Search谷粒搜索

# 1 es相关介绍

|  |
| --- |
| 1 搜索引擎  elasticSearch6（和elasticSearch5的区别在于，root用户权限、一个库能否建立多个表）  2 搜索引擎    文本搜索(以空间换时间算法)  于同类产品相比(solr、hermes),和solr一样都是基于lucene(apache)，默认以集群方式工作  搜索引擎(以百度和goole为例)的工作原理是什么？  a 爬虫  b 分析  c 查询  3 elasticSearch(搜索引擎)的算法  倒排索引(在内容上建立索引，用内容去匹配索引)  Btree（balance tree b-tree）  B+tree |

# 2 安装和启动

|  |  |
| --- | --- |
| 1 安装环境  Centos7 + jdk8  2 配置文件  elasticSearch.yml（集群配置文件）、jvm.Opitons(jvm配置文件)    3 创建目录、上传，解压  Mkdir -p /opt/es  上传  Tar – zxvf elasticsearch-6.3.1.tar.gz  4 配置  Es使用最大线程数、最大内存数、访问的最大文件数  启动脚本      需要改成其他非root用户才能启动  Adduser es  Su es  Es使用的jvm的内存大小    访问权限：es无法访问data目录(data是es的软件和日志数据目录)    5 启动  Es的权限问题：  首先用root用户解压  Tar – zxvf elasticsearch-6.3.1.tar.gz  然后用root用户授权  Chmod 777 -R elasticsearch-6.3.1  启动后配置  elasticSearch.yml、jvm.Opitons  jvm.Opitons中配置es能够使用jvm内存大小    elasticSearch.yml中配置es的host地址(配成本机地址，允许访问)    配置完毕启动es(必须切换到非root用户下)    启动后会报错(linux的默认线程数、最大文件数、最大内存数都不够)    6 修改linux的配置(配合es的启动需求)  两处修改  A修改linux的limits配置文件，设置内存线程和文件    B修改linux的sysctl配置文件，配置系统使用内存    整个es的配置有四处文件需要修改   |  | | --- | | elasticSearch.yml es的启动host地址  jvm.options配置es的虚拟机内存  limits.conf配置linux的线程内存和文件  sysctl.conf配置系统允许的软件运行内存 |   7 启动安装 |

# 3 es的基本概念

|  |
| --- |
| 1 通过(9200端口)http协议进行交互    http://192.168.222.20:9200/\_cat/indices?v  2 基本概念  Index 库  Type 表  Document 行(一条数据)  Field 字段  3 开发工具(kibana)  配置      启动kibana    查看kibana    端口号5601 |

# 4 es简单办的增删改查

|  |  |
| --- | --- |
| PUT  DELETE  POST  GET    查询带“红”的所有内容   |  | | --- | | GET movie\_index/movie/\_search  {  "query": {  "match": {  "name": "red"  }  }  } | |

# 5 对于中文的分词

|  |
| --- |
| GET \_analyze  {  "text": "红海行动"  }    加入中文分词器  进入插件目录    在plugins目录下解压一层目录就可以  Ik(中英文分词器)有两个：  1 ik\_smart（简易分词）  2 ik\_max\_word（尽最大可能分词）  GET \_analyze  {  "analyzer": "ik\_smart",  "text": "我是中国人"  } |

|  |
| --- |
| 相关性算分数：  查询的关键字在进行分词处理后，与es的文档（分词处理并建立了索引）之间的相关度的高低 |

# 6 es集群

|  |
| --- |
| 1 克隆和一台es的服务器        2 启动克隆的机器，修改ip      3 重启网络  Service network restart |

# 7 节点、分片、索引

|  |
| --- |
| 1 节点  一个节点就是一个es的服务器，es集群中，主节点负责集群的管理和任务的分发，一般不负责文档的增删改查  2 片  分片是es的实际物理存储单元(一个lucene的实例)  3 索引  索引是es的逻辑单元，一个索引一般建立在多个不同机器的分片上  4 复制片  每个机器的分片一般在其他机器上会有两到三个复制片(目的是提高数据的容错率)  5 容错  一旦集群中的某些机器发生故障，那么剩余的机器会在主机点的管理下，重新分配资源(分片)  6 分片的路由  写操作(新建、删除)只在主分片上进行，然后将结果同步给复制分片  Sync 主分片同步给复制成功后，才返回结果给客户端  Async 主分片在操作成功后，在同步复制分片的同时返回成功结果给客户端  读操作(查询)可以在主分片或者复制分片上进行 |

# 8 谷粒搜索模块

步骤

|  |
| --- |
| 1 数据结构的准备  2 项目的初始化  3 es客户端的整合  4 谷粒搜索代码的开发  A 搜索功能  B 面包屑功能 |

数据结构

|  |
| --- |
| 1 商品名称(展示/查询)  2 商品价格(展示/查询)  3 商品图片(展示)  4 平台属性和属性值的列表(查询)  5 商品描述(展示/查询)  6 热度值(查询)  7 三级分类id(查询)  8 商品id  9 主键 |

参数结构

|  |
| --- |
| 关键字(商品名称(展示/查询) 5 商品描述(展示/查询) 2 商品价格(展示/查询))  平台属性和属性值的列表(查询)  三级分类id(查询) |

# 9 通过es的mapping定义商品的数据结构

|  |
| --- |
| Es的mapping定义是基于整个库的  Mysql数据结构字段定义是基于整合表  Es的默认数据类型 |

|  |
| --- |
| PUT gmall0105  {  "mappings": {  "PmsSkuInfo":{  "properties": {  "id":{  "type": "keyword",  "index": true  },  "skuName":{  "type": "text",  "analyzer": "ik\_max\_word"  },  "skuDesc":{  "type": "text"  , "analyzer": "ik\_smart"  },  "catalog3Id":{  "type": "keyword"  },  "price":{  "type": "double"  },  "skuDefaultImg":{  "type": "keyword",  "index": false  },  "hotScore":{  "type": "double"  },  "productId":{  "type": "keyword"  },  "skuAttrValueList":{  "properties": {  "attrId":{  "type":"keyword"  },  "valueId":{  "type":"keyword"  }  }  }  }  }  }  } |

Jest：

Es

|  |
| --- |
|  |

# 10 如何定义复杂查询

|  |
| --- |
| 查询一下条件的手机：  1 查询所有名字中带华为  2 查询所有4寸以下，16G内存  Query{  Bool:{// 先过滤，后查询  Filter:{term,term}  must:{match}  }  }    在一个过滤条件中加入并集    查询api  Search search = **new** Search.Builder(**"dsl的json语句"**).addIndex(**"索引名"**).addType(**"表名"**).build();  查询dsl的封装工具类    Term条件    Match条件 |

# 11 前端静态资源

|  |
| --- |
| 1 三级分类静态资源文件的加载路径    修改路径    2 修改list.gamll.com修改search.gmall.com:8083    3 三级分类的加载文件的js，请求url改为search.gmall.com:8083 |

# 12 业务实现

|  |  |
| --- | --- |
| 1 controller参数传递  增加一个PmsSearchParam参数类  2 service的业务逻辑编写  之前在测试是已完成  3排序和高亮  排序  GET movie\_index/movie/\_search  {  "query":{  "match": {"name":"red sea"}  }  , "sort": [  {  "doubanScore": {  "order": "desc"  }  }  ]  }   |  | | --- | | *// sort* searchSourceBuilder.sort(**"id"**,SortOrder.***DESC***); |   高亮  "highlight" : {  "fields" : {  "skuName" : { }  }  } |