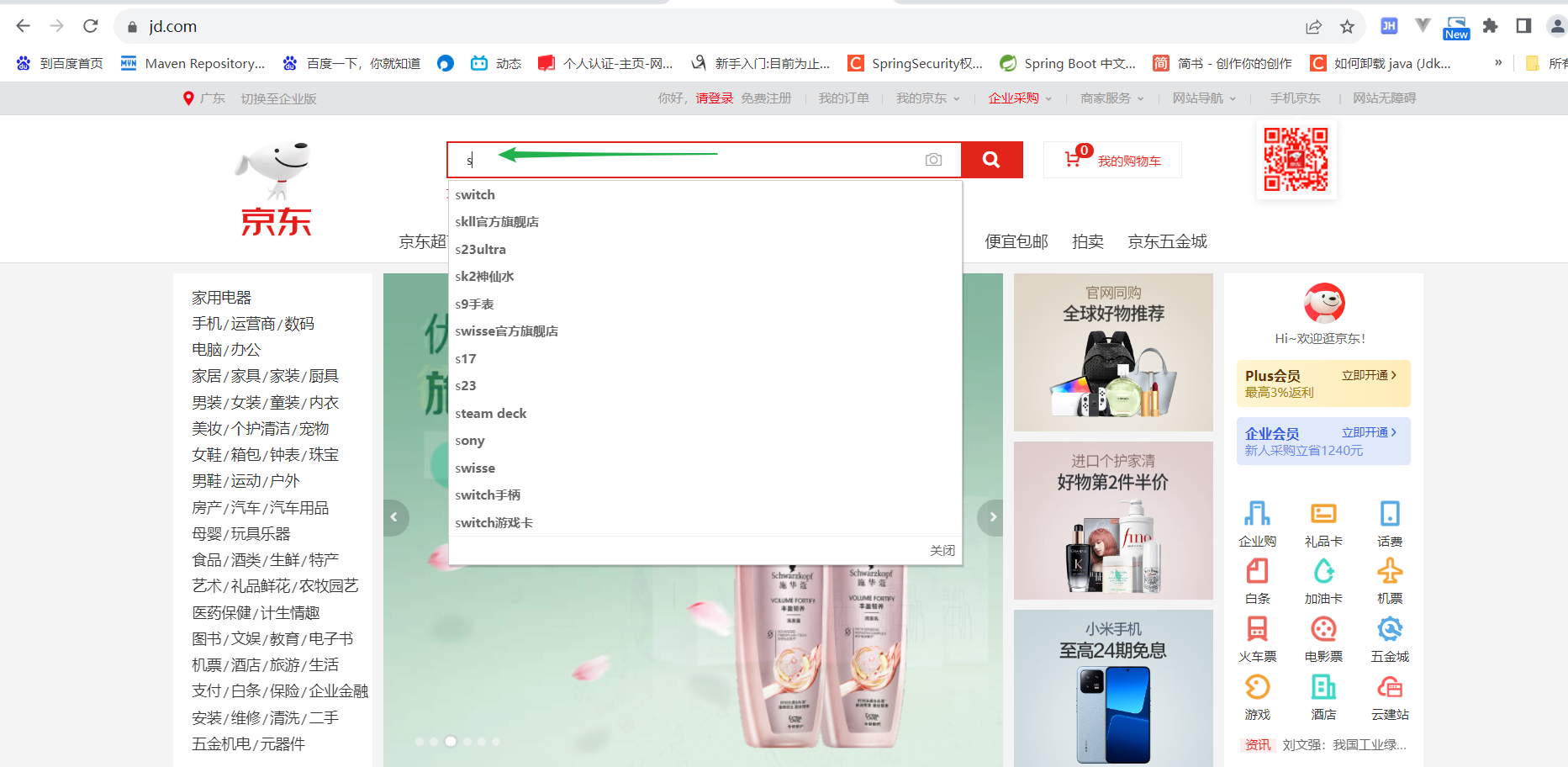
# 自动补全查询

## 自动补全介绍

同学们，接下来我们学习自动补全查询，什么是自动补全呢？我们现看一下效果，比如我们在京东网站搜索框中输入想要搜索的内容，我们只需要输入一部分字符，你就会看到京东自动给我们弹出很多内容！如下图，这就是自动不全了！

自动补全可以很方便地让用户尽可能的搜索到想要的东西，而不需要用户去搜索框输入完整的内容！

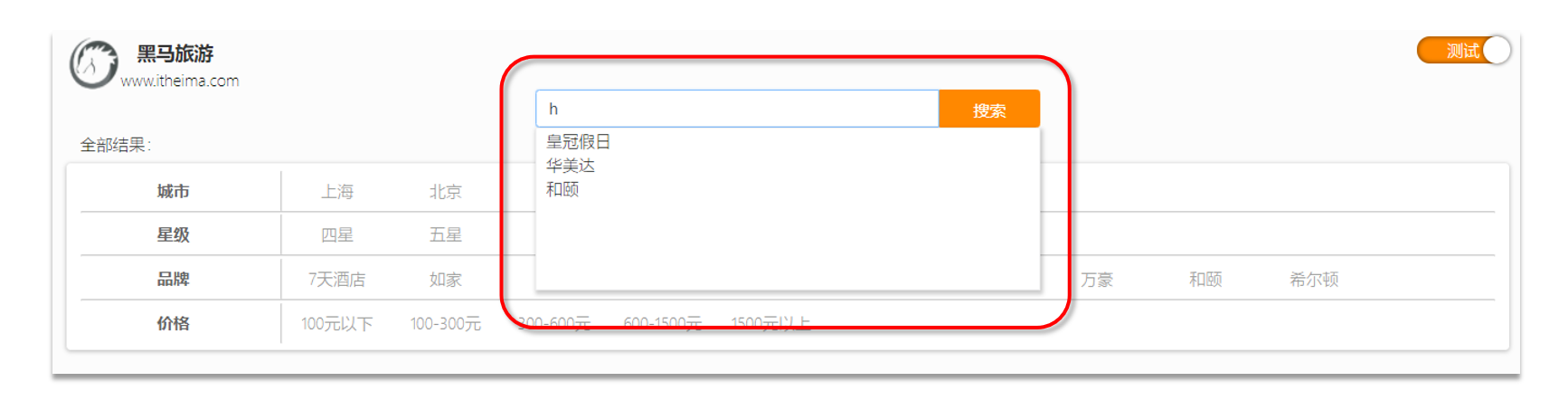


要想实现这个自动补全功能，就必须先学系一下拼音分词器！

## 拼音分词器

我们中国人实际场景里，可以看到是用拼音的方式去做补全和提示的，因为我们绝大多数人用的是拼音输入法。我们还需要对拼音分词器做一些自定义的配置，形成自己的自定义分词器！

自定义分词器搞定了之后，我们就可以去学习ES中的自动补全功能了，最后我们会利用学习的自动补全功能去实现酒店搜索框的自动补全！



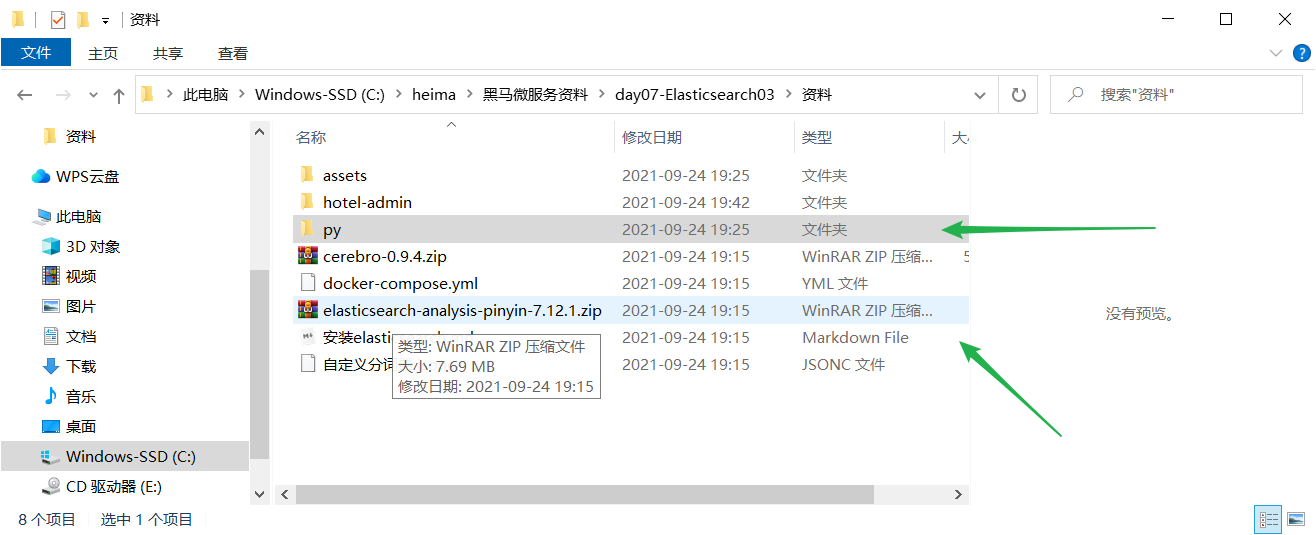
## 拼音分词器安装

要实现根据字母做补全，就必须对文档按照拼音分词。

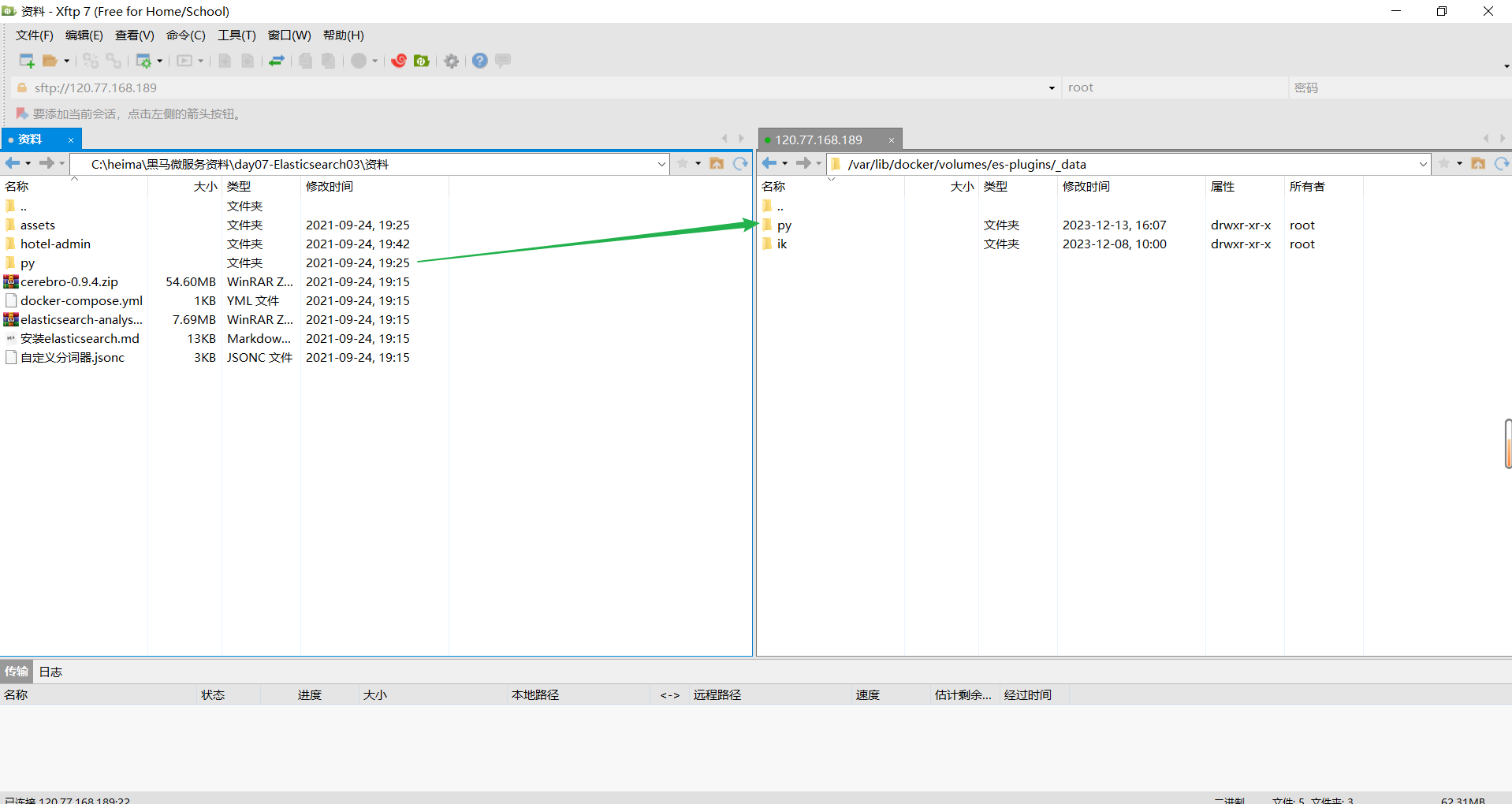
在GitHub上恰好有ElasticSearch的拼音分词插件。官方地址为：<https://github.com/medcl/elasticsearch-analysis-pinyin>

拼音分词器安装方式与IK分词器一样，分三步：

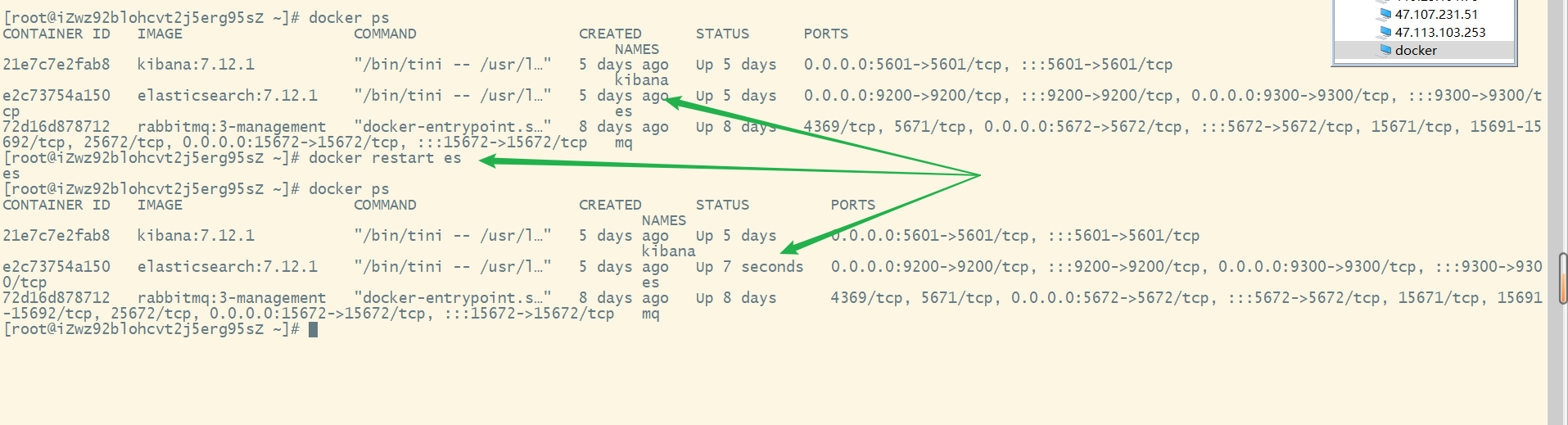
### 解压拼音分词器插件



### 上传到虚拟机中，elasticsearch的plugin目录



### 重启elasticsearch容器



### 在DevTools测试拼音分词器

# 测试py分词器

**POST /\_analyze**

**{**

**"text": ["如家酒店还不错！"],**

**"analyzer": "pinyin"**

**}**

### 在DevTools控制台查看结果

可以看到每一个字一分，分成一个拼音，还有每个首字母拼在一起！这是拼音分词器分词的两种方式！

**{**

**"tokens" : [**

**{**

**"token" : "ru",**

**"start\_offset" : 0,**

**"end\_offset" : 0,**

**"type" : "word",**

**"position" : 0**

**},**

**{**

**"token" : "rjjdhbc",**

**"start\_offset" : 0,**

**"end\_offset" : 0,**

**"type" : "word",**

**"position" : 0**

**},**

**{**

**"token" : "jia",**

**"start\_offset" : 0,**

**"end\_offset" : 0,**

**"type" : "word",**

**"position" : 1**

**},**

**{**

**"token" : "jiu",**

**"start\_offset" : 0,**

**"end\_offset" : 0,**

**"type" : "word",**

**"position" : 2**

**},**

**{**

**"token" : "dian",**

**"start\_offset" : 0,**

**"end\_offset" : 0,**

**"type" : "word",**

**"position" : 3**

**},**

**{**

**"token" : "hai",**

**"start\_offset" : 0,**

**"end\_offset" : 0,**

**"type" : "word",**

**"position" : 4**

**},**

**{**

**"token" : "bu",**

**"start\_offset" : 0,**

**"end\_offset" : 0,**

**"type" : "word",**

**"position" : 5**

**},**

**{**

**"token" : "cuo",**

**"start\_offset" : 0,**

**"end\_offset" : 0,**

**"type" : "word",**

**"position" : 6**

**}**

**]**

**}**

### 总结：

**这就是拼音分词器的基本用法了，以后我们就可以在创建索引库的时候，通过mapping映射去定义拼音分词器，作为字段的分词器了！**

## 自定义分词器

朋友们，我们上面已经学习了拼音分词器的基本用法，但是这个时候的拼音分词器还没有办法直接应用在生产环境！还存在一些问题，比如我们上面测试拼音分词器的时候，可以看到分词后的结果是**每个首字母拼在一起，这就是拼音分词器的第一个问题：不会分词！第二个问题就是拼音分词器还把每一个字都形成了一个拼音，是没什么用的。第三个问题是分词结果没有汉字，只剩下了拼音，其实我们用拼音搜索的情况是占少数的，大多数情况下我们是想用中文去搜索的，所以说分词后有拼音是锦上添花，但分此后你不能把汉字扔了啊，汉字也得保留啊！**

以上上拼音分词器面临的几个问题！那我们就必须得对拼音分词器做一些配置或者自定义了！那怎么样才能自定义分词器呢？请看下面。

### ES分词器结构

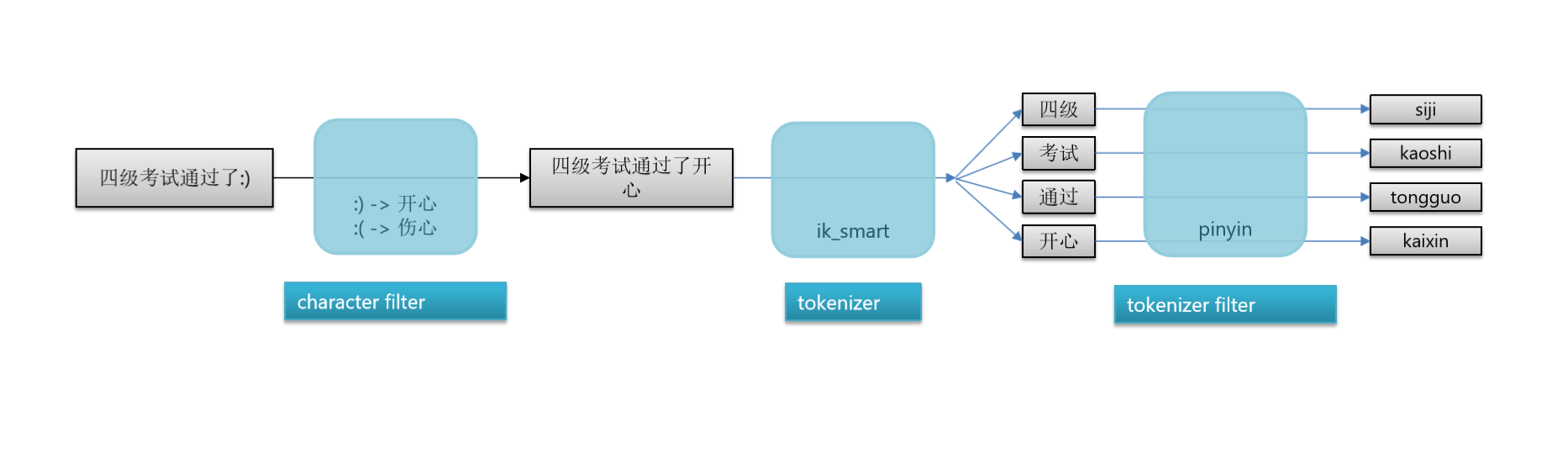
要想自定义分词器，你必须先了解ES中分词器的构成，一个分词器包含三部分：

**1：character filters：**在tokenizer之前对文本进行处理，例如删除字符、替换字符等；

**2：tokenizer：**将文本按照一定的规则切割成词条（term），是真正的分词的地方，例如keyword、ilk\_smart

**3：tokenizer filter：**将tokenizer输出的词条进一步处理。例如大小写转换、同义词处理、拼音处理等

**总结来说，分词器分词的过程就是：在分词前对内容前置处理；对内容按发按分此规则分词；对分词后得到的词条后置处理。见下图:**



你看，这个时候不是解决了我们上面提出的拼音分词器存在问题了吗？

你拼音分词器不会分词，那我先在这个tokenizer部分用ik\_smart对内容分词，再在tokenizer filter部分用pinyin做过滤处理，形成拼音不就行了吗！ 那到底该怎么实现这样的自定义分词器呢？请看语法：

### 自定义分词器DSL语法

我们可以在创建索引库的时候，通过setting来配置自定义analyser分词器。

语法其实非常简单，

1：要想实现自定义分词器，一定是在创建索引库的时候，于是可以看到PUT /test，我们创建一个test索引库；

2：自定义分词器的配置都是放在setting属性里去配置的；

3：又因为自定义分词器是在创建索引库的时候设置的，所以这个自定义分词器只会对当前索引库有效；

4：analyzer就是自定义分词器，分词器的名字就是my\_analyzer

5：自定义分词器由三部分组成，这里没有去定义character filters了，只定义了tokenizer和tokenizer filter，分别是ik\_smart和pinyin，character 是对特殊字符做处理的，我们又没有特殊字符，所以自定义分词器的时候不一定三部分都定义



以上只解决了拼音分词器的一个问题：就是拼音分词器不分词的问题，但是我们讲拼音分词器害有两个问题：一是会把分好的词语转成拼音的时候是一个字一个字转成拼音，而是不保留中文。

所以我们直接交给拼音过滤器处理是有问题的，我们需要对拼音过滤器做进一步的定制，见下图：

可以看到我们在my\_analyzer中的filter定义成了py，意思就是引用了下面filter中定义的py自定义过滤器，这个py过滤器的本质还是pinyin，但是我们在指定拼音时指定各种属性！



注意，我们还需要对tokenizer filter的pinyin

### 自定义拼音分词器DSL语句

**PUT /test**

**{**

**"settings": {**

**"analysis": {**

**"analyzer": {**

**"my\_analyzer": { // 自定义分词器**

**"tokenizer": "ik\_max\_word",**

**"filter": "py"**

**}**

**},**

**"filter": { // 自定义tokenizer filter**

**"py": { // 自定义过滤器名字**

**"type": "pinyin", // 过滤器类型，这里时pinyin**

**"keep\_full\_pinyin": false,**

**"keep\_joined\_full\_pinyin": true,**

**"keep\_original": true,**

**"limit\_first\_letter\_length": 16,**

**"remove\_duplicated\_term": true,**

**"none\_chinese\_pinyin\_tokenize": false**

**}**

**}**

**}**

**},**

**"mappings": {**

**"properties": {**

**"name": {**

**"type": "text",**

**"analyzer": "my\_analyzer"**

**}**

**}**

**}**

**}**

## 自定义分词器演示流程：

### 1：创建索引库test

定义自定义分词器my\_analyzer，my\_analyzer自定义分词器中的tokenizer filter使用我们自定义的filter：proyin\_pro，即对加强升级版的拼音分词器，在mapping映射定义name字段的时候使用我们的自定义分词器my\_analyzer！

**PUT /test**

**{**

**"settings": {**

**"analysis": {**

**"analyzer": {**

**"my\_analyzer": {**

**"tokenizer": "ik\_max\_word",**

**"filter": "pinyin\_pro"**

**}**

**},**

**"filter": {**

**"pinyin\_pro": {**

**"type": "pinyin",**

**"keep\_full\_pinyin": false,**

**"keep\_joined\_full\_pinyin": true,**

**"keep\_original": true,**

**"limit\_first\_letter\_length": 16,**

**"remove\_duplicated\_term": true,**

**"none\_chinese\_pinyin\_tokenize": false**

**}**

**}**

**}**

**},**

**"mappings": {**

**"properties": {**

**"name": {**

**"type": "text",**

**"analyzer": "my\_analyzer"**

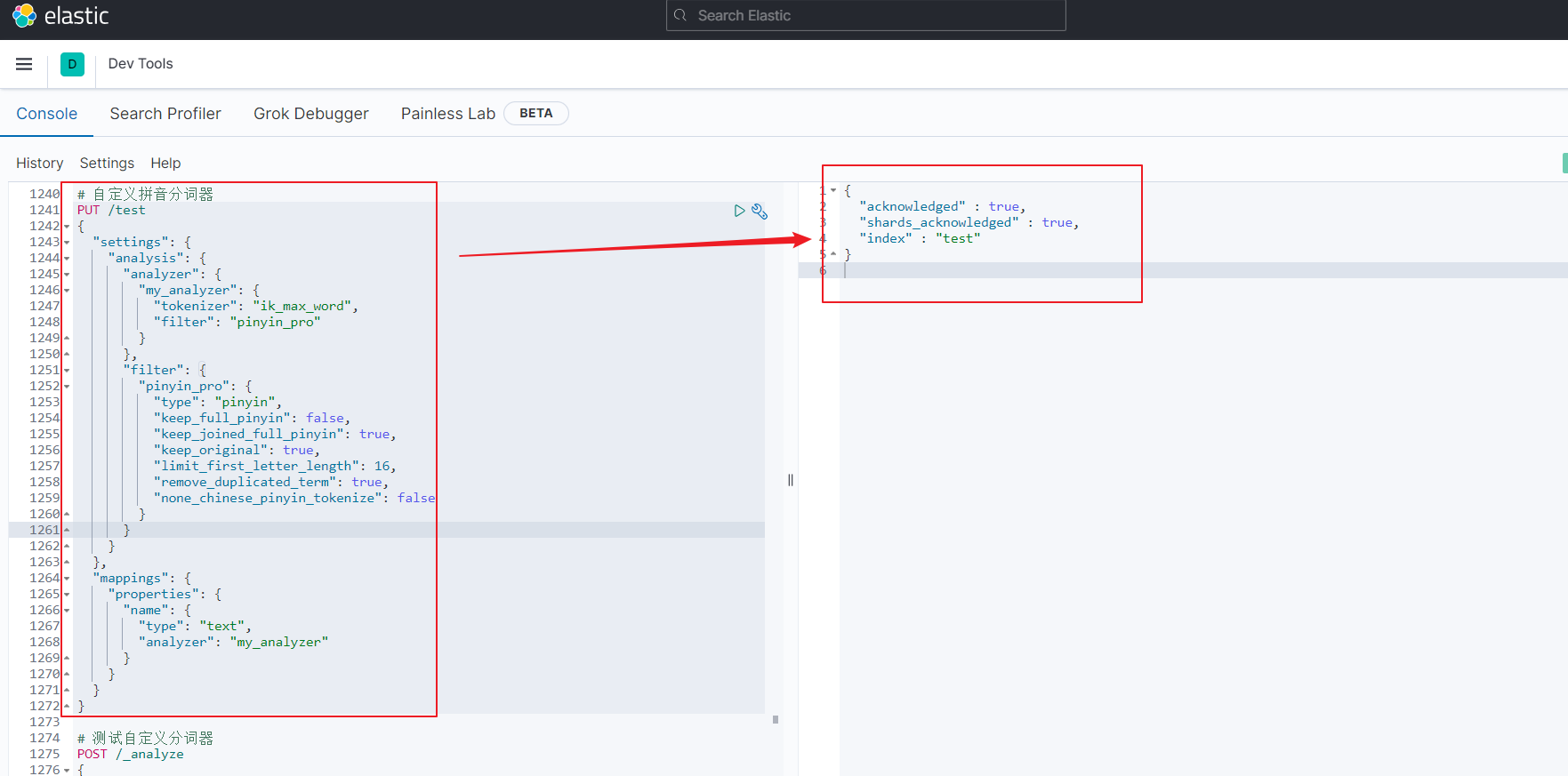
**}**

**}**

**}**

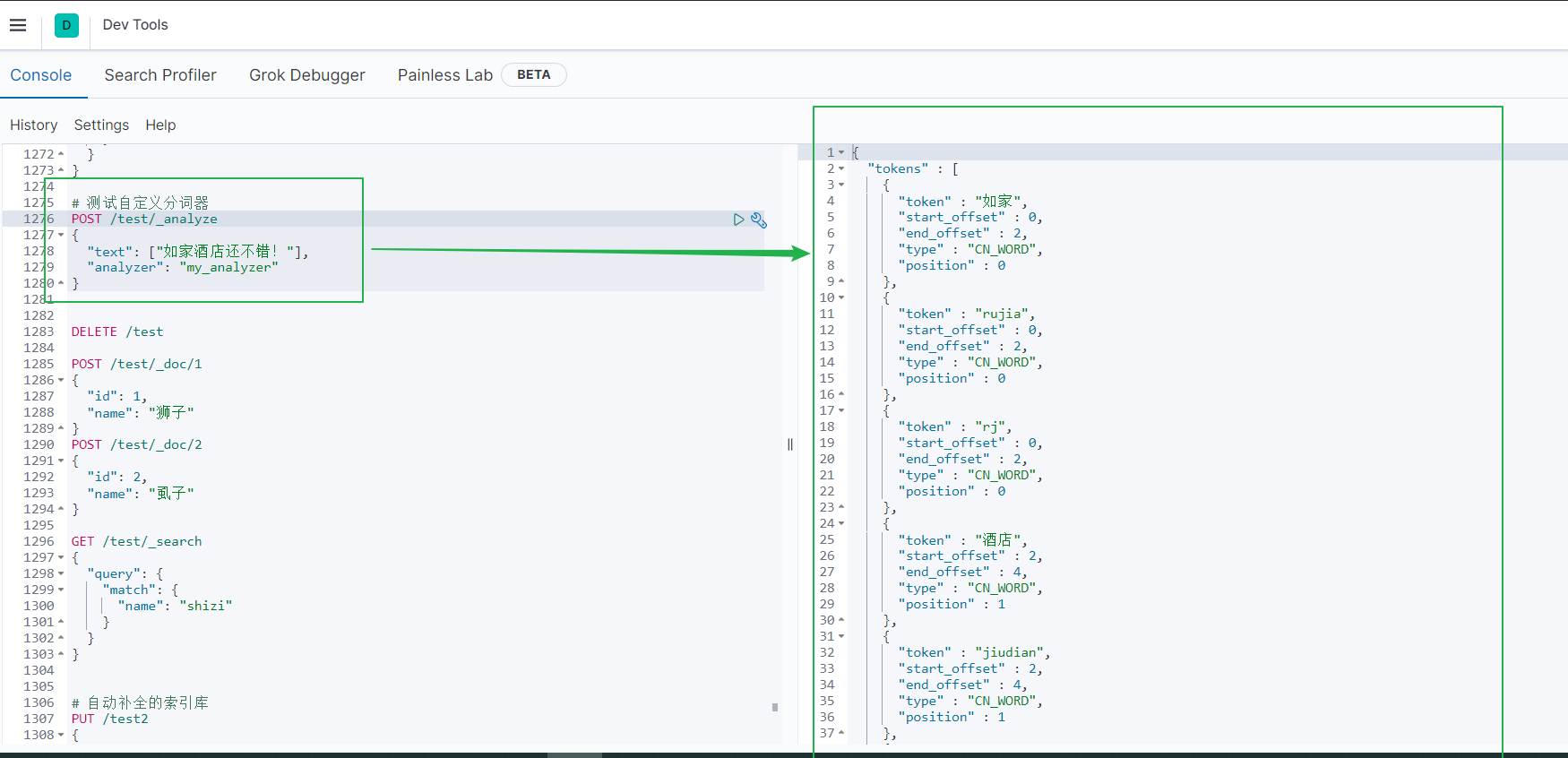
**}**

### 2：创建索引库test

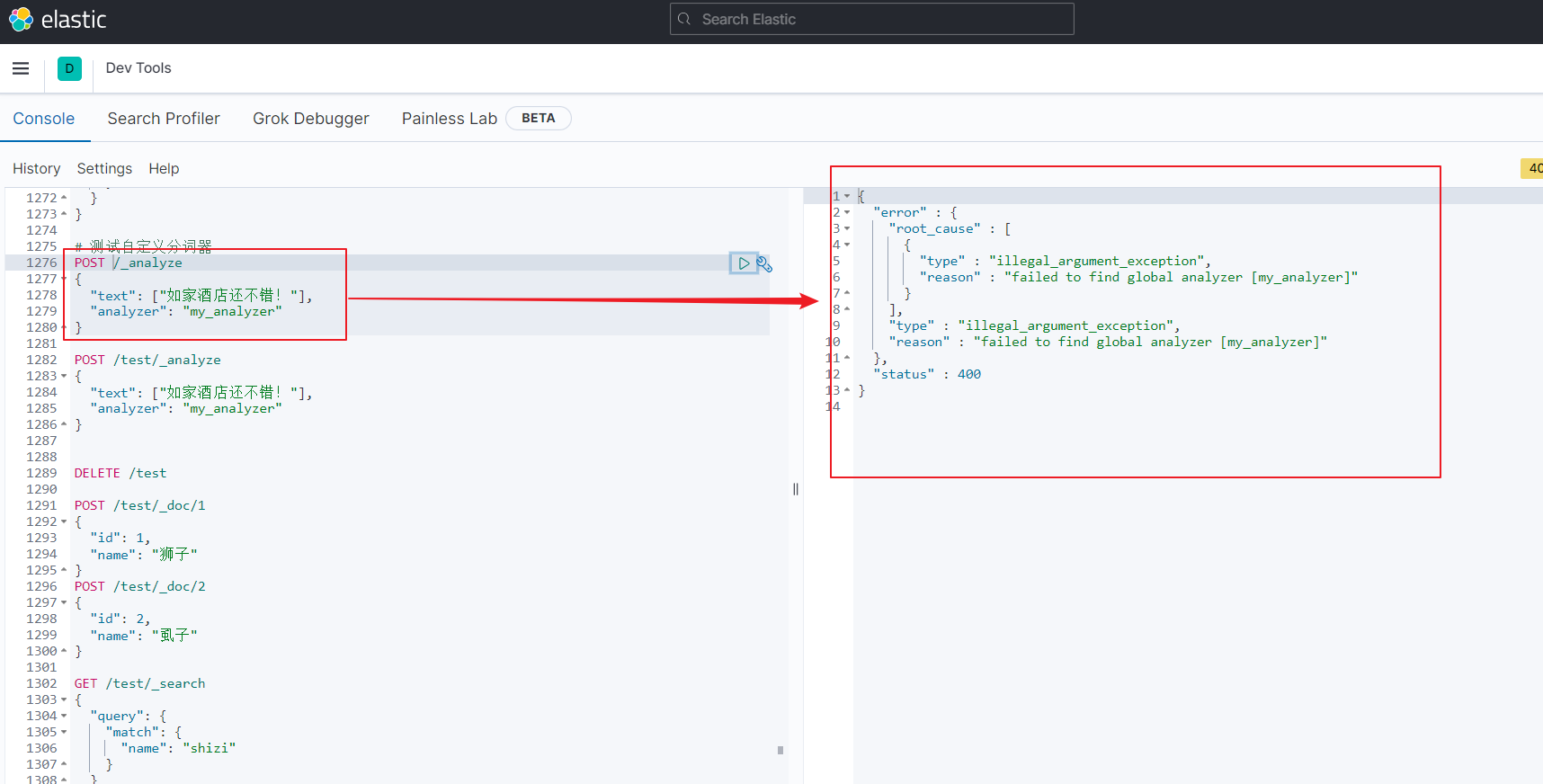


### 3：测试自定义分词器效果

可以看到，完美！



注意，使用POST /\_analyze是会报错的，当然会报错啊，上面已经解释过了，在setting中定义自定义分词器只针对当前索引库有效！要使用自定义分词器分词，需要指明对应索引库！



### 4：给索引库test添加两个文档

可以看到我们像索引库添加了name为狮子和虱子的文档，拼音是一样的，字段name会使用分词器my\_analyser做分词，然后做倒排索引。

**POST /test/\_doc/1**

**{**

**"id": 1,**

**"name": "狮子"**

**}**

**POST /test/\_doc/2**

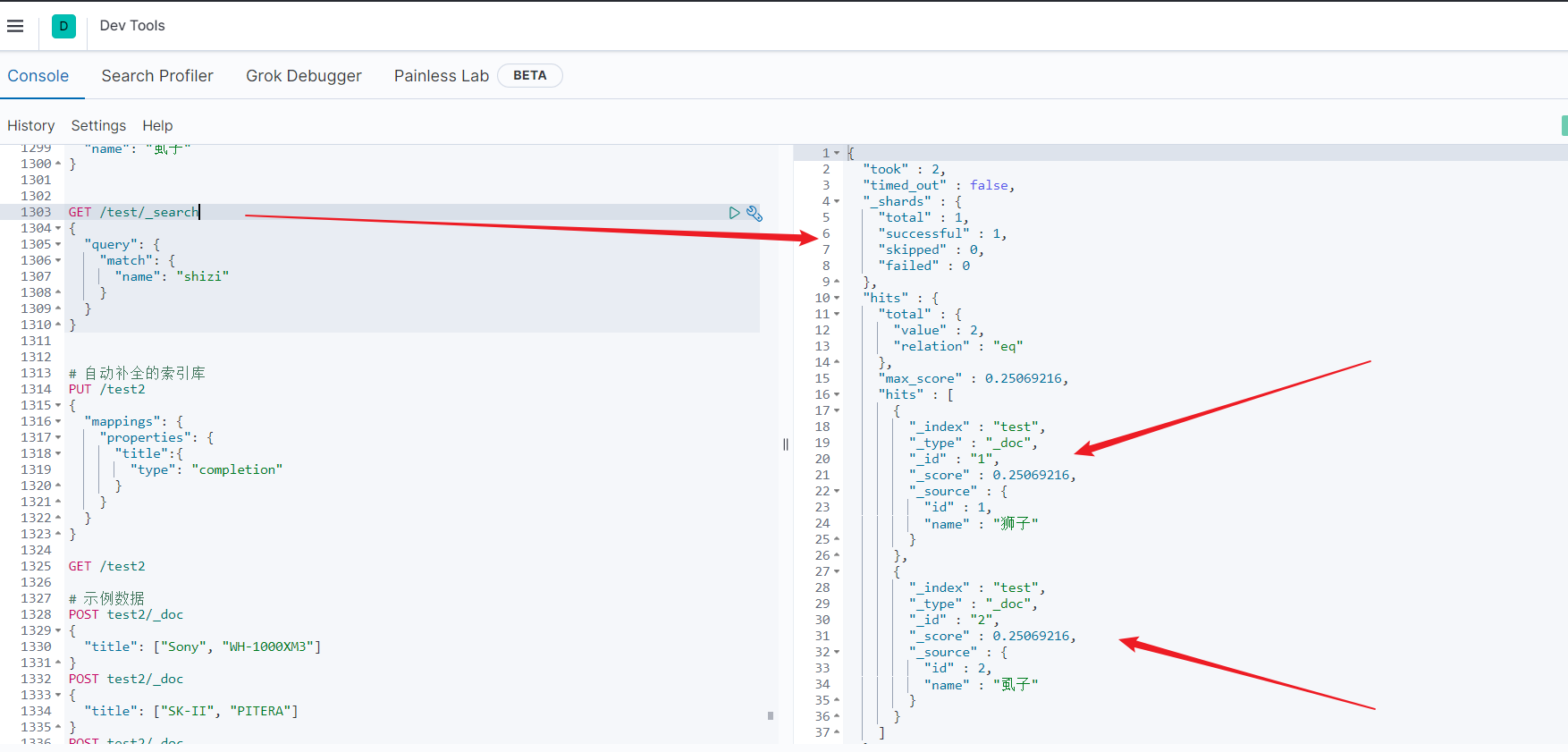
**{**

**"id": 2,**

**"name": "虱子"**

**}**

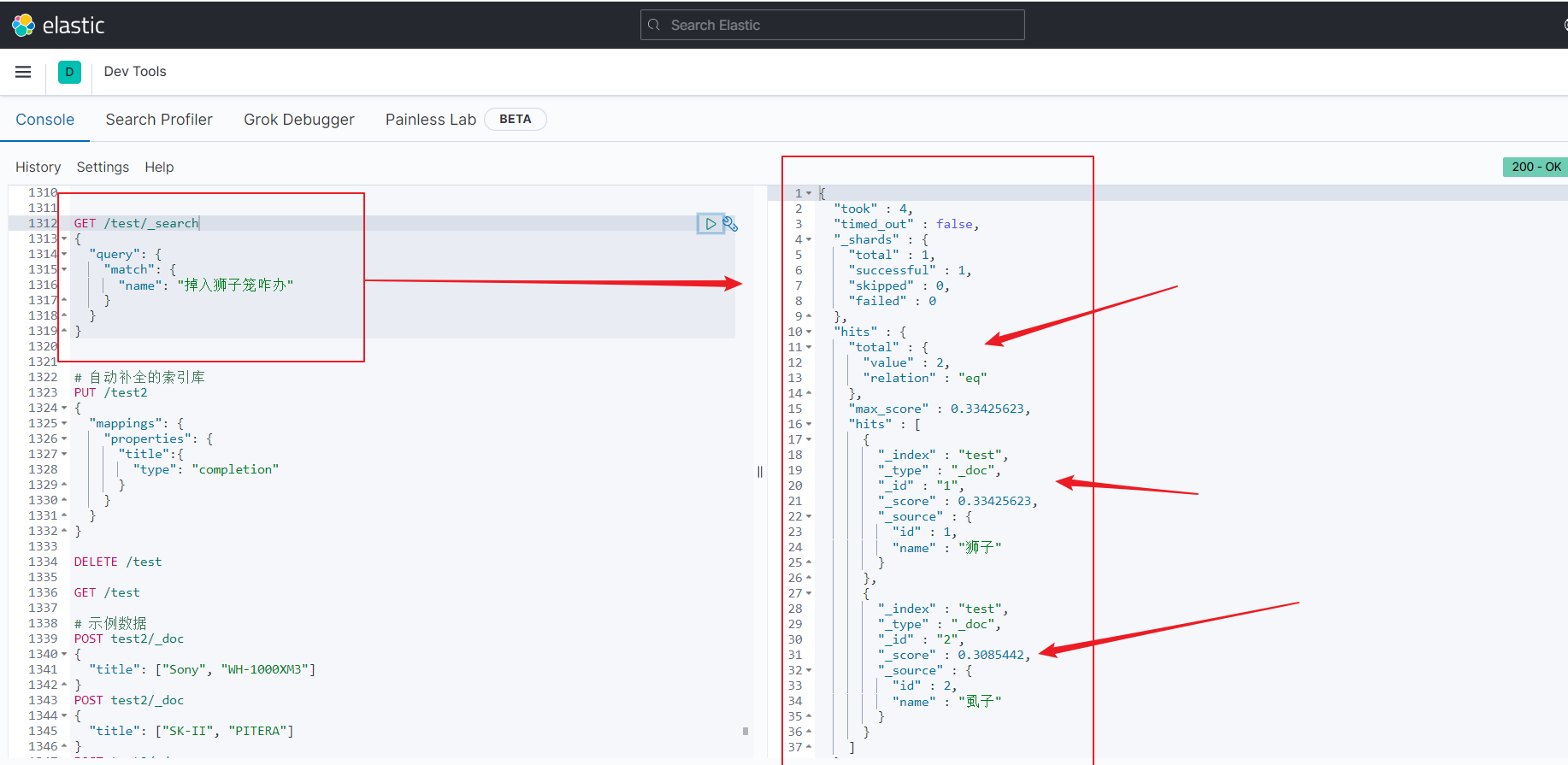
### 5：使用拼音shizi去索引库test中搜索一下，会发现两个文档都能搜索到！

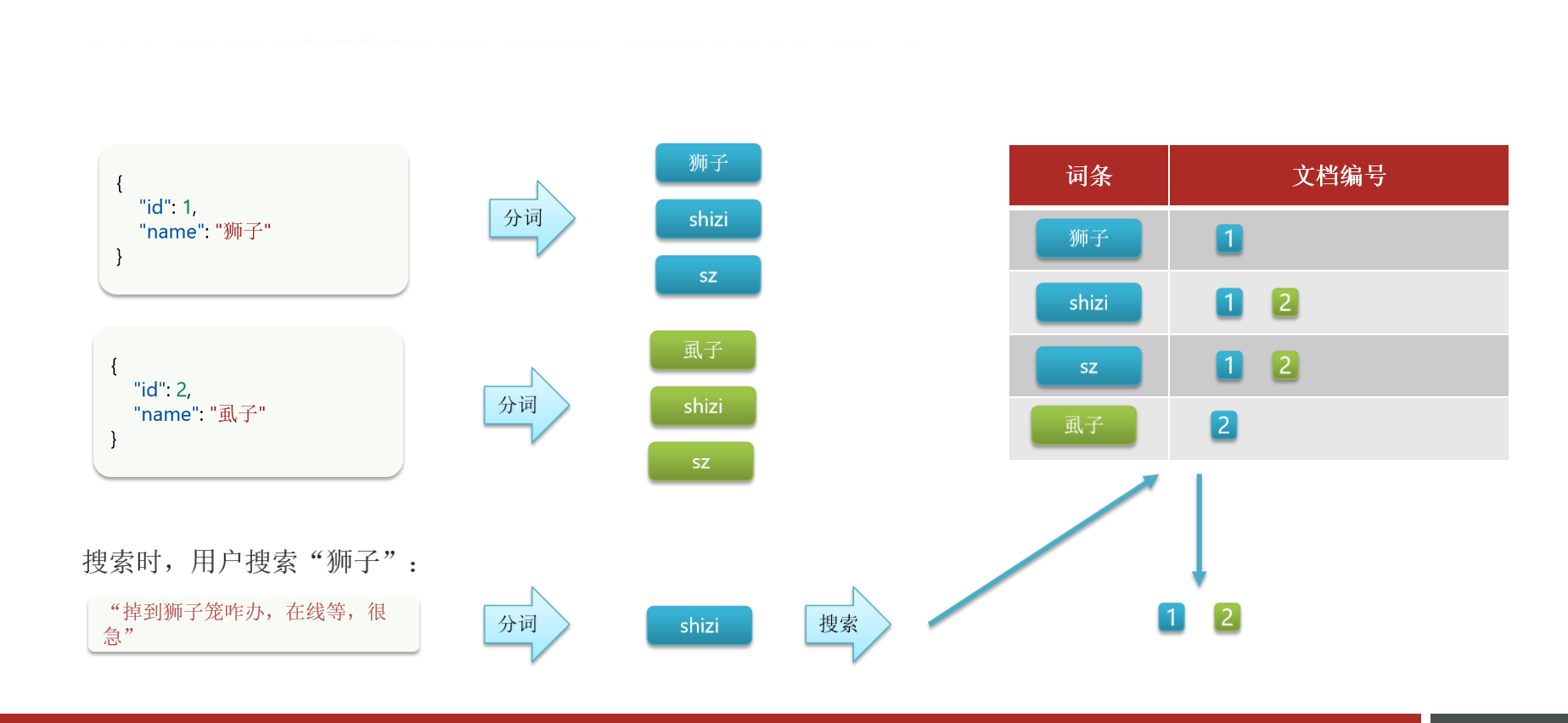


## 拼音分词器注意事项

拼音分词器适合在创建倒排索引的时候使用，但不能在搜索的时候使用。为甚？

比如，向索引库test插入狮子文档，你创建倒排索引的时候，按照我们的分词方式my\_analyzer，字段name的内容“狮子”会分成三个字条：狮子、shizi、sz。然后在创建成倒排索引形成词条和文档编号，但是如果你继续插入虱子文档，因为虱子和狮子的拼音是一样的，所以虱子分词后的词条，除了中文之外，剩下的“shizi”、“sz”都一样！将来再往倒排索引表中插的时候虱子作为全新词条，而因为”shizi“，”sz“已经存在了，就只会记录新的文档ID。因此就会导致“狮子”和“虱子”这两个文档的拼音词条是一样的，这个时候如果别人搜索时，搜“掉到狮子笼咋办”，会分词出“shizi”这个拼音，一艘就能搜出两个文档了！原因就是你在搜索时你也用了拼音分词器，中文“狮子”也处理成了拼音“shizi”，你拿拼音去搜，可不就把“虱子”文档也搜索出来了吗！





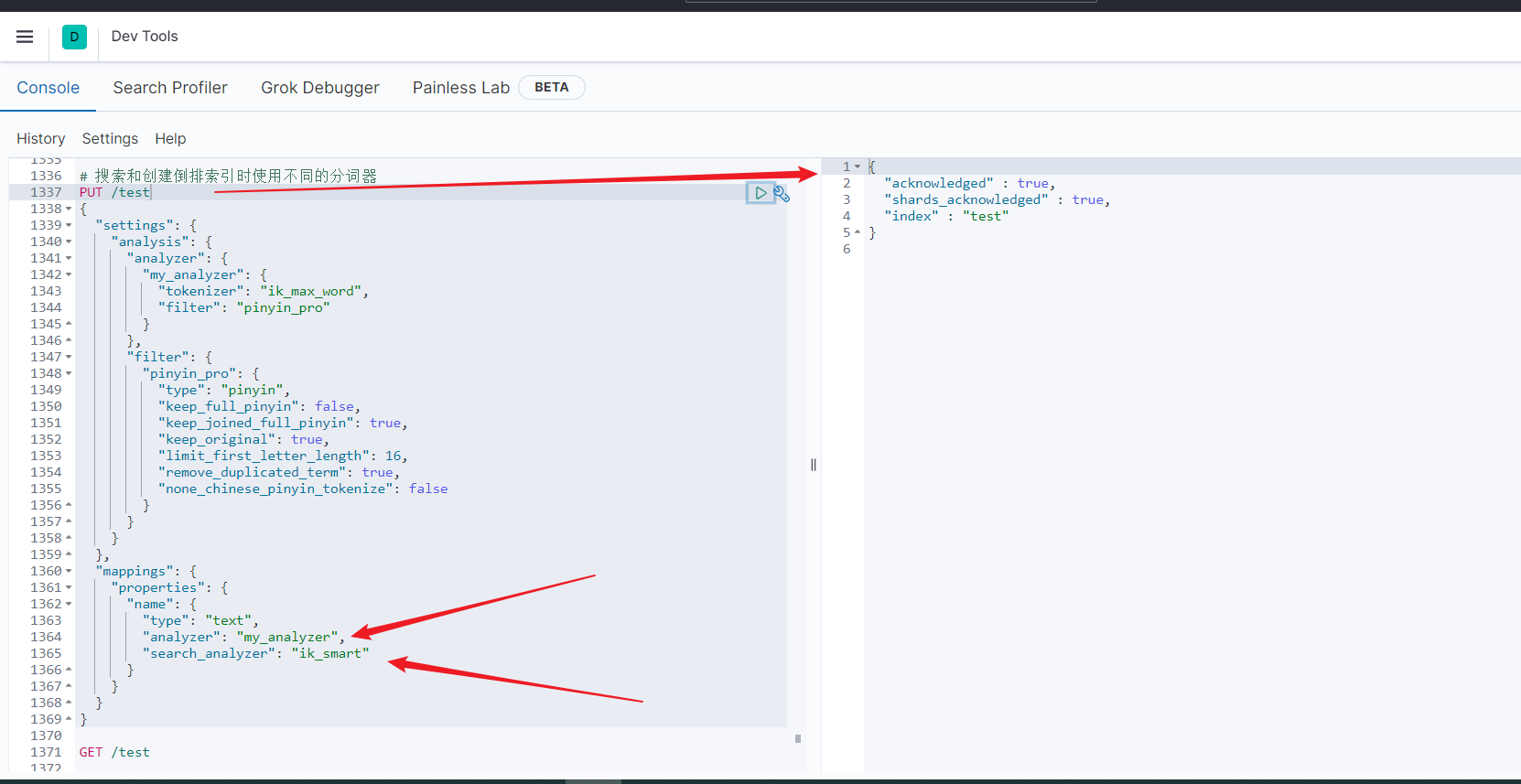
以上问题不是我们希望看到的，因此创建倒排索引的时候你用拼音分词器没问题，一点毛病都没有，但是你在搜索的时候不应该用拼音分词器，搜索的时候用户如果搜索的是中文，你就应该用中文去搜，用户输入的是拼音，你才拿拼音去搜。所以说我们要分开，创建时和搜索时用不同的分词器！那怎么样才能分开呢？请看语法：

就是在定义索引库mapping映射时，定义字段的分词器的时候指定两个，一个时analyzer，一个时search\_analyzer，analyzer是创建索引时候用的，search\_analyzer是在搜索时候用的！

测试：

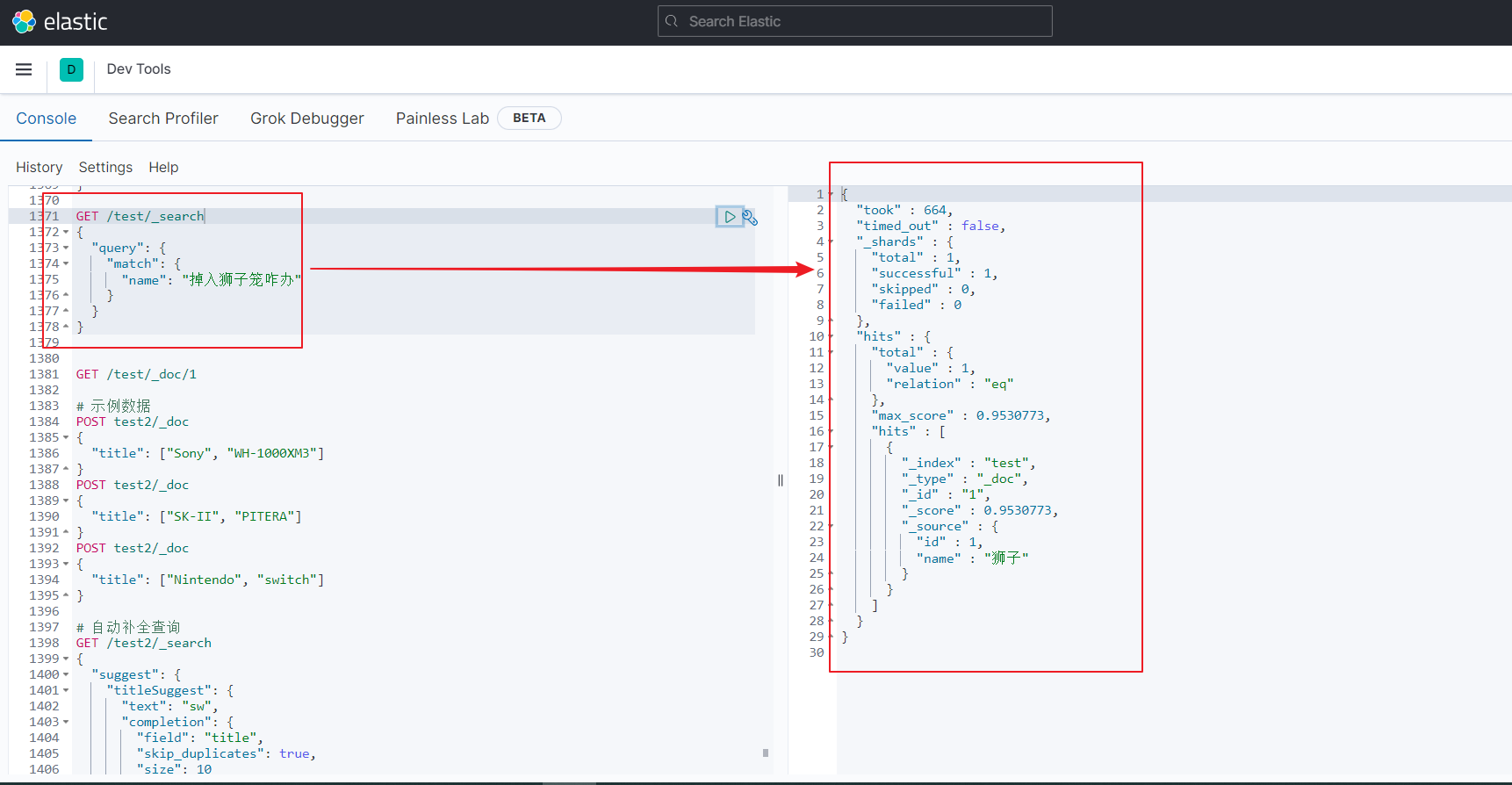
### 1：先删除test索引库： DELETE /test

### 2：修改DSL语句，创建test索引库，在mapping定义字段的分词器的时候指定两个分词器，一个是analyzer，一个是search\_analyzer



### 3：同样向索引库test插入“虱子”和“狮子”文档

### 4：搜索“掉进狮子笼咋办”，可以发现只搜索出“狮子”文档了，问题解决



## 总结：

### 如何使用拼音分词器？

#### 1：下载pinyin分词器

#### 2：解压并放到elasticsearch的plugin目录

#### 3：重启es即可

### 如何自定义分词器？

#### 创建索引库时，在settings中配置，可以包含三部分

##### 1：character filter

##### 2：tokenizer

##### 3：filter

### 拼音分词器注意事项？

#### 为了避免搜索到同音字，搜索时不要使用拼音分词器