第5章 搜索解决方案-3 分页与排序

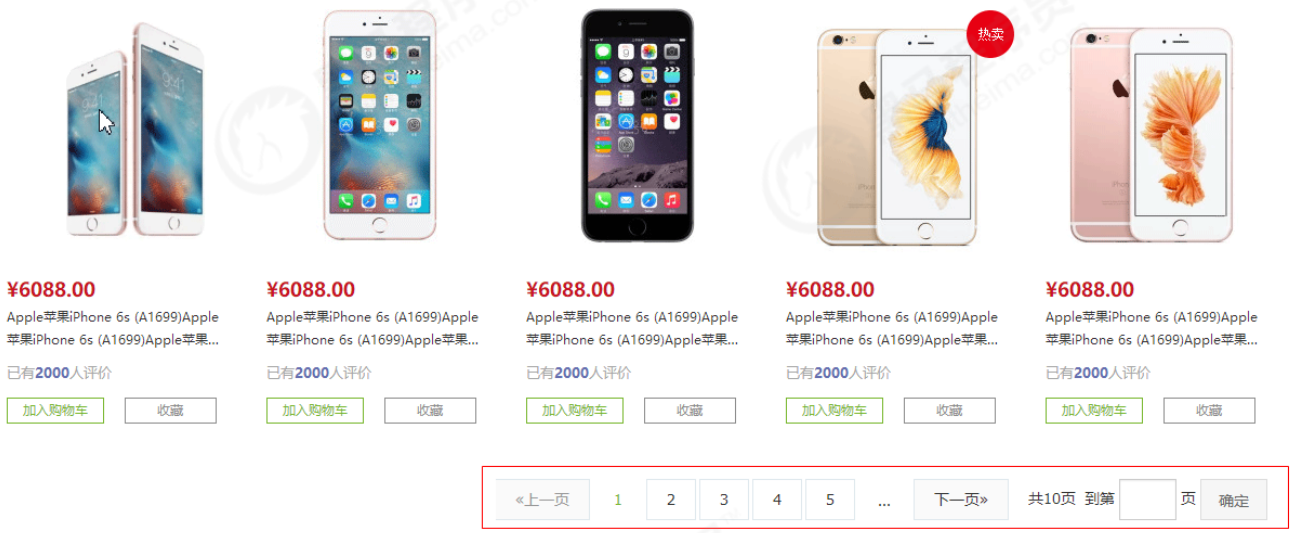
学习目标：

- 完成搜索分页功能
- 完成搜索排序功能
- 完成搜索高亮功能

1 搜索分页

1.1 需求分析

实现搜索分页功能，每页显示30条记录。实现商品列表下方分页栏。



1.2 分页语法与代码

1.2.1 分页语法

需求：每页显示30条记录，查询第3页内容。分页语法如下：

```
GET sku/_search
{
  "from": 60,
  "size": 30
}
```

1.2.2 分页代码

需求：每页显示30条记录，查询第3页内容。分页代码如下：

```
searchSourceBuilder.size(30);
searchSourceBuilder.from(60);
```

1.3 代码实现

1.3.1 分页查询逻辑

实现思路：前端向后端传递参数 pageNo （页码）

（1）修改SkuSearchServiceImpl的search方法，添加代码

```
//分页
Integer pageNo = Integer.parseInt(searchMap.get("pageNo"));//页码
Integer pageSize = 30;//页大小
//起始记录下标
int fromIndex = (pageNo - 1) * pageSize;
searchSourceBuilder.from(fromIndex);//开始索引
searchSourceBuilder.size(pageSize);//页大小
```

（2）修改SearchController的search方法，添加代码

```
//接受页码处理,如果不传递页码默认为第1页
if(searchMap.get("pageNo")==null){
    searchMap.put("pageNo","1");
}
```

1.3.2 页码渲染

实现思路：后端逻辑根据总记录数和每页记录数计算总页数，模板根据总页数渲染生成页码。

（1）修改SkuSearchServiceImpl的search方法，添加代码

```
//计算总页数
long totalCount = searchHits.getTotalHits();//总记录数
long pageCount = (totalCount % pageSize == 0) ? totalCount / pageSize :
(totalCount / pageSize + 1);
resultMap.put("totalPages", pageCount);
```

（2）修改SearchController的search方法，添加代码

```
int pageNo =Integer.parseInt(searchMap.get("pageNo"))  ;//当前页
model.addAttribute("pageNo",pageNo);
```

（3）修改search.html页码部分

```
<li th:class="${page==pageNo?'active':''}"
th:each="page:${#numbers.sequence(1,result.totalPages)}">
    <a
th:href="${#strings.replace(url,'&pageNo='+searchMap.pageNo,'&pageNo='+pa
ge )}" th:text="${page}">
    </a>
</li>
```

numbers.sequence（开始索引，截至索引）可以产生从开始索引到截至索引的数组

1.3.3 页码数量控制

我们刚才的代码显示页码会全部显示，如果页码很多，也不利于显示，所以，我们可以显示当前页前后的5个页码即可

（1）修改SearchController的search方法，添加代码

```
Long totalPages=  (Long) result.get("totalPages");//得到总页数
int startPage=1;//开始页码
int endPage =totalPages.intValue();//截至代码
if(totalPages>5){
    startPage=pageNo-2;
    if(startPage<1){
        startPage=1;
    }
    endPage=startPage+4;
}
model.addAttribute("startPage",startPage);
model.addAttribute("endPage",endPage);
```

（2）修改search.html 的页码部分

```
<li class="dotted" th:if="${startPage>1}"><span>...</span></li>
<li th:class="${page==pageNo?'active':''}"
th:each="page:${#numbers.sequence(startPage,endPage)}">
    <a
th:href="${#strings.replace(url,'&pageNo='+searchMap.pageNo,'&pageNo='+pa
ge )}" th:text="${page}"></a>
</li>
<li class="dotted" th:if="${endPage<result.totalPages}"><span>...</span>
</li>
```

1.3.4 上一页与下一页

修改search.html 上一页

```
<li class="prev" th:if="${pageNo>1}">
    <a th:href="${#strings.replace(url,'&pageNo='+pageNo ,'&pageNo='+
(pageNo-1)  )}">«</a>
</li>
<li class="prev disabled" th:if="${pageNo<=1}">
    <a>«</a>
</li>
```

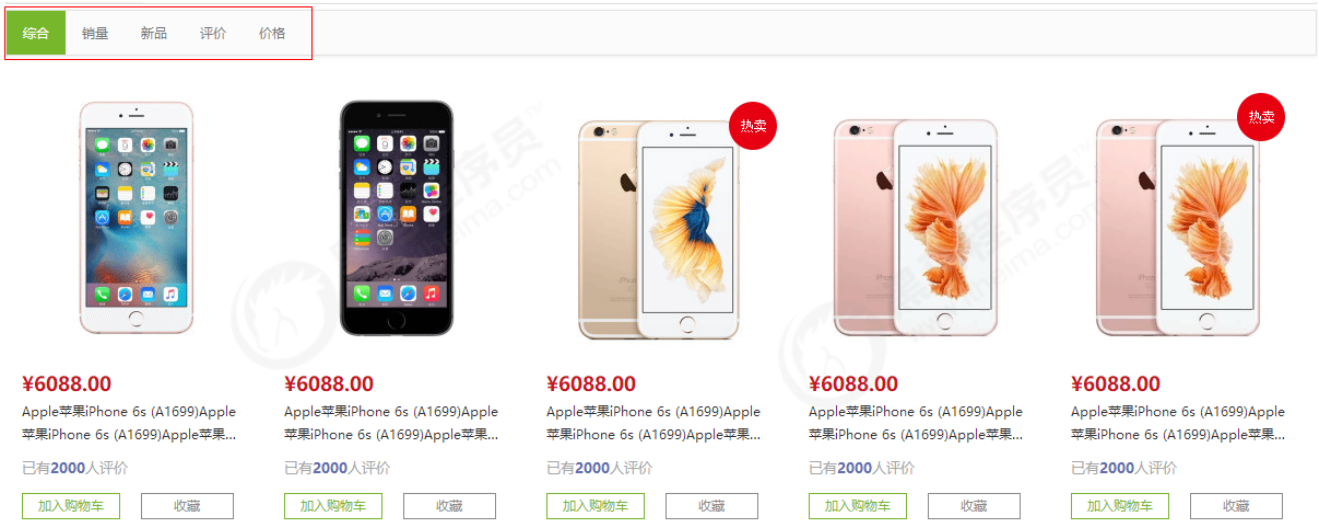
下一页

```
<li class="next" th:if="${pageNo<result.totalPages }">
    <a th:href="${#strings.replace(url,'&pageNo='+pageNo, '&pageNo='+
(pageNo+1)  )}">»</a>
</li>
<li class="next disabled" th:if="${pageNo>=result.totalPages }">
    <a>»</a>
</li>
```

2 搜索排序

2.1 需求分析

实现按销量排序、新品排序、评论排序和价格排序



2.2 排序语法与代码

2.2.1 排序语法

需求：按价格升序排序，语法如下：

```
GET sku/_search
{
  "sort": [
    {
      "price": {
        "order": "asc"
      }
    }
  ]
}
```

如果是降序，则指定order为desc

2.2.2 排序代码

需求：按价格升序排序，代码如下：

```
searchSourceBuilder.sort("price", SortOrder.ASC);
```

如果是降序，则指定为SortOrder.DESC

2.3 代码实现

2.3.1 销量排序

（1）修改SearchController的search方法，添加代码

```
//排序参数容错处理
if(searchMap.get("sort")==null){
    searchMap.put("sort","");
}
if(searchMap.get("sortOrder")==null){
    searchMap.put("sortOrder","DESC");
}
```

（2）修改SkuSearchServiceImpl的searchList方法，添加代码

```
//排序
String sort = searchMap.get("sort");//排序字段
String sortOrder = searchMap.get("sortOrder");//排序规则
if(!"".equals(sort)){
    searchSourceBuilder.sort(sort, SortOrder.valueOf(sortOrder));
}
```

（3）修改search.html 的排序标签

```
<li th:class="${searchMap.sort==''?'active':''}">
    <a
th:href="${#strings.replace(url,'&sortOrder='+searchMap.sortOrder+'&sort=
'+searchMap.sort,'&sortOrder=DESC&sort=' )}">综合</a>
</li>
<li th:class="${searchMap.sort=='saleNum'?'active':''}">
    <a
th:href="${#strings.replace(url,'&sortOrder='+searchMap.sortOrder+'&sort=
'+searchMap.sort,'&sortOrder=DESC&sort=saleNum' )}">销量</a>
</li>
```

2.3.2 新品排序

```
<li th:class="${searchMap.sortField=='createTime'}?'active':''">
    <a th:href="|${#strings.replace(url,'&sortRule='+
searchMap.get('sortRule') +'&sortField='+searchMap.get('sortField')
,'')}&sortRule=DESC&sortField=createTime|">新品</a>
</li>
```

2.3.3 评论排序

```
<li th:class="${searchMap.sortField=='commentNum'}?'active':''">
    <a th:href="|${#strings.replace(url,'&sortRule='+
searchMap.get('sortRule') +'&sortField='+searchMap.get('sortField')
,'')}&sortRule=DESC&sortField=commentNum|">评价</a>
</li>
```

2.3.4 价格排序

修改search.html

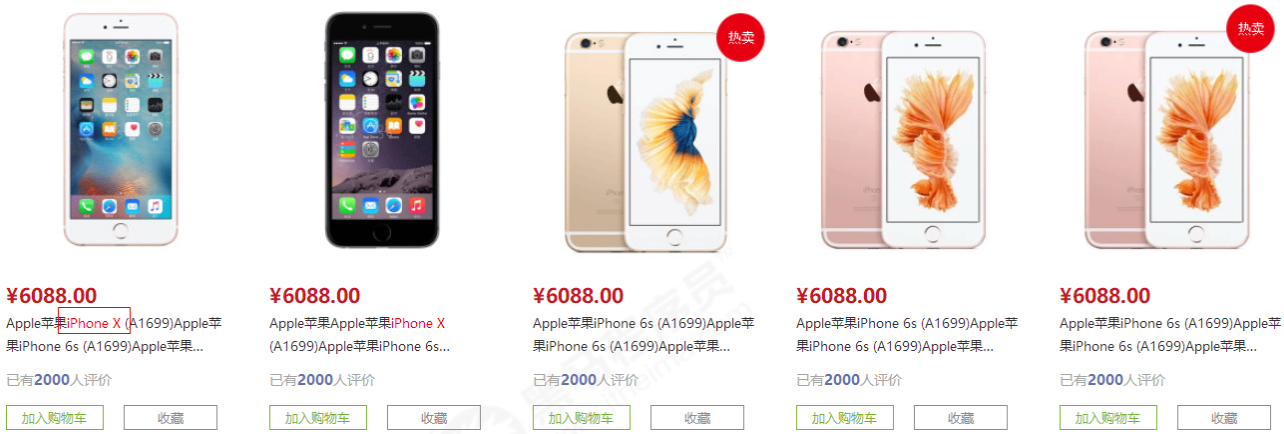
```
<li
th:class="${searchMap.sortField=='price'&&searchMap.sortRule=='ASC'}?'act
ive':''">
    <a th:href="|${#strings.replace(url,'&sortRule='+
searchMap.get('sortRule') +'&sortField='+searchMap.get('sortField')
,'')}&sortRule=ASC&sortField=price|">价格</a>
</li>
<li
th:class="${searchMap.sortField=='price'&&searchMap.sortRule=='DESC'}?'ac
tive':''">
    <a th:href="|${#strings.replace(url,'&sortRule='+
searchMap.get('sortRule') +'&sortField='+searchMap.get('sortField')
,'')}&sortRule=DESC&sortField=price|">价格</a>
</li>
```

3 搜索高亮

3.1 需求分析

所谓高亮，就是使用特别的样式修饰某字段中包含的搜索关键字。

需求：实现搜索高亮，商品名称使用红色显示搜索关键字。



3.2 高亮语法与代码

3.2.1 高亮语法

使用默认高亮显示来获取每个搜索命中title字段的高亮显示，在指定title字段的查询请求中包含高亮显示对象。

执行查询

```
GET /sku/doc/_search
{
  "query": {
    "match": {
      "name":"手机"
    }
  },
  "highlight":{
    "fields": {
      "name": {}
    }
  },
  "size": 2
}
```

返回查询结果：

```
{
  "took" : 1,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 9419,
    "max_score" : 3.689363,
    "hits" : [
      {
        "_index" : "sku",
        "_type" : "doc",
        "_id" : "34803399300",
        "_score" : 3.689363,
        "_source" : {
          "commentNum" : 0,
          "image" :
"https://m.360buyimg.com/mobilecms/s720x720_jfs/t1/2706/35/9961/98599/5bc
940f4Eecc2753b/22432d389910dec2.jpg!q70.jpg.webp",
          "brandName" : "Apple",
          "createTime" : "2019-04-30T16:00:00.000Z",
          "price" : 35800,
          "name" : "Apple 苹果 iPhone XR 手机 全网通4G手机 黑色 64GB",
          "saleNum" : 0,
          "categoryName" : "手机",
          "spec" : {
            "颜色" : "蓝色",
            "版本" : "64G"
          }
        },
        "highlight" : {
          "name" : [
            "Apple 苹果 iPhone XR <em>手机</em> 全网通4G<em>手机</em> 黑色
64GB"
          ]

        }
```

```
      },
      {
        "_index" : "sku",
        "_type" : "doc",
        "_id" : "34803424702",
        "_score" : 3.689363,
        "_source" : {
          "commentNum" : 0,
          "image" :
"https://m.360buyimg.com/mobilecms/s720x720_jfs/t1/2706/35/9961/98599/5bc
940f4Eecc2753b/22432d389910dec2.jpg!q70.jpg.webp",
          "brandName" : "Apple",
          "createTime" : "2019-04-30T16:00:00.000Z",
          "price" : 46400,
          "name" : "Apple 苹果 iPhone XR 手机 全网通4G手机 黑色 128GB",
          "saleNum" : 0,
          "categoryName" : "手机",
          "spec" : {
            "颜色" : "黄色",
            "版本" : "128G"
          }
        },
        "highlight" : {
          "name" : [
            "Apple 苹果 iPhone XR <em>手机</em> 全网通4G<em>手机</em> 黑色
128GB"
          ]
        }
      }
    ]
  }
}
```


自定义高亮 执行查询：

```
GET /sku/doc/_search
{
  "query": {
    "match": {
      "name":"手机"
    }
  },
  "highlight":{
    "fields": {
      "name": {
        "pre_tags":"<font style='color:red'>",
        "post_tags":"</font>"
      }
    }
  },
  "size": 2
}
```

返回结果（高亮部分）

```
"highlight" : {
   "name" : [
       "Apple 苹果 iPhone XR <font style='color:red'>手机</font> 全网通
4G<font style='color:red'>手机</font> 黑色 128GB"
   ]
}
```

3.2.2 高亮代码

高亮设置：

```
//设置高亮
HighlightBuilder highlightBuilder =  new HighlightBuilder();
highlightBuilder.field("name").preTags("<font
style='color:red'>").postTags("</font>");
searchSourceBuilder.highlighter(highlightBuilder);
```

获取高亮结果：

```
for(SearchHit hit:hits){
    //提取高亮内容
    Map<String, HighlightField> highlightFields =
hit.getHighlightFields();
    HighlightField highlightFieldName = highlightFields.get("name");

    Text[] fragments = highlightFieldName.fragments();
    String name = fragments[0].toString();
    System.out.println(name);
}
```

3.3 代码实现

（1）修改SkuSearchServiceImpl的search方法，在第一个代码块中添加高亮显示处理代码

```
//设置高亮
HighlightBuilder highlightBuilder =  new HighlightBuilder();
highlightBuilder.field("name").preTags("<font
style='color:red'>").postTags("</font>");
searchSourceBuilder.highlighter(highlightBuilder);
```

修改//2.1 商品列表部分代码

```
//2.1 商品列表
List<Map<String,Object>> resultList=new ArrayList<Map<String, Object>>();
for(SearchHit hit:hits){
    Map<String, Object> skuMap = hit.getSourceAsMap();
    //name高亮处理
    Map<String, HighlightField> highlightFields =
hit.getHighlightFields();
    HighlightField name = highlightFields.get("name");
    Text[] fragments = name.fragments();
    skuMap.put("name",fragments[0].toString());//用高亮的内容替换原内容
    resultList.add(skuMap);
}
resultMap.put("rows",resultList);
```

（3）修改模板中商品名称部分

```
<div class="attr">
    <a target="_blank" href="item.html"  th:utext="${sku.name}"></a>
</div>
```