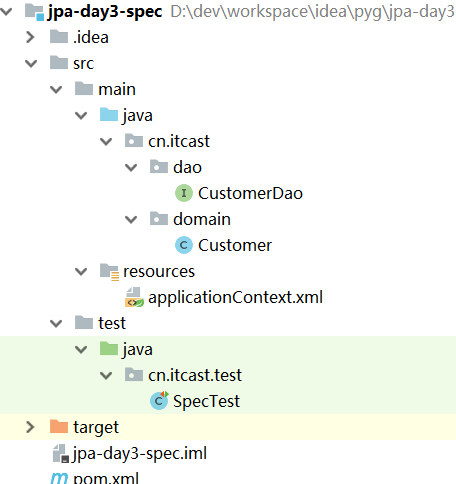
无

##### 1.3.5 习题答案

无

#### 1.4 搭建测试环境

##### 1.4.1 知识概述



##### 1.4.2 视频详情



##### 1.4.3 总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 课堂提问与练习

让学员完成工程搭建

##### 1.4.5 习题答案

无

#### 1.5 构造查询条件的步骤分析

##### 1.5.1 知识概述



##### 1.5.2 视频详情



##### 1.5.3 总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 1.5.4课堂提问与练习

无

##### 1.5.5习题答案

无

#### 1.6 使用动态查询完成查询单个对象

##### 1.6.1 知识概述

案例：根据客户名称查询，查询客户名为传智播客的客户

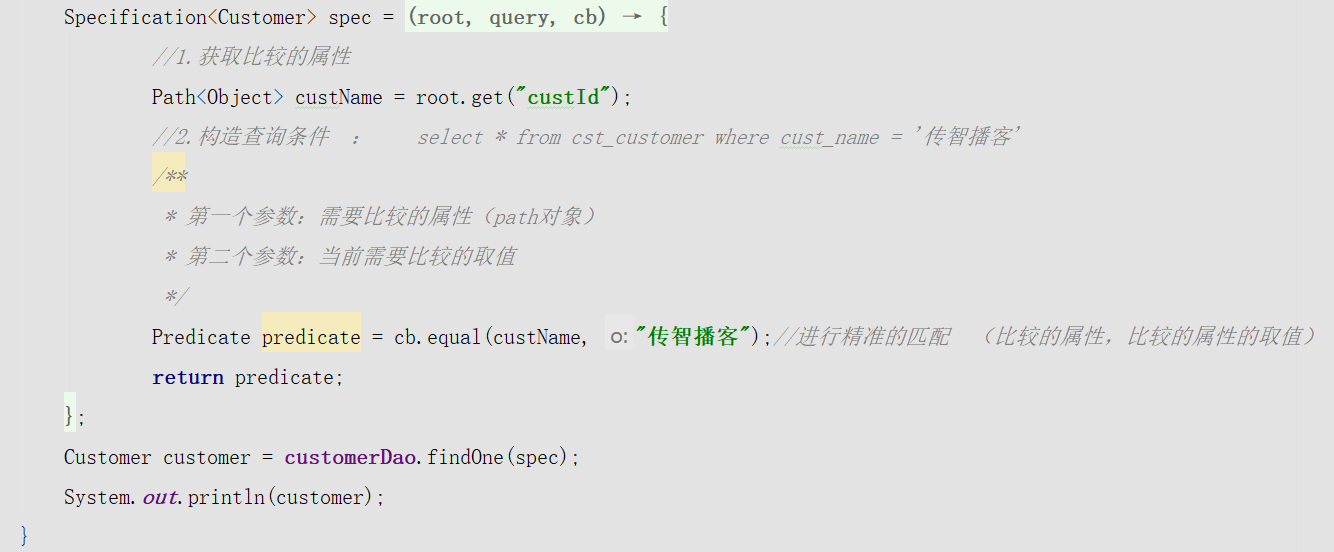
查询条件

1.查询方式

cb对象

2.比较的属性名称

root对象



##### 1.6.2 视频详情



##### 1.6.3 总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 1.6.4课堂提问与练习

让学员实现查询单个对象

##### 1.6.5习题答案

无

### 第二堂课

**本节知识点**

动态查询完成多条件拼接

模糊匹配查询列表

排序

分页

**本节目标**

完成多条件动态查询

完成条件模糊查询

完成排序查询

完成分页查询

#### 动态查询完成多条件拼接

##### 2.1.1知识概述

多条件查询

案例：根据客户名（传智播客）和客户所属行业查询（it教育）

root:获取属性

客户名

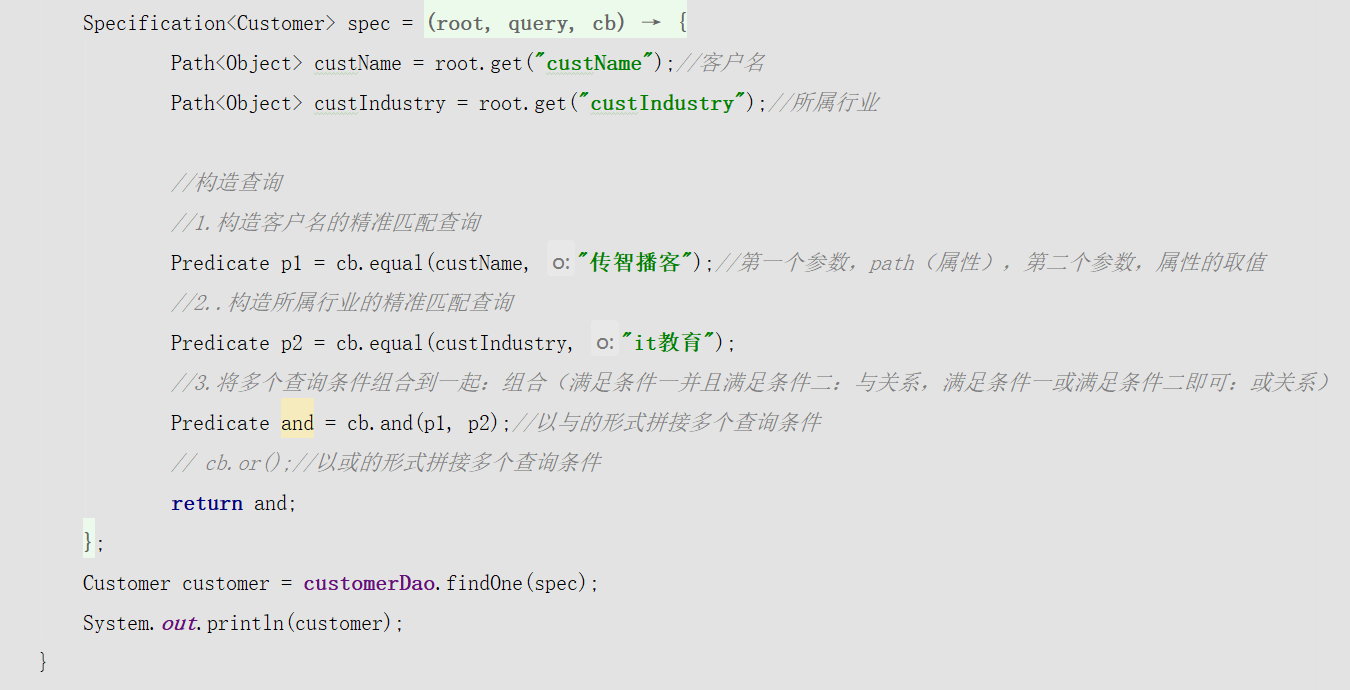
所属行业

cb：构造查询

1.构造客户名的精准匹配查询

2.构造所属行业的精准匹配查询

3.将以上两个查询联系起来



##### 2.1.2视频详情



##### 2.1.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 2.1.4课堂提问与练习

无

##### 2.1.5习题答案

无

#### 模糊匹配查询列表

##### 2.2.1知识概述

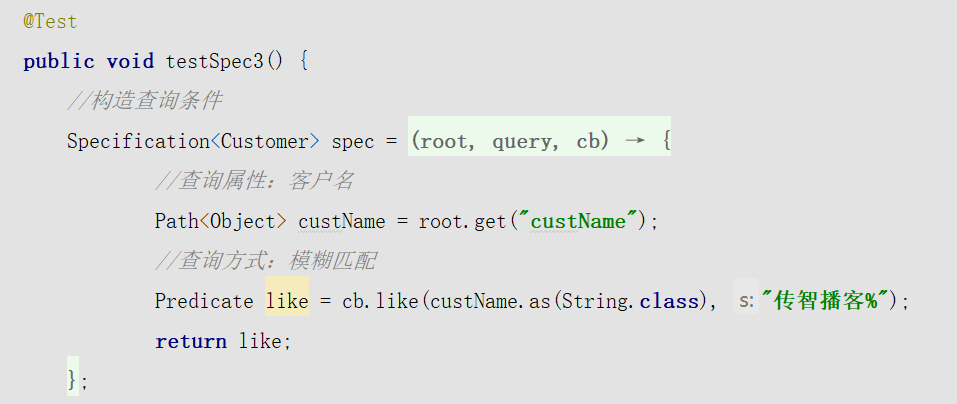
案例：完成根据客户名称的模糊匹配，返回客户列表

客户名称以 ’传智播客‘ 开头

equal ：直接的到path对象（属性），然后进行比较即可

gt，lt,ge,le,like : 得到path对象，根据path指定比较的参数类型，再去进行比较

指定参数类型：path.as(类型的字节码对象)



##### 2.2.2视频详情



##### 2.2.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 2.2.4课堂提问与练习

让学员完成模糊匹配查询列表

##### 2.2.5习题答案

#### 排序

##### 2.3.1知识概述

添加排序

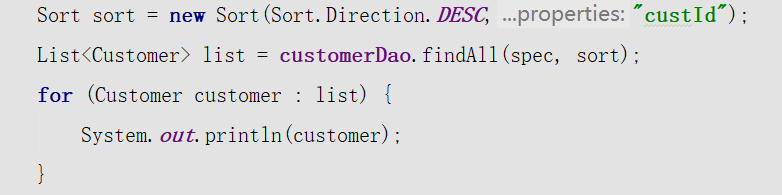
创建排序对象,需要调用构造方法实例化sort对象

第一个参数：排序的顺序（倒序，正序）

Sort.Direction.DESC:倒序

Sort.Direction.ASC ： 升序

第二个参数：排序的属性名称



##### 2.3.2视频详情



##### 2.3.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 2.3.4课堂提问与练习

无

##### 2.3.5习题答案

无

#### 分页

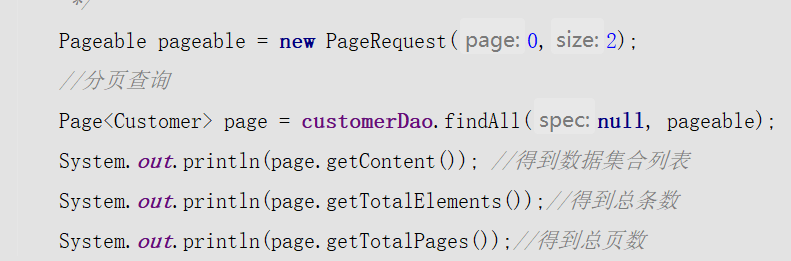
##### 2.4.1知识概述

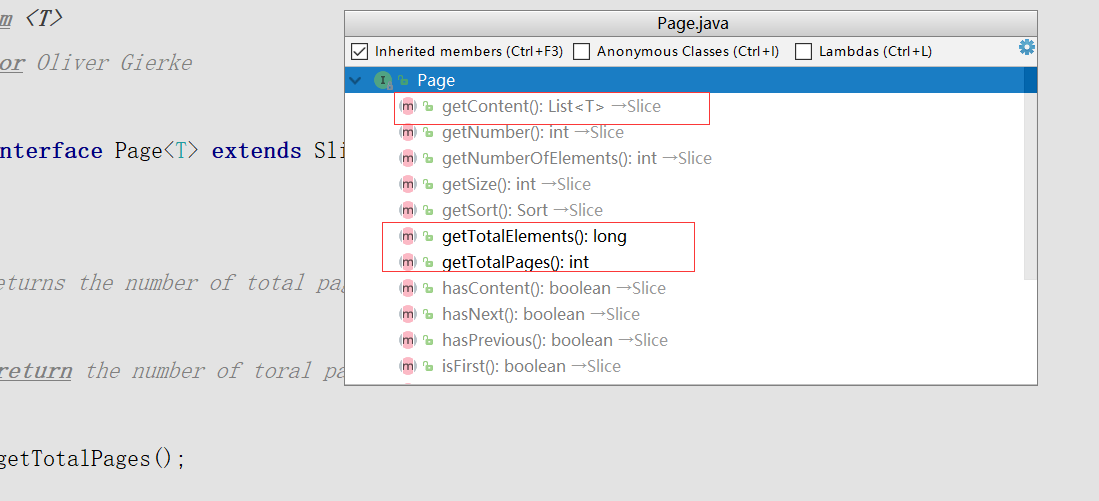
PageRequest对象是Pageable接口的实现类

创建PageRequest的过程中，需要调用他的构造方法传入两个参数

第一个参数：当前查询的页数（从0开始）

第二个参数：每页查询的数量2.4.2视频详情





##### 2.3.2视频详情



##### 2.4.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 2.4.4课堂提问与练习

无

##### 2.4.5习题答案

无

### 第三堂课

**本节知识点**

specification查询总结

多表关系回顾

多表操作的分析步骤

一对多：案例分析

一对多：一对一的测试环境搭建

**本节目标**

掌握specification条件查询对象使用

理解关系型数据库多表之间关系

完成一对一测试环境的搭建

#### 3.1 specification查询总结

##### 3.1.1知识概述

Specification 条件对象

root：查询的根对象（查询的任何属性都可以从根对象中获取）

criteriaQuery：顶层查询对象，自定义查询方式（了解：一般不用）

criteriaBuilder：查询的构造器，封装了很多的查询条件

Pageable 分页查询对象

sage 当前查询页数 从0开始

size 每页查询的数量

Sort 排序对象

Sort.Direction.DESC: 倒序

Sort.Direction.ASC ：升序

properties: 需要排序的字段 多个逗号分割

##### 3.1.2视频详情



##### 3.1.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 3.1.4课堂提问与练习

无

##### 3.1.5习题答案

无

#### 多表关系回顾

##### 3.2.1知识概述

多表之间的关系和操作多表的操作步骤

表关系

一对一

一对多：

一的一方：主表

多的一方：从表

外键：需要再从表上新建一列作为外键，他的取值来源于主表的主键

多对多：

中间表：中间表中最少应该由两个字段组成，这两个字段做为外键指向两张表的主键，又组成了联合主键

讲师对学员：一对多关系

实体类中的关系

包含关系：可以通过实体类中的包含关系描述表关系

继承关系

##### 3.2.2视频详情



##### 3.2.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 3.2.4课堂提问与练习

无

##### 3.2.5习题答案

无

#### 多表操作的分析步骤

##### 3.3.1知识概述

分析步骤

1.明确表关系

2.确定表关系（描述 外键|中间表）

3.编写实体类，再实体类中描述表关系（包含关系）

4.配置映射关系

##### 3.3.2视频详情



##### 3.3.3总结与补充

###### 简单:

ER模型可以通过业务设计表之间的关系

###### 中等:

###### 复杂:

##### 3.3.4课堂提问与练习

无

##### 3.3.5习题答案

无

#### 一对多：案例分析

##### 3.4.1知识概述

一对多操作

案例：客户和联系人的案例（一对多关系）

客户：一家公司

联系人：这家公司的员工

一个客户可以具有多个联系人

一个联系人从属于一家公司

分析步骤

1.明确表关系

一对多关系

2.确定表关系（描述 外键|中间表）

主表：客户表

从表：联系人表

\* 再从表上添加外键

3.编写实体类，再实体类中描述表关系（包含关系）

客户：再客户的实体类中包含一个联系人的集合

联系人：在联系人的实体类中包含一个客户的对象

4.配置映射关系

\* 使用jpa注解配置一对多映射关系

##### 3.4.2视频详情



##### 3.4.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 3.4.4课堂提问与练习

无

##### 3.4.5习题答案

无

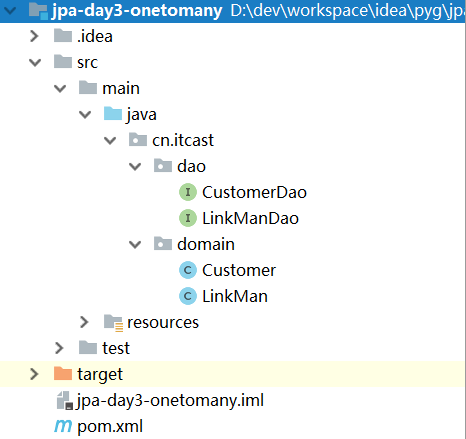
#### 一对多：一对一的测试环境搭建

##### 3.5.1知识概述

导入sql脚本

参照 :

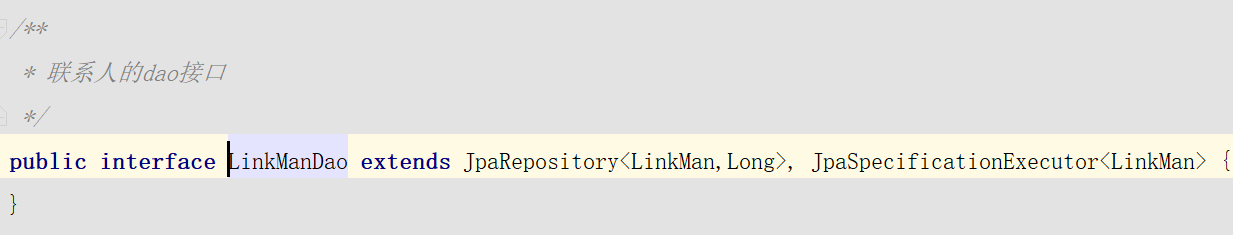
jpa-day3-spec 创建 jpa-day3-onetomany工程



创建联系人实体：



编写dao接口



##### 3.5.2视频详情



##### 3.5.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 3.5.4课堂提问与练习

无

##### 3.5.5习题答案

无

### 第四堂课

**本节知识点**

一对多：配置一对多和多对一

一对多：保存操作-上

一对多：放弃外键维护

一对多：删除的说明和级联的引入

**本节目标**

完成实体客户与联系人之间的关系配置

完成实体客户与联系人保存

完成实体客户放弃外键维护

理解实体删除的说明

理解级联操作的概念

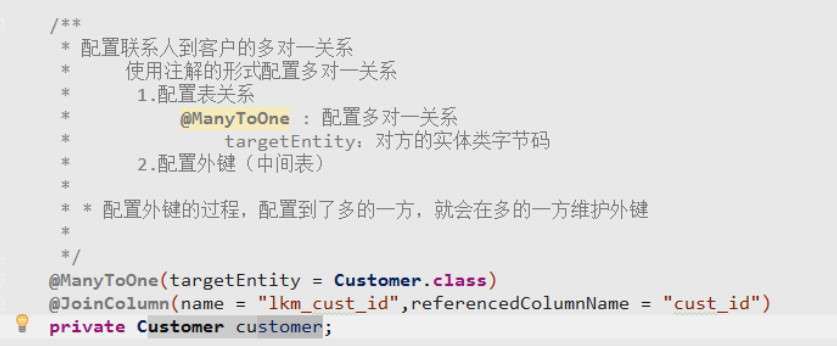
#### 4.1 一对多：配置一对多和多对一

##### 4.1.1知识概述

客户一对多实体配置



联系人对多一实体配置



@JoinColumn

字段来维护外键关系

##### 4.1.2视频详情



##### 4.1.3总结与补充

###### 简单:

一般情况下 一对多，一的一方是放弃维护外键

###### 中等:

如果一的一方需要维护外键，并且需要排序

@OrderColumn(name = "索引列")

private List<多方实体>

###### 复杂:

##### 4.1.4课堂提问与练习

无

##### 4.1.5习题答案

无

#### 4.2 一对多：保存操作-上

##### 4.2.1知识概述

1. Jpa配置信息



2. 编写单元测试类 客户与联系人 保存



##### 4.2.2视频详情



##### 4.2.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 4.2.4课堂提问与练习

无

##### 4.2.5习题答案

无

#### 一对多：放弃外键维护

##### 4.3.1知识概述

联系人关联客户

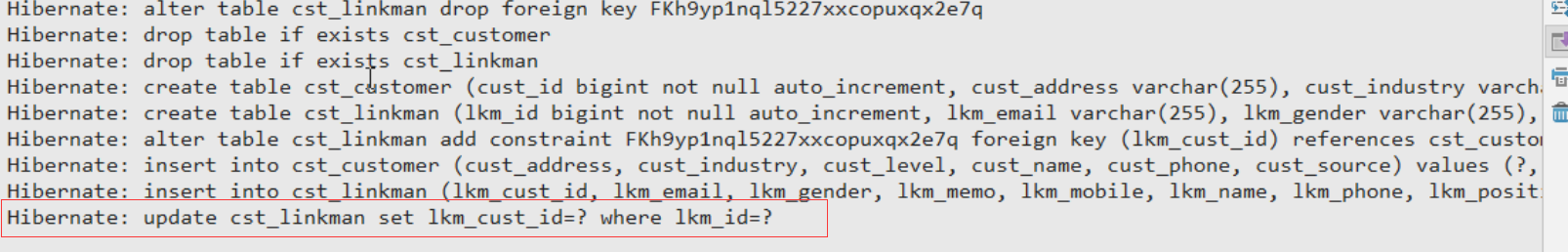
//由于配置了多的一方到一的一方的关联关系（当保存的时候，就已经对外键赋值）

linkMan.setCustomer(customer);

客户关联联系人

//由于配置了一的一方到多的一方的关联关系（发送一条update语句）

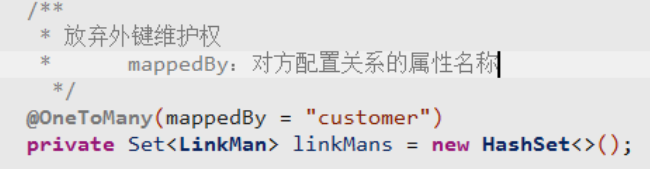
customer.getLinkMans().add(linkMan);

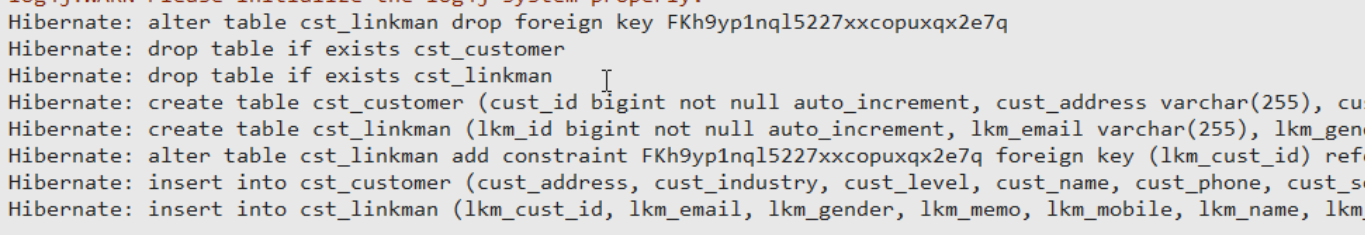


由上图得知 ，会有一条多余的update语句

由于一的一方可以维护外键：会发送update语句

解决此问题：只需要在一的一方放弃维护权即可





修改后运行，此时已经没有 update语句

##### 4.3.2视频详情



##### 4.3.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 4.3.4课堂提问与练习

无

##### 4.3.5习题答案

无

#### 一对多：删除的说明和级联的引入

##### 4.4.1知识概述

删除操作的说明如下：

**删除从表数据：可以随时任意删除。**

**删除主表数据：**

* 有从表数据

1、在默认情况下，它会把外键字段置为null，然后删除主表数据。如果在数据库的表 结构上，外键字段有非空约束，默认情况就会报错了。

2、如果配置了放弃维护关联关系的权利，则不能删除（与外键字段是否允许为null， 没有关系）因为在删除时，它根本不会去更新从表的外键字段了。

3、如果还想删除，使用级联删除引用

* 没有从表数据引用：随便删

在实际开发中，级联删除请慎用！(在一对多的情况下)

级联：

操作一个对象的同时操作他的关联对象

级联操作：

1.需要区分操作主体

2.需要在操作主体的实体类上，添加级联属性（需要添加到多表映射关系的注解上）

3.cascade（配置级联）

级联添加，

案例：当我保存一个客户的同时保存联系人

级联删除

案例：当我删除一个客户的同时删除此客户的所有联系人

##### 4.4.2视频详情



##### 4.4.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 4.4.4课堂提问与练习

无

##### 4.4.5习题答案

无

### 第五堂课

**本节知识点**

一对多：级联操作

多对多：案例以及搭建基本框架

多对多：映射配置

多对多：放弃维护权

**本节目标**

完成一对多级联保存与删除

完成多对多项目搭建

完成用户与角色实体与映射配置

完成角色对外键维护权

#### 一对多：级联操作

##### 5.1.1知识概述

级联添加：保存一个客户的同时，保存客户的所有联系人

需要在操作主体的实体类上，配置casacde属性

|  |
| --- |
| @Test @Transactional *//配置事务* @Rollback(**false**) *//不自动回滚* **public void** testCascadeAdd() {  Customer customer = **new** Customer();  customer.setCustName(**"百度1"**);   LinkMan linkMan = **new** LinkMan();  linkMan.setLkmName(**"小李1"**);   linkMan.setCustomer(customer);  customer.getLinkMans().add(linkMan);   **customerDao**.save(customer); } |

级联删除：

删除1号客户的同时，删除1号客户的所有联系人

|  |
| --- |
| @Test @Transactional *//配置事务* @Rollback(**false**) *//不自动回滚* **public void** testCascadeRemove() {  *//1.查询1号客户* Customer customer = **customerDao**.findOne(1l);  *//2.删除1号客户* **customerDao**.delete(customer); } |

cascade : 配置级联（可以配置到设置多表的映射关系的注解上）

CascadeType.all : 所有

MERGE ：更新

PERSIST ：保存

REMOVE ：删除

##### 5.1.2视频详情



##### 5.1.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 5.1.4课堂提问与练习

无

##### 5.1.5习题答案

无

#### 多对多：案例以及搭建基本框架

##### 5.2.1知识概述

1.多对多操作

案例：用户和角色（多对多关系）

用户：

角色：

分析步骤

1.明确表关系

多对多关系

2.确定表关系（描述 外键|中间表）

中间间表

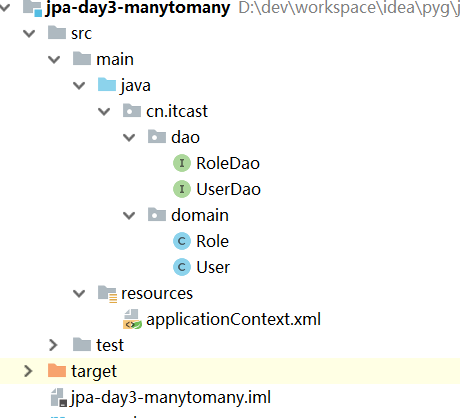
3.编写实体类，再实体类中描述表关系（包含关系）

用户：包含角色的集合

角色：包含用户的集合

4.配置映射关系

2.参考jpa-day3-onetomany 搭建jpa-day3-manytomany工程



3.实体

User

|  |
| --- |
| @Entity @Table(name = **"sys\_user"**) **public class** User {   @Id  @GeneratedValue(strategy = GenerationType.***IDENTITY***)  @Column(name=**"user\_id"**)  **private** Long **userId**;  @Column(name=**"user\_name"**)  **private** String **userName**;  @Column(name=**"age"**)  **private** Integer **age**; |

Role

|  |
| --- |
| @Entity @Table(name = **"sys\_role"**) **public class** Role {   @Id  @GeneratedValue(strategy = GenerationType.***IDENTITY***)  @Column(name = **"role\_id"**)  **private** Long **roleId**;  @Column(name = **"role\_name"**)  **private** String **roleName**; |

##### 5.2.2视频详情



##### 5.2.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 5.2.4课堂提问与练习

无

##### 5.2.5习题答案

无

#### 多对多：映射配置

##### 5.3.1知识概述

配置用户到角色的多对多关系

配置多对多的映射关系

1.声明表关系的配置

@ManyToMany(targetEntity = Role.class) //多对多

targetEntity：代表对方的实体类字节码

2.配置中间表（包含两个外键）

@JoinTable

name : 中间表的名称

joinColumns：配置当前对象在中间表的外键

@JoinColumn的数组

name：外键名

referencedColumnName：参照的主表的主键名

inverseJoinColumns：配置对方对象在中间表的外键

|  |
| --- |
| @ManyToMany(targetEntity = Role.**class**,cascade = CascadeType.***ALL***) @JoinTable(name = **"sys\_user\_role"**,  *//joinColumns,当前对象在中间表中的外键* joinColumns = {@JoinColumn(name = **"sys\_user\_id"**,referencedColumnName = **"user\_id"**)},  *//inverseJoinColumns，对方对象在中间表的外键* inverseJoinColumns = {@JoinColumn(name = **"sys\_role\_id"**,referencedColumnName = **"role\_id"**)} ) **private** Set<Role> **roles** = **new** HashSet<Role>(); |

##### 5.3.2视频详情



##### 5.3.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 5.3.4课堂提问与练习

无

##### 5.3.5习题答案

无

#### 多对多：放弃维护权

##### 5.4.1知识概述

角色放弃主键维护

|  |
| --- |
| *//配置多对多* @ManyToMany(mappedBy = **"roles"**) *//配置多表关系* **private** Set<User> **users** = **new** HashSet<User>(); |

保存一个用户，保存一个角色

多对多放弃维护权：被动的一方放弃

|  |
| --- |
| @Test @Transactional @Rollback(**false**) **public void** testAdd() {  User user = **new** User();  user.setUserName(**"小李"**);   Role role = **new** Role();  role.setRoleName(**"java程序员"**);   *//配置用户到角色关系，可以对中间表中的数据进行维护 1-1* user.getRoles().add(role);   *//配置角色到用户的关系，可以对中间表的数据进行维护 1-1* role.getUsers().add(user);   **userDao**.save(user);  **roleDao**.save(role); } |

##### 5.4.2视频详情



##### 5.4.3总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 5.4.4课堂提问与练习

无

##### 5.4.5习题答案

无

### 第六堂课

**本节知识点**

多对多：级联操作

对象导航查询介绍

对象导航查询-延迟加载

对象导航查询：从多方查询一方

对象导航查询的总结

**本节目标**

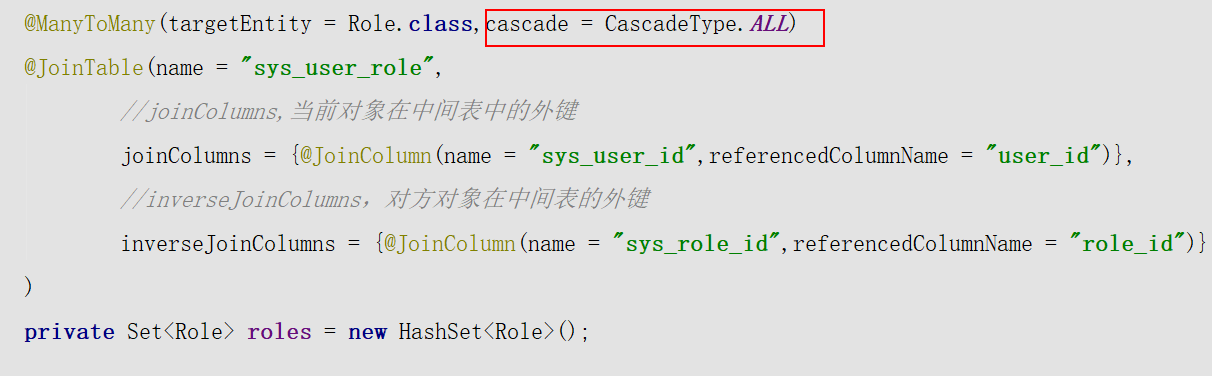
完成品牌删除操作

完成品牌条件查询

#### 6.1 多对多：级联操作

##### 6.1.1 知识概述

1.开启级联操作



2.级联添加 保存一个用户的同时保存用户的关联角色

|  |
| --- |
| @Test @Transactional @Rollback(**false**) **public void** testCasCadeAdd() {  User user = **new** User();  user.setUserName(**"小李"**);   Role role = **new** Role();  role.setRoleName(**"java程序员"**);   *//配置用户到角色关系，可以对中间表中的数据进行维护 1-1* user.getRoles().add(role);   *//配置角色到用户的关系，可以对中间表的数据进行维护 1-1* role.getUsers().add(user);   **userDao**.save(user); } |

3.级联删除 删除id为1的用户，同时删除他的关联对象

|  |
| --- |
| @Test @Transactional @Rollback(**false**) **public void** testCasCadeRemove() {  *//查询1号用户* User user = **userDao**.findOne(1l);  *//删除1号用户* **userDao**.delete(user);  } |

##### 6.1.2 视频详情



##### 6.1.3 总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 6.1.4 课堂提问与练习

无

##### 6.1.5 习题答案

无

#### 6.2 对象导航查询介绍

##### 6.2.1 知识概述

对象导航查询

查询一个对象的同时，通过此对象查询他的关联对象

案例：客户和联系人

|  |
| --- |
| *//could not initialize proxy - no Session //测试对象导航查询（查询一个对象的时候，通过此对象查询所有的关联对象）* @Test @Transactional *// 解决在java代码中的no session问题* **public void** testQuery1() {  *//查询id为1的客户* Customer customer = **customerDao**.getOne(1l);  *//对象导航查询，此客户下的所有联系人* Set<LinkMan> linkMans = customer.getLinkMans();   **for** (LinkMan linkMan : linkMans) {  System.***out***.println(linkMan);  } } |

##### 6.2.2 视频详情



##### 6.2.3 总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 6.2.4 课堂提问与练习

无

##### 6.2.5 习题答案

无

#### 6.3 对象导航查询-延迟加载

##### 6.3.1 知识概述

对象导航查询：

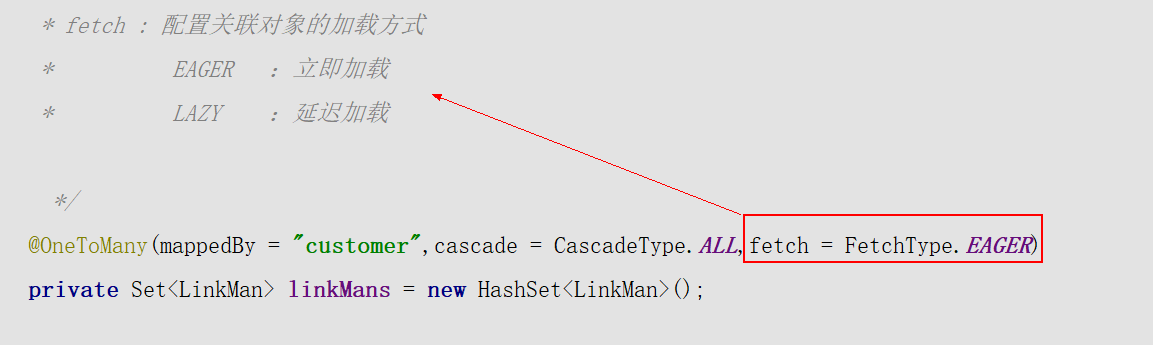
默认使用的是延迟加载的形式查询的

调用get方法并不会立即发送查询，而是在使用关联对象的时候才会差和讯

延迟加载！

修改配置，将延迟加载改为立即加载

fetch，需要配置到多表映射关系的注解上



##### 6.3.2 视频详情



##### 6.3.3 总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 6.3.4 课堂提问与练习

无

##### 6.3.5 习题答案

无

#### 6.4 对象导航查询：从多方查询一方

##### 6.4.1 知识概述

多对一查询时，默认立即加载



##### 6.4.2 视频详情



##### 6.4.3 总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 6.4.4 课堂提问与练习

无

##### 6.4.5 习题答案

无

#### 6.5 对象导航查询的总结

##### 6.5.1 知识概述

从一方查询多方

\* 默认：使用延迟加载（\*\*\*\*）

从多方查询一方

\* 默认：使用立即加载

##### 6.5.2 视频详情



##### 6.5.3 总结与补充

###### 简单:

###### 中等:

###### 复杂:

##### 6.5.4 课堂提问与练习

无

##### 6.5.5 习题答案

无