Springboot+es全文检索

<https://blog.csdn.net/weixin_37281289/article/details/101483434>

网盘下载地址

**查看所有的索引(索引就像是数据库一样) :**

postman方法get请求。<http://localhost:9200/_all>

kibana方法 get.       GET \_all即可查询(不加[http://localhost:9200/](http://localhost:9200/_all)是因为kibana默认以及连接了es)

**创建索引:**

**postman用PUT请求 [http://localhost:9200/test意思就是创建一个test索引](http://localhost:9200/test%E6%84%8F%E6%80%9D%E5%B0%B1%E6%98%AF%E5%88%9B%E5%BB%BA%E4%B8%80%E4%B8%AAtest%E7%B4%A2%E5%BC%95)**

**kibana方法**

**删除索引:**

**Postman   DELETE方法请求。   <http://localhost:9200/test>**

**kibana方法**

**为所有添加数据(相当于表):**

**Postman PUT。<http://localhost:9200/person/_doc/1>**

在raw中输入数据{

"first\_name" : "John",

"last\_name":"Smith",

"age":25,

"about":"I love go to rock climbing",

"interests":["sports","music"]

}

即可创建一条数据在\_doc中

**kibana方法**

**logstash设置mysql连接**

**用logstash对mysql的条件是需要有id和时间的字段**

**1.打开文件引入mysql连接的jar**

**2.到config文件夹中新建mysql.conf文件**

**内容如下**

**input{**

**jdbc{**

**#jdbc驱动包位置**

**jdbc\_driver\_library =>"/Users/zhangxianzeng/Downloads/elasticsearchquanbuwenjian/logstash-6.3.2/mysql-connector-java-8.0.16.jar"**

**#要使用的驱动包类，有过开发经验的都应该知道这个**

**jdbc\_driver\_class =>"com.mysql.jdbc.Driver"**

**#mysql数据库连接信息**

**jdbc\_connection\_string =>"jdbc:mysql://127.0.0.1:3306/blog"**

**#mysql用户名**

**jdbc\_user =>"root"**

**#mysql密码**

**jdbc\_password =>"zhangxian11"**

**#定时任务，多久执行一次查询，默认是一分钟，如何想要没有延迟，可以使用schedule =>"\* \* \* \* \*"**

**schedule =>"\* \* \* \* \*"**

**#清空上一次的sql\_last\_value记录**

**clean\_run =>true**

**#你想要执行的语句**

**statement =>"SELECT \* FROM t\_blog WHERE update\_time > :sql\_last\_value AND update\_time < NOW() ORDER BY update\_time desc"**

**}**

**}**

**output{**

**elasticsearch{**

**#es host:port**

**hosts=>["127.0.0.1:9200"]**

**#索引**

**index=>"blog"**

**#\_id**

**document\_id=>"%{id}"**

**}**

**}**

启动命令bin目录下 **logstash -f ../config/mysql.conf**

**查看是否有数据**

**GET /blog/\_stats**

**查看是否有数据**

POST /blog/\_search

{

}

停止重启会出现问题

**查询数据:**

**postman get方法   [http://localhost:9200/person/\_doc/1根据id查询](http://localhost:9200/person/_doc/1%E6%A0%B9%E6%8D%AEid%E6%9F%A5%E8%AF%A2)**

**postman get方法 根据其他数据查询。[http://localhost:9200/person/\_doc/\_search?q=first\_name:zhang1根据姓名查询](http://localhost:9200/person/_doc/_search?q=first_name:zhang1%E6%A0%B9%E6%8D%AE%E5%A7%93%E5%90%8D%E6%9F%A5%E8%AF%A2)**

[http://localhost:9200/person/\_doc/\_search?q=age:25根据年龄查询](http://localhost:9200/person/_doc/_search?q=age:25%E6%A0%B9%E6%8D%AE%E5%B9%B4%E9%BE%84%E6%9F%A5%E8%AF%A2)

**kibana方法**

GET /person/\_doc/1根据id

GET /person/\_doc/\_search?q=age:25根据age

**POST /person/\_search**

**{**

**"query":{**

**"bool": {**

**"should": [**

**{**

**"match": {**

**"last\_name": "Smith"**

**}**

**}**

**]**

**}**

**}**

**}根据姓名等用post就需要这么写**

**第二种多个条件用should时代表的是or的意思**

**POST /person/\_search**

**{**

**"query":{**

**"bool": {**

**"should": [**

**{**

**"match": {**

**"last\_name": "Smith"**

**}**

**}**

**,**

**{**

**"match": {**

**"about": "rock"**

**}**

**}**

**]**

**}**

**}**

**}**

**第三种多个条件下用must代表and**

**POST /person/\_search**

**{**

**"query":{**

**"bool": {**

**"must": [**

**{**

**"match": {**

**"last\_name": "Smith"**

**}**

**}**

**,**

**{**

**"match": {**

**"about": "rock"**

**}**

**}**

**]**

**}**

**}**

**}**

**为所有修改数据(相当于表) ：**

**PostmanPUT。<http://localhost:9200/person/_doc/1>**

在raw中输入数据{

"first\_name" : "John1”,

"last\_name":"Smith",

"age":25,

"about":"I love go to rock climbing",

"interests":["sports","music"]

}

修改时会判断如果没有此数据那么就是新增操作，如果有此数据那么就是修改操作

**kibana方法**

**分词器**

**es内置分词器**

**standard分词器**

POST \_analyze

{

  "analyzer": "standard",

  "text": "我是中国人"

}

英文可以但是中文分词效果不佳

**IK分词器**

下载地址 <https://github.com/medcl/elasticsearch-analysis-ik>

[https://github.com/medcl/elasticsearch-analysis-ik/releases找到对应的版本和es版本一致](https://github.com/medcl/elasticsearch-analysis-ik/releases%E6%89%BE%E5%88%B0%E5%AF%B9%E5%BA%94%E7%9A%84%E7%89%88%E6%9C%AC%E5%92%8Ces%E7%89%88%E6%9C%AC%E4%B8%80%E8%87%B4)

下载完成后需要去到es安装目录中的plugins目录中新建一个文件夹，将下载后的ik包解压后把里面的所有东西都复制到新建的文件夹中即可，之后需要重启es

第一种方式

POST \_analyze

{

  "analyzer": "ik\_smart",

  "text": "我是中国人"

}

第二种将词分为更多的部分

POST \_analyze

{

  "analyzer": "ik\_max\_word",

  "text": "我是中国人"

}

ik/config/main.dic中包括很多的词

如果一个新的词可以在此文件中进行添加，比如我是慕课网，那么就在文件最下面写

我 我是 慕课网 等分开表示

**连接springboot需要注意springboot版本，6.3.2的es在2.2.0的springboot上可以运行高版本springboot会报错**

需要修改es的

yml文件

将注释打开my-application

然后重启es

springboot结合

Application.properties文件中的配置

*#es配置***spring.data.elasticsearch.cluster-nodes**=**127.0.0.1:9300  
  
spring.data.elasticsearch.cluster-name**=**my-application**

pojo对象

**package** com.example.springbootelasticsearch.espojo;  
  
**import** lombok.Data;  
**import** org.springframework.data.annotation.Id;  
**import** org.springframework.data.elasticsearch.annotations.DateFormat;  
**import** org.springframework.data.elasticsearch.annotations.Document;  
**import** org.springframework.data.elasticsearch.annotations.Field;  
**import** org.springframework.data.elasticsearch.annotations.FieldType;  
  
**import** javax.persistence.Column;  
**import** java.util.Date;  
@Data  
*//相当于表*@Document(indexName = **"blog"**,type = **"doc"**,useServerConfiguration = **true**,createIndex = **false**)  
**public class** ESBlog {  
 @Id  
 **private** Integer **id**;  
 *//添加需要分词的字段用 @Field(type = FieldType.Text,analyzer = "ik\_max\_word")* @Field(type = FieldType.***Text***,analyzer = **"ik\_max\_word"**)  
 **private** String **title**;  
 @Field(type = FieldType.***Text***,analyzer = **"ik\_max\_word"**)  
 **private** String **author**;  
 @Field(type = FieldType.***Text***,analyzer = **"ik\_max\_word"**)  
 **private** String **content**;  
 @Field(type = FieldType.***Date***,format = DateFormat.***custom***,pattern = **"yyyy-MM-dd HH:mm:ss||yyyy-MM-dd||epoch\_millis"**)  
 **private** Date **createTime**;  
 @Field(type = FieldType.***Date***,format = DateFormat.***custom***,pattern = **"yyyy-MM-dd HH:mm:ss||yyyy-MM-dd||epoch\_millis"**)  
 **private** Date **updateTime**;  
}

DAO类继承ElasticsearchRepository用来连接es使用es提供的dao接口

**package** com.example.springbootelasticsearch.esdao;  
  
**import** com.example.springbootelasticsearch.espojo.ESBlog;  
**import** org.springframework.data.elasticsearch.repository.ElasticsearchRepository;  
  
**public interface** EsBlogDao **extends** ElasticsearchRepository<ESBlog,Integer> {  
}

controller中类似于写sql语句

BoolQueryBuilder builder= QueryBuilders.*boolQuery*();

//should是或者的关系  
builder.should(QueryBuilders.*matchPhraseQuery*(**"title"**,blogvo.getKeyword()));  
builder.should(QueryBuilders.*matchPhraseQuery*(**"content"**,blogvo.getKeyword()));

//迭代器可以用Page<ESBlog>来表示  
Page<ESBlog> search= (Page<ESBlog>) **esBlogDao**.search(builder);

//得到满足条件的数据content   
List<ESBlog> content = search.getContent();  
map.put(**"list"**,content);