



Practica de Elasticsearch

**Jazmín Susana De la Cruz
García**

SPSolutions

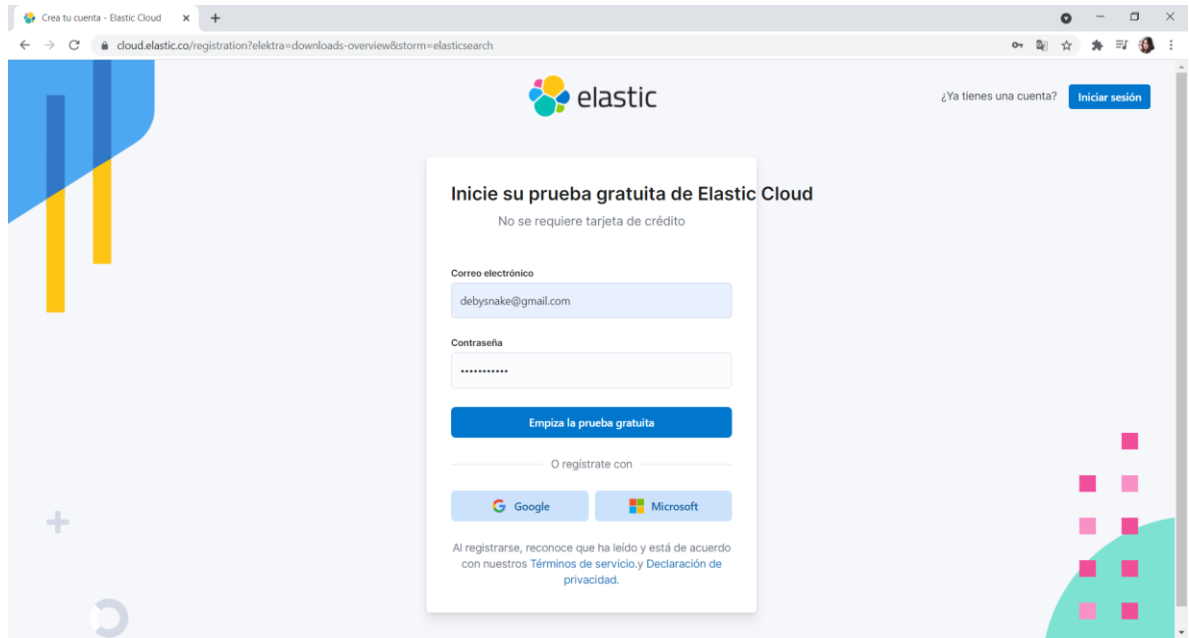


1. Contenido

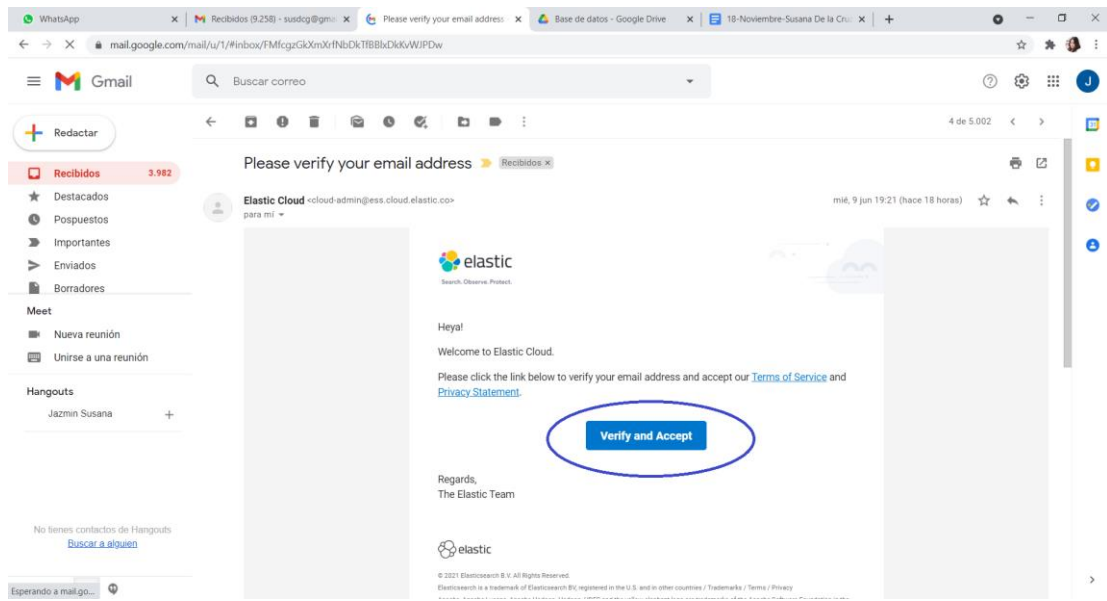
1. CONTENIDO	2
2. CREAR UNA CUENTA EN ELASTIC	3
3. CREAR UN DEPLOYMENT	5
4. CREAR UN ÍNDICE	7
5. CARGA DE DATOS CON EL API BULK.....	10
6. REALIZAR BÚSQUEDAS SOBRE EL ÍNDICE LOG_CONSULTAS	12
7. CREACIÓN DE UN INDEX PATTERNS.....	16
8. CREACIÓN DE UN DASHBOARD	18
9. CREACIÓN DE VISTAS	21

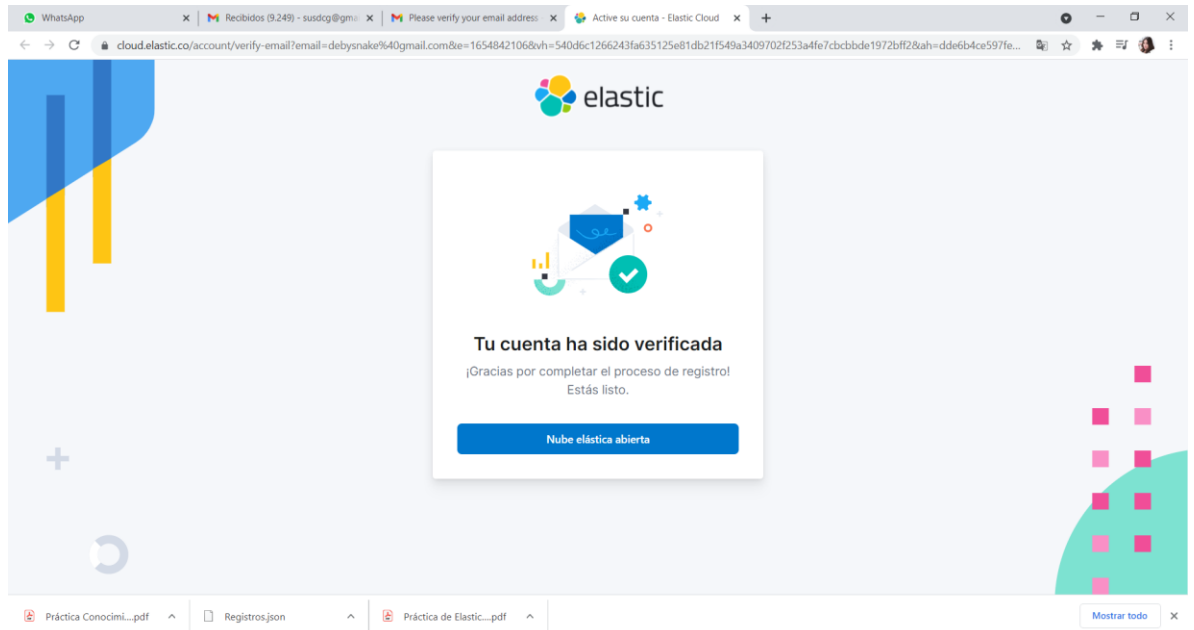
2. Crear una cuenta en elastic

Creamos una cuenta de elastic con el correo debysnake@gmail.com



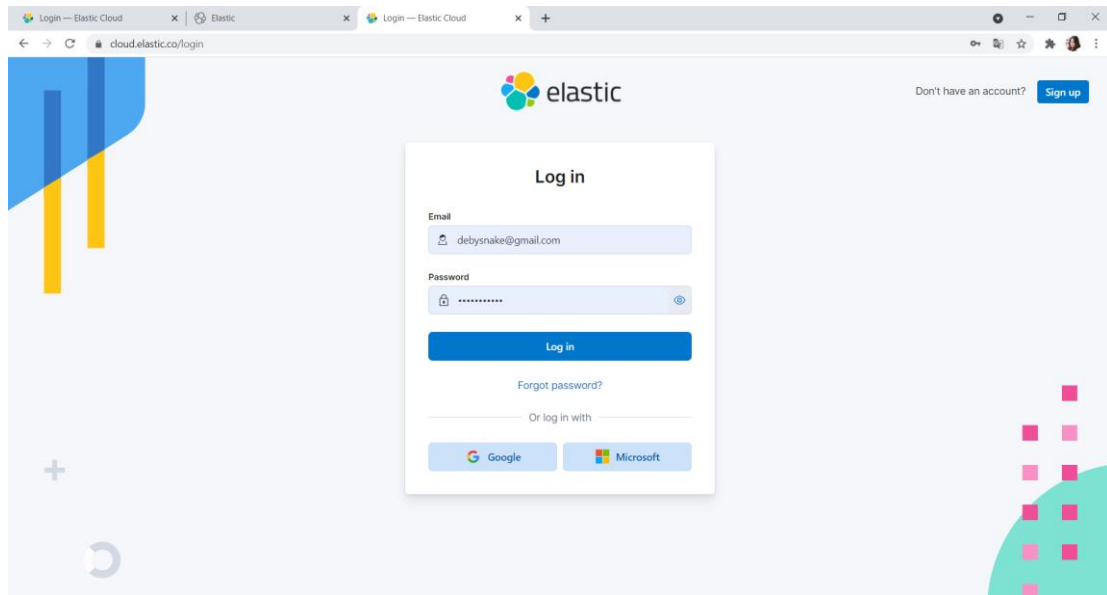
Llega un correo para verificar la cuenta. Damos clien en “Verify and Accept”



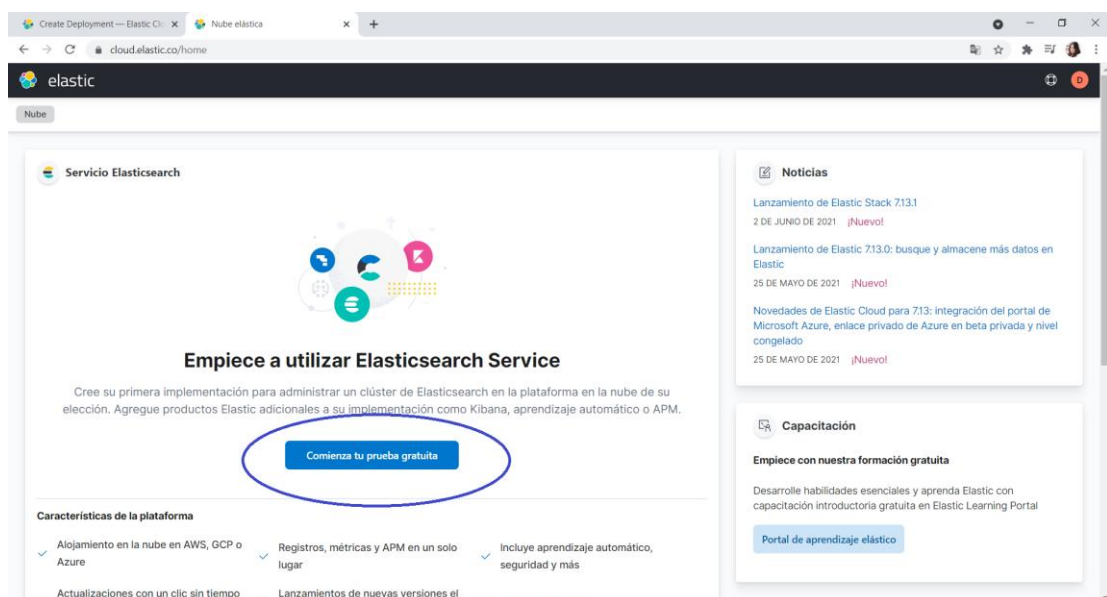


3. Crear un deployment

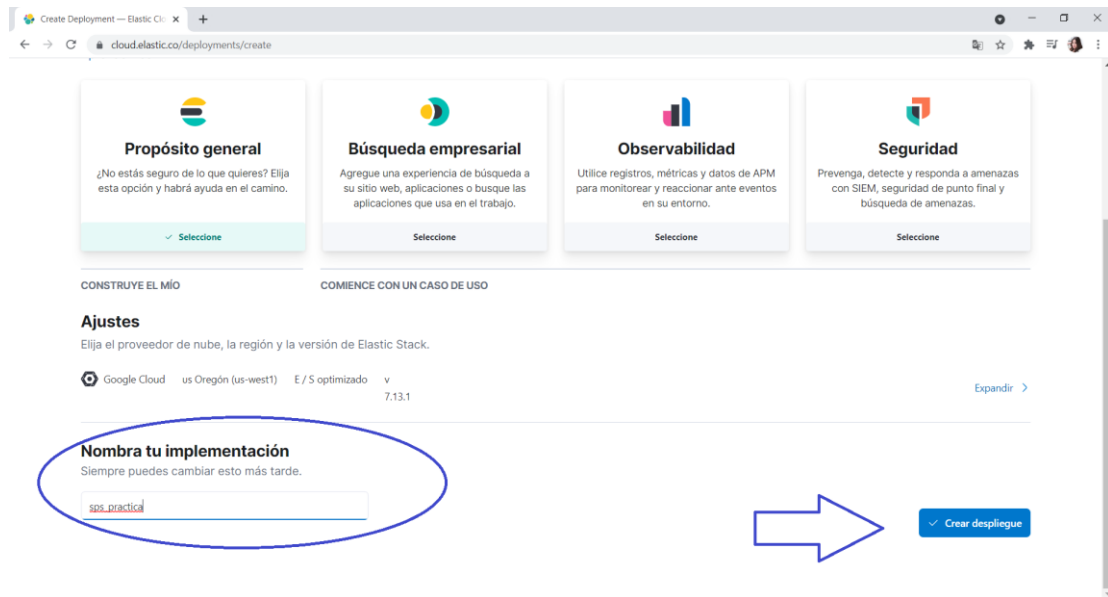
Nos dirigimos a la página <https://cloud.elastic.co/login> y nos logueamos con nuestras credenciales



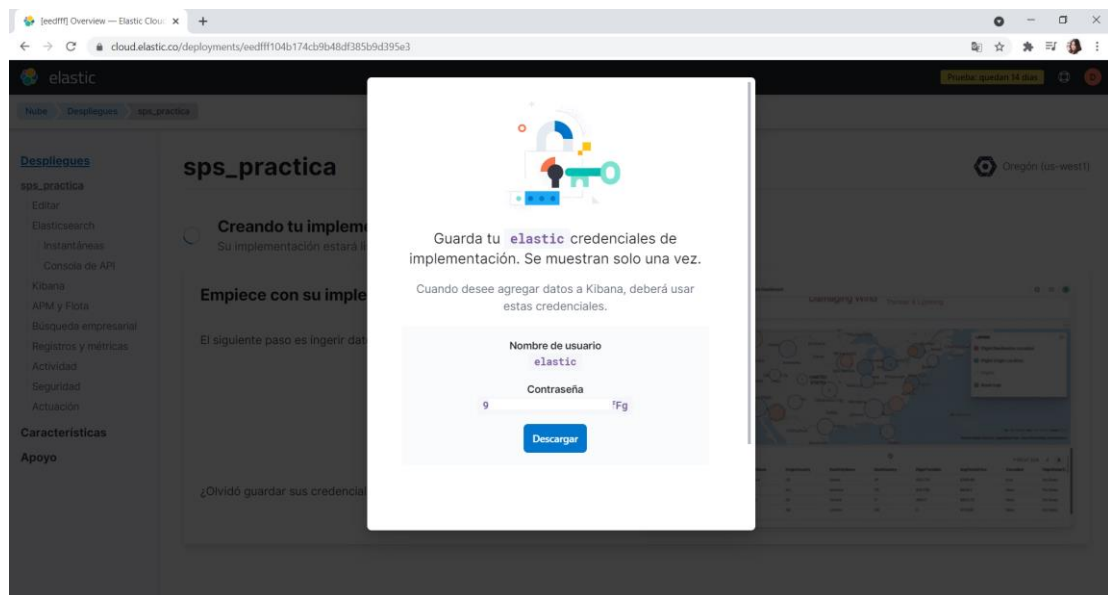
Damos click en “Comenzar tu prueba gratuita”



Nombramos la implementación como “sps_practica” y damos clic en “Crear despliegue”



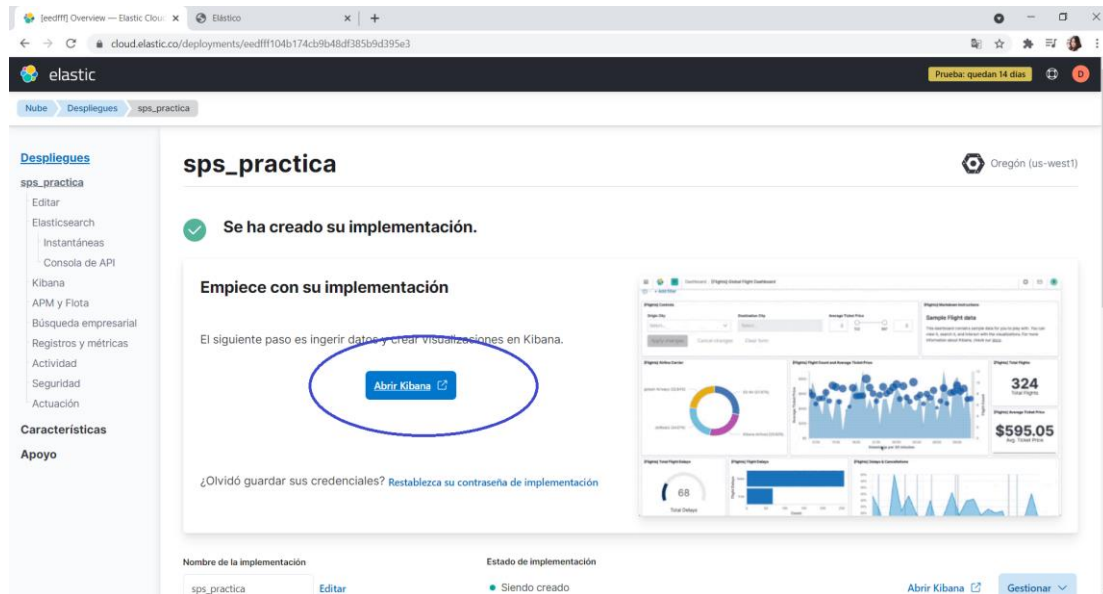
Nos muestra las credenciales generadas:



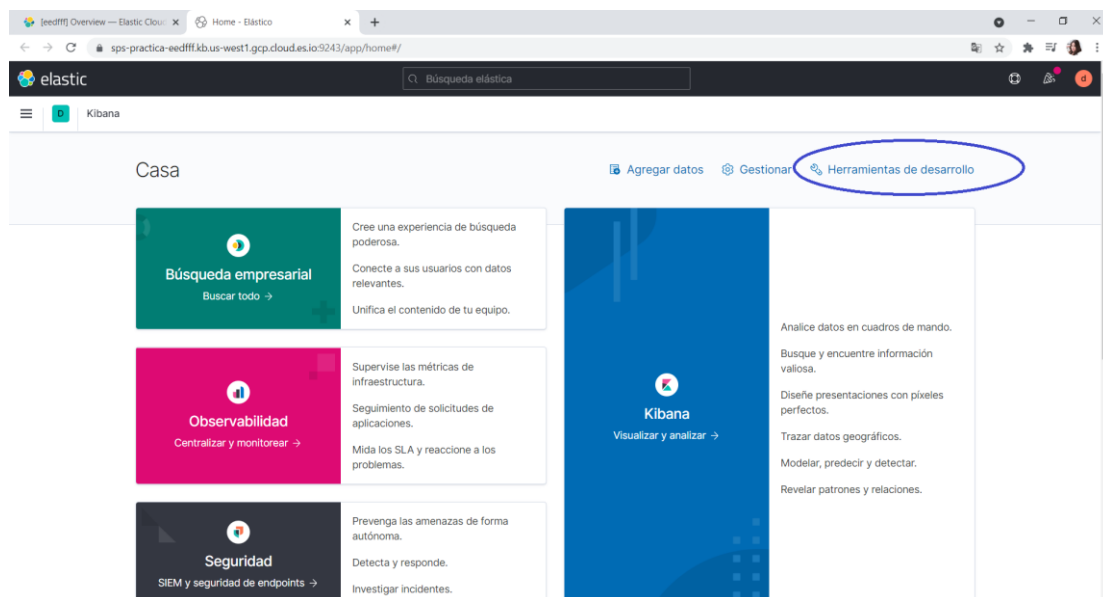
La implementación se creó con éxito.

4. Crear un índice

Damos click en el botón “Abrir Kibana”



Damos click en la opción “Herramientas de desarrollo”

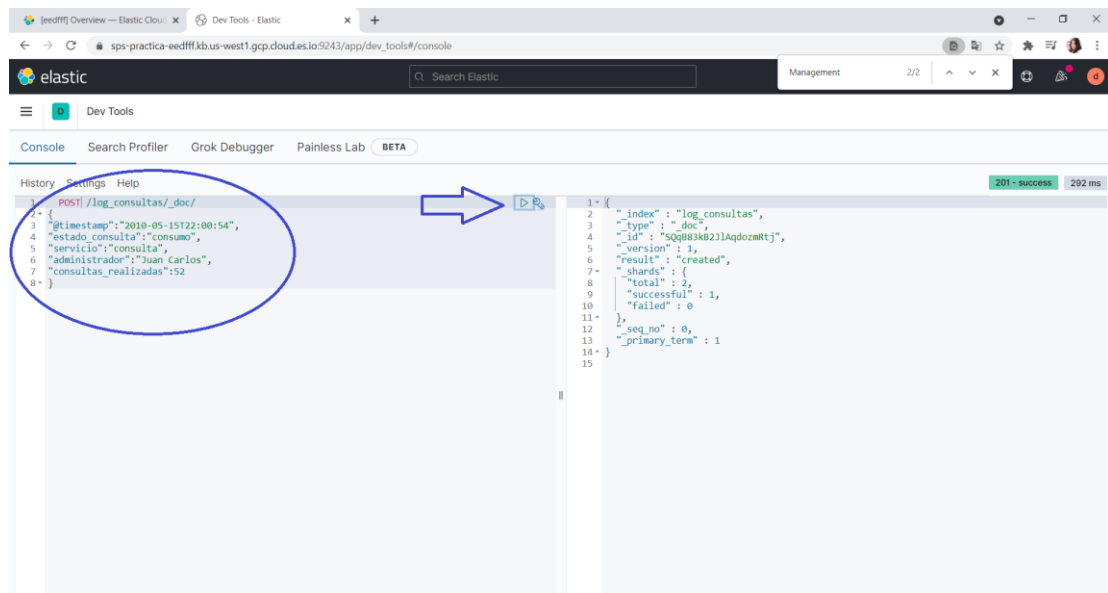


Creamos el índice log_consultas con la siguiente instrucción:

```
POST /log_consultas/_doc/
{
```

```
"@timestamp":"2010-05-15T22:00:54",
"estado_consulta":"consumo",
"servicio":"consulta",
"administrador":"Juan Carlos",
"consultas_realizadas":52
}
```

Damos click en ejecutar y la respuesta la veremos del lado derecho:



Para verificar que el índice se creó correctamente ejecutamos:

```
GET /log_consultas/_search?q=*
```

Mostrando el resultado en la parte derecha.

elastic Search Elastic Management 2/2

Dev Tools

Console Search Profiler Grok Debugger Painless Lab BETA

History Settings Help 200 - success 144 ms

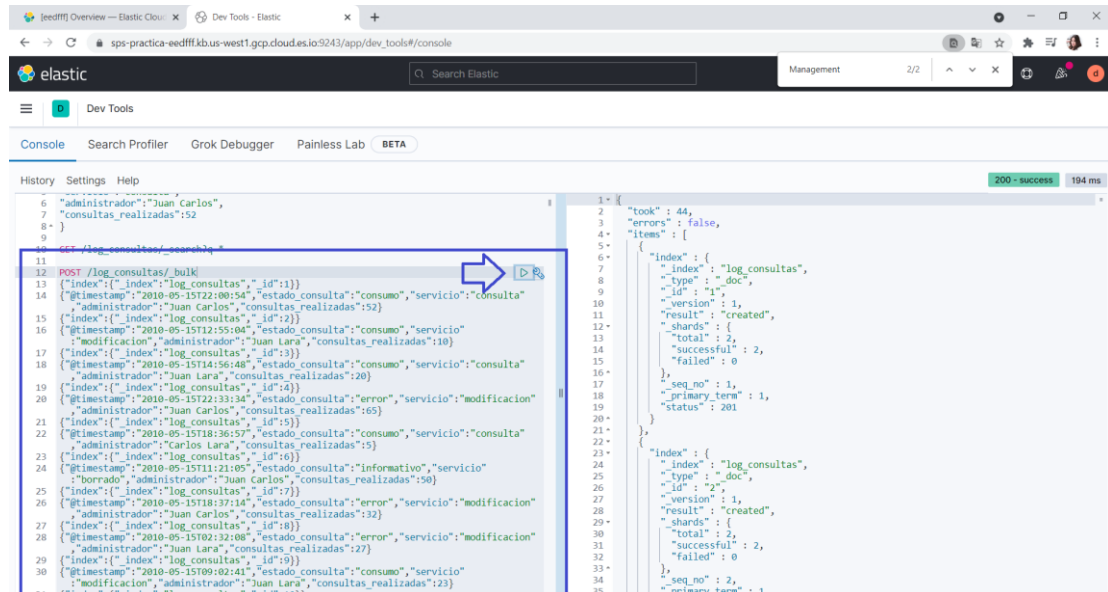
```
1 POST /log_consultas/_doc/
2 {
3   "@timestamp":"2010-05-15T22:00:54",
4   "estado_consulta":"consumo",
5   "servicio":"consulta",
6   "administrador":"Juan Carlos",
7   "consultas_realizadas":52
8 }
9
```

GET /log_consultas/_search?q=

```
1 {
2   "took": 12,
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": 1,
13      "relation": "eq"
14    },
15    "max_score": 1.0,
16    "hits": [
17      {
18        "_index": "log_consultas",
19        "_type": "_doc",
20        "_id": "5QqB83k2ZlAqozmRtj",
21        "_score": 1.0,
22        "_source": {
23          "@timestamp": "2010-05-15T22:00:54",
24          "estado_consulta": "consumo",
25          "servicio": "consulta",
26          "administrador": "Juan Carlos",
27          "consultas_realizadas": 52
28        }
29      }
30    ]
31  }
32 }
```

5. Carga de datos con el API bulk

Abrir el archivo Registros.json e insertarlos mediante el método POST. Damos click en “Ejecutar” y veremos la respuesta del lado derecho.



The screenshot shows the Elastic Dev Tools console with a POST request to the bulk API. The request body is a JSON array of documents. The response on the right shows a successful bulk operation with 200 items indexed.

```
12 POST /log_consultas/_bulk
13 [{"index":{"_index":"log_consultas","_id":1}}
14 {"@timestamp":"2010-05-15T22:00:54","estado_consulta":"consumo","servicio":"consulta",
15 "administrador":"Juan Carlos","consultas_realizadas":52}
16 {"index":{"_index":"log_consultas","_id":2}}
17 {"@timestamp":"2010-05-15T12:55:04","estado_consulta":"consumo","servicio":"consulta",
18 "modificacion":"administrador":"Juan Lara","consultas_realizadas":10}
19 {"index":{"_index":"log_consultas","_id":3}}
20 {"@timestamp":"2010-05-15T14:56:48","estado_consulta":"consumo","servicio":"consulta",
21 "administrador":"Juan Lara","consultas_realizadas":20}
22 {"index":{"_index":"log_consultas","_id":4}}
23 {"@timestamp":"2010-05-15T22:33:34","estado_consulta":"error","servicio":"modificacion",
24 "administrador":"Juan Carlos","consultas_realizadas":65}
25 {"index":{"_index":"log_consultas","_id":5}}
26 {"@timestamp":"2010-05-15T18:36:57","estado_consulta":"consumo","servicio":"consulta",
27 "administrador":"Carlos Lara","consultas_realizadas":5}
28 {"index":{"_index":"log_consultas","_id":6}}
29 {"@timestamp":"2010-05-15T11:21:05","estado_consulta":"informativo","servicio":"borrado",
30 "administrador":"Juan Carlos","consultas_realizadas":50}
31 {"index":{"_index":"log_consultas","_id":7}}
32 {"@timestamp":"2010-05-15T18:37:14","estado_consulta":"error","servicio":"modificacion",
33 "administrador":"Juan Carlos","consultas_realizadas":32}
34 {"index":{"_index":"log_consultas","_id":8}}
35 {"@timestamp":"2010-05-15T02:32:08","estado_consulta":"error","servicio":"modificacion",
36 "administrador":"Juan Lara","consultas_realizadas":27}
37 {"index":{"_index":"log_consultas","_id":9}}
38 {"@timestamp":"2010-05-15T09:02:41","estado_consulta":"consumo","servicio":"modificacion",
39 "administrador":"Juan Lara","consultas_realizadas":23}
40 ]}
```

```
1 {
2   "took": 44,
3   "errors": false,
4   "items": [
5     {
6       "index": {
7         "_index": "log_consultas",
8         "_type": "_doc",
9         "_id": "1",
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": "log_consultas",
25        "_type": "_doc",
26        "_id": "2",
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      }
37    }
38  ]
39 }
```

Para verificar que los datos se insertaron bien, ejecutamos el comando:

[GET /log_consultas/_search?q=*](#)

Damos click en “Ejecutar” y veremos el resultado del lado derecho.

elastic Dev Tools

Console Search Profiler Grok Debugger Painless Lab BETA

History Settings Help

200 - success 511 ms

```
6 "administrador": "Juan Carlos",
7 "consultas_realizadas": 52
8 }
9 GET /log_consultas/_search?q=*
10
11 POST /log_consultas/_bulk
12 {
13   {"index":{"_index":"log_consultas","_id":"11"}}
14   {"@timestamp":"2010-05-15T22:00:54","estado_consulta":"consumo","servicio":"consulta",
15     "administrador":"Juan Carlos","consultas_realizadas":52}
16   {"index":{"_index":"log_consultas","_id":"22"}}
17   {"@timestamp":"2010-05-15T12:55:04","estado_consulta":"consumo","servicio":
18     "modificacion","administrador":"Juan Lara","consultas_realizadas":10}
19   {"index":{"_index":"log_consultas","_id":"33"}}
20   {"@timestamp":"2010-05-15T14:56:48","estado_consulta":"consumo","servicio":"consulta",
21     "administrador":"Juan Lara","consultas_realizadas":20}
22   {"index":{"_index":"log_consultas","_id":"44"}}
23   {"@timestamp":"2010-05-15T22:33:34","estado_consulta":"error","servicio":"modificacion",
24     "administrador":"Juan Carlos","consultas_realizadas":65}
25   {"index":{"_index":"log_consultas","_id":"55"}}
26   {"@timestamp":"2010-05-15T18:36:57","estado_consulta":"consumo","servicio":"consulta",
27     "administrador":"Carlos Lara","consultas_realizadas":5}
28   {"index":{"_index":"log_consultas","_id":"66"}}
29   {"@timestamp":"2010-05-15T11:21:05","estado_consulta":"Informativo","servicio":
30     "borrado","administrador":"Juan Carlos","consultas_realizadas":50}
31   {"index":{"_index":"log_consultas","_id":"77"}}
32   {"@timestamp":"2010-05-15T18:37:14","estado_consulta":"error","servicio":"modificacion",
33     "administrador":"Juan Carlos","consultas_realizadas":32}
34   {"index":{"_index":"log_consultas","_id":"88"}}
35   {"@timestamp":"2010-05-15T02:32:08","estado_consulta":"error","servicio":"modificacion",
36     "administrador":"Juan Lara","consultas_realizadas":27}
37   {"index":{"_index":"log_consultas","_id":"99"}}
38   {"@timestamp":"2010-05-15T09:02:41","estado_consulta":"consumo","servicio":
39     "modificacion","administrador":"Juan Lara","consultas_realizadas":23}
40 }
41
42 {
43   "@timestamp": "2010-05-15T22:00:54",
44   "estado_consulta": "consumo",
45   "servicio": "consulta",
46   "administrador": "Juan Carlos",
47   "consultas_realizadas": 52
48 }
49
50 {
51   "index": "log_consultas",
52   "type": "doc",
53   "id": "1",
54   "score": 1.0,
55   "source": {
56     "@timestamp": "2010-05-15T22:00:54",
57     "estado_consulta": "consumo",
58     "servicio": "consulta",
59     "administrador": "Juan Carlos",
60     "consultas_realizadas": 52
61   }
62 }
63
64 {
65   "index": "log_consultas",
66   "type": "doc",
67   "id": "2",
68   "score": 1.0,
69   "source": {
70     "@timestamp": "2010-05-15T12:55:04",
71     "estado_consulta": "consumo",
72     "servicio": "modificacion",
73     "administrador": "Juan Lara",
74     "consultas_realizadas": 10
75   }
76 }
77
78 {
79   "index": "log_consultas",
80   "type": "doc",
81   "id": "3",
82   "score": 1.0,
83   "source": {
84     "@timestamp": "2010-05-15T14:56:48",
85     "estado_consulta": "consumo",
86     "servicio": "consulta",
87     "administrador": "Juan Lara",
88     "consultas_realizadas": 20
89   }
90 }
91
92 {
93   "index": "log_consultas",
94   "type": "doc",
95   "id": "4",
96   "score": 1.0,
97   "source": {
98     "@timestamp": "2010-05-15T22:33:34",
99     "estado_consulta": "error",
100    "servicio": "modificacion",
101    "administrador": "Juan Carlos",
102    "consultas_realizadas": 65
103  }
104 }
```

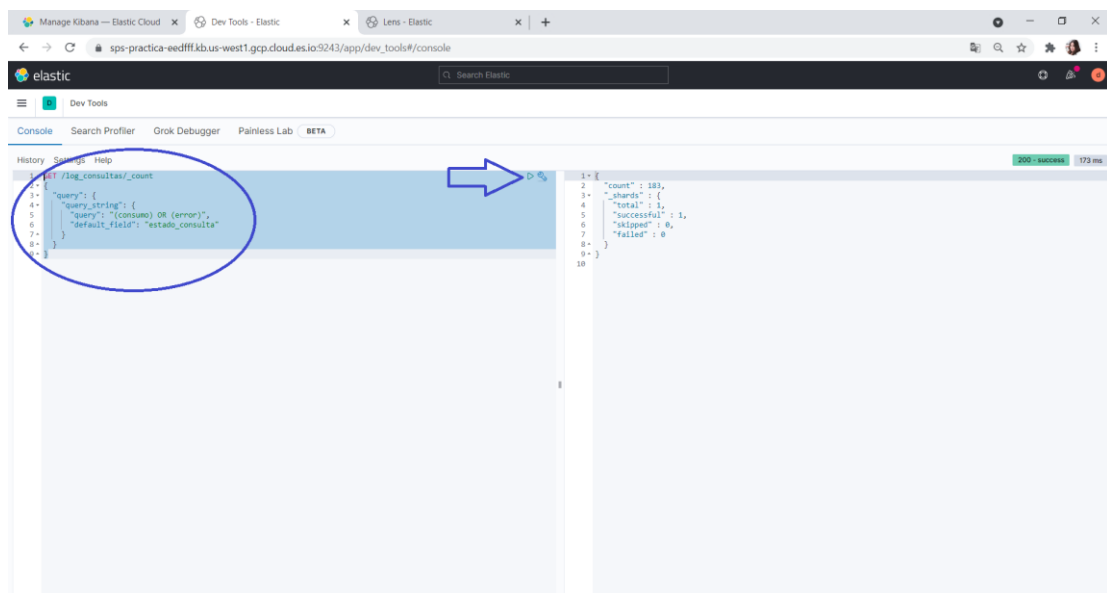
6. Realizar búsquedas sobre el índice log_consultas

Consulta 1. Obtener el número de registros con estado_consulta igual a error y consumo.

Ejecutamos el comando:

```
GET /log_consultas/_count
{
  "query": {
    "query_string": {
      "query": "(consumo) OR (error)",
      "default_field": "estado_consulta"
    }
  }
}
```

Damos click en ejecutar y veremos la respuesta del lado derecho:

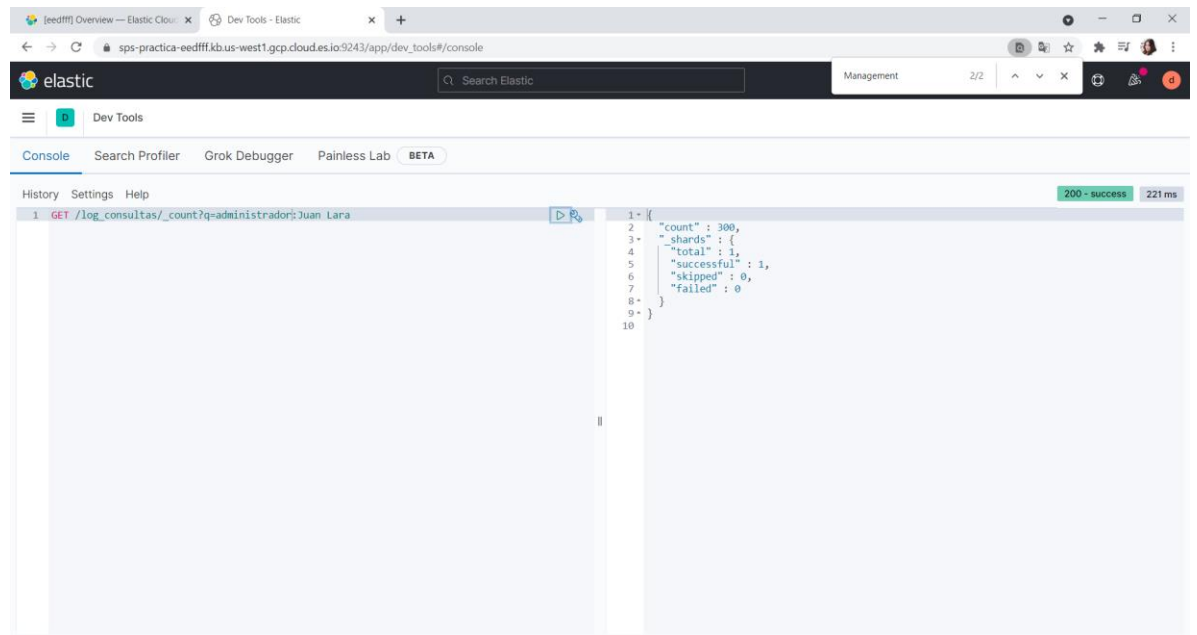


Consulta 2. Obtener el número de registros realizados por el administrador Juan Lara.

Ejecutamos el comando:

[GET /log_consultas/_count?q=administrador:Juan Lara](#)

Damos click en ejecutar y veremos la respuesta del lado derecho:

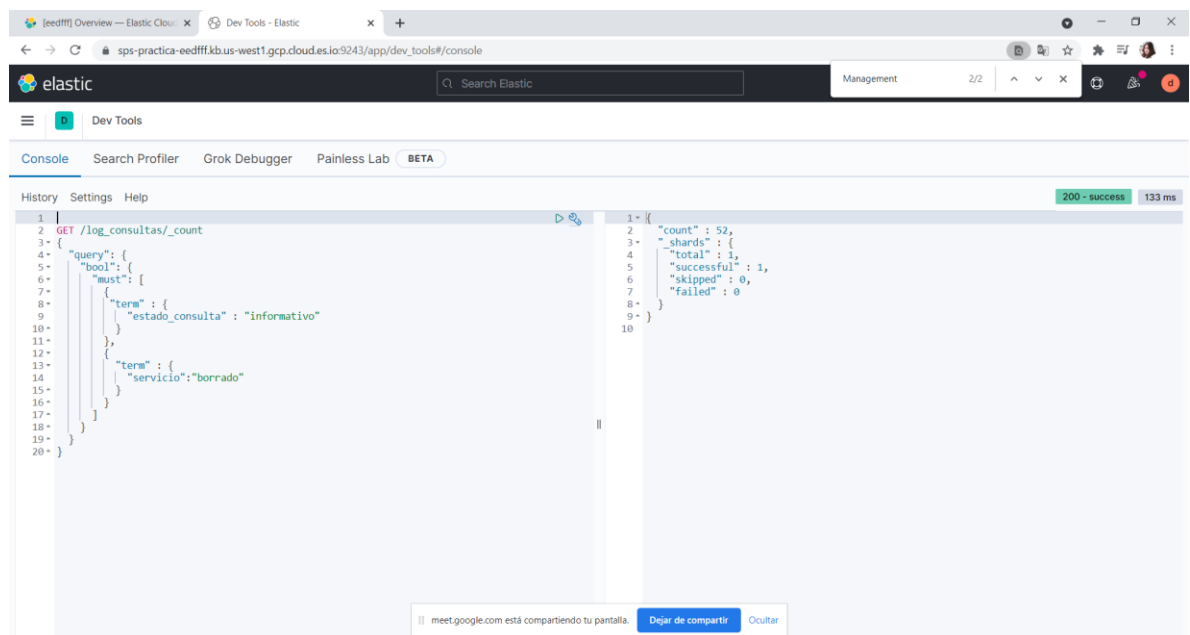


Consulta 3. Obtener el número de registros con estado_consulta igual a informativo y servicio igual a borrado

Ejecutamos el comando:

```
GET /log_consultas/_count
{
  "query": {
    "bool": {
      "must": [
        {
          "term": {
            "estado_consulta": "informativo"
          }
        },
        {
          "term": {
            "servicio": "borrado"
          }
        }
      ]
    }
  }
}
```

Damos click en ejecutar y veremos la respuesta del lado derecho:



The screenshot shows the Elastic Dev Tools interface. The console tab is active, displaying a successful GET request to `/log_consultas/_count`. The request body is a JSON object with a `query` field containing a `bool` query with two `must` clauses: one for `estado_consulta: informativo` and another for `servicio: borrado`. The response is a JSON object with the following structure:

```
{
  "count": 52,
  "_shards": {
    "total": 1,
    "successful": 1,
    "skipped": 0,
    "failed": 0
  }
}
```

The status bar at the top right of the console indicates `200 - success` and `133 ms`. At the bottom of the console, there is a sharing notification: `meet.google.com está compartiendo tu pantalla.` with buttons for `Dejar de compartir` and `Ocultar`.

Consulta extra. Obtener la suma de los valores en consultas realizadas con estado_consulta igual a error

Ejecutamos el comando:

`POST /log_consultas/_search?size=0`

```
{
  "query": {
    "constant_score": {
      "filter": {
        "match": { "estado_consulta": "error" }
      }
    }
  },
  "aggs": {
    "sum_consultas": { "sum": { "field": "consultas_realizadas" } }
  }
}
```

Damos click en ejecutar y veremos la respuesta del lado derecho:

The screenshot shows the Elastic Dev Tools interface. The left pane displays the executed query, and the right pane shows the JSON response. The status bar at the top right indicates '200 - success' and '149 ms'.

Query (Left Pane):

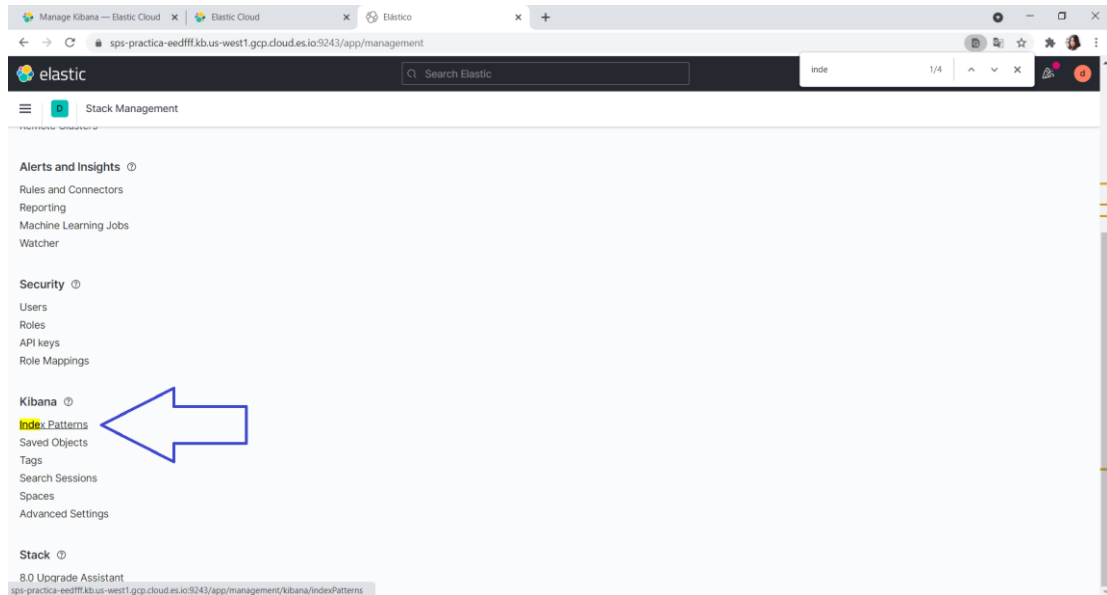
```
1 POST /log_consultas/_search?size=0
2 {
3   "query": {
4     "constant_score": {
5       "filter": {
6         "match": { "estado_consulta": "error" }
7       }
8     }
9   },
10  "aggs": {
11    "sum_consultas": { "sum": { "field": "consultas_realizadas" } }
12  }
13 }
14 }
```

Response (Right Pane):

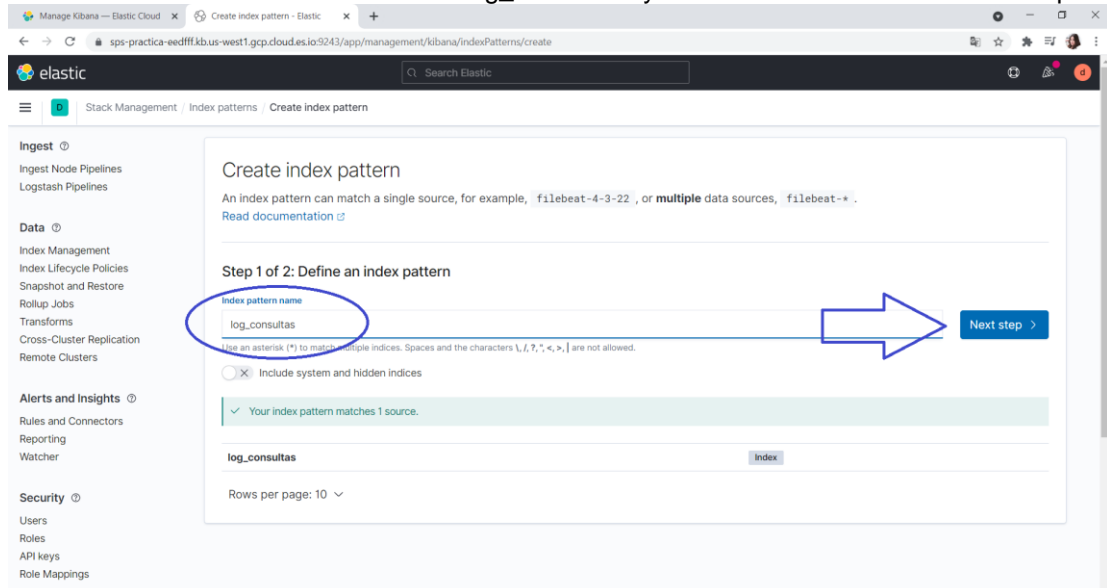
```
1 {
2   "took": 2,
3   "timed_out": false,
4   "shards": {
5     "total": 1,
6     "successful": 1,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": {
12      "value": 78,
13      "relation": "eq"
14    },
15    "max_score": null,
16    "hits": [ ]
17  },
18  "aggregations": {
19    "sum_consultas": {
20      "value": 2865.0
21    }
22  }
23 }
24 }
```

7. Creación de un Index Patterns

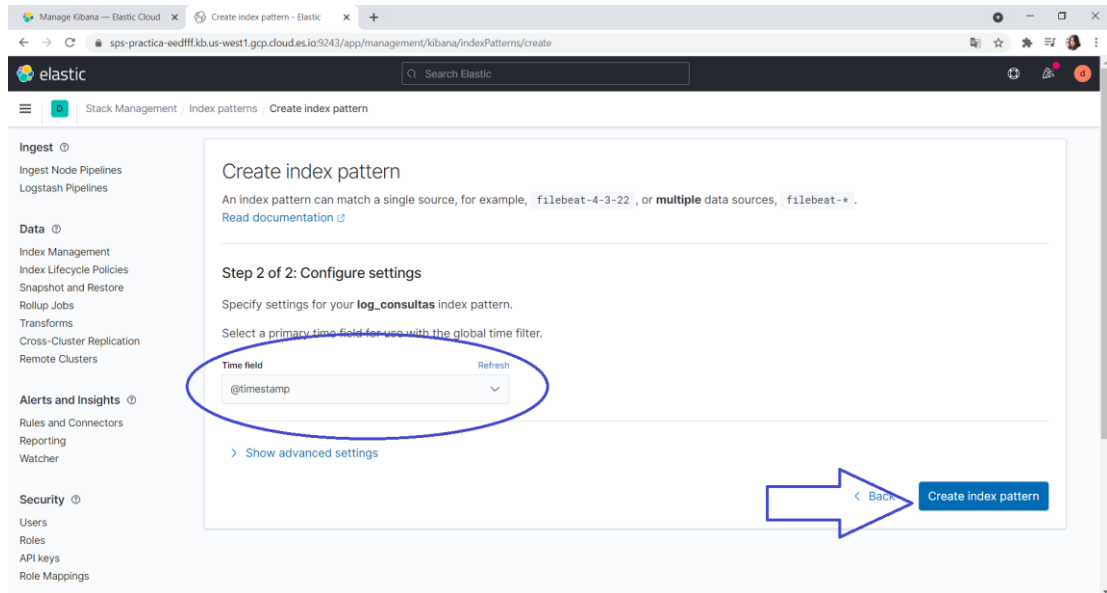
Desde el administrador, damos click en la opción “Index Patterns”



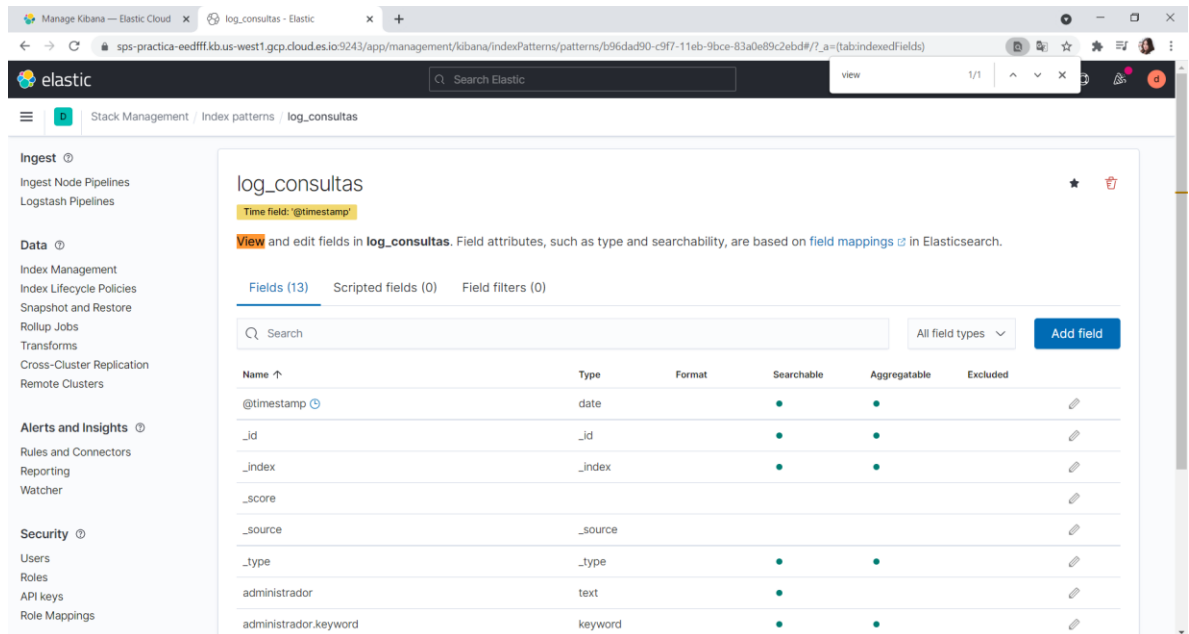
Definimos el nombre del índice como “log_consultas” y damos click en el botón “Next step”



Seleccionamos el campo @timestamp como Time Filter y damos click en el botón "Create index pattern"

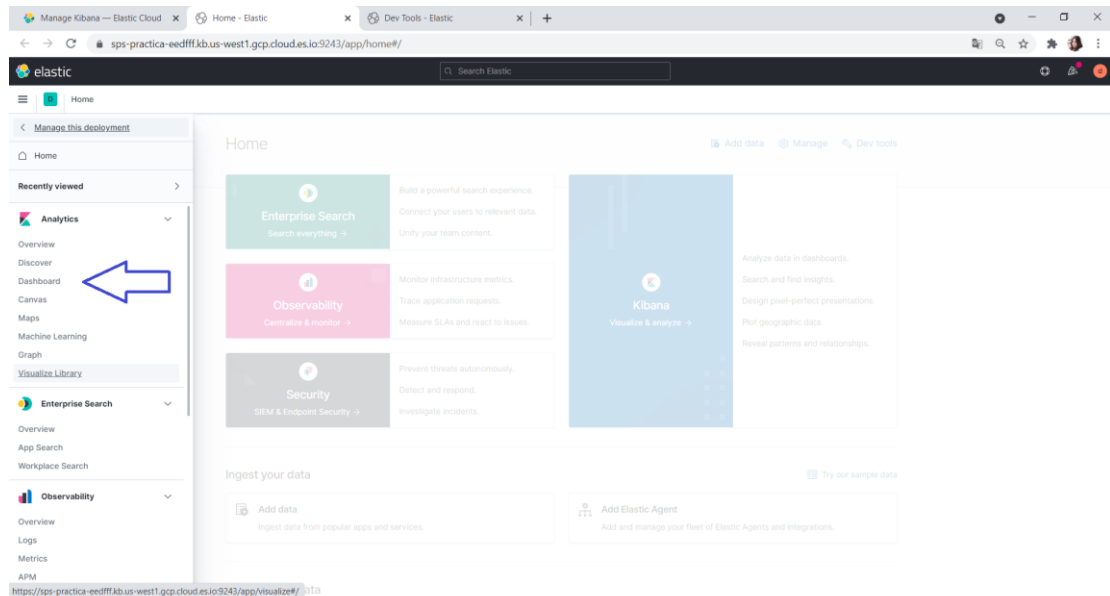


El índice se ha creado de forma correcta:

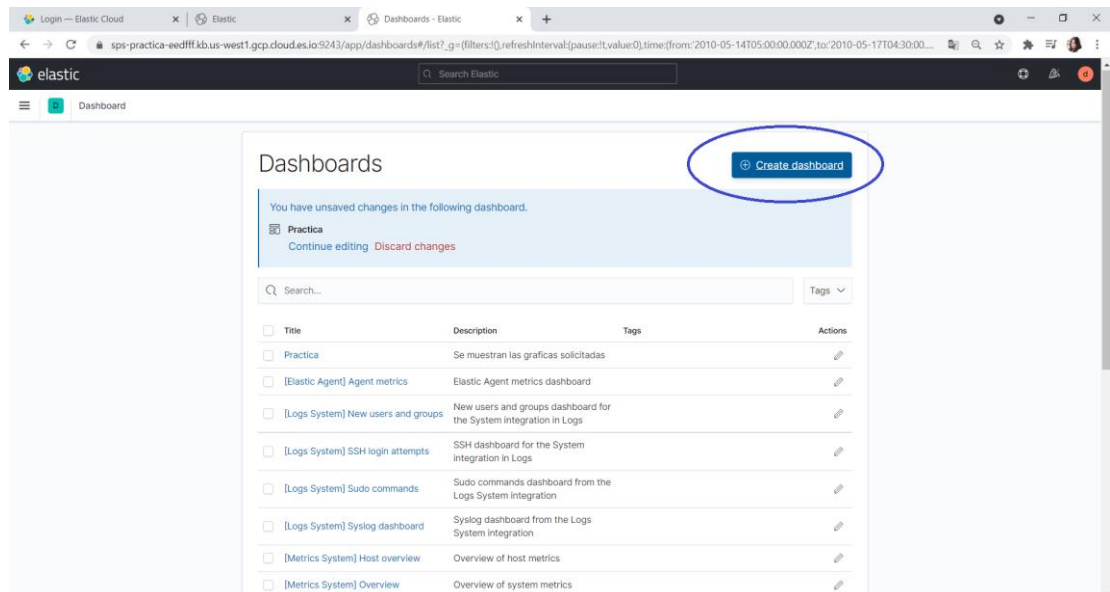


8. Creación de un dashboard

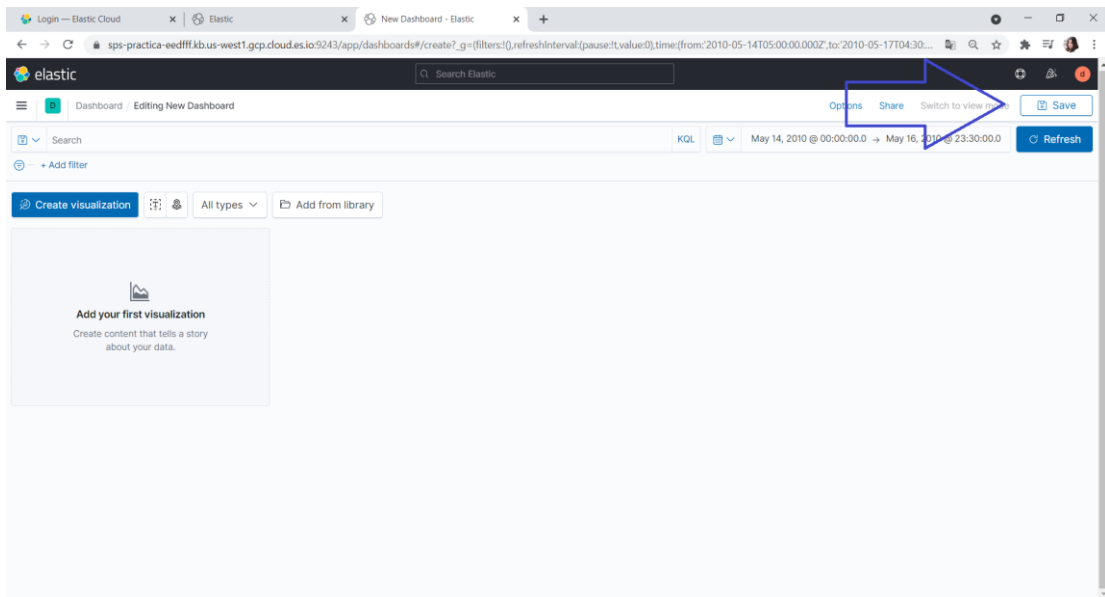
Dentro de la opción “Analytics” seleccionamos “Dashboard”.



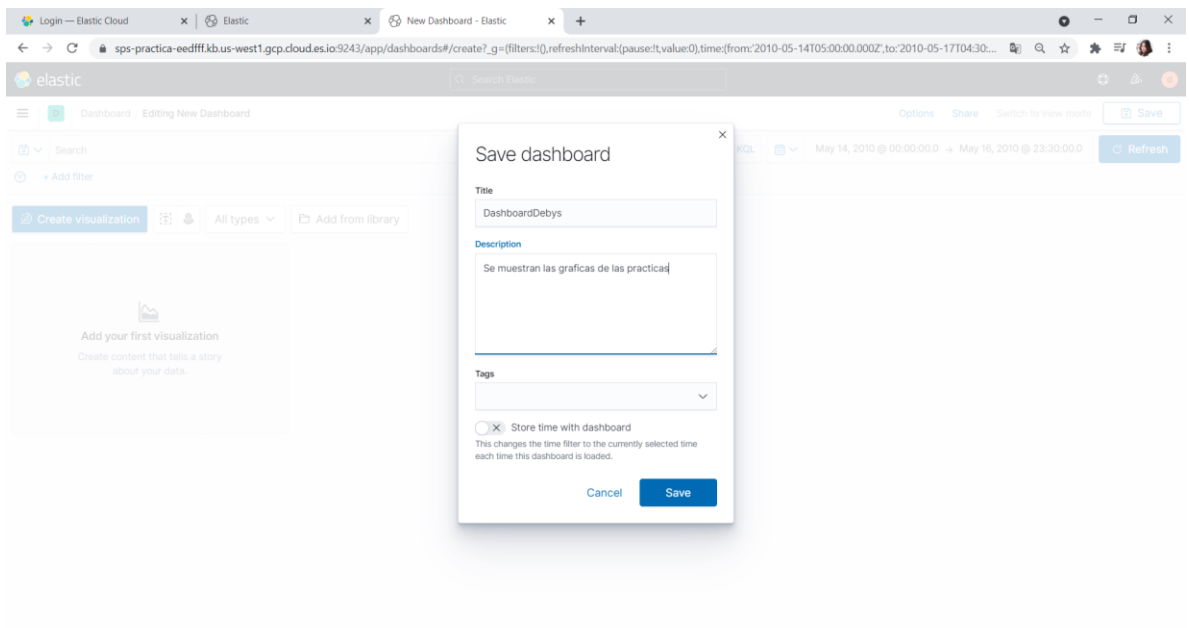
Damos click en el botón “Create dashboard”



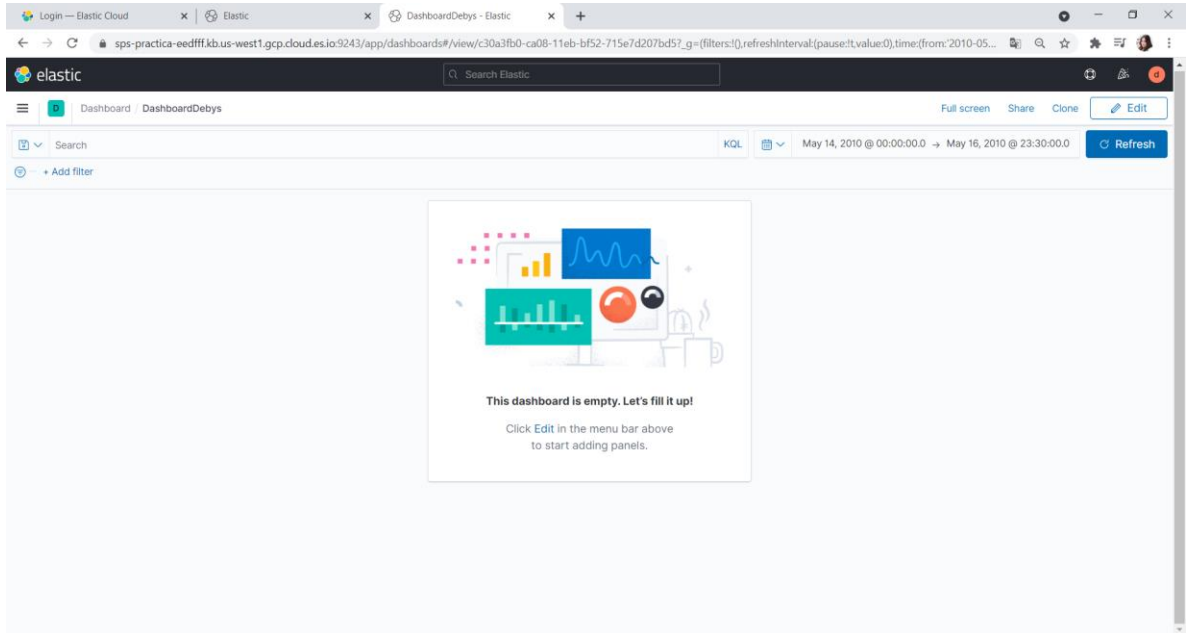
Dar click en el botón “Save”



Introducimos un título para el dashboard y una descripción. Damos click en el botón “Save”

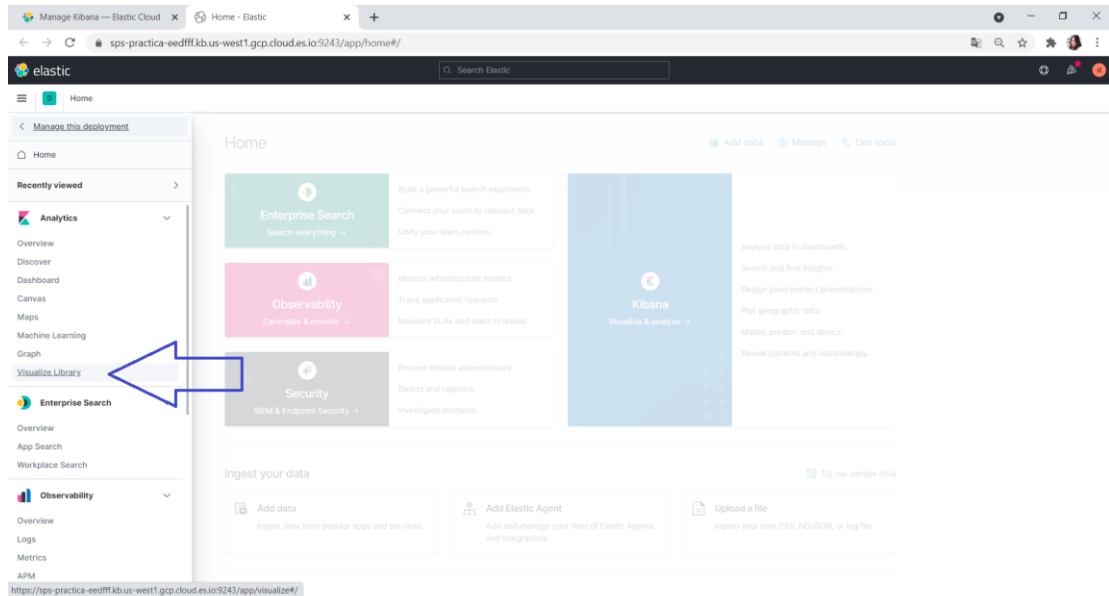


Nuestro dashborad se ha guardado



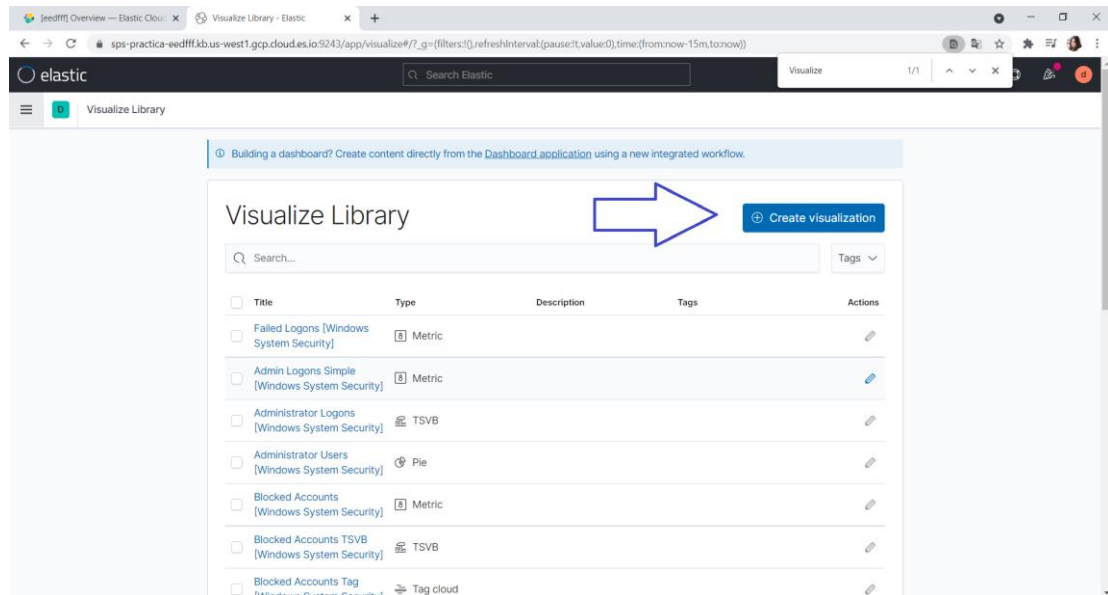
9. Creación de vistas

Desde el menú seleccionamos la opción “Visualize Library” que se encuentra en Analytics.

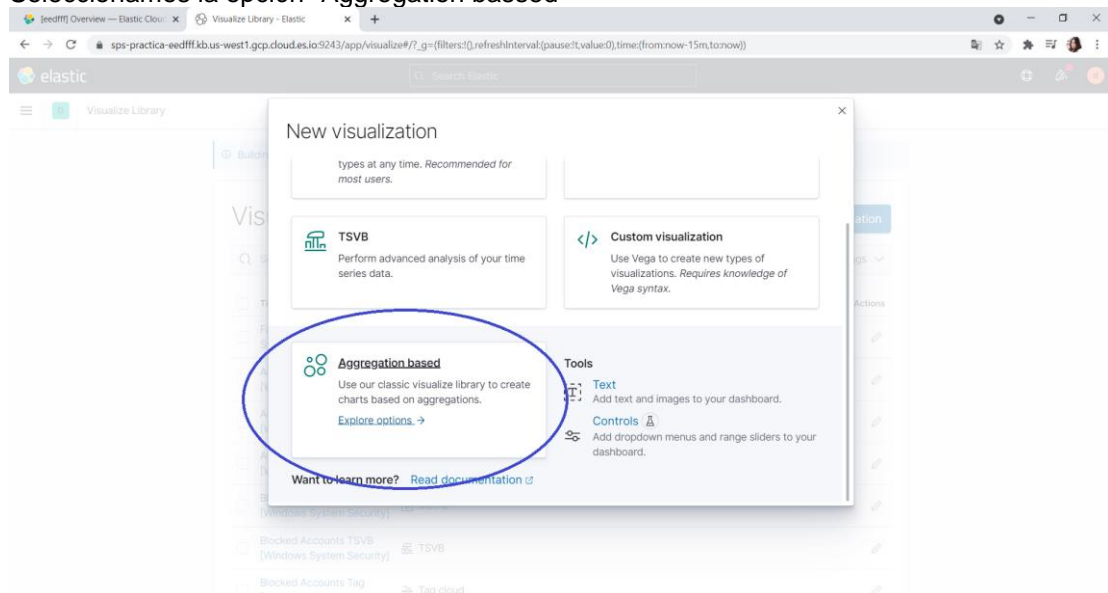


Visita 1. Vista de heat map, donde mostraras el número de servicios realizados por administrador.

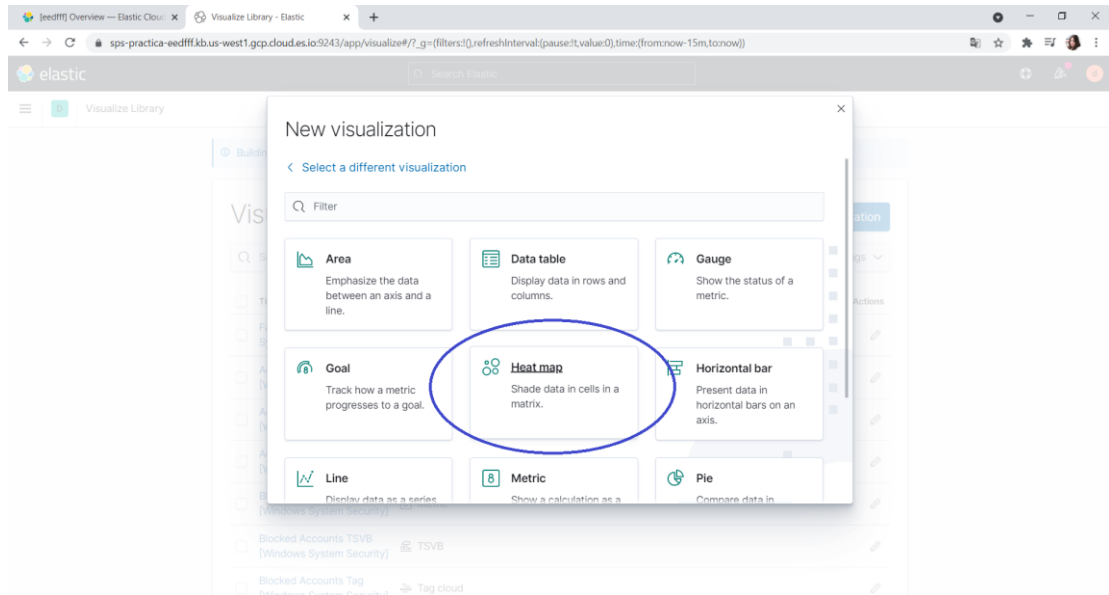
Damos click en el botón “Create visualization”



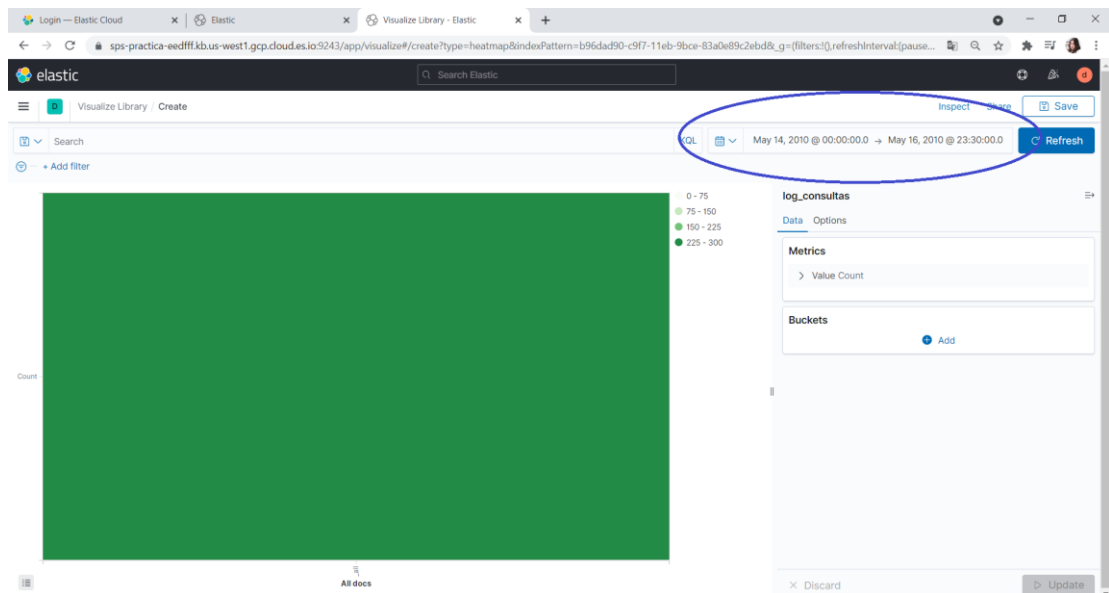
Seleccionamos la opción “Aggregation based”



Seleccionamos la opción “Heat map”



Para poder ver información debemos introducir el rango de fecha correcto con base a los datos que cargamos para el índice `log_consultas`. Damos click en el botón “Refresh”



Dentro del menu derecho, configuramos los datos que se mostraran en el eje x.

Modificamos los siguientes valores:

Aggregation = Terms

Field = administrador.keyword

Custom label = Administradores

Y damos click en el boton “Update”

InspectShareCancelSave to librarySave and return

May 14, 2010 @ 00:00:00.0 → May 16, 2010 @ 23:30:00.0

Refresh

log_consultas

DataOptions

Metrics

> Value Count

Buckets

X-axis

Aggregation

Terms

Field

administrador.keyword

Order by

Metric: Count

Order

Descending

Size

5

☐ Group other values in separate bucket

☐ Show missing values

Custom label

Administradores

Discard

Update

Posteriormente configuramos los valores para el eje y
Modificamos los siguientes valores:

Sub aggregation = Terms

Field = servicio.keyword

Custom label = Servicios

Y damos click en el boton "Update"

The screenshot shows the Kibana interface for a dashboard named 'log_consultas'. The 'Data' tab is selected, and the 'Buckets' section is expanded. The 'X-axis' is configured with 'administrador.keyword' in descending order. The 'Y-axis' is configured with a sub-aggregation of 'Terms' for the field 'servicio.keyword'. The 'Order by' is set to 'Metric: Count', and the 'Order' is 'Descending' with a 'Size' of 5. The 'Custom label' is set to 'Servicios'. The 'Update' button is highlighted with a blue arrow.

log_consultas

Data Options

Buckets

> X-axis administrador.keyword: Descending

Y-axis

Sub aggregation

Terms

Field

servicio.keyword

Order by

Metric: Count

Order

Descending

Size

5

Group other values in separate bucket

Show missing values

Custom label

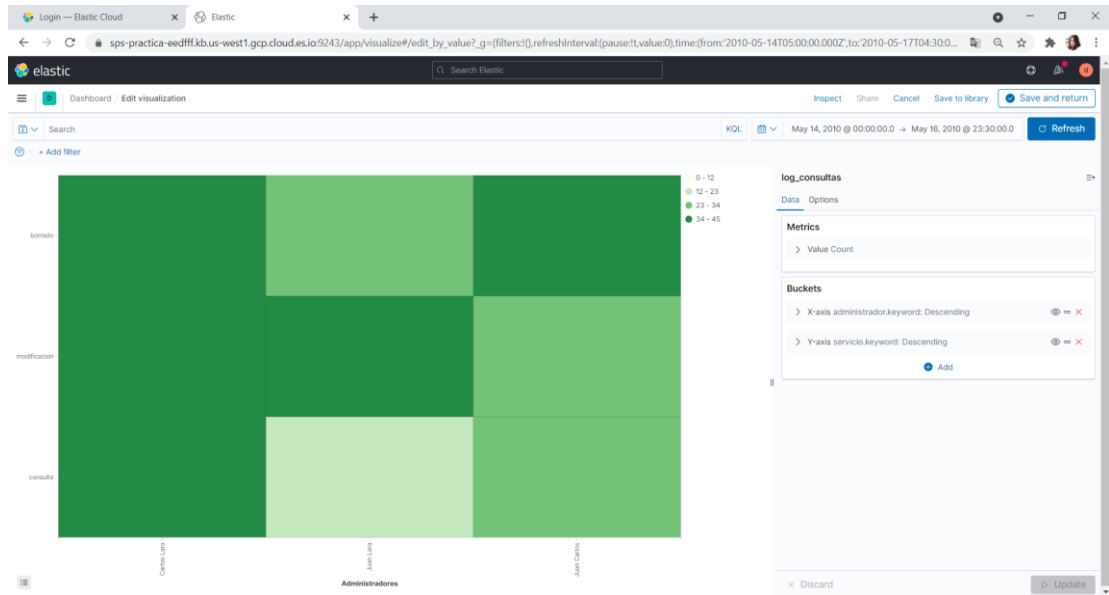
Servicios

> Advanced

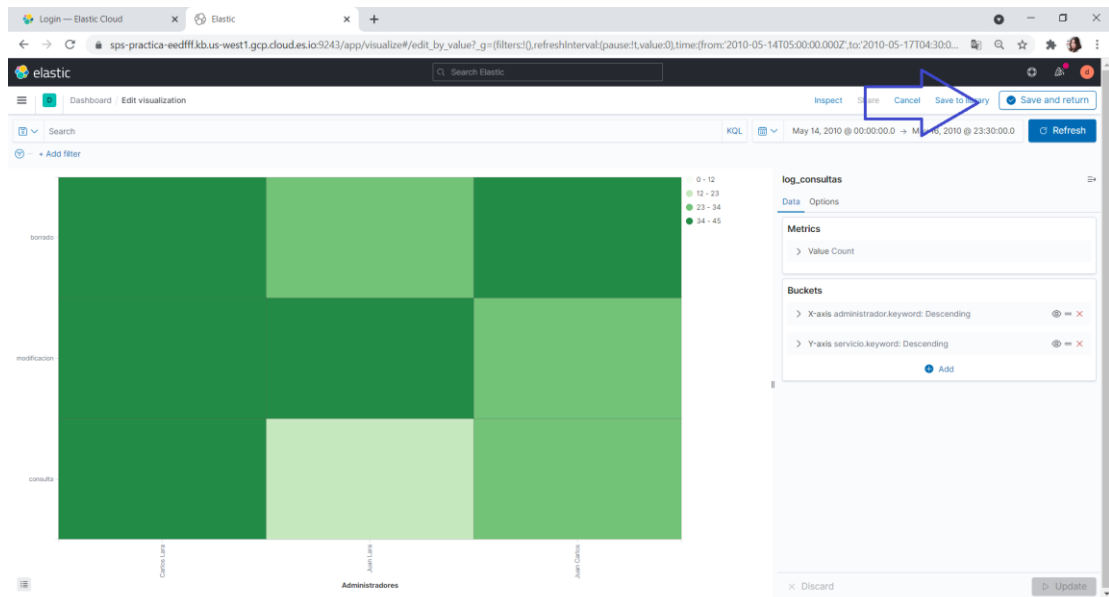
+ Add

Discard Update

La grafica resultante se muestra a continuación:

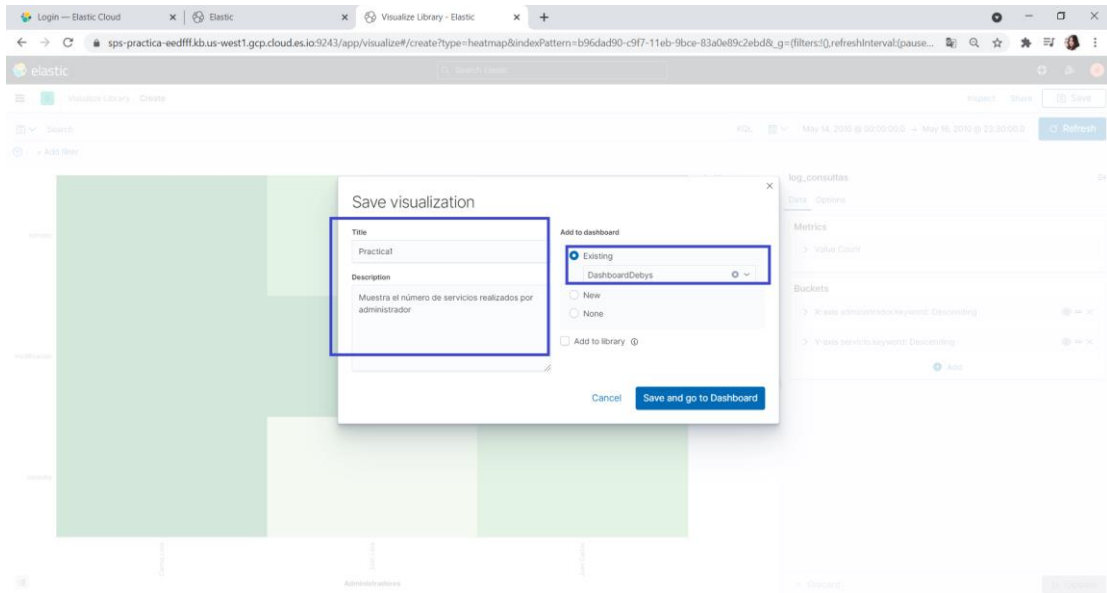


Damos click en el botón Save

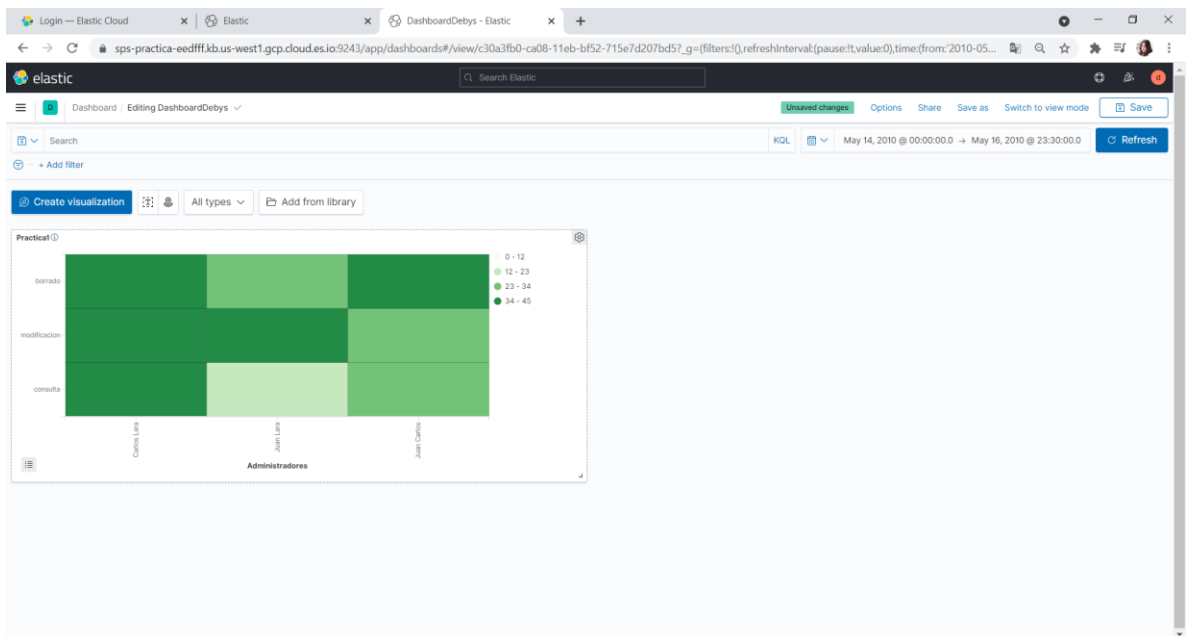


Introducimos un título, una descripción en indicamos que queremos guardarlo dentro del dashborad que creamos en la sección 8 de este documento.

Por ultimo damos click en “Save and go to Dashboard”

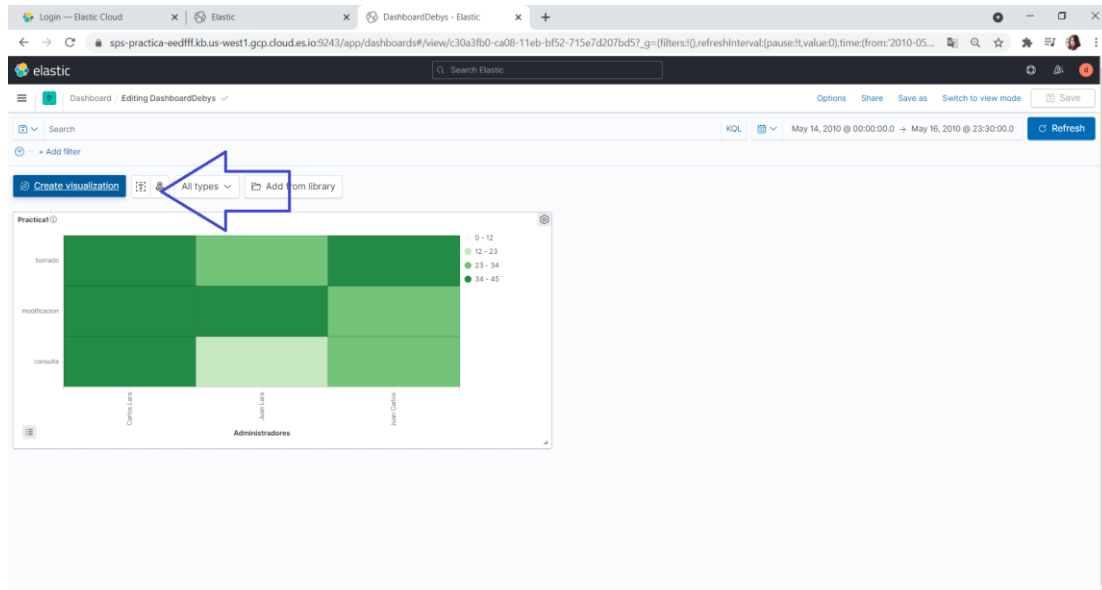


Veremos la vista en nuestro dashboard

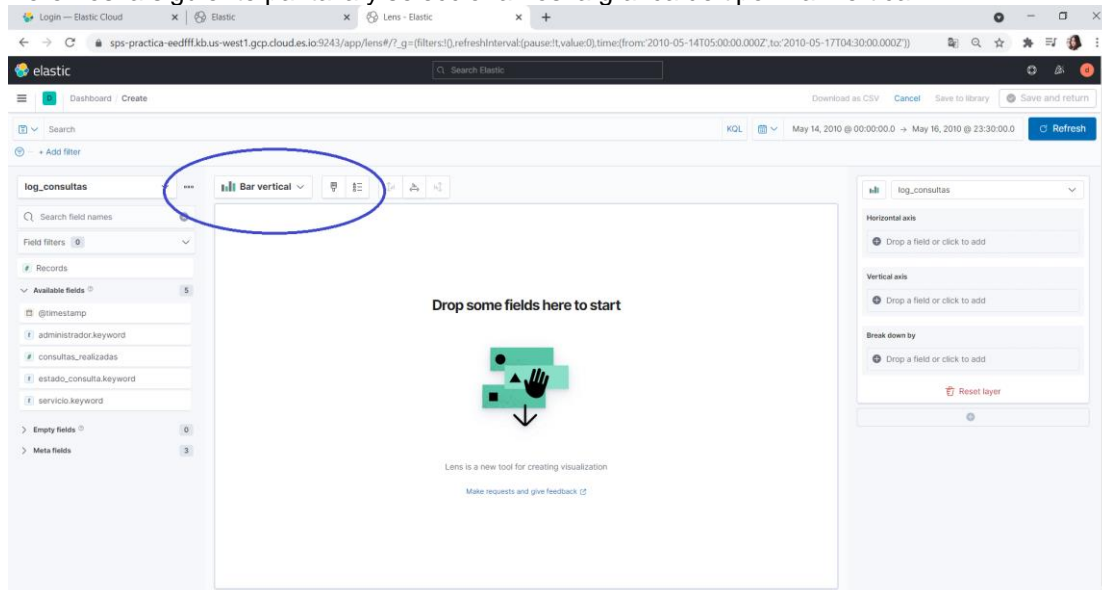


Vista 2. Vista de Barras, donde se grafique el número de registros con **estado_consulta** igual a error a través del tiempo.

Estando dentro de nuestro dashboard, damos click en el botón “Create visualization”



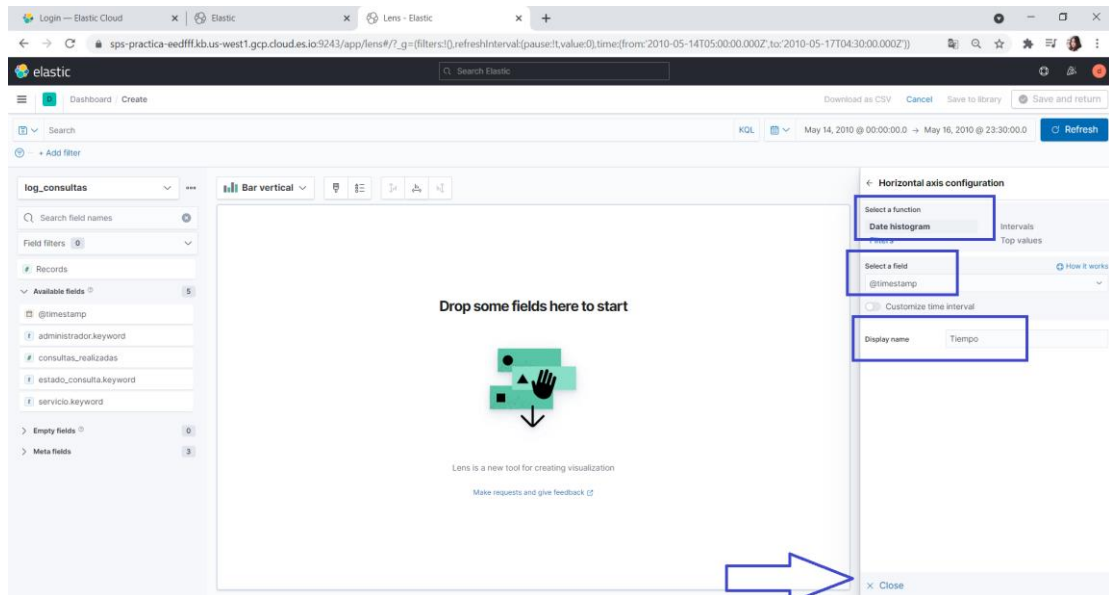
Veremos la siguiente pantalla y seleccionamos la grafica de tipo “Bar vertical”



En el menú derecho, configuramos los valores para el eje horizontal
Modificamos los siguientes valores:

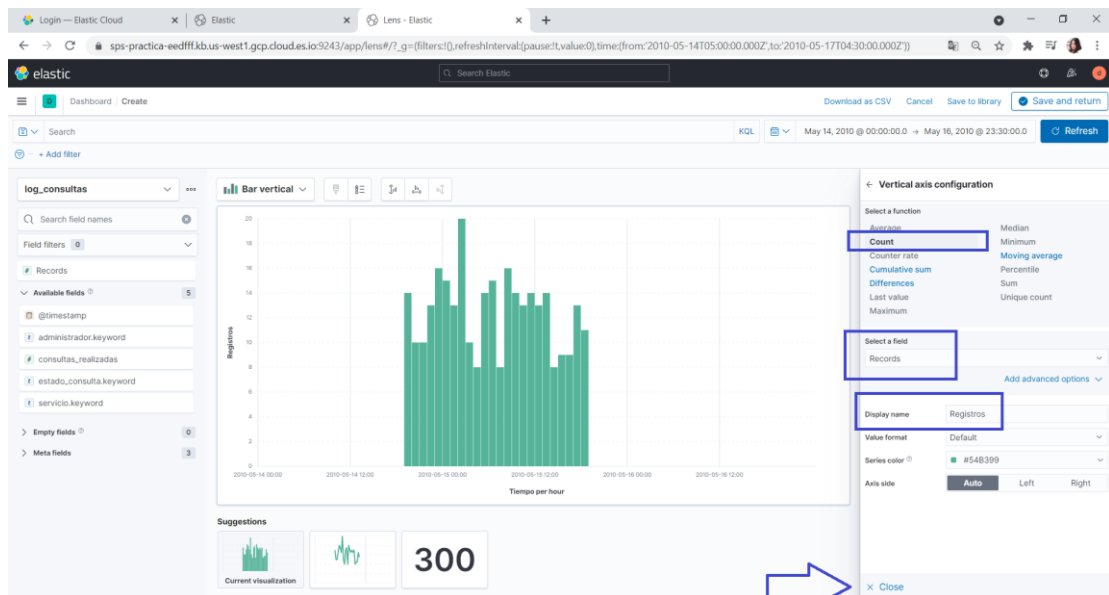
Select a function= *Date Histogram*
Select a field = *@timestamp*
Display name = *Tiempo*

Y damos click en el boton “Close”



En el menú derecho, configuramos los valores para el eje vertical
Modificamos los siguientes valores:
Select a function= Count
Select a field = Records
Display name = Registros

Y damos click en el boton “Close”



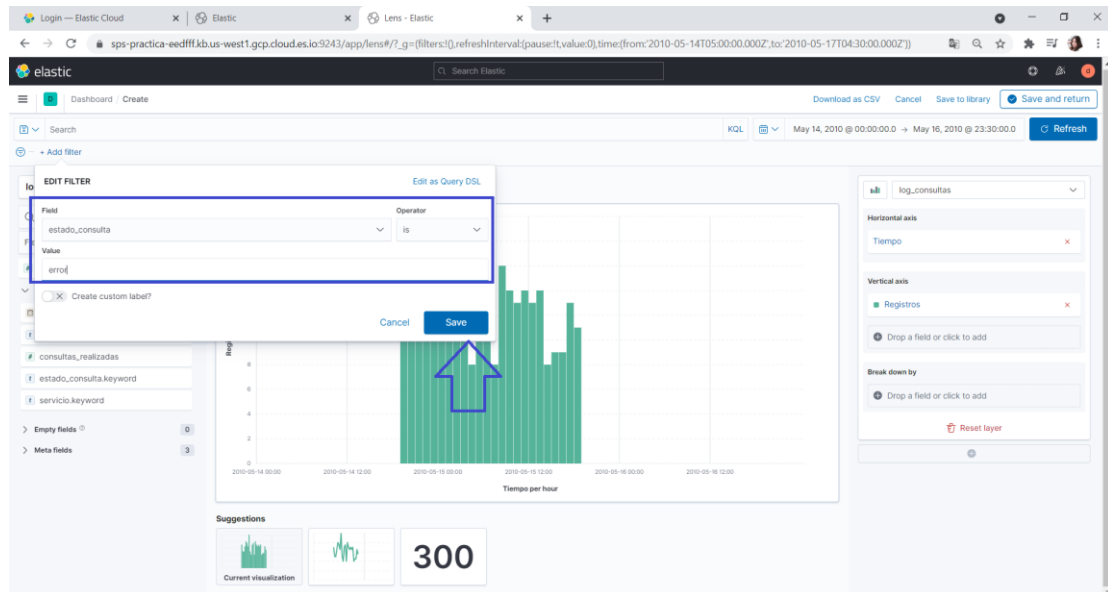
Para configurar los filtros, vamos a la opción Add filter e indicamos los siguientes valores:

Field = estado_consulta

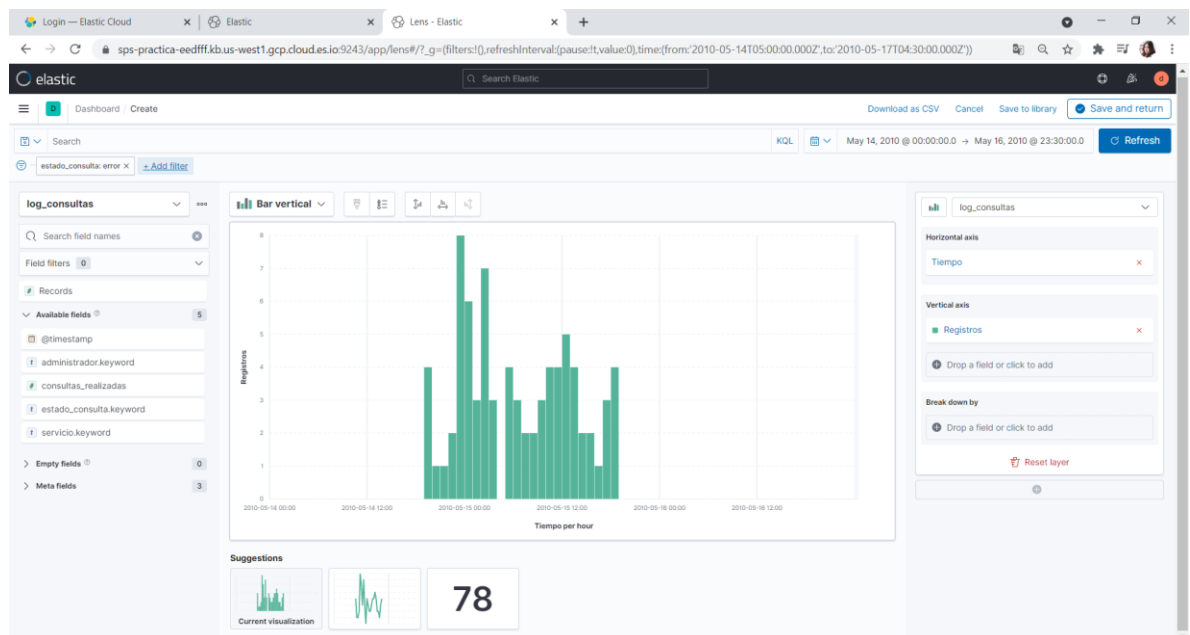
Operator = is

Value = error

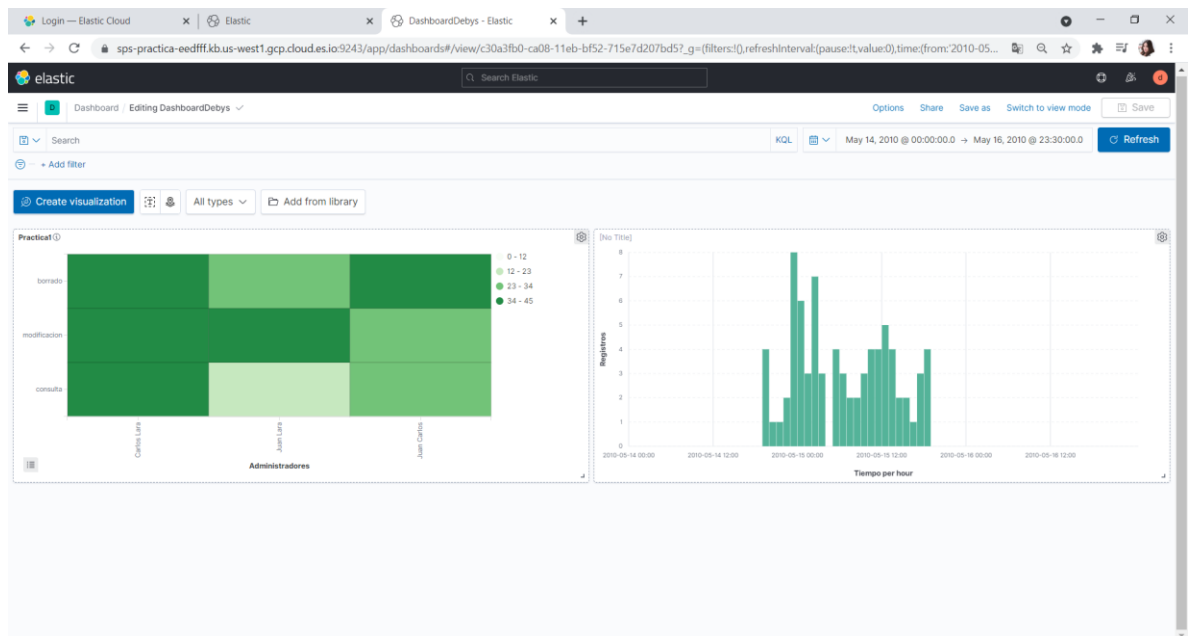
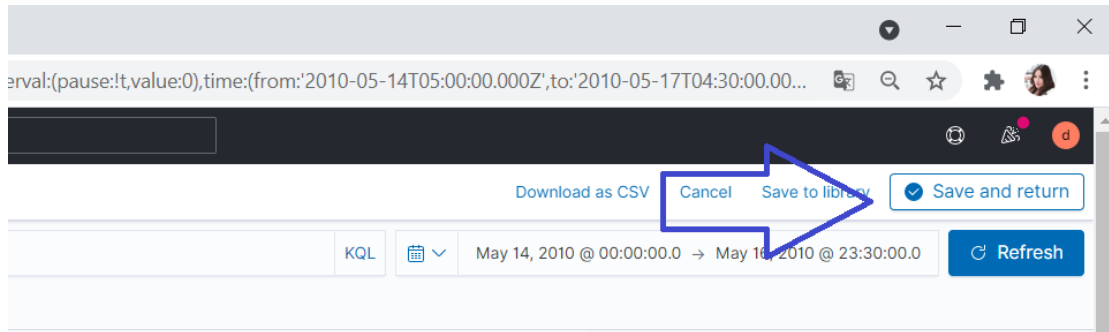
Damos click en el botón “Save”



Damos click en el botón “Refresh” y veremos la siguiente vista:



Damos click en “Save and return” para regresar al dashbord y ver las 2 vistas que creamos



Felices trazos ¡ =)