

# Visualizando Resultados ECL

Equipe de documentação de Boca Raton



## Visualizando Resultados ECL

Equipe de documentação de Boca Raton

Copyright © 2023 HPCC Systems®. All rights reserved

Sua opinião e comentários sobre este documento são muito bem-vindos e podem ser enviados por e-mail para <docfeedback@hpccsystems.com>

Inclua a frase **Feedback sobre documentação** no campo assunto e indique o nome do documento, o número das páginas e número da versão atual no corpo da mensagem.

LexisNexis e o logotipo Knowledge Burst são marcas comerciais registradas da Reed Elsevier Properties Inc., usadas sob licença.

HPCC Systems® HPCC Systems® é uma marca registrada da LexisNexis Risk Data Management Inc.

Os demais produtos, logotipos e serviços podem ser marcas comerciais ou registradas de suas respectivas empresas.

Todos os nomes e dados de exemplo usados neste manual são fictícios. Qualquer semelhança com pessoas reais, vivas ou mortas, é mera coincidência.

2023 Version 8.12.22-1

<i>Introdução a Visualização</i> .....	4
Instalação .....	5
Usando a biblioteca de Visualização .....	6
Exibindo a Visualização .....	7
Propriedades de Aparência .....	8
<i>Métodos</i> .....	9
Métodos de Visualização Bidimensional .....	10
Métodos de Visualização Multidimensional .....	11
Método de Visualização Geral .....	12
Método de Visualização Geoespacial .....	13
<i>Métodos Bidimensionais</i> .....	14
Bolha .....	15
Pizza .....	16
Resumo .....	17
Nuvem de Palavras .....	18
<i>Métodos Multidimensionais</i> .....	19
Área .....	20
Barras .....	21
Colunas .....	22
Hexágono .....	23
Linha .....	24
Dispersão .....	25
Escada .....	26
<i>Métodos Geoespaciais</i> .....	27
Estados do EUA .....	28
Condados do EUA .....	29
Euro .....	30
<i>Métodos Gerais de Visualização</i> .....	31
Grade .....	32
Grade Handson .....	33

# ***Introdução a Visualização***

O pacote de Visualização é um complemento de código público à plataforma do HPCC que permite criar visualizações dos resultados das consultas gravadas no ECL.

As visualizações são um meio importante de transmissão das informações dos dados massivos. Uma boa representação visual pode ajudar a gerar uma análise acionável. Uma representação visualmente abrangente das informações pode ajudar a tornar o obscuro em algo mais óbvio.

As visualizações são um meio importante de transmissão das informações dos dados massivos. O processamento de big data é apenas uma parte da solução; também é preciso compreender a complexidade dos dados. Os métodos de visualização dos dados simplificam aquilo que é complexo.

O pacote Visualização aumenta a funcionalidade da plataforma do HPCC permitindo a esquematização dos dados em diagramas, gráficos e mapas para adicionar uma representação visual que possa ser facilmente compreendida.

Além disso, a estrutura de visualização subjacente suporta recursos avançados para permitir a combinação de gráficos para criar painéis interativos.

# Instalação

Para instalar o pacote de Visualização, use a interface de linha de comando do ecl.

```
ecl bundle install https://github.com/hpcc-systems/Visualizer.git
```

Para usar o comando "ecl bundle install <git url>", é preciso ter o git instalado e configurado em seu sistema. O Git precisa estar acessível ao usuário (no variável path).

# Usando a biblioteca de Visualização

Uma vez instalado você Importa (IMPORT) a biblioteca e chama qualquer método que seja adequado para a forma dos seus dados.

Os seguintes passos devem ser seguidos para criar uma visualização básica:

1. Criação de um dataset adequado.
2. Crie um dataset com um nome adequado para que a visualização possa localizar os dados.
3. Crie (e gere) a visualização, fazendo referência ao nome criado na etapa 2.

Para alterar a aparência da visualização, as propriedades visuais podem ser fornecidas durante a criação da visualização ou aplicadas no tempo de execução através do editor de propriedades visuais em tempo real.

Por exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5},
                {'History', 17},
                {'Geography', 7},
                {'Chemistry', 16},
                {'Irish', 26},
                {'Spanish', 67},
                {'Biologiy', 66},
                {'Physics', 46},
                {'Math', 98}],
              {STRING subject, INTEGER4 year});
data_example := OUTPUT(ds, NAMED('Chart2D__test'));
data_example;
viz_bubble := Visualizer.TwoD.Bubble('bubble',, 'Chart2D__test');
viz_bubble;
```

Este código cria um gráfico de bolha como mostrado abaixo:



Consulte Métodos para obter detalhes sobre as categorias dos métodos de visualização e todos os métodos disponíveis.

## Exibindo a Visualização

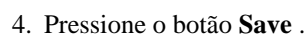
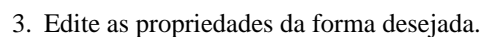
Após executar uma consulta com a visualização incluída, a visualização pode ser vista no ECL Watch. As visualizações são adicionadas às tarefas como *recursos* adicionais e podem ser vistas na guia **Resources** localizada na página Detalhes da Workunit.

Abra a workunit selecione a aba **Resources**.



O **Editor de visualização** permite alterar a aparência da visualização após ela ter sido executada. As mesmas propriedades podem ser especificadas no ECL no momento em que a visualização for criada.

1. Abra a workunit e seleciona a aba **Resources**.
2. Pressione o botão **Properties**.





# ***Métodos***

Os métodos são categorizados pela forma dos dados suportados. As categorias são:

Bidimensional

Multidimensional

Geoespacial

Geral

# Métodos de Visualização Bidimensional

Este MÓDULO contém uma seleção de visualizações de dados bidimensionais.

Bolha

Pizza

Resumo

Nuvem de palavras

# Métodos de Visualização Multidimensional

Este MÓDULO contém uma seleção de visualizações de dados multidimensionais.

Área

Barras

Colunas

Hexágono

Linha

Dispersão

Escada ou Degrau

# Método de Visualização Geral

Estados dos EUA

Condados dos EUA

Euro

EuroIE

EuroGB

Observação: Os métodos EuroIE e EuroGB são fornecidos como exemplos. Para criar outros métodos específico para um determinado país, basta usar o método Euro e fornecer o código do país de dois caracteres no parâmetro `_region`.

# Método de Visualização Geoespacial

Grade

Grade Handson

## ***Métodos Bidimensionais***

Esta seção abrange os métodos de criação de visualizações bidimensionais contidas no módulo TwoD (2D). Esses métodos são as formas de representação dos dados em um ambiente bidimensional.

## Bolha

**Visualizer.TwoD.Bubble**(( *id* , [*dataSource*], [*outputName*], [*filteredBy*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Mapeia rótulos humanos <--> Ids de campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O **método de visualização Bolha** cria um gráfico de bolhas a partir de dados bidimensionais. O gráfico de bolhas é uma variação do gráfico de pizza, onde os pontos dos dados são mostrados na forma de bolhas e o tamanho da bolha é representado pela segunda dimensão dos dados.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5},
                {'History', 17},
                {'Geography', 7},
                {'Chemistry', 16},
                {'Irish', 26},
                {'Spanish', 67},
                {'Biology', 66},
                {'Physics', 46},
                {'Math', 98}],
              {STRING subject, INTEGER4 year});
data_example := OUTPUT(ds, NAMED('Chart2D__test'));
data_example;
viz_bubble := Visualizer.TwoD.Bubble('bubble',, 'Chart2D__test');
viz_bubble;
```

## Pizza

**Visualizer.TwoD.Pie**(( *id* , [*dataSource*], [*outputName*], [*filteredBy*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Mapeia rótulos humanos <--> Ids de campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

**O método de visualização Pizza** cria um gráfico de pizza a partir de dados bidimensionais. O gráfico de pizza constitui em uma representação gráfica na forma de um círculo dividido em seções que representam a parte de um todo.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5},
                {'History', 17},
                {'Geography', 7},
                {'Chemistry', 16},
                {'Irish', 26},
                {'Spanish', 67},
                {'Biologiy', 66},
                {'Physics', 46},
                {'Math', 98}],
              {STRING subject, INTEGER4 year});
data_example := OUTPUT(ds, NAMED('Chart2D__test'));
data_example;
viz_pie := Visualizer.TwoD.Pie('pie',, 'Chart2D__test');
viz_pie;
```



## Resumo

**Visualizer.TwoD.Summary**(( *id* , [*dataSource*], [*outputName*], [*filteredBy*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Mapeia rótulos humanos <--> Ids de campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O **método de visualização** Resumo cria um gráfico de resumo a partir dos dados bidimensionais. O gráfico de resumo é padronizado para gráficos que mostram valores de dados de rolagem.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5},
                {'History', 17},
                {'Geography', 7},
                {'Chemistry', 16},
                {'Irish', 26},
                {'Spanish', 67},
                {'Biologiy', 66},
                {'Physics', 46},
                {'Math', 98}],
              {STRING subject, INTEGER4 year});
data_example := OUTPUT(ds, NAMED('Chart2D__test'));
data_example;
viz_Summary := Visualizer.TwoD.Summary('Summary',, 'Chart2D__test');
viz_Summary;
```

# Nuvem de Palavras

**Visualizer.TwoD.WordCloud**(( *id* , [*dataSource*], [*outputName*], [*filteredBy*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Mapeia rótulos humanos <--> Ids de campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O **método de visualização** Nuvem de Palavras cria uma nuvem de palavras a partir de dados bidimensionais. A Nuvem de Palavras constitui uma lista de palavras visualmente ponderadas. Trata-se de uma representação visual de dados de texto normalmente usada para retratar o peso ou a importância das palavras através do tamanho da fonte ou cor.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ { 'English', 5},
                { 'History', 17},
                { 'Geography', 7},
                { 'Chemistry', 16},
                { 'Irish', 26},
                { 'Spanish', 67},
                { 'Biology', 66},
                { 'Physics', 46},
                { 'Math', 98}],
              {STRING subject, INTEGER4 year});
data_example := OUTPUT(ds, NAMED('Chart2D__test'));
data_example;
viz_WordCloud := Visualizer.TwoD.WordCloud('WordCloud', , 'Chart2D__test');
viz_WordCloud;
```

## ***Métodos Multidimensionais***

Esta seção abrange os métodos de criação de visualizações multidimensionais contidas no módulo MultiD. Esses métodos oferecem formas de representação dos dados em um ambiente multidimensional.

## Área

**Visualizer.MultiD.Area**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da Coluna de mapeamento <--> ID do campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Área** exibe dados quantitativos na forma gráfica. A área entre o eixo e cada linha é mostrada e pode ser destacada através de sombreado ou uso de cores. Isso é normalmente usado para comparar duas ou mais quantidades.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5, 43, 41, 92},
                {'History', 17, 43, 83, 93},
                {'Geography', 7, 45, 52, 83},
                {'Chemistry', 16, 73, 52, 83},
                {'Spanish', 26, 83, 11, 72},
                {'Biology', 66, 60, 85, 6},
                {'Physics', 46, 20, 53, 7},
                {'Math', 98, 30, 23, 13}],
              {STRING subject, INTEGER4 year1,
               INTEGER4 year2, INTEGER4 year3, INTEGER4 year4});
data_exams := OUTPUT(ds, NAMED('MultiD__test'));
data_exams;

viz_area := Visualizer.MultiD.Area('area',, 'MultiD__test');
Viz_area;
```

## Barras

**Visualizer.MultiD.Bar**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da Coluna de mapeamento <--> ID do campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Barras** exibe um gráfico contendo barras horizontais retangulares onde cada comprimento representa o valor dos dados.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5, 43, 41, 92},
                {'History', 17, 43, 83, 93},
                {'Geography', 7, 45, 52, 83},
                {'Chemistry', 16, 73, 52, 83},
                {'Spanish', 26, 83, 11, 72},
                {'Biology', 66, 60, 85, 6},
                {'Physics', 46, 20, 53, 7},
                {'Math', 98, 30, 23, 13}],
              {STRING subject, INTEGER4 year1,
               INTEGER4 year2, INTEGER4 year3, INTEGER4 year4});
data_exams := OUTPUT(ds, NAMED('MultiD__test'));
data_exams;

viz_bar := Visualizer.MultiD.Bar('bar',, 'MultiD__test');
viz_bar;
```

## Colunas

**Visualizer.MultiD.Column**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da Coluna de mapeamento <--> ID do campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Colunas** exibe um gráfico contendo barras verticais retangulares onde cada comprimento representa o valor dos dados.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5, 43, 41, 92},
                {'History', 17, 43, 83, 93},
                {'Geography', 7, 45, 52, 83},
                {'Chemistry', 16, 73, 52, 83},
                {'Spanish', 26, 83, 11, 72},
                {'Biology', 66, 60, 85, 6},
                {'Physics', 46, 20, 53, 7},
                {'Math', 98, 30, 23, 13}],
              {STRING subject, INTEGER4 year1,
               INTEGER4 year2, INTEGER4 year3, INTEGER4 year4});
data_exams := OUTPUT(ds, NAMED('MultiD__test'));
data_exams;

viz_Column := Visualizer.MultiD.Column('column',, 'MultiD__test');
viz_Column;
```

# Hexágono

**Visualizer.MultiD.HexBin**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da Coluna de mapeamento <--> ID do campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Hexágono** exibe um gráfico com caixas hexagonais indicando duas ou mais variáveis contínuas entre si. A categorização hexagonal é útil para agregar valores de dados em uma exibição mais ampla. Por exemplo, em vez de mostrar milhares de pontos em uma representação dispersa, é possível combinar pontos em alguns hexágonos para mostrar a distribuição.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5, 43, 41, 92},
                {'History', 17, 43, 83, 93},
                {'Geography', 7, 45, 52, 83},
                {'Chemistry', 16, 73, 52, 83},
                {'Spanish', 26, 83, 11, 72},
                {'Biology', 66, 60, 85, 6},
                {'Physics', 46, 20, 53, 7},
                {'Math', 98, 30, 23, 13}],
              {STRING subject, INTEGER4 year1,
               INTEGER4 year2, INTEGER4 year3, INTEGER4 year4});
data_exams := OUTPUT(ds, NAMED('MultiD__test'));
data_exams;

viz_hexBin := Visualizer.MultiD.HexBin('hexBin',, 'MultiD__test');
viz_hexBin;
```

## Linha

**Visualizer.MultiD.Line**(( id , [dataSource], [outputName], [mappings], [properties]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da Coluna de mapeamento <--> ID do campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Linha** exibe um gráfico de linha que usa pontos conectados por linhas para mostrar a mudança dos valores.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5, 43, 41, 92},
                {'History', 17, 43, 83, 93},
                {'Geography', 7, 45, 52, 83},
                {'Chemistry', 16, 73, 52, 83},
                {'Spanish', 26, 83, 11, 72},
                {'Biology', 66, 60, 85, 6},
                {'Physics', 46, 20, 53, 7},
                {'Math', 98, 30, 23, 13}],
              {STRING subject, INTEGER4 year1,
               INTEGER4 year2, INTEGER4 year3, INTEGER4 year4});
data_exams := OUTPUT(ds, NAMED('MultiD__test'));
data_exams;

viz_line := Visualizer.MultiD.Line('line',, 'MultiD__test');
viz_line;
```



# Dispersão

**Visualizer.MuiltD.Scatter**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da Coluna de mapeamento <--> ID do campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Dispersão** exibe um gráfico de dispersão usando os eixos horizontal e vertical para representar os pontos dos dados.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5, 43, 41, 92},
                {'History', 17, 43, 83, 93},
                {'Geography', 7, 45, 52, 83},
                {'Chemistry', 16, 73, 52, 83},
                {'Spanish', 26, 83, 11, 72},
                {'Biologiy', 66, 60, 85, 6},
                {'Physics', 46, 20, 53, 7},
                {'Math', 98, 30, 23, 13}],
              {STRING subject, INTEGER4 year1,
               INTEGER4 year2, INTEGER4 year3, INTEGER4 year4});
data_exams := OUTPUT(ds, NAMED('MultiD__test'));
data_exams;

viz_scatter := Visualizer.MultiD.Scatter('scatter',, 'MultiD__test');
viz_scatter;
```

## Escada

**Visualizer.MultiD.Step**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da Coluna de mapeamento <--> ID do campo
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O **método de visualização** Escada mostra um gráfico de escada constituído por linhas em intervalo ou “degraus” horizontais.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5, 43, 41, 92},
                {'History', 17, 43, 83, 93},
                {'Geography', 7, 45, 52, 83},
                {'Chemistry', 16, 73, 52, 83},
                {'Spanish', 26, 83, 11, 72},
                {'Biology', 66, 60, 85, 6},
                {'Physics', 46, 20, 53, 7},
                {'Math', 98, 30, 23, 13}],
              {STRING subject, INTEGER4 year1,
               INTEGER4 year2, INTEGER4 year3, INTEGER4 year4});
data_exams := OUTPUT(ds, NAMED('MultiD__test'));
data_exams;

viz_step := Visualizer.MultiD.Step('step',, 'MultiD__test');
viz_step;
```

## ***Métodos Geoespaciais***

Esta seção abrange os métodos geoespaciais que criam visualizações na forma de mapas geográficos contidas no módulo Choropleth. Esses métodos projetam dados em mapas usando o sombreamento para representar os valores.

## Estados do EUA

**Visualizer.Choropleth.USStates**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da colunas de mapeamento <-->ID do campo ID
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Estados do EUA** o método de visualização representa os dados em um mapa do EUA.

Exemplo:

```
IMPORT Visualizer;
_usStates := DATASET([ {'AL', 4779736},
                      {'AK', 710231},
                      {'AZ', 6392017},
                      {'AR', 2915918}],
                    {STRING State, INTEGER4 weight});
data_usStates := OUTPUT(_usStates, NAMED('choro_usStates'));
data_usStates;
viz_usstates := Visualizer.Choropleth.USStates('usStates',, 'choro_usStates');
viz_usstates;
```

## Condados do EUA

**Visualizer.Choropleth.USCounties**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da colunas de mapeamento <-->ID do campo ID
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Condados do EUA** representa os dados em um mapa dos condados do EUA.

Exemplo:

```
IMPORT Visualizer;

_usCounties := DATASET([      {1073,29.946185501741},
                                {1097,0.79566003616637},
                                {1117,1.5223596574691},
                                {4005,27.311773623042}],
                        {STRING FIPS, INTEGER4 weight});

data_usCounties := OUTPUT(_usCounties, NAMED('choro_usCounties'));
data_usCounties;
viz_uscounties := Visualizer.Choropleth.USCounties('usCounties', , 'choro_usCounties');
viz_uscounties;
```

## Euro

**Visualizer.Choropleth.Euro**(( *id* ,[*region*] [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>region</i>	Código europeu de 2 letras ( GB, IE, etc.)
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da colunas de mapeamento <-->ID do campo ID
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Euro** representa os dados em um mapa da Europa. Os **métodos EuroIE e EuroGB** são fornecidos como exemplos. Para criar outros métodos específicos para um determinado país, basta usar o método Euro e fornecer o código do país de dois caracteres no parâmetro *\_region*.

Exemplo:

```
IMPORT Visualizer;
_euroIE := DATASET([      {'Carlow', '27431', '27181', '54612'},
                           {'Dublin City', '257303', '270309', '527612'},
                           {'Kilkenny', '47788', '47631', '95419'},
                           {'Cork', '198658', '201144', '399802'}],
                   {STRING region, INTEGER4 males, INTEGER4 females,
                    INTEGER4 total});
data_euroIE := OUTPUT(_euroIE, NAMED('choro_euroIE'));
data_euroIE;

viz_euroIE := Visualizer.Choropleth.EuroIE('euroIE',, 'choro_euroIE',
      DATASET([{'County', 'region'}, {'Population', 'total'}], Visualizer.KeyValueDef),,
      DATASET([{'paletteID', 'Greens'}], Visualizer.KeyValueDef));
viz_euroIE;
```

# ***Métodos Gerais de Visualização***

Esta seção abrange os métodos de criação de visualizações genéricas contidas em qualquer módulo. Essas visualizações são úteis para testar e examinar sua fonte de dados.

## Grade

**Visualizer.Any.Grid**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da colunas de mapeamento <-->ID do campo ID
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização **Grade** cria uma tabela simples ou grade a partir de quaisquer dados.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5},
                {'History', 17},
                {'Geography', 7},
                {'Chemistry', 16},
                {'Irish', 26},
                {'Spanish', 67},
                {'Biology', 66},
                {'Physics', 46},
                {'Math', 98}],
              {STRING subject, INTEGER4 year});
data_example := OUTPUT(ds, NAMED('Chart2D__test'));
data_example;
viz_Grid := Visualizer.Any.Grid('Grid',, 'Chart2D__test');
viz_Grid;
```



## Grade Handson

**Visualizer.Any.HandsonGrid**(( *id* , [*dataSource*], [*outputName*], [*mappings*], [*properties*]);

<i>id</i>	Identificador único da visualização.
<i>dataSource</i>	Localização do conjunto de resultados (WUID, Arquivo lógico, resultado da consulta Roxie), padrão para a WU
<i>outputName</i>	Nome do conjunto de resultados (ignorado para arquivos lógicos)
<i>mappings</i>	Nome da colunas de mapeamento <-->ID do campo ID
<i>filteredBy</i>	Condição de filtro (também usado para interações de widgets)
<i>properties</i>	Propriedades de Visualização Consulte Propriedades de Visualização
Return:	Um resultado “meta” que descreve a visualização.

O método de visualização Grade Handsosn cria uma tabela ou grade a partir de quaisquer dados.

Exemplo:

```
IMPORT Visualizer;
ds := DATASET([ {'English', 5},
                {'History', 17},
                {'Geography', 7},
                {'Chemistry', 16},
                {'Irish', 26},
                {'Spanish', 67},
                {'Biologiy', 66},
                {'Physics', 46},
                {'Math', 98}],
              {STRING subject, INTEGER4 year});
data_example := OUTPUT(ds, NAMED('Chart2D__test'));
data_example;
viz_HandsonGrid := Visualizer.Any.HandsonGrid('HandsonGrid',, 'Chart2D__test');
viz_HandsonGrid;
```