# SERAM User Manual

## Sustainable Enterprise Reporting And Management

## by Sirius Technologies AG

# Notices

## Copyright

## Disclaimer

The information in this manual is provided on an “as is” basis, without warranty. While every effort has been taken by the author in the preparation of this manual, the author shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this manual.

This manual may contains links to third-party web sites that are not under the control of the author. The author is not responsible for the content of any linked site. Inclusion of a link in this manual does not imply that the author endorses or accepts any responsibility for the content of that third-party site.

## Trademarks

All terms mentioned in this manual that are known to be trademarks or service marks have been capitalized as appropriate. Use of a term in this manual should not be regarded as affecting the validity of any trademark or service mark.

## Versions

Software release: Banker (Build 2472)
Document revision: 495

# Contents

# Chapter 1

# What is SERAM?

SERAM is a software tool that enables companies to collect and report upon their sustainability data. This includes topics like health and safety, waste, $CO_2$ and other air emissions, water consumption and pollution, energy use, gender and race diversity, child labor, human rights, fraud and corruption cases, investment in the community.

In essence, the software allows companies to collect all the data they would usually put into their Corporate Responsibility (CR) reports. In addition it allows companies to report to sustainability bodies such as *GRI*, *Dow Jones Sustainability Index* and *FTSE4good*.

Since SERAM is an online tool, access to SERAM only requires a modern web browser and an internet connection.

**Note:** If SERAM is deployed on premise, the requirements for accessing SERAM may differ. Your SERAM administrator will communicate the details to you.

## What data can SERAM manage?

SERAM collects, computes, manages and analyzes information.

In its core, SERAM is a fully configurable business reporting system, with specific functionality and work-flows to support high-quality data collection processes. In consequence, SERAM can be used in many different settings, including:

- **CSR / HSE reporting**
  - Environment
    - Performance
    - Compliance
  - Social
    - Health & Safety
    - Labor practices
  - Economic
    - Community
    - Governance
- **Internal performance management**
  - CO2 footprint
  - Water balance
  - Target setting
  - Compliance summary
  - Injury & illness rates
  - Child labor report
  - Employee well-being
  - Community investment
  - SOX compliance
- **External reporting**
  - CR report

- Annual report
- Web reporting
- DJSI
- GRI
- Global Compact
- SRI analysis
- GHG protocol
- FTSE4good
- Regulatory reporting
- Industry reports

## Technical requirements for using SERAM

SERAM is a browser-based web application which uses the latest HTML5 technologies. Even though compatibility with older browsers has been added, SERAM has some minimal requirements that need to be fulfilled in order to run properly.

- JavaScript and cookies must be enabled and allowed in the browser
- Network access to `*.seram.ch` or `*.seram.app` via HTTPS must be allowed

Any modern HTML5 browser should work fine with SERAM, even if they are not listed in the following browser compatibility list:

**Google Chrome**

Current versions of Google Chrome, including Chrome for Android.

**Tip:** This is our preferred browser, since our tests have shown that the overall performance and feature set is currently the best for running SERAM.

**Firefox**

Current versions of Firefox.

**Edge**

Current versions of Edge.

**Brave**

Current versions of Brave.

**Internet Explorer**

IE 11 is fully supported. Older versions of IE are not longer supported.

**Safari**

Current versions of Safari.

**Tip:** This includes the iOS versions, such as on iPhone and iPad.

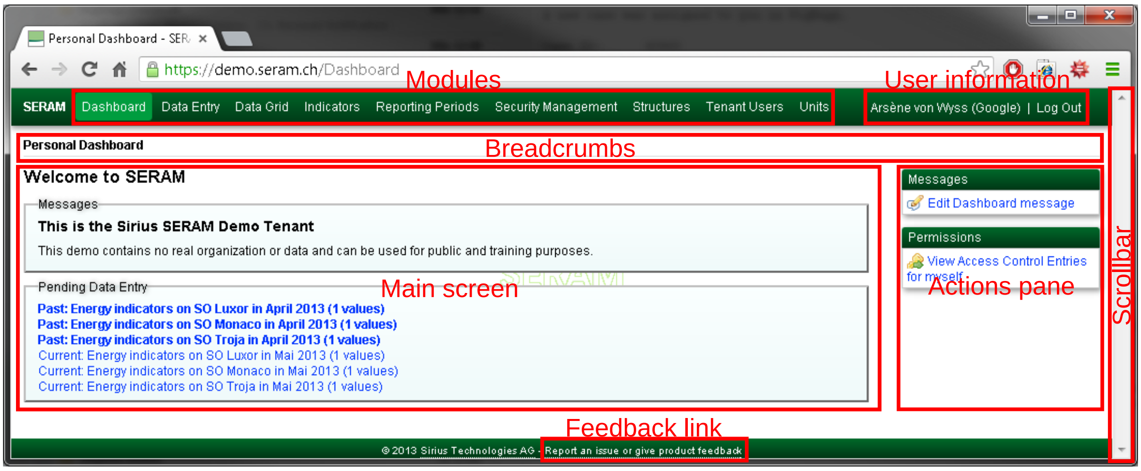
**Others**

Untested/unsupported, but users have reported successful usage of SERAM with various browsers.

# Chapter 2

# SERAM User Interface

The SERAM user interface always follows the same basic layout, so that it is easy to use.



Welcome to SERAM

Messages

**This is the Sirius SERAM Demo Tenant**

This demo contains no real organization or data and can be used for public and training purposes.

Pending Data Entry

**Past: Energy indicators on SO Luxor in April 2013 (1 values)**
**Past: Energy indicators on SO Monaco in April 2013 (1 values)**
**Past: Energy indicators on SO Troja in April 2013 (1 values)**
Current: Energy indicators on SO Luxor in Mai 2013 (1 values)
Current: Energy indicators on SO Monaco in Mai 2013 (1 values)
Current: Energy indicators on SO Troja in Mai 2013 (1 values)

## Modules

The Modules can be accessed directly through the links on the top of the window.

Modules in SERAM bundle related functionality. The different modules of SERAM are listed and described in the topic *SERAM Modules*.

**Note:** Depending on your access permissions, you may have access to fewer modules than shown on the sample screen-shot. This is not an error. Please contact your company SERAM support/administrator if you need access to a module which is not listed.

## User Information

The top right corner will always display the identity of the user currently logged on, and provide a log-off link which destroys the session, preventing other users from impersonating someone else.

**Note:** When SERAM is installed on premise with transparent Windows authentication, the log-off link will not be available. However, SERAM creates no session information in this case, so that logging off from Windows securely logs you off SERAM as well (even if the computer files were compromised).

## Scroll-bar

The scroll-bar provides vertical scrolling in SERAM.

The vertical scroll bar allows to navigate large pages, trees etc. in SERAM. The whole window except for the *Action Pane* is scrolled, allowing you to efficiently use your display.

**Note:** Due to the nature of the data grid in the *Data Grid*, where large amounts of information are expected but headings should remain on screen, dedicated scroll-bars are used for scrolling the data grid instead of using the main scroll-bar.

## Breadcrumbs

The Breadcrumbs provide an overview of where the user is in a module, and provide backward navigation.

Many actions cause the main screen content to switch to a different view. Each new view which is added on top of the module or previously visible view is recorded in the Breadcrumbs, thereby creating a trail of the steps performed. Each of these entries can be clicked to directly return to the corresponding step.

**Note:** Unless noted otherwise (when performing Data Entry for instance), navigating back using a breadcrumb entry **discards** all collected information. The only way to get back to the later step is to re-do the necessary steps again.

**Note:** The Back button of the internet browser returns one step in the breadcrumbs. The Forward button however doesn't change the view.

## Main Screen

The Main Screen is where SERAM provides and collects data.

The main screen changes depending on the module and the actions invoked. All relevant information is collected and displayed in this screen area.

Please refer to the different *SERAM Modules* descriptions for details on the provided functionality.

## Action Pane

The Action Pane is the central UI element, offering contextual actions to the user.

Depending on the context (module, selected items in the main screen, state, wizard step etc.), the system offers direct access to *actions* in the form of links.

The different actions are explained in detail in the different tasks throughout the SERAM documentation. Since actions are contextual, they sometimes also display related contextual information in addition to the link.

**Note:**

The Action Pane will always stay on screen, even if the *window scroll-bar* is used to scroll the browser contents. Depending on your screen resolution and browser window it can sometimes happen that there are more actions displayed than there is space on the screen, but you cannot scroll the Action Pane because it stays on screen during the scroll.

In order to make the bottom actions accessible, collapse the action groups by clicking their header. You can revert this at any time by clicking the header again.

### Actions

Most operations in SERAM are triggered by an action.

The actions are displayed in categories on the *action pane* to the right side of the main screen, or in the bottom part of the main screen. Please refer to the *SERAM Modules* and their views for more information about available actions.

**Note:** The actions displayed in the bottom area can be mirrored in the top area through configuration. This can be useful for configurations with many indicators or structures, so that expanded trees don't require the user to scroll to the bottom for triggering an action.

## Feedback Link

The feedback link provides direct access to the SERAM development.

At Sirius Technologies AG, we strongly believe that listening to customer feedback is the best way to constantly make our product even better. Therefore, we welcome all users to submit feedback such as suggestions of bug reports. Clicking on the feedback link in the footer will open a new window or tab where you can provide us with your valued feedback.

☞ **Note:** While you may submit issue reports anonymously, we kindly ask you to provide your e-mail address so that we can get in touch regarding the reported issue if necessary. Your privacy is important to us, therefore we guarantee that your e-mail address will be used for this purpose only.

Chapter

3

# SERAM Modules

All functionality in SERAM is accessed through a set of so-called modules.

## Dashboard

The Dashboard module is the landing page for all SERAM users. It provides useful information such as messages from the administrators and links for pending Data Entry.

Whenever a user logs into a SERAM tenant without using an explicit module URL, he will be presented with his personalized Dashboard.

The contents shown on the Dashboard is:

- Messages posted by the tenant administrators, which could contain information such as deadlines, contact information, and specific usage hints. Administrators will also find an *Actions* to edit the message.
- A list with pending Data Entry. This list is computed from all *Structure Node* on which the currently logged-on user has been assigned appropriate *Roles*.

  **Note:** If the list would get too big to be useful, a corresponding message will appear. This is not an error but prevents the users from being overwhelmed by a huge list of entries to be processed.
- An *action* for viewing the permissions of the logged-on user.

### Data Status Controlling

If you are a *Data Verifier,* you see the **Data Status Controlling** summary on your *Dashboard*. The entries are colored.

**Note:** The data status of a value is usually represented with a color coded background. In the default SERAM setup, orange means a value is open, yellow means it is entered with pending verification, and green means it has been verified successfully.

## Data Entry

The Data Entry module provides a simple and guided way for entering data into SERAM.

Users which are responsible for entering specific data points have the Data Entry module at their disposal to do this in a simple yet efficient way even if they are novices and not regular computer users. The Data Entry module reduces the number of decisions required from the user to a bare minimum, so that little upfront training is required.

**Note:** The same Data Entry interface is also available from within the *Dashboard* and from within the *Data Grid*, but without the selection steps.

## Data Grid

The Data Grid module is the main module dealing with indicator values. It allows performing extensive data analysis tasks, and also provides the tools for implementing data validation work-flows.

At the center of the data management with SERAM is the Data Grid module. It provides access to all indicator data in SERAM, and provides tools for implementing the core SERAM work-flow tasks.

**Note:** Users migrating from SERAM v1 used to know this module under the name Data Analysis. However, since more functionality has been added to the module, it has been renamed to reflect its support for non-analysis tasks.

Indicator data points can be broken down and analyzed from different viewpoints, for instance data entered on sites in an organization may be analyzed in relation to the countries those sites are in, or depending on the site type.

Indicators can be edited per value point or in batches, and the module provides tools for exporting and importing data from respectively to SERAM.

## Indicators

The Indicators Module provides administration facilities for indicators and control over indicator permissions.

Administration of *Indicator Tags* and associated *permissions* as well as the *Indicators* are managed in the Indicators module.

**Note:** The indicator definitions have changed significantly in SERAM v2 compared to SERAM v1. Nevertheless, this module mostly corresponds to a combination of the Master Data Indicators and Indicators modules in the old SERAM version.

Additionally, the module can be used as entry point to manage *activated indicators* in any *Reporting Period*.

## Reporting Periods

The reporting processes in SERAM are managed by the Reporting Periods module.

The Reporting Periods module provides the administrative interface to manage reporting periods, perform workflow tasks such as restating forecasted values, and also servers as an entry point for managing activated indicators.

## Security Management

SERAM provides a sophisticated role-based permission management, which allows very fine-grained control over visibility and permitted actions on indicators, structures and in the system in general.

This module allows to manage roles (what the different roles can do in the system), assign tenant-wide permissions, and define special role-based behaviors.

## Structures

Structures are key in SERAM - they are used as baseline for indicators, permissions, data consolidation and much more.

The whole management of the historized structures is performed through the Structures Module.

## Tenant Users

Every SERAM tenant can have its own set of user meta-data, and every user action can be audited. This module allows to manage these.

## Units

SERAM comes with a full set of SI units, such as distance, mass or energy, but additional units and also conversions between arbitrary units can be added easily.

## Global Tenants

While most customers use a single tenant, SERAM allows the setup and use of an arbitrary number of tenants.

## Global Users

SERAM users are recognized by so-called claims globally above the tenant level. Every user logging into SERAM must have associated claims that identify him or her.

# Chapter 4

# Indicators

The main entities for collecting and reporting data are indicators.

In SERAM, indicators are the data entities for collecting and reporting data. A good understanding of indicators and associated functionality is therefore key to use SERAM to its fullest.

## Indicator Tags

All indicators are assigned to at least one Indicator Tag. These tags control the visibility and access rights of the contained indicators.

The Indicator Tags divide the indicators in indicator sets, and the permissions set on indicator tags control the permissions on all indicators of that set. Only indicators assigned to tags which are visible to a user will also be visible to the user.

**Note:** Indicator Tags and *Group Indicators* are distinct. The first is used to control visibility and access, while the latter is an indicator used to structure the indicators.

### Creating an Indicator Tag

New Indicator Tags can be created in this task.

1. Switch to the *Indicators Module*.
2. Click the **Manage Indicator Tags** action (located in the “Indicator Tags” category).
3. In the new view, click the **Add Indicator Tag** action (located in the “Indicator Tags” category).
4. Enter the code for the tag. The code has to be unique across the tenant. It can be up to 100 alpha-numeric characters long.
5. Enter the name and description for the tag.
6. Optional: In the “Translate” category, you may translate the name and the description to other languages.
7. Optional: Create the new tag by clicking **Save**.

By default, the new tag will only have permissions as defined by the *Automatic Role Permissions Grants*. Therefore, it may be necessary to add new permissions as described in the task *Viewing and editing Permissions on an Indicator Tag* in order to enable proper access for non-administrative users.

### Editing an Indicator Tag

The code, name and description and translations of an existing Indicator Tag can be changed in this task.

1. Switch to the *Indicators Module*.
2. Click the **Manage indicator tags** action (located in the “Tags” category).
3. Select the tag in the list.

   **Note:** The list can quickly be narrowed down to the relevant tags by typing text fragments into the **Search** field above the list header.
4. Click the **Edit ...** action (located in the “Selected Tag” category).
5. Make the desired changes to the code, name, description and translations.

   **Note:** The code has to be unique across the tenant. It can be up to 100 alpha-numeric characters long.
6. Save the changes by clicking **Save**.

## Deleting an Indicator Tag

Indicator Tags which are not exclusive to any Indicator can be deleted in this task.

Perform these steps to completely remove a tag from the system, along with all its indicator tag assignments. The tag may only be deleted when it is not exclusive to any *indicator*, that is, no indicator may be orphaned (left without any tag assignment) when deleting a tag.

1. Switch to the *Indicators Module*.
2. Click the **Manage indicator tags** action (located in the “Tags” category).
3. Select the tag in the list.

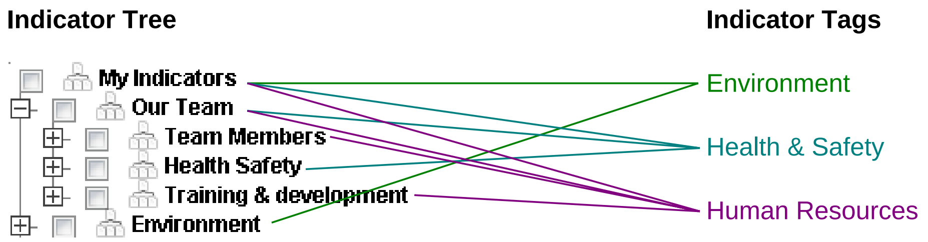
   **Note:** The list can quickly be narrowed down to the relevant tags by typing text fragments into the **Search** field above the list header.
4. Click the **Delete ...** action in the “Selected Tag” category, and confirm in the popup dialog.

## Indicator Tag Assignments

Since the tags control access and visibility of indicators, it is important to assign all applicable tags to an indicator.

This is not only important for input indicators, but also for indicators which are used to form the tree structure, such as *Group Indicators* and *Calculation Indicators*.

The following example shows how the assignments could look like:



The indicators (shown in the left column) are assigned to tags in the right column. The color code of the tags and lines is for simpler recognition of the assignment on the indicator.

Here, the “My Indicators” is assigned to all three tags, so that any user with access to a subset of the tags still sees this indicator. Any user with access to “Health & Safety” or “Human Resources” would see “Our Team”, but a user with access to “Environment” only would not see “Our Team” or any of the indicators below.

## Assigning Indicator Tags to Indicators

The assignments of Indicator Tags to Indicators can be modified in this task.

1. Switch to the *Indicators Module*.
2. Select the indicator in the list.

   **Note:** The list can quickly be narrowed down to the relevant indicators by typing text fragments into the **Search** field above the list header.
3. Click the **Edit ...** action (located in the “Selected Indicator” category).
4. Make the desired changes to tag assignments in the bottom section.
5. Save the changes by clicking **Save**.

## Indicator Tag Permissions

Indicators do not have permissions. However, permissions are applied through their indicator tag assignments.

Every indicator is assigned to at least one *Indicator Tag*. The Indicator Tag Permissions set on the Indicator Tag are then applied to all indicators assigned to it. As such it is additive.

**Note:** The effective permissions when entering data of a *persistent indicator* is determined by making the intersection of the permissions on the indicator tag(s) and the permissions on the structure node.

If a user has no access to any of the tags assigned to an indicator, he will not be able to see the indicator at all - not even the code or name.

If a user may edit values of tag T1 and only access tag T2, and an indicator is assigned to both tags, then he will be able to edit values of this indicator.

### Viewing and editing Permissions on an Indicator Tag

Use this task to modify access permissions and audit settings on indicator tags.

1. Switch to the *Indicators Module*.
2. Click the **Manage indicator tags** action (located in the “Tags” category).
3. Select the tag in the list for which you want to modify permissions.

   **Note:** The list can quickly be narrowed down to the relevant tags by typing text fragments into the **Search** field above the list header.
4. Click the **View permissions of: Tag ...** action (located in the “Permissions” category).
5. View and edit the tag permissions as described in the topic *Using the Permissions View*.

## Indicator Master Data

The Indicator Master Data provides consistent naming and reporting across reporting periods.

All indicators in SERAM are created equal. The indicator meta-data such as the code, name and description as well as *indicator tag* associations and the *owning structure* are common to all indicators in SERAM.

**Note:** This is unlike SERAM v1 where the indicator type as well and many settings were part of the master data. In SERAM v2, the same indicator can have different settings and even different *indicator types* across the different reporting periods.

### Indicator Properties

Properties offered by the Indicator Master Data.

| Name | Description | Characteristics |
|---|---|---|
| Code | The code of the indicator is an alpha-numeric identifier which is language-invariant. | Unique across the system, maximum length 31 characters |
| Name | The name of the indicator is used for display purposes. | Translatable. |
| Description | The description is shown to the user when he enters data, and in the Indicator Details view. | Translatable, and *Markdown* syntax can be used to include formatting and links. |
| *Sort Key* | The sort key controls the order in unordered indicator lists. | Alphanumeric. When not specified, the code (see above) is used for sorting. |
| *Owner Structure* | Only the structure and its children will be able to use this indicator, thereby allowing the creation of limited-scope indicators (e.g. only for a specific business unit, or countries in Europe etc.) | Users do not need access rights to the owner structure in order to see the indicator. Indicator visibility is controlled by the permissions on its assigned indicator tags. |
| *Indicator Tags* | Indicator tags control the access and filtering of indicators. | Every indicator must be assigned to at least one tag, which in turn control the access to the indicator.<br>• For managing the indicators, the user must have the |

| Name | Description | Characteristics |
|---|---|---|
| | | Indicators:ManageIndicator function on **all** assigned tags<br>• For changing values of this indicator (entering data/ comments or changing the status), the user must have the corresponding Indicators:SetValue, Indicators:SetComment or Status:* functions on **any** assigned tag<br>• For enabling external applications to change the value and/or status, the application must have the Indicators:SetByApplication function on **any** assigned tag |

### Indicator Owner Structure

Every Indicator in SERAM is owned by a structure. This defines on which structures data may be collected for this indicator.

The owner structure is defined at the master data level of the indicator, and it cannot be changed after the indicator has been created. Only structures below the indicator will have the indicator in their scope.

There are three typical usage scenarios for the owner structure:

- Indicators which are owner by the primary structure type can be seen as the primary indicators of the system. These are always visible by the default selections of the *reference environment*, for instance when using the *Data Grid Module*. In typical SERAM installations, the majority of indicators will be defined like this.
- Indicators which are owned by other structure types are typically used as attribute-dependent factors in calculations. For instance, one might define a CO2 emission factor which depends on countries, or an efficiency factor which depends on a site type.
- Indicators defined on non-type structures can be used to collect data specific to this entity. For instance, a business unit may collect specific resource usage, or some compliance information might be collected only for sites in the US.

### Indicator Sort Key

A sort key can be assigned to indicators, which is used as primary key for sorting in unordered indicator lists.

Sort keys are alphanumeric strings. They are used as primary sort key, and the indicator code is used as secondary sort key. Therefore, if the sort key is not used, indicators will still be consistently ordered by their code.

**Note:** The sort is alphabetic, not numeric. Therefore the system will sort the keys “1”, “2” and “10” as follows: “1”, “10”, “2”. Therefore, if you want to use numbers as sort key, make sure to pad them with “0” characters, so that they all have the same number of characters. In the previous example, the following could have been used as sort keys to ensure numeric order: “001”, “002”, “010”

## Creating an Indicator

The meta-data, such as name, description and scope of an indicator, can be defined in this task.

Perform these steps when you want to define new indicator meta-data. The indicator data defined here is valid across all reporting periods. Therefore, if you want to activate a previously used indicator on another period, you need to activate an *Indicator Types* on that period instead of performing this task.

1. Switch to the *Indicators Module*.
2. Click the **Add Indicator** action (located in the “Indicators” category).
3. Enter the code for the indicator. The code has to be unique across the tenant. It can be up to 100 alpha-numeric characters long.
4. Enter the name and description for the indicator.
5. Optional: In the “Translate” category, you may translate the name and the description to other languages.

6. Optional: Enter a tenant-wide alphanumeric *sort key* for the indicator.
7. Optional: Add *Applications* code mappings (only available if an application has been created).
8. Pick the *Indicator Tag(s)* to assign to this indicator. At least one tag has to be assigned.
9. Create the new indicator by clicking **Save**.

### Editing an Indicator

The meta-data of an existing indicator, such as name and description, can be edited in this task.

Perform these steps when you want to change the meta-data of an existing indicator. The indicator data changed in this task is valid across all reporting periods.

1. Switch to the *Indicators Module*.
2. Select the indicator in the list.

   **Note:** The list can quickly be narrowed down to the relevant indicators by typing text fragments into the **Search** field above the list header.
3. Click the **Edit ...** action located in the “Selected Indicator” category.
4. Make the desired changes to the code, name, description, translations, *sort key*, *application mappings* and *Indicator Tag* assignments.

   **Note:** The code has to be unique across the tenant. It can be up to 31 alpha-numeric characters long.
5. Save the changes by clicking **Save**.

### Deleting an Indicator

The meta-data of indicators which are not used can be deleted in this task.

Perform these steps to remove the meta-data of an unused indicator. Unused means that it may not be activated and used any reporting period, including past ones. If you want to de-activate an indicator in a specific reporting period, perform the task *Deactivating an Indicator* instead.

1. Switch to the *Indicators Module*.
2. Select the indicator in the list.

   **Note:** The list can quickly be narrowed down to the relevant indicators by typing text fragments into the **Search** field above the list header.
3. Optional: Click the **View details of ...** action in the “Selected Indicator” category to check that this indicator is not used anymore. This is shown in the table at the bottom of the new view. Click **Back** to return after checking.
4. Click the **Delete ...** action in the “Selected Indicator” category, and confirm in the popup dialog.

### Checking the Indicator Activation Status

This task allows to check the indicator details, including what reporting periods it has been activated on.

Perform these steps to find out on which reporting periods a specific indicator has been activated. This can for instance be important when writing or verifying plausibility checks which access the data of past periods.

1. Switch to the *Indicators Module*.
2. Select the indicator in the list.

   **Note:** The list can quickly be narrowed down to the relevant indicators by typing text fragments into the **Search** field above the list header.
3. Click the **View details of ...** action in the “Selected Indicator” category to open the indicator details view.

## Indicators activated on Reporting Periods

For using the indicators in SERAM, they must be activated on reporting periods.

Effectively, the activation of *indicators* assigns them to a specific *indicator type* on a *reporting period*. Since reporting periods cover a specific date range, SERAM will use the reporting period and the Structure Scope for determining what indicators are available on each *structure node*.

## Scope Structure

Every indicator in SERAM is scoped to a *Structure* (either a *Structure Type* or a *Structure Node*).

The *Scope Structure* is key to the behavior of the indicator in SERAM especially when multiple *Structure Types* are being used.

### Indicator Scope

For *Indicator Values* in the same *Structure Type* as the *Structure Scope*, only the *Structure Nodes* under (or equal to) the *Structure Scope* will be considered when computing or entering values. The scope can be used to make some indicators specific to only a part of a structure type.

### Data Flow

For *Indicator Values* in a different *Structure Type* as the *Structure Scope*, the *Data Flow between Structure Types* will apply.

## Indicators with Members

Some of the indicator types can have members, which are used to construct the indicator tree. More specifically, these are the *Group Indicator* type and - depending on the setup - also the *Calculation Indicator* type.

The indicator tree is constructed by evaluating all indicators visible to the user in the specific context. Any indicator which is not listed as a member of another indicator will be shown on the top of the tree, that is, as a root. It is therefore not necessary (and not possible) to explicitly manage the tree layout for the indicators.

Strictly speaking, while SERAM displays a tree, the indicators can form a graph since indicators may re-appear as members of different indicators. These are the same indicators despite being distinct entries in the tree.

## Persistent Indicators

Persistent indicators are used for storing data. While there are different types of persistent indicators for different types of data, they all share some common functionality.

Each of these indicators is persisted on a specific *structure level*. Depending on the setup of the specific indicator, data may be computed for other levels above or below as well, but this data is always computed from the data stored at the persistence level.

Also depending on the setup is the frequency where data is persisted. The available frequency choices depend on the *reporting period type* used. In the most common case, the Fiscal Year, the frequency can be yearly, quarterly or monthly. Again, the indicator may yield computed data for other frequencies depending on its type and setup.

### Frequencies

Frequencies define the time slices in a reporting period. In the common case, the frequencies are Yearly, Quarterly or Monthly.

**Note:** There is no UI for defining custom frequencies in SERAM. However, these can be created by your SERAM supplier at the database level as a customization.

### Indicator to Structure Assignments

Persistent indicators can be configured for each applicable structure node, including their activation status.

If an indicator is mandatory, it will be active on all structures below the owner structure which are on the persistence level. Otherwise, you can enable or disable the indicator on each of these structures, depending whether this structure is supposed to collect this indicator or not. Also, some indicator types allow further configuration on the structure level, for instance *Reporting Indicators* can define a custom input unit or a different input frequency per structure.

## Activating an Indicator (Indicators Module)

Indicators need to be activated to become available on reporting periods. This task describes how to activate indicators from within the Indicators module, for instance after creating a new indicator.

Perform these steps when you have created indicators and you want to activate them for use on a specific reporting period. If you just want to change the structure assignment of an indicator already activated, perform the task *Configuring Indicator on Structures by Persistent Indicator* instead.

1. Switch to the *Indicators Module*.
2. Select the indicator(s) in the list.

   You may select multiple indicators if you want to batch-activate them all with the same type and settings.

   **Note:** The list can quickly be narrowed down to the relevant indicators by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.
3. Click the **Activate ...** action (located in the "Selected Indicator" category).
4. Pick the *Indicator Type* you want to use for this indicator activation, and the *Reporting Period* on which the indicator(s) are to be activated.

   **Note:** Only reporting period on which the indicator has not yet been activated can be picked.
5. Click the **Activate ... on ...** button on the bottom to continue.

Continue with the activation as described in the activation task for the indicator type you picked.

## Activating an Indicator (Reporting Periods Module)

Indicators need to be activated to become available on reporting periods. This task describes how to activate indicators from within the Reporting Periods module, for instance when managing the indicators of a reporting period.

Perform these steps when you have indicators which you want to activate on a specific reporting period. The indicator may already be active in other reporting periods, but not in the one you want to activate it on. If you just want to change the structure assignment of an indicator already activated, perform the task *Configuring Indicator on Structures by Persistent Indicator* instead.

1. Switch to the *Reporting Periods*.
2. Choose the reporting period in the list.
3. Click the **Indicators for ...** action (located in the "Selected Reporting Period" category).
4. Click the **Activate indicators on this period** action (located in the "Selected Reporting Period" category).
5. Pick the *indicator owner structure* which owns the indicator(s) to be activated, and click the "Use ... for indicators" button on the bottom.

   **Note:** To pick a structure type instead of a specific structure node, click the tab on the top with the structure type name.
6. Select the indicator(s) in the list.

   You may select multiple indicators if you want to batch-activate them all with the same type and settings.

   **Note:** The list can quickly be narrowed down to the relevant indicators by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.
7. Click the **Activate ...** button on the bottom.
8. Pick the *Indicator Type* you want to use for this indicator activation.
9. Click the **Activate ... on ...** button on the bottom to continue.

Continue with the activation as described in the activation task for the indicator type you picked.

## Deactivating an Indicator

When an active indicator is no longer needed on a reporting period, it can be deactivated using this task.

The indicator to be deactivated must not be used by other indicators, including group indicators and calculations. Therefore, you may need to edit the indicator(s) using the indicator you want to deactivate in order to detach it.

The deactivation completely removes an indicator and all its associated data from a reporting period. If you only wish to enable or disable a persistent indicator on a structure node, perform the task *Configuring Indicator on Structures by Persistent Indicator* instead.

1. Switch to the *Reporting Periods*.
2. Choose the *reporting period* in the list.
3. Click the **Indicators for ...** action (located in the "Selected Reporting Period" category).
4. Select the indicator(s) in the tree.

   **Note:** To quickly locate an indicator, the tree can be searched by typing text fragments into the **Search** field above the tree and clicking **Find**.

☞ **Note:** If the indicator is not in the root level of the tree, then you first need to edit the parent *indicator with member* to exclude this indicator, or the deactivation will fail.

5. Click the **Deactivate ...** action in the “Selected Indicator” category, and confirm in the popup dialog.

## Configuring Indicator on Structures by Persistent Indicator

A persistent indicator can be configured per applicable structure node using this task.

Perform these steps to change the activation and other structure-specific options (depending on the *indicator type* of the indicator to be edited) per applicable structure node.

1. Switch to the *Reporting Periods*.
2. Choose the reporting period in the list.
3. Click the **Indicators for ...** action located in the “Selected Reporting Period” category.
4. Select the indicator(s) in the tree.

   ☞ **Note:** To quickly locate an indicator, the tree can be searched by typing text fragments into the **Search** field above the tree and clicking **Find**.
5. Click the **View indicator assignments for ...** action located in the “Selected Indicator” category.
6. Select the structure node(s) in the list.

   ☞ **Note:** The list can quickly be narrowed down to the relevant structure nodes by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.

   The list contains all structure nodes on which this indicator is available and of the same persistence level.
7. Depending on the selection made, you can now edit (single selection only), activate or deactivate the indicator on the selected structure node(s). These actions are located in the “Selected Indicator Assignment” category.

## Configuring Indicators on Structure by Structure Node

All persistent indicators of a specific structure node can be configured using this task.

Perform these steps to change the activation and other structure-specific options (depending on the *indicator type* of the indicator to be edited) for all persistent indicators of a structure node.

1. Switch to the *Structures*.
2. Select the appropriate *structure type* and *structure node* for which you want to configure the indicators.
3. Click the **View ... indicator assignments** action in the “Selected Node” category.
4. Choose the reporting period in the list, and click the **Use ...** button on the bottom to continue.
5. Click the **Indicators for ...** action located in the “Selected Reporting Period” category.
6. Select the indicator(s) in the list.

   ☞ **Note:** The list can quickly be narrowed down to the relevant indicators by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.

   The list contains all persistent indicators applicable to this structure node.
7. Depending on the selection made, you can now edit (single selection only), activate or deactivate the selected indicator(s) on the structure node. These actions are located in the “Selected Indicator Assignment” category.

## Copy Indicator Assignments to other Periods by Indicator

Some indicator assignments can be mass copied per location to existing indicators and locations in other periods using this task.

Perform these steps to mass copy indicator assignments (depending on the *indicator type* of the copy source and the copy destination) per location to existing indicators and locations in other periods per applicable *Reporting Periods*.

1. Switch to the *Indicators Module*.
2. Click the **Manage Activated Indicators** action in the “Indicators” category.
3. Choose the Source Reporting Period and click **Use...**.
4. Click the **Copy Indicator Assignments to other Periods** action in the “Selected Reporting Period” category.
5. Optional: Use **Replace (source active)** and/or **Deactivate (source inactive)** to change the default behavior.

   ☞ **Note:** By default, this task doesn't overwrite active target *Assignments*.

6. Expand the **Structures** tree and chose the *Structure Nodes* to be affected by the copying process.

   **Note:** The tree only shows the structures of the source period.

7. Choose the *Indicators* for which you want to copy the *Assignments*.

   **Note:** If the *Indicator Type* in the target period is different from the source period, this task might not copy all parts of the *Assignment*.

8. Choose the **Target Reporting Period(s)**.

   **Note:** Only compatible *Reporting Periods* are listed.

9. Click the **Copy Indicator Assignments** button.

   **Note:** The button appears upon completion of a valid selection.

A confirmation dialog with the number of successfully copied *Assignments* is shown.

**Note:** In general this should sum up to *Indicators***Structures***Periods*. Non-applicable *Assignments* will not be listed separately.

# Chapter 5

# Indicator Types

Every indicator can be activated on periods with multiple indicator types. They define how the indicator behaves, and what data is offered by the indicator data.

Indicators are type-independent, therefore the same indicator can be set up as a different type in different reporting periods. This enables progression in the indicator definitions: while an indicator may be a simple reporting indicator in one year, it can be changed to be a calculation using multiple indicators as source in the next year.

## Group Indicator

Group indicators discard numbers, but they allow structuring indicators into a more accessible indicator graph.

In contrast to the *calculation*, the order of the indicators used in the group can be explicitly configured. The groups are primarily used to build the logical/visual indicator relationships.

Group indicators however also process some of the underlying data, especially the status, so that they display a consolidate status view for all contained indicators.

### Group Indicator Properties

The following properties are available on Group Indicators in addition to the indicator master data.

| Name | Description | Characteristics |
|---|---|---|
| Members | The members define which other indicators are part of this group. | Only indicators activated on the same reporting period can be members. The order of the members is retained. |

### Activating a Group Indicator

Activate a Group Indicator for logical grouping of indicators, so that the system can build a meaningful tree of indicators, using this task.

You must have started the activation of a Group Indicator, for instance by using Group Indicator as indicator type at the end of one of the tasks *Activating an Indicator (Indicators Module)* or *Activating an Indicator (Reporting Periods Module)*.

1. Pick the *Scope Structure* for the indicator.
2. In the tree, check all indicators to be included in the group.

   **Note:** To quickly locate an indicator, the tree can be searched by typing text fragments into the **Search** field above the tree and clicking **Find**.
3. Arrange the order of the indicators by dragging the indicators into their position.
4. Activate the Group Indicator by clicking the **Save** button on the bottom.

### Edit a Group Indicator

Make changes to a Group Indicator using this task.

1. Switch to the *Reporting Periods*.
2. Choose the reporting period in the list.

3. Click the **Indicators for ...** action (located in the “Selected Reporting Period” category).
4. Select the group indicator in the tree.

   **Note:** To quickly locate an indicator, the tree can be searched by typing text fragments into the **Search** field above the tree and clicking **Find**.
5. Click the **Edit ...** action in the “Selected Indicator” category.
6. Optional: Check and uncheck indicators in the tree to match the selection you want to be included in the group.
7. Optional: Arrange the order of the indicators by dragging the indicators into their position.
8. Save the Group Indicator by clicking the **Save** button on the bottom.

# Calculation Indicator

The calculation indicators are the analytic indicator in SERAM. They can compute numeric results using simple or complex formulas.

The calculations also aggregate the other data dimensions and compute the numeric output with the given formula. The input for the formula can be any indicator which produces a numeric result, usually *reporting indicators*, but also *boolean indicators* with a unit, *factor indicators* or just other calculations.

The syntax elements of the formula language are explained in a separate section, *Formulas*.

A flag called show details controls whether the indicators (of the same period) used in the calculation are displayed underneath the calculation or nor, thereby allowing to use calculations to build a graph of indicators.

The calculation level can also be defined, which is important to make the formula perform as expected when using non-summing operators and functions. For instance, if an emission factor is used, it is important to compute this on the same level where the organization is assigned to the country providing the emissions factor, or the total will not be correct.

Let's assume we have indicators `A` and `B` which are collected on structure `1` and `2`, so that the value `A1`, `A2`, `B1` and `B2` are available to the calculation. Also, let's assume a calculation which computes `A*B`.

- When no level is set, the calculation is performed once, and each indicator value referenced in it is used as consolidated value. Therefore, the effective computation performed is `(A1+A2)*(B1+B2).`
- When the level is set to the structure level of the structures, then the formula is performed on each of the structures, thereby computing it as `(A1*B1)+(A2*B2).`

Examples of computations which use the different computation levels:

- An efficiency ratio is computed by taking input and output indicators, that is, by dividing the output by the input. The ratio cannot be summed up, therefore it has to be computed at the current level and not on a specific level.
- An emission factor is stored as an indicator value, and multiplied by an input, to generate CO2 emission equivalents. The CO2 equivalents must be summed up and it cannot be computed from a summed or averaged emission factor, therefore it has to be computed at the level of the input data where the emission factor exists.

## Calculation Indicator Properties

The following properties are available on Calculation Indicators in addition to the indicator master data.

| Name | Description | Characteristics |
|---|---|---|
| Members | The members are used to build the indicator tree. | This property is computed and therefore read-only. Its content depends on the Show Details and the Calculation properties. The order of the members is following the *indicator sort key*. |

| **Name** | **Description** | **Characteristics** |
|---|---|---|
| Calculation | The calculation formula is used to compute the results. | The formula needs to produce a numeric result, and it may use all syntax elements described in the section *Formulas*. Indicators can be referenced as well as long as they have a numeric result. |
| Display Formula | **Optional.** The display formula is used to compute a derivation based on the calculation result. The formula can output a text value. | The formula may use all syntax elements described in the section *Formulas*. Indicators can be referenced as well as long as they have a numeric result. |
| Calculation Result Unit | The (default) result *unit* reported along with the number. | |
| Method-Specific Calculation Result Unit | The result unit reported along with the number when the value is in the context of a specific *Method*. | A unit may be chosen for each *Method* defined in the system. If left blank, the unspecific *Calculation Result Unit* is used. |
| Data Flow | Controls the way the calculation is consolidated and how data flows between different structure types when this indicator is being used. | **No data flow**<br>The calculation produces no value when queried across structure types.<br>**Direct**<br>The direct (outgoing) structure link is used to pull the data to the target structure. Values are propagated to child structure nodes, but not consolidated to parent structure nodes. This mode should be used when the value cannot be summed up, for instance some emission factor.<br>**Consolidate**<br>The data is consolidated from the structures nodes linking to the target structure node. Also, the data is consolidated to the parent structure nodes, but not propagated to child structure nodes. This mode should be used when the value is supposed to be summed up, for instance some energy use. |
| Show Details | When enabled, all indicators of the same reporting period used in the calculation will be handled as members of this indicator, thereby forming the indicator tree. | |

| Name | Description | Characteristics |
|---|---|---|
| Computation Level | Controls how calculations are handled when consolidating over structure nodes and across structure links.<br>**Note:** This is usually needed when a calculation performs a multiplication. | **None**<br>When no computation level is set, the calculation will always be performed once using the indicator values available on the same structure node for which it is computed. This mode is the most common case.<br>**Specific level**<br>The calculation is performed on the specified level, and the results are consolidated. This mode can be used to make calculations work with indicator values which cannot be summed up. For instance, if an emission factor is available on the site level of an organization, the CO2 equivalents need to always be computed on the site level as well because the emission factors cannot be consolidated. A business unit with multiple sites will then sum up the calculation results of the child sites. |
| Computation Frequency | Controls how calculations are handled when consolidating over time.<br>**Note:** This is usually needed when a calculation performs a multiplication.<br>**Note:** When using a specific frequency, using a specific level is likely also appropriate. | **None**<br>When no computation frequency is set, the calculation will always be performed once using the indicator values available on the same time index for which it is computed. This mode is the most common case.<br>**Specific frequency**<br>The calculation is performed on the specified time frequency, and the results are consolidated. This mode can be used to make calculations work with indicator values which cannot be summed up. For instance, if you want to compute the ton-kilometers as multiplication of two inputs (tons and kilometers), the result needs to be computed on the input frequency of these values since multiplying the time- |

| Name | Description | Characteristics |
|---|---|---|
| | | consolidated totals will not yield the expected results. |

## Activating a Calculation Indicator

Activate a Calculation Indicator for performing computations using other indicators, and optionally also for structuring the indicator tree, using this task.

You must have started the activation of a Calculation Indicator, for instance by using Calculation Indicator as indicator type at the end of one of the tasks *Activating an Indicator (Indicators Module)* or *Activating an Indicator (Reporting Periods Module)*.

1. Pick the *Scope Structure* for the indicator.
2. Click the **Change the formula of Calculation Indicator** action in the “Formulas” category to edit the calculation formula, and complete the process by pressing the **Use this formula** button.

   Detailed information and instructions about the formulas are available in the topic *Formula Syntax Elements*.
3. Choose the result unit to be used by the calculation.
4. Optional: If the calculation shall be used as a parent node in the indicator tree (listing the indicators used in the formula as members), check the **Show details** option.
5. Optional: Pick a computation level on which the calculation shall be performed.
6. Optional: Select a data flow kind to use.

   In most cases, the **Consolidate** data flow kind is appropriate.

## Edit a Calculation Indicator

Make changes to a Calculation Indicator using this task.

1. Switch to the *Reporting Periods*.
2. Choose the reporting period in the list.
3. Click the **Indicators for ...** action (located in the “Selected Reporting Period” category).
4. Select the calculation indicator in the tree.

   **Note:** To quickly locate an indicator, the tree can be searched by typing text fragments into the **Search** field above the tree and clicking **Find**.
5. Click the **Edit ...** action in the “Selected Indicator” category.
6. Optional: Edit the formula and the other properties as required.
7. Save the Calculation Indicator by clicking the **Save** button on the bottom.

# Reporting Indicator

Reporting indicators are the core of the data collection process in SERAM.

Most data in SERAM is collected through reporting indicators defined on the structures. Since their main task is collecting data, they have an input and a reporting value. The input can be configured per structure node, such as picking a specific input unit or input frequency, to facilitate the data collection process.

To ensure high data quality, the indicator can be configured with constraints and plausibility checks which validate the values entered and control the status changes allowed.

In addition to this, the reporting indicator can also be configured to use a factor provided by a *Factor Indicator* when processing data, so that it automatically applies emission factors or currency conversions entered in the corresponding factor indicators.

## Reporting Indicator Properties

### Persistence

| Name | Description | Characteristics |
|---|---|---|
| Active | Controls whether the indicator has to be reported on this structure node or not. When inactive, it will have no value for this structure node. | Configurable per structure node only. |
| Mandatory | When an indicator is mandatory, all structures of the given level have to report it. | When set to "true", this flag overrides the per-structure "active" setting. |
| Reporting Frequency | The *Frequencies* at which the data for this indicator is to be reported. | Can be any of the applicable frequencies. |
| Reporting Level | The *Structure Level* on which the indicator is collected and stored. | This level has to be of the same *Structure Type* as the *Indicator Master Data*'s owner structure. |

### Numeric Reporting

| Name | Description | Characteristics |
|---|---|---|
| Reporting Unit | The unit for the numeric value reported by this indicator. | Required. |
| Allowed Input Units | A list of allowed input units, when no Factor Indicator is set. | Mutually exclusive with Factor Indicator. The Reporting Unit is always implicitly allowed, even when no Allowed Input Unit has been selected. |
| Factor Indicator | Use another indicator as input unit and conversion source . | Mutually exclusive with Allowed Input Units. The factor indicator unit must match the reporting unit of this indicator. |
| Time Aggregation | Over time, values can either be summed (for instance Energy Use) or averaged (for instance Headcount). | Required, defaults to “sum”.<br>**Sum**<br>The values are summed together towards the lower frequencies compared to the Reporting Frequency. Depending on the application set-up, values are interpolated for the higher frequencies. However, in no case does the system fill in missing values, so that the summed number is a year-to-date number, not a year-end-estimate number.<br>**Average**<br>The values are averaged, taking only non-empty values into consideration. Higher frequencies inherit |

| Name | Description | Characteristics |
|---|---|---|
| | | the value entered at the reporting frequency. |
| No Consolidation | If this indicator collects a value which cannot be summed up on the organization, such as a factor, this needs to be checked to avoid summation. When not consolidated, values will propagate to the descendant structure nodes instead. | Default is to consolidate. This also controls the data flow across structure types (direct when checked, consolidated otherwise). |
| Default Is Forecasted | Impose the forecasted flag whenever a default value is applied. | Only relevant when a default value is defined. |
| Default Value | The default value is returned by the system whenever the input value is empty, thereby allowing users to not enter an input value but still follow the proper reporting work-flow. | Optional. Can be any *expression formula*, but the most common value is just `0`. |
| Constraint Check | The constraint makes sure that values not matching the constraint will not be used or accepted by the system. The user will get a feedback message stating the violation. | Optional. Must be a *predicate formula*, with some common constraints available for selection in the UI. |
| Plausibility Check | The plausibility check is similar to the constraint in that it checks a value, but a violation doesn't prevent the value from being used, and as soon as a comment is entered the system will also accept a status change. | Optional. Must be a *predicate formula*, common checks are available for direct selection in the UI. |
| Input Unit | Defined on the indicator structure assignment, this defines the unit which this structure uses for input. The system will automatically apply *Unit Conversions* to compute the result in the reporting unit.<br>**Note:** When the conversion is missing, the *reporting value* will remain empty. | Defaults to the Reporting Unit. Only applicable when no Factor Indicator has been selected; otherwise the source unit assigned to the Factor Indicator will be used.. |
| Input Frequency | The frequency at which an assigned structure inputs the data. | Defaults to the reporting frequency. The input frequency can only be higher than the reporting frequency (that is, require more values which are then aggregated according to the time consolidation). |

## Activating a Reporting Indicator

Activate a Reporting Indicator for numeric data collection. This is the most common indicator type.

You must have started the activation of a Reporting Indicator, for instance by using “Reporting Indicator” as indicator type at the end of one of the tasks *Activating an Indicator (Indicators Module)* or *Activating an Indicator (Reporting Periods Module)*.

1. Pick the *Scope Structure* for the indicator.
2. Select the unit to be used for the values reported using this indicator.

   **Note:** The list can quickly be narrowed down to the relevant units by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.

   Click the **Next** button on the bottom to move on to the input unit selection.
3. Make your selection as to which input units are valid for the indicator.

   - Use a specific list of allowed input units by picking any number of units from the list.
   - Use a factor indicator, which has to have the same target unit as the reporting unit selected in the previous step. The input unit will be supplied by the factor indicator.

     **Note:** If the list of factor indicators does not contain the indicator you're looking for, make sure to double-check that the reporting unit does indeed match the unit of the factor indicator.

   Click the “Next” button on the bottom to move on to the indicator attributes selection.
4. Select the appropriate indicator attributes.

   The attributes are described in the *Reporting Indicator Properties*. Click the **Next** button on the bottom to move on to the reporting level selection.
5. Pick the *Structure Level* at which the indicator data has to be entered.

   **Warning:** The structure level selection cannot be changed after activation without deactivating the indicator.
6. Activate the indicator by clicking the **Save** button on the bottom.

## Edit a Reporting Indicator

Make changes to a Reporting Indicator using this task.

1. Switch to the *Reporting Periods*.
2. Choose the reporting period in the list.
3. Click the **Indicators for ...** action (located in the “Selected Reporting Period” category).
4. Select the reporting indicator in the tree.

   **Note:** To quickly locate an indicator, the tree can be searched by typing text fragments into the **Search** field above the tree and clicking **Find**.
5. Click the **Edit ...** action in the “Selected Indicator” category.
6. You can now change the unit to be used for the values reported using this indicator.

   **Note:** The list can quickly be narrowed down to the relevant units by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.

   Click the **Next** button on the bottom to move on to the input unit selection.
7. You can now change the selection of valid input units for the indicator, including switching between the Factor Indicator and Allowed Input Unit selections.

   Click the **Next** button on the bottom to move on to the indicator attributes selection.
8. Select the appropriate indicator attributes.

   The attributes are described in the *Reporting Indicator Properties*.
9. Save the changes by clicking the **Save** button on the bottom.

## Assigning a Reporting Indicator

Configure specific options on structures for a reporting indicator.

You must have opened the indicator assignments of the reporting indicator, for instance by initiating the edit operation while performing the steps outlined in the *Configuring Indicator on Structures by Persistent Indicator* or *Configuring Indicators on Structure by Structure Node* tasks.

1. You can activate and de-activate the indicator for the structure node selected previously.

**Note:** If the indicator has been marked as being mandatory, the selection made here has no effect.

2. The input frequency can be changed to a higher frequency than the reporting frequency.
3. If no Factor Indicator is being used on this Reporting Indicator, you can select the input unit to use for the structure node selected previously.

   **Note:** If no input unit is selected, the reporting unit will be used as default.

4. Click the **Save** button on the bottom to save the assignment.

# Boolean Indicator

Boolean indicators are used to collect yes/no question answers, such as “ISO9001 certified?” or “Is there a water treatment plant?”. They can also output a numeric value which is available for further processing, for instance as part of a calculation.

Whenever a unit is assigned, the indicator will generate a number 0 or 1 for “no” respectively “yes” as numeric output. When the reporting frequency is higher than the year, the numeric value will be averaged over time.

> If the said certification was awarded in April, then Q1 would be “no” (0) and Q2 onwards would be “yes” (1), the year numeric value would then become 0.75 because the certification was in place 75% of the time.

## Boolean Indicator Properties

### Persistence

| Name | Description | Characteristics |
|---|---|---|
| Active | Controls whether the indicator has to be reported on this structure node or not. When inactive, it will have no value for this structure node. | Configurable per structure node only. |
| Mandatory | When an indicator is mandatory, all structures of the given level have to report it. | When set to "true", this flag overrides the per-structure "active" setting. |
| Reporting Frequency | The *Frequencies* at which the data for this indicator is to be reported. | Can be any of the applicable frequencies. |
| Reporting Level | The *Structure Level* on which the indicator is collected and stored. | This level has to be of the same *Structure Type* as the *Indicator Master Data*'s owner structure. |

### Boolean Reporting

| Name | Description | Characteristics |
|---|---|---|
| Consolidate | Controls whether the numeric value of this boolean indicator is consolidated on parent structures. When enabled, the parent structure will represented as sum. | This flag also controls the Data Flow across structure types. |
| Numeric Unit | When set, the indicator will also provide a numeric result which reflects the value entered, 0 for false and 1 for true. | Optional. |

## Activating a Boolean Indicator

Activate a Boolean Indicator for collection of yes/no answers.

You must have started the activation of a Boolean Indicator, for instance by using Boolean Indicator as indicator type at the end of one of the tasks *Activating an Indicator (Indicators Module)* or *Activating an Indicator (Reporting Periods Module)*.

1. Pick the *Scope Structure* for the indicator.
2. Select the appropriate indicator attributes.

   The attributes are described in the *Boolean Indicator Properties*.
3. Pick the *Structure Level* at which the indicator data has to be entered.
4. Optional: Select the unit to be used for the values reported using this indicator.

   **Note:** The list can quickly be narrowed down to the relevant units by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.
5. Activate the indicator by clicking the **Save** button on the bottom.

## Edit a Boolean Indicator

Make changes to a Boolean Indicator using this task.

1. Switch to the *Reporting Periods*.
2. Choose the reporting period in the list.
3. Click the **Indicators for ...** action (located in the “Selected Reporting Period” category).
4. Select the boolean indicator in the tree.

   **Note:** To quickly locate an indicator, the tree can be searched by typing text fragments into the **Search** field above the tree and clicking “Find”.
5. Click the “Edit ...” action in the “Selected Indicator” category.
6. Make the changes as required.

   **Note:** The unit list can quickly be narrowed down to the relevant units by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.
7. Save the changes by clicking the **Save** button on the bottom.

# Text Indicator

Text indicators are used to collect textual data on a specific level.

Since text cannot be consolidated across the structure, their scope is limited. An example of a text indicator may be “Feedback to management”, or “Other certifications held”.

## Text Indicator Properties

### Persistence

| Name | Description | Characteristics |
|---|---|---|
| Active | Controls whether the indicator has to be reported on this structure node or not. When inactive, it will have no value for this structure node. | Configurable per structure node only. |
| Mandatory | When an indicator is mandatory, all structures of the given level have to report it. | When set to "true", this flag overrides the per-structure "active" setting. |
| Reporting Frequency | The *Frequencies* at which the data for this indicator is to be reported. | Can be any of the applicable frequencies. |
| Reporting Level | The *Structure Level* on which the indicator is collected and stored. | This level has to be of the same *Structure Type* as the *Indicator Master Data*'s owner structure. |

### Text Reporting

| Name | Description | Characteristics |
|---|---|---|
| Constraint Pattern | When the text should match a specific pattern (such as a comma-separated list of single words), this can be defined here to enforce it. | A regular expression which is implicitly rooted at the beginning and the end.<br>**Note:** For an introduction to Regular Expressions, you can follow this link: *http://www.regular-expressions.info/* |

## Activating a Text Indicator

Activate a Text Indicator for collection of textual answers, or for text fragments usable in *Reports*.

You must have started the activation of a Text Indicator, for instance by using Text Indicator as indicator type at the end of one of the tasks *Activating an Indicator (Indicators Module)* or *Activating an Indicator (Reporting Periods Module)*.

1. Pick the *Scope Structure* for the indicator.
2. Select the appropriate indicator attributes.

   The attributes are described in the *Text Indicator Properties*.
3. Pick the *Structure Level* at which the indicator data has to be entered.

   **Warning:** The structure level selection cannot be changed after activation without deactivating the indicator.
4. Activate the indicator by clicking the **Save** button on the bottom.

## Edit a Text Indicator

Make changes to a Text Indicator using this task.

1. Switch to the *Reporting Periods*.
2. Choose the reporting period in the list.
3. Click the **Indicators for ...** action (located in the “Selected Reporting Period” category).
4. Select the text indicator in the tree.

   **Note:** To quickly locate an indicator, the tree can be searched by typing text fragments into the **Search** field above the tree and clicking **Find**.
5. Click the **Edit ...** action in the “Selected Indicator” category.
6. Make the changes as required.
7. Save the changes by clicking the **Save** button on the bottom.

# Factor Indicator

Factor indicators are simple numeric indicators used for storing numeric factors, such as *combustion to CO2*, or *currency exchange rates*.

The factors are simple numeric indicators which also hold a specific unit information. These indicators can be used as unit and unit conversion factor in the configuration of *reporting indicators*. In addition to that, they may also be used as factors in *calculations*, typically when the calculation is set to be computed on a specific level.

The factor always has a reporting unit, and the value stored is the equivalent of “input” required to compute the “output”. So for instance if currencies are to be converted, and the reporting is done in EUR, when CHF is chosen as unit in the indicator assignments, a value such as 1.22 would be correct, when this is the amount of CHF required to obtain 1 EUR.

In contrast to other indicator types, their latest value is carried over to a new *Reporting Period* whenever it is *copied to a new reporting year*, thus carrying over the factors.

## Factor Indicator Properties

### Persistence

| Name | Description | Characteristics |
|---|---|---|
| Active | Controls whether the indicator has to be reported on this structure node or not. When inactive, it will have no value for this structure node. | Configurable per structure node only. |
| Mandatory | When an indicator is mandatory, all structures of the given level have to report it. | When set to "true", this flag overrides the per-structure "active" setting. |
| Reporting Frequency | The *Frequencies* at which the data for this indicator is to be reported. | Can be any of the applicable frequencies. |
| Reporting Level | The *Structure Level* on which the indicator is collected and stored. | This level has to be of the same *Structure Type* as the *Indicator Master Data*'s owner structure. |

### Factor Reporting

| Name | Description | Characteristics |
|---|---|---|
| Input Unit | The input unit for this factor, that is, the “source” unit of the factor. The factor “n” is applied so that `n InputUnit == 1 ReportingUnit`. | Configurable per structure node only. |
| Reporting Unit | The reporting unit for this factor, that is, the “target” unit of the factor. | Required. |

## Activating a Factor Indicator

Activate a Factor Indicator for sorting computation factors, such as currency conversions or CO2e-emission factors.

You must have started the activation of a Factor Indicator, for instance by using Factor Indicator as indicator type at the end of one of the tasks *Activating an Indicator (Indicators Module)* or *Activating an Indicator (Reporting Periods Module)*.

1. Pick the *Scope Structure* for the indicator.
2. Select the target unit of the factors to be entered in this indicator.

   **Note:** The list can quickly be narrowed down to the relevant units by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.

   Click the **Next** button on the bottom to move on to the indicator attributes selection.
3. Select the appropriate indicator attributes.

   The attributes are described in the *Factor Indicator Properties*. Click the **Next** button on the bottom to move on to the reporting level selection.
4. Pick the *Structure Level* at which the indicator data has to be entered.

   **Warning:** The structure level selection cannot be changed after activation without deactivating the indicator.
5. Activate the indicator by clicking the **Save** button on the bottom.

## Edit a Factor Indicator

Make changes to a Factor Indicator using this task.

1. Switch to the *Reporting Periods*.
2. Choose the reporting period in the list.
3. Click the **Indicators for ...** action (located in the “Selected Reporting Period” category).
4. Select the factor indicator in the tree.

   **Note:** To quickly locate an indicator, the tree can be searched by typing text fragments into the **Search** field above the tree and clicking **Find**.
5. Click the **Edit ...** action in the “Selected Indicator” category.
6. You can now change the target unit to be used for the factor.

   **Note:** The list can quickly be narrowed down to the relevant units by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.

   Click the “Next” button on the bottom to move on to the indicator attributes selection.
7. Select the appropriate indicator attributes.

   The attributes are described in the *Factor Indicator Properties*.
8. Save the changes by clicking the **Save** button on the bottom.

## Assigning a Factor Indicator

Configure specific options on structures for a factor indicator.

You must have opened the indicator assignments of the factor indicator, for instance by initiating the edit operation while performing the steps outlined in the *Configuring Indicator on Structures by Persistent Indicator* or *Configuring Indicators on Structure by Structure Node* tasks.

1. You can activate and de-activate the indicator for the structure node selected previously.

   **Note:** If the indicator has been marked as being mandatory, the selection made here has no effect.
2. You can change the unit to be used as source unit for the factor entered on this indicator.

   **Note:** The list can quickly be narrowed down to the relevant units by typing text fragments into the **Search** field above the list header. The selection made is retained when paging or searching.
3. Click the **Save** button on the bottom to save the assignment.

# Method Factor Indicator

*Method Factor Indicators* are used to retrieve *Method Factor* total or partial factors.

In contrast to a persistent *Factor Indicator*, the factors are not entered and persisted using *Data Entry* but rather retrieved from a database.

## Method Factor Indicator Properties

The *Method Factor Indicator* has no specific properties, they are controlled by the *Method Factors*.

## Activating a Method Factor Indicator

Activate a *Method Factor Indicator* for retrieving factor values from the *Method Factor* databases.

You must have started the activation of a *Method Factor Indicator*, for instance by using Method Factor Indicator as indicator type at the end of one of the tasks *Activating an Indicator (Indicators Module)* or *Activating an Indicator (Reporting Periods Module)*.

1. Pick the *Scope Structure* for the indicator.
2. Activate the indicator by clicking the **Save** button on the bottom.

# Chapter 6

# Indicator Values

All indicators generate indicator values as output for further processing, such as for display or reporting.

## Data Dimensions

The indicator value is composed of several data dimensions.

Each of the *Indicator Types* contributes a different set of data dimensions to the values. Even though each of the indicator types has different value generation rules, the final values can be generically used for reporting or data processing by accessing the value's data dimensions.

### Status

The indicator status is related to the quality work-flow of an indicator.

Whenever multiple indicator values are consolidated, the least status is taken for the consolidated value.

**Note:** The data status of a value is usually represented with a color coded background. In the default SERAM setup, orange means a value is open, yellow means it is entered with pending verification, and green means it has been verified successfully.

If Site A reported a value in the Verified status and Site B has still a value in the Completed status, the resulting value on the Business Unit AB will be of the Completed status as well.

#### Indicator States

A SERAM installation defines an indicator status workflow consisting of always one editable state and usually multiple non-editable states.

**Note:** The number of states and their transitions and required rights can be customized per tenant by the supplier if the default work-flow is not suitable for your specific needs. Changing the status attributes, such as the name, description or color code, can be done directly by the customer through the UI.

**Open**

The Open status is the only editable status. Every persistent indicator starts out as Open, and all editing operations are allowed in this status. Also, whenever related data such as emission factors or unit conversions changes, the resulting reporting value will reflect these right away. After entering the data in the Open state, the responsible person changes the status to Entered. Doing this freezes the value, any subsequent changes to related data will not have any effect unless the value is restated to Open at a later time.

**Entered**

This is the first frozen state. The comment may still be edited, but everything else is now immutable. The person verifying the data quality has the choice to either confirm the validity of the data by setting the status to Verified, or to re-set it to Open for correction.

**Verified**

The data has reached the final data status. As with Entered, only the comment remains editable. Also, unless the value is marked as being forecasted, only users with administrative power can re-set it to Open.

#### Modify Indicator Status Attributes

Indicator status attributes such as the name or the color code can be modified using this task.

1. Switch to the *Indicators Module*.

2. Click the **Edit indicator statuses** action (located in the “Indicator Statuses” category).
3. In the new view, select the indicator status you with to edit in the list.
4. Click the **Edit ...** action (located in the “Indicator Status” category).
5. Make the changes as required.
6. Optional: In the “Translate” category, you may translate the name and the description to other languages.
7. Optional: Save the changes by clicking **Save**.

## Numeric Value

Representing the reporting value, the numeric value is the most versatile and also most used data dimension.

It holds a possibly empty numeric value, a unit, and possibly an information about a failed constraint. Also, in some cases the value may directly be editable (such as in Factor Indicators).

## Previous Numeric Value

In *Data Entry*, the previous numeric value is shown.

The previous value is in relation to its frequency, e.g. it refers to the last value before the current one with the same frequency.

## Numeric Input Value

Depending on conversions, factors, input frequency or even defaults, the input and the reporting values are handled separately.

A unit is part of the input value, and an arbitrary number of values. When the input and the reporting frequencies are the same, then there is a single value, otherwise the appropriate input indices will be stored in the Numeric Input Value.

**Note:** The input value is usually only shown when editing an indicator value, and not as part of compact display such as in the Data Grid.

> An indicator has a default value set, and otherwise input and reporting attributes identical. Whenever the input is empty the default value will be returned as reporting value.

> An indicator is reported per year but a site chose to input it monthly. The input value will consist of all twelve months of the year, but no quarters.

## Boolean Value

Simple true/false or yes/no values are represented using a boolean value.

**Note:** Boolean values (“yes”/“no”) cannot be consolidated. However, the *Boolean Indicator* type supports emitting numbers as well, which are then consolidated.

## Text Value

Text fragments, which are not to be confused with comments, can be stored as text value.

## Comment

Persistent values may carry comments, which serve to explain the value entered.

By default, the comment is optional, but some situations may require a comment, for instance when a plausibility check fails.

Additionally, the comment can also be changed when the value is no longer in the *open* state depending on the permissions.

**Note:** Having the permission to set the value of an indicator implies the permission to set the comment, but the permission to set comments can be controlled separately.

**Note:** Not all persistent indicator types support comments.

## Previous Comment

In *Data Entry*, the previous comment is shown.

The previous comment is in relation to its frequency, e.g. it refers to the comment of the last value before the current one with the same frequency.

## File Attachment

Any number of files of any type may be attached to values.

When values are consolidated, aggregated or otherwise computed, the resulting value has no File Attachments, but related File Attachments may be searched.

## Data Quality

Each value can carry a quantitative information about its quality.

By default, the quality may be (best to worst) *Measured*, *Calculated* or *Estimated*. When values are consolidated, aggregated or otherwise computed, the resulting Data Quality is the lowest Data Quality of all its contributing values.

## Plausibility Error

Data quality is essential to SERAM, and the plausibility rules allow to add automated data verification to indicators. Whenever a value does fail the plausibility check, a plausibility error message will be added to the indicator value.

## Forecasted

Some values are entered as estimated forecasted values, or otherwise with the possibility of future change. These values are then marked as being forecasted.

Since a forecasted value is designed to be changed some time in the future, special restatement (re-opening) rules exist. On one side the permissions for this type of restatement may be granted to other user groups, on the other side batch operations are provided to restate many indicators across multiple structure nodes in one operation.

When values are consolidated, aggregated or otherwise computed, the resulting value is marked as forecasted if any of the contributing values were forecasted.

## Completeness

Computed, aggregated and consolidated values may be the product of incomplete data. The Completeness tracks how much of the source data was available for computing the value.

If no data is available (e.g. all empty or no value), the completeness will be 0. If all data is available it will be 1. Whenever the data was partial, such as 3 of 4 computation values were available, the completeness will reflect this ration (e.g. 0.75).

**Note:** Partial values are typically emphasized using an italic font, while complete values are shown in the regular font.

## Created By User

On every status change, a value history record is created. The user who triggered the status change will be set as the creator of the value with the new status.

This property will be empty when values have not gone though a status change yet, and it cannot be modified manually at any time.

## Edited By User

Whenever a user makes a change to a value, he or she will be listed as the Edited By user.

In consequence and in contrast to the Created By User property, this property may change over time.

## Version

The version of an indicator value is an automatically computed revision number.

The version is increased every time a status change is performed. When values are consolidated, aggregated or otherwise computed, the resulting Data Quality is the sum of the versions of all its contributing values.

**Note:** The version is not shown in the user interface.

# Chapter 7

# Units

The units are describing what type of data the value represents. SERAM supports a wide range of standard (SI) units which have pre-defined conversions, but it allows the addition of any number of custom units and conversions.

**Note:** When working with units, the term *Quantity* refers to the class of the unit, such as mass or length, since this refers to the physical quantity in the *scope of SI units*.

## Create a Custom Unit

New units can be created in this task.

1. Switch to the *Units* module.
2. Click the **Add a new unit** action (located in the **Units** category).
3. In the new view, enter the unit name and symbol.
4. Create the new custom unit by clicking **Save**.

## Unit Conversions

When the input unit is different from the reporting unit, general conversions between the units are applied.

These conversions are universal and therefore they can only be used for conversions which are always correct, regardless of the circumstances.

**Note:** For contextual conversions, such as when converting currencies or emission factors, you have to use a *Factor Indicator* instead of a unit conversion.

### View Unit Conversions

Conversions of a specific unit can be viewed using this task.

1. Switch to the *Units*.
2. Select the unit in the list.

   **Note:** The list can quickly be narrowed down to the relevant tags by typing text fragments into the **Search** field above the list header.
3. Click on the **View conversions for ...** action in the “Selected Unit” category.

All conversions from this unit to another unit will be shown.

**Note:** Custom conversions are uni-directional, therefore even if a custom conversion A to B is defined, you will not see a conversion B to A unless it is also added.

### Create a Custom Conversion

New custom conversions can be created using this task.

You are viewing the custom conversions list of a unit, as described in the *View Unit Conversions* task.

1. Click on the **Add a new conversion** action in the “Conversions” category.
2. On the new view, pick the destination unit.
3. Optional: Enter a description for this conversion.
4. Enter a factor for the conversion, and (if required) an offset. The offset 0 will be used if left empty.

The conversion is always applied as follows (assuming that the “source” value is using the “from” unit, and the “target” value is of the “to” unit):

```
target := (source * factor) + offset
```

**Note:** A factor `0` is invalid.

5. Click the **Save** button on the bottom to save the new conversion.

# Chapter 8

# Methods

SERAM can compute results for different methodologies (methods) by applying different factor sets to the configured calculations.

## Method

A *Method* is used to represent a reporting methodology, such as Greenhouse Gas (GHG) emissions, Primary Energy (PE) or Environmental Impact Points (EIP).

A common calculation pattern in reporting is to take an input multiplied by a factor to get a result, and to sum these results to get a total. The actual calculations for different methodologies usually follow this pattern and are in fact identical except for which factor is used as multiplier, and what the result unit of the calculations is.

Another use case is when the same methodology is to be used with different factor sets, for instance different based on different factor databases (or versions thereof) such as ecoinvent 2 and ecoinvent 3, or factors mandated by a specific organization or nation.

Yet another potential use case is to compute the actual emissions of different greenhouse gases, e.g. not only $CO_2$ equivalents, but rather actual $CO_2$, $CH_4$, $N_2O$, $SF_6$, $NF_3$, HFCs and PFCs as well.

### Method Part

The result of a *Method* may actually be the sum of multiple *Method Parts*, which are dependent of the methodology.

For reports following the GHG protocol, the parts would be the Scopes 1, 2 and 3; their sum represents the total emissions of the reporting entity. For Primary Energy, one may define renewable and non-renewable as parts.

**Note:** Each*Method* must have at least one part.

## Creating a Method with Method Parts

New *Methods* are created using this task.

1. Switch to the *Units* module.
2. Click the **View Methods** action (located in the **Methods** category).
3. Click the **Add Method** action (located in the **Methods** category).
4. Enter the code for the method. The code has to be unique across the tenant.
5. Enter the name and description for the method.
6. Optional: In the **Translate** category, you may translate the name and the description to other languages.
7. Add *Method Parts* by repeating the following steps:
   a. Click the **Add Method Part** action (located in the **Method Parts** category).
   b. Enter the code for the part. The code has to be unique across the method (but may be the same as in other methods).
   c. Enter the name and description for the part.
   d. Optional: In the **Translate** category, you may translate the name and the description to other languages.
   e. Append the part by clicking **Apply**.
8. Optional: Change the position at which the new method shall be inserted.
9. Create the new *Method* by clicking **Save**.

## Editing a Method and its Method Parts

Existing *Methods* and/or their *Method Parts* are edited or reordered using this task.

1. Switch to the *Units* module.
2. Click the **View Methods** action (located in the **Methods** category).
3. Select the *Method* in the list.
4. Click the **Edit Method ...** action (located in the **Selected Method** category).
5. Optional: Modify the code for the method. The code has to be unique across the tenant.
6. Optional: Modify the name and description for the method.
7. Optional: In the **Translate** category, you may translate the name and the description to other languages.
8. Optional: Add *Method Parts* by repeating the following steps:

   **Note:** Existing *Method Factors* will have a value of `0.0` for the new part.

   a. Click the **Add Method Part** action (located in the **Method Parts** category).
   b. Enter the code for the part. The code has to be unique across the method (but may be the same as in other methods).
   c. Enter the name and description for the part.
   d. Optional: In the **Translate** category, you may translate the name and the description to other languages.
   e. Append the part by clicking **Apply**.
9. Optional: Edit *Method Parts* by repeating the following steps:

   a. Select the *Method Part* in the list.
   b. Click the **Edit Method Part ...** action (located in the **Select Method Part** category).
   c. Optional: Modify the code for the part. The code has to be unique across the method (but may be the same as in other methods).
   d. Optional: Modify the name and description for the part.
   e. Optional: In the **Translate** category, you may translate the name and the description to other languages.
   f. Apply the changes of the part the part by clicking **Apply** (this does not yet save the changes).
10. Optional: Remove *Method Parts* by repeating the following steps:

    **Warning:** Existing *Method Factors* will lose the value of the deleted part, thereby changing their total factor sum. You cannot restore these destroyed values by re-adding a *Method Part*, even if it has the same code.

    a. Select the *Method Part* in the list.
    b. Click the **Delete Method Part ...** action (located in the **Select Method Part** category).
11. Optional: Reorder the *Method Parts* using the **Move Method Part ... ...** actions (located in the **Select Method Part** category).
12. Optional: Change the position of the method.
13. Save the *Method* and its parts by clicking **Save**.

## Deleting a Method

Existing *Methods* can be deleted using this task.

1. Switch to the *Units* module.
2. Click the **View Methods** action (located in the **Methods** category).
3. Select the *Method* in the list.
4. Click the **Delete Method ...** action (located in the **Selected Method** category).
5. Confirm the deletion.

**Warning:** The method and all associated *Method Parts* and *Method Factors* will be irrevocably deleted.

# Method Factor

Each *Method* has a database of associated factors called *Method Factors*.

The *Method Factors* of a *Method* are specific to a *Structure* (*Structure Type* or *Structure Node*), an *Indicator*, and valid starting from a specific date.

Each factor has a value for each *Method Part* of the *Method*; the total factor for the *Method* is always the sum of the partial factors. The partial factors and the total share the same *Units*.

**Note:** The *Unit* is always common to all *Method Factors* of the one *Indicator* of a *Method*.

In order to use the factors defined in the *Method Factors* database, a matching *Method Factor Indicator* must be activated.

## Method Factor Matching

For *Indicator Values* of a *Method Factor Indicator*, specific rules are applied to identify the *Method Factor* that is to be used as source for the (total or partial) factor values.

### Indicator

The referenced *Indicators* of the *Method Factor* and the activated *Method Factor Indicator* must match.

### Time

The latest *Method Factor* of a given *Reporting Period* is used for the whole period of the activated *Method Factor Indicator*. SERAM therefore uses the end date of the *Reporting Period* as reference date for searching Method Factors.

**Note:** Even if different *Method Factors* with validity dates within the *Reporting Period* start and end dates have been added to the database, only the last one will be used even when requesting data for a non-year frequency such as a month. The reason for this behavior is to avoid inconsistent time aggregation, e.g. that the sum of all values of a non-yearly *frequency* differs from the yearly total.

### Structure

The following algorithm is used to find the matching *Method Factor* structure:

**Exact Match**

If there is a *Method Factor* exactly matching the structure of the *Indicator Value*, this factor is used.

**Same Scope Structure Type**

If the *Scope Structure* of the activated *Method Factor Indicator* matches the structure of the *Indicator Value*, factors of parent structures nodes are considered (up to and including the *Scope Structure*).

**Note:** This allows for specifying unspecific and specific factors, such as a fallback global factor and country-specific factors where known.

**Different Scope Structure Type**

If the *Scope Structure* of the activated *Method Factor Indicator* does not match the structure of the *Indicator Value*, a direct *Structure Link* to the *Scope Structure Type* is used to get the *Method Factor*.

**Note:** The applicable *Structure Date* for the structure relationships (both parents and links) is the end date of the *Reporting Period* (same as described in the *Time* section).

# Manage Factors of a Method

Use this task to view and manage the *Method Factors* for a specific *Method*.

1. Switch to the *Units* module.
2. Click the **View Methods** action (located in the **Methods** category).
3. Select the *Method* in the list for which you want to manage the *Method Factors*.
4. Click the **View Factors of Method ...** action (located in the **Selected Method** category).

You are now on the *Method Factor* view for the selected *Method*.

Each *Structure Type* has a tab with its method factors.

## Adding a Method Factor

Add a new *Method Factor* using this task.

You must be on the *Method Factors* view for a *Method*, see task *Manage Factors of a Method*.

1. Click the **Add Method Factor** action (located in the **Method Factors** category).
2. Select the *Indicator* to be associated with the new *Method Factor*.

   **Note:** Only indicators which have been activated as *Method Factor Indicator* in any *Reporting Period* are listed here.
3. Select the *Structure* (*Structure Type* or *Structure Node*) to be associated with the new *Method Factor*.

   **Note:** The structure resolution is described in the topic *Method Factor Matching*.
4. Select the start date when this factor is valid.

   **Note:** The date resolution is described in the topic *Method Factor Matching*.

   **Note:** When adding multiple *Method Factors*, SERAM will suggest the last used date.
5. Enter each *Method Part*'s factor.

   **Note:** No *unit* can be selected here because the unit is common to all *Method Factors* of the same *Method* and *Indicator*. To configure the unit, use the task *Configure Unit or Single Mode of Method Factors*
6. Optional: Enter the comment to be used on *Indicator Values* displaying this *Method Factor*.
7. Optional: Enter the reference for this *Method Factor* (only informative).
8. Save the *Method Factor* by clicking **Save**.

## Editing a Method Factor

Edit an existing *Method Factor* using this task.

You must be on the *Method Factors* view for a *Method*, see task *Manage Factors of a Method*.

1. Select the *Method Factor* you wish to edit.

   **Note:** You cannot change the *Indicator*, *Structure* or validity date of a *Method Factor*. To do so, add and delete *Method Factors* as needed.
2. Click the **Edit Method Factor** action (located in the **Selected Method Factor** category).
3. Optional: Modify the *Method Part*'s factors as needed.

   **Note:** No *unit* can be selected here because the unit is common to all *Method Factors* of the same *Method* and *Indicator*. To configure the unit, use the task *Configure Unit or Single Mode of Method Factors*
4. Optional: Modify the comment to be used on *Indicator Values* displaying this *Method Factor*.
5. Optional: Modify the reference for this *Method Factor* (only informative).
6. Save the *Method Factor* by clicking **Save**.

## Delete a Method Factor

Delete an existing *Method Factor* using this task.

You must be on the *Method Factors* view for a *Method*, see task *Manage Factors of a Method*.

1. Select the *Method Factor* you wish to delete.
2. Click the **Delete Method Factor** action (located in the **Selected Method Factor** category).

3. Confirm the deletion on the prompt.

## Configure Unit or Single Mode of Method Factors

Configure the *Indicator* in a *Method* using this task.

You must be on the *Method Factors* view for a *Method*, see task *Manage Factors of a Method*.

1. Select a *Method Factor* with the *Indicator* for which you want to change the *Unit*.
2. Click the **Configure Indicator ...** action (located in the **Selected Indicator** category).

   **Note:** Within the *Method*, the *Unit* and *Single Mode* is common to all *Method Factors* with the same *Indicator*.
3. Optional: Select the *Unit* to be used for this *Indicator*.
4. Optional: Toggle the *Single Mode* to be used for this *Indicator*.
5. Save the *Indicator* configuration by clicking **Save**.

All *Method Factors* in this *Method* with the same *Indicator* now use the new *Unit* and *Single Mode*.

## Exporting Method Factors to Excel

Export all *Method Factors* of a *Method* using this task.

You must be on the *Method Factors* view for a *Method*, see task *Manage Factors of a Method*.

1. Click the **Export Method Factors to Excel** action (located in the **Method Factors** category).

   **Note:** Only the *Method Factors* of the currently selected *Structure Type* will be exported to the Excel file. If you want to export all *Method Factors*, press the `Shift` key on the keyboard when clicking the action.
2. Save the download.

The downloaded Excel file contains the *Method Factors* for the current *Method* as factor parts.

## Importing Method Factors from Excel

Import *Method Factors* into the current *Method* using this task.

You must be on the *Method Factors* view for a *Method*, see task *Manage Factors of a Method*.

1. Click the **Import Method Factors to Excel** action (located in the **Method Factors** category).
2. Select (or drop) the Excel file containing the *Method Factors*.
3. Optional: Select the **Replace All** checkbox to remove any *Method Factor* (of the current *Method*) which is not contained in the imported file.

   **Note:** Matching is done by code, therefore a file may be imported to another *Method* (even from another tenant) as long as the *Method Parts* match.
4. Optional: Select the **Replace Unit Configuration** checkbox to replace the *Method Factor Unit* configuration. (of the current *Method*)
5. Optional: Select the **Replace Single Configuration** checkbox to replace the *Method Factor Single Mode* configuration (of the current *Method*).
6. Optional: Select the **Remove Redundant Factors (identical but newer)** checkbox to remove any *Method Factor* entries (of the current *Method*) which are identical to another entry except for the *From* date.
7. Import the *Method Factors* using the **Import** button.

   **Note:** The columns of the file to be imported must match the method. If the columns do not match, a message will appear with the information about the expected columns.

   **Note:** Any *Indicator* or *Structure* which cannot be matched to an existing entity will be ignored. A message will inform you of the skipped entities.

The *Method Factors* contained in the file are now in the system.

# Chapter 9

# Managing Reporting Periods

## Reporting Period Types

Depending on the reporting needs, SERAM allows using different reporting period types.

While the most common reporting is based on the Fiscal Year model, other models are used depending on business, country and industry.

**Warning:** Due to the difference in the layout of the different reporting periods, it is not possible to change reporting period types after creation without performing an involved data migration at the database level. Also, the interaction between different reporting period types is limited. For instance, SERAM will not be able to find previous values across reporting period boundaries if they have different types. Please do take this into consideration when choosing a reporting period type.

### Time Indices

Each time span of a reporting period is assigned an index, which is then used throughout SERAM to identify the time span of values.

The different time indices for a hierarchy, where the first index (0) is always the year.

### Reporting Period Types

**Note:** Not all Reporting Period Types are available for selection in the default setup. The SERAM supplier will add these on request.

**Fiscal Year**

The *Fiscal Year* is the most common and intuitive reporting period type.

**Note:** The start of a reporting period can be any time, so that the 1st month may be different from January.

**Note:** Since week and month boundaries do usually not match, the months cannot be broken down into weeks in the Fiscal Year.

| Year | Quarter | Month | Description |
|---|---|---|---|
| 0 | | | Year |
| | 1 | | 1st Quarter |
| | | 5 | 1st Month |
| | | 6 | 2nd Month |
| | | 7 | 3rd Month |
| | 2 | | 2nd Quarter |
| | | 8 | 4th Month |
| | | 9 | 5th Month |
| | | 10 | 6th Month |

| Year | Quarter | Month | Description |
|---|---|---|---|
| | 3 | | 3rd Quarter |
| | | 11 | 7th Month |
| | | 12 | 8th Month |
| | | 13 | 9th Month |
| | 4 | | 4th Quarter |
| | | 14 | 10th Month |
| | | 15 | 11th Month |
| | | 16 | 12th Month |

### Gregorian Year

The Gregorian Year represents the *Gregorian Calendar* with only the year and the months.

**Note:** The start of a reporting period can be any time, so that the 1st month may be different from January.

| Year | Month | Description |
|---|---|---|
| 0 | | Year |
| | 1 | 1st Month |
| | 2 | 2nd Month |
| | 3 | 3rd Month |
| | 4 | 4th Month |
| | 5 | 5th Month |
| | 6 | 6th Month |
| | 7 | 7th Month |
| | 8 | 8th Month |
| | 9 | 9th Month |
| | 10 | 10th Month |
| | 11 | 11th Month |
| | 12 | 12th Month |

### 4-4-5 / 4-5-4 / 5-4-4 Calendar Year

The *4–4–5 calendar* and its siblings divide a year into 4 quarters of 13 weeks each. These are then grouped into two 4-week “months” and one 5-week “month” arranged according to its name.

**Note:** This reporting period type is unsuitable for comparing months, since one month in each quarter has 25% more than the other two months.

**Note:** The following table shows the 4-4-5 layout. The week numbers are partially shifted to different months in other layouts.

| Year | Quarter | Month | Week | Description |
|---|---|---|---|---|
| 0 | | | | Year |
| | 1 | | | 1st Quarter |
| | | 5 | | 1st Month |
| | | | 17..20 | 4 Weeks |

| Year | Quarter | Month | Week | Description |
|---|---|---|---|---|
| | | 6 | | 2nd Month |
| | | | 21..24 | 4 Weeks |
| | | 7 | | 3rd Month |
| | | | 25..29 | 5 Weeks |
| | 2 | | | 2nd Quarter |
| | | 8 | | 4th Month |
| | | | 30..33 | 4 Weeks |
| | | 9 | | 5th Month |
| | | | 34..37 | 4 Weeks |
| | | 10 | | 6th Month |
| | | | 38..42 | 5 Weeks |
| | 3 | | | 3rd Quarter |
| | | 11 | | 7th Month |
| | | | 43..46 | 4 Weeks |
| | | 12 | | 8th Month |
| | | | 47..50 | 4 Weeks |
| | | 13 | | 9th Month |
| | | | 51..55 | 5 Weeks |
| | 4 | | | 4th Quarter |
| | | 14 | | 10th Month |
| | | | 56..59 | 4 Weeks |
| | | 15 | | 11th Month |
| | | | 60..63 | 4 Weeks |
| | | 16 | | 12th Month |
| | | | 64..68 | 5 Weeks |

### Equal Month Year

The *International Fixed Calendar* is very suitable for comparing monthly and weekly data, but it uses 13 months instead of the 12 of the Gregorian calendar.

| Year | Month | Week | Description |
|---|---|---|---|
| 0 | | | Year |
| | 1 | | 1st Month |
| | | 14..17 | 4 Weeks |
| | 2 | | 2nd Month |
| | | 18..21 | 4 Weeks |
| | 3 | | 3rd Month |
| | | 22..25 | 4 Weeks |
| | 4 | | 4th Month |

| Year | Month | Week | Description |
|---|---|---|---|
| | | 26..29 | 4 Weeks |
| | 5 | | 5th Month |
| | | 30..33 | 4 Weeks |
| | 6 | | 6th Month |
| | | 34..37 | 4 Weeks |
| | 7 | | 7th Month |
| | | 38..41 | 4 Weeks |
| | 8 | | 8th Month |
| | | 42..45 | 4 Weeks |
| | 9 | | 9th Month |
| | | 46..49 | 4 Weeks |
| | 10 | | 10th Month |
| | | 50..53 | 4 Weeks |
| | 11 | | 11th Month |
| | | 54..57 | 4 Weeks |
| | 12 | | 12th Month |
| | | 58..61 | 4 Weeks |
| | 13 | | 13th Month |
| | | 62..65 | 4 Weeks |

## Reporting Period

All activated indicators and all data is bound to reporting periods.

In order to start activating specific *Indicator Types* and collect data, a reporting period must be created. When created, you have to pick one of the available *Reporting Period Types*, and a start date for the period.

**Important:** The key date of a reporting period is always the end of the periods' last day. This affects all operations where the date of a reporting period is used, especially when resolving the structures for determining availability of structure nodes, and their relationships when computing data.

**Note:** A reporting period is determined by its type, year and month. This combination must be unique per tenant. It is however possible to create multiple reporting periods which overlap when the month or the type differ. In places where SERAM needs to use data from a previous or following reporting period, it will search a compatible period and use the year difference as offset. A period is compatible when both the type and the start month match up.

### Add a Reporting Period

A new, empty reporting period is created using this task.

**Warning:** The selections made during the creation of the reporting period cannot be changed later.

1. Switch to the *Reporting Periods*.
2. Click the **Add a reporting period** action (located in the “Reporting Periods” category).
3. Choose the *period index type*, the year and the start month.
4. Click the **Save** button on the bottom to create the new period.

## Copy a Reporting Period for a new reporting year

Typically, the configuration of an existing reporting period shall be carried over to the next reporting year. This is done by copying the reporting period.

Near the end of a reporting period, a new reporting period has to be created for collecting the next year's values. Instead of re-defining every single indicator and indicator assignment, reporting periods can be copied to provide the same configuration as baseline for the next year.

1. Switch to the *Reporting Periods*.
2. In the list, select the reporting period to be copied.
3. Click the **Copy reporting period** action (located in the “Selected Reporting Periods” category).
4. Choose the year for the new period.

   **Note:** The reporting period type and the start month are carried over and cannot be changed. This is required for consistency and continuity reasons.

5. Click the **Copy Period** button on the bottom to create the new period.

   **Note:** The copy process can take a few minutes to complete. Please be patient.

The new period contains the same configuration as the reporting period used as template, but the two periods are completely independent. Therefore, making any change to either period after the copy will not affect the other period.

Any formula using values from other reporting periods will be adjusted to use the same relative offset, e.g. 1 year back will remain 1 year back.

No data is copied except for factors stored in any *Factor Indicator*.

## Delete a Reporting Period

Delete an existing reporting period using this task.

1. Switch to the *Reporting Periods*.
2. In the list, select the reporting period to be deleted.
3. Click the **Delete** action in the “Selected Reporting Period” category.
4. Confirm the security message.

# Chapter 10

# Managing Structures

Structures represent the environment where SERAM is used, independent of any reporting periods or data. They are used to carry information such about the organization and its changes over time, assignments to regions, finance codes and other relations, and also serve as base for defining indicators.

## Scenario

Structure information can be grouped into different scenarios.

SERAM allows the definition of an arbitrary number of scenarios to customize the data analysis process. The first scenario is the default scenario which is used initially in the *Data Grid* when loading data.

Each *Structure Node* can be part of any number of scenarios.

| **Mode** | **Description** |
|---|---|
| Exclude | All *Structure Nodes* are in the scenario, even deleted structures or structures being created in the future. *Structure Nodes* can be **excluded** from the scenario. |
| Include | *Structure Nodes* must be **included** explicitly in order to be part of a scenario. |

### Create a Scenario

Use this task to create an additional scenario.

1. Switch to the *Structures*.
2. Click on **View Scenarios** in the “Structure Scenarios” category.
3. Click the **Add a new scenario** action in the “Structure Scenarios” category.
4. In the new view, enter the scenario's code. It has to be unique across the tenant and it may be up to 50 characters long.
5. Optional: Enter a name and description. If required, you can translate the name and description using the action in the “Translations” category.
6. Optional: Change the order of the structure to define its placement. By default, new scenarios are inserted at the end.
7. Choose the mode of the scenario.

   **Exclude**

   All current, past and future Structure Nodes, except for the ones excluded, are part of the scenario.

   **Include**

   Structure Nodes must explicitly be included to be part of the scenario.
8. Create the new scenario by clicking **Save**.

Include or exclude the structure nodes for the new scenario.

### Edit a Scenario

The code, name and description (including translations), the mode and the order of a scenario can be edited using this task.

1. Switch to the *Structures*.
2. Click on **View Scenarios** in the “Structure Scenarios” category.
3. Click the **Edit ...** action in the “Structure Scenarios” category.
4. Make the changes to the code, name, description and order.

   **Note:** The code has to be unique across the tenant and it may be up to 50 characters long.

5. Change the mode of the scenario.

   **Exclude**

   All current, past and future Structure Nodes, except for the ones excluded, are part of the scenario.

   **Include**

   Structure Nodes must explicitly be included to be part of the scenario.

   **Invert structure selection**

   When toggling the mode, the structure selection can be inverted, e.g. structures which were included will be excluded and the other way around.

6. Optional: Manage the translations of the name and description by using the action in the “Translations” category.
7. Click the **Save** button on the bottom to save the changes.

## Delete a Scenario

Delete an existing scenario using this task.

At least one scenario must exist in SERAM. If you want to return to the initial default (one *Exclude* scenario without any exclusions), create such a new scenario first.

1. Switch to the *Structures*.
2. Click on **View Scenarios** in the “Structure Scenarios” category.
3. Select the scenario you want to delete.
4. Click the **Delete ...** action in the “Structure Scenarios” category, and confirm the deletion.

# Structure Type

Each of the structures is based on a type, such as organization or regions.

SERAM allows the definition of an arbitrary number of structure types, so that it can be customized to match any environment.

The types' order can be controlled manually, and the first type is used as the default structure type of SERAM. By default, this is the Organization type. However, the functionality available on the different structure types is equal, unrelated of their order.

## Create a Structure Type

Use this task to create an additional structure type.

**Note:** This task is not suitable for creating a *Structure Level* or *Structure Node*. Please refer to their respective tasks for creating one of these.

1. Switch to the *Structures*.
2. Click the **Add a new type** action in the “Structure Types” category, and confirm the notice dialog.
3. In the new view, enter the type's code. It has to be unique across the tenant and it may be up to 50 characters long.
4. Optional: Enter a name. If required, you can translate the name using the action in the “Translations” category.
5. Optional: Change the order of the structure to define its placement. By default, new types are inserted at the end.
6. Create the new structure type by clicking **Save**.

Make sure that appropriate permissions are set on the new structure type.

**CAUTION:** Users may have no access to any structure node after the creation of a new type if the permissions are not properly set up. For details, please read the *Structure Permissions* topic.

## Edit a Structure Type

The code, name (including translations) and the order of a structure type can be edited using this task.

1. Switch to the *Structures*.
2. Click on the tab header of the structure type you want to edit.
3. Click the **Edit ...** action in the “Structure Types” category.
4. Make the changes to the code, name and order.

   **Note:** The code has to be unique across the tenant and it may be up to 50 characters long.

5. Optional: Manage the translations of the name by using the action in the “Translations” category.
6. Click the **Save** button on the bottom to save the changes.

## Delete a Structure Type

Delete an existing structure type using this task.

Only Structure Types which have no nodes (even historical) may be deleted. You may need to undo the creation of any nodes in order to succeed with the deletion.

1. Switch to the *Structures*.
2. Click on the tab header of the structure type you want to delete.
3. Click the **Delete ...** action in the “Structure Types” category, and confirm the deletion.

# Structure Level

Every structure type has at least one level. The levels are like ordered sub-types of the corresponding structure type.

Each *Structure Node* will be assigned to exactly one level, thereby organizing the structure nodes in a type even when no tree has been built. Also, *Persistent Indicators* will be collecting their data at a specific level of a structure.

> An Organization type may be using levels such as Group, Division, Business Unit and Site, while a geographic type may be using levels such as World, Continent, Region and Country.

## Create a Structure Level

Use this task to create an additional structure level to a structure type.

**Note:** This task is not suitable for creating a *Structure Node*. Please refer to the respective task for creating one of these.

1. Switch to the *Structures*.
2. Click on the tab header of the structure type to which the level shall be added.
3. Click the **Add a level to ...** action in the “Structure Types” category.
4. Choose the insert position for the new level.

   **CAUTION:** The insert position cannot be changed after creation, since this would break the parent and child relationships of any contained node.

5. In the new view, enter the level's code. It has to be unique across its structure type and it may be up to 50 characters long.
6. Optional: Enter a name. If required, you can translate the name using the action in the “Translations” category.
7. Create the new structure level by clicking **Save**.

## Edit a Structure Level

The code and name (including translations) of a structure level can be edited using this task.

1. Switch to the *Structures*.
2. Click on the tab headers of the structure type and level you want to edit.
3. Click the **Edit ...** action in the “Selected Level” category.
4. Make the changes to the code and name.

   **Note:** The code has to be unique across the type and it may be up to 50 characters long.

5. Optional: Manage the translation of the name by using the action in the “Translations” category.
6. Click the **Save** button on the bottom to save the changes.

### Delete Structure Level

Delete an existing structure level using this task.

Only Structure Levels which have no nodes (even historical) may be deleted. You may need to undo the creation of any nodes in order to succeed with the deletion.

1. Switch to the *Structures*.
2. Click on the tab headers of the structure type and level you want to delete.
3. Click the **Delete ...** action in the “Selected Level” category, and confirm the deletion.

### The Structure Level Schema

Each structure level can define a schema for the meta-data of all its nodes.

The schema can define additional fields for addresses, remarks etc..

#### Define a Structure Level Schema

The schema for the meta-data of structure nodes using the level defined using this task.

1. Switch to the *Structures*.
2. Click on the tab headers of the structure type and level you want to edit.
3. Click the **Edit ... schema** action in the “Selected Level” category.
4. Make the changes to the schema by adding, editing, moving or deleting entries.

   **Note:** Removing entries does not delete the actual meta-data from the node. Therefore, you can always re-add an entry with the same key and it will make the still existing meta-data available again.

   **Note:** The group name is only used to generate visual groupings in the UI.

5. Click the **Save** button on the bottom to save the changes.

## Structure Date

The date when operations related to structure nodes and structure links are performed is recorded which allows to keep a full history of the structures.

In consequence, SERAM can compute the effective structure view for any given date. This is essential to keep the historic data consistent and allow for changes in the structures as they naturally evolve.

The Structure Date represents the date for the representation of the structure view. By default the date is “today”, which represents the organization as of the current date, ignoring any future changes, if any. When making changes to structure nodes and links, the Structure Date is used as date for recording the change, allowing the user to make changes in the past, present and future.

**Tip:** Most often, the changes to the structures are recorded in SERAM on a different date than when they are actually taking place. Therefore, make sure to always pick the date of when a change takes place as Structure Date before making any modifications. SERAM will check the date and notify you if you try to make changes at the current date, giving you the opportunity to double-check that the date is indeed correct.

> An old Site is re-assigned to a new Business Unit on February 1st 2012. When choosing a Structure Date before this date, the structure view will still reflect the original Site to Business

Unit relationship. When looking at the current date, the Site will be shown under its new Business Unit.

A Site is deleted by he 1st January of the following year. When looking at the current date, the Site will be visible, but when moving forward in the calendar, the site will disappear by the new year.

### Choose the Structure Date

Change the structure date using this task.

1. Switch to the *Structures*.
2. Change the structure date as desired.

   You can use any mic of the following to do so:

   - By using the date picker labeled **Structures as of**
   - By using the slider underneath the date picker. It changes the date in 1-year increments, always moving to the end of the year (December 31st).

## Structure Node

Structure nodes are the entities which serve as base for data collection and analysis.

All data-related tasks are always in the context of structures, most commonly organization structure nodes. However, data may also be recorded on other structure nodes, for instance when setting emission factors on country structure nodes or when defining currency conversions on currency structure nodes.

For each structure node, the system tracks all changes made by their date, so that a complete history of actions is available. This starts with the creation of a structure node, its linkages to parent nodes and other structure types, and finally the deletion of the structure.

**Tip:** This history is a very powerful feature, and it is essential in SERAM for producing consistent results. However, you have to think of structures as something dynamically evolving over time, and not a static as-is state. It is important to always remember the time dimension represented through the *Structure Date* when making changes to structures, or when performing structure-related operations. For instance, if a structure node is created as per today, this structure node will not be available for assignment or for collecting data in any previous year.

### Create a Structure Node

Use this task to create an additional structure level to a structure type.

The *Structure Type* and *Level* to which the Structure Node shall be added need to exist before starting this task.

1. Switch to the *Structures*.
2. Select the *Structure Date* for the creation of the Structure Node.

   **CAUTION:** The date cannot be changed once the Structure Node has been created.

3. Click on the tab header of the structure type to which the node shall be added.
4. Click the **Add a ... structure node** action in the “Selected Level” category.
5. Enter the code / short name of the Structure Node. It has to be unique across the Structure Type and may be up to 50 characters long.
6. Optional: Enter a name and fill in the meta-data as required. You can translate the name and comment using the actions in the “Translations” category.
7. Create the new structure node by clicking **Save**.

### Edit a Structure Node

Use this task to edit the code and meta-data of a structure node.

**Tip:** Unlike the other operations such as creating, moving, linking or deleting a structure node, the meta-data is not tracked in the history. Any modifications made replace the meta-data for the Structure Node right from the creation date. In some cases, for instance when a company name changes from specific date on, you can manually keep a record of the old information in the comment field.

1. Switch to the *Structures*.
2. Click on the tab header of the structure type containing the node to be edited, then select the structure node.
3. Click the **Edit ...** action in the “Selected Node” category.
4. Make the changes to the code, name and meta-data.

   **Note:** The code has to be unique across the type and it may be up to 50 characters long.

5. Optional: Manage the translations of the name and comment by using the action in the “Translations” category.
6. Click the **Save** button on the bottom to save the changes.

## Move a Structure Node

Use this task to move a structure node to another parent node on a specific date.

1. Switch to the *Structures*.
2. Select the *Structure Date* at which the move shall be recorded.

   **Tip:** If you want to change it from the beginning of the structure node's existence, make sure to pick the node creation date (available by *viewing the history*).

3. Click on the tab header of the structure type to which the node shall be added.
4. Click the **Change parent of ...** action in the “Selected Node” category.
5. Select the new parent for the structure node.

   **Note:** Only structure nodes from higher structure levels are available for selection.

6. Click the **Save** button on the bottom to record the change.

## Delete a Structure Node

Use this task to mark a structure node as deleted on a specific date.

This operation does not destroy the Structure Node. Instead, it will be marked as deleted on the selected Structure Date and therefore become unavailable from that date on. If you wish to destroy the Structure Node, perform the *Undo a Structure Operation* task on the Structure Node creation history entry.

1. Switch to the *Structures*.
2. Select the *Structure Date* at which the delete shall be recorded.
3. Click on the tab header of the structure type containing the node to be edited, then select the structure node.
4. Click the **Delete ...** action in the “Selected Node” category.
5. Confirm the action in the confirmation dialog to record the deletion.

## View a Structures History

Use this task to view the history of a structure node.

1. Switch to the *Structures*.
2. Click on the tab header of the structure type containing the node to be inspected, then select the structure node.
3. Click the **View history of ...** action in the “Selected Node” category.

The history will be displayed in a new view. To return, use the **Back** button on the bottom of the screen.

## Undo a Structure Operation

The operations performed on the Structure Node can be selectively undone.

The *history of a Structure Node* must be shown on screen.

If any operation (except editing meta-data) was performed which is not correct for any reason, this operation can be removed from the history.

**Tip:** The operation undone does not have to be the last operation performed; any operation shown in the history can be undone in any order.

1. Select the operation in the history list.
2. Click the **Undo ...** action in the “Selected Item” category, and confirm the action in the confirmation dialog.

**Note:** If you undo the node creation, all information related to the node including its history will be destroyed, therefore the history view will close.

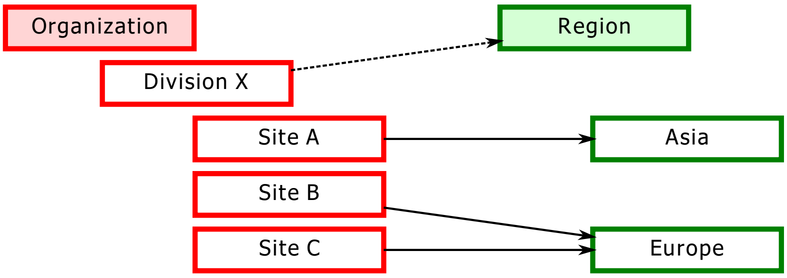
## Structure Permissions

The permissions on structures play a central role for determining what users can see and do in SERAM.

Users will only see the structures to which they have access, so that restricted views to the structures can be created to present limited choices to end users. The less a user has access to, the more SERAM will make default selections for him or her, thereby simplifying tasks such as Data Entry or using the Data Grid.

In order to see a structure, a user must have the Generic:Access *Functions and User Permissions* not just on the structure itself, but also on all linked structures. In an abstract way, the rights on structures of each linked type can be seen as *sets* in the mathematical sense, and the user will have access to the nodes in the intersection of all these sets.

Let's assume the two structure types “Organization” and “Region”. The node “Site A” is linked to “Asia”, the nodes “Site B” and “Site C” are linked to “Europe”, and the “Division X” is not explicitly linked, making it implicitly linked to the “Region” type.



| Organization Permission | Region Permission | Resulting Org. Access |
|---|---|---|
| “Division X” and children | None | None |
| “Division X” and children | “Region” and children | All nodes |
| “Division X” and children | “Asia” | Only “Site A” |
| “Division X” and children | “Europe” | “Site B” and “Site C” |
| “Site A” | “Region” and children | “Site A” |
| “Site A” | “Europe” | None |

**Tip:** Whenever a structure type is not used for permission filtering, set up an inheriting permission granting Generic:Access to “Authenticated Users” on the type. This makes all the nodes of this type accessible and therefore prevents filtering.

## Manage Structure Permissions

Use this task to modify access permissions and audit settings on structure nodes or types.

1. Switch to the *Structures*.
2. Click on the tab header of the relevant structure type.
3. If you want to change the permissions for a specific structure node and not the complete type, select the structure node.
4. Click the **View permissions of: ...** action (located in the “Permissions” category) for the type of node.
5. View and edit the tag permissions as described in the topic *Using the Permissions View*.

# Structure Links

Structure Nodes can be linked to one Structure Node of each other Structure Type. The links enable data flows between these structures.

Just like the other structure operations, the changes to links are recorded with their date, and a complete history remains available.

The links are directional, meaning that linking A to B does not represent a link from B to A.

## Data Flow between Structure Types

When indicators scoped to a different *Structure Type* than the selected *Structure Node* is requested, SERAM will use the *Structure Links* to let the data flow between these *Structure Types*. This process is called *Data Flow*.

There are different ways how data can flow through links:

**Direct: Pull data through outgoing link**

The requested indicator data is taken from the explicitly referenced link to the structure node of the indicator owner structure type.

**Consolidate: Collect and consolidate through incoming links**

The requested indicator data is taken from all structures referencing the current structure node in their link of the indicator owner structure type.

Assume the following structures and reporting indicators:

| Structure Type | Structure Node | Structure Link |
|---|---|---|
| Organization | O1 | Linked to Country C1 |
| Organization | O2 | Linked to Country C1 |
| Country | C1 | *(not linked)* |
| Country | C2 | *(not linked)* |

| Indicator Code | Indicator Scope | Indicator Data Flow |
|---|---|---|
| IO | Organization | Consolidate |
| IC | Country | Direct |

Assume the following values (denoted with a placeholder variable) are stored in the indicators::

| Indicator Code | Structure Node | Value |
|---|---|---|
| IO | O1 | x |
| IO | O2 | y |
| IC | C1 | a |
| IC | C2 | b |

Retrieving data through indicator links will then behave as follows:

| Indicator Code | Structure Node | Value |
|---|---|---|
| IC | O1 | a |
| IC | O2 | a |
| IO | C1 | x + y |
| IO | C2 | *(none)* |

## Link a Structure Node

Use this task to link a structure to structures of other structure types.

1. Switch to the *Structures*.
2. Select the *Structure Date* at which the linking shall be recorded.

   **Tip:** If you want to change it from the beginning of the structure node's existence, make sure to pick the node creation date (available by *viewing the history*).
3. Click on the tab header of the relevant structure type.
4. Select the structure node which shall be linked to other structure nodes.
5. Click the **Structure links of ...** action in the “Selected Node” category.
6. In the new view, select the Structure Type you want to link to.
7. Click the **Set new node link** action (located in the “Selected Item” category).
8. In the new view, select the destination structure node, or None to un-link the structure from the target structure type.
9. Optional: Repeat steps 6-8 to set the links for other structure types using the same structure date.

Chapter

11

# Security Management

SERAM has an extensive and very configurable access control system, allowing exact control over visibility of nodes and indicators as well as controlling what actions can be performed by specific users in the system.

## Functions and User Permissions

When users perform actions, SERAM will check if they are allowed to do so by determining what functions the user may perform on a particular resource.

### SERAM functions

**Note:** The following is the standard set of SERAM functions. Additional functions can be defined for *Status* transitions if required for implementing special work-flows.

**AccessControl:ChangeAllow**

The user may change the Allow flag of *access control entries*.

**Note:** This is only applicable at the tenant level.

**AccessControl:ChangeAudit**

The user may change the Audit flag of access control entries.

**Note:** This is only applicable at the tenant level.

**AccessControl:ChangeDeny**

The user may change the Deny flag of access control entries.

**Note:** This is only applicable at the tenant level.

**Important:** Using Deny permissions is dangerous if their impact is not well understood; an administrator can easily lock himself out of SERAM with them. Therefore, this function is not granted to the built-in “Administrator” role. The administrator can, however, create a new role granting himself this function, enabling the use of Deny permissions.

**AccessControl:ViewPermissions**

The user may look at access control entries on entities (such as structures or indicator tags) and users.

**Note:** Users may always check their own access control entries, even when they have not been granted this function.

**Audit:ViewAudit**

The user may browse the audit log.

**Note:** This is only applicable at the tenant level.

⚠ **Warning:** The audit log may disclose some information such as structure names which the user cannot see otherwise.

**DataAnalysis:ModuleAccess**

The user will see the **Data Grid** navigation entry and will be allowed to access the *Data Grid*.

**DataAnalysis:ValueAnalytics**

The user will see actions to perform analytics in the **Data Grid**, such as Find Restatements or Value Drilldown. Remove this function so simplify the **Data Grid** module for users.

**Note:** This function is only applicable on the tenant level.

**DataEntry:ManageShare**

The user will be able to share *Data Entry* forms.

**Note:** This function is only applicable on the tenant level.

**DataEntry:ModuleAccess**

The user will see the **Data Entry** navigation entry and will be allowed to access the *Data Entry*.

**Generic:Access**

The user is allowed to access (and see) a particular resource. This is used for *structure permission filtering*, and can also serve to enable read-only access to indicators assigned to an *indicator tag*.

**Generic:Query**

The user is allowed to query the application via *Query API*.

**Global:ApplicationAccess**

The user may log on to SERAM and use the web application. Without this permission, the user is not granted access to SERAM.

**Indicators:GetValue**

The user may read *Indicator Values*, not just the metadata of *indicators*.

**Indicators:ManageCache**

The user may trigger a cache compaction, or clear the cache.

**Note:** This function is only applicable on the tenant level.

**Indicators:ManageIndicator**

The user may create, change and delete *Indicators* (metadata and activations).

**Indicators:ManageMethod**

The user may create, change and delete *Methods*.

**Indicators:ManageTag**

The user may create, change and delete indicator tags.

**Indicators:ModuleAccess**

The user will see the **Indicators** navigation entry and will be allowed to access the *Indicators*.

**Indicators:SetByApplication**

External applications are allowed to set data and status information on this indicator.

**Indicators:SetComment**

The user is allowed to change *comments* on *persistent indicators*, but not other parts of the indicator values.

**Indicators:SetValue**

The user is allowed to edit the indicator value on persistent indicators, including the comment.

**Periods:ManagePeriod**

The user may create, delete and perform other actions on *reporting periods*, including making copies or restating the period.

**Periods:ModuleAccess**

The user will see the **Reporting Periods** navigation entry and will be allowed to access the *Reporting Periods*.

**Reports:ManageReport**

The user may create, change and delete *Reports*.

**Reports:ModuleAccess**

The user will see the **Reports** navigation entry.

**Reports:RunReport**

The user may execute a *Report*, filling the variables based on his access permissions.

**Reports:ShareReport**

The user may share a *Rendered Report* as anonymous web link.

**Security:ManageApplications**

The user may create, modify and delete *Applications*.

**Note:** This function is only applicable on the tenant level.

**Security:ManageRoles**

The user may create, modify and delete *Roles*.

**Note:** This function is only applicable on the tenant level.

**Warning:** Any user who has this role can change whatever role is assigned to him and grant himself any function, so that granting this role effectively grants the potential to acquire administrative powers.

**Security:ManageSettings**

The user may configure any settings on the tenant, changing its behavior. This function is only applicable on the tenant level.

**Security:ModuleAccess**

The user will see the **Security Management** navigation entry and will be allowed to access the *Security Management*.

**Selections:Share**

The user may share Saved Selections created in the Data Grid module for other users to use.

**Note:** This function is only applicable on the tenant level.

**Selections:TakeOwnership**

The user may take the ownership of shared “Saved Selections” in the Data Grid, thereby transferring the modification rights to this selection.

**Note:** This function is only applicable on the tenant level.

**Status:Commit**

The user may commit values which are in the Open state, so that they become “Entered”.

**Status:Reject**

The user may reject values which are in the “Entered” state, so that they are Open and editable again, for instance for fixing errors.

**Status:Restate**

The user may restate Verified values back to the Open state, making them editable again.

**Status:RestateForecasted**

The user may restate Verified values back to the Open state, but only if they have been marked as being "*Forecasted*".

**Status:Verify**

The user may verify a Completed value to Verified or reject it and set it back to Open.

**Structures:CreateChildNode**

The user may create new *structure nodes* below the *Structure Node* with that function (especially useful for the *CreateStructure API*).

**Structures:ManageLevel**

The user may create, modify or delete *structure levels*.

**Structures:ManageNode**

The user may create, modify, link or delete *structure nodes*.

**Structures:ManageScenario**

The user may create, modify or delete *Scenarios*.

**Note:** This function is only applicable on the tenant level.

**Structures:ManageType**

The user may create, modify or delete *Structure Type*.

**Structures:ModuleAccess**

The user will see the Structures navigation entry and will be allowed to access the *Structures*.

**Structures:Undo**

The user will be allowed to *undo operations* made to structure nodes.

**Note:** Since this is a destructive operation, the ability to do so is not granted by the Structures:ManageNode permission.

**Tenant:ViewLicenseUsage**

The user gets a license usage and transaction summary on the dashboard.

**Note:** This function is only available on tenants where pay-as-you-go licensing has been enabled.

**Note:** This function is only applicable on the tenant level.

**Units:ManageConversions**

The user may create, modify or delete custom *unit conversions*.

**Note:** This is only applicable at the tenant level.

**Units:ManageUnits**

The user may create, modify or delete custom *units*.

**Note:** This is only applicable at the tenant level.

**Units:ModuleAccess**

The user will see the **Units** navigation entry and will be allowed to access the *Units*.

**Users:ManageUsers**

The user may create, modify or delete user meta-data, and create tenant user reports.

**Note:** This is only applicable at the tenant level.

**Users:ModuleAccess**

The user will see the **Users** navigation entry and will be allowed to access the *Tenant Users*.

## Roles

The roles are used for user permission assignments. Each role represents a set of functions provided by SERAM.

In addition to functions, some meta-data can also be added to roles which controls some special behavior or functionality related to the role.

### Key Person Role

There is a special Key Person role available, which is used to deal with roles outside of the SERAM web application context, such as a “Financial Manager” or “Chief Officer”.

The key person role can be used in access control entries along with the other roles, but it does not affect the effective rights of the user. This role can for instance be used to “tag” structures with people having roles outside of the SERAM web application context, such as a “Financial Manager” or “Chief Officer”. So, people not using SERAM can be represented by meta data such as for example their email address.

## Dashboard Data Entry Access

The Dashboard will show the pending data entry information to people holding specific roles.

This setting controls which roles are to be used for this computation. SERAM will do a look-up of all structures a user has access to and find out on which of these he has been assigned this role. The system then has a list of all structures which need to be inspected for pending data entry for this user, and this information is then transferred to the Dashboard.

**Note:** The functions assigned to the role are irrelevant in regards to this functionality, but in order to provide a meaningful experience for the end user this role setting should only be set to the role(s) used for “Data Encoder” people, including the appropriate functions.

## Automatic Role Permissions Grants

By default, new objects have no access permissions assigned to them. While this does not affect global administrators, it can severely affect all users which do not have full access rights to everything. This is especially true for the creation of new *Structure Type*, which will cause permission filtering to kick in as described in the *Structure Permissions* topic, effectively making all structures unavailable to the non-administrative users upon creation of a new structure type.

To address this and similar issues, SERAM can automatically generate access control entries on several object types when they are created. The definition of the entries to be created are done on the roles that shall be assigned; when the new object is created, SERAM will check all roles for automatic grants on the specific object type.

This definition is textual, it has the form of a comma-separated list of special name tokens, optionally with an additional inheritance specifier:

| Name identifier | Access Control Entry subject | Default inheritance |
|---|---|---|
| Authenticated User | Authenticated Users | This object only |
| Everyone | Everyone | This object only |
| Anonymous | Anonymous | This object only |
| Creator | The user creating the object | This object and children |

The optional inheritance can be set through the following identifiers:

| Inheritance identifier | Access Control Entry inheritance |
|---|---|
| Self | This object only |
| Inherit | This object and children |
| Children | Children only |

```
Creator, Authenticated User:Inherit
```

This would create two Access Control Entries on the new object; one for the user creating the object and one for the Authenticated Users group, both set to This object and children inheritance.

## Create a Role

Use this task to create a new role.

1. Switch to the *Security Management*.
2. Click the **Add Role** action in the “Roles” category.
3. On the new view, enter the name / identifier for the role.

**Warning:** This name / identifier is used internally as key in the permission system. Therefore, it cannot be changed once it has been created.

4. Optional: Enter a description for the tole. You can also translate the description using the action in the translate category..
5. Optional: Select all functions which shall be assigned to this role.

**Note:** If you select no functions, the role will be a *Key Person Role*.

6. If this role shall be used to determine pending Data Entry on the Dashboard, check the **Role used for Data Entry direct access** check-box.
7. Optional: If *Access Control Entries* with this role shall be created for new objects, enter the *Automatic Role Permissions Grants*
8. Create the new role by clicking **Save**.

## Edit a Role

Use this task to edit an existing role.

1. Switch to the *Security Management*.
2. Select the role in the list.

**Note:** The list can quickly be narrowed down to the relevant roles by typing text fragments into the **Search** field above the list header.

3. Click the **Edit** action in the “Selected Role” category.
4. On the new view, make the required changes.

**Note:** The role name / identifier is used internally as key in the permission system and cannot be once it has been created.

5. Save the changes to the role by clicking **Save**.

**Note:** The changes to function assignments are immediately applied by the server, but the client may require a page refresh to reflect the changes to the UI of currently logged on users (e.g. navigation entries for the modules, or pending data entry information).

## Delete a Role

Use this task to delete an existing role.

1. Switch to the *Security Management*.
2. Select the role in the list.

**Note:** The list can quickly be narrowed down to the relevant tags by typing text fragments into the **Search** field above the list header.

3. Click the **Delete ...** action in the “Selected Role” category, and confirm the deletion in the prompt.

**Note:** When a role is deleted, the changes are immediately applied by the server, but the client may require a page refresh to reflect the changes to the UI of currently logged on users (e.g. navigation entries for the modules, or pending data entry information).

**CAUTION:** Deleting a role does not delete associated *Access Control Entries*, leaving stale entries in the system. This is not a problem in general, but if a role with the same name/identifier is re-created at a later point in time, these old access control entries will become applicable again.

## Common Role Definitions

Role configurations which are commonly used in SERAM tenants.

**Access**

This role is usually used in relation to *Structure Permissions* for access filtering (or to prevent filtering when applied on a structure type), or on *Indicator Tags* to provide read-only access to indicators and their values.

| | |
|---|---|
| **Assigned functions** | Generic:Access |

| | |
|---|---|
| **Grant on Structure Type creation** | Authenticated Users:Inherit |

**Administrator**

This built-in role cannot be deleted or modified by the end user. This role includes all SERAM functions but AccessControl:Deny and Indicators:SetByApplication.

**Data Analyst**

This role is used for people analyzing data. They can access SERAM and the *Data Grid*, but they cannot make changes to the data. This role is usually used in access control entries on structure nodes.

| | |
|---|---|
| **Assigned functions** | DataAnalysis:ModuleAccess, Generic:Access, Global:ApplicationAccess |

**Data Encoder**

This role is used for people entering data. Users with this role will see a list of Pending Data Entry tasks on their *Dashboard*, allowing them to quickly access the data sets to be entered. Besides accessing SERAM and the*Data Entry* and *Data Grid* modules, this role allows editing open values and to commit them (set them to entered). This role is usually used in access control entries on structure nodes.

| | |
|---|---|
| **Assigned functions** | DataAnalysis:ModuleAccess, DataEntry:ModuleAccess, Generic:Access, Global:ApplicationAccess, Indicators:SetValue, Status:Commit |
| **Used for Data Entry direct access** | Yes (checked) |

**Data Verifier**

This role is used for people verifying data. Users with this role will see a Data Status Controlling summary on their *Dashboard*. Besides accessing the application and the *Data Grid* modules, this role allows changing the status of entered values either to verified or back to open, and verified values with the forecasted flag set may also be re-set to open. Values may not be edited except for the comment. This role is usually used in access control entries on structure nodes.

| | |
|---|---|
| **Assigned functions** | DataAnalysis:ModuleAccess, Generic:Access, Global:ApplicationAccess, Indicators:SetComment, Status:RestateForecast, Status:Verify |
| **Used for Data Status controlling** | Yes (checked) |

**Indicator Data Editor**

This role is used as default on *Indicator Tags*, since it makes the indicators of the tags editable in general.

**Note:** Users need to be granted appropriate permissions on both the structure and the indicator to be able to perform a data-related function, such as modifying a value or changing the status. Therefore granting this role to all users on an indicator tag will not make these indicators editable for these users unless they also have the corresponding functions through structure permissions, typically in the form of a Data Encoder or Data Verifier role.

| | |
|---|---|
| **Assigned functions** | DataAnalysis:ModuleAccess, DataEntry:ModuleAccess, Generic:Access, Global:ApplicationAccess, Indicators:SetByApplication, Indicators:SetValue, Status:Commit |
| **Grant on Indicator Tag creation** | Authenticated Users |

**Modify (for Audit)**

This role covers all functions modifying non-value data, such as making changes to structures or indicators. It is designed to be used for audit access control entries only, and not for granting permissions.

**Warning:** This role should not be granted to users, as this would grant them near-administrative rights to the system. It is designed to be used as audit-role only.

| | |
|---|---|
| **Assigned functions** | AccessControl:ChangeAllow, AccessControl:ChangeAudit, AccessControl:ChangeDeny, Indicators:ManageIndicator, Indicators:ManageTag, Indicators:SetComment, Indicators:SetValue, Periods:ManagePeriod, Security:ManageRoles, Security:ManageSettings, Structures:ManageLevel, Structures:ManageNode, Structures:ManageType, Structures:Undo, Units:ManageConversions, Units:ManageUnits, Users:ManageUsers |

## Access Control Entry

Access control entries represent single permission entries.

Each entry is a distinct combination of the following properties:

| **Property** | **Description** |
|---|---|
| Kind | Allow, Deny or Audit. Deny always takes precedence over Allow, Audit is always complementary to an Allow (for details see the *Audit Log* topic). |
| Subject | The user, application or group to which this is entry has been assigned. |
| Object | The resource to which this entry applies (such as a *Structure Type*, a *Structure Node* or an *Indicator Tag*), or even the whole tenant. |
| Right | The *role* used for this entry. |
| Apply to | Either the object only, or the object and all its children (descendant objects). |

## Access Control List

The Access Control List is a collection of Access Control entries.

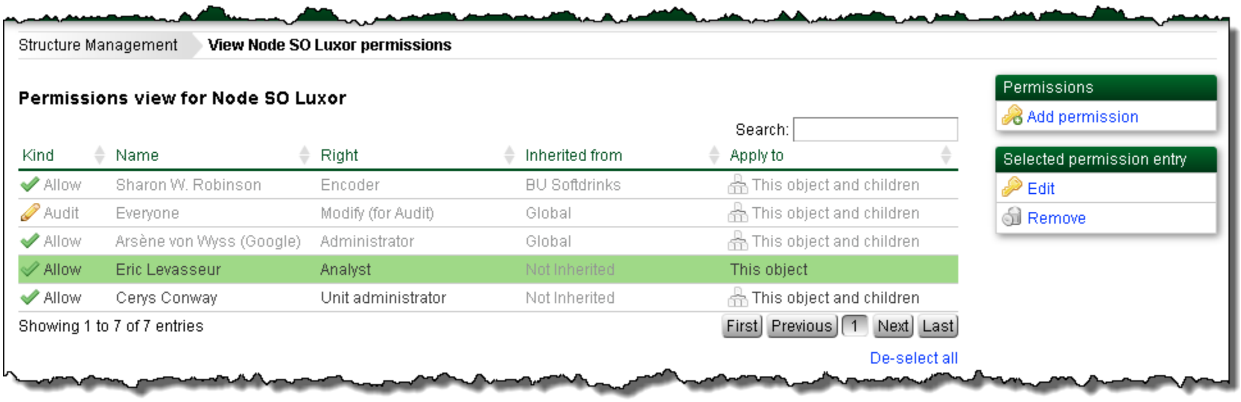
Each permission-sensitive object in SERAM has an Access Control List. The list has explicit entries and may also have inherited entries from objects higher in the permission inheritance tree.

**Note:** In the permission system, all object relationships end on the tenant. Therefore, if an Access Control Entry is created on the tenant level and it is set to apply to its children as well, it will apply to all objects in this tenant and therefore be listed in all Access Control Lists.

## Using the Permissions View

The permissions view displays the Access Control List of an object, and it allows to add entries on that object or to change non-inherited entries of that object.

The permissions view is available for different object types such as structure types, structure nodes, and indicator tags, and also the whole tenant.



**Permissions view for Node SO Luxor**

Search:

| Kind | Name | Right | Inherited from | Apply to |
|---|---|---|---|---|
| ✓ Allow | Sharon W. Robinson | Encoder | BU Softdrinks | This object and children |
| Audit | Everyone | Modify (for Audit) | Global | This object and children |
| ✓ Allow | Arsène von Wyss (Google) | Administrator | Global | This object and children |
| ✓ Allow | Eric Levasseur | Analyst | Not Inherited | This object |
| ✓ Allow | Cerys Conway | Unit administrator | Not Inherited | This object and children |

Showing 1 to 7 of 7 entries

# Managing Tenant Permissions

Use this task to change permissions globally for the whole tenant.

**Note:** Some *SERAM Functions* are only applicable to the whole tenant, and have no effect when granted on other objects (these are described as "only applicable at the tenant level" in the functions reference).

1. Switch to the *Security Management*.
2. Click the **View permissions of: Tenant-wide** action in the "Permissions" category.
3. Manage the global permissions by using the *Permissions View*.

# Audit Log

The configurable audit log allows to record actions performed by users in the tenant.

The audit functionality is tightly integrated into the permission system. Just as users can be granted certain SERAM functions on objects, users can also be audited by using the Audit kind in *Access Control Entries*. When a user performs an action, SERAM will resolve the permissions for that user and not only check if he has been grated the required function(s) through roles, but also whether these function(s) have been marked for audit.

When audit is enabled, the system will record the outcome of the action, be it successful or not, along with a short description of the action performed.

## View Audit Log

Use this task to review the audit log for users or the whole tenant.

1. Switch to the *Tenant Users*.
2. Optional: If you want to look at the audit log for a particular user only, select the user in the list.

   **Note:** The list can quickly be narrowed down to the relevant users by typing text fragments into the **Search** field above the list header.
3. In the "Audit" category, use the appropriate **View ...** action for either the selected user or the whole tenant.
4. Optional: You can use the **Change filter** action in the "Filter" category to narrow down the audit log display.

# Chapter 12

# Data Entry

The *Data Entry* module allows users to submit data of locations for a specific time period. The information reported within this module requires users to select their various reporting criteria and then enter their performance data. Based upon your SERAM permissions, the Data Entry module of SERAM may automatically make selections for you if they are unique.

## Data Entry Wizard

The *Data Entry Wizard* guides you through the required selections:

**Reporting Period**

The time period for which you are entering data. This is most commonly the current reporting year.

**Structure**

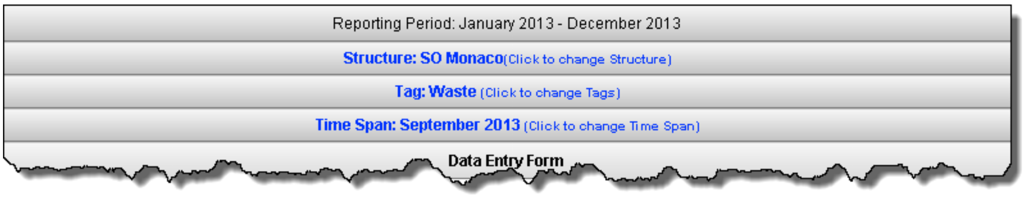
The location for which you wish to report data.

**Indicator Tag(s)**

The indicator tags are typically used to categorize indicators. Therefore, you can use them to assist in filtering the data you wish to report down to a specific set or single indicator. You can select multiple tags to be displayed at once. Typically, the more tags you select at any one time, the more indicators you will be presented with on the *Data Entry Form*.

**Time Span**

This is the year, quarter, month or week for which you are reporting data, the available selections depend on the indicators.



Reporting Period: January 2013 - December 2013

Structure: SO Monaco(Click to change Structure)

Tag: Waste (Click to change Tags)

Time Span: September 2013 (Click to change Time Span)

Data Entry Form

After making these selections, you'll see the indicators that met the criteria selected above displayed in the *Data Entry Form* section.

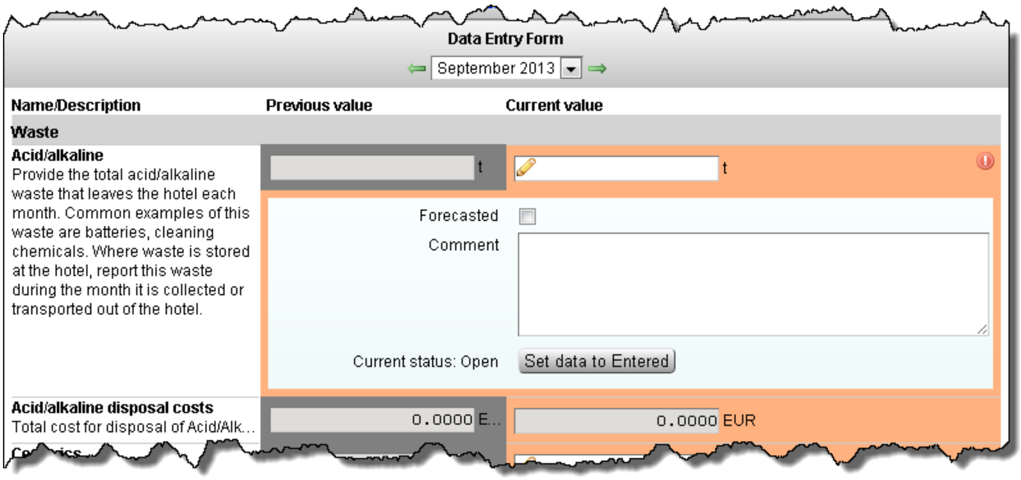
**Note:** To change your selection, you can always return to the selection steps which were not automatically done for you by clicking on the selection heading.

## Data Entry Form

Based upon the information provided in the *Data Entry Wizard*, the *Data Entry Form* enables users to enter their data for the indicators selected.

One indicator and its value is selected and displayed in an expanded form, allowing you to see and edit all facets of the indicator value. In the following screen shot, the first value is selected and expanded. When you select another indicator, the previous selection will be collapsed and the new selection will expand.

**Note:** The system constantly saves the information you have entered in the *Data Entry Form*, so there is no need to click a save button.



Data Entry Form

⇦ September 2013 ▾ ⇨

| Name/Description | Previous value | Current value |
|---|---|---|
| **Waste** | | |
| **Acid/alkaline**<br>Provide the total acid/alkaline waste that leaves the hotel each month. Common examples of this waste are batteries, cleaning chemicals. Where waste is stored at the hotel, report this waste during the month it is collected or transported out of the hotel. | t | t |
| | Forecasted | □ |
| | Comment | |
| | Current status: Open | Set data to Entered |
| **Acid/alkaline disposal costs**<br>Total cost for disposal of Acid/Alk... | 0.0000 E... | 0.0000 EUR |

### Current Value

Users are asked to provide their indicator values in this field. The unit of reporting should always be the local unit, e.g. currency, volume, mass. The pencil icon informs the user that they are able to type values.

**Note:** The value will have a red background when it has been entered but fails the constraint checks of the indicator. It will remain red until the problem has been resolved.

### Previous value

To provide a quick data quality reference, the previous value is displayed. Simply reviewing the current value and the previous value can often help to remove obvious errors.

### Constraint and plausibility checks

A red exclamation mark ❗ or a colored flag ⚑ to the right of the current value informs the user that the value does not meet the data quality checks performed by SERAM on this indicator. The user will not be able to change the data status until this issue has been addressed. In the case of the flag symbol, which indicates an unmet plausibility rule, a comment providing the reason for the non-conformance can be given to enable the status change. By clicking on the symbol, a short explanation of the problem will be displayed.

### Forecasted

Values that are entered but may need to be changed in future are called Forecasted. For example Forecasting is used for values that are estimated and need to be updated once the real values have been obtained at a later date.

### Comment

Users should provide any additional information that relates to the value given, e.g. sources, descriptions, further details etc. It is mandatory to provide a comment for values that fail their plausibility checks (flag symbol).

### Current status, Data Status buttons

Once data has been successfully entered and the user is confident that the value is correct, it is necessary to set the *Data Status* to *Entered*. Clicking the *Set data to Entered* button will change the *Data Status* to entered, if there are no constraint violations. Once an indicator has been set to *Entered*, users will no longer be able edit value or other information. If you set the status to *Entered* by mistake or notice that the value was not correct after all, contact the person in charge of verifying this data or the system administrator. They will reset the status to *Open* for you.

The color coding of the indicator values provides a quick overview of their current data status'. In the default configuration, the colors are the following:

### Orange

The value is *Open* and may be edited.

### Yellow

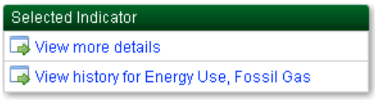
The value has been set to *Entered*, and it is now ready for verification. Editing the value is no longer possible from here on, unless it is re-set to *Open* by the verifier or system administrator.

### Green

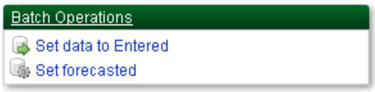
The verifier has accepted the value and set it to the *Verified* status.

### Further actions when entering data

While working in the *Data Entry Form*, some actions are available to you on the right hand side:

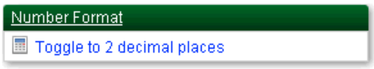


View information relative to the currently selected indicator and its value, e.g. the change history.



Perform the clicked operation for all values shown in the *Data Entry Form* where the operation is applicable. For instance, to mark all *Open* values displayed in the *Data Entry Form* as being forecasted, click the *Set Forecasted* action.

**Note:** What actions are available depends on several factors such as the value, the data status and your permissions.



Toggle the number of displayed decimal places between 2 and 4. This does not affect the storage or entry of values.

Once you have finished entering your indicator data you may return to the wizard at any step to continue entering other data, you may move to another module in the menu bar or you may close the application, either by logging out or closing the browser.

# Anonymous Data Entry

Data Entry Forms can be shared to allow link-based anonymous access to these.

In situations where the Data Entry process shall be delegated to a person who does either not have a regular account, or where the individual person is not known, SERAM allows to create anonymous links to Data Entry Forms.

**Warning:**

Permitting anonymous access to the system is - by design - an insecure method. By the design of the application, people obtaining a link to an anonymous Data Entry may - within the permissions of to the user impersonated by the link - read all metadata (indicators, structures, units, reporting periods...) and perform Data Entry tasks. As such it may disclose more information than intended, and using this feature may violate company policies.

Therefore, the required function is by default not granted to any *Roles* except for the Administrator role. The administrator has to make sure that the function is only used where appropriate, and links should only be handed to trusted parties.

**Note:** When creating anonymous Data Entry links, best security practice is to create a user with minimal required permissions and impersonate this user with the link. The user does not need to have any claims (logins).

## Sharing a Data Entry Form

Use this task to create a link which can be used anonymously to enter data in a Data Entry Form.

You need to have a Data Entry Form loaded in order to share it. Also, sharing is only possible for users which have the tenant-wide `DataEntry:ManageShare` function.

1. Click the **Share Data Entry Form** action in the “Data Entry” category.
2. Optional: Edit the name. This name is only used for managing the link, it is not shown to the user accessing the Data Entry Form anonymously.
3. Optional: Add a description (*Markdown Syntax* supported). This description is shown on the Data Entry Form when accessed via link.

4. Optional: Set an expiration date after which the link is invalid.
5. Optional: Choose the user to impersonate when the link is used. If you don't choose a user, your own account will be impersonated.
6. Click the **Share Data Entry Form** button to share the form and generate a link.
7. Optional: Copy the link from the confirmation dialog to the clipboard for further use. The link cannot be clicked on since you're logged on.

## View Shared Data Entry Form Links

Use this action to access the list of your shared Data Entry Form links.

1. Switch to the *Data Entry*.
2. Click the **View Shared Data Entry Forms** action in the “Data Entry” category.

You can now view, copy and manage your links.

## Configure a Date Entry Form Link

Use this task to configure a previously created Date Entry Form Link.

You must be viewing your shared Data Entry Form links in order to perform this action.

1. Select the link in the list.
2. Click the **Configure Selected Link** action in the “Data Entry” category.
3. On the new view, make the required changes.
4. Save the changes to the role by clicking **Share Data Entry Form**.

## Delete Date Entry Form Links

Use this task to configure a previously created Date Entry Form Link.

You must be viewing your shared Data Entry Form links in order to perform this action.

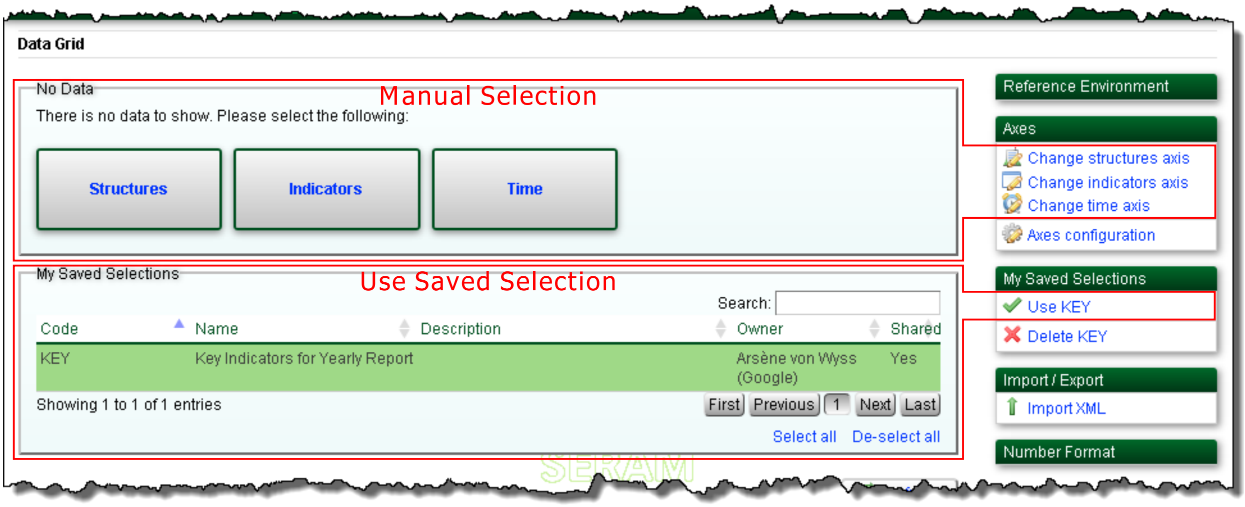
1. Select the links to remove in the list.
2. Click the **Delete Selected Link** action in the “Data Entry” category.

# Chapter 13

# Data Grid / Data Analysis

The *Data Grid* module is used for analysis and presentation of data. Additionally data may also be entered in the Data Grid and batch operations can be carried out, e.g. changing data statuses in large blocks.

The system provides a simple step-by-step process for selecting the data to be loaded into the grid. Your system administrator may also have prepared selections for you, and you can save your own selections for quick access to frequently used selections.



There is no data to show. Please select the following:

| Code | Name | Description | Owner | Shared |
|---|---|---|---|---|
| KEY | Key Indicators for Yearly Report | | Arsène von Wyss (Google) | Yes |

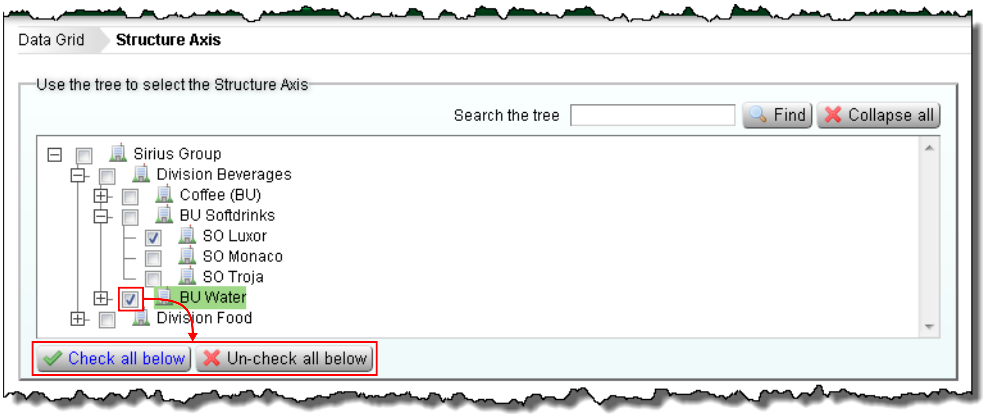
Showing 1 to 1 of 1 entries

When making a manual selection or completing a partial saved selection, you may select the dimensions of analysis in any order you choose by clicking the boxes or by using the actions on the right hand side.

## Structures Axis

Here you select the organizations, locations, regions etc. that you wish to analyze. These are called *Structures*.

The structures available to you are based upon your access permissions, i.e. if you only have permission for a part of the organization, you will only be able to analyze data for it. If you have access to higher structures in the tree, you have the ability to select entire branches. By selecting the highest level structure you may check or uncheck its descendants by clicking *Check all below* or *Un-check all below*, respectively:



You may select multiple structures by checking them. Once you have selected the structures you move onto the next selection step by clicking either: ⇐ Time Axis or ⇒ Indicator Axis.

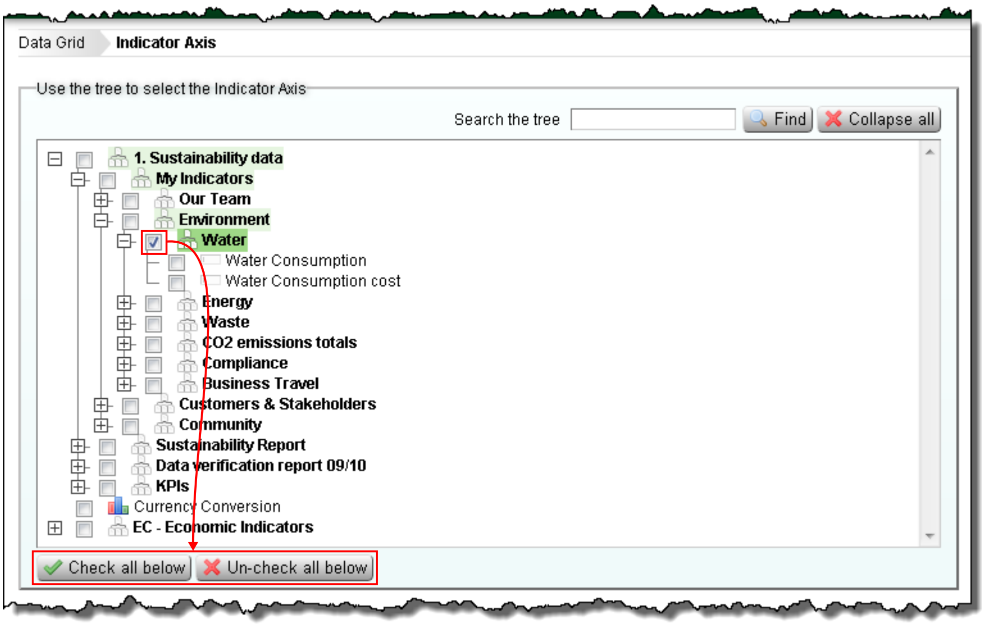
If you have entered all three axes you may load the data into the *Data Grid* by clicking the *Done* button.

**Tip:** If you have a large and/or expanded structure, these buttons may have moved out of sight and you will need to scroll down to find them.

## Indicators Axis

Here you select the indicators that you wish to analyze. Their configuration varies for each SERAM configuration and it may also change for different reporting periods.

The indicators available to you are based upon your access permissions. By selecting an indicator higher up in the tree you may check or uncheck its descendants by clicking *Check all below* or *Un-check all below*, respectively:



You may select multiple indicators by checking them.

**Note:** Some indicators may appear at multiple places in the tree. If you check one of these indicators, SERAM will expand all branches where this specific indicator is used. Unless you check multiple parents of the checked indicator as well, it will only appear once in the loaded grid. Checking or unchecking such an indicator, however, always affects all instances in the tree.

Once you have selected the indicators you move onto the next selection step by clicking either: ⇐ Structure Axis or ⇒ Time Axis.

If you have entered all three axes you may load the data into the *Data Grid* by clicking the *Done* button.

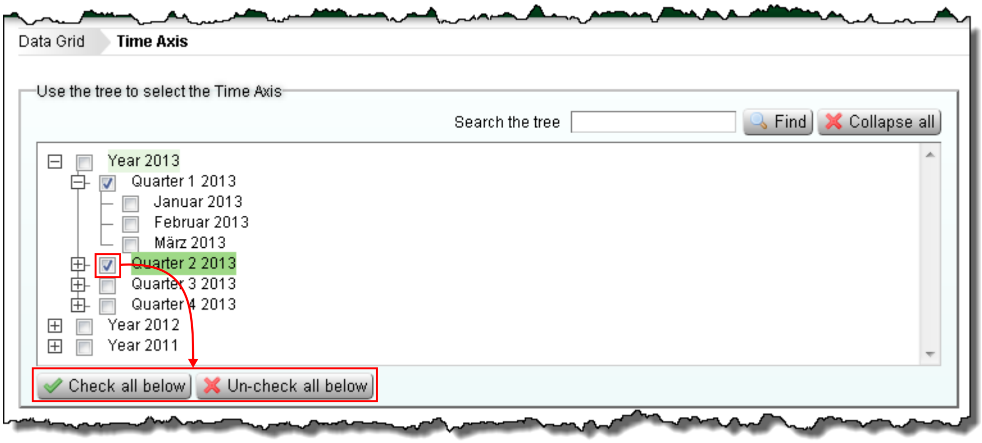
**Tip:** If you have a large and/or expanded indicator tree, these buttons may have moved out of sight and you will need to scroll down to find them.

## Time Axis

You may view data on a weekly, monthly, quarterly or yearly basis - or a mixture of all of them.

**Note:** The availability of the different time-frames depends on the reporting period setup, so that you may not have all choices listed above.

You may check or uncheck entire periods and their descendants may check or uncheck its descendants by clicking *Check all below* or *Un-check all below*, respectively:



Data Grid | **Time Axis**

Use the tree to select the Time Axis

Search the tree [ ] Find | Collapse all

- ⊟ □ Year 2013
  - ⊟ ☑ Quarter 1 2013
    - □ Januar 2013
    - □ Februar 2013
    - □ März 2013
  - ⊞ ☑ Quarter 2 2013
  - ⊞ □ Quarter 3 2013
  - ⊞ □ Quarter 4 2013
- ⊞ □ Year 2012
- ⊞ □ Year 2011

You may make multiple time selections by checking them. Once you have selected the time-frames you move onto the next selection step by clicking either: ⇐ Indicator Axis or ⇒ Structure Axis.

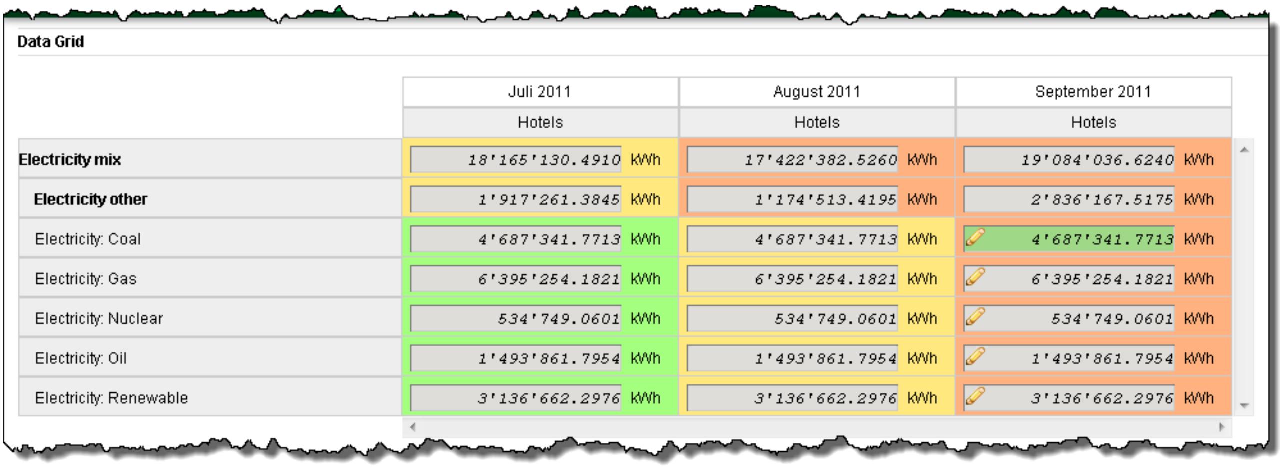
If you have entered all three axes you may load the data into the *Data Grid* by clicking the *Done* button.

**Tip:** If you have a large and/or expanded time tree, these buttons may have moved out of sight and you will need to scroll down to find them.

# Method Axis

# The Data Analysis Grid

SERAM uses a grid structure to display and analyze data across multiple axes. The system automatically displays the results in the structure it believes is most useful, however users can change the way the data is presented in whichever way they feel is most relevant.



**Data Grid**

| | Juli 2011 | August 2011 | September 2011 |
|---|---|---|---|
| | Hotels | Hotels | Hotels |
| **Electricity mix** | 18'165'130.4910 kWh | 17'422'382.5260 kWh | 19'084'036.6240 kWh |
| **Electricity other** | 1'917'261.3845 kWh | 1'174'513.4195 kWh | 2'836'167.5175 kWh |
| Electricity: Coal | 4'687'341.7713 kWh | 4'687'341.7713 kWh | ✎ 4'687'341.7713 kWh |
| Electricity: Gas | 6'395'254.1821 kWh | 6'395'254.1821 kWh | ✎ 6'395'254.1821 kWh |
| Electricity: Nuclear | 534'749.0601 kWh | 534'749.0601 kWh | ✎ 534'749.0601 kWh |
| Electricity: Oil | 1'493'861.7954 kWh | 1'493'861.7954 kWh | ✎ 1'493'861.7954 kWh |
| Electricity: Renewable | 3'136'662.2976 kWh | 3'136'662.2976 kWh | ✎ 3'136'662.2976 kWh |

Values are displayed in the corporate reporting unit, which can be different from the input unit by individual structures. For example, in the above screen-shot, the reporting unit for electricity from oil is kWh, but through country-specific conversion factors the data may actually have been entered using cubic meters as input unit.

The same color coding scheme is used in the *Data Grid* as when using the *Data Entry Form*.

Additionally, a pen ✎ symbol in the value is a visual indicator that this value can be edited.

### The Grid Axes

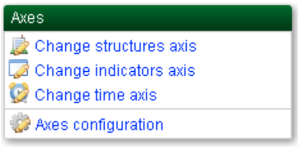
The three selected dimensions, i.e. structures, indicators and time periods, are used as axes for the grid. One axis is displayed vertically and two axes are displayed horizontally; the items of the lower horizontal axis is repeated for each item of the upper horizontal axis.

In the above screen-shot, the 7 selected indicators occupy the vertical axis, the 3 selected time periods occupy the upper horizontal axis, and the single selected structure occupies the lower horizontal axis, repeated for each time period.

Unless manually overridden, SERAM assumes that the most prominent selection shall be vertical. Therefore, it will assign the dimension with the most selections to the vertical axis and the dimension with the least selections to the lower horizontal axis. If you were to choose all time periods of a year but only few indicators, these two axes would be swapped automatically for better display.

## Changing Grid Axes

If you wish to change the setup or content of the grid, you may do so through the Axes box on the right hand side:



**Change structure axis**

Allows you to *reselect the structures* you wish to be displayed.

**Change indicator axis**

Allows you to *reselect the indicators* you wish to be displayed.

**Change time axis**

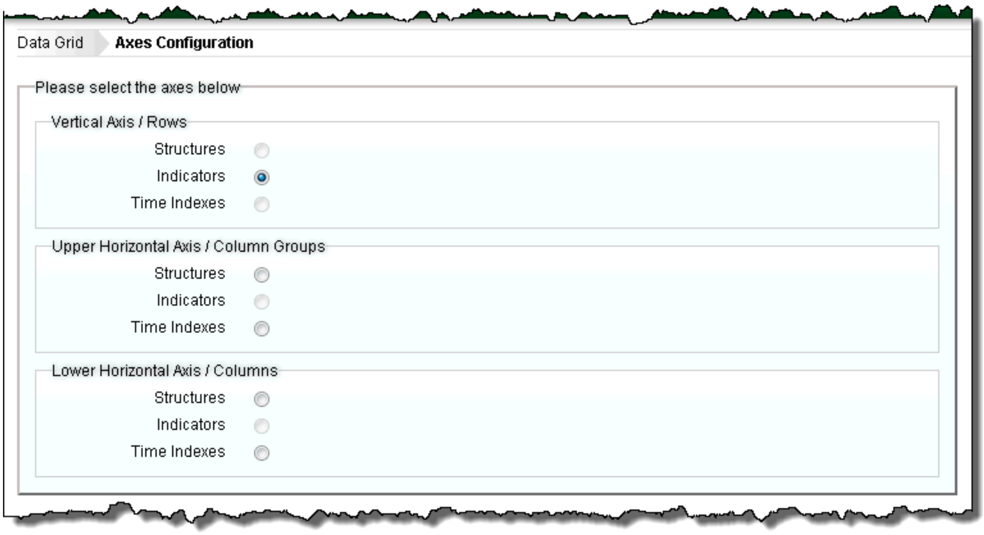
Allows you to *reselect the time periods* you wish to be displayed.

**Change method axis**

Allows you to *reselect the methods* you wish to be displayed.

**Axes configuration**

If you wish to change the way in which your data is displayed you can change the Axes configuration:



Data Grid **Axes Configuration**

Please select the axes below

Vertical Axis / Rows
- Structures ○
- Indicators ●
- Time Indexes ○

Upper Horizontal Axis / Column Groups
- Structures ○
- Indicators ○
- Time Indexes ○

Lower Horizontal Axis / Columns
- Structures ○
- Indicators ○
- Time Indexes ○

Whenever you want to make a new selection, click the *Clear selected axes* action on the right-hand side of this screen. This will make all radio-buttons available for selection.

You may select the axis representation of the *Structures*, *Indicators* and *Time Indexes* dimensions by clicking the corresponding radio buttons in the axis groups. Note that each dimension can only be selected once, so that the available dimensions are reduced after making a selection. In the above screen-shot, the *Indicators* have been assigned to the vertical axis, therefore the *Indicators* dimension cannot be used in the horizontal axes.

Once your selection is made, click ✓ Use selected axes. You will then be returned to the grid, which will now use the new axes setup. If the configuration you selected is not what you wanted, you can return to the *Axes configuration* and make different choices, or use ✓ Use auto selection for axes to return to the *automatic axes selection*.

**Note:** SERAM will keep your manual axes configuration for future use of the *Data Grid*. Therefore, if the arrangement of the axes seems odd after loading a new set of data, you may have to go to the Axes configuration and re-enable the automatic selection for axes.

## Details of Data Grid values

After selecting a single indicator value, the user may view further details of that indicator or value using the actions in the *Selected Indicator* category on the right-hand side of the screen.

### Indicator details

To view details about a selected value in the Data Grid, you can click the *View more details* action to get details about the indicator of this value. The exact information displayed depends on the type of the indicator.

### Drill-down into indicator value

The drill-down opens a new view with a list of directly contributing indicator values, including a chart visually displaying the breakdown.

**Note:** This is only available for values which are computed or consolidated in any way, and not for values entered as these cannot be broken down.

For instance, drilling into a calculation will return all the distinct values used in the calculation. Drilling into a structure-consolidated value (even cross-type) will return all values of the contributing structure level. Time-consolidated values will return the time indices used, e.g. a yearly value in a fiscal year will return quarterly values.

Any value in the drill-down list can be drilled down further down to the input value(s).

### View or edit value

By clicking the *View/Edit the value for this indicator* action a new view opens up showing a *Data Entry Form* view of the selected value. If the value is editable, you can also modify its value here.

**Tip:** Double-clicking a value in the grid invokes this action as well.

### View history of value changes

The changes made to values are recorded by SERAM. You can inspect this value history log by clicking the *View history for ...* action.

**Note:** This is only available for values entered, and not for computed or consolidated values, which often make up the majority of values displayed in the grid.

## Batch operations on Data Grid values

You can select value ranges in the *Data Grid* by clicking on the row and column headers. After making such a selection, batch operations can be performed on all selected values to which the operation is applicable, using the actions in the *Batch Operations* category on the right hand side.

Depending on the values and their status, the following actions can be executed:

✗ **Clear selection**

De-select the values. This does not modify any data.

**Set data to ...**

Change the status of the values to a new status. The status will only be changed according to your permissions and the general rules for status changes.

**Set or remove forecasted**

Change the forecasted flag for all values.

**Fill empty values with 0**

This option allows the user to replace any blank value with a zero. Many companies require that locations positively report zero values (i.e. don't leave blanks) and this function allows for quickly filling in zeroes where no value has to be supplied.

**Edit values**

Open a *Data Entry Form* view, allowing you to edit a series of values at once using the familiar interface.

**View values as chart**

Display the selection of data as chart.

**Note:** If a batch operation fails, the operation will not make any changes and the whole batch will be discarded. However, you are presented with the option to re-try the operation and skip errors. This option will perform the batch operation and silently ignore any errors, basically performing a best-effort run on the batch operation.

## Reference Environment

**Note:** This function is advised for advanced users who have undergone specific training.

By default, the data analysis grid uses the most commonly used defaults for the reference environment, e.g. that you wish to view data based on the standard company organization structure of the current time. SERAM however also allows for alternative structures to be analyzed, where they are in place in the company. For example, in addition to the typical company hierarchy structure, the tool may also collect data along geographical, legal or business type lines. Furthermore the tool allows for analysis of data in a historical mode i.e. reviewing performance of the company’s structure at a past time point e.g. before acquisitions or mergers. By clicking the *Change reference environment* button, the user may change the standard settings of the data grid to a more advanced and alternative view.

Examples which require changing the reference environment:

- Compute organizational indicator data by countries.
- Manage country-specific emission factors or currency-specific exchange rates.
- Deal with indicators defined and used only by specific structures.
- Look up data of structures which have been deleted in the mean time.

## Exporting and importing data

### Exporting the grid to Excel

SERAM uses Microsoft Excel as the default output tool for reporting. The tool will create an excel file for the user based upon the selection of the data grid. Therefore you have to finalize your grid selections before you can export the data to Excel. By using the *Export Data Grid to Excel* action on the right-hand side, SERAM will generate an excel-based replica of the Data Grid, and offer it as download in the browser.

### Using XML files

SERAM allows for data to be imported from external sources. As standard, you will be able to import the data stored in XML files previously generated by SERAM and populated with data by a 3rd-party application. It is imperative that the `.xml` files being imported are in the same format as exported by SERAM – the import does not support arbitrary `.xml` files. If you wish to import files of a different format or if you have any problems uploading `.xml` files, please contact your administrator.

As a first step, you need to generate the XML file to be populated. This is done by using the *Export as XML* action. You will be prompted for the file format to generate:

**Web XML (legacy SeramLite format)**

Choose this format only if you're generating files to be used in SERAM lite, or a compatible tool.

**Flat XML (for databases)**

Choose this format for generating files to be used with external databases. The XML contained uses a flat, table-like structure and contains a schema definition of the file. Also, the name of the file (also used for naming the data root element) has to be set prior to the export. The files generated are suitable for XML enabled databases such as Microsoft Access.

After populating the data in the file by the 3rd party tool, the file has to be imported into SERAM. This is done by using the *Import XML* action, which allows you to pick and upload the file.

The import process is subject to the same checks and rules as manually entered data. Therefore errors may occur even for correctly formatted files, such as when value is no longer *Open* or editable and may therefore not be modified. By default, whenever an error occurs, SERAM will abort the whole import and discard any changes, so that the data remains consistent (all or nothing). If you wish to perform a best-effort import, skipping over any erroneous values, you can tick the *Continue if error occurs* check-box. With this check-box ticked, the import will always succeed, but it may be ignoring some or even all values which failed to import.

After importing data, SERAM offers the choice to load the *Data Grid* with the data that has just been imported. This allows you to verify the data and also to change the status to *Entered*, which cannot be done through the data import itself.

Chapter

14

# Applications

SERAM allows interfacing with external systems. Each such external system is called an *Application*. External data from an application can be sent to SERAM. Another data workflow than the default UI-driven workflow is supported in this case.

The applications are first-class members similar to SERAM users. Applications are granted access with *Access Control Entries* in the same granular level as users, and actions performed are also logged in the standard *Audit Log*.

Any number of applications can be added to a SERAM tenant, each configured to access the indicators and values of all *Reporting Periods* of one *Reporting Period Type* starting on a specific calendar month (such as all Financial Year periods starting in January). Incoming dates are mapped to a reporting period (e.g. 2014) and period index (e.g. January) in SERAM. This association allows to unambiguously identify values by a calendar date even if you have multiple overlapping reporting periods. In addition, applications can have optional mappings of *structure names*, *indicator names* and *data status names*, and a *list of indicator or structure identifiers to be ignored* when the external application is for instance sending data to calculated indicators or time-consolidated values.

Applications access SERAM through a programmatic interface (API). This interface is accessible through HTTP(S) in a REST-ful manner with JSON data. An application key is associated to build the access URL, and the key does also authenticate the application. For a detailed description of the interface, see the *SetValues API* reference documentation.

### Some necessary preparatory steps

To use an external application with SERAM

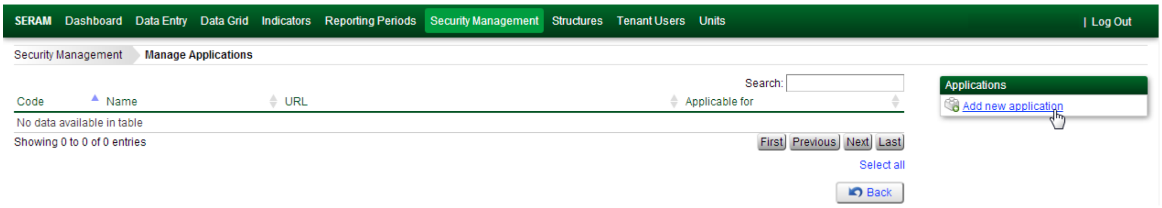
- *Add a new application*
- *SERAM automatically generates an API key*, which you can *reset* if necessary (with UI).
- *Assign access rights to the application*
- *Replace the data source of an indicator*

## Adding a new application

Before you can use the SetValues API and replace the default data source by an external application, you have to add a new application to the **Security Management** module.

To add an external application to SERAM

1. Switch to the **Security Management** module, and then, in the **Applications** category, click the **Manage applications** action.
2. On the **Manage Applications** page, point to the **Add new application** action (located in the **Applications** category).



| Code | Name | URL | Applicable for |
|---|---|---|---|
| No data available in table | | | |

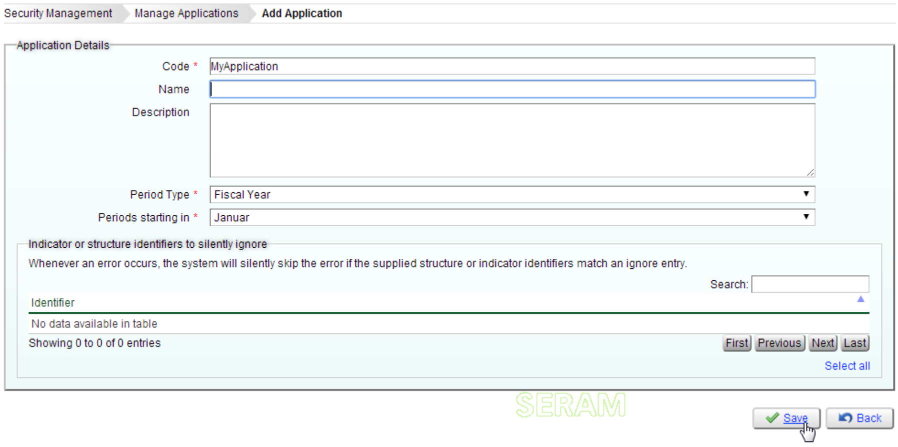
Showing 0 to 0 of 0 entries

3. Click **Add new application**.

The **Application Details** dialog appears.

4. In the **Code** box, type the label of the application (e.g. MyApplication).

   If your *Reporting Period Management* is using another type than the default Fiscal Year, select the appropriate item from the **Period Type** list. If your Periods start in another month than the default January, select the appropriate item from the **Periods starting in** list.



Security Management > Manage Applications > **Add Application**

Application Details

Code * MyApplication

Name

Description

Period Type * Fiscal Year

Periods starting in * Januar

Indicator or structure identifiers to silently ignore

Whenever an error occurs, the system will silently skip the error if the supplied structure or indicator identifiers match an ignore entry.

Search:

| Identifier |
|---|
| No data available in table |

Showing 0 to 0 of 0 entries

5. Click **Save**.
   SERAM adds your application entry to the list and generates an API key, which is used as access token to write data to SERAM. This API key is part of the data **URL** for the requests sent to SERAM by an external application and it basically authenticates and authorizes the request.

## Resetting an API key

You can reset the application key (URL) of an external application if necessary.

To reset the application key (URL)

1. Switch to the **Security Management** module.
2. Select the application from the list, and then, in the **Selected Application** category, click the **Reset the application key (URL)** action.

## Adding sufficient access rights

Applications are subject to normal permission checks. Therefore, each application represents a subject (virtual user) which also appears in the value history. To set data, the application needs the *Indicators:SetByApplication* permission on both the structure and the indicator, and the period must be open.

The most convenient and yet secure way to configure the access rights of your external application is to grant the following rights:

- *Access* to Authenticated Users on all structure types except Organization,
- *Indicator Data Editor* to Applications on Organization, and
- *Indicator Data Editor* to Authenticated Users on all indicator tags.

⚠ **Warning:** The recommended configuration allows your external application to read all indicator data. If you need restricted application access, remove the one Applications permission for the Organization structure type and assign more fine-grained access rights.

ℹ **Tip:** To verify the correct permissions for your external application, use the permission view on the **Security Management** > **Manage Applications** > **Access control entries** page.

## Managing the Ignore List

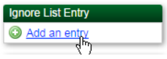
The *Ignore List* consists of indicator or structure identifiers. An external application may supply values for calculated indicators which are by their nature read-only in SERAM. The *Ignore List* tells SERAM to silently skip these instead of returning an error.

An external application is for instance sending data to calculated indicators or time-consolidated values, which are read-only in SERAM. This external data must not be imported in SERAM. An error occurs.

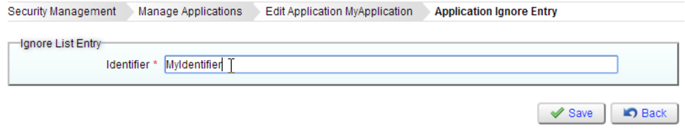
Use the *Ignore List* to suppress the error

1. Switch to the **Security Management** module, and then, in the **Applications** category, click **Manage applications**.
2. Select an application form the list, and then, in the **Selected Application** category, click the **Edit application** action.

   Whenever an external application related error occurs, the system will silently skip the error if the supplied structure or indicator identifiers match an ignore entry.

3. In the **Ignore List entry** category, click the **Add an entry** action.
   The **Ignore List Entry** dialog appears:

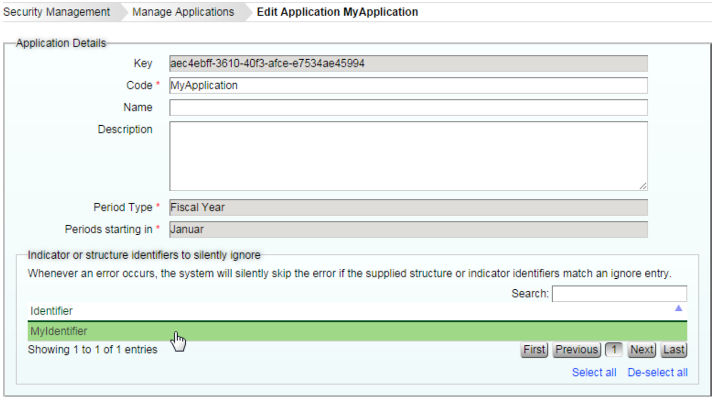
   

   Identifier * MyIdentifier

4. **Note:** Indicator or structure identifiers can contain the wildcards `?` (any single character) and `*` (any number of characters, including none).

   In the **Identifier** box, type the name of the structure or indicator to be ignored, and then click **Save**.

   The **Applications Details** dialog now shows your new entry in the **Indicator or structure identifiers to silently ignore** section:

   Key aec4ebff-3610-40f3-afce-e7534ae45994
   Code * MyApplication
   Name
   Description
   Period Type * Fiscal Year
   Periods starting in * Januar

   Whenever an error occurs, the system will silently skip the error if the supplied structure or indicator identifiers match an ignore entry.

   | Identifier |
   |---|
   | MyIdentifier |

   Showing 1 to 1 of 1 entries

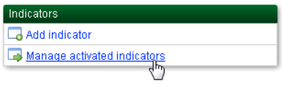
## Setting the source of an indicator

Usually, data is collected in SERAM with the help of the Data Entry module. The *SERAM UI* is the data source in this case. External applications can replace the default data source.

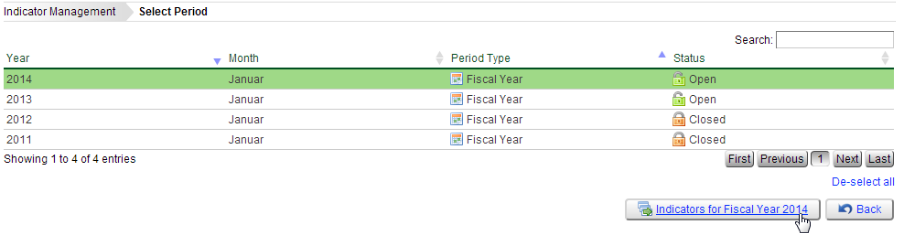
Basic prerequisite to replace the data source of an indicator is *an application managed by SERAM*.

To set an external application as data source for an indicator in SERAM

1. Switch to the **Indicators** module.

2. In the **Indicators** category, click the **Manage activated indicators** action.
3. Select a Period, for example the Year 2014.



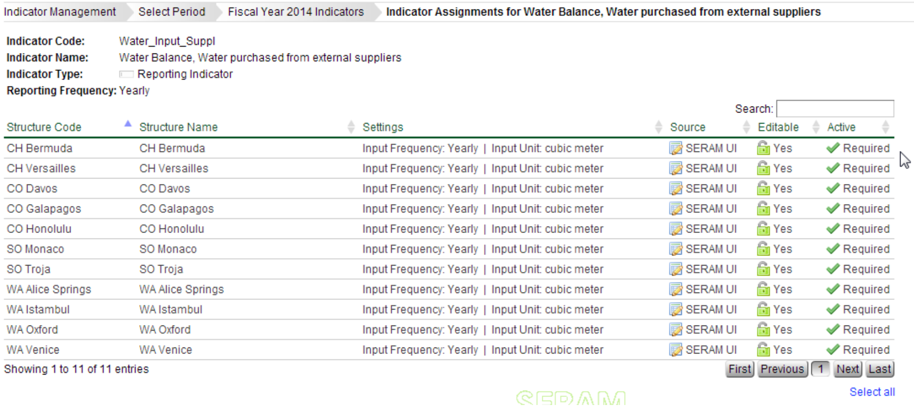
Indicator Management > **Select Period**

Search:

| Year | Month | Period Type | Status |
|---|---|---|---|
| 2014 | Januar | Fiscal Year | Open |
| 2013 | Januar | Fiscal Year | Open |
| 2012 | Januar | Fiscal Year | Closed |
| 2011 | Januar | Fiscal Year | Closed |

Showing 1 to 4 of 4 entries

4. Click, for example, **Indicators for Fiscal Year 2014** or rather for your selected Period Type and Year.
5. Select your indicator, and then, in the **Selected Indicator** category, click the **View indicator assignments** action.



Indicator Management > Select Period > Fiscal Year 2014 Indicators > **Indicator Assignments for Water Balance, Water purchased from external suppliers**

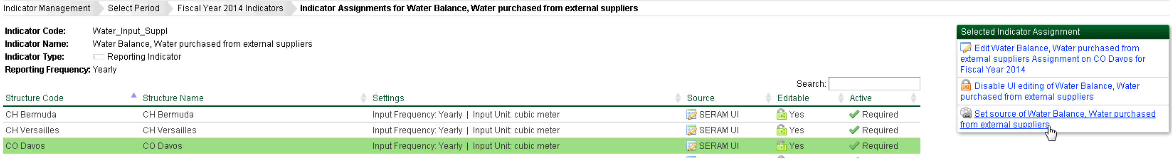
**Indicator Code:** Water_Input_Suppl
**Indicator Name:** Water Balance, Water purchased from external suppliers
**Indicator Type:** Reporting Indicator
**Reporting Frequency:** Yearly

Search:

| Structure Code | Structure Name | Settings | Source | Editable | Active |
|---|---|---|---|---|---|
| CH Bermuda | CH Bermuda | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| CH Versailles | CH Versailles | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| CO Davos | CO Davos | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| CO Galapagos | CO Galapagos | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| CO Honolulu | CO Honolulu | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| SO Monaco | SO Monaco | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| SO Troja | SO Troja | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| WA Alice Springs | WA Alice Springs | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| WA Istambul | WA Istambul | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| WA Oxford | WA Oxford | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| WA Venice | WA Venice | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |

Showing 1 to 11 of 11 entries

6. Select a structure entry from the list.



Indicator Management > Select Period > Fiscal Year 2014 Indicators > **Indicator Assignments for Water Balance, Water purchased from external suppliers**

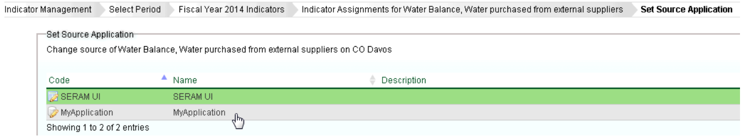
**Indicator Code:** Water_Input_Suppl
**Indicator Name:** Water Balance, Water purchased from external suppliers
**Indicator Type:** Reporting Indicator
**Reporting Frequency:** Yearly

Search:

| Structure Code | Structure Name | Settings | Source | Editable | Active |
|---|---|---|---|---|---|
| CH Bermuda | CH Bermuda | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| CH Versailles | CH Versailles | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| CO Davos | CO Davos | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |

Selected Indicator Assignment
- Edit Water Balance, Water purchased from external suppliers Assignment on CO Davos for Fiscal Year 2014
- Disable UI editing of Water Balance, Water purchased from external suppliers
- Set source of Water Balance, Water purchased from external suppliers

7. In the **Selected Indicator Assignment** category, click the **Set source** action.



Indicator Management > Select Period > Fiscal Year 2014 Indicators > Indicator Assignments for Water Balance, Water purchased from external suppliers > **Set Source Application**

Set Source Application

Change source of Water Balance, Water purchased from external suppliers on CO Davos

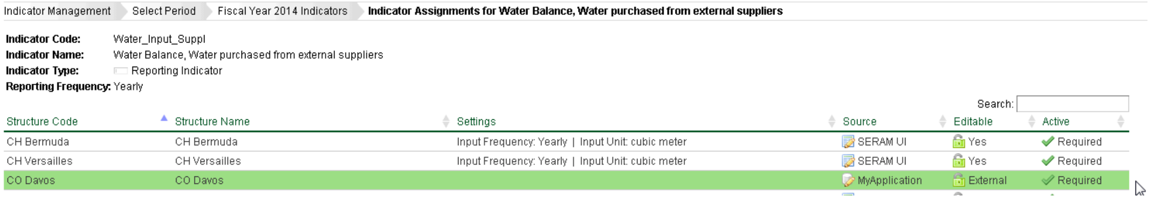
| Code | Name | Description |
|---|---|---|
| SERAM UI | SERAM UI | |
| MyApplication | MyApplication | |

Showing 1 to 2 of 2 entries

Now you can change the source of the indicator from **SERAM UI** to an external application.

8. Click **Save**.

In the list, you see the source now set to the external application and the value of the **Editable** column changed to **External**.



Indicator Management > Select Period > Fiscal Year 2014 Indicators > **Indicator Assignments for Water Balance, Water purchased from external suppliers**

**Indicator Code:** Water_Input_Suppl
**Indicator Name:** Water Balance, Water purchased from external suppliers
**Indicator Type:** Reporting Indicator
**Reporting Frequency:** Yearly

Search:

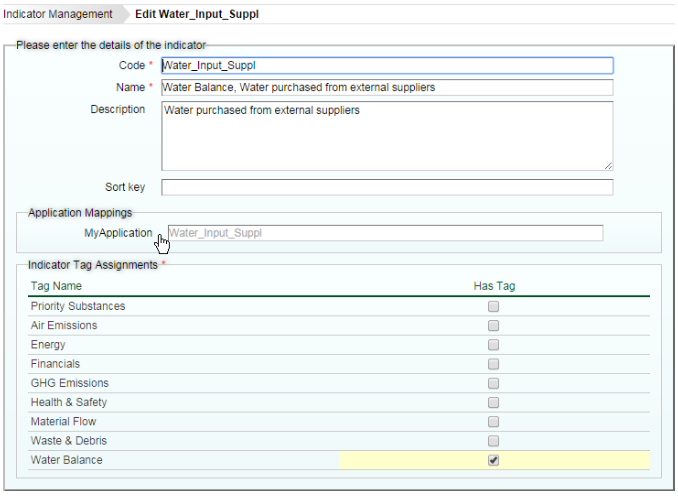
| Structure Code | Structure Name | Settings | Source | Editable | Active |
|---|---|---|---|---|---|
| CH Bermuda | CH Bermuda | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| CH Versailles | CH Versailles | Input Frequency: Yearly \| Input Unit: cubic meter | SERAM UI | Yes | ✓ Required |
| CO Davos | CO Davos | | MyApplication | External | ✓ Required |

## Using mapped indicator names

You can to manage (UI) and use mapped indicator names, so that the indicators in SERAM can be mapped to data received from an external application.

Because indicator names in external applications can have different names as in SERAM, you need a way to manage these different names. The **Indicators** module of SERAM provides a mapping.

1. Switch to the **Indicators** module.
2. In the list, search and select the indicator, whose application mappings you want to edit, and then, in the **Selected Indicator** category, click the **Edit**action.



Indicator Management › Edit Water_Input_Suppl

Please enter the details of the indicator

| | |
|---|---|
| Code * | Water_Input_Suppl |
| Name * | Water Balance, Water purchased from external suppliers |
| Description | Water purchased from external suppliers |
| Sort key | |

Application Mappings

| | |
|---|---|
| MyApplication | Water_Input_Suppl |

Indicator Tag Assignments *

| Tag Name | Has Tag |
|---|---|
| Priority Substances | ☐ |
| Air Emissions | ☐ |
| Energy | ☐ |
| Financials | ☐ |
| GHG Emissions | ☐ |
| Health & Safety | ☐ |
| Material Flow | ☐ |
| Waste & Debris | ☐ |
| Water Balance | ☑ |

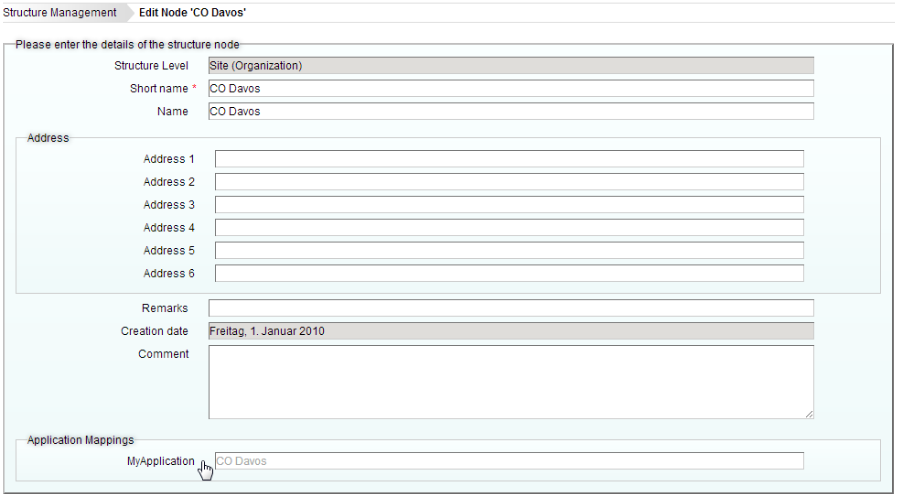
3. In the **Application Mappings** section, you find a list of the external applications configured in SERAM (e.g. MyApplication). In the box corresponding to your external application, (in this example the **MyApplication** box), type the name of the indicator as it appears in your external application.
4. Click **Save**.

## Using mapped organization structure names

You can manage (UI) and use mapped organization structure names, so that the structures in SERAM can be mapped to data received from an external application.

Because organizational nodes in external applications can have different names as in SERAM, you need a way to manage these different names. The **Structures** module of SERAM provides a mapping.

1. Switch to the **Structures** module.
2. Select the node, whose application mappings you want to edit.
3. In the **Selected Node** category, click the **Edit** action.



Structure Management › Edit Node 'CO Davos'

Please enter the details of the structure node

| | |
|---|---|
| Structure Level | Site (Organization) |
| Short name * | CO Davos |
| Name | CO Davos |

Address

| | |
|---|---|
| Address 1 | |
| Address 2 | |
| Address 3 | |
| Address 4 | |
| Address 5 | |
| Address 6 | |

| | |
|---|---|
| Remarks | |
| Creation date | Freitag, 1. Januar 2010 |
| Comment | |

Application Mappings

| | |
|---|---|
| MyApplication | CO Davos |

4. In the **Applications Mappings** section, you find a list of the external applications configured in SERAM (e.g. MyApplication). In the box corresponding to your external application, (in this example the **MyApplication** box), type the name of the structure node received from the external application to be mapped to the SERAM name.
5. Click **Save**.

## Using mapped data status names

You can manage (UI) and use mapped data status names, so that the data status in SERAM can be mapped to data received from an external application.

Because data status names in external applications can have different names as in SERAM, you need a way to manage these different names. The **Indicators** module of SERAM provides a mapping.

1. Switch to the **Indicators** module.
2. In the **Indicator Data States** category, click the **Manage Indicator Data States** action.
3. Select a **Status** entry.
4. In the **Indicator Status** category, click the **Edit** action.
5. In the **Application Mappings** section, you find a box for every external application configured in SERAM.
6. In the box corresponding to your external application, for example the **MyApplication** box, type your mapping.

   Each text line of a mapping represents a distinct mapping, so that multiple names of the external application can be mapped to one entity.
7. Click **Save**.

# Query API

Query indicator values and entities of SERAM.

### Overview

The Query API allows external applications to query data from SERAM. There are two different types of queries; the value queries which retrieve indicator values, and the entity queries which retrieve data about entities in the system, such as structures.

### Request

The request is sent via HTTP `GET` or `POST` to the `Query` URI including the application key as specified in the SERAM Applications management page. The query is transferred as query argument `q` in the case of `GET`, and as JSON object with a property `q` in the case of `POST` (content-type `application/json`, preferred encoding is UTF-8 but others are also acceptable if correctly declared in the header).

### Syntax Convention

The following syntax convention is used in this section for text used in the query:

| **Syntax** | **Description** |
|---|---|
| `plain text` | Plain text to be entered literally |
| *`<italic>`* | Placeholder for the specified fragment. These can, in some situations, contain recursions. |
| `( ... )*` | Used for specifying repetition patterns of any number of repetitions (including zero times). |
| `[ ... ]` | Optional component. |

The query is made up of four parts:

### Query Part 1: Environment Setup

The first part of the query can be used to modify the *Reference Environment* used for the query. When not specified, the defaults are applied in the same fashion as when using the *Data Grid*. The entire part is optional, when used the first instruction has to be prefixed with the keyword `with`, and each additional instruction is appended with an `and`.

| **Syntax** | **Description** |
|---|---|
| `structure date` *`<Date Ref>`* | Changes the date used for retrieving the structures. By default, this is "now". |

| Syntax | Description |
|---|---|
| `indicator period <Period Ref>` | The reporting period used for indicator entities. By default, this is the current year period. This setting has not effect on the value queries. |
| `indicator tags <Tags Ref>` | The indicator tags to take into account. Only indicators tagged with at least one of the tags will be available. By default, all indicator tags are available. |
| `root structure <Root Structure Ref>` | Defines the scope of available structures and indicators. This can be used to retrieve data with cross-type data flows. By default, the first structure type is selected. |
| `scenario <Scenario Ref>` | Use a specific structure scenario for the query (instead of the default scenario). |
| `as scenario` | Switches to scenario computation mode, where only data of the selected structures is computed. This setting has not effect on entity queries. |

### Query Part 2: Data Query

The query part defines what data (values or entities) shall be loaded and processed. This is the only mandatory part.

| Syntax | Description |
|---|---|
| `get indicator [<Indicators Ref>] values on <Structure Nodes Ref> in <Times Ref> [for <Methods Ref>]` | Queries *Indicator Values* on specified *Structures* with explicit selection of *Times* (e.g. combinations of a *Reporting Period* and a *Time Index*) and optionally for specific *Method(s)* or *Method Part(s)*.<br>When no *Indicator* reference is specified, then all indicators which are in scope (e.g. are associated with a tag configured in the environment) will be retrieved.<br>When no *Method* reference is specified, then the total of the default *Method* will be retrieved. |
| `get selection <Selection Ref> values [with <Axis Ref> (and <Axis Ref>)*]` | Queries *Indicator Values* based on the specified *Saved Selection* (as saved in the *Data Grid / Data Analysis*.<br>Optionally, structures, indicators, times and methods to be loaded can be specified (e.g. when the selection does not include these parts). |
| `get <Entity Type>` | Queries the specified entity such as structures. |

### Query Part 3: Filter

Each query may be filtered to only include the results which fulfill a condition.

| Syntax | Description |
|---|---|
| `where <Predicate>` | Defines a filter predicate for the query results. The predicate uses the same syntax as the other *Formulas*, but the available properties depend on the items in the query result. |

### Query Part 4: Output Transformation

The result of the query can be rendered to different output formats to match the requirements of the external application.

When no transformation is specified, the result is rendered as a JSON object with a `success` and `result` property:

```
{
 "success":true,
 "result":[{
  "id":"4fad0962-a656-f74a-8718-f9ac4caa938d",
  "code":"SA",
  "name":"Site A",
  "parent":"Division X",
  "type":"Organization",
  "level":"Site",
  "linksto":[],
  "linkedfrom":[]
 }, {
  "id":"dc3e4f8d-041b-214c-9b86-0facbcb6a455",
  "code":"SB",
  "name":"Site B",
  "parent":"Division X",
  "type":"Organization",
  "level":"Site",
  "linksto":[],
  "linkedfrom":[]
 }]
}
```

| **Syntax** | **Description** |
|---|---|
| `as json` *`<JSON>`* | Render the result to as an array using the specified JSON template. The template is specified in-place as normal JSON, but instead of property values a *Formula* is expected which retrieves the value. Most of the time the formula will just be a property access on the result object, but it can also contain complex computations if required.<br>Sample for a custom JSON rendering of structures, which only renders the two JSON properties "key" and "name":<br>`as json { "key": code, "name": name }`<br>This would generate an output like this (indentation has been added for readability and may not be included in the output):<br>`[`<br>`{ "key": "SA", "Name": "Site A" },`<br>`{ "key": "SB", "Name": "Site B" }`<br>`]` |
| `as json` *`<Variable>`* `:=` *`<JSON>`* `in` *`<JSON>`* | Render the result as array to a variable, and then render the JSON output document which should reference the specified variable, but it cannot access the properties of a specific item. This JSON mode is an extended version of the mode above, allowing to generate complex JSON documents instead of only an array. |

<table>
<tr><th>Syntax</th><th>Description</th></tr>
<tr><td></td><td>Sample with the same data as the simple JSON data:<br><code>as json $j := { "key": code, "name": name } in { "structures": $j }</code><br>This would generate an output like this:<br><code>{ "structures": [</code><br><code>{ "key": "SA", "Name": "Site A" },</code><br><code>{ "key": "SB", "Name": "Site B" }</code><br><code>]}</code></td></tr>
<tr><td><code>as csv <i>&lt;Column&gt;</i>(, <i>&lt;Column&gt;</i>)* [with separator <i>&lt;Separator&gt;</i>]</code></td><td>Render the result as Comma-Separated-Values (CSV) text, optionally using a custom separator. Each column is a pair of a name and an expression formula, just like the JSON properties.<br>Sample with the same data as the simple JSON data:<br><code>as csv "key": code, "name": name with separator ","</code><br>This would generate an output like this:<br><code>"key","name"</code><br><code>"SA","Site A"</code><br><code>"SB","Site B"</code></td></tr>
<tr><td><code>as xls <i>&lt;Column&gt;</i>(, <i>&lt;Column&gt;</i>)*</code></td><td>Render the result as Excel binary file. The column definition is identical to the CSV.<br>Sample with the same data as the simple JSON data:<br><code>as xls "key": code, "name": name</code><br>This would generate an output like this when viewed in Excel:
<table>
<tr><th>key</th><th>name</th></tr>
<tr><td>SA</td><td>Site A</td></tr>
<tr><td>SB</td><td>Site B</td></tr>
</table></td></tr>
<tr><td><code>as xml <i>&lt;Variable&gt;</i> := (<i>&lt;XElement&gt;</i>)* in &lt;XElement&gt;</code></td><td>Render the result to a list of XML elements using the template and store into the specified variable, then render the XML document element which should reference the variable. This is conceptually identical with the complex JSON output.<br>Most XML features are supported, including nested elements, attributes and namespaces (by using xmlns attributes and prefixes).<br>Sample with the same data as the simple JSON data:<br><code>as xml $x := &lt;structure key=code&gt;name&lt;/structure&gt; in &lt;doc&gt;$x&lt;/doc&gt;</code></td></tr>
</table>

| Syntax | Description |
|---|---|
| | This would generate an output like this (indentation has been added for readability and may not be included in the output):<br>`<?xml version="1.0"?>`<br>`<doc>`<br>` <struct key="SA">Site A</struct>`<br>` <struct key="SB">Site B</struct>`<br>`</doc>` |

### Entity References

In the *Environment Setup* and the *Query* parts, references to some entities of the systems can be used:

**`<Date Ref>`: Date reference**

| Syntax | Description |
|---|---|
| *`<Date>`* | Dates are specified in ISO 8601 format, e.g. `YYYY-MM-DD`. When the day part is omitted, the 1st of the month is assumed. |

**`<Period Ref>`: *Reporting Period* reference**

| Syntax | Description |
|---|---|
| `[`*`<Quoted>`*`]` *`<Integer>`* | An integer number specifying the year only. The *Reporting Period Type* (in double quotes) is optional but can be used to make the match distinct. The system will search for a matching *Reporting Period*. The resolution will only be successful if exactly one match is found, it will fail if multiple periods exist for the same year. |
| `[`*`<Quoted>`*`]` *`<Date>`* | Search for a *Reporting Period* by its start date and optionally the *Reporting Period Type* (in double quotes). This can be different from the year for periods which do not start in January. The resolution is only successful if exactly one match is found. |
| *`<Guid>`* | Specify the *Reporting Period* by its GUID, written in curly brace format. |

**`<Scenario Ref>`: *Scenario* reference**

| Syntax | Description |
|---|---|
| *`<Quoted>`* | Match the *Scenario* by its code, written in double quotes. |
| *`<Guid>`* | Specify the *Scenario* by its GUID, written in curly brace format. |

**`<Tag Ref>`: *Indicator Tag* reference**

| Syntax | Description |
|---|---|
| *`<Quoted>`* | Match the *Indicator Tag* by its code, written in double quotes. |
| *`<Guid>`* | Specify the *Indicator Tag* by its GUID, written in curly brace format. |

### `<Tags Ref>`: Set of *Indicator Tag* references

| Syntax | Description |
|---|---|
| *`<Tag Ref>`* `(,` *`<Tag Ref>`*`)*` | One or more `<Tag Ref>` separated by comma. |

### `<Indicators Ref>`: Set of *Indicator* reference

**Note:** To create the union of multiple *Indicator* sets, write them as list separated with commas.

| Syntax | Description |
|---|---|
| *`<Quoted>`* | Match the *Indicator* by its mapped code, written in double quotes. |
| `tagged` *`<Tag Ref>`* | Matches all indicators tagged with the specified tag. |
| *`<Guid>`* | Specify the *Indicator* by its GUID, written in curly brace format. |

### `<Methods Ref>`: Set of *Method* or *Method Part* reference

**Note:** To create the union of multiple *Method* or *Method Part* sets, write them as list separated with commas.

| Syntax | Description |
|---|---|
| *`<Quoted>`* | Match the *Method* (total) by its code, written in double quotes. |
| *`<Quoted> <Quoted>`* | Match the *Method* and one of its *Method Parts* by their codes, written in double quotes. |
| *`<Guid>`* | Specify the *Method* or *Method Part* by its GUID, written in curly brace format. |

### `<Frequency Ref>`: *Frequency* reference

| Syntax | Description |
|---|---|
| *`<Identifier>`* | Match the *Frequency* by its code, written as kexword or in double quotes. |
| *`<Guid>`* | Specify the *Frequency* by its GUID, written in curly brace format. |

### `<Structure Type Ref>`: *Structure Type* reference

| Syntax | Description |
|---|---|
| `type` *`<Quoted>`* | Match the *Structure Type* by its code, written in double quotes. |
| `type` *`<Guid>`* | Specify the *Structure Type* by its GUID, written in curly brace format. |

### `<Structure Level Ref>`: *Structure Level* reference

| Syntax | Description |
|---|---|
| *`<Quoted>`* | Match the *Structure Level* by its code, written in double quotes. |
| *`<Guid>`* | Specify the *Structure Level* by its GUID, written in curly brace format. |

### <Structure Node Ref>: *Structure Node* reference

**Note:** Only structures which are in scope, e.g. equal or below the *Root Structure*, will be searched and matched for this reference.

| Syntax | Description |
|---|---|
| *<Quoted>* | Match the *Structure Node* by its mapped code, written in double quotes. |
| parent of *<Structure Node Ref>* | Get the parent of another *Structure Node*. |
| *<Structure Level Ref>* ancestor of *<Structure Node Ref>* | Get the ancestor of the specified *Structure Level* of another *Structure Node*. |
| *<Guid>* | Specify the *Structure Node* by its GUID, written in curly brace format. |

### <Structure Nodes Ref>: Set of *Structure Node* references

**Note:** To create the union of multiple *Structure Node* sets, write them as list separated with commas.

| Syntax | Description |
|---|---|
| *<Structure Node Ref>* | A one-item set with the specified *Structure Node*. |
| all | All *Structure Nodes* which are in scope. |
| all *<Structure Level Ref>* | All *Structure Nodes* of the specified *Structure Level* which are in scope. |
| all *<Structure Level Ref>* descendants | All *Structure Nodes* which are below the specified *Structure Node* (and also in scope). |
| all *<Structure Level Ref>* descendants of *<Structure Node Ref>* | All *Structure Nodes* of the specified *Structure Level* which are below the specified *Structure Node* (and also in scope). |

### <Root Structure Ref>: *Indicator Root Structure* reference

| Syntax | Description |
|---|---|
| *<Quoted>* of *<Structure Type Ref>* | Search a structure with the given name in the specified type. This is the preferred method for defying an indicator root structure. |
| *<Structure Node Ref>* | Find a matching structure in the default *Structure Type*, which is usually the organizational structure. |
| *<Structure Type Ref>* | Specify a *Structure Type* as root. |

### <Times Ref>: Set of *Reporting Period Index* references

**Note:** To create the union of multiple *Reporting Period Index* sets, write them as list separated with commas.

| Syntax | Description |
|---|---|
| *<Period Ref>* | All time indexes of the specified *Reporting Period*. |
| *<Frequency Ref>* of *<Period Ref>* | All time indexes of the specified *Frequency* in the specified *Reporting Period*. |
| *<Integer> <Period Ref>* | The numeric index in the specified *Reporting Period*. For a list of the numeric indexes, see *Reporting Period Types*. |

| **Syntax** | **Description** |
|---|---|
| `<Identifier> <Period Ref>` | The time index by its code in the specified *Reporting Period* written without quotes, such as `Year` or `Q1` . |
| `<Frequency Ref> on <Date Ref>` | The time index which matches specified *Frequency* on the given date. |

**`<Axis Ref>`: *Axis* reference**

| **Syntax** | **Description** |
|---|---|
| `structures <Structure Ref>` | Structure references. |
| `indicators <Indicator Ref>` | Indicator references. |
| `times <Time Ref>` | Time references. |
| `methods <Method Ref>` | Method references. |

### Available Value Properties

The following properties can be used in output definitions:

| **`Identifier`** | **Type** | **Default** |
|---|---|---|
| `perioddate` | String (ISO date format) | Visible |
| `year` | Integer | Hidden |
| `frequency` | String | Visible |
| `identity` | LongValueId | Hidden |
| `index` | String | Hidden |
| `indicator` | String (mapped code) | Visible |
| `indicatorInfo` | Indicator | Hidden |
| `indicatorType` | String | Visible |
| `tagNames` | String | Hidden |
| `structure` | String (mapped code) | Visible |
| `structureInfo` | Structure | Hidden |
| `value` | Number, Boolean or String | Visible |
| `booleanvalue` | Boolean | Hidden |
| `textvalue` | String | Hidden |
| `numericvalue` | Number | Hidden |
| `unit` | String | Visible |
| `comment` | String | Visible |
| `lastmodified` | String (ISO date format) | Visible |
| `lastmodifiedBy` | String | Hidden |
| `createdBy` | String | Hidden |
| `status` | String | Visible |
| `quality` | Number (Integer 0..100) | Visible |
| `forecasted` | Boolean | Visible |

| **Identifier** | **Type** | **Default** |
|---|---|---|
| `completeness` | Number (0..1) | Visible |
| `published` | Boolean | Visible |
| `editable` | Boolean | Hidden |
| `valueid` | String | Hidden |

## SetValues API

Store values in SERAM without workflow restrictions.

### Overview

The SetValues API allows external applications to directly write data to *Persistent Indicators*. In contrast to the manually executed XML file imports, data imported through the API does not have to follow the *Indicator Status workflows*. Also, the subject recorded in the Value History and Audit Log will be the external software application instead of a SERAM user.

### JSON Request

The request is sent via HTTP `POST` to the `SetValues` URI including the application key as specified in the SERAM Applications management page. The payload is JSON data (content-type `application/json`, preferred encoding is UTF-8 but others are also acceptable if correctly declared in the header).

The following example is a complete request including HTTP headers for a fictional application with the key `12345678-abcd-1234-abcd-12345678abcd`:

```
POST /REST/SetValues/12345678-abcd-1234-abcd-12345678abcd HTTP/1.1
Content-type: application/json; charset=utf-8
Content-Length: 1606

{
 "values": [
  {
   "structure": "Site-A",
   "indicator": "Electricity",
   "perioddate": "2023-08",
   "status": "Verified",
   "value": 8452
  },
  {
   "structure": "Site-B",
   "indicator": "Fatalities",
   "perioddate": "2014-08",
   "status": "Draft",
   "value": 1,
   "comment": "Staff member died from a hearth attack.",
   "lastmodified": "2023-08-06T04:30:00Z",
   "forecasted": true,
   "dataquality": 100,
   "attachments": [
    {
     "filename": "IncidentReportAug2023.pdf",
     "mimetype": "application/pdf",
     "content": "(BASE64-Encoded)"
    }
   ]
  }
 ]
}
```

**Tip:** The formatting and whitespace in the example above is insignificant as per JSON format specifications. Also, the order of the object properties is not important.

| **Name** | **Data Type** | **Description** |
|---|---|---|
| `structure` | String | The name of the structure. If present on the structure, the mapped name is used, otherwise the code of the structure is used. |
| `indicator` | String | The name of the indicator. If present on the indicator, the mapped name is used, otherwise the code of the indicator is used. |
| `perioddate` | String with ISO 8601 date (short format without day may be used for reporting period types with monthly or coarser resolution) | The date for association to the correct value in SERAM. The value index is determined by finding the matching reporting period containing the given date by using the period type and period start month specified in the application definition to identify possible reporting periods.<br>For the selected indicator in the matched reporting period, the reporting frequency is used to find the exact value which shall be used to store the given data.<br>**Important:** While the `perioddate` only has to fall into the date range of the value, the value sent to SERAM will replace any existing reporting value. The input frequency cannot be different from the reporting frequency because the data status, comment, forecasted flag and data quality information is not handled at the input frequency level. |
| `status` | String | The data status to set the value to. If present, the mapped names are used, otherwise the code of the data status is used. |
| `value` | Number or `null` | The value to store. When missing or `null`, an empty value will be stored. |
| `comment` | String | The comment or text to store in the value. If not present or `null`, an empty comment will be stored. |
| `lastmodified` | String with ISO 8601 date and time | The date to record in the value history. If empty or missing, the current date and time is recorded. The modification date is not used to detect changes compared to existing data in SERAM. |

| **Name** | **Data Type** | **Description** |
|---|---|---|
| `forecasted` | Boolean | *Optional.* When `true`, the value is marked as forecasted. This only applies when forecasting is enabled for the tenant. |
| `dataquality` | Number | *Optional.* The data quality to record (20=estimated, 60=calculated, 100=measured). This only applies when data quality is enabled for the tenant and indicator. |
| `attachments` | Array of attachment objects:<br>`{ "filename":` *`string`*`,`<br>`"mimetype":` *`string`*`,`<br>`"content":` *`base64-encoded-string`* `}` | *Optional.* File attachments of the value. This only applies when attachments are enabled for the tenant and indicator. |

### JSON Success Response

If all values were handled or ignored by the application ignore list, the changes are saved and the response will state a success (HTTP status 200).

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 73

{
 "success": true,
 "changedvaluecount": 1,
 "errors": []
}
```

| **Name** | **Data Type** | **Description** |
|---|---|---|
| `success` | Boolean | Specifies that the request was successful (`true`). |
| `changedvaluecount` | Number | The number of values which were changed and saved. This may be less than the values sent to SERAM, since only changed values are saved and some erroneous values may also be silently ignored if specified in the applications ignore list. |
| `errors` | Array | This array will always be empty for successful requests. |

### JSON Failure Response

The request succeeds or fails as a complete unit. Therefore, even if some values were acceptable, no modification to any data will occur if the response states a failure. The following example is a complete response including HTTP headers (note that the HTTP status is 422):

```
HTTP/1.1 422 Unprocessable Entity
Content-Type: application/json; charset=utf-8
Content-Length: 265

{
 "success": false,
 "changedvaluecount": 0,
 "errors": [
  {
   "structure": "Site-A",
   "indicator": "Electricity",
   "perioddate": "2014-08-01T00:00:00.000Z",
   "message": "The reporting period for the value is closed"
  }
 ]
}
```

| Name | Data Type | Description |
|---|---|---|
| `success` | Boolean | Specifies that the request was not successful (`false`). |
| `changedvaluecount` | Number | Always 0 on failed requests. |
| `errors` | Array | This array contains all errors detected by SERAM during the processing of the request. Each error contains a message and (if applicable) the data required to identify the erroneous value. |

## CreateStructure API

Create a new structure node.

### Overview

The CreateStructure API allows external applications to create *Structure Nodes*, and optionally configure *Indicator to Structure Assignments* of the newly created *Structure Node*.

**Note:** The API needs to have appropriate permissions, e.g. `Structures:ManageNode` or `Structures:CreateChildNode` on the parent, or `Structures:ManageNode` on the tenant or *Structure Type* level.

### JSON Request

The request is sent via HTTP `POST` to the `CreateStructre` URI including the application key as specified in the SERAM Applications management page. The payload is JSON data (content-type `application/json`, preferred encoding is UTF-8 but others are also acceptable if correctly declared in the header).

The following example is a complete request including HTTP headers for a fictional application with the key `12345678-abcd-1234-abcd-12345678abcd`:

```
POST /REST/CreateStructure/12345678-abcd-1234-abcd-12345678abcd HTTP/1.1
Content-type: application/json; charset=utf-8
Content-Length: 168

{
    "created": "2020-01-01",
    "code": "API Node",
    "metadata": [
        {
            "name": "API Node Name"
        }
    ],
    "level": "Site",
    "parent": "API Parent Node",
    "changeSource": true,
    "activateAll": true
}
```

**Tip:** The formatting and whitespace in the example above is insignificant as per JSON format specifications. Also, the order of the object properties is not important.

| **Name** | **Data Type** | **Description** |
|---|---|---|
| `code` | String | The code of the new *Structure Node*. |
| `created` | String with ISO 8601 date | The creation date for the new *Structure Node*. |
| `metadata` | JSON array | *Optional.* The metadata items to apply on the new *Structure Node*. The example shows how to set the name; any other metadata property can be added by adding another object to the JSON array. |
| `level` | String | The code of the *Structure Level* of the new *Structure Node*. |
| `type` | String | *Optional.* The *Structure Type* where the new *Structure Node* is to be added (only needed when the *Structure Level* codes are ambiguous across *Structure Types*). |
| `parent` | String | *Optional.* The code of the parent *Structure Node*. |
| `changeSource` | Boolean | If `true` or not specified, the source of all indicator assignments will be changed to use the current application. This makes sure that the application can use the *SetValues API* to write indicator data to the node. |
| `activateAll` | Boolean | If `true`, all *Indicator to Structure Assignments* are activated. Otherwise, only mandatory indicators will be active. |

### JSON Success Response

Response if the *Structure Node* was successfully created (HTTP status 200).

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 275

{
 "success": true,
 "structure": {
  "id": "abcd1234-1234-abcd-1234-abcd12345678",
  "code": "API Node",
  "name": "API Node Name",
  "parent": "API Node Parent",
  "type": "Organization",
  "level": "Site",
  "metadata": [
   {
    "name": "API Node Name"
   }
  ],
  "linksto": [],
  "linkedfrom": []
 },
 "updatedAssignments": 190
}
```

| Name | Data Type | Description |
|---|---|---|
| `success` | Boolean | Specifies that the request was successful (`true`). |
| `structure` | JSON object | The JSON data of the newly created *Structure Node*. |
| `updatedAssignments` | Number | The count of the updates indicator structure assignments. |

### JSON Failure Response

The following example is a complete failed response due to a structure code conflict, including HTTP headers (note that the HTTP status is not 200; for conflicting structure code, it is 409):

```
HTTP/1.1 409 Conflict
Content-Type: application/json; charset=utf-8
Content-Length: 135

{
 "success": false,
 "error": "This operation causes a structure name collision (same code,
 same type, and overlapping date range)."
}
```

| Name | Data Type | Description |
|---|---|---|
| `success` | Boolean | Specifies that the request was not successful (`false`). |
| `error` | String | The error explaining the reason of the failure. |

## RenderReport API

Render a *Report* to a `docx` or `pdf` document.

### Overview

The RenderReport API allows external applications to render existing *Reports* to a file which can then be downloaded anonymously.

**Note:** The API needs to have appropriate permissions, e.g. `Reports:RunReport` and `Reports:ShareReport` on the *Report* to be rendered.

### JSON Request

The request is sent via HTTP `POST` to the `RenderReport` URI including the application key as specified in the SERAM Applications management page. The payload is JSON data (content-type `application/json`, preferred encoding is UTF-8 but others are also acceptable if correctly declared in the header).

The following example is a complete request including HTTP headers for a fictional application with the key `12345678-abcd-1234-abcd-12345678abcd`:

```
POST /REST/RenderReport/12345678-abcd-1234-abcd-12345678abcd HTTP/1.1
Content-type: application/json; charset=utf-8
Content-Length: 118
Accept-Language: de

{
 "report": "Company KPI Report",
 "config": {
   "time": [{ "period":"2021", "index":0 }],
   "company": ["Company Node"]
  }
 }
```

**Tip:** The formatting and whitespace in the example above is insignificant as per JSON format specifications. Also, the order of the object properties is not important.

**Note:** The language to use is specified in the `Accept-Language` header. If you cannot control that header, you can also append a `?lang=XX` query string to the URL to force a specific language.

| Name | Data Type | Description |
|---|---|---|
| `report` | String | The code of the *Report* to be rendered. |
| `config` | JSON object | The report configuration (variable values) to be sent to the report. The content and format depends on the variables to be passed in. |
| `comment` | String | *Optional.* The comment to be passed to the report renderer. |
| `pdf` | Boolean | If `true` or not specified, the created document will be a PDF document. If `false`, a Word / OpenOffice XML (`.docx`) document will be created. |

### JSON Success Response

Response if the *Report* was successfully rendered.

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 156

{
 "success": true,
 "title": "Company Name KPI Report",
 "mime-type": "application/pdf",
 "path": "/Reports/ReportDownload/12345678-5745-4e11-b82e-81e584d3d846"
}
```

| Name | Data Type | Description |
|---|---|---|
| `success` | Boolean | Specifies that the request was successful (`true`). |
| `title` | String | The generated report title, this can be used for display purposes. |
| `mime-type` | String | The MIME type of the file. |
| `path` | String | The path for the download (on the same host and port as the API call was made). |

### JSON Failure Response

The following example is a complete failed response:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 98
```

```
{
 "success": false,
 "error": "Report Company KPI Report not found."
}
```

| Name | Data Type | Description |
|---|---|---|
| `success` | Boolean | Specifies that the request was not successful (`false`). |
| `error` | String | The error explaining the reason of the failure. |

## CheckSyntax API

Check the syntax of different text-based inputs.

### Overview

The syntax of the following text-based inputs can be checked by this endpoint:

| Kind | Description |
|---|---|
| `query` | A query as used for the *Query API*, or when specifying queries in report definitions.<br>**Note:** The query is not semantically checked, e.g. the validity of identifiers, IDs etc. is not part of the check. |
| `expression` | An expression formula, usually returning a numeric value.<br>`"Employee Hours" * 25%` |
| `predicate` | A predicate formula, usually some comparison.<br>`("Employee Hours" / "Employee Count") > 40` |
| `formula` | Either an `expression` or a `predicate`. |
| `xml` | Plain XML data. While not specific to SERAM, XML is used in different parts of report definitions. |

### JSON Request

The request is sent via HTTP `GET` or `POST` to the `CheckSyntax` URI including the application key as specified in the SERAM Applications management page. For `GET`, the payload must be passed via query string; for `POST`, the payload is JSON data (content-type `application/json`, preferred encoding is UTF-8 but others are also acceptable if correctly declared in the header).

The following example is a complete request including HTTP headers for a fictional application with the key `12345678-abcd-1234-abcd-12345678abcd`:

```
POST /REST/CheckSyntax/12345678-abcd-1234-abcd-12345678abcd HTTP/1.1
Content-type: application/json; charset=utf-8
Content-Length: 606

{
    "kind": "query",
    "data": "get indicator values on $structure in YR $period\nwhere
 indicatorType!='Group Indicator'"
}
```

**Tip:** The formatting and whitespace in the example above is insignificant as per JSON format specifications. Also, the order of the object properties is not important.

| Name | Data Type | Description |
|---|---|---|
| `kind` | String | The data kind, see table above. |
| `data` | String | The text data to be checked. |

### JSON Success Response

If the syntax is correct, the response indicates this in the `success` property and provides also a pretty-printed version of the formula as well as a list of unbound variable names.

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 151

{
 "success": true,
 "pretty": "get indicator values on $structure in YR $period where
 indicatorType != 'Group Indicator'",
 "variables": [
  "structure",
  "period"
 ]
}
```

| Name | Data Type | Description |
|---|---|---|
| `success` | Boolean | Specifies that the request was successful (`true`). |
| `pretty` | String | Pretty-printed version of the data supplied. |
| `variables` | Array | The array contains all unbound variable names, if any. |

### JSON Failure Response

When the syntax is invalid, the response returned (wuth a HTTP status 200) has the following format:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 149

{
 "success": false,
 "message": "LexicalError at line 1 column 39 (index 38): [...ues on
 $structure in # ? $period where indi...]",
 "line": 1,
 "column": 39
}
```

| Name | Data Type | Description |
|---|---|---|
| `success` | Boolean | Specifies that the syntax was invalid (`false`). |
| `message` | String | A textual message describing the problem. |

| Name | Data Type | Description |
|---|---|---|
| | | ☞ **Note:** Only the first problem is reported, therefore repeating the call with the given problem fixed may still fail the syntax check for another reason. |
| `line` | Number | The line of the error (1-based), or 0 if the error was generic. |
| `column` | Number | The column of the error (1-based), or 0 if the error was generic. |

# Cache API

Control cache of SERAM.

### Overview

The Cache API allows external applications to clear the cache, or trigger cache maintenance. This operation can be useful in automation scenarios, or if the cache got out of sync for any reason.

### Maintenance Request

The request for performing a cache maintenance is sent via HTTP `POST` to the `Cache` URI including the application key as specified in the SERAM Applications management page. A `force` flag can be passed as JSON object or via query parameter to force the operation.

### Clear Request

The request for performing a cache maintenance is sent via HTTP `DELETE` to the `Cache` URI including the application key as specified in the SERAM Applications management page. No payload is expected.

### Response

If no error occurred, the HTTP status 200 is returned (no body), otherwise an error message is returned as JSON.

# Appendix A

# Formulas

The great configuration flexibility in SERAM is achieved in parts by using a rich formula language describing calculations, default values, constraints, plausibility rules etc.

The formulas come in two flavors:

- **Expressions** for generating usually numeric results, such as calculations and default values.
- **Predicates** for evaluating conditions, such as whether a constraint or a plausibility check holds.

The flavor of the formula is determined by the outermost syntax element: if it is a comparison operator, logic operation or just a boolean literal, then the outcome will be a predicate, otherwise it will be an expression.

**Note:** In the formula, expressions may use predicates (for instance in a conditional) and predicates will usually also use numeric expressions (such as the operands of a comparison). Therefore, the presence of either element inside the formula does not affect the type of formula.

## Formula Syntax Elements

All formulas are based on a common language and syntax.

### General Syntax

The formula language always describes a single formula. The different tokens are separated by white-space.

**Note:** The kind of white-space used when writing a formula is irrelevant. Since the application uses a different format for internal storage to ensure that the indicator references remain valid (e.g. indicator codes can be changed without breaking their references in formulas), the white-space is normalized, so that any white-space formatting (such as newlines or indentation) is not retained. Also, comments are not supported for the same reason.

**Note:** The formulas are side-effect free, that is, they do not change the state of the application or indicators. For this reason, no re-assignment operator exists to change any value, and defined variables remain static after their definition.

### Primitives and Simple Values

**Indicator Reference (IndicatorValue)**

A double-quoted identifier referencing the code of an indicator is used to retrieve the full value of an indicator in the same reporting period as the indicator using the formula.

```
"My Indicator"
```

An optional integer offset can be used to specify the reporting period in which this indicator shall be retrieved.

```
"My Indicator"(2009)
```

When a sign is prepended (+ or `-`), then the offset is seen as being a year offset relative to the reporting period of the defining indicator.

```
"My Indicator"(+1)
```

```
"My Indicator"(-1)
```

### Numeric Literal (number)

Numeric literals can have different representation formats; integer, floating-point or exponential are all valid. However, in order to avoid culture ambiguities, the number format is culture-independent: the dot (`.`) is always the fraction separator and no thousand separators or commas are allowed.

```
12
```

```
6.32
```

```
4e6
```

```
1.5e-3
```

### Boolean Literal (boolean)

Comparison operators return a boolean value. However, they can also directly be written as literals.

```
true
```

```
false
```

### Null Literal

The null literal represents a missing/non-existent value or object. While this is not a common literal, it can for instance be used to return an empty value depending on a condition.

```
null
```

### Variable Reference

Variables can be defined as part of the formula. They always start with a `$` sign, followed by a single-word identifier or a quoted identifier for complex variable names.

```
$var
```

```
$"Some Var"
```

### Constant Properties

Some predefined values exist which can be retrieved by their name.

**Note:** This is also true for the actual value of an indicator, which is constant in the scope of the formula evaluation.

```
value
```

### Function Call

Functions can be called by specifying their name, followed by braces holding their arguments (comma-separated if there is more than one argument).

```
sqrt("A")
```

## Numeric Operators

### Addition and Subtraction

These are represented by the traditional `+` and `-` symbols, and they can be used as both unary and binary operators.

**Note:** During computation of these operators, any `null`/`NaN` operand (for instance an indicator without a value) is interpreted like a `0`. This behavior is different from most formulas interpreters where a missing or invalid (NaN) value invalidates the complete result. A practical usage of this bahavior is in calculations, when multiple indicators which may be disabled on some structure nodes are summed together.

```
+$a
```

```
-4.5
```

```
"A" + "B"
```

```
"Indicator" - 10
```

### Multiplication and Division

These are represented by the traditional `*` and `/` symbols.

**Note:** If any of the operands is `null`/`NaN`, the result will also be `null`/`NaN`.

```
"Electricity Use" * "Electricity Emission Factor"
```

```
"Total Turnover" / "Headcount"
```

### Modulo

The modulo operator is written `mod`.

**Note:** If any of the operands is `null`/`NaN`, the result will also be `null`/`NaN`.

```
"Items" mod "Packaging Size"
```

### Percentage

The `%` post-unary operator is equivalent to dividing a value by 100, and it is mostly used to make formulas more intuitive to read.

```
"Total Employee Hours" * 25%
```

## Comparisons

### Relational Operators

The formula language supports the common relational operators for equality and relative comparison: equal = (alternate representation ==), not-equal != (alternate representation <>), greater-than >, greater-than-or-equal >=, less-than <, less-than-or-equal <=.

```
"Waste Tons" < 5
```

### Range Comparison

The operator takes three operands: the value to compare and the lower and upper bounds (inclusive).

```
"Temperature °C" is between 20 and 25
```

**Note:**

The example formula above performs the same comparison as the following with relational operators:

```
("Temperature °C" >= 20) and ("Temperature °C" <= 25)
```

### Relative Comparison

This operator takes at least three operands: a value to compare, a deviation (often expressed in percent) and one or more comma-separated target values to compare against. The compared value must match all targets for this operator to return true.

```
"Emissions" is within 10% of "Compensation"
```

**Note:**

The example formula above performs the same comparison as the following with relational operators:

```
$offset := abs("Compensation" * 10%),
("Emissions" >= ("Compensation" - $offset)) and
("Emissions" <= ("Compensation" + $offset))
```

### Comparing against `null`/`NaN`

Since `null` and `NaN` have a special meaning, values can be checked against these using `is null` respectively `is not null` or `is NaN` respectively `is not NaN`. By using this type of comparison, you can work around the special handling of `NaN` as defined in the IEEE 754 standard (used for the number representation in SERAM) which mandates that a `NaN` is not equal to another `NaN`, e.g. `NaN == NaN` is false.

```
"Electricity" * (if "Factor" is not null then "Factor" else 1.5 end)
```

## Logical Operators

### Logical Or

The logical `or` returns true when either of the operands is true.

```
($a < 0) or ($a > 100)
```

### Logical And

The logical `and` returns true only when both of the operands are true.

```
($b >= 0) and ($b <= 100)
```

### Logical Not

The unary logical `not` returns the opposite of its operand.

```
not ($c = 0)
```

## Numeric Variables

### Defining a Variable

Numeric variables can be defined almost in any place of the formula, but they only remain accessible in their scope (that is, the expression or predicate following the declaration).

**Note:** Due to the side-effect free nature of the formulas, the value of an existing variable cannot be replaced by re-defining the variable.

### Using a Variable

Variables defined can be accessed everywhere using their name.

```
$energyPerEmployee := "Energy Use" / "Employee Count",
"Energy Emission Factor" * $energyPerEmployee
```

## Conditional Evaluation

### Single Conditional Test

When the condition operand returns true, the first expression is evaluated, otherwise the second expression is evaluated.

**Note:** This can be seen as a ternary operator.

```
if "A" < 0 then "B" else "C" end
```

The `else` part may be omitted, which then returns a `null` value.

```
if "A" > 0 then "C" end
```

### Multiple Conditional Test

A single numeric expression can also be compared for equality against a series of numeric values.

**Note:** This could also be achieved by using a chain of single conditional tests.

```
choose "A"
when 1 then "B"
when 2, 3, 10 then "C"
else NaN
end
```

The `else` part may be omitted, which then returns a `null` value.

```
choose "A"
when 1 then "B"
when 2, 3, 10 then "C"
end
```

## Numeric Constants and Functions

### NaN (number)

The “not a number” constant defines a missing value.

```
NaN
```

### Euler's number

The *Euler's number e* is used in relation to natural logarithms.

```
e
```

### Pi (number)

The constant *Pi* is used for geometric computations.

```
pi
```

### Absolute (number → number)

This function returns the absolute value, that is, without its sign.

```
abs("Energy Use" - "Energy Use"(-1))
```

### Average (number... → number)

This function accepts any number of arguments and computes their average.

**Note:** Missing (`NaN`) values are ignored in the computation.

```
avg("Energy Use", "Energy Use"(-1), "Energy Use"(-2))
```

**Ceiling (number → number)**

This function returns the largest integer greater than or equal to the specified argument.

```
ceil($x)
```

```
ceiling($x)
```

**Coalesce (number... → number)**

This function returns the first non-missing (`NaN`) value from any number of arguments.

```
coalesce("Custom Factor 1", "Custom Factor 2", 2.54)
```

**Concatenate (any... → string)**

This function takes all arguments and builds a string from them, converting each the same way as the `string()` function does.

```
concat("A", '; ', "B")
```

**Contains (string, string → boolean)**

This function searches the input string (1st argument) for the given pattern (2nd argument). The search is case-insensitive.

```
contains("A", 'test')
```

**Exponent (number → number)**

This function returns *e* raised to the power specified as argument.

```
exp($x)
```

**Floor (number → number)**

This function returns the largest integer less than or equal to the specified argument.

```
floor($x)
```

**Length (string → number)**

This function gets the length of the input string. If the string is `null` or empty, `NaN` will be returned.

```
length("A")
```

**Logarithm (number, number → number)**

This function returns the logarithm of a specified argument and with the specified base.

```
log($x, 2)
```

**Match Regular Expression (string, string → boolean)**

This function matches the input string (1st argument) against the given regular expression pattern (2nd argument). The search is case-sensitive, but this can be controlled by using the `(?i)` mode modifier.

```
matches("A", '\w\w\w')
```

**Note:** An invalid regular expression will always have the function return `false`, the formula will not fail.

### Natural Logarithm (number → number)

This function returns the natural (base e) logarithm of a specified argument.

```
log($x)
```

**Note:** This example is equivalent to `log($x, e)`.

### Decimal Logarithm (number → number)

This function returns the base 10 logarithm of a specified argument.

```
log10($x)
```

**Note:** This example is equivalent to `log($x, 10)`.

### Maximum (number... → number)

This function returns the largest of the arguments, ignoring any missing (`NaN`) value.

```
max($x, "A", 5)
```

### Minimum (number... → number)

This function returns the smallest of the arguments, ignoring any missing (`NaN`) value.

```
min($x, "A", 5)
```

### Number Representation (any → number)

Convert the given argument to a number. If the argument is a *IndicatorValue* (e.g. as returned by the named indicator reference), then it will extract the numeric reporting value of the indicator. Otherwise, the system will try to convert the input to a number in a culture-invariant manner (same number format as the number constants in the formula). If the conversion fails, a `NaN` will be returned.

```
number("A")
```

### Power (number, number → number)

This function returns a specified argument raised to the specified power.

```
pow($x, 2)
```

```
power($y, 10)
```

### Round (number, number? → number)

This function returns the specified argument to an optionally specified integer number of fractional digits. If the fractional digits are not specified, it is rounded to the next integer number.

**Note:** Rounding is always done away from zero, e.g. 0.5 is rounded to 1, and -0.5 is rounded to -1.

```
exp($x)
```

### Square (number → number)

This function squares the specified argument, that is, it is multiplied by itself.

```
sqr($x)
```

### Square Root (number → number)

This function returns the square root of the specified argument.

```
sqrt($x)
```

### Standard Deviation (number... → number)

This function computes the *Corrected Sample Standard Deviation* from the given arguments.

```
std($x, $y, $z)
```

### String Representation (any → string)

Convert the given argument to a string. If the argument is a *IndicatorValue* (e.g. as returned by the named indicator reference), then it will extract the text indicator value of the indicator. Otherwise, the system will try to convert the input to a number in a culture-invariant manner. If the input is `null`, a `null` will be returned.

```
string("A")
```

## Indicator Formulas

The formulas used as part of indicators (e.g. in Reporting Indicators or Calculations) may use additional constants and functions.

### Convert Unit (number, string, string → number)

This function performs a unit conversion of the numeric argument from the unit symbol in the 2nd argument to the unit symbol in the 3rd argument. This will use the conversions as defined in the *Units* module.

```
convert(value, 'btu', 'kWh')
```

**Note:** If no such unit or conversion exists, the result will be `NaN`, the formula will not fail.

### Convert Unit of Indicator Value (IndicatorValue, string → number)

This function performs a unit conversion just like the `convert()` function with 3 arguments, but it retrieves the numeric reporting value and reporting unit directly from the indicator value. This is particularly useful when dealing with indicator references, since the source unit doesn't have to be specified.

```
convert("A", 'kWh')
```

### Unit Of Indicator (IndicatorValue → string)

This function retrieves the reporting unit from an indicator value. The second usage example converts the indicator A value to the same unit as indicator B and then compares their numeric values.

```
unitOf("A")
```

```
if convert("A", unitOf("B")) = "B" then ...
```

### Comment Of Indicator (IndicatorValue → string)

This function retrieves the comment test from an indicator value.

```
commentOf("A")
```

### Completeness Of Indicator (IndicatorValue → number)

This function retrieves the completeness from an indicator value (0..1).

```
completenessOf("A")
```

### Data Quality Of Indicator (IndicatorValue → number)

This function retrieves the numeric data quality number from an indicator value (0..100).

```
dataQualityOf("A")
```

### File Attachment Count Of Indicator (IndicatorValue → number)

This function retrieves the number of file attachments from an indicator value.

```
fileAttachmentCountOf("A")
```

### Forecasted Of Indicator (IndicatorValue → bool)

This function retrieves the forecasted flag from an indicator value.

```
forecastedOf("A")
```

### Status Of Indicator (IndicatorValue → string)

This function retrieves the invariant status name from an indicator value.

```
statusOf("A")
```

### Id Of Indicator (IndicatorValue → string)

This function retrieves the persistent value ID from an indicator value. Note that consolidated, aggregated, computed and flowed values do not have a persistent ID.

```
idOf("A")
```

### Is Editable Indicator Status (IndicatorValue → bool)

This function returns true if the indicator status is editable (open), and false otherwise. Note that this does not specify whether the value is editable or not.

```
isEditableStatus("A")
```

### Time Index (number)

This constant represents the reporting period time index of the value being computed. The exact mapping of the index to a time span is defined in the *Reporting Period Types*.

```
index
```

### Time Index Count (number)

This constant represents the count of reporting period indexes of the current frequency of the value being computed. For instance, for a Quarter index, this would return 4.

```
indexCount
```

### Time Index Days (number)

This constant represents the duration in days of the reporting period index of the value being computed, not taking into account the structure existence. For instance, for a January index, this would return 31.

```
indexDays
```

### Days (number)

This constant represents the effective duration in days of the value being computed. This duration depends on the index of the currently computed value (`indexDays`) and the structure creation/deletion dates. For instance, if a structure was created on Jan 10th, and the value time index being computed is January, then the `days` would be 22 days (10th to 31th) instead of 31.

```
days
```

**Structure (StructureInfo/Entity)**

This constant represents the structure of the current computation. It can be used to get metadata from the structure, find ancestors, children, or links.

```
structure
```

## Calculation Formulas

The formulas used when editing a Calculation have some additional constants available.

**Calculation Unit (string)**

The calculation result unit symbol, which can for instance be used to convert values with the `convert()` functions.

```
calculationUnit
```

**StatusOnly (IndicatorValue → IndicatorValue)**

Only return the data status part of an indicator value. This can be used to include indicator values without affecting the numeric result of the computation.

```
statusOnly("A")
```

**Meteo Data (number, number, string → number)**

Retrieve historical meteorological data, such as temperature (average, minimal, maximal), precipitation, snow height or wind speed.

```
$lat := structure.metadataNumber('lat'),
$lon := structure.metadataNumber('lon'),
meteoData($lat, $lon, 'tavg')
```

The following values are currently available:

| Key | Value |
|---|---|
| `tavg` | Temperature Average (in °C) |
| `tmin` | Temperature Minimum (in °C) |
| `tmax` | Temperature Maximum (in °C) |
| `prcp` | Precipitation Total (in mm) |
| `snow` | Snow Depth Maximum (in mm) |
| `wspd` | Wind Speed Average (in km/h) |
| `wpgt` | Wind Peak Gust (in km/h) |
| `pres` | Pressure Average (in hPa at sea level) |

## Reporting Indicator Value Formulas

The value-related formulas used when editing Reporting Indicators have some additional constants and functions available.

**Numeric Value (number)**

The numeric value is a constant holding the current numeric indicator value.

```
value
```

**Attachments (number)**

The number of files attached to the value.

```
attachments
```

**Completeness (number)**

The completeness of an indicator value can be queried using this constant, which can have any value ranging from 0 to 1.

**Note:** When a Reporting Indicator has a higher input frequency than its reporting frequency and the consolidated value is computed from a partial value set, then the completeness will be a fraction. For instance, a yearly reported indicator where data was entered quarterly will have completeness of 0.25 if only the first quarter contributed to the yearly value.

```
completeness
```

**Forecasted (number)**

The forecasted status of an indicator value can be queried using this constant, which can have the value 0 (not forecasted) or 1 (forecasted).

**Note:** This has only a value when forecasting has been enabled for the indicator.

```
forecasted
```

**Data Quality (number)**

The data quality of an indicator value can be queried using this constant. The values for the default data qualities are as follows:

| | |
|---|---|
| 20 | Estimated |
| 60 | Calculated |
| 100 | Measured |

**Note:** This has only a value when data quality has been enabled for the indicator.

```
dataquality
```

**Relative Numeric Value (integer-constant → number)**

The `relative` function allows to retrieve previous (or following) values, for instance to compare the numeric value to the previous numeric value in a plausibility check. The single argument must be an integer literal.

**Note:** The offset is always in reporting frequency increments. Therefore, if an indicator is quarterly, the offset `-1` will be the previous quarter (even across reporting periods), and the offset `-4` will be the same quarter in the previous reporting period. This differs from the normal indicator reference, which always refers to the same time index in different reporting period.

```
relative(-1)
```

## Reporting Indicator Custom Conversion Formulas

The formulas used when defining Reporting Indicator Custom Conversions have some additional constants available.

**Numeric Input Value (number)**

The numeric value is a constant holding the current numeric indicator input value.

```
inputValue
```

**Previous Numeric Input Value (number)**

The previous numeric input value. This value may be retrieved across reporting period boundaries if the time index is the first of its frequency in the reporting period.

```
previousInputValue
```

**Input Unit Symbol (string)**

The unit symbol of the input (source) value.

```
inputUnit
```

**Reporting Unit Symbol (string)**

The unit symbol of the reporting (target) value.

```
reportingUnit
```

## Create a calculation with meteorological data

SERAM can retrieve meteorological data in calculations, based on location.

A valid RapidAPI key must have been configured for the tenant, and the SERAM server must be allowed to query the RapidAPI servers (firewall etc.).

This task describes the setup required to retrieve meteorological data in a calculation.

**1.** Set up location information on structure nodes:<br>
**1.** Switch to the *Structures*.<br>
**2.** Click on the tab headers of the structure type and level you want to edit, e.g. *Site*.<br>
**3.** Click the **Edit … Level** action in the “Selected Level” category.<br>
**4.** Click the **Add Entry** action in the “Schema” category.<br>
**5.** Enter `lat` as “Key”, `Latitiude` as “Label”, and `Geolocation` as “Group Name”.<br>
**6.** Click **Apply** to add the latitude schema entry.<br>
**7.** Click the **Add Entry** action in the “Schema” category.<br>
**8.** Enter `lon` as “Key”, `Longitude` as “Label”, and `Geolocation` as “Group Name”.<br>
**9.** Click **Apply** to add the longitude schema entry.<br>
**10.** Click the **Save** button on the bottom to save the changes.<br>
**2.** Configure location information on structure node (repeat as needed):<br>
**11.** Navigate to and select the structure node (site) to which you want to add location information.<br>
**12.** Click the **Edit …** action in the “Selected Structure” category.<br>
**13.** Enter the “Latitude” and “Longitude” as decimal number. If unknown, you can look it up online, for instance in *Google Maps* or at *https://www.latlong.net/* (we are not affiliated).<br>
**14.** Click the **Save** button on the bottom to save the changes.<br>
**3.** Add an indicator to display the average temperature:<br>
**15.**<br>
**16.** Switch to the *Indicators Module*.<br>
**17.** Add a new indicator by using the **Add Indicator** action in the “Indicators” category. Fill out the metadata as you see fit.<br>
**18.** Click the **Activate ...** action in the “Selected Indicator” category to activate the newly created indicator.<br>
**19.** Pick the *Calculation Indicator* type and a *Reporting Period* on which the indicator is to be activated.<br>
**20.** Click the **Activate ... on ...** button on the bottom to continue.<br>
**21.** Enter `meteodata(structure.metadataNumber('lat'), structure.metadataNumber('lon'), 'tavg')` as formula.

22. Choose °C as calculation result unit.
23. Check the **Perform calculation on a specific level** checkbox, and choose the level where the geolocation has been configured (e.g. *Site*).
24. In the “Data Flow”, check **Direct** to prevent summation/consolidation of the temperatures.
25. Click the **Save** button on the bottom to save the new calculation.

You can now switch to the *Data Grid* module and load data for a structure and period where the new indicator has been configured. The average temperature of this location should appear as value.

# Appendix B

# Markdown Syntax

Markdown allows writing text using an easy-to-read, easy-to-write plain text format, which is then automatically converted to rich text format upon display.

### Overview

Markdown is intended to be plain text expressing formatting semantics while remaining as easy-to-read and easy-to-write as possible.

Readability, however, is emphasized above all else. A Markdown-formatted document should be self-explanatory as-is, as plain text, without looking like it's been marked up with tags or formatting instructions. The biggest source of inspiration for Markdown's syntax is the format of plain text email.

To this end, Markdown's syntax is comprised entirely of punctuation characters, which punctuation characters have been carefully chosen so as to look like what they mean. E.g., asterisks around a word actually look like *emphasis*. Markdown lists look like, well, lists. Even block-quotes look like quoted passages of text, assuming you've ever used plain-text email.

### Block Elements

**Paragraphs and line breaks**

A paragraph is simply one or more consecutive lines of text, separated by one or more blank lines. (A blank line is any line that looks like a blank line — a line containing nothing but spaces or tabs is considered blank.) Normal paragraphs should not be indented with spaces or tabs.

The implication of the “one or more consecutive lines of text” rule is that Markdown supports “hard-wrapped” text paragraphs, where the input text lines will be re-assembled into a continuous text flow (like Outlook does for plain text emails).

When you do want to insert an explicit line break using Markdown, you end a line with two or more spaces, then type return.

**Headers**

Headers are “underlined” using equal signs (for first-level headers) and dashes (for second-level headers). For example:

```
This is a Heading 1
===================

This is a Heading 2
-------------------
```

**Note:** Any number of underlining ='s or -'s will work.

Headings may also be created by starting the line with the corresponding number of hash (#) symbols:

```
### This is a Heading 3

#### This is a Heading 4
```

**Note:** The hash syntax is the only way to get 3rd-level and higher headings.

### Block-quotes

Markdown uses email-style > characters for block-quoting. If you're familiar with quoting passages of text in a plain-text email message, then you know how to create a block-quote in Markdown. It looks best if you hard wrap the text and put a > before every line:

```
> This is a block-quote with two paragraphs. Lorem ipsum
> dolor sit amet, consectetuer adipiscing elit. Aliquam
> hendrerit mi posuere lectus. Vestibulum enim wisi,
> viverra nec, fringilla in, laoreet vitae, risus.
>
> Donec sit amet nisl. Aliquam semper ipsum sit amet velit.
> Suspendisse id sem consectetuer libero luctus adipiscing.
```

Markdown allows you to be lazy and only put the > before the first line of a hard-wrapped paragraph:

```
> This is a blockquote with two paragraphs. Lorem ipsum
dolor sit amet, consectetuer adipiscing elit. Aliquam
hendrerit mi posuere lectus. Vestibulum enim wisi, viverra
nec, fringilla in, laoreet vitae, risus.

> Donec sit amet nisl. Aliquam semper ipsum sit amet velit.
Suspendisse id sem consectetuer libero luctus adipiscing.
```

### Unordered Lists

Unordered lists use asterisks, pluses, and hyphens — interchangeably — as list markers:

```
*   Red
*   Green
*   Blue
```

is equivalent to:

```
+   Red
+   Green
+   Blue
```

and:

```
-   Red
-   Green
-   Blue
```

### Ordered Lists

Ordered lists use numbers followed by periods:

```
1.  Bird
2.  McHale
3.  Parish
```

☞ **Note:** It's important to note that the actual numbers you use to mark the list have no effect on the HTML output Markdown produces, even if there are repetitions or skipped numbers in the text.

☞ **Note:**

It's possible to trigger an ordered list by accident, by writing something like this:

```
1986. What a great season.
```

In other words, a number-period-space sequence at the beginning of a line. To avoid this, you can backslash-escape the period:

```
1986\. What a great season.
```

### List Item Format

List markers typically start at the left margin, but may be indented by up to three spaces. List markers must be followed by one or more spaces or a tab.

To make lists look nice, you can wrap items with hanging indents:

```
*   Lorem ipsum dolor sit amet, consectetuer adipiscing
    elit. Aliquam hendrerit mi posuere lectus.
    Vestibulum enim wisi, viverra nec, fringilla in,
    laoreet vitae, risus.
*   Donec sit amet nisl. Aliquam semper ipsum sit amet
    velit. Suspendisse id sem consectetuer libero
    luctus adipiscing.
```

But if you want to be lazy, you don't have to:

```
*   Lorem ipsum dolor sit amet, consectetuer adipiscing
elit. Aliquam hendrerit mi posuere lectus. Vestibulum
enim wisi, viverra nec, fringilla in, laoreet vitae,
risus.
*   Donec sit amet nisl. Aliquam semper ipsum sit amet
velit. Suspendisse id sem consectetuer libero luctus
adipiscing.
```

List items may consist of multiple paragraphs. Each subsequent paragraph in a list item must be indented by either 4 spaces or one tab:

```
1.  This is a list item with two paragraphs. Lorem
    ipsum dolor sit amet, consectetuer adipiscing elit.
    Aliquam hendrerit mi posuere lectus.

    Vestibulum enim wisi, viverra nec, fringilla in,
    laoreet vitae, risus. Donec sit amet nisl. Aliquam
    semper ipsum sit amet velit.

2.  Suspendisse id sem consectetuer libero luctus
    adipiscing.
```

It looks nice if you indent every line of the subsequent paragraphs, but here again, Markdown will allow you to be lazy:

```
*   This is a list item with two paragraphs.

    This is the second paragraph in the list item.
You're only required to indent the first line. Lorem
ipsum dolor sit amet, consectetuer adipiscing elit.

*   Another item in the same list.
```

**Note:** You may nest all Markdown elements (including nested lists) into list items by applying the proper indentation.

### Code Blocks

Pre-formatted code blocks are used for writing about programming or markup source code. Rather than forming normal paragraphs, the lines of a code block are interpreted literally. To produce a code block in Markdown, simply indent every line of the block by at least 4 spaces or 1 tab. For example, given this input:

```
This is a normal paragraph:

    This is a code block.
```

**Note:** Regular Markdown syntax is not processed within code blocks. E.g., asterisks are just literal asterisks within a code block.

### Horizontal Rules

You can produce a horizontal rule (separator line) by placing three or more hyphens, asterisks, or underscores on a line by themselves. If you wish, you may use spaces between the hyphens or asterisks. Each of the following lines will produce a horizontal rule:

```
* * *

***

*****

- - -

---------------------------------------
```

### Tables

Tables can be specified in Markdown, including an optional heading row and text alignment of columns. Span formatting is also supported in cells. The pipe characters do not have to be exactly aligned for the table to be recognized:

```
| Column 1 Heading | Centered | Right |
| ---------------- |:---------:| -----:|
| Normal text      | *Italic* | $1234 |
| Not Aligned | | $10 |

| - |
| Single Column |
| No Heading |
```

## Text Span Formatting

### Emphasis

Markdown treats asterisks (`*`) and underscores (_) as indicators of emphasis. Text wrapped with one `*` or _ will be displayed as italic text; double `*`'s or _'s will be displayed as bold. For example:

```
*single asterisks is italic*

_single underscores is italic_

**double asterisks is bold**

__double underscores is bold__
```

### Links

The link text is delimited by `[`square brackets`]`. To create an inline link, use a set of regular parentheses immediately after the link text's closing square bracket. Inside the parentheses, put the URL where you want the link to point, along with an optional title for the link, surrounded in quotes. For example:

```
This is [an example](http://example.com/ "Title") inline link with a
 title.

[This link](http://example.net/) has no title.
```

**Note:** Common URLs (`http://...`) will automatically be recognized as links without the need for an explicit link.

**Note:** You can create e-mail links using the `mailto:...` prefix (e.g. `mailto:info@seram.ch`).

Links may also be created as references, which allows re-use of links:

```
[This link][ref] uses a reference.

[ref]: http://example.net/
```

### Images

Markdown uses an image syntax that is intended to resemble the syntax for links. Inline image syntax looks like this:

```
![Alt text](/path/to/img.jpg)

![Alt text](/path/to/img.jpg "Optional title")

![Alt text][ref]

[ref]: /path/to/img.jpg
```

That is:

- An exclamation mark: `!`
- followed by a set of square brackets, containing the alt attribute text for the image
- followed by a set of parentheses, containing the URL or path to the image, and an optional title attribute enclosed in double or single quotes

**Note:** You can use `data:` URIs in order to embed images. In this case the use of the reference syntax works best in order to avoid big chunks of data in the middle of the markdown text.

**Note:** An advanced feature is image sizing and positioning, which is done by adding attributes in the syntax `{ attr=value }` after the URL (after the closing parens). The `width`, `height` and `float` attributes are translated to styles. For HTML output, `.class` is converted to a CSS class, `#id` to an element ID, and all other attributes are converted to `data-*` attributes.

### Code

To indicate a span of code, wrap it with backtick quotes (`` ` ``). Unlike a pre-formatted code block, a code span indicates code within a normal paragraph. For example:

```
Use the `relative()` function.
```

### Placeholders

In some places such as report text blocks, placeholders with formulas may be embedded by using the same syntax as XML/HTML processing instructions, e.g. between `<?` and `?>`:

```
This report was created by <? $username ?> on <? now ?>.

<? markdown($text) ?>
```

## Footnotes

**Note:** This functionality is only available for Markdown in reports.

Footnotes can be used to add notes and references without cluttering the body of the document. When a footnote is created, a superscript number appears where you added the footnote reference.

### Footnote Reference (Span)

Footnote references are created by adding a caret and an identifier inside brackets (`[^1]`). Identifiers can be numbers or words.

**Note:** Identifiers only correlate the footnote reference with the footnote itself - in the output, footnotes are numbered sequentially.

### Footnote Content (Block)

The footnotes are added using another caret and number inside brackets with a colon and text (`[^1]: My footnote.`) as dedicated line.

**Note:** Footnotes do not have to be at the end of the document, they can be put anywhere except inside other elements like lists, block quotes, and tables.

```
Here's a simple footnote[^1], and a longer one[^XL].

[^1]: Just a simple footnote.

[^XL]: Here's one with multiple paragraphs and complex content.

    Indent paragraphs to include them in the footnote.

    `{ my code }`

    Add as many paragraphs as you like.
```