第10天

品优购电商系统开发

# 上节内容回顾

1. 什么是solr
2. solr的域及安装
3. SpringDataSolr
4. spring data solr操作solr索引库，并完成

# 今日学习内容(学习内容按视频的顺序进行归纳总结)

1. 搜索结果高亮显示功能
2. 搜索的业务规则和实现思路
3. 查询分类列表的功能
4. 缓存品牌和规格数据的功能
5. 显示品牌和规格数据的功能
6. 过滤条件构建的功能
7. 过滤查询的功能

# 课程介绍及需求(视频01) [了解]

## 今日目标

继续学习solr，并对solr进行一些稍微高级一点的查询应用，包括：高亮，过滤等

# 高亮显示(视频02~03) [撑握]

## 高亮显示-后端

把搜索的关键字用特殊的颜色显示出来

我们要修改一下我们查询的内容

|  |
| --- |
| @Autowired  SolrTemplate solrTemplate;  @Override  **public** Map<String, Object> search(Map<String, Object> searchMap) {  Map<String, Object> map = **new** HashMap<String, Object>();  /\* Query query = new SimpleQuery(); \*/  SimpleHighlightQuery query = **new** SimpleHighlightQuery();  HighlightOptions hos = **new** HighlightOptions();  hos.setSimplePostfix("</em>");  hos.addField("item\_title");  hos.setSimplePrefix("<em style='color:red'>");  query.setHighlightOptions(hos);  // 添加查询条件  Criteria criteria = **new** Criteria("item\_keywords").is(searchMap.get("keywords"));  query.addCriteria(criteria);  /\* ScoredPage<TbItem> page = solrTemplate.queryForPage(query, TbItem.class); \*/  HighlightPage<TbItem> page = solrTemplate.queryForHighlightPage(query, TbItem.**class**);  List<HighlightEntry<TbItem>> highlighted = page.getHighlighted();  **for** (HighlightEntry<TbItem> h : highlighted) {  TbItem item = h.getEntity();// 获取原实体类  **if** (h.getHighlights().size() > 0 && h.getHighlights().get(0).getSnipplets().size() > 0) {  item.setTitle(h.getHighlights().get(0).getSnipplets().get(0));// 设置高亮的结果  }  }  map.put("rows", page.getContent());  **return** map;  } |

查看页面



这是不对的。

## 高亮显示-前端

上面的html直接不解析，这是angular的问题，这里面这样做主要是为了防止html攻击。我们如何在页面上显示html的结果呢？我们会用到$sce服务的trustAsHtml方法来实现转换。

1. 修改 base.js

// 定义模块:

**var** app = angular.module("pinyougou",[]);

/\*$sce服务写成过滤器\*/

app.filter('trustHtml',['$sce',**function**($sce){

**return** **function**(data){

**return** $sce.trustAsHtml(data);

}

}]);

1. 使用过滤器

ng-bind-html 指令用于显示 html 内容

竖线 | 用于调用过滤器

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| <**div class="attr" ng-bind-html="item.title|trustHtml"**>  <**em**>{{item.title}}</**em**>  </**div**> |  |  |  |  |

*|*就是竖线，看起来有点斜是因为字体原因。

查看页面



# 整理复杂搜索思路(视频04~05) [撑握]

## 业务规则-需求分析

搜索关键字和商品分类有关：

C:\Users\work\AppData\Local\Temp\snap_screen_20180219201027.png

如果商品分类确定了，商品的大品牌就可以显示出来了。



如果分类有多个，以第一个分类为准。

如果商品分类定了，那么规格相应的也定了



而且上面的所有数据都是可以进行筛选的。

业务规则：

1. 当用户输入关键字搜索后，除了显示列表结果外，还应该显示通过这个关键字搜索到的记录都有哪些**商品分类**。
2. 根据第一个商品分类查询对应的**模板**，根据模板查询出品牌列表
3. 根据第一个商品分类查询对应的模板，根据模板查询出规格列表
4. 当用户点击搜索面板的商品分类时，显示按照这个关键字查询结果的基础上，筛选此分类的结果。
5. 当用户点击搜索面板的品牌时，显示在以上结果的基础上，筛选此品牌的结果
6. 当用户点击搜索面板的规格时，显示在以上结果的基础上，筛选此规格的结果
7. 当用户点击价格区间时，显示在以上结果的基础上，按价格进行筛选的结果
8. 当用户点击搜索面板的相应条件时，隐藏已点击的条件。

## 实现思路与代码结构优化

搜索面板的商品分类需要使用Spring Data Solr的分组查询来实现

为了能够提高查询速度，我们需要把查询面板的品牌、规格数据提前放入redis

查询条件的构建、面板的隐藏需要使用angularJS来实现

后端的分类、品牌、规格、价格区间查询需要使用过滤查询来实现

因为业务逻辑太多，所以先修改一下我们的方法，使之结构更清晰

|  |
| --- |
| @Override  **public** Map<String, Object> search(Map<String, Object> searchMap) {  Map<String, Object> map = **new** HashMap<String, Object>();  searchList(searchMap, map);  **return** map;  }  **private** **void** searchList(Map<String, Object> searchMap, Map<String, Object> map) {  /\* Query query = new SimpleQuery(); \*/  SimpleHighlightQuery query = **new** SimpleHighlightQuery();  HighlightOptions hos = **new** HighlightOptions();  hos.setSimplePostfix("</em>");  hos.addField("item\_title");  hos.setSimplePrefix("<em style='color:red'>");  query.setHighlightOptions(hos);  // 添加查询条件  Criteria criteria =  **new** Criteria("item\_keywords").is(searchMap.get("keywords"));  query.addCriteria(criteria);  /\* ScoredPage<TbItem> page = solrTemplate.queryForPage(query, TbItem.class); \*/  HighlightPage<TbItem> page = solrTemplate.queryForHighlightPage(query, TbItem.**class**);  List<HighlightEntry<TbItem>> highlighted = page.getHighlighted();  **for** (HighlightEntry<TbItem> h : highlighted) {  TbItem item = h.getEntity();// 获取原实体类  **if** (h.getHighlights().size() > 0 && h.getHighlights().get(0).getSnipplets().size() > 0) {  item.setTitle(h.getHighlights().get(0).getSnipplets().get(0));// 设置高亮的结果  }  }  map.put("rows", page.getContent());  } |

# 商品分类查询-分组查询(视频06~07) [撑握]

**需求分析**:根据搜索关键字查询商品分类名称列表



## 商品分类列表-后端（分组查询）

商品分类是分组查询所得

|  |
| --- |
| /\*\*  \* 分组查询商品分类  \*  \* **@param** searchMap  \* **@param** map  \*/  **private** **void** searchGoodsType(Map<String, Object> searchMap, Map<String, Object> map) {  List<String> list = **new** ArrayList<String>();  Query query = **new** SimpleQuery();  // 添加查询条件  Criteria criteria = **new** Criteria("item\_keywords").is(searchMap.get("keywords"));  query.addCriteria(criteria);  // 设置分组选项  GroupOptions groupOptions = **new** GroupOptions().addGroupByField("item\_category");  query.setGroupOptions(groupOptions);  GroupPage<TbItem> page = solrTemplate.queryForGroupPage(query, TbItem.**class**);  // 根据列得到分组结果集  GroupResult<TbItem> groupResult = page.getGroupResult("item\_category");  // 得到分组结果入口页  Page<GroupEntry<TbItem>> groupEntries = groupResult.getGroupEntries();  // 得到分组入口集合  List<GroupEntry<TbItem>> content = groupEntries.getContent();  **for** (GroupEntry<TbItem> entry : content) {  list.add(entry.getGroupValue());// 将分组结果的名称封装到返回值中  }  map.put("categoryList", list);  } |

## 商品分类列表-前端

在页面上进行循环就好了

|  |
| --- |
| <div class=*"type-wrap"* ng-if=*"resultMap.categoryList!=null"*>  <div class=*"fl key"*>商品分类</div>  <div class=*"fl value"*>  <a href=*"#"* ng-repeat=*"cat in resultMap.categoryList"*>{{cat}}</a>  </div>  <div class=*"fl ext"*></div>  </div> |

# 品牌和规格数据查询显示(视频08~13) [撑握]

## 缓存品牌和规格数据的整体思路

在商家里面才可以确定品牌和规格。商品分类与品牌与规格表之前的关系是模版的关系。

将商品分类数据、品牌数据、和规格数据都放入Redis存储。

1. 当用户进入运营商后台的商品分类页面时，将商品分类数据放入缓存（Hash）。以分类名称作为key ,以模板ID作为值
2. 当用户进入运营商后台的模板管理页面时，分别将品牌数据和规格数据放入缓存（Hash）。以模板ID作为key,以品牌列表和规格列表作为值。

现在我们知道的是商品分类名称，用这个名字查询出模版ID。用模版ID查询商品的品牌和商品规格。

## 缓存模板ID

上面说要先用名称查询模版，这里就先添加缓存

|  |
| --- |
| @Autowired  RedisTemplate redisTemplate;  **private** **void** addCatAndTemplate() {  // 每次执行查询的时候，一次性读取缓存进行存储 (因为每次增删改都要执行此方法)  List<TbItemCat> list = findAll();  **for** (TbItemCat itemCat : list) {  redisTemplate.boundHashOps("itemCat").put(itemCat.getName(), itemCat.getTypeId());  }  } |

## 缓存品牌列表和规格列表

缓存品牌和规格列表数据

（1）修改pinyougou-sellergoods-service的TypeTemplateServiceImpl.java

|  |
| --- |
| @Autowired  **private** RedisTemplate redisTemplate;    /\*\*  \* 将数据存入缓存  \*/  **private** **void** saveToRedis(){  //获取模板数据  List<TbTypeTemplate> typeTemplateList = findAll();  //循环模板  **for**(TbTypeTemplate typeTemplate :typeTemplateList){  //存储品牌列表  List<Map> brandList = JSON.*parseArray*(typeTemplate.getBrandIds(), Map.**class**);  redisTemplate.boundHashOps("brandList").put(typeTemplate.getId(), brandList);  //存储规格列表  List<Map> specList = findSpecList(typeTemplate.getId());//根据模板ID查询规格列表  redisTemplate.boundHashOps("specList").put(typeTemplate.getId(), specList); }  } |

（2）在查询分页方法(findPage) 时调用此方法

|  |
| --- |
| **public** PageResult findPage(TbTypeTemplate typeTemplate, **int** pageNum, **int** pageSize) {  ......  saveToRedis();//存入数据到缓存  **return** **new** PageResult(page.getTotal(), page.getResult());  } |

这样在增删改后会自动调用该方法.

## 缓存数据测试

访问模版管理和分类处理，查看redis数据。

## 显示品牌和规格列表-后端

用品牌分类名称去查询结果

|  |
| --- |
| /\*\*  \* 查询品牌和规格列表。  \*  \* **@param** cat  \* 品牌分类的名称  \* **@param** map  \*/  **private** **void** searchBrandAndSpec(Map<String, Object> map) {  List<String> list = (List<String>) map.get("categoryList");  **if** (list == **null** || list.size() == 0) {  **return**;  }  Long typeId = (Long) redisTemplate.boundHashOps("itemCat").get(list.get(0));// 获取模板ID  **if** (typeId != **null**) {  // 根据模板ID查询品牌列表  List brandList = (List) redisTemplate.boundHashOps("brandList").get(typeId);  map.put("brandList", brandList);// 返回值添加品牌列表  // 根据模板ID查询规格列表  List specList = (List) redisTemplate.boundHashOps("specList").get(typeId);  map.put("specList", specList);  }  } |

## 显示品牌和规格列表-前端

把我们存进去的数据取出来

#### 获取品牌列表

修改页面 search.html，实现品牌列表

<**div class="type-wrap logo" ng-if="resultMap.brandList!=null"**>  
 <**div class="fl key brand"**>品牌</**div**>  
 <**div class="value logos"**>  
 <**ul class="logo-list"**>  
 <**li ng-repeat="brand in resultMap.brandList"**>  
 {{brand.text}}  
 </**li**>  
 </**ul**>  
 </**div**>  
 <**div class="ext"**>  
 <**a href="javascript:void(0);" class="sui-btn"**>多选</**a**>  
 <**a href="javascript:void(0);"**>更多</**a**>  
 </**div**>  
</**div**>

#### 获取规格列表

修改页面 search.html，实现规格列表

<**div class="type-wrap" ng-repeat="spec in resultMap.specList"**>  
 <**div class="fl key"**>{{spec.text}}</**div**>  
 <**div class="fl value"**>  
 <**ul class="type-list"**>  
 <**li ng-repeat="option in spec.options"**>  
 {{option.optionName}}  
 </**li**>  
 </**ul**>  
 </**div**>  
 <**div class="fl ext"**></**div**>  
</**div**>

# 搜索过滤条件构建(视频14~17) [撑握

## 搜索条件构建-需求与实现思路

查询条件有一个，但过滤条件有很多



这是查询条件：

$scope.searchEntity={};//定义搜索对象

上面的要添加一个：

$scope.searchEntity={item\_keywords:””,};//定义搜索对象



还有商品分类，品牌，以及网络制式

$scope.searchEntity={item\_keywords:””,cat:””,brand:””,spec:{网络：‘’}};//定义搜索对象

## 搜索条件构建-增加查询条件

#### 添加搜索项方法

修改 pinyougou-search-web 的 searchController.js

//搜索对象

$scope.searchMap={'keywords':'','category':'','brand':'','spec':{}};

//添加搜索项

$scope.addSearchItem=**function**(key,value){

**if**(key=='category' || key=='brand'){//如果点击的是分类或者是品牌

$scope.searchMap[key]=value;

|  |  |
| --- | --- |
| } | }**else**{  $scope.searchMap.spec[key]=value;  } |

#### 点击搜索项

修改 pinyougou-search-web 的 search.html ，为搜索面板添加点击事件

点击商品分类标签

<**span ng-repeat="category in resultMap.categoryList"**>  
 <**a href="#" ng-click="addSearchItem('category', category)"**>{{category}}</**a**>  
</**span**>

点击品牌标签

<**li ng-repeat="brand in resultMap.brandList"**>  
 <**a href="#" ng-click="addSearchItem('brand', brand.text)"**>{{brand.text}}</**a**>  
</**li**>

点击规格标签

<**li ng-repeat="option in spec.options"**>  
 <**a href="#" ng-click="addSearchItem(spec.text, option.optionName)"**>{{option.optionName}}</**a**>  
</**li**>

#### 显示面包屑

修改 pinyougou-search-web 的 search.html，用面包屑形式显示搜索条件

<ul class="fl sui-breadcrumb">搜索条件： </ul>

<ul class="tags-choose">

<li class="tag" ng-if="searchMap.category!=''">商品分类： {{searchMap.category}}<i

class="sui-icon icon-tb-close"></i></li>

<li class="tag" ng-if="searchMap.brand!=''">品牌： {{searchMap.brand}}<i

class="sui-icon icon-tb-close"></i></li>

<li class="tag" ng-repeat="(key,value) in searchMap.spec">{{key}}:{{value}}<i

class="sui-icon icon-tb-close"></i></li>

## 搜索条件构建-撤销查询条件

#### 撤销搜索项的方法

修改 pinyougou-search-web 工程 searchController.js

//移除复合搜索条件

$scope.removeSearchItem=**function**(key){  
 **if**(**'category'**==key||**'brand'**==key) {  
 $scope.**searchMap**[key]=**''**;  
 }**else** {  
 **delete** $scope.**searchMap**.**spec**[key];  
 }  
}

#### 页面调用方法

pinyougou-search-web 工程的 search.html

<**ul class="tags-choose"**>  
 <**li class="tag" ng-if="searchMap.category!=''" ng-click="removeSearchItem('category')"**>商品分类： {{searchMap.category}}<**i class="sui-icon icon-tb-close"**></**i**></**li**>  
 <**li class="tag" ng-if="searchMap.brand!=''" ng-click="removeSearchItem('brand')"**>品牌： {{searchMap.brand}}<**i class="sui-icon icon-tb-close"**></**i**></**li**>  
 <**li class="tag" ng-repeat="(key,value) in searchMap.spec" ng-click="removeSearchItem(key)"**>{{key}}:{{value}}<**i class="sui-icon icon-tb-close"**></**i**></**li**>  
</**ul**>

## 搜索条件构建-隐藏查询面板

当我们改变变量的时候，我们要隐藏面板或显示面板

#### 隐藏分类面板

修改 search.html

<**div class="type-wrap" ng-if="resultMap.categoryList!=null&&searchMap.category==''"**>  
 <**div class="fl key"**>商品分类</**div**>  
 <**div class="fl value"**>  
 <**span ng-repeat="category in resultMap.categoryList"**>  
 <**a href="#" ng-click="addSearchItem('category', category)"**>{{category}}</**a**>  
 </**span**>  
 </**div**>  
 <**div class="fl ext"**></**div**>  
</**div**>

#### 隐藏品牌面板

修改 search.html

<**div class="type-wrap logo" ng-if="resultMap.brandList!=null&&searchMap.brand==''"**>  
 <**div class="fl key brand"**>品牌</**div**>  
 <**div class="value logos"**>  
 <**ul class="logo-list"**>  
 <**li ng-repeat="brand in resultMap.brandList"**>  
 <**a href="#" ng-click="addSearchItem('brand', brand.text)"**>{{brand.text}}</**a**>  
 </**li**>  
 </**ul**>  
 </**div**>  
 <**div class="ext"**>  
 <**a href="javascript:void(0);" class="sui-btn"**>多选</**a**>  
 <**a href="javascript:void(0);"**>更多</**a**>  
 </**div**>  
</**div**>

#### 隐藏规格面板

修改 search.html

<**divclass="type-wrap" ng-repeat="spec in resultMap.specList" ng-if="searchMap.spec[spec.text]==null"**>  
 <**div class="fl key"**>{{spec.text}}</**div**>  
 <**div class="fl value"**>  
 <**ul class="type-list"**>  
 <**li ng-repeat="option in spec.options"**>  
 <**a href="#" ng-click="addSearchItem(spec.text, option.optionName)"**>{{option.optionName}}</**a**>  
 </**li**>  
 </**ul**>  
 </**div**>  
 <**div class="fl ext"**></**div**>  
</**div**>

### 提交查询

修改 searchController.js 在添加和删除筛选条件时自动调用搜索方法

//添加复合搜索条件

$scope.addSearchItem=**function**(key,value){

**if**(key=="category" || key=="brand"){//如果是分类或品牌

$scope.searchMap[key]=value;

}**else**{//否则是规格

$scope.searchMap.spec[key]=value;

}

$scope.search();//执行搜索

}

//移除复合搜索条件

$scope.removeSearchItem=**function**(key){

**if**(key=="category" || key=="brand"){//如果是分类或品牌

$scope.searchMap[key]=""; }**else**{//否则是规格

**delete** $scope.searchMap.spec[key];//移除此属性

}

$scope.search();//执行搜索

}

# 搜索过滤条件构建(视频18~21) [撑握

### 需求分析

根据上一步构建的查询条件，实现分类、品牌和规格的过滤查询



## 过滤查询-分类过滤

修改 pinyougou-search-service 工程的 SearchItemServiceImpl.java

**private** Map searchList(Map searchMap){  
 *//.......  
 //1.1 关键字查询......  
 //1.2 按分类筛选* **if**(!**""**.equals(searchMap.get(**"category"**))){  
 Criteria filterCriteria=**new** Criteria(**"item\_category"**).is(searchMap.get(**"category"**));  
 FilterQuery filterQuery=**new** SimpleFilterQuery(filterCriteria);  
 query.addFilterQuery(filterQuery);  
 }  
 *//高亮显示处理.....*}

## 过滤查询-品牌过滤

修改 pinyougou-search-service 工程的 SearchItemServiceImpl.java

**private** Map searchList(Map searchMap) {  
 *//.......  
 //1.1 关键字查询  
 //.......  
 //1.2 按分类筛选  
 //.......  
 //1.3 按品牌筛选* **if** (!**""**.equals(searchMap.get(**"brand"**))) {  
 Criteria filterCriteria = **new** Criteria(**"item\_brand"**).is(searchMap.get(**"brand"**));  
  
 FilterQuery filterQuery = **new** SimpleFilterQuery(filterCriteria);  
 query.addFilterQuery(filterQuery);  
 }  
 *//高亮显示处理...............*}

## 过滤查询-规格过滤

实现思路：规格有多项，需要循环过滤。循环规格查询条件，根据 key 得到域名城，根据 value 设置过滤条件。

修改 pinyougou-search-service 工程的 SearchItemServiceImpl.java

**private** Map searchList(Map searchMap){  
 *//......  
 //1.1 关键字查询  
 //....  
 //1.2 按分类筛选  
 //.....  
 //1.3 按品牌筛选  
 //......  
 //1.4 过滤规格* **if**(searchMap.get(**"spec"**)!=**null**){  
 Map<String,String> specMap= (Map) searchMap.get(**"spec"**);  
 **for**(String key:specMap.keySet() ){  
 Criteria filterCriteria=**new** Criteria(**"item\_spec\_"**+key).is( specMap.get(key) );  
 FilterQuery filterQuery=**new** SimpleFilterQuery(filterCriteria);  
 query.addFilterQuery(filterQuery);  
 }  
 }  
 *//高亮显示处理.....* }

## 过滤查询-根据分类查询品牌和规格列表

**public** Map<String, Object> search(Map searchMap) {  
 Map<String, Object> map = **new** HashMap<>();  
 *//1.按关键字查询（高亮显示）  
 //......  
 //2.根据关键字查询商品分类  
 //......  
 //3.查询品牌和规格列表* String categoryName = (String) searchMap.get(**"category"**);  
 **if** (!**""**.equals(categoryName)) {*//如果有分类名称* map.putAll(searchBrandAndSpecList(categoryName));  
 } **else** {*//如果没有分类名称，按照第一个查询* **if** (categoryList.size() > 0) {  
 map.putAll(searchBrandAndSpecList(categoryList.get(0)));  
 }  
 }  
 **return** map;  
}

# 总结：

1. 在原始的查询基础上添加高亮
2. 观察我们的查询块：商品分类约定了品牌和规格参数
3. 商品分类是分组查询出来的
4. 商品分类查询出来提是商品分类名称，用名称找商品分类的ID，用ID找相关联的模版，用模版找关联的品牌
5. 把这些数据收集起来用于过滤查询。