



**Persistência e Pesquisa de Dados**

**Capítulo 6. Pesquisa Indexada em Bases Não Estruturadas**

**Prof. Gustavo Aguilar**



## **Aula 6.1. Introdução à pesquisa indexada**

- ☐ Introdução à pesquisa indexada.
- ☐ Ferramentas de mercado.

- **Pesquisar dados:**

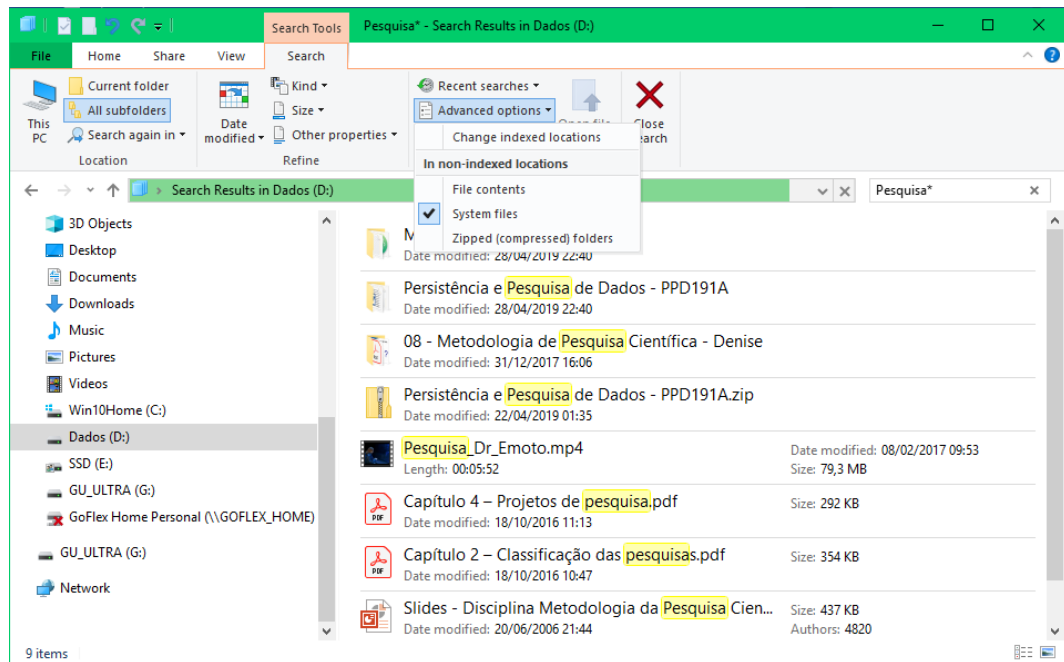
- Necessidade intrínseca quando se armazena dados.
- Pesquisas no sistema operacional, banco de dados, aplicação, página web, etc.

- **Aumento da velocidade e assertividade das pesquisas:**

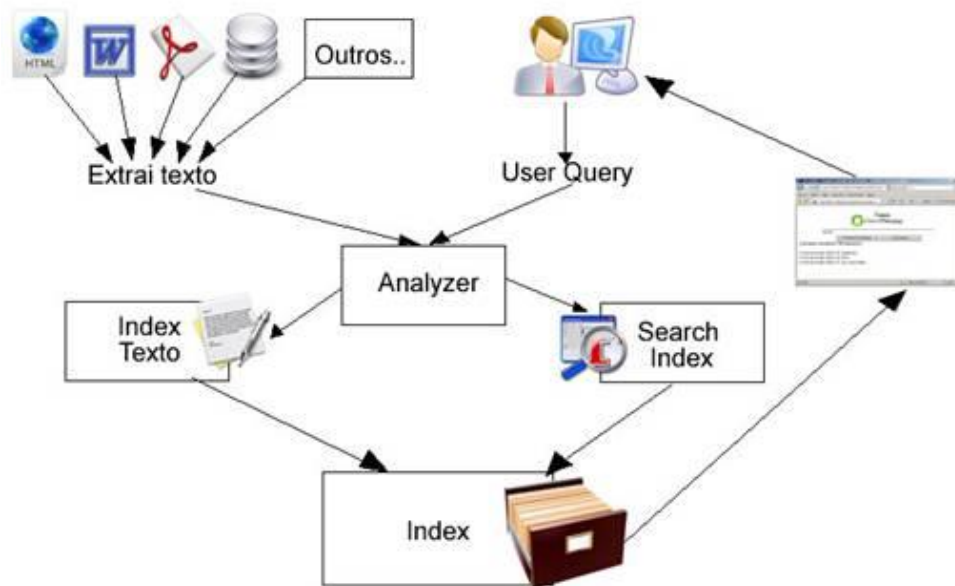
- Mecanismos de indexação no banco de dados → índices e full text search.
- Indexação na web → Google Search.
- Indexação para aplicações → Lucene.
- Indexação no sistema operacional → ex. Windows Desktop Search.

# Introdução à pesquisa indexada

- Windows Desktop Search:



- Indexação de texto:



- Boom da Era Digital.
  - **Projetos de Big Data / BI:** manipulação e pesquisa em volumes gigantescos de dados, mudança dinâmica de regras e análises realtime.
  - **E-commerce:** sugestões de produtos relacionados, recursos de autocompletar, personalização de produtos e promoções em tempo real de acordo com o perfil do cliente, etc.
  - **Buscadores de preços:** informar para um usuário quando algum produto atingir o preço configurado por ele.
  - **Tendências de mercado:** prever comportamentos e desejos dos usuários, antecipar procuras ou compras.

- Grande lacuna: ferramentas de pesquisa indexada mais robustas, ágeis, escaláveis e dinâmicas → “Pesquisa elástica”.
- Apache → **Solr**.
  - Abstração construída em cima do Lucene.
  - API HTTP trocando dados com XML/JSON, permitindo usar a busca e consumir o serviço pronto pela web.
  - Filtragem e pesquisa gerenciada (auxiliadores de busca como sugestão).
  - Pesquisa geoespacial e cache com atualização incremental.
  - Distribuição e replicação de dados.







- Elastic → com o **Elasticsearch**.
  - Concorrente direto do Solr, com os mesmos recursos e mais além.
  - Mais moderno, com mais facilidades e atualização dos índices próxima do tempo real.
  - Alta disponibilidade e capacidade para tratar grandes volumes de dados.
  - Pesquisas full-text, geográficas e analíticas.
  - Disponibiliza uma API REST full.
  - Possibilidade de implementação de clusters, sendo totalmente escalável.

- Microsoft → **Azure Search**.



- Serviço oferecido na nuvem (Microsoft Azure).
- Funcionalidade exposta por meio de uma API REST ou SDK do .NET.
- Integração nativa com o Azure Blob Storage (armazenamento).
- Pesquisa cognitiva, geográfica e análise linguística.
- Modelagem de relevância (pontuação) dos valores.



ElasticSearch Service



## ▪ Amazon Elasticsearch Service.

- Serviço gerenciado que facilita a implantação, a operação e o dimensionamento de clusters do Elasticsearch na nuvem (AWS).
- Recursos para dimensionar, controlar segurança, estabilidade e integração com outros serviços:
  - Amazon CloudWatch (para monitoramento).
  - AWS CloudTrail (para auditoria).
  - Amazon S3 (armazenamento).
  - DynamoDB (banco de dados).

- ☑ Pesquisa indexada de forma elástica como uma nova necessidade na era digital, para lidar de forma ágil com o crescente e gigantesco volume de dados.
- ☑ Não há uma grande pluralidade de ferramentas de mercado com essa proposta, mas as existentes atendem, cada uma com suas particularidade e usos mais indicados.

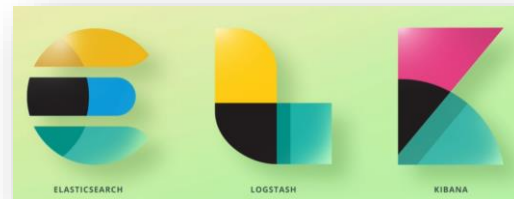
- ☐ Introdução ao Elastic Stack.



## **Aula 6.2. Introdução ao Elastic Stack**

- ☐ Elastic Stack e seus componentes.
- ☐ Cenários de uso.

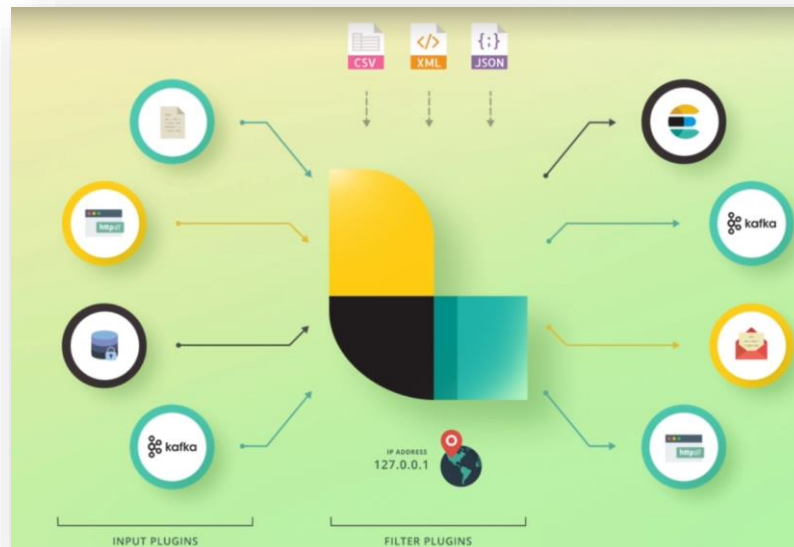
- O Elastic Stack é um grupo de produtos de código aberto da Elastic.
  - Ajudar os usuários a coletar dados de qualquer tipo de fonte e em qualquer formato.
  - Pesquisar, analisar e visualizar esses dados em tempo real ou não.
- Inicialmente conhecido como **ELK Stack**:
  - Letras representavam as iniciais do nome dos primeiros produtos do pacote: **Elasticsearch**, **Logstash** e **Kibana**.





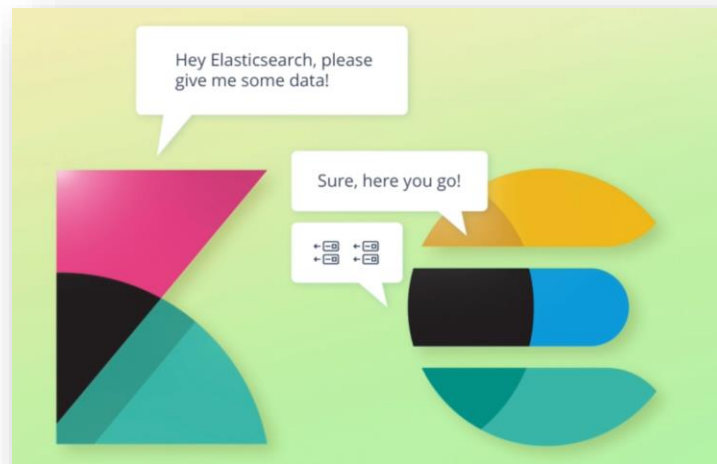
# Elastic Stack e seus componentes

- **Elasticsearch (E):** mecanismo de pesquisas e análises distribuído.
- **Logstash (L):** ferramenta de processamento de dados.
  - Coletar dados de várias fontes.
  - Transformar dados.
  - Enviar dados.
  - Filtros predefinidos.
  - Mais de 200 plugins.



## ▪ Kibana (K):

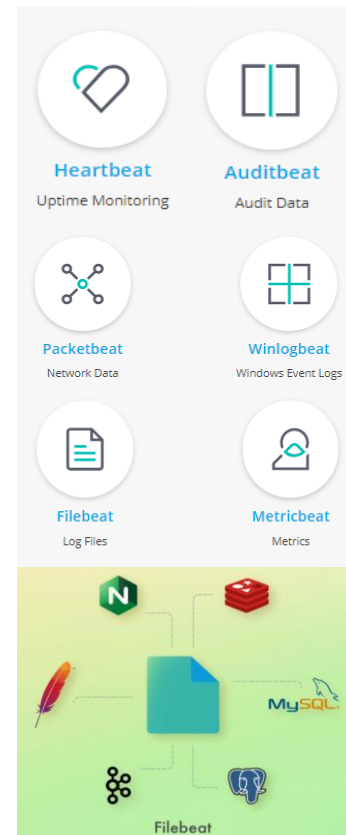
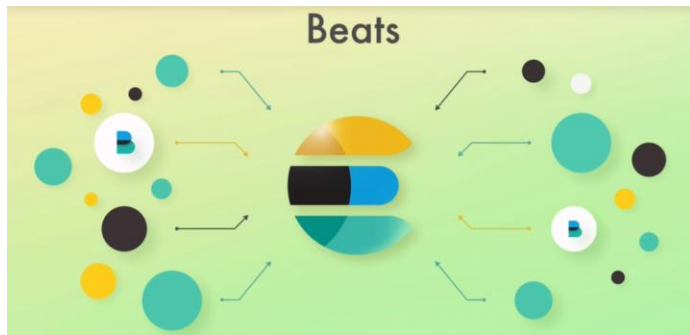
- Ferramenta de visualização e exploração de dados.
- Oferece gráficos interativos e fáceis de usar.
- Agregações e filtros pré-construídos.
- Suporte geoespacial.
- Alta integração com o Elasticsearch.



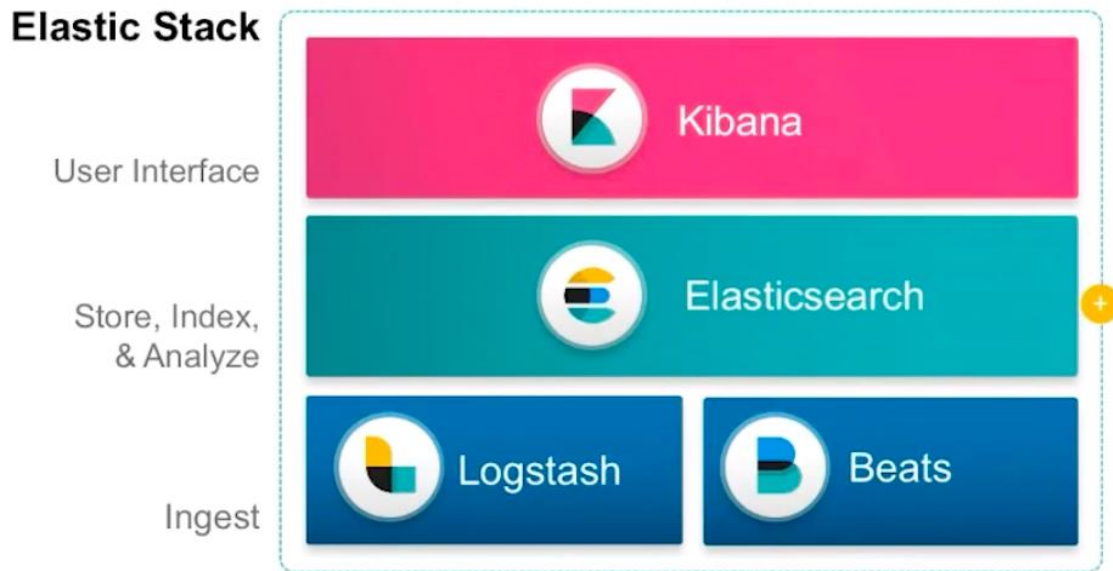
# Elastic Stack e seus componentes

## ■ Beats:

- Quarto produto adicionado posteriormente ao pacote.
- Plataforma com o propósito único de enviar dados.
- Envio de dados diretamente para o Elasticsearch...
- ... ou através do Logstash: tratar o dado antes.



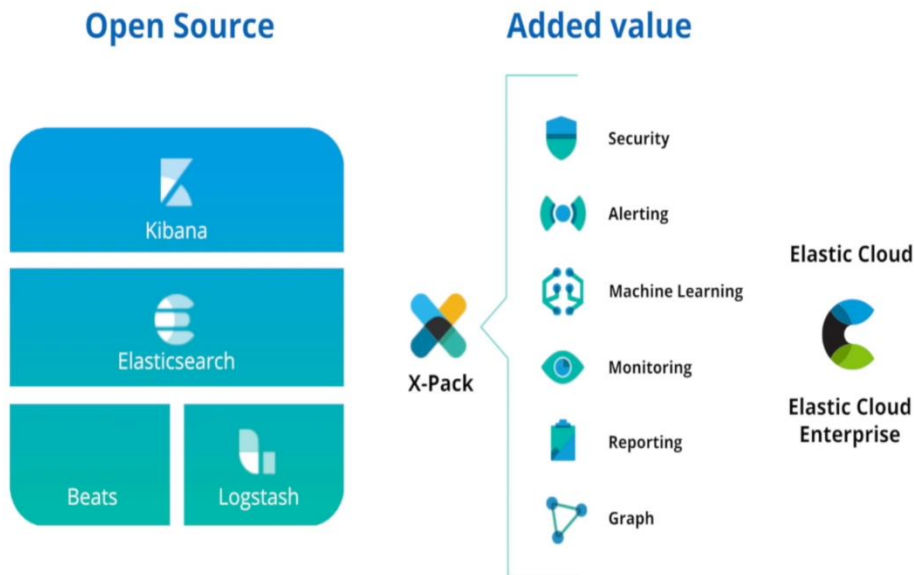
- Grupo passou a ser referenciado como **Elastic Stack**.



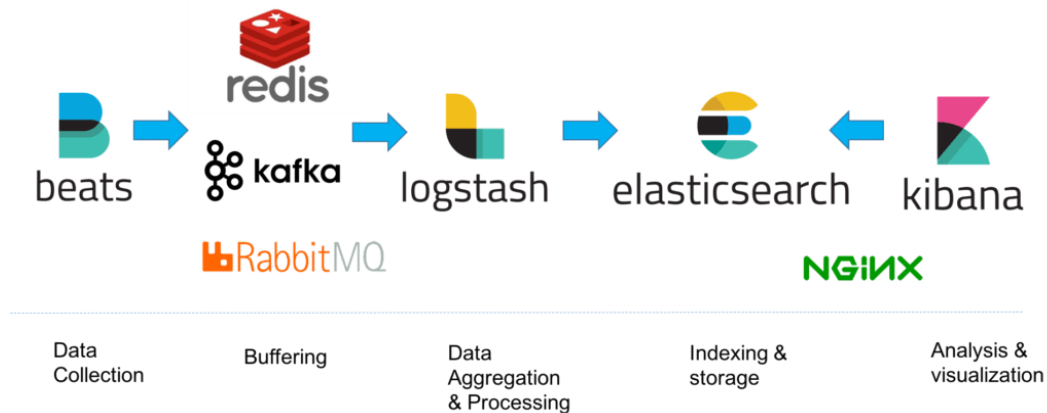
- Os produtos no Elastic Stack são projetados para serem usados juntos:
  - Releases são sincronizados para simplificar o processo de instalação e atualização.
  - Dessa forma, ao instalar o Elastic Stack é preciso usar a mesma versão em todos os componentes do pacote.
  - Há também a possibilidade da instalação isolada de somente um componente do pacote.

# Elastic Stack e seus componentes

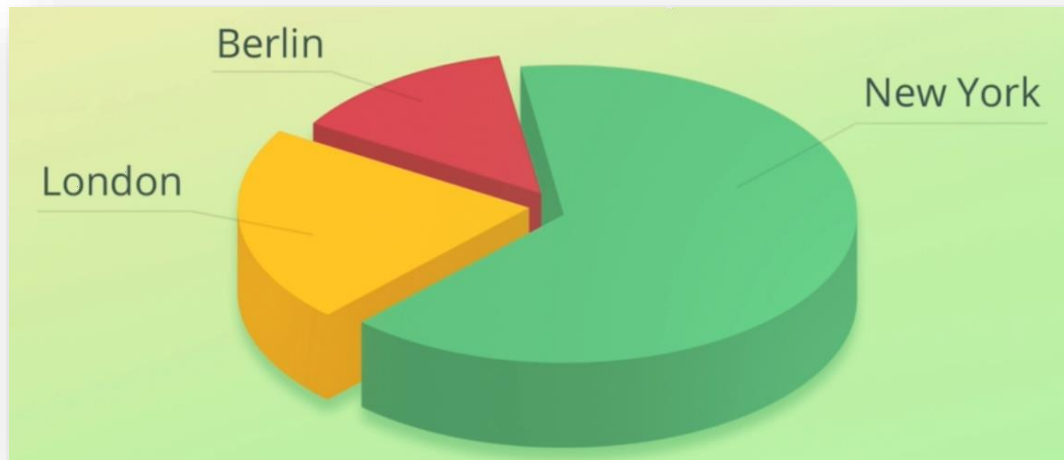
- **X-Pack:** conjunto de produtos não gratuitos, desenvolvidas pela Elastic, que prometem agregar valor ao pipeline de dados:



- As ferramentas do Elastic Stack podem ser utilizadas em conjunto com produtos de outros fornecedores.
  - Plataformas de armazenamento de dados em memória (Redis / Kafka / RabbitMQ) para serem utilizadas como buffer.

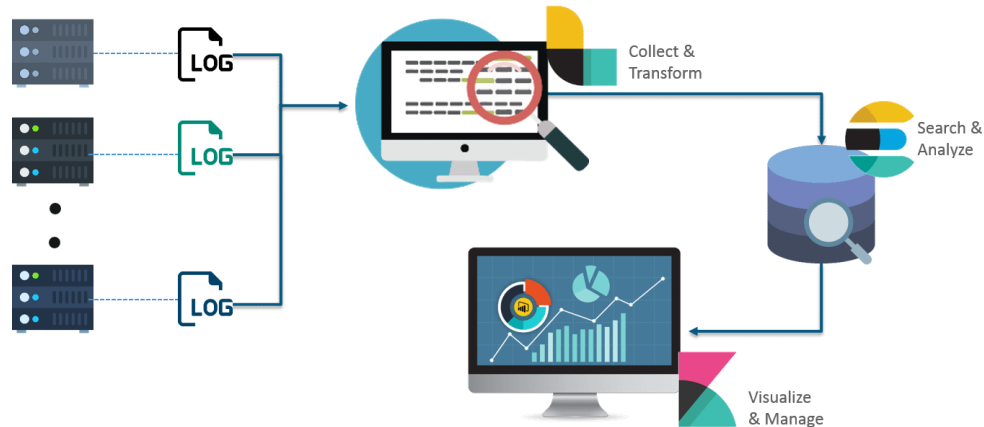


- Agregações com Elasticsearch.





- Solução de gerenciamento de performance de aplicação (*Application Performance Management – APM*).
  - Analisa logs de aplicações, banco de dados, sistema operacional ou servidor web → Fins de acompanhamento de métricas.



- ☑ Componentes iniciais do Elastic Stack eram o Elasticsearch, Logstash e o Kibana, sendo chamado inicialmente de ELK Stack.
- ☑ O Beats e outros foram sendo adicionados, passando-se a se chamar ElasticStack.
- ☑ Podem trabalhar de forma isolada ou integrada, inclusive com produtos de outros fornecedores.
- ☑ Cenários de uso vão desde dashboards em tempo real à monitoração de aplicações baseada em métricas (APM).

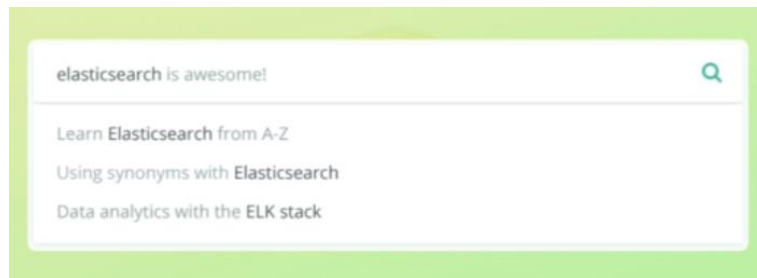
- ❑ Conceitos básicos e arquitetura do Elasticsearch.



## **Aula 6.3. Conceitos básicos e arquitetura do Elasticsearch**

- ☐ Conceitos básicos do Elasticsearch.
- ☐ Arquitetura do Elasticsearch.

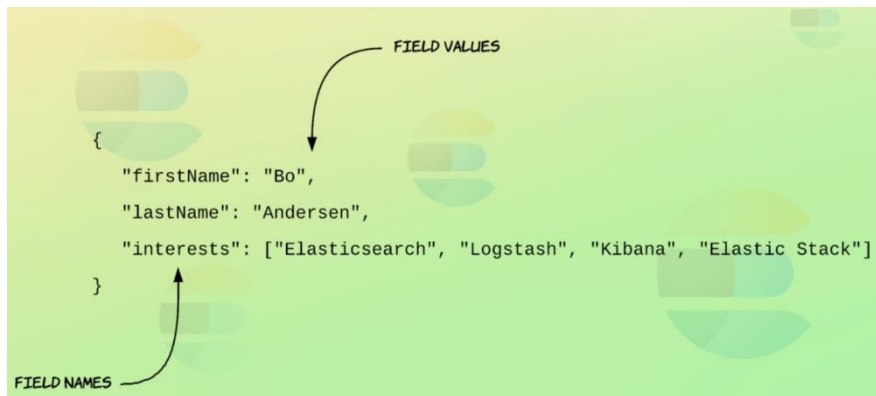
- Elasticsearch é uma ferramenta de pesquisa indexada opensource.
- Capacidade para tratar grandes volumes de dados em tempo real.



- Tem sido utilizada por grandes empresas como Facebook, Google, eBay, Mercado Livre, Netflix, Netshoes, GitHub, SoundCloud, Twitter, Uber e Verizon.

- Baseado no *Apache Lucene*.
- Desenvolvido por Shay Bannon, em Java, e lançado em 2010.
- Utilizando uma interface comum (JSON sobre HTTP), ele possui client para as principais linguagens de programação.
  - Java, Go, .NET, PHP, Perl, Python e Ruby.
  - Clients de contribuição da comunidade também.
  - <https://www.elastic.co/guide/en/elasticsearch/client/index.html>.
- Foi desenvolvido do zero com o objetivo de ser escalável, ou seja, podendo ser utilizado de forma distribuído em clusters.

- **Dados são armazenados como documentos**, no formato **JSON**.
- Documento corresponde ao conceito de linha (tupla) em SGBDR.
- Documento contém **campos (*fields*) e valores**:
  - Conceito correspondente ao de tabela em um banco de dados relacional.





- Cada documento possui seu ID:
  - Identificador único do documento dentro de um índice.
  - Pode ser atribuído automaticamente ou manualmente via código.
- Um documento é uma **unidade básica de informação** que pode ser **indexada**.
- Para indexar os documentos → Elasticsearch usa **índice** (*index*).
- Os índices são o cerne da engine de armazenamento e busca do Elasticsearch.

## ▪ Índices:

- Coleção de documentos com características similares.
- Identificados por um nome.
- Nome em letras minúsculas.
- Nome usado nas operações.
- Correspondente ao conceito de database.
  - Ex: índice para dados do cliente.



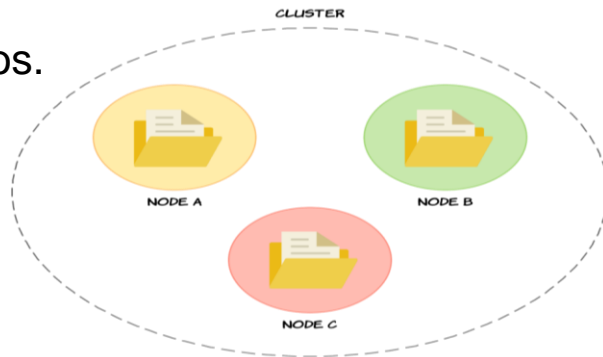
- Dentro de um índice: armazenar quantos documentos desejar ou couberem no local de armazenamento de dados do Elasticsearch.

- **Nó (*node*):**

- Servidor (físico ou virtual) que armazena dados.
- Instância do Elasticsearch.

- **Cluster:**

- Coleção de nós.
- Armazena e contém o conjunto de dados inteiro do cluster Elasticsearch.
- Cada nó participa das operações de indexação e pesquisa do cluster quando uma determinada operação envolve os dados que ele armazena ou armazenará.

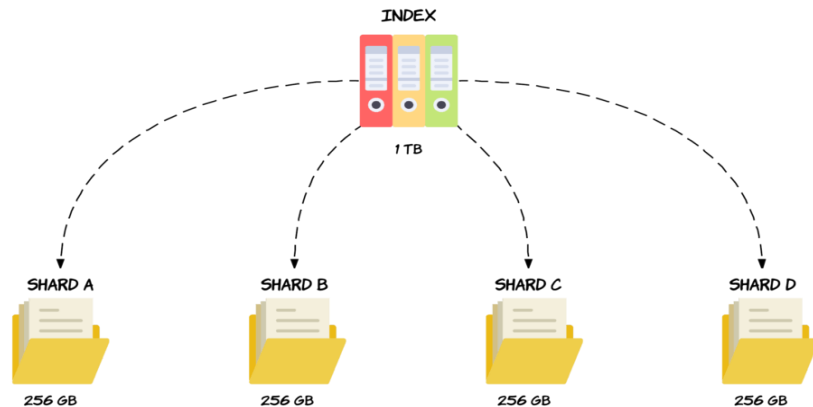


- Cluster expõe uma **REST API HTTP**.
  - Qualquer nó do cluster pode manipular solicitações HTTP.
- Cada nó dentro do cluster possui informações dos demais nós.
  - Capaz de encaminhar solicitações usando a camada interna de transporte.
- Todo cluster possui um **nó mestre (*master*)**:
  - Qualquer nó pode ser designado para ser o nó mestre por padrão.
  - Nó responsável por coordenar as alterações no cluster, como adicionar ou remover nós, criar ou remover índices, etc.
  - Incumbido de atualizar o estado do cluster.

- Todo cluster e nó são identificados por nomes exclusivos.
- Para **clusters**, o nome padrão é ***elasticsearch***.
- Para os **nós**, é um identificador universal exclusivo: **UUID**.
- É possível alterar esse padrão, de forma a personalizar os nomes.

## ▪ Sharding:

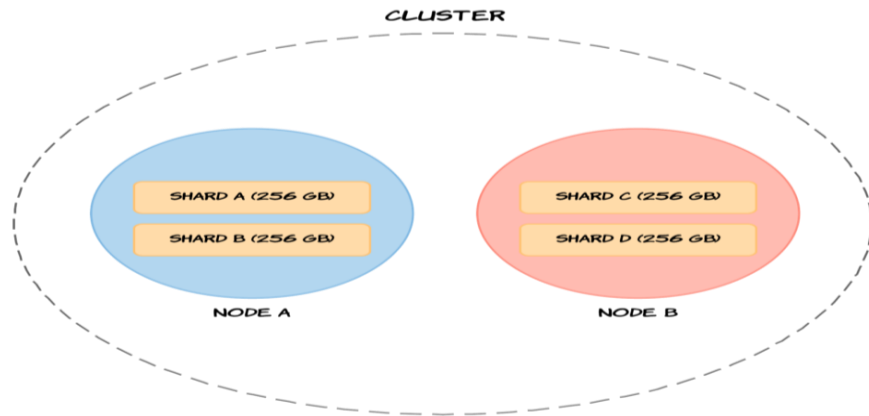
- Para **escalar** na topologia em cluster.
- Particionamento horizontal de dados.
- Divide os índices em partes menores.
  - **Shards** (fragmentos).



- Um shard só contém um subconjunto de dados de um índice.
- Um documento é armazenado em apenas um dos shards.
  - Não há divisões (*splits*) no nível de documentos.

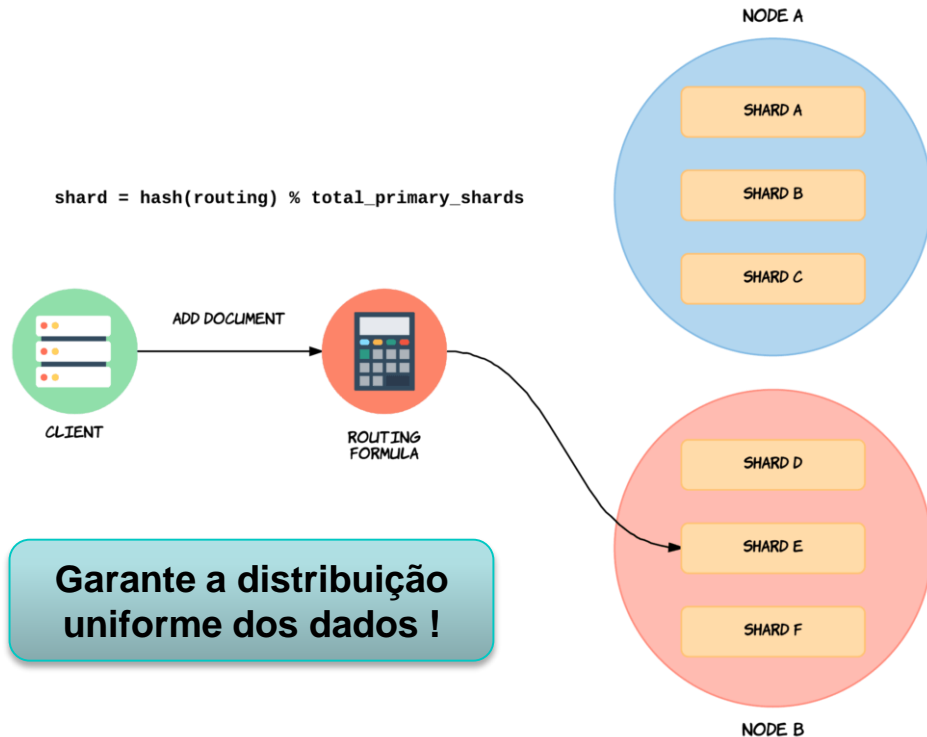
## ▪ Sharding:

- Quantidade de shards especificada no momento da criação do índice.
- Default de **5 shards**.
- Os shards podem ser hospedados em qualquer nó dentro do cluster.



## ▪ Sharding.

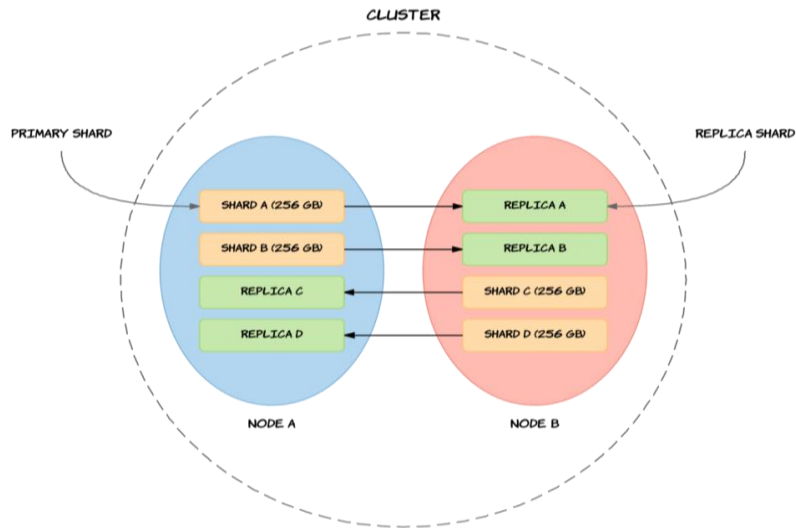
- Roteamento (*routing*): para saber em qual shard armazenar um novo documento e como encontra-lo, de forma transparente para os usuários.





## ▪ Replicação:

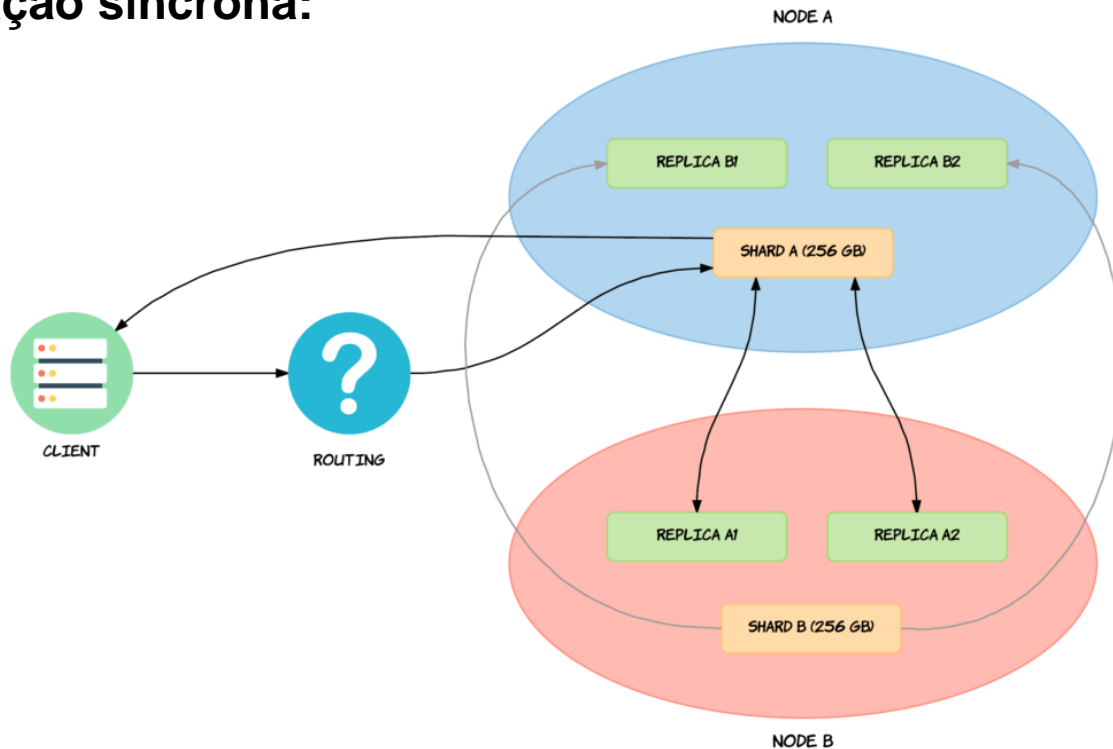
- Garantir a alta disponibilidade do ambiente e dos dados.
- Tolerância a falhas.
- Cópia de shards para outros nós:
  - **Fragmento primário (*primary shard*).**
  - Fragmentos replicados: **réplicas.**
  - **Grupo de replicação.**
- Default: 1 réplica por shard.
  - 1 shard primário e 1 shard réplica.



## ▪ Replicação:

- Topologia de replicação hierárquica: ***primary backup***.
- Operações que afetam o índice (adição, atualização ou remoção de documentos) são feitas apenas no shard primário.
- Shard primário é responsável por validar as operações e garantir que tudo esteja OK, estruturalmente e semanticamente falando.
- Shard primário, após validar e executar localmente a operação, encaminha para cada um dos shards uma réplica no grupo de replicação em questão.

- Replicação síncrona:



- ☑ Elasticsearch foi desenvolvido sobre o Lucene e possui client para diversas linguagens de programação.
- ☑ Armazena os dados em documentos, sendo os mesmos armazenados em índices, que são o cerne da engine.
- ☑ O Elasticsearch pode funcionar em topologia stand alone, mas o mais comum é em cluster com diversos nodes.
- ☑ Implementando shard e replicação, garante a alta disponibilidade, escalabilidade e processamento paralelo de consultas.

- ❑ Instalação e configuração do Elasticsearch e do Kibana.



## **Aula 6.4. Instalação e configuração do Elasticsearch e do Kibana**

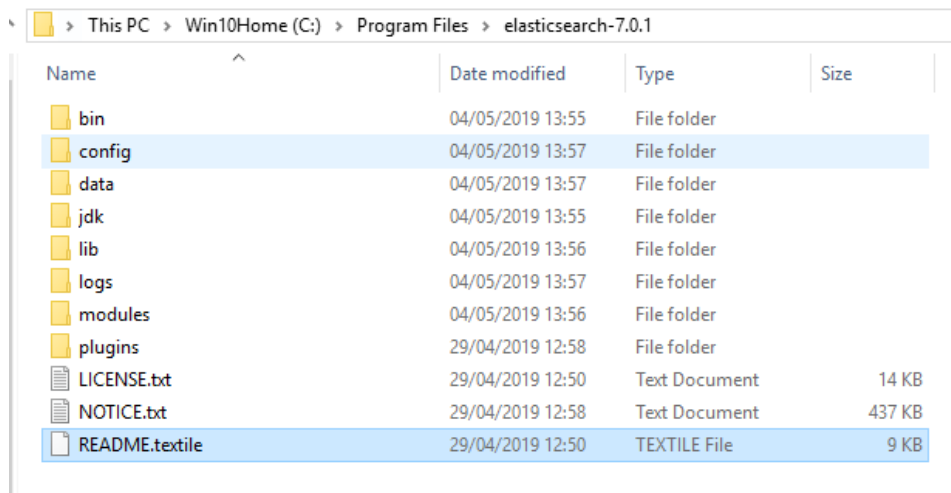
- ☐ Instalação e configuração do Elasticsearch.
- ☐ Instalação e configuração do Kibana.

- O Elasticsearch, assim como a maioria dos componentes do Elastic Stack, podem ser instalados em diversas plataformas com sistemas operacionais Windows, Linux e iOS, além de containers Docker.
- O Elasticsearch foi construído usando Java:
  - Inclui uma versão integrada do OpenJDK, com uma JVM empacotada.
  - Para o Elasticsearch funcionar é necessário o Java SE.
- Não oferece ainda a opção de instalar com repositório:
  - É preciso fazer o download.
    - Download em: <https://www.elastic.co/downloads/elasticsearch>.



# Instalação e configuração do Elasticsearch

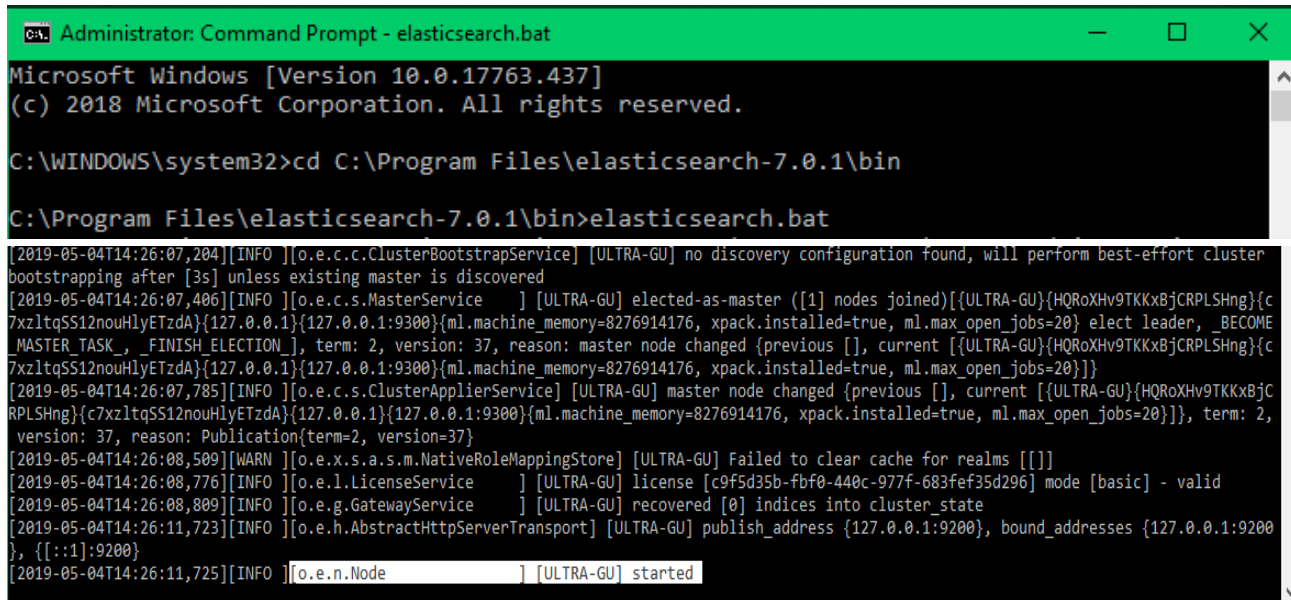
- No Windows → Há também um instalador MSI.
- Demais sistemas operacionais: baixar o pacote do produto compactado e descompactá-lo.



Name	Date modified	Type	Size
bin	04/05/2019 13:55	File folder	
config	04/05/2019 13:57	File folder	
data	04/05/2019 13:57	File folder	
jdk	04/05/2019 13:55	File folder	
lib	04/05/2019 13:56	File folder	
logs	04/05/2019 13:57	File folder	
modules	04/05/2019 13:56	File folder	
plugins	29/04/2019 12:58	File folder	
LICENSE.txt	29/04/2019 12:50	Text Document	14 KB
NOTICE.txt	29/04/2019 12:58	Text Document	437 KB
README.textile	29/04/2019 12:50	TEXTILE File	9 KB

# Instalação e configuração do Elasticsearch

- Para inicializar o Elasticsearch:
  - Executar **elasticsearch.bat** (Windows) / **elasticsearch** (demais).



```
Administrator: Command Prompt - elasticsearch.bat
Microsoft Windows [Version 10.0.17763.437]
(c) 2018 Microsoft Corporation. All rights reserved.

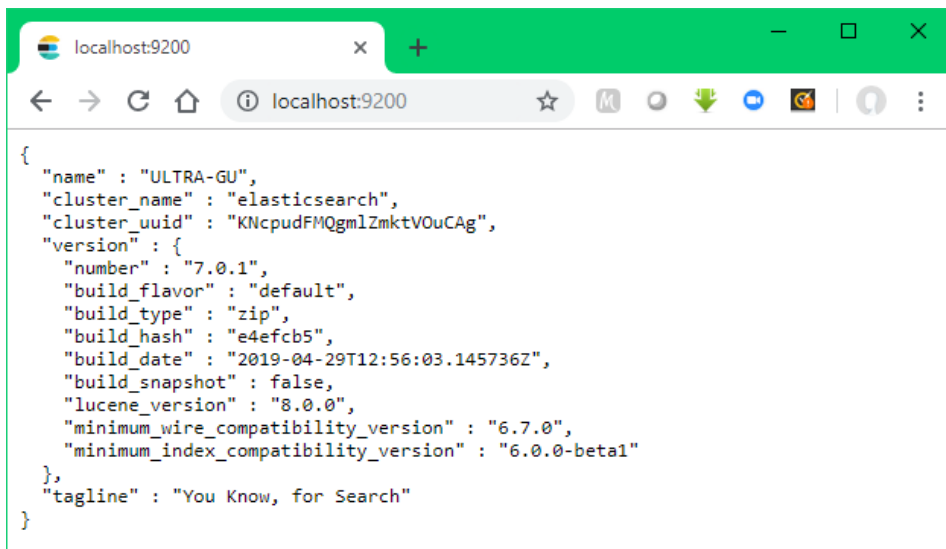
C:\WINDOWS\system32>cd C:\Program Files\elasticsearch-7.0.1\bin

C:\Program Files\elasticsearch-7.0.1\bin>elasticsearch.bat

[2019-05-04T14:26:07,204][INFO ][o.e.c.c.ClusterBootstrapService] [ULTRA-GU] no discovery configuration found, will perform best-effort cluster
bootstrapping after [3s] unless existing master is discovered
[2019-05-04T14:26:07,406][INFO ][o.e.c.s.MasterService] [ULTRA-GU] elected-as-master ([1] nodes joined){[ULTRA-GU]{HQRoXHv9TKKxBjCRPLSHng}{c
7xzltqSS12nouHlyETzdA}{127.0.0.1}{127.0.0.1:9300}{ml.machine_memory=8276914176, xpack.installed=true, ml.max_open_jobs=20} elect leader, _BECOME
_MASTER_TASK_, _FINISH_ELECTION_], term: 2, version: 37, reason: master node changed {previous [], current [{ULTRA-GU}{HQRoXHv9TKKxBjCRPLSHng}{c
7xzltqSS12nouHlyETzdA}{127.0.0.1}{127.0.0.1:9300}{ml.machine_memory=8276914176, xpack.installed=true, ml.max_open_jobs=20}}}
[2019-05-04T14:26:07,785][INFO ][o.e.c.s.ClusterApplierService] [ULTRA-GU] master node changed {previous [], current [{ULTRA-GU}{HQRoXHv9TKKxBjC
RPLSHng}{c7xzltqSS12nouHlyETzdA}{127.0.0.1}{127.0.0.1:9300}{ml.machine_memory=8276914176, xpack.installed=true, ml.max_open_jobs=20}]}, term: 2,
version: 37, reason: Publication{term=2, version=37}
[2019-05-04T14:26:08,509][WARN ][o.e.x.s.a.s.m.NativeRoleMappingStore] [ULTRA-GU] Failed to clear cache for realms [[]]
[2019-05-04T14:26:08,776][INFO ][o.e.l.LicenseService] [ULTRA-GU] license [c9f5d35b-fbf0-440c-977f-683fef35d296] mode [basic] - valid
[2019-05-04T14:26:08,809][INFO ][o.e.g.GatewayService] [ULTRA-GU] recovered [0] indices into cluster_state
[2019-05-04T14:26:11,723][INFO ][o.e.h.AbstractHttpServerTransport] [ULTRA-GU] publish_address {127.0.0.1:9200}, bound_addresses {127.0.0.1:9200
}, {[::1]:9200}
[2019-05-04T14:26:11,725][INFO ][o.e.n.Node] [ULTRA-GU] started
```

# Instalação e configuração do Elasticsearch

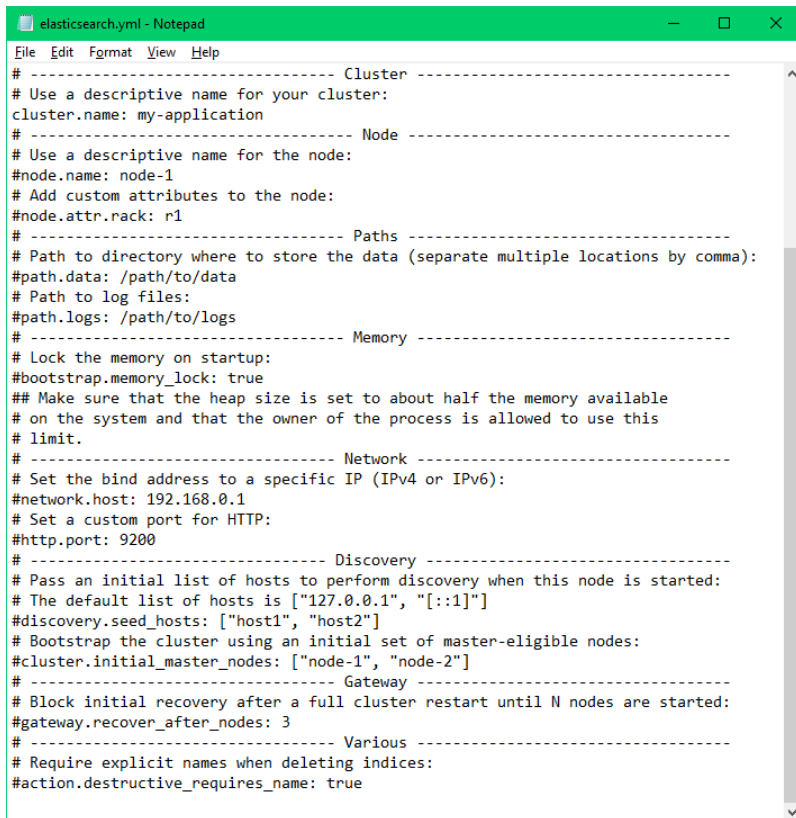
- Verificar se a instância do Elasticsearch está respondendo:
  - <http://localhost:9200/>

A screenshot of a web browser window with a green header bar. The address bar shows 'localhost:9200'. The main content area displays a JSON response from the Elasticsearch API. The JSON includes cluster information such as name, uuid, version (7.0.1), build details, and the tagline 'You Know, for Search'.

```
{
  "name" : "ULTRA-GU",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "KNcpudFMQgmlZmktVOuCAg",
  "version" : {
    "number" : "7.0.1",
    "build_flavor" : "default",
    "build_type" : "zip",
    "build_hash" : "e4efcb5",
    "build_date" : "2019-04-29T12:56:03.145736Z",
    "build_snapshot" : false,
    "lucene_version" : "8.0.0",
    "minimum_wire_compatibility_version" : "6.7.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

## ■ Configurações do Elasticsearch:

- Arquivo *elasticsearch.yml*
- No diretório */config*
- **cluster.name**
- **node.name**
- **network.host**
- **http.port**
- **discovery.seed\_hosts**

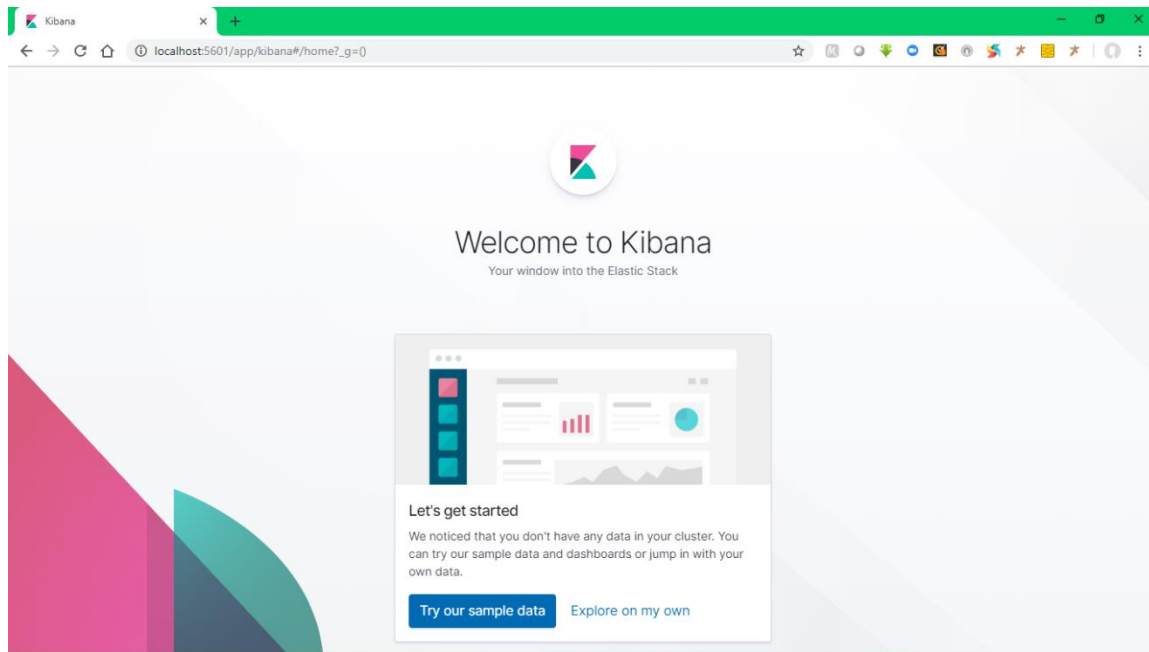


```
elasticsearch.yml - Notepad
File Edit Format View Help
# ----- Cluster -----
# Use a descriptive name for your cluster:
cluster.name: my-application
# ----- Node -----
# Use a descriptive name for the node:
#node.name: node-1
# Add custom attributes to the node:
#node.attr.rack: r1
# ----- Paths -----
# Path to directory where to store the data (separate multiple locations by comma):
#path.data: /path/to/data
# Path to log files:
#path.logs: /path/to/logs
# ----- Memory -----
# Lock the memory on startup:
#bootstrap.memory_lock: true
## Make sure that the heap size is set to about half the memory available
# on the system and that the owner of the process is allowed to use this
# limit.
# ----- Network -----
# Set the bind address to a specific IP (IPv4 or IPv6):
#network.host: 192.168.0.1
# Set a custom port for HTTP:
#http.port: 9200
# ----- Discovery -----
# Pass an initial list of hosts to perform discovery when this node is started:
# The default list of hosts is ["127.0.0.1", "[::1]"]
#discovery.seed_hosts: ["host1", "host2"]
# Bootstrap the cluster using an initial set of master-eligible nodes:
#cluster.initial_master_nodes: ["node-1", "node-2"]
# ----- Gateway -----
# Block initial recovery after a full cluster restart until N nodes are started:
#gateway.recover_after_nodes: 3
# ----- Various -----
# Require explicit names when deleting indices:
#action.destructive_requires_name: true
```

- Necessário fazer o download também:
  - [www.elastic.co/downloads/kibana](http://www.elastic.co/downloads/kibana)
- Baixar o pacote do produto compactado e descompactá-lo.
- Para **inicializar o Kibana**: executar ***kibana***, na pasta ***/bin***.
  - Será iniciado um web server no host em questão, na porta 5601.
  - Tentará conectar com o Elasticsearch, no mesmo host e na porta 9200.
- **Configurações do Kibana**:
  - Arquivo ***kibana.yml***, localizado no diretório ***/config***;
  - Relação completa: [www.elastic.co/guide/en/kibana/current/settings.html](http://www.elastic.co/guide/en/kibana/current/settings.html).

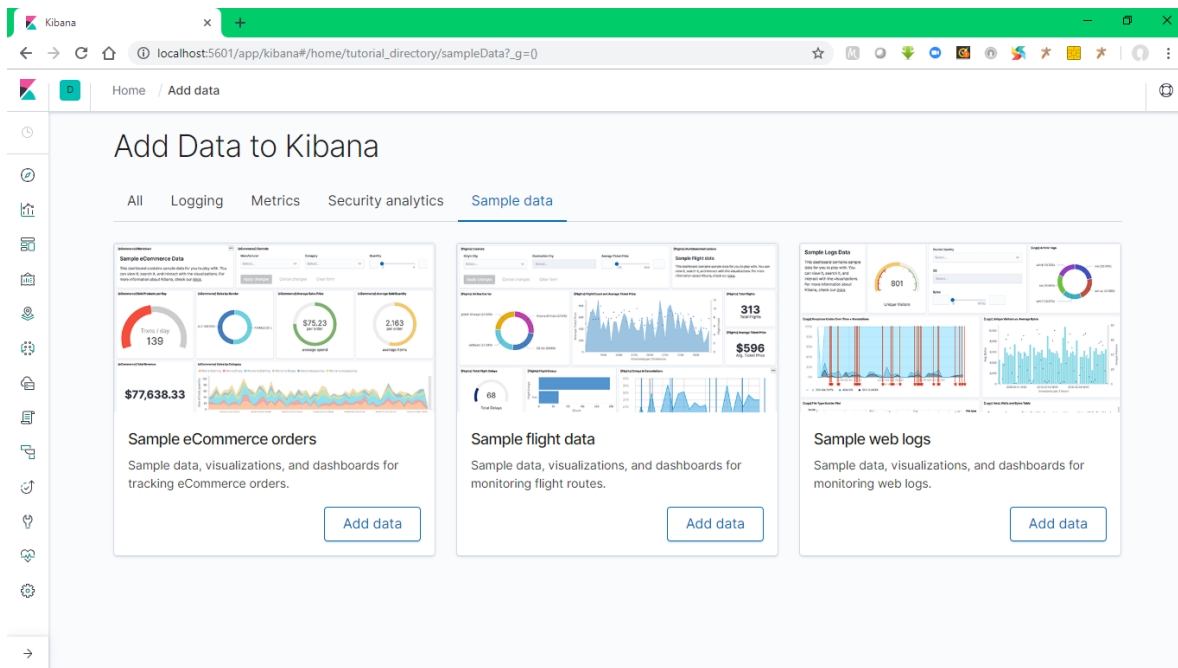
# Instalação e configuração do Kibana

- Utilização do Kibana: <http://localhost:5601/>



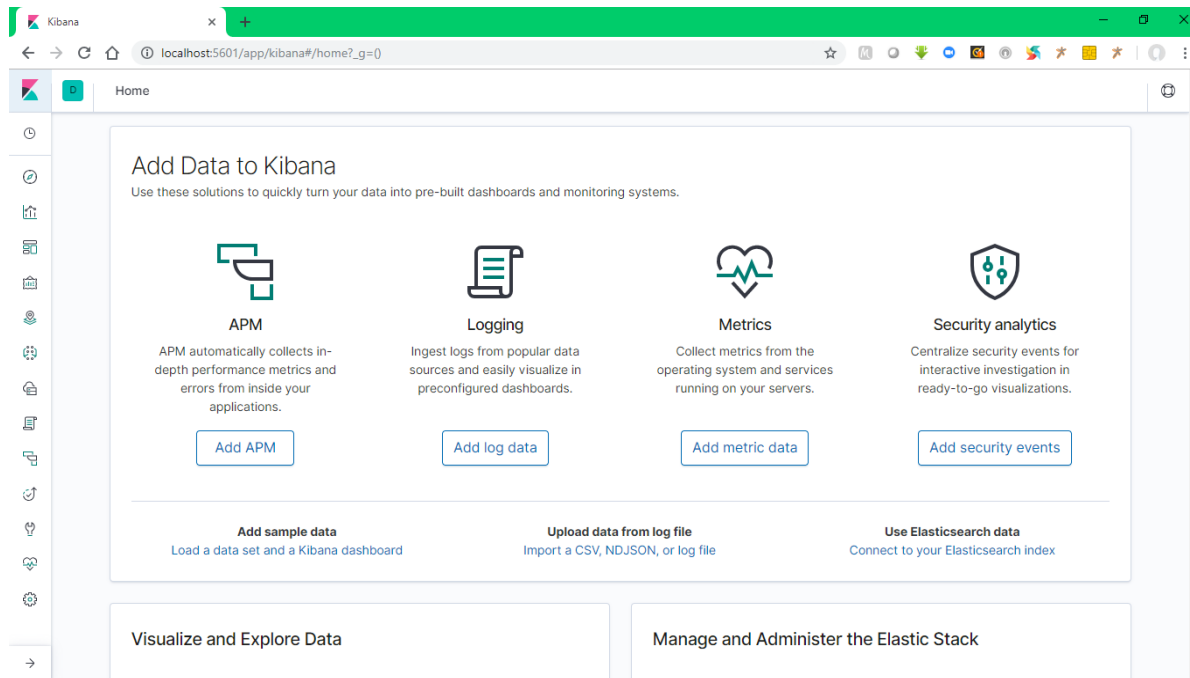
# Instalação e configuração do Kibana

- Importando dados de exemplo:



# Instalação e configuração do Kibana

- Página Home do Kibana.

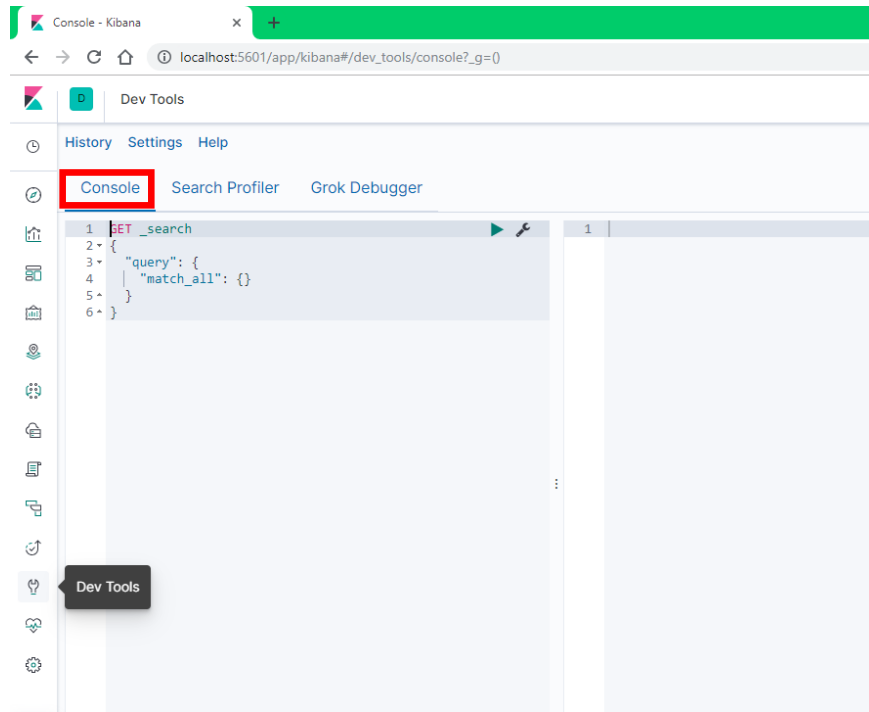




# Instalação e configuração do Kibana

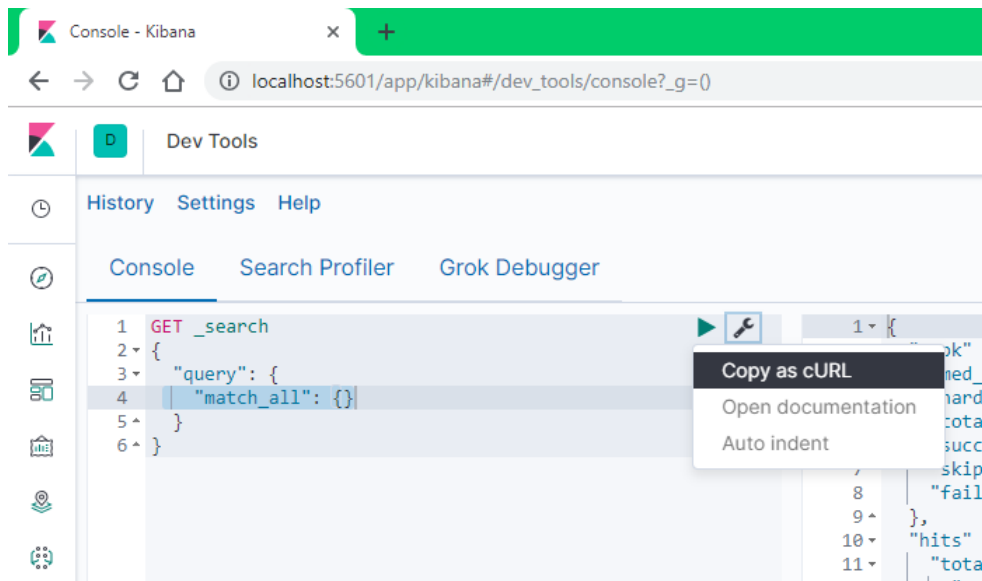
## ▪ **Console do *Dev Tools*:**

- Uma das interfaces mais utilizadas do Kibana.
- Criar / gerenciar índices.
- Inserir e manipular documentos.
- Sem preocupar com cabeçalhos HTTP ou em formatar respostas.
- Envio das requisições HTTP:
  - Verbo HTTP (*GET / POST / PUT / DELETE*) + Requisição URI + corpo (*body*) da requisição.



# Instalação e configuração do Kibana

- **Console** do **Dev Tools**:
  - Atalho para gerar o script no formato de requisições **cURL**.



- ☑ Instalação do Elasticsearch e do Kibana, podendo ser feita em praticamente todas as plataformas e sistemas operacionais.
- ☑ Instalação consiste basicamente em baixar o pacote do produto, descompactar e inicializar o serviço.
- ☑ Configurações são feitas nos arquivos de configuração (**elasticsearch.yml** e **kibana.yml**).
- ☑ Console do Dev Tools do Kibana é o client usado para interagir de forma direta com o Elasticsearch de uma maneira bem intuitiva.

- ❑ Definindo e administrando índices no Elasticsearch.

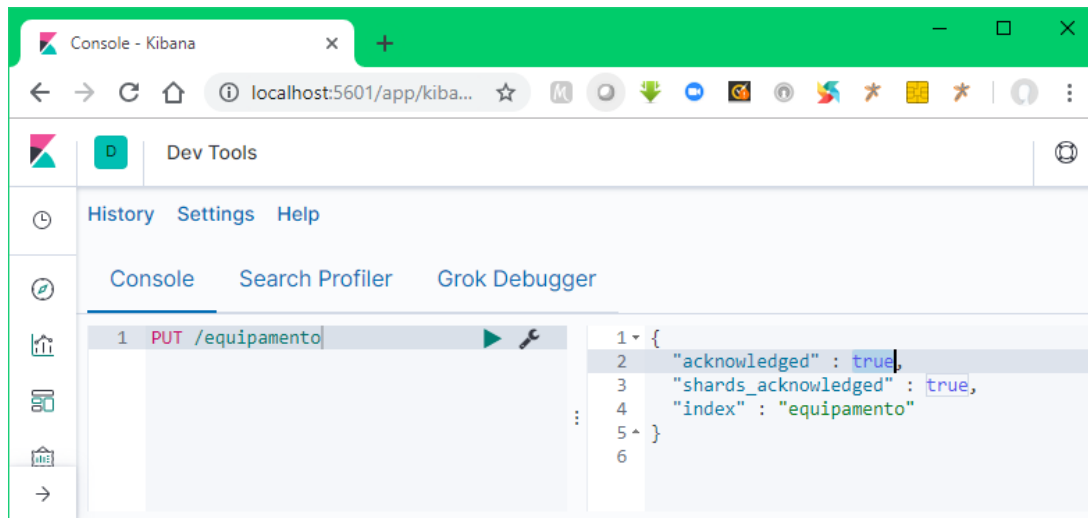


## **Aula 6.5. Definindo e administrando índices no Elasticsearch**

- ☐ Criando índices.
- ☐ Listando informações dos índices.
- ☐ Excluindo índices.

- Pode-se usar a Console do Dev Tools no Kibana, ou a interface de linha de comando com requisições cURL.

– ***PUT /NOME\_DO\_INDICE***



- **Regras do nome do índice:**

- Apenas letras minúsculas.
- Não pode conter \, /, \*, ?, ", <, >, |, espaço, vírgula, # ou :(a partir da 7.0).
- Não pode iniciar com -, \_, ou +.
- Não pode ser somente . ou ..
- Tamanho máximo de 255 **bytes**.

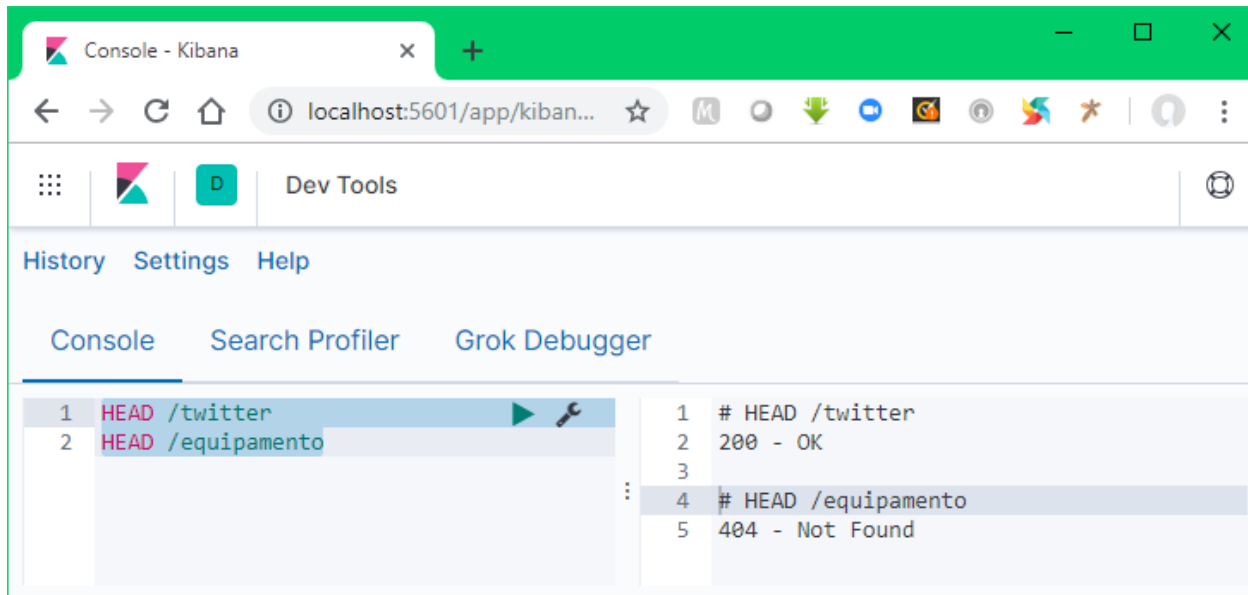
- **Configurações adicionais.**

```
PUT twitter
{
  "settings": {
    "number_of_shards": 3,
    "number_of_replicas": 2
  }
}
```



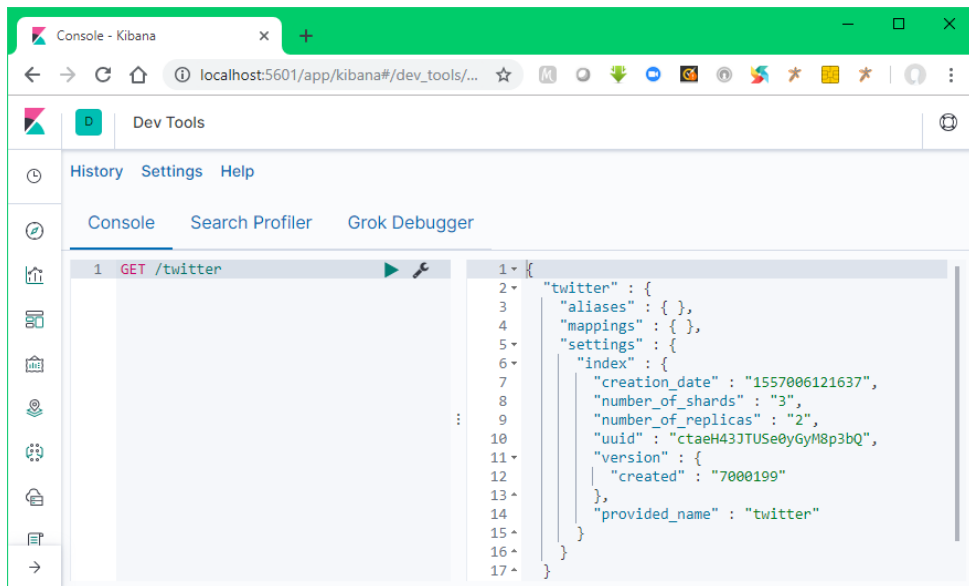
# Listando informações dos índices

- Verificar se um índice existe: ***HEAD /NOME\_DO\_INDICE.***



# Listando informações dos índices

- Listar informações e detalhes acerca de um índice:
  - GET /NOME\_DO\_INDICE.***

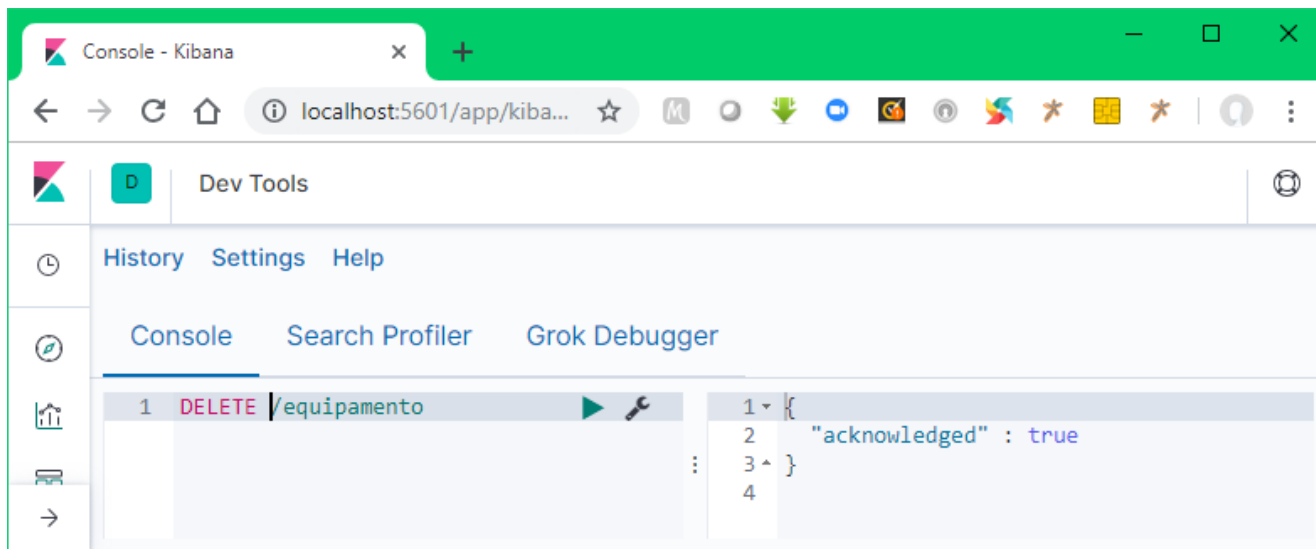


The screenshot shows the Kibana Dev Tools console. The left pane contains a single line of code: `1 GET /twitter`. The right pane displays the JSON response for this request, which is a detailed configuration for the 'twitter' index. The response includes metadata such as creation date, number of shards and replicas, UUID, and version information, as well as the provided name of the index.

```
1 {  
2   "twitter" : {  
3     "aliases" : { },  
4     "mappings" : { },  
5     "settings" : {  
6       "index" : {  
7         "creation_date" : "1557006121637",  
8         "number_of_shards" : "3",  
9         "number_of_replicas" : "2",  
10        "uuid" : "ctaeH43JTUSE0yGyM8p3bQ",  
11        "version" : {  
12          "created" : "7000199"  
13        },  
14        "provided_name" : "twitter"  
15      }  
16    }  
17  }
```

# Excluindo índices

- ***DELETE /NOME\_DO\_INDICE.***



- ☑ Criação de índices é feita usando o verbo HTTP **PUT** e há algumas regras básicas para a formação do nome do índice.
- ☑ É possível verificar a existência de um índice com **HEAD** e listar informações do índice com **GET**.
- ☑ Para excluir índices, deve-se usar o verbo HTTP **DELETE**.

- ☐ Trabalhando com documentos no Elasticsearch.



## **Aula 6.6. Trabalhando com documentos no Elasticsearch**

- ☐ Inserindo documentos.
- ☐ Consultando documentos.
- ☐ Alterando documentos.
- ☐ Excluindo documentos.

- Para inserir documentos no índice: requisição **POST** para uma URI.
  - Formada pelo nome do índice (não precisa estar criado previamente) +
  - Um *endpoint* (**\_doc** ou **\_create**) +
  - ID do documento (opcional, pois caso não seja informado a API invocada gerará um automaticamente) +
  - Documento JSON com os dados do documento que deseja-se inserir.



# Inserindo documentos

- Inserção de documento com ID explícito.

The screenshot shows the Kibana Dev Tools Console interface. The left pane displays the REST client request, and the right pane displays the JSON response.

**Request (Left Pane):**

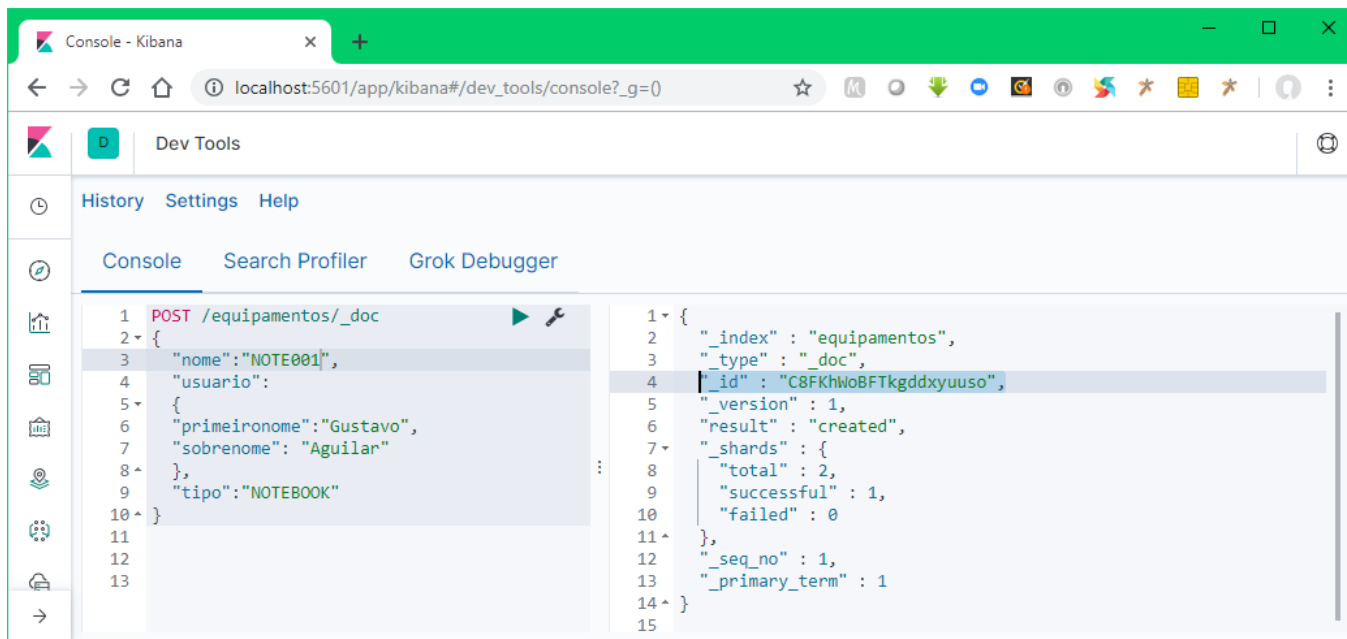
```
1 POST /equipamentos/_doc/1
2 {
3   "nome": "NOTE001",
4   "usuario":
5   {
6     "primeironome": "Gustavo",
7     "sobrenome": "Aguilar"
8   },
9   "tipo": "NOTEBOOK"
10 }
```

**Response (Right Pane):**

```
1 {
2   "_index": "equipamentos",
3   "_type": "_doc",
4   "_id": "1",
5   "_version": 1,
6   "result": "created",
7   "_shards": {
8     "total": 2,
9     "successful": 1,
10    "failed": 0
11  },
12   "_seq_no": 0,
13   "_primary_term": 1
14 }
```

# Inserindo documentos

- Inserção de documento com ID automático.



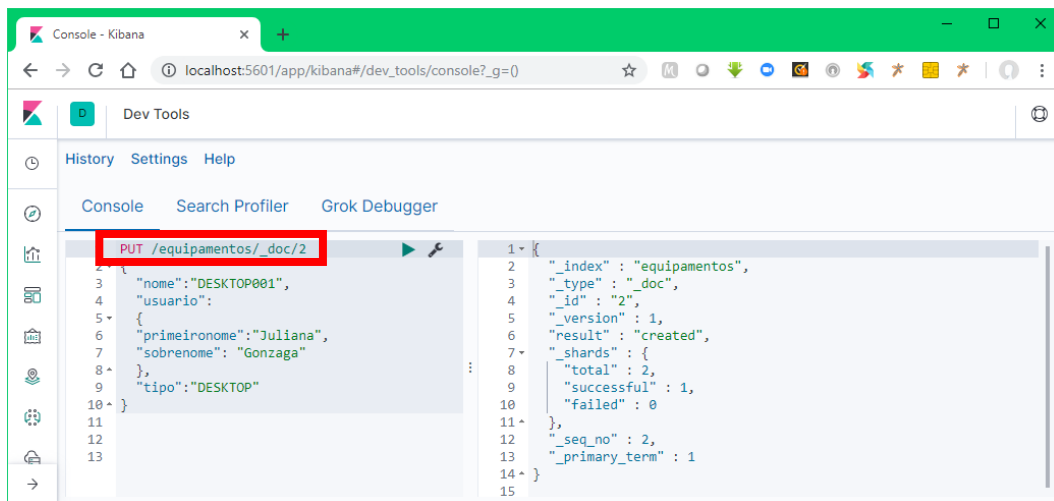
The screenshot shows the Kibana Dev Tools console interface. The left pane displays the REST client with a POST request to `/equipamentos/_doc`. The request body is a JSON document with the following fields: `nome` (NOTE001), `usuario` (Gustavo Aguilar), and `tipo` (NOTEBOOK). The right pane shows the response, which is a JSON object indicating a successful creation of the document. The response includes the index name (`equipamentos`), type (`_doc`), a version number (`1`), and an automatically generated ID (`C8FKhWoBFTkgddxyuuso`). The response also shows the number of shards (`2`), the success status (`1`), and the failure status (`0`).

```
1 POST /equipamentos/_doc
2 {
3   "nome": "NOTE001",
4   "usuario": {
5     "primeironome": "Gustavo",
6     "sobrenome": "Aguilar"
7   },
8   "tipo": "NOTEBOOK"
9 }
10
11
12
13
```

```
1 {
2   "_index": "equipamentos",
3   "_type": "_doc",
4   "_id": "C8FKhWoBFTkgddxyuuso",
5   "_version": 1,
6   "result": "created",
7   "_shards": {
8     "total": 2,
9     "successful": 1,
10    "failed": 0
11  },
12   "_seq_no": 1,
13   "_primary_term": 1
14 }
```

# Inserindo documentos

- Pode-se usar também o verbo HTTP **PUT**:
  - De forma similar ao POST.
  - É preciso especificar, de forma explícita, o ID do documento.



The screenshot shows the Kibana DevTools Console interface. The address bar indicates the URL is `localhost:5601/app/kibana#/dev_tools/console?_g=0`. The console is divided into two main sections: the left pane shows the raw HTTP request, and the right pane shows the JSON response.

In the left pane, the request is a `PUT` to `/equipamentos/_doc/2`. The request body is a JSON object:

```
1 {
2   "name": "DESKTOP001",
3   "usuario": {
4     "primeironome": "Juliana",
5     "sobrenome": "Gonzaga"
6   },
7   "tipo": "DESKTOP"
8 }
```

In the right pane, the response is a JSON object indicating the document was created successfully:

```
1 {
2   "_index": "equipamentos",
3   "_type": "_doc",
4   "_id": "2",
5   "_version": 1,
6   "result": "created",
7   "_shards": {
8     "total": 2,
9     "successful": 1,
10    "failed": 0
11  },
12   "_seq_no": 2,
13   "_primary_term": 1
14 }
```

## ▪ Observação sobre tipos:

- Desde a primeira versão até a versão 7, os endpoints podiam ter um tipo (*type*) → documento armazenado em um índice e atribuído à um tipo.
- Cada tipo podia ter seus próprios campos.
- Por exemplo, um índice do twitter → tipo *usuário* e um tipo *tweet*.
  - Tipo usuário: campo *nome*, campo *nomeusuario* e um campo *email*.
  - Tipo tweet: campo *conteudo*, *datapostagem* e o campo *nome\_usuario*.
- Entretanto, já está sendo tratado como descontinuado.
- Nas versões novas, deve-se usar os endpoints sem tipo (`{{index}}/_doc/{id}`, `{{index}}/_doc`, ou `{{index}}/_create/{id}`);

- **Observação sobre tipos:**

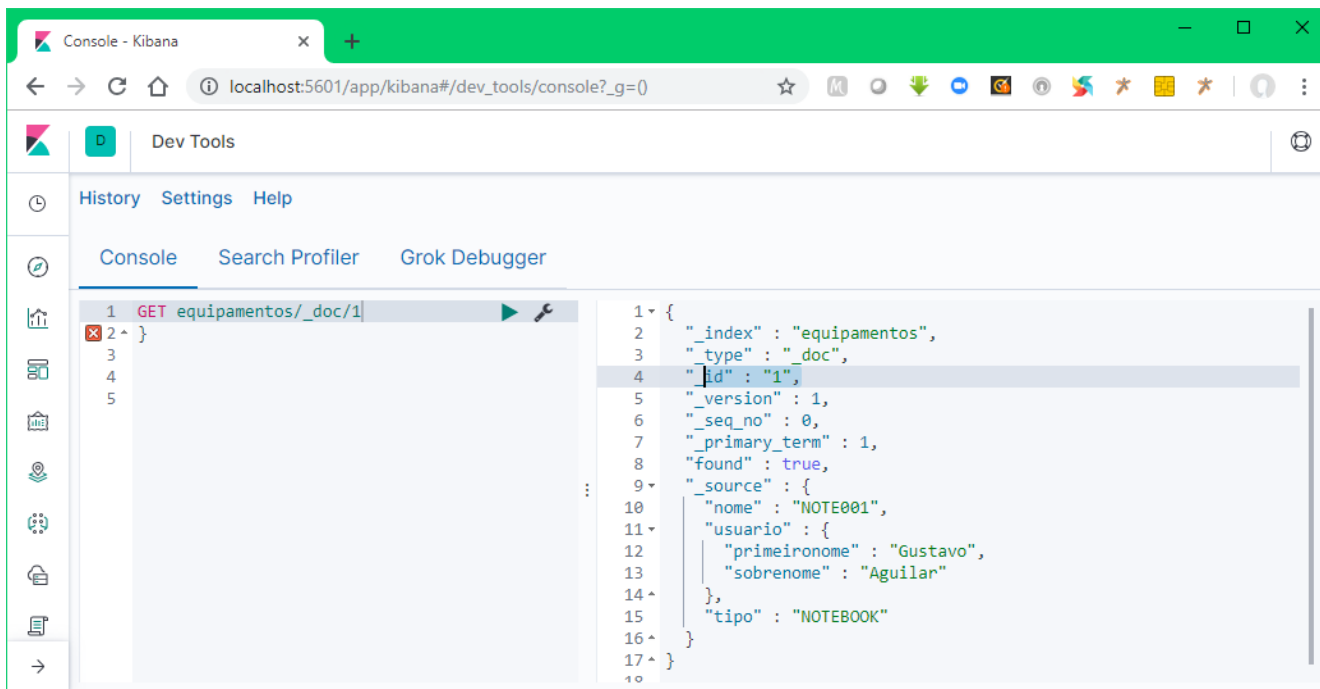
- Na versão 7 ainda aceita-se a sintaxe antiga, mas exibe um aviso:

```
1 PUT /equipamentos2/default/1
2 {
3   "nome": "DESKTOP001",
4   "usuario":
5   {
6     "primeironome": "Juliana",
7     "sobrenome": "Gonzaga"
8   },
9   "tipo": "DESKTOP"
10 }
```

```
1 #! Deprecation: [types removal] Specifying types in document
  index requests is deprecated, use the typeless endpoints
  instead ({index}/_doc/{id}, {index}/_doc, or {index}
  /_create/{id}).
2 {
3   "_index" : "equipamentos2",
4   "_type" : "default",
5   "_id" : "1",
6   "_version" : 1,
7   "result" : "created",
8   "type" : "DESKTOP"
```

# Consultando documentos

- Consultar documentos pelo ID → Verbo HTTP **GET**.



The screenshot shows the Kibana Dev Tools console interface. The browser address bar indicates the URL is `localhost:5601/app/kibana#/dev_tools/console?_g=()`. The console tab is active, showing a list of commands on the left and the response on the right. The command entered is `GET equipamentos/_doc/1`. The response is a JSON object representing a document.

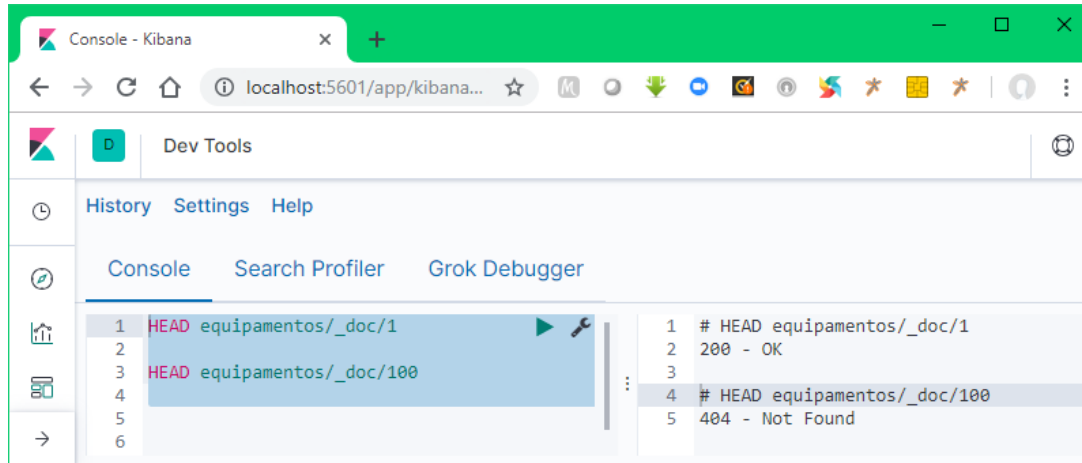
```
1 GET equipamentos/_doc/1
2 }
3
4
5
```

```
1 {
2   "_index" : "equipamentos",
3   "_type" : "_doc",
4   "_id" : "1",
5   "_version" : 1,
6   "_seq_no" : 0,
7   "_primary_term" : 1,
8   "found" : true,
9   "_source" : {
10    "nome" : "NOTE001",
11    "usuario" : {
12      "primeironome" : "Gustavo",
13      "sobrenome" : "Aguilar"
14    },
15    "tipo" : "NOTEBOOK"
16  }
17 }
```

- Consultar mais documentos → API **\_mget** em conjunto com o **GET**:

*GET equipamentos/\_doc/\_mget { "ids": [ "1","2" ] }*

- Verificar existência de um documento pelo ID → verbo **HEAD**:

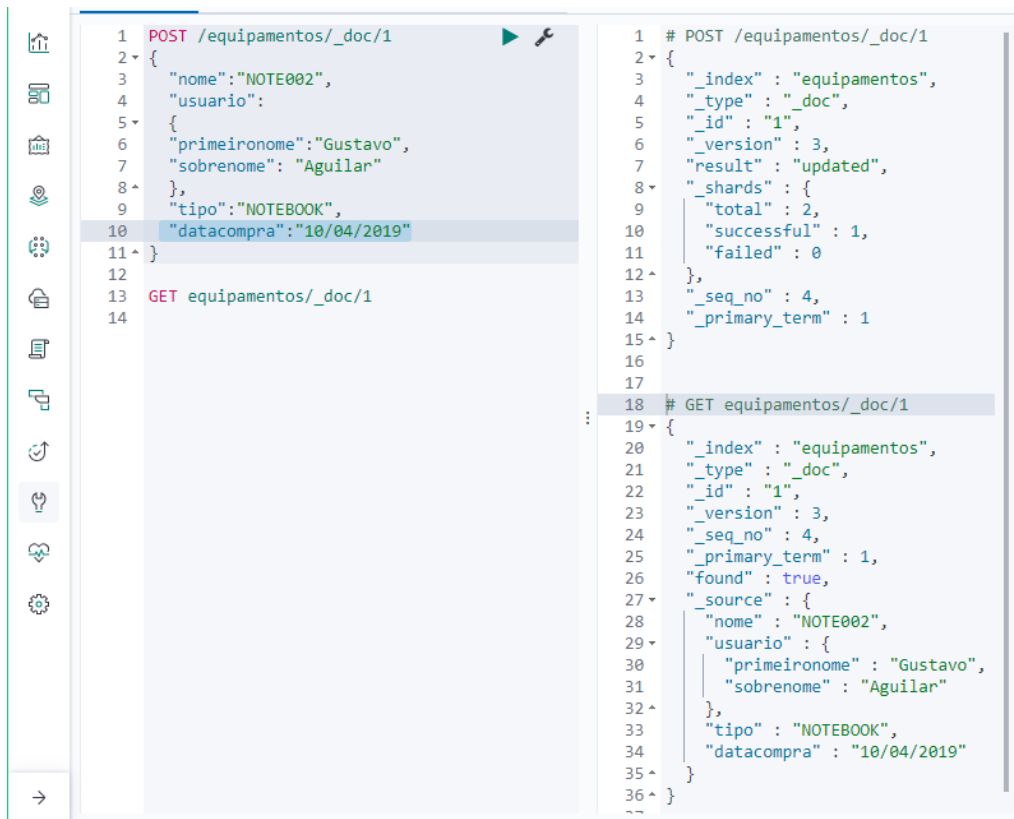


- Duas estratégias: **substituição (*replace*)** e **atualização (*update*)**.
- **Substituição (*replace*):**
  - Substituir um documento, atualizando o valor de um ou mais campos, removendo ou acrescentando novos campos.
  - Comando de criação de documento.
  - Passar o ID do documento a ser substituído.



# Alterando documentos

- Substituição (*replace*):



```
1 POST /equipamentos/_doc/1
2 {
3   "nome": "NOTE002",
4   "usuario": {
5     {
6       "primeironome": "Gustavo",
7       "sobrenome": "Aguilar"
8     },
9     "tipo": "NOTEBOOK",
10    "datacompra": "10/04/2019"
11  }
12 }
13 GET equipamentos/_doc/1
14
```

```
1 # POST /equipamentos/_doc/1
2 {
3   "_index": "equipamentos",
4   "_type": "_doc",
5   "_id": "1",
6   "_version": 3,
7   "result": "updated",
8   "_shards": {
9     "total": 2,
10    "successful": 1,
11    "failed": 0
12  },
13   "_seq_no": 4,
14   "_primary_term": 1
15 }
16
17
18 # GET equipamentos/_doc/1
19 {
20   "_index": "equipamentos",
21   "_type": "_doc",
22   "_id": "1",
23   "_version": 3,
24   "_seq_no": 4,
25   "_primary_term": 1,
26   "found": true,
27   "_source": {
28     "nome": "NOTE002",
29     "usuario": {
30       "primeironome": "Gustavo",
31       "sobrenome": "Aguilar"
32     },
33     "tipo": "NOTEBOOK",
34     "datacompra": "10/04/2019"
35   }
36 }
```

# Alterando documentos

## ■ Atualização (*update*):

- Endpoint *\_update*
- *POST NOME\_INDICE/\_update/ID*

{

“doc”:

{ ATUALIZACAO\_AQUI }

}

- Mantém os outros campos.

```
1 POST /equipamentos/_update/2
2 {
3   "doc": { "nome": "DESKTOP002",
4           | "fornecedor": "DELL"
5         }
6 }
7
8 GET equipamentos/_doc/2
9
10
11
```

```
1 # POST /equipamentos/_update/2
2 {
3   "_index" : "equipamentos",
4   "_type" : "_doc",
5   "_id" : "2",
6   "_version" : 2,
7   "result" : "updated",
8   "_shards" : {
9     "total" : 2,
10    "successful" : 1,
11    "failed" : 0
12  },
13   "_seq_no" : 9,
14   "_primary_term" : 1
15 }
16
17
18 # GET equipamentos/_doc/2
19 {
20   "_index" : "equipamentos",
21   "_type" : "_doc",
22   "_id" : "2",
23   "_version" : 2,
24   "_seq_no" : 9,
25   "_primary_term" : 1,
26   "found" : true,
27   "_source" : {
28     "nome" : "DESKTOP002",
29     "usuario" : {
30       "primeironome" : "Juliana",
31       "sobrenome" : "Gonzaga"
32     },
33     "tipo" : "DESKTOP",
34     "fornecedor" : "DELL"
35   }
36 }
```

## ▪ *Scripted updates:*

- Atualizar um documento via script e não apenas com um valor literal.
- Permite fazer-se cálculos e operações mais complexas de atualização.
- Usa o campo de metadado **source**, junto com as variáveis **ctx** e **params**.

- Possível acessar os campos do documento.
- Passar valores de parâmetros para o script.
- Definir a linguagem a ser utilizada no código do script.

```
POST postagens/_update/1
{
  "script" : {
    "source": "ctx._source.contador += params.contagem",
    "lang": "painless",
    "params" : {
      "contagem" : 4
    }
  }
}
```

- Atualizar mais de um documento por vez → ***update\_by\_query***

```
POST postagens/_update_by_query
{
  "script": {
    "source": "ctx._source.curtidas++",
    "lang": "painless"
  },
  "query": {
    "term": {
      "usuario": "Gustavo"
    }
  }
}
```

# Excluindo documentos

- Deletar apenas um documento:
  - Verbo HTTP ***DELETE***

```
1 DELETE postagens/_doc/1
2
3 GET postagens/_doc/1
4
5
```

```
1 # DELETE postagens/_doc/1
2 {
3   "_index" : "postagens",
4   "_type" : "_doc",
5   "_id" : "1",
6   "_version" : 6,
7   "result" : "not_found",
8   "_shards" : {
9     "total" : 2,
10    "successful" : 1,
11    "failed" : 0
12  },
13   "_seq_no" : 5,
14   "_primary_term" : 1
15 }
16
17
18 # GET postagens/_doc/1
19 {
20   "_index" : "postagens",
21   "_type" : "_doc",
22   "_id" : "1",
23   "found" : false
24 }
25
```

- Deletar um conjunto de documentos:

- Verbo HTTP **POST**;
- API ***\_delete\_by\_query***;
- Passar a variável ***query***
  - Será o filtro (condição *where*).

```
POST equipamentos/_delete_by_query
{
  "query": {
    "match": {"tipo": "DESKTOP"}
  }
}
```

*POST NOME\_INDICE/\_delete\_by\_query*

```
{
  "query": {
    "match": {"campo": "valor" }
  }
}
```

- ☑ A inserção, alteração e exclusão de documentos são feitas usando requisições REST.
- ☑ Há opções para atuar sobre apenas um documento ou sobre um conjunto de documentos.

- ☐ Overview de agregações no Elasticsearch.





## **Aula 6.7. Overview de gregações no Elasticsearch**

- ❑ Overview de agregações no Elasticsearch.

- **Objetivo:** fornecer dados sumarizados agrupados com base em uma pesquisa;
- Uma unidade de trabalho que cria informações analíticas sobre um conjunto de documentos;
- Existem vários tipos de agregações, cada uma com sua finalidade e saída de dados;
- São divididas em **quatro famílias**.

## ▪ Bucketing:

- Família de agregações que criam depósitos (buckets), onde cada depósito é associado a uma chave e a um critério de documento;
- Quando a agregação é executada, todos os critérios dos buckets são avaliados em cada documento;
- Quando há uma correspondência do critério, considera-se que o documento "está incluído" no intervalo relevante;
- No final do processo de agregação, haverá uma lista de intervalos, cada um com seu conjunto de documentos.

- **Matriz:**
  - Ações que operam em vários campos do documento;
  - Produzem um resultado de matriz com base nos valores extraídos dos campos do documento solicitado.
- **Métrica:** agregações que acompanham e calculam as métricas em um conjunto de documentos.
- **Pipeline:** agregações que consolidam a saída de outras agregações e suas métricas associadas.

- Agregações mais utilizadas → da família de métricas;
- Utilizadas em larga escala no mundo relacional, para somar, contar, calcular média, valor mínimo, máximo, etc;
- Usa-se o verbo HTTP **POST** junto da API **aggs**, aliada à função desejada:

```
POST /vendas/_search?size=0
```

```
{  
  "aggs" : {  
    "valor_maximo" : { "max" : { "field" : "total_da_venda" } }  
  }  
}
```

- ☑ Agregações são recursos muito úteis para agrupar, segmentar e sumarizar documentos e são divididas em quatro famílias, cada uma com seus tipos de agregações e funcionalidades específicas.

# ■ Próxima aula

- ❑ Overview de Mapping.





## **Aula 6.8. Overview de Mapping**

- ☐ Overview de Mapping.

- **Mapeamento (*Mapping*):** definição de como um documento e os campos que ele contém serão armazenados e indexados:
  - Quais campos do tipo *string* devem ser tratados como campos *full text*;
  - Quais campos contêm números, datas ou geolocalização;
  - O formato de datas;
  - Regras customizadas para controlar o mapeamento para campos adicionados dinamicamente.

- Cada campo dos índices tem um tipo de dados que pode ser:
  - Um tipo simples como texto, palavra-chave, data, long, double, boolean ou IP;
  - Um tipo que suporta a natureza hierárquica do JSON, como *object* ou *nested* ou um tipo especializado como *geo\_point* ou *geo\_shape*.
- **Mapeamento Dinâmico:**
  - Mapeamento não precisa ser definido antes;
  - Novos campos são adicionados automaticamente, apenas pela indexação de um novo documento.

## ▪ Mapeamento Explícito:

- Especifica de forma explícita os tipos de dados;
- Criando o índice e adicionando campos à ele com o verbo **PUT**.

```
PUT my_index ①
{ "mappings" : { "properties" : {

  ②
  "title" : { "type" : "text" },           ③
  "name" : { "type" : "text" },           ④
  "age" : { "type" : "integer" },          ⑤
  "created" : { "type" : "date" ,
    ⑥
    "format" : "strict_date_optional_time || epoch_millis" } } } }
```

- ☑ *Mapping* é o processo de mapear os tipos e formatos de dados armazenados nos documentos dentro do Elasticsearch.
- ☑ Pode ser feito de forma dinâmica, bastando indexar um documento novo, ou de forma explícita com o verbo PUT.