Kibana & Canvas
Elastic stack

Developed by Yaniv Cohen Devopshift CEO and Head of devops JB

JOHN BRYCE
Leading in IT Education
matrix company

ABOUT ME

Ex CEO , CTO and SVP Engineering for multiple Israeli startups with 25 Years experience in the world of Software architecture , IT and Management.

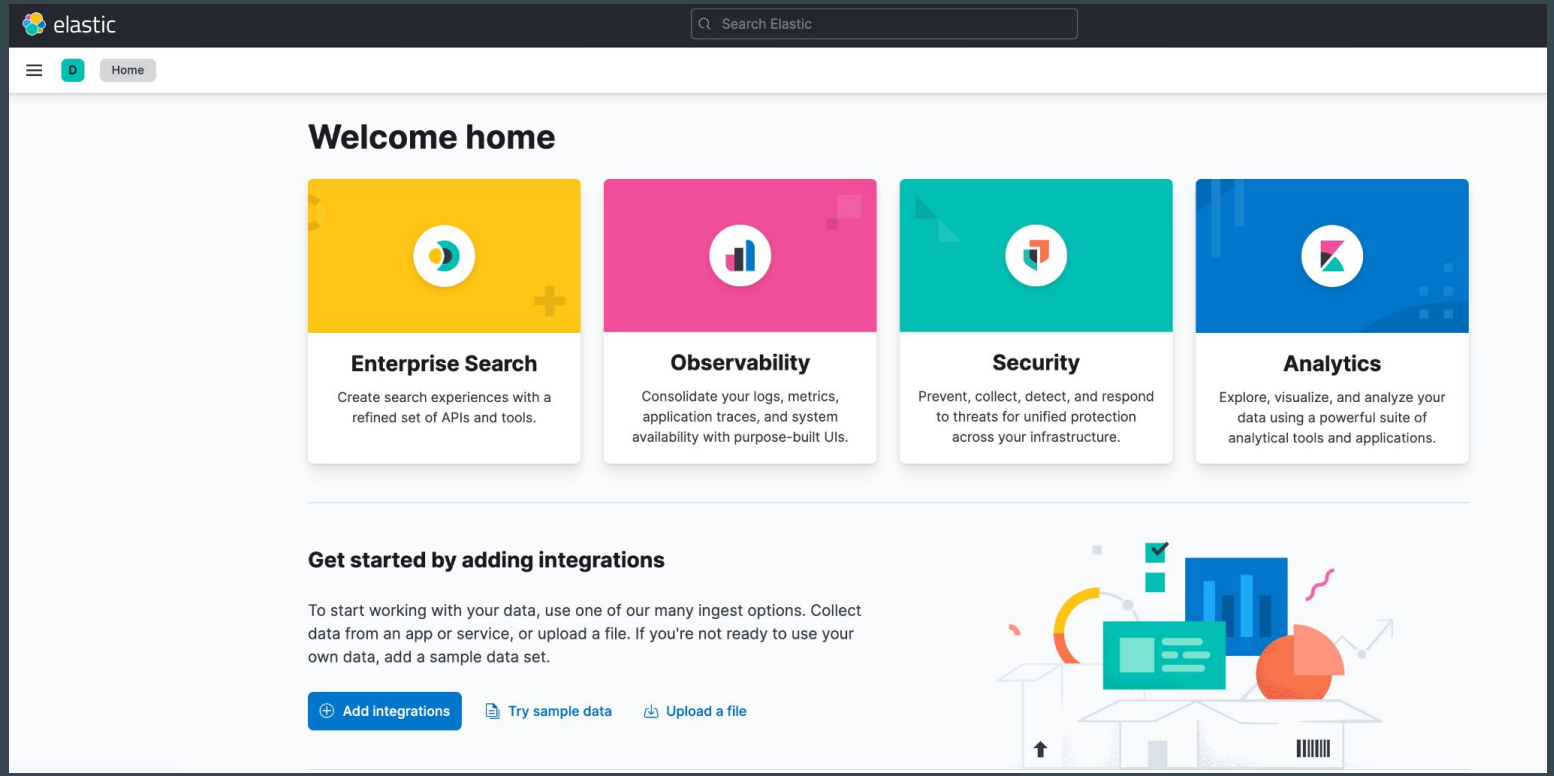
Founder & CEO of :
DevopShift Israel - Devops solutions & consulting
Founder & CEO of of DevopShift Vienna

Im also Acting as Head of Of Devops for John Bryce - Israeli biggest computer academy.
Last but not least a Devops and Microservices design lecturer around the world (14,000 Students in just the last 5 Years).

Linkedin profile:
http://bit.ly/yaniv_linkedin
EMAIL: yaniv@ynot.work

- KIBANA -

KIBANA INTRODUCTION



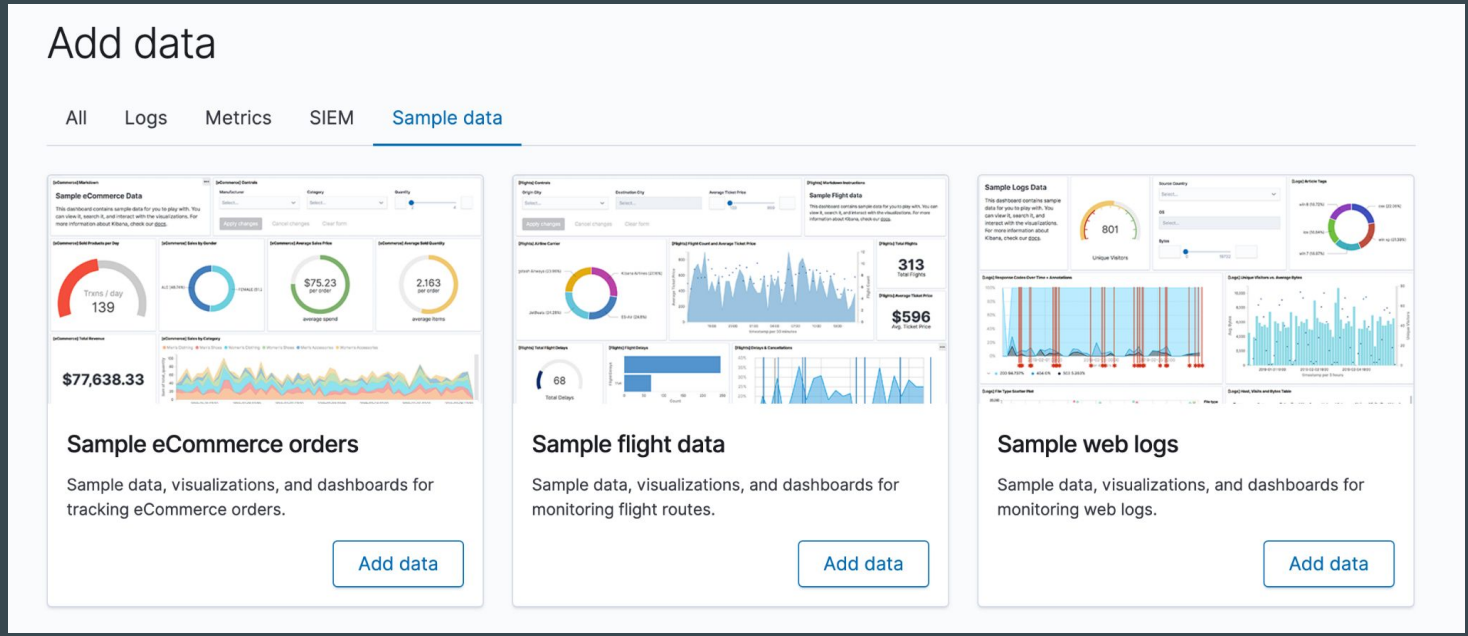
- KIBANA -
Searching with kibana

KIBANA INTRODUCTION

Running our first queries

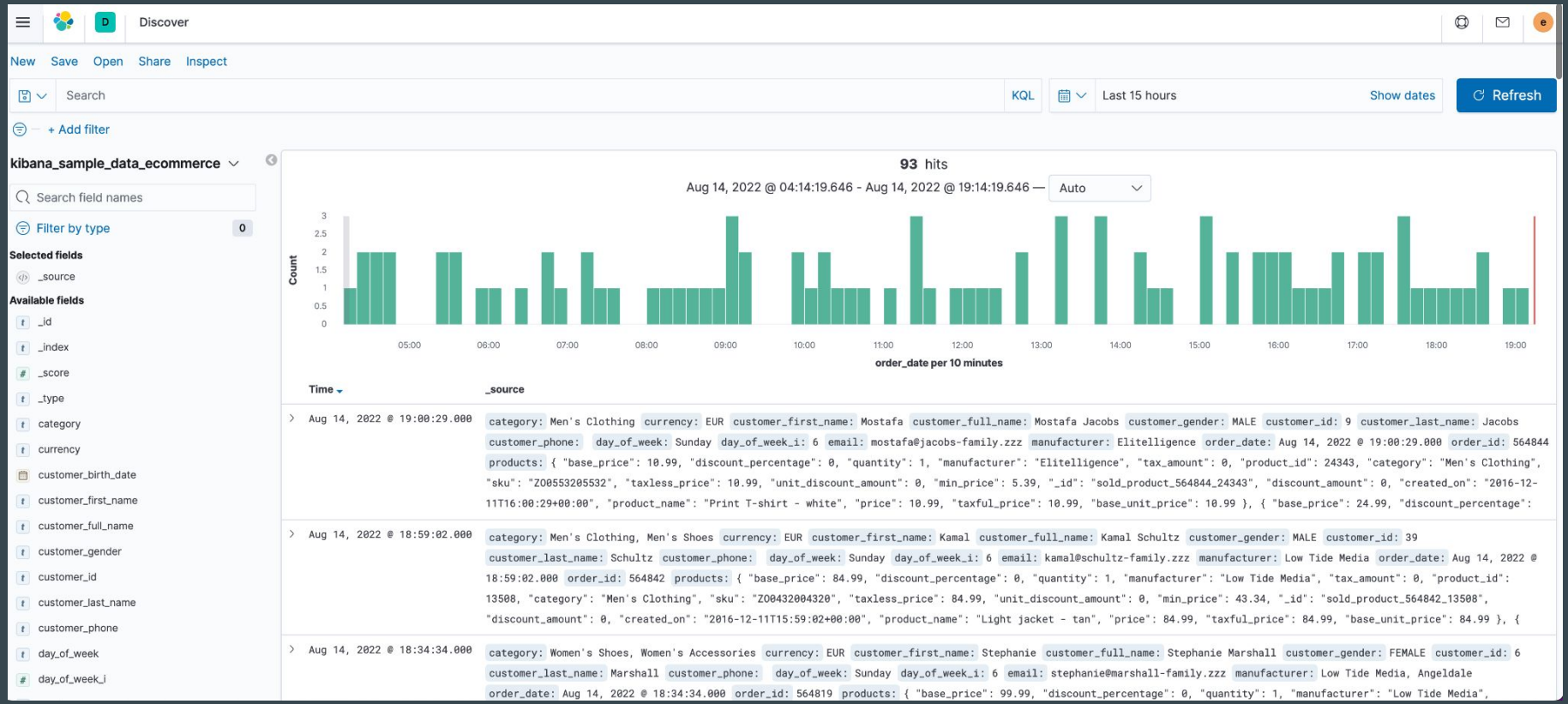
Let's start by adding KIBANA SAMPLE DATA

Add all 3 datasets



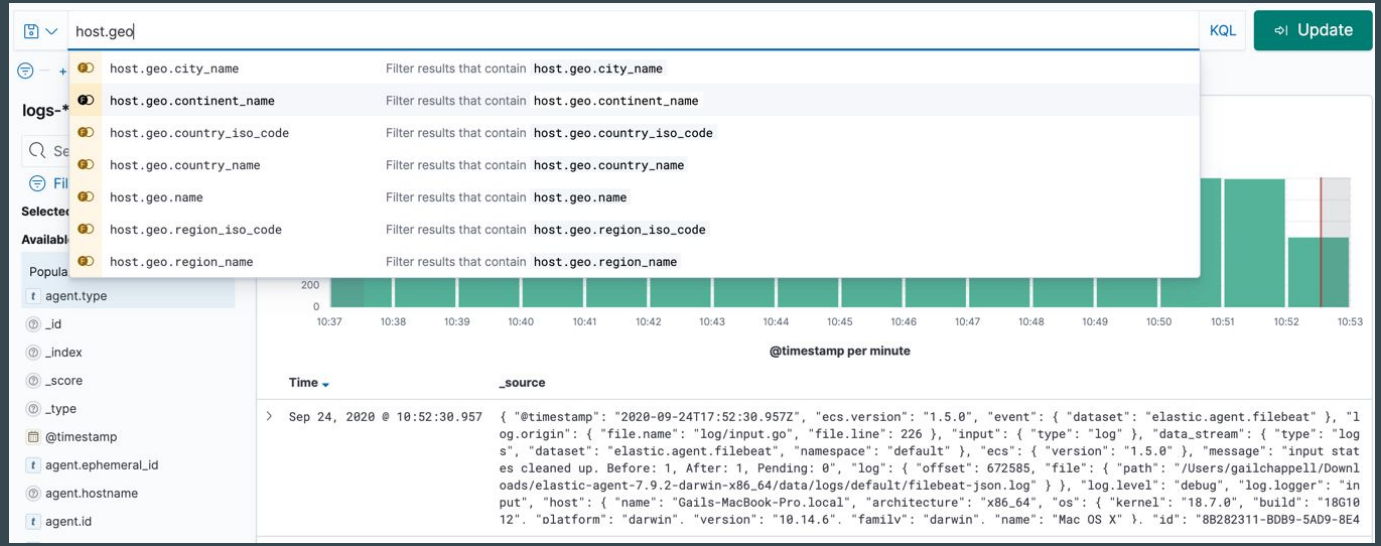
KIBANA INTRODUCTION

Running our first queries using Kibana Discover



KIBANA INTRODUCTION
KQL

The Kibana Query Language (KQL) is a simple syntax for filtering Elasticsearch data using free text search or field-based search. KQL is only used for filtering data, and has no role in sorting or aggregating the data.

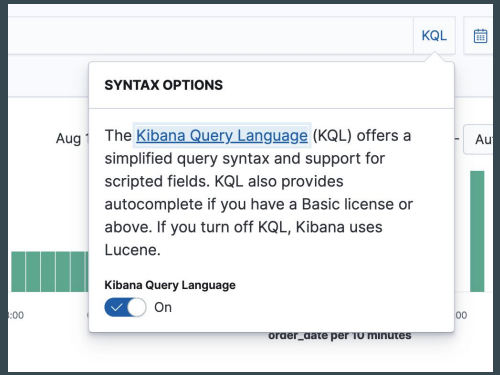


KIBANA INTRODUCTION

Running our first queries

KQL or Lucene

KQL (Kibana Query Language) is a query language available in Kibana, that will be handled by Kibana and converted into Elasticsearch Query DSL. Lucene is a query language directly handled by Elasticsearch. In nearly all places in Kibana, where we can provide a query we can see which one is used by the label on the right of the search box. Clicking on it allows we to disable KQL and switch to Lucene.



KIBANA INTRODUCTION

Running our first queries

KQL VS LUCENE

KQL

- Supports auto completion of fields and values
- Supports searching on scripted fields
- Supports wildcard on field names
- Supports querying for nested fields
- Simpler syntax for some operators
- More resilient in where we can use spaces

LUCENE

- Supports regular expressions
- Supports fuzzy search
- Supports boosting

Which one should we use? Always start with KQL — which is also the default in recent Kibana versions and just fall back to Lucene if we need specific features that are not available in KQL.

KIBANA INTRODUCTION

Running our first queries

PLEASE NOTE: - LUCENE SPACE IN QUERIES

Lucene is rather sensitive to where spaces in the query can be,
For example (using ␣ here to represent a space):

user:eva, user␣:␣eva and user␣:eva are all equivalent, while price:>42 and price:>␣42 are actually searching for different documents. When using Lucene, It's always recommend to not put any spaces around the operators to be safe.

KQL is more resilient to spaces and it doesn't matter where around the operator we'll put spaces.

- KIBANA -
FINDING VALUES

KIBANA INTRODUCTION

Running our first queries

FINDING ANY FIELD MATCHES ANY OF THE WORDS / TERM

The term must appear as it is in the document, in the below example this query won't match documents containing the word "darker"

KQL

```
dark light
```

Lucene

```
dark light
```

Use and/or and parentheses to define that multiple terms need to appear. This query would find all documents that have the term "orange" and either "dark" or "light" (or both) in it.

KQL

```
orange and (dark or light)
ℹ️ Use quotes to search for the word "and"/"or"
"and" "or" xor
```

Lucene

```
⚡ AND/OR must be written uppercase
orange AND (dark OR light)
```

KIBANA INTRODUCTION

Running our first queries

To find values only in specific fields we can put the field name before the value e.g. this query will only find “orange” in the color field.

KQL

```
color : orange
title : our planet or title : dark
```

Lucene

```
color:orange
⚡ Spaces need to be escaped
title:our\ planet OR title:dark
```

Putting quotes around values makes sure they are found in that specific order (“match a phrase”) e.g. if we want to make sure to only find documents containing “our planet” and not “planet our” we’d need the following query:

KQL

```
"our planet"
title : "our planet"
```

Lucene

```
"our planet"
ℹ️ No escaping of spaces in phrases
title:"our planet"
```

KIBANA INTRODUCTION

Running our first queries

Wildcards

We can use the wildcard * to match just parts of a term/word, e.g. this query will find anything beginning with “dark” like “darker”, “darkest”, “darkness”, etc.

KQL `dark*`

Lucene `dark*`

Wildcards can be used anywhere in a term/word. ⚡ Using a wildcard in front of a word can be rather slow and resource intensive for our Elasticsearch — use with care.

KQL `d*k *les`

Lucene `d*k *les`

KIBANA INTRODUCTION

Running our first queries

We can use the * wildcard also for searching over multiple fields in KQL e.g. this query will search “fakestreet” in all fields beginning with “user.address.”.

KQL	user.address.* : fakestreet

Lucene	**Not supported**

Wildcards cannot be used when searching for “phrases” i.e. "our plan*" will not retrieve results containing “our planet”.

Comparing values

Compare numbers or dates. Those operators also work on text/keyword fields, but might behave not very intuitive and thus I'd recommend avoiding usage with text/keyword fields.

KQL

```
price >= 42 and price < 100
time >= "2022-04-10"
```

Lucene

```
price:>=42 AND price:<100
⚡ No quotes around the date in Lucene
time:>=2022-04-10
```

Lucene supports a special range operator to search for a range(besides using comparator operators shown above).

Lucene

```
price:[4000 TO 5000]
ℹ️ Excluding sides of the range using curly braces
price:[4000 TO 5000}
price:{4000 TO 5000}
ℹ️ Use a wildcard for having an open sided interval
price:[4000 TO *]
price:[* TO 5000]
```

KIBANA INTRODUCTION

Running our first queries

Special queries

Find documents in which a specific field exists (i.e. that does have a non null value for that field).

KQL	destination : *
Lucene	_exists_:destination

Querying nested fields is only supported in KQL. The syntax is a bit more complex given the complexity of nested queries. Thus I'd recommend reading the official documentation.

KQL	products:{ name:pencil and price > 10 }
Lucene	Not supported

KIBANA INTRODUCTION

Running our first queries

Lucene has the ability to search for regular expressions. ⚡ This can be rather slow and resource intensive for our Elasticsearch — use with care.

KQL	Not (yet) supported (see #46855)

Lucene	mail:/mailbox\.org$/

Fuzzy search allows searching for strings, that are very similar to the given query.

KQL	Not (yet) supported (see #54343)

Lucene	user:maria~

KIBANA INTRODUCTION

Running our first queries

Hands On LAB

Select one of the indexes in Kibana discover and make sure we use each of the queries options we just learn **on the data of our choosing (and fields).**

- KIBANA - VISUALIZATION

KIBANA LENS

KIBANA INTRODUCTION

Visualizing our data

Kibana Lens is an easy-to-use, intuitive UI that simplifies the process of data visualization through a drag-and-drop experience. Whether we're exploring billions of logs or spotting trends from our website traffic, Lens gets we from data to insights in just a few clicks — no prior experience in Kibana required.

KIBANA INTRODUCTION

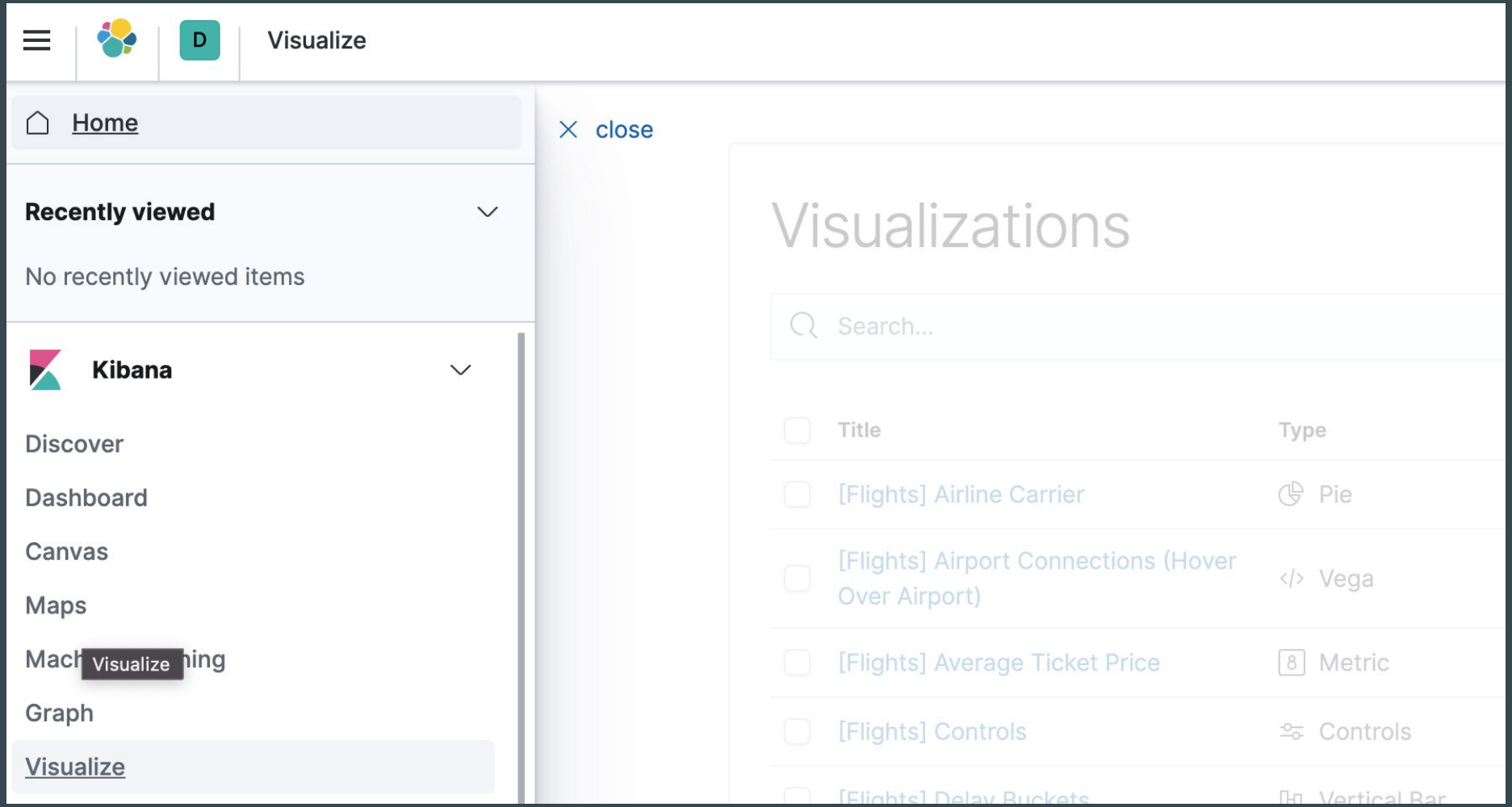
Visualizing our data

A few key benefits of Kibana Lens include:

- Convenient features for fields such as:
  - Showing their distribution of values
  - Searching fields by name for quickly tracking down the data we want
- Quick aggregation metrics like:
  - min, max, average, sum, count, and unique count
- Switching between multiple chart types after the fact, such as:
  - bar, area, line, and stacked charts
- The ability to drag and drop any field to get it immediately visualized or to breakdown the existing chart by its values
- Automatic suggestions on other possible visualization types
- Showing the raw data in data tables
- Combining the visualization with searching and filtering capabilities
- Combining data from multiple index patterns
- And quickly saving the visualization allowing for easy dashboard composition

PREPARATION

Make sure we've added
SERVER LOGS & FLIGHT SAMPLE DATA

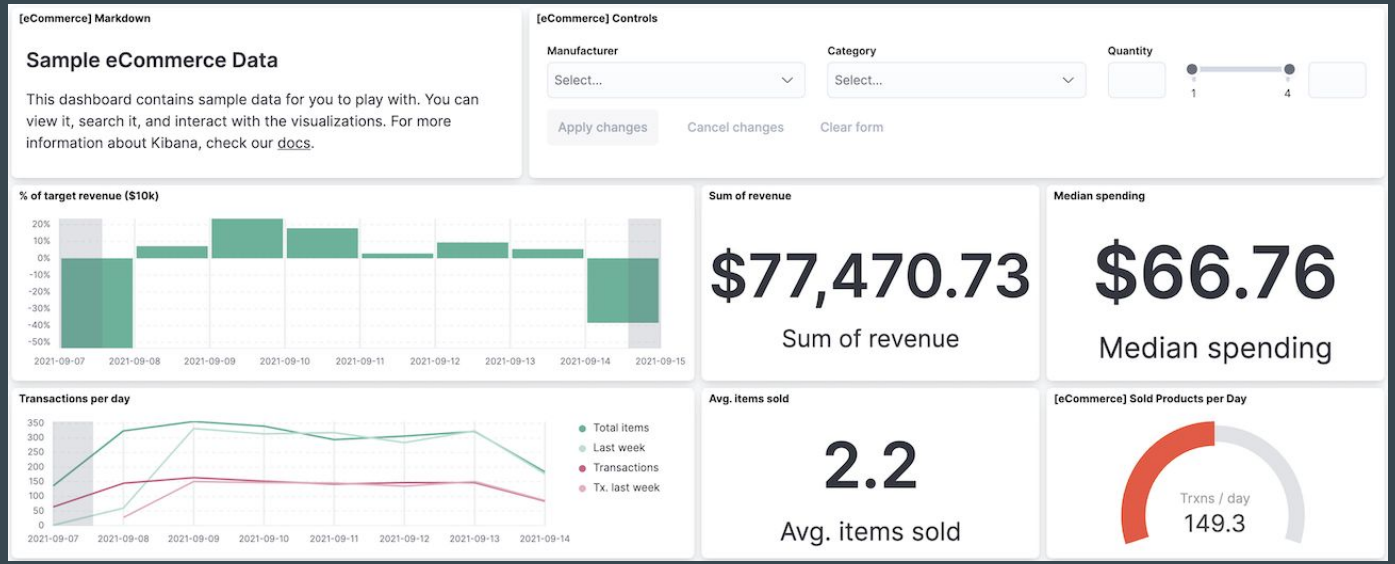

Visualize
Home
Recently viewed
No recently viewed items
Kibana
Discover
Dashboard
Canvas
Maps
Visualize
Graph
Visualize
close
Visualizations
Search...
Title
Type
[Flights] Airline Carrier
Pie
[Flights] Airport Connections (Hover Over Airport)
Vega
[Flights] Average Ticket Price
Metric
[Flights] Controls
Controls

KIBANA INTRODUCTION

Visualizing our data

Visualize our data with dashboards.

The best way to understand our data is to visualize it. With dashboards, we can turn our data from one or more data views into a collection of panels that bring clarity to our data, tell a story about our data, and allow us to focus on only the data that's important to us.



KIBANA INTRODUCTION

Visualizing our data

Visualize our data with dashboards.

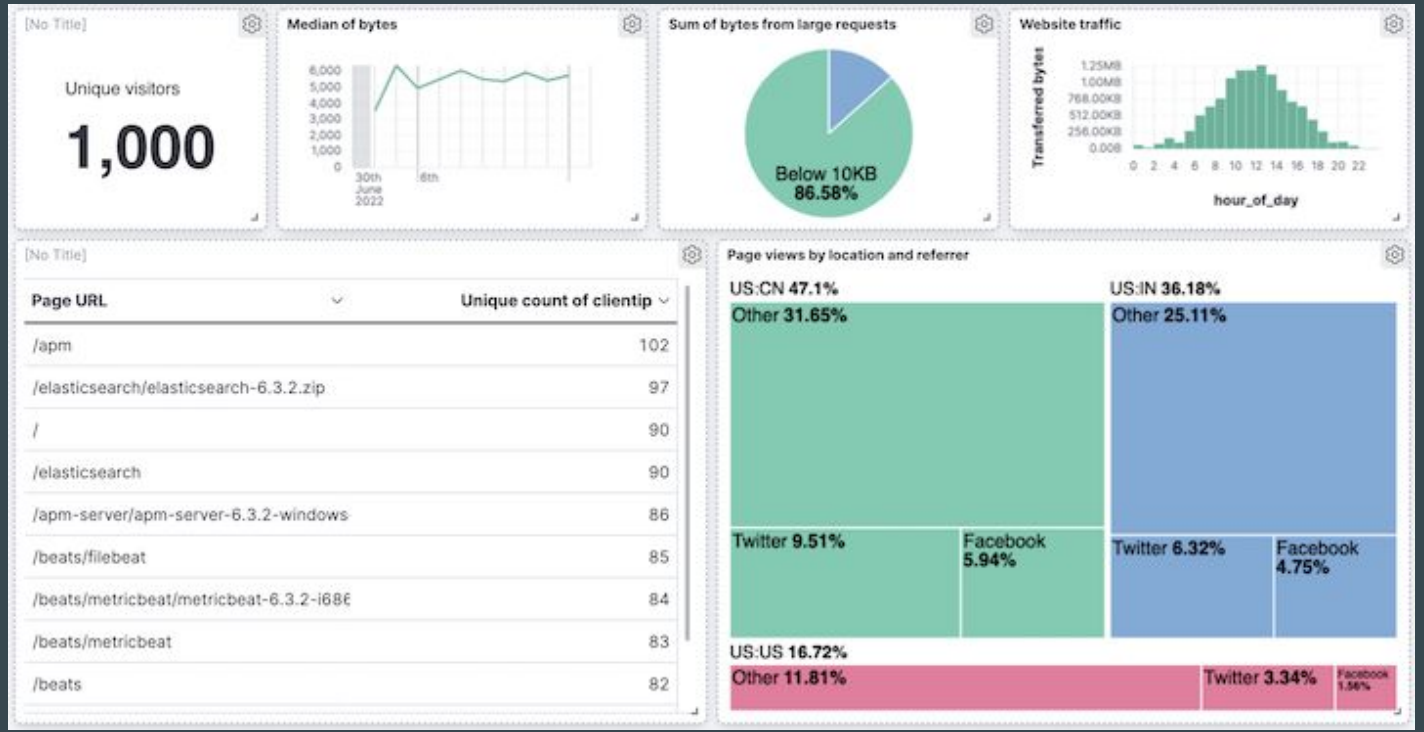
Panels display our data in charts, tables, maps, and more, which allow we to compare our data side-by-side to identify patterns and connections. Dashboards support several types of panels to display our data, and several options to create panels.

- KIBANA -
CREATING OUR FIRST DASHBOARD

KIBANA INTRODUCTION

Visualizing our data

Follow through : when we're done, we'll have a complete overview of the sample web logs data.



KIBANA INTRODUCTION

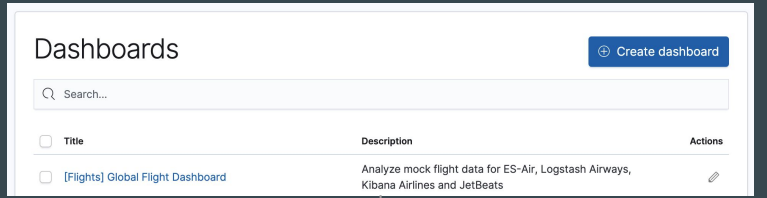
Visualizing our data

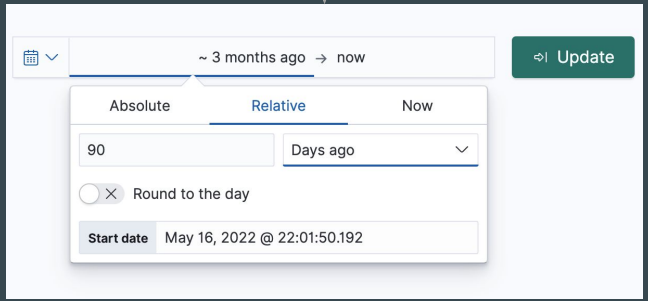
Add the data and create the dashboard

Add the sample web logs data, and create and set up the dashboard.

In case it was missed:

1. Go to the Home page, then click Try sample data.
2. **On the Sample web logs card, click Add data.**





Create the dashboard where you'll display the visualization panels.

1. Open the main menu, then click Dashboard.
2. Click Create dashboard.
3. Set the time filter to Last 90 days.

KIBANA INTRODUCTION

Visualizing our data

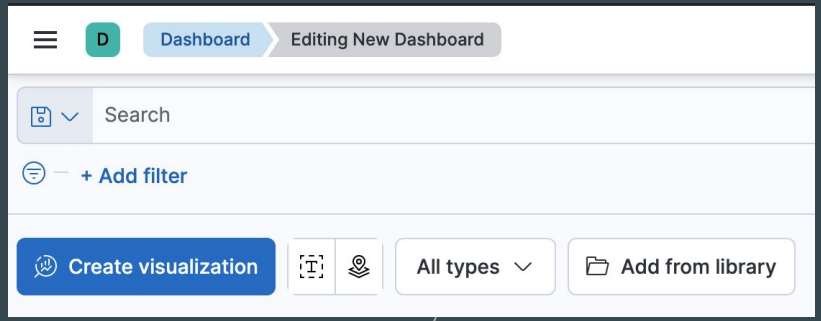
Open the visualization editor and get familiar with the data

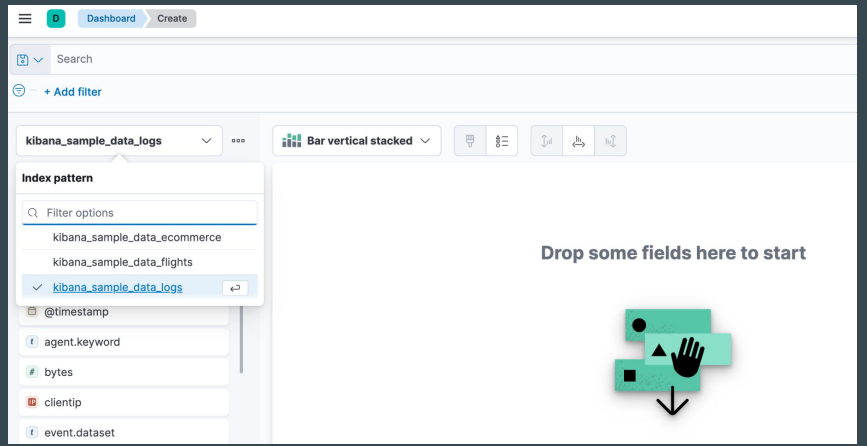
Open the visualization editor, then make sure the correct fields appear.

1. On the dashboard, click Create visualization.
2. Make sure the kibana_sample_data_logs index appears.

To create the visualizations in this tutorial, you'll use the following fields:

- Records
- timestamp
- bytes
- clientip
- Referrer (Defaulted to keyword - 7.17)





KIBANA INTRODUCTION

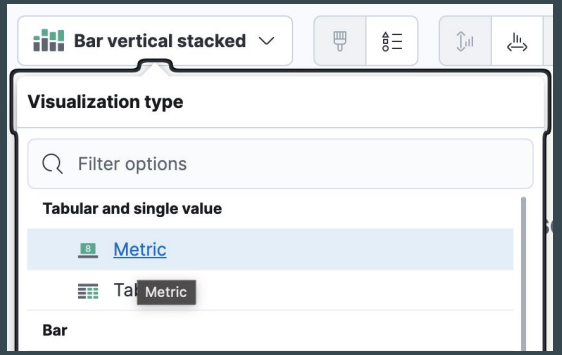
Visualizing our data

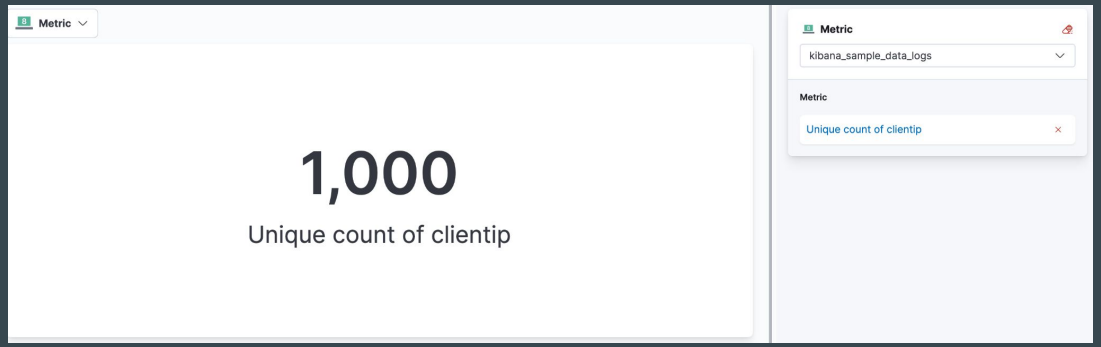
Create our first visualization

Pick a field you want to analyze, such as clientip. To analyze only the clientip field, use the Metric visualization to display the field as a number.

The only number function that you can use with clientip is Unique count, also referred to as cardinality, which approximates the number of unique values.

1. Open the Visualization type dropdown, then select Metric.
2. From the Available fields list, drag clientip to the workspace or layer pane.

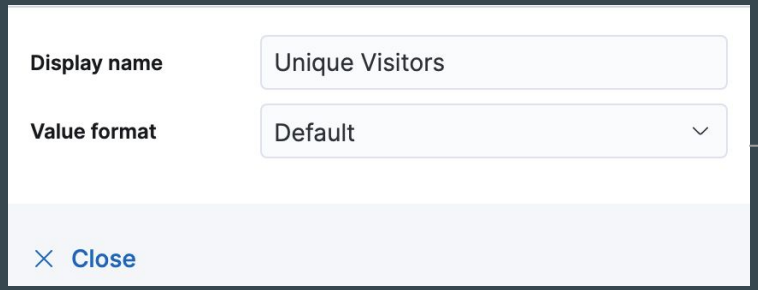


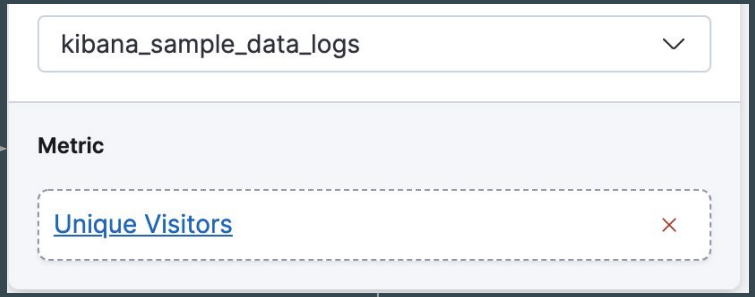


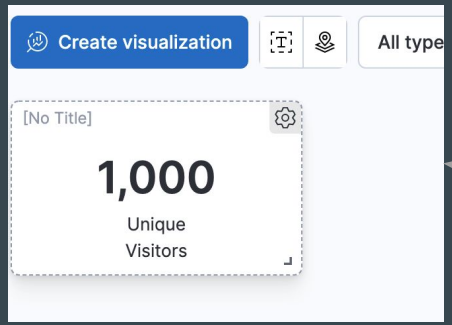
KIBANA INTRODUCTION

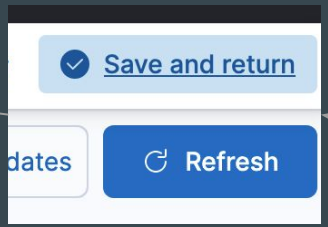
Visualizing our data

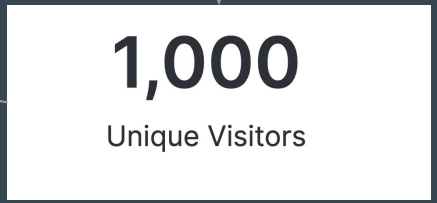
1. In the layer pane, click Unique count of clientip.
   a. In the Display name field, enter Unique visitors.
   b. Click Close.











KIBANA INTRODUCTION

Visualizing our data

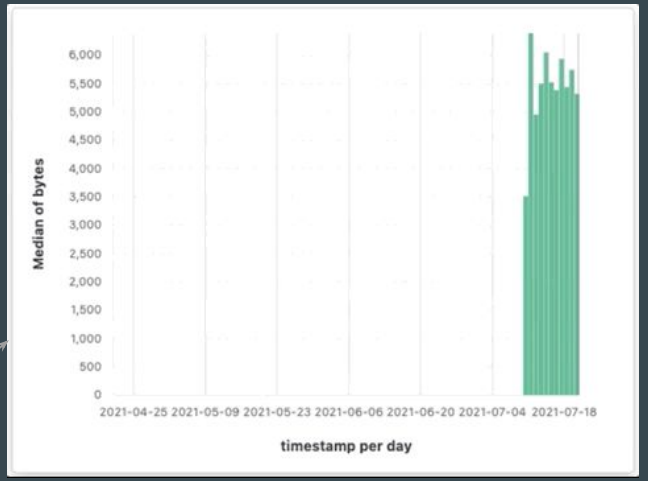
View a metric over time

There are two shortcuts you can use to view metrics over time. When you drag a numeric field to the workspace, the visualization editor adds the default time field from the index pattern. When you use the Date histogram function, you can replace the time field by dragging the field to the workspace.

To visualize the bytes field over time:

1. On the dashboard, click Create visualization.
2. From the Available fields list, drag **bytes** to the workspace.

   The visualization editor creates a bar chart with the timestamp and Median of bytes fields.
3. To zoom in on the data, click and drag your cursor across the bars.

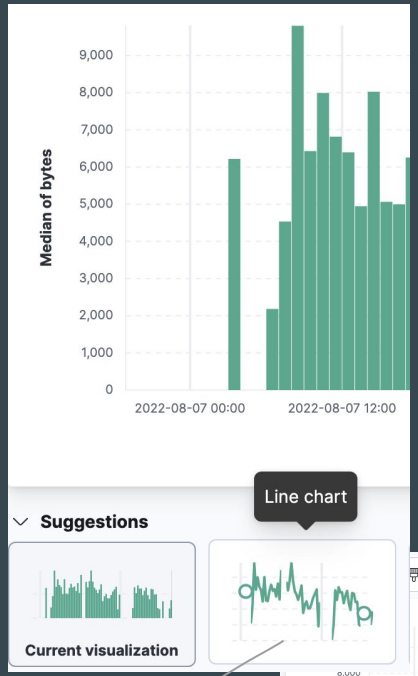


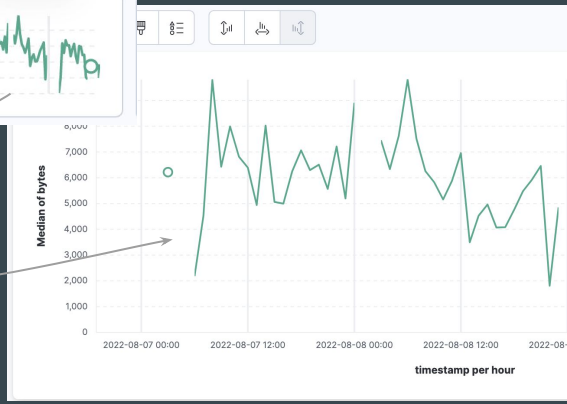
KIBANA INTRODUCTION

Visualizing our data

To emphasize the change in Median of bytes over time, change the visualization type to Line with one of the following options:

- In the Suggestions, click the line chart.
- In the editor toolbar, open the Visualization type dropdown, then select Line.
- In the layer pane, open the Layer visualization type menu, then click Line.





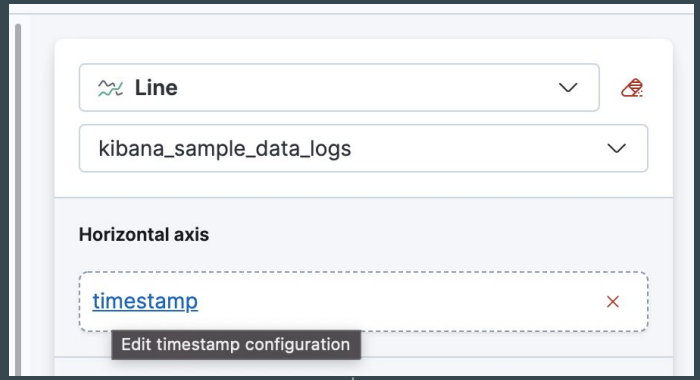
KIBANA INTRODUCTION

Visualizing our data

To increase the minimum time interval:

1. In the layer pane, click timestamp.
2. Select Customize time interval.
3. Change the Minimum interval to 1 days, then click Close.

You can increase and decrease the minimum interval, but you are unable to decrease the interval below the Advanced Settings.

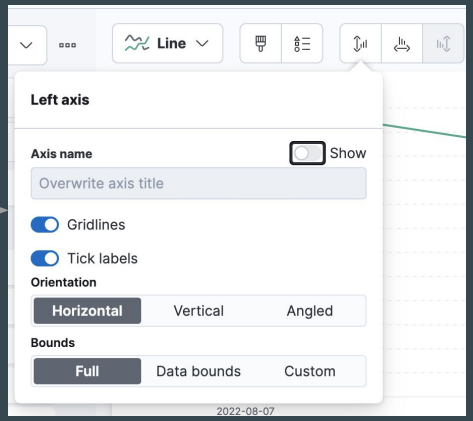


KIBANA INTRODUCTION

Visualizing our data

To save space on the dashboard, hide the axis labels.

1. Open the Left axis menu, then deselect Show.



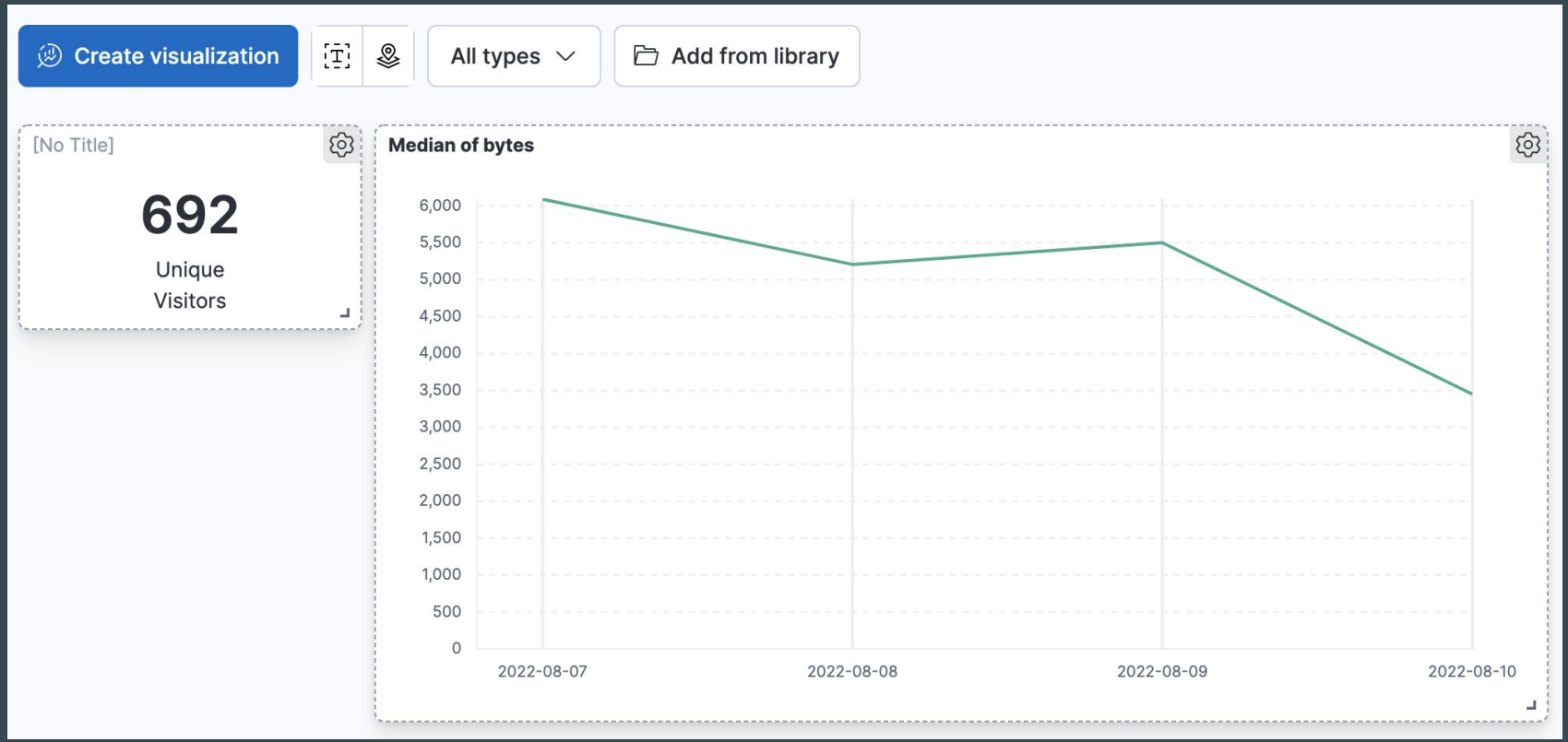
2. Click Sa**ve and return** to dashboard

Since you removed the axis labels, add a panel title:

1. Open the panel menu, then select Edit panel title.
2. In the Panel title field, enter **Median of bytes**, then click Save.

KIBANA INTRODUCTION

Visualizing our data

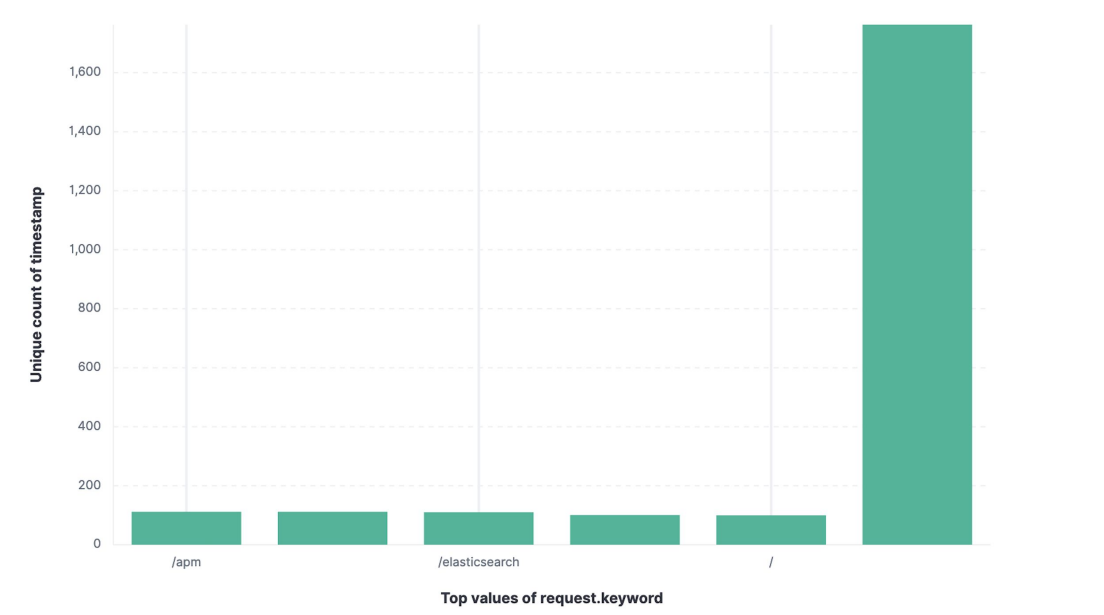


KIBANA INTRODUCTION

Visualizing our data

View the top values of a field

Create a visualization that displays the most frequent values of **request.keyword** on your website, ranked by the unique visitors.



KIBANA INTRODUCTION

Visualizing our data

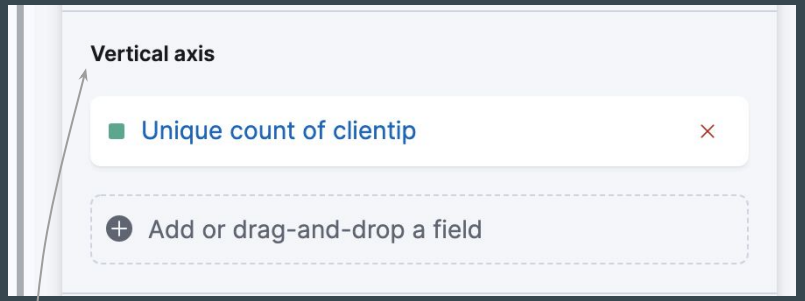
To create the visualization, use **Top values of request.keyword** ranked by **Unique count of clientip**, instead of being ranked by **Count of records**.

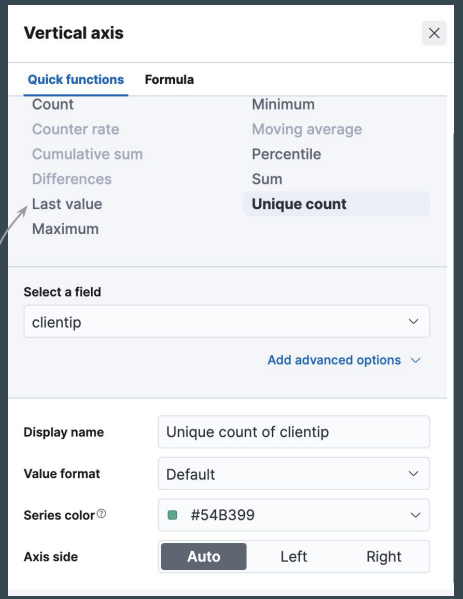
The Top values function ranks the unique values of a field by another function. The values are the most frequent when **ranked by a Count function**, and the largest when ranked by the Sum function.

1. On the dashboard, click Create visualization.
2. From the Available fields list, drag **clientip** to the **Vertical axis** field in **the layer pane.**

   The visualization editor automatically applies the **Unique count function**. If you drag clientip to the workspace, the editor adds the field to the incorrect axis.

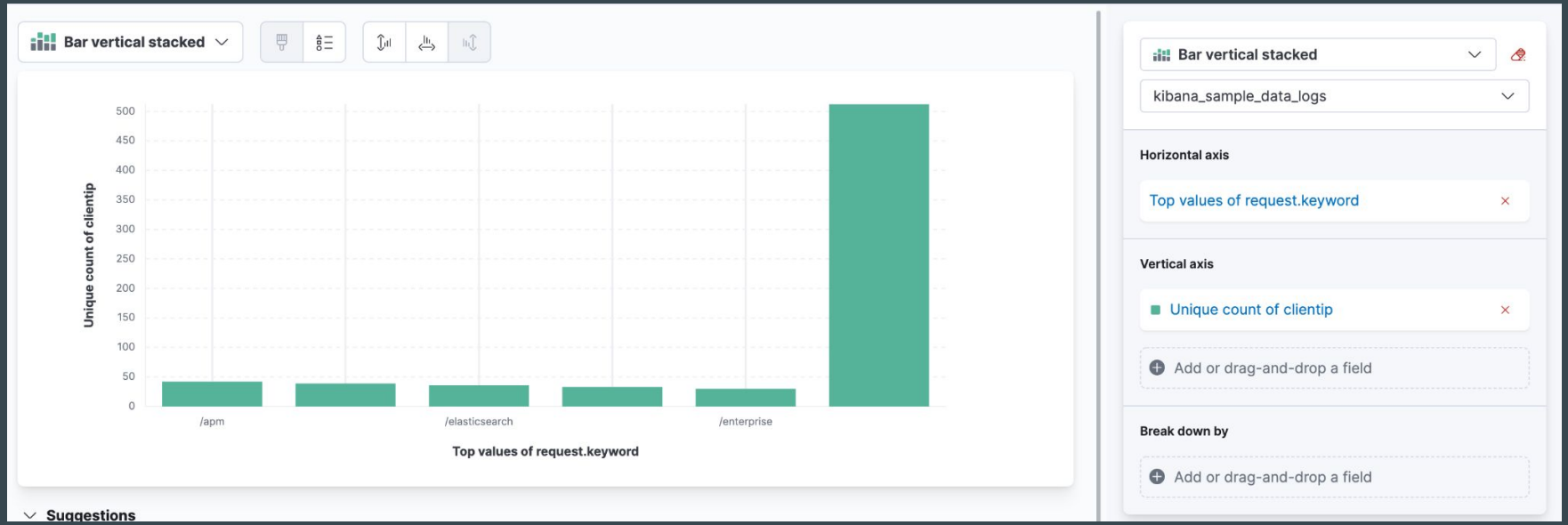
3. Drag request.keyword to the workspace.





KIBANA INTRODUCTION

Visualizing our data



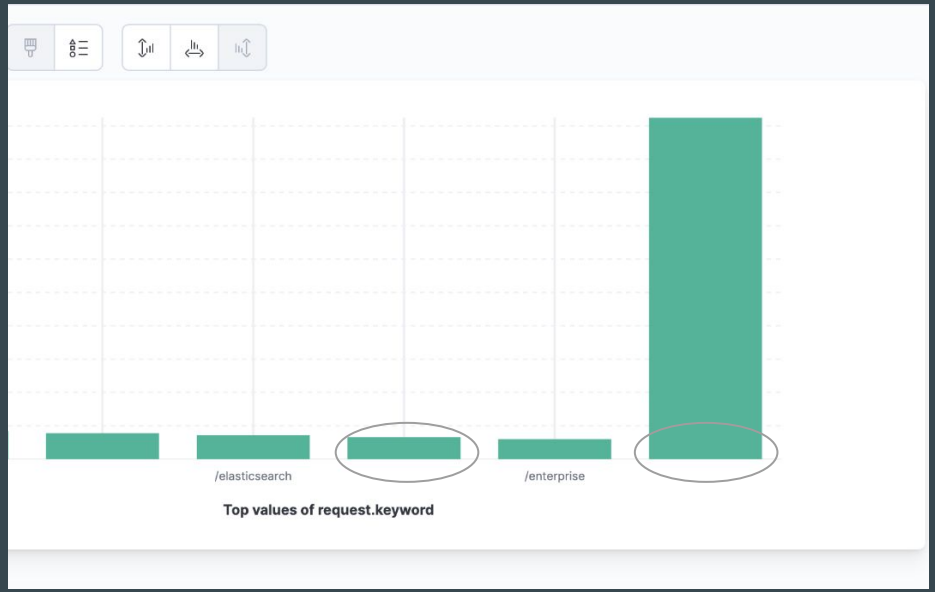
KIBANA INTRODUCTION

Visualizing our data

When you drag a text or IP address field to the workspace, the editor adds the Top values function ranked by **Count of records** to show the most frequent values.

NOTE

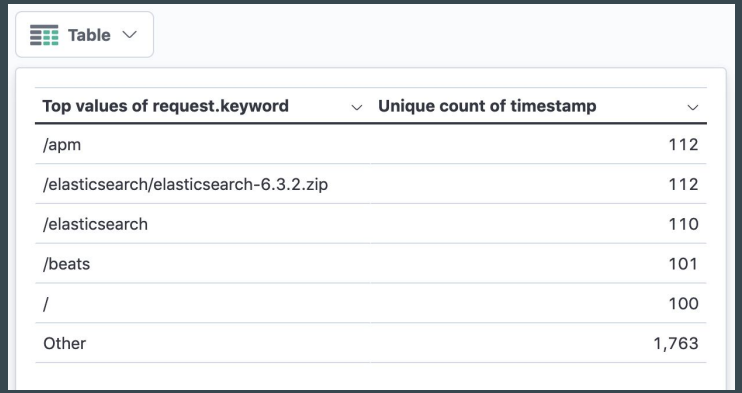
The chart labels are unable to display because the **request.keyword** field contains long text fields. You could use one of the **Suggestions**, but the suggestions also have issues with long text. The best way to display long text fields is with the **Table visualization**.



KIBANA INTRODUCTION

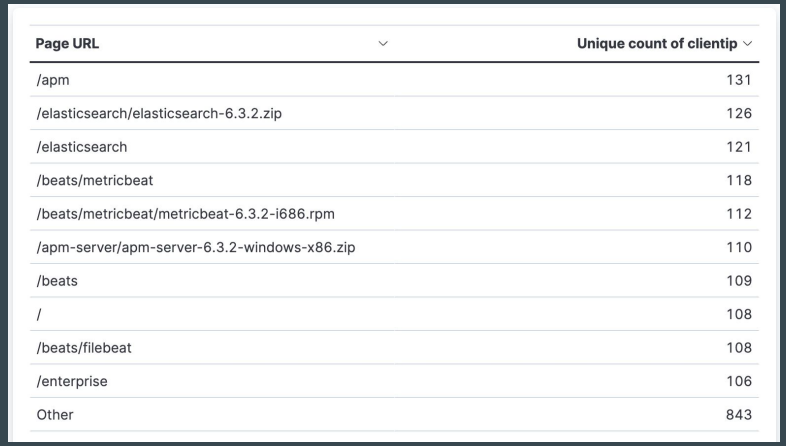
Visualizing our data

1. Open the Visualization type dropdown, then select Table.



2. In the layer pane, click Top values of request.keyword.

   a. In the Number of values field, enter 10.
   b. In the Display name field, enter Page URL.
   c. Click Close.



3. Click Save and return.

Since the table columns are labeled, you do not need to add a panel title.

- LAB -
CONTINUE ON YOUR OWN

KIBANA INTRODUCTION

Visualizing our data

View the distribution of a number field

The distribution of a number can help you find patterns. For example, you can analyze the website traffic per hour to find the best time for routine maintenance.

1. On the dashboard, click Create visualization.
2. From the Available fields list, drag bytes to Vertical axis field in the layer pane.
3. In the layer pane, click Median of bytes.
    a. Click the Sum function.
    b. In the Display name field, enter Transferred bytes.
    c. From the Value format dropdown, select Bytes (1024), then click Close.
4. From the Available fields list, drag hour_of_day to Horizontal axis field in the layer pane.
5. In the layer pane, click hour_of_day, then slide the Intervals granularity slider until the horizontal axis displays hourly intervals.

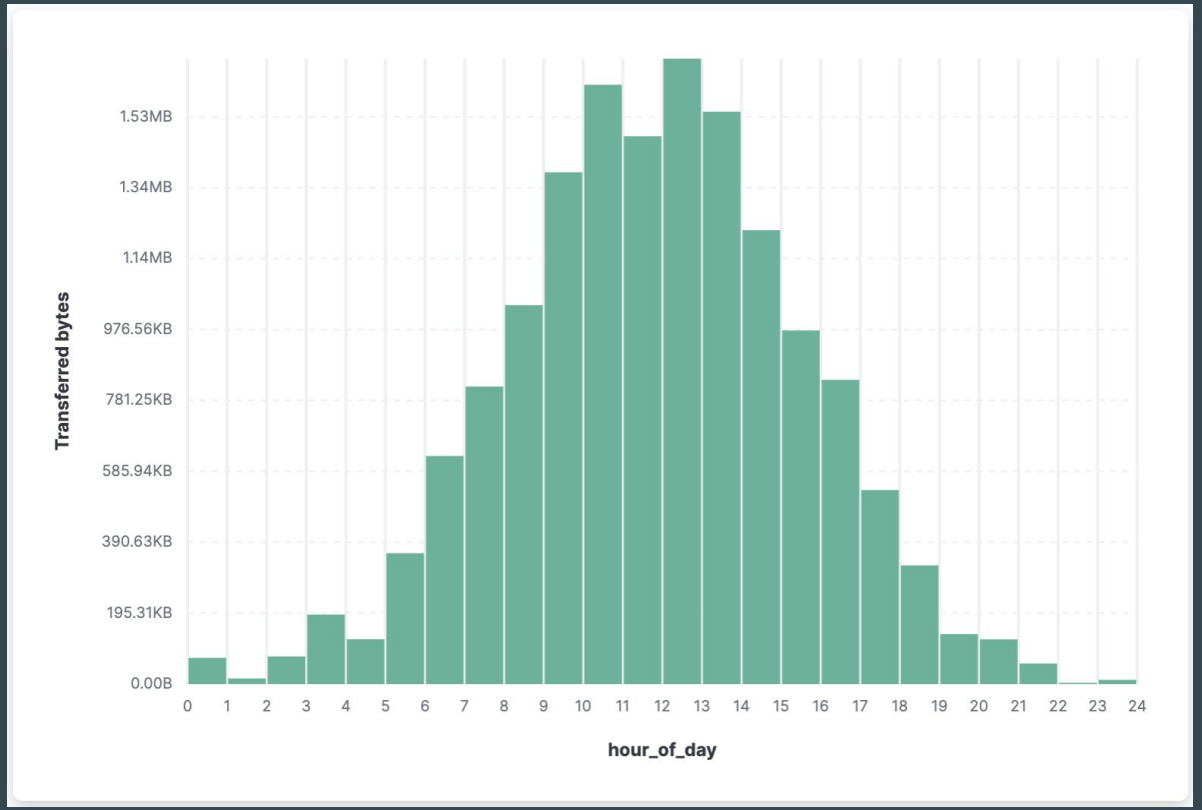
KIBANA INTRODUCTION

Visualizing our data

Click Save and return.
Add a panel title:

Open the panel menu, then select Edit panel title.
In the Panel title field, enter Website traffic, then click Save.

Final Results



KIBANA INTRODUCTION

Visualizing our data

Create a multi-level chart

Table and Proportion visualizations support multiple functions. For example, to create visualizations that break down the data by website traffic sources and user geography, apply the Filters and Top values functions.

1. On the dashboard, click Create visualization.
2. Open the Visualization type dropdown, then select Treemap.
3. From the Available fields list, drag Records to the Size by field in the layer pane.
4. In the editor, click Add or drag-and-drop a field for Group by.

KIBANA INTRODUCTION

Visualizing our data

Create a filter for each website traffic source:

1. From Select a function, click Filters.
2. Click All records, enter the following in the query bar, then press Return:

- KQL — referer : *facebook.com*
- Label — Facebook

3. Click Add a filter, enter the following in the query bar, then press Return:

- KQL — referer : *twitter.com*
- Label — Twitter

4. Click Add a filter, enter the following in the query bar, then press Return:

- KQL — NOT referer : *twitter.com* OR NOT referer: *facebook.com*
- Label — Other

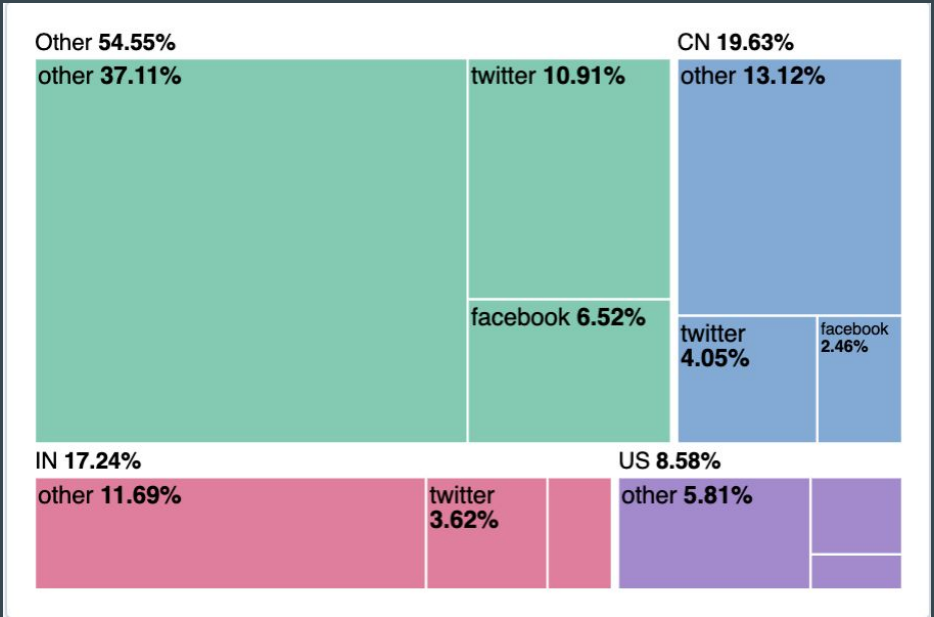
Click Close.

KIBANA INTRODUCTION

Visualizing our data

Add the user geography grouping:

1. From the Available fields list, drag geo.srcdest to the workspace.
2. To change the Group by order, drag Top values of geo.srcdest in the layer pane so that appears first.



KIBANA INTRODUCTION

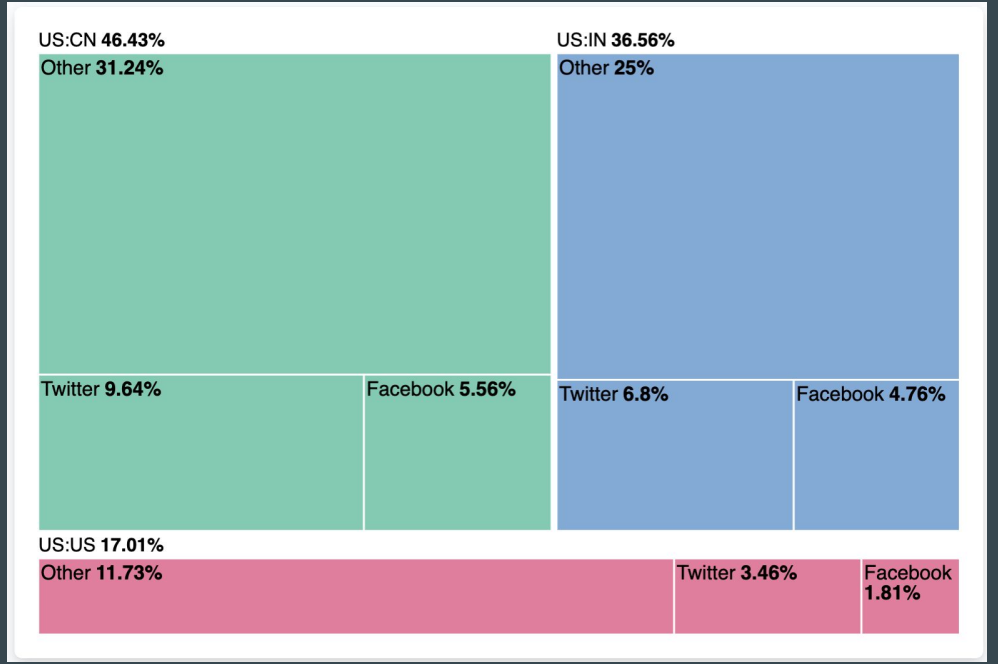
Visualizing our data

Remove the documents that do not match the filter criteria:

1. In the layer pane, click Top values of geo.srcdest.
2. Click Advanced, then deselect Group other values as "Other", the click Close.
3. Click Save and return.

Add a panel title:

1. Open the panel menu, then select Edit panel title.
2. In the Panel title field, enter Page views by location and referrer, then click Save.



KIBANA INTRODUCTION

Visualizing our data

Add on your own
PIE that needs to have the following:

Bytes by large filter (from X to *)

KIBANA INTRODUCTION

Visualizing our data

Arrange the dashboard panels

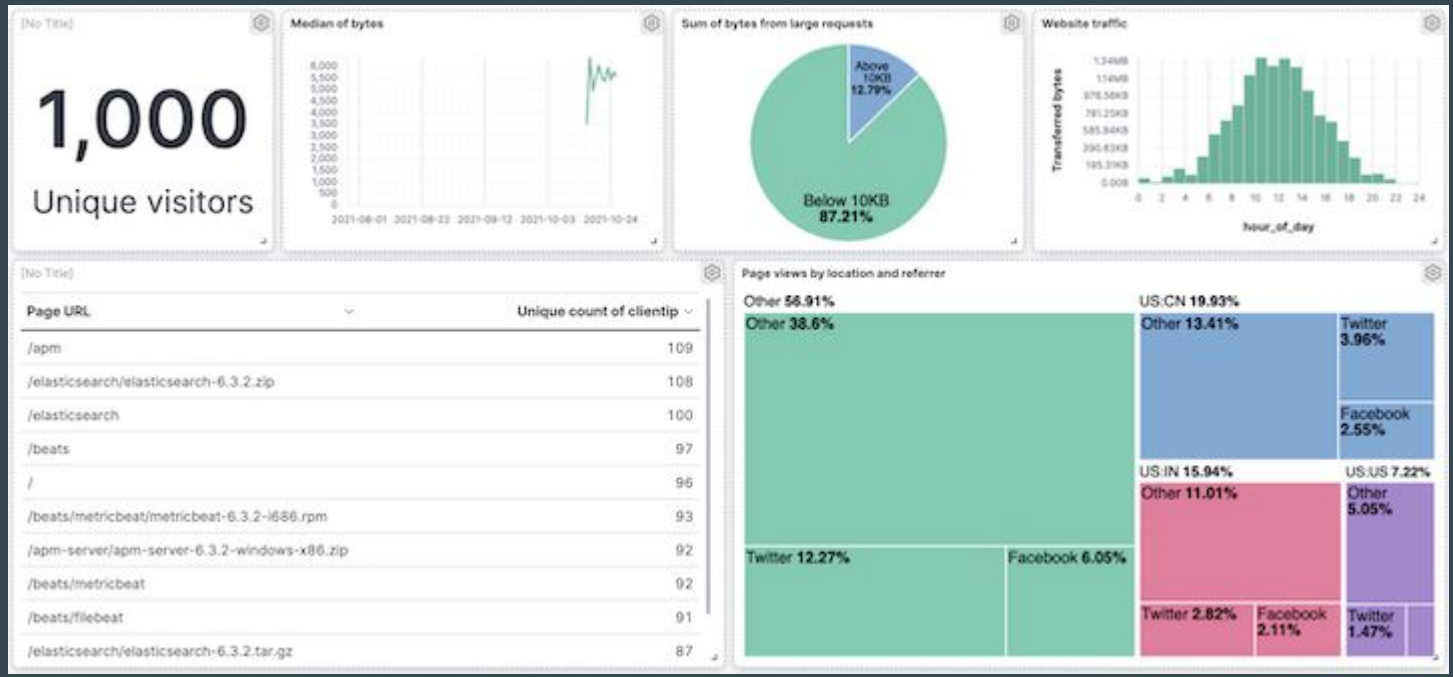
Resize and move the panels so they all appear on the dashboard without scrolling.

Decrease the size of the following panels, then move the panels to the first row:

- Unique visitors
- Median of bytes
- Sum of bytes from large requests
- Website traffic

KIBANA INTRODUCTION

Visualizing our data



KIBANA INTRODUCTION

Visualizing our data

Save the dashboard

Now that you have a complete overview of your web server data, save the dashboard.

1. In the toolbar, click Save.
2. On the Save dashboard window, enter Logs dashboard in the Title field.
3. Select Store time with dashboard.
4. Click Save.
5. Send me the link to your Dashboard

- LAB -
END

KIBANA INTRODUCTION

Visualizing our data

HANDS ON

KIBANA INTRODUCTION

Visualizing our data

We will start by installing metricbeat on our remote linux machines and run some stress test on it for the purpose of creating diverse statistics of our system performance.

Installing metricbeat using YUM:

Adding elasticsearch repo

sudo vim /etc/yum.repos.d/elastic.repo (and paste the following)

```
[elastic-7.x]
name=Elastic repository for 7.x packages
baseurl=https://artifacts.elastic.co/packages/7.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md
```

KIBANA INTRODUCTION

Visualizing our data

Now we are ready to install and configure it

Installing metricbeat using YUM:

sudo yum install -y metricbeat

Configure auth:

sudo vim /etc/metricbeat/metricbeat.yml

Uncomment the user/pass

```
# Authentication credentials - either API key or username/password.
#api_key: "id:api_key"
username: "elastic"
password: "changeme"
```

KIBANA INTRODUCTION

Visualizing our data

Now we are ready to install and configure it

Start our metricbeat service and validate logs

```
sudo service metricbeat start
```

Validate it works

```
sudo tail -f /var/log/metricbeat/metricbeat
```

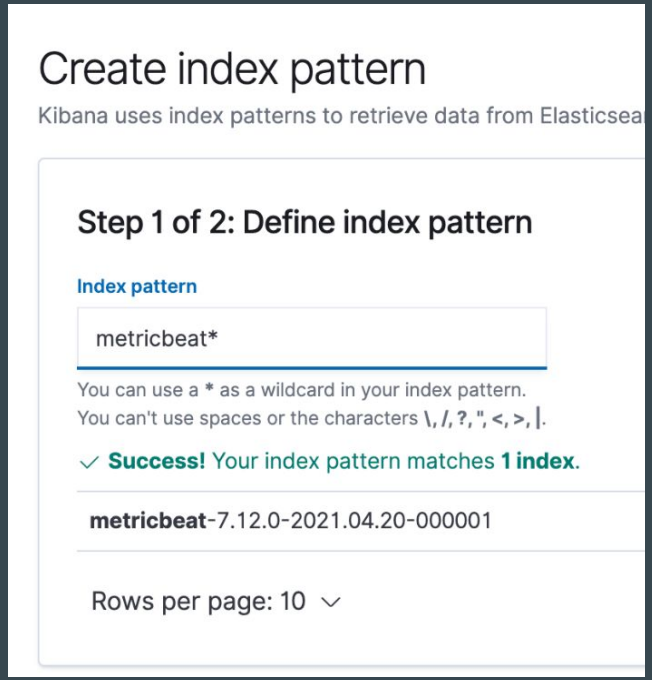
```
2021-04-20T08:14:43.518Z INFO [publisher_pipeline_output] pipeline/output.go:151 Connection to backoff(elasticsearch(http://localhost:9200)) established
```

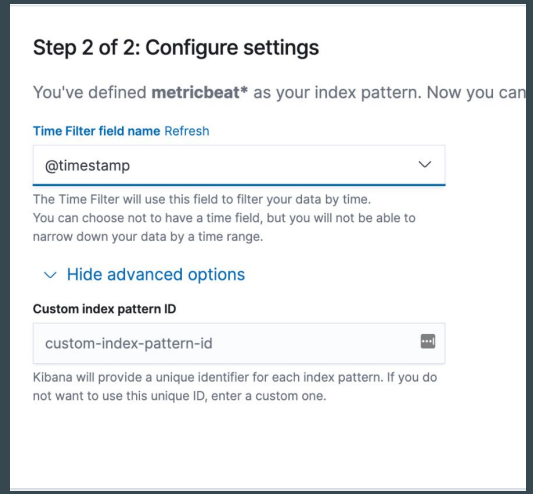
KIBANA INTRODUCTION

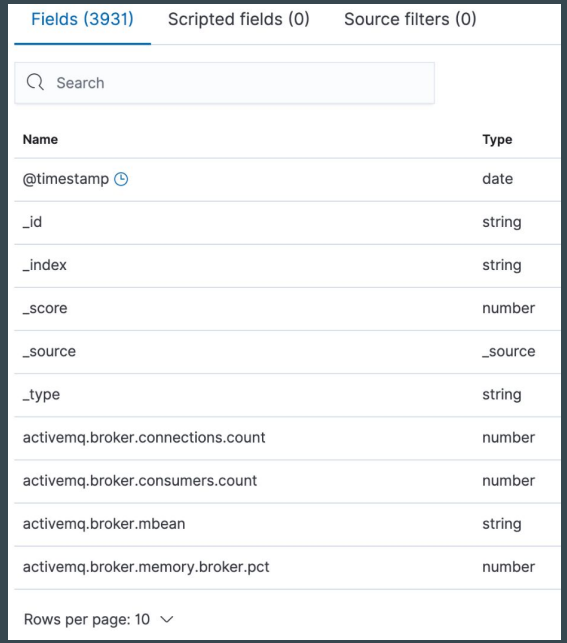
Visualizing our data

Create Kibana index pattern for our metricbeat:
We need an index pattern that will “point” to the indices that we want to draw the data from. So let’s go ahead and open the Management app → Index Patterns → Create index pattern → and create one for metricbeat* indices. Use @timestamp as the Time Filter.

KIBANA INTRODUCTION
Visualizing our data







KIBANA INTRODUCTION

Visualizing our data

Creating custom load on our system using “stress” tool

```
sudo yum install stress -y
Checking how many cores and how much RAM we got
# processor cores
nproc
# memory
free -h
```

KIBANA INTRODUCTION

Visualizing our data

Creating custom load on our system using “stress” tool

Starting our stress test

First spinning two workers which will max the CPU cores for 2 minutes (120 sec):

```
nohup stress --cpu 2 --timeout 120 &
```

Second running 5 workers that should allocate 256MB of memory each for 3 minutes:

```
nohup stress --vm 5 --timeout 180 &
```

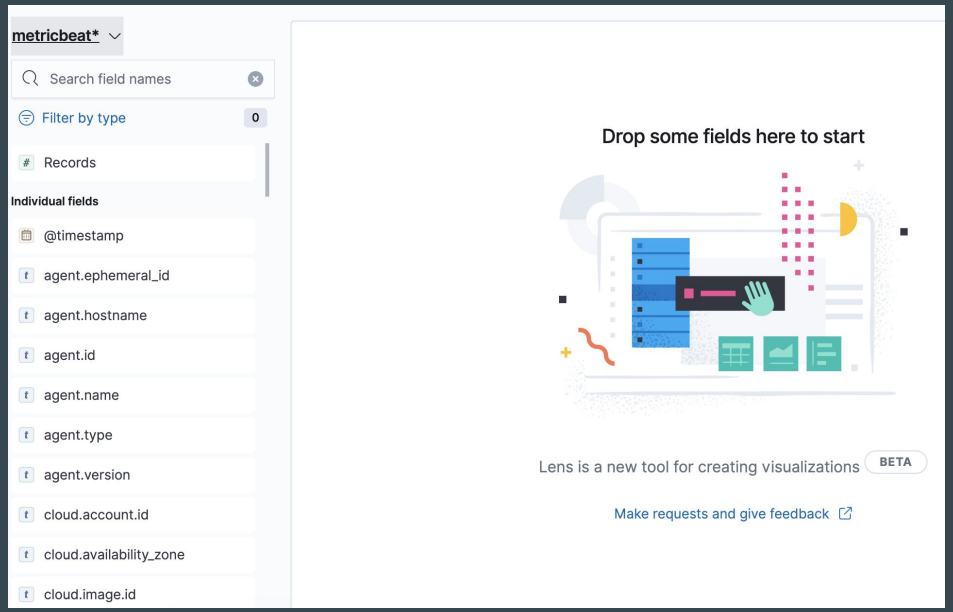
CREATING A VISUALIZATION

KIBANA INTRODUCTION

Visualizing our data

Now we can open the Visualize app in Kibana.
Create new visualization → and then pick the Lens visualization type

There we select our Metricbeat index

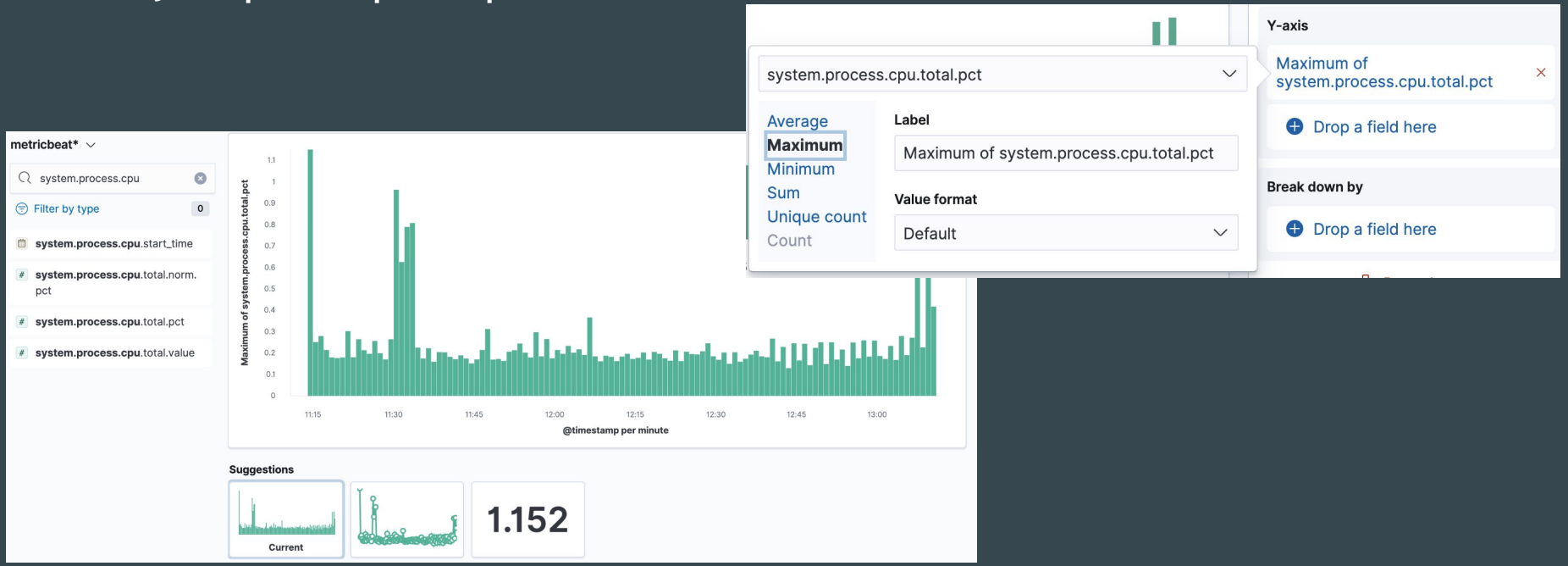


KIBANA INTRODUCTION

Visualizing our data

Let's add some metrics such as:

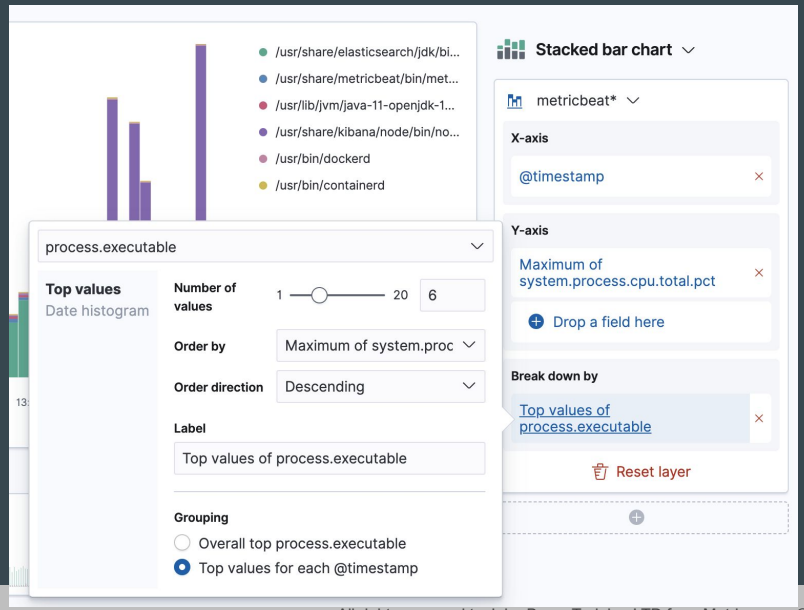
- system.process.cpu.total.pct

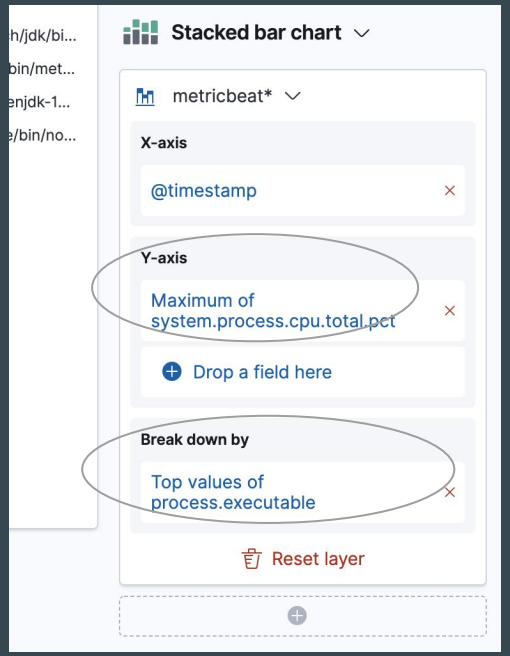


KIBANA INTRODUCTION

Visualizing our data

Now lets aggregate the data along with another metric called “process.executable” that we will drag to the right side of the screen under “break down by”.
This will allow us to see which binaries are / was running
In the process.



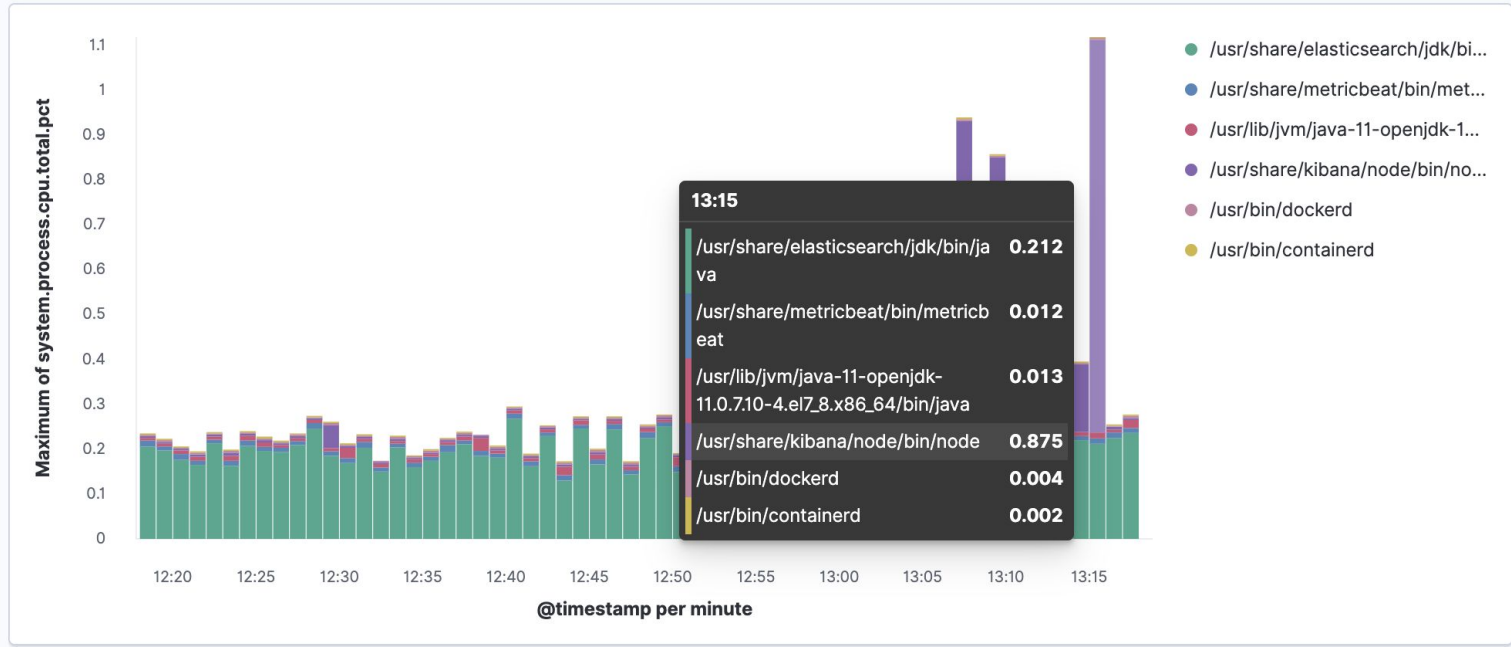


KIBANA INTRODUCTION

Visualizing our data

Before we continue,

We should see something similar to the following:



KIBANA INTRODUCTION

Visualizing our data

Adding additional visualization

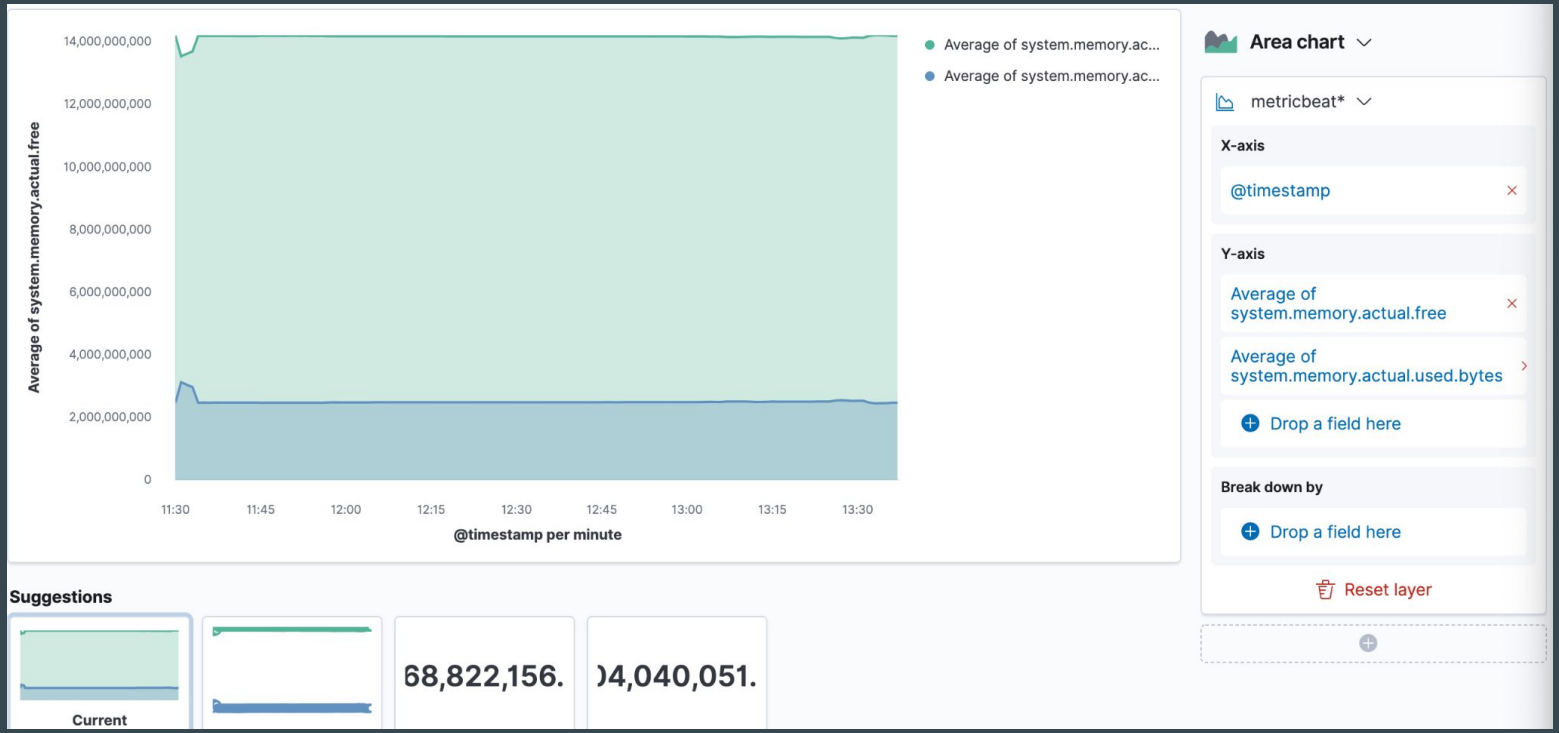
Hands on:

1. Create another new visualization type lens
2. Search for “memory.actual” and drag the following to the main area:
   a. System.memory.actual.used.bytes
   b. System.memory.actual.free
3. Change granularity of the displayed data to 10 seconds interval (metricbeat default) (under @timestamp...)
4. Change the lawet from **Barchart** to a **Stacked area chart**...

KIBANA INTRODUCTION

Visualizing our data

Should look like the following



KIBANA INTRODUCTION

Visualizing our data

LAYERS

The last feature we are going to try out is the ability to stack multiple layers to combine different types of charts in the same visualization.

Again create a new Lens visualization and search for the socket.summary metrics, which is what we are going to use for this step.

Drag and drop the system.socket.summary.all.count field → change the chart type to Line chart → and change the Time interval to 1 minute. Easy!

We can find the layers under the “RESET LAYER” on the lower right screen.

KIBANA INTRODUCTION

Visualizing our data

LAYERS

Now click the plus button in the right pane which will add a new visualization layer → change the chart type to Bar chart **(we need to do it with the small chart icon of the given layer)** → and drop in the @timestamp for the X-Axis and listening, established, time_wait, close_wait from system.socket.summary.tcp.all. → additionally we can add also system.socket.summary.udp.all.count to also see the UDP protocol sockets. Lastly, change the time granularity to the same value as the second layer.

- KIBANA - WATCHER

KIBANA INTRODUCTION

WATCHER

INTRO

Watcher is an Elasticsearch feature that we can use to create actions based on conditions, which are periodically evaluated using queries on our data. Watches are helpful for analyzing mission-critical and business-critical streaming data. For example, we might watch application logs for performance outages or audit access logs for security threats.

KIBANA INTRODUCTION

WATCHER

Watch definition

Trigger

Determines when the watch is checked. A watch must have a trigger.

Input

Loads data into the watch payload. If no input is specified, an empty payload is loaded.

Condition

Controls whether the watch actions are executed. If no condition is specified, the condition defaults to always.

Transform

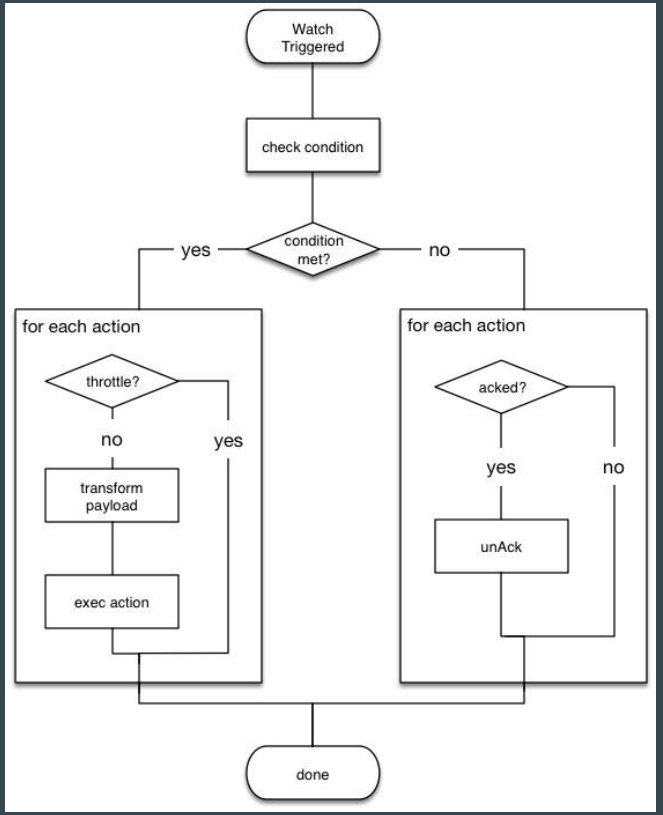
Processes the watch payload to prepare it for the watch actions. We can define transforms at the watch level or define action-specific transforms. Optional.

Actions

Specify what happens when the watch condition is met.

KIBANA INTRODUCTION
WATCHER

The following diagram shows the watch execution process:



KIBANA INTRODUCTION
WATCHER

BASICS

```
PUT _watcher/watch/watch_name
{
  "trigger" : {},
  "input" : {},
  "condition" : { },
  "transform" : { },
  "actions" : {}
}
```

This is a simple elastic call that can be executed in Kibana console. This call will create an Elastic Watcher with the name “watch_name”.
We will discuss each element separately:

KIBANA INTRODUCTION

WATCHER

TRIGGER

```
"trigger" : {
  "schedule" : {
    "interval" : "5m"
  }
}
```

Determines when the watch execution process should start. We can either provide a cron expression or specify a period in terms of intervals.
This is an interval based trigger that will execute in a five-minute interval once the Watcher is created.

INPUT

Input is the source from where we fetch the data and load it into the execution context. The result from this input is called a “watcher payload” or “context payload”. Watcher supports four types of inputs:
>>>>>

KIBANA INTRODUCTION
WATCHER

INPUT CONTINUE

simple: load static data into the execution context
search: load the results of a search into the execution context
http: load the results of an HTTP request into the execution context
chain: use a series of inputs to load data into the execution context

CONDITION

It is used to decide whether we should invoke a Watcher action or not. Default is true.

The way it's been calculated is by checking the payload of the search and loading it Into ELASTICSEARCH CTX

```
"condition": {
   "compare": {
     "ctx.payload.hits.total": {
       "gt": 0
     }
   }
 }
```

KIBANA INTRODUCTION
WATCHER

ELASTICSEARCH CTX?

ctx is a special variable that allows we to access the source of the object

Accessing the search results

Conditions, transforms, and actions can access the search results through the watch execution context. For example:

- To load all of the search hits into an email body, use ctx.payload.hits.
- To reference the total number of hits, use ctx.payload.hits.total.
- To access a particular hit, use its zero-based array index. For example, to get the third hit, use ctx.payload.hits.hits.2.
- To get a field value from a particular hit, use ctx.payload.hits.hits.<index>.fields.<fieldname>. For example, to get the message field from the first hit, use ctx.payload.hits.hits.0.fields.message.

KIBANA INTRODUCTION

WATCHER

TRANSFORMATION

A “Transform” processes and changes the payload in the watch execution context to prepare it for the watch actions.

ACTION

It is executed when conditions are met. Eg. Send an email, send a slack message, call a rest service, add loggers, etc.

```
"actions": {
    "log": {
      "logging": {
        "text": "We got the expected error"
      }
    }
  }
```

KIBANA INTRODUCTION
WATCHER

WATCHER APIs

Create Watcher

```
CREATE:
PUT _watcher/watch/watch_name
{
  \\Body

}

GET _watcher/watch/watch_name

ACTIVATE/DEACTIVATE:  PUT _watcher/watch/watch_name/_activate/ OR _deactivate

EXECUTE WATCHER:  PUT _watcher/watch/watch_name/_execute
```

KIBANA INTRODUCTION

WATCHER

WATCHER APIs

WATCHER EXAMPLE >>> COPY IT TO KIBANA DEV FOR REVIEW

Highlights about the Watcher definition on the right:

- This Watcher will execute once every 10 minutes
- It will look for a particular log with the specified status in the input i.e. the search query
- In the event that the condition is fulfilled, it will execute the action
- In the action, we simply add a logger in the elastic log, which we can check in the elastic console

```
PUT /_watcher/watch/log_error_watch
{
  "trigger": {
    "schedule": {
      "interval": "10m"
    }
  },
  "input": {
    "search": {
      "request": {
        "indices": ["logs"],
        "body": {
          "query": {
            "match": {
              "message": "error"
            }
          }
        }
      }
    }
  },
  "condition": {
    "compare": {
      "ctx.payload.hits.total": {
        "gt": 0
      }
    }
  },
  "actions": {
    "log": {
      "logging": {
        "text": "We got the expected error"
      }
    }
  }
}
```

KIBANA INTRODUCTION

WATCHER

TESTING

1. Let's add a new INDEX named LOG with the following mapping

```
"request":     { "type": "keyword" },
"status_code": { "type": "keyword" },
"message":     { "type": "keyword" },
"timestamp":   { "type": "date"    }
```

2. Add new Document into the Index

```
"timestamp" : "2015-05-17T18:12:07.613Z",
"request" : "GET index.html",
"status_code" : 404,
"message" : "Error: File not found"
```

```
PUT /_watcher/watch/log_error_watch
{
 "trigger": {
   "schedule": {
     "interval": "10m"
   }
 },
 "input": {
   "search": {
     "request": {
       "indices": ["logs"],
       "body": {
         "query": {
           "match": {
             "message": "error"
           }
         }
       }
     }
   }
 },
 "condition": {
   "compare": {
     "ctx.payload.hits.total": {
       "gt": 0
     }
   }
 },
 "actions": {
   "log": {
     "logging": {
       "text": "We got the expected error"
     }
   }
 }
}
```

KIBANA INTRODUCTION
WATCHER

TESTING HANDS ON

3. Execute the Watcher
4. View the log in elasticsearch logs (docker logs -f elkstack-docker_elasticsearch_1)

Advanced WATCHER ACTIONS:
https://www.elastic.co/guide/en/elasticsearch/reference/master/actions.html

```
PUT /_watcher/watch/log_error_watch
{
  "trigger": {
    "schedule": {
      "interval": "10m"
    }
  },
  "input": {
    "search": {
      "request": {
        "indices": ["logs"],
        "body": {
          "query": {
            "match": {
              "message": "error"
            }
          }
        }
      }
    }
  },
  "condition": {
    "compare": {
      "ctx.payload.hits.total": {
        "gt": 0
      }
    }
  },
  "actions": {
    "log": {
      "logging": {
        "text": "We got the expected error"
      }
    }
  }
}
```

- KIBANA - MANAGEMENT

KIBANA INTRODUCTION

MANAGING KIBANA

SPACES

Kibana is considered the “window” to Elasticsearch and indeed it’s a powerful UI for searching, filtering, analyzing, and visualizing Elasticsearch data, but Kibana settings are also used to configure, administer and monitor the Elasticsearch cluster. In this lesson, we’re going to explore how Kibana settings can be tweaked for collaborative teamwork.

KIBANA INTRODUCTION

MANAGING KIBANA

In Kibana spaces we will learn about

- Spaces enable we to organize various Dashboards, Visualizations, Searches, and other so-called Saved Objects on our Kibana instance, into distinct “spaces” where different user groups can access what they need.
- This can be especially useful for segregating various business users according to departments like Marketing, Security, Development, Operations, Finance etc.
- We can also assign access to multiple spaces for specific users, in which case they’ll see their accessible spaces when logging in.

KIBANA INTRODUCTION

MANAGING KIBANA

In Kibana spaces we will learn about

Export Kibana Dashboards

- In relation to Spaces, Kibana allows we to transfer a selected set of Saved Objects either by copying them between Spaces within a single Kibana instance or by exporting them in JSON format from one Kibana instance and importing into another instance. This can be useful in multi-cluster environments.

Advanced Kibana Settings

- Lastly, we'll take a look at some settings related to user setup and spaces such as setting a default landing page when entering a space or switching to Dark mode.

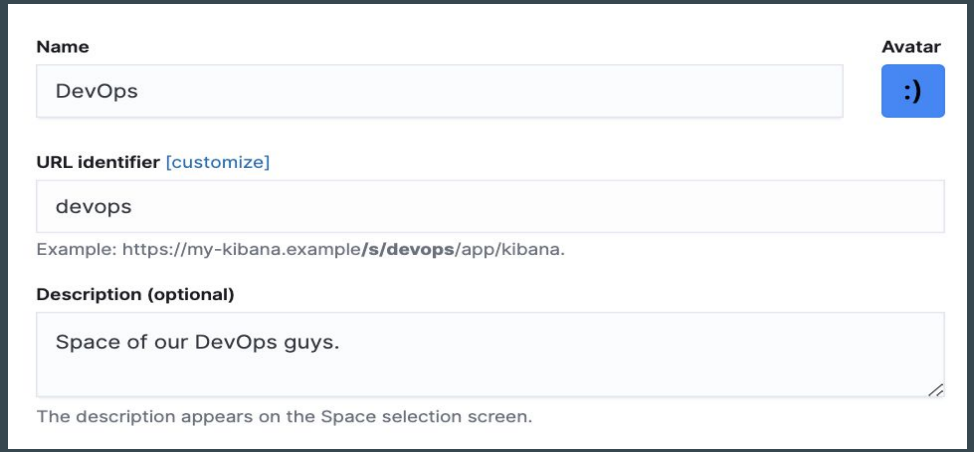
KIBANA SPACES

Let's start by navigating to the Management app in Kibana → and from there, head over to the Spaces configuration in the Kibana section. It can be found here running on our remote host and default port http://REMOTEHOST:5601/app/kibana#/management/spaces/list?_g=()

As we can see right away, there's already one space created for us by default (named Default). It's always present if we haven't explicitly disabled spaces. All objects that are contained inside the default space are shared between all users.

KIBANA SPACES

Let's create a new space by clicking on Create a space. We'll add the name "DevOps" and we can optionally provide a description. Lastly, we have the option to choose a color for space. This is helpful to visually differentiate spaces for users who have access to multiple spaces.



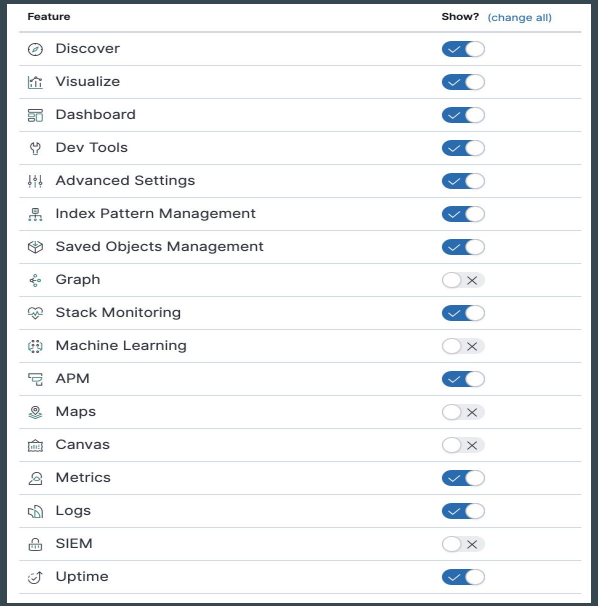
KIBANA SPACES

Then, scroll down to the Filters section where we can also filter out any sections that are not relevant for the specific target group. Our DevOps example would likely need access to most or all of these, but if we would be preparing a space for marketing, for example, we'd probably want to hide most of these and just enable access to the Dashboard menu option.

Bear in mind that this only hides or displays the menu options in the Kibana UI, but users would still be able to reach those areas via a direct URL. In other words, these Filters are not a replacement for access control features which would need to be handled separately.

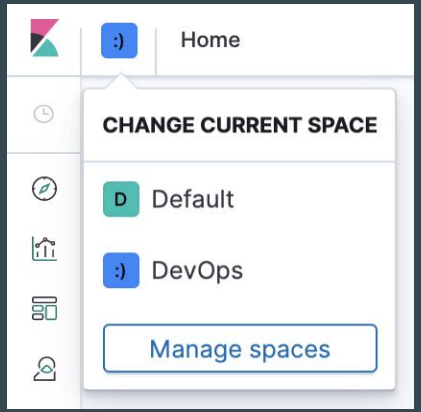
KIBANA INTRODUCTION
MANAGING KIBANA

KIBANA SPACES

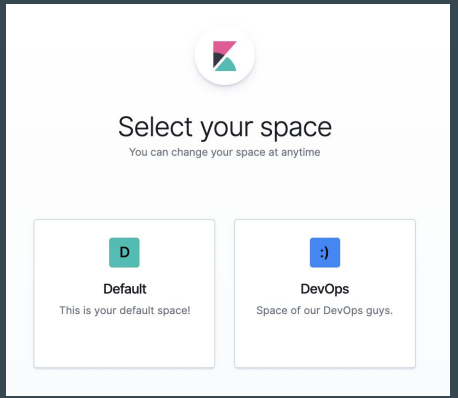


Once finished click create

Now we can see the new space in the top left menu by clicking on the Spaces icon next to the Kibana logo. This menu is great for quickly jumping between different spaces.



NEW LOGIN SCREEN

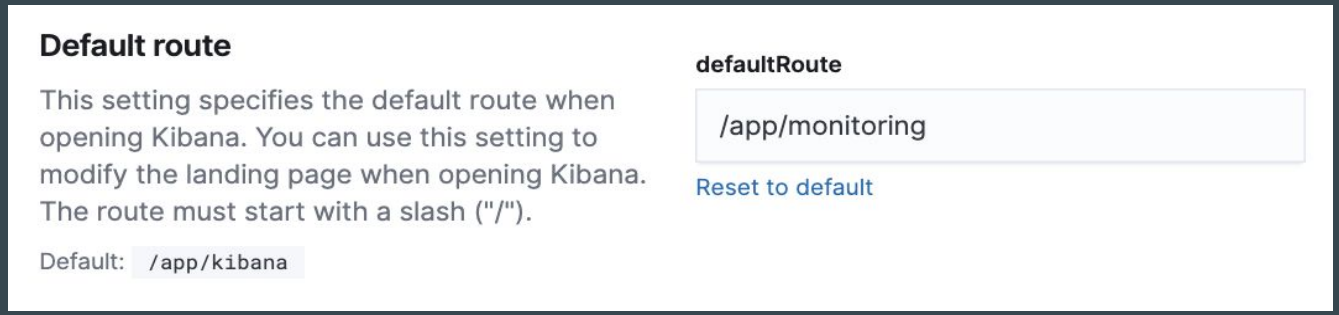


Advanced Kibana settings

For our last example, let's head over to the Advanced Settings section in the Management app where many settings are concentrated.

First of all, bear in mind the warning box that is displayed on top of the page saying "Caution: We can break stuff here". This is true. All sorts of settings, both "high" and "low" level, are found here in one place, so always double-check what we're about to tweak. Secondly, these settings are mostly applied to the active space (i.e. not globally), which is useful for our case.

Advanced Kibana settings



We can change the default landing page to which the users are redirected when they switch to a given space. Do so with the Default route setting. Here we have switched the default landing page to the Monitoring app which aggregates information about our Elastic Stack components. For example, our Operations team can go straight to the overview of the cluster, similarly, we can guide marketing users to their main dashboard.

KIBANA INTRODUCTION

MANAGING KIBANA

Advanced Kibana settings

More advanced options can be seen in the following link
https://www.elastic.co/guide/en/kibana/current/advanced-options.html

UPDATED TO ELK V.7.17

ELK STACK

Module - CANVAS

Provided by DevOpShift

Driver

Schroeder

Position

1

Lap Times

Average

6.067 s

Best

5.159 s

Speed

Fastest

1.064 m/s

Position

2

Driver

Hello Kitty

Lap Times

Average

6.676 s

Best

4.931 s

Speed

Fastest

1.064 m/s

Driver

Count Chocula

Position

3

Lap Times

Average

5.432 s

Best

4.131 s

Speed

Fastest

1.329 m/s

Position

4

Driver

Cat in the Hat

Lap Times

Average

4.956 s

Best

4.366 s

Speed

Fastest

1.257 m/s

TOPICS

ELK STACK CANVAS

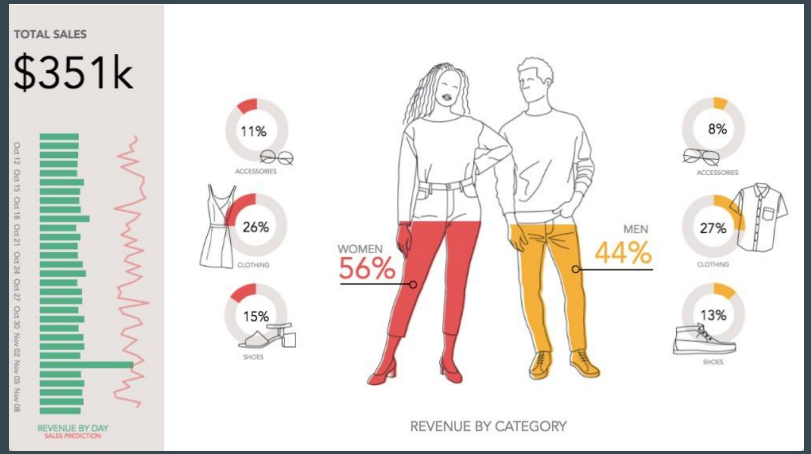
- OVERVIEW
- DATA EXPRESION
- PRESENTING
- REPORTING
- SQL
- EXTEND
- WATCH

- OVERVIEW -
WHAT IS CANVAS

SHOWCASE YOUR DATA IN CANVAS

Canvas is a presentation tool — built into Kibana — that allows users to create pixel-perfect presentations and slide decks that pull live data directly from Elasticsearch.

This means no manual updates for presentations, resulting in up-to-the-minute visuals, chart elements, and graphic



WHAT CAN WE DO WITH CANVAS

Canvas tools have rich styling capabilities, allowing users to change colors and styles of both static and dynamic elements. Canvas is designed that we can tweak an element's content and style using the UI, or go deeper and edit the expressions that create the element or the css styling. In short, Canvas gives we everything we need to showcase our live data, our way.

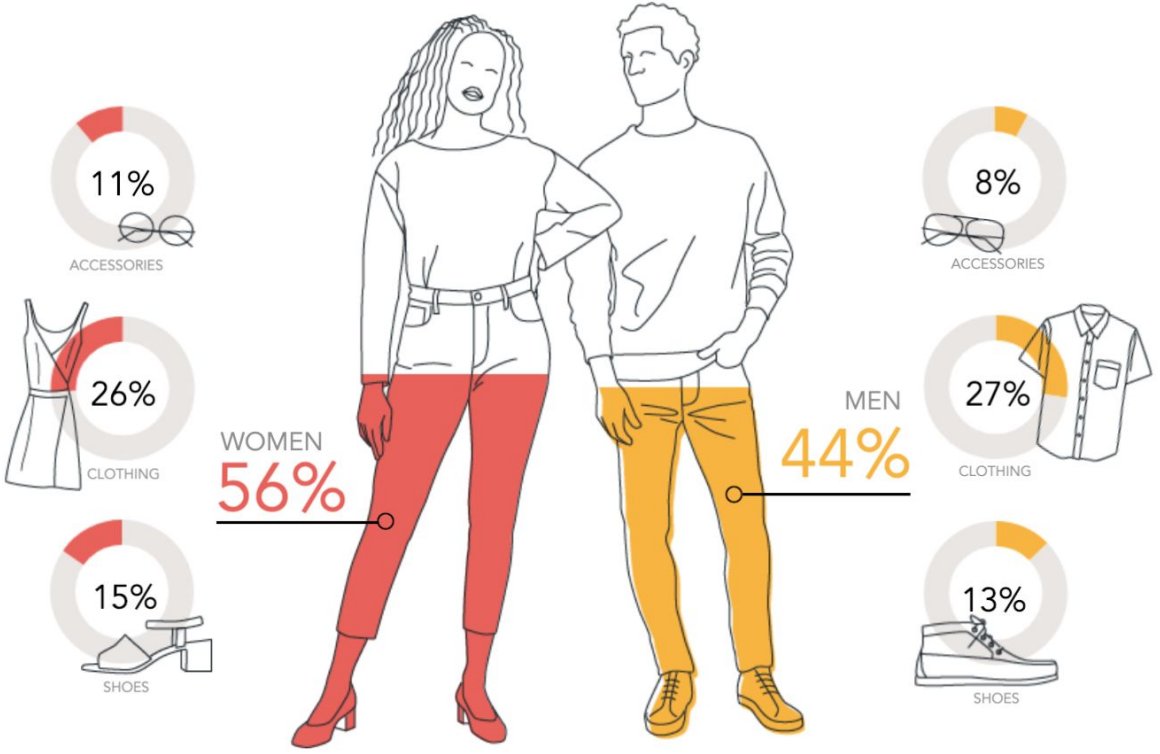
- BUILDING A NEW CANVAS - E-COM DEMO

KNOW YOUR WORKSPACE

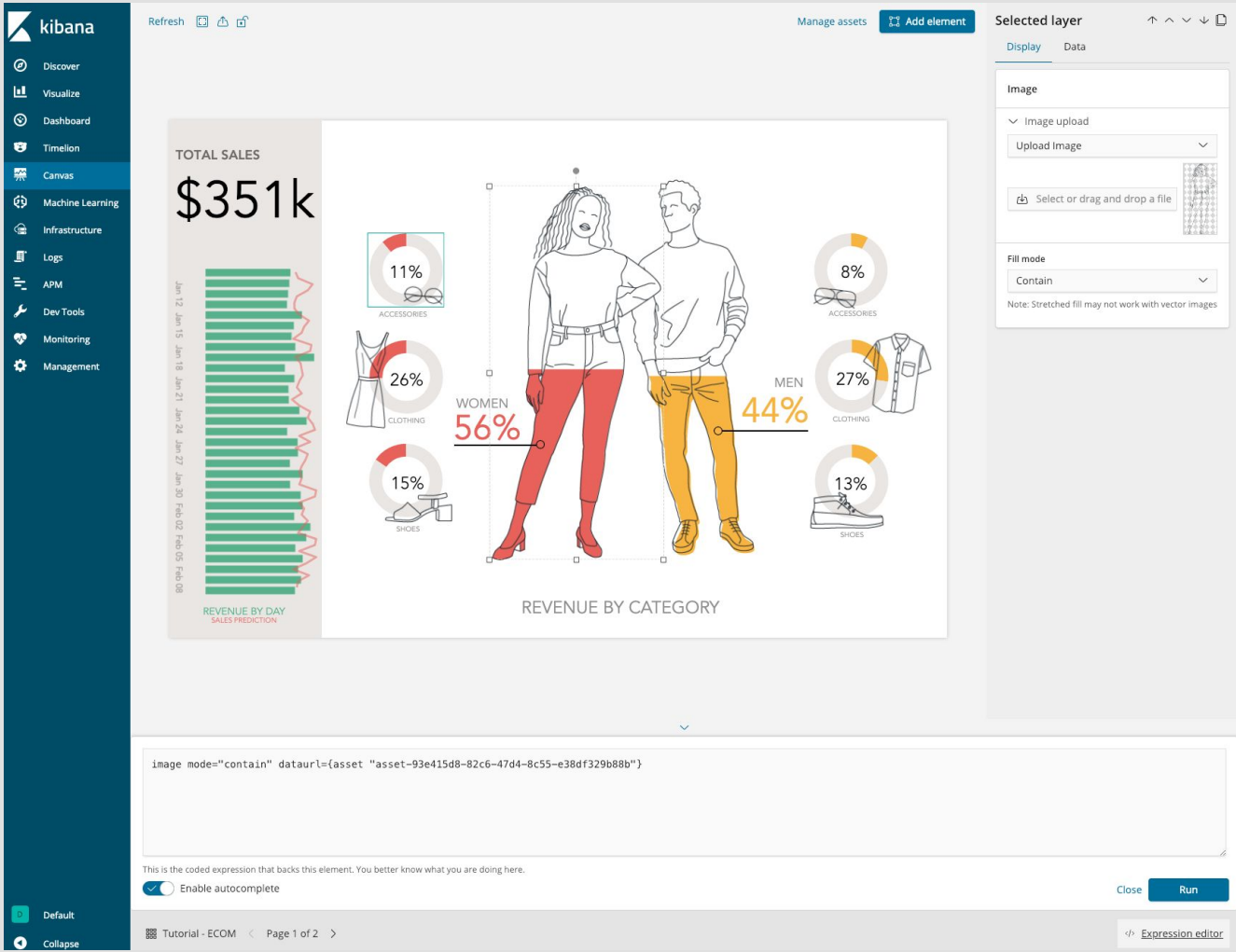
TOTAL SALES

$351k

REVENUE BY DAY
SALES PREDICTION



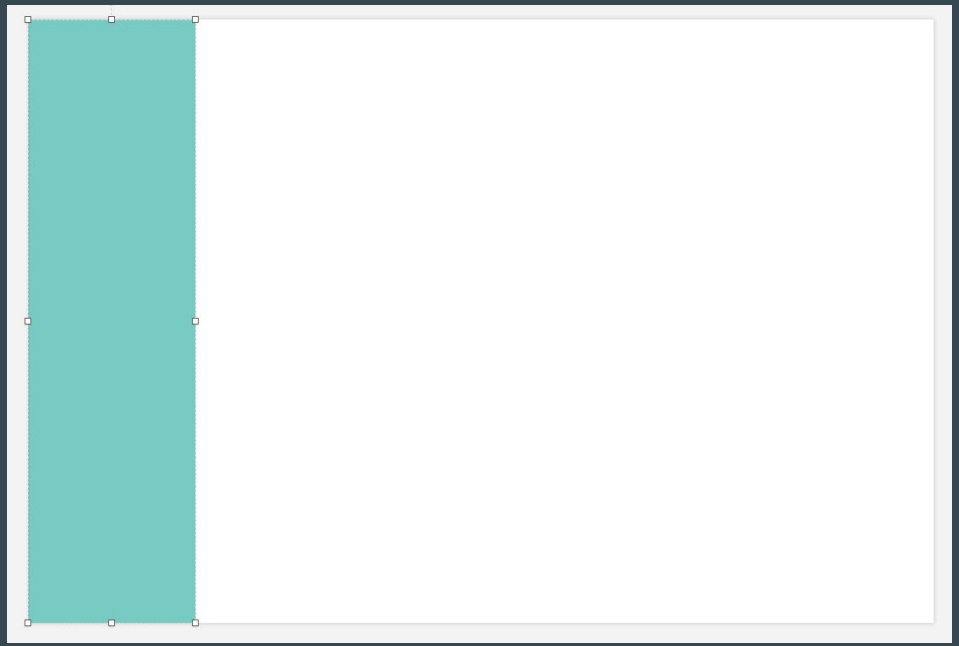
REVENUE BY CATEGORY

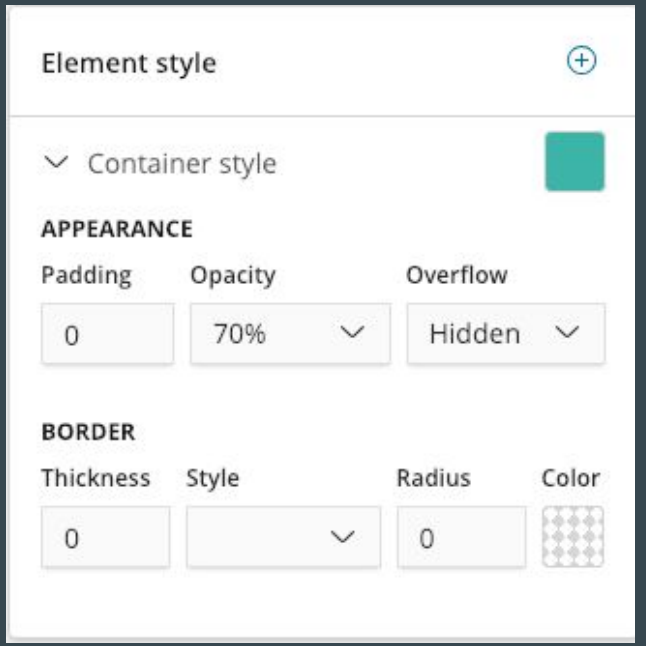

kibana
Discover
Visualize
Dashboard
Timelion
Canvas
Machine Learning
Infrastructure
Logs
APM
Dev Tools
Monitoring
Management
Default
Collapse
Refresh
Manage assets
Add element
TOTAL SALES
$351k
Jan 12 Jan 15 Jan 18 Jan 21 Jan 24 Jan 27 Jan 30 Feb 02 Feb 05 Feb 08
REVENUE BY DAY
SALES PREDICTION
11%
ACCESSORIES
26%
CLOTHING
15%
SHOES
WOMEN
56%
MEN
44%
8%
ACCESSORIES
27%
CLOTHING
13%
SHOES
REVENUE BY CATEGORY
Selected layer
Display
Data
Image
Image upload
Upload Image
Select or drag and drop a file
Fill mode
Contain
Note: Stretched fill may not work with vector images
image mode="contain" dataurl={asset "asset-93e415d8-82c6-47d4-8c55-e38df329b88b"}
This is the coded expression that backs this element. You better know what you are doing here.
Enable autocomplete
Close
Run
Tutorial - ECOM
Page 1 of 2
Expression editor

LET'S START

BUILDING OUR OWN CANVAS

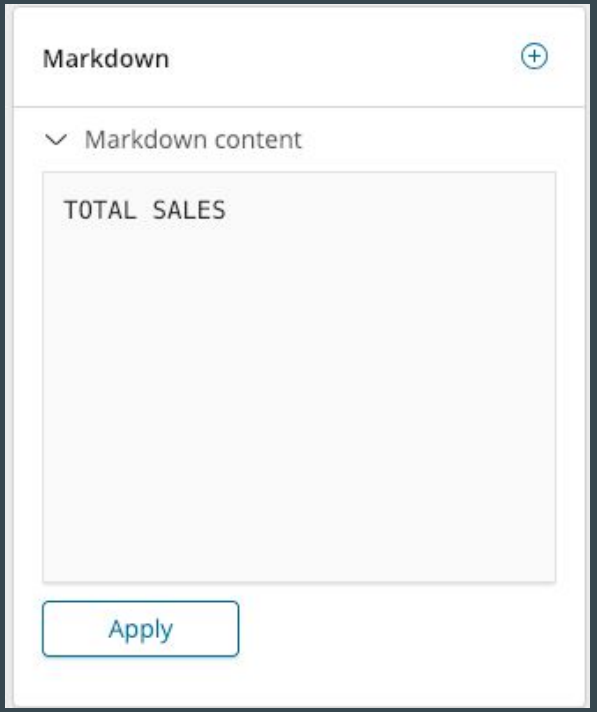
1. **Adding a new Shape**
   a. **Select “element style - Container style”**

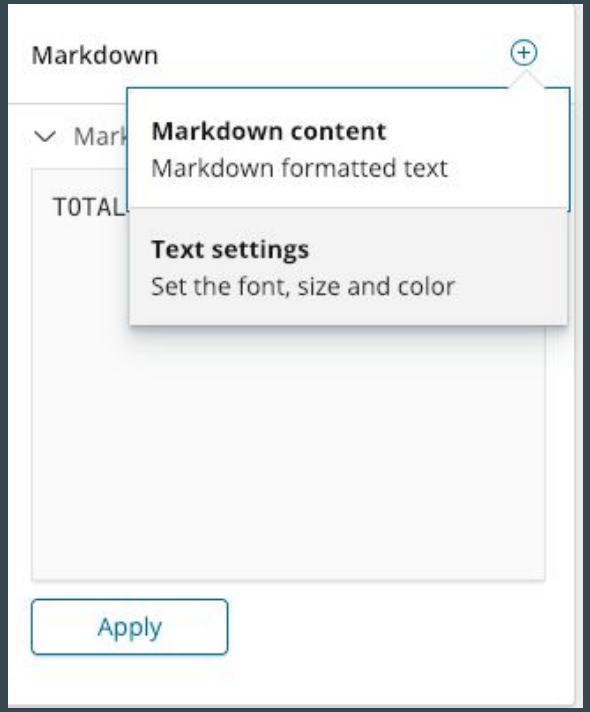


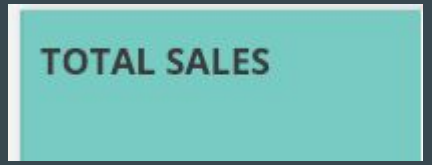


BUILDING OUR OWN CANVAS

1. **Adding Markdown**
   a. Type: “TOTAL SALES”
   b. Add Markdown “Text Setting”

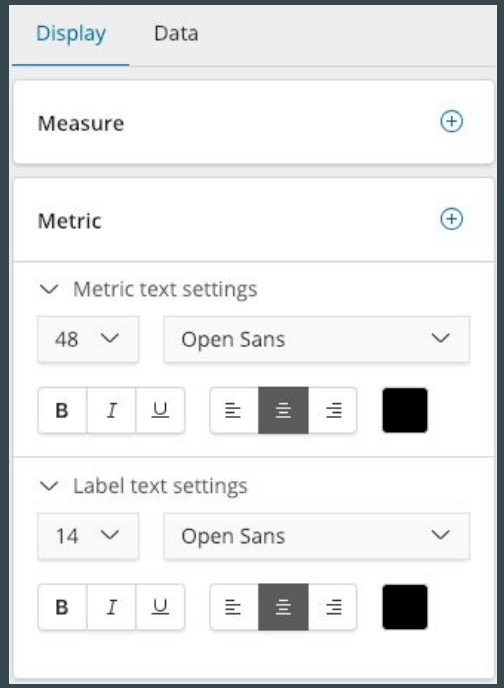


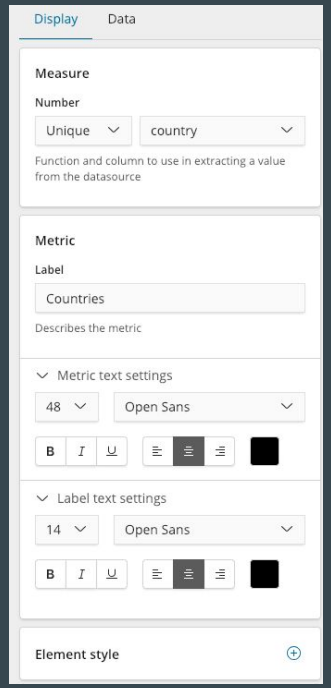


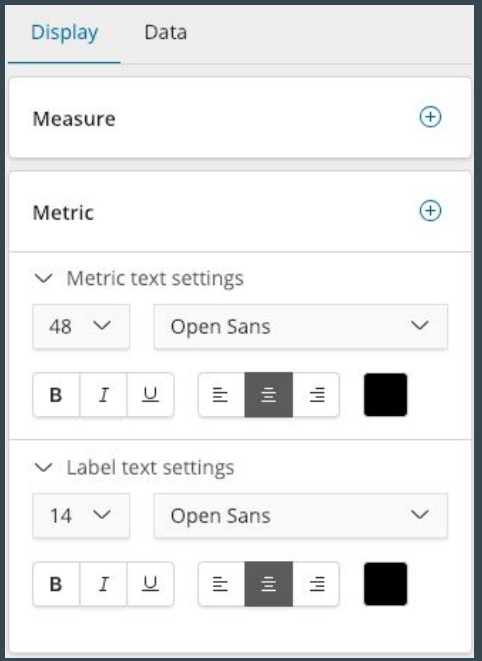


BUILDING OUR OWN CANVAS

1. **Adding - Metric**
   a. Delete Label , Measure
2. Click Data and click -> “Change our data source” -> “ElasticSeacrh SQL”



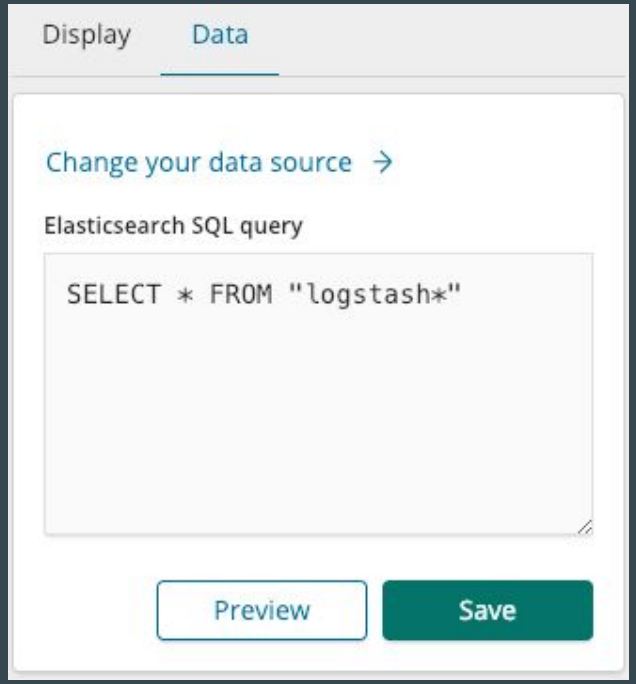


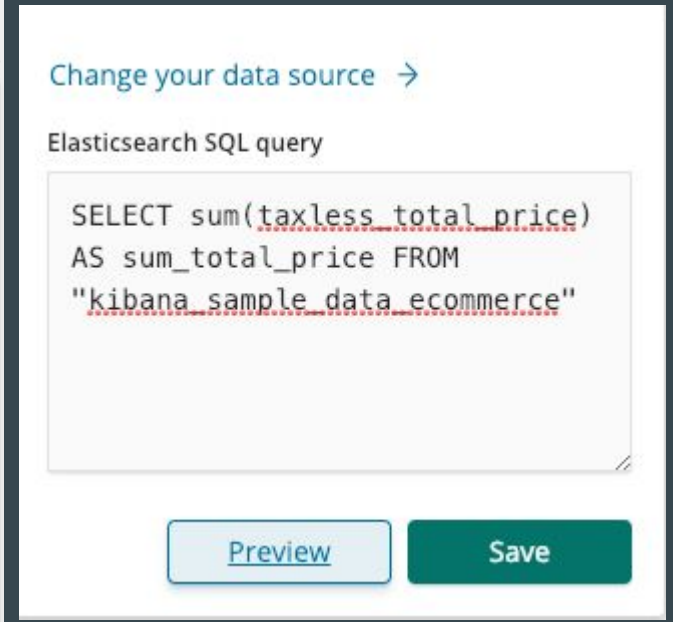


BUILDING OUR OWN CANVAS

1. Update the following Query into the SQL QUERY

```
SELECT sum(taxless_total_price) AS sum_total_price
FROM "kibana_sample_data_ecommerce"
```





EXPRESSION EDITOR

BUILDING OUR OWN CANVAS

1. The power of Canvas lay on the bottom right of the canvas workspace

the expression language is what drives canvas using built in functions.
Formatting and Functions documentation can be found here:

https://www.elastic.co/guide/en/kibana/current/canvas-common-functions.html

BUILDING OUR OWN CANVAS

1. Using the “Expression editor” we want to transform a number 350001 to $351k
   a. In the formating and functions documentation we can see what type of formats are available to use https://www.elastic.co/guide/en/kibana/current/canvas-function-reference.html

   *** format number reference:
   https://www.elastic.co/guide/en/kibana/6.8/canvas-common-functions.html#_formatnumber

```
filters -- > Collects element filters on the workpad, usually to provide them to a data source.

| essql ---> Specifies an Elasticsearch SQL.
  query="SELECT sum(taxless_total_price) AS sum_total_price FROM \"kibana_sample_data_ecommerce\""
| math "sum(sum_total_price)" ---> TinyMath provides a set of functions that can be used with the Canvas
| formatNumber "$0a" ---> Turns a number into a string using a Number.js format.
| metric ---> A number with a label.
    metricFont={font family="'Avenir', Helvetica, Arial, sans-serif" size=72 align="left" color="#000000" weight="normal" underline=false italic=false}
| render
```

BUILDING OUR OWN CANVAS

1. Copy the below code and paste it into the “expression editor” and click run

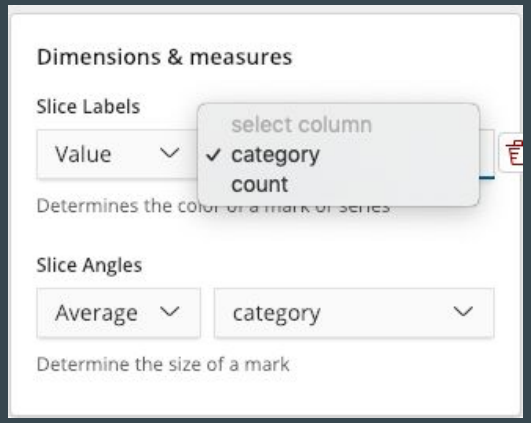
```
essql
  query="SELECT sum(taxless_total_price) AS sum_total_price FROM \"kibana_sample_data_ecommerce\""
| math "sum_total_price"
| formatNumber "$0a"
| metric
  metricFont={font family="'Avenir', Helvetica, Arial, sans-serif" size=72 align="left" color="#000000" weight="normal" underline=false italic=false}
| render
```

GOING DEEPER THE RABBIT HOLE

BUILDING OUR OWN CANVAS

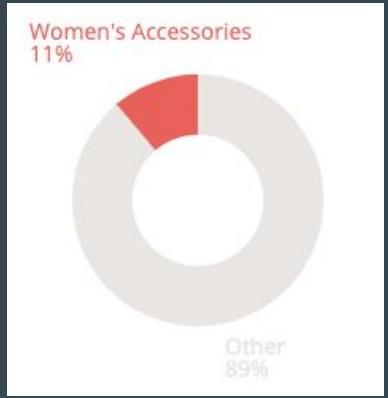
1. Building a chart with custom fields
   a. Add a new chart element
   b. Change data source to elasticsearch SQL
   c. Type the following:

```
SELECT category, COUNT(category) AS count FROM
"kibana_sample_data_ecommerce" GROUP BY category
```



BUILDING OUR OWN CANVAS

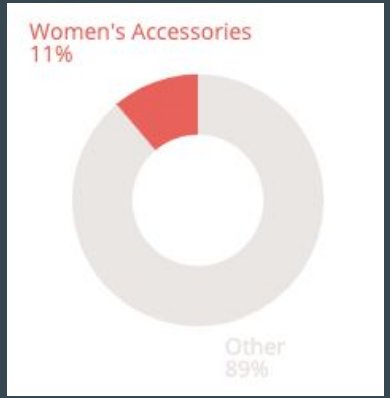
1. Now we wish to customize our chart to look as this --->
   a. First problem is that in the Dimensions & measures we don't have percentage option To slice our data
   b. Second problem is that in the chart in the right we see that the data present Women's Accessories info and “other” instead of all the categories



To accomplish this we need to use the Expression editor again

```
essql
  "SELECT category, COUNT(category) AS count FROM \"kibana_sample_data_ecommerce\" GROUP BY category"
| mapColumn "category" → Adds a column calculated as the result of other columns, or not.
  fn={if {getCell "category" | eq "Women's Accessories"} then={getCell "category"} else="Other"}
| ply by="category" fn={math "sum(count)" | as "count"} → Subdivides a datatable and passes the resulting tables into an expression, then merges the output.
| staticColumn "total" value={math "sum(count)"} → Adds a column with a static value.
| mapColumn "percentage" fn={math "round(count/total * 100, 2)"}
| pointseries color="category" size="percentage" → Turns a datatable into a point series model to be used in a pie chart
| pie hole=39 labels=true palette={palette "#ede9e7" "#eb6c66" gradient=false}
```

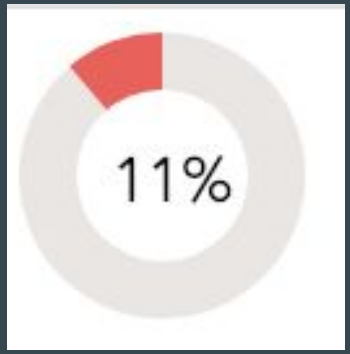
BUILDING OUR OWN CANVAS




```
essql
  "SELECT category, COUNT(category) AS count FROM \"kibana_sample_data_ecommerce\" GROUP BY category"
| mapColumn "category"
  fn={if {getCell "category" | eq "Women's Accessories"} then={getCell "category"} else="Other"}
| ply by="category" fn={math "sum(count)" | as "count"}
| staticColumn "total" value={math "sum(count)"}
| mapColumn "percentage" fn={math "round(count/total * 100, 2)"}
| pointseries color="category" size="percentage"
| pie hole=39 labels=true palette={palette "#ede9e7" "#eb6c66" gradient=false}
```

BUILDING OUR OWN CANVAS

2. Adding Pie percentage marker




```
essql 'SELECT category, COUNT(category) AS count FROM "kibana_sample_data_ecommerce" GROUP BY category'
| mapColumn category fn={if {getCell category | eq "Women's Accessories"} then={getCell category} else="Other"}
| ply by=category fn={math 'sum(count)' | as count}
| staticColumn total value={math 'sum(count)'}
| mapColumn percentage fn={math 'round(count/total * 100, 0)'}
| filterrows {getCell "category" | any {eq "Women's Accessories"}}
| markdown {getCell "percentage"} "%" font={font family="Avenir" size=24 align="center" color="#000" weight="normal" underline=false italic=false}
```

LET’S BREAK IT APART

BUILDING OUR OWN CANVAS

1. Breaking it apart by running part after part

```
essql
  "SELECT category, COUNT(category) AS count FROM \"kibana_sample_data_ecommerce\" GROUP BY category"
| render
```

2.

```
essql
  "SELECT category, COUNT(category) AS count FROM \"kibana_sample_data_ecommerce\" GROUP BY category"
| mapColumn "category"
  fn={if {getCell "category" | eq "Women's Accessories"} then={getCell "category"} else="Other"}
```

BUILDING OUR OWN CANVAS

4.

```
essql
  "SELECT category, COUNT(category) AS count FROM \"kibana_sample_data_ecommerce\" GROUP BY category"
| mapColumn "category"
  fn={if {getCell "category" | eq "Women's Accessories"} then={getCell "category"} else="Other"}
| ply by="category" fn={math "sum(count)" | as "count"}
| staticColumn "total" value={math "sum(count)"}
```

5.

```
essql
  "SELECT category, COUNT(category) AS count FROM \"kibana_sample_data_ecommerce\" GROUP BY category"
| mapColumn "category"
  fn={if {getCell "category" | eq "Women's Accessories"} then={getCell "category"} else="Other"}
| ply by="category" fn={math "sum(count)" | as "count"}
| staticColumn "total" value={math "sum(count)"}
| mapColumn "percentage" fn={math "round(count/total * 100, 2)"}
```

BUILDING OUR OWN CANVAS

6.

```
essql
  "SELECT category, COUNT(category) AS count FROM \"kibana_sample_data_ecommerce\" GROUP BY category"
| mapColumn "category"
  fn={if {getCell "category" | eq "Women's Accessories"} then={getCell "category"} else="Other"}
| ply by="category" fn={math "sum(count)" | as "count"}
| staticColumn "total" value={math "sum(count)"}
| mapColumn "percentage" fn={math "round(count/total * 100, 2)"}
| pointseries color="category" size="percentage"
| pie hole=39 labels=true palette={palette "#ede9e7" "#eb6c66" gradient=false}
```

NOW BUILD THE REST TWO PIES

BUILDING OUR OWN CANVAS

1. For the other two pies we need it to map “Women's Clothing” & “Women's Shoes”

2. Add a Markdown for each Pie by using the same Expressions we used but this time add the following instead of the last two lines that builds our PIE

```
| filterrows {getCell "category" | any {eq "Women's Clothing"}}
| markdown {getCell "percentage"} "%" font={font family="Avenir" size=24 align="center" color="#000" weight="normal" underline=false italic=false}
```

NOW BUILD THE MEN'S SALES PIE'S

BUILDING OUR OWN CANVAS

Make sure to use different colors

1. ACCESSORIES
2. CLOTHING
3. SHOES

Use the following to build a table and view what values we have

```
essql
  "SELECT category, COUNT(category) AS count FROM \"kibana_sample_data_ecommerce\" GROUP BY category"
```

BUILDING IMAGE REVEAL

BUILDING OUR OWN CANVAS

Image reveal is the most “complex” one but supercool.
With image reveal we will create an image that is being filled in real time when new data is being ingested base on the filter we build. - Link for the images

Image reveal is built on top 3 images.

- Background
- Colord
- Sketch

BUILDING OUR OWN CANVAS

1. Using “esdocs” to query ES and get back raw documents

```
esdocs index="kibana_sample_data_ecommerce" sort="order_date, desc" fields="customer_gender"
```

2. Map new Column with Field name “FemaleCount (and MaleCount)”

```
| mapColumn "FemaleCount" exp={getCell "customer_gender" | if {compare to="FEMALE"} then=1 else=0}
```

3. Interprets a math expression

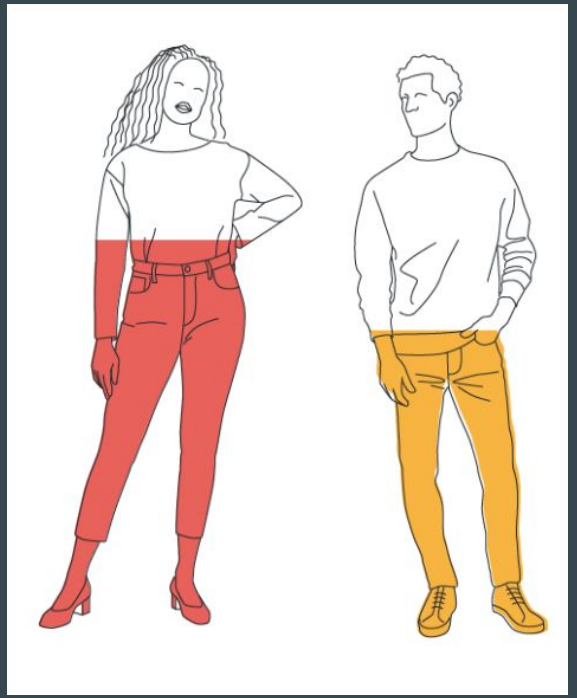
```
| math "sum(FemaleCount) / count(FemaleCount)"
```

4. Use RevealImage

```
| revealImage origin="bottom" image={asset "asset-d6c4f476-5589-49ac-b320-e855114f0d38"} emptyImage={asset
"asset-777ab887-da06-4e37-9ea8-9941899e06f7"}
```

BUILDING OUR OWN CANVAS

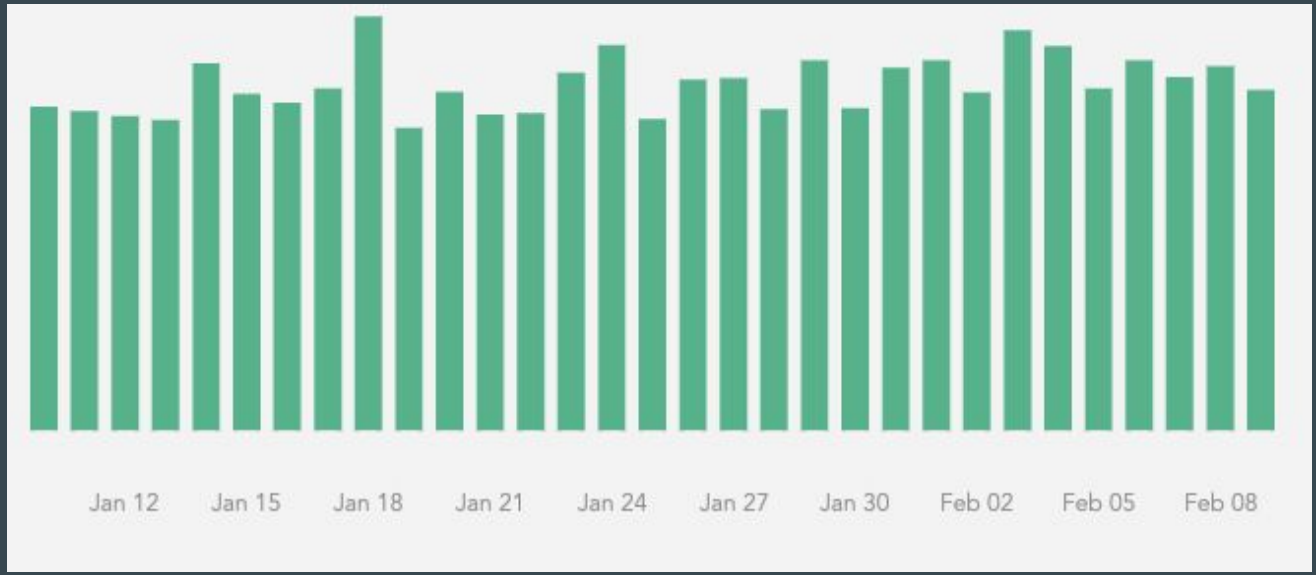
OUTCOME



BUILDING A PLOT

BUILDING OUR OWN CANVAS

Building a PLOT for review by DAY.
To build a time series data base on our documents we need to use TIMELION



BUILDING OUR OWN CANVAS

1. Utilize Timelion MIN / MAX

```
timelion
  ".es(index=kibana_sample_data_ecommerce, metric=sum:taxless_total_price, timefield=order_date)" interval="1d" from={essql "SELECT order_date as min FROM kibana_sample_data_ecommerce order by order_date limit 1" | getCell min} to={essql "SELECT order_date as max FROM kibana_sample_data_ecommerce order by order_date DESC limit 1" | getCell max}
```

2. Select pointers for x / y axies

```
| pointseries x="@timestamp" y="value"
```

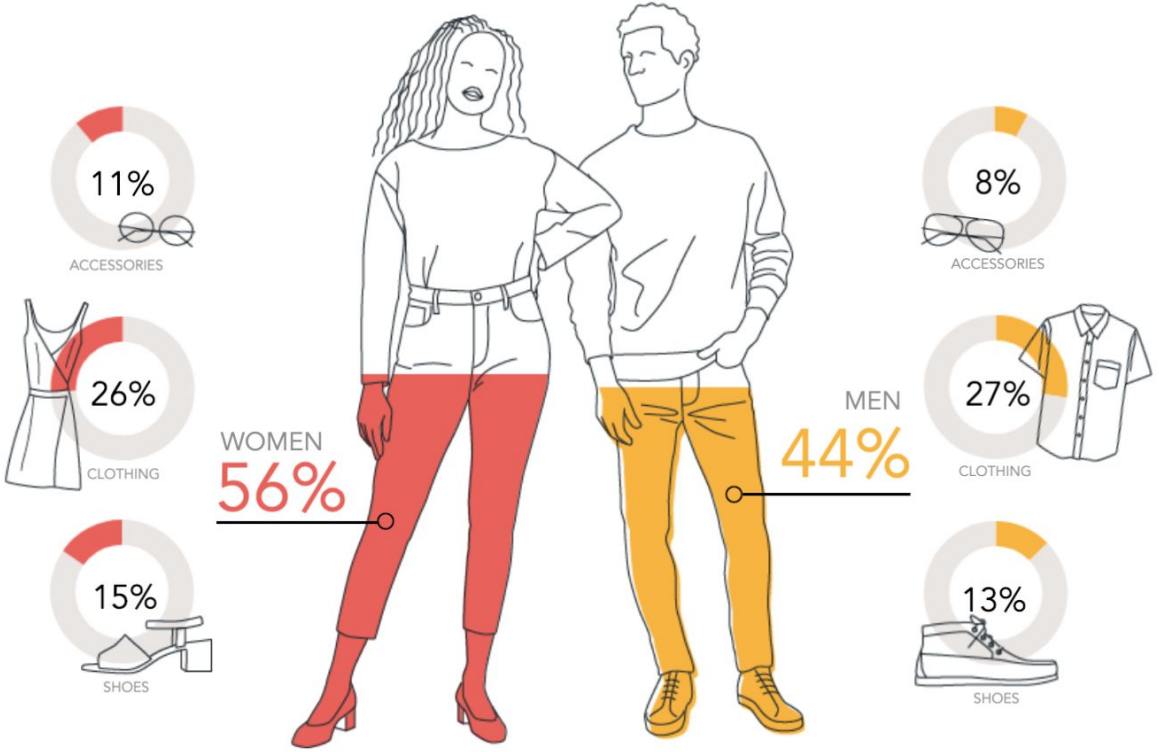
3. Plot the data

```
| plot yaxis=false defaultStyle={seriesStyle points="0" bars="50000000" lines="0" color="#62bb96"} font={font size=12 family="Avenir" color="#999" align="left"}
```

TOTAL SALES

$351k

REVENUE BY DAY
SALES PREDICTION



REVENUE BY CATEGORY

END

Module - CANVAS

Provided by DevOpShift & Presented by Yaniv Cohen