

**ELASTICSEARCH**

**Grimaldo Oliveira**



# O que é o ELASTIC SEARCH

1. O Elasticsearch é um mecanismo criado para análise de dados e busca, faz parte de uma arquitetura central conhecida como Elastic Stack. A função principal é armazenar os seus dados para facilitar as buscas por dados de forma extremamente rápida.
2. Permite realizar e combinar muitos tipos de buscas — estruturadas, não estruturadas, dentre outras .
3. O Elasticsearch usa APIs RESTful e JSON padrão.
4. Uso de múltiplas linguagens para interação com os dados: C#  
Curl, Python, SQL, etc.

# Sites Importantes

## ELASTICSEARCH

[https://www.elastic.co/  
pt/](https://www.elastic.co/pt/)

## LOGSTASH

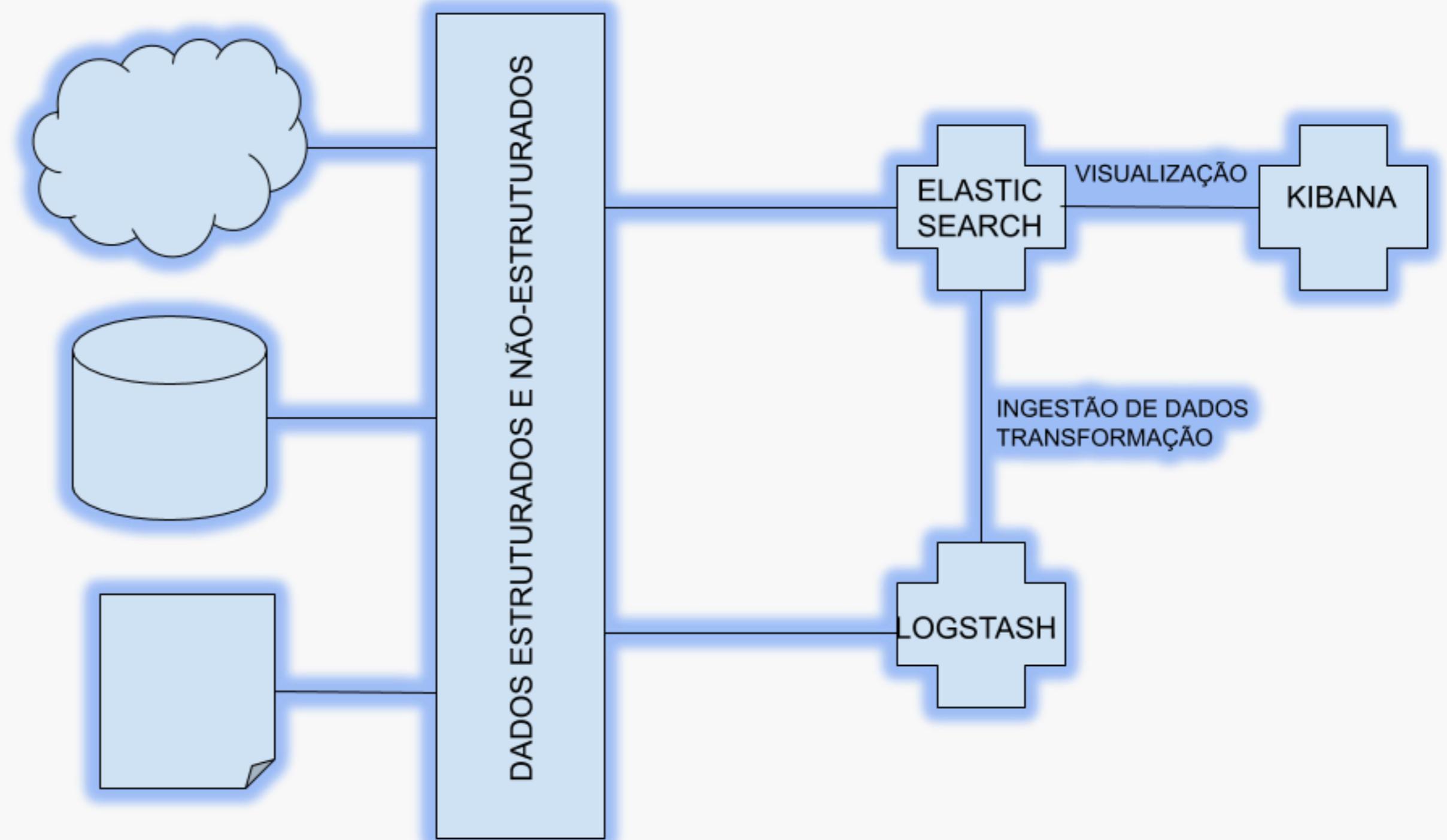
[https://www.elastic.  
co/pt/logstash](https://www.elastic.co/pt/logstash)

## KIBANA

[https://www.elastic.  
co/pt/kibana](https://www.elastic.co/pt/kibana)

# Ecossistema

Como funcionam as ferramentas para  
ingestão de dados , visualização de  
dados, monitoramento e DEV TOOLS



# Elasticsearch

O coração do Elastic Stack

## Elasticsearch

É o componente principal do Elastic Stack, permite que você armazene os dados e possibilite todos os recursos necessários para que sejam realizadas as análises dos dados. Além disso permite a busca por elementos armazenados.



# elasticsearch

# Logstash

O que realiza a ingestão de dados

## Logstash

Logstash é conhecido no ecossistema do Elastic Stack como o meio de coleta de dados. O Logstash pode trabalhar com vários tipos de dados, transformar dados e colocar diretamente no Elasticsearch. É originalmente usado para coleta de log, mas pode ser utilizado para leitura de diversos tipos de dados.



logstash

# Kibana

É o responsável pela visualização de dados e monitoramento

## Kibana

É responsável pela busca e visualização de dados no Elasticsearch. Ele fornece recursos de visualização sobre o conteúdo indexado no Elasticsearch. Os usuários podem criar gráficos de diversos formatos: barra, linha e dispersão, mapas, etc. Também possui as funções de monitoramento, gerenciamento e DEV TOOLS.



# kibana

# Qual é a ideia do nosso curso

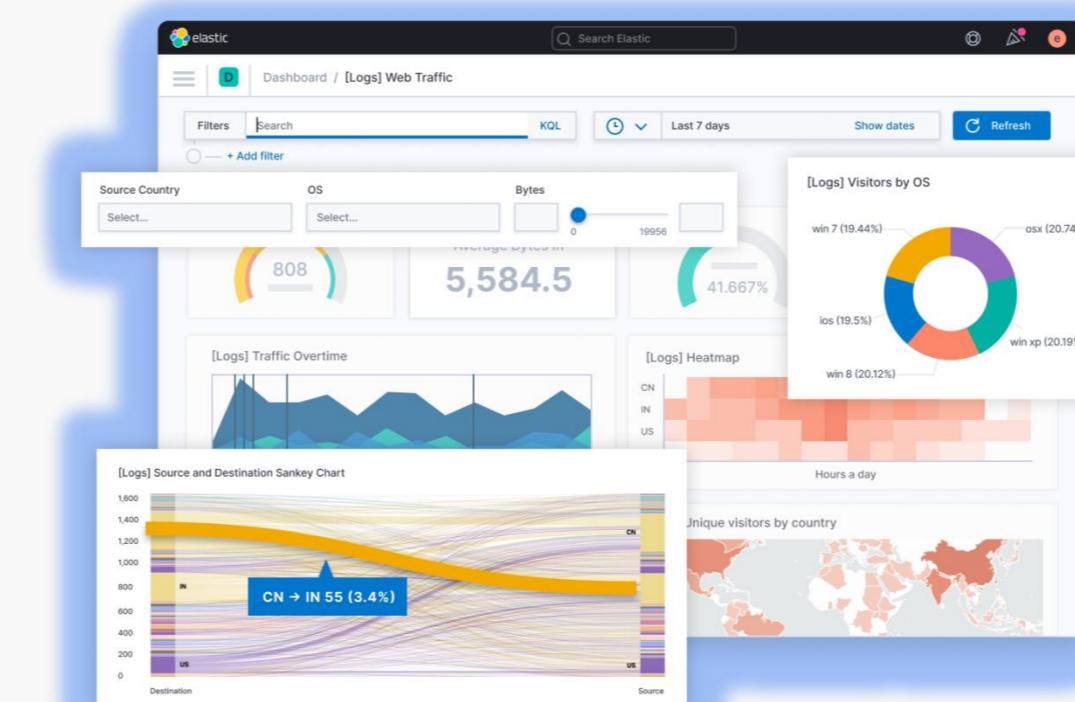
Veja o que pretendemos criar

Teremos que ter em mente, que iremos carregar uma lista de dados utilizando o **Logstash**, que em seguida, será armazenada no **Elastic Search** e consequentemente poderemos visualizar dentro do **Kibana**.



## Executa Script

```
PUT /my-index-000001
{
  "mappings": {
    "properties": {
      "@timestamp": {
        "type": "date"
      },
      "aborted_count": {
        "type": "long"
      },
      "another_field": {
        "type": "keyword" ①
      },
      "clientip": {
        "type": "keyword"
      }
    }
  }
}
```



## VISUALIZAÇÃO

# Pontos Principais Elastic Search

1. Mecanismo de busca e análise
  2. Código aberto
  3. Baseado no Apache Lucene
  4. Utiliza API RESTful para execução de ordens
  5. Utilizados por gigantes: wikipedia, google, dentre outros



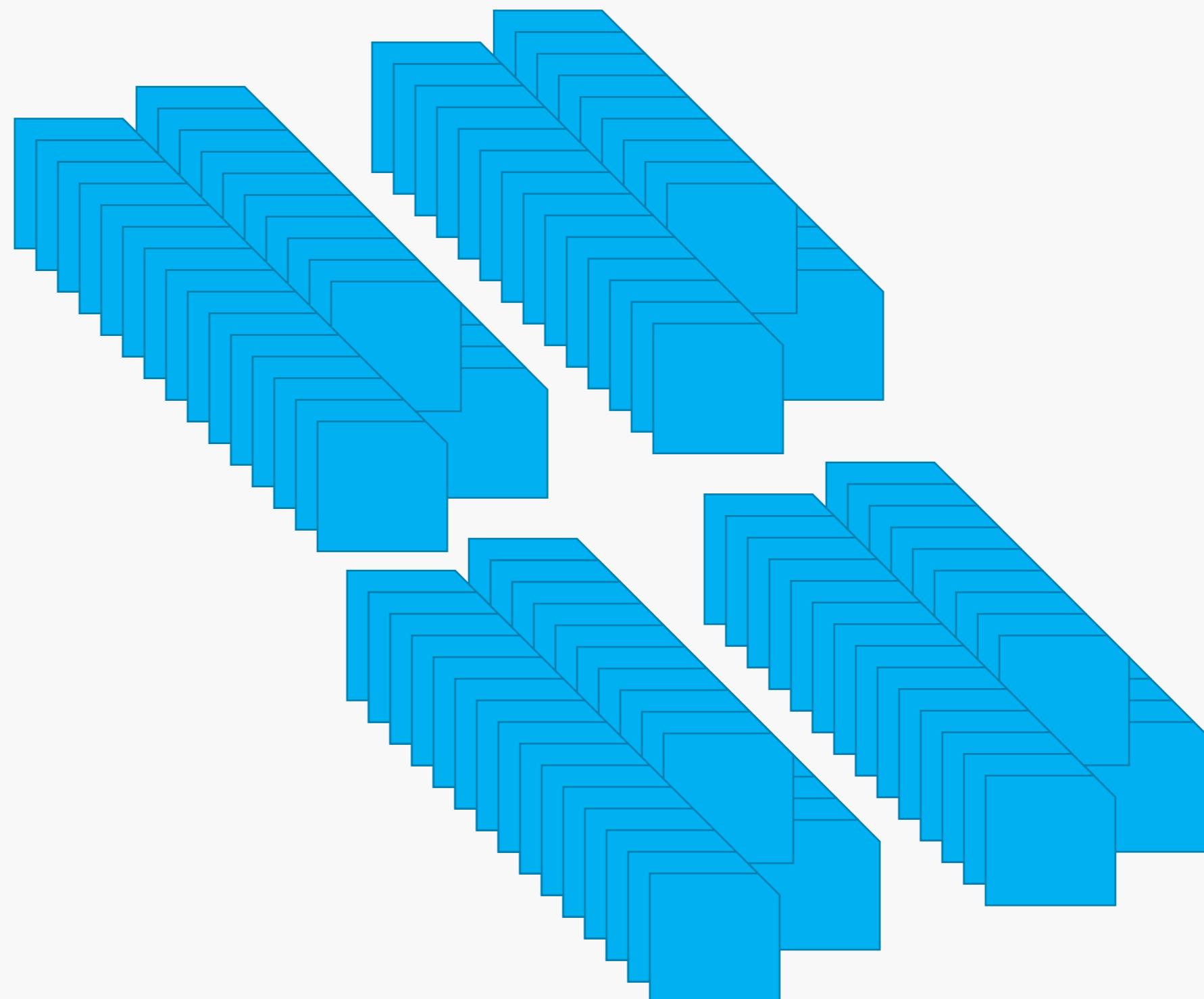
# Qual é o ponto forte do Elasticsearch?

Entendendo como ele trabalha

## Baseado em documentos

Todos os dados inseridos serão armazenados como documentos. Estes documentos uma vez lidos são armazenados no formato JSON que permitirá buscas em seu conteúdo via comandos.

Documentos



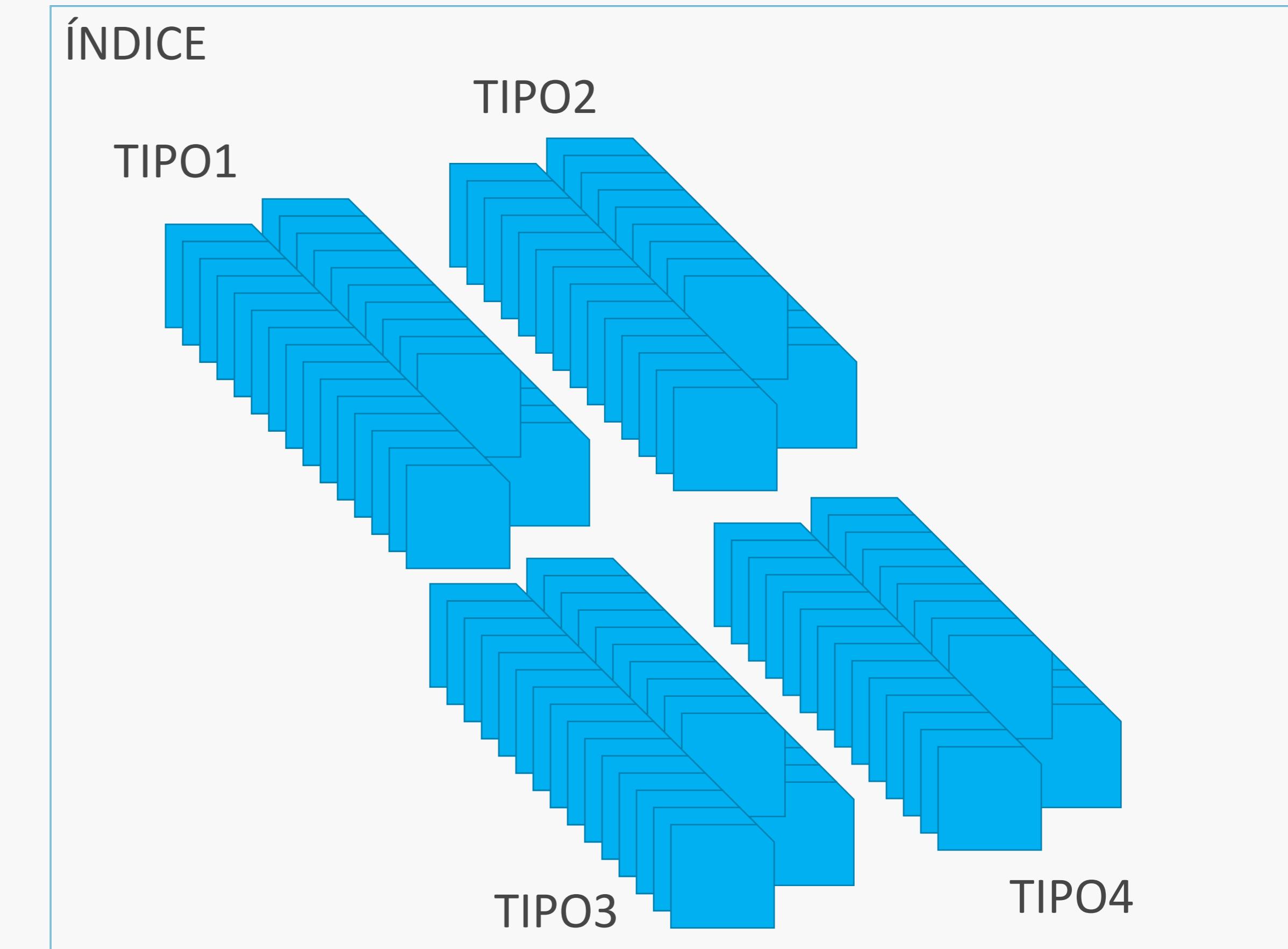
# O que é gerenciado?

Entendendo o controle sobre os documentos

## Como gerenciamos os documentos

Todos os documentos recebem uma nomenclatura que chamamos de **TIPO**, um ou mais TIPOS são guardados em **ÍNDICES**.

**Analogia:** ÍNDICES-> BANCO DE DADOS ; TIPOS-> TABELAS;  
DOCUMENTOS->REGISTROS ; AS COLUNAS DOS DOCUMENTOS->  
CAMPOS

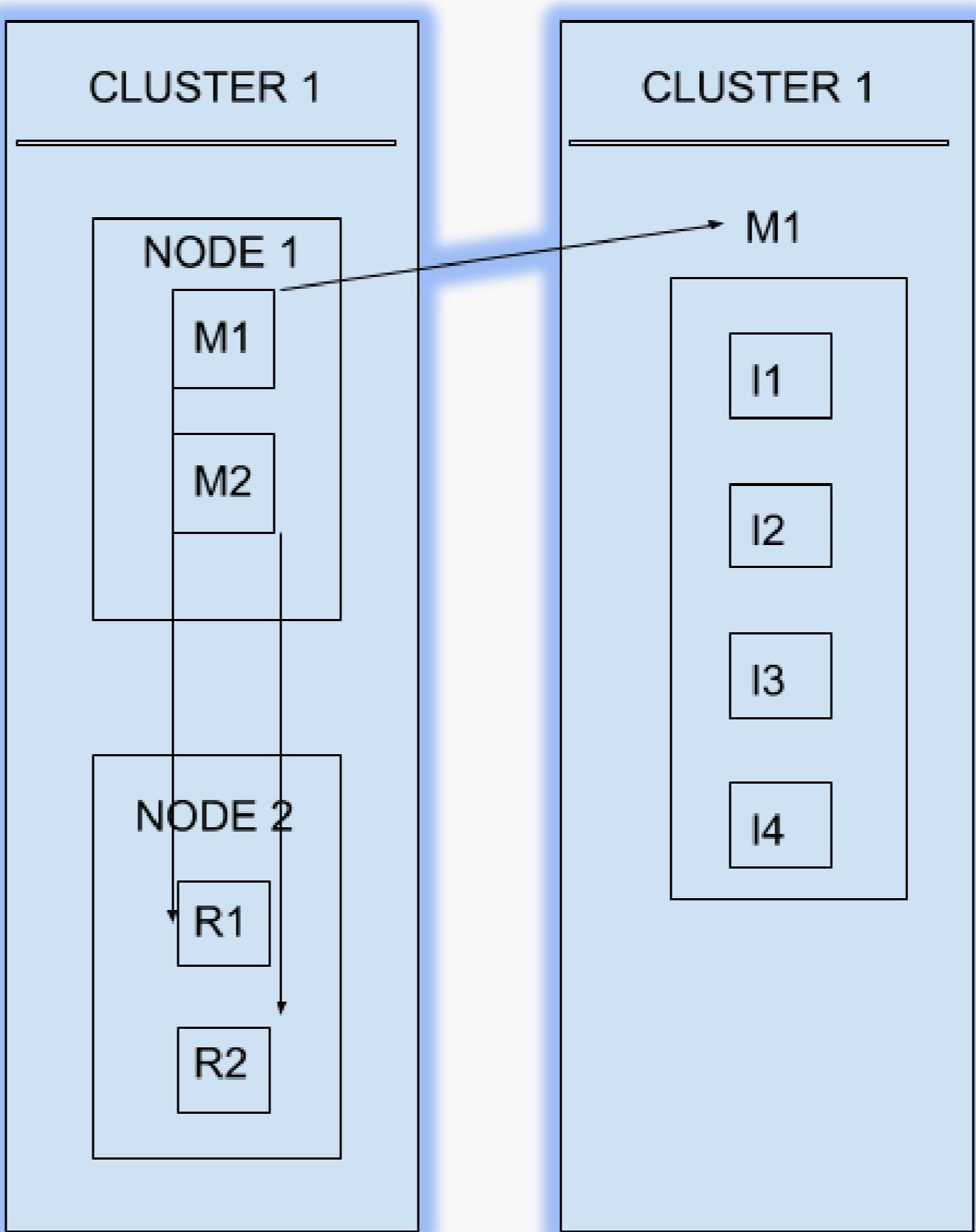


# A guarda dos dados no Elasticsearch

Entendendo como ele gerencia os dados

## Shards

É a estrutura de dados mais elementar que existe no Elasticsearch, ele que concentra as partições dos índices dentro de um *cluster*. Todo o gerenciamento é por meio de nós(NODE), que representa de forma efetiva a alocação e de quanto é utilizado o espaço no disco de dados. Há réplica dos dados em caso de excesso de consumo em determinado shards e em caso de perda.



CLUSTER1 -> DISCO DE ARMAZENAMENTO  
NODE1,NODE2-> ARMAZENA OS SHARDS  
M1,M2 -> SÃO OS SHARDS  
R1,R2 -> SÃO AS RÉPLICAS  
I1,I2,I3,I4 -> SÃO OS ÍNDICES

\* SÓ EXISTE UM NÓ MESTRE DENTRO DE UM CLUSTER.

# A força do Elasticsearch!

Entendendo como se busca  
documentos

## Busca Request Body

O Elasticsearch já vem com uma API REST, por padrão, que recebe requisições dos quatro verbos HTTP: GET, POST, PUT e DELETE. Você necessitará fazer uma requisição HTTP à API do Elasticsearch por meio do envio de objetos JSON.

```
POST http://localhost:9200/brasil/presidente
{
  "nome": "Juscelino Kubitschek",
  "moradia": "Brasília",
  "estado civil": "Casado"
}
```

## Resultado

Após o endereço e porta (<http://localhost:9200>), informamos que desejamos indexar o **documento** do tipo presidente no **índice** brasil.

# A força do Elasticsearch!

Entendendo como se busca documentos

## Busca Request Body

O Elasticsearch já vem com uma API REST, por padrão, que recebe requisições dos quatro verbos HTTP: GET, POST, PUT e DELETE. Você necessitará fazer uma requisição HTTP à API do Elasticsearch por meio do envio de objetos JSON.

**GET**

**http://localhost:9200/brasil/presidente/\_search**

```
{  
  "query": {  
    "match": {  
      "nome": {  
        "query": "Juscelino"  
      }  
    }  
  }  
}
```

## Resultado

Por meio do verbo **GET** informamos que desejamos realizar uma busca através da declaração **\_search** após o índice e o documento. Definimos que estamos buscando por um documento que tenha o seu campo **nome** a query igual a **Juscelino**

# A força do Elasticsearch!

Entendendo como se busca documentos

## Busca URI

Geralmente são buscas mais simples com parâmetros nas consultas diretamente .

```
curl "localhost:9200/_search?q=nome:Juscelino"
```

## Resultado

A consulta acima procurará documentos de todo o seu cluster com um campo de nome igual a “Juscelino”

# A força do Elasticsearch!

Entendendo como se busca  
documentos

## Como funciona as pesquisas

- **Analisadores:** Quando um documento é indexado no Elasticsearch, os conteúdos são separado em “termos” de busca, conhecidos como tokens, para facilitar o encontro dos dados. Este processo ocorre em campos do tipo **Text**.

Toda a busca requer o uso de um processo de escolhas e filtragem para encontrar o conteúdo pesquisado.

- ❑ **Character filters:** Irá receber o texto como um stream de caracteres e então pode realizar a transformação desse stream, excluindo, adicionando ou alterando caracteres.
- ❑ **Tokenizer:** Irá quebrar todo o texto em termos, separando termo a termo.
- ❑ **Token filters:** Pode-se alterar, excluir ou incluir novos tokens.

## Existe a possibilidade de não usar analisador

Caso você deseje pesquisar o conteúdo: “Vamos para praia”. Se você optar na opção de indexação do Elasticsearch por **not\_analyzed**, o retorno será exatamente o que foi digitado. Mas se optar pela opção **analyzed**, que é mais comum, o retorno poderia ser resultado dos seguintes tokens: “Vamos”, “para”, “praia”.

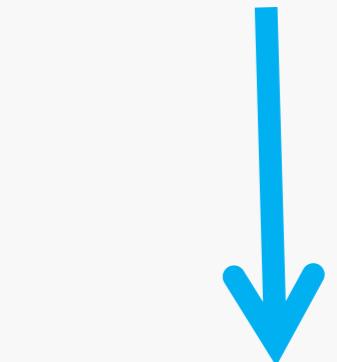
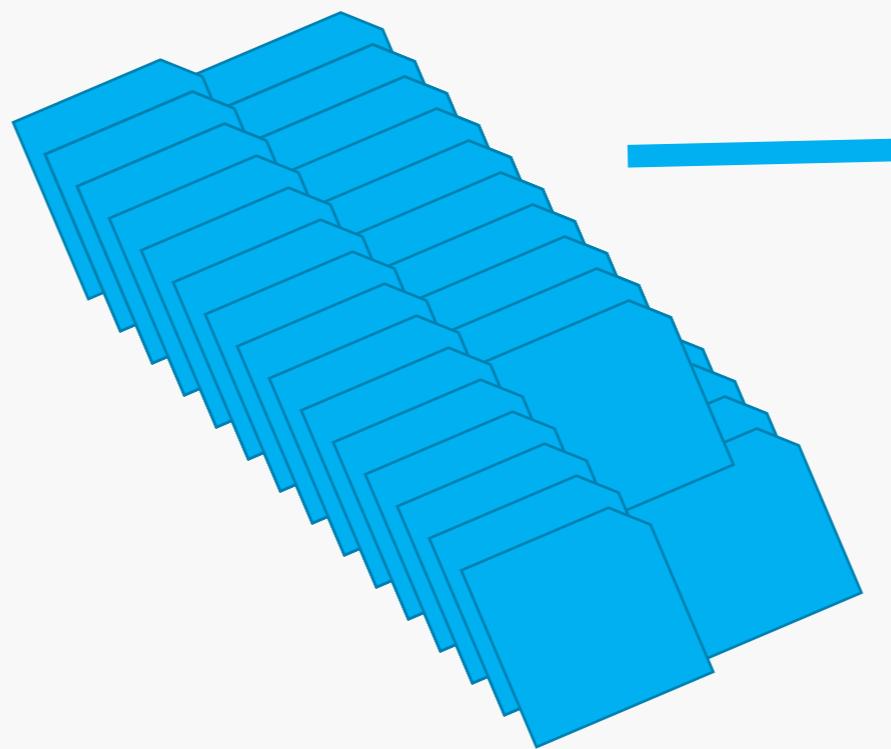
# Vamos começar!

Instalação dos programas

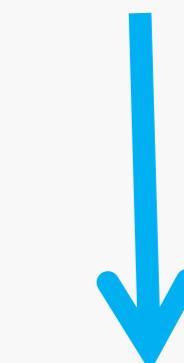
## Instalação

Faremos a instalação do Elasticsearch, logstash, kibana

Documentos



elasticsearch



kibana

## Carregando os dados!

Vamos utilizar o logstash com um script para você carregar os dados.

### 1- Procurar o dataset (arquivo)

[Kaggle: Your Home for Data Science](https://www.kaggle.com/sakshigoyal7/credit-card-customers)

### 2- download e preparação



<https://www.kaggle.com/sakshigoyal7/credit-card-customers>

### 3- Carga no logstash

Criar a pasta -> config

...logstash-7.10.0-windows-x86\_64\logstash-7.10.0\bin

Criar o script -> bancocliente.conf

# Carregando os dados!

Vamos utilizar o logstash com um script para você carregar os dados.

## 4- Informações do Script bancocliente.conf

```
input {
  file {
    path => "C:/Temp/BankChurners.csv"
    start_position => "beginning"
    since_db_path => "NUL"
  }
}

filter {
  csv {
    separator => ";"
    columns =>
      ['CLIENTNUM','Attrition_Flag','Customer_Age','Gender','Dependent_count','Education_Level','Marital_Status','Income_Category','Card_Category','Months_on_book','Total_Relationship_Count','Months_Inactive_12_mon','Contacts_Count_12_mon','Credit_Limit','Avg_Open_To_Buy','Total_Trans_Amt','Total_Trans_Ct']
  }
  mutate {
    convert => {
      "CLIENTNUM" => "string"
      "Attrition_Flag" => "string"
      "Customer_Age" => "integer"
      "Gender" => "string"
      "Dependent_count" => "integer"
      "Education_Level" => "string"
      "Marital_Status" => "string"
      "Income_Category" => "string"
      "Card_Category" => "string"
      "Months_on_book" => "integer"
      "Total_Relationship_Count" => "integer"
      "Months_Inactive_12_mon" => "integer"
      "Contacts_Count_12_mon" => "integer"
      "Credit_Limit" => "float"
      "Avg_Open_To_Buy" => "float"
      "Total_Trans_Amt" => "float"
      "Total_Trans_Ct" => "float"
    }
  }
}

output {
  stdout {
    codec => rubydebug
  }
}
```

## Carregando os dados!

Vamos utilizar o logstash com um script para você carregar os dados.

## 5- executar em linha de comando

```
logstash-7.10.0-windows-x86_64\logstash-7.10.0\bin>logstash -f config/bancocliente.conf
```

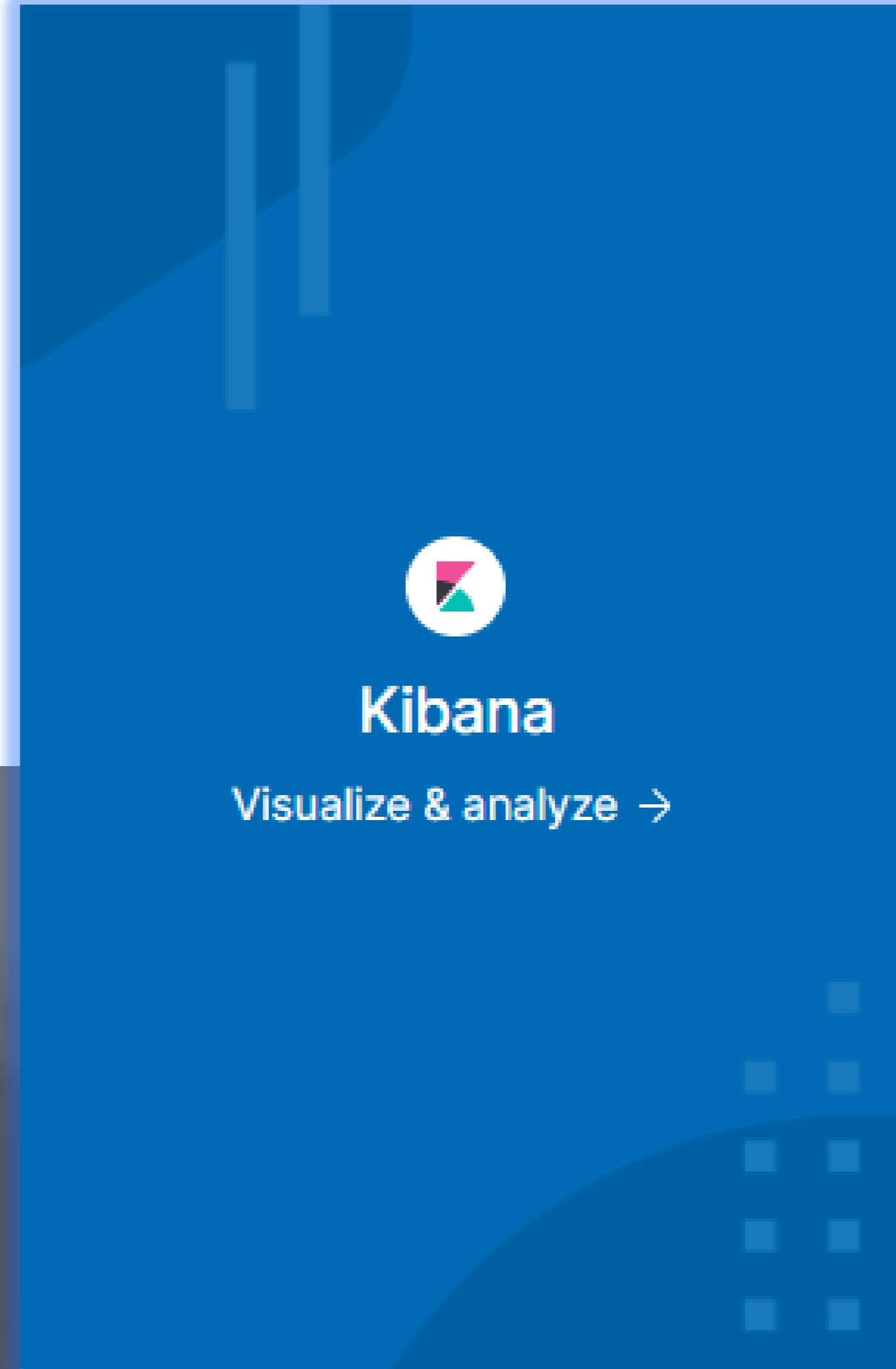
Logstash -f config/bancocliente.conf

## 6- Depurando o resultado na tela

```
{
    "Total_Trans_Amt" => 14999.0,
    "message" => "715923408;Existing Customer;30;F;0;College;Married;Less than $40K;Blue;19;4;2;3;4184;2930;14999;110\r",
    "Attrition_Flag" => "Existing Customer",
    "Total_Relationship_Count" => 4,
    "Card_Category" => "Blue",
    "Total_Trans_Ct" => 110.0,
    "host" => "Grimaldo",
    "Months_on_book" => 19,
    "Education_Level" => "College",
    "Marital_Status" => "Married",
    "Income_Category" => "Less than $40K",
    "Customer_Age" => 30,
    "Avg_Open_To_Buy" => 2930.0,
    "path" => "C:/Temp/BankChurners.csv",
    "Contacts_Count_12_mon" => 3,
    "@version" => "1",
    "@timestamp" => 2020-12-09T19:01:43.713Z,
    "Months_Inactive_12_mon" => 2,
    "Gender" => "F",
    "Dependent_count" => 0,
    "CLIENTNUM" => "715923408",
    "Credit_Limit" => 4184.0
}
```

1. Discover
2. Dashboard
3. Machine Learning
4. Visualize

## Pontos Principais Kibana



Analyze data in dashboards.

Search and find insights.

Design pixel-perfect presentations.

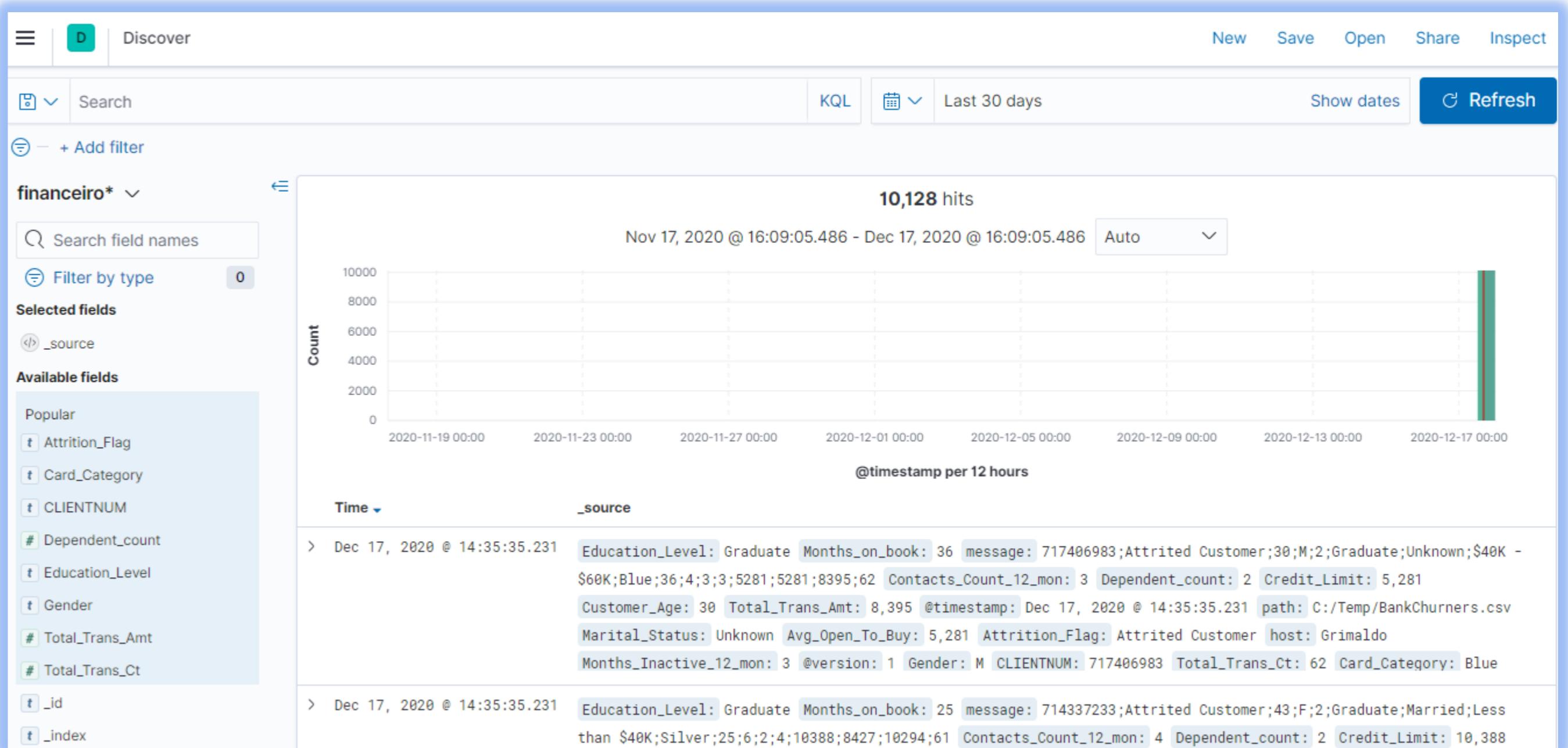
Plot geographic data.

Model, predict, and detect.

# Vamos descobrir os dados!

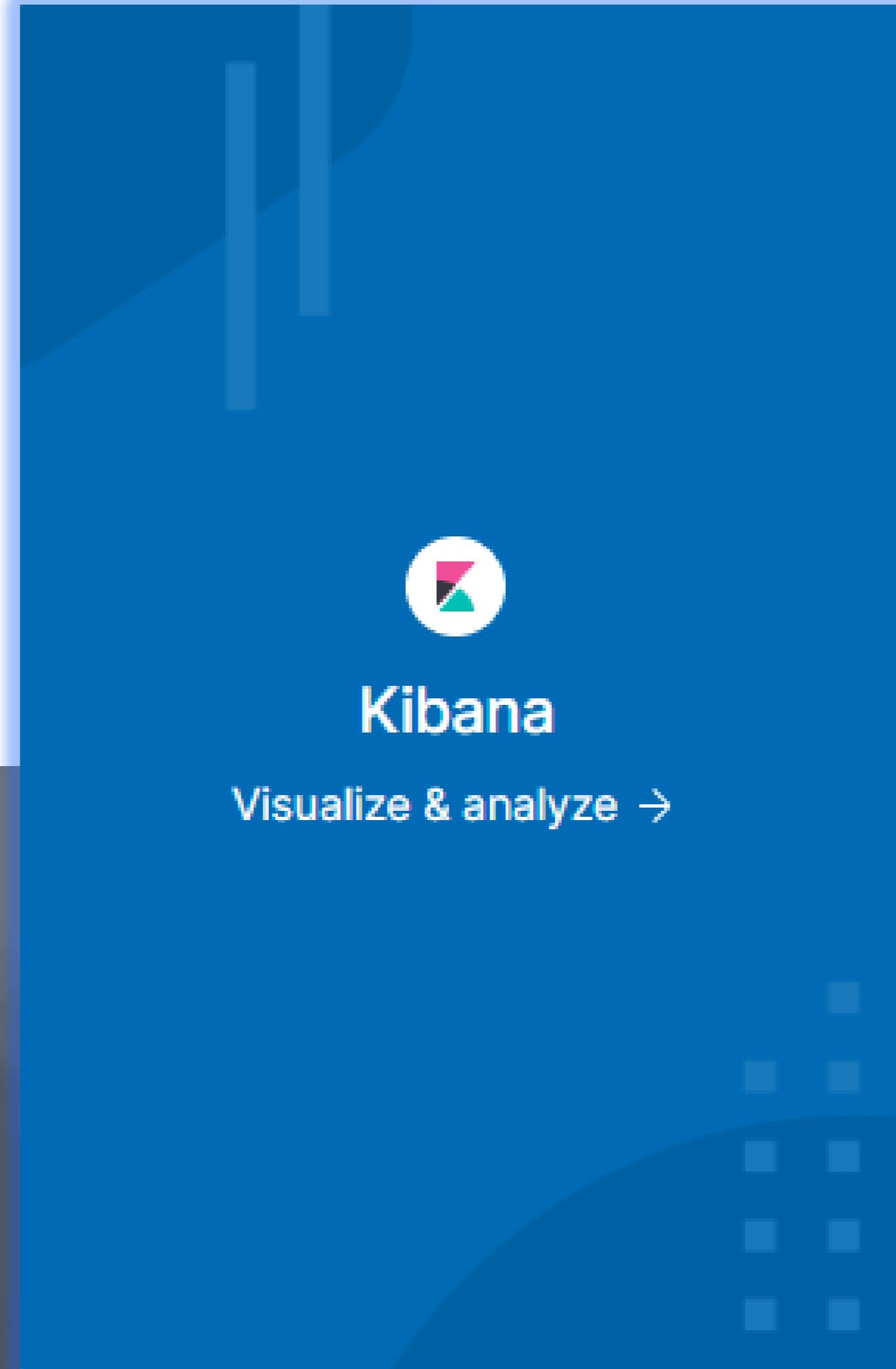
Vamos realizar perguntas via Kibana.

É possível realizar filtros, escolha de campos, verificação de estatísticas de inserção de campos, detalhes da pesquisa dos dados, etc.



1. Discover
2. Dashboard
3. Machine Learning
4. Visualize

## Pontos Principais Kibana



Analyze data in dashboards.

Search and find insights.

Design pixel-perfect presentations.

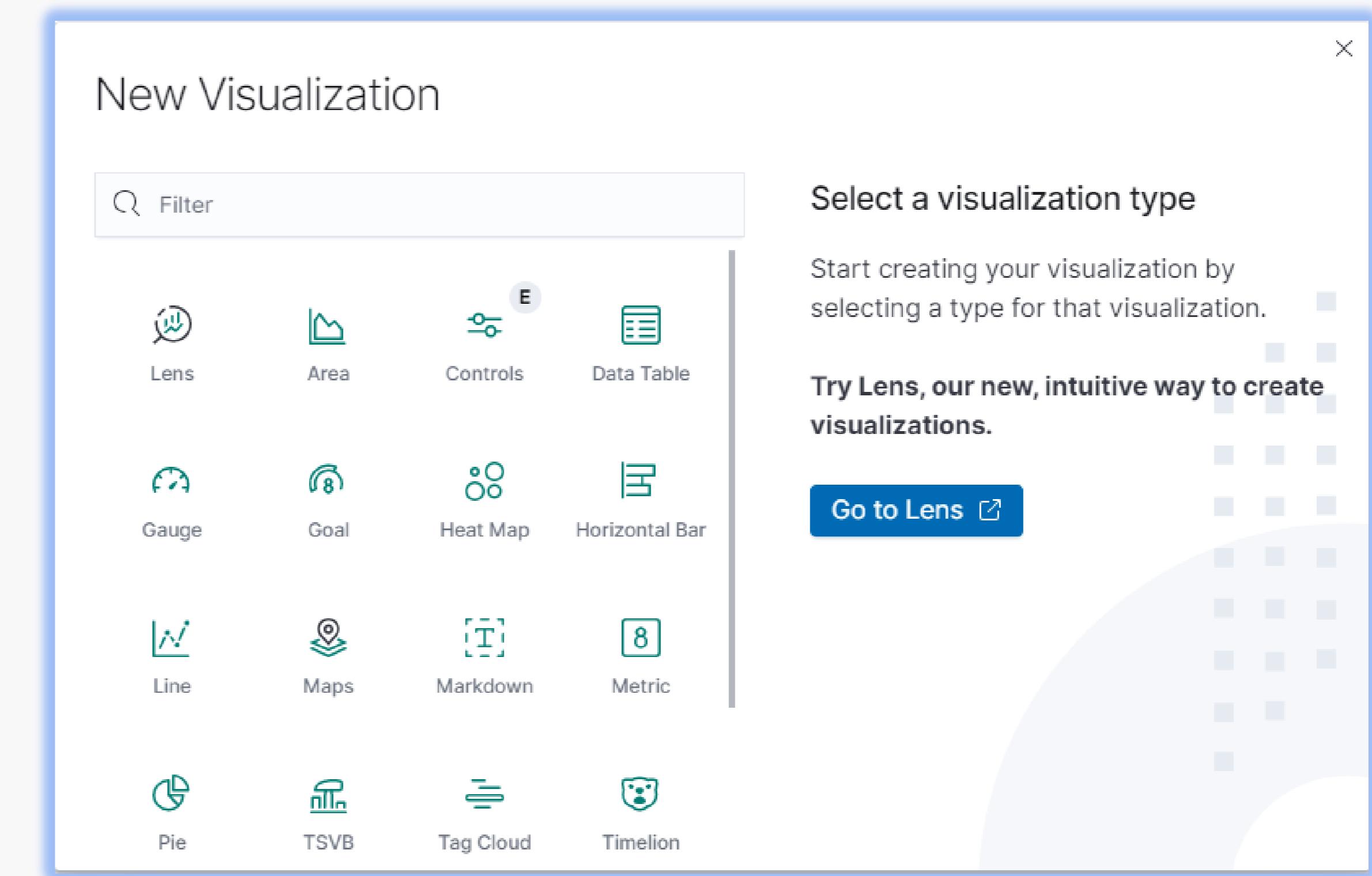
Plot geographic data.

Model, predict, and detect.

# Vamos criar visualizações com os dados

O primeiro passo para construirmos dashboards.

## Construindo visualizações com os dados



# Vamos criar visualizações com os dados

Vamos criar algumas visualizações para criar um dashboard.

A primeira Visualização será do tipo TABELA

The screenshot displays a data visualization interface with four tables side-by-side:

- Married: Estado Civil**: Shows counts for Males (M) and Females (F) across categories like Silver, Platinum, Gold, and Blue, further broken down by marital status (Married, Single).
- Single: Estado Civil**: Shows counts for Males (M) and Females (F) across categories like Silver, Platinum, Gold, and Blue, further broken down by marital status (Single).
- Divorced: Estado Civil**: Shows counts for Males (M) and Females (F) across categories like Silver, Platinum, Gold, and Blue, further broken down by marital status (Divorced).
- Unknown: Estado Civil**: Shows counts for Males (M) and Females (F) across categories like Silver, Platinum, Gold, and Blue, further broken down by marital status (Unknown).

An export dialog box is visible, containing fields for "Search" and "NOT Marital\_Status: Unknown".

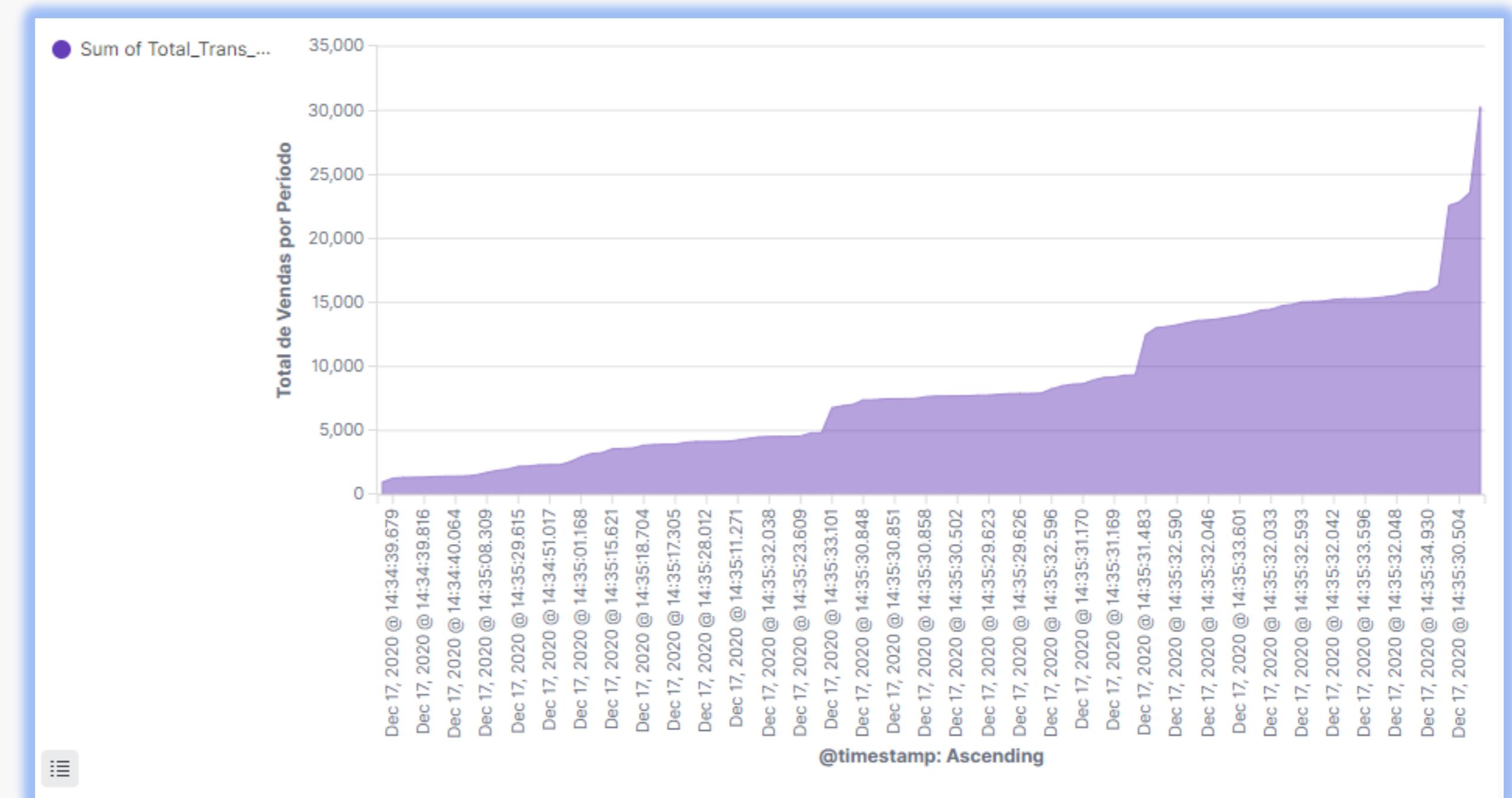
A modal window titled "Married: Estado civil" is open, showing the top 4 unusual terms and their percentages:

Customer_Age: Descending	Education_Level.keyword: Descending	Gender.keyword: Descending	Estado civil: Descending	Income_Category.keyword: Count	Count percentages
49	Unknown	M	Married	\$120K +	3 0.335%
49	College	M	Married	\$60K - \$80K	3 0.335%
49	College	M	Married	\$80K - \$120K	3 0.335%
44	High School	M	Married	\$120K +	3 0.335%
44	Unknown	M	Married	\$120K +	3 0.335%
44	Uneducated	M	Married	\$60K - \$80K	3 0.335%
44	Uneducated	M	Married	\$40K - \$60K	3 0.335%
44	College	M	Married	\$40K - \$60K	3 0.335%
46	Graduate	M	Married	\$120K +	3 0.335%
46	College	F	Married	Unknown	3 0.335%
45	Graduate	M	Married	\$120K +	3 0.335%
45	High School	M	Married	\$40K - \$60K	3 0.335%
45	Unknown	M	Married	\$60K - \$80K	3 0.335%

# Vamos criar visualizações com os dados

Vamos criar algumas visualizações para criar um dashboard.

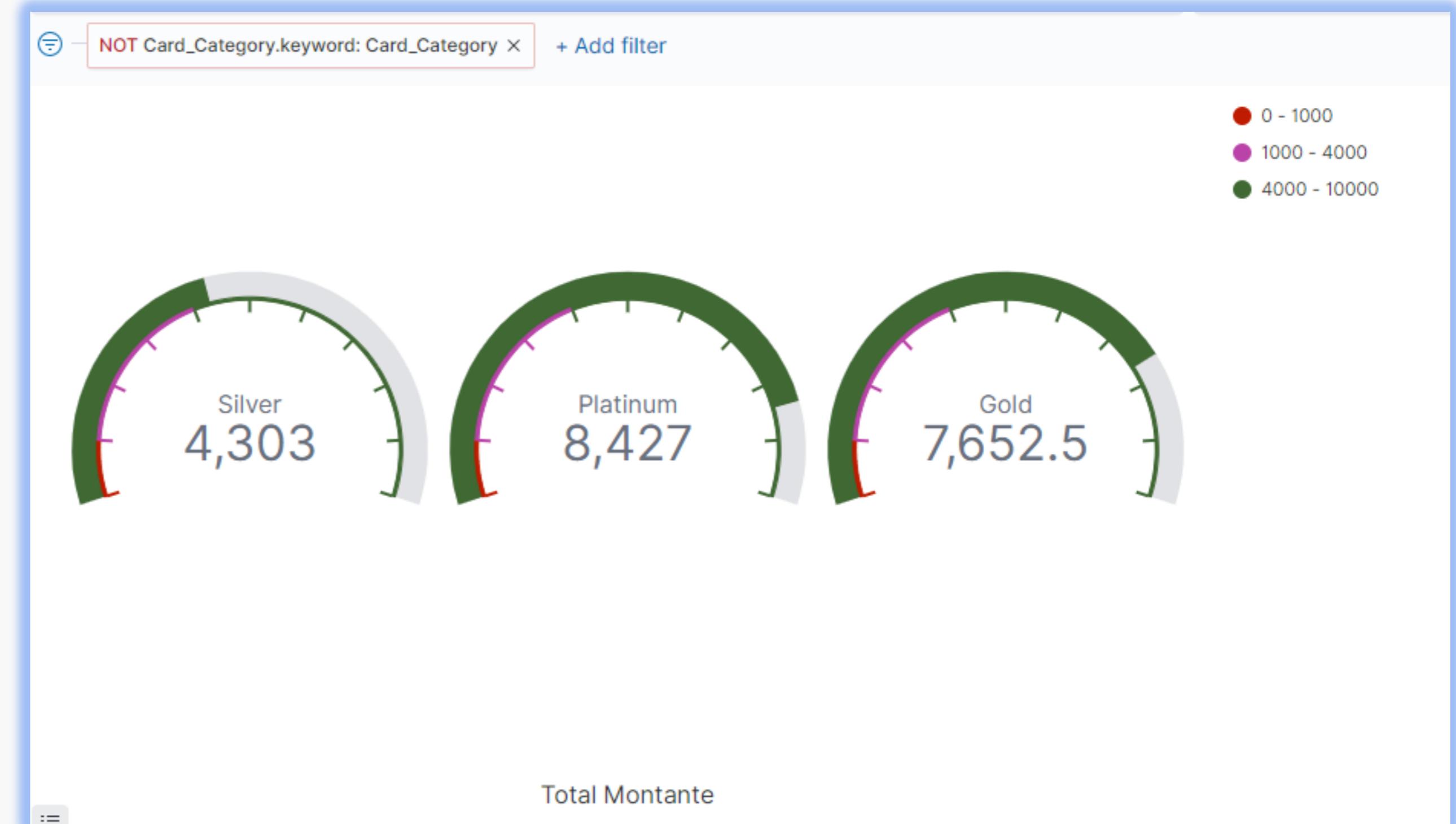
A visualização será do tipo ÁREA



# Vamos criar visualizações com os dados

Vamos criar algumas visualizações para criar um dashboard.

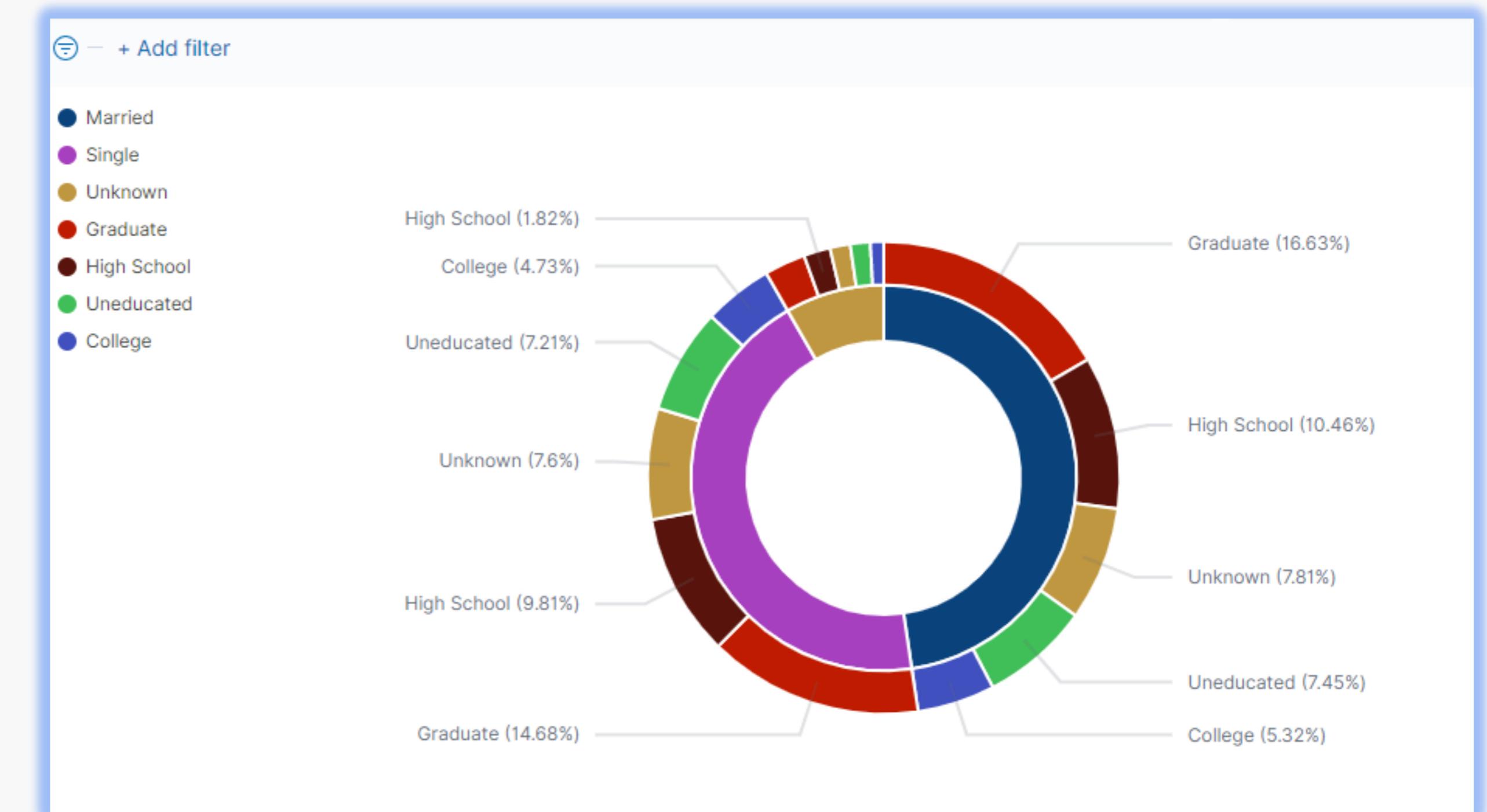
A visualização será do tipo **GAUGE**



# Vamos criar visualizações com os dados

Vamos criar algumas visualizações para criar um dashboard.

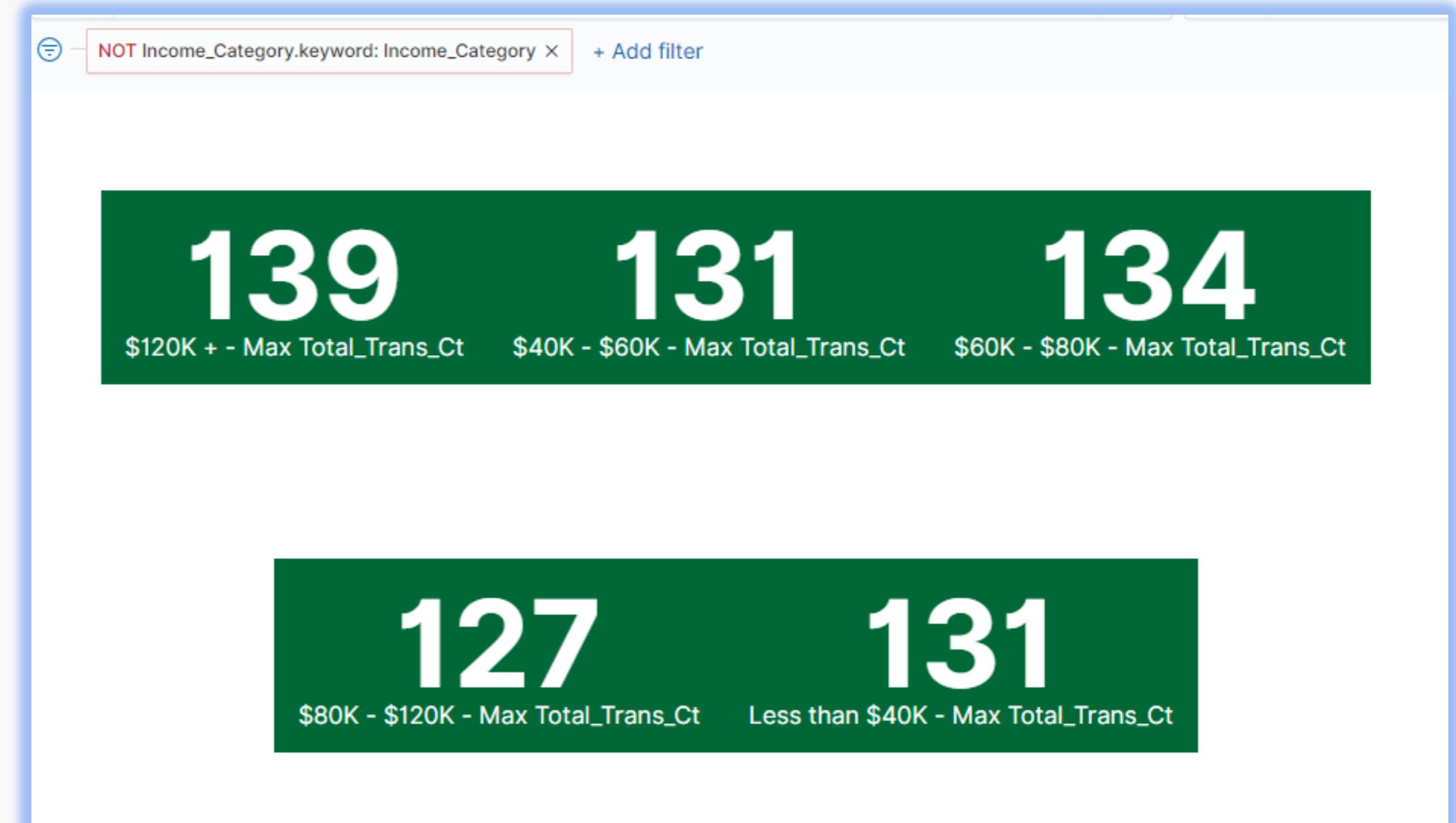
A visualização será do tipo Setores



# Vamos criar visualizações com os dados

Vamos criar algumas visualizações para criar um dashboard.

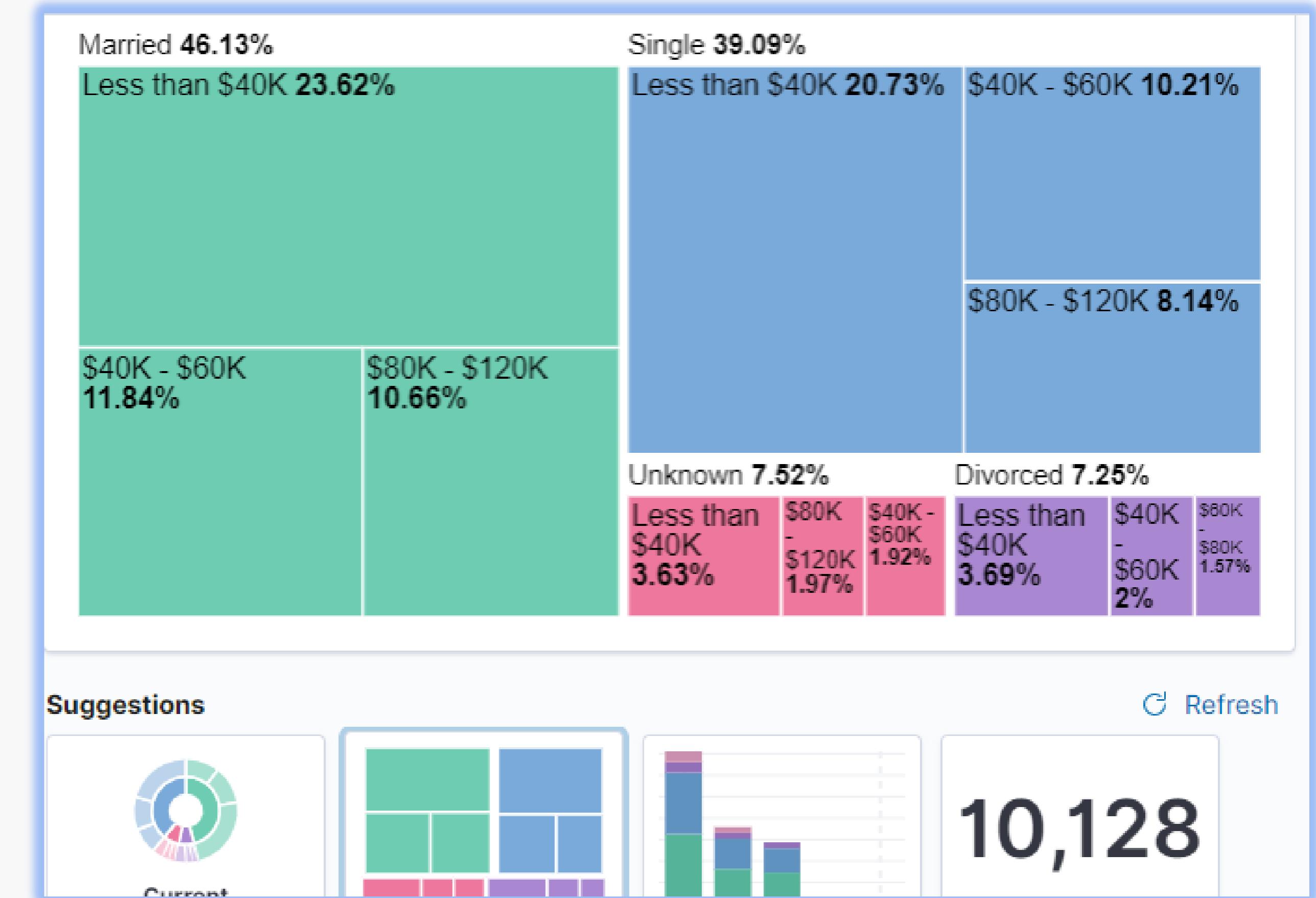
A visualização será do tipo Metric (KPI)



# Vamos criar visualizações com os dados

Vamos criar algumas visualizações para criar um dashboard.

A visualização será do tipo Lens

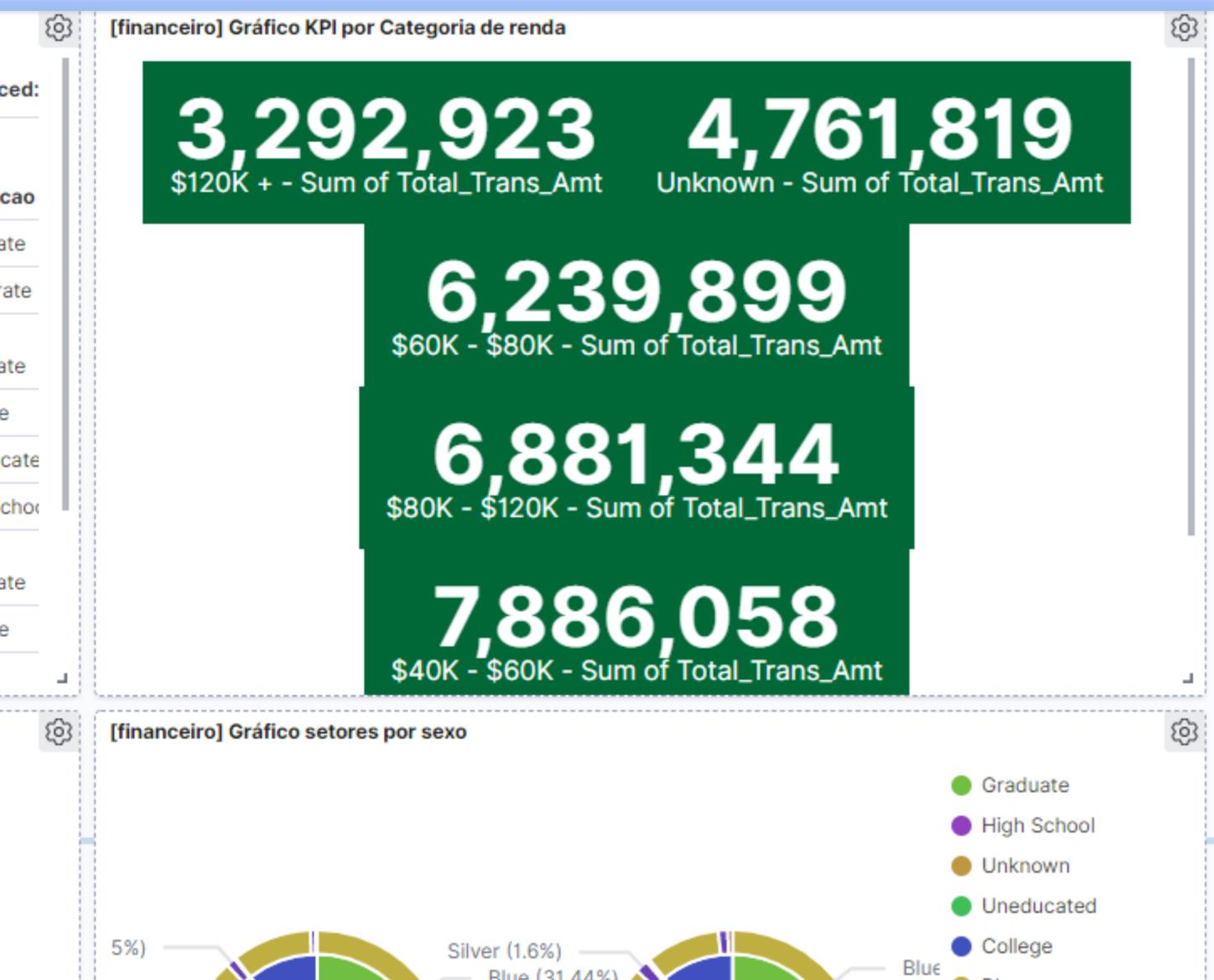


# Vamos construir nosso dashboard

Vamos criar um dashboard com as nossas visualizações.

# Ambiente de construção dashboard

Married: Estado Civil				Unknown: Estado Civil				Divorced: Estado Civil			
Sexo	Formacao	Estado Civil	Count	Sexo	Formacao	Estado Civil	Count	Sexo	Formacao	Estado Civil	Count
F	Graduate	Married	778	F	Graduate	Unknown	126	F	Graduate	Unknown	126
F	Doctorate	Married	126	F	Doctorate	Unknown	14	F	Doctorate	Unknown	14
F	Post-Graduate	Married	126	F	Post-Graduate	Unknown	19	F	Post-Graduate	Unknown	19
F	College	Married	239	F	College	Unknown	37	F	College	Unknown	37
F	Uneducated	Married	342	F	Uneducated	Unknown	55	F	Uneducated	Unknown	55
M	High School	Married	469	M	High School	Unknown	81	M	High School	Unknown	81
M	Post-Graduate	Married	117	M	Post-Graduate	Unknown	24	M	Post-Graduate	Unknown	24
M	College	Married	228	M	College	Unknown	37	M	College	Unknown	37
				2,425				393			



**Idade**

46 X    47 X |

44  
45  
49

**Faixa de gastos**

6039

0    18484

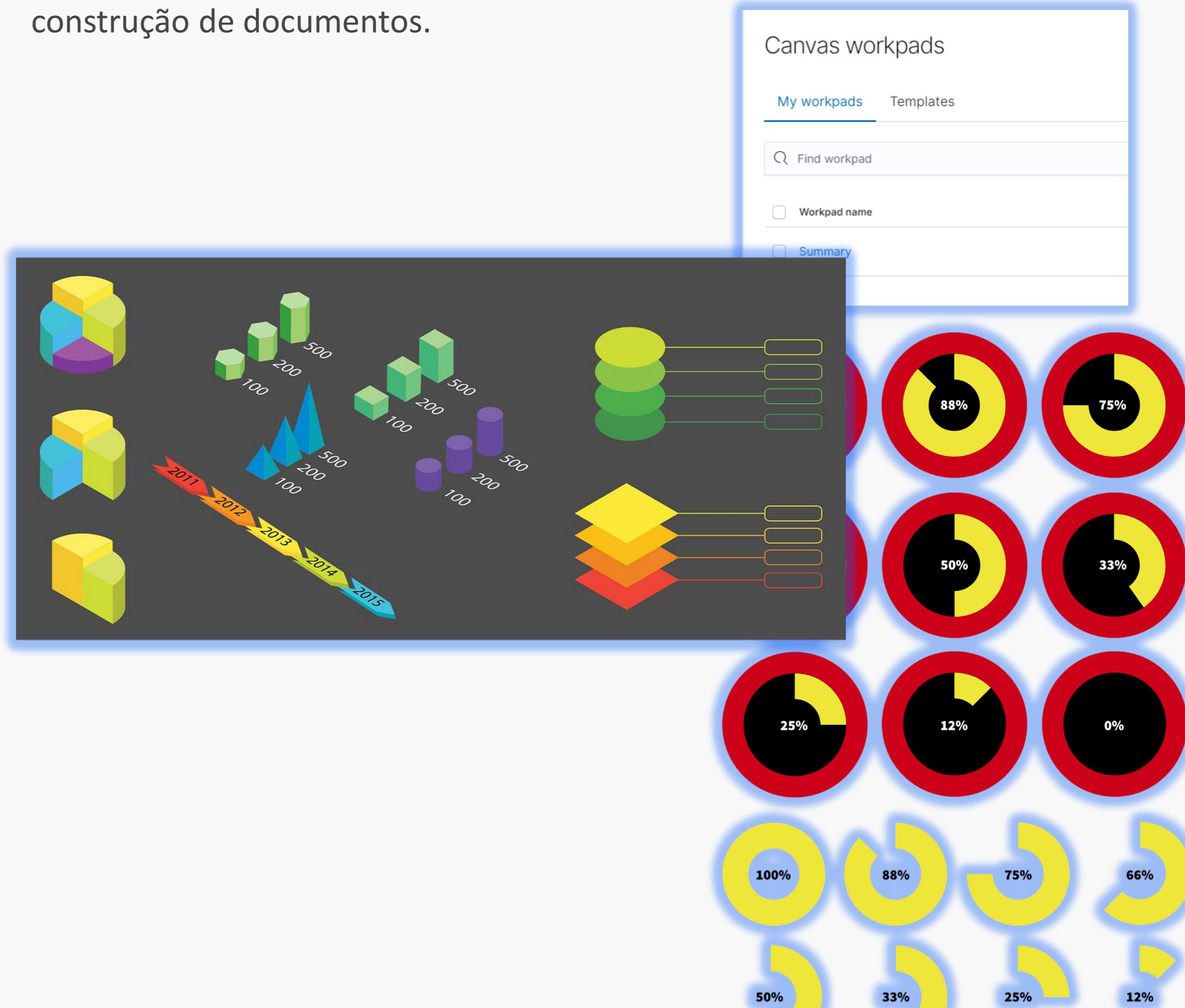
18484

# Canvas

# Construção de projetos de designer com o Elasticsearch

# Canvas

Ferramenta integrada ao Elasticsearch para que você possa desenvolver projetos gráficos como infográficos, elementos de designer, e construção de documentos.



# Linguagem DSL

Utilizamos a execução na opção

Management -> DEV TOOLS

## Linguagem DSL

A Elasticsearch fornece uma DSL de consulta completa (Linguagem Específica do Domínio) com base no JSON para definir consultas aos dados. Veja exemplo abaixo.

```
GET /financeiro/_search
{
  "query": {
    "bool": {
      "must": [
        {
          "match": {
            "Education_Level": "Graduate"
          }
        },
        {
          "should": [
            {
              "match": {
                "Gender": "M"
              }
            }
          ]
        },
        {
          "must_not": [
            {
              "match": {
                "Card_Category": "Blue"
              }
            }
          ]
        }
      ]
    }
  }
}
```

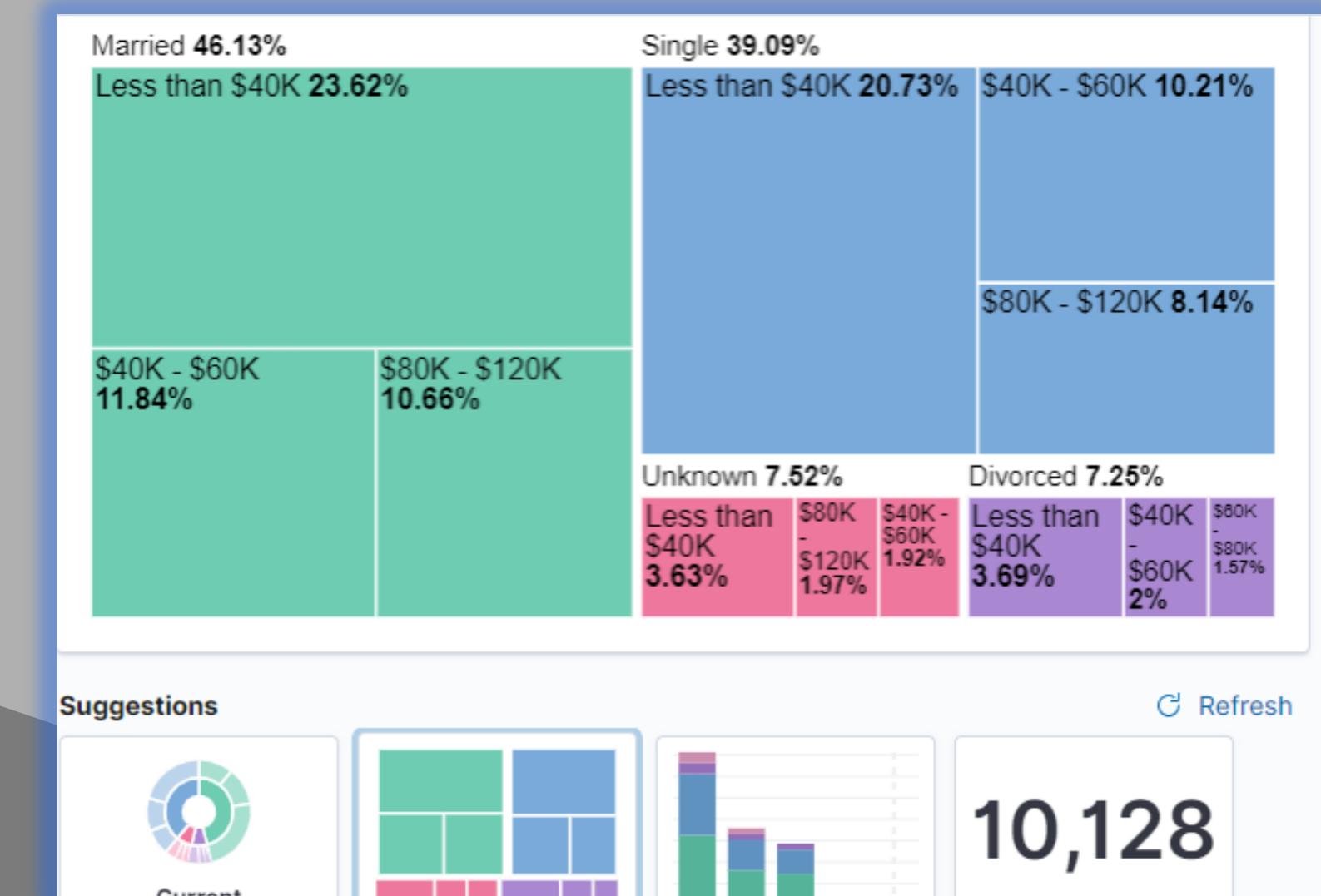
# PRÁTICA

**ENVIE AO PROFESSOR**

## PREPARE UM ESTUDO

Carregando qualquer arquivo do site **kaggle.com** e crie visualizações, dashboards e análises pela uso do Dev tools.

Muito boa sorte e conte comigo!



```
GET /financeiro/_search
{
  "query": {
    "bool": {
      "must": [
        {
          "match": {
            "Education_Level": "Graduate"
          }
        },
        {
          "should": [
            {
              "match": {
                "Gender": "M"
              }
            }
          ]
        },
        {
          "must_not": [
            {
              "match": {
                "Card_Category": "Blue"
              }
            }
          ]
        }
      ]
    }
  }
}
```