



## WHAT'S NEW AND RELEASE NOTES

Release: 2020.10

IN THIS RELEASE

**NEW FEATURES AND IMPROVEMENTS..... 2**

DATA PREPARATION..... 2

    Preview input data for vector & scalar calculations ..... 2

    Preview results for scalar calculations ..... 4

    Search for schemas when connecting to Oracle ..... 5

    Search for cubes & dimensions when connecting to TM1 Cube ..... 6

STORYBOARDS..... 7

    Change column’s number format..... 7

    Change column’s date & time format ..... 9

    Preview input data for vector & scalar calculations ..... 11

    Preview results for vector & scalar calculations..... 13

    View & copy a widget ID..... 14

INSTALLATION & CONFIGURATION ..... 15


    Check system readiness before platform installation ..... 15

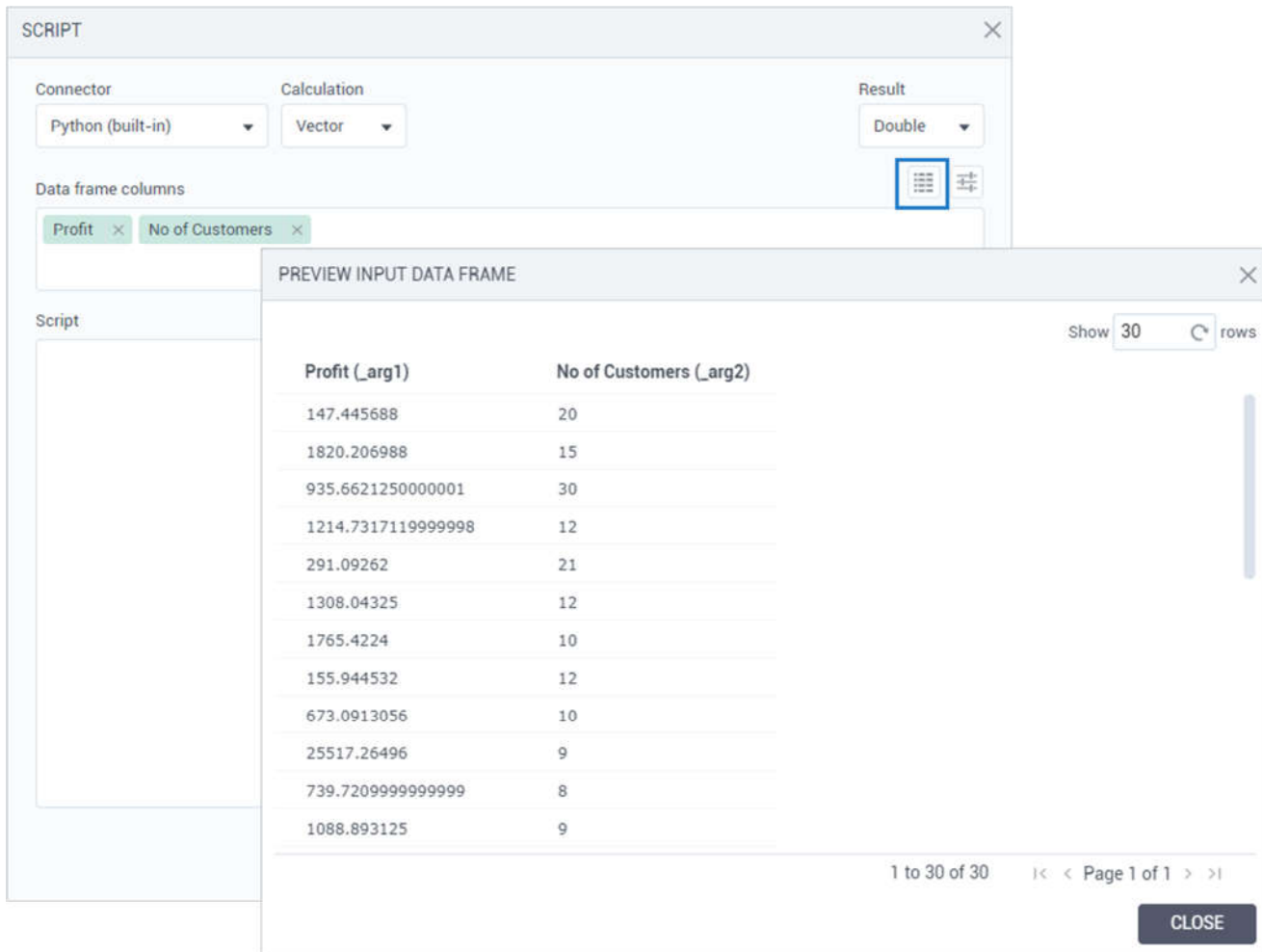
## NEW FEATURES AND IMPROVEMENTS

### DATA PREPARATION

#### *Preview input data for vector & scalar calculations*

In this release, when adding vector or scalar calculations, data science users can preview the input data frame for the selected arguments and auto-generate the data frame stub directly in the **Script** editor.

After selecting columns as the script arguments, you can preview the raw data values by clicking  **Preview input data frame**.




The screenshot displays the DataClarity interface. The main window is titled "SCRIPT" and contains the following elements:

- Connector:** Python (built-in)
- Calculation:** Vector
- Result:** Double
- Data frame columns:** Profit, No of Customers
- Script editor:** A large empty text area for writing code.

A "PREVIEW INPUT DATA FRAME" window is overlaid on the script editor, showing a table of data. The table has two columns: "Profit (\_arg1)" and "No of Customers (\_arg2)". The data is as follows:

Profit (_arg1)	No of Customers (_arg2)
147.445688	20
1820.206988	15
935.6621250000001	30
1214.7317119999998	12
291.09262	21
1308.04325	12
1765.4224	10
155.944532	12
673.0913056	10
25517.26496	9
739.7209999999999	8
1088.893125	9

The preview window also includes a "Show 30 rows" control and a "CLOSE" button at the bottom right.

You can also benefit from the auto-generated stub template. Click  **Generate data frame stub**, and then click **Copy** and reuse it in a script.

```
DATA FRAME STUB ×

import pandas as pd

# initialize list for input column names
columns = ['Profit','No of Customers']

# initialize list for input data
data = [_arg1, _arg2]

# transform columns and data into dataframe
df = pd.DataFrame(dict(zip(columns, data)))

# select a single column (i.e., series) to return
series = df.iloc[:, 0]

# convert this single column to a Python list
series = series.tolist()

# return results
return series
```

CLOSE COPY

### Preview results for scalar calculations

Now, for scalar calculations, you can preview and evaluate the calculation results directly in the **Script** editor. This enhancement allows data scientists to verify the calculation before saving it to a dataset.

If you use measures, ensure the columns' aggregations are set to **None**, and then click **Preview result**. The results appear in the **Calculation** column.

The screenshot shows the DataClarity interface. The main window is titled "SCRIPT" and contains the following elements:

- Connector:** Python (built-in)
- Calculation:** Scalar
- Result:** String
- Data frame columns:** Product Type
- Script:** `return _arg1.upper()`
- PREVIEW RESULT** button (highlighted with a blue box)

The "PREVIEW RESULTS" window is open, displaying a table with the following data:

Product Type	Calculation
Televisions	TELEVISIONS
Media Players	MEDIA PLAYERS
Portable DVD	PORTABLE DVD
Home Theater	HOME THEATER
All-In-One	ALL-IN-ONE
Computer Parts	COMPUTER PARTS
Networking	NETWORKING
Printers	PRINTERS
Laptops	LAPTOPS
Software	SOFTWARE
Phones	PHONES
Android Phones	ANDROID PHONES

The "PREVIEW RESULTS" window also includes a "Show 30 rows" control, a "1 to 30 of 30" indicator, and a "Page 1 of 1" indicator. A "CLOSE" button is located at the bottom right of the window.

## Search for schemas when connecting to Oracle

Now, when creating a data connection to an Oracle data source, you can search for schemas directly in the **Schema** field after clicking **Load list**. When searching, you will see the schema names that contain your input, sorted alphabetically.

**ADD DATA CONNECTION** [X]

Connection details    Caching

ORACLE

Oracle data source connection

Test your connection  
After you fill in the data connection details, test your connection. If something goes wrong, an error message will be displayed.

TEST CONNECTION

Port: 1521

Service ID: orcl12

Username: qasample

Password: .....

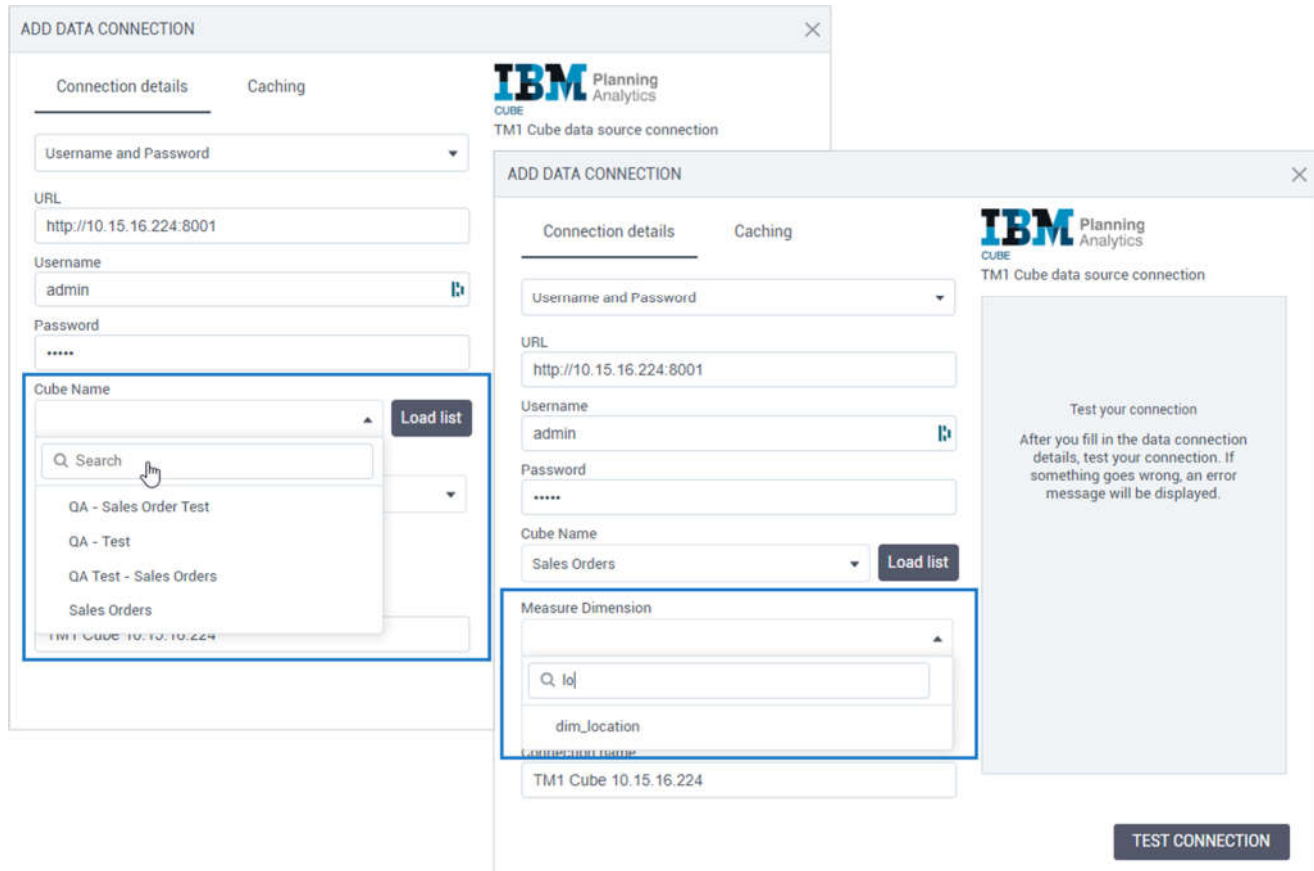
Schema: [dropdown] **Load list**

Q sale

- GOSALES
- GOSALESDW
- SALES
- SALESD

## Search for cubes & dimensions when connecting to TM1 Cube

Now, when creating a data connection to a TM1 cube, users can search for cubes and measure dimensions. You can start typing a cube name directly in the **Cube Name** field. After selecting a cube, you can also search for a dimension with measures in the **Measure Dimension** field. When searching, you will see the items containing your input, sorted alphabetically.




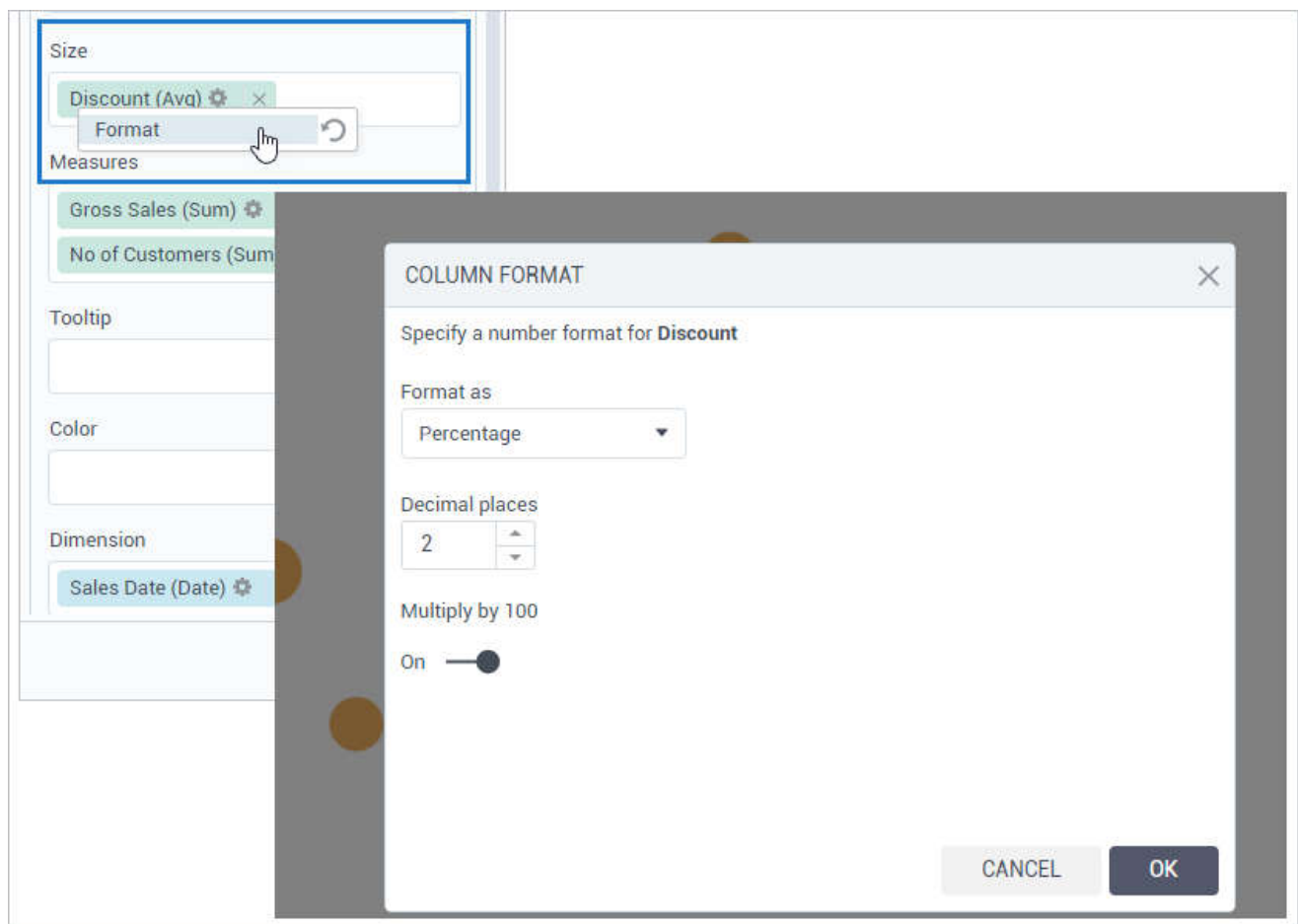
## STORYBOARDS

### *Change column's number format*

Starting from this release, storyboards creators can control how to display values in numeric columns, overriding formats set for a dataset. In this case, the local format applies to the visualization where it was added. However, you can reuse the formats if you duplicate or copy a widget.

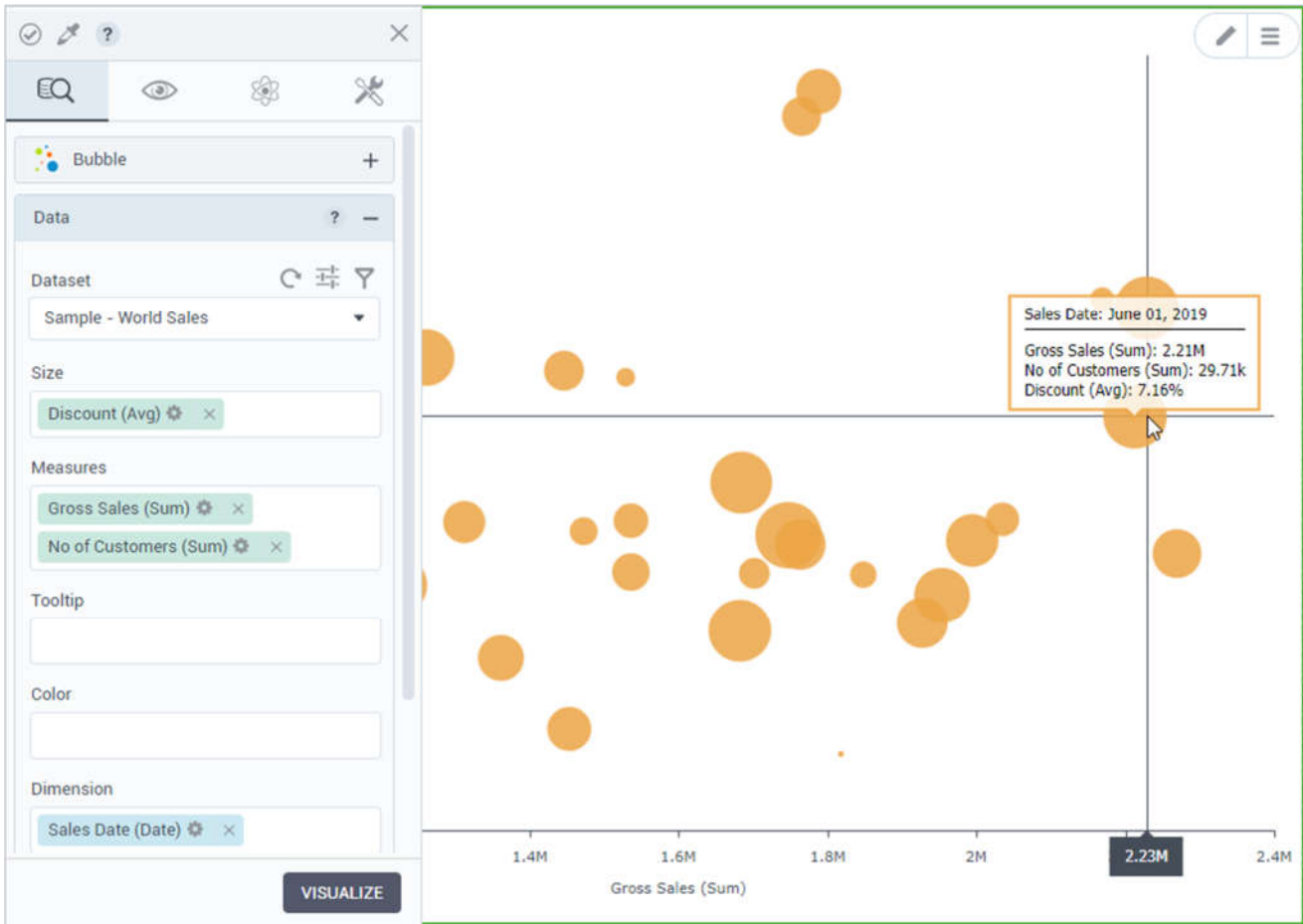
You can format numeric columns as number, currency, or percentage. Each format has its own set of properties. For example, you can define how many decimals to show, what separator symbols to use for decimals and thousands, select the appearance of negative numbers, or whether to use abbreviations for large numbers such as K, M, G, B, or T.

You can define a number format on the **data** tab of the widgets settings pane. In a data field, point to a numeric column, click  **Options**, and then select **Format**.

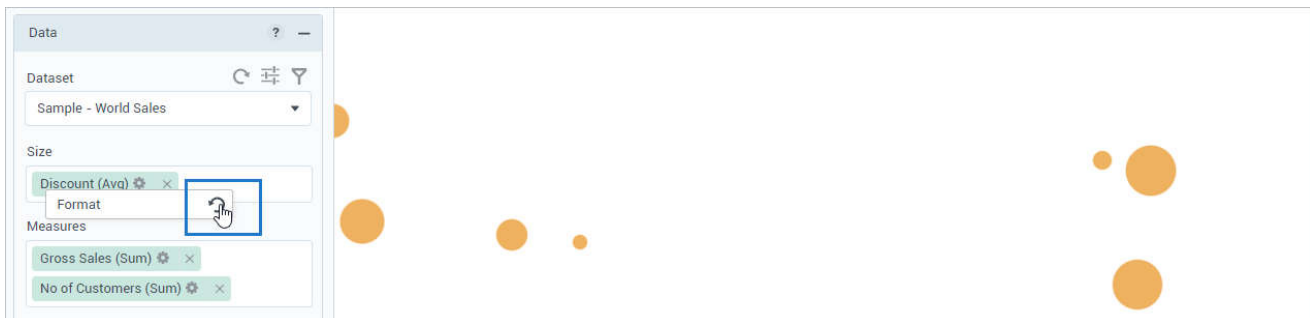




In the **Column format** dialog, select which format type to apply: **Number**, **Currency**, or **Percentage**. Then, you can customize the individual format properties as needed. Click **OK** to apply the new format.



The format defined for a column overrides the dataset format and the default formats defined on the **appearance** tab. However, you can quickly return to the original dataset format by clicking **Reset** next to the **Format** menu option.




### Change column's date & time format

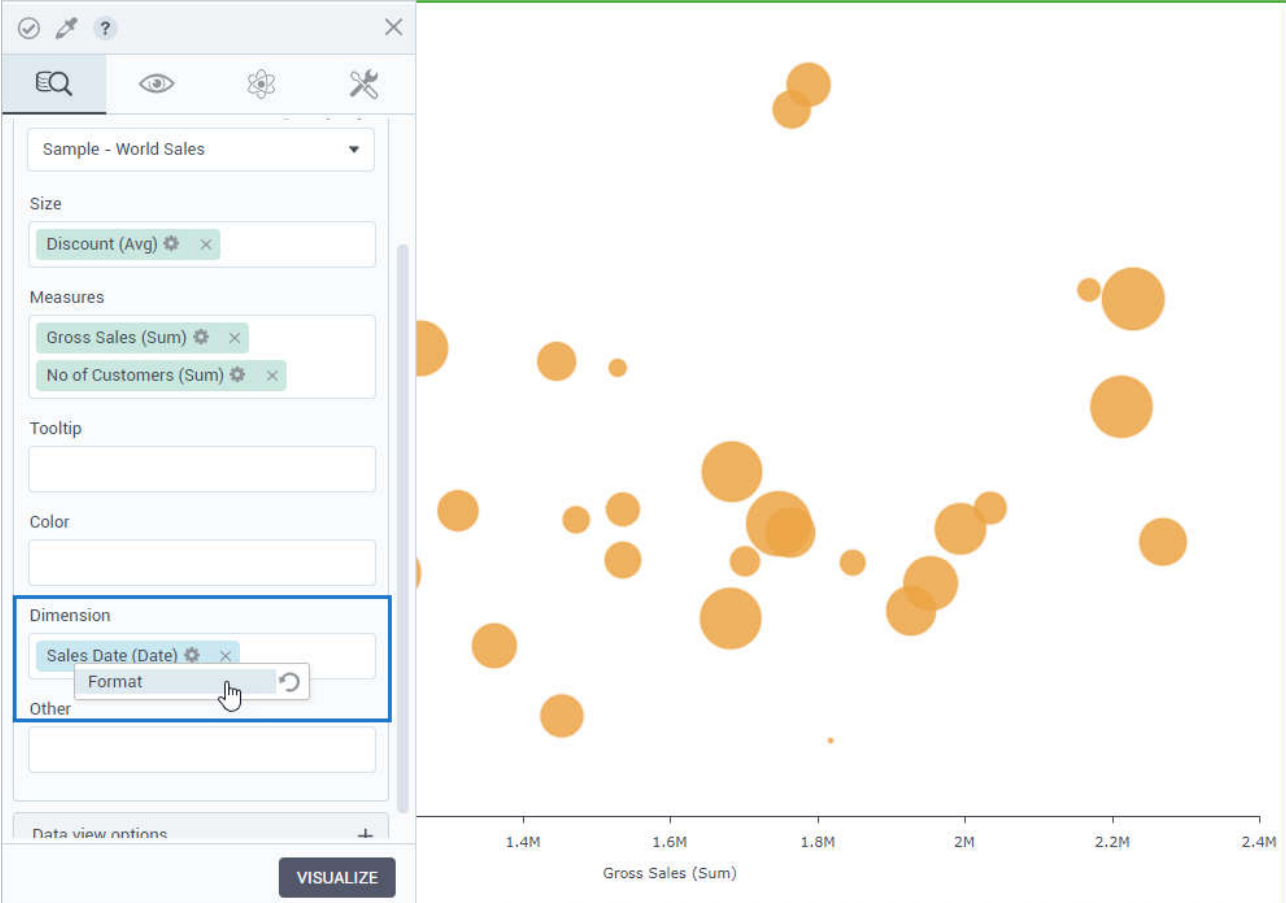
This release also allows storyboards creators to control the appearance of the columns containing dates, date-time, or time, overriding formats set for a dataset. In this case, the local format applies to the visualization where it was added. However, you can reuse the formats if you duplicate or copy a widget.

For a date column, you can select a format from a predefined list. For example, you can choose to display a long date, a short date, include a weekday, or spell out month names:

- 03-Dec-20
- 12/03/20
- 03/12/2020
- 2020-12-03
- December 03, 2020
- 03-Dec-20 1:30:00 PM
- Thu, December 3, 2020

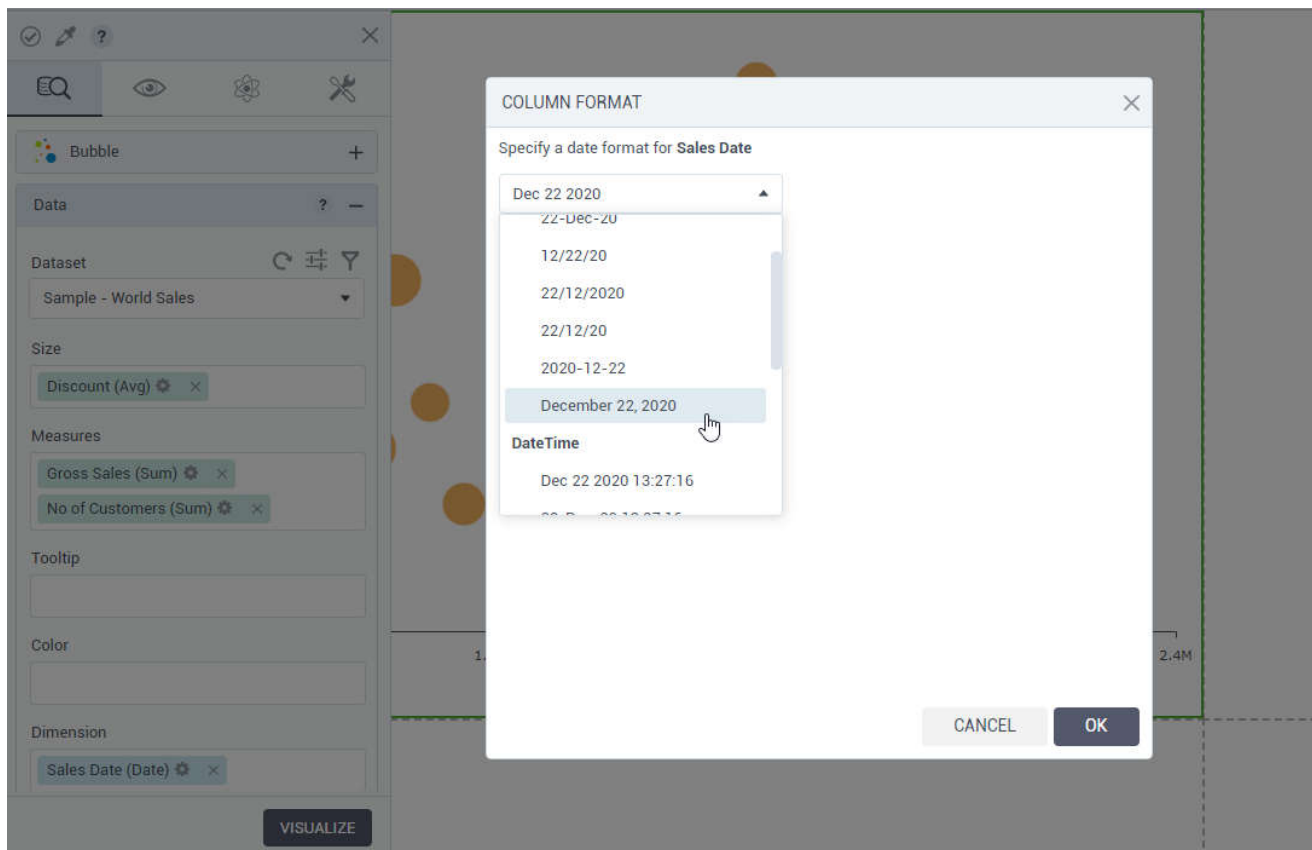
For a time column, you can also select one of the predefined time formats.

You can define a column format on the **data** tab of the widgets settings pane. In a data field, point to a column, click  **Options**, and then select **Format**. Note that the column should have the aggregation set to **Default**.

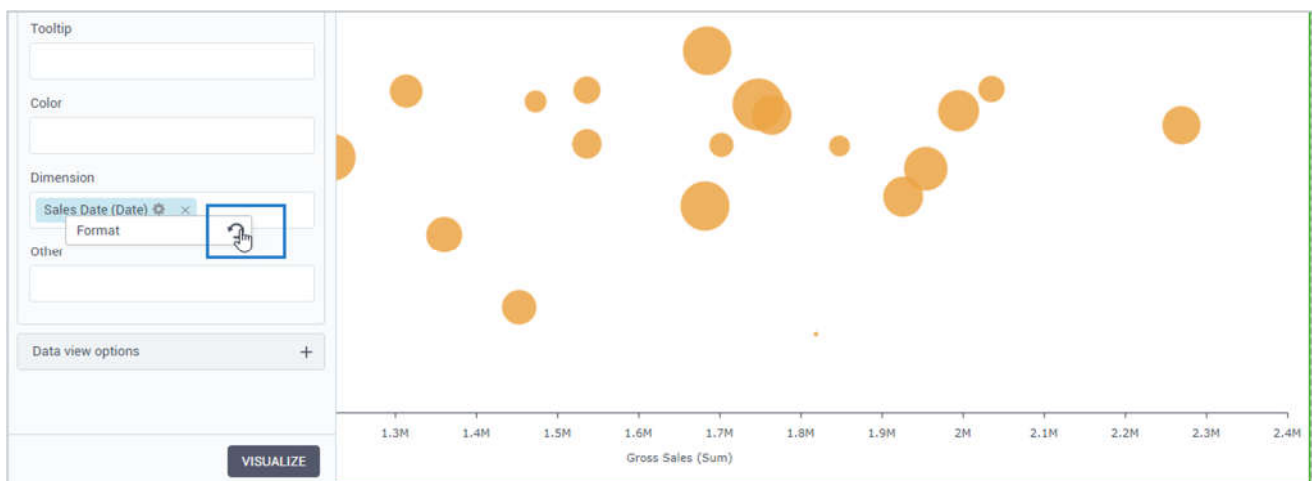


The screenshot shows the DataClarity interface. On the left is a settings pane for a widget titled 'Sample - World Sales'. The 'Dimension' field is selected, and a dropdown menu is open, showing the 'Format' option. The main area displays a bubble chart with orange bubbles of varying sizes. The x-axis is labeled 'Gross Sales (Sum)' and ranges from 1.4M to 2.4M. The y-axis is not explicitly labeled but represents the 'Sales Date (Date)' dimension. A 'VISUALIZE' button is at the bottom of the settings pane.

In the **Column format** dialog, select a format and click **OK** to apply the new format.




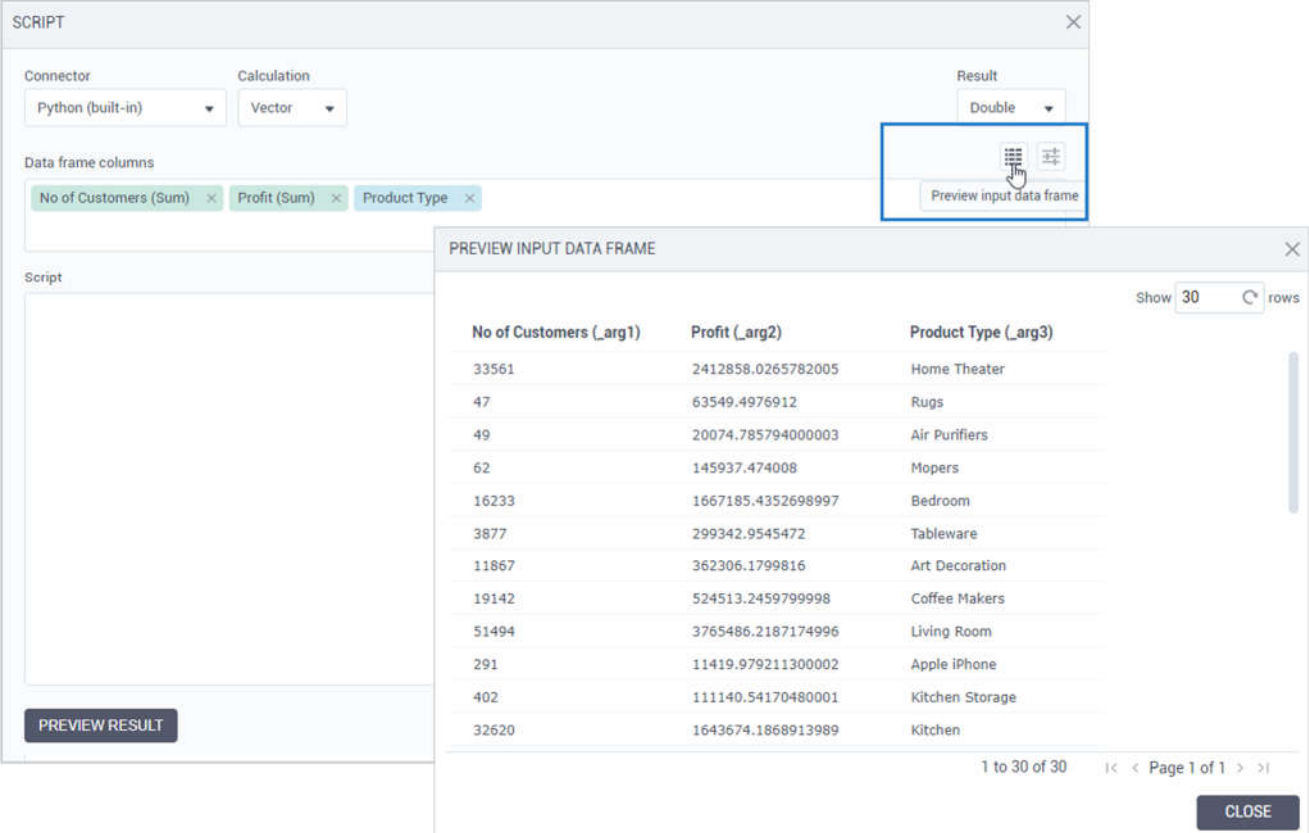
The format defined for a column overrides the dataset format and the default formats defined on the **appearance** tab. However, you can quickly return to the original dataset format by clicking **Reset** next to the **Format** menu option.



### Preview input data for vector & scalar calculations

In this release, when adding vector or scalar calculations, data science users can preview the input data frame for the selected arguments and auto-generate the data frame stub directly in the **Script** editor.

After selecting columns as the script arguments, you can preview the column values by clicking  **Preview input data frame**.




The screenshot shows the SCRIPT editor interface. The 'Connector' is set to 'Python (built-in)', the 'Calculation' is 'Vector', and the 'Result' is 'Double'. Under 'Data frame columns', three columns are selected: 'No of Customers (Sum)', 'Profit (Sum)', and 'Product Type'. A button labeled 'Preview input data frame' is highlighted with a blue box. Below this, a 'PREVIEW INPUT DATA FRAME' dialog is open, displaying a table with 30 rows of data. The table has three columns: 'No of Customers (\_arg1)', 'Profit (\_arg2)', and 'Product Type (\_arg3)'. The data is as follows:

No of Customers (_arg1)	Profit (_arg2)	Product Type (_arg3)
33561	2412858.0265782005	Home Theater
47	63549.4976912	Rugs
49	20074.785794000003	Air Purifiers
62	145937.474008	Mopers
16233	1667185.4352698997	Bedroom
3877	299342.9545472	Tableware
11867	362306.1799816	Art Decoration
19142	524513.2459799998	Coffee Makers
51494	3765486.2187174996	Living Room
291	11419.979211300002	Apple iPhone
402	111140.54170480001	Kitchen Storage
32620	1643674.1868913989	Kitchen

The dialog also shows 'Show 30 rows' and '1 to 30 of 30' at the bottom, along with a 'CLOSE' button.

The aggregation is considered for preview only if you add a dimension for grouping the selected measures. If you have only measures, the data is non-aggregated in preview. For scalar calculations, set the aggregation to **None**.

You can also benefit from the auto-generated stub template. Click  **Generate data frame stub**, and then click **Copy** to reuse it in a script.

```
DATA FRAME STUB ×

import pandas as pd

# initialize list for input column names
columns = ['No of Customers','Profit','Product Type']

# initialize list for input data
data = [_arg1, _arg2, _arg3]

# transform columns and data into dataframe
df = pd.DataFrame(dict(zip(columns, data)))

# select a single column (i.e., series) to return
series = df.iloc[:, 0]

# convert this single column to a Python list
series = series.tolist()

# return results
return series
```

CLOSE COPY

*Preview results for vector & scalar calculations*

In this release, data science users can preview and evaluate the results of a vector or scalar calculation directly in the **Script** editor before saving it for a widget.

After adding the code in the **Script** field, click **Preview result**, and find the results in the **Calculation** column.

The screenshot displays the 'SCRIPT' editor window with the following configuration:

- Connector: Python (built-in)
- Calculation: Vector
- Result: Double
- Data frame columns: No of Customers (Sum)

The script editor contains the following Python code:

```
import numpy as np
from sklearn.cluster import DBSCAN
from sklearn.preprocessing import StandardScaler
X = np.column_stack([_arg1, _arg2])
X = StandardScaler().fit_transform(X)
db = DBSCAN(eps=1, min_samples=5)
result = db.labels_.tolist()
return result
```

The 'PREVIEW RESULTS' window shows a table with the following data:

No of Customers	Profit	Product Type	Calculation
47	63550.1296112	Rugs	0
49	20077.395338000006	Air Purifiers	0
62	145972.417084	Mopers	0
14691	1550256.52025	Bedroom	0
3746	297349.9923732	Tableware	0
10884	344237.445962	Art Decoration	0
18442	507926.85180399999	Coffee Makers	0
49149	3645176.212574002	Living Room	-1
402	111118.58368960001	Kitchen Storage	0
27547	1430092.6275771996	Kitchen	0
5570	609449.6152785	Steam Cleaners	0
8715	672784.3042983996	Refrigerators	0

The 'PREVIEW RESULT' button in the script editor and the 'Calculation' column in the table are highlighted with blue boxes. The table also includes a 'Show 30 rows' control and a 'CLOSE' button at the bottom right.

### View & copy a widget ID

In this release, power users can view and copy the widget's ID to be used in their custom code in the **HTML code widget** or the **Custom JavaScript code** section of a storyboard's page properties pane.

You can find the code in the widget's settings pane, on the **advanced** tab, in the **Info** section.

The screenshot shows a bar chart with blue bars representing data points over time. The x-axis is labeled with months and years: Nov 2017, Jan 2018, Mar 2018, May 2018, Jul 2018, Sep 2018, Nov 2018, Jan 2019, Mar 2019, May 2019, Jul 2019. The y-axis represents the magnitude of the data points. A settings pane is overlaid on the left side of the chart, with the 'Info' section expanded. The 'Widget ID' is displayed as '16076998606576692\_160769912E' and a 'COPY' button is visible next to it.

## INSTALLATION & CONFIGURATION

### *Check system readiness before platform installation*

In this release, administrators no longer need to check basic system requirements manually before platform installation. The new standalone script verifies the minimum Proof of Concept requirements and allows you to assess the following aspects of environment readiness:

- The minimum required CPUs (4)
- The minimum required RAM (16 GB)
- The required free disk space for the `/var/snap/microk8s` partition (30 GB)
- The needed ports are open (443, 80, data server port)
- The required external repositories are accessible