# 0.简介

logstash是基于nodejs的前端项目，提供了访问elasticsearch服务的可视化界面。

logstash体系结构:源数据源>输入编解码插件>事件队列>过滤器插件>输出编解码插件>目标数据源

# 1.安装

## 1.1.解压方式

### 1.1.1. 解压

tar -xvf kibana-7.7.0-linux-x86\_64.tar.gzip

# 2. 配置

vi /config/kibana.yml

server.host: "${外网ip}" #如"172.18.194.140"

elasticsearch.hosts: ["http://172.18.192.140:9200"] #配置elasticsearch类的请求转发到哪个elasticsearch服务器

# 3.运行

## 3.1. 执行运行命令

./bin/kibana

# 4.浏览器访问

enjoy-elk.com:5601 #enjoy-clk.com域名与network.host配置的ip对应

# 5.界面功能

## 5.1. Dev Tools

enjoy-elk.com:5601/app/kibana#/dev\_tools/console

#创建索引 enjoy\_test为索引名称

put enjoy\_test

#查看索引 enjoy\_test为索引名称

get enjoy\_test

#创建文档 enjoy\_test为索引名称,\_doc数据类型,1数据名称

put enjoy\_test/\_doc/1

{

"msg":"hello,world",

"name":"Apache Hadoop",

"lang":"Java",

"corp":"Apache",

"stars":200,

/\*properties.${field}.type设置字段的类型，不显示设置则根据传入的属性值智能自动设置\*/

"properties":{

"name":{"type":"text"},

"lang":{"type":"text"},

"corp":{"type":"text"},

/\*fields定义子字段\*/

"fields":{

"raw":{

"type":"keyword"

},

length:{

"type":"token\_count",

/\*analyzer设置分词器类型\*/

"analyzer":"standard"

}

}

}

}

#查看文档，enjoy\_test为索引名称,\_doc数据类型,1数据名称

get enjoy\_test/\_doc/1

#查看索引的映射信息，enjoy\_test为索引名称

get enjoy\_test/\_mapping

# ，enjoy\_test为索引名称

get enjoy\_test/\_close

# ，enjoy\_test为索引名称

get enjoy\_test/\_open

#查看索引的配置信息，enjoy\_test为索引名称，不指明特定索引则查询全部\_setting，查询某几个特定索引可用“,”逗号分隔

get enjoy\_test/\_setting

#修改索引的配置信息，enjoy\_test为索引名称

put enjoy\_test/\_setting

{

"refresh\_interval":"2s"

}

#\_update接口，保存id为1的文档数据，doc\_as\_upsert为true表示id数据记录存在则更新，不存在则新增

post enjoy\_test/\_update/1

{

"doc":{

"star":500

},

"doc\_as\_upsert":true

}

#\_search搜索接口,from从第几条开始读数据,size每页条数,includes包含哪种字段,excludes去除哪种字段

#sort排序

#term精确查找，terms

get /kibana\_sample\_data\_flights/\_search

{

from:100,

size:20,

query:{

term:{

DestCountry:"CN"

}

},

\_source:{

includes:["\*.lon","\*.lat","DistanceKilometers","FlightNum"],

excludes:"Destlocation.\*"

},

sort:[{"DistanceKilometers":"asc"},{"FlightNum":"desc"}]

}

#可以查询到的数据行最大值，默认值10000，(from+size)>index.max\_result\_window

index.max\_result\_window

#基于词项搜索

term terms range exists prifix wildcard

#全文检索

match multi\_match match\_phrase match\_phrase\_prifix 模糊查询

#类似联查

query:{

terms:{

\_id:{

index:"user",/\*索引user中id为1的articles字段\*/

id:1,

path:"articles"

}

}

}