Rule Templates

Related article: Query Types for the Creation of Rule Templates

Rules can be parametrized with rule templates. Based on a selected query, the rule template that uses these parameters is instantiated by the In-Memory DB into several rule instances in which the parameters are filled. For run-time calculation, only the rule instances are used. The user can update the query-based rule at various points in the life cycle of a database model, such as after the result of the query has been modified or after changing the syntax of the rule template.

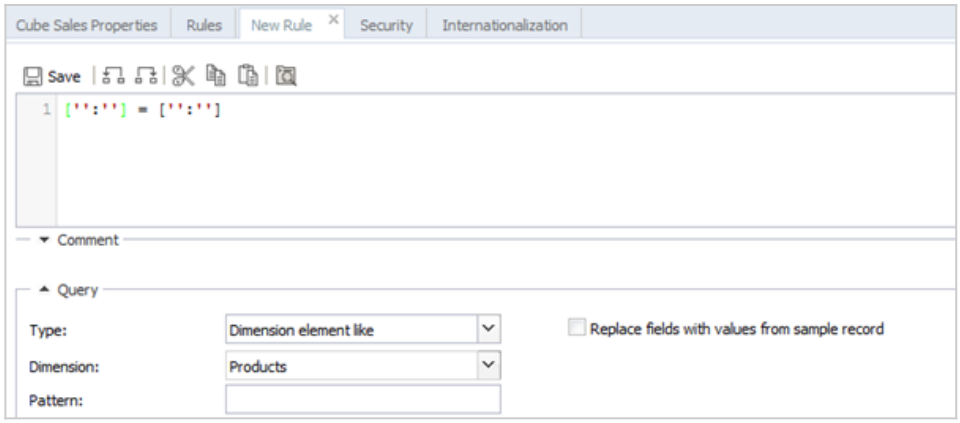
Imagine, for example, a rule that has to be defined for each of the twelve base-month elements in a time dimension. Using static rules, the user would have to write the same rule twelve times. But with rule templates, the user only writes one rule template, which the In-Memory DB uses to generate the twelve rule instances automatically. Also, if the rule syntax changes, the user only has to change the rule template; the change is then propagated to all rule instances.

The use of rule templates (creation and instantiation) requires Supervision Server to be activated. For details, see **SSO Configuration of Jedox Server on Windows or Linux**.

Creation of rule templates

Rule templates are created in the new graphical Rule Editor in the Jedox Web Modeler. The first step to creating a rule template is to

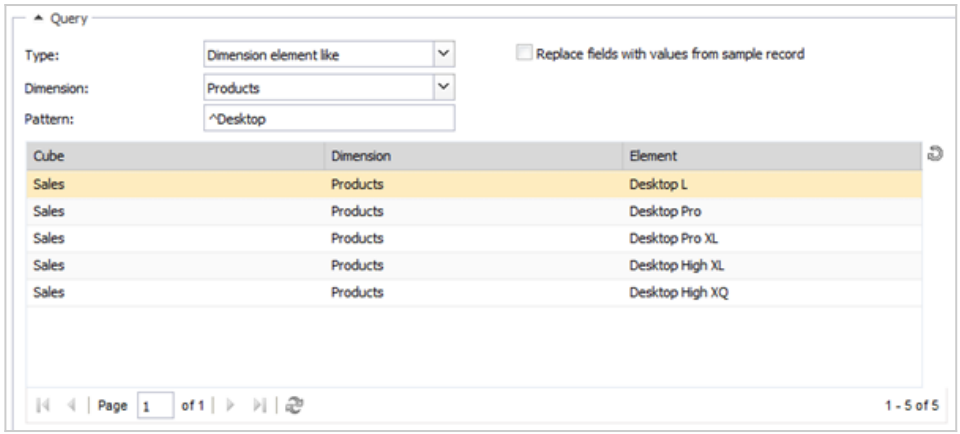
select a query type. There are several query types available, each allowing a different form of instantiation. Each query defines a set of user-specified input parameters. Based on the values for the input parameters, the query is executed.



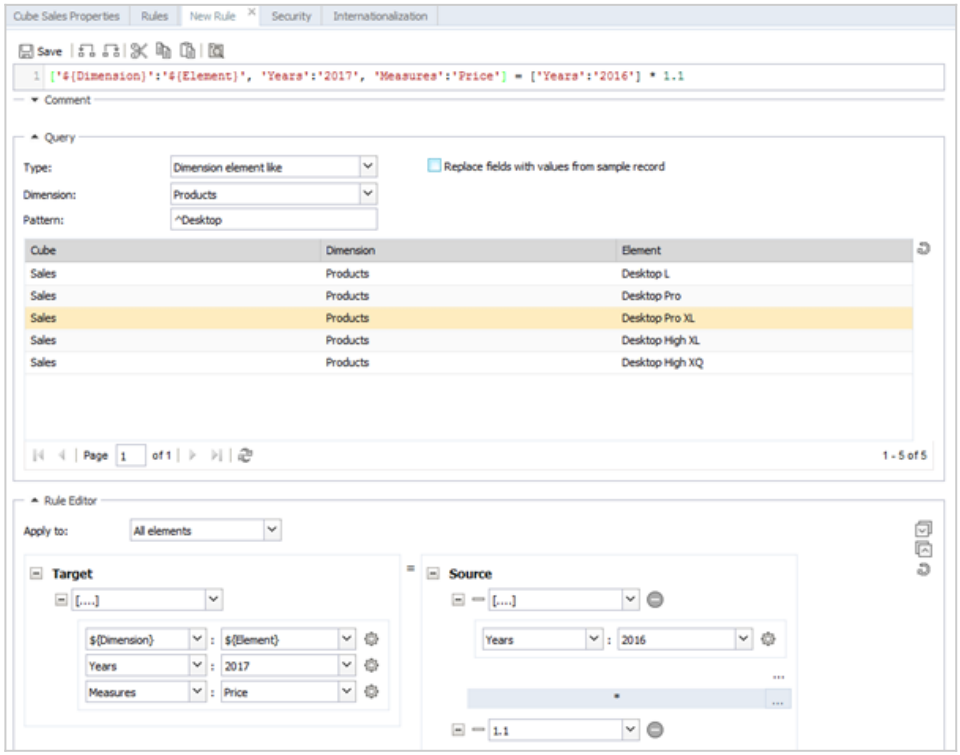
The result of a query is a record set consisting of rows and columns. Each column represents a potential variable that can be used in a rule definition; each row represents a record that will later be instantiated. Thus, for each returned row, there will be one rule instance generated, replacing the parameters from the rule template definition with the specific values from that row.

Note: rule templates use the same syntax for variables as Integrator rule extracts and loads: **${varname}**
If a rule template and an Integrator task use a variable with the same name, the variable will be handled by the Integrator task, not the rule template. Problems can be averted by changing the Integrator variable name on a project level.

In the following example, a rule template is generated to create instance rules for each Desktop... element in a Products dimension. To achieve this, the query type **Dimension element like** is used. The query expects two input parameters: a dimension (Products, in this case) and a RegEx filter pattern, which will be applied to all element names in the dimension. Each matching element is returned as a row in the query's record set.



The columns represent variables to be used in the rule definition. The variables can also be chosen from the combobox controls in the rule definition section. Variables are defined in the rule definition using the ${varname} decoration.

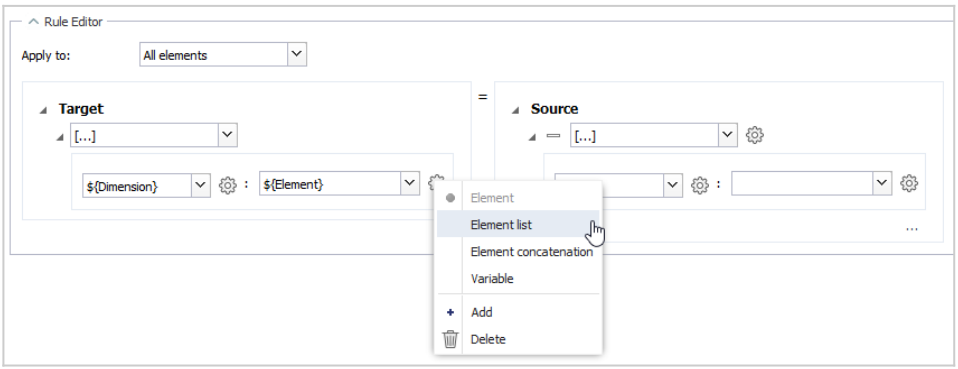


If you select a row in the query's returned record set and use the "Replace variables with fields from sample record" option, an example instantiation of the rule template is shown in the rule definition field.

When saving the rule template, the rule instances are automatically generated based on the returned record set. In the list of rules, instances are not shown by default, but can be shown by checking the "Show instance rules" checkbox. The Template column then shows the ID of the rule template from which the rule was generated.

Some queries (such as **Dimension element like (list)**) return an array, or a list of elements in a single record. To use the variables of these record sets with correct syntax in the rule controls, select **Element list** from the cog wheel, as shown below. Afterwards, on the

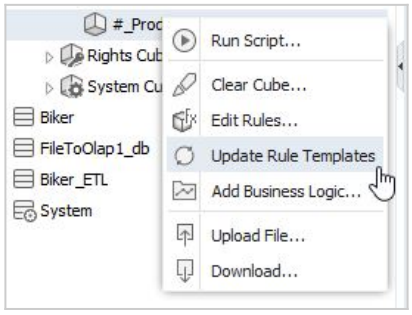
inner menu, select Variable.



In some cases, it may be necessary to concatenate a static string and the result of a field in a rule query within a rule parameter. This can be done by selecting **Element concatenation** from the cog wheel. You can then create multiple fields within the concatenation, each of which can be either a static text or a variable from the query:

Management and update of rule templates

Rule templates can be dynamically updated if the database structure changes. In the example above, the “Products” dimension could be extended with a new type of Desktop product, which also requires a rule. In this case, it is enough to update the instance rules, which can be done by right-clicking a specific cube or a database in the database tree in Modeler and selecting the “Update rule templates” option from the context menu.



You can also edit the rule template itself, like any other static rule. Upon saving the changes, the rule instances of the template are also updated.

Note: rule template definitions must be manually updated and instances refreshed if an object name changes. Objects include databases, cubes, dimensions, elements, and attributes.

Update of instances via Integrator

After a database is modified in the execution of a Jedox Integrator project (load or job), it may be necessary to update rule instances; for example, if the source dimension for a rule template changes. To update the rule instances, use the Integrator load type JedoxDatabase.