Analytics Dashboard JSON Developer Guide

Salesforce, Spring '25

Last updated: November 14, 2024

CONTENTS

CRM ANALYTICS DASHBOARD JSON OVERVIEW

The easiest way to build a dashboard in CRM Analytics is to use the dashboard designer. However, if needed, you can further customize dashboards by editing their JSON files. The JSON defines the components of the dashboard and how they interact.

Modify a dashboard's JSON file to perform advanced customization tasks that can't be accomplished in the designer's user interface, like:

- Manually set up bindings to override the default faceting behavior and specify the relationships between the steps that aren't bound by default.
- Set query limits.
- Specify columns for a values table.

VIEW OR MODIFY A DASHBOARD JSON FILE

Use the JSON Editor to modify the JSON for a dashboard or lens.

JSON Editor displays the JSON of a lens or dashboard and lets you quickly see the effect of your edits in the running asset.

1. To access JSON Editor, open the lens or dashboard you want to edit, and press CTRL+E for PCs or CMD+E for Macs.
2. Modify the JSON in the editor. You can use standard keyboard shortcuts for editing functions and search.
3. To go back to the user interface and see how edits to the JSON appear in the lens or dashboard, click **Done**.
4. To retain your edits, save the lens or dashboard. Changes made in the JSON editor are not saved until you explicitly save the lens or dashboard.

In JSON Editor, the following shortcuts let you perform basic actions from your keyboard.

JSON Editor Keyboard Shortcut	Description
CRTL+3 (Windows); CMD+3 (Mac)	Disregard changes and load the original JSON
CRTL+X (Windows); CMD+X (Mac)	Cut
CRTL+C (Windows); CMD+C (Mac)	Copy
CRTL+V (Windows); CMD+V (Mac)	Paste
CRTL+Z (Windows); CMD+Z (Mac)	Undo
SHIFT+CRTL+Z (Windows); SHIFT+CMD+Z (Mac)	Redo
CRTL+F (Windows); CMD+F (Mac)	Search (RegExp, case-sensitive, or whole word searches available)
CRTL+E (Windows); CMD+E (Mac)	View dashboard with changes to JSON

EDITIONS

Available in Salesforce Classic and Lightning Experience.

Available with CRM Analytics, which is available for an extra cost in **Enterprise**, **Performance**, and **Unlimited** Editions. Also available in **Developer** Edition.

USER PERMISSIONS

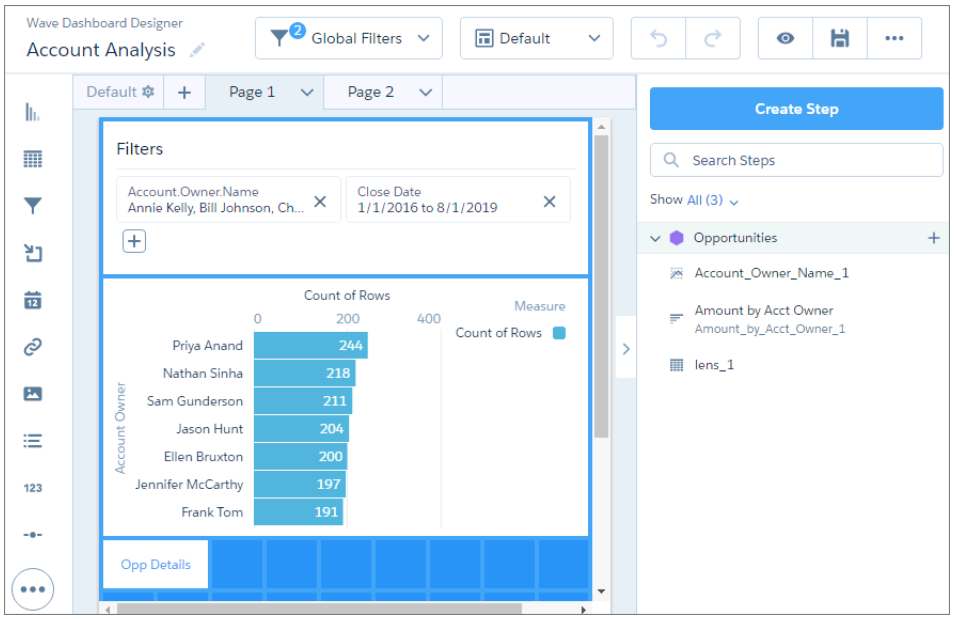
To modify the JSON file that defines a dashboard:

- Create and Edit CRM Analytics Dashboards

CRM ANALYTICS DASHBOARD JSON EXAMPLE

The JSON for each CRM Analytics dashboard contains multiple levels of properties. Review the sample JSON provided in this section to learn about the basic JSON structure of a dashboard.

Example: The sample JSON defines this dashboard.



The Account Analysis dashboard consists of the following objects.

- Layouts:
  - Default, which has pages:
    - Page 1
    - Page 2
  - Mobile
- Widgets:
  - Values table `table_1` based on step `lens_1` (Not shown in the screenshot because it exists only on Page 2.)
  - Global filter panel `filterpanel_1`
  - Link `link_1`
  - Horizontal Bar Chart `chart_1` based on step `Amount_by_Acct_Owner_1`
- Steps:
  - `Amount_by_Acct_Owner_1`
  - `Account_Owner_Name_1`
  - `lens_1`

- Dataset `0FbB000000017QxKAI`

```
{
    "label": "Account Analysis",
    "mobileDisabled": false,
    "state": {
        "dataSourceLinks": [],
        "filters": [
            {
                "dataset": {
                    "id": "0FbB000000017QxKAI",
                    "name": "opportunity",
                    "url": "/services/data/v41.0/wave/datasets/0FbB000000017QxKAI"
                },
                "fields": [
                    "Account.Owner.Name"
                ],
                "label": "Account.Owner.Name",
                "locked": false,
                "operator": "in",
                "value": [
                    "Annie Kelly",
                    "Bill Johnson",
                    "Chan Lao",
                    "Ellen Bruxton",
                    "Erin Baker",
                    "Frank Tom",
                    "Jason Hunt",
                    "Jennifer McCarthy",
                    "Leslie Pham",
                    "Michelle Nguyen",
                    "Nathan Sinha",
                    "Norman Truman",
                    "Priya Anand",
                    "Rob Wilkens",
                    "Sam Gunderson",
                    "Valerie Thom",
                    "Walker Chan",
                    "Wang Lee"
                ]
            },
            {
                "dataset": {
                    "id": "0FbB000000017QxKAI",
                    "name": "opportunity",
                    "url": "/services/data/v41.0/wave/datasets/0FbB000000017QxKAI"
                },
                "fields": [
                    "Close Date"
                ],
                "label": "Close Date",
                "locked": false,
                "operator": ">=<=",
                "value": [
                    [
```

```
                1451635200000,
                1564642800000
            ]
        ]
    }
],
"gridLayouts": [
    {
        "name": "Default",
        "numColumns": 9,
        "pages": [
            {
                "label": "Page 1",
                "name": "41f040d6-48a6-4dc7-9166-eb985df7e9d8",
                "widgets": [
                    {
                        "colspan": 9,
                        "column": 0,
                        "name": "filterpanel_1",
                        "row": 0,
                        "rowspan": 3,
                        "widgetStyle": {
                            "borderEdges": []
                        }
                    },
                    {
                        "colspan": 9,
                        "column": 0,
                        "name": "chart_1",
                        "row": 3,
                        "rowspan": 5,
                        "widgetStyle": {
                            "borderEdges": []
                        }
                    },
                    {
                        "colspan": 2,
                        "column": 0,
                        "name": "link_1",
                        "row": 8,
                        "rowspan": 1,
                        "widgetStyle": {
                            "borderEdges": []
                        }
                    }
                ]
            },
            {
                "label": "Page 2",
                "name": "ac750443-1729-47fb-8d60-a35703076bf7",
                "widgets": [
                    {
                        "colspan": 9,
                        "column": 0,
```

```
                        "name": "filterpanel_1",
                        "row": 0,
                        "rowspan": 3,
                        "widgetStyle": {
                            "borderEdges": []
                        }
                    },
                    {
                        "colspan": 9,
                        "column": 0,
                        "name": "table_1",
                        "row": 3,
                        "rowspan": 7,
                        "widgetStyle": {
                            "borderEdges": []
                        }
                    }
                ]
            }
        ],
        "rowHeight": "normal",
        "selectors": [],
        "style": {
            "alignmentX": "left",
            "alignmentY": "top",
            "backgroundColor": "#44A2F5",
            "cellSpacingX": 4,
            "cellSpacingY": 4,
            "fit": "original",
            "gutterColor": "#A3B8CC"
        },
        "version": 1
    },
    {
        "name": "Mobile",
        "numColumns": 2,
        "pages": [
            {
                "label": "Page 1",
                "name": "41f040d6-48a6-4dc7-9166-eb985df7e9d8",
                "widgets": [
                    {
                        "colspan": 2,
                        "column": 0,
                        "name": "filterpanel_1",
                        "row": 0,
                        "rowspan": 2,
                        "widgetStyle": {
                            "borderEdges": []
                        }
                    },
                    {
                        "colspan": 2,
                        "column": 0,
```

```
                            "name": "chart_1",
                            "row": 2,
                            "rowspan": 10,
                            "widgetStyle": {
                                "borderEdges": []
                            }
                        }
                    ]
                }
            ],
            "rowHeight": "normal",
            "selectors": [
                "maxWidth(599)"
            ],
            "style": {
                "alignmentX": "left",
                "alignmentY": "top",
                "backgroundColor": "#2EC2BA",
                "cellSpacingX": 4,
                "cellSpacingY": 4,
                "fit": "original",
                "gutterColor": "#091A3E"
            },
            "version": 1
        }
    ],
    "layouts": [],
    "steps": {
        "Amount_by_Acct_Owner_1": {
            "broadcastFacet": true,
            "datasets": [
                {
                    "id": "0FbB0000000017QxKAI",
                    "label": "Opportunities",
                    "name": "opportunity",
                   "url": "/services/data/v41.0/wave/datasets/0FbB0000000017QxKAI"

                }
            ],
            "isGlobal": false,
            "label": "Amount by Acct Owner",
            "query": {
                "measures": [
                    [
                        "count",
                        "*"
                    ]
                ],
                "groups": [
                    "Account.Owner.Name"
                ],
                "order": [
                    [
                        "count",
```

```
            {
                "ascending": false
            }
        ]
    ]
},
"receiveFacet": true,
"selectMode": "multi",
"type": "aggregateflex",
"useGlobal": true,
"visualizationParameters": {
    "options": {},
    "parameters": {
        "autoFitMode": "none",
        "showValues": true,
        "bins": {
            "breakpoints": {
                "high": 100,
                "low": 0
            },
            "bands": {
                "high": {
                    "color": "#008000",
                    "label": ""
                },
                "low": {
                    "color": "#B22222",
                    "label": ""
                },
                "medium": {
                    "color": "#ffa500",
                    "label": ""
                }
            }
        },
        "legend": {
            "showHeader": true,
            "show": true,
            "customSize": "auto",
            "position": "right-top",
            "inside": false
        },
        "axisMode": "multi",
        "tooltip": {
            "showBinLabel": true,
            "measures": "",
            "showPercentage": false,
            "showDimensions": true,
            "showMeasures": true,
            "customizeTooltip": false,
            "dimensions": ""
        },
        "visualizationType": "hbar",
        "title": {
```

```
                    "fontSize": 14,
                    "subtitleFontSize": 11,
                    "label": "",
                    "align": "center",
                    "subtitleLabel": ""
                },
                "binValues": false,
                "trellis": {
                    "flipLabels": false,
                    "showGridLines": true,
                    "size": [
                        100,
                        100
                    ],
                    "enable": false,
                    "type": "x",
                    "chartsPerLine": 4
                },
                "measureAxis2": {
                    "sqrtScale": false,
                    "showTitle": true,
                    "showAxis": true,
                    "title": "",
                    "customDomain": {
                        "showDomain": false
                    }
                },
                "measureAxis1": {
                    "sqrtScale": false,
                    "showTitle": true,
                    "showAxis": true,
                    "title": "",
                    "customDomain": {
                        "showDomain": false
                    }
                },
                "theme": "wave",
                "dimensionAxis": {
                    "showTitle": true,
                    "customSize": "auto",
                    "showAxis": true,
                    "title": "",
                    "icons": {
                        "useIcons": false,
                        "iconProps": {
                            "fit": "cover",
                            "column": "",
                            "type": "round"
                        }
                    }
                }
            },
            "type": "chart",
            "visualizationType": "hbar"
```

```
        }
    },
    "Account_Owner_Name_1": {
        "broadcastFacet": true,
        "datasets": [
            {
                "id": "0FbB0000000017QxKAI",
                "label": "Opportunities",
                "name": "opportunity",
               "url": "/services/data/v41.0/wave/datasets/0FbB0000000017QxKAI"

            }
        ],
        "isGlobal": false,
        "query": {
            "measures": [
                [
                    "count",
                    "*"
                ],
                [
                    "avg",
                    "Amount"
                ]
            ],
            "groups": [
                "Account.Owner.Name"
            ],
            "order": [
                [
                    "count",
                    {
                        "ascending": false
                    }
                ]
            ]
        },
        "receiveFacet": true,
        "selectMode": "single",
        "type": "aggregateflex",
        "useGlobal": true,
        "visualizationParameters": {
            "parameters": {
                "autoFitMode": "none",
                "showPoints": false,
                "legend": {
                    "showHeader": true,
                    "show": true,
                    "customSize": "auto",
                    "position": "right-top",
                    "inside": false
                },
                "axisMode": "multi",
                "tooltip": {
```

```
        "showBinLabel": true,
        "measures": "",
        "showPercentage": false,
        "showDimensions": true,
        "showMeasures": true,
        "customizeTooltip": false,
        "dimensions": ""
    },
    "visualizationType": "line",
    "dashLine": {
        "measures": "",
        "showDashLine": false
    },
    "title": {
        "fontSize": 14,
        "subtitleFontSize": 11,
        "label": "",
        "align": "center",
        "subtitleLabel": ""
    },
    "trellis": {
        "flipLabels": false,
        "showGridLines": true,
        "size": [
            100,
            100
        ],
        "enable": false,
        "type": "x",
        "chartsPerLine": 4
    },
    "fillArea": true,
    "showZero": true,
    "measureAxis2": {
        "sqrtScale": false,
        "showTitle": true,
        "showAxis": true,
        "title": "",
        "customDomain": {
            "showDomain": false
        }
    },
    "measureAxis1": {
        "sqrtScale": false,
        "showTitle": true,
        "showAxis": true,
        "title": "",
        "customDomain": {
            "showDomain": false
        }
    },
    "theme": "wave",
    "dimensionAxis": {
        "showTitle": true,
```

```
                    "customSize": "auto",
                    "showAxis": true,
                    "title": "",
                    "icons": {
                        "useIcons": false,
                        "iconProps": {
                            "fit": "cover",
                            "column": "",
                            "type": "round"
                        }
                    }
                },
                "drawArea": {
                    "measure": "",
                    "showDrawArea": false,
                    "bounding1": "",
                    "bounding2": ""
                }
            },
            "type": "chart",
            "options": {}
        }
    },
    "lens_1": {
        "datasets": [
            {
                "id": "0FbB000000017QxKAI",
                "label": "Opportunities",
                "name": "opportunity",
               "url": "/services/data/v41.0/wave/datasets/0FbB000000017QxKAI"

            }
        ],
        "isFacet": true,
        "isGlobal": false,
        "label": "lens_1",
        "query": {
            "values": [
                "Name",
                "Owner.Name",
                "StageName",
                "Amount",
                "CloseDate",
                "ForecastCategoryName",
                "Account.Owner.Name"
            ]
        },
        "selectMode": "none",
        "type": "grain",
        "useGlobal": true,
        "visualizationParameters": {
            "options": {},
            "parameters": {
                "borderColor": "#e0e5ee",
```

```
                    "borderWidth": 1,
                    "cell": {
                        "backgroundColor": "#ffffff",
                        "fontColor": "#16325c",
                        "fontSize": 12
                    },
                    "columns": [],
                    "customBulkActions": [],
                    "header": {
                        "backgroundColor": "#f4f6f9",
                        "fontColor": "#16325c",
                        "fontSize": 12
                    },
                    "innerMajorBorderColor": "#a8b7c7",
                    "innerMinorBorderColor": "#e0e5ee",
                    "maxColumnWidth": 300,
                    "minColumnWidth": 40,
                    "mode": "variable",
                    "numberOfLines": 1,
                    "totals": true,
                    "verticalPadding": 8
                },
                "type": "table",
                "visualizationType": "valuestable"
            }
        }
    },
    "widgetStyle": {
        "backgroundColor": "#FFFFFF",
        "borderColor": "#E6ECF2",
        "borderEdges": [],
        "borderRadius": 0,
        "borderWidth": 1
    },
    "widgets": {
        "table_1": {
            "parameters": {
                "columnProperties": {
                     "Amount": {
                       "type": "bar",
                       "parameters": {}
                      }
                },
                "borderColor": "#e0e5ee",
                "borderWidth": 1,
                "cell": {
                    "backgroundColor": "#ffffff",
                    "fontColor": "#16325c",
                    "fontSize": 12
                },
                "columns": [],
                "customBulkActions": [],
                "exploreLink": true,
                "header": {
```

```
                "backgroundColor": "#f4f6f9",
                "fontColor": "#16325c",
                "fontSize": 12
            },
            "innerMajorBorderColor": "#a8b7c7",
            "innerMinorBorderColor": "#e0e5ee",
            "maxColumnWidth": 300,
            "minColumnWidth": 40,
            "mode": "variable",
            "numberOfLines": 1,
            "showRowIndexColumn": true,
            "step": "lens_1",
            "totals": true,
            "verticalPadding": 8,
            "evenRowColor": null,
            "oddRowColor": null
        },
        "type": "table"
    },
    "filterpanel_1": {
        "parameters": {
            "filterItemOptions": {
                "backgroundColor": "#FFFFFF",
                "borderColor": "#E6ECF2",
                "borderRadius": 4,
                "propertyColor": "#54698D",
                "valueColor": "#16325C",
                "borderWidth": 1
            },
            "itemsPerRow": 2,
            "title": {
                "separatorColor": "#E6ECF2",
                "text": {
                    "align": "left",
                    "color": "#091A3E",
                    "fontSize": 16,
                    "label": "Filters"
                },
                "visible": true
            }
        },
        "type": "filterpanel"
    },
    "link_1": {
        "parameters": {
            "destinationLink": {
                "name": "ac750443-1729-47fb-8d60-a35703076bf7"
            },
            "destinationType": "page",
            "fontSize": 12,
            "includeState": false,
            "text": "Opp Details",
            "textAlignment": "center",
            "textColor": "#44A2F5"
```

```
        },
        "type": "link"
    },
    "chart_1": {
        "parameters": {
            "autoFitMode": "none",
            "showValues": true,
            "bins": {
                "breakpoints": {
                    "high": 100,
                    "low": 0
                },
                "bands": {
                    "high": {
                        "color": "#008000",
                        "label": ""
                    },
                    "low": {
                        "color": "#B22222",
                        "label": ""
                    },
                    "medium": {
                        "color": "#ffa500",
                        "label": ""
                    }
                }
            },
            "legend": {
                "showHeader": true,
                "show": true,
                "customSize": "auto",
                "position": "right-top",
                "inside": false
            },
            "axisMode": "multi",
            "tooltip": {
                "showBinLabel": true,
                "measures": "",
                "showPercentage": false,
                "showDimensions": true,
                "showMeasures": true,
                "customizeTooltip": false,
                "dimensions": ""
            },
            "visualizationType": "hbar",
            "exploreLink": true,
            "title": {
                "fontSize": 14,
                "subtitleFontSize": 11,
                "label": "",
                "align": "center",
                "subtitleLabel": ""
            },
            "binValues": false,
```

```
                    "trellis": {
                        "flipLabels": false,
                        "showGridLines": true,
                        "size": [
                            100,
                            100
                        ],
                        "enable": false,
                        "type": "x",
                        "chartsPerLine": 4
                    },
                    "measureAxis2": {
                        "sqrtScale": false,
                        "showTitle": true,
                        "showAxis": true,
                        "title": "",
                        "customDomain": {
                            "showDomain": false
                        }
                    },
                    "measureAxis1": {
                        "sqrtScale": false,
                        "showTitle": true,
                        "showAxis": true,
                        "title": "",
                        "customDomain": {
                            "showDomain": false
                        }
                    },
                    "theme": "wave",
                    "step": "Amount_by_Acct_Owner_1",
                    "dimensionAxis": {
                        "showTitle": true,
                        "customSize": "auto",
                        "showAxis": true,
                        "title": "",
                        "icons": {
                            "useIcons": false,
                            "iconProps": {
                                "fit": "cover",
                                "column": "",
                                "type": "round"
                            }
                        }
                    }
                },
                "type": "chart"
            }
        }
    },
    "datasets": [
        {
            "id": "0FbB000000017QxKAI",
            "label": "Opportunities",
```

```
            "name": "opportunity",
            "url": "/services/data/v41.0/wave/datasets/0FbB0000000l7QxKAI"
        }
    ]
}
```

DASHBOARD JSON PROPERTIES

The dashboard JSON consists of properties that define layouts, pages, widgets, and steps. Some properties are exposed and editable in the dashboard designer user interface. Others are only editable via JSON.

Each dashboard JSON contains the following high-level properties.

Property Name	Details
`label`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** Name of the dashboard.
`mobileDisabled`	**Type** Boolean **Exposed in the Dashboard Designer's User Interface** Yes **Description** Specifies whether the dashboard can be accessed in the mobile app. Use this parameter to hide dashboards that don't perform well on mobile devices. Default is `false`.
`description`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** Description of the dashboard.
`state`	**Type** Array **Exposed in the Dashboard Designer's User Interface** No **Description** Specifies properties for all layouts, widgets, and steps defined in the dashboard. When you save a dashboard using the dashboard designer, the state of the dashboard is persisted in the JSON.
`datasets`	**Type** Array

Property Name	Details
	Exposed in the Dashboard Designer's User Interface No **Description** Specifies all datasets used by steps in the dashboard.

The following sections describe the different properties nested under `state`.

dataSourceLinks JSON
The `dataSourceLinks` section defines all data sources configured for the dashboard.

gridLayouts JSON
The `gridLayouts` section defines all layouts built for the dashboard.

widgetStyle JSON and Properties
The `widgetStyle` key contains the default widget properties that can be applied to each widget.

steps JSON
The `steps` key defines all steps available in a CRM Analytics dashboard. It contains a separate node for each step. Each step node has properties that define the query or list of static values. It also contains properties that control the behavior of the step, like whether to facet the step. The properties and JSON syntax vary based on the step type and whether the step is in compact form or SAQL form.

widgets JSON
The `widgets` section defines the widgets that appear in the dashboard. Each widget has a name.

filters JSON
The `filters` section defines the global filters added to a global filter panel widget, which is available in the dashboard designer.

`dataSourceLinks` JSON

The `dataSourceLinks` section defines all data sources configured for the dashboard.

For more information about connected data sources, see Configure Cross-Dataset Faceting with Connected Data Sources.

Example:

```
"dataSourceLinks": [
    {
        "fields": [
            {
                "dataSourceName": "ServiceOpportunity3",
                "dataSourceType": "saql",
                "fieldName": "AccountId"
            },
            {
                "dataSourceName": "account",
                "dataSourceType": "saql",
                "fieldName": "Id"
            }
```

```
        ],
        "label": "ServiceOpportunities Dataset to Account Dataset: Account ID",
        "name": "Link_970"
    },
    {
        "fields": [
            {
                "dataSourceName": "ServiceOpportunity3",
                "dataSourceType": "saql",
                "fieldName": "StageName"
            },
            {
                "dataSourceName": "Static_Opp_Stage_1",
                "dataSourceType": "static",
                "fieldName": "value"
            }
        ],
        "label": "Static Opp Stage to ServiceOpportunities Dataset",
        "name": "Link_953"
    }
]
```

dataSourceLinks Properties

The `dataSourceLinks` key defines all data source connections for the dashboard. It contains a separate node for each connection. Each connection has properties about each data source.

`dataSourceLinks` Properties

The `dataSourceLinks` key defines all data source connections for the dashboard. It contains a separate node for each connection. Each connection has properties about each data source.

Property Name	Details
`fields`	**Type** Array **Exposed in the Dashboard Designer's User Interface** Yes. **Description** List of data sources included in the connection. Each data source contains the following properties. **dataSourceName** API name of the dataset or ID of the static step. **dataSourceType** The type of data source: "`dataset`" for a dataset or "`static`" for a static step. **fieldName** Name of the field used to match records between the data sources.

Property Name	Details
`label`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes. **Description** Display label for the data source connection.
`name`	**Type** String **Exposed in the Dashboard Designer's User Interface** No. **Description** API name of the data source connection.

`gridLayouts` JSON

The `gridLayouts` section defines all layouts built for the dashboard.

For more information about layouts for CRM Analytics dashboards, see Generate Unique Dashboard Layouts for Different Devices. .

Example:

```
"gridLayouts": [
    {
        "name": "Default",
        "numColumns": 9,
        "pages": [
            {
                "label": "Page 1",
                "name": "41f040d6-48a6-4dc7-9166-eb985df7e9d8",
                "widgets": [
                    {
                        "colspan": 9,
                        "column": 0,
                        "name": "filterpanel_1",
                        "row": 0,
                        "rowspan": 3,
                        "widgetStyle": {
                            "borderEdges": []
                        }
                    },
                    {
                        "colspan": 9,
                        "column": 0,
                        "name": "chart_1",
                        "row": 3,
                        "rowspan": 5,
```

```
                    "widgetStyle": {
                        "borderEdges": []
                    }
                },
                {
                    "colspan": 2,
                    "column": 0,
                    "name": "link_1",
                    "row": 8,
                    "rowspan": 1,
                    "widgetStyle": {
                        "borderEdges": []
                    }
                }
            ]
        },
        {
            "label": "Page 2",
            "name": "ac750443-1729-47fb-8d60-a35703076bf7",
            "widgets": [
                {
                    "colspan": 9,
                    "column": 0,
                    "name": "filterpanel_1",
                    "row": 0,
                    "rowspan": 3,
                    "widgetStyle": {
                        "borderEdges": []
                    }
                },
                {
                    "colspan": 9,
                    "column": 0,
                    "name": "table_1",
                    "row": 3,
                    "rowspan": 7,
                    "widgetStyle": {
                        "borderEdges": []
                    }
                }
            ]
        }
    ],
    "rowHeight": "normal",
    "selectors": [],
    "style": {
        "alignmentX": "left",
        "alignmentY": "top",
        "backgroundColor": "#44A2F5",
        "cellSpacingX": 4,
        "cellSpacingY": 4,
        "fit": "original",
        "gutterColor": "#A3B8CC"
    },
```

```
        "version": 1
    },
    {
        "name": "Mobile",
        "numColumns": 2,
        "pages": [
            {
                "label": "Page 1",
                "name": "41f040d6-48a6-4dc7-9166-eb985df7e9d8",
                "widgets": [
                    {
                        "colspan": 2,
                        "column": 0,
                        "name": "filterpanel_1",
                        "row": 0,
                        "rowspan": 2,
                        "widgetStyle": {
                            "borderEdges": []
                        }
                    },
                    {
                        "colspan": 2,
                        "column": 0,
                        "name": "chart_1",
                        "row": 2,
                        "rowspan": 10,
                        "widgetStyle": {
                            "borderEdges": []
                        }
                    }
                ]
            }
        ],
        "rowHeight": "normal",
        "selectors": [
            "maxWidth(599)"
        ],
        "style": {
            "alignmentX": "left",
            "alignmentY": "top",
            "backgroundColor": "#2EC2BA",
            "cellSpacingX": 4,
            "cellSpacingY": 4,
            "fit": "original",
            "gutterColor": "#091A3E"
        },
        "version": 1
    }
]
```

The `gridLayouts` key defines all layouts for the dashboard. It contains a separate node for each layout. Each layout has properties that provide information about the devices that can use the layout and the placement of each widget in the layout. It also contains dashboard properties, like cell spacing in the grid and the dashboard's background color or image.

`gridLayouts` Properties

The `gridLayouts` key defines all layouts for the dashboard. It contains a separate node for each layout. Each layout has properties that provide information about the devices that can use the layout and the placement of each widget in the layout. It also contains dashboard properties, like cell spacing in the grid and the dashboard's background color or image.

Property Name	Details
`name`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes. **Description** Name of the layout.
`maxWidth`	Maximum width (in pixels) that the dashboard can use. If needed, CRM Analytics rearranges the existing dashboard widgets based on this setting in the layout.
`numColumns`	**Type** Integer **Exposed in the Dashboard Designer's User Interface** Yes. **Description** The number of columns in the designer grid for this layout.
`pages`	**Type** Array **Exposed in the Dashboard Designer's User Interface** No **Description** Contains properties that specify the name and ID of the page, and determine the placement of each widget in the dashboard layout.
`rowHeight`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes. **Description** The height of each row in the designer grid for this layout. Valid values are `fine` and `normal` (default).

Property Name	Details
`selectors`	**Type** Array **Exposed in the Dashboard Designer's User Interface** Yes. **Description** Device requirements that help CRM Analytics choose the optimal layout for the device accessing the dashboard.
`style`	**Type** Array **Exposed in the Dashboard Designer's User Interface** Yes. **Description** Properties about the designer grid, including columns, rows, cell spacing, and background.

`pages` Properties

The `pages` key contains properties that determine the placement of each widget in the dashboard layout. Currently, dashboard designer supports only one page for each layout.

Property Name	Details
`label`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** Label of the page in the dashboard layout.
`name`	**Type** String

Property Name	Details
	Exposed in the Dashboard Designer's User Interface No **Description** ID of the page in the dashboard layout.
`widgets`	**Type** Array **Exposed in the Dashboard Designer's User Interface** No **Description** Contains properties that determine the height and width of each widget, and where it's placed on the dashboard layout.

widgets Properties

The `widgets` key contains properties that determine the height and width of each widget, and where it's placed on the dashboard layout. Because the dashboard designer uses a grid, you specify the properties in terms of rows and columns. For example, you specify the number of columns to determine the width of a widget.

`widgets` Properties

The `widgets` key contains properties that determine the height and width of each widget, and where it's placed on the dashboard layout. Because the dashboard designer uses a grid, you specify the properties in terms of rows and columns. For example, you specify the number of columns to determine the width of a widget.

Property Name	Details
`name`	**Type** String **Exposed in the Dashboard Designer's User Interface** No **Description** Internal name of the widget. This name is used to reference the widget in the dashboard JSON.
`column`	**Type** Integer **Exposed in the Dashboard Designer's User Interface** Yes. Value is determined based on the widget's placement. **Description** The column number where the widget starts. `Column` and `row` specify the top left corner of the widget. If this widget is included in a container, these properties are relative to the container widget.

Property Name	Details
`row`	**Type** Integer **Exposed in the Dashboard Designer's User Interface** Yes. Value is determined based on the widget's placement. **Description** The row number where the widget starts. `Column` and `row` specify the top left corner of the widget.
colspan	**Type** Integer **Exposed in the Dashboard Designer's User Interface** Yes. Value is determined based on the widget's placement. **Description** The number of columns that a widget spans—the width of the widget. If the dashboard doesn't have enough columns to accommodate the specified width, then columns are added to the dashboard.
rowspan	**Type** Integer **Exposed in the Dashboard Designer's User Interface** Yes. Value is determined based on the widget's placement. **Description** The number of rows that a widget spans—the height of the widget. If the dashboard doesn't have enough rows to accomodate the specified height, then rows are added.
`widgetStyle`	**Type** Array **Available for These Widgets** • All widgets **Exposed in the Dashboard Designer's User Interface** No **Description** Contains properties that set the border type, border color, and background color.

widgetStyle Properties

The `widgetStyle` key contains properties that set the border type, border color, and background color of the widget. You can specify these attributes at two levels. To set the default for all dashboard widgets, use the `widgetStyle` field under `gridLayouts`. To set a specific widget, use the `widgetStyle` field under `widgets`. This setting overrides the default settings for all widgets.

`widgetStyle` Properties

The `widgetStyle` key contains properties that set the border type, border color, and background color of the widget. You can specify these attributes at two levels. To set the default for all dashboard widgets, use the `widgetStyle` field under `gridLayouts`. To set a specific widget, use the `widgetStyle` field under `widgets`. This setting overrides the default settings for all widgets.

Property Name	Details
`backgroundColor`	**Type** String **Available for These Widgets** All widgets **Exposed in the Dashboard Designer's User Interface** Yes **Description** Background color of the widget. The default is `#FFFFFF`.
`borderColor`	**Type** String **Available for These Widgets** All widgets **Exposed in the Dashboard Designer's User Interface** Yes **Description** Color of the widget's border. The default is `#FFFFFF`.
`borderEdges`	**Type** List **Available for These Widgets** All widgets **Exposed in the Dashboard Designer's User Interface** Yes **Description** A list of values that specify which edges of the widget have a border. Valid values are `left`, `right`, `top`, `bottom`, and `all`. Default is no border.
`borderRadius`	**Type** Integer **Available for This Widget** All widgets **Exposed in the Dashboard Designer's User Interface** Yes **Description** The roundness of the border corners.

Property Name	Details
	Valid values are: `0` (not rounded, default), `4`, `8`, and `16`. The higher the value, the more rounded the corner.
`borderWidth`	**Type** Integer **Available for These Widgets** All widgets **Exposed in the Dashboard Designer's User Interface** Yes **Description** Width of the widget's border. Valid values are `1`, `2` (default), `4`, and `8`.

`selectors` Properties

The `selectors` key contains layout properties that specify the layout name, designer grid settings, background settings, and requirements for devices that can use this layout.

Property Name	Details
`minWidth(`*`<width>`*`)`	**Type** Integer **Exposed in the Dashboard Designer's User Interface** Yes **Description** Minimum width (in pixels) of the devices supported by this layout.
`maxWidth(`*`<width>`*`)`	**Type** Integer **Exposed in the Dashboard Designer's User Interface** Yes **Description** Maximum width (in pixels) of the devices supported by this layout.
`orientation(`*`<orientation>`*`)`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** Orientation of the devices supported by this layout. Valid values are: `portrait` or `landscape`. If this property is not specified, then the layout supports both orientations.

Property Name	Details
`platform(<platform>)`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** Platform of the devices supported by this layout. Valid values are: `iOS` or `Android`. If this property is not specified, the layout supports both platforms.

`style` Properties

The `style` key contains the dashboard properties, like cell spacing in the grid, as well as the dashboard's background color or image.

Property Name	Details
`alignmentX`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** The horizontal alignment of the background image applied to the dashboard. Valid values are: `left` (default), `center`, and `right`.
`alignmentY`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** The vertical alignment of the background image applied to the dashboard. Valid values are: `top` (default), `center`, and `bottom`.
`backgroundColor`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** Background color of the dashboard, specified in hex color code. The default is `#FFFFFF`.
`cellSpacingX`	**Type** Integer

Property Name	Details
	Exposed in the Dashboard Designer's User Interface Yes **Description** Horizontal spacing (in pixels) between cells in the dashboard grid. Valid values are `0`, `4`, `8` (default), and `16`.
`cellSpacingY`	**Type** Integer **Exposed in the Dashboard Designer's User Interface** Yes **Description** Vertical spacing (in pixels) between cells in the dashboard grid. Valid values are `0`, `4`, `8` (default), and `16`.
`documentId`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** The 15-character document ID of the image to apply as the dashboard's background. To ensure security, upload the image file to Salesforce as a document, and select the **Externally Available Image** option. If this option is not selected or the referenced document is not an image, the image doesn't show up.
`fit`	**Type** String **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates how to scale the image. Valid values are: `original` (default), `stretch`, `tile`, `fitwidth`, and `fitheight`.
`image`	**Type** Array **Exposed in the Dashboard Designer's User Interface** Yes **Description** Identifies the image using the following properties. **name** Name of the image.

Property Name	Details

	namespace
	Optional. Namespace of the image. Default is null.
	Example
	`"image": {`
	`"name": "My_Corporate_Logo",`
	`"namespace": ""`
	`}`

`widgetStyle` JSON and Properties

The `widgetStyle` key contains the default widget properties that can be applied to each widget.

Note: You can specify these attributes at two levels. To set the default for all dashboard widgets, use the `widgetStyle` field under `gridLayouts`. To set a specific widget, use the `widgetStyle` field under `widgets`. Settings at the widget level override the default settings for all widgets.

Property Name	Details
`backgroundColor`	**Type** String **Available for This Widget** All widgets **Exposed in the Dashboard Designer's User Interface** Yes **Description** Color of the widget's background, specified in hex color code. The default is `#FFFFFF`.
`borderColor`	**Type** String **Available for This Widget** All widgets **Exposed in the Dashboard Designer's User Interface** Yes **Description** Color of the widget's border, specified in hex color code. The default is `#FFFFFF`. If no border is specified, the widget has no border.
`borderEdges`	**Type** List **Available for These Widgets** All widgets

Property Name	Details
	Exposed in the Dashboard Designer's User Interface Yes **Description** A list of values that specify which edges of the widget have a border. Valid values are `left`, `right`, `top`, `bottom`, and `all`. Default is no border.
`borderRadius`	**Type** Integer **Available for These Widgets** All widgets **Exposed in the Dashboard Designer's User Interface** Yes **Description** Roundness of the border corners. Valid values are: `0`(not rounded, default), `4`, `8`, and `16`. The higher the value, the more rounded the corner.
`borderWidth`	**Type** Integer **Available for These Widgets** All widgets **Exposed in the Dashboard Designer's User Interface** Yes **Description** Thickness of the border. Valid values are: `1`, `2` (default), `4`, and `8`. The higher the value, the thicker the border.

`steps` JSON

The `steps` key defines all steps available in a CRM Analytics dashboard. It contains a separate node for each step. Each step node has properties that define the query or list of static values. It also contains properties that control the behavior of the step, like whether to facet the step. The properties and JSON syntax vary based on the step type and whether the step is in compact form or SAQL form.

aggregateflex Step Type Properties

Use the `aggregateflex` step type to perform aggregate queries on a Analytics dataset. An aggregate query summarizes rows, like returning one row per grouping. For example, this step type can return the total amount per sales rep.

apex Step Type Properties

Use to include custom Apex functionality in a dashboard to access Salesforce platform features that aren't inherently supported in Analytics. For example, pull in data using any API, manipulate data using your Apex classes, or apply simple case statements or complex machine learning. You can even harness AppExchange for things like integrating Twitter with Analytics—all in a way that is familiar.

columns Properties

Use the `columns` property to add a column to a step query source in Compact form 2.0.

grain Step Type Properties

Use the `grain` step type for a values table. Values tables have no groupings, just a list of dataset fields to display as columns in the table.

saql Step Type Properties

Use the `saql` step type for special cases when querying a Analytics dataset. With this step type, you can write a custom SAQL query to create derived fields in a values table. You can specify dimensions without groupings. Also, you can bind the dataset name or entire query. For example, you can bind this step type to a static step that provides different SAQL queries or datasets based on a selection.

soql Step Type Properties

Use to directly query Salesforce objects—both standard and custom—to get Salesforce data that's not available in datasets. You can also query external objects created with an OData adapter for Salesforce Connect. To view the results in the dashboard, the user viewing the dashboard must have access to the object and fields queried by the `soql` step. The SOQL query returns only records to which the user has access.

source Step Type Properties

Use the `sources` property to add `columns`, `groups`, `filters`, and formulas to a step query.

staticflex Step Type Properties

Use the `staticflex` step type to manually define your own set of data. For example, you can use this step to populate a list of static values in a toggle or list widget. It can also be used to provide values to a binding. For example, it can provide possible filters, groups, measures, sort order, and limits.

visualizationParameters Properties

The `visualizationParameters` key contains chart properties defined for the step. When you associate the step with a widget, the widget properties override these settings.

filters Properties

Use the `filters` property to add a filter to a step query. Although you can create filters for query steps in the user interface, you have to manually define filters for static steps in the dashboard JSON.

`aggregateflex` Step Type Properties

Use the `aggregateflex` step type to perform aggregate queries on a Analytics dataset. An aggregate query summarizes rows, like returning one row per grouping. For example, this step type can return the total amount per sales rep.

Field Name	Description
`datasets`	An array of datasets used by this step. Specify the alias of each dataset. If the `pigql` attribute references a dataset that's not specified here, the dashboard doesn't render.
`broadcastFacet`	Controls whether the step's selections are broadcasted as facets. Faceting is when a selection in a widget filters other steps in the dashboard. Default is `true`.

Field Name	Description
`receiveFacetSource`	Controls whether the step in the receiving query listens for broadcasted facets and applies them as filters. The mode can be "`all`", "`none`", "`include`", or "`exclude`". Use `steps` to list which steps to include or exclude. Default is `all`.
`isGlobal`	This applies to global filters created before Winter '18 only. Newer global filters, those created in the global filter panel widget, don't require a step. Default is false. You can apply this property only on steps that are connected to a global filter widget—all other steps ignore this property. A global filter filters other steps in the dashboard that have `useGlobal` set to `true` and reference the same dataset.
`label`	Step label, which is primarily used for display in the dashboard designer user interface.
`query`	The query used to retrieve results from a dataset. It must contain at least one grouping and can be in SAQL or compact form. Use a query in SAQL form to customize the query in a way that can't be done using compact form. For compact form (2.0), the query can contain the following properties. **`sources`** The array of sources where data comes from. For more information, see sources Properties on page 67. **`sourceFilters`** The filters applied to `sources`. When you use this object, the `sources` name is the `steps` key. the For more information, see filters Properties. **`aggregateFilters`** The array of filters applied on aggregate columns, which are fields where an aggregate function, such as average or sum, is applied to a measure column. Aggregate columns are defined in `columns` under `sources`. For more information, see filters Properties. **`orders`** The array of sort orders applied to data in a pivot table. An order can contain the following properties: **`name`** The column or group name used to apply the sort order. **`ascending`** The sort order applied to the data. Indicates whether to order the data in ascending order (`true`) or not (`false`). **`filters`** The filters used to sort a measure column in a pivot table and is applied on a dimension value in the pivoted group. For more information about sorting data in a pivot table, see Organize and Summarize Data in a Pivot Table. **`nulls`** The sort order applied to null values in the data. To sort null values last, set `nulls`to `last`To sort null values first, set `nulls`to `first`. **`rowTotals`** The array of dimension groups to which the values of each row are rolled up. To calculate `rowTotals`, you must also include `columnGroups` for the data.

Field Name	Description

`columnTotals`
The array of dimension groups to which the values of each column are rolled up. If the query is a grain query, that is, all the columns are non-aggregate columns, or if the query is not grouped by any dimension columns, to calculate totals, set `columnTotals` to `all` .

`columnGroups`
The pivoted dimension group, which is the last-defined group. If the defined group is a compound date, such as Year-Month for order date, then include each date grain in `columnGroups`. For example, `["OrderDate_Year", "OrderDate_Month"]`.

`limit`
The maximum number of results that the step can return. When you create an `aggregateflex` step, by default, CRM Analytics sets `limit` to 2,000. To return more results, set the `limit` attribute accordingly. The higher you increase the limit, the longer the query takes. When a limit isn't set, Analytics returns up to 10,000 results. For more information, see filters Properties.

Compact Form 2.0 Query Example

```
"query": {
  "sources": [
    {
        "columns": [
          {
            "field": ["sum", "Amount"],
            "name": "A"
          }
        ],
        "groups": [
            "Product_Container",
            "Order_priority"
        ],
        "filters": [
          [
            "Profit",
            [16000],
            "<",
         ]
        ],
        "name": "SuperStoreSales"
    },
    {
        "columns": [
          {
            "formula": "A*2",
            "name": "B"
          }
        ],
        "groups": [],
        "filters": []
      ]
    },
  ],
```

Field Name	Description

```
  "sourceFilters": {
    "SuperStoreSales" {
        "filters": [
          [
            "Sales",
            [
              1,
              100000
            ],
            ">=<="
          ],
          [
            "Customer_Segment",
            [
              "Consumer",
              "Corporate",
            ],
            "in"
          ]
        ]
    }
  },
  "aggregateFilters": [
    [
        "A",
        [10],
        ">"
    ]
  ],
  "orders": [
    {
        "name": "Product_Container",
        "ascending": true,
        "filters": [],
        "nulls": "last"
    },
    {
        "name": "Order_Priority",
        "ascending": true,
        "filters": [],
        "nulls": "last"
    }
  ],
  "rowTotals": ["Product_Container"],
  "columnTotals": [
    "Product_Container",
    "Order_Priority"
  ],
  "columnGroups": ["Order_Priority"],
  "limit": 2000
}
```

For compact form (1.0), the query can contain the following properties.

Field Name	Description

`filters`
The filters to apply to the data. For more information, see filters Properties on page 73.

`groups`
The dimension to group by.

`limit`
The maximum number of results that the step can return. When you create an `aggregateflex` step, by default, CRM Analytics sets `limit` to 2,000. To return more results, set the `limit` attribute accordingly. The higher you increase the limit, the longer the query takes. When a limit isn't set, Analytics returns up to 10,000 results.

Note: Limit only impacts the number of records returned for display. The limit doesn't impact calculations across all records in the dataset. For instance, a query groups by Account Name and there are one million Account Names in a dataset. When the limit is 20, Analytics returns 20 records for display. But the summary row provides a total for the one million records.

`measures`
The measures returned by the query.

Note: If you provide an aggregate function for a measure, then the measure value must be a string, not an array.

`order`

Sort order (ascending or descending) of the first specified measure. To order the results in ascending order, set `ascending` to `true`. To order the results in descending order, set `ascending` to `false`. If you don't want to impose a specific order, remove the entire `"order"` parameter.

Compact-form (1.0) Query Example

```
"query": {
  "filters": [
    [
      "Account.Industry",
      [
        "Agriculture",
        "Apparel",
        "Banking",
        "Biotechnology",
        "Consulting",
        "Education",
        "Electronics",
        "Energy",
        "Engineering",
        "Finance",
        "Healthcare",
        "Insurance",
        "Manufacturing",
        "Media",
        "Retail",
        "Technology",
        "Telecommunications",
```

Field Name	Description

```
        "Transportation",
        "Utilities"
      ],
      "in"
    ]
  ],
  "groups": [ "Account.Industry" ],
  "measures": [
    [
      "avg",
      "Amount"
    ]
  ],
  "order": [
    [
      -1,
      { "ascending": false }
    ]
  ]
}
```

For SAQL form, the query can contain the following properties

`pigql`
Specify the SAQL query to retrieve data from a dataset. When you specify a SAQL query, you must specify the filters, limits, and ordering inside the `pigql` attribute. CRM Analytics ignores the following attributes if they are set under the `query` attribute: `filters`, `limit`, and `order`.

`measures`
Defines the fields included as measures. When using a SAQL-form query, you must include each measure in this parameter and in the `pigql` parameter. You can change the UI label of a measure by setting the `display` option.

```
 "count", "*", null, {
    "display": "% of total flights"
}
```

`groups`
Defines the dimension fields to group by. When using a SAQL-form query, you must specify the group-by dimension in this parameter and in the `group` property in the `pigql` parameter.

SAQL-form Query Example

```
"query": {
  "pigql": "q = load \"ServiceOpportunity3\";\n
            q = filter q by 'Account.Industry' in
                [\"Agriculture\", \"Apparel\", \"Banking\",
\"Biotechnology\",
                 \"Consulting\", \"Education\", \"Electronics\",
 \"Energy\",
                 \"Engineering\", \"Finance\", \"Healthcare\",
\"Insurance\",
                 \"Manufacturing\", \"Media\", \"Retail\",
\"Technology\",
```

Field Name	Description

	<pre>\"Telecommunications\", \"Transportation\", \"Utilities\"];\n
	q = group q by 'Account.Industry';\n
	q = foreach q generate 'Account.Industry' as 'Account.Industry',
	count() as 'count';\n
	q = order q by 'count' desc;\n
	q = limit q 1000;",
	"measures": [
	[
	"count",
	"*",
	"count"
	]
	],
	"groups": ["Account.Industry"],
	"measuresMap": {}
	}</pre>
	For more information about SAQL queries, see the Analytics SAQL Developer Guide

`selectMode`	Determines the selection interaction. The options for charts, tables, lists, and toggle selectors are: `none`, `single`, `singlerequired`, `multi` and `multirequired`. Note: `selectMode` doesn't apply to number, values table, date, range, and global filter widgets.
`start`	The initial selections that are applied to the step when the dashboard first opens.
`type`	Step type. Set to `aggregateflex`. Note: If you bind a step property for an `aggregateflex` step, you must use the correct bindings syntax. For more information about bindings, see the Analytics Bindings Developer Guide.
`useGlobal`	Indicates whether to apply global filters to this step (`true`) or not (`false`). If the step is in SAQL form, you must also set `autoFilter` to `true` to apply the global filters. By default, the global filter widget filters compact-form steps only.
`visualizationParameters`	Visualization details about the step. For more information, see visualizationParameters Properties.

Example: `aggregateflex` Step

```
"steps": {
            "Account_Owner_1": {
                "type": "aggregateflex",
                "query": {
                    "measures": [
                        [
                            "avg",
```

```
                "Amount"
            ]
        ],
        "groups": [
            "Account_Owner"
        ],
        "order": [
            [
                "avg_Amount",
                {
                    "ascending": false
                }
            ]
        ]
    },
    "visualizationParameters": {
        "options": {},
        "type": "chart",
        "parameters": {
            "visualizationType": "hbar",
            "autoFitMode": "keepLabels",
            "theme": "wave",
            "title": {
                "label": "",
                "fontSize": 14,
                "subtitleLabel": "",
                "subtitleFontSize": 11,
                "align": "center"
            },
            "showValues": true,
            "axisMode": "multi",
            "binValues": false,
            "bins": {
                "breakpoints": {
                    "low": 0,
                    "high": 100
                },
                "bands": {
                    "low": {
                        "label": "",
                        "color": "#B22222"
                    },
                    "medium": {
                        "label": "",
                        "color": "#ffa500"
                    },
                    "high": {
                        "label": "",
                        "color": "#008000"
                    }
                }
            },
            "dimensionAxis": {
                "showAxis": true,
```

```
        "showTitle": true,
        "title": "",
        "customSize": "auto",
        "icons": {
            "useIcons": false,
            "iconProps": {
                "column": "",
                "fit": "cover",
                "type": "round"
            }
        }
    },
    "measureAxis1": {
        "sqrtScale": false,
        "showAxis": true,
        "customDomain": {
            "showDomain": false
        },
        "showTitle": true,
        "title": ""
    },
    "measureAxis2": {
        "sqrtScale": false,
        "showAxis": true,
        "customDomain": {
            "showDomain": false
        },
        "showTitle": true,
        "title": ""
    },
    "legend": {
        "show": true,
        "showHeader": true,
        "inside": false,
        "descOrder": false,
        "position": "right-top",
        "customSize": "auto"
    },
    "tooltip": {
        "customizeTooltip": false,
        "showDimensions": true,
        "dimensions": "",
        "showMeasures": true,
        "measures": "",
        "showPercentage": true,
        "showNullValues": true,
        "showBinLabel": true
    },
    "trellis": {
        "enable": false,
        "showGridLines": true,
        "flipLabels": false,
        "type": "x",
        "chartsPerLine": 4,
```

```
                        "size": [
                            100,
                            100
                        ]
                    },
                    "applyConditionalFormatting": true,
                    "showActionMenu": true,
                    "columnMap": {
                        "trellis": [],
                        "plots": [
                            "avg_Amount"
                        ],
                        "dimensionAxis": [
                            "Account_Owner"
                        ]
                    }
                }
            },
            "datasets": [
                {
                    "id": "0FbB00000000xHDKAY",
                    "name": "DTC_Opportunity_SAMPLE",
                    "label": "DTC Opportunity",
                   "url": "/services/data/v48.0/wave/datasets/0FbB00000000xHDKAY"

                }
            ],
            "useGlobal": true,
            "isGlobal": false,
            "label": "Account_Owner_1",
            "broadcastFacet": true,
            "receiveFacetSource": {
                "mode": "all",
                "steps": []
            },
            "selectMode": "single"
        }
    }
```

steps Properties for Compact Form and SAQL Form

The properties and JSON syntax in the `query` node of an `aggregateflex` step type vary based on whether the step is in compact form or SAQL form.

`steps` Properties for Compact Form and SAQL Form

The properties and JSON syntax in the `query` node of an `aggregateflex` step type vary based on whether the step is in compact form or SAQL form.

These examples display the `aggregateflex`step type in compact form and in SAQL form. For an explanation of the `aggregateflex` step type and its properties, see aggregateflex Step Type Properties

Example: **Compact-Form `aggregateflex` Step**

```
        "steps": {
            "all_Amount_1": {
                "type": "aggregateflex",
                "query": {
                    "measures": [
                        [
                            "avg",
                            "Amount"
                        ]
                    ],
                    "groups": [
                        "Account_Name"
                    ],
                    "order": [
                        [
                            "avg_Amount",
                            {
                                "ascending": false
                            }
                        ]
                    ]
                },
                "visualizationParameters": {
                    "options": {},
                    "type": "chart",
                    "parameters": {
                        "visualizationType": "hbar",
                        "autoFitMode": "keepLabels",
                        "theme": "wave",
                        "title": {
                            "label": "",
                            "fontSize": 14,
                            "subtitleLabel": "",
                            "subtitleFontSize": 11,
                            "align": "center"
                        },
                        "showValues": true,
                        "axisMode": "multi",
                        "binValues": false,
                        "bins": {
                            "breakpoints": {
                                "low": 0,
                                "high": 100
                            },
                            "bands": {
                                "low": {
                                    "label": "",
                                    "color": "#B22222"
                                },
                                "medium": {
                                    "label": "",
                                    "color": "#ffa500"
                                },
```

```
            "high": {
                "label": "",
                "color": "#008000"
            }
        }
    },
    "dimensionAxis": {
        "showAxis": true,
        "showTitle": true,
        "title": "",
        "customSize": "auto",
        "icons": {
            "useIcons": false,
            "iconProps": {
                "column": "",
                "fit": "cover",
                "type": "round"
            }
        }
    },
    "measureAxis1": {
        "sqrtScale": false,
        "showAxis": true,
        "customDomain": {
            "showDomain": false
        },
        "showTitle": true,
        "title": ""
    },
    "measureAxis2": {
        "sqrtScale": false,
        "showAxis": true,
        "customDomain": {
            "showDomain": false
        },
        "showTitle": true,
        "title": ""
    },
    "legend": {
        "show": true,
        "showHeader": true,
        "inside": false,
        "descOrder": false,
        "position": "right-top",
        "customSize": "auto"
    },
    "tooltip": {
        "customizeTooltip": false,
        "showDimensions": true,
        "dimensions": "",
        "showMeasures": true,
        "measures": "",
        "showPercentage": true,
        "showNullValues": true,
```

```
                    "showBinLabel": true
                },
                "trellis": {
                    "enable": false,
                    "showGridLines": true,
                    "flipLabels": false,
                    "type": "x",
                    "chartsPerLine": 4,
                    "size": [
                        100,
                        100
                    ]
                },
                "applyConditionalFormatting": true,
                "showActionMenu": true,
                "columnMap": {
                    "trellis": [],
                    "plots": [
                        "avg_Amount"
                    ],
                    "dimensionAxis": [
                        "Account_Name"
                    ]
                }
            }
        },
        "datasets": [
            {
                "id": "0Fb6g000000HTE1CAO",
                "name": "DTC_Opportunity_SAMPLE",
                "label": "DTC Opportunity",
               "url": "/services/data/v48.0/wave/datasets/0Fb6g000000HTE1CAO"

            }
        ],
        "useGlobal": true,
        "isGlobal": false,
        "label": "all_Amount_1",
        "broadcastFacet": true,
        "receiveFacetSource": {
            "mode": "all",
            "steps": []
        },
        "selectMode": "single"
    },
```

Example: **SAQL-Form `aggregateflex` Step**

When the step is in SAQL form, notice how each group and measure are defined in the `groups` and `measures` properties, respectively, and also in the `pigql` property. Other parts of the query—like filters, limits, and order—only need to be defined one time in the `pigql` property. You specify the compact form elements of `"groups"` and `"measures"` so that the associated chart widget can render the correct projections.

In the following sample step, notice that the `'sum_Amount'` and `'sum_quantity'` projections in the `pigql` property are referenced in `"measures"` as `[[ "count", "*", "sum_Amount" ], [ "count", "*", "sum_quantity" ]]`. Measure projections in the `pigql` property always include the aggregation, underscore (_), and the name of the measure (`'sum_Amount'`) so that they can be referenced in the compact form.

```
"steps": {
    "Product_StageName_2": {
        "type": "aggregateflex",
        "visualizationParameters": {
            "options": {}
        },
        "query": {
            "pigql": "q = load \"Flexy_Sales\";\n
                      q = group q by ('Product', 'StageName');\n
                      q = foreach q generate 'Product' as 'Product',
                                             'StageName' as 'StageName',
                                             sum('Amount') as 'sum_Amount',
                                             sum('quantity') as 'sum_quantity';\n
                      q = filter q by 'sum_Amount' >= 14550720 && 'sum_Amount' <=
58807698;\n
                      q = order q by 'sum_Amount' desc;\nq = limit q 10000;",
            "measures": [
                [
                    "count",
                    "*",
                    "sum_Amount"
                ],
                [
                    "count",
                    "*",
                    "sum_quantity"
                ]
            ],
            "groups": [
                "Product",
                "StageName"
            ]
        },
        "broadcastFacet": true,
        "receiveFacetSource": {
            "mode": "all",
            "steps": []
        },
        "useGlobal": true,
        "isGlobal": false,
        "datasets": [{
            "name": "Flexy_Sales",
            "url": "/services/data/v38.0/wave/datasets/0FbB00000000q5gKAA",
            "id": "0FbB00000000q5gKAA"
        }]
    }
}
```

`apex` Step Type Properties

Use to include custom Apex functionality in a dashboard to access Salesforce platform features that aren't inherently supported in Analytics. For example, pull in data using any API, manipulate data using your Apex classes, or apply simple case statements or complex machine learning. You can even harness AppExchange for things like integrating Twitter with Analytics—all in a way that is familiar.

To set up an `apex` step, create an Apex class that returns data in a shape that Analytics can consume. And then define the step with the `apex` step type in the dashboard JSON. The step calls the Apex REST endpoint to return the data from the Apex controller class.

Like a `soql` step, the Apex controller can return tabular data. Unlike `saql` and `soql` step types, the `apex` step type doesn't define the `"strings"`, `"numbers"`, and `"groups"` arrays. The Apex class response must declare these column types.

When you define an `apex` step, you can use a selection or results binding on the `body` parameter in the step JSON. You can also reference this step type in a results binding. This step type doesn't support faceting. If you run a Analytics REST API query using an `apex` step, the query runs as the logged-in user. Each REST API query counts against the org's API limits.

Note the following limitations with `apex` steps:

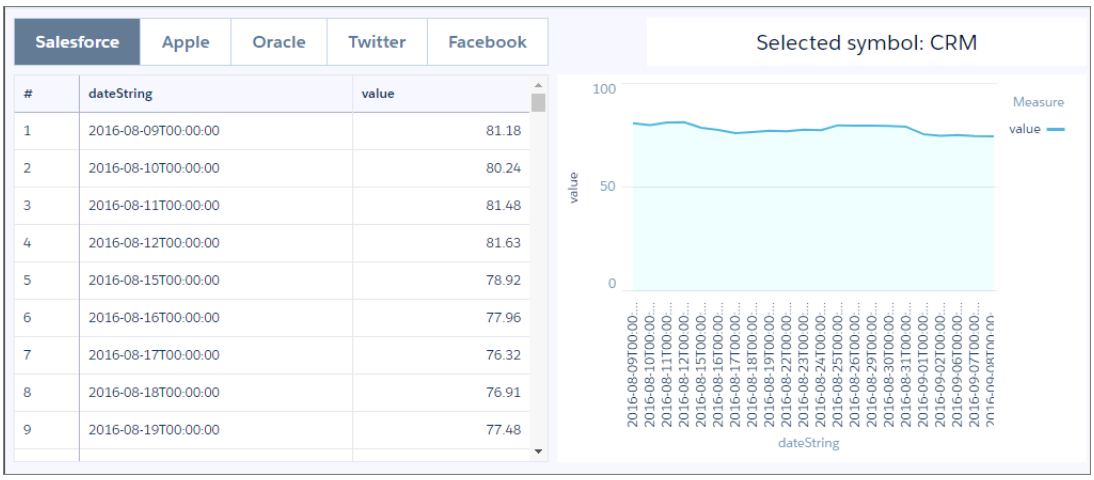
- If you include dashboards in a package, `apex` steps aren't included. You must migrate the Apex classes separately.
- The Android mobile app doesn't support this type of step.

Field Name	Description
`type`	Step type. Set to `apex`.
`label`	Step label, which is primarily used for display in the dashboard designer user interface.
`query`	Query that returns the results. Can consist of the following parameters: **`body`** Optional. Blob containing the input parameters needed by the Apex controller class. **`path`** Required. Specifies the class path and name of the Apex controller class.

Example: `apex` Step

```
"GetStockData": {
    "query": {
        "body": {
            "symbol": "CRM"
        },
        "path": "stocks"
    },
    "type": "apex"
}
```

Example: You want to display real-time stock data from a website in your dashboard. You want to fetch the data from an external API and add logic to determine the time of day for each stock price. Here's the goal for this dashboard.



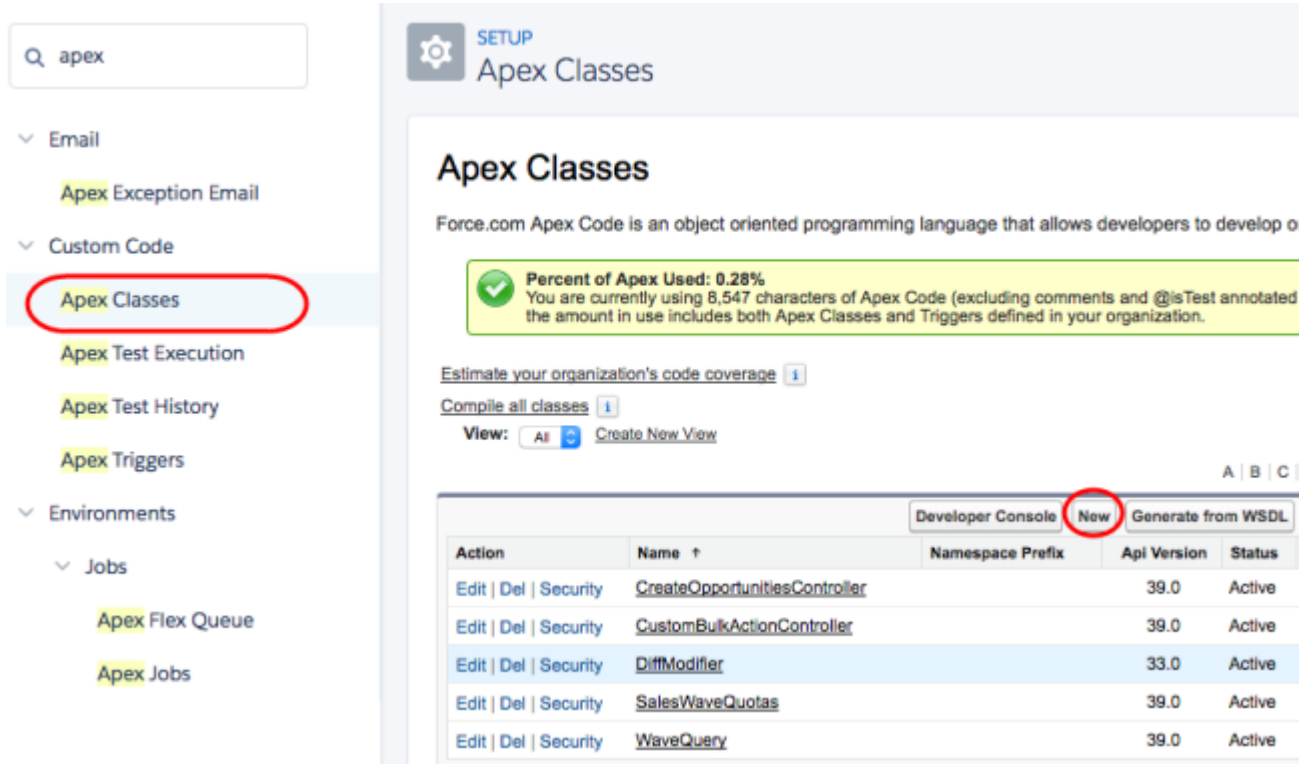
To create this dashboard, complete the following steps:

1. Create the Apex controller class that gets the data from the stock website.
2. Add the stock website to the allowed sites in Salesforce.
3. Create the `apex` step in addition to the other dashboard widgets.

Create the Apex Class

Define the Apex controller class and methods that return the stock price for different companies over time.

1. From setup, enter *`Apex Classes`* in the Quick Find box, and select **Apex Classes**.
2. Click **New**.



3. Add the following code.

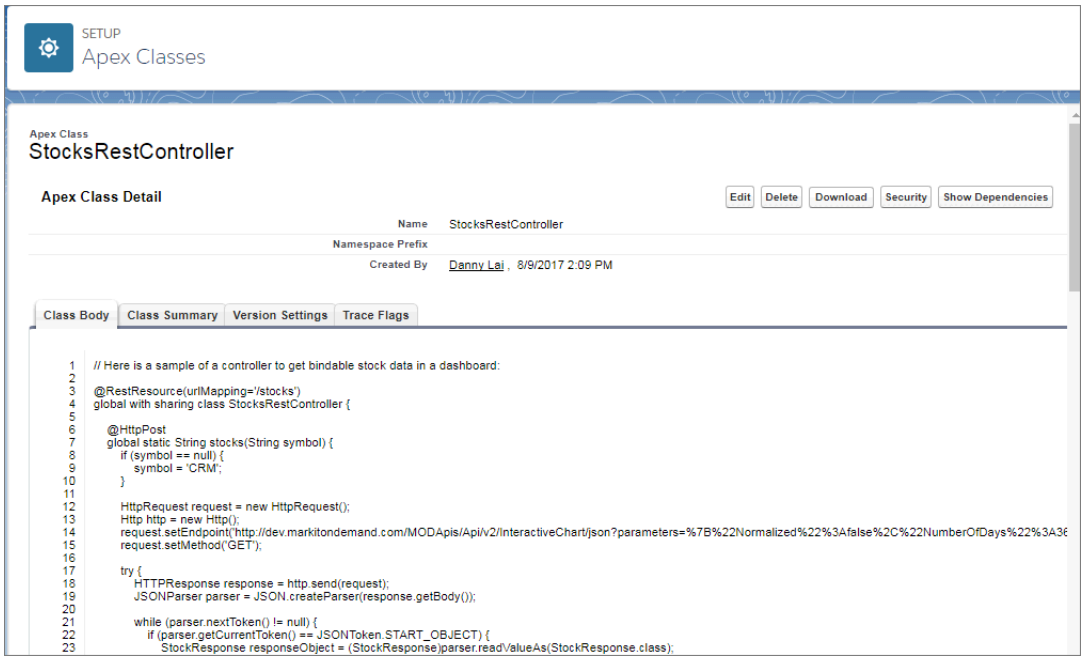
```
@RestResource(urlMapping='/stocks')
global with sharing class StocksRestController {

    @HttpPost
    global static String stocks() {
       ApexStepRequest stepRequest = new ApexStepRequest(new ApexStepRequest.Parameter[]{

            new ApexStepRequest.Parameter('symbol',
ApexStepRequest.ParameterType.STRING_PARAM)
        });

        // fetch some stock data
        HttpRequest request = new HttpRequest();
        Http http = new Http();
        // make sure this domain is whitelisted in the proxy

request.setEndpoint('https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&interval=5min&apikey=1QRP8AID7134OMPA&symbol='

            // default to CRM
            + stepRequest.getStringParam('symbol', 'CRM')
        );
        request.setMethod('GET');

        try {
            HTTPResponse response = http.send(request);
            JSONParser parser = JSON.createParser(response.getBody());
```

```
            List<Map<String, Object>> returnItems = new List<Map<String, Object>>();
            while (parser.nextToken() != null) {
                if (parser.getCurrentToken() == JSONToken.START_OBJECT &&
parser.getCurrentName() != null && parser.getCurrentName().startsWith('20')) {
                    String dateLabel = parser.getCurrentName();
                    parser.nextToken();
                    parser.nextToken();
                    System.debug(parser.getText());

                    Map<String, Object> curRow = new Map<String, Object>();
                    curRow.put('date', dateLabel);
                    curRow.put('value', Double.valueOf(parser.getText()));
                    returnItems.add(curRow);
                }
            }

            return JSON.serialize(new ApexStepResponse(returnItems));
        } catch(Exception exp) {
            System.debug('exception '+exp);
        }


        return '';
    }

}
```

4. Add two more Apex classes for `ApexStepRequest` and `ApexStepResponse` to support the Apex stocks class.

   The code for these classes, along with the stock step code, can be found in this public `AnalyticsApexSteps` GitHub repo, in the `/force-app/main/apex/common` directory. This GitHub repo also contains other Apex step examples.

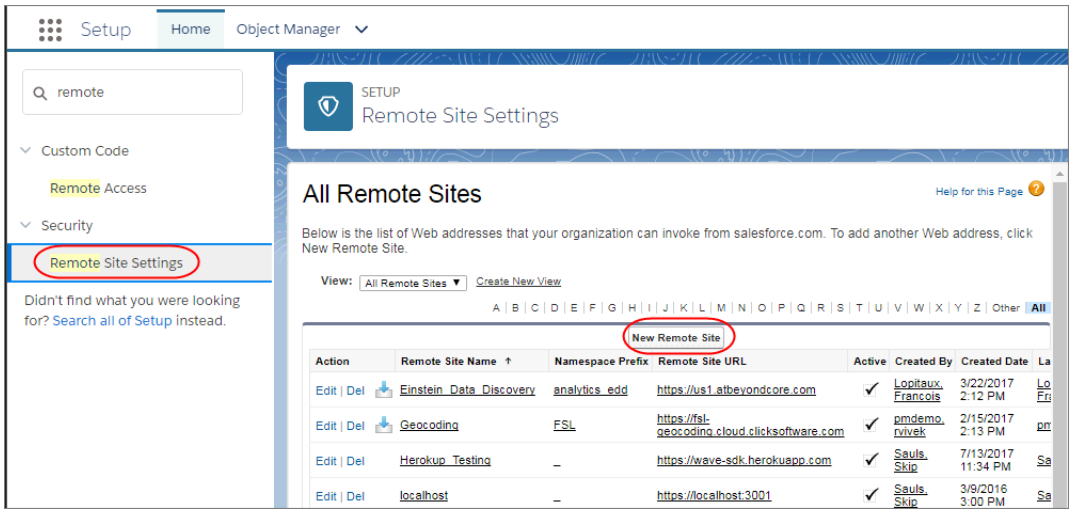
5. Click **Save**.



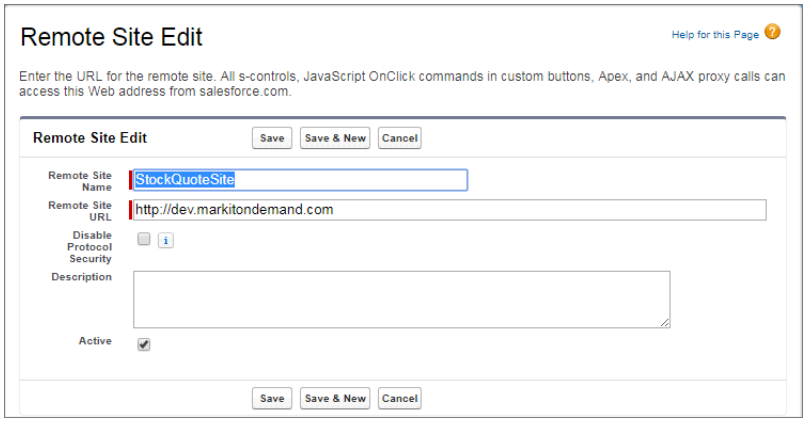
Allow the External Website

To enable Apex to query external REST endpoints, allow the website in Salesforce.

1. From setup, enter `Remote` in the Quick Find box, and select **Remote Site Settings**.
2. Click **New Remote Site**.



3. Enter the remote site name and URL.



4. Click **Save**.

Create the Apex Step

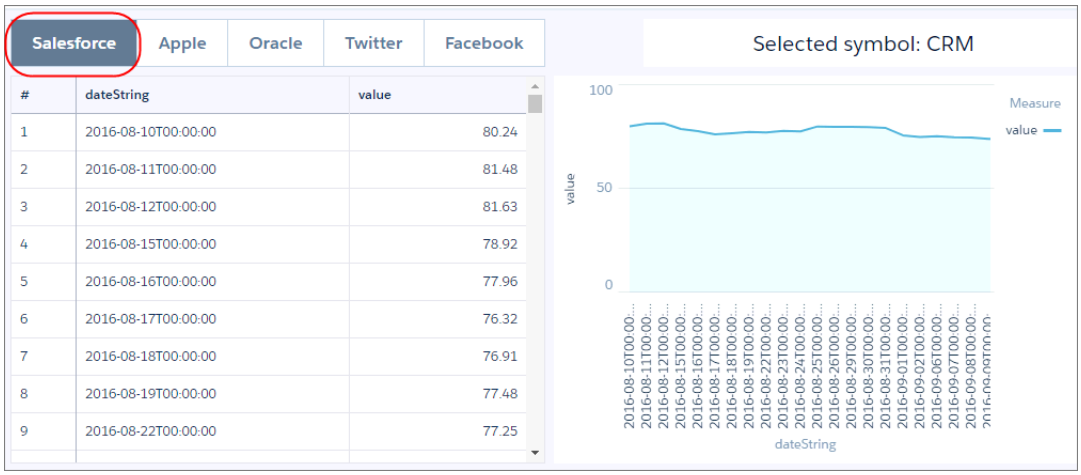
Manually define the `apex` step in the dashboard JSON to get the stock results from the Apex class that you previously created.

In the dashboard JSON, add the following `apex` step.

```
"StockData": {
    "query": {
        "body": {
            "symbol": "{{cell(CompaniesList_1.selection, 0, \"value\").asString()}}"
        },
        "path": "stocks"
    },
    "type": "apex"
}
```

The `apex` step query contains the following parameters.

`body`
The binding retrieves the stock symbol when a company is selected in the toggle widget.



`path`
Specifies the REST API resource path to the Apex controller, as specified in the `urlMapping` of the `RestResource` annotation on the Apex controller.

The dashboard shows four widgets. The toggle widget is based on a static step that maps predefined company names to stock symbols. The table widget and line chart use the `apex` step to retrieve the results. The text widget uses a binding to concatenate "Selected symbol:" and the stock symbol of the selected company in the toggle widget.

Here's the final dashboard JSON.

```
{
    "label": "Stock Quotes",
    "mobileDisabled": false,
    "state": {
        "dataSourceLinks": [],
        "filters": [],
        "gridLayouts": [
            {
                "name": "Default",
                "numColumns": 12,
                "pages": [
                    {
                        "label": "Untitled",
                        "name": "b177a6e6-8ab6-4914-af36-09c7fa23e0c8",
                        "widgets": [
                            {
                                "colspan": 6,
                                "column": 0,
                                "name": "pillbox_1",
                                "row": 0,
                                "rowspan": 1,
                                "widgetStyle": {
                                    "borderEdges": []
                                }
                            },
                            {
                                "colspan": 5,
                                "column": 7,
                                "name": "text_1",
                                "row": 0,
                                "rowspan": 1,
                                "widgetStyle": {
                                    "borderEdges": []
                                }
                            },
                            {
                                "colspan": 6,
                                "column": 0,
                                "name": "table_1",
                                "row": 1,
                                "rowspan": 7,
                                "widgetStyle": {
                                    "borderEdges": []
                                }
                            },
```

```
                        {
                            "colspan": 6,
                            "column": 6,
                            "name": "chart_1",
                            "row": 1,
                            "rowspan": 7,
                            "widgetStyle": {
                                "borderEdges": []
                            }
                        }
                    ]
                }
            ],
            "rowHeight": "normal",
            "selectors": [],
            "style": {
                "alignmentX": "left",
                "alignmentY": "top",
                "backgroundColor": "#F2F6FA",
                "cellSpacingX": 8,
                "cellSpacingY": 8,
                "fit": "original",
                "gutterColor": "#C5D3E0"
            },
            "version": 1
        }
    ],
    "layouts": [],
    "steps": {
        "CompaniesList_1": {
            "datasets": [],
            "dimensions": [],
            "groups": [],
            "label": "CompaniesList",
            "numbers": [],
            "selectMode": "singlerequired",
            "strings": [],
            "type": "staticflex",
            "values": [
                {
                    "display": "Salesforce",
                    "value": "CRM"
                },
                {
                    "display": "Apple",
                    "value": "AAPL"
                },
                {
                    "display": "Oracle",
                    "value": "ORCL"
                },
                {
                    "display": "Twitter",
                    "value": "TWTR"
```

```
                    },
                    {
                        "display": "Facebook",
                        "value": "FB"
                    }
                ],
                "broadcastFacet": true
            },
            "StockData": {
                "query": {
                    "body": {
                        "symbol": "{{cell(CompaniesList_1.selection, 0,
\"value\").asString()}}"
                    },
                    "path": "stocks"
                },
                "type": "apex"
            }
        },
        "widgetStyle": {
            "backgroundColor": "#FFFFFF",
            "borderColor": "#E6ECF2",
            "borderEdges": [],
            "borderRadius": 0,
            "borderWidth": 1
        },
        "widgets": {
            "table_1": {
                "parameters": {
                    "borderColor": "#e0e5ee",
                    "borderWidth": 1,
                    "cell": {
                        "backgroundColor": "#ffffff",
                        "fontColor": "#16325c",
                        "fontSize": 12
                    },
                    "columns": [],
                    "customBulkActions": [],
                    "exploreLink": false,
                    "header": {
                        "backgroundColor": "#f4f6f9",
                        "fontColor": "#16325c",
                        "fontSize": 12
                    },
                    "innerMajorBorderColor": "#a8b7c7",
                    "innerMinorBorderColor": "#e0e5ee",
                    "mode": "fittocontainer",
                    "numberOfLines": 1,
                    "step": "StockData",
                    "verticalPadding": 8,
                    "evenRowColor": null,
                    "oddRowColor": null
                },
                "type": "table"
```

```
            },
            "pillbox_1": {
                "parameters": {
                    "compact": false,
                    "exploreLink": false,
                    "step": "CompaniesList_1"
                },
                "type": "pillbox"
            },
            "text_1": {
                "parameters": {
                    "fontSize": 20,
                    "text": "Selected symbol: {{ cell(CompaniesList_1.selection, 0,
\"value\").asString() }}",
                    "textAlignment": "center",
                    "textColor": "#091A3E"
                },
                "type": "text"
            },
            "chart_1": {
                "parameters": {
                    "autoFitMode": "keepLabels",
                    "showPoints": false,
                    "legend": {
                        "showHeader": true,
                        "show": true,
                        "customSize": "auto",
                        "position": "right-top",
                        "inside": false
                    },
                    "axisMode": "multi",
                    "tooltip": {
                        "showBinLabel": true,
                        "measures": "",
                        "showPercentage": false,
                        "showDimensions": true,
                        "showMeasures": true,
                        "customizeTooltip": false,
                        "dimensions": ""
                    },
                    "visualizationType": "line",
                    "dashLine": {
                        "measures": "",
                        "showDashLine": false
                    },
                    "exploreLink": false,
                    "title": {
                        "fontSize": 14,
                        "subtitleFontSize": 11,
                        "label": "",
                        "align": "center",
                        "subtitleLabel": ""
                    },
                    "trellis": {
```

```
            "flipLabels": false,
            "showGridLines": true,
            "size": [
                100,
                100
            ],
            "enable": false,
            "type": "x",
            "chartsPerLine": 4
        },
        "fillArea": true,
        "showZero": true,
        "measureAxis2": {
            "sqrtScale": false,
            "showTitle": true,
            "showAxis": true,
            "title": "",
            "customDomain": {
                "showDomain": false
            }
        },
        "measureAxis1": {
            "sqrtScale": false,
            "showTitle": true,
            "showAxis": true,
            "title": "",
            "customDomain": {
                "showDomain": false
            }
        },
        "theme": "wave",
        "step": "StockData",
        "dimensionAxis": {
            "showTitle": true,
            "customSize": "auto",
            "showAxis": true,
            "title": "",
            "icons": {
                "useIcons": false,
                "iconProps": {
                    "fit": "cover",
                    "column": "",
                    "type": "round"
                }
            }
        },
        "drawArea": {
            "measure": "",
            "showDrawArea": false,
            "bounding1": "",
            "bounding2": ""
        }
    },
    "type": "chart"
```

```
            }
        }
    },
    "datasets": []
}
```

`columns` Properties

Use the `columns` property to add a column to a step query source in Compact form 2.0.

Example: **Aggregate measure column defined in Compact form 2.0**

```
"columns": [
    {
        "field": [
            "sum",
            "Amount"
        ]
    }
]
```

Example: **Non-aggregate columns defined in Compact form 2.0**

```
"columns": [
    {
        "field": "Customer_Name",
        "name": "Customer_Name"
    },
    {
        "field": "Product_Name",
        "name": "Product_Name"
    },
    {
        "field": "Profit",
        "name": "Profit"
    }
]
```

Example: **Formula column defined in Compact form 2.0**

```
"columns": [
    {
        "formula": "A*2",
        "name": "B"
    }
]
```

`grain` Step Type Properties

Use the `grain` step type for a values table. Values tables have no groupings, just a list of dataset fields to display as columns in the table.

Field Name	Description
`datasets`	An array of datasets used by this step. Specify the alias of each dataset. If the `pigql` attribute references a dataset that's not specified here, the dashboard doesn't render. Note: A `grain` step can only have one dataset.
`isFacet`	Enables this step to facet and be faceted by other steps. Faceting is when a selection in a widget filters other steps in the dashboard.
`isGlobal`	This applies to global filters created before Winter '18 only. Newer global filters, those created in the global filter panel widget, don't require a step. Default is false. You can apply this property only on steps that are connected to a global filter widget—all other steps ignore this property. A global filter filters other steps in the dashboard that have `useGlobal` set to `true` and reference the same dataset.
`label`	Step label, which is primarily used for display in the dashboard designer user interface.

`query`	The query used to retrieve results from a dataset. The query can be in compact form only and can contain the following properties:
	
	`filters`
	The filter conditions to apply to the data.
	
	`values`
	List of dataset fields to show as table columns.
	
	`limit`
	The maximum number of results that the step can return. When you create a `grain` step, by default, Analytics sets `limit` to 100. To return more results, set the `limit` attribute accordingly. The higher you increase the limit, the longer the query takes. When a limit isn't set, CRM Analytics returns up to 10,000 results.
	
	Note: Limit only impacts the number of records returned for display. The limit doesn't impact calculations across all records in the dataset. For instance, a query groups by Account Name and there are one million Account Names in a dataset. When the limit is 20, CRM Analytics returns 20 records for display. But the summary row provides a total for the one million records.
	
	`"steps": {`
	`"lens_1": {`
	`"type": "grain",`
	`"visualizationParameters": {`
	`"visualizationType": "valuestable",`
	`"options": {}`
	`},`
	`"query": {`
	`"values": [`
	`"Case.IsEscalated",`
	`"AccountId",`
	`"StageName",`
	`"ForecastCategory",`
	`"IsClosed",`
	`"Amount"`

Field Name	Description

	`],`
	`"limit": 500`
	`},`
	`"isFacet": true,`
	`"useGlobal": true,`
	`"isGlobal": false,`
	`"label": "",`
	`"datasets": [`
	`{`
	`"name": "ServiceOpportunity16",`
	`"url": "/services/data/v39.0/wave/datasets/0FbB00000000kOSKAY",`
	`"id": "0FbB00000000kOSKAY"`
	`}`
	`]`
	`}`
	`},`
	For more information SAQL queries, see the Analytics SAQL Developer Guide .

`type`	Step type. Set to `grain`.
`useGlobal`	Indicates whether to apply global filters to this step (`true`) or not (`false`).
`visualizationParameters`	Visualization details about the step. For more information, see visualizationParameters Properties.

Example: `grain` Step

```
"lens_1": {
  "type": "grain",
  "visualizationParameters": {
    "visualizationType": "valuestable",
    "options": {
      "totals": true
    }
  },
  "query": {
    "filters": [
      [
        "Amount",
          [[
            1000000,
            7780844
          ]],
       ">=<="
      ]
    ],
    "values": [
      "AccountId",
      "ForecastCategory",
      "CloseDate",
```

```
      "Amount",
      "Account.Name",
      "StageName"
    ]
  },
  "isFacet": true,
  "useGlobal": true,
  "isGlobal": false,
  "label": "",
  "datasets": [
    {
      "name": "ServiceOpportunity3",
      "url": "/services/data/v39.0/wave/datasets/0FbR0000000012uKAA",
      "id": "0FbR0000000012uKAA"
    }
  ]
}
```

`saql` Step Type Properties

Use the `saql` step type for special cases when querying a Analytics dataset. With this step type, you can write a custom SAQL query to create derived fields in a values table. You can specify dimensions without groupings. Also, you can bind the dataset name or entire query. For example, you can bind this step type to a static step that provides different SAQL queries or datasets based on a selection.

Field Name	Description
`type`	Step type. Set to `saql`.
`label`	Step label, which is primarily used for display in the dashboard designer user interface.
`query`	The SAQL query used to retrieve results. For more information about SAQL queries, see Analytics SAQL Developer Guide. When you create a `saql`-type step, by default, no `limit` is set in the query. When a limit isn't set, a step can return up to 10,000 results. To return more results, set the `limit` attribute accordingly. The higher you increase the limit, the longer the query takes. Note: Limit only impacts the number of records returned for display. The limit doesn't impact calculations across all records in the dataset. For instance, a query groups by Account Name and there are one million Account Names in a dataset. When the limit is 20, Analytics returns 20 records for display. But the summary row provides a total for the one million records. You can bind a `saql` step type to dynamically set the dataset used in the query or change the entire query based on a selection in another step. For example, you can create a toggle based on a static step that allows the dashboard viewer to select a query. Each toggle option contains a valid SAQL query. Each query can be based on different datasets. (See the JSON example at the end of this section.)
`broadcastFacet`	Controls whether the step's selections are broadcasted as facets. Faceting is when a selection in a widget filters other steps in the dashboard. Default is `true`.

Field Name	Description
`receiveFacetSource`	Controls whether the step in the receiving query listens for broadcasted facets and applies them as filters. The mode can be "`all`", "`none`", "`include`", or "`exclude`". Use `steps` to list which steps to include or exclude. Default is `all`.
`useGlobal`	Indicates whether to apply global filters to this step (`true`) or not (`false`). If the step is in SAQL form, you must also set `autoFilter` to `true` to apply the global filters. By default, the global filter widget filters compact-form steps only.
`selectMode`	Determines the selection interaction. The options for charts are: `none`, `single`, and `singlerequired`. The options for list, range, and toggle selectors are: `single`, `singlerequired`, `multi`, and `multirequired`. Note: `selectMode` doesn't apply to number, values table, compare table, date, and global filter widgets.

`start`	The initial selections that are applied to the step when the dashboard first opens.
	Note: A widget with a `saql`-type step can return up to 10,000 results, by default. If Analytics doesn't find the initial value in those results, it ignores this setting.
	Single-selection example:
	`"start": [ "06 - Proposal/Price Quote" ]`
	Multi-selection example:
	`"start": [ "06 - Proposal/Price Quote", "01 - Prospecting" ]`

`strings`	Flags the specified fields as non-grouping dimensions. For example, you can flag a field as a dimension for a values table in which no groupings are allowed. If you use a binding to determine the field, specify the binding here as well.
`numbers`	Flags the specified fields as measures. If you use a binding to determine the field, specify the binding here as well.
`groups`	Flags the specified fields as groupings. For example, you can flag a field as a grouping for a pivot table or chart. If you use a binding to determine the field, specify the binding here as well.

Example: `saql` Step

```
      "steps": {
           "Amount_1": {
               "type": "saql",
               "query": "q = load \"DTC_Opportunity_SAMPLE\";\nq = group q by
'Account_Name';\nq = foreach q generate 'Account_Name' as 'Account_Name', count() as
'count';\nq = order q by 'Account_Name' desc;\nq = limit q 1000;",
               "useGlobal": true,
               "numbers": [],
               "groups": [],
```

```
"strings": [],
"visualizationParameters": {
    "type": "chart",
    "parameters": {
        "visualizationType": "hbar",
        "autoFitMode": "keepLabels",
        "theme": "wave",
        "title": {
            "label": "",
            "fontSize": 14,
            "subtitleLabel": "",
            "subtitleFontSize": 11,
            "align": "center"
        },
        "showValues": true,
        "axisMode": "multi",
        "binValues": false,
        "bins": {
            "breakpoints": {
                "low": 0,
                "high": 100
            },
            "bands": {
                "low": {
                    "label": "",
                    "color": "#B22222"
                },
                "medium": {
                    "label": "",
                    "color": "#ffa500"
                },
                "high": {
                    "label": "",
                    "color": "#008000"
                }
            }
        },
        "dimensionAxis": {
            "showAxis": true,
            "showTitle": true,
            "title": "",
            "customSize": "auto",
            "icons": {
                "useIcons": false,
                "iconProps": {
                    "column": "",
                    "fit": "cover",
                    "type": "round"
                }
            }
        },
        "measureAxis1": {
            "sqrtScale": false,
            "showAxis": true,
```

```
                    "customDomain": {
                        "showDomain": false
                    },
                    "showTitle": true,
                    "title": ""
                },
                "measureAxis2": {
                    "sqrtScale": false,
                    "showAxis": true,
                    "customDomain": {
                        "showDomain": false
                    },
                    "showTitle": true,
                    "title": ""
                },
                "legend": {
                    "show": true,
                    "showHeader": true,
                    "inside": false,
                    "descOrder": false,
                    "position": "right-top",
                    "customSize": "auto"
                },
                "tooltip": {
                    "customizeTooltip": false,
                    "showDimensions": true,
                    "dimensions": "",
                    "showMeasures": true,
                    "measures": "",
                    "showPercentage": true,
                    "showNullValues": true,
                    "showBinLabel": true
                },
                "trellis": {
                    "enable": false,
                    "showGridLines": true,
                    "flipLabels": false,
                    "type": "x",
                    "chartsPerLine": 4,
                    "size": [
                        100,
                        100
                    ]
                },
                "applyConditionalFormatting": true,
                "showActionMenu": true
            }
        },
        "label": "Amount_1",
        "selectMode": "single",
        "broadcastFacet": true,
        "receiveFacetSource": {
            "mode": "all",
            "steps": []
```

```
                }
            }
        },
```

Example: `saql` Step with a Bound Query

```
"query": "{{cell(static_1.selection, 0, \"query\").asString()"
```

The `static_1` step looks like this:

```
values: [
{query: "q = load \"opp\"; ..."}
{query: "q = load \"account\"; ..."}
]
```

Tip: Every dataset referenced in a binding of a `saql` step must be referenced by another step in the dashboard. If not, Analytics removes the dataset from the `datasets` attribute in the dashboard JSON. As a result, widgets based on the `saql` step display an error because it can't find the dataset.

`soql` Step Type Properties

Use to directly query Salesforce objects—both standard and custom—to get Salesforce data that's not available in datasets. You can also query external objects created with an OData adapter for Salesforce Connect. To view the results in the dashboard, the user viewing the dashboard must have access to the object and fields queried by the `soql` step. The SOQL query returns only records to which the user has access.

For more information about using Salesforce Connect to access external data, see the Salesforce Connect Salesforce Help.

Field Name	Description
`type`	Step type. Set to `soql`.
`label`	Step label, which is primarily used for display in the dashboard designer user interface.
`query`	The SOQL query used to retrieve results from a Salesforce object. Because Salesforce—not Analytics—executes the query, the maximum number of returned results depends on the SOQL query limit. For more information about SOQL queries, see SOQL and SOSL Reference. **Note:** Every field listed in your SOQL query must be listed in one of the metadata properties: `strings`, `numbers`, or `groups`.
`strings`	Flags the specified fields as non-grouping dimensions. For example, you can flag a field as a dimension for a values table in which no groupings are allowed. If you use a binding to determine the field, specify the binding here as well.
`numbers`	Flags the specified fields as measures. If you use a binding to determine the field, specify the binding here as well.
`groups`	Flags the specified fields as groupings. For example, you can flag a field as a grouping for a pivot table or chart. If you use a binding to determine the field, specify the binding here as well.

Note: This step type doesn't support faceting. If needed, you can use a binding to filter other steps based on a selection in a `soql` step.

Example: **`soql` Step**

```
"soql": {
  "type": "soql",
  "query": "SELECT Name from ACCOUNT",
  "strings": ["Name"],
  "numbers": [],
  "groups": [],
  "selectMode": "single"
}
```

Example: **Total Number of Active Salesforce Users**

The following query gets the total number of active users from the Salesforce User object. In this example, we name the count expression as "foo" and then reference it by name in the `numbers` parameter.

```
"soql": {
  "type": "soql",
  "query": "SELECT count(id) foo FROM User u WHERE u.IsActive = true",
  "strings": [],
  "numbers": ["foo"],
  "groups": [],
  "selectMode": "single"
}
```

If you don't name the count expression, the query produces an "expr0" field that you can use in the `numbers` parameter.

```
"soql": {
  "type": "soql",
  "query": "SELECT count(id) FROM User u WHERE u.IsActive = true",
  "strings": [],
  "numbers": [ "expr0" ],
  "groups": [],
  "selectMode": "single"
}
```

`source` Step Type Properties

Use the `sources` property to add `columns`, `groups`, `filters`, and formulas to a step query.

Field Name	Description
`columns`	The columns for the source. For more information, see columns Properties.
`groups`	The dimensions to group the data in the source.
`filters`	The filters applied to the columns in the source. For more information, see filters Properties
`name`	The name of the data source. If the query has formula columns, they're added as the last source without the `name` property.

Field Name	Description
`cogroupType`	When there are multiple data sources with different `name` properties, this parameter is used to define which records are included in the blended data for this query. Valid values for `cogroupType` are `left`, `right` `inner` `full`. For `left` and `right` blends, the initial dataset or blend is considered the left one. For more information on data blending, see Explore Multiple Datasets with a Single Query.

Example: **Grain Query**

```
"sources": [
    {
        "columns": [
          {
            "field": "Customer_Name",
            "name": "Customer_Name"
          },
          {
            "field": "Customer_Segment",
            "name": "Customer_Segment"
          },
          {
            "field": "Product_Name",
            "name": "Product_Name"
          },
          {
            "field": "City",
            "name": "City"
          },
          {
            "field": "Order_ID",
            "name": "Order_ID"
          },
        ],
        "groups": [],
        "filters": [],
        "name": "SuperStoreSales"
    }
]
```

Example: **Query without groupings**

When a query isn't grouped by a dimension, set `groups` to `["all"]`

```
"sources": [
    {
        "columns": [
          {
            "field": ["sum", "Amount"]
            "name": "A"
          },
        ],
        "groups": ["all"],
        "filters": [],
```

```
        "name": "SuperStoreSales"
    }
]
```

Example: **Query with groupings**

```
"sources": [
    {
        "columns": [
          {
            "field": ["sum", "Amount"]
            "name": "A"
          },
        ],
        "groups": ["Customer_Segment", "Order_priority"],
        "filters": [],
        "name": "SuperStoreSales"
    }
]
```

Example: **Query with formula column**

```
"sources": [
    {
        "columns": [
          {
            "field": ["sum", "Amount"]
            "name": "A"
          },
          {
            "formula": "case \nwhen starts_with('Product_Container', \"Small|") then
null\nelse count()\nend\n",
            "name": "B"
          }
        ],
        "filters": [],
        "groups": [
            "Order_Priority",
            "Product_Container"
        ],
        "name": "SuperStoreSales"
    }
]
```

Example: **Query with columns referencing formulas**

Formula columns are defined in a separate source without a `name` property.

```
"sources": [
    {
        "columns": [
          {
            "field": ["sum", "Amount"]
            "name": "A"
```

```
            }
        ],
        "groups": ["Customer_Segment", "Order_Priority"],
        "filters": [],
        "name": "SuperStoreSales"
    },
    {
        "columns": [
          {
            "forumula": "A*2",
            "name": "B"
          },
          {
            "forumula": "B/100",",
            "name": "C"
           }
         ]
    }
]
```

Example: **Query with multiple data sources**

```
"sources": [
    {
        "columns": [
          {
            "field": ["avg", "Profit"]
            "name": "A"
          }
        ],
        "groups": ["City"],
        "filters": [],
        "name": "SuperStoreSales",
        "cogroupType": "left"
    },
    {
        "columns": [
          {
            "field": ["sum", "Amount"],
            "name": "B"
          }
        ],
        "groups": ["City"],
        "filters": [],
        "name": "Opportunity"
    }
]
```

`staticflex` Step Type Properties

Use the `staticflex` step type to manually define your own set of data. For example, you can use this step to populate a list of static values in a toggle or list widget. It can also be used to provide values to a binding. For example, it can provide possible filters, groups, measures, sort order, and limits.

Field Name	Description
`type`	Step type. Set to `staticflex`.
`label`	Step label, which is primarily used for display in the dashboard designer user interface.
`values`	Values for the static step. You can have multiple fields for each static value, where each field provides different information about the value, like a label, measurement, or range. When the static step is associated with a widget, the widget uses the first specified field as the display label. You can use other fields to specify values or ranges that you can use to facet or bind steps. For more information about binding a static step to another step, see the Analytics Bindings Developer Guide. For more information about faceting a static step with another data source, see Configure Cross-Dataset Faceting with Connected Data Sources. If you use the static step wizard to create the step, the step contains the following default fields: `display` and `value`, as shown in the JSON example. You can change these arbitrary field names and add more fields. The values in each field must have the same datatype, like numbers, strings, or arrays. For instance, if a row has `"value": "123"`, another row can't have `"value": [123]`.
`broadcastFacet`	Controls whether the step's selections are broadcasted as facets. Faceting is when a selection in a widget filters other steps in the dashboard. Default is `true`. To enable this step to broadcast facets, set this property to `true` and connect this step with another data source. This step type can't receive facets.
`isGlobal`	Not applicable. You can only apply this property on steps that are connected to a global filter widget—all other steps ignore this property.
`useGlobal`	Indicates whether to apply global filters to this step (`true`) or not (`false`).
`selectMode`	Determines the selection interaction. The options for charts are: `none`, `single`, and `singlerequired`. The options for list, range, and toggle selectors are: `single`, `singlerequired`, `multi`, and `multirequired`. Note: `selectMode` doesn't apply to number, values table, compare table, date, and global filter widgets.
`start`	The initial selections that are applied to the step when the dashboard first opens. Static example that sets an initial selection in a dashboard: `"start": { "display": [ "Atlanta" ] }`

Field Name	Description
	Binding example that selects the logged in user manager ID in a dashboard : `"start": { "display" : "{{column(LoggedinUserInfo_1.result, [\"ManagerId\"]).asObject()}}" }`

Example: `staticflex` Step

```
"my_opps_1": {
    "numbers": [],
    "strings": [],
    "groups": [],
    "broadcastFacet": true,
    "selectMode": "single",
    "datasets": [],
    "dimensions": [],
    "type": "staticflex",
    "label": "my opps",
    "values": [
        {
            "display": "All opps",
            "value": "false"
        },
        {
            "display": "My opps",
            "value": "true"
        }
    ]
}
```

`visualizationParameters` Properties

The `visualizationParameters` key contains chart properties defined for the step. When you associate the step with a widget, the widget properties override these settings.

Field Name	Description
`options`	Specifies chart properties for steps added or clipped to the designer. CRM Analytics overrides these options when they are defined in the widget parameters. For more information about chart properties, see Visualizing Data With Charts.
`visualizationType`	Specifies the chart type. You can override the chart type at the widget level. Valid values for visualizationType are: • `calheatmap`— calendar heat map • `choropleth` — choropleth (map) • `combo` — lines and bars to show multiple metrics

Field Name	Description
	• `flatgauge` — flat gauge • `funnel` — funnel • `hbar` — horizontal bar • `hdot` — horizontal dot plot • `heatmap`— heat map • `matrix`— matrix • `parallelcoords` — parallel coordinates • `pie` — donut • `pivottable` — pivot table • `polargauge` — polar gauge • `pyramid` — pyramid • `rating` — rating • `scatter` — scatter plot • `stackhbar` — stacked horizontal bar • `stackpyramid` — stacked pyramid • `stackvbar` — stacked vertical bar • `stackwaterfall` — stacked waterfall • `time` — timeline • `time-bar` — time bar • `time-combo` — time bar • `vbar` — vertical bar • `vdot` — vertical dot plot • `waterfall` — waterfall

`filters` Properties

Use the `filters` property to add a filter to a step query. Although you can create filters for query steps in the user interface, you have to manually define filters for static steps in the dashboard JSON.

The syntax for a filter in the step definition varies based on whether the step is in compact or SAQL form. This section describes the filter syntax for compact-form steps, including a description and example of every operator. For information about the filters for SAQL-form steps, see the Analytics SAQL Developer Guide.

Filter Syntax for Compact Form 2.0

Filters defined in compact-form steps have the following syntax.

```
"filters": [[
    "field",
    [value],
    "operator"
]]
```

Example: For example, the following filter shows only records where `Sales` is between `1` and `100,000` and `Customer_Segment` is either `Consumer` or `Corporate`.

```
"filters": [
    [
        "Sales",
        [
          1,
          100000
        ],
        ">=<="
    ],
    [
        "Customer_Segment",
        [
          "Consumer"
          "Corporate"
        ],
        "in"
    ]
]]
```

Example: Examples of dimension filters

```
   "filters": [Region", ["East", "West"], "in"]
   "filters": [Region", ["st"], "matches"]
   "filters": [Region", [], "isnull"]
   "filters": [["year", "OrderDate"], [2020, 2022], "in"]
```

Example: Examples of measure filters

```
   "filters": ["Profit", [], "isnull"]
   "filters": ["Profit", [1], ">="]
   "filters": ["Profit", [1, 100], ">=<="]
```

Example: Examples of absolute date filters

```
   "filters": ["OrderDate", ["2019-10-12", "2019-10-22"], "between"
   "filters": ["OrderDate", ["2019-10-12", null], "between"]
   "filters": ["OrderDate", [null, "2019-10-22"], "between"]
```

Example: Example of a relative date filter

```
   "filters": ["OrderDate", [[fiscal_year", -1], ["day", 0]], "between"]]"between"]
```

Example: Example of a compound filter and a compound date filter in a time-zone enabled org

```
"filters": [
    [
        [
```

```
            "Type",
            "LeadSource"
          ],
          [
            "Existing Business",
            "Advertisement"
          ],
          "in"
      ]
      [
          [
            ["year", "CloseDate"],
            ["quarter", "CloseDate"]
          ],
          [
            [2020, 2],
            [2021, 3]
          ],
          "in"
      ]
]
```

Filter Syntax for Compact Form 1.0

Filters defined in compact-form steps have the following syntax.

```
"filters": [[
    "field",
    [value],
    "operator"
]]
```

For example, the following filter shows only records with a “Customer” account type.

```
"filters": [[
    "Account_Type",
    ["Customer"],
    "in"
]]
```

To compare against multiple values, include the values in an array, like this.

```
"filters": [[
    "field",
    [[value1, value2, value3]],
    "operator"
]]
```

To specify an absolute date value for a date filter, specify the value in epoch format, where the value is the number of milliseconds since January 1, 1970 midnight UTC (1970-01-01 00:00:00). The following example shows dataset rows with a close date on or before January 1, 2016.

```
"filters": [[
    "Close Date",
```

```
    [[
        1451606400000,
        null
    ]],
    ">="
]]
```

Operators

You can use different operators in a filter. The supported operators depend on the field type. If you don't specify the operator, Analytics applies the == operator.

Operator	Description	Compact-Form 2.0 Example	Compact-Form 1.0 Example
`in`	Value of dataset field equals one of the specified values. Applies to dimensions only.	`["Dimension",["Value1","Value2"],"in"]`	`["Dimension",["Value1","Value2"],"in"]`
`not in`	Value of dataset field is not in the specified list of values. Applies to dimensions only.	`["Dimension",["Value1","Value2"],"not in"]`	`["Dimension",["Value1","Value2"],"not in"]`
`matches`	Value of dataset field contains the specified value. This operator is not case-sensitive. Applies to dimensions only.	`["Dimension",["Val"],"matches"]`	`["Dimension",["Val"],"matches"]`
`is null`	Value of dataset field is null. Applies to measures, dimensions and absolute dates in Compact form 2.0 and to measures only in Compact form 1.0.	`["Measure",[],"isnull"]`	`["Measure",[[]],"isnull"]`
`is not null`	Value of dataset field is not null. Applies to measures, dimensions and absolute dates in Compact form 2.0 and to measures only in Compact form 1.0.	`["Measure",[],"isnotnull"]`	`["Measure",[[]],"isnotnull"]`
`==`	Value of dataset field equals the specified value. Applies to measures only.	`["Measure",[1],"=="]`	`["Measure",[[1]],"=="]`

Operator	Description	Compact-Form 2.0 Example	Compact-Form 1.0 Example
`!=`	Value of dataset field does not equal the specified value. Applies to measures only.	`["Measure",[1],"!="]`	`["Measure",[[1]],"!="]`
`<`	Value of dataset field is less than the specified value. Applies to measures only.	`["Measure",[10],"<"]`	`["Measure",[[10]],"<"]`
`>`	Value of dataset field is greater than the specified value. Applies to measures only.	`["Measure",[1],">"]`	`["Measure",[[1]],">"]`
`<=`	Value of dataset field is less than or equal to the specified value. Applies to measures only in Compact form 2.0 and to measures and absolute dates only in Compact form 1.0.	`["Measure",[10],"<="]`	• `["Measure",[[10]],"<="]` • `["Date",[[1294000000]],"<="]`
`>=`	Value of dataset field is greater than or equal to the specified value. Applies to measures only in Compact form 2.0 and to measures and absolute dates only in Compact form 1.0.	`["Measure",[1],">="]`	• `["Measure",[[1]],">="]` • `["Date",[[-345600000]],">="]`
`>=<=` or `between` with measures and dates in Compact form 1.0	Value of dataset field is between the specified values, inclusive. For relative dates, you can specify the following time periods: "year", "quarter", "month", "week", and "day". Applies to measures only in Compact form 2.0 and measures and dates only in Compact form 1.0. **Note:** You can also use these operators with	`["Measure",[1,10],">=<="]`	• `["Measure",[[1,10]],">=<="]` • `["Date",[[-345600000,1294000000]],">=<="]` • `["Date",[["fiscalMonth",-1],["fiscalMonth",1]],">=<="]` • `["Close Date",[null,1451606400000]],"<="]`

Operator	Description	Compact-Form 2.0 Example	Compact-Form 1.0 Example
	measures in Compact form 2.0 and measures and dates in Compact form 1.0: **>=<** Greater than or equal to one value, but less than another. **><=** Greater than one value, but less than or equal to another. **><** Between two values, exclusive.		
`between` with dates in Compact form 2.0	Value of dataset field is between the specified values, inclusive. For relative dates, you can specify the following time periods: "year", "quarter", "month", "week", and "day".	• `["Date",[null, 1388448000000], "between"]` • `["Date",[1388448000000, null], "between"]` • `["Date",[["year", -1], ["year", 1]], "between"]`	

To view a complete example with a binding syntax, see the Bind the Initial Filter Selection documentation.

`widgets` JSON

The `widgets` section defines the widgets that appear in the dashboard. Each widget has a name.

Example:

```
"widgets": {
 "text_1": {
  "parameters": {
   "fontSize": 20,
   "text": "Grouping",
   "textAlignment": "center",
   "textColor": "#091A3E"
  },
  "type": "text"
 },
 "pillbox_1": {
  "parameters": {
   "compact": false,
   "exploreLink": false,
   "step": "StaticSAQLMinRanges"
  },
  "type": "pillbox"
 },
 "chart_1": {
  "parameters": {
   "autoFitMode": "fit",
   "showValues": true,
   "legend": {
    "showHeader": true,
    "show": true,
    "position": "right-top",
    "inside": false
   },
   "axisMode": "multi",
   "visualizationType": "hbar",
   "exploreLink": true,
   "title": {
    "label": "",
    "align": "center",
    "subtitleLabel": ""
   },
   "trellis": {
    "enable": false,
    "type": "x",
    "chartsPerLine": 4
   },
   "measureAxis2": {
    "showTitle": true,
    "showAxis": true,
    "title": ""
   },
   "measureAxis1": {
    "showTitle": true,
    "showAxis": true,
    "title": ""
   },
   "theme": "wave",
   "step": "Account_BillingCount_1",
```

```
   "dimensionAxis": {
    "showTitle": true,
    "showAxis": true,
    "title": ""
   }
  },
  "type": "chart"
 }
}
```

widget Properties

The `widgets` key defines all widgets that are available in the dashboard. It contains a separate node for each widget. Each widget appears in all layouts to which it's added. The properties available for each widget depend on the widget type. For example, a chart widget has the `legend` property, but a text widget doesn't.

`widget` Properties

The `widgets` key defines all widgets that are available in the dashboard. It contains a separate node for each widget. Each widget appears in all layouts to which it's added. The properties available for each widget depend on the widget type. For example, a chart widget has the `legend` property, but a text widget doesn't.

Property Name	Description
`parameters`	Widget parameters vary depending on the type of widget and, if applicable, type of chart. The `step` element defines the step attached to a widget. For detailed information about different widget parameters, see parameters Properties.
`type`	The widget type specifies one of the other supported widget types. The value of this field must be a string. • `chart` • `comparetable` • `container` • `dateselector` • `globalfilters` • `filterpanel` • `image` • `link` • `listselector` • `number` • `pillbox` • `rangeselector` • `table` • `text` • `valuestable`

parameters Properties

The `parameters` key contains a list of properties that control the appearance of the widget. Each widget type, including each chart type, contains a unique set of properties.

`parameters` Properties

The `parameters` key contains a list of properties that control the appearance of the widget. Each widget type, including each chart type, contains a unique set of properties.

Note: You can dynamically set properties for number and chart widgets in CRM Analytics dashboards based on the selection or results of another step. For example, you can change the map type in a chart based on a selection in a toggle widget. For more information, see the Analytics Bindings Developer Guide.

Chart widgets have many properties that vary based on the chart type. For chart-specific properties for CRM Analytics dashboards, see Visualizing Data with Charts.

The widget properties set in the `parameters` property are:

Property Name	Details
`absoluteModeEnabled`	**Type** Boolean **Available for This Widget** • `dateselector` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Show absolute dates in the date widget when users view the dashboard. `true`, by default. Default is `true`.
`alignmentX`	**Type** String **Available for This Widget** • `image` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates the horizontal alignment of the image in the widget. Valid values are: `left` (default), `center`, and `right`.
`alignmentX`	**Type** String **Available for This Widget** • `image`

Property Name	Details
	Exposed in the Dashboard Designer's User Interface Yes **Description** Indicates the horizontal alignment of the image in the widget. Valid values are: `left` (default), `center`, and `right`.
`alignmentY`	**Type** String **Available for This Widget** • `image` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates the vertical alignment of the image in the widget. Valid values are: `top` (default), `center`, and `bottom`.
`calendarTypeSwitchingAllowed`	**Type** Boolean **Available for This Widget** • `dateselector` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether the dashboard viewer can switch between the fiscal and calendar year in the date widget. Default is `false`.
`columns`	**Type** Array **Available for This Widget** • `chart` • `table` **Exposed in the Dashboard Designer's User Interface** Yes (for `aggregateflex` and `saql` step types only) **Description** List of API names of step fields to include in the widget. Use this property to include specific fields in the widget or to change the field order for the widget. This property appears when you reorder or hide a field while editing a widget that's based an `aggregateflex` (for a compare table) or a `saql` step type. To specify the fields

Property Name	Details
	or change their order for widgets based on other step types, you have to manually add and set this property in the JSON. Example: A step returns data in the following fields: Id, Name, Amount, and Profit. To hide the Id and Profit fields from the chart widget (chart_2), set the "columns" property. `"chart_2": { "parameters": { "columns": [ "Name", "Amount" ], "visualizationType": "hbar", "step": "Amount_Prob_1", ... }, "type": "chart", .... }`
`columnProperties`	**Type** Object **Available for This Widget** • table **Exposed in the Dashboard Designer's User Interface** Yes **Description** An object representation of the column properties. This includes the`type` enum with valid values of `bar`, `image`, and `text`. For each type, the `parameters` object specifies addtional attributes. **parameters (bar)** A bar column type. This type contains the following parameter: • `width` (integer) - The column's width. **parameters (image)** An image column type. This type contains the following parameters: • `alignment` (enum) - The column's width. • `height` (integer) - The image's height. • `width` (integer) - The column's width. **parameters (text)** A text column type. This type contains the following parameters: • `alignment` (enum) - The column's width. • `styles` (object) - The image's style, where `bold` is either true or false, and `linkColor` is the text color when the text is a link. • `width` (integer) - The column's width.

Property Name	Details

	Examples
	This example shows the `bar` type:
	`"columnProperties": [ { "type": "Bar", "parameters": { "width": 100 } } ]`
	This example shows the `image`type.
	`"columnProperties": [ { "type": "Image", "parameters": { "alignment": "Center|Left|Right", "height": 100, "width": 100 } } ]`
	This example shows the `text`type.
	`"columnProperties": [ { "type": "Text", "parameters": { "alignment": "Center|Left|Right", "styles": [ { "bold": false, "linkColor": "#0000FF" } ] "width": 100 } } ]`

`compact`	**Type** Boolean

Property Name	Details
	Available for These Widgets • `listselector` • `number` • `pillbox` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether displayed numbers are abbreviated (`true`) or not (`false`). For example, if `true`, the number 48,081 appears as 48k. Although the number appears to be rounded, it is not. The value 48,081 is preserved in charts and when performing calculations. If `false`, then 48,081 appears as 48,081. For number widgets, shortened numbers are rounded to the nearest 10th. Default is `false`.
`computeTotal`	**Type** Boolean **Available for These Widgets** • `chart` (only when `visualizationType` is `stackwaterfall` and `waterfall`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether to include the total measure column (`true`) or not (`false`). Default is `true`.

`containedWidgets`	**Type**
	List
	Available for This Widget
	• `container`
	Exposed in the Dashboard Designer's User Interface
	Yes
	Description
	A list of all widgets inside the container widget.
	Example
	This example shows 2 widgets (`meafilter_1` and `chart_1`) included in the container widget (`container_1`).
	`"container_1": {`
	`"type": "container",`
	`"position":{`
	`"x": 0,`

Property Name	Details

	<pre>"y": 0
	},
	"parameters":{
	"containedWidgets": [
	"meafilter_1",
	"chart_1"
	]
	}
	}</pre>

customBulkActions	**Type**
	Boolean
	Available for This Widget
	• table
	Exposed in the Dashboard Designer's User Interface
	Yes
	Description
	Specifies the following details about custom bulk action.
	label
	Display label for the button in the table widget's action menu. The dashboard viewer clicks the button to execute the action.
	visualforce
	The name and namespace prefix of the Visualforce page that defines the bulk action. Namespace prefix is optional.
	Example
	<pre>"customBulkActions":
	[
	{
	"label": "Generate Opportunities",
	"visualforce":
	{
	"name": "VF_Create_Opportunities",
	
	"namespace": "Prefix"
	}
	}
	]</pre>

defaultFiscalMode	**Type** Boolean **Available for This Widget** • dateselector **Exposed in the Dashboard Designer's User Interface** Yes

Property Name	Details
	Description Indicates whether the date widget displays dates using the fiscal year, by default. If not, then it uses the calendar year. Default is `false`.
`destination`	**Type** String **Available for This Widget** • `link` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The ID of the dashboard, lens, step, or page. Default is null.
`destinationLink`	**Type** String **Available for This Widget** • `link` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The object to which you are linking. One of the following properties identifies the linked object. **`name`** The ID of the dashboard, lens, step, or page. Default is null. **`url`** The URL that the link points to. **`tooltip`** The tooltip that displays when you hover over a link to a URL.
`destinationType`	**Type** String **Available for This Widget** • `link` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The destination type of a link. Possible values are:

Property Name	Details
	• `dashboard` — a dashboard • `explore` — a step • `lens` — a lens • `page` — a page • `url` — a website Default is `lens`.
`displayTemplate`	**Type** String **Available for This Widget** • `listselector` • `pillbox` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Specifies the string of grouping and measure fields to display in the widget. Fields must be enclosed in square brackets. By default, all groupings are included. **Example** This example displays the value for the Account.Type dimension, Account.BillingCountry dimension, and Amount measure. `"displayTemplate": "[Account.Type] - [Account.BillingCountry] ([avg_Amount])"`
`documentId`	**Type** String **Available for This Widget** • `image` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The 15-character document Id of the image file that you want to apply as the background. To ensure security, the image file must be uploaded to Salesforce as a document and the **Externally Available Image** option must be selected. If this option is not selected or the referenced document is not an image, the image doesn't show up in the widget. Default is null. **Example** This example image widget (`image_1`) displays an image with ID `015R0000000DC1P`. `"image_1": { "type": "image",`

Property Name	Details

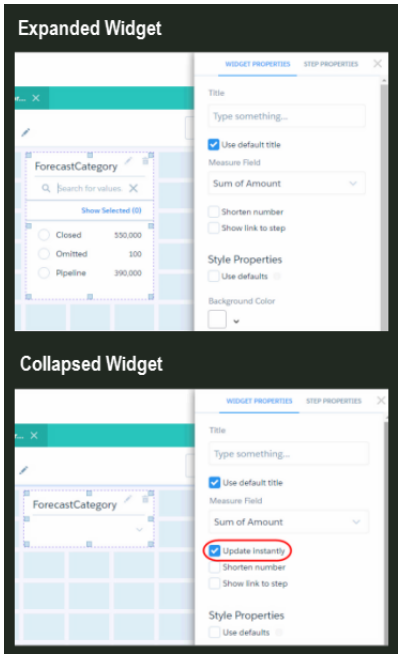
	<pre>"parameters": {
	"documentId": "015R0000000DClP",
	"fit": "stretch",
	"alignmentX": "center",
	"alignmentY": "center"
	}
	}</pre>

`dualAxis`	**Type** Boolean **Available for These Widgets** • `chart` (only when `visualizationType` is `combo`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether to include an axis for each of the two measures (`true`) or not (`false`). Default is `true`.
`exploreLink`	**Type** Boolean **Available for These Widgets** • `chart` • `comparetable` • `listselector` • `number` • `pillbox` • `valuestable` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether the widget shows the explore icon that dashboard viewers can click to explore the widget as a lens (`true`) or not (`false`). This option only applies to widgets based on steps in compact form. You can't explore widgets that are built on SAQL form steps. Default is `true`. Mobile devices display the icon, regardless of this setting.
`filterItemOptions`	**Type** List

Property Name	Details
	Available for This Widget • `filterpanel` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Details about each global filter shown in the global filter panel widget. You can configure the following details, which are also available in the user interface. **`propertyColor`** Color of the field name **`valueColor`** Color of the field value **`backgroundColor`** Color of the filter background **`borderColor`** Color of the filter border **`borderWidth`** Width of the filter border **`borderRadius`** Radius of the filter border
`fit` (for chart widgets)	**Type** Boolean **Available for This Widget** • `chart` (only when `visualizationType` is `scatter`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether the axis of a chart is in the center of the data (`true`) or at (0, 0) (`false`). Default is `false`.
`fit` (for image widgets)	**Type** String **Available for This Widget** • `image` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates how to scale the image. Valid values are: `original` (default), `stretch`, `tile`, `fitwidth`, and `fitheight`.

Property Name	Details
`fontSize`	**Type** Integer **Available for These Widgets** • `link` • `number` • `text` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The font size of a number or of text. Defaults are: • `number`: `36` • `text`: `26`
`hideHeaderColumn`	**Type** Boolean **Available for These Widgets** • `chart` • `valuestable` **Exposed in the Dashboard Designer's User Interface** No. Only editable via JSON. **Description** Indicates whether the first column in a raw data table—which is simply a count of rows—is hidden (`true`) or not (`false`). Default is `false`. This setting doesn't apply when viewing the widget on mobile devices.
`image`	**Type** Array **Available for This Widget** • `container` • `image` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Identifies the image using the following properties. **name** Name of the image.

Property Name	Details
	namespace Optional. Namespace of the image. Default is null. **Example** `"image": { "name": "My_Corporate_Logo", "namespace": "" }`
`includeState`	**Type** Boolean **Available for This Widget** • `link` **Exposed in the Dashboard Designer's User Interface** Yes **Description** If set to `true`: • CRM Analytics passes selections in chart, list, toggle, range, and date widgets as selections in the linked asset if they apply. For example, you select North America in a list widget based on the Region dataset field. CRM Analytics applies that same selection to faceted queries in the linked dashboard that have a grouping based on the Region field in the same dataset. • CRM Analytics passes global filters as filters to a linked asset as long as the filters apply. CRM Analytics ignores a global filter if the linked asset doesn't use the dataset that the global filter is defined on. If the global filter is locked and the incoming filter is defined on the same field, CRM Analytics ignores the incoming filter. If it's unlocked, the incoming filter overrides the global filter defined in the dashboard. Default is `false`.
`instant`	**Type** Boolean **Available for These Widgets** • `dateselector` • `listselector` • `rangeselector` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether other faceted widgets immediately update (`true`) or not (`false`) when a dashboard viewer makes a selection in this widget.

Property Name	Details
	When `false`, dashboard viewers must click **Update** for their changes to cascade to faceted widgets. When `true`, the **Update** button is hidden. Defaults are: • `dateselector`: `false` • `listselector`: `true` • `rangeselector`: `false` For list, range, or date widgets that are expanded in the dashboard designer, this widget property is always enabled. Selections in these widgets instantly update other widgets. While these widgets are expanded, you can't change this setting. 
`itemsPerRow`	**Type** Integer **Available for This Widget** • `filterpanel` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Number of global filters to show per row in the global filter panel. Default is `6`.
`legend`	**Type** Boolean

Property Name	Details
	Available for This Widget • `chart` (only when `visualizationType` is `hbar`, `vbar`, `stackhbar`, `stackvbar`, `pie`, `scatter`, `time`, `time-bar`, `time-combo`, `hdot`, `vdot`, `matrix`, `calheatmap`, `heatmap`, `parallelcoords`, `stackwaterfall`, `funnel`, or `choropleth`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether to display a legend (`true`), or not (`false`). Default is `false` for all chart types, except `pivottable`. Mobile devices can only display legends for `pie` widgets.
`legendHideHeader`	**Type** Boolean **Available for This Widget** • `chart` (only when `visualizationType` is `hbar`, `vbar`, `stackhbar`, `stackvbar`, `pie`, `scatter`, `time`, `time-bar`, `time-combo`, `hdot`, `vdot`, `matrix`, `calheatmap`, `heatmap`, `stackwaterfall`, `combo`, `combo`, or `parallelcoords`) **Exposed in the Dashboard Designer's User Interface** No. Only editable via JSON. **Description** Indicates whether the legend has a title (`true`) or not (`false`). The title is always the name of the dimension that the legend describes. Default is `false` for all chart types except `pivottable`. This setting doesn't apply when viewing the widget on mobile devices.
`legendWidth`	**Type** Integer **Available for This Widget** • `chart` (only when `visualizationType` is `hbar`, `vbar`, `stackhbar`, `stackvbar`, `pie`, `scatter`, `time`, `time-bar`, `time-combo`, `hdot`, `vdot`, `matrix`, `calheatmap`, `heatmap`, `stackwaterfall`, `combo`, or `parallelcoords`) **Exposed in the Dashboard Designer's User Interface** No. Only editable via JSON. **Description** The width of the legend area in pixels. Default is `145` for all chart types except `pivottable`. This setting doesn't apply when viewing the widget on mobile devices.

Property Name	Details
`measureField`	**Type** String **Available for These Widgets** • `listselector` • `number` • `pillbox` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The mathematical function performed on data. Specify the `measureField` in this format: *<formula>_<field>*. *<formula>* must match one of the formulas specified in the `measures` step property. Possible values for *<formula>* are: • `avg` — calculate the mathematical average (mean) • `max` — the maximum value • `min` — the minimum value • `sum` — add all the values • `unique` — count the number of unique values. For example, use to count the number of unique dimensions. The *<field>* paired with the *<formula>* must match the field name that is specified in `measures`. For example, if the `measures` step property is: `"measures": [ [ "sum", "Profit" ], [ "avg", "Discount" ], [ "count", "ModelNumber" ] ]` Then `measureField` must be `sum_Profit`, `avg_Discount`, or `unique_ModelNumber`. The `measureField` can't be `avg_Profit` because `avg` and `Profit` aren't paired together in the `measures` step property.

Property Name	Details
	Unlike for measures, a count on a dimension in the user interface calculates the number of unique dimension values. As a result, `measureField` in the underlying JSON shows the `unique` formula, like `unique_<dimension_field_name>`. Default is null.
`multiMetrics`	**Type** Boolean **Available for This Widget** • `chart` (only when `visualizationType` is `hbar` or `vbar`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether two or more measures are displayed as adjacent bars under each grouping (`true`) or as individual, adjacent graphs (`false`). Default is `false` (available only for bar charts and column charts).
`negativeColor`	**Type** String **Available for These Widgets** • `chart` (only when `visualizationType` is `waterfall`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** The color of the measure columns that have decreased in value in the chart. Specify the color in this format: `rgb(a, b, c, d)`. Using a number between zero and 255, ***a*** indicates how much red is in the color, ***b*** how much green, and ***c*** how much blue. A value of 0 indicates the absence of a color, and a value of 255 indicates the full expression of a color. Using a number between zero and one, ***d*** indicates the level of transparency. A value of `0` is invisible and `1` is opaque. For example, `rgb(0, 0, 0, 0.93)` sets the color to a nearly opaque black. `rgb(255, 0, 0, 0.14)` sets the color to a nearly invisible red. Alternatively, the color can be set using hexadecimal notation. When using hexadecimal notation, transparency can't be set. All hexadecimal colors default to opaque. `#000000` indicates black in hexadecimal. `#ff0000` indicates red.
`normalize`	**Type** Boolean

Property Name	Details
	Available for This Widget • `chart` (only when `visualizationType` is `stackhbar` or `stackvbar`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether charts are displayed using a logarithmic scale (`true`) or a linear scale (false). Default is `false` (available only for `stackhbar` and `stackvbar`).
`numberColor`	**Type** String **Available for This Widget** • `number` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The font color of the number. Specify the color in this format: `rgb(a, b, c, d)`. Using a number between zero and 255, ***a*** indicates how much red is in the color, ***b*** how much green, and ***c*** how much blue. A value of 0 indicates the absence of a color, and a value of 255 indicates the full expression of a color. Using a number between zero and one, ***d*** indicates the level of transparency. A value of `0` is invisible and `1` is opaque. For example, `rgb(0, 0, 0, 0.93)` sets the color to a nearly opaque black. `rgb(255, 0, 0, 0.14)` sets the color to a nearly invisible red. Alternatively, the color can be set using hexadecimal notation. When using hexadecimal notation, transparency can't be set. All hexadecimal colors default to opaque. `#000000` indicates black in hexadecimal. `#ff0000` indicates red. Default is `#000`.
`numberSize`	**Type** Integer **Available for This Widget** • `number` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The font size of the number. Default is 26.

Property Name	Details
`pivoted`	**Type** Boolean **Available for This Widget** • `table` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether the table is pivoted. A pivot table requires the underlying step to have at least one grouping. CRM Analytics pivots the table on the last defined grouping. Default is `false`.
`positiveColor`	**Type** String **Available for These Widgets** • `chart` (only when `visualizationType` is `waterfall`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** The color of the measure columns that have increased in value or remained the same in the chart. Specify the color in this format: `rgb(`***`a`***`, `***`b`***`, `***`c`***`, `***`d`***`)`. Using a number between zero and 255, ***`a`*** indicates how much red is in the color, ***`b`*** how much green, and ***`c`*** how much blue. A value of 0 indicates the absence of a color, and a value of 255 indicates the full expression of a color. Using a number between zero and one, ***`d`*** indicates the level of transparency. A value of `0` is invisible and `1` is opaque. For example, `rgb(0, 0, 0, 0.93)` sets the color to a nearly opaque black. `rgb(255, 0, 0, 0.14)` sets the color to a nearly invisible red. Alternatively, the color can be set using hexadecimal notation. When using hexadecimal notation, transparency can't be set. All hexadecimal colors default to opaque. `#000000` indicates black in hexadecimal. `#ff0000` indicates red.
`presetsEnabled`	**Type** Boolean **Available for This Widget** • `dateselector` **Exposed in the Dashboard Designer's User Interface** Yes

Property Name	Details
	Description Show preset dates in the date widget when users view the dashboard. `true`, by default. Default is `true`.
`relativeModeEnabled`	**Type** Boolean **Available for This Widget** • `dateselector` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Show relative dates in the date widget when users view the dashboard. `true`, by default. Default is `true`.
`showValues`	**Type** Boolean **Available for This Widget** • `chart` (only when `visualizationType` is `stackwaterfall` or `waterfall`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether to display the values of each measure column (`true`) or not (`false`). Default is `true`.
`splitAxis`	**Type** Boolean **Available for This Widget** • `chart` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether each dimension in a chart is measured on its own axis (`true`) or a shared axis (`false`). Only applicable when `multiMetrics` is `true`. Default is `false` (available only for bar charts and column charts).

Property Name	Details
	This setting doesn't apply when viewing the widget on mobile devices.
`sqrt`	**Type** Boolean **Available for This Widget** • `chart` (only when `visualizationType` is `parallelcoords`, `hdot`, `vdot`, `time`, `time-bar`, `time-combo`, `scatter`, `stackhbar`, `stackvbar`, `hbar`, `stackwaterfall`, or `vbar`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether charts are displayed using a logarithmic scale (`true`) or a linear scale (false). Default is `false` (available only for bar charts, column charts, line charts, and time series). This setting doesn't apply when viewing the widget on mobile devices.
`startColor`	**Type** String **Available for These Widgets** • `chart` (only when `visualizationType` is `waterfall`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** The color of the first measure column in the chart. Specify the color in this format: `rgb(a, b, c, d)`. Using a number between zero and 255, ***a*** indicates how much red is in the color, ***b*** how much green, and ***c*** how much blue. A value of 0 indicates the absence of a color, and a value of 255 indicates the full expression of a color. Using a number between zero and one, ***d*** indicates the level of transparency. A value of `0` is invisible and `1` is opaque. For example, `rgb(0, 0, 0, 0.93)` sets the color to a nearly opaque black. `rgb(255, 0, 0, 0.14)` sets the color to a nearly invisible red. Alternatively, the color can be set using hexadecimal notation. When using hexadecimal notation, transparency can't be set. All hexadecimal colors default to opaque. `#000000` indicates black in hexadecimal. `#ff0000` indicates red.
`step`	**Type** String

Property Name	Details
	Available for These Widgets • `chart` • `comparetable` • `dateselector` • `globalfilters` • `listselector` • `number` • `pillbox` • `rangeselector` • `valuestable` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The name of the step that supplies data for the widget. Default is null.
`stretch`	**Type** Boolean **Available for This Widget** • `box` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether an image's width and height are set to the same values of the widget's width and height (`true`) or not (`false`). Default is `false`.
`stretchImage`	**Type** Boolean **Available for This Widget** • `container` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether an image's width and height are set to the same values of the widget's width and height (`true`) or not (`false`). Default is `false`.

Property Name	Details
`text`	**Type** String **Available for This Widget** • `link` • `text` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The message rendered in a text widget. For example, if `text` is assigned the value `"Hello, World!"`, then "Hello, World!" appears in the text widget. Default is null.
`textAlignment`	**Type** String **Available for This Widget** • `link` • `number` • `text` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The alignment of text. Possible values include `left`, `center`, and `right`. If no value is specified, text alignment defaults to center. Defaults are: • `number`: `right` • `text`: `center`
`textColor`	**Type** String **Available for These Widgets** • `link` • `text` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The font color of text. Specify the color in this format: `rgb(a, b, c, d)`.

Property Name	Details
	Using a number between zero and 255, ***a*** indicates how much red is in the color, ***b*** how much green, and ***c*** how much blue. A value of 0 indicates the absence of a color, and a value of 255 indicates the full expression of a color. Using a number between zero and one, ***d*** indicates the level of transparency. A value of `0` is invisible and `1` is opaque. For example, `rgb(0, 0, 0, 0.93)` sets the color to a nearly opaque black. `rgb(255, 0, 0, 0.14)` sets the color to a nearly invisible red. Alternatively, the color can be set using hexadecimal notation. When using hexadecimal notation, transparency can't be set. All hexadecimal colors default to opaque. `#000000` indicates black in hexadecimal. `#ff0000` indicates red. Default is `#000`.
`title` (For widgets except global filter panel)	**Type** String **Available for These Widgets** • `dateselector` • `listselector` • `number` • `pillbox` • `rangeselector` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The title of a widget. Default is null.
`title` (for global filter panel widgets only)	**Type** List **Available for This Widget** • `filterpanel` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Details about the title of the global filter panel widget. You can configure the following details, which are also available in the user interface. **`visible`** Indicates whether the title is visible when viewing the dashboard **`text`** Details about the title: title (`label`), color of title (`color`), font size of title (`fontSize`), and horizontal alignment (`align`)

Property Name	Details
	`separatorColor` Color of the line that separates the widget title from the global filters
`titleColor`	**Type** String **Available for This Widget** • `number` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The font color of the title. Specify the color in this format: `rgb(a, b, c, d)`. Using a number between zero and 255, ***a*** indicates how much red is in the color, ***b*** how much green, and ***c*** how much blue. A value of 0 indicates the absence of a color, and a value of 255 indicates the full expression of a color. Using a number between zero and one, ***d*** indicates the level of transparency. A value of `0` is invisible and `1` is opaque. For example, `rgb(0, 0, 0, 0.93)` sets the color to a nearly opaque black. `rgb(255, 0, 0, 0.14)` sets the color to a nearly invisible red. Alternatively, the color can be set using hexadecimal notation. When using hexadecimal notation, transparency can't be set. All hexadecimal colors default to opaque. `#000000` indicates black in hexadecimal. `#ff0000` indicates red. Default is `#000`.
`titleSize`	**Type** Integer **Available for This Widget** • `number` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The font size of the title. Default is 26.
`tooltip`	**Type** String **Available for These Widgets** • `image` • `text`

Property Name	Details
	Exposed in the Dashboard Designer's User Interface Yes **Description** Text that appears when you hover over the widget while viewing the dashboard. Use a tooltip to add context to the text or image.
`totalColor`	**Type** String **Available for These Widgets** • `chart` (only when `visualizationType` is `waterfall`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** The color of the total measure column in the chart. Specify the color in this format: `rgb(a, b, c, d)`. Using a number between zero and 255, ***a*** indicates how much red is in the color, ***b*** how much green, and ***c*** how much blue. A value of 0 indicates the absence of a color, and a value of 255 indicates the full expression of a color. Using a number between zero and one, ***d*** indicates the level of transparency. A value of `0` is invisible and `1` is opaque. For example, `rgb(0, 0, 0, 0.93)` sets the color to a nearly opaque black. `rgb(255, 0, 0, 0.14)` sets the color to a nearly invisible red. Alternatively, the color can be set using hexadecimal notation. When using hexadecimal notation, transparency can't be set. All hexadecimal colors default to opaque. `#000000` indicates black in hexadecimal. `#ff0000` indicates red.
`totals`	**Type** Boolean **Available for These Widgets** • `chart` (only when `visualizationType` is `pivottable`) **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether to include a row that displays the sum of all the values in each measure column (`true`) or not (`false`). Default for `chart` is `false` (available only for `pivottable`). This setting doesn't apply when viewing the widget on mobile devices.
`trellis`	**Type** Boolean

Property Name	Details
	Available for This Widget • `chart` **Exposed in the Dashboard Designer's User Interface** Yes **Description** Indicates whether the last grouping displays on its own axis (`true`) if a step has multiple groupings and one measure. If `false`, it displays all groupings on the same axis. Default for `chart` is `false` (available only for bar charts and column charts). This setting doesn't apply when viewing the widget on mobile devices.
`videoSize`	**Type** String **Available for This Widget** • `url` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The dimensions of a YouTube video. Possible values are: • `(4/3) 240 x 180` • `(4/3) 420 x 315` • `(4/3) 480 x 360` • `(4/3) 640 x 480` • `(4/3) 960 x 720` • `(16/9) 320 x 180` • `(16/9) 560 x 315` • `(16/9) 640 x 360` • `(16/9) 853 x 480` • `(16/9) 1280 x 720` Default is `(4/3) 240 x 180`. Mobile devices don't display `url` widgets.
`visualizationType`	**Type** String **Available for These Widgets** • `chart` **Exposed in the Dashboard Designer's User Interface** Yes

Property Name	Details
	Description The type of chart used to show data. Possible values are: • `calheatmap`— calendar heat map • `choropleth` — choropleth (map) • `combo` — lines and bars to show multiple metrics • `flatgauge` — flat gauge • `funnel` — funnel • `hbar` — horizontal bar • `hdot` — horizontal dot plot • `heatmap`— heat map • `matrix`— matrix • `parallelcoords` — parallel coordinates • `pie` — donut • `pivottable` — pivot table • `polargauge` — polar gauge • `pyramid` — pyramid • `rating` — rating • `scatter` — scatter plot • `stackhbar` — stacked horizontal bar • `stackpyramid` — stacked pyramid • `stackvbar` — stacked vertical bar • `stackwaterfall` — stacked waterfall • `time` — timeline • `time-bar` — time bar • `time-combo` — time bar • `vbar` — vertical bar • `vdot` — vertical dot plot • `waterfall` — waterfall
`url`	**Type** ConnectUri **Available for This Widget** • `url` **Exposed in the Dashboard Designer's User Interface** Yes **Description** The URL of a YouTube video. Default is null.

Property Name	Details
	Mobile devices don't display `url` widgets.

`filters` JSON

The `filters` section defines the global filters added to a global filter panel widget, which is available in the dashboard designer.

Example: **Global Filters**

```
"filters": [
    {
        "label": "Account.Type",
        "fields": [
            "Account.Type"
        ],
        "operator": "in",
        "locked": false,
        "dataset": {
            "name": "opportunity"
        },
        "value": [
            "Customer"
        ]
    },
    {
        "label": "ForecastCategory",
        "fields": [
            "ForecastCategory"
        ],
        "operator": "in",
        "locked": true,
        "dataset": {
            "name": "opportunity"
        },
        "value": [
            "BestCase"
        ]
    }
],
```

filters Properties

The `filters` key defines all global filters included in the dashboard. It contains a separate node, with configurable properties, for each global filter. Global filters apply to all layouts, but you can specify whether each widget applies the global filters.

`filters` Properties

The `filters` key defines all global filters included in the dashboard. It contains a separate node, with configurable properties, for each global filter. Global filters apply to all layouts, but you can specify whether each widget applies the global filters.

Property Name	Description
`label`	Field label displayed in the global filter
`fields`	API name of the field by which to filter the records; a global filter can't be based on multiple fields
`operator`	Filter operator; supported operators are `in` (equals), `not in` (doesn't equal), and `matches` (contains)
`locked`	Prevents dashboard viewers from being able to change the global filter and for an incoming filter passed from a linked dashboard to overwrite the global filter
`dataset`	Contains the `name` field, which specifies the API name of the dataset that contains the filter field Example: `"dataset": { "name": "opportunity" },`
`value`	Array of field values used to filter the records

To view an example of the `filters` key configured with the properties in the table, see filters JSON. Each hash contains the configured properties for a single filter.