Ansys GRANTA MI 2021 R1

GRANTA MI Schema Guide

Published in the U.S.A.

Contents

1 About this document

This document describes the GRANTA MI schema and is intended as a reference source for GRANTA database administrators who manage the back-end organization of data. The content is compatible with and updated with the latest versions of all Granta products referred to in the document.

1.1 What is covered in this document?

The focus of this document is on the GRANTA MI data schema, and how to ensure that your company's materials data can be captured, maintained, analyzed, and deployed effectively. This document covers:

- Data schema objects—all of the objects defined and/or managed in the MI:Admin Schema tool, such as Tables, Attributes, Subsets, link groups, templates.
- Some system-level schema objects—databases, currencies, Profiles.

This document does not cover:

- Access control. System and database security group setup, and the different options for data access control (Attribute-based and Permission-based) are covered in the *GRANTA MI Access Control and Security Guide*.
- Home Page definition and configuration. See the *GRANTA MI:Viewer Home Page Author Guide*.

1.2 Using this document

You can read this document straight through, or simply jump to a specific topic directly. To introduce yourself to the key concepts in data schema design for GRANTA MI, we suggest you look at sections in this order:

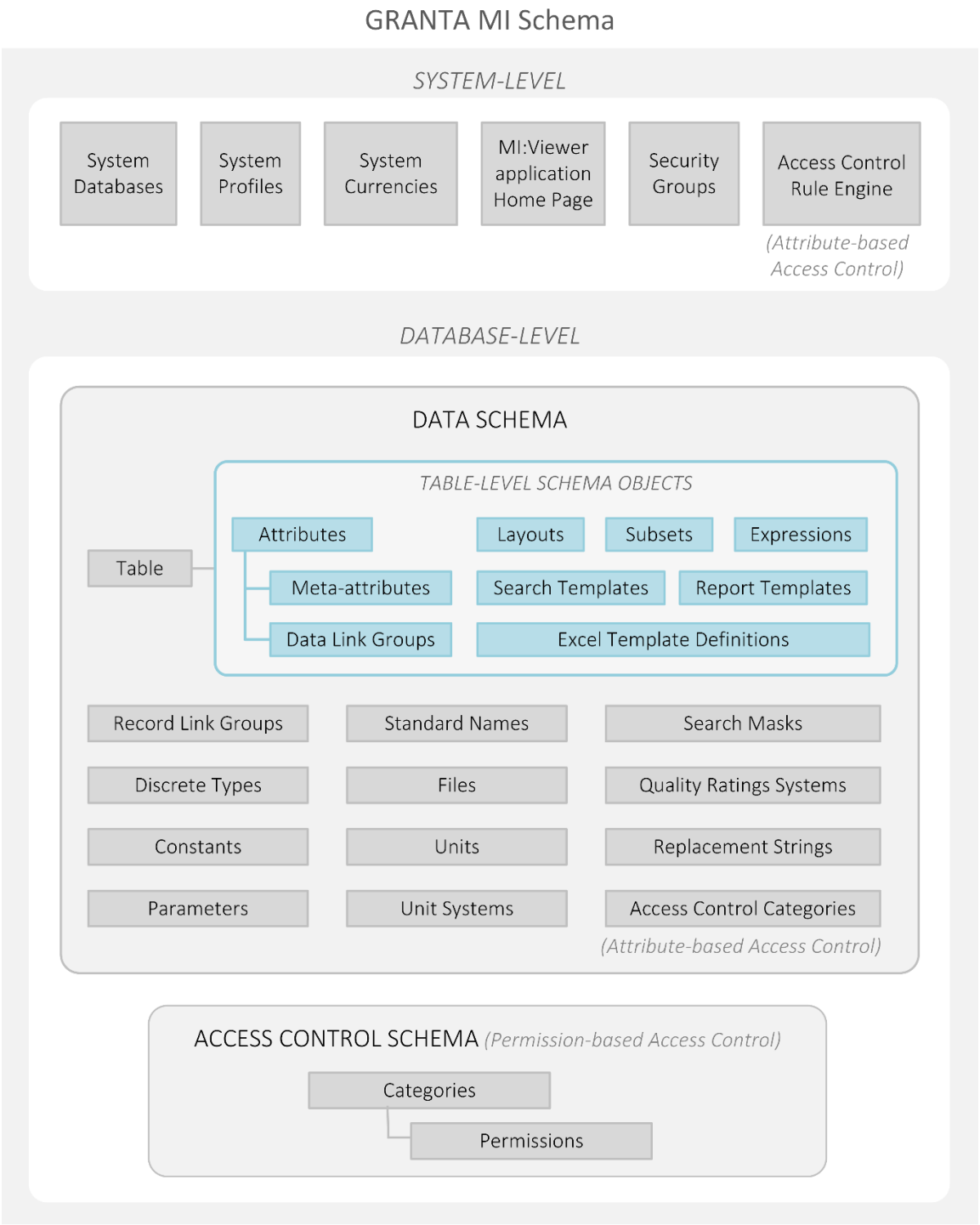
- Tables
- Attributes
- Record Link Groups
- Schema design best practice
- Layouts
- Subsets

1.3 Examples in this document

Most of the examples used in this document are based on the MI:Training database, a lightweight, tutorial database that contains a small selection of Granta data, suitable for use in training classes and for users who wish to familiarize themselves with GRANTA MI.

2 GRANTA MI Schema overview

The GRANTA MI schema defines the special data structures and objects that make up a GRANTA MI database, and the rules that govern them.



The following GRANTA MI tools can be used to view/modify the schema objects shown above:

Tool	Capabilities
MI:Admin	• View and modify system database properties • View and modify the data schema objects in each database. • Define System Profiles. • Define a Permission-based Access Control schema for a database. See the GRANTA MI Access Control and Security Guide.
MI:Server Manager	• Add databases to your GRANTA MI system and remove them; set/modify the database server connection properties • Manage System Currencies and update exchange rates • Define system security roles and database security roles. See the GRANTA MI Access Control and Security Guide. • Configure the Rule Engine for Attribute-based Access Control. See the GRANTA MI Access Control and Security Guide.

3 Schema design best practice

The database schema defines how your data is structured, in order to make it accessible, meaningful and useful. The GRANTA MI database schema:

- Organizes data into a hierarchy of Tables containing folders and records, and defines how data is linked
- Defines capabilities such as Units and Unit Systems, and Standard Names for FEA Export.
- Defines user experience (UX) elements for application users, including Subsets, datasheet Layouts, and templates for searching and reporting.

When designing the schema of a GRANTA MI database, the Database Admin needs to consider four objectives:

- Efficiency & maintainability—Reduce duplication and make easy to maintain.
- User experience—Does the schema represent the end-user's view of the data effectively? Can the data be understood by users with little training and interpretation?
- Capabilities—Can the database support your business processes?
- Traceability—Does the schema make it possible to track the trail of data generated during product development? People need to be able to answer questions like “How was this design allowable made?”, “Where did this alloy batch come from?”

Design decisions for GRANTA MI databases fall into 3 main categories:

- Defining the database Tables and the links between them
- Defining the data hierarchy—how records in the database are organized into folders and subfolders for easier access.
- Selecting the best data types for your data.

3.1 Tables and links

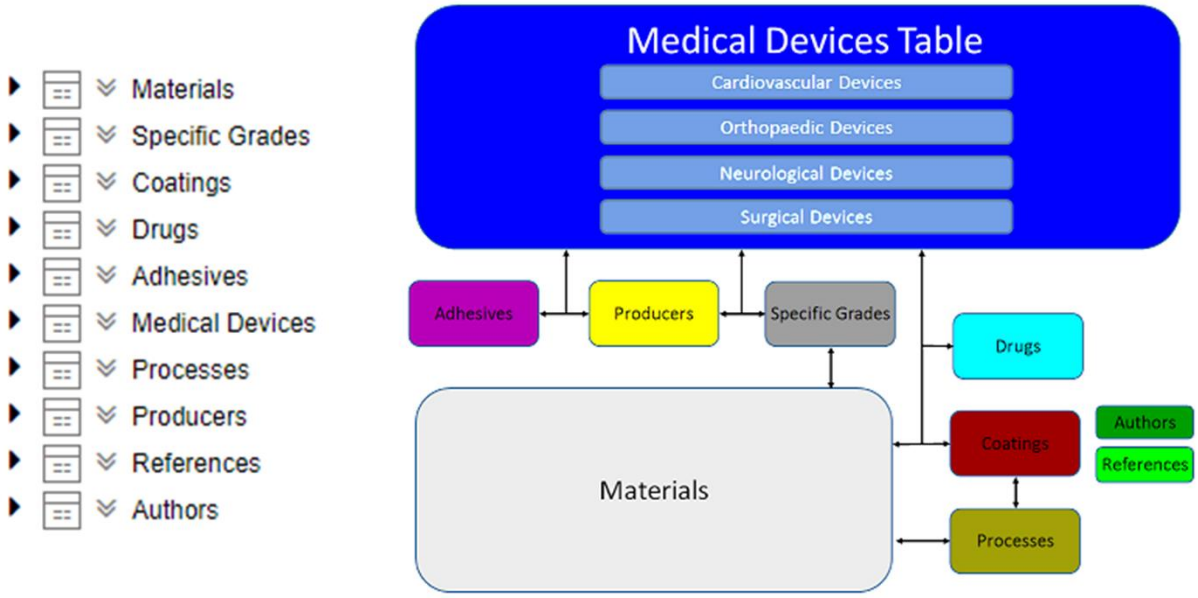
When making decisions about what Tables are needed in your GRANTA MI database, you need to consider:

- **What data** do I need to store? Legacy data could be in various forms, such as specifications, or declarations from suppliers.
- How should the data be **linked**?

3.1.1 Example: Materials in Medical Devices Reference

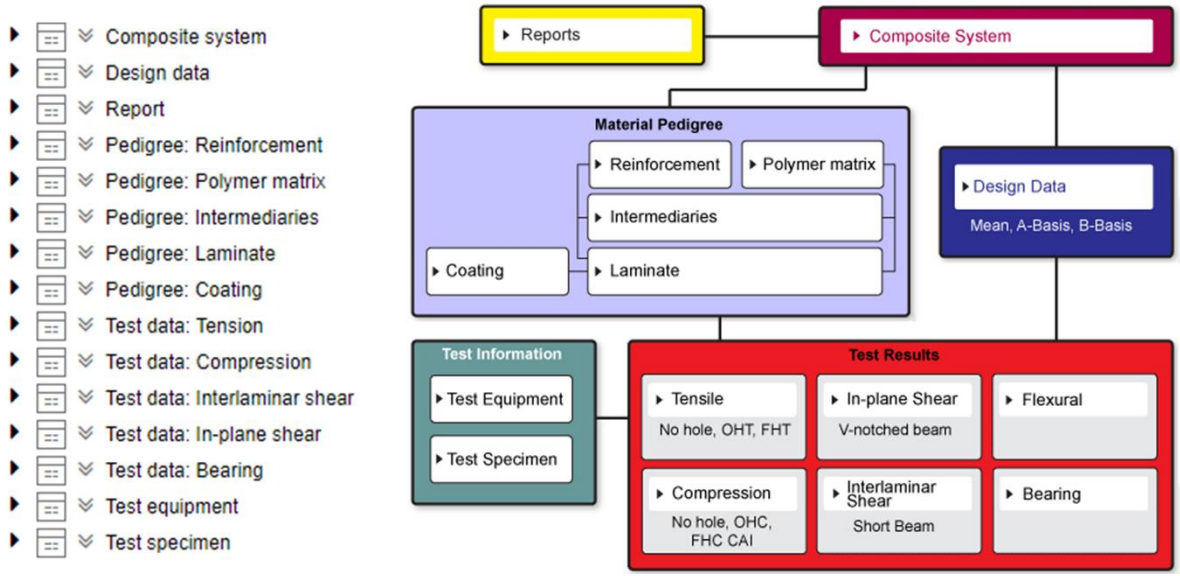
The ASM Medical Materials database is one of Granta's reference databases, used to screen, analyze, select, and source candidate materials and coatings for medical devices. In this database, medical **devices** use specific material **grades**, which are examples of a generic **material**. A device might also

use a **coating**, an **adhesive**, or be associated with/used to deliver a **drug**. Each device is made by a **producer**.



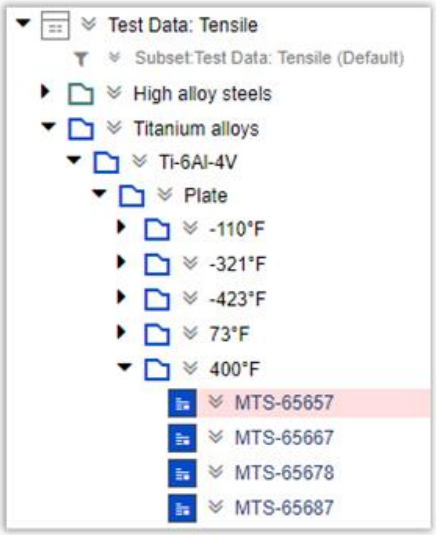
3.1.2 Example: Composites Template

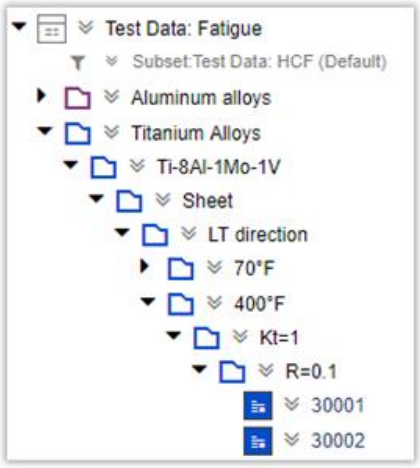
This Composites Template database contains design, materials pedigree, and supporting test data in 15 separate, linked Tables:



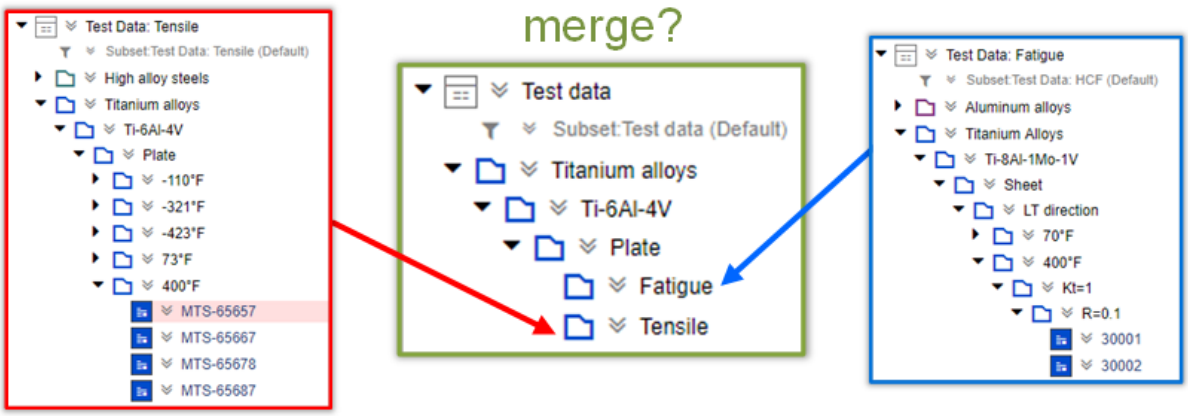
3.1.3 Making design decisions

Tables store "similar data". Consider this example where test data for tensile tests and test data for fatigue tests are stored in 2 separate tables:





Could this tensile and fatigue data could be merged into different records in the same table, or even into the same records in a single *Test data* table?



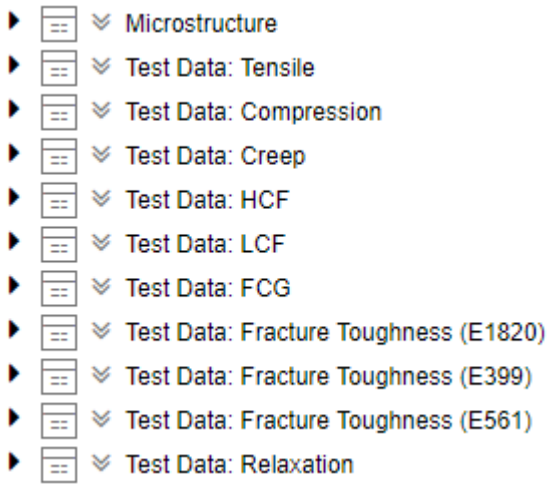
In order to make the right decision in this case, the MI Database Administrator needs to balance the objectives of efficiency and maintainability, display (user experience), capabilities, and traceability.

Reasons to keep the data in separate tables	Reasons to merge the data into a single table
User experience. Because key attributes are different in these two data sets, merging the tables would result in more Attributes, potentially leading to users having to scroll through long datasheets to find the data they	**Efficiency and maintenance.** Many Attributes are the same in these two Tables. Merging them will help to avoid duplication in the schema, making the database easier to maintain (e.g. when creating new Attributes, managing customized importers…).

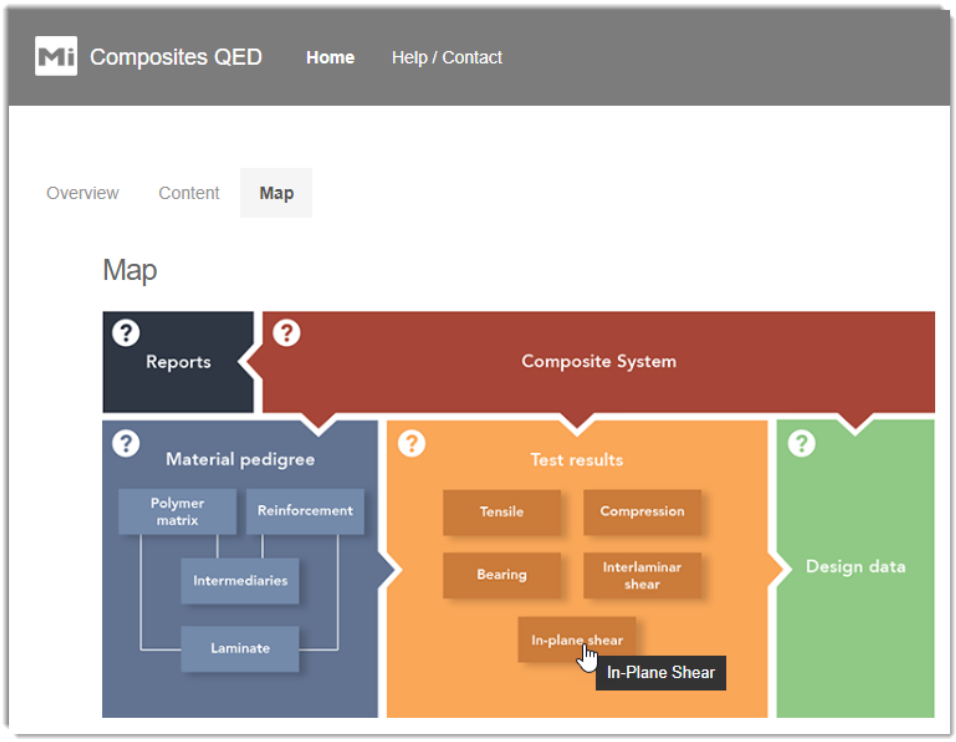
Reasons to keep the data in separate tables	Reasons to merge the data into a single table
are interested in, and also making it harder to select Attributes for searching a reporting.	
Traceability. Separation of the two different types of test data ensures clearer traceability paths.	**Capabilities.** Comparing and combining data is easier when the data is in the same table, but in this case, you are never likely to need to compare or combine tensile and fatigue data).
User experience. Having the data organized into two, parallel data 'trees' makes it easier for users visualize the "shape" of the data hierarchy, and easily spot where folders or records are 'missing' from one tree.	**Capabilities.** To perform an Attribute search across the test data, all of the data must be in the same table.

Large databases with many tables present a usability challenge, potentially overwhelming database users. However, there are some steps you can take to mitigate this in GRANTA MI:

- Use consistent and logical table naming conventions to help to clarify what type of data is in each table, and how one table relates to another. For example, in this database, related tables have a similar naming format:

- Provide a useful database Home page with information to help users understand how the database is organized, such as a schema diagram. For example:

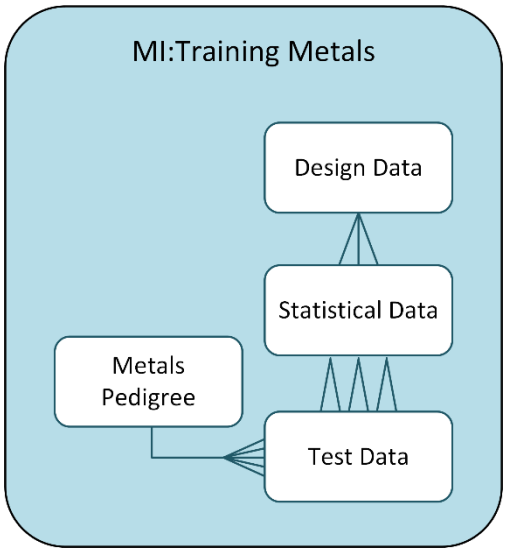


- Show only the data that's relevant to specific groups of users:
  - Use Profiles to provide a selective view of your data. Profiles can include Tables from one or more databases, for example, an *Aerospace* profile could include specific Tables from your alloys, composite test data, coatings, and creep/fatigue data databases.
  - Subsets and Layouts can be used to filter the Tables, folders, records, and data available within a database to MI:Viewer and MI:Explore users, so they only see the data that's relevant to them.
  - The MI:Explore application provides simple, single-table search/edit capability, with users selecting a "data view" to access specific data; different data views may be defined to provide access to other data in the same or in different tables.

Links between tables

Links typically represent the "flow" of data in a database and provide traceability. For example, in the database below, supplier information in the Metals Pedigree Table below is linked to data from tests

performed on the specimen, and this data is then rolled up with analysis in the Statistical Data Table and into the Design Data Table:



There are five different ways of providing between records in different tables in GRANTA MI:

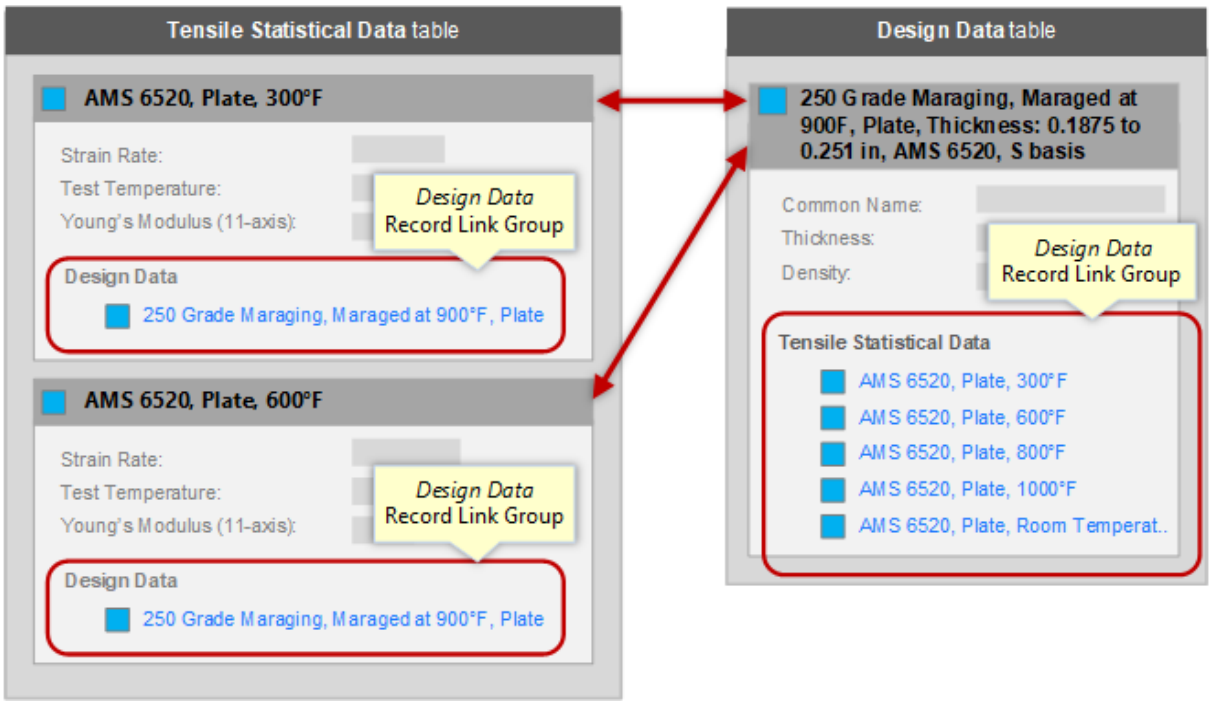
- Static record links
- Smart record links
- Data links
- Tabular data links
- Associated records

Static record links

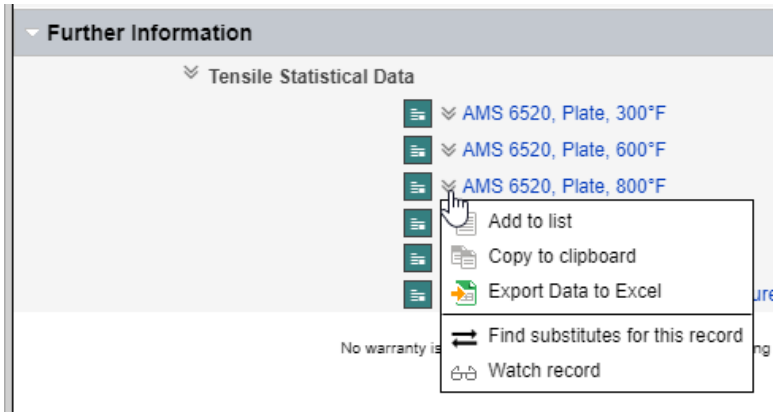
These are simple record-to-record links implemented via Record Link Groups.

Advantages	Disadvantages	Use when:
• Simple to set up - no rules required • Users can search on Attributes in linked tables	• Links are added manually • Links need active maintenance if record values change • Record-to-record only • Links are not "data, and so are not included when exporting data from records	• Linking records where rules are complex, or not well defined • Linking is static • Where records are imported, and auto-linking can be used

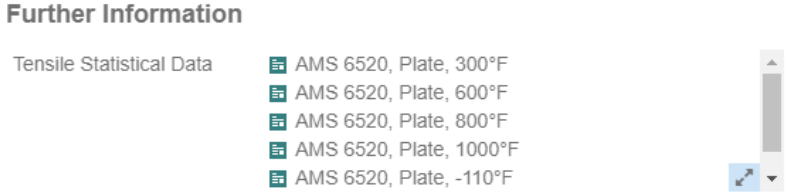
Example: the static links in this *Design Data* Record Link Group provide fixed links between test records and corresponding design data records:



In MI:Viewer, static record links are listed under a heading as shown here, and users can click on the link to open the datasheet for the linked record, as well as add linked records to the Record List, copy them to the clipboard, and export them, as required:



In MI:Explore, linked records are simply listed on the datasheet:



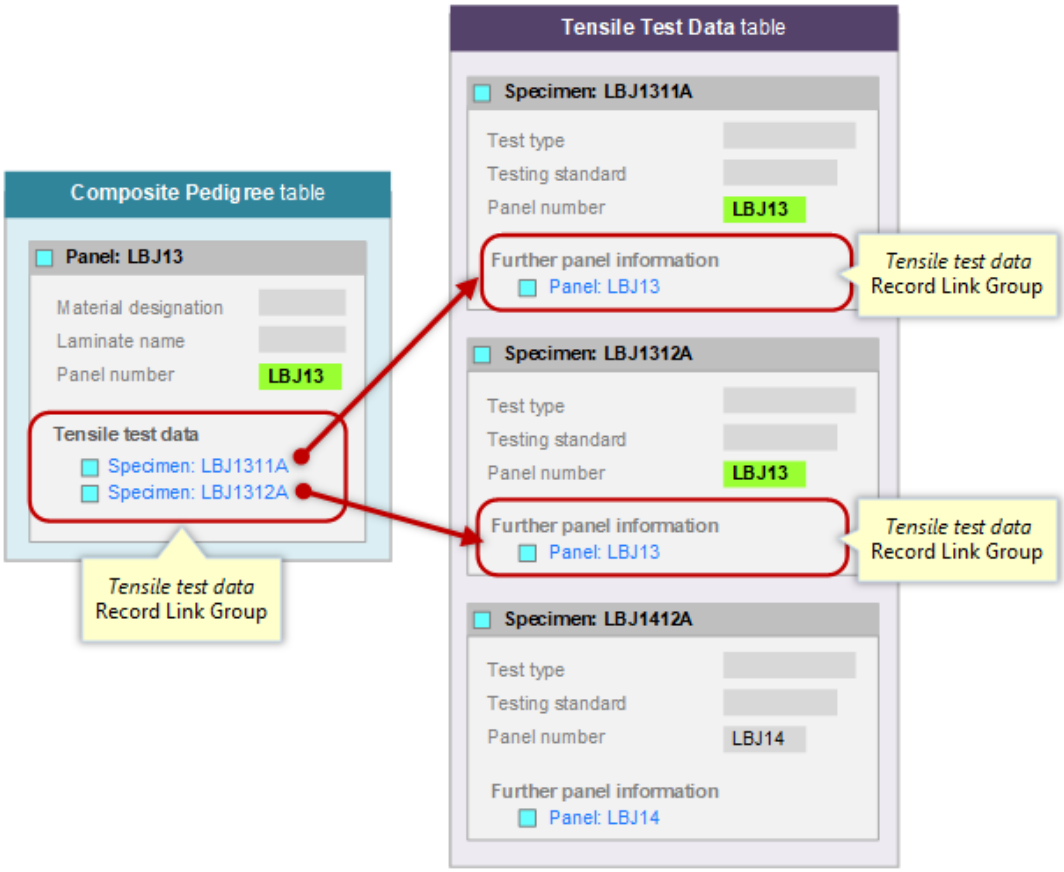
See also Static links for more detailed information about how Static Record Link Groups are defined in GRANTA MI.

Smart record links

Smart links can be used to provide dynamic links to records in other Tables based on the value of key Attributes in the source and target records: when the values match, the Records are linked. Smart links look the same as Static links on datasheets, but are set up in a different way, with Smart link criteria specifying the conditions that must be met for the link to be made.

Advantages	Disadvantages	Use for:
• Never need populating or updating • Allow searching on Attributes in linked tables • Up to 3 linking Attributes can be specified	• Only simple linking rules (A=B AND C=D) • Record-to-record only	• Linking records where rules are well defined • When records are created manually (in MI:Viewer or MI:Explore) • When linking values may change often

Example: this Record Link Group provides smart links between a panel record and the relevant tensile test data records by matching on the panel number in the two records:



In both MI:Viewer and MI:Explore, smart record links look and behave exactly the same as static links on datasheets.

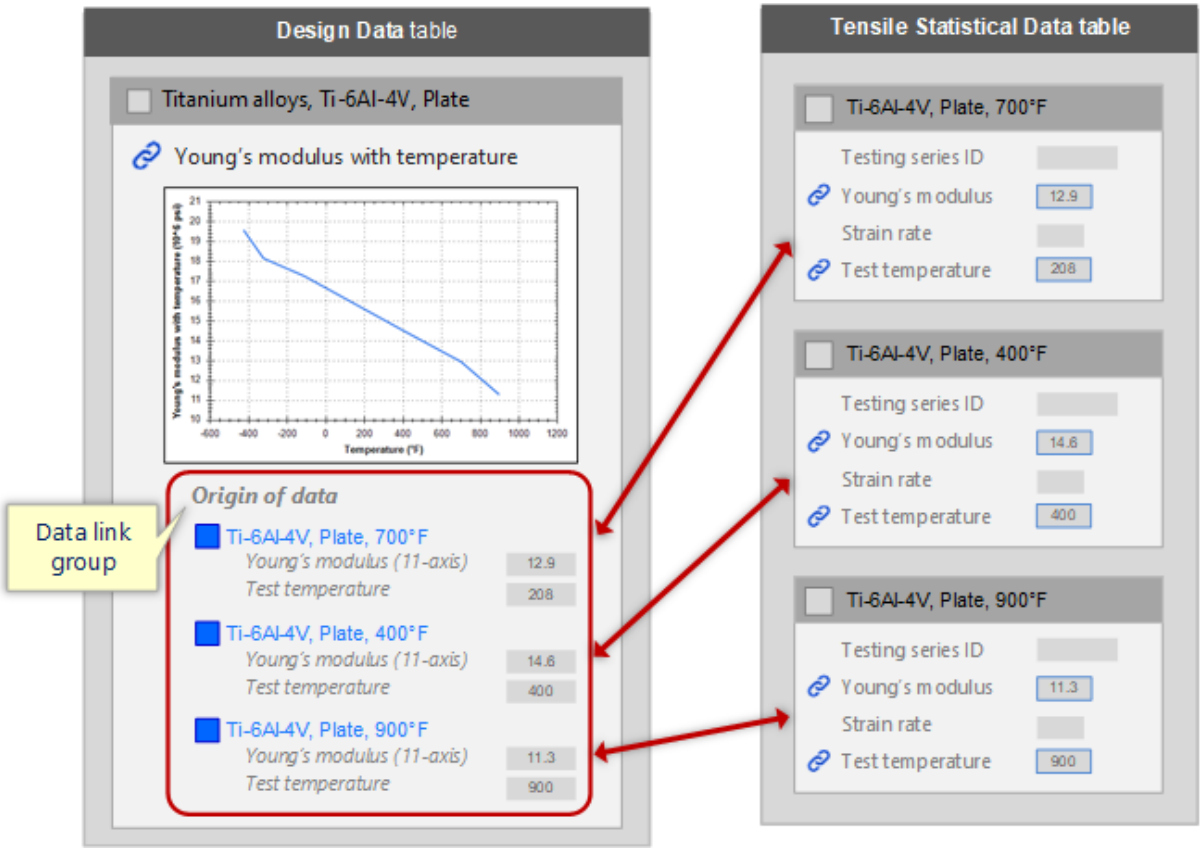
See also Smart links for more detailed information about how Smart Record Link Groups are defined in GRANTA MI.

Data links

Data links provide links from an Attribute to another Record or to specific data in another Record. For example, a mean value could be linked to the source values used to generate it, or statistical data could be linked to the relevant test data.

Advantages	Disadvantages	Use for
• Data-to-data or data-to-record • Users can see the linked data	• Static only • Linked values are not searchable • No access via Scripting Toolkits	• Data value sourcing; reference citations (populate on import)

Example: the Data Link Group in the design record shown below provides links from the *Young's Modulus with Temperature* functional data in the *Design Data* table to the source data in the *Tensile Statistical Data* table.



See also Data links for more detailed information about how Data Links are defined in GRANTA MI.

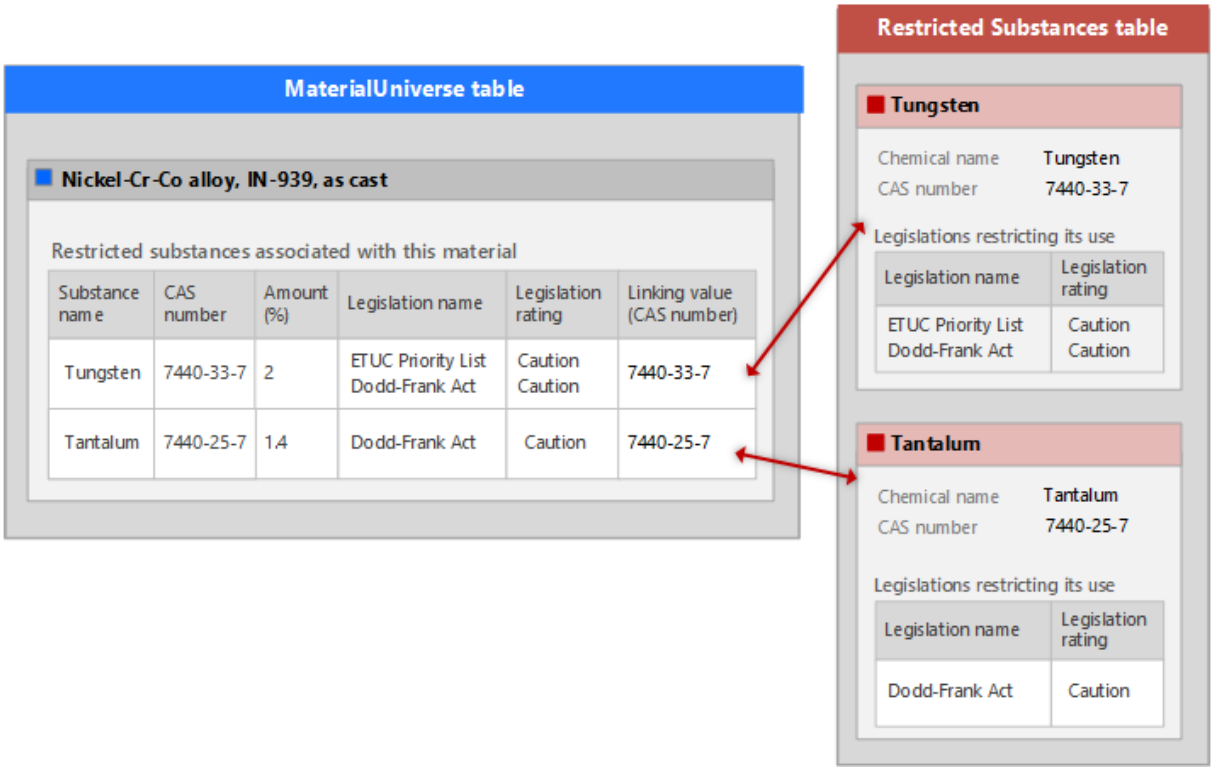
In MI:Viewer, Data links appear on the datasheets under the Attribute data; in MI:Explore datasheets, Data links are not displayed.

Tabular Attributes

Tabular Attributes are a complex data type that can combine both local data and data from other records via Tabular data links.

Advantages	Disadvantages	Use for
• Logical, clear display of linked data • Can blend linked and local data • Users can search on linked (and local) values	• Very simple linking rules (1 attribute) • Data storage is expensive (every row is a hidden record) • Links are unidirectional (unless you use Associated Records) • Can't add linked records to record list in this form (see Associated Records)	• Test summaries, substance queries, complex material pedigrees…

Example: this Tabular Attribute in a material record summarizes data about substances associated with it, and it includes linked data from the relevant records in the *Restricted Substances* table:



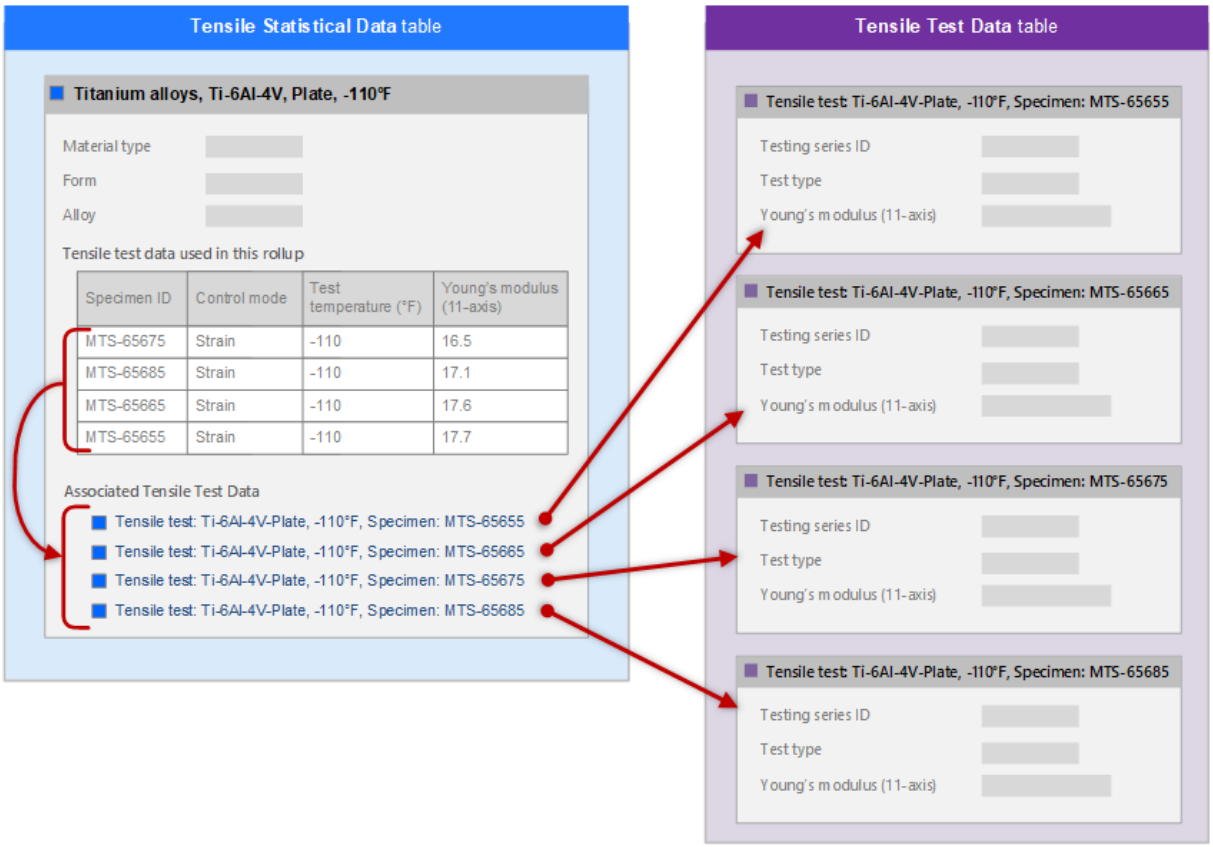
See also Tabular Attributes for information about how Tabular Attributes are defined in GRANTA MI, and for more examples.

Associated Records (via Tabular Attributes)

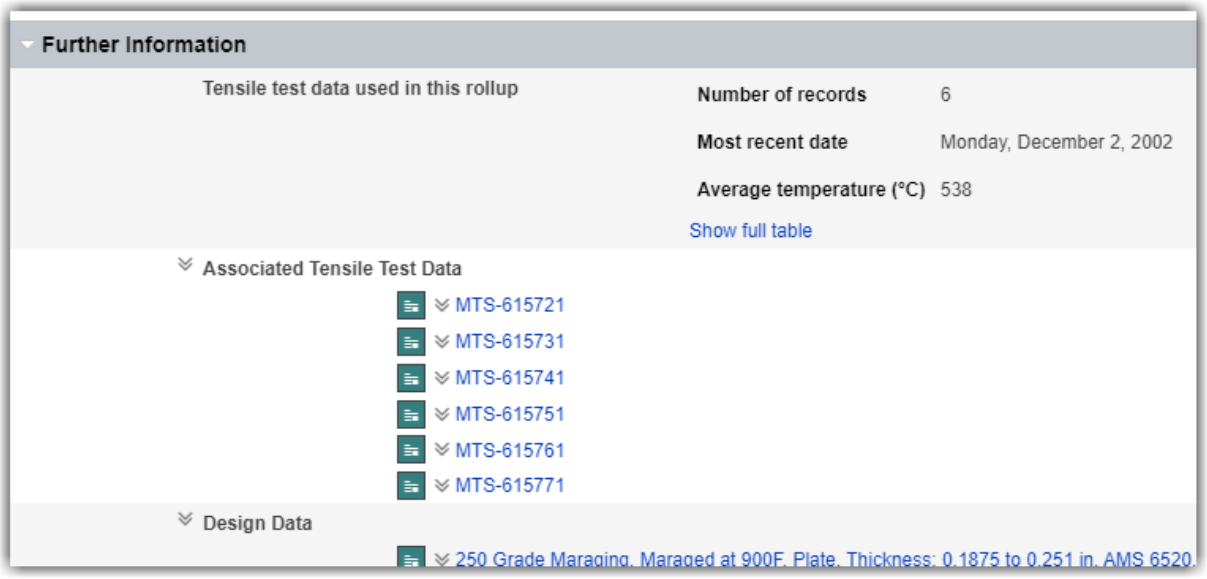
For records that include Tabular data, another type of link can be provided on datasheets in MI:Viewer—**Associated Records**. Associated Records are a Layout feature that allows a record datasheet to automatically list all of the records that are linked to via a specific Tabular attribute in the record.

Advantages	Disadvantages	Use for
• Can add to record list • Reverse links • The linked records are visible on datasheets at-a-glance, without having to open/view the Tabular data from which the links are derived. • Multiple hops in 1 click e.g. design data -> (statistical data) -> raw test data. If you have tabular links A->B and B->C, you don't need to make an extra link A->C. You can build up complex linkage with relatively few "link groups" (tabular attributes).	• All the disadvantages of Tabular data (data heavy, simple linking rules) • Conceptually complex • Depend on (multiple) record IDs being properly filled in • **MI:Viewer only;** Associated Record links are not shown in datasheets in MI:Explore	• Test summaries, substance queries, complex material pedigrees… • Alternative to smart links

Example: the list of associated test records in this tensile statistical datasheet is derived from the data in the *Tensile data used in this rollup* Tabular Attribute:



In the datasheet below, the associated records are visible when the Tabular data from which they are derived are not shown by default, for example, because the Tabular Attribute is configured to initially hide the full data:



Associated Records can also be bidirectional: in this case, the Tensile Test records can link back to the statistical rollup record, via another Associated Records item on the layout.

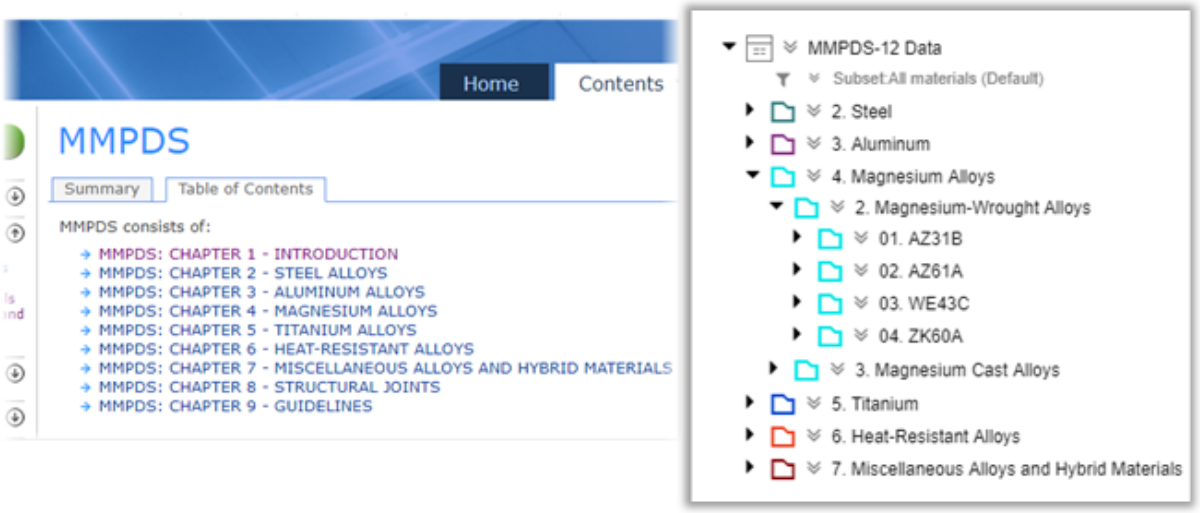
Associated Records can even follow a chain of more than one tabular link: if Design Data is linked by a tabular attribute to Statistical Data, and Statistical Data is linked to Test Data, you can make an Associated Records item from Design Data to Test Data (and from Test Data to Design Data). This provides comprehensive linking without maintaining several link groups.

Links summary

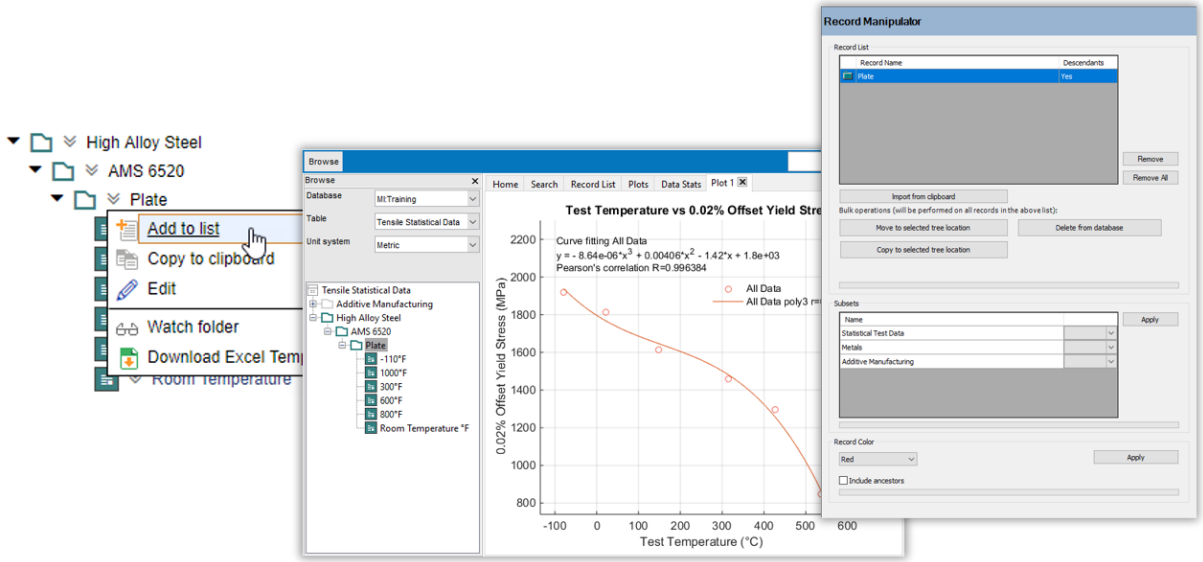
		Static Record Links	Smart Record Links	Data Links	Tabular Attribute	Associated Records
Link Nature	Can force one-to-one links?	No	Yes (referential integrity)	No	No	No
	Auto-create links on import?	Yes (by rules or GUID)	Not necessary	Yes (by rules or GUID)	Yes, but may need multiple imports	Yes, but may need multiple imports
	Granular (link individual data points)?	No	No	Yes	Linked attributes can be viewed	No
	Flexible (not rules-based)?	Yes	No	Yes	Yes	Yes
	Pulls linked data onto datasheet?	No	No	Yes	Yes	No
	Blend linked and local data?	No	No	Yes, on datasheet	Yes	No
	Multiple hops in one click?	No	No	No	No	Yes
	Display notes on links?	Yes	No	Yes	Yes (local data column)	No
Admin	Fast setup for the admin?	Fastest	Medium (must define rules)	Fast	Not fast	Not fast
	Link maintenance burden?	Heavy (manual editing)	Light	Heavy	Light	Light
Database Performance	Impacts database size?	Yes	Smallest	Yes (big impact)	Yes	No more than tabular attribute alone
	Impacts datasheet loading time?	No	Yes	No	Yes (big impact)	Yes
Search & Export	Add to record list from datasheet?	Yes	Yes	No	Not in this form	Yes
	Search on attributes in linked tables?	Yes	Yes	No	Yes	No
	Export links to Excel with the record?	No	No	No	Yes	No
	Export links to text/FEA with the record?	No	No	No	Yes	No
	Scripting Toolkit access?	Yes	No	No	Yes	No

3.2 Data hierarchy

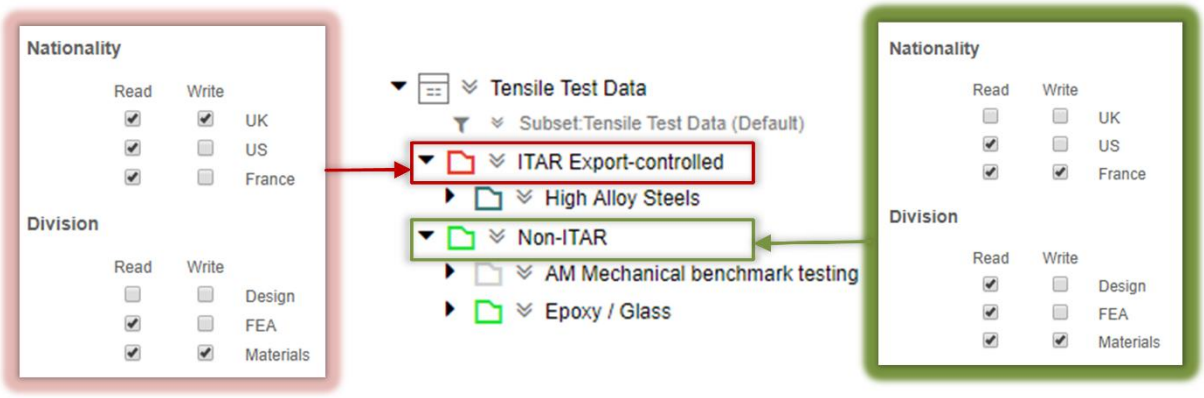
The way that data is organized into records and folders in your database can be used to model the way that the data is categorized, such as by internal or external industry standards. In the *MMPDS-12* Granta reference database, for example, the organization of alloys data follows the structure of the MMPDS handbook:



Folders may also be used to group together Records that often need to be accessed at the same time, for example, for reporting, analysis, or manipulation:

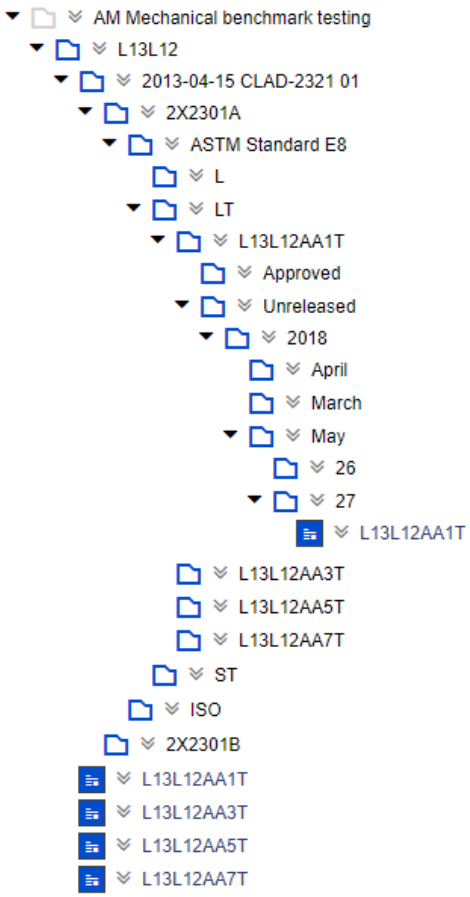


Another way in which Folders can also be used is purely for (permission-based) access control, for example:

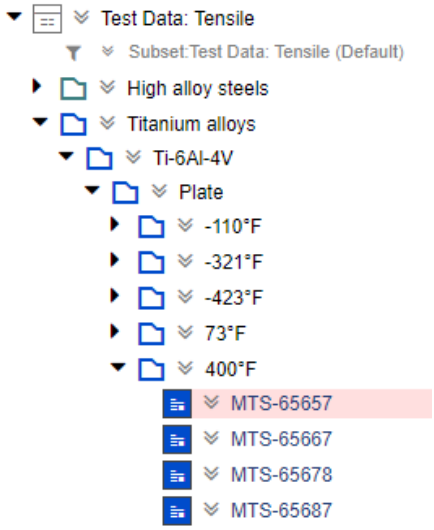


Decisions about whether a deep or shallow data hierarchy is preferable will have a significant impact on usability for application users who need to browse the data. For example:

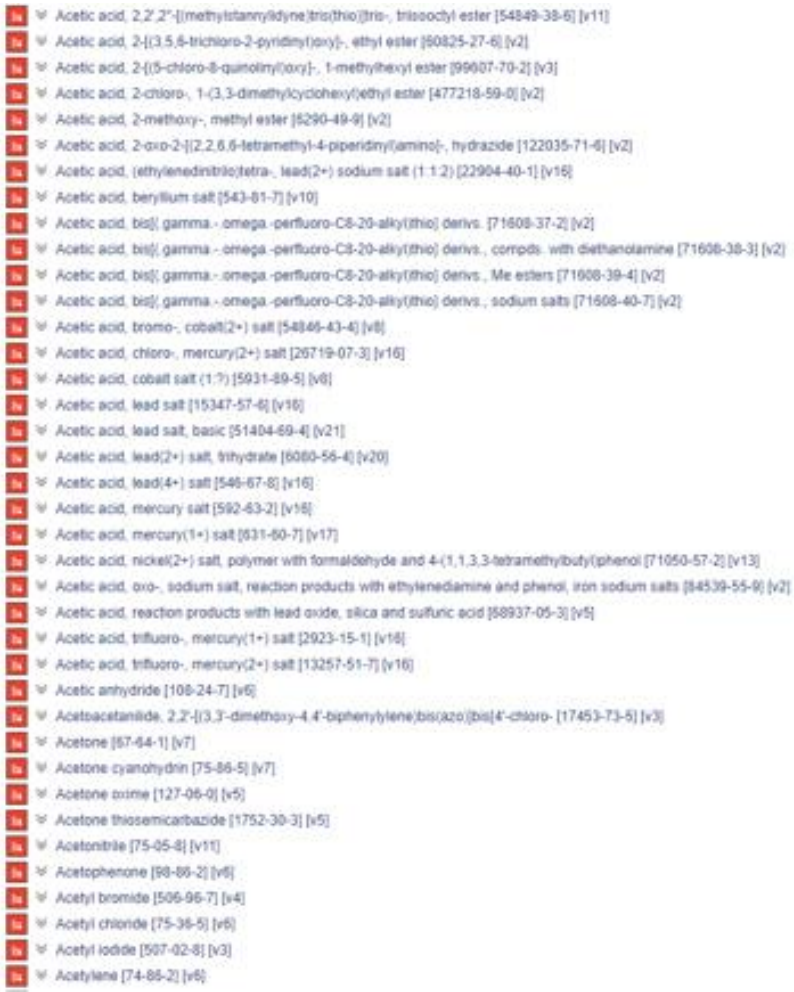
Deep structure — slow browsing:



Moderately structured — optimized for faster browsing (5 - 20 items per folder):



Shallow structure — slow browsing:



In MI:Explore, users work with lists of records that are filtered by data values, and never see the data "tree", so the folder hierarchy is not relevant.

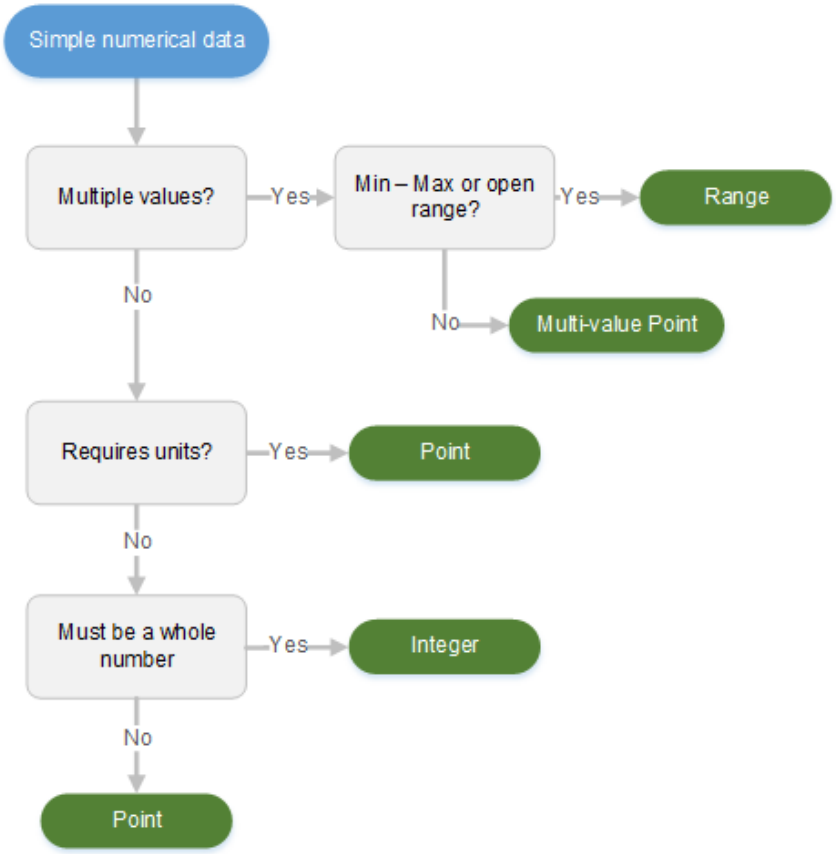
3.3 Choosing the right data type

GRANTA MI supports 15 data types, and choosing the correct type to store your data is crucial.

3.3.1 Simple numerical data

Simple numerical data in GRANTA MI may be one of 3 data types:

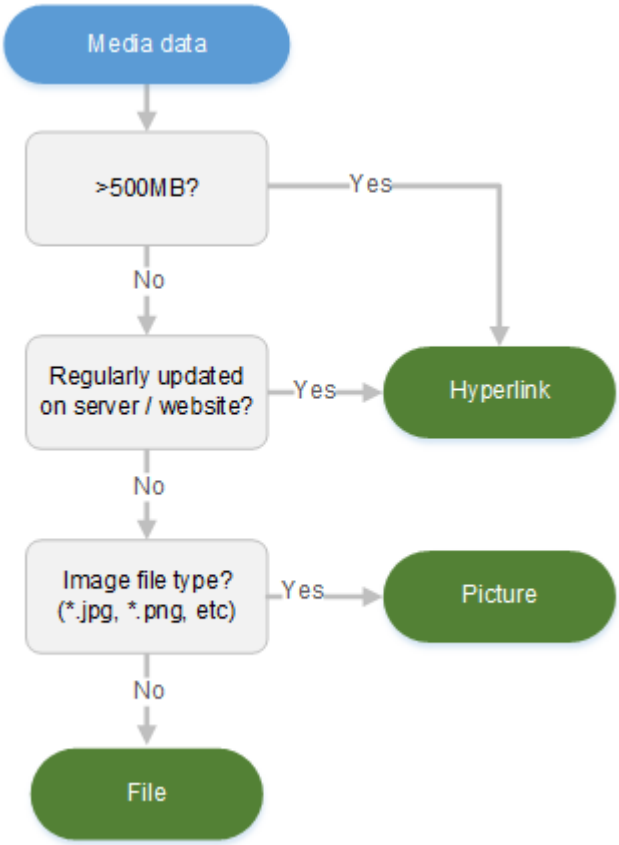
Data Type	Symbol	Code	Use to store
Integer		INT	Numbers without a decimal point
Point		PNT	Floating point numbers
Range		RNG	A pair of numeric values representing the upper and lower bounds of the value range



3.3.2 Media data

Media data in GRANTA MI may be one of 3 data types:

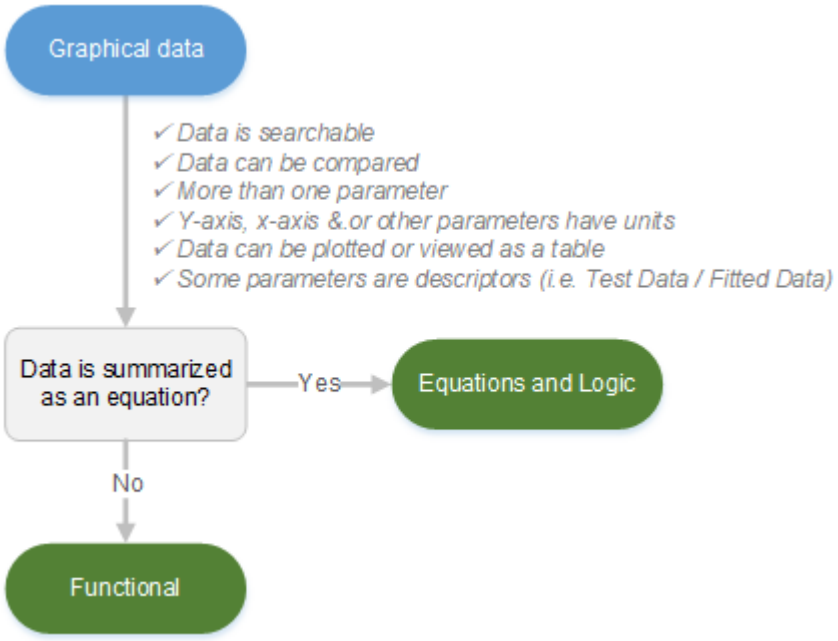
Data Type	Symbol	Code	Use to store
File		FIL	Entire data files e.g. text files, Office documents, PDF files, video files.
Picture		PIC	Image files such as photos or schematics
Hyperlink		HYP	Hyperlinks (URLs, email links)



3.3.3 Graphical data

Graphical data in GRANTA MI may be one of 2 data types:

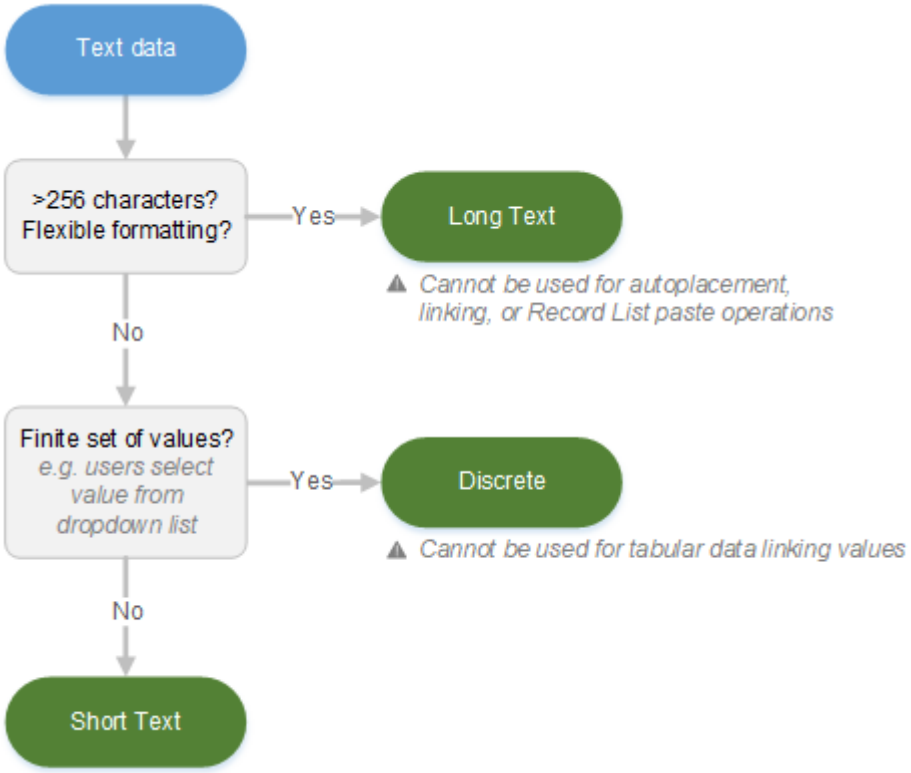
Data Type	Symbol	Code	Use to store
Float Functional	F	FDA	Numerical data that is a function of one or more Parameters.
Equations and Logic	Fn	MAFN	Numerical data calculated from other Attribute values using an equation.



3.3.4 Text data

Text data in GRANTA MI may be one of these 3 data types:

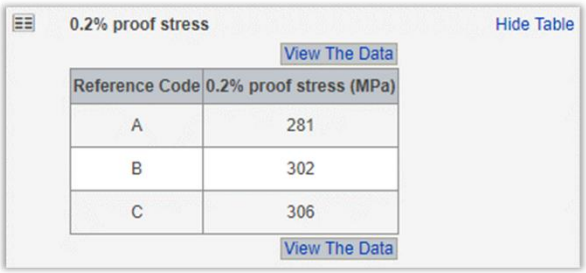
Data Type	Symbol	Code	Use to store
Short text	t	STXT	Text up to 255 characters (including spaces and punctuation).
Long Text	T	LTXT	Text up to 1,048,576 characters.
Discrete	☰	DCT	A discrete set of non-numerical data values.



3.3.5 Complex data

Complex data that needs to be displayed in rows and/or columns, can be stored in a number of different data types. For example:

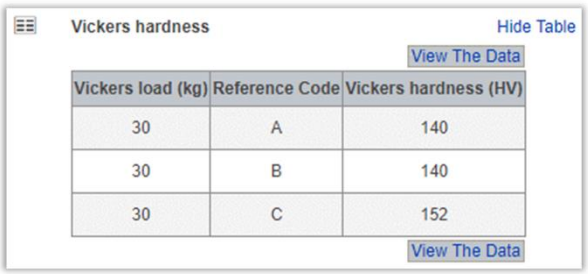
Functional data



0.2% proof stress — Hide Table — View The Data

Reference Code	0.2% proof stress (MPa)
A	281
B	302
C	306

View The Data

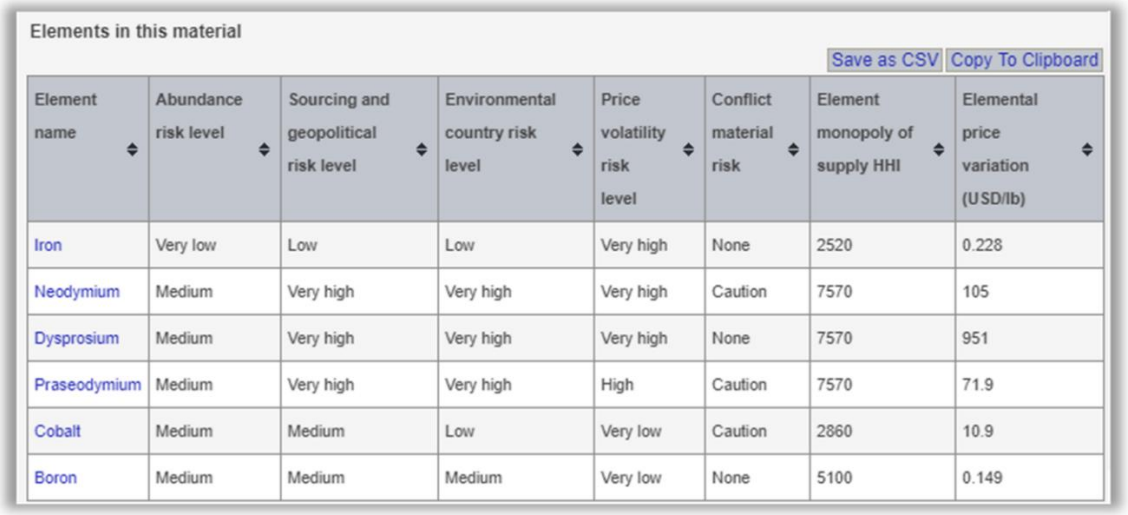


Vickers hardness — Hide Table — View The Data

Vickers load (kg)	Reference Code	Vickers hardness (HV)
30	A	140
30	B	140
30	C	152

View The Data

Tabular data



Elements in this material

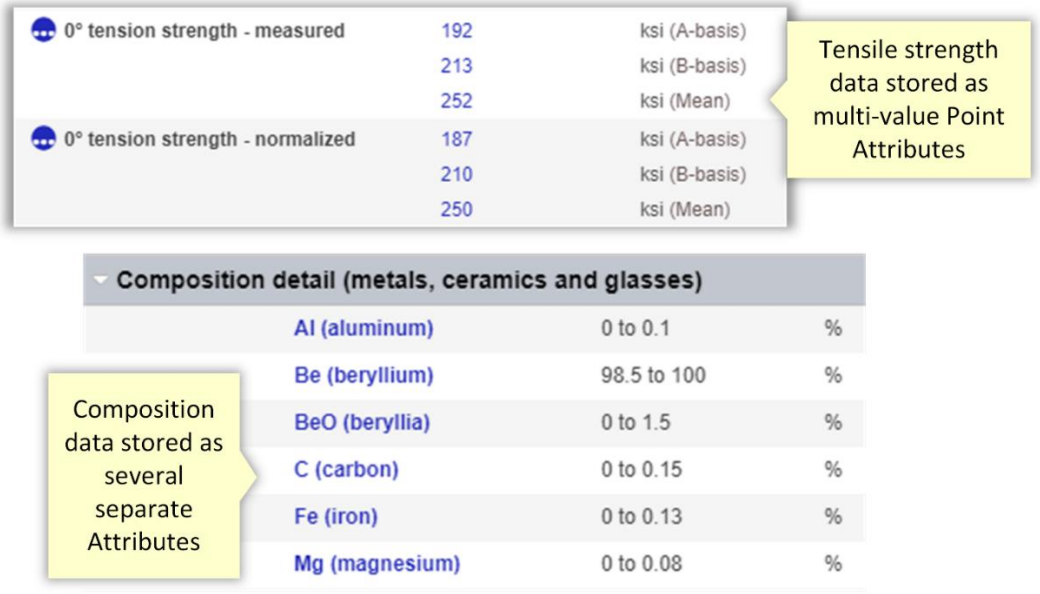
Save as CSV | Copy To Clipboard

Element name	Abundance risk level	Sourcing and geopolitical risk level	Environmental country risk level	Price volatility risk level	Conflict material risk	Element monopoly of supply HHI	Elemental price variation (USD/lb)
Iron	Very low	Low	Low	Very high	None	2520	0.228
Neodymium	Medium	Very high	Very high	Very high	Caution	7570	105
Dysprosium	Medium	Very high	Very high	Very high	None	7570	951
Praseodymium	Medium	Very high	Very high	High	Caution	7570	71.9
Cobalt	Medium	Medium	Low	Very low	Caution	2860	10.9
Boron	Medium	Medium	Medium	Very low	None	5100	0.149

The decision about whether to use a Functional or Tabular Attribute needs to consider:

- Linking: can the data content be stored in another record to reduce data duplication & maintenance?
  - Tabular data can pull linked data from a master record in another table
- Searching: exactly what queries will users need to do on the data?
  - Tabular search queries are of the form: “Does a row exist where column x=[value]…”
  - Functional search queries are of the form: “What is the y-value at this x-value?”
- Import & Export processes: make data imports as easy and robust as possible
  - Functional data imports can be from the Text Importer or the Excel Importer, but must handle the Series/Grid distinction (see Functional Attributes section).
  - Tabular data is incompatible with the Text Importer.
- Performance & Storage
  - Tabular data can lead to comparatively larger databases than functional data.

For simple/small tables, a table of data within a datasheet may not be the best presentation. Consider multi-value Point Attribute, or several separate Attributes, for example:



4 Tables

Tables in GRANTA MI are used to organize and group the data in a database by common characteristics or properties, for example, a *Materials* table containing records that represent materials and their properties, a *Tensile Test Data* table with records that capture test data about those materials.

Each table defines:

- Attributes—enabling systematic characterization of the data. The Attributes in each table are typically specialized for the type of data and metainformation in that table, so a *Metals* table may include Attributes like *Density*, *Composition*, and *Young's Modulus*, while a *Test data* table could additional Attributes for *Test Type*, *Specimen ID*, and *Tensile Response*.
- Subsets and Layouts—used to present different data views to users, for example, by hiding tables, records, or data are not relevant.
- Search Templates, Report Templates, Excel Template Definitions—used for searching, reporting, and data import/export in MI:Viewer.

Tables may also include Expressions—mathematical expressions for use in the evaluation of Equations and Logic data.

Table properties provide additional options for controlling the visibility and searchability of data in the table, assigning a Quality Ratings System to a table, defining datasheet headers/footers, and enabling Record Version Control on the table.

4.1 Viewing and managing Tables

To view and manage the Tables in a GRANTA MI database: in the MI:Admin Schema tool, select the database, then click the **Tables** link. On the **Tables** page you can see a list of all the tables currently defined in the database.

To edit the properties of an existing table, select it in the list and click **Edit**; see Table properties. To change the order in which the tables are displayed (in MI:Admin and MI:Viewer), use the **Move Up**/**Move Down** and **Save Order** buttons.

4.1.1 Deleting a table

You cannot delete a table if:

- It is the source or destination table for a Record Link Group. To delete the table, you must first delete the Record Link Group.
- It is a destination table for a Tabular Attribute. To remove the table, you must first delete the Tabular Attribute.
- Version control has been enabled for that table. As a workaround, you can hide the table instead.

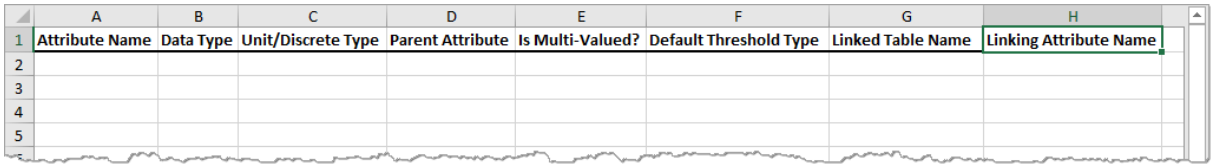
To view and manage the table Attributes, Subsets, Layouts, Expressions, and templates, expand the table name in the list of tables in the left pane, then click on the appropriate item.

4.1.2 Creating a Table

To create a new Table, copy an existing one and edit it, or click **Add**. You can either create a blank new table, with no Attributes, or use the Create new table wizard create the new table, add Attributes to it, and configure the default Layout.

4.1.3 Importing Attribute definitions

A quick way of adding multiple Attributes to a new table is to define them in Excel and copy/paste them into the Create new table wizard. The wizard creates an empty Excel document for you which is already formatted with the correct columns, for example:

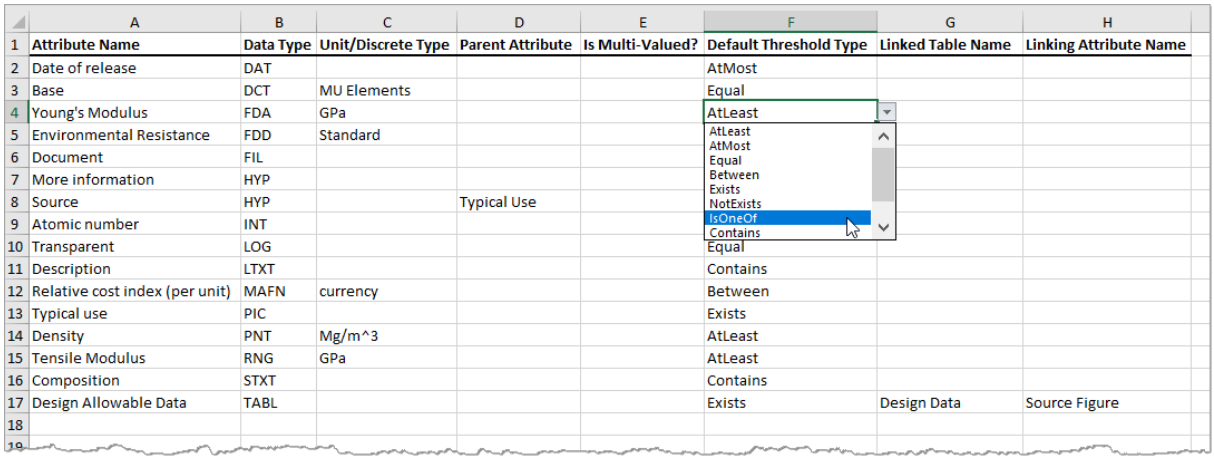


	A	B	C	D	E	F	G	H
1	Attribute Name	Data Type	Unit/Discrete Type	Parent Attribute	Is Multi-Valued?	Default Threshold Type	Linked Table Name	Linking Attribute Name
2								
3								
4								
5								

To define Attributes in this worksheet.

- Use the dropdown menus in the *Data Type* and *Default Threshold Type* columns to ensure the Attribute data type codes and threshold settings are correctly specified.
- Columns G and H are used for Tabular (TABL) Attributes only; either both or neither of these columns must be populated. The linking Attribute must be of type Short Text.

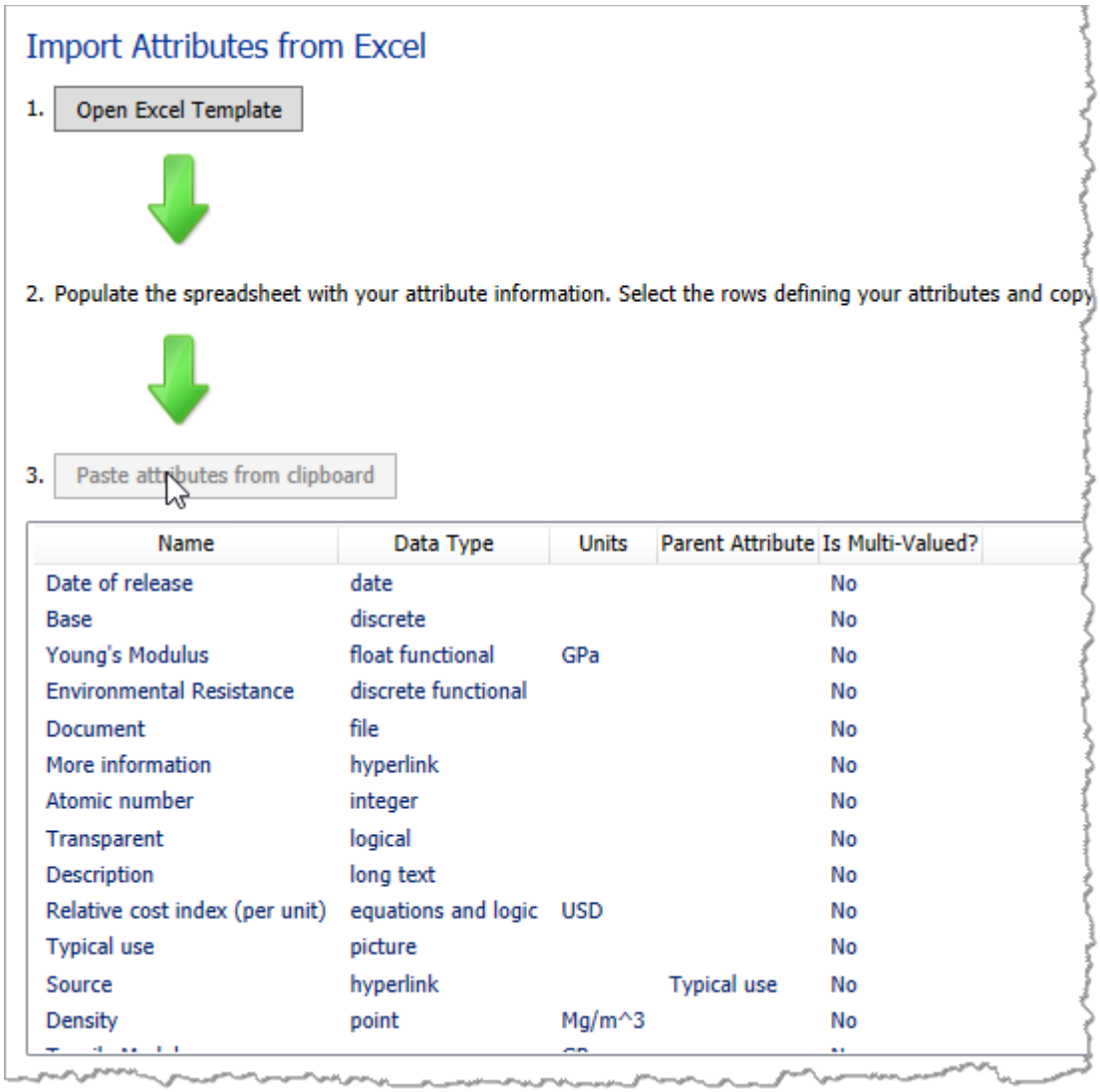
Example Excel Attribute definitions:



	A	B	C	D	E	F	G	H
1	Attribute Name	Data Type	Unit/Discrete Type	Parent Attribute	Is Multi-Valued?	Default Threshold Type	Linked Table Name	Linking Attribute Name
2	Date of release	DAT				AtMost		
3	Base	DCT	MU Elements			Equal		
4	Young's Modulus	FDA	GPa			AtLeast		
5	Environmental Resistance	FDD	Standard					
6	Document	FIL						
7	More information	HYP						
8	Source	HYP		Typical Use				
9	Atomic number	INT						
10	Transparent	LOG				Equal		
11	Description	LTXT				Contains		
12	Relative cost index (per unit)	MAFN	currency			Between		
13	Typical use	PIC				Exists		
14	Density	PNT	Mg/m^3			AtLeast		
15	Tensile Modulus	RNG	GPa			AtLeast		
16	Composition	STXT				Contains		
17	Design Allowable Data	TABL				Exists	Design Data	Source Figure
18								

AtLeast
AtMost
Equal
Between
Exists
NotExists
IsOneOf
Contains

When you have defined all of the required Attributes in the Excel worksheet, you select and copy them to the clipboard, and then paste them into the Create Table wizard:



Name	Data Type	Units	Parent Attribute	Is Multi-Valued?
Date of release	date			No
Base	discrete			No
Young's Modulus	float functional	GPa		No
Environmental Resistance	discrete functional			No
Document	file			No
More information	hyperlink			No
Atomic number	integer			No
Transparent	logical			No
Description	long text			No
Relative cost index (per unit)	equations and logic	USD		No
Typical use	picture			No
Source	hyperlink		Typical use	No
Density	point	Mg/m^3		No

4.2 Table properties

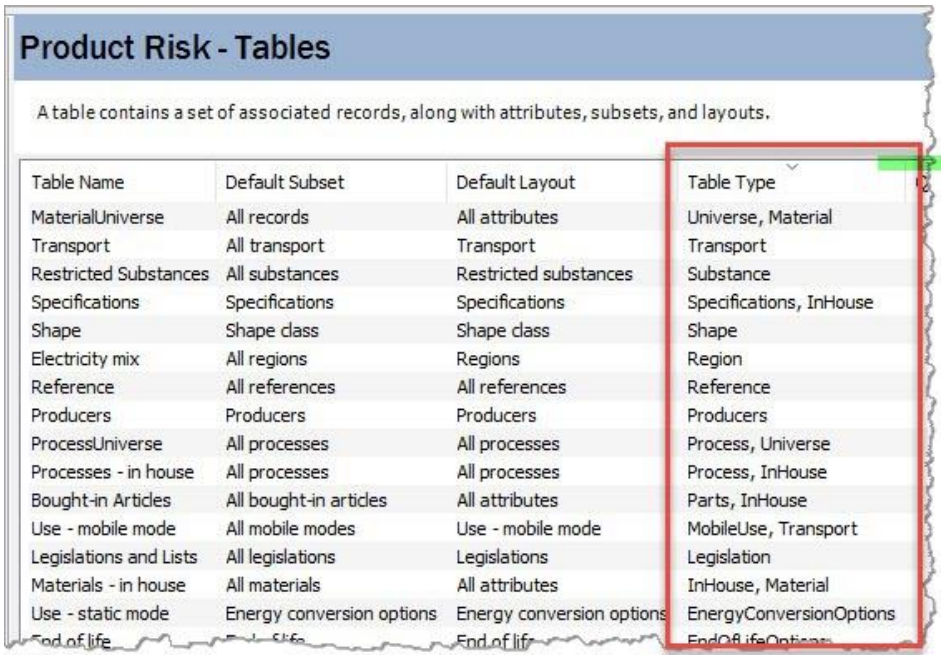
Tables in GRANTA MI have the following properties.

4.2.1 Name

This may be up to 255 characters long. The name entered here must be unique within the database; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same database (e.g. "Design data" and "Design Data").

4.2.2 Table type

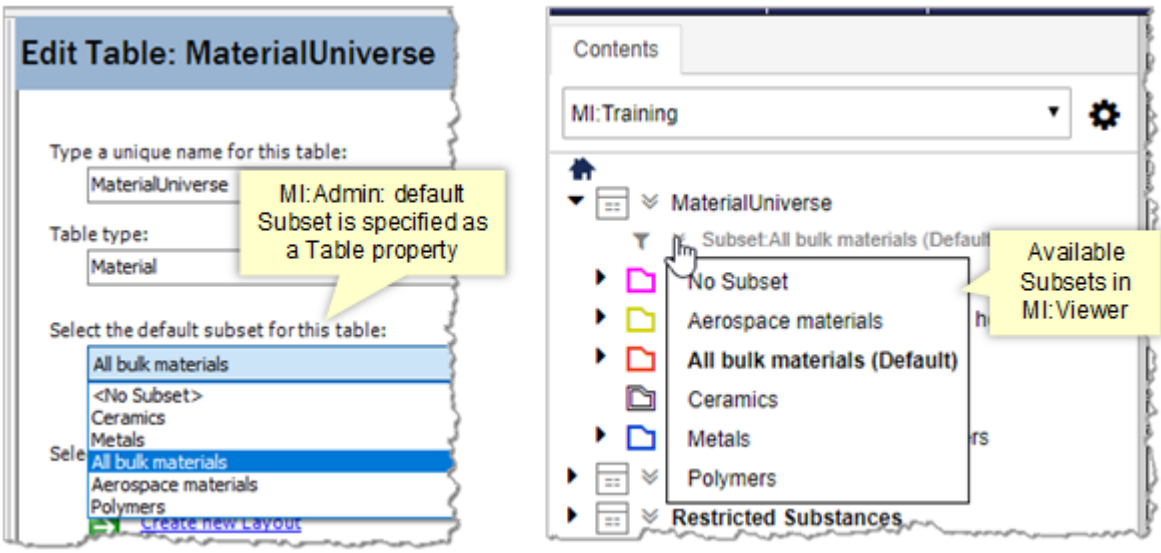
(*Optional*) This property is used for data assignment and reporting in MI:Materials Gateway. For example, Table type is set on the Product Risk database tables as shown here:



This may be up to 255 characters long.

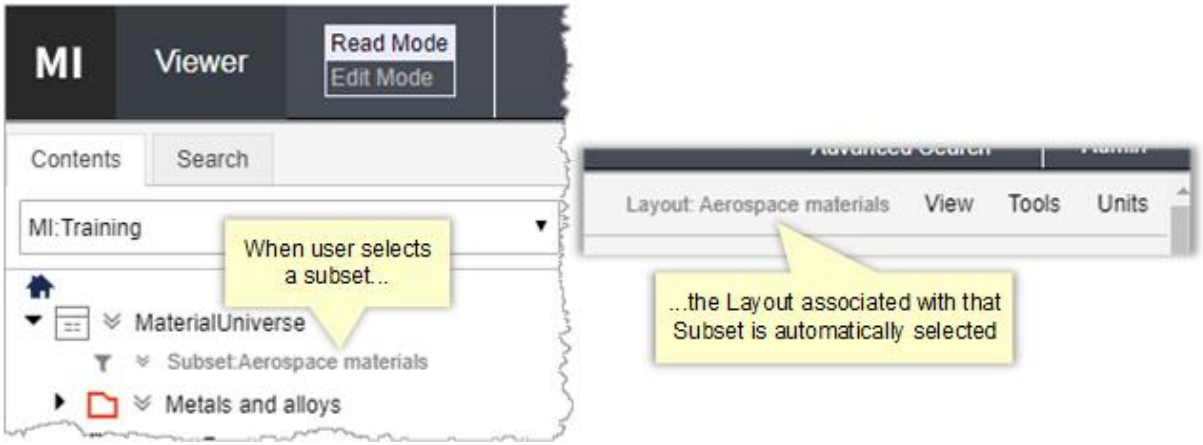
4.2.3 Default subset

The default Subset for a table determines which Subset will be selected by default in the MI:Viewer Contents pane. For example, the default Subset for the *MaterialUniverse* table in the MI:Training database is *All bulk materials*:



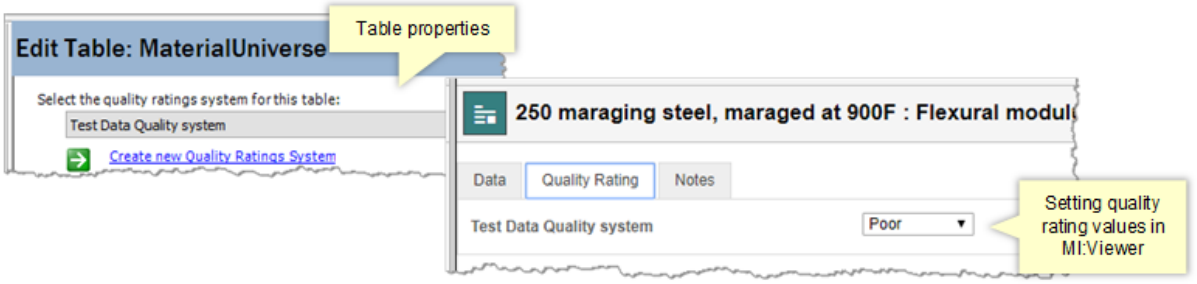
4.2.4 Default layout

The default Layout for the Table is used in MI:Viewer when 'No Subset' is selected in the Contents pane. (When a Subset is selected, the Layout associated with that Subset is used.)



4.2.5 Quality ratings system

Assign a Quality Ratings System to a Table to allow quality ratings to be applied to all data in the Table. For example, the *Test Data Quality System* is specified for the *MaterialUniverse* table in the MI:Training database, allowing quality ratings to be assigned to data on datasheets in MI:Viewer:



Changing or removing the Quality Ratings System will delete all ratings in the database which have been assigned to data values (data values themselves are unaffected).

4.2.6 Hide table completely, Browse table, Search table

These options control whether or not the table is available for browsing and searching in GRANTA MI applications.

Hide table completely

The **Hide table completely** option can be used to hide the table from application users. This option overrides the **Browse table** and **Search table** settings, which can be used to independently control the ability of users to browse and search a Table.

The effect of turning the **Hide table completely** option ON is as follows:

Application	Effect
MI:Viewer	The table does not appear in the Browse tree. No Attributes in the table are available for selection in the Search for attributes and data list or from the Attribute browser on the Advanced Search page. No records in the table will be returned in text searches because the text data in the table is not indexed.
MI:Explore	In MI:Explore, the effect of the Hide table completely option varies depending on the user's role. Read, Write, Grant, and Power Users: • Do not see any Attributes from hidden Tables in the list of filters in the Search pane (even if they are included in the search Layout). • Do not see search results from hidden Tables (because the text data in the table is not indexed). Admin users: the Hide table completely setting is ignored; Admin users can always see Attributes from hidden Tables in the Search pane and records from hidden Tables in search results.
MI:Toolbox	By default, the table will not appear in the Browse tree. This default can be configured by changing the Show Hidden Tables application setting in Toolbox.exe.config. Data can still be imported into the table.
Other applications (e.g. MI:Mat Analyzer, MI:Workflow)	Admin users can always see/access hidden Tables; Read, Write, Grant, and Power User cannot see or access hidden Tables.

When you change the value of the **Hide table completely** option, the table will automatically be re-indexed.

Hiding a table using this option may be useful in cases where the table is the destination table for some tabular data, but you only want to see that data in tabular format, not by itself. It could also be used to hide a table from users temporarily, for example, while it is being initially populated with data.

Browse table

The **Browse table** option controls whether or not the table appears in the Browse (Contents) tree in MI:Viewer; it has no effect on behavior in any other applications.

Search table

The **Search table** option controls:

- Whether or not text data in the table is indexed, and is therefore searchable.
- Whether or not Attributes in the table can be used to filter searches in MI:Viewer and MI:Explore.

When you change the value of this option, the table will automatically be re-indexed.

Search table = ON:

- The text data in the table is indexed, and records in the table will be returned in text searches in MI:Viewer and MI:Explore.
- In MI:Viewer, the Attributes in the table will be available for selection in the **Search for attributes and data** list and in the Attribute browser on the Advanced Search page.
- In MI:Explore, the Attributes in the table will appear in the Search filter panel (if they are included in the relevant Layout).

Search table = OFF:

- Text data in the table is not indexed, and therefore no records in the table will be returned in text searches in MI:Viewer or MI:Explore.
- The Attributes in the table will not be available for selection on the Advanced Search page in MI:Viewer or in the Search filter panel in MI:Explore. Note that this will override the **Attribute can be used as a search filter** setting on individual Attributes.

Summary of table Hide/Browse/Search options

	Table appears in Browse/Contents tree (Viewer and Toolbox)	Attributes in the table can be used as search criteria/filters (Viewer and Explore)	Records in the table are returned in text searches (Viewer and Explore)
Hide table completely = OFF	yes	yes	yes
Hide table completely = ON	no	no	no
Browse table = ON	yes		
Browse table = OFF	no		
Search table = ON		yes	yes
Search table = OFF		no	no

4.2.7 Datasheet headers and footers

Datasheet header and/or footer text defined as Table properties will appear in MI:Viewer on all datasheets for the records in the Table. For example:

```
<p style="color:gray; font-style:Italic""> &copy; MyCompany 2019. For
corrections please email <a href="mailto:data@mycompany.com?Subject=DATA
QUERY">data@mycompany.com</a></p>
```

Images can also be included in datasheet footers and headers by linking to them with an HTML `<img>` tag, allowing a company logo, for example, to be included on all datasheets. To reference an image file stored in the database, for example, one of the database home page resources, you can specify a relative pathname by including the database key in curly braces as follows (the database key is not case-sensitive):

```
<img src="homepagefiles/{MYCO_MATERIALS_DB}/MYCO_logo-footer.png">
```

This will display the image *MYCO_logo-footer.png* located in the *homepagefiles* folder in the MYCO_MATERIALS_DB database at the bottom of every datasheet.

To define the header and footer content, enter text in the **Set a header note...** and **Set a footnote...** boxes. You can enter plain text or HTML (HTML will not be validated).

4.3 Version Control—tracking data changes in a Table

In many businesses, such as aerospace and the medical device industry, the traceability of data is crucial to fulfil regulatory and compliance requirements. Record version control in GRANTA MI ensures that data used during the design of a product is automatically tracked, and can be accessed throughout the lifetime of that product even if that data has been superseded many times.

Version control is enabled on individual Tables in a database and applies to all of the records and data in that Table.

- Any changes to data in the Table, including metadata, and quality ratings, are tracked.
- Changes to record properties, including Full name, Short name, Short code, Color, Record type are tracked.
- Changes to access control settings are not tracked.
- Changes to record and data links are not tracked.
- Data schema changes (e.g. adding/deleting Attributes, Subsets, Layouts) are not tracked.

To ensure that full traceability is maintained:

- Version control for a Table cannot be turned off once it has been enabled.
- Version-controlled Tables cannot be deleted.

See the *GRANTA MI Record Version Control* guide for more information.

A ***version control signature*** may be displayed at the top of datasheets in MI:Viewer, for example, showing the date the record was released, and the user name of the person who released it. The

signature is also displayed when viewing the data for an individual Attribute for a released record. The signature is shown in Read and Edit mode, to all users who have privileges to view the record.

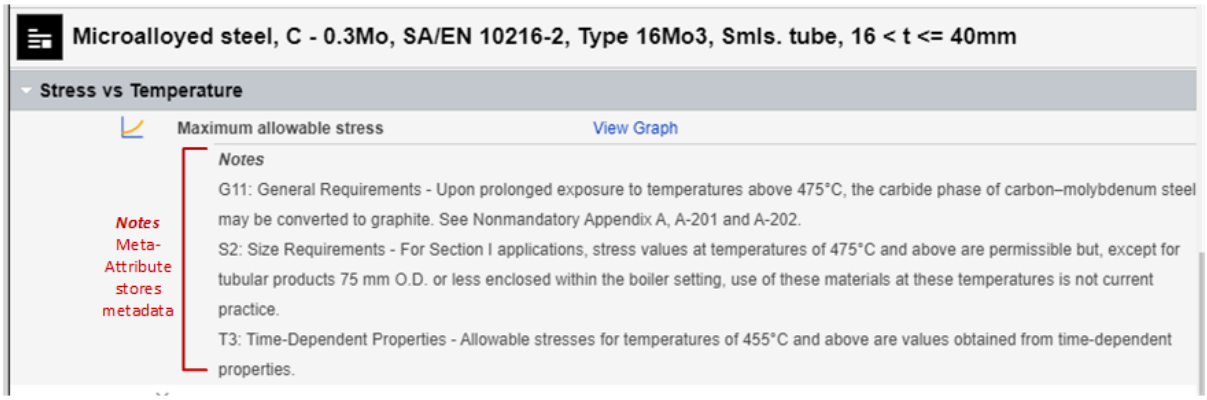
5 Attributes

5.1 Overview

In GRANTA MI, Attributes are used to define the characteristics of data. An Attribute's data type determines what kind of data it can store, and the characteristics and behavior of the stored data, such as the units for numerical data or the parameters for functional data.

For example, a polymer record may include Attributes that capture general information such as designation and trade names, physical properties such as *density*, mechanical properties such as *tensile strength* or *flexural modulus*, and bio-data such as *sterilizability* and *radiopacity*. Test data records may include Attributes that specify the *test ID*, *operator*, and the *date* the test was performed, as well as the test results.

Subsidiary information such as statistical information or notes about the sources of data, can be stored alongside the data value in **Meta-Attributes**, for example:



Meta-Attributes are table-level objects with the same properties as Attributes, and can be any GRANTA MI data type.

In the MI:Viewer and Explore applications, the visibility of Attributes and Meta-Attributes in datasheets is determined by the Layout configuration and, for MI:Viewer users, on additional end-user datasheet view preferences; see Layouts Meta-Attributes in Layouts.

5.2 Data types

The data type of an Attribute or Meta-Attribute determines what kind of data it can store. GRANTA MI supports the following data types:

Category	Icon	Data Type	Units?	Code	Use to store	Example Attributes
Numerical data		Integer	✔	INT	Numbers without a decimal point.	Failure strength points, Maximum Number of Passes, Pretest Cycles
		Point	✔	PNT	Floating point numbers.	Density, 0.2% Offset Yield Stress
		Range	✔	RNG	A pair of numeric values representing the upper and lower bounds of the value range.	Grain Size, Casting energy, Cutting speed
Text data		Short text		STXT	Text up to 255 characters (including spaces and punctuation).	Common Name, Color, Specimen ID
		Long Text		LTXT	Text up to 1,048,576 characters. Long Text values can include basic text formatting using Markdown formatting syntax.	Analysis Notes, Designation
		Discrete		DCT	Non-numerical data that can only take values from a discrete value set.	Polymer Class, Geographical Area, Material Type
Functional data		Float Functional	✔	FDA	Numerical data that is a function of one or more Parameters.	Tensile Stress/Strain, L, Thermal Conductivity with Temp.
		Equations and Logic	✔	MAFN	Numerical data calculated from other Attribute values using an equation.	Relative Cost Index (per unit), Specific Strength
		Discrete Functional		FDD	Discrete, non-numerical values, where the value is determined by one or more parameters.	Environmental Resistance
Media data		File		FIL	Entire data files—Word documents, PDF files, text files, video files.	Substances list, Analysis File, Test File
		Picture		PIC	Image files such as photos or schematics	Process schematic, BH curve, Fractography
		Hyperlink		HYP	Hyperlinks (URLs, email links)	Reference, More information

Category	Icon	Data Type	Units?	Code	Use to store	Example Attributes
Other		Date		DAT	Dates	Effective date, Date of Analysis
		Logical		LOG	Boolean data, where values are either True/Yes or False/No	Thermally conductive, Extrusion, Tests performed?
		Tabular	✔	TAB	Complex data organized into rows and columns; multiple, independent values each with their own data type	Test details, Restricted substances associated with this material

- The data type codes shown above are used when copying and pasting Attribute definitions between MI:Admin and the clipboard, and when importing/exporting data. Note that the Excel Importer and Excel Exporter plug-ins use two different codes for functional data to differentiate between series and grid functional data:
- FDA = series functional data
- GFA = grid functional data

5.3 View and manage Attributes

To view and manage the Attributes and Meta-Attributes in a Table: in the MI:Admin Schema tool, double-click to expand the table in the lower left Tables pane, then click on **Attributes**. All of the Attributes defined in the Table are listed.

5.4 Attribute properties

Attributes in GRANTA MI may have the following properties.

5.4.1 **Name** (all data types)

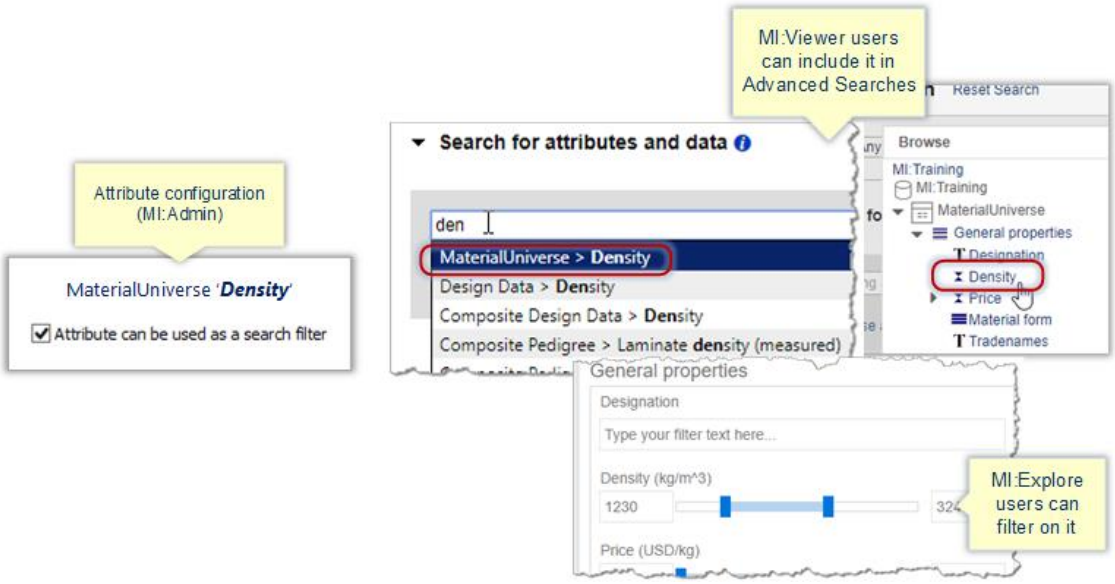
This may be up to 255 characters long. It cannot start or end with a space character, and any leading or trailing spaces will be stripped on saving the Attribute definition. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Polymer type" and "Polymer Type").

5.4.2 Attribute can be used as a search filter (all data types)

This property controls whether or not the Attribute will be available for selection as a search criteria in MI:Viewer Advanced Searches, and as a search filter in MI:Explore.

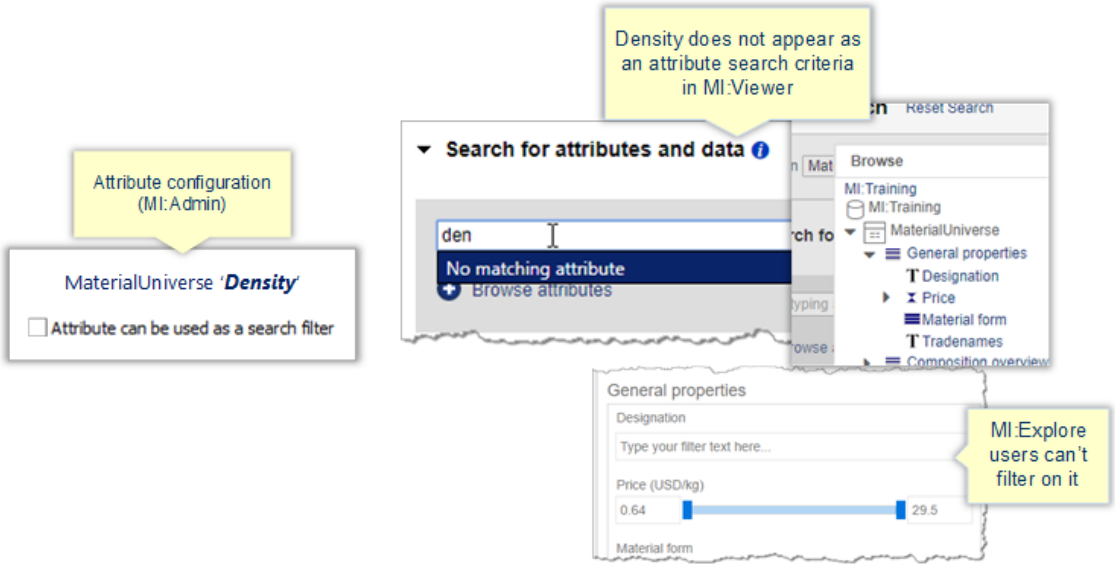
Attribute can be used as a search filter = ON

MI:Viewer users can select the Attribute from the **Search for attributes and data** list or from the Attribute browser on the Advanced Search page; MI:Explore users will see the Attribute in the list of filters in the Search pane (if it is included in the relevant Layout).



Attribute can be used as a search filter = OFF

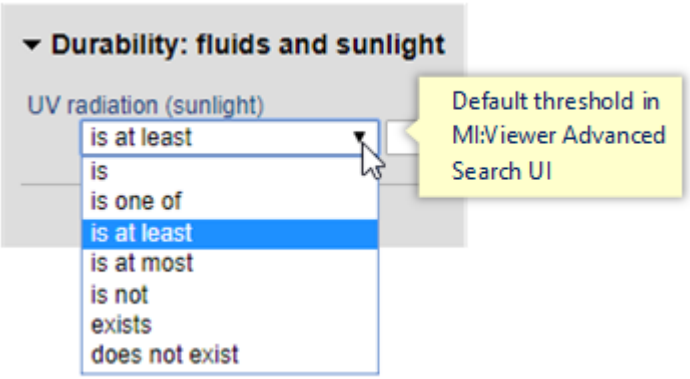
In an MI:Viewer Advanced Search, the Attribute will not appear in the **Search for attributes and data** list or in the Attribute browser; in MI:Explore, it will not appear as a filter in the Search pane (even if it is in the relevant Layout).



Note that the **Attribute can be used as a search filter** property does not determine whether or not text data in Attributes of this type is searchable: text values and text in embedded files is indexed—extracted, stored, and made searchable—automatically when the database is initially loaded into GRANTA MI, and when Records are added to a database. See also Searchable text.

5.4.3 Default Threshold Type (all data types)

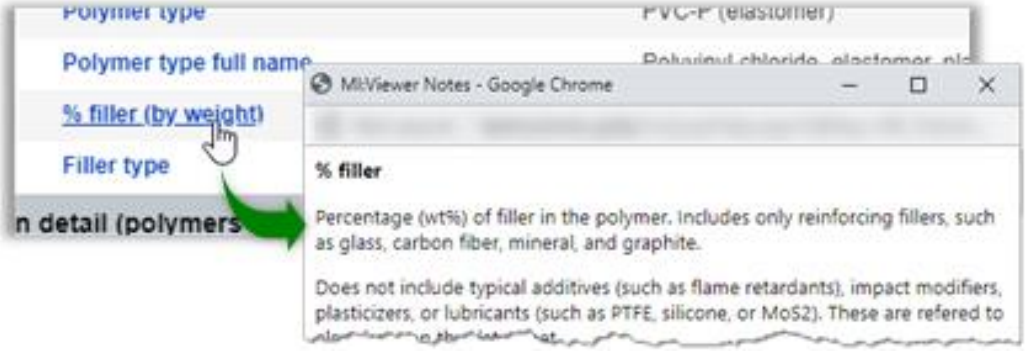
For Attributes that can be used as search filters, this specifies the default option offered when choosing the search criteria threshold on the **Advanced Search** page in MI:Viewer.



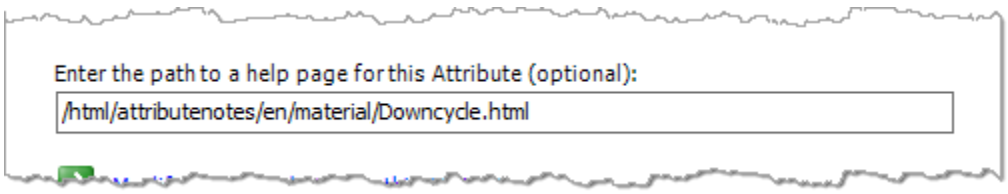
5.4.4 Help page (all data types)

Information about the Attributes in your system can be provided in the form of help topics that are displayed in a pop up window when users click on an Attribute name in MI:Viewer.

For example:



HTML-based help (HTML, CSS, images) may be created outside of GRANTA MI and then imported into the database on the **Files > Help Pages** tab in MI:Admin Schema tool. Attributes can then be linked with the relevant help topics by entering the help file path in the Attribute help page field, for example:



- A leading forward slash '/' is optional.
- The *Help Pages* folder name (under which all Attribute and Parameter help files are located, see Files) should not be included in this path.
- Multiple Attributes can be linked to the same help page.

5.4.5 **Values must be unique** (Integer, Short Text Attributes)

Enforces a uniqueness constraint to ensure that no duplicate values can be entered for the same Attribute within a table. When determining the uniqueness of a datum:

- Case is not significant: *YD0314A* is the same as *yd0314A*.
- Leading white space is significant; trailing white space is not significant.
- Accents are significant: the character ö is not the same as o.
- Unpopulated data and Not Applicable data are not considered, so a table can have multiple records with no data (or not applicable data).
- Meta-Attribute values are not considered.
- For access-controlled records, uniqueness is guaranteed across the whole data set (not just the records that the current user has access to).
- For version-controlled records, the value must not be in use in the latest *Unreleased* or latest *Released* version of any other records, but may be set to the current *released* value of this record.

It is not possible to set the **Values must be unique** property on an existing Attribute unless all records already have unique values for that Attribute.

When importing data using the Text Importer, Excel Importer, or Bulk Data importer, records are imported one by one; successfully imported records are not removed if a subsequent record fails to import because of a duplicate value conflict. The Data Updater will also apply uniqueness constraints during data updates, and record any errors in the Conflict Report.

5.4.6 **Discrete type** (Discrete Attributes)

Identifies the Discrete Type for Discrete (text) and Discrete Functional (numeric) Attributes; the available Discrete Types are defined at database level, and may be used in several different Tables and Attributes.

5.4.7 **Allow multiple values** (Discrete, Point Attributes)

For Discrete Attributes, this property allows the Attribute have more than one value at the same time. See Discrete Attributes.

For Point Attributes, this property allows the value to be a function of one or more parameters. See Point Attributes.

5.4.8 Treated as zero for Substitution if empty (Point, Range Attributes)

When using the Substitute module in MI:Viewer to perform substitution calculations, Attributes with no data can be ignored i.e. excluded from substitution calculations (this is the default behavior) or they can be treated as having a value of zero and included in the substitute calculations.

5.4.9 Attribute contains range data (Float Functional Attributes)

Specifies whether the Attribute can contain range data or point data. See Functional Attribute properties.

5.4.10 Axis Label (Float Functional, Equations & Logic Attributes)

Specifies the label which will appear on the chart X axis on datasheets. See Functional Attribute properties.

5.4.11 Allow extrapolation (Equations & Logic Attributes)

Specifies whether or not extrapolated values are shown on the data chart in MI:Viewer. Note that this property only affects the display of charts in datasheets; when searching, extrapolated values are not used. See Functional Attribute properties.

5.4.12 Show as a range (Equations & Logic Attributes)

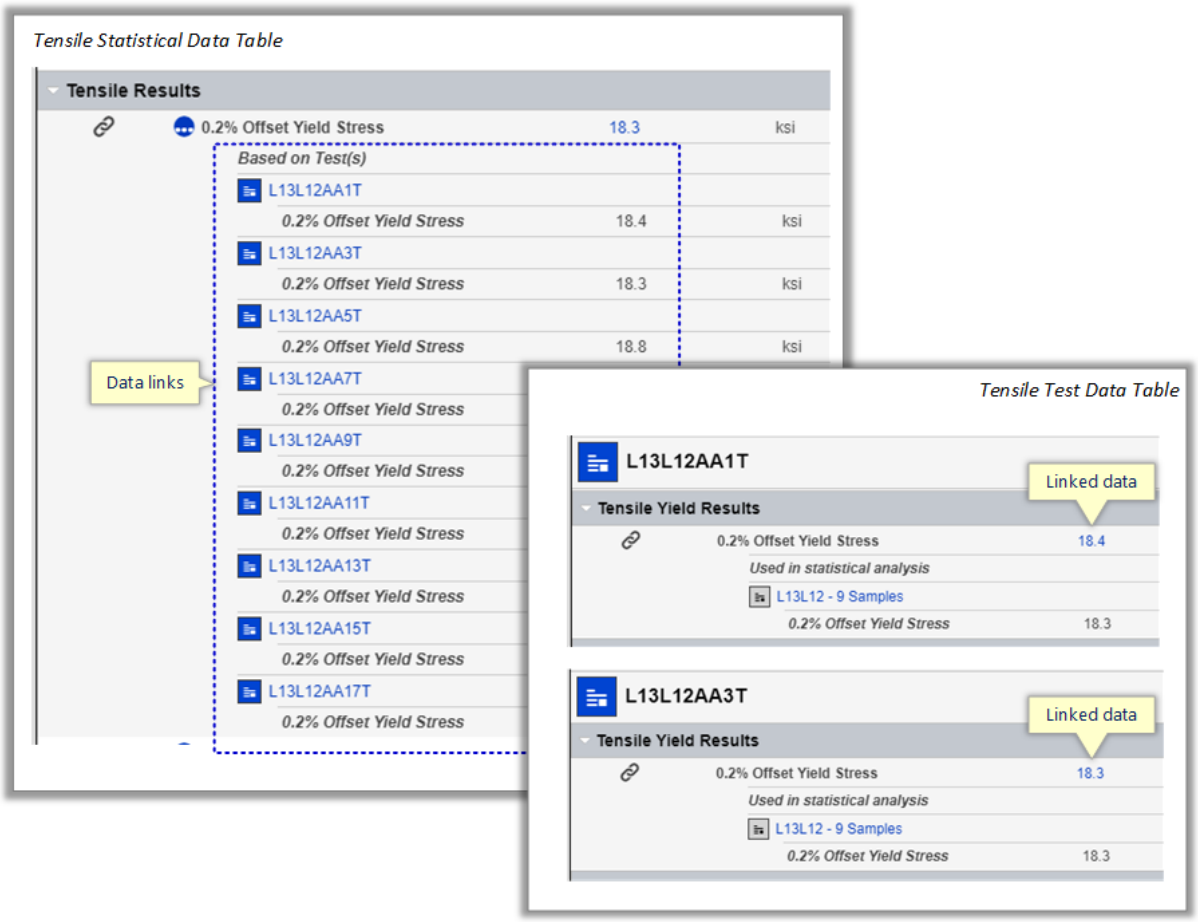
Specifies whether the data will be displayed a range or as a point. See Functional Attribute properties.

5.5 Data Link Groups

Data Link Groups allow additional/related data to be shown on a datasheet alongside the Attribute's data in MI:Viewer, for example, where a value has been derived from one or more values in other records. (Data links are not shown in MI:Explore).

Data Link Groups include static, permanent links from an Attribute in one record to an Attribute in another record, in the same or in a different Table. The Data Link Group specifies the target Table and Attribute; data links are then added to the group or removed from it manually in MI:Viewer, or via data import (auto-links).

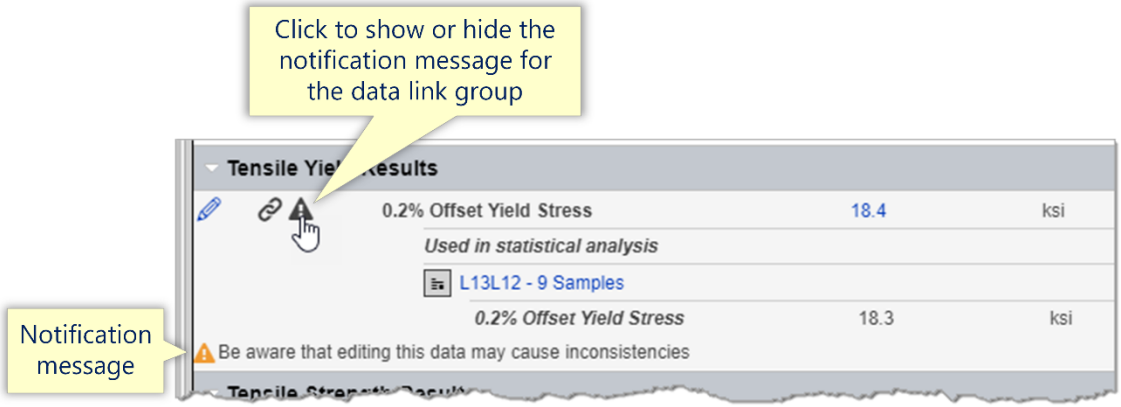
Example: the *Based on Test(s)* Data Link Group here shows the test data values that were used to derive the *0.2% Offset Yield Stress* value in a statistical data record:



When linked data for an Attribute is available, a Data link symbol appears next to the Attribute name in the datasheet, and users can click this to view the data.

The Data Link Groups for an Attribute are defined on the Attribute in the MI:Admin Schema tool, and specify:

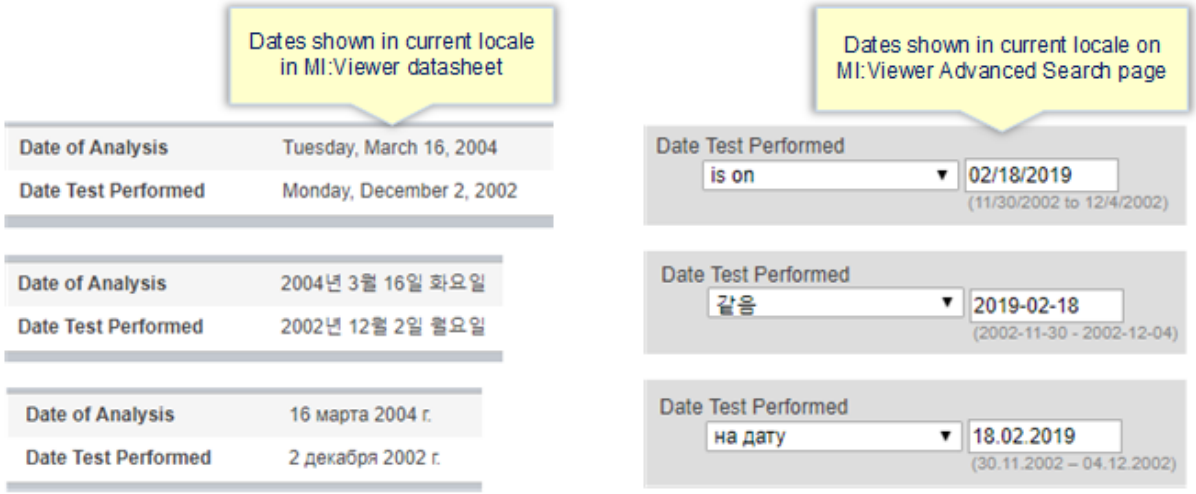
- The target Attribute(s) or Meta-Attributes, which may be in the same Table, or in a different Table.
- The name of the group, which is used as the heading on datasheets for the links in the forward direction (the links to the target records /Attributes). In the example above, this is *Based on Test(s*).
- A label which is used as the heading on datasheets for the link in the reverse (target -> source) direction. In the example above, this is *Used in statistical analysis*.
- An optional notification message which, when set, is shown in the target record datasheet.



5.6 Date Attributes

Date Attributes (DAT) are used to store calendar dates, for example, Saturday, January 1, 2000. Searches can be performed on date fields using the search criteria *is on*, *is on or after*, *is on or before*, *is between*, *exists*, and *does not exist*.

Note that the display of Date values in MI applications depends on regional date and time settings, for example:



5.6.1 Date Attribute properties

Name

This may be up to 255 characters long. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Date of build" and "Date of Build").

Attribute can be used as a search filter

Determines whether or not the Attribute can be used as a search filter in MI:Explore and MI:Viewer.

Default Threshold Type

Specifies the default option offered when choosing the search criteria threshold for the Attribute on the **Advanced Search** page in MI:Viewer.

For Date Attributes, this can be set to one of: *is on*, *is on or after*, *is on or before*, *is between*, *exists*, *does not exist*.

Help page (optional)

The location and name of a file containing help information about the Attribute/data ("Attribute Note"); when this is configured, MI:Viewer users can view the notes by clicking on the Attribute name in a datasheet in MI:Viewer.

5.7 Media Attributes (File, Picture, Hyperlink)

File, image, and hyperlink data are stored in GRANTA MI databases as *embedded media*. Where the appropriate tools are available on the client PC and the IIS configuration permits it, embedded media items can be viewed within the Granta application window, or in a separate tab/window. If an embedded media item cannot be opened, security permitting, users will be able to download a copy of the file to the client PC.

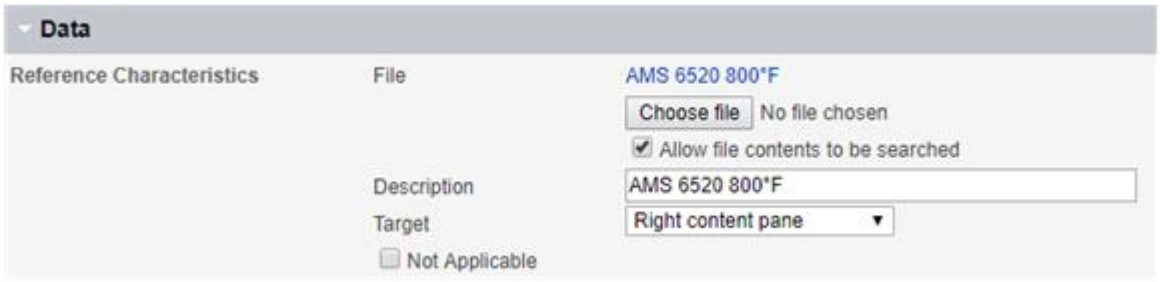
- The contents of embedded media items cannot be edited in GRANTA MI.
- It is not possible to compare embedded media items in a report.
- Access control permissions (in a permission-based access control system) can be applied to File and Picture Attributes in the normal way, though these permissions are not applied to the file itself.

5.7.1 File Attributes

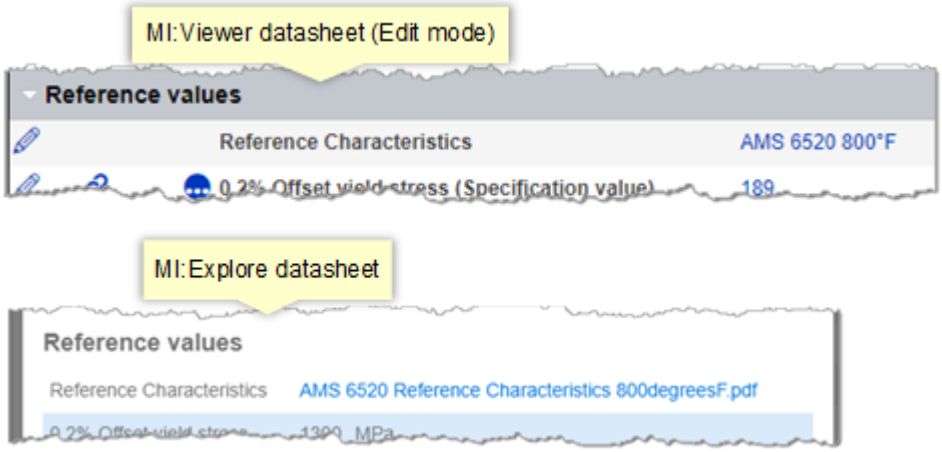
File Attributes (FIL) may be used to store data files up to 500 MB in size, for example, text or PDF files, Word documents, video files. (Files that are larger than this (up to a maximum of 2 GB) may be

stored outside the MI database and accessed from within MI via specially-configured hyperlink Attributes.)

File Attribute in MI:Viewer Edit mode:

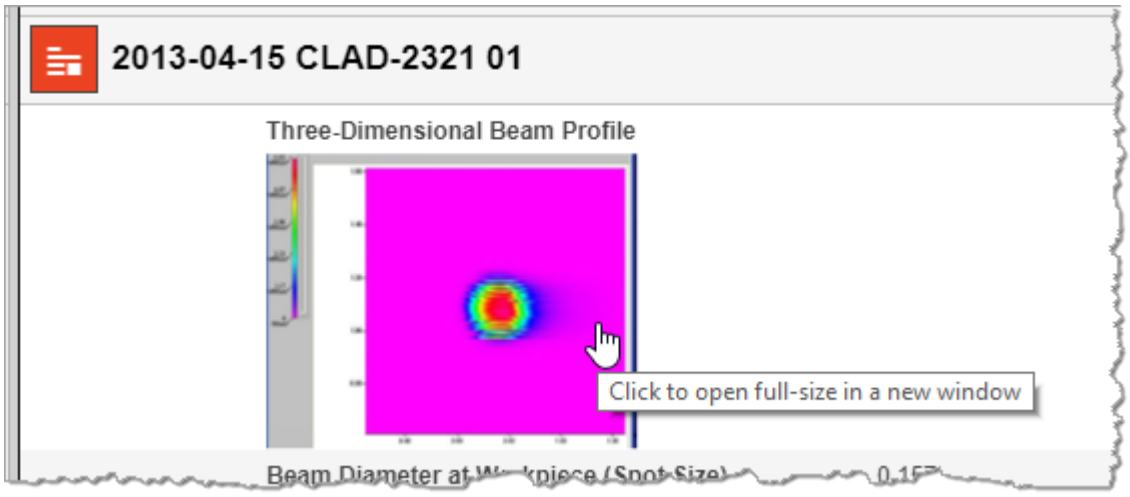


Corresponding File data viewed in MI:Viewer and MI:Explore (when specified, the Description is shown instead of the filename in MI:Viewer):



5.7.2 Picture Attributes

Picture Attributes (PIC) can be used to store images in records, for example, schematics or BH curves. Image files up to 10 MB in size are supported.



5.7.3 Hyperlink Attributes

Hyperlink Attributes (HYP) can be used to store Uniform Resource Locator (URL) data consisting of a protocol and location e.g. http://www.grantadesign.com. They are typically used for Website links and email addresses, for example.

Hyperlink Attributes may include Replacement strings, which are placeholders that allow part of the Address URL to be defined in MI:Admin and then inserted into the URL when it is resolved in the user's MI:Viewer browser.

Using Hyperlinks to provide access to large files on disk

Files that are too large to store in GRANTA MI (larger than 500 MB, up to a maximum of 2 GB) can be stored on disk on the same machine as MI:Server, and then downloaded direct from the server to the MI:Viewer client machine via specially-configured Hyperlink Attributes in datasheets. To configure this feature:

- Define a Hyperlink Attribute with a Short Text Meta-Attribute that will be used to specify the location of linked files.
- Edit the MI:Viewer configuration file ViewerSettings.config to add a new <remotefiles> element that identifies (a) the root folder of the target file(s), and (b) the Meta-attribute used to store the relative path to files within this root folder.
- In MI:Viewer, for records that will include links to external files, edit the Hyperlink Attribute to specify the target filename and to specify the address string in the following format: /mi/RemoteFile/Get?AttributeId={AttributeId}&recordid={recordid}&databaseKey={databaseKey}

For full details of the required configuration to enable this feature, see the *GRANTA MI Administrator's Guide*.

5.7.4 Searchable text

Text that has been indexed can be searched. Indexing—extracted text, storing it, and making it searchable—happens automatically when a database is initially loaded into GRANTA MI, and whenever data is added to a database.

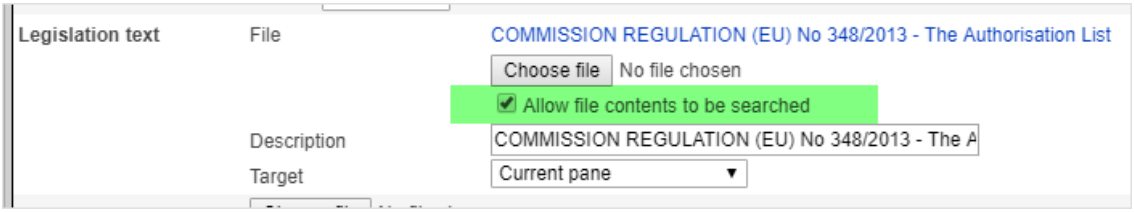
- The *Description* field for File and Hyperlink data is indexed, and therefore may be searched.
- Filenames and hyperlink URLs are not indexed, and cannot be searched.

The text in individual embedded data files (e.g. documents, spreadsheets, text files, PDFs) may or may not be indexed:

- These common file types are indexed by default: .doc .docx .html .key .mbox .msg .ods .odt .odp .pages .pdf .ppsm .ppsx .pptm .ppt .pptx .txt .rtf .xls .xlsx .xml

  Embedded files that are not one of these file types will not be indexed, and so will not be searchable even if the **Allow file contents to be searched** option is set on the datum in MI:Viewer (see below). You can specify the file types that are indexed by default via search configuration settings in the MI:Server configuration file MIServer.exe.config; see the *GRANTA MI Configuration Guide* for details.

- An **Allow file contents to be searched** option can be set on the file datum in MI:Viewer to prevent an individual file from being indexed, and so exclude it from searches, for example:



Note that the **Attribute can be used as a search filter** property on the Attribute does **not** determine whether or not **data** stored in a File Attribute is indexed/searchable, only whether or not the Attribute can be used as a search criteria/search filter in MI:Viewer and MI:Explore.

If a File Attribute has the **Attribute can be used as a search filter** option deselected, and a File datum has the **Allow file contents to be searched option** enabled:

- Users will not be able to select the Attribute as a search criteria on the Advanced Search page in MI:Viewer, or use it as a filter in MI:Explore.
- Text searches performed in MI:Viewer and MI:Explore will return the record if the file data or description contains the search term.

5.7.5 Media Attribute properties

Name

This may be up to 255 characters long. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Drawing file" and "Drawing File").

Attribute can be used as a search filter

Determines whether or not the Attribute can be used as a search filter in MI:Explore and MI:Viewer.

Default Threshold Type

Specifies the default option offered when choosing the search criteria threshold for the Attribute on the **Advanced Search** page in MI:Viewer.

For Picture Attributes, you can choose from: *exists*, *does not exist*. For File and Hyperlink Attributes, you can choose from: *contains, exists*, *does not exist*.

Help page (optional)

The location and name of a file containing help information about the Attribute/data ("Attribute Note"); when this is configured, MI:Viewer users can view the notes by clicking on the Attribute name in a datasheet in MI:Viewer.

5.8 Functional Attributes

Functional Attributes are used to store data which is a function of one or more parameters. Typical material database examples include *Young's Modulus vs Temperatur*e (1 parameter, *Temperature*), *Stress vs Time* (2 parameters, *Time* and *Temperature*).

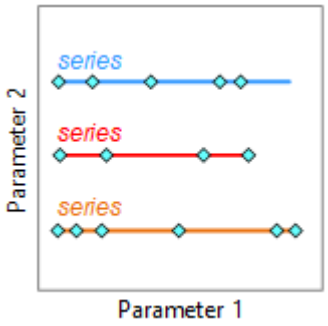
GRANTA MI supports 3 different functional data types: Float Functional, Discrete Functional, and Equations and Logic.

5.8.1 Float Functional Attributes

Float Functional Attributes store a point or range value which is dependent on one or more discrete (text) or numerical parameters. Values are stored to a maximum of 15 significant figures.

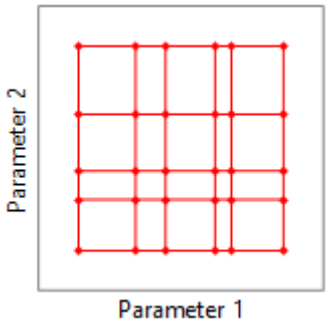
The data points for Float Functional Attributes may be specified in series or gridded format:

Series data:



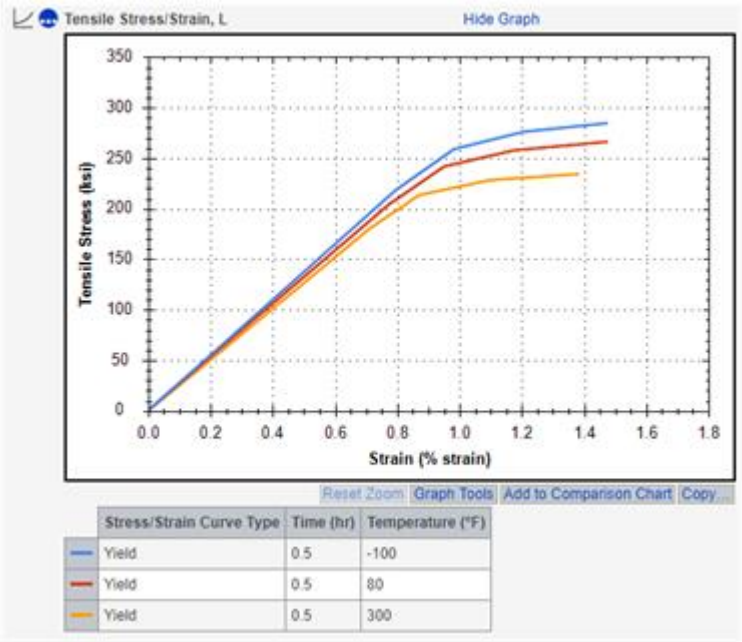
- Data points are specified in more or more series, where only the parameter changes
- The number of points across each series can vary.

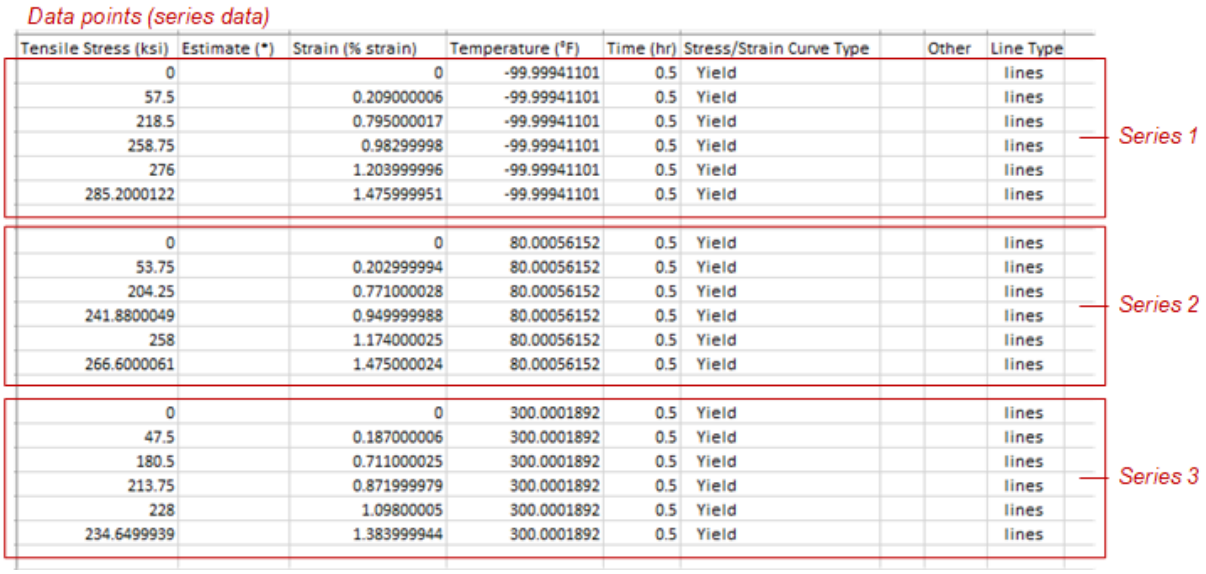
Gridded data:



- Data points are specified as a grid
- All parameter values are specified for each data point
- Suitable for data that is regular, complete, highly curated, or where you want users to be able to change the x-axis, or where you want to do multi-dimensional interpolation in GRANTA MI.

Example: this series Float Functional Attribute includes tensile stress/strain data:





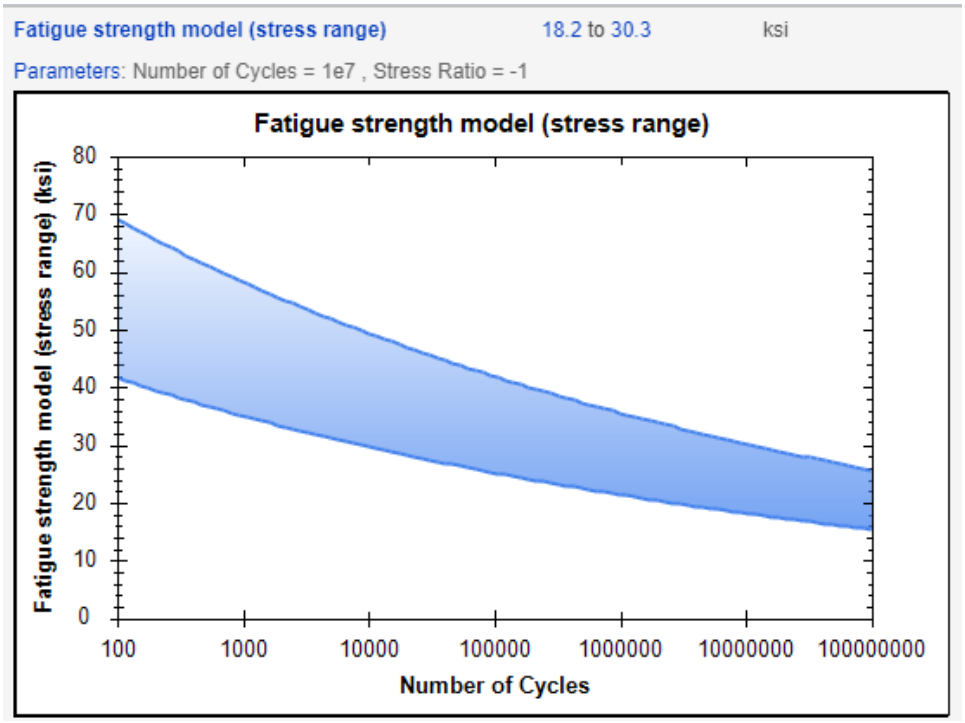
Note that while the internal code used for Float Functional data is generally FDA (for example, when pasting Attribute definitions from the clipboard), when importing and exporting functional data, the Excel Importer and Excel Exporter plug-ins use different codes to differentiate between series and grid functional data:

- FDA = series functional data
- GFA = grid functional data

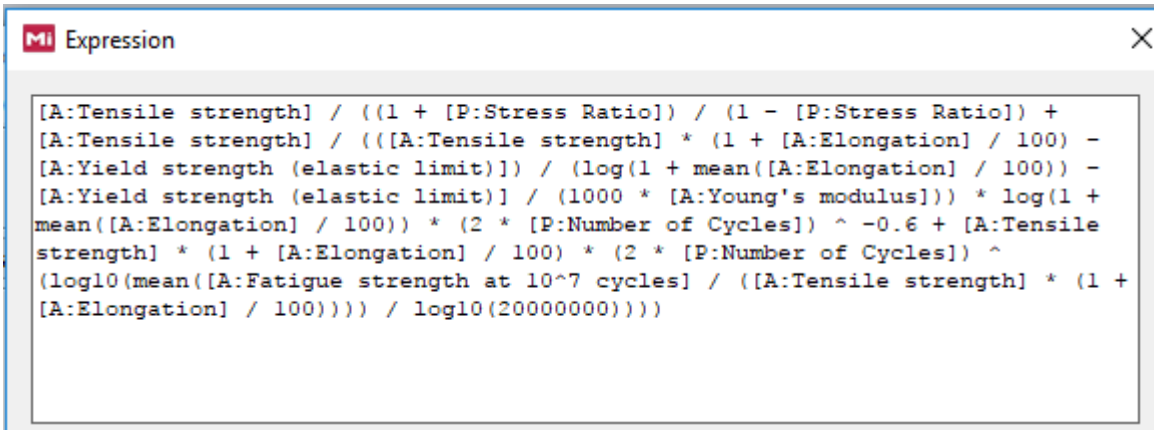
5.8.2 Equations and Logic Attributes

Equations and Logic Attributes store numerical data as an expression that describes how the value varies dependent on one or more parameters. You may also see this type of Attribute also referred to as "math functional" or "EEL". Values are stored to a maximum of 15 significant figures.

For example, the data on applied stress vs. number of cycles to failure in this steel record is provided by the Equations and Logic Attribute *Fatigue strength model (stress range)*:



This data is calculated using the *Fatigue Model* Expression:

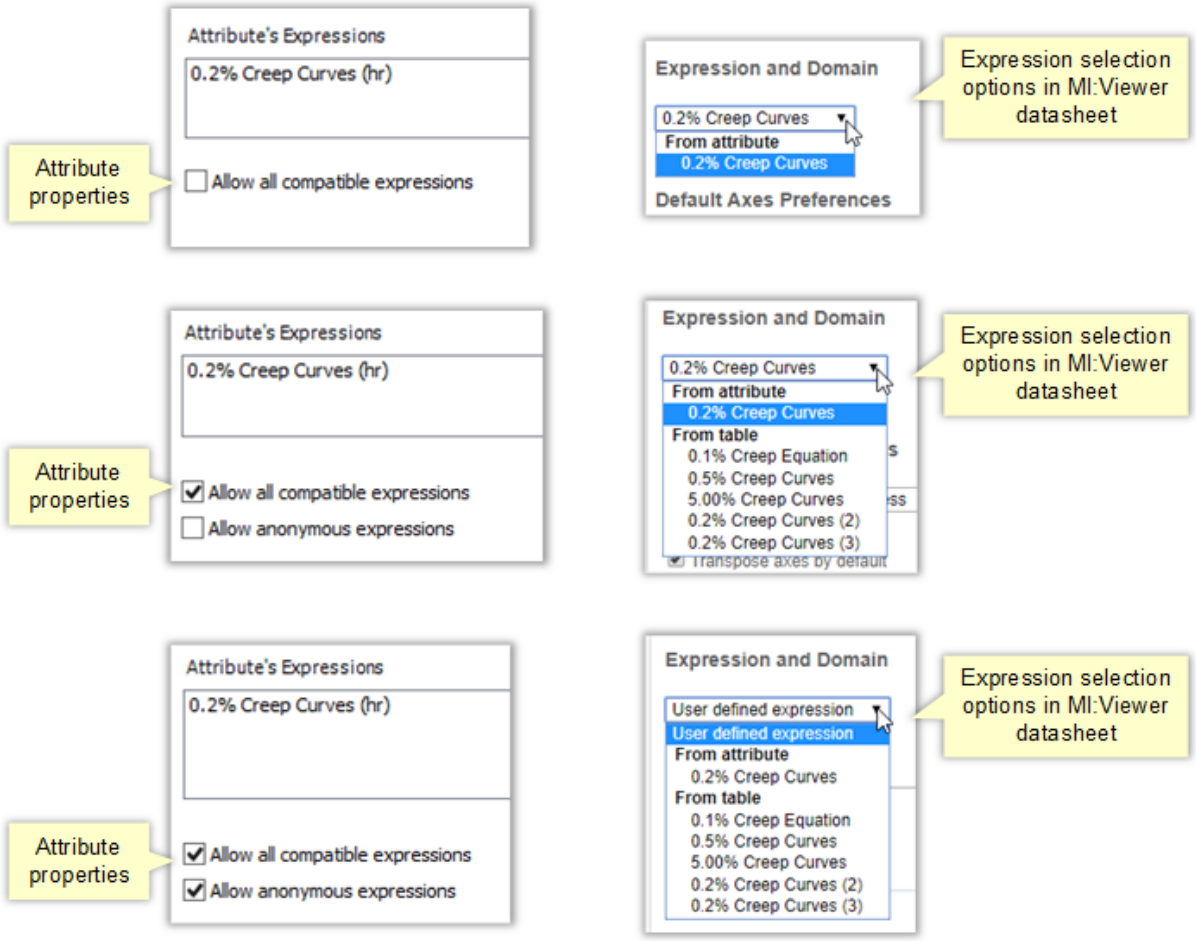


The expressions used to evaluate Equations and Logic data are defined per-table in the MI:Admin Schema tool, and may include:

- numeric Attributes (e.g. *Elongation*, *Tensile strength*)
- parameters (e.g. *Stress Ratio*) and constants

- functions (e.g. log, mean)

Equations and Logic Attributes can be configured to permit MI:Viewer users to specify a different expression, if required:



5.8.3 Discrete Functional Attributes

Discrete Functional Attributes store a discrete text value which is dependent on one or more discrete text or numerical parameters.

Example: an Attribute representing chemical resistance data could have a five-point scale: 'Very good' to 'Very poor', where the value depends on the chemical environment (Fresh Water, Salt Water, and so on).

5.8.4 Functional Attribute properties

Name

This may be up to 255 characters long. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Young's modulus with temperature" and "Young's Modulus with temperature").

Attribute can be used as a search filter

Determines whether or not the Attribute can be used as a search filter in MI:Explore and MI:Viewer.

Default Threshold Type

Specifies the default option offered when choosing the search criteria threshold for the Attribute on the **Advanced Search** page in MI:Viewer.

- For numeric functional Attributes, choose from: *is*, *is at least*, *is at most*, *is between*, *exists*, *does not exist*.
- For Discrete functional Attributes, choose from: *is*, *is one of*, *exists*, *does not exist*.

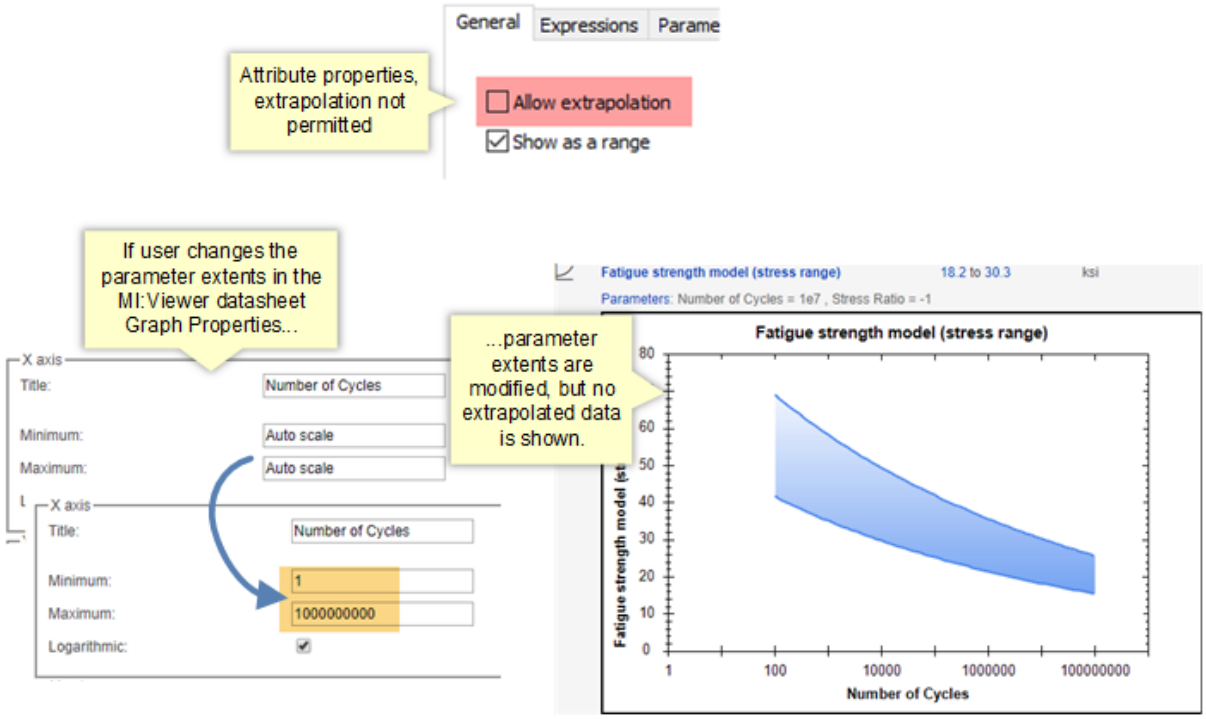
Help page (optional)

The location and name of a file containing help information about the Attribute/data ("Attribute Note"); when this is configured, MI:Viewer users can view the notes by clicking on the Attribute name in a datasheet in MI:Viewer.

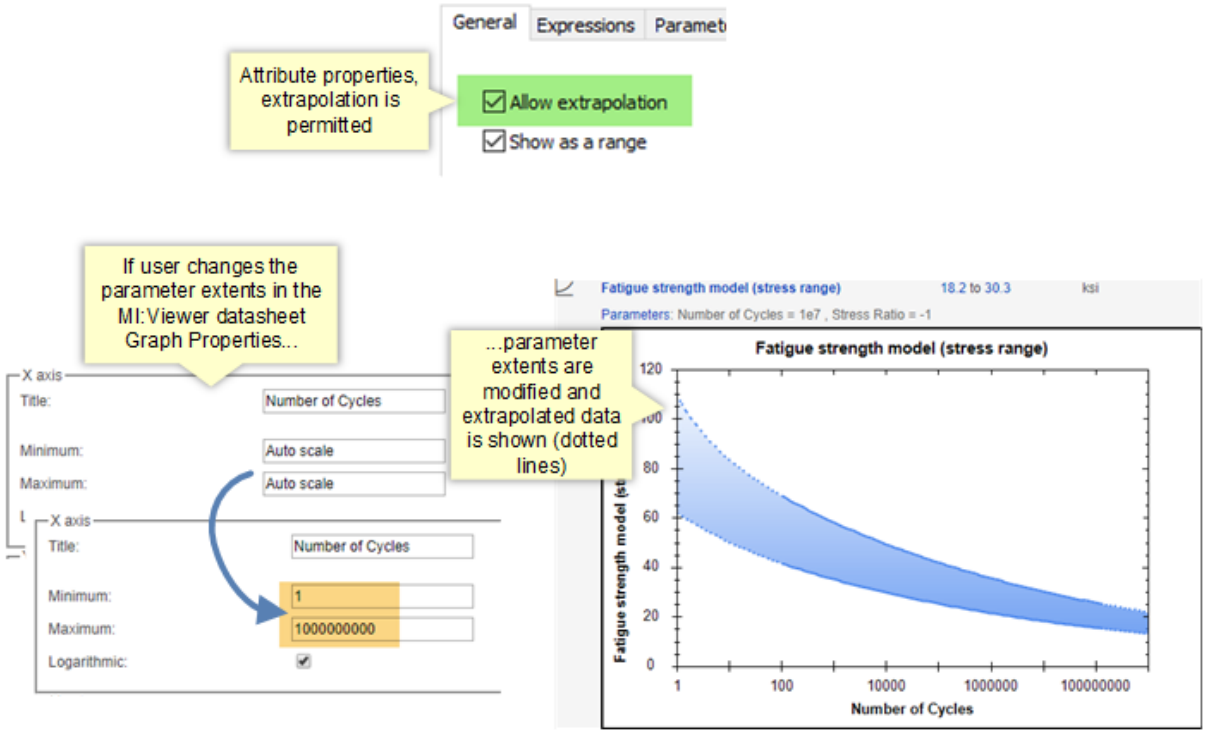
Allow extrapolation (Equations and Logic Attributes)

Specifies whether or not extrapolated values may be shown on the data chart. Note that this property only affects the display of charts in datasheets; when searching, extrapolated values are not used.

Extrapolation not allowed:



Extrapolation allowed:

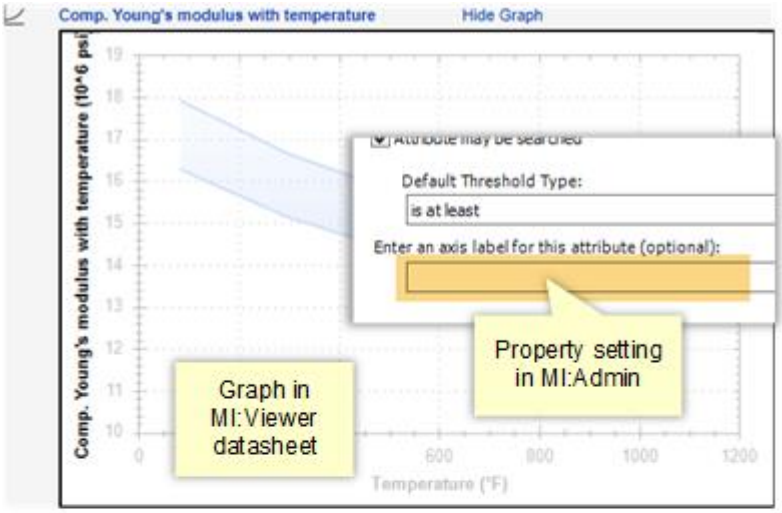


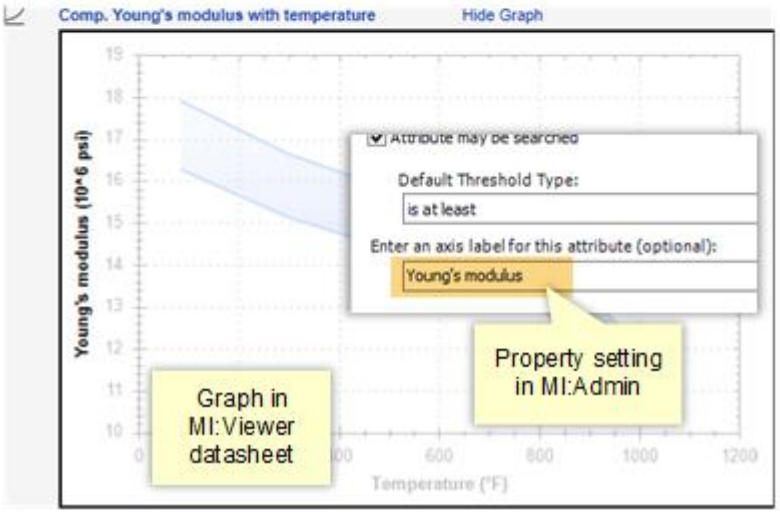
Attribute contains range data (Float Functional)

Specifies whether or not the Attribute includes range data.

Axis Label (Float Functional and Equations and Logic)

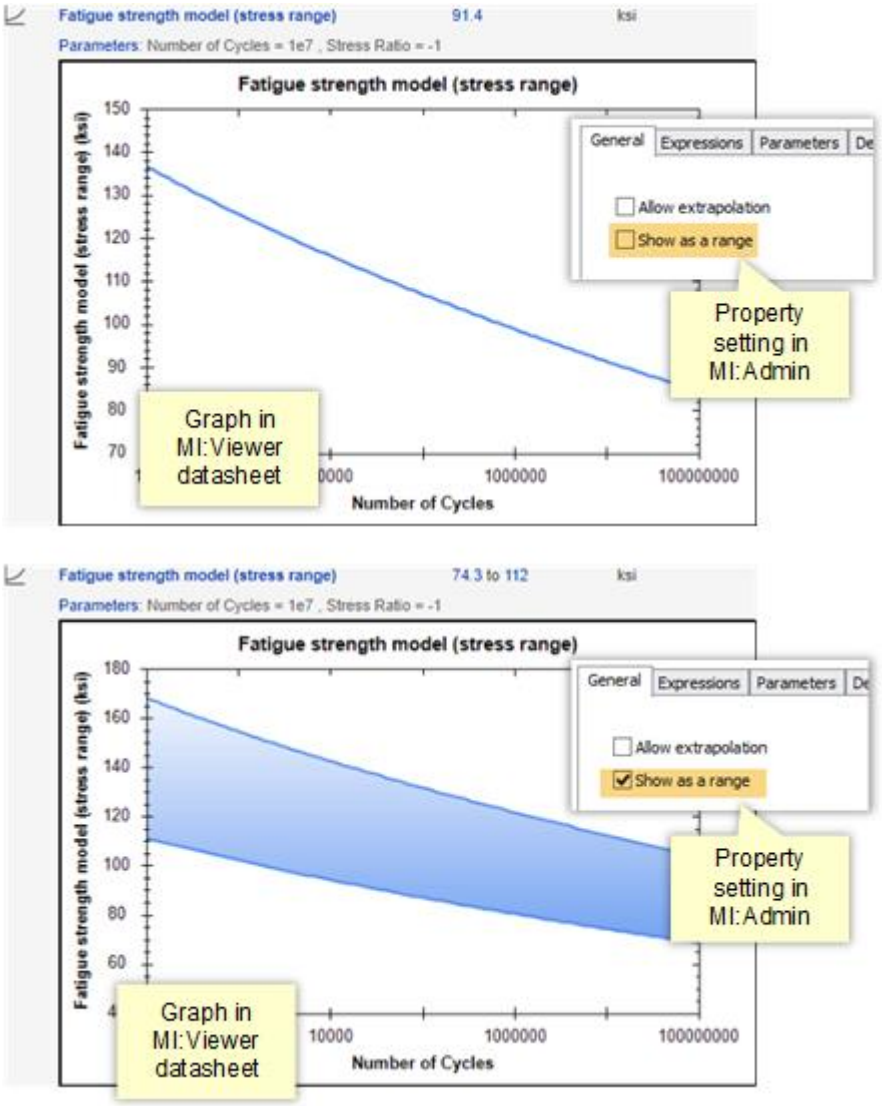
By default, the chart X axis label will be the Attribute name and unit. You can specify a different label for the X axis, if required. For example, in the picture below, the label "Young's modulus" is specified:





Show as a range (Equations and Logic Attributes)

Specifies whether the data will be displayed on the chart a range or as a point.



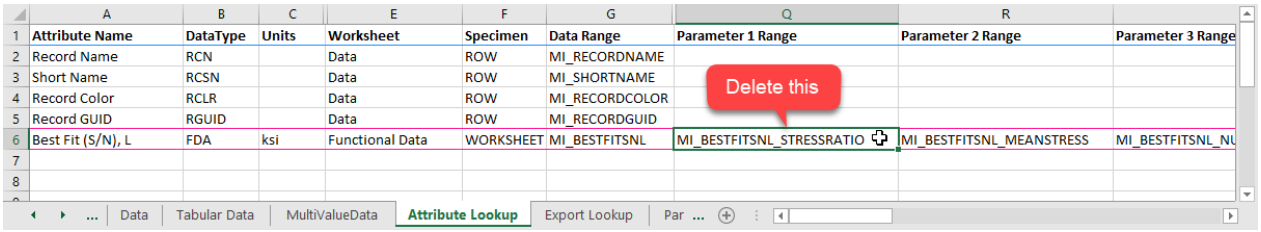
Parameters

The parameters for a functional Attribute may be selected from the available Parameters defined in the database. Click on the **Parameters** tab and add Parameters from the list. The Parameter at the top of the list of selected Parameters will be the default x-axis for new data; use the Move Up and Move Down buttons to select a different Parameter as the default x-axis.

A Parameter should only be removed from a functional Attribute if you are certain that there is no functional data where it is used, for example, graphs that include references to it. **Removing a Parameter that is being used will result in corruption of all functional data that uses that parameter.** This data will not be recoverable. If you need to delete a Parameter from a functional

Attribute in a populated database, we recommend that you follow this procedure to ensure that data integrity is preserved:

1. In MI:Toolbox, export all the Attribute data using the Excel Exporter.
2. Edit the exported Excel file to remove the parameter from the relevant row in the Attribute Lookup sheet. For example, to delete the *Stress Ratio* parameter from this *Best Fit (S/N), L* functional Attribute:



	A	B	C	E	F	G	Q	R	
1	**Attribute Name**	**DataType**	**Units**	**Worksheet**	**Specimen**	**Data Range**	**Parameter 1 Range**	**Parameter 2 Range**	**Parameter 3 Range**
2	Record Name	RCN		Data	ROW	MI_RECORDNAME			
3	Short Name	RCSN		Data	ROW	MI_SHORTNAME			
4	Record Color	RCLR		Data	ROW	MI_RECORDCOLOR			
5	Record GUID	RGUID		Data	ROW	MI_RECORDGUID			
6	Best Fit (S/N), L	FDA	ksi	Functional Data	WORKSHEET	MI_BESTFITSNL	MI_BESTFITSNL_STRESSRATIO	MI_BESTFITSNL_MEANSTRESS	MI_BESTFITSNL_NU
7									
8									

3. In MI:Admin, delete the original Attribute and then recreate it with the required Parameters.
4. Import the modified export file containing the data back into the database.

Default Content

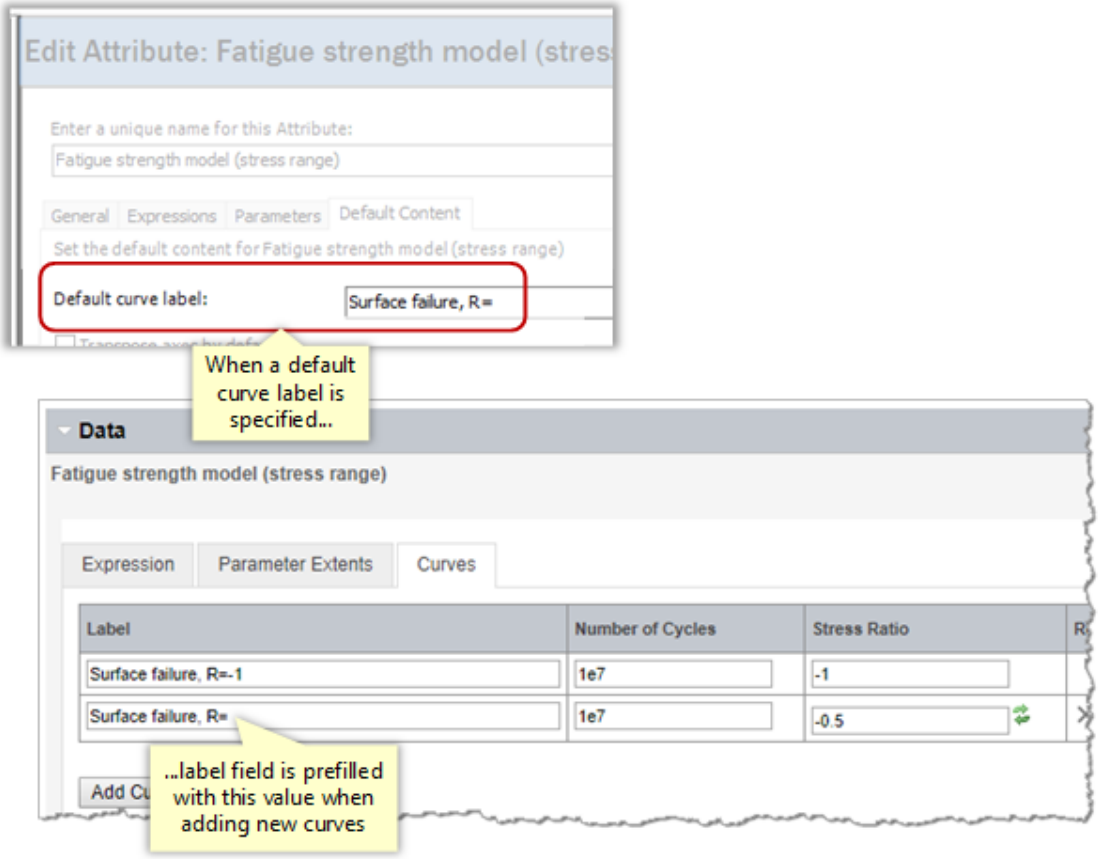
You can specify a number of default settings for Functional Attributes that are used when a new datum is added (that is, when new data is added to an 'empty' Attribute). These setting only affect new data; changing the Default Content setting will have no affect on any existing data.

Float Functional Attribute parameter defaults:

- **Interpolation Type**—you can select one of Linear, Cubic Spline, or none, or choose **Inherit from Parameter** to use the parameter's interpolation type.
- **Scale**—you can select Linear or Logarithmic scale, or choose **Inherit from Parameter** to use the parameter's scale.
- **Default Value**—you can enter a specific value, or choose **Inherit from Parameter** to use the parameter's default value.

Equations & Logic Attribute defaults:

Default curve label—A default label for new curves; this can be edited in MI:Viewer



- **Transpose axes by default**—When adding a new datum, invert the chart axes i.e. show the Attribute value on the x-axis and the Parameter value on the y-axis. Note that, when importing Equations and Logic data, an Invert option (TRUE/FALSE) allows this setting to be specified explicitly; if the Invert option is not specified in the import file, the default behavior is NOT to transpose the incoming data, and the **Transpose axes by default** setting on the Attribute is ignored.
- Logarithmic scale
- Default expression
- Default x-axis Parameter
- Parameter defaults—**Default Value** and **Default Range**, as for Float Functional Attributes

5.9 Logical (Boolean) Attributes

Logical (LOG) Attributes are used to store Boolean data (Yes/No, True/False). For example:

Hydrogen embrittling?	No
Internal	Yes
Military Qualified	No
RoHS compliant?	Yes
Wear protection, abrasion	Yes

5.9.1 Logical Attribute properties

Name

This may be up to 255 characters long. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Valid Test?" and "Valid test?").

Attribute can be used as a search filter

Determines whether or not the Attribute can be used as a search filter in MI:Explore and MI:Viewer.

Default Threshold Type

Specifies the default option offered when choosing the search criteria threshold for the Attribute on the **Advanced Search** page in MI:Viewer.

For Logical Attributes, this can be set to one of: *is*, *exists*, *does not exist*.

Help page (optional)

The location and name of a file containing help information about the Attribute/data ("Attribute Note"); when this is configured, MI:Viewer users can view the notes by clicking on the Attribute name in a datasheet in MI:Viewer.

5.10 Numerical Attributes

5.10.1 Integer Attributes

Store numbers that are whole, or without a decimal point. For example:

Maximum Number of Layers	60
Maximum Number of Passes	1

Possible values: -2147438648 to 2147438648

5.10.2 Point Attributes

Store exact numeric data with decimal points. For example:

First ionization energy	9.79	eV
Heat of vaporization	90000	ft.lbf/mol
Herfindahl-Hirschman Index (HHI)	5330	
Lattice parameter, a	1.63e-7	mil
Magnetic susceptibility	-0.00000528	
Melting temperature	1500	°F
Molar volume	0.79	in^3/mol

Point values may be entered as an integer or decimal, or formatted with an exponent, and values are stored to a maximum of 15 significant figures.

Values must be within the range:

- 1.5e-45 to 3.4e38
- 1e-039 to 1e+038 for positive values

Multi-value Point Attributes can have a range of data values based on one or more parameters. For example, the value for these two Point Attributes depends on the value of the *Basis* parameter (*Mean*, *A-basis*, or *B-basis*):

0° compression strength - measured	64.7	ksi (A-basis)
	80.2	ksi (B-basis)
	102	ksi (Mean)
0° compression strength - normalized	63.9	ksi (A-basis)
	78.7	ksi (B-basis)
	99.9	ksi (Mean)

5.10.3 Range Attributes

Store a pair of numeric values representing the upper and lower bounds of a value range. For example:

Vaporization CO2	817 to 901	lb/lb
Vaporization energy	4.69e6 to 5.17e6	BTU/lb
Vaporization water	131000 to 188000	in^3/lb
Warmth to touch	7.03 to 7.25	
Water usage	1250 to 1360	in^3/lb
Welding (electric) CO2	0.0806 to 0.168	lb/ft

Range Attributes can be **open,** meaning that only the maximum, or only the minimum value is defined:

Al (aluminum)	0.1	% (minimum)
Annual world production	2.26e9	ton/yr (maximum)

Values may be entered as an integer or decimal, or formatted with an exponent.

Values are stored to a maximum of 15 significant figures, and must be within the range:

- ±1.5×10-45 to ±3.4×1038
- 1e-039 to 1e+038 for positive values

5.10.4 Numerical Attribute properties

Name

This may be up to 255 characters long. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Aging Temperature" and "Aging temperature").

Attribute can be used as a search filter

Determines whether or not the Attribute can be used as a search filter in MI:Explore and MI:Viewer.

Default Threshold Type

Specifies the default option offered when choosing the search criteria threshold for the Attribute on the **Advanced Search** page in MI:Viewer.

For numerical Attributes, you can choose from: *is*, *is at least*, *is at most*, *is between*, *exists*, *does not exist*.

Help page (optional)

The location and name of a file containing help information about the Attribute/data ("Attribute Note"); when this is configured, MI:Viewer users can view the notes by clicking on the Attribute name in a datasheet in MI:Viewer.

Treated as zero for Substitution if empty

This property allows Point and Range Attributes with no data to be included in substitution calculations and treated as having a value of zero. The default behavior is to exclude empty Attributes from substitution calculations.

Unit

Specifies the unit for Point and Range Attributes. Available units are defined at the database level, and may be used in several different Tables and Attributes.

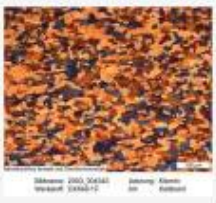
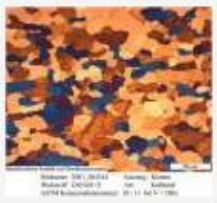
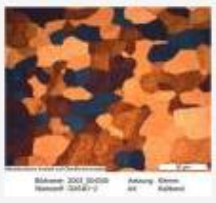
Values must be unique (Integer only)

Enforces a uniqueness constraint on Integer Attributes that are used a record identifiers, to ensure that no duplicate values can be entered within the same Table.

5.11 Tabular Attributes

Tabular Attributes are used to store a collection of data within a record, typically a summary of information related to the record, for example:

Metallographic examination tabular data in a steel record:

Type	10x	20x	50x	100x	200x	500x	1000x	700x
light microscopy								

Lay-up sequence details tabular data in a composite record:

Ply/layer number	Ply/layer material name	Ply/layer manufacturer	Ply/layer lot number	Ply/layer form	Ply/layer architecture	Ply/layer orientation
6	S-Glass Unitape S2/SP381	3M	22E7P	Prepreg	1-D tape	0°

Specifications tabular data in a Coating record:

Specification	Class	Type	Coating Name	Specification Type
AMS 2427				AMS
Bombardier PPS 24.02				Not Milspec or AMS or API
McDonnell Douglas PS 13143				Not Milspec or AMS or API
MIL-C-81706			IVD Al - contact resistance	Milspec
MIL-DTL-83488B	1, 2, 3	I, II		Milspec

Legislations restricting its use tabular data in a substance record:

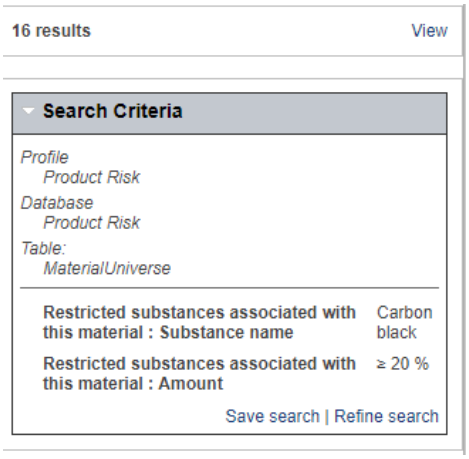
Legislation name	Legislation rating	Effective date	Geographical area
REACH - The Candidate List	High risk of phase-out	Tuesday, October 28, 2008	EU
TSCA Section 6	Banned with conditions	Saturday, January 1, 1977	US

The data stored as a Tabular Attribute is organized into rows and columns:

- Each row represents a separate data item; for example, a relevant test, specification, or legislation.
- Each column stores a specific property of that data, such as an ID, date, amount, status, rating.
- The data in Tabular columns may be local data (stored within the same record), or it may be retrieved from records in other tables (linked data).
- All data types except Functional data and Tabular data can be stored in Tabular Attributes. (See Media data for maximum supported file sizes for File and Picture data; however, note that including lots of files as Tabular data can quickly make datasheets impractical to load and/or edit. This is because editing Tabular data requires fetching all of the data (including media data) from the database, and saving back again, and so is computationally expensive.)

Local Tabular data can be searched via the **Quick Search** box in the MI:Viewer toolbar, or in the Search box in MI:Explore. Where the Tabular Attribute has the *Attribute can be used as a search filter* property enabled, its linked and local Tabular data may be searched via an Attribute search in MI:Viewer. This allows searches to be performed like the one below, where *Substance name* is a Linked Attribute and *Amount* is local data in the *Restricted substances associated with this material* Tabular Attribute.

"In the Product Risk database, find all materials which contain greater than 20% of carbon black":



Linked Columns and Linked Attribute columns may be included in search filters if the underlying linked Attribute has the Attribute can be used as a search filter property enabled. This is independent of the state of this property on the Tabular Attribute itself.

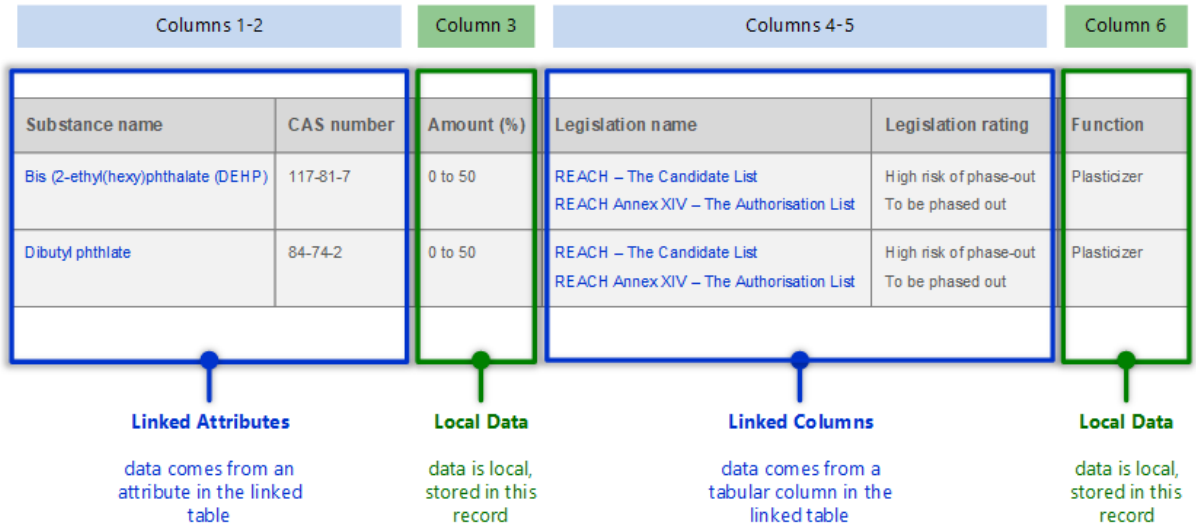
Note: the tabular columns that are shown on datasheets in MI:Viewer and MI:Explore, and the order in which the columns appear on datasheets, are determined by the Layout used when viewing the data, and not by the Attribute definition. A Layout may include some or all of the tabular columns defined in the Attribute, in any order; see Layouts. If you make changes to the ordering of tabular columns in the Attribute definition, or add columns, users will not see those changes until you update the relevant Layouts in which the Attribute is included.

5.11.1 Tabular columns

Tabular columns have a 'type' which determines what can appear in the column:

- *Local Data*—data which is defined in the source record
- *Linked Attribute*—the data from an Attribute in another table
- *Linked Column*—the data from an entire tabular data column in another other table
- *Linked Records*—a link to a record in another table

This example shows a Tabular Attribute in a material record that includes a mixture of local data and data from records in a separate Restricted Substances table:



Taking a closer look at this example:

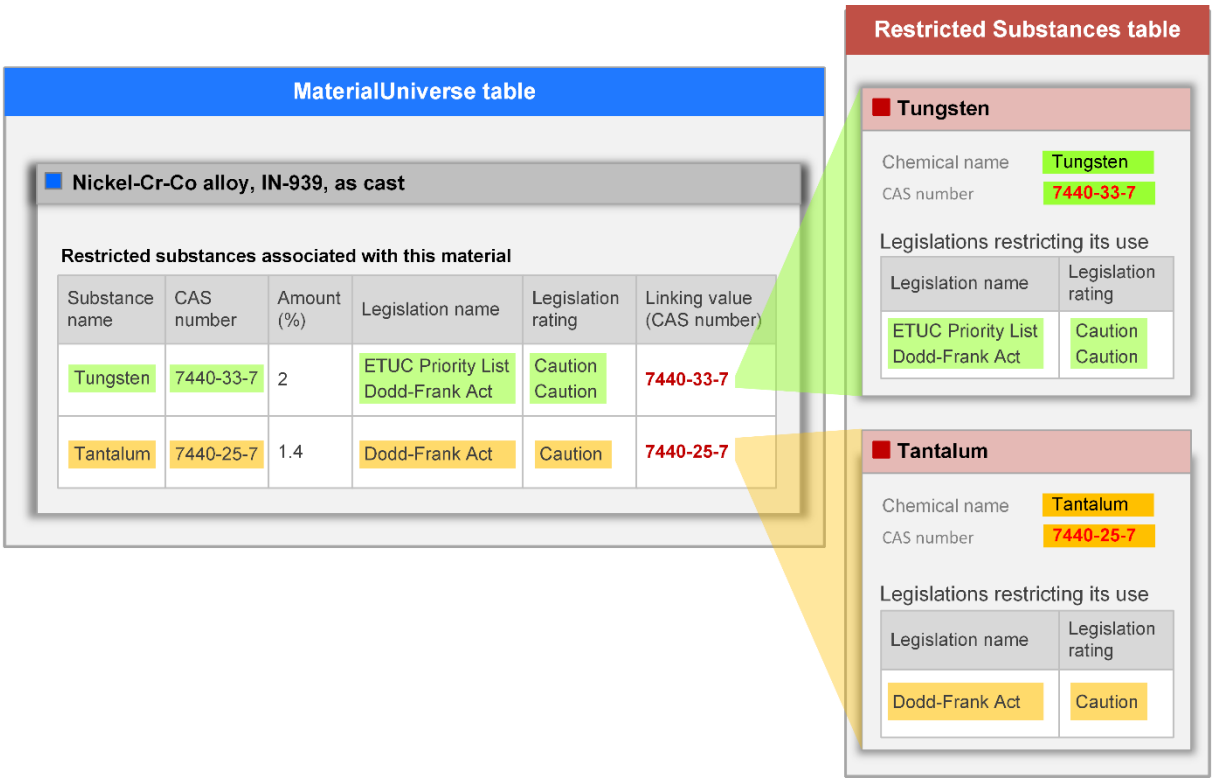
- ***Substance name***—the data in this column is obtained from the *Chemical name* Attribute in the Restricted Substances table.
- ***CAS number***—the data in this column is obtained from the *CAS number* Attribute in the Restricted Substances table.
- ***Amount*** and ***Function***—these are Local Data columns, and so the values in these columns are defined in the material record.
- ***Legislation name***—the data in this column is obtained from the *Legislation name* tabular column of the *Legislations restricting its use* Tabular Attribute in the Restricted Substances table.
- ***Legislation rating***—the data in this column is obtained from the *Legislation rating* tabular column of the *Legislations restricting its use* Tabular Attribute in the Restricted Substances table.

5.11.2 Linking values

Linked data in a Tabular Attribute is retrieved from other records by matching a *linking value* specified in the source tabular row against a Short Text *linking Attribute* value in the target table.

When the *linking value* in the Tabular row matches the linking Attribute value in a record in the target table, data from that record can be retrieved and shown in the Tabular row.

For example, *CAS number* is used as the linking value in this Tabular Attribute; when the specified value matches a *CAS number* value in Restricted Substances records, data from those substance records is "pulled into" the material record:



The linking value appears in an extra column when editing Tabular data in MI:Viewer and MI:Explore; when viewing Tabular data, it is not shown.

Tabular rows that include only local data (no linked record exists) may be shown or hidden on datasheets via a configuration option on the Tabular Attribute definition, Hide rows that have a linking value but no corresponding records.

5.11.3 Tabular Attribute properties

As well as properties that determine how the Attribute may be used in MI:Viewer Advanced searches, there are a number of additional properties that can be set on Tabular Attributes which control how the tabular data is displayed on datasheets in MI:Viewer.

Name

This may be up to 255 characters long. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Test details" and "Test Details").

Attribute can be used as a search filter

Determines whether or not the Attribute can be used as a search filter in MI:Explore and MI:Viewer.

Default Threshold Type

Specifies the default option offered when choosing the search criteria threshold for the Attribute on the **Advanced Search** page in MI:Viewer.

For Tabular Attributes, you can choose from: *exists*, *does not exist*.

Help page (optional)

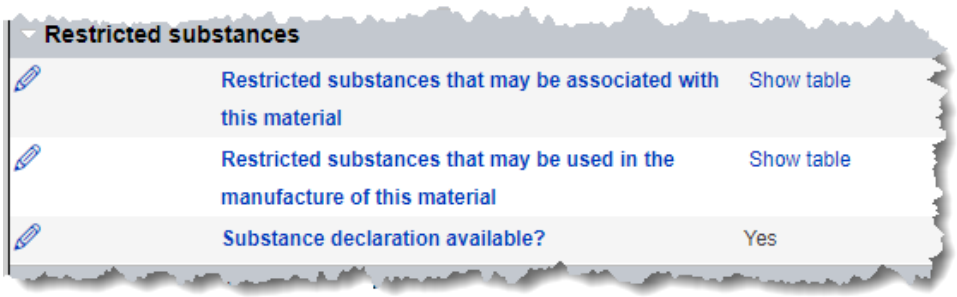
The location and name of a file containing help information about the Attribute/data ("Attribute Note"); when this is configured, MI:Viewer users can view the notes by clicking on the Attribute name in a datasheet in MI:Viewer.

Showing/hiding all data on datasheet load

The entire table of data may be visible by default, or hidden by default, when a datasheet is opened. This feature is controlled with a configuration option on the Tabular Attribute definition, **Show full table by default.**

When hidden by default, a **Show table** link (or **Show full table** where a summary table is provided) appears on the datasheet which allow users to view the data; for Tabular Attributes with a large number of rows/columns, this may be the preferred display option.

All data hidden by default:

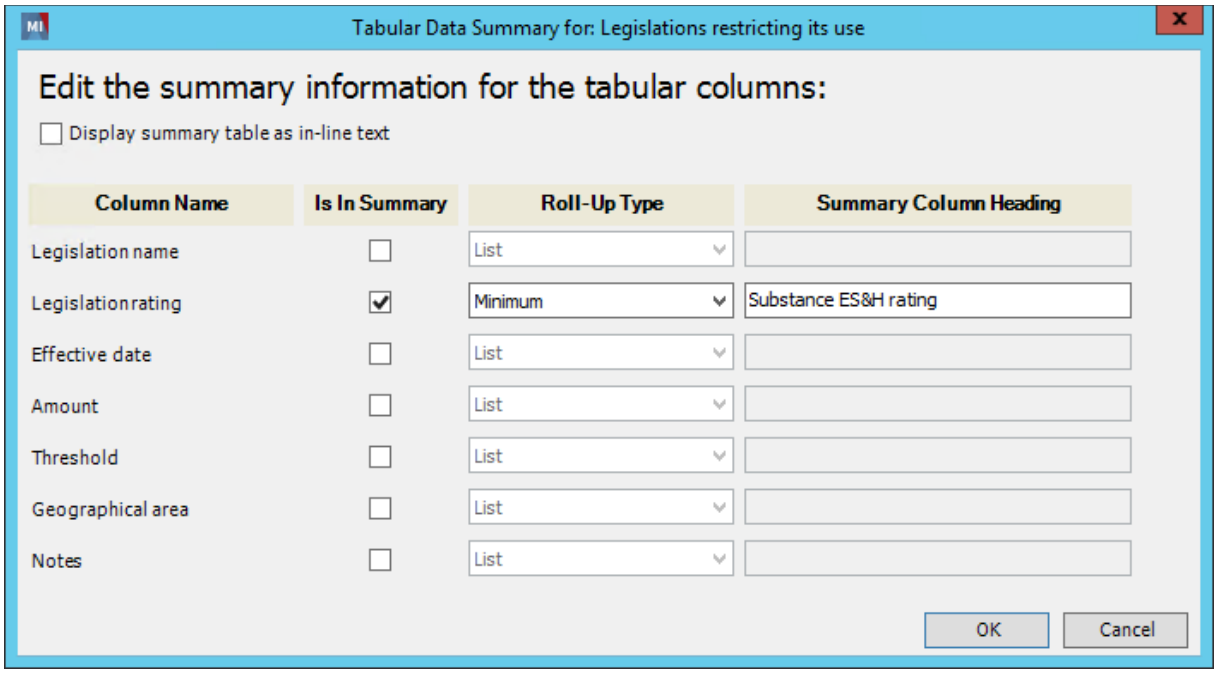


All data shown by default:

Data Summary

Where the Tabular data is complex, with many rows/columns, a Data Summary can be defined to provide a quick overview of the key data points without taking up too much space on the datasheet. Where a Data Summary is defined, this is always shown instead of the full table of data when the datasheet is opened, with a link that users can click to open the full table in the datasheet.

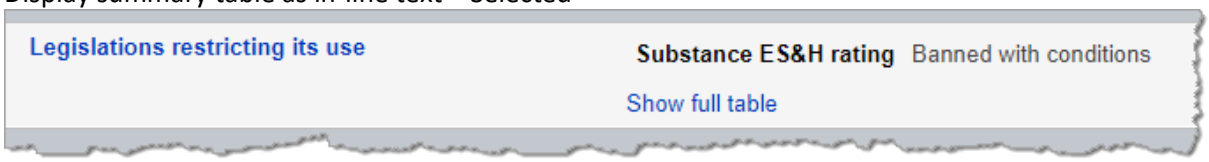
Any of the columns in the Tabular Attribute can be included in the Data Summary; the summary columns are configured independently of the Tabular columns, with their own headings and roll-up type. For example, in the Attribute shown above, the *Legislation rating* Tabular column (column name="*Legislation rating*", Roll-up type=*List*) is included in the Data Summary with a different column heading ("*Substance ES&H rating*") and a different roll-up type (*Minimum*):



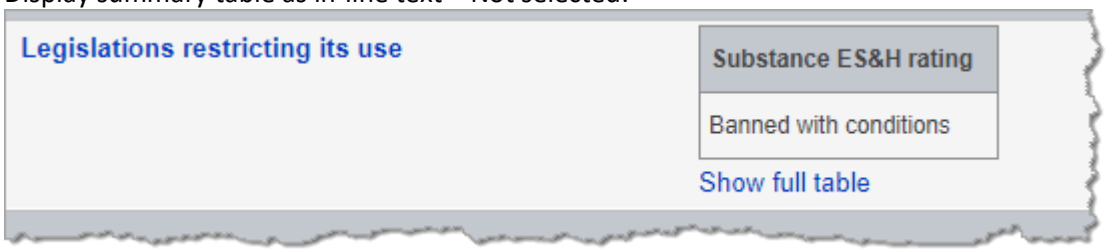
Inline vs rows/columns format

A Data Summary may be displayed inline or as rows/column; this is controlled with a configuration option on the summary table definition, **Display summary table as in-line text**:

Display summary table as in-line text = Selected



Display summary table as in-line text = Not selected:



Showing/hiding rows with no linked records

If no record is found in the target Table to match the linking Attribute in a row, the row can be hidden by default, removing rows with no useful data. This only affects the view of data in Read mode; Edit users will always see all rows. This feature is controlled with a configuration option on the Tabular Attribute definition, **Hide rows that have a linking value but no corresponding records.**

Rows with linking value but no linked record are hidden (Read mode):

Substance name	CAS number	Amount (%)	Legislation name	Legislation rating	Function
Bis (2-ethyl(hexy)phtha....	117-81-7	0 to 50	REACH – The Candidate List REACH Annex XIV – The Authorisation List	High risk of phase-out To be phased out	Plasticizer
Dibutyl phthlate	84-74-2	0 to 50	REACH – The Candidate List REACH Annex XIV – The Authorisation List	High risk of phase-out To be phased out	Plasticizer

Rows with linking value but no linked record are shown (Read mode):

Substance name	CAS number	Amount (%)	Legislation name	Legislation rating	Function
Bis (2-ethyl(hexy)phtha....	117-81-7	0 to 50	REACH – The Candidate List REACH Annex XIV – The Authorisation List	High risk of phase-out To be phased out	Plasticizer
		0 to 5			Pigment
		0 to 50			Plasticizer
Dibutyl phthlate	84-74-2	0 to 50	REACH – The Candidate List REACH Annex XIV – The Authorisation List	High risk of phase-out To be phased out	Plasticizer

Rows with no linked data

Rows with linking value but no linked record (Edit mode) - in Edit mode, the Linking value and linked record columns are always shown:

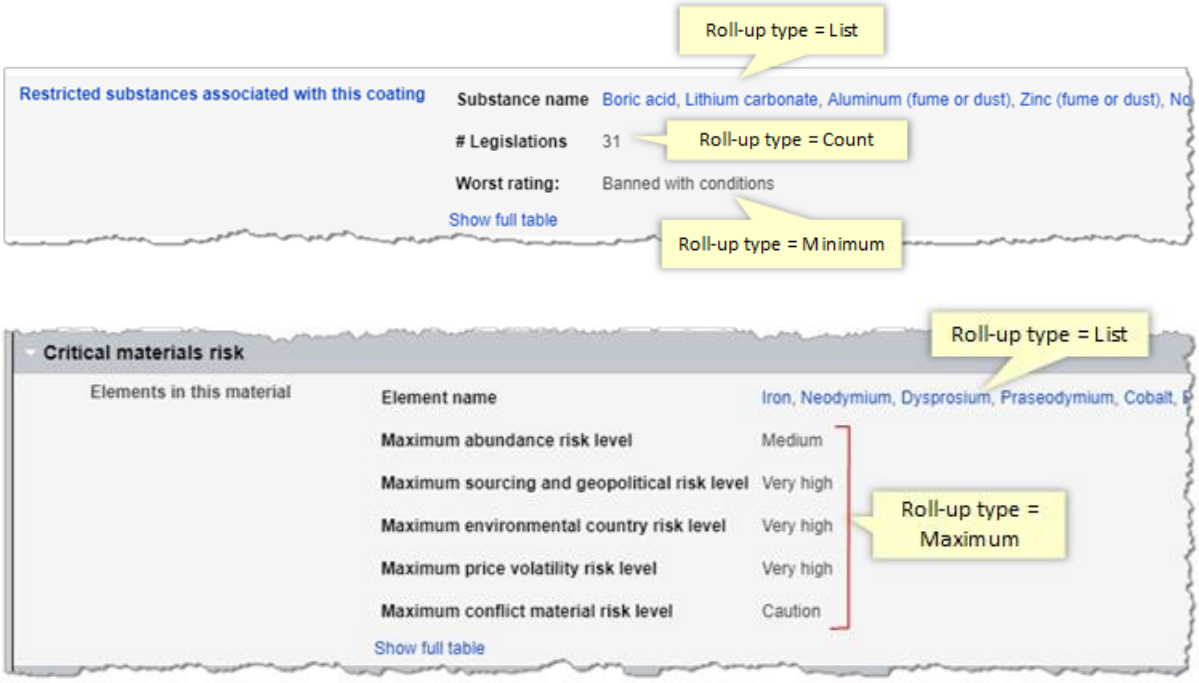
Substance name	CAS number	Amount (%)	Legislation name	Legislation rating	Function	Linking value (CAS number)	Linked records found
Bis (2-ethyl(hexy)phtha....	117-81-7	0 to 50	REACH – The Candidate List REACH Annex XIV – The Authorisation List	High risk of phase-out To be phased out	Plasticizer	117-81-7	Bis (2-ethyl(hexyl)phthalate) (DEHP)....
		0 to 5			Pigment	12656-85-8	
		0 to 50			Plasticizer	84-69-5	
Dibutyl phthlate	84-74-2	0 to 50	REACH – The Candidate List REACH Annex XIV – The Authorisation List	High risk of phase-out To be phased out	Plasticizer	84-74-2	Dibutyl phthlate [84-74-2] [v1]

Rows with no linked data

Roll-up type for tabular columns

Tabular columns have a **roll-up type**, which determines how multiple values are displayed when a cell contains more than one value, e.g. list of all of the values, the total number of values, the maximum or minimum value. The valid roll-up types for a column depend on the data type of the linked data. Local Data columns do not have a roll-up type.

Roll-up types are particularly useful for Data Summary columns, where they can be used to show a list of values, the total number values, or maximum or minimum values, for example:



Roll-up type	Behavior	Available for these data types
List	Show all values, including duplicate values.	All
List Distinct	Show all values excluding any duplicate values. For Hyperlink data, the basis of distinction is the URL, and the target and description are ignored; for Picture data, it is the database identity.	All
Count	Show the total number of values, including any duplicate values.	All
Count Distinct	Show the total number of values excluding duplicate values.	All
Count	Show the total number of values, including any duplicate values.	All
Minimum	Show only the minimum value. For Range data, the arithmetic mean value for each range is compared.	Range, Point, Integer, Date, Discrete
Maximum	Show only the maximum value.	Range, Point, Integer, Date, Discrete

Roll-up type	Behavior	Available for these data types
Mean	Show the arithmetic mean. For Range data, the arithmetic mean value for each range is used in the calculation.	Range, Point, Integer
Std Dev	Show the standard deviation of the sample. For Range data, the arithmetic mean value for each range is used in the calculation.	Range, Point, Integer
Sum	Show the total value when all the values are added. For Range data, the arithmetic mean value for each range is used in the calculation.	Range, Point, Integer

5.12 Text Attributes

5.12.1 Short Text Attributes

Use for short (up to 255 characters) alphanumeric values.

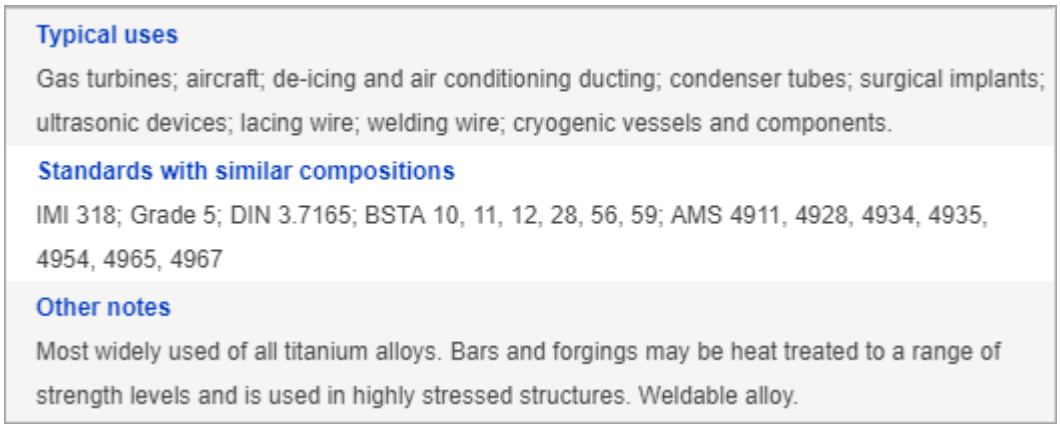
For example, these 5 Short Text Attributes appear in a record in the *Restricted Substances* table of the MI:Training database:

CAS number	84-74-2
Chemical name	Dibutyl phthalate
CI Hazard Class and Category Code(s)	Repr. 1B, Aquatic Acute 1
CI Hazard Statement Code(s)	H360Df, H400
Classification	Repr. Cat. 2; R61, Repr. Cat. 3; R62, N; R50

5.12.2 Long Text Attributes

Use for long blocks of text, up to a maximum of 1,048,576 characters.

Examples from a record in the *MaterialUniverse* table of the MI:Training database:

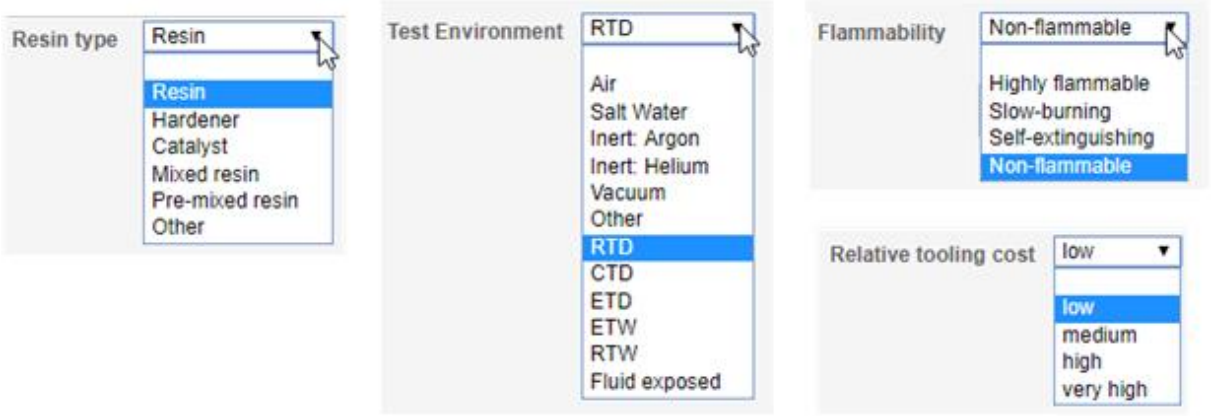


The text in Long Text Attributes can be formatted with Markdown if required, for example, to provide headings, and font and list formatting.

5.12.3 Discrete (text) Attributes

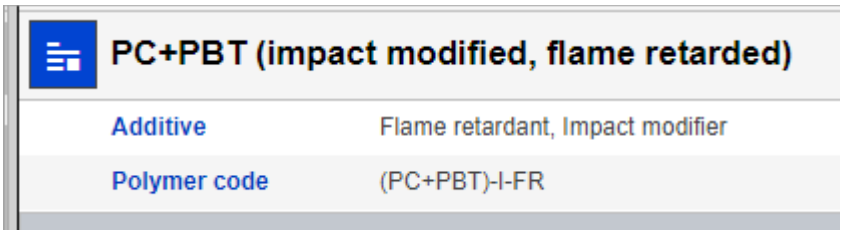
Use to provide a choice of one or more discrete text values. Values may be ordered or unordered, and there is no limit on the number of values in the set.

Examples from the MI:Training database:



Discrete Attributes which can have more than one value at the same time are known as **multi-value Discretes**; the set of permitted values is defined by the Discrete Type of the Attribute. For example:

- The MU Additive(s) Discrete Type in the MaterialUniverse database defines 7 possible additive values: None, Flame retardant, Impact modifier, Plasticizer/oil, Plasticizer/phthalate, Anti-friction/wear lubricant, UV stabilizer).
- The Additive Discrete in the MaterialUniverse table has the MU Additive(s) Discrete Type, and is configured to permit multiple values, allowing this polymer records to include more than one additive value:



To configure a multi-value Discrete Attribute, click on the Multiple Values tab and select the **Allow multiple values** option.

5.12.4 Text Attribute properties

Name

This may be up to 255 characters long. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Batch Number" and "Batch number").

Attribute can be used as a search filter

Determines whether or not the Attribute can be used as a search filter in MI:Explore and MI:Viewer.

Default Threshold Type

Specifies the default option offered when choosing the search criteria threshold for the Attribute on the **Advanced Search** page in MI:Viewer. For text Attributes, you can choose from: *contains*, *exists*, *does not exist*.

Help page (optional)

The location and name of a file containing help information about the Attribute/data ("Attribute Note"); when this is configured, MI:Viewer users can view the notes by clicking on the Attribute name in a datasheet in MI:Viewer.

Values must be unique (Short Text only)

Enforces a uniqueness constraint on Short Text Attributes that are used as record identifiers, to ensure that no duplicate values can be entered within the same Table, for example, where records are identified by a CAS Number value.

6 Record Link Groups

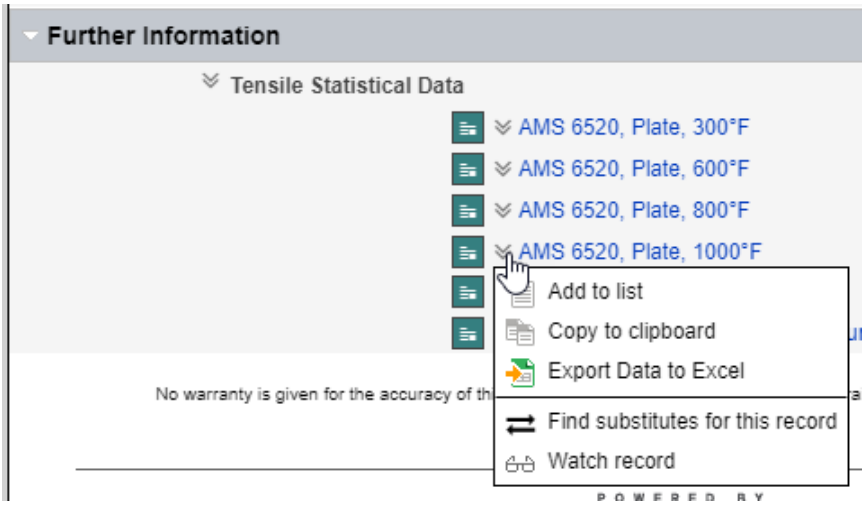
Related records in different tables can be linked using Record Link Groups. These are named collections of bi-directional links between tables, and they can be used to minimize data duplication/reduce data errors, and to help data maintenance.

Record Link Groups may include static or smart links:

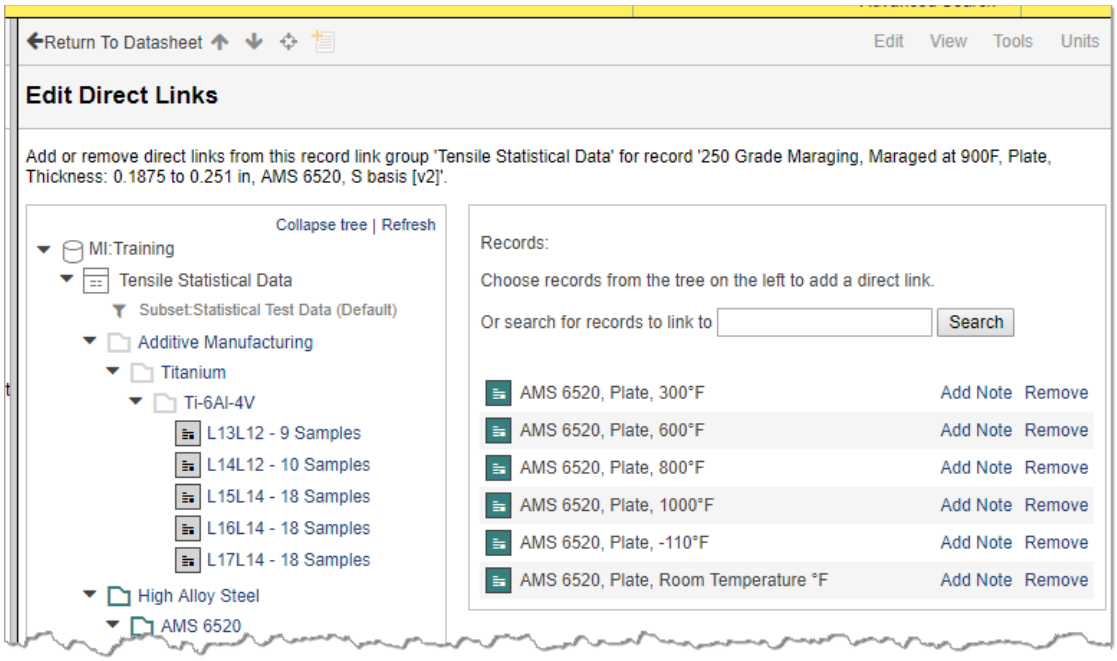
- ***Static links*** are direct, permanent links from one record to another record. The Record Link Group defines the source table and target table, and the records in the group are added or removed manually (in MI:Viewer) or during import (auto-links) or using MI:Toolbox (Automatic Link Creator).
- ***Smart links*** are links from one record to another record that are made automatically using defined rules (*Smart link criteria)*: when the value of key Attributes in the two records match (for example, a Batch number), the records are linked; if the values do not match, there is no link. Smart Link Groups can't be edited manually (that is, users can't add or remove records): records are added or removed automatically based on the current value of the relevant Attributes in the records.

When a Record Link Group is included in the relevant Layout(s), a list of the records it contains is shown in datasheets:

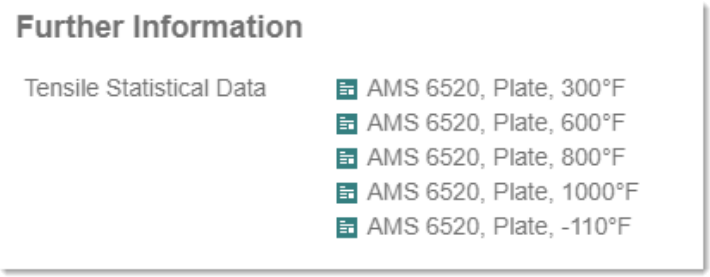
In MI:Viewer, the records in the Record Link Group appear as links on the datasheet; users can click on the link and view the related record, and they can also add some or all of the linked records to the Record List, copy them to the clipboard, export them to Excel, or add them to their watch list:



MI:Viewer users with sufficient privileges can edit Static Record Link Groups to add or remove records:



In MI:Explore, the records in the Record Link Group can be viewed on datasheets, but are not clickable, and the group cannot be edited to add or remove records:

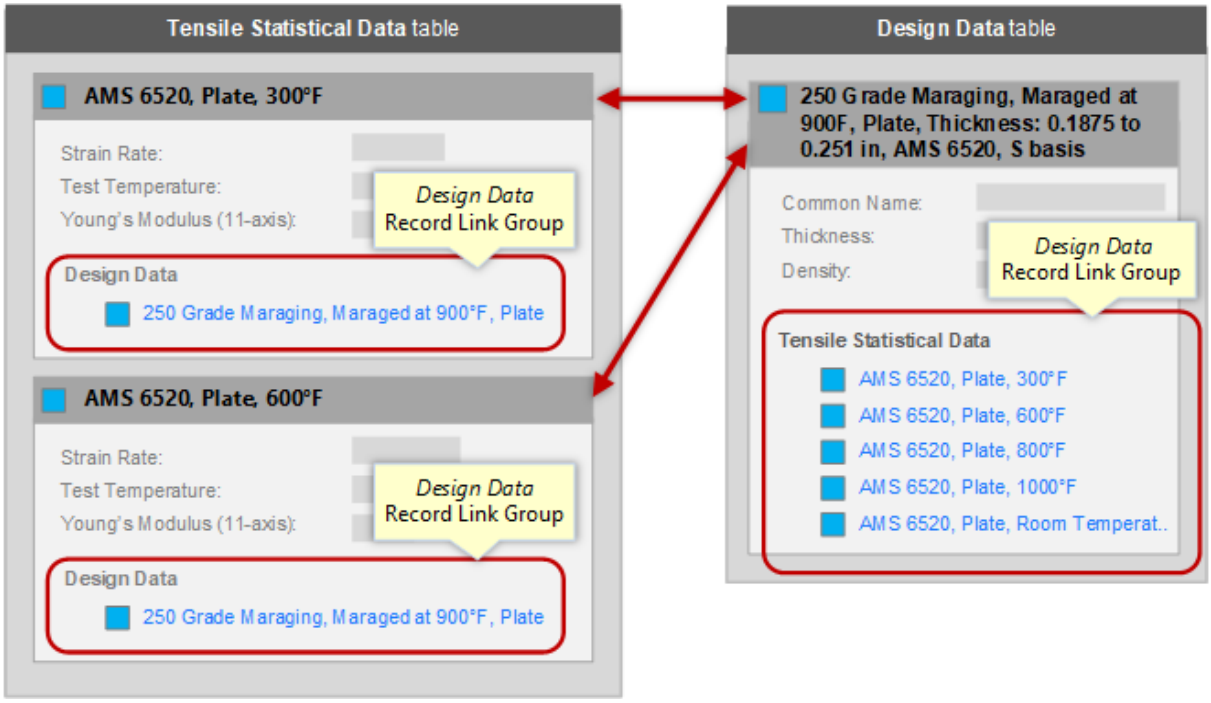


See also Links between tables for a comparison of the relative benefits of Static Links vs Smart Links.

6.1 Static links

Static links in GRANTA MI are fixed, permanent links between records in one table and records in another table, in the same database or in a different database.

For example, the *Design Data* Record Link Group shown below provides permanent links between test records and corresponding design data records:

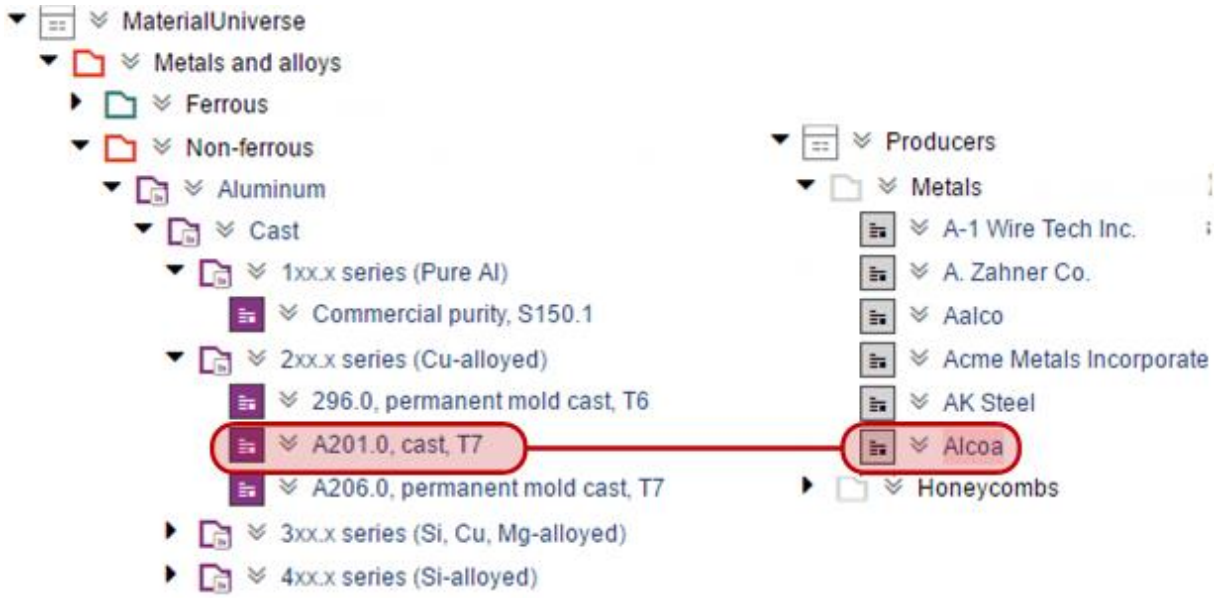


- The Record Link Group name is "Design Data".
- The source table is the *Tensile Statistical Data* table, and the destination table is the *Design Data* table.
- Links from the source table are labeled with the Record Link Group name, "Design Data".
- The reverse links to the source table are labeled "Tensile Statistical Data"; this label is specified in the Record Link Group definition.

Static links may be direct or indirect.

6.1.1 Direct links

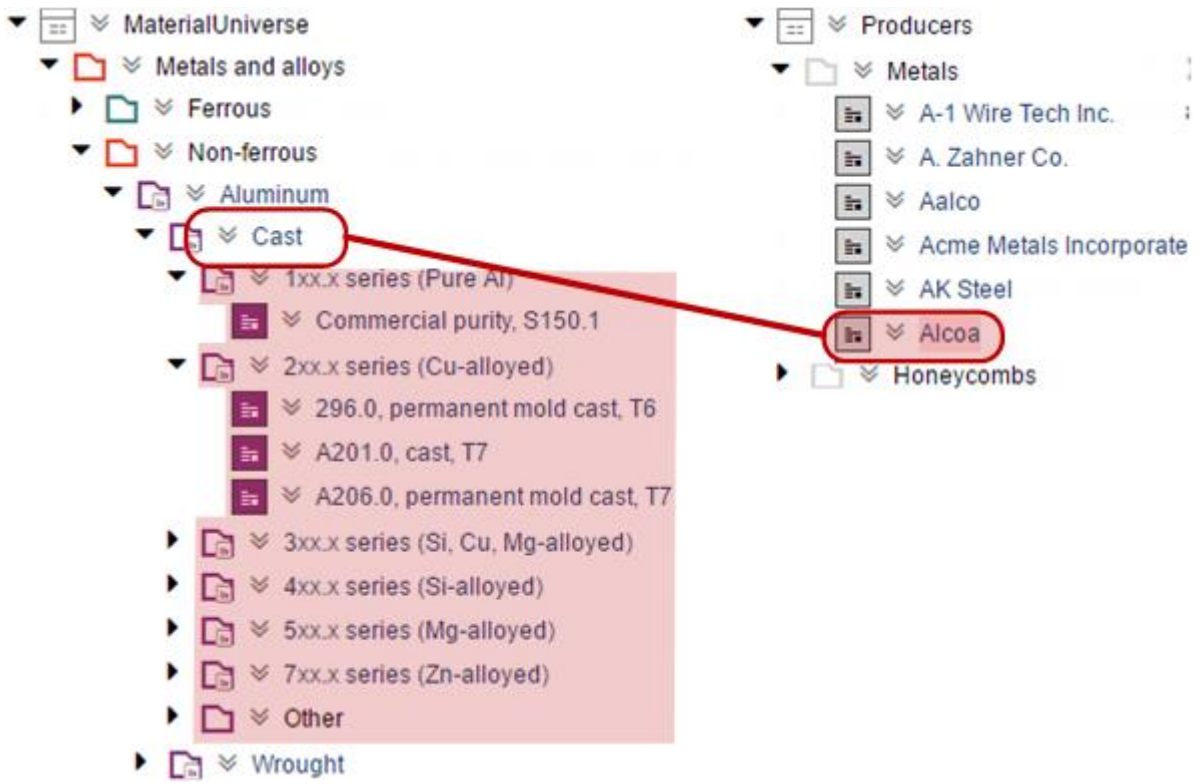
A **direct link** is a link from one record to another record. For example, the aluminum *Cast* record in the *MaterialUniverse* table below is linked directly to the *Alcoa* record in the *Producers* table;



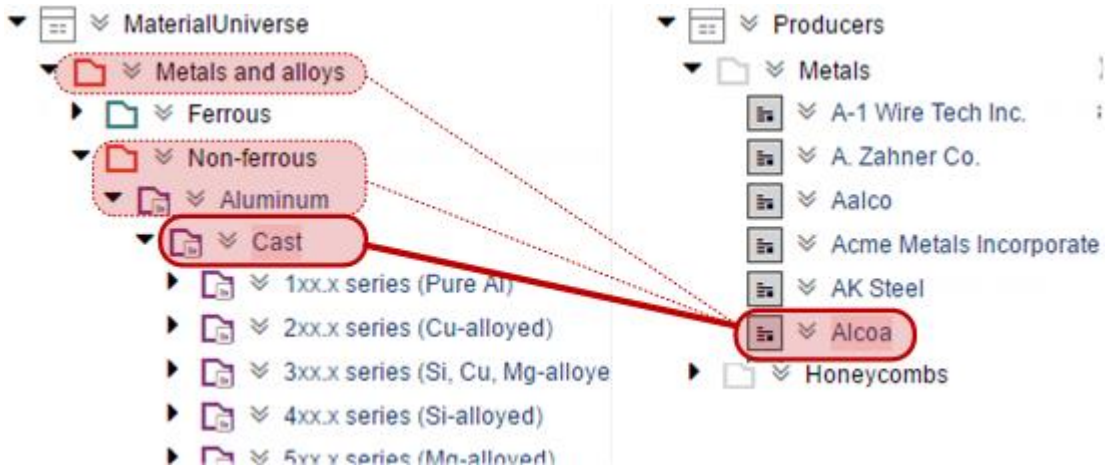
6.1.2 Indirect links

As well as record-to-record links, Static Record Link Groups can also include links to folders and generic records. In this case, all of the records inside the folder/generic are also linked: this type of implicit link is referred to as an ***indirect*** link.

For example, the aluminum *Cast* generic record in the *MaterialUniverse* table below may be linked to the *Alcoa* record in the *Producers* table, establishing indirect links from all of the records and folders within the *Cast* generic record to the *Alcoa* record; this is like saying "all Alcoa plants can supply all cast aluminum alloys":



The 'parents' of the linked records can also be considered as linked together. This is again a function of the hierarchical structure of the database. In this example, it is like saying that at least one metals supplier (Alcoa) can supply at least one aluminum (all cast aluminum alloys), and by implication at least one non-ferrous metal and by implication again, at least one metal/alloy.

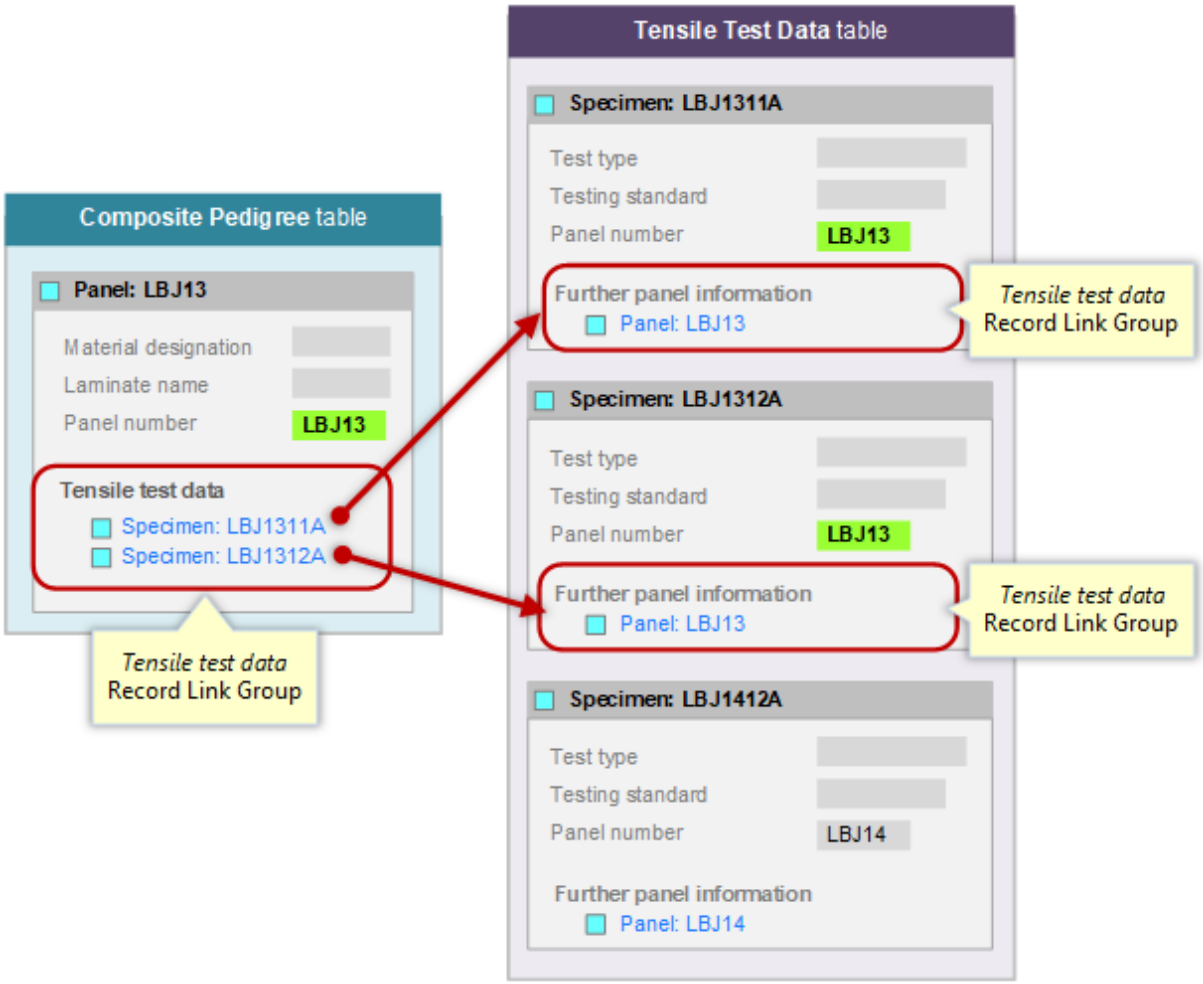


6.2 Smart links

Smart links can be used to provide dynamic, data-driven links to records in the same database. When the values of key Attributes in the source and target records match, a link exists between the two

record; if the values do not match, the records are not linked. Smart links are not stored in the database, but are created whenever the key Attribute data values match.

Example: this Record Link Group includes smart links between panel records and the relevant tensile test data records by matching the panel number in the records:



- The Record Link Group name is "Tensile test data".
- The source table is the *Composite Pedigree* table, and the destination table is the *Tensile Test Data* table.
- Links from the source table are labeled with the Record Link Group name, "Tensile test data".
- The reverse links to the source table are labeled "Further panel information"; this is specified in the Record Link Group definition in MI:Admin.
- The smart link criteria here is that the *Panel number* value must match in both tables; any records with the same *Panel number* value will be linked.

6.2.1 Smart link criteria

Smart link criteria determine the conditions that must be met for a link to exist, that is which Attribute values have to match. To define the criteria for smart links, you specify:

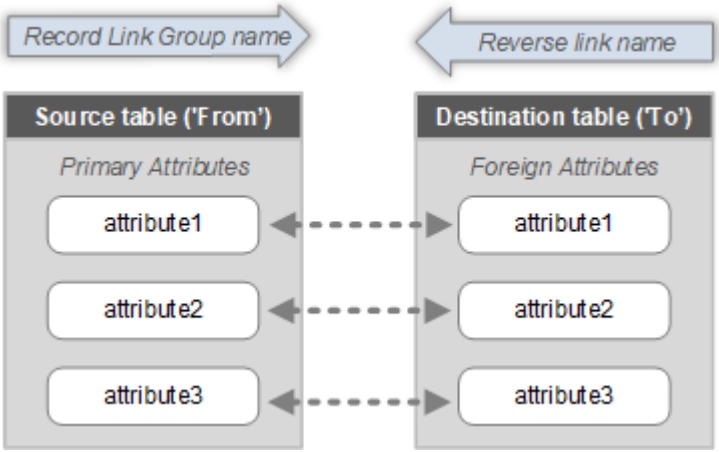
- Up to 3 Foreign Key-Primary Key pairings.
- The referential integrity relationship that will be enforced between the two records (One-to-One, One-to-Many, Many-to-Many).
- Whether or not orphan records will be permitted, that is, whether records may exist in the foreign table that do not link to any records in the primary table.

Primary and Foreign Attributes

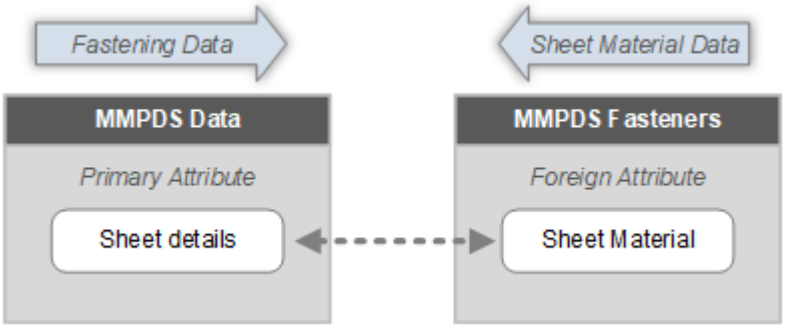
Key Attributes may have the following data type: Integer, Discrete, Short Text, Logical, and Date.

Up to three pairs of Attributes can be matched.

For each pair of Key Attributes, both of the Attributes must have the same data type.



Example: in this *Fastening Data* smart Record Link Group, records in the two tables will be linked when the *Sheet details* data value in the *Data* table matches the *Sheet Material* data value in the *Fasteners* table:



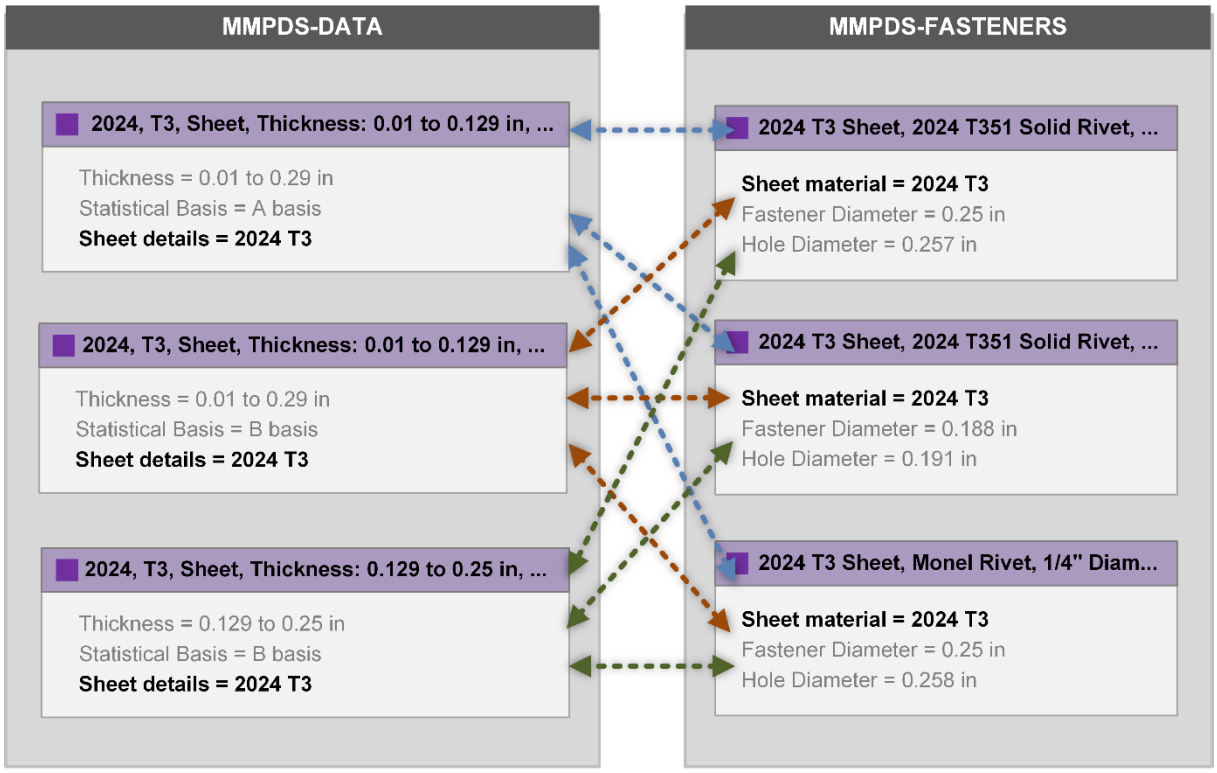
Referential integrity model

To ensure data integrity and data consistency, the referential integrity relationship between the linked records must be specified as part of the Smart Record Link Group definition.

In a **Many to Many** relationship:

- There may be one or more records in the primary table that fulfill the criteria for the primary Attribute.
- There may be one or more records in the foreign table that fulfill the criteria for the foreign Attribute.

For example: each sheet record in this *MMPDS-Data* table has multiple matching records in the *Fasteners* table, and each fastener record is linked to several sheet records.



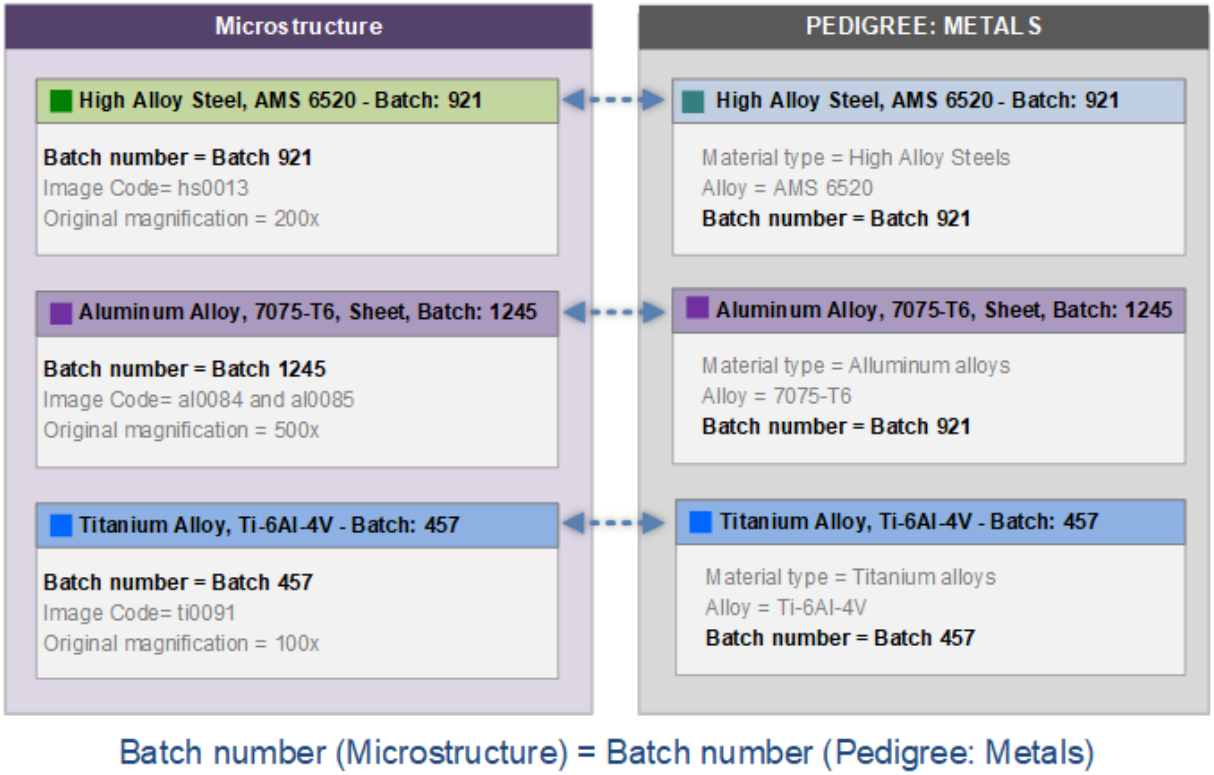
Sheet details (MMPDS-Data) = Sheet material (MMPDS-Fasteners)

In a **One to One** relationship:

- There can be only one record in the primary table that fulfills the criteria for the primary Attribute.
- There can be only one record in the foreign table that fulfills the criteria for the foreign Attribute.

For example: each record in this source *Microstructure* table can have only one matching record in the destination *Pedigree: Metals* table, and each record in the *Pedigree: Metals* table can have only one matching record in the *Microstructure* table. Any attempt to create a 2nd record in the same

table with the same batch number will fail, as the system will prevent violation of the one-to-one constraint.

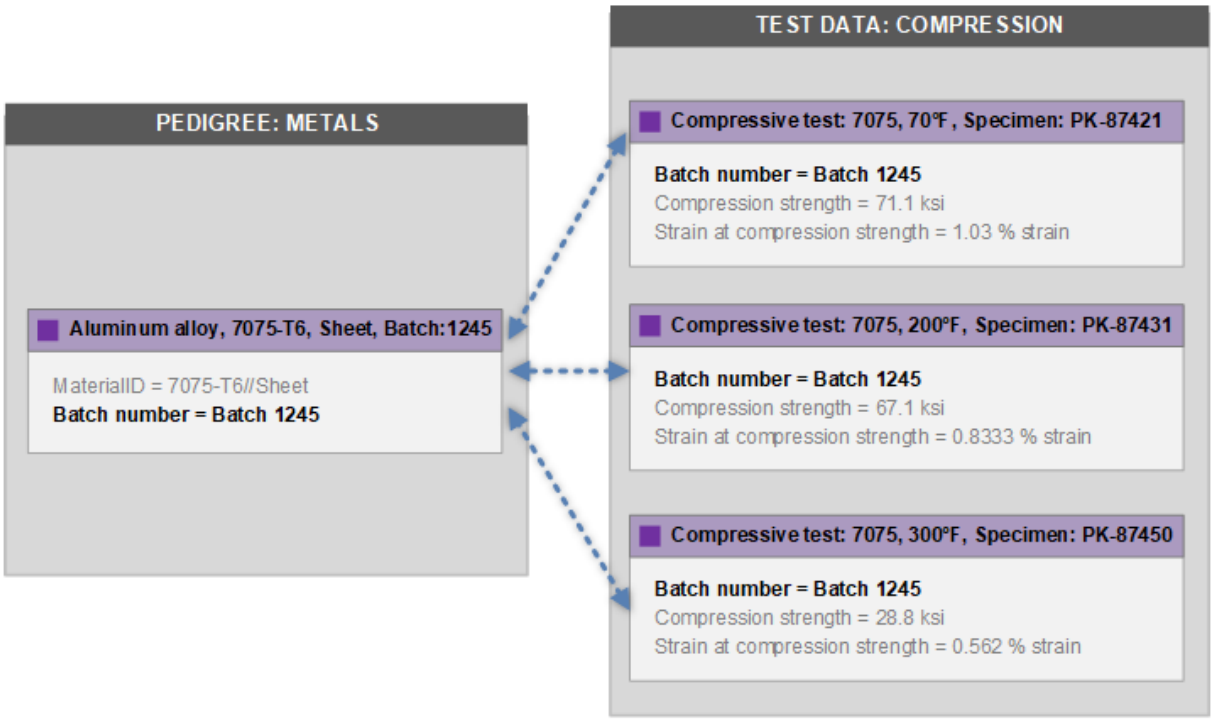


Batch number (Microstructure) = Batch number (Pedigree: Metals)

In a **One to Many** relationship:

- There can be only one record in the primary table that fulfills the criteria for the primary Attribute.
- There can be more than one record in the foreign table that fulfills the criteria for the foreign Attribute.

For example, each batch record in this source *Pedigree: Metals* table may have multiple matching test records in the *Test Data: Compression* table, and each test record may have only one matching record in the *Pedigree* table.



Batch number (Pedigree: Metals) = Batch number (Test Data: Compression)

Orphan records

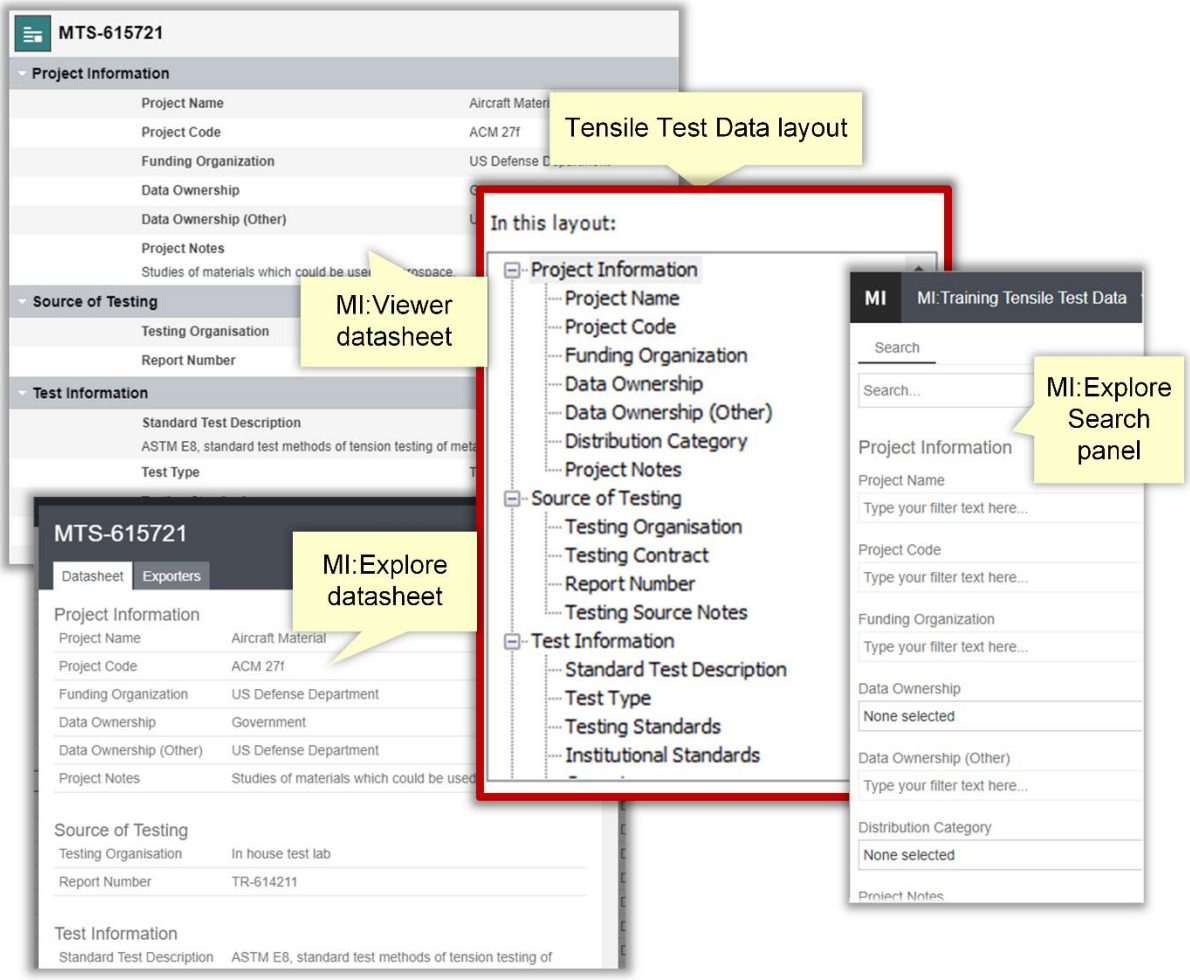
As part of the Smart Link Criteria for a Record Link Group with a 'One to One' or 'One to Many' referential integrity model, you can specify whether or not **orphan records** are permitted. An orphan record is a record in the foreign Table that is not linked to any record in the primary Table.

- **Orphans allowed**: records in the foreign Table do not need to be matched with a record in the primary Table.
- **Orphans not allowed**: all records in the foreign Table **must** be matched with a record in the primary Table.

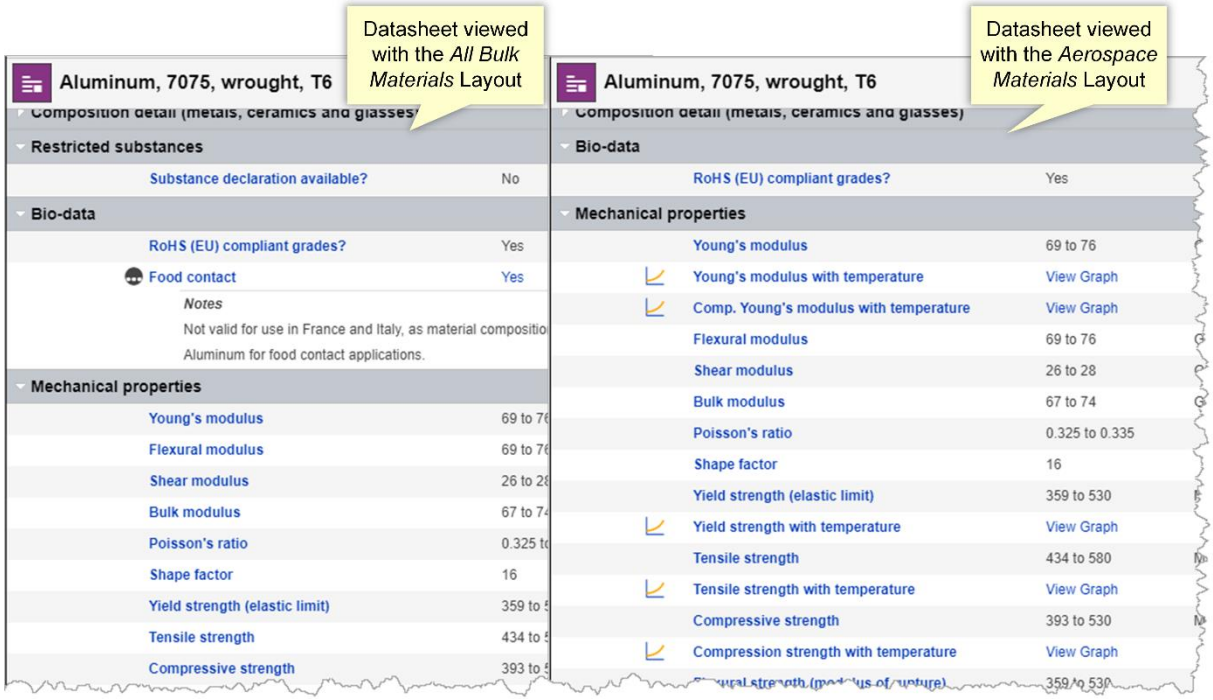
7 Layouts

Layouts provide different views of the data stored in a database to application users:

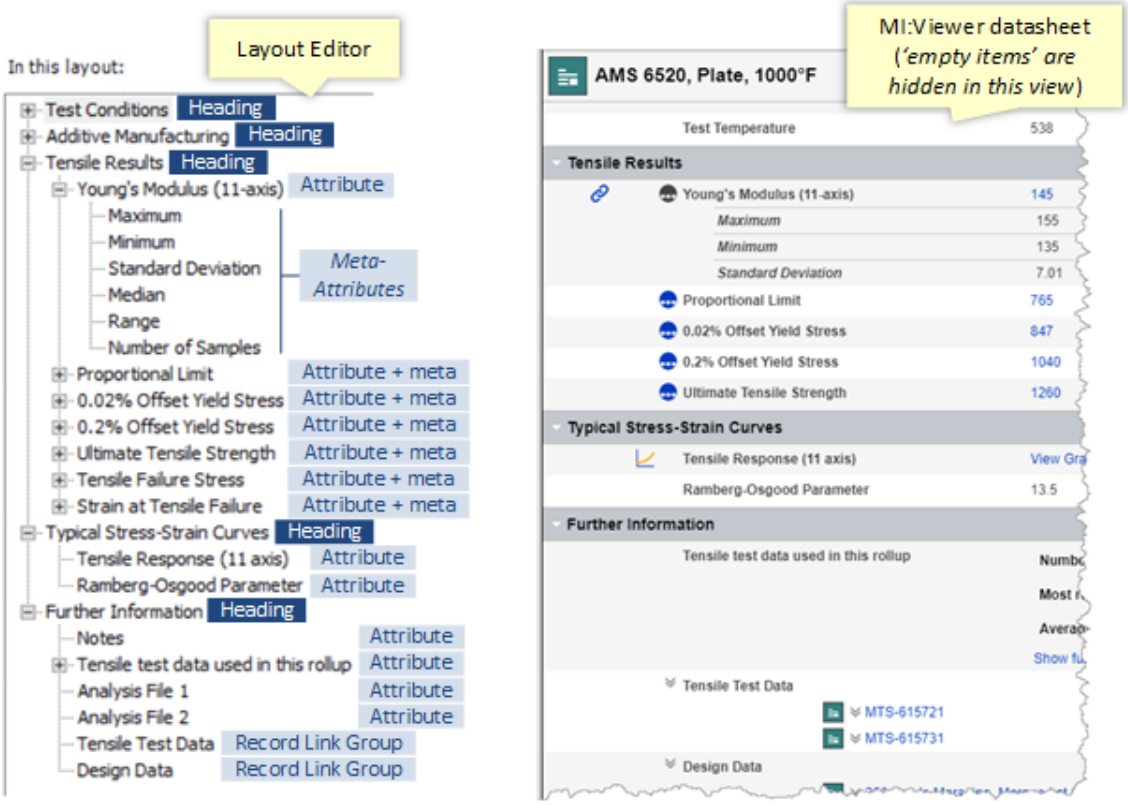
- In MI:Viewer and MI:Explore, the Layout defines which Attributes, Meta-Attributes, Tabular columns, record links, and headings appear on datasheets, and how they are all organized.
- In MI:Explore, they are also used to define the Attributes, Meta-Attributes, and headings that appear in the application Search pane and in the forms used to edit data.



Layouts are specific to one Table, and can be used to present different views of the data stored in that Table, showing different Attributes and different headings, in a different order. Attributes and Meta-Attributes that are not in a Layout will not be seen on the record datasheet when using that Layout. For example, the same record is viewed here in MI:Viewer using two different Layouts; you can see that the *Aerospace Materials* Layout includes functional data that is not shown when viewing the datasheet with the *All Bulk Materials* Layout:



In the MI:Admin Schema tool Layout Editor, you add Attributes, Meta-Attributes, Tabular Columns, Record Link Groups, and Associated Record links individually to the Layout, and organize them under headings. For example:



7.1 Meta-Attributes in Layouts

It is possible to include an Attribute in a Layout and all, some, or none of its Meta-Attributes. The visibility of Meta-Attributes in datasheets in MI:Explore and MI:Viewer depends on whether or not the Meta-Attribute is explicitly included in the relevant Layout, and, in MI:Viewer, also on the user's current datasheet view preferences.

7.1.1 Meta-Attribute visibility in MI:Viewer

On datasheets in MI:Viewer, Meta-attributes may be:

- Always visible
- Never visible
- Hidden, but showable
- Shown, but hideable

Attributes with hidden Meta-Attributes are flagged on datasheets in MI:Viewer with a Metadata symbol. To view hidden metadata, users can:

- Click on the Metadata symbol in front of the Attribute name to view the hidden data.
- Click **Expand all metadata** on the datasheet **View** menu to see all hidden metadata on the datasheet
- Switch to the Full datasheet view, where hidden metadata for all Attributes on the datasheet will be shown along with all hidden graphs, Tabular data, discrete functional data, and data links.

The default display behavior for metadata in MI:Viewer datasheets is determined by a combination of:

- Whether the Meta-attribute is included in the Layout being used for the datasheet
- Whether any metadata currently exists (i.e. is the Meta-Attribute populated or empty?)
- The user's current datasheet view options (selected on the datasheet **View** menu) :
  - Viewing the Summary datasheet or the Full datasheet
  - Viewing Empty items (i.e. unpopulated Attributes/Meta-Attributes), or not

In summary:

Meta-attribute is	Empty items hidden		Empty items shown	
	Summary datasheet	*Full datasheet*	*Summary datasheet*	*Full datasheet*
✓ In the layout ✓ Has data	Always visible	Always visible	Always visible	Always visible
✓ In the layout ✗ Has data	Never visible	Never visible	Always visible	Always visible
✗ In the layout ✓ Has data	Hidden	Shown	Hidden	Shown
✗ In the layout ✗ Has data	Never visible	Never visible	Hidden	Hidden

7.1.2 Meta-Attribute visibility in MI:Explore

In MI:Explore, there is no end-user show/hide capability, and Meta-Attributes are either included in datasheets, or not included, depending on:

- Whether the Meta-attribute is included in the Layout used in the data view.
- Whether any metadata currently exists (i.e. is the Meta-Attribute populated?) AND whether the data view is configured to show or hide Attributes/Meta-Attributes with no data (via **showBlanks** data view property).

In summary:

Meta-attribute is...	showBlanks = false	showBlanks = true
✓ In the layout ✓ Has data	In datasheet	In datasheet
✓ In the layout * Has data	Not in datasheet	In datasheet
* In the layout ✓ Has data	Not in datasheet	Not in datasheet
* In the layout * Has data	Not in datasheet	Not in datasheet

7.1.3 Including Meta-attributes in Layouts, or not

Meta-Attributes must be explicitly included in Layouts intended for use in MI:Explore if you want them to appear in datasheets and/or the search panel.

In Layouts intended for use in MI:Viewer:

- Include a Meta-Attribute in the Layout if the metadata is important and you want to ensure that users will always see it alongside the relevant data, without having to click to view it. Examples: information about calculations used to provide the data, or notes on usage.
- Leave a Meta-Attribute out of the Layout if the metadata simply provides some supplementary information or notes; this means it will be hidden by default on datasheets, saving screen space, but users can still view it if they want to see it.

Note that only Meta-attributes that are included in the Layout will be exported when using the data export options in MI:Viewer and MI:Explore.

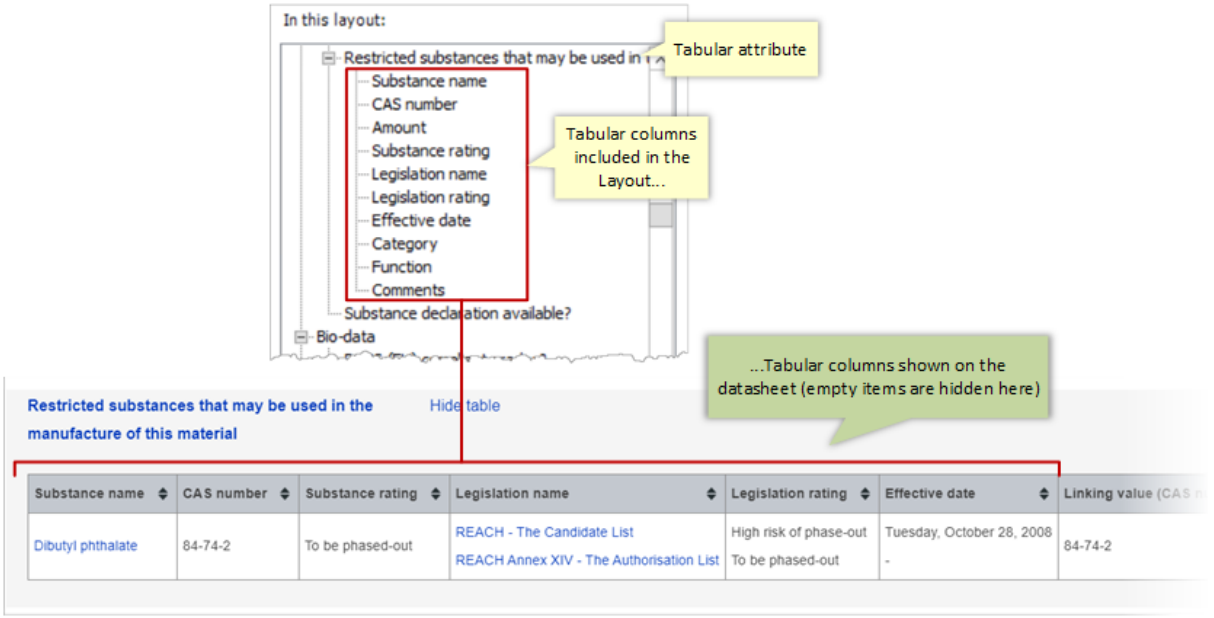
7.2 Tabular Attributes in Layouts

For Tabular Attributes, the tabular columns that appear in datasheets, and the order in which they appear, are specified in the Layout.

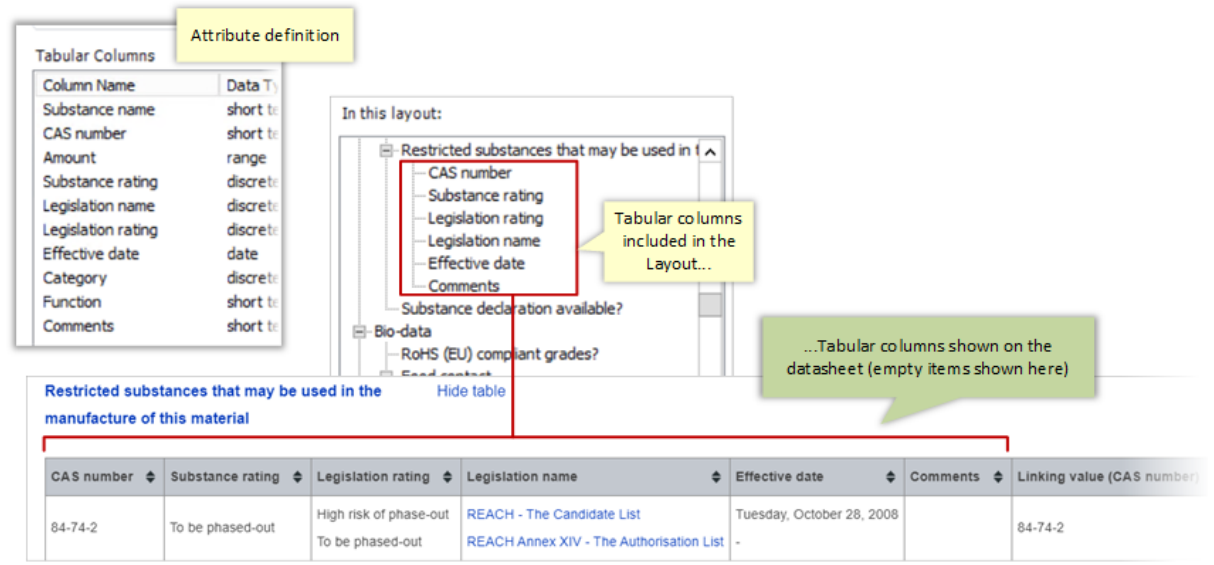
All or only some of the tabular columns defined in the Attribute may be included, and the columns in the Layout can be in any order (they do not have to be in the same order as in the Attribute

definition. Note also that, if a Tabular Attribute is edited to add or to re-arrange the tabular columns, any Layouts that include the Attribute must be updated to reflect this.

When you add a Tabular Attribute to a Layout, each tabular column must be individually added to the Layout, for example:



You can add some or all tabular columns, and order them as you prefer. For example, here's the same Tabular Attribute and data shown above, with a different Layout; note that only some of the tabular columns defined in the Attribute are included in the Layout, and they are in a different order from the Attribute definition:



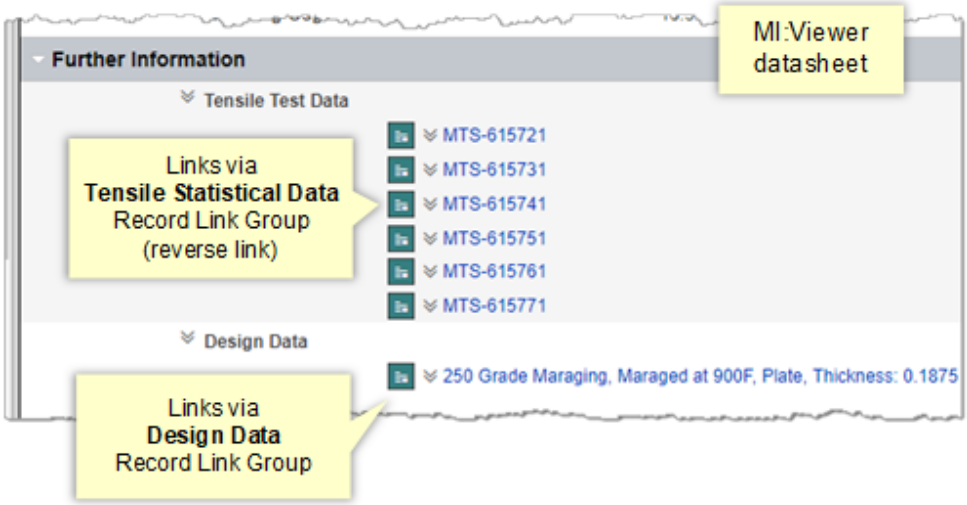
7.3 Links to other records

Layouts can include Record Link Groups and Associated Records as well as Attributes, allowing links to relevant records in other tables to be included in datasheets.

7.3.1 Record Link Groups

Including a **Record Link Group** in a Layout adds a heading on the datasheet under which linked records are listed. For example, the Layout used for the record in the *Tensile Statistical Data* table below includes two Record Link Groups that provide links to related records in other tables:

- Links to records in the *Tensile Test Data* table are provided by the *Tensile Statistical Data* Record Link Group (via the reverse link *Tensile Test Data*).
- The link to the record in the *Design Data* table is provided by the *Design Data* Record Link Group.

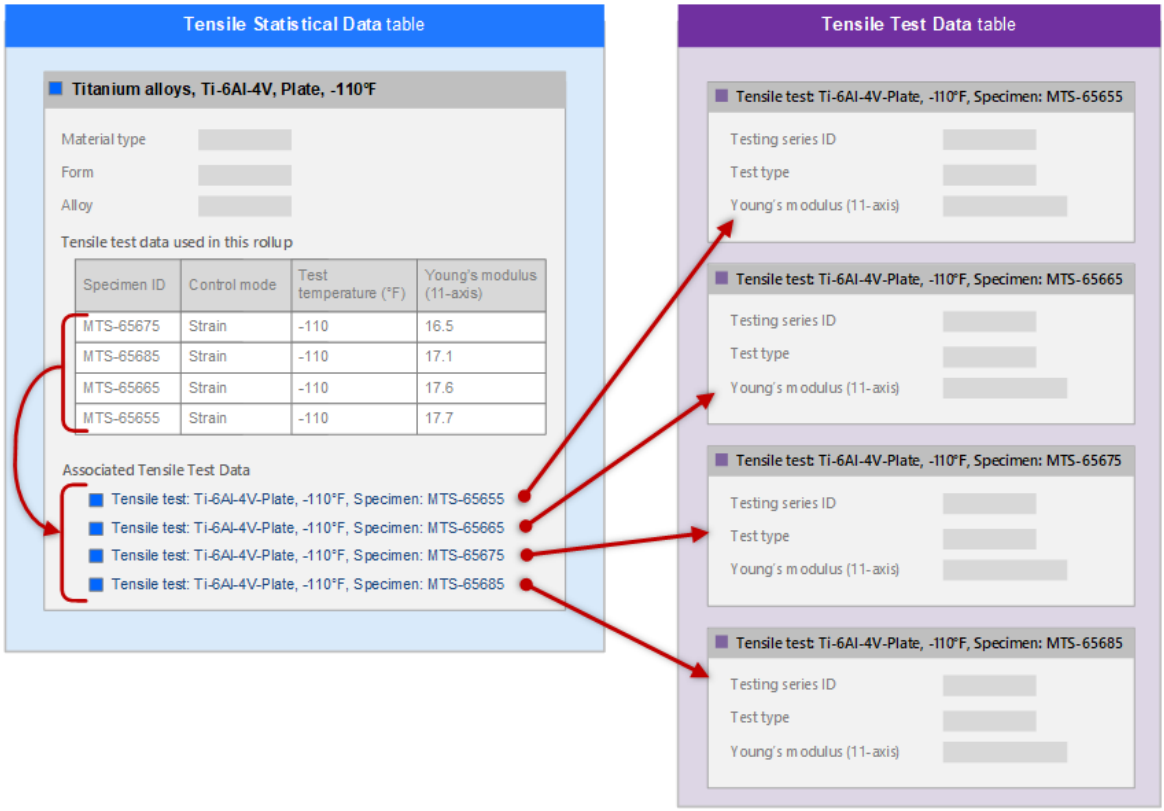


7.3.2 Associated Record links

Associated Records in a Layout allow datasheets in MI:Viewer to include lists of records that are linked via a Tabular Attribute in the record. The linked records are presented on datasheets in the

same way as records in a Record Link Group, under a heading defined in the Layout. Note that Associated Record links are not show in MI:Explore datasheets.

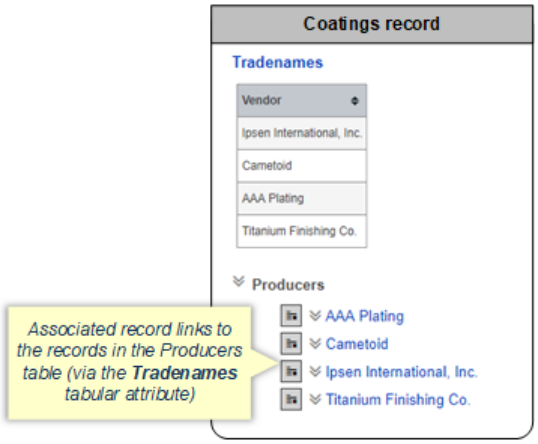
For example, the Layout used for the *Tensile Statistical Data* record below includes Associated Record links under the heading *Associated Tensile Test Data*, with links to relevant records in the *Tensile Test Data* table via the *Tensile test data used in this rollup* Tabular Attribute:



When adding Associated Record items to a Layout, the relationship between the source record and the associated records may be expressed as a "From" link or a "To" link.

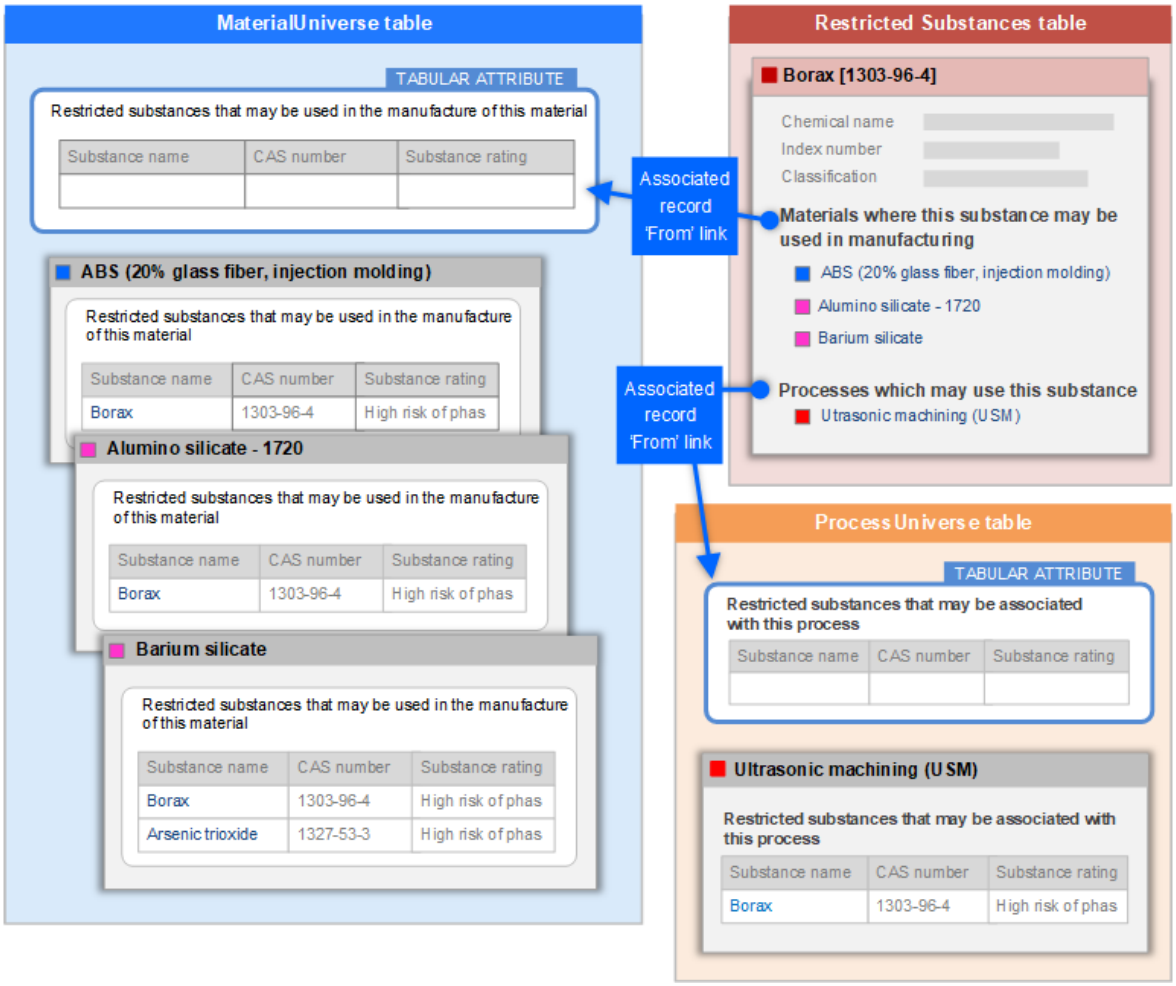
'To' links

The source table contains a Tabular Attribute that references records in the linked table:



'From' links

The linked table contains a Tabular Attribute that references records in the source table:

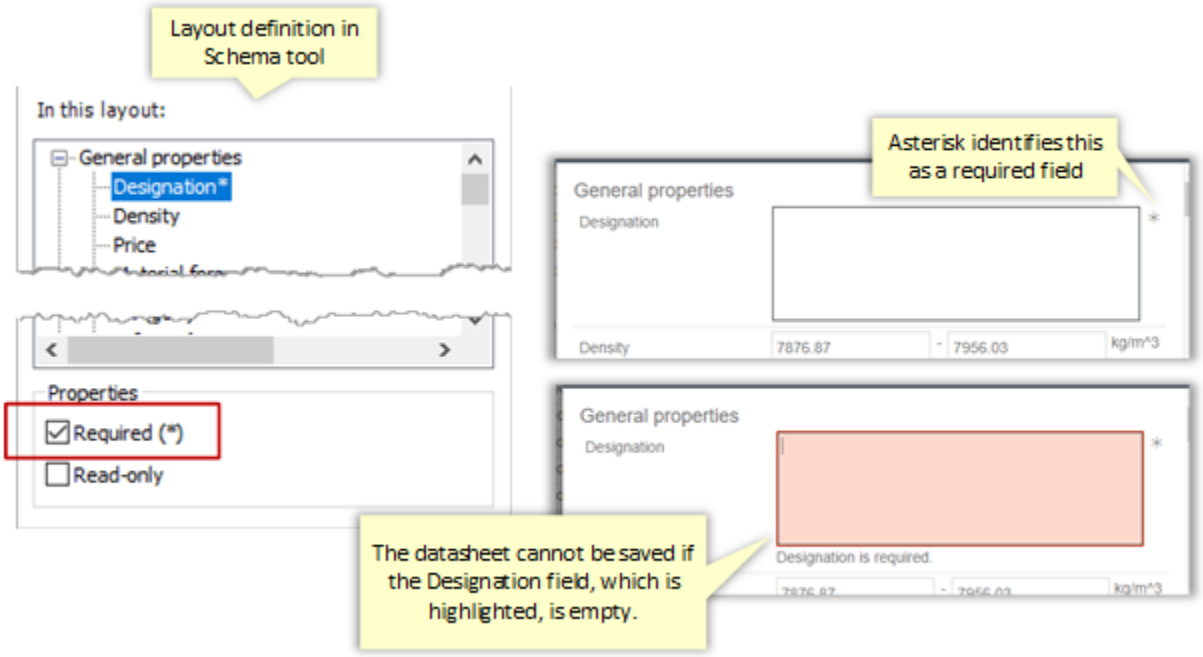


7.4 'Required' flag on Attributes in a Layout

Flagging an Attribute as *Required* in a Layout ensures that, when that Layout is used, users will not be able to save or submit changes to datasheets unless that Attribute has some data.

Note that this is a Layout property and not a property of the Attribute itself; the same Attribute may be marked as *Required* in one Layout, and not in another. If you want an Attribute to always be required, it must be flagged as *Required* in every Layout.

In this example from MI:Explore, *Designation* is required; it is identified by an asterisk, and the field is highlighted if the user attempts to save the datasheet without entering some data:



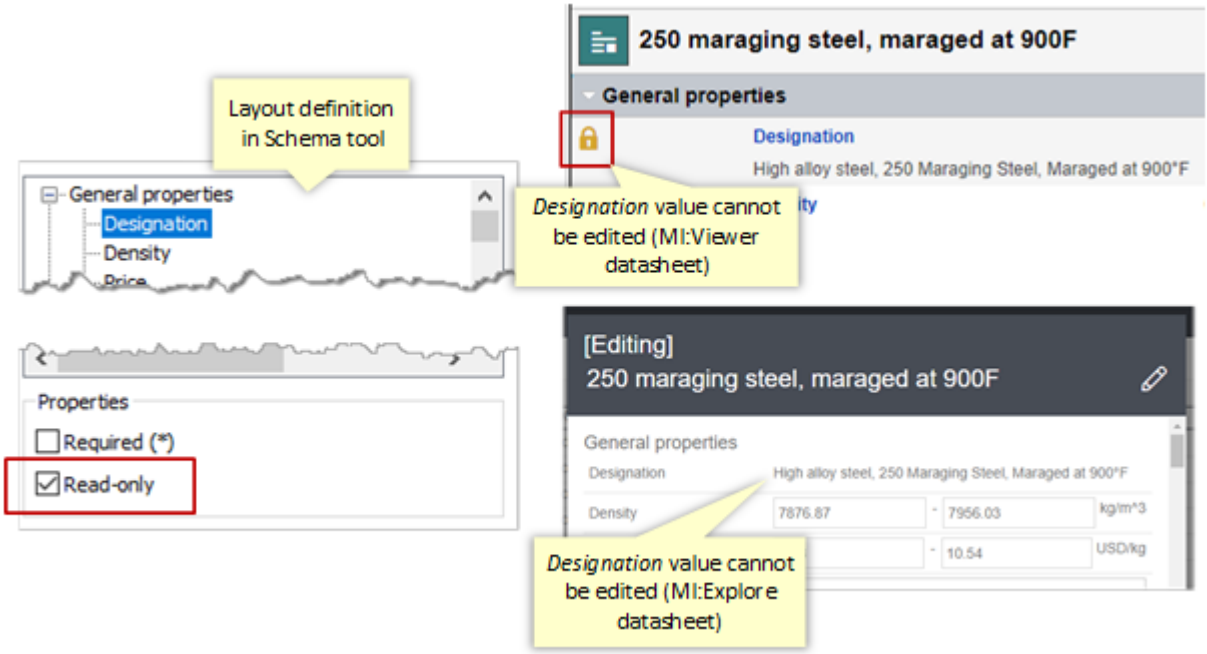
7.5 'Read-only' flag on Attributes in a Layout

Flagging an Attribute as *Read-only* in a Layout ensures that, when that Layout is used, users will not be able to edit the data for that Attribute.

Note that this is a Layout property and not a property of the Attribute itself; the same Attribute may be marked as *Read-only* in one Layout, and not in another. To prevent users from ever being able to edit the Attribute value, you will need to flag it as *Read-only* in every Layout.

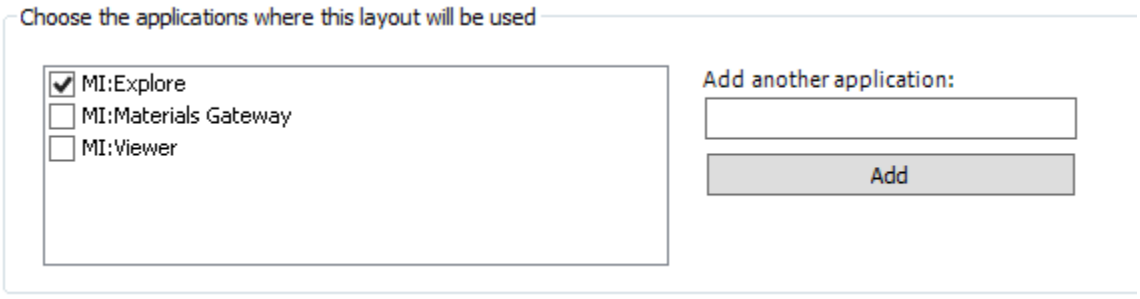
However, note also that MI:Viewer users can bypass the *Read-only* and *Required* properties set in Layouts by viewing datasheets without any Layout selected (**View > Change Layout > No Layout**); in this case, the Attribute would be editable, despite being flagged as Read-only in all Layouts. (To systematically prevent users from editing data in specific Attributes, use Permissions-based Access Control; see the *GRANTA MI Access Control and Security Guide* for details.)

In MI:Viewer, Read-only Attributes are identified in datasheets with a padlock icon; in MI:Explore, they are simply not editable. For example:



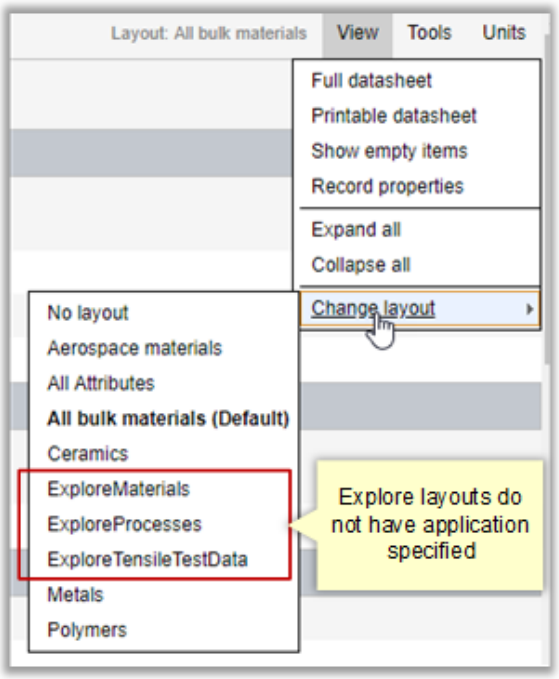
7.6 Layout target applications

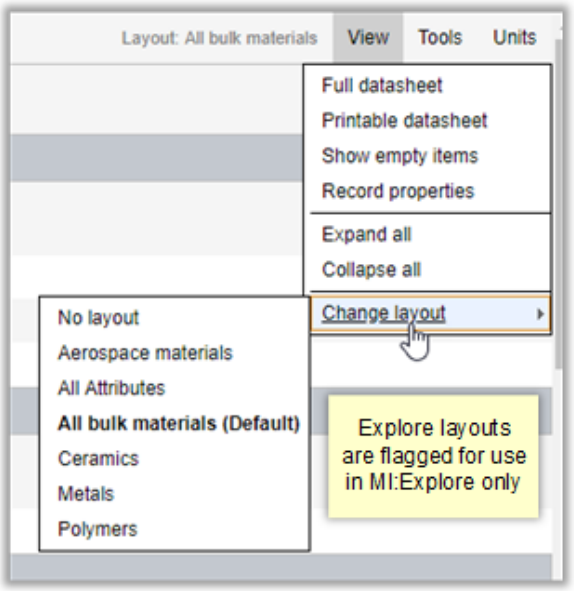
The applications in which a Layout is intended to be used can be specified as a property of the Layout. This can be used to ensure that only relevant Layouts are available within each application, for example, so users in one application don't see Layouts developed for use in other applications.



You can select any number of applications. If no application is selected, the Layout will be available in all applications. By default, the MI:Viewer, MI:Explore, and MI:Materials Gateway applications are listed; for applications developed using the MI SDK, you can enter the application name and click **Add** to add it to the list, then tick the check box.

For example, there are 3 Layouts here that have been developed for use in the MI:Explore application: *ExploreMaterials*, *ExploreProcesses*, and *ExploreTensileTestData*. When these 3 Layouts have no application flag set, MI:Viewer users can see and select then in the **Change layout** list in MI:Viewer. When these Layouts are flagged as intended for use in MI:Explore, they are not shown in MI:Viewer:





7.7 Access control details flags

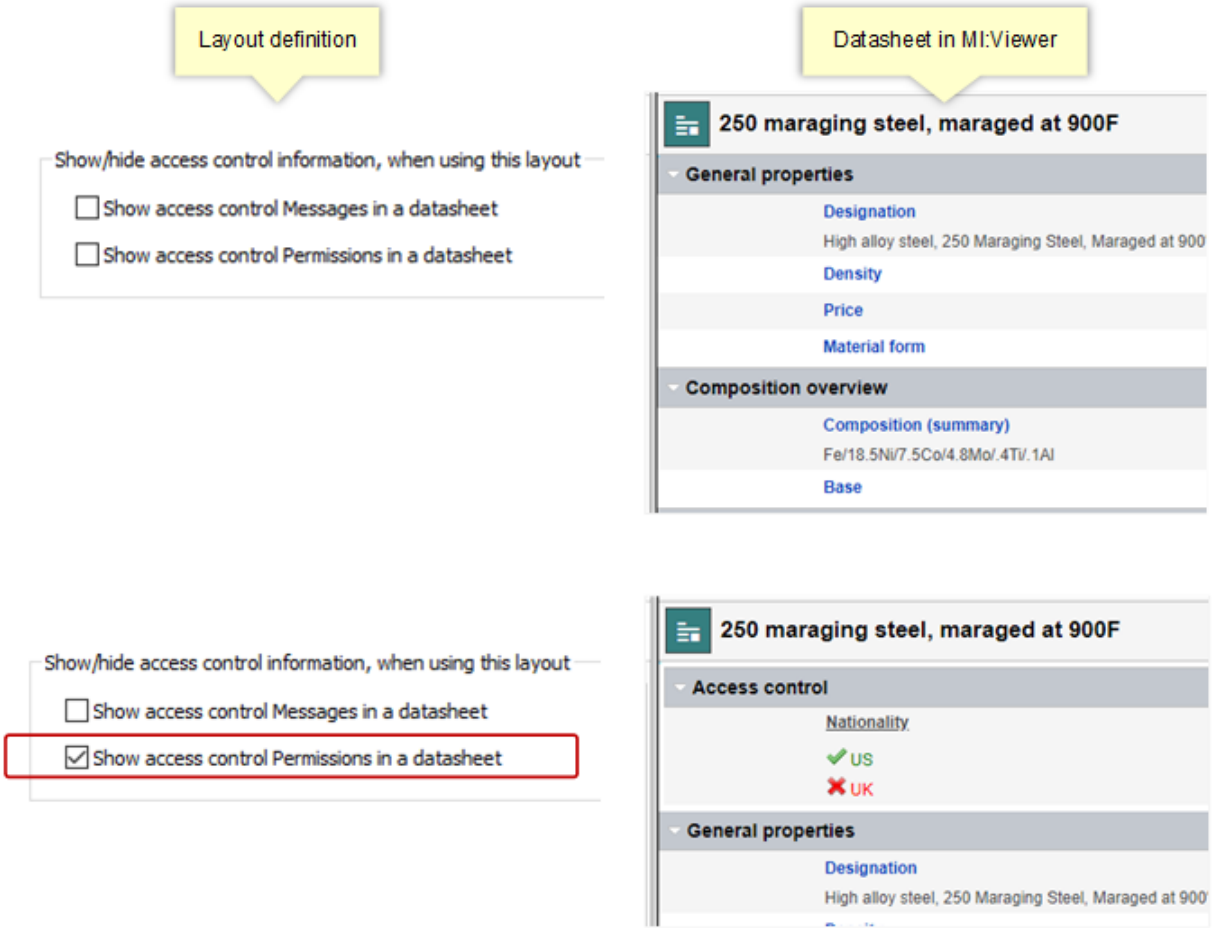
In databases with permission-based access control implemented, information about the access control permissions currently set on a record can be shown on datasheets in MI:Viewer via Layout properties.

7.7.1 Show access control permissions

The Layout property **Show access control Permissions in a datasheet** allows access control permissions currently set on a record to be displayed be displayed at the top of datasheets in MI:Viewer.

- An **Access control** heading will appear at the top of datasheets
- Permissions set on the record will be shown below it.

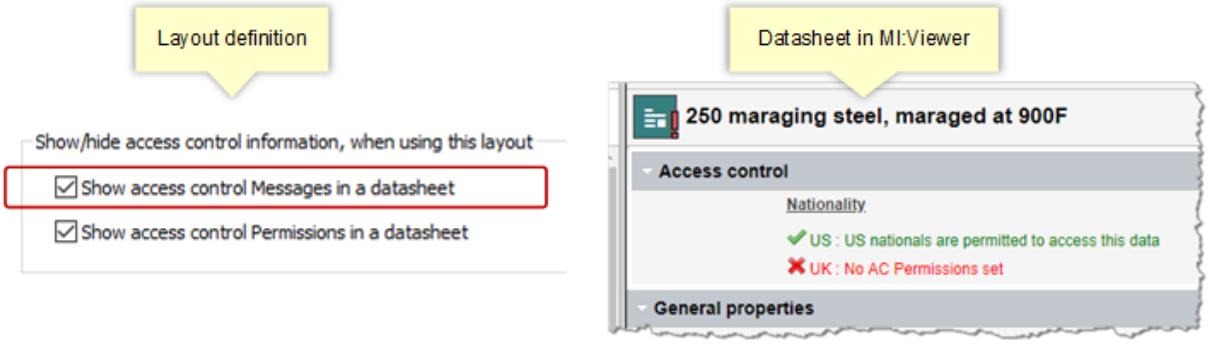
For example:



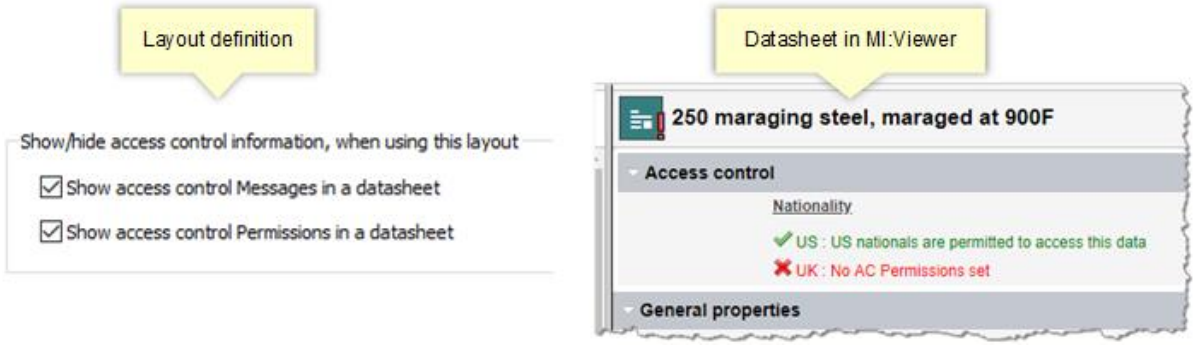
7.7.2 Show access control messages

The Layout property **Show access control Messages in a datasheet** allows access control "Set Message" and "Unset Message" strings defined in the Access Control Schema to be displayed at the top of datasheets in MI:Viewer.

For example:



For example:



7.8 Managing Layouts

To view and manage the Layouts in a table: in the MI:Admin Schema tool, select the database, double-click on the table, then click **Layouts**

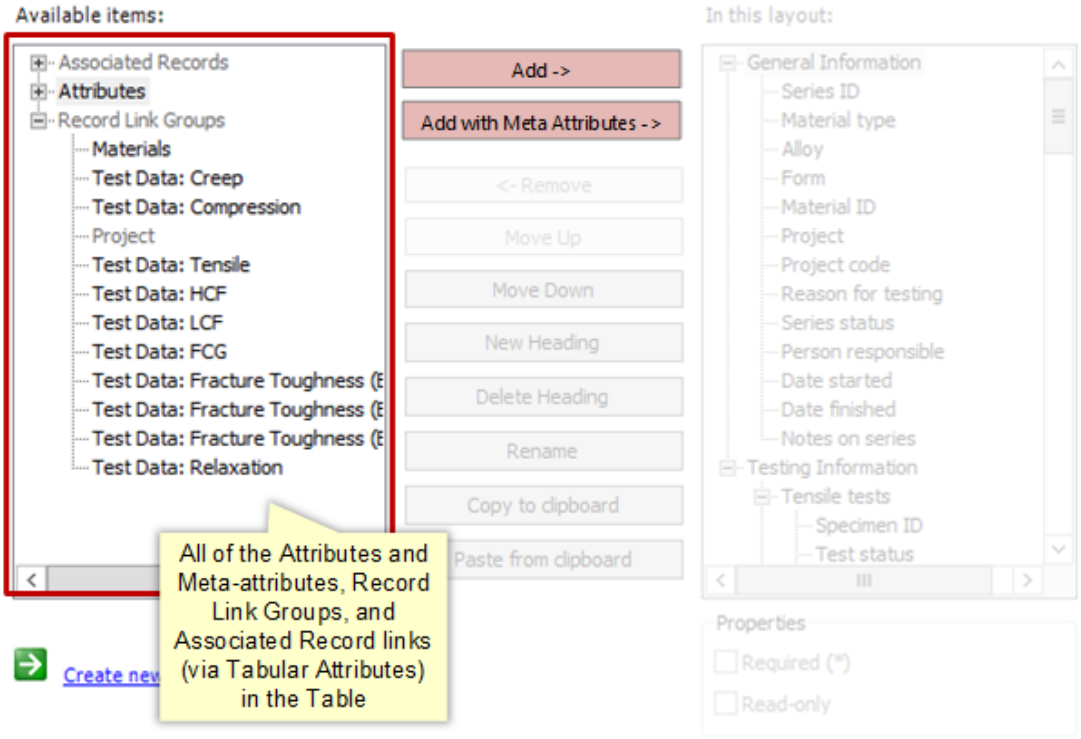
To create a new Layout, click **Add** or copy and then edit an existing Layout (**Copy** and **Paste** on the shortcut menu, or CTRL+C/CTRL+V) or paste a Layout definition created in Excel into the Layouts list from the clipboard.

7.8.1 Layout Name

This may be up to 255 characters long. The name entered here must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Polymers" and "polymers").

7.8.2 Available items list

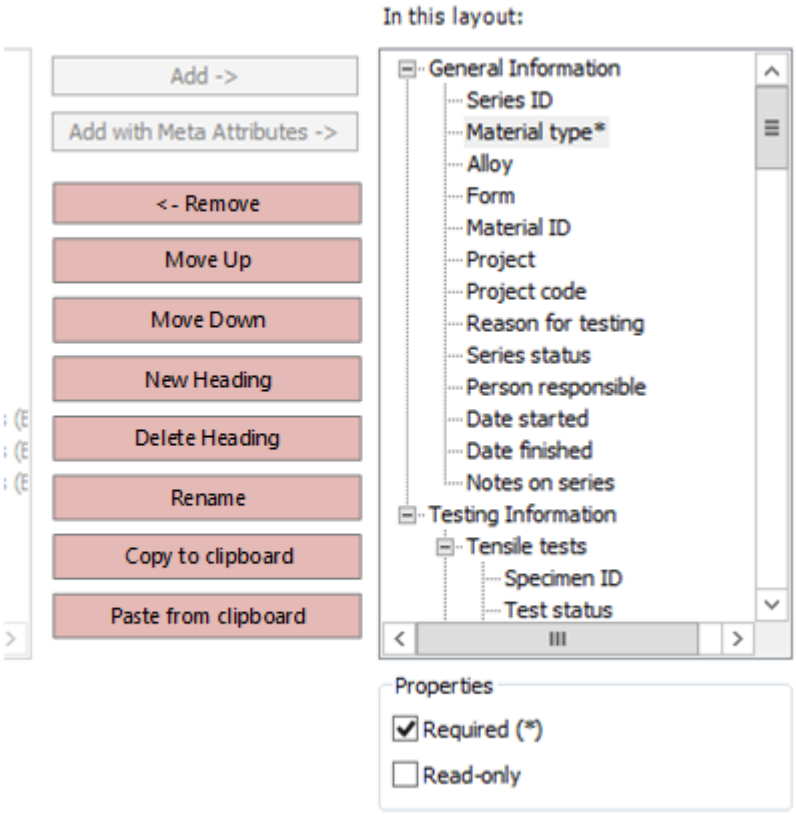
The **Available items** list shows all of the Attributes and Meta-Attributes, Record Link Groups, and Associated Record links available in the selected table that are not currently in the Layout.



Double-click an item in the **Available items** list to add it to the **In this Layout** list, or select it and use the **Add** buttons; ***Add with Meta Attributes->*** adds an Attribute and ***all*** of its Meta-attributes. See Meta-Attributes in Layouts for information about how metadata is displayed on datasheets in MI:Viewer and MI:Explore.

7.8.3 In this layout list

The **In this layout** list shows the Headings, Attributes, Meta-Attributes, Record Link Groups, and Associated Record links currently in the Layout. For example:



Click **New Heading** / **Delete Heading** to insert or remove Layout headings.

Use the **Move Up** and **Move Down** buttons to change the order of the headings and the items under them.

Use the **Copy to/Paste from clipboard** buttons to export or import a Layout via the clipboard; see Copying and pasting Layout definitions below.

Attribute Properties

Select an Attribute in the list and use the *Required* and *Read-only* check boxes to set these properties, if required. See 'Required' flag on Attributes in Layouts, 'Read-only' flag on Attributes in Layouts.

7.8.4 Show/hide access control information

Include permission-based access control details on datasheets in MI:Viewer, if required; see Access control details flags.

7.8.5 Applications where this Layout will be used

Specify which applications the Layout is intended to be used with. See Layout target applications for details.

- You can select any number of applications. If no application is selected, the Layout will be available in **all** applications.
- By default, the MI:Viewer, MI:Explore, and MI:Materials Gateway applications are listed; for applications developed using the MI SDK, enter the application name and click **Add** to add it to the list, then tick the check box.

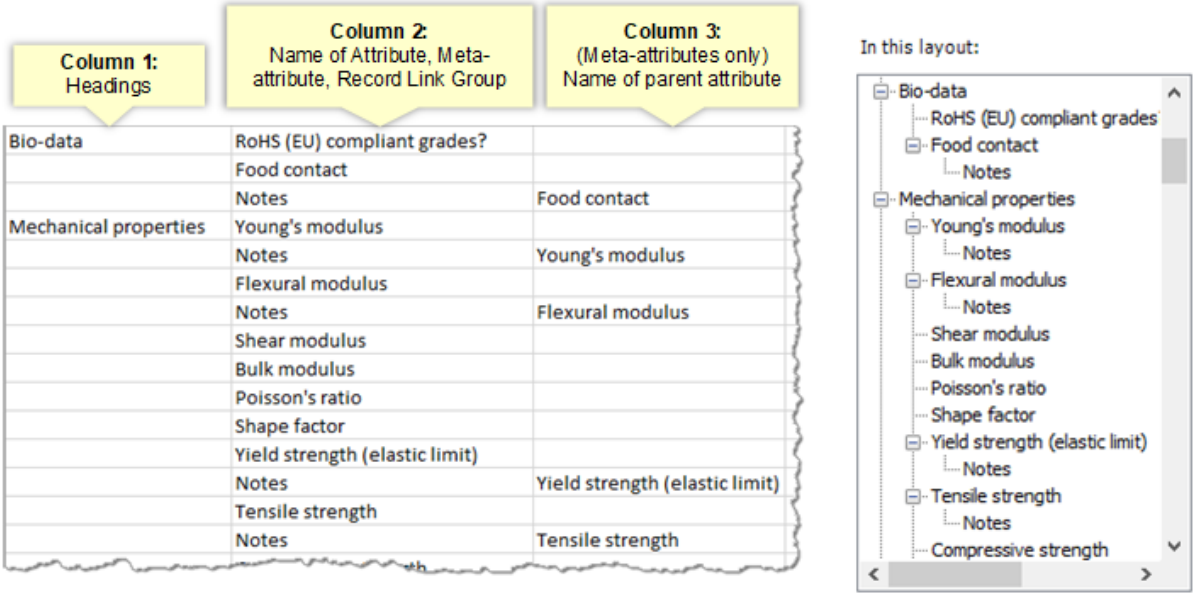
7.8.6 Copying and pasting Layout definitions from the clipboard

Layout definitions can be copied into and out of MI:Admin via the clipboard, allowing you to document your database schema, as well as define and modify it outside of MI:Admin, if needed. Layout headings, Attributes, Meta-attributes, and static and dynamic Record Link Groups may all be copied.

To copy a Layout definition from the Schema tool to the clipboard: click **Copy to clipboard** in the **Create a new Layout** or **Edit Layout** page. The Layout data is copied to the clipboard, and can then be pasted into a spreadsheet or a text file. Note that if there is one Tabular Attribute included in the Layout, then all tabular columns are exported to the clipboard. Because of this, if you copy an exported Layout which includes only some of the tabular columns back into MI:Admin, you will end up with a Layout that includes all of the original Tabular columns.

To paste a Layout definition into the Schema tool from the clipboard: select the data in the worksheet or text file and copy it to the clipboard, then click **Paste from clipboard** on the **Create a new Layout** or **Edit Layout** page.

Each Attribute to be included in the Layout must on a separate row. In a text editor, the columns should be tab-separated. For example: this Microsoft Excel worksheet defines the Layout items shown on the right when pasted into the Layout Editor.

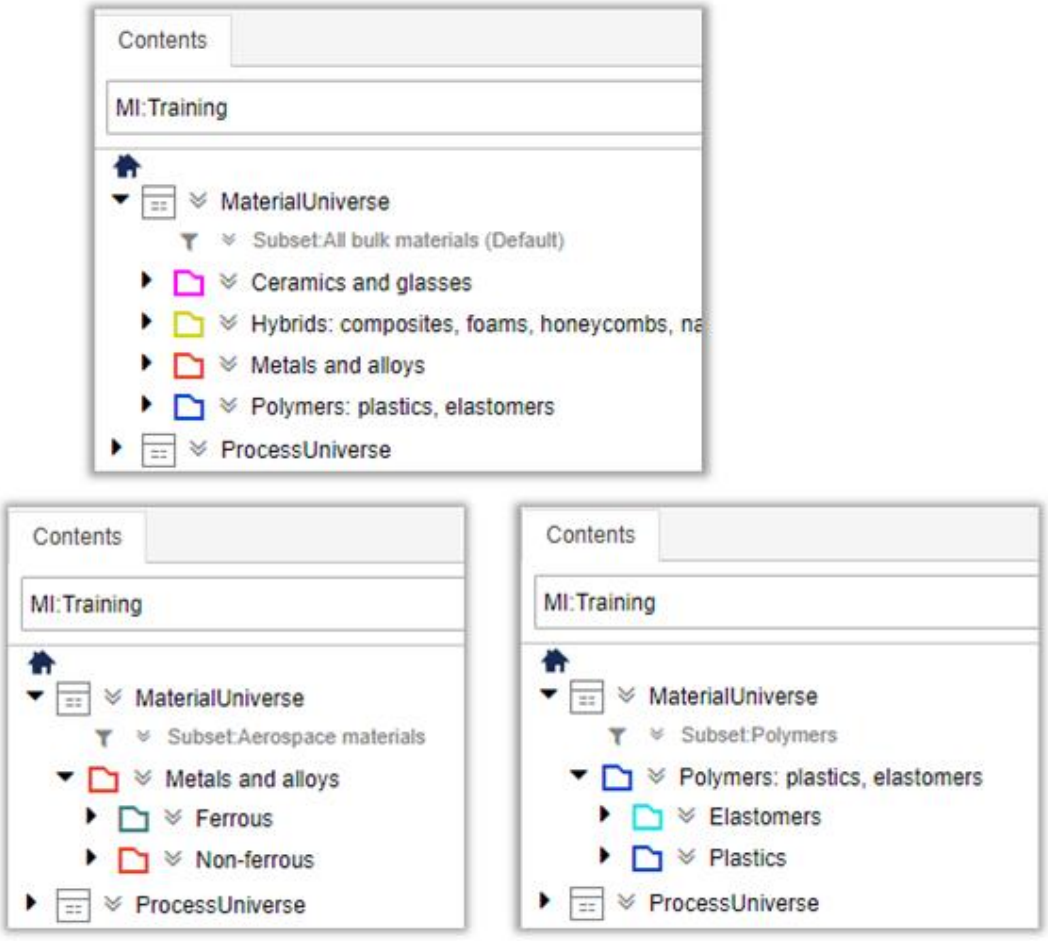


8 Subsets

Subsets are used in GRANTA MI to group records in a Table with common properties together, making it easier for MI users to work with smaller, targeted collections of data of interest for a specific purpose. Subsets can be associated with a specific Layout, and, the Subset and Layout together may be used to:

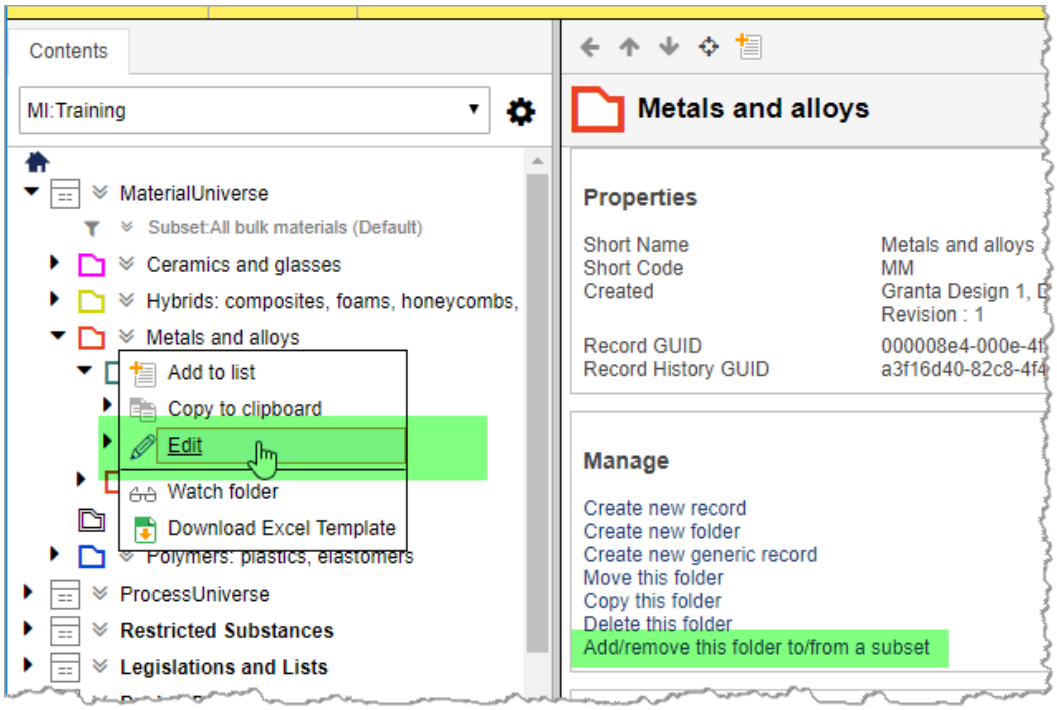
- Restrict access to part of a Table,
- Simplify data, by showing only relevant records or Attributes to users,
- Provide a different perspective on the data.

When an MI:Viewer user selects a Subset for a Table in the Contents tree, only records in the specified Subset will be visible in the tree view or found when searching. For example, here you can see how Subset selection in the MaterialUniverse Table results in a different set of available folders and records in that Table:



The Subsets for a table are defined in the MI:Admin Schema tool on the **Subsets** page for the table (**Table > Subsets**).

Users with Write privileges can move records and folders into and out of the Subsets in a table via the record/folder Properties in MI:Viewer (Edit mode). Subset membership can also be set/changed when importing data via GRANTA MI importers, and using the Record Manipulator plug-in in MI:Toolbox.



Defining multiple Subsets is optional. When you create a new table, it automatically has 1 default Subset in it. All of the records in the table can remain in the default Subset without any further action.

8.1 Viewing and managing Subsets

To view and manage the Subsets in a table: in the MI:Admin Schema tool, select the database, double-click on the table, then click **Subsets.**

8.2 Properties

Subsets in GRANTA MI have the following properties.

8.2.1 Name

This may be up to 255 characters long. The name entered here must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "All attributes" and "All Attributes").

8.2.2 Layout

(Optional) A Layout which will be associated with this Subset. In MI:Viewer, when a user selects the Subset, this Layout will also be selected automatically. If no Layout is associated with a Subset, the default Layout for the Table will be used when the Subset is selected.

9 Records

Data is stored in records. A record is composed of

- Attributes that hold all the data about one particular material, process, coating, substance, test, and so on, in the database. (aka Fields)
- Links to other records via Record Link Groups

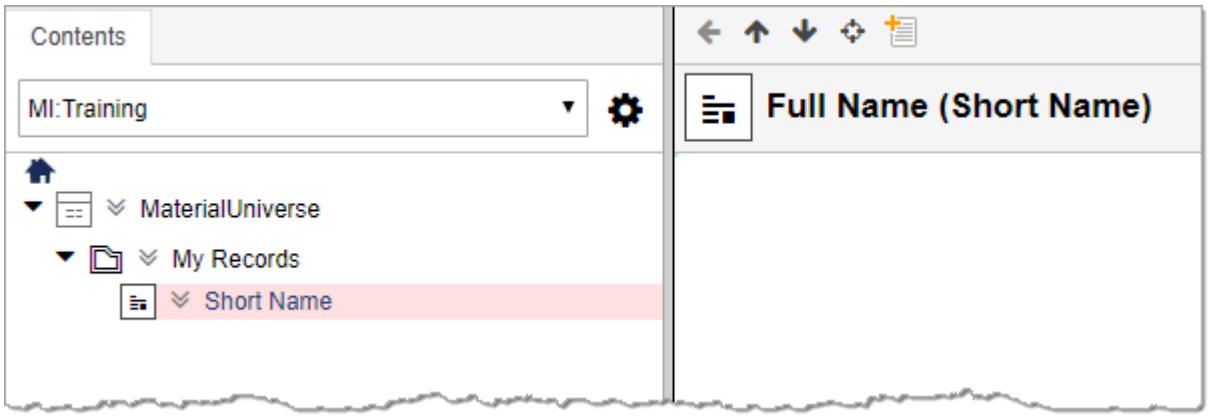
Records have the following properties.

9.1 Full name

The record full name. May be up to 255 characters long.

9.2 Short name

A shorter version of the full name, up to 255 characters long. This is used to label the record in the MI:Viewer Contents tree, and can also appear after the full name in datasheets and in lists of records.



You can't have two records with the same Short Name in the same folder.

9.3 Short code

An optional code which could be used, for example, in a material taxonomy system.

9.4 Color

Records can be assigned a color which can be used in applications such as MI:Viewer to differentiate categories of record in the Browse tree and in record lists, for example, by material type, process type, test type, or by properties such as recyclability or restricted substance content.

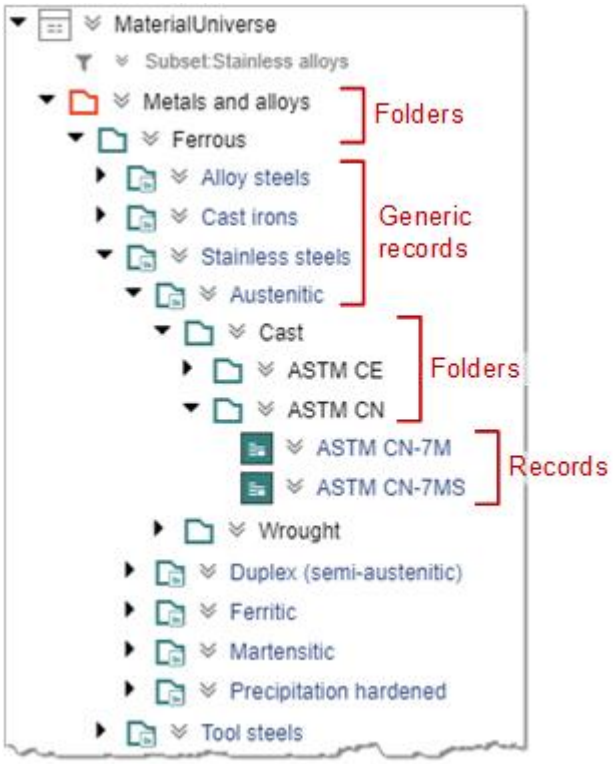
In MI:Viewer, record color can be used as a search criteria, and may also be included in Comparison Table reports.

9.5 Record type

The record Type may be one of Record, Folder, or Generic:

- **Record—**a data record
- **Folder**—an organizational object used to group together data records to make browsing records easier in applications such as MI:Viewer and MI:Toolbox. Folders do not contain any data.
- **Generic record**—a record that is also a folder. Generic records are typically used to organize groups of similar records and to capture data that is common to all of the records beneath them.

For example, in this screenshot from MI:Viewer, steel, cast iron, stainless steel, and tool steel records in the MaterialUniverse Table are organized into a series of nested folders and generic records. The *Stainless steels* generic record here includes general information about strengths and limitations, composition, and processing properties common to the steel records below it in the tree:



In MI:Viewer, the Record Type property can be used as a search criteria and may also be included as a column in Comparison Table reports.

9.6 Date created, Created by

The date when the record was created, and the user who created it. Both of these record properties can be used as search criteria in MI:Viewer, and may be included in Comparison Table reports.

9.7 Date last modified, Modified by

The date when the record was last modified, and the user who carried out the modification. Both of these record properties can be used as search criteria in MI:Viewer, and may be included in Comparison Table reports.

Operations that will cause a change to a record's modification date are:

- Data changes (creation, modification, or deletion of datum values).
- Changes (creation, modification, deletion) to Meta-attribute data.
- Changes to quality ratings.
- Changes to the record Full Name, Short Name, Short Code, or Color.
- Changes to the record type.

The record last modified date will not be changed as a result of:

- Adding the record to/removing it from a Subset.
- Adding or removing static record links in the record.
- Changes to data derived from linked records via Smart Record links.
- Edits to Discrete Type values; see Editing Discrete Type values for details.

9.8 Record GUID (RGUID)

This is a GUID (an integer number in 8-4-4-4-12 format) that uniquely identifies the record. It is unique across every table, every database, every server.

In version-controlled tables, where multiple record versions exist, each version has a different Record GUID, and so the RGUID can be used to identify a specific record version, for example, when exporting or importing data.

In MI:Viewer, the Record GUID can be included in Comparison Table reports.

9.9 Record History GUID (RHGUID)

This is another GUID which can be used to identify the record.

Where multiple record versions of the same record exist, all versions have the same Record History GUID. Accessing a version-controlled record by its RHGUID, for example, via a URL, will always return the latest version of the record that the user is permitted to see (for example, for Read users this would be the Released record, while Write users would see the unreleased record). To access a specific version, use the Record GUID (RGUID).

9.10 Record History ID

This is an internal identifier.

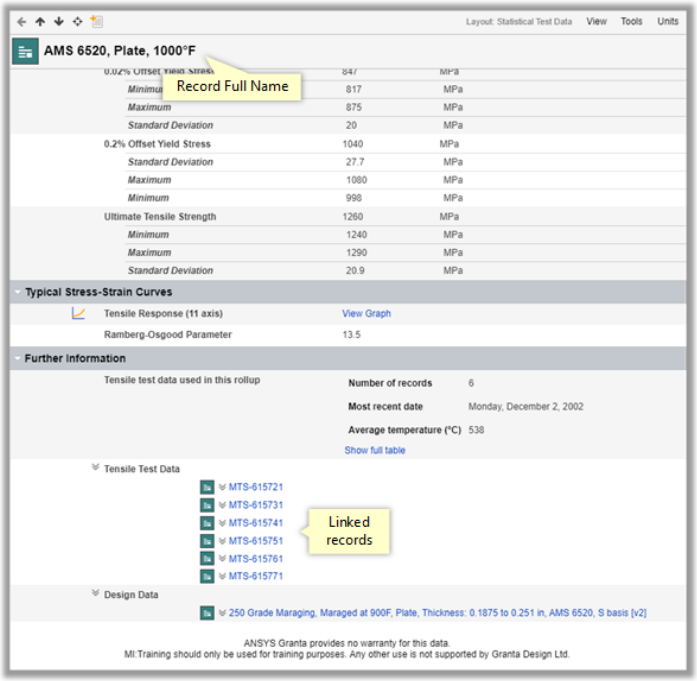
9.11 Datasheets

Datasheets are how MI:Viewer and MI:Explore users view and interact with the data in a record.

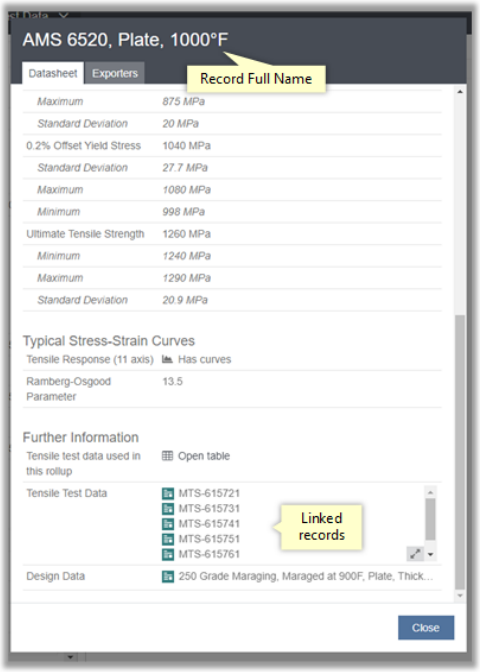
The record Full name is shown at the top of the datasheet, and everything else in the datasheet—headings, Attributes, links to other records—is determined by a Layout, which can be used to present different sets of data for different audiences. Datasheets in MI:Viewer may also include a standard header and/or footer, defined as a property of the table.

Example tensile statistical data record datasheet in MI:Viewer (left) and MI:Explore (right):

MI:Viewer datasheet



Explore datasheet



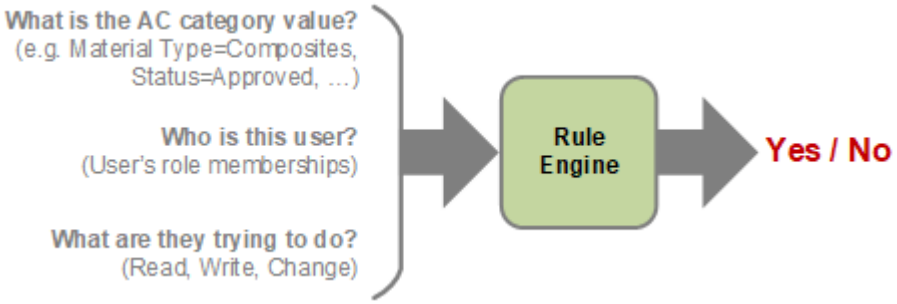
10 Access Control Categories

Attribute-based access control (ABAC) in GRANTA MI allows access to individual records to be granted or denied based on the value of specially-configured 'security' Attributes in those records. For example, you could restrict who can see or edit material records based on the material's development status (Submitted, Approved, In testing, etc), ensuring that employees in Test and Production groups don't have access to new materials until they have been approved.

Access Control Categories are central to the configuration of ABAC in GRANTA MI. For more information about Attribute-based Access Control for GRANTA MI, see the *GRANTA MI Access Control and Security Guide*.

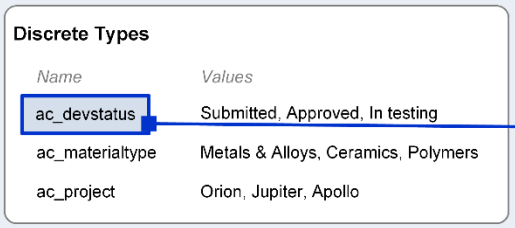
In summary, the Attribute-based Access Control workflow in GRANTA MI is this:

- Access Control Categories are database level objects with a Discrete Type that defines a set of permitted values.
- The permitted values of each Access Control Category are mapped to system security roles.
- Users must be assigned to the relevant system security roles in your authorization system (e.g. AD, User Manager).
- Each Access Control Category may be mapped to a Discrete Attribute with the same Discrete Type in one or more tables.
- The access control Rule Engine then determines, for a given record and user, what the various possible permutations of Access Control Category values imply for that user in terms of their permissions to (a) Read the record, (b) Write to the record and (c) Set or change the value of the Access Control Category in the record.

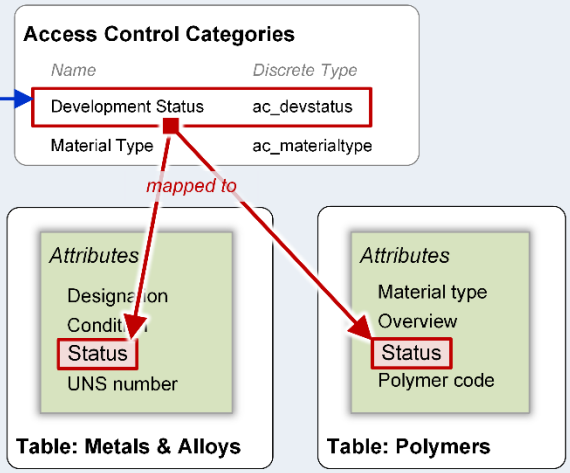


For example, the workflow for restricting access to a material record based on the material's development status could be as follows:

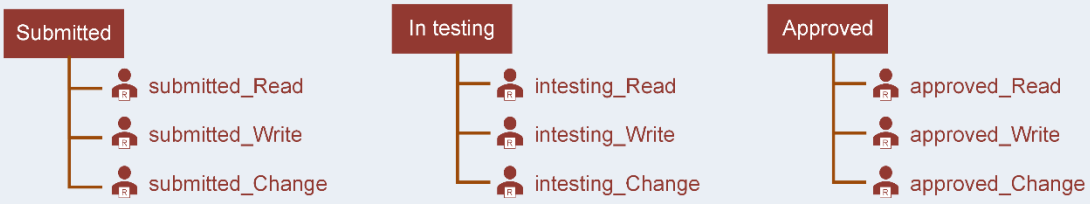
1. Define the Discrete Type



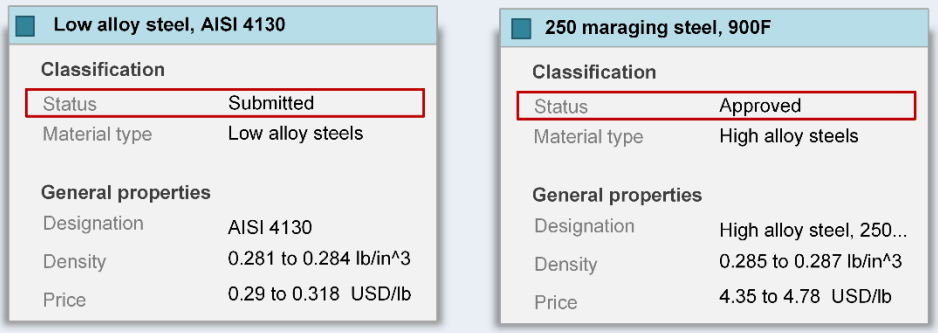
2. Define the Access Control Category & attributes



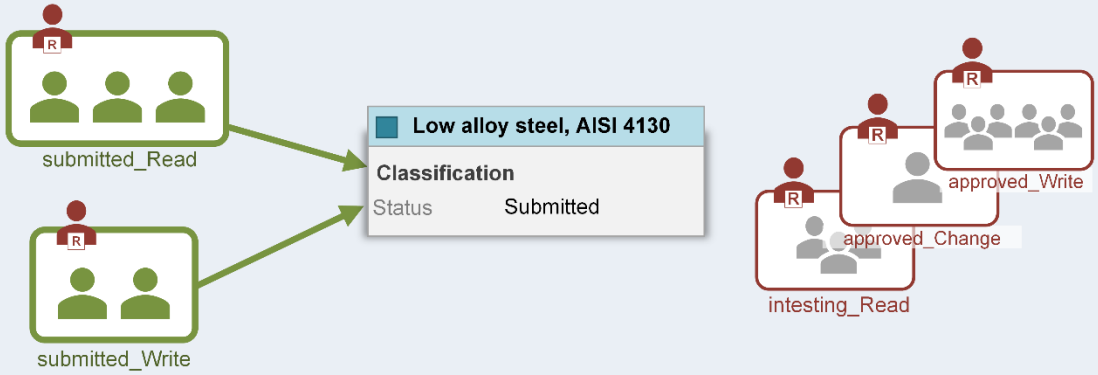
3. Configure the access control rules in the Rule Engine (map values to roles)



4. Set access control values for records



5. Only users who are members of the mapped roles can access the records



10.1 Viewing and managing Access Control Categories

To view and manage the Access Control Categories in a GRANTA MI database: in the MI:Admin Schema tool, select the database then click on the **Edit Access Control Categories** link. All of the Access Control Categories currently defined in the database are listed, showing the name and Discrete Type.

Note that the **Edit Access Control Categories** link will only be available in MI:Admin if Attribute-based Access Control has been enabled in your system (in MI:Server Manager, **Access Control Settings>Attribute-based**).

Before creating new Access Control Categories, you should create the required Discrete Types which will define the permitted values; see Discrete Types.

Note that deleting an Access Control Category will not delete any of the Discrete Attributes to which it is mapped in the database Tables; to remove these, you will need to delete them individually from each Table.

10.2 Properties

Access Control Categories in GRANTA MI have the following properties.

10.2.1 Name

Maximum 255 characters. The name entered here must be unique within the database; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same database (e.g. "ac_materialtype" and "AC_materialtype").

10.2.2 Discrete Type

The Discrete Type that defines the range of values for the Access Control Category.

10.2.3 Attribute

The Access Control Category is a database-level object, and it must be mapped to a Discrete Attribute in each Table where you want to use it to control access to records. The target Attribute must have the same Discrete Type as the Access Control Category.

This mapping is performed in the **New/Edit Access Control Category** page in the MI:Admin Schema tool:

- If a suitable Discrete Attribute already exists in the Table, you can simply select it from the list. Only Attributes with the same Discrete Type as the Access Control Category are listed.
- If no suitable Discrete Attribute exists in the Table, you can create one and map it in a single operation by selecting **<create new Attribute>** in the list. The new Attribute will have the same name and Discrete Type as the Access Control Category. If a Discrete Attribute of the same name already exists in that Table, the new one will have a number in parentheses at the end of its name e.g. *ac_materialtype (2)*.

Note that you will also need to ensure that the Discrete Attributes to which the Access Control Category is mapped are included in the relevant Layouts in each Table.

Once this mapping between Access Control Category and Discrete Attribute has been made in MI:Admin, users who have Change permission for the Access Control Category can set the Attribute value in the same way as any other Attribute, by editing the record in an MI application.

10.2.4 Table-level Values

A **Table-level Values** option in the **New/Edit Access Control Category** page allows the Granta Administrator to set the value of a mapped Discrete Attribute on the Table itself, effectively controlling access to the whole Table and all of the records in it in a single operation.

- The value is set on the table 'root node' only; setting a Table-level value does not set or change the current value (set or unset) of the Discrete Attribute in individual records in the Table.
- The Table-level value combines by a logical AND operation with any values set for the Discrete Attribute in individual records in the table. So, to see a record, the application user must be permitted to see the Table-level value AND the record value. If the value of the Discrete Attribute set on the record permits an application user to see the record, but the Table-level Attribute value prevents them from seeing the Table, the record will be hidden from the user, and they will not be able to browse it (e.g. via links) or search it.

If the Discrete Attribute in the target Table is configured to allow multiple values, you will be able to select more than one Table-level value in the **New/Edit Access Control Category** page.

11 Constants

A GRANTA MI data schema may define Constants which can be included in expressions used in Equations and Logic Attributes, for examples: *Electron Charge*, *Planck Constant*. Constants are database-level objects and can be used in expressions defined in any of the tables in the database.

To view and manage the Constants in a GRANTA MI database: in the MI:Admin Schema tool, select the database then click on the **Edit Constants** link. All of the Constants currently defined in the database are listed, showing the Constant name, unit, and value, and the name of any Equations and Logic attributes in which it is currently being used.

11.1 Properties

Constants in GRANTA MI have the following properties.

11.1.1 Name

Maximum 255 characters. The name entered here must be unique within the database; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same database (e.g. "Electron Charge" and "Electron charge"). When you change the name of an existing Constant, its name is updated wherever it is used.

11.1.2 Unit *(optional)*

You can select an existing unit, or create a new one. Note that changing the unit of a Constant will not change its numerical value.

11.1.3 Value

A numeric value.

12 Discrete Types

Discrete Types are used to define the set of permitted values for a Discrete Attribute, which is a text Attribute that takes one or more values from a predefined or 'discrete' set of values. Any number of Discrete Types can be defined in the database, and used by Discrete Attributes and Meta-Attributes in any of the Tables in the database, and also for Access Control Categories.

The same Discrete Type can be used by several different Discrete Attributes, for example, *Standard* or *Capital cost* or *ChemRes Ranking*. Discrete Types may be **ordered** or **unordered**. For example:

Ordered:

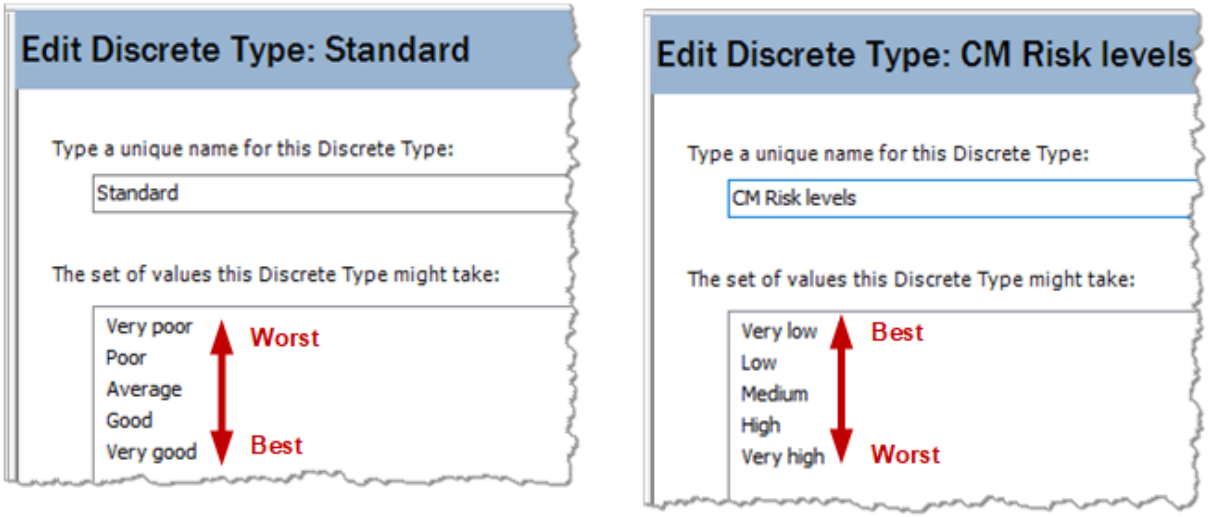
- *Standard* (Very poor, Poor, Average, Good, Very good)
- *Flammability* (Highly flammable, Slow-burning, Self-extinguishing, Non-flammable)
- *Electrical conductivity* (Electrical insulator, Semiconductor, Electrical conductor)
- *ChemRes Ranking* (Unsatisfactory, Doubtful, Limited, Probably satisfactory, Satisfactory)

Unordered:

- *Cross-section geometry* (Rectangular, Circular, Tubular, Other)
- *Impregnation process* (Hot melt, Solvent, Other)
- *Material Family* (Metal, Plastic, Elastomer, Ceramic, Glass, Natural)
- *Additive* (None, Flame retardant, Impact modifier, Plasticizer/oil, Plasticizer/phthalate, Anti-friction/wear lubricant, UV stabilizer)

12.1 Discrete Type thresholds

The order of Discrete Type values ('best' to 'worst', or 'worst' to 'best) is significant for Attribute-based searches in MI:Viewer.



Where values are ordered 'worst' value first, 'best' value last (e.g. the *Standard* Discrete Type in the example shown above):

- Searches with a threshold of 'is at least' will return records with that value *and* better
- Searches with a threshold of 'is at most' will return records with that value *and* worse.

For example, consider a search in the MI:Training database based on the *Toughness rating* Discrete Attribute, which uses the *Standard* Discrete Type shown above:

- 'Toughness rating **is at least** Average' will return all records with a *Toughness rating* of Average, Good, or Very good.
- 'Toughness rating **is at most** Average' will return all records with a *Toughness rating* of Average, Poor, or Very poor.

Where values are ordered 'best' value first, 'worst' value last (e.g. *CM Risk levels* in the example shown above) :

- Searches with a threshold of 'is at least' will return records with that value *and* worse.
- Searches with a threshold of 'is at most' will return records with that value *and* better.

For example, consider a search in the Product Risk database based on the *Abundance risk level* Discrete Attribute, which uses the *CM Risk levels* Discrete Type shown above:

- 'Abundance risk level **is at least** Medium' will return all records with Abundance Risk Level values of Medium, High, or Very high.
- 'Abundance risk level **is at most** Medium' will return all records with Abundance Risk Level values of Medium, Low, or Very low.

12.2 Viewing and managing Discrete Types

To add or edit the Discrete Types in a GRANTA MI database: in the MI:Admin Schema tool, select the database and then click on the **Edit Discrete Types** link. The list shows the name, type (ordered or unordered), and in which Attributes they are currently used.

12.3 Properties

Discrete Types in GRANTA MI have the following properties.

12.3.1 Name

This may be up to 255 characters long. The name must be unique within the database; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same database (e.g. "Laminate Process" and "Laminate process").

12.3.2 Values

Click **Add** to add a new value, or **Paste New** to paste the clipboard contents as a new value.

Use the **Move up** and **Move down** buttons to order the values (for ordered Discretes).

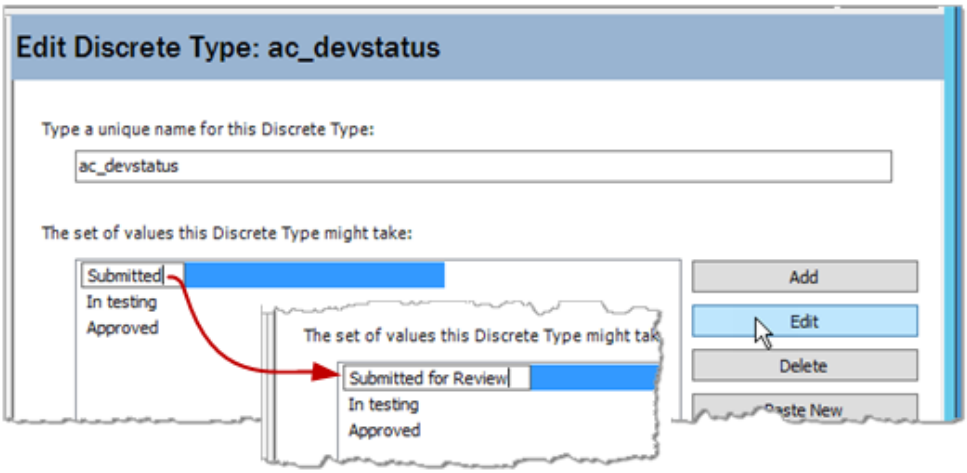
To remove a value from the list, select it and click **Delete**; you will not be able to remove values that are currently assigned in any records in the database. See below for information about editing existing values.

12.3.3 Ordered

Select the check box for a Discrete Type with ordered values (e.g. Poor, Average, Good, Very good); clear the check box if the values should not be ordered (e.g. Circular, Rectangular, Tubular, Other).

12.4 Consequences of editing Discrete Type values

When you edit an existing Discrete Type value, for example, to correct a typo ("Durablity" to "Durability"), or to make some other change to it (as in the image below), the value will be changed throughout the system: the new value will replace the original value everywhere it appears.



Read the information below carefully to ensure that you understand the consequences of changing Discrete Type values, particularly when the Discrete Type is used in an Access Control Category.

12.4.1 Record modification date, record version

Discrete Type values are part of the database schema and are not treated as a data. Therefore, modifying a Discrete Type value:

- Will **not** trigger a change to the modification date of records in which the original value was used.
- Will **not** be tracked in version-controlled records, and will not result in a new record version being created.

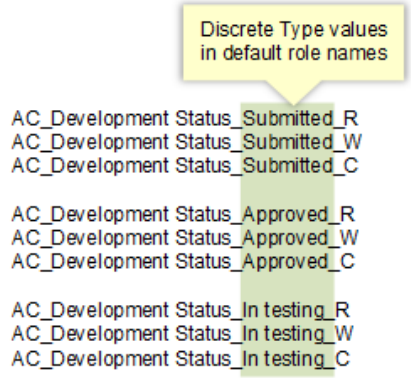
12.4.2 Attribute-based Access Control (ABAC) role names

Note In systems where Attribute-based Access Control is used, modifying the value of a Discrete Type that is mapped to an Access Control Category will have an impact on the security roles defined in the Rule Engine and may affect users' access to data.

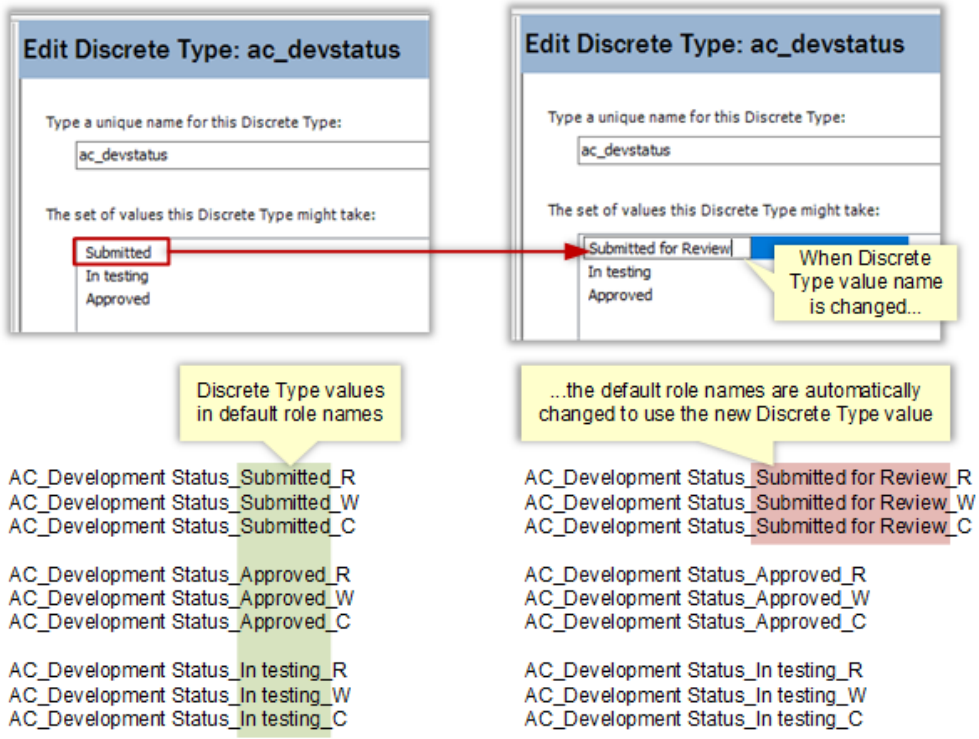
In Attribute-based Access Control, Discrete Types are used to define the permitted values for Access Control Categories. The permitted values of each Access Control Category are mapped in the Rule Engine to system security roles. Default role names are constructed by concatenating the domain name (in systems with Windows authentication), the Access Control Category name, the **Discrete Type value**, and a permissions identifier, as follows:

domain\AC_<*AC Category Name*>_<*Discrete Type Value*>_R (role with Read permission)
domain\AC_<*AC Category Name*>_<*Discrete Type Value*>_W (role with Write permission)
domain\AC_<*AC Category Name*>_<*Discrete Type Value*>_C (role with Change permission)

For example: an Access Control Category called *Development Status* is defined with the Discrete Type *ac_devstatus*; the *ac_devstatus* Discrete Type has the values *Submitted*, *Approved*, and *In testing*. The default role names in the Rule Engine are therefore:



If the Discrete Type value that is used in an Access Control Category is modified, the default ABAC roles will be automatically renamed to reflect the modified value. For example:



In addition, any roles that have been entered manually in the Rule Engine configuration tool (overriding the suggested default role name) **will be replaced** by a new default role name that includes the modified Discrete Type value.

13 Excel Template Definitions

13.1 Excel Templates in GRANTA MI

'Excel templates' are specialized Microsoft© Excel workbooks that can be used to export data from GRANTA MI, and to import new data, or update existing data in GRANTA MI. Excel templates allow users with no knowledge of GRANTA MI or the structure of the database to enter, edit, or analyze data in Excel, and the data can then be imported into GRANTA MI via Remote Import or the MI:Toolbox Excel Importer plug-in.

GRANTA MI Excel templates include one or more data worksheets where data values may be specified, and additional worksheets that specify the data structure and options including:

- Boilerplate text and graphics
- For data import: what type of data can be entered into cells in the data worksheets; options relating to record naming and placement, auto-linking, conflict resolution, and access control;
- For data export: which records and Attributes will be exported.

While Excel Templates can be defined manually, this is a specialist task, requiring familiarity both with Microsoft Excel and with the structure of the database into which the data is being imported; see the PDF document *GRANTA MI Excel Importer Reference Guide* for details.

Excel Template Definitions allow MI:Viewer users to generate Excel Templates for data import and for exporting individual records, without requiring any specialist knowledge of the data schema or of export/import template structure/configuration.

13.2 Excel Template Definitions

Excel template definitions are created in MI:Admin and specify options that are used in Excel Templates generated in MI:Viewer, including

- Style elements for the data worksheets, such as a company logo or heading color, and boilerplate text such as notes for data input users on how to fill in data values.
- Import options including name format for new records, data conflict resolution preferences, and options for import of Functional and multi-value Discrete data.
- Placement options for new records.
- Data validation options.

Note: Equations and Logic (EEL) Attributes are not included in Excel template definitions created in MI:Admin. If you need a template that can export/import Equations and Logic Attributes, you will need to edit the template in Microsoft Excel to add the required expression, parameter, and curve details, as covered in the *GRANTA MI Excel Importer Reference Guide* and *GRANTA MI Excel Exporter Reference Guide.*

13.3 Viewing and Managing Excel Template Definitions

To add or edit the Excel Template Definitions in a GRANTA MI table: in the MI:Admin Schema tool, select the database, click to expand the table, then click on the **Edit Excel Template Definitions** link.

14 Expressions

Expressions may be defined for use in the evaluation of Equations and Logic data within a table. An Expression represents a single mathematical expression that can include numeric Attributes from within the same Table, as well as Parameters, Constants, and Functions, referenced as follows:

Type	Format	Examples
Parameter	[P:<name>]	[P:Temperature] [P:Cycles]
Constant	[C:<name>]	[C:Electron Charge] [C:Volume of Ideal Gas at STP]
Attribute	[A:<name>]	[A:Density] [A:Shear modulus]
Meta-Attribute	[A:<parent-name>#<meta-name>]	[A:Density#Sample size] [A:Shear modulus#Std Dev]

For Attributes and Meta-Attributes only, the `A:` prefix may be omitted. e.g. `[Density]`.

The calculation of Expressions is performed in database units; the result will then be converted to the relevant display unit, if it is different.

14.1 Viewing and managing Expressions

To add or edit the Expressions in a GRANTA MI table: in the MI:Admin Schema tool, select the database, double-click on the table, then click **Expressions**.

14.2 Properties

Expressions in GRANTA MI have the following properties.

14.2.1 Name

This may be up to 255 characters long. The name must be unique within the database; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same database (e.g. "Process Model" and "Process model").

Maximum 255 characters. When you change the name of an existing Expression, its name is updated wherever it is used.

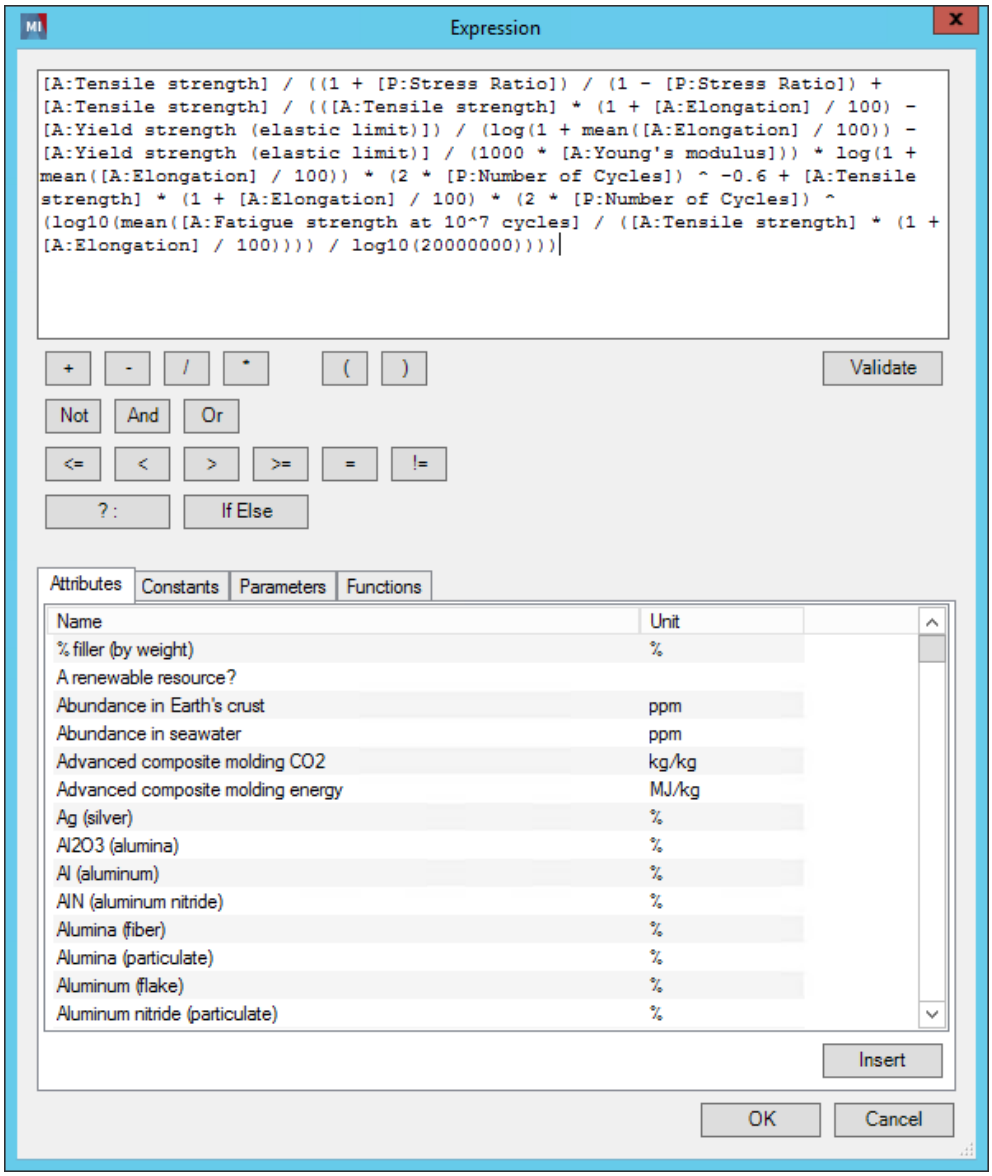
14.2.2 Unit *(optional)*

You can select an existing unit, or create a new one.

14.2.3 Value

Type the expression in the form, or click **Edit** and use the Expression editor to construct it from the available Attributes, Constants, Parameters, Functions, and operators in the database/Table. To check the Expression syntax, click *Validate*; for detailed information about Expression syntax in GRANTA MI, see Expression syntax reference.

Expression editor:

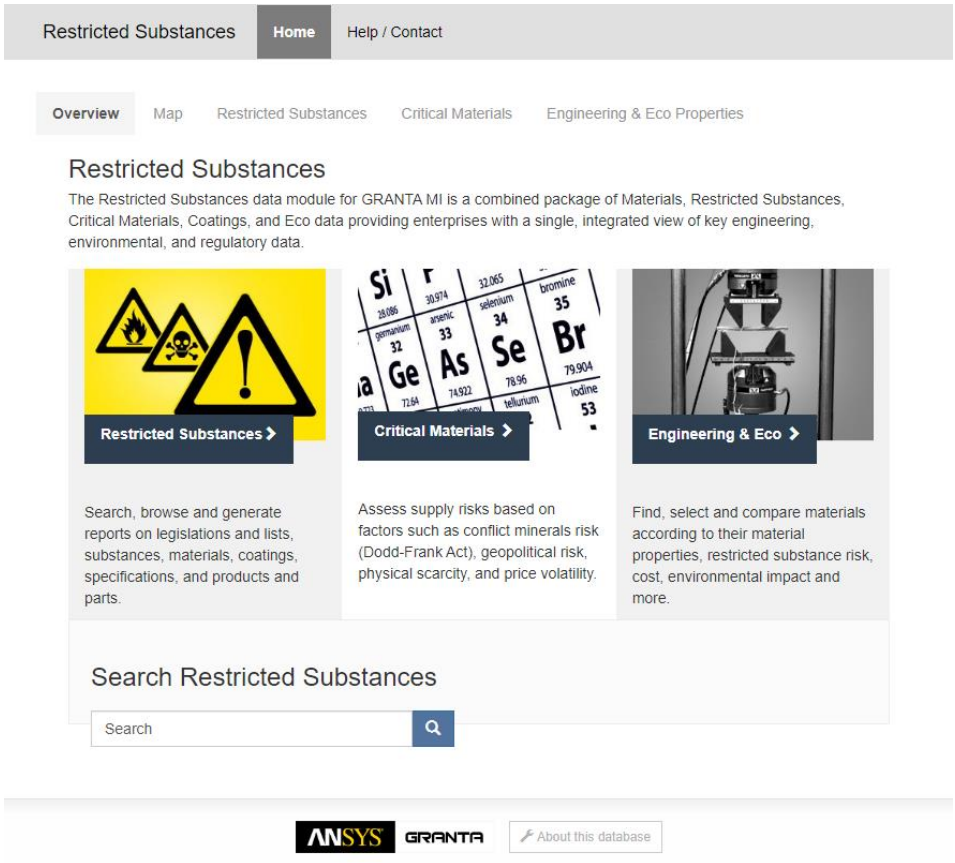


15 Files

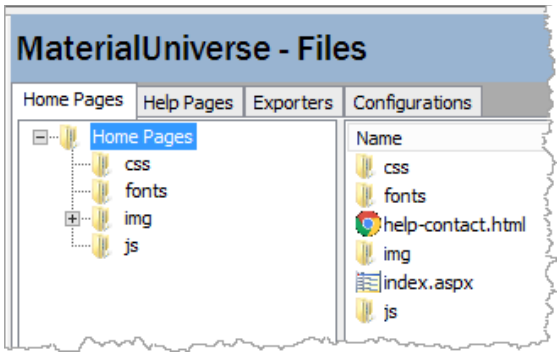
A number of files that contain supporting information for database users are stored in the database and managed in MI:Admin. These database 'metadata' files should not be confused with data files (such as a PDF or text files), which are stored in the database as File data.

15.1 Database home pages

Database home pages in MI:Viewer provide a user-friendly 'landing page' for databases, typically including links to useful resources, focused search/filter tools, and information about the database content, for example:



Database home page files may include ASPX or HTML files, and related CSS, image, font, and JavaScript files. These files are stored in the database and managed on the Home Pages tab of the Files page in the Schema tool:



The ASPX or HTML home page for the database must be stored at the top level of the *Home Pages* folder; other resource files used for the home page can be stored at the same level or in subfolders. In this example, the database home page is index.aspx, and the other resources have been stored in subfolders.

The home page file can be any file that can be served by IIS, but note that your Windows system administrator may have restricted the types of files that can be served. In a default installation of MI:Viewer, IIS searches for a default document in the following order. The first match it finds becomes the database home page.

1. index.aspx
2. index.html
3. index.htm
4. default.html
5. default.htm

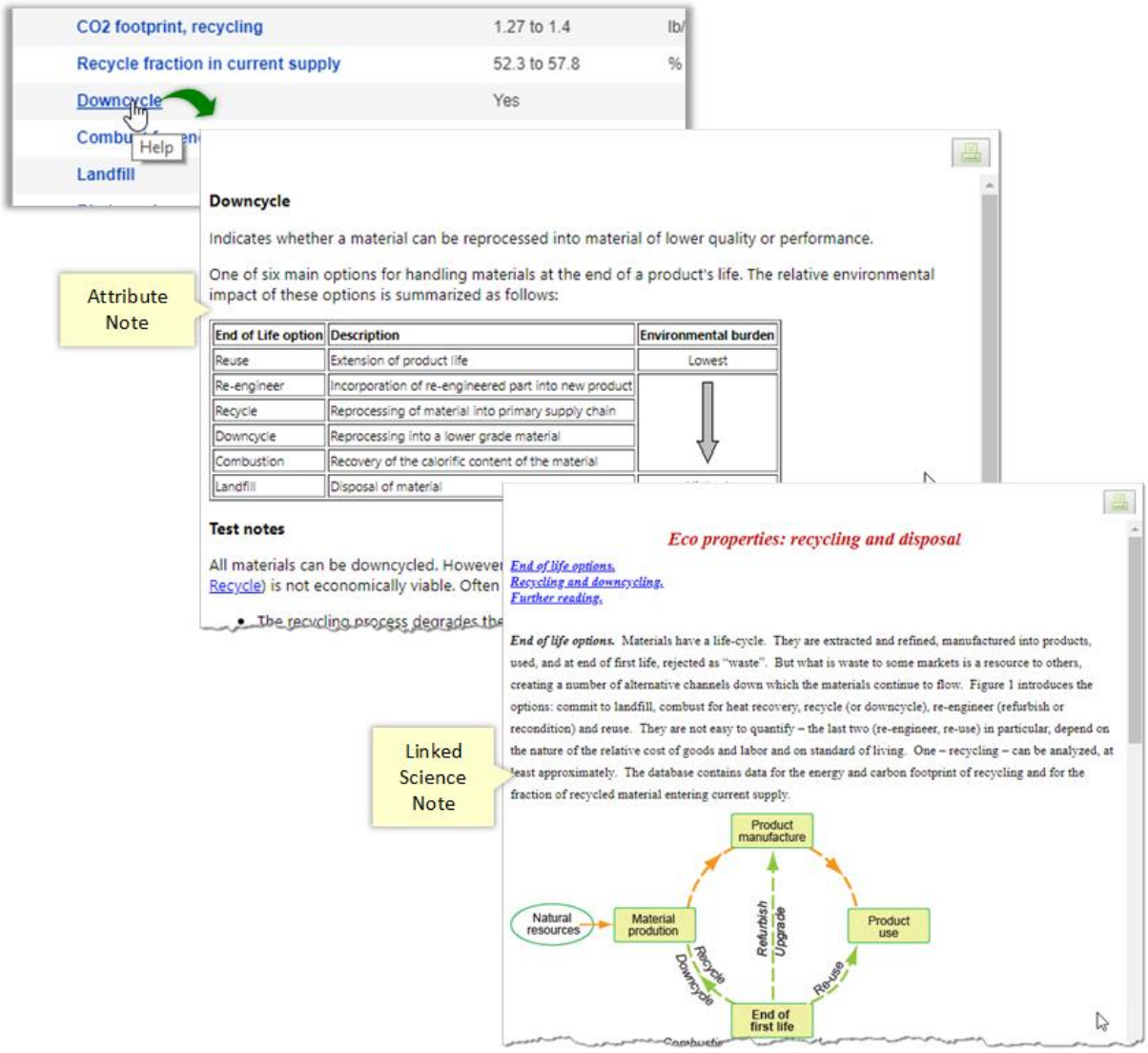
Database home page files are stored on the MI:Viewer web server in a *homepagefiles* subfolder under the *mi* web application root folder, in a folder with the database name, for example:

C:\inetpub\wwwroot\mi\homepagefiles\MI_Training\

15.2 Attribute Help pages

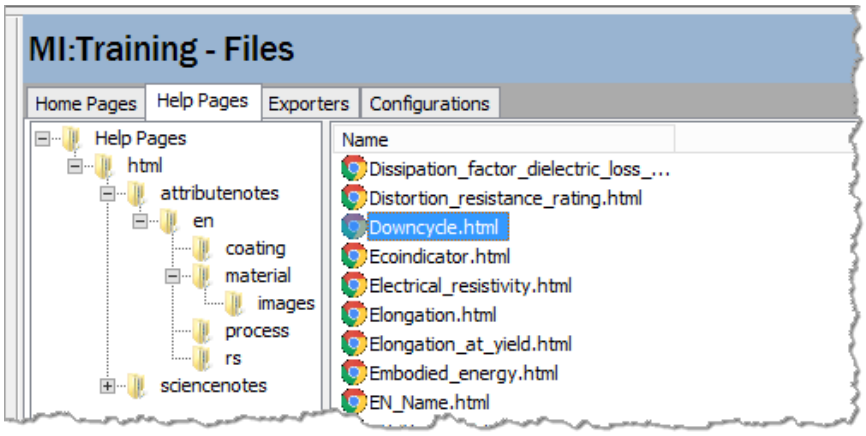
A GRANTA MI database can include help pages that provide descriptions of, or additional information about, the Attributes in the database, for example, the "Attribute Notes" and "Science Notes" in GRANTA MI reference databases. When help pages are configured for an Attribute, MI:Viewer users can click on an Attribute name in a datasheet to view information about it. For example, in this picture from the MI:Training database, the *Downcycle* Attribute has an Attribute Note, which includes a link to a Science Note that provides general reference information about recycling and

disposal properties; both the Attribute Note and Science Note in this example are defined as Help Pages in the MI:Training database:



Help pages can be provided in the same way for the Parameters defined in a GRANTA MI database.

Help pages are authored outside of GRANTA MI, and stored in the database and managed using the MI:Admin Schema tool:



To add help pages for an Attributes, you need to:

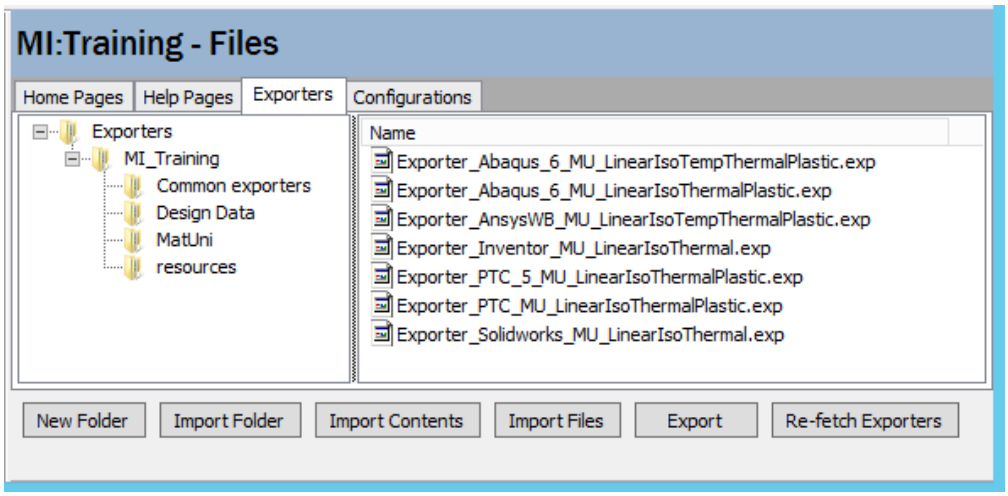
1. Create the help files (HTML, CSS, image files) outside of GRANTA MI.
2. Import the files to the database on the Home Pages tab of the Files page in the Schema tool. See Managing files below.
3. Edit the Attribute properties to include a link to the relevant help files. See Attribute properties: Help page.

15.3 FEA Exporters

The FEA exporter files for each database are managed using the MI:Admin Schema tool. Each exporter is specific to a particular database and Table, and consists of:

- an ***exporter configuration file*** (.exp), an XML file which contains all the information about the exporter: the target package and model, the Attribute data to be exported, and the transform files that can be used.
- one or more ***transform files*** (.xslt), which convert the data from the database into the format required by the target package.

All exporter files must be located in the *Exporters* folder as shown below, but below that, you can organize the files into sub-folders in any way you choose.



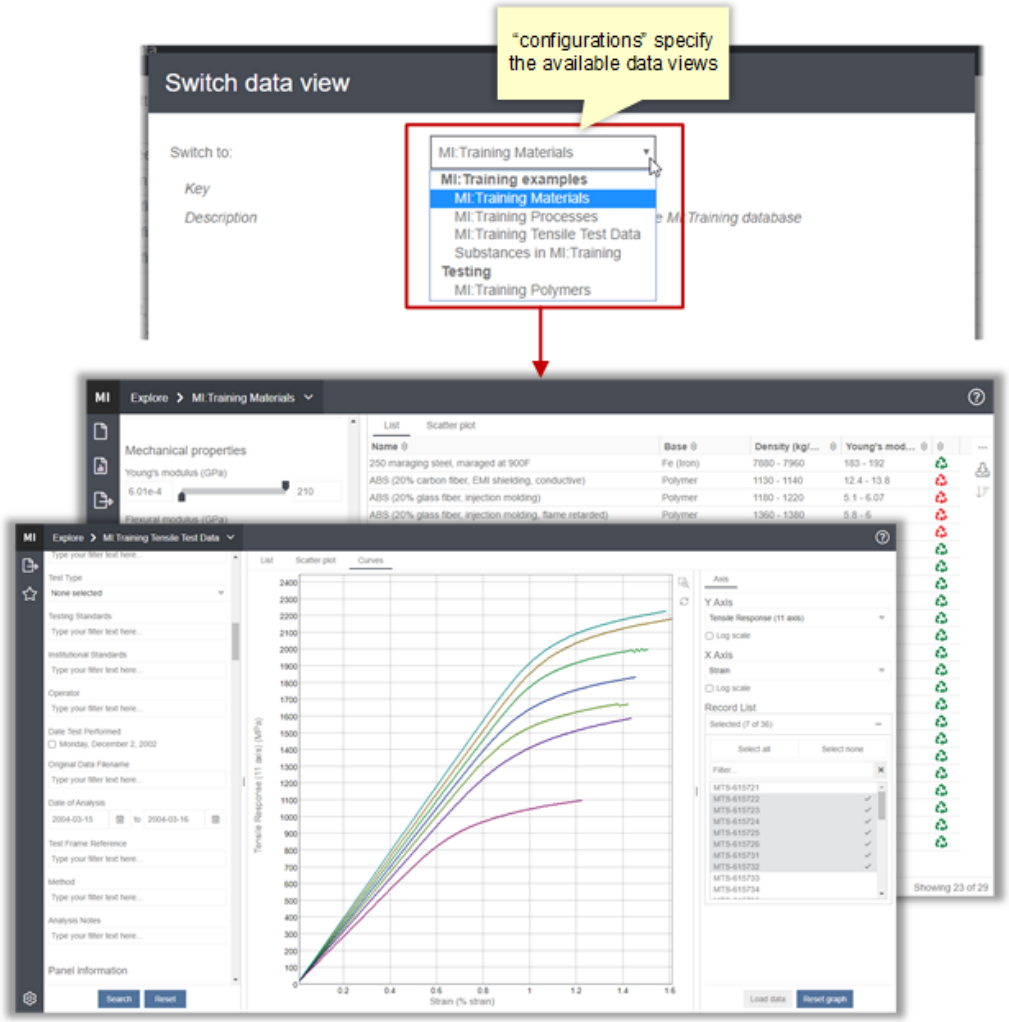
The available FEA exporters are copied from the database to the following folder on the GRANTA MI server whenever MI:Server is restarted and whenever the **Re-fetch Exporters** button in MI:Admin is clicked:

C:\Program Files\Granta\GRANTA MI\Server\exporters\

15.4 MI:Explore configuration files

MI:Explore application configuration files are used to define different data views for MI:Explore users. One file may include a number of 'configurations' specified in JSON (JavaScript Object Notation) format, each of which defines a different data view, specifying:

- the Database, Table, and Subset containing the data of interest
- the Layouts used to filter and group the available Attributes within the MI:Explore application, both in the Search panel and in datasheets
- the availability of features within the application, such as the ability to add new records, to view or edit tabular data, to plot curves, or to generate reports.

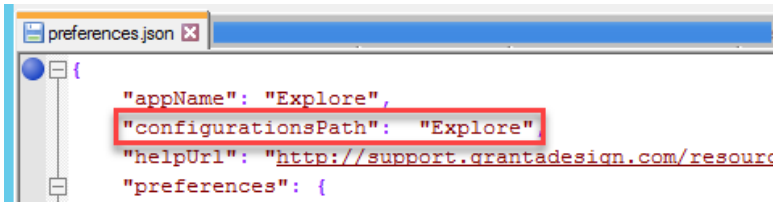


Refer to the *MI:Explore Configuration Guide* for information about all the available application settings for MI:Explore.

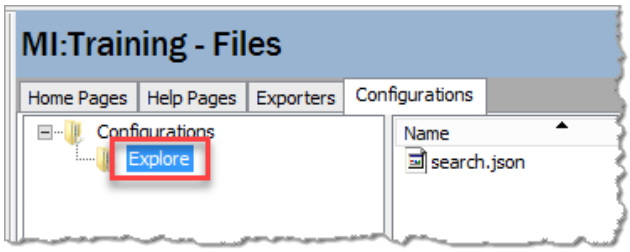
MI:Explore configuration files are imported, organized, and exported in the same way as home page, help, and exporter files. After a configuration file has been added to or removed from the database, no additional action is necessary; MI:Explore users will see all the available configurations by simply refreshing their browser.

In MI:Admin, configuration files for MI:Explore are located in a subfolder under *Configurations*. The subfolder name must match the name specified in the configurationsPath setting in the MI:Explore application preferences.json file; by default, this is *Explore*. For example:

preferences.json file:



Corresponding folder in MI:Admin:



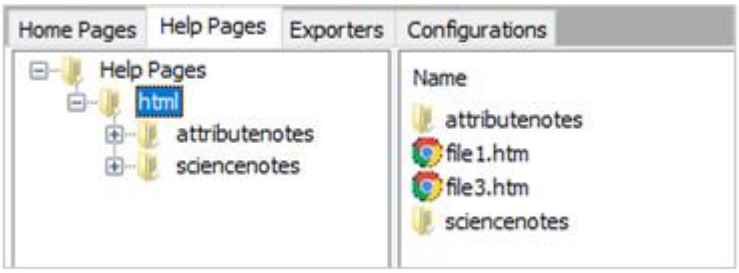
If you export and then modify MI:Explore configurations, or develop your own configurations, you can check that the JSON is valid using the **Validate Configurations** button at the bottom of the Configurations page after uploading them; validation results can be saved in a validation report for troubleshooting.

15.5 Managing files

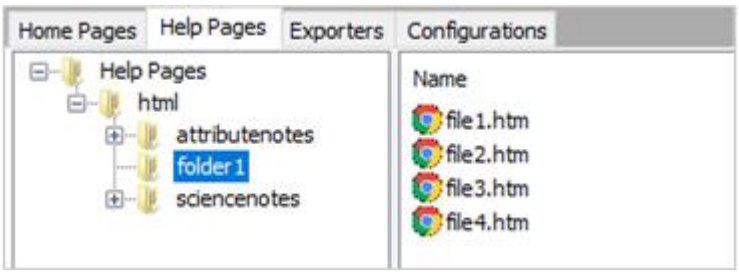
To manage the home page, help page, exporter, and configuration files for a GRANTA MI database: in the MI:Admin Schema tool, select the database then click on the **Edit Files** link.

Click **Import Files**, **Import Folder**, or **Import Contents** to import folders/files. All three of these options are also available on the shortcut menu. The different import options work as shown in this example:

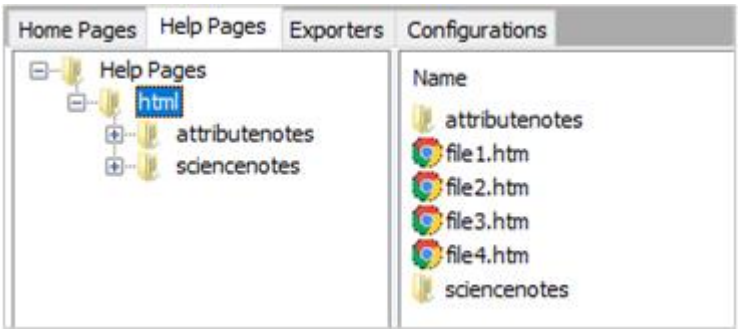
Import Files (the files you select are imported):



Import Folder (the folder you select, and its contents, are imported):



Import Contents (all of the files in the selected folder are imported, the folder is not imported):



If a file or folder with the same name already exists, you will need to specify how to resolve the file naming conflict:

- Choose **Skip** to skip the import of that file/folder, and move to the next one.
- Choose **Replace** to replace the existing file/folder with the imported one. Note that, for folders, the folder and ***all*** of its contents will be replaced.
- Choose **Auto Rename** to automatically rename the imported file/folder; for example *Density.html -> Density(1).html*.
- Select **Apply to all** to apply the same name conflict resolution to all items being imported.

After importing new or modified files, you may need to do some additional tasks to make them available to database users; see Making file changes visible to application users below.

To delete files, select them, then right-click and select **Delete** on the shortcut menu. You may need to do some additional tasks to publish the database changes in applications; see Making file changes visible to application users below. Note that, when removing a database home page, if you delete the home page file but leave other files (such as images or linked resources) in the *Home Pages* folder structure, an error is reported in MI:Viewer when it tries to display the database home page. This error does not affect the other functionality of MI:Viewer (e.g., you can still browse and search, etc.). To return to using the default database home page provided by Granta, you must remove ***all*** files from the top-level *Home Pages* folder.

15.6 Making file changes visible to application users

After adding or deleting files, you may need to do some additional tasks to publish the changes in applications such as MI:Viewer or MI:Explore, so users can see the latest files.

File type	Action required to publish the file changes
Database home page files	Copy the files into the MI:Viewer application: log in to MI:Viewer using an account with system or database Admin privileges, open the Admin>General page, select the database from the list, and click **Refresh home pages**. This will copy all of the home page files for the database into a subfolder in the MI:Viewer application homepagefiles folder, C:\inetpub\wwwroot\mi\homepagefiles\
Help pages	If you added or deleted Help files, you will need to create/update the link to them from the relevant Attributes or parameters. If you updated existing help files, you do not need to do anything.
FEA Exporter files	Copy the files to the MI:Server application server by clicking **Re-fetch Exporters** at the bottom of the Exporters tab in MI:Admin. This copies the exporter files for the database into the application exporters folder, for example C:\Program Files\Granta\GRANTA MI\Server\exporters\MI_Training.
MI:Explore configuration files	No additional action is required: MI:Explore users will see the latest application configuration when they refresh their browser.

16 Parameters

A GRANTA MI data schema may define Parameters that can be used in the evaluation of functional data, for example, temperature, mass, or number of cycles.

16.1 Numeric Parameters

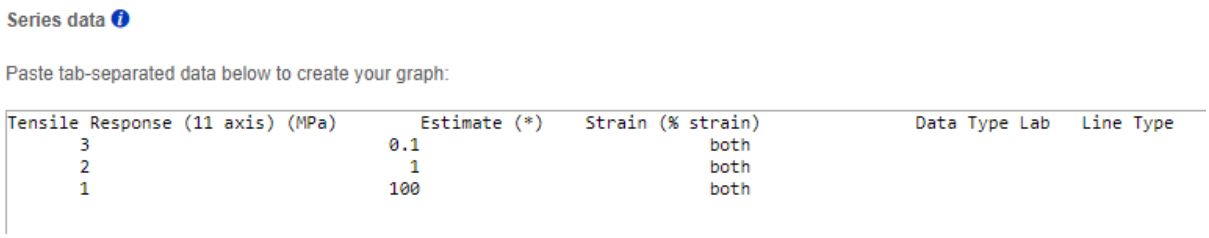
Numeric Parameters have numerical values that may be displayed with or without units; they can be **unrestricted** (may be set to any value, interpolation is permitted) or **restricted** (only a defined set of values is permitted, no interpolation possible). Examples from the MI:Training database:

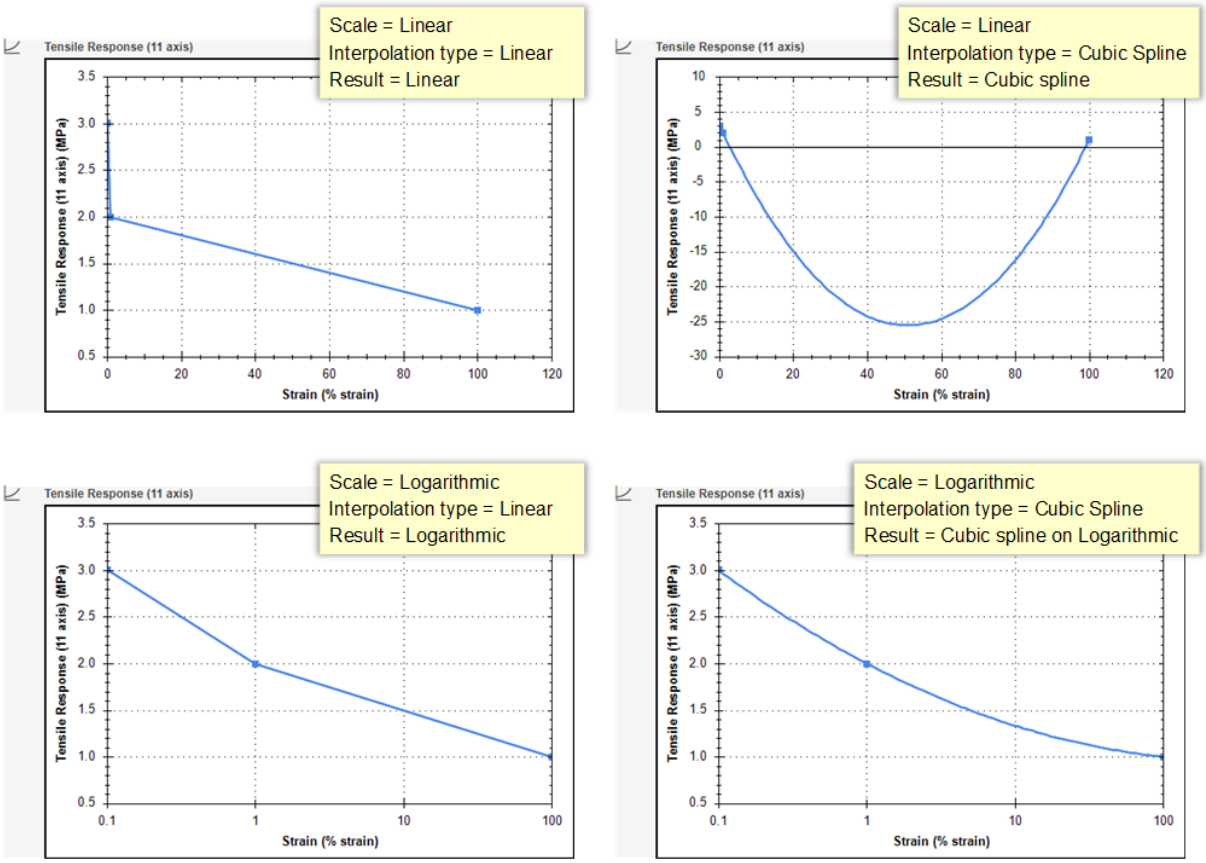
- *Strain* (unit: % strain), used in the Attributes *Isothermal Stress Rupture*, *Tensile Stress/Strain, L*, *Compressive Stress/Strain*.
- *Material Cost* (currency/kg), used in the Attribute *Relative cost index per unit.*

You can set the following scale and interpolation type options for numeric Parameters:

Scale	Interpolation type	Result
Linear	Linear	Linear
Linear	Cubic spline	Cubic spline
Logarithmic	Linear	Logarithmic
Logarithmic	Cubic spline	Cubic spline on Logarithmic

The effect of the different Scale and Interpolation type options for the *Strain* Parameter can be seen in these examples:





16.2 Discrete Parameters

Discrete Parameters have text values. Examples from the MI:Training database include:

- *Statistical Basis*, with the values *Mean*, *A-value* or *B-value*; used in the functional Attribute *Yield strength*, and for some multi-value point Attributes in the Composite Design Data table.
- *Stress/Strain Curve Type*, with the values *Full Range* or *Yield*; used in the Attributes *Tensile Stress/Strain, L*,*Tensile Stress/Strain, ST*, and *Tensile Stress/Strain, LT*

16.3 Viewing and managing Parameters

To view and manage the Parameters in a database: in the MI:Admin Schema tool, select the database and then click on the **Edit Parameters** link. The list shows the name of each Parameter, its type (numeric or discrete), values, default value, and which Attributes the Parameter is currently used in. For numeric Parameters, the scale and interpolation type are also shown.

16.4 Properties

Parameters may have the following properties in GRANTA MI.

16.4.1 Name

The name may not be blank and it may not contain left square bracket ([ ), right square bracket ( ]) or colon (:) characters. Maximum 255 characters. When you change the name of an existing Parameter, its name is updated wherever it is used.

16.4.2 Values

One or more values must be specified.

Use **Set as Default** to specify the default value for this Parameter; for numeric Parameters, this specifies the value used for Interpolation or Selection. If no default value is set, the first value in the list is assumed to be the default value.

For Discrete Parameters, you can order the values using the **Move up** and **Move down** buttons.

16.4.3 Help page

(*Optional*) The location and name of a file containing help information about the Parameter/data ("Parameter Notes"). This is configured in the same way as Attribute Notes; see Configure help pages for Attributes.

16.4.4 Unit (Numeric Parameters)

You can assign an existing unit, or create a new one.

16.4.5 Restrict parameter to these values only (Numeric Parameters)

This property determines whether the Parameter is restricted or unrestricted.

- Clear the check box to create a *Numeric unrestricted* Parameter; this may have any value and must be interpolated.
- Select the check box to create a *Numeric restricted* Parameter; **this has** a restricted set of values and cannot be interpolated.

16.4.6 Interpolation type (Numeric Parameters)

One of: *none*, *Linear*, or *Cubic spline*. By default, Functional Attributes will use the Parameter's interpolation and scale settings unless overridden in the Attribute's Default Content.

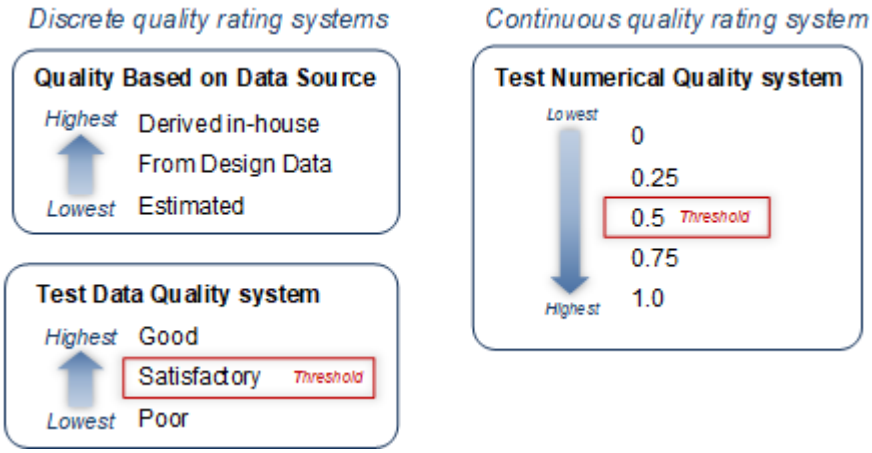
16.4.7 Scale (Numeric Parameters)

One of: Logarithmic or Linear. By default, Functional Attributes will use the Parameter's interpolation and scale settings unless overridden in the Attribute's Default Content.

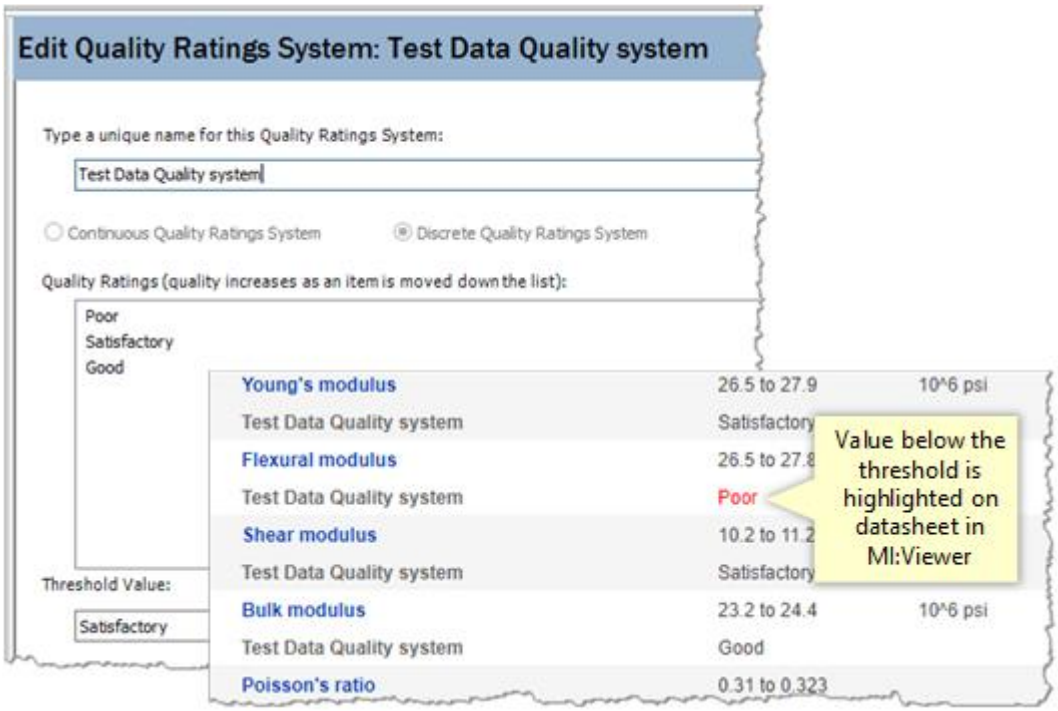
17 Quality Ratings Systems

Quality Ratings Systems in GRANTA MI allow quality ratings to be associated with data. They are defined at the database level, and can be used in ("assigned to") any of the Tables in the database, either when the Table is created or by editing the Table properties later. Each Table can have only one Quality Ratings System in use.

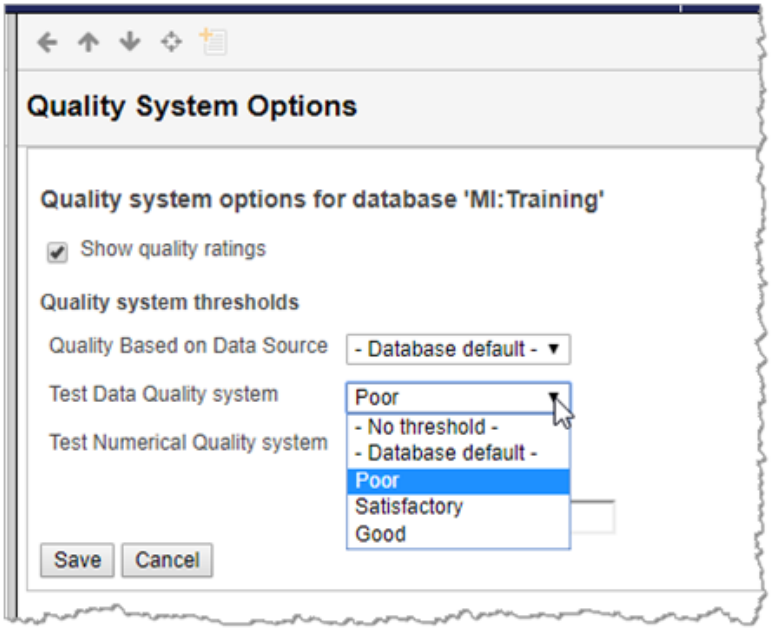
A Quality Ratings System may have discrete values (a specified set of string values) or continuous numerical values:



For both discrete and continuous Quality Ratings Systems, a threshold can be specified, and values below the threshold will be highlighted in red on datasheets in MI:Viewer. For example:



In MI:Viewer, users can choose to show or hide quality rating data on datasheets, and *change the threshold* (changes apply in the current MI:Viewer session only).



17.1 View and manage Quality Ratings Systems

To view and manage the Quality Ratings Systems in database: in the MI:Admin Schema tool, select the database and then click on the **Edit Quality Ratings Systems** link.

You cannot delete a Quality Ratings System that is currently being used in any Tables in the database.

17.2 Properties

17.2.1 Name

Maximum 255 characters. When you change the name of an existing system, its name is updated wherever it is used.

17.2.2 Continuous Quality Ratings System

Select this option for numeric rating values and then specify whether quality increases or decreases with numeric value:

- Select **Quality increases with numeric value** for higher numbers = higher quality (e.g. 1=Poor, 5=Excellent)
- Clear **the Quality increases with numeric value** check box for lower numbers = higher quality (e.g. 1=Excellent, 5=Poor)

17.2.3 Discrete Quality Ratings System

Select this option for a discrete (text) rating system, and then define the set of rating values. Use the buttons to add values, or to rename, delete, and reorder existing values.

17.2.4 Threshold Value

Optionally, set the database default threshold value for the rating. If a threshold is set, any quality rating below the threshold value is highlighted in MI:Viewer (see above for an example). This threshold value can be overridden temporarily by users on the **Options** page in MI:Viewer.

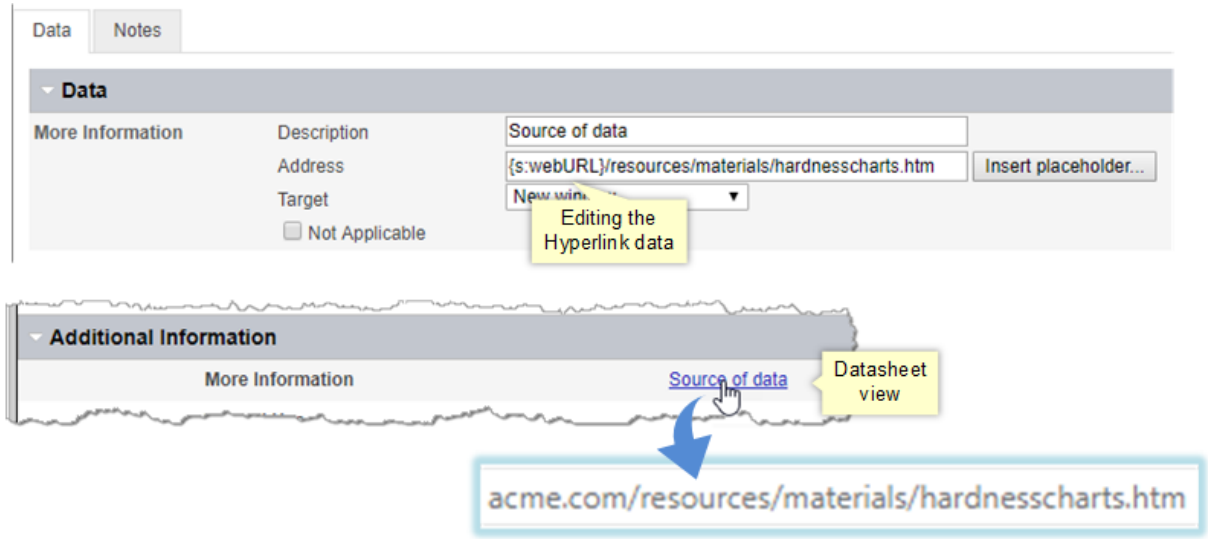
18 Replacement Strings

Replacement Strings are used to define a string that is inserted into a Hyperlink address when the URL is resolved. The ability to define part of a URL once in the database, and then reference it in records makes it much easier to maintain hyperlink data and to repair broken links, for example, after a site move or a domain name change. Replacement Strings may include:

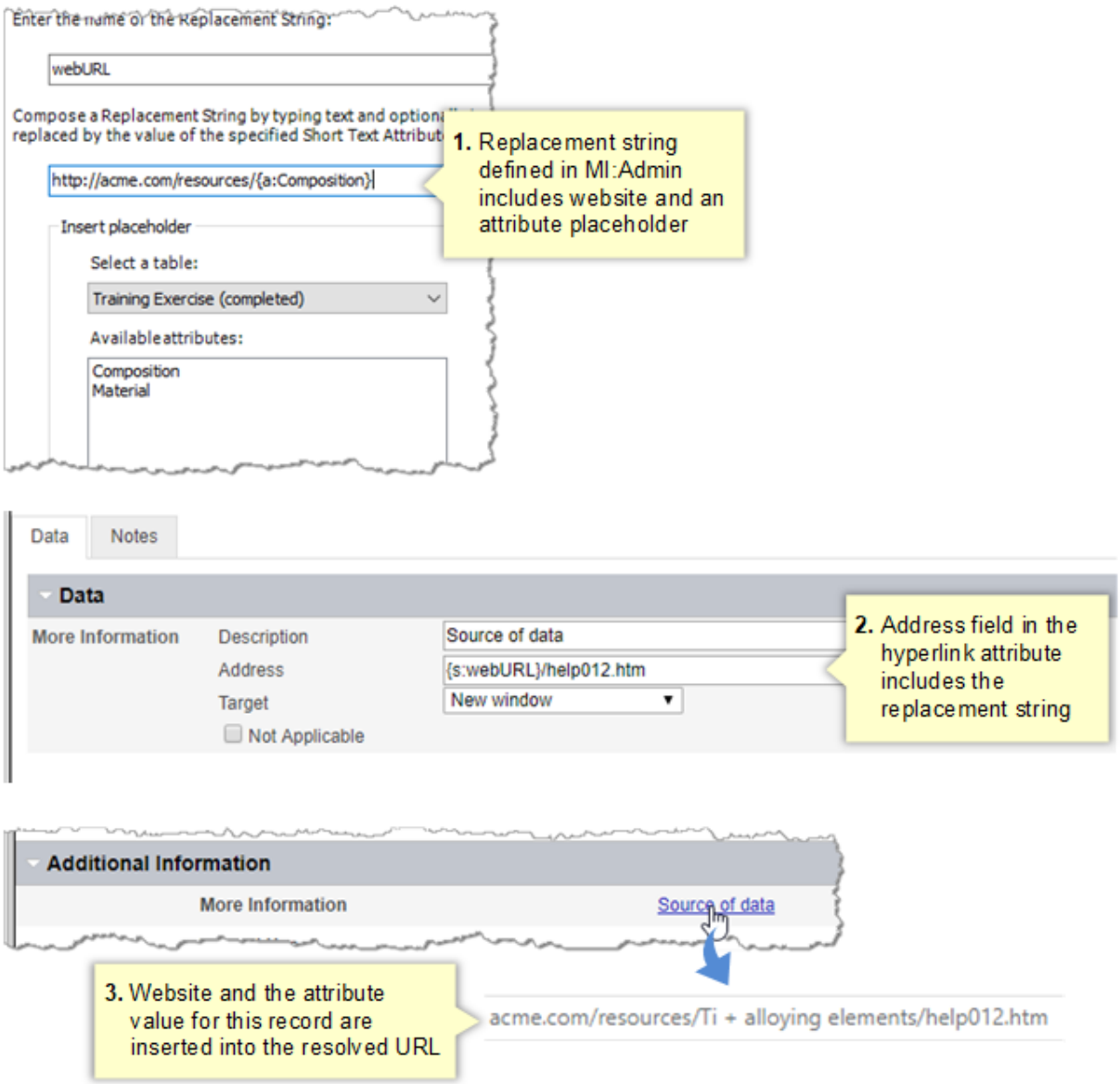
- Text, for example, a protocol, domain name, hostname, and/or path.
- Placeholders that allow data from Short Text Attributes in the same record to be dynamically inserted into the address when a user clicks on the hyperlink in a datasheet and the URL is resolved.

Replacement Strings are database-level objects defined in MI:Admin, and are referenced in the Address field of Hyperlink Attributes using the syntax: {s:*replacement_string_name*}

Example: in the *More Information* Hyperlink Attribute shown here, the web server address part of the Address field is specified using the Replacement String called *webURL*, which is defined in MI:Admin as **http://acme.com**. When the link in the datasheet is clicked, the URL is resolved as shown:



In the example below, the *webURL* Replacement String includes a placeholder for the Short Text Attribute *Composition*, as well as a protocol, domain, and path:



18.1 Viewing and managing Replacement Strings

To view and manage the Replacement Strings in a database: in the MI:Admin Schema tool, select the database and then click on the **Edit Replacement Strings** link.

18.2 Properties

Replacement Strings in GRANTA MI have the following properties.

18.2.1 Name

Maximum 255 characters. Note that if you change the name of an existing Replacement String, the name will not be updated anywhere where the original name is already referenced in existing data values.

18.2.2 Value

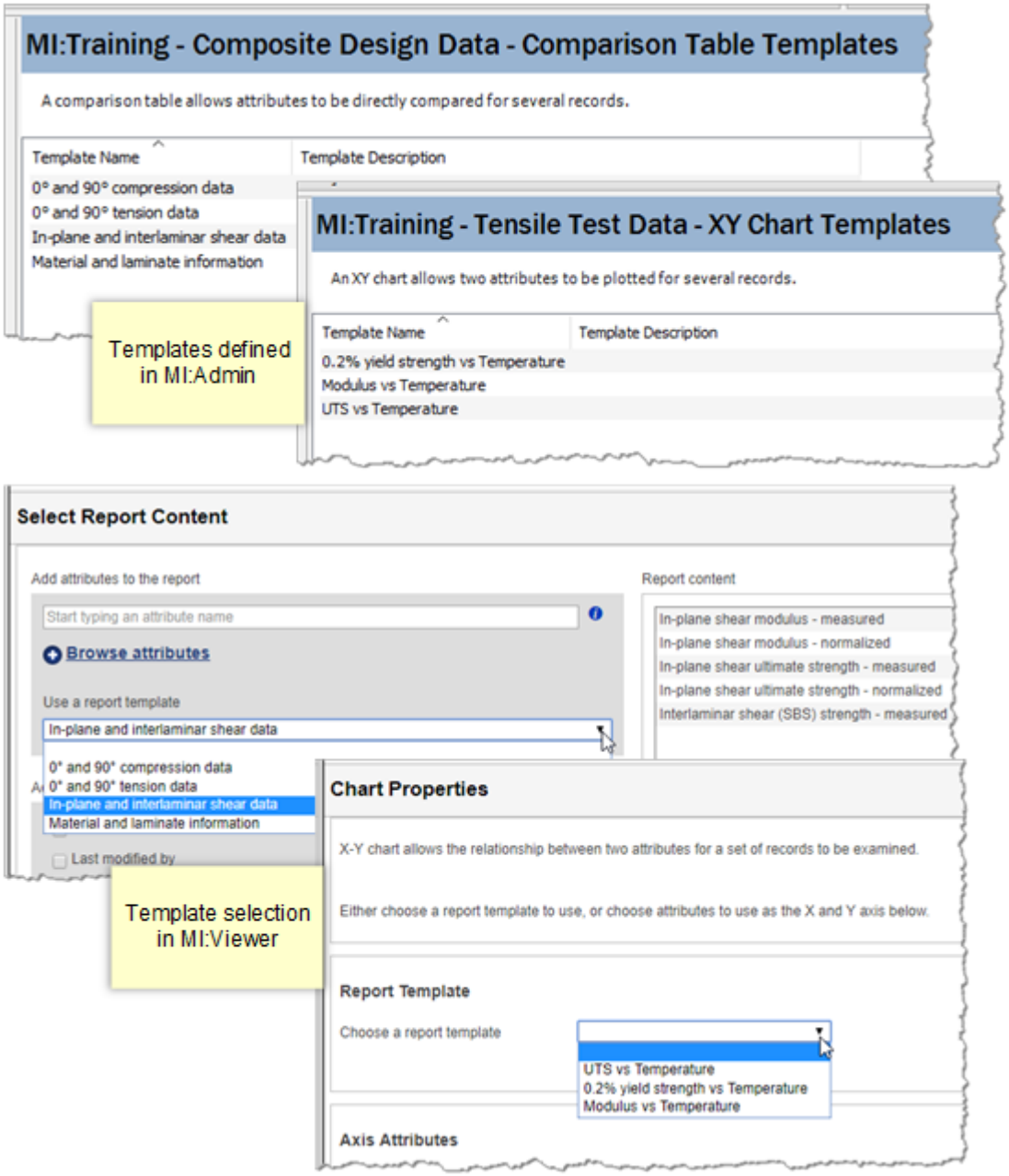
A string that includes some/all elements of a URL (e.g. protocol and host, plus path...). To include Attribute placeholders within the string, for example, in the path, under **Insert placeholder**, select a table and then double-click the Attribute name in the list.

Replacement string examples:

```
http://mycompany.com/library/index.htm
https://www.myorg.com/projects/{a:Project Code}/data/
http://acmedesign.local.com/resources/{a:Manufacturer}/datasheet.htm
```

19 Report Templates

Report templates provide a quick way for MI:Viewer users to select Attributes for a Comparison Table or X-Y Chart report, saving them from having to find and add individual Attributes one-by-one.



19.1 Viewing and managing Report Templates

Report templates are defined per-table in the MI:Admin Schema tool: click to expand the table in the **Tables** pane in the lower left of the Schema tool, and then click on Report Templates.

19.2 Comparison Table Report template properties

Comparison Table Report Templates in GRANTA MI have the following properties.

19.2.1 Name

The template name may be up to 255 characters long. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Summary Report" and "Summary report").

19.2.2 Description

An optional description of the template. The name and description both appear in the template dropdown list on the Select Report Content page in MI:Viewer.

19.2.3 Attributes

Choose the Attributes to include in the report from the list on the left and add them to the **Chosen Attributes** list on the right.

Use the **Move Up** and **Move Down** buttons to specify the default order of the Attributes in the report. Users will be able add more Attributes, remove Attributes, and reorder them in MI:Viewer.

To sort the records in the Comparison Table on one of the Attributes in the table, click **Sort by Attribute**, select the Attribute from the list, and specify whether to sort by ascending or descending values.

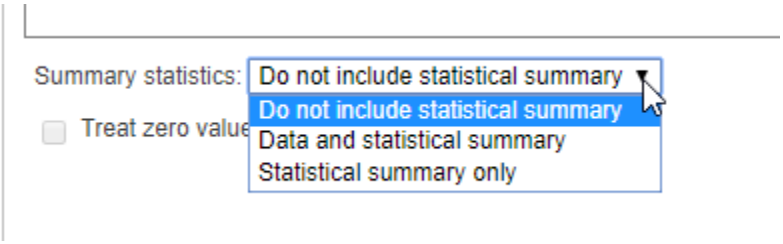
19.2.4 Parameters

You can optionally set the parameter values used for Functional data in Comparison Table reports based on the template.

By default, when users create a Comparison Table that includes functional data, the parameter default values are used, and users can pick a different value for the report if they wish via a **Tools > Change parameters** option on the Comparison Table Report page in MI:Viewer. However, you can set up the template to specify the parameter values to use for functional data; to do this, click **Set Parameters** and specify the required values. This also has the effect of removing the **Tools > Change parameters** option from the Comparison Table Report page in MI:Viewer, preventing users from changing the parameter values used in the report.

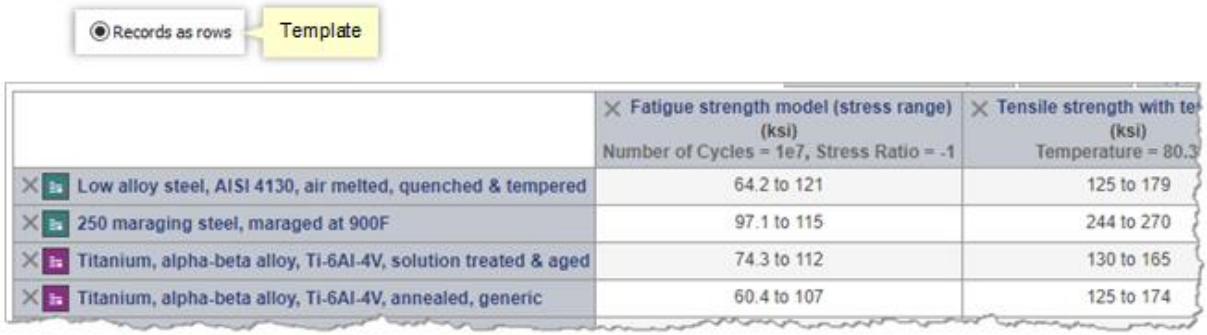
19.2.5 Statistical Summary options

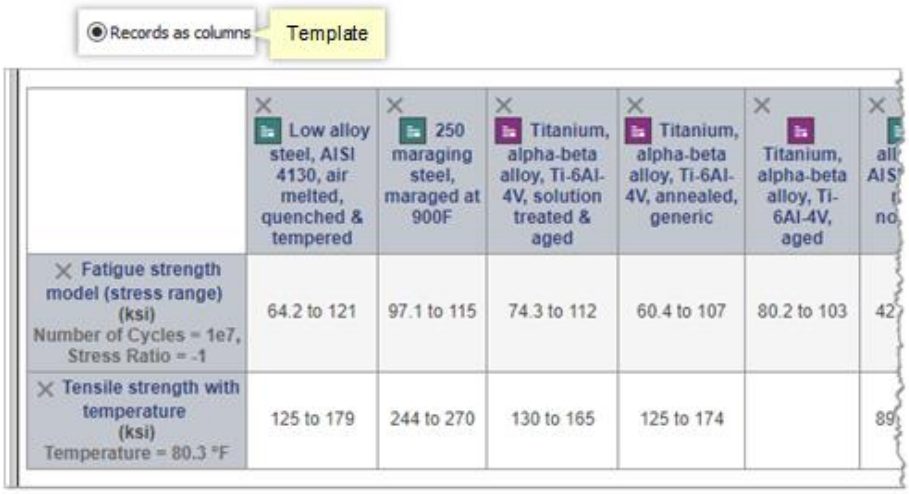
Specify how statistical summary data will be displayed in the report by default; MI:Viewer users will be able to change this before generating the report, for example:



19.2.6 Records as rows/Records as columns

By default, the rows in the Comparison Table contains records, and the column contain Attributes. Selecting the **Records as columns** option here will transpose this, so when the user views the Comparison table on screen, the rows will be Attributes and the columns will be records, for example::





However, note that this will only affect the data when viewed on screen: when saving the report data to the clipboard, to a CSV file, or to an Excel report, records are always shown as rows.

19.3 X-Y Chart template properties

X-Y Chart Templates in GRANTA MI have the following properties.

19.3.1 Name

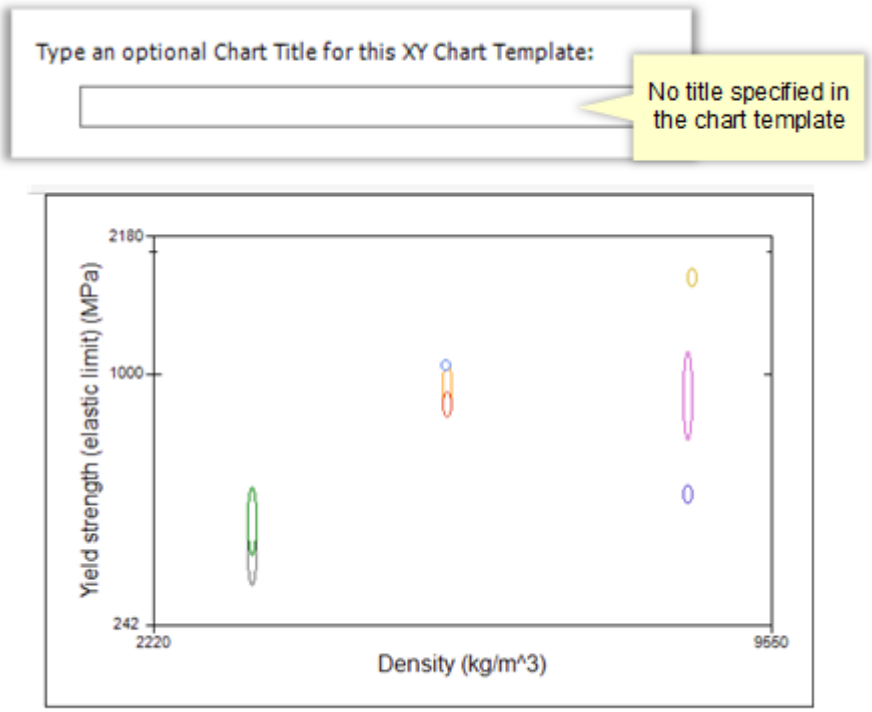
The template name may be up to 255 characters long. The name must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Young's Modulus vs Temperature" and "Young's Modulus vs temperature").

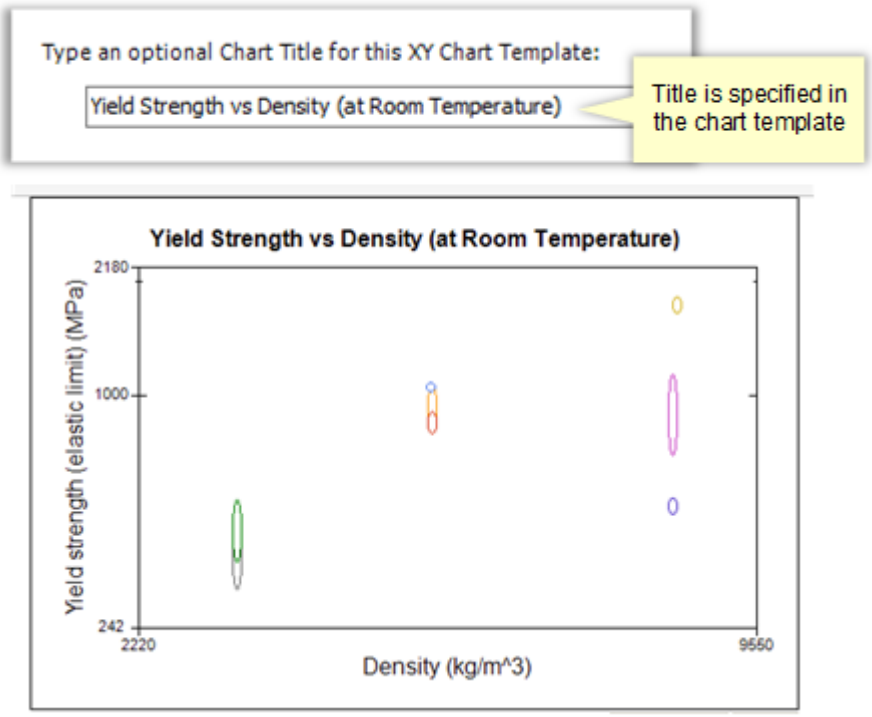
19.3.2 Description

An optional description of the template. The name and description both appear in the template dropdown list in MI:Viewer.

19.3.3 Chart Title

You can optionally specify the chart title. If not specified, the chart will not have any title. For example:





19.3.4 X and Y axis properties

Specify the X and Y axis Attributes and set the axis scale and range properties.

19.3.5 Parameters

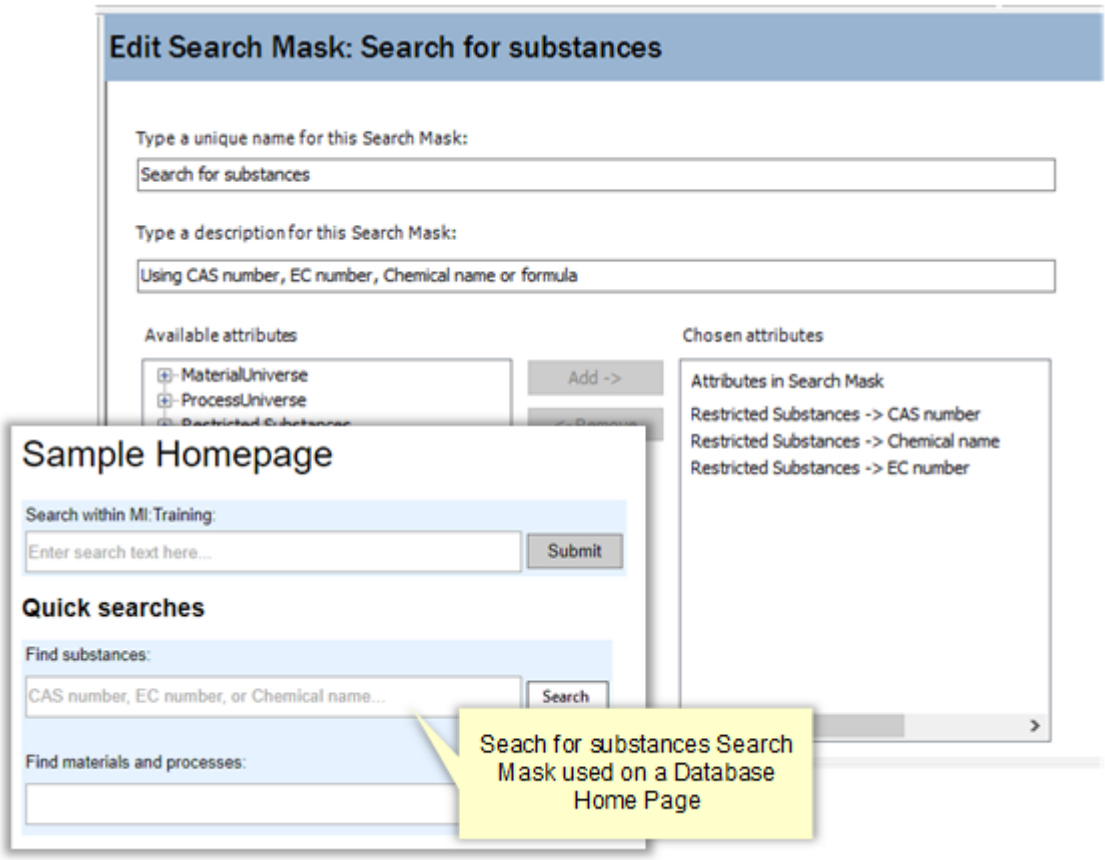
You can optionally set the parameter values used for functional data in X-Y charts based on the template.

By default, when users create a chart that includes functional data, the parameter default values are used, and users can pick a different value for the report if they wish via a **Tools > Change parameters** option on the X-Y Chart page in MI:Viewer. However, you can set up the template to specify the parameter values to use for functional data; to do this, click **Set Parameters** and specify the required values. This also has the effect of removing the **Tools > Change parameters** option from the X-Y Chart page in MI:Viewer, preventing users from changing the parameter values used in the chart.

20 Search Masks

Search Masks are text search filters for home pages. They can be used to provide quick search functionality that's limited to a specific set of text Attributes in a specific table. In searches based on Search Masks:

- Data in the specified Short Text, Long Text, and Discrete Attributes is searched; link notes are also searched.
- By default, record names are not searched. This can be configured by modifying the Search setting IncludeRecordNamesWhenUsingSearchMask in the MIServer.exe.config application configuration file.



For information on how to configure application, Profile, and database home pages to use Search Masks, as shown in the example above, see the *MI:Viewer Home Page Author Guide*.

20.1 Viewing and managing Search Masks

To view and manage Search Masks in GRANTA MI: in the MI:Admin Schema tool, select the database and then click on the **Edit Search Masks** link.

20.2 Properties

Search Masks in GRANTA MI have the following properties.

20.2.1 Name

This may be up to 255 characters long. The name entered here must be unique within the database; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same database (e.g. "Search for Substances" and "Search for substances").

20.2.2 Description

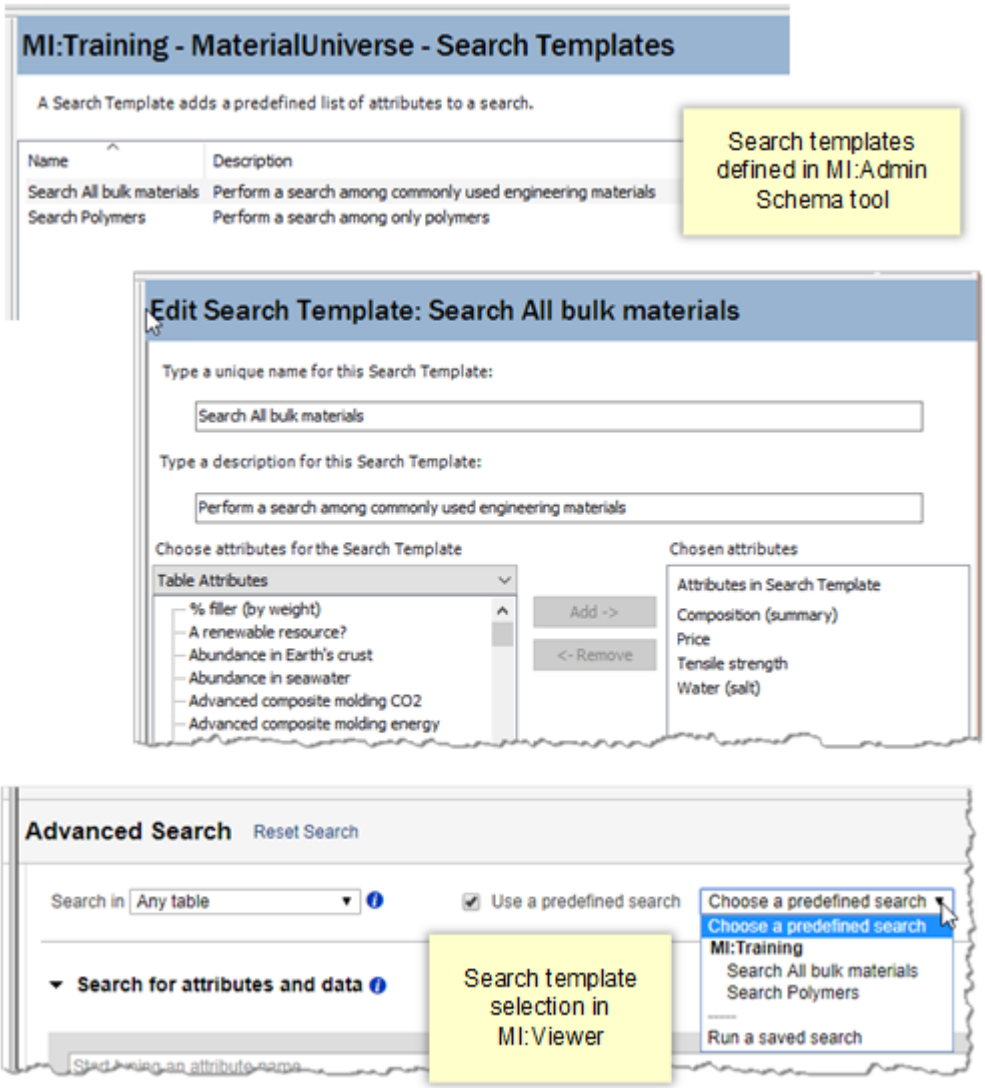
An optional description of the template. This only appears in MI:Admin.

20.2.3 Chosen attributes

Choose the Attributes to include in the Search Mask by selecting them from the list on the left and adding them to the **Chosen Attributes** list on the right.

21 Search Templates

Search Templates define a collection of Attributes for Attribute-based search queries in MI:Viewer. When a search template is selected in MI:Viewer on the Advanced Search page, all of the specified Attributes are added as search criteria in a single click, saving the user from having to find and add each Attribute to the query individually. Users can then remove individual Attribute criteria from their search or add more Attributes as required, and also specify search thresholds and parameter values, before running their search.



21.1 View and manage Search Templates

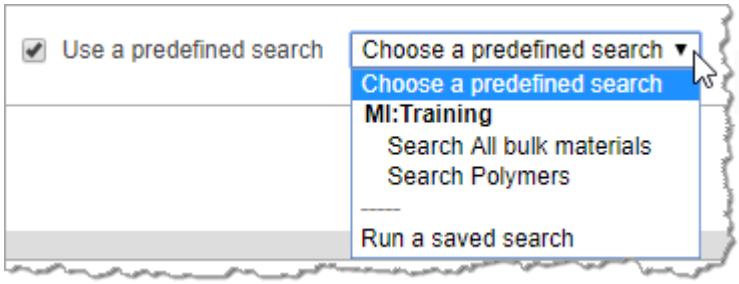
Search Templates are defined per-table. You can view and manage the Search Templates currently defined in a table on **the Search Templates** page in the MI:Admin Schema tool. Expand the table in the Tables pane and then click on Search Templates. Click on a column header to sort the list.

21.2 Search Template properties

Search Templates in GRANTA MI have the following properties.

21.2.1 Name

This may be up to 255 characters long. The name entered here must be unique within the Table; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "Search polymers" and "Search Polymers"). The name appears in the list of predefined searches on the MI:Viewer Advanced Search page, for example:



21.2.2 Description

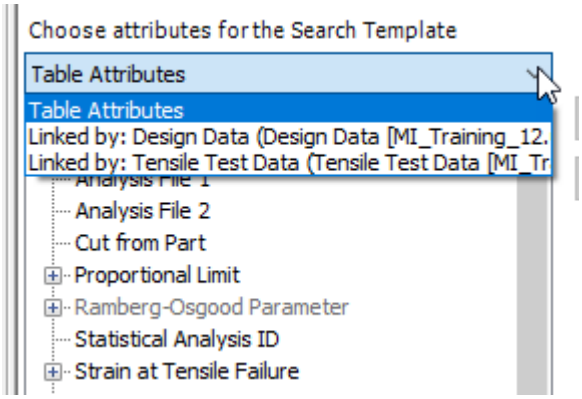
(Optional) This appears only in MI:Admin on the Search Templates page.

21.2.3 Chosen attributes

The template can include Attributes and Meta-attributes from within the same Table as the template, or from other Tables that are linked via static Record Link Groups.

Select **Table Attributes** above the list to choose Attributes in the same Table as the template.

If the Table includes any Static Record Link Groups, Attributes from the linked Tables may be included in the template by choosing the Record Link Group from the menu above the list. Selecting a *Linked by* item from this menu will show the Attributes in the linked Table. For example, this Search Template in the *Tensile Test Data* Table may include Attributes from other Tables via the *Design Data* or *Tensile Test Data* Record Link Groups:



22 Standard Names

A GRANTA MI data schema may include Standard Names that are used to uniquely identify properties, parameters, or relationships in the database. Standard Names can provide a way of handling diversity in terminology, where multiple synonyms are used for the same physical or real-life property:

> ***Young's Modulus***: E, Elastic Modulus, Modulus of Elasticity, Stretch Modulus, Monotonic Modulus, Static Modulus, Tensile Modulus, Coefficient of Elasticity, Modulus in Tension, Deflection Resistance, Extensional Modulus

Standard names in GRANTA MI are database-level objects and may be mapped to Attributes, Parameters, and Record Link Groups in the same database. For example, the MI:Training database includes the following standard name mappings:

Standard name	Mapped to these Attributes
Poisson's ratio	***0° tension Poisson's ratio - normalized*** in the Composite Design Data table ***Poisson's Ratio, L*** in the Design Data table ***Poisson's ratio*** in the MaterialUniverse table
Tensile strength	***0° tension strength - normalized*** in the Composite Design Data table ***Tensile strength*** in the MaterialUniverse table
Tensile strength, yield with temperature	***Tens. Yield Stress (L-dir) with Temp.*** in the Design Data table ***Yield strength with temperature*** in the MaterialUniverse table
Tensile modulus	***Modulus, L*** in the Design Data table ***Young's modulus*** in the MaterialUniverse table
Young's modulus	***0° tension modulus - normalized*** in the Composite Design Data table ***Young's Modulus*** in the MaterialUniverse table

Standard names are often used when configuring a database for use with the Service Layer and Reports, or when setting up fallback links between a customer's own GRANTA MI database and a data module supplied by Granta Design.

22.1 Viewing and managing Standard Names

Click on the **Edit Standard Names** link on the left of the Schema tool. The list shows each standard name, and the Attributes, parameters, and Record Link Groups to which it is currently mapped.

22.2 Properties

Standard Names in GRANTA MI have the following properties.

22.2.1 Name

This may be up to 255 characters long. The name entered here must be unique within the database; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same database (e.g. "Chemical name" and "Chemical Name").

22.2.2 Mapped Attributes

The Standard Name can be used to reference Attributes in any of the Tables in the database.

22.2.3 Mapped Parameters

The Standard Name can be used to reference any Parameters defined in the database.

22.2.4 Mapped Record Link Groups

The Standard Name can be used to reference any static Record Link Groups defined in the database (both forward and/or reverse direction links).

23 Units and Unit Systems

23.1 Units

Units define the unit of measurement in which numeric data is stored in the database. GRANTA MI database units may be defined as:

- Base units, for example **metre**, **kilogram**, **mole**, **second**, **strain**, **Ampere**, **Temperature (Kelvin/Celsius)**. These are units which are not derived from any other units.
- Units that are a multiple of another unit, for example, **inch**=0.0254*[m], **foot**=12*[in], **angstrom**=1e-010*[m], **minute**=60*[s]
- Derived units that are defined in terms of other units via an equation, for example:
  - **Watt** is defined as: [J]/[s]
  - **pascal** is defined as: [N]/([m]^2)
  - **% strain/°C** is defined as: [%strain]/[K]
  - **BTU/hr.ft.°F** is defined as: [BTU]*[ft]/([hr]*[ft]^2*[°R])

  Unit conversion equations can include numbers, unit symbols enclosed in square brackets [ ], parentheses for nesting, and the operators / (divide by), * (multiply by), and ^ (raise to the power of).

Attributes, Meta-attributes, parameters, Constants, and expressions can all have units. All numeric data is stored in the database unit assigned to the Attribute but data can be displayed in units other than the database unit via a Unit System which converts the underlying data value into the equivalent value with a different unit.

23.2 Unit Systems

Unit Systems in GRANTA MI allow numeric values to be converted to units other than the underlying database unit when the data is displayed, edited, or exported. GRANTA MI databases include a number of common unit systems like *Metric*, *US Customary*, *UK Imperial*, and *SI (Consistent)*.

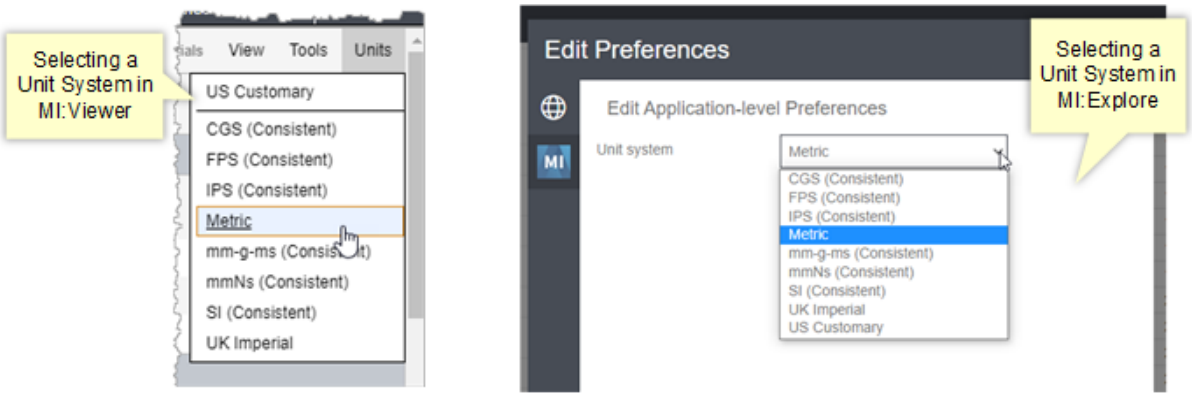
Some units are common across all Unit Systems, for example, the volt (V); all Attributes measured by voltage are stored in this unit, and there is no need to convert the data values for display or export in any Unit System. In general, however, for each unit in the database, it is necessary to specify its ***equivalent unit*** in each Unit System in the database so that values are converted to the appropriate units in that system.

Examples of Unit System equivalents defined in the MI:Training database:

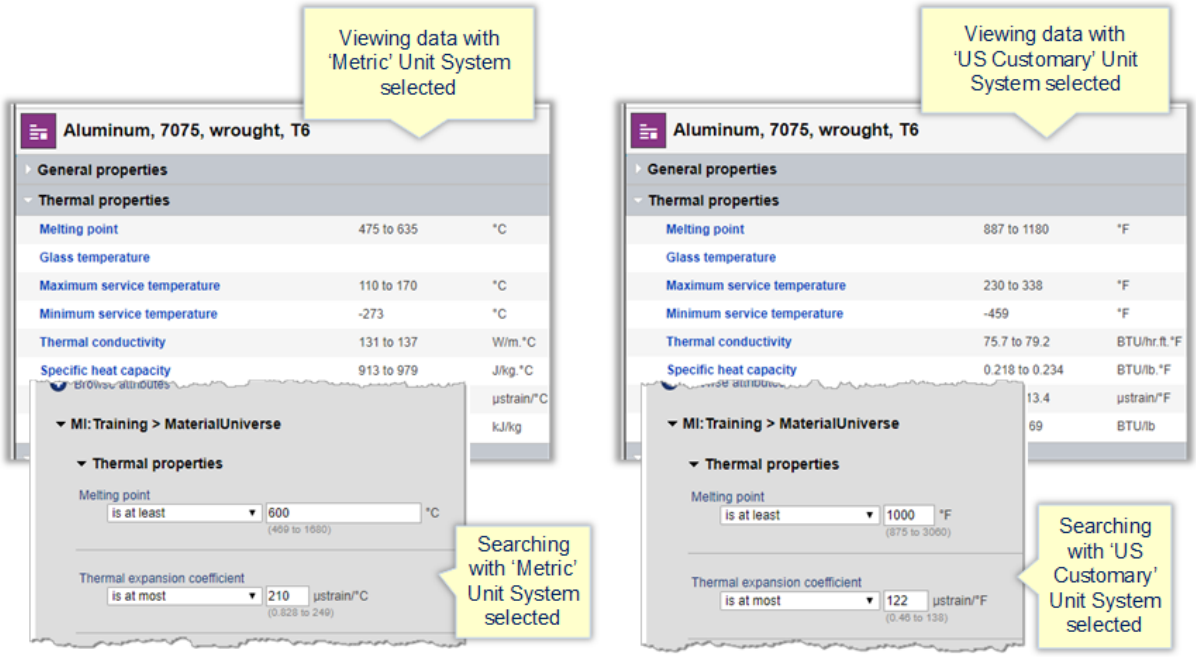
Measure	Database unit name	Unit	Metric equivalent	SI equivalent	CGS (Consistent) equivalent	US Customary equivalent
Length	metre	m	1 m	1 m	100 cm	3.2808 ft
Weight	newton	N	1 N	1 N	100000 dyn	0.2248 lbf

Measure	Database unit name	Unit	Metric equivalent	SI equivalent	CGS (Consistent) equivalent	US Customary equivalent
Temperature	Temperature (Kelvin/Celsius)	K	1 K	1 K	1 K	1.8 °R
Pressure	megapascal	MPa	1 MPa	1e6 Pa	1e7 dyn/cm^2	0.14504 ksi
Voltage	volt	V	1 V	1 V	1 V	1 V

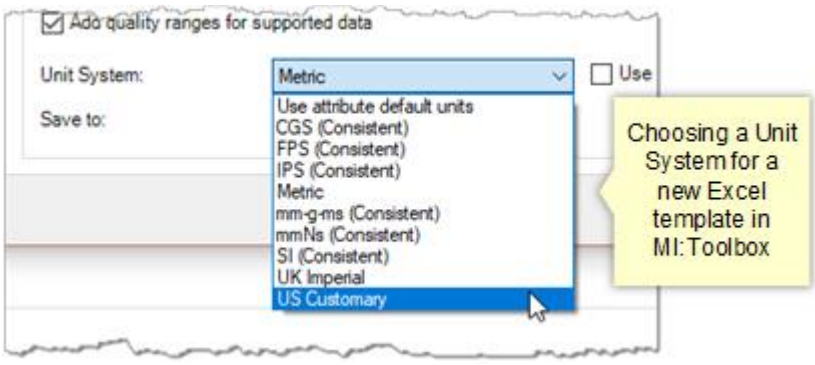
In MI:Viewer and MI:Explore, users can choose the Unit System to use for viewing and editing data; this determines the units used throughout the application to display and enter numerical values; any data entered is converted to the underlying database unit when it is saved in the database.



Example from MI:Viewer showing effect of changing the Unit System on viewing and searching numerical data:



When generating Excel export templates in MI:Toolbox, it is possible to specify a Unit System which will be used to convert underlying numeric values from Database units to the required equivalent values/units in the export file.



All GRANTA MI databases contain a range of standard Unit Systems. You should normally never need to create a new Unit System and we strongly recommend that wherever possible, you should use the standard Granta Unit Systems. If you do need to define a new one, we recommend that you do so by copying and pasting an existing Unit System that is as similar as possible to the one you want to create, and then editing it as required; for each unit in the database whose equivalent unit differs between the original and new Unit Systems, you will then need to update the equivalent unit in the new system.

23.3 Currency units

GRANTA MI databases have a default currency, specified as a database property. All currency-related data is converted to the database currency before being stored in the database.

The *currency* unit may be included in unit expressions as [currency], for example:

- currency (e.g. Capital cost, Tooling cost)
- [currency]/[m]^3 (e.g. Cost of consumables)
- [currency]/[kg] (e.g. Price)

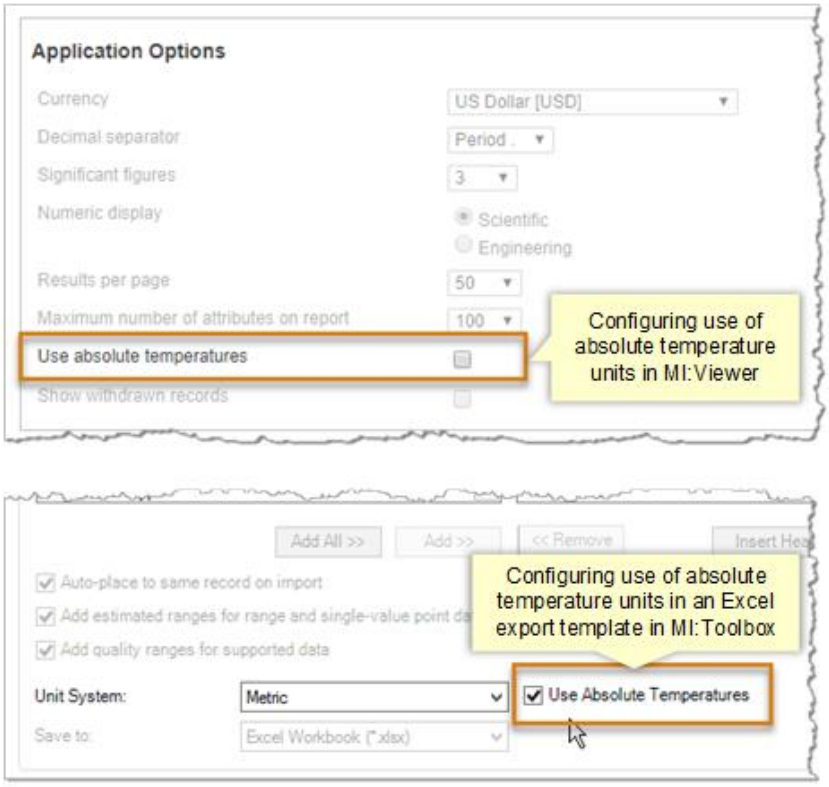
23.4 Temperature units

Temperature data is stored in an absolute unit, **Kelvins** or **degrees Rankine**; temperature data can be displayed or exported using the **absolute unit** or a **display unit**.

Data stored in	Absolute unit	Display unit
Kelvin	K	degrees Centigrade (°C)
Rankine	R	degrees Fahrenheit (°F)

Absolute units are always used when a temperature unit is combined in a unit expression.

Use of absolute units for display and export of temperature values is a configuration option in MI:Viewer and in the Excel export templates generated in MI:Toolbox:



23.5 Viewing and managing Units

You can view the units that are currently defined in a GRANTA MI database, and add, remove, and edit units (for instance, to update equivalent units in Unit Systems) on the **Units** page in the MI:Admin Schema tool.

23.6 Unit Properties

Units in GRANTA MI have the following properties.

23.6.1 Name

This may be up to 255 characters long. The Unit name must be unique within the database; it is not treated as case-sensitive and so the same name with different cases is not permitted within the same Table (e.g. "kilogram" and "Kilogram").

23.6.2 Symbol

The Unit symbol must be unique within the database. If you change the symbol for an existing unit, the symbol will be changed in all Attributes and unit conversions where it is currently used.

23.6.3 Unit Conversion

A Unit may be derived from other Units in the database, either as a simple multiple of another Unit, or via an expression which may include multiple other Units. Three unit conversion options are provided in the MI:Admin Schema tool:

- **No conversion**—the Unit is not derived from any other Units. For example, Ampere, metre, Radians.
- **Direct conversion**—the Unit is a multiple of a single existing Unit, for example: mm = 0.001*[m]; Msi = 1*(1000000*[psi]); megaWatt = 1e6*[W]. The factor may be an integer for multiplication (e.g., 1000) or a decimal number for division (e.g., 0.1).
- **Derived from other units**—the Unit is a product of one or more other Units via an expression that may include combinations of multiplications (*), divisions (/), and raise to the power of (^) exponents. For example: MJ/kg=1e6*[J]/[kg]; microstrain/°C = [µstrain]/[K]; newton = [kg]*[m]/([s]^2).

23.6.4 Unit System Equivalents

This specifies the equivalent Units in each of the Unit Systems in the database. For example, in the MI:Training database, the *metre* Unit is defined with the following Unit System Equivalents:

Unit System	Conversion	Equivalent Unit
CGS (Consistent)	100	cm
FPS (Consistent)	3.2808	ft
IPS (Consistent)	39.37	in
Metric	1	m
mm-g-ms (Consistent)	1000	mm
mmNs (Consistent)	1000	mm
SI (Consistent)	1	m
UK Imperial	3.2808	ft
US Customary	3.2808	ft

:

23.7 Copying and pasting unit definitions from the clipboard

Unit definitions can be copied into and out of MI:Admin via the clipboard, allowing you to document your database schema, as well as define and modify it outside of MI:Admin, if needed.

To copy Unit definitions to the clipboard, on the Units page (**Schema > Edit Units**), select the Units in the list and then click **Copy** on the right-click (shortcut) menu, or press CTRL+C. The unit data is copied to the clipboard, and can then be pasted into a spreadsheet or a text file.

To paste new Unit definitions into MI:Admin from an Excel or text file, select the data in the worksheet or text file and copy it to the clipboard, then paste it into the Units page in MI:Admin.

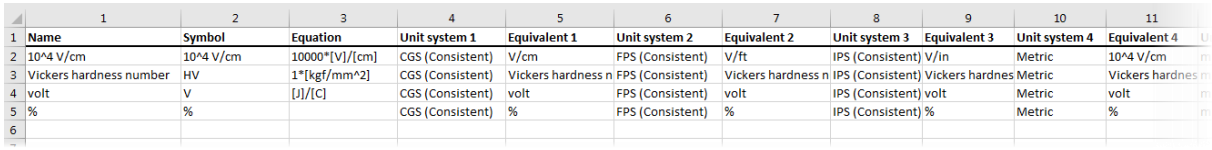
Each Unit definition must be on a separate row. Header rows included for clarity in your spreadsheet e.g. Row 1 in the Excel example below, should not be copied.

Column 1	Column 2	Column 3
Unit Name	Unit symbol	Equation, for derived Units

Optionally, additional pairs of columns may be included for Unit equivalents in different Unit Systems, with one column pair for each equivalent (Unit System name, and Equivalent Unit name)

- Equivalents do not need to be defined for all Unit Systems.
- You can have different number of Unit equivalents on different rows.
- Unit equivalents do not need to be defined for all Units.

This Microsoft Excel worksheet example shows unit definitions for 4 units, with equivalents in 4 different unit systems:



	1	2	3	4	5	6	7	8	9	10	11
1	**Name**	**Symbol**	**Equation**	**Unit system 1**	**Equivalent 1**	**Unit system 2**	**Equivalent 2**	**Unit system 3**	**Equivalent 3**	**Unit system 4**	**Equivalent 4**
2	10^4 V/cm	10^4 V/cm	10000*[V]/[cm]	CGS (Consistent)	V/cm	FPS (Consistent)	V/ft	IPS (Consistent)	V/in	Metric	10^4 V/cm
3	Vickers hardness number	HV	1*[kgf/mm^2]	CGS (Consistent)	Vickers hardness n	FPS (Consistent)	Vickers hardness n	IPS (Consistent)	Vickers hardnes	Metric	Vickers hardnes
4	volt	V	[J]/[C]	CGS (Consistent)	volt	FPS (Consistent)	volt	IPS (Consistent)	volt	Metric	volt
5	%	%		CGS (Consistent)	%	FPS (Consistent)	%	IPS (Consistent)	%	Metric	%
6											

24 Database properties

The MI:Admin tool may be used to set/modify a number of properties for GRANTA MI databases: in the MI:Admin Schema tool, select the database then click on the Edit Database link.

24.1 Database properties

24.1.1 Database name

The name displayed in GRANTA MI applications. It must be unique, is limited to 255 characters, and may not start or end with a space character.

24.1.2 Default Table

Specifies the table into which data will be imported when doing a command-line import if no table is specified in the command-line arguments.

24.1.3 Database currency

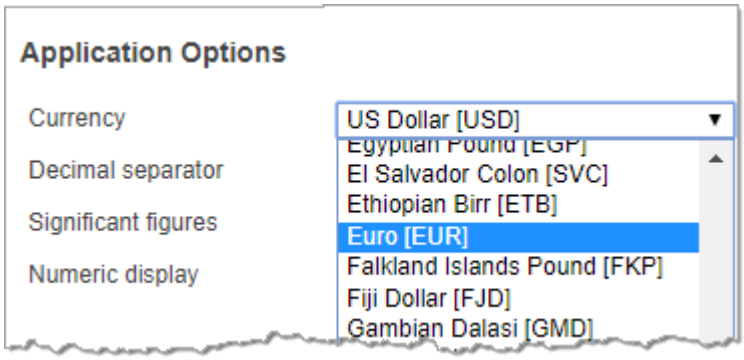
Currency values are stored in the GRANTA MI base currency, US Dollars (USD).

The **Database currency** property allows users to work in a currency that is different from the base currency when viewing and modifying monetary data such as material price.

You can choose any system currency as the Database currency. Monetary data added to the database will be stored in the system base currency, with the exchange rates specified in the currency definition used to convert values (see System currencies).

The Database currency may be used as a "variable" in database unit definitions, for example, *currency/kg*, *currency/lb*, *currency/hr*; see Units and Unit Systems.

In MI:Viewer, users can choose the currency they want to work in, which may be different from the database currency (**Settings>Options>Application Options>Currency)**; changing the currency in the application has no effect on the currency in which the data is stored.



A blank GRANTA MI database is created with a default **Database currency** setting, but this can be changed prior to adding data. We strongly recommend that you do not change the **Database currency** setting in a database that already contains currency data; if you do, any existing currency

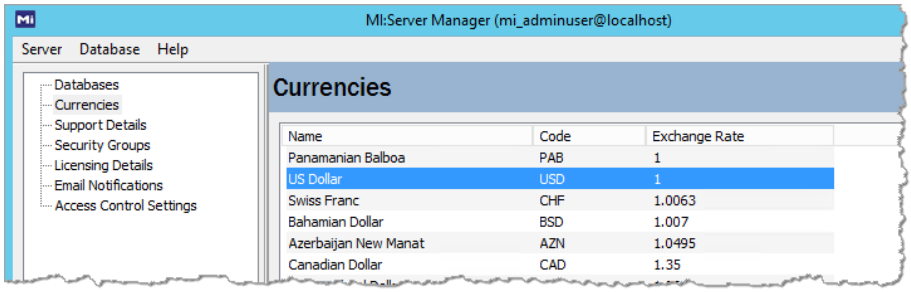
data values that use a unit that refers to 'currency' (for example, *currency/kg)* will not be converted to the new database currency but will simply be assigned the new unit, for example, *7 USD/kg* will simply become *7 GBP/kg*.

24.1.4 Author, Company, Notes

Optional. *Author* and *Company* are limited to 255 characters; *Notes* can be up to 1048576 characters.

25 System currencies

The base currency in GRANTA MI is the US Dollar (USD). System currencies are defined in the GRANTA MI configuration database, and may be modified/updated in the MI:Server Manager tool. The system currency definitions specify the currency name, code, and an exchange rate which is used to convert values between different currencies.



Each database may have a Database currency specified, which may be different from the system base currency. The exchange rates specified in the system currency definitions are used to convert monetary data between the base currency and the database currency.

MI:Viewer users can also choose their preferred currency for viewing and editing currency data within the application; again, the exchange rates specified in the system currency definitions are used to perform the required currency conversion.

25.1 Properties

System currency definitions have the following properties.

25.1.1 Name

Description of the currency.

25.1.2 Code

A unique, three letter ISO 4217 currency code.

25.1.3 Exchange rate

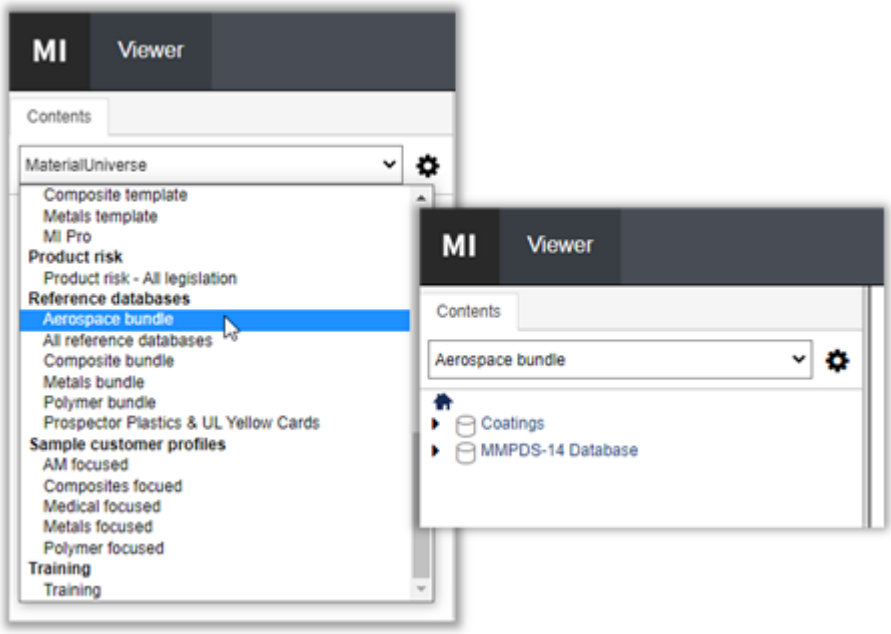
The exchange rate used to convert values to this currency from the system base currency (US Dollar). A maximum of 15 significant figures is allowed. For example:

Name	Code	Exchange Rate
US Dollar	USD	1
Australian Dollar	AUD	1.45
British Pound	GBP	0.786639
Canadian Dollar	CAD	1.345
Euro	EUR	0.893280

26 Profiles

Profiles provide a way of grouping together individual databases and tables for particular audiences in MI:Viewer; custom profiles can also be used to provide a quick way of specifying a set of Records and Attributes to include in a Data Update.

In MI:Viewer, the selected Profile determines which databases and Tables are available for browsing and searching within the application, for example:



(Users with system security Grant or Admin privileges can see an additional 'Everything' Profile that shows all available databases and Profiles.)

There are two types of Profile:

- A **Standalone** database Profile consists of an individual database and its home page. The Profile will display all the tables in the database, in database order. You cannot edit a standalone database Profile to modify the tables it displays or the order in which they are shown.
- A **Custom** database Profile can show tables from one or more databases, in whatever order you specify. You can set a preferred Subset and Layout for each table in the Profile. (MI:Viewer users will still be able to change Subset and Layout when using a custom Profile, but they will be warned that they are not using the preferred settings.)

When a database is added to the GRANTA MI system, it automatically has a standalone database Profile. Users with System security Admin privileges can enable and disable individual standalone database Profiles, and manage custom Profiles in the Profiles tool.

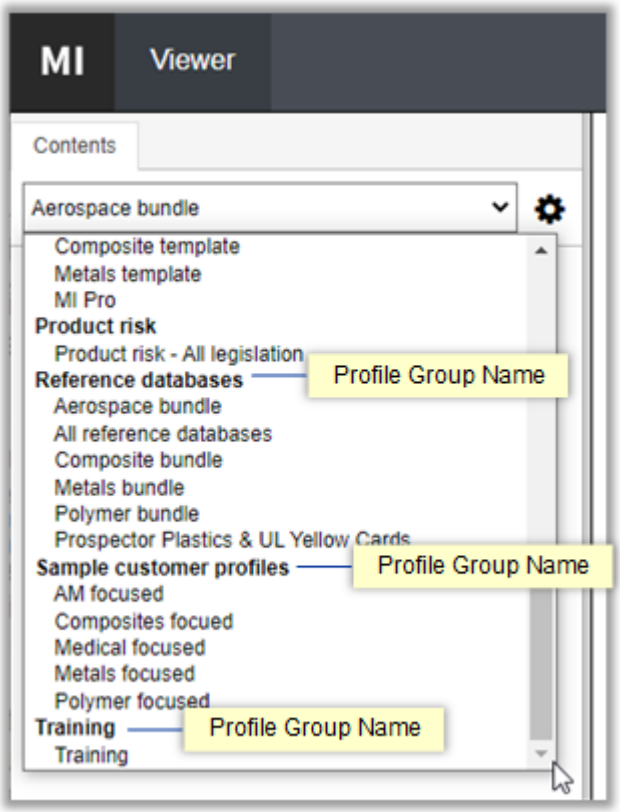
26.1 Profile properties

26.1.1 Profile Name

The Profile name should not contain the ampersand (&) character, or any other character that requires special URL encoding for a web browser.

26.1.2 Group Name

If required, you can group Profiles together by specifying a Group Name as a Profile property; all of Profiles with the same Group name will appear below a Group name heading in the Profile selector in MI:Viewer, for example:

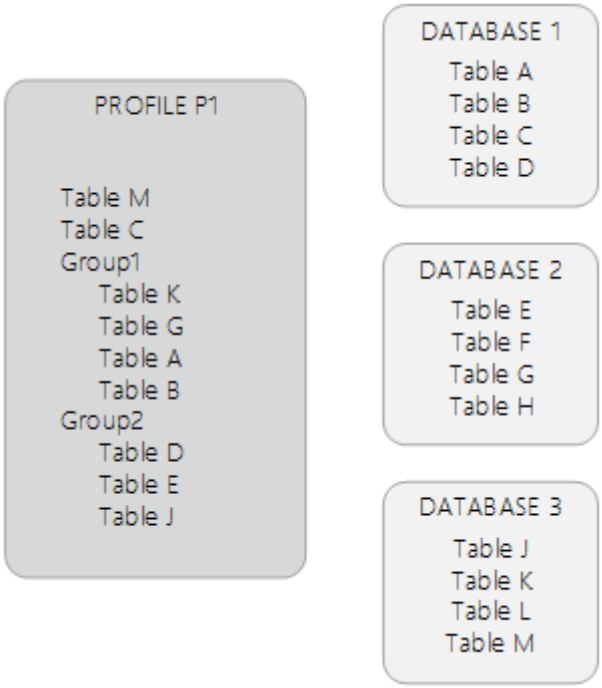


26.1.3 Description

An optional description of the Profile that appears on the default Profile home page in MI:Viewer.

26.1.4 Tables

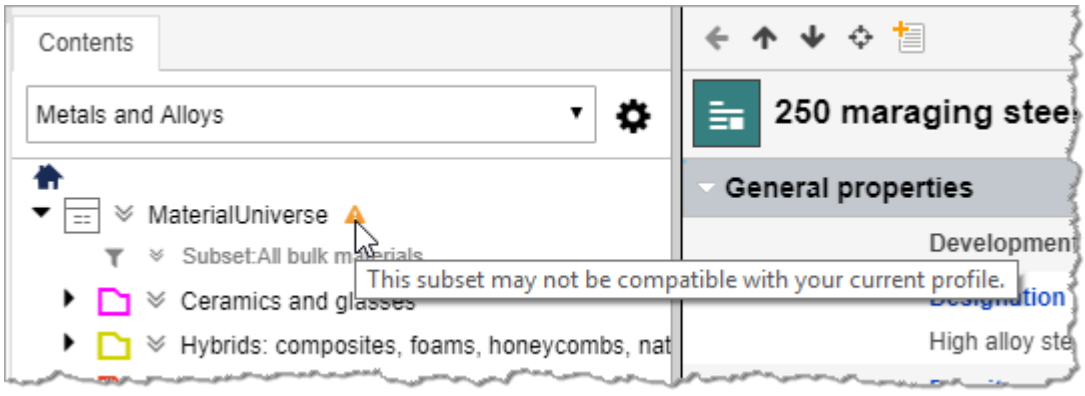
The Profile may include Tables from any of the databases in your GRANTA MI system, in any order. For example:



26.1.5 Preferred Subset and Layout

If required, you can associate a specific Subset and Layout with each Table in the Profile. Where a preferred Subset and Layout have been specified, when MI:Viewer users select the Table from the Profiles list, the preferred Subset and Layout will also be selected; users will still be able to switch to a different Subset and/or Layout, but will see a notification that they are not using the preferred settings.

For example, the preferred Subset for the *MaterialUniverse* Table in this *Metals and Alloys* profile is the *Metals* Subset; when the user selects a different Subset in the Contents tree, a small warning triangle appears next to the Table name:

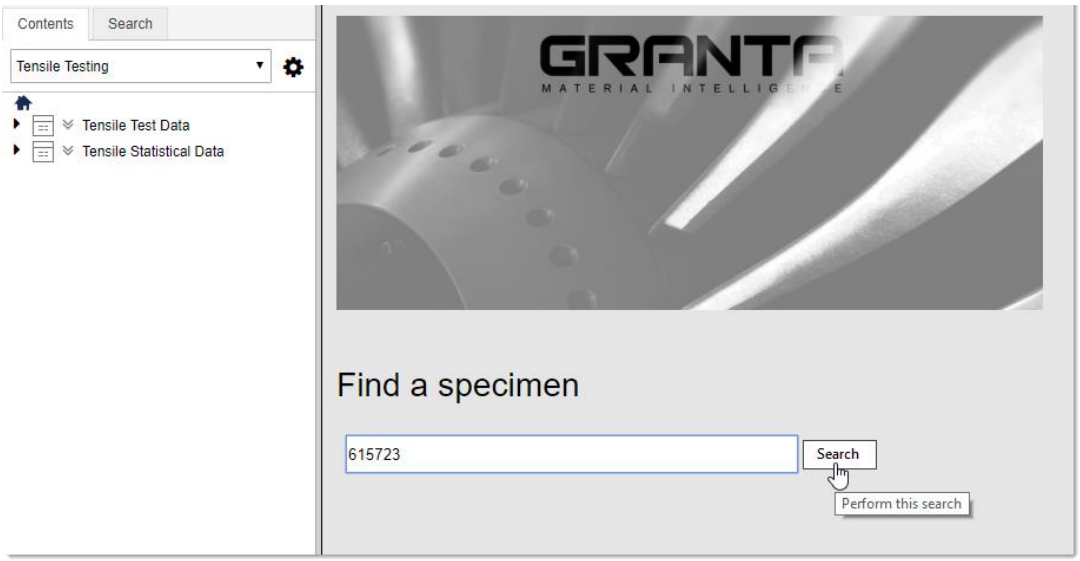


26.2 Profile home pages

A Profile home page is an HTML or ASPX file that may include some or all of the following:

- text and images, such as a description of the databases and tables included in the Profile, or your company logo,
- links to applications or tools, or to other web pages
- controls to perform text searches
- any other valid HTML, or ASPX elements such as form or script tags.

For example this Tensile Testing profile home page includes an image and a search form:



You can create HTML or ASPX Home pages for your Custom Profiles outside of GRANTA MI using the editor of your choice, and then add them to the Profile in the MI:Admin Profiles tool. For information about writing Profile home pages, see the PDF document *GRANTA MI:Viewer Home Page Author Guide*.

Defining home pages for Custom Profiles is optional:

- If you set up a home page, it will be displayed when the user selects that Profile in MI:Viewer.
- If you don't set up a home page for the Profile, and all the Tables in the Profile are from the same database, the database home page is displayed.
- If you don't set up a home page for the Profile, and the Profile contains Tables from more than one database, a default Profile home page is shown. This will include the Profile *Description* set in the Profile wizard when creating the custom Profile .

Note that the entire Profile home page is visible to all users who have permission to view the Profile. It is not possible to restrict display of parts of the home page dependent on a user's permissions, for example, to hide links on the home page to specific records.

Appendices

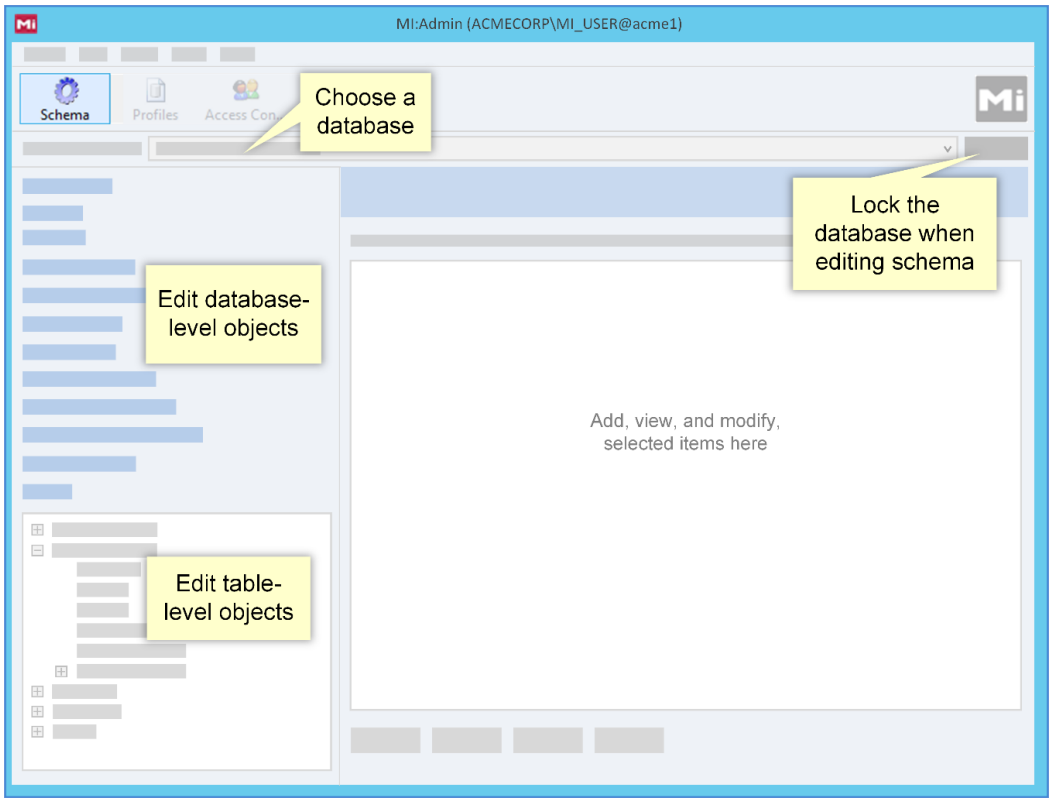
Appendix A. Naming rules for schema objects

The following rules apply to schema object names:

- Names must be unique. Database-level objects must have unique names within the database, for example, Units, Parameters, Standard Names, Record Link Groups. Table-level objects (for example, Attribute, Subset and Layout names, and the table name itself) must have unique names within the table, but the same name can be used in different tables within the same database.
- Names are case insensitive. MI:Admin will treat names as case insensitive but will preserve case. For example, if you use the Attribute name 'Strain', you will not be able to call another Attribute in the same table 'strain'. You will still be able to use 'strain' as a name for a different type of object in the same table, for example a parameter name (although doing so may cause confusion), or for Attributes in other tables.
- The maximum number of characters is 255.
- No leading or trailing spaces allowed in an object name, although you can embed spaces within a name. Leading and trailing spaces are removed in MI:Admin.
- Parameter names cannot include left square bracket ([ ), right square bracket ( ]) or colon (:) characters.

Appendix B. Schema tool

The Schema tool in MI:Admin is used to modify GRANTA MI database schema.

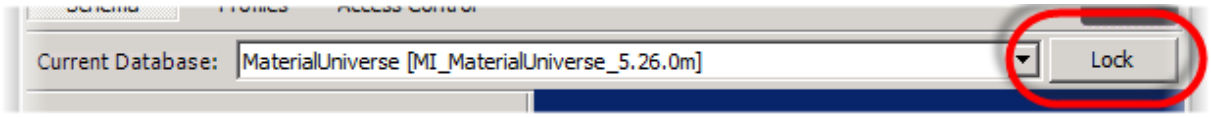


B.1 Selecting a database

Choose the database to view or modify from the **Current Database** list.

B.2 Locking the database while editing the schema

Always lock the database before making any schema changes so that non-administrator users are prevented from viewing or editing data while you are making changes. Note that means that non-administrator users of all GRANTA MI applications, including MI:Materials Gateway, will be unable to see the database until it is unlocked again.



Appendix C. Markdown text formatting

Markdown syntax can be used to format text data in GRANTA MI, for example in Long Text Attributes such as notes and descriptions. Note that inline HTML formatting **cannot** be used to format text in GRANTA MI.

C.1 Span elements

Formatting	Markdown syntax	Example	Output
Emphasis	Italic: single asterisk (*) or single underscore (_) on each side of the text Bold: two asterisks (**) or two underscores (__) on each side of the text	`Some *italic* or _italic_ text.` `Some **bold** or __bold__ text.`	Some *italic* or *italic* text. Some **bold** or **bold** text.
Links	`[link text](http://url.com "Title")` "Title" is optional and specifies the text displayed when the mouse is moved over the link text.	`A link to my [website](http://acme.com/ "ACME website")`	A link to my website ACME website
Images	`![alt text](filepath "Title")` [alt text] and "Title" are optional. [alt text] specifies text to be displayed instead of the specified image when the image is not available. "Title" specifies the text displayed when the mouse is moved over the image.	`![ACME Co Logo](images/logo.png "ACME logo")`	
Linked images	`[![alt text](filepath "Title")](http://url.com)` To link an image, place square brackets around the image code, and then place the link URL in round brackets immediately after it.	`[![Link to ACME website](images/logo.png "ACME website")](http://acme.com)`	
Code	Single backtick (`) quote on each side of the text.	`Example: `Domain\Group``	Example: `Domain\Group`

Formatting	Markdown syntax	Example	Output
Superscript	Single caret ^ character on each side of the text, for example: ^x^ The scripted text cannot include spaces.	x^2^ b^4^/12 θ^16^ 2^(10-3)^ T^max^ L^∞^ example^(Δ~i~)^ 2L(C~e~i^2^t)^1/2^	x^2 $b^4/12$ θ^{16} $2^{(10-3)}$ T^{max} L^{∞} $example^{(\Delta_i)}$ $2L(C_ei^2t)^{1/2}$
Subscript	Single tilde (swung dash) ~ character on each side of the text; for example: ~x~ The scripted text cannot include spaces.	H~2~0 l~y~ = π (d~o~^4^ - d~i~^4^)/64 example~(10-2)~ example~(a^2^+1)~ Cu~2-K~Te£ σ~max~ example~⊥~	H_20 $I_y = \pi (d_o^4 - d_i^4)/64$ $example_{(10-2)}$ $example_{(a^2+1)}$ $Cu_{2-K}Te£$ σ_{max} $example_{\perp}$

C.2 Block elements

Formatting	Markdown syntax	Example	Output
Headers	Top-level heading: three or more consecutive equals (`===`) characters on a line. Second-level heading: three or more consecutive hyphens (`---`) on a line.	Heading 1 === some text Heading 2 --- some more text	**Heading 1** some text **Heading 2** some more text
Lists, ordered	Line starting with any number followed by a period (`.`), followed by a space	6. First list item 11. Second list item 1. Third list item	1. First list item 2. Second list item 3. Third list item
Lists, unordered	Line starting with an asterisk followed by a space (first list item must be preceded by a blank line).	Commonly used in: * Lenses * light covers * glazing (particularly in	Commonly used in: • Lenses • light covers • glazing (particularly in aircraft) • light pipes

Formatting	Markdown syntax	Example	Output
		aircraft) * light pipes	
Horizontal rule	Three or more asterisks (***), or underscores (___) on a line by themselves.	some text *** some more text ___	some text some more text

C.3 Miscellaneous elements

Formatting	Markdown syntax	Examples	Output
Backslash escapes	Use a backslash escapes to generate literal characters which would otherwise have special meaning in Markdown's formatting syntax.	* asterisk * \\ backslash \\ \{ curly braces \} \{ curly braces \} \! exclamation mark \! \# hash mark \# \- minus sign or hyphen \- \+ plus sign \+ \(parentheses \) \[square brackets \] _ underscore _	* asterisk * \ backslash \ { curly braces } ! exclamation mark ! # hash mark # - minus sign or hyphen - + plus sign + (parentheses) [square brackets] _ underscore _

You can refer to the Daring Fireball Markdown (external link) site for additional syntax examples and a useful cheat sheet, but remember that inline HTML in Markdown-formatted text will not work in GRANTA MI.

Appendix D. Expressions reference

This section provides reference information for use when creating Expressions for Equations and Logic Attributes.

D.1 Expression types

Every element or sub-expression within an expression has a specific type, which is one of the following:

- *Logical* — represents a logical value, true or false.
- *Integer* — represents an whole number, 0, ±1, ±2, ±3, ...
- *Point* — represents any floating-point number. e.g. 1.5 or 3.14159.
- *Range* — represents an interval of floating-point number. e.g. 1.5 - 2.5 or 300 - 500.
- *Undefined* — represents an unspecified value of an unknown type.

The *Integer*, *Point* and *Range* types are collectively referred to as *Numeric* types. The *Undefined* type is reserved for elements where the type cannot be automatically inferred from the context.

D.2 Literals and object references

Literals are the simplest elements that an expression can contain. They correspond to simple values such as numbers (e.g. 1, 2.5) or logical states (e.g. true, false).

Object references also refer to simple values such as numbers or logical states, but the actual value used in the expression is derived from the context in which the expression is evaluated. For example, when an expression is evaluated against a specific record, and the expression contains an Attribute reference, then the value used in the expression is derived from the record's corresponding data value. So if a record has a Density of 2200 kg/m3, then the expression `[A:Density] / 1000` will be equivalent to 2200 / 1000 when it is evaluated for the record.

D.2.1 Literals

Expressions support the following literals:

Type	Examples	Notes
Logical	true false	Logical literals are case-insensitive. TRUE or False are also valid.
Integer	1 2 +3 -4	Strictly speaking, the + and - prefixes are not part of an Integer literal. They are unary operators, described later.

Type	Examples	Notes
Point	1.5 -2.5 3e6 4e-3 .5	There are two additional special cases, pi and e, which correspond to the mathematical Constants π and e. For example, 2 * pi or e ^ 2.
Range	range(1, 2) range(4.5, undefined) range(undefined, 5.9)	Ranges aren't truly supported as literals, since range(low, high) is actually a built-in function. It is often useful to think of the range function as a literal, nonetheless.
Undefined	undefined	The Undefined literal is case-insensitive. UNDEFINED or Undefined are also valid.

D.2.2 Object references

Parameters, Constants and Attributes can be referenced in an expression, as follows:

Type	Format	Examples
Parameter	[P:<name>]	[P:Temperature] [P:Cycles]
Constant	[C:<name>]	[C:Electron Charge] [C:Volume of Ideal Gas at STP]
Attribute	[A:<name>]	[A:Density] [A:Shear modulus]
Meta-attribute	[A:<parent-name>#<meta-name>]	[A:Density#Sample size] [A:Shear modulus#Std Dev]

For Attributes and Meta-attributes only, the `A:` prefix may be omitted. e.g. `[Density]`.

Since `[`, `]`, `#` and `:` are treated as special characters, if they occur within the name of a parameter, Constant or Attribute, they must be escaped by preceding them with a backslash (`\`) character. The backslash character itself must also be escaped in this way. For example:

- Donkey #123 → `[A:Donkey \#123]`
- Tool life [units] → `[A:Tool life \[units\]]`

Currently, only Logical, Integer, Point and Range Attributes are supported.

D.3 Operators

Expressions can contain a host of different operators, which are described in the following sub-sections.

D.3.1 Numeric operators

Expressions have support for the four basic arithmetic operators, `+` `-` `*` and `/`, and an additional power operator, `^`, used to raise a value to the power of another value. In addition, `+` and `-` also act as unary operators.

These operators are only defined for *Numeric* types – specifying a *Logical* operand to any of these operators results in an invalid expression. Furthermore, the power operator will not accept a *Range* type as the power operand. i.e. while `range(2, 3) ^ 2` is valid, `2 ^ range(2, 3)` is not.

Plus, Minus and Times operators

The plus operator is a binary infix operator that represents the addition operation applied to two *Numeric* sub-expressions. It has the following form:

```
expression1 + expression2
```

The minus operator is a binary infix operator that represents the subtraction operation applied to two *Numeric* sub-expressions. It has the following form:

```
expression1 - expression2
```

The times operator is a binary infix operator that represents the multiplication operation applied to two *Numeric* sub-expressions. It has the following form:

```
expression1 * expression2
```

The following table describes the type of all three operators, based on the type of the two operands.

expression1 type	expression2 type				
	Undefined	Logical	Integer	Point	Range
Undefined	Undefined	*Invalid*	Integer	Point	Range
Logical	*Invalid*	*Invalid*	*Invalid*	*Invalid*	*Invalid*
Integer	Integer	*Invalid*	Integer	Point	Range
Point	Point	*Invalid*	Point	Point	Range
Range	Range	*Invalid*	Range	Range	Range

Notes:

- These operators only support *Numeric* operands. Specifying a *Logical* operand results in an invalid expression.
- If one operand is *Undefined*, the overall type of the operator is derived from the other operand.
- If both operands are *Undefined*, the overall type of the operator cannot be automatically determined, so it defaults to *Undefined*.

Divide operator

The divide operator is a binary infix operator that represents the division operation applied to two *Numeric* sub-expressions. It has the following form:

```
expression1 / expression2
```

The following table describes the type of the operator, based on the type of the two operands.

expression1 type	expression2 type				
	Undefined	Logical	Integer	Point	Range
Undefined	Undefined	*Invalid*	Point	Point	Range
Logical	*Invalid*	*Invalid*	*Invalid*	*Invalid*	*Invalid*
Integer	Point	*Invalid*	Point	Point	Range
Point	Point	*Invalid*	Point	Point	Range
Range	Range	*Invalid*	Range	Range	Range

Notes:

- The divide operator only supports *Numeric* operands. Specifying a *Logical* operand will result in an invalid expression.
- The type will never be *Integer*, even if both operands are of *Integer* type. This is because division of two integers can produce a non-integer result. e.g. `5 / 2 = 2.5`.
- If one operand is *Undefined*, the overall type of the operator is derived from the other operand.
- If both operands are *Undefined*, the overall type of the operator cannot be automatically determined, so it defaults to *Undefined*.

Power operator

The power operator is a binary infix operator that represents the exponentiation operation applied to two *Numeric* sub-expressions, whereby the first sub-expression is raised to the power of the second sub-expression. It has the following form:

```
expression1 ^ expression2
```

The following table describes the type of the operator, based on the type of the two operands.

expression1 type	expression2 type				
	Undefined	Logical	Integer	Point	Range
Undefined	Undefined	*Invalid*	Point	Point	*Invalid*
Logical	*Invalid*	*Invalid*	*Invalid*	*Invalid*	*Invalid*
Integer	Point	*Invalid*	Point	Point	*Invalid*
Point	Point	*Invalid*	Point	Point	*Invalid*
Range	Range	*Invalid*	Range	Range	*Invalid*

Notes:

- The power operator only supports *Numeric* operands. Specifying a *Logical* operand results in an invalid expression.
- The power operand cannot be a *Range* – it must be an *Integer* or a *Point*. Specifying a *Range* results in an invalid expression.
- The type will never be *Integer*, even if both operands are *Integer*. This is because raising one integer to the power of another integer can produce a non-integer result. e.g. `2 ^ -1 = 0.5`.
- If one operand is *Undefined*, the overall type of the operator is derived from the other operand.
- If both operands are *Undefined*, the overall type of the operator cannot be automatically determined, so it defaults to *Undefined*.

Unary plus and minus operators

As well as being used to denote the addition and subtraction operations, the `+` and `-` symbols also act as unary prefix operators. They take the following form:

```
+ expression
- expression
```

The `+` is rarely used in practice, but the `-` may be commonly used to negate a numeric value. This is often seen in negative *Integer* and *Point* literals — for example `-123` is actually interpreted as the unary `-` operator applied to the *Integer* literal `123`. The use of the unary `-` operator becomes necessary when a non-literal expression needs to be negated, as in the following examples:

```
-([P:Temperature]
-([A:Density] / 1000)
-([A:Donkey] + [A:Kong]) / [A:Junior]
```

As with the other numeric operators, the + and - operators will only accept a Numeric operand. Specifying a Logical operand results in an invalid expression.

D.3.2 Logical operators

Expressions have support for the three basic logic operators, AND, OR and NOT. All three operators accept only Logical operands - the use of Numeric operands results in an invalid expression.

AND and OR operators

The AND operator is a binary infix operator that represents the logical AND operation applied to two *Logical* sub-expressions. It has the following form:

```
expression1 AND expression2
```

The OR operator is a binary infix operator that represents the logical OR operation applied to two *Logical* sub-expressions. It has the following form:

```
expression1 OR expression2
```

The following table describes the type of either operator, based on the type of the two operands.

expression1 type	expression2 type				
	Undefined	Logical	Integer	Point	Range
Undefined	*Logical*	*Logical*	*Invalid*	*Invalid*	*Invalid*
Logical	*Logical*	*Logical*	*Invalid*	*Invalid*	*Invalid*
Integer	*Invalid*	*Invalid*	*Invalid*	*Invalid*	*Invalid*
Point	*Invalid*	*Invalid*	*Invalid*	*Invalid*	*Invalid*
Range	*Invalid*	*Invalid*	*Invalid*	*Invalid*	*Invalid*

NOT operator

The NOT operator is a unary prefix operator that represents the logical NOT or inverse operation applied to a single *Logical* sub-expression. It has the following form:

```
NOT expression
```

The type of the operator is always *Logical*, even if the operand is *Undefined*. Specifying a *Numeric* operand will result in an invalid expression.

D.3.3 Relational and equality operators

To bridge the gap between *Numeric* and *Logical* types, expressions have support for the following relational and equality operators: <, <=, >, >=, = and !=. All of these are binary infix operators that accept *Numeric* operands, and they all evaluate to a *Logical* type. In addition, the equality operators, = and !=, will also accept *Logical* operands.

Relational operators

The relational operators are binary infix operators that perform a comparison between two *Numeric* operands. They have the following forms:

```
expression1 < expression2
expression1 <= expression2
expression1 > expression2
expression1 >= expression2
```

Only *Integer* and *Point* operands are supported. Comparisons between *Range* values are not well defined and should be avoided – specifying a *Range* operand results in an invalid expression. If you must compare ranges, the current recommendation is to use the `min(range)`, `max(range)` and `mean(range)` built-in functions to derive *Point* values, for which the behavior of comparison operations is well defined.

Equality operators

The equality operators are binary infix operators that compare two operands, either for equality or inequality. The have the following forms:

```
expression1 = expression2
expression1 != expression2
```

These operators support both *Numeric* and *Logical* operands, but mixing operand types is not permitted:

- `Numeric = Numeric` and `Numeric != Numeric` are valid.
- `Logical = Logical` and `Logical != Logical` are valid.
- `Numeric = Logical` and `Numeric != Logical` are not valid.
- `Numeric = Logical` and `Numeric != Logical` are not valid.

Conditional operators¶

Expressions support two forms of conditional operations, IF/ELSE and the ternary operator, ?:, that is commonly used in many other scripting languages.

?? operator

The ?? operator evaluates to one of two sub-expression, based on the evaluation result of the first sub-expression. It takes the following form:

```
first-expression ?? second-expression
```

Essentially, the `first-expression` is evaluated. If it evaluates to anything other than an undefined value, that is the result of the `??` operation, and the `second-expression` is not evaluated. However, if the `first-expression` is undefined, the `second-expression` is evaluated and used as the result of the `??` operation. Consider the following example:

```
[A:Density] ?? [A:Mass] / [A:Volume]
```

This expression will evaluate to the value of Density, if one is available. Otherwise, it will evaluate to Mass / Volume as an alternative means of obtaining a density value.

The following table describes the inferred type of the ?? operator, based on the types of the `first-expression` and `second-expression` operands.

expression1 type	expression2 type				
	Undefined	Logical	Integer	Point	Range
Undefined	Undefined	*Logical*	Integer	Point	Range
Logical	*Logical*	*Logical*	*Invalid*	*Invalid*	*Invalid*
Integer	Integer	*Invalid*	Integer	Point	Range
Point	Point	*Invalid*	Point	Point	Range
Range	Range	*Invalid*	Range	Range	Range

IF/ELSE operator

The IF/ELSE operator evaluates to one of two sub-expression, based on the evaluation result of a condition expression. It takes the following form:

```
IF (condition) true-expression ELSE false-expression
```

Essentially, the Logical `condition` expression is evaluated. If it evaluates to true, the `true-expression` is evaluated. If it evaluates to false, the `false-expression` is evaluated. Consider the following example:

```
5 + IF ([P:Temperature] > 100) 3 ELSE 4
```

This expression will evaluate to 5 + 3 when the temperature is greater than 100, or 5 + 4 when it is less than or equal to 100.

The `condition` operand must be of *Logical* type. Specifying a *Numeric* type will result in an invalid expression. The `true-expression` and `false-expression` operands must either both be *Logical* or both be *Numeric*. Specifying a mixture of *Logical* and *Numeric* operands will result in an invalid expression.

The following table describes the inferred type of the IF/ELSE operator, based on the types of the `true-expression` and `false-expression` operands.

expression1 type	expression2 type				
	Undefined	Logical	Integer	Point	Range
Undefined	Undefined	*Invalid*	Integer	Point	Range
Logical	*Invalid*	*Invalid*	*Invalid*	*Invalid*	*Invalid*
Integer	Integer	*Invalid*	Integer	Point	Range
Point	Point	*Invalid*	Point	Point	Range
Range	Range	*Invalid*	Range	Range	Range

Notes:

- If either one of `true-expression` or `false-expression` is Undefined, the overall type of the operator is derived from the other.
- If both `true-expression` and `false-expression` are Undefined, the overall type of the operator cannot be automatically determined, so it defaults to Undefined.

Ternary operator

The ternary operator is an alternate version of the IF/ELSE operator. It has the following form:

```
condition ? true-expression : false-expression
```

It is functionally identical to the IF/ELSE operator:

```
IF (condition) true-expression ELSE false-expression
```

However, unlike the `ELSE false-expression` clause of the IF/ELSE operator, all parts of the ternary operator are required.

D.4 WHERE clauses

The WHERE clause allows an expression to be broken up into parts by permitting a main expression to contain variables which themselves defined as sub-expressions. This is perhaps best illustrated by an example.

```
a + b + c
WHERE
 a = 5,
 b = 10,
 c = a + b
```

This consists of a main expression, a + b + c, followed by a list of sub-expressions which define the variables, a, b and c used in the main expression. When evaluated, the expression yields the value 30.

The general form of a where clause is as follows:

```
<main-expression>
WHERE
  <identifier1> = <sub-expression1> [, <identifier2> = <sub-expression2>, ...,
 <identifierN> = <sub-expressionN>]
```

Variable identifiers consist of an initial letter (A-Z, a-z) followed by zero or more letters (A-Z, a-z), numbers (0-9) or underscores (_). The following key words and reserved words are not permitted to be used as variable identifiers:

AND	FALSE	LOOP	SWITCH
AS	FOR	MATCH	TRUE
BREAK	FN	NOT	UNDEFINED
CASE	FUNC	NOTSET	UNTIL
DEFAULT	FUNCTION	NOT_SET	VAL
DO	IF	OR	VAR
ELSE	IMPORT	PI	WHERE
E	LET	RETURN	WHILE

D.4.1 Sub-expressions cannot be a where clause

The sub-expression of an identifier definition cannot itself consist of a WHERE clause, though it may contain WHERE clauses. For example, the following is not permitted, as the sub-expression that defines *a* is a WHERE clause.

```
a + b
WHERE
 a = c + d
 WHERE
   c = 1,
   d = 2,
 b = 3
```

This can be easily worked around by enclosing the WHERE clause in parentheses, thus:

```
a + b
WHERE
 a = (c + d
  WHERE
    c = 1,
    d = 2),
  b = 3
```

D.4.2 Cyclic references

The variable definitions are separated by commas (,) and may occur in any order as long as cyclic references are avoided. For example, the following is not permitted since the definitions of `a` and `b` are co-dependent.

```
a + b
 WHERE
   a = 2 + b,
   b = 2 * a
```

D.4.3 Variable scope

A variable declared in a WHERE clause can only be referenced from within the main expression and sub-expressions of the WHERE clause. For example, the following expression is invalid because the ELSE clause references *a*, but *a* is limited in scope and can only be referenced from within the WHERE clause.

```
if ([someAttribute] > 1000)
 a + b
   WHERE
      a = 10,
      b = 20
else
 a + 50
```

In the following expression, the variables *a* and *b* are defined twice. This is permitted because the scopes of each pair of variables are completely separate.

```
if ([someAttribute] > 1000)
  a + b
    WHERE
      a = 10,
      b = 20
else
  a + b
    WHERE
      a = 30,
      b = 40
```

D.5 Local variables

For complex expressions, you may want to break the expression into parts, and then refer to the parts in the main expression. This can be done using local variables.

```
<expression>
  WHERE <identifier> = <expression> [,<identifier> = <expression>]*
```

You can write an expression that contains identifiers, and then provided a comma-separated list of identifier declarations after the WHERE keyword.

```
a + b WHERE a = 10, b = 20
```

evaluates to 30

```
a + b WHERE a = b, b = 20
```

evaluates to 40

Note that the sub-expressions are also allowed to reference identifiers, too:

```
a + b WHERE a = 10, b = (c WHERE c = 20)
```

evaluates to 30

You can nest WHERE expressions, using parentheses.

```
a + b WHERE a = (c WHERE c = 10), b = (c WHERE c = 20)
```

evaluates to 30

Note that C is "scoped". It can be redefined in separate sub-expressions.

D.5.1 Identifier names

Identifier names are subject to the following limitations:

- The initial character in an identifier name must belong to one of the following:
  - The 26 standard Latin capital letters, from A to Z [ABCDEFGHIJKLMNOPQRSTUVWXYZ]
  - The 26 standard Latin small letters, from a to z [abcdefghijklmnopqrstuvwxyz]
  - Other Latin letters with common accents (from Unicode character 00C0 to Unicode character 017E)
  - Capital Greek letters (from Unicode character 0391 to Unicode character 03A9, excluding Unicode character 03A2)
  - Lower-case Greek letters (from Unicode character 03B1 to Unicode character 03C9)
- Any subsequent character in an identifier name must either be a valid initial character, or must belong to one of the following:
  - The 10 standard digit characters, from 0 to 9 [0123456789]
  - The underscore character, _

D.6 Example expression for the material index 'cost per unit volume'

```
1000*[A:Density]*[A:Price]
```

where

- Units for expression: currency/m^3
- Component items:
  - Attribute: Database Units
  - Density: Mg/m^3
  - Price: currency/kg
- Unit conversion: 1000 kg = 1 Mg

Appendix E. Functions reference

This section provides reference information on Functions that many be used in Expressions for Equations and Logic Attributes.

E.1 Comparison functions

Function	Usage	Evaluation

limit	`integer limit(integer value, integer low, integer high)`
	`point limit(point value, point low, point high)`
	`point limit(point value, range bounds)`
		Evaluates to *x* only if *x* lies within the range specified by *bounds*. Otherwise evaluates to *undefined*.
		If *lowValue(bounds)* is not *undefined*, and *x* < *lowValue(bounds)*, this function evaluates to *undefined*.
		If *highValue(bounds)* is not *undefined*, and *x* > *highValue(bounds)*, this function evaluates to *undefined*.
		Otherwise evaluates to *x*.

max	`point max(range x)`
	`point max(range x, range y)`
	`point max(point x, point y)`
	`integer max(integer x, integer y)`
		If there are two arguments, this evaluates to the greater of the two arguments. If either argument is undefined, this function evaluates to *undefined*.
		If the argument is a range, this evaluates to the high value of the range, or the low value if the high value is undefined.
		Note that this is not equivalent to `max(lowValue(x), highValue(x))`

min	`point min(range x)`
	`point min(range x, range y)`
	`point min(point x, point y)`
	`integer min(integer x, integer y)`
		If there are two arguments, this evaluates to the lesser of the two arguments. If either argument is undefined, this function evaluates to *undefined*.
		If the argument is a range, this evaluates to the low value of the range, or the high value if the low value is undefined.
		Note that this is not equivalent to `min(lowValue(x), highValue(x))`

E.2 Exponential and logarithmic functions

Function	Usage	Evaluation
exp	`range exp(range x)` `point exp(point x)`	Evaluates *e*x, the Constant *e* raised to the power of *x*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `e ^ x`

log	`range log(range x)`
	`point log(point x)`
	`range log(range x, point logBase)`
	`point log(point x, point logBase)`
		Evaluates *logb(x)*, the base *b* logarithm of *x*.
		If *x <= 0, b <= 0, x* is undefined or *b* is undefined, this function evaluates to *undefine*d.
		`log(x)` is equivalent to `log(x, e)`

log10	`range log10(range x)` `point log10(point x)`	Evaluates *log10(x)*, the base-*10* logarithm of *x*. If *x <= 0*, or x is undefined, this function evaluates to *undefined*. This function is equivalent to `log(x, 10)`
sqrt	`range sqrt(range x)` `point sqrt(point x)`	Evaluates *√x*, the square-root of *x*. When *x* is of type *Point*, if *x <= 0* or *x* is undefined, this function evaluates to *undefined*. When *x* is of type *Range*, if *min(x) <= 0* or *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `x ^ (1/2)`

pow	`range pow(range x, range y)`
	`range pow(range x, point y)`
	`range pow(range x, integer y)`
	`point pow(point x, point y)`
	`point pow(point x, integer y)`
		Evaluates xy, x raised to the power of y.
		If either *x* or *y* is undefined, this function evaluates to *undefined*.
		This function is equivalent to `x ^ y`

pow10	`range pow10(range x)` `point pow10(point x)`	Evaluates *10*x, ten raised to the power of *x*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `10 ^ x`

E.3 Hyperbolic functions

Function	Usage	Evaluation
arccosh	`point arccosh(point x)`	Evaluates *cosh*-1*(x)*, the inverse hyperbolic cosine of *x*, returning a value between 0 and +infinity. If *x < 1*, or *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `log(x + sqrt(x - 1) * sqrt(x + 1))`
arccoth	`point arccoth(point arg)`	Evaluates *coth*-1*(x)*, the inverse hyperbolic cotangent *x*. If *-1 < x < 1*, or *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `log((x + 1) / (x - 1)) / 2`
arccsch	`point arccsch(point arg)`	Evaluates *csch*-1*(x)*, the inverse hyperbolic cosecant of *x*. If *x* is undefined, this function evaluates to *undefined*.
arcsech	`point arcsech(point arg)`	Evaluates *sech*-1*(x)*, the inverse hyperbolic secant of *x*, returning a value between 0 and +infinity. If *x* lies outside of the interval 0 to 1, or *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `log(sqrt(1 / x - 1) * sqrt(1 / x + 1) + 1 / x)`
arcsinh	`point arcsinh(point x)`	Evaluates *sinh*-1*(x)*, the inverse hyperbolic sine of *x*, returning a value between 0 and +infinity. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `log(x + sqrt(x^2 + 1))`
arctanh	`point arctanh(point x)`	Evaluates *tanh*-1(x), the inverse hyperbolic tangent of *x*, returning a value between 0 and +infinity. If *x* lies outside the interval -1 to 1, or x is undefined, this function evaluates to *undefined*. This function is equivalent to `log((1 + x) / (1 - x)) / 2`

Function	Usage	Evaluation
cosh	`point cosh(point x)`	Evaluates *cosh(x)*, the hyperbolic cosine of *x*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `(e^x + e^-x) / 2`
coth	`point coth(point x)`	Evaluates *coth(x)*, the hyperbolic cotangent of *x*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `(e^x + e^-x) / (e^x - e^-x)`
csch	`point csch(point x)`	Evaluates *csch(x)*, the hyperbolic cosecant of *x*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `2 / (e^x - e^-x)`
sech	`point sech(point x)`	Evaluates *sech(x)*, the hyperbolic secant of *x*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `2 / (e^x + e^-x)`
sinh	`point sinh(point x)`	Evaluates *sinh(x)*, the hyperbolic sine of *x*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `(e^x - e^-x) / 2`
tanh	`point tanh(point x)`	Evaluates *tanh(x)*, the hyperbolic tangent of *x*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `(e^x - e^-x) / (e^x + e^-x)`

E.4 Range functions

Function	Usage	Evaluation
range	`range range(point lowValue, point highValue)`	Evaluates to a range value with a low value of *x* and a high value of *y*. If *x* > *y*, this function evaluates to *undefined*.
lowValue	`point lowValue(range range)`	Evaluates the low value of a range. If *x* is undefined, this function evaluates to *undefined*.
highValue	`point highValue(range range)`	Evaluates the high value of a range. If *x* is undefined, this function evaluates to *undefined*.

E.5 Statistical functions

Function	Usage	Evaluation
erf	`point erf(point x)`	Evaluates *erf(x)*, the error function of *x*. If *x* is undefined, this function evaluates to *undefined*. See the error function Wikipedia article (external link) for further details.
erfc	`point erfc(point x)`	Evaluates *erfc(x)*, the complementary error function of *x*. See the error function Wikipedia article (external link) for further details. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `1 - erf(x)`

geometric_mean	`point geometric_mean(range x)`
	`point geometric_mean(point x)`
	`point geometric_mean(point x, point y)`
		Evaluates the geometric mean of x and y. If either x or y is undefined, this function evaluates to undefined. Otherwise this function is evaluated as follows:

inverf	`point inverf(point x)`	Evaluates *erf*-1*(x)*, the inverse error function of *x*. See the error function Wikipedia article (external link) for further details. If *x* lies outside the interval -1 to 1, or *x* is undefined, this function evaluates to *undefined*.

	x < 0	*x = 0*	*x > 0*
y < 0	-sqrt(x * y)	undefined	undefined
y = 0	undefined	undefined	undefined
y > 0	undefined	undefined	sqrt(x * y)

<table>
<tr><th>Function</th><th>Usage</th><th>Evaluation</th></tr>
<tr><td>inverfc</td><td><code>point inverfc(point x)</code></td><td>Evaluates erfc-1(x), the inverse complementary error function of x. See the error function Wikipedia article (external link) for further details.
If x lies outside the interval -1 to 1, or x is undefined, this function evaluates to undefined.</td></tr>
<tr><td>mean</td><td><code>point mean(range x)</code>
<code>point mean(point x)</code>
<code>point mean(point x, point y)</code></td><td>point mean(range x) evaluates the arithmetic mean of a range. This function is evaluated according to the following rules:
<table>
<tr><th></th><th>HIGHVALUE(X) DEFINED</th><th>HIGHVALUE(X) UNDEFINED</th></tr>
<tr><td>LOWVALUE(X) DEFINED</td><td>(highValue(x) + highValue(x)) / 2</td><td>lowValue(x)</td></tr>
<tr><td>LOWVALUE(X) UNDEFINED</td><td>highValue(x)</td><td>undefined</td></tr>
</table>
Note that this is not equivalent to mean(lowValue(x), highValue(x)).
point mean(point x, point y) evaluates the arithmetic mean of x and y. If either x or y is undefined, this function evaluates to undefined.
This function is equivalent to <code>(x + y) / 2</code></td></tr>
<tr><td>phi</td><td><code>point phi(point x)</code></td><td>Evaluates Φ(x), the normal cumulative distribution function of x. See the normal cumulative distribution function Wikipedia article (external link) for further details.
If x is undefined, this function evaluates to undefined.
This function is equivalent to <code>(erf(x / sqrt(2)) + 1) / 2</code></td></tr>
<tr><td>probit</td><td><code>point probit(point x)</code></td><td>Evaluates probit(x), the inverse normal cumulative distribution function of x. See the probit function Wikipedia article (external link) for further details.
If x is undefined, this function evaluates to undefined.
This function is equivalent to <code>sqrt(2) * inverf(2*x - 1)</code></td></tr>
</table>

E.6 Trigonometric functions

Function	Usage	Evaluation
sin	`point sin(point x)`	Evaluates *sin(x)*, the sine of *x*. If *x* is undefined, this function evaluates to *undefined*.
cos	`point cos(point x)`	Evaluates *cos(x)*, the cosine of *x*. If *x* is undefined, this function evaluates to *undefined*.
tan	`point tan(point x)`	Evaluates *tan(x)*, the tangent of *x*. If *x* is undefined, this function evaluates to *undefined*.
csc	`point csc(point x)`	Evaluates *csc(x)*, the cosecant of *x*. If *x* is undefined, this function evaluates to *undefined*.
sec	`point sec(point x)`	Evaluates *sec(x)*, the secant of *x*. If *x* is undefined, this function evaluates to *undefined*.
cot	`point cot(point x)`	Evaluates *cot(x)*, the cotangent of *x*. If *x* is undefined, this function evaluates to *undefined*.
arcsin	`point arcsin(point x)`	Evaluates *sin*-1*(x)*, the inverse sine of *x*, returning a value between 0 and π. If *x* lies outside of the interval -1 to 1, or *x* is undefined, this function evaluates to *undefined*.
arccos	`point arccos(point x)`	Evaluates *cos*-1*(x)*, the inverse cosine of *x*, returning a value between 0 and π. If *x* lies outside of the interval -1 to 1, or if *x* is undefined, this function evaluates to *undefined*.
arctan	`point arctan(point x)` `point arctan(point x, point y)`	Evaluates *tan*-1*(x)*, the inverse tangent of *x*, returning a value between -π/2 and π/2. If *x* is undefined, this function evaluates to *undefined*.
arccsc	`point arccsc(point x)`	Evaluates *csc*-1*(x)*, the inverse cosecant of *x*, returning a value between -π/2 and π/2. If *-1 < x < 1*, or if *x* is undefined, this function evaluates to *undefined*.

Function	Usage	Evaluation
arcsec	`point arcsec(point x)`	Evaluates *sec*-1*(x)*, the inverse secant of *x*, returning a value between 0 and π. If *-1 < x < 1*, or if *x* is undefined, this function evaluates to *undefined*.
arccot	`point arccot(point x)` `point arccot(point x, point y)`	Evaluates *cot*-1*(x)*, the inverse cotangent of *x*, returning a value between 0 and π. If *x* is undefined, this function evaluates to *undefined*.

E.7 Miscellaneous functions

Function	Usage	Evaluation
abs	`point abs(point x)` `integer abs(integer x)`	Evaluates the absolute value of *x*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `IF (x > 0) x ELSE -x`

hasValue	`bool hasValue(bool x)`
	`bool hasValue(integer x)`
	`bool hasValue(point x)`
	`bool hasValue(range x)`
		Determines whether or not the argument represents an actual value, rather than being undefined.
		Returns *false* if *x* is *undefined*. Otherwise returns *true*

sgn	`integer sgn(point x)` `integer sgn(integer x)`	Evaluates *sgn(x)*, the sign or signum function of *x*. if *x < 0*, *sgn(x)* evaluates to *-1* if *x = 0*, *sgn(x)* evaluates to *0* if *x > 0*, *sgn(x)* evaluates to *1* if *x* is undefined, this function evaluates to *undefined*.
sinc	`point sinc(point x)` `integer sinc(integer x)`	Evaluates *sinc(x)*, the normalised sinc function of *x*. If *x* is undefined, this function evaluates to *undefined*. Note that, if x is of Integer type, then *sinc(x)* is also of Integer type, whereby *sinc(x)* is equivalent to `IF (x = 0) 1 ELSE 0` This function is equivalent to `sin(pi * x) / (pi * x)`

Function	Usage	Evaluation
unsf	`integer unsf(point x)` `integer unsf(integer x)`	Evaluates H1(*x)*, the unit or Heaviside step function of *x*. Note that *unsf(0) = 1*. If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `IF (x >= 0) 1 ELSE 0`
unsfz	`integer unsfz(point x)` `integer unsfz(integer x)`	Evaluates $H_0(x)$, the unit or Heaviside step function of x. Note that *unsfz(0) = 0* If *x* is undefined, this function evaluates to *undefined*. This function is equivalent to `IF (x > 0) 1 ELSE 0`

valueOrDefault	`bool valueOrDefault(bool x, bool y)`
	`integer valueOrDefault(integer x, integer y)`
	`point valueOrDefault(point x, point y)`
	`range valueOrDefault(range x, range y)`
		Evaluates to *x*, or *y* if *x* is *undefined*.
		This function is equivalent to `IF (hasValue(x)) x ELSE y`