APIs Elastic Stack

Índice de contenidos

1. Introducción. ¿Qué es una API?

- ***Application programming interface***: es un conjunto de subrutinas y funciones para ser utilizada por otro software como una capa de abstracción.
- Acepta llamadas a bibliotecas que acceden a servicios.
  - Cluster API
  - Cat API
  - Index API
  - Search API
  - Documents API
  - Ingest API
  - ...

1. API. Usos en Elasticsearch

- *Kibana, Logstash, Beats* y otras aplicaciones de terceros usan estas API para la comunicación.
- *Cluster* y *Cat* API nos darán información de recogida de información, recogida de datos, estado de nodos, *shards*, etc.
- *Index API. Consultas a índices e interactuar con ellos en la misma BD.*
- *Documents* API. Inserta y elimina documentos entre otras acciones.
- *Search* API. Realizar búsquedas.

2. Cluster API. *Query Cluster Health*

- Consulta el estado del *cluster* (*name, status by colour, number of nodes, shards, tasks…*):
  - ***Query Cluster Health***:
    - `GET _cluster/health?pretty`
- Estado del *cluster*:
  - ***Green***: Todos los *shards* están bien indexados.
  - ***Yellow***: *Shards* primarios correctos pero fallo en las réplicas.
  - ***Red***: *Shards* primarios no indexados.

2. Cluster API. *Query Cluster Stats*

- Consulta estadísticas del *cluster* (*number of shards, storage use, memory use…*):
  - ***Query Cluster Stats***:
    - `GET _cluster/stats?human&pretty`
- Devuelve también información sobre los nodos:
  - *Status* y cantidad
  - Índices, Documentos
  - Uso de cache
  - Nodos (SO, uso de memoria, procesos,)
- Nota: human nos permite leer las cantidades de memoria de modo más fácil.

2. Cluster API. *Query Cluster Nodes*

- Información de los nodos del *cluster* (name, listening port, IP, SO, proceses, installed plugins…):
  - ***Query Cluster Nodes Info***:
    - `GET _nodes?pretty`
    - `GET _nodes/nodeID1,nodeID2?pretty`
- Devuelve también información sobre los nodos:
  - *Status* y cantidad
  - Índices
  - Documentos
  - Uso de cache
  - Nodos (SO, uso de mem, procesos,)

2. Cat API. *Query Health Info*

- En la API *Cluster* la información viene formateada en JSON y es de difícil lectura por humanos.
- La **API Cat** viene a resolver esto: Muestra la información de forma compacta y tabulada.
  - ***Query Health Info***: (= *Query Cluster Health*)
    - `GET _cat/health?v`

2. Cat API.

- Información sobre el nodo máster de un *cluster*.
  - ***Query Master Info***:
    - `GET _cat/master?v`
- Listado de índices (BBDDs) (*status, number of primary and replicated shards, documents* y su *storage size*)
  - ***Query* Índices *Info***:
    - `GET _cat/indices?v`
- Información sobre los *shards* (*location, pri/rep, status*...)
  - ***Query Shards Info***:
    - `GET _cat/shards?v`

2. REST APIs.

- En la web oficial de *Elastic* se puede consultar todas las REST APIs actuales.
- https://www.elastic.co/guide/en/elasticsearch/reference/current/rest-apis.html

- **Hacer las actividades 04.1_API_REST_estado_Preguntas.odt**

3. API REST. Manipulación de *index* y *docs*

- Veremos la interacción de *Logstash*, *Kibana* y aplicaciones de terceros con los datos, índices y documentos.
- Insertar documentos, como se consultan, etc.
- Como se pueden actualizar, modificar, crear y borrar índices
- Trabajaremos con 3 APIs que interactúan con la información almacenada.
  - Document API
  - Index API
  - Search API

3. API REST. Manipulación de *index* y *docs*

- Por norma general ***Documents*** e ***Indices API*** insertan datos y ***Search API*** consulta datos.
- ***Beats*** y ***Logstash*** usan las ***APIs Documents*** e ***Indices***
- ***Kibana*** y ***Grafana*** usan la ***API Search***

3. Documents APIs. Index API

- https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-index_.html

- Añade un documento JSON al flujo de datos o índice especificado y lo hace consultable. Si el destino es un índice y el documento ya existe, la solicitud actualiza el documento e incrementa su versión.
- Estructura de la URL `IP/target/subconjunto/ID`

```
curl -XPUT '10.0.0.10:9200/agenda/_doc/1?pretty'
-H 'Content-Type: application/json' -d'

{"user": "vicent",

"tel": "693826549"}'
```

3. Documents APIs. Get API

- https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-get.html

- Recupera el documento JSON especificado de un índice.
- Es la consulta más básica, pasando un solo índice.

```
curl -XGET '10.0.0.10:9200/agenda/_doc/2?pretty'
```

3. Documents APIs. Delete API

- https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-delete.html

- Elimina el documento JSON del índice especificado.

```
curl -XDELETE '10.0.0.10:9200/agenda/_doc/2?pretty'
```

3. Documents APIs. Update API

- https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-update.html

- Actualiza un documento utilizando el *script* especificado. Consulta el documento, aplica el *script* y vuelve a indexar el documento.

```
curl -XPOST '10.0.0.10:9200/agenda/tel/2/_update?
pretty' -H 'Content-Type: application/json' -
d'{"script":"ctx._source.email = \"neus@ono.es\""}'
```

3. Search APIs. URI Search API

- https://www.elastic.co/guide/en/elasticsearch/reference/current/search-search.html
- La *query* se inserta como parámetro de la URI.
- Devuelve los resultados de la búsqueda que coinciden con la consulta definida en la solicitud.
- Busca en el índice *twitter* los *tweets* del *user* user1
  - # curl -XGET '10.0.0.10:9200/agenda/_search?q=user:vicent&pretty'
- Búsqueda multi-índice. Busca los documentos que tienen el campo *likes*
  - # curl -XGET '10.0.0.10:9200/_search?q=email:neus@ono.es&pretty'

3. Search APIs. Request Body Search API

- *Elasticsearch* proporciona un completo *Query DSL* (*Domain Specific Language*) basado en JSON para definir consultas.
- La *query* se encapsula en el JSON.

```
curl -XGET '10.0.0.10:9200/agenda/tel/_search?pretty' -H
'Content-Type: application/json' -d'

{

"query": {

        "term": {

            "user": "neus"

        }

    }

}'
```

3. Search APIs. *Search templates*

- https://www.elastic.co/guide/en/elasticsearch/reference/current/search-template.html

- Una plantilla de búsqueda que se puede ejecutar con diferentes variables permite cambiar las búsquedas sin modificar el código de la aplicación.
- Ver consulta en siguiente diapositiva:

3. Search APIs. *Search templates*

```
curl -XGET '10.0.0.10: 9200/_search/template?pretty' -H 'Content-Type: application/json' -d' {
    "source": {
        "query": {
            "match": {
                "{{my_field}}": "{{my_value}}"
            }
        },
        "size": "{{my_size}}"
    },
    "params": {
        "my_field": "user",
        "my_value": "neus",
        "my_size": 4
    }
}'
```

3. Search APIs. *Search Shards*

- https://www.elastic.co/guide/en/elasticsearch/reference/current/search-shards.html
- Devuelve los índices y *shards* contra los que se ejecutaría una búsqueda.
- Útil para la búsqueda de errores (*troubleshooting*) o planificar optimizaciones con preferencias de enrutamiento y *shards*.

```
# curl -XGET 10.0.0.10:9200/agenda/_search_shards?pretty
```

- En mi caso (al igual que debe pasar con los ejercicios) solo devuelve un shard, el primario. Nada que observar.

3. Index APIs. Create index API

- https://www.elastic.co/guide/en/elasticsearch/reference/current/indices-create-index.html
- Añade un índice al *cluster* de *Elasticsearch.*
- Se pueden especificar parámetros del índice como n.º de *shards* y réplicas; mapeo para los campos en el índice; alias del índice.

```
curl -XPUT '10.0.0.10:9200/index_name?pretty' -H 'Content-Type: application/json' -d'
{
    "settings": {
        "index": {
            "number_of_shards": 3,
            "number_of_replicas": 2
        }
    },
    "mappings": {
        "properties": {
          "field1": { "type": "text" }
        }
      }
}'
```

3. Index APIs. Delete index API

- https://www.elastic.co/guide/en/elasticsearch/reference/current/indices-delete-index.html
- Al eliminar un índice se borran sus documentos, *shards* y metadatos. No se eliminan los componentes de Kibana relacionados con el índice.
- Se puede borrar más de un índice, pasando una lista separada por comas o todos los índices con "`_all`" o "`*`". Para evitar el borrado total accidental, hay que configurar el fichero `elasticsearch.yml`

```
# curl -XDELETE 10.0.0.10:9200/new_index,nou_index?pretty
{
  "acknowledged" : true
}
```

3. Index APIs. Open/Close index API

- https://www.elastic.co/guide/en/elasticsearch/reference/current/indices-open-close.html
- Abre/Cierra (habilita/deshabilita) indices.
- Tener índices cerrados evita sobrecargar al *cluster* al evitar los accesos a los índices.

```
curl -XPOST '10.0.0.10:9200/agenda/_close?pretty'

curl -XPOST '10.0.0.10:9200/agenda/_open?pretty'
```

3. Index APIs. Get Mapping

- https://www.elastic.co/guide/en/elasticsearch/reference/current/indices-get-mapping.html
- Recupera las definiciones de mapeo para uno o más índices. Muestra la definición de los campos: *integer*, *string*, *date*... Es como los tipos de datos en C o Java.
- En el caso de los flujos de datos, la API recupera las asignaciones de los índices de respaldo del flujo.

```
curl -XGET '10.0.0.10:9200/twitter/_mapping?pretty'
```

- Las APIs que aceptan datos en breve serán desaprobadas.

3. Index APIs. Index stats API

- https://www.elastic.co/guide/en/elasticsearch/reference/current/indices-stats.html
- Devuelve las estadísticas de los índices.
- Es más bien una API de monitorización de índices: cantidad de documentos, tamaño del almacenamiento, uso de memoria de los segmentos.
- En el caso de los flujos de datos, la API recupera las estadísticas de los índices de respaldo del flujo.

```
curl -XGET '10.0.0.10:9200/_stats?pretty'
```

3. REST APIs. Actividades

- Hacer las actividades
  04.2_API_REST_indices_y_docs_Preguntas.odt

4. Dev Tools

- Herramientas para interactuar con los datos.
- **Consola**: Interactúa con la API REST de Elasticsearch, incluyendo el envío de solicitudes y la visualización de la documentación de la API.
- **Search Profiler**: Inspecciona y analiza las consultas de búsqueda.
- **Depurador Grok**: Crear y depurar patrones Grok antes de usarlos en procesamiento de datos.
- **Painless Lab**: En desarrollo.

4. Dev Tools

- Necesitamos entrar en Kibana y acceder a “Dev tools”
  - http://10.0.0.10:5601/
- Nos ahorra introducir en cada consulta la ip del servidor Elasticsearch.
- Nos facilita el acceso a las APIs conforme vamos escribiendo.
- Tenemos que pinchar el botón verde para ejecutar la consulta seleccionada. La llave inglesa ofrece más opciones.
- En el panel de resultados nos muestra una cabecera con sugerencias de configuración o parámetros utilizados como puede ser securizar nuestro servicio o el uso de parámetros deprecated.

4. Dev Tools. Console

Dev tools

```
GET _search
{
  "query": {
    "match_all": {}
  }
}

GET _cat/indices

GET _cluster/stats

GET /twitter/tweet/5

GET _search
{
  "query": {
    "term": {
      "user": "user1"
    }
  }
}
```

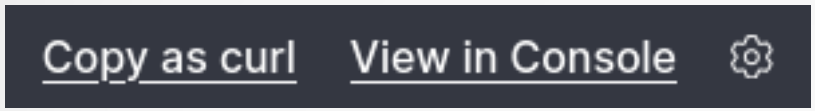
4. Dev Tools. Console

Dev tools

```
23  GET _search/template
24  {
25    "source": {
26      "query": {
27        "match": {
28          "{{my_field}}": "{{my_value}}"
29        }
30      },
31      "size": "{{my_size}}"
32    },
33    "params": {
34      "my_field": "user",
35      "my_value": "user1",
36      "my_size": 2
37    }
38  }
39
40  GET _template
41
```

4. Dev Tools. Console

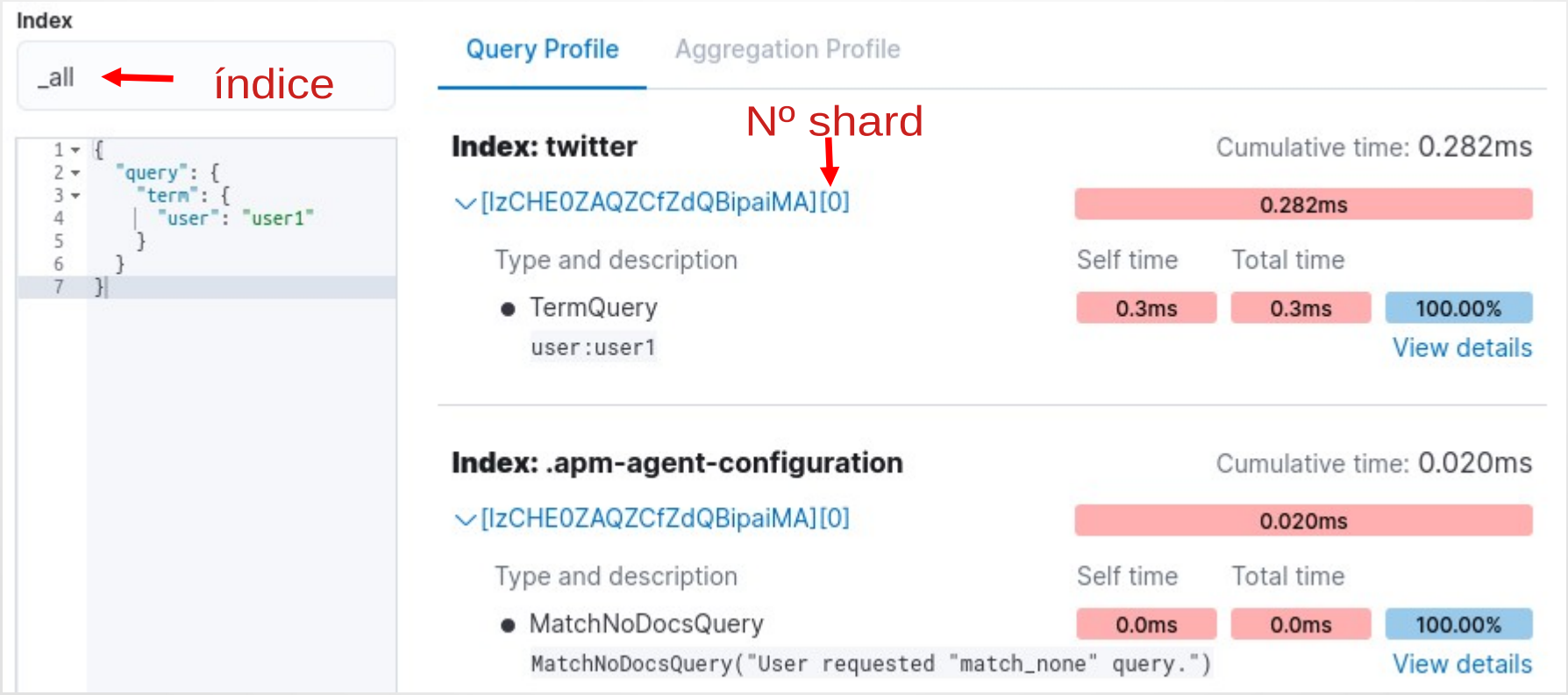
- Desde la Web, en la documentación de las API. Podemos configurar la ubicación de nuestro server.



4. Dev Tools. Search Profiler

Dev tools

- En Search Profiler podemos analizar las estadísticas de las consultas que hagamos.
- Vamos a buscar el *user user1* en todos los shards de todos los índices de *Elasticsearch*.



4. Dev Tools. Search Profiler

- En Search Profiler podemos analizar las estadísticas de las consultas que hagamos.
- Si damos al enlace de la derecha “View details” veremos las estadísticas de cada operación de la consulta.
- De este modo podemos ver en consultas complejas de más de 100 líneas donde se ataca a bases de datos de GiB o de TiB de tamaño que es lo que ralentiza la consulta y realizar consultas más eficientes.
- Ejemplo: Podemos ver que instrucciones tardan más y que resulta más eficiente p.e. capturar campos y realizar operaciones o realizar operaciones y discriminar después valores…

Webgraphy

- https://www.elastic.co/guide/en/elasticsearch/reference/current/rest-apis.html
- https://www.elastic.co/guide/en/kibana/current/devtools-kibana.html
- https://www.udemy.com/
- https://duckduckgo.com/ images
-
-