ADMINISTRATOR GUIDE

JACAMAR®

Version 5

KATLA GmbH
Immermannstraße 28
D - 39108 Magdeburg

Version: 5.201
Issue Date: 2015-06-16

Copyright © 2013-2015 Katla GmbH. All rights reserved.

Information in this document is subject to change without notice and does not represent a commitment on the part of Katla GmbH.

Katla's software and documentation have been tested and reviewed. Nevertheless, Katla makes no warranty or representation, either express or implied, with respect to the software and documentation included with Katla product. In no event will Katla be liable for direct, indirect, special, incidental or consequential damages resulting from any defect in the software or documentation included with this product. In particular, Katla shall have no liability for any programs or data used with this product, including the cost of recovering programs or data.

JACAMAR® is a registered trademark of Katla GmbH Magdeburg, Germany.

Microsoft, Windows 98, Windows NT, Windows 2000, Windows XP, Windows Vista, Windows 7, Windows 8 are either registered trademarks or trademarks of Microsoft Corporation.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle Corporation in the United States and other countries. For a complete, current list of the terms and conditions applicable to java.com, go to http://www.oracle.com/html/terms.html.

All trademarks and copyrights mentioned in this documentation are the property of their respective holders.

INHALTSVERZEICHNIS

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

1 Structure of this admin guide

This guide provides JACAMAR® administrators and users with all the information needed to build up complex structures within the software.

It starts with some information on how the JACAMAR software works and the ranges of use. This chapter lists the most important features for JACAMAR users as well as for administrators.

The term Repository will be explained in the first section and will describe how to work with repositories with comprehensible examples and screen shot illustration, including a brief guide to the menu options and icons.

The next two chapters explain the Meta Model, its structure, object types, classification and correlation are clearly defined, with vivid explanations on how to use the software and understand the model navigation language.

How to configure Views regarding user needs and how to use and display them within JACAMAR can be found in Chapters 6 and 7 .

Filter functions offer important possibilities on user-specific adjustments in JACAMAR. Therefore its description and handling is explained in chapter 8.

How to set up and maintain business-related user administration is contained in chapter 9.

The next sections are data assurance and the updating processes within JACAMAR and then the guide ends with an explanation of the configuration file (`jacamar.ini`).

JACAMAR administrators will be supported and guided by this book to do their daily business. They are the people in charge of the JACAMAR application (=Repository).

Administrators are the owner of the repositories:

- They define the structure of the Meta Model.
- They maintain all users, their roles and define their permissions.
- They provide the users with views according to their needs.
- They define and maintain global filters.

Administrators operate with special rights:

- There are no access restrictions to the data and views.
- There are more additional features:
  - access to data spreadsheets without views

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

- possibility to backup manually (e.g. before changing a Meta Model)
- additional functions for optimising and repairing data (e.g. defining relations right after changing the Meta Model)
- advanced export function
- access to user sessions to list all active users or to close selected user sessions

For better understanding and structure of this guide the following formatting is used:

Marking	Special words are formatted in **bold** when they appear for the first time.
COMMAND → FUNCTION	Commands and Functions are noted with SMALL CAPITALS.
Link	Links are marked in blue font colour and underlined.
`Code`	Source code, file names and system language are written in `mono space typewriter font`
'Proper name'	Names in examples are 'quoted'.
[ENTER]	Keys are marked with [CAPITAL LETTERS] in squared brackets.
Button	The formatting for a button is overbar and underline.
	This sign indicates advice, a recommendation or best practice.

Common abbreviations:

RMB Right Mouse Button

GMM Graphical Meta Model

TMM Tabular Meta Model

DND Drag and drop function

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

2 Overview of JACAMAR

The standard JACAMAR software enables you to build up team applications without any knowledge in coding. The software provides you with smart graphic features to create a team application, including individual views, of your data easily.

In JACAMAR you enter or edit data and share it with your colleagues at the same time.

It is very easy to build a JACAMAR application and to adjust it later. This flexibility enables you to start right now with a small project and it can be expanded when you want it to.

JACAMAR user benefits:

- to easily update data and views (instead of row numbers 2 and 245)
- only to adjust data they have the rights for
- to see all data at a glance instead of searching in several spreadsheets and wrong file versions
- to keep track of the information with the help of filter, search and sort functions
- to view data trees, that show clearly the data structure
- to work with helpful drop-down menus and RMB context options
- to create or update configurations with the drag and drop (DND) function
- the inheritance of data characteristics
- to get additional information (e.g. arranged individual views or sub views with extra information of the content)
- to import and export data easily
- to re-sort columns easily for a better individual view

JACAMAR is an excellent addition in line with spreadsheet and database applications, because JACAMAR administrators are able to:

- create, adjust and enhance team applications without coding
- ensure data security by assigning individual user rights

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

- build up highly complex structures within the software
- use a lot of beneficial features by building data models such as:
  - defining relations (between data objects)
  - free arrangement of data objects (very important for highly complex structures)
  - analysing data models via XML tools
- define work and analyse views to enable users for smart management of their shared data
- configure views with the help of intelligent user interfaces for example:
  - several types of views: spreadsheet, tree structure and sub view
  - defining start and end points (root rule and relation rules) supported by offering fitting possibilities and features in the input area
  - individual colour definition for several levels of data and cell background
  - defining columns using DND right out of the data model, sorting them in the same way
  - setting variables to use complex formulas in an easy way
  - providing a view in several languages
- maintain user rights and authorisation groups to a high level in JACAMAR:
  - two options for log-in: authentication directory access or use individual log-in ID with password
  - assign users, permissions and authorisation rights using DND

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

3 Repositories

3.1 Application rollout

1. Copy the **`JACAMAR-setup.exe`** (Windows) or the **`Zip`**- file to a temporary directory.
2. Start the **`JACAMAR-setup.exe`** or **`unzip`** JACAMAR.
3. You will be guided through the installation process.
    - Accept the Katla License agreement
    - The default installation location is **`/Katla/JACAMAR`**
4. After starting the JACAMAR Application you will be asked for a license (if you don't have a license you can continue with the JACAMAR Express Edition)

This Process is explained in detail in the **JACAMAR installation guide**.

3.2 Getting started – creating a repository

3.2.1 Generally

As the Administrator you must ensure that you have access to the entire data pool that you wish to include in the repository, this could be a sole project database or entire departmental and operational business data.

Before creating a repository you must ensure that you have decided on a Meta Model, a business model of information flow. The first option is a flat model, based on a single worksheet where all the data is stored in one table, or the second option a more business-related model, that separates data into various business-related areas and their correlation.

3.2.2 Wine Supply Sample

This guide is prepared in conjunction with a sample spreadsheet provided as a *Template* on the JACAMAR website. Download the Wine Supply spreadheet from our Website www.jacamar.de/service/templates and save it. There you will also find the completed Wine Supply repository, .jcmr file, for you to save in a specified folder[1] and use as a comparison of your progress.

The **Wine Supply** spreadsheet data is fictional and only intended to be used throughout this handbook as a guide to the practical application of the JACAMAR software and functionality. The Wine Supply data pool originates from multiple worksheets in a spreadsheet however data can be copied from multiple sources and moved into JACAMAR accordingly. This handbook refers to the Wine Supply data pool in various sections and you will need access to it to follow the guide.

Launch the JACAMAR software from your stored location (desktop, start menu, quicklaunch) and open the file menu to create a new repository (FILE → NEW REPOSITORY). Give the file a name e.g. WineSupply2, save it, and continue to section 4.5 to the next step of the Wine Supply example. (Using FILE → OPEN REPOSITORY you can browse and view the completed WineSupply.jcmr from its location for comparison at any time.)

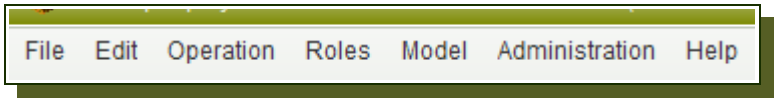
1 You need write access to that folder

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

3.3 Menus and Toolbar explained

3.3.1 Introduction

When opening a JACAMAR session, as an administrator, the menu bar and toolbar will appear and this section will explain the options and functionality (or chapter reference), from left to right across the screen. Depending on your JACAMAR edition and the application user role (normal user/administrator) the appearance of the menu may vary.



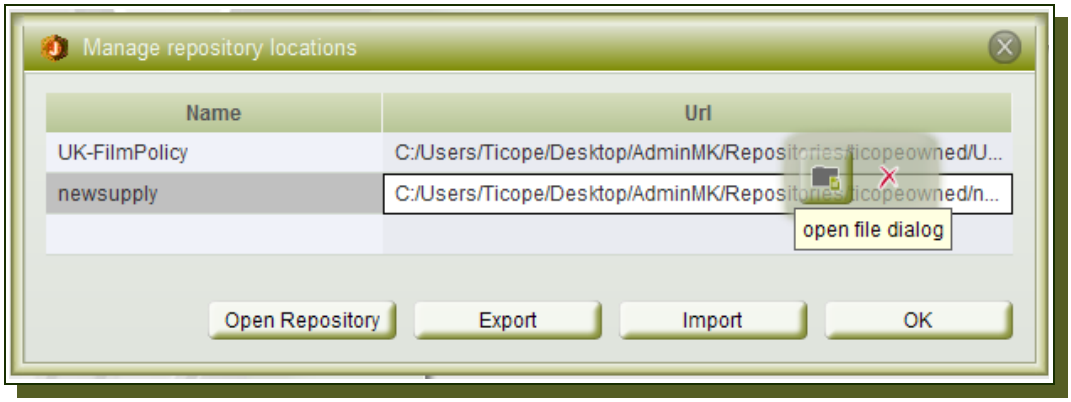
3.3.2 File Menu

NEW REPOSITORY prompts you to select a location and filename for your new repository.

OPEN REPOSITORY prompts you to browse to find the location of an existing repository.

RECENT REPOSITORIES gives you a submenu of repositories recently opened.

MANAGE REPOSITORIES opens a dialog where you can manage the repository list, export and import repositories.
RMB on a listed location allows you to open the file dialog box and to remove the location from the list. Selecting a location and clicking on OPEN REPOSITORY opens the repository directly.



CLOSE applies to the view open.

CLOSE ALL applies to all of the views open.

REFRESH updates the screen.

SAVE saves any changes made to the repository (and also clears the undo and redo lists).

EXPORT VIEW, allows you to place, name and save a view as a file for you to store or distribute to others. Select one view in the explorer, (if multiple views are selected the first one is used). An XML representation of the view will be stored in the local file system with the extension `*.view`.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

IMPORT VIEW allows you to locate and import an exported or saved view into your repository.

EXIT closes the JACAMAR session.

3.3.3 Edit menu

UNDO and REDO goes back or forward an action respectively.

DATA HISTORY shows all the saved changes to data.

COPY and PASTE are used for selected data from or into a view.

FIND is used to search for text in a view or in active JACAMAR windows.

Select all highlights all data in a view.

3.3.4 Operation menu

FILTER opens the filter dialog for a view, see section 8.3.

CLEAR ALL FILTERS closes all filters and returns you to the original view.

PARAMETRIC FILTERS allows administrators to create global filters for all users to apply in their views, see section 8.5.

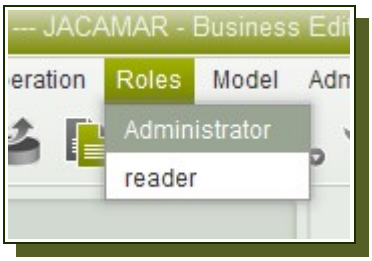
GLOBAL VARIABLES allows administrators to define navigation language statements as global variables to ease daily work. These variables can be used by all users in any navigation language context.

EXPORT DATA allows you to export all data from a view, see section 7.6.4.

START DATA MERGER only applies to users with a Data Merger license and an assigned role which enables you to import large volumes of data, via a schema based definition, from a **`.csv`** file and convert it across to a new or existing repository, see **Data Merger handbook** for details[2].

3.3.5 Roles menu

When a user has been assigned multiple roles in user administration for different permissions this menu option will enable the user to switch between the roles. It also allows an administrator to switch to a different role to test user administration role settings and for non-admin daily tasks.



2 A separate Data Merger license is required

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

3.3.6 Model menu

OPEN RAW TABLE gives an administrator a submenu to choose from and view or edit the raw data from any one of the Meta Model types. See below for the comment raw table, detailing the element Id, related element ids and comment element data.

Comment

Element-Id	children	parent	invoice	date	content
0			Invoice(0)	03-02-2014	Order received, Bob Young
1			Invoice(0)	10-02-2014	Missing 1 bottle Amarone
2			Invoice(0)		Late delivery, transport problem

EDIT META MODEL is also only available to administrators to make changes to the Meta Model.

TABULAR META MODEL opens the Meta Model tabular view used for DND

GRAPHICAL META MODEL opens the Meta Model graphical view used for DND

- Administrators can switch between edit and drag mode in the GMM in each of these Meta Model menu options.

3.3.7 Administration menu

OPEN USER ADMINISTRATION allows an administrator to manage users, roles and permission groups within a repository, see chapter 9.

CHECK ONLINE USERS allows administrators to view session locks and delete active user sessions.

SHOW ADMIN HISTORY allows you to see all changes made under Admin permissions, (Meta Model, view configuration, user administration).

BACKUP REPOSITORY allows administrators to manually create a backup of the repository.

- Non-admin users can only view user sessions and session locks.

3.3.8 Help menu

GETTING STARTED takes you to an online tutorial.

SWITCH LANGUAGE gives you a submenu to choose from English, German or French to change the language in the JACAMAR application.

LICENSE prompts you to the license dialog where you can enter your license order key, create a license file request, enter a valid license file or continue with the Express Edition.

ABOUT JACAMAR gives you software information and the registered license details.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

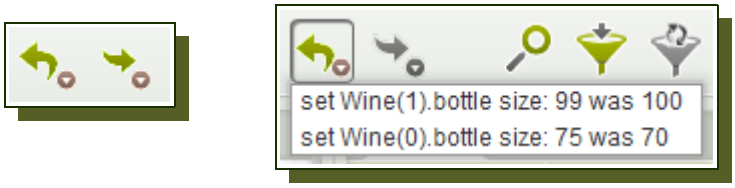
3.4 Toolbar Icons

RELOAD DATA FROM REPOSITORY allows a user to refresh the repository data when working offline.

SAVE ALL DATA saves any changes made to the repository (and also clears the undo and redo lists).

EXPORT DATA allows you to export all data from a view, see section 7.6.4.

COPY TEXT and PASTE TEXT are used for selected data from or into a view.



UNDO LAST ACTION and REDO LAST ACTION goes back or forward an action respectively and where multiple changes exist a list can be displayed when clicking on the white drop down arrow and you can select how far back or forward you want to go.

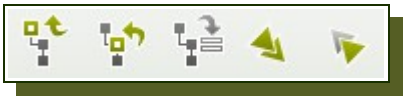
FIND TEXT is used to search for text in a view or in active JACAMAR windows.

OPEN FILTER DIALOG opens the filter dialog for a view, see section 8.3.

REFRESH CURRENT FILTER allows you to refresh the filter to reflect any changes in data.

CLEAR ALL FILTERS closes all filters and returns you to the original view.

SET INDIVIDUAL CELL COLOUR allows a user to open the dialog for colour selection and once a cell or cell range has been selected the colour can be applied, see section 7.6.5.



SHOW ALL LEVELS OF THE TREE displays all the branches of the tree structure before and up to the selected branch and its own branches.

GO UP ONE LEVEL displays the branch of the tree above that the selected branch is related to.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

SHOW SELECTED BRANCH ONLY displays only the branch selected in the tree and the related branches below it.

EXPAND ALL RELATIONS OF THE SELECTED BRANCH will expand all related lines for the branch selected, if no branch is selected all branches in the view will be expanded.

COLLAPSE ALL RELATIONS OF THE SELECTED BRANCH will collapse all related lines for the branch selected, if no branch is selected all branches in the view will be collapsed.

SHOW META MODEL opens the Meta Model window and can only be used for DND.

SHOW/HIDE VIEW CONFIGURATION is used for defining a view and its contents, see chapter 6.

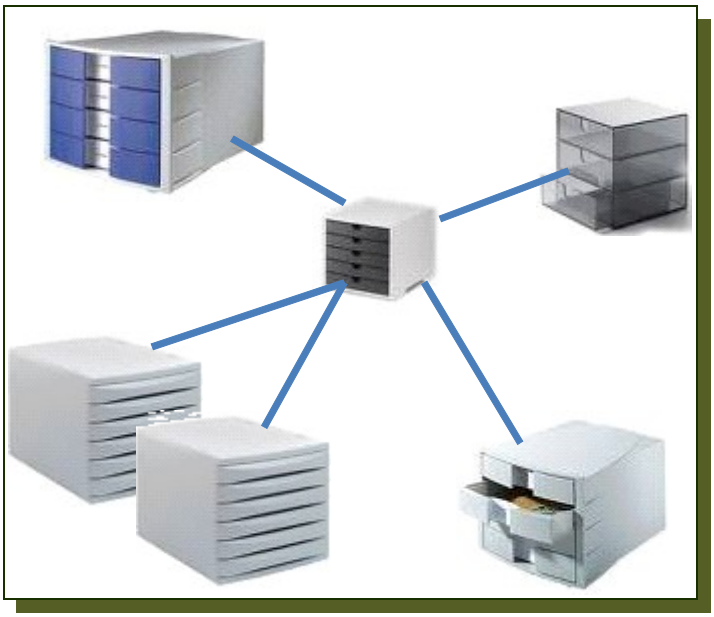
©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

4 Meta Model

4.1 Introduction

For better understanding a picture shall help to explain the structures and relations of a Meta Model in JACAMAR.

JACAMAR works with a Meta Model (business model) where lots of components can be structured and correlated. The components are free definable for designing every possible business process.



It is helpful to imagine an **object type** as a chest of drawers. Every drawer in a chest is called a **property,** and all the data is stored in the properties as **elements** in various data formats.

The Meta Model can be structured by defining several object types as different business needs are fulfilled by various chests of drawers.

The object types can be correlated by tying elastic bands between them. These bands are called **relations**.

By designing the Meta Model cleverly the relations between object types (referred to as **types**) and the data stored in their properties, referred to as **props**, is gettable as this can be navigated from one type to another.

The initiative of an administrator is to create a smart structure of types and props. There is more than one possibility to reach that goal: a lot of types filled with less information or less types filled with a lot of information.

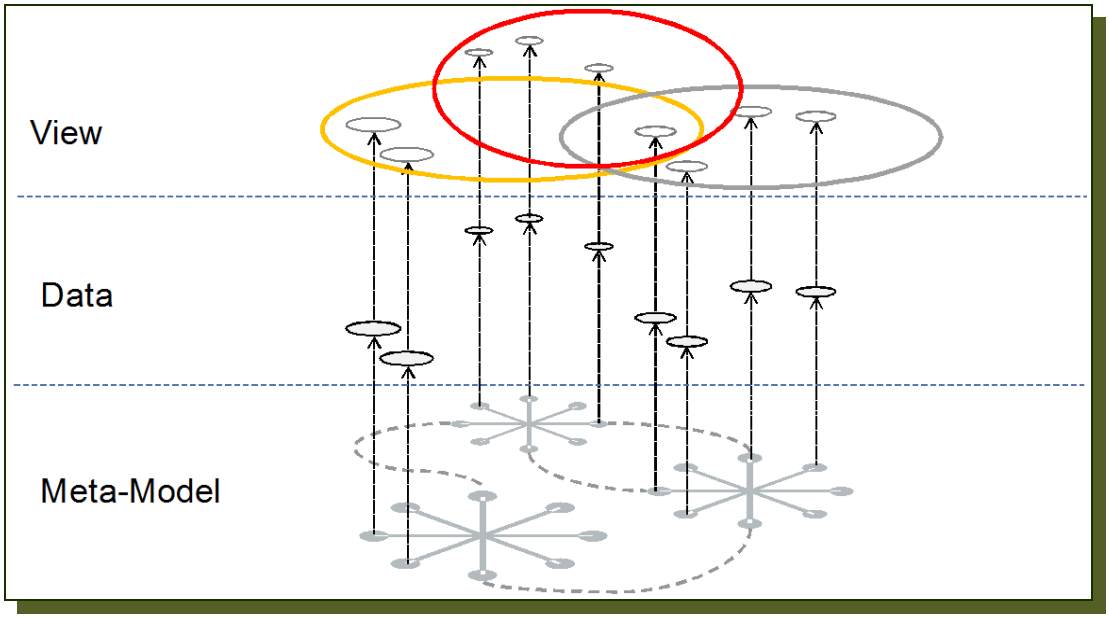
4.2 Design principles

Generally the business aspects have to be separated. That is the basis for defining transparent data models.

With JACAMAR you will get a dynamic flexibility of components and it is possible to update and enhance the model, even if you are already working with the configuration, as long as there is a rule for how to converse the Meta Model and data can be adjusted.

So, you can start with a small idea and a little Meta Model and it will get bigger when your business needs it to.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de



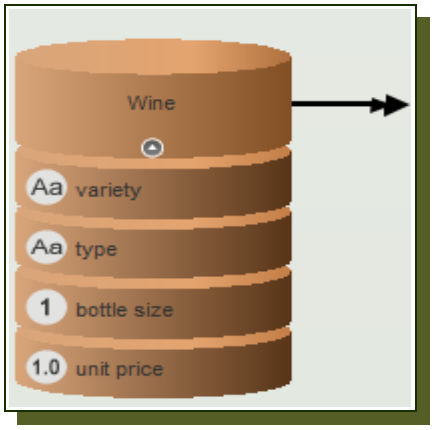
4.3 Types, properties and relations

4.3.1 Definitions

A specific kind of business need is defined by an **object type.**[3]

The attributes within each type are described as **properties**.

The connections between types are characterised by a **relation**.



Name	Data Type	Fix values
◢ Wine		
order	Order	to many
variety	String	
type	String	
bottle size	Integer	
unit price	Decimal	

3 In a database design or in a spreadsheet application it is called a table.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

4.3.2 Type definition

Within the graphical Meta Model types are displayed as a barrel and their characteristics can be changed by double clicking on a type in edit mode, see section 4.4.2. Within the tabular Meta Model types are marked with a light orange background and by expanding the type all attributed props and relations of that type will appear, as shown above.

A **type** is characterised by:

Name: Every type has to be named clearly.

Display-path: Here you may define a different name to display, e.g. a type „person" could be displayed as „first name, last name". If this field is left blank, the type name will be displayed togther with the element Id, e.g. person (35).

Dependendent of: There could be dependencies between types: a basic type is a composition of all „dependent of" types. That means one cannot exist without the other and:

- If you delete an element of a basic type, all elements of the dependent type will be deleted too.
- If a type is write-enabled, all dependent types are write-enabled too, without defining an explicit permission rule for the types and corresponding properties.

prnt/chld: Here you can create an internal parent child relation within an object type, whereby 1:n is when an element can be created within an element, and m:n is where an element can be created within many elements. This is required later in the Wine Supply example, see section 4.5, using 1:n with object type Comments whereby comments within comments within comments are possible.

Refer a type to itself (as structuring

a section containing

a section containing

a section)

description: Here you can leave a description of the type and notes regarding responsibility of the type.

controlled paths: Here you can name and create a built in variable used to differentiate multi-occurring elements as explained later in section 5.3.4.3.

artist symbol: Here you can change the colour of the type in the graphical Meta Model.

4.3.3 Property definition

A type can hold any number of properties and in the graphical Meta Model properties are listed under their corresponding type and in edit mode the characteristics of a property can be changed by double clicking on the property. Within the tabular Meta Model properties are marked with a cream coloured background.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

A **property** is characterised by:

Name: Every property has to be named clearly within the type.

Data type: Every property stores a specific data type format.

Allowed data types are: `String`, `Integer`, `Decimal`, `Boolean` and `Date`.

- `String`: is the default data type of a property and can be a combination of characters, numbers and symbols.
- `Integer`: is used for whole numbers (including negative whole numbers i.e. -2,-1,0,1,2).
- `Decimal`: is set for decimal values with the default being to two decimal places.
- `Boolean`: displays false or true statements, empty values will default to false, and the system recognises the following for pasted data:

no	false	0	nein	falsch
yes	true	1	ja	wahr

- `Date`: is used for date and time values where the system default is YYYY MM DD hh mm ss.

Fix values: A comma separated list of fixed values of valid data can be defined here and users can only set a value from this list.

Headline text: This text will be displayed in a column heading. The name of the property is set as default and can be adjusted.

Description: Here you can leave a description of the property and notes regarding responsibility of the property.

4.3.4 Relation definition

A relation defines a correlation between types. There are four relations that can be applied:

to-one: A to-one relation (e.g. between a country and its capital).

to-many: A to-many relation (e.g. between a country and its cities).

relation only directed to the other end: one way relation (e.g. between a postcode and a city).

many to many relation (e.g. a time zone related to its countries, a (large) country related to its time zones).

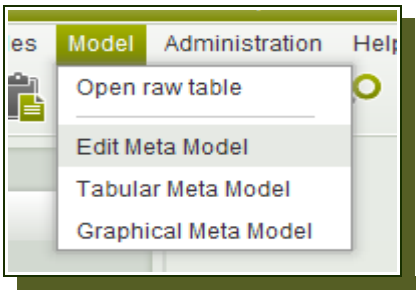
4.4 Handling of Meta Model editor

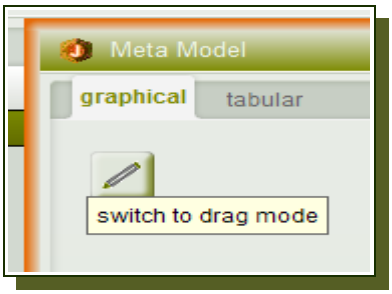
An administrator accessing the Meta Model within the model menu has two functions: the edit mode for making changes to the Meta Model components and the drag mode which is used for drag and drop when building views and filters. Changing between them is done with the pencil symbol switch toggle which appears in the top left of the graphical Meta Model window.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

4.4.1 Opening the Meta Model dialogue

The editor mode can be opened by selecting 'edit Meta Model' from the Model menu. When in edit mode the Meta Model window appears with a red border indicating that the edit function is open although the administrator can switch to drag mode using the switch.



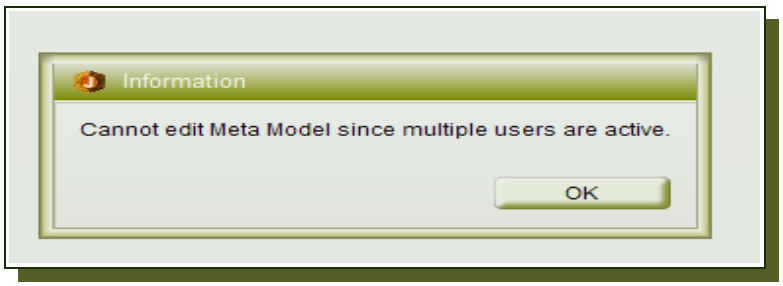


All registered users are allowed to view the Meta Model configuration. For editing some additional conditions have to be fulfilled:

- The user must have an administrator role and has to change the role to administrator.
- At the time of opening the editor no user is allowed to be logged into the repository.

Because adjusting a Meta Model means a big change of data structure it makes sense to define special maintenance time-frames for editing the Meta Model so as to limit the impact on users.

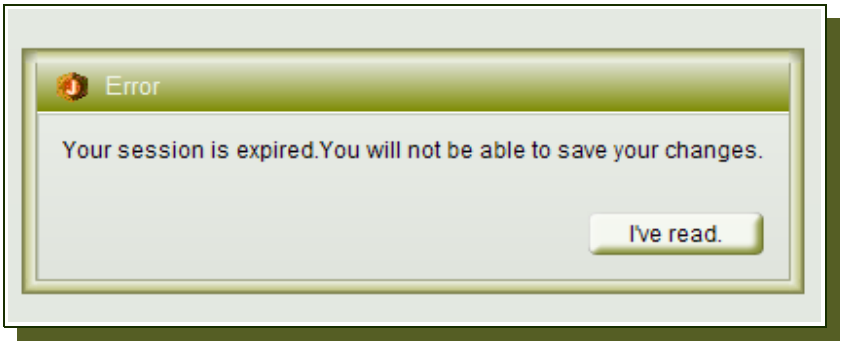
If users are signed in to the repository when opening the edit Meta Model function the administrator will receive the following message, and will either ask the users to logout or will need to delete the active user sessions using the 'check online users' function, see User Administration section 9.3.1.



If a user is trying to log in while a Meta Model is being edited, the following message occurs:

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

When a user has had their session deleted and they attempt to save any changes the following message appears:



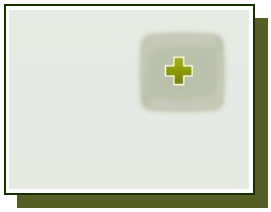
When the administrator has finished and switched from edit mode users can log into the repository again.

Changes to a Meta Model are directly updated in the GMM and TMM window and all connected types and properties will be updated. So administrators have to be pretty careful when adjusting a Meta Model and only SAVE the changes after checking for accuracy first.

A new repository Meta Model always opens as default with one type containing three properties with which you change and build your Meta Model according to the business model. Any changes to the Meta Model must be configured in the edit mode.

4.4.2 Creating, editing and deleting a type

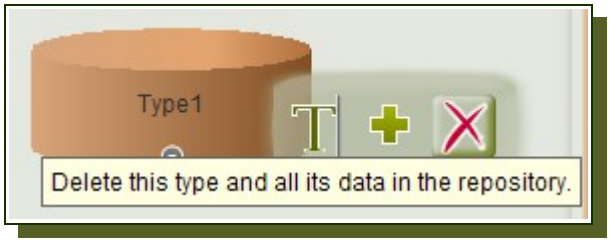
In the graphical Meta Model (GMM) a new type can be added by clicking the right mouse button (RMB) in an empty area and in the Tabular Meta Model (TMM) by using RMB when the cursor points on a type. The default name „new type" has to be adjusted.

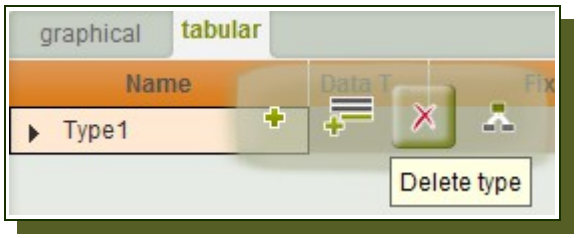


All of the characteristics of a new or existing type can be edited by double click or RMB edit on the type in the GMM, see section 4.3.2 for defintions, (only the type name, description and responsibility can be edited in the TMM)

©2015 KATLA GmbH | Immermannstraße 28 | D - 39108 Magdeburg | www.katla-gmbh.de

To delete a type simply RMB on the type in the GMM or TMM and select the red cross to delete the type. CAUTION must be taken when deleting a type as all of the properties, relations and data belonging to the type will also be deleted.

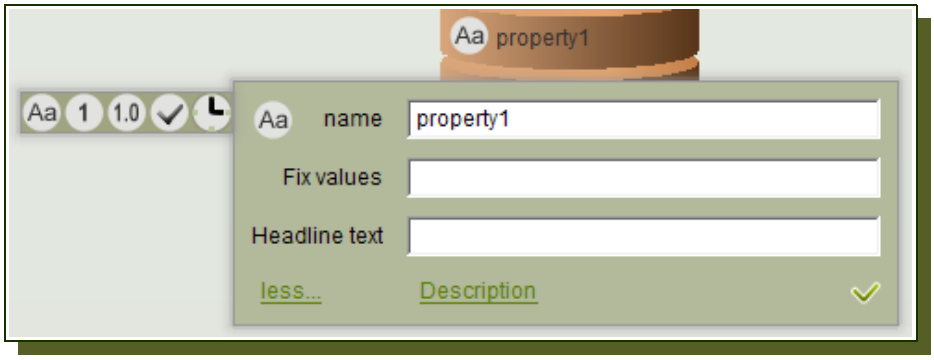




4.4.3 Creating, editing and deleting a property

A new property can be created in the GMM or TMM using RMB when the cursor points on a type or a property. If the cursor points on a type the new property will be inserted at the top of the list, and if the cursor points on a property the new property will appear directly after the selected one. The name „new property" has to be adjusted.

All of the characteristics of a new or existing property can be edited by double click or RMB edit on the property in the GMM. To change the data type click on the Aa symbol as shown below, (the characteristics can also be manually edited in the TMM using double click in the appropriate field, however data type formatting options are not provided in the TMM).



©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

The default of a property data type is String (Aa) and when changing a property data type all existing values will be changed too. You can't undo that so it is best to ensure the data type is correct for the data that is to be stored, see section 4.3.3 for definitions.

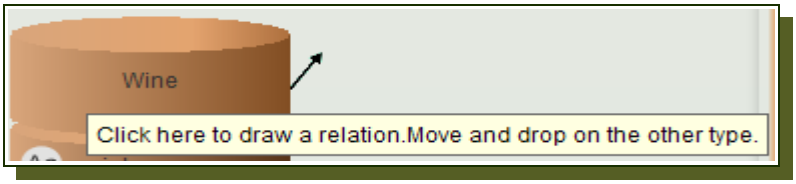
To delete a property simply RMB on the property in the GMM or TMM and select the red cross to delete the property. CAUTION must be taken when deleting a property as all of the data belonging to the property will also be deleted.

4.4.4 Creating, editing and deleting a relation

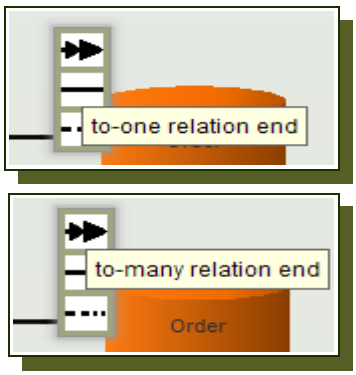
Relations can only be added, edited and deleted in the GMM.

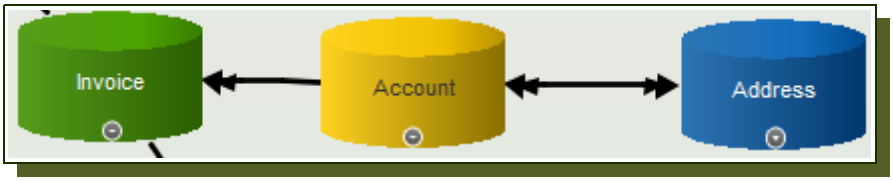
First you need to define the correlation between the types, this can be to-one, to-many, one way directed or many to many.

To create a relation you must position the cursor on the edge of a type, the cursor changes to a hand symbol and a small arrow appears, then click and drag the relation to the other correlated type.



By default the relation shows as a one-to-one relation. To edit the relation place the cursor at the end of the relation you wish to change, the cursor changes to a hand symbol, and click to choose the appropriate relation. Relations need to be edited at both ends for a many to many relation.

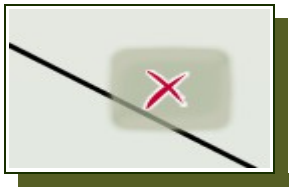


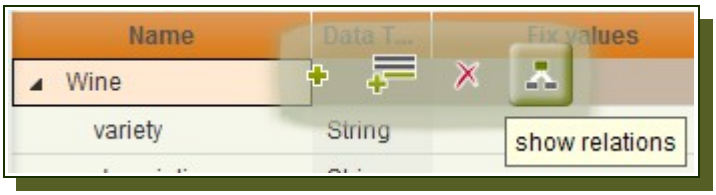


The example above shows a one (account)-to-many (invoices) and a many (addresses)-to-many (invoices).

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

To delete a relation simply RMB cursor on the relation and click the red cross. CAUTION must be taken when deleting a relation as all of the relations of elements between the connected types will be deleted.





You can view the relations in the TMM by RMB cursor on a type and select „show relations" and expand the types to see their properties and relations. To hide relations in the TMM it is the same process or you can RMB on a relation and select hide.

4.4.5 Re-sorting properties and types in the TMM

For display purposes it is also possible to change the order of the Meta Model types and properties within the TMM when in the edit mode. Select the type or property you wish to move and position the cursor on the bottom line of the component and when the cursor changes to the directional symbol you can drag and drop it to the preferred location in the TMM edit mode. As the component is dragged a black line appears in the background indicating where the component will be inserted. It is of course only possible to reposition a property somewhere within its type.

4.5 Creating your business Meta Model – Wine Supply example

As mentioned before, the process for building a business Meta Model is to define the different business areas, their specific properties and the correlation of data between the business areas.

In our Wine Supply data pool we have identified six business areas (types), their corresponding properties and the relations between the types as follows:

- Wine: variety, description, bottle size and unit price,
- Item Order: number, quantity and selling price,
- Invoice: number and paid,
- Account: number and name,
- Address: street, city, postcode and category,
- Comment: date and content.

A wine is related to many item orders, however one item order relates to one wine only therefore one to many, as also applies to an invoice which relates to many item orders, but only one item order correlates to one invoice. An account also has a one to many relation with invoices as an account can have many invoices but one invoice belongs to only one account. In our example an invoice can also contain many

©2015 KATLA GmbH | Immermannstraße 28 | D - 39108 Magdeburg | www.katla-gmbh.de

comments but each comment applies to one invoice, therefore a one to many relation again. And finally an Account can relate to many addresses and an address can relate to many accounts (due to an account having a delivery address and an invoice address, and an address having more than one client account).

The significance of the colour coding, type=dark blue, property=light blue and relation=green will be explained in the View Configuration chapter in section 6.2.4.

Following the steps explained in this chapter and using the business Meta Model described you can now use the edit Meta Model function and create the types, properties and relations in conjunction with the Wine Supply example, saving the changes regularly. As with our example above it is very important to ensure the data type format for the properties have also been edited to reflect the data to be stored or the data that is coming from the spreadsheet e.g. variety is String, bottle size is Integer (whole number), unit price is Decimal value, date is Date, paid is Boolean, refer to section 4.3.3 for a detailed explanation.

The three other attributes that need to be applied to the Wine Supply Meta Model are: Type Comment needs to be set to 1:n for a child/parent relation, allowing comments within comments, the Comment property date needs a default format DD-MM-YYYY and the Invoice property paid needs comma separated fix values of N,Y to replace the False/True default.

Once the Wine Supply Meta Model has been built and saved you can proceed to section 6.4 to continue with the next step of the Wine Supply example and configure the views.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

5 Model navigation-language

5.1 Introduction

The chapter before describes the structure and adjustment of a repository Meta Model.

Now it will be explained how to contact specific MM types out of the data. A special navigation path is needed to follow the relations that end at the information.

The navigation-path applies to:

- definition of the lines and columns that are configured and displayed within a view
- filter definition (complex filters and parametric filters)
- definition of the display-path name of object types

5.2 Structure of a navigation-path

A navigation path is structured as follows:

```
Type[FilterStatement].relatedType[FilterStatement].
property[FilterStatement]
```

In the beginning the root type is located. After that the related type follows and at the end the property appears.

Every segment is separated with a . (dot) and each segment can be specialised by an optional filter function [in squared brackets].

Not all segments of a navigation path are required as explained later in chapter 6, View Configuration.

The realisation of a navigation path is called evaluation.

5.2.1 Finding elements (absolute navigation path)

To define lines in a view an absolute navigation path is needed, which begins with the primary type (root rule).

```
Wine
```

This recognises all of the elements within this type from the background tables, for example all wine varieties.

5.2.2 Navigation from an element to a related element

This example routes from the elements of type „Wine" to the related elements of type „Item Order".

```
Wine.item order
```

The navigation path evaluation depends on the relation definition in the Meta Model.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

A defined to-one relation recognises for each „Wine" element one or no „Item Order" element.

A to-many relation provides for each „Wine„ element a list of „Item Order" elements which contain one or more or no elements.

5.2.3 Multi-cascaded navigation paths

A navigation path can pass across several levels as defined in the Meta Model.

```
Wine.item order.invoice
```

For each „Wine" a list of related „Item Order" elements will be recognised, which then moves across to the next level „Invoice" and therefore recognising those Item Order related „Invoice" elements. So for each „Wine" a list of of related invoices will be evaluated. The intermediate „Item Order" elements are invisible.

Handling to-many cascades are also possible. For each element of an intermediate level all elements of the next level will be collected. For dealing with such a big amount it is useful to work with filters.

5.2.4 Last cascade – a relation or a property

It is not only possible to navigate from a start-element to a related element but also to end a navigation-path with a property. The result will be one or more elements of the type.

The structure of the navigation-path depends on the user needs.

For **line definition** of views in view configuration the last segment has to be a type (or type `[FilterStatement]`), see section 6.2.2.

For all other cases the last segment has to be a property (or property `[FilterStatement]`).

If there is more than one element at the end of the evaluation of a navigation-path they will be displayed as a comma-separated list.

If there are real elements evaluated instead of properties, for each element the display-path name (see section 4.3.2) will be delivered as text and if the element is without a display-path name the type-name will be displayed with its element Id[4] in brackets.

number	name	address	invoice
1	Waldorf Ltd	Address(0), Address(1)	Invoice(0), Invoice(1)
2	Redstar Ltd	Address(0), Address(2)	Invoice(2), Invoice(3)
3	Mayflower Co.	Address(3)	

4 Primary key – consecutive number of an element <element.pk()>

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

5.3 Filter Statements

5.3.1 Kinds of filter statements

There are three kinds of filter statements that can be used within a navigation-path:

- range-filters
- comparison filters
- filter-functions

5.3.2 Range-filter

This filter reverts to a range of a result-list.

Here are some possibilities for syntax

`[3]`	Filters the 3rd element in the list
`[2..5]`	Filters from the 2nd to the 5th (incl.) element
`[2..]`	Filters all elements starting with the 2nd (counting starts at 0)
`[..4]`	Filters all elements up to the 4th (incl.), the same as [0..4]

5.3.3 Comparison-filter

This is the syntax of comparison-filters:

```
property = property
```

or

```
property = „hardcoded text"
```

besides the equal sign there are the following operators to use:

`>..`	starts with ... (for text)
`..<`	ends with ... (for text)
`<..>`	includes ... (for text)
`>=`	greater than or equal to
`>`	greater than
`<=`	less than or equal to
`<`	less than

A „!" in front of each comparison-operator inverts it e.g. „!=" for „not equal".

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

5.3.4 Filter-functions

There are several filter-functions that steer various evaluation-processes and deliver specific results. Some provide a list of elements, others numeric values.

Advice for syntax: the parameter *prop* means property-path. So the current element of the function will be navigated along this path and the list of results will be used as function-parameters.

5.3.4.1 List-functions

`Last()`
This function delivers the last element of a list.

A `first()` function is not needed because there is the range-function `[0]`.

`empty(prop)`

The input value list will be assumed completely, if the parameter-list is empty.

```
Wine.empty(item Order)
```

This delivers all „Wine" elements that don't have a connection to an „Item Order" element. Meaning which variety was not sold.

`has(prop)`

The input value list will be assumed completely, in case the parameter-list contains at least one element.

```
Wine.has(item Order)
```

So, all „Wine" elements will be delivered, that hold at least one „Item Order" element.

Other list functions include: `allElements()`, `parent()` and `this()`.

5.3.4.2 Tree structure functions

`hasChildren()`

From the input value all elements listed with child lines will be assumed, that means all elements that have child branches.

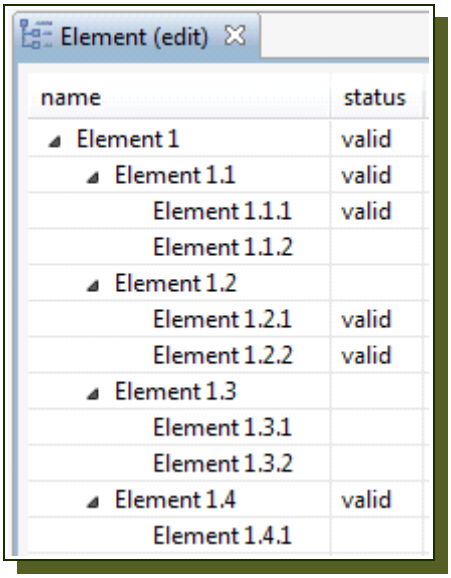
Don't confuse this function with the `has()` function in connection with the often used related lines like `has(relations)`.

This function works perfectly in connection with special filters for cascaded parent-chilld-trees, where the child rows come up out of the same line definition as the parent-row.

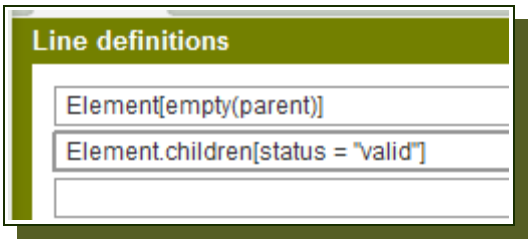
Element	----	----	----
children	Element	Many	parent
parent	Element	One	children
name	String	----	----
status	String	----	----

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

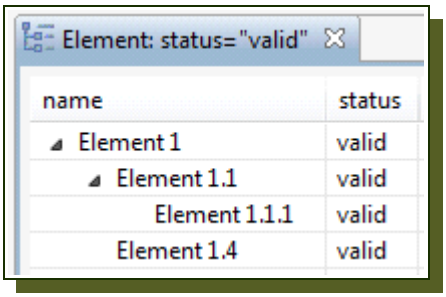
This example on the right shows the base type „Element" with its children elements and data-structure:



When the line defintions are for instance:



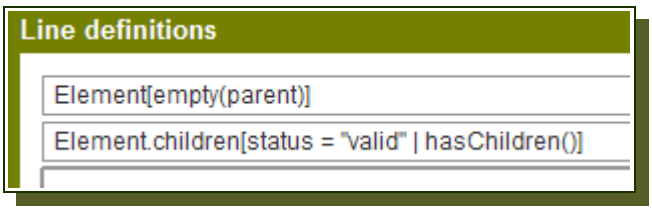
The filter results would be:

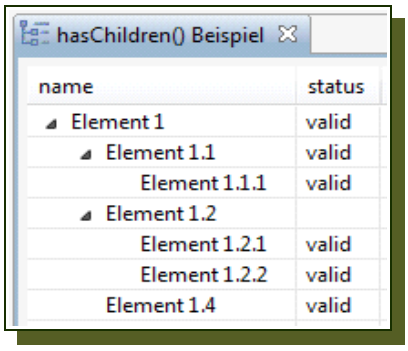


Based on the type „Element" and following the line definitions the results would mean that if an intermediate element doesn't fulfil the requirements (see element 1.2), its child-rows won't be displayed even if its children fulfil the requirement (like element 1.2.1 and element 1,.2.2 do).

With the OR-condition `haschildren()` function the valid sub-elements can be displayed, without destroying the overall structure.

This shows the result:



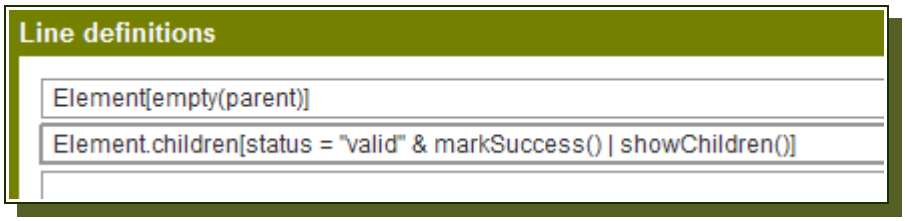


`markSuccess(),showChildren()`

These functions deliver all child-rows below a branch when a special requirement is fulfilled, no matter if each child fulfils it or not.

In the example above the element 1.4 is valid, its children aren't so they won't be displayed.

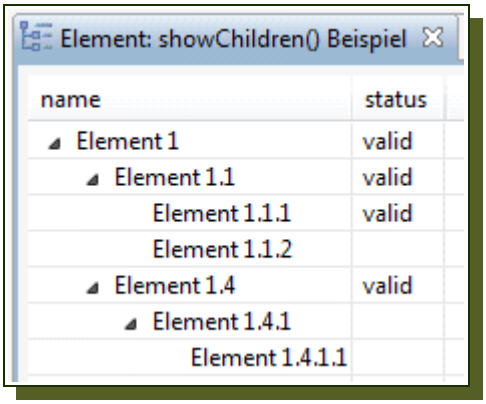
©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de



If a condition AND-connected with a `markSuccess()` function then all fitting elements will be marked as successful.

The OR-connected `showChildren()` function checks if there are successful marked elements in any parent levels and displays the current row, even if the requirement to the left of the OR-condition isn't fulfilled.

The filtered **tree-structure** would looks like this: with element 1.4.1 and several sub-elements being displayed.

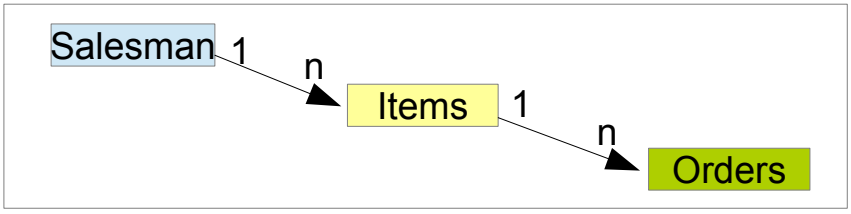


5.3.4.3 Multi-occurrence function

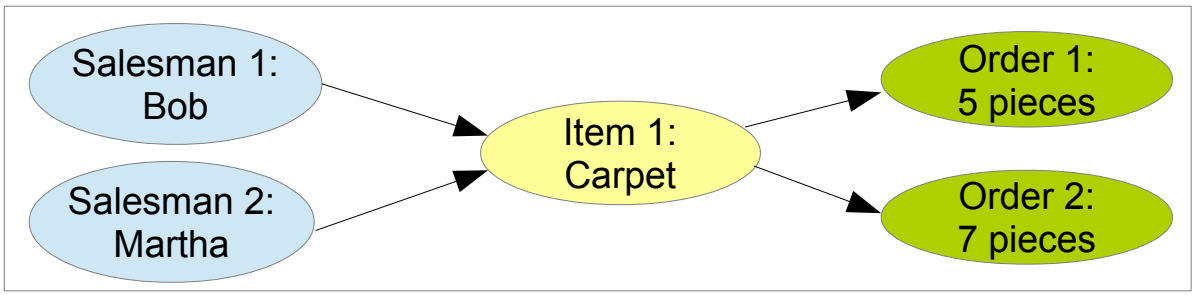
`cPath(prop)`

With the help of this function multi-occurring elements can be differentiated.

For example the following Meta Model: „Salesman" has a to-many relation to „Items" they sell. These „Items" are to-many related to associated „Orders". The following scheme can be built:

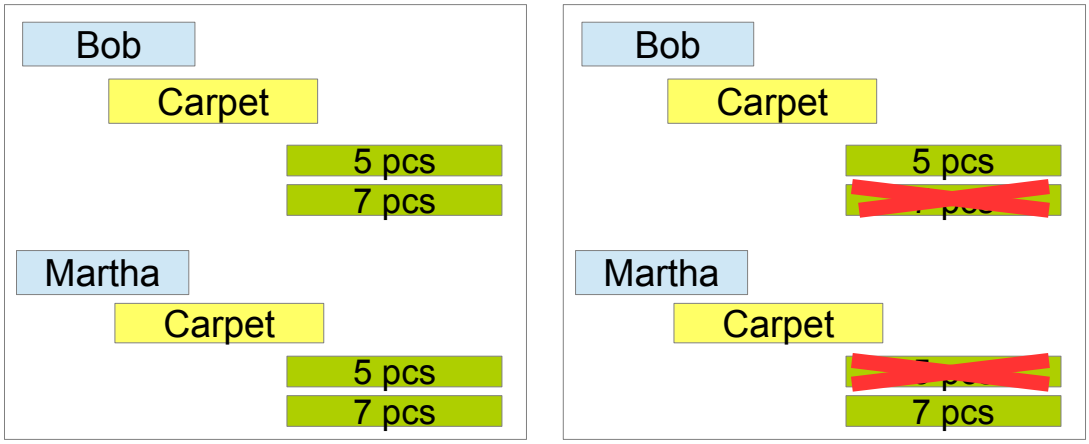


What shall be done if two salesmen sell the same item?



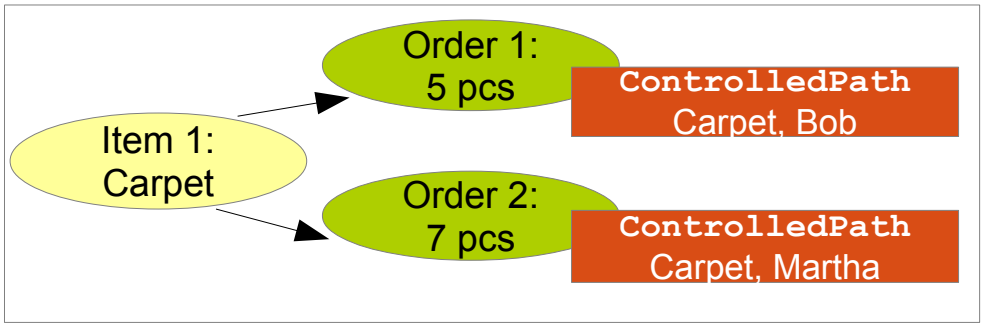
©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

If the data would be shown in a tree-structured view in the form: „Show all your connected elements", the carpet will occur two times in the structure with all its orders shown below it.



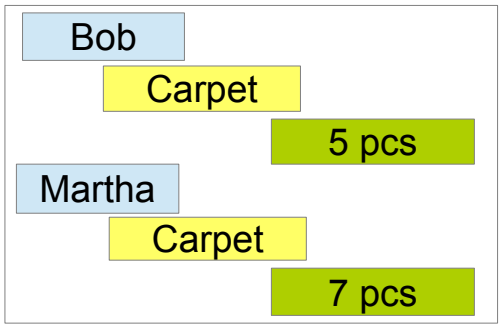
What is needed is an overview of the carpets ordered per salesman.

Therefore the `cPath()` function has to be used.



Within the Meta Model the type „Order" gets a special attribute called a **ControlledPath**. This path is then able to follow the sequence of elements above in a tree structure back to its roots. When a new order comes in this sequence it will be recorded in the ControlledPath.

In the view of the tree-structure this recorded sequence will be compared to all current branch elements above and only the orders according to „its" salesmen will be displayed.



5.3.4.4 Parametric views

Parametric views are based on view configurations that work with global filters. They are used to filter similar data with several parameters.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

The relating function is named:

`filter(categoryName)`

This function is a place-holder for special filters, which are defined by an administrator and have a special category-name, see section 8.5. While opening a view, that contains such filter() functions a dialog appears from which to choose the concrete filter, see section 7.7.

5.4 Advanced functions

5.4.1 Numeric functions

`sum(prop1,prop2,…)`

This sums up all the numeric values defined in the property-paths.

All numbers will be handled as numeric values, all words and other elements will be set to zero (0).

It has to be remembered that as long as integer data types will be summed up, the result will be an integer too. If there are decimals, the result will be delivered in decimal form even if the sum is integer.

`treeSum(prop)`

This function works within tree-structures where the related line elements below equals iteratively the line elements above.

The value of the property will be multiplied with the corresponding value above.

Here is an example of an iterative structure of quantity values:

```
[treeSum(quantity)]
```

The following cascaded result will be delivered:

element-name	*quantity*	*treeSum(quantity)*
- element 1	2	2
- - sub-element 2	3	6
- - - sub-element 3	4	24

`size(prop)`

This delivers the quantity of elements in a parameter-list.

This function works for column-definition or within variables.

Because this function does not deliver a list of elements, it can't be used for line definitions.

This is an example for a **column definition** in type Item Order:

```
size(wine)
```


©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

For the underlying „Item Order" element the quantity of connected „Wine" elements will be provided.

Because this is a column definition no path of a start type is needed. This is given in context of the column definition in the corresponding type line.

Without a property in front of this function the filter will directly display the underlying elements of the row.

Other numeric functions include: `average()`,`cube()`,`divisible()`,`max()`,`min()`,`mod()`, `random()`,`round()`,`sqrt()` and `square()`.

5.4.2 Date-Time functions

Date and time functions include: `date()`,`today()`,`week()` and `year()`.

5.4.3 Other

Other functions include: `ascending()`,`color()`,`descending()`,`if()`,`level()`,`not()`, `pattern()`,`pk()`,`single()`,`substring()`,`toBoolean()`,`toDate()`,`toMonth()`, `toNumber()`,`toString()` and `toYear()`.

For specific details of the functions mentioned in this chapter please refer to the Java-Tutorial at http://docs.oracle.com/javase/tutorial/java/

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

6 View Configuration

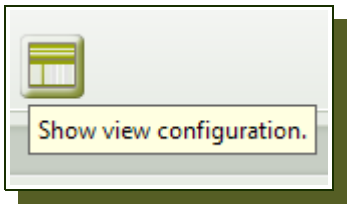
6.1 Introduction

Within the view configuration the view-display of data will be defined. The basic principle in JACAMAR is the separation of the background tables from the display of the data in the views. Each view or subview needs to be configured before any data can be viewed.

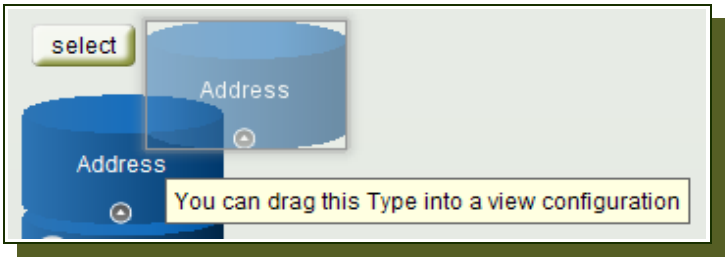
Intelligent navigation concepts support the administrator by defining line rules and column contents:

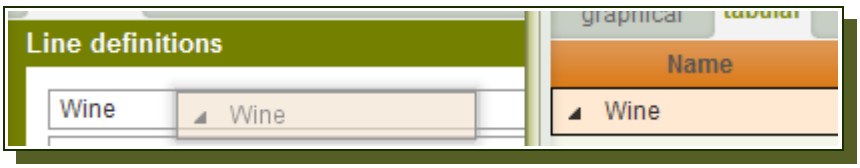
- Within a view several lines can be defined by navigation paths for special criteria
- It is definable which columns to display
- Background colour of lines enables easy viewing

View configuration is used, when you have a view open, by clicking on the 'table' symbol in the top right of the JACAMAR session. Only administrators have these rights unless the permission is given to other roles in user administration, see chapter 9.



The Meta Model types, properties and relations can be dragged and dropped (DnD) into view configuration for practical and quick navigation. In the GMM „drag" mode simply click and drag the component into the required line or column definition. Within the TMM view hover the cursor over the bottom line of a component and when the cursor changes to the directional symbol drag and drop it into view configuration.





©2015 KATLA GmbH | Immermannstraße 28 | D - 39108 Magdeburg | www.katla-gmbh.de

6.2 View configuration dialogue

6.2.1 Line and column defintions

The basis of view configuration is the definition of Line and Column Rules, which is assisted by the navigation language and variables.

In JACAMAR data will be displayed in complex **tree-structures**. Therefore certain steps are needed:

- Line defintions: it has to be defined in the first line definition which is the base object type (-root rule) for the lines of a view (or type[`FilterStatement`])
- Additional line defintions: it has to be defined which **related-lines** (-**relation rules**) have to be displayed for cascaded tree-structure views.
- Column definitions: for every line type defined it has to be decided which information shall be displayed within the columns of the lines in the view. The column definition titles are labelled according to the origin type of the line.

An example of a typical cascaded view configuration dialogue:

Please note that for cascaded branch lines the line definitions must traverse from the previous line type for successful navigation path evaluation.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

6.2.2 Line definition

As mentioned previously, it has to be defined on which base object type (-root rule) the view navigates from. The root rule describes the root-type of the tree. If only the root-rule is defined with column defintions a flat table (non-branched list without tree-structure) will be generated.

For a branched list further line definitions (relation rules) need to be entered.

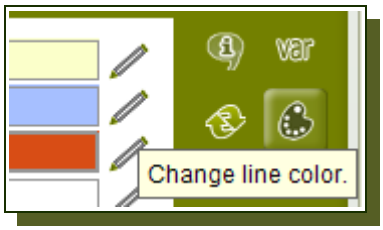
Relation rules define related-lines that shall be displayed beyond the junction. For every root-type element all related-rules will be checked in line with the navigation-path. The related-type elements will be recognised beyond the root-type within the tree. The configuration order of the line definitions determines the order and appearance of the related lines in the view.

The screenshot above only defines „Wine.item Order" and „Item Order.invoice" as related lines and therefore for every wine all relating item order elements then all relating invoice elements will be recognised in their corresponding lines. (Column definitions and line colour have been included for display purposes).

Using the DND function from the Meta Model window you can insert the root rule and subsequent relation rules easily. The manual input of line definitions is explained in section 6.2.4.

Once the lines have been defined you need to update the columns to reflect the view configuration of the lines. This is achieved by clicking the green arrows symbol to the right of the line definitons and the column defintion lines will be updated.

Each line, as shown above, can also be displayed in a different colour within the view by highlighting a line defintion and clicking on the artist symbol to select a desired colour. This is particularly useful to identify different lines in a cascaded view.



The i symbol opens a dialogue where you can leave a description about the view and details about the associated responsibilty.

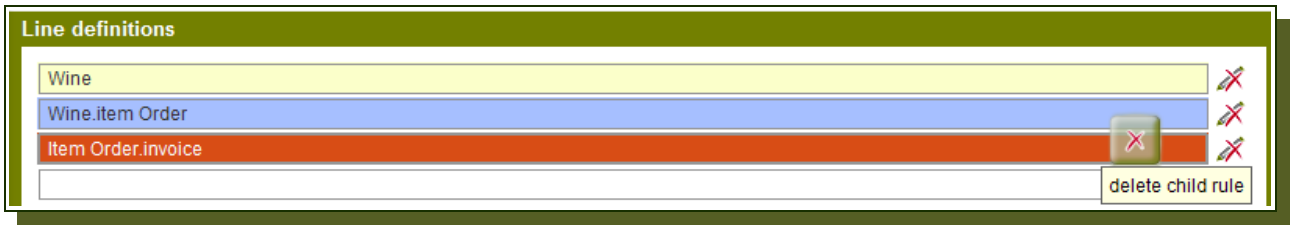
The var symbol opens the variables dialog for the view, where you can create, edit and delete variables. Saved variables can then be selected as a shortcut in the manual input of navigation paths for the view, see section 6.2.4.

Using double click, you give the variable a name, select the type that corresponds to the view root type, and enter a navigation path in the corresponding fields. Editing an existing variable is done the same way by replacing the existing variable parameters, and to delete a variable simply RMB on the variable and click on the red cross.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

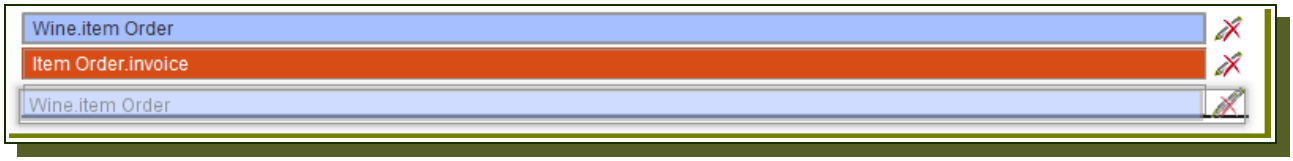
The pencil symbol at the end of each line defintion is the edit on/off function for each defined line. If the administrator of the view doesn't want users to be able to delete or add a line in the view for one or more of the defined lines the edit off can be selected for any line defintion.

This only applies to adding a line or deleting a line in a view and not entering or amending data.



To delete a related line from configuration simply RMB on the line and then on the red cross to delete the related line from the view configuration.

To change the configuration sequence of the related lines for the view select the related line definition you wish to move then mouse cursor on the bottom and when the directional symbol appears drag it to the preferred position in the view. A black will appear to indicate where the line definition will be inserted. Remember to update columns after any changes to the line definitions.



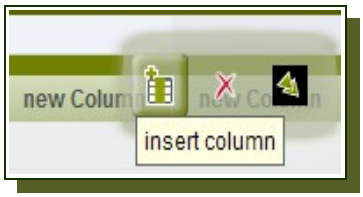
6.2.3 Column definition

Several object types can be defined in the line definition and for each type a definition of column contents has to be given. The column definitions will appear. are labelled accordingly, and they each represent the type or related type from the line definitions.

Using DND from the TMM or GMM you can place your properties in the appropriate columns to display the data in your views. The column headings will default to the dropped property however by using double click on the the column heading they can be renamed at any time.

To insert or delete a column, or to change a column to be expandable, simply RMB on the column heading and select the option from the icons. By default the first column is expandable when a relation rule exists.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de



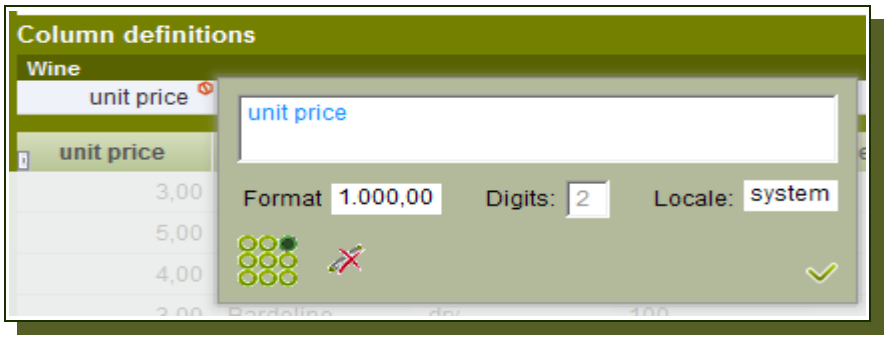
To change the sequence of the columns simply click and hold the column heading and drag it to the preferred horizontal position in the view. The column width can also be changed when the cursor is at the end of the column heading, it changes to a double arrow, and can be dragged left or right to change the column width.

The display of the content in each column can also be formatted in view configuration by using double click on the defined column as follows:

Alignment Grid– column content can be aligned to the left, centrally or to the right by selecting the appropriate circle in the grid.

Pencil symbol Edit on/off – Edit content can be switched off for a column by clicking on the pencil symbol for a red cross to appear.

This applies to all users who open the view and will be unable to edit the content within that column.



The column definition example above has been aligned to the right, and edit column has been switched off. Property data types integer, decimal and date also provide formatting options that can be changed for the column view-display.

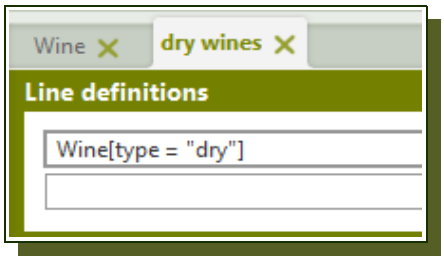
6.2.4 Manual input of navigation paths for line and column definitions

The previous two sections describe using drag and drop from the Meta Model to enter the line and column defintions, however they can be entered manually when you are familiar with the Meta Model components and the navigation paths and functions.

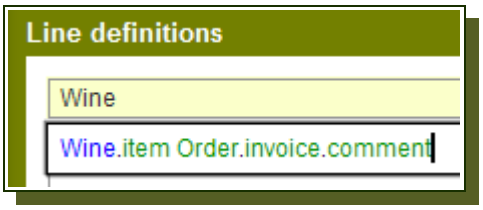
When entering the definitions or using double click in the line definition or column definition fields, the system generates a drop down list of available options to be used. Manually typing a navigation path continues to refresh the drop down list available and it is important to include the . (dot) for a new segment so the available options refresh accordingly.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

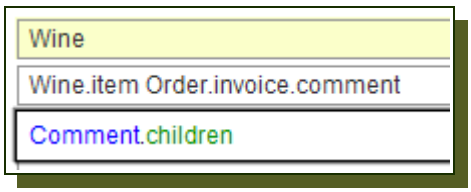
- 1st line definition always offers Meta Model object types (dark blue font), because a base object type (-root rule) has to be defined for the lines of a view (or type[`FilterStatement`]).



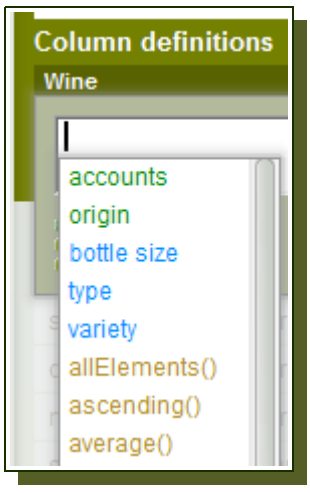
- 2nd or subsequent line defintions can also be added to show cascaded lines for related types or filters. Line definitions must end in a type or a filter function of a type. And to show cascaded lines you must traverse from the previous line type to a related type. The example on the right starts with the previous line type Wine, then passes across two related types, Item Order and Invoice, before ending with type Comment.



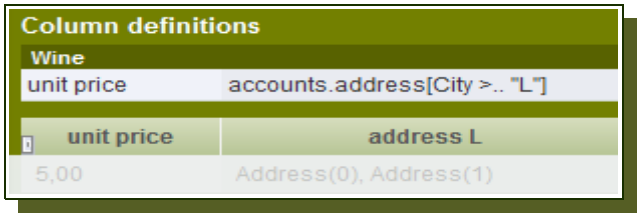
- If however you want to show a child/parent cascade then first you need to navigate to the child type and in the next line definiton traverse to the related children, as shown on the right.



- column definitions correspond to the navigation path for the line and the drop down offers relations (green font) and properties (light blue font), and the functions (light orange font) as detailed in section 5.3.4. If a relation is selected then a . (dot) must be used to end the path with a property (light blue font) from the list. When passing across related types the properties listed will reflect the related type.

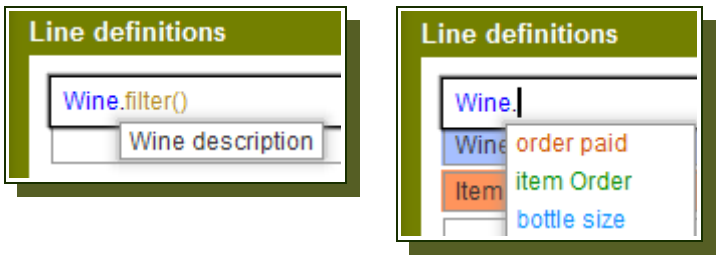


- When filter statements are used in line defintions and column definitions they are a fixed part of the view, example below shows a path from line Wine to account addresses starting with „L".

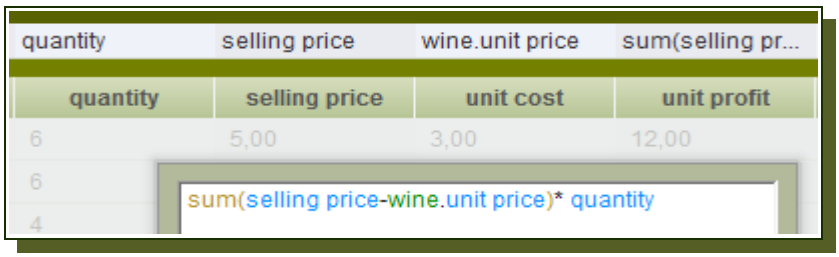


©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

- Parametric filters, section 8.5 and view variables (dark orange font), section 6.2.2, can also be included in the navigation path in line definitions and column defintions.



- As detailed in chapter 5, filter statements and functions can be configured in line and column definitions, see the example below.



6.3 Sub views

6.3.1 Definiton

For every view a so-called sub view can be defined additionally. They display more details regarding the chosen line. The „white lines" symbol at the end of a column definition line opens the sub view dialog.

Such a sub view is recommended while working with a lot of related data, to get an overview or a breakdown of specific secondary information. (This information could also be shown directly in the main view but limited by common column names and width)

6.3.2 Configuration of sub views

After clicking on the white lines symbol and the dialog opens you must double click on the `<Dbl-Click for new sub view>` option and give your sub view a name. You can have multiple sub views for each line in your main view. To rename or delete a sub view simply RMB on the sub view and select the appropriate option.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

After entering the name you can double click on it to open the configuration for your sub view. The sub view base type is assumed from the navigation path in the line definition selected.

Base element: shows the element representation the sub view refers to. The sub view only will appear within the context menu of the RMB when choosing a line including elements of this type.

The synchronise „on" option allows a user to move within a view and the sub view content will update with each new selected area in the view. When sychronise is „off" then the sub view data stays the same and further sub views can be opened and can be compared simultaneously. See 7.4 for usage of sub views.

The sub view is then configured, remembering that the root rule within a sub view is not an absolute but a relative path without a type-name in front. The basic-type is defined already in the main view. The column definition in sub views operates the same way as in the main view configuration.

6.4 View Configuration – Wine Supply example

6.4.1 Introduction

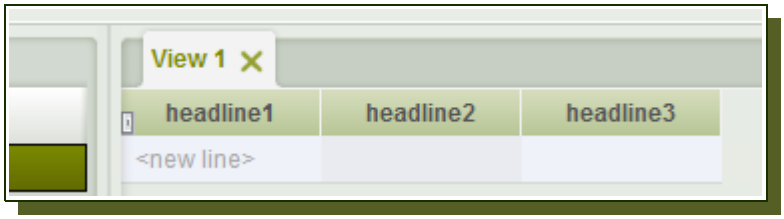
Following on from the completion of the Wine Supply Meta Model in section 4.5, and utilising the model navigation language described in chapter 5, we will now explain how to configure the views in preparation for the moving of data and data correlation from the wine supply data pool.

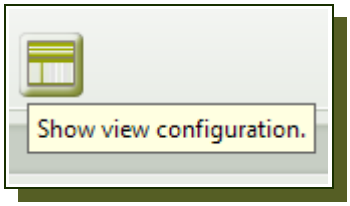
©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

Every new repository opens with a default views section and view1, however more view sections and views can be added as outlined in chapter 7.

6.4.2 Base view configuration

Select View1 from the tab across the top or open it by using double click on View1 in the Views section. Everytime a view or raw table is opened they will appear as a tab and can be closed by clicking on the corresponding cross.

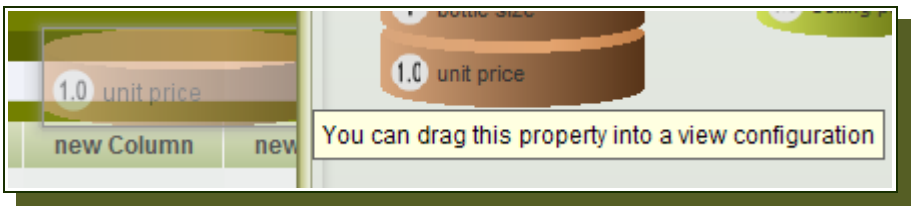




Once the view is open you need to show the view configuration (as above) in order to define the lines, columns and the column headlines.

Every new repository displays view configuration for view1 with a default root rule line definition (first type saved in the Meta Model) and three default column defintions based on the first three props within that type, and those headlines can be labeled accordingly by using double click and renaming them.

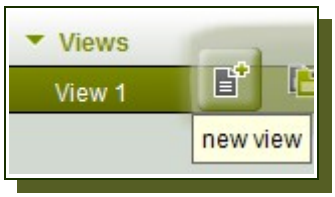
If required the default line and column definitions can also be changed by using DND in the MM drag mode.



Where possible in the configuration of our views in this section we will use the drag mode in the graphical Meta Model, as above, however the Tabular Meta Model in drag mode and the manual input of line and column definitions is also possible, as explained in 6.2.4, when you are familiar with the structure of your Meta Model and navigation language.

View configuration and a view needs to be created for each Meta Model type so as to record the base data. In our example type „Wine" has only four properties: variety, description, bottle size and unit price, and once the property unit price has been dragged across and the headlines have been updated the changes can be saved and view configuration closed.

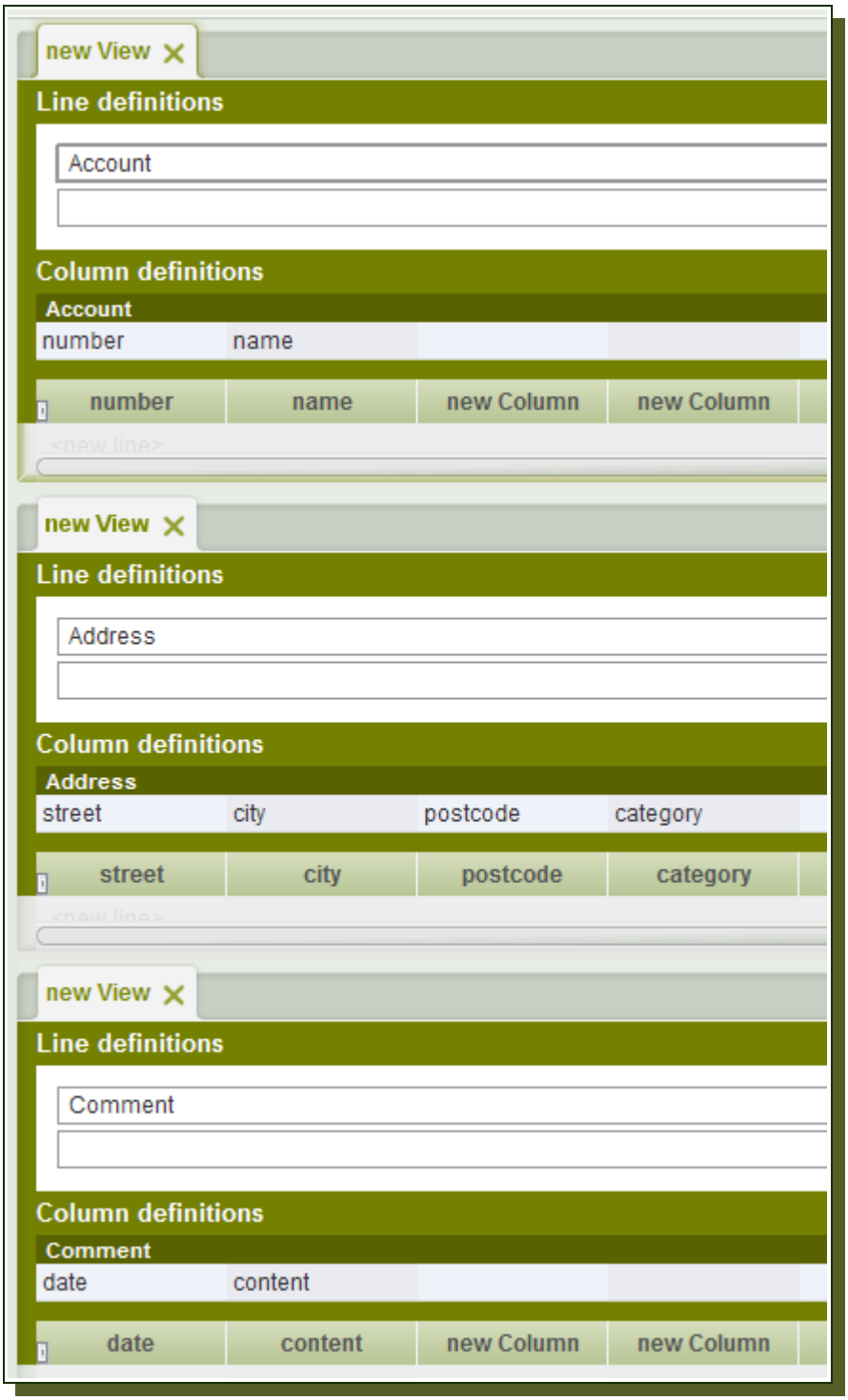
The Wine Supply Meta Model has a further five base types: Item Order, Invoice, Account, Address and Comment that also need to be configured in views. More details on views are explained in chapter 7, however for ease right click on View1 and add five more views to the view section.



©2015 KATLA GmbH | Immermannstraße 28 | D - 39108 Magdeburg | www.katla-gmbh.de

Open each of the new views separately and in view configuration drag and drop each type into the line definition, remembering to update columns, and then the corresponding properties into the column defintions for each view, see example on the right for the view configuration of types: Account, Address and Comment.

It is now advisable to jump to moving data to base views, section 7.8.1 before configuring the relation data views.



©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

6.4.3 Related view configuration

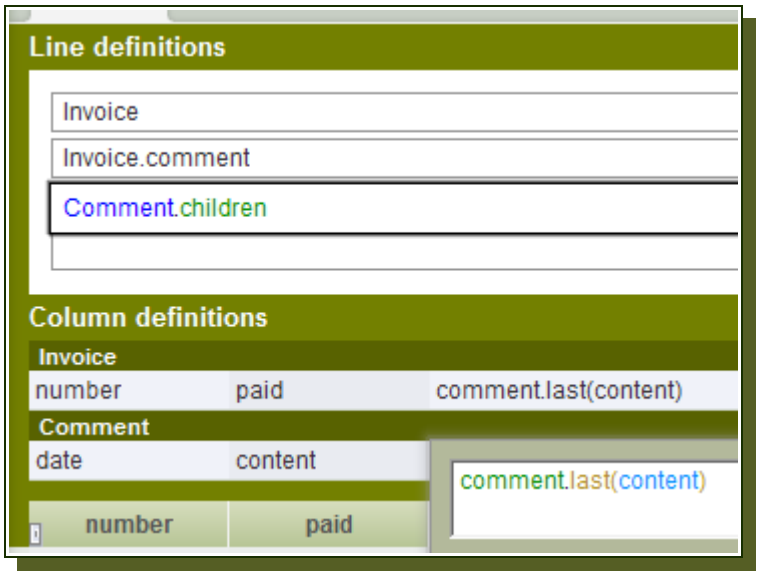
As identified in our business model we have a few relations between the business areas (types) and now we have to configure the views to correlate the related elements. It will be useful to keep the Meta Model window open as a reference for the next steps and for DnD functionality.

We will start with type Item Order and it's relation with the Wine and Invoice elements, so open the Item Order view and show view configuration. Based on our Wine Supply we can see that each item order is related to a wine variety and an invoice number (as it is referenced in the cell i.e. =Wine!A3 for Amarone).

Using DnD from the MM place the wine variety property and invoice number property into the next two available columns, amending some of the headlines for clarity, as shown. And if you wish to add a formula into the next available column to calculate the total selling price for each item order this could be done manually, saving your changes.

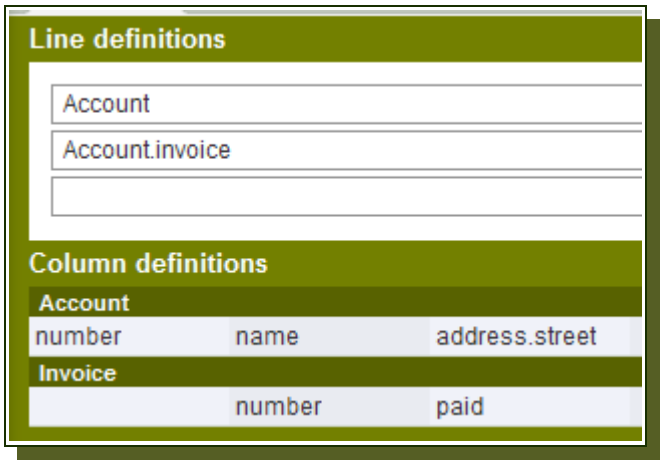
Now we need to correlate the relation between an invoice and its comments. Open the invoice view and in view configuration manually input a second line definition traversing from the root rule Invoice to the relation rule Comment, remembering to use a . (dot) between segments, and update your columns. In the new column definition DnD the comment properties into the first two columns.

We also need to diplay the children in this view so comments within comments are possible, therefore in a third line defintion manually add the comment.children path. (As an extra if you wish to view the last comment of an invoice you could use the list function `last()` manually input on the invoice line in the third column as shown).



©2015 KATLA GmbH | Immermannstraße 28 | D - 39108 Magdeburg | www.katla-gmbh.de

The last view we need to configure is the relation between an account and its invoices and addresses so open the account view and show view configuration. Here we need to relate the key elements from the other types in the view for the most appropriate display. At first you can drag the address street property to the next available column in the account line. Then to display the invoices as a cascaded view you need to manually add a related line (2nd line definition) traversing from Account type to Invoice type and update columns. Then in the second and third columns, for display purposes, of the new Invoice line you can drag across the invoice number and paid properties.



Now the views are configured for the relations to be moved across you can proceed to section 7.8.2 and complete the views with the related elements in line with the Wine Supply data pool.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

7 Views and tables

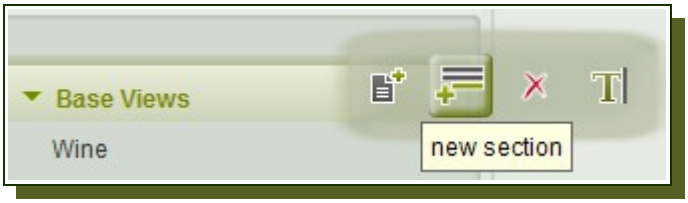
7.1 Introduction

As administrator you can access all the tables and views in a repository, except private views which are user specific.

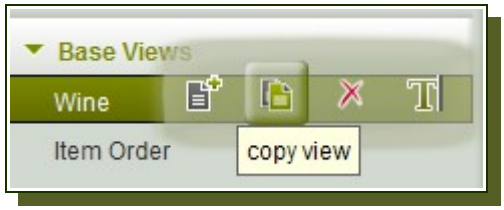
This section is aimed at an administrator working on moving data to a new repository and moving data from a spreadsheet or database to JACAMAR views, with reference to the Wine Supply example in section 7.8.

7.2 Managing View Explorer

Views that are created and saved will be listed on the left in the view explorer. The first section (Views) and the first view (View1) are system generated when opening a repository for the first time. Using RMB on a section you can add a new view, add a new section, delete the section or rename the section.



Using RMB on an individual view you can add a new view, copy the view, delete the view and rename the view. A view section cannot be deleted if it contains any views so the list of views must be selected and deleted first to allow the corresponding view section to be deleted.



You can also change the order of the view sections or location of the views by selecting the one you wish to move and cursor on the bottom line and, when the directional symbol appears, drag and drop it to the preferred location.

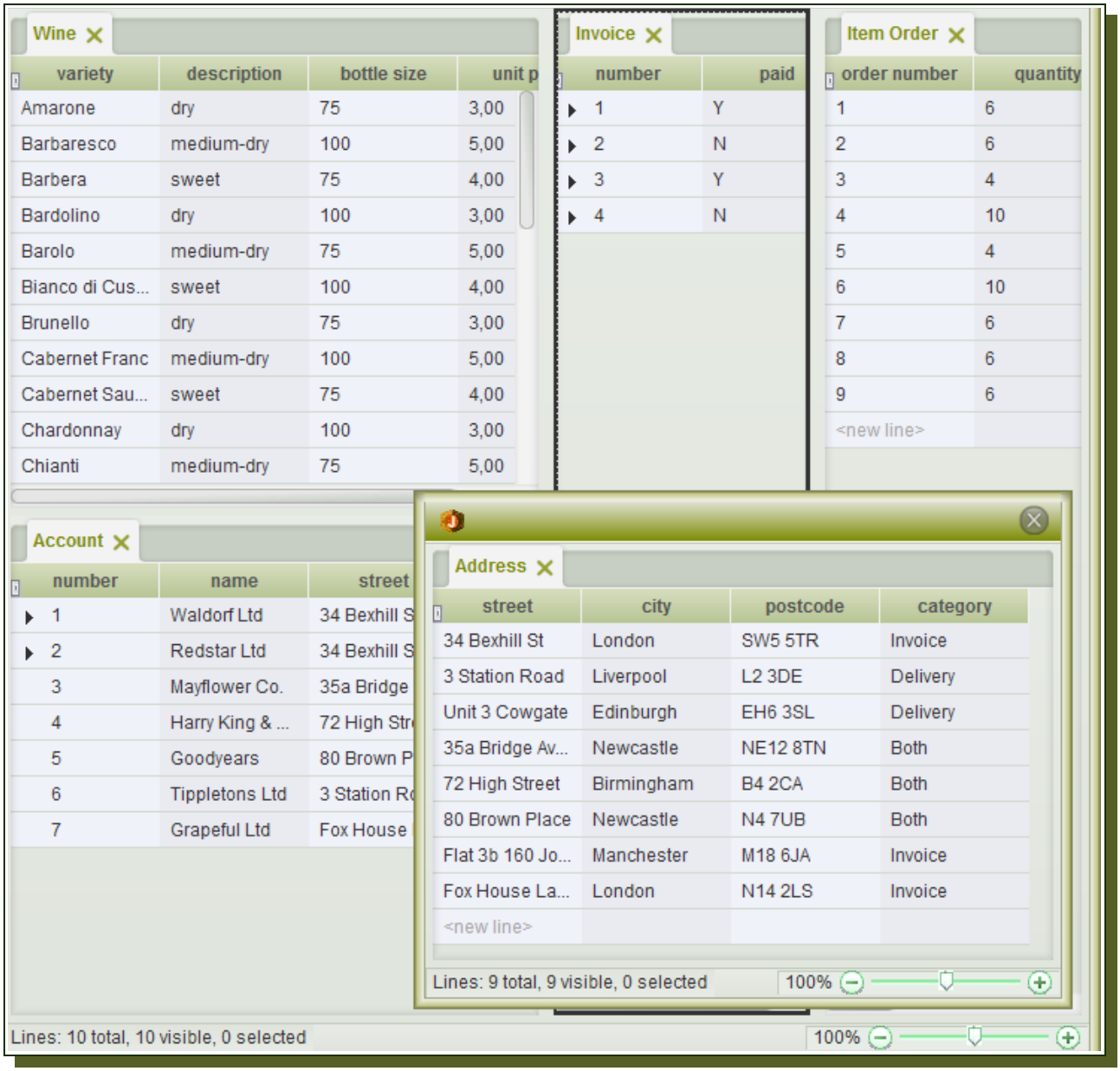
It is recommended to create a base view section with individual views at the beginning for you to move the base data across when you are creating a new repository.

7.3 Displaying multiple views

Each view opens in the main window and appears as a tab across the top. When multiple views are open it is possible to display them all within the main view or as secondary windows. Selecting and dragging the tab from across the top the user can tile the views vertically, horizontally and externally to display the tables to their needs. A black frame appears and indicates where the dragged view will be inserted. To drag a secondary window view back into the main window simply click and drag the tab to the position you want.

Views can be closed individually by clicking on the cross at the top right of the view or using the FILE MENU → CLOSE ALL to close all open views.

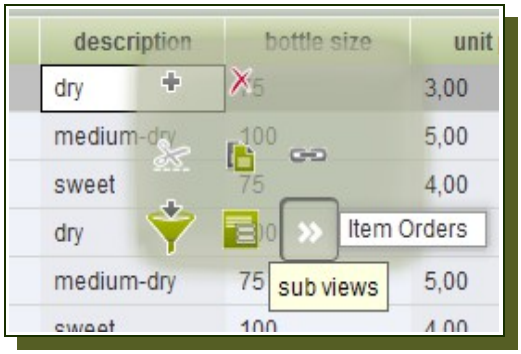
©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de



7.4 Usage of sub views

A sub view can be opened using the RMB when on an element within a view. The list of sub views available will appear from which to be selected, however if no sub view is available the white arrows symbol will not appear. Sub views open as a secondary window and can be closed using the top right cross in the corner.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

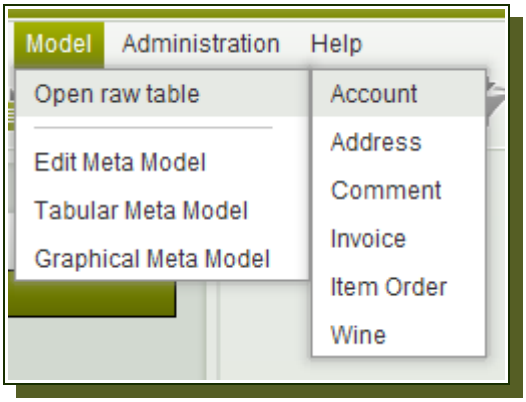


As default the synchronise check-box is activated. That makes the sub view update its information automatically when clicking into another element of the main view.

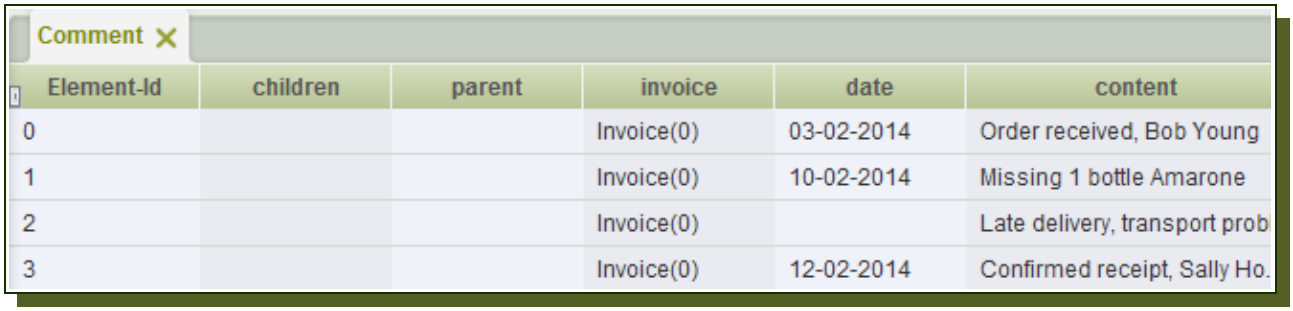
If the check-box is deactivated. Several sub view windows can be opened parallel and their information can be compared.

7.5 Open raw tables

As administrator of the repository you also have access to the raw data tables which are located in the MODEL-MENU → OPEN RAW TABLE option. Clicking on the open raw table option provides a list of all the types in your Meta Model and by selecting a type the raw table is opened as a view displaying all the relations and contents of the type.



The raw tables are displayed with their unique element-Id, which is system generated and represents the data records belonging to the type. Each element will display their related children, parents and elements where relations exists in the Meta Model. Each element of course also lists the data belonging to the properties within the raw table type, as shown below.



©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

The view options within the raw table views are exactly the same as in normal views for an administrator. Raw tables will automatically close if an admistrator changes to another role or if a changed is saved to the Meta Model.

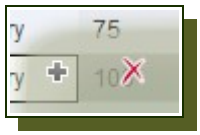
7.6 View options

7.6.1 Introduction

As administrator you have additional functionality within a view, however these permissions can be given to other roles in user administration see chapter 9.

7.6.2 Flat table views

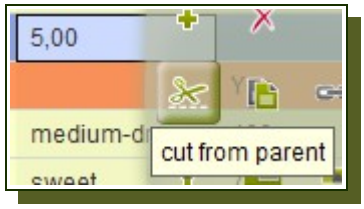
RMB into a space or on a new line and you can add a new element to the view.



RMB on a record in a line and a list of options will appear. The availability of those options will depend on which record of the element you have highlighted and what user permissions you have. Adding, as explained previously, and deleting an element are available to an adminstrator.

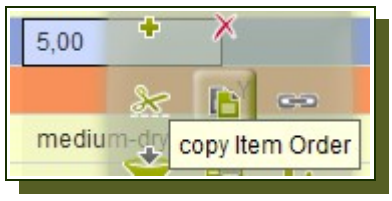
Caution needs to be taken when deleting an element as all relations and data for the element selected are deleted.

Where a relation exists an administrator has the ability to cut the element from the related line and position it elsewhere in the view.



Within the relations there is an important difference between cutting and deleting. If you cut a relation you don't delete any elements, even if you don't see them in your view anymore.

The copy option allows all users to copy the element or related element to another part of the view.



When cutting or copying an element it's important to know that the elements children will also be included in this action.

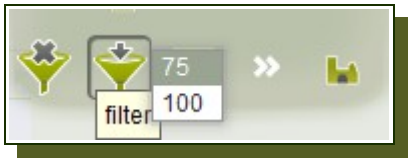
Once an element is cut or copied the link becomes available in the RMB options menu and offers three methods of how you can link the cut/copied element.

©2015 KATLA GmbH | Immermannstraße 28 | D - 39108 Magdeburg | www.katla-gmbh.de

You need to first choose a location where you want the cut/copied element to be inserted and RMB on the location before selecting an option.

- link – is used to place the element below the location selected as a sub-element (branch).
- link behind – is used to place the element on the same branch level as the location selected.
- replace – is used to replace the selected location with the cut/copied element.

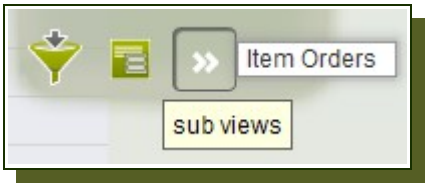
Clear column filter will appear when a column filter is applied and when selected the column filter will be removed.



Column filter gives the user the option to apply a column filter and a list of column values from to which to choose a filter.

The green square symbol provides all users with all the property contents of the element, including relations.

If sub views are connected to the view the white arrows symbol appears and a list of sub views are available to choose from, that relate to the selected element as detailed in section 7.4.



The green castle symbol allows all users to view the history of the selected field of the element and provides details of the changes, corresponding dates and times and which user entered the changes.

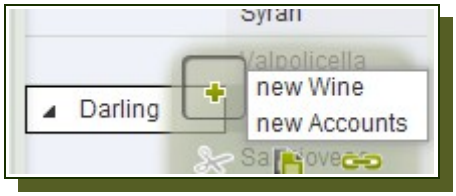
Using the mouse it is also possible to change the column width and change the column sequence, by holding down and dragging, to the desired display. By clicking on the column headings you can also sort the column contents into ascending, descending or rever to the original display.

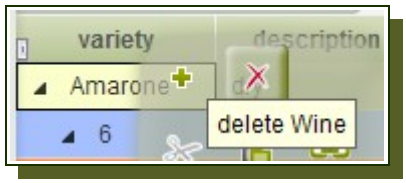
The RMB cut and copy function in a view is for the element and it is not to be confused with the standard copy and paste function in the edit menu and toolbar. The edit menu and toolbar copy function is used to copy data, whether a single value or a range of values, to another location. The edit menu and toolbar paste is used to paste data from a copied source into a view.

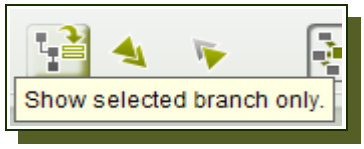
©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

7.6.3 Cascaded views

In a cascaded view the RMB options are the same as in a flat table view however their function differs due to the branches and related lines which are defined. When adding, deleting and cutting in a cascaded view the system will assist you with your options.



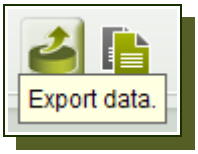




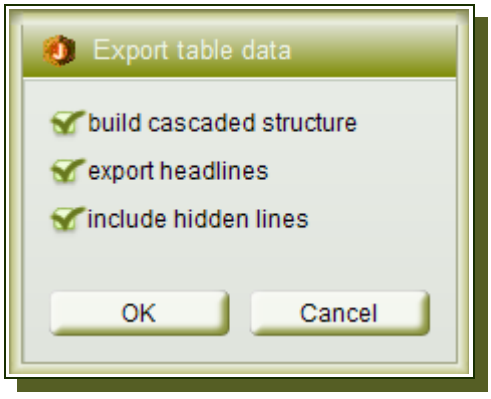
Pasting to cascaded views can be problematic so you have to be careful to ensure that the pasted data is entered in the appropriate fields within the branches. If multiple branches are displayed and empty lines within the branches are not added the pasted data carries forward into the next branch in the view. It is recommended that you apply a filter to the cascaded view so that the specific area is displayed for you to paste the appropriate data., therefore the workaround is to show the selected branch only, add a new line and paste the data into the branch.

7.6.4 Export data

When you have a view open it is possible to export the data from the view into clipboard by selecting OPERATION → EXPORT DATA from the menu bar or the export data icon in the toolbar.



When selected the export table data dialog appears and gives you options for how you want the exported data to appear:



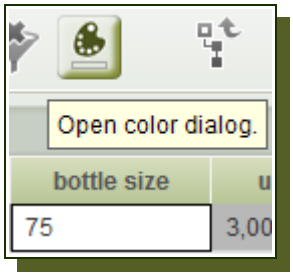
- build cascaded structure – when selected will show the exported data as a cascaded structure in the display (If not selected it will appear as a flat table).
- export headlines – when selected will include the column headlines in the exported table.
- include hidden lines – when selected will show all of the collapsed branches in the exported table.

When clicking on ok a message will appear confiming the data has been copied to the clipboard.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

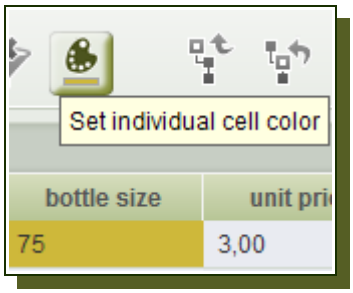
7.6.5 Individual cell colour

You can apply a colour to a cell, or multiple cells, in a view by using the artist symbol, set individual cell colour, from the toolbar.



The icon has two functions and the mouse cursor changes, and a message appears, to signify which function will apply. First you must select the area in the view you wish to change and then when the mouse cursor changes to the hand symbol on the bottom of the icon you need to open the colour dialog.

When the select colour dialog opens there are several ways to choose your preferred colour: select one of the options provided in the grid, use the mouse on the larger square to specify a shade within a colour range, or use the sliding scale on the right. Once the desired colour is found, as displayed in the rectangle, click ok to apply the colour to your selected area. Once applied the icon will appear with your last selected colour and further areas in the view can be selected and changed to this same colour by clicking on the artist symbol without opening the dialog.

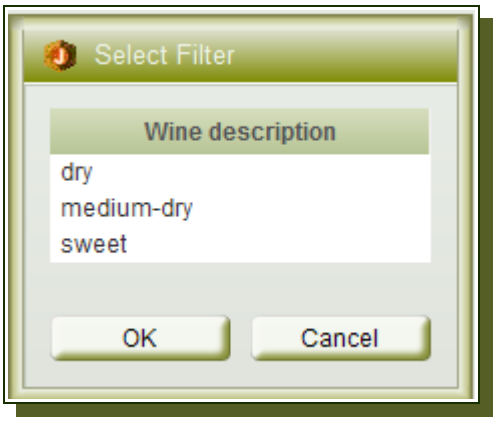


If a user has some specific colours that they use regularly and wish to save it is also possible to define them in the select colour dialog by dragging them from the rectangle and placing them in the user defined colours section below.

©2015 KATLA GmbH | Immermannstraße 28 | D - 39108 Magdeburg | www.katla-gmbh.de

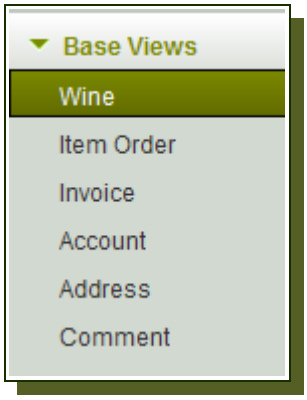
7.7 Usage of parametric filtered views

After a parametric filter has been configured to a view when you open the view a prompt will appear asking you to select a predefined parameter to view. The parameter selected will display a view based on the parameter filter you have selected.



7.8 Moving data into views – Wine Supply example

7.8.1 Base view data



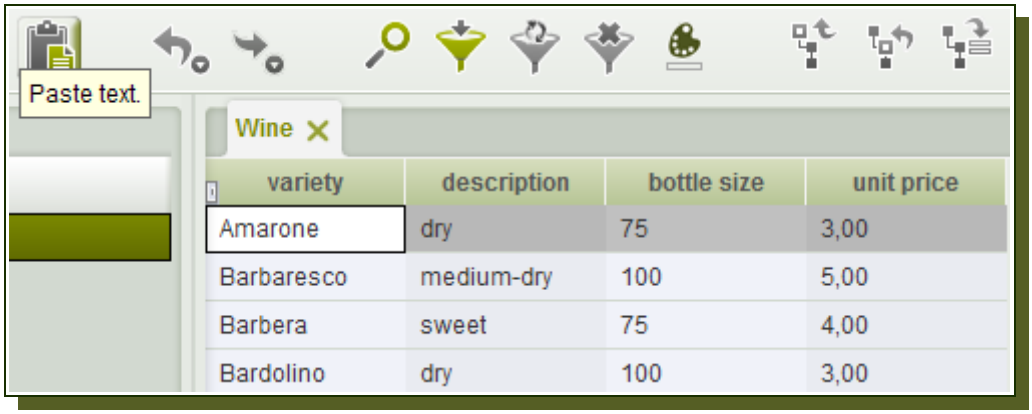
Following on from section 6.4.2 where you have configured the base views you must now use the Wine Supply sample spreadsheet to begin the moving of data into your views.

It would be useful at this stage to name your view section and views, as explained in 7.2, for example: „Views" Section could be labeled Base Views, and the individual views could be labelled to correspond with the Meta Model design, as shown on the right.

As the view configuration for the views is complete you can open the views, and hide view configuration if it is shown. As mentioned before, the spreadsheet data must be prepared (format and layout) prior to moving and the data type and format of the properties in the Meta Model must represent the data to be moved into it (integer, decimal etc.).

Starting with the wine view and using the wine worksheet in the spreadsheet copy the cell range A3:D32 (Amarone to the last unit price listed) and paste the data into the view from the start point cell1 under variety. When copying data remember to copy only the data and not the headings, and when pasting large cell ranges it is important to make sure the column sequence is the same for both so the data is pasted correctly.

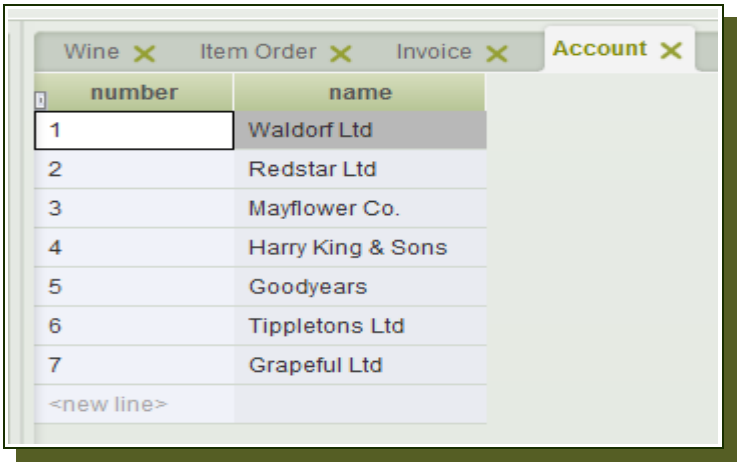
©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de



Next we will move the basic order data and as we can see from the worksheet that each line order has an associated wine variety, this data however is related and will be copied across in section 7.8.2. Therefore copy cell range A4:C12 only and move it across to the order view, see below.

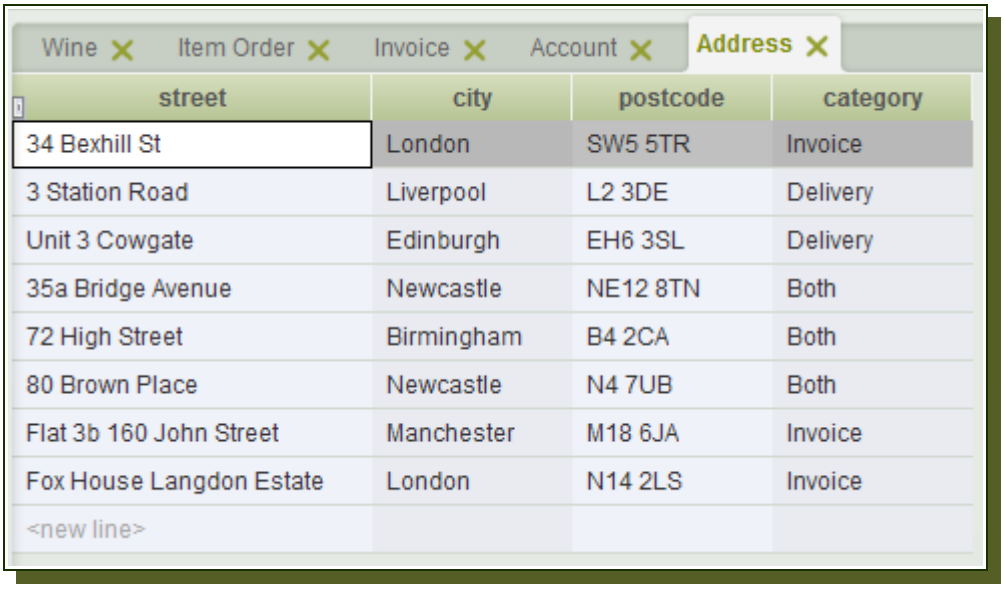
The intial data for the invoice view is quite simple and can be input manually: invoice numbers 1-4 and invoices 1 and 3 have been paid as detailed in the Acc1 and Acc2 worksheets, as shown below

In the Accounts worksheet we can see that there are duplicated account numbers and account names because some of the accounts have multiple addresses. We only need to copy the individual account numbers and names here to avoid duplication and the relations will be correlated later.



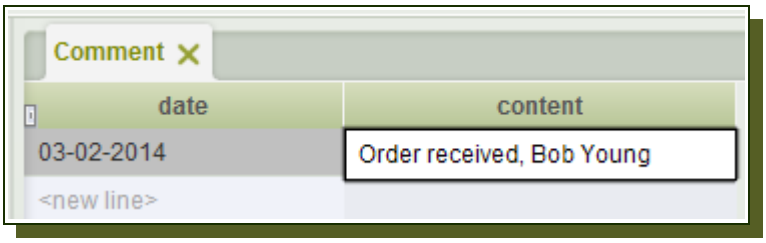
©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

Next we can move across the address data into our address view and similarly we have to group the data first if there are any identical records to avoid duplication. Any differences between records are to be listed separately e.g. if an address is an invoice address for one company but a delivery address for a different company, the address would be listed twice with differing categories.



street	city	postcode	category
34 Bexhill St	London	SW5 5TR	Invoice
3 Station Road	Liverpool	L2 3DE	Delivery
Unit 3 Cowgate	Edinburgh	EH6 3SL	Delivery
35a Bridge Avenue	Newcastle	NE12 8TN	Both
72 High Street	Birmingham	B4 2CA	Both
80 Brown Place	Newcastle	N4 7UB	Both
Flat 3b 160 John Street	Manchester	M18 6JA	Invoice
Fox House Langdon Estate	London	N14 2LS	Invoice
<new line>			

And finally we can move across our comment data , however as our comments are related to specific invoices we will only enter the first comment record here and add the rest later, always remember to save your changes regularly. Each base type view needs at least one record (element) to start with.



date	content
03-02-2014	Order received, Bob Young
<new line>	

Now you should return to section 6.4.3 to configure the views for the related data before completing the next section.

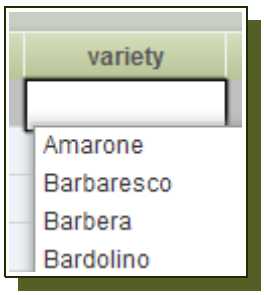
7.8.2 Related view data

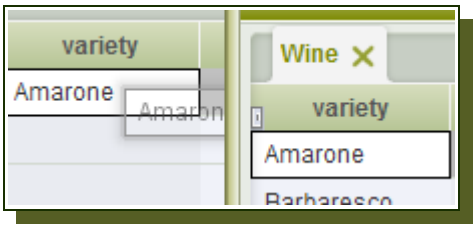
In the same order as we configured the related views, we will start with the Item Order view. Open the view and hide view configuration if it is open. There are different ways of moving the related elements into the view:

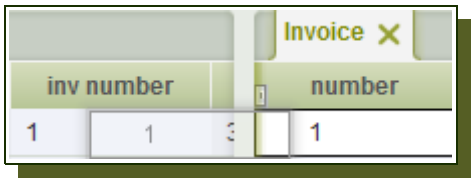
- double click in an empty field and choose from the list provided which is generated from the background tables,
- open the wine view and invoice view and either tile them vertically or as secondary windows and DnD the corresponding element directly from the view, using the directional cursor on the bottom line of an element,

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

- paste the varieties and invoice numbers across from the orders worksheet in the Wine Supply spreadsheet.







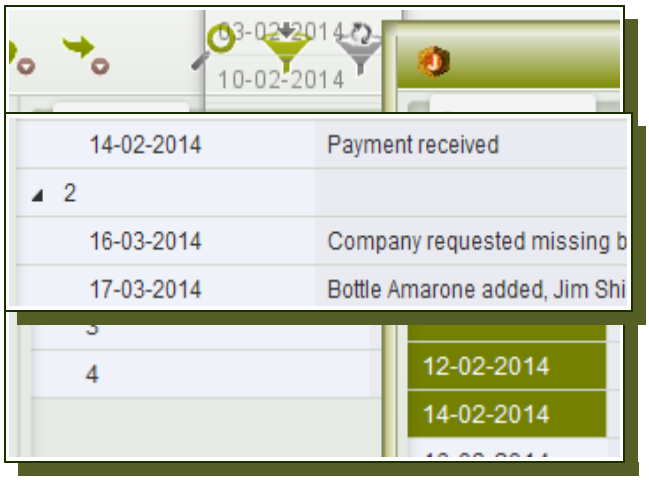
Once complete, and after referencing the source spreadsheet for accuracy, you will have an up to date order list with related wine varieties and invoice numbers, as partially shown below.

Item Order

order number	quantity	selling price	variety	inv number	order total
1	6	5,00	Amarone	1	30,00
2	6	6,00	Brunello	1	36,00
3	4	6,00	Cabernet Franc	2	24,00

Now open the invoice and comment views, here we can first complete the comments view by pasting across the remaining comments. Carefully copying the combined dates and comments from the four invoices separately from the Acc1 and Acc2 worksheets into the comment table, including the empty fields where no date is present and save the changes.

Display the views as tiled or in separate windows, as before, and this time we will highlight a group of comment elements and DND them onto their corresponding invoice. This can be done holding down cntrl and selecting each individual element or using shift and cntrl for a range and when the directional cursor appears the elements can be moved and dropped directly on their corresponding invoice.

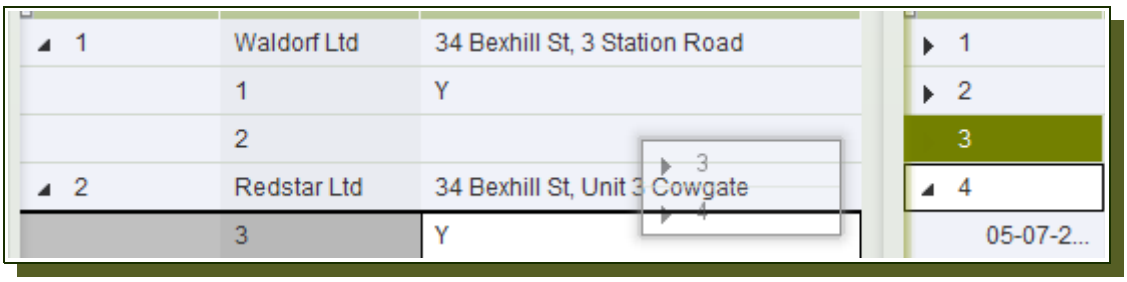


Each time a list of comment elements is moved to an invoice a cascaded branch appears under the invoice in line with our view configuration line defintion, and once each invoice has its associated comments, save the changes and close the view.

The last related view we need to complete is the account view where we also have a cascaded line defintion. Open the account view and the address and invoice views and display them appropriately for use. Again, using the

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

spreadsheet as the reference, DnD the corresponding address elements to their related account, where an account has many addresses they will be comma separated, and DnD the invoice elements to their corresponding account.



All of the views now should represent all of the related elements that you have in your spreadsheet.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

8 Filters

8.1 Introduction

Section 5.3 describes the filter navigation for a list of results in a view. When these filter statements are configured in a line or column definition they are a fixed part of the view.

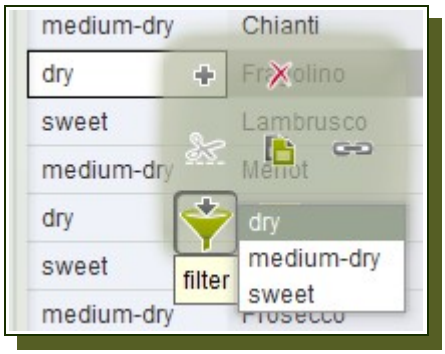
To filter a view temporarily without limiting the view permanently there are the following possibilities:

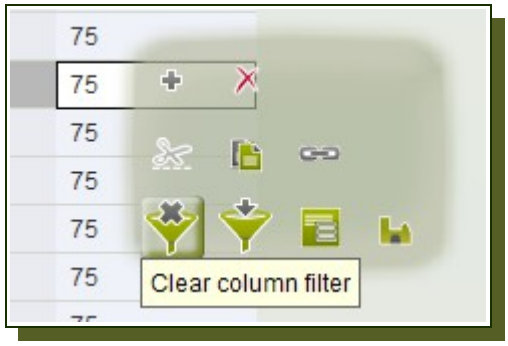
8.2 Column-filters

The simplest kind of filter is to constrain the values to only one valid element. This equals the so called „auto-filter" function known from spreadsheet applications.

To apply a column filter simply RMB on an element within the column of the view and place the cursor over the filter icon at which point a list of values appear from which to filter. The value-list is displayed in numeric then alphabetical order. You can apply multiple column filters to the same view however if a new filter is set to an already filtered column the new filter supercedes the previous one for that column. To remove a column filter simply RMB on an element in the column and select the clear column filter option, or alternatively you can clear all filters from the toolbar or operation menu.

N.B. All values in the background table will be listed however if those values don't appear in the view the filter results will appear blank.

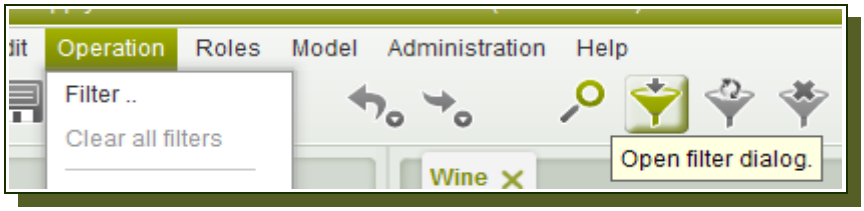




8.3 View-depending filters

8.3.1 Filter dialogue introduction

To define or to start complex filters the filter dialogue has to be opened. All users have access to the filter dialog when a view is open by selecting the *filter..* option in the operation menu or the filter icon on the toolbar.



©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

The filter dialog opens in the tabular mode and can be switched to complex filter mode using the complex filter toggle.

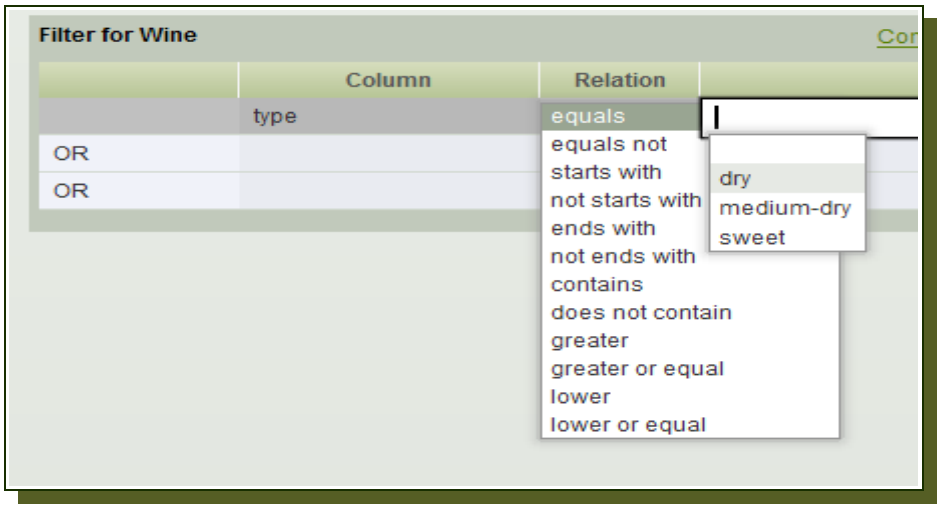
Each new filter for a view in the dialog can be given a name for future reference and is entered in the Filter name: field, which can be edited anytime the filter is opened.

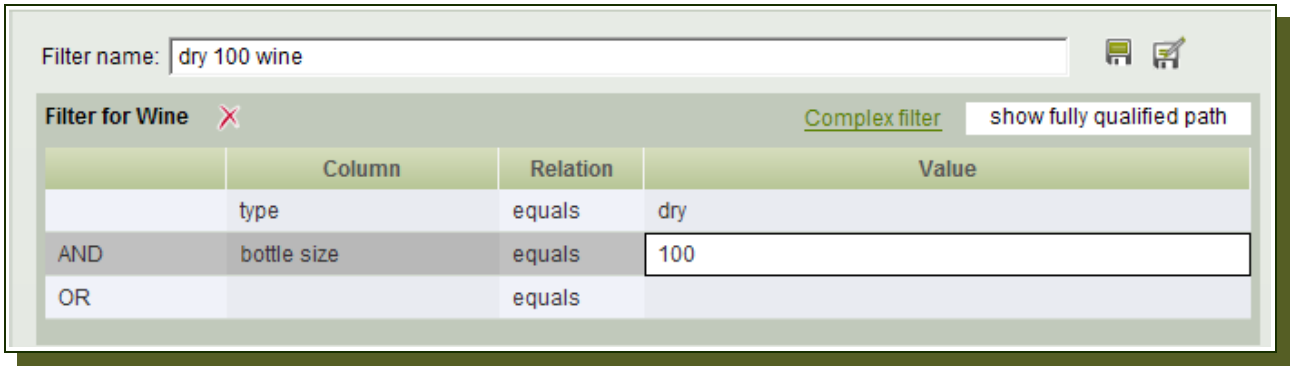
The filter explorer on the left side lists all saved filters for the view, see section 8.3.4.

8.3.2 Filter dialog tabular mode

To define simple comparison filters, tabulation is recommended, but this view is limited.

The tabular filter mode offers four filter parameters: the OR/AND, column reference, relation and value. The first line of a filter does not allow the AND/OR as you always need the first filter criteria to base conditions on, then the OR/AND condition allows you to add further conditions to your filter. The column reference, using double click, will provide you with a column list to choose from. The relation condition gives you a pre-defined choice of parameters,see below, and the value field gives you the list of values belonging to the column you have selected.





The example above shows the filter has been named „dry 100 wine" and that the filter has two conditions, that the wine description equals dry AND the bottle size equals 100.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

To view the filter based on the conditions, click apply at the bottom of the dialog and the filter results will be displayed in the view. As stated in section 8.2, you can RMB to remove a column filter or use the clear all filters in the filtered view.

8.3.3 Filter dialogue complex mode

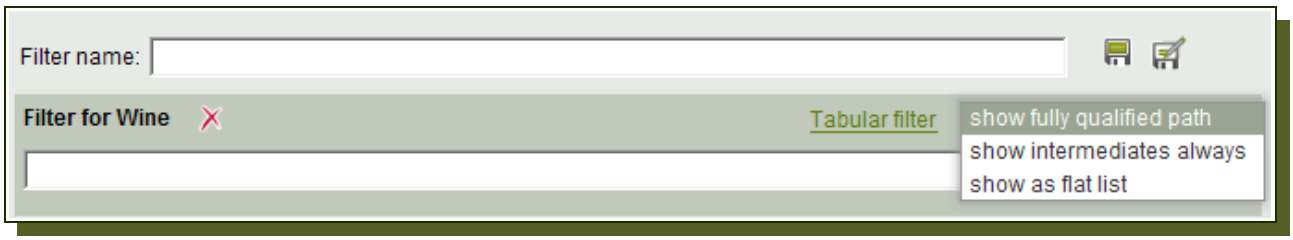
The complex filter mode can be accessed by selecting the „complex filter" toggle after which you are provided with a navigation language line in which to manually create your complex filter. Within the complex filter mode you can use all of the filter functions in JACAMAR as desribed in section 5.3.4, such as sum(), size() and last(), and it also possible to create cascaded tree structures or use extended bracket terms.

The difficulty of filtering tree-structures is how to operate with filter conditions, which won't be fulfilled by an element, but by its child lines, therefore in the complex mode there are three ways in which you can view the filtered results for cascaded tree structure filters:

show fully qualified path: this displays all of the lines, and their related lines, that meet the filter conditions.

show intermediates always: this displays only the branches that lead to the filtered results.

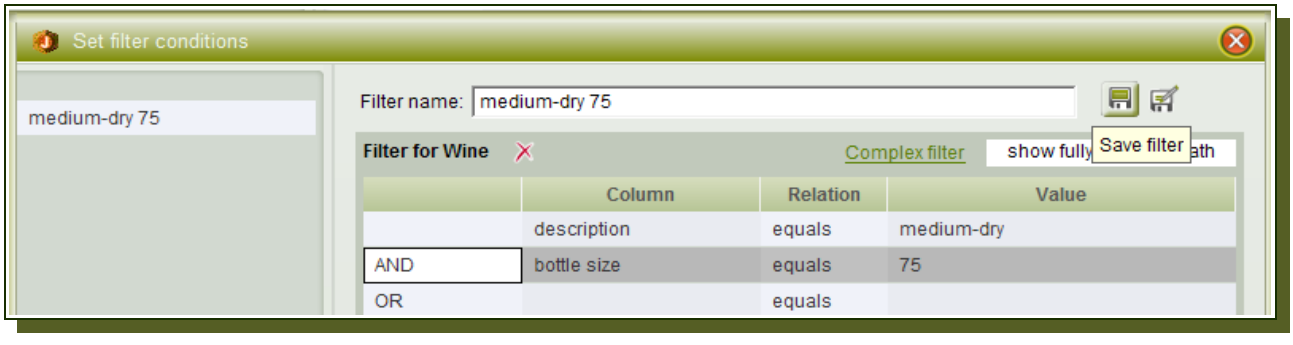
show as flat list: appears as the default, and this displays only all of the lines that meet the filter conditions ignoring the hierarchy.



8.3.4 Saving filters

It is recommended to save a set of often used filter conditions.

After naming the filter and entering the conditions, you can save the filter by clicking on the disk icon at the end of the filter name line and then it will be stored in the filter explorer on the left side of the dialog.



©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

8.3.5 Activating stored filters

When you open the filter dialog for a view the stored filters are listed on the left in filter explorer and by using double click on the filter name the filter selected will be applied.

8.4 Adding elements in filtered views

In filtered views only those elements which meet the filtered conditions will be displayed in the view.

When adding new line elements to filtered views JACAMAR adapts new elements automatically based on the filtered conditions so that the conditions are fulfilled minmally. This especially affects the comparison-filters and the `cPath()` function.

For OR-conditions, the left requirement will be fulfilled, e.g. if filter conditions are: equals"A"OR equals"B" OR equals"C"then element value „A" will be assumed. For AND-conditions both sides will be considered. Where a complex filter such as „starts with" or „contains" is present and too vague to populate the elements will be left blank.

8.5 Parametric Filters

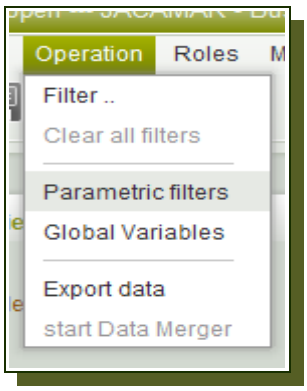
Besides the view-related filters it is possible to define common global filters to use in several views. These are so-called Parametric filters.

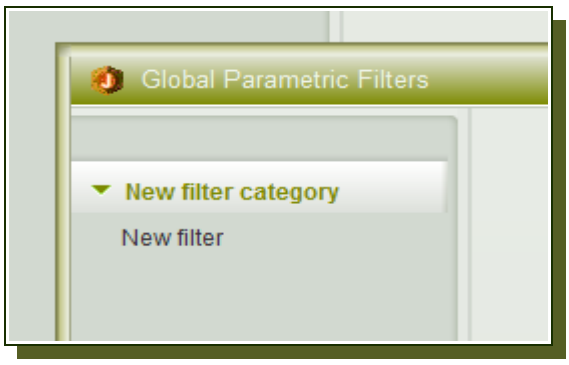
These parametric filters can only be created by an administrator.

It is possible to subsume similar filters under one category-name. While opening a view containing the function `filter`(category-name) a dialog occurs to decide which filter shall be applied on this view. In this way the view will have parameters set.

8.5.1 Creating, editing and deleting global filters

As an administrator the global filter dialog can be opened by selecting OPERATION → PARAMETRIC FILTERS from the menu bar.





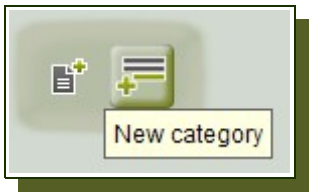
A global filter is always defined by name and belongs to a defined category. Once a category name has been saved it becomes available in the parametric filter function list see section 6.2.4.

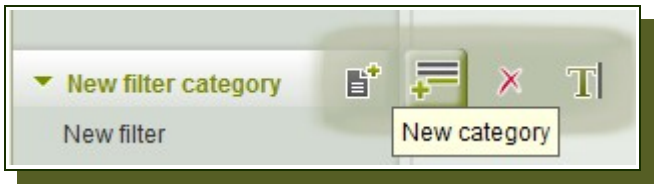
In contrast to view-depending filters, whereby types are predefined in the view, for global filters the underlying basic type has to be defined first, then the filter condtions can be set.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

Filter conditions can be defined for as many types as needed within global filters.

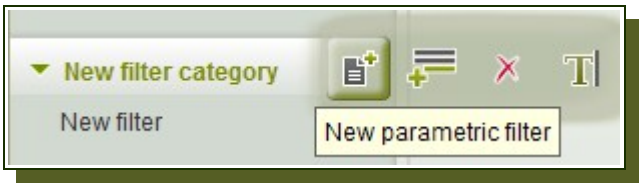
To add a new category simply RMB on an existing category or RMB into a blank area and select the „new category" option. The name „new filter category" has to be adjusted.

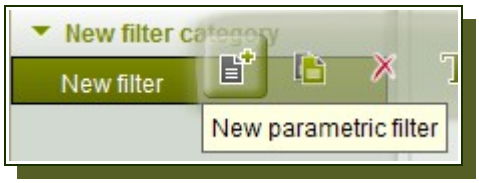




To delete a category or edit the name of a category simply use the RMB on the category and select the red cross to delete and the T font symbol to rename it.

To add a new filter simply RMB on the corresponding category or RMB on an existing filter and select the „new parametric filter" option.





If the cursor points on a category the new filter will be inserted at the top of the list, and if the cursor points on an existing filter the new filter will appear directly after the selected one. The name „new filter" has to be adjusted.

To delete a filter, make a copy of the filter or change the name of the filter simply RMB on the filter and select the red cross to delete, the copy icon to make a copy and the T font symbol to rename it.

To define the parameters of a filter, within a category, double click on a filter to open it, where the filter name can also be changed, and here we specify the basic type and the filter rule.

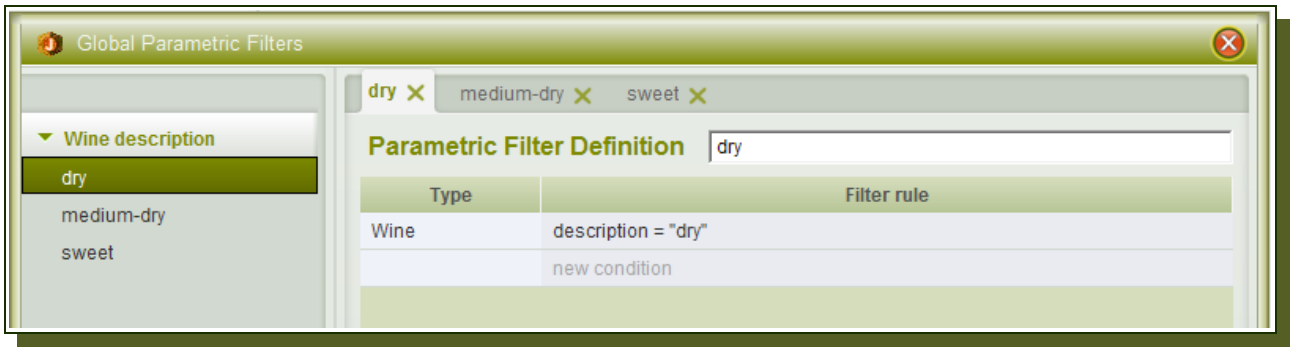
The Meta Model types will appear when using double click in the type field with options to choose from, then the filter rule must be applied to the selected type.The filter rule, as with view configuration, can be a property, related type and property or a filter function.

You can apply as many conditions to as many types as are necessary however the more complex a global filter the less views it can be applied to.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

8.5.2 Global filters – Wine example

Using our Wine Supply business model here is an example of a duly completed global filter dialog:



Within the category „Wine description" three filters have been created: dry, medium-dry and sweet.

In this example each filter contains only one condition as shown above for the dry filter. The medium-dry and sweet filter conditions are similar with the appropriate text replacing the word „dry" in the filter conditions.

The category name is what is used as the reference for the specfic filters within it and the category name is what is used in the navigation language, see section 6.2.4.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

9 User Administration

9.1 Introduction

User administration is found in the adminstration menu and at first no access-limits are defined in a new repository. In this regard JACAMAR can be used as a single-user application. User administration would only be necessary to secure secret or confidential data.

Because JACAMAR is a multi-user application it is important to define special authorisation rights and authorisation groups and in user administration it can be defined who gets access to what data and what permissions within each view. Therefore users can only view the information they need and can only update data that they have the rights for.

In this way it is possible to apply all business processes and work-flows to JACAMAR combined with high data integrity.

9.2 Defining users and access rights

9.2.1 Basic principles

JACAMAR provides a simple as well as flexible user administration model.

Users wil be identified by their user ID when logging-in to a repository.

Users access rights (privileges) on several components are defined within permission groups.

Within a permission group all rights on views and Meta Model components can be mixed. Best practice is to separate the 'read-permission group' from the 'write -access-group'.

9.2.2 Display of user administration

The function ADMINISTRATION → OPEN USER ADMINISTRATION in the menu bar opens the window for user maintenance.

This function only works, if the user-role is switched to administrator.

Opening user administration for the first time in a repository will display only one user and its current operating system user ID.

On the left side, referred to as user admin explorer, there are three sections displayed: Users, Roles and Permission groups. Each of these sections can be viewed as separate windows by placing the cursor on the section title and dragging it outside of user admin explorer and when closed they will return to their former location.

The individual users, roles and permission groups that are listed can also be opened individually by using double click on a listed item. All open tabs can also be dragged out of the window, or tiled horizontally or vertically inside the window for better display options when in user administration.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

9.2.3 Adding, editing and deleting a user

To add a new user double click on the new line 'Dbl-click for new user', or to open an existing user use double click on the user from the list.

User Profile then needs to be completed according to the user or business needs, which must include the UserID and a system generated 'Active since' date, while the other fields, Last name, First name, Business Unit, and Admin notices are optional if required.

If you edit a users UserID they will be unable to sign into the system.

You are unable to change your own UserID or delete your own user.

Each user must have at least one role and a corresponding permission group including privileges to be able to see views and data in the repository. To apply a role to a user simply highlight a role from the list and on the bottom line, when the cursor changes to the hand symbol, drag and drop the role into the applied roles section of the intended user.

To delete an applied role from a defined user simply drag and drop the role outside of the user administration window or RMB on the role and click on the red cross.

To delete a user simply right click on the user in the list and select the red cross.

9.2.4 Adding, editing and deleting a role

As with a user you double click on a new line in the role section to add a new role or double click on a listed role to open an exisiting one. The Administrator and Import Manager are built-in roles and cannot be amended or deleted.

Every role requires a name, for example Finance, Read-only or Stock(Editor) depending on the business needs. And within each role there needs to be users and permission groups assigned.

To assign users and permission groups to the role you drag them from the lists on the left and drop them into the appropriate column in the role. To remove users or access groups from a role you can drag and drop them outside of the user administration window or RMB on the user or group and click on the red cross.

To delete a role from the list simply RMB on the role and select the red cross.

9.2.5 Adding, editing and deleting a permission group

Using double click on a new line in the permission groups section will open a new permision group and double click on a listed group opens an existing permission group.

Every permission group requires a name, for example Finance, Sales(Edit) or Marketing depending on the business needs, and within each group there needs to be views and roles assigned. If a group requires further privileges for Meta Model components then they must also be assigned to the group.

To assign roles to the permission group you drag them from the list of roles and drop them in the role name column.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

The assigned view or component column can include: any view from the list of views and the components of your Meta Model (object type, property or relation).

Views from the View Explorer list can be moved across by highlighting one from the list, hovering on the bottom line and when the cursor changes to the hand symbol it can be dragged and dropped into the assigned view column in user administration. To remove a view from a group simply DND the view outside of the user admin window or RMB on the view in the group and click on the red cross.

Either from the GMM or TMM in the drag mode the Meta Model components can also directly be moved into the assigned component column. This is done by highlighting and using the hand symbol cursor below the component in the TMM or by simply clicking on the object type or property in the GMM and dragging it across.

Once listed in the assigned view or component column they can be dragged outside of the user administration window, or RMB on the listed item clicking on the red cross, to be removed from the group.

9.2.6 Authorisation rights on views and Meta Model components

In JACAMAR there are four kinds of components that can have privileges assigned:

- Views, Types, Properties and Relations

And there are four authorisation levels (privileges):

- Hidden – this applies to MM components added to the PROTECTED group only!
- Read only, Read/Write and Delete – for views and MM components within a group.

The following table shows the impact of the privilege on the view or component.

	Views	Types	Properties	Relations
Hidden- (PROTECTED Permission Group)	N/a (A view is hidden by default unless assigned to a permission group)	This type its properties, relations and resulting values of the type are hidden*	This property and all resulting base values of this property are hidden*	This relation and all resulting values through this relation are hidden*
Read Only	View is visible and read only	Allows a „Protected Type" its properties, relations and resulting values to be visible	Allows a „Protected Property" and its resulting values to be visible	Allows a „Protected Relation" and its resulting values to be visible
Read/Write	N/a	Elements of this type can be created in a view	Cell values of this property can be updated/ edited	Related elements can be cut, copied, linked or deleted in cascaded views

* Unless the privilege will be overwritten

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

Delete	N/a	Elements of this type can be deleted in a view	N/a	N/a

The first step in assigning privileges to a permission group is to assign the views that the roles in this group should have access to, then if editing or deleting privileges are to be allowed the corresponding components and privileges must be added.

When a view is added to the assigned view column it becomes read only. The default privilege for Meta Model components is read/write, however the privilege for types can be changed to delete using double click on the privilege level and select the delete option.

The „read only" privilege for Meta Model components only becomes available if the corresponding component has been placed in the PROTECTED group.

9.2.7 Permission group – PROTECTED

The PROTECTED permission group applies to all non-admin users and, unless further privileges have been assigned, all components listed in the PROTECTED group in the TMM, GMM and resulting values within the views are hidden. These components are also unavailable as part of navigation paths or column definition evaluations.
This is designed so that any sensitive or private business model information or values will not be visible unless *read only*, *read/write* or *delete* privileges have been assigned for the components for a specific group, its roles and the users belonging to the role.

For edit or delete privileges in user admin to be effective the view configuration for each view also needs to correspond as a third stage of the data integrity and control, as explained in section 6 View Configuration.

9.2.8 Specific admin-rights

Every user can be authorised by an administrator to get administration rights.

To work as an administrator the user first has to switch the role in the Roles menu in the toolbar. This principle enables all JACAMAR administrators to work in daily business as a „normal co-worker" without having several logins. The roles menu will only appear if the user has been assigned multiple roles by the administrator.

When an authorised user switches ~~their~~ his role to Administrator all limitations will stop and full permissions will apply.

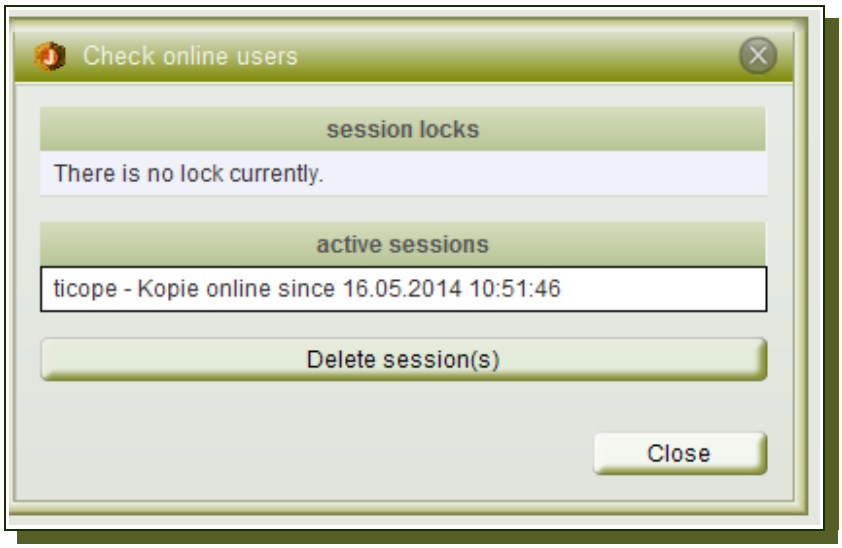
Administrators also have some special rights e.g. to start a backup manually or to edit the Meta Model.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

9.3 Administration extras

9.3.1 Check online users

In a multi-user environment this function gives you an overview of connected and active working users. You are able to delete user sessions, however please be careful executing this command as this may cause a loss of data and it is always advisable to contact the users and ask them to save their changes and exit their session first.



9.3.2 Show admin history

This logfile allows you to see all changes made under Admin permissions, (Meta Model, view configuration, user administration).

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

10 Backup of repositories

10.1 Introduction

It is mandatory for administrators to secure all data periodically.

Because of file-based user collaboration, it is very important in JACAMAR to regenerate the correct data status after a software crash.

Therefore JACAMAR provides a smart backup mechanism.

10.2 Automatic system backups

The automatic backup mechanism works in the background once the first user session is opened each day. Right after loading all data from the repository it is then saved in the backup/ directory.

Daily backups will be stored in a zip file named: **`runtime/backup/backup_YYMMDD.zip.`**

This zip file contains:

- functional data (including change-archive)
- Access control data
- View and filter configuration data
- all error-logs of the day before

10.3 Manual backups

According to requirements an administrator is free to do additional backups.

It is recommended to do additional backups especially before making changes in the Meta Model. If the changes do not work , the status can be restored.

This function is located in the ADMINISTRATION MENU by selecting BACKUP REPOSITORY: It is only displayed when the user role is switched to administrator. In contrast to a system back up file the manual backup file adds the clock time beside the calendar date in the file name.

Manual backups will be stored in a zip file with a time stamp: in the following format: **`runtime/backup/backup_YYMMDD_HHMM.zip`**

10.4 Loading backups

A backup file is a common zip-file. Entries and paths are saved according to the normal structure of a repository.

This zip file has to be copied to the original location of the (corrupted) repository and has to be renamed to **`*.jcmr.`**

When opening the repository next time the latest version will be processed.

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

11 Updates & Rollouts

11.1 Update Mechanism

Due to the fact that many users working with a common JACAMAR repository at the same time it is necessary that all users have the same version of JACAMAR software installed.

JACAMAR supports an Update-Process which is easy to maintain and ensures a centralised roll-out within a companys infrastructure. In every ***config.ini*** after a users installation the parameter `jacamar.updateIndex` has to be set. This value is a path to a file containing all Update-statements. This file can be stored in a shared drive or a web location where the users have read permissions. If such a file is present all statements in the file will be executed before starting the JACAMAR application.

The file can hold the following statements for example:

```
load:plugins/de.jacamar_5.2.58.v20120210.jar

# Note: You no longer need org.eclipse.help.appserver plugin in 3.4 (Ganymede).
delete:plugins/org.eclipse.help.appserver_3.1.400.v20090429_1800.jar
load:plugins/org.eclipse.help.webapp_3.4.1.v20091009_35x.jar

load:configuration/config.ini force
```

Every command starts with `load` or `delete`. Comments start with a hash sign "#".

11.2 Loads

A load command has the following syntax

```
load:relative_path/filename [force]
```

The given path is relative to the index file. If this file is already present the file will be overwritten if it is followed by the option [`force`].

A specific feature exists for ***Plugins***: if the file name contains an '_' all files having the same string before the '_' in the filename within the folder will be deleted.

It is easy to replace obsolete versions of JACAMAR modules. For instance *de.jacamar_5.0.2.jar* replaces the new version *de.jacamar_5.0.3.jar* .

11.3 Deletes

JACAMAR modules no longer used, can be deleted using the *DELETE*-command.

```
delete:relative_path/filename
```


©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de

12 Online Information

Up to date information will always be published at the JACAMAR website.

http://docu.jacamar.biz/

Further information about Advanced functions like Arithmetics, DateTime and Casting Operation can be found online at

http://docs.oracle.com/javase/tutorial/java/

©2015 KATLA GmbH | *Immermannstraße* 28 | D - 39108 Magdeburg | www.katla-gmbh.de