

# SIEMENS

Insights Hub

Dashboard Designer v10

System Manual


04/2024


Introduction	1
Identity and Access Management	2
Navigating	3
Getting Started	4
Creating Dashboards	5
Functions	6
Visualizations	7


## Legal information

### Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

 <b>DANGER</b>
indicates that death or severe personal injury <b>will</b> result if proper precautions are not taken.

 <b>WARNING</b>
indicates that death or severe personal injury <b>may</b> result if proper precautions are not taken.

 <b>CAUTION</b>
indicates that minor personal injury can result if proper precautions are not taken.

<b>NOTICE</b>
indicates that property damage can result if proper precautions are not taken.


If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

### Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

### Proper use of Siemens products

Note the following:

 <b>WARNING</b>
Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

### Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

### Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

# Table of contents

<b>1. Introduction</b> .....	4
1.1. Introduction.....	4
<b>2. Identity and Access Management</b> .....	7
2.1. Identity and Access Management.....	7
<b>3. Navigating</b> .....	9
3.1. Navigating Dashboard Designer.....	9
<b>4. Getting Started</b> .....	14
4.1. Getting Started.....	14
<b>5. Creating Dashboards</b> .....	17
5.1. Creating Dashboards.....	17
<b>6. Functions</b> .....	33
6.1. Functions.....	33
<b>7. Visualizations</b> .....	61
7.1. Visualizations.....	61

# Introduction

---

hide\_license\_text: true

## 1.1 Introduction

For this document in PDF format, click here: [Download](#).

### Overview

Dashboard Designer is a robust application designed to empower users with the ability to transform raw data into comprehensive, visually engaging representations. With a host of data exploration tools and an extensive library of visualizations, Dashboard Designer simplifies the complex process of creating insightful dashboards.

### Data Visualization and Query Tools

The platform offers a diverse range of tools aimed at translating data into meaningful visualizations. Users can select their preferred data sources and harness powerful query functions to precisely target and transform the data. This customization allows for tailored and accurate data representation.

### Dashboard Linking and Sharing

Apart from facilitating dashboard creation, Dashboard Designer allows users to import and share panels among various dashboards. Moreover, it offers the flexibility to link dashboards to both internal and external Internet locations. This interlinking of dashboards enables seamless navigation and accessibility across multiple interfaces.

### Accessibility and Ease of Use

An essential aspect of Dashboard Designer is its user-friendly interface, enabling individuals to create informative dashboards without the need for programming knowledge. This accessibility is

coupled with a comprehensive library of visualizations, ensuring that every user, regardless of their technical background, can create stunning and meaningful dashboards.

## Variety of Visualizations

The application offers a wide array of visualizations, ranging from fundamental visualizations like graphs and pie charts, advanced visualizations like heatmaps and singlestat, and third-party visualizations like SVG and traffic light that extend Dashboard Designer's visual capabilities.

Dashboard Designer includes:

- Time series
- Bar chart
- Stat
- Gauge
- Bar guage
- Table
- Pie chart
- State timeline
- Heatmap
- Status history
- Histogram
- Text
- Dashboard list
- Candlesticks
- Canvas
- Flame graph
- HTML graphics
- Geomap
- Clock
- Logs
- SVG
- Cal-Heatmap
- Plotly

- Node Graphs
- Traffic Lights
- Breadcrumb \_ Traces
- Trend
- Apache EChart
- XY Chart Beta

## Enhancements and Future Releases

While the application currently offers a set list of visualizations, future releases are expected to introduce additional visualizations to extend the capabilities of Dashboard Designer.

## Third-Party Visualizations

Users cannot add third-party visualizations or plugins to Dashboard Designer. More visualizations are coming in future releases. See the *Visualizations* topic in this documentation for more information on capabilities.

# Identity and Access Management

# 2

## 2.1 Identity and Access Management

### For Organizations

Your subtenants are represented as "Organizations" (Orgs) in Dashboard Designer. Orgs are automatically created when a subtenant user first logs in.

### User Roles

Roles are set by Admins to determine what users can view and the actions they can perform. This briefly describes the user roles in Dashboard Designer:

Permission	Admin	Editor	Viewer
View dashboards	Yes	Yes	Yes
Add, edit, delete dashboards	Yes	Yes	No
Add, edit, delete folders	Yes	Yes	No
View playlists	Yes	Yes	Yes
Add, edit, delete playlists	Yes	Yes	No
Create library panels	Yes	Yes	No
View annotations	Yes	Yes	Yes
Add, edit, delete annotations	Yes	Yes	No
Access explore	Yes	Yes	No
Query data sources directly	Yes	Yes	Yes

### Usage Quotas

Dashboard Designer subscriptions include a number of named licenses and a limited number of dashboards. Please see [Technical Limitations](#).

## Teams

Teams are groups of users.



# Navigating

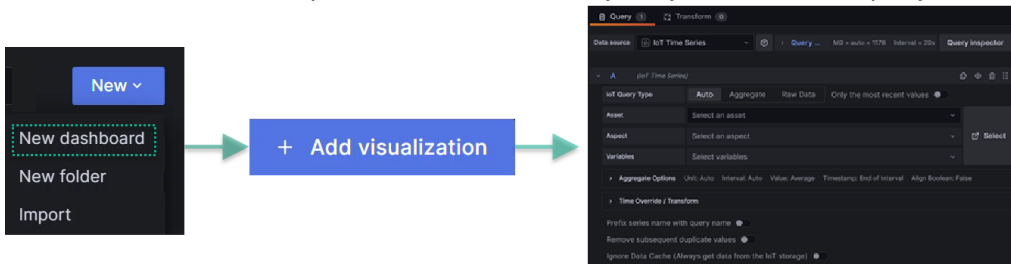
# 3

## 3.1 Navigating Dashboard Designer

A number of Dashboard Designer features are accessible directly from the top two rows of the user interface (UI). The major features -- queries and visualization are described below, as well as the icons that lead to other features.

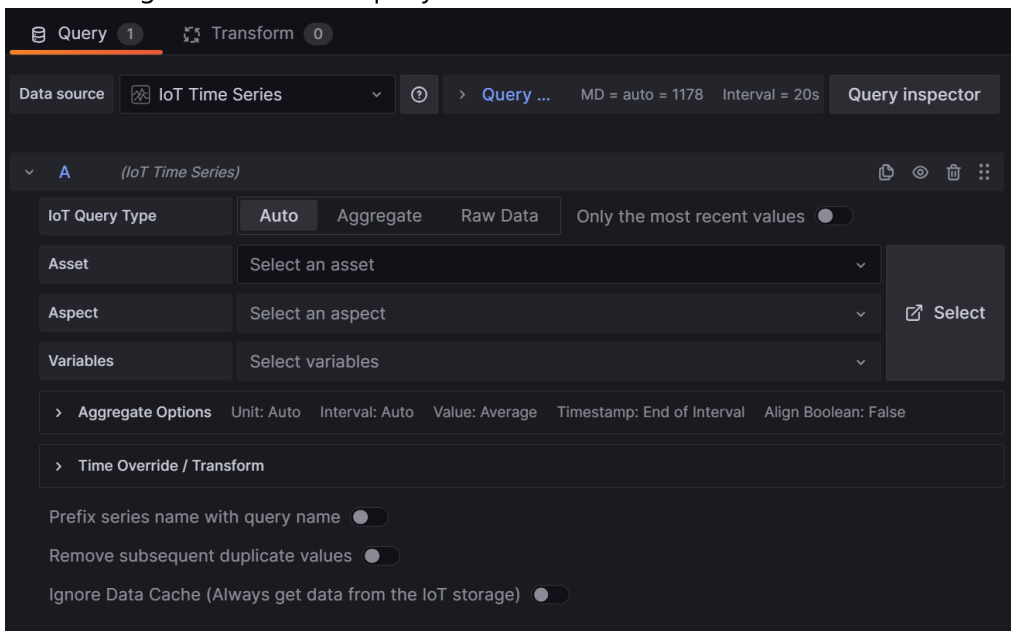
### Adding a Data source

You have to add a data source before you can build a query or create a visualization for a dashboard. This image shows the two clicks that take you to the query editor with your selected data source (in this example, IoT time series) ready for you to use in a query.



### Query Panel

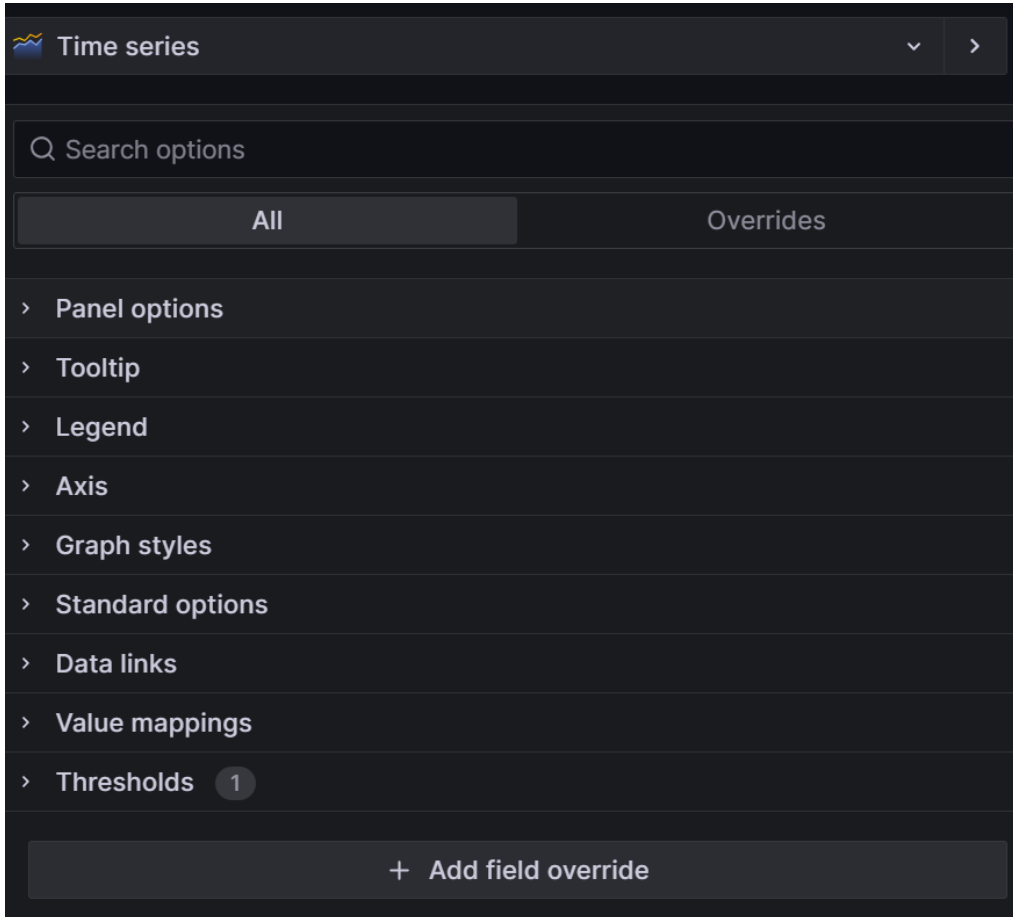
Here's a larger version of the query editor:



For more information on using the query editor, see the *Creating Dashboards* topic. When you finish configuring your query, you can customize your visualization.

## Visualization Panel

Here's an example of the visualization panel fields (shown collapsed):



For more information on using the visualization panel, see the *Creating Dashboards* topic.


## Icons and Features

When you open Dashboard Designer, icons along the top of the user interface (UI) connect to various features. This table describes the icons and features in the order they appear on the landing page, left to right.

Location	Feature or Icon	Description
Landing page	What's new & next	Displays news and information about Dashboard Designer development
Home menu	Starred	Signifies favorite items that you can click to navigate to

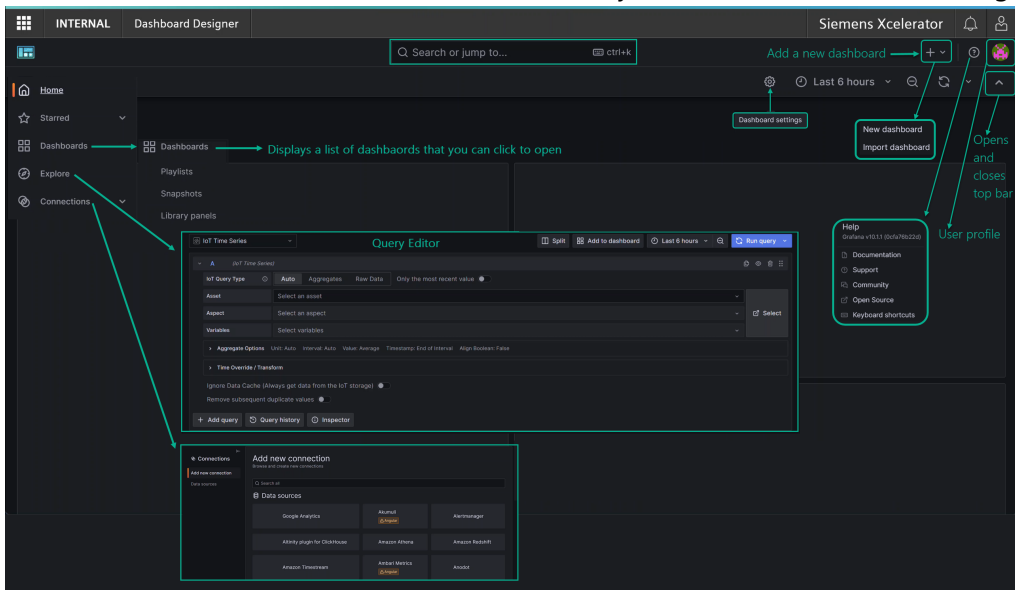
Location	Feature or Icon	Description
Home menu	Dashboards	Click to open the <b>folders and files</b> where your dashboards are stored
Primary row	Explore	Explore data and try out various queries and graph settings with the option of creating a dashboard from your exploration
Primary row	Settings	Displays settings for the open dashboard
Primary row	General	Displays general information for the open dashboard
Primary row	Annotations	Displays annotations for the open dashboard
Primary row	Variables	Displays variables for the open dashboard
Primary row	Asterisk icon	Click to view keyboard shortcuts
Primary row	Links	Displays links for the open dashboard
Primary row	Versions	Displays version information for the open dashboard
Primary row	Permissions	Displays permission settings for the open dashboard
Primary row	Refresh icon	Click to update the data or to select the rate at which to refresh dashboard data
Primary row	Collapse icon	Closes and opens the secondary menu row
Primary row	Time range	Displays the time range of the open dashboard
Secondary row	Landing page icon	Click any time, anywhere to return to the landing page
Secondary row	Search	Enter search terms and click to jump to a feature
Secondary row	Jump	Click to jump to recently accessed actions, dashboards, pages, preferences

Location	Feature or Icon	Description
Secondary row	Asterisk icon	Click to create a new dashboard or import one
Secondary row	Question mark icon	Links to documentation, support, community, open source resources, and keyboard shortcuts
Secondary row	Tenant icon	opens your user profile and notification history

 Dashboard Designer does not support the Connections feature.

## Landing Page

Here is an image of the Dashboard Designer landing page, with pointers for locating various features. The section below describes the features you can access from the landing page.



## Exploring Data without Creating a Dashboard

From the Home menu, select 'Explore'. A query panel opens in which you can select your data source, apply configuration options, and run the query. As you adjust panel settings, the graph reflects your changes. You can also switch between lines, bars, points, etc. on the graph. When you like what you see, click the 'Add to dashboard' button to turn your exploration into a dashboard panel.



To explore data, you must first add the data source. See the *Creating Dashboards* topic for details.

## User profiles

Your user profile opens when you click the icon indicated in the landing page image, although your icon will differ from the example.



**Keyboard Shortcuts:** Dashboard Designer has many built-in keyboard shortcuts for general purposes, time range, and panels. You can view them by clicking the ? icon on the primary menu bar.

# Getting Started

# 4

## 4.1 Getting Started

### Prerequisites

Before starting, please make sure you have:

- Editor/admin access to Dashboard Designer
- Some asset data in your current set-up

### Dashboards

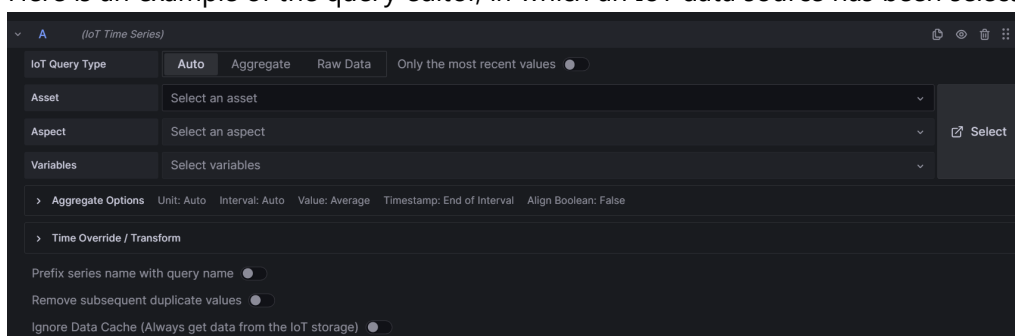
Dashboards give you an at-a-glance view of your data and let you track metrics through various visualizations.

### Panels

Panels are the main component of dashboards and are laid out like blocks on a grid. Each panel consists of a query and a visualization. The query defines the data and the visualization defines how the data is presented.

### Queries

A query retrieves specific data from the data source. Use the query editor to select the specific data you want to use in your panel and add additional options like aggregation, time overrides, or transformations according to your objectives. See the *Creating Dashboards* topic for details. Here is an example of the query editor, in which an IoT data source has been selected:

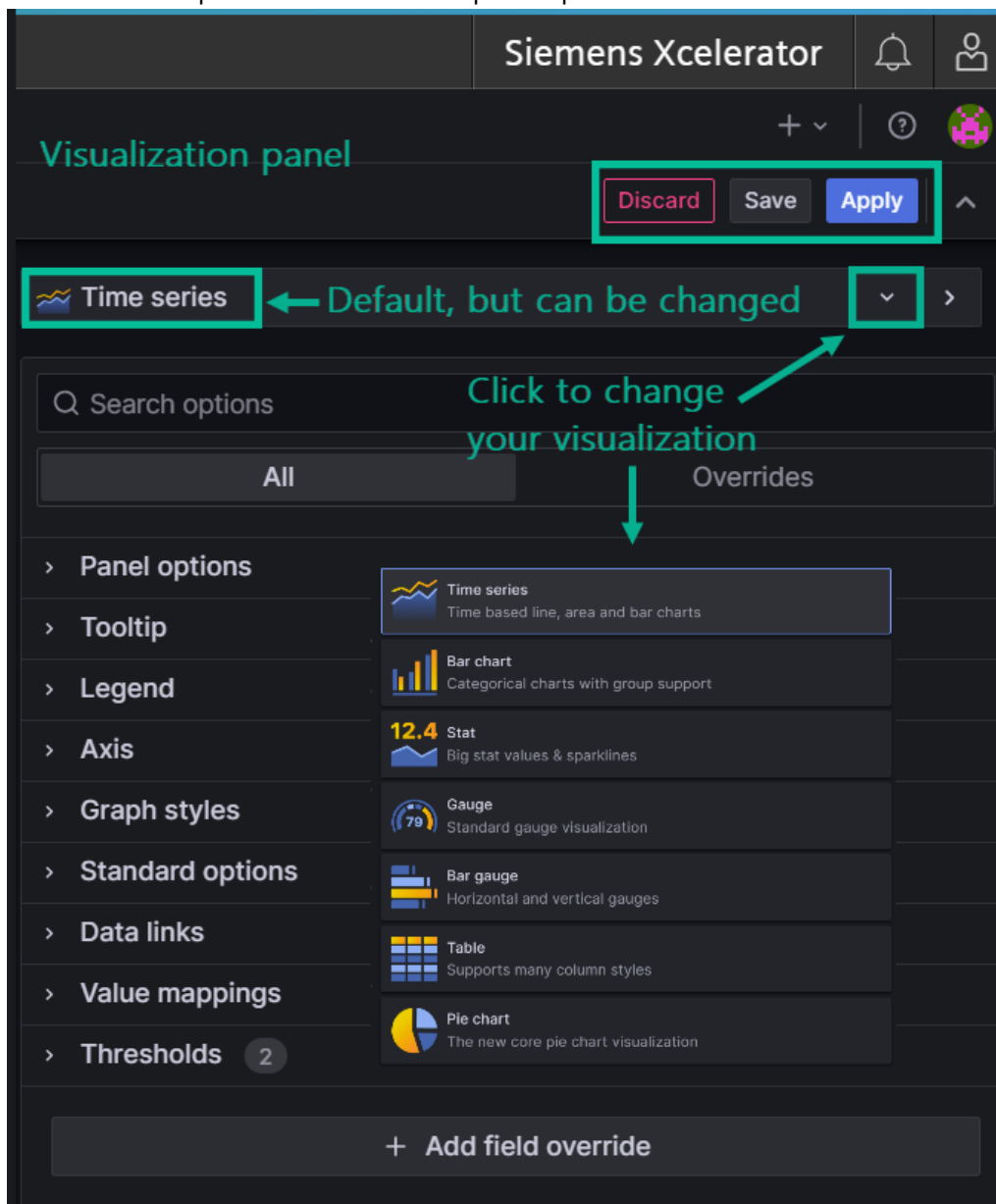


### Visualizations

The Visualization Panel provides tools for configuring and customizing the many visualization options available. It offers multiple sections and settings for tailoring the appearance and behavior of visual elements to suit your analytical requirements. Visualization panel options include specifying virtually all elements related to a dashboard, from tooltips to value mapping and field overrides. With Dashboard Designer's responsive user interface, you can instantly see how different visualization selections change the data representation. Additional options include linking to other Dashboard Designer data, setting thresholds, and mapping values with text and colors. See the *Creating Dashboards* topic for details.

Our diverse visualization options enhance data analysis capabilities, and enable users to more effectively present, interpret, and draw insights from their data.

Here is an example of the visualization panel options:



## Best Way to Get Started

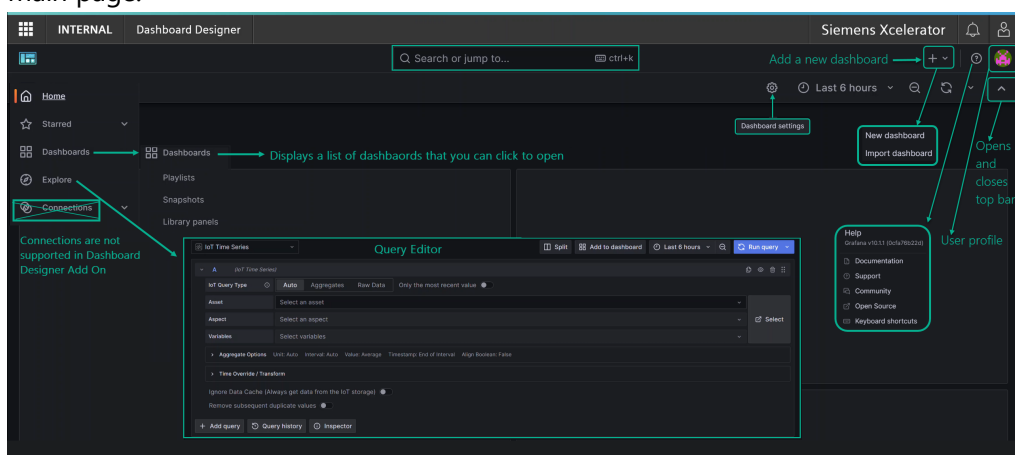
Open two pages of Dashboard Designer side-by-side and experiment with the query editor and visualizations. Any selections you make instantly reflect in the visualization, so it is a great way to get to know how your choices affect the display of your data. You can also configure your data, then try various visualizations until you find the best way to show your data. Then try the visualization panel, which presents choices for every aspect of the visualization from legends to tooltips.

## Pointers for Getting Started

Here are some helpful pointers for getting started:

- **Data sources:** the data source you want to use in a dashboard must be added to the dashboard first. See *Prerequisites* in the *Creating Dashboards* topic for details.
- **Explore:** use the explore feature when you just want to explore data without creating a dashboard. You can always turn your exploration into a dashboard.
- **Dashboards:** the starting point for creating a dashboard is the "+" icon in the upper right, as highlighted in the image below. If you don't see the + icon, click the caret in the top row to open the top bar.
- **Alerts:** click the bell icon at the very top right to view or manage notifications.
- **User preferences:** click the human icon in the very top right, then click 'User Preferences' to open the Settings app.
- **Logout:** log out of Dashboard Designer by clicking the human icon, then selecting 'Logout'.
- **Search:** clicking in the search bar displays a list of dashboards, actions, pages, and preferences for quick navigation.

This image shows some icons, information, and navigation pointers on the Dashboard Designer main page:





# Creating Dashboards

# 5

## 5.1 Creating Dashboards

Creating informative and visually appealing dashboards is pivotal in modern data analytics. Dashboard Designer offers a robust toolkit with a comprehensive data querying editor and a rich assortment of visualizations that provide at-a-glance insights into data metrics. This topic serves as a guide to creating dynamic dashboards that effectively represent complex datasets in visually compelling ways.

With an interconnected and comprehensive dashboard experience, you can:

- Import dashboards
- Link your dashboards to internal and external Internet locations
- Copy panels from other Dashboard Designer dashboards
- Add links to other Dashboard Designer dashboards



Only Admin and Creator users can create dashboards.

### Overview

There are three areas to configure when you create a new dashboard:

**Data source**--select the data source to use for the dashboard.

**Query**--select which data from the data source to use, and optionally add a transformation to the query results.

**Visualization**--select options for visualizing the query results on a dashboard panel.



Dashboard Designer includes a predefined set of data sources and does not support adding additional external data sources.

### About Internet of Things (IoT) Time Series Data sources

IoT time series data represents measurements of a physical asset's property taken at regular intervals over time; visualizing the series of data points makes it possible to see trends, patterns,

and anomalies in the 'big picture' data.

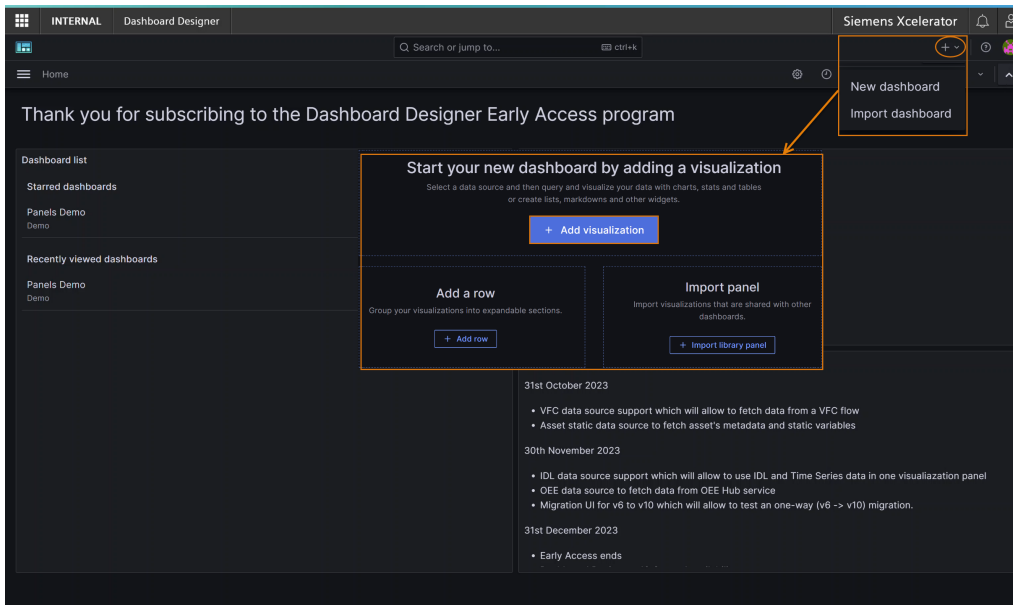
This topic uses the *IoT Time Series* data source to introduce and walk through the steps in creating a dashboard, but you can use these steps for any of the predefined data sources. Bear in mind that data sources use different types data and each displays their own unique settings in the query editor and visualization panel.



Go [here](https://documentation.mindsphere.io/MindSphere/apis/iot-iottimeseries/api-iottimeseries-apiratelimits.html) for details on how rate limits may apply to your time series data sources.

## Prerequisite: Add the Data Source to the Dashboard

First, you have to add the data source to your dashboard. This image shows how to navigate the data source selections described below:



### How to Add a Data Source to a Dashboard

Follow these steps to add an IoT time series data source to your dashboard:

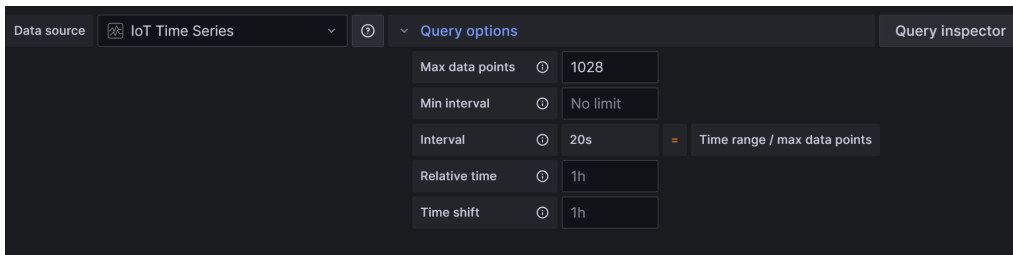
1. Click the + icon, and select "New dashboard" from the drop-down list. A pop-up window displays.
2. Click the "+ Add visualization" button. Available data sources display.
3. Select IoT Time Series. The data source is added and displays as Query 1 in the "Panel edit" area.



To better view all query editor selections, scroll down the page, or you can minimize the visualization by dragging up the divider bar beneath it.

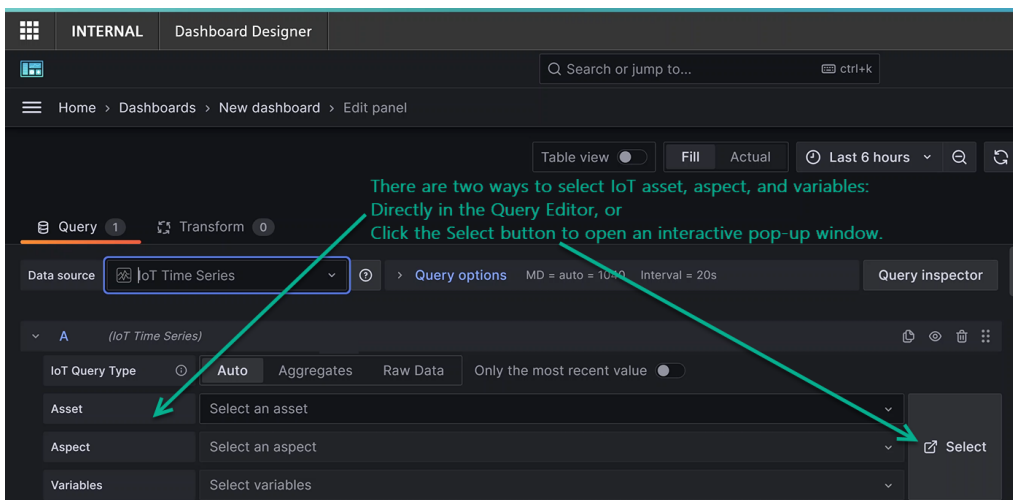
## Configuring the Panel Query

Query options appear above your first query in the panel. These options apply to **all queries** on your panel. The info icon details can help you determine which options to use for your panel, as shown here:



There are two ways you can select data for your query:

- Select options from the panel itself
- Click the large 'Select' button (shown below) to open a pop-up window:



The pop-up window allows you to see more of the aspects and variables available in an asset; the 'select all' feature is helpful when selecting multiple variables.

## Panel Edit/Query Editor

Fine-tuning query options is crucial to harness the full potential of data visualization. This section guides you through editing your query, with introductory information for each section. Feel free to skip the intro material and jump to numbered steps.

## How to Configure an IoT Time Series Query

Follow these steps to start configuring your IoT Time Series query:

1. Select Auto, Aggregate, or Raw Data for your query.
2. Toggle on 'Only the most recent values' if needed, and select a date range.
3. Select an asset from the Asset drop-down list. The next field displays available aspects.
4. Select an aspect from the Aspect drop-down list. The next field displays available variables.
5. Select variables one-at-a-time from the Variables drop-down list, or use the Select All check box to use all variables.



If you select the asset, aspect, and variable(s) directly in the query editor, you do not need to click the 'Select' button.

## Configuring Aggregate and Time Override/Transform



The transform option to increase the time range applies only to the query you're currently working with.

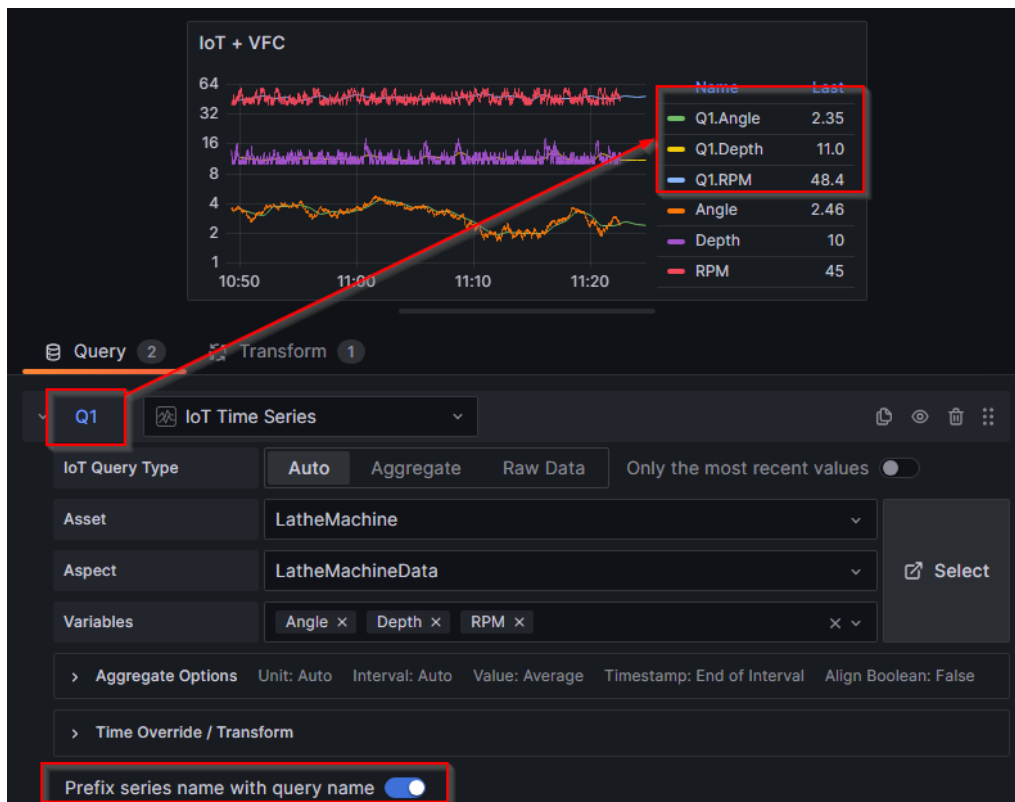
## How to Configure Aggregate and Time Override/Transform

Follow these steps to configure the remaining options for your query:

1. Click Aggregate Options (optional) to set options for unit, interval, value, whether to use the beginning or ending of timestamps, and whether or not to align boolean values.
2. Click Time Override/Transform (optional) to set options for increasing the time range, shift time, value to % of time, or to group by time.
3. Toggle on the 'Prefix series name with query name' if needed.
4. Toggle on 'Remove subsequent duplicate values' if needed.
5. Click "+ Query" to add more queries to the panel, otherwise continue to the next section *Configure the Visualization*. Your selections appear along the top of the Aggregate Options and Time Override/Transform sections.

## About Multiple Query Panels with Same Series Names

If you plan to add multiple queries to your panel that will produce datasets with the same series names, we recommend setting the 'Prefix series name with query name' toggle to "on". Here is an example of when the 'Prefix series name with query name' is toggled "on":



## Configuring Visualizations

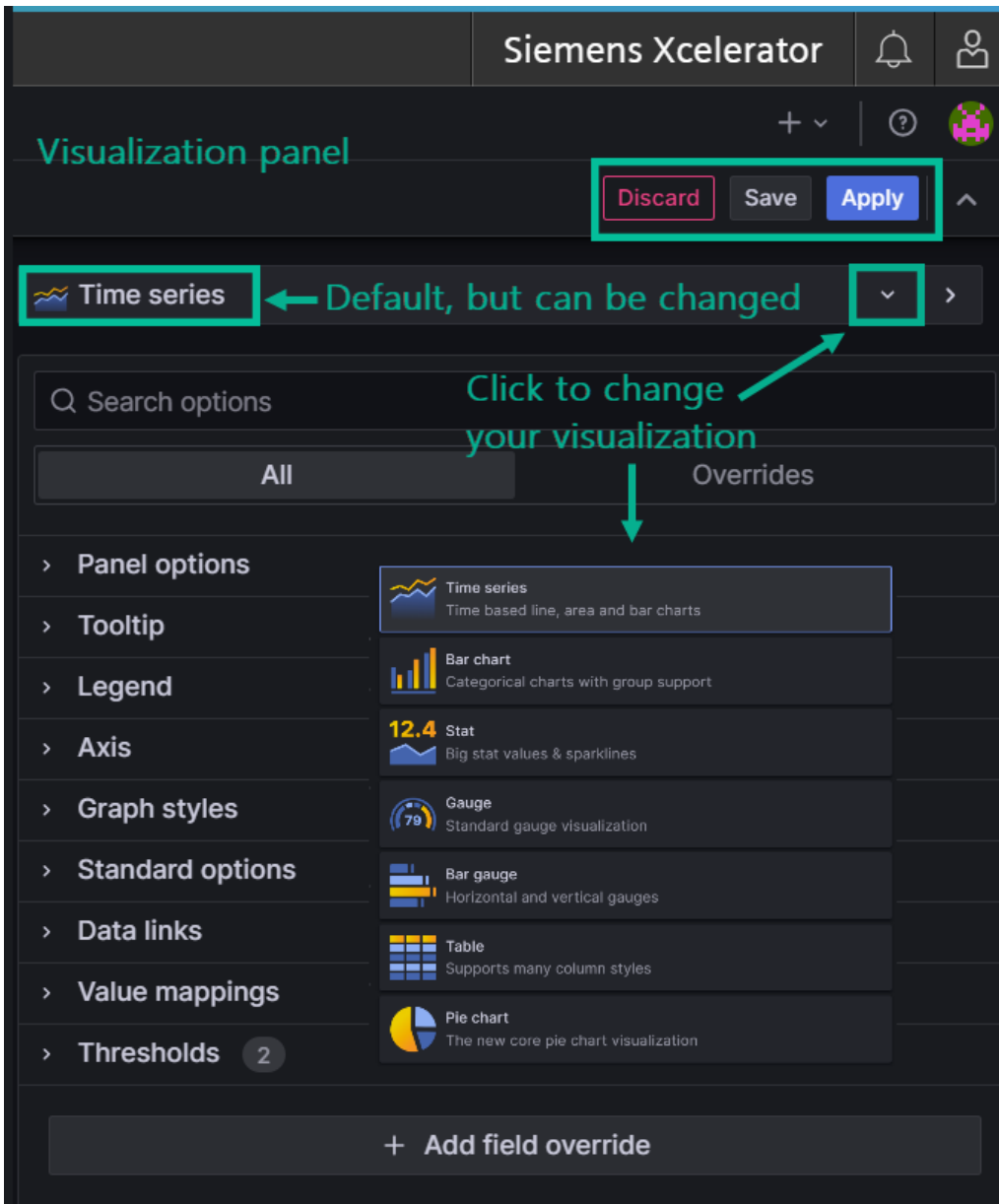
Just like each data source has certain fields that appear in the query editor, Dashboard Designer also loads the default visualization which, for the IoT data source is *Time series*, but this can be changed as shown in the image below. Based on your data, the suggestion tab can be helpful in exploring other visualizations.

## Visualization Panel Overview

Dashboard Designer includes a stunning number of ways to customize and refine visualizations by specifying values for anything that appears in or on a visualization - from the colors, shapes, and lines to the data, legends, and labels.

### Navigation Pointers

This image shows some navigation pointers for the Visualization panel:



The number of settings you configure in any of these sections displays next to the heading label.

## Panel Options

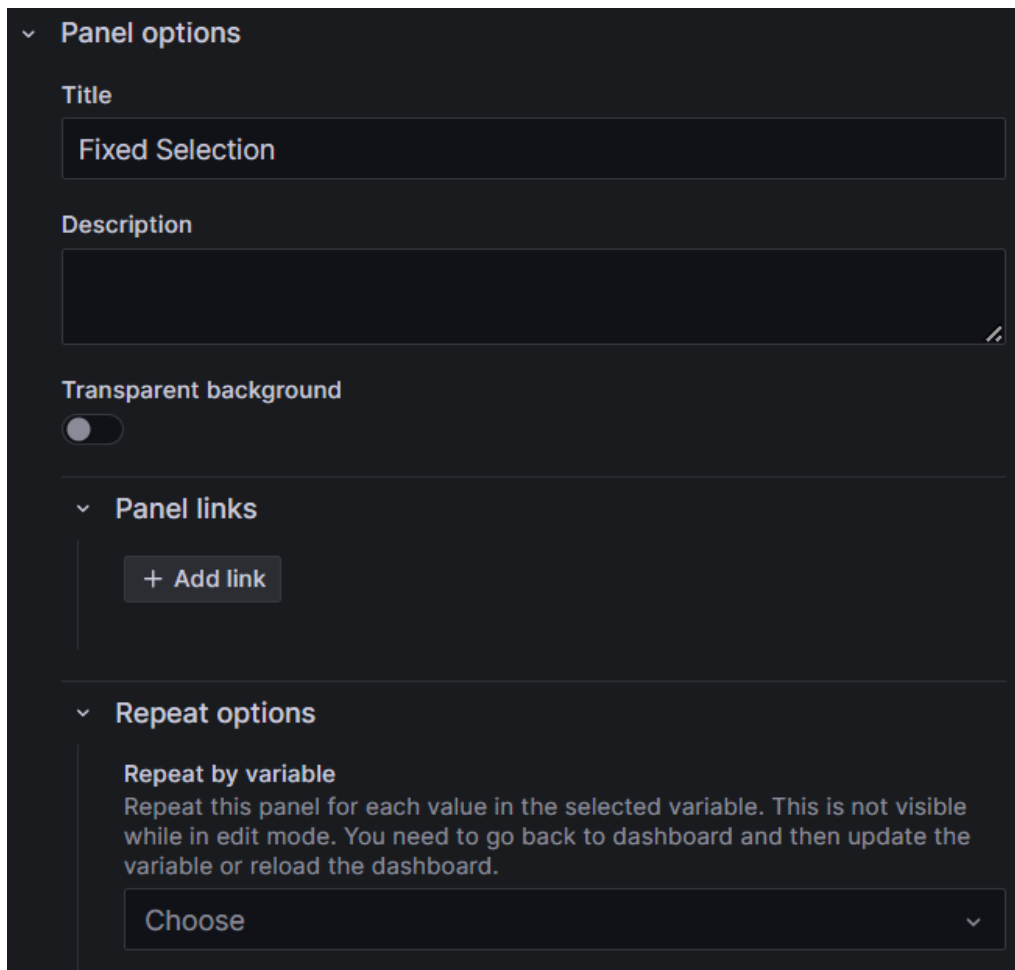
The first section, *Panel options* is where you name and optionally describe the panel and also add links, and set other preferences.

The *repeat* feature allows Dashboard Designer to repeatedly generate your panel for each variable of an asset or aspect you specify.

## How to Configure Panel Options

Follow these steps to configure the panel options:

1. Enter a title for your panel. Optionally enter a description.
2. If you want the background of the panel to be transparent, set the toggle to "on".
3. Click the '+ Add link' button to enter a link title, the URL, specify whether or not to open the link in a new tab, and save.



## Fine-Tuning Visualizations

You can navigate through the various panel features by expanding and collapsing the sections. Each area, described briefly below offers in-depth configuration options to tailor a visualization according to analytical requirements and audience.

## Tooltips

Configure visibility, sort order, and other tooltip-related settings.

## Legends

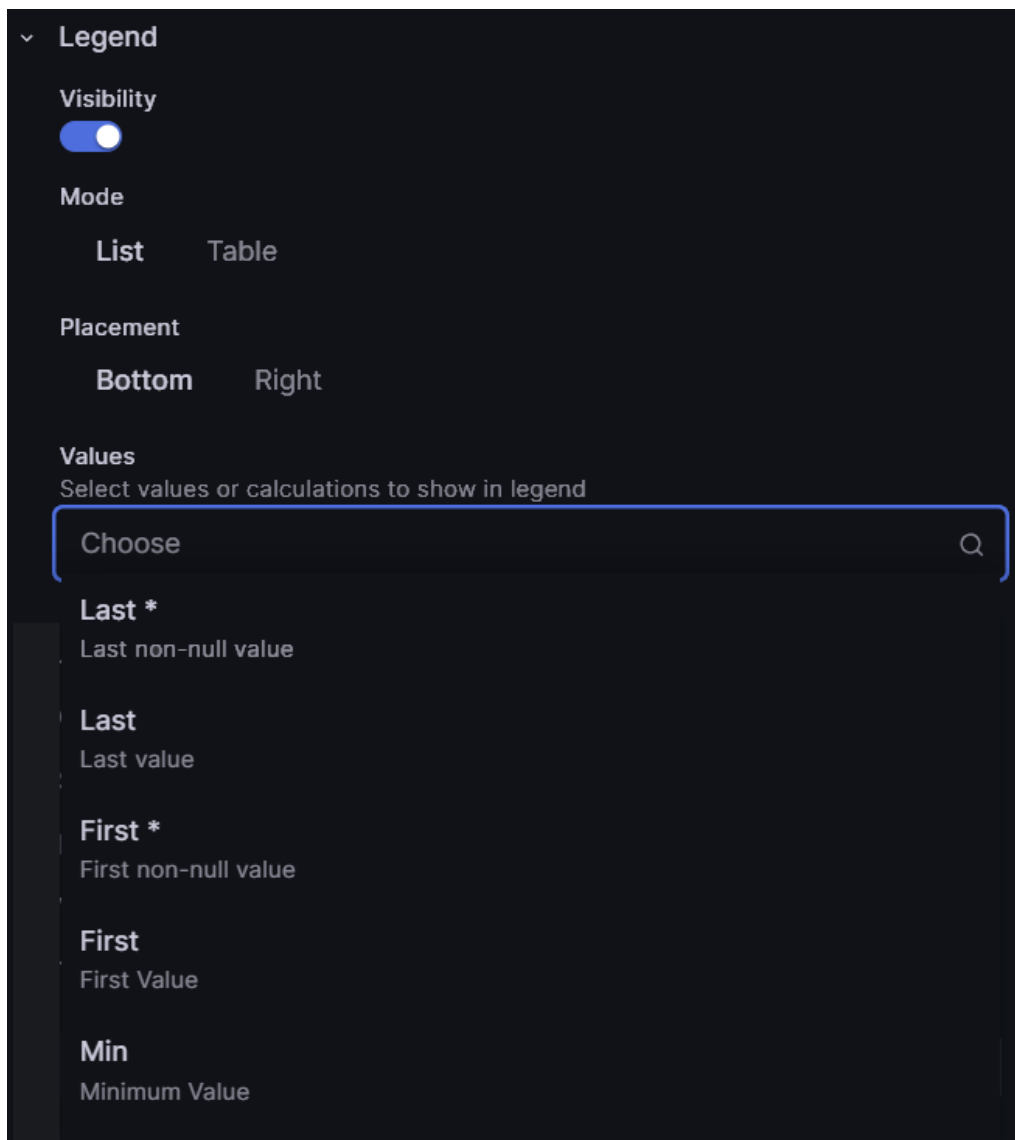
Tailor the display and positioning of the legend, choose list or table format, and select specific values to appear in the legend.

## How to Configure Legend Options

Follow these steps to set legend options:

1. Toggle the visibility to 'on' to display the legend on your visualization.
2. Set whether to display the legend as a list or table and whether or not to place it at the bottom or right of the visualization.

3. If you want to display values in the legend, select the values you want to show from the drop-down list. Your selections display in the Values field as you choose them.



## Axis Settings

Adjust visual aspects such as placement, color, grid lines, and more, related to the axes. Here is an example of the axis settings:



Axis

Time zone  
Default

Placement  
Auto Left Right Hidden

Label  
Optional text

Width  
Auto

Show grid lines  
Auto On Off

Color  
Text Series

Scale  
Linear Logarithmic Symlog

Centered zero

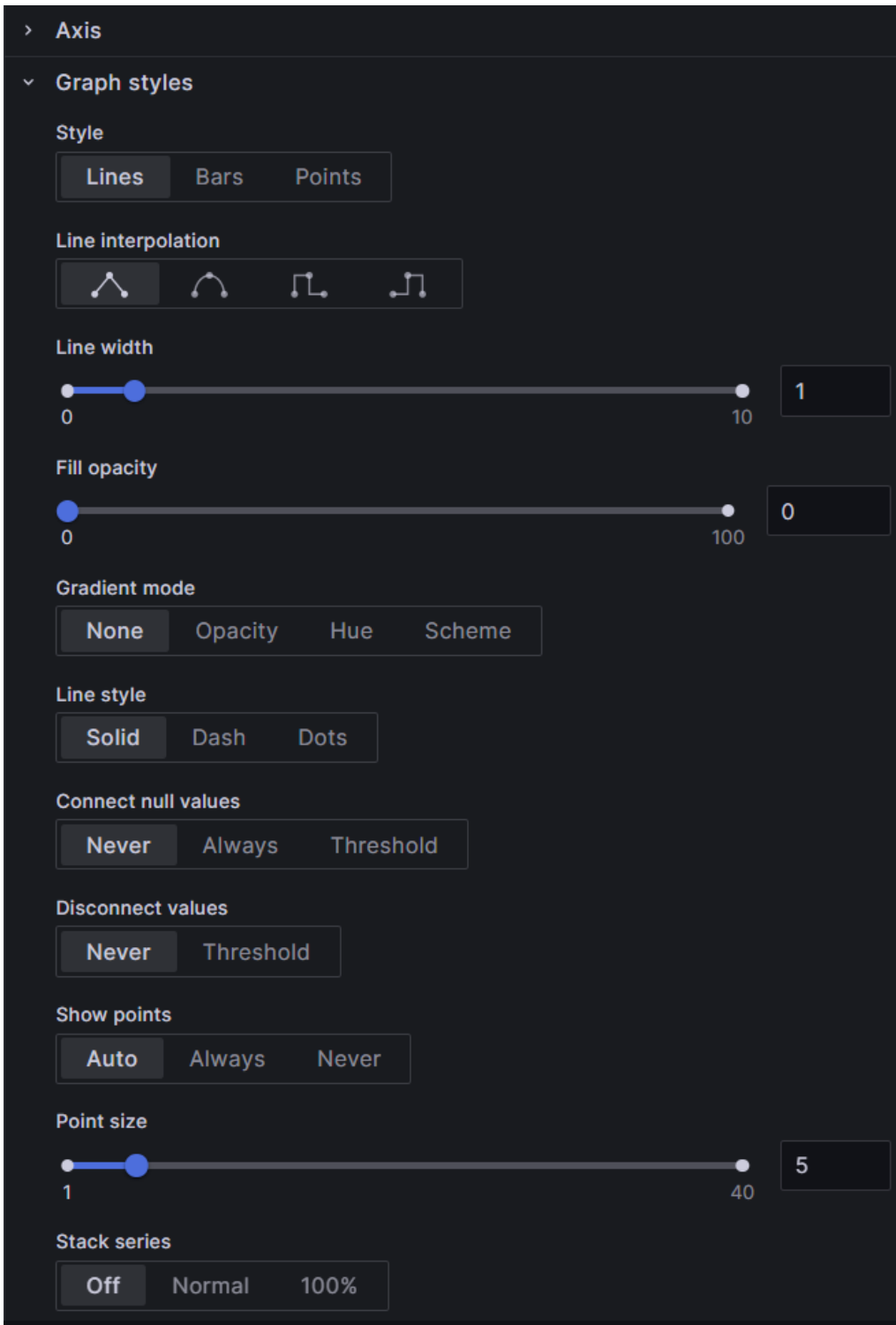
Soft min  
See: Standard options > Min

Soft max  
See: Standard options > Max

## Graph Styles

Modify graph styles, including lines, bars, points, line characteristics, color, and additional visual options.

Here is an example of the graph style settings:



### Standard Options

Customize various fields affecting the panel's appearance, such as units, color palette, min-max values, decimals, etc. The first selection, for example, is for the unit in which graph or chart data will be rendered, followed by the measurement standard, such as degrees, arc minutes, etc. for the selected unit. Other standard selections include color palette, min, max, decimals, and so on.

### How to Configure Standard Options

Follow these steps to set standard options:

1. Select a unit from the drop-down list. A list displays various measurement standards for the unit.
2. Select the measurement to use.
3. Continue to accept the defaults or make selections for the rest of the fields for your visualization.

Here is an example of the standard options:

Standard options

Unit  
Choose

Min  
Leave empty to calculate based on all values  
auto

Max  
Leave empty to calculate based on all values  
auto

Decimals  
auto

Display name  
Change the field or series name  
none

Color scheme  
Classic palette

No value  
What to show when there is no value  
-

## Data Links

Add and configure links to external resources relevant to the visualization. You can only set one-at-a-time.

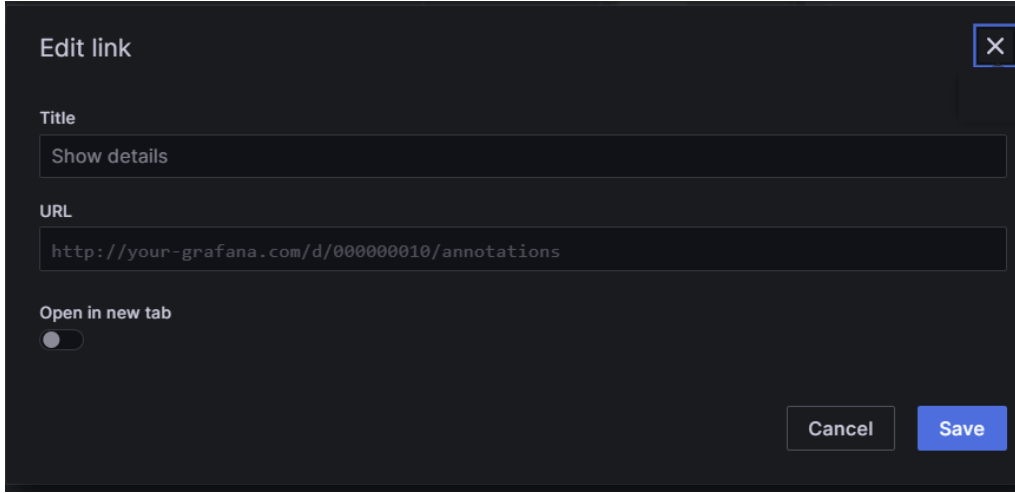
## How to Create a Data Link

Follow these steps to set data links:

1. Click the '+Add link' button. The 'Edit link' pop-up window displays.
2. Enter a title that will display on your panel.

3. Enter the URL for the link - according to the example in the URL field.
4. If you want the link to open in a new tab, move the toggle to 'on'.
5. Click the 'Save' button.

Here is an example of the data links pop-up window:



## Value Mapping

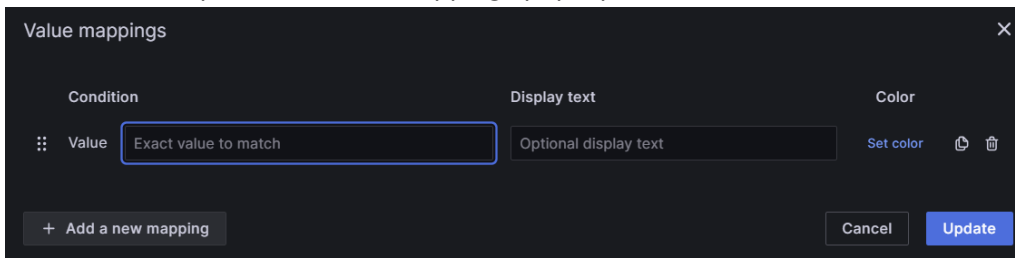
Assign specific colors and optional text to values in the data set.

## How to Map Values

Follow these steps to set value mappings:

1. Click the 'Add value mapping' button. The 'Value mapping' pop-up window displays.
2. Enter a specific value in the 'Condition' field.
3. If you want the value to display in color, click the 'Set color' link. A color palette displays.
4. Select a color.
5. Continue to add additional mappings by clicking the '+ Add a new mapping' button.
6. Click the 'Update' button.

Here is an example of the value mappings pop-up window:



## Threshold Options

Set thresholds for values and define display preferences for them, including absolute or percentage-based, and visualization options.

## How to Configure Thresholds

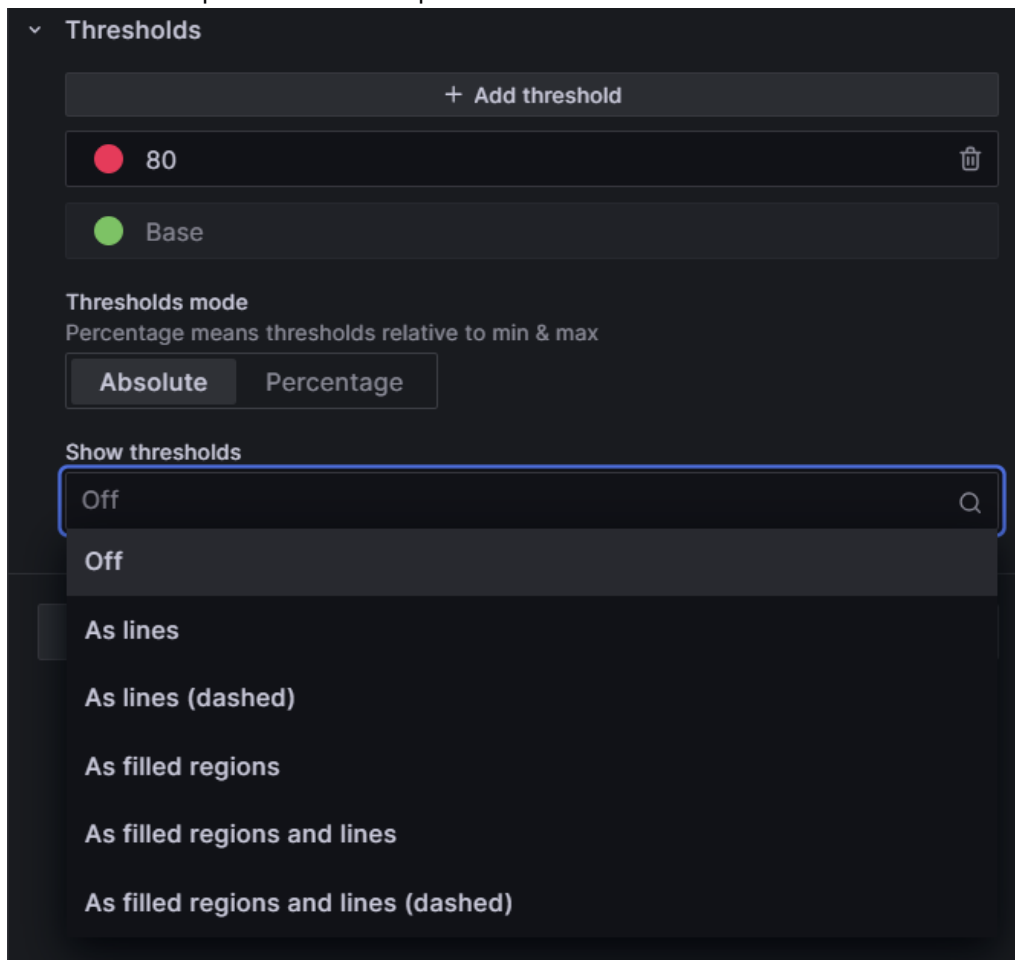
Follow these steps to set threshold options:

1. Click the '+Add threshold' button. A threshold row appears at the top of the thresholds list with a default value and color.
2. Enter a number to replace the default value, or use the up/down arrows.
3. Click the default color dot and select a color from the color palette.
4. Select the delete icon to delete any threshold rows you don't want.
5. Leave the 'Threshold mode' on 'Absolute', unless your threshold values are relative to min and max values; if so, slide the toggle to 'Percentage'.
6. Select how you want the thresholds to display from the 'Show thresholds' drop-down list. (See image below).



The 'Base' threshold row cannot be deleted.

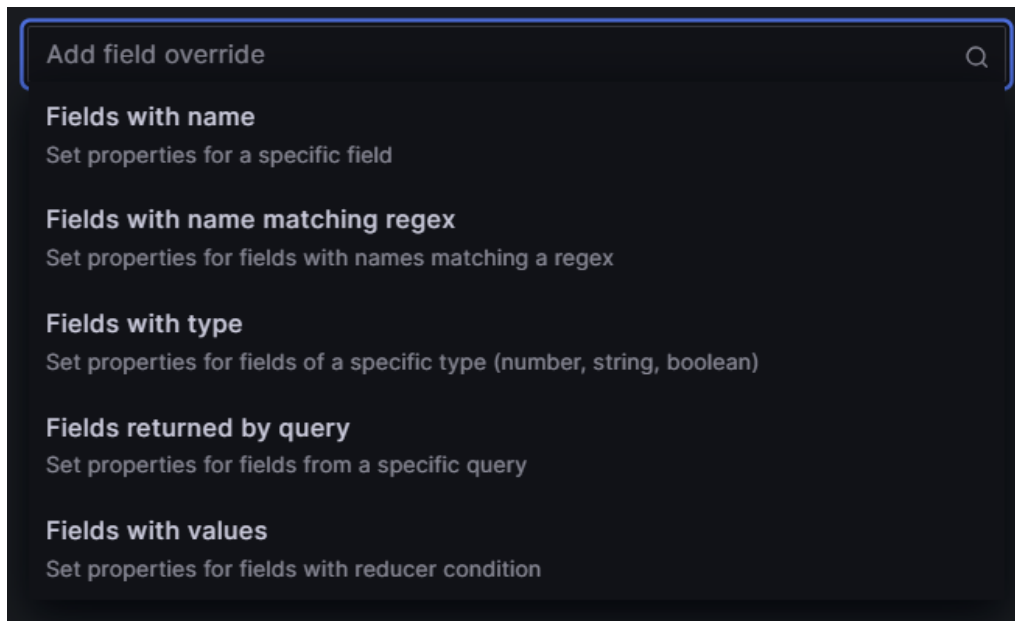
Here is an example of threshold options:



## Overriding Fields

Specify overrides for specific fields or field types, allowing for fine-tuning according to the data properties and queries.

You can override settings for any visualization field. Here is an example that illustrates categories of fields you can search by to locate the kind of field you want to override:



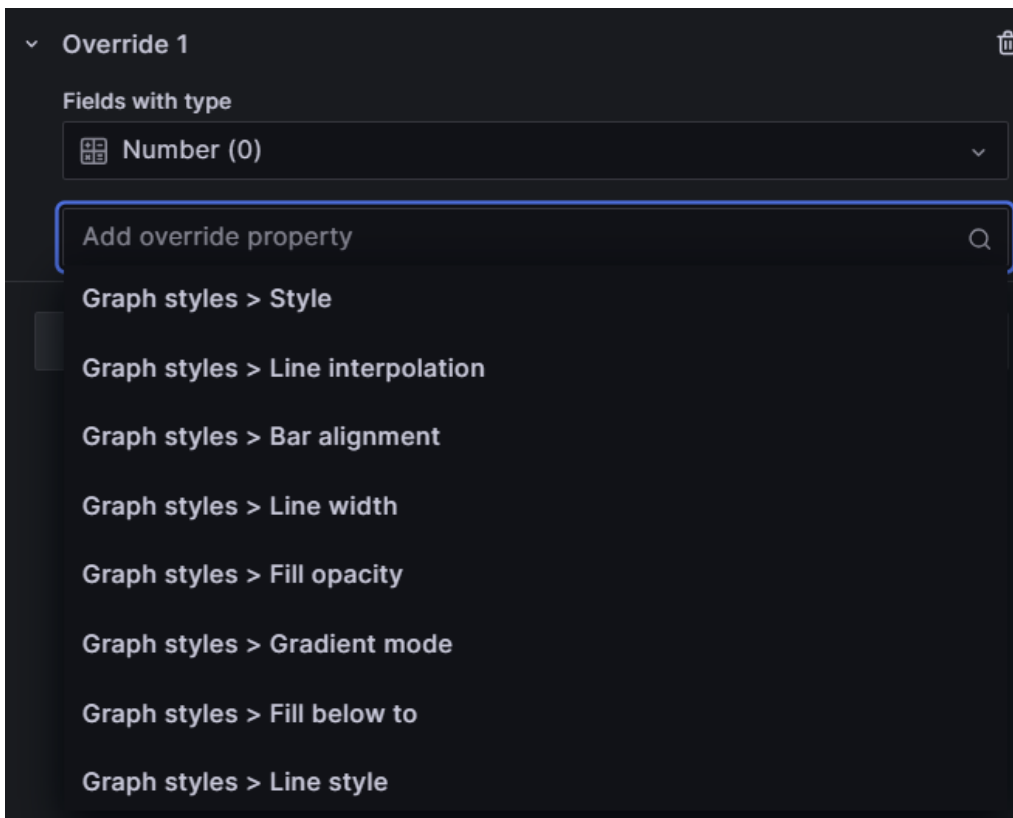
The following step-by-step uses 'Fields with type', then 'Number'. The steps are the same for any of the 'field types' you select. The subsequent drop-down list displays properties relevant to the field type you select.

## How to Override a Field

Follow these steps to configure the field override option:

1. Click the '+Add field override' button. A drop-down of types displays. (Image below.)
2. Select 'Fields with type'. The drop-down list closes and a 'Fields with type' label displays over a drop-down list of field types.
3. Select 'Number'.
4. Click the '+ Add override property' button. (Image below.)
5. Select an option from the 'Add override property' drop-down list.
6. Select and option from the additional values that display.

Here is an example of field override options for the 'number' field type:

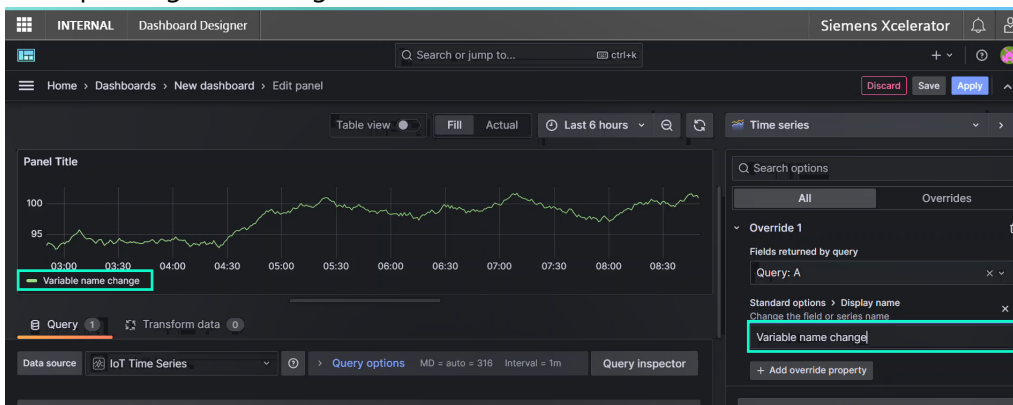


## Changing a Variable Name in a Visualization

Overriding and changing the display name of an IoT variable returned from a query involves modifying the variable settings.

### Illustration of a Variable Name Change Visualization Panel

This shows the variable name being changed on the Time series visualization panel and the corresponding label change on the visualization:



## How to Change the Variable Name that Displays

Open the dashboard with the query variable you want to change and follow these steps:


1. Click the '+Add field override' button as shown in the image above. A drop-down of types displays.

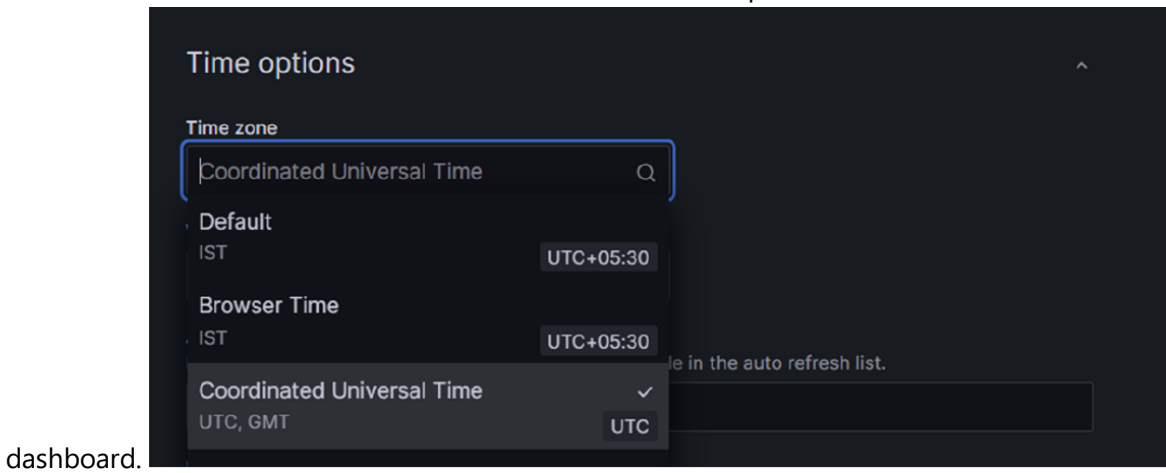
2. Scroll and select 'Fields returned by query'. The drop-down list closes and a 'Query: A' field displays a list of variables.
3. Select the variable you want to change the name of. A list of change options displays.
4. Scroll and select 'Standard options > Display name'. A field opens for entering the new name for the variable.
5. Enter the new name and move the cursor out of the field. The new name displays on the visualization. (image below)

## Configuring Dashboard Designer Time Zone

By default, Dashboard Designer displays data according to the browser's time range, but data from IDL and IOT are stored in UTC time range. To get optimal results when comparing data between Asset Manager and Dashboard Designer, it is recommended to modify Dashboard Designer's configuration to display data in UTC.

To change the time zone configuration, proceed with the following steps.

1. Click the  icon in the toolbar of "New dashboard" page.
2. Select Coordinated Universal Time from the Time zone drop-down list and save the





# Functions

# 6

## 6.1 Functions

Functions can transform raw data into useful formats for meaningful dashboards and they only display for Insights Hub or Default data sources.

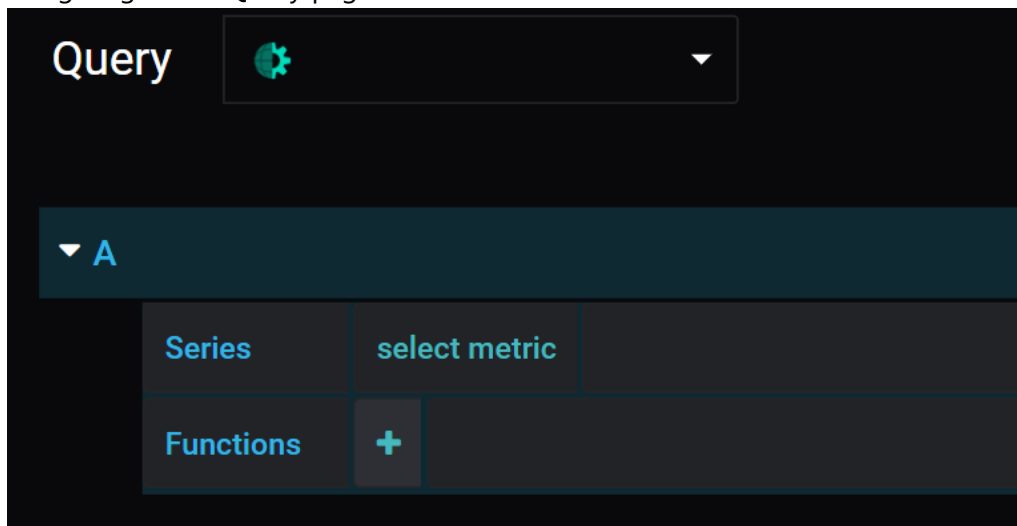
Data requested by Dashboard Designer is automatically aggregated by default. If your raw data must be queried in a different way, you can use functions to modify your query. Any functions added to a query override default query settings.



Dashboard Designer is not available for Private Cloud.

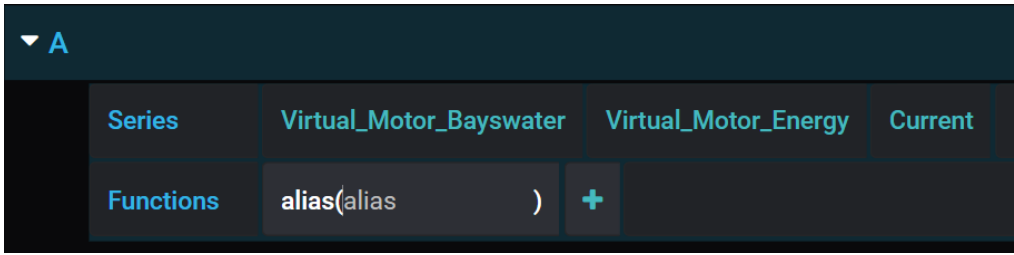
## Functions Overview

A query can have multiple functions and you can access them by opening a panel and navigating to the Query page:



## Adding a Function to a Query

When you build a query, you can add functions to it by hovering over the + symbol. Click the + symbol to open a scrollable drop-down list of functions. You can add multiple functions to a single query.



### How to Add Functions to a Query

Follow these steps to add a function to a query:

1. Click the + icon next to "Functions".
2. Hover over the data source name ("Insights Hub") and select a function from the drop-down list.

### Removing a Function from a Query

Click a function name to open a pop-up window, and select "x" to remove a function.

### Getting Help on Using a Function

Click a function name to open a pop-up window, and select "?" to see information on the function.

### How to Create an Alias for a Query String

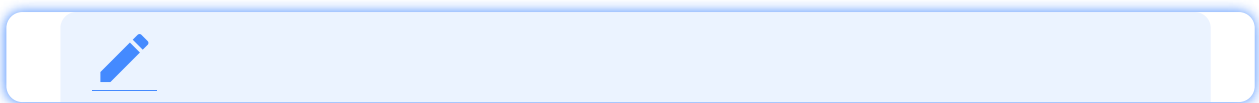
In graphs and charts, each variable displays the full query string in the legend. Queries are usually long and can clutter up the legend, so the alias function allows you to substitute a short name for the query string.

Syntax: `alias(QUERY, alias)`

Follow these steps to create an alias:

1. Create a query and select alias from the list of functions. The tag 'alias' displays in the function row and a text cursor displays inside the bracket.
2. Enter inside the brackets the alias you want to use for the variable.

To change the name again, click the text inside the brackets. Spaces are ignored but underscores can be used.



If you deselect the function before typing anything inside the brackets you will have to remove the function and add it again to rename the variable.

## Dashboard Designer Functions and Syntaxes

This section describes the various functions available, when to use each function, and gives examples where helpful.

### Alias

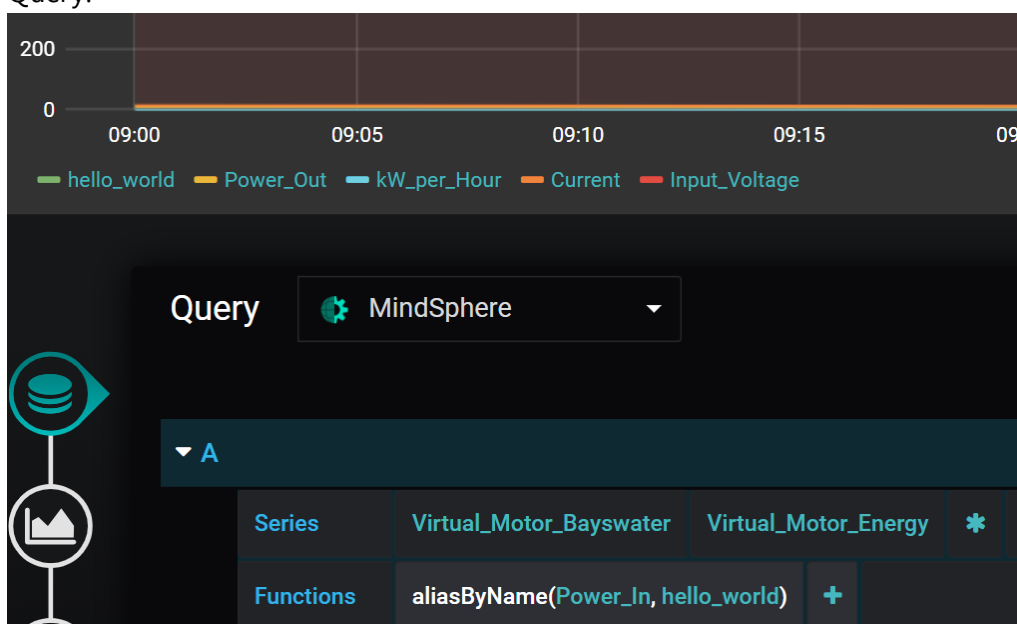
For single variable queries, renames a series to the user's input.

Syntax: `alias(QUERY, alias)`

### AliasByName

For multi-variable queries, renames a series to the user's input; use it when you want to call an entire aspect. Add \* as the final query in the string to return all aspect variables; in this scenario, the function will rename all variables to have the same name.

If you want to rename individual variables, select the `aliasByName` function from the function drop-down list. You can also stack this function with itself, using `aliasByName_stacked` which allows you to rename each variable. Here is an image that shows the 'aliasByName' function in a Query:



Syntax: `aliasByName(QUERY, name, Replacement String)`

### AliasByTime

Returns grouped series under aliases based on time, and is useful for comparing data from different time periods. The input for `aliasType` can be `weekdays`, `weekNumbers`, `months`, or `years`. The `timezoneOffset` input adds a time offset from UTC in hours and can use positive or negative numbers.

Syntax: `aliasByTime(QUERY, aliasType, timezoneOffset)`

The first image below shows data returned with no `AliasByTime` function, followed by the same data, but grouped by the weekday of the data's timestamp:





### IgnoreValue

Returns queried data without the value(s) specified, and is useful for filtering out certain values from a query.

Syntax: `ignoreValue(QUERY, value)`

Value input can be:

Syntax	Meaning
*	wildcard - matches everything
?	matches any single character
[seq]	matches any character in seq
[!seq]	matches any character NOT in seq

### IgnoreVariable

Returns queried data without the variable(s) specified, and is useful for filtering variables when the query has a variable wildcard e.g.Asset/Aspect/\*.

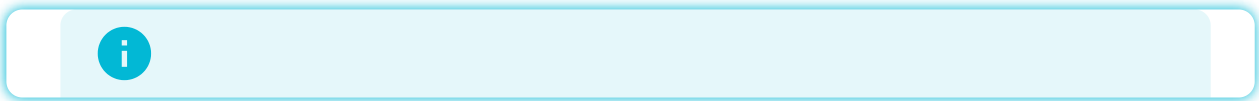
Syntax: `ignoreVariable(QUERY, variable)`

#### Variable input can be

Syntax	Meaning
*	wildcard - matches everything
?	matches any single character
[seq]	matches any character in seq
[!seq]	matches any character NOT in seq

For example, for this list of asset names, [Motor\_12, Motor\_15, Motor\_17, Gear, Shaft], your input would be:

```
Syntax: Motor_* -> [Motor_12, Motor_15, Motor_17]
Motor_1[27] -> [Motor_12, Motor_17]
??a* -> [Gear, Shaft]
[MG]*[!57] -> [Motor_12, Gear]
```

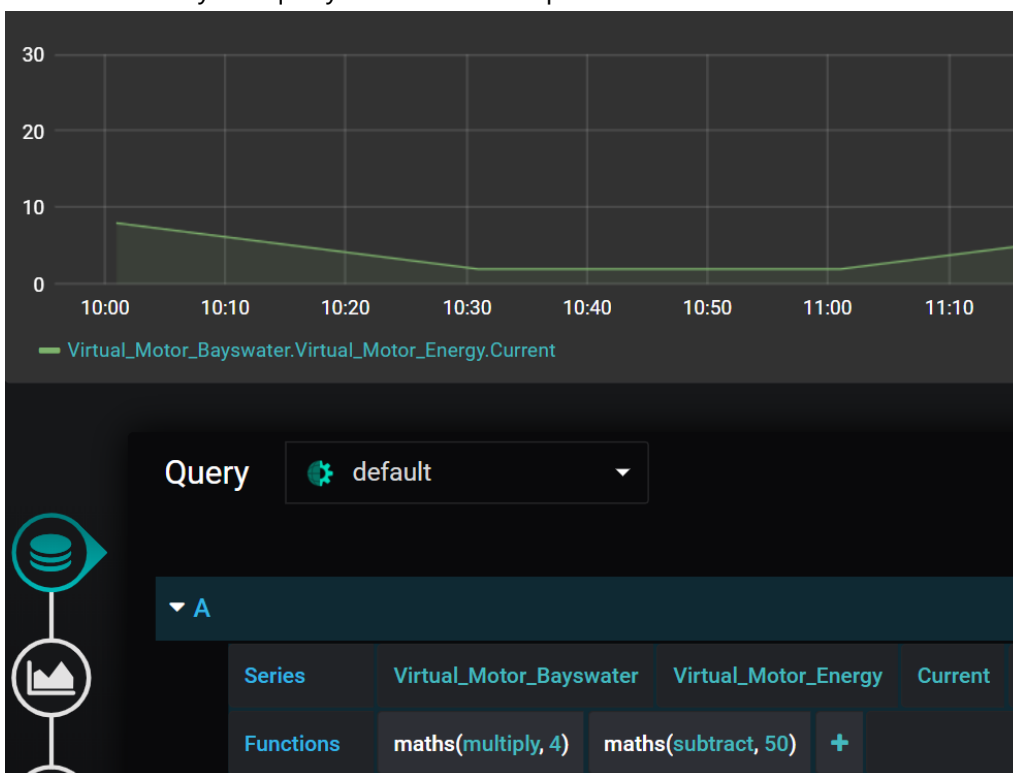


The 'ignoreVariable' function utilizes a Unix-style 'fnmatch' pattern and you can find more information here: <https://docs.python.org/3/library/fnmatch.html>

## Math

Performs basic mathematical operations on returned variable data.

Math functions can be stacked, and they execute in the order written. Math functions apply to all data returned by the query. Here is an example:



Syntax: `maths(QUERY, function, value)` or `math_stacked`

## MSrawTimeseries

Returns raw data from a Insights Hub data source and overrides the default aggregation; instead, it returns non-aggregated or non-interpolated time series data.

When querying strings, use this function to retrieve the data, as Insights Hub's default aggregation cannot aggregate strings. This function can only retrieve up to the 2000 most-recent time series data points; a data point can contain multiple values if they share the same time index.

Syntax: `MSrawTimeseries(QUERY)`

## MSassetID

Returns Insights Hub's Asset ID for the asset of the variable you select.

Syntax: `MSassetID(QUERY)`

## MSmostRecentValueWithinTimeRange

Retrieves the most recent value within the queries time window using the IOT TimeSeries API.

If this function does not retrieve values reliably, please use *MSaggregate(lastvalue)*.

This function retrieves the latest time index of the aspect that the variable is within. If the variable is not present in the latest time index it returns nothing. To avoid, try separating the data points you always wish to retrieve the last value of and put them in their own aspect or make sure their data is written to the latest time index.

Syntax: `MSmostRecentValueWithinTimeRange(QUERY)`

## MSmostRecentValueWithinNowto90Days

Retrieves the most recent value within the last 90 days, using the IoT TimeSeries API, and regardless of the query's time window.

This function is useful for dashboards that look within a small range of time, such as a day, when you want to display the most recent value of less frequent data that does not fall within the time range. If this function does not retrieve values reliably, please use `MSaggregate(lastvalue)`.

It retrieves the latest time index of an aspect that the variable is within. If the variable is not present in the latest time index it will return nothing. To avoid, try separating the data points. you always want to retrieve the last value of and put them in their own aspect or make sure their data is written to the latest time index.

Syntax: `MSmostRecentValueWithinNowto90Days(QUERY)`

## MSboolAggregates

Returns a boolean value, according to automatically selected time ranges.

Standard Insights Hub aggregation rules cause boolean values to be misrepresented because the average aggregation rule causes the booleans to be displayed as decimals between 0 and 1.

This function aggregates the booleans and returns a boolean value by looking at the average value of booleans within the aggregated time range. If the value is below 0.5 it becomes a 0, if it is above it becomes a 1.

It allows aggregation of booleans, but still accurately represents them.

Syntax: `MSboolAggregate(QUERY)`

## MSautoAggregate

Returns automatically aggregated data based on aggregation method.

Dashboard Designer automatically calculates time ranges in order to automatically aggregate data using the 'average' aggregation method. This function gives power users the option to choose their own aggregation method, without having to calculate time ranges and groupings of data.

The IoT Time Series (TS) Aggregates Service creates aggregated summaries of numeric time series data and provides interfaces to read them. This allows applications to retrieve smaller datasets that cover a long time range with much better performance than processing all the raw time series data.

For example, an aspect could create new data every second, which adds up to ~2.5 million records per month. An application could use the IoT TS Aggregates Service to retrieve a summary for each day of the month, obtaining only 30 records.

Syntax: `MSautoAggregate(QUERY, method)`

## **MSaggregate**

Returns aggregated data based on the specified intervals and method.

Dashboard Designer automatically calculates time ranges to automatically aggregate data by default. This function gives users the option to fully customize how aggregates are calculated.

The IoT Time Series (TS) Aggregates Service creates aggregated summaries of numeric time series data and provides interfaces to read them. This allows applications to retrieve smaller data sets that cover a long time range with much better performance than processing all the raw time series data.

For example, an aspect could create new data every second, which adds up to ~2.5 million records per month. An application could use the IoT TS Aggregates Service to a summary for each day of the month, obtaining only 30 records.

Syntax: `MSaggregate(QUERY, intervalUnit, intervalValue, method)`

## **onlyChanges**

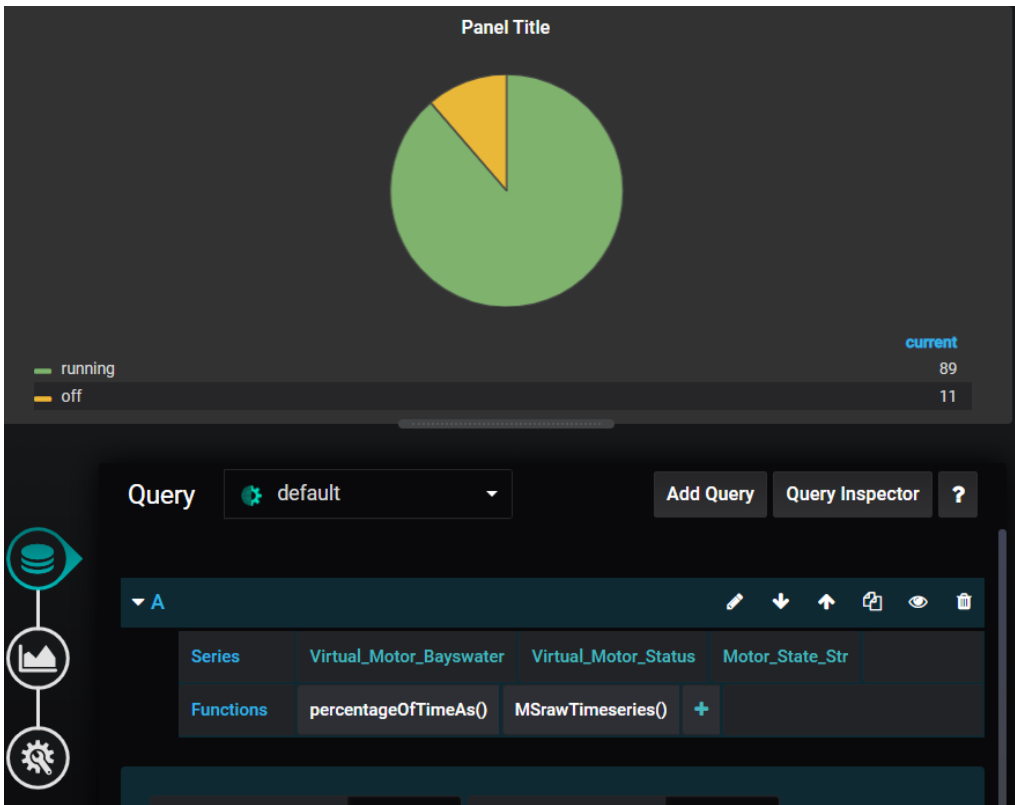
Returns only the rising and falling edges of the query.

Syntax: `onlyChanges(QUERY)`

## **percentageOfTimeAs**

Returns the amount of time each unique value was held as a percentage of the total time range, which is useful for displaying machine states and summarizing the time elapsed in each state.

This function can also stack with other functions; for example, if a variable is a string, you can stack "percentageOfTimeAs" with "MSrawTimeseries", as shown here:

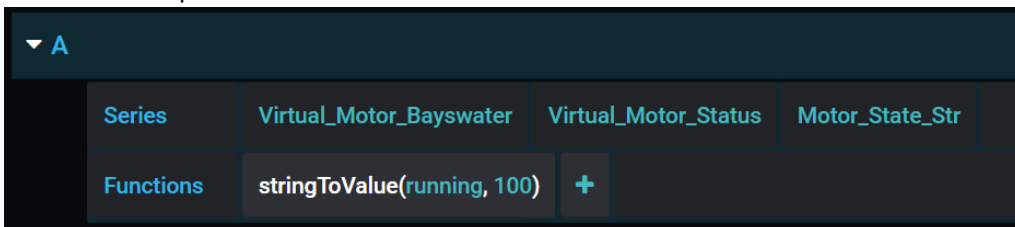


Syntax: PercentageOfTimeAs(QUERY)

### StringToValue

Replaces strings with a number.

When displaying strings, such as in a table, you may need to change a word to a number or another descriptor, as shown here:



For example, whenever the string 'running' is found in the data it will be returned as the new designated value of '100':

Time	Virtual_Motor_Bayswater.Virtual_Motor_Status.Motor_State_Str
2020-06-02 17:32:57	100.00
2020-06-02 17:29:45	off
2020-06-02 17:28:19	100.00
2020-06-02 17:26:57	100.00

Syntax: stringToValue(QUERY, string, replacementValue)

### TimeOf

Returns the timestamp of the data points in Unix time. Use this function when you need to know the timestamp of a dataset, you can from the function.

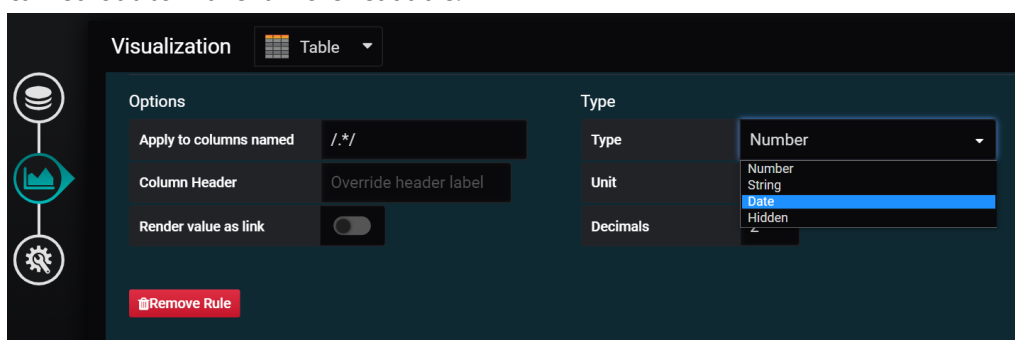
### unixTime



Making timestamps more readable in a Table panel requires that you define the datatype as 'date' in the table options. Navigate to options in the visualization tab and select 'date' from the dropdown list next to 'type'.

Time	Virtual_Motor_Bayswater.Virtual_Motor_Status.Motor_State
2020-06-02 23:04:00	1.59 Tri
2020-06-02 22:08:00	1.59 Tri
2020-06-02 21:04:00	1.59 Tri
2020-06-02 17:36:00	1.59 Tri
2020-06-02 17:28:00	1.59 Tri

Because the timestamp is returned as a Unix timestamp, some additional steps may need to be carried out to make it more readable:



The results are much more readable:

Time	Virtual_Motor_Bayswater.Virtual_Motor_Status.Motor_State
2020-06-02 23:04:00	2020-06-02T23:04:00+10:00
2020-06-02 22:08:00	2020-06-02T22:08:00+10:00
2020-06-02 21:04:00	2020-06-02T21:04:00+10:00
2020-06-02 17:36:00	2020-06-02T17:36:00+10:00
2020-06-02 17:28:00	2020-06-02T17:28:00+10:00

Syntax: `timeOf(QUERY)`

### date\_type

Displays the value in a human readable fashion.

Syntax: `date_type`

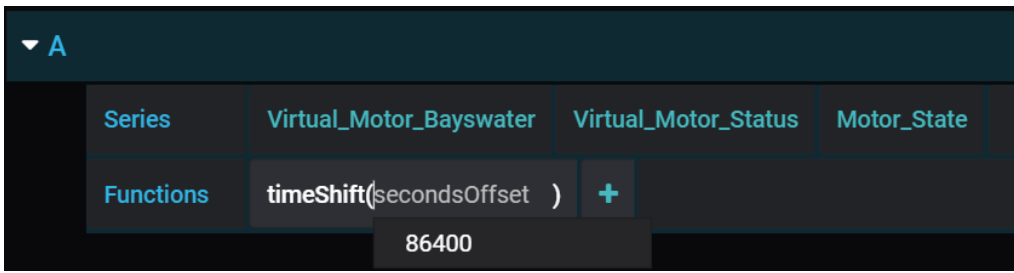
### date\_readable

Other panel types can also support dates (e.g. single stat). When using the `timeOf` function be sure to change the type or unit of the panel if it is to be displayed in a readable manner.

Syntax: `date_readable`

### timeShift

Offsets a query by 86400 seconds (24 hours) by default. This is useful for overlaying data from the previous day with today's data, or overwriting the time zone of the data:



When you set up two queries for the same variable, you can overlay one with the offset and one without:



Syntax: `timeShift`

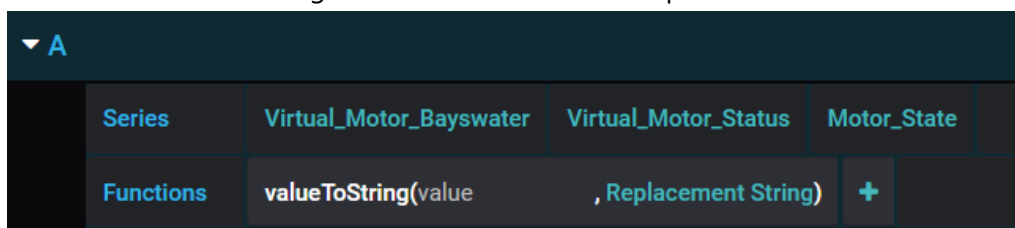
### timeShift\_noalias

When you notice that data displays with incorrect coloring and joining between points, this is because the graph colors and joins data based on its name. You can fix this by using the alias function to rename any variables that share the same name.

Syntax: `timeShift_noalias`

### ValueToString

Replaces a value with a string. For example, in a table that displays machine state values, you can use this function to change the value to a more descriptive word:



Here is an example of a query looking for instances of "0" in the data and returning a user-selected string, "Machine\_off":

Time	Virtual_Motor_Bayswater.Virtual_Motor_Status.Motor_State
2020-06-02 23:04:00	Machine_off
2020-06-02 22:08:00	Machine_off
2020-06-02 21:04:00	Machine_off
2020-06-02 17:36:00	0.33
2020-06-02 17:28:00	0.33

Syntax: `valueToString(QUERY, value, replacementString)`

### valueToString\_filled

When the value '0' is found in data, it returns the new designated string, for example, 'Machine\_off'. Replacement strings do not support spaces.

Syntax: `valueToString_filled(QUERY, value, replacementString)`

### VFCrequest

Creates a request to the http in blocks in Visual Flow Creator (VFC).

The method parameter in the VFCrequest function defines the method of the call. The format of the endpoint parameter requires an endpoint in this format:

`/public/<tenant>/<route>?key= <secret>`

This function triggers a flow in VFC, and data can be returned from VFC to Dashboard Designer for visualization, which requires a subscription and access to VFC.

There are also limitations with this function, including:

- Only the 'Get' method is supported and Dashboard Designer selects it by default.
- Panels that use this function cannot have other queries added to the panel.

### queryParams

Additional custom parameters sent to VFC as a part of the request. You can create a compatible flow in VFC by setting up the http in node as 'httpin\_node'. Double click to open settings and enter the endpoint name to use.

Syntax: `VFCrequest(method, endpoint, queryParams) httpin_endpoint`

### httpin\_access

Changes the access method to public access using keys: `httpin_generate`, saves the flow, and copies the link address of the http in node.

## httpin\_key

As the endpoint parameter, this prepares the URL for use by the `VFCrequest` function in Dashboard Designer.

When copying the URL, please remove the host URL preceding `"/public"`.

The flow must have an "http out" at the end to cause VFC to respond, as shown in the example flows below.

Once the endpoint is created, the flow will run every time the panel refreshes.

To trigger a VFC flow that displays data in Dashboard Designer, use the following JSON structure:

```
[
  {
    "datapoints": [
      [
        value,
        unix timestamp
      ],
      [
        value,
        unix timestamp
      ],
      [
        value,
        unix timestamp
      ]
    ],
    "tag": {
      "name": "variablename1"
    },
    "target": "variablename1"
  },
  {
    "datapoints": [
      [
        value,
        unix timestamp
      ],
      [
        value,
        unix timestamp
      ],
      [
        value,
        unix timestamp
      ]
    ],
    "tag": {
      "name": "variablename2"
    },
    "target": "variablename2"
  }
]
```

Here is an example that uses real data:

```
[
  {
    "datapoints": [
      [
        2384.045,
        1598382900
      ],
      [
        2387.24,
        1598383200
      ],
      [
        2414.01,
        1598383500
      ]
    ],
    "tags": {
      "name": "POWER_GENERATION"
    },
    "target": "POWER_GENERATION"
  }
]
```

## Security information

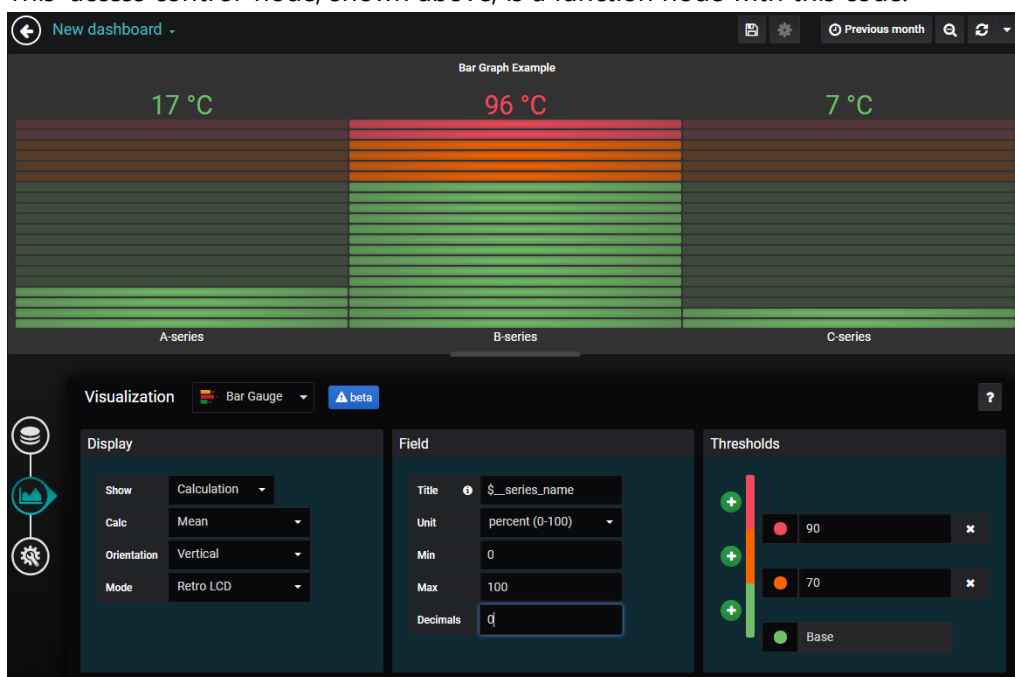
Creating an http in node with a public access key means that the particular flow triggered by the http in node can potentially be accessed externally. Thus, it is important to protect your data and prevent unauthorized access. Please handle the access keys with care.

## Creating an Access Control Node

Create an "access control" node after the http in node so it's possible to determine that the flow is being requested by Dashboard Designer.



This 'access control' node, shown above, is a function node with this code:



## Example Flows

Follow these steps to import the example flow code:

1. Copy the code displayed below to your clipboard.
2. Open Visual Flow Creator.
3. Select "Import" and "New flow" from the app menu and paste the code.

Double click the ReadMe nodes for more information.

## Basic Flow Example

```
[
{
  "id": "6dc5e5d4.be494c",
  "type": "comment",
  "z": "392f23c2.fd446c",
  "name": "Read me",
  "info": "Access Control Node:\n    The access control node is to reduce the risks of using a public key.\n    This node checks the referer of the incoming request - if the referer does \n    not match the expected origin (the Dashboard Designer data source) \n    then an error 403 \"forbidden\" will be returned to the requester.\n    \n    \n\nTime Conversion Node:\n    Converts Dashboard Designer timestamps to the format that VFC uses\n    \n\nRead Timeseries Node:\n    Select the variable(s) you wish to manipulate and display. \n    Make sure that the \"Mode\" is set to \"Interval\" and that the interval fields \n    are left blank if you want the time range to match that of the dashboard \n    making the request.\n    \n\n\"Calculation\" Function Node:\n    The Calculation node in this example flow is a Function node and has a very\n    simple calculation - it divides all the values by 2.\n    \n\n\"Dashboard Designer Formatter\" Function Node:\n    This is an example node which provides a method of converting a typical\n    MindSphere response and a typical analytics node response. \n    If custom analytics functions are being implemented then it is likely that\n    this node will need adjustment.",
  "sticky": 0,
  "x": 340,
  "y": 1200,
  "wires": [],
  "_type": "node"
},
{
  "id": "11c225e8.96268a",
  "type": "debug",
  "z": "392f23c2.fd446c",
  "name": "",
  "active": true,
```

```

    "console": "false",
    "xaxis": "_time",
    "complete": "true",
    "x": 1250,
    "y": 1220,
    "wires": []
  },
  {
    "id": "5bc50450.2a2fec",
    "type": "function",
    "z": "392f23c2.fd446c",
    "name": "Access Control",
    "func": "let refererURL = \"https://<tenantname>-dashboarddesigner-dat
asource\";\nif(msg.req.headers.referer == refererURL){\nreturn [msg,null];
\n}else{\n  console.log(\"Forbidden\");\n  msg = {};\n  msg.statusCo
de = 403;\n  \n  return [null,msg];\n}\n",
    "outputs": "2",
    "noerr": 0,
    "x": 340,
    "y": 1260,
    "wires": [
      [
        "e8fdc546.0c8628"
      ],
      [
        "37466b30.e8dfa4"
      ]
    ]
  },
  {
    "id": "e8fdc546.0c8628",
    "type": "function",
    "z": "392f23c2.fd446c",
    "name": "Time Conversion",
    "func": "msg.from = new Date(parseInt(msg.req.query.from));\nmsg.to =
new Date(parseInt(msg.req.query.to));\nreturn msg;",
    "outputs": 1,
    "noerr": 0,
    "x": 540,
    "y": 1240,
    "wires": [
      [
        "285dd047.2206f"
      ]
    ]
  }
}

```

```

    ]
  ]
},
{
  "id": "f09ef087.5c316",
  "type": "function",
  "z": "392f23c2.fd446c",
  "name": "Calculation",
  "func": "for (let i in msg.payload){ \n      for (let key in msg.payload
[i]) {\n          if (key != \"_time\") {\n              msg.payload[i][key] =
(msg.payload[i][key]/2);\n          }\n      }\n\nreturn msg;\"",
  "outputs": 1,
  "noerr": 0,
  "x": 910,
  "y": 1240,
  "wires": [
    [
      "fe227dcc.84658"
    ]
  ]
},
{
  "id": "fe227dcc.84658",
  "type": "function",
  "z": "392f23c2.fd446c",
  "name": "Dashboard_Designer Formatter",
  "func": "let msVariables = {};\n\n// this function makes a number of a
ssumptions about the data coming in and was written for data outputs of th
e VFC analytics nodes\nif(!Array.isArray(msg.payload)){ // this function c
heck to see if the payload is an array, which is required for the Dashboar
d_Designer formatting\n  console.log(\"payload is not the correct forma
t, attempting to readjust\"); \n  if(typeof(msg.payload)!=='object' && m
sg.parameter){\n    msg.payload = { [msg.parameter]: msg.payload };\n
}\n  if(!msg.payload._time){ // assuming there is only one data point in
the payload that triggers this function - typical outputs of analytics nod
es will have one value and no timestamp - this ensures the data will popul
ate the panel in Dashboard_Designer\n    msg.payload._time = msg.to;\n
}\n  msg.payload = [msg.payload]; // once the above checks have been mad
e we must make an array for the next function to execute properly.\n}\nif
(Array.isArray(msg.payload)){\n  for (let i in msg.payload){ \n      f
or (let key in msg.payload[i]) {\n          if (key != \"_time\") {\n
if (!msVariables[key]) {\n              msVariables[key] = { \"datap
oints\": [], \"tag\": {\"name\": key}, \"target\": key }; //our Dashboard_

```



```

Designer object\n
                                }\n
                                msVariables[key].datapoints.push([msg.payload[i][key],new Date(msg.payload[i]._time).getTime()/1000]);\n
                                }\n
                                }\n
                                }\n
                                msg.payload = [];\n
                                for (let i in msVariables) {\n
                                msg.payload.push(msVariables[i]);\n
                                }\n
return msg;\n}
else{\n
console.log(\"Data could not be formatted correctly\");\n}\n",
  "outputs": 1,
  "noerr": 0,
  "x": 1090,
  "y": 1240,
  "wires": [
    [
      "11c225e8.96268a",
      "3c3e8ca3.ac8b14"
    ]
  ]
},
{
  "id": "9368a33c.14e98",
  "type": "http in",
  "z": "392f23c2.fd446c",
  "name": "http in",
  "endpoint": "",
  "method": "get",
  "upload": false,
  "access": "private",
  "key": "",
  "users": "",
  "x": 170,
  "y": 1260,
  "wires": [
    [
      "5bc50450.2a2fec"
    ]
  ]
},
{
  "id": "3c3e8ca3.ac8b14",
  "type": "http response",
  "z": "392f23c2.fd446c",
  "name": "",
  "statusCode": "",
  "headers": {},

```

```
    "x": 1250,
    "y": 1260,
    "wires": []
  },
  {
    "id": "37466b30.e8dfa4",
    "type": "http response",
    "z": "392f23c2.fd446c",
    "name": "",
    "statusCode": "403",
    "headers": {},
    "x": 520,
    "y": 1280,
    "wires": []
  },
  {
    "id": "285dd047.2206f",
    "type": "read timeseries",
    "z": "392f23c2.fd446c",
    "name": "",
    "topic": "",
    "topicLabel": "",
    "assetName": "",
    "period": "60",
    "offset": "0",
    "mode": "interval",
    "from": "",
    "datetimepickerFrom": "",
    "to": "",
    "datetimepickerTo": "",
    "timezoneoffset": 0,
    "x": 720,
    "y": 1240,
    "wires": [
      [
        "f09ef087.5c316"
      ]
    ]
  }
]
```

### Analytics Example: Moving Average

```

[
{
  "id": "693080.833c5f8",
  "type": "comment",
  "z": "392f23c2.fd446c",
  "name": "Read me",
  "info": "Access Control Node:\n    The access control node is to reduce the risks of using a public key.\n    This node checks the referer of the incoming request - if the referer does \n    not match the expected origin (the Dashboard_Designer data source) \n    then an error 403 \"forbidden\" will be returned to the requester.\n\n\nTime Conversion Node:\n    Converts Dashboard_Designer timestamps to the format that VFC uses \n\nRead Timeseries Node:\n    Select the variable(s) you wish to manipulate and display. \n    Make sure that the \"Mode\" is set to \"Interval\" and that the interval fields \n    are left blank if you want the time range to match that of the dashboard \n    making the request.\n\n\"Dashboard_Designer Formatter\" Function Node:\n    This is an example node which provides a method of converting a typical\n    MindSphere response and a typical analytics node response. \n    If custom analytics functions are being implemented then it is likely that\n    this node will need adjustment.",
  "sticky": 0,
  "x": 320,
  "y": 1440,
  "wires": [],
  "_type": "node"
},
{
  "id": "4dea3605.6899e8",
  "type": "debug",
  "z": "392f23c2.fd446c",
  "name": "",
  "active": true,
  "console": "false",
  "xaxis": "_time",
  "complete": "true",
  "x": 1230,
  "y": 1460,
  "wires": []
},
{
  "id": "2ec0625a.b5ad7e",
  "type": "function",

```

```

    "z": "392f23c2.fd446c",
    "name": "Access Control",
    "func": "let refererURL = \"https:// Dashboard_Designer-datasource-sil
opsms.apps.eu1.mindsphere.io\";\nif(msg.req.headers.referer == refererURL)
{\nreturn [msg,null];  \n}else{\n  console.log(\"Forbidden\");\n  ms
g = {};\n  msg.statusCode = 403;\n  \n  return [null,msg];\n}\n",
    "outputs": "2",
    "noerr": 0,
    "x": 320,
    "y": 1500,
    "wires": [
      [
        "aee3ad9b.e4ad"
      ],
      [
        "62e6b5bb.c58bcc"
      ]
    ]
  },
  {
    "id": "aee3ad9b.e4ad",
    "type": "function",
    "z": "392f23c2.fd446c",
    "name": "Time Conversion",
    "func": "msg.from = new Date(parseInt(msg.req.query.from));\nmsg.to =
new Date(parseInt(msg.req.query.to));\nreturn msg;",
    "outputs": 1,
    "noerr": 0,
    "x": 520,
    "y": 1480,
    "wires": [
      [
        "8ab9a317.bcd2a"
      ]
    ]
  },
  {
    "id": "f0f3fae1.aff638",
    "type": "function",
    "z": "392f23c2.fd446c",
    "name": "Dashboard_Designer Formatter",
    "func": "let msVariables = {};\n\n// this function makes a number of a
ssumptions about the data coming in and was written for data outputs of th

```

```

e VFC analytics nodes\nif(!Array.isArray(msg.payload)){ // this function c
heck to see if the payload is an array, which is required for the Dashboar
d_Designer formatting\n    console.log(\"payload is not the correct forma
t, attempting to readjust\"); \n    if(typeof(msg.payload)!=='object' && m
sg.parameter){\n        msg.payload = { [msg.parameter]: msg.payload }; \n
}\n    if(!msg.payload._time){ // assuming there is only one data point in
the payload that triggers this function - typical outputs of analytics nod
es will have one value and no timestamp - this ensures the data will popul
ate the panel in Dashboard_Designer\n        msg.payload._time = msg.to;\n
}\n    msg.payload = [msg.payload]; // once the above checks have been mad
e we must make an array for the next function to execute properly.\n}\nif
(Array.isArray(msg.payload)){\n    for (let i in msg.payload){ \n        f
or (let key in msg.payload[i]) {\n            if (key != \"_time\") {\n
if (!msVariables[key]) {\n                msVariables[key] = { \"datap
oints\": [], \"tag\": {\"name\": key}, \"target\": key }; //our Dashboard_
Designer object\n            }\n            msVariables[key].datap
oints.push([msg.payload[i][key],new Date(msg.payload[i]._time).getTime()/1
000]);\n        }\n    }\n    }\n    msg.payload = [];\n    for (l
et i in msVariables) {\n        msg.payload.push(msVariables[i]);\n    }\n
return msg;\n}else{\n    console.log(\"Data could not be formatted correct
ly\");\n}\n",
    "outputs": 1,
    "noerr": 0,
    "x": 1070,
    "y": 1480,
    "wires": [
        [
            "4dea3605.6899e8",
            "ee0fa9f.f331758"
        ]
    ]
},
{
    "id": "b19229df.5310f8",
    "type": "http in",
    "z": "392f23c2.fd446c",
    "name": "",
    "endpoint": "",
    "method": "get",
    "upload": false,
    "access": "private",
    "key": "",
    "users": "",

```

```
    "x": 170,  
    "y": 1500,  
    "wires": [  
      [  
        "2ec0625a.b5ad7e"  
      ]  
    ]  
  },  
  {  
    "id": "ee0fa9f.f331758",  
    "type": "http response",  
    "z": "392f23c2.fd446c",  
    "name": "",  
    "statusCode": "",  
    "headers": {},  
    "x": 1230,  
    "y": 1500,  
    "wires": []  
  },  
  {  
    "id": "62e6b5bb.c58bcc",  
    "type": "http response",  
    "z": "392f23c2.fd446c",  
    "name": "",  
    "statusCode": "403",  
    "headers": {},  
    "x": 500,  
    "y": 1520,  
    "wires": []  
  },  
  {  
    "id": "63fcd020.91557",  
    "type": "moving average",  
    "z": "392f23c2.fd446c",  
    "name": "",  
    "parameter": "",  
    "parameterout": "",  
    "windowSize": 3,  
    "algorithm": "simple",  
    "alpha": 0.5,  
    "x": 880,  
    "y": 1480,  
    "wires": [  
      [  
        "2ec0625a.b5ad7e"  
      ]  
    ]  
  }  
]
```

```

        [
            "f0f3fae1.aff638"
        ]
    ],
    {
        "id": "8ab9a317.bcd2a",
        "type": "read timeseries",
        "z": "392f23c2.fd446c",
        "name": "",
        "topic": "",
        "topicLabel": "",
        "assetName": "",
        "period": "60",
        "offset": "0",
        "mode": "interval",
        "from": "",
        "datetimepickerFrom": "",
        "to": "",
        "datetimepickerTo": "",
        "timezoneoffset": 0,
        "x": 700,
        "y": 1480,
        "wires": [
            [
                "63fcd020.91557"
            ]
        ]
    }
]

```

### Access Query Parameters Sent to VFC

```

[
{
    "id": "3703d4c5.4b51ac",
    "type": "comment",
    "z": "392f23c2.fd446c",
    "name": "Read me",
    "info": "This flow provides an example of how the query parameters can
be accessed from within a Visual Flow Creator flow.",
    "sticky": 0,
    "x": 540,
    "y": 1600,

```

```

    "wires": [],
    "_type": "node"
  },
  {
    "id": "1edecaa1.50b9b5",
    "type": "function",
    "z": "392f23c2.fd446c",
    "name": "Access Control",
    "func": "let refererURL = \"https:// Dashboard_Designer-datasource-sil
opsms.apps.eu1.mindsphere.io\";\nif(msg.req.headers.referer == refererURL)
{\nreturn [msg,null];  \n}else{\n  console.log(\"Forbidden\");\n  ms
g = {};\n  msg.statusCode = 403;\n  \n  return [null,msg];\n}\n",
    "outputs": "2",
    "noerr": 0,
    "x": 551.5,
    "y": 1661,
    "wires": [
      [
        "3f44162d.f2f52a"
      ],
      [
        "fcfa3a54.111b18"
      ]
    ]
  },
  {
    "id": "3f44162d.f2f52a",
    "type": "function",
    "z": "392f23c2.fd446c",
    "name": "debug query param",
    "func": "console.log(msg.req.query.exampleQueryParam);\n\nreturn ms
g;",
    "outputs": 1,
    "noerr": 0,
    "x": 761.5,
    "y": 1641,
    "wires": [
      []
    ]
  },
  {
    "id": "e9b94d90.5d4a6",
    "type": "http in",

```



```

    "z": "392f23c2.fd446c",
    "name": "",
    "endpoint": "",
    "method": "get",
    "upload": false,
    "access": "private",
    "key": "",
    "users": "",
    "x": 381.5,
    "y": 1661,
    "wires": [
      [
        "1edecaa1.50b9b5"
      ]
    ]
  },
  {
    "id": "fcfa3a54.111b18",
    "type": "http response",
    "z": "392f23c2.fd446c",
    "name": "",
    "statusCode": "403",
    "headers": {},
    "x": 731.5,
    "y": 1681,
    "wires": []
  }
]

```

## string\_table

Note: Replacement strings do not support spaces.

## VFCrequest

Creates a request to the http in blocks in Visual Flow Creator (VFC). Data can be returned from VFC to Dashboard Designer for visualization. This requires a subscription and access to Visual Flow Creator.

Important Points About VFC Requests:

- Only the GET method is currently supported and is selected by default.
- A panel that uses VFCrequest cannot have additional queries in the same panel.

- No asset query is required for this function, as the data is retrieved via Visual Flow Creator.
- The method parameter defines the method of the call.

The format of the endpoint parameter requires an endpoint which follows this format:  
`/public/<tenant>/<route>?key=<secret>`.

Syntax: `VFCrequest(method, endpoint, queryParams)`

### queryParams

Additional custom parameters that can be sent to VFC as a part of the request; this creates a compatible flow in Visual Flow Creator when you set up the http in node in Visual Flow Creator.

### httpin\_node

Double click to open the settings and enter the endpoint name that you wish to use.

### httpin\_endpoint

Changes the Access method to Public access using keys.

### httpin\_generate

Save this flow and copy the link address of the http in node.

### httpin\_key

The URL used by the VFCrequest function as the endpoint parameter. If you copied the URL manually, be sure to remove the host URL preceding `/public`. For Visual Flow Creator to respond, the flow should have an `http` out at the end as seen in the example flows below.

Once the endpoint is created, the flow will run every time the panel refreshes, as long as it is formatted correctly. Dashboard Designer can only display the data configured with the correct JSON structure, as shown in the example below.

## Security Information

Creating an http in node with a public access key means that the particular flow triggered by the http in node can potentially be accessed externally. Thus, it is important to protect your data and prevent unauthorized access. Please handle the access keys with care.

It is recommended to create an "access control" node after the http in node so that the flow can determine if the flow is being requested by Dashboard Designer.

The "access control" node shown above is a function node with the code:

Importing an Example Flow

To import an example flow from below, copy the code and select Import > new flow from the app menu in Visual Flow Creator. Double-click the Read me nodes for more information.

## Example: Access Query Parameters Sent to VFC

```
[
{
  "id": "3703d4c5.4b51ac",
  "type": "comment",
  "z": "392f23c2.fd446c",
  "name": "Read me",
  "info": "This flow provides an example of how the query parameters can
be accessed from within a Visual Flow Creator flow.",
  "sticky": 0,
  "x": 540,
  "y": 1600,
  "wires": [],
  "_type": "node"
},
{
  "id": "1edecaa1.50b9b5",
  "type": "function",
  "z": "392f23c2.fd446c",
  "name": "Access Control",
  "func": "let refererURL = \"https://Dashboard_Designer-datasource-silo
psms.apps.eu1.mindsphere.io\";\nif(msg.req.headers.referer == refererURL)
{\nreturn [msg,null];  \n}else{\n  console.log(\"Forbidden\");\n  ms
g = {};\n  msg.statusCode = 403;\n  \n  return [null,msg];\n}\n",
  "outputs": "2",
  "noerr": 0,
  "x": 551.5,
  "y": 1661,
  "wires": [
    [
      "3f44162d.f2f52a"
    ],
    [
      "fcfa3a54.111b18"
    ]
  ]
},
{
  "id": "3f44162d.f2f52a",
  "type": "function",
  "z": "392f23c2.fd446c",
  "name": "debug query param",
```

```
    "func": "console.log(msg.req.query.exampleQueryParam);\n\nreturn ms
g;",
    "outputs": 1,
    "noerr": 0,
    "x": 761.5,
    "y": 1641,
    "wires": [
      []
    ]
  },
  {
    "id": "e9b94d90.5d4a6",
    "type": "http in",
    "z": "392f23c2.fd446c",
    "name": "",
    "endpoint": "",
    "method": "get",
    "upload": false,
    "access": "private",
    "key": "",
    "users": "",
    "x": 381.5,
    "y": 1661,
    "wires": [
      [
        "1edecaa1.50b9b5"
      ]
    ]
  },
  {
    "id": "fcfa3a54.111b18",
    "type": "http response",
    "z": "392f23c2.fd446c",
    "name": "",
    "statusCode": "403",
    "headers": {},
    "x": 731.5,
    "y": 1681,
    "wires": []
  }
]
```

## 7.1 Visualizations

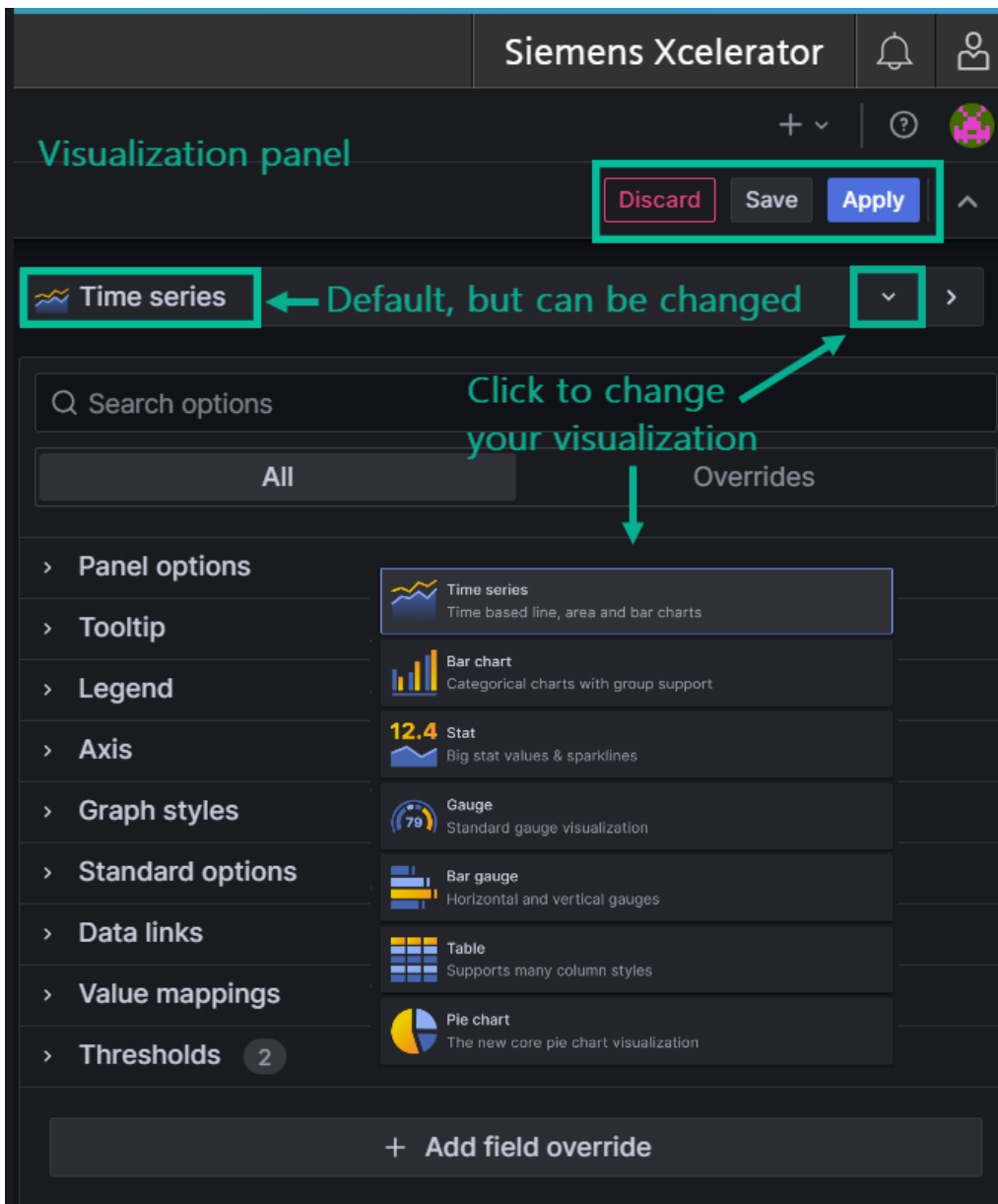
Visualizations play a pivotal role in understanding and interpreting data, providing for comprehensive analysis and decision-making. This section provides an overview of the various visualization panels available within Dashboard Designer, empowering users to explore, configure, and optimize visual data analysis.



While this topic covers all Dashboard Designer Add-On visualization panels, please note that there are various packages and not all visualization panels are supported in all subscriptions.

### Some Navigation Pointers for the Visualization Panel

This image illustrates the visualization panel with the sections collapsed. The multiple sections and settings allow you to tailor the appearance and behavior of the visual elements to suit your requirements. The image also shows a miniature version of the drop-down list that makes it easy to switch visualizations.

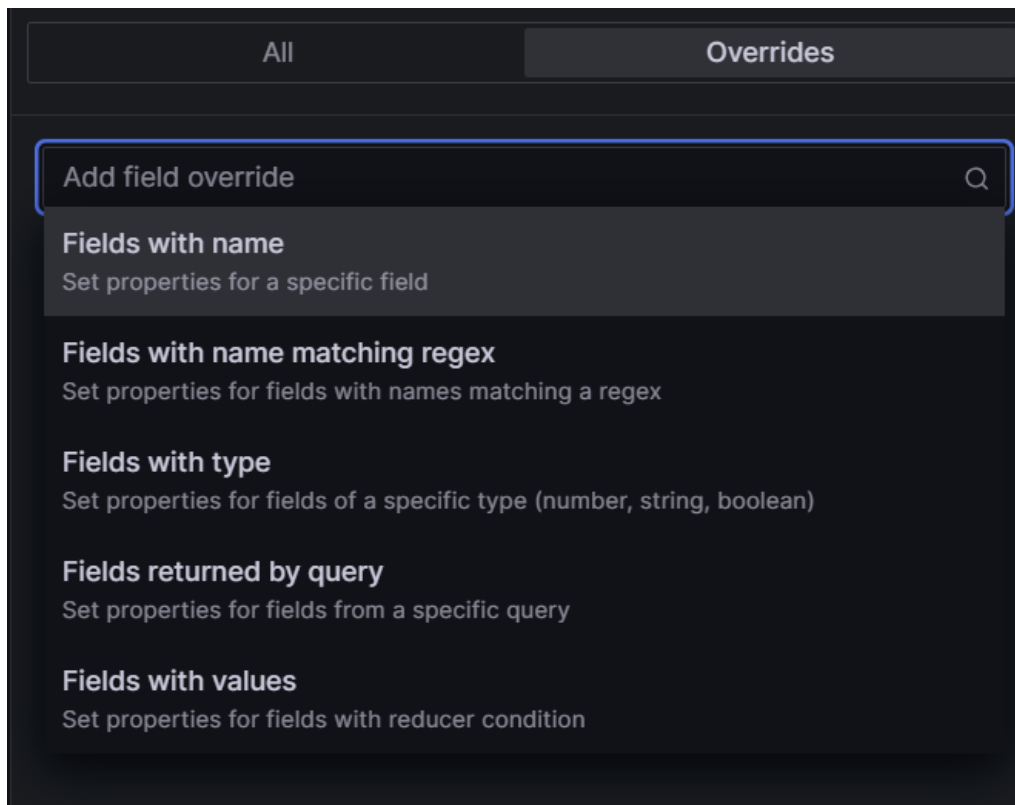


## Visualization Panel Overview

The Visualization Panel serves as a hub for configuring and customizing different visualization options. Each section holds specific controls with default settings that are automatically applied unless you opt to make changes to fine-tune your visualization.

### About the Overrides Tab

You can override settings for any visualization field. Here is an example that illustrates categories of fields you can search by to locate the field you want to override:



## How to Configure Visualization Options

The following section guides you through configuring each visualization area. The number of settings you configure in a section displays next to its heading label.

### Setting Panel Options

The first section, *Panel options* is where you define the panel title, description, transparency, and include links, among other preferences.

The *repeat* feature allows Dashboard Designer to repeatedly generate a separate panel for each variable contained in an asset or aspect.

Follow these steps to configure the panel options:

1. Enter a title for your panel. Optionally enter a description.
2. If you want the background of the panel to be transparent, set the toggle to "on".
3. Click the '+ Add link' button to enter a link title, the URL, specify whether or not to open the link in a new tab, and save.

Panel options

Title

Fixed Selection

Description

Transparent background

Panel links

+ Add link

Repeat options

Repeat by variable

Repeat this panel for each value in the selected variable. This is not visible while in edit mode. You need to go back to dashboard and then update the variable or reload the dashboard.

Choose

## Tooltips

Configure visibility, sort order, and other tooltip-related settings.

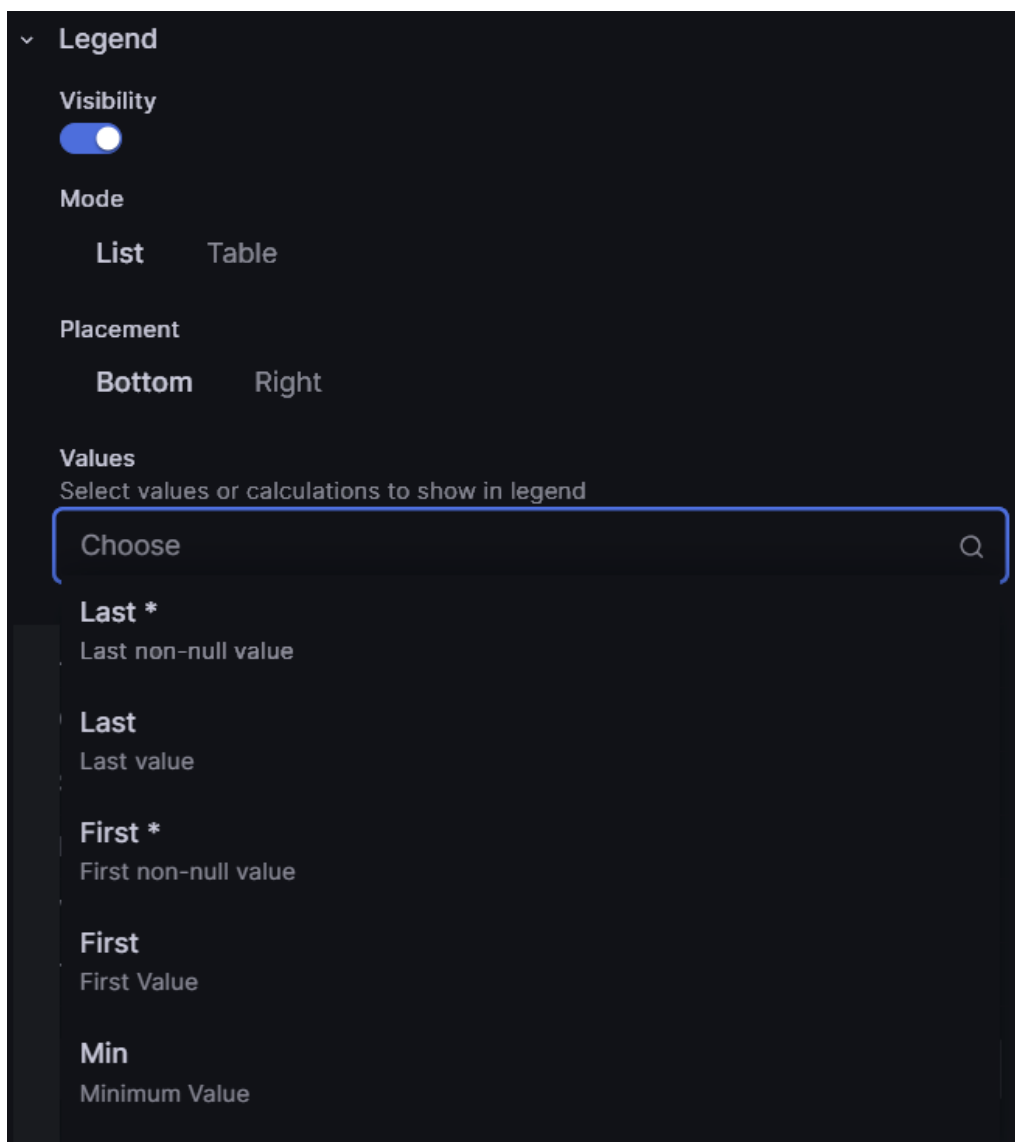
## Legends

Tailor the display and positioning of the legend, choose list or table format, and select specific values to appear in the legend.

Follow these steps to set legend options:

1. Toggle the visibility to 'on' to display the legend on your visualization.
2. Set whether to display the legend as a list or table and whether or not to place it at the bottom or right of the visualization.
3. If you want to display values in the legend, select the values you want to show from the drop-down list. Your selections display in the Values field as you choose them.





## Axes

Adjust visual aspects such as placement, color, grid lines, and more, related to the axes. Here is an example of the axis settings:

▼ **Axis**

**Time zone**

Default ▼ +

**Placement**

Auto Left Right Hidden

**Label**

Optional text

**Width**

Auto

**Show grid lines**

Auto On Off

**Color**

Text Series

**Scale**

Linear Logarithmic Symlog

**Centered zero**

**Soft min**

See: Standard options > Min

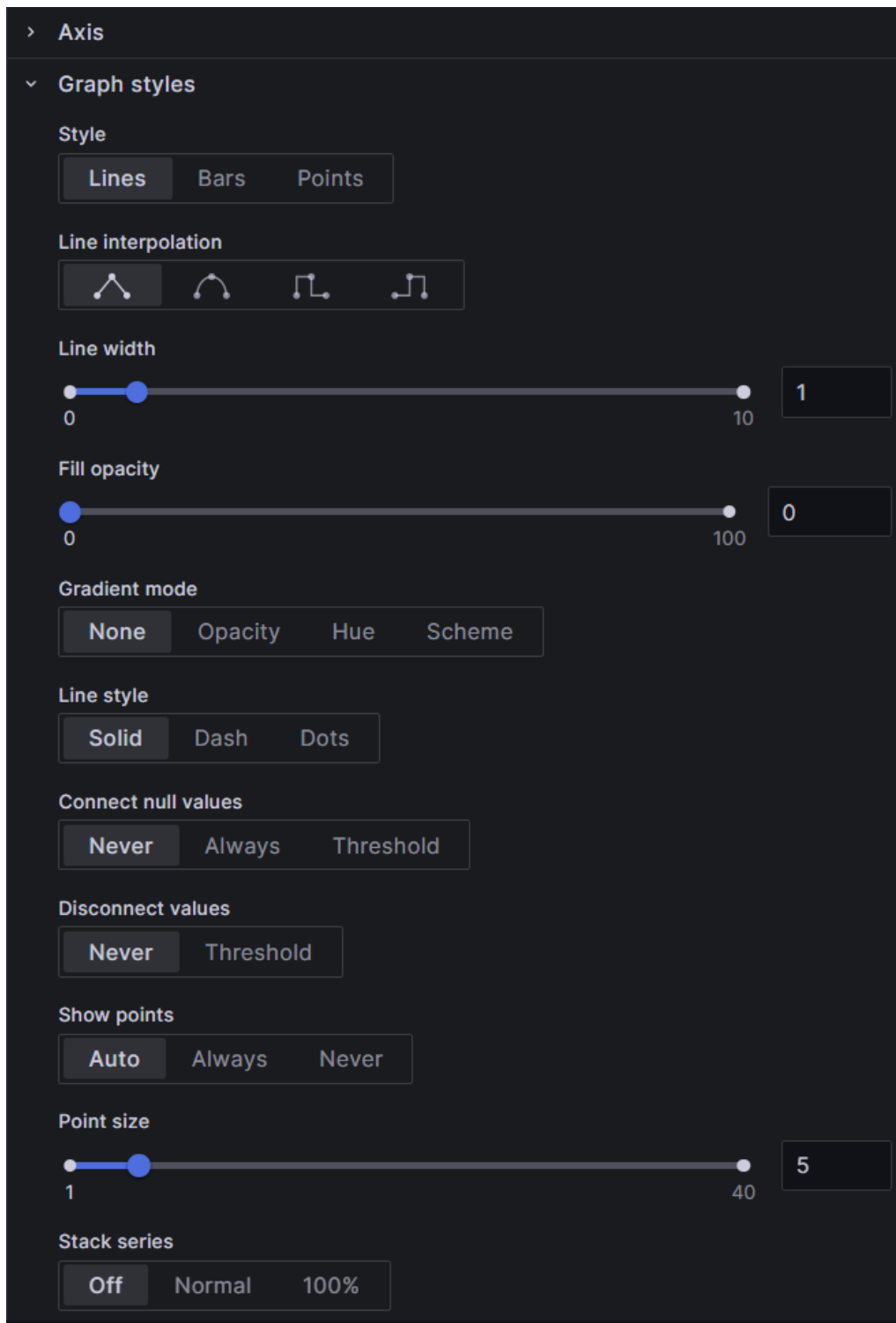
**Soft max**

See: Standard options > Max

## Graphs

Modify graph styles, including lines, bars, points, line characteristics, color, and additional visual options, as well as choosing whether or not to show various visual options for your graph.

Here is an example of the graph style settings:



## Standard Options

Customize various fields affecting the panel's appearance, such as units, color palette, min-max values, decimals, etc. Most of the fields in the standard options are discretionary and determine how your panel appears. The first selection, for example, is for the unit in which graph or chart data will be rendered, followed by the measurement standard, such as degrees, arc minutes, etc. for the selected unit. Other standard selections include color palette, min, max, decimals, and so on.

Follow these steps to set standard options:

1. Select a unit from the drop-down list. A list displays various measurement standards for the unit.
2. Select the measurement to use.
3. Continue to accept the defaults or make selections for the rest of the fields for your visualization.

Here is an example of the standard options:

Standard options

Unit  
Choose

Min  
Leave empty to calculate based on all values  
auto

Max  
Leave empty to calculate based on all values  
auto

Decimals  
auto

Display name  
Change the field or series name  
none

Color scheme  
Classic palette

No value  
What to show when there is no value  
-

## Data Links

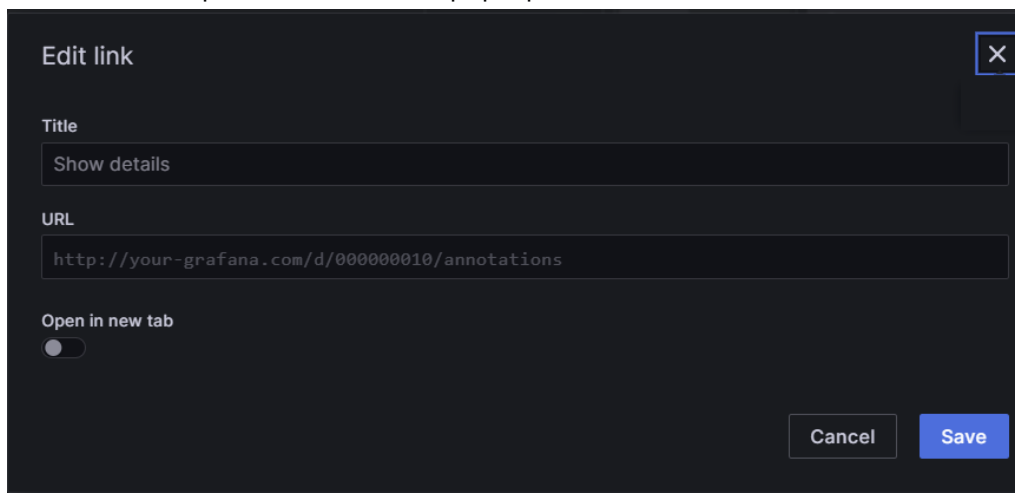
Add and configure links to external resources relevant to the visualization.

Follow these steps to set data links:

1. Click the '+Add link' button. The 'Edit link' pop-up window displays.
2. Enter a title that will display on your panel.
3. Enter the URL for the link - according to the example in the URL field.
4. If you want the link to open in a new tab, move the toggle to 'on'.

5. Click the 'Save' button.

Here is an example of the data links pop-up window:



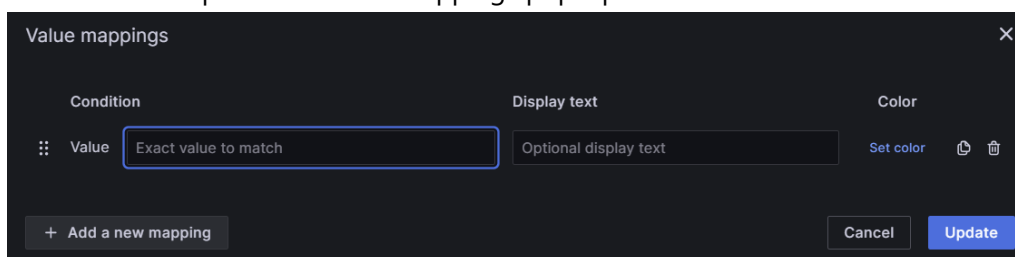
## Value Mappings

Assign specific colors and optional text to render values in the data set.

Follow these steps to set value mappings:

1. Click the 'Add value mapping' button. The 'Value mapping' pop-up window displays.
2. Enter a specific value in the 'Condition' field.
3. If you want the value to display in color, click the 'Set color' link. A color palette displays.
4. Select a color.
5. Continue to add additional mappings by clicking the '+ Add a new mapping' button.
6. Click the 'Update' button.

Here is an example of the value mappings pop-up window:



## Thresholds

Set thresholds for values and define their display preferences, including absolute or percentage-based and visualization options.

Follow these steps to set threshold options:

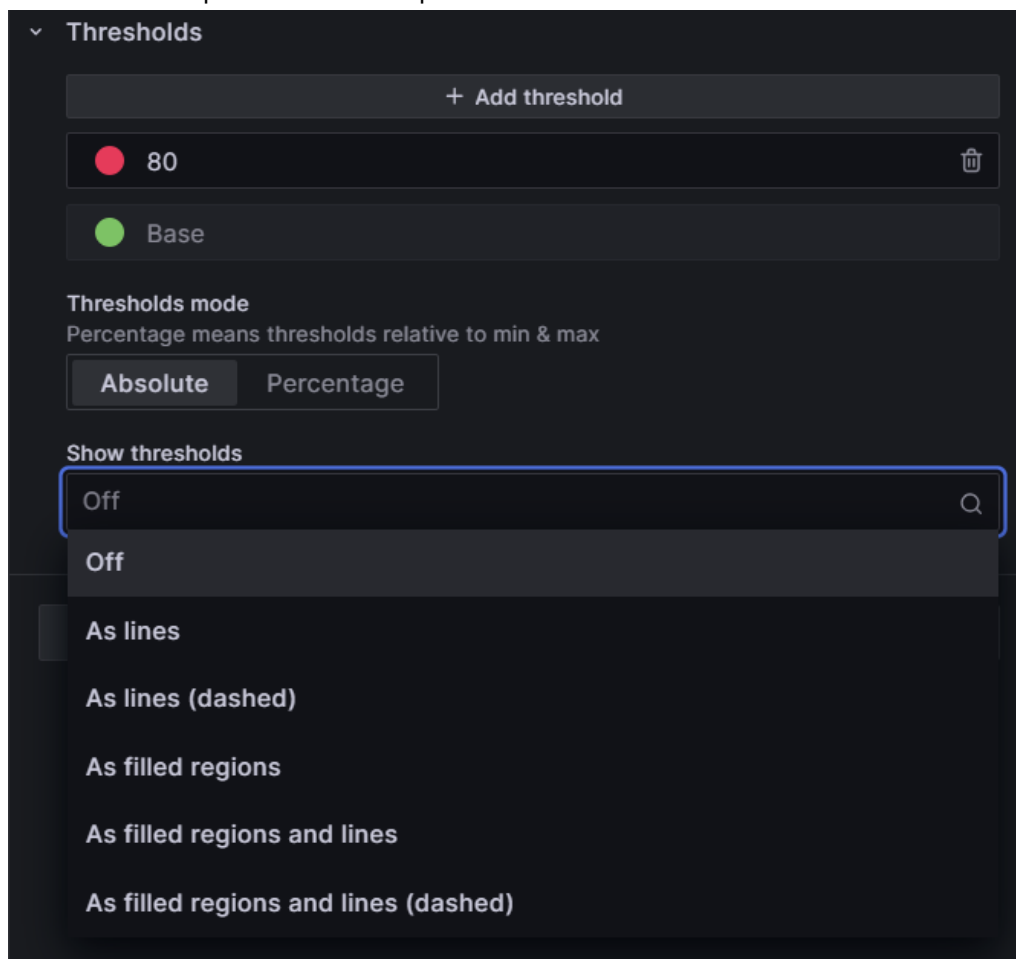
1. Click the '+Add threshold' button. A threshold row appears at the top of the thresholds list with a default value and color.

2. Enter a number to replace the default value, or use the up/down arrows.
3. Click the default color dot and select a color from the color palette.
4. Select the delete icon to delete any threshold rows you don't want.
5. Leave the 'Threshold mode' on 'Absolute', unless your threshold values are relative to min and max values; if so, slide the toggle to 'Percentage'.
6. Select how you want the thresholds to display from the 'Show thresholds' drop-down list. (See image below).



The 'Base' threshold row cannot be deleted.

Here is an example of threshold options:



## Field Overrides

Specify overrides for specific fields or field types, allowing for fine-tuning according to the data properties and queries.



The following step-by-step uses 'Fields with type', then 'Number'. The steps are the same for any of the 'field types' you select. The subsequent drop-down list displays properties relevant to the field type selected. All field options that display for this feature are, of course, relative to the query data.

Follow these steps to configure the field override option:

1. Click the '+Add field override' button. A drop-down of types displays. (Image below.)
2. Select 'Fields with type'. The drop-down list closes and a 'Fields with type' label displays over a drop-down list of field types.
3. Select 'Number'.
4. Click the '+ Add override property' button. (Image below.)
5. Select an option from the 'Add override property' drop-down list.
6. Select and option from the additional values that display.

These images show examples of the field override options:

