



## Documentación de la práctica Elasticsearch

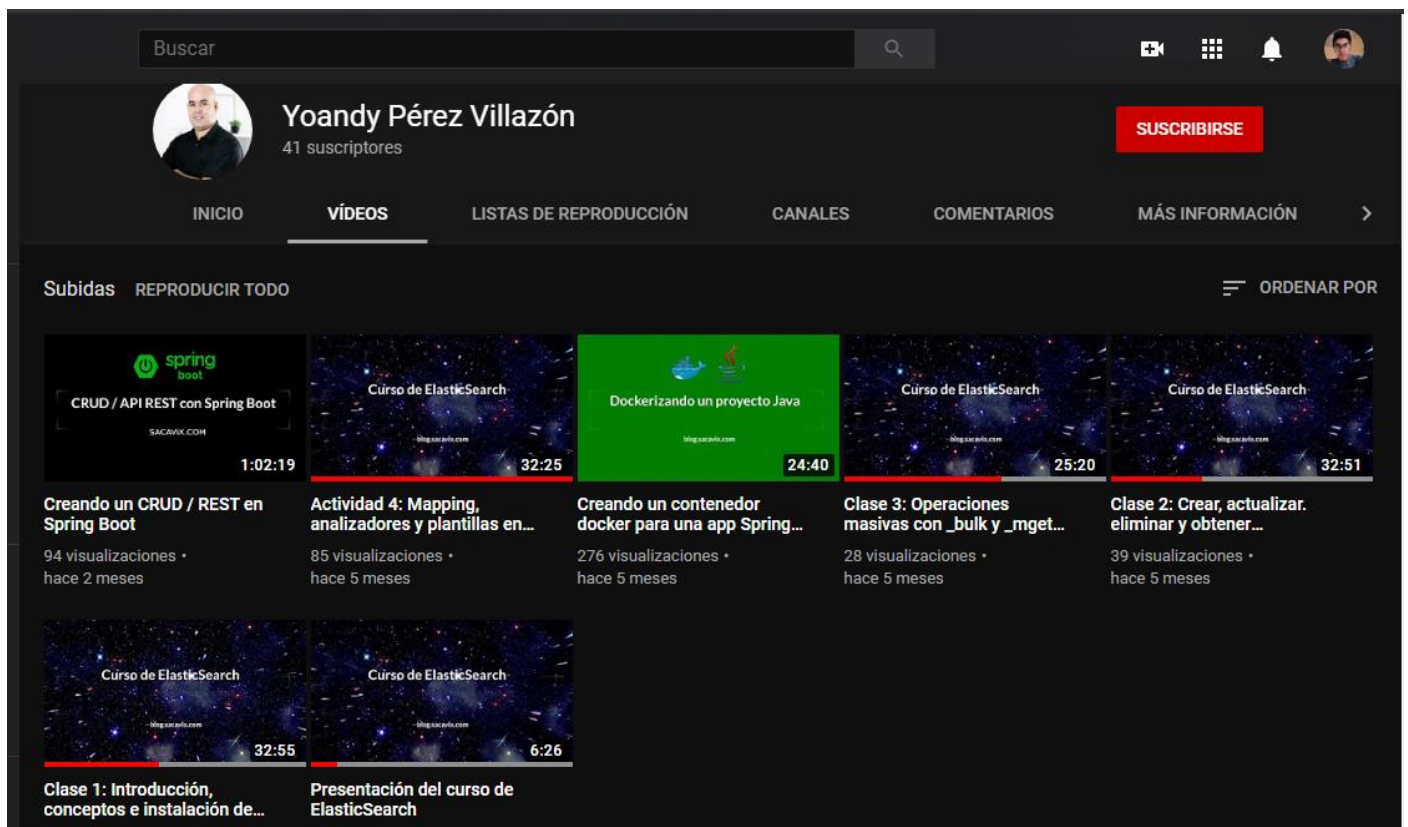
Realmente no conocía esta tecnología, por ende tuve que investigar sobre de que se trataba. Revisando algunos videos en YouTube y leyendo un poco la documentación me di cuenta del potencial que tiene elasticsearch al obtener información de manera masiva.

Por lo que tengo entendido, elasticsearch es un motor de búsquedas orientado a documentos json de código abierto, centrado en APIS basado en Apache Lucene.

Apache Lucene es una API de recuperación de información, desarrollado en Java, basado en índices invertidos. En la siguiente tabla se muestra un ejemplo de índices invertidos.

Indice – Texto	Texto	Indice	Posición
1. Cómo se llama	Cómo	1	1
		3	1
2. Se llama silla	Se	1	2
		2	1
3. Como esa silla	Llama	1	3
		2	2
	Esa	3	2
	Silla	2	3
		3	3

Investigando en YouTube me econtre con un canal que explica los primero pasos de elasticsearch. Con estos videos me base y guie para realizar la practica.



## 1. Conceptos

Cluster	Grupo de instancias de ES con el mismo nombre.
Nodo	Instancia de ES en sí.
Inidice	Colección de documentos. (Similar como las tablas de un BD).
Documento	Unidad básica de información. Se indexa y recupera (Registros).
Sahrd	Fragmento de un indice con parte de los documentos.
Replica	Copia de shard primario.

## 2. Verbos HHTP

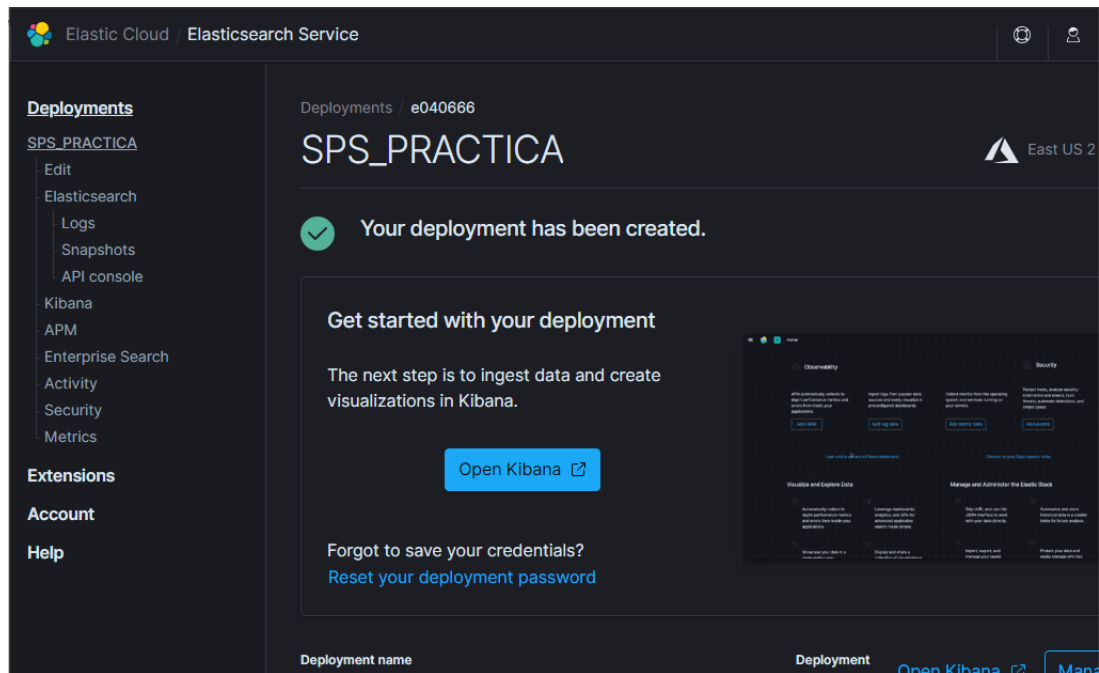
Son métodos de petición para indicar que acción se va a realizar.

POST	Crear	POST /nombre/_doc{...}
PUT	Actualizar	GET /nombre/_doc/<ID>
GET	Leer	PUT /nombre/_doc/<ID>{...}
DELETE	Eliminar	DELETE /nombre/_doc/<ID>

### 3. Preparando el entrono

Para realizar esta practica es necesario crear una cuenta. Yo la vincule con mi cuenta de Google para realizar le registro mucho más fácil.

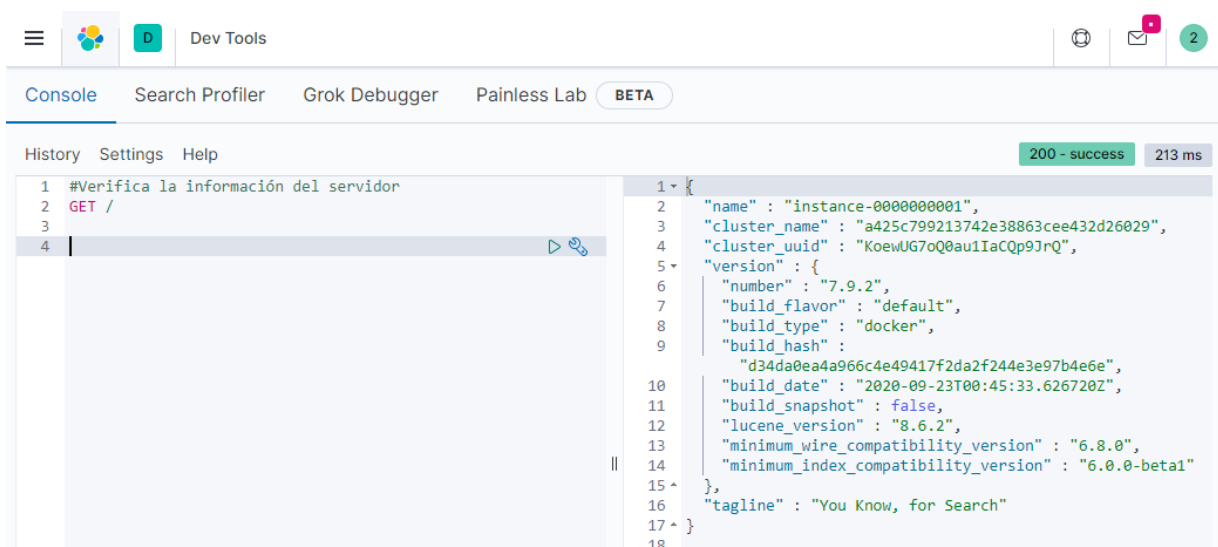
Después, para crear el deployment seleccione la opción de “Launch on elastic Cloud”, y seguí la configuración documentada.



### 4. Primeros comandos

Para iniciar con los comandos, me dirigí a la opción Dev Tools para iniciar.

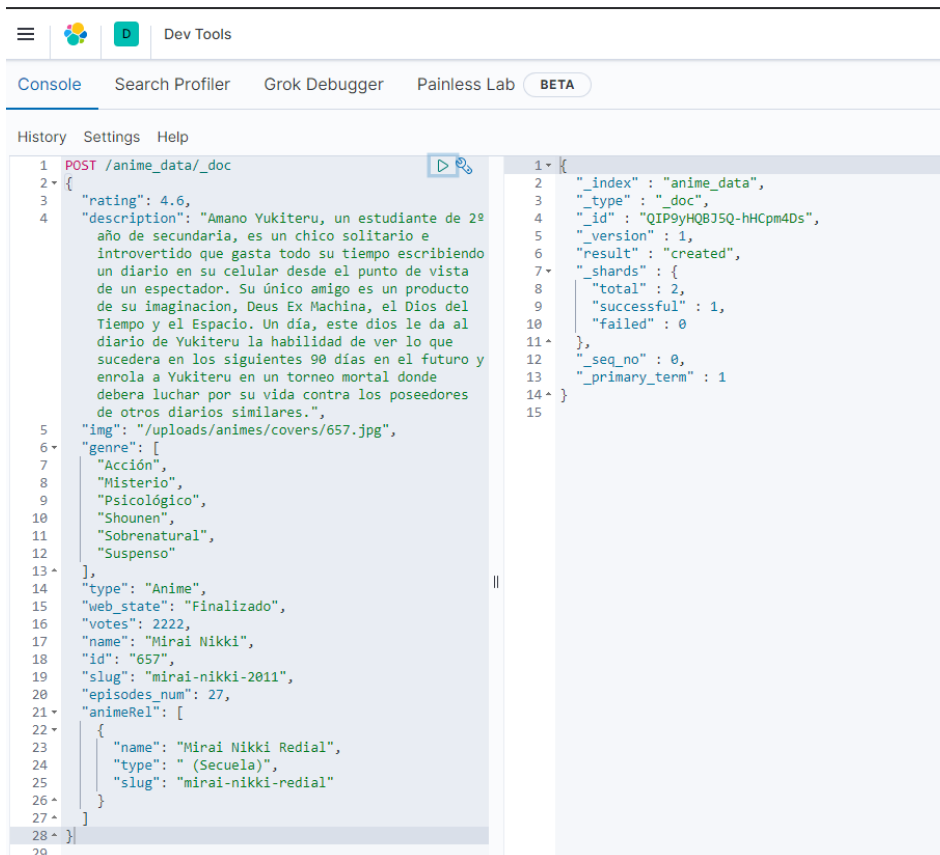
El primer comando que ejecute, es para ver la información del servidor.



## 5. Creando índice

Se me indica que realice un índice a partir de JSON proporcionado, por lo tanto debo de crear el índice junto con el documento, por así decirlo, junto con el primer registro.

Si todo esta correctamente, en la consola muestra los detalles del índice. En el campo llamado "result", muestra que se ha creado correctamente. A continuación se muestra como es la ejecución de este comando.

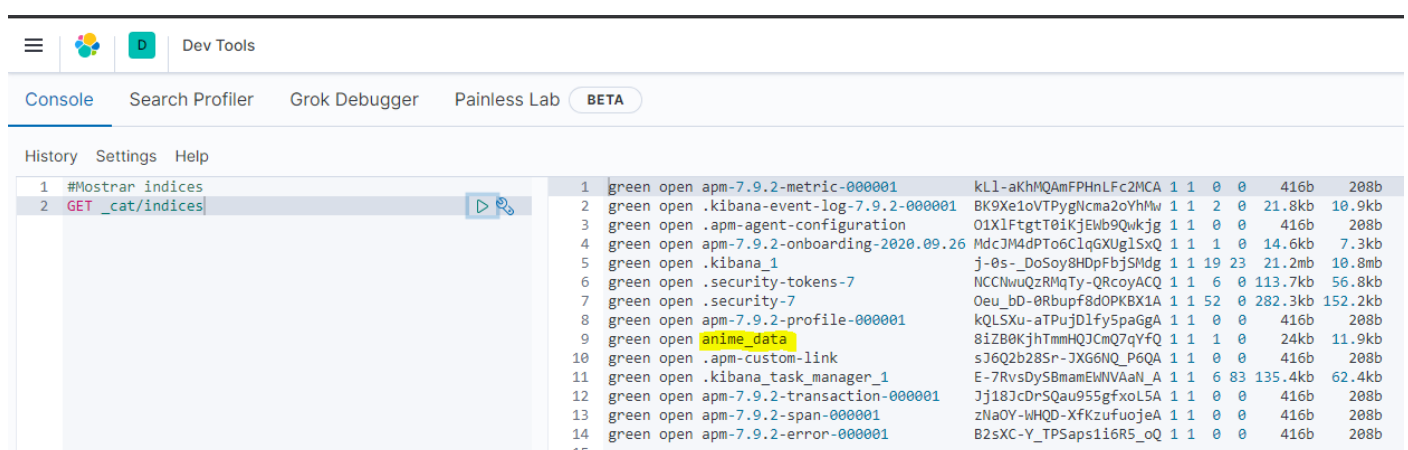


```
1 POST /anime_data/_doc
2 {
3   "rating": 4.6,
4   "description": "Amano Yukiteru, un estudiante de 2º
5     año de secundaria, es un chico solitario e
6     introvertido que gasta todo su tiempo escribiendo
7     un diario en su celular desde el punto de vista
8     de un espectador. Su único amigo es un producto
9     de su imaginación, Deus Ex Machina, el Dios del
10    Tiempo y el Espacio. Un día, este dios le da al
11    diario de Yukiteru la habilidad de ver lo que
12    sucedera en los siguientes 90 días en el futuro y
13    enrolla a Yukiteru en un torneo mortal donde
14    debera luchar por su vida contra los poseedores
15    de otros diarios similares.",
16   "img": "/uploads/animes/covers/657.jpg",
17   "genre": [
18     "Acción",
19     "Misterio",
20     "Psicológico",
21     "Shounen",
22     "Sobrenatural",
23     "Suspense"
24   ],
25   "type": "Anime",
26   "web_state": "Finalizado",
27   "votes": 2222,
28   "name": "Mirai Nikki",
29   "id": "657",
30   "slug": "mirai-nikki-2011",
31   "episodes_num": 27,
32   "animeRel": [
33     {
34       "name": "Mirai Nikki Redial",
35       "type": "(Secuela)",
36       "slug": "mirai-nikki-redial"
37     }
38   ]
39 }

1 {
2   "_index": "anime_data",
3   "_type": "_doc",
4   "_id": "QIP9yHQBJ5Q-hHCpm4Ds",
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 }
```

## 6. Mostrar índices

Para validar la creación del índice, ejecute el siguiente comando.



```
1 #Mostrar indices
2 GET _cat/indices

1 green open apm-7.9.2-metric-000001 kLL-aKhMQAMFPHnLFc2MCA 1 1 0 0 416b 208b
2 green open .kibana-event-log-7.9.2-000001 BK9Xe1oVTPygNcma2oYhMw 1 1 2 0 21.8kb 10.9kb
3 green open .apm-agent-configuration 01X1FtgtT0iKjEWb9Qwkjg 1 1 0 0 416b 208b
4 green open apm-7.9.2-onboarding-2020.09.26 MdcJM4dPT06ClqGXUg1SxQ 1 1 1 0 14.6kb 7.3kb
5 green open .kibana_1 j-0s-_DoSoy8HdpFbj5Mdg 1 1 19 23 21.2mb 10.8mb
6 green open .security-tokens-7 NCCNwUQzRMqTy-QRcoyACQ 1 1 6 0 113.7kb 56.8kb
7 green open .security-7 Oeu_bd-0Rbupf8d0PKBX1A 1 1 52 0 282.3kb 152.2kb
8 green open apm-7.9.2-profile-000001 kQLSXu-aTPuJd1fy5paGgA 1 1 0 0 416b 208b
9 green open anime_data 8iZB0KjhTmmHQJcmQ7qYfQ 1 1 1 0 24kb 11.9kb
10 green open .apm-custom-link sJ6Q2b28Ssr-JXG6NQ_P6QA 1 1 0 0 416b 208b
11 green open .kibana_task_manager_1 E-7RvsDySBmamEWNVAaA 1 1 6 83 135.4kb 62.4kb
12 green open apm-7.9.2-transaction-000001 Jj18JcDrSQau955gfoxLSA 1 1 0 0 416b 208b
13 green open apm-7.9.2-span-000001 zNaOY-wHQD-XfKzufuojeA 1 1 0 0 416b 208b
14 green open apm-7.9.2-error-000001 B2sXC-Y_TPSaps1i6R5_oQ 1 1 0 0 416b 208b
15
```

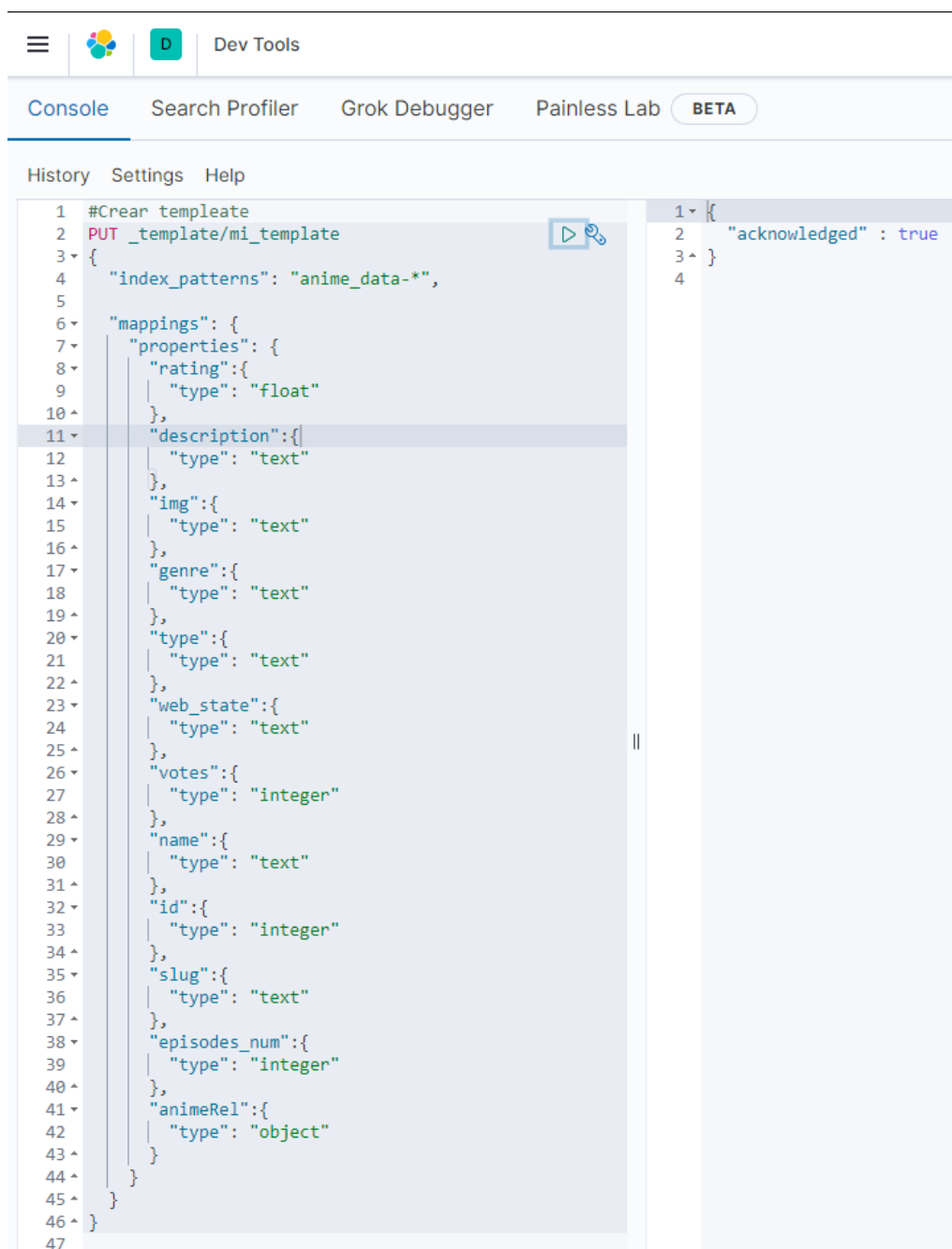
## 7. Crear template y mapping

El template es un plantilla donde se puede indicar la estructura del índice e indicar un patrón para los próximos documentos. En este caso el patrón es “anime\_data”, con “-\*”, estoy diciendo que todos los documentos que tengan ese patrón. Seguirán la estructura del template.

Junto con el template, está la opción del mapping, en donde se establecen los tipos de datos, de acuerdo a los campos establecidos en el JSON anterior.

Para entender mejor sobre los tipos de datos, viste la documentación de ES.

<https://www.elastic.co/guide/en/elasticsearch/reference/current/mapping-types.html>



The screenshot shows a web IDE interface with a top bar containing a menu icon, a logo, a 'D' icon, and the text 'Dev Tools'. Below this is a navigation bar with tabs: 'Console' (selected), 'Search Profiler', 'Grok Debugger', 'Painless Lab', and a 'BETA' badge. The main area has a sub-header with 'History', 'Settings', and 'Help'. The 'Console' tab is active, displaying a REST client interface. On the left, a PUT request is shown with the following JSON body:




```
1 #Crear template
2 PUT _template/mi_template
3 {
4   "index_patterns": "anime_data-*",
5
6   "mappings": {
7     "properties": {
8       "rating": {
9         "type": "float"
10      },
11     "description": {
12       "type": "text"
13     },
14     "img": {
15       "type": "text"
16     },
17     "genre": {
18       "type": "text"
19     },
20     "type": {
21       "type": "text"
22     },
23     "web_state": {
24       "type": "text"
25     },
26     "votes": {
27       "type": "integer"
28     },
29     "name": {
30       "type": "text"
31     },
32     "id": {
33       "type": "integer"
34     },
35     "slug": {
36       "type": "text"
37     },
38     "episodes_num": {
39       "type": "integer"
40     },
41     "animeRel": {
42       "type": "object"
43     }
44   }
45 }
46 }
```

On the right, the response is shown as a JSON object:

```
1 {
2   "acknowledged" : true
3 }
```

## 8. API BULK

La forma correcta de ingresar datos a un índice de manera masiva, se realiza con el API bulk. Con esto puedo ingresar todos los datos que están en el archivo Anime.json. Realmente es un archivo de tipo ndjson para indicar a que patrón va dirigido los registros.



Dev Tools

Console

Search Profiler

Grok Debugger

Painless Lab

BETA

History

Settings

Help

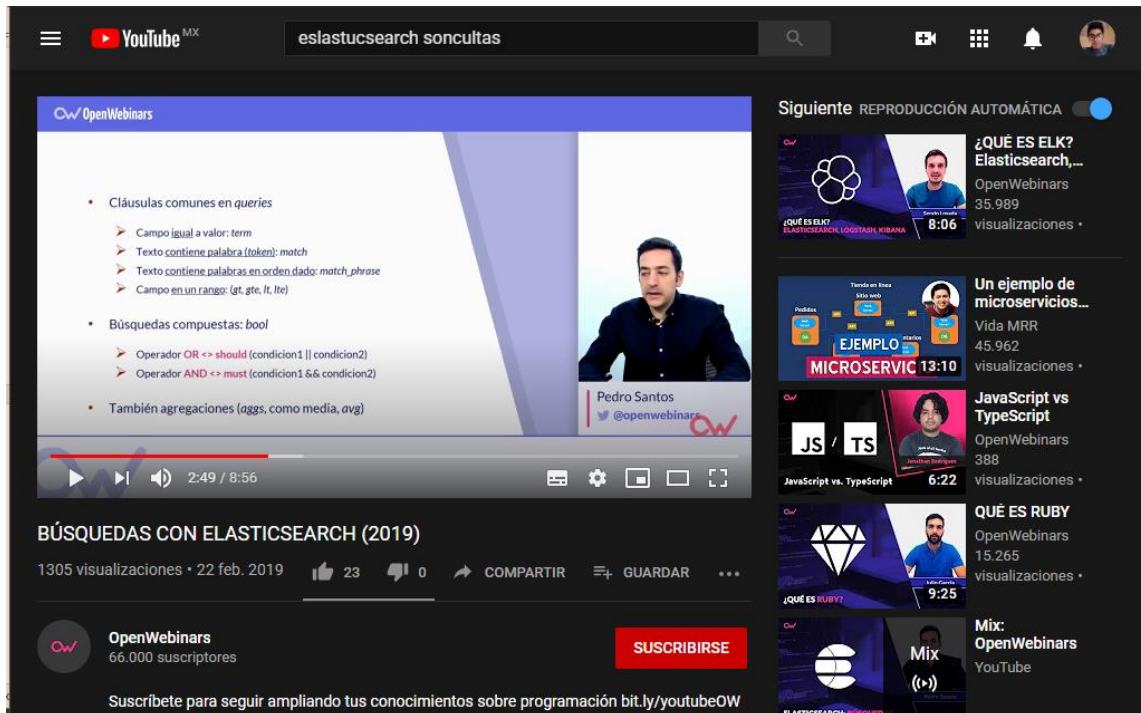
```
1 #Operación masiva
2 POST _bulk
3 {"index": {"_index": "anime_data"}}
4 {"rating": "4.5", "description": "La historia se
centra en Hotaru, una chica que se pierde en un
bosque en el que se dice habitan muchos
fantasmas. Ante ella se le aparece Gin, un
joven muchacho que le embelesa con una primera
mirada pero al cual ella no se atreve ni a
tocar por miedo a que desaparezca.\n", "img": "
/uploads/animes/covers/745.jpg", "genre":
["Drama", "Romance", "Shoujo", "Sobrenatural"],
"type": "Pel\u00e9cula", "web_state":
"Finalizado", "votes": 1329, "name": "Hotarubi
no Mori e", "id": "745", "slug": "hotarubi-no
-mori-e", "episodes_num": 1, "animeRel": []}
5 {"index": {"_index": "anime_data"}}
6 {"rating": "3.5", "description": "En un futuro en
el que una anomal\u00eda del Sol ha destrozado
la atm\u00f3sfera de la Tierra, con la
consiguiente devastaci\u00f3n en el ecosistema.
El deterioro del ADN ha reducido la tasa de
natalidad de manera dram\u00e1tica, lo que ha
forzado a los gobernantes a crear unos clones
llamados IC (Ideal Children), as\u00ed como un
gobierno para ellos. El protagonista de la
serie es Sam Coin, un pirata de la arena? que
comercia en el desierto mientras busca un
misterioso objeto gigantesco llamado Ozuma, que
en su d\u00eda caus\u00f3 da\u00f1o a su
hermano. En una de sus correr\u00edas rescata a
una muchacha llamada Maya que est\u00e1 siendo
perseguida por un destructor de la arena
propiedad del ej\u00e9rcito IC ?Shishiasu?. Sam
se la lleva a la nave de los piratas de la
arena, Barudonosu, situada en el Puerto de
Oaaze. Sin embargo, el destructor de la arena
de los IC asedia a la Barudanosu.", "img": "
/uploads/animes/covers/749.jpg", "genre":
["Acci\u00f3n", "Ciencia Ficc\u00eda"], "type":
"Anime", "web_state": "Finalizado", "votes":
14, "name": "Ozuma", "id": "749", "slug":
"ozuma", "episodes_num": 6, "animeRel": []}
7 {"index": {"_index": "anime_data"}}
8 {"rating": "4.7", "description": "Relata la
historia original de lo que sucede despu\u00e9s
del arco de Ragnarok.", "img": "/uploads/animes
/covers/751.jpg", "genre": ["Acci\u00f3n",
```

```
1 {
2   "took" : 63,
3   "errors" : false,
4   "items" : [
5     {
6       "index" : {
7         "_index" : "anime_data",
8         "_type" : "_doc",
9         "_id" : "m6wByXQBZ-UEj5SSpqtv",
10        "_version" : 1,
11        "result" : "created",
12        "_shards" : {
13          "total" : 2,
14          "successful" : 2,
15          "failed" : 0
16        },
17        "_seq_no" : 1,
18        "_primary_term" : 1,
19        "status" : 201
20      },
21    },
22    {
23      "index" : {
24        "_index" : "anime_data",
25        "_type" : "_doc",
26        "_id" : "nKwByXQBZ-UEj5SSpqtv",
27        "_version" : 1,
28        "result" : "created",
29        "_shards" : {
30          "total" : 2,
31          "successful" : 2,
32          "failed" : 0
33        },
34        "_seq_no" : 2,
35        "_primary_term" : 1,
36        "status" : 201
37      },
38    },
39    {
40      "index" : {
41        "_index" : "anime_data",
42        "_type" : "_doc",
43        "_id" : "nawByXQBZ-UEj5SSpqtv",
44        "_version" : 1,
45        "result" : "created",
46        "_shards" : {
47          "total" : 2,
48          "successful" : 2,
```

## 9. API SEARCH

El API SEARCH, está destinado para realizar consultas dentro de un índice.

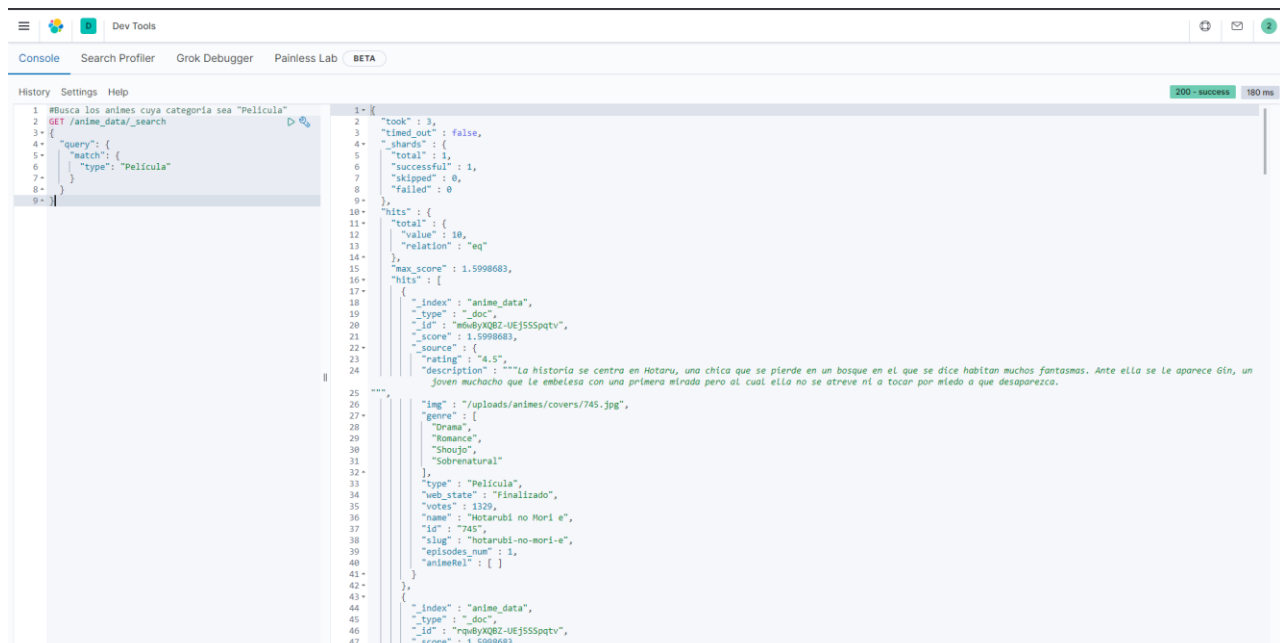
Para esta actividad, tuve que investigar tanto en su documentación como en videos de YouTube. El video de OpenWebinars, me ayudo a comprender sobre el tema.



## 10. Primer consulta

La primer consulta se me pide que busque los animes que tengan la categoría de “Película”.

Con ayuda del video anterior, me base para realizar esta consulta.

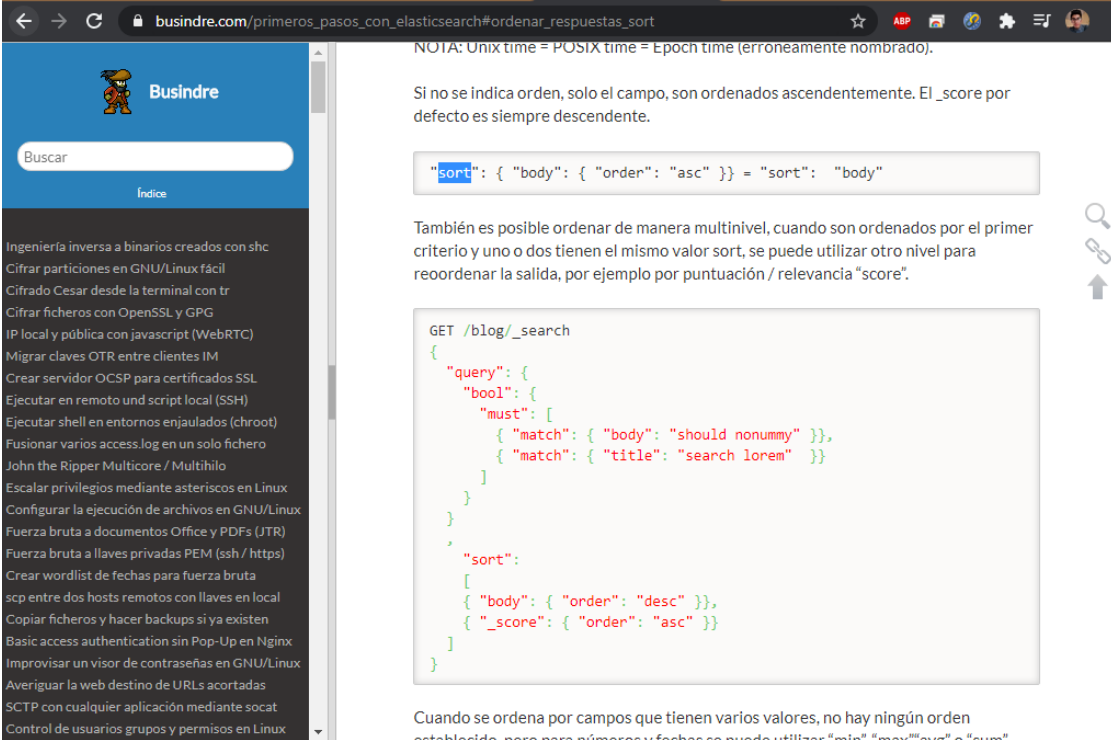




## 11. Segunda consulta

En esta parte contiene varias subconsultas, lo primero que hice fue realizar las consultas de manera individual, una por una, para tener un orden y no confundirme fácilmente.

De igual forma me base del video anterior e investigue la forma de como ordenar los datos como se me pide. Para ello en la siguiente pagina se me da un ejemplo del sorteo de datos y lo adecúe a la consulta que se me indica.



NOTA: Unix time = POSIX time = Epoch time (erroneamente nombrado).

Si no se indica orden, solo el campo, son ordenados ascendentemente. El `_score` por defecto es siempre descendente.

```
"sort": { "body": { "order": "asc" } } = "sort": "body"
```

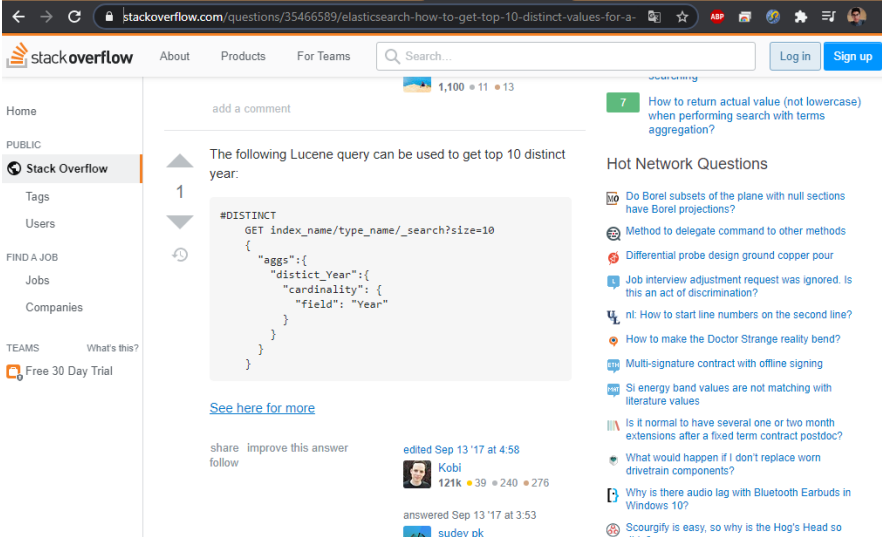
También es posible ordenar de manera multinivel, cuando son ordenados por el primer criterio y uno o dos tienen el mismo valor sort, se puede utilizar otro nivel para reordenar la salida, por ejemplo por puntuación / relevancia "score".

```
GET /blog/_search
{
  "query": {
    "bool": {
      "must": [
        { "match": { "body": "should nonummy" } },
        { "match": { "title": "search lorem" } }
      ]
    }
  },
  "sort": [
    { "body": { "order": "desc" } },
    { "_score": { "order": "asc" } }
  ]
}
```

Cuando se ordena por campos que tienen varios valores, no hay ningún orden establecido, pero para números y fechas se puede utilizar "min" "max" "sum" o "cum"

[https://www.busindre.com/primeros\\_pasos\\_con\\_elasticsearch#ordenar\\_respuestas\\_sort](https://www.busindre.com/primeros_pasos_con_elasticsearch#ordenar_respuestas_sort)

De igual forma, investigue en internet como realizar un TOP para obtener una cantidad de datos, lo encontré en stackoverflow.



add a comment

1

The following Lucene query can be used to get top 10 distinct year:

```
#DISTINCT
GET index_name/type_name/_search?size=10
{
  "aggs": {
    "distinct_year": {
      "cardinality": {
        "field": "Year"
      }
    }
  }
}
```

[See here for more](#)

share improve this answer follow

edited Sep 13 '17 at 4:58  
Kobi  
121k ● 39 ● 240 ● 276

answered Sep 13 '17 at 3:53  
sudev pk

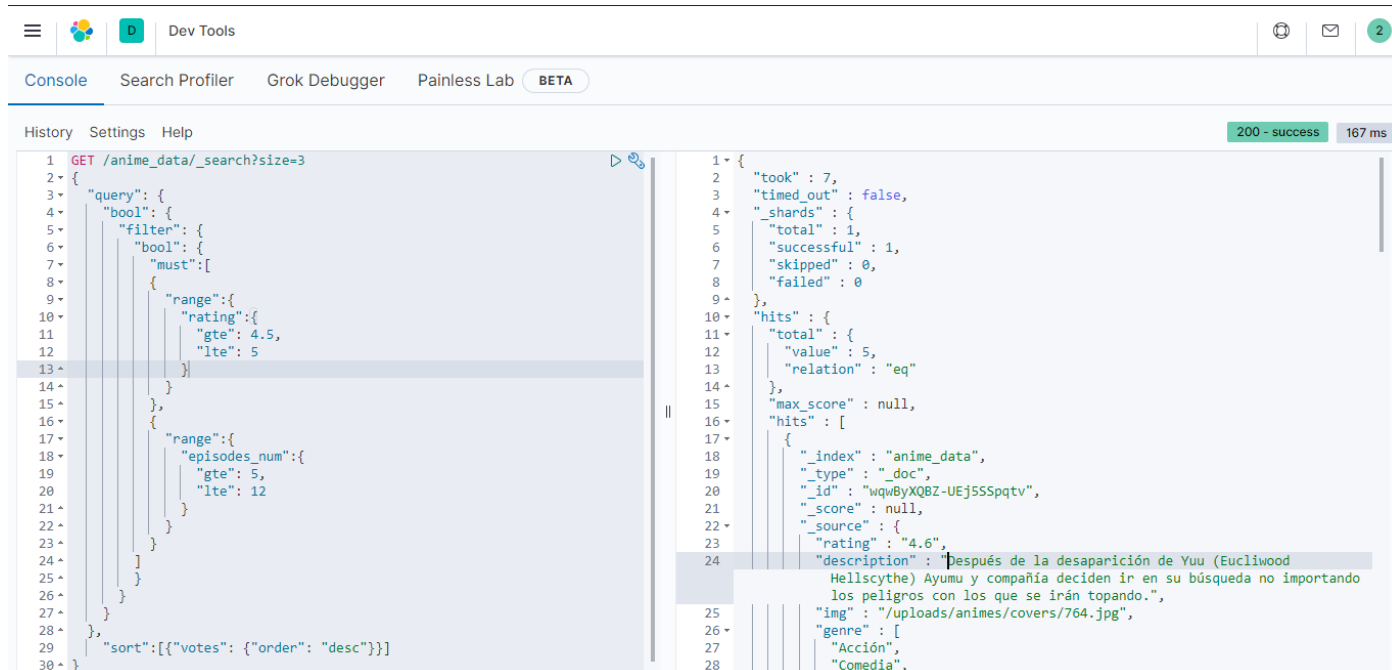
7 How to return actual value (not lowercase) when performing search with terms aggregation?

Hot Network Questions

- Do Borel subsets of the plane with null sections have Borel projections?
- Method to delegate command to other methods
- Differential probe design ground copper pour
- Job interview adjustment request was ignored. Is this an act of discrimination?
- How to start line numbers on the second line?
- How to make the Doctor Strange reality bend?
- Multi-signature contract with offline signing
- Si energy band values are not matching with literature values
- Is it normal to have several one or two month extensions after a fixed term contract postdoc?
- What would happen if I don't replace worn drivetrain components?
- Why is there audio lag with Bluetooth Earbuds in Windows 10?
- Scourgify is easy, so why is the Hog's Head so dirty?



Uniendo las piezas, pude realizar la consulta con éxito.



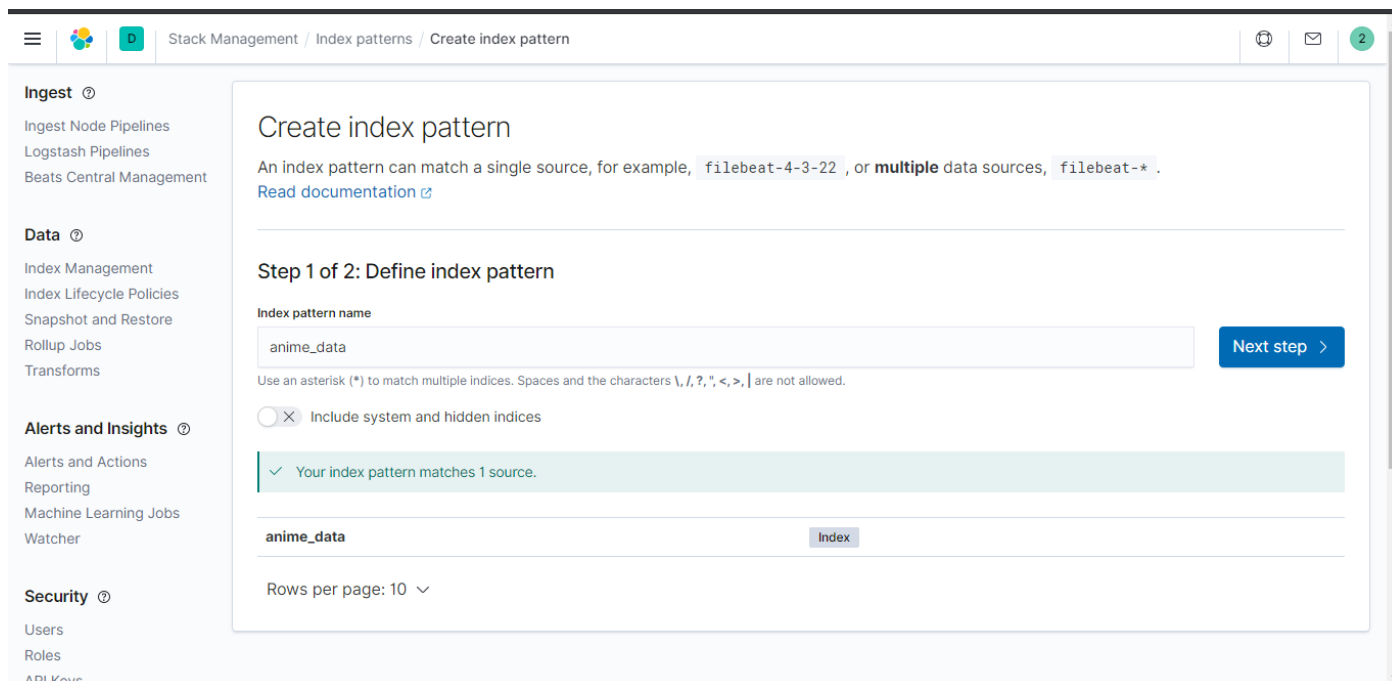
```
1 GET /anime_data/_search?size=3
2 {
3   "query": {
4     "bool": {
5       "filter": {
6         "bool": {
7           "must": [
8             {
9               "range": {
10                "rating": {
11                  "gte": 4.5,
12                  "lte": 5
13                }
14              }
15            },
16            {
17              "range": {
18                "episodes_num": {
19                  "gte": 5,
20                  "lte": 12
21                }
22              }
23            }
24          ]
25        }
26      }
27    },
28    "sort": [{ "votes": { "order": "desc" } }]
29  }
30 }
```

```
1 {
2   "took": 7,
3   "timed_out": false,
4   "_shards": {
5     "total": 1,
6     "successful": 1,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": {
12      "value": 5,
13      "relation": "eq"
14    },
15    "max_score": null,
16    "hits": [
17      {
18        "_index": "anime_data",
19        "_type": "_doc",
20        "_id": "wqwByXQBZ-UEj5SSpqtvt",
21        "_score": null,
22        "_source": {
23          "rating": "4.6",
24          "description": "Después de la desaparición de Yuu (Eucliwood Hellscythe) Ayumu y compañía deciden ir en su búsqueda no importando los peligros con los que se irán topando.",
25          "img": "/uploads/animes/covers/764.jpg",
26          "genre": [
27            "Acción",
28            "Comedia",

```

## 12. Index Patterns

Para mostrar la información de mi índice con visualizaciones, es necesario crear un index pattern. Por lo tanto seguí las instrucciones para crearlo con los datos del índice de anime\_data.



Stack Management / Index patterns / Create index pattern

**Ingest** ⓘ

- Ingest Node Pipelines
- Logstash Pipelines
- Beats Central Management

**Data** ⓘ

- Index Management
- Index Lifecycle Policies
- Snapshot and Restore
- Rollup Jobs
- Transforms

**Alerts and Insights** ⓘ

- Alerts and Actions
- Reporting
- Machine Learning Jobs
- Watcher

**Security** ⓘ

- Users
- Roles
- API Keys

### Create index pattern

An index pattern can match a single source, for example, `filebeat-4-3-22`, or **multiple** data sources, `filebeat-*`.  
[Read documentation](#)

#### Step 1 of 2: Define index pattern

Index pattern name

Use an asterisk (\*) to match multiple indices. Spaces and the characters \, /, ?, ", <, >, | are not allowed.

☐ Include system and hidden indices

✓ Your index pattern matches 1 source.

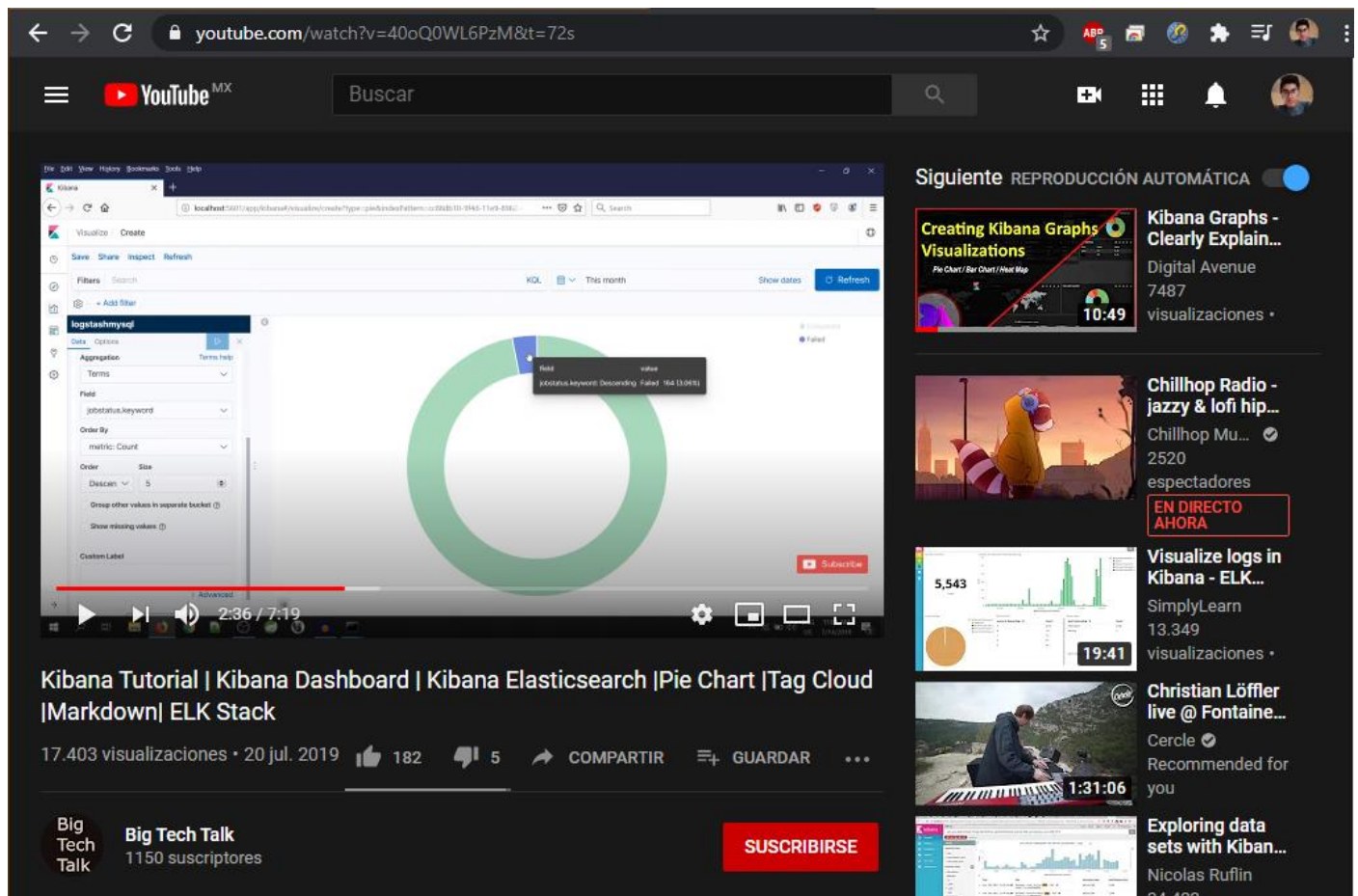
anime_data	Index
------------	-------

Rows per page: 10 ▾

### 13. Visualizaciones

Intentando crear las visualizaciones por mi mismo, me di cuenta que era muy complicado, ya que no entendía bien como se debia seleccionar los datos que queria mostrar y mejor opte por investigar y ver videos tutoriales.

El video de un estadounidense me ayudo muchísimo a entender como es que se realizaban y como se llenaban de información.

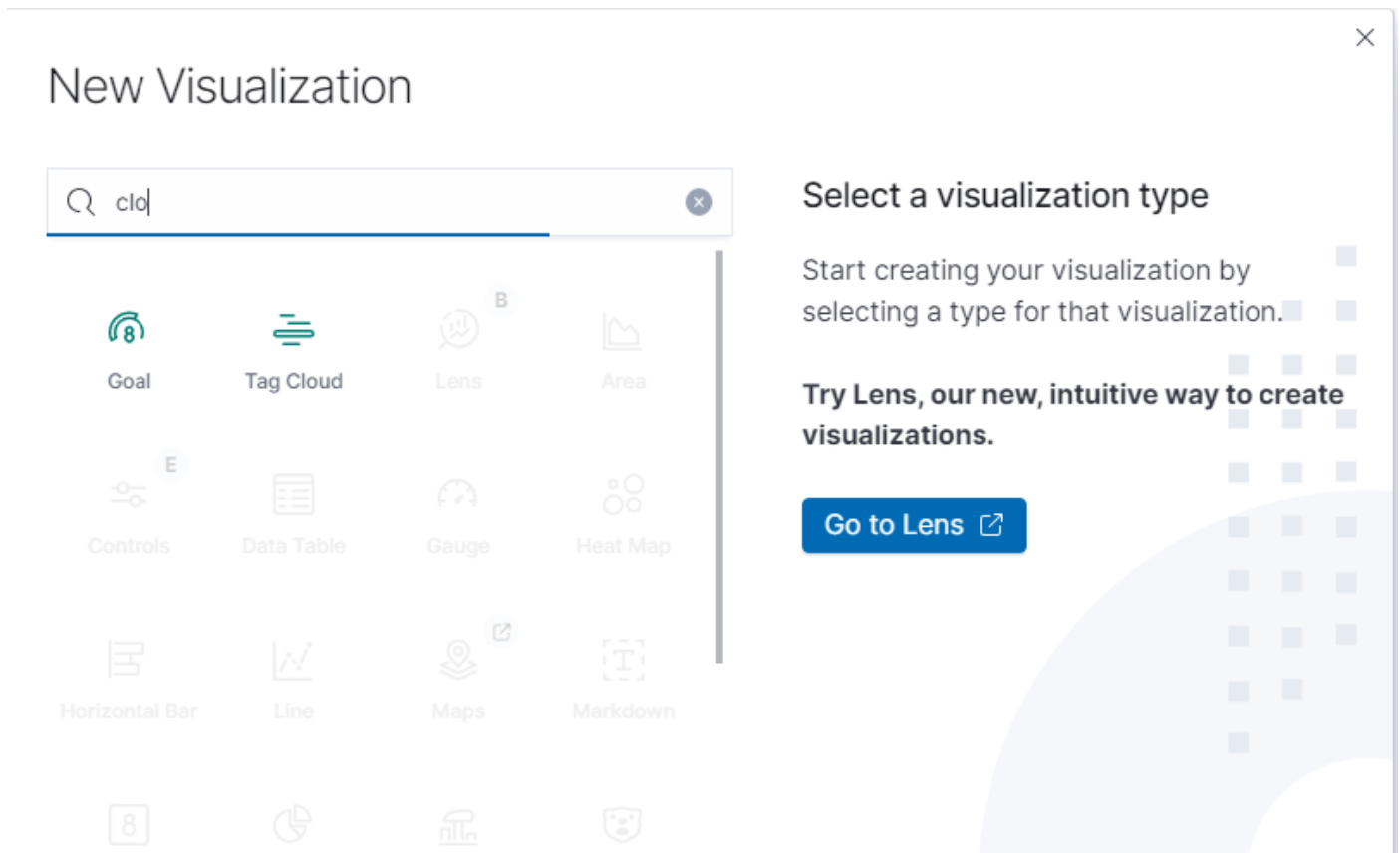
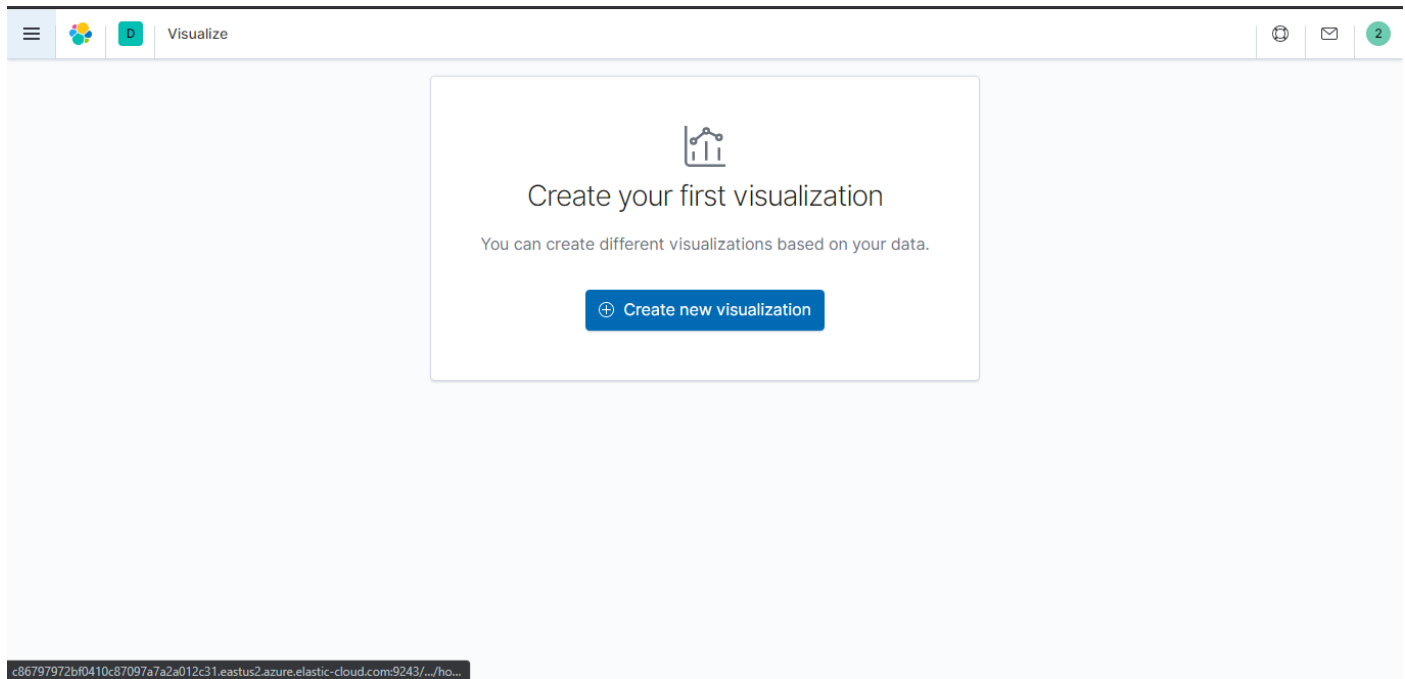


### 14. Visualizacion de nube

Para crear esta visualización, busque y seleccione el item. Posteriormente, le indique el indice con el que quiero trabajar y finalmente me mando a el área de trabajo.

En el área de trabajo me dirigí a la opción de “Data“, y seleccioné la agregación de tipo condición, para que obtenga todos los géneros de los animes. De esta forma pude crear la primera visualización.

Finalmente guarde la visualización para utilizarla en algún futuro.



# New Tag Cloud / Choose a source

Sort ▾

Types 2 ▾

 anime\_data

Visualize / Create

Save

Share Inspect

genre : \*

KQL

Off

Refresh

+ Add filter



anime\_data

Data Options

Aggregation

Terms [Terms help](#)

Field

genre.keyword

Order by

Metric: Count

Order

Descending

Size

30

☐ Group other values in separate bucket

☐ Show missing values

Discard

Update

## Save visualization

Title

Géneros

Description

Contiene todos los géneros del índice

Cancel

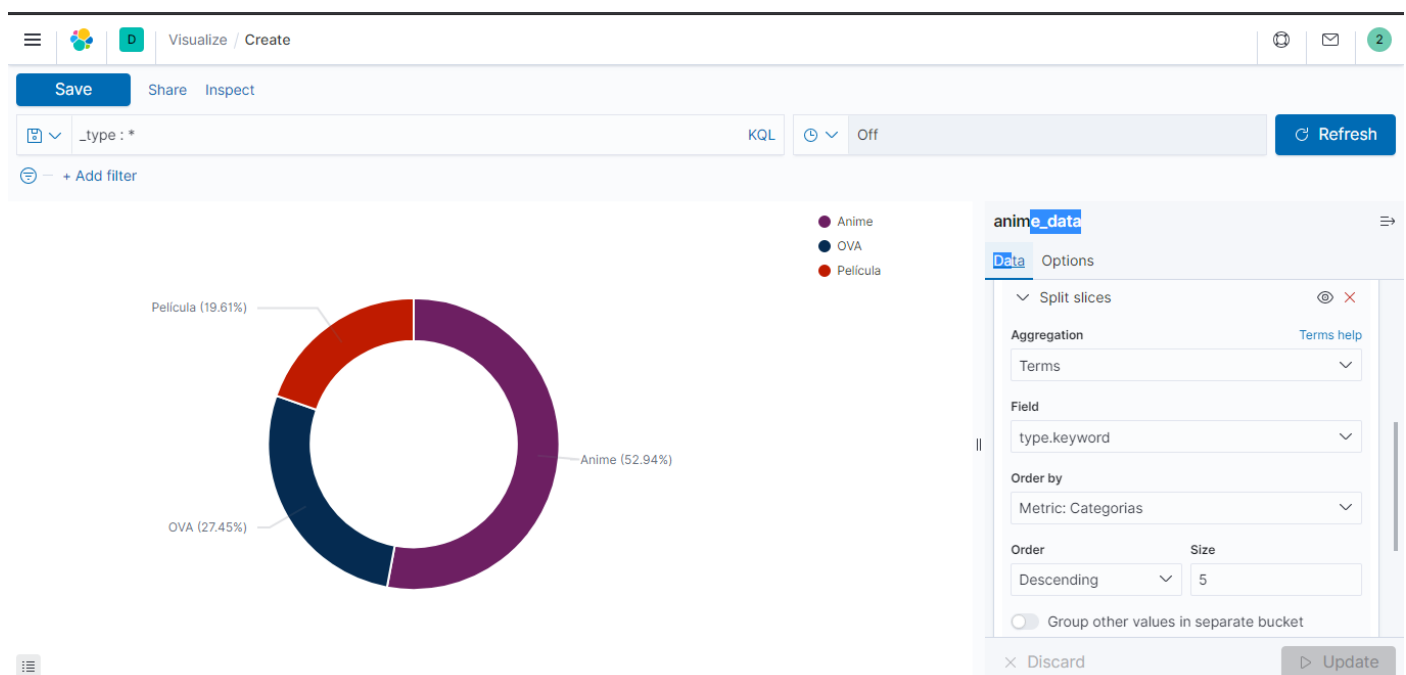
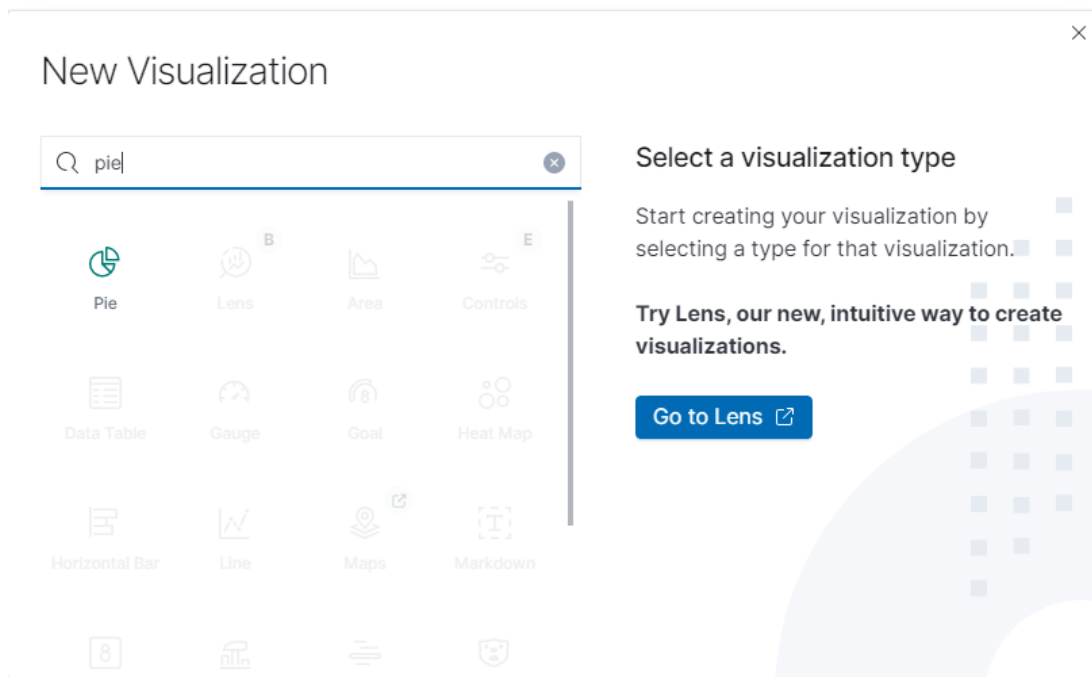
Save

## 15. Vista de PIE

Para realizar esta vista, repetí los pasos anteriores, cambiando el ítem a una gráfica de PIE.

Con ella realice la siguiente condición de mostrar las categorías que existen, para mostrarlas en la gráfica.

Para que se vean las categorías en la gráfica, me dirigí a las opciones y active el switch para que se visualizaran correctamente.



**Labels settings**

☒ Show labels

☒ Show top level only

☒ Show values

Truncated

×

## Save visualization

**Title**

Categorías

**Description**

Categorías de los animes

Cancel

Save

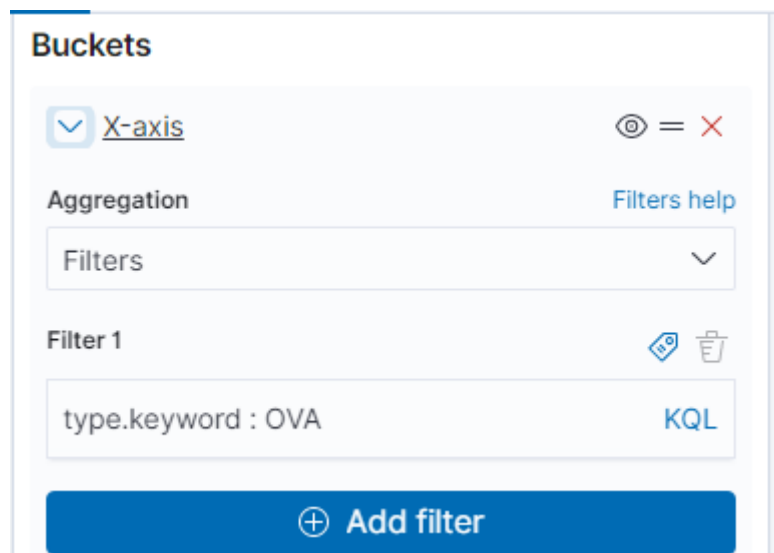
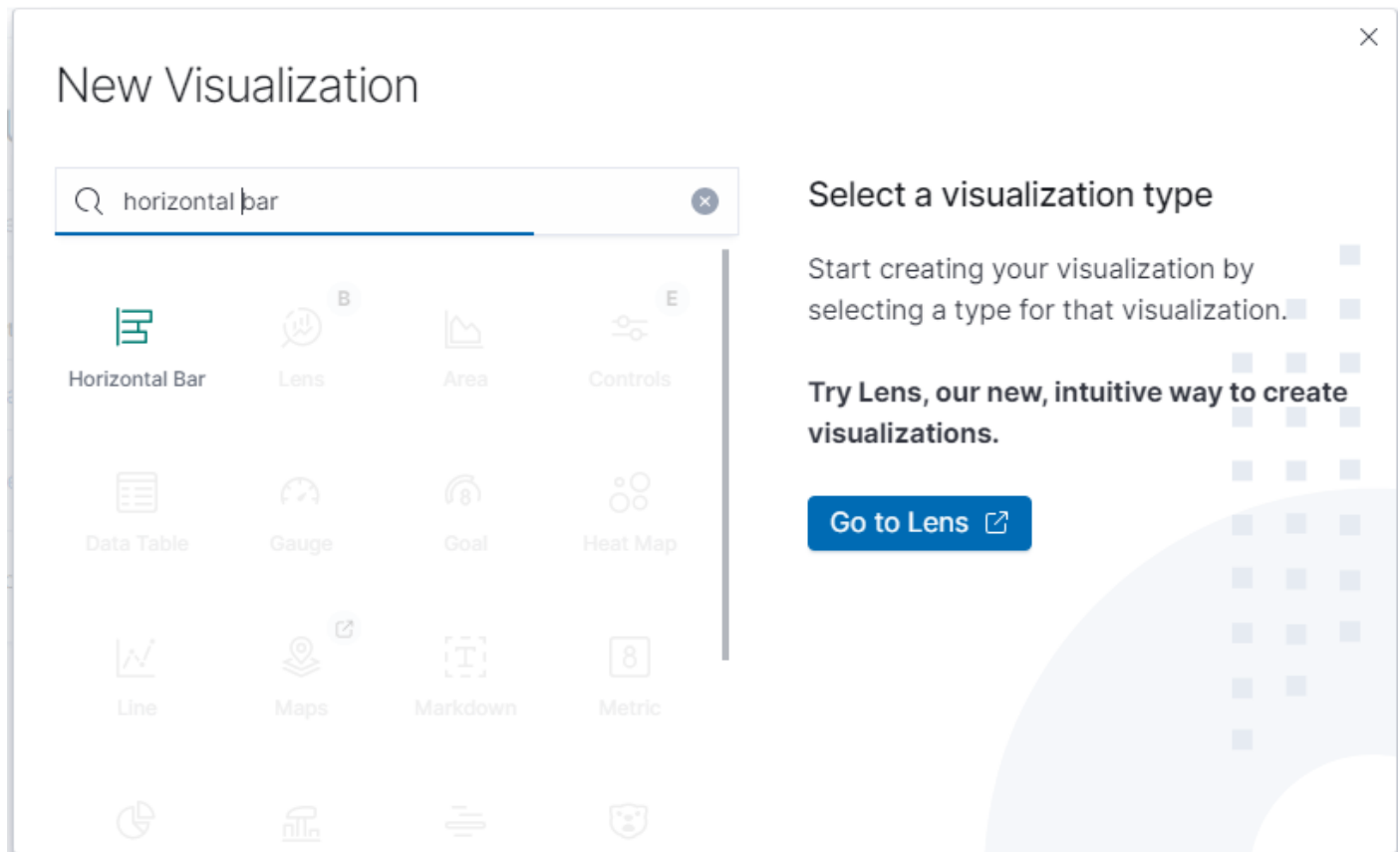
## 16. Vista de Barras

Esta vista se me hizo muy difícil, ya que no lograba realizar las consultas que se me pedia, y por poco decido no hacerla. Afortunadamente viendo el video tutorial de nuevo y observando como estaba estructurada la grafica en el archivo PDF, pude terminar la vista.

En la opción “Buckets”, para el eje x, seleccione la ag agregación reacción de tipo condición para que busque los animes con la ca categoría tegoria OVA.

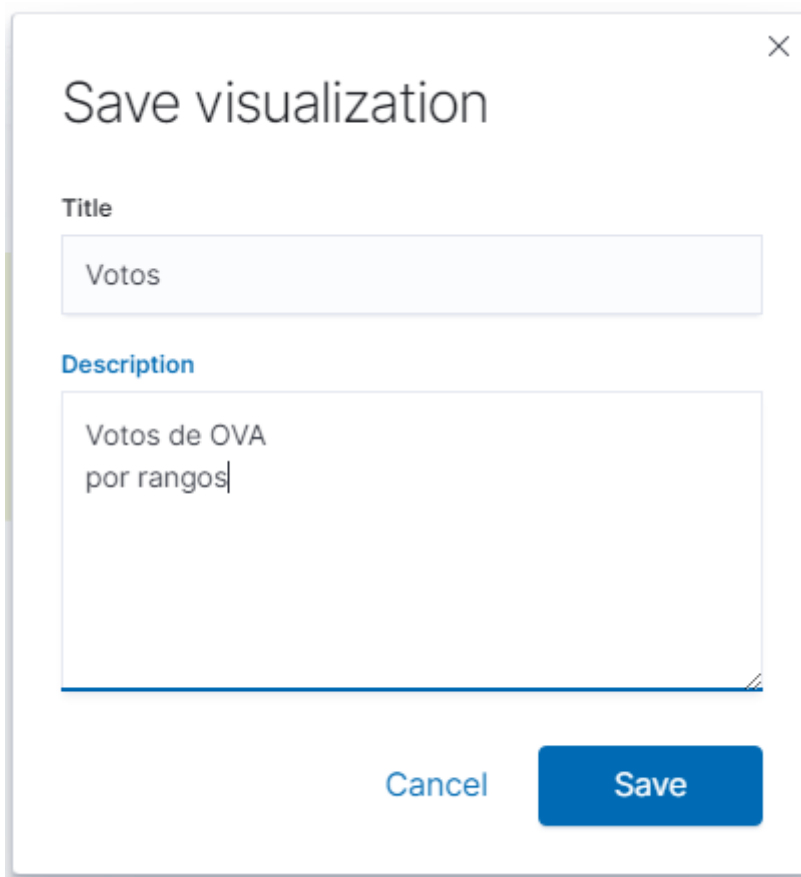
De igual forma, para los rangos de votos, selecciones un “Split\_series”, con una sub agregación de tipo rango, estableciendo los rangos solicitados.

Y por último, con la opción "Panel Settings", active el switch para que se mostraran los valores en la grafica.









A dialog box titled "Save visualization" with a close button (X) in the top right corner. It contains two input fields: "Title" with the text "Votos" and "Description" with the text "Votos de OVA por rangos". At the bottom, there are two buttons: "Cancel" and "Save".

Save visualization

Title

Votos

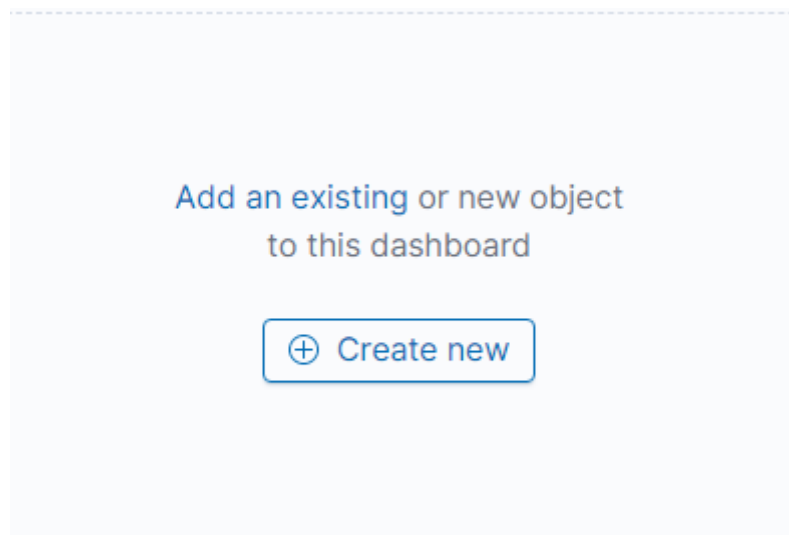
Description

Votos de OVA  
por rangos

Cancel Save

## 17. Dashboards

Por último, para crear el tablero con las visualizaciones guardadas, me dirigí a la opción de Dashboards y seleccioné la opción de "Add an existing", para agregar las visualizaciones creadas anteriormente.



A dialog box with a dashed border. It contains the text "Add an existing or new object to this dashboard" and a button labeled "Create new" with a plus icon.

Add an existing or new object  
to this dashboard

+ Create new

# Add panels



Sort ▾

Types

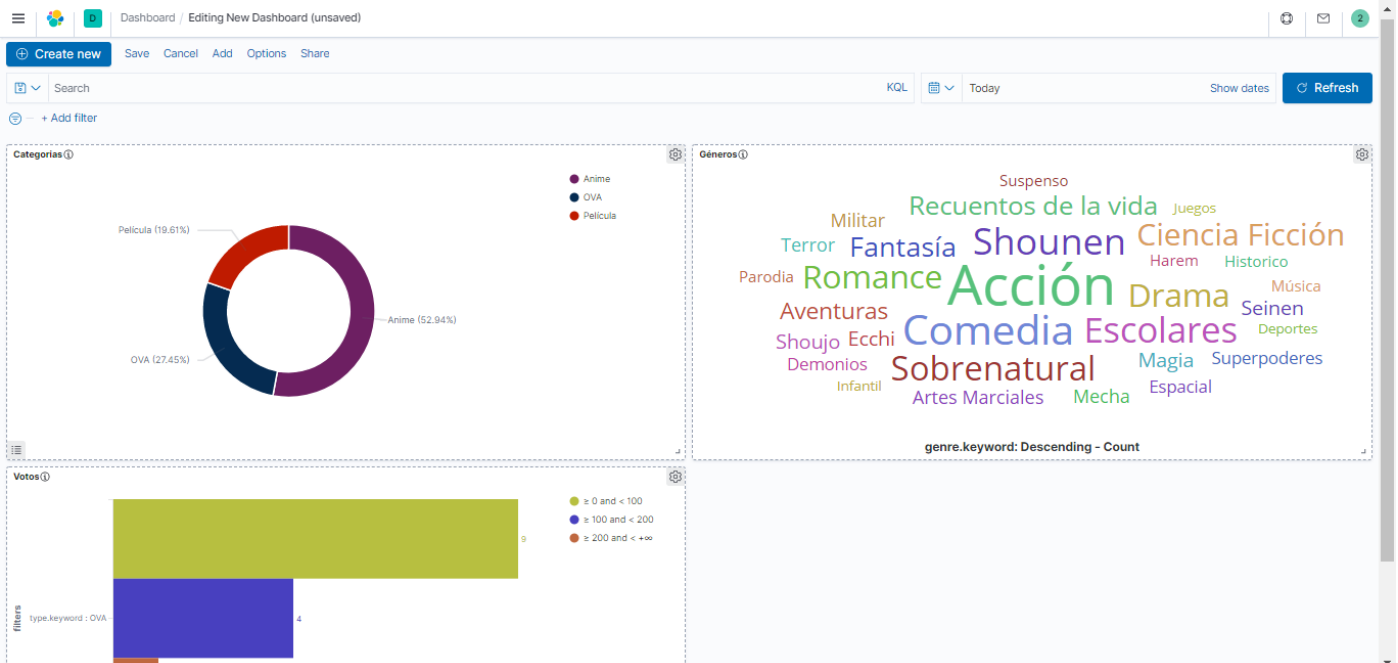
4 ▾

[+ Create new](#)

 Categorías

 Géneros

 Votos



## Save dashboard ×

Title

Mi tablero

Description

Mi tablero con las visualizaciones creadas

☐ Store time with dashboard

This changes the time filter to the currently selected time each time this dashboard is loaded.

Cancel

Save