03

Quebrando os textos com Analyzers

Full-text search e analyzers

Analyzers são algoritmos fundamentais para o que chamamos de *full-text search*, ou busca de texto cheio. Vale destacar um conceito fundamental em relação a busca de texto cheio. Quando fazemos buscas exatas o resultado é binário, ou seja, "sim" caso o termo procurado exista da maneira que foi informado ou "não", caso contrário. No caso de busca de texto cheio, a ideia é diferente. Ao invés de perguntarmos "Este documento possui exatamente os termos utilizados na busca", estamos interessados em "O quão bem este documento casa com os termos da busca".

Vamos ver alguns exemplos. No primeiro exemplo, queremos procurar pelo termo "EUA" mas estamos satisfeitos com documentos com os termos "Estados Unidos da América" ou "Estados Unidos". No segundo exemplo, queremos procurar pelo termo "musica", porém estamos satisfeitos se encontrarmos documentos com os termos "bossa nova", "rock", "pagode" e "samba", pois eles se relacionam com música.

Analyzers são utilizados para processar nossos documentos e construir uma estrutura comum no mundo de buscas chamada de índice invertido. De maneira simplista, podemos pensar nos analyzers como algoritmos que processam o texto e geram entradas relevantes com os termos do documento, possivelmente com sinônimos, no índice invertido. Estas entradas possuem ponteiros para o documento. Os termos consultados também passam pelos mesmos algoritmos e os termos processados são utilizados para fazer a busca contra o índice invertido. Este processo ficará bem claro em alguns instantes.

Analyzers existentes

ElasticSearch possui diversos analyzers pré-definidos que podem ser associados à atributos durante a criação do mapping para o tipo. Destacamos 4 analyzers que devemos conhecer.

- **Analyzer padrão**: Este é o analyzer padrão usado pelo ElasticSearch e em geral funciona bem independente do idioma. Ele funciona quebrando o texto em palavras removendo pontuações e passando todo conteúdo para letras minúsculas. Números existentes no texto são mantidos. Por exemplo: "Eu nasci há 10 mil (sim, 10 mil) anos atrás" gera as seguintes entradas "eu", "nasci", "há", "10", "mil", "sim", "10", "mil", "anos", "atrás".
- **Analyzer simples**: Quebra o texto em tudo o que não seja uma letra e passando todo o texto para letras minúsculas. Como números não são letras, eles não geram entradas. E.g.: "Eu nasci há 10 mil (sim, 10 mil) anos atrás" gera as seguintes entradas "eu", "nasci", "há", "mil", "sim", "mil", "anos", "atrás".
- **Analyzer de espaço em branco**: Quebra o texto por espaços em branco. Não há alteração na caixa das letras. Por exemplo: "Eu nasci há 10 mil (sim, 10 mil) anos atrás" gera as seguintes entradas "Eu", "nasci", "há", "10", "mil", "(sim", "10", "mil)", "anos", "atrás".
- **Analyzers específicos para idiomas**: São analyzers que quebram o texto assim como o analyzer padrão, porém são capazes de aplicar peculiaridades do idioma e melhorar a geração das entradas para um idioma em específico. Técnicas como singularização dos termos, remoção de palavras que não possuem relevância para o resultado, como palavras comuns do idioma e uso da palavra na sua forma mais raíz (conhecido como *stemming*), são aplicadas.

Como passo adicional, analyzers ainda podem ser customizados caso necessário. Para mais detalhes veja https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-analyzers.html (https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-analyzers.html)

O mais legal é que existe uma API chamada `_analyze` que nos permite observar como um texto será analisado.

Utilizando a API _analyze

Vamos comparar os analyzers *standard, whitespace, simple* e *portuguese* para a frase "Eu nasci há 10 mil (sim, 10 mil) anos atrás".

Nossa requisição será:

```
GET /_analyze?analyzer=standard&text=Eu+nasci+a+10+mil+(sim,+10+mil)+anos+atrás
```

E o resultado:


```
{
  "tokens": [
    {
      "token": "eu",
      "start_offset": 0,
      "end_offset": 2,
      "type": "<ALPHANUM>",
      "position": 0
    },
    {
      "token": "nasci",
      "start_offset": 3,
      "end_offset": 8,
      "type": "<ALPHANUM>",
      "position": 1
    },
    {
      "token": "há",
      "start_offset": 9,
      "end_offset": 10,
      "type": "<ALPHANUM>",
      "position": 2
    },
    {
      "token": "10",
      "start_offset": 11,
      "end_offset": 13,
      "type": "<NUM>",
      "position": 3
    },
    {
      "token": "mil",
      "start_offset": 14,
      "end_offset": 17,
      "type": "<ALPHANUM>",
      "position": 4
    },
    {
      "token": "sim",
      "start_offset": 19,
      "end_offset": 22,
```

```
      "type": "<ALPHANUM>",
      "position": 5
    },
    {
      "token": "10",
      "start_offset": 24,
      "end_offset": 26,
      "type": "<NUM>",
      "position": 6
    },
    {
      "token": "mil",
      "start_offset": 27,
      "end_offset": 30,
      "type": "<ALPHANUM>",
      "position": 7
    },
    {
      "token": "anos",
      "start_offset": 32,
      "end_offset": 36,
      "type": "<ALPHANUM>",
      "position": 8
    },
    {
      "token": "atrás",
      "start_offset": 37,
      "end_offset": 42,
      "type": "<ALPHANUM>",
      "position": 9
    }
  ]
}
```


Note cada *token* e seu tipo.

Vejamos agora a diferença dos analyzers *standard* e *portuguese* para o texto "Música". Para o analyzer *standard*, utilizamos a seguinte requisição:

```
GET /_analyze?analyzer=standard&text=Música
```

E o resultado obtido é:


```
{
  "tokens": [
    {
      "token": "música",
      "start_offset": 0,
      "end_offset": 6,
      "type": "<ALPHANUM>",
      "position": 0
    }
  ]
}
```

Já para o analyzer *portuguese*, utilizamos a seguinte requisição:

```
GET /_analyze?analyzer=portuguese&text=Música
```

E obtemos o seguinte resultado:

```
{
  "tokens": [
    {
      "token": "music",
      "start_offset": 0,
      "end_offset": 6,
      "type": "<ALPHANUM>",
      "position": 0
    }
  ]
}
```

Repare na diferença dos *tokens* gerados. Enquanto o analyzer padrão gerou o token *música*, o analyzer para lingua portuguesa gerou o token *music*. Como até o momento usamos apenas o analyzer padrão, fica claro o motivo de nossas buscas não terem funcionado da forma flexível como foi prometido. Quando utilizando o termo *musica* anteriormente, este termo, mesmo após o processamento pelo analyzer, continua sendo musica e não música, como a entrada criada no índice invertido. Vamos corrigir essa inconveniência.

Recriando nosso índice com analyzers

Agora que entendemos um pouco melhor como o ElasticSearch processa nossos documentos e os prepara para busca, podemos recriar nosso índice de exemplo e resolver os problemas de busca que tivemos até então, como por exemplo, a busca pelo termo musica encontrar os documentos com música. Basta criamos um índice com o seguinte *mapping*. Note que estão definindo o analyzer *portuguese* também para o campo *_all*. Tecnicamente falando, esta alteração já seria suficiente para resolver a inconveniência que estamos enfrentando. Contudo, estamos alterando os analyzers de alguns atributos para estender o suporte a buscas em atributos específicos.

```
PUT /catalogo_v2

{
  "settings": {
    "index": {
      "number_of_shards": 3,
      "number_of_replicas": 0
    }
  },
  "mappings": {
    "pessoas_v2": {
      "_all": {
        "type": "string",
        "index": "analyzed",
        "analyzer": "portuguese"
      },
      "properties": {
        "cidade": {
```

```
            "type": "string",
            "index": "analyzed",
            "analyzer": "portuguese"
          },
          "estado": {
            "type": "string"
          },
          "formação": {
            "type": "string",
            "index": "analyzed",
            "analyzer": "portuguese"
          },
          "interesses": {
            "type": "string",
            "index": "analyzed",
            "analyzer": "portuguese"
          },
          "nome": {
            "type": "string",
            "index": "analyzed",
            "analyzer": "portuguese"
          },
          "país": {
            "type": "string",
            "index": "analyzed",
            "analyzer": "portuguese"
          }
        }
      }
    }
}
```

Note que utilizamos o sufixo **_v2** tanto no nome do índice quanto no nome do tipo, porém o objetivo é apenas evitar alterações no primeiro índice que criamos.

Índice invertidos

Caso queira entender melhor como a busca de termos funciona quando utilizamos um índice invertido, veja o link: https://www.elastic.co/guide/en/elasticsearch/guide/current/inverted-index.html (https://www.elastic.co/guide/en/elasticsearch/guide/current/inverted-index.html)

O que aprendemos?

- Como a busca exata e a busca de texto cheio são diferentes.
- O que são e como usar os analyzers para incrementar nossas buscas.
- Como verificar como um analyzer vai indexar um determinado texto.
- Como alterar o analyzer do campo _all do padrão para portuguese e melhorar nossas buscas.