课程介绍

- 编写爬虫抓取房源数据
- 开发搜索房源接口服务
- 整合前端开发实现搜索功能
- 优化搜索功能增加高亮和分页功能
- 热词推荐功能实现
- 拼音分词

1、WebMagic抓取数据

为了丰富我们的房源数据，所以我们采用WebMagic来抓取一些数据，目标网站是上海链家网。

English

WebMagic是一个简单灵活的Java爬虫框架。基于WebMagic，你可以快速开发出一个高效、易维护的爬虫。

1.1、引入依赖

依然在itcast-es工程中，编写爬虫相关的代码。

```
<dependency>
     <groupId>us.codecraft</groupId>
     <artifactId>webmagic-core</artifactId>
     <version>0.7.3</version>
</dependency>
<dependency>
    <groupId>us.codecraft</groupId>
    <artifactId>webmagic-extension</artifactId>
    <version>0.7.3</version>
</dependency>

<dependency>
    <groupId>commons-io</groupId>
    <artifactId>commons-io</artifactId>
    <version>2.6</version>
</dependency>
```

1.2、编写LianjiaPageProcessor

```java
package cn.itcast.es.wm;

import us.codecraft.webmagic.Page;
import us.codecraft.webmagic.Site;
import us.codecraft.webmagic.Spider;
import us.codecraft.webmagic.processor.PageProcessor;
import us.codecraft.webmagic.selector.Html;

public class LianjiaPageProcessor implements PageProcessor {

    private Site site = Site.me().setRetryTimes(3).setSleepTime(200);

    @Override
    public void process(Page page) {
        Html html = page.getHtml();
        page.addTargetRequests(html.css(".content__list--item--title a").links().all());

        page.putField("title", html.xpath("//div[@class='content clear w1150']/p/text()").toString());
        page.putField("rent", html.xpath("//p[@class='content__aside--title']/span/text()").toString());
        page.putField("type", html.xpath("//p[@class='content__article__table']/allText()").toString());
        page.putField("info", html.xpath("//div[@class='content__article__info']/allText()").toString());
        page.putField("img", html.xpath("//div[@class='content__article__slide__item']/img").toString());

        if (page.getResultItems().get("title") == null) {
            page.setSkip(true);

            //分页
            for (int i = 1; i <= 100; i++) {
                page.addTargetRequest("https://sh.lianjia.com/zufang/pg" + i);
            }
        }

    }

    @Override
    public Site getSite() {
        return site;
    }

    public static void main(String[] args) {
        Spider.create(new LianjiaPageProcessor())
```

```
                .addUrl("https://sh.lianjia.com/zufang/")
                .thread(5)
                .addPipeline(new MyPipeline())
                .run();
    }
}
```

1.3、编写MyPipeline

```
package cn.itcast.es.wm;

import com.fasterxml.jackson.databind.ObjectMapper;
import org.apache.commons.io.FileUtils;
import org.apache.commons.io.IOUtils;
import org.apache.commons.lang3.StringUtils;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.HttpClientBuilder;
import us.codecraft.webmagic.ResultItems;
import us.codecraft.webmagic.Task;
import us.codecraft.webmagic.pipeline.Pipeline;

import java.io.File;
import java.util.HashMap;
import java.util.Map;

public class MyPipeline implements Pipeline {

    private static final ObjectMapper MAPPER = new ObjectMapper();

    @Override
    public void process(ResultItems resultItems, Task task) {
        Map<String, Object> data = new HashMap<>();

        data.put("url", resultItems.getRequest().getUrl());
        data.put("title", resultItems.get("title"));//标题
        data.put("rent", resultItems.get("rent"));//租金

        String[] types = StringUtils.split(resultItems.get("type"), ' ');
        data.put("rentMethod", types[0]);//租赁方式
        data.put("houseType", types[1]);//户型，如：2室1厅1卫
        data.put("orientation", types[2]);//朝向

        String[] infos = StringUtils.split(resultItems.get("info"), ' ');
        for (String info : infos) {
            if (StringUtils.startsWith(info, "看房：")) {
                data.put("time", StringUtils.split(info, '：')[1]);
            } else if (StringUtils.startsWith(info, "楼层：")) {
                data.put("floor", StringUtils.split(info, '：')[1]);
            }
        }
```

```java

        String imageUrl = StringUtils.split(resultItems.get("img"), '"')[3];
        String newName = StringUtils
                .substringBefore(StringUtils
                        .substringAfterLast(resultItems.getRequest().getUrl(), "/"), ".") + ".jpg";


        try {
            this.downloadFile(imageUrl, new File("F:\\code\\images\\" + newName));
            data.put("image", newName);
            String json = MAPPER.writeValueAsString(data);
            FileUtils.write(new File("F:\\code\\data.json"), json + "\n", "UTF-8", true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
     * 下载文件
     *
     * @param url  文件url
     * @param dest 目标目录
     * @throws Exception
     */
    public void downloadFile(String url, File dest) throws Exception {
        HttpGet httpGet = new HttpGet(url);
        CloseableHttpResponse response = HttpClientBuilder.create().build().execute(httpGet);
        try {
            FileUtils.writeByteArrayToFile(dest, IOUtils.toByteArray(response.getEntity().getContent()));
        } finally {
            response.close();
        }
    }

}

```

1.4、开始抓取数据

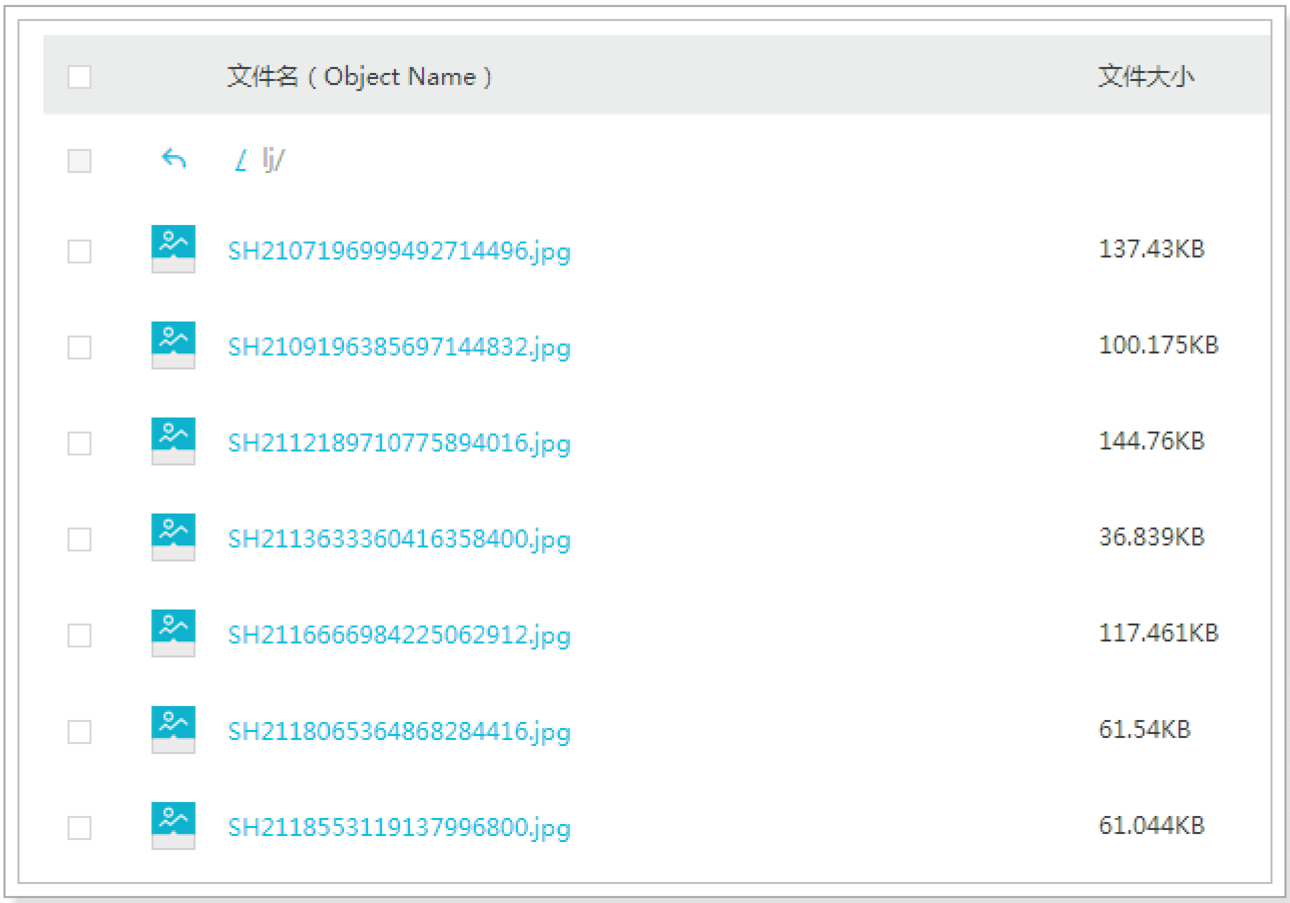
抓取的数据：

```
{"image":"SH2136963681764769792.jpg","orientation":"40㎡","houseType":"1室1厅1卫","rentMethod":"整租","time":"需提前预约","title":"整租 · 精装，可做两室的一房，2楼采光好安静卫生，居家舒适","rent":"4800","floor":"低楼层/6层","url":"https://sh.lianjia.com/zufang/SH2136963681764769792.html"}

{"image":"SH2118607017555279872.jpg","orientation":"68㎡","houseType":"2室1厅1卫","rentMethod":"整租","time":"需提前预约","title":"整租 · 长春新苑 2居室 7300","rent":"7300","floor":"高楼层/6层","url":"https://sh.lianjia.com/zufang/SH2118607017555279872.html"}

```

一共抓取到2010条房源数据。

1.5、将图片上传到OSS

```java
package cn.itcast.es;

import com.aliyun.oss.OSSClient;
import com.fasterxml.jackson.databind.JsonNode;
import com.fasterxml.jackson.databind.ObjectMapper;
import org.apache.commons.io.FileUtils;
import org.junit.Test;

import java.io.File;
import java.util.List;

public class TestOSS {

    @Test
    public void testOss() throws Exception{
        ObjectMapper mapper = new ObjectMapper();
        String endpoint = "http://oss-cn-qingdao.aliyuncs.com";
        String accessKeyId = "LTAIfC7fUsPj7Rfq";
        String accessKeySecret = "c2Vo3q1AmivtY8lxFnfsCfkO2c2HCk";
        String bucketName="itcast-haoke";
        String urlPrefix="http://itcast-haoke.oss-cn-qingdao.aliyuncs.com/";

        OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeySecret);

        List<String> lines = FileUtils.readLines(new File("F:\\code\\data.json"), "UTF-8");
        for (String line : lines) {
            JsonNode jsonNode = mapper.readTree(line);
            String image = jsonNode.get("image").asText();

            ossClient.putObject(bucketName, "lj/"+image, new File("F:\\code\\images\\"+image));
            System.out.println(image);
        }

    }
}

```



	文件名（Object Name）	文件大小
	/ lj/	
	SH2107196999492714496.jpg	137.43KB
	SH2109196385697144832.jpg	100.175KB
	SH2112189710775894016.jpg	144.76KB
	SH2113633360416358400.jpg	36.839KB
	SH2116666984225062912.jpg	117.461KB
	SH2118065364868284416.jpg	61.54KB
	SH2118553119137996800.jpg	61.044KB

1.6、将数据导入到Elasticsearch

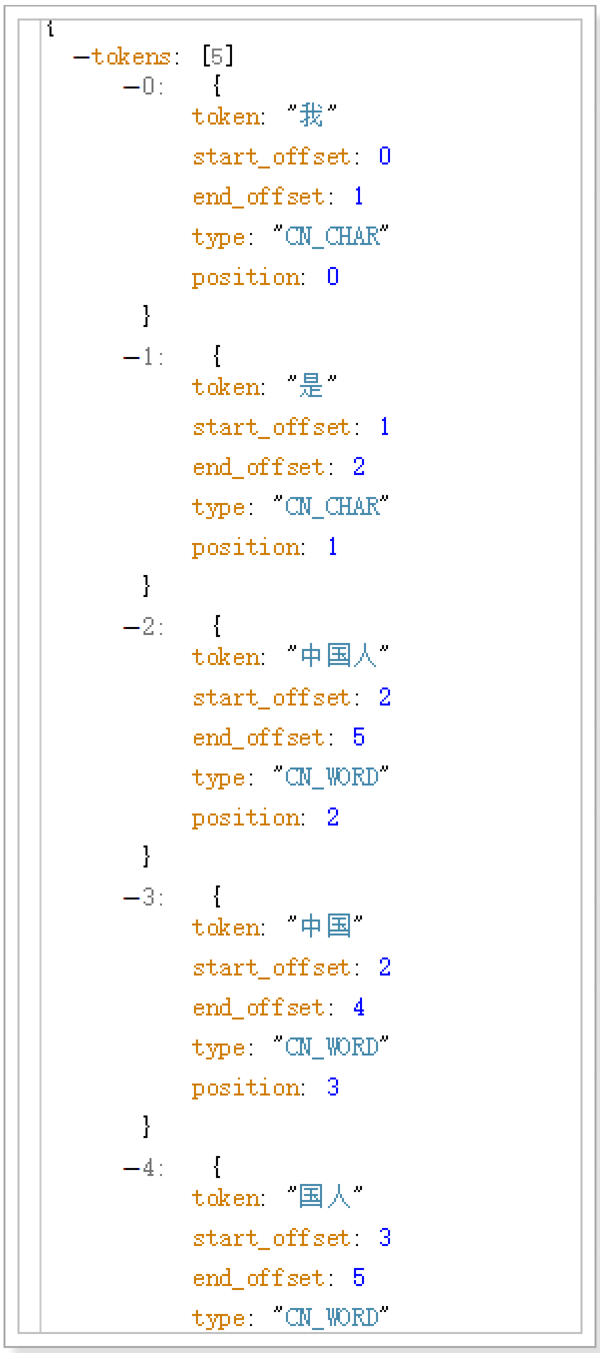
1.6.1、设置IK分词器

```
cd /haoke/es-cluster/ik
#将IK的zip压缩包解压到该目录

#停止、删除现有容器
docker stop es-node01 es-node02 es-node03
docker rm es-node01 es-node02 es-node03

#重新创建容器，注意ik目录的挂载
docker create --name es-node01 --net host -v /haoke/es-cluster/node01/elasticsearch.yml:/usr/share/elasticsearch/config/elasticsearch.yml -v /haoke/es-cluster/node01/jvm.options:/usr/share/elasticsearch/config/jvm.options -v /haoke/es-cluster/ik:/usr/share/elasticsearch/plugins/ik -v /haoke/es-cluster/node01/data:/usr/share/elasticsearch/data elasticsearch:6.5.4

docker create --name es-node02 --net host -v /haoke/es-cluster/node02/elasticsearch.yml:/usr/share/elasticsearch/config/elasticsearch.yml -v /haoke/es-cluster/node02/jvm.options:/usr/share/elasticsearch/config/jvm.options -v /haoke/es-cluster/ik:/usr/share/elasticsearch/plugins/ik -v /haoke/es-cluster/node02/data:/usr/share/elasticsearch/data elasticsearch:6.5.4

docker create --name es-node03 --net host -v /haoke/es-cluster/node03/elasticsearch.yml:/usr/share/elasticsearch/config/elasticsearch.yml -v /haoke/es-cluster/node03/jvm.options:/usr/share/elasticsearch/config/jvm.options -v /haoke/es-cluster/ik:/usr/share/elasticsearch/plugins/ik -v /haoke/es-cluster/node03/data:/usr/share/elasticsearch/data elasticsearch:6.5.4
```

测试：

```
POST http://192.168.1.7:9200/_analyze
{
    "analyzer": "ik_max_word",
    "text": "我是中国人"
}
```

结果：



```
{
  -tokens: [5]
    -0: {
        token: "我"
        start_offset: 0
        end_offset: 1
        type: "CN_CHAR"
        position: 0
      }
    -1: {
        token: "是"
        start_offset: 1
        end_offset: 2
        type: "CN_CHAR"
        position: 1
      }
    -2: {
        token: "中国人"
        start_offset: 2
        end_offset: 5
        type: "CN_WORD"
        position: 2
      }
    -3: {
        token: "中国"
        start_offset: 2
        end_offset: 4
        type: "CN_WORD"
        position: 3
      }
    -4: {
        token: "国人"
        start_offset: 3
        end_offset: 5
        type: "CN_WORD"
```

1.6.2、文档mapping

```
PUT http://192.168.1.7:9200/haoke

{
    "settings": {
        "index": {
            "number_of_shards": 6,
            "number_of_replicas": 1
        }
    },
    "mappings": {
        "house": {
            "dynamic": false,
            "properties": {
                "title": {
                    "type": "text",
                    "analyzer":"ik_max_word"
                },
                "image": {
                    "type": "keyword",
                    "index":false
                },
                "orientation": {
                    "type": "keyword",
                    "index":false
                },
                "houseType": {
                    "type": "keyword",
                    "index":false
                },
                "rentMethod": {
                    "type": "keyword",
                    "index":false
                },
                "time": {
                    "type": "keyword",
                    "index":false
                },
                "rent": {
                    "type": "keyword",
                    "index":false
                },
                "floor": {
                    "type": "keyword",
                    "index":false
                }
            }
        }
    }
}
```

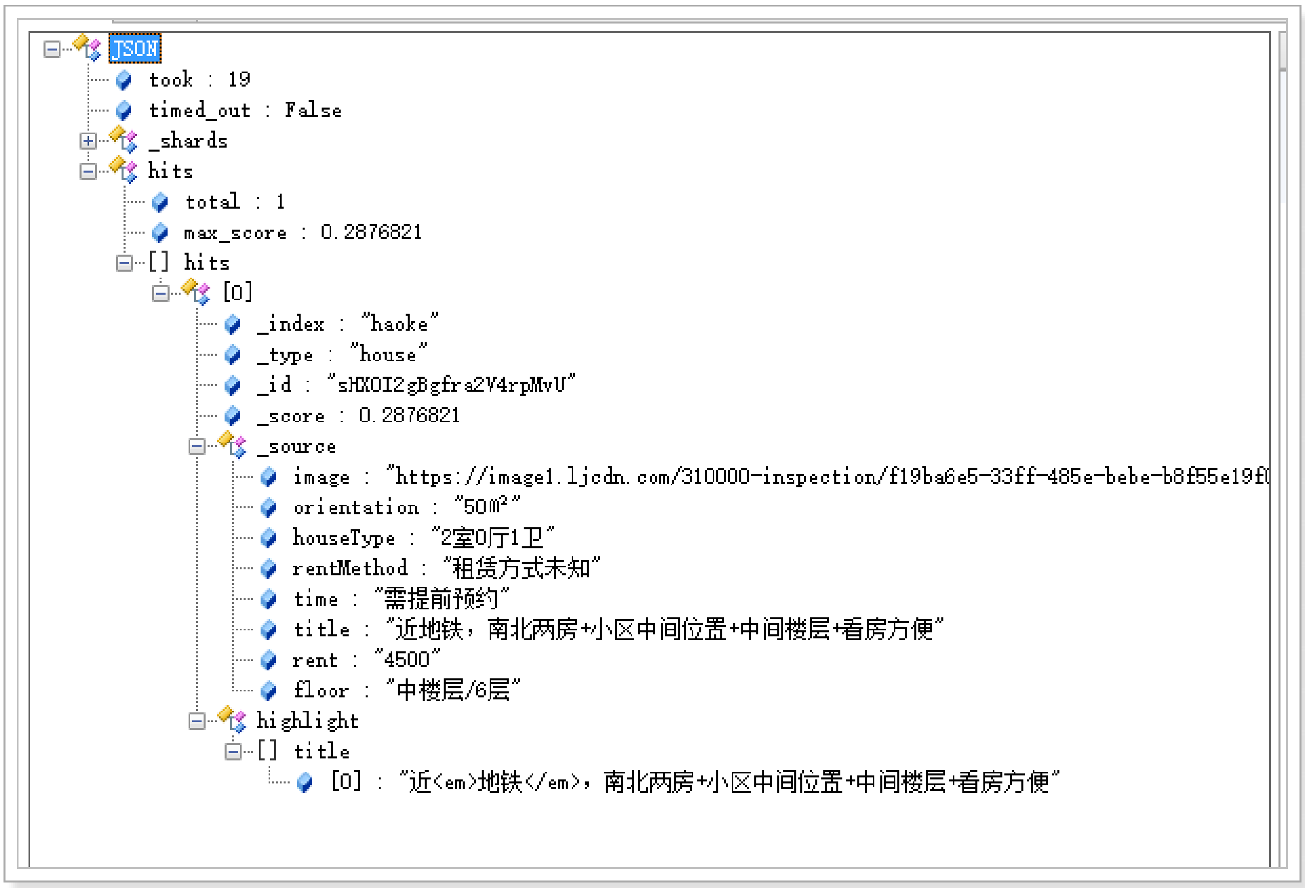
> 说明：
>
> - dynamic
>   - dynamic 参数来控制字段的新增
>   - true：默认值，表示允许选自动新增字段
>   - false：不允许自动新增字段，但是文档可以正常写入，但无法对字段进行查询等操作
>   - strict：严格模式，文档不能写入，报错
> - index
>   - index参数作用是控制当前字段是否被索引，默认为true，false表示不记录，即不可被搜索。

插入测试数据：

```
POST http://192.168.1.7:9200/haoke/house

{
    "image": "SH2137695162426728448.jpg",
    "orientation": "53㎡",
    "houseType": "2室1厅1卫",
    "rentMethod": "租赁方式未知",
    "time": "需提前预约",
    "title": "婚房装修，品牌家具，塘桥四号线地铁口精装两房，",
    "rent": "6200",
    "floor": "高楼层/6层",
    "url": "https://sh.lianjia.com/zufang/SH2137695162426728448.html"
}
```

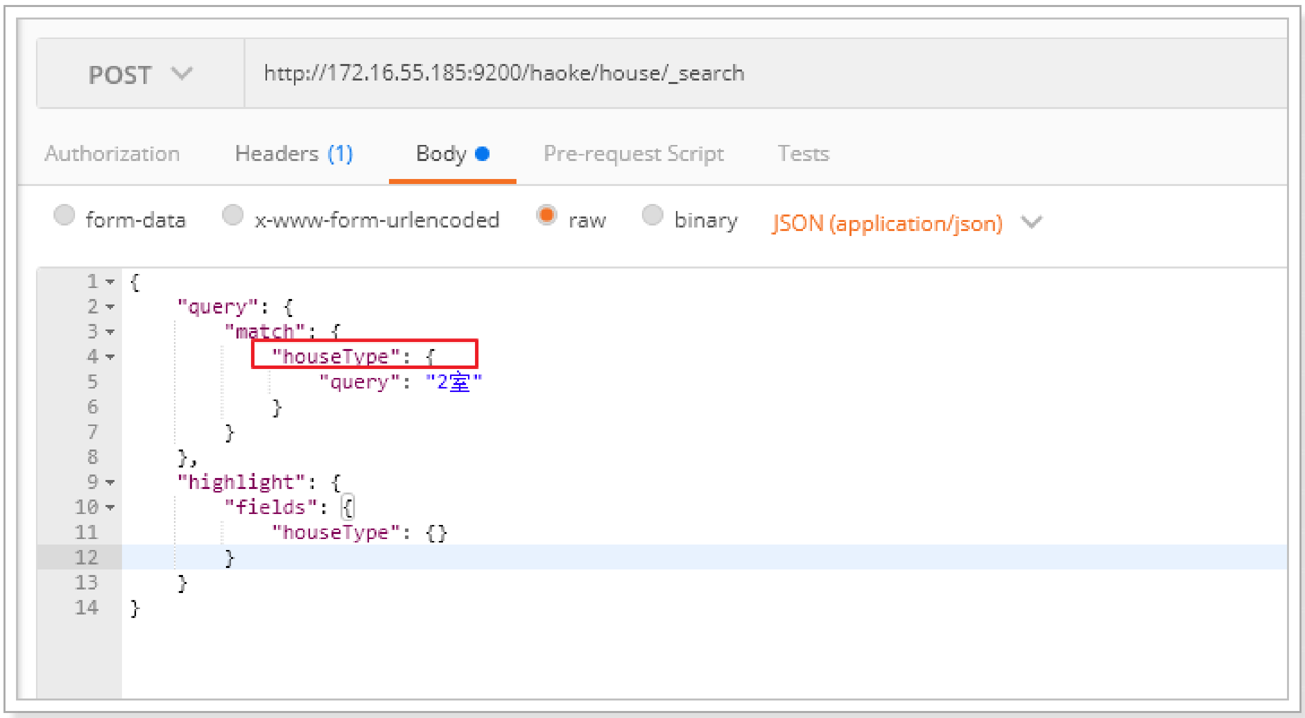
进行搜索测试：

```
POST http://192.168.1.7:9200/haoke/house/_search
{
    "query": {
        "match": {
            "title": {
                "query": "地铁"
            }
        }
    },
    "highlight": {
        "fields": {
            "title": {}
        }
    }
}
```



```
JSON
  took : 19
  timed_out : False
  _shards
  hits
    total : 1
    max_score : 0.2876821
    [] hits
      [0]
        _index : "haoke"
        _type : "house"
        _id : "sHXOI2gBgfra2V4rpMvU"
        _score : 0.2876821
        _source
          image : "https://image1.ljcdn.com/310000-inspection/f19ba6e5-33ff-485e-bebe-b8f55e19f
          orientation : "50㎡"
          houseType : "2室0厅1卫"
          rentMethod : "租赁方式未知"
          time : "需提前预约"
          title : "近地铁，南北两房+小区中间位置+中间楼层+看房方便"
          rent : "4500"
          floor : "中楼层/6层"
        highlight
          [] title
            [0] : "近<em>地铁</em>，南北两房+小区中间位置+中间楼层+看房方便"
```

如果使用其他字段搜索：

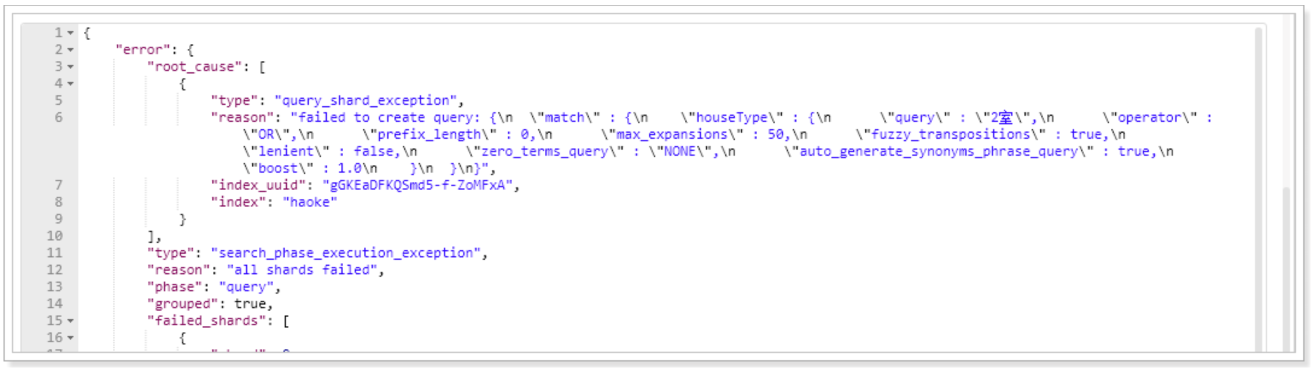


POST　http://172.16.55.185:9200/haoke/house/_search

Authorization　Headers (1)　Body ●　Pre-request Script　Tests

form-data　x-www-form-urlencoded　[✓ raw]　binary　JSON (application/json)

```
{
    "query": {
        "match": {
            "houseType": {
                "query": "2室"
            }
        }
    },
    "highlight": {
        "fields": {
            "houseType": {}
        }
    }
}
```

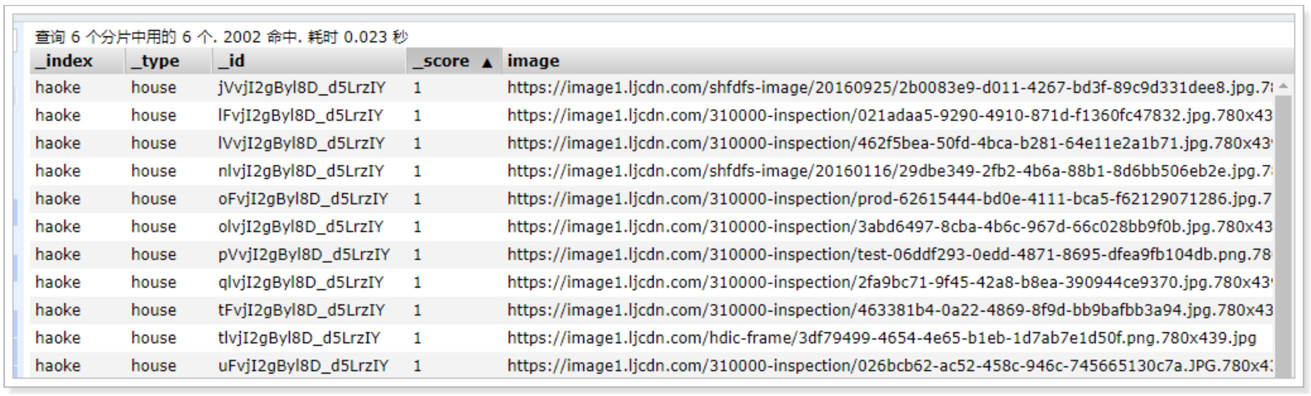


```
{
    "error": {
        "root_cause": [
            {
                "type": "query_shard_exception",
                "reason": "failed to create query: {\n  \"match\" : {\n    \"houseType\" : {\n      \"query\" : \"2室\",\n      \"operator\" :
                    \"OR\",\n      \"prefix_length\" : 0,\n      \"max_expansions\" : 50,\n      \"fuzzy_transpositions\" : true,\n
                    \"lenient\" : false,\n      \"zero_terms_query\" : \"NONE\",\n      \"auto_generate_synonyms_phrase_query\" : true,\n
                    \"boost\" : 1.0\n    }\n  }\n}",
                "index_uuid": "gGKEaDFKQSmd5-f-ZoMFxA",
                "index": "haoke"
            }
        ],
        "type": "search_phase_execution_exception",
        "reason": "all shards failed",
        "phase": "query",
        "grouped": true,
        "failed_shards": [
            {
```

被设置为index为false的字段不能进行搜索操作。

1.6.3、批量导入数据

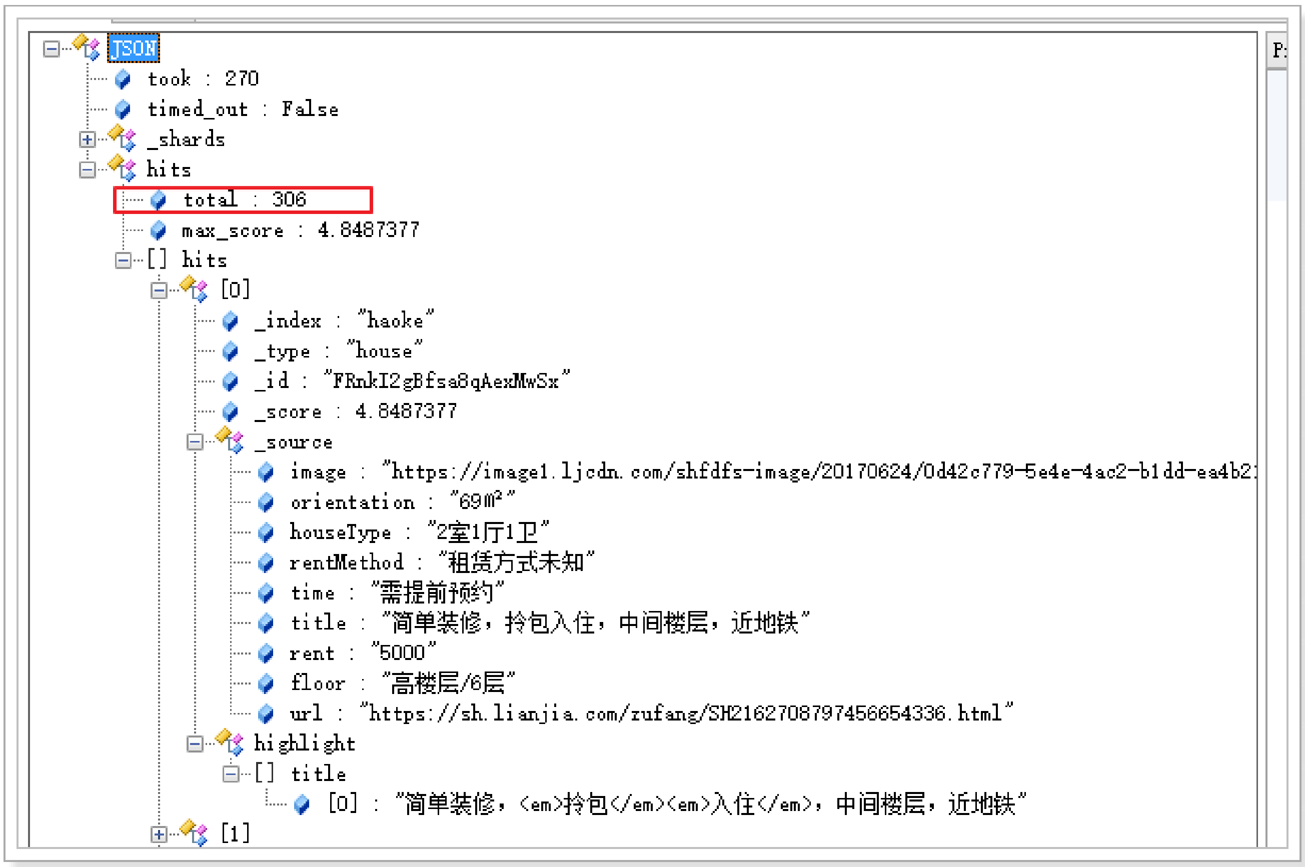
```
@Test
    public void tesBulk() throws Exception {
        Request request = new Request("POST", "/haoke/house/_bulk");
        List<String> lines = FileUtils.readLines(new File("F:\\code\\data.json"),
"UTF-8");
        String createStr = "{\"index\":
{\"_index\":\"haoke\",\"_type\":\"house\"}}";
        StringBuilder sb = new StringBuilder();
        int count = 0;
        for (String line : lines) {

            sb.append(createStr + "\n" + line + "\n");

            if (count >= 100) {

                request.setJsonEntity(sb.toString());
                Response response = this.restClient.performRequest(request);
                System.out.println("请求完成 -> " + response.getStatusLine());
                System.out.println(EntityUtils.toString(response.getEntity()));

                count = 0;
                sb = new StringBuilder();
            }
            count++;

        }

    }
```


查询 6 个分片中用的 6 个. 2002 命中. 耗时 0.023 秒

_index	_type	_id	_score ▲	image
haoke	house	jVvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/shfdfs-image/20160925/2b0083e9-d011-4267-bd3f-89c9d331dee8.jpg.7
haoke	house	lFvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/310000-inspection/021adaa5-9290-4910-871d-f1360fc47832.jpg.780x43
haoke	house	lVvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/310000-inspection/462f5bea-50fd-4bca-b281-64e11e2a1b71.jpg.780x43
haoke	house	nlvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/shfdfs-image/20160116/29dbe349-2fb2-4b6a-88b1-8d6bb506eb2e.jpg.7
haoke	house	oFvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/310000-inspection/prod-62615444-bd0e-4111-bca5-f62129071286.jpg.7
haoke	house	olvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/310000-inspection/3abd6497-8cba-4b6c-967d-66c028bb9f0b.jpg.780x43
haoke	house	pVvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/310000-inspection/test-06ddf293-0edd-4871-8695-dfea9fb104db.png.78
haoke	house	qlvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/310000-inspection/2fa9bc71-9f45-42a8-b8ea-390944ce9370.jpg.780x43
haoke	house	tFvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/310000-inspection/463381b4-0a22-4869-8f9d-bb9bafbb3a94.jpg.780x43
haoke	house	tlvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/hdic-frame/3df79499-4654-4e65-b1eb-1d7ab7e1d50f.png.780x439.jpg
haoke	house	uFvjI2gByl8D_d5LrzIY	1	https://image1.ljcdn.com/310000-inspection/026bcb62-ac52-458c-946c-745665130c7a.JPG.780x4

进行搜索测试：

```
POST http://192.168.1.7:9200/haoke/house/_search
{
    "query": {
        "match": {
            "title": {
                "query": "拎包入住"
            }
        }
    },
    "highlight": {
        "fields": {
            "title": {}
        }
    }
}
```



```
JSON
  took : 270
  timed_out : False
  _shards
  hits
    total : 306
    max_score : 4.8487377
    [] hits
      [0]
        _index : "haoke"
        _type : "house"
        _id : "FRnkI2gBfsa8qAexMwSx"
        _score : 4.8487377
        _source
          image : "https://image1.ljcdn.com/shfdfs-image/20170624/0d42c779-5e4e-4ac2-b1dd-ea4b2
          orientation : "69㎡"
          houseType : "2室1厅1卫"
          rentMethod : "租赁方式未知"
          time : "需提前预约"
          title : "简单装修，拎包入住，中间楼层，近地铁"
          rent : "5000"
          floor : "高楼层/6层"
          url : "https://sh.lianjia.com/zufang/SH2162708797456654336.html"
        highlight
          [] title
            [0] : "简单装修，<em>拎包</em><em>入住</em>，中间楼层，近地铁"
      [1]
```

2、开发搜索接口

在itcast-haoke-manage-api-server工程中，实现对外的搜索接口。

2.1、导入依赖

```xml
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-elasticsearch</artifactId>
</dependency>
```

2.2、添加配置

```properties
spring.data.elasticsearch.cluster-name=es-itcast-cluster
spring.data.elasticsearch.cluster-nodes=192.168.1.7:9300,192.168.1.7:9301,192.168.1.7:9302
```

2.3、编写vo

```java
package cn.itcast.haoke.dubbo.api.vo;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import org.springframework.data.annotation.Id;
import org.springframework.data.elasticsearch.annotations.Document;

@Data
@AllArgsConstructor
@NoArgsConstructor
@Document(indexName = "haoke", type = "house", createIndex = false)
public class HouseData {

    @Id
    private String id;
    private String title;
    private String rent;
    private String floor;
    private String image;
    private String orientation;
    private String houseType;
    private String rentMethod;
    private String time;


}
```

```java
package cn.itcast.haoke.dubbo.api.vo;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import java.util.List;

@Data
@AllArgsConstructor
@NoArgsConstructor
public class SearchResult {

    private Integer totalPage;

    private List<HouseData> list;

}

```

2.4、编写Controller

```java
package cn.itcast.haoke.dubbo.api.controller;

import cn.itcast.haoke.dubbo.api.service.SearchService;
import cn.itcast.haoke.dubbo.api.vo.SearchResult;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.io.UnsupportedEncodingException;
```

```java

@RequestMapping("search")
@RestController
@CrossOrigin
public class SearchController {

    @Autowired
    private SearchService searchService;

    @GetMapping
    public SearchResult search(@RequestParam("keyword") String keyWord,
                               @RequestParam(value = "page", defaultValue = "1")
Integer page) {
        if(page > 100){ //防止爬虫抓取过多的数据
            page = 1;
        }

        return this.searchService.search(keyWord, page);

    }

}
```

2.5、编写Service

```java
package cn.itcast.haoke.dubbo.api.service;

import cn.itcast.haoke.dubbo.api.vo.HouseData;
import cn.itcast.haoke.dubbo.api.vo.SearchResult;
import org.elasticsearch.index.query.Operator;
import org.elasticsearch.index.query.QueryBuilders;
import org.elasticsearch.search.fetch.subphase.highlight.HighlightBuilder;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.elasticsearch.core.ElasticsearchTemplate;
import org.springframework.data.elasticsearch.core.aggregation.AggregatedPage;
import org.springframework.data.elasticsearch.core.query.NativeSearchQueryBuilder;
import org.springframework.data.elasticsearch.core.query.SearchQuery;
import org.springframework.data.mongodb.core.aggregation.ArrayOperators;
import org.springframework.stereotype.Service;

@Service
public class SearchService {

    @Autowired
    private ElasticsearchTemplate elasticsearchTemplate;

    public static final Integer ROWS = 10;


    public SearchResult search(String keyWord, Integer page) {

        PageRequest pageRequest = PageRequest.of(page - 1, ROWS); //设置分页参数

        SearchQuery searchQuery = new NativeSearchQueryBuilder()
                .withQuery(QueryBuilders.matchQuery("title",
keyWord).operator(Operator.AND)) // match查询
                .withPageable(pageRequest)
                .withHighlightFields(new HighlightBuilder.Field("title")) // 设置高亮
                .build();

        AggregatedPage<HouseData> housePage =
                this.elasticsearchTemplate.queryForPage(searchQuery,
HouseData.class);

        return new SearchResult(housePage.getTotalPages(), housePage.getContent());
    }
}
```

2.5、启动测试

启动，发现报错：

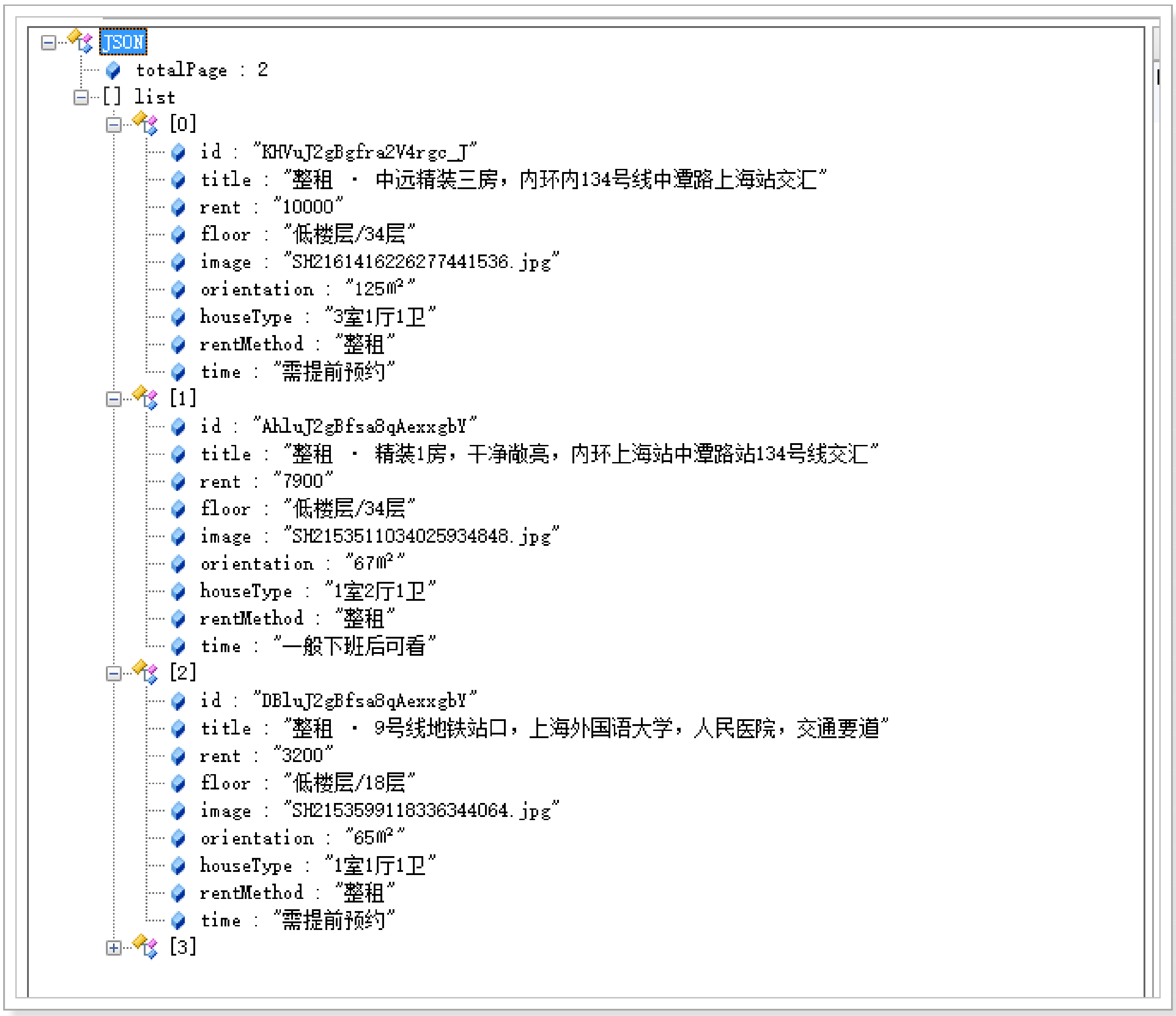
```
'elasticsearchClient' threw exception; nested exception is
java.lang.IllegalStateException: availableProcessors is already set to [3], rejecting
[3]
```

原因是整合了Redis后，引发了netty的冲突，需要在启动类中加入：

```java
package cn.itcast.haoke.dubbo.api;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DubboApiApplication {

    public static void main(String[] args) {
        System.setProperty("es.set.netty.runtime.available.processors","false");
        SpringApplication.run(DubboApiApplication.class, args);
    }
}
```

重新启动，发现问题解决了。

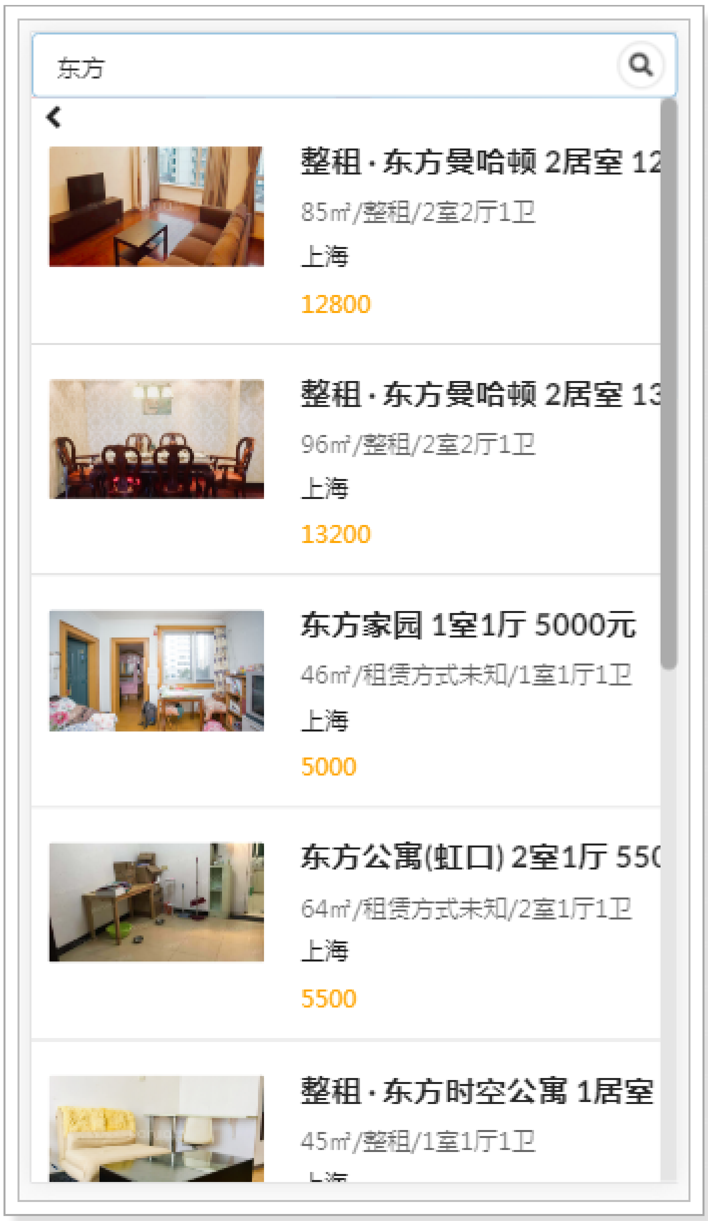
```
GET http://127.0.0.1:18080/search?keyWord=上海&page=2
```



```
JSON
  totalPage : 2
  [] list
    [0]
      id : "KHVuJ2gBgfra2V4rgc_J"
      title : "整租 · 中远精装三房，内环内134号线中潭路上海站交汇"
      rent : "10000"
      floor : "低楼层/34层"
      image : "SH2161416226277441536.jpg"
      orientation : "125㎡"
      houseType : "3室1厅1卫"
      rentMethod : "整租"
      time : "需提前预约"
    [1]
      id : "AhluJ2gBfsa8qAexxgbY"
      title : "整租 · 精装1房，干净敞亮，内环上海站中潭路站134号线交汇"
      rent : "7900"
      floor : "低楼层/34层"
      image : "SH2153511034025934848.jpg"
      orientation : "67㎡"
      houseType : "1室2厅1卫"
      rentMethod : "整租"
      time : "一般下班后可看"
    [2]
      id : "DBluJ2gBfsa8qAexxgbY"
      title : "整租 · 9号线地铁站口，上海外国语大学，人民医院，交通要道"
      rent : "3200"
      floor : "低楼层/18层"
      image : "SH2153599118336344064.jpg"
      orientation : "65㎡"
      houseType : "1室1厅1卫"
      rentMethod : "整租"
      time : "需提前预约"
    [3]
```

3、整合前端开发

实现的效果：


东方

整租·东方曼哈顿 2居室 12
85㎡/整租/2室2厅1卫
上海
12800

整租·东方曼哈顿 2居室 13
96㎡/整租/2室2厅1卫
上海
13200

东方家园 1室1厅 5000元
46㎡/租赁方式未知/1室1厅1卫
上海
5000

东方公寓(虹口) 2室1厅 550
64㎡/租赁方式未知/2室1厅1卫
上海
5500

整租·东方时空公寓 1居室
45㎡/整租/1室1厅1卫

3.1、编写home.js

```
import React from 'react';
// import Carousel from 'nuka-carousel';
import ImageGallery from 'react-image-gallery';
import { withRouter } from 'react-router';
import { Input,Grid,Icon,Item,Button,Dimmer,Loader} from 'semantic-ui-react'
import './home.css';
import "react-image-gallery/styles/css/image-gallery.css";
import axios from 'axios';
import config from '../../common.js';
import MapHouse from './maphouse.js';
import Calculator from './calc.js';
import SearchBar from './searchbar.js';
import ApolloClient from "apollo-boost";
import gql from "graphql-tag";

const client = new ApolloClient({
    uri: "http://127.0.0.1:18080/graphql"
});

//定义查询
const GET_INDEX_ADS = gql`
    {
        IndexAdList{
            list{
                original
            }
        }
    }
`;

class Home extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      swipeData: [],
      swipeLoading: false,
      menuData: [],
      menuLoading: false,
      infoData: [],
      infoLoading: false,
      faqData: [],
      faqLoading: false,
      globalLoading: true,
      mapShowFlag: false,
      calcShowFlag: false,
      searchBarFlag: false,
      searchData:[]
    };
  }
  componentDidMount = () => {
    let swipe = new Promise((resolve, reject) => {
        client.query({query: GET_INDEX_ADS}).then(result =>
            resolve(result.data.IndexAdList.list));
    })

    let menu = new Promise((resolve, reject) => {
        axios.get('http://127.0.0.1:18080/mock/indexMenu').then((data)=>{
            resolve(data.data.list);
        });
    })
    let info = new Promise((resolve, reject) => {
      axios.get('http://127.0.0.1:18080/mock/index/info').then((data)=>{
        resolve(data.data.list);
      });
    })
    let faq = new Promise((resolve, reject) => {
      axios.get('http://127.0.0.1:18080/mock/index/faq').then((data)=>{
        resolve(data.data.list);
      });
    })
    let house = new Promise((resolve, reject) => {
      axios.get('http://127.0.0.1:18080/mock/index/house').then((data)=>{
        resolve(data.data.list);
      });
    })
    Promise.all([swipe, menu, info, faq, house]).then((result)=>{
      this.setState({
        swipeData: result[0],
        menuData: result[1],
        infoData: result[2],
        faqData: result[3],
        houseData: result[4],
        menuLoading: true,
        swipeLoading: true,
        infoLoading: true,
        faqLoading: true,
        houseLoading: true,
        globalLoading: false
      });
    })
  }
  hideMap = () => {
    this.setState({mapShowFlag:false});
  }
  hideCalc = () => {
    this.setState({calcShowFlag:false});
  }
  hideSearchBar = () => {
    this.setState({searchBarFlag:false});
  }

  search = (event, data) =>{
      let value = data.value;
      let _this =this;
      _this.searchHandle();
      axios.get('http://127.0.0.1:18080/search?
keyword='+value+'&page=1').then((data)=>{
          _this.setState({searchData:data.list});
      });
  }
  handleMenu = (name) => {
    switch(name){
      case '地图找房':
        this.setState({mapShowFlag:true});
        break;
      case '计算器':
        this.setState({calcShowFlag:true});
        break;
      case '二手房':
        this.props.history.push('/home/list',{query:{name:name,type:1}});
        break;
      case '新房':
          this.props.history.push('/home/list',{query:{name:name,type:2}});
          break;
        case '租房':
          this.props.history.push('/home/list',{query:{name:name,type:3}});
          break;
        case '海外':
          this.props.history.push('/home/list',{query:{name:name,type:4}});
          break;
        case '问答':
          this.props.history.push('/home/find',{query:{flag:true}});
          break;
        default:
          break;
      }
    }
    searchHandle = () => {
      this.setState({
        searchBarFlag: true
      })
    }
    render() {
      // 轮播图渲染
      const swipeLoading = this.state.swipeLoading;
      const swipeData = this.state.swipeData;
      let swipe = null;
      if(swipeLoading) {
        swipe = <ImageGallery
                  preventDefaultTouchmoveEvent={true}
                  autoPlay={true}
                  disableSwipe={false}
                  showThumbnails={false}
                  items={swipeData} />
      }
      // 菜单渲染
      const menuLoading = this.state.menuLoading;
      const menuData = this.state.menuData;
      let menu = null;
      if(menuLoading) {
        let list = menuData.map(item => {
          return (
            <Grid.Column onClick={this.handleMenu.bind(this,item.menu_name)} key=
{item.id}>
              <div className='home-menu-item'>
                <Icon name='home' size='big' />
              </div>
              <div>{item.menu_name}</div>
            </Grid.Column>
          )
        })
        menu = (
          <Grid padded divided >
            <Grid.Row columns={4}>
              {list}
            </Grid.Row>
          </Grid>
        )
      }
      // 渲染资讯
      let infos = null;
      if(this.state.infoLoading) {
        infos = this.state.infoData.map(item=>{
          return (
            <Item.Header key={item.id}>
              <span>限购 ●</span>
              <span>{item.info_title}</span>
            </Item.Header>
          );
        })
      }
      // 渲染问答
      let faq = null;
      if(this.state.faqLoading) {
        faq = this.state.faqData.map(item=>{
          return (
            <li key={item.question_id}>
              <div>
                <Icon name='question circle outline' />
                <span>{item.question_name}</span>
              </div>
              <div>
                {item.question_tag.split(',').map((tag,index)=>{return <Button key={index} basic color='green' size='mini'>{tag}</Button>})}
                <div>{item.atime} ● <Icon name='comment alternate outline' />{item.qnum}</div>
              </div>
            </li>
          );
        })
      }
      // 渲染房屋
      let newHouse = [];
      let oldHouse = [];
      let hireHouse = [];
      if(this.state.houseLoading) {
        this.state.houseData.forEach(item=>{
          let listInfo = (
            <Item key={item.id}>
              <Item.Image src={config.imgBaseUrl+'public/home.png'}/>
              <Item.Content>
                <Item.Header>{item.home_name}</Item.Header>
                <Item.Meta>
                  <span className='cinema'>{item.home_desc}</span>
                </Item.Meta>
                <Item.Description>
                  {item.home_tags.split(',').map((tag,index)=>{return <Button key={index} basic color='green' size='mini'>{tag}</Button>})}
                </Item.Description>
                <Item.Description>{item.home_price}</Item.Description>
              </Item.Content>
            </Item>
          );
          if(item.home_type === 1) {
            newHouse.push(listInfo);
          }else if(item.home_type === 2) {
            oldHouse.push(listInfo);
          }else if(item.home_type === 3) {
            hireHouse.push(listInfo)
          }
        })
      }
      return (
        <div className='home-container'>
          {this.state.mapShowFlag?<MapHouse hideMap={this.hideMap}/>:null}
          {this.state.calcShowFlag?<Calculator hideCalc={this.hideCalc}/>:null}
          {this.state.searchBarFlag?<SearchBar hideSearchBar={this.hideSearchBar} searchData={this.state.searchData}/>:null}
          <Dimmer inverted active={this.state.globalLoading} page>
            <Loader>Loading</Loader>
          </Dimmer>
          <div className="home-topbar">
              {/*onBlur={this.hideSearchBar}*/}
              {/*onClick={this.searchHandle}*/}
            <Input fluid onChange={this.search.bind(this) } icon={{ name: 'search', circular: true, link: true }}  placeholder='搜房源...' />
          </div>
          <div className="home-content">
            {swipe}
            {menu}
            <div className='home-msg'>
              <Item.Group unstackable>
                <Item className='home-msg-img' >
                  <Item.Image size='tiny' src={config.imgBaseUrl+'public/zixun.png'} />
                  <Item.Content verticalAlign='top'>
                    {infos}
                    <div className="home-msg-more">
                      <Icon name='angle right' size='big' />
                    </div>
                  </Item.Content>
                </Item>
              </Item.Group>
            </div>
            <div className='home-ask'>
              <div className='home-ask-title'>好客问答</div>
              <ul>
                {faq}
              </ul>
            </div>
            <div>
```

```
              <div className='home-hire-title'>最新开盘</div>
              <Item.Group divided unstackable>
                {newHouse}
              </Item.Group>
            </div>
            <div>
              <div className='home-hire-title'>二手精选</div>
              <Item.Group divided unstackable>
                {oldHouse}
              </Item.Group>
            </div>
            <div>
              <div className='home-hire-title'>组一个家</div>
              <Item.Group divided unstackable>
                {hireHouse}
              </Item.Group>
            </div>
          </div>
        </div>
      );
    }
  }
  export default withRouter(Home);

```

3.2、编写searchbar.js

```
import React from 'react';
import { Icon,Item } from 'semantic-ui-react';

class SearchBar extends React.Component {

    hideSearchBar = () => {
        this.props.hideSearchBar();
    }

  render() {
    return (
      <div className = 'search-bar' >
          <Icon onClick={this.hideSearchBar} name = 'angle left' size = 'large'/>
          <div className = "search-bar-content">
            <Item.Group divided unstackable>
                {
                    this.props.searchData.map(item => {
                        return (
                            <Item key={item.id}>
                                <Item.Image src={"https://itcast-haoke.oss-cn-
qingdao.aliyuncs.com/lj/" + item.image}/>
                                <Item.Content>
                                    <Item.Header><div className='house-title'>
{item.title}</div></Item.Header>
                                    <Item.Meta>
```

```
                                                    <span className='cinema'>
{item.orientation}/{item.rentMethod}/{item.houseType}</span>
                                                </Item.Meta>
                                                <Item.Description>
                                                    上海
                                                </Item.Description>
                                                <Item.Description>{item.rent}
</Item.Description>
                                            </Item.Content>
                                        </Item>
                                    )
                                })
                            }

                        </Item.Group>
                    </div>
                </div>
            );
        }
    }
export default SearchBar;
```

3.3、修改home.css

```
.search-bar {
  position: fixed;
  bottom: 50px;
  top: 38px;
  z-index: 9999;
  height: 100%;
  width: 100%;
  background-color: #fff;
  overflow-y: auto; /**这里做了修改，y轴方向有滚动条**/
}
```

3.4、新增search.css

```
.house-title{
    overflow: hidden;
    white-space: nowrap;
}
```

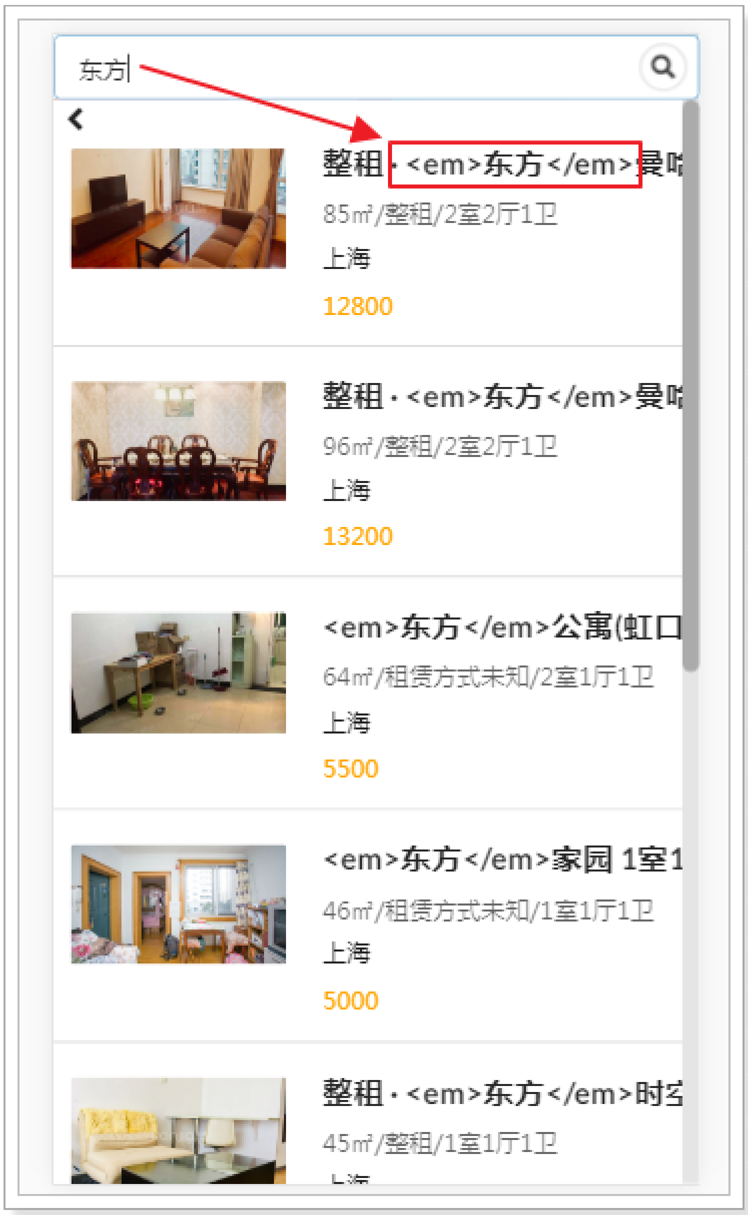
4、优化搜索功能

4.1、高亮

```
package cn.itcast.haoke.dubbo.api.service;

import cn.itcast.haoke.dubbo.api.vo.HouseData;
import cn.itcast.haoke.dubbo.api.vo.SearchResult;
import org.apache.commons.lang3.ClassUtils;
import org.apache.commons.lang3.reflect.FieldUtils;
import org.elasticsearch.action.search.SearchResponse;
import org.elasticsearch.common.text.Text;
import org.elasticsearch.index.query.Operator;
import org.elasticsearch.index.query.QueryBuilders;
import org.elasticsearch.search.SearchHit;
import org.elasticsearch.search.SearchHits;
import org.elasticsearch.search.fetch.subphase.highlight.HighlightBuilder;
import org.elasticsearch.search.fetch.subphase.highlight.HighlightField;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.cglib.core.ReflectUtils;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.data.elasticsearch.core.ElasticsearchTemplate;
import org.springframework.data.elasticsearch.core.SearchResultMapper;
import org.springframework.data.elasticsearch.core.aggregation.AggregatedPage;
import
org.springframework.data.elasticsearch.core.aggregation.impl.AggregatedPageImpl;
import org.springframework.data.elasticsearch.core.query.NativeSearchQueryBuilder;
import org.springframework.data.elasticsearch.core.query.SearchQuery;
import org.springframework.data.mongodb.core.aggregation.ArrayOperators;
import org.springframework.stereotype.Service;

import java.lang.reflect.Field;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Map;

@Service
public class SearchService {

    @Autowired
    private ElasticsearchTemplate elasticsearchTemplate;

    public static final Integer ROWS = 10;


    public SearchResult search(String keyWord, Integer page) {

        PageRequest pageRequest = PageRequest.of(page - 1, ROWS); //设置分页参数

        SearchQuery searchQuery = new NativeSearchQueryBuilder()
                .withQuery(QueryBuilders.matchQuery("title",
keyWord).operator(Operator.AND)) // match查询
                .withPageable(pageRequest)
                .withHighlightFields(new HighlightBuilder.Field("title")) // 设置高
亮
                .build();

        AggregatedPage<HouseData> housePage =
```

```java
                this.elasticsearchTemplate.queryForPage(searchQuery, HouseData.class, new SearchResultMapper() {
                    @Override
                    public <T> AggregatedPage<T> mapResults(SearchResponse response, Class<T> clazz, Pageable pageable) {
                        List<T> result = new ArrayList<>();

                        if (response.getHits().totalHits == 0) {
                            return new AggregatedPageImpl<>(Collections.emptyList(), pageable, 0L);
                        }

                        for (SearchHit searchHit : response.getHits()) {
                            // 通过反射写入数据到对象中
                            T obj = (T) ReflectUtils.newInstance(clazz);

                            try {
                                //写入id
                                FieldUtils.writeField(obj, "id", searchHit.getId(), true);
                            } catch (IllegalAccessException e) {
                                e.printStackTrace();
                            }

                            Map<String, Object> sourceAsMap = searchHit.getSourceAsMap();
                            for (Map.Entry<String, Object> entry : sourceAsMap.entrySet()) {

                                Field field = FieldUtils.getField(clazz, entry.getKey(), true);
                                if (null == field) {
                                    continue;
                                }
                                try {
                                    FieldUtils.writeField(obj, entry.getKey(), entry.getValue(), true);
                                } catch (IllegalAccessException e) {
                                    e.printStackTrace();
                                }
                            }

                            // 处理高亮
                            for (Map.Entry<String, HighlightField> stringHighlightFieldEntry : searchHit.getHighlightFields().entrySet()) {
                                try {
                                    Text[] fragments = stringHighlightFieldEntry.getValue().fragments();
                                    StringBuilder sb = new StringBuilder();
                                    for (Text fragment : fragments) {
                                        sb.append(fragment.toString());
                                    }
                                    FieldUtils.writeField(obj,
stringHighlightFieldEntry.getKey(), sb.toString(), true);
                                } catch (IllegalAccessException e) {
                                    e.printStackTrace();
                                }
                            }

                            result.add(obj);
                        }
                        return new AggregatedPageImpl<>(result, pageable,
response.getHits().totalHits);
                    }
                });

        return new SearchResult(housePage.getTotalPages(),
housePage.getContent());
    }
}

```

整合到前端进行测试：


东方

整租·<em>东方</em>曼吲
85㎡/整租/2室2厅1卫
上海
12800

整租·<em>东方</em>曼吲
96㎡/整租/2室2厅1卫
上海
13200

<em>东方</em>公寓(虹口
64㎡/租赁方式未知/2室1厅1卫
上海
5500

<em>东方</em>家园 1室1
46㎡/租赁方式未知/1室1厅1卫
上海
5000

整租·<em>东方</em>时宫
45㎡/整租/1室1厅1卫

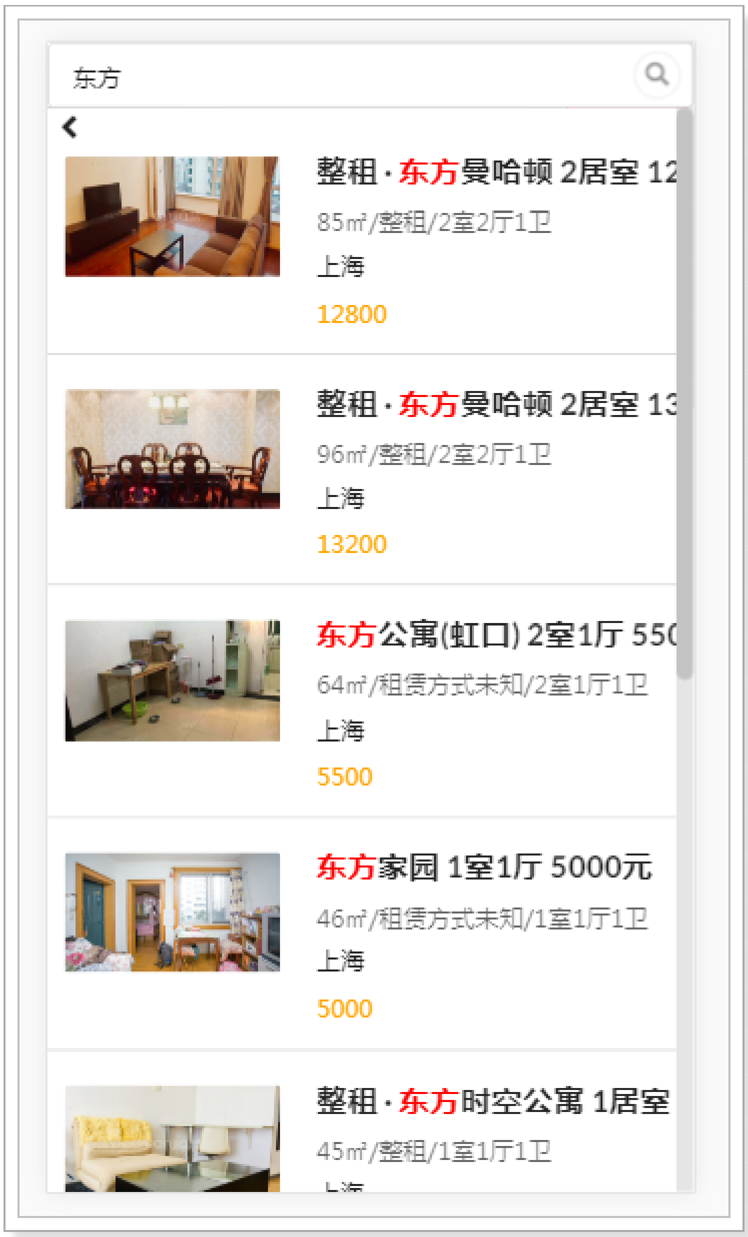
需要在页面中显示em标签以及定义其样式：

```
<Item.Header><div className='house-title' dangerouslySetInnerHTML=
{{__html:item.title}}></div></Item.Header>
```

样式：

```
.house-title em{
    font-style: normal;
    color: red;
}
```

效果：


东方

整租 · 东方曼哈顿 2居室 12
85㎡/整租/2室2厅1卫
上海
12800

整租 · 东方曼哈顿 2居室 13
96㎡/整租/2室2厅1卫
上海
13200

东方公寓(虹口) 2室1厅 550
64㎡/租赁方式未知/2室1厅1卫
上海
5500

东方家园 1室1厅 5000元
46㎡/租赁方式未知/1室1厅1卫
上海
5000

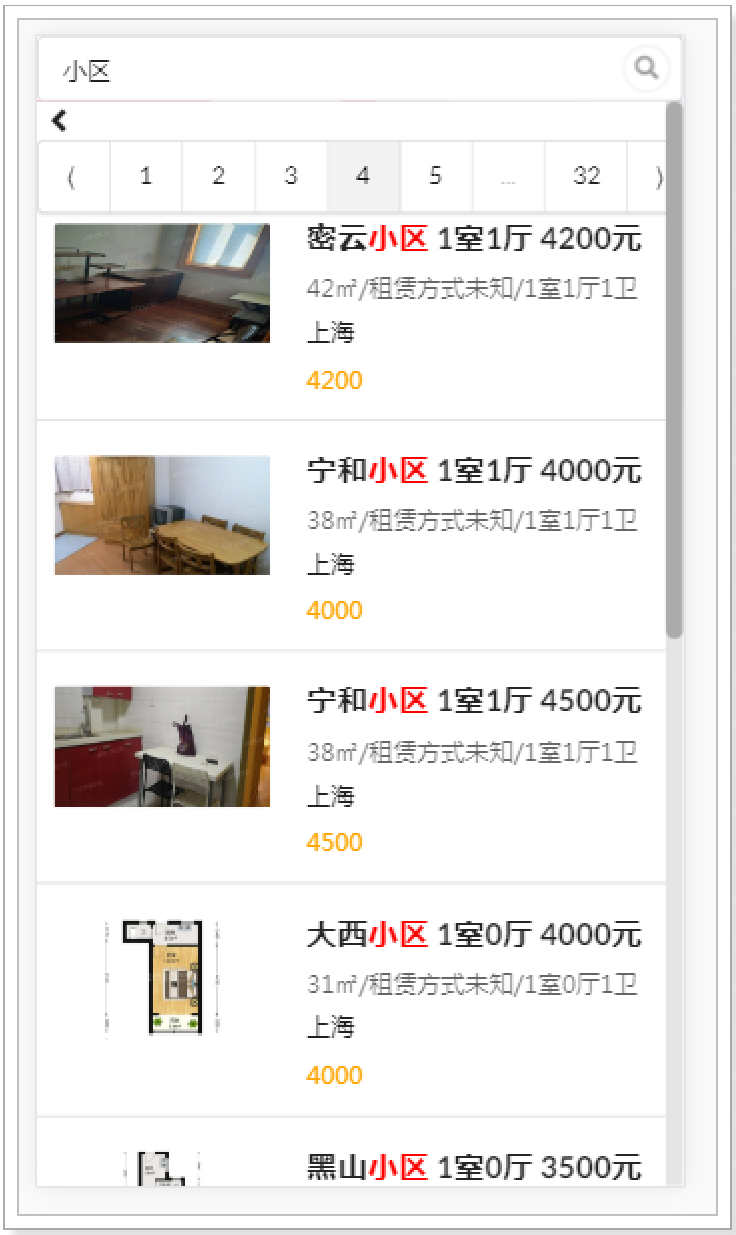
整租 · 东方时空公寓 1居室
45㎡/整租/1室1厅1卫

4.2、分页

4.2.1、添加分页组件

```
import { Icon,Item,Pagination } from 'semantic-ui-react';
.........
handlePageChange = (e, { activePage }) =>{
        this.props.searchPage(null,{
            page:activePage
        });
}
........
{
    this.props.totalPage > 1 ? (
        <Pagination
            defaultActivePage={1}
            firstItem={null}
            lastItem={null}
            totalPages={this.props.totalPage}
            onPageChange={this.handlePageChange}
            />
    ) : null
}

---home.js----
search = (event, data) =>{
      let value = data.value ? data.value : this.state.searchKeyword;
      let page = data.page ? data.page : 1;
      let _this =this;
      _this.searchHandle();
      axios.get('http://127.0.0.1:18080/search?keyword='+value+'&page=' +
page).then((data)=>{

 _this.setState({searchData:data.list,totalPage:data.totalPage,searchKeyword:value}
);
      });
  }
..........
{this.state.searchBarFlag?<SearchBar hideSearchBar={this.hideSearchBar} searchPage=
{this.search} totalPage={this.state.totalPage} searchData=
{this.state.searchData}/>:null}
..........

```

4.2、测试


小区

(1 2 3 4 5 ... 32)

密云小区 1室1厅 4200元
42㎡/租赁方式未知/1室1厅1卫
上海
4200

宁和小区 1室1厅 4000元
38㎡/租赁方式未知/1室1厅1卫
上海
4000

宁和小区 1室1厅 4500元
38㎡/租赁方式未知/1室1厅1卫
上海
4500

大西小区 1室0厅 4000元
31㎡/租赁方式未知/1室0厅1卫
上海
4000

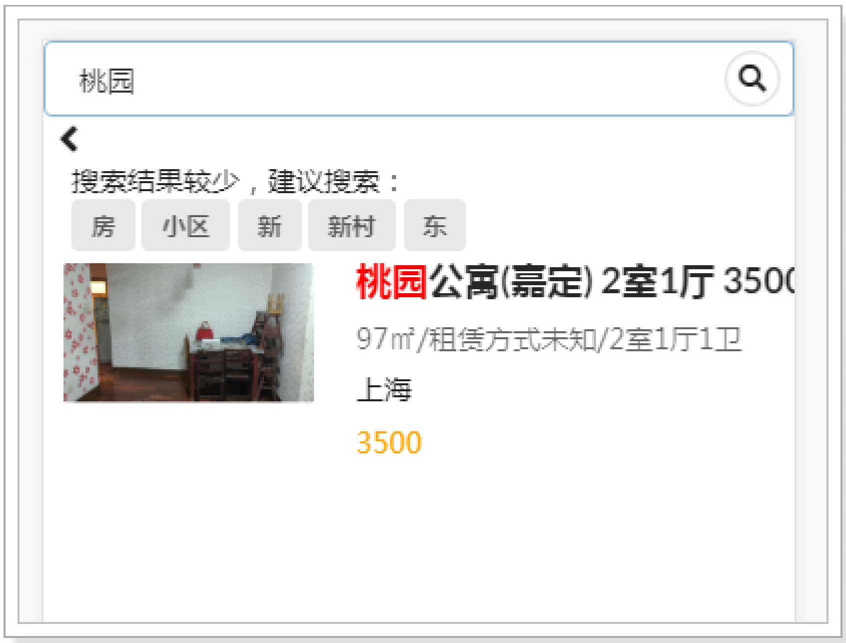
黑山小区 1室0厅 3500元

5、热词搜索

需求：当无搜索结果或搜索结果只有一页时，显示搜索热词。最多显示5个热词。

热词：按照用户搜索的关键字以及搜索到的结果数量进行排序，数量越多的越排到前面，从而得到热词。

效果：


桃园

搜索结果较少，建议搜索：

房 小区 新 新村 东

桃园公寓(嘉定) 2室1厅 350

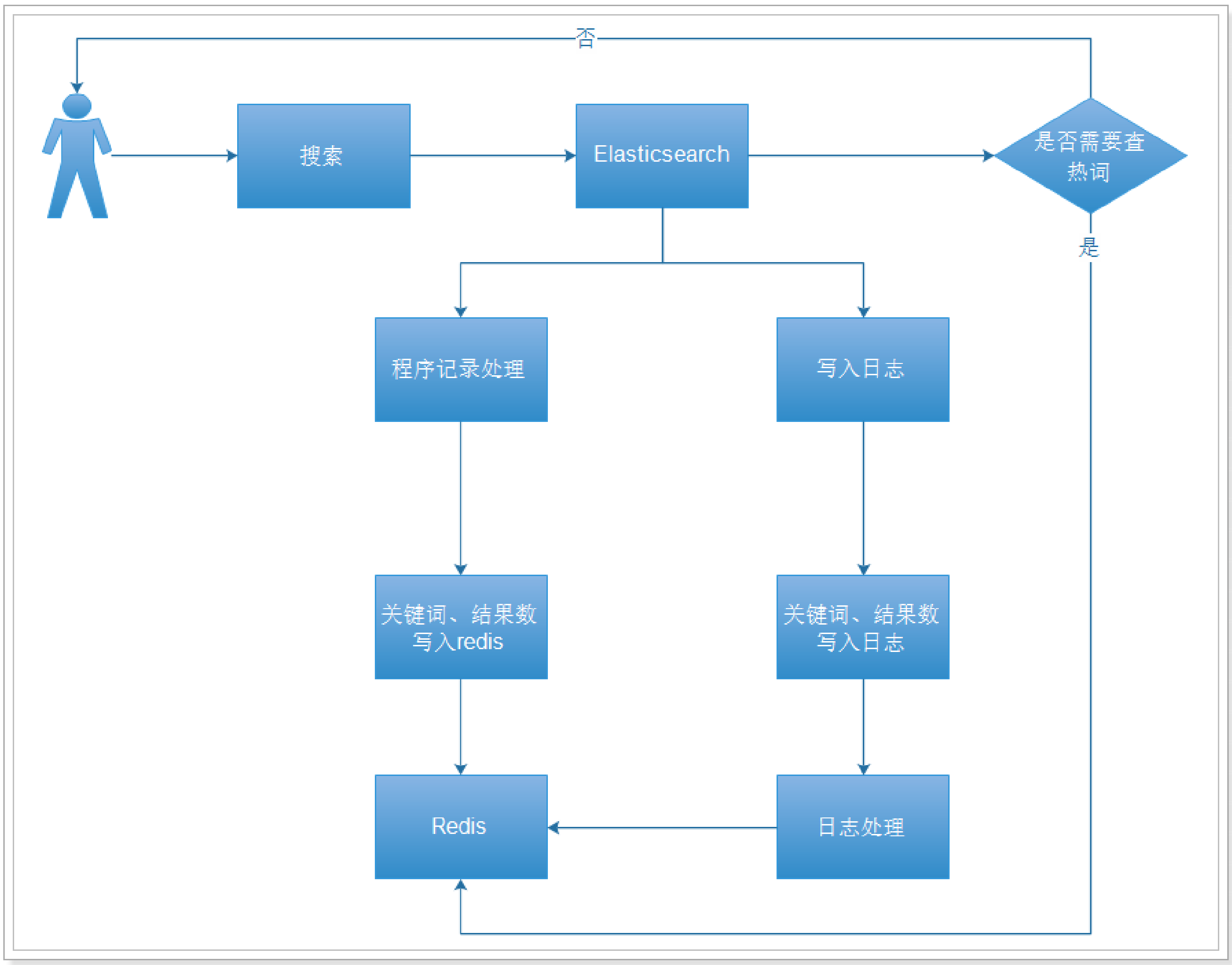
97㎡/租赁方式未知/2室1厅1卫

上海

3500

5.1、实现分析

根据热词的定义，我们可以知道，热词是来源于用户的搜索，那么就要记录用户的搜索关键字以及结果数量，然后再排序得到热词。



说明：

1. 用户搜索数据，首先进行Elasticsearch搜索
2. 在搜索完成后，进行判断，是否需要查询热词
3. 如果不需要，直返返回用户数据即可
4. 如果需要查询，则进行再Redis中查询热词
5. 对于用户搜索词的处理有两种方案
    1. 第一种方案，是在程序中进行处理，并且把搜索词以及命中的数据数量存储到redis中。该方案是同步进行。
    2. 第二种方案，是将查询信息先记录到日志文件中，由后续的程序做处理，然后再写入到Redis中。该方案是异步进行。

5.2、后台实现

```java
package cn.itcast.haoke.dubbo.api.controller;

import cn.itcast.haoke.dubbo.api.service.SearchService;
import cn.itcast.haoke.dubbo.api.vo.SearchResult;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.redis.core.RedisTemplate;
import org.springframework.web.bind.annotation.*;

import java.io.UnsupportedEncodingException;
import java.util.Set;

@RequestMapping("search")
@RestController
@CrossOrigin
public class SearchController {

    private static final Logger LOGGER = 
LoggerFactory.getLogger(SearchController.class);

    @Autowired
    private SearchService searchService;

    @Autowired
    private RedisTemplate redisTemplate;

    @GetMapping
    public SearchResult search(@RequestParam("keyWord") String keyWord,
                               @RequestParam(value = "page", defaultValue = "1")
Integer page) {
        if (page > 100) { //防止爬虫抓取过多的数据
            page = 1;
        }

        SearchResult search = this.searchService.search(keyWord, page);
        String redisKey = "HAOKE_HOT_WORD";

        if (search.getTotalPage() <= 1) {
            //需要查询热词，按照得分倒序排序，获取前5条数据
            Set<String> set = this.redisTemplate.opsForZSet().reverseRange(redisKey, 0, 4);
            search.setHotWord(set);
        }

        // 处理热词
        Integer count = ((Math.max(search.getTotalPage(), 1) - 1) * SearchService.ROWS) + search.getList().size();
        // 采用zset方式进行存储，值所对应的得分是数据条数
        this.redisTemplate.opsForZSet().add(redisKey, keyword, count);

        // 记录日志
        LOGGER.info("[Search]搜索关键字为：" + keyword + "，结果数量为：" + count);

        return search;

    }

}

```

5.3、整合前端实现

searchbar.js：

```
import React from 'react';
import { Icon,Item,Pagination,Label,Container } from 'semantic-ui-react';
import "./search.css"

class SearchBar extends React.Component {

    hideSearchBar = () => {
        this.props.hideSearchBar();
    }

    handlePageChange = (e, { activePage }) =>{
        this.props.searchPage(null,{
            page:activePage
        });
    }
    handleHotSearch = (e,data) =>{
        this.props.searchPage(null,{value:data.children});
    }

  render() {
    return (
      <div className = 'search-bar' >
          <Icon onClick={this.hideSearchBar} name = 'angle left' size = 'large'/>
          {
              this.props.totalPage > 1 ? (
                  <Container>
```

```
                      <Pagination
                          defaultActivePage={1}
                          firstItem={null}
                          lastItem={null}
                          totalPages={this.props.totalPage}
                          onPageChange={this.handlePageChange}
                      />
                  </Container>
              ) : null
          }
          {
              this.props.hotword ? (
                  <Container>搜索结果较少，建议搜索：<br/>
                      <span>
                          {
                              this.props.hotword.map(item => {
                                  return (
                                      <Label onClick={this.handleHotSearch}>{item}</Label>
                                  )
                              })
                          }
                      </span>
                  </Container>
              ): null
          }

          <div className = "search-bar-content">
            <Item.Group divided unstackable>
                {
                    this.props.searchData.map(item => {
                        return (
                            <Item key={item.id}>
                                <Item.Image src={"https://itcast-haoke.oss-cn-qingdao.aliyuncs.com/lj/" + item.image}/>
                                <Item.Content>
                                    <Item.Header><div className='house-title' dangerouslySetInnerHTML={{__html:item.title}}></div></Item.Header>
                                    <Item.Meta>
                                        <span className='cinema'>{item.orientation}/{item.rentMethod}/{item.houseType}</span>
                                    </Item.Meta>
                                    <Item.Description>
                                        上海
                                    </Item.Description>
                                    <Item.Description>{item.rent}</Item.Description>
                                </Item.Content>
                            </Item>
                        )
                    })
                }

                </Item.Group>
            </div>
          </div>
      );
    }
}
export default SearchBar;

```

home.js:

```
...................
search = (event, data) =>{
      let value = data.value ? data.value : this.state.searchKeyWord;
      let page = data.page ? data.page : 1;
      let _this =this;
      this.setState({
          searchKeyWord:value
      });
      _this.searchHandle();
      axios.get('http://127.0.0.1:18080/search?keyWord='+value+'&page=' +
page).then((data)=>{

 _this.setState({searchData:data.list,hotWord:data.hotWord,totalPage:data.totalPage
});
      });
  }
................
{this.state.searchBarFlag?<SearchBar hotWord={this.state.hotWord} hideSearchBar=
{this.hideSearchBar} searchPage={this.search} totalPage={this.state.totalPage}
searchData={this.state.searchData}/>:null}
.................

<Input fluid onChange={this.search.bind(this) } value={this.state.searchKeyWord}
icon={{ name: 'search', circular: true, link: true }}  placeholder='搜房源...' />

```

6、拼音分词

搜索时，需要对拼音也要支持的，如下：地铁、地tie、ditie 等都应该能够搜索到包含“地铁”的数据。

6.1、添加拼音分词插件

插件源码地址：https://github.com/medcl/elasticsearch-analysis-pinyin

下载：https://github.com/medcl/elasticsearch-analysis-pinyin/releases/download/v6.5.4/elasticsearch-analysis-pinyin-6.5.4.zip

```
#将zip压缩包，解压到/haoke/es-cluster/pinyin
unzip elasticsearch-analysis-pinyin-6.5.4.zip

#重新创建容器

docker stop es-node01 es-node02 es-node03
docker rm es-node01 es-node02 es-node03

docker create --name es-node01 --net host -v /haoke/es-cluster/node01/elasticsearch.yml:/usr/share/elasticsearch/config/elasticsearch.yml -v /haoke/es-cluster/node01/jvm.options:/usr/share/elasticsearch/config/jvm.options -v /haoke/es-cluster/ik:/usr/share/elasticsearch/plugins/ik -v /haoke/es-cluster/pinyin:/usr/share/elasticsearch/plugins/pinyin -v /haoke/es-cluster/node01/data:/usr/share/elasticsearch/data elasticsearch:6.5.4

docker create --name es-node02 --net host -v /haoke/es-cluster/node02/elasticsearch.yml:/usr/share/elasticsearch/config/elasticsearch.yml -v /haoke/es-cluster/node02/jvm.options:/usr/share/elasticsearch/config/jvm.options -v /haoke/es-cluster/ik:/usr/share/elasticsearch/plugins/ik -v /haoke/es-cluster/pinyin:/usr/share/elasticsearch/plugins/pinyin -v /haoke/es-cluster/node02/data:/usr/share/elasticsearch/data elasticsearch:6.5.4

docker create --name es-node03 --net host -v /haoke/es-cluster/node03/elasticsearch.yml:/usr/share/elasticsearch/config/elasticsearch.yml -v /haoke/es-cluster/node03/jvm.options:/usr/share/elasticsearch/config/jvm.options -v /haoke/es-cluster/ik:/usr/share/elasticsearch/plugins/ik -v /haoke/es-cluster/pinyin:/usr/share/elasticsearch/plugins/pinyin -v /haoke/es-cluster/node03/data:/usr/share/elasticsearch/data elasticsearch:6.5.4
```

6.2、测试拼音分词

```
PUT /medcl/
{
    "index" : {
        "analysis" : {
            "analyzer" : {
                "pinyin_analyzer" : {
                    "tokenizer" : "my_pinyin"
                    }
            },
            "tokenizer" : {
                "my_pinyin" : {
                    "type" : "pinyin",
                    "keep_separate_first_letter" : false,
                    "keep_full_pinyin" : true,
                    "keep_original" : true,
                    "limit_first_letter_length" : 16,
                    "lowercase" : true,
                    "remove_duplicated_term" : true
                }
            }
        }
    }
}
```

参数说明：

> - keep_first_letter：启用此选项时，例如：刘德华> ldh，默认值：true
> - keep_separate_first_letter：启用该选项时，将保留第一个字母分开，例如：刘德华> l，d，h，默认：假的，注意：查询结果也许是太模糊，由于长期过频
> - keep_full_pinyin：当启用该选项，例如：刘德华> [liu，de，hua]，默认值：true
> - keep_original：当启用此选项时，也会保留原始输入，默认值：false
> - limit_first_letter_length：设置first_letter结果的最大长度，默认值：16
> - lowercase：小写非中文字母，默认值：true
> - remove_duplicated_term：当启用此选项时，将删除重复项以保存索引，例如：de的> de，默认值：false，注意：位置相关查询可能受影响

测试分词：

```
GET /medcl/_analyze
{
  "text": ["刘德华"],
  "analyzer": "pinyin_analyzer"
}
```

测试结果：

```
{
    "tokens": [
        {
            "token": "liu",
            "start_offset": 0,
            "end_offset": 0,
            "type": "word",
            "position": 0
        },
        {
            "token": "刘德华",
            "start_offset": 0,
            "end_offset": 0,
            "type": "word",
            "position": 0
        },
        {
            "token": "ldh",
            "start_offset": 0,
            "end_offset": 0,
            "type": "word",
            "position": 0
        },
        {
            "token": "de",
            "start_offset": 0,
            "end_offset": 0,
            "type": "word",
            "position": 1
        },
        {
            "token": "hua",
            "start_offset": 0,
            "end_offset": 0,
            "type": "word",
            "position": 2
        }
    ]
}
```

创建mapping：

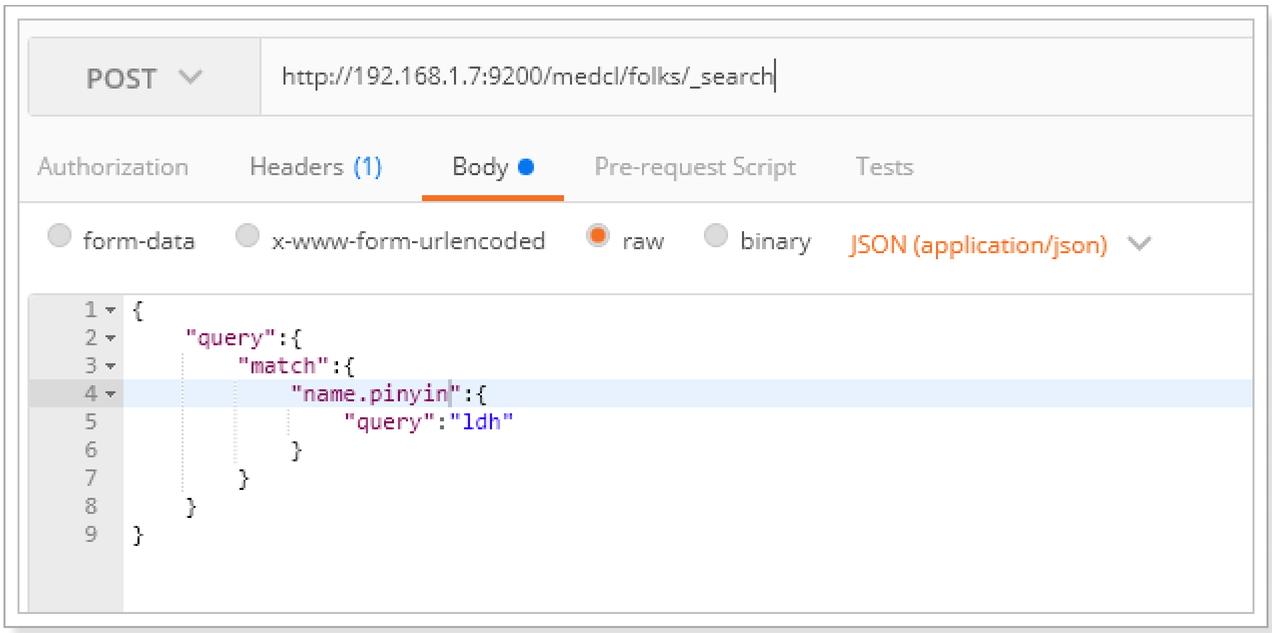
```
POST /medcl/folks/_mapping
{
    "folks": {
        "properties": {
            "name": {
                "type": "keyword",
                "fields": {
                    "pinyin": {
                        "type": "text",
                        "store": false,
                        "term_vector": "with_offsets",
                        "analyzer": "pinyin_analyzer",
                        "boost": 10
                    }
                }
            }
        }
    }
}
```

> 说明：
>
> 这里使用的是name的子字段，通过fields指定。

插入数据：

```
POST /medcl/folks/andy
{"name":"刘德华"}
```

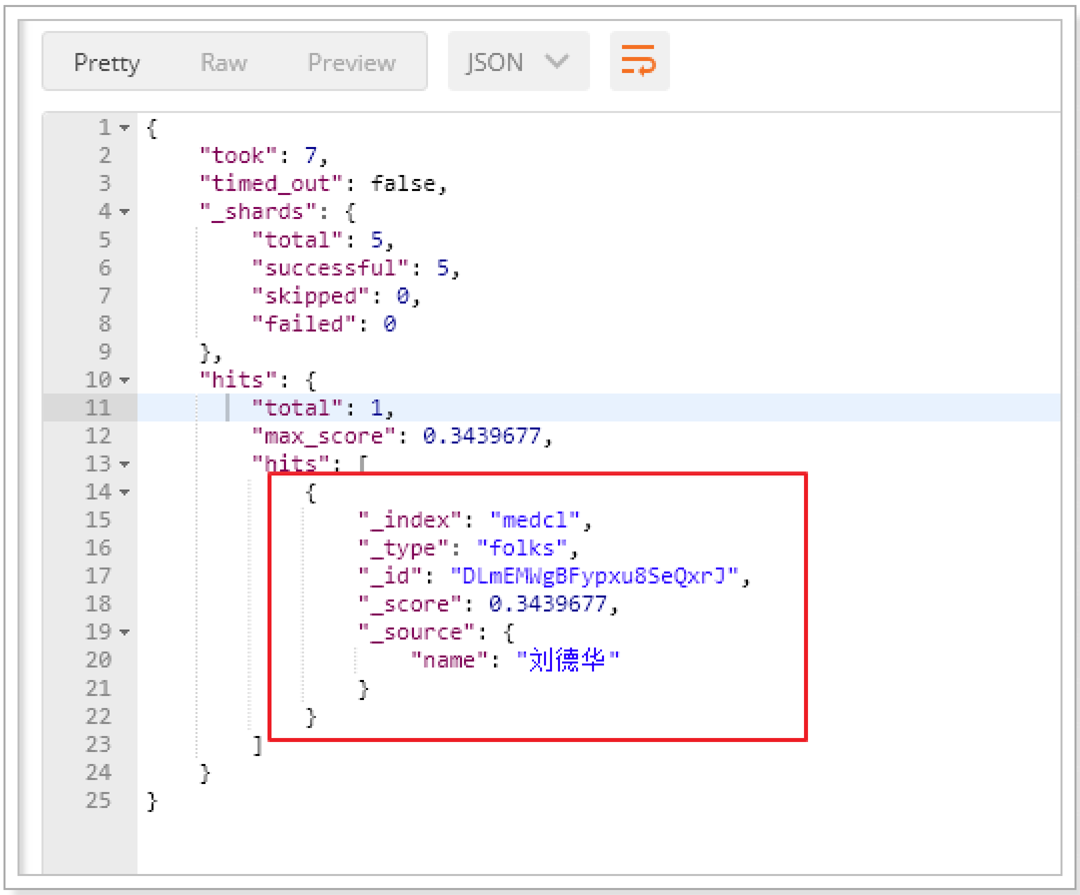
搜索：



http://192.168.1.7:9200/medcl/folks/_search

```
{
    "query":{
        "match":{
            "name.pinyin":{
                "query":"ldh"
            }
        }
    }
}
```

结果：

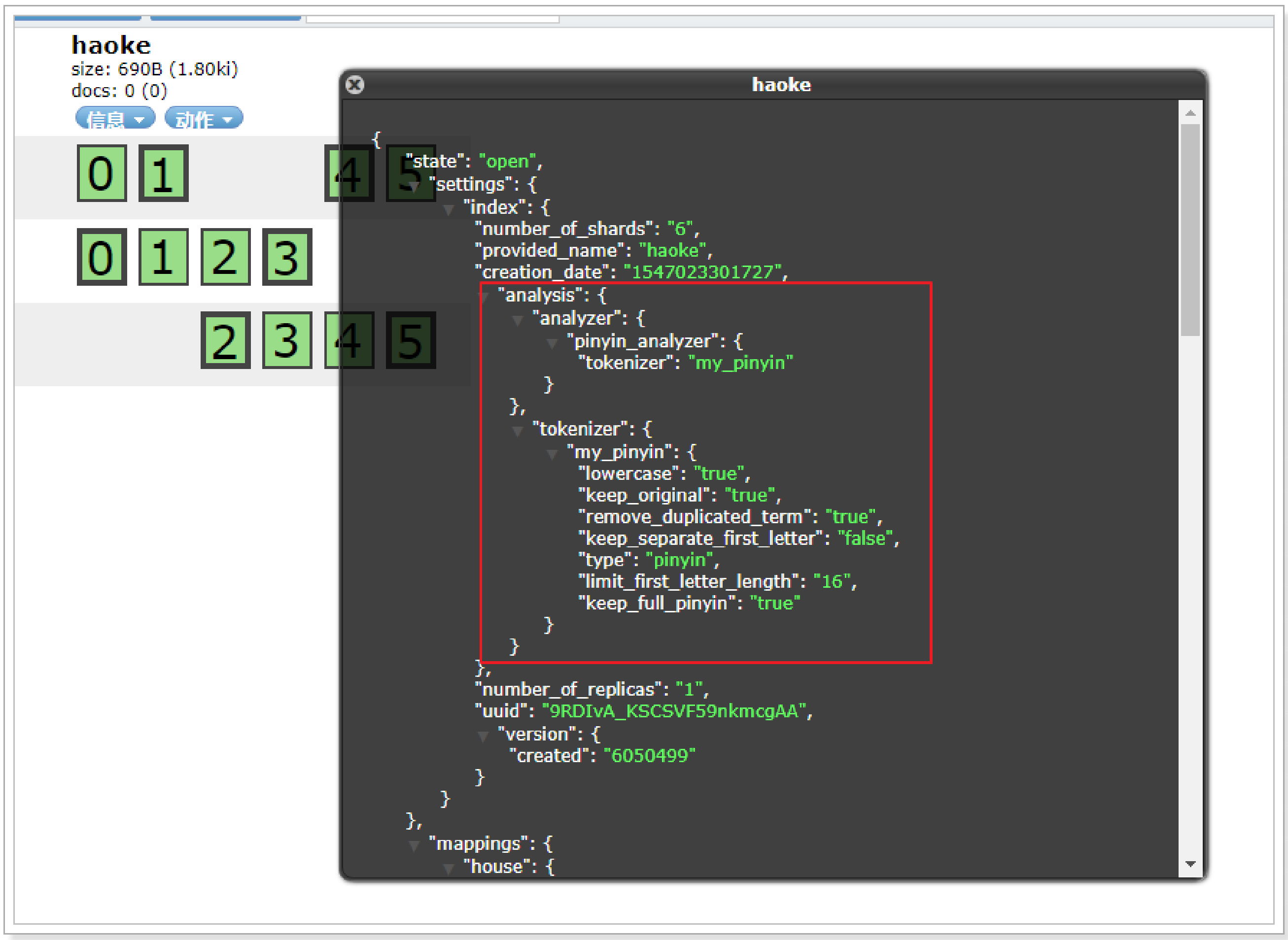


```
{
    "took": 7,
    "timed_out": false,
    "_shards": {
        "total": 5,
        "successful": 5,
        "skipped": 0,
        "failed": 0
    },
    "hits": {
        "total": 1,
        "max_score": 0.3439677,
        "hits": [
            {
                "_index": "medcl",
                "_type": "folks",
                "_id": "DLmEMWgBFypxu8SeQxrJ",
                "_score": 0.3439677,
                "_source": {
                    "name": "刘德华"
                }
            }
        ]
    }
}
```

6.3、房源索引增加拼音支持

```
PUT http://192.168.1.7:9200/haoke/

{
    "settings": {
        "index": {
            "number_of_shards": 6,
            "number_of_replicas": 1,
        "analysis": {
                "analyzer": {
                    "pinyin_analyzer": {
                        "tokenizer": "my_pinyin"
                    }
                },
                "tokenizer": {
                    "my_pinyin": {
                        "type": "pinyin",
                        "keep_separate_first_letter": false,
                        "keep_full_pinyin": true,
                        "keep_original": true,
                        "limit_first_letter_length": 16,
                        "lowercase": true,
                        "remove_duplicated_term": true
                    }
                }
            }
        }
    },
    "mappings": {
        "house": {
            "dynamic": false,
            "properties": {
                "title": {
                    "type": "text",
                    "analyzer":"ik_max_word",
                    "fields":{
                        "pinyin":{
                            "type": "text",
                            "analyzer": "pinyin_analyzer"
                        }
                    }
                },
                "image": {
                    "type": "keyword",
                    "index":false
                },
                "orientation": {
                    "type": "keyword",
                    "index":false
                },
                "houseType": {
                    "type": "keyword",
                    "index":false
                },
                "rentMethod": {
                    "type": "keyword",
                    "index":false
                },
                "time": {
                    "type": "keyword",
                    "index":false
                },
                "rent": {
                    "type": "keyword",
                    "index":false
                },
                "floor": {
                    "type": "keyword",
                    "index":false
                }
            }
        }
    }
}
```

创建完成：



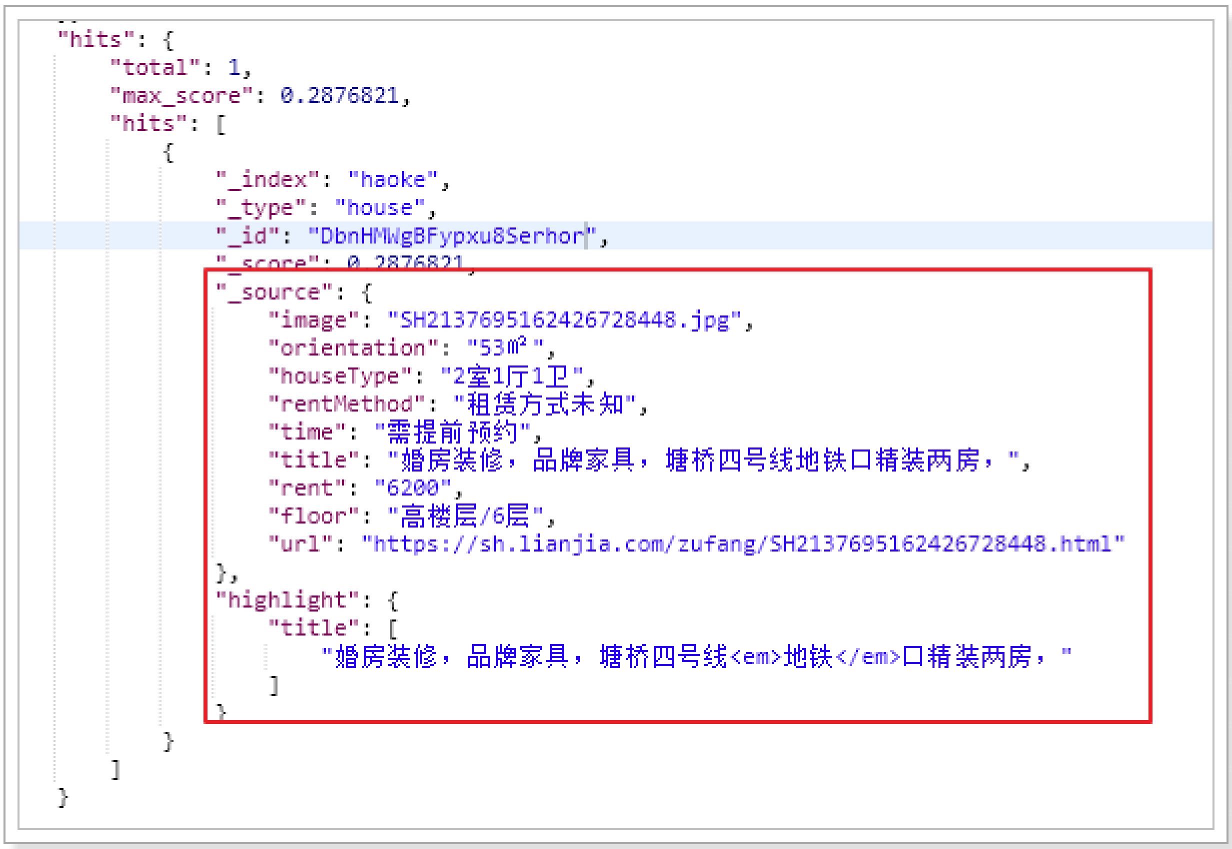
```
{
    "state": "open",
    "settings": {
        "index": {
            "number_of_shards": "6",
            "provided_name": "haoke",
            "creation_date": "1547023301727",
            "analysis": {
                "analyzer": {
                    "pinyin_analyzer": {
                        "tokenizer": "my_pinyin"
                    }
                },
                "tokenizer": {
                    "my_pinyin": {
                        "lowercase": "true",
                        "keep_original": "true",
                        "remove_duplicated_term": "true",
                        "keep_separate_first_letter": "false",
                        "type": "pinyin",
                        "limit_first_letter_length": "16",
                        "keep_full_pinyin": "true"
                    }
                }
            },
            "number_of_replicas": "1",
            "uuid": "9RDIvA_KSCSVF59nkmcgAA",
            "version": {
                "created": "6050499"
            }
        }
    },
    "mappings": {
        "house": {
```

插入测试数据：

```
POST http://192.168.1.7:9200/haoke/house
{
    "image": "SH2137695162426728448.jpg",
    "orientation": "53㎡",
    "houseType": "2室1厅1卫",
    "rentMethod": "租赁方式未知",
    "time": "需提前预约",
    "title": "婚房装修，品牌家具，塘桥四号线地铁口精装两房，",
    "rent": "6200",
    "floor": "高楼层/6层",
    "url": "https://sh.lianjia.com/zufang/SH2137695162426728448.html"
}
```

搜索“地铁”测试：

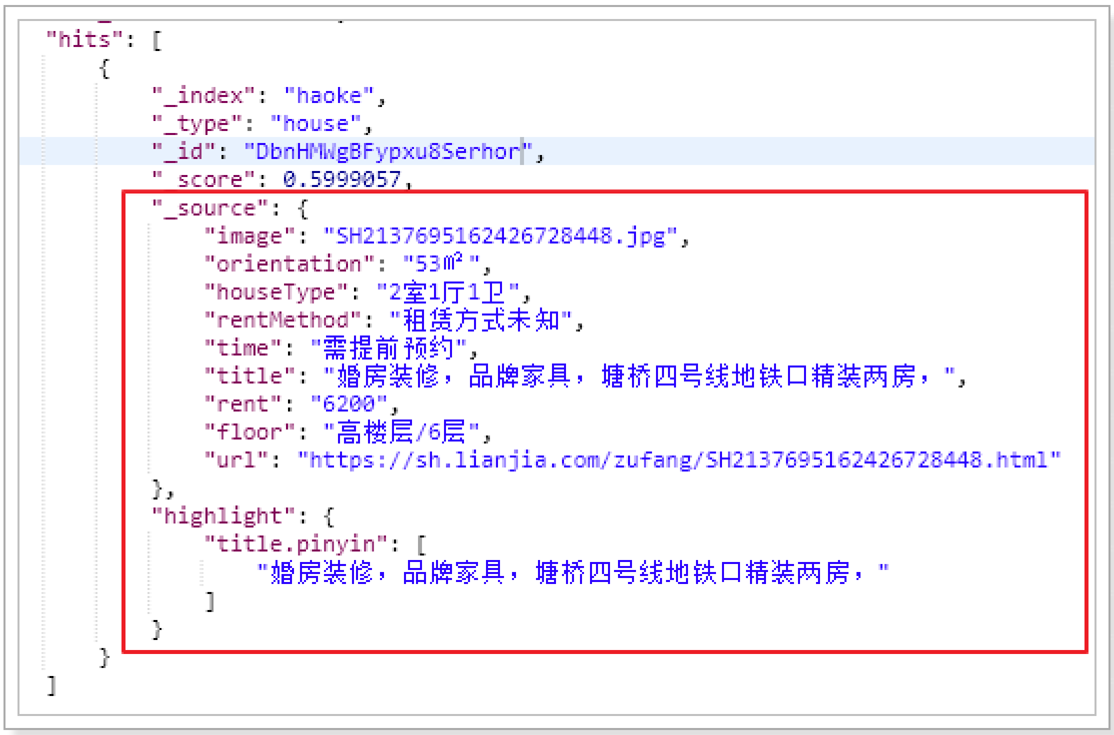
```
POST http://192.168.1.7:9200/haoke/house/_search
{
  "query": {
    "match": {
      "title": "地铁"
    }
  },
  "highlight": {
    "fields": {
      "title": {}
    }
  }
}
```



```
"hits": {
    "total": 1,
    "max_score": 0.2876821,
    "hits": [
        {
            "_index": "haoke",
            "_type": "house",
            "_id": "DbnHMWgBFypxu8Serhor",
            "_score": 0.2876821,
            "_source": {
                "image": "SH2137695162426728448.jpg",
                "orientation": "53㎡",
                "houseType": "2室1厅1卫",
                "rentMethod": "租赁方式未知",
                "time": "需提前预约",
                "title": "婚房装修，品牌家具，塘桥四号线地铁口精装两房，",
                "rent": "6200",
                "floor": "高楼层/6层",
                "url": "https://sh.lianjia.com/zufang/SH2137695162426728448.html"
            },
            "highlight": {
                "title": [
                    "婚房装修，品牌家具，塘桥四号线<em>地铁</em>口精装两房，"
                ]
            }
        }
    ]
}
```

测试“ditie”搜索：

```
POST http://192.168.1.7:9200/haoke/house/_search
{
  "query": {
    "match": {
      "title.pinyin": "ditie"
    }
  },
  "highlight": {
    "fields": {
      "title.pinyin": {}
    }
  }
}
```

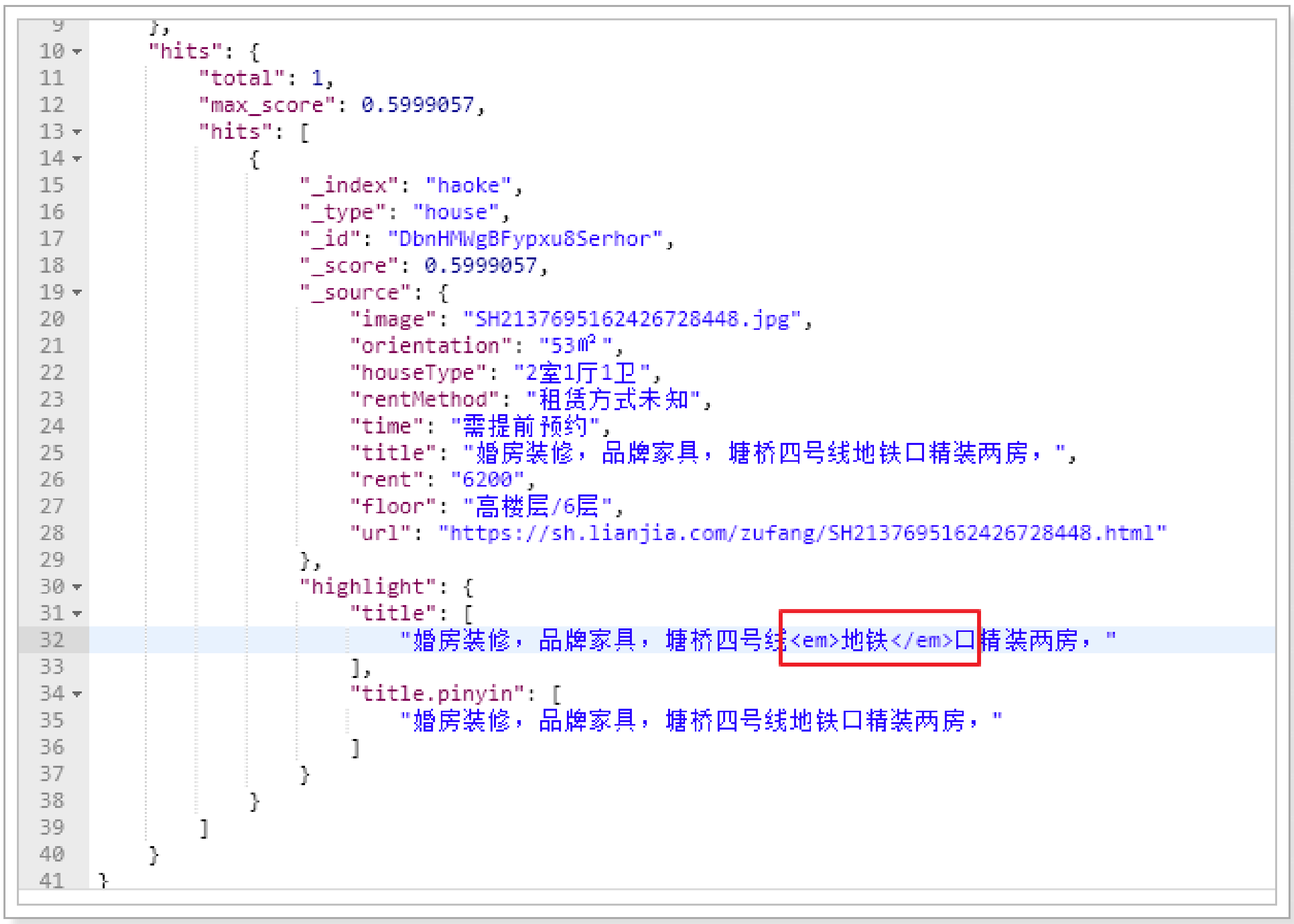


```
"hits": [
    {
        "_index": "haoke",
        "_type": "house",
        "_id": "DbnHMWgBFypxu8Serhor",
        "_score": 0.5999057,
        "_source": {
            "image": "SH2137695162426728448.jpg",
            "orientation": "53㎡",
            "houseType": "2室1厅1卫",
            "rentMethod": "租赁方式未知",
            "time": "需提前预约",
            "title": "婚房装修，品牌家具，塘桥四号线地铁口精装两房，",
            "rent": "6200",
            "floor": "高楼层/6层",
            "url": "https://sh.lianjia.com/zufang/SH2137695162426728448.html"
        },
        "highlight": {
            "title.pinyin": [
                "婚房装修，品牌家具，塘桥四号线地铁口精装两房，"
            ]
        }
    }
]
```

可以看到，通过拼音也可以搜索到数据，但是高亮显示中没有把拼音对应的中文高亮。

中文和拼音混合搜索：

```
POST http://192.168.1.7:9200/haoke/house/_search
{
    "query": {
        "multi_match": {
            "query":"地铁kou",
            "fields":["title","title.pinyin"]
        }
    },
    "highlight": {
        "fields": {
            "title.pinyin": {},
            "title": {}
            }
        }
}
```



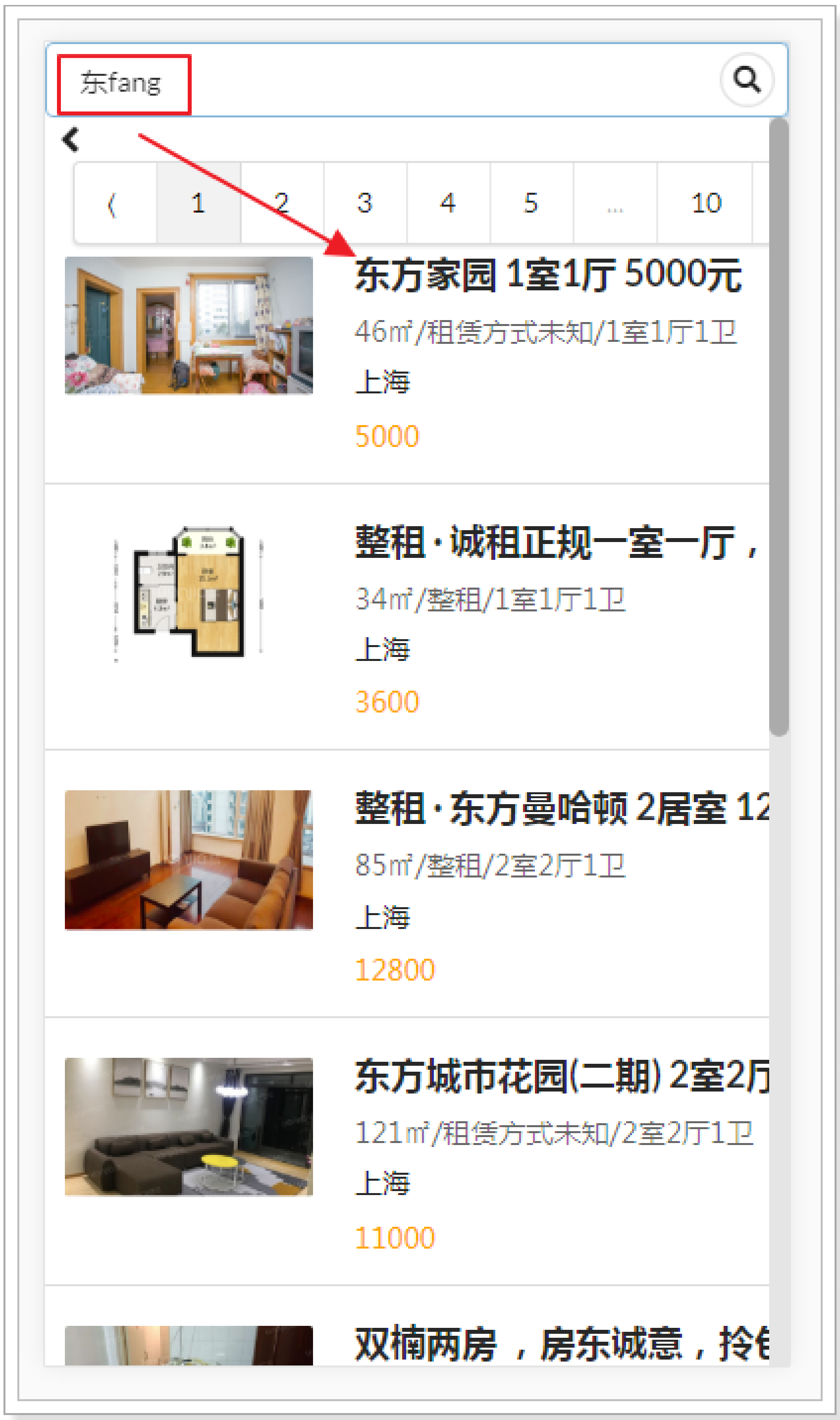
```
    },
    "hits": {
        "total": 1,
        "max_score": 0.5999057,
        "hits": [
            {
                "_index": "haoke",
                "_type": "house",
                "_id": "DbnHMWgBFypxu8Serhor",
                "_score": 0.5999057,
                "_source": {
                    "image": "SH2137695162426728448.jpg",
                    "orientation": "53㎡",
                    "houseType": "2室1厅1卫",
                    "rentMethod": "租赁方式未知",
                    "time": "需提前预约",
                    "title": "婚房装修，品牌家具，塘桥四号线地铁口精装两房，",
                    "rent": "6200",
                    "floor": "高楼层/6层",
                    "url": "https://sh.lianjia.com/zufang/SH2137695162426728448.html"
                },
                "highlight": {
                    "title": [
                        "婚房装修，品牌家具，塘桥四号线<em>地铁</em>口精装两房，"
                    ],
                    "title.pinyin": [
                        "婚房装修，品牌家具，塘桥四号线地铁口精装两房，"
                    ]
                }
            }
        ]
    }
}
```

6.4、重新导入数据

重新导入数据以及修改查询逻辑：

```java
SearchQuery searchQuery = new NativeSearchQueryBuilder()
        .withQuery(QueryBuilders.multiMatchQuery(keyword, "title",
"title.pinyin").operator(Operator.AND)) // match查询
        .withPageable(pageRequest)
        .withHighlightFields(new HighlightBuilder.Field("title")) // 设置高亮
        .build();
```

测试：



东fang

〈 1 2 3 4 5 ... 10

东方家园 1室1厅 5000元
46㎡/租赁方式未知/1室1厅1卫
上海
5000

整租·诚租正规一室一厅，
34㎡/整租/1室1厅1卫
上海
3600

整租·东方曼哈顿 2居室 12
85㎡/整租/2室2厅1卫
上海
12800

东方城市花园(二期) 2室2厅
121㎡/租赁方式未知/2室2厅1卫
上海
11000

双楠两房 ，房东诚意，拎包