品优购电商系统开发

第 13 章

搜索解决方案-Solr [2]

传智播客.黑马程序员

课程目标

目标 1：完成 solr 环境安装、中文分析器和业务域的配置

目标 2：会使用 Spring Data Solr 完成增删改查操作

目标 3：完成批量数据导入功能

目标 4：完成按关键字搜索功能

1.品优购-高亮显示

1.1 需求分析

将用户输入的关键字在标题中以红色的字体显示出来，就是搜索中常用的高亮显示.

1.2 后端代码

修改服务层代码 ItemSearchServiceImpl.java

创建私有方法，用于返回查询列表的结果（高亮）

```
    /**
     * 根据关键字搜索列表
     * @param keywords
     * @return
     */
    private Map searchList(Map searchMap){
        Map map=new HashMap();
        HighlightQuery query=new SimpleHighlightQuery();
        HighlightOptions highlightOptions=new
HighlightOptions().addField("item_title");//设置高亮的域
        highlightOptions.setSimplePrefix("<em style='color:red'>");//高亮前缀
        highlightOptions.setSimplePostfix("</em>");//高亮后缀
        query.setHighlightOptions(highlightOptions);//设置高亮选项
        //按照关键字查询
        Criteria criteria=new
Criteria("item_keywords").is(searchMap.get("keywords"));
        query.addCriteria(criteria);
        HighlightPage<TbItem> page = solrTemplate.queryForHighlightPage(query,
TbItem.class);
        for(HighlightEntry<TbItem> h: page.getHighlighted()){//循环高亮入口集合
            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;
    }
```

修改 ItemSearchServiceImpl 的 search 方法，调用刚才编写的私有方法

```
    @Override
    public Map<String, Object> search(Map searchMap) {
        Map<String,Object> map=new HashMap<>();
        //1.查询列表
        map.putAll(searchList(searchMap));
        return map;
    }
```

1.3 前端代码

我们测试后发现高亮显示的 html 代码原样输出，这是 angularJS 为了防止 html 攻击采取的安全机制。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。

因为这个功能具有一定通用性，我们可以通过 angularJS 的过滤器来简化开发，这样只写一次，调用的时候就非常方便了，看代码：

（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);

    }

}]);
```

（2）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"></div>
```

|就是竖线，看起来有点斜是因为字体原因。

2.条件查询

2.1 需求分析

点击搜索面板上的分类、品牌和规格，实现查询条件的构建。查询条件以面包屑的形式显示。

当面包屑显示分类、品牌和规格时，要同时隐藏搜索面板对应的区域。

用户可以点击面包屑上的 x 撤销查询条件。撤销后显示搜索面包相应的区域。

2.2 添加搜索项

2.2.1 添加搜索项方法

修改 pinyougou-search-web 的 searchController.js

```
$scope.searchMap={'keywords':'','category':'','brand':'','spec':{}};//搜索对象

//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;

    }

}
```

2.2.2 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

```
<a href="#" ng-click="addSearchItem('category',category)">{{category}}</a>
```

点击品牌标签

```
<a href="#" ng-click="addSearchItem('brand',brand.text)">{{brand.text}}</a>
```

点击规格标签

```
<a href="#"  ng-click="addSearchItem(spec.text,pojo.optionName)">

{{pojo.optionName}}</a>
```

2.2.3 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>

</ul>
```

2.3 撤销搜索项

2.3.1 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(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
        }
    }
```

2.3.2 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>
```

2.4 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
    }
```

2.5 代码实现

2.5.1 分类和品牌过滤

修改 pinyougou-search-service 工程的 SearchItemServiceImpl.java ，在 search 方法中添加代码

```
@Override
public Map<String, Object> search(Map searchMap) {
    Map<String,Object> map=new HashMap<>();
    //1.查询商品信息
    Query query=new SimpleQuery();
    //1.1 添加关键字条件
    Criteria criteria1=new Criteria("item_keywords").is(searchMap.get("keywords"));
    query.addCriteria(criteria1);
    //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);

    }

    //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);

    }

    ScoredPage<TbItem> page1 = solrTemplate.queryForPage(query, TbItem.class);

    map.put("rows", page1.getContent());

    //2.根据关键字查询商品分类

    .......

    //3.查询品牌列表和规格列表

    .......

    return map;

}
```

2.5.2 规格过滤

修改 pinyougou-search-service 工程的 SearchItemServiceImpl.java ，在 search 方法中添加代码

```
@Override

public Map<String, Object> search(Map searchMap) {

    Map<String,Object> map=new HashMap<>();

    //1.查询商品信息
```

```
Query query=new SimpleQuery();

//1.1 添加关键字条件

......

//1.2 过滤分类

......

//1.3 过滤品牌

......

//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);

        }

}

ScoredPage<TbItem> page = solrTemplate.queryForPage(query, TbItem.class);

map.put("rows", page.getContent());

//2.根据关键字查询商品分类

......

//3.查询品牌列表

......

return map;
```

```
}
```

3.按价格区间筛选

3.1 需求分析

点击搜索面板上的价格区间，实现按价格筛选

3.2 前端代码

3.2.1 修改前端控制层

（1）修改 pinyougou-search-web 的 searchController.js　搜索条件的定义

```
$scope.searchMap={'keywords':'','category':'','brand':'','spec':{},'price':'' };//搜索条件封装对象
```

（2）修改 pinyougou-search-web 的 searchController.js　添加搜索项和删除搜索项的方法

```
    //添加搜索项
    $scope.addSearchItem=function(key,value){
        if(key=='category' || key=='brand' || key=='price'){//如果点击的是分类或品牌
            $scope.searchMap[key]=value;
        }else{//如果是规格
            $scope.searchMap.spec[key]=value;
        }
        $scope.search();
    }
```

```
    //移除复合搜索条件
    $scope.removeSearchItem=function(key){
        if(key=="category" ||  key=="brand" || key=='price'){//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();
    }
```

3.2.2 修改页面

（1）修改页面 search.html ,在标签上调用方法

```
<ul class="type-list" ">
    <li>
        <a ng-click="addSearchItem('price','0-500')">0-500 元</a>
    </li>
    <li>
        <a ng-click="addSearchItem('price','500-1000')">500-1000 元</a>
    </li>
    <li>
        <a ng-click="addSearchItem('price','1000-1500')">1000-1500 元</a>
    </li>
    <li>
        <a ng-click="addSearchItem('price','1500-2000')">1500-2000 元</a>
```

```
    </li>
    <li>
        <a ng-click="addSearchItem('price','2000-3000')">2000-3000 元 </a>
    </li>
    <li>
        <a ng-click="addSearchItem('price','3000-0')">3000 元以上</a>
    </li>
</ul>
```

（2）修改 search.html，增加面包屑

```
<li class="tag" ng-if="searchMap.price!=''" ng-click="removeSearchItem('price')">价格:
{{searchMap.price}}<i  class="sui-icon icon-tb-close"></i></li>
```

3.3 后端代码

修改 pinyougou-search-service 的 ItemSearchServiceImpl.java

```
    @Override
    public Map<String, Object> search(Map searchMap) {
        ......
        //1.1 按关键字搜索
        //1.2 按分类筛选
        //1.3 按品牌筛选
        //1.4 按规格筛选
        //1.5 按价格筛选
        if(!"".equals(searchMap.get("price"))){
            String[] price = ((String) searchMap.get("price")).split("-");
```

```
            if(!price[0].equals("0")){//如果区间起点不等于 0
                Criteria filterCriteria=new
Criteria("item_price").greaterThanEqual(price[0]);
                FilterQuery filterQuery=new SimpleFilterQuery(filterCriteria);
                query.addFilterQuery(filterQuery);
            }
            if(!price[1].equals("0")){//如果区间终点不等于 0
                Criteria filterCriteria=new
Criteria("item_price").lessThanEqual(price[1]);
                FilterQuery filterQuery=new SimpleFilterQuery(filterCriteria);
                query.addFilterQuery(filterQuery);
            }
        }
      ......
}
```

4.排序

4.1 按价格排序

实现价格的排序（升降序可切换）

4.1.1 后端代码

修改 pinyougou-search-service 的 ItemSearchServiceImpl.java 添加排序的代码

```
/**
 * 根据关键字搜索列表
 * @param keywords
 * @return
 */
private Map searchList(Map searchMap){
    ........
    //1.7 排序
    String sortValue= (String) searchMap.get("sort");//ASC  DESC
    String sortField= (String) searchMap.get("sortField");//排序字段
    if(sortValue!=null && !sortValue.equals("")){
        if(sortValue.equals("ASC")){
            Sort sort=new Sort(Sort.Direction.ASC, "item_"+sortField);
            query.addSort(sort);
        }
        if(sortValue.equals("DESC")){
            Sort sort=new Sort(Sort.Direction.DESC, "item_"+sortField);
            query.addSort(sort);
        }
    }
    //高亮显示处理
    ......
    return map;
```

```
    }
```

4.1.2 前端代码

（1）修改 searchController.js 的 searchMap， 增加排序

```
$scope.searchMap={'keywords':'','category':'','brand':'','spec':{},'price':'','page
No':1,'pageSize':40 ,'sortField':'','sort':'' };//搜索对象
```

（2）修改 searchController.js ，增加方法实现查询

```
    //设置排序规则
    $scope.sortSearch=function(sortField,sort){
        $scope.searchMap.sortField=sortField;
        $scope.searchMap.sort=sort;
        $scope.search();
    }
```

（3）修改页面 search.html

```
<div class="navbar-inner filter">
    <ul class="sui-nav">
        <li class="active">
            <a href="#" ng-click="sortSearch('','')">综合</a>
        </li>
        <li>
            <a href="#">销量</a>
        </li>
        <li>
            <a href="#">新品</a>
```

```
        </li>
        <li>
            <a href="#">评价</a>
        </li>
        <li>
            <a href="#" ng-click="sortSearch('price','ASC')">价格↑</a>
        </li>
        <li>
            <a href="#" ng-click="sortSearch('price','DESC')">价格↓</a>
        </li>
    </ul>
</div>
```

4.2 按上架时间排序

4.2.1 增加域定义

修改 solrhome 的 schema.xml 添加域定义

```
<field name="item_updatetime" type="date" indexed="true" stored="true" />
```

4.2.2 修改实体类

为 updatetime 属性添加注解

```
    @Field("item_updatetime")
    private Date updateTime;
```

4.2.3 重新运行导入程序

重新启动 solr

安装 pinyougou-pojo

重新运行 pinyougou-solr-util

4.2.4 修改页面

修改 search.html

```
<li>
    <a href="#" ng-click="sortSearch('updatetime','DESC')">新品</a>
</li>
```

4.3 按销量排序（实现思路）

（1）增加域 item_salecount 用于存储每个 SKU 的销量数据

（2）编写定时器程序，用于更新每个 SKU 的销量数据（查询近 1 个月的销量数据，不是累计数据）

（3）定时器每天只需执行一次，可以设定为凌晨开始执行。

定时器可以使用 spring task 技术来实现，学员们自行百度。

4.4 按评价排序（实现思路）

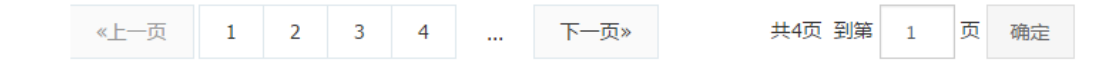
与按销量排序思路基本相同，有一个细节需要注意：

评论分为好评、中评、差评，我们不能简单地将评论数相加，而是应该根据每种评论加权进行统计。比如好评的权重是 3 ，中评的权重是 1，而差评的权重是 -3，这样得出的是评价的综合得分。

5.搜索结果分页

5.1 需求分析

在上述功能基础上实现分页查询



5.2 后端代码

5.2.1 分页逻辑

修改 pinyougou-search-service 工程 ItemSearchServiceImpl.java

```
@Override
public Map<String, Object> search(Map searchMap) {
    .......
    //1.6 分页查询
    Integer pageNo= (Integer) searchMap.get("pageNo");//提取页码
    if(pageNo==null){
        pageNo=1;//默认第一页
    }
    Integer pageSize=(Integer) searchMap.get("pageSize");//每页记录数
    if(pageSize==null){
        pageSize=20;//默认 20
    }
    query.setOffset((pageNo-1)*pageSize);//从第几条记录查询
```

```
        query.setRows(pageSize);

        ScoredPage<TbItem> page = solrTemplate.queryForPage(query,TbItem.class);

        map.put("rows", page.getContent());

        map.put("totalPages", page.getTotalPages());//返回总页数

        map.put("total", page.getTotalElements());//返回总记录数

        ........

}
```

5.3 前端代码

5.3.1 构建分页标签

（1）修改 searchController.js 实现页码的构建

```
    //搜索

    $scope.search=function(){

        searchService.search( $scope.searchMap ).success(

            function(response){

                $scope.resultMap=response;//搜索返回的结果

                //构建分页栏(totalPages 为总页数)

                $scope.resultMap.pageLabel=[];//新增分页栏属性

                for(var i=1;i<=$scope.resultMap.totalPages;i++){

                    $scope.resultMap.pageLabel.push(i);

                }

            }

        );
```

```
    }
```

（2）修改 search.html，循环产生页码

```
<ul>
    <li class="prev disabled">
        <a href="#">«上一页</a>
    </li>
    <li ng-repeat="p in resultMap.pageLabel">
        <a href="#" >{{p}}</a>
    </li>
    <li class="dotted"><span>...</span></li>
    <li class="next">
        <a href="#">下一页»</a>
    </li>
</ul>
```

5.3.2 提交页码查询

（1）如果我们需要修改默认页码和每页记录数，可以修改 searchController.js 的 searchMap，为搜索对象添加属性。

```
$scope.searchMap={'keywords':'','category':'','brand':'','spec':{},'price':'',
    'pageNo':1,'pageSize':40 };//搜索条件封装对象
```

（2）在 searchController.js 增加方法，修改页码执行查询

```
    //根据页码查询
```

```
    $scope.queryByPage=function(pageNo){

        //页码验证

        if(pageNo<1 || pageNo>$scope.resultMap.totalPages){

            return;

        }

        $scope.searchMap.pageNo=pageNo;

        $scope.search();

    }
```

（3）修改页码调用方法

```
<div class="sui-pagination pagination-large">

    <ul>

        <li class="prev disabled">

            <a href="#" ng-click="queryByPage(searchMap.pageNo-1)">«上一页</a>

        </li>

        <li ng-repeat="p in resultMap.pageLabel">

            <a href="#" ng-click="queryByPage(p)">{{p}}</a>

        </li>

        <li class="dotted"><span>...</span></li>

        <li class="next">

            <a href="#" ng-click="queryByPage(searchMap.pageNo+1)">下一页»</a>

        </li>

    </ul>

    <div><span>共{{resultMap.totalPages}}页 </span>
```

```
    <span>  到第 <input type="text" class="page-num" ng-model="pageNo">页

    <button class="page-confirm"  ng-click="queryByPage(pageNo)" >确定
</button></span></div>

</div>
```

（4）修改 search 方法，在执行查询前，转换为 int 类型，否则提交到后端有可能变成字符串

```
//搜索

    $scope.search=function(){

        $scope.searchMap.pageNo= parseInt($scope.searchMap.pageNo) ;

        .......

    }
```

5.3.3 页码不可用样式

修改 searchController.js 增加方法

```
    //判断当前页为第一页

    $scope.isTopPage=function(){

        if($scope.searchMap.pageNo==1){

            return true;

        }else{

            return false;

        }

    }



    //判断当前页是否未最后一页

    $scope.isEndPage=function(){
```

```
        if($scope.searchMap.pageNo==$scope.resultMap.totalPages){
            return true;
        }else{
            return false;
        }
    }
```

（2）修改页面

```
<ul>
    <li class="prev {{isTopPage()?'disabled':''}}">
        <a href="#" ng-click="queryByPage(searchMap.pageNo-1)">«上一页</a>
    </li>
    <li ng-repeat="p in resultMap.pageLabel">
        <a href="#" ng-click="queryByPage(p)">{{p}}</a>
    </li>
    <li class="dotted"><span>...</span></li>
    <li class="next {{isEndPage()?'disabled':''}}">
        <a href="#" ng-click="queryByPage(searchMap.pageNo+1)">下一页»</a>
    </li>
</ul>
```

5.4 搜索起始页码处理

测试：如果我们先按照“手机”关键字进行搜索，得出的页数是 19 页，然后我们点击第 18 页进行查询，然后我们再根据“三星”关键字搜索，会发现没有结果显示。是因为当前页仍然为 18，而三星的结果只有 4 页，所以无法显示。我们需要在每次点击查询时将页码设置为 1 。

修改搜索按钮，调用搜索前将起始页码设置为 1

```
<button class="sui-btn btn-xlarge btn-danger" ng-click="searchMap.pageNo=1;search()"
type="button">搜索</button>
```