Ansys GRANTA MI 2021 R1

GRANTA MI Explore Configuration Guide

Table of contents

1 Introduction

This document is aimed at Ansys GRANTA MI administrators who need to configure Explore application data views for different groups of users.

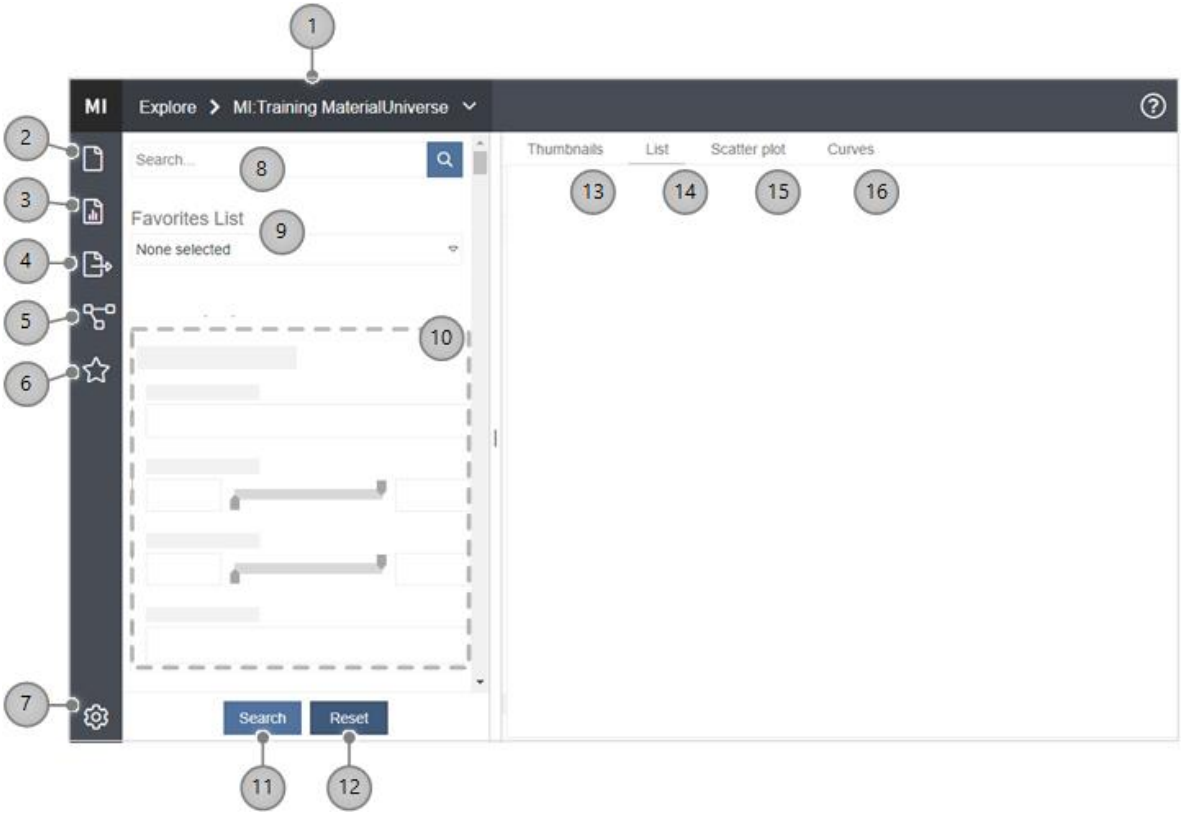
The Explore application is one of a number of integrated apps provided in the GRANTA MI enterprise web application ("One MI"). It enables users to find, visualize, and enter materials information in a GRANTA MI database, and quickly perform key tasks such as filtering, comparing, and visualizing data. The application is highly-configurable, allowing different data and feature sets to be presented to different user groups, ensuring that only the data and tools relevant to that user community are presented.

- Alternate application front-ends, known as "data views", defining the data and functionality available to users, are specified in one or more JSON-format application configuration files.
- Application-level preferences such as the unit system used are stored in **settings.app-preferences.json** in the GRANTA MI Settings Service.
- Some additional, global preferences which apply to the Explore app and other GRANTA MI applications are stored in **settings.app-preferences.json** in the Settings Service.

The code samples in this document used to show the effect of different configuration options were developed against the MI:Training database. This is a lightweight tutorial database with a small selection of Granta data, suitable for use in training classes and for users who wish to familiarize themselves with GRANTA MI features, and is available as database backup file in a standalone download package. The MI:Training database includes the example file shown in Section 9 as an embedded configuration.

2 Explore user interface

The application user interface includes the following features.



1 Choose a different data view
2 Create a new record
3 Run a report on the selected records
4 Export data for CAE
5 **Link Visualizer app** – graph visualization tool for viewing record links
6 **Favorites app –** manage personal Favorites lists, subscribe to a public Favorites list
7 **Preferences** – choose the application units and number display format
8 **Text search** – find all records that include the specified term
9 **Favorites List** – choose a Favorites list with the records you want to work with
10 **Search filters** – narrow down search results by applying filters based on data values
11 **Search**– click to get search results from the database ('data on-demand' mode only)
12 **Reset**– clear all search criteria
13 **Thumbnails view** – view search results displayed in a thumbnail gallery
14 **List view** – view search results as a list of records
15 **Scatter plot view** – view search results as a scatter (bubble) plot
16 **Curves view** – plot functional data curves from the data in the search results

3 Supported data types

The following GRANTA MI Attribute types can be used when searching, viewing, plotting, and editing data in the Explore app:

Data type	Attribute type	Search pane	List	Scatter plots	Curves	Datasheet view	Datasheet edit
Numerical	Integer	Y	Y	Y		Y	Y
	Point (single-value)	Y	Y	Y		Y	Y
	Point (multi-value)	Y	Y			Y	
	Range	Y	Y	Y		Y	Y
Text	Short Text	Y	Y			Y	Y
	Long Text	Y	Y			Y	Y
	Discrete	Y	Y			Y	Y
Functional	Float Functional (point)	Exists	Exists		Y	Exists	
	Float Functional (range)	Exists	Exists		Y	Exists	
	Equations & Logic						
	Discrete Functional						
Media	Picture					Y	Y
	Hyperlink	Exists	Y			Y	Y
	File (embedded media)	Exists	Y			Y	Y
Other	Date	Y	Y	Y		Y	Y
	Logical	Y	Y			Y	Y
	Tabular					Y	Y

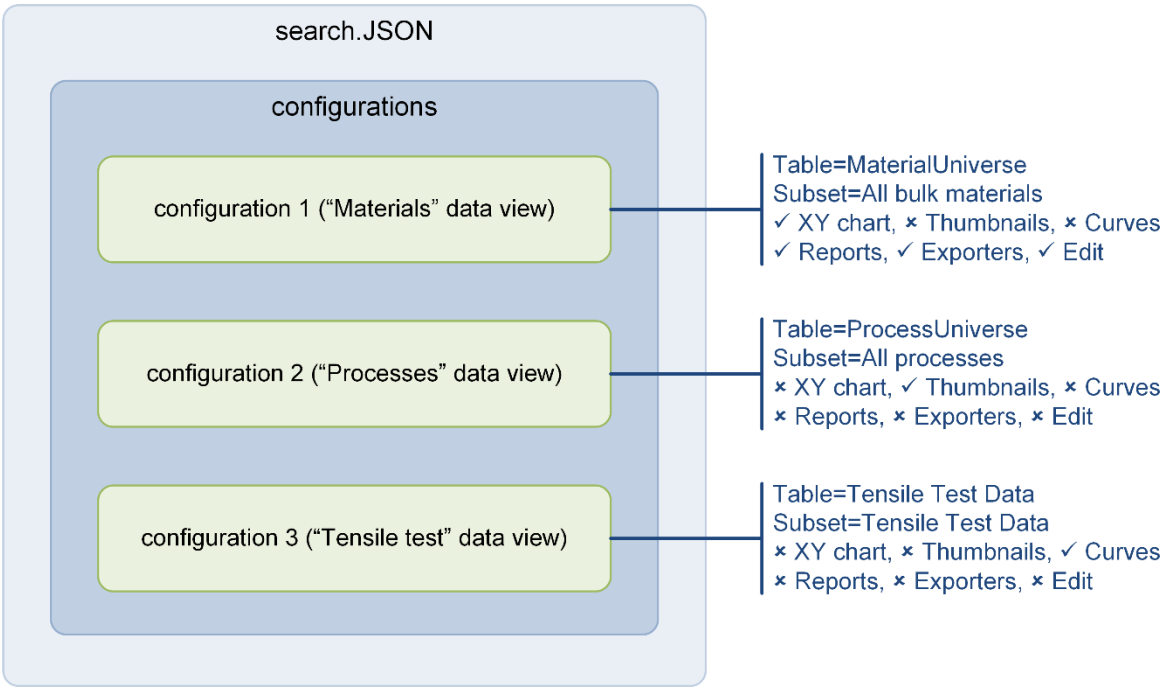
In this table:

- "Y" means users can search on, filter, plot, view, or edit values for this type of Attribute.
- "Exists" means users can search and filter on whether data exists for Attributes of this type, but not on the value; on datasheets, users can see that data exists, and view curves, but data points cannot be edited.
- A blank cell means that data of this type is not supported for the specified functionality.

4 Data view options

Alternate application front-ends, known as “data views”, define the data and functionality available to users.

Each data view is defined as a separate ‘configuration’ object in the search.JSON configuration file, which defines the available data and functionality, for example:



This section provides an overview of the supported data view configuration options, and describes how they affect the user experience for application users. See Section 8 for detailed reference information and syntax for the corresponding JSON settings. Many of these options can also be configured in the Switch data view UI within the Explore application; see Section 4.12.

4.1 Data view key

Each data view is identified by a unique name which is specified using the key property in the data view configuration.

The Explore application may be opened with a specific data view selected by including the key in a query string in the URL as follows:

http://<*your_mi_server*>/grantami/#/explore;searchConfigKey=*key*

For example:

http://acmeserver1/grantami/#/explore;searchConfigKey=training-materials

If the application is called without any search config specified in the URL, The Explore application will open with the default configuration, identified with `"default": true` ; in the data view config - see the first configuration in the example below.

This extract shows three configurations providing data views on three different Tables in the MI:Training database:

```
{
      "configurations": [
            {
                  "key": "training-materials",
                  "displayName" : "MI:Training Materials",
                  "default": true,
                  "databaseKey": "MI_Training",
                  "Table": "MaterialUniverse",
            ...
            },
            {
                  "key": "training-processes",
                  "displayName" : "MI:Training Processes",
                  "databaseKey": "MI_Training",
                  "Table": "ProcessUniverse",
            ...
            },
            {
                  "key": "training-tensiletest",
                  "displayName" : "MI:Training Tensile Test Data",
                  "databaseKey": "MI_Training",
                  "Table": "Tensile Test Data",
            ...
            }
}
```

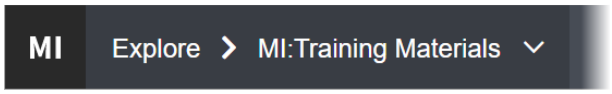
Note that this “default” flag will be overridden by global, role-specific default data view preferences specified in **settings.one-mi.json** in the Settings Service.

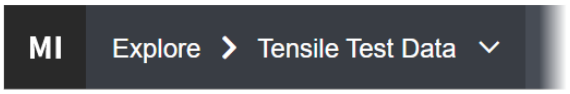
4.2 Data view name and description

The name of the data view is specified in the data view configuration with the displayName property. This name appears in the application header and in the **Switch to** list in the Switch data view dialog. For example:

```
{
      "configurations": [
            {
                  "key": "training-materials",
                  "displayName": "MI:Training Materials",
                  ...
            },
            {
                  "key": "training-tensile-test",
                  "displayName": "Tensile Test Data",
                  ...

            },
```





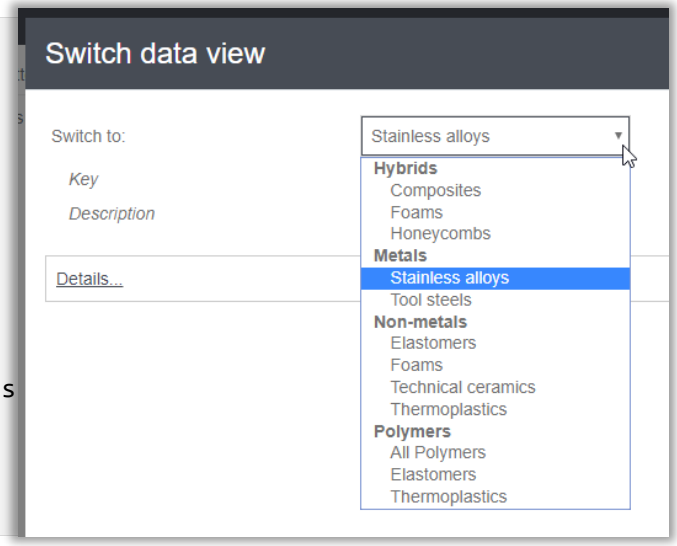
A description may also be defined to give users some additional information in the Switch data view dialog. This is specified in the data view configuration with the description property. For example:

```
{
      "configurations": [
            {
                  "key": "training-materials",
                  "description": "Explore tensile test data in the MI:Training
database. \nThe data does not load on startup; you must click the Search button
to load data.",
                  ...
            },
```

4.3 Grouping data views

Data views can be grouped to assist users when switching data views. Grouping is done using the groups property in the data view configuration. For example:

```
{
      "configurations": [
      {
            "key": "alloys",
            "displayName" : "Stainless
alloys",
            "groups": ["Metals"],
            ...
      },
      {
            "key": "thermoplastics",
            "displayName" : "Thermoplastics
            "groups": ["Polymers", "Non-
metals"],
            ...
      },
```



Each data view may appear in one group, many groups, or no groups. It is also possible to limit the data views available when switching views to only those within a certain group; see section *4.12*.

4.4 Database, Table, and Subset

The GRANTA MI database, Table, and Subset from which the Explore application will retrieve records are specified in the data view configuration using the databaseKey, table, and subset properties.

For example this *polymers* data view includes records from the *Polymers* subset in the *MaterialUniverse* Table in the MI:Training database:

```
{
      "configurations": [
      {
            "key": "polymers",
            "databaseKey": "MI_Training",
            "table": "MaterialUniverse",
            "subset": "Polymers",
            ...
            },
```

4.4.1 Database key

The database key only needs to be specified in the data view configuration if the configuration file is not stored in the database. See Managing configuration files.

4.4.2 Table

A Table must be specified in the data view configuration.

4.4.3 Subset

Specifying a Subset is optional. When a Subset is specified in the data view configuration, search and filter operations in the Explore application will be limited to only the records in this Subset. If no Subset is specified, search and filter operations will be applied to all records in the Table.

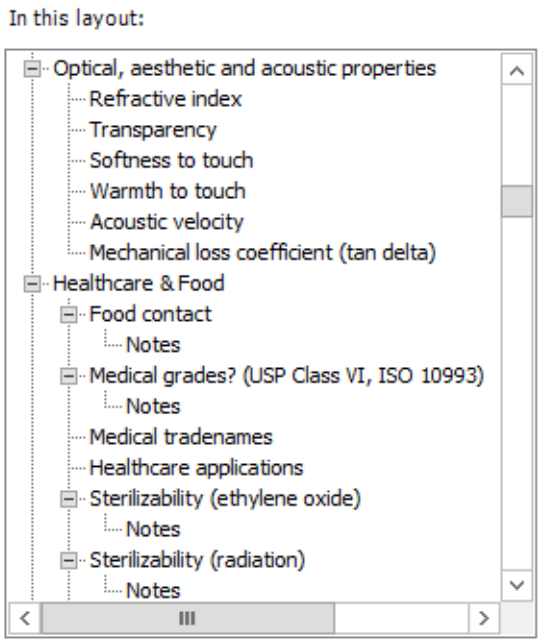
Using a Subset allows you to narrow down the set of records presented in Explore to only records that are relevant to users of that data view. For example, the above configuration defines the data of interest as only records in the *Polymers* Subset.

Where users have the ability to add records (see Section *4.12*), then all new records will be placed in the specified Subset.

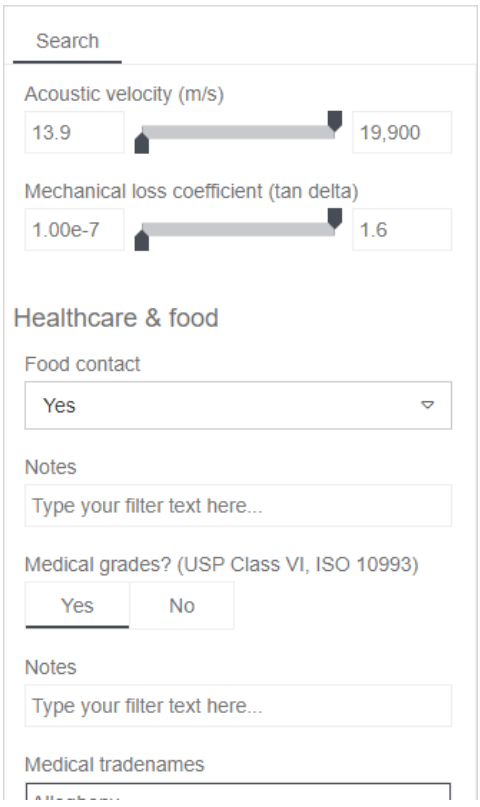
4.5 Search panel

The Attributes available in the Search panel in Explore, and the headings under which they are grouped, are determined by the Layout specified in the data view configuration with the searchLayout property.

Layout definition (in MI:Admin tool)



Corresponding Headings and Attributes in the Search panel

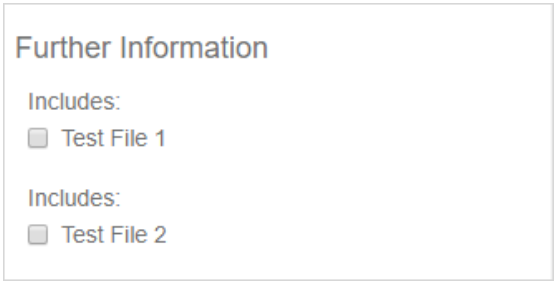


4.5.1 Filter controls in the Search panel

Different filtering controls will be available in the Search panel, depending on the type of the Attributes included in the search Layout. For example, there may be text fields for Short and Long Text Attributes, sliders for range Attributes, lists for Discrete Attributes, and buttons for Logical Attributes.

Not all Attributes in the Layout may be available for use as search filter criteria:

- Date, Discrete, Integer, Logical, Long text, Point, Range, Short text, and Attributes can all appear in the Search panel, and their values can be used as filter criteria.
- Functional (point and range), File, and Hyperlink Attributes will appear in the Search panel, but their values cannot be used as filter criteria. However, users can filter on whether or not data exists for that Attribute. For example, selecting the check box for the *Test File 1* or *Test File 2* File Attributes here means "Only show records where there is some data for this Attribute", i.e. records which include a test file:



- Equations and Logic, Discrete Functional, Picture, and Tabular Attributes cannot be used a search /filter criteria and will simply not appear in the Search panel.

See Section *3, Supported data types* for a table summarizing which Attribute types are supported in the Search pane, in Datasheets, and in the different tabs.

> **Note:** Data view load time will be affected if the search Layout includes a large number of Attributes, because data for each Attribute has to be fetched from the database. To optimize startup time, you may wish to create multiple data views configurations for the same database and Table, each with a smaller number of targeted search Attributes.

Logarithmic sliders

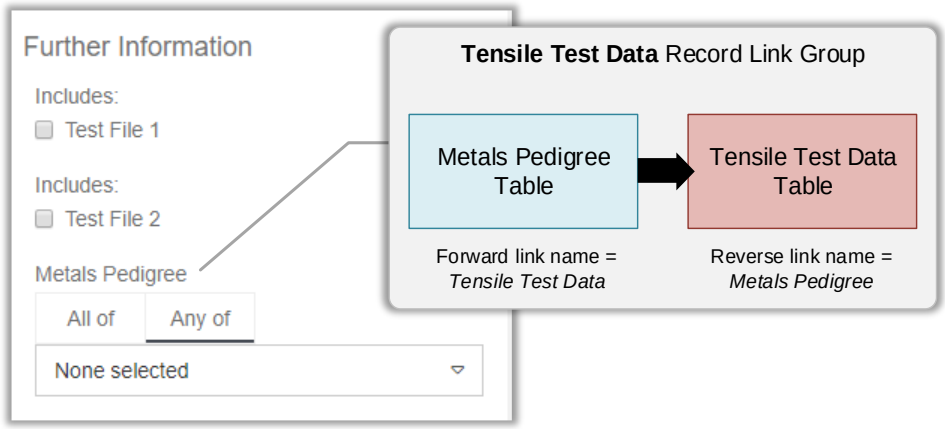
Most material properties extend over several orders of magnitude, so logarithmic scales can be used on slider controls in the Explore application. This feature is enabled in the data view configuration with the logSliders property in the data view configuration.

4.5.2 Filtering by linked records

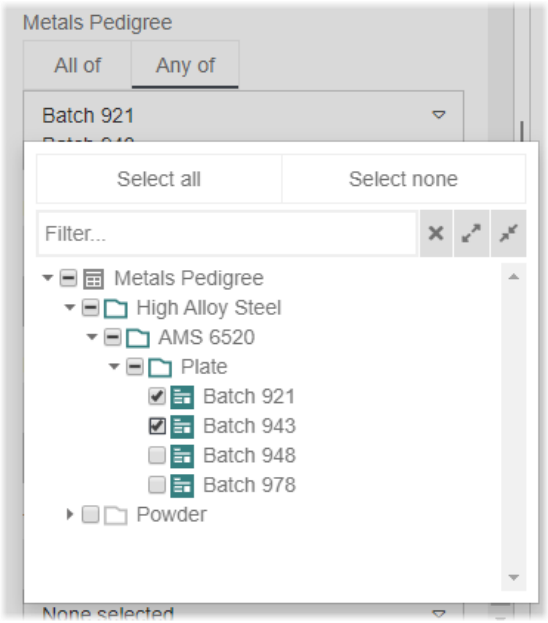
Each data view is constrained to show records from only one database, Table, and Subset. Where the search Layout includes a Record Link Group which provides links to records in other Tables, the Search panel will include a control that allows users to filter records based on whether or not they are linked to records in the other Table.

For example, the MI:Training database includes a Record Link Group called Tensile Test Data, which provides links from records in the *Tensile Test Data* Table to related records in the *Metals Pedigree* Table. In the example shown below, the search Layout includes this Record Link Group, and therefore

the Search panel includes a control which allows users to filter records based on whether or not they are linked to selected records in the linked *Metals Pedigree* Table.



To show only tensile test data records with certain batch properties, users can then browse the linked Table and select the relevant linked records, for example:



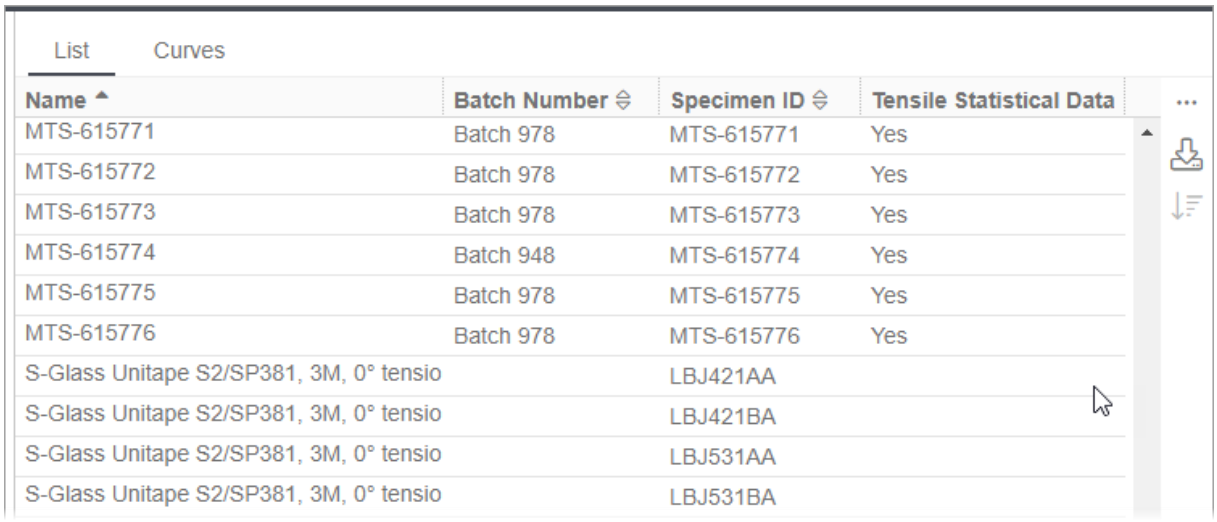
4.6 List view

On the List view, search results are displayed in a list; each row is a record, and the columns show the record name and/or data values.

Users can add columns to show additional data by clicking on the Add columns button ⋯, and also re-order columns by dragging them. Any of the Attributes in the search Layout can be added as columns. Changes made by users to the visible columns will persist between the Explore application sessions.

Record Links Groups in the search Layout can be added as columns in the List view, enabling users to see whether or not there is related data in another Table. For example, in the MI:Training database,

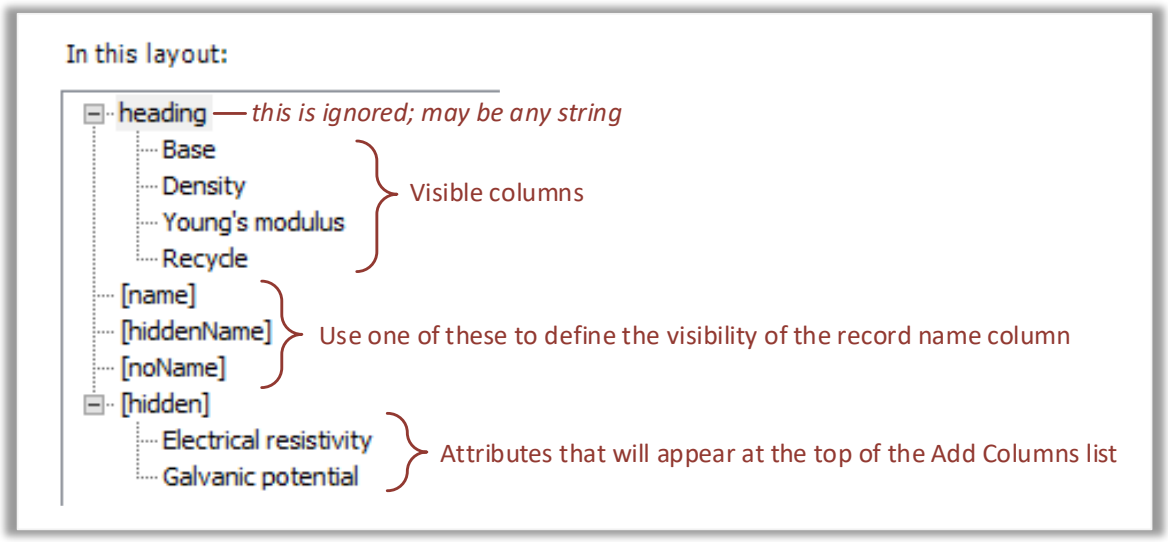
the *Tensile Test Data* Record Link Group enables links between records in the *Tensile Test Data* and *Statistical Test Data* Tables. In the picture below, showing search results from the *Tensile Test Data* Table, you can see that the Record Link Group appears as the rightmost column: a "Yes" in this column indicates that the record is linked to a record in the other Table:



By default, the only column initially visible in the List view is the record name, and user can add more columns for data they want to see here.

A specially-configured Layout – specified in the data view configuration with the searchListLayout property – can be used to define the columns that are initially shown on the List view as well as/instead of the record name, and in what order they appear by default. The searchListLayout can also be used to specify which Attributes should appear at the top of the 'Add Columns' list, to make it quicker to add them.

The *search list Layout* definition in the MI:Admin Schema tool should include the required columns (Attributes and Record Link Groups), and may also include some special headings that control the visibility/position of the record name column, and the position of Attributes in the Add Columns list:



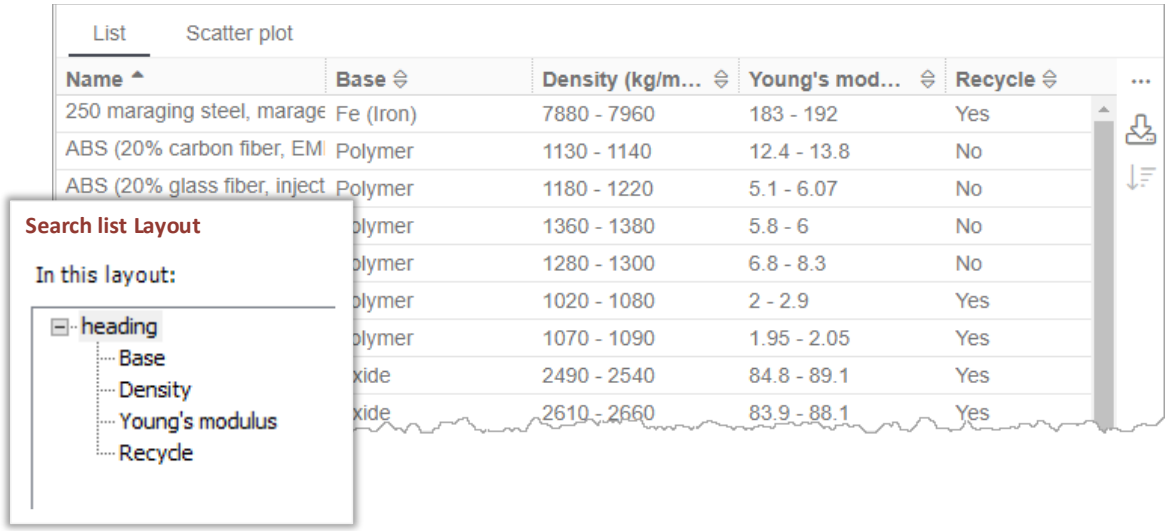
The Layout Editor requires that Attributes in the Layout must be under a Layout Heading. However, apart from the special headings listed below, the Explore application ignores headings in the *search list Layout*, and only looks at the Attributes and Record Link Groups listed. These special headings must entered in the Layout Editor exactly as shown, including the square brackets.

Layout Heading	Behavior
[name]	Use this heading to determine the position of the Record name column. By default, the Record name will appear as the first (leftmost) column. This heading can be used to position the Record name column after other columns. See Example 2.
[hiddenName]	Specifies that the Record name column will not be visible in the List view by default; it will appear at the top of the Add Columns list, and so users can see it if they want to. See Example 3.
[noName]	Specifies that the Record name column will not be visible in the List view by default, and will not appear in the Add Columns list, so users can never see it or add it the List view. See Example 3.
[hidden]	Placing Attributes under a [hidden] heading in the Layout means they will not be shown in the List view by default, but they will appear at the top of the Add Columns list, making it easier to add them. See Example 4.

Some additional styling of columns is possible using data formatting elements in the data view configuration, for example, you can set the column width, text alignment, specify whether or not the column header is shown, and replace the data values with images; see section *5.1, List view*.

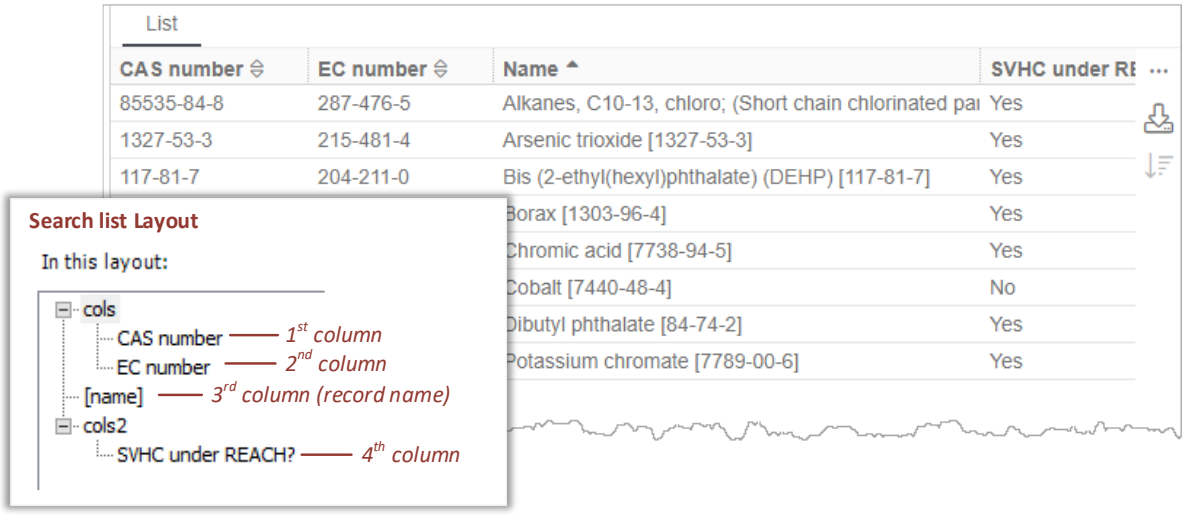
4.6.1 Example 1: specifying columns

Columns showing the values for the *Base*, *Price*, *Density*, *Young's Modulus*, and *Recycle* Attributes are shown here, in addition to the record name column which is the first column by default:



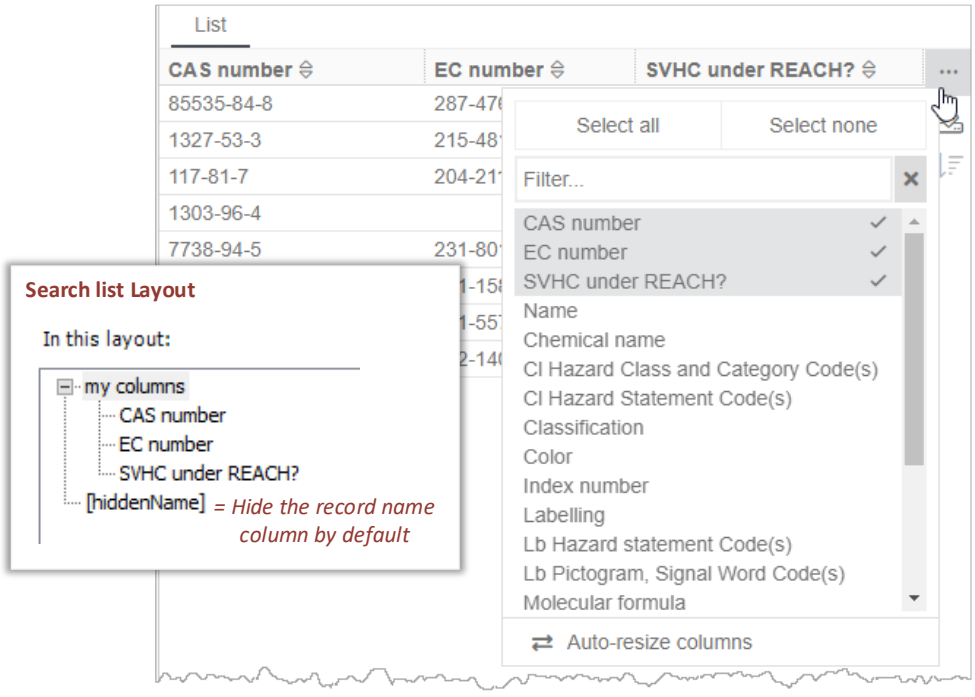
4.6.2 Example 2: moving the record name column

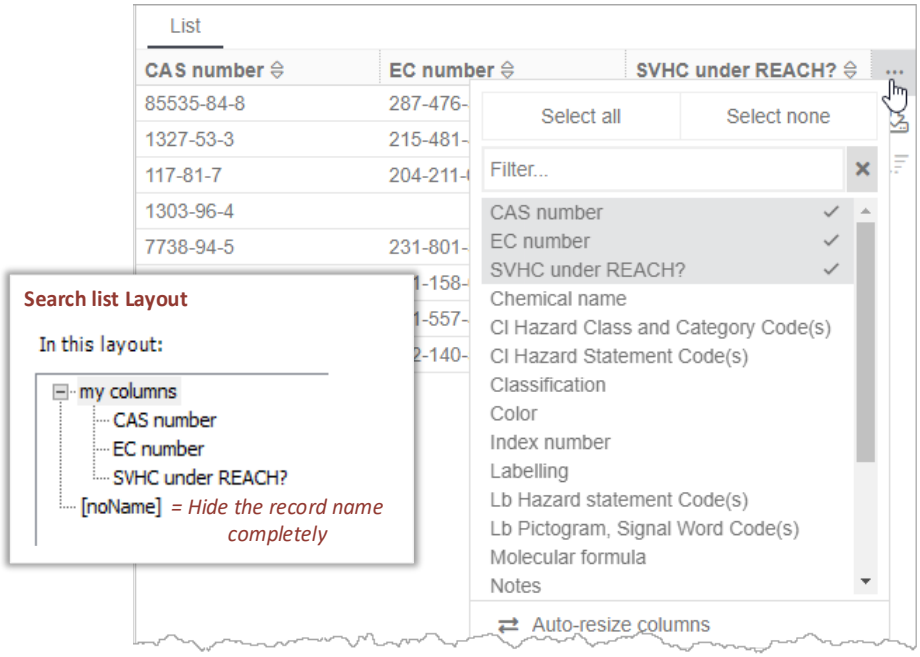
This Layout definition specifies 4 columns, with record name as the third column:



4.6.3 Example 3: hiding the record name column

If users typically identify records in the data view using an Attribute that provides a unique identifier, such as Specimen ID, part number, CAS number, or similar instead of the record name, the record name column may be hidden. Users then may or may not be able to add it from the Add Columns list, depending on the Layout heading configuration key (**[hiddenName]** or **[noName]**). For example:

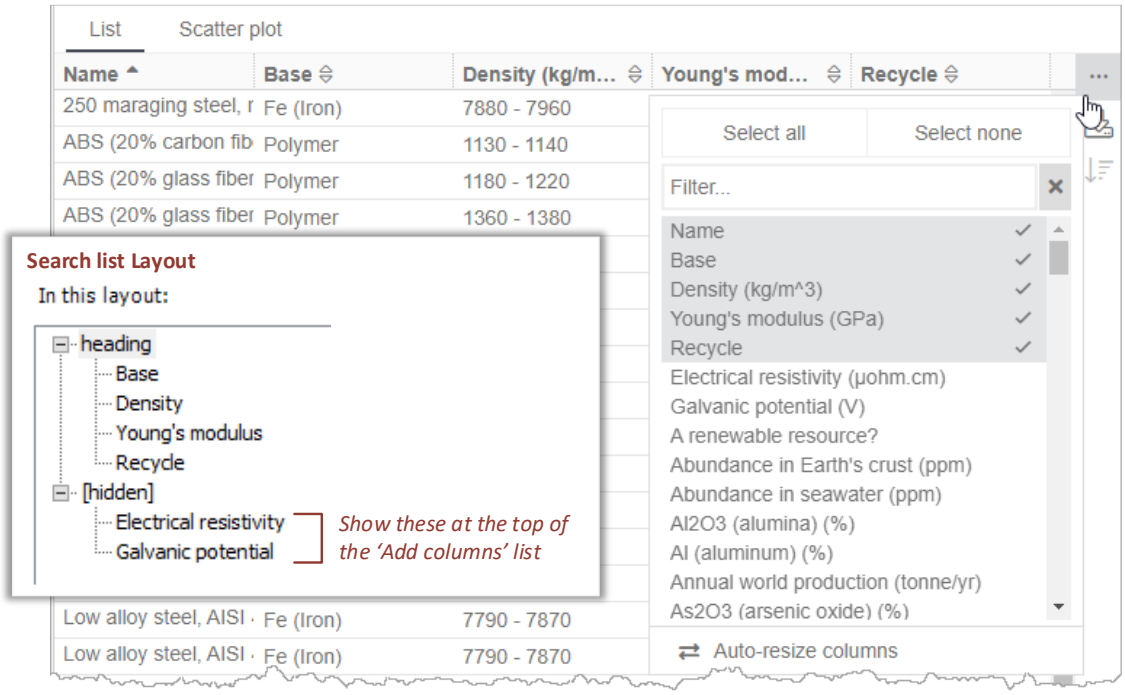




Note that the list will always be sorted by the record name by default, even if the record name column is hidden; however, users can click on any column header to sort the list by that column.

4.6.4 Example 4: additional columns

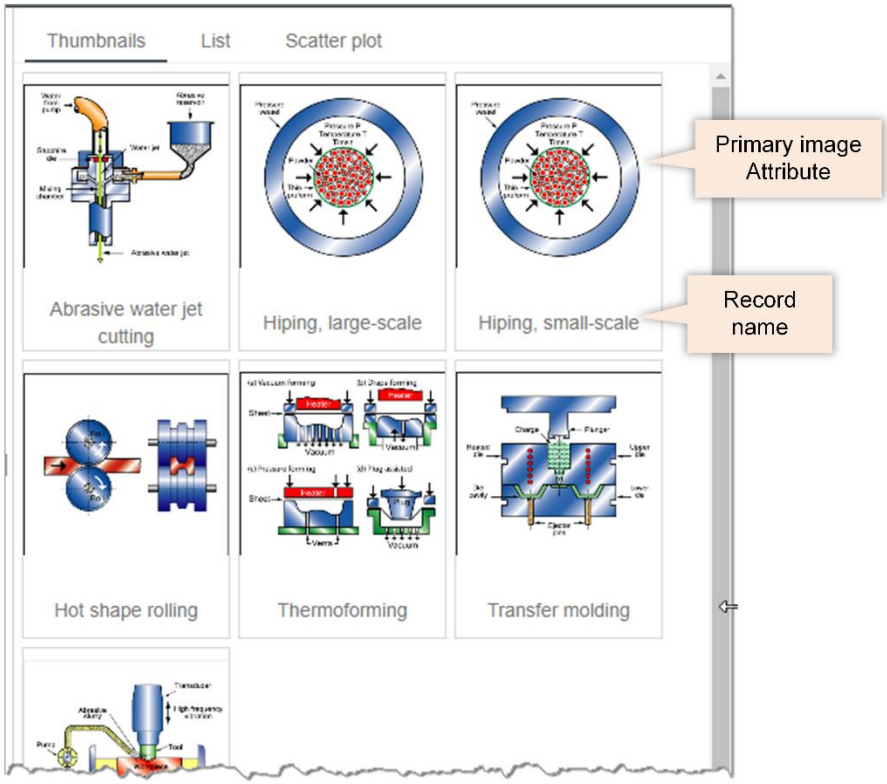
To specify the Attributes that should be shown at the top of the list when users add more columns, making them easier to find when the search Layout includes many Attributes, place the Attributes under a **[hidden]** heading in the search list Layout. For example:



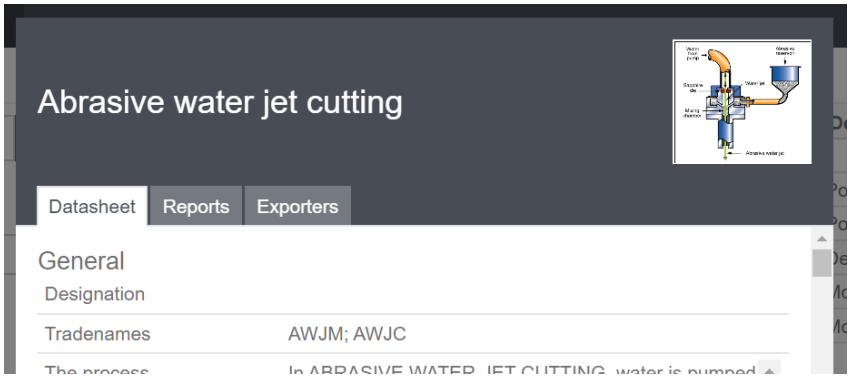
4.7 Thumbnails view

The optional Thumbnails view can display search results as a thumbnail gallery of images.

The Picture Attribute that contains the images on the Thumbnails view is specified in the data view configuration using the primaryImageAttributeName property.



This image is also shown on the datasheet header, for example:



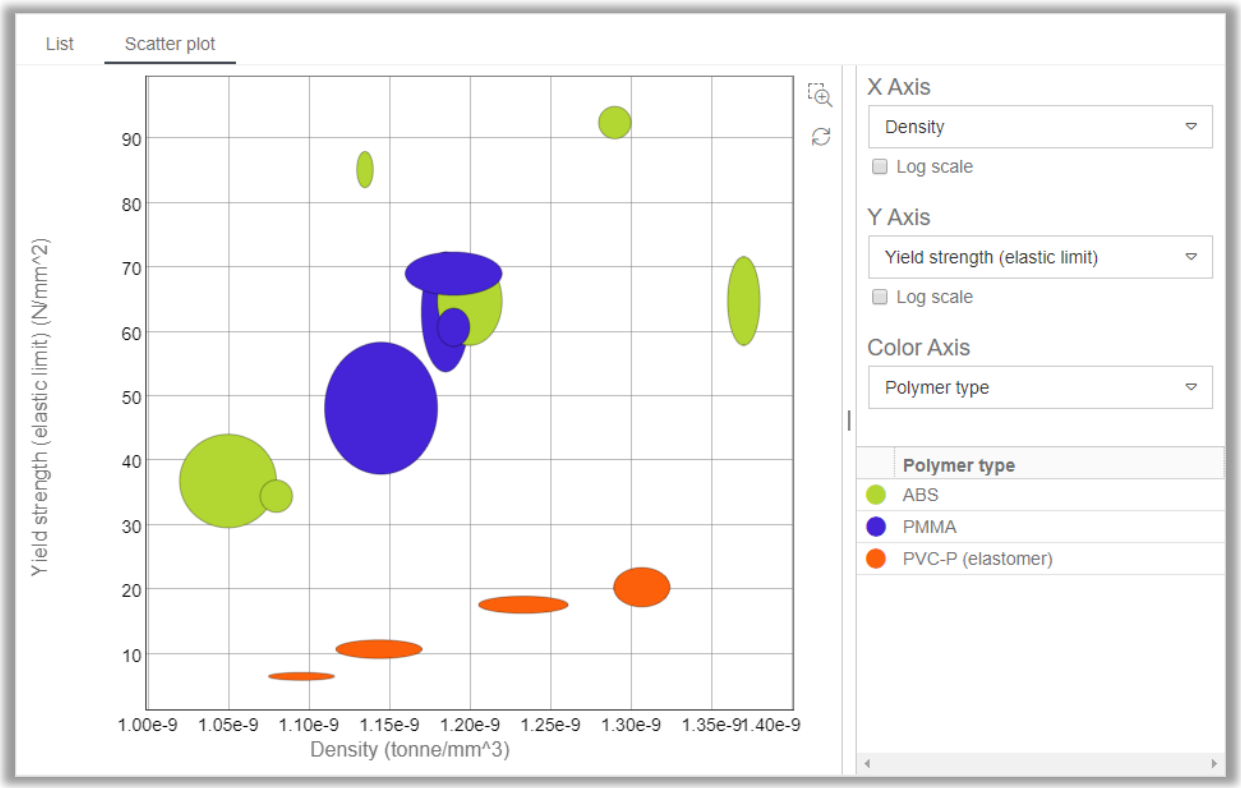
Example:

```
"primaryImageAttributeName": "Process schematic",
```

If no primary image Attribute is specified, then the Thumbnails view will not appear in the application.

4.8 Scatter plots view

Search results can be displayed on a scatter (XY, bubble) plot, for example:



Options for scatter plots are configured with the xyChart property in the data view configuration. You can:

- Enable/disable the Scatter plot view
- Specify the default Attributes used for the x- and y-axes (`xAttribute` and `yAttribute` properties).
- Specify the default scale (linear or logarithmic) used for the x- and y-axes (`AxisLogarithmic` and `yAxisLogarithmic` properties).
- Enable/disable the ability for users to change the plot axes, for example, to choose different Attributes or to change the axis scales (`preventAxisChange` property).
- Where custom data formatting has been configured to enable a Color Axis, the default Color Axis Attribute may also be specified (`colorAttribute` property).

The Attributes that may be used for the plot axes must be included in the data view search Layout, and must be of the following type:

- **x- and y- Axes**: Integer, Range, single-value Point, Date
- **Color Axis**: Logical, Discrete, Integer, Point, Range. % is the only permitted unit.

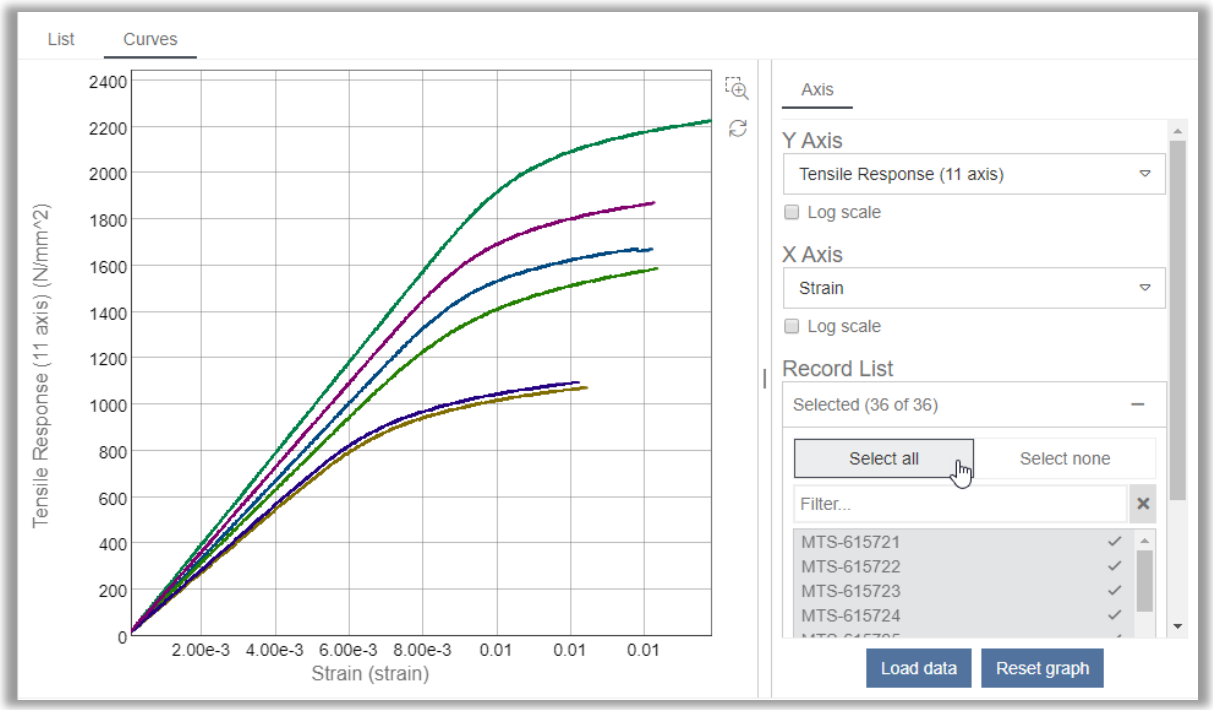
Example:

```
"xyChart": {
      "colorAttribute": "Polymer type",
      "preventAxisChange": false,
      "xAxisLogarithmic": false,
      "yAxisLogarithmic": false,
      "xAttribute": "Density",
      "yAttribute": "Yield strength (elastic limit)"
      }
```

The specified Color Axis Attribute must be available in the search Layout, and must be mapped to a custom formatter object as described in Section *5.4, Custom formatting* .

4.9 Curves view

Where records include functional data, curves can be plotted on the Curves view. For example:



Users can plot curves from multiple records, and may also be able to:

- choose the y-axis Attribute and the x-axis parameter
- select the axis scales (linear or log, if applicable)
- adjust the parameter values to show more or less data on the plot.

The Attributes available for plotting are any series functional Attributes included in the search Layout; gridded functional data cannot be plotted. Point data and Range data can be plotted.

Options for curves are specified with the `curves` property in the data view configuration, and include:

- Enabling/disabling the Curves view
- Specifying the default y-axis Attribute (`yAttribute`) and x-axis parameter (`xParameter`).
- Specifying the default scale (linear or logarithmic) for the x- and y-axes (`xAxisLogarithmic`, `yAxisLogarithmic`).

Example:

```
"curves": {
      "yAttribute": "Tensile Response (11 axis)",
      "xParameter": "Strain",
      "xAxisLogarithmic": false,
      "yAxisLogarithmic": true
      }
```

4.10 Link Visualizer

Link Visualizer is a graph visualization tool for viewing links between records. When enabled, the Link Visualizer icon  appears on the application toolbar. Users can select one or more records in the Explore app, and then open Link Visualizer to view links to other records and build up a graphic visualization that shows the relationships between records.

> **Note:** Cross-database links can only be viewed or navigated in one direction, from the originating database to the destination database.

Options are specified using the linkVisualization property in the data view configuration, and include:

- Enabling or disabling Link Visualizer
- Specifying a default *Link Visualizer configuration* for this data view
- Specifying a list of Link Visualizer configurations that will be available for users to choose from.

Link Visualizer configurations specify the layout and styling of the visualization, and are defined in **settings.link-visualizer.json** in the Settings Service; see Section *6.3, Link Visualizer settings*.

The default settings defined in **settings.link-visualizer.json** provide two pre-defined Link Visualizer configurations, which you can make available to users as required:

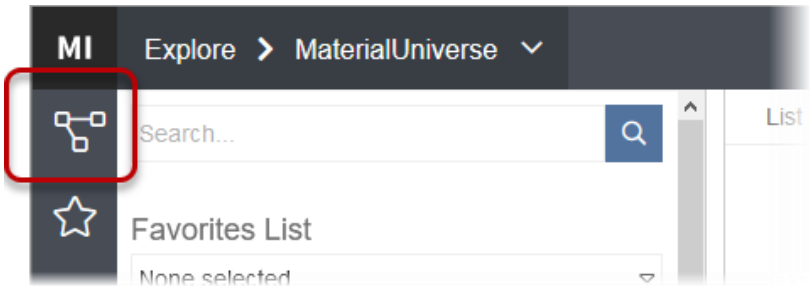
- *Configuration A (Hierarchical view)* shows linked records in a structured format, with the starting records at the left. This view is useful for displaying simple data structures with linear relationships. It is not recommended for data structures that may contain circular link paths.
- *Configuration B (Organic view)* distributes the linked records across the visualization, with the starting records near the center. This view is useful for displaying complex data structures.

You can define additional configurations, if required.

To enable the Link Visualizer in a data view, set `linkVisualization` to *true*:

```
"linkVisualization": true,
```

This adds the Link Visualizer button to the Explore application toolbar.



Note that this does not restrict the Link Visualizer configurations that are available, or specify a default. If there is more than one configuration defined in **settings.link-visualizer.json**, the first one listed will be used by default, and users will be able to select from any of the configurations.

To set a configuration to be the default, specify it by name, as a string. For example:

```
linkVisualization": config2,
```

This sets *config2* as the default configuration. Users can still select any of the other configurations.

To limit the configurations that are available to users, specify one or more configurations as an array. For example:

```
linkVisualization": [config3, config2],
```

Users can select from the named configurations only, and the first one in the array (config3) will be used by default.

4.11 Datasheets

The content and functionality available in datasheets in the Explore application is defined in the data view configuration. Options include:

- Specifying the Layout that defines the headings and Attributes in datasheets; see section *4.11.1* below.
- Specifying whether or not Attributes that have no value should be shown in datasheets; see section *4.11.2* below.
- Specifying whether or not users can edit the data. See Section *4.13*.
- Specifying styling (text and background color, alignment, images) for specific Attributes; see Section *5, Data formatting options*.

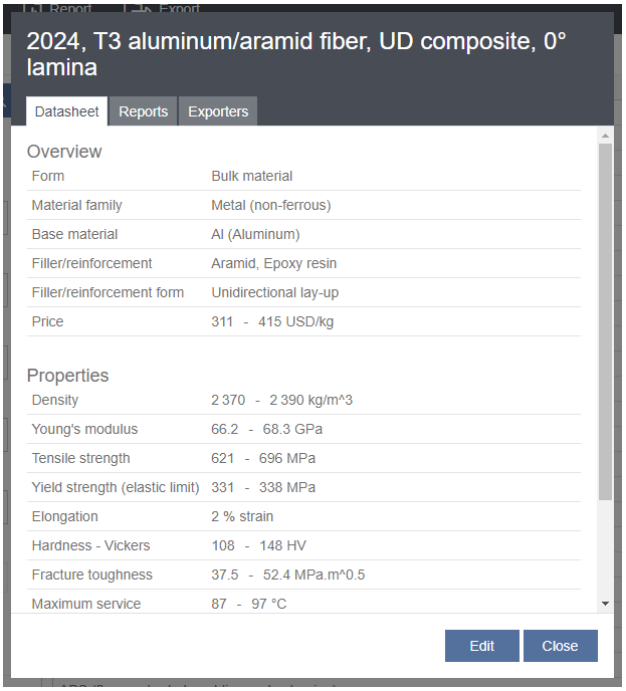
4.11.1 Layouts – datasheet Attributes and headings

Each data view configuration must include a `dataSheetLayout` property which specifies a Layout that defines the headings and Attributes shown when viewing datasheets in the Explore application.

If users will need to edit data, the data view configuration must also include an `editableDatasheetLayout` property, which specifies a Layout that defines the headings and Attributes shown when adding or editing data.

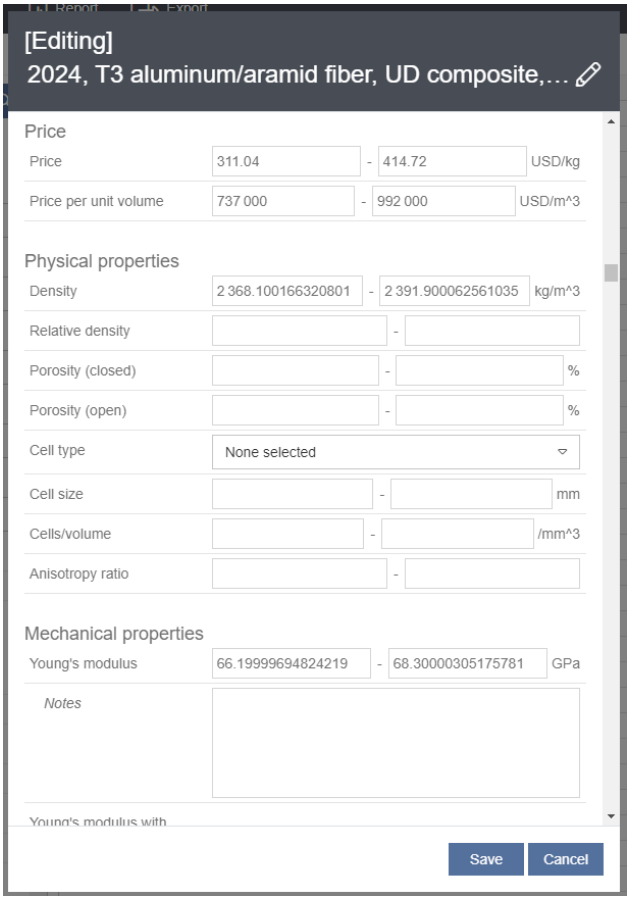
The same Layout may be used for viewing and editing data, or different Layouts may be used. For example, using a Layout with fewer Attributes may make it easier for users viewing the data, while Edit users may expect to more detailed data, with additional Attributes.

Examples



`"dataSheetLayout": "Explore search"`

Here, a record in the MaterialUniverse Table is viewed using the *Explore search* Layout which includes a subset of the Attributes defined in the Table, organized under two headings, *Overview* and *Properties*.



`"editableDataSheetLayout": "All attributes`

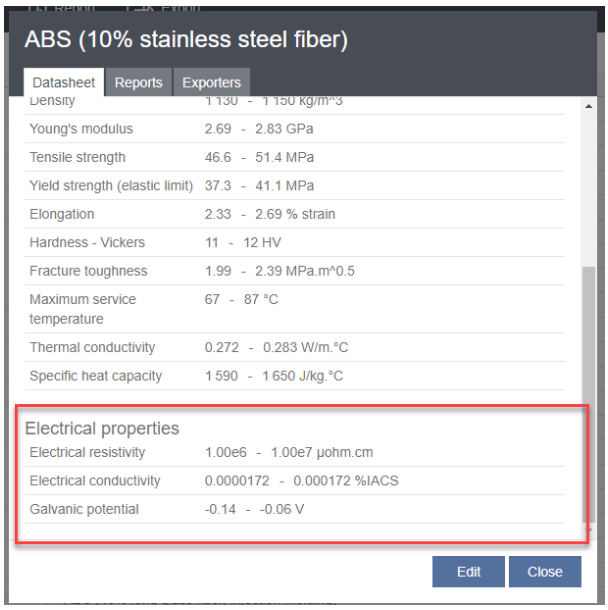
The Layout used for editing the same data includes all of the Attributes defined in the Table, grouped under different headings (compare with the headings and Attributes visible when viewing the datasheet, above).

4.11.2 Showing/hiding Attributes with no value

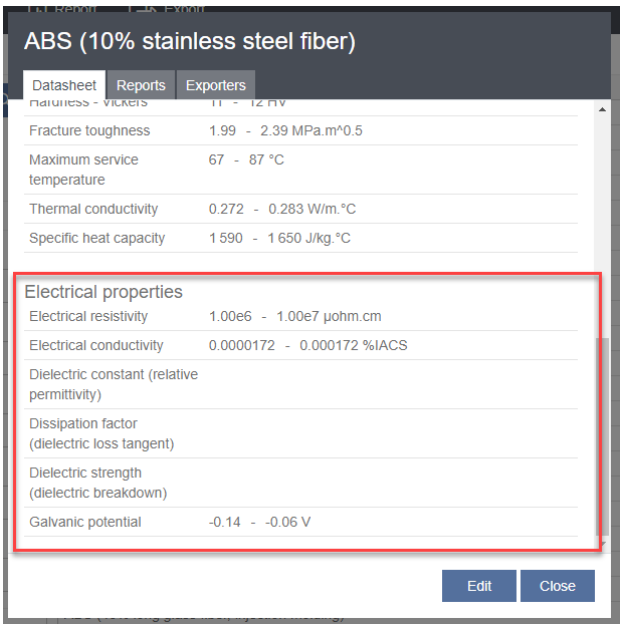
The `showBlanks` setting controls whether Attributes with no value are shown or hidden when datasheets are viewed in the Explore application.

When `false`, Attributes with no data value are hidden when viewing datasheets; when `true`, empty Attributes are shown.

`"showBlanks": false,`



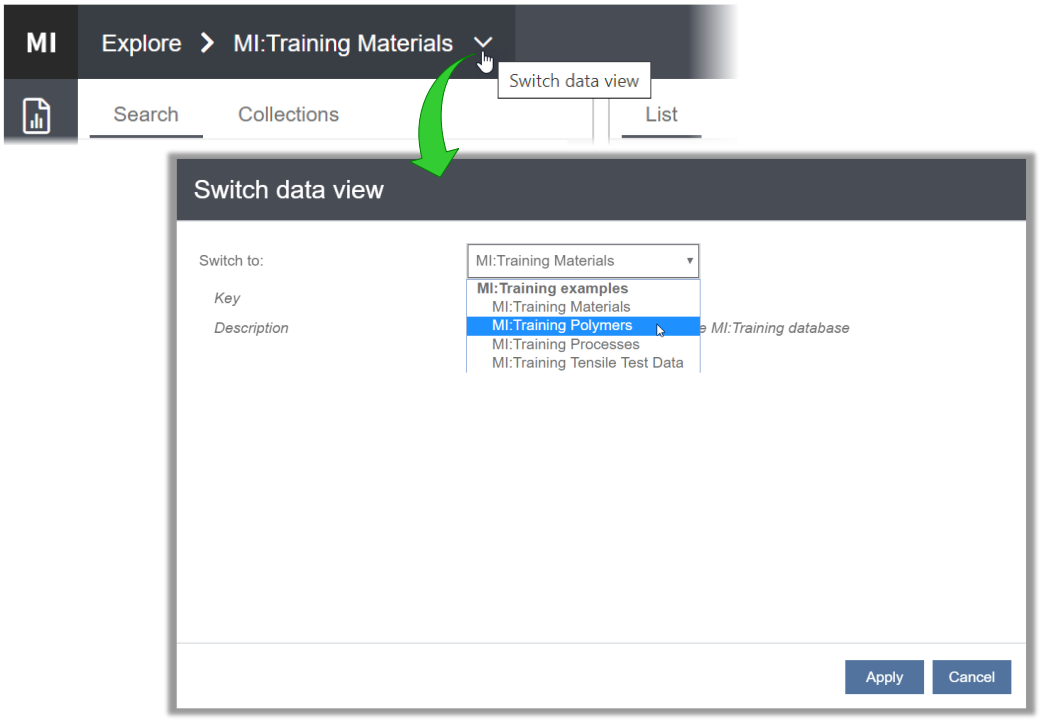
`"showBlanks": true,`



Note that the `showBlanks` setting only applies when *viewing* datasheets; when *editing* datasheets, all Attributes in the Layout are always shown.

4.12 Data view switch functionality

The Switch data view window in the Explore application allows users to choose different data views, where more than one data view is configured.



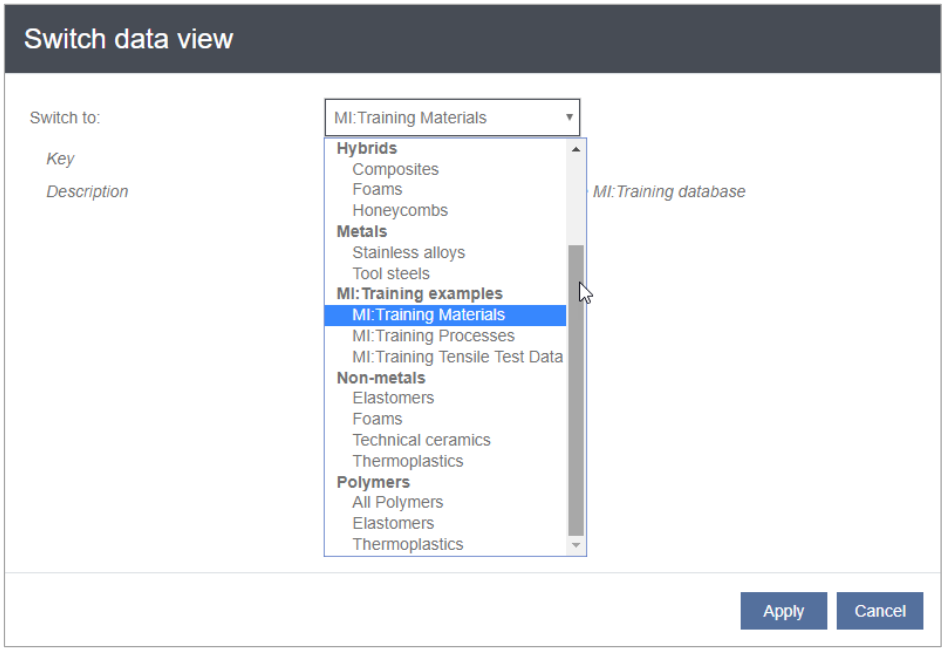
You can determine how much data view switching functionality, if any, is available to users with the `configSwitching` property in the data view configuration. By default, all config switching functionality is enabled. Five different levels of data switch functionality may be configured with the `configSwitching` property:

	Users can…		
configSwitching value	Choose a different data view	Choose a different database, Table, and/or Layout	Choose other data view configuration options*
`hidden`			
`keyOnly`	✓		
`keyOrData`	✓	✓	
`advanced`	✓	✓	✓
`keyInGroup`	Users can choose other data views in in the same group		

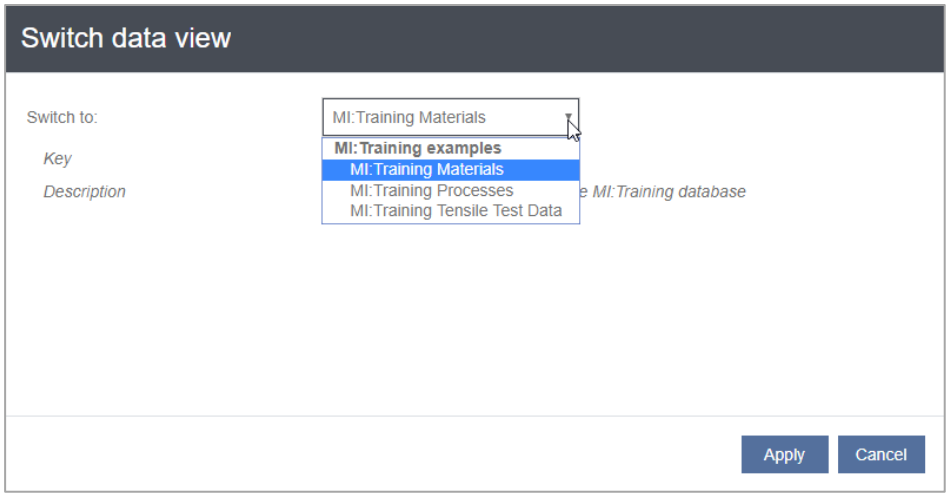
*Note that not all of the settings that can be specified in a data view configuration are available through the “Advanced options” UI in the Switch data view dialog.

Examples:

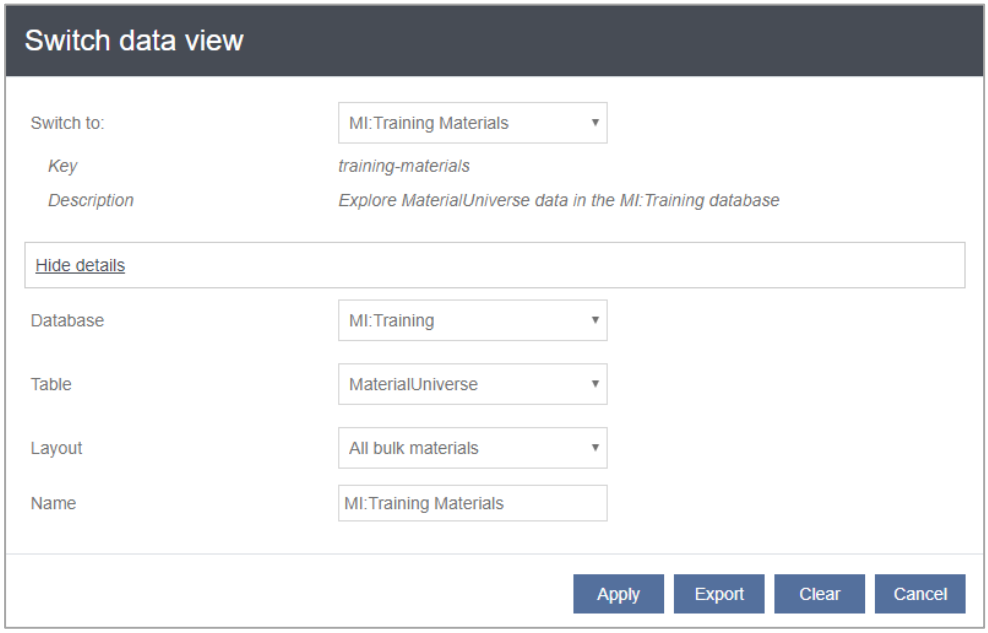
```
"configSwitching": "keyOnly"
```



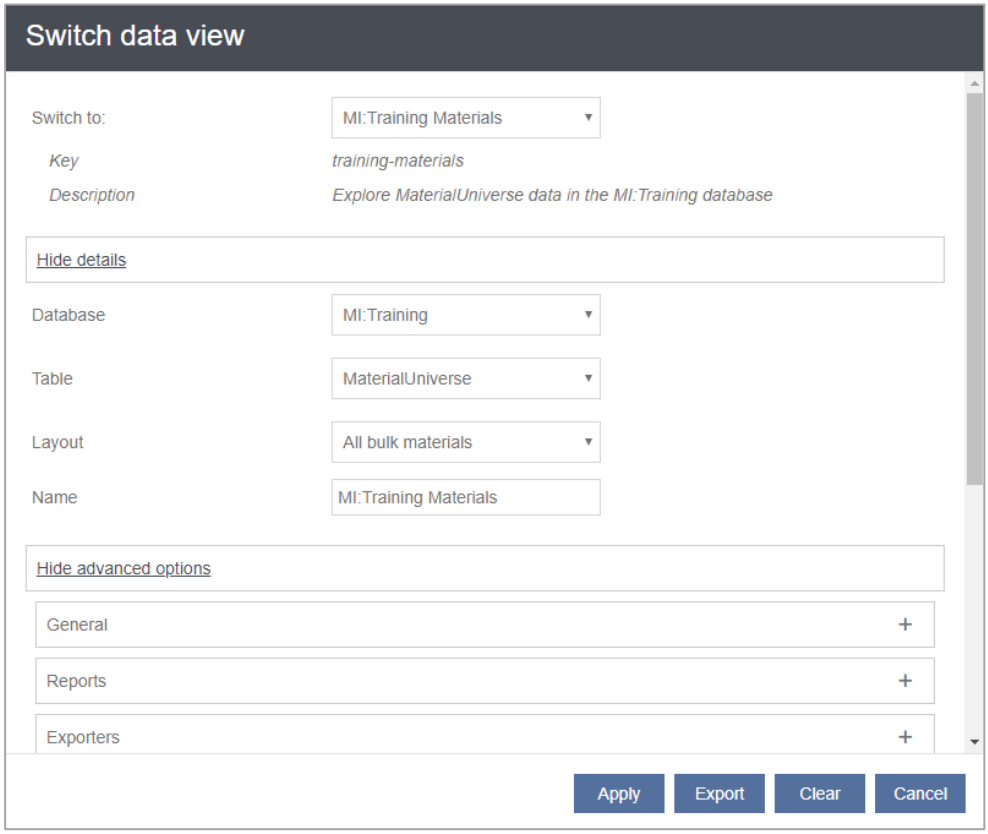

```
"configSwitching": "keyInGroup"
```



```
"configSwitching": "keyOrData"
```




```
"configSwitching": "advanced"
```



4.13 Data add and edit functionality

4.13.1 Enabling data edit capability

Data edit capability is enabled or disabled with the `editableDatasheetLayout` property in the data view configuration.

This setting *both* enables/disables creation of new records and editing of data, *and* specifies the Layout used when entering or editing data. (To allow users to create new records, a `subset` must also be specified in the data view configuration.)

When `editableDatasheetLayout` is specified in the data view configuration:

- A  **New** button appears on the toolbar; users with the necessary permissions can click this to add new records.
- An **Edit** button appears on datasheets, allowing users with the necessary permissions to edit data values.

  (Some addition configuration is required if users will need to edit tabular Attributes; see section *4.13.2*)

The specified Layout defines the headings and Attributes shown when entering or editing data (see Section *4.11.1*) and may also identify certain Attributes as ***read-only***, meaning they can be viewed but not modified, or ***required***, meaning they must be completed in order to save the record.

If the `editableDatasheetLayout` property is not specified in the data view configuration, the **New** button will not appear on the toolbar and the **Edit** button will not appear in Explore datasheets.

User permissions in version-controlled Tables

Note that Explore application users who need to edit version-controlled records must be members of the GRANTA MI *Power User* system security role.

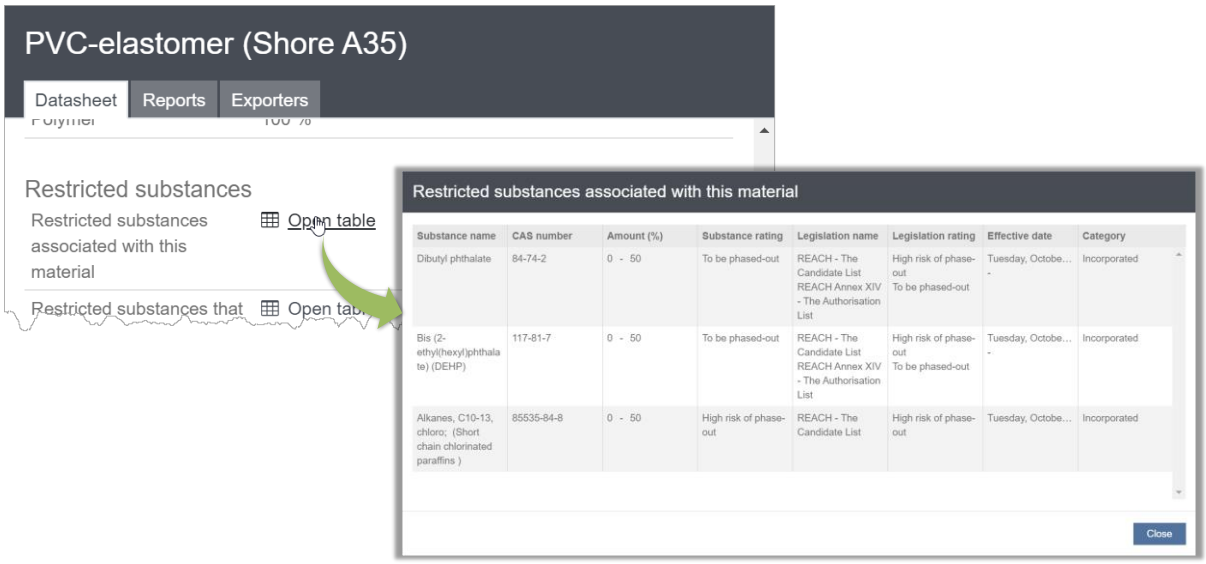
This is because Explore automatically tries to release new record versions; if a user does not have sufficient privileges to be able to release records (i.e. they are not members of the *Power User* role, or higher), the record will be updated and the new record version created, but it cannot be released, and so the user will see an error.

4.13.2 Viewing and editing tabular data

Tabular Attributes are used in GRANTA MI to store complex data, with data points (simple numeric, text, media, logical, or date data) organized into columns and rows; tabular Attributes may include local data as well as links to data stored in other Tables.

Viewing tabular data in the Explore application

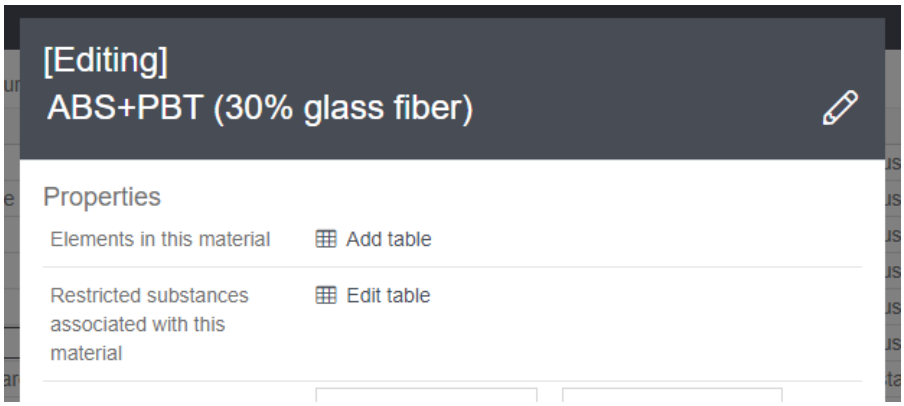
When a tabular Attribute is included in the Layout specified in dataSheetLayout, an **Open table** option will appear on the datasheet, enabling users to view the data. For example:



Adding and editing tabular data in the Explore application

The ability to edit tabular data in the Explore application is controlled with the tabularEditingEnabled property in the data view configuration.

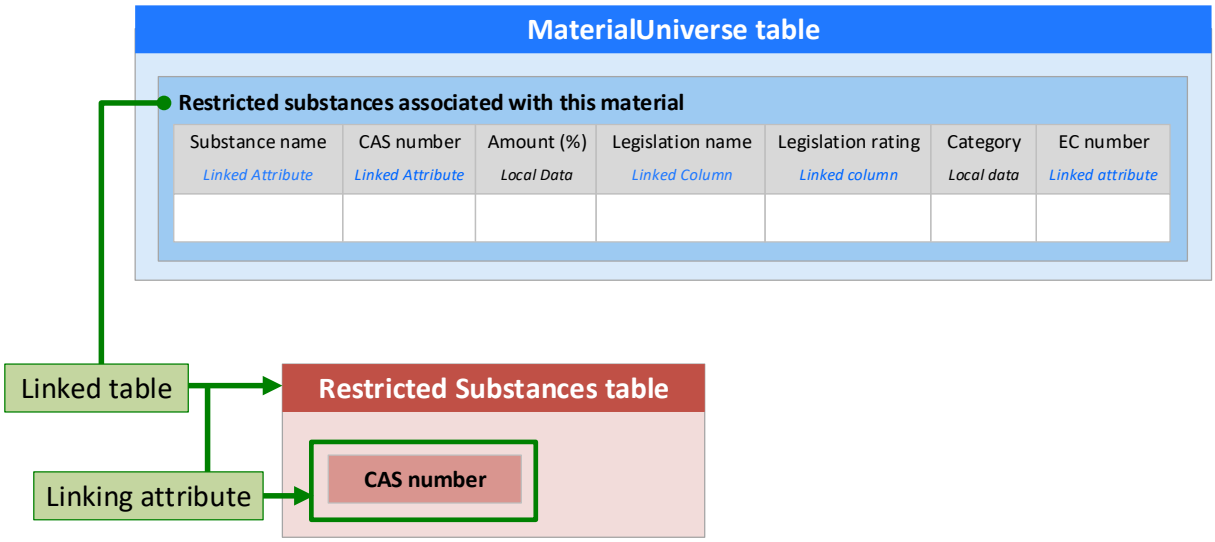
When tabular data editing is enabled, **Edit table** or **Add table** options will appear on the editable datasheet for Tabular Attributes. For example:



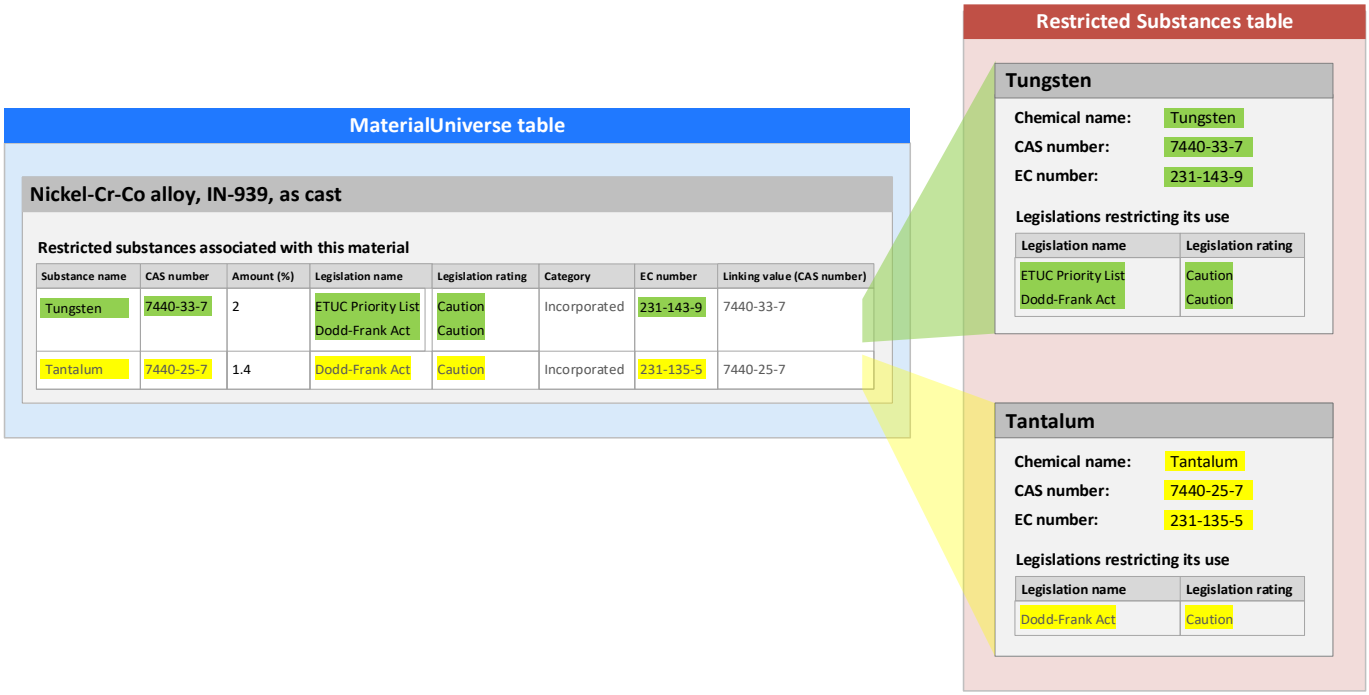
Editing tabular data linking values

Tabular Attributes in GRANTA MI may include data from other Tables, as well as local data. To include data from other records, each row of tabular data uses the value of a **linking Attribute** to link to the relevant record in the **linked Table**.

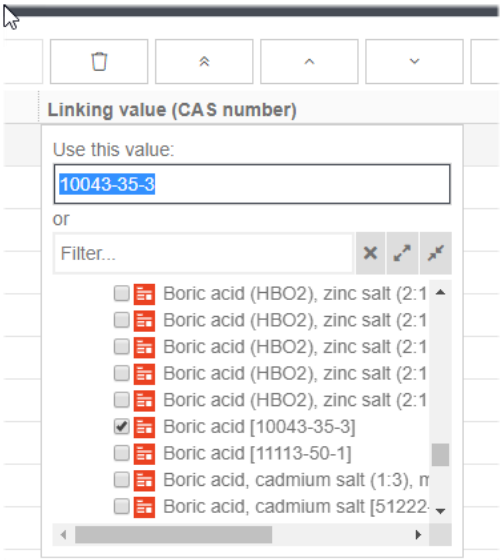
For example, in the *Product Risk* database *MaterialUniverse* Table, the tabular Attribute *Restricted substances associated with this material* is linked to the *Restricted Substances* Table via the CAS number ***linking Attribute*** value.



The linking value connects the tabular data row in the *MaterialUniverse* Table to a record in the *Restricted Substances* Table, allowing data values from the *Restricted Substances* record to be included in the tabular data row, for example:



In the Explore application, users can edit linking values in tabular data Attributes by typing in the value, or by selecting a record from which the linking value can be retrieved. For example:

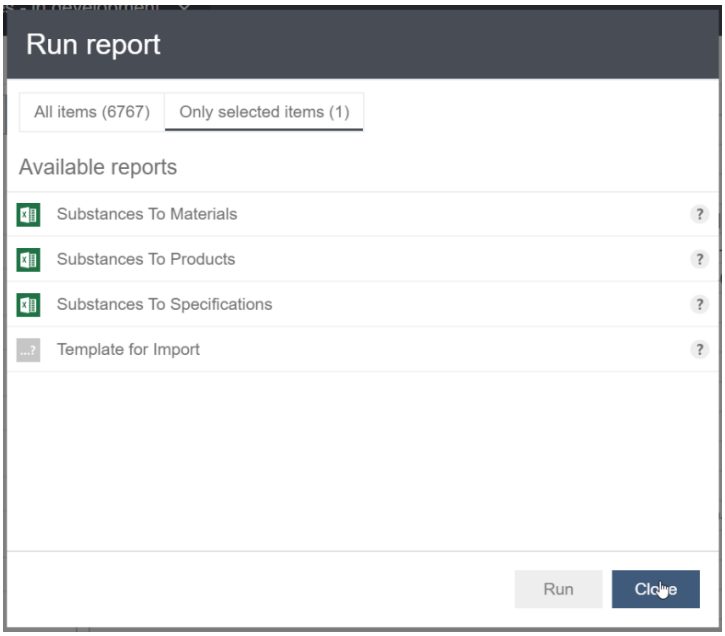


4.14 Reports

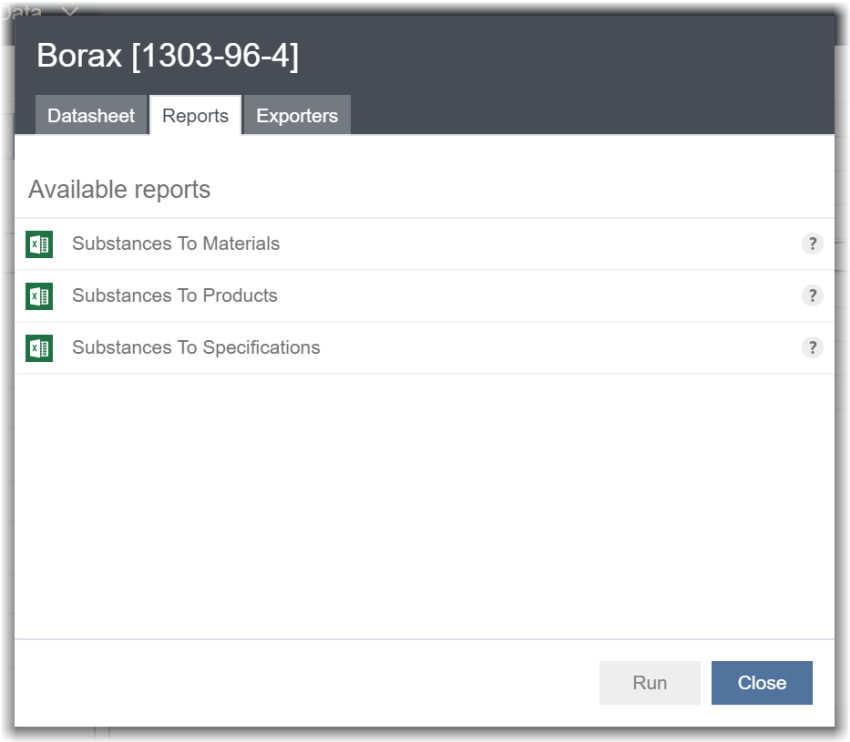
Where reports from the MI:Reports package are installed in your GRANTA MI environment, reports that analyze a list of records can be run from within the Explore application via the **Report** option on the toolbar.

Reports functionality is enabled and disabled with the reportsDisabled property in the data view configuration. When reports are enabled:

- The **Run a report** button appears on the toolbar. Users can click this to open a dialog where they can select and run available reports on all records, or on only the selected records:



- A **Reports** tab appears on record datasheets showing the reports that can be run on that record:



4.15 Data exporters

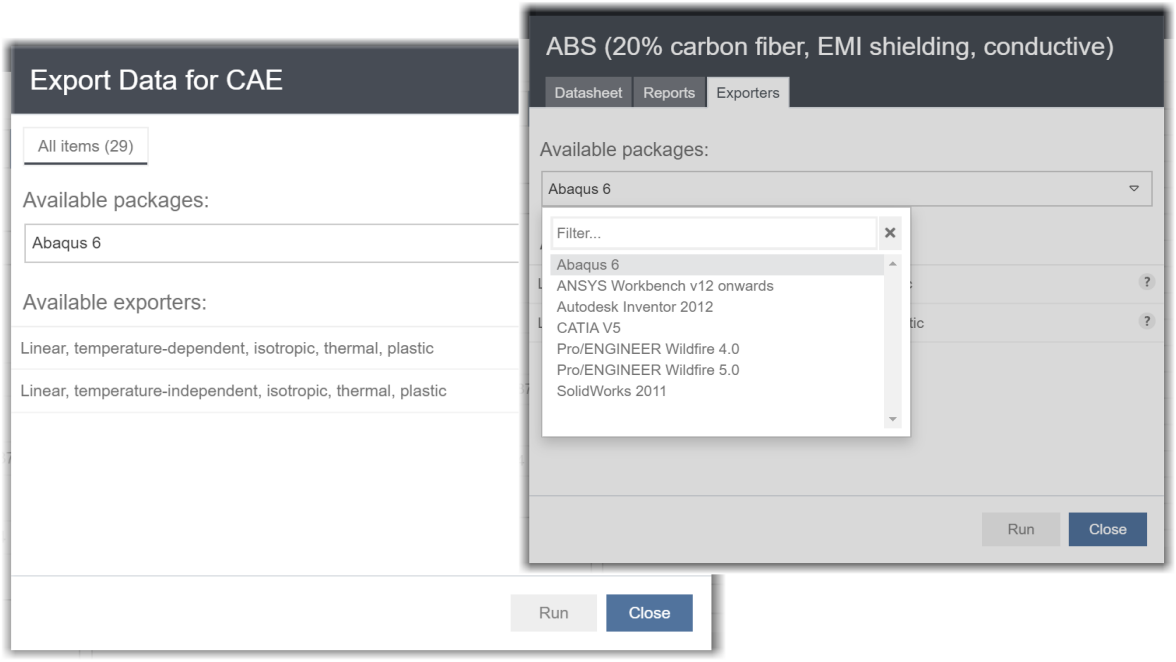
Materials data can be exported from GRANTA MI in a range of different formats for use in CAE packages such as Abaqus, ANSYS, or NastranNX. Where exporters are available in your GRANTA MI database, these can be run from within the Explore application via the **Export Data for CAE** option on the toolbar.

4.15.1 Enabling exporters

Exporter functionality is enabled and disabled with the exportersDisabled property in the data view configuration.

When the exporters functionality is enabled:

- An **Export Data for CAE** button appears on the toolbar. Users can click this to open a dialog where they can select one of the available exporters, and export data from the selected records:
- An **Exporters** tab appears on record datasheets showing the packages and exporters that can be used to export the record data:



4.15.2 Export units

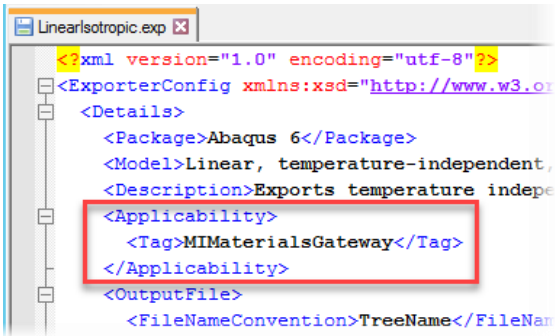
By default, data is exported using the Unit System specified in the exporter configuration.def file. Alternatively, the Unit System specified in the user's Explore application preferences may be used to export data. This is specified using the exportersUseCurrentUnitSystem property in the configuration file:

- When **true**, data will be exported using the Unit System specified in the user's Explore preferences
- When **false**, or not specified in the data view configuration, data will be exported using the unit system specified in the exporter .exp file.

Note: The current Unit System may not necessarily be supported by the selected exporter.

4.15.3 Hiding application-specific exporters

FEA exporters in a GRANTA MI database may be configured for use with specific applications such as MI:Materials Gateway, or MI:Viewer/Explore, via a setting in the exporter configuration (.exp) file, for example:



This `<Applicability>` tag in the exporter configuration file specifies which applications the exporter is configured for, for example:

- MIMaterialsGateway = MI: Materials Gateway only
- MIViewer = MI:Viewer or Explore

This ensures that only relevant exporters are shown when users are exporting data from different applications.

To ensure that Explore application users see only exporters that are configured for MI:Viewer or Explore, the exporterApplicability property in the data view configuration should be set to match the value of the `<Applicability>` tag in the exporter configuration file.

For example, in the Explore data view configuration:

```
"exporterApplicability": "MIViewer"
```

and in the Exporter .exp file:

```
    <Applicability>
      <Tag>MIViewer</Tag>
    </Applicability>
```

Note that:

- If the exporter .exp config file does not include an `<Applicability>` tag, the exporter will always be shown in the Explore application.
- If the `exporterApplicability` setting is missing from the configuration file, all exporters in the database are shown.

4.16 Options for placement of new records

Records created in the Explore application can be placed into specific folders in the GRANTA MI database tree structure. This is configured by specifying the destination folder path using the newRecordLocation property in the data view configuration. For example:

```
"newRecordLocation": "Metals and alloys/NEW RECORDS",
```

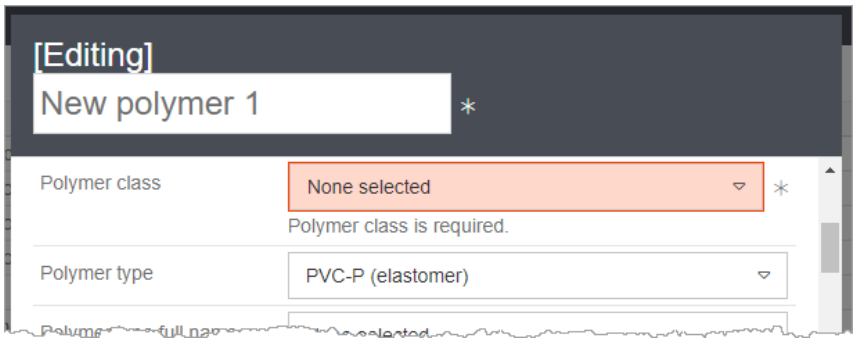
- The folder will be created if it does not already exist.
- If the `newRecordLocation` property is not specified in the data view configuration, new records will be placed in the top-level (root) folder by default.
- To create records, users must have write permission to the parent folder.

As well as literal strings, the specified path can include placement tokens that allow an Attribute value in the record to be used to identify the destination folder. When the new record is created, the value of the named Attribute will replace the token in the folder path. The folder will be created if it does not already exist.

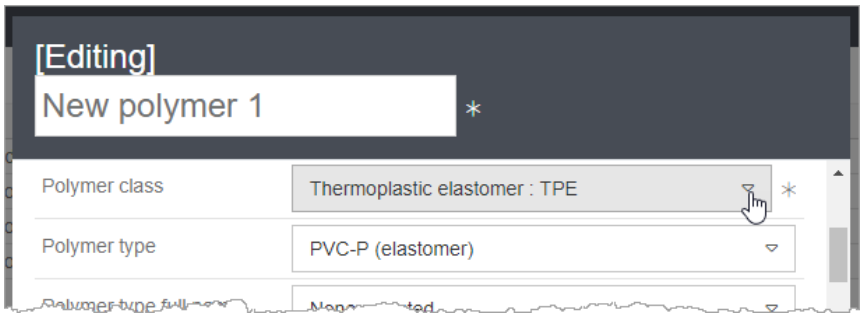
Placement tokens are specified using curly brackets `{}`. For example, in this code, the folder name is the value of the *Polymer class* Attribute:

```
"newRecordLocation": "My Records/{Polymer class}",
```

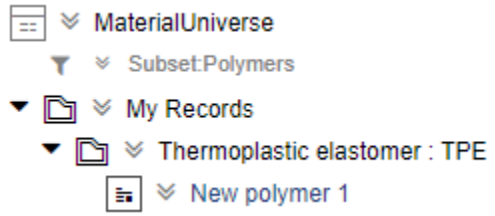
When creating a new record, **Polymer class** will be a required field, and users will not be able to create the new record unless a value is specified:



Once the polymer class has been specified, the new record can be created:



It will be placed in the correct location in the database tree, in this example a new folder under the *My Records* folder, as specified in the `newRecordLocation` property:



The Attribute types that can be used as placement tokens are:

- Integer
- Single-value Point (but units are ignored)
- Range (but units are ignored)
- Single-value Discrete
- Date
- Short text. Alphanumeric characters other than underscore (_) , dash (-) , period (.), forward slash (/), space, and left/right parentheses ( ) are stripped out.

Multi-value Point and Discrete types cannot be used.

If an invalid Attribute name is given, it will be interpreted literally, that is, it will fall back to a fixed path. For example, the following are equivalent:

```
"newRecordLocation": "Metals/New records/{invalidAttributeName}",
"newRecordLocation": "Metals/New records/invalidAttributeName",
```

4.17 Advanced options for auto-naming of new records

Automatic naming and numbering conventions can be configured for records created in the Explore application using a number of configuration settings the configuration file.

A system of transforms can be used to construct the new record name, for example, to:

- Take two Attribute values and concatenate them into a third Attribute value;
- Increment the value of an integer Attribute;
- Pad the integer Attribute to a fixed length string;
- Concatenate two Attributes and write them into the record name.

A number of useful transforms are supplied as part of the Granta Web Platform; the system is extensible, however, and you can write your own.

Table 1. Available transforms

Record Transform	Function	Standard Names	Mapped to Attribute of type	Required?
Counter	Increments an integer Attribute with every new record created	Counter	Integer	Y
Id	Concatenates two Attribute values into a third Attribute	Id_Prefix	Discrete	N
		Id_Suffix	Integer, Short text	N
		Id	Short text	Y
RecordNameConcatenator	Concatenates two Attribute values into the record name	Name_Prefix	Integer, Short text, Discrete	Y
		Name_Suffix	Integer, Short text, Discrete	N

If a required Attribute is missing, the transform will fail. If a non-required Attribute is missing, the transform will succeed but the concatenation will be incomplete.

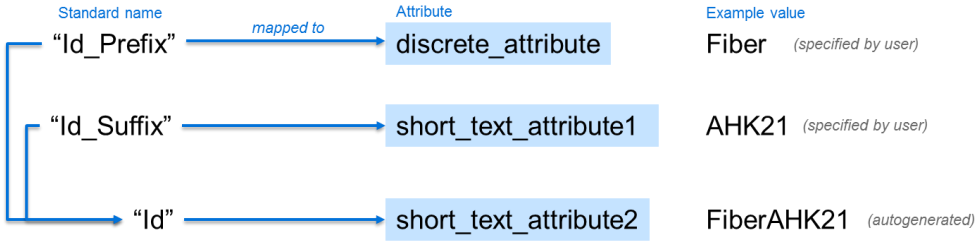
4.17.1 The Id transform

This transform concatenates two Attribute values into a third Attribute.

Record Transform	Standard Names	Attribute type	Required?
Id	Id_Prefix	Discrete	N
	Id_Suffix	Integer or Short text	N
	Id	Short text	Y

The **Id_Prefix** and **Id_Suffix** standard named Attributes are both optional. If neither is supplied, the result will be that the **Id** standard named Attribute will be empty.

Example



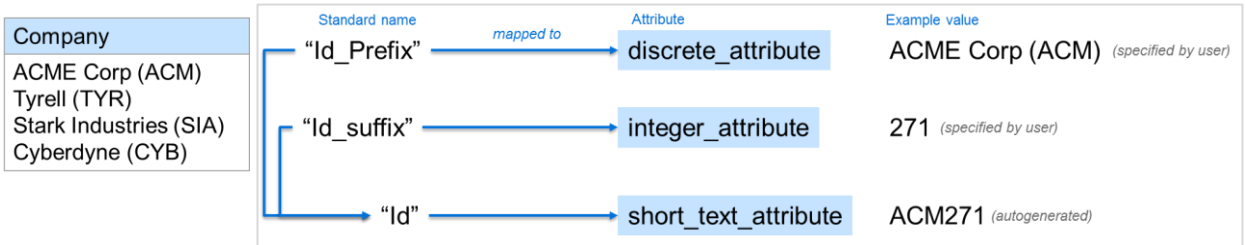
To set this up:

1. Set up three standard named Attributes: **Id**, **Id_Prefix,** and **Id_Suffix**.
1. Add the **Id** transform to the list of transforms in the configuration file: `"newRecordTransformation": ["Id"]`
2. When a new record is created, the **Id** Attribute in the record is populated automatically from the values specified in the other two Attributes.

Text in parentheses

If the value of the **Id_Prefix** standard named Attribute is a string containing a set of parentheses ( ), then only the substring within the brackets is concatenated with the **Id_Suffix** standard named Attribute to produce the value written into the **Id** standard named Attribute.

For example, for an **Id_Prefix** standard named Attribute "Company", with a value of "ACME Corp (ACM)", only the string within parentheses, "ACM", is included in the **Id**.

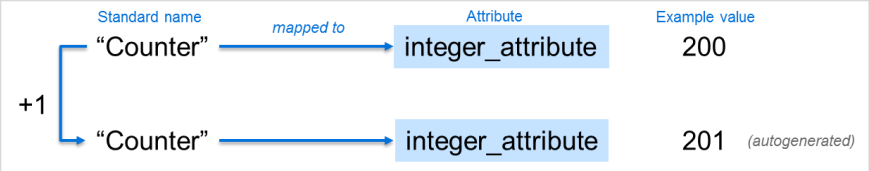


4.17.2 The Counter transform

This transform increments an integer Attribute with every new record created.

Transform	Standard Names	Mapped to Attribute of type	Required?
Counter	Counter	Integer	Y

Example:



To set this up:

3. Set up one standard named Attribute, **Counter**, of type integer.
4. Add `Counter` to the list of transforms in the configuration file: `"newRecordTransformation": ["Counter"]`

Each time a new record is created, the value of the **Counter** Attribute is incremented.

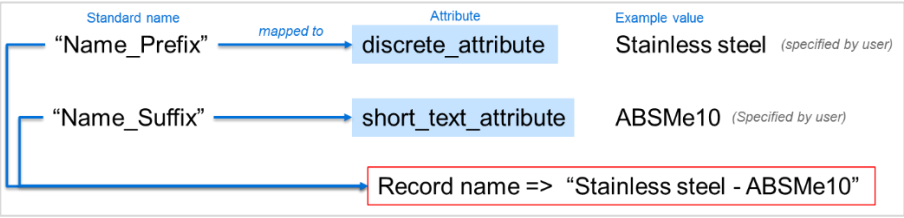
4.17.3 The RecordNameConcatenator transform

This transform concatenates two Attribute values into the record name.

Record Transform	Standard Names	Attribute type	Required?
RecordNameConcatenator	Name_Prefix	Integer, Short text, Discrete	Y
	Name_Suffix	Integer, Short text, Discrete	N

The **Name_Suffix** standard named Attribute is optional. If it is not supplied, the resulting record name will include only the **Name_Prefix** value.

Example:



To set this up:

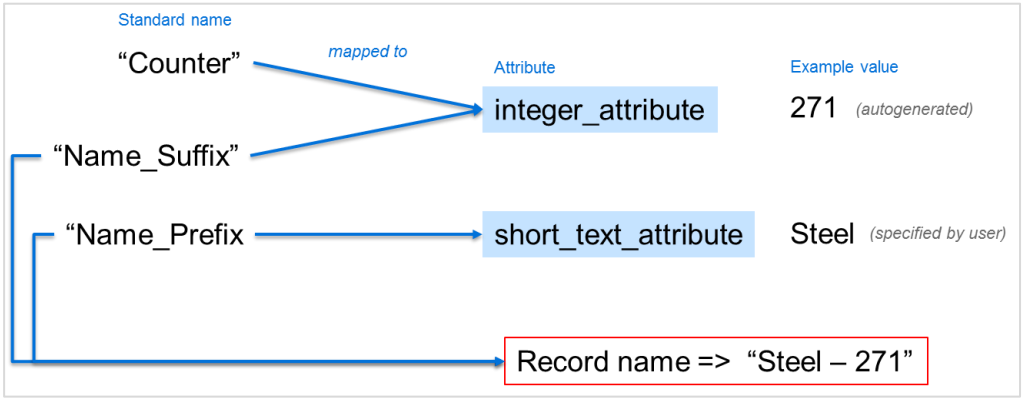
1. Set up two standard named Attributes (Integer, Short text, or Discrete) to use for the first and second part of the record name; for example, **Name_Prefix** and **Name_Suffix**.

2. Add `RecordNameConcatenator` to the list of transforms in the configuration file: `"newRecordTransformation": ["RecordNameConcatenator"]`
3. When a new record is created, it is automatically named by concatenating the two Attribute values, separated by a hyphen (-).

4.17.4 Combining two transforms

Multiple transforms may be specified in the configuration file; they are applied in order. For example, here, the `Counter` and `RecordNameConcatenator` transforms are combined:

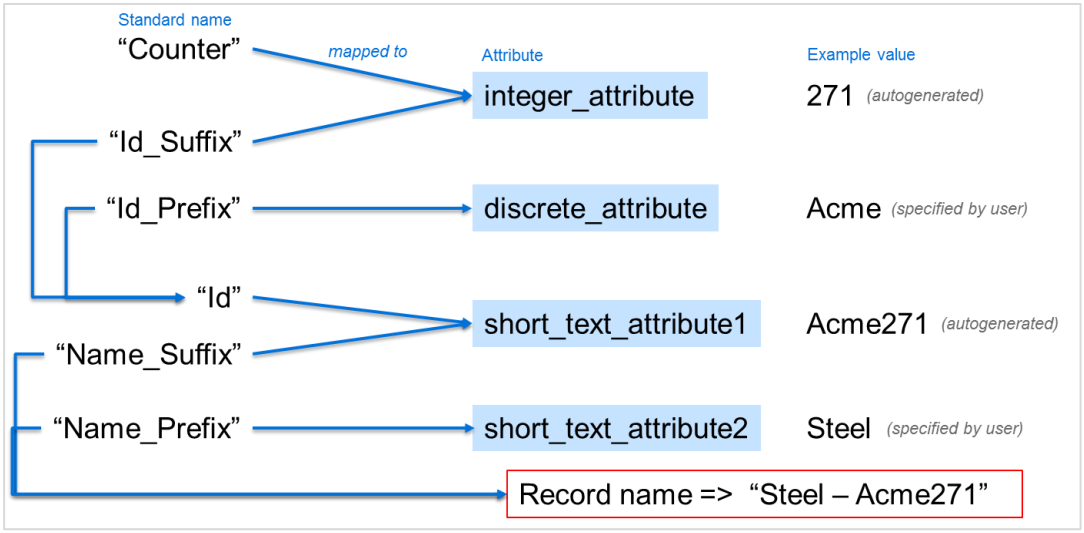
```
"newRecordTransformation": ["Counter", "RecordNameConcatenator"]
```



4.17.5 Combining three transforms

All three Transforms may be specified in the configuration file. For example, here, the `Counter`, `ID`, and `RecordNameConcatenator` transforms are used:

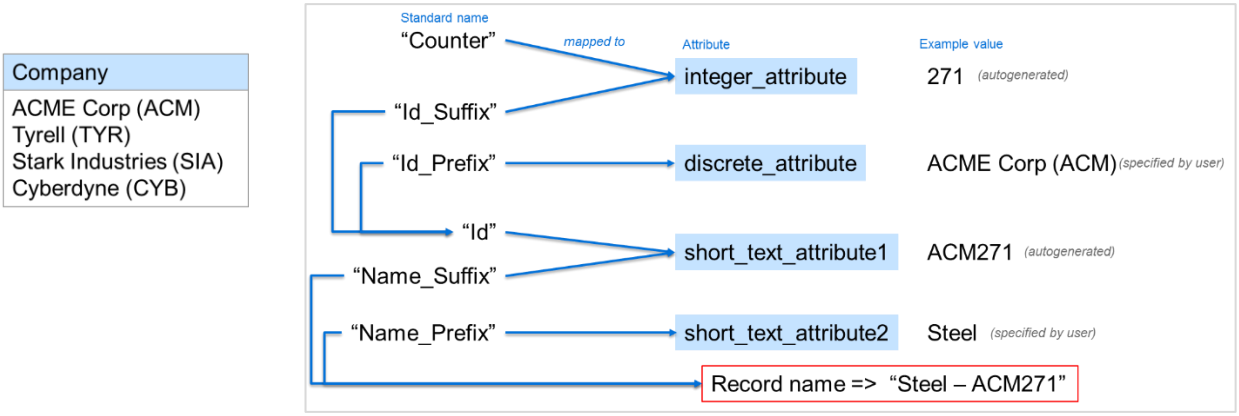
```
"newRecordTransformation": ["Counter", "Id", "RecordNameConcatenator"]
```



4.17.6 Additional transform examples

Text in parentheses example 2

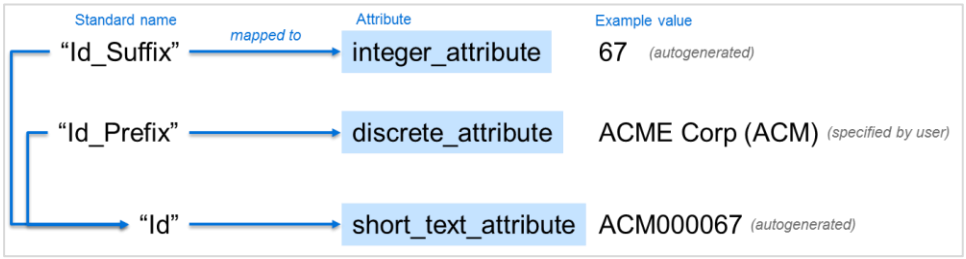
If the value of the **Id_Prefix** includes text within parentheses ( ), that text will be used in the record name and not the whole string. For example:



Padding characters

The **Id** transform may include padding on integer Attributes. For example, in this configuration, the numeric counter part of the record name is padded with zeroes to a length of 6 characters:

```
"newRecordPaddingOptions": {
      "padLength": 6,
      "padString": "0"
}
```



4.17.7 More information

The **Counter**, **Id**, and **RecordNameConcatenator** transforms are available in the Granta Web Platform. Additional custom transforms may be developed using the `IRecordTransformer` interface; see the Web Platform SDK documentation for details.

4.18 Loading data on demand

Data can be loaded on demand, or on application startup.

With the *load on startup* option, all data is loaded when the application is started, or when switching to a different data view. This is recommended for small datasets.

Having data *load on demand* can reduce the startup time of the application when using large datasets.

The *Load data on demand* feature is enabled and disabled with the loadDataOnDemand property in the data view configuration:

- When **false**, all data will be loaded on startup. As users change their search criteria, the search results update instantaneously.
- When **true**, data will be loaded into the application whenever the **Search** button at the bottom of the Search panel is clicked.

Note that the Curves panel is less responsive in data on demand mode, as data has to be loaded from the server before updating the curve plot.

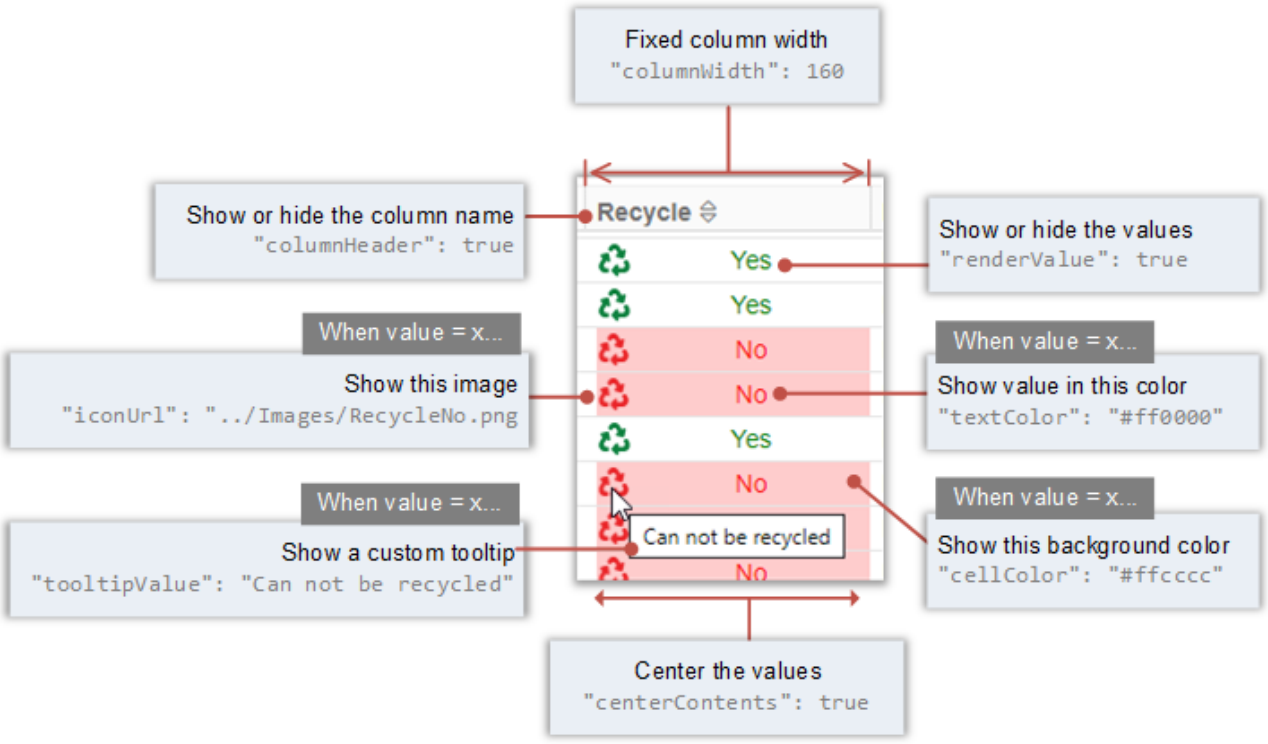
5 Data formatting options

This section provides an overview of data formatting options which allow you to change the way data values are displayed in datasheets, and in the List and Scatter plot views. For reference information and syntax for the corresponding JSON settings, Section *8*.

Data formatting options allow you to specify styling for specific Attributes, with additional formatting overrides that are applied whenever the Attribute has a specific value. For example, for a numeric Attribute, values above a certain threshold could have a different text color and background color, while for a Boolean Attribute such as *Recycle*, you could represent the different values with different images.

5.1 List view column styling options

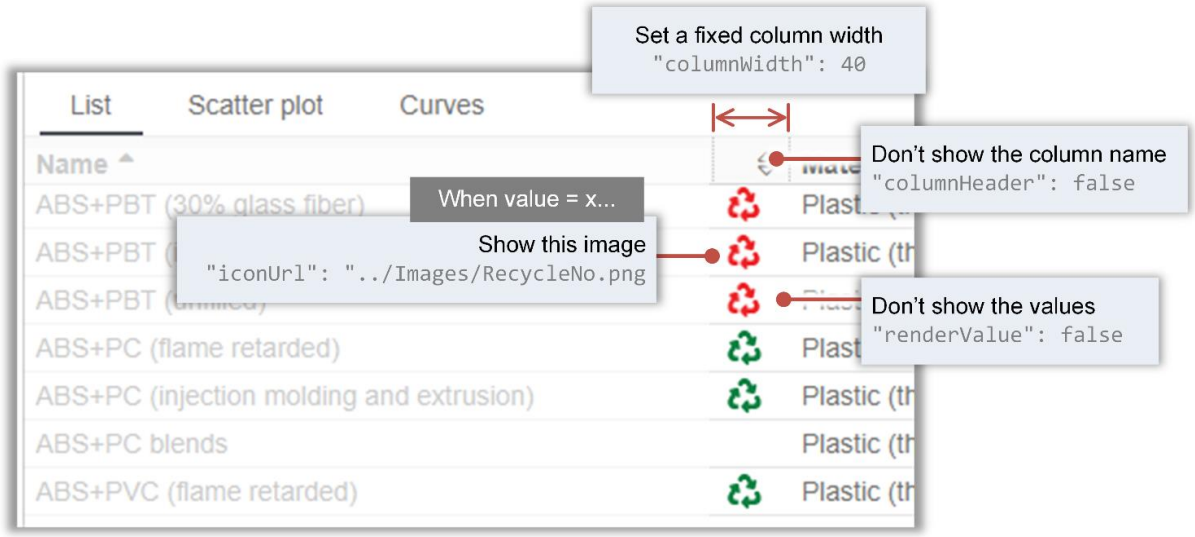
These UI elements can be formatted to customize the appearance of data in the List view:



See Section *5.4, Custom formatting* for information on how Attribute value formatting is configured.

Required configuration to use images instead of data values, as shown in this example, is covered in Section *5.4.3*.

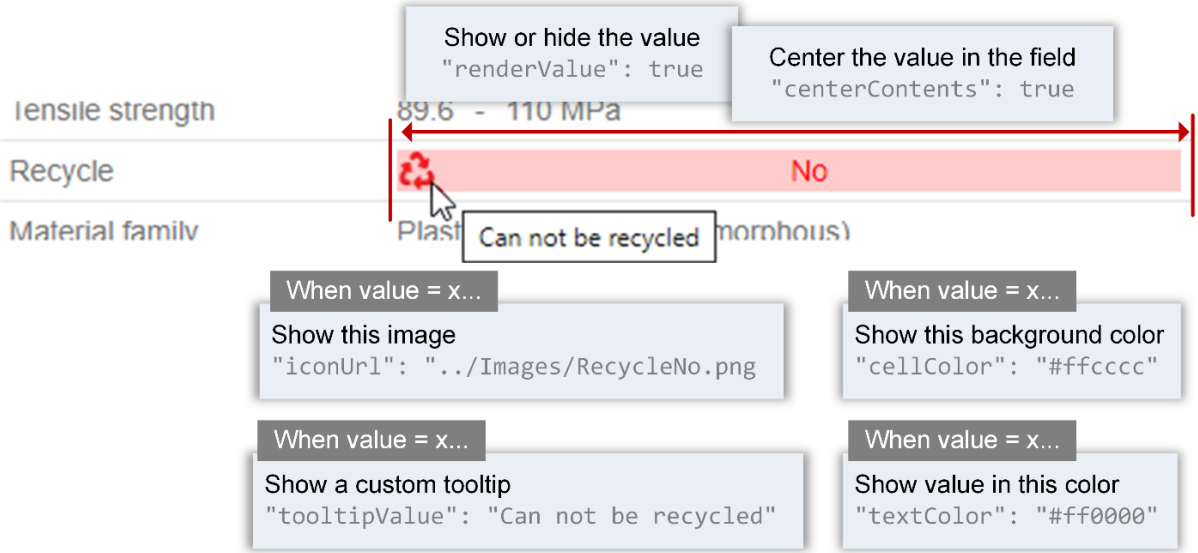
In this example, the *Recycle* column shows an image instead of the value, the column has a fixed width, and no column header:



See Section *5.4.3* for details of required configuration for using images to represent data values as shown in this example.

5.2 Datasheet styling options

These UI elements can be formatted to customize the appearance of data in datasheets:

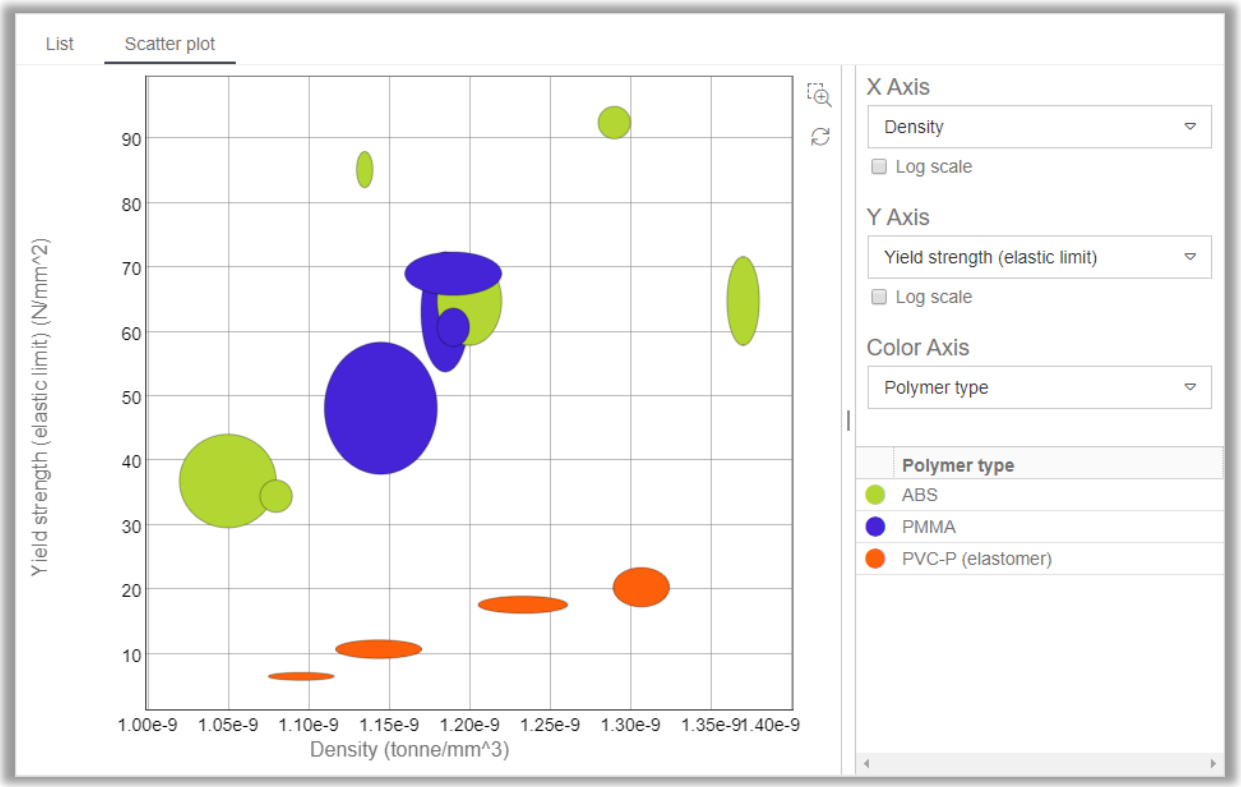


See Section *5.4, Custom formatting* for information on how Attribute value formatting is configured.

5.3 Scatter plot styling options

Styling options for scatter plots allow the bubble colors to be dependent on the value of an Attribute. Where this feature is configured, a third axis control–**Color Axis**–appears in addition to the X Axis and Y Axis controls in the Scatter plot view, and users can choose the Attribute to use as the bubble color variable. A legend is also provided showing the possible values and the colors they are mapped to.

For example, *Polymer type* is selected below on the Color Axis and so the bubble colors are determined by the *Polymer type* value in each record.



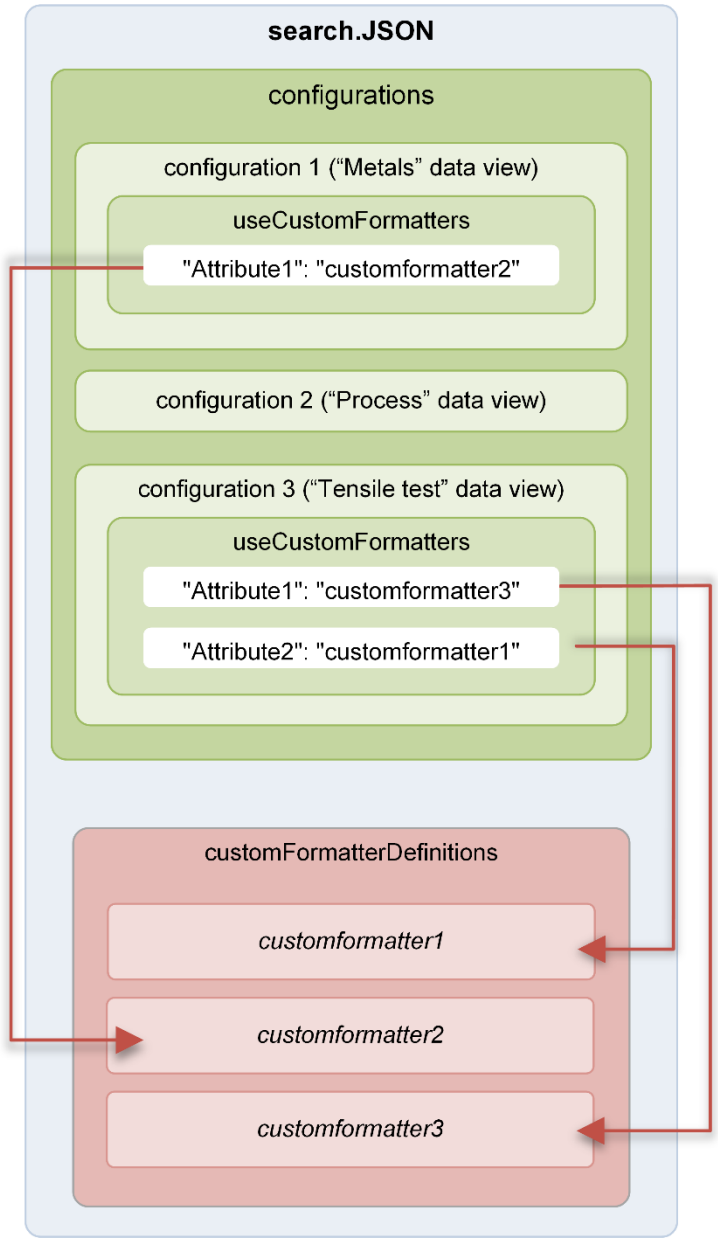
See Section *5.4, Custom formatting* for information on how Attribute value formatting is configured.

5.4 Custom formatting configuration

Custom formatting rules are defined in the application configuration file under the customFormatterDefinitions property. This property defines one or more *custom formatters*, each of which specifies the styling elements (text color, background colors, images, tooltips, etc.) and the rules for when and where the styling should be applied.

Custom formatting rules are defined once in the configuration file, and can be used in multiple data views.

To use custom formatting in a data view, Attributes in the data view must be mapped to *custom formatters* with the data view configuration property `useCustomFormatters`. For example:



5.4.1 Custom formatter definitions

Each custom formatter definition specifies:

- the required styling elements – whether or not values are displayed, text color and alignment, background color, inclusion of images as well as/instead of values, custom tooltips;
- **where** in the user interface the styling should be applied – in datasheets, in the List view, and/or on the Scatter plot view;
- **when** the styling should be applied – wherever the Attribute appears (specified with AttributeFormatters), or only when the Attribute has a specific value (specified with valueFormatters)

5.4.2 Mapping Attributes to custom formatters

The useCustomFormatters property in the data view configuration maps Attributes in a data view to custom formatters. The following Attribute types can be mapped:

- Logical (Boolean) e.g. *Recycle*, *Biodegrade*, *Extruded*
- Discrete e.g. *Flammability*, *Polymer type*, *Material family*, *Base*
- Numeric. This can be Integer, Point, or Range Attributes, but note that % is the only permitted unit; data with any other units cannot be formatted.
  - Integer: *Atomic number*, *REACH Candidate List indicator*
  - Point: *Shape factor*, *Chem. resistance index*, *Strain at tensile failure*
  - Range: *Recycle fraction in current supply*, *Renewable content*

Syntax

```
"useCustomFormatters": {
      "attribute_name": "customFormatterDefinition_name"
      }
```

Example

```
"useCustomFormatters": {
      "Recycle": "recyclingBool",
      "Polymer type": "polymertypeDiscrete"
      }
```

where `recycling_Bool` and `polymertypeDiscrete` are custom formatter definitions specified under the customFormatterDefinitions property, for example:

```
"customFormatterDefinitions": {
            "recyclingBool": {
                  ... // Formatting options
            },
            "polymertypeDiscrete": {
                  ... // Formatting options
            }
      }
```

For further details, see: useCustomFormatters , Custom data formatting properties

5.4.3 Representing data values with images

It is possible to shown images instead of, or as well as, values in the List view and/or in datasheets. For example, the value of the *Recycle* Discrete Attribute below is represented with an image instead of showing the value (*Yes* or *No*) in this List view:

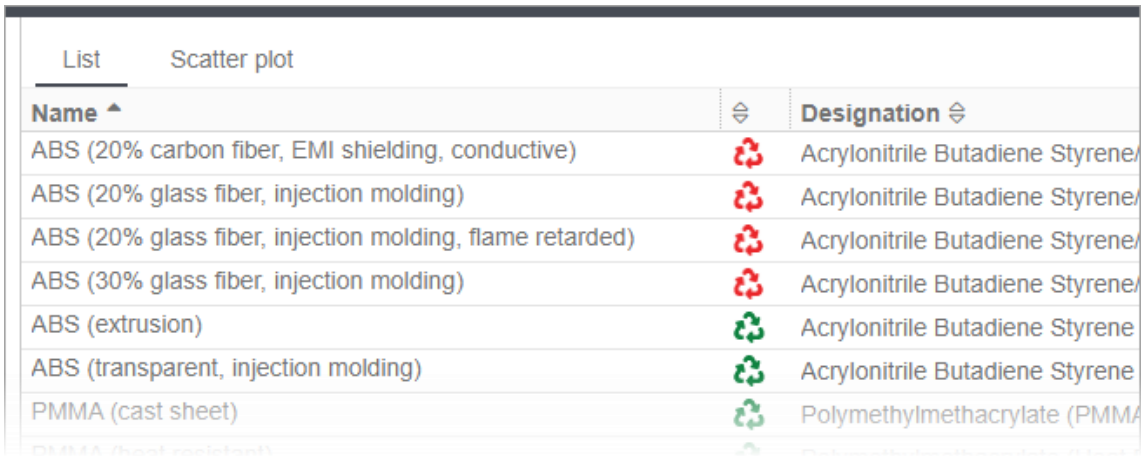
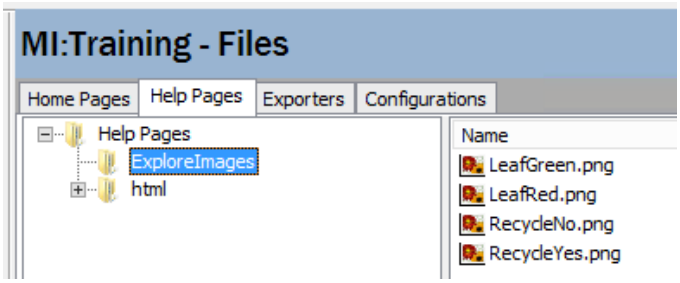


Image files required for view formatting can be embedded in the database, or stored on the server file system, and then referenced with a URL in the `iconURL` property under valueFormatters.

To use image files stored in the database:

1. In the MI:Admin tool upload the image files to the database:
   a. Click on Edit Files, and click the *Help Pages* tab.
   b. Upload the image files to the database by clicking **Import Files**. You can place the image files at the top level or in a folder, for example:



2. In the data view configuration, specify the image file URL with the `iconURL` property under valueFormatters using the following format:

```
"iconUrl": "/mi/help.ashx/<db_key>/<filename>",
```

For example:

```
"valueFormatters": [
  {
    "match": true,
    "list": {
      "iconUrl": "/mi/help.ashx/MI_Training/ExploreImages/RecycleYes.png",
      "tooltipValue": "Can be recycled"
      },
```

Alternatively, you can use image files stored on the server file system and specify a relative path in the `iconURL` property like this for example:

```
"iconUrl": "../configImages/RecycleYes.png",
```

6 Preferences and settings

Application settings and preferences for the Explore application are defined in the GRANTA MI Settings Service. Settings are stored as key-value pairs, where the values are JSON format configuration information.

- Global preferences, which apply to multiple applications, are stored in **settings.preferences.json**
- Application-level preferences are stored in **settings.app-preferences.json**

Where preferences are defined both globally and at application-level, application-level values will override the global preferences.

6.1 Application-level preferences

Application-level preferences are stored in **settings.app-preferences.json** in the Settings Service.

For the Explore application, the default unit system and the ability for application users to choose a different unit system may be specified here.

Syntax

```
{
  "explore": {
    "preferences": {
      "unitSystem": {
        "userEditable": true|false,
        "defaultValue": "unit_system_name"
      }
    }
  },
...
```

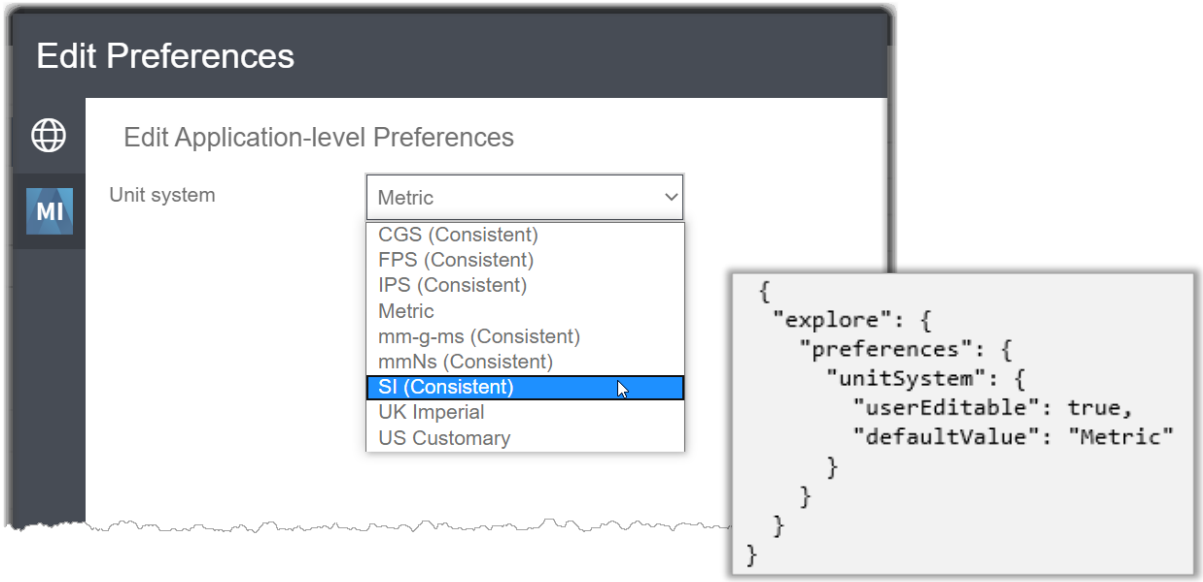
Properties

Property	Value type	Description
`userEditable`	Boolean	Defines whether or not the Unit System selection control is shown in the Explore user interface. • `true` = users can choose a different Unit system in the Edit Preferences UI • `false` = users can see the current unit system, but can't change it.
`defaultValue`	String	Specifies a Unit System name.

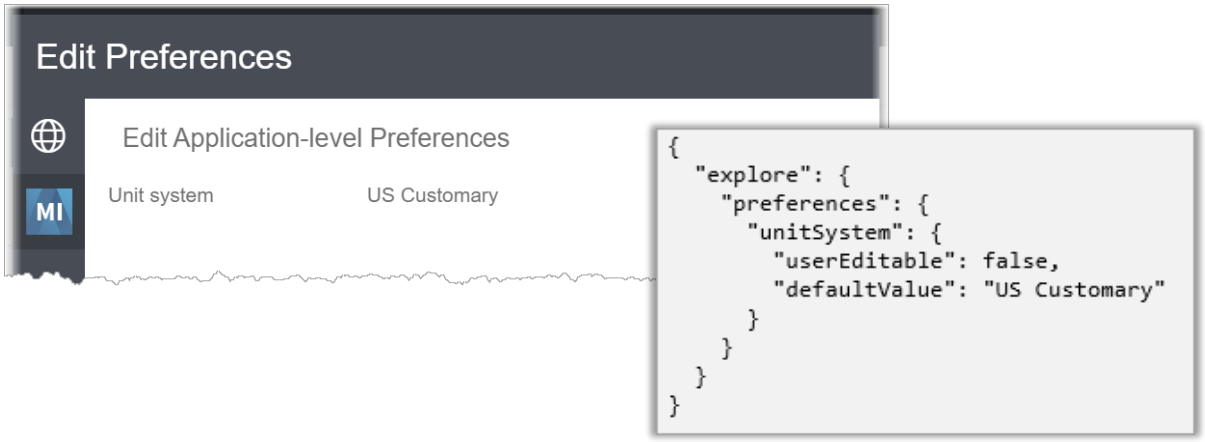
Where the data is stored in a different Unit System, the Explore application accesses services in GRANTA MI to perform unit conversions, using information in the current database.

The Unit System used when exporting data may be specified independently with exportersUseCurrentUnitSystem, see Section *4.15.2, Export units*.

When `userEditable` is *true*, users can choose their preferred units in the **Edit Application-level Preference**s page, for example:



To enforce the use of a specific Unit System, specify the Unit System name in `defaultValue` and set `userEditable` to *false*:

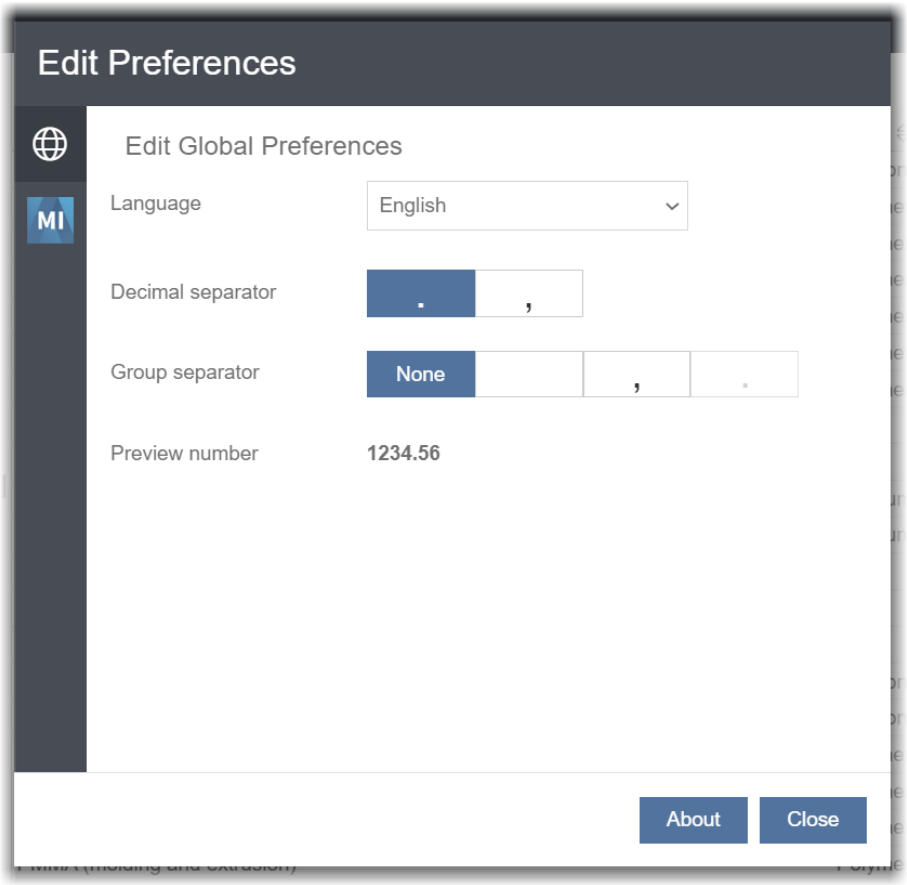


6.2 Global preferences

Preferences that apply across multiple GRANTA MI applications, including the Explore application, are stored in **settings.preferences.json** in the Settings Service.

6.2.1 UI language and number formatting settings

Where user editing has been enabled, the UI language and number formatting options can be accessed on the Edit Global Preferences page in the Explore UI:



Syntax

```
"preferences": {
  "language": {
    "userEditable": true|false,
    "defaultValue": "locale_code"
  },
  "numberFormatting": {
    "userEditable": true|false,
    "defaultValue": {
      "groupSeparator": ["", "\u2009", ",", "."],
      "decimalSeparator": [".", ","]
     }
  }
}
```

Language preferences

The default user interface language for the GRANTA MI (One MI) application, and the ability for users to choose a different language, is specified with the `language` property in **settings.preferences.json** in the Settings Service. For example:

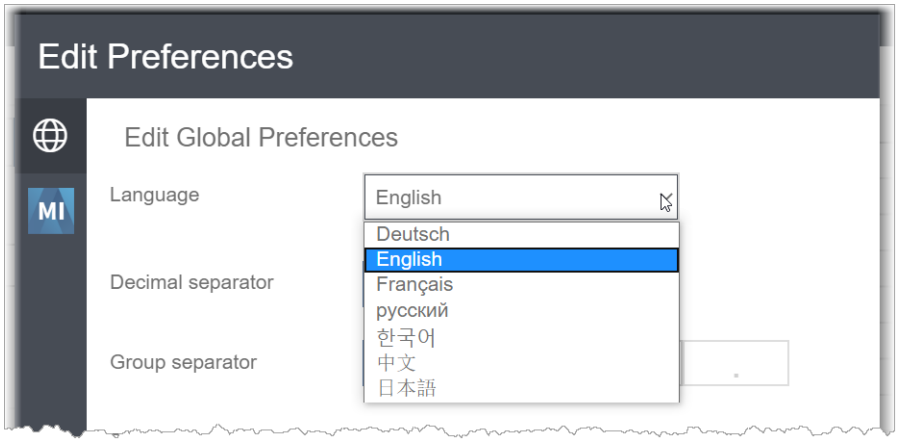
```
    "language": {
      "userEditable": true,
      "defaultValue": "en"
    },
```

Properties

Property	Value type	Description
userEditable	Boolean	Defines whether or not the language selection control is shown in the Application-level preferences user interface. • `true` = show the language selector in the Edit Preferences UI • `false` = do not show the language selector control
defaultValue	String	Specifies the default language for the application user interface. One of: de, en, fr, ja, ko, ru, zh

You can enforce the application language by setting `userVisible` to false and specifying the required language with `defaultValue`.

When `userVisible` is *true*, users can choose their preferred UI language on the Edit Global Preferences tab in the **Edit Preferences** dialog, but note that this will affect all GRANTA MI (One MI) applications, not just the Explore application:



Number formatting preferences

Group and number separator default settings for all GRANTA MI (One MI) applications, and the ability for application users to choose a different separator, are specified using the `numberFormatting` property in **settings.preferences.json** in the Settings Service.

```
    "numberFormatting": {
      "userEditable": true|false,
      "defaultValue": {
        "groupSeparator": ["", "\u2009", ",", "."],
        "decimalSeparator": [".", ","]
      }
    }
```

Properties

Property	Value type	Description
`userEditable`	Boolean	Defines whether or not the number formatting selection controls are shown in the application user interface. • `true` = show the Decimal separator and Group separator controls in the Edit Preferences UI • `false` = do not show the number formatting controls
`defaultValue`		Defines the number formatting character defaults.
`groupSeparator`	Array	Specifies the characters which may be selected as the digit group separator in the application Edit Preferences UI. "groupSeparator": [",", "", "\u2009", "."], Examples: • Comma: 1,234,567 • None: 1234567 • Space (`"\u2009"`): 1 234 567 • Period: 1.234.567 The first item in the array will be the default UI option.
`decimalSeparator`	Array	Specifies the characters which may be selected as the decimal separator in the application Edit Preferences UI. `"decimalSeparator": [".", ","]` Examples: • Period: 567.89 • Comma: 567,89 The first item in the array will be the default UI option.

For example, 1,234,567.89 has comma digit group separators and a period decimal separator.

Note that the same character cannot be specified as both the decimal and group separator.

You can enforce the formatting of separators by setting `userVisible` to false and entering only one separator type in each array.

If the `numberFormatting` setting is not defined in **settings.preferences.json** in the Settings Service, the Explore application will use the default operating system settings.

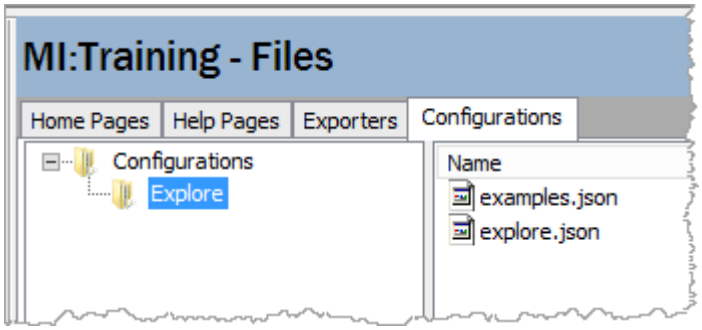
When `userVisible` is *true*, users can choose their preferred number formatting settings on the Edit Global Preferences tab in the **Edit Preferences** UI, but note that this will affect all GRANTA MI applications, not just Explore.

6.2.2 ConfigPaths - location of application configuration files

The default location of Explore application configuration files is specified in `configPaths` in **settings.preferences.json** in the Settings Service. For example:

```
"configPaths": {
  "explore": "Explore",
  "homePage": "HomePage"
},
```

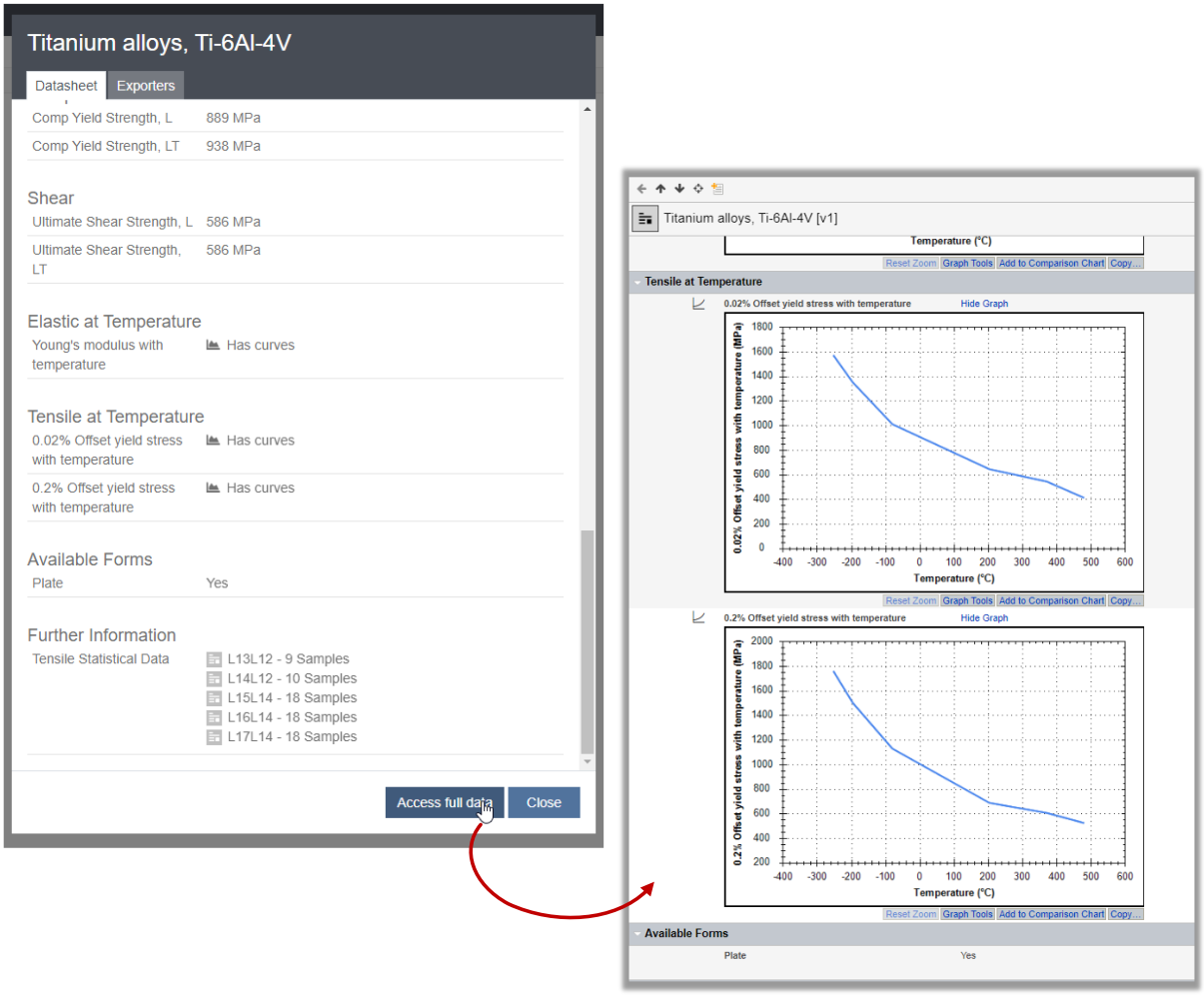
where “Explore” is the name of the folder where the config files are organized in MI:Admin:



See Section *7, Managing configuration files*, for more information.

6.2.3 MI:Viewer URL

An **Access full data** button at the bottom of Explore datasheets enables users to open a standalone, MI:Viewer-style datasheet, where additional data and tools may be available:



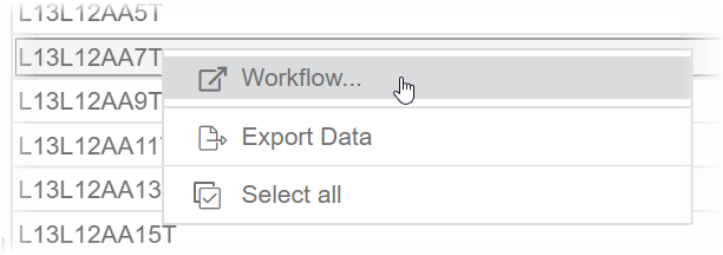
To enable this integration capability, the MI:Viewer application URL must be specified in **settings.preferences.json**, for example:

```
},
"viewerUrl": "http://my_miserver/mi",
"workflowUrl": "",
...
```

If no MI:Viewer URL is specified in **settings.preferences.json**, the **Access full data** button will not appear in any datasheets in the Explore application.

6.2.4 MI:Workflow URL

When the MI:Workflow Manager URL is specified in **settings.preferences.json**, Explore users can carry out workflow activities on selected records in the Workflow Manager via a **Workflow** option on the right-click menu:



Clicking this option will pass the list of selected records through to the Workflow Manager and open the application, showing all of the user's pending workflow actions on those records.

To enable this integration capability, both of the following configuration steps must be carried out:

1. The MI:Workflow URL must be specified in **settings.preferences.json** in the Settings Service, for example:

```
},
"viewerUrl": "http://my_miserver/mi",
"workflowUrl": "http://my_miserver/grantami/#/workflow",
...
```

2. The **Workflow** shortcut (right-click) menu option in the Explore application must be enabled in the data view configuration with the workflowEnabled setting.

6.3 Link Visualizer settings

The linkVisualization property that enables the Link Visualizer tool (see *4.10*) refers to *Link Visualizer configurations*, which control the layout and appearance of the visualization. These are defined in **settings.link-visualizer.json** in the Settings Service, and they are made available for users within a data view, as described in *4.10*.

Settings are defined as a list of named configurations. The names must be unique.

```
{
  "displayOptions": {
    "My config 1": {
      ……
    },
    “My config 2”: {
      ……
    }
  }
}
```

The contents of each named configuration is as follows:

Property	Value type	Description
`displayName`	String	Defines how the configuration will appear in the menu in the UI.
`tooltip`	String	Defines a tooltip that will appear on the menu item in the UI. (Optional)
`clusterByLinkType`	Boolean	Defines whether or not links of the same type are clustered together and represented by a node. • `true` = cluster the links of the same type into a node • `false` = show links of the same type as separate edges If you set this option to `true`, you may also want to hide line labels by default (`label = false`)
`layout`	String	Choose between the available supported layouts. • `"Organic"` – nodes are spread across the graph using a force-directed algorithm • `"Hierarchical"` – nodes appear in a structured tree style, arranged horizontally with the starting records on the left • `"VerticalHierarchical"` – nodes appear in a structured tree style, arranged vertically with the starting records at the top
`maxLinksToDisplayPerLinkType`	Integer	When a link type is added to the graph, if there are more than this number of records with that link type, the results will be clustered into a link node, even if `clusterByLinkType = false`. This improves performance and prevents the graph from becoming too cluttered. Users can then do any of the following: • filter the linked records to reduce their number so that they can be displayed; • view a list of the linked records and select specific records to display; • or choose to display all of the linked records anyway, provided there are no more than the maximum number specified by `maxLinksToDisplayWhenForced`. If a value is not specified, the default is 10.

Property	Value type	Description
`maxLinksToDisplayWhenForced`	Integer	When links are shown as a node because the number of linked records exceeds `maxLinksToDisplayPerLinkType` (above), users can choose to display the linked records anyway, provided there are no more than this maximum number. If a value is not specified, the default is 200.
`chartFormat`	Json	Contains formatting characteristics of the supported graph elements. • `startNode` – the record(s) which were selected when Link Visualizer was launched • `recordNode` – all record nodes which are not start nodes • `linkNode` – any node representing clustered links of the same type (if `clusterByLinkType = true`, or where the number of links exceeds `maxLinksToDisplayPerLinkType`) • `line` – any line joining two nodes Note: `startNode` and `linkNode` are optional. If omitted, these elements use the same styling as `recordNode`

The contents of the four `chartFormat` configurations are as follows:

Property	Value type	Applies to elements	Description

`shape`	String	`startNode`, `recordNode`, `linkNode`	Defines the shape of the node
			`"roundedRectangular"`
			`"rectangular"`
			`"angledRectangular"`

`color`	String	`startNode`, `recordNode`, `linkNode`, `line`	Defines the color of the text in the node or line label . The value may be any CSS color, by name, or expressed as a hex code, for example `"lightcoral"` or `"#F08080"` If omitted, the color defaults to white.

Property	Value type	Applies to elements	Description
`selectedColor`	String	`startNode`, `recordNode`, `linkNode`, `line`	Defines the color of the text in the node or line label when it is in a selected state. The value may be any CSS color, by name, or expressed as a hex code, for example `"lightcoral"` or `"#F08080"` If omitted, the color defaults to white.
`highlightColor`	String	`startNode`, `recordNode`, `linkNode`, `line`	Defines the color of the text in the node or line label when it is in a highlighted state. The value may be any CSS color, by name, or expressed as a hex code, for example `"lightcoral"` or `"#F08080"` If omitted, it defaults to white.
`background`	String	`startNode`, `recordNode`, `linkNode`, `line`	Defines the background color of the graph element. The value may be any CSS color, by name, or expressed as a hex code, for example `"lightcoral"` or `"#F08080"`
`selectedBackground`	String	`startNode`, `recordNode`, `linkNode`, `line`	Defines the background color of the graph element when it is in a selected state. The value may be any CSS color, by name, or expressed as a hex code, for example `"lightcoral"` or `"#F08080"`
`highlightBackground`	String	`startNode`, `recordNode`, `linkNode`, `line`	Defines the background color of the graph element when it is in a highlighted state. The value may be any CSS color, by name, or expressed as a hex code, for example `"lightcoral"` or `"#F08080"`
`lineStyle`	String	`line`	Defines the preferred style of lines linking nodes. `"line"` – nodes are joined with straight lines `"arrow"` – nodes are joined with straight lines with a directional arrow `"rectilinear"` – nodes are joined with perpendicular straight lines that meet at

Property	Value type	Applies to elements	Description
			right angles. If using this option, you may also want to hide link labels by default (`label = false`) . `"polyLineIfParallel"` – nodes are joined with angled lines when there are multiple lines between two nodes. Otherwise, straight lines are used.
`label`	Boolean	`line`	Defines whether or not to add a text label to the lines between nodes. [Note: this option is redundant if `clusterByLinkType = true`, as the same information is already on the link node.]

Example

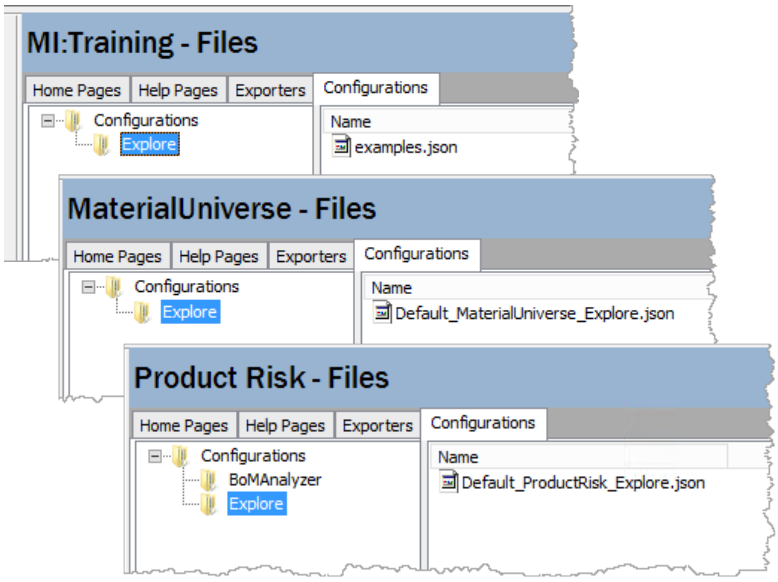
```
{
  "displayOptions": {
    "My config 1": {
      "displayName": "Configuration 1",
      "tooltip": "Useful for config 1",
      "clusterByLinkType ": false,
      "layout": "Hierarchical",
      "maxLinksToDisplayPerLinkType": 10,
      "chartFormat": {
        "startNode": {
          "shape": "roundedRectangular",
          "background": "seagreen",
          "selectedBackground": "springgreen",
          "highlightBackground": "springgreen",
          "color": "#fff",
          "selectedColor": "#fff"        },
        "recordNode": {
          "shape": "rectangular",
          "background": "#51739D",
          "selectedBackground": "#53A0ff",
          "highlightBackground": "#519D7B",
          "color": "#fff",
          "selectedColor": "#fff",
          "highlightColor": "#fff"
        },
        "linkNode": {
          "shape": "angledRectangular",
          "background": "#ddd",
          "color": "#717171",
          "selectedBackground": "#474c55",
          "selectedColor": "#fff",
          "highlightBackground": "#519D7B"
```

```
        },
        "line": {
          "background": "#DDD",
          "selectedBackground": "#474c55",
          "highlightBackground": "#519D7B",
          "lineStyle": "rectilinear",
          "label": false
        }
      }
    }
  }
}
```

7 Managing configuration files

Data views are defined in one or more configuration files, which are typically stored in the database alongside database home page files, Attribute help files, and FEA exporter files, and managed on the *Files > Configurations* page in the MI:Admin tool. (It is possible to use a search.json file stored in a folder in the Explore application instead of embedded configuration files; see Section 7.1 below.)

By default, Explore looks for configuration files in each GRANTA MI database. Configuration files can have any name, but they must be located in a folder with the same name in each database; by default, this is *Explore*. For example:

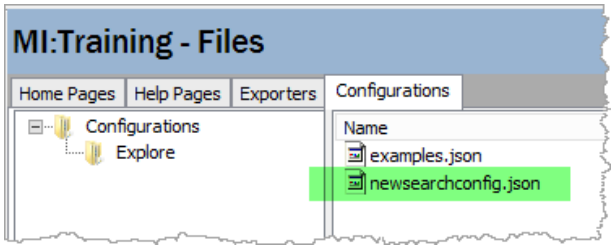


This folder name is specified under `configPaths` in **settings.preferences.json** in the Settings Service:

```
{
  "configPaths": {
    "explore": "Explore",
    "homePage": "HomePage"
  },
```

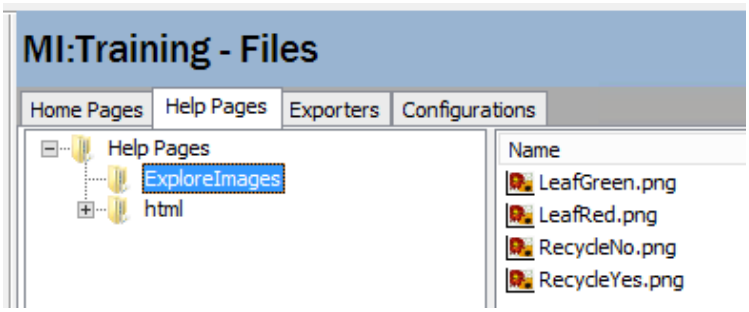
Only one configuration path may be specified. This means that the same folder name must be used for the configuration files in all databases, as shown above.

Users with MI Administrator privileges can upload (*Import*) and download (*Export*) configuration files in MI:Admin, and also validate the configuration file JSON. To edit a configuration file, you need to download it, edit it locally, and then import it back into the database, for example:



For further information about managing configuration files stored in GRANTA MI databases, see the *MI:Admin Help*.

Note that image files referenced in Explore application configuration files should be stored under *Files > Help Pages* page in MI:Admin, for example:



See Section *5.4.3, Representing data values with images*, for full details of required configuration to allow image files to be used to represent data values in the Explore application.

7.1 Using an application configuration file on the server file system

Although Explore application configuration files are typically stored in the database to which they apply , it is possible to use a single configuration file stored on the server file system to define the available data views for application users.

This file must be called *search.json*, and it must be located in the *settings* folder under the GRANTA MI web application root directory, typically:

C:\inetpub\wwwroot\grantami\settings\search.json

To have the Explore application look for data view configurations in this file instead of in your databases, you must edit `configPaths` in **settings.preferences.json** in the Settings Service and delete the `explore` line highlighted here:

```
{
  "configPaths": {
    "explore": "Explore",
    "homePage": "HomePage"
  },
```

Note also that the databaseKey must be specified in each data view configuration defined in the search.json file.

8 Explore configuration file reference

Explore application data configuration files are specified in JSON format.

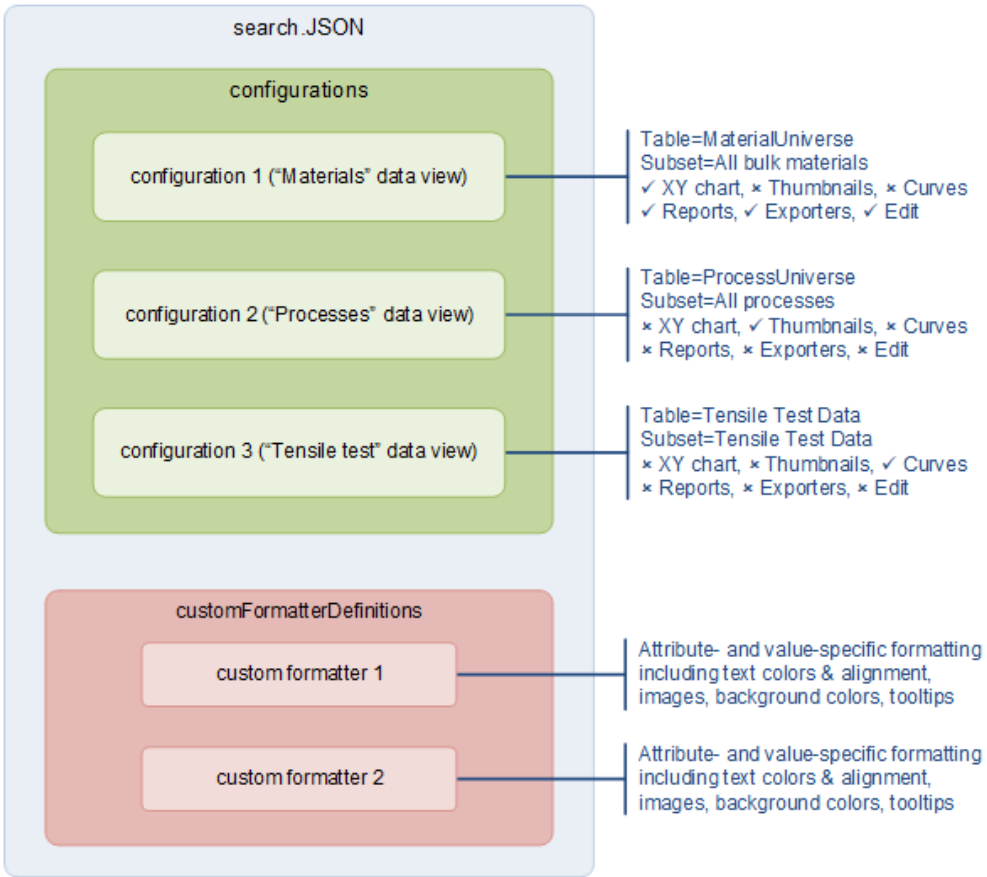
- Data is specified in name/value pairs consisting of a field name (in double quotes), followed by a colon, followed by a value.
- Data is separated by commas.
- Curly braces { } hold objects.
- Square brackets [ ] hold arrays.

If the JSON is not correctly formatted, it will not be loaded into the application. To avoid problems, always validate the basic JSON syntax before you deploy the configuration file to your GRANTA MI environment using a tool such as JSONLint.

8.1 Configuration file structure

Each Explore configuration file may include the following top-level objects:

- `configurations` defines an array of *configuration* objects, each of which defines a *data view*. See Section *8.2*.
- `customFormatterDefinitions` defines an array of custom formatter objects, each of which specifies custom formatting rules. See Section *8.3*.



8.2 Data view configuration properties

Each data view *configuration* object may include the following properties.

Property	Summary
`key`	Mandatory. Specifies a string which identifies the data view configuration. This must not include spaces or special characters. See also: Data view key
`table`	Mandatory. Specifies the name of the GRANTA MI Table where the data for this data view is stored. See also: Database, Table, and Subset
`searchLayout`	Mandatory. Specifies the Layout that defines the search Attributes for the data view. The Layout must be in the Table specified in the data view `table` property. See also: Search panel
`configSwitching`	Optional. Defines the data view switch and modify capability. Possible values: • `hidden` : disables the Switch data view dialog for users • `keyOnly` : users can switch data view but cannot edit a data view • `keyInGroup` : users can switch data view only between views in the same group as the default view, and cannot edit data views • `keyOrData` : users can switch data view and change database, Table, and Layout, but cannot edit advanced options • `advanced` : (Default) Users can switch and edit a data view. The `configSwitching` setting is specified in (and read from) the default data view in the configuration file. If there is no default data view, this setting should be specified in the first data view in the configuration file. If `configSwitching` is not specified in the data view configuration, the default behavior is to enable all data switch and edit functionality (= `advanced`). See also: Data view switch functionality

Property	Summary

`curves`	Optional. Defines default plot axis properties for the Curves view.
	
	Properties (all optional)
	
	• `yAttribute` : specifies the Attribute shown on the Y-axis. If not specified, the first Attribute found will be used.
	• `xParameter` : specifies the parameter shown on the X-axis.
	• `xAxisLogarithmic` : specifies the X-axis scale type.
	  ○ `true` = logarithmic scale
	  ○ `false` = linear scale
	• `yAxisLogarithmic` : specifies the Y-axis scale type.
	  ○ true = logarithmic scale
	  ○ `false` = linear scale
	
	Note that if loadDataOnDemand = `true`:
	
	• Functional Attributes will only be visible on the Curves view if the Attribute is marked as searchable in the MI:Admin database schema.
	• The axis scales will always be linear regardless of the data view configuration setting.
	
	See also: Curves view

`databaseKey`	Specifies the GRANTA MI database key (dbkey). If not specified in the data view configuration, the value defaults to the key of the database that the configuration file is stored with. Database keys are not case-sensitive in GRANTA MI. A database key must be specified if the configuration file is not embedded in the database (see Managing configuration files). See also: Database, Table, and Subset
`dataSheetLayout`	Optional. Specifies the name of the Layout that defines the Attributes shown on datasheets. The Layout must be in the Table specified in the data view `table` property. See also: Datasheets

Property	Summary
`default`	Optional. A Boolean that identifies the data view as the default data view. This default data view is the one loaded on starting the application if no query string is specified in the URL. Possible values: `true` , `false` Default value: `false` If more than one data view configuration includes `"default": true`, the first one in the configuration file will be treated as the default. See also: Data view key
`description`	Optional. Specifies a description of the data view which is shown in the Switch data view dialog. See also: Data view name and description
`displayName`	Optional. Specifies the name of the data view as displayed in the header of the Explore application and in the Switch data view dialog. See also: Data view name and description
`editableDatasheetLayout`	Optional. Specifies the name of the Layout that defines the Attributes available when adding or editing data in the data view. The Layout must be in the Table specified in the data view `table` property. If `editableDatasheetLayout` is not specified in the data view configuration, users will not be able to edit data when using the data view. See also: Data add and edit functionality
`exportersDisabled`	Optional. A Boolean that enables FEA data exporter functionality in the data view. Possible values • `true` : data exporter functionality will not be available in the data view • `false` : data exporter functionality will be enabled in the data view (Export data button on the toolbar, Exporters tab on datasheets) Default value: `false` (data exporter functionality is enabled) See also: Data exporters

Property	Summary
`exportersUseCurrentUnitSystem`	Optional. A Boolean that specifies which Unit System will be used when exporting FEA data in the data view. Possible values • `true` : use the Unit System currently selected in the application Preferences • `false` : use the Unit System specified in the exporter .exp configuration file. Default value: `false` (the exporter Unit System is used) See also: Data exporters
`exporterApplicability`	Optional. Defines a string which is matched against the `Applicability` tag value in the exporter configuration .exp file. When the value specified here matches the `Applicability` tag value in the exporter config, then the exporter will be shown in the Available exporters list in the data view. If the values do not match, the exporter will not be listed. See also: Hiding application-specific exporters
`groups`	Optional. Defines an array that specifies the groups of which the data view is a member. Syntax: `"groups": ["group1", "group2"]` A data view may belong to one group, many groups, or no groups. If `configSwitching` is set to `keyInGroup`, then only the data views in the same group as the default data view will be available in the Switch data views dialog.
`linkVisualization`	Optional. Defines properties for the Link Visualizer app, as accessed from this data view. Possible values • `false` : Link Visualizer is not enabled • `true` : Link Visualizer is enabled, with no default configuration specified • the name of a Link Visualizer configuration, e.g., `"myConfig1"` : Link Visualizer is enabled, with the named configuration selected by default, and users can

Property	Summary
	select from a list of all valid configurations defined in the file link-visualizer.json • an array of names of Link Visualizer configurations, e.g., `["myConfig1","myConfig2"]` : Link Visualizer is enabled, with the first named configuration selected by default, and users can select from a list of the named configurations only Default value: `false` Link Visualizer configurations are defined in settings.link-visualizer.json in the Settings Service; see *6.3* See also: Link Visualizer
`loadDataOnDemand`	Optional. A Boolean that specifies whether data is loaded only when required ("data on demand"), or whether all data is loaded on opening the data view. Possible values • `true` : data is loaded on demand • `false` : all data is loaded on opening the data view Default value: `false` See: Loading data on demand
`logSliders`	Optional. A Boolean that specifies whether or not slider controls in the Search panel are logarithmic when the data spread is larger than one order of magnitude. Possible values • `true` : slider controls use a logarithmic scale when required • `false` : slider controls will always use a linear scale Default value: `false` (sliders always have linear scale) See: Filter controls in the Search panel
`newRecordLocation`	Optional. Specifies the folder path where records created by users will be placed. The folder will be created if it does not already exist in the database tree. This path may include *placement tokens* of the format {*attributename*} which allow Attribute values to be inserted into the folder path; see Options for placement of new records.

Property	Summary
	If `newRecordLocation` is not specified in the data view configuration, new records will be placed in the top-level (root) folder by default. Note that, to be able to add or edit data, editableDatasheetLayout must also be specified.
`newRecordTransformation`	Optional. Defines an array of transforms for generating record names. Array elements • `Counter` : increments an integer Attribute with every new record created. • `Id` : concatenates two Attribute values into a third Attribute. • `RecordNameConcatenator` : concatenates two Attribute values into the record name. See Advanced options for auto-naming of new records.
`newRecordPaddingOptions`	Optional. Defines padding options for autogenerated record names. Properties • `padLength` : the padding length (number of digits) • `padString` : a padding string See also: Advanced options for auto-naming of new records.
`primaryImageAttributeName`	Optional. Specifies the Attribute that provides the primary image used in the Thumbnails view. See also: Thumbnails view
`reportsDisabled`	Optional. A Boolean that enables reporting functionality in the data view. Possible values • `true` : reporting functionality will not be available • `false` : reporting functionality will be enabled (Report button on the toolbar, Reports tab on datasheets) Default value: `false` (reporting functionality is enabled) See also: Reports

Property	Summary
`searchListLayout`	Optional. Specifies the name of a Layout that defines the Attributes/columns initially visible in the List view. The Layout must be in the Table specified in the data view `table` property. If `searchListLayout` is not specified in the data view configuration, only the record name column is shown by default. See also: List view
`showBlanks`	Optional. A Boolean that specifies whether or not Attributes with no data are shown when viewing datasheets. Possible values • `true` : Attributes with no data are visible in datasheets • `false` : Attributes with no data do not appear in datasheets Default value: `true` (show empty Attributes) Note that this option only applies when *viewing* datasheets; empty Attributes are always shown when editing data. See also: Showing/hiding Attributes with no value
`subset`	Optional. Specifies the name of the Subset that contains the records for this data view. This Subset must be in the Table specified in the data view `table` property. If the `subset` property is not specified in the data view configuration, search and filter operations will be applied to all records in the Table. If data edit functionality is enabled in the data view (i.e. if an editableDatasheetLayout is specified), all new records will be placed into this Subset. See also: Database, Table, and Subset

Property	Summary
`tabularEditingEnabled`	Optional. A Boolean that specifies whether or not Tabular data can be edited in this data view. Possible values • `true` : Tabular data can be edited • `false` : Tabular data cannot be edited; users may view Tabular data but not edit it Default value: `false` (Tabular data cannot be edited) Note that, to be able to add or edit data, editableDatasheetLayout must be specified in the data view configuration, and the user must be a member of a role with Write access to the database. See also: Viewing and editing tabular data
`useCustomFormatters`	Optional. Maps Attributes to custom formatter definitions specified under customFormatterDefinitions. Only Attributes of these types can be mapped: • Logical (Boolean) • Discrete • Numeric (Integer, Point, Range) Attributes. % is the only permitted unit; data with any other unit cannot be formatted. See also: Custom formatting rules
`workflowEnabled`	Optional. A Boolean that enables a Workflow option on the right-click menu in the Explore application, allowing users to launch the MI:Workflow application from within Explore. Possible values • `true` : include a Workflow option on the right-click menu • `false` : do not include a Workflow menu option Default value: `false` (no Workflow menu option) Note that a valid MI:Workflow URL must also be specified in settings.preferences.json for this menu option to be available; see MI:Workflow .

8.3 Custom data formatting properties

Formatting options for styling values in datasheets, on the List tab, and on the Scatter plot tab are defined with the `customFormatterDefinitions` top-level property. See Data formatting options for an overview of formatting capabilities.

`customFormatterDefinitions` defines an array of *custom formatter* objects, each of which specifies a collection of formatting settings. Custom formatter definitions are mapped to specific Attributes with the `useCustomFormatters` property within the data view configuration.

Syntax

```
...
      "customFormatterDefinitions": {
            "formatter_id": {
                  "attributeFormatter": {
                        "rangeMatch": "min|mid|max",
                        "datasheet": {
                              ... // See below
                        },
                        "list": {
                              ... // See below
                        }
                  }
                  "valueFormatters": [
                        "match": "matchstring",
                        "datasheet": {
                              ... // See below
                        },
                        "list": {
                              ... // See below
                        },
                        "chart": {
                              ... // See below
                        }
                  ]
            }
      }
...
```

8.3.1 attributeFormatter properties

`attributeFormatter` defines formatting that will be applied to the Attribute wherever it appears in datasheets and/or lists. This can be overridden by data-dependent formatting specified under valueFormatters.

Syntax

```
...
      "customFormatterDefinitions": {
            "formatter_id": {
                  "attributeFormatter": {
                        "rangeMatch": "min|mid|max",
                        "dataSheet": {
                              "centerContents": false,
```

```
                        "renderValue": false,
                        "tooltipValue": false
                },
                "list": {
                        "centerContents": false,
                        "columnHeader": false,
                        "columnWidth": value,
                        "renderValue": false,
                        "tooltipValue": false
                }
        },
        "valueFormatters": [
        ...
```

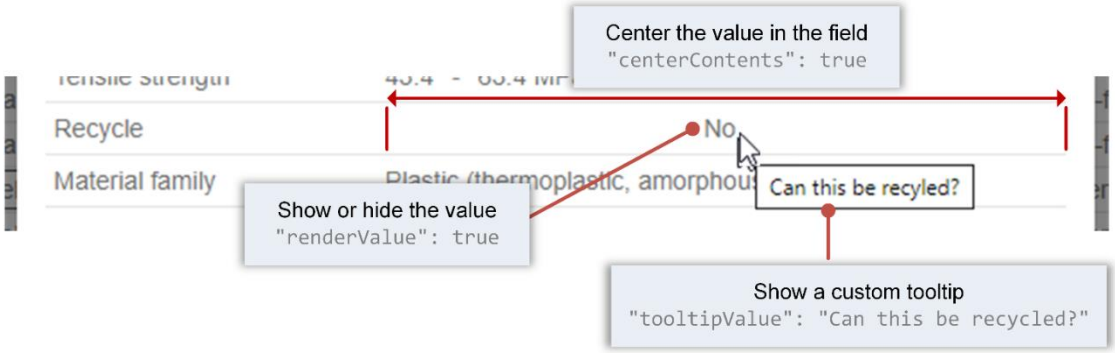
The following Table describes the top-level properties under `attributeFormatter`.

Property	Description
rangeMatch	Mandatory for Range Attributes only. Specifies where in the data range all the *matchvalues* defined under valueFormatters are located: at the bottom, top or mid-point of the range. Possible values: `min`, `mid`, `max`
datasheet	Optional. Defines the formatting to be applied on datasheets.
list	Optional. Defines the formatting to be applied on the List tab.

Attribute-level styling properties for datasheets

The optional `datasheet` property under `attributeFormatter` specifies Attribute-level formatting in datasheets. It may include the following properties.

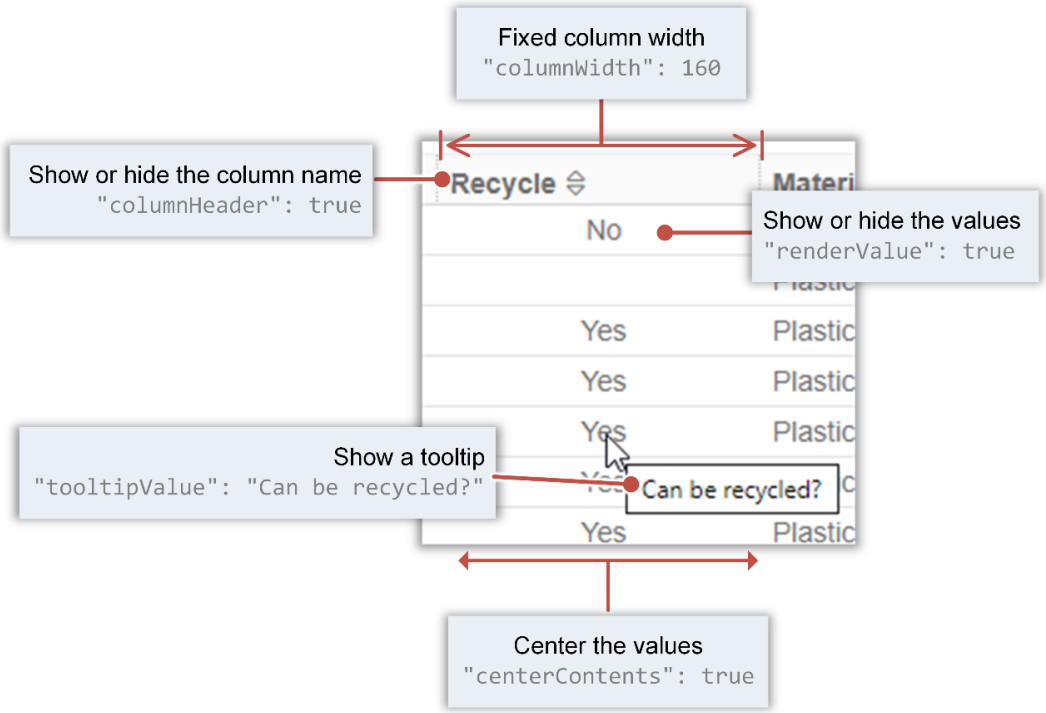
Property	Value type	Description	Default value
centerContents	Boolean	Optional. Center the value in the field.	false
renderValue	Boolean or String	Optional. `false` = Show no value `true` = Show the datum value `string` = Show the specified string	false
tooltipValue	Boolean or String	Optional. `false` = Do not show a tooltip `true` = Show the datum value in the tooltip `string` = Show the specified string	false



Attribute-level styling properties for the List view

The optional `list` property under `attributeFormatter` specifies Attribute-level formatting in the List view. It may include the following properties.

Property	Value type	Description	Default value
centerContents	Boolean	Optional. Center the value in the column or field.	false
renderValue	Boolean or String	Optional. `false` = Show no value `true` = Show the datum value `string` = Show the specified string	false
tooltipValue	Boolean or String	Optional. `false` = Do not show a tooltip `true` = Show the datum value in the tooltip `string` = Show the specified string	false
columnHeader	Boolean or String	Optional. `false` = No column header `true` = Use the Attribute name as the column header `string` = Use the specified string	true
columnWidth	Number	Optional. Specifies a fixed width for the column in pixels.	



8.3.2 valueFormatters properties

The `valueFormatters` property defines data-dependent formatting that is applied only when the Attribute datum value matches the specified *matchvalue*, overriding any Attribute-level format settings specified under `attributeFormatter`.

For Range data, the *matchvalue* is taken as the range minimum, mid or maximum value, as determined by the `rangeMatch` property under `attributeFormatter`.

`valueFormatters` includes an ordered array of JSON objects, each specifying the formatting for a different data value in list, datasheet and/or chart views.

Syntax

```
...
      "customFormatterDefinitions": {
            "formatter_id": {
                  "attributeFormatter": {
                        "rangeMatch": "min|mid|max",
                        ...
                        }
                  "valueFormatters": [
                        {
                              "match": "matchvalue",
                              "list": {
                                    "centerContents": false,
                                    "renderValue": false,
                                    "tooltipValue": false,
                                    "iconUrl": "url",
                                    "textColor": "color",
                                    "cellColor": "color"
                              },
                              "datasheet": {
                                    "centerContents": false,
                                    "renderValue": false,
                                    "tooltipValue": false,
                                    "iconUrl": "url",
                                    "textColor": "color",
                                    "cellColor": "color"
                              },
                              "chart": {
                                    "renderColor": "color"
                              }
                        }
                  ]
            ...
```

The following Table describes the top-level properties under `valueFormatters`:

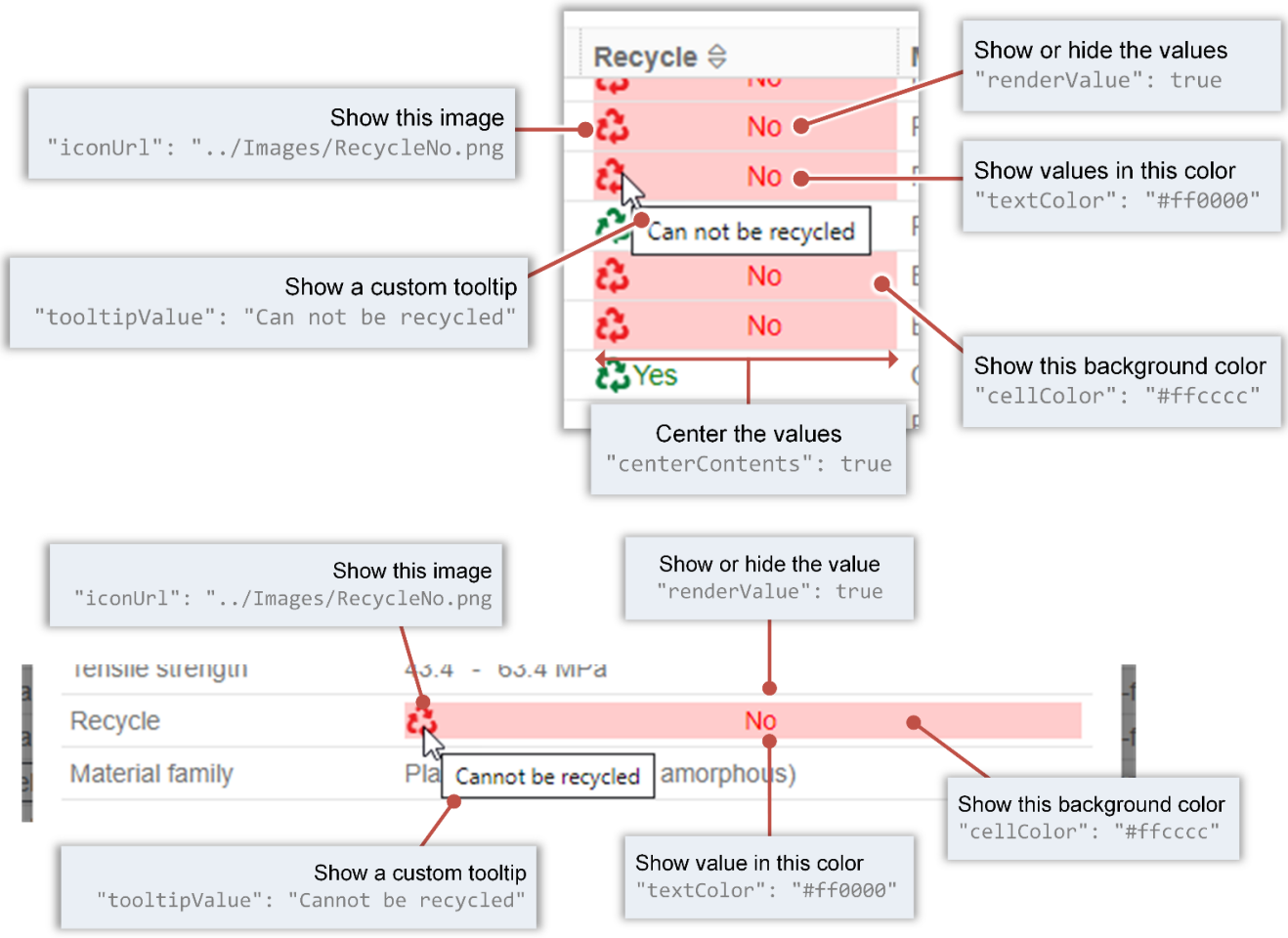
Property	Description
match	Mandatory. Specifies the match value. Formatting will be applied when the Attribute value either: • Matches the *matchvalue* exactly (for Discrete and Logical Attributes) • Is Greater than or equal to the *matchvalue* (for Numeric Attributes). For Range data, the *matchvalue* is matched to the range minimum, mid, or maximum value, depending on the `rangeMatch` property specified under AttributeFormatter.
list	Optional. Defines the formatting to be applied to the value in the List view.
datasheet	Optional. Defines the formatting to be applied to the value in datasheets.

Property	Description
chart	Optional. Defines the formatting to be applied to the value in scatter plots.

Value-dependent styling for datasheets and lists

The `datasheet` and `list` properties under `valueFormatters` are used to specify datum-level formatting for the Attribute values in datasheets and in the List tab. They may include the following format properties.

Property	Value type	Description	Default value
centerContents	Boolean	Optional. Center the value in the cell/field.	false
renderValue	Boolean or String	Optional. FALSE = Show no value TRUE = Show the datum value STRING = Show the specified string	false
tooltipValue	Boolean or String	Optional. FALSE = Do not show a tooltip TRUE = Show the datum value in the tooltip STRING = Show the specified string in the tooltip	false
iconUrl	String	Optional. Specifies the URL for an image that will accompany/replace the value (depending on the value of `renderValue`). See Section *5.4.3* for information on the required configuration to enable use of images.	
textColor	Any valid CSS color value	Optional. Specifies the color of the rendered text (depending on the value of `renderValue`)	The web page font color
cellColor	Any valid CSS color value	Optional. Specifies the background color of the cell/field.	



Value-dependent styling for scatter plots

The `chart` property under `valueFormatters` is used to specify datum-level formatting for the Attribute values in scatter plots. It may include one property:

Property	Value type	Description
renderColor	Any valid CSS color value	Optional. Specifies the color of the point/bubble for this value when the Attribute is selected as the Color axis.

See Styling in Scatter plots for examples of usage.

8.3.3 Examples

There are two custom formatter definitions in the example below:

- `recyclingBool` defines formatting for a Boolean “recyclability” property in the List and Scatter plot views
- `polymertypeDiscrete` defines formatting for different polymer types in the Scatter plot view

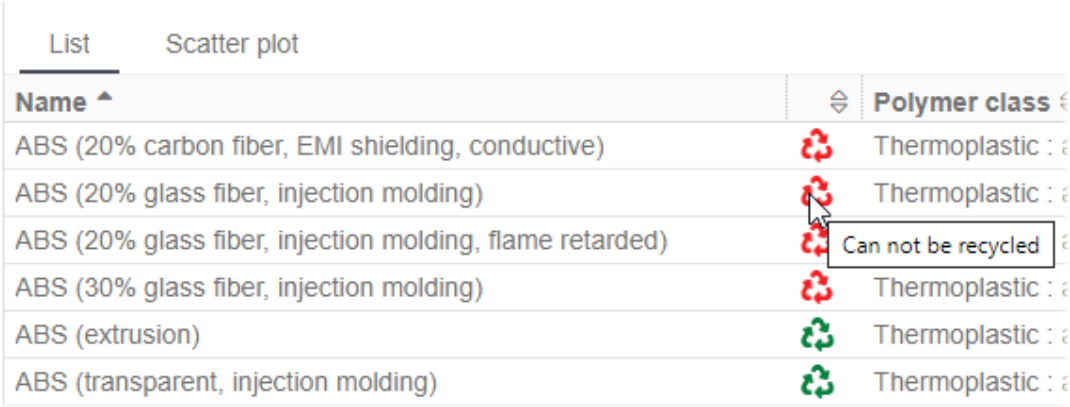
```
	"customFormatterDefinitions": {
		"recyclingBool": {
			"attributeFormatter": {
				"list": {
					"columnHeader": false,
					"columnWidth": 40
				}
			},
			"valueFormatters": [
				{
					"match": true,
					"list": {
						"iconUrl":
"/mi/help.ashx/MI_Training/ExploreImages/RecycleYes.png",
						"tooltipValue": "Can be recycled"
					},
					"chart": {
						"renderColor": "green"
					}
				},
				{
					"match": false,
					"list": {
						"iconUrl":
"/mi/help.ashx/MI_Training/ExploreImages/RecycleNo.png",
						"tooltipValue": "Can not be recycled"
						},
					"chart": {
						"renderColor": "red"
					}
				}
			]
		},
		"polymertypeDiscrete": {
			"valueFormatters": [
				{
					"match": "ABS",
					"chart": {
						"renderColor": "#B2D732"
					}
				},
				{
					"match": "PMMA",
					"chart": {
						"renderColor": "#4424D6"
					}
				},
				{
					"match": "PVC-P (elastomer)",
					"chart": {
						"renderColor": "#FC600A"
					}
				}
			]
		}
	}
```

Attributes in the Polymers data view configuration are then mapped to these custom formatters with this useCustomFormatters property:

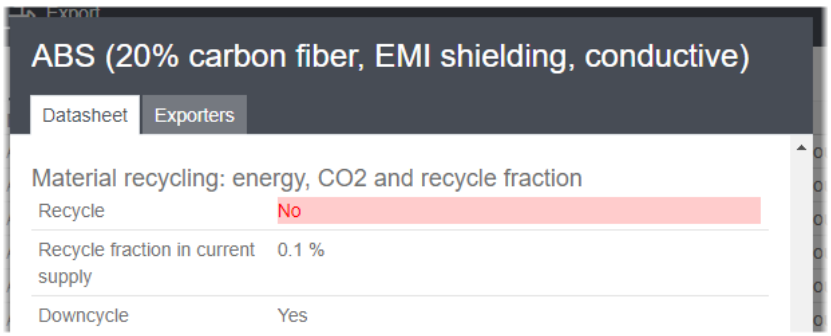
```
{
	"key": "polymers",
	...
	"useCustomFormatters": {
		"Recycle": "recyclingBool",
		"Polymer type": "polymertypeDiscrete"
		}
	},
```

The effect of applying the formatting shown above is as follows:

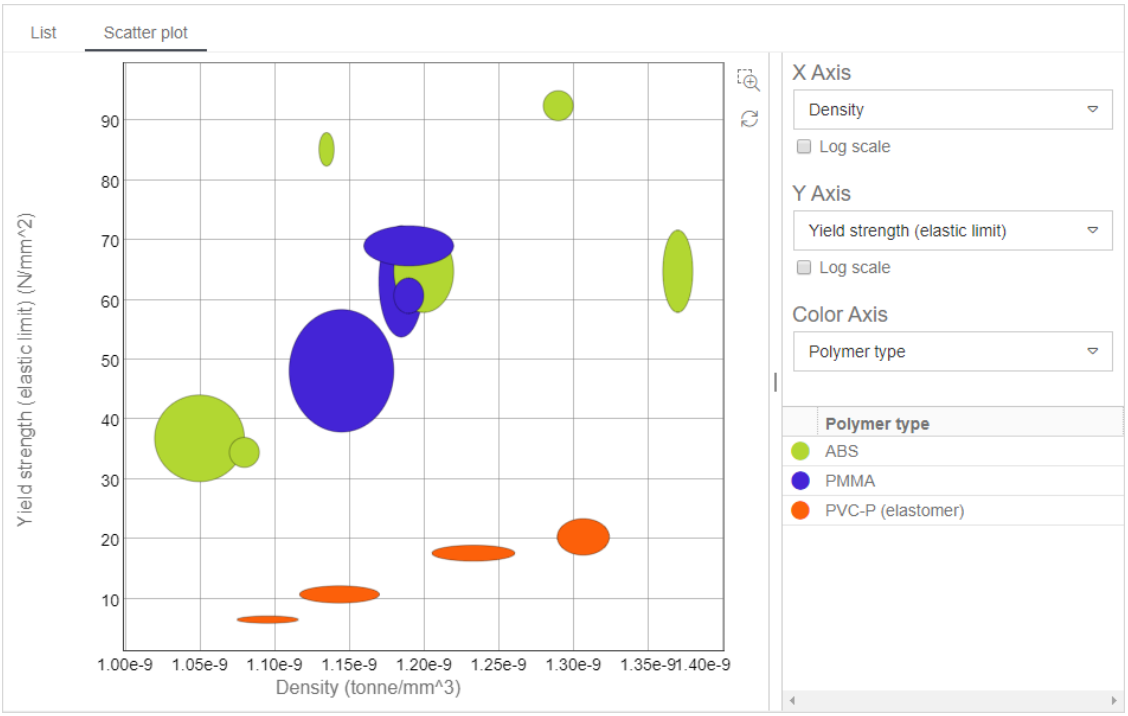
***Recycle** Attribute in the List view:*



***Recycle** Attribute in a datasheet:*



Polymer type *as the Color Axis variable in a Scatter plot:*



9 Sample configuration file

This simple configuration file defines some data views for the MI:Training database to illustrate a range of different configuration options.

The *training-materials* and *training-polymers* data view configurations in this file both use images for custom formatting that have been uploaded to the MI:Training database (dbkey=*MI:Training*) Help Pages and referenced with the URL `/mi/help.ashx/MI_Training/RecycleYes.png`, where *miserver* is the name of the application web server.

```
{
	"configurations": [
	{
			"key": "training-materials",
			"displayName" : "MI:Training Materials",
			"default": true,
			"groups": ["MI:Training examples"],
			"table": "MaterialUniverse",
			"loadDataOnDemand": false,
			"description": "Explore MaterialUniverse data in the MI:Training
database",
			"searchLayout": "All bulk materials",
			"dataSheetLayout": "All bulk materials",
			"editableDatasheetLayout": "All bulk materials",
			"searchListLayout": "Explore list layout Materials",
			"newRecordLocation": "NEW RECORDS/{Base}",
			"subset": "All bulk materials",
			"logsliders": true,
			"linkVisualization": true,
			"xyChart": {
				"xAttribute": "Density",
				"yAttribute": "Yield strength (elastic limit)",
				"preventAxisChange": false,
				"xAxisLogarithmic": true,
				"yAxisLogarithmic": true
			},
			"showBlanks": false,
			"reportsDisabled": false,
			"exportersDisabled": false,
			"tabularEditingEnabled": true,
			"configSwitching": "keyOnly",
			"useCustomFormatters": {
				"Recycle": "recyclingBool",
				"Recycle fraction in current supply": "recyclingPercent"
			}
		},
			{
			"key": "training-tensiletest",
			"displayName" : "MI:Training Tensile Test Data",
			"default": false,
			"groups": ["MI:Training examples"],
			"table": "Tensile Test Data",
			"loadDataOnDemand": true,
```

```
                    "description": "Explore tensile test data in the MI:Training
database. \nThe data does not load on startup; you must click the Search button to load
data.",
                    "searchLayout": "Tensile Test Data",
                    "dataSheetLayout": "Tensile Test Data",
                    "subset": "Tensile Test Data",
                    "logsliders": true,
                    "curves": {
                           "yAttribute": "Tensile Response (11 axis)",
                           "xParameter": "Strain",
                           "xAxisLogarithmic": false,
                           "yAxisLogarithmic": true
                    },
                    "showBlanks": false,
                    "reportsDisabled": true,
                    "exportersDisabled": false,
                    "exporterApplicability": "MIViewer"
             },
             {
                    "key": "training-processes",
                    "displayName" : "MI:Training Processes",
                    "default": false,
                    "groups": ["MI:Training examples"],
                    "loadDataOnDemand": false,
                    "description": "Explore ProcessUniverse data in the MI:Training
database",
                    "table": "ProcessUniverse",
                    "searchLayout": "All processes",
                    "dataSheetLayout": "All processes",
                    "subset": "All processes",
                    "logSliders": true,
                    "primaryImageAttributeName": "Process schematic",
                    "xyChart": {
                           "xAttribute": "Capital cost",
                           "yAttribute": "Mass range",
                           "preventAxisChange": false,
                           "xAxisLogarithmic": true,
                           "yAxisLogarithmic": true
                    },
                    "showBlanks": false,
                    "reportsDisabled": true,
                    "exportersDisabled": true,
                    "tabularEditingEnabled": true
             },
             {
                    "key": "training-polymers",
                    "displayName": "MI:Training Polymers",
                    "description": "Explore polymer data in the MI:Training database",
                    "default": false,
                    "groups": ["MI:Training examples"],
                    "table": "MaterialUniverse",
                    "searchLayout": "Polymers",
                    "subset": "Polymers",
```

```
                    "dataSheetLayout": "Polymers",
                    "searchListLayout": "Explore list layout Polymers",
                    "xyChart": {
                           "colorAttribute": "Polymer type",
                           "preventAxisChange": false,
                           "xAxisLogarithmic": false,
                           "yAxisLogarithmic": false,
                           "xAttribute": "Density",
                           "yAttribute": "Yield strength (elastic limit)"
                    },
                    "logSliders": false,
                    "showBlanks": false,
                    "reportsDisabled": true,
                    "exportersDisabled": true,
                    "tabularEditingEnabled": true,
                    "loadDataOnDemand": false,
                    "useCustomFormatters": {
                           "Recycle": "recyclingBool",
                           "Polymer type": "polymertypeDiscrete"
                    }
             }
      ],
      "customFormatterDefinitions": {
             "recyclingBool": {
                    "attributeFormatter": {
                           "list": {
                                  "columnHeader": false,
                                  "columnWidth": 40
                           }
                    },
                    "valueFormatters": [
                           {
                                  "match": true,
                                  "list": {
                                         "iconUrl":
"/mi/help.ashx/MI_Training/ExploreImages/RecycleYes.png",
                                         "tooltipValue": "Can be recycled"
                                  },
                                  "chart": {
                                         "renderColor": "green"
                                  }
                           },
                           {
                                  "match": false,
                                  "list": {
                                         "iconUrl":
"/mi/help.ashx/MI_Training/ExploreImages/RecycleNo.png",
                                         "tooltipValue": "Can not be recycled"
                                         },
                                  "chart": {
                                         "renderColor": "red"
                                  }
                           }
```

```
                ]
        },
        "polymertypeDiscrete": {
                "valueFormatters": [
                        {
                                "match": "ABS",
                                "chart": {
                                        "renderColor": "#B2D732"
                                }
                        },
                        {
                                "match": "PMMA",
                                "chart": {
                                        "renderColor": "#4424D6"
                                }
                        },
                        {
                                "match": "PVC-P (elastomer)",
                                "chart": {
                                        "renderColor": "#FC600A"
                                }
                        }
                ]
        },
        "recyclingPercent":{
                "attributeFormatter": {
                        "rangeMatch": "min"
                        },
                "valueFormatters": [
                        {
                                "match": 0.1,
                                "list": {
                                        "cellColor": "#cc3300",
                                        "renderValue": true,
                                        "textColor": "#ffffff"
                                        },
                                "datasheet": {
                                        "textColor": "#b22222",
                                        "renderValue": true
                                        },
                                "chart": {
                                        "renderColor": "#cc3300"
                                        }
                        },
                        {
                                "match": 3.5,
                                "list": {
                                        "cellColor": "#ffc266",
                                        "renderValue": true,
                                        "textColor": "white"
                                        },
                                "datasheet": {
                                        "textColor": "#b8850b",
```

```
                                        "renderValue": true
                                        },
                                "chart": {
                                "renderColor": "#ffc266"
                                }
                        },
                        {
                                "match": 20.0,
                                "list": {
                                        "cellColor": "#66cc00",
                                        "renderValue": true,
                                        "textColor": "white"
                                        },
                                "datasheet": {
                                        "textColor": "#3cb371",
                                        "renderValue": true
                                        },
                                "chart": {
                                        "renderColor": "#3399ff"
                                }
                        },
                        {
                                "match": 39.0,
                                "list": {
                                        "cellColor": "#196619",
                                        "renderValue": true,
                                        "textColor": "#ffffff"
                                        },
                                "datasheet": {
                                        "textColor": "#228b22",
                                        "renderValue": true
                                },
                                "chart": {
                                        "renderColor": "#196619"
                                }
                        }
                ]
        }
    }
}
```

Appendix A. Troubleshooting

A.1 View the log file

Event information for the Explore application, which may be useful when troubleshooting problems, is written to the Service Layer trace log file similar to this:

C:\inetpub\wwwroot\mi_servicelayer\logs\MIServiceLayer.TraceMessages.csv

This is a CSV file which can be opened using Microsoft® Excel® or another program that reads files in CSV format. For each event logged, the file includes information about the application connecting to the Service Layer, the event type and description, the IP address of the client, and client browser details.

A.2 Enable pop-ups in your browser

If your browser is configured to block pop-ups then some features in the Explore application, such as Reports, may be blocked.

To enable pop-ups in Google Chrome:

1. Go to **Settings**.
2. Scroll down to **Advanced**, and under Privacy and security, click **Content settings**.
3. Click **Popups**. Edit the settings to either allow pop-ups globally, or add the *grantami* website URL e.g. http://*<mi_server>*/grantami/

To enable pop-ups in Firefox:

1. Go to **Options**.
2. Click **Privacy and Security**, and scroll down to **Permissions**.
3. Either clear the **Block pop-up windows** option, or click **Exceptions** and add the *grantami* website URL e.g. http://*<mi_server>*/grantami/