
Knowage

Knowage Labs

Aug 04, 2023

Table of Contents

1	How to use Knowledge	3
1.1	Data security and access management	3
1.1.1	Behavioural Model	3
1.1.2	User profile and roles	3
1.1.2.1	User Management	6
1.1.2.1.1	User unlock	6
1.1.2.1.2	Roles settings	8
1.1.3	Analytical Driver	8
1.1.4	List Of Value	12
1.1.4.1	Parametrizing LOVs	16
1.1.4.2	Creating a validation rule	17
1.1.4.3	Creating an analytical driver	19
1.1.4.4	Creating an analytical driver for a spatial filter	23
1.1.4.5	Analytical driver's use modes	24
1.1.4.6	Behavioural Model Lineage	24
1.1.5	Analytical Document	24
1.1.5.1	Main concepts	24
1.1.5.1.1	Document types	26
1.1.5.2	Register an analytical document	28
1.1.5.2.1	Analytical documents on Server	28
1.1.5.2.1.1	Document lifecycle	31
1.1.5.2.1.2	Template Versioning	32
1.1.5.2.1.3	Document Visibility	32
1.1.5.3	Visibility rules	34
1.1.5.4	Association with analytical drivers	34
1.1.5.4.1	Correlation between parameters	36
1.1.5.4.2	Correlation through LOV and drivers	38
1.1.5.4.3	Controlled visibility	38
1.1.5.5	Cross Navigation	40
1.1.5.5.1	Declaration of the output parameters	41
1.1.5.5.2	Cross navigation definition	41
1.1.6	Repository Management	43
1.1.6.1	Repository structure and rights	43
1.1.7	Menu management	44
1.1.7.1	Menu configuration	44
1.1.7.1.1	Setting menu icon	45

1.2	Create a new Data Source	49
1.2.1	Big Data and NoSQL	53
1.2.1.1	Hive	53
1.2.1.2	Spark SQL	54
1.2.1.3	Impala	54
1.2.1.4	MongoDB	54
1.2.1.5	Cassandra	55
1.2.1.6	Google Big Query	55
1.2.1.7	Google Cloud Spanner	56
1.2.1.8	Amazon RedShift	56
1.2.1.9	Azure Synapse	56
1.3	Create a new Data set	56
1.3.1	SQL Query Data Set	57
1.3.2	File Data Set	58
1.3.3	Flat Data Set	58
1.3.4	REST Data Set	58
1.3.5	SolR Data Set	63
1.3.6	SparkQL Data Set	64
1.3.7	CKan Data set	64
1.3.8	Python Data Set	64
1.3.9	QBE Data Set	65
1.4	Create a New Dashboard	65
1.4.1	My first Cockpit	67
1.4.1.1	Text widget	68
1.4.1.2	Image widget	68
1.4.1.3	Chart widget	71
1.4.1.4	Table widget	100
1.4.1.5	Cross Table widget	108
1.4.1.6	Document section	116
1.4.1.7	Active Selections widget	116
1.4.1.8	Selector Widget	118
1.4.1.9	HTML Widget	120
1.4.1.10	Map Widget	124
1.4.1.11	Discovery Widget	129
1.4.1.12	Python/R Widget	131
1.4.1.13	Custom Chart Widget	136
1.4.1.14	Cross Navigation	149
1.4.1.15	Widget properties	152
1.4.2	General configuration	152
1.4.3	Data configuration	155
1.4.3.1	Source	156
1.4.3.1.1	Parametric sources management	156
1.4.3.2	Associations	156
1.4.3.3	Indexes	159
1.4.3.4	Frequency	159
1.4.3.5	Variables	160
1.4.3.6	Template	160
1.4.4	Selections	160
1.4.5	Clear cache	160
1.4.6	Save	162
1.4.7	Multisheet functionality	162
1.4.8	Export cockpit	162
1.5	Create Report	164
1.5.1	Create a Birt report	164

1.5.1.1	Developing a BIRT report	164
1.5.1.1.1	Adding parameters to reports	172
1.5.1.2	Cross Navigation for BIRT Reports	172
1.5.2	Jasper reporting	178
1.5.2.1	Document definition*	178
1.6	Create Free Inquiry	182
1.6.1	How to build a Metamodel	182
1.6.1.1	Metamodel creation	183
1.6.1.2	Association with analytical drivers	183
1.6.1.2.1	Correlation between parameters	185
1.6.1.2.2	Correlation through LOV and drivers	186
1.6.1.2.3	Create an empty model	186
1.6.1.2.4	Editing the metamodel	187
1.6.1.2.5	Create a new relationship	191
1.6.1.2.6	SQL Filter	192
1.6.1.2.7	Create a new business class	193
1.6.1.2.8	Table property list	195
1.6.1.2.9	Column property list	195
1.6.1.2.10	Generate the datamart	196
1.6.1.2.11	Additional functions for business model	200
1.6.2	How to build a QBE	201
1.6.2.1	My first Query By Example	201
1.6.2.1.1	Query design and execution	202
1.6.2.1.1.1	Datamart Schema	202
1.6.2.1.1.2	Calculated fields management	206
1.6.2.1.1.3	Filters	206
1.6.2.1.1.4	Query Preview	211
1.6.2.1.2	Advanced QbE functionalities	212
1.6.2.1.2.1	Spatial fields usage	212
1.6.2.1.2.2	Subqueries	216
1.6.2.1.2.3	Parameters	216
1.7	Create a new Registry	216
1.7.1	Registry features	218
1.7.1.1	JPivot Registry characteristics	220
1.7.2	Registry development	221
1.7.2.1	Filters	222
1.7.2.2	Analytical driver	223
1.7.2.3	Profile attributes	223
1.7.2.3.1	Multivalue	223
1.7.2.4	JPivot Registry instance	223
1.7.3	Logging & auditing	224
1.8	Create location Intelligence	225
1.8.1	Create GIS document	225
1.8.1.1	Basic concepts	225
1.8.1.2	More on GIS and Spatial Data	226
1.8.1.2.1	Spatial Data	226
1.8.1.2.2	GIS	228
1.8.1.3	Analytical document execution	229
1.8.1.3.1	Extra functionalities	231
1.8.1.4	GEOReport Engine	233
1.8.1.5	Template building with GIS designer	235
1.8.1.6	Designer sections	236
1.8.1.6.1	Dataset & Layer	236
1.8.1.6.2	Dataset join	238

1.8.1.6.3	Indicators	238
1.8.1.6.4	Filters & Menu	238
1.8.1.6.5	Edit map	238
1.8.1.7	Template building with GIS designer for technical user	240
1.8.1.8	Cross navigation definition	240
1.8.2	Create SVG document	241
1.8.2.1	My first SVG Map or design	241
1.8.2.1.1	Technical activities	244
1.8.2.1.1.1	SVG Catalogue	244
1.8.2.1.1.2	SVG Format	244
1.8.2.1.1.3	Datasets definition	246
1.8.2.1.1.4	Template building	246
1.8.2.1.1.5	Advanced functionalities	253
1.9	Create Data Mining analysis	253
1.9.1	Functions Catalog	254
1.9.1.1	The General tab	255
1.9.1.2	The Input tab	256
1.9.1.3	The Script tab	256
1.9.1.4	The Output tab	258
1.9.2	Engine description	259
1.9.3	Use a function inside documents	259
1.10	Create Multi dimensional analysis	264
1.10.1	Create OLAP	264
1.10.1.1	OLAP user manual step by step	264
1.10.1.1.1	The filter panel	264
1.10.1.1.2	The filter cards	264
1.10.1.1.3	Axes panel	265
1.10.1.1.4	Pivot table	267
1.10.1.1.5	Side bar	267
1.10.1.2	Functionalities	271
1.10.1.2.1	Placing hierarchies on axes	271
1.10.1.2.2	Swaping axes	272
1.10.1.2.3	Selecting different hierarchies on dimension	272
1.10.1.2.4	Slicing	274
1.10.1.2.5	Filtering	276
1.10.1.2.6	Drill down and drill up	276
1.10.1.2.7	Drill through	277
1.10.1.2.8	Refreshing model	280
1.10.1.2.9	Showing MDX	280
1.10.1.2.10	Showing parent members	280
1.10.1.2.11	Hiding/showing spans	281
1.10.1.2.12	Showing properties	281
1.10.1.2.13	Suppressing empty columns/rows	282
1.10.1.2.14	Sorting	282
1.10.1.3	Creation of an OLAP document	283
1.10.1.3.1	About the engine	283
1.10.1.3.2	Development of an OLAP document	284
1.10.1.3.2.1	Schema modelling	284
1.10.1.3.2.2	Engine catalogue configuration	286
1.10.1.3.2.3	OLAP template building	286
1.10.1.3.2.4	Creating the analytical document	288
1.10.1.3.3	OLAP Designer	289
1.10.1.3.3.1	Profiled access	293
1.10.1.3.4	Cross Navigation	294

1.10.1.3.4.1	Cross navigation on members	295
1.10.1.3.4.2	Cross navigation from a cell of the pivot table	295
1.10.2	Create What-If	296
1.10.2.1	Interface details	297
1.10.2.2	Meta-language description	299
1.10.2.3	What-if analysis implementation	302
1.10.2.3.1	Workflow description	302
1.10.2.3.2	Schema definition	302
1.10.2.3.3	Changes in the mondrian schema	304
1.10.2.3.4	Changes in the designer	306
1.11	Create a Dossier	307
1.11.1	XML Template	307
1.11.1.1	Tags and properties	307
1.11.2	Image adding (PPT_TEMPLATE)	309
1.11.3	Image replacing (DOC_TEMPLATE)	309
1.11.4	My first dossier	310
1.12	Create Performance Analysis	312
1.12.1	Create KPI	312
1.12.1.1	KPI development	313
1.12.1.2	Creation of a KPI document	329
1.12.2	Create Scorecard	331
1.12.2.1	Scorecard development	331
1.12.2.2	Creation of a Scorecard document	337
1.12.3	Create Alert	339
1.12.3.1	Alert implementation	339
1.13	Create an ETL Process	342
1.13.1	KnowageTalendEngine	342
1.13.2	Job design	344
1.13.3	Job deploy	344
1.13.4	Template building	345
1.13.5	Creating the analytical document	345
1.13.6	Job execution	345
1.13.7	Job scheduling	347
1.14	How use Localization Management	347
1.14.1	System labels	347
1.14.2	User labels	348
1.14.3	BIRT labels	348
1.15	How create a Data Preparation	348
1.15.1	What is that for?	348
1.15.2	Preparing a dataset	350
1.15.3	Data Preparation Transformations	350
1.15.4	Data Preparation technical detail	354
1.15.5	Saving and Using a prepared dataset	360
2	Knowage management	367
2.1	Add new Data Source	367
2.1.1	Connect to your data	367
2.1.2	Big Data and NoSQL	369
2.1.2.1	Hive	370
2.1.2.2	Spark SQL	371
2.1.2.3	Impala	371
2.1.2.4	MongoDB	371
2.1.2.5	Cassandra	372
2.1.2.6	Google Big Query	372

2.1.2.7	Google Cloud Spanner	373
2.1.2.8	Amazon RedShift	373
2.1.2.9	Azure Synapse	373
2.2	Scheduler	374
2.2.1	Create an Activity	374
2.2.1.1	Save as snapshot	377
2.2.1.2	Save as file	377
2.2.1.3	Save as document	377
2.2.1.4	Send to Java class	379
2.2.1.5	Send mail	381
2.2.1.5.1	Static list	381
2.2.1.5.2	Dynamic list with mapping dataset	382
2.2.1.5.3	Dynamic List with script	382
2.2.2	Scheduling panel	382
2.2.3	Scheduler Monitor	383
2.3	Server manager	383
2.3.1	Tenants	385
2.3.2	Templates	385
2.3.3	Themes	386
2.3.4	Download installation configuration	387
2.3.5	Licenses	387
2.3.6	Events	387
2.4	Import/Export	388
2.4.1	Artifacts	389
2.4.2	Documents	392
2.4.3	Menu	396
2.4.4	Users	398
2.4.5	Catalogs	399
2.4.6	KPIs	400
2.4.7	Analytical Drivers	401
2.4.8	Glossary	403
2.4.9	Alerts	406
2.5	Time Span	406
2.5.1	Create a new Timespan	406
2.6	News	408
2.6.1	How to publish news	408
2.6.2	How the end user can read the news	409
2.7	Glossary	409
2.7.1	Glossary management	409
2.7.2	Glossary Usage	414
2.7.3	Help Online functionality	415
2.8	Resource manager	417
2.8.1	Resource Manager functionalities	417
2.9	Widget gallery	426
2.9.1	Gallery management	426
2.9.2	Dashboard gallery	428
3	Installation and configuration	431
3.1	Requirements	431
3.1.1	Operating systems	431
3.1.2	Disk usage	432
3.1.3	Java environment	432
3.1.3.1	Linux	432
3.1.3.2	Windows	432

3.1.4	Application server	432
3.1.4.1	Tomcat 9.0	432
3.1.4.1.1	Tomcat on Linux	432
3.1.4.1.2	Tomcat on Windows	434
3.1.5	Database schema for metadata	434
3.1.6	Database schema for data	434
3.1.7	NodeJS requirements	434
3.1.7.1	CentOS	435
3.1.7.2	Ubuntu	435
3.1.8	Chromium requirements	436
3.1.9	Support to non-latin languages	436
3.1.10	Supported browsers	437
3.1.11	Data Preparation requirements	437
3.2	KNOWAGE Installation	437
3.2.1	Software release	437
3.2.1.1	Optional: using MAC address for license	438
3.2.2	Manual installation	438
3.2.2.1	Metadata database initialization	438
3.2.2.2	Using Tomcat	438
3.2.2.2.1	Dependencies	438
3.2.2.2.2	File system resources	439
3.2.2.2.3	Connection to metadata database	439
3.2.2.2.4	Cache database connection	439
3.2.2.2.5	Connection to business data	440
3.2.2.2.6	Environment variables definition	440
3.2.2.2.7	Changing the secret key for password encryption	441
3.2.2.2.8	Mandatory configuration	441
3.2.2.2.9	Recommended configuration	442
3.2.2.2.10	Applications deploy	442
3.2.2.2.11	Thread pool definition	442
3.2.2.2.12	Advanced memory settings	442
3.2.2.2.13	Advanced Connector settings	443
3.2.2.3	Datasource link within the applications	443
3.2.2.4	Configuration of the metadata db dialect	443
3.2.2.5	Modification of the Quartz configuration	444
3.2.2.5.1	Clustering	444
3.2.2.6	Logging	445
3.2.2.7	Enable Java Security Manager	445
3.2.2.8	Installation of Chromium Cockpit Export script	446
3.2.2.9	Configuring environment for Data Preparation	446
3.2.3	Python Engine	447
3.2.3.1	Install knowage-python webservice	448
3.2.3.2	Run knowage-python webservice	448
3.2.3.3	Configure Knowage to enable Python/R functionalities	448
3.2.4	R Engine	449
3.2.4.1	Install knowage-r webservice	449
3.2.4.2	Run knowage-r webservice	449
3.2.5	Knowage CE Installer	449
3.2.5.1	Server-side requirements	449
3.2.5.1.1	Operating system	449
3.2.5.1.2	Java platform	450
3.2.5.1.3	Memory	450
3.2.5.1.4	Disk usage	450
3.2.5.1.5	Database	450

	3.2.5.1.6	Application server	450
	3.2.5.1.7	Proxy settings	450
	3.2.5.2	Client-side requirements	451
	3.2.5.2.1	Browser	451
	3.2.5.2.2	Proxy settings	451
	3.2.5.3	Launching	451
	3.2.5.3.1	Windows	451
	3.2.5.3.2	Linux/macOS	451
	3.2.5.4	Managing Knowage CE	451
	3.2.5.4.1	Windows	452
	3.2.5.4.2	Windows (embedded MariaDB option)	452
	3.2.5.4.3	Linux/macOS	452
3.3		How to upgrade KNOWAGE	452
	3.3.1	Preliminary operations	452
	3.3.2	Upgrade operations	453
3.4		How to configure the KNOWAGE Single Sign On	454
	3.4.1	CAS SSO	454
	3.4.1.1	Deploy of the CAS application	454
	3.4.1.2	HTTPS certificate	455
	3.4.1.3	Configuration of the HTTPS protocol for Tomcat	455
	3.4.1.4	Knowage configuration	456
	3.4.2	Azure SSO	458
	3.4.3	LDAP	461
	3.4.3.1	Authentication	461
	3.4.3.2	Authentication + authorization	463
	3.4.4	Google SSO	464
3.5		Server settings	466
	3.5.1	Thread manager	466
	3.5.2	Cache parameters	467
	3.5.3	Logging	468
	3.5.4	Mail server	469
	3.5.5	Maximum file size	470
	3.5.6	Date format	470
	3.5.7	Language	471
	3.5.8	Adding new languages	471
	3.5.9	Password constraints settings	471
	3.5.10	Login security settings	472
	3.5.11	Resource export folder cleaning settings	472
	3.5.12	Import / Export	473
	3.5.12.1	Users	473
	3.5.13	Main Menu	473
	3.5.14	Changing the secret key for password encryption	473
	3.5.15	Audit Table Management	473

Knowage is the professional open source suite for modern business analytics over traditional sources and big data systems. Knowage is the new brand for SpagoBI project: this new brand marks the value of the well-known open source business intelligence suite after significant functional and technological transformations and a new offering model. The suite is composed of several modules, each one conceived for a specific analytical domain. They can be used individually as complete solution for a certain task, or combined with one another to ensure full coverage of user' requirements.

1.1 Data security and access management

1.1.1 Behavioural Model

An important topic related to the visibility on documents and data according to the roles and profiles of the end users, is the **Behavioural Model**. The behavioural model is based on four main concepts:

- *user profile*, defining the user roles and attributes;
- *repository rights*, defining the users rights in terms of document accessibility;
- *analytical drivers*, defining which data of a document can be shown to the user;
- *presentation environment* settings, defining how the user can reach and run his own documents.

and mainly answers the following questions:

- *WHO* uses the business intelligence solution (user profile);
- *WHAT* is visible to users, in terms of documents and data (repository rights and analytical drivers);
- *HOW* users work with their documents (analytical drivers and presentation environment settings).

1.1.2 User profile and roles

Knowage users are defined by:

- identities,
- roles,
- profiles.

The *identity* of a user consists of a set of data used to identify that user, i.e., a username and a password, as well as a human readable full name.

The *profile* of a user consists of a set of properties called attributes, describing general information about the user, e.g., age and gender, but also domain-specific properties, such as the organizational unit to which he belongs. Some attributes, such as name and email, are defined by default in Knowage. Others can be added by the model administrator, as explained in the following sections.

The *role* of a user represents a categorization of a group of users. These roles may correspond to specific positions in the company, e.g., “general manager” or a “sales director”, or to a position with respect to the BI project, e.g., “data administrator” and “BI developer”. Different users may have the same role, as well as the same user may have multiple roles.

You will not have grants to create new roles or users, but you are asked to match them during document profilation phases. In the following we are going to describe the elements needed for adding parameters. This elements involves profilation too. To conclude we will see how to manage accessibility while creating a document.

Table 1.1: Knowage Role Types.

Role Type	Description	Standard User
ADMIN	General administrator. Manages all Knowage functionalities.	biadmin
MODEL_ADMIN	Model administrator. Manages the Behavioural Model and its associated functionalities.	bimodel
DEV_ROLE	Developer. Creates and modifies datasets and documents.	bidev
TEST_ROLE	Test user. Tests analytical documents.	bitest
USER	End user. Executes documents visible to him and creates ad-hoc reporting analysis.	biuser

Knowage allows you to create several roles, according to your project needs. However, all roles must belong to a specific *role type*. A role type is a higher-level categorization used by Knowage, in order to map roles for the different features of the suite.

Pre-defined roles are summarized in the Table 5.1. The first four roles are technical roles, while the last one, the user, is the actual end user. Each role type has a default user associated to it. Other users can be created and associated to a role type.

When a user logs in into Knowage, his profile is automatically loaded. The full name is visible by clicking on the info button at the bottom left corner of the page.

Authentication can be handled internally by Knowage or delegated to an external Single Sign On (SSO) system.

Hint:

Authentication Management: The choice of handling authentication internally or delegating it to an external SSO system typically depends on the presence of an authentication system already in place. If this is the case, Knowage can seamlessly integrate with the existing authentication infrastructure.

Once the user has logged in, his role is loaded. Roles are managed internally. In case the user is associated with multiple roles, he will be asked to choose one.

Alternatively, from the Knowage Administrator menu, he can select a default role that will be kept valid until he logs out.

Steps to follow in the definition of a behavioural model:

- Create profile attributes;

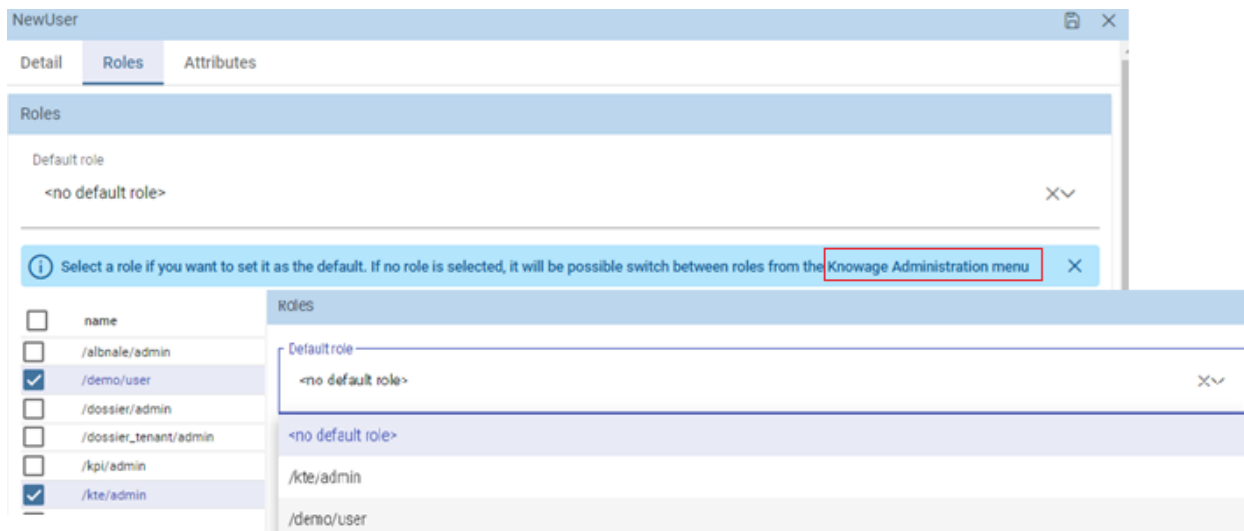


Fig. 1.1: User roles in Knowage.

- Create roles;
- Create users and associate profile attribute values and roles to them.

Knowage supports the management of user profiles and roles through the **Profile Management** menu section. This menu is only visible to Knowage administrator and to the model administrator, since users and roles management is a critical operation that requires an appropriate level of responsibility.

The **Profile Management** menu section contains three sub-menu items:

- **Profile Attributes:** to define new profile attributes and manage the existing ones.
- **Roles:** to create new roles and manage permissions for each role.
- **Users:** to create users, manage their identities, assign values to their profile attributes and associate them with roles.

In the following, we show how the model administrator can define user profiles and roles using these functionalities. Remember that Knowage profile management can also be integrated with external profiling systems.

Clicking on **Profile Attributes**, the list of currently defined attributes is shown. To add a new attribute, click on the **Plus icon**: a new row is added to the list, where you can insert the Name, the Description and the Data type. To delete an attribute, select the corresponding row and click on the **Delete** icon.

Attributes defined in this section will be available to all user profiles. It is not mandatory to assign a value to each attribute for each user, since profile attributes without values will not be considered in the definition of the user profile.

In addition to the profile attributes created by administrator, by default Knowage provides the following profile attributes:

- **user_id:** set with the user unique identifier;
- **user_roles:** set with user roles selected from the ROLES tab in Users Management menu;
- **TENANT_ID:** set with the tenant unique identifier;
- **user_session_roles:** set like *user_roles* attribute, if no default role is set. Set with default role selected, otherwise.
- **language:** set with the language selected by the user

The image below, shows a new *Profile attribute* named *Country* manually fed. The alternative option to feed the attribute is by means of a Lov, from where a value is going to be picked up when assigning the attribute to a user. Multivalue option in this case is not enabled, so only one value is inserted.

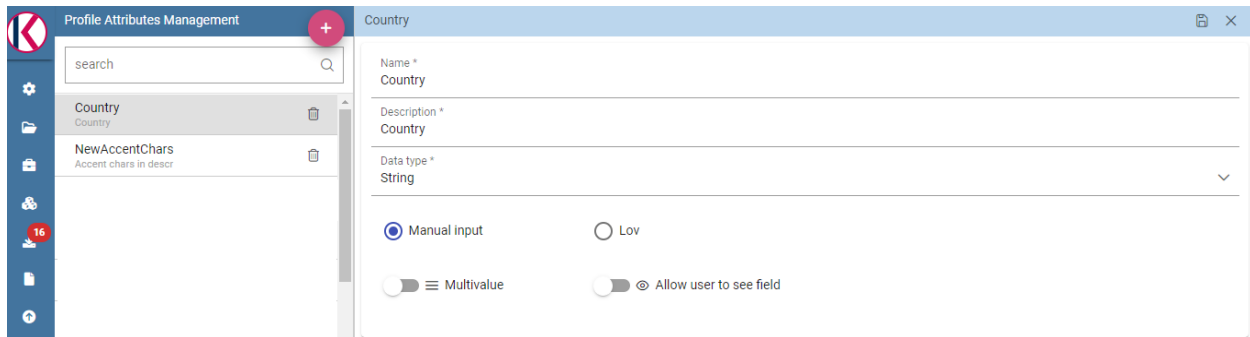


Fig. 1.2: Profile attributes Management.

Once the attributes are defined, the model administrator can define roles, using the **Roles** functionality. The role management tool is two-sided: on the left you can see the list of already defined roles. At the beginning of a project, only default roles are visible. To add a new role, click the **Plus** icon and move to the right panel. To delete a role, simply click on the **Delete** icon available for that role once saved.

Hint:

Role Management: The behavioural model should be built taking into account the specificity of each organization and the needs of the BI project. Therefore, it is a good practice to define specific roles for the BI project and avoid using Knowage technical roles only.

The right panel contains the following tabs. The **Detail** tab allows the administrator to define role name and role type (mandatory). The role type regulates the visibility of that role based on the types already described. A code and a description can be added too, as shown below.

The **Authorizations** tab allows you to assign permissions to each role. Rights are predefined and grouped into categories, as shown above.

The **Business Models**, **Data sets** and **KPI Categories** tabs are intended to assign specific categories to each role, in a way that each user can only see the business models, datasets or KPI that belong to the categories associated with his role.

The **Business Models** tab is only available for modules KnowageBD and KnowageSI, while the **KPI Categories** tab is only available for KnowagePM. More details on business models and KPIs can be found in the corresponding chapters.

1.1.2.1 User Management

The **User Management** section includes a left panel that allows the administrator to create and delete users, and a right panel that allows the management of user details, roles and attributes.

1.1.2.1.1 User unlock

If user reaches the maximum number of failed login attempts (editable in advanced configurations), it will be blocked by Knowage and access will be denied. By accessing Knowage with a user having user management privileges, the blocked user will be displayed with a red warning sign and it will be possible to unlock it using the “unlock user” button. After that, the user will be able to log in using the latest set of credentials.

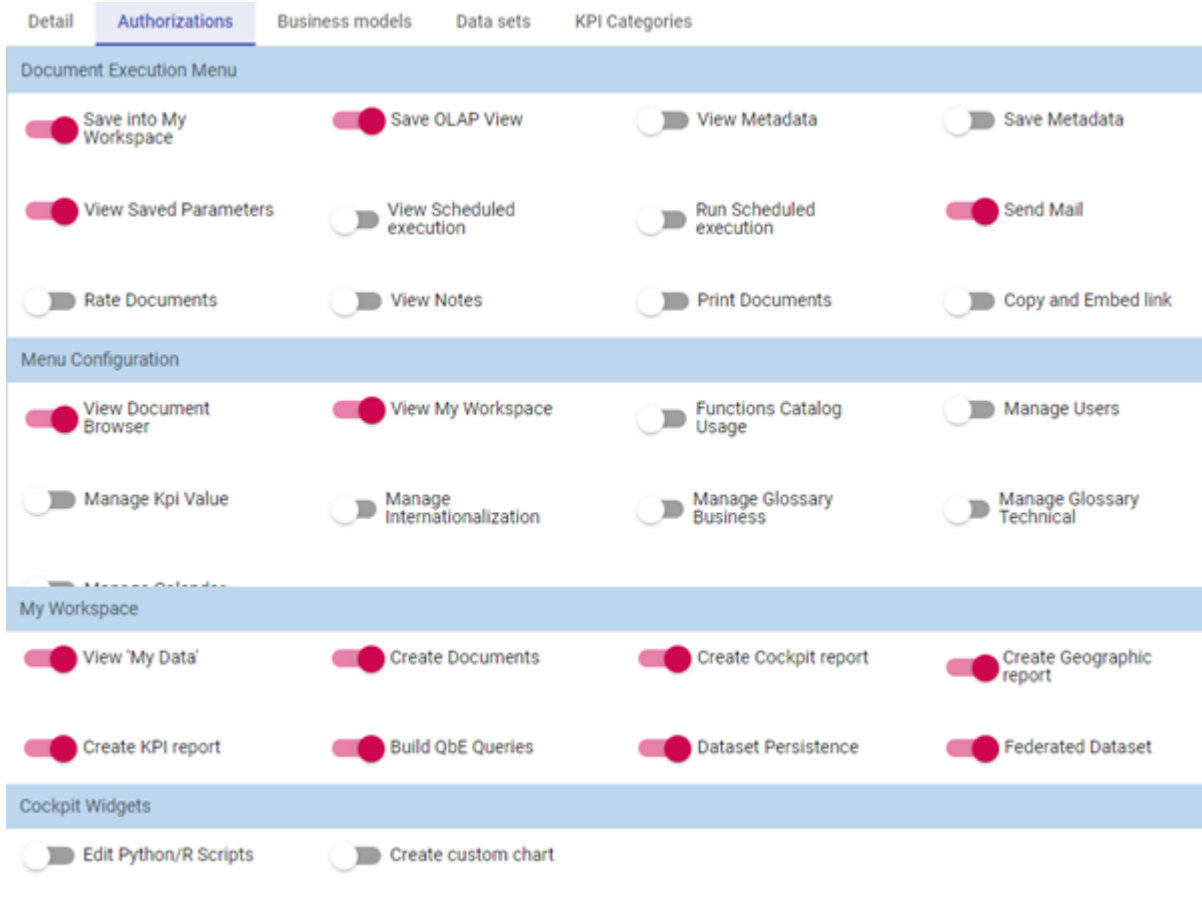


Fig. 1.3: Roles Management.



Fig. 1.4: Users Management.

Fig. 1.5: Users Management - Roles settings example

1.1.2.1.2 Roles settings

Clicking on the ROLES tab you have to select one or more roles to associate with the user. After that, if more than one role is associated to the user, you can choose the default role by selecting it from the combo box on the top of the page.

Default role is optional: if you don't select a default role, at login time all the available roles for the user will be loaded. If you select a role, at login time it will be the session role selected.

Fig. 1.6: Users Management - Roles settings example

In the example above, for the user “*Prep_admin*” you can choose from “*admin*” and “*kte_admin*” as default role.

You can also assign an attribute profile to a user. In this case it is enough to valorize the attribute that you want to assign. The below image, shows an example. The attribute *Country* formerly created as a *Profile attribute* has been assigned to the user *kte_admin*. The image shows that some other attributes are available but they are not considered as they remain empty.

1.1.3 Analytical Driver

An analytical driver (hereafter simply driver) models a concept or a piece of data frequently used as a distinguishing criterion on the global data context. A driver highlights the concepts guiding the analysis, providing a unique repre-

Fig. 1.7: Users Management - Attributes settings example

sentation of them and describing how they are shown and checked according to the end users' roles. When connected to analytical documents, a driver produces an explicit or implicit parameter used to filter data.

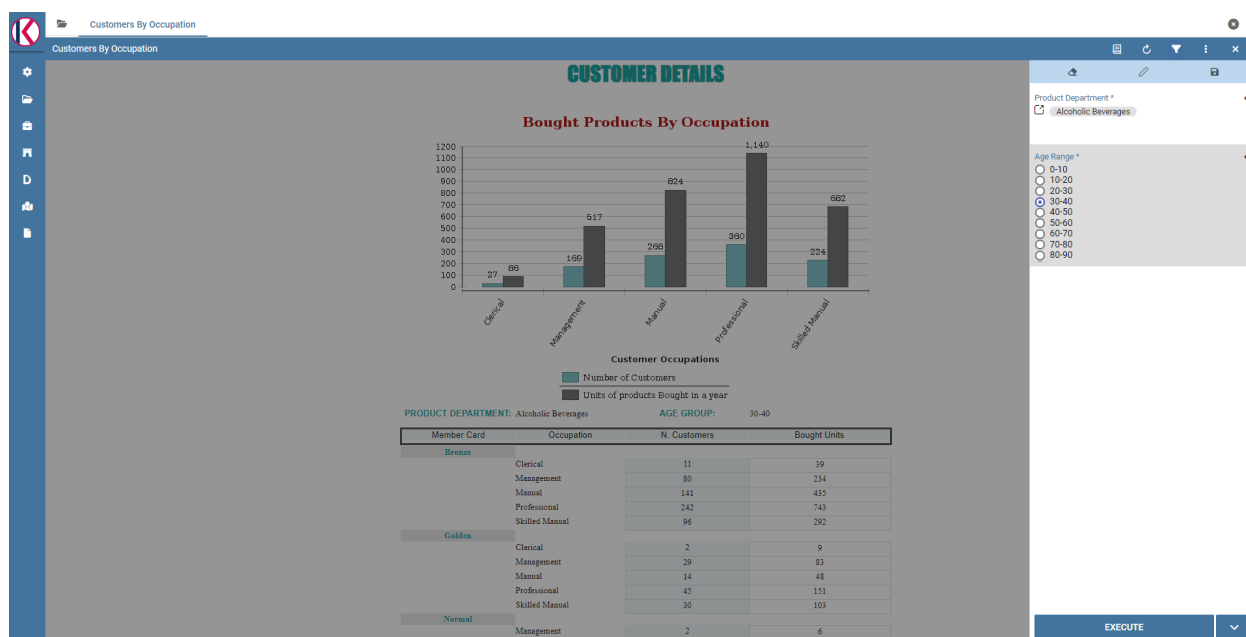


Fig. 1.8: Parametric Report.

The Figure above represents a report with two parameters:

- the Department, a mandatory field, displayed as a lookup and with possible values: Alcoholic Beverages, Baked Goods, Baking Goods and so on;
- the Age Range, a mandatory field, displayed as list of values and with possible values 0-10, 10-20 and so on.

All these aspects are regulated by the analytical driver behind each parameter. In particular, each driver provides many *use modes*, defining:

- Who is involved in a specific use mode, in terms of a list of end user roles, considering that a role can be associated to a single use mode only.

- What data he can access and how they are presented to the end user for his potential selection. This information is provided by the so called *List of Value (LOV)*.
- How to check the validity of the chosen values. This information is provided by the so called *Check*.

In other terms, each use mode refers to an initial visualization method and content (LOV), to one or more validation rules (check) and to one or more end user roles (roles). The logic of a driver is represented in Figure below.

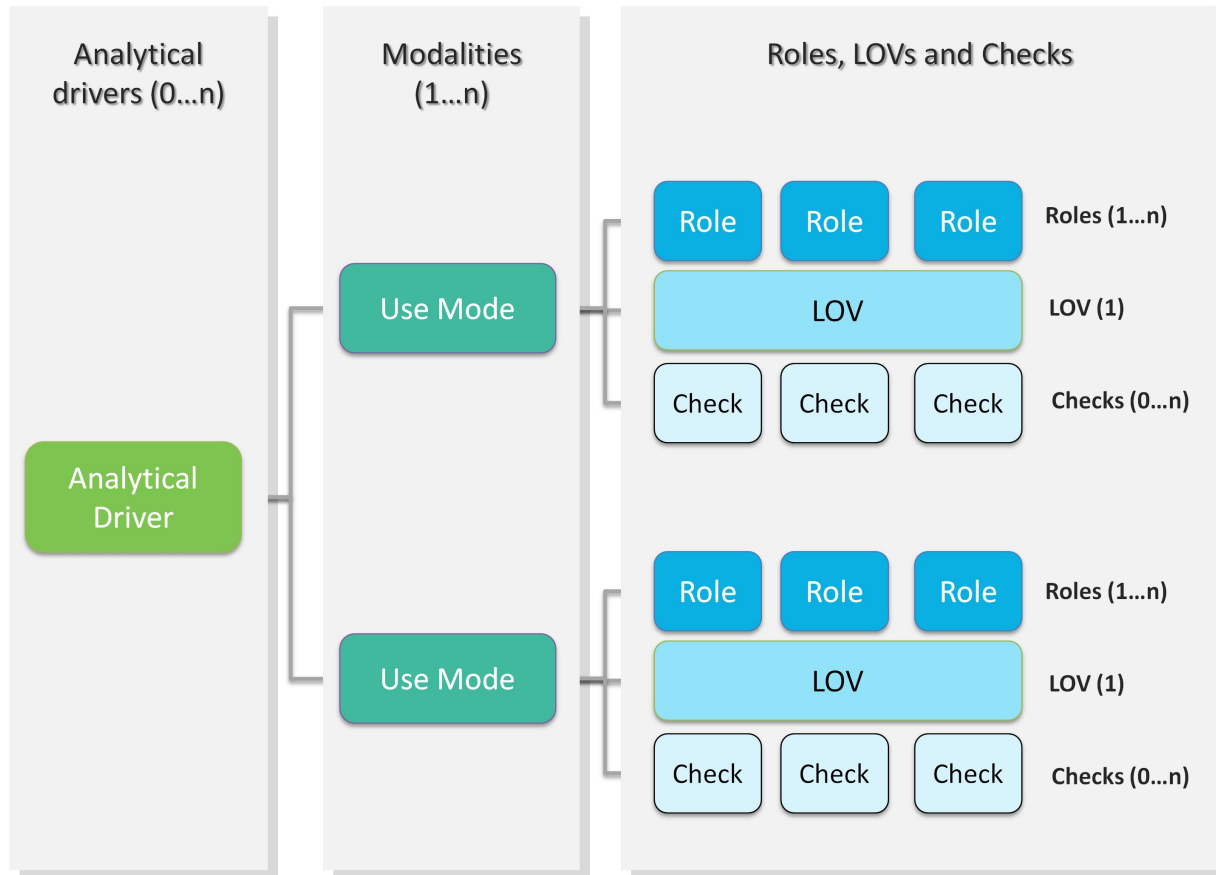


Fig. 1.9: Analytical driver schema.

Let's consider the following example. We need to represent the concept of "product family". Since this is a common driver and discriminator for the enterprise analysis, an analytical driver will be coded, with all its behavioural rules, such as:

- if the user is a call center operator or a user that provides internal support, he can manually write the product family he wants to select. This value will be formally verified (it must be a text) and checked on the product family registry.
- if the user is a product brand director or an operative secretary, he can choose the value from a preloaded list of all the product families belonging to his brand. For this reason, the value does not need any check.

Once defined, a driver can be related to many documents, driving their behaviour and filters in a common way. This way, a user who runs different documents that use the same drivers always receives the same parameter form, applying the same filters over shown data. In fact, when an authenticated user (with its roles and profile) runs an analytical document, its technical metadata are read, mainly in terms of document template and related drivers. Based on them, a customized page for the parameters input is produced, according to the driver logic for the end user role. The selected values are then validated and the final output comes to the user. Next figure shows this process.

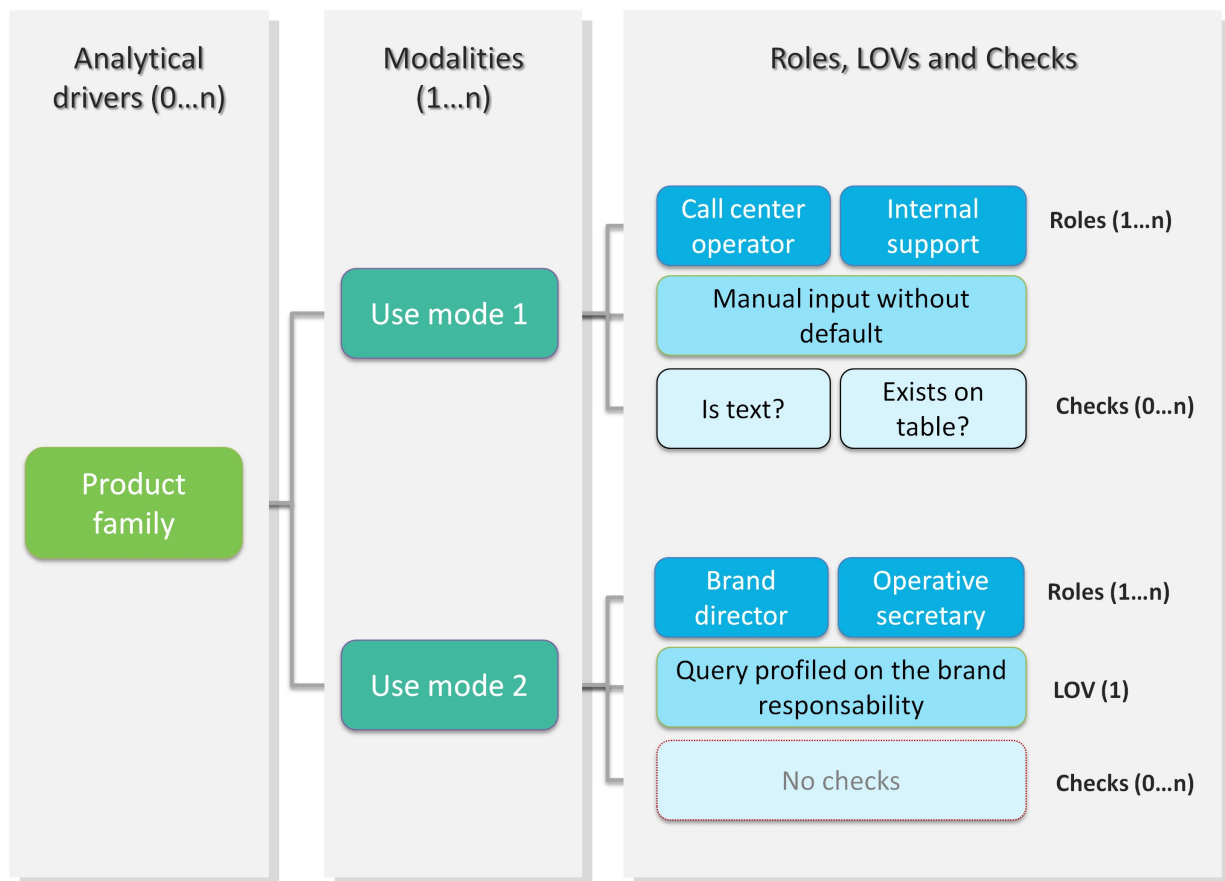


Fig. 1.10: Analytical driver schema - Example.

Thanks to analytical drivers, a single document is able to cover the analytical demands of various categories of users, with noticeable advantages in terms of:

- reduction of the number of documents to be developed and maintained,
- consistency in the request for parameters,
- complexity reduction in the development of documents, thanks to the separation between security matters and massive development,
- simple maintenance of the security (visibility over data) over time, despite the increase of developed documents or added engines.

In the next paragraphs we explain how to create a new analytical driver together with its basic components.

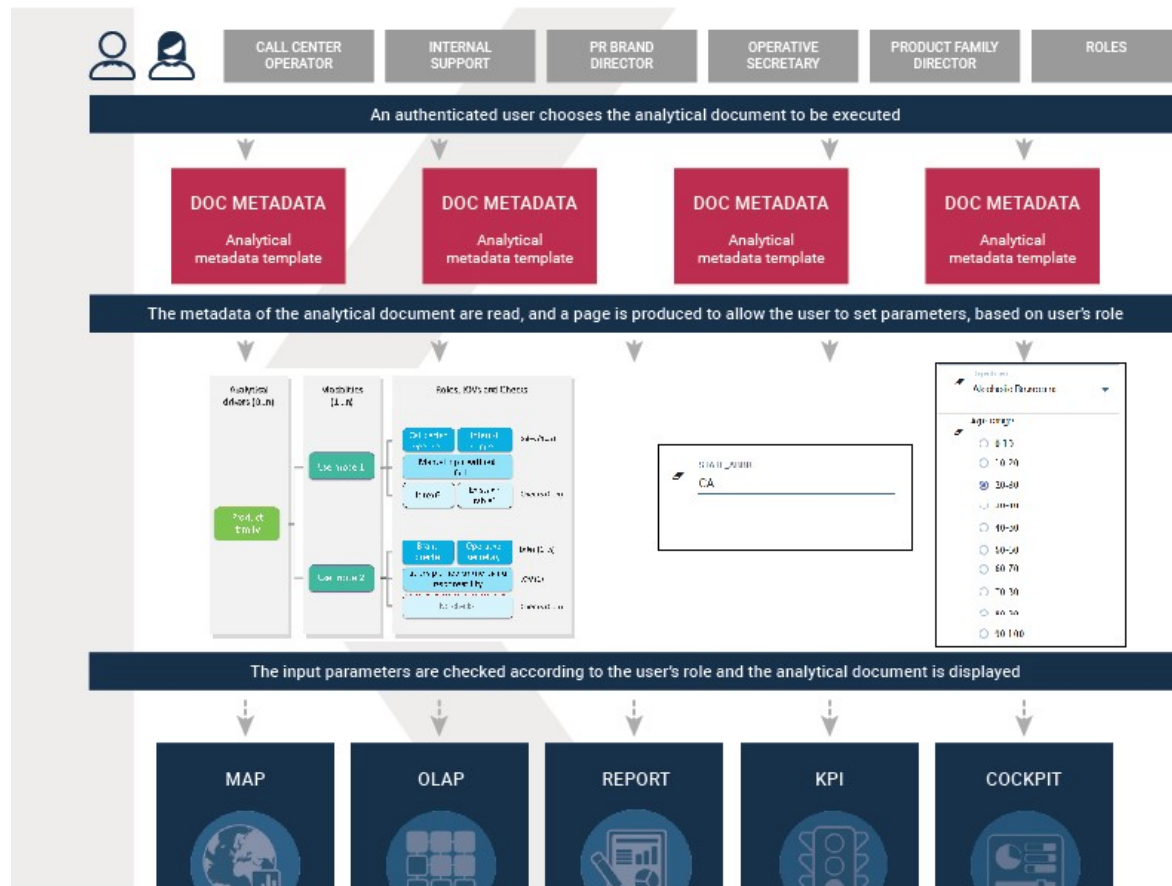



Fig. 1.11: Overall process.

1.1.4 List Of Value

A *List Of Value* (LOV), is a collection of data organized in attribute-value fashion. For example, the LOV in LOV example retrieves id, name and food family for a product.

Listing 1.1: LOV example

```
1 {195, High Top Almonds, Food};
2 {522, Tell Tale Walnuts, Food};
3 {844, Very Good Soda, Drink};
```


To create a new LOV, click on the icon  at the top right corner of the page. The LOV creation interface will open, where you can set label, name and description, choose the LOV type and define its values accordingly.

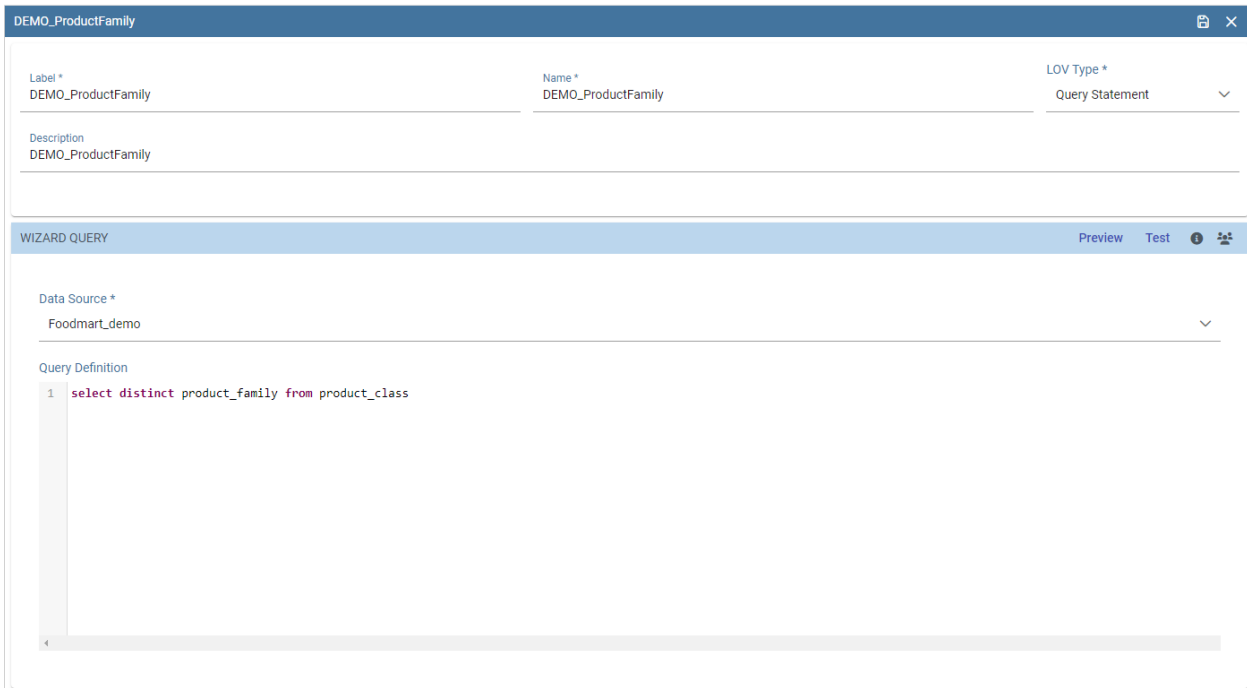


Fig. 1.13: LOV Creation interface.

Once completed the form, click on **Preview** button to enable the **Test** button. Notice that you cannot save the LOV without testing it, since this allows to detect errors before the LOV is actually used in a driver and associated to a document. After testing, you will be able to define which column is the actual value of the LOV, i.e., which value will be passed to the analytical driver using this LOV. Only *one* column can be the value attribute and only *one* column can be chosen as Descriptive attribute, while the others can be visible. The two figures below exhibit an example. Columns that are not visible can be used for correlating drivers.

Note: Correlating analytical drivers

Drivers can be correlated so that the value of the first driver is used as a parameter to select values in the second. Read more at *Analytical document* chapter.

We stress that the visibility of specific fields serve to improved human readability when applying filters to documents handled by third users. Moreover it is possible to choose (refer to next figure) between **simple**, **tree** and **tree with selectable internal nodes** typology of LOV. The last two are hierarchical and let the user visualize the parameters together with their logical tree structure.

Note: Create a LOV for the default value of an analytical driver of type Manual Input Date

This note is useful when using an analytical driver of type Date with an input of type Manual. In the case you want to use a particular date as default value for that driver, you have to use this syntax for the LOV: select '2017-09-10#yyyy-MM-dd' as fixed_date. Instead of the fixed date 2017-09-10 you can also use as default date the today date for example; in this case you can use a query of this type: select concat(to_date(now()), '#yyyy-MM-dd') as today. The most important thing is to concat to the default date you want to use the string #yyyy-MM-dd.

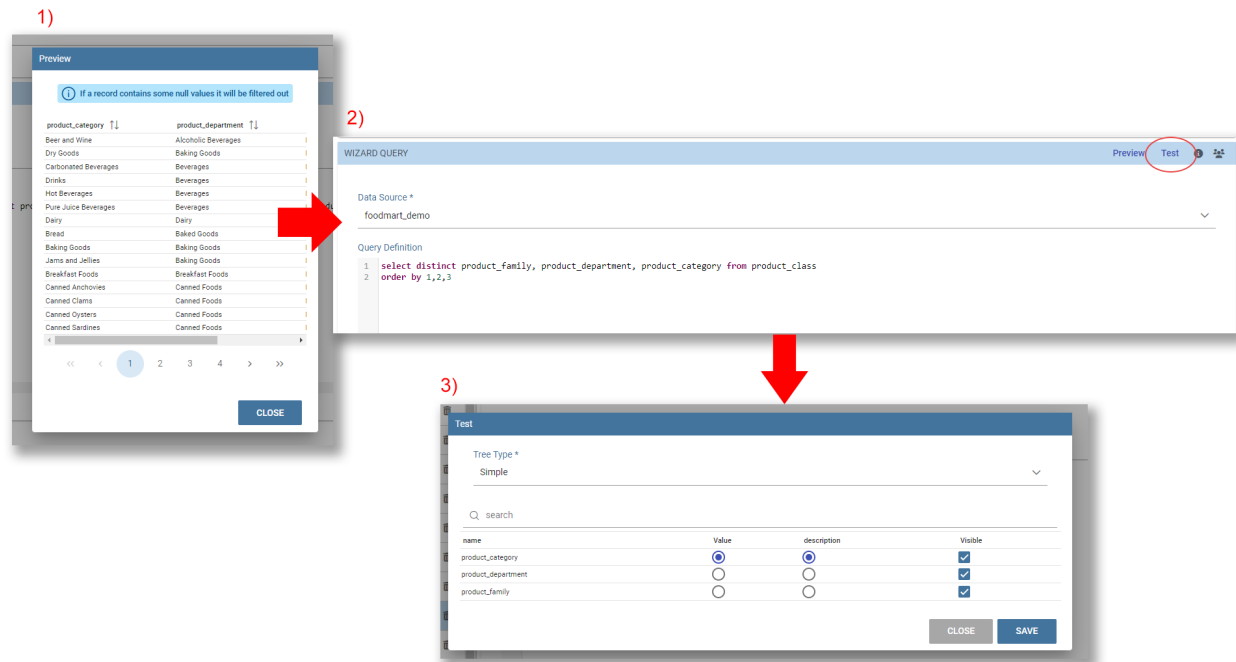


Fig. 1.14: Preview and Test of the LOV.

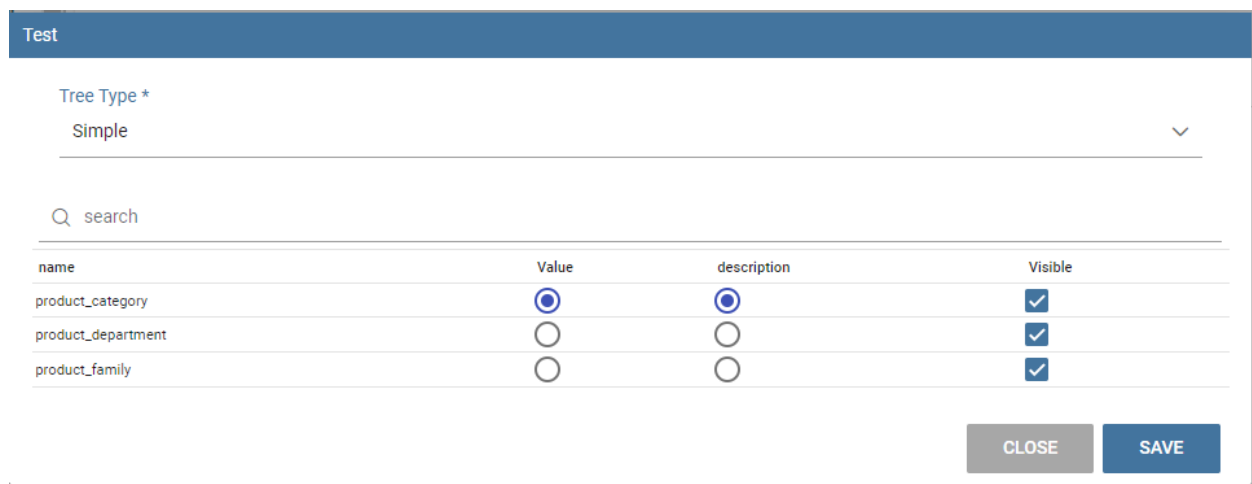


Fig. 1.15: Test of the LOV.

The screenshot shows a 'Test' window with a 'Tree Type' dropdown menu. The dropdown is open, showing four options: 'Tree', 'Simple', 'Tree' (which is highlighted), and 'Tree selectable inner nodes'. Below the dropdown, there is a list of product categories: 'product_category', 'product_department', and 'product_family'. At the bottom right of the window, there are two buttons: 'CLOSE' and 'SAVE'.

Fig. 1.16: Hierarchical LOV definition.

Note: Create a LOV for the default value of an analytical driver with a hierarchical LOV

In case you want to add a default value to an analytical driver with an input of type hierarchical LOV you need to use another hierarchical LOV with the default values desired. If the analytical driver LOV is of type *Tree* then the default LOV need to be of type *Tree* too. The LOV need to have values for the leaf level only. Otherwise, if the analytical driver LOV is of type *Tree selectable inner nodes* the default LOV need to be of the same type. The default LOV may have values for one of the level used in the hierarchical LOV. For example, suppose you have an analytical driver with a hierarchical LOV having levels Product Family > Product Category > Product Department. If the hierarchical LOV is of type *Tree* then in the default LOV you need to insert one or more values for the level Product Department. Your default LOV have one level, the Product Department. In case the LOV is of type *Tree selectable inner nodes* you can choose one of the three levels. Your default LOV have one level between Product Family, Product Category or Product Department.

1.1.4.1 Parametrizing LOVs

Suppose that you need to retrieve a list of values representing all brand names of your products. Then you can use a Query LOV like in Query LOV example:

Listing 1.2: Query LOV example

```

1 SELECT DISTINCT PRODUCT_FAMILY, BRAND_NAME
2 FROM PRODUCT

```

This is suitable for end users like the general manager who need to see all brands for every product family. Suppose now that another end user is, for example, the food manager. He should not see every brand name, but only those related to the Food product family. This could be done using user's profile attributes.

In particular, all query except the List of fixed values type can be parameterized using profile attributes. This means that, at LOV execution time, the value of the attribute in the user's profile is assigned to a placeholder in the LOV query/script. Suppose that, in our example, the food manager user has the profile attribute `pr_family` equal to Food.

You can write this second Query LOV using the placeholder with the standard syntax `${profile_attribute_name}`, as shown in Parametric query.

Listing 1.3: Parametric query

```
1 SELECT DISTINCT PRODUCT_FAMILY, BRAND_NAME
2 FROM PRODUCT
3 WHERE C.PRODUCT_FAMILY = '${pr_family}'
```

Then, at LOV execution time, for the user food manager the query becomes as shown in Runtime placeholder substitute and hence the corresponding LOV will return only the brand names related to the Food product family.

Listing 1.4: Runtime placeholder substitute

```
1 SELECT DISTINCT PRODUCT_FAMILY, BRAND_NAME
2 FROM PRODUCT
3 WHERE C.PRODUCT_FAMILY = 'Food'
```

This means that if you are the food manager and your user has the profile attribute `pr_family=Food`, then you will see only the brand related to the food family as a result of this LOV; while if you are the drink manager and your user has consequently the profile attribute `pr_family=Drink`, you will see only the brand related to drink family products.

Note: Standard profile attributes

There are some standard profile attributes always available that don't need to be defined for each user. These profile attributes are:

- `user_id` contains the user id of the logged in user
- `user_roles` contains the current user's roles, joined as a SQL IN clause fashion, for example: 'general_management','human_resources_management'
- `TENANT_ID` contains the tenant to which the user belongs


Note that an information button and a profile attribute button are available to guide user in writing the code properly, using the syntax correctly and typing the right profile attribute name.

1.1.4.2 Creating a validation rule

Knowage supports the validation of the document's input parameters via validation rules. Validation rules can be defined in **Behavioural model > Constraints Management**. A validation rule checks parameter values as given by LOVs to verify that they comply with the defined constraints.

Knowage default checks are:

- **Alphanumeric:** it checks if the parameter is alphanumeric;
- **Numeric:** it checks if the parameter is numeric;
- **Letter String:** it checks if the parameter is a letter string;
- **E-Mail:** it checks if the parameter is an e-mail;
- **Fiscal Code:** it checks if the parameter has the correct syntax of a fiscal code;
- **Internet Address:** it checks if the parameter is an internet address.

If the administrator needs to create additional validation rules, he can click on  to open the rule creation interface. Here he can define a customized validation rule using the available check options:

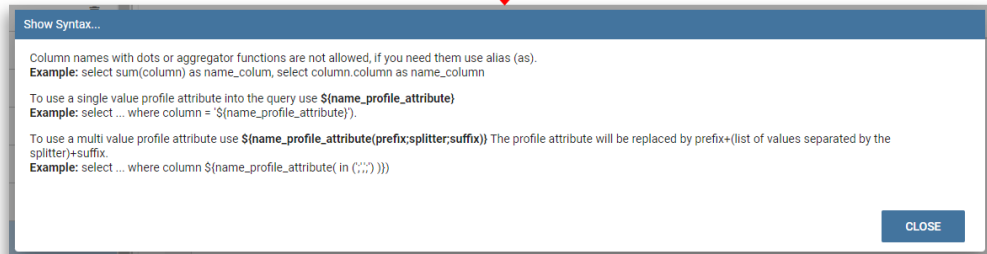
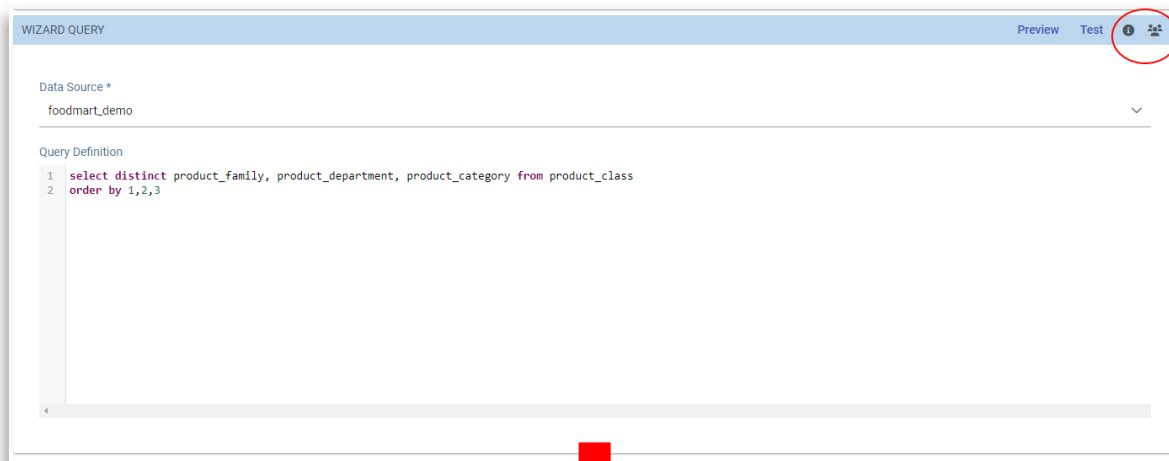


Fig. 1.17: Assistance in retrieving syntax and profile attributes.

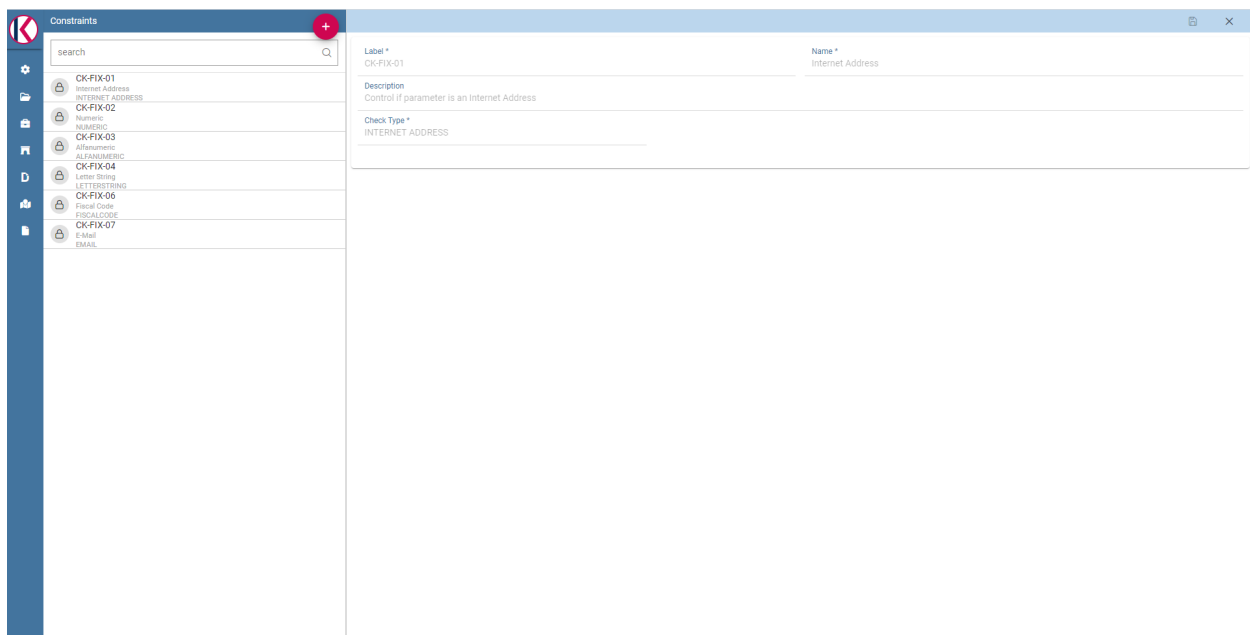


Fig. 1.18: Constraints Management.

- **Date:** here you can set a customized format type of date;
- **Regular Expression:** to set a regular expression validation rule;
- **Max/Min Length:** it lets you set the maximum and/or minimum character parameters length;
- **Range:** to set a range the parameters value has to satisfy;
- **Decimal:** to set a maximal decimal places for the parameters.

1.1.4.3 Creating an analytical driver

As explained at the beginning of this section, analytical drivers use information about users, their roles and profiles to filter data returned by their associated LOVs. Users, roles and profiles must have been already defined in the project context so that they are available to the driver.

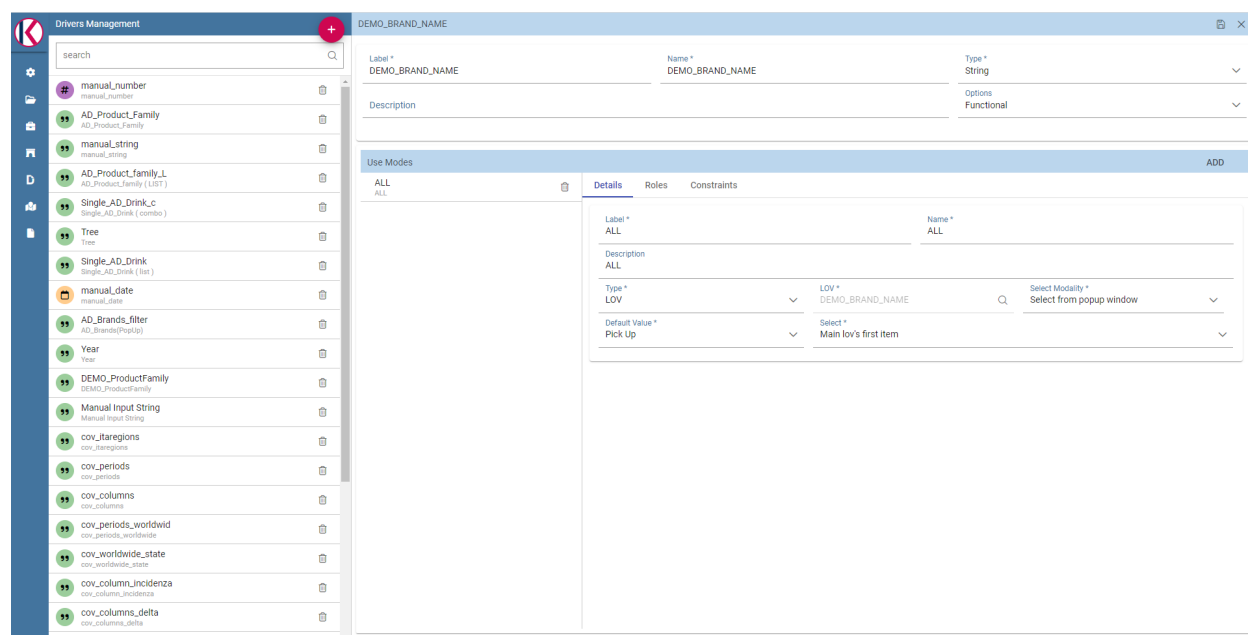




Fig. 1.19: Analytical Driver Management.

To create a driver, select Behavioural Model > Analytical Drivers Management from the developer menu. Here, you will see the entire list of available drivers. For each driver, the list shows unique label, description and type. To explore details the user must just select one menu item from the list and they will appear in the half right side, as shown in the

figure above. Otherwise to delete one analytical driver the user must use the icon  available at the end of each row of the list. Notice that you cannot delete a driver if a document is currently using it.

To create a new driver, click on  at the top right corner. The driver creation interface will open. At first execution only the upper part of the window is visible, as shown in the figure below. The upper part is the **Detail** section, where you can set the label, name and description. Choose the type between Date, String or Number depending on the type of expected data. Select Functional or Temporal if the driver is used by an end user or a scheduler, respectively. A click on the save button, enabled as soon as the form is filled in, will save the driver and let the section below appear.

In the Analytical Driver Use Mode Details section, one or more LOVs are linked to the current driver, as well as roles and checks are assigned via the so-called *use modes*.

New

Label *

Name *

Type *


Description

Options

Use Modes

ADD

No data found



Use modes

No use mode selected

Fig. 1.20: Driver creation.

To associate LOVs to the driver, switch to the “Analytical Driver Use Mode Details” tab. Here the user must set label and name of that specific use mode, the kind of input among **LOV input**, **Manual input** and **Map input**, as shown in below.

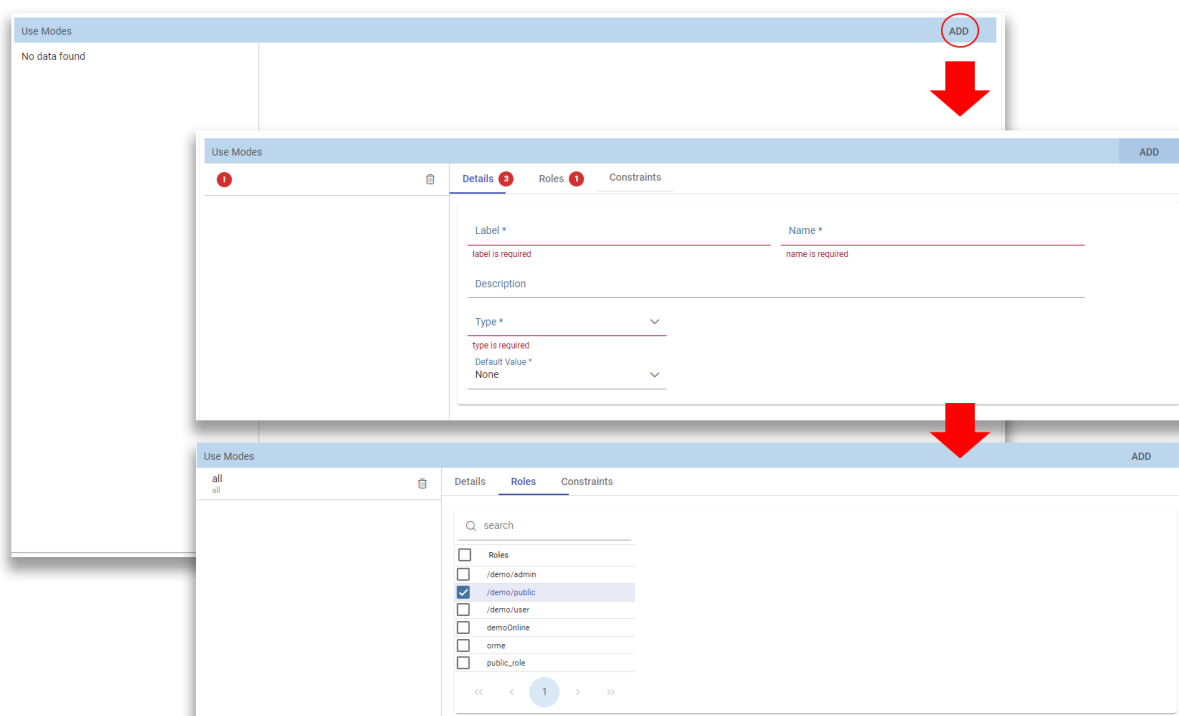


Fig. 1.21: Detail panel of Analytical Driver creation, second step.

The first type allows the user to pick values from a previously defined LOV. When selecting this option the interface spread out the configuration panel where the user is asked to select a LOV from the list and a **Modality**. The latter defines how values are displayed and selectable when executing the document. In fact the user can choose among:

- **Select from list:** all admissible values will be displayed directly within the drivers panel;
- **Select from popup window:** user will be able to select between admissible values by a lookup table displayed within a popup window;
- **Select from tree:** conceived for hierarchical LOVs, lets the users navigate the values in a hierarchical way;
- **Select from combobox:** the driver will look like a drop down menu.

The second kind of input expects the user to type manually the value. Otherwise the third opens a map from which the user must select one or more regions accordingly to the layer property. When selecting this option the interface spread out the configuration panel where the user is asked to choose a layer and the layer property. More details are supplied in next sections for this kind of input.

Moreover the user can add default values (namely values that will be passed to the document at its first execution) using the dedicated area. Here it is possible to pick default values from another LOV or to pick the first or the latter value of the current LOV (if the LOV input type was selected).

In case of Manual Input Date the user can specify a maximum value driven by a LOV:

During execution of a document, the date picker will be limited by that value:

The screenshot shows the 'New' form for creating an Analytical Driver. The form has a header bar with 'New' and a close button. Below the header, there are fields for 'Label *', 'Name *', 'Type *', 'Description', and 'Options'. The 'Type *' dropdown is set to 'Date' and is circled in red. The 'Description' field is empty. Below the form, there is a 'Use Modes' section with a tabbed interface. The 'Details' tab is selected, showing a form with fields for 'Label *', 'Name *', 'Description', 'Type *', 'Default Value *', 'Max Value *', and 'LOV *'. The 'Max Value *' dropdown is set to 'Use a LOV' and is circled in red. The 'LOV *' field contains the text 'cov_prev_sel_da'.

Fig. 1.22: Detail panel of Analytical Driver creation, specification of a maximum value.

The screenshot shows a date picker for August 2022. The calendar displays the days of the month from 1 to 31. The 1st day is highlighted. The calendar has a header with 'August 2022' and navigation arrows. The days of the week are listed as Su, Mo, Tu, We, Th, Fr, Sa.

Su	Mo	Tu	We	Th	Fr	Sa
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3

Fig. 1.23: Detail of a date picker for a date parameter with maximum value specified.

Note: Analytical driver of type Manual Input Date with a default value and/or max value

In the case you want to use an analytical driver of type Manual Input Date with a particular date as default value and/or a maximum value, you have to use a particular syntax for the LOVs query. See the note *Create a LOV for the default value of an analytical driver of type Manual Input Date* in the section *Creating a List Of Value* for more details.

Note: Analytical driver with hierarchical LOV and default LOV

In the case you want to use an analytical driver with a hierarchical LOV and a default LOV the latter need to be hierarchical too. For more details see *Create a LOV for the default value of an analytical driver with a hierarchical LOV* note in the section *Creating a List Of Value*.

At the bottom of the page the user must associate roles to the “use mode”. This action is mandatory. The user connects the user’s roles that he/she wants to be allowed to see a certain list of values or certain regions or be able to type values at his/her convenience.

Therefore, since an admin user can decide to separate values according to the other users’ roles, the analytical driver definition allows to configure different use mode. We can also set validation checks if needed. Then it is sufficient to save each use mode and click on **new use mode** to set a new one. We repeat the same procedure for all the use modes. Each use mode is represented in a separate tab. We will go deeper into this at the end of the section.

All the selections can be multi-valued, but note that this option has to be set directly on the document detail during analytical driver association.

1.1.4.4 Creating an analytical driver for a spatial filter

In previous section we explained how to configure a driver and how it can be linked to different kind of inputs. In this part we linger on the possibility to define a spatial analytical driver. Referring to the following figure, we notice that for setting the geographical driver we must select the **map input** option: here, expanding the combobox you choose the layer on which the filter will act. It is then necessary that the layer has been previously created and uploaded into Knowage **Layers catalog**. Then it is mandatory to specify the property name of the geometry in use using the manual text box just below. Remember that the property name must be exactly the same, therefore respect the upper and the lowercase of the string.

The screenshot displays the 'Use Modes' configuration window. On the left, there is a list of use modes with 'all' selected. The main area shows the configuration for the selected mode. The configuration form has the following fields:

- Label ***: all
- Name ***: all
- Description**: (empty text area)
- Type ***: Map Input (selected from a dropdown)
- Layer ***: usa_states_file (selected from a dropdown)
- Layer Property**: STATE_ABBR
- Default Value ***: None (selected from a dropdown)

Fig. 1.24: Spatial analytical driver settings.

These few steps will implement the spatial analytical driver to be associated to a document and be used to set a spatial filter.

1.1.4.5 Analytical driver's use modes

Sometimes the same analytical driver (i.e., the same concept, like the concept of product brand) should display different values according to the user that is executing it.

Suppose you have a report on sales and costs like the one in the first figure of this chapter and you want to add to it the possibility to filter also on product brands. If you load the report as the general manager, you should choose between all the possible product brands in the corresponding parameter. If instead you load it as, for instance, the food manager, then you should be able to filter only on product brands related to the Food family.

In order to do this, let us focus again on the definition of the LOV and check that the already defined use mode **All Brands** is associated to the correct role **general_manager**. Here you can add a second tab, called for instance **Profiled_Brands**, and associate it to the role **product_manager**. This is because the food manager user has **product_manager** role with profile attribute **pr_family = Food**.

Finally, we choose the second LOV created, the one returning only those brands that belong to a specific family (see the code example in section Parametrizing LOVs). The family is selected by checking the value of the family attribute in the user profile.

Notice that here you can also choose a different type of display mode for the LOV. In other terms, different use modes correspond not only to different LOVs, but also to (possibly) different display mode (pop-up windows, combobox, ...). For instance, you can select a combobox display mode for the **All Brands** use mode and the pop up window display mode for the **Profiled_Brands** use mode.

Once you have saved the LOV, just log out from Knowage and log in with a different user role, i.e. as a general manager, food manager and drink manager. Executing your report on sales and costs you can now notice the differences on the values and on the display mode of the Product Brand parameters according to the different users. Notice that, for food manager and drink manager, the parameters are always displayed as a pop-up window, while for the general manager also the display mode of the parameter varies.

1.1.4.6 Behavioural Model Lineage

It is possible to show a summary of the links between the LOVs, the analytical driver and the documents by selecting **Behavioural Model > Behavioural Model Lineage**.

The entire list of available LOVs, analytical driver and documents appears, as shown in figure below.

By selecting one LOV or Analytical Driver or Documents the other will refresh showing only the elements associated with the selection done. To come back to the original situation click the refresh button on the top right corner.

1.1.5 Analytical Document

The *analytical model* is the core of Knowage Server and covers the whole range of analytical needs, providing many solutions for each analytical area, like reports, cockpits, OLAP documents, KPIs and so on.

The main element of the analytical model is the so called *analytical document*, a word used to group under a common concept all different types of documents that can be developed with Knowage (report, cockpit, etc.) when performing a BI analysis.

In this chapter we describe step by step how to create a new analytical document. There exist many different document types, each with its own peculiarities. Here we provide a generic overview on common features, will focus on available types peculiarities in each dedicated part.

1.1.5.1 Main concepts

The creation and management of analytical documents in Knowage involves different elements:

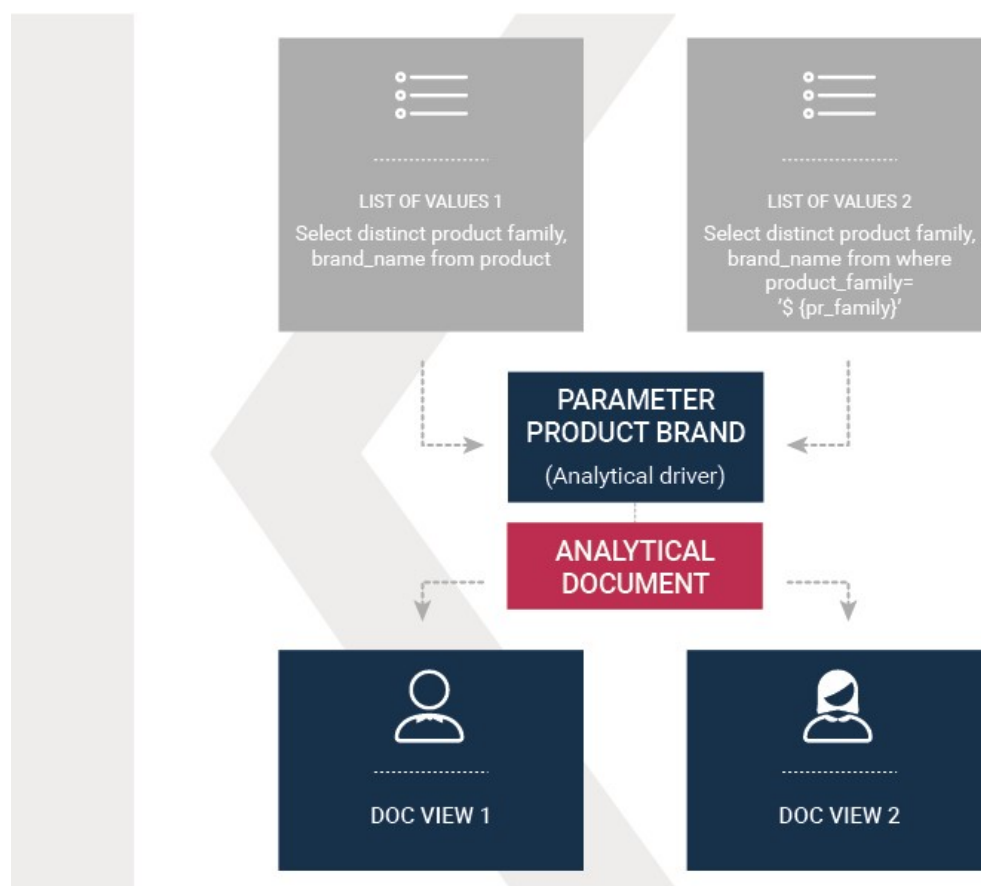


Fig. 1.25: Behavioural Model Schema.

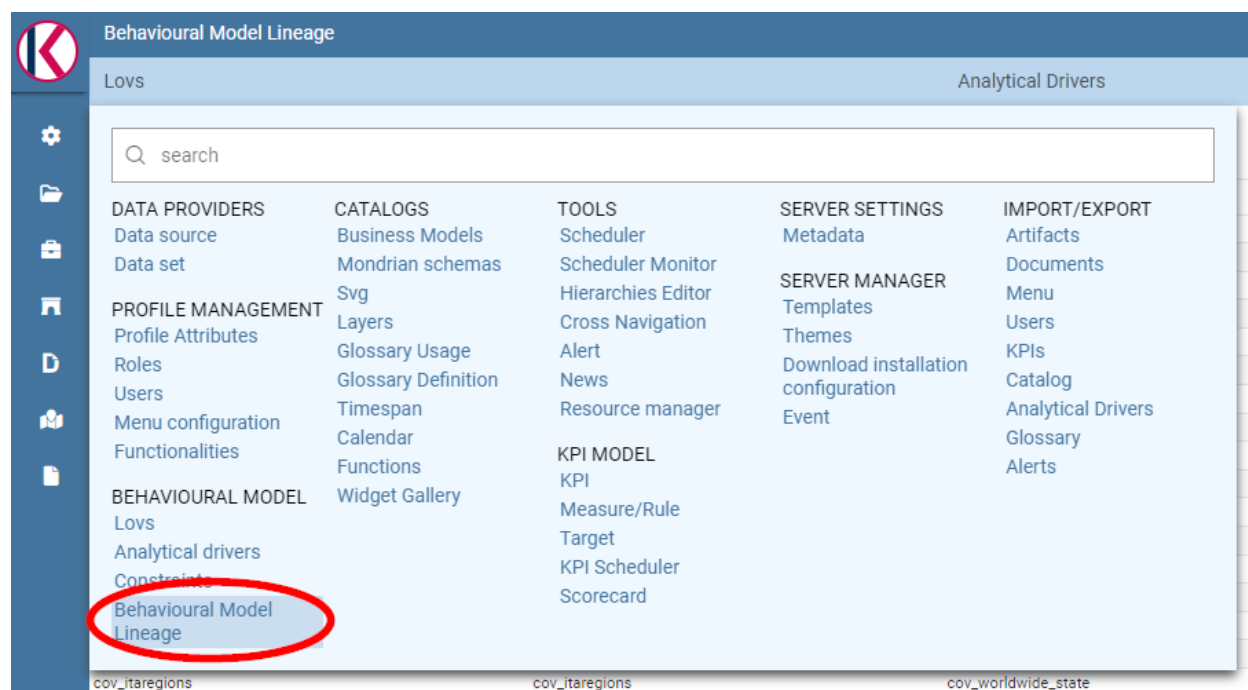


Fig. 1.26: Behavioural Model Lineage.

Behavioural Model Lineage					
LOVs		Analytical Drivers		Documents	
Q search		Q search		Q search	
LABEL T[]	NAME T[]	LABEL T[]	NAME T[]	LABEL T[]	NAME T[]
DEMO_BRAND_NAME	DEMO_BRAND_NAME	manual_number	manual_number	NYC Traffic Accidents	Accident_NYC
DEMO_ProductCategory	DEMO_ProductCat	AD_ProductFamily	AD_ProductFamily	BeerProductSingleP	BeerProductSingleP
DEMO_ProductFamily	DEMO_ProductFamily	manual_string	manual_string	CUSTOMER RETENTION	CUSTOMER RETENTION
Product_BrandAll	Product_BrandAll	AD_ProductFamily (LIST)	AD_ProductFamily_L	Covid19 Worldwide	Covid19 Worldwide
Product_Family	Product_Family	Single_AD_Drink (combo)	Single_AD_Drink_c	Customer_Review	Customer_Review
Single_LOV_Drink	Single_LOV_Drink	Tree	Tree	Customers Survey	Customers Survey
TST_AgeRange	TST_AgeRange	Single_AD_Drink (list)	Single_AD_Drink	Store Sales Analysis	DEMO_Report
TST_AgeRange_Default	TST_AgeRange_Def	manual_date	manual_date	Customers by Occupation	DM_CustOccupation
TST_ProductDepartment	TST_ProdDep	AD_BrandAllPopUp	AD_BrandAll	USA MAP	DM_PromotionMap_File
Tree	Tree	Year	Year	Distribution costs and sales for States and Regions	DM_SIVL_STORE2
Year	Year	DEMO_ProductFamily	DEMO_ProductFamily	HR VIEW	HR VIEW
cov_col_variazione	cov_col_variazione	Manual Input String	Manual Input String	Html	Html
cov_col_incidenza	cov_col_incidenza	cov_taregions	cov_taregions	INVENTORY	INVENTORY
cov_columns	cov_columns	cov_periods	cov_periods	Inventory analysis	Inventory analysis
cov_columns_delta	cov_columns_delta	cov_columns	cov_columns	KPI Inventory Turn	KPI Inventory Turn
cov_columns_worldwide	cov_columns_worldwide	cov_periods_worldwide	cov_periods_worldwide	Kpi card	KPI CARD
cov_taregions	cov_taregions	cov_worldwide_state	cov_worldwide_state	All Kpi	KPI LIST
cov_periods	cov_periods	cov_col_incidenza	cov_col_incidenza	Kpi Gauge	KPI Spedo
cov_periods_worldwide	cov_periods_worldwide	cov_columns_delta	cov_columns_delta	Prima dashboard	My dash
cov_prev_sel_da	cov_prev_sel_da	cov_col_variazione	cov_col_variazione	Olap cube	OLAP SALES
cov_worldstate	cov_worldstate	cov_columns_worldwide	cov_columns_worldwide	PRODUCT ANALYSIS	PRODUCT ANALYSIS
product_tree	product_tree	DEMO_BRAND_NAME	DEMO_BRAND_NAME	Parameters	Parameters
		DEMO_ProductCategory	DEMO_ProductCat	RETAIL INSIDE	RETAIL INSIDE
		TST_AgeRange	TST_AgeRange	SALES	SALES
		AD_Product_Tree	AD_Product_Tree	Sales analysis	Sales analysis

Fig. 1.27: List of LOVs, analytical driver and documents.

Template The template defines the standard layout of a document, including specific information on its appearance and the way contents should be displayed. Templates can be encoded by hand or using Knowage designers, when available. For each analytical document the history of templates is maintained. Old templates can be restored if needed. A new version is saved at each deployment, either manual or from Knowage Studio.

Dataset Each document is associated to one or more datasets. The dataset provides actual data that will be represented according to the defined template. That is to say, the dataset provides the actual content of the analysis and the template is in charge of giving it a meaningful structure.

Data source In order to retrieve data via the dataset, a source of information must be defined. Depending on the type of document, the data source may be associated to the document either directly or implicitly (via the dataset).

Parameters Parameters allow the connection between the document and analytical drivers associated to it. In other words, at document execution time, each driver generates a value that is assigned to the corresponding parameter.

These elements are then combined inside each document engine, in order to produce different analytical documents. This process generates an HTML output, which can be accessed and navigated using a web browser. Other output formats are supported, including XLS, CSV, PDF, XML.

1.1.5.1.1 Document types

Regardless of their type, all analytical documents interact with some generic components (such as cross-services) and with the specific engine that generate them (e.g. Report engine, OLAP engine). Therefore, all analytical documents are managed in the same way in terms of:

- document storage and versioning;
- document life cycle, based on a specific approval process including different status (development, test, released, suspended);
- multiple positioning on the repository and indirectly first visibility level;
- rules to restrict document visibility to some end user profiles;

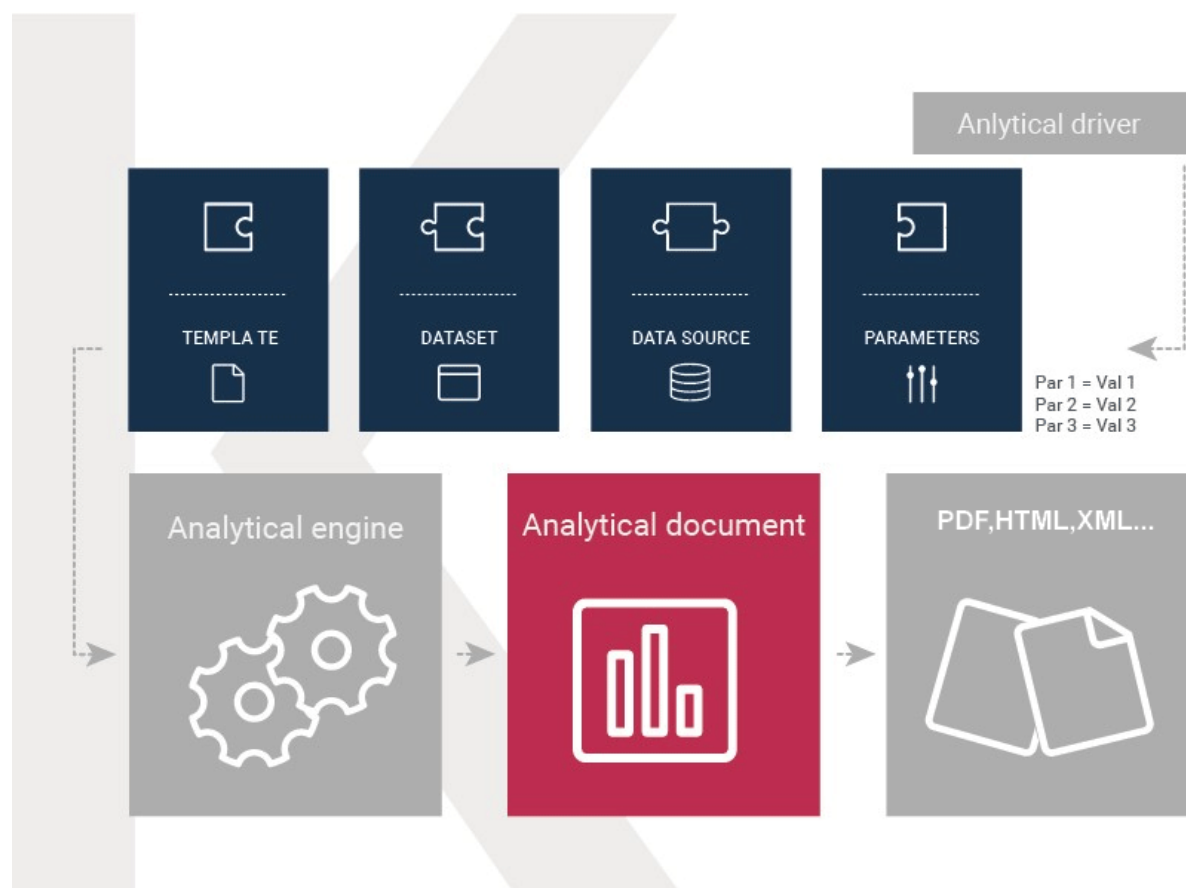


Fig. 1.28: Components of Knowage analytical document.

- management of analytical drivers;
- multi-format export logic;
- attribution of business metadata;
- scheduled execution;
- collection of auditing and monitoring data;
- adding end user notes;
- adding bookmarks;
- end user evaluation;
- sending the document by email;
- on-line or off-line (scheduled) execution.

This means that the above mentioned features are also inherited by every new engine that is developed or integrated into Knowage.

In the next sections we describe in detail how to create and manage analytical documents in Knowage.

1.1.5.2 Register an analytical document

There are two different ways to create a new document on Knowage Server. The first option involves Knowage Studio: within it you can simply click on **Deploy** and the document window will open with pre-filled options. Please note that Knowage Studio can be used to create Birt document only.

Note: Deploy a document from Knowage Studio

Knowage Studio is the tool that allows to design and upload documents onto Knowage Server. Please refer to the dedicated section for full details and examples.

The second option is to manually create the document on the Server. This is the most general way since the Studio designer is not yet available for all documents types.




1.1.5.2.1 Analytical documents on Server

First of all click on **Document Development** from the BI functionalities menu, as shown .

By default the page is divided into two parts, as shown in figure below: in the left side there is the functionality tree representing the folder structure, while on the right you can see the list of all documents contained in the selected folder.

Each line shows the label, the name, the author and the type of the document, while the play button at the end of each row executes the document. Moreover, clicking on a line opens a side panel on the right of the page. Here you can see more metadata information such as the document description, the state and the creation date.

At the top of this side panel you find four button:

-  execute the document;
-  access document details;
-  clone the item;

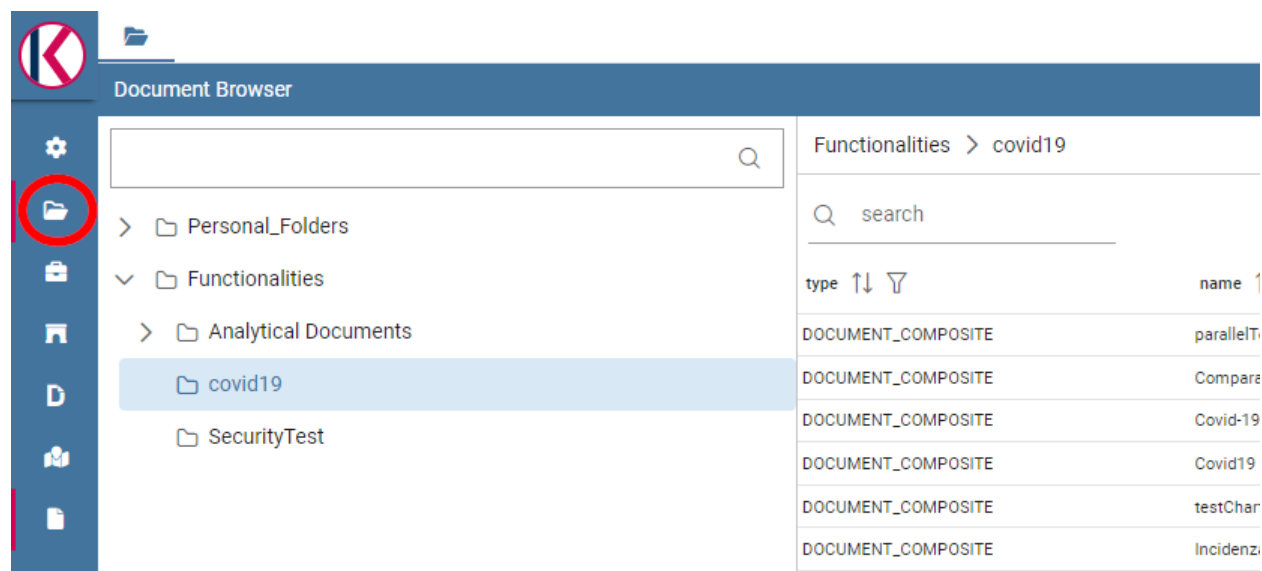


Fig. 1.29: Documents Development button.

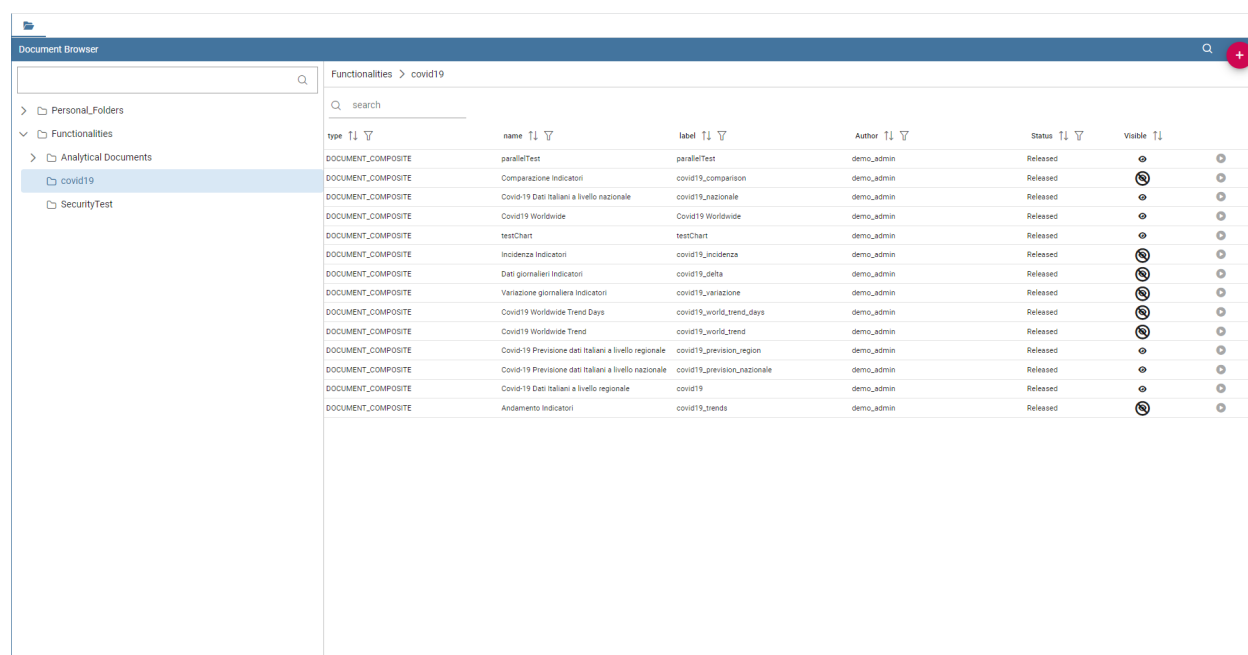


Fig. 1.30: Documents Development section.

Functionalities > covid19							
Q search							
type ↑↓	name ↑↓	label ↑↓	Author ↑↓	Status ↑↓	Visible ↑↓		
DOCUMENT_COMPOSITE	parallelTest	parallelTest	demo_admin	Released			Path /Functionalities/covid19 name Covid19 Worldwide label Covid19 Worldwide Author demo_admin State Released type DOCUMENT_COMPOSITE Creation Date Apr 02, 2020 6:31:53 PM Visibility Visible
DOCUMENT_COMPOSITE	Comparazione Indicatori	covid19_comparison	demo_admin	Released			
DOCUMENT_COMPOSITE	Covid-19 Dati Italiani a livello nazion...	covid19_nazionale	demo_admin	Released			
DOCUMENT_COMPOSITE	Covid19 Worldwide	Covid19 Worldwide	demo_admin	Released			
DOCUMENT_COMPOSITE	testChart	testChart	demo_admin	Released			
DOCUMENT_COMPOSITE	Incidenza Indicatori	covid19_incidenza	demo_admin	Released			
DOCUMENT_COMPOSITE	Dati giornali Indicatori	covid19_delta	demo_admin	Released			
DOCUMENT_COMPOSITE	Variazione giornaliera Indicatori	covid19_variazione	demo_admin	Released			
DOCUMENT_COMPOSITE	Covid19 Worldwide Trend Days	covid19_world_trend_days	demo_admin	Released			
DOCUMENT_COMPOSITE	Covid19 Worldwide Trend	covid19_world_trend	demo_admin	Released			
DOCUMENT_COMPOSITE	Covid-19 Previsione dati Italiani a liv...	covid19_prevision_region	demo_admin	Released			
DOCUMENT_COMPOSITE	Covid-19 Previsione dati Italiani a liv...	covid19_prevision_nazionale	demo_admin	Released			
DOCUMENT_COMPOSITE	Covid-19 Dati Italiani a livello regiona...	covid19	demo_admin	Released			
DOCUMENT_COMPOSITE	Andamento Indicatori	covid19_trends	demo_admin	Released			

Fig. 1.31: Side panel.

- erase the document.

The figure below shows the detail panel of a document. On the left, document details are shown, including name, type, dataset and state. On the right, you can see the functionality tree and the document position. If you want to copy or move a document from a folder into another, check or uncheck the corresponding folders.

Covid19 Worldwide Trend

Document Details

Information

Drivers

Output Parameters

Data Lineage

History

Upload Template

Label *

covid19_world_trend

Name *

Covid19 Worldwide Trend

Description

Preview Image

Type *

Cockpit

Engine *

Cockpit Engine

State *

Released

Refresh (Seconds)

0

Select the number of milliseconds after the document will refresh

Visible

Locked

Position

Parameters Panel Position

Position

Right

Visibility Restrictions

Defined Restrictions

Select Attribute

=

Enter Value

Select The Functionalities Where The Document Will Be Visible

Functionalities

Analytical Documents

Reports

Cockpit

Dynamic Maps

target document (cross navigation)

SvgViewer

Multidimensional Analysis

KPI

covid19

Fig. 1.32: Detail panel of Knowage analytical document.

In order to create a new document you need to click on the red plus button in the top right corner of the **Document Development** page. The different types of documents that you can create are: **Cockpit** and **Generic Document**. Please note that not all of them are available in all Knowage products.

To create a new generic document click the above-mentioned button and select **Generic Document**. You will be shown

30

Chapter 1. How to use Knowage

a window like the one in figure above but with empty fields, in order to allow you to define the document details.

First of all, choose a label, a name and a description. It is important to point out that the label is the unique identifier of the document in Knowage Server. Then, select the type of document and the appropriate engine from the drop down menus, according to the document you are developing (see figure below).

Fig. 1.33: Select Type and Engine for a new document.

Now you have to select the datasource that will feed your document with data.

You can select the data source from the drop down menu. And select the dataset from the pop-up window and click save.

Note that some types of document do not require the definition of a dataset at this point because they use embedded datasets.

It is advisable to regularly save the document in this process, by clicking the related button save at the top right corner of the window.

1.1.5.2.1.1 Document lifecycle

The next step is to choose the status of the document using the **State** drop down menu. At any time in fact, each document is associated to a state, which will typically change over time following the development of the project. Those states are:

- development;
- test;
- released;
- suspended.

Upon creation, the document is by default in development state. Any time you upload a new template or make changes to the document, it is recommended that the state is updated so as to reflect its actual development state.

The screenshot shows the 'Document Details' window in Knowage. The 'Informations' tab is selected, displaying a form with fields for 'Label *', 'Name *', 'Description', 'Type *', and 'Engine *'. A red box highlights the 'Data Source' dropdown menu, which is open and shows a list of options: 'Covid19', 'cache', 'knowage', and 'foodmart_demo'. The 'Position' tab is also visible, showing 'Visibility Restrictions' and a tree view of functionalities.

Fig. 1.34: Selecting a datasource for the document.

The main reason for this is that the state of the document has an impact on its accessibility. As discussed in the behavioural model, Knowage defines role types (administrator, developer, tester, user). States are compatible with the corresponding role type. Administrators can change the state of documents at any time. Developers can not access only the documents with test state. Testers can not see documents in development or suspended state. Users can execute only documents in released state. Note that a tester may change the state of a document from test back to development.

Important: Enterprise Edition only

In KnowageER you may also decide to temporary “lock” a document while he is working with it: it is enough to set the **Lock by user** item. This prevent other developers from modifying the same document you are working on.

1.1.5.2.1.2 Template Versioning

When you register a document on the Server, you need to associate a template to it. Click on tab **History** and then on button **Add** to choose a template from your local file system and upload it.

You may have edited the template by hand or using the Studio designer. Clearly you will not have to upload the template if you are using the automatic deploy from the Studio.

Knowage Server supports versioning of uploaded templates, as shown below. To view them, click on tab **History**. All templates are saved with their date and name, and can be easily uploaded or deleted. To upload a template, click on button **Add** to choose a template from your local file system and upload it: the new template will be uploaded. Using the same list you can download or delete a template.

1.1.5.2.1.3 Document Visibility

After having defined all details, you need to choose where the analytical document shall be saved in the functionality tree. This choice has an impact on the visibility of the document. Since folders in the functionality tree are subject to

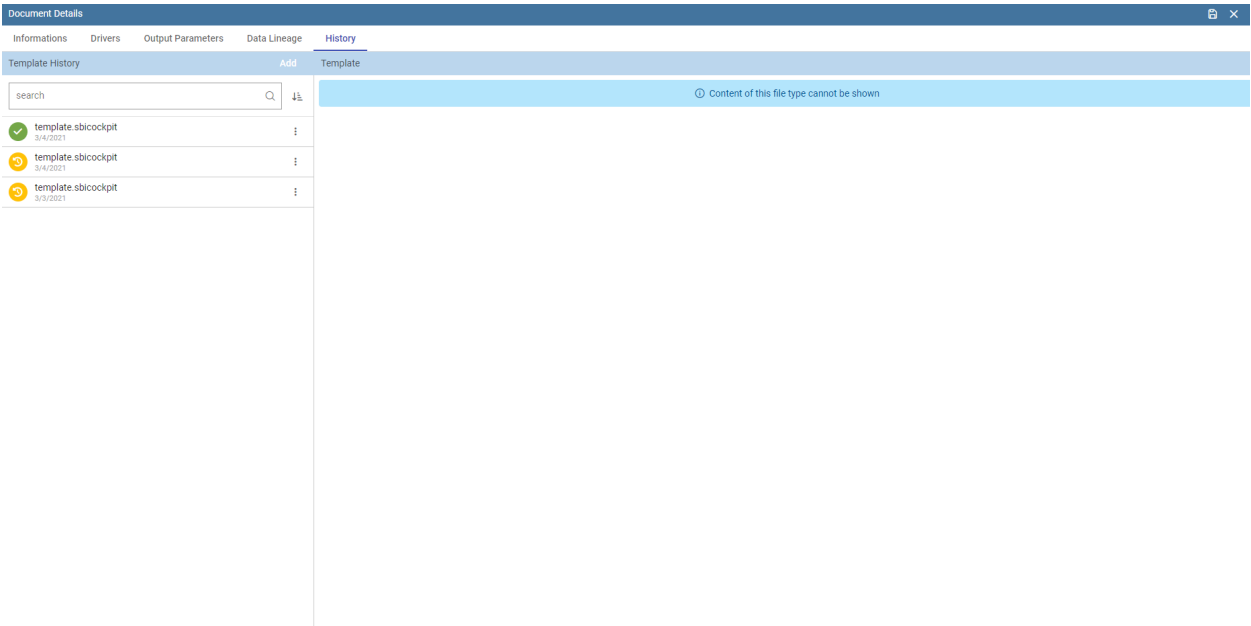
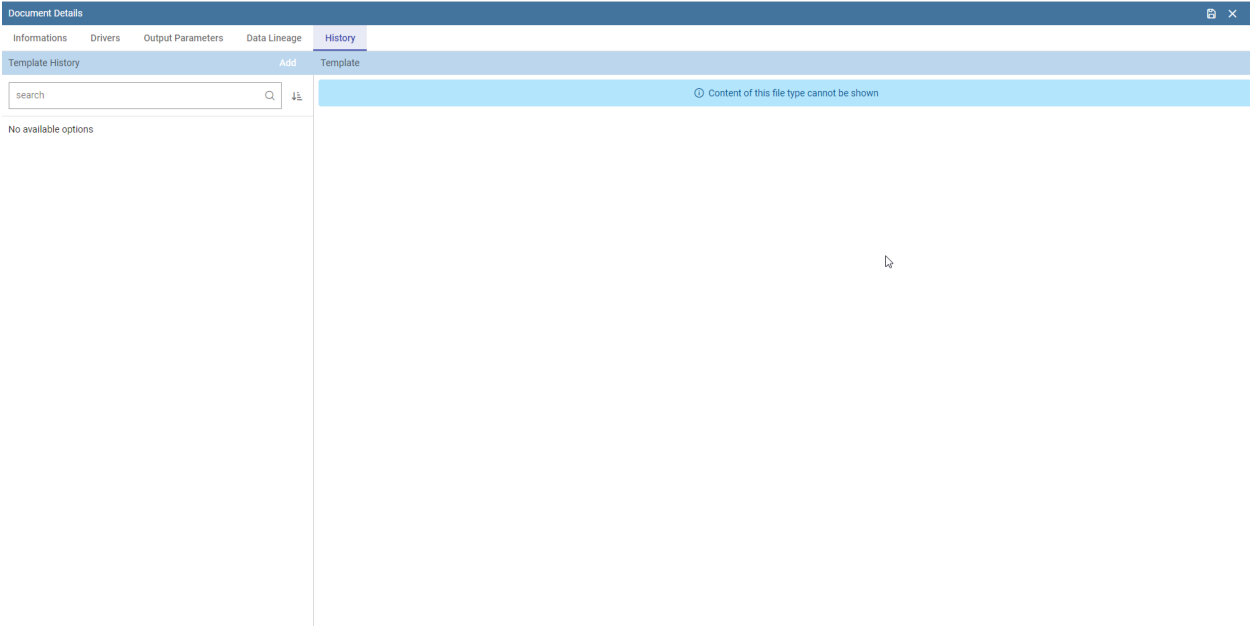


Fig. 1.35: Template versioning for analytical documents.

different access policies, which can be set when creating the node, then each document saved in that folder will inherit permissions accordingly.

Warning: Repository structure and rights

The **Functionalities tree** is Knowage document repository. It is managed by administrator, who is in charge to profile user visibility too.

Note that the same document can be saved in different points of the functionality tree. This allows the administrator to make the document accessible to multiple roles based on visibility rules defined for the containing folder(s). On the right, you can choose where you wish to save the document by ticking the corresponding folder in the tree. If you wish to save it at multiple locations, tick all of them before saving. Each user having access to the containing folder will see the document.

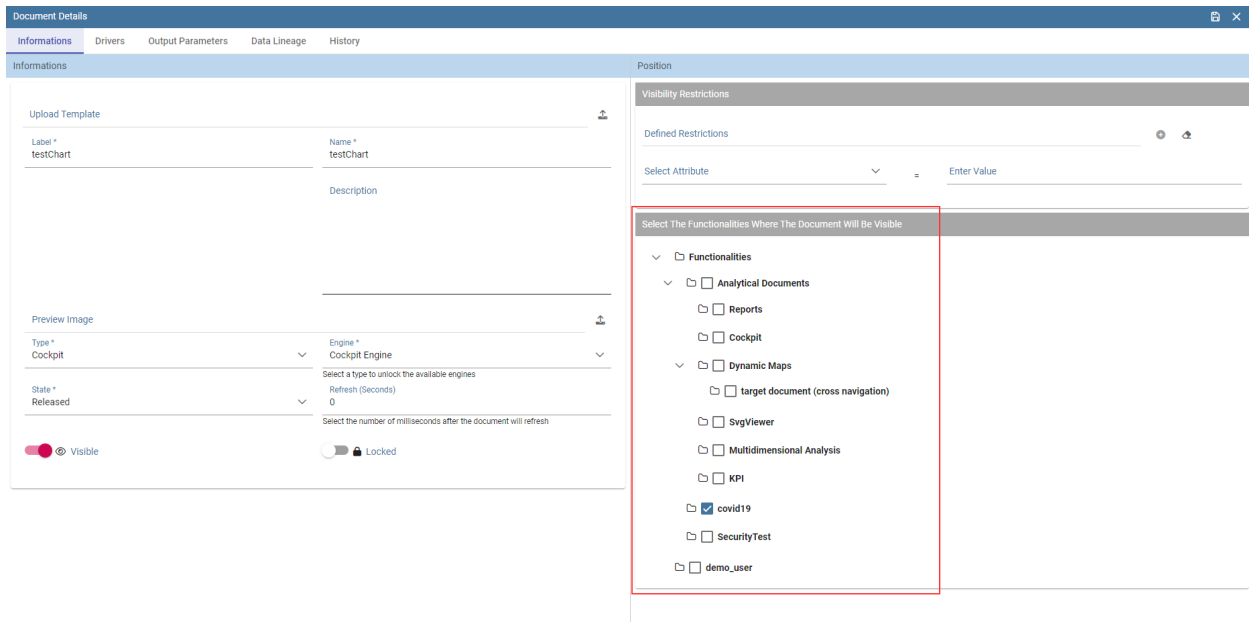


Fig. 1.36: Functionality Tree, document saving settings.

1.1.5.3 Visibility rules

In addition to the standard mechanism supported by the functionalities tree, it is possible to further customize access to a document based on user profile attributes. This allows administrators to rule access to documents at a very fine-grained level, beyond simple repository-based policies.

This can be done by editing conditions in the **Visibility restrictions** section on the right side of **Information** panel. To add a new condition pick a profile attribute from the drop down menu and assign it a value. This will add a new condition that must be verified to allow a user to access the document. In the same way you can add further conditions, and possibly remove all of them by clicking on the eraser.

1.1.5.4 Association with analytical drivers

We have already discussed the role of analytical drivers and how they are connected to analytical documents via parameters. In this section we will show how to practically define this association.

Fig. 1.37: Visibility restrictions.

We assume that the document template and datasets are correctly set in terms of parameter definition. In particular, they should have been correctly referenced with their URL.

To add a new parameter, you can click on the tab **Drivers** and then on a **Add** button, see the next figure.

Fig. 1.38: Association with analytical driver panel.

Choose a name for the title of driver. Then choose analytical driver from drop-down menu that you wish to associate to the document.

Once you have selected the driver, you should write the **exact URL** of the corresponding parameter. Then set the different features associated to the driver: you can set its visibility and decide if it is required and multivalue. By default the parameter is visible, not mandatory and not multivalue.

If you want the document not to be visible to end users, untick the **Visible** checkbox. Note that the parameter will still exist and receive values from the associated driver. However, this will be hidden and the end user will not be able to choose any value for this parameter.

If you want to set it as a required parameter just click on **true**. In this case, no default value is set. The end user will be asked to choose the value of the parameter before opening the document.

Similarly to set a parameter as multivalue click on **true**, in this way the user can perform multiple selections on among its values.

After you have completed the definition of a parameter you can save it by clicking on main **Save** button in the upper right corner. To add further parameters, click on the **Add** button. Repeat the same procedure as many times you want. At this point you may wish to change the order of parameters (i.e., how they are presented to the user). To do so, click on the arrow in the list of drivers.

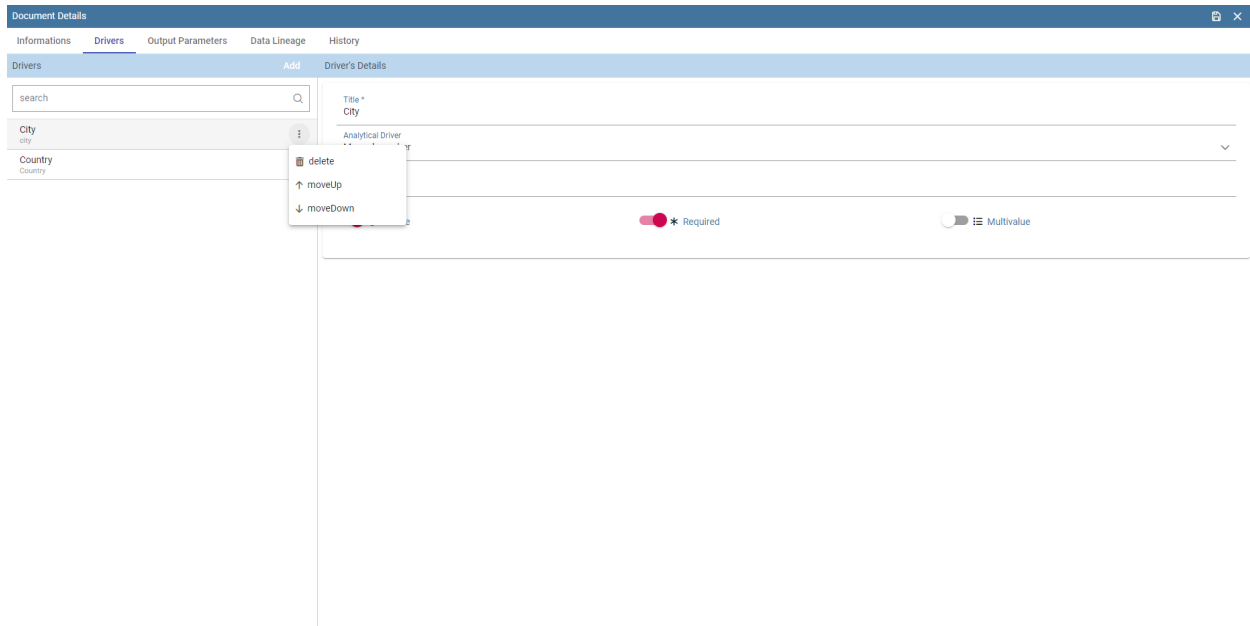


Fig. 1.39: Association with analytical driver panel.

In the following we will see some special operations that can be performed on drivers associated to a document.

1.1.5.4.1 Correlation between parameters

In the context of a document, two different parameters may be connected to each other: this means that the possible values of a parameter are limited by the value(s) of another parameter.

This feature can be useful when two (or more) parameters are logically related. For example, suppose to have a parameter for all the possible countries and another one for all the possible cities. If the user selects a region, it is meaningless to show him all cities: he should only be enabled to choose among the cities in the selected region.

In general, to configure a correlation within a document you should make sure that the LOV associated with the parent parameter and the one associated to the child parameter share at least one column. This column defines which value from the parent parameter will be applied to the child, in order to constrain the results.

To set the correlation, select child parameter which will show you the details of that particular driver and then click on the **Add condition** button to open pop-up window for defining data correlation.

Here you need to define:

- the parent parameter;
- the type of logical operator, in order to compare values of the parent parameter with values of the child parameter;
- the column, generated by the child parameter, whose value will be compared with the value of the same column in the parent parameter.

If a parameter depends on multiple parent parameters, you can define multiple correlations.

Driver Visualization

① Filter operator is used for comparing a column value of the AD

Driver AD Depends From The AD Country	Filter Operator Equal	Logical Operator AND
--	--------------------------	-------------------------

Modality: a

☐ Add correlation with this use mode

LOV's Column
Select LOV's Column

CANCEL SAVE

Fig. 1.40: Definition of the correlation.

Driver Data Conditions Add Condition

No data condition selected

Fig. 1.41: Adding data correlation.

Document Details SAVE CANCEL

INFORMATIONS **DRIVERS** OUTPUT PARAMETERS DATA LINEAGE HISTORY

Drivers

Country	State province	City
countryUrl	provinceUrl	cityUrl

Driver's details

Title *
City

Analytical driver *
City

Url name *
cityUrl

☒ Visible ☒ Required ☐ Multivalue

Driver data conditions ADD CONDITION

equal value of the parameter countryUrl	
equal value of the parameter provinceUrl	

Driver visibility conditions ADD CONDITION

NO VISIBILITY CONDITION SELECTED

Fig. 1.42: Multiple correlations.

Once defined the correlation, the child parameters will display the labels during the runtime in italics.

1.1.5.4.2 Correlation through LOV and drivers

In previous sections we saw how to set correlation through the GUI available in the document detail panel, but there is also the possibility to get the same result using the link between LOV and analytical drivers. More in depth, the user must have previously configured a driver that runs values that can be used in the “where” clause of a SQL query. Then the user must set a query-type LOV using the syntax

We stress that the `AD_name` is the name of the driver the administrator is trying to reach. Syntax for setting correlation through LOV configuration is:

Listing 1.5: Syntax for setting correlation through LOV configuration

```
$P{AD_name}
```

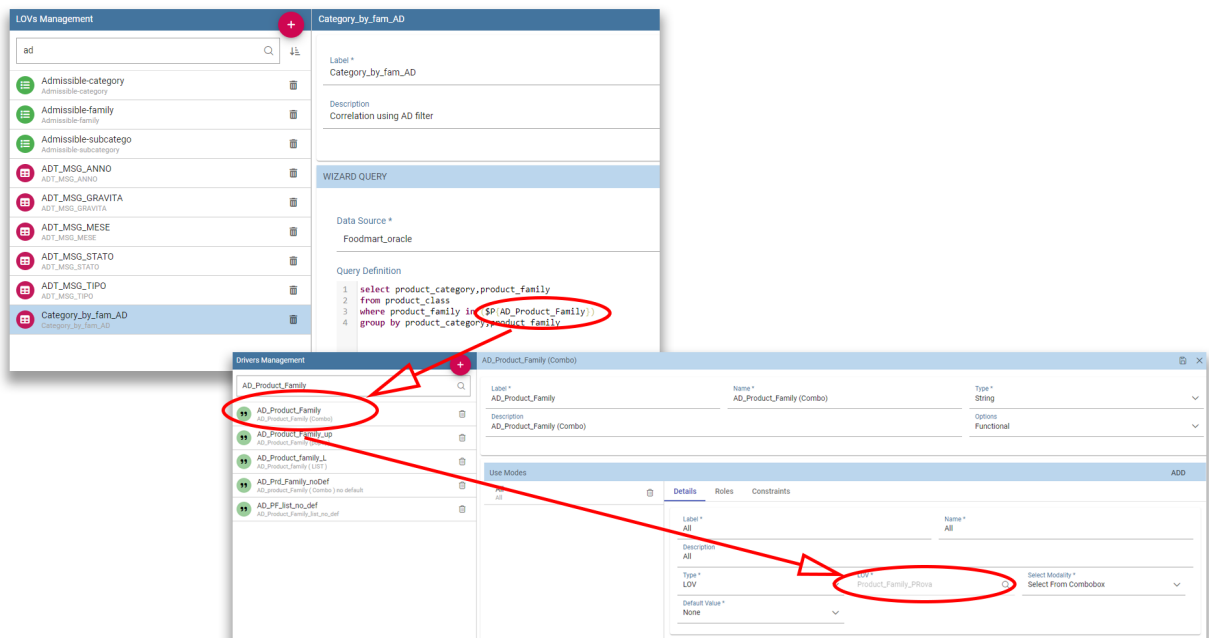


Fig. 1.43: Correlation passing driver values to LOV query .

As a result, at document execution, as soon as the user pick up a value from the “free” parameter, the other one is filtered and will show only the value related to the previous selection, as shown in Figure below.

1.1.5.4.3 Controlled visibility

Another type of relation between parameters is supported by Knowage. It is possible to define values of a parent parameter that force hiding or showing of a child parameter in the parameters mask. Note that in the first case, the child parameter is hidden by default, while in the second case the parameter is shown by default.

To set a visibility expression, click on the **Add condition** button on the **Driver visibility conditions** card.

In the graphical editor you can define visibility rules similarly to correlation ones, as shown in figure below.

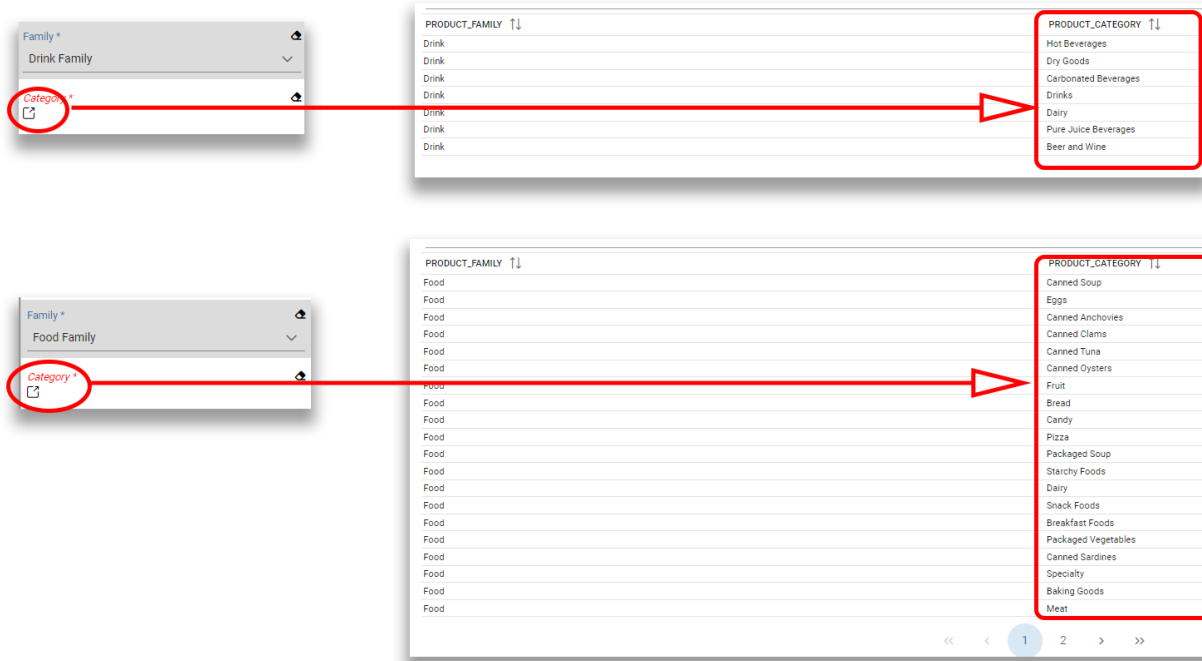


Fig. 1.44: Filtering with correlation.



Fig. 1.45: Adding visual correlation

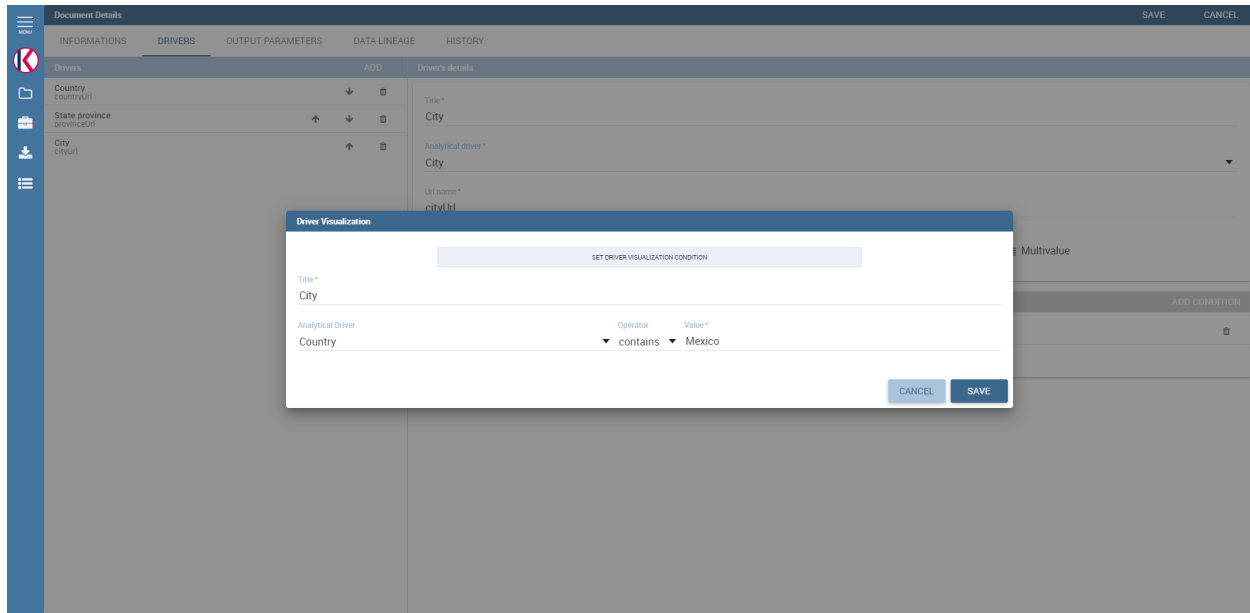


Fig. 1.46: Visibility expressions.

1.1.5.5 Cross Navigation

A powerful feature of Knowage analytical documents is cross-navigation, i.e., the ability to navigate documents in a browser-like fashion following logical data flows. Although crossnavigation is uniformly provided on all documents executed in Knowage Server, each type of document has its own modality to set the link pointing to another document.

Notice that the pointer can reference any Knowage document, regardless of the source document. For example, a BIRT report can point to a chart, a console, a geo or any other analytical document.

In Knowage there are two main typologies of cross navigation: *internal* and *external*.

Internal cross navigation updates one or more areas of a document by clicking on a series, a text, an image or in general on a selected element of the document.

External cross navigation opens another document by clicking on an element of the main document, allowing in this way the definition of a *navigation path* throughout analytical documents (usually, from very general and aggregated information down to the more detailed and specific information)). Indeed, you can add cross navigation also to a document reached by cross navigation. This can be helpful to go deeper into an analysis, since each cross navigation step could be a deeper visualization of the data displayed in the starting document.

It is obviously possible to associate more than one cross navigation to a single document. It means that by clicking on different elements of the same document the user can be directed to different documents.

In this chapter we will examine in depth how to set output/input parameters on documents and, consequently, how to activate the cross navigation.

The first step is to define the parameters of the target document. These do not necessarily coincide with all the filters applied to the document. Please refer to Chapter of Behavioural model for more detail on how to manage parameters and their association to documents.

Therefore it is required to state which parameters among the ones associated to the target document are going to be involved in the navigation. Parameters coming out from the source document are said **output parameters** while the ones that receive values through the association (with the source document) are said **input parameters**. By the way, when declaring the parameters they will be called equally **output parameters** at first, since there is no criterion to distinguish output from input before the navigation is configured.

The definition of the output parameters is performed using the **Manage outputparameters** button but it differs from document to document, according to its type. We will describe these differences in detail in each dedicated chapter, here we explain the common steps.

1.1.5.5.1 Declaration of the output parameters

Enter the **Document details** of the document of interest. Then click on **Output parameters** tab and then on the button **Add** for adding new output parameter.

Here you have to state which parameters are going to be used as output parameters. If, for instance, you select the Date type (see next figure), it is possible to choose the format in which your date has been coded. The default value is related to the location defined in (**Menu > Languages**).

The screenshot shows the 'Document Details' window with the 'Output Parameters' tab selected. On the left, there is a search bar and a list of parameters, currently showing 'Output_param1'. On the right, there is a form to add a new parameter. The 'Parameter Name' field is filled with 'Output_param1' and the 'Parameter Type' dropdown is set to 'NUM'.

Fig. 1.47: Setting an output parameter.

1.1.5.5.2 Cross navigation definition

Finally you need to select the **Cross Navigation Definition** item from the menu to configure the cross navigation. The figure below shows the cross navigation definition window.

The screenshot shows the 'Cross Navigation Definition' window. It has a form with the following fields: 'Name *' (with a dropdown arrow), 'Modality' (set to 'Normal'), 'Description' (with a help icon), 'Text To Show On Bread Crumbs' (with a help icon), 'Origin Doc' (with a 'SELECT' button), 'Target Doc' (with a 'SELECT' button), and two columns for 'Available Input/Output Parameters' and 'Available Input Parameters'.

Fig. 1.48: Cross navigation GUI.

It is required to give a name to the navigation; then select the document from which to start the navigation and the target document. The selecting of a document will cause the loading of input/output parameters related to the starting document in the left column and of the possible input parameters of the target document in the right column.

It is possible to configure the associations between input/output parameters by simply dragging and dropping a parameter from the left column on another of the right column.

Once set, the association is highlighted as in Figure below.

To assign fixed values to target parameters it is necessary to edit first the box labelled **Fixed value parameter** and click on the **plus** icon. Then the value can be associated as fixed value of the one or more target parameters. Remember to click on the **Ok** button to save the cross navigation just set.

As you know, it is possible to define multiple cross navigation starting by the same document. In this case the system will show a popup window to choose the one that you want execute. It is possible set a specific description for each cross navigation so that will be easy to recognize the right navigation definition to use. In the same way it is possible

Name *	CROSSCOCKPIT2	Modality	Normal
Description	Spaccato per Quarter : \${P(quarter)} - Amount: \${P(amount)}	Text To Show On Bread Crumbs	Dettagli per Quarter: \${P(quarter)}
Origin Doc	TC_CROSS_02	Target Doc	TC_PARA_IN
Available Input/Output Parameters		Available Input Parameters	
Fixed Value Parameter	ADD	amount	par4
≡ p1	Output	FIXvalue	par3
≡ quarter	Output	store	par2
≡ store	Output	quarter	PAR1
≡ amount	Output		

Fig. 1.49: Setting the cross navigation.

Name *	CROSSCOCKPIT2	Modality	Normal
Description	Spaccato per Quarter : \${P(quarter)} - Amount: \${P(amount)}	Text To Show On Bread Crumbs	Dettagli per Quarter: \${P(quarter)}
Origin Doc	TC_CROSS_02	Target Doc	TC_PARA_IN
Available Input/Output Parameters		Available Input Parameters	
Fixed Value Parameter	ADD	amount	par4
≡ p1	Output	FIXvalue	par3
≡ quarter	Output	store	par2
≡ store	Output	PAR= quarter	Input
≡ amount	Output		

Fig. 1.50: Relating parameters.

store par2

Fig. 1.51: Association between parameters.

set the text of the bread crumb and personalize it. For both attributes it is possible show parameters (of input or output type) values through the syntax `$P{parameter_name}`. Just parameters of source documents are available.

Description	Text To Show On Bread Crumbs
Data for Quarter : <code>\$P{quarter}</code> - Amount: <code>\$P{amount}</code>	Detail per Quarter: <code>\$P{quarter}</code>

Fig. 1.52: Example of parametric description and breadcrumb text

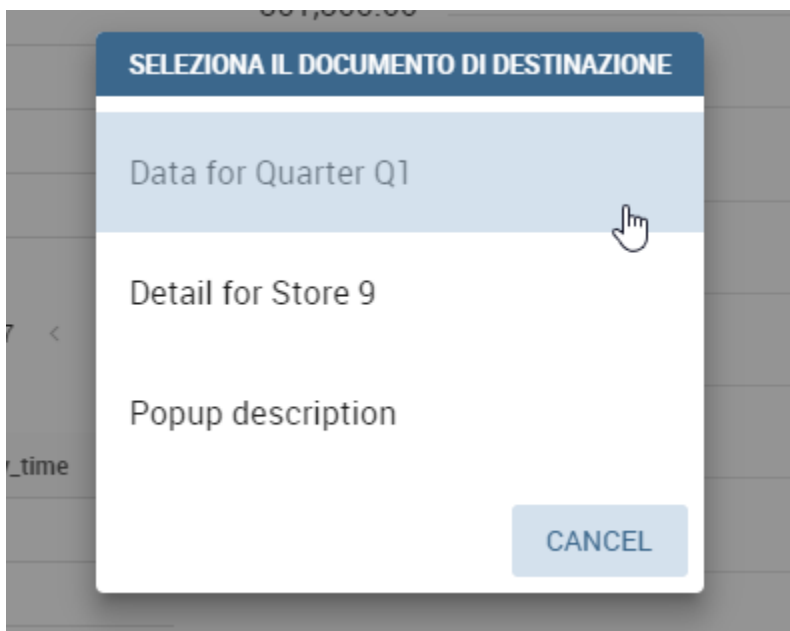


Fig. 1.53: Example of popup selection for more cross navigation definition (with params)

1.1.6 Repository Management

1.1.6.1 Repository structure and rights

Knowage adopts a virtual folder structure to organize analytical documents in hierarchies and folders. This structure is called the Functionalities tree and is accessible via **Profile Management > Functionalities management**.

There are two main reasons for organizing documents into folders and hierarchies: to facilitate their search and accessibility, and to effectively manage visibility on documents according to user roles.

By default *permissions are set at folder level*. This guarantees that a user can not see anything outside that folder (unless he has permissions on other folders as well). It is also possible to further restrict the visibility scope of a user by associating rights to specific values of the profile attributes.

Besides visibility limitations inherited by the containing folders, the developer can add further restrictions to a single document.

To create a new folder, select **Profile Management > Functionalities Management**. The functionality tree is shown. Clicking on an item you can select one of the following options:

- Insert: to add a new child to the tree. Select this to create a new folder and go to the next step.
- Detail: to see details of an item.
- Erase: to delete an item. This option is available only if the folder does not have any children nodes in the tree.

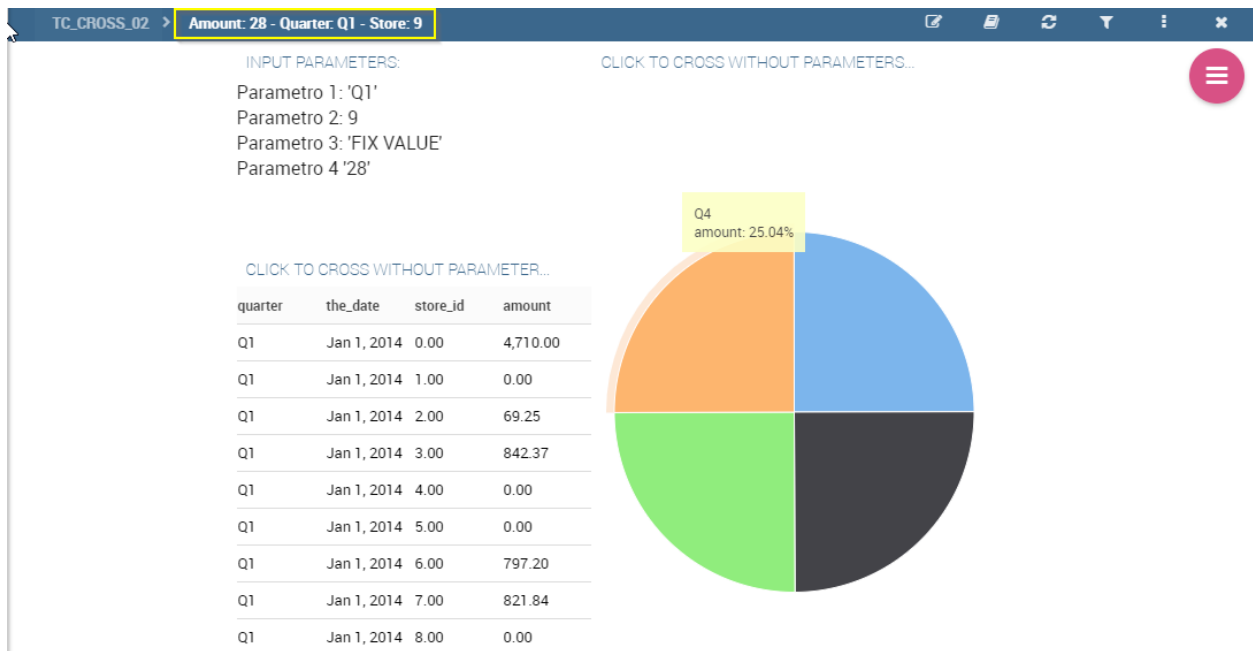


Fig. 1.54: Example of breadcrumb with params

- Move up: to move the item up into the hierarchy.
- Move down: to move the item down into the hierarchy.

Once you select **Insert**, the functionality details opens.

Enter a label and name for the new folder (functionality). In the table, assign permissions to roles. There are four types of permission:

- **Development:** to create, edit and delete analytical documents;
- **Test:** to execute the document and modify its status from test to released;
- **Execution:** to execute the document;
- **Creation:** to create ad-hoc reporting documents like worksheets and cockpits (for the end user).

To assign permissions to roles, check the related boxes. Each user with that role will have permissions on that folder, except in case of specific restrictions on the single document.

Warning:

Permission Inheritance A subfolder inherits the permissions assigned to the parent folder. While it is possible to further restrict inherited permissions, the opposite is not allowed: rights cannot be extended going down the hierarchy.

1.1.7 Menu management

1.1.7.1 Menu configuration

Knowage allows the definition of a menu for the end user. This menu will be displayed in the left bar of Knowage homepage, under the Knowage icon. It is possible to associate to each node a static page, a document, a functionality

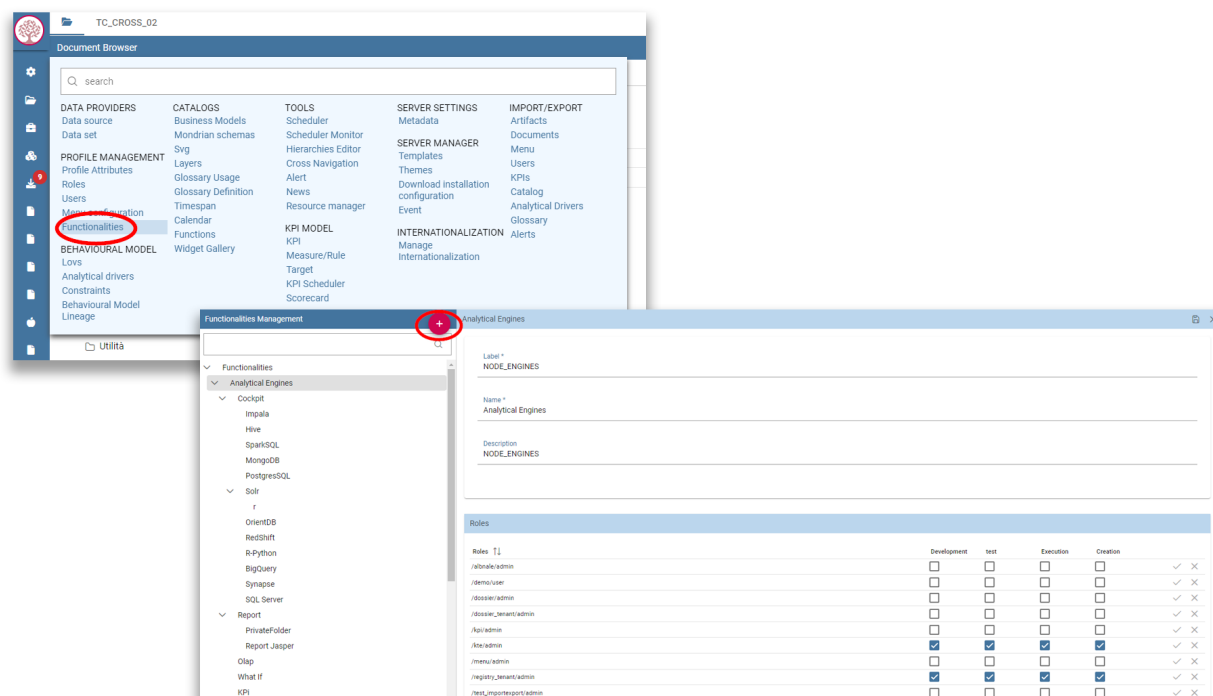


Fig. 1.55: Create a new folder and assign permissions.

(as a folder) or nothing (empty node). Every node can be associated to different roles. This menu structure can be created and modified exclusively by the administrator in the **Tools** area. To access the Menu configuration area, go to **Profile Management > Menu Configuration** from the Main Menu.

It's sufficient to click on the "Plus" of the Menu Configuration page to add a new folder to the **Menu Tree**. When selecting one node of the tree and clicking on the "Plus" icon, the user can add a new folder as a child of the former one.

In general you can:

- define a name: the name is a mandatory field and has to be unique;
- define a description: the description is a mandatory field and it is displayed in the main menu;
- choose an icon to associate to the menu and to be shown in the main menu;
- define the content of the menu item;
- choose the roles eligible to see that particular menu item. Only users associated with the selected roles will see this menu item.

Observe that when one inserts a menu item as a child, this will inherit the general details of a menu node.

1.1.7.1.1 Setting menu icon

Creating a menu, you can choose an icon to be shown in the main menu. **This feature is available only for first level menu.**

To associate an icon to the menu, click on "**CHOOSE ICON**" and then you can:

- choose icon from the predefined icons included in Knowage;

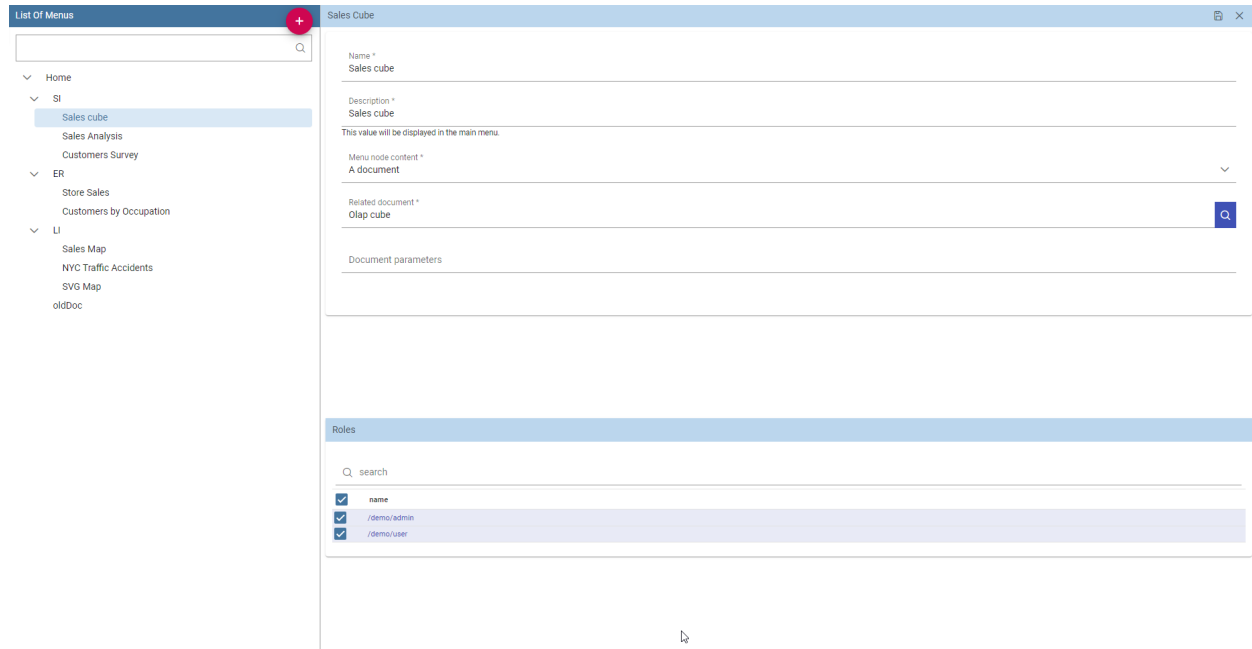


Fig. 1.56: Menu tree actions.



Fig. 1.57: Menu choose icon feature

- upload a new file containing a custom icon (*File type allowed are .ico, .svg, .png and the maximum size allowed is 50 KB*).

To remove the menu icon, click on the trash basket.

Dialog shown in figure below allows you to select an icon or click on “Browse...” to choose the image on your pc and upload it. After choose is completed, click on “**CHOOSE**” button to apply the changes. If no errors occurred, you’ll come back to menu configuration and see the icon you choose. If you want to filter predefined icons, you can start writing in the text field on the top of the dialog.

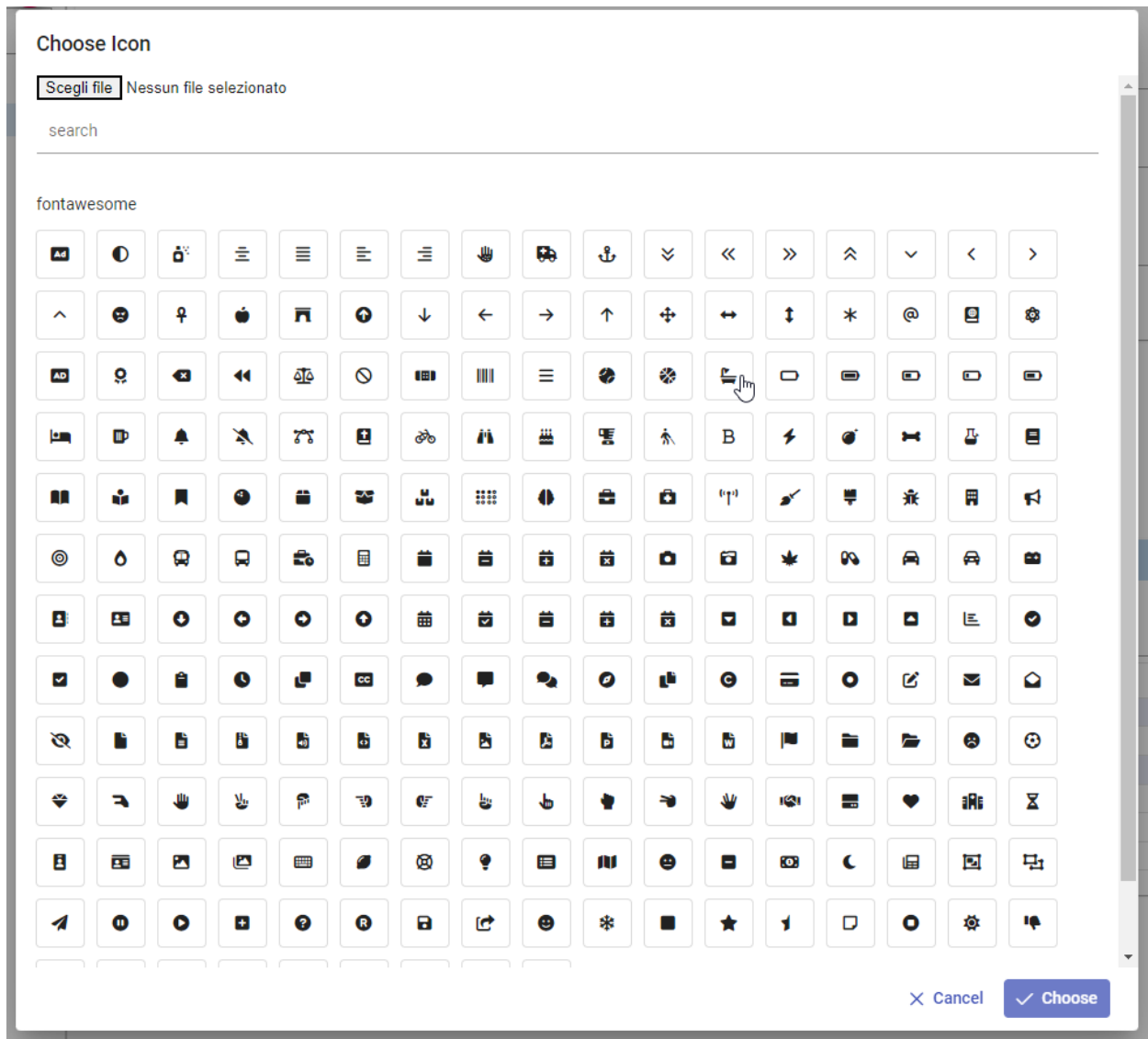


Fig. 1.58: Menu choose icon dialog

If you decide to choose a predefined icon, select it click on “**CHOOSE**” to apply the changes.

If you decide to upload a new image, click on “*Browse...*” and select your file. After that, the icon will be displayed as in the image below.

There are four types of menu item content: empty, document, static page and functionality.

The **empty** content type corresponds to a blank page, and it is typically chosen for father nodes.

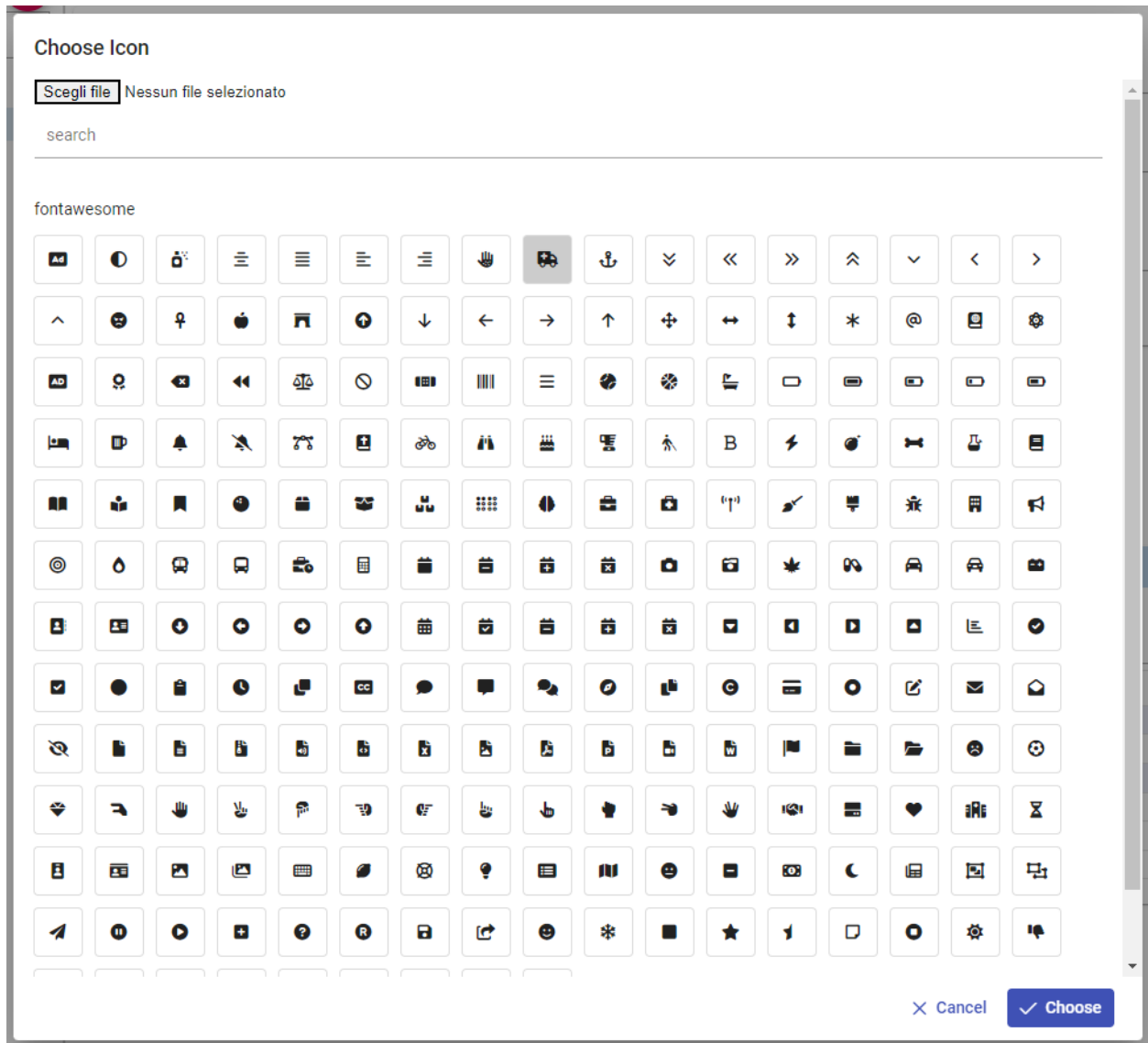


Fig. 1.59: Menu choose selected icon

Fig. 1.60: Menu choose uploaded icon

The **document** content type runs directly a document. For this type you have to choose a related document through the lookup button and define the list of parameters in the standard URL (i.e.: `par1=val1&par2=val2&...`). You can also choose to hide the toolbar or the slider panel.

The **static page** content type shows a static HTML page. In this case, the administrator must define the static page that he wants to load. The HTML page combo is loaded with all HTML pages found in a folder called **static-menu** that must be located under the path defined in the system variable called `knowage_resource_path`.

Finally, the **external application** content type, see Figure below, runs a URL address.

1.2 Create a new Data Source

In order to connect to your data, you have to define a new data source connection.

Knowage manages two types of data source connections:

- connections retrieved as JNDI resources, which are managed by the application server on which Knowage is working. This allows the application server to optimize data access (e.g. by defining connection pools) and thus are the preferred ones. Here you can find information how create connection pool in Tomcat : <https://tomcat.apache.org/tomcat-9.0-doc/jndi-datasource-examples-howto.html>
- direct JDBC connections, which are directly managed by Knowage;

Important: How to create connection JNDI on Tomcat

- Create connection pool on `TOMCAT_HOME/conf/server.xml`
 - Add ResourceLink on `context.xml`
-

To add a new connection, first add the relative JDBC driver to the folder `TOMCAT_HOME/lib` and restart Knowage. Then, login as administrator (user: *biadmin*, password: *biadmin* are the default credential) and select the **Data source** item from the **Data provider** panel in the administrator menu.

By clicking the **Add** button on the top right corner of the left panel, an empty form will be displayed on the right.

The detail page of each data source (on the right side as shown in the figures above) includes the following properties:

Label Mandatory identifier of the data source.

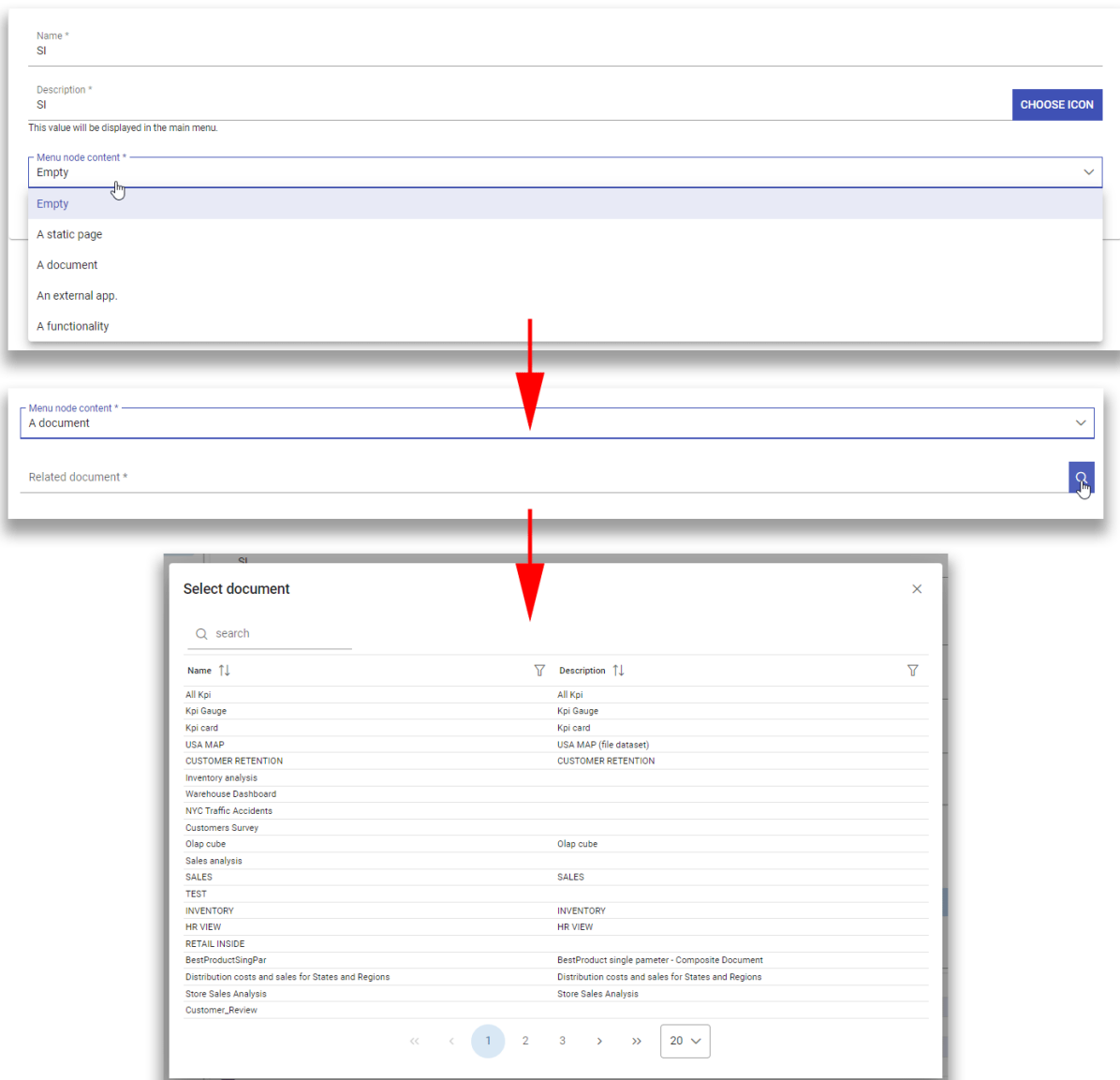


Fig. 1.61: **Empty** (left) and **document** (right) content type.



Fig. 1.62: External application content type.

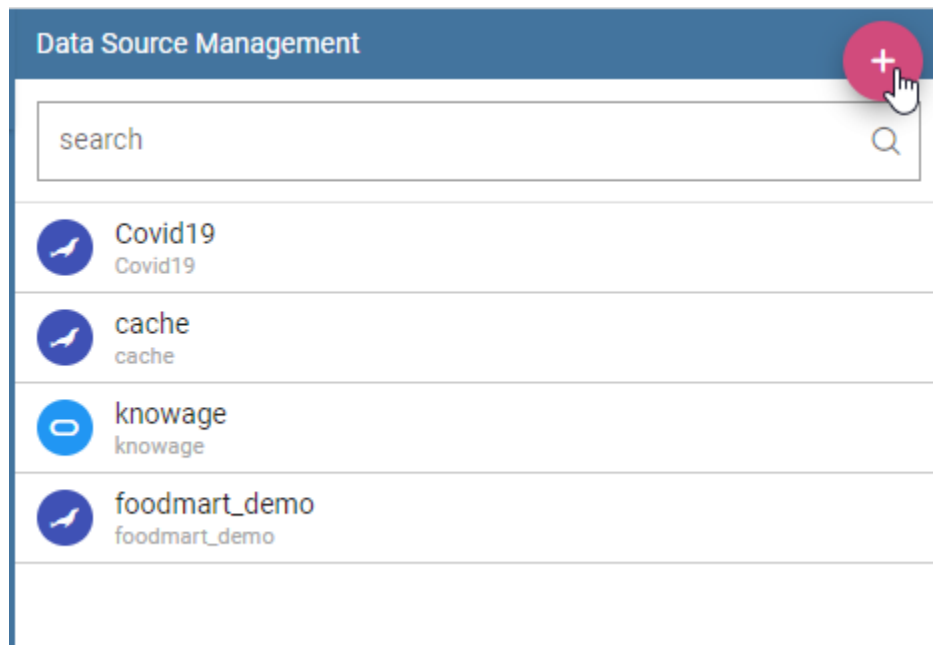


Fig. 1.63: Add a new data source

Name *	Dialect *	▼
Description		
Read Only <input checked="" type="radio"/> Read Only <input type="radio"/> Read and write		
Type <input checked="" type="radio"/> JDBC <input type="radio"/> JNDI		
User	Password (Only If Changed)	
URL *		
Driver *		

Fig. 1.64: Data source details.

Description Description of the data source.

Dialect The dialect used to access the database. Supported dialects are:

Table 1.2: Certified Data Sources

Certified Data Sources	
Oracle	11, 12
MySQL	5.7, 8
PostgreSQL	8.2, 9.1, 12.3
Maria DB	10.1, 10.2, 10.3
Teradata	15.10.0.7
Vertica	9.0.1-0
Cloudera	5.8.9
Apache Hive 1	1.1.0
Apache Hive 2	2.3.2
Apache Impala	2.6.0
Apache Spark SQL	2.3.0
Apache Cassandra	2.1.3
Mongo DB	3.2.9
Orient DB	3.0.2
Google Big Query	•
Amazon RedShift	(JDBC driver v1)
Azure Synapse	•

Read Only Available options are: *Read Only* and *Read-and-write*. In case the data source is defined as read-and-write, it can be used by Knowage to write temporary tables.

Write Default If a data source is set as *Write Default* then it is used by Knowage for writing temporary tables also coming from other *Read Only* data sources. Note that each Knowage installation can have only one *Write Default* data source.

Type The available options are

- If you want to define a direct **JDBC** connection, then you have to also set the following fields:
 - **URL** Database URL. An example for MySQL databases is `jdbc:mysql://localhost:3306/foodmart_key`
 - **User** Database username.
 - **Password** Database password.
 - **Driver** Driver class name. An example for MySQL databases is `com.mysql.jdbc.Driver`.
- If instead you want to define a **JNDI** connection, fill in the following fields:
 - **Multischema** Available options are *Yes* or *No*. If *Yes*, the JNDI resource full name is calculated at runtime by appending a user's profile attribute (specified in the *Multischema attribute* field) to the JNDI base name defined in the `server.xml`, we suppose it has been told at the end of installation or during server configuration.
 - **Schema attribute** The name of the profile attribute that determines the schema name.
 - **JNDI NAME** It depends on the application server. For instance, for Tomcat 7 it has the format `java:comp/env/jdbc/<resource_name>`. If the data source is multischema, then the string is `java:comp/env/jdbc/<prefix>`.

Once you have filled the form, you can test the new data source by clicking on the *Test* button at the top right corner of the page and then save it.

Now you are connected to your data and you can start a new Business Intelligence project with Knowage!

1.2.1 Big Data and NoSQL

In this section we describe how you can connect Knowage to different Big Data data sources.

Important: Enterprise Edition only

Please note that these connections are available for products KnowageBD and KnowagePM only.

1.2.1.1 Hive

Apache Hive is a data warehouse infrastructure built on top of Hadoop for providing data summarization, query, and analysis. Apache Hive supports analysis of large datasets stored in Hadoop's HDFS and compatible file systems such as Amazon S3 filesystem. It provides an SQL-like language called HiveQL with schema on read and transparently converts queries to map/reduce, Apache Tez and Spark. All three execution engines can run in Hadoop YARN.

Every distribution of Hadoop provides its JDBC driver for Hive. We suggest you to use or the Apache one or the one specific of your distribution. In general the JDBC driver for Hive is composed by different .jars, and so you should deploy the JDBC driver with all dependencies in your application server. If you are creating a model you should create a new *Data Source Connection* and import the JDBC driver and all the dependencies.

For example suppose you want to connect to Hive using Apache driver you should include these libraries (according to your Hive version) shown in Figure below.

```
hadoop-core-0.19.0.jar
hive_metastore.jar
hive-common-0.14.0.jar
hive-exec-0.14.0.jar
hive-jdbc-0.14.0.jar
hive-metastore-0.14.0.jar
hive-service-0.14.0.jar
httpclient-4.2.5.jar
httpcore-4.2.5.jar
TCLIServiceClient.jar
libthrift-0.9.0.jar
commons-logging-1.1.3.jar
slf4j-api-1.6.1.jar
slf4j-log4j12-1.6.1.jar
log4j-1.2.16.jar
```

Fig. 1.65: Libraries to include in the apache driver.

If you forget to add one or more libraries, you will likely get a `NoClassDefFoundError` or `ClassNotFoundException`.

The parameters for the Hive connection are:

- **Dialect:** Hive QL;
- **Driver Class:** `org.apache.hive.jdbc.HiveDriver` (if you are not using some specific driver of some distribution. In this case search in the documentation of the distribution);

- **Connection URL:** `jdbc:hive2://<host1>:<port1>,<host2>:<port2>/dbName;sess_var_list?hive_conf_list#hive_var_list`.

Here `<host1>:<port1>,<host2>:<port2>` is a server instance or a comma separated list of server instances to connect to (if dynamic service discovery is enabled). If empty, the embedded server will be used.

A simple example of connection URL is `jdbc:hive2://192.168.0.125:10000`.

1.2.1.2 Spark SQL

Spark SQL reuses the Hive front end and metastore, giving you full compatibility with existing Hive data, queries and UDFs. Simply install it alongside Hive. For the installation of Spark we suggest you to look at the spark website <http://spark.apache.org/>. To create a connection to the Spark SQL Apache Thrift server you should use the same JDBC driver of Hive.

- **Driver Class:** `org.apache.hive.jdbc.HiveDriver` (if you are not using some specific driver of some distro. In this case search in the documentation of the distro);
- **Connection URL:** `jdbc:hive2://<host1>:<port1>,<host2>:<port2>/dbName;sess_var_list?hive_conf_list#hive_var_list`.

Look at the Hive section for the details about parameters. The port in this case is not the port of Hive but the one of Spark SQL thrift server (usually 10001).

1.2.1.3 Impala

Impala (currently an Apache Incubator project) is the open source, analytic MPP database for Apache Hadoop. To create a connection to Impala you should download the jdbc driver from the Cloudera web site and deploy it, with all dependencies, on the application server. The definition of the URL can be different between versions of the driver, please check on the Cloudera web site.

Example parameters for Impala connection are:

- **Dialect:** Hive SQL;
- **Driver Class:** `com.cloudera.impala.jdbc4.Driver`;
- **Connection URL:** `jdbc:impala://dn03:21050/default`.

1.2.1.4 MongoDB

MongoDB is an open-source document database that provides high performance, high availability, and automatic scaling. MongoDB obviates the need for an Object Relational Mapping (ORM) to facilitate development.

MongoDB is different from the other dbs Knowage can handle, because it doesn't provide a JDBC driver, but a Java connector. The MongoDB Java driver (at this moment version 3.5.0 is included) is already included inside Knowage so isn't required to download and add it to the application server.

Example parameters for the connection are:

- **Dialect:** MongoDB;
- **Driver Class:** `mongo`;
- **Connection URL:** `mongodb://localhost:27017/foodmart`(please don't include user and password in this URL).

Also please pay attention that the user must have the correct privileges to access the specified database. So for example on MongoDB you can create a user with this command on the Mongo shell:

Listing 1.6: User creation.

```

1 db.createUser(
2   {
3     user: "user",
4     pwd: "user",
5     roles: [ { role: "readWrite", db: "foodmart" } ]
6   }
7 )

```

Then you must create a role that is able to run functions (this is the way used by Knowage to run the code wrote in the MongoDB's dataset definition) and assign it to the user:

Listing 1.7: Role assignment .

```

1 use admin
2 db.createRole( { role: "executeFunctions", privileges: [ { resource: { anyResource: true }, actions: [
3   ↪ "anyAction" ] } ], roles: [ ] } )
4 use foodmart
5 db.grantRolesToUser("user", [ { role: "executeFunctions", db: "admin" } ])

```

See also this useful links: - (<https://docs.mongodb.com/manual/tutorial/enable-authentication/>)
 - (<https://www.claudiokuenzler.com/blog/555/allow-mongodb-user-execute-command-eval-mongodb-3.x#.W59wiaYzaUI>)

1.2.1.5 Cassandra

Apache Cassandra is an open source distributed database management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. Cassandra offers robust support for clusters spanning multiple datacenters, with asynchronous masterless replication allowing low latency operations for all clients.

There are different ways to connect Knowage to Cassandra.

If you are using DataStax Enterprise you can use Spark SQL connector and query Cassandra using pseudo standard SQL (https://github.com/datastax/spark-cassandra-connector/blob/master/doc/2_loading.md)

Another solution is to download the JDBC Driver suitable for your Cassandra distribution and query Cassandra using the CQL language. You must deploy the JDBC driver with all dependencies in your application server (copy them into TOMCAT_HOME/lib folder and restart).

Refer to the JDBC driver documentation in order to see how to configure the JDBC connection parameters.

Unless you are using Spark SQL to read from Cassandra, the definition of a business model over Cassandra data using Knowage Meta will be available in the next releases.

1.2.1.6 Google Big Query

Knowage supports Google Big Query datasources via Simba JDBC Driver: see [official documentation](#).

For example, to create a JDBC connection to a Google Big Query dataset using a service account you can add the following configuraiton to TOMCAT_HOME/conf/server.xml:

```

<Resource auth="Container" driverClassName="com.simba.googlebigquery.jdbc42.Driver" logAbandoned="true"
↪ maxActive="20" maxIdle="4"
  maxWait="300" minEvictableIdleTimeMillis="60000" name="jdbc/my-bigquery-ds" removeAbandoned="true"
↪ removeAbandonedTimeout="3600"
  testOnReturn="true" testWhileIdle="true" timeBetweenEvictionRunsMillis="10000" type="javax.sql.DataSource"
  url="jdbc:bigquery://https://www.googleapis.com/bigquery/v2:443;ProjectId=<<project-id>>;OAuthType=0;
↪ OAuthServiceAcctEmail=<<service-account-email>>;OAuthPvtKeyPath=<<json-key>>;DefaultDataset=<<dataset>>;FilterTablesOnDefaultDataset=1;"/>

```

1.2.1.7 Google Cloud Spanner

Knowage supports Google Cloud Spanner datasources via the official open source JDBC driver: see [official documentation](#).

For example, to create a JDBC connection to a Google Cloud Spanner dataset using a service account you can add the following configuration to `TOMCAT_HOME/conf/server.xml`:

```
<Resource auth="Container" driverClassName="com.google.cloud.spanner.jdbc.JdbcDriver" logAbandoned="true"
↪maxActive="20" maxIdle="4"
    maxWait="300" minEvictableIdleTimeMillis="60000" name="jdbc/my-spanner-ds" removeAbandoned="true"
↪removeAbandonedTimeout="3600"
    testOnReturn="true" testWhileIdle="true" timeBetweenEvictionRunsMillis="10000" type="javax.sql.DataSource"
    url="jdbc:cloudspanner:/projects/<<project-id>>/instances/<<instance-name>>/databases/<<db-name>>;
↪credentials=${catalina.home}/conf/google-cloud-spanner-auth-key.json"/>
```

1.2.1.8 Amazon RedShift

Knowage supports Amazon RedShift datasources via Official v1 JDBC Driver: see [official reference](#). According to documentation using JDBC drivers v1 a RedShift connection configuration can be done exactly like a PostgreSQL configuration. You can test it creating an example db like this one: [official sample testing db](#). To create a JDBC connection to an Amazon RedShift dataset using a RedShift-only connection you can add the following configuration to `TOMCAT_HOME/conf/server.xml`:

```
<Resource auth="Container" driverClassName="com.amazon.redshift.jdbc.Driver" logAbandoned="true" maxActive="10
↪maxIdle="1" minEvictableIdleTimeMillis="60000" name="jdbc/redshift" password="password" removeAbandoned=
↪"true" removeAbandonedTimeout="3600" testOnReturn="true" testWhileIdle="true" timeBetweenEvictionRunsMillis=
↪"10000" type="javax.sql.DataSource" url="jdbc:redshift://examplecluster.abc123xyz789.us-west-2.redshift.
↪amazonaws.com:5439/dev" username="user" validationQuery="SELECT 1"/>
```

1.2.1.9 Azure Synapse

Knowage supports connections to Azure Synapse datasources via SQL Server JDBC Driver ([official documentation](#)).

The following example shows how to create a JDBC connection to an Azure Synapse dataset, by adding the following configuration to `TOMCAT_HOME/conf/server.xml`:

```
<Resource auth="Container" driverClassName="com.microsoft.sqlserver.jdbc.SQLServerDriver" logAbandoned="true"
↪maxIdle="4" maxTotal="50" maxWait="-1"
    minEvictableIdleTimeMillis="60000" removeAbandoned="true" removeAbandonedTimeout="3600" testOnReturn=
↪"true" testWhileIdle="true"
    timeBetweenEvictionRunsMillis="10000" type="javax.sql.DataSource" name="jdbc/synapse" username="<user>
↪password="<password>"
    url="jdbc:sqlserver://your-synapse-instance.sql.azuresynapse.net:1433;database=<database>"
↪validationQuery="select 1"/>
```

1.3 Create a new Data set

A dataset allows you to read data from multiple sources and represents the portion of data used by the analytical documents.

1.3.1 SQL Query Data Set

Selecting the query option requires the BI developer to write an SQL statement to retrieve data.

Remember that the SQL dialect depends on the data source that has been chosen. The SQL text must be written in the Query text area. Below a SQL query example.

Listing 1.8: SQL query example

```

1 SELECT p.media_type as MEDIA, sum(s.store_sales) as SALES
2 FROM sales_fact_1998 s
3 JOIN promotion p on s.promotion_id=p.promotion_id
4 GROUP BY p.media_type

```

It is also possible to dynamically change the original text of the query at runtime. This can be done by defining a script (Groovy or JavaScript) and associating it to the query. Click on the **Edit Script** section (see next figure) to open the script editor and write your script. The base query is bounded to the execution context of the script (variable query) together with its parameters (variable parameters) and all the profile attributes of the user that executes the dataset (variable attributes).

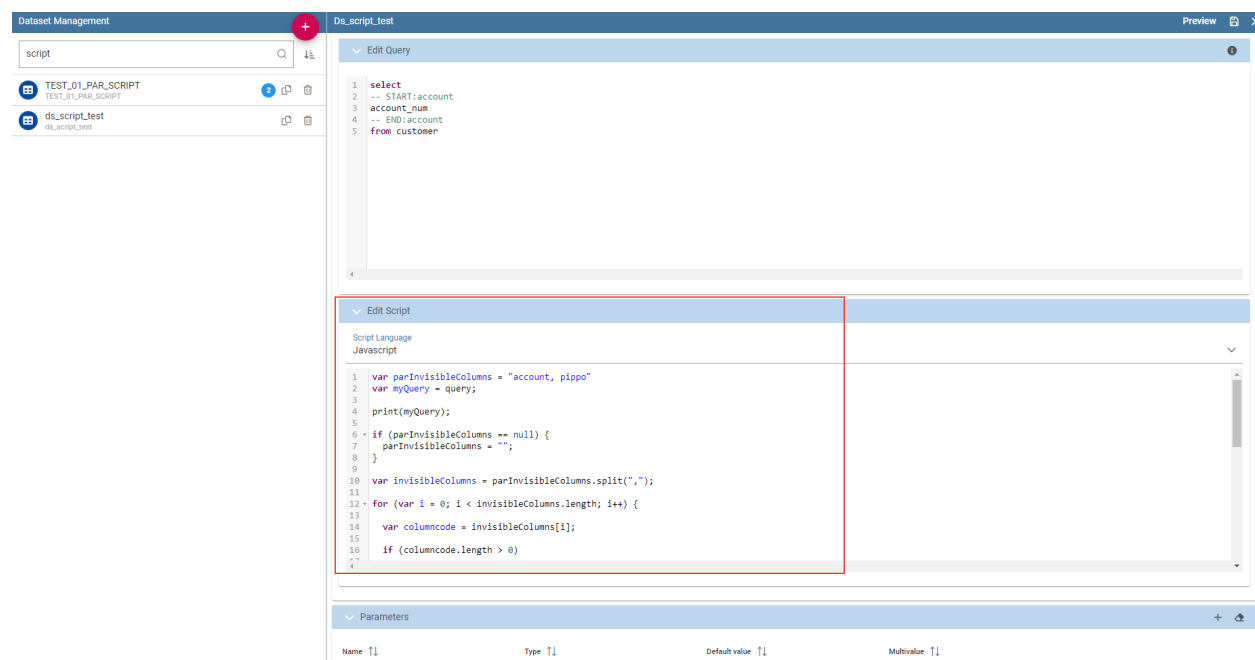


Fig. 1.66: Script editing for dataset.

In the script below we use JavaScript to dynamically modify the FROM clause of the original query according to the value of the parameter *year* selected at runtime by the user. You can notice that the variable *query*, provided by Knowage, contains the text of the SQL query.

Listing 1.9: Example of a script in a *Query* dataset

```

1 if(isValid('year')) {
2     if( parameters.get('year') == 1997 ) {
3         query = query.replace('FROM sales_fact_1998', 'FROM sales_fact_1997');
4     } else {
5         query = query; // do nothing
6     }
7 }

```

In the above example we use the function *isValid* to test whether the parameter *year* contains a *null* value or not.

1.3.2 File Data Set

A dataset of type File, see the following figure, reads data from an XLS or CSV file. To define a **File Dataset** select the File type, then upload the file by browsing in your personal folders and set the proper options for parsing it.



Fig. 1.67: File Dataset.

Once you have uploaded the file, you can check and define the metadata (measure or attribute) of each column.

1.3.3 Flat Data Set

A flat dataset allows the retrieval of an entire table from a data source. In other words, it replaces a dummy query like `select * from sales` by automatically retrieving all rows in a table. To create a flat dataset, simply enter the table and the data source name, as shown below.

1.3.4 REST Data Set

The REST dataset enables Knowage to retrieve data from external REST services. The developer of the dataset is free to define the body, method, headers and parameters of the request; then he has to specify how to read data from the service response using JSON Path expressions (at the moment no other ways to read data is available, therefore the REST service is presumed to return data in JSON format).

Let's make an example in order to understand how it works. Suppose an external REST service providing data from sensors, we want to retrieve values from prosumers electricity meters, a prosumer being a producer/consumer of electricity, and that the request body should be something like:

Listing 1.10: Request body code

```

1 { "entities": [ {
2   "isPattern": "true",
3   "id": ".*",
4   "type": "Meter"
5 } ]
6 }
```

while querying for `Meter` entities, and that the JSON response is something like:

Listing 1.11: RJSON response code

```

1 {
2   "contextResponses": [
3     {
4       "contextElement": {
5         "id": "pros6_Meter",
6         "type": "Meter",
```

(continues on next page)

Inventory_flat Preview ×

Detail **Type** Metadata Advanced

Data Set Type *
Flat

Table Name *
Inventory

Data Source *
foodmart_demo

Inventory_flat CLOSE

the_month	month_of_year	the_year	product_family	product_department	product_category	product_subcategory	brand_name	warehouse_city	warehouse_state_province	warehouse_country	warehouse_name	store_city	Units_Ordered	Units_In_Stock
string	string	int	string	string	string	string	string	string	string	string	string	string	float	float
January	01	2014	Drink	Beverages	Pure Juice Beverages	Juice	Washington	Los Angeles	CA	USA	Antesia Warehousing, Inc.	San Francisco	94	94
May	05	2015	Drink	Beverages	Pure Juice Beverages	Juice	Washington	Camacho	Zacatecas	Mexico	Anderson Warehousing	Springfield	32	32
May	05	2014	Drink	Beverages	Drinks	Flavored Drinks	Washington	San Diego	CA	USA	Jorgensen Service Storage	San Diego	112	112
January	01	2015	Drink	Beverages	Drinks	Flavored Drinks	Washington	Camacho	Zacatecas	Mexico	Anderson Warehousing	Springfield	70	70
June	06	2015	Drink	Beverages	Drinks	Flavored Drinks	Washington	Vancouver	BC	Canada	Bellmont Distributing	Annapolis	35	35
July	07	2015	Drink	Beverages	Drinks	Flavored Drinks	Washington	Portland	OR	USA	Quality Distribution, Inc.	Trenton	88	88
August	08	2015	Drink	Beverages	Drinks	Flavored Drinks	Washington	Portland	OR	USA	Quality Distribution, Inc.	Trenton	31	31
October	10	2014	Drink	Beverages	Drinks	Flavored Drinks	Washington	Salem	OR	USA	Treehouse Distribution	Charleston	14	14
December	12	2014	Drink	Beverages	Drinks	Flavored Drinks	Washington	Spokane	WA	USA	Jones International	Phoenix	74	74
March	03	2015	Drink	Beverages	Drinks	Flavored Drinks	Washington	Hidalgo	Zacatecas	Mexico	Worthington Food Products	Jefferson City	77	77
April	04	2015	Drink	Beverages	Drinks	Flavored Drinks	Washington	San Andres	DF	Mexico	Derby and Hunt	Denver	55	55
July	07	2015	Drink	Beverages	Drinks	Flavored Drinks	Washington	Acapulco	Guanajuato	Mexico	Salka Warehousing	Austin	51	50
July	07	2015	Drink	Beverages	Drinks	Flavored Drinks	Washington	Orizaba	Veracruz	Mexico	Jamison, Inc.	Raleigh	72	72
September	09	2015	Drink	Beverages	Drinks	Flavored Drinks	Washington	Camacho	Zacatecas	Mexico	Anderson Warehousing	Springfield	74	74
April	04	2014	Drink	Beverages	Carbonated Beverages	Soda	Washington	Bremerton	WA	USA	Destination, Inc.	Tallahassee	15	15

Showing 1 to 15 of 10397 << < 1 2 3 4 5 > >>

Fig. 1.68: Flat Dataset.

(continued from previous page)

```

7      "isPattern": "false",
8      "attributes": [
9          {
10             "name": "atTime",
11             "type": "timestamp",
12             "value": "2015-07-21T14:49:46.968+0200"
13         },
14         {
15             "name": "downstreamActivePower",
16             "type": "double",
17             "value": "3.8"
18         },
19         {
20             "name": "prosumerId",
21             "type": "string",
22             "value": "pros3"
23         },
24         {
25             "name": "unitOfMeasurement",
26             "type": "string",
27             "value": "kW"
28         },
29         {
30             "name": "upstreamActivePower",
31             "type": "double",
32             "value": "3.97"
33         }
34     ],
35 },
36 "statusCode": {
37     "reasonPhrase": "OK",
38     "code": "200"
39 },
40 {
41     {
42         "contextElement": {
43             "id": "pros5_Meter",
44             "type": "Meter",
45             "isPattern": "false",
46             "attributes": [
47                 {
48                     "name": "atTime",
49                     "type": "timestamp",
50                     "value": "2015-08-09T20:29:45.698+0200"
51                 },
52                 {
53                     "name": "downstreamActivePower",
54                     "type": "double",
55                     "value": "1.8"
56                 },
57                 {
58                     "name": "prosumerId",
59                     "type": "string",
60                     "value": "pros5"
61                 },
62                 {
63                     "name": "unitOfMeasurement",
64                     "type": "string",
65                     "value": "kW"
66                 },
67                 {
68                     "name": "upstreamActivePower",
69                     "type": "double",
70                     "value": "0"
71                 }
72             ]

```

(continues on next page)

(continued from previous page)

```
73     },
74     "statusCode": {
75       "reasonPhrase": "OK",
76       "code": "200"
77     }
78   }
79   ]
80 }
```

In this example we have two **Context Elements** with the following attributes:

- **atTime** ;
- **downstreamActivePower**;
- **prosumerId**;
- **unitOfMeasurement**;
- **upstreamActivePower**.

Let’s see how to define a Knowage dataset:

DetailTypeMetadataAdvanced

DataSet Type *
REST

Address *
http://localhost:8080/knowage/restful-services/2.0/datasets

Collection

Request Body

HTTP Methods *
Get

Request Headers

JSON Path Items
\$

Use directly JSON Attributes: ☐

NGSI: ☐

JSON Path Attributes

Offset Param

Fetch Size Param

Max Results Param

Parameters

Name	Type	Default value	Multivalued
Click on the + icon to insert data.			

Fig. 1.69: REST dataset interface.

We specified

- the URL of the REST service;
- the request body;
- the request headers (in this example we ask the service for JSON data);
- the HTTP method;

- the JSONPath to retrieve the items (see below), i.e. the JSONPath where the items are stored;
- the JSONPaths to retrieve the attributes (see below), i.e. the JSONPaths useful to retrieve the attributes of the items we are looking for; those paths are relative to the “JSON Path items”;
- offset, fetch size and max results parameters, in case the REST service has pagination.

Once followed the steps above the user obtains upstream/downstream active power for each prosumer.

NGSI checkbox is specific for NGSI REST calls: it permits easy the job when querying the Orion Context Broker (<https://github.com/telefonicaid/fiware-orion>) and to omit some of the REST fields (since the JSON format from NGSI specifications is fixed): you don't need to specify headers, JSONPath items, JSONPath attributes (all available attributes are fetched) and pagination parameters (offset and fetch size).

When checking the **Use directly JSON attributes** checkbox, you can skip the definition of the JSONPath attributes, since the JSON structure is presumed to be fixed as in the following example:

Listing 1.12: Use directly JSON attributes

```

1 {
2   "contextResponses": [
3     {
4       "prosumerId": "pros1",
5       "downstreamActivePower": 3.1,
6       "upstreamActivePower": 0.0
7     }, {
8       "prosumerId": "pros2",
9       "downstreamActivePower": 0.5,
10      "upstreamActivePower": 2.4
11     }
12   ]
13 }
```

Then it will be enough to define only the **JSON Path Items** and check **Use directly JSON Attributes** without defining the attributes; the attributes will be retrieved automatically from the JSON object.

In the above examples, the JSON Path Items will be \$.contextResponses[:sub: `*\`] and the dataset result will look like:

Table 1.3: Dataset result

prosumerId	downstreamActivePower	upstreamActivePower
pros1	3.1	0.0
pros2	0.5	2.4

The REST dataset permits usage of profile attributes and parameters using the same syntax as for other dataset types: \$<profile attribute> and \$P<parameter>. You can use both of them as placeholders in every field: most likely you need to use them in REST service URL or on the request body. As an example, suppose you want to retrieve the value of just one prosumer that is specified by the prosumerId parameter, you have to set the request body as:

Listing 1.13: Request body for prosumerId parameter

```

1 {
2   "entities": [
3     {
4       "isPattern": "true",
5       "type": "Meter",
6       "id": "$P{prosumerId}"
7     }
8   ]
9 }
```

1.3.5 Solr Data Set

A dataset of type Solr, see the following figure, reads data from the popular Search Engine Solr. To define a **Solr Dataset** select the Solr type and fill in the required settings.

Fig. 1.70: Solr Dataset configuration.

The **Query** field is the Solr query using the Solr standard query syntax. The **Collection** field is the **core**, in Solr, the term core is used to refer to a single index and associated transaction log and configuration files (including the solrconfig.xml and Schema files, among others). Your Solr installation can have multiple cores if needed, which allows you to index data with different structures in the same server, and maintain more control over how your data is presented to different audiences. In SolrCloud mode you will be more familiar with the term collection. Behind the scenes a collection consists of one or more cores.

Documents

According to the Solr official documentation, Solr's basic unit of information is a document, which is a set of data that describes something. A recipe document would contain the ingredients, the instructions, the preparation time, the cooking time, the tools needed, and so on. A document about a person, for example, might contain the person's name, biography, favorite color, and shoe size. A document about a book could contain the title, author, year of publication, number of pages, and so on.

In the Solr universe, documents are composed of fields (these fields can be put into section **document field list**), which are more specific pieces of information. Shoe size could be a field. First name and last name could be fields.

Request header, if there is the need, it is possible to customize the request header of the post http request, adding optional parameters.

Fig. 1.71: Solr Dataset, Optional fields for filtering parameters.

Solr dataset can also use Profile Attributes. The syntax to include attributes into the dataset text is `${attribute_name}`. Profile attributes can be single-value or multivalue.

The **filter query parameter** is the Solr fq parameter and defines a query that can be used to restrict the superset of documents that can be returned, without influencing score. It can be very useful for speeding up complex queries, since

the queries specified with fq are cached independently of the main query. These parameters can be used in combo with **document parameters** using the P{ } notation like the example picture shows.

Fields Mapping

It is important to set field types correctly in order to use a Solr dataset without problems. A field type defines the analysis that will occur on a field when documents are indexed or queries are sent to the index.

A field type definition can include four types of information:

The name of the field type (mandatory). An implementation class name (mandatory). If the field type is a number and it has decimals it must be set as pdouble (not int or string!!). If the field type is TextField, a description of the field analysis for the field type. Field type properties - depending on the implementation class, some properties may be mandatory.

Example: `<field name="REG_T_MP" type="pdouble" indexed="true" required="false" stored="true" multiValued="false"/>`

1.3.6 SparkQL Data Set

1.3.7 CKan Data set

Important: Enterprise Edition

If you purchased Knowage EE, this feature is available only in KnowageBD and KnowageSI

A Ckan dataset let you use open data as resource. You have to fill all the settings fields properly to let the dataset work successfully. Let's have a look on them:

- **File Type:** this field specifies the type of the file you want to import. Allowed ones are: CSV or XML;
- **Delimiter Character:** Here you have to insert the delimiter used in the file. Allowed values are: ; , \t |
- **Quote Character:** Allowed values for this field are: " or ';
- **Encoding:** Here you have to specify the encoding typology used. Allowed values are: UTF-8, UTF-16, windows-1252 , ASCII or ISO-8859-1;
- **Skip rows:** the number inserted stands for the rows not to be imported;
- **Limit rows:** it is the maximum number of rows to be imported. If you leave it blank all rows are uploaded;
- **XLS numbers:** it is the number of sheets to be imported;
- **CKAN ID :** here you have to insert the ID of the resource you are interested in. Look for it among the additional information in Ckan dataset webpage.
- **CKAN url:** it is the direct link to download the resources available on Ckan dataset webpage.

We marked with the * symbol the mandatory fields. We suggest to do a preview of your dataset before saving it to be sure everything have been correctly configured.

1.3.8 Python Data Set

The Python/R dataset enables users to create a dataset by writing a Python or R script that directly retrieves data. The developer of the dataset is free to write code which has to produce a **dataframe** variable as output (for Python scripts we refer to pandas dataframes). This variable will contain data that Knowage will later convert into its own format.

Fig. 1.72: Python/R dataset interface.

As shown in the picture in the field **Dataframe variable name** the developer has to specify the name of the variable in which the final output of the script is stored in the form of a dataframe.

In the field **Python environment** the user can select a working environment among the available ones defined in the **Configuration Management** section.

Inside the scripts it is possible to use parameters by the usual syntax $\$P\{\}$.

1.3.9 QBE Data Set

Important: Enterprise Edition

If you purchased Knowage EE, this feature is available only in KnowageBD and KnowageSI

The QbE dataset type option allows the definition of dataset results based on a query defined over a metamodel. To define a QbE dataset you need to select the Data Source and Datamart that you want to use. Once chosen your datamart you can click the lookup button of the Open QbE field and a pop up window will appear showing a QbE interface where you can define your query. Once saved, you can check the generated query thanks to the View QbE Query.

All these features are exhibited below.

1.4 Create a New Dashboard

Knowage allow end users to *self-build interactive cockpits* through an intuitive and interactive interface, with a few clicks and simple drag and drop. This allows you to compose your analytical documents with multiple widgets and define associations among them, so that clicking on one widget data are automatically updated in other widgets.

It enables *data mash-up* to integrate enterprise data and externally sourced data.

Cockpit document can be created and executed both by technical users and end users and are part of Knowage ad-hoc reporting system. A key aspect is that different widget can rely on different datasets and hence on different data sources. The only requirement needed to define associations between two or more datasets is the presence in each of them of one or more columns containing the same data.

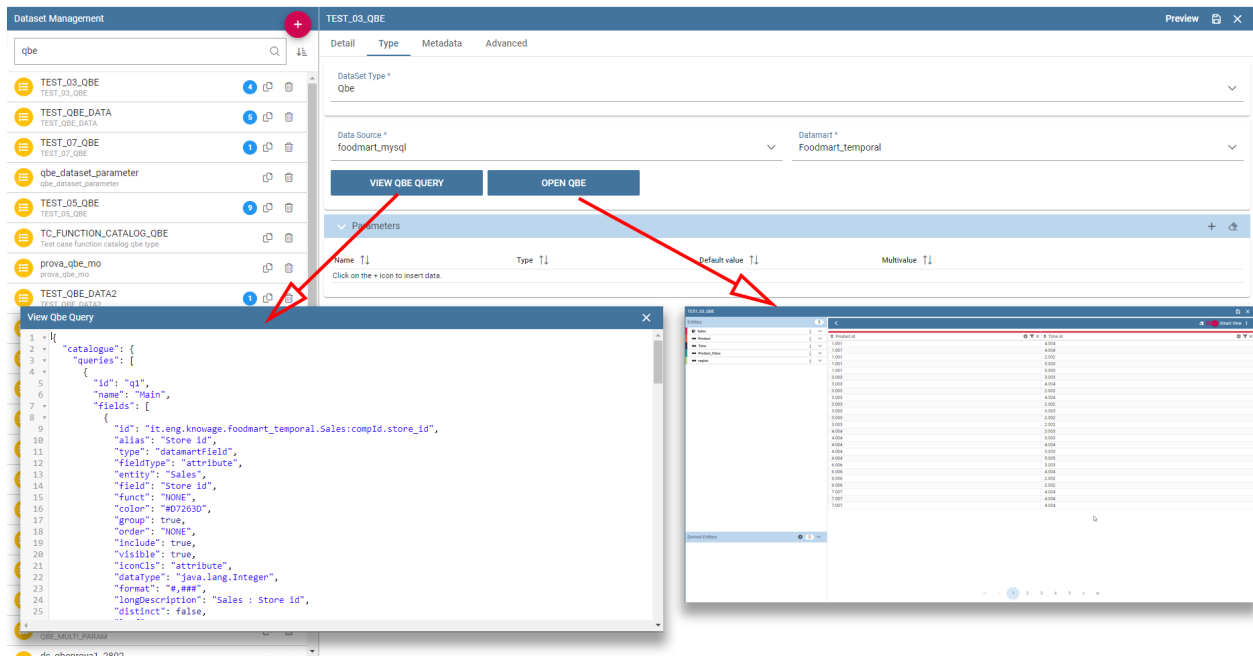


Fig. 1.73: QbE Dataset.

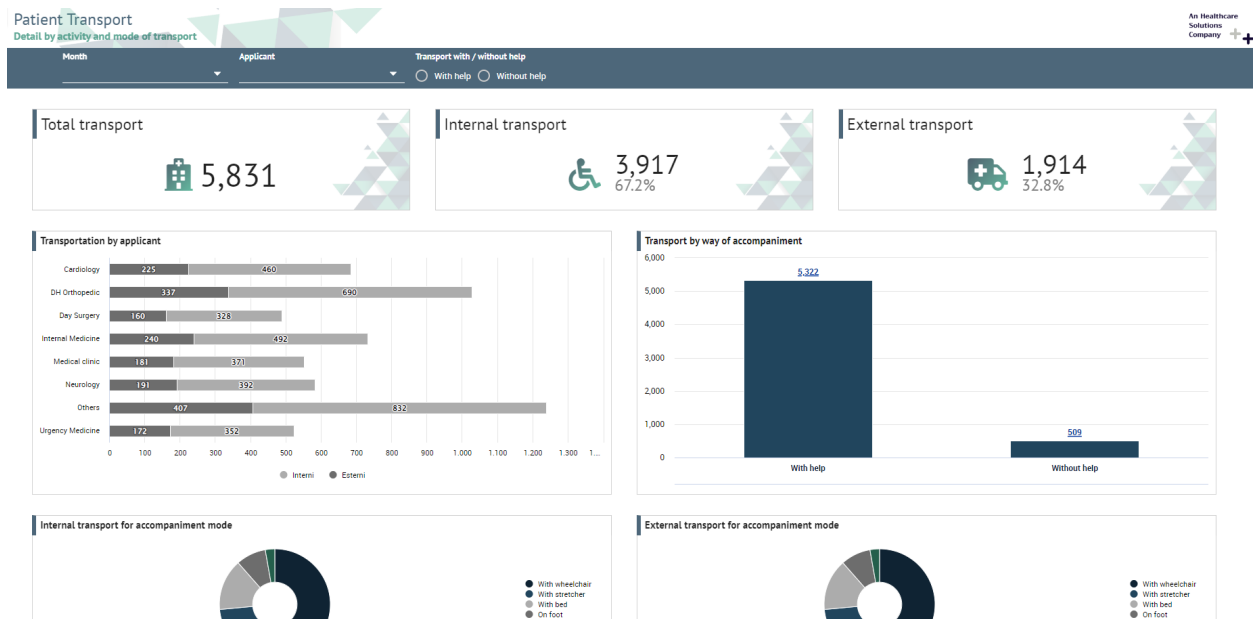


Fig. 1.74: Cockpit document example.

Warning: Section structure exception

Since there are no differences between the cockpit interface reached by a final user and the one reached by a technical user, the cockpit designer is described in one unique My first Cockpit for both those kind of users. By the way, when necessary we will highlight how the same functionality can be exploited accordingly to the user's role.

1.4.1 My first Cockpit








You can create your new Cockpit from the **Analysis** area of the **Workspace** by clicking on the “Plus” icon and selecting **Cockpits** if you enter Knowage Server as final user, while you can enter the document browser and start a new cockpit using the “Plus” icon if you enter Knowage Server as admin.

Important: Reaching the cockpit designer

We stress that the cockpit interface is reached by the final user and the administrator following two different paths.

Let us see how to build a cockpit and how the interface is displayed within the server. Once opened, the cockpit interface is an empty page with a toolbar containing different options described in Table below.

Table 1.4: Cockpit editor toolbar.

Icon	Name	Function
	Cockpit menu	Configuration menu of Cockpit.
	Add widget	It opens a window where you can create a new chart or table, add texts, images or Knowage documents.
	General configuration	It opens the window where you set the general cockpit options (name, label, show menu, etc.) and widget style (header, titles, borders, etc.).
	Data configuration	It opens a window where you can manage the dataset, the association between datasets and the refresh frequency.
	Selections	It opens a window where you can manage selections.
	Clear Cache	It cleans temporary data.
	Save as	It opens the window to save the cockpit document as a new document.

By clicking the button **Add Widget** you can add a widget containing a **Text**, an **Image**, a **Chart**, an **HTML**, a **Table**,

a **Cross table**, a **Document**, a **Map**, the **Active selections** or the **Selector** to your cockpit, as shown below. **Python**, **R**, **Discovery** and **Custom charts** are available according to the license installed.

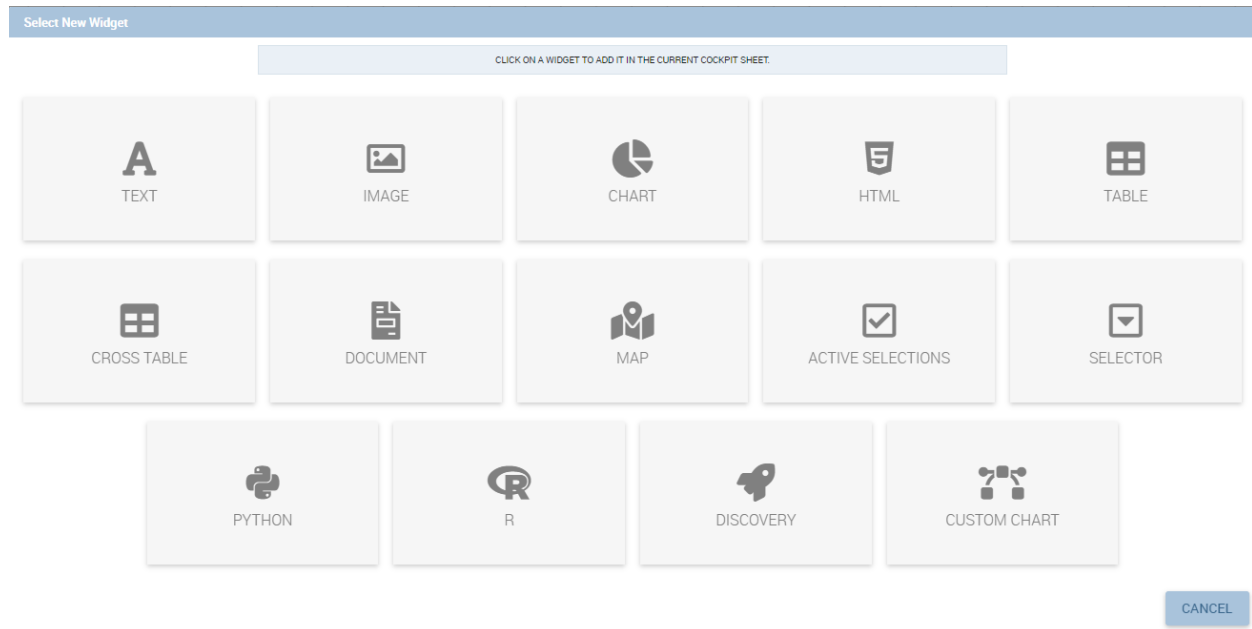


Fig. 1.75: Widget Type.

In the following we go into details of each available widget.

1.4.1.1 Text widget

By clicking the button Text Widget you can add text to your cockpit. As shown in figure below, the widget editor opens and it is divided in three tabs: the **Text editor**, the **Style**, the **Dataset** and the **Filters** tab.

On the “Text editor” tab you can type the desired text in center panel and customize it. Using the dataset tab it is possible to associate dataset values to the text and read it real time at each execution. Move to the dataset tab to add a dataset to the widget. Then, going back to the Text editor tab, the user will find the dataset columns on the right side, as well as a set of functions to eventually apply to the fields. We summed up main steps in the Figure below. To add a function to a measure first select the desired function and then the field of numeric type.

It’s also possible to add variables values if at least one of them is defined. During the execution of the widget the value will be displayed based on the current variable value.

On the “Style” tab you can customize the text widget. We have provided all details about this tab in the Table widget. On the “Dataset” tab you can add more dataset to be used in the dynamic value. Finally, the “Filters” tab can be used to extract limited output from the dataset. We put details off to the table widget subsection.

1.4.1.2 Image widget

By clicking the button **Image Widget** you can add images to your cockpit. As already seen the widget editor opens and it is divided in three sections.

On the **Gallery** tab you can upload an image, delete it or select one from the gallery. Refer to the following figure.

On the **Style** tab you can configure the style of your image widget with the different options offered by this tab. Many of them are defined in the table widget that you will find later.

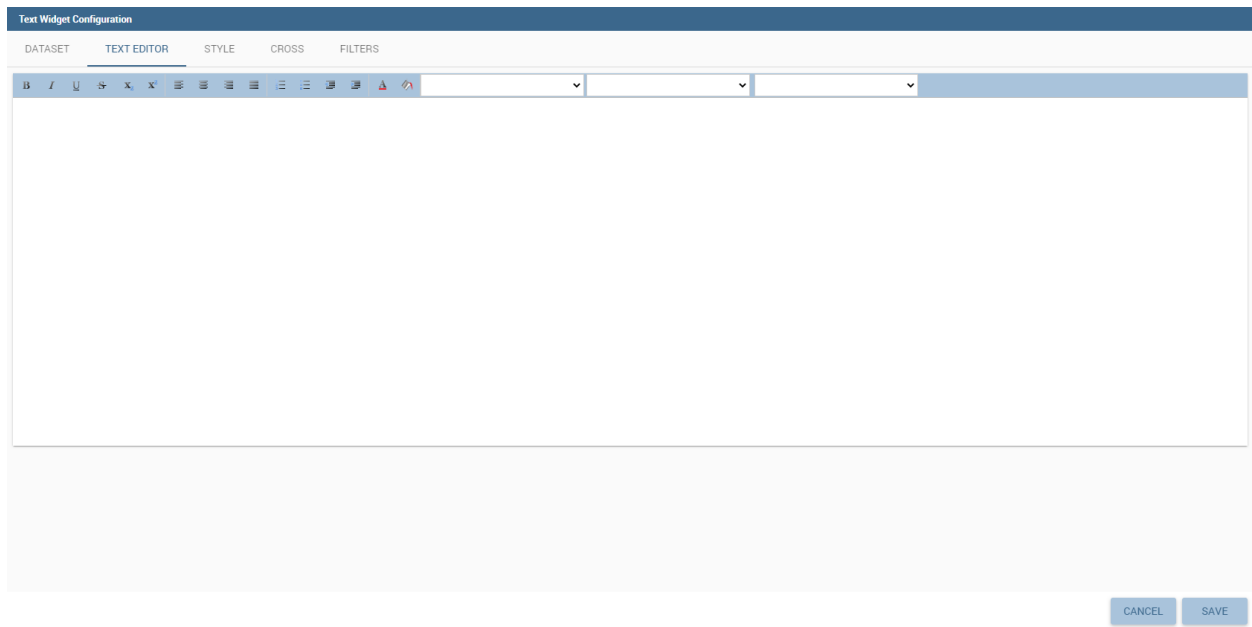


Fig. 1.76: Text editor of text widget configuration.

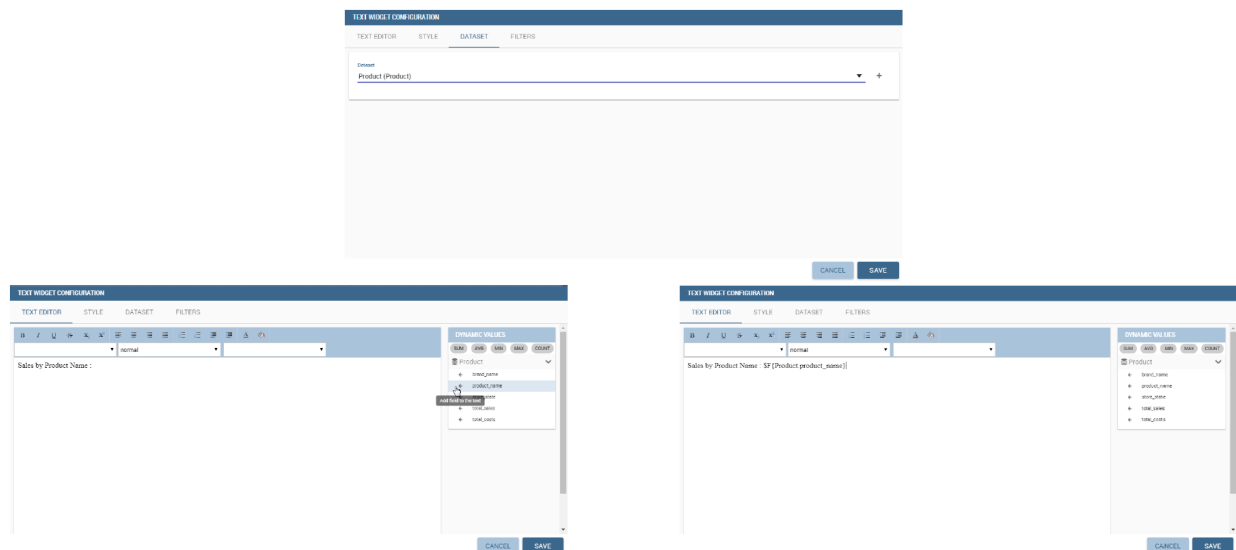


Fig. 1.77: Editing a dynamic text.



Fig. 1.78: Variables in text widget.

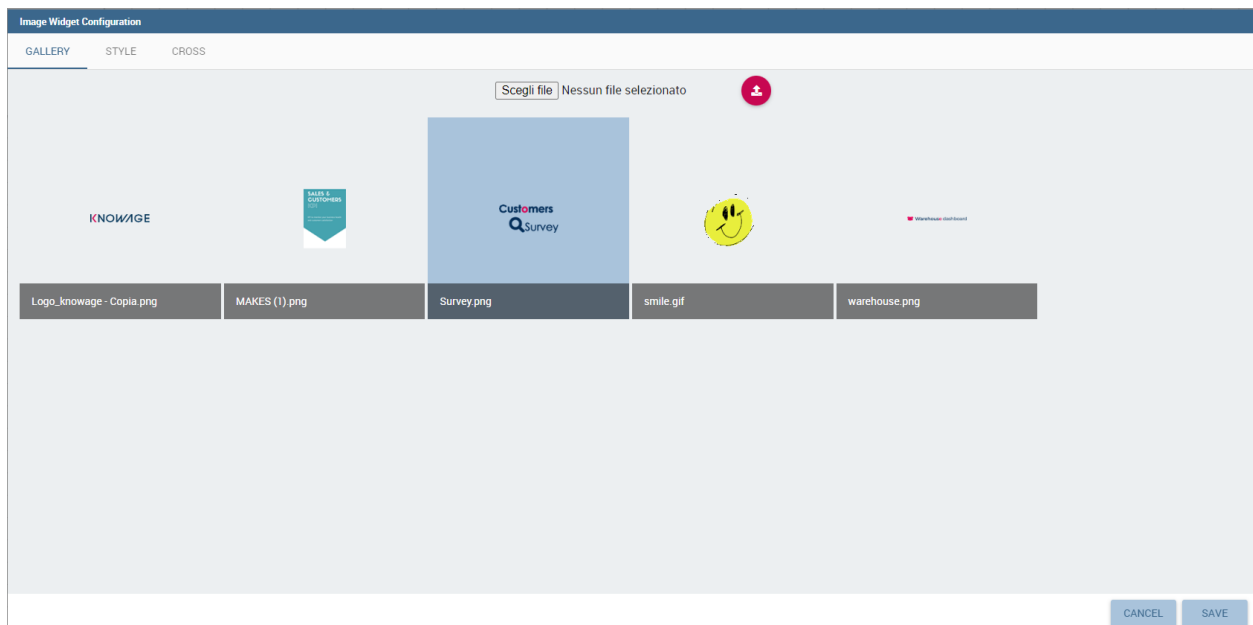


Fig. 1.79: Gallery tab of Image Widget Configuration.

On the **Cross** tab you can define navigation to another document, as shown in figure below.

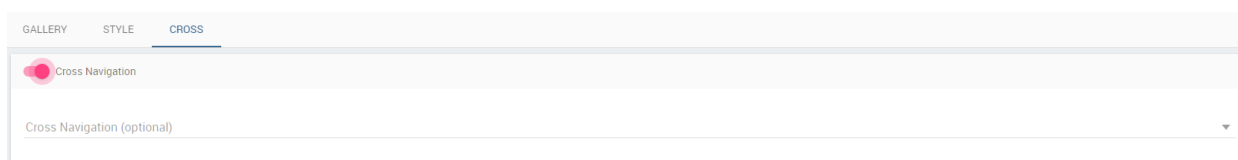


Fig. 1.80: Cross tab of Image Widget Configuration.

Warning: Cross navigation only for technical users

Due to the fact that parameters can only be managed by a technical user the cross navigation cannot be implemented by the final user.

For this purpose, you must activate **Enable cross navigation** flag and select the destination document through the list of cross navigation definition. This last flag is optional. If you select a cross navigation definition, when you launch the cross navigation it will go to the document of arrival directly. If the cross navigation definition is not defined, then when you launch the image widget cross navigation will be shown a pop up (refer to figure below) with the list of cross navigation definition that exist for this cockpit.

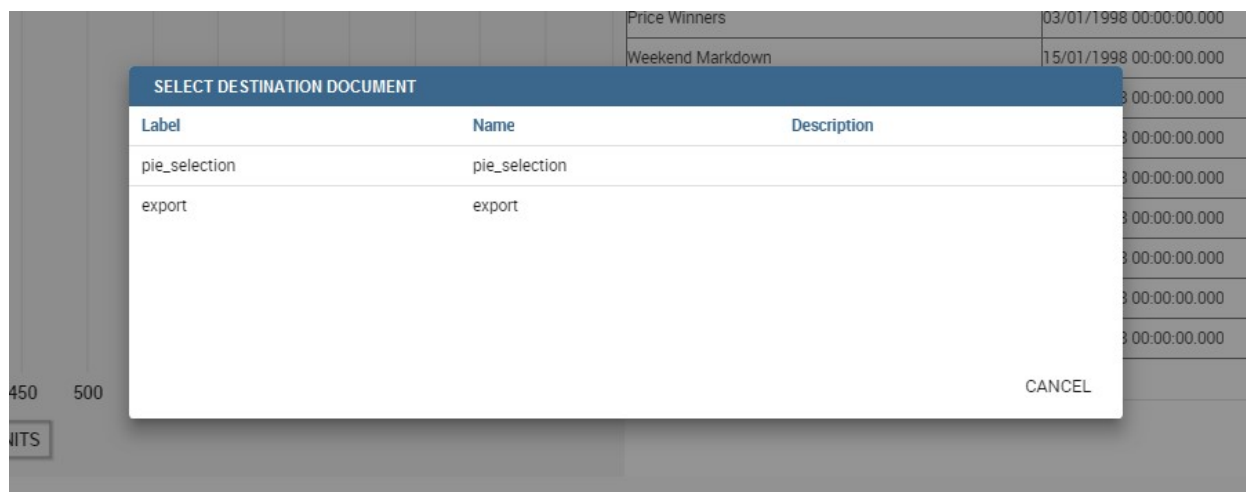


Fig. 1.81: Cross navigation multiple choices.

1.4.1.3 Chart widget

Charts are an essential representation of data, Knowage let you use many different charts type and configure them according to your needs. We have provided all details about charts type and configuration in Chart chapter.

We recall that also for chart widget it is possible to set cross navigation on elements.

Warning: Cross navigation only for technical users

Due to the fact that parameters can only be managed by technical user the cross navigation cannot be implemented by the final user.

As shown in next figure, it is mandatory to enable the cross navigation feature by using the dedicate tab of chart editor GUI. It is mandatory to choose the column element to be passed to the destination document and associate it to the right output parameter (previously added to the document using the detail interface).

The cross navigation name can be left empty. In case multiple cross navigation definitions have been configured for the document, a pop up will be displayed, letting the user to choose which destination to reach (exactly as we saw earlier for Image widget in the last figure of that paragraph).

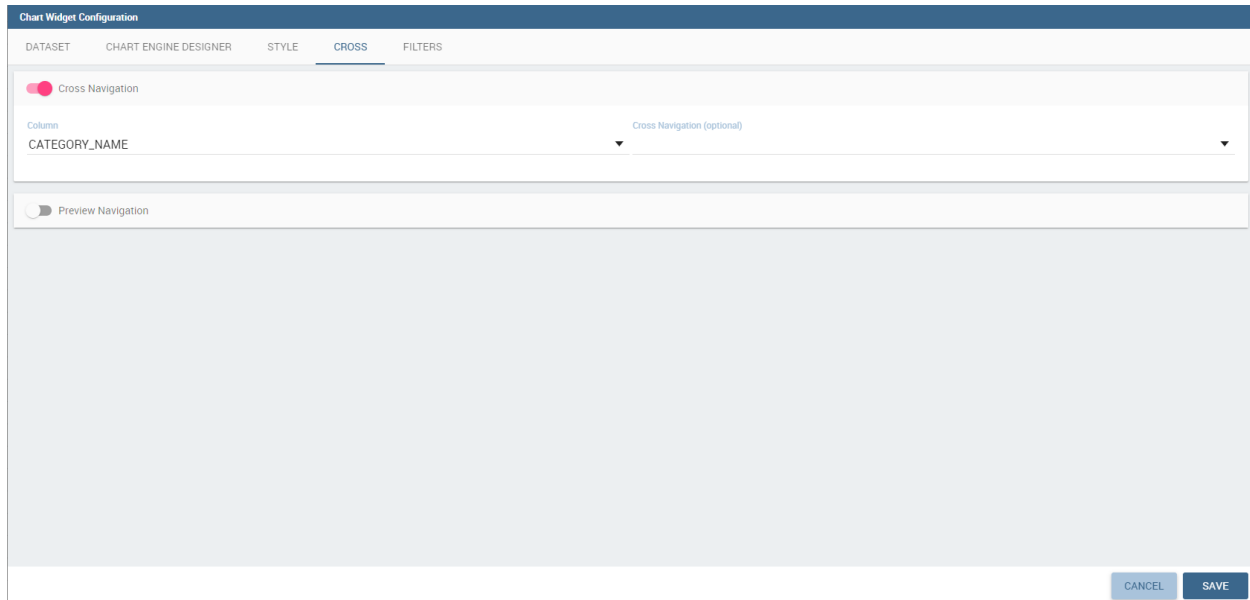


Fig. 1.82: Cross navigation for chart widget.

In addition, if the navigation expects other parameters to be passed, use the bottom part of the page to add the additional parameters. Figurebelow shows an example.

It is also possible using Solr datasets and calculated fields with charts.

Dataset fields editing and configuration is possible using the first tab inside widget edit mode, in this way, for example, users can modify aggregations for measures:

In this way it is possible adding new calculated fields on chart by clicking on “Add Calculated field”, the standard calculated field editing mode will appear.

Adding new calculated fields is easy as using other measure fields, using Chart Engine Designer structure tab:

As said before charts are the most adopted method in presenting BI data since they allow an immediate perception of a phenomenon and are easily understandable. That is why it is worth having an overview through them when creating a dashboard that includes a chart widget.

Knowage provides a chart engine to create several types of charts, including:

- Bar
- Line
- Pie
- Sunburst
- Wordcloud
- Treemap

Chart Widget Configuration

DATASET CHART ENGINE DESIGNER STYLE **CROSS** FILTERS

☒ Cross Navigation

Column: CATEGORY_VALUE Output parameter: quarter Cross Navigation (optional): CROSSCOCKPIT2

Output parameters list

Checkbox	Parameter	Type	Value	Column	Dataset	Column
<input checked="" type="checkbox"/>	p1	Static	10			
<input checked="" type="checkbox"/>	amount	Dynamic		SERIE_VALUE		
<input checked="" type="checkbox"/>	store	Selection			TEST_03	STORE_ID

☐ Preview Navigation

CANCEL SAVE

Fig. 1.83: Add all output parameters involved in the cross navigation.

Chart Widget Configuration

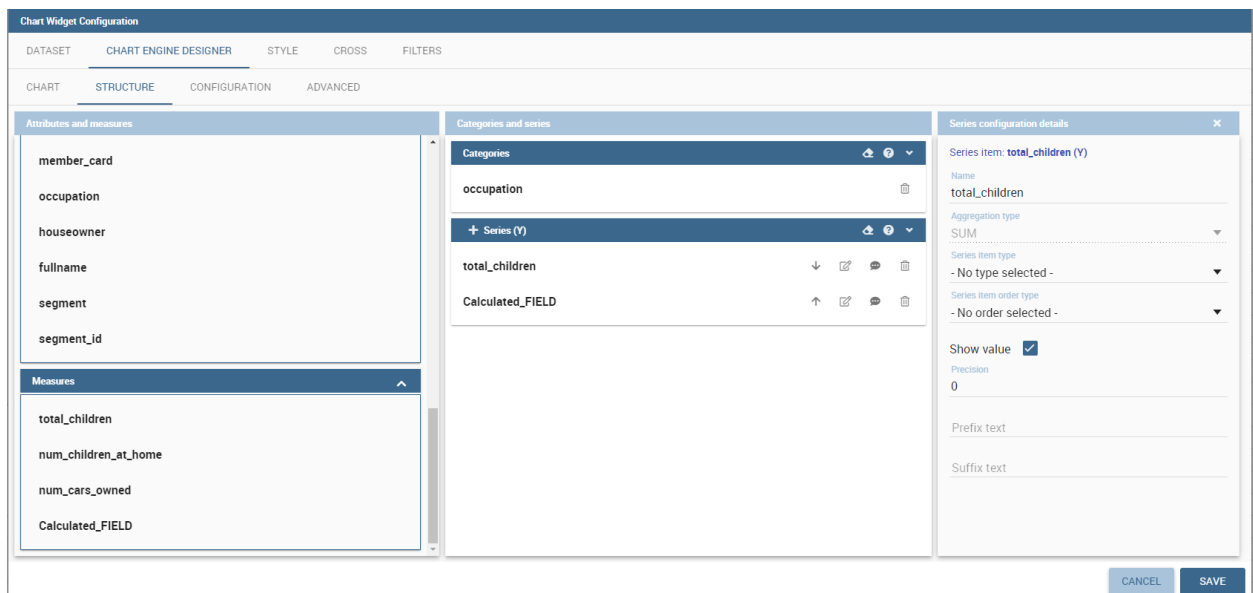
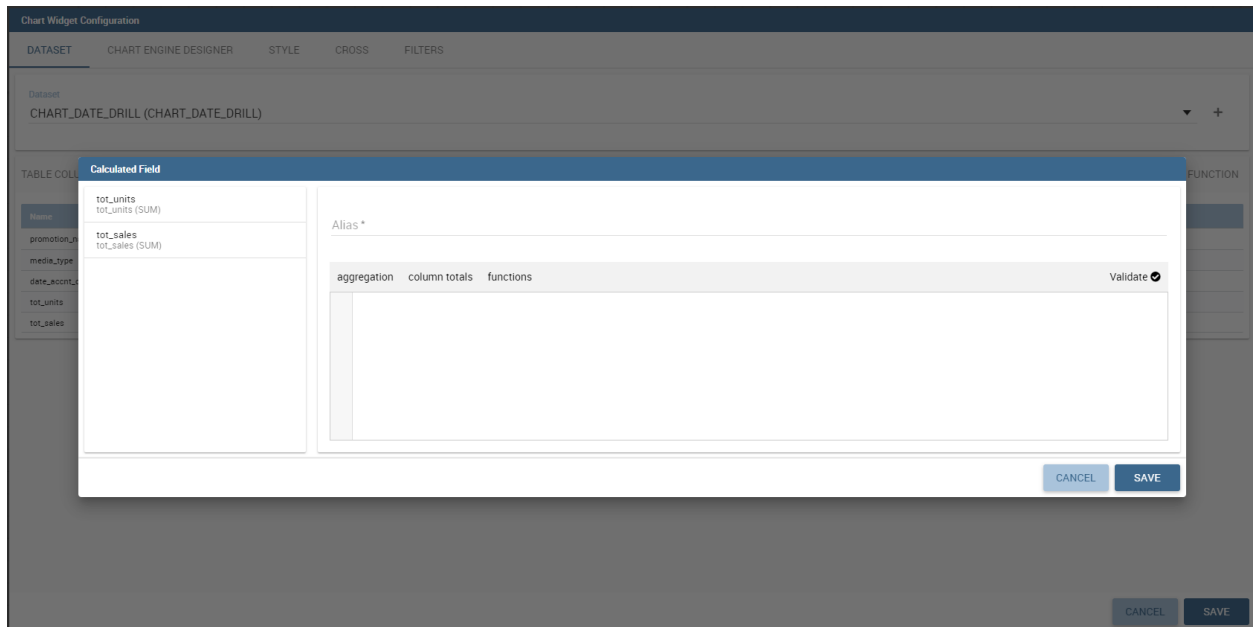
DATASET CHART ENGINE DESIGNER STYLE CROSS FILTERS

Dataset: Dumbbell Example (Dumbbell_custom_chart)

TABLE COLUMNS ADD CALCULATED FIELD USE FUNCTION

Name	Alias	Type	Data Type	Aggregation
employee_id	employee_id	MEASURE	# integer	SUM
full_name	full_name	ATTRIBUTE	string	
first_name	first_name	ATTRIBUTE	string	
last_name	last_name	ATTRIBUTE	string	
position_id	position_id	MEASURE	# integer	SUM
position_title	position_title	ATTRIBUTE	string	
store_id	store_id	MEASURE	# integer	SUM
department_id	department_id	MEASURE	# integer	SUM
birth_date	birth_date	ATTRIBUTE	date	
hire_date	hire_date	ATTRIBUTE	timestamp	
end_date	end_date	ATTRIBUTE	timestamp	
salary	salary	MEASURE	# float	SUM
supervisor_id	supervisor_id	MEASURE	# integer	SUM
education_level	education_level	ATTRIBUTE	string	
marital_status	marital_status	ATTRIBUTE	string	
gender	gender	ATTRIBUTE	string	
management_role	management_role	ATTRIBUTE	string	

CANCEL SAVE



- Parallel
- Radar
- Scatter
- Heatmap
- Chord
- Gauge
- Bubble

Once you enter the Knowage environment as a final user, enter the **Analysis** area under the **Workspace** menu item, click on the **Create Analysis** icon and choose **Cockpit**.

Important: Enterprise Edition only

Please note that this operation is available only in KnowageBD and KnowageSI. Using the KnowagePM license, only a technical user can create Cockpit document.

Once opened, the cockpit interface is an empty page with a toolbar containing different options, the second of which is the **Add chart** feature.

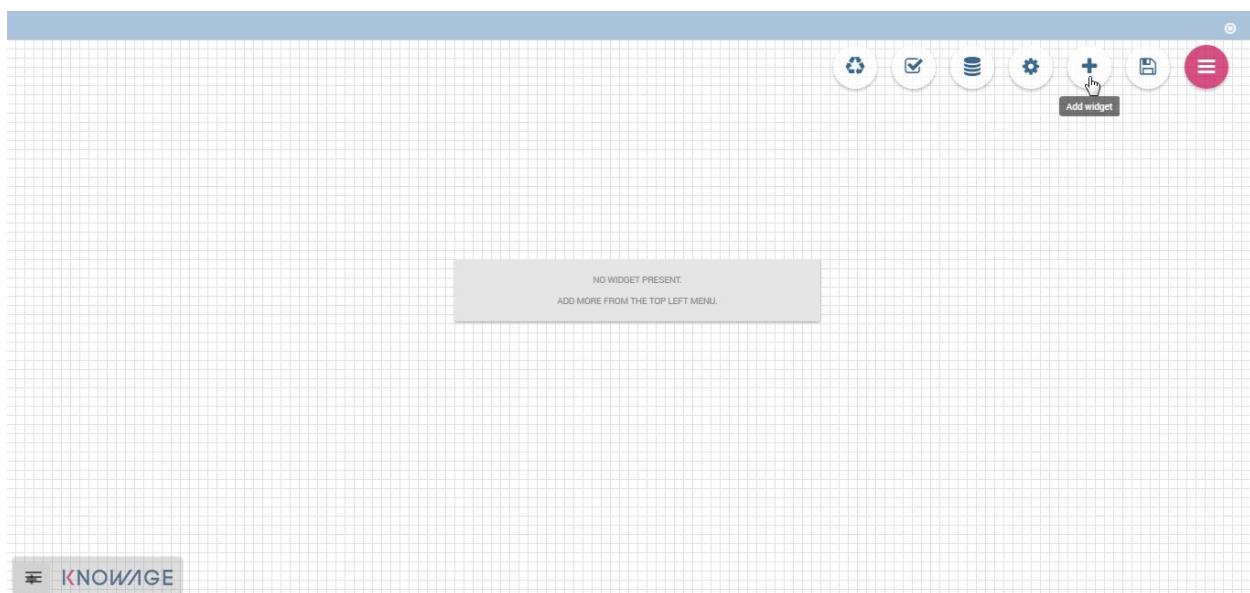


Fig. 1.84: Add a chart to a cockpit.

Note: Cockpit

The Cockpit Engine allows the user to self-build interactive cockpits through an intuitive and dynamic interface. Read more in *Cockpit* chapter.

Clicking on the **Add Chart** icon, you will be asked to choose among some available widgets. Pick out the **Chart** one and let's now go into details on how to build a chart from scratch. The designer editor is divided into four principal tabs: **Dataset**, **Chart Engine Designer**, **Style**, **Cross** and **Filters**. As soon as the user clicks on the “Add Chart” button, he/she enters the “Dataset” tab editor. Here the user must select, using the “little plus” icon placed just aside

the combobox line, one dataset. Then the user must switch to the “Chart Engine Designer” tab and choose a chart type among the available ones, as shown in figure below.

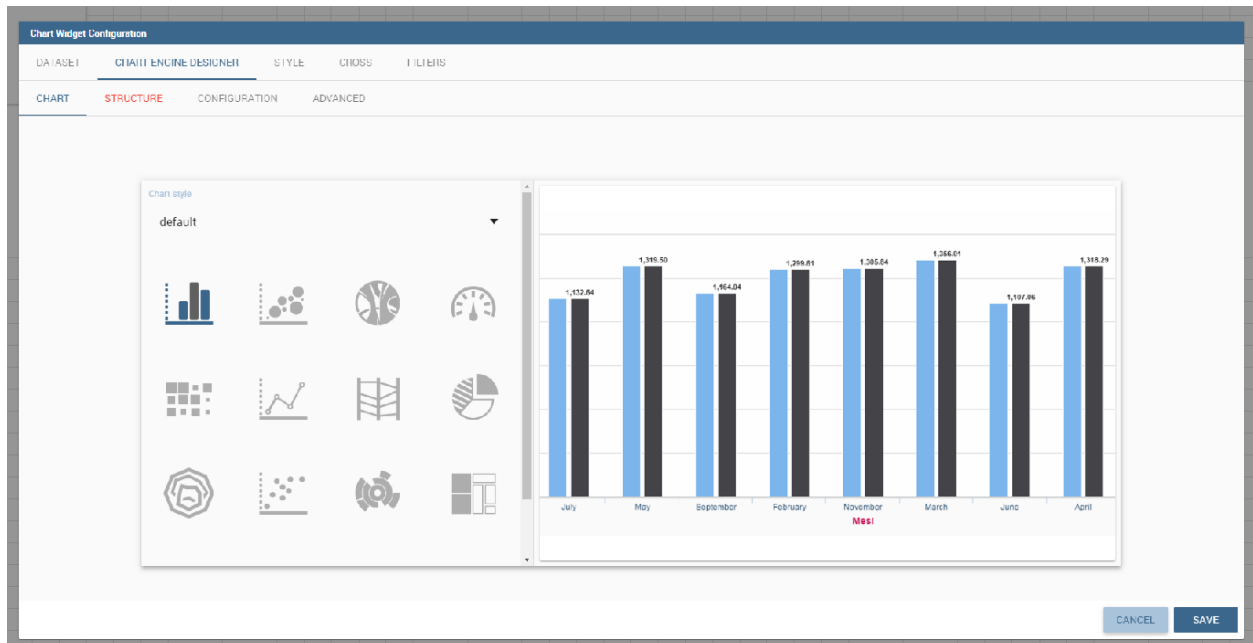


Fig. 1.85: Chart editor.

After choosing the appropriate chart type you must go into the **Structure** page. Here it is possible to select the measures and the attributes chosen for the chart.

Clicking on the **Configuration** page you will found eight different blocks as you can see in figure below.

In detail these blocks concern:

- **Generic Details**, as the orientation of the chart, the family and the size font.
- **Title and Subtitle details**
- **No data message** where it is possible to put a message where the data are not founded.
- **Legend Title**
- **Legend Items**
- **Color Palette**
- **Advanced Series Configuration**
- **Custom Colors**

These eight blocks are common to all chart types; anyway, there could be some additional blocks for specific types.

The **Advanced** tab contains extra features, usually exploited by an expert user. Here the user can see all settable properties associated to the chart: it reflects the property tabs that an expert user should manually edit to generate a json template.

In the next subsections, the available functionalities of the Structure, the Configuration and the Advanced tabs are described in a more specific way.

Important: Enterprise Edition only

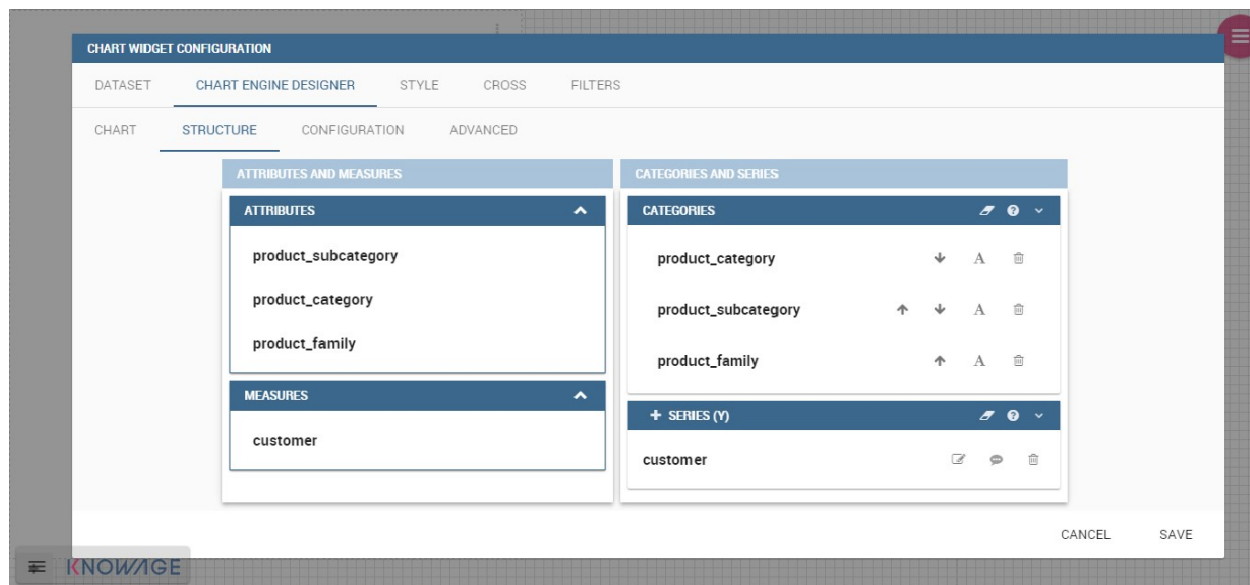


Fig. 1.86: Chart structure.

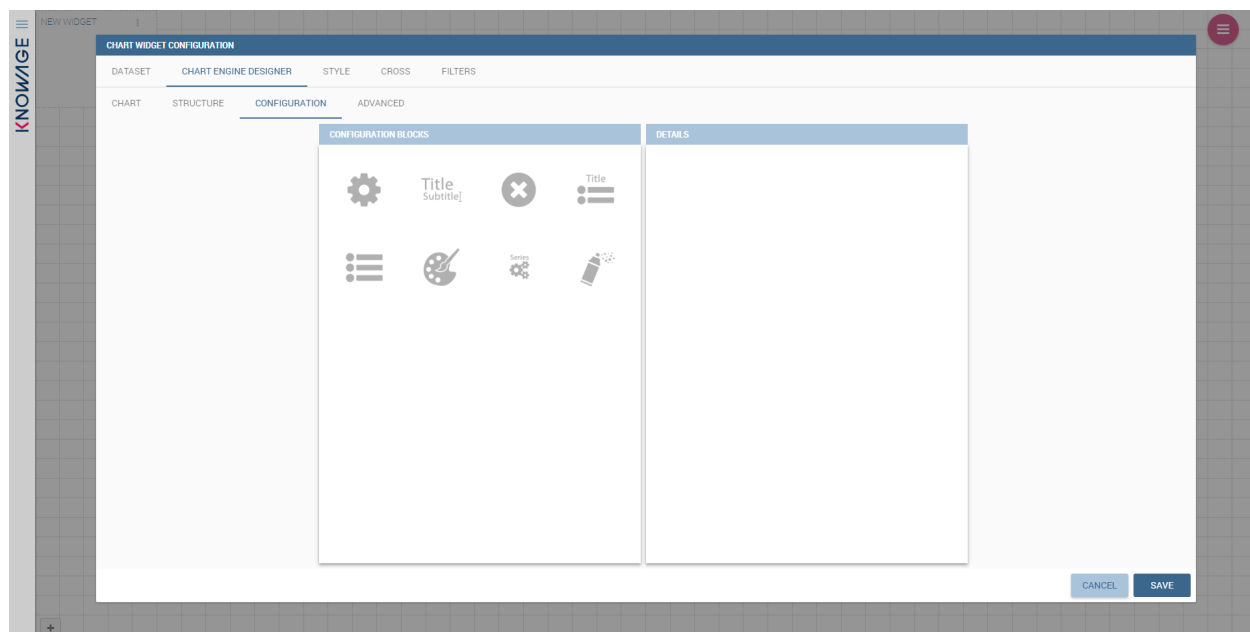


Fig. 1.87: Chart configuration.

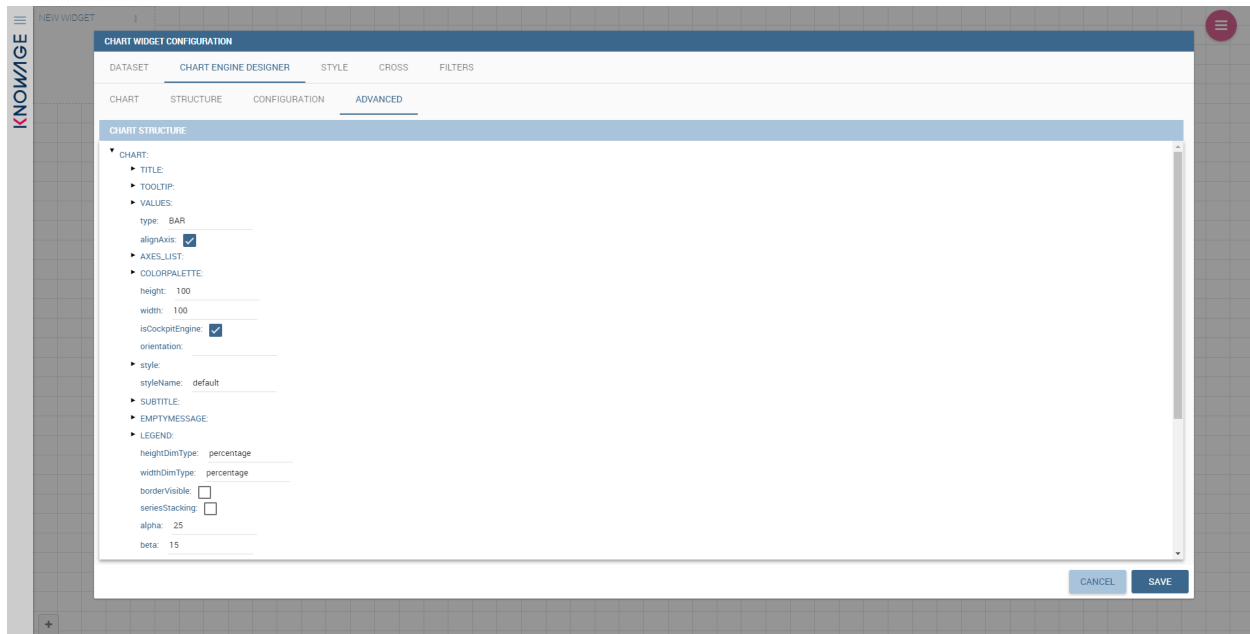


Fig. 1.88: Chart Advanced Features.

Please note that the “Advanced” tab is available only in Knowage Enterprise Edition.

The “Structure” tab of the designer is the core of the Chart development. Here it is possible and mandatory to choose the measures and the attributes. When selected, the tab shows a two axes panel. The horizontal axis indicates the X-axis where you must choose one or more attributes. As well, the left axis is the Y-axis and here you must choose measures. You can also insert manually the axis title for both the X and the Y axis if the chart is configured to have axis titles.

Warning: Chart type changemens may cause broke down

Before creating any chart, it is convenient to be sure of what kind of chart you want to develop. We stress that the user can change the chart type afterwards, but at the expense of a loss of just defined settings.

In this section it’s possible to customize the labels of the axis, title and grid style clicking on different buttons. With the arrow button, on the top of the Y-axis and X-axis, it’s possible to choose the axis configuration detail, the axis title configuration, the major and minor grid configuration (just for Y-axis) and ordering column (just for X-axis). With the pencil button opens a window on the right with the series configuration details where it’s possible to choose the aggregation way, the order type of the series, if the data will be shown e so on. Finally, with the strip cartoon button you can choose the features of the tooltip (font color, text alignment, ecc). If the chart in place does not allow the customization of the axes the specific button will be disabled or not visible. The Figure below will show in detail the three buttons above explained:

The **Configuration** section contains options to define the generic style of the chart. Here you can set the dimensions of the chart, the background color, insert the title and subtitle and define their style, choose the series palette, associate a specific color to a particular serie or category, add and configure the legend. The listed options are an example of what you can configure in the tab.

Note that for the color palette details you can use one already in the list or you can choose any color inserting the hex color code with the hashtag symbol. This is a very useful feature to customize the output.

In particular, in the 6.3 version, it has been introduced a new configuration option: the Custom Color.

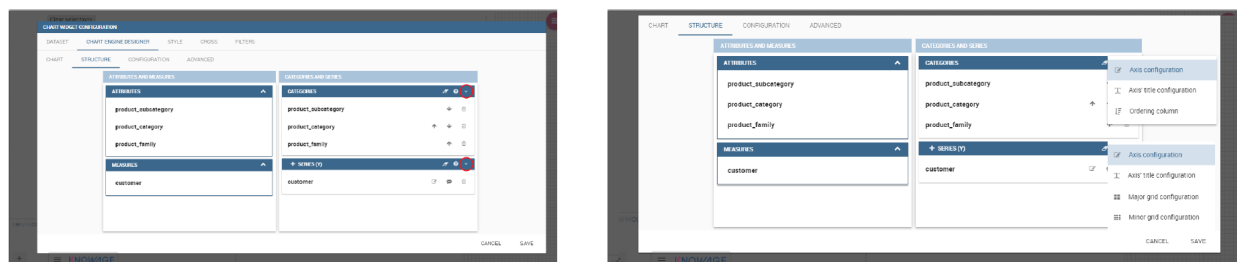


Fig. 1.89: From left to right: (a) Generic configuration axis (the specific arrow). (b) Generic configuration axis.

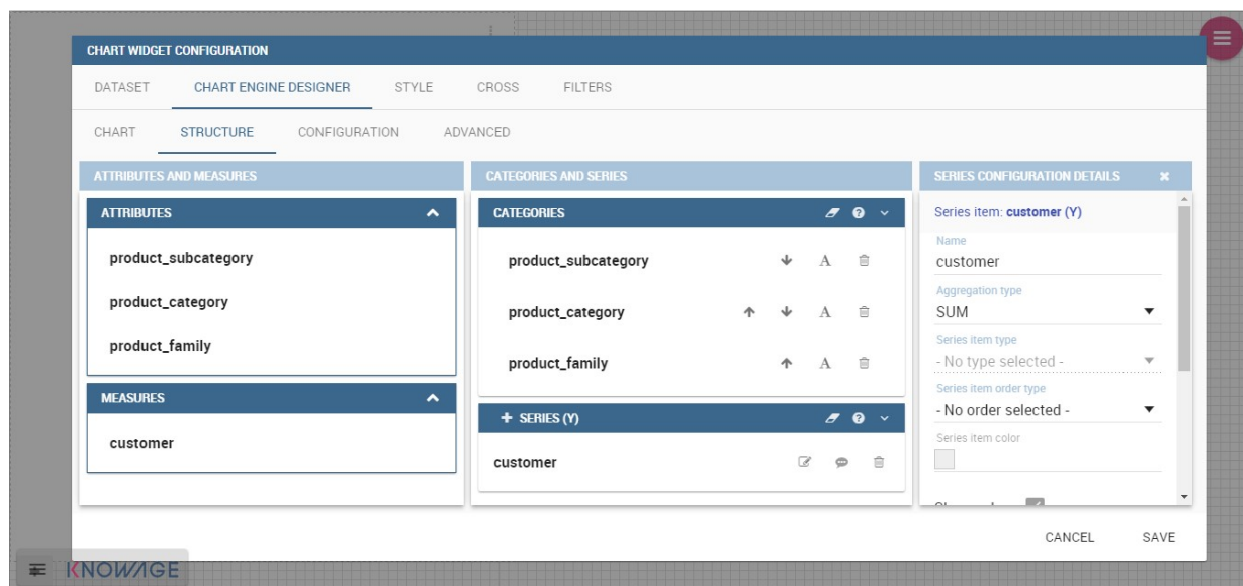


Fig. 1.90: Series style configuration.

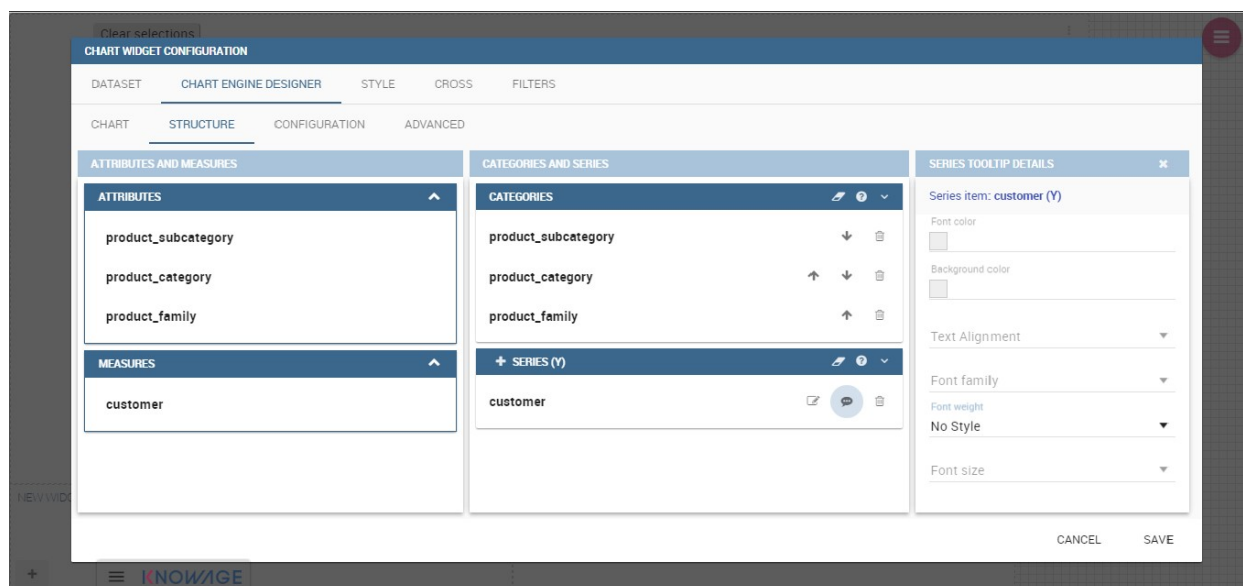


Fig. 1.91: Series tooltip details.

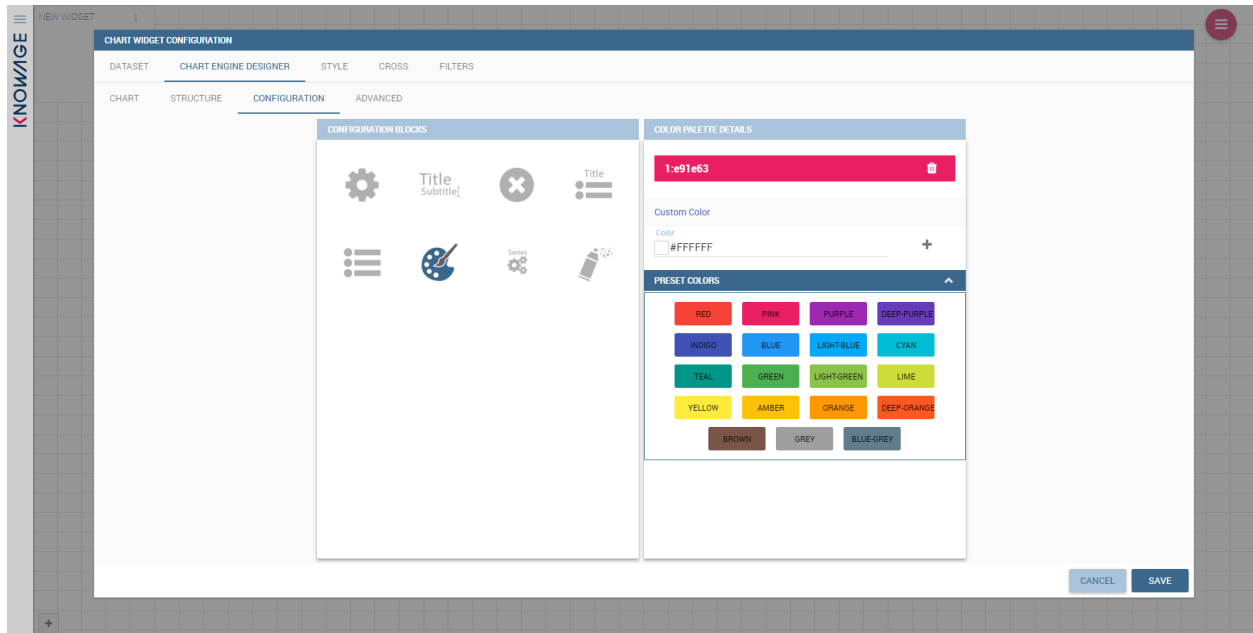


Fig. 1.92: Color box editing.

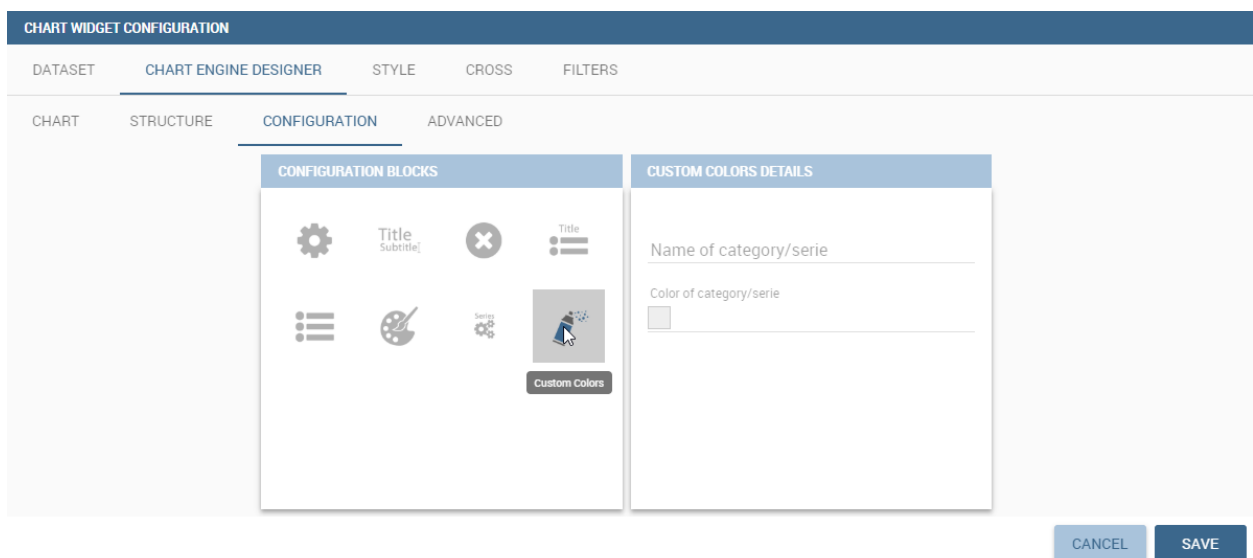


Fig. 1.93: Custom Colors details.

With this new option it is possible to assign a specific color to a particular category and/or serie or to a particular value of a category and/or serie. Look at the following figure for an example.

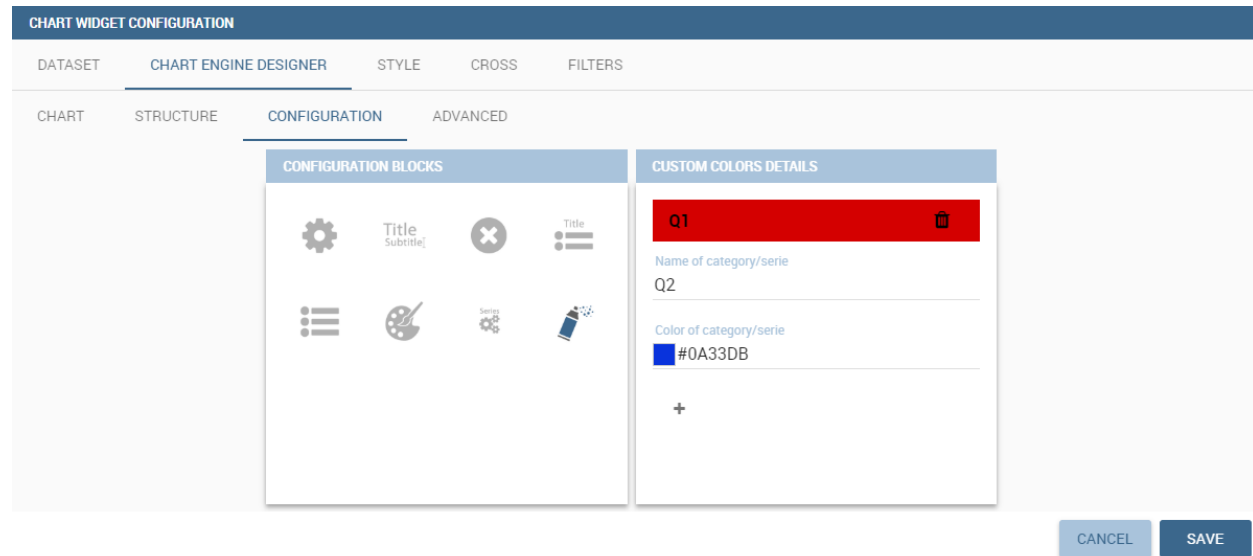


Fig. 1.94: Custom Colors example.

To add a custom color simply write the category/serie value or name, select a color with the color picker and then click on the plus button. In the figure example it is assigned a color for each value of the 'QUARTER' category.

Indeed, the options available in this tab change according to the chart selected enabling different configurations. See Chart types in detail for a detailed description of the specific options of each chart.

The **Advanced** tab contains some advanced options to more customize the chart. Here it is possible, for example, to set the tooltip options, the widget dimensions, if the chart is stacking or not, the grouping type.

Down here are listed some of the most useful and new options.

The **dataLabels** option can be found under the path VALUES -> SERIE -> 0 or another serie -> dataLabels. The option is available only for measures. Here it is possible to set the labels style such as the color, font family or font weight.

The **TOOLTIP** option allows to set the width and the radius of the tooltip's border.

The **plotBands** and **plotLines** options can be found under the path AXES_LIST -> AXIS -> 0 or another serie. With these options it is possible to plot respectively bands and lines on the chart with fixed values and to set their style, like the line width and the line type or the band color.

The **min** and **max** options are under the path AXES_LIST -> AXIS -> 0 or another serie. They are available only for series and allow to set the maximum and minimum axis value for the selected series's axis.

Following, a description on how to create charts within the **Chart Engine** of Knowage.

Traditional charts

Knowage allows you to create the so-called *traditional charts* like bar, line, pie, radar and scatter chart in a fancy way.

Each chart type is built on a specific dataset. Despite all, there are some general rules that can be applied to those "simpler" and common charts. The minimum requirement is to define/have a dataset with at least one attribute column and one measure column. Then you can select the type of chart you want to use from the **Chart** section; meanwhile using the **Structure** section you can fill in the category box with one or more attributes (typically these will be placed in the X-axis) and in the series box with one or more measures (typically placed as Y-axis' values). Refer to *Chart Structure* figure as example.

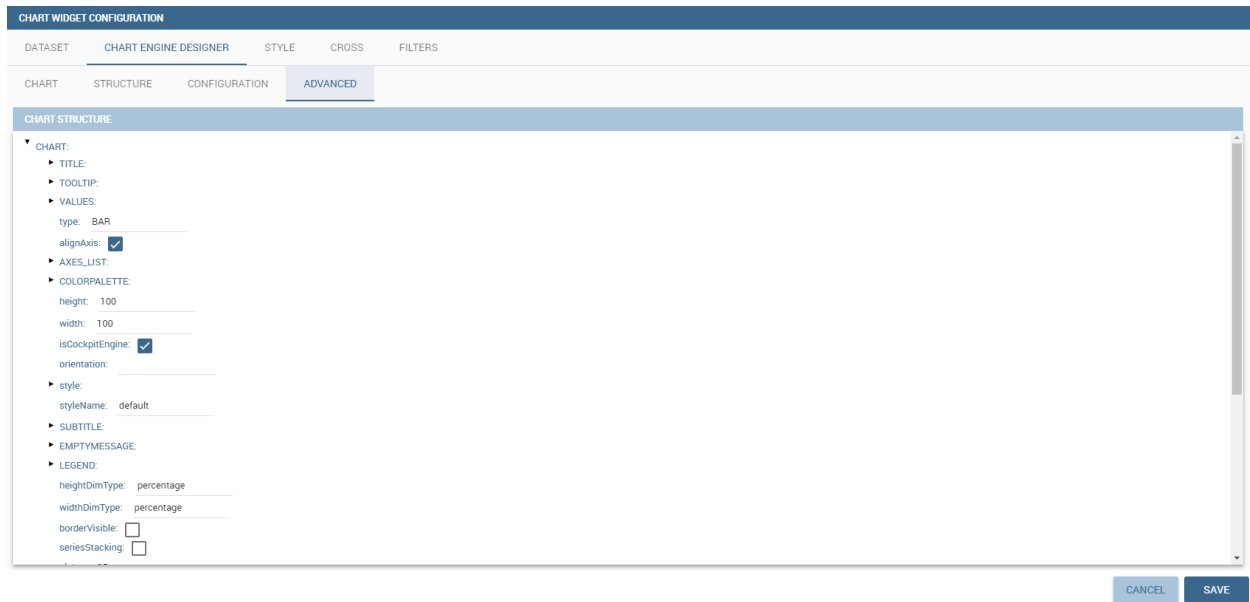


Fig. 1.95: Advanced tab.

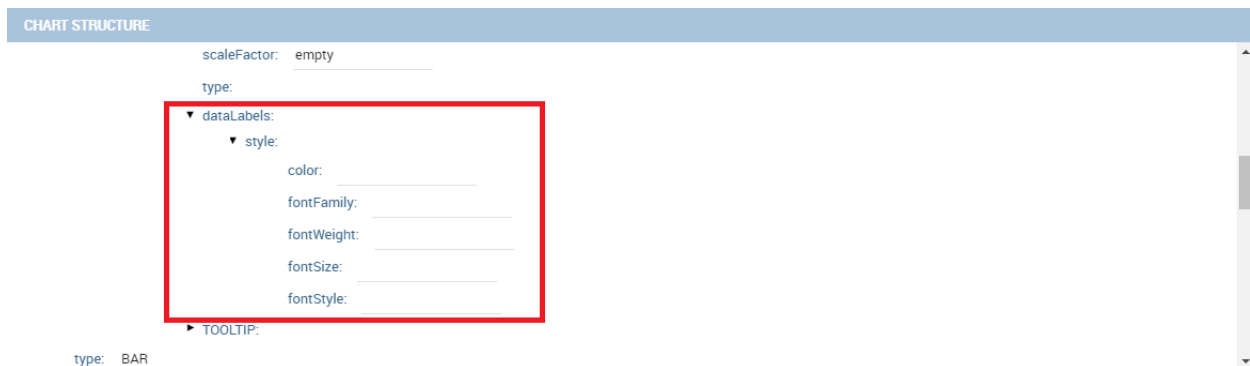


Fig. 1.96: dataLabels option.



Fig. 1.97: plotBands option.



Fig. 1.98: min and max options.

Once you have selected the attributes and measures you can edit the series style and axis style configurations as explained in My first Chart. Then go to **Configuration** to set the chart dimension, the title, the legend and to choose how to associate colors to series.

Some charts are endowed with datetime and grouping functions. In particular, it is possible to enable the grouping/splitting functions to **Bar** and **Line** charts.

The user can reach those functions just clicking on the “little arrow” located at the right end of category bar.



Fig. 1.99: Datetime and grouping function.

The grouping functions can be implemented only through specific categories and series configurations. As shown in figure below, the grouping function cannot be applied with just one attribute as category. To allow the function to be applied, the user must define two attributes as category fields.

As well, the user can use the splitting functions to divide one series over the second one or over the second category.

To split the first series over the second one, remember that it is necessary to choose only one attribute as category field and two measures as series values. The following figure shows an example.

Meanwhile to split a measure over second category it is mandatory to choose exactly two attributes as category field and only one measure as series value, as shown in figure below.

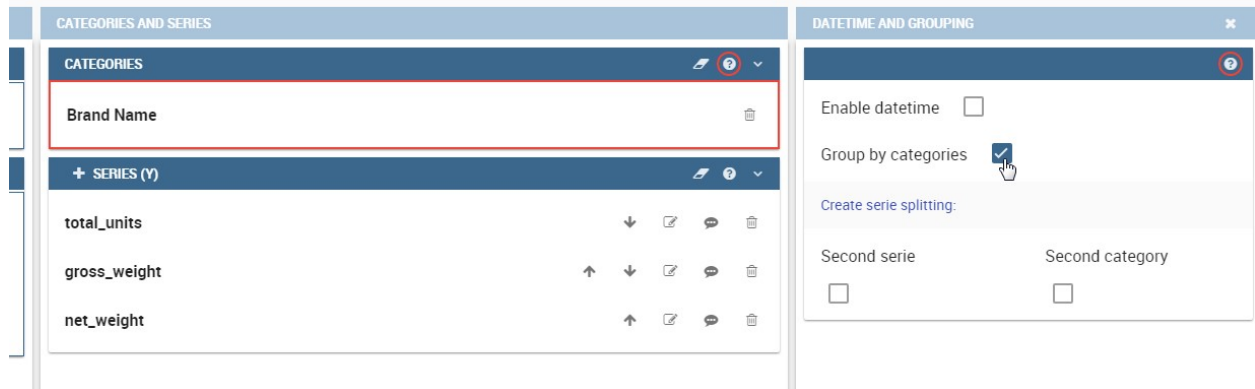


Fig. 1.100: Error alarm when enabling the grouping function.

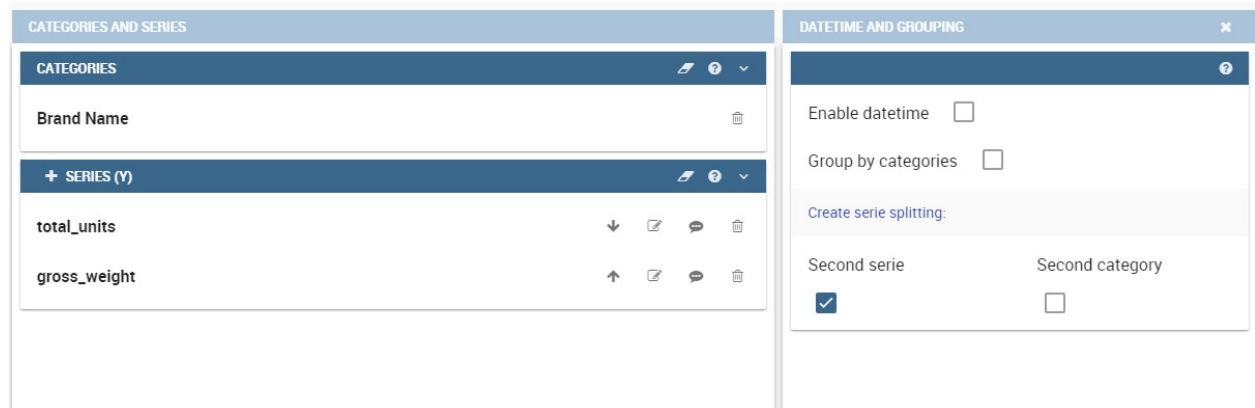


Fig. 1.101: Split over second series.

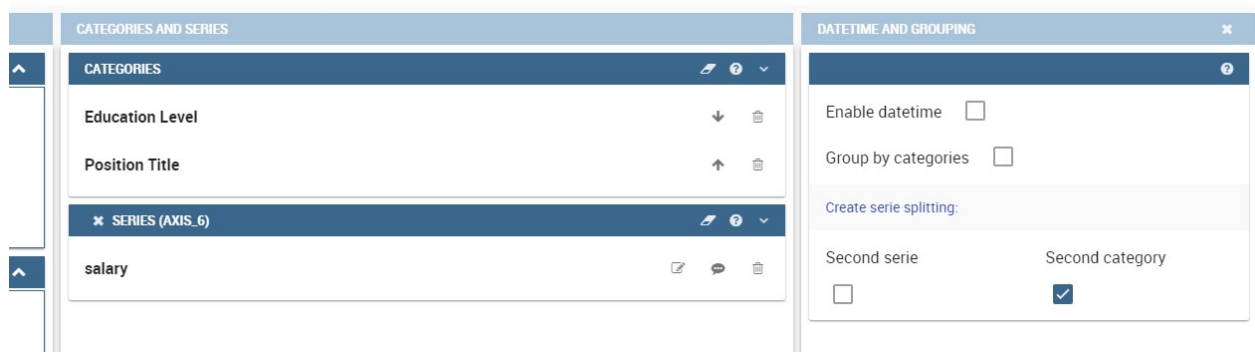


Fig. 1.102: Split over second category.

Futhermore, in the occurance the chart uses one datetime attribute as category field, the user can improve visualization applying the datetime function to custom date format.

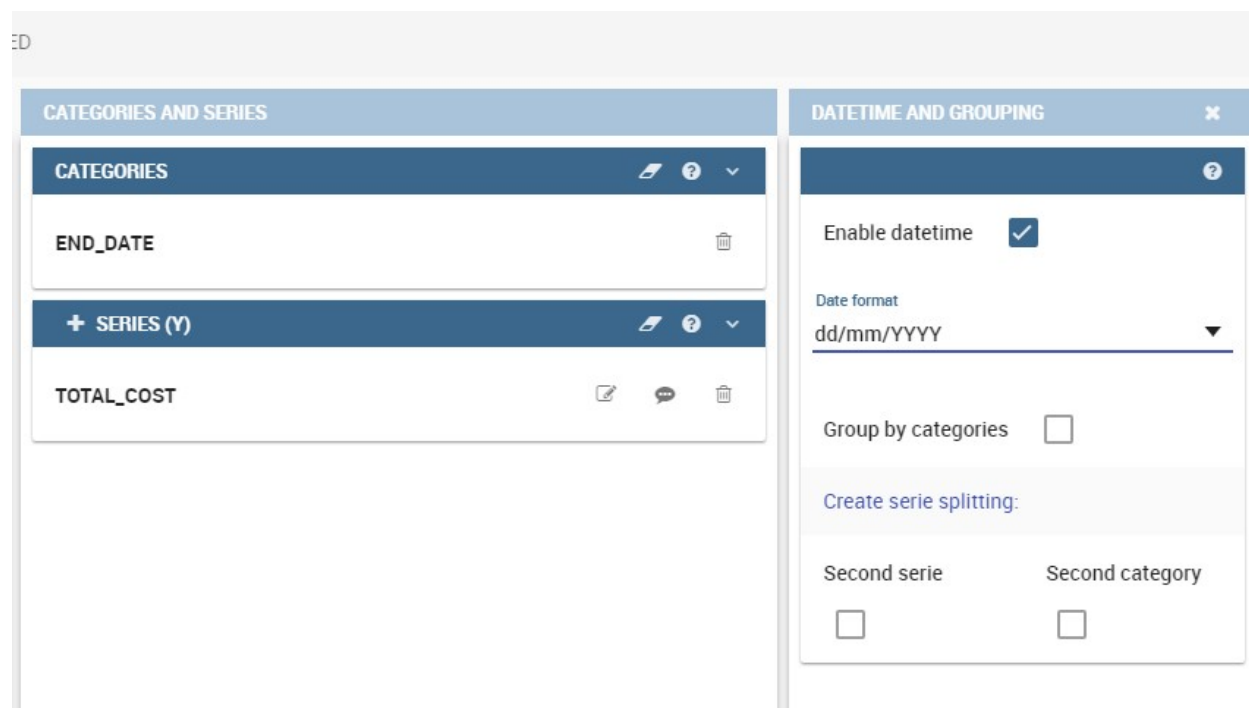


Fig. 1.103: Datetime function usage.

For bar and line chart you can add more then one container for adding series in **Structure** section. In that case you will have in your chart more then one axis for series. In **Advanced** section you can specify to align these axis to 0 (zero) value. It is check box **alignAxis** where checked means that axes will be aligned to 0, and unchecked means that they will not be aligned.

For pie chart inside **Advanced** section you can set configuration for your tooltip: to show/hide absolute value and/or percentage. Inside **tooltip** property of serie object you can find properties **showAbsValueTooltip** and **showPercentageTooltip**.

Scatter chart

A scatter chart is a graphical representation of scattering phenomenon of data. It is useful when the user wants to underlight the density of data upon certain spots to the detriment of readability of single points. If you select a scatter chart in the **Configuration** section you will have Ticks and Lables Details instead of Advanced Series Configuration. Be carefull to fill in the **Scatter configuration** with the **Zoom type**, as showed below.

You must check if you want that the values in the Y-axis start (or end) in the first (last) tick or in the first (last) value of the dataset and if you want that the last label of the category axis should be showed.

Sunburst chart

The sunburst chart is a graph with a radial layout which depicts the hierarchical structure of data displaying a set of concentric rings. The circle in the center represents the root nodes, with the hierarchy moving outward from the center. The slices in the external rings are children of the slice in the inner circle which means they lie within the angular sweep of the inner circle. The area of each slice corresponds to the value of the node. Even if sunburst charts are not efficient space-wise, they enable the user to represent hierarchies in a more immediate and fascinating way.

To create a sunburst chart in Knowage you just have to select a dataset with at least two attribute columns describing the hierarchy and at least a measure column that indicates the width of the slices. An example of dataset for the sunburst chart is showed in Table below.

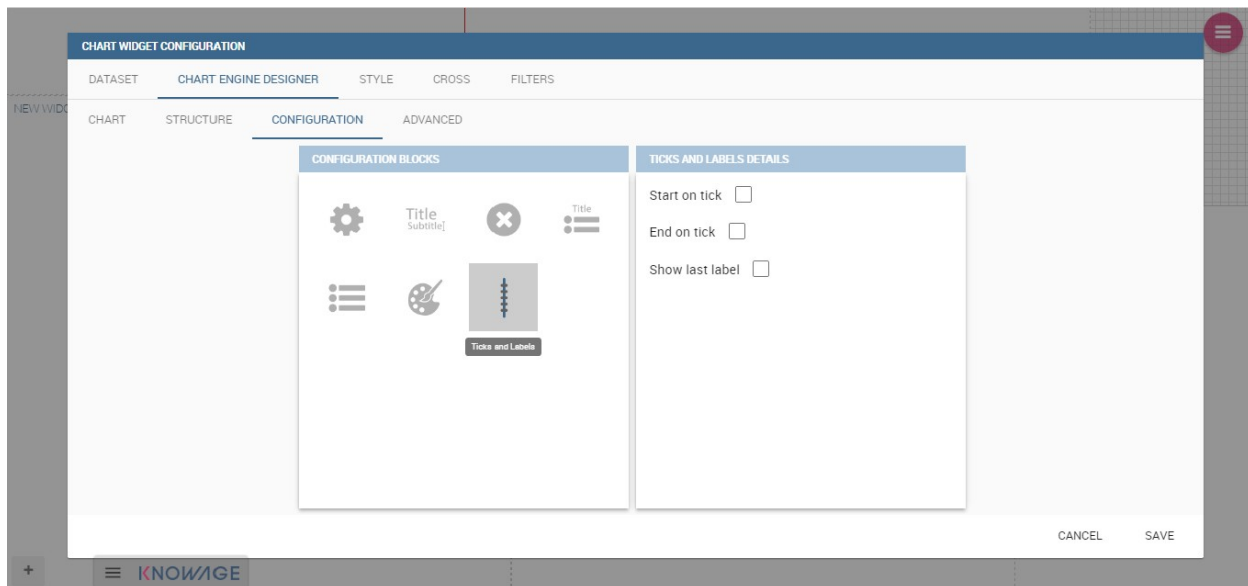


Fig. 1.104: Scatter Chart, ticks and labels details.

Table 1.5: Example of dataset for the sunburst chart.

CATEGORY	SUBCATEGORY	UNIT
Baking Goods	Cooking Oil	349
Baking Goods	Sauces	109
Baking Goods	Spices	290
Baking Goods	Sugar	205
Bathroom Products	Conditioner	64
Bathroom Products	Mouthwash	159
Bathroom Products	Shampoo	254
Bathroom Products	Toilet Brushes	92
Bathroom Products	Toothbrushes	94

Once you selected the dataset and the type of chart, choose at least two attributes in the X-axis panel and a measure in the Y-axis panel as showed in the following figure.

Then click on **Configuration**. As you can see the features are not exactly the same as traditional chart. We give some tips on most important sunburst settings.

Using the **Generic** button you can set the **opacity** on mouse movement and choose how to display the measure values: absolute, percentage or both. These two features allow the visualization of data just moving the mouse over the slice: the slice is highlighted and values are shown in the center of the ring while the root-node path for the node selected is displayed on the left bottom corner of the page. Opacity and Breadcrumb configuration are available only on Community Edition. The tooltip is a mandatory field since it shows the value of the selected slice. Therefore be sure to have filled it before saving by using the **Explanation detail** panel. On Community Edition you have option custom the root-node path, clicking on the **Sequence** icon and choose position, label tail size and text style. **Sequence** option is not available on Enterprise edition, it is deprecated. Figure below sums up the three features.

In Figure below you find the sunburst obtained with data of `exampleofdatasetsunburst`.

Inside **Advanced** section you can set value for scale that will increase/decrease your chart. You need to set numeric value for property **scale**.

Wordcloud chart

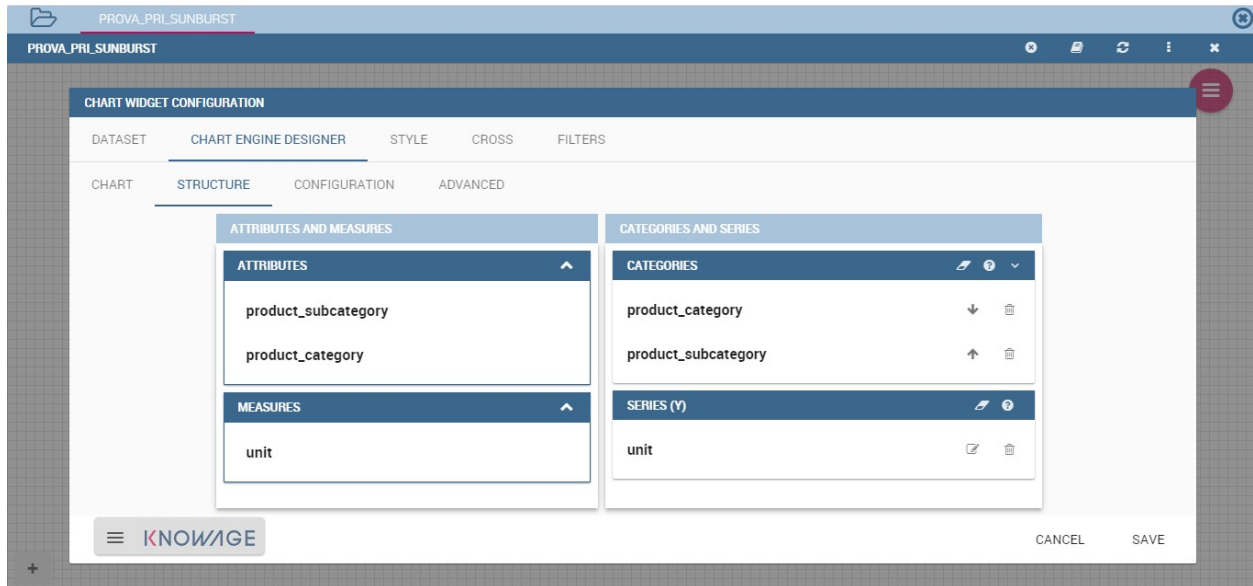


Fig. 1.105: Sunburst configuration.

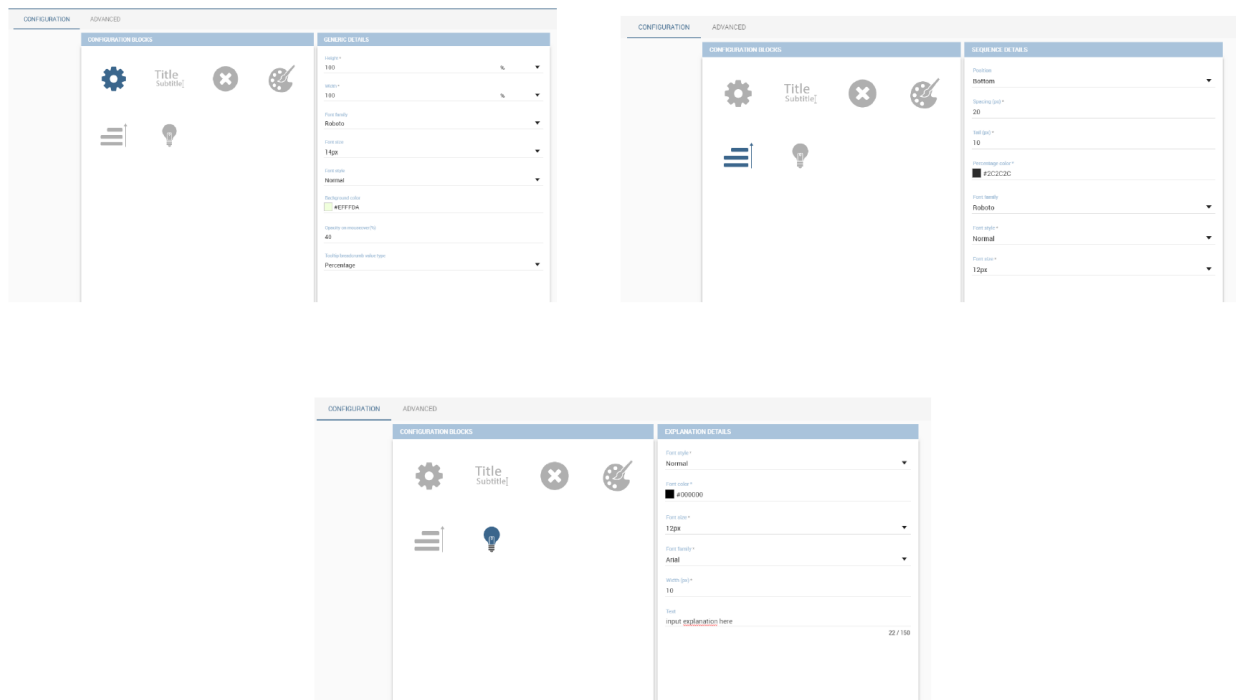


Fig. 1.106: Generic, Sequence and Explanation configuration

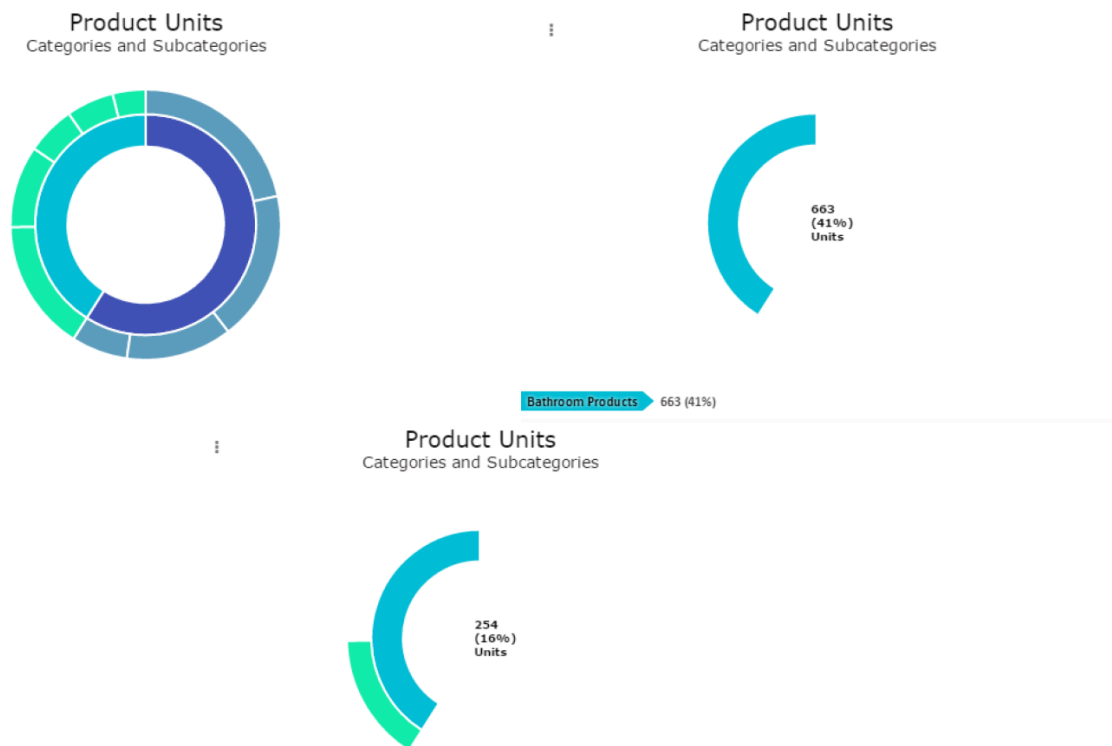


Fig. 1.107: From left to right: (a) Sunburst. (b) Sunburst category.(c) Sunburst subcategory.

The wordcloud chart is a graphic to visualize text data. The dimension of the words and colors depend on a specified weight or on the frequency of each word.

The dataset to create a wordcloud should have at least a column with attributes and only one column with numerical data which represents the weight to assign to each attribute. Choose one attribute as category field (the wordcloud accept only one attribute in the category box) and a measure as series field.

Switch to the **Configuration** section to set the generic configuration of the chart and to custom fields of the **Word settings details**. Here the use can decide if to resize the words accordingly to the measure retrieved in the dataset (**Series** option) or accordingly to the frequency of the attributes in the dataset (**Occurrences** option). Moreover it is possible to set the maximum number of words that you want to display, the padding between the words, the word layout and whether or not you want to prevent overlap of the words as showed in Figure below.

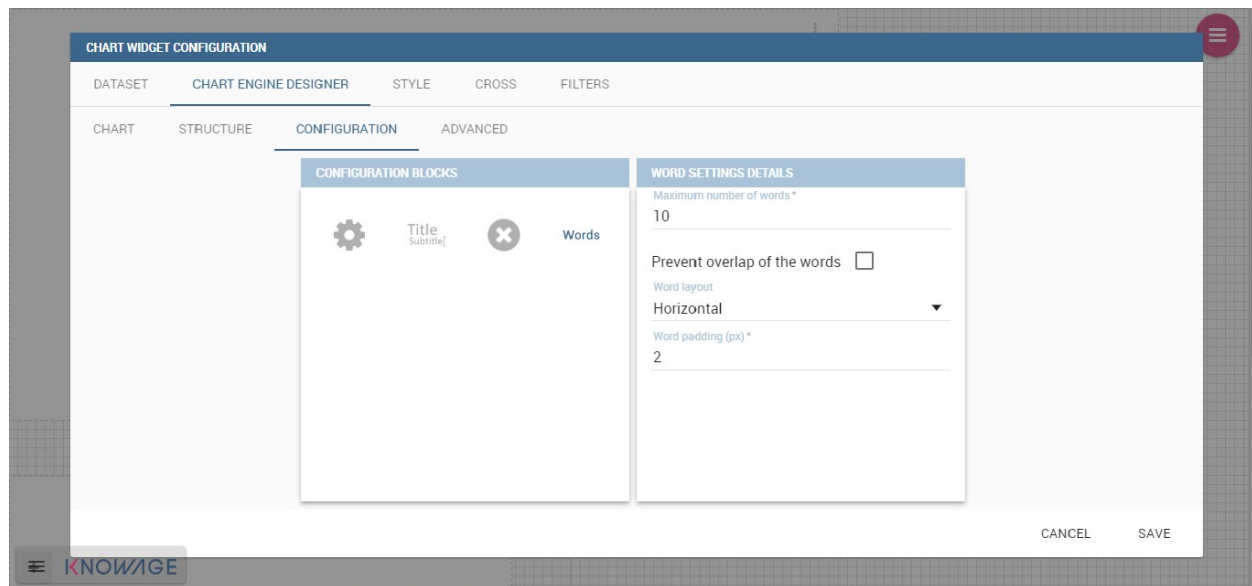


Fig. 1.108: Wordcloud chart specific configuration.

Treemap chart

The treemap is a graphical representation of hierarchical data, which are displayed as nested rectangles. Each branch of the tree is given by a rectangle, which is tiled with smaller rectangles representing sub-branches. The area of the rectangles is proportional to a measure specified by a numerical attribute. The treemap is usefull to display a large amount of hierarchical data in a small space.

To create a treemap chart you have to select a dataset as the one described for the sunburst chart in the Parallel chart.

Once you have selected the dataset, choose the treemap chart type in the designer and then at least two attributes into the X-axis panel. The order of the attributes in the X-axis panel must reflects the order of the attributes in the hierarchy starting from the root to the top.

Finally you can set generic configurations and colors palette in the **Configuration** tab and advanced configurations in **Advanced editor** tab.

In Figure below we show the Treemap resulting with data of our example

Parallel chart

The parallel chart is a way to visualize high-dimensional geometry and multivarious data. The axes of a multidimensional space are represented by parallel lines, usually equally spaced-out, and a point of the space is represented by a broken line with vertices on the parallel axes. The position of the vertex on an axis correspond to the coordinate of the point in that axis.

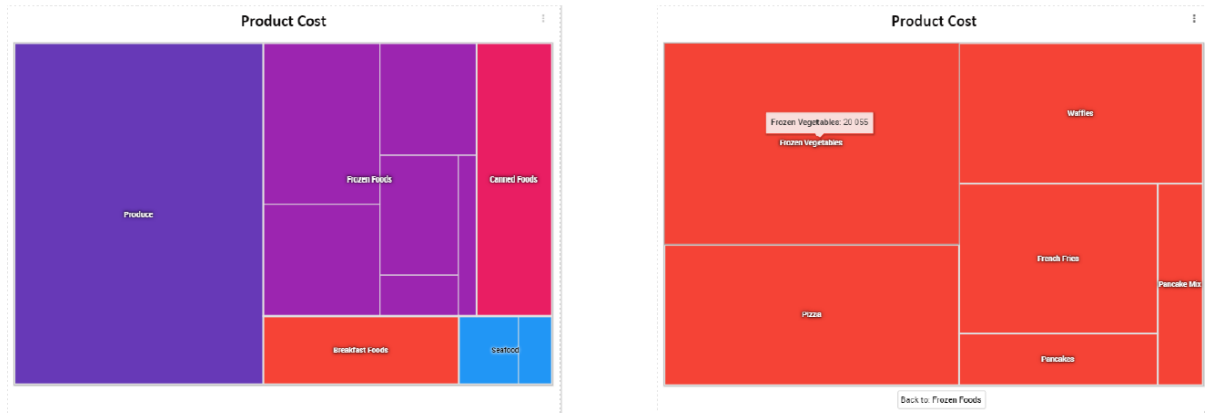


Fig. 1.109: From left to right: (a) Treemap. (b) Treemap sub-branches.

To create a parallel chart select a dataset with at least one attribute and two columns with numerical values. You can find an interesting example of dataset in the next table where we display some of its rows.

Table 1.6: Example of dataset for the parallel chart.

ID	sepal_length	sepal_width	petal_length	petal_width	class
36	5.0	3.2	1.2	0.2	Iris-setosa
37	5.5	3.5	1.3	0.2	Iris-setosa
38	4.9	3.1	1.5	0.1	Iris-setosa
39	4.4	3.0	1.3	0.2	Iris-setosa
40	5.1	3.4	1.5	0.2	Iris-setosa
41	5.0	3.5	1.3	0.3	Iris-setosa
42	4.5	2.3	1.3	0.3	Iris-setosa
43	4.4	3.2	1.3	0.2	Iris-setosa
44	5.0	3.5	1.6	0.6	Iris-setosa
45	5.1	3.8	1.9	0.4	Iris-setosa
66	6.7	3.1	4.4	1.4	Iris-versicolor
67	5.6	3.0	4.5	1.5	Iris-versicolor
68	5.8	2.7	4.1	1.0	Iris-versicolor
69	6.2	2.2	4.5	1.5	Iris-versicolor
70	5.6	2.5	3.9	1.1	Iris-versicolor
71	5.9	3.2	4.8	1.8	Iris-versicolor
101	6.3	3.3	6.0	2.5	Iris-virginica
102	5.8	2.7	5.1	1.9	Iris-virginica
103	7.1	3.0	5.9	2.1	Iris-virginica
104	6.3	2.9	5.6	1.8	Iris-virginica
105	6.5	3.0	5.8	2.2	Iris-virginica
106	7.6	3.0	6.6	2.1	Iris-virginica
107	4.9	2.5	4.5	1.7	Iris-virginica
108	7.3	2.9	6.3	1.8	Iris-virginica

In this example three different classes of iris are studied. Combining the values of some sepal and petal width or length, we are able to find out which class we are looking at. In Figure below (a part) you can find the parallel chart made with the suggested dataset. While in next figure (b part) it is easy to see, thanks to selection, that all iris with petal length between 2,5 and 5,2 cm and petal width 0,9 and 1,5 cm belong to the iris-versicolor class.

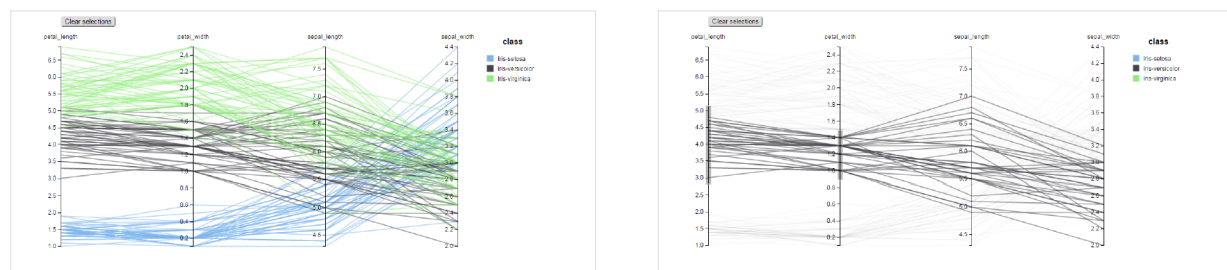


Fig. 1.110: From left to right: (a) Parallel. (b) Parallel chart selection.

Therefore, select **parallel** as chart type using the designer interface, then choose one or more attributes in the X-axis panel and one or more measures in the Y-axis panel.

On the **Configuration** tab you can set the generic configuration for the chart and you must fill the **Series as filter column** filed under "Limit configuration". Under "Tooltip configuration" there is new property available - **Maximum number of records to show tooltip**. It is used to limit showing tooltip in case there are lot of records returned from dataset, which make chart more readable.

Heatmap chart

Heatmap chart uses a chromatic Cartesian coordinate system to represent a measure trend. Each point of the Cartesian system is identified by a couple of attributes. Note that one attribute must be a datetime one. Meanwhile, each couple corresponds to a measure that serves to highlight the spot with a certain color according to the chosen gradient. Figure below gives an example of how an heatmap chart looks like inside Knowage.

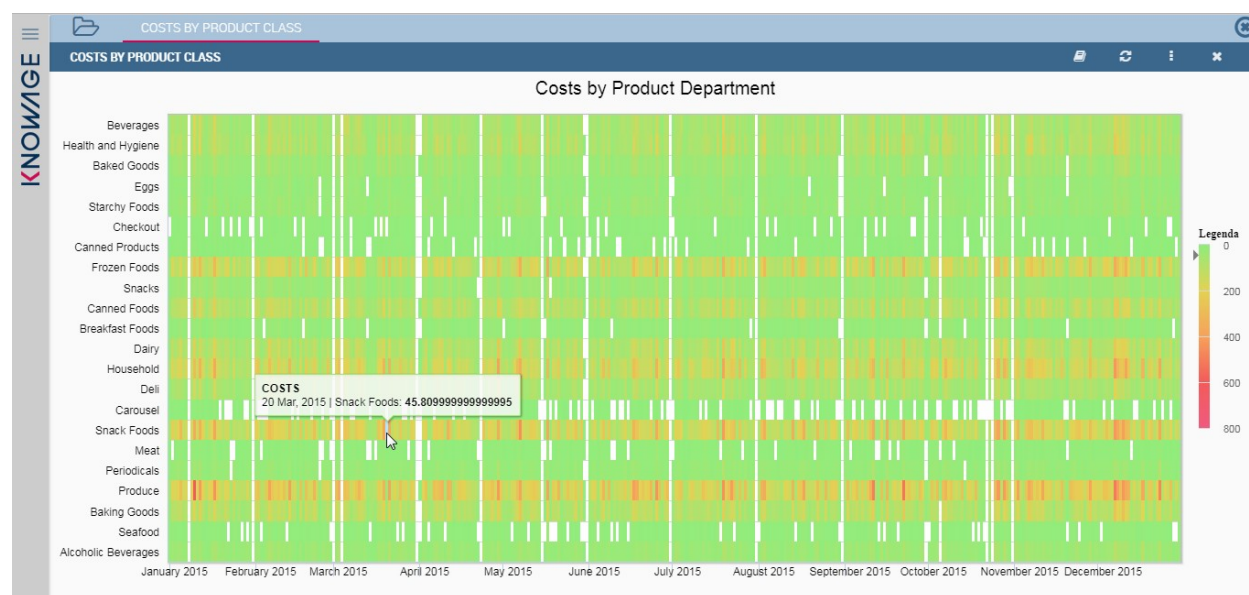


Fig. 1.111: Heatmap example.

Before configuring a heatmap chart, be sure that your dataset returns at least two attributes, one of which **must** be a datetime one, and (at least) one measure. Once entered the chart designer, choose the "Heatmap" type and move to the "Structure" tab. Use the datetime attribute and an other attribute as category fields and one measure as series fields. Figure below shows an example.

Note that for series axis it is possible to specify the values' range by assigning a minimum and the maximum value, as shown in figure below. Otherwise, the engine will automatically link the axis scale to dataset results set.

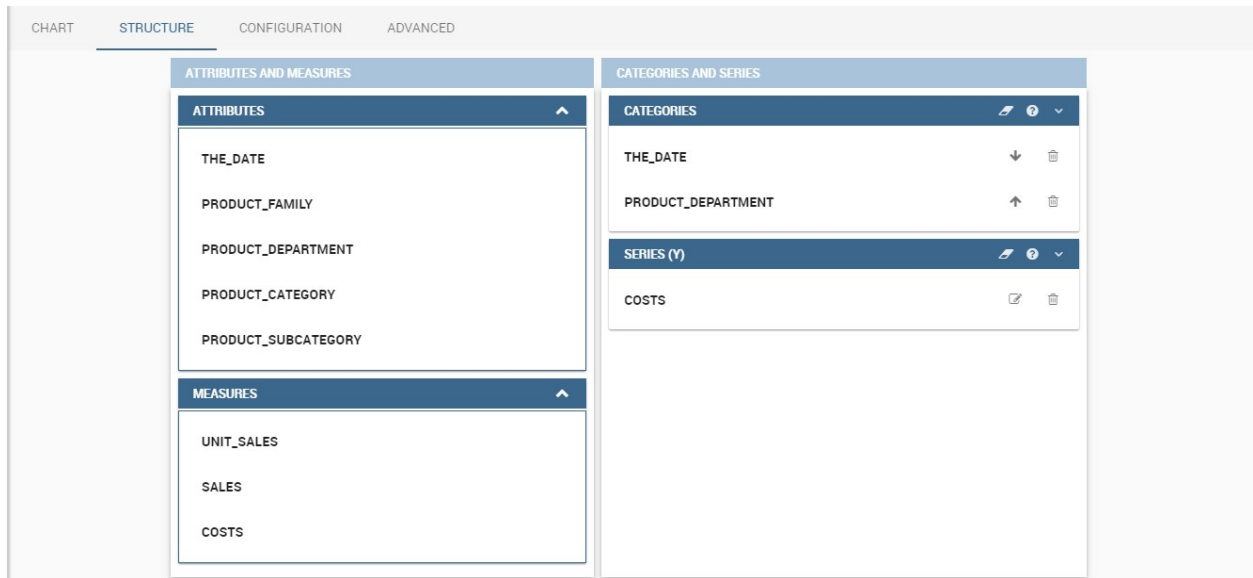


Fig. 1.112: Configuring the attributes and the series for the heatmap chart.

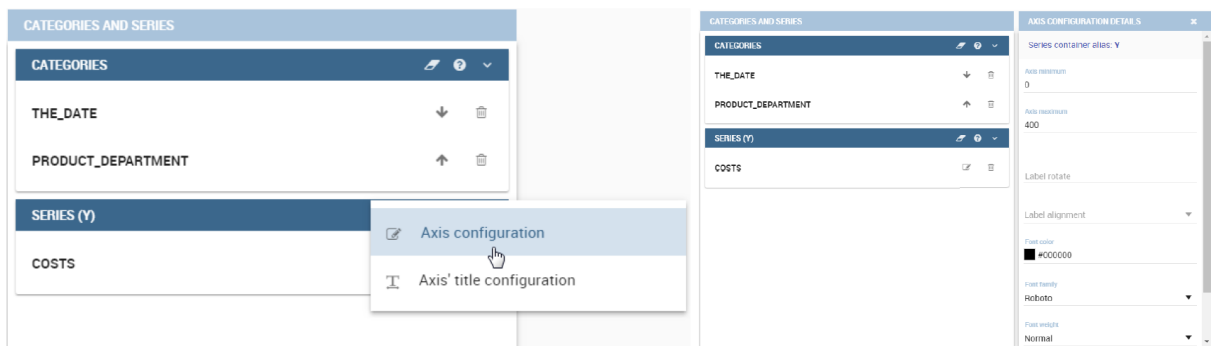


Fig. 1.113: Configure min and max values for series.

The next step is to move to **Configuration** tab and select the **Color palette** icon. Here (figure below) the user has to define the chromatic scale which will be associated to the measure values. The panel will demand the user to insert the first, the last color and the number of bands that will constitute the color scale.

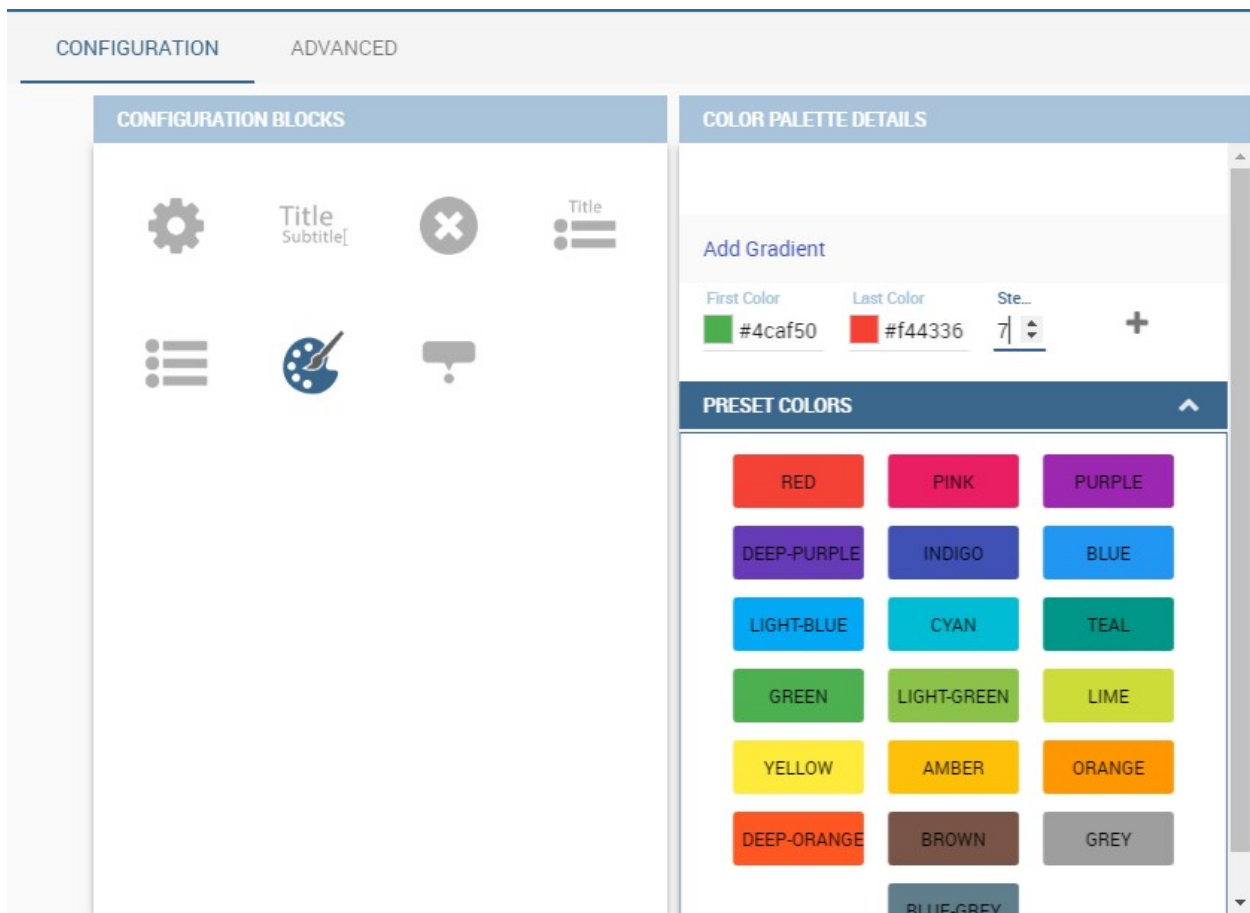


Fig. 1.114: Add gradient panel.

The engine will create a progressive color scale as shown in the left image of figure below. To custom the scale the user can use the Preset colors and use the arrow to move up and down Heatmap chart the added color or the user can increase the number of steps and then some intermediate color to leave more contrast between them.

Remember to edit both **Legend** and **Tooltip** configuration in the **Tooltip details** panel to improve the readability of the chart.

Chord chart

Chord diagram is a graph which allows to show relationship between entities and between data in a matrix. The entities can belong to an unique category while the arc be non-oriented or belong to two different categories. In this latter case, they have direct arcs. The data are arranged radially with arcs that represent the connection between points. The width of the arc connecting two points depends on the weight assigned to the edge connecting these two points. This graphic is usefull when you want to represent a large number of data in a small space.

The chord diagram requires a dataset that have a column with numerical values. These represent the weight of the arc connecting two points. It also must have two columns with the entries for the entities to be connected in the diagram. These two columns must have the same set of values so that the engine can understand the relation between all the entities. If there is not a relation between two entities the weight of the arc is zero. Note that when you create a directed chord diagram with two different categories, all the relations between entities of the same category have a zero weight.

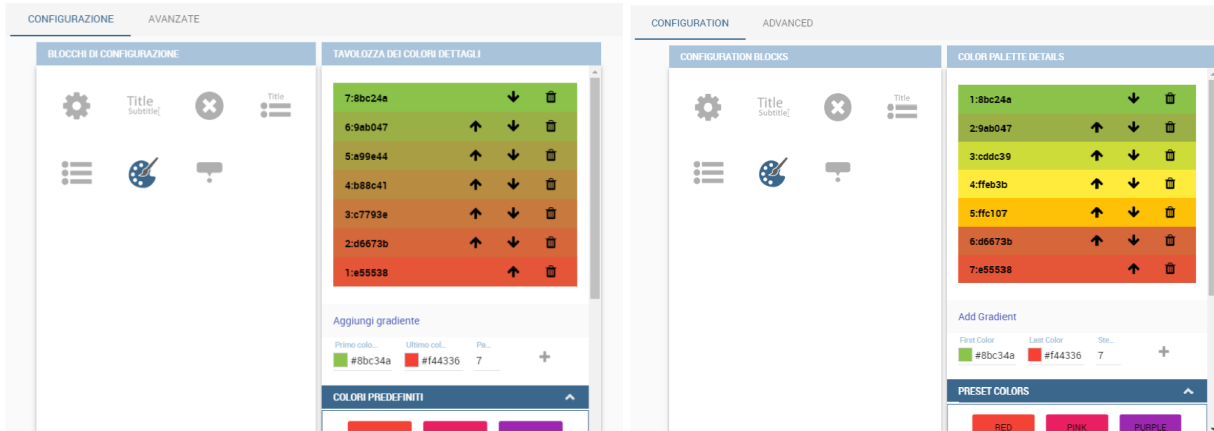


Fig. 1.115: Custom color scale.

An example of dataset for the chord chart is represented in Table below:

Table 1.7: Example of dataset for the chord chart.

CUSTOMER_ CITY	STORE_ CITY	VALUE
Beaverton	Portland	4609.0000
Lake Oswego	Portland	4201.0000
Milwaukie	Portland	5736.0000
Oregon City	Portland	3052.0000
Portland	Portland	3984.0000
W. Linn	Portland	3684.0000
Albany	Salem	5544.0000
Corvallis	Salem	8542.0000
Lebanon	Salem	8015.0000
Salem	Salem	6910.0000
Woodburn	Salem	6335.0000
Albany	Albany	0.0000
Beaverton	Beaverton	0.0000
Corvallis	Corvallis	0.0000
Lake Oswego	Lake Oswego	0.0000
Lebanon	Lebanon	0.0000
Milwaukie	Milwaukie	0.0000
Oregon City	Oregon City	0.0000
Portland	Portland	0.0000
Salem	Salem	0.0000
W. Linn	W. Linn	0.0000

Once you have selected the dataset open the designer and select chord chart type. Then choose the two entities in the X-axis panel and the value in the Y-axis panel as showed in figure below. Now you are ready to customize the chart setting the generic configuration and the palette on **Configuration**.

Gauge chart

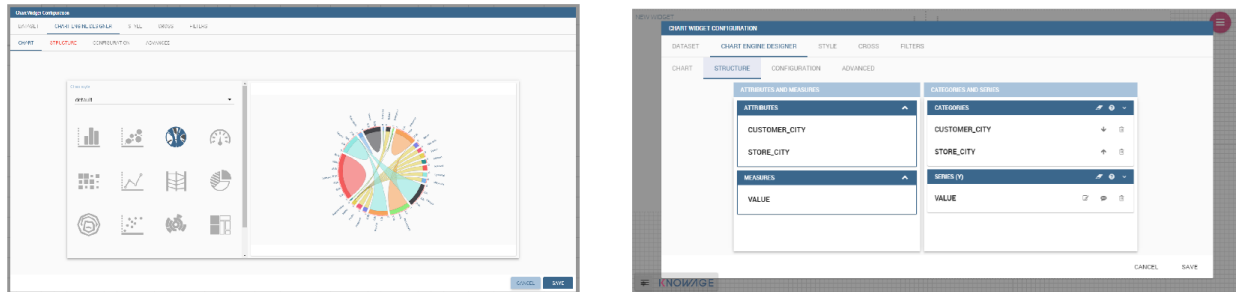


Fig. 1.116: Chord configuration.

Gauge chart uses needles to show information as a dial reading. It allows to visualize data in a way that resembles a real-life speedometer needle. The value of the needle is read on a colored data scale. Colors are used to provide additional performance context (typically green for good and red for bad). This chart type usually is used in dashboards to show key performance indicators or any measure having reference values.

For gauge chart you should have only series items, the one that gives you values for the chart. So, the defined dataset to be used should provide numerical data for the Y-axis for the gauge chart. After selecting the dataset go to the designer and select **gauge** in chart type combobox. Then choose one or more measure on the Y-axis panel on the **Structure**. Moreover you must not forget to provide all data needed for the **Axis style configuration** of the Y-axis.

When you finished to set all the mandatory and optional parameters and configurations in the **Structure** tab you can select the **Configuration** tab and set the generic configuration of the chart.

Bubble chart

A bubble chart requires three dimensions of data; the x-value and y-value to position the bubble along the value axes and a third value for its volume, z-value. It is a generalization of the scatter plot, replacing the dots with bubbles.

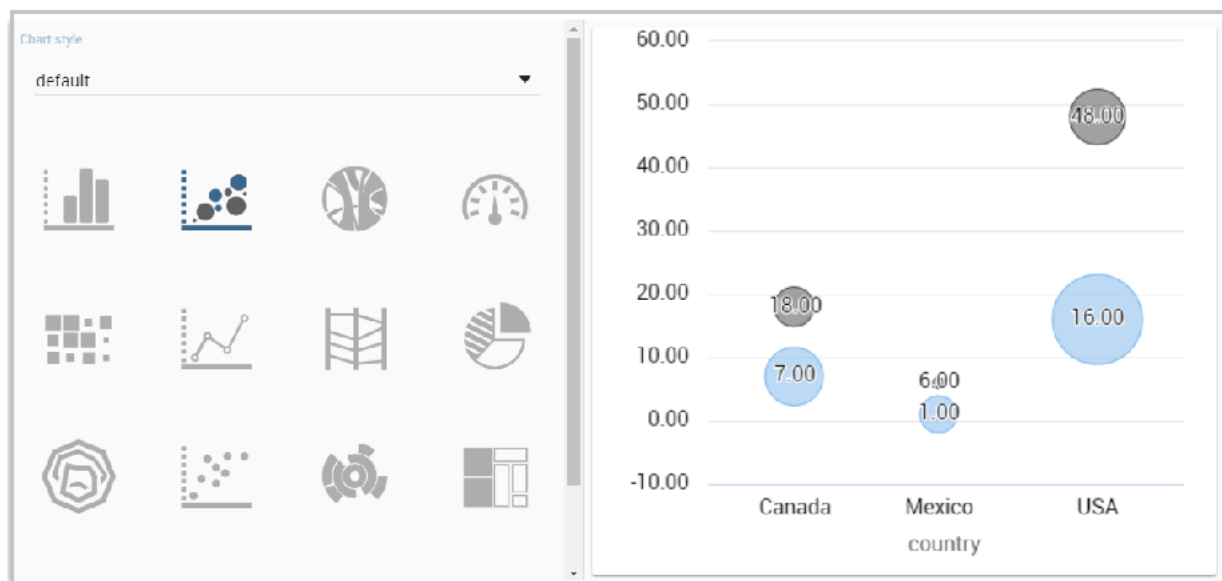


Fig. 1.117: Bubble chart.

Inside X,Y,Z containers, user can put only **measure values**. Inside Categories container user can put **attributes** that he wants to see in the **tooltip**.

Advanced functionalities

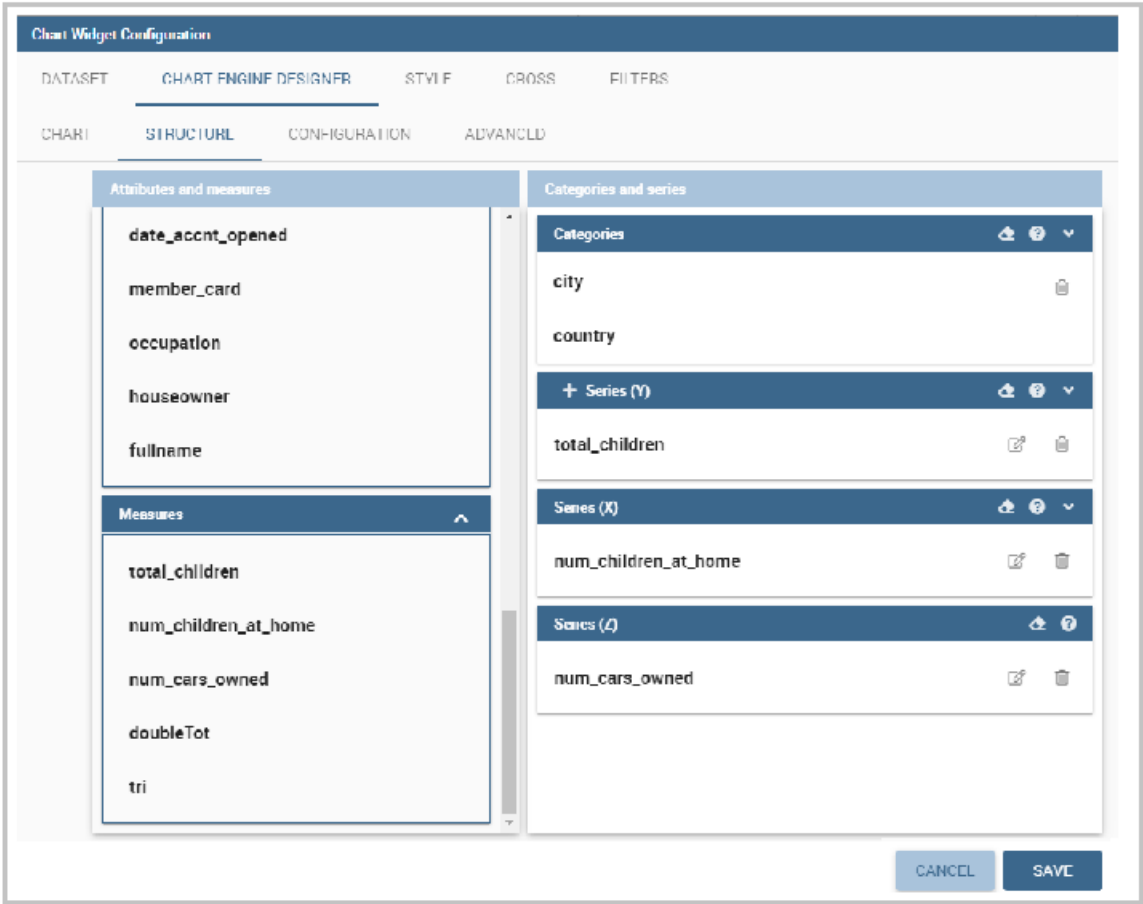


Fig. 1.118: Bubble configuration.

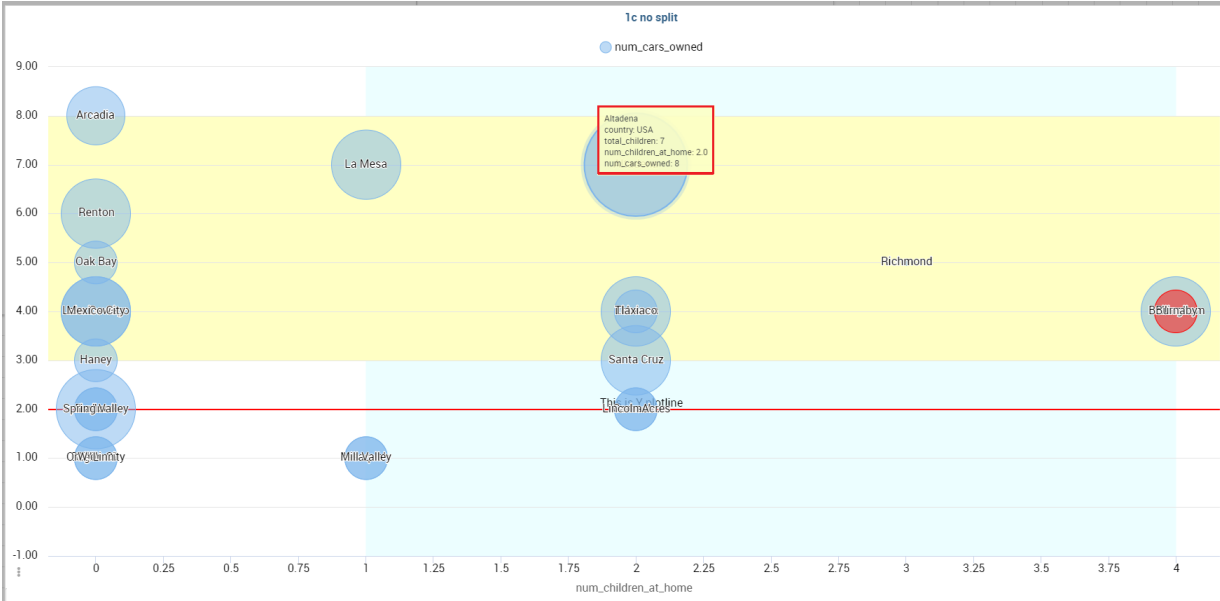


Fig. 1.119: Bubble tooltip.

Inside **Advanced tab**, user can find configuration for **plotband** and **plotline** of xaxis and yaxis.

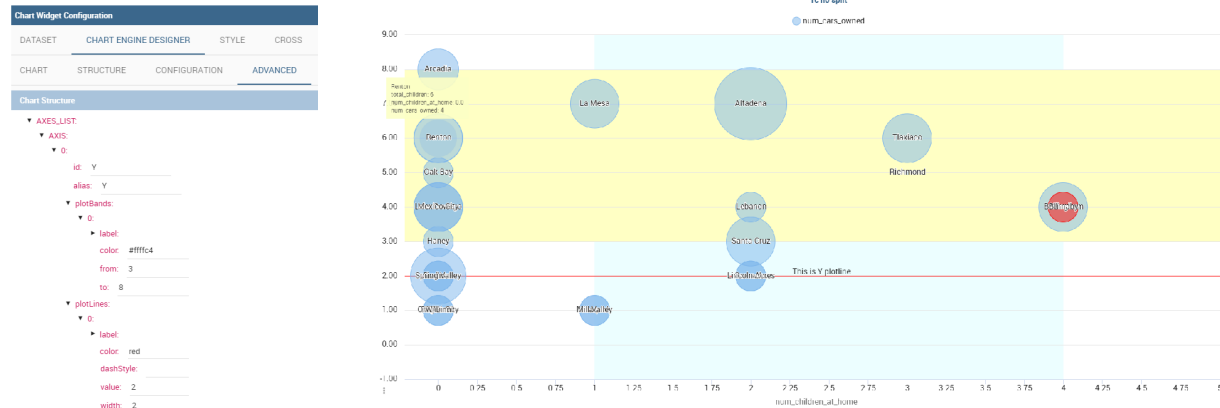


Fig. 1.120: Bubble plotband and plotline configuration.

User also has option for **splitting serie** by one of two categories. Split option has limit on **maximum** two categories. User can set which category will be used for coloring bubbles. And also can show/hide values that are inside bubbles.

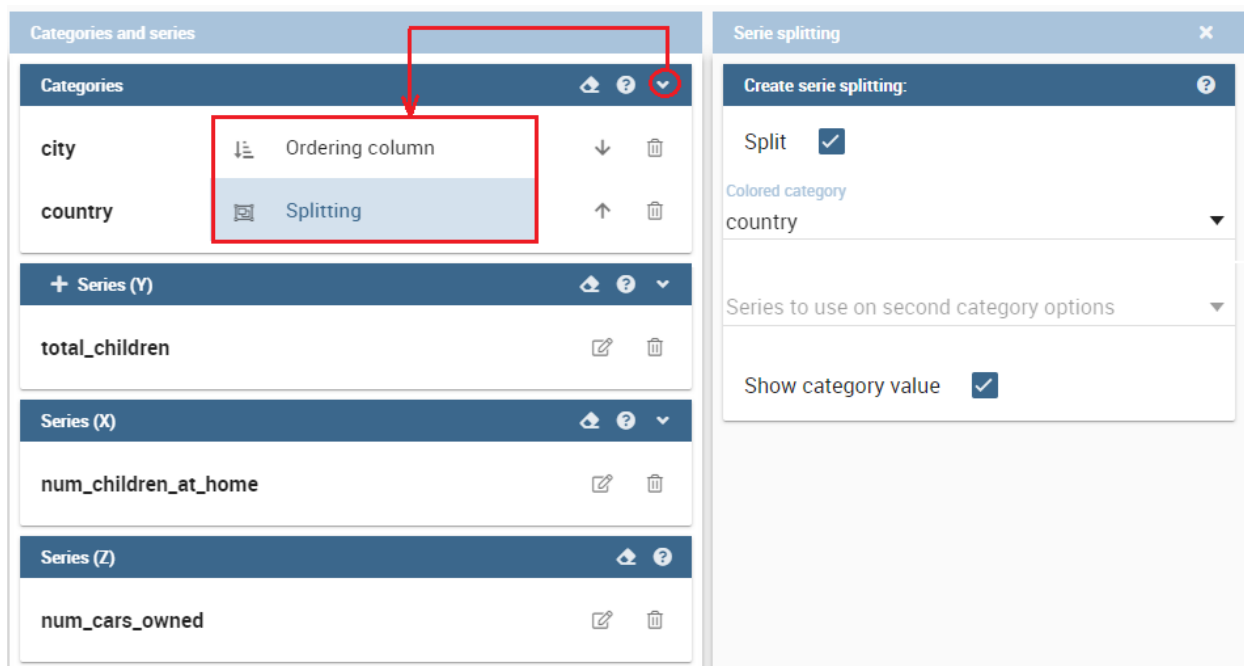


Fig. 1.121: Bubble split configuration.

Difference between **One category - split disabled** and **Two categories - split disabled** is what is present in the tooltip.

Knowledge **Chart Engine** allows you to drill down into categories. This means that the user can explore the details of each category as many times as configured. Indeed, to let the chart admits the drill down, it is necessary first that the chart in place allows it. Secondly the user must have dragged and dropped multiple attributes into the category axis in the **Configuration** tab. The order of the attributes in the X-axis panel determines the sequence in which the categories are going to be showed. When executing the chart the label of the category is linkable and it is possible to click on the label to drill down.

The chart that enables the drill down are:



Fig. 1.122: Bubble split examples.

- Bar Chart
- Line Chart
- Pie Chart
- Treemap

To give an idea of the outcome, we take as instance the Bar Chart drill down. In the following example, the selected categories are four and called: `product_family`, `product_department`, `product_category` and `product_subcategory`. Once we open the document, we get as shown below:

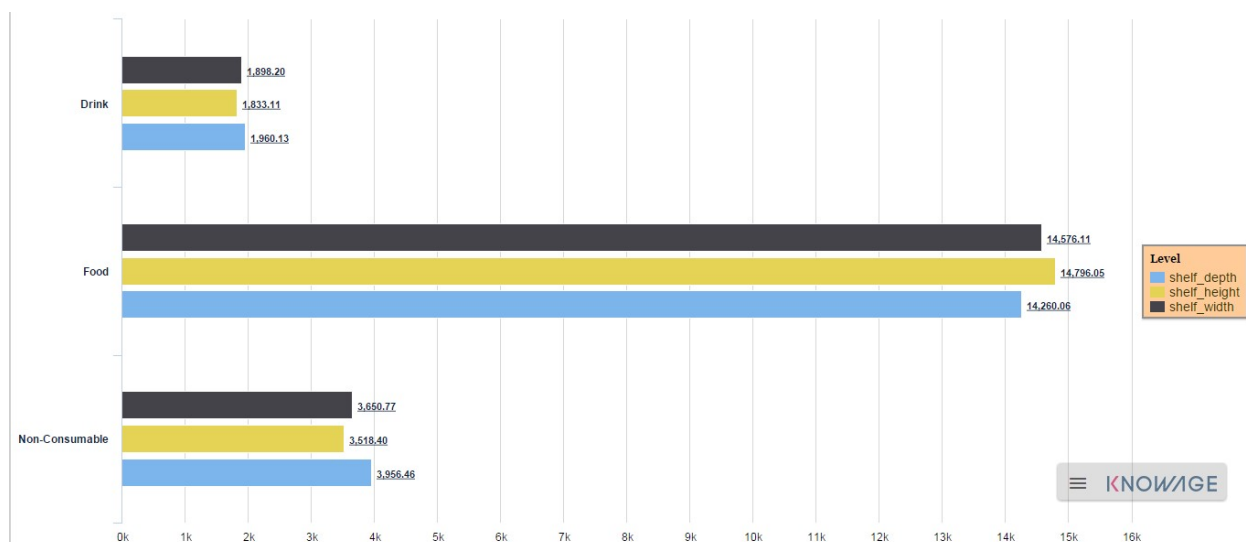


Fig. 1.123: Drillable Bar Chart

When selecting `shelf_depth` measure of the Food category one gets (see next figure):

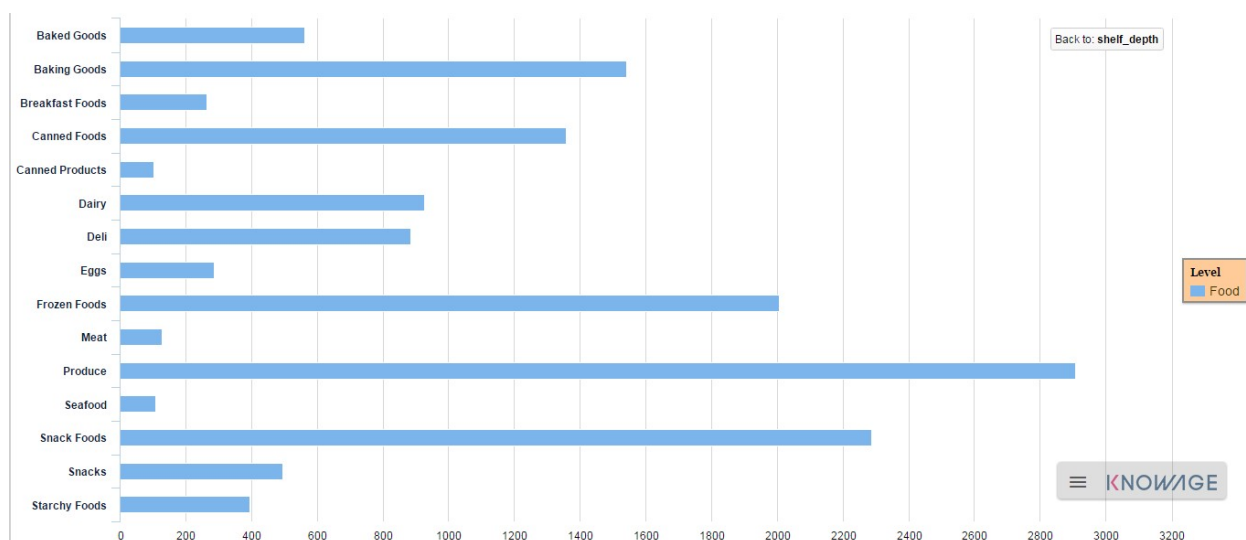


Fig. 1.124: Drillable Bar Chart: first drill

Once again, we can select `Frozen` food subcategory and drill to a second sub-level as below:

And so on to the fourth subcategory. Selecting the “Back to: ...” icon available at the right corner of the graphic, the

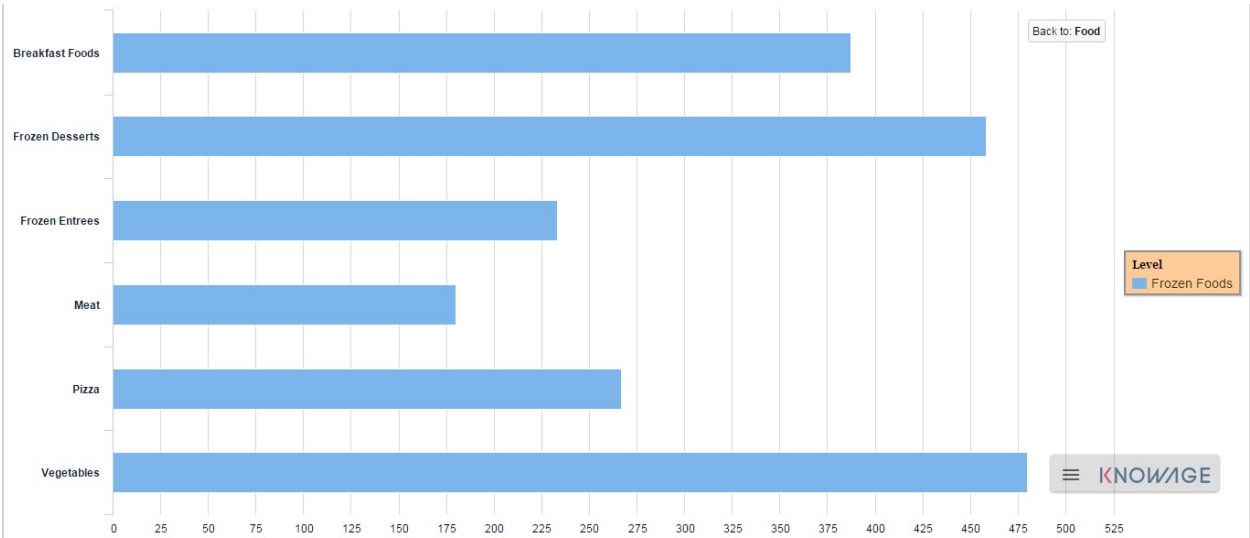


Fig. 1.125: Drillable Bar Chart: second drill

user can get back to the previous level. This efficient feature allows the user to have a deep insight of the analysis and draw important conclusions from it.

1.4.1.4 Table widget

The **Widget table configuration** opens and it guides you through the steps to configure the widget. The pop up opens showing the **column** tab, as you can see from Figure below. In details, it is mandatory to select a dataset using the combobox (only if at least one dataset has been loaded using the **Data Configuration** feature) or clicking on the icon **+** available just aside the combobox line. You can page the table specifying the number of rows per sheet. Consequently the user can set columns properties.

Table Widget Configuration

COLUMNS

STYLE

CROSS

FILTERS

Dataset

+

Pagination

Max Rows Number

10

Frontend pagination

TABLE COLUMNS

MANAGE COLUMN GROUPS

ADD COLUMN

ADD CALCULATED FIELD

USE FUNCTION

Sorting column

Sorting order

Summary row

ADD

CANCEL

SAVE

Fig. 1.126: Table configuration window.

In fact, the column area is divided into two parts: on the upper part there are the buttons to add a new column or a calculated field. Also the new functionality of column grouping is available here. In the lower section, as soon as the dataset is selected, you can indicate the sorting column or modal selection column. The modal selection serves to specify which value will be passed to other widgets (if interaction is enabled) when clicking on the cell at document execution time. You can specify this field by selecting a value from the combobox. In the same way, you indicate the sorting column and the order type that steers the rows ordering. You can select the field and the order from the dedicated comboboxes.

When a dataset is added to a table widget, all of its columns are listed below. If the user doesn't wish to show some of them, he can use the delete button available at the end of each column row, as shown below.

Table Widget Configuration

COLUMNS STYLE CROSS FILTERS

TABLE COLUMNS MANAGE COLUMN GROUPS ADD COLUMN ADD CALCULATED FIELD USE FUNCTION

Name	Alias	Type	Data Type	Aggregation	
<input checked="" type="checkbox"/> employee_id	<input checked="" type="checkbox"/> employee_id	<input checked="" type="checkbox"/> MEASURE	# integer	<input checked="" type="checkbox"/> SUM	
<input checked="" type="checkbox"/> full_name	<input checked="" type="checkbox"/> full_name	<input checked="" type="checkbox"/> ATTRIBUTE	string		
<input checked="" type="checkbox"/> first_name	<input checked="" type="checkbox"/> first_name	<input checked="" type="checkbox"/> ATTRIBUTE	string		
<input checked="" type="checkbox"/> last_name	<input checked="" type="checkbox"/> last_name	<input checked="" type="checkbox"/> ATTRIBUTE	string		
<input checked="" type="checkbox"/> position_id	<input checked="" type="checkbox"/> position_id	<input checked="" type="checkbox"/> MEASURE	# integer	<input checked="" type="checkbox"/> SUM	
<input checked="" type="checkbox"/> position_title	<input checked="" type="checkbox"/> position_title	<input checked="" type="checkbox"/> ATTRIBUTE	string		
<input checked="" type="checkbox"/> store_id	<input checked="" type="checkbox"/> store_id	<input checked="" type="checkbox"/> MEASURE	# integer	<input checked="" type="checkbox"/> SUM	
<input checked="" type="checkbox"/> department_id	<input checked="" type="checkbox"/> department_id	<input checked="" type="checkbox"/> MEASURE	# integer	<input checked="" type="checkbox"/> SUM	
<input checked="" type="checkbox"/> birth_date	<input checked="" type="checkbox"/> birth_date	<input checked="" type="checkbox"/> ATTRIBUTE	date		
<input checked="" type="checkbox"/> hire_date	<input checked="" type="checkbox"/> hire_date	<input checked="" type="checkbox"/> ATTRIBUTE	timestamp		
<input checked="" type="checkbox"/> end_date	<input checked="" type="checkbox"/> end_date	<input checked="" type="checkbox"/> ATTRIBUTE	timestamp		
<input checked="" type="checkbox"/> salary	<input checked="" type="checkbox"/> salary	<input checked="" type="checkbox"/> MEASURE	# float	<input checked="" type="checkbox"/> SUM	
<input checked="" type="checkbox"/> supervisor_id	<input checked="" type="checkbox"/> supervisor_id	<input checked="" type="checkbox"/> MEASURE	# integer	<input checked="" type="checkbox"/> SUM	
<input checked="" type="checkbox"/> education_level	<input checked="" type="checkbox"/> education_level	<input checked="" type="checkbox"/> ATTRIBUTE	string		
<input checked="" type="checkbox"/> marital_status	<input checked="" type="checkbox"/> marital_status	<input checked="" type="checkbox"/> ATTRIBUTE	string		
<input checked="" type="checkbox"/> gender	<input checked="" type="checkbox"/> gender	<input checked="" type="checkbox"/> ATTRIBUTE	string		
<input checked="" type="checkbox"/> management_role	<input checked="" type="checkbox"/> management_role	<input checked="" type="checkbox"/> ATTRIBUTE	string		
<input checked="" type="checkbox"/> female_salary	<input checked="" type="checkbox"/> female_salary	<input checked="" type="checkbox"/> MEASURE	# float	<input checked="" type="checkbox"/> SUM	
<input checked="" type="checkbox"/> male_salary	<input checked="" type="checkbox"/> male_salary	<input checked="" type="checkbox"/> MEASURE	# float	<input checked="" type="checkbox"/> SUM	

☐ Summary row

ADD

CANCEL SAVE

Fig. 1.127: Delete a column.

In case of accidental cancellation or new table requirements, it is possible to re-add columns. In order to add a new column you have to click on the **Add Column** icon on the top right of the second box. Once opened you can select one or more columns. When you have finished selecting the desired columns you can click on save button and your new columns will appear in the field list. Refer to Figure below.

ADD COLUMN

Search

Column	Field Type	Type
<input checked="" type="checkbox"/> employee_id	MEASURE	java.lang.Integer
<input type="checkbox"/> full_name	ATTRIBUTE	java.lang.String
<input type="checkbox"/> first_name	ATTRIBUTE	java.lang.String
<input type="checkbox"/> last_name	ATTRIBUTE	java.lang.String
<input type="checkbox"/> position_id	MEASURE	java.lang.Integer
<input type="checkbox"/> position_title	ATTRIBUTE	java.lang.String
<input type="checkbox"/> store_id	MEASURE	java.lang.Integer

1 to 7 of 19 |< < Page 1 of 3 > >|

CANCEL SAVE

Fig. 1.128: Add a new column.

Manage Columns Groups will open a menu to add or remove columns groups and to set their style. A column group

is a container for more than one column that will show a common header between them.

testGroup					
id	first_name	int	email	gender	float
766.00	Abigail	7,721.00	awykel9@pen.io	Female	41.35
626.00	Adah	6,358.00	apierreponthd@sun.com	Female	91.35
270.00	Adams	7,206.00	afinham7h@nature.com	Male	4.48
68.00	Adara	1,030.00	agibling1v@cnn.com	Female	61.65
408.00	Addy	8,793.00	asartonbb@alibaba.com	Male	84.19
868.00	Adeline	4,657.00	astacko3@latimes.com	Female	10.28
728.00	Adiana	7,889.00	abalchenk7@ed.gov	Female	17.40
7.00	Adora	609.00	ablabe6@ft.com	Female	26.00
958.00	Adrienne	8,805.00	adrezzerql@mtv.com	Female	35.54
669.00	Adrienne	2,544.00	aleadleyik@whitehouse.gov	Female	50.82

1 to 10 of 1000 < > Page 1 of 100 > >1

Fig. 1.129: Example column group.

Likewise, to add a calculated field you have to click on the **Add Calculated field** icon next to add column icon. Once opened the Calculated Field Wizard you have to type an alias for your calculated field in the dedicated area at the top corner of the wizard. Then you can choose from the left sidebar list the fields that you want to use in your formula. You can also use arithmetical functions or use the functions available in the menu (**aggregations, column totals, variables**). If you prefer you can create or modify the expression manually directly in the editable panel. When you are satisfied with your expression you can click on validate to check your formula syntax or save button and your calculated field appears in the field list. We provide an example in the following figure.

Calculated Field

UNIT_SALES
UNIT_SALES (SUM)

STORE_COST
STORE_COST (SUM)

Alias*

MyCalculatedField

aggregationcolumn totalsfunctions

1 SUM("UNIT_SALES")*SUM("STORE_COST")

Validate

CANCEL
SAVE

Fig. 1.130: Add a calculated field.

If Dataset is of type Solr, the columns displayed on the right panel are dataset columns fields and the calculated field formula elaboration is calculated on the fly.

If variables are set for the present cockpit, the variable menu button will appear, making it possible to add variable values in the calculated field expression.

In the bottom section of the window, you can see the table fields (with their aggregation type) listed and you also can sort columns displayed in the table by dragging them up or down, insert a column alias and customize it by adding font and style configurations using the brush shaped icon, as you can see from figure below. Here you can find configuration features to adjust the column size, max cell characters, hide column or column header options, and the row spanning

Calculated Field

annullato (annullato)

discaricato (discaricato)

emesso (emesso)

iddocumento (iddocumento)

idemissione (idemissione)

importoattivo (importoattivo)

incassato (incassato)

notificato (notificato)

residuo (residuo)

Alias *

aggregation

Validate

CANCEL

SAVE

Alias *

aggregation

column totals

variables

1

SUM("int") + SUM("float")

\$V{ myVariable }

\$V{ anotherVariable }

Fig. 1.131: Variables menu

toggler.

TABLE COLUMNS					MANAGE COLUMN GROUPS	ADD COLUMN	ADD CALCULATED FIELD	USE FUNCTION
Modal selection column		Sorting column		Sorting order	Ascending			
Name	Alias	Type	Data Type	Aggregation				
<input checked="" type="checkbox"/> QUARTER	<input checked="" type="checkbox"/> QUARTER	<input checked="" type="checkbox"/> ATTRIBUTE	string		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> STORE_ID	<input checked="" type="checkbox"/> STORE_ID	<input checked="" type="checkbox"/> ATTRIBUTE	float		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> UNITS_ORDERED	<input checked="" type="checkbox"/> UNITS_ORDERED	<input checked="" type="checkbox"/> MEASURE	float	SUM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> UNITS_SHIPPED	<input checked="" type="checkbox"/> UNITS_SHIPPED	<input checked="" type="checkbox"/> MEASURE	float	SUM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> WAREHOUSE_COST	<input checked="" type="checkbox"/> WAREHOUSE_COST	<input checked="" type="checkbox"/> MEASURE	float	SUM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> SUPPLY_TIME	<input checked="" type="checkbox"/> SUPPLY_TIME	<input checked="" type="checkbox"/> MEASURE	float	SUM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Summary row					ADD			

Fig. 1.132: Columns Settings

If you hide the column (from this view or from the column list), the column will not be visible, but will still be used for aggregation purposes. If you enable the **row span feature**, all the same visible values of the column will be collapsed in one, to avoid repetitions.

If the column is a measure, more functionalities will become available:

- **Inline chart mode:** you can choose the visualization type of the measure, and if you choose chart and maximum/minimum values, a chart will appear in the view to represent the cell measure.
- **Thresholds:** you can choose to set some thresholds that will trigger font color, background color or will show icons if the chosen condition is satisfied.
- **Format:** you can choose the prefix, suffix and the precision (i.e. 9.8 m/s). Please be aware that the data will be formatted following the locale of the user. Otherwise you can choose to treat it as string.

TABLE COLUMNS					MANAGE COLUMN GROUPS	ADD COLUMN	ADD CALCULATED FIELD
Modal selection column		Sorting column		Sorting order			
Group	Name	Alias	Type	Data Type	Aggregation		
<input checked="" type="checkbox"/> testGroup	<input checked="" type="checkbox"/> id	<input checked="" type="checkbox"/> id	<input checked="" type="checkbox"/> MEASURE	float	SUM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> testGroup	<input checked="" type="checkbox"/> first_name	<input checked="" type="checkbox"/> first_name	<input checked="" type="checkbox"/> ATTRIBUTE	string		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> int	<input checked="" type="checkbox"/> int	<input checked="" type="checkbox"/> MEASURE	float	SUM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> email	<input checked="" type="checkbox"/> email	<input checked="" type="checkbox"/> ATTRIBUTE	string		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> gender	<input checked="" type="checkbox"/> gender	<input checked="" type="checkbox"/> ATTRIBUTE	string		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> float	<input checked="" type="checkbox"/> float	<input checked="" type="checkbox"/> MEASURE	float	SUM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Summary row					ADD		

Fig. 1.133: Column settings.

Note that here you can indicate the column type and the aggregation. To add an aggregation to a column you must control the type of data that column has. An aggregation can only be added if the column value is of “number” type . The different aggregation functions are: *none* (you also can not add any aggregation function), *Sum*, *Average*, *Maximum*, *Minimum*, *Count* and *Count distinct*.

If a column group has been set another option will become available in order to set the optional group belonging of the column.

For all the columns, if at least one variable is set, the variables settings box will appear. Depending on the variable usage it will be possible to set a dynamic header or to hide the column conditionally.

Clicking on the plus button you can add one or more conditions. The possible actions are:

- **Hide column:** the column will be shown conditionally depending on the condition set.
- **Set column header name:** the column name will be replaced by the variable value.

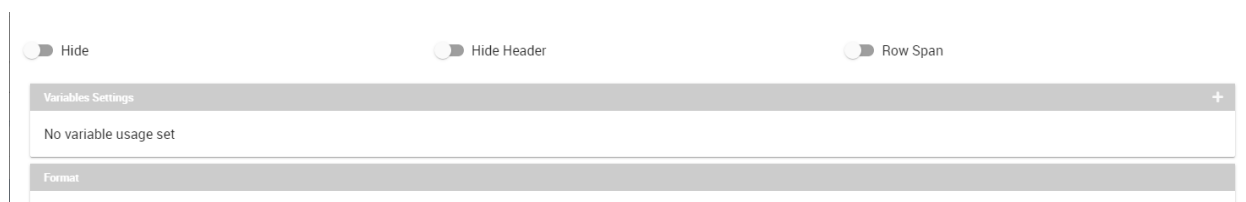


Fig. 1.134: Variable settings box.

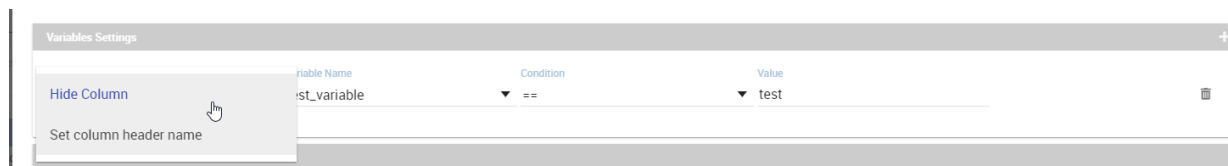


Fig. 1.135: Variable action set.

Multiple conditions can be set, but a condition fulfilling the “=” will have higher priority.

The **Style** tab is where you can customize the table by using the different options of style. It is divided into eight parts:

- In the **Header Style** section you can find different options for styling the table header. Refer to Figure below.



Fig. 1.136: Header style section of the Style tab.

- In the **Rows** section you can set the generic style for all the rows (such as alternate color background) or specify different styles according to specific conditions on values. While the multiselectable option allows you to select multiple values and pass them to other cockpit widgets or other external documents. Refer to figure below.
- In the **Summary** section you can customize the appearance of the total of columns using font and style configurations. Refer to Figure below.
- In the **Titles** section you can add the titles to the widget and modify the font size and weight. In this section you can also change the height of the widget title. Refer to Figure below.
- In the **Borders** section you can add a border to the widget and customize it by using the colors, thickness and style. Refer to the following figure.
- In the **Padding** section you can add a padding inside the widget area, specifying a value for top, bottom, left and right zones. Refer to the following figure.
- In the **Other Options** section you can add the shadows in the widget, you can set the background color of the widget and it is possible to disable or enable the screenshot option or the Excel export (when available) for that particular widget. Refer to the following figure.
- In the **Export PDF** section you can enable the export of the widget in PDF format and also specify the orientation and size. Refer to the following figure.

Once the table style settings have been implemented you can switch to the next tab. The “**Cross**” tab is where the navigation to other documents is defined. It is visible to final users but yet only configurable by a technical user (like an administrator). In this tab is possible to select between three different type of interactions: cross navigation to another document, open a preview popup of a dataset content or link to an external URL.

Rows				
<input type="checkbox"/> Multiselectable	Selection highlight Color <input type="color" value="#E4E8EC"/>			
<input type="checkbox"/> Show indexes column	Rows Height			
<input checked="" type="checkbox"/> Alternate rows	Even-rows color <input type="color" value="#E4E8EC"/>	Odd-rows color <input type="color" value=""/>		
<input type="checkbox"/> Hide "No Rows Available" message	Custom empty rows message			
<input checked="" type="checkbox"/> Row Style Conditions ADD				
<div> <div></></div> <div>Columns</div> <div>QUARTER</div> </div>	<div> <div>▼</div> <div>Condition</div> <div>==</div> </div>	<div> <div>▼</div> <div>Compare value type</div> <div>static</div> </div>	<div> <div>▼</div> <div>Compare value</div> <div>Q1</div> </div>	<div> <div>example</div> <div>✎</div> <div>🗑</div> </div>

Fig. 1.137: Rows section of the Style tab.

Table Widget Configuration			
COLUMNS	STYLE	CROSS	FILTERS
Summary			
Font-family	Font Style		
Font Size <small>px, rem or % measure units are available</small>	Font-weight	Font-color <input type="color" value=""/>	Background-color <input type="color" value=""/>

Fig. 1.138: Summary section of the Style tab.

Titles			
Title text My widget title	Horizontal alignment Center	Font Family	
Font Size 16px <small>px, rem or % measure units are available</small>	Font Style	Font Weight Bold	
Header Height <small>Height in px of the space available to title</small>	Title Color <input type="color" value="rgb(59, 103, 140)"/>	Title Background Color <input type="color" value=""/>	

Fig. 1.139: Titles section of the Style tab.

BORDERS			
Borders Style	Borders Thickness	Borders Color <input type="color" value=""/>	
Border radius top left	Border radius top right	Border radius bottom left	Border radius bottom right

Fig. 1.140: Borders section of the Style tab.

Padding			
<div> <div>🔗</div> <div>Padding Left</div> <div>10px</div> <div><small>use px measure unit, ie: 5px</small></div> </div>	<div> <div>Padding Top</div> <div>10px</div> <div><small>use px measure unit, ie: 5px</small></div> </div>	<div> <div>Padding Right</div> <div></div> <div><small>use px measure unit, ie: 5px</small></div> </div>	<div> <div>Padding Bottom</div> <div></div> <div><small>use px measure unit, ie: 5px</small></div> </div>

Fig. 1.141: Padding section of the Style tab.

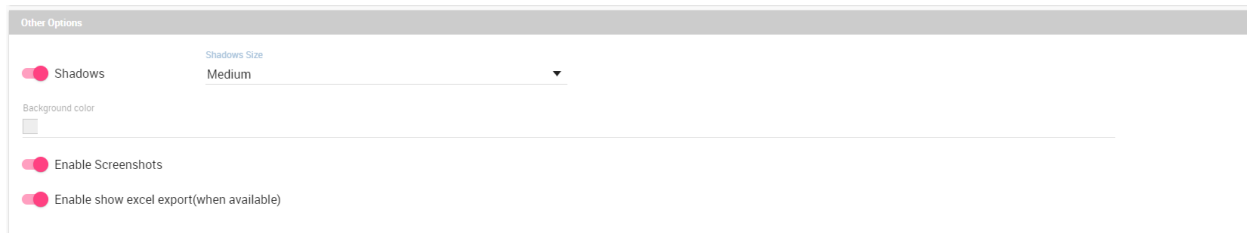


Fig. 1.142: Other Options section of the Style tab.



Fig. 1.143: Export PDF section of the Style tab.

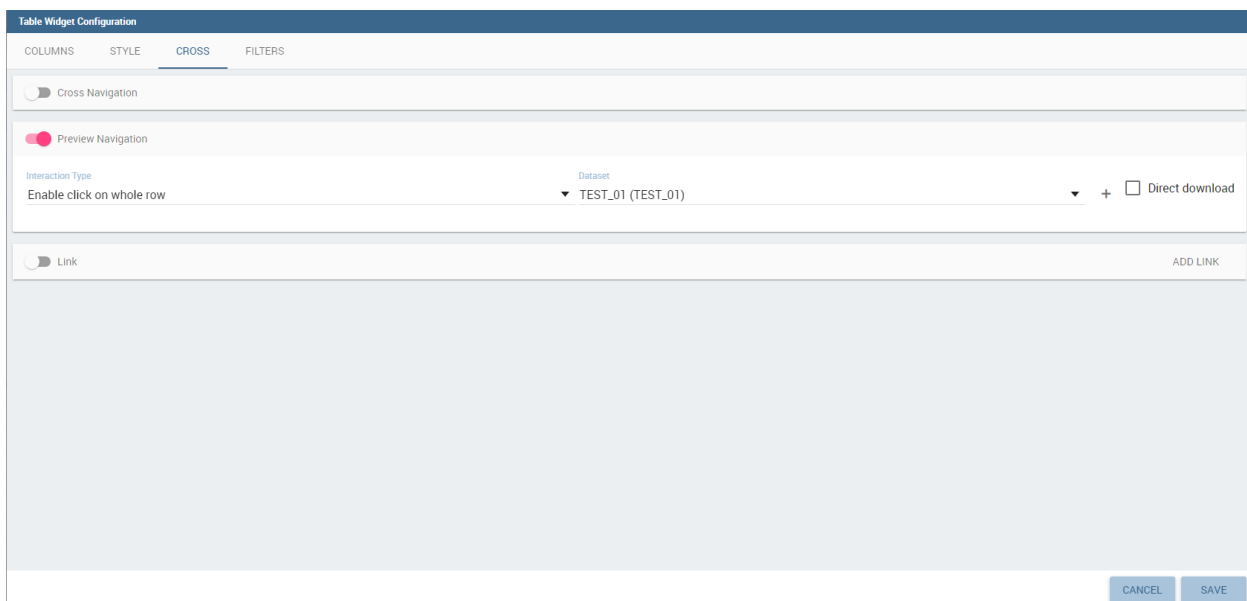


Fig. 1.144: Cross tab with preview of a dataset enabled.

Finally, the **“Filters”** tab is where you can filter the table results by adding a limit to the rows or a conditions in the columns. the following figure shows an example of how to set the limit rows or a conditions on dataset columns.

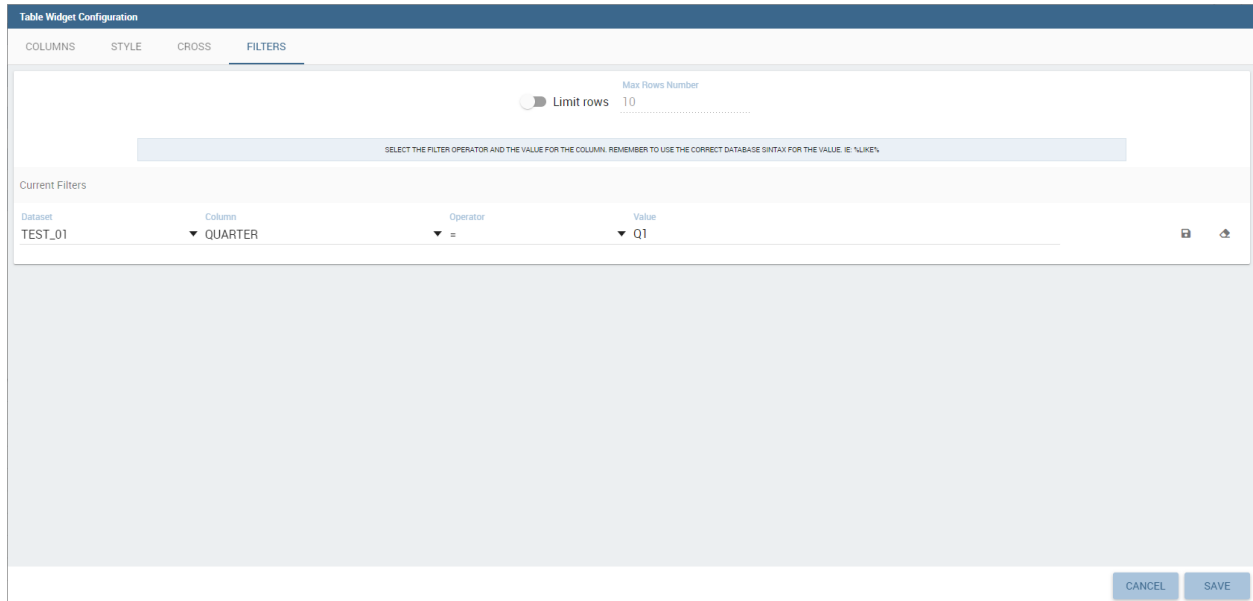


Fig. 1.145: Filters tab of the table widget configuration.

Once you have finished setting the different configuration options of the table widget, then just click on **“Save”** and your new widget is displayed inside the cockpit.

1.4.1.5 Cross Table widget

Similar configurations are available also for the Cross Table widget. In this data visualization option, you still have the tabs: **Dataset** tab, **Configuration** tab, the **Style** tab and the **Filters** tab as you can see below.

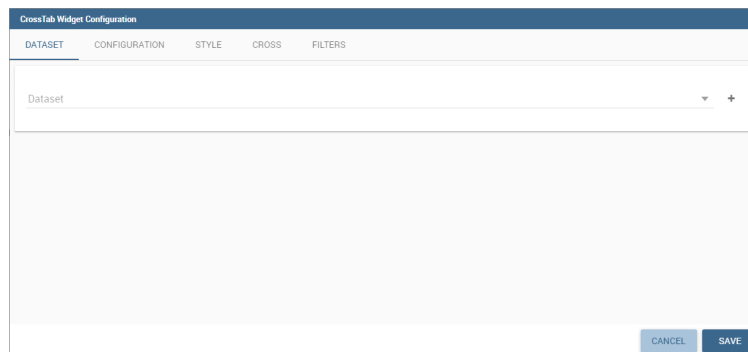


Fig. 1.146: Dataset section of the crosstab widget configuration.

Using the “Dataset” tab the user can add the dataset to take values from. Consequently, it is necessary to select the fields you wish to appear as columns, those as row and measures to be exhibited in the pivot table. See figure below. Remember to set column and row fields as attributes, while measure fields as numbers.

There is also the possibility to add calculated fields as measures clicking on the “Add calculated field button”. The calculated field will work exactly as in the table widget. After creating a calculated field there will be the possibility to edit, remove it or set the properties for it’s column clicking on the new measure buttons.

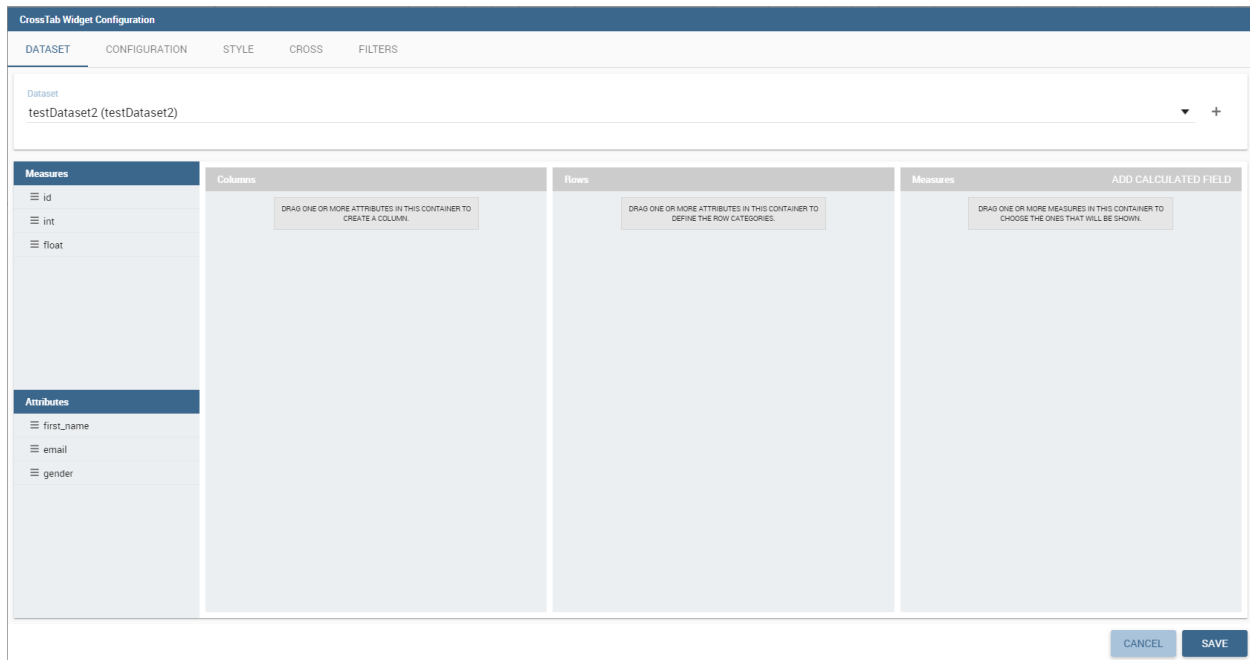


Fig. 1.147: Selecting columns, rows and measures of the crosstab.

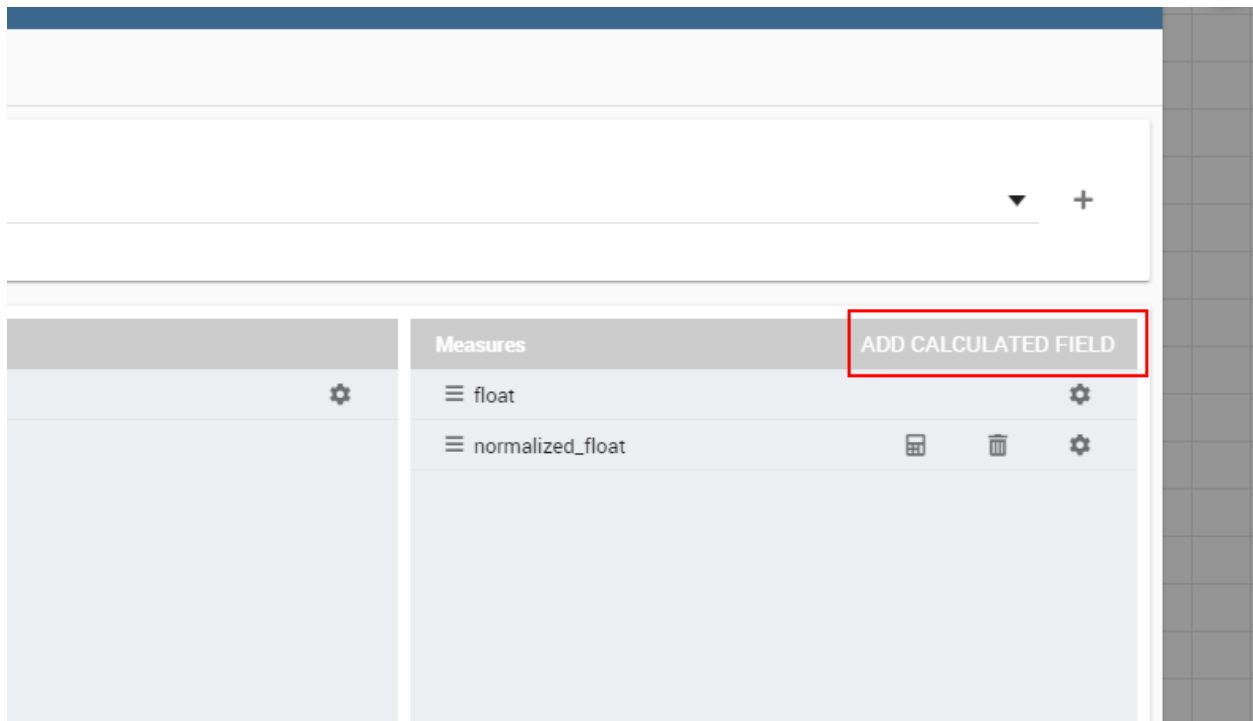


Fig. 1.148: Calculated field detail.

Once the columns, rows and measures have been selected the style of each column can be set by clicking on the cog settings icon. A popup will open with different options for the selected column. See figure below.

Fig. 1.149: Column style popup.

It is possible to sort the crosstab according to the values of the selected column or, alternatively, according to columns not visible in the crosstab. It can also be set the style of the column, such as the font size, the font weight or the cell alignment. There is also the possibility to specify the size of the column in pixels (you can also use percent values but it is better to use pixels).

In case the selected column is of type measure, there is a dialog to configure the behaviour of that field:

Fig. 1.150: Measure configuration dialog.

A particular option for a measure is **Exclude from Total and SubTotal**: that checkbox excludes the measure from the sums of Total and SubTotal fields making the relative table cells empty.

As figure above shows, you can also manage threshold. For measures only, it is possible to associate a particular icon or a specific background color to a particular measure's value or range. To do so add a new threshold, set a condition for it to appear, and choose the icon from the list or select the color that will be changed to the cell. It is possible to add more or to remove thresholds using the add or delete button.

If one or more variables are set, in column and measure settings another field will appear. It is possible to set the header name to be dynamic using one of the variables set. If one of the variables are selected in the combo as in example, the

Condition	Threshold	Text/icon color	Background-color/chart color
>	20	rgb(255, 158, 14)	Select a color
>	40	rgb(255, 0, 0)	rgb(247, 255, 39)

Fig. 1.151: Measure threshold setting.

Column Name	Title	Variables (optional)	Aggregation function
int	int	testVariable	sum

Fig. 1.152: Variables header value.

header name will be changed depending on the current variable value.

Style
for measures only
Prefix
Suffix
Format
Precision
Column Size
Font-color
Background-color
Font-weight
Font-size
Cell horizontal alignment
Cell vertical alignment

Fig. 1.153: Column style.

It's also possible to set style elements for both attributes and measures. In measures will also be possible to set the precision and prefix/suffix of the cell value. The comma and dot used for decimals and thousands will be automatically changed depending on user's locale.

Once the dataset has been properly configured, you can proceed to the "Configuration" tab.

The latter is made up of three sections: **General**, **On rows** and **On columns**, as Figure below shows.

In the "**General**" section you can set the following features:

- define the maximum cell number to show;
- decide to hook measures to columns or rows;
- decide to show percentages of measures related to columns or rows.

Thanks to the "**On rows**" feature, you can easily compute totals or subtotals on rows. Figure below exhibit an example.

Otherwise, thanks to the "**On columns**" feature, you can easily compute totals or subtotals on columns. Figure below exhibit an example.

Selecting the "**Hide rows on just null values**" will hide all the rows with just 0 or null values, avoiding space waste if unneeded.

CrossTab Widget Configuration

DATASET CONFIGURATION STYLE CROSS FILTERS

General

Max Cells Number

Measures on ☒ Columns ☐ Rows

Percentage calculated on ☐ Columns ☐ Rows ☒ no

On rows

☐ show totals Labels for Totals Total ☐ show sub-totals Labels for Subtotals SubTotal

☐ Hide rows on just null values ☐ Fix Attribute's columns ☐ Expand/collapse rows

On columns

☐ show totals Labels for Totals Total ☐ show sub-totals Labels for Subtotals SubTotal

CANCEL SAVE

Fig. 1.154: Configuration tab interface.

↑ gender															
F							M						Total		
↑ occupation							↑ occupation						↑ Total		
	Clerical	Management	Manual	Professional	Skilled	Manual	SubTotal	Clerical	Management	Manual	Professional	Skilled	Manual	SubTotal	Total
↑ product family	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales
Drink	1,328.00	11,848.00	17,696.00	26,187.00	21,191.00		78,250.00	1,270.00	10,247.00	19,566.00	23,637.00	20,418.00		75,138.00	153,388.00
Food	9,817.00	97,784.00	150,429.00	216,781.00	171,188.00		645,999.00	11,982.00	86,671.00	160,463.00	198,064.00	173,250.00		630,430.00	1,276,429.00
Non-Consumable	2,872.00	25,584.00	38,205.00	57,796.00	45,271.00		169,728.00	3,378.00	23,505.00	43,803.00	52,820.00	45,636.00		169,142.00	338,870.00

Fig. 1.155: Computing totals and/or subtotals on rows.

↑ month_of_year

		↑ the_month	↑ the_month	↑ the_month	↑ the_month	↑ the_month	↑ the_month	↑ the_month	↑ the_month	↑ the_month	↑ the_month	↑ the_month	↑ the_month
		January	February	March	April	May	June	July	August	September	October	November	December
↑ gender	↑ product family	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales
F	Drink	6,008.00	5,931.00	6,586.00	5,950.00	6,194.00	6,468.00	6,890.00	5,926.00	6,907.00	5,941.00	7,440.00	8,009.00
	Food	53,303.00	50,148.00	54,783.00	51,018.00	51,447.00	50,352.00	55,768.00	51,517.00	54,231.00	48,437.00	59,720.00	65,275.00
	Non-Consumable	13,449.00	13,572.00	14,453.00	13,253.00	13,490.00	13,063.00	14,488.00	13,632.00	14,032.00	13,512.00	15,768.00	17,016.00
	SubTotal	72,760.00	69,651.00	75,822.00	70,221.00	71,131.00	69,883.00	77,146.00	71,075.00	75,170.00	67,890.00	82,928.00	90,300.00
M	Drink	6,400.00	6,069.00	6,195.00	5,795.00	5,868.00	6,136.00	5,972.00	5,991.00	6,260.00	5,843.00	7,146.00	7,463.00
	Food	50,846.00	50,357.00	52,360.00	49,271.00	49,477.00	52,678.00	51,837.00	50,604.00	50,471.00	47,932.00	61,425.00	63,172.00
	Non-Consumable	13,953.00	12,838.00	14,216.00	12,749.00	13,546.00	13,980.00	14,576.00	13,570.00	13,639.00	13,408.00	16,089.00	16,578.00
	SubTotal	71,199.00	69,264.00	72,771.00	67,815.00	68,891.00	72,794.00	72,385.00	70,165.00	70,370.00	67,183.00	84,660.00	87,213.00
Total		143,959.00	138,915.00	148,593.00	138,036.00	140,022.00	142,677.00	149,531.00	141,240.00	145,540.00	135,073.00	167,588.00	177,513.00

Fig. 1.156: Computing totals and/or subtotals on columns.

The “Fix attribute’s columns” will pin to the left the columns containing the attributes, so side scrolling will be available without losing those columns.

	↑ QUARTER							
	Q1			Q2			Q3	
↑ PRODUCT_FAMILY	SALES	STORE_COST_NULL	STORE_COST	UNIT_SALES	STORE_COST_NULL	STORE_COST	UNIT_SALES	STORE_COST_NULL
Drink_à	12	12,712	10,201.6121	12,273		9,778.6164	12,917	
Food	46	98,346	84,045.6925	97,802		83,433.6075	99,694	
Non'Consumablè	20	26,020	22,265.3859	25,670		21,868.1079	26,801	

Fig. 1.157: Pinned columns example.

The “**expand/collapse**” functionality will add a + and - button in your rows, in order to easily aggregate your data. In the widget menu you will also find the expand all/collapse all buttons, in order to reset closing or opening your whole widget.

Be aware that to use this functionality columns subtotal should be selected. If not the functionality check will enable it automatically.

↑ gender	↑ email	int
Female ⊕	SubTotal	2,450,264
	aalman6@howstuffworks.com	7,649
	abaish9t@myspace.com	3,272
	abarnissp@skyrock.com	3,339
	abesantes@examiner.com	2,414
	ablewettp3@dion.ne.jp	2,219
	ablewittlp@about.com	1,755
	abohjelm@newsvine.com	4,500
	abramhallko@mediafire.com	1,247
	abuncombeap@dropbox.com	1,107
	acestardqt@e-recht24.de	1,173

Fig. 1.158: Expand/collapse example.

Switching to the “**Style**” tab you can find the general style settings available for the crosstab.

- **Crosstab General Options** where the rows’ height, the general font and font size can be set; in particular, the layout combo determines how the columns resize themselves in respect of the contained value;



Fig. 1.159: General style options for crosstab.

- **Crosstab Headers Font Options** where you can configure the header style settings as color, background, font, etc.

Fig. 1.160: Crosstab Headers Font Options for crosstab.

- **Measures Font Options** where you can configure several style options for measures, such as color, background, font size, etc.

Fig. 1.161: Measures Font Options for crosstab.

- Using the **Grid** section you can mark (or not) grid borders, decide for border style, thickness and color. You can also alternate row indicating different colors.

Fig. 1.162: Grid Options for crosstab.

- In the **Measures Headers** section you can configure different style option for measure headers, such as color, background, font size, etc.
- In the **Total** section you can set color and background of totals (if any).
- In the **Subtotal** section you can set color and background of subtotals (if any).
- In the **Titles** section you can add titles to widget and customize them using different styles.
- In the **Borders** section you can add borders to widgets and customize them using different styles.
- In the **Other Options** section you can add a shadow to widget layout and indicate its measure, color the widget background at convenience and it is possible to disable or enable the screenshot option or the Excel export for that particular widget.

Opening the **Cross** tab is possible to set a cross navigation to another document or an external link. In addition to other cross-navigations, for cross table widget it is possible to set as a dynamic value the name of the selected measure column or the selected category. The choice is available from the combobox.

Once some or all (at least the mandatory) of the above mentioned setting features have been set you can save and the widget will be inserted into the cockpit area.

MEASURES HEADERS		
Font Size <small>px, rem or % measure units are available</small>	Font Family	Font Weight
Text Decoration	Color Select a color	Background Select a color

Fig. 1.163: Measures Headers Option for crosstab.

TOTALS	
Color Select a color	Background Select a color

Fig. 1.164: Color settings for Totals.

SUBTOTALS	
Color Select a color	Background Select a color

Fig. 1.165: Color settings for Subtotals.

TITLES		
Title text	Horizontal alignment	Font Family
Font Size <small>px, rem or % measure units are available</small>	Font Style	Font Weight
Header Height <small>Height in px of the space available to title</small>	Title Color Select a color	Title Background Color Select a color

Fig. 1.166: Title settings.

BORDERS			
Borders Style	Borders Thickness	Borders Color	
Border radius top left <small>use px measure unit, ie: 5px</small>	Border radius top right <small>use px measure unit, ie: 5px</small>	Border radius bottom left <small>use px measure unit, ie: 5px</small>	Border radius bottom right <small>use px measure unit, ie: 5px</small>

Fig. 1.167: Border settings.

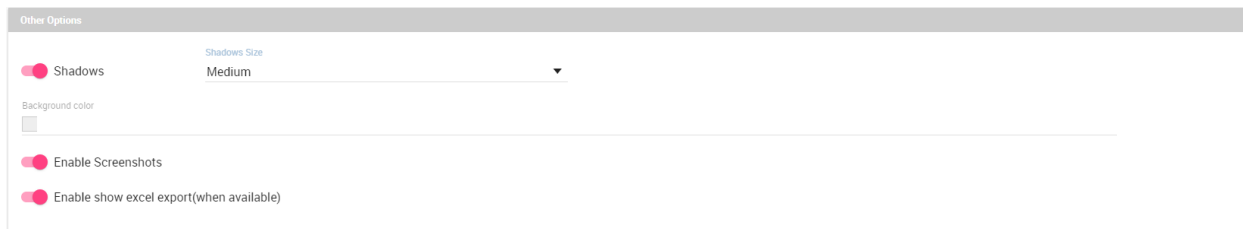


Fig. 1.168: Other Options for crosstab.

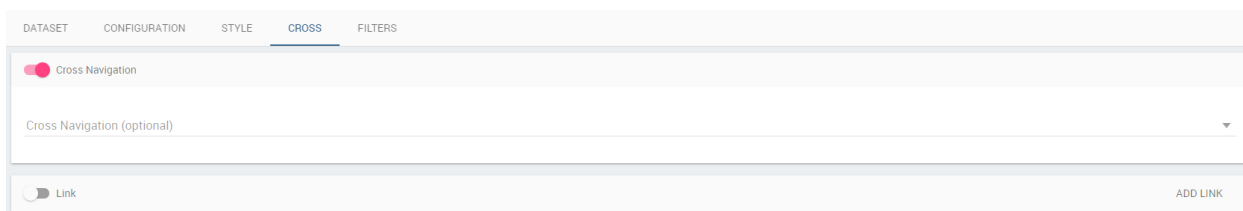


Fig. 1.169: Cross navigation for cross table widget.

1.4.1.6 Document section

The Document widget allows to add an external document into the cockpit area. This widget supports documents like reports, maps, etc.

Use the Data configuration button to add a document source to the cockpit. Click on the “Plus” icon on the right half of the page to choose among all available documents.

The Document Widget configuration is divided into two parts: **Custom** tab and **Style** tab as you can see from Figure below.

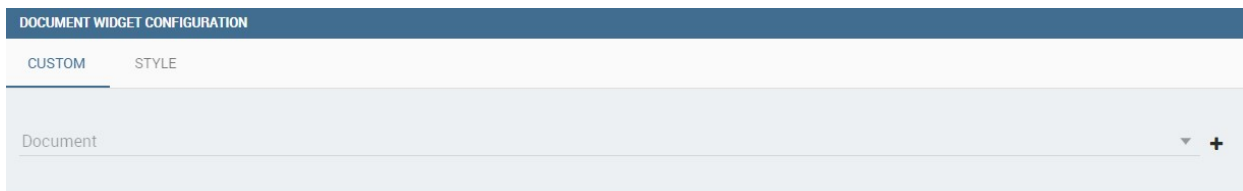


Fig. 1.170: Custom tab of the Document widget.

The Custom tab is the place where the document is uploaded while the Style tab is where all style options are set.

1.4.1.7 Active Selections widget

To enable the Active Selections widget, which means the possibility to have all the currently applied selections listed and accessible on a widget, the user must open the “**Active Selections**” feature through the “**Add widget**” functionality and configure the demanded options. Figure below shows the “**Active Selections widget configuration**” interface.

The Active Selections will display the elements selected by the user. Figure below shows an example.

If global associations have been set, clicking on table, cross table or chart elements will update all corresponding widgets. Otherwise, only the widget on which selection has been made will be updated. In both cases the Selection widget will display the highlighted attribute values.

The dashboard displays sales data for 'Food' products across various quarters. The main bar chart shows sales for 19 quarters, with values ranging from 367.54 to 6,844.00. A table below provides detailed data for the first 10 quarters. A pie chart on the left shows the distribution of sales across quarters, with Q1 being the largest. A zoomed-in bar chart on the right highlights the sales for 'Food' products, showing values of 98,346.00 and 84,045.69.

QUARTER	THE_DATE	STORE_ID	PRODUCT_FAM	UNIT_SALES	STORE_COST
Q1	January 12, 1998 12:00	1.00	Food	415.00	373.43
Q1	January 21, 1998 12:00	1.00	Food	292.00	270.03
Q1	March 25, 1998 12:00	1.00	Food	321.00	245.11
Q1	January 21, 1998 12:00	2.00	Food	26.00	24.96
Q1	January 9, 1998 12:00	3.00	Food	280.00	259.83
Q1	March 25, 1998 12:00	2.00	Food	39.00	32.60

1.4. Create a New Dashboard

1.4.1.8 Selector Widget

The **Selector Widget** is a way to include a dataset filter, directly inside the cockpit, that can be displayed like a combobox, radio button or checkboxes.

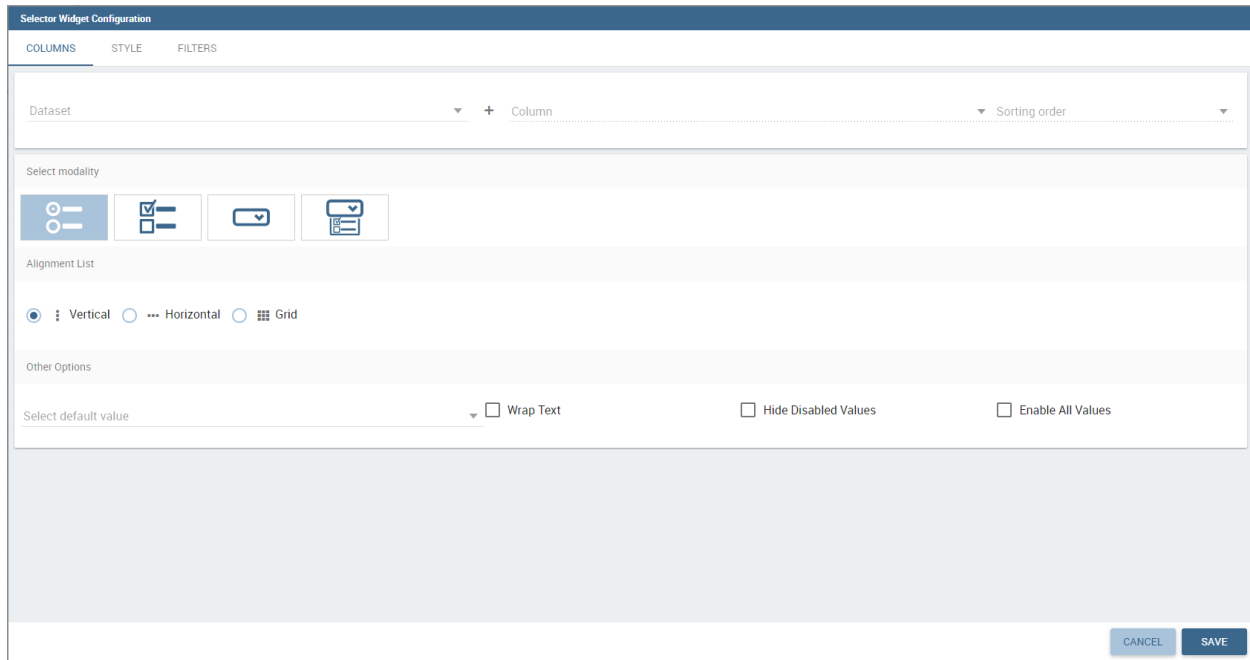


Fig. 1.173: Selector widget outlook.

In detail, use the **Columns** tab to select the dataset and the dataset column on which you want to apply the filter. Then select the **Select modality** options; for instance, choose between single or multivalue or to use a list or a combobox. Note that for the list option you can further choose among “vertical”, “horizontal” or “grid”. You can also decide to add a default value, chosen from main column’s first item, main column’s last item or to simply assign a static value. Finally, by clicking on the Wrap Text option it is possible to wrap the text shown in the selector; this option is useful when the categories to choose from are string of long dimensions.

In the case of the selector of type list “grid” it is also possible to set the grid columns width.

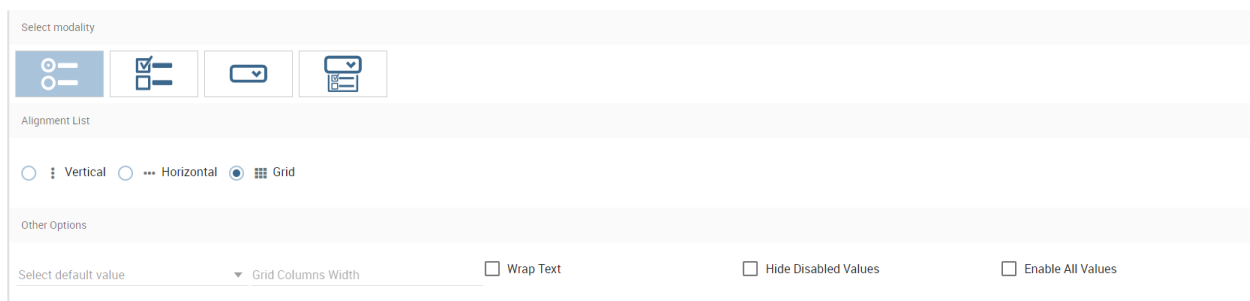


Fig. 1.174: Grid columns width.

Move to the **Style** tab to set the widget style in terms of: label, titles, borders, shadows and background color. Figure below shows a customization example.

Finally use the **Filters** tab to handle pagination or filter on a dataset column.

The Selector widget works effectively as a ready-to-use filter panel.

SELECTOR WIDGET CONFIGURATION

COLUMNS **STYLE** FILTERS

LABEL

Font-family: Roboto Font Size: 12px (px, rem or % measure units are available) Font-weight: Font Style: Label Height: 50% Label Font Color: rgb(89, 89, 89) Background-color:

TITLES

Title text: Horizontal alignment: Font Family: Font Size: (px, rem or % measure units are available) Font Style: Font Weight: Header Height: (Height in px of the space available to title) Title Color: Select a color Title Background Color: Select a color

BORDERS

Borders Style: Borders Thickness: Borders Color: Border radius top left: (use px measure unit, ie: 5px) Border radius top right: (use px measure unit, ie: 5px) Border radius bottom left: (use px measure unit, ie: 5px) Border radius bottom right: (use px measure unit, ie: 5px)

OTHER OPTIONS

Fig. 1.175: Selector widget configuration.

Selector Widget Configuration

COLUMNS STYLE **FILTERS**

☐ Limit rows Max Rows Number: 10

SELECT THE FILTER OPERATOR AND THE VALUE FOR THE COLUMN. REMEMBER TO USE THE CORRECT DATABASE SYNTAX FOR THE VALUE. IE: %LIKE%

Current Filters:

Fig. 1.176: Selector filters.

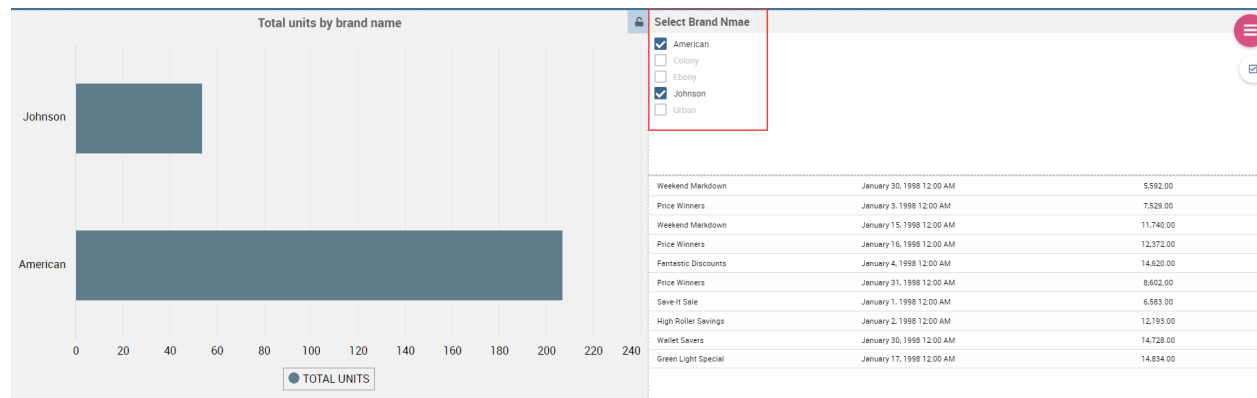


Fig. 1.177: Selector widget execution example.

1.4.1.9 HTML Widget

The HTML widget allows to add customized HTML and CSS code to implement very flexible and customized dynamic elements to the cockpit. This widget supports all HTML5 standard tags and CSS3 properties.



Warning: For security reasons no custom Javascript code can be added to html tags. Every tag considered dangerous will be deleted after saving the document.

The **Edit** section of the widget is composed by five tabs: the Dataset, HTML editor, Style, Cross and Filters. In the editor tab is possible to add the code that will be shown in the widget. Clicking on the top expander section in the tab, the one named “CSS” also the CSS editor will be available.

Important: A CSS property will be extended to all the classes in the cockpit with the same name, to apply the property only to the current widget use the id prefix shown in the info panel of the CSS editor

In the right side of the editor is possible to explore the available tags that can be copied inside the code, those tags will be explained in details in the following paragraphs. Given that is not possible to add custom JavaScript code inside the html editor, this available tags are the tools to make the widget dynamic and to use the dataset data.

The **Dataset** tab allows the user to select a dataset to make the Widget dynamic and to bind it to dataset data. After choosing a dataset the list of available columns will be shown. Those names will be useful inside the dynamic tags. Here it is also possible to order the dataset according to a column and to select the ordering type (ascending or descending).

By clicking on the icon  of a specific column the dataset will be ordered by that column by default by ascending order. In order to select the descending ordering type you have to click another time on the icon (the icon will be now like this ).

Available Tags

kn-column

```
[kn-column='COLUMN-NAME' row='COLUMN-ROW-NUMBER' aggregation='COLUMN-AGGREGATION'
precision='COLUMN-DECIMALS']
```

The **kn-column** tag is the main dynamic HTML Widget tool, it allows to select a column name from the selected dataset and to display its values. The value of the kn-column attribute should be the name of the column value you

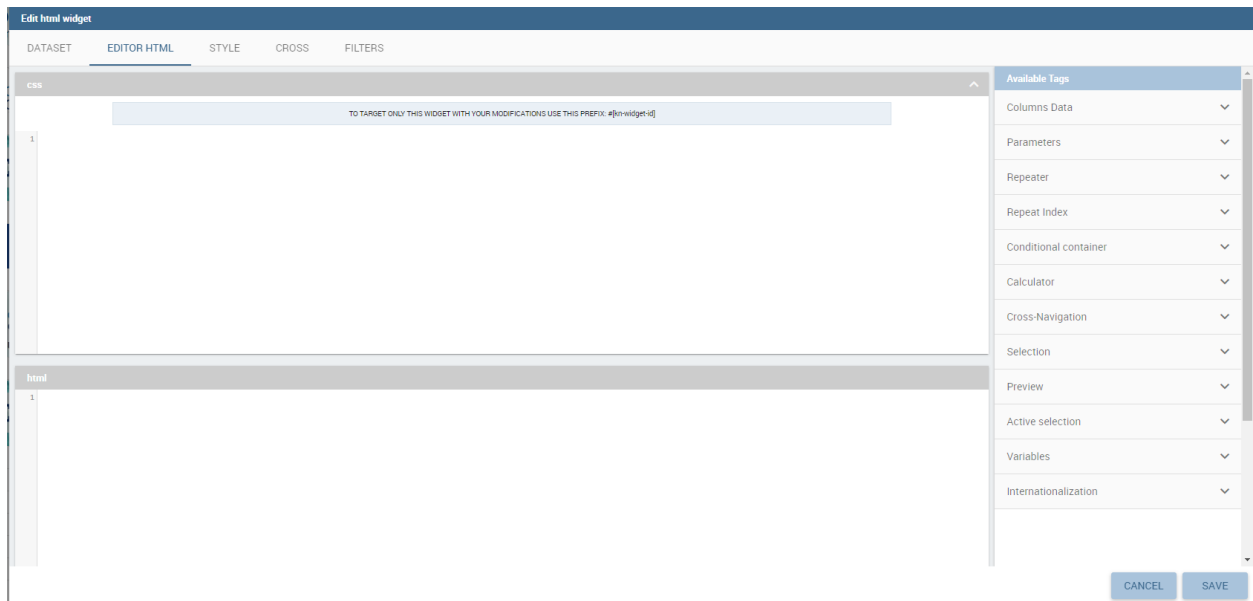


Fig. 1.178: HTML widget editor

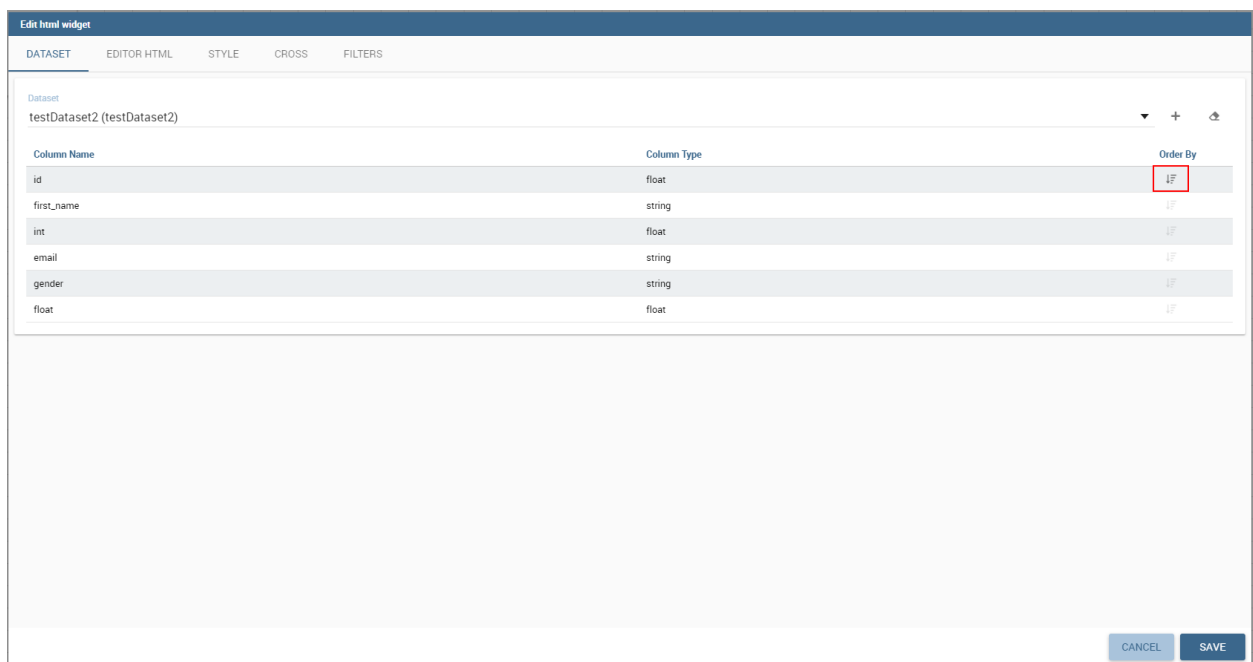


Fig. 1.179: Dataset selection

want to read in execution.

The **row** attribute is optional and is a number type attribute. This attribute can let you retrieve a specific row according to the position in the dataset. If no row is selected the first row column value will be shown.

The **aggregation** attribute is optional and is a string type attribute. If inserted the value shown will be the aggregation of all column rows values. The available aggregations are: AVG, MIN, MAX, SUM, COUNT_DISTINCT, COUNT, DISTINCT COUNT.

The **precision** attribute is optional and is a number type attribute. If added and if the result value is a number, the decimal precision will be forced to the selected one.

kn-parameter

```
[kn-parameter='PARAMETER-NAME']
```

The **kn-parameter** tag is the tool to show a dataset parameter inside the widget execution. The value of the **kn-parameter** attribute should be the name of the parameter to display.

kn-calc

```
[kn-calc=(CODE-TO-EVALUATE) precision='VALUE-PRECISION']
```

The **kn-calc** tag is the tool to calculate expressions between different values on widget execution. Everything inside the brackets will be evaluated after the other tags substitution, so will be possible to use other tags inside.

The **precision** attribute is optional and is a number type attribute. If added and if the result value is a number, the decimal precision will be forced to the selected one.

kn-repeat

```
<div kn-repeat="true" limit="LIMIT-NUMBER"> ... REPEATED-CONTENT ... </div>
```

The **kn-repeat** attribute is available to every HTML5 tag, and is a tool to repeat the element for every row of the selected dataset.

This attribute is naturally linked to **kn-column** tag. If inside a **kn-column** tag without a **row** attribute is present, the **kn-repeat** will show the column value for every row of the dataset.

Inside a **kn-repeat** is possible to use the specific tag **[kn-repeat-index]**, that will print the index of the repeated column row.

The **limit** attribute is optional and is a number type attribute. If added the number of row repeated will be limited to the selected number. If no limit is provided just the first row will be returned. If you want to get all records, you can set it to -1, but be careful because big datasets can take a while to load completely.

kn-if

```
<div kn-if="CODE-TO-EVALUATE"> ... </div>
```

The **kn-if** attribute is available to every HTML5 tag and is a way to conditionally show or hide an element based on some other value. The attribute content will be evaluated after the other tags substitution, so it will be possible to use other tags inside. If the evaluation returns true the tag will be shown, otherwise it will be deleted from the execution.

kn-cross

```
<div kn-cross> ... </div>
```

The **kn-cross** attribute is available to every HTML5 tag and is a way to make the element interactive on click. This attribute makes the element clickable to open the cross navigation specified in the widget settings. If there is no cross navigation set this tag will not work.

kn-preview

```
<div kn-preview="DATASET-TO-SHOW"> ... </div>
```

The **kn-preview** attribute is available to every HTML5 tag and is a way to make the element interactive on click. This attribute makes the element clickable to open the dataset preview dialog. The attribute value will be the *dataset label* of the dataset that you want to open. If a dataset is not specified the cockpit will use the one set for the widget. If no dataset has been set and the attribute has no value this tag will not work.

kn-selection

```
<div kn-selection-column="COLUMN-NAME" kn-selection-value="COLUMN-VALUE"> ... </div>
```

The **kn-selection-column** attribute is available to every HTML5 tag and is a way to make the element interactive on click. This attribute makes the element clickable to set the chosen column and value as a selection filter in the cockpit. The default will use as a selection the first row value of the column.

The **kn-selection-value** attribute is optional and will let you specify a specific value as a column selection filter.

kn-variable

```
[kn-variable='VARIABLE-NAME' key='VARIABLE-KEY']
```

The **kn-variable** tag is the tool to read the runtime value of one of the defined variables. It will change depending on the current value and can be used inside **kn-if** and **kn-calc**.

The **key** attribute is optional and will select a specific key from the variable object if the variable is “Dataset” type, returning a specific value instead of a complete dataset.

Warning: Banned Tags In order to avoid Cross-site scripting and other vulnerabilities, some tags are *not allowed* and will automatically be removed by the system when saving the cockpit:

- `<button></button>`
- `<object></object>`
- `<script></script>`

If you need to simulate a button behaviour use a `div` (or another allowed tag) and replicate the css style like in the following example:

```
1 <div class="customButton">Buttonlike div</div>
```

```
1 .customButton {
2   border: 1px solid #ccc;
3   background-color: #ededed;
4   cursor: pointer;
5 }
6 .customButton:hover {
7   background-color: #d8d8d8;
8 }
```

Warning: Whitelist

Base paths to external resources (images, videos, anchors, CSS files and inline frames) must be declared within `TOMCAT_HOME/resources/services-whitelist.xml` XML file inside Knowage Server, otherwise those external links will be removed by the system. This whitelist file contains safe and trusted websites, to restrict end users of providing unsafe links or unwanted web material. Knowage Server administrator can create or edit it (directly on the file system) to add trusted web sites. Here below you can see an example of `services-whitelist.xml` file; as you can see, its structure is quite easy: `baseurl` attributes refer to external services, `relativepath` must be used for Knowage Server internal resources instead:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <WHITELIST>
3   <service baseurl="https://www.youtube.com" />
4   <service baseurl="https://player.vimeo.com" />
5   <service baseurl="https://vimeo.com" />
6   <service baseurl="https://media.giphy.com" />
7   <service baseurl="https://giphy.com" />
8   <service baseurl="https://flic.kr" />
9   <service relativepath="/knowage/themes/" />
10  <service relativepath="/knowage/icons/" />
11  <service relativepath="/knowage/restful-services/1.0/images/" />
12 </WHITELIST>

```

Like other widgets the **“Style”** tab and the **“Filters”** tab are available in order to set the general style options for the widget and to filter the results displayed in the HTML widget.

1.4.1.10 Map Widget

The Map Widget is useful when a user needs to visualize data related to a geographic position. The widget supports multiple layers, one for every dataset added to widget configuration, and one data field for every layer: the user can switch on-the-fly between all data available on the layer.

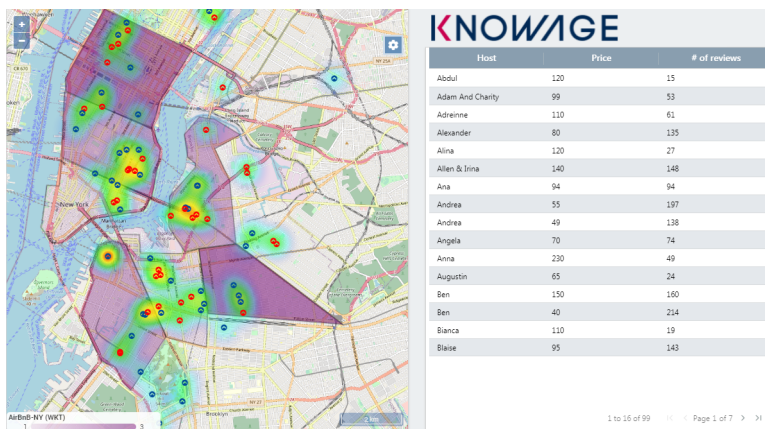



Fig. 1.180: Map widget.


In Map Widget configuration a user can add and remove layers, set the format of the spatial attribute to use and specify the attributes to display on map and on the detail popup:

Column	Alias	Type	Aggregation function	Aggregate by	Show on detail	Show on map	Show on filter	Show on tooltip	Modal column
host_name	host_name	ATTRIBUTE			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
coordinate		SPATIAL_ATTRIBUTE			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
price	price	MEASURE	SUM		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Fig. 1.181: Map widget configuration.

Every dataset with a spatial attribute is eligible to become a layer in map widget. Only one layer of the widget can be susceptible to user selection: that layer will be the only one with **Target** slide set to on. For each layer a user can also specify its default visibility with **Default visible** slide. Enabling **Static** switch on a layer make it visible and non

clickable, useful when a user wants a fixed background layer with dynamic data from a dataset. With buttons  and

 the user can set the metadata and the layer style respectively.

In layer's metadata, the user can add calculated fields (more on that later) and set the spatial attribute of the dataset that will be used to display a markers on the map. Actually, many spatial attribute types are supported:

- String format: where the value specify two decimal numbers representing latitude and longitude separated by a space;
- JSON: where the value is a text string in [GeoJSON](#) format;
- WKT: where the value is a text string in [Well-known Text](#) format;

Important: Geographic coordinates format

For every format above user have to specify what is the format of geographic coordinate: user have to specify if latitude comes first or vice versa.

The first field listed in metadata is the spatial attribute and Knowage let the user to set if the spatial attribute need to be part of the aggregation or not: this let the user to create special query; for example, you may need to just list all the records of a dataset without any aggregations and in this case you can simply uncheck all the aggregate by checks and clean up the aggregation function for the spatial attribute; another example is where the spatial attribute at database side is of a special type like CLOB on Oracle, in that case the user cannot use that field for the aggregation but the user can exclude the spatial attribute from the aggregation, converting the field to measure and setting an aggregation function.

Every field of the dataset, except for the spatial one, can have a custom alias to show on map widget: just double click the label to edit it. A user can also specify if a field have to be shown on detail popup.

For measures a user could specify the aggregation function, if it has to be shown on detail popup and if it has to be shown on map: at least one field has to be shown on map.

For attributes a user could specify if it has to be shown on detail popup or if it has to be show as a filter: in that case, the attribute will be available in the control panel with its all distinct values to let the user to have an immediate evidence of which markers have the selected value for the measure

The user could also select if a specific attribute should be displayed in the tooltip that will be shown when the user hovers a specific feature on the map.

The 3-dots-menu on the right of each column of the dataset contains additional functionalities: for measures, for example, there is the possibility to specify thresholds.

The threshold menu open a dialog where the user can customize marker color by value range: that's very useful when a user wants to immediately identify a marker by it's value.

For all the attributes that are filters, a user could select the relative value from the control panel:

As said, Map widget supports calculated fields, a way for a user to calculate additional data to show on map or to display into popup detail:

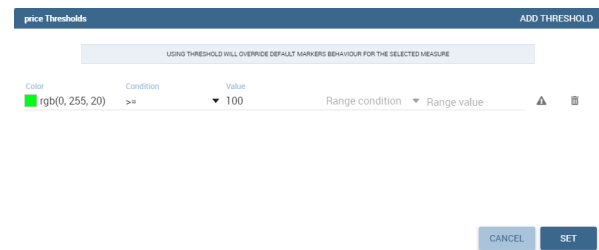


Fig. 1.182: Threshold dialog.

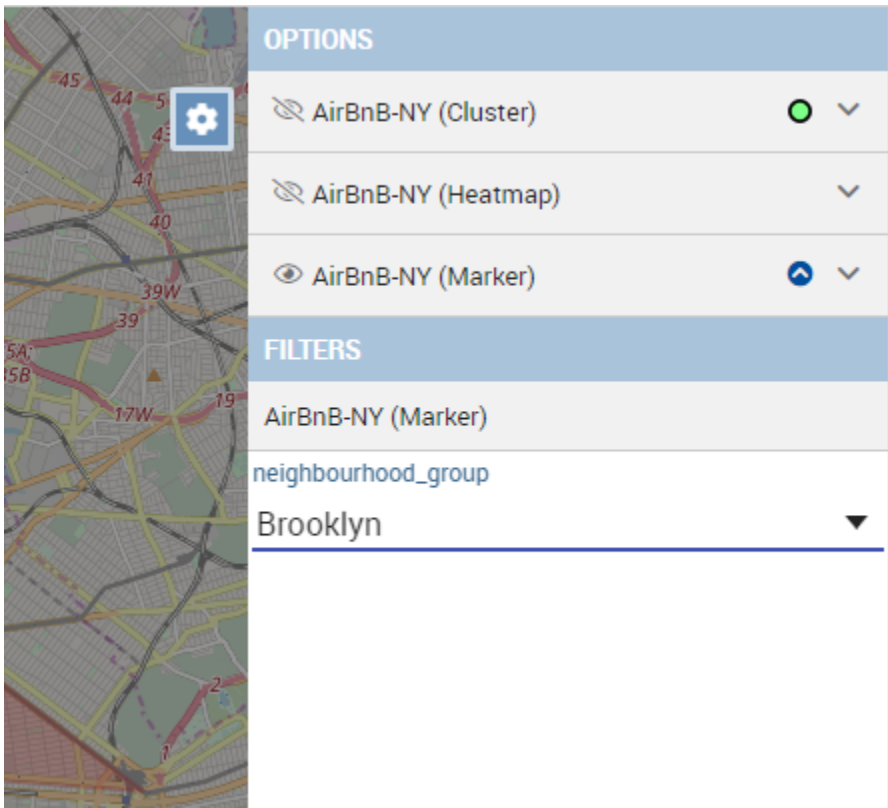


Fig. 1.183: Filter selection.

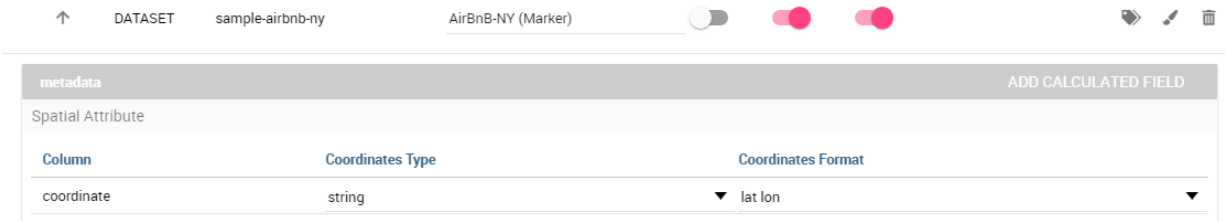


Fig. 1.184: Add calculated field button in layer’s metadata.

From the calculated field's dialog a user can combine measures and operations to add more data to the layer. The user can use a SQL-like syntax to create a statement that describe the new calculated field:



Fig. 1.185: Calculated Field's dialog.

The newly calculated field added by the user is shown as a measure in layer's dataset: from the 3-dots menu on the right of the field a user can update or delete the calculated field.

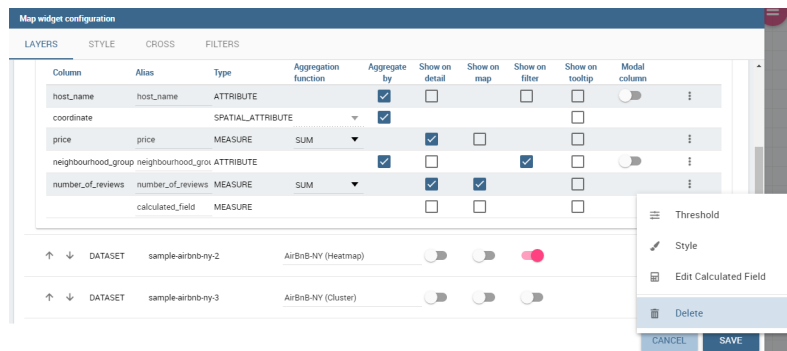


Fig. 1.186: The 3-dots menu on calculated field.

For every layer, a user can specify the way the data will be displayed on map: the user can choose between a markers, cluster, heatmaps and choropleth.

For marker there are multiple choices between a user can select. The first one is the standard marker, where a user can select only the marker color:

The second possibility is to use a custom color and custom scale with a custom marker, for example and icon available in Font Awesome catalog:

A user can also use an image from Knowage media as a marker:

Finally a user can use an image from external URL as a marker:

Cluster visualization renders circles of different size where every circle aggregating positions by relative values. A user can zoom in to disaggregate the cluster until he see the single data. For this type of visualization, a user can set size and color of the circle and the size and the color of the font used to display the aggregated value:

When heatmap is selected, a user can display values by areas colored by a color range from green to red where the values are respectively lower and higher. Setting the radius and the blur, a user can specify the scale of the areas and

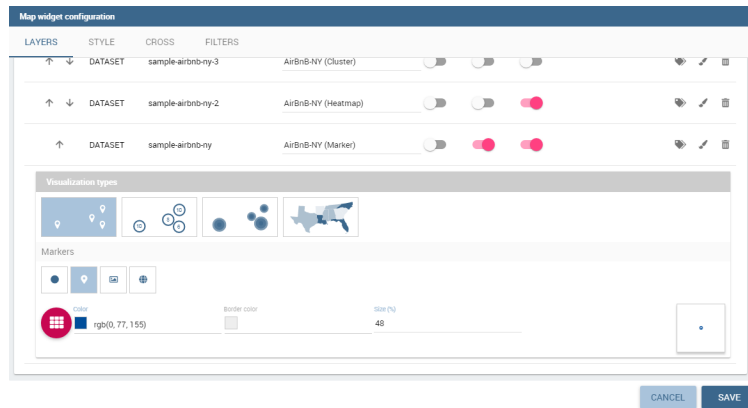


Fig. 1.187: Style configuration for every layer.



Fig. 1.188: Standard marker configuration.

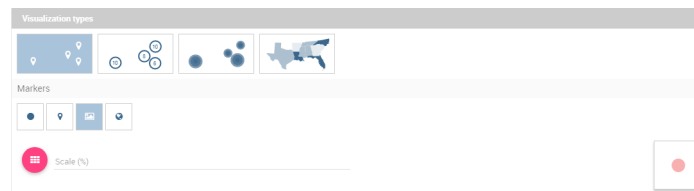


Fig. 1.189: Custom marker configuration.

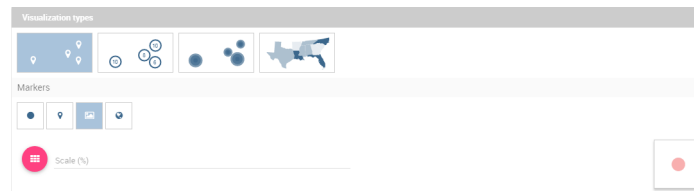


Fig. 1.190: Marker from Knowage images.

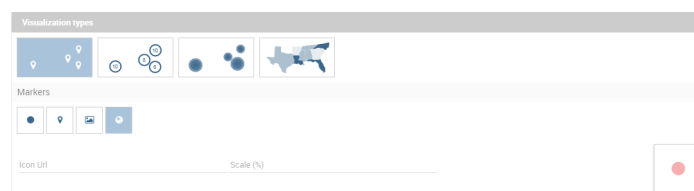


Fig. 1.191: Marker from Knowage images.

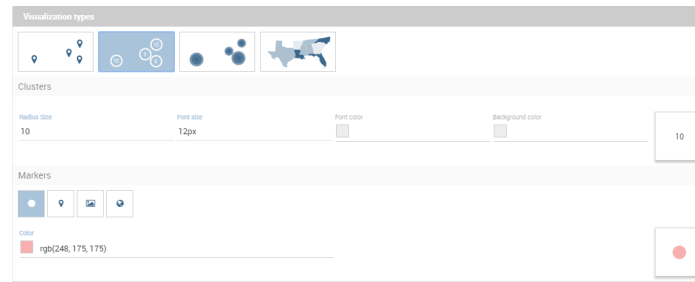


Fig. 1.192: Cluster configuration.

the scale of the blur around it:

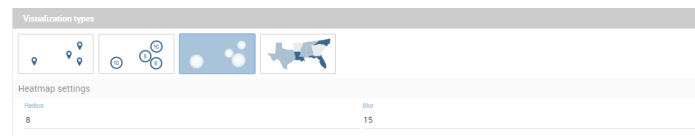


Fig. 1.193: Heatmap configuration.

The choropleth visualization allows a user to precisely associate values to areas, very useful when spatial attribute specify a geometry instead of a single point. The classes method specify the subdivision algorithm and the classes number specify how many subdivision to make; the colors specify the start and the end of the range color that will follow the same range of the values:

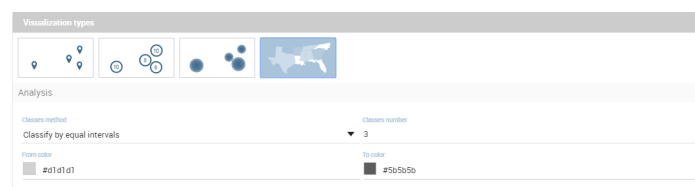


Fig. 1.194: Choropleth configuration.

1.4.1.11 Discovery Widget

The Discovery Widget is used to easily use and navigate into a Solr Dataset using facets aggregation and a table results. In order to make searches, aggregations using facets and so on, after selecting the Solr dataset it is possible to choose the fields that should be shown as the result. The table result can also be configured to show a limited set of fields, as you can see in the widget configuration:

Settings

The settings tab contains the management of the 3 elements that compose a directive:

- Data table: enabled by default is the grid containing data. You can choose the number of item per page.
- Facets: if enabled the sidepanel with the facets will appear. It is also possible to configure facets options:
 - *enable selection on facets*, if enabled a user click on the facets will throw a cockpit selection instead of just filtering the table.
 - *closed by default*, if enabled the facets will be visible as closed groups by default.

Q

datenotifica	↑	aggregazio...	annoemiss...	categoriaemissione	annull...	codiceb...	codic...	com...	comu...	conso...
October 25, 2018 2:00 AM	13482	NUOVO CIRCOI	2015	Standard	0	E289	CONSEGTI	CIRC001	IMOLA	Si
October 26, 2018 2:00 AM	12885	NUOVO CIRCOI	2015	Standard	0	E289	CONSEGTI	CIRC001	IMOLA	Si
September 15, 2017 2:00 AM	12724	NUOVO CIRCOI	2015	Standard	0	E289	CONSEGTI	CIRC001	IMOLA	Si
October 18, 2018 2:00 AM	12476	NUOVO CIRCOI	2015	Standard	0	E289	CONSEGTI	CIRC001	IMOLA	Si
February 21, 2018 1:00 AM	10472	NUOVO CIRCOI	2015	Standard	0	E289	CONSEGTI	CIRC001	IMOLA	Si
October 22, 2018 2:00 AM	10267	NUOVO CIRCOI	2015	Standard	0	E289	CONSEGTI	CIRC001	IMOLA	Si
November 20, 2018 1:00 AM	9306	NUOVO CIRCOI	2015	Standard	0	E289	CONSEGTI	CIRC001	IMOLA	Si
February 19, 2018 1:00 AM	9243	NUOVO CIRCOI	2015	Standard	0	E289	CONSEGTI	CIRC001	IMOLA	Si
September 5, 2017 2:00 AM	8884	NUOVO CIRCOI	2015	Standard	0	E289	COMGIACI	CIRC001	IMOLA	Si
February 22, 2018 1:00 AM	8797	NUOVO CIRCOI	2015	Standard	0	E289	COMGIACI	CIRC001	IMOLA	Si
denominazione lista	↑	NUOVO CIRCOI	2015	Standard	0	E289	COMGIACI	CIRC001	IMOLA	Si
FERRARA_ING_FISCALI_CDS_65_20102017	85262									
TARI_CARICO_2016_I_LOTTO	81067									
SANZIONI CODICE DELLA STRADA (CDS)	66519									
FERRARA_ING_FISCALI_CDS_CO08BL	61348									
TARI2017_PARZIALI LOTTO II	61275									
Accertamenti Tares (file corretto)	53251									
TARI 2014 I LOTTO OMESSI VERSAMENTI	52616									
TARI CARICO 2016 I LOTTO II	50000									

1 to 10 of 216284 | < < Page 1 of 216284 > > |

Edit Discovery widget

DATASET SETTINGS STYLE CROSS FILTERS

Dataset

emissioni_rer_solr (emissioni_rer_solr)

Columns

Column	Alias	Type	Style	Show Column	Show Facet	Enable text search
aggregazione	aggregazione	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
annoemissione	annoemissione	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
annullato	annullato	MEASURE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
categoriaemissione	categoriaemissione	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
codicebelfiore	codicebelfiore	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
codicecausalenotifica	codicecausalenotifica	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
commissa	commissa	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
comune	comune	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
consolidato	consolidato	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
datadocumenti	datadocumenti	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
dataemissione	dataemissione	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
datenotifica	datenotifica	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
datapostalizzazione	datapostalizzazione	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
denominazioneentrata	denominazioneentrata	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
denominazione lista	denominazione lista	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
discaricato	discaricato	MEASURE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
emissioni	emissioni	MEASURE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

CANCEL SAVE

- *facets column width*, this setting allows to choose the dimension of the facets column in px, rem or percentage values.
- *facets max number*, this setting allows to choose the maximum number of facets visible for every field.
- Text search: if enabled a searchbar will appear at the top of the widget. It is possible to set a default search for widget initialization.

Important The options “show column” and “show facets” are only frontend side. They don’t affect the real backend Solr query, discovery widget will search for every field even though they are frontend omitted.

Facets column ordering

It is possible to change the facets column ordering, for example if there is the need to move up a field.

As shown in this example, “aggregazione” should be shown upper, just go to the edit widget section:

And change the columns order dragging the field to the right position.

Changing Date Format for discovery table date columns

It is also possible to change the format used to show date columns inside discovery table: In order to do that, click on style for date columns fields in edit mode

And change the “date format” property

1.4.1.12 Python/R Widget

The Python/R widgets allow to directly embed Python or R scripts inside the cockpit in order to create advanced custom analytics.

In the editor tab it is possible to add the script that will be sent to the execution engine.

Before writing the code it is necessary to specify the **type** of the output produced by the script. Knowage has support for two different output types:



- Image

annoemissione 	
2018	795855
2017	644009
2016	418385
2015	216086
2014	71076
2106	6381
2019	3881
aggregazione 	
MODENA	505576
NUOVO CIRCONDARIO IMOLESE	367450
FERRARA	273149
CASALECCHIO DI RENO	212909
FORLI'	190713
RAVENNA	157556
RIMINI	156257
CESENA	133426
CESENATICO	80068

Edit Discovery widget							
DATASET SETTINGS STYLE CROSS FILTERS							
Dataset							
emissioni_rer_sor (emissioni_rer_sor)							
Columns							
Column	Alias	Type	Style	Show Column	Show Facet	Enable text search	
aggregazione	aggregazione	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
categoriaemissione	categoriaemissione	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
annoemissione	annoemissione	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
annullato	annullato	MEASURE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
codicebelfiore	codicebelfiore	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

aggregazione	
MODENA	505576
NUOVO CIRCONDARIO IMOLESE	367450
FERRARA	273149
CASALECCHIO DI RENO	212909
FORLI'	190713
RAVENNA	157556
RIMINI	156257
CESENA	133426
CESENATICO	80068
FAENZA	51491
annoemissione	
2018	795855
2017	644009
2016	418385
2015	216086
2014	71076
2106	6381

Columns

Column	Alias	Type	Style	Show Column <input checked="" type="checkbox"/>	Show Facet <input checked="" type="checkbox"/>	Enable text search <input checked="" type="checkbox"/>
datanotifica	datanotifica	ATTRIBUTE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
datapostalizzazione	datapostalizzazione	ATTRIBUTE	 Column Style	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

COLUMN STYLE

Font-color

Select a color

Background-color

Select a color

Font-family

▼ Font Size

px, rem or % measure units are available

Font-weight

▼ Font Style

Cell horizontal align...

▼

Date Format

19/12/2019 15:50:37

▼

CANCEL

SAVE

COLUMN STYLE

Font-color
 Select a color

Background-color
 Select a color

Font-family

▼ Font Size
px, rem or % measure units are available

December 19, 2019 3:50 PM

Dec 19, 2019 3:50 PM

19/12/2019 15:50:37

19/12/2019 15:50

December 19, 2019

Python Widget Configuration

DATASET

PYTHON EDITOR

ENVIRONMENT

STYLE

CROSS

FILTERS

Output type
Image

Output file name
my_output.png

Python script

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3
4 toy_dataset = pd.DataFrame({'x': 'name', 'y': 'age'})
5 plt.savefig('my_output.png')
```

Available Tags

Columns Data▼

Parameters▼

CANCEL

SAVE

- HTML

It is also necessary to specify the name of the file in which the script will save its output.

The Dataset tab allows the user to select a dataset that will be accessible directly from the code. After choosing a dataset the list of available columns will be shown. Here it is also possible to order the dataset according to a column and to select the ordering type (ascending or descending).

Python Widget Configuration

DATASET PYTHON EDITOR ENVIRONMENT STYLE CROSS FILTERS

Dataset
toy_dataset (toy_dataset)

TABLE COLUMNS ADD COLUMN ADD CALCULATED FIELD

Name	Alias	Type	Aggregation
id	<input checked="" type="checkbox"/> id	MEASURE	<input checked="" type="checkbox"/> SUM
name	<input checked="" type="checkbox"/> name	ATTRIBUTE	
age	<input checked="" type="checkbox"/> age	MEASURE	<input checked="" type="checkbox"/> SUM
gender	<input checked="" type="checkbox"/> gender	ATTRIBUTE	
state	<input checked="" type="checkbox"/> state	ATTRIBUTE	
num_children	<input checked="" type="checkbox"/> num_children	MEASURE	<input checked="" type="checkbox"/> SUM
num_pets	<input checked="" type="checkbox"/> num_pets	MEASURE	<input checked="" type="checkbox"/> SUM

CANCEL SAVE

Fig. 1.195: Dataset selection

Once a dataset has been chosen, it will be possible to access it directly from the code via a **dataframe** variable. This variable will have the same name of the dataset label.

The Environment tab allows the user to choose among a list of available Python/R environments previously defined inside the **Configuration Management**. To support this kind of choice a list of available libraries is displayed for each selected environment.

Inside Python and R scripts it is possible to access analytical drivers by the usual placeholder syntax $\$P\{\}$.

Warning: This widget is sensible to associative logic, meaning that the widget is updated every time an association is changed, but it DOES NOT trigger associative logic itself.

1.4.1.13 Custom Chart Widget

The Custom Chart allows the user to directly embed html, css and js code using a supported external chart library and integrating with Knowage data and interactions using custom API.

Important: Chart libraries

As a default Knowage supports natively Chart.js (version 1.0.2) for the Community edition and Highcharts.js (version 7.1.1) for the Enterprise Edition. In CE and EE, Knowage supports d3.js library (version 3.5.5). It is possible also to include other libraries adding the CDN script tag in the html Editor. Be aware that url not set in the whitelist will be deleted on save. To use this import use the kn-import tag like the following example:

Python Widget Configuration

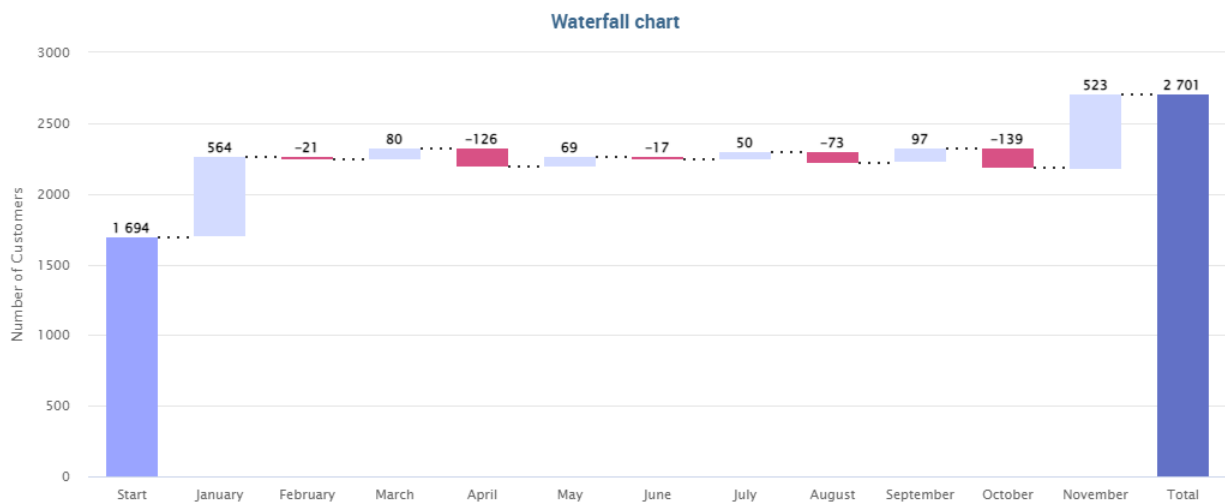
DATASET PYTHON EDITOR ENVIRONMENT STYLE CROSS FILTERS

Python environment
python.virtualenv.1.frontend.url

Library	Version
Werkzeug	0.16.1
urllib3	1.25.8
tornado	6.0.3
requests	2.22.0
PyYAML	5.3
PyJWT	1.7.1
Pillow	7.0.0
packaging	20.1
MarkupSafe	1.1.1
Jinja2	2.11.0
itsdangerous	1.1.0
idna	2.8
Flask	1.1.1
Flask-Cors	3.0.8
Click	7.0

CANCEL SAVE

Fig. 1.196: Environment selection



```
<kn-import src="yourCDNurl"></kn-import>
```

This widget will be available only if the *create custom chart widget* option is enable for the specified user's role.

The Edit section of the widget is composed by five tabs: dataset, editor, style, cross and filters.

The **dataset tab** allows to select a specific dataset to use as a referral for the API. Once the dataset has been selected a table with the columns list will appear below. In the table will be possible to change column alias, the column aggregation for measures and delete columns interacting with the selected column line. Clicking on *add column* or *add calculated field* buttons on top a popup will appear allowing to choose one of the dataset column to add or to insert the calculated field formula.

Edit custom chart widget

DATASETEDITORSTYLECROSSFILTERS

Dataset
prova_customChart (prova_customChart)

ADD COLUMNADD CALCULATED FIELDUSE FUNCTION

Name	Alias	Type	Data Type	Aggregation
PRODUCT_FAMILY	<input checked="" type="checkbox"/> PRODUCT_FAMILY	ATTRIBUTE	string	
PRODUCT_DEPARTMENT	<input checked="" type="checkbox"/> PRODUCT_DEPARTMENT	ATTRIBUTE	string	
PRODUCT_CATEGORY	<input checked="" type="checkbox"/> PRODUCT_CATEGORY	ATTRIBUTE	string	
PRODUCT_SUBCATEGORY	<input checked="" type="checkbox"/> PRODUCT_SUBCATEGORY	ATTRIBUTE	string	
UNIT_SALES	<input checked="" type="checkbox"/> UNIT_SALES	MEASURE	float	<input checked="" type="checkbox"/> SUM
STORE_COST	<input checked="" type="checkbox"/> STORE_COST	MEASURE	float	<input checked="" type="checkbox"/> SUM
STORE_SALES	<input checked="" type="checkbox"/> STORE_SALES	ATTRIBUTE	float	

CANCELSAVE

The **Editor tab** allows to insert custom code and it's splitted into three components: CSS, HTML, JavaScript.

Edit custom chart widget

DATASETEditorSTYLECROSSFILTERS

css

html

JavaScript

```

1 var canvas = document.getElementById('myChart');
2 var ctx = document.getElementById(myChart).getContext('2d');
3 var tempLabels = datasetstore.hierarchy({levels:['first_name'], measures:['int':'MAX']}).getChildId(e).name;
4 var labels = datasetstore.filter({first_name:'Ada'}).getColumn('first_name');
5
6 labels[0] = tempLabels;
7
8 var data = {
9   labels: labels,
10  datasets: [
11    {
12      label: '# of Votes',
13      data: datasetstore.filter({first_name:'Ada'}).getColumn('int'),
14      fillColor: "rgba(226,151,236,0.8)"
15    }
16  ]
17 };
18 var options = {
19   scales: {
20     yAxes: [
21       {
22         ticks: {
23           beginAtZero: true
24         }
25       }
26     ]
27   };
28   var myChart = new Chart(ctx).Bar(data,options);
29
30 canvas.onclick = function(evt){
31   var point = myChart.getBarsAtEventCross(evt)[0];
32   datasetstore.clickManager('first_name',point,label)
33 }

```

CANCELSAVE

The CSS component allows to insert css classes that will be used by the HTML code of the widget. It's also possible to use `@import` command if the referred url is inside the whitelist.

The HTML component allows to insert HTML tags in order to create a structure to host the custom chart and additional structural informations.

The JavaScript component is the code section, and allows to insert the custom chart code, custom Javascript code and the API usage.

To use the API the keyword is **datastore**. Datastore is an object that contains the actual data; it has methods to iterate over results and get all values plus some other methods as the following:

getDataArray

returns: *data array*

params: *custom user function*

example:

```
1 datastore.getDataArray(function(record){
2     return {
3         name: record.city,
4         y: record.num_children_at_home
5     }
6 })
```

result:

```
1 [
2     {
3         name: 'New York',
4         y: 5
5     },
6     {
7         name: 'Boston',
8         y: 3
9     }
10 ]
```

getRecords

returns: array of objects; each object has nameOfDsColumn: value

params: no params

example:

```
1 datastore.getRecords()
```

result:

```
1  [  
2      {  
3          city: 'New York',  
4          total_children: 5,  
5          country: 'USA'  
6      },  
7      {  
8          name: 'Boston',  
9          total_children: 3,  
10         country: 'USA'  
11     }  
12 ]  
13
```

getColumn

returns: array of *unique* values for one dataset column

params: dataset's column name

example:

```
1  datastore.getColumn('country')
```

result:

```
1  ['USA', 'Mexico', 'Canada']
```

getSeriesAndData

returns: array of series with data for each series

params: serie/measure name, custom user function

example:

```
1  datastore.getSeriesAndData('PRODUCT_FAMILY', function(record){  
2      return {  
3          y: record.UNIT_SALES,  
4          name: record.QUARTER  
5      }  
6  })
```

result:

```
1  [  
2      {  
3          name: 'Drink',  
4          data: [  
5              {  
6                  y: 5000,
```

(continues on next page)

(continued from previous page)

```

7         name: 'Q1'
8     },
9     {
10        y: 7000,
11        name: 'Q2'
12    }
13 ]
14 },
15 {
16     name: 'Food',
17     data: [
18         {
19             y: 6000,
20             name: 'Q1'
21         },
22         {
23             y: 4000,
24             name: 'Q2'
25         },
26         {
27             y: 3000,
28             name: 'Q3'
29         }
30     ]
31 }
32 ]
33 }
34 ]
35 ]

```

sort - angular sort service (sorting is executed on the client side)

returns: datastore sorted by dataset's column/s

params: dataset's column name

optional: sort type object {column:'asc/desc'}

example1:

```

1 datastore.sort('STORE_ID') //by default, it is asc
2 OR:
3 datastore.sort({'STORE_ID':'asc'})

```

filter - angular filter service (filtering is executed on the client side)

returns: datastore filtered by some value for dataset's column/s

params: object that contains dataset's columns names for properties -> value to be filtered, an optional boolean to enable the strict comparison (false as default)

example:

```

1 datastore.filter({'QUARTER':'Q1','STORE_ID':'1'}, true)

```

hierarchy

returns: hierarchy object with its functions and tree

params: object that contains property levels -> array of dataset's columns names

optional: same object with optional property measures -> object that contains dataset's columns names for properites
-> aggregation function (sum, min, max)

example:

```
1 var hierarchy = datastore.hierarchy({'levels':['QUARTER','PRODUCT_FAMILY'],'measures': {'UNIT_SALES':'SUM'}})
```

result:

```
1 [
2   {
3     "name": "Q1",
4     "children": [
5       {
6         "name": "Non-Consumable",
7         "children": [],
8         "UNIT_SALES": 7.4571
9       },
10      {
11        "name": "Food",
12        "children": [],
13        "UNIT_SALES": 12
14      }
15    ],
16    "UNIT_SALES": 19.4571
17  },
18  {
19    "name": "Q2",
20    "children": [
21      {
22        "name": "Non-Consumable",
23        "children": [],
24        "UNIT_SALES": 9.9429
25      },
26      {
27        "name": "Food",
28        "children": [],
29        "UNIT_SALES": 7.2
30      }
31    ],
32    "UNIT_SALES": 17.1429
33  }
34 ]
```

getChild

returns: node of hierarchy (node is Node object)

params: index of child in hierarchy

example:

```
1 hierarchy.getChild(0)
```

result:

```

1 {
2   "name": "Q1",
3   "children": [
4     {
5       "name": "Non-Consumable",
6       "children": [],
7       "UNIT_SALES": 7.4571
8     },
9     {
10      "name": "Food",
11      "children": [],
12      "UNIT_SALES": 12
13    }
14  ],
15  "UNIT_SALES": 19.4571
16 }

```

getLevel

returns: array of nodes of hierarchy on specific level

params: index of level in hierarchy

example:

```

1 hierarchy.getLevel(0)

```

result:

```

1 [
2   {
3     "name": "Q1",
4     "children": [
5       {
6         "name": "Non-Consumable",
7         "children": [],
8         "UNIT_SALES": 7.4571
9       },
10      {
11        "name": "Food",
12        "children": [],
13        "UNIT_SALES": 12
14      }
15    ],
16    "UNIT_SALES": 19.4571
17  },
18  {
19    "name": "Q2",
20    "children": [
21      {
22        "name": "Non-Consumable",
23        "children": [],
24        "UNIT_SALES": 9.9429
25      },
26      {
27        "name": "Food",
28        "children": [],
29        "UNIT_SALES": 7.2
30      }
31    ]
32  }
33 ]

```

(continues on next page)

(continued from previous page)

```
31         ],  
32         "UNIT_SALES": 17.1429  
33     }  
34 ]
```

node is an instance of Node object. It has convenient functions to explore the node:

```
1 var node = hierarchy.getChild(0)
```

result:

```
1 {  
2     "name": "Q1",  
3     "children": [  
4         {  
5             "name": "Non-Consumable",  
6             "children": [],  
7             "UNIT_SALES": 7.4571  
8         },  
9         {  
10            "name": "Food",  
11            "children": [],  
12            "UNIT_SALES": 12  
13        }  
14    ],  
15    "UNIT_SALES": 19.4571  
16 }
```

getValue

returns: a measure's value for a specific hierarchy's child(node)

params: dataset's measures's name

example:

```
1 node.getValue('UNIT_SALES')
```

result: 19.4571

getChild

returns: a specific node's child

params: index of nodes's child

example:

```
1 node.getChild(0)
```


result:

```
1 {  
2   "name": "Non-Consumable",  
3   "children": [],  
4   "UNIT_SALES": 7.4571  
5 }
```

getParent

returns: a node parent of specific child

params: no params

example:

```
1 node.getChild(0).getParent()
```

result:

```
1 {  
2   "name": "Q1",  
3   "children": [  
4     {  
5       "name": "Non-Consumable",  
6       "children": [],  
7       "sales": 7.4571  
8     },  
9     {  
10      "name": "Food",  
11      "children": [],  
12      "sales": 12  
13    }  
14  ],  
15  "sales": 19.4571  
16 }
```

getChildren

returns: an array of node's children

params: no params

example:

```
1 node.getChildren()
```

result:

```
1  [
2      {
3          "name": "Non-Consumable",
4          "children": [],
5          "sales": 7.4571
6      },
7      {
8          "name": "Food",
9          "children": [],
10         "sales": 12
11     }
12 ]
```

getSiblings

returns: an array of node siblings to a specific child

params: no params

example:

```
1  node.getChild(0).getSiblings()
```

result:

```
1  [
2      {
3          "name": "Non-Consumable",
4          "children": [],
5          "sales": 7.4571
6      },
7      {
8          "name": "Food",
9          "children": [],
10         "sales": 12
11     }
12 ]
```

variables

returns: a key/value object with all the declared variables and values

params: no params

example:

```
1  var myvariables = datastore.variables;
```

result:

```

1 {
2   variableCity: 'New York',
3   variableNum: 100
4 }

```

profile

returns: a key/value object with all the declared profile attributes for the user

params: no params

example:

```

1 var user = datastore.profile;

```

result:

```

1 {
2   name: 'My Name',
3   tenant: 'Knowage',
4   customProfileAttribute: 'Test value',
5   role: 'user'
6 }

```

selections

returns: an array with all the selections done; each selection has informations about the dataset where the selection has been done, the column e the value passed through the selection

params: no params

example:

```

1 var activeSelection = datastore.selections;

```

result:

```

1 [
2   {
3     "ds": "FOODMART_SALES",
4     "column": "PRODUCT_FAMILY",
5     "value": "Food"
6   },
7   {
8     "ds": "FOODMART_COST",
9     "column": "QUARTER",
10    "value": "Q1"
11   }
12 ]

```

parameters

returns: a key/value object with all the parameters associated to the dashboard

params: no params

example:

```
1 var myParameters = datastore.parameters;
```

result:

```
1 {  
2   "par_family": "Non-Consumable",  
3   "par_number": 10  
4 }
```

It is also possible to interact with the other cockpit widgets, to do so it's possible to use the **clickManager**:

```
1 datastore.clickManager(columnName, columnValue);
```

This method can be added everywhere the code is managing a click event, and will notify Knowage about the interaction. The default case (if no cross-navigation or preview-navigation is set) will throw a selection filter with the dataset column name and column value set in the method. If a cross-navigation or a preview has been set in the cross tab, those will have priority on the selection and will trigger the specified interaction. The dynamic values used will be the ones set in the method arguments.

Warning: Whitelist

For security reasons no dangerous Javascript code can be added to html tags. Every tag considered dangerous will be deleted on save by the system. Base paths to external resources (images, videos, anchors, CSS files and inline frames) must be declared within `TOMCAT_HOME/resources/services-whitelist.xml` XML file inside Knowage Server, otherwise those external links will be removed by the system. This whitelist file contains safe and trusted websites, to restrict end users of providing unsafe links or unwanted web material. Knowage Server administrator can create or edit it (directly on the file system) to add trusted web sites. Here below you can see an example of `services-whitelist.xml` file; as you can see, its structure is quite easy: `baseurl` attributes refer to external services, `relativepath` must be used for Knowage Server internal resources instead:

```
linenos  
  
<?xml version="1.0" encoding="UTF-8"?> <WHITELIST>  
  
  <service baseurl="https://www.youtube.com" /> <service baseurl="https://player.vimeo.  
com" /> <service baseurl="https://vimeo.com" /> <service baseurl="https://media.giphy.  
com" /> <service baseurl="https://giphy.com" /> <service baseurl="https://flic.kr" /> <ser-  
vice relativepath="/knowage/themes/" /> <service relativepath="/knowage/icons/" /> <ser-  
vice relativepath="/knowage/restful-services/1.0/images/" />  
  
</WHITELIST>
```

Like other widgets the **“Cross”**, **“Style”**, and the **“Filters”** tab are available in order to set the general style options for the widget and to filter the results displayed in the Custom Chart widget.

1.4.1.14 Cross Navigation

Warning: Cross navigation tab is only for technical users

Due to the fact that parameters can only be managed by technical user the cross navigation cannot be implemented by the final user.

All widgets (except selector and active selections) have the **Cross** tab available, that allows the user to interact with the widget in different ways:

- setting a Cross-navigation between different documents
- setting a Preview of a specific dataset in a popup
- opening an external link

The interactions are mutually exclusive, so just one type can be chosen for every widget.

Cross-navigation

The cross-navigation gives the possibility to connect two documents clicking on a widget as a starting point, opening the second one as arrival and showing the breadcrumbs on top.

Fig. 1.197: Table widget cross-navigation configuration.

To enable one, first of all is necessary to set a cross navigation inside the “*Cross-navigation definition*” functionality from the administration menu. There is possible to set the starting/arrival point and to set the connection between different output/input parameters.

After this passage you will have the possibility to enable the cross-navigation in the widget’s tab and choose the user interaction that will start the navigation, if more than one are available.

The most complex example is the table widget cross-navigation, because it allows 3 different interactions:

- *Click on the whole row*, where the interaction will start clicking on any row
- *Click on a single column*, where the editor will choose a specific column that will start the interaction (the user will see the column values underlined)
- *Click on an icon*, where the editor will choose an icon positioned to the right side that will start the interaction.

Once the interaction has been chosen you will be able to select the cross-navigation defined before. If you have created more cross-navigation related to the same document you will be able to choose between them. If you leave this field blank the user will be able to choose that himself.

Preview

The preview configuration is very similar to the cross-navigation one. First of all you will need to enable the navigation using the switch button. After that, you will need to choose an interaction type and a target dataset. The selected dataset will be opened in a popup window but, if you check the “Direct download” property, you will be able to get the dataset content directly in the *download list* functionality.

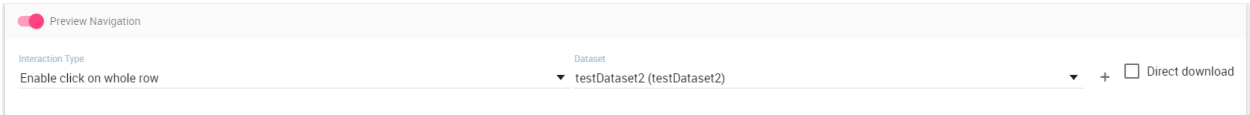


Fig. 1.198: Table widget preview configuration.

Dataset Preview

EXPORT CSV EXPORT XLSX

country string	city string	color string	currency string	gender string	company string
France	Marseille	Turquoise	Euro	Female	Trantow Group
France	Calais	Goldenrod	Euro	Female	Sawayn LLC
Russia	Glinka	Green	Ruble	Male	Hintz, Streich and Rippin
Russia	Uritsk	Crimson	Ruble	Male	Crooks, Hintz and Kiehn
France	Montluçon	Red	Euro	Female	Yundt, Sawayn and Morar
France	Paris 02	Purple	Euro	Male	Hodkiewicz Inc
Russia	Porkhov	Purple	Ruble	Female	Jacobs, Wunsch and Sauer
Russia	Taldan	Blue	Ruble	Male	Schaden, Shields and Toy
Russia	Vodstroy	Green	Ruble	Female	Zemlak-Reynolds
Russia	Novoenninskiy	Crimson	Ruble	Female	Bins-Langworth
Russia	Bobrov	Khaki	Ruble	Male	Rice LLC
Russia	Kukuhtan	Purple	Ruble	Female	Oberbrunner-Cole
Russia	Vostryakovo	Violet	Ruble	Female	Botsford Group
France	Marseille	Fuscia	Euro	Male	Stark, Mann and Ortiz
Russia	Malinovsky	Green	Ruble	Female	Littel and Sons

Fig. 1.199: Preview example.



Fig. 1.200: Download list functionality icon.

Cross and Preview Parameters



Output parameters list			
	Type	Value	
<input checked="" type="checkbox"/> parameter2	Static	▼ 42	
<input type="checkbox"/> parameter1			

Fig. 1.201: Parameters example.

Both cross-navigation and preview navigation share the parameters management. If one or more output parameters are available, you will find the list below the navigation page. You can choose which values to use clicking on the checkbox at the right side of the name, then you will have the possibility to choose between different modes to get the value:

- *Static*, entering a static value
- *Dynamic*, passing the value of the column at the selected row (or passing the column name if *Selected Column Name* is chosen)
- *Selection*, passing the current value of the selection for the specified dataset and column. Warning: the selection is not triggered clicking on the navigation, so to pass this value the selection must already be present.

Once the user will click on the widget, those parameters will be evaluated before the navigation.

Link

The link configuration is very similar to the cross-navigation one too. First of all you will need to enable the link navigation using the switch button. Then you will be able to create one or more link navigations. You will need to choose an interaction type and a base url. The base url will be the url opened by the user click. You can also decide the type of link between the opening of a new page or the document replace opening in the same page.



 Link				ADD LINK
Interaction Type	Column	Base URL	Link type	
Enable click on a single column	▼ first_name	▼ http://www.test.com <small>The URL of the page that will be opened</small>	Open in new page	
URL Parameters				ADD PARAMETER
Parameter key	Parameter type			
testJSON	JSON			
<pre> 1 { 2 "parameters": { 3 "variable": \$v(myvariable), 4 "selection": \$p(mytable), 5 "parameterValue": \$p(myParameter) 6 } 7 }</pre>				

Fig. 1.202: Link configuration with JSON parameter example.

Link Parameters

The link parameters are different from the previous because they will be used to make the baseurl selected more precise. The resulting example url will be something similar: `http://www.knowage-suite.com?**parameter1**=value&**parameter2**=value2`

To do so you will have to create and name different parameters, the selected name will be the one used in the url construction. You can choose how to get the parameter value with the following modes:

- *Static*, entering a static value
- *Dynamic*, passing the value of the column at the selected row (or passing the column name if *Selected Column Name* is chosen)
- *Selection*, passing the current value of the selection for the specified dataset and column. Warning: the selection is not triggered clicking on the navigation, so to pass this value the selection must already be present.

- *Analytical Driver*, passing a document parameter (input driver)
- *JSON*, passing an escaped JSON inside the url parameter. You can use the editor to create the desired JSON and use the placeholder to set the value at runtime. The placeholders are the usuals $\$F\{field\}$ for the fields, $\$P\{parameter\}$ for the parameters, $\$V\{variable\}$ for the variables.
- *JWT*, passing the JWT token of the user

1.4.1.15 Widget properties

Once one or more (above mentioned) widgets have been implemented, the technical user has some more options exploring the icon available at the left bottom corner of the widget itself, as Figure below highlights.



Fig. 1.203: Widget properties.

Here the user can:

- move the widget in the cockpit area at convenience;
- modify its dimension;
- delete it;
- activate the on-click interaction of the widget with the other ones;
- activate the updating of widget data due to the interaction with other widgets.

When executing the cockpit in visualization mode, the user has also some more options for widgets. For all widget





the user can use the icon  to expand the widget to all page and use the icon  to reduce it again. There are also two other widget options: using the icon  it is possible to capture the screenshot of the widget and clicking on the icon  the data plotted on a chart or displayed in a table or crosstab are exported in an Excel file.

Chart widget are endowed with an additional option that allows the user to change the chart type, as you can see in Figure below.

In this case, the available chart types are: parallel, scatter, wordcloud, line, radar, bar and pie. These charts depends on the original chart type, not all can be available at the same time.

Pay attention though to the fact that when grouping functions have been used, the change chart type may not report the same level of aggregation. In fact, not all type of chart allows the grouping function. Refer to Chart types in detail to read more about each chart type configuration. Pay also attention when a two-series chart is chained with a single-series one. For instance the parallel chart works only when (at least) two series have been set, while the wordcloud works with only one series.

1.4.2 General configuration

This option allows the user to manage all cockpit general settings that we are going to describe through images of the interface. Clicking on the **General configuration** button the window in figure below opens. This contains the **General Settings** tab, the **Widget Style** tab and the **CSS Editor** tab.

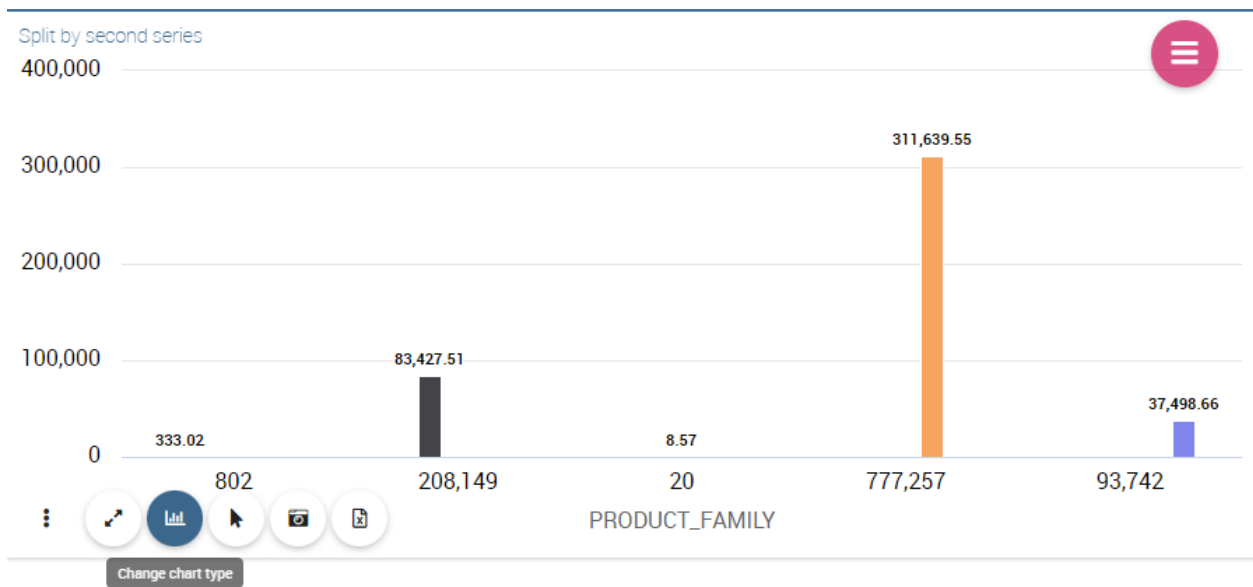


Fig. 1.204: Change chart type button.

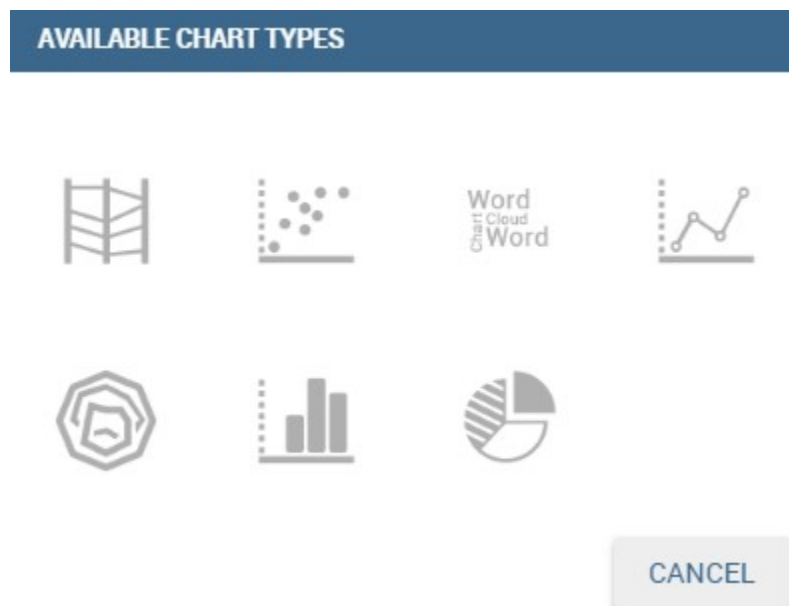


Fig. 1.205: Available chart types.

The image shows the 'General Cockpit Settings' window with the 'GENERAL SETTINGS' tab selected. The window has three tabs: 'GENERAL SETTINGS', 'WIDGETS STYLE', and 'CSS EDITOR'. The 'GENERAL SETTINGS' tab contains the following sections:

- Informations:**
 - Cockpit name: My Document
 - Description: (empty text area with a character count of 0 / 150)
- Background:**
 - Sheets background color: (color picker)
 - Sheets background image url: (text input)
 - Sheets Background size: (text input)
- Menu and Widgets:**
 - ☒ Show Cockpit Menu button on visualization mode
 - ☐ Hide widgets functionalities on visualization mode
 - ☒ Always show selection button
 - ☒ Enable screenshot functionality on widgets
 - ☒ Enable excel export functionality on widgets

Fig. 1.206: General configuration window.

Editing the fields of the first tab, **General Settings**, you can add or change the name and/or the description of your cockpit; moreover here you can choose the sheet color or a background image and its size. In particular, in order to add a background image for the sheets, firstly you have to add the image to the catalogue of the image widget and then copy the link of the image. It is also possible to decide to enable the menu and the widgets functionalities when the document runs in display mode or to disable the screenshot functionality for every widgets.

The second tab, **Widget Style**, (Figure below) allows to configure a default style for the widgets, like borders, shadows, titles and background color.

The image shows the 'General Cockpit Settings' window with the 'WIDGETS STYLE' tab selected. The window has three tabs: 'GENERAL SETTINGS', 'WIDGETS STYLE', and 'CSS EDITOR'. The 'WIDGETS STYLE' tab contains the following sections:

- Titles:**
 - Title text: (text input)
 - Font Size: 16px (with a note: 'px, rem or % measure units are available')
 - Header Height: (text input with a note: 'Height in px of the space available to title')
 - Horizontal alignment: Center
 - Font Family: (dropdown menu)
 - Font Style: Bold
 - Title Color: rgb(59, 103, 140)
 - Title Background Color: (color picker with a note: 'Select a color')
- Borders:**
 - Borders Style: Solid
 - Borders Thickness: 1px (with a note: 'px, rem or % measure units are available')
 - Borders Color: rgb(212, 212, 212)
 - Border radius top left: (text input with a note: 'Use px measure unit, ie: 5px')
 - Border radius top right: (text input with a note: 'Use px measure unit, ie: 5px')
 - Border radius bottom left: (text input with a note: 'Use px measure unit, ie: 5px')
 - Border radius bottom right: (text input with a note: 'Use px measure unit, ie: 5px')
- Padding:**
 - Padding: (text input with a note: 'Use px measure unit, ie: 5px')
- Other Options:**
 - Shadows: Medium (with a note: 'Shadows Size')

Fig. 1.207: Widget style tab.

The third tab allows to specify CSS properties for the whole cockpit and widgets. Here is also possible to override common CSS properties of the widgets. The editor will highlight possible syntax errors.



Fig. 1.208: Css editor tab.

1.4.3 Data configuration

This feature manages the data storage and usage. In fact, here there is the possibility to save data in cache, create associations between datasets, create indexes on cached data, schedule the (data) refresh frequency and so on. Referring to the figure below, the feature is implemented through several tabs: the **Source** tab, the **Associations** tab, the **indexes**, the **Frequency** and the **Template** tab.

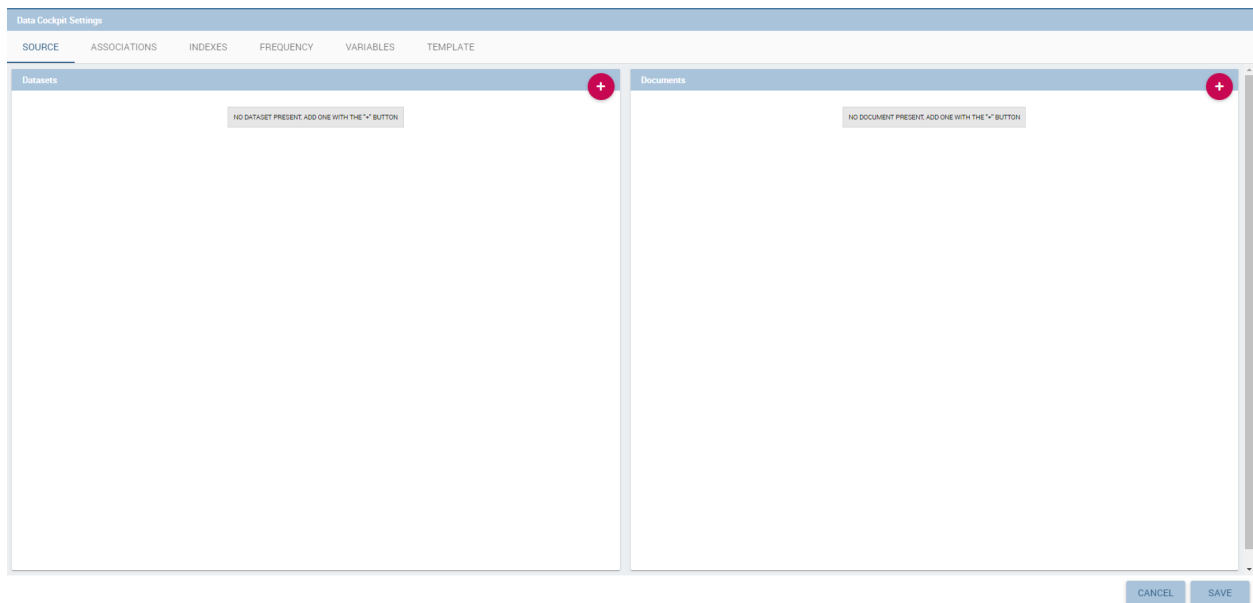


Fig. 1.209: Data configuration window.

1.4.3.1 Source

The Source tab is split into two areas. On the left side the user can find the list of those dataset that are currently used by the cockpit. Here it is possible to add new dataset that will be passed to widgets. In other words, datasets inserted in this area will be listed in the dataset combobox of widgets like the Table, the Pivot Table and the Chart one. Note that the user can delete datasets as well.

1.4.3.1.1 Parametric sources management

If the user is adding a parametric dataset the window will exhibit them in an expandable box right below. It is also mandatory to give default values or to associate proper drivers to the document to secure its correct execution. By the way, a final user has no access to parametric dataset and he/she cannot handle analytical drivers, therefore **parametric sources can be managed only by an admin user**. We stress that the user must also type the driver name in the field box as highlighted in Figure below. You can type it manually or use the look up just aside the parameter line.

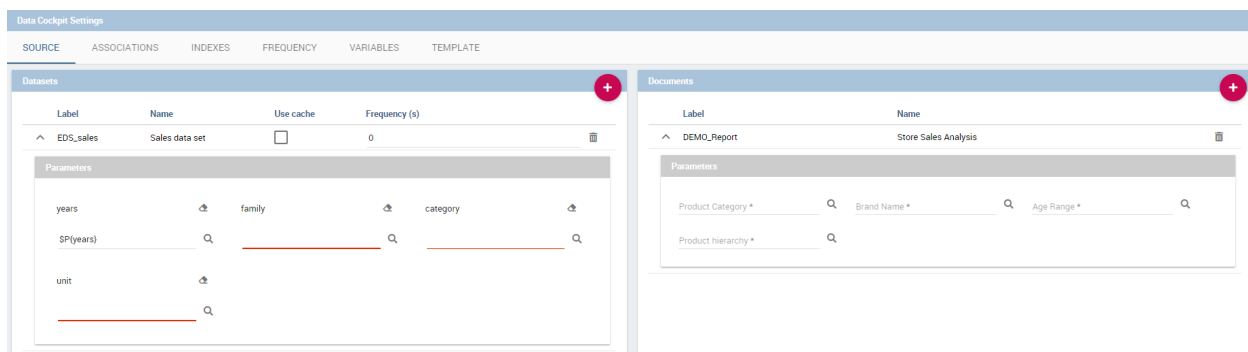


Fig. 1.210: Dataset management.

On the right side of the window the user finds the list of external documents that can be added to the cockpit (through Document widgets), or as well as for the dataset case, of documents that are already in use in (previously set) Document widgets. In the occurrence of Associations parametric documents, parameter boxes are shown below. Note that it is mandatory to link them to analytical drivers (previously hooked to the document) or be assigned a fixed (default) value.

1.4.3.2 Associations

If your goal is to show data from a single dataset, it is not necessary to define any association. *Associations should be set within the designer when widgets are built on different datasets*. Associations can be set with the elements: dataset columns, dataset parameters and document parameters. Note that to implement an association the user must have at least one column. We show some examples in the following.

The following figure shows the association between two datasets. In this case the user must detect one field from the first dataset, the same field (in terms of values) in the other one. The relation will appear right below. Click on the save button to confirm the association. If the associations rely on multiple columns the user must add them one by one.

The same procedure can be done in the case of dataset columns and dataset parameters, as shown below.

Another example is supplied in Figure below. Here the association is performed between a dataset column and document parameter.

Once you have defined the associations, as soon as you refresh one widget, all related widgets are refreshed simultaneously on data update.

Auto Detect functionality

TestASSAlias

Data Cockpit Settings

SOURCE ASSOCIATIONS INDEXES FREQUENCY VARIABLES TEMPLATE

TEST_01	TEST_02
EXPORT_VARIABLE (String)	QUARTER (String)
QUARTER (String)	THE_DATE (TIMESTAMP)
THE_DATE (TIMESTAMP)	STORE_ID (BigDecimal)
STORE_ID (BigDecimal)	UNITS_ORDERED (BigDecimal)
PRODUCT_FAMILY (String)	UNITS_SHIPPED (BigDecimal)
LINK (String)	WAREHOUSE_COST (BigDecimal)
UNIT_SALES (BigDecimal)	SUPPLY_TIME (BigDecimal)
STORE_COST (BigDecimal)	UNIT_SALES (BigDecimal)
STORE_COST_NULL (BigDecimal)	PRODUCT_FAMILY (String)
STATIC_VALUE (String)	

Associations List

AUTO DETECT CLEAR ALL

TEST_01.QUARTER ↔ TEST_02.QUARTER

Fig. 1.211: Associations between dataset columns.

Data Cockpit Settings

SOURCE ASSOCIATIONS INDEXES FREQUENCY VARIABLES TEMPLATE

TEST_MULTIVALUE_CUSTOMER_LIST	TEST_MULTIVALUE_PAR_ASSOC
NAME (String)	PRODUCT_FAMILY (String)
YEARLY_INCOME (String)	PRODUCT_DEPARTMENT (String)
GENDER (String)	QUARTER (String)
CITY (String)	NUM_PRODUCTS_SOLD (BigDecimal)
	SP(par_customer) (String)

Associations List

AUTO DETECT CLEAR ALL

TEST_MULTIVALUE_PAR_ASSOC.SP(par_customer) ↔ TEST_MULTIVALUE_CUSTOMER_LIST.NAME

Fig. 1.212: Associations between dataset column and dataset parameter.

Data Cockpit Settings

SOURCE ASSOCIATIONS INDEXES FREQUENCY VARIABLES TEMPLATE

CUSTOMER_TABLE	Report-single-value-parameter
customer_id (Integer)	SP(state) (String)
account_num (Long)	
lname (String)	
fname (String)	
mi (String)	
address1 (String)	
address2 (String)	
address3 (String)	
address4 (String)	
city (String)	
state_province (String)	

Associations List

CLEAR ALL

CUSTOMER_TABLE.country ↔ Report-single-value-parameter.SP(state)

Fig. 1.213: Associations between dataset column and document parameter.

To correctly set up meaningful associations, the user must have knowledge of extracted data contained in each dataset. There might be though possible matches between datasets that are not known by the user or the columns used are not so similar as thought. Therefore the user can demand to Knowage to retrieve all possible columns that match, to report the percentage of matching values and get knowledge of other possible unexpected matches.

In the “Data cockpit settings” seen in the previous section, we find the “Auto Detect” button that redirects the user to the auto detect page. The button is at the right top corner of the association list area.

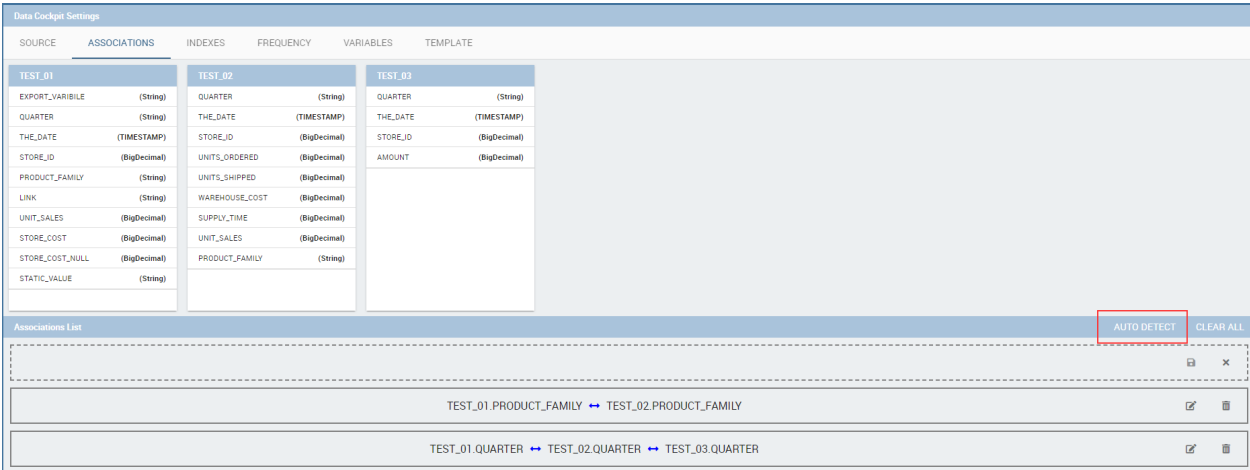


Fig. 1.214: Auto Detect button.

Here the user can decide a minimum percentage of match that Knowage will use to compute and return matching columns. For instance a 20% of similarity means that the values contained by two or more columns of different datasets are the same at least for the 20% of them. Following the example shown in Figure below, we can read that between the three dataset, only two of them have possible associations. The user can use a 100% similarity match using the “product_family” field, which means that the two columns contained in two different dataset return exactly the same list of distinct values. Or the user can set an association with a 75.23% similarity using “the_date” field which means that some values of one column are not contained in the other column. The minimum length allows the user to specify the lowest number of datasets to be joined.

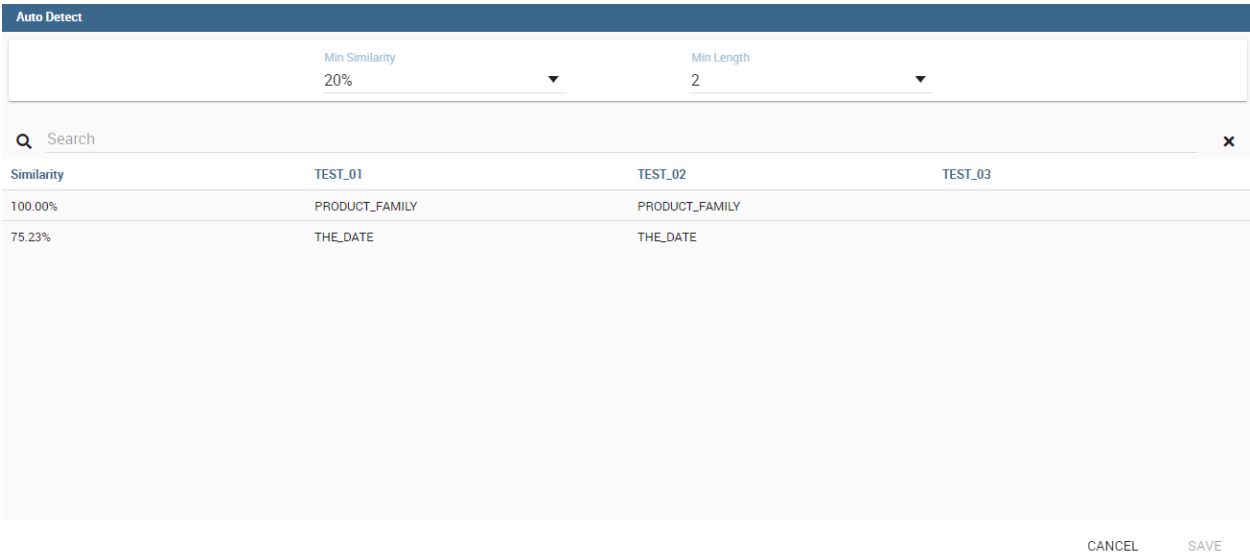


Fig. 1.215: Auto Detect page.

To select one of suggested associations, simply click on the specific row and then on the save button. Save the settings and save the dashboard. The new association is ready to be used.

1.4.3.3 Indexes

To have faster loading time of the cockpit, is possible to create indexes on cached data. This feature is available only for cached dataset.

If you want to create an index on a column you have to choose that column by clicking on it. The name of the column will appears in the Indexes List section of the page. If you want to confirm your choose, click on the save icon. If you want to cancel it, click on the cross icon. After saving a index you'll see in the list surrounded by a continuous border.

The screenshot shows the 'Data Cockpit Settings' interface with the 'INDEXES' tab selected. It displays two datasets: 'ALL_CUSTOMERS' and 'SPARKSQL_STORE'. The 'SPARKSQL_STORE' dataset has 'store_name' selected. Below the datasets, the 'Indexes list' shows two entries: 'SPARKSQL_STORE.store_name' and 'ALL_CUSTOMERS.customer_id'. The 'ALL_CUSTOMERS.customer_id' entry is highlighted with a continuous border. At the bottom right, there are 'CANCEL' and 'SAVE' buttons.

Fig. 1.216: Indexes settings example

For example, in the figure above index on the column “customer_id” of ALL_CUSTOMERS dataset is already saved. “store_name” column of “SPARKSQL_STORE” dataset is selected. If you want to create an index on it, you have to save it.

1.4.3.4 Frequency

The Frequency tab defines a scheduling over dataset involved in the associations. An example is supplied in the next figure. This means that associations are activated automatically and data are reloaded according to this feature. In particular, groups of realtime datasets that compose one or more associations can have different update frequencies. We stress that, in order to secure the right document execution, the group frequency do not affect the other ones and each group is reloaded at different times. In addition, realtime dataset that are not involved in any association can have their own frequency.

The screenshot shows the 'Data Cockpit Settings' interface with the 'FREQUENCY' tab selected. It displays a frequency setting of 5 for the 'TEST_01' dataset. The 'TEST_01' dataset is highlighted with a continuous border. At the bottom right, there are 'CANCEL' and 'SAVE' buttons.

Fig. 1.217: Frequency settings example.

1.4.3.5 Variables

In this tab the user can define the variables that will be available inside the cockpit.

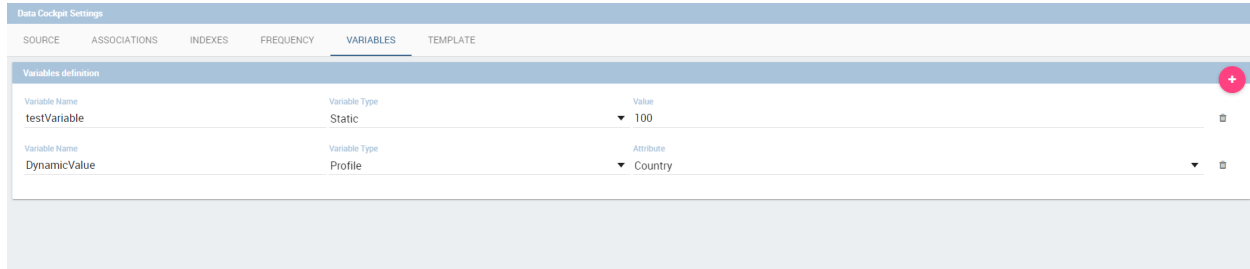


Fig. 1.218: Variables tab

Every variable needs a name that will be used to call it and get the value, a type and a value. The available types are the following:

- *Static*: a static number or string.
- *Dataset*: the value of a selected dataset column. If a column is not selected a set of variables key/value will be created using the first two columns of the selected dataset.
- *Profile*: a set of profile attributes available. (ie. the username)
- *Driver*: the value of a selected analytical driver.

The variables will be available inside the widgets with the `$V{variablename}` format.

1.4.3.6 Template

In this tab the user can find the json code (at the current stage of the work) which composes the template. Figure below shows an example.

1.4.4 Selections

Is possible to keep track of the **Selections** applied to your widgets, namely the possibility to filter widget data according to selection made through the click on a specific item of the cockpit (cell value, chart bar, etc.), also from the cockpit menu. You can check which selections are active on your cockpit at anytime thanks to the *Selections list* functionality. Previously, we already described how to add the “*Active Selections*” widget inside the cockpit area. If the user don’t want to add a widget that stay visible inside the cockpit, selections can still be accessed and managed through the cockpit menu by clicking on the “*Selections*” icon (see the figure).

Here all selections and associations are listed, as shown in Figure below. The “Delete” button is available just aside each row to let the user to remove that specific selections. Click on the “Cancel” button to exit the window.

1.4.5 Clear cache

The **Clear cache** button lets you realign the data shown in your widget to the ones in your database. When you create your widget and associate your datafields, a photo of data is made and stored in temporary tables. This means that your cockpit will display the same data at each execution until you clean the cache, by clicking on the dedicated button, and execute the document again. Now your data are refreshed and updated to the one contained in your database at last execution time. As discussed before this button is available also in “View mode” modality for end users.

Data Cockpit Settings

SOURCE ASSOCIATIONS INDEXES FREQUENCY VARIABLES **TEMPLATE**

☒ Smart Visualization

```

Object
└─ sheets: Array [1]
   └─ 0: Object
      label: "New Sheet"
      └─ widgets: Array [2]
         index: 0
         $$hashKey: "object:36"
      └─ configuration: Object
         showMenuOnView: true
         showSelectionButton: true
         └─ style: Object
            titles: false
            └─ title: Object
               borders: true
               └─ border: Object
                  shadows: true
                  └─ shadow: Object
                     datasets: Array [2]
                        └─ 0: Object
                           └─ 1: Object
                              documents: Array [0]
                                 []
                              associations: Array [1]
                                 └─ 0: Object
                              aggregations: Array [1]
                                 └─ 0: Object
                              filters: Object
                                 No properties
                              variables: Array [0]
                                 []
                              showScreenshot: true
                              showExcelExport: true
                              indexes: Array [0]
                                 []
                              aliases: Array [0]
                                 []
            knowageVersion: "8.1.1-S"

```

Fig. 1.219: Template example.

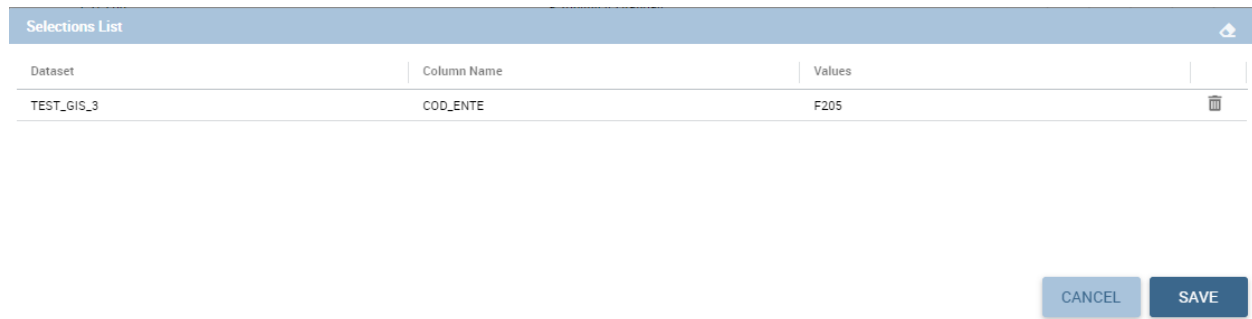


Fig. 1.220: Selections icon from cockpit menu.

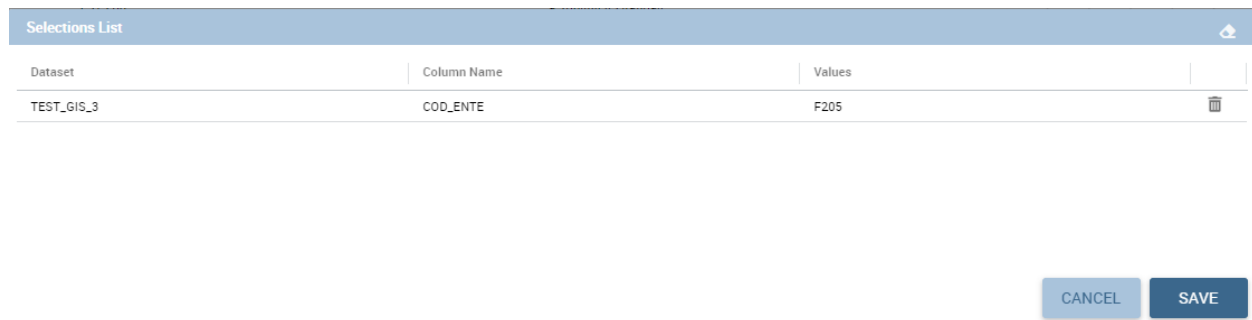


Fig. 1.221: Selection window.

1.4.6 Save

You can save the cockpit by clicking on the save button in the right-top corner. The document will be saved in the personal folder (technical users) or in the **My Analysis** section. We remember that it is possible to share the new cockpit with other users clicking on the dedicated icon. You can also choose in which folder, among the ones visible to your role, to place your shared document.

1.4.7 Multisheet functionality

The Cockpit engine allows to manage contents in multiple sheets. In each sheet the user can find and employ the features shown above. Indeed, it is possible to perform a new analysis (as highlighted in Figure below) selecting different datasets and gadgets. The multisheet functionality is particularly useful to focus the analysis in a single spot and have a general overview over it in few clicks at the same time.

A user can also take advantage of the “move widget” functionality to bring a widget from one sheet to another. Furthermore it is possible, but not mandatory, to set associations between datasets underlying different sheets.

1.4.8 Export cockpit

Cockpit document allows to export data into csv file without executing document. This is very useful when you produce data using a heavy query. This option is available if your document has parameters. When you start execution of your document, you will get opened filter panel so you can fill values. To start export, you should click on drop down menu, next to execute button, as on image below.

After process is finished, you will get notification on **download** icon  and you can find your file in **Download manager** section.

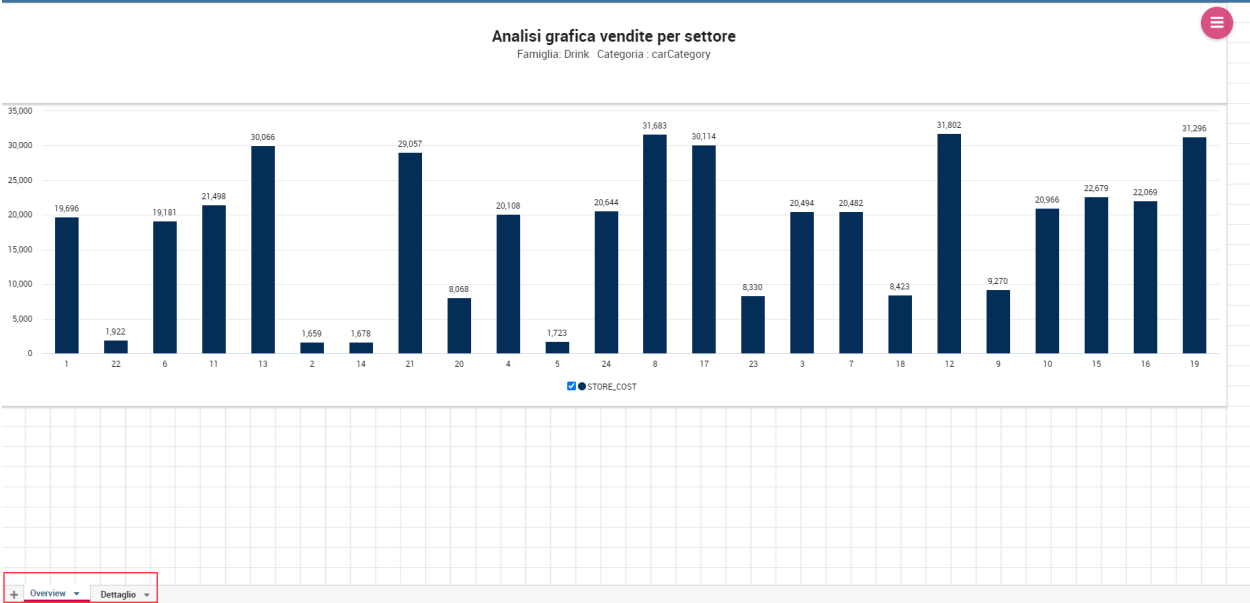


Fig. 1.222: Multisheet cockpit example.

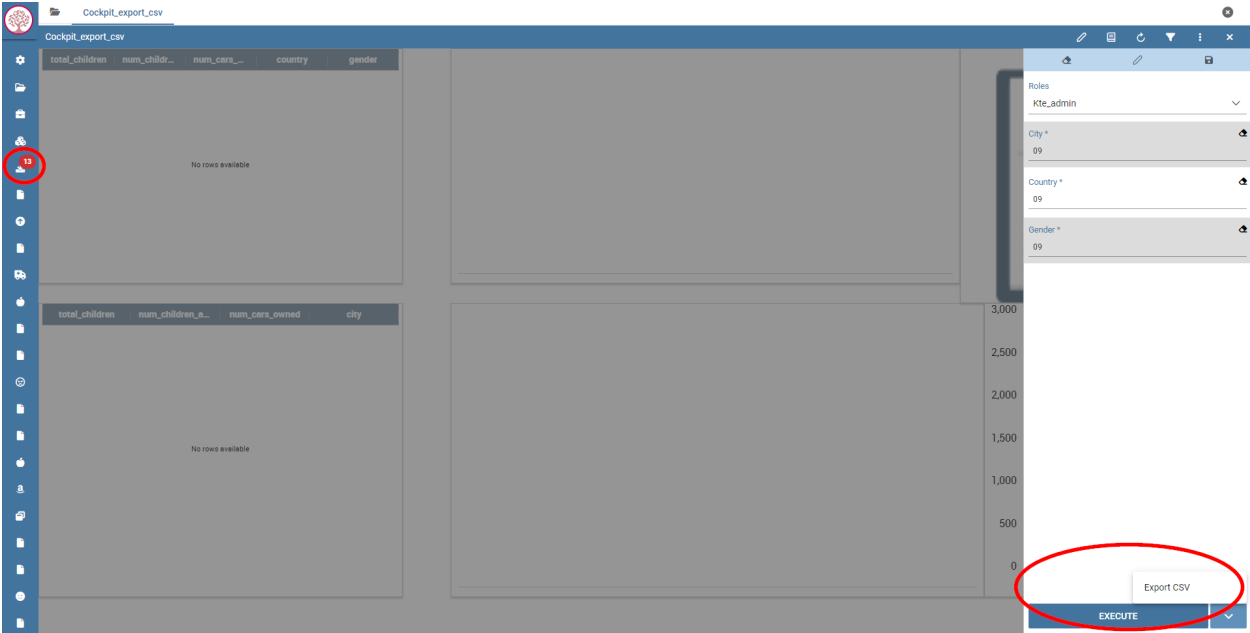


Fig. 1.223: Export cockpit into csv.

Clicking on download icon, **Download manager** will open, and you will be able to download zip file that contains csv file/files, depends of how many widgets (chart or table) you have in your document.

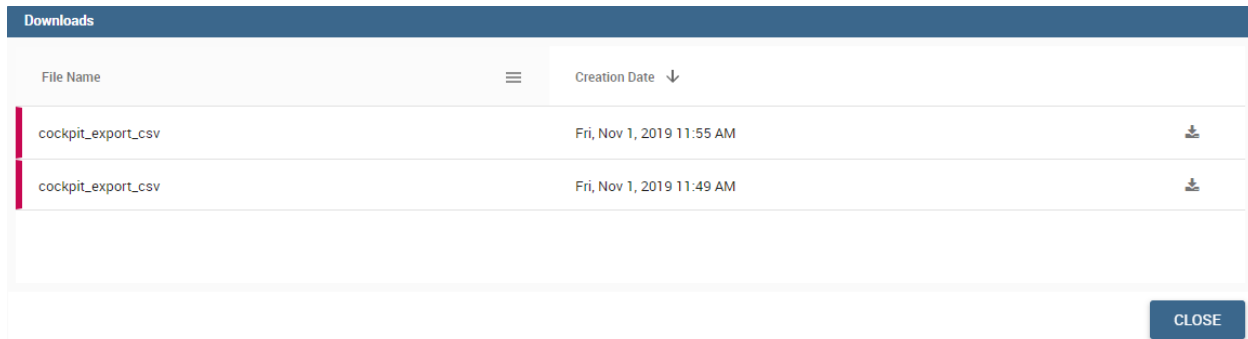


Fig. 1.224: Download manager section.

1.5 Create Report

1.5.1 Create a Birt report

Reports represent the most popular type of business analysis. Indeed, they allow the visualization of data in a structured way and accordingly to predefined formats. The main characteristics of a report are:

- combination of numerical data (tables, lists, cross tables), charts and images;
- static and pixel-perfect layout;
- multi-page and multi-format output (PDF, HTML, DOC, etc.);
- organization of data by groups and sections;
- universal usage (summary, detail, analytical or operational);
- being suitable for off-line production and distribution (scheduled execution);
- ease of use.

For these reasons reports usually have a pervasive level of usage: they are used by developers to perform both synthetic and detailed analysis, having a particularly low level of difficulty.

BIRT, acronym for **Business Intelligence and Reporting Tools**, is an open source technology platform used to create data visualizations and reports. In Figure below you can see an example of BIRT report.

1.5.1.1 Developing a BIRT report

Firt of all you have to download the Birt report designer from here [Download Eclipse](#).

Here you can find some information about how install and use Birt report designer: [How install and use birt](#)

After that you can use the designer to create a new report.

- 1) create a new **Report Project**
- 2) create a new **Report**
- 3) Configure che data source, create the data set and design the report

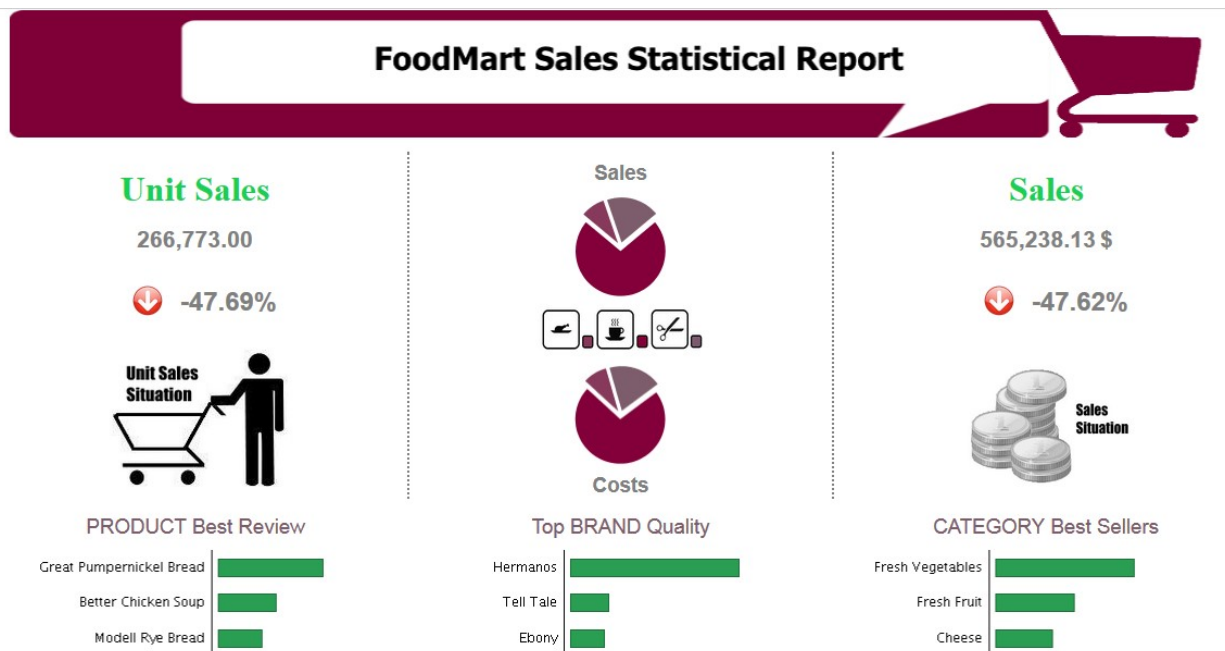


Fig. 1.225: Example of a BIRT report.

Using the preview function (under the *Run* menu) you can test the report.

Once the document is designed, it is stored as a local file, marked out with an icon and a specific file extension:

- **.rptdesign**: document template for reports that use the BIRT engine.

4) Deploy on KNOWAGE server: using the KNOWAGE document browser you can install the report in the Server and test it.

In this specific example, we will show how to create a report with an internal dataset. First of all, in case of an internal dataset, define a **JDBC Data Source**.

Right click on the **Data Source** item and select the corresponding data source. A pop up editor will open, prompting you the connection settings:

- **Driver class**
- **Database URL**
- **Username and password**

Note that these configuration parameters will be used by the birt designer to connect to the database and let the report to be executed locally. Make sure that the database set in the Knowage server share the same *schema* of that defined in the designer.

Since you are setting a local reference to a database inside the report, you need to add an additional information to enable Knowage Server to correctly execute the report by connecting to the data source referenced within the server and not inside the report. Basically, you need to tell the server to override the data source configuration of the report. Therefore, add a parameter to the report, called **connectionName**, by right-clicking on the “Report Parameters” menu item and selecting “New Parameter”. Fill in the form as suggested below.

Then go to **Property Binding** in the Data Source editor and set the property JNDI URL to the value of the **connectionName** parameter, as shown below.

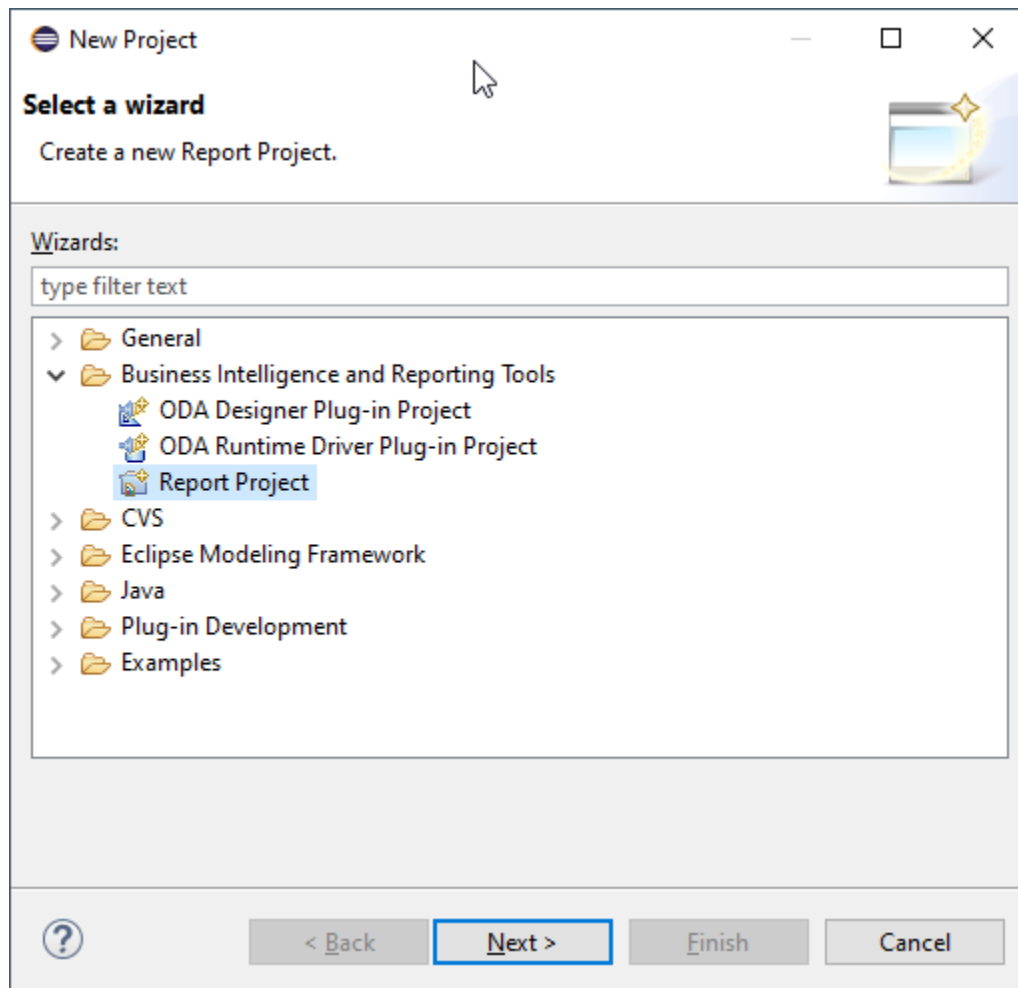


Fig. 1.226: Create a new Report Project.

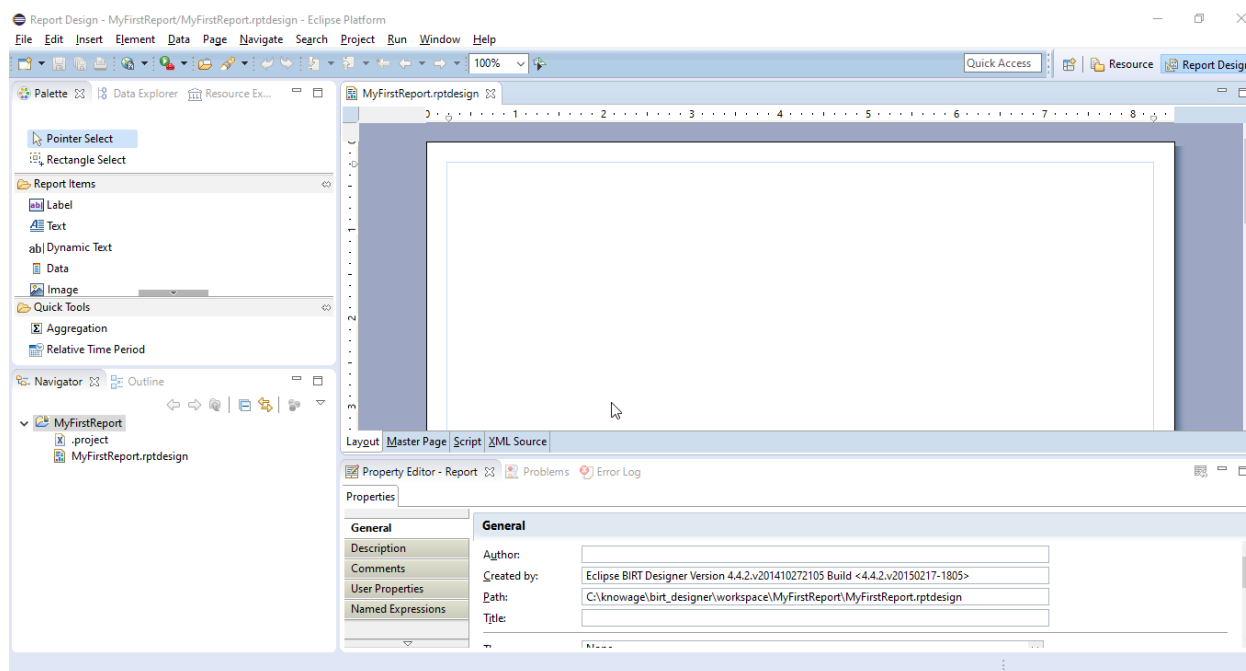


Fig. 1.227: Create a new Report.

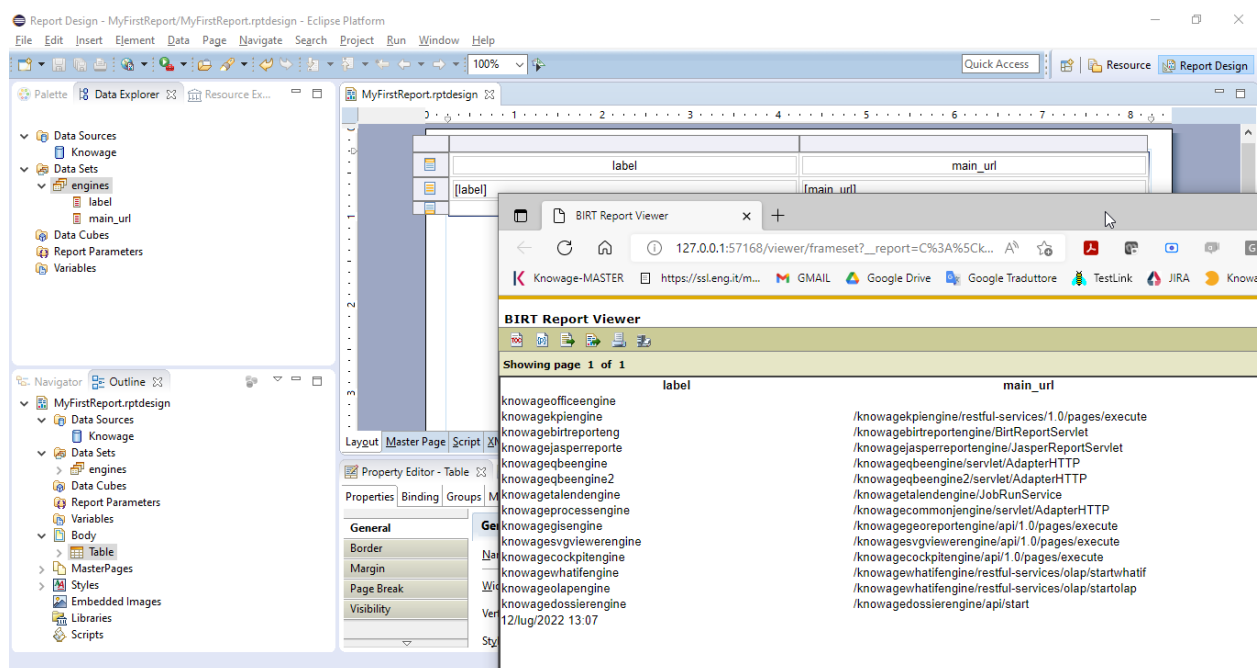


Fig. 1.228: Design the report

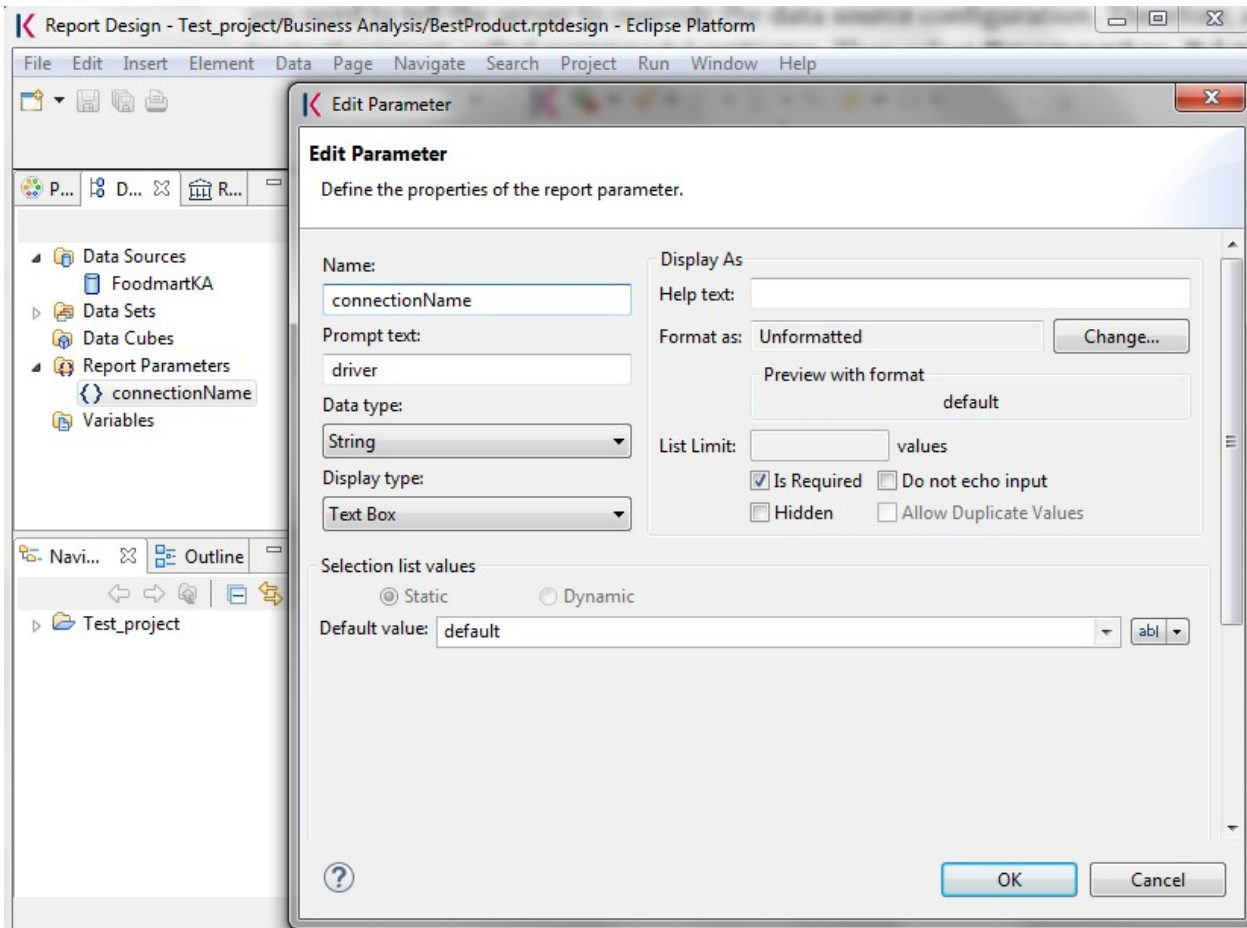


Fig. 1.229: Adding connectionName Parameter.

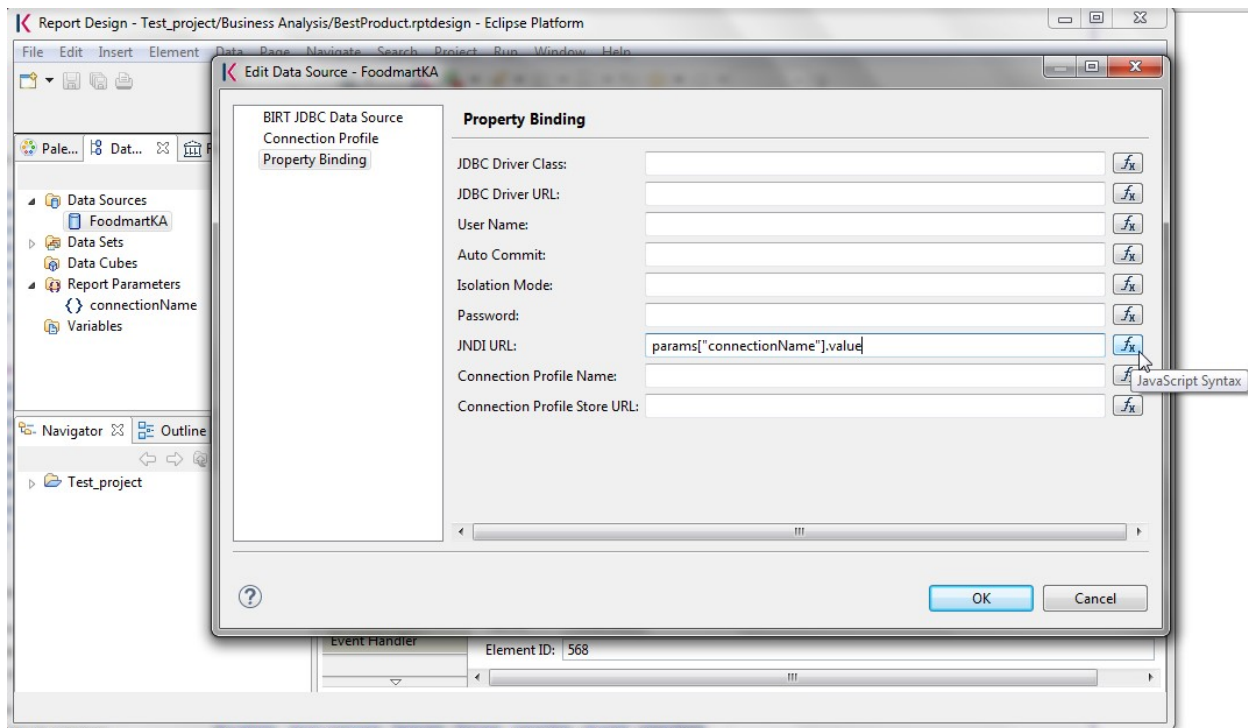


Fig. 1.230: Setting the connectionName parameter in the Data Source editor

Warning: JNDI URL

Do not forget to define the connectionName parameter in your BIRT report and set the JNDI URL accordingly. Without these settings your BIRT report may be unable to access data once it is deployed on the server. In addition, if database and connection properties change, you need to change the connection properties only in Knowage server.

Once the data source has been configured, you can proceed with the creation of a dataset. Therefore, right-click on the **Data Set** item and select **New Data Set**. In the next window, select the data source, the type of query and give a name to the dataset, as exhibited below. The scope of this name is limited to your report, because we are defining an internal dataset.

Now you can define your dataset by writing the SQL query in the editor and testing the results (see figure below). At any time, you can modify the dataset by clicking on it, which will re-open the query editor.

Let us design a very simple report, which contains a table showing the data from the defined dataset. The easiest way to create a table from a dataset is to drag & drop the dataset from the tree menu into the editor area.

The most generic way, which applies to all graphical elements, consists in switching to the **Palette** menu on the left panel, keeping the designer in the central panel. Drag and drop the table into the editor area. Consider that this can be done with all other elements listed in the Palette. At this point, you can edit the table (as well as any other graphical element on the report) using the **Property Editor** tab below the editor area.

While developing a report, it is particularly useful to test it regularly. To this end, click on the **Run** menu on top of the screen and select **Run as HTML**. To revert back to the editor, just click on the **Layout** tab. In the **Master Page** tab, you can set the dimensions and layout of the report; the **Script** tab supports advanced scripting functionalities; finally, the **XML Source** tab shows the editable source code of your report.

Once your report is done, you can deploy it on Knowage Server.

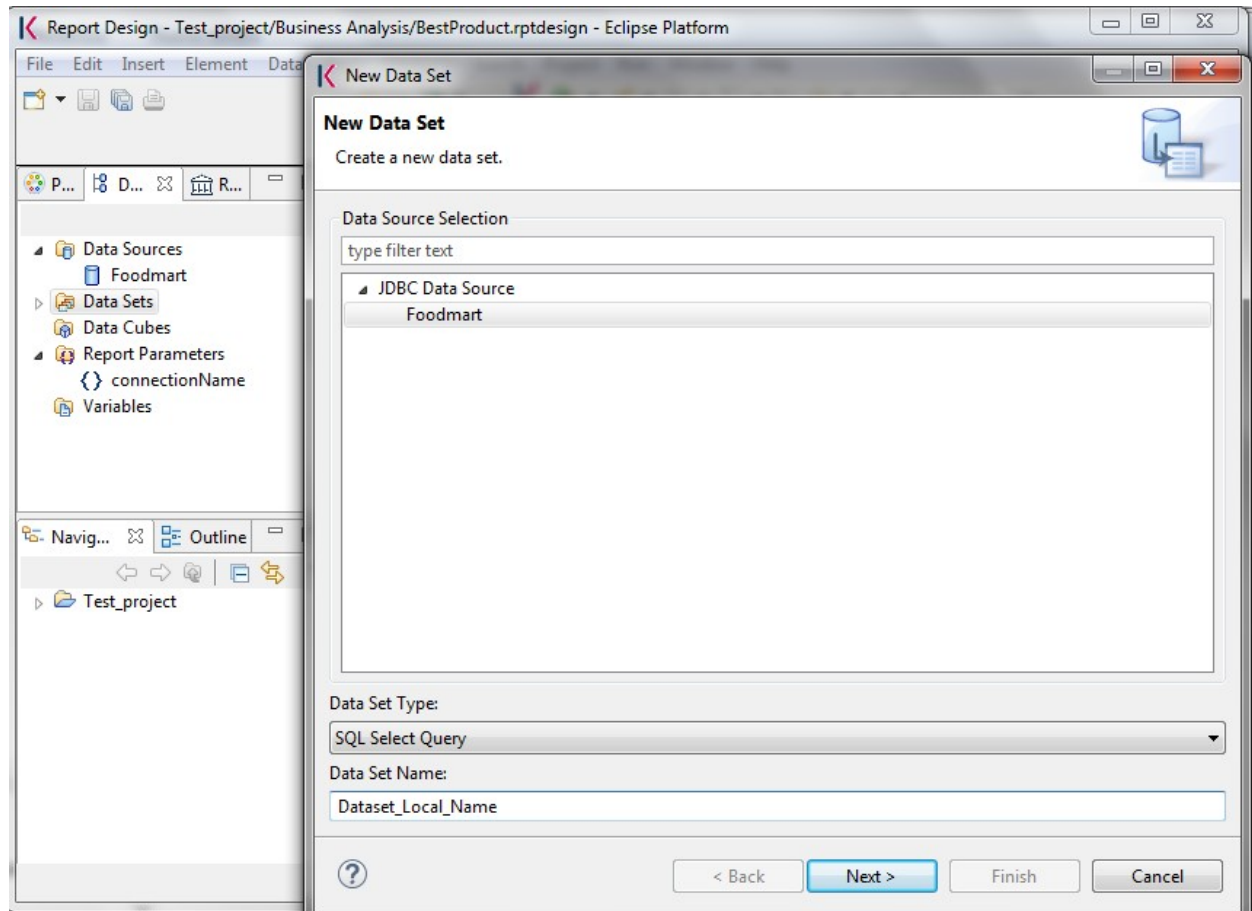


Fig. 1.231: Dataset definition.

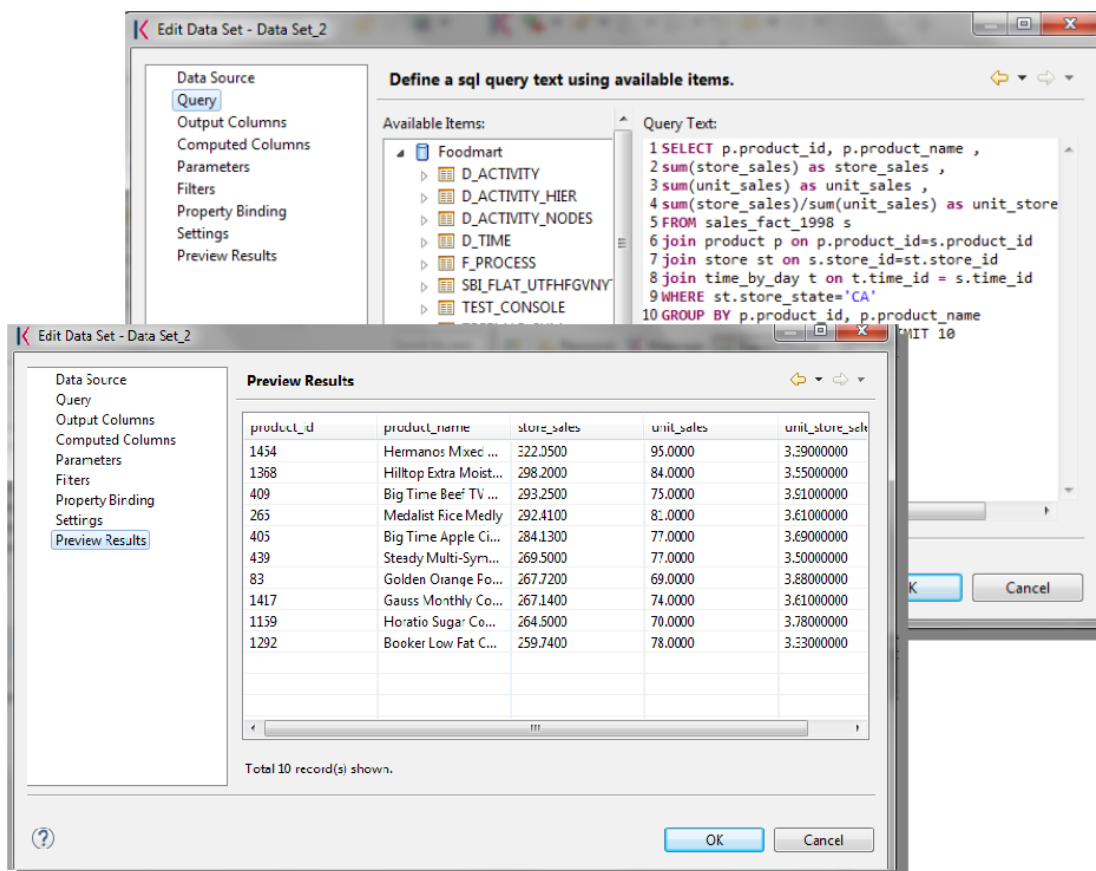


Fig. 1.232: Dataset editor, with preview.

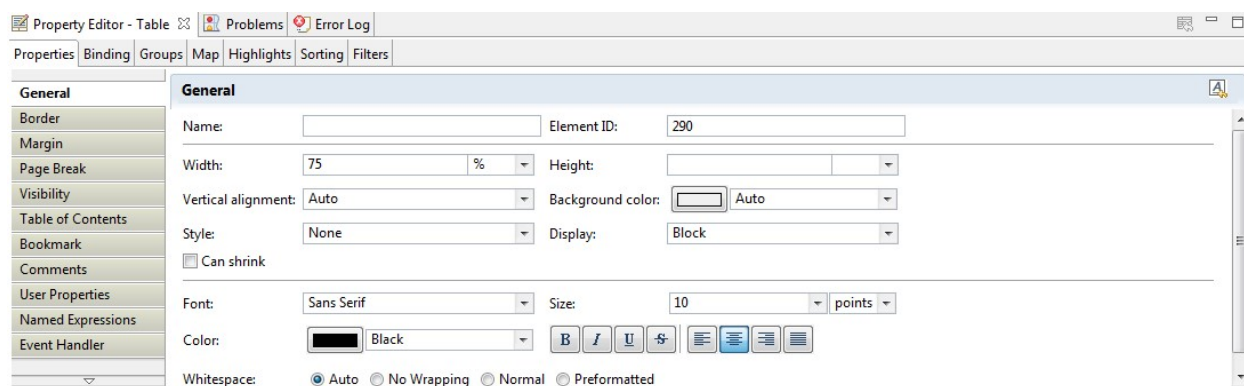


Fig. 1.233: BIRT Property Editor.

Note: Deploy on Knowage Server

To deploy the report you have to create a new *Generic document* and then upload the template file directly in the document detail page.

The BIRT report designer allows the creation of complex reports, with different graphical elements such as cross tabs, charts, images and different text areas. In this section we do not provide any details on graphical development but we focus on specific aspects of Knowage BIRT Report Engine.

Note: BIRT Designer

For a detailed explanation of report design, please refer to BIRT documentation at <https://www.eclipse.org/birt/>.

1.5.1.1.1 Adding parameters to reports

Most times reports show data analysis that depend on variable parameters, such as time, place, type. Birt designer allows to add parameters to a report and link them to analytical drivers defined in Knowage Server.

To use these parameters, you first need to add them to your report. Right-click on **Report Parameters** in the tree panel and select **New Parameter**. Here you can set the data type and choose a name for your parameter.

Warning: Parameters URL

Be careful when assigning a name to a parameter inside a report. This name must correspond to the parameters URL when you deploy the document on Knowage Server.

Once you have defined all parameters, open the (or create a new) dataset. Parameters are identified by a question mark ? . For each ? that you insert in your query, you must set the corresponding link in the **Parameters** tab: this will allow parameters substitution at report execution time.

Note that you must set a link for each question mark as shown below, even if the same parameter occurs multiple times in the same query.

Warning: Transfer reports from Birt designer to Server and vice versa

Any valid BIRT template can be directly uploaded in Knowage Server using the web interface for document management.

Parameters can also be used within some graphical elements, such as dynamic text, with the following syntax:

Listing 1.14: Parameters syntax

```
params[name_of_parameter].value
```

1.5.1.2 Cross Navigation for BIRT Reports

A powerful feature of Knowage analytical documents is cross-navigation, i.e., the ability to navigate documents in a browser-like fashion following logical data flows. Although crossnavigation is uniformly provided on all documents executed in Knowage Server, each type of document has its own modality to set the link pointing to another document.

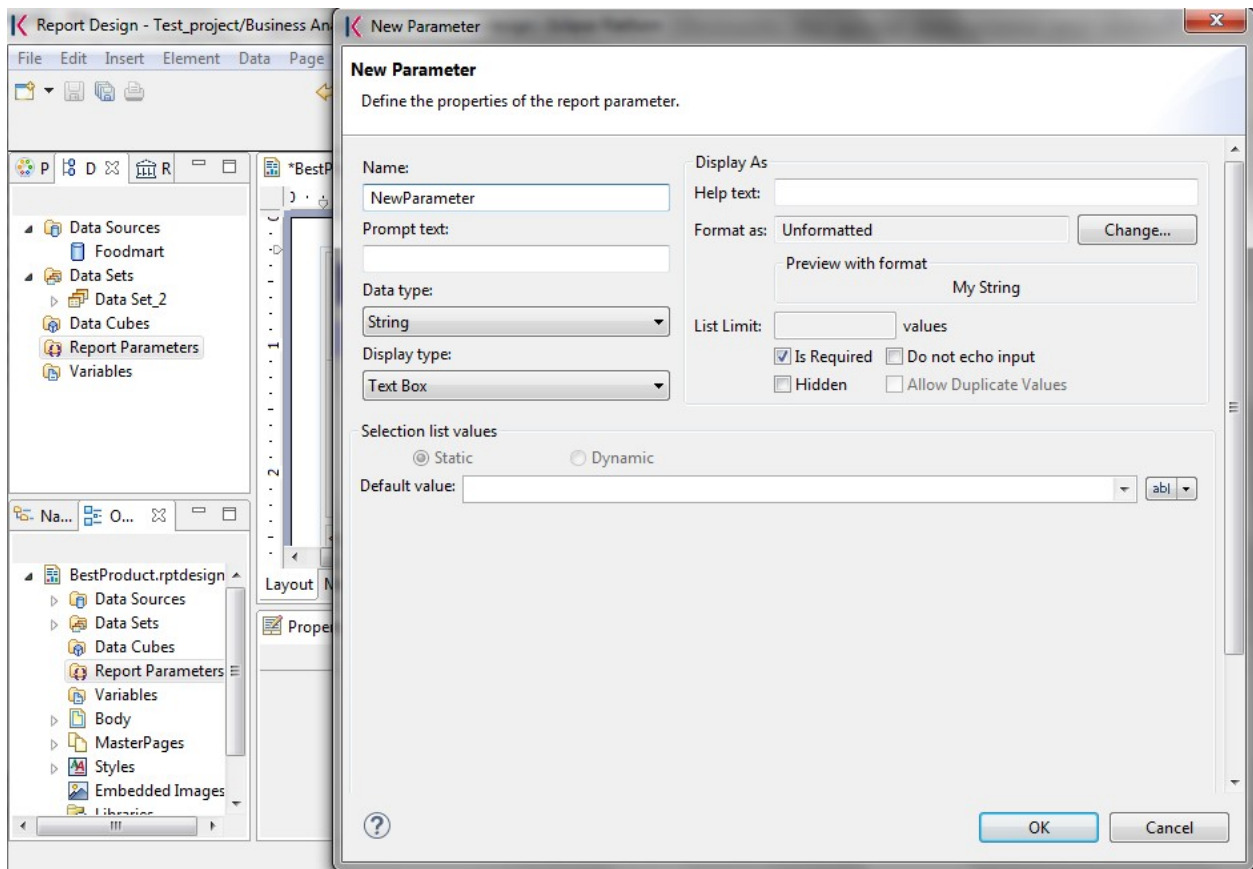


Fig. 1.234: Creation of a new parameter in a BIRT report.

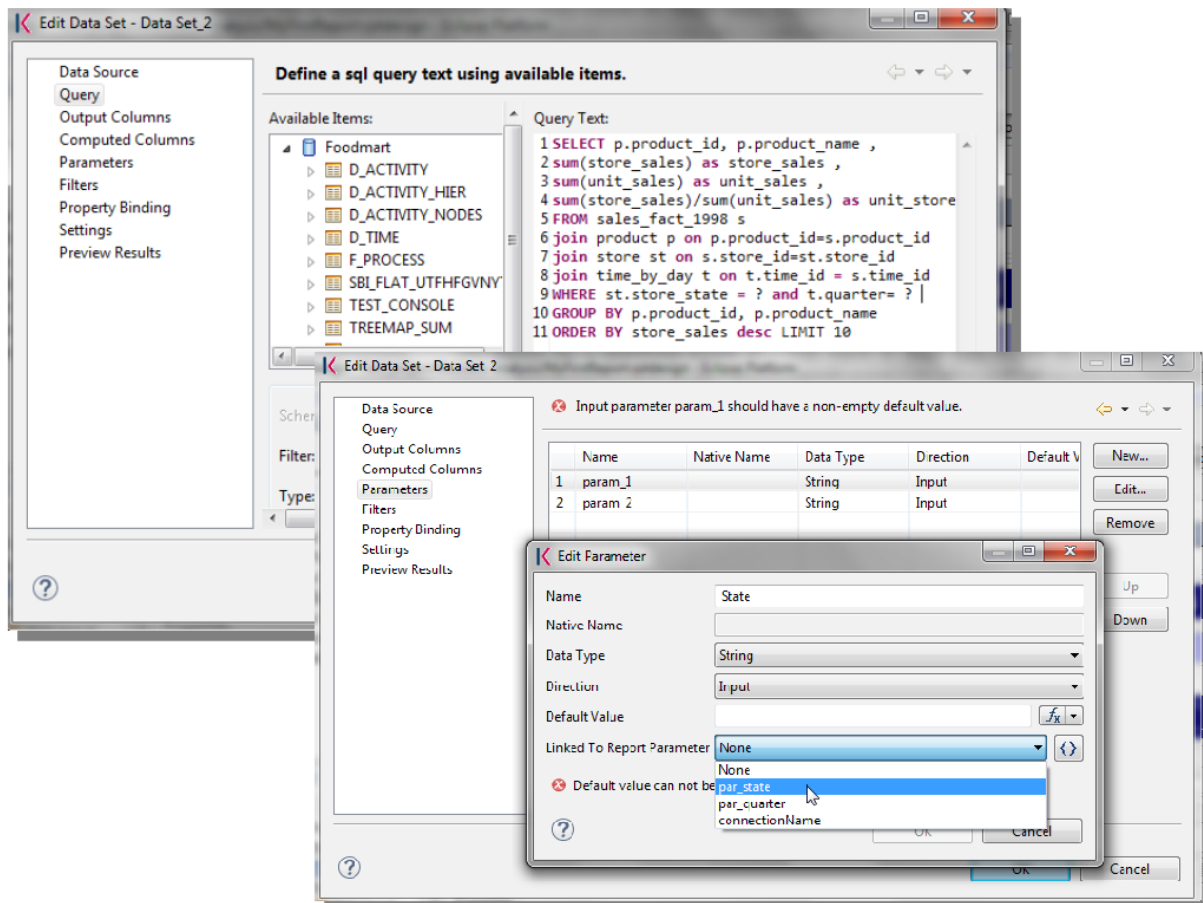


Fig. 1.235: Insert parameters into the dataset definition.

Notice that the cross navigation can reference any Knowage document, regardless of the type of the source document. For example, a BIRT report can point to another BIRT report, to a dashboard, a geo or any other analytical document.

It is obviously possible to associate more than one cross navigation to a single document. It means that by clicking on different elements of the same document the user can be directed to different documents.

To allow the cross-navigation in a BIRT report, you need to add a hyperlink to the element you want to be clickable using the **Properties** tab of the designer. Most report elements can host a hyperlink. For example, let us add a hyperlink to a cell in the table.

Click on the table cell and select the **Hyperlink** item in the **Properties** tab. By clicking on Edit, the hyperlink editor will open and show three input fields:

- **Location:** write here the URI,
- **Target:** select Self,
- **Tool Tip:** write the text you wish to appear on the link, as showed in the following Figure below.

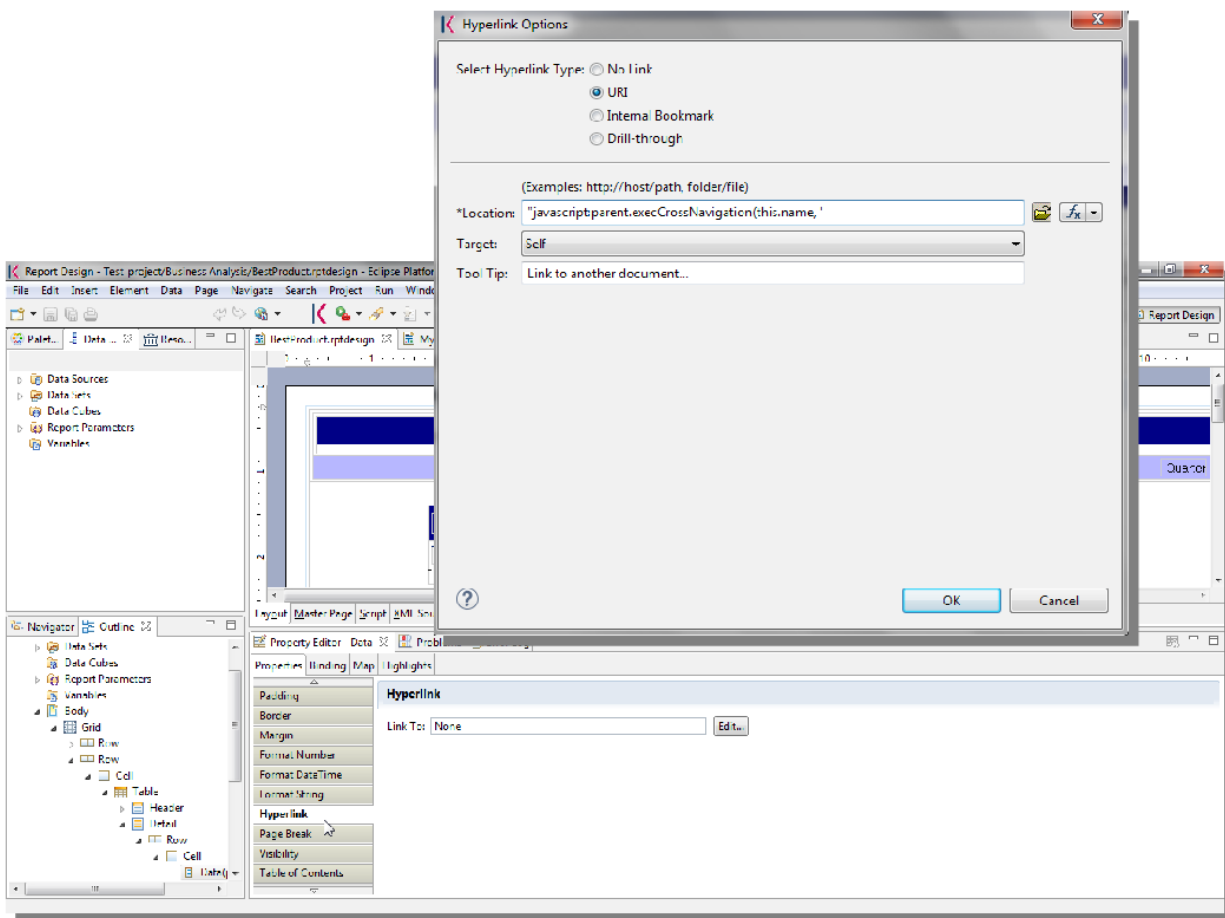


Fig. 1.236: Hyperlink editor.

To edit the Location, click on the right drop down button and select the JavaScript syntax. This will open BIRT JavaScript editor. Here you must write down the javascript function `javascript:parent.execExternalCrossNavigation` passing JSON arguments like `{ParName1: string}, null` and `Cross_Navigation_Name: string`.

In Cross Navigation syntax we give an idea of how the syntax should be like:

Listing 1.15: Cross Navigation syntax.

```
1 "javascript:parent.execExternalCrossNavigation("+
2 "{OUT_PAR:''+params["par_period"].value+''"+
3 ",OUT_STRING:''+string_text+''"+
4 ",OUT_NUM:''+numberX+
5 ",OUT_ManualSTRING:'foo'"+
6 ",OUT_ARRAY:['A','B','5']}"
7 ",null,"
8 "'Cross_Navigation_Name');"
```

Warning: Type the right cross navigation name

It is important to underline that the "Cross_Navigation_Name" of Cross Navigation syntax is the cross navigation name related to the document and set using the “Cross Navigation Definition” feature we described in *Analytical Document* Chapter, *Cross Navigation* Section.

It will be necessary to type the right cross navigation name related to the document as defined using the “Cross Navigation definition” feature of Knowage server and to define those parameters (OUT_PAR, OUT_STRING, etc.) as output parameters in the deployed document on the Server (see *Analytical Document* Chapter, *Cross Navigation* Section).

Note that the syntax of the string is fixed, while you need to assign values to the parameters that will be passed to the destination document. The JavaScript editor helps you to insert dataset column bindings, as shown in Figure below, and report parameters automatically.

To manage multi-value parameters is enough to list all values between brackets separating them with commas, as reported in the code above. More specifically, the array must contain values of the same type. For example:

```
1 OUT_SeveralNames: ['Michael', 'Paul', 'Sophia']
```

or

```
1 OUT_SeveralValues: [5,9,31938]
```

Finally, it is possible to set a sort of “multi”-cross navigation if for example the source document is related to more than one document through the Cross Navigation Definition. Let suppose that the source document is linked to a target document with the navigation called “CrossNav1” and simultaneously the source document is linked to a second target document with the navigation called “CrossNav2”. If in the JavaScript function of *Cross Navigation syntax* code the “Cross_Navigation_Name” is left empty, as in the code below, when the user clicks on the object for which the navigation has been enabled a pop up opens asking to the user to choose between the “CrossNav1” navigation or the “CrossNav2” one. This procedure allows the user to have more than one possible navigation starting from the same object.

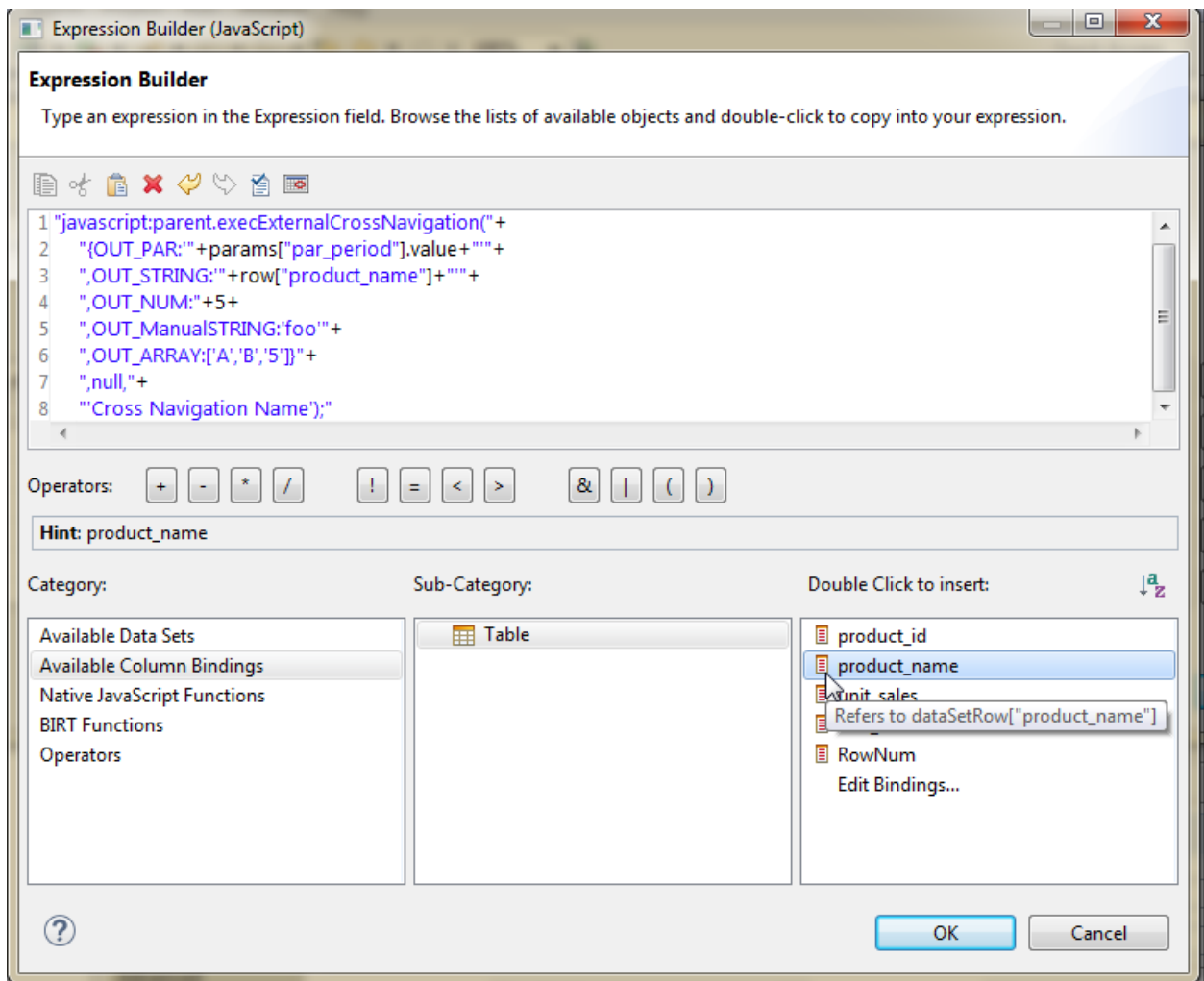


Fig. 1.237: Column bindings.

Listing 1.16: Cross Navigation syntax

```

1  "javascript:parent.execExternalCrossNavigation("+
2  "{OUT_PAR:''+params["par_period"].value+''"+
3  ",OUT_STRING:''+string_text+''"+
4  ",OUT_NUM:''+numberX+
5  ",OUT_ManualSTRING:'foo'+
6  ",OUT_ARRAY:['A','B','5']}"+"
7  ",null,"+
8  "''');"

```

1.5.2 Jasper reporting

Jasper is a stand-alone reporting tool developed by the Jaspersoft Community. An example of Jasper report is represented in the next figure. Jasper Report Engine manages Jasper report templates inside Knowage Server. A report template for Jasper is a text file with .jrxml extension that can be manually modified by very expert users by editing XML code. Otherwise, *iReport*, a graphical template designer, is provided for all developers who want to easily design a report for this engine.

Salary of VP Country Manager

Salary VP Country Manager						
	Name		Gender	Store Name	City	State
40000.00	Derrick Whelply	M	HQ	Alameda	CA	
	Michael Spence	M	HQ	Alameda	CA	
35000.00	Laurie Borges	F	HQ	Alameda	CA	
	Maya Gutierrez	F	HQ	Alameda	CA	
	Pedro Castillo	M	HQ	Alameda	CA	
30000.00	Beverly Baker	F	HQ	Alameda	CA	

Fig. 1.238: Example of a Jasper report.

Please note that this engine is available only in KnowageER.

1.5.2.1 Document definition*

Unlike the BIRT designer, iReport is not integrated in Eclipse. Therefore, before starting developing the report, you must download it from the website <http://community.jaspersoft.com/project/ireport-designer/releases>.

Note: iReport download

Download the iReport designer at <http://community.jaspersoft.com/project/ireport-designer/releases>.
Knowage support any kind of version.

Then you will have to set the path to the iReport designer in Knowage Studio. Open Window > Preferences > iReport Editor Configuration and set the correct path to the ireport.exe file.

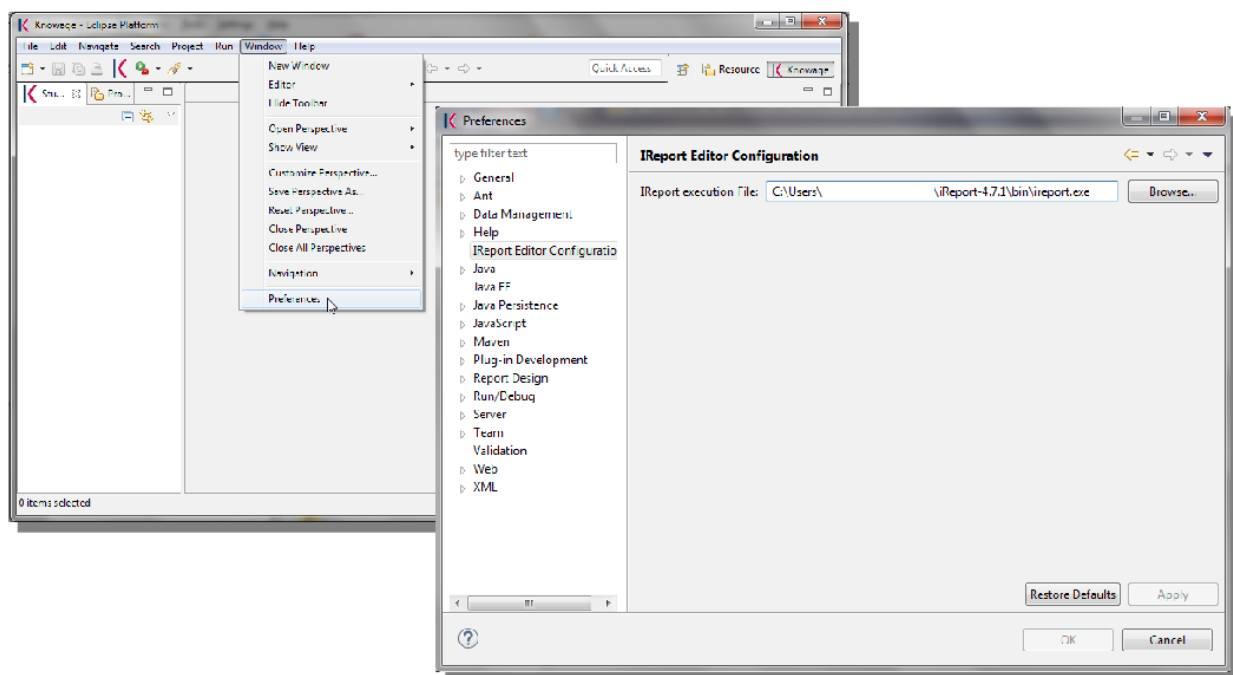


Fig. 1.239: iReport Configuration in Knowage Studio.


Similarly to the case of a BIRT report, the design and deployment of a Jasper report with Knowage Studio consists of the following steps:

- create the empty document,
- switch to the report designer perspective,
- create the data source,
- create the dataset,
- design the report via the graphical interface,
- deploy the report on the server.

To create a new Jasper report, right-click on the **Business Analysis** folder and select **Report > Report with Jasper**. This will open an editor where you can choose a name for your document. The new document will be created under the Business Analysis folder.

Double click on the report to open the editor: the iReport editor will be launched inside Knowage Studio. Now you are ready to start to develop your report.

The next steps consist in the creation of a datasource and of a dataset. As described in section “Dataset definition”, Knowage Studio allows the development of analytical documents using either internal or external datasets. In this example we will show how to create a report with an internal dataset.

Click on the small icon  of the menu bar. The data source creation editor will open.

Select a JDBC data source, set the appropriate parameters and give the data source a name. Figure below shows the wizards that present in this procedure.

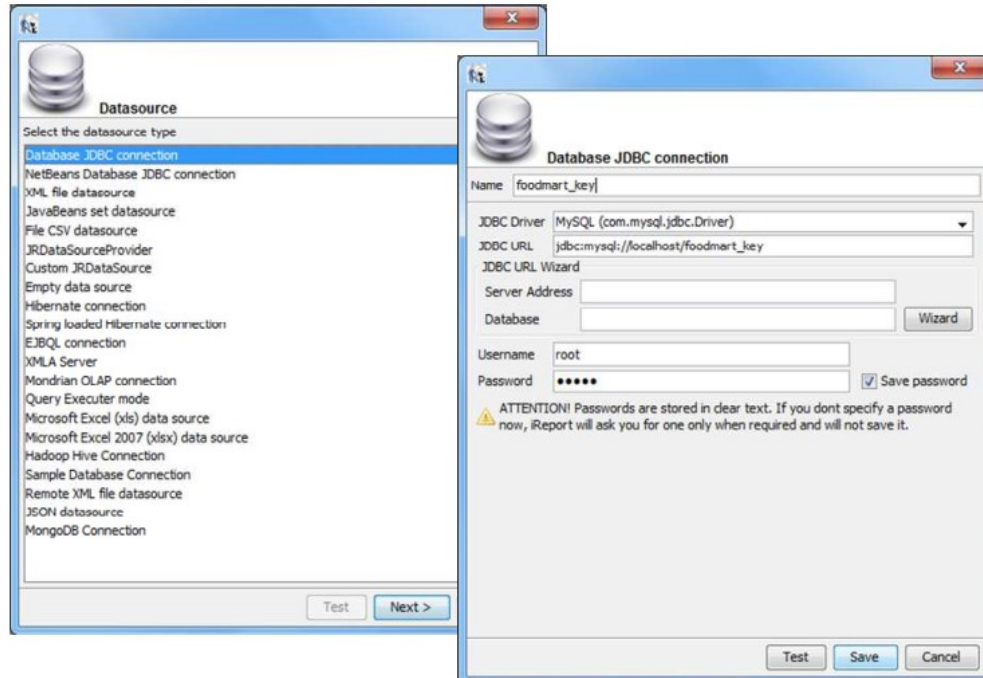



Fig. 1.240: Creation of a JDBC data source in a Jasper report.

Once you have defined the data source, create your dataset. Note that Jasper only allows one main dataset. By the way, secondary datasets can be added if needed. Right-click on the report item and select **Add dataset**. The dataset editor

will guide you through the dataset definition. You can either manually edit the SQL query or use the design query tool provided by iReport.

The tree located in the left part of the window shows the elements of the report. On the right, the **Palette** shows all graphical elements you can add. You can choose to see your report within the **Designer**, to inspect the XML code or

to look at the **Preview** of your report. If you click on the  icon you can edit and preview your query. As you can see, the iReport designer allows the creation of complex reports, with different graphical elements such as cross tabs, charts, images and different text areas. We see now shortly how to design a very simple report, i.e. a report containing a table showing data from the defined dataset.

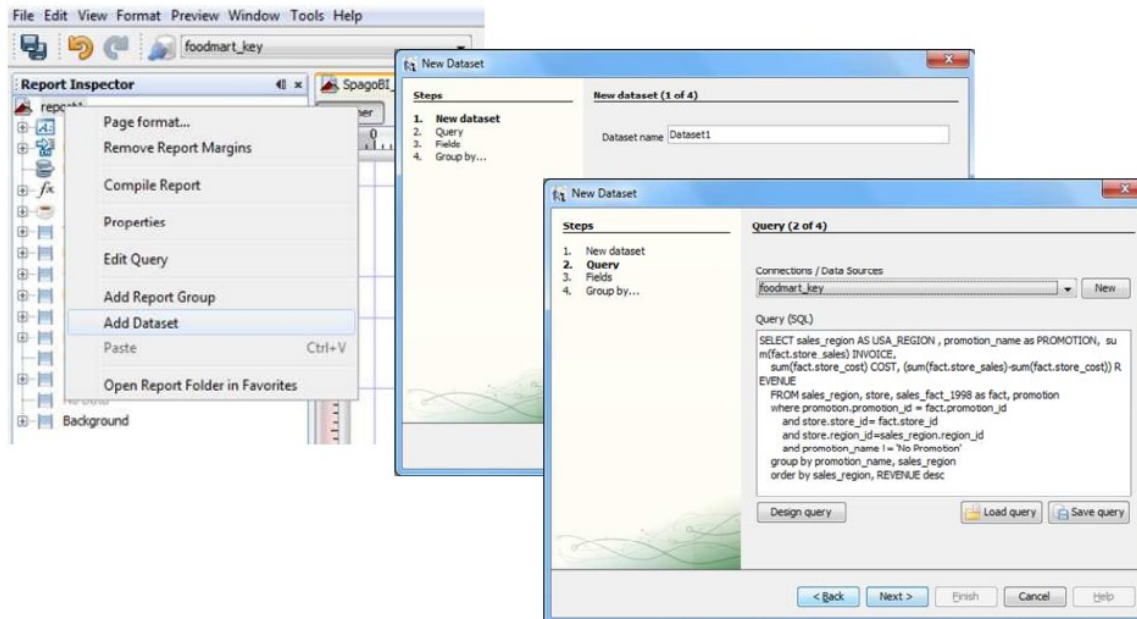


Fig. 1.241: Data Set definition.

To create a table, click on the element in the Palette and use the wizard. Figure below exhibits the Jasper editor interface.

To insert dataset columns into the report, use the syntax showed in following syntax to insert dataset columns.

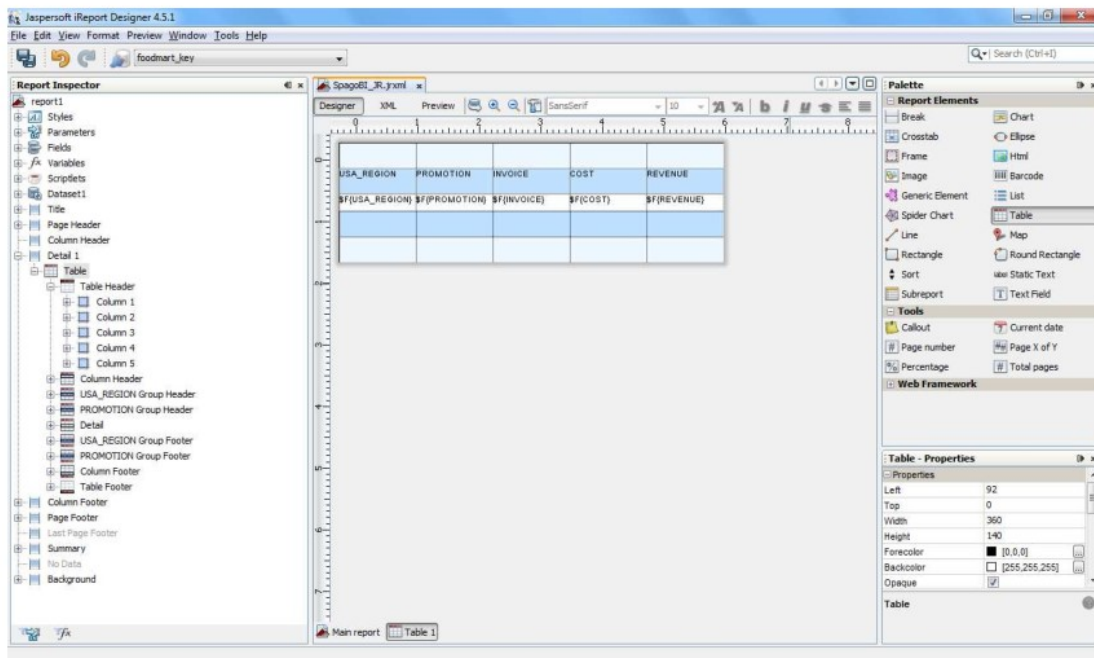


Fig. 1.242: iReport graphical editor.

Listing 1.17: Syntax to insert dataset columns.

```
$F{name_of_dataset_column}
```

Once the document has been developed, technical users can use Knowage Studio deployment service to easily register the report with its template on Knowage Server. Alternatively, any valid Jasper template (developed with or without Knowage Studio) can be directly uploaded to Knowage Server using the web interface for the document management.

This section does not provide any further detail about graphical development since it focuses on specific aspects of Knowage Jasper Report Engine. All Jasper standard functionalities work with Jasper Report Engine. For a full overview of Jasper reporting tool and a detailed developer guide, please refer to the official documentation at <http://community.jaspersoft.com/>.

1.6 Create Free Inquiry

1.6.1 How to build a Metamodel

In this chapter we go into details of how to build your own metamodel. Knowage allows to develop and manage metadata through the use of a web interface that is called **Meta Web**. We recall that dealing with metadata means to manage data that describe the structure of a relational model, namely to deal with the relationship between tables, table columns, keys and so on.

The Meta Web allows the user to access these information through the usage of a graphic interface and to easily combine, redefine and query them on an abstract model, guaranteeing the safety of the source model. In addition, we stress that the users can perform queries over data without the usage of a data query language.

1.6.1.1 Metamodel creation

Using the Meta Web application, it is possible to reverse the content of a database and manipulate this information creating a new model that can fit the user's needs. In this section we will see what are the steps needed in order to create a metamodel and query it with the QBE.

To create a Metamodel enter the **Business Model Catalogue** and add a new model clicking on the “Plus” icon. Referring to next figure, you will be prompted to enter the following fields:

- *Name* (mandatory): Name of the model (cannot be changed after the save).
- *Description*: A longer description of your model.
- *Category* (mandatory): Select, from the ones available, a category that the model belongs to.
- *Data Source* (mandatory): select the datasource that will be used to create your model (so the one that contains the tables that you need).

The screenshot shows a web form titled 'Details' for creating a metamodel. It contains the following fields and controls:

- Name ***: A text input field containing 'MyModel'.
- Description**: A text input field containing 'a description'.
- Category ***: A dropdown menu showing 'Big Data'.
- Data Source ***: A dropdown menu showing 'foodmart'.
- Upload File:** A section with a 'Scegli file' button and the text 'Nessun file selezionato'.
- Lock Model**: A toggle switch that is currently turned off.
- Use Advanced View As QBE Default**: A toggle switch that is currently turned off.

Fig. 1.243: Setting the metamodel basic information.

After you have compiled these information, you can use the browse button to upload a model developed through an external (and specific) tool or you can click on “Save” on the top right corner of the screen and use the Meta Web engine to build it through Knowage interface. Now click on the switch **Enable Meta Web** that will show up a button **Meta Web**, click on that and you are ready to design the model.

1.6.1.2 Association with analytical drivers

With this feature in Meta web is possible to create filters using associations with analytical drivers. In this section we will show how to practically define this kind of association.

To add a new parameter, you can click on the tab **Drivers** and then on a **Add** button, see the next figure.

Choose a name for the title of the driver. Then choose an analytical driver from drop-down menu that you wish to associate to the meta model.

Once you have selected the driver, you should write the **exact URL** of the corresponding parameter. Then set if the driver is multivalue or not. By default, drivers are set to be mandatory so you don't have the option to change that like you have for analytical documents.

After you have completed the definition of a parameter you can save it by clicking on main **Save** button in the upper right corner. To add further parameters, click on the **Add** button. Repeat the same procedure as many times you want. At this point you may wish to change the order of parameters (i.e., how they are presented to the user). To do so, click on the arrow in the list of drivers.

In the following chapter we'll see some special operations that can be performed on drivers associated to a meta model.

Fig. 1.244: Association with analytical driver panel.

Fig. 1.245: Association with analytical driver panel.

1.6.1.2.1 Correlation between parameters

In the context of a meta model, two different parameters may be connected to each other: this means that the possible values of a parameter are limited by the value(s) of another parameter.

This feature can be useful when two (or more) parameters are logically related. For example, suppose to have a parameter for all the possible countries and another one for all the possible cities. If the user selects a country, it is meaningless to show him all cities: he should only be enabled to choose among the cities in the selected country.

To set the correlation, select the child parameter which will show you the details of that particular driver and then click on the **Add condition** button to open a pop-up window for defining a data correlation.



Fig. 1.246: Adding data correlation.

In general, to configure a correlation within a meta model you should make sure that the LOV associated with the parent parameter and the one associated to the child parameter share at least one column. This column defines which value from the parent parameter will be applied to the child, in order to constrain the results.

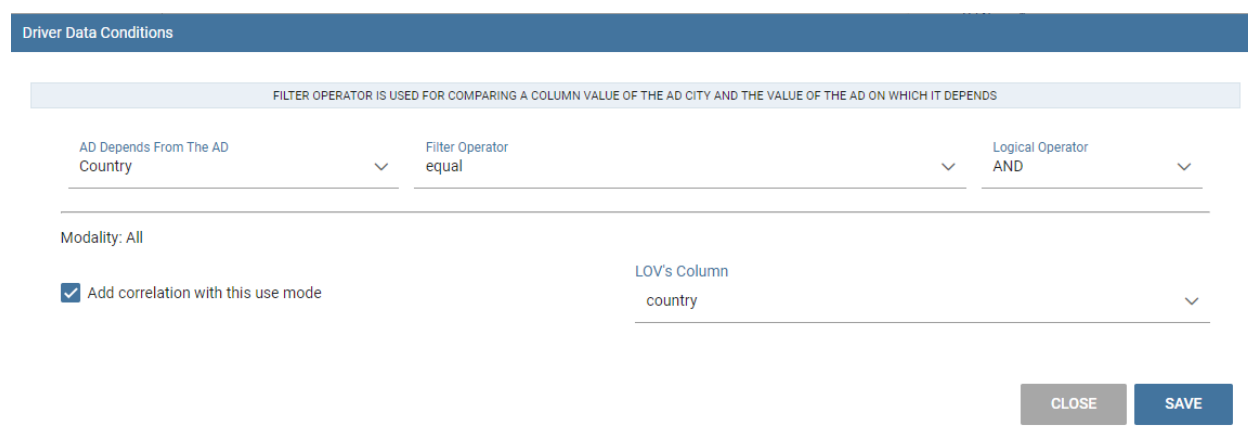


Fig. 1.247: Definition of the correlation.

Here you need to define:

- the parent parameter's AD;
- the type of filter operator, in order to compare values of the parent parameter with values of the child parameter;
- the logical operator to chain this correlation condition with other possible correlation (if any)
- (for each AD modality) the column, generated by the current parameter's LOV, whose value will be compared with the value of the same column in the parent parameter.

If a parameter depends on multiple parent parameters, you can define multiple correlations.

Once the correlation is defined, the child parameters will have the labels displayed in italic during the execution (with QBE).

Drivers

Country
par_country

City
par_city

Family
family

Province
province

ADD

Driver's Details

Title *
City

Analytical Driver *
City

Uri Name *
par_city

Multivalue

Driver Data Conditions

ADD CONDITION

equal value of the parameter province

equal value of the parameter par_country

Fig. 1.248: Multiple correlations.

1.6.1.2.2 Correlation through LOV and drivers

In previous sections we saw how to set correlation through the GUI available in meta model panel, but there is also the possibility to get the same result using the link between LOV and analytical drivers. More in depth, the user must have previously configured a driver that runs values that can be used in the “where” clause of a SQL query. Then the user must set a query-type LOV using the syntax:

Listing 1.18: Syntax for setting correlation through LOV configuration

```
1  $P{AD_name}
```

We stress that the AD_name is the name of the driver the administrator is trying to reach. As a result, when opening meta model, as soon as the user pick up a value from the “free” parameter, the other one is filtered and it will show only the value related to the previous selection, as shown in Figure below.

1.6.1.2.3 Create an empty model

The first time you enter the Meta Web, the interface will show you the available tables extracted from the selected data source.

For each table you can decide if you want to include it in your metamodel. More in detail a metamodel is divided in two model:

- **Physical Model:** it represents a “snapshot” of the database at the moment of the creation of you metamodel. The physical model contains a list of tables and information like columns and foreign keys retrieved from the database. The Physical Model cannot be modified but could be updated to reflect changes made on the database after the creation.
- **Business Model:** it is based on the physical model but let the user recombine some of his information. For example is possible to create a Business Class that contains only some of the columns of a Physical Table and create new relationships between Business Classes that are not defined on the physical database.

If you choose to include a table only in the physical model is always possible to create a corresponding business class later during the editing. When you have finished to select the tables you can proceed to the editing clicking on the **Continue** button.

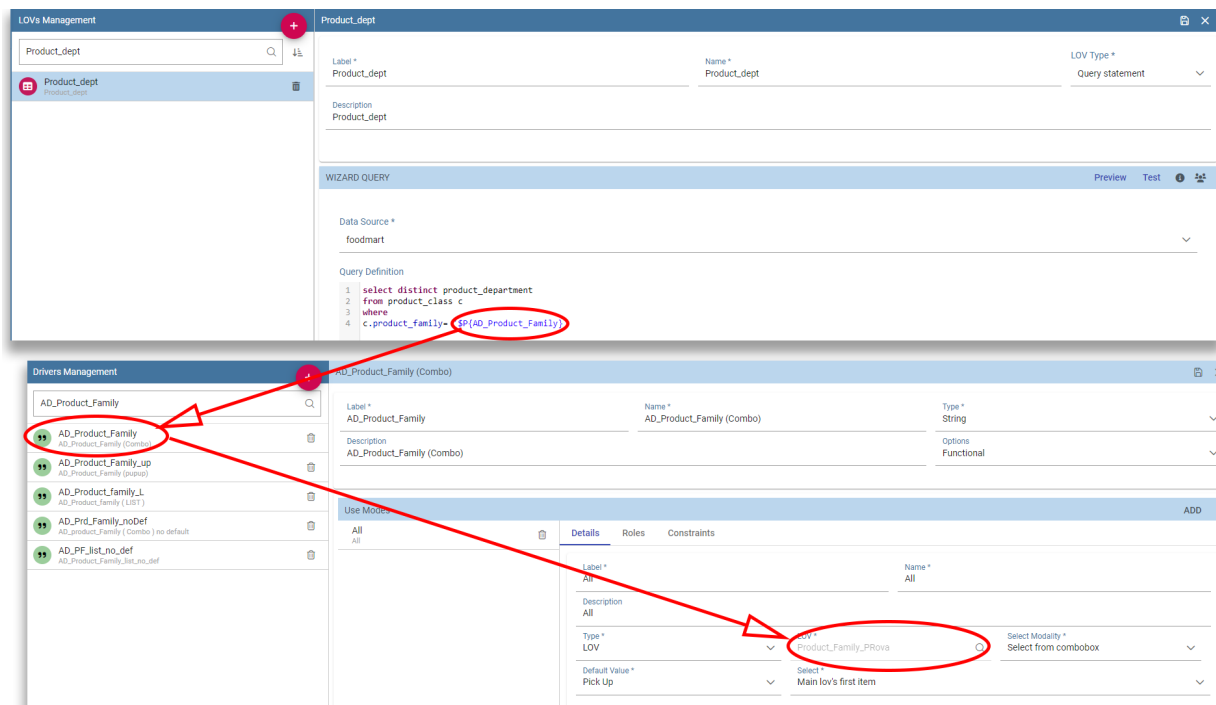


Fig. 1.249: Correlation passing driver values to LOV query .

1.6.1.2.4 Editing the metamodel

The Meta Web Editor is divided in two main tabs **Business Model** and **Physical Model** corresponding to the related models. Clicking on one of this tab will change the view showing the elements of the specific model.

The “Physical Model” tab contains the tables that the user has checked earlier. On the left side of the interface you will see a tree like structure with the list of tables imported in the Physical Model (see figure below).

The “hamburger-like” icon lets the user to update the Physical Model at any time. Referring to the figure below, selecting the “Update Physical Model” option the user can refresh the model metadata.

As shown below, the interface shows if tables have been added or deleted to the datasource and lets the user to add tables to the Physical Model.

Each table of Physical Model brings the information read from data base. Selecting each table, the interface shows on the right the list of its properties (**Property List** tab) and its foreign keys (**Foreign Keys** tab). Clicking on the icon on the left of each Physical Table, it is possible to expand the corresponding node. Highlight each column name to see (on the right side of the screen) a list of properties, like data type or length.

The Business Model tab, shown below, allows the user to custom the model in terms of column name, type, visibility, format, etc.

In this view, you see all the Business Class created at the first initialization. As well, the Business Classes are represented in a tree structure on the left side of the page. Clicking on each business class name, generic information are reported in the five tabs available on the right side of the page (Figure below).

If you want to change order of the business classes, you can do it using drag and drop functionality.

Using the **Property List** tab, the user can custom the business class name, type a description and see the corresponding physical table name. Here the user can also choose to hide the business class setting its visibility to false. Furthermore, when specifying the business class type, the user activates some peculiar functions that can be used in the QbE interface

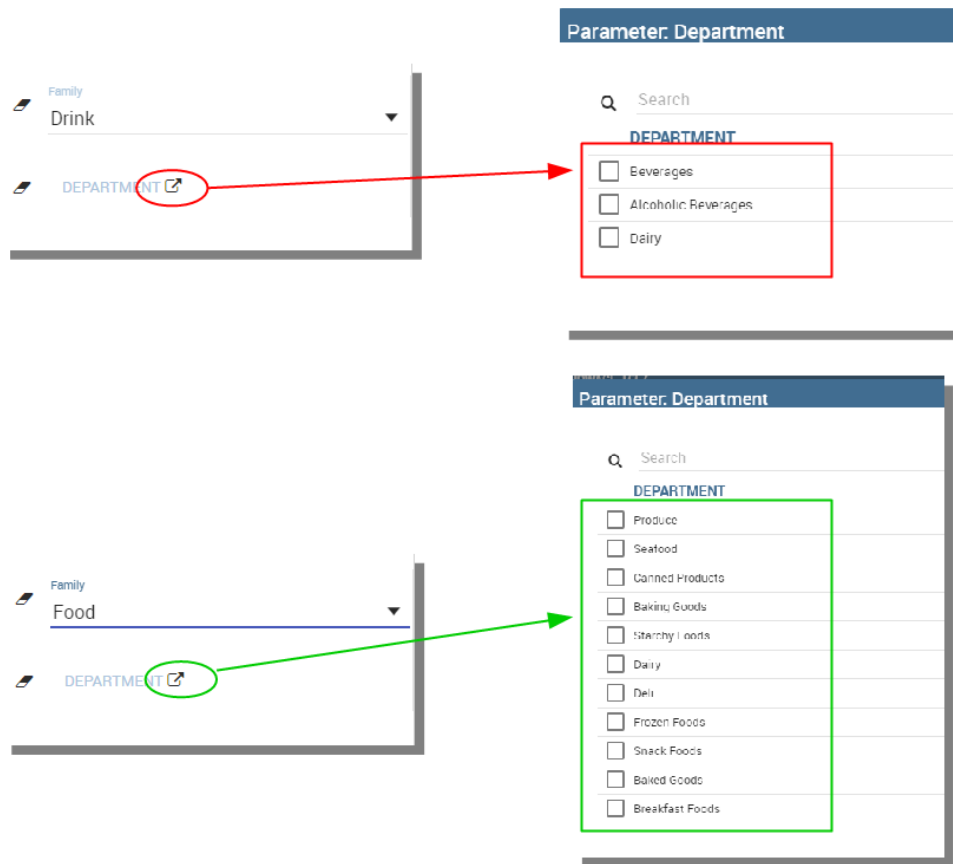


Fig. 1.250: Filtering with correlation.

TestDependenciesCrud			CLOSE	CONTINUE
<div> <div>Q search</div> </div>				
Table Name	<input type="checkbox"/> Physical Model	<input type="checkbox"/> Business Model		
agg_pl01_sales_fact_1997	<input type="checkbox"/>	<input type="checkbox"/>		
agg_ll01_sales_fact_1997	<input type="checkbox"/>	<input type="checkbox"/>		
agg_lc100_sales_fact_1997	<input type="checkbox"/>	<input type="checkbox"/>		
agg_lc06_sales_fact_1997	<input type="checkbox"/>	<input type="checkbox"/>		
agg_ll05_sales_fact_1997	<input type="checkbox"/>	<input type="checkbox"/>		
agg_ll04_sales_fact_1997	<input type="checkbox"/>	<input type="checkbox"/>		
agg_ll03_sales_fact_1997	<input type="checkbox"/>	<input type="checkbox"/>		
agg_gms_pcat_sales_fact_1997	<input type="checkbox"/>	<input type="checkbox"/>		
agg_c_special_sales_fact_1997	<input type="checkbox"/>	<input type="checkbox"/>		

Fig. 1.251: Metaweb interface.

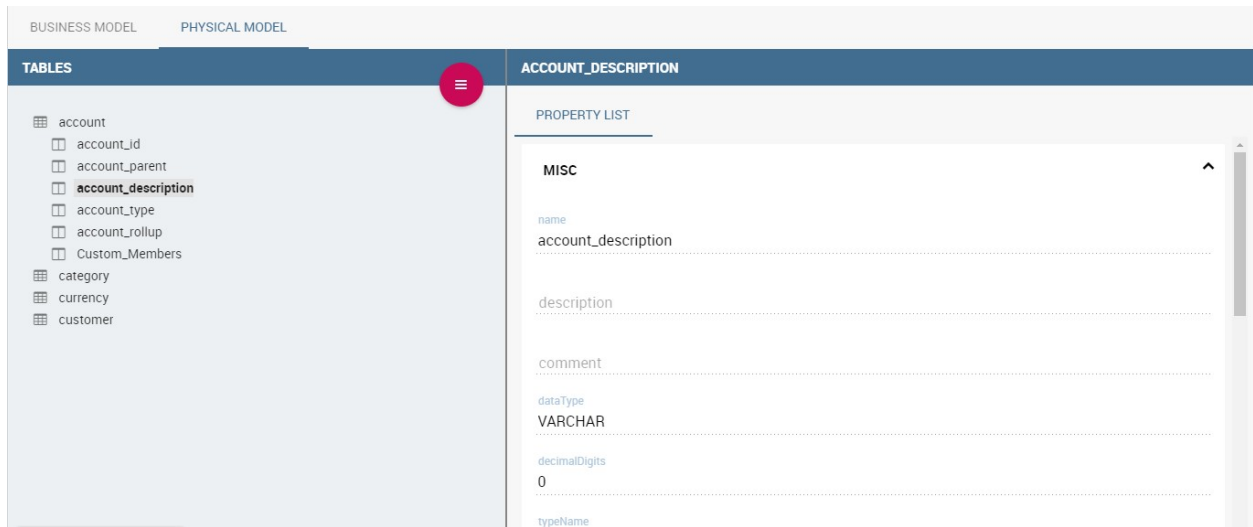


Fig. 1.252: Physical Model Tab.

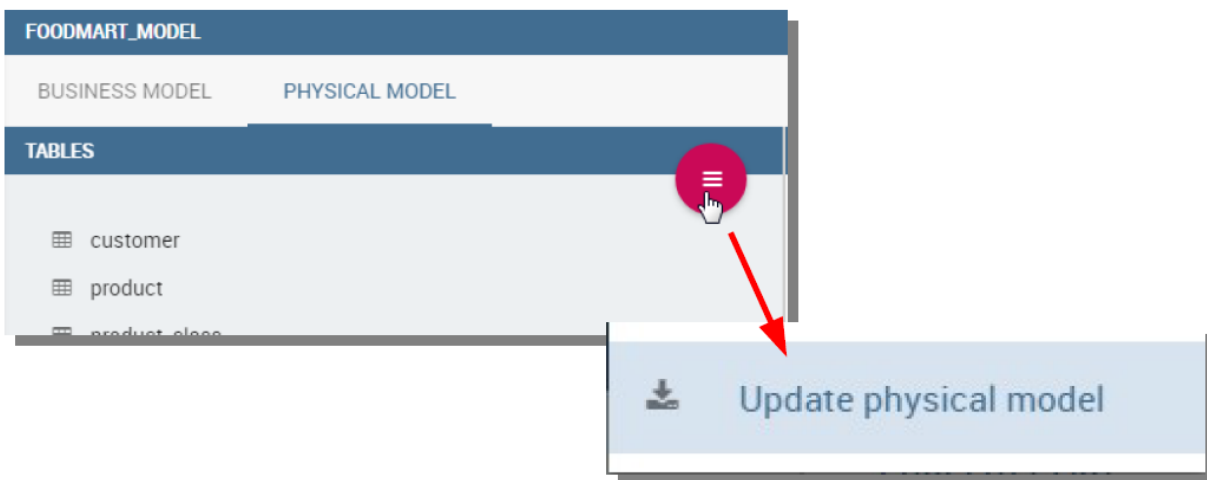


Fig. 1.253: Update the physical model.

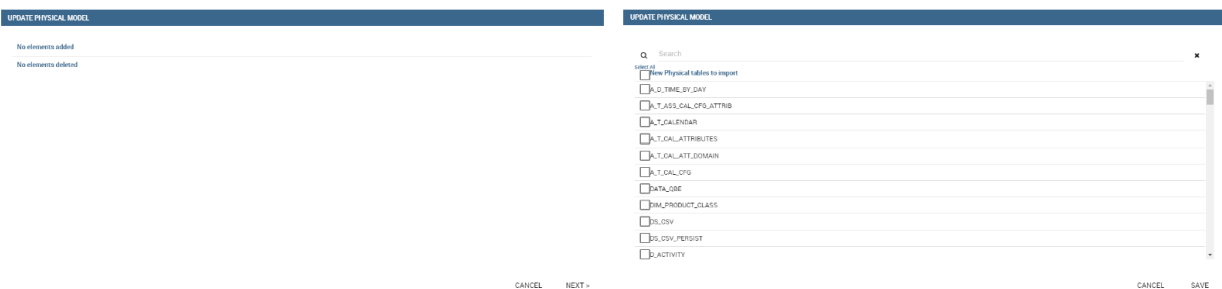


Fig. 1.254: Update the physical model.

BUSINESS MODEL		PHYSICAL MODEL										
Business Class/Business View		Customer						DELETE				
Business Class												
Customer 29 Attributes		PROPERTY LIST						ATTRIBUTES	CALCULATED FIELD	INBOUND	OUTBOUND	FILTER
Sales 11 Attributes												
Product 14 Attributes												
Store 24 Attributes												
Time 14 Attributes												
Product Class 5 Attributes												
Promotion 7 Attributes												

The **Attributes** tab lets the user to define which columns to be used as primary keys and which are effectively functional for the Business Class (not to be confused with the visibility condition). Note that, for instance, it is not possible to disable the “In Use” option when the field has already been set as foreign key.

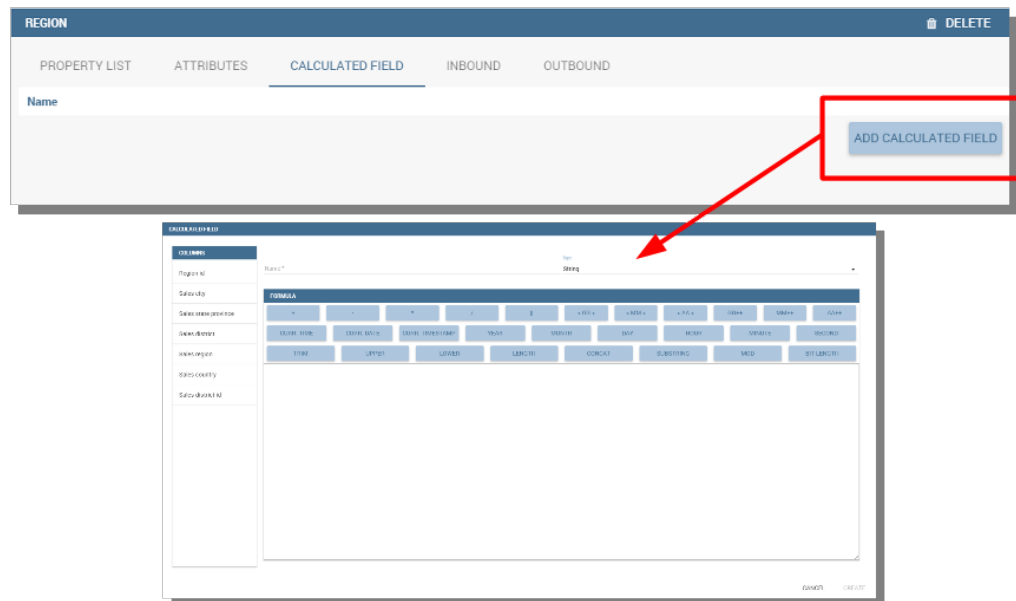


Fig. 1.257: Add calculated fields.

The **Inbound** and **Outbound** tabs are described in the following section.

1.6.1.2.5 Create a new relationship

In the Business Model is possible to define new relationships between Business Classes that are not inherited from the physical foreign keys. The Business Relationships are divided in two types:

- **Inbound:** relationships that have the selected Business Class as a target (so they are entering);
- **Outbound:** relationships that have the selected Business Class as a source (so the starts from).

The two relationships differ then for the direction of the bounds between tables that they establish.

To create a new relationship, just select the tab “Inbound” or “Outbound” after selecting one Business Class. Then click on the button “Add” and you will see a dialog.

In Figure above the outbound relationship is shown. Here you have to:

- enter the business relationship name,
- select the cardinality of the relationship (1 to N is suggested),

OUTBOUND

Insert a Business Relationship Name *

MyNewRelationship

Cardinality



1 to N ▼

Source Business Class

Customer

Target Business Class

Account ▼

SOURCE ATTRIBUTES	TARGET ATTRIBUTES
Customer id	Account id  Customer id 
Account num	Account parent
Lname	Account description
Fname	Account type

CANCEL CREATE

Fig. 1.258: Setting the outbound relationship.

- select the Source and Target Business Classes,
- Then is possible to drag and drop a Business attribute from the source Business Class to another Business attribute in the target Business Class. This will create a link between the two attributes.

When all these steps are accomplished, click on “Create” to save.

We stress that the cardinality of the outbound relationship can be of two types:

- 1 to N,
- 1 to N*.

Use the second type of cardinality when the type of cardinality can be optional.

As well, the cardinality of the inbound relationship can be of two types:

- N to 1,
- N* to 1.

Use the second type of cardinality when the type of cardinality can be optional.

1.6.1.2.6 SQL Filter

There is a new feature that is added in meta web. It is SQL Filter which we can define in Filter tab in meta web as you can see in the figure below. SQL Filter is used for applying already defined drivers in query.

SQL filter is expression that is added in the end of query as part of where clause. The right syntax for sql filter is: column_name = \$P{url_name_of_the_driver}. For example: city = \$P{cityUrl}. If you want to add more than one filter, you can connect them with an operator (AND, OR...) as you can see in an example in figure below.

If you want to add filter for multivalue driver the right syntax is this: column_name IN (\$P{url_name_of_the_driver}). For example: city IN (\$P{cityUrl}).

Customer						DELETE
PROPERTY LIST	ATTRIBUTES	CALCULATED FIELD	INBOUND	OUTBOUND	<u>FILTER</u>	
<p>SQL expression</p>						

Customer						DELETE
PROPERTY LIST	ATTRIBUTES	CALCULATED FIELD	INBOUND	OUTBOUND	<u>FILTER</u>	
<p>SQL expression</p> <p>country = \$P{countryUrl} AND city = \$P{cityUrl}</p>						

1.6.1.2.7 Create a new business class

In the “Business Model” tab, the sandwich icon lets the user add other Business Classes (from the tables of the Physical Model) or a Business View (a combination of more tables with a predefined join path).

When clicking on the icon, as shown in Figure above), and selecting “New Business Class”, a new dialog asks to the users to:

- select a Physical Table from the available ones;
- insert a description for this new business class;
- select one or more columns.

Then click on save to add the business class.

As well, when clicking on “New Business View”, as reported in Figure below the user is asked to select two or more tables from the available ones and insert a description for this new business view.

Then, moving to the next step, the user must join tables through specific columns, typically the tables’ foreign keys. Figure below shows an example.

For each business view, the interface reports the same property tabs we saw for each business class. In addition, the user finds the **Join relationships** tab and the **Physical table** tab, as highlighted in the following figure. The “Join relationships” tab shows the join clauses set to create the business view while the “Physical Table” tab recalls the physical table names.

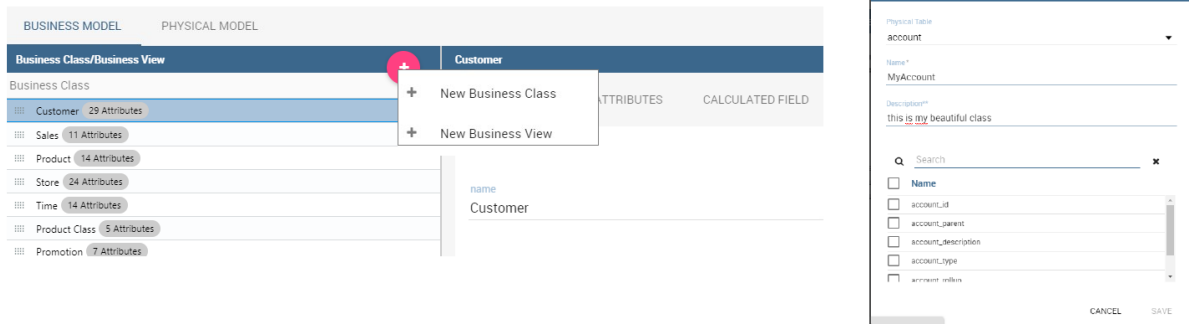


Fig. 1.259: Create a new business class.

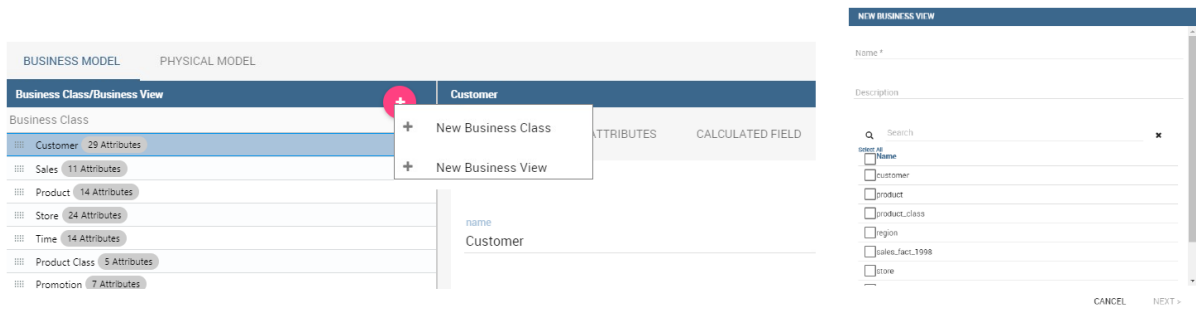


Fig. 1.260: Create a new business view.

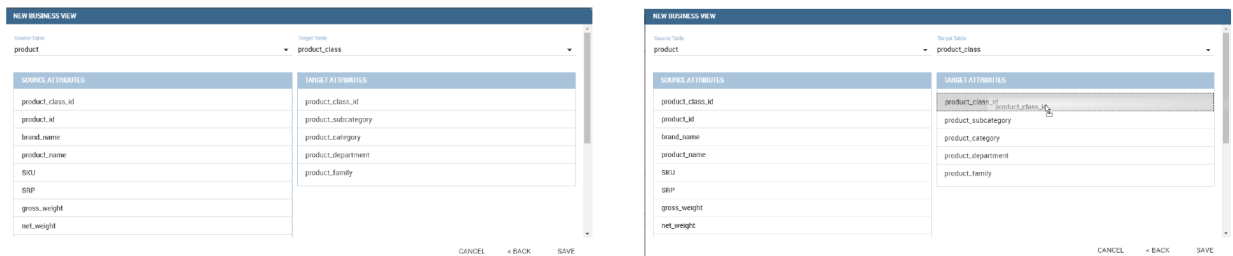


Fig. 1.261: Create a new business view.

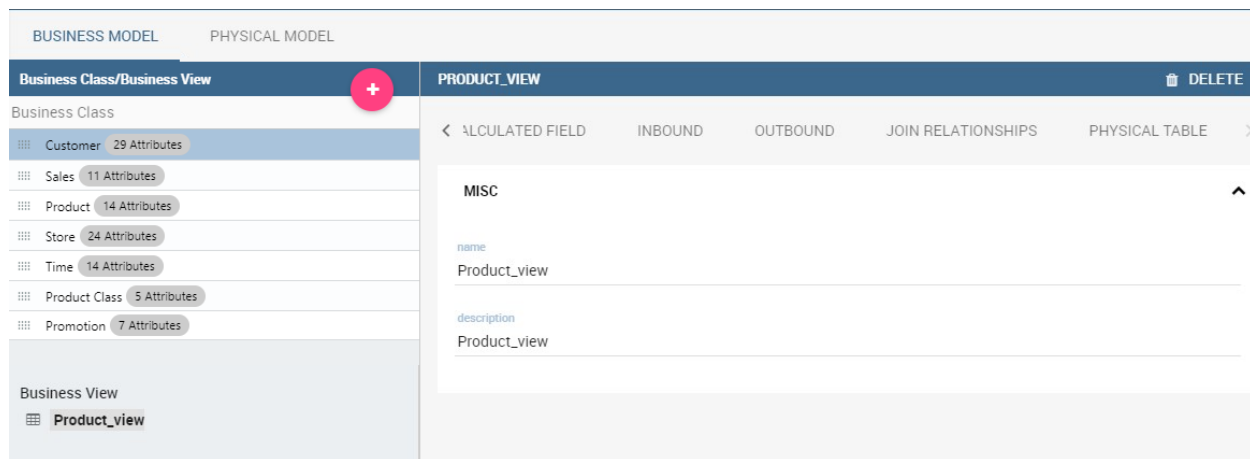


Fig. 1.262: Additional property tabs for business view.

1.6.1.2.8 Table property list

Scrolling the table “Property list” tab, the user finds the **Type** menu item. Expanding the related combobox the user can custom the table type among the ones available and listed below.

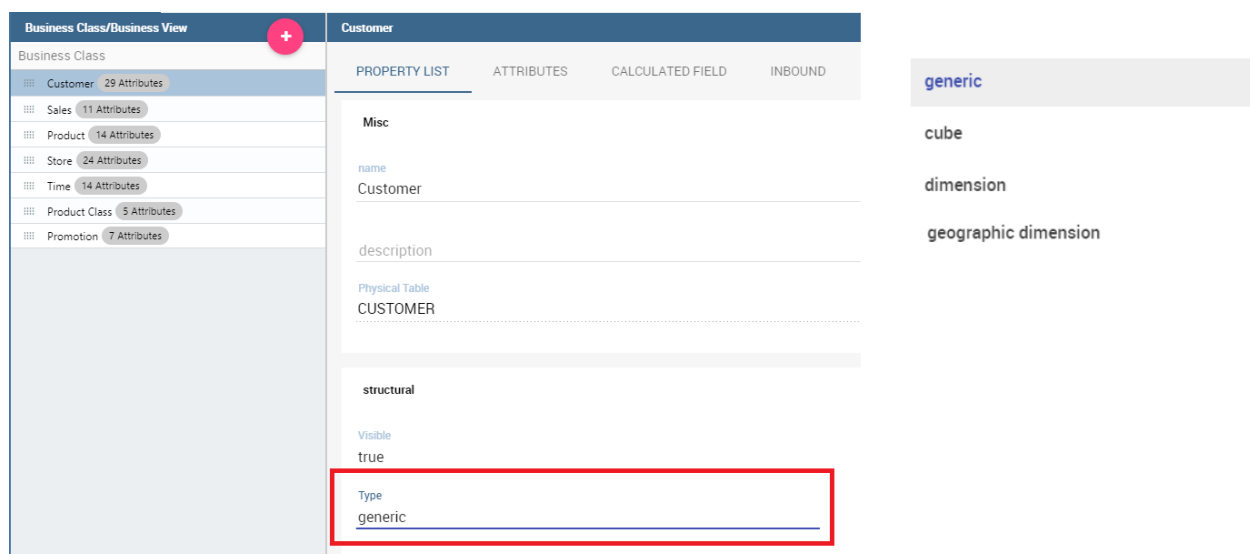


Fig. 1.263: Table property list.

1.6.1.2.9 Column property list

As well, the user can employ each field property list (see next figure) to both inspect the object and custom it.

The **Structural** area covers an important role for the field properties. Here the user can set:

- **Visibility** over the field,
- **Type**, among measure, attribute, calendar, temporal_id, the_date and hour_id,
- **Aggregation type** for measure field type,

The screenshot shows a dialog box titled "Attribute's Detail" with a blue header. It contains several sections for configuring a column:

- Name:** The month
- Description:** (empty field)
- Type:** attribute
- Structural:**
 - Aggregation Type:** COUNT (with a dropdown arrow)
 - Data Type:** VARCHAR (with a dropdown arrow)
 - Roles Enabled:** (empty dropdown menu)
 - Physical Table:** INVENTORY
 - Profile Attribute Filter Type:** EQUALS TO (with a dropdown arrow)
- Profile Attribute:** the_month (with a dropdown arrow)
- Custom Function:** (empty field)

At the bottom right, there are two red buttons: "CANCEL" and "SAVE".

Fig. 1.264: Column property list.

- **Format string**, to custom the format of the string for measure field type,
- **Profile attribute**, to filter the field (and then the table records) by the user profile attributes (note that the combobox lists the available profile attributes),
- **Profile attribute filter type**, to define the filter operator among “equals to”, “in”, “like”,
- **Data type**, to indicate the field data type.

In the **Behavioural Model** area, the user can assign the field’s visibility permission to specific roles.

In the **Physical** area, recalls the physical table and field name from which the field have been take.

Add new column into business class

If you did not choose all columns from physical table, when you were creating new business class, you can do it easily. Click on business class in which you want to add new column. Open Attributes tab and click on above column name.

If you want to remove column from business table, you need to click on business class from which you want to remove column. Open Attributes tab and click on three dots of column you want to delete. It will open details panel. Click on delete button.

If you want to change order of columns, you can do it using drad and drop functionality.

1.6.1.2.10 Generate the datamart

After the editing of the metamodel, click on “Save” on the Meta Web toolbar on the upper right corner. Now you have a metamodel that can be compiled and used to generate a datamart. Now if you go back to the Business Model catalog you will see that near the “Meta Web” button there is a “Generate” button. Clicking on it, a dialog will open:

If you just press “Create” the generation of the datamart begins otherwise clicking on the switch “Show Advanced options” (see figure below) the user can modify model name, change the schema or the catalogue of the database used to query the metamodel. This option is useful when the user wishes to build the model on a source schema and produce the datamart on a different one. Furthermore, the user can check the **Generate for registry** box. In this instance, the generated datamart will be used as a registry (but will not be exploited as a QbE). The **Include source code** produces a “file.jar” containing both the compiled code (.class) and the source files (.java), useful for the debugging process.

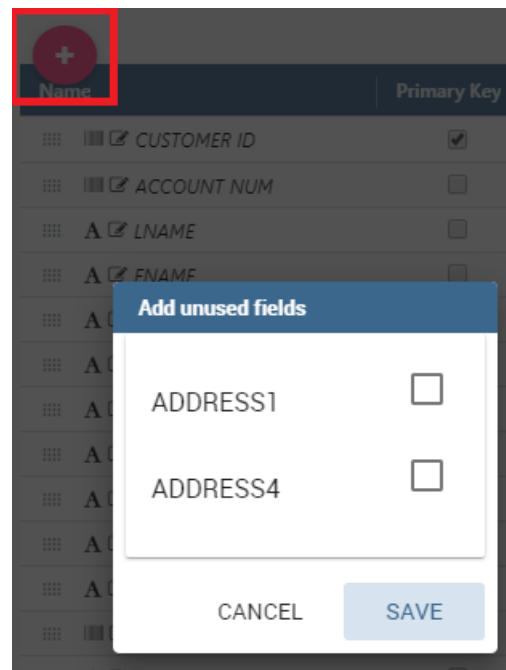


Fig. 1.265: Remove existing column from business class

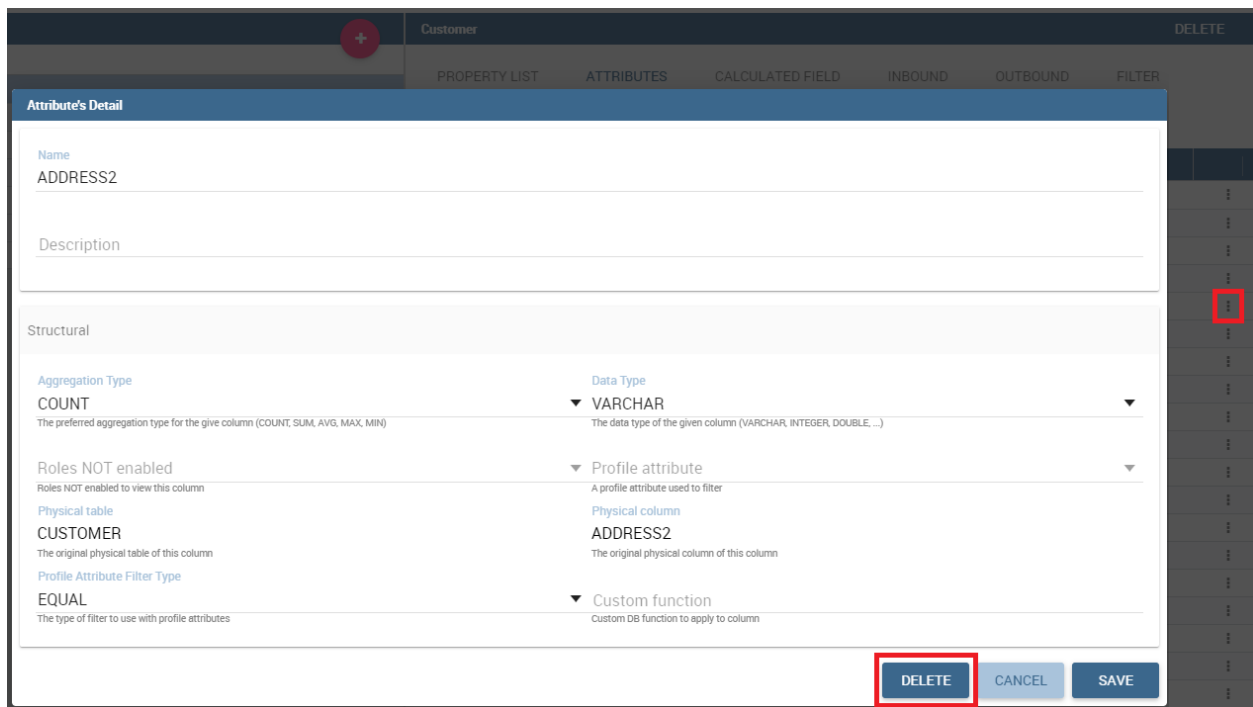


Fig. 1.266: Change the order of the columns

Customer

DELETE

PROPERTY LIST

ATTRIBUTES

CALCULATED FIELD

INBOUND

OUTBOUND

FILTER

+

Name	Primary Key	visibility	Type	Description
CUSTOMER ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
ACCOUNT NUM	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A LNAME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A FNAME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A ADDRESS2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A ADDRESS3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A MI	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A CITY	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null

Customer

DELETE

PROPERTY LIST

ATTRIBUTES

CALCULATED FIELD

INBOUND

OUTBOUND

FILTER

+

Name	Primary Key	visibility	Type	Description
CUSTOMER ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
ACCOUNT NUM	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A LNAME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A MI MI	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A FNAME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A ADDRESS2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A ADDRESS3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A CITY	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null
A STATE PROVINCE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute	null

GENERATE DATAMART

Click on Create to generate the datamart.jar with the default values (recommended), otherwise change the options values for different datamart (for example on another schema/catalog).

☐ Show advanced options

CANCEL CREATE

Fig. 1.267: Generate datamart dialog.

GENERATE DATAMART

Click on Create to generate the datamart.jar with the default values (recommended), otherwise change the options values for different datamart (for example on another schema/catalog).

 Show advanced options

Model Name

Foodmart_model

Schema Name

Catalog Name

foodmart

☐ Generate for registry ☐ Include source code

CANCEL

CREATE

Fig. 1.268: Generate datamart dialog: advanced options.

When the datamart is generated it will be possible to query the metamodel accessing it in the Workspace interface.

1.6.1.2.11 Additional functions for business model

In this section, we briefly describe the generic available options for business model development. Referring to figure below, the user first finds the **Lock Model**: if enabled, only the user who developed the model can modify it.

The screenshot shows a web interface for managing a business model named 'Foodmart_model'. The interface has a top bar with 'Foodmart_model', 'SAVE', and 'CLOSE' buttons. Below this is a tabbed interface with 'DETAIL', 'METADATA', 'SAVED VERSIONS', and 'DRIVERS' tabs. The 'DETAIL' tab is active, showing a form with the following fields:

- Name***: Foodmart_model
- Description**: Foodmart_model
- Category***: Default Model Category
- Data Source***: Foodmart

At the bottom of the form, there is an 'Upload File:' section with a 'BROWSE' button. To the right of the 'BROWSE' button are two toggle switches: 'Enable Meta Web' (which is currently turned off) and 'Lock model' (which is currently turned on).

Fig. 1.269: Business model lock.

Once the model has been saved, some more options are enabled. In fact, the user can make advantage of the **Metadata** section. Clicking the **Import metadata** button, the metadata information related to the business classes (their composition, properties, etc.) are stored into the (metadata) Knowage database. Those information can then be visualized via specific document (developed for the data lineage context).

The screenshot shows the same web interface as Figure 1.269, but with the 'METADATA' tab selected. The 'DETAIL' tab is no longer active. The 'IMPORT METADATA' button is prominently displayed in the center of the page.

Fig. 1.270: Importing metadata.

Finally the **Saved versions** section the user keeps trace of model changes over time. Furthermore it is possible to restore old versions by checking the active column. Selecting the “three-dots” icon the user can download the jar file or the model itself or delete the version. Figure below shows an example.

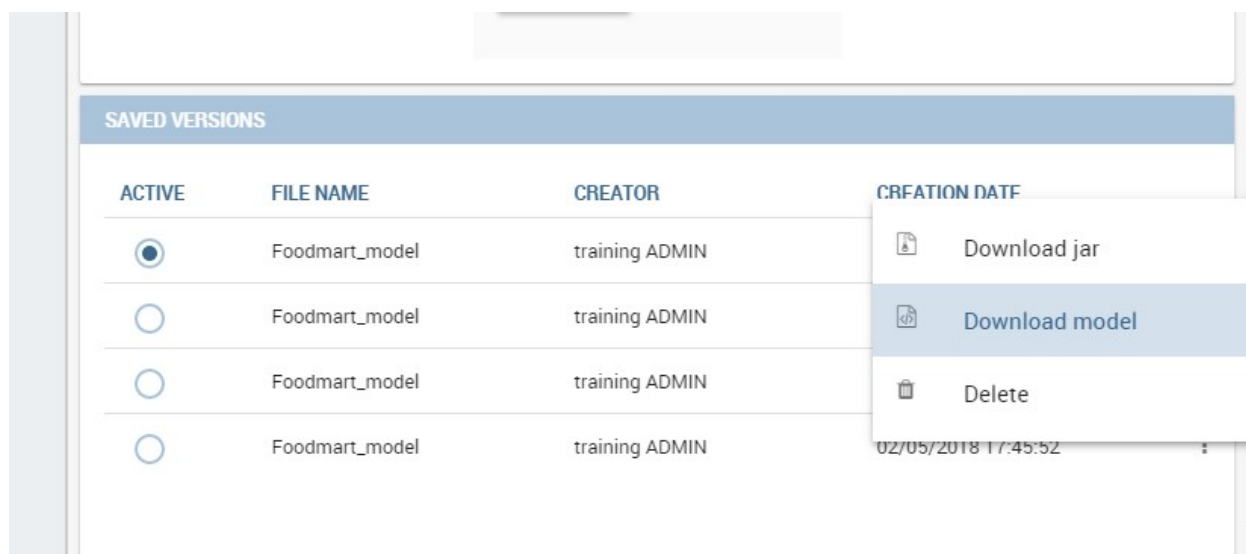


Fig. 1.271: Saved version functionalities.

1.6.2 How to build a QBE

This detailed user guide is dedicated to the Qbe (Query By Example), a Free Inquiry instrument which empowers users with easy and free access to information via graphical interfaces.

Free Inquiry indicates the modus operandi of analysts and operational users that are usually seeking for business analysis that are not limited to pre-arranged lists of results. This method requires a medium level of complexity as an adequate knowledge of data management and a structured organization of work are required.

QbE is a tool that allows you to develop your free inquiry through an entirely graphical modality. Moreover, you can execute the query, check the results, export them and save the query for further use.

The material will be divided in two main sections. The first is dedicated to build queries in the Knowage Server environment, supposing the availability of a business model to analyse. In the second part, we will provide the user with the principal steps to build a proper business model through the Qbe designer, available in Knowage Meta.

1.6.2.1 My first Query By Example

QbE allows you to query (a subset of) a database through a high-level representation of the entities and relations. Qbe main features are:

- rich end user GUI;
- selection of attributes and setting of filters;
- no knowledge of data structures;
- semantic knowledge of data;
- management of results free;
- support of export capabilities;
- repeatable execution of inquiries.

Building a QbE query does not require any technical knowledge, but data domain knowledge: technical aspects, such as creating filters, aggregation and ordering criteria, are managed by a user-friendly graphical interface.

Let's suppose that an administrator has built a business model and, consequently, released it on Knowage Server. This allows the user to access the model, query the available entities and save the results as a dataset, for further usage in different Knowage documents, such as cockpits.

In the following we will discuss each step in detail, showing the basic and advanced functionalities of the **QbE Editor**.

1.6.2.1.1 Query design and execution

To open the QbE editor, access the **Models** section, available in the end user **Workspace**. Then, simply click on the model icon to reach the QbE graphical interface.

In this paragraph we show how to build a simple query with the QbE editor.

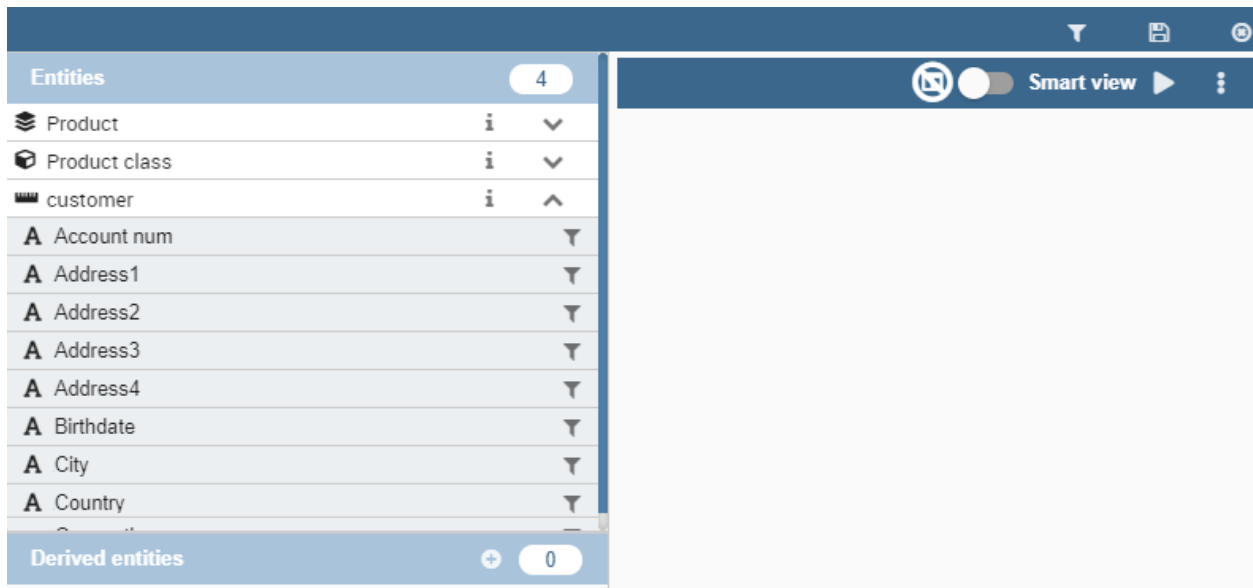


Fig. 1.272: QbE editor.

The above figure shows the **Query designer**. In next sections we will explain in detail all the components related to the **Query Designer**, the **Datamart Schema** tab, the query editor and the hidden tab dedicated to the management of queries, subqueries and parameters catalogue.

1.6.2.1.1.1 Datamart Schema

Starting from the left side: The upper Panel shows the searchable logical schema and the list of entities that can be queried to generate the query. Entities are represented in a tree structure, with user-defined names. Fields can be added to the query (right side) by clicking on them. The lower Panel shows the list of created subqueries in a tree structure where the children are the fields of a subquery

Types of entities are: *facts*, represented by a cube symbol.(i.e., the Sales entity), *dimensions*, represented by a four-arrows symbol (i.e., the Product entity), *geographical dimension*, represented by *earth* icon.

Each single entity contains a title, a list of attributes or measures and relationships with other entities. Relations are available clicking on the *i* icon of a single entity. In particular, by exploding the content of an entity (i.e. Sales), you may encounter the following elements:

- **measure:** refers to fields related to numeric data (e.g. UNIT SALES);
- **attribute:** refers to fields that can be associated to a category (e.g. PRODUCT ID);

- **relation**: refers to relationships or connections between two entities (e.g. relationship between the sales and the product dimension).

Sales				
Relation name ↑↓	Source fields ↑↓	Target entity ↑↓	Target fields ↑↓	Join type ↑↓
time	time_id	Time	time_id	INNER
product	product_id	Product	product_id	INNER

OK

Fig. 1.273: Relations of an entity.

There are two available views: Smart and Advanced. When the QbE is opened, by default the user will see the Smart view. To add a field of a specific entity to the query, just click on it and no other user interaction will be necessary to display the results.

Brand name	Product name	SKU	Gross weight	Recyclable pack...
Washington	Washington Berry Juice	90,748,583,674	8	0
Washington	Washington Diet Soda	85,561,191,439	7	1
Washington	Washington Cola	29,804,642,796	16	0
Washington	Washington Diet Cola	20,191,444,754	18	1
Washington	Washington Cranberry Juice	49,395,100,474	7	0
Washington	Washington Apple Juice	22,114,084,362	8	1
Jeffers	Jeffers Oatmeal	49,031,038,880	9	1
Jeffers	Jeffers Corn Puffs	13,229,009,509	10	0
Jeffers	Jeffers Wheat Puffs	92,942,813,038	22	1
Jeffers	Jeffers Grits	26,378,549,933	21	0
Blue Label	Blue Label Canned Beets	62,908,702,492	21	0
Blue Label	Blue Label Creamed Corn	79,484,335,780	7	1
Blue Label	Blue Label Canned String Bean	85,252,254,605	13	1
Blue Label	Blue Label Chicken Soup	47,163,524,031	15	1
Blue Label	Blue Label Canned Yams	22,169,209,122	21	0
Blue Label	Blue Label Vegetable Soup	43,318,244,814	20	0
Blue Label	Blue Label Canned Tomatos	77,561,696,171	15	0
Blue Label	Blue Label Noodle Soup	22,620,225,627	11	1

Fig. 1.274: Smart view.

The user can display the Advanced view by switching off the Smart view option displayed on the top right corner. The user can continue adding fields to the query but without seeing result. A specific icon has to be clicked to run the query.

The structure of the panel containing the fields of the query as well as the functionalities, changes depending which view is displayed. For example, to remove a field from the query editor, just click on the **X** icon available for each column in the Smart view or click on the **delete** item of the three dots menu available in the Advanced view. In any case, the Smart view, shows a table structure where each column of the table corresponds to a field of the query and contains:

- a **gear** icon providing functionalities like *Show field*, *Rename Alias*, *Group*
- a **deleting** icon
- a **filter** icon

The Advanced view shows a set of rows where any row contains the **Entity name**, the **Field name**, the **Alias** plus some other functionalities:

- **Alias**: define aliases for fields
- **Aggregation**: define the aggregation function (e.g., **SUM**, **AVERAGE**, ...) on non-grouped items;

Entity	Field	Alias	Group	Order	Aggr.	Show	In use
Product	Brand name	Brand name	<input checked="" type="checkbox"/>	NONE	NONE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Product	Product name	Product name	<input checked="" type="checkbox"/>	NONE	NONE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Product	SKU	SKU	<input type="checkbox"/>	NONE	SUM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Product	Gross weight	Gross weight	<input checked="" type="checkbox"/>	NONE	NONE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Product	Recyclable package	Recyclable package	<input checked="" type="checkbox"/>	NONE	NONE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Product	Units per case	Units per case	<input checked="" type="checkbox"/>	NONE	NONE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Product	Shelf depth	Shelf depth	<input checked="" type="checkbox"/>	NONE	NONE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Fig. 1.275: Advanced view

The day	The year	Product subcategory
01/01/2014		2,014
01/02/2014		2,014
01/03/2014		2,014
01/04/2014		2,014
01/05/2014		2,014
01/06/2014	Monday	2,014
01/07/2014	Tuesday	2,014

Fig. 1.276: Smart view functionalities.

- **Order:** define a sorting criteria: click on the arrow to select an ascending or descending criteria;
- **Group:** in case of aggregations, define the attribute that you want to group on (due to the SQL syntax, these attributes appear in the GROUP BY clause);
- **Show:** indicate whether a column shall be visible in the result or not (hidden attributes are used and returned by the generated query, but are not shown in the result table);
- **In use:** indicate whether a column shall be used in the select clause of the query or not
- **Filter:** to add a filter criteria;

Entity	Field	Alias	Group	Order	Aggr.	Show	In use
Time	The date	The date	<input checked="" type="checkbox"/>	NONE	NONE	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Time	The day	The day	<input checked="" type="checkbox"/>	NONE	NONE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Time	The year	The year	<input checked="" type="checkbox"/>	NONE	NONE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Product_Class	Product subcategory	Product subcategory	<input checked="" type="checkbox"/>	NONE	NONE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Fig. 1.277: Advanced view functionalities.

The below image shows how to change the alias for a specific field from the Advanced view. Just click on the cell containing the alias name to be able to edit it.

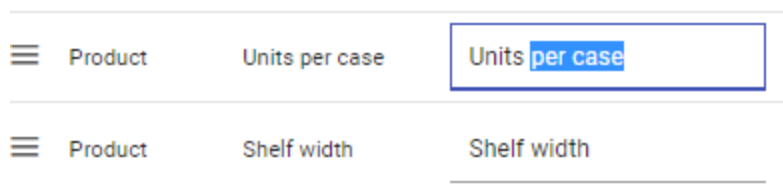


Fig. 1.278: Change alias example.

To change the alias of a field in the Smart view, just click on the **gear** icon and select *Rename Alias*. Pay attention to grouping options: if you want to define an aggregation function on a field (like, for instance, the **COUNT** of the sold items), you shall tick the **Group** checkbox for all the other fields added in the query editor, without an aggregation function defined, otherwise you will get an SQL exception. The possible grouping functions are shown in the following figure.

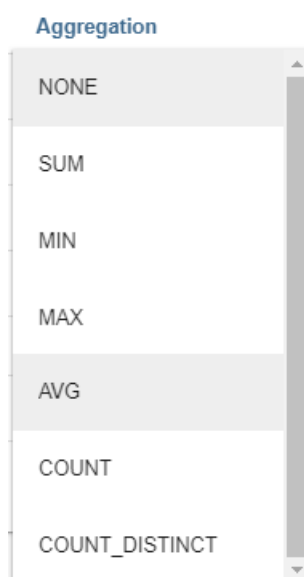


Fig. 1.279: Aggregation functions.

When you drag attributes belonging to entities that are linked through a relationship path, the QbE automatically resolves relationships between attributes (implicit join).

Moreover, multiple relationships may occur among entities. A typical example concerns dates. Suppose you have two relationships between the **Order** fact table and the **Time** dimension table: the first links the *order_date* column of the first table to the *time_id* column of the latter table, while the second relationship joins the *shipping_date* column to the *time_id* column.

In this case, when dragging fields from both the **Order** entity and the **Time** entity you may want to specify which relationship joins the two tables: for instance, you may want to know the total number of orders according to the ordering month, the shipping month or for both. In all these situations, you can set the relationship to be used by clicking the **Join Definitions** option of the three dots menu at the top right corner of the panel. A pop up window opens where you can change the path to be used. Please refer to Multiple relationships section for all the details regarding the ambiguity on relationships.

The below table summarizes some of the toolbar functionalities.

1.6.2.1.1.2 Calculated fields management

You can also add calculated fields to a query. This functionality is only available in the Advanced view through the item **Calculated field** of the three dots menu.

To build a calculated field, you shall define a:

- **Column Name**;
- **Type**: string, number or date;
- **Column Type**: measure or attribute;
- **Formula**: drag and drop the fields included on the left and build the formula using the available functions.

The image below shows the wizard.

Fig. 1.280: Calculated field wizard.

1.6.2.1.1.3 Filters

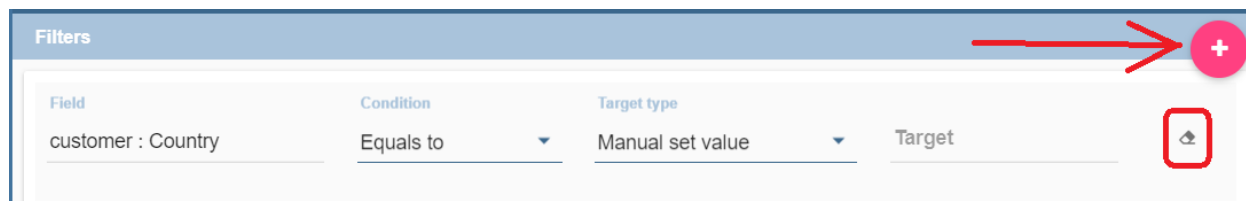
The **Filters** panel allows you to define a filter criteria, a WHERE clause to add to the query. Filters are expressions of type:

Left operand + Operator + Right operand.

How to add a filter: - in the Smart view, clicking on **Funnel icon** - in the Advanced view, selecting **Filters** from the *three dots* field menu

In both cases a pop up opens and click on the + icon to add your filter.

To remove the filter just click on **eraser** icon as shown in the below image.



The Filters panel contains the following components:

- **Field, Condition, Target**, these components allow you to define filters according to the syntax defined above.
- **Target type**, defines the types of right operand: *Manual set value*, *Value of the field*, *Another entity*, *Parameter*, *Subquery*;

Above, an example of filter with *Target type* set to **Manual set value**: the **Target** will contain the value to be used as the right operand.

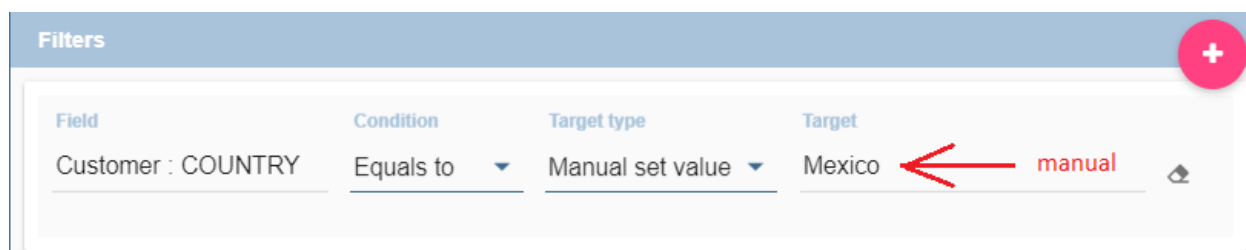


Fig. 1.281: Manual *Target type*

Manual set value can be also used choosing **Between**, **Not between** as *Condition*. In this case, a Low and High Limit is needed for *Target* values. When the user chooses **Value of the field** as *Target type*, a lookup function is automatically activated on the field values, to facilitate the selection of values to be used as the right operand. To use a range of values as a right operand, the *Condition* should be to **In** or **Not in**.

The option **Another entity** allows the selection of a field from another entity, to be used as a right operand.

Subquery and **Parameter** will be treated in detail later.

Important: Enterprise Edition only

Filtering data on fields with type of Date/Time/Timestamp through Calendar/Time functions is only available for the Enterprise Edition.

If you have the SI license file, you could filter your data with fields type of date/time/timestamp using Calendar/Time/Calendar + Time options. This depends on which data type your field is: the data type is assigned when creating the metamodel.

The following table summarizes the possible types of filters in the QbE. The use of subqueries in filters will be explained later in the *Advanced QbE functionalities* paragraph.

Valori	
<input type="checkbox"/>	BBB Best Columbian Coffee
<input type="checkbox"/>	BBB Best Decaf Coffee
<input type="checkbox"/>	BBB Best French Roast Coffee
<input type="checkbox"/>	BBB Best Hot Chocolate
<input type="checkbox"/>	BBB Best Regular Coffee
<input type="checkbox"/>	Booker 1% Milk
<input type="checkbox"/>	Booker 2% Milk
<input type="checkbox"/>	Booker Buttermilk
<input type="checkbox"/>	Booker Chocolate Milk
<input type="checkbox"/>	Booker Whole Milk

1 to 10 of 146 < < Page 1 of 15 > >|

Fig. 1.282: Filter lookup for right operand selection.

Filters			
Field	Condition	Target type	Target
Customer : CITY	Equals to	Another entity	<div><div>REGION ID</div><div>STORE NAME</div><div>STORE NUMBER</div><div>STORE CITY</div><div>STORE STATE</div></div> <div><div>CUSTOMER</div><div>SALES</div><div>PRODUCT</div><div>STORE</div><div>PRODUCT CLASS</div></div>

Property List **Attributes** Calculated Field Inbound Outbound Filter

name	primary key	visibility	type	description	personal	decrypt	subject id	ADD
TIME DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	attribute		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
DAY DESC	<input type="checkbox"/>	<input checked="" type="checkbox"/>						
DAY NAME	<input type="checkbox"/>	<input checked="" type="checkbox"/>						

Attribute's Detail

TIME DATE

Description

Type
attribute

Structural

Aggregation Type
COUNT

The preferred aggregation type for the give column (COUNT, SUM, AVG, MAX, MIN)

Index enabled
Index enabled to view this column

Physical Table
Index

The original physical table of this column

Data Type
INTEGER

DATE
TIME
TIMESTAMP
DECIMAL
BIGINT

Fig. 1.283: Creation of a Metamodel.

Filters

Field	Condition	Target type
Time : THE DATE	Equals to	Manual set value

11/01/2019

Filters

Field	Condition	Target type
Time : THE DATE	Between	Manual set value

11/01/2019 11/01/2019

Fig. 1.284: Filters on date fields.

The figure shows two examples of the 'Filters' configuration for time fields. Both examples have a blue header bar with the word 'Filters' and a red '+' button in the top right corner.

Top Screenshot:

- Field:** Time : THE DATE
- Condition:** Equals to
- Target type:** Manual set value
- Value:** 10 : 39

Bottom Screenshot:

- Field:** Time : THE DATE
- Condition:** Between
- Target type:** Manual set value
- Value 1:** 10 : 39
- Value 2:** 10 : 39

Fig. 1.285: Filters on time fields.

The figure shows two examples of the 'Filters' configuration for timestamp fields. Both examples have a blue header bar with the word 'Filters' and a red '+' button in the top right corner.

Top Screenshot:

- Field:** Time : THE DATE
- Condition:** Equals to
- Target type:** Manual set value
- Date:** 11/01/2019
- Time:** 10 : 38

Bottom Screenshot:

- Field:** Time : THE DATE
- Condition:** Between
- Target type:** Manual set value
- Date 1:** 11/01/2019
- Time 1:** 10 : 39
- Date 2:** 11/01/2019
- Time 2:** 10 : 39

Fig. 1.286: Filters on timestamp fields.

Table 1.8: Possible combinations of filters in the QbE.

Filter type	Left operand	Operator	Right operand	Example
Basic	Entity.attr ibute	Any	value	Prod.family = 'Food'
Basic	Entity.attr ibute	Any	Entity.attr ibute	Sales.sales > Sales.cost
Parametric	Entity.attr ibute	Any	[parameter]	Prod.family = [p_family]
Dynamic	Entity.attr ibute	Any	prompt	Prod.family = ?
Value list from subquery	Entity.attr ibute	In /not in	subquery	Sales.custo mer in subquery
Single value from subquery	subquery	< = >	value	Subquery > 0

When filtering a date attribute or a time attribute it is possible to apply a timespan to ease the insertion of values. Following the images below, we can see that the Timespan button appears when filterting, for instance, a date attribute. We recall that is it possible to configure a new timespan using the dedicated Knowage functionality that we described in the administrator guide.

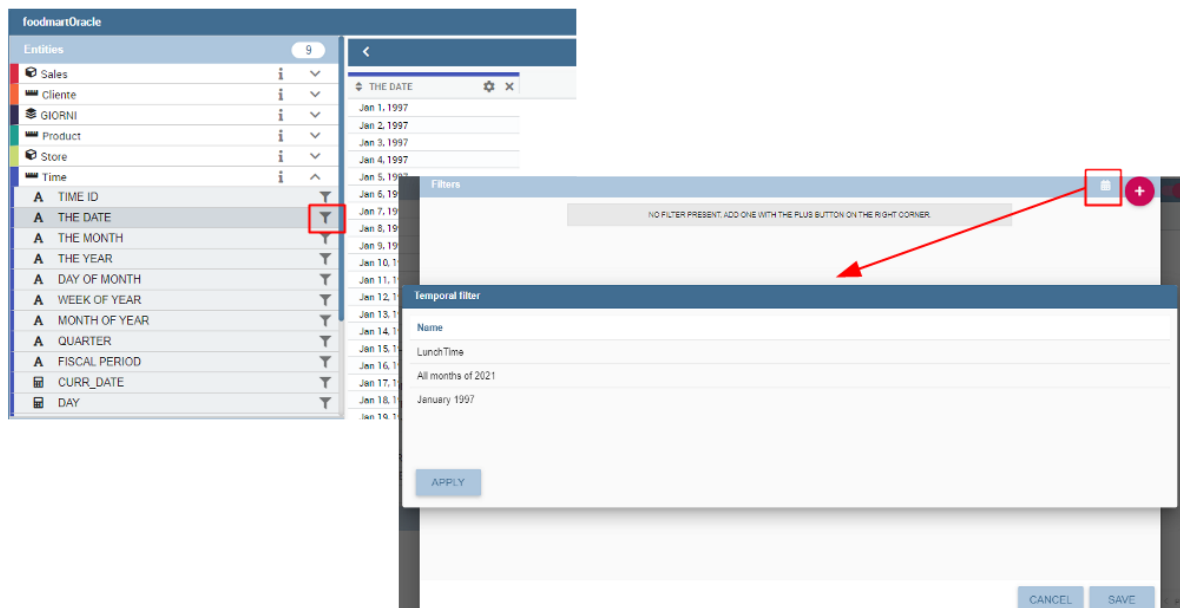


Fig. 1.287: Filtering date attribute: use a timespan.

After selecting a timespan and clicking on apply, the user has to insert the start and end date values. Save now to filter data accordingly.

1.6.2.1.1.4 Query Preview

The Smart automatically shows the preview of you query. The Advanced view contains a **Play** icon located in the top right corner of the panel, that opens a window with the results of the query. Just close the window to go back to the **Designer**.

In case of starting the QbE editor directly from a model **My Data > Models**, you can also save your query as a new dataset by clicking on the **Save** icon located in the top right corner of the page; the dataset would be reachable later from **My Data > Dataset**. Please note that this operation saves the *definition* of your query and not the snapshot of

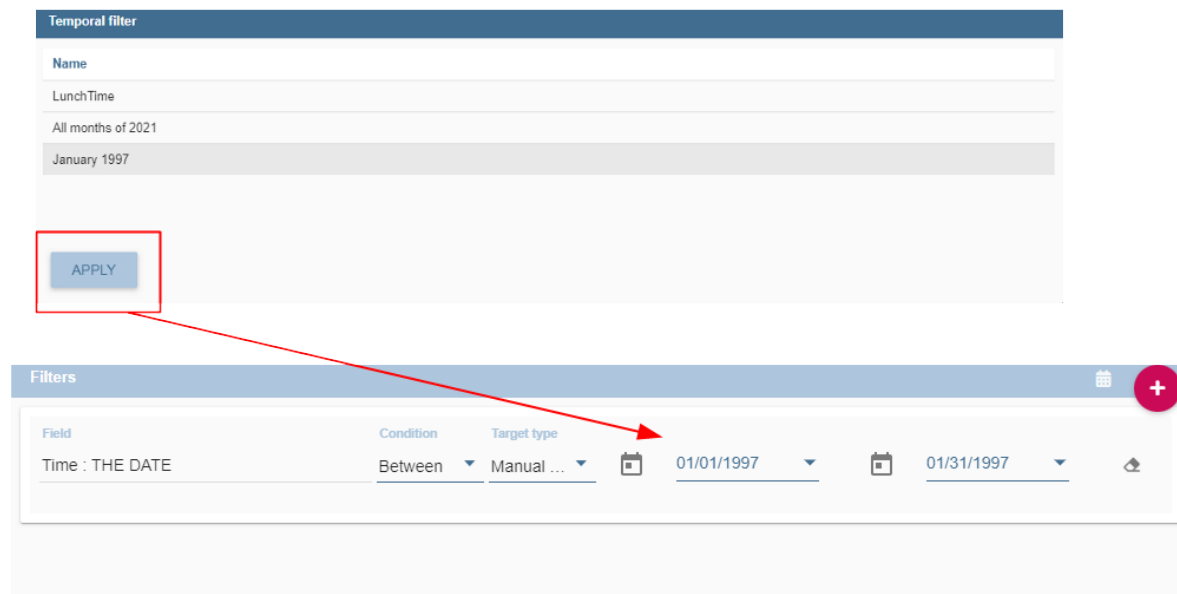


Fig. 1.288: Filtering date attribute: apply a timespan.

the query result. This means that every time you re-execute the saved dataset, a query on the database is performed to recover the updated data.

When saving your query as dataset, a pop up opens asking you to fill in some information, split in three tabs:

- **Details**, in this tab you set basic information for your dataset like its **Label**, **Name**, **Description**, **Category** and **Scope**.
- **Persistence**, you can persist your dataset, i.e., to write it on the default database. Making a dataset persistent may be useful in case dataset calculation takes a considerable amount of time. Instead of recalculating the dataset each time the documents using it are executed, the dataset is calculated once and then retrieved from a table to improve performance. You can also decide to schedule the persistence operation: this means that the data stored will be updated according to the frequency defined in the **scheduling** options.
- **Metadata**, contains the metadata associated to the fields involved in your query.

1.6.2.1.2 Advanced QbE functionalities

In this section we focus on advanced features, which can be comfortably managed by more expert users.

1.6.2.1.2.1 Spatial fields usage

Important: Enterprise Edition only

Spatial dimension is only available for Enterprise Edition with LI licence.

Preview			
COUNTRY	CITY	FNAME	LNAME
Canada	Burnaby	Alexandra	Wellington
Canada	Burnaby	Ana	Quick
Canada	Burnaby	Anastacio	Butcher
Canada	Burnaby	Ann	Pearce
Canada	Burnaby	Barbara	Serventi
Canada	Burnaby	Beth	Moore
Canada	Burnaby	Betty	McMenama
Canada	Burnaby	Candice	Ashe
Canada	Burnaby	Celeste	Triebel
Canada	Burnaby	Chris	Trigg
Canada	Burnaby	Cindy	Dodd
Canada	Burnaby	Cindy	Golden
Canada	Burnaby	Crystal	McDonald
Canada	Burnaby	Daniel	Waskey
Canada	Burnaby	Danielle	Hurtado
Canada	Burnaby	Daryl	Ives
Canada	Burnaby	Dave	Garner
Canada	Burnaby	David	Staisteven
Canada	Burnaby	Derek	Armstrong
Canada	Burnaby	Dorothy	Weir

< 1 2 3 ... 515 > 10281

Fig. 1.289: Preview wizard.

Saving QBE Dataset

Detail

Metadata

Persistence

Label *

Name *

Description

Scope *


Category *

CANCEL

SAVE

Fig. 1.290: Save qbe dataset.

The Qbe engine supports *spatial* queries through a set of operators (that return true or false) or a set of functions (that usually return a measure). This feature is only available for the Location Intelligence (LI) license and when data is stored in Oracle 12c databases. It is also fundamental that the Business Model has to be tagged as *geographical* model. You can refer to Meta Web Section to have details on how to set the geographical option using Knowage Meta.

In a BM with geographical dimensions enabled (by a technical user), the dimensions which has spatial fields are marked with the compass icon . Expanding the spatial dimension the list of fields is shown and there is no way to distinguish between geographical and non geographical attributes. Therefore the user has to be already aware of which fields have geometrical properties.

Entities14

Partenze

Arri

Compagnia

Tipologia Aeromobile

Flight status ACDM

Tempo giornaliero

Tempo orario

Aeroporto Arrivi

c Inizio validità

c Fine validità

c Codice ICAO

c Codice IATA

EntityFieldAliasGroupOrderAggregationShow field

Aeroporto ArriviCodice IATACodice IATANONECOUNT

Fig. 1.291: QbE spatial dimensions.

Also for a geographical dimension is possible to add fields to the query, including *Calculated* fields. This functionality is the same as shown before, as the three dots menu of the Advanced view contains the **Calculated field** option. Note that a wizard opens: you can use this editor to insert a new field obtained through a finite sequence of operation on the selected fields. In this case, the wizard offers a set of spatial functions that can be used in your formula.

Remember to select *SPATIAL* from the Category menu list to see all the spatial functions. Drag and drop the fields and your function(s) to the text editor and refer to the Oracle function description for a proper use.

In addition to the *SPATIAL* functions, the **Category** field provides some more options:

Calculated Field

Choose a name, select a type and add an expression for the calculated field. Drag and drop the functions and the fields from the list into the expression text area. Note that fields should be written the format \$F(fieldName)

Column Name	Type	Column Type
	String	Attribute

FIELDS

- Product id
- Brand name
- Product name

FUNCTIONS

- centroid(geom1, tol)
- distance(geom1, geom2, tol, unit)
- intersection(geom1, geom2, tol)
- length_spa(geom, tol, unit)
- relate(geom1, mask, geom2, tol)

Category
 SPATIAL

Calculated field
 Click on one of the listed functions to get more information about it

1

CANCEL

APPLY

Fig. 1.292: Calculated field wizard with spatial filters.

FIELDS

- Product id
- Brand name
- Product name

FUNCTIONS

- centroid(geom1, tol)
- distance(geom1, geom2, tol, unit)
- intersection(geom1, geom2, tol)
- length_spa(geom, tol, unit)
- relate(geom1, mask, geom2, tol)

Category
 SPATIAL

RELATE
 dataPreparation.relate

1

```
relate(geom1, mask, geom2, tol)
```

Fig. 1.293: Spatial functions.

- AGGREGATION functions,
- STRING functions,
- TIME functions,
- SQL functions,
- CUSTOM functions (if previously developed).

Time functions are only available for the Enterprise Edition with SI licence.

In addition to calculated fields, it is possible to filter on spatial fields using specific geometric operators

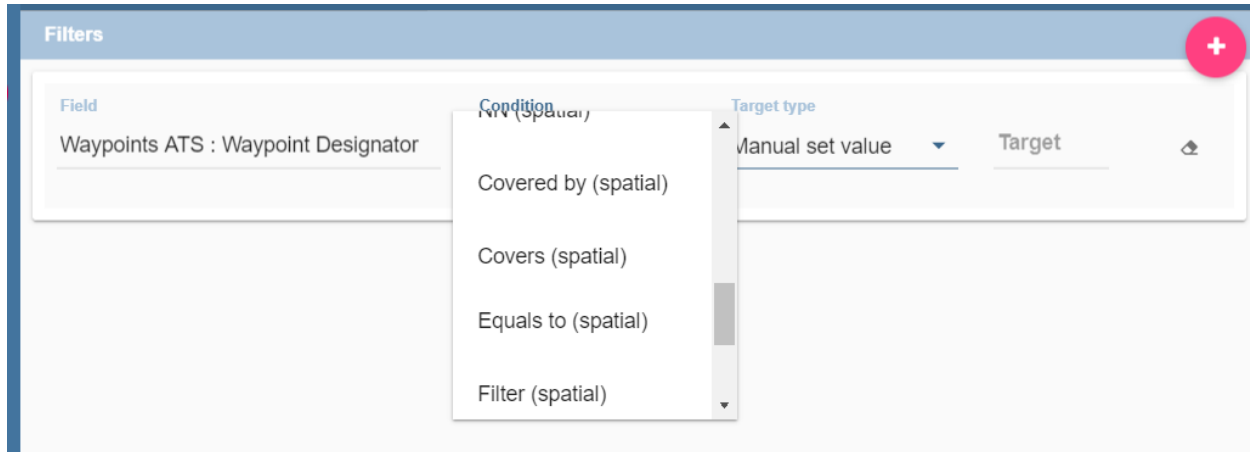


Fig. 1.294: Spatial filters.

1.6.2.1.2.2 Subqueries

The **QbE Engine** also supports the definition and usage of SQL subqueries to use within a filter in association to the **In/Not in** operator, as shown in the figure below. To create a subquery, click on **+** icon, on the right of **Derived entities**.

You can easily add fields and return to the main query clicking on **MAIN** link in the query editor toolbar.

To use the subquery inside the main query, simply choose **Subquery** from *Target type* and set the *Condition* to **(IN or NOT IN)**.

1.6.2.1.2.3 Parameters

The **QbE Engine** also supports the definition and usage of parameters that can be used to filter data using the QbE filter. To create a new parameter to be used as a filter inside the main query, click on **Parameters** from the three dots menu of the main query toolbar.

To use the parameter inside the main query, simply choose **Parameter** from *Target type* and choose parameter name from the **Target** list.

1.7 Create a new Registry

A Registry document allows users to write, cancel and modify items of a datamart. Knowage allows users to implement a Registry document through the **QbE Engine**. By the way it has a different graphical interface compared to the Qbe

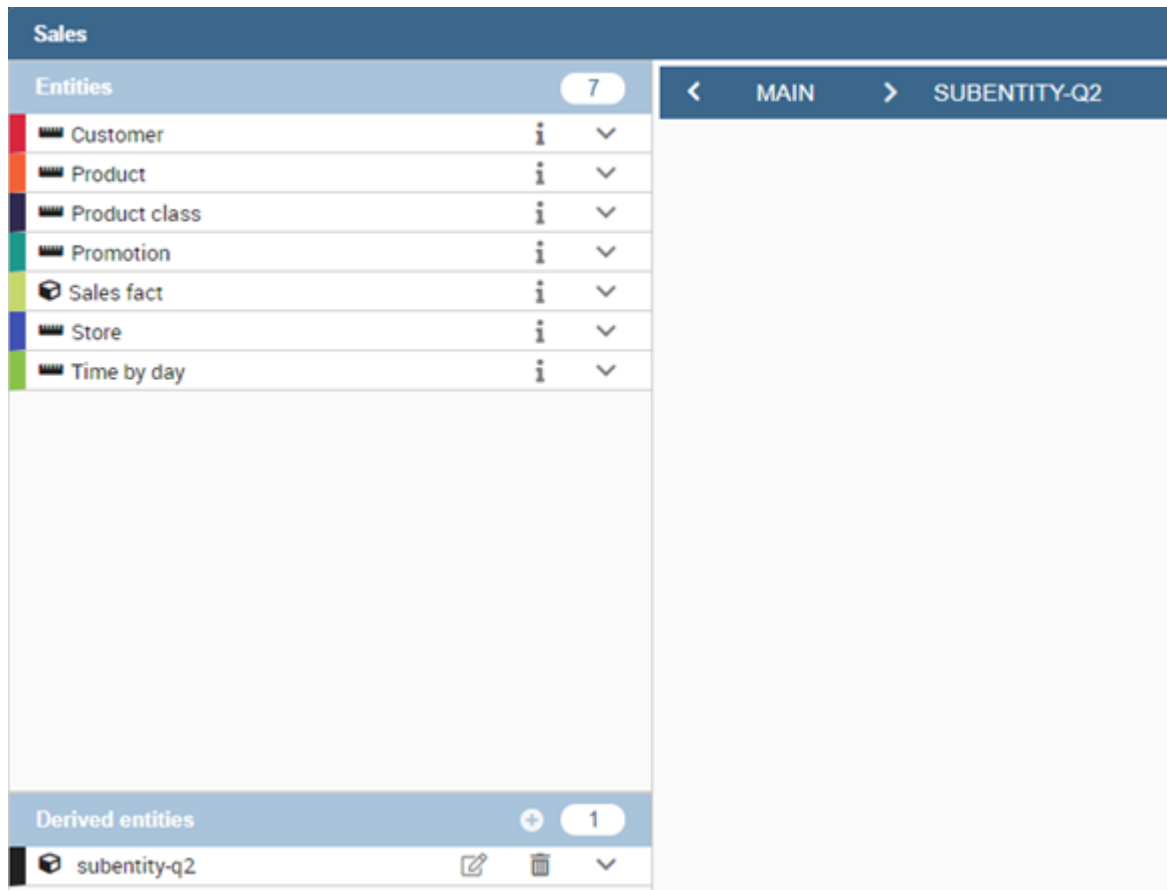


Fig. 1.295: QbE subquery view.

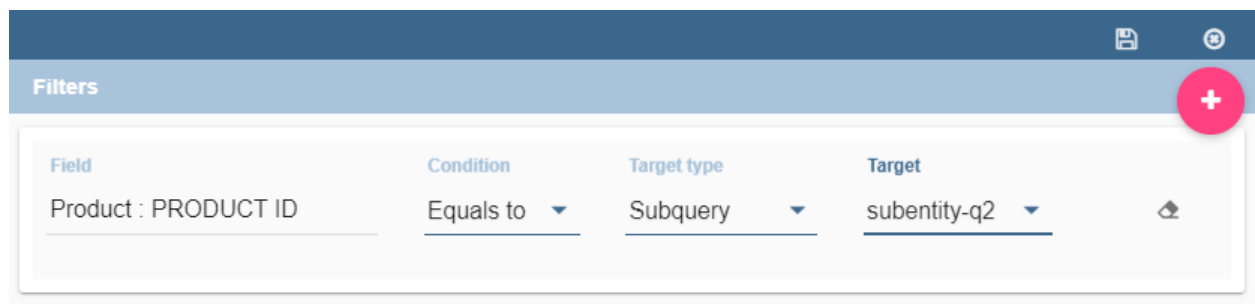


Fig. 1.296: QbE query: Use of a subquery in a filter.

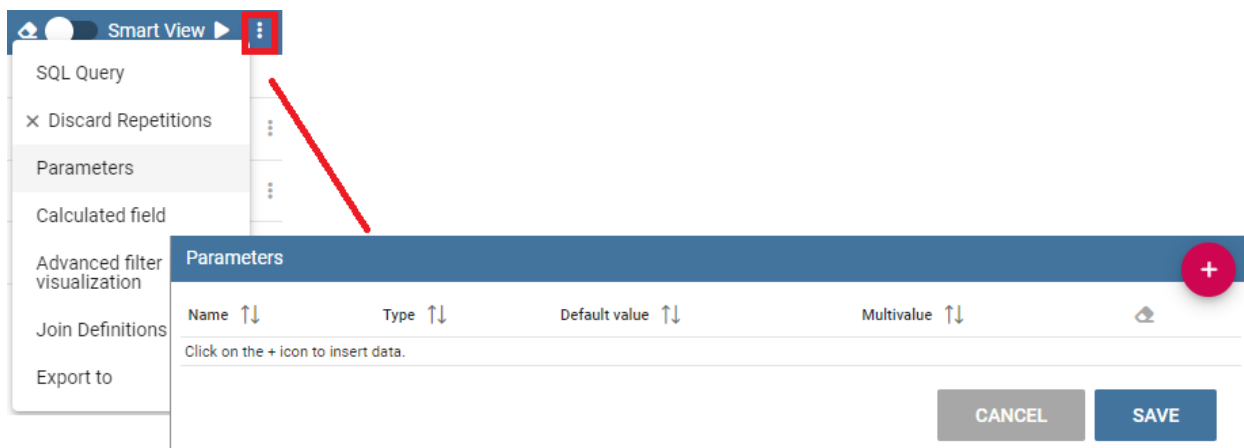


Fig. 1.297: QBE parameter view.

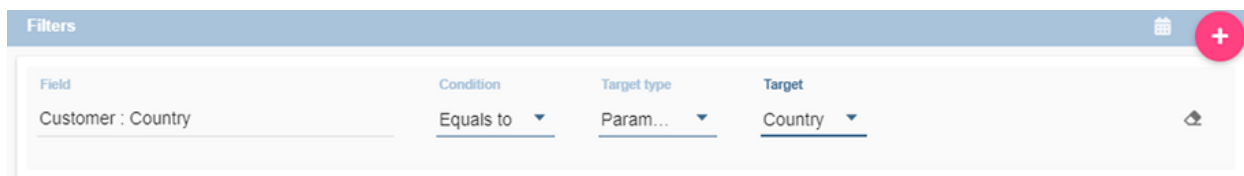




Fig. 1.298: QbE query: use of a parameter in a filter.


one. An example is given in the following figure. In next chapters we will see how to navigate a Registry document (*Registry features* paragraph) and how to create a new one (*Registry development* paragraph).


1.7.1 Registry features

The execution of a Registry document opens a plain table: the records are shown in rows and they can be browsed using the pagination available at the bottom of the window. We underline that it is possible to edit each item of the table if inside template that column is set to be editable. Just click on a cell you may wish to rearrange and type a string or a numeric value accordingly. Some examples are highlighted below.

Moreover, you can add new rows from scratch selecting the “Add row” button  in the header of last column. Insert in each cell the corresponding value: string, number or date. “Add row” button will be available if inside template there is a configuration: `<CONFIGURATION name=”enableAddRecords” value=”true”/>`

Vice versa, you can delete one or more rows using the “Trash” icon  in the last column. “Trash” button will be available if inside template there is a configuration: `<CONFIGURATION name=”enableDeleteRecords” value=”true”/>`

It is important to click on the “Save” button  to store the adjustments done in the datamart. “Save” button is available in Functionality bar, above table.

Furthermore you can use filters, if implemented, available in the Functionality bar. Click on the “Filter” icon  to run the functionality. Otherwise, click on the “Cancel” icon to clear the boxes off.

Note that, since records are displayed in a plain table, it is available a combobox (see figure below) which allows the user to visualize all fields related to the record of the previous cell and then change from one to another to get all data.

Testlink - Registry simple

Registry Document

Filters

Store type Sales city

FILTER

	store name	store type	store city	store state	has florist	has coffee bar	has video store	first opened date	store square foot	Foreign key - city
1	HQhjdjsa	My store type 2	Beverly Hills	CA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	09/05/2019	5,432	Newton
2	Store 11	àèiôu??	Portland	OR	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	09/16/1976	20,319	Portland
3	Store 11	Supermarket	Acapulco	Guerrero	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	01/08/1982	23,593	Acapulco
4	Store 124	Deluxe Supermarket	Hidalgo	Zacatecas	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	03/24/1968	30,584	Hidalgo
5	Store 13	Deluxe Supermarket	Salem	OR	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	04/01/1957	27,694	Salem
6	Store 15	Supermarket	Seattle	WA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	07/23/1969	21,215	Seattle
7	Store 16	Supermarket	Spokane	WA	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	08/23/1974	30,268	Spokane
8	Store 17	Deluxe Supermarket	Tacoma	WA	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	05/30/1970	33,858	Tacoma
9	Store 18	Mid-Size Grocery	Hidalgo	Zacatecas	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	06/28/1969	38,382	Hidalgo
10	Store 19	Deluxe Supermarket	Vancouver	BC	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	03/27/1977	23,112	Vancouver
11	Store 2	Small Grocery	Bellingham	WA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	04/02/1970	28,206	Bellingham
12	Store 20	Mid-Size Grocery	Victoria	BC	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	02/05/1980	34,452	Victoria
13	Store 21	Deluxe Supermarket	San Andres	DF	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	02/07/1986		San Andres
14	Store 22	Small Grocery	Walla Walla	WA	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	01/23/1951		Walla Walla
15	Store 23	Mid-Size Grocery	Yakima	WA	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	07/16/1977		Yakima

1 to 15 of 29 < > Page 1 of 2 > > >

Fig. 1.299: Example of Registry document.

Testlink - Registry simple

Registry Document

Filters

Store type Sales city

FILTER

	store name	store type	store city	store state	has florist	has coffee bar	has video store	first opened date	store square foot	Foreign key - city
1	HQhjdjsa	My store type 2	Beverly Hills	CA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	09/05/2019	5,432	Newton
2	Store 11	àèiôu??	Portland	OR	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	09/16/1976	20,319	Portland
3	Store 11	Supermarket	Acapulco	Guerrero	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	01/08/1982	23,593	Acapulco
4	Store 124	Deluxe Supermarket	Hidalgo	Zacatecas	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	03/24/1968	30,584	Hidalgo
5	Store 13	Deluxe Supermarket	Salem	OR	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	04/01/1957	27,694	Salem
6	Store 15	Supermarket	Seattle	WA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	07/23/1969	21,215	Seattle
7	Store 16	Supermarket	Spokane	WA	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	08/23/1974	30,268	Spokane
8	Store 17	Deluxe Supermarket	Tacoma	WA	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	05/30/1970	33,858	Tacoma
9	Store 18	Mid-Size Grocery	Hidalgo	Zacatecas	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	06/28/1969	38,382	Hidalgo
10	Store 19	Deluxe Supermarket	Vancouver	BC	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	03/27/1977	23,112	Vancouver
11	Store 2	Small Grocery	Bellingham	WA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	04/02/1970	28,206	Bellingham
12	Store 20	Mid-Size Grocery	Victoria	BC	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	02/05/1980	34,452	Victoria
13	Store 21	Deluxe Supermarket	San Andres	DF	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	02/07/1986		San Andres
14	Store 22	Small Grocery	Walla Walla	WA	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	01/23/1951		Walla Walla
15	Store 23	Mid-Size Grocery	Yakima	WA	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	07/16/1977		Yakima

1 to 15 of 29 < > Page 1 of 2 > > >

Fig. 1.300: Editing table cells.

Testlink - Registry simple

Registry Document

Filters

Store type Sales city

FILTER

	store name	store type	store city	store state	has florist	has coffee bar	has video store	first opened date	store square foot	Foreign key - city
1	HQhjdjsa	My store type 2	Beverly Hills	CA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	09/05/2019	5,432	Newton
2	Store 11	àèiôu??	Portland	OR	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	09/16/1976	20,319	Portland
3	Store 11	Supermarket	Acapulco	Guerrero	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	01/08/1982	23,593	Acapulco
4	Store 124	Deluxe Supermarket	Hidalgo	Zacatecas	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	03/24/1968	30,584	Hidalgo
5	Store 13	Deluxe Supermarket	Salem	OR	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	04/01/1957	27,694	Salem
6	Store 15	Supermarket	Seattle	WA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	07/23/1969	21,215	Seattle
7	Store 16	Supermarket	Spokane	WA	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	08/23/1974	30,268	Spokane
8	Store 17	Deluxe Supermarket	Tacoma	WA	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	05/30/1970	33,858	Tacoma
9	Store 18	Mid-Size Grocery	Hidalgo	Zacatecas	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	06/28/1969	38,382	Hidalgo
10	Store 19	Deluxe Supermarket	Vancouver	BC	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	03/27/1977	23,112	Vancouver
11	Store 2	Small Grocery	Bellingham	WA	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	04/02/1970	28,206	Bellingham
12	Store 20	Mid-Size Grocery	Victoria	BC	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	02/05/1980	34,452	Victoria
13	Store 21	Deluxe Supermarket	San Andres	DF	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	02/07/1986		San Andres
14	Store 22	Small Grocery	Walla Walla	WA	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	01/23/1951		Walla Walla
15	Store 23	Mid-Size Grocery	Yakima	WA	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	07/16/1977		Yakima

1 to 15 of 29 < > Page 1 of 2 > > >

Fig. 1.301: Adding a new row.

Filters

Store type Sales city ▼

FILTER

Fig. 1.302: Functionality bar.

Testlink - Registry simple

Registry Document

Filters

Store type Sales city ▼

FILTER

	store name	store type	store city	store state	has florist	has coffee bar	has video store	first opened date	store square foot	Foreign key - city
3	Store 11	Mid-Size Grocery	terro	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	01/08/1982	23,593	Acapulco
4	Store 124		itecas	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	03/24/1968	30,584	Hidalgo
5	Store 13			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	04/01/1957	27,694	Salem
6	Store 15	Gourmet Supermarket		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	07/23/1969	21,215	Seattle
7	Store 16			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	08/23/1974	30,268	Spokane
8	Store 17	Deluxe Supermarket		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	05/30/1970	33,858	Tacoma
9	Store 18			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	06/28/1969	38,382	Hidalgo
14	Store 22			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	01/23/1951		Walla Walla
15	Store 23	Mid-Size Grocery	Yakima	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	07/16/1977		Yakima

1 to 15 of 25 < > Page 1 of 2 >

Fig. 1.303: Select one field from a combobox.

1.7.1.1 JPivot Registry characteristics

It is possible to implement also a JPivot Registry document. The graphical features are very similar to the ones exposed in *Registry development* paragraph. An example is given below.

Testlink - Registry pivot table

Registry Document

SAVE

Filters

Store type

FILTER

store country	store state	store city	store type	store name	first opened date	store square foot
			测试店	测试店	08/16/1983	20
			Deluxe Supermarket	测试店	03/01/2020	33
			Gourmet Supermarket	测试店	01/31/2020	30
			Mid-Size Grocery	TestStoreName	05/18/2020	45
			My store type 2	My Store2		0
				Mmm	02/03/2019	0
			Supermarke	Store for test	09/05/2019	5,432
				Store 8		0
			Supermarket	9999	09/05/2019	0
				Store for test	04/16/2020	5,432
Canada	BC	Vancouver	Deluxe Supermarket	Store 19	03/27/1977	23,112
	Victoria	Mid-Size Grocery	Store 20	02/05/1980	34,452	
Mexico	DF	Mexico City	Mid-Size Grocery	Store 9	03/18/1955	36,509
	San Andres	Deluxe Supermarket	Store 21	02/07/1986	0	

1 to 15 of 25 < > Page 1 of 3 >

Fig. 1.304: Example of Jpivot Registry document.

In this case the table shows columns organized in a hierarchical way and a grouping function is implemented. From the left to the right the columns contain fields at different detail levels. The last column in our example in the figure above contains numeric data. Such a field is grouped at the “country” level. The grouping level depends on the configurations made on template building.

In the JPivot instance it is not allowed to add, modify or cancel rows. Furthermore, it is not permitted to edit cells which contain string items while the numeric ones are still changeable. If implemented, filters are still available.

1.7.2 Registry development

To create a Registry document there must be available a datamart schema on Knowage Server. Then you must edit an XML template. The latter is very similar to the one produced under the Qbe development but in this case you must add an appropriate tag. Indeed, if the template file has the **<REGISTRY>** tag the engine shows data in registry modality; namely it shows a table whose rows are manageable by end users by adding, editing or deleting them.

Here we exhibit a possible syntax for a Registry document.

Listing 1.19: Example (a) of template for Registry.

```

1  <?xml version="1.0" encoding="windows-1250"?>
2  <QBE>
3      <DATAMART name="RegFoodmartModel" />
4      <REGISTRY>
5          <ENTITY name="it.eng.knowage.meta.regfoodmartmodel.Product">
6              <FILTERS>
7                  <FILTER title="Class" field="product_subcategory" presentation="COMBO" />
8                  <FILTER title="Product name" field="product_name" presentation="COMBO" />
9              </FILTERS>
10             <COLUMNS>
11                 <COLUMN field="product_id" unsigned="true" visible="false" editable="false
12                 ↪ " format="####" />
13                 <COLUMN field="product_name" title="Product name" size="200"
14                 editor="MANUAL" sorter="ASC"/>
15                 <COLUMN field="product_subcategory" title="Class" size="200"
16                 ↪ product_class" />
17                 <COLUMN field="SKU" title="SKU" size="200" editor="MANUAL" />
18                 <COLUMN field="gross_weight" title="Gross weight" size="200" editor="MANUAL
19                 ↪ " />
20                 <COLUMN field="net_weight" title="Net weight" size="200" editor="MANUAL" />
21             </COLUMNS>
22             <CONFIGURATIONS>
23                 <CONFIGURATION name="enableDeleteRecords" value="true"/>
24                 <CONFIGURATION name="enableAddRecords" value="true"/>
25                 <CONFIGURATION name="isPkAutoLoad" value="true"/>
26             </CONFIGURATIONS>
27         </ENTITY>
28     </REGISTRY>
29 </QBE>

```

In particular, we give some details for each tag and main attributes.

- **ENTITY**: the entity name as in the model. It must be the fully-qualified name of the class representing your registry in the model;
- **FILTERS**: possibility to define filters by specifying the title, the field (among shown columns) and the type among COMBO, MANUAL or DRIVER: in this last case user has also to specify the analytical driver that take this filter's value;
- **COLUMNS**: columns list specifying:
 - **field name**: the reference to the field identifier into the model;
 - **title**: the title of the column shown (optional);
 - **visible**: the visibility of the column (optional, default true);
 - **editable**: the editability of the column (optional, default true);
 - **color and format for numbers**: optional;
 - **size**: the width of the column (optional);

- **editor**: the editor. Default type is free-text for simple column (not FK values), but for date is possible to show the picker through the type PICKER. The format option specifies the format date;
- **subEntity**: if the column is a reference key, the user can specify the subentity referred and the foreign key name. This value must be equals to the name of the relationship object created in the model. The field shown will be of the entity referred and will be shown as COMBO if editable;
- **foreignKey**: if the subEntity property is set, foreignKey property must be set with the name of the foreign key (to lower case);
- **dependsFrom**: if the column content is logically correlated to other registry's column, it is possible to specify this logic through this parameter. DependsFrom identifies the field name on which it depends (Optional);
- **dependsFromEntity**: usable only with dependsFrom parameter. It defines a different entity to resolve the correlation (optional);
- **orderBy**: is used in case of foreign key. The combo box is ordered by the column here indicated, by default is the column extracted (optional);
- **infoColumn**: if true ignore the column when inserting or updating the record (optional);
- **defaultValue**: defines the default value for the field; if the user does not set any value for this field during insertion, this value will be set automatically (optional, not allowed if subEntity or foreignKey property is set). For date fields, the correct pattern is "yyyy-MM-dd'T'HH:mm:ss.xxx'Z'".

We stress that it is mandatory to point at one datamart table using a column with a numeric key. The code line is highlighted in figure below. While, if not elsewhere specified, a descriptive column will be displayed by default.

Listing 1.20: Pointing at a numerical column.

```

1 <COLUMNS>
2   <COLUMN field="store_id" visible="false" editable="false" />
3   ...
4 </COLUMNS>

```

Still referring to the code above, we underline that the “product_subcategory” field is used as a subcategory. It belongs in fact to another table. In this case it is enough to add the attributes: subEntity=”rel_product_class_id_in_product_class” foreignKey=”rel_product_class_id_in_product_class”.

1.7.2.1 Filters

Listing 1.21: Filter definition example.

```

1 <FILTERS>
2   <FILTER title="Store type" field="store_type" presentation="MANUAL" />
3   <FILTER title="Sales city" field="sales_city" presentation="COMBO" />
4   <FILTER title="Sales first opened date" field="first_opened_date" static="true" visible="true"
5   ↪filterValue="29/05/2020 02:00:00.0" />
6 </FILTERS>

```

Filter definition allows to set different properties:

- **title**: the title of the filter;
- **field**: the reference to the field identifier into the model;
- **presentation**: COMBO/DRIVER/MANUAL (optional if static=”true”);
- **visible**: the visibility of the filter (optional, default false);
- **static**: true/false. Set this property if you want to limit filter value to a specific value (optional);

- **filterValue**: the specific value you want to set for the filter (mandatory if static="true"). For date fields, the correct pattern is " %d/%m/%Y %h:%i:%s".

1.7.2.2 Analytical driver

Registry filtering by analytical driver is possible using DRIVER value for presentation property in filter TAG. Registry template must contains FILTERS tag. Below an example of configuration for a driver named "UNIT_SALES_AD" insisting on the column "UNIT_SALES".

Listing 1.22: Pointing at a numerical column.

```

1 <FILTERS>
2   <FILTER title="UNIT_SALES_AD_title" field="UNIT_SALES" presentation="DRIVER" driverName="UNIT_SALES_AD" />
3 </FILTERS>

```

1.7.2.3 Profile attributes

Another way to filter registry content is using profile attributes. If you want to use profile attributes to filter values you have to follow these steps:

- Create a profile attribute (if necessary) from the Manage Profile Attributes menu
- Associate the profile attribute with the column during model creation

This way, your data will be filtered by this attribute (if not empty) both when viewing data and when inserting or updating records.

1.7.2.3.1 Multivalue

If your profile attribute is a multivalue one, you have to:

- set IN clause as "Profile attribute Filter Type" during model's creation
- set profile attribute values respecting this format 'value1','value2',...,'valueN' or {,{value1,value2,...,valueN}}.

In this way, profile attribute value will be treated as a list of values and filter will be applied with this criteria.

1.7.2.4 JPivot Registry instance

The Registry instance allows to develop also a Jpivot table. See the last figure (above) to have an idea while the syntax example is given in the next code:

Listing 1.23: Example (b) of template code for Registry.

```

1 <QBE>
2   <DATAMART name="foodmart" />
3   <REGISTRY pagination = "false" summaryColor="#00AAAA">
4     <ENTITY name="it.eng.knowage.meta.foodmart.Store">
5       <FILTERS>
6         <FILTER title="Store Type" field="store_type" presentation="COMBO" />
7       </FILTERS>
8       <COLUMNS>
9         <COLUMN field="store_id" visible="false" editable = "false" />
10        <COLUMN field="store_country" title="store country" visible="true"
11        type="merge" editable = "false" sorter = "ASC" summaryFunction="sum" />
12        <COLUMN field="store_state" title="store state" visible="true"
13        type=" merge" editable = "false" sorter = "ASC" />

```

(continues on next page)

(continued from previous page)

```

14         <COLUMN field="store_city" title="store city" visible="true"
15         type="merge" editable="false" sorter="ASC" />
16         <COLUMN field="store_type" title="store type" type="merge" sorter="ASC" />
17         <COLUMN field="store_number" title="Number" size="150"
18         editable="true" format="#####" color="#f9f9f8" type="measure"/>
19     </COLUMNS>
20     <CONFIGURATIONS>
21         <CONFIGURATION name="enableDeleteRecords" value="true"/>
22         <CONFIGURATION name="enableAddRecords" value="true"/>
23     </CONFIGURATIONS>
24 </ENTITY>
25 </REGISTRY>
26 </QBE>

```

Note that to activate the JPivot modality it is important to add the attribute type="merge" and have at least one numeric field. Furthermore the selected column fields must be hierarchically structured.

1.7.3 Logging & auditing

The Registry engine is logging changes performed by users when interacting with Registry documents (insertions/updates/deletions of entries).

By default, the engine is logging messages such as

```

01 feb 2021 11:40:49,750: User <name of the user> is performing operation <INSERTION/UPDATE/DELETION> on
↪ entity <name of the entity> from model <model name> for record: old one is ..., new one is ..., number of
↪ changes is ...

```

into the TOMCAT_HOME/logs/knowageQbeEngineAudit.log file.

In case you want those information to be stored into a database table (for analytical and visualization purposes), you have to create it and then to configure the engine logging system accordingly, following the below example based on MySQL.

Let's create a table:

```

1 CREATE TABLE `LOG_REGISTRY` (
2   `AUDIT_ID` INT NOT NULL AUTO_INCREMENT,
3   `AUDIT_DATETIME` DATETIME NULL,
4   `AUDIT_OPERATION` VARCHAR(45) NULL,
5   `AUDIT_USER` VARCHAR(100) NULL,
6   `AUDIT_CHANGES_NO` INT NULL,
7   `ENTITY_NAME` VARCHAR(100) NULL,
8   `MODEL_NAME` VARCHAR(100) NULL,
9   `ATTRIBUTES_OLD` TEXT NULL,
10  `ATTRIBUTES_NEW` TEXT NULL,
11  PRIMARY KEY (`AUDIT_ID`));

```

then edit TOMCAT_HOME/webapps/knowageqbeengine/WEB-INF/classes/log4j.properties and add:

```

1 # Define the SQL appender
2 log4j.appender.sql=it.eng.spagobi.utilities.logging.Log4jJNDIAppender
3 # JNDI connection to be used
4 log4j.appender.sql.jndi=java:comp/env/jdbc/knowage
5 # Set the SQL statement to be executed.
6 log4j.appender.sql.sql=INSERT INTO LOG_REGISTRY (AUDIT_DATETIME,AUDIT_OPERATION,AUDIT_USER,AUDIT_CHANGES_NO,
↪ ENTITY_NAME,MODEL_NAME,ATTRIBUTES_OLD,ATTRIBUTES_NEW) VALUES (now(),'%X{operation}','%X{userId}','%X
↪ {variations}','%X{entityName}','%X{modelName}','%X{oldRecord}','%X{newRecord}')
7 # Define the xml layout for file appender
8 log4j.appender.sql.layout=org.apache.log4j.PatternLayout
9

```

(continues on next page)

(continued from previous page)

```

10 log4j.logger.it.eng.qbe.datasource.jpa.audit.JPAPersistenceManagerAuditLogger=INFO, FILE_AUDIT
11 log4j.additivity.it.eng.qbe.datasource.jpa.audit.JPAPersistenceManagerAuditLogger=false

```

pay attention to the JNDI name (in case you created the table within Knowage metadata database, then `java:comp/env/jdbc/knowage` is fine) then restart Knowage server: this way, when user is interacting with a registry document, the LOG_REGISTRY (as per the SQL script above) table will contain:

- AUDIT_DATETIME: the date and time when the operation was performed
- AUDIT_OPERATION: one of the following values: INSERTION/UPDATE/DELETION
- AUDIT_USER: the user who performed the operation
- AUDIT_CHANGES_NO: number of attributes that were actually changed in case of an UPDATE, null otherwise
- ENTITY_NAME: name of the modified entity type
- MODEL_NAME: name of the business model
- ATTRIBUTES_OLD: previous attributes state in case of an UPDATE or DELETION
- ATTRIBUTES_NEW: new attributes state in case of an INSERTION or UPDATE

1.8 Create location Intelligence

1.8.1 Create GIS document

Location intelligence is based on the idea that geographical spaces are a particular analytical dimension in the BI domain. It is based on:

- the geographical representation of data,
- interaction with GIS systems,
- spatial data,
- spatial operators.

Location Intelligence usually guarantees:

- an immediate perception of a phenomena distribution over a geographical area,
- interactivity,
- multivariate analysis,
- temporal snapshots.

Location Intelligence is becoming widely used, mostly thanks to the emergence of location services such as Google Maps. This domain is very easy to use for all kinds of users, usually analysts and operational profiles. By contrast, its management is not as easy, especially if it implies an internal management of the geographical data base.

1.8.1.1 Basic concepts

The term Location Intelligence refers to all those processes, technologies, applications and practices capable to join spatial data with business data, in order to gain critical insights, to better support decisional processes and to optimize business activities.

At the technological level, this correlation is the result of the integration between the software systems that manage these two heterogeneous types of data: geographic information systems (GIS), which manage spatial data, and Business Intelligence systems (BI), which manage business data. This integration gives rise to new technological tools supporting decision-making processes, and the analysis on those business data that are directly or indirectly related to a geographic dimension.

Location Intelligence applications significantly improve the quality of users' analysis based on a geographic dimension. Indeed, a Data Warehouse (DWH) almost always included such information. By representing the geographic distribution of one or more business measures on interactive thematic maps, users can quickly identify patterns, trends or critical areas, with an effectiveness that would be unfeasible using traditional analytical tools.

1.8.1.2 More on GIS and Spatial Data

1.8.1.2.1 Spatial Data

The term *spatial data* refers to any kind of information that can be placed in a real or virtual geometric space. In particular, if the spatial data is located in a real geometric space — which is a geometric space that models the real space — it can be defined as *geo-referenced data*.

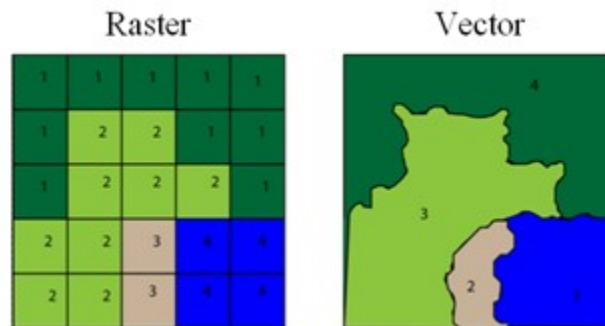


Fig. 1.305: A base layer in raster and vector format.

Spatial data are represented through graphical objects called maps. Maps are a portrayal of geographic information as a digital image file suitable for display on a computer screen.

According to the *Open Geospatial Consortium* (OGC) definition, a map is made of overlapping *layers*: a *base layer* in raster format (e.g. satellite photo) is integrated with other layers (*overlays*) in vector format. Each overlay is made of homogeneous spatial information, which models a same category of objects, called *features*.

A feature is called *geographic feature* when the constituting objects are abstractions of real-world physical objects and can be located univocally within a referenced coordinate system, according to their relative position.

A feature includes:

- a set of attributes that describes its geometry (vector encoding). Geometric attributes must describe its relative shape and position in an unambiguous way, so that the feature can be properly drawn and located on the map, according to the other features of the layers.
- a set of generic attributes related to the particular type of physical object to be modeled. Generic attributes are not defined: they vary according to the type of abstraction that users want to give to each real-world physical object.

There is a wide range of standards that can be used for the vector encoding of spatial data (e.g. GeoJSON, GML, Shape File, etc.). Most geographic information systems can perform the needed conversions among various encodings.

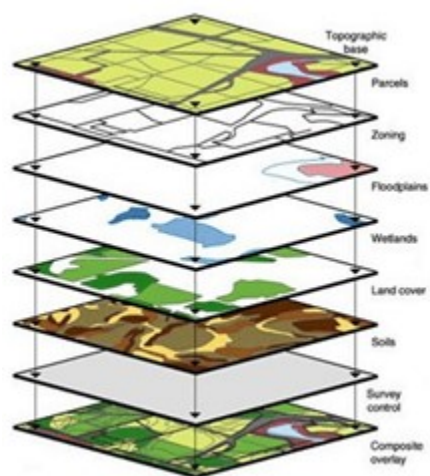


Fig. 1.306: Overlapping layer.

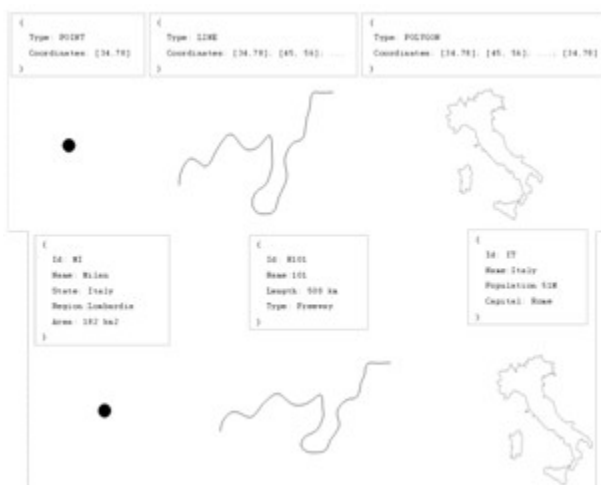


Fig. 1.307: Examples of feature.

1.8.1.2.2 GIS

Geographic Information Systems (GIS) provide a set of software tools designed to capture, store, extract, transform and display spatial data. Therefore, the term GIS refers to a set of technological components that manage the spatial data during its whole life cycle, starting from the capture of the data up to its representation and re-distribution.

From a logical point of view, the key functionalities of a GIS do not differ from those of a BI system. Both systems are characterized by some specific components supporting the effective storage of data, some others supporting their manipulation, their re-distribution or their visualization. On the other hand, the implementation of these functionalities deeply differs between GIS and BI systems, since they deal with two different types of data (alphanumeric and spatial data).



Fig. 1.308: Definition of GIS, BI, spatial data and business data.

Unlike the market of BI suites, the market of GIS is characterized by a wide spread of open standards, adopted by all main vendors, which regulate the interaction among the various components of the system at all architectural levels.

Note: Open Geospatial Consortium (OGC)

The most important International organization for standardization in the GIS domain is the Open Geospatial Consortium (OGC), involving 370 commercial, governmental, non-profit and research organizations. Read more at www.opengeospatial.org.

As for the integration between GIS and BI systems, the OGC has defined two main standards supporting the re-distribution of the spatial data:

- the *Web Map Service* (WMS). It describes the interface of services that allow to generate maps in a dynamic way, using the spatial data contained in a GIS.
- the *Web Feature Service* (WFS). It describes the interface of services that allow to query a GIS, in order to get the geographic features in a format that allows their transformation and/or spatial analysis (e.g. GML, GeoJson, etc.).

Note: WMS and WFS standards for spatial data distribution

Full documentation about the WMS and WFS standards can be found at www.opengeospatial.org/standards/wms and www.opengeospatial.org/standards/wfs.

Knowage suite offers an engine supporting the Location Intelligence analytical area, the **GEOReport Engine**, generating thematic maps.

1.8.1.3 Analytical document execution

Let's have a look on the user interface of Knowage Location Intelligence features.

Figure below provide an example of a BI analysis carried out thanks to map. In our example, the colour intensity of each state shows proportionally increases according to the value of the indicator selected. States who have no record connected are not coloured at all.

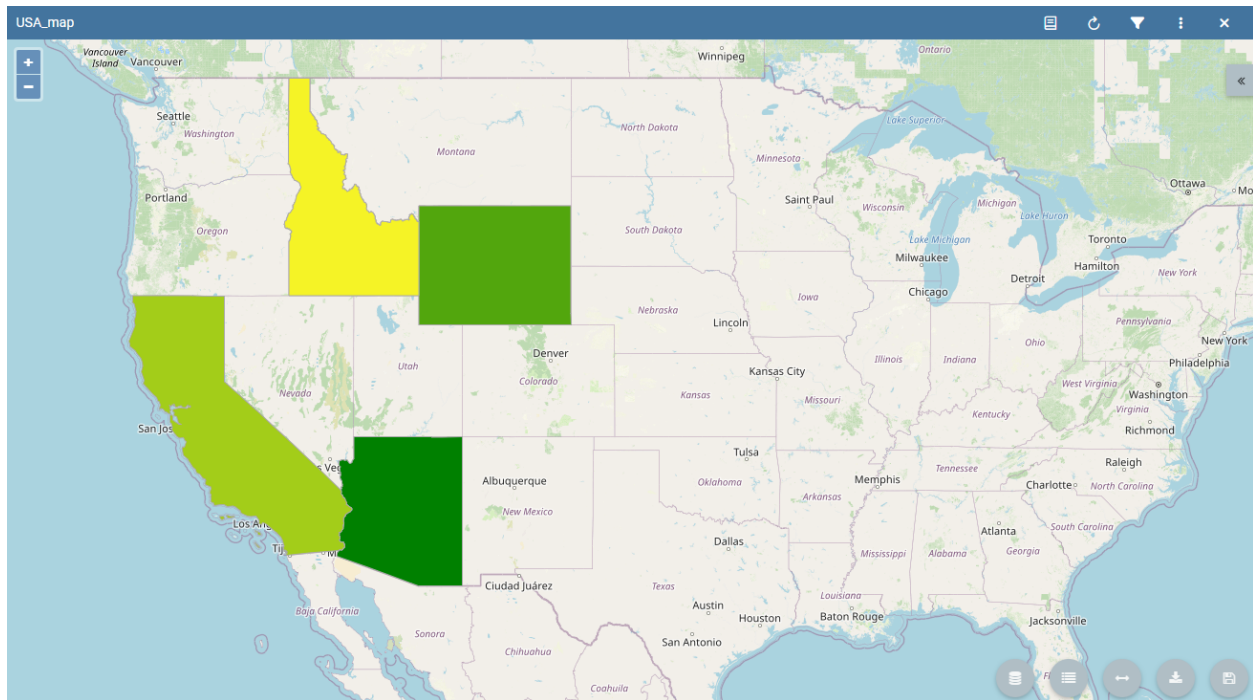


Fig. 1.309: Example of GIS document. USA sales per store

Click on the arrow on the top right to open the Location Intelligence options panel. Here you can choose the **Map Type**, the indicators to be displayed on the map and you can enter filters.

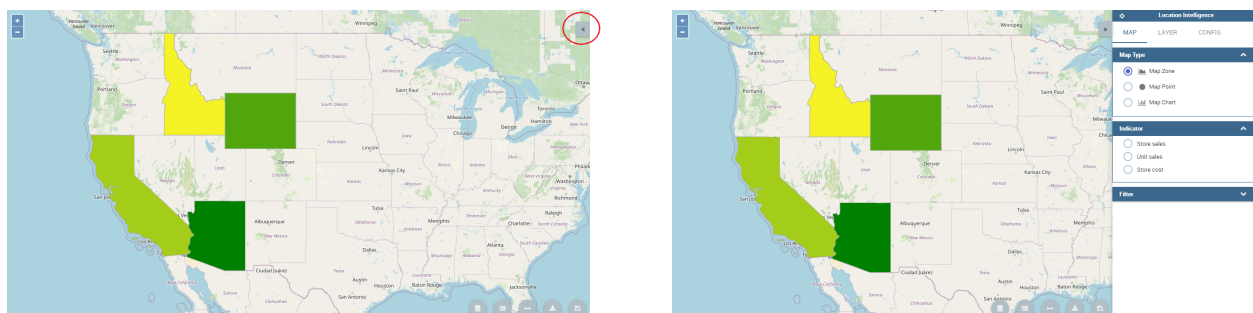


Fig. 1.310: Arrow button (left), Location Intelligence options panel (right) .

The **Map Type** available are:

- **Map Zone:** the different map zone are filled with different colour range according to the indicator values

- **Map Point:** the indicator values are displayed by points with different radius. A bigger radius means a higher indicator's value.
- **Map Chart:** thanks to this visualization type you can compare more than one indicators simultaneously. Choose which indicators compare among the available ones. You have to mark them in the **indicator** panel area to visualize them. The charts appears on the map displaying the selected indicators' values.

These three typologies of data visualization on map are compared below.

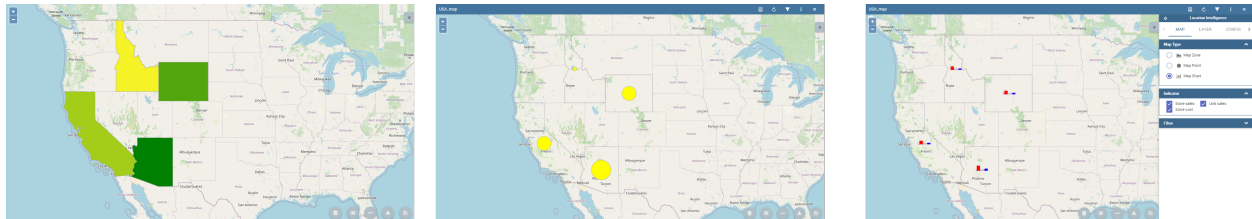


Fig. 1.311: Map Zone (left), Map Point (center) and Map Chart (right).

Now you can add extra layers on the default one. Switch to the **layer** tab of the Location Intelligence options panel.

Here click on the plus button near the **Layer** section and choose the layers you want to add. Mark them once added in the Location Intelligence area in the Layer box and the selected layer are displayed.

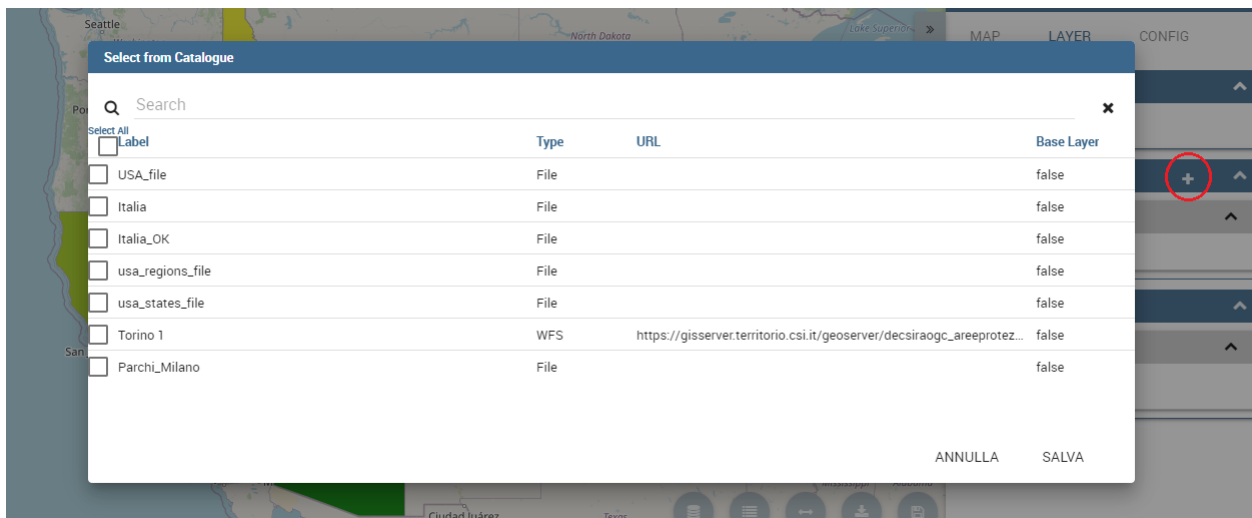


Fig. 1.312: Steps for layer adding

In our example we upload some waypoints, you can see the results obtained in next figure.

Now let's focus on **Configuration** tab of Location Intelligence panel option. Here you can set some extra configurations. Let's have a look on them for each data visualization typology.

For the **Map Zone** you can set:

- **Method:** the available ones are quantiles or equal intervals. If you choose quantiles data are classified into a certain number of classes with an equal number of units in each class. If you choose equal intervals the value are divided in ranges for each class, the classes are equal in size and their number can be set. The entire range of data values (max - min) is divided equally into classes however many classes have been chosen.
- **N° of classes:** the number of intervals in which data are divided.
- **Range colors:** you can choose the first and the last color of the range. For both of them you can use a color pixel by clicking on the coloured square. An example is provided below.

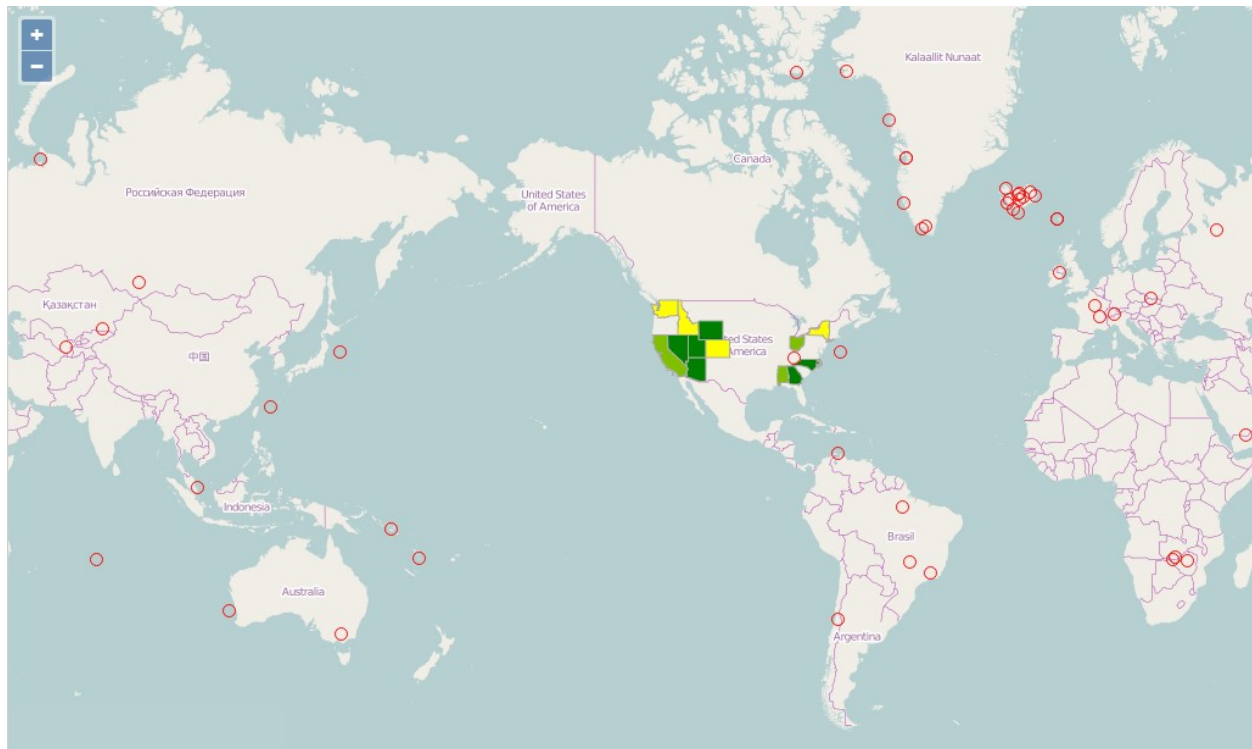


Fig. 1.313: Map with two layers

For the **Map Point** you can set:

- **Color:** the color of the circle.
- **Min/Max value:** the minimum and the maximum circles radius.

For the **Map Chart** you can set the color of each chart's bar.

We can conclude our overview on GIS document describing the buttons located at the bottom right corner, you can see them underlined in the following figure. From the left to the right this buttons can be used for: see the preview of the linked dataset, have a look at the legend, compute a measure of an area of the map, do the .pdf export of the map and save the map.

1.8.1.3.1 Extra functionalities

Let's come back to Location Layer main tab and focus on the **Select Mode** area. If cross navigation has been set you find two options: **Identify** and **Cross navigation**.

Selecting **Cross Navigation** the **Spatial Item** tab appears. In this tab you can configure your selection. To make your selection press CTRL key and choose the area on the map with the mouse. If you choose **near**, the features in the Km set are selected. If you choose **intersect**, the features which borders intersect your designed area. If you choose **inside**, only the features completely inside your area of selection are considered for the cross navigation.

When the selection is made, a box appears. In this box you find cross navigation information. The number of features selected and a button to perform the cross navigation with the active selection.

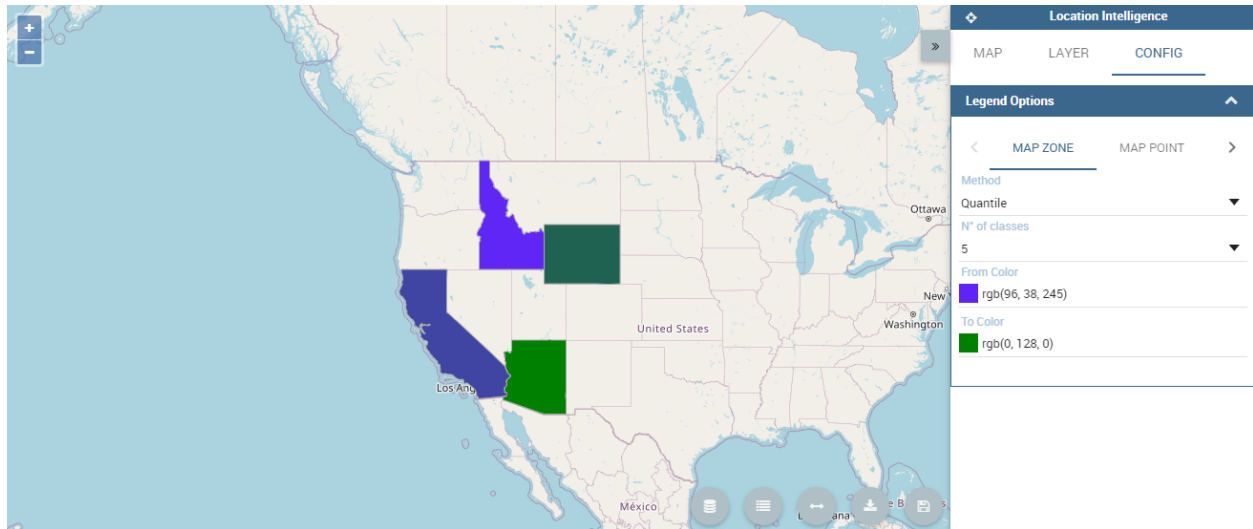


Fig. 1.314: Map Zone extra configurations

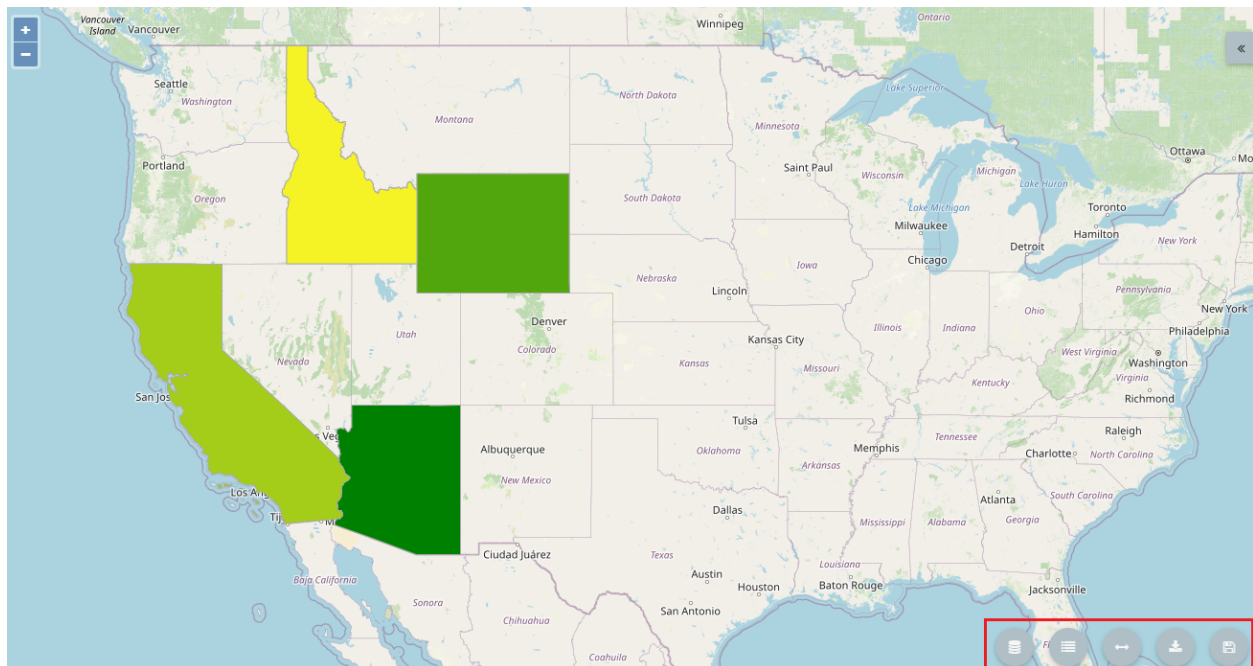


Fig. 1.315: From the left to the right: Legend, Measure and Export bottom.

1.8.1.4 GEOReport Engine

The **GEOReport Engine** implements a *bridge integration* architecture.

Generally speaking, a bridge integration involves both the BI and the GIS systems, still keeping them completely separated. The integration between spatial data and business data is performed by a dedicated application that acts as a *bridge* between the GIS and the BI suite. This application extracts the spatial data from the GIS system and the business data from the BI suite, to answer the users' requests. Afterwards, it joins them and provides the desired results.

In particular, the **GEOReport Engine** extracts spatial data from an external GIS system and join them dynamically with the business data extracted from the Data Warehouse, in order to produce a thematic map according to the user's request. In other words, it acts as a *bridge* between the two systems, which can consequently be kept totally decoupled.

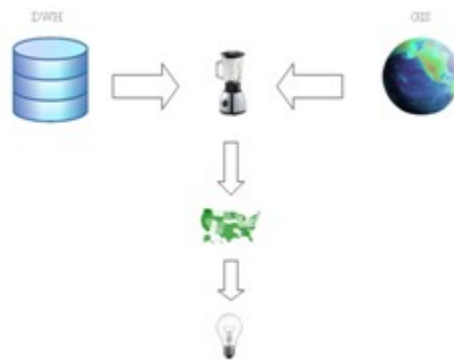


Fig. 1.316: Bridge integration architecture of the **GEOReport Engine**.

The thematic map is composed of different overlapping layers that can be uploaded from various GIS engines at the same time. Among them just one layer is used to produce the effective thematization of the map: this is called *target layer*.

You can manage your layers inside the **Layers Catalogue**.

Here you can upload the following layer types:

- File;
- WFS;
- WMS;
- TMS;
- Google;
- OSM.

Go to **Catalogs > Layers** in the Knowage menu, as shown below.

Here there is the list of already created layers and you can create a new one clicking on the dedicated plus icon. On the right side you are asked to fill few settings before saving the new layer, like a label, a name and a type. At the bottom part of layer configuration you can manage the layer visibility. Mark the role you want to give visibility privileges on this layer. If none is marked, the layer is visible to all role by default. The first settings are equals for all types of layers. Once you choose the layer type, instead, some fields may change according to the layer needs. For example if you choose **File** as type you have the possibility to choose your own .json file and upload it. After having done this, the path where your file is been uploaded is shown among the setting. If you choose **WFS** or **WMS** you are asked to insert a specific url. Below you can find an example of creation of a new layer of type file.

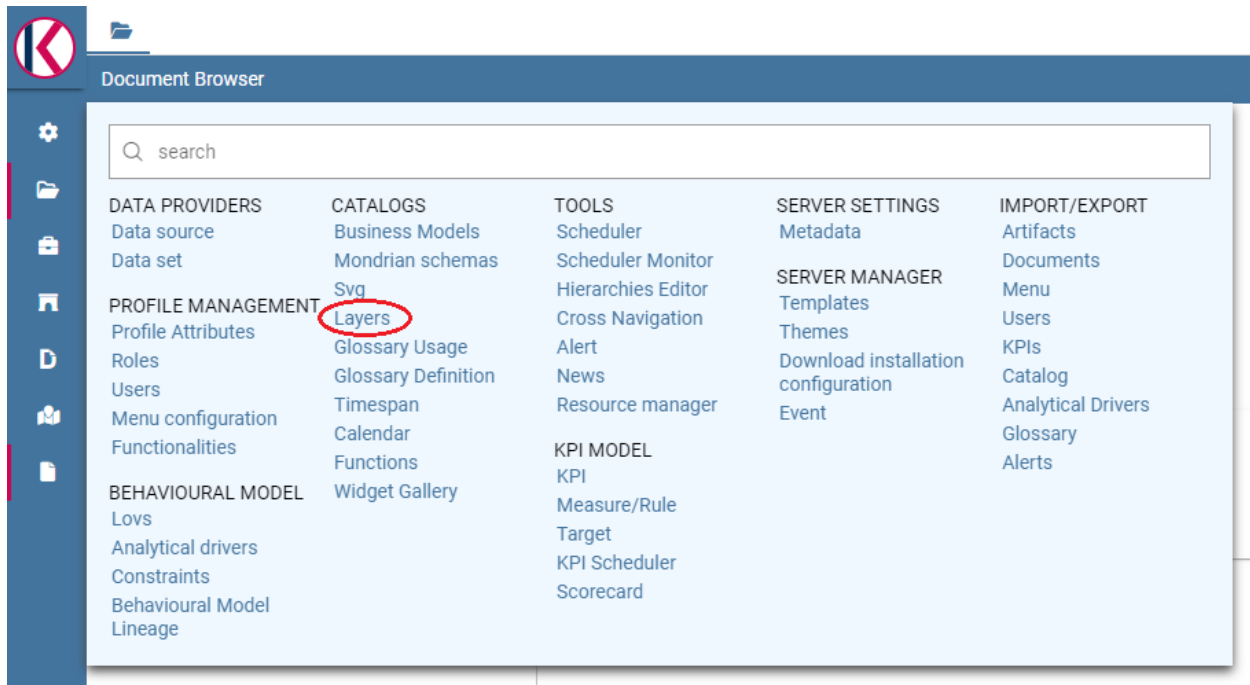


Fig. 1.317: Layers catalog menu item

USA_file

Layer Filter

Label *	Name *
USA_file	USA_file
Description	Category
USA_file	Default Layer Category

Layer Label *	Layer Name *
USA_file	USA_file
Base Layer	Layer ID *
	USA_file
Layer Order *	Roles
1	kte_admin

Type *
File
File Location
/opt/knowage/application-servers/knowage-server-8.1/apache-tomcat-9.0.46/resources/kte/Layer/USA_file

Fig. 1.318: Creating a new file layer

Once you have set all layer configuration you can switch to filter setting. Click on the tab you can find in the upper part of the screen, as the following figure shows.

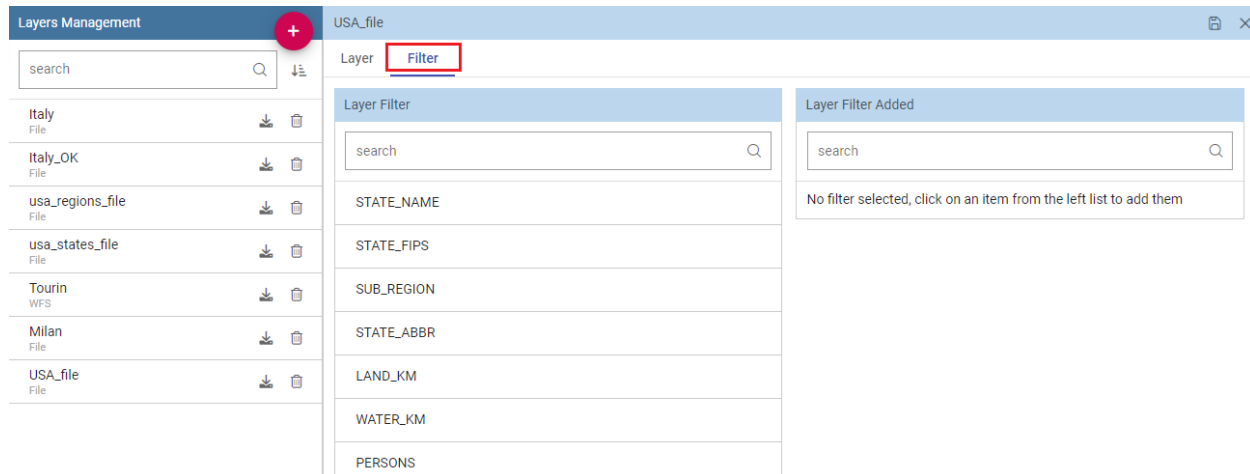


Fig. 1.319: Filter tab

Here you can choose which filters will be active during visualization phase. Choose among the properties of your layer, the available ones are only the string type.

Now you need to have a well-configured dataset to work with the base layer. The dataset has to contain one column matching a property field as type and values otherwise you will not be able to correctly visualize your data on the map.

For example you can use a query dataset, connected to the foodmart data source, whose SQL query is shown in the following code.

Listing 1.24: GeoJSON file except.

```

1  SELECT r.region_id,
2     s.store_country,
3     r.sales_state,
4     r.sales_region,
5     s.store_city,
6     sum(f.store_sales) as store_sales,
7     avg(f.unit_sales) as unit_sales,
8     sum(f.store_cost) as store_cost
9  FROM sales_fact_1998 f,
10     store s,
11     time_by_day t,
12     sales_region r
13  WHERE s.store_id=f.store_id
14     AND f.time_id=t.time_id
15     AND s.region_id = r.region_id
16     AND STORE_COUNTRY = 'USA'
17  GROUP BY region_id, s.store_country, r.sales_state, r.sales_region, s.store_city

```

Create and save the dataset you want to use and go on preparing the document template.

1.8.1.5 Template building with GIS designer

GIS engine document template can now be built using GIS designer. Designer is available both for administrator users and for end users. The first can create a new GIS document in the document browser section (for this part refer to **Template building with GIS designer for technical user** section) while an end user must use the workspace section to create a new document. The creation process for an end user and designer sections are described in the text below.

A GIS document can be created by a final user from workspace area of Knowage Server. Follow **My Workspace » My Analysis** and click on the “Plus” icon available at the top right corner of the page and launch a new **Geo-referenced analysis**.

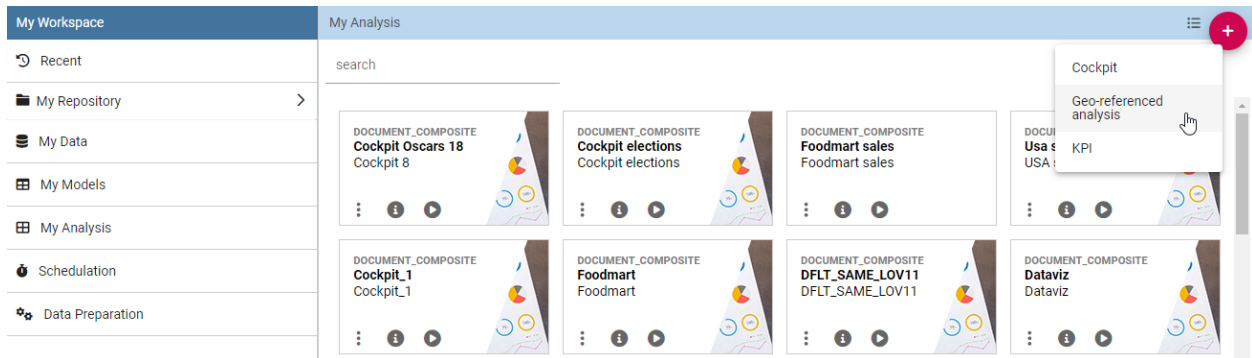


Fig. 1.320: Start a new Geo-referenced analysis.

The designer is divided in four sections that will be described in detail in the following.

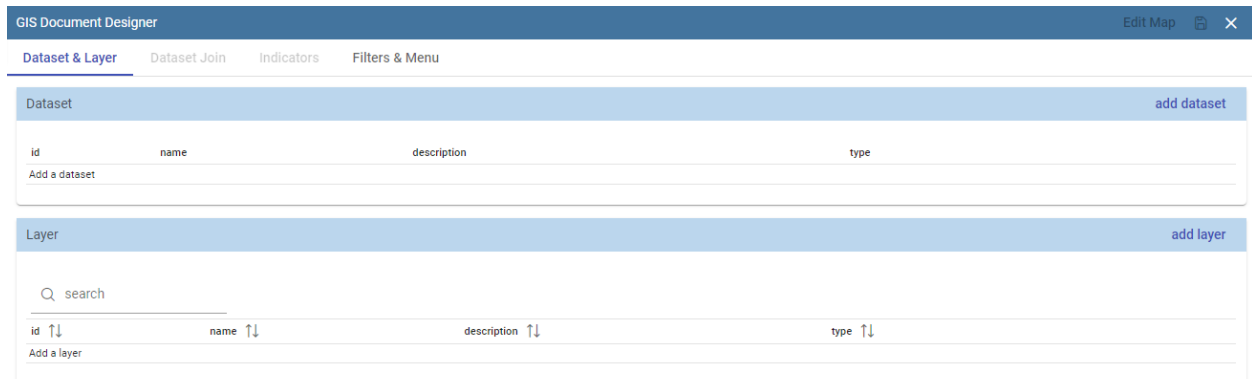


Fig. 1.321: GIS designer.

1.8.1.6 Designer sections

1.8.1.6.1 Dataset & Layer

In the first section the user can choose a dataset for joining spatial data and business data and define the target layer. Click on **add dataset** to choose among the available datasets and on **add layer** to select a target layer. These buttons will open a popup with the list of all available datasets and layer catalogs, selecting one item from the list and clicking save the selected item will be chosen for template.

Once the dataset and the layer have been selected the Dataset join and Indicators sections will be enabled. The user can also change the dataset and layer in a second moment through the buttons **change dataset** and **change layer**.

It is also possible to create map without business data. When there is no dataset multiple layers can be selected, like figure below shows.

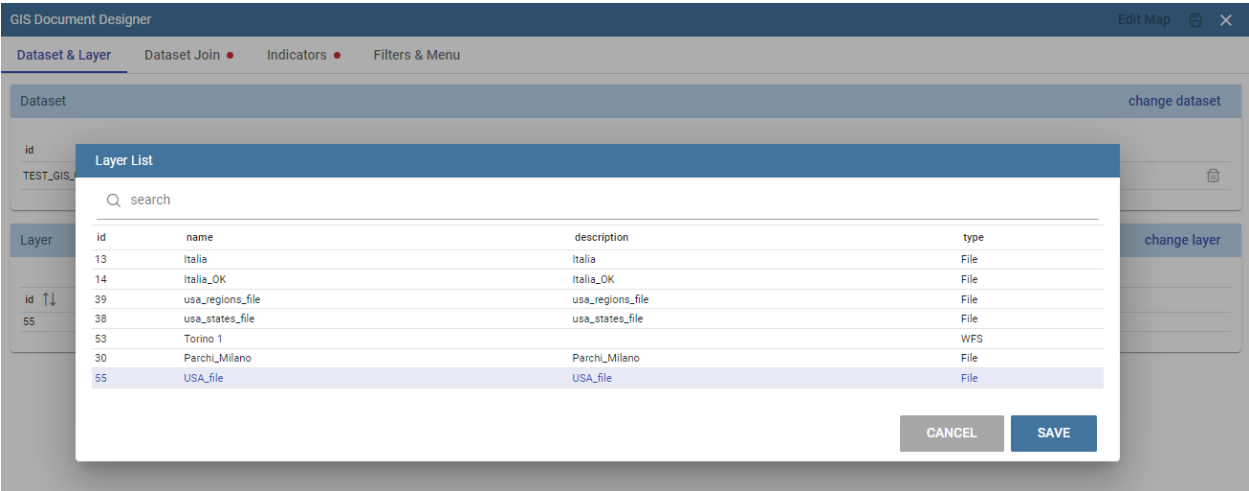


Fig. 1.322: List of available layer catalogs.

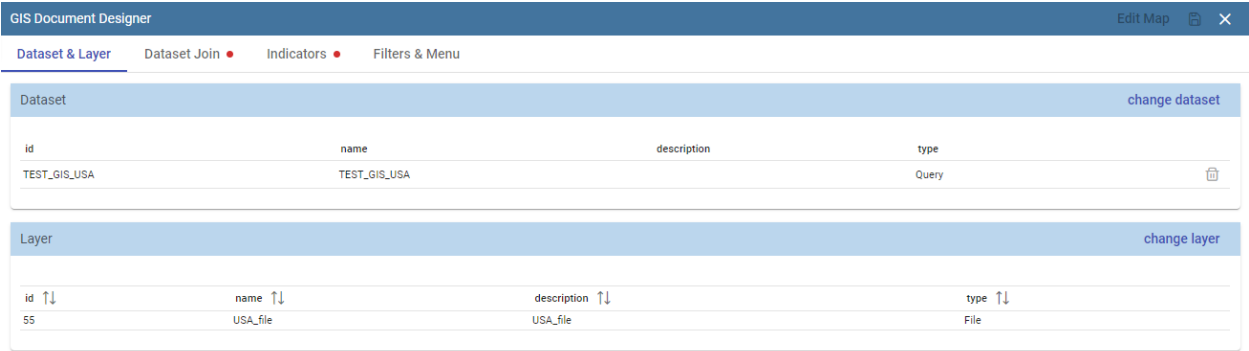


Fig. 1.323: Dataset and target layer definition.

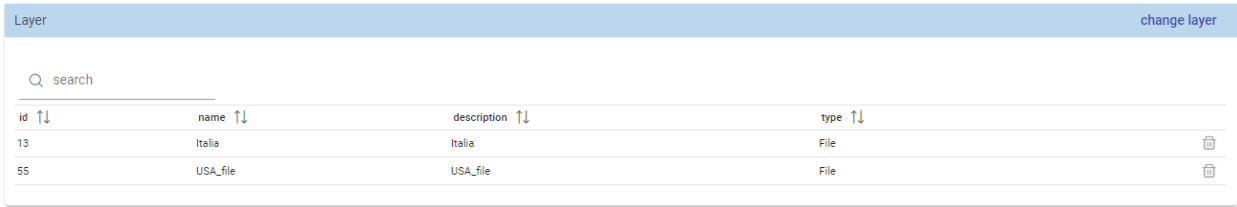


Fig. 1.324: Multiple selection of available layers.

1.8.1.6.2 Dataset join

Dataset join section is for configuring joining spatial data and business data. This section is only available when the dataset is selected for the document. Clicking on **add join column** a new empty row appears with two comboboxes with which the user has to select the dataset column and layer column to join.

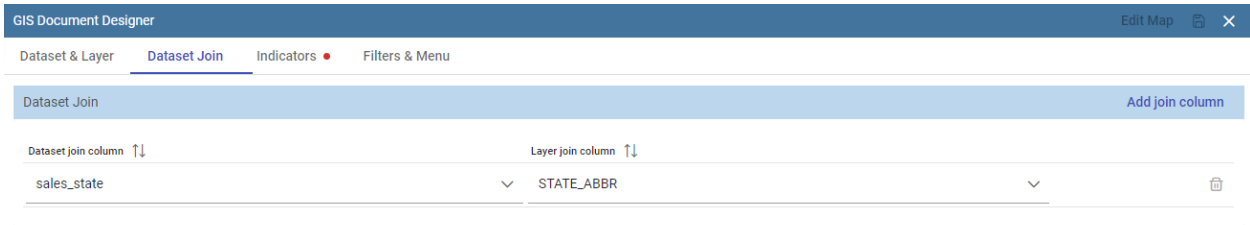


Fig. 1.325: Dataset join columns interface.

1.8.1.6.3 Indicators

Measures definition is configurable by adding indicators. The interface is shown below. This section is available only when dataset is chosen for the document. In order to add a new indicator the user must click on **add indicator** and choose the measure field from selected dataset and a corresponding label that will be used on map. Label should be inserted as free text by editing corresponding table column.

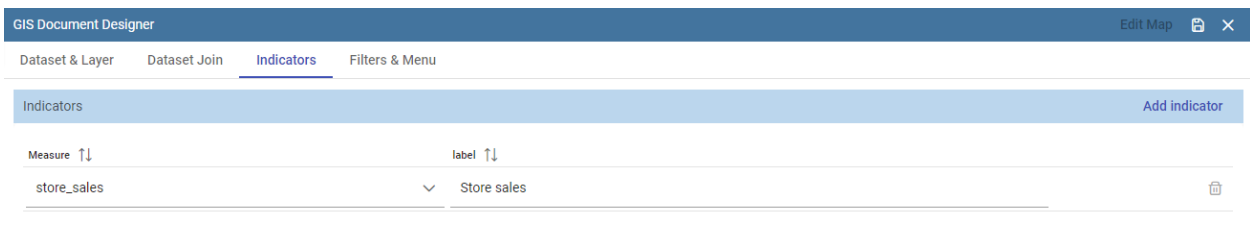


Fig. 1.326: Indicators interface.

1.8.1.6.4 Filters & Menu

Through the **Menu** panel the user can enable or disable some available map functions and features, like the legend, the distance calculator and so on.

Using the filtering dedicated area you can define which dataset attributes can be used to filter the geometry. Each filter element is defined by a name (e.g. "store_country") and a label (e.g. "COUNTRY"). The first value is the name of the attribute as it is displayed among the dataset attribute fields. The second one is the label that will be displayed to the user. This section is only present when dataset is chosen for the document. Clicking on add filter creates empty pair.

1.8.1.6.5 Edit map

When all required fields are filled in the basic template can be saved. From workspace user is first asked to enter label and description of new created document as in the following figure.

When the template is saved successfully EDIT MAP button is enabled on the top right corner of the main toolbar. Clicking the edit map button will open created map. An example is given below. In edit mode you are able to save

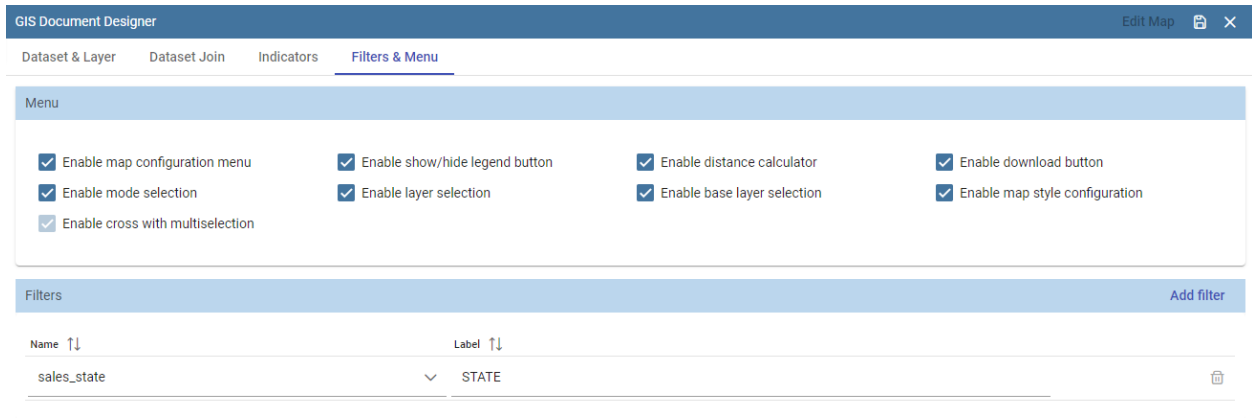


Fig. 1.327: Filters & Menu interface.

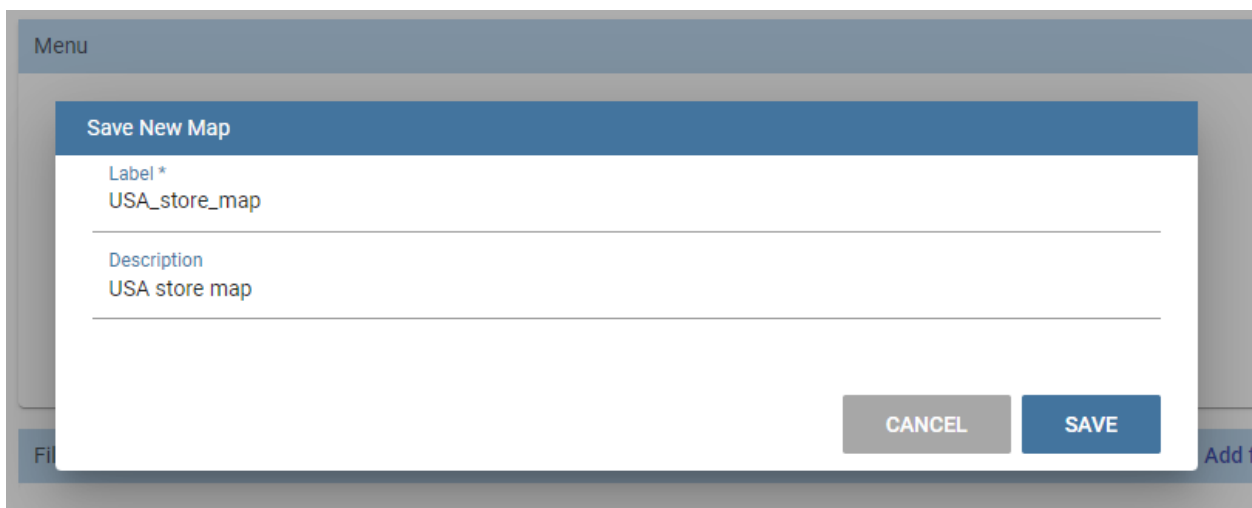


Fig. 1.328: Saving a new geo document for end user.

all custom setting made on map; all the available settings are explained in the previous section **Analytical document execution**.

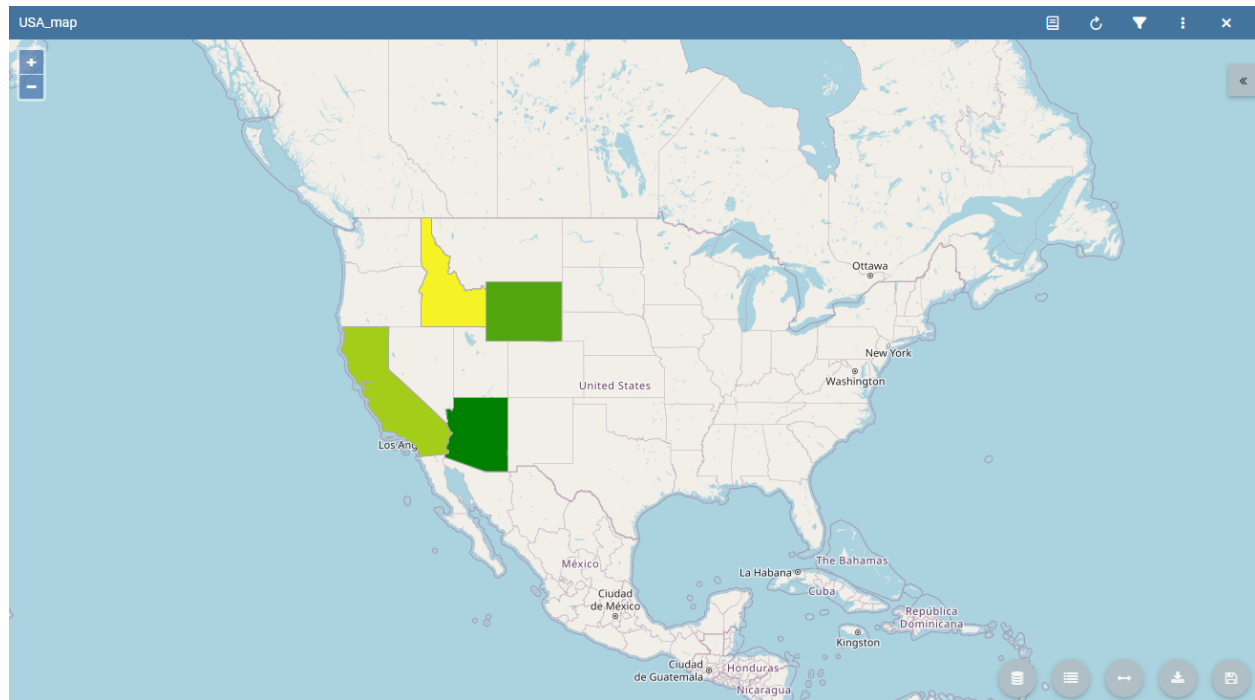


Fig. 1.329: Map in edit mode with save template available.

1.8.1.7 Template building with GIS designer for technical user

The administrator can create a new GIS document clicking on the plus button in the document browser section and selecting “Gneric document”. Choose a Label and a Name, **Location Intelligence** as Type and **GIS Engine** as Engine and the State. The selection of a dataset and the corresponding data source from which the data come from are optional.

Open then the designer clicking on the **Open Designer** button. When the designer is opened the interface for building a basic template is different depending on if the dataset is chosen for the document or not.

We have already described the Gis Designer in the previous section, when it is created by a final user. The difference relies only in how the designer is launched so we will not repeat the component part and recall to *Designer section* paragraph for getting details.

1.8.1.8 Cross navigation definition

It is possible to enable cross navigation from a map document to other Knowage documents. This means that, for instance, clicking on the state of Texas will open a new detail documents with additional information relative to the selected state.

You need to define the output parameters as described in Section *Cross Navigation* of *Analytical Document* Chapter. The possible parameters that can be handled by the GIS documents are the attribute names of the geometries of layers.

Once you have created a new Cross Navigation in the Cross Navigation Definition menu in Tools section, it is possible to navigate from the GIS document to a target document. There is still a little step to do to activate the cross navigation.

Open the **layer** tab of the Location Intelligence options panel and click on cross navigation select mode. Now the cross navigation is activated and if you click, for example, on one of the state it will compare the above popup.

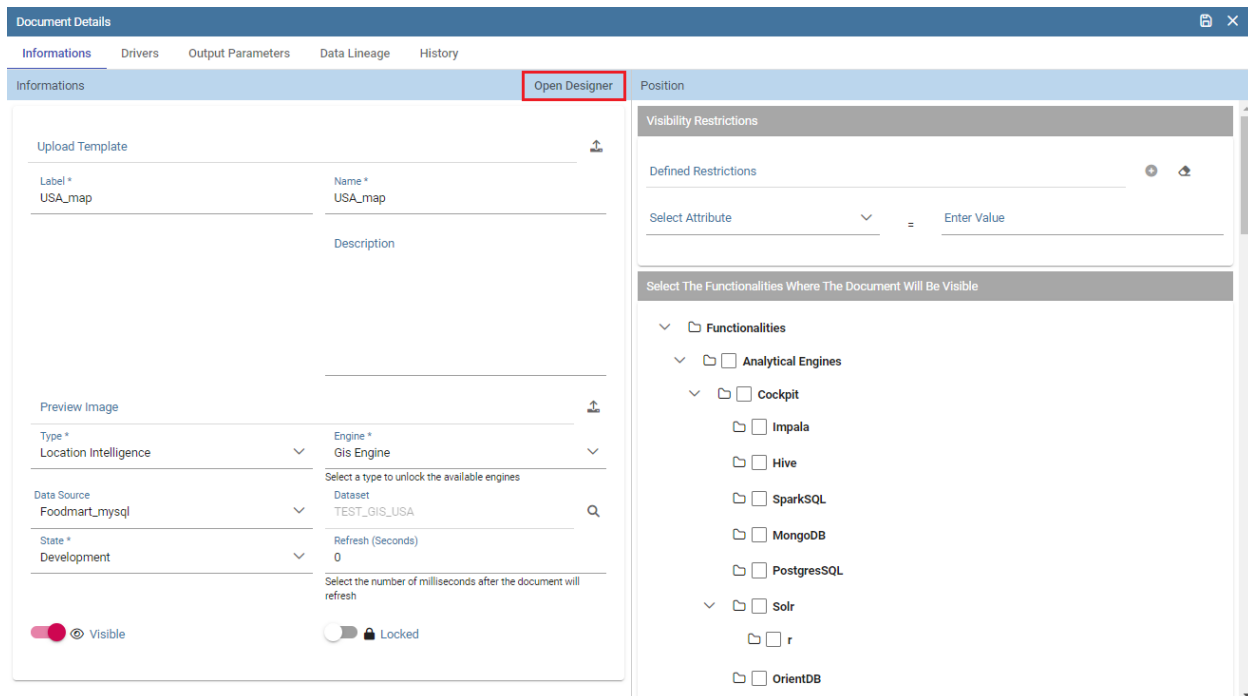


Fig. 1.330: Gis document creation interface.

By clicking on the play button the target document will open.

1.8.2 Create SVG document

In this chapter we will suppose that a technical user has created an SVG document and that the final user is enabled to visualize and use it. He can choose which KPI to show on the “map” and analyze its values, then could drill down into sub-members or other Knowage documents to view other details and other analysis.

The images in Figure above shows how is possible to change KPI analysis and drill towards other SVG documents of the same hierarchy.

1.8.2.1 My first SVG Map or design

The SVG Viewer Engine is a tool that lets you develop documents based on the SVG, acronym for Scalable Vector Graphics, format. It permits to show different business information directly on each area, and permits the drill action to other more detailed SVG files using a logical hierarchy. This viewer is divided into two sections:

- a panel with many dynamic details such measures, layers and legend plus an optional section with specific information about the active document,
- the svg document.

To give an example, we can imagine to visualize through an SVG the USA map. At first we can show data at the “Regions” level and then through the click / drill - show the same or other information at the States “level”. We give an example of map document produced with the SVG engine in the two figures below.

Like other Knowage documents type there is a set of activities managed by the technical users and others used by the final users. These last ones are specifically about consulting.

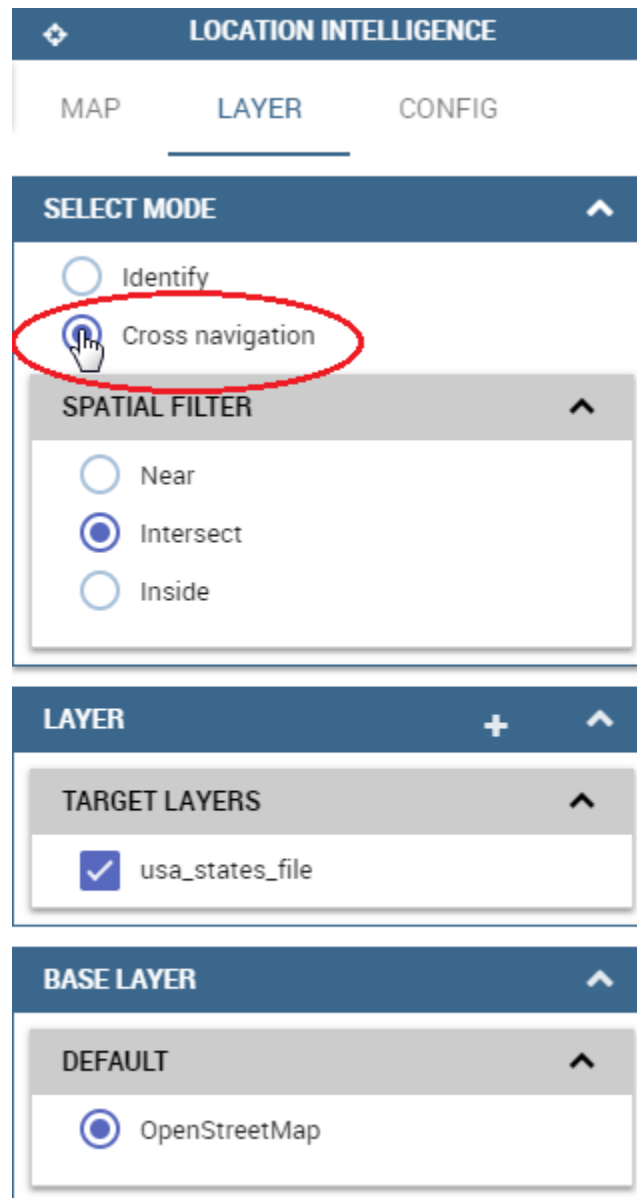


Fig. 1.331: Cross navigation option.

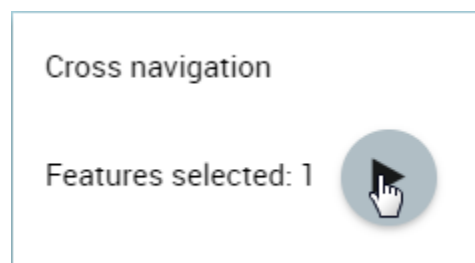


Fig. 1.332: Cross navigation popup.

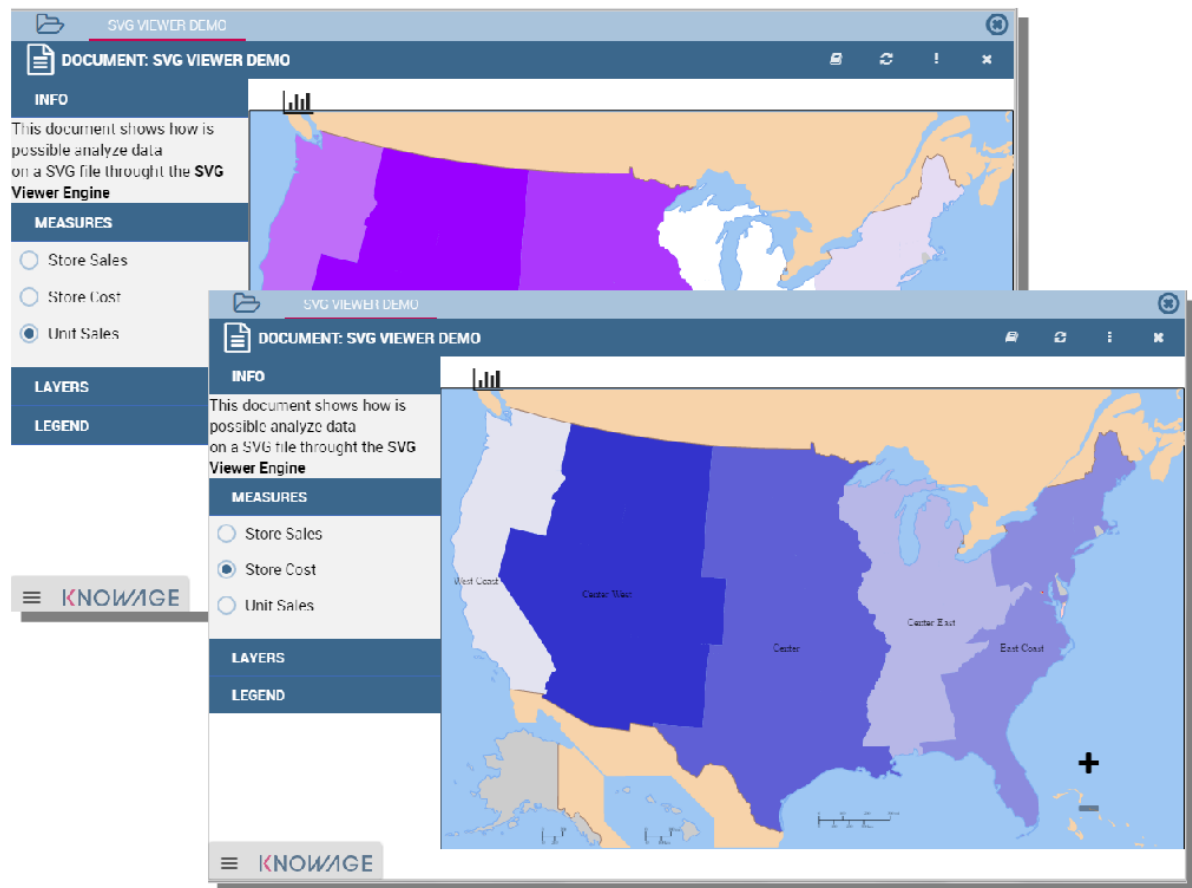


Fig. 1.333: SVG document visualization example.



Fig. 1.334: SVG document example at the USA Regions level.

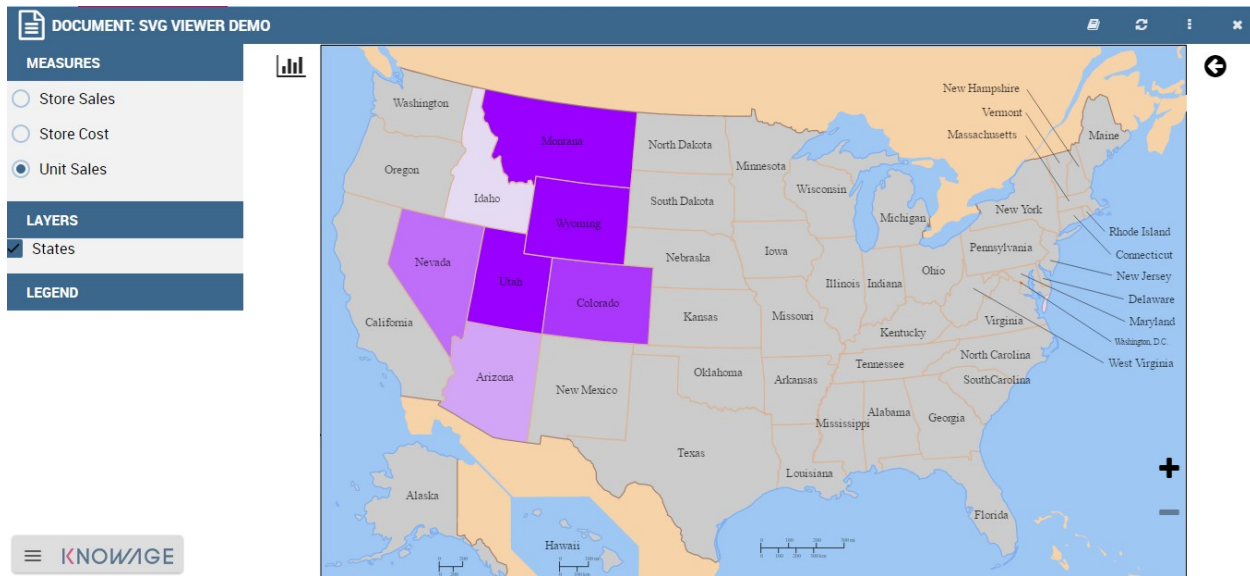


Fig. 1.335: SVG document example at the States level after the selection of the “Center West” Region.

1.8.2.1.1 Technical activities

First of all, a technical user needs to configure the logical hierarchy of the SVG and to define datasets with the business data he/she wishes to show. Finally he/she must type the document template. We will give details about these points in the following sections.

1.8.2.1.1.1 SVG Catalogue

The first activity that you need to do as administrator is to find or create an SVG file. Any file saved in SVG format is a text file in XML format. As a consequence, they can easily be queried, indexed, enriched with scripts and, if necessary, zipped. The SVG final output could represent everything: geographical areas (like USA in the previous example), concepts (like the item production steps) and so on.

1.8.2.1.1.2 SVG Format

The Scalable Vector Graphics, SVG, refers to an XML-based encoding format, used to describe two dimensional vector graphical objects. SVG is an open standard, defined by the World Wide Web Consortium (W3C), which released its first version in 1999. Read more at <http://www.w3.org/Graphics/SVG/>.

Not all graphical objects of an SVG can be thematized. Using the SVG grouping operator `<g>`, the developer can create one or more subsets of graphical objects and specify which groups should be subject to thematization. Each group has a unique name, corresponding to the value of the id attribute of the `<g>` tag (e.g. `<g id="regions">`). Considering that, graphical objects grouped in an SVG file are usually homogeneous elements (in other words, they model a same category of objects: regions, towns, streets, etc.), we can consider these groups as layers and the objects can be considered as features.

Once obtained the SVG file, you should register it into Knowage SVG catalogue.

The Svg catalogue contains all SVG that can be used with this engine through specific hierarchies. In this context a hierarchy is a definition of three concepts:

- the hierarchy itself,

- the level,
- the member.

These three information are used from the system to recover the correct SVG into the catalogue.

SVG DETAIL	
Name	Map USA Regions *
Description	Map USA Regions
Hierarchy	USA
Level	1
Member	regions
Template	<input type="button" value="Scegli file"/> Nessun file selezionato <input type="button" value="Download Svg..."/>

FEATURE DETAIL	
regions	centroidi_regions
State_borders	Country_borders
Labels_Region_Names	Labels_State_Names
New...	

Name	regions
Description	
Type	

Fig. 1.336: Entering the hierarchy details.

As you can see in the figure above, you must insert a name and an optional description of the new SVG component, then you need to specify a logical hierarchy's label, its number of the level and a logical name for the member that it represents. At last you need to upload the SVG file. When this configuration will be saved, the system will read the SVG content and for each group (or tag <g>) will be created a layer. All layers will be shown into the "Feature Detail" section (read only section).

In this first example in the figure above we defined an SVG component for the USA regions specifying that it's the first level (in other words it's the first SVG of the "USA" hierarchy).

The second level (the more detailed SVG) is about the USA states and it's defined like the next example below:

SVG DETAIL	
Name	Map USA States *
Description	Map USA States
Hierarchy	USA
Level	2
Member	states
Template	<input type="button" value="Scegli file"/> Nessun file selezionato <input type="button" value="Download Svg..."/>

FEATURE DETAIL	
State_borders	Country_borders
Labels_State_Names	Frames
State_borders_old	Ocean_borders
Great_Lakes_borders	
New...	

Name	State_borders
Description	
Type	

Fig. 1.337: Entering the hierarchy details.

As you can see the principal differences between these configurations are only about the level content and the member label. This means that both will be used in the same hierarchy's context and that from the "Regions" SVG will be possible to drill on the "States" SVG. Anyway it is not mandatory to define more than one level: it depends from each project implementation.

1.8.2.1.1.3 Datasets definition

After that all SVG was loaded, you must define a dataset (one for each level) that you want to use for getting and showing business information from your DWH. You can refer to Chapter 3 of this manual to know how to create datasets. Here in the following figure a dataset of our example:

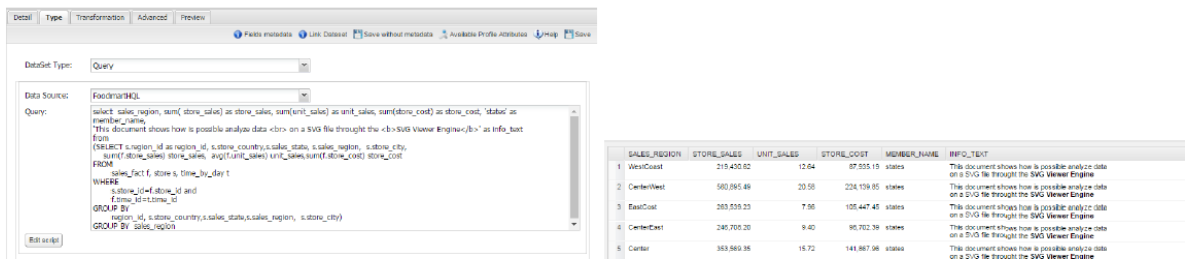


Fig. 1.338: Left. Dataset behind the SVG document. Right. Dataset preview.

1.8.2.1.1.4 Template building

The template allows the SVG viewer to properly join business data (Knowage dataset) and spatial data (SVG included in the catalog), in order to produce the analytical documents.

At the moment there is not yet a designer to create a template for this engine, anyway, it's an XML file very simple to define.

An example below.

Listing 1.25: Example of SVG code for template file.

```

1  <MAP>
2    <DATAMART_PROVIDER>
3      <HIERARCHY name="USA">
4        <MEMBER name="regions" measure_dataset="ds_regions" level="1">
5          <MEMBER name="states" measure_dataset="ds_states" level="2">
6            <HIERARCHY>
7              <DATAMART_PROVIDER>
8            <MAP>

```

Basically, it's necessary to specify the hierarchy that we want to use, as well as its members (remember that with member we are considering a specific SVG).

We recap the meaning of the main tag in the next table *Recap of properties and function*.

After, we need to define each member and first of all we can note that is composed by three sections: METADATA, LAYERS and MEASURE, as in Code below:

Listing 1.26: Example of SVG code for template file.

```

1  <MEMBER name ="regions" measure_dataset = "ds_regions" level="1" >
2    <METADATA>
3      <LAYERS>
4      <MEASURES default_kpi="UNIT_SALES">
5  </MEMBER>

```

Let us see each of them in more depth.

- **METADATA.** This is the section where we define the dataset metadata, in fact, each COLUMN tag defines the dataset columns that we want to use as attribute, as measure (used for thematize the SVG) or other technical meaning usefull for the engine.

Listing 1.27: Example of SVG code for template file.

```

1  <METADATA>
2    <COLUMN TYPE="geoid" column_id="sales_region" />
3    <COLUMN TYPE="measure" column_id="store_sales" />
4    <COLUMN TYPE="measure" column_id="store_costs" />
5    <COLUMN TYPE="measure" column_id="unit_sales" />
6    <COLUMN TYPE="drillid" column_id="member_name" />
7    <COLUMN TYPE="info" column_id="info_text" />

```

Once again we give some details on metadata in next table.

- **LAYERS.** In this section we define all layers that we want to enable in the document. Each layer will be shown into the detail panel “Layers section” as you can see in figure below and could be activated or disactivated directly by an action of the the final user. At least one layer must be defined.



Fig. 1.339: Available layers set by a technical user.

Listing 1.28: Code relative to the LAYER setting.

```
1      <LAYERS>
2          <LAYER name="regions" description="Regions" selected="true" />
3          <LAYER name="Labels_Regions_Name" description="Labels_Regions_Name" selected="false" />
4      </LAYERS>
```


Table 1.9: Recap of properties and function.

Tag	Property	Note
HIERARCHY	name	Mandatory. The name of the hierarchy that we want use. The name must match to an existing hierarchy into the SVG catalogue.
MEMBER	name	Mandatory. The name of the member that we want use. The name must match to an existing member for the hierarchy specified into the SVG catalogue. Is too possible get its value dynamically through an analytical driver by using the standard syntax \$P<driver_url>
MEMBER	measure_dataset	Mandatory. The label of the dataset defined in Knowage Dataset configuration.
MEMBER	level	Mandatory. The number of the level. This value must match the level property into the catalogue for the hierarchy and the member specified.
COLUMN	TYPE	Mandatory. The type of the specific column. Possible values are: <ul style="list-style-type: none"> • geoid: mandatory. The engine uses this column to join the dataset records and the corresponding features in the svg. Also, it's the default value passed within the drill action to the svg of lower level (alternatively to the drillid property). • measure: mandatory. Defines the column like measure. All measures defined in this section will be shown into the detail panel (Measure section). • drillid: optional. Defines the alternative value to pass within the drill action to the next svg • parentid: optional. Defines the column that the system need to use for get correctly data linked to the parent value selected. • crosstype: optional. Defines the column that set the cross navigation type. Possible values are "cross" for external navigation or "drill" for internal navigation. If the single element returns null the link will be disabled • visibility: optional. Defines the column that through
1.8. Create location Intelligence		

- **MEASURES** Measures are all the business values (KPI) that the user want to monitor through this document type. Each measure defined in this section will be shown into the detail panel (“Measures” section) with a specific thematization and could be enabled or disabled directly by an action of the final user. When the measure is active all its values are shown onto the SVG and each area has a specific tonality of the color according to the threshold definition and its real value. All thresholds range are visualized into the “Legend” section of the detail panel as highlight in the following figure. Is possible to choose the thematization logic that it could be as quantile, percentage, uniform or static. Next, we’ll see both definitions (see Thresholds details). Remember, that at least one measure must be defined.

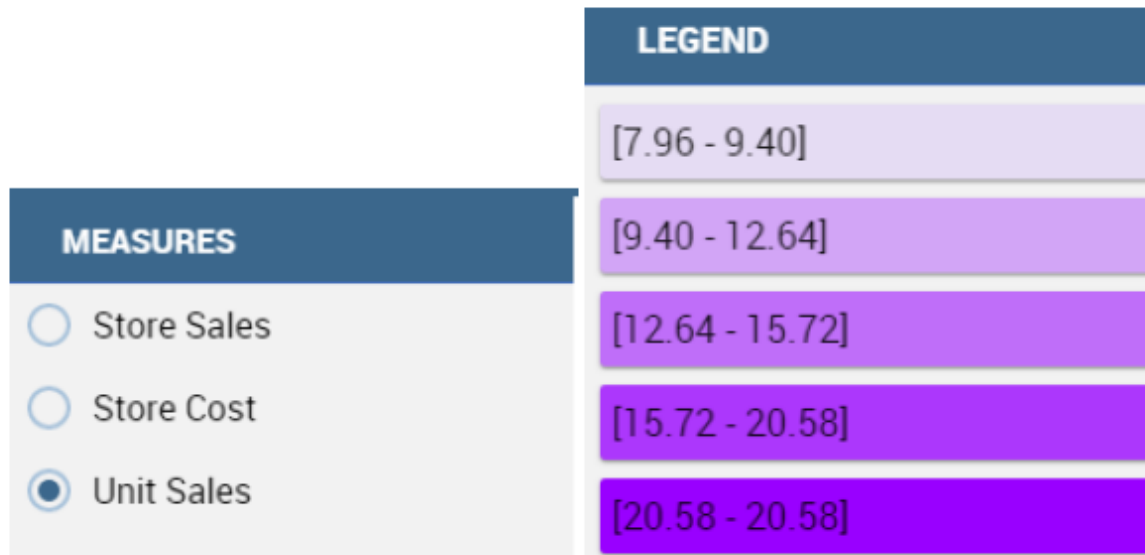


Fig. 1.340: Left. Measure panel. Right. Legend panel.

Listing 1.29: Code for setting the KPI into SVG document.

```

1  <MEASURES default_kpi="UNIT_SALES">
2    <KPI column_id="STORE_SALES" description="Store Sales" >
3      <THRESHOLDS type="quantile" lb_value="0" ub_value="none" >
4        <PARAM name="GROUPS_NUMBER" value="5" />
5      </THRESHOLDS>
6      <COLOURS type="grad" outbound_colour="FFFFFF" null_values_color="CCCCCC" >
7        <PARAM name="BASE_COLOR" value="#009900" />
8        <!--<PARAM name="opacity" value="0.5" />--> </COLOURS>
9      </KPI>
10     <KPI column_id="STORE_COST" description="Store Cost" >
11     <KPI column_id="UNIT_SALES" description="Unit Sales" >
12  </MEASURE>

```

We report the next table for further details on THRESHOLDS and COLOURS tag. This table includes the heuristics supporting value interval partition into a finite number of subintervals (type attribute of the THRESHOLDS tag).

While the following table defines the heuristics supporting color definition for each value sub-interval (type attribute of the COLOURS tag).

Sometimes users need to color the map and, at the same time, to continue to see the underlying objects, through a transparency effect (e.g. a raster image). In this case, specify the opacity parameter in order to properly regulate the transparency level of colors (1 = no transparency; 0 = invisible).

Now, after the template definition, you can create it into Knowage. Remember that it must be a “Location Intelligence”

document type with the engine “SVG Viewer Engine”.

Table 1.10: Recap of layer tag properties and function.

Tag	Property	Note
MEASURES	default_kpi	Mandatory. Defines the default kpi or the kpi that we want enable at the beginning, when we start the document execution. Its value must exist into the METADATA section as measure type.
KPI	column_id	Mandatory. The column_id property the measure that you are defining. Its value must exist into the METADATA section as measure type.
KPI	Description	Mandatory. The label that you want show into the detail panel.
THRESHOLDS	type	Mandatory. The type of logic to use to define the thematization. It could be: <ul style="list-style-type: none"> • quantile: it partitions the interval into N quintiles. • perc: it partitions the interval into subintervals whose extent represents a specific fraction of the overall interval extent. • uniform: it partitions the interval into N subintervals of a same extent. • static: it partitions the interval into smaller fixed-size subintervals, statically defined by the RANGE parameter
THRESHOLDS	lb_value	Mandatory. The lower value outside of which no value is considered.
THRESHOLDS	ub_value	Mandatory. The upper value outside of which no value is considered.
PARAM	name	Mandatory. Specify the parameter value necessary to define correctly the thematization. Its value depends by the threshold type. This attribute could be present more than once.
PARAM	value	Mandatory. It's the parameter name value.
PARAM	label	Optional. Specify the static labels for the legend when thresholds type is "static".
PARAM	value	Optional. It's the parameter label value.
COLOURS	type	Mandatory. Specify the logic type for defining colors range. It could be: <ul style="list-style-type: none"> • static: it assigns each sub-interval a specific color that is statically defined. • grad: it assigns each sub-interval a specific color that

1.8.2.1.1.5 Advanced functionalities

Other the default drill navigation that you have if for the document are defined more than one member, is it possible to cross versus other Knowage documents. To enable this feature, is necessary to set the enableExternalCross property for the MEMBER tag. Here an example:

Listing 1.30: Code for enabling external cross navigation.

```
1 <MEMBER name="states" level="2"
2   measure_dataset="ds_states"
3   enableExternalCross="true">
```

With this setting, you are able to create a “Cross Navigation Definition” with the standard Knowage functionality, where for default you’ll find the element_id as output parameter as shown in figure below. It means that the identifier of the area selected is able to be passed. Other default output parameters are **Hierarchy**, **Member** and **Level**.

Navigation name *

Origin doc

SVG Viewer Demo

SELECT

AVAILABLE INPUT/OUTPUT PARAMETERS

Fixed value parameter

Input	Output
ELEMENT_ID	Output

Fig. 1.341: Using the Cross Navigation definition to link to external documents.

In a cross navigation it is also possible to pass the dataset column values. It is only necessary that a technical user prepares specific output parameters, setting the name like the alias of the dataset columns.

1.9 Create Data Mining analysis

Knowage supports advanced data analysis allowing you to extract knowledge from large volumes of data, to improve your decision-making and business strategies. In particular, Knowage **Data Mining Engine** integrates Python scripting capabilities.

Thanks to Knowage **Data Mining Engine**, it is possible to execute Python scripts in an interactive way and enrich traditional datasets with new information. This means that it allows users to perform statistical or data mining analysis on different Knowage datasets.

The data scientists can thus integrate its own algorithm within Knowage and deliver their output to the end user, together with new advanced visualization options useful to discover meaningful insights hidden in the data.

1.9.1 Functions Catalog

The Data Mining can be managed through the **Functions** framework. In this section we will see how to explore and handle this part, while in *Use a function inside documents* we will see how to use functions.

First click on the **Functions** menu under the **Catalogs** section from the Knowage main page as shown below.

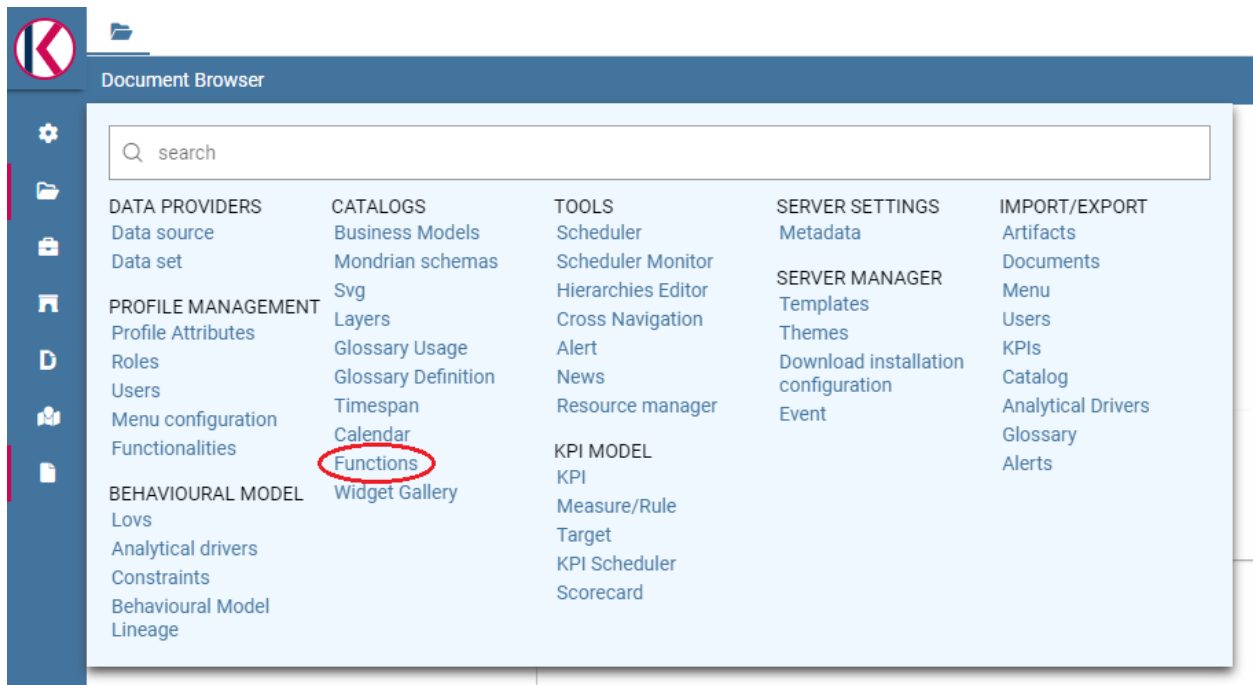


Fig. 1.342: Functions Catalog from Knowage menu.

You will enter a page like the one shown in figure below.

The actions that a user can perform depend on the user's role. However, independently from the user's role, in the interface all functions are shown by default. Referring to the figure above, the page is composed by:

- **categories:** these are set by an administrator user and are used to classify the functions accordingly to their definition and goals. Moreover they help in browsing the functions; only the admin user can add and/or modify categories.
- **search:** to easily search for a specific functions in the list;
- **list of functions** (if there are any): these are visible and explorable by any kind of user. Anyway only an admin user can add and/or modify them.

Hint: Add or modify the categories

The admin can add a new category using the Domain management available on Knowage Server under the Server Settings section. To know more about this section, please refer to Section "Server settings" under the Installation and configuration section.

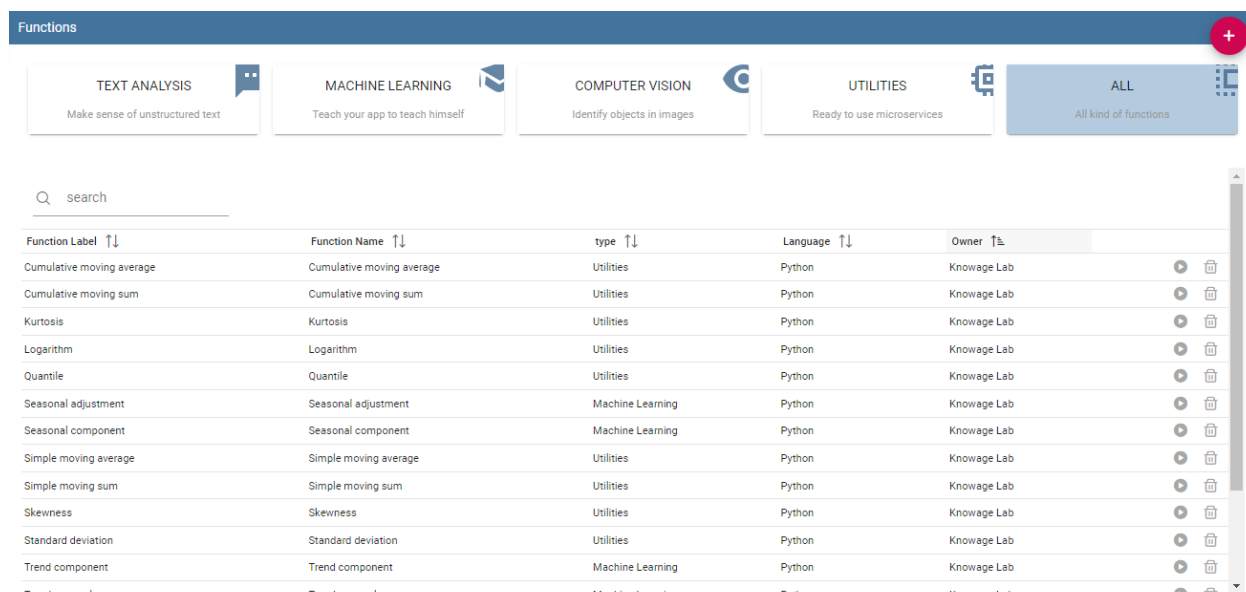




Fig. 1.343: Functions Catalog interface.

The categories for functions depends on an admin user. Taking *Functions Catalog interface* figure as an example, we have:

1. **Text Analysis:** make sense of unstructured text,
2. **Machine Learning:** teach your app to teach himself,
3. **Computer Vision:** identify objects in images,
4. **Utilities:** ready to use microservices,
5. **All:** visualizes all your functions; this is the only category that cannot be changed or removed.

It is possible to search for a specific function in two ways: using the categories and clicking on one of them in order to filter the functions list or using the search box on the top of the list.

A preview of the function can be executed using the icon  which opens a dialog, here you can select and configure a dataset among the available ones in order to test the function. Use the icon  for deleting the function. Functions cannot be deleted if they are used inside one or more documents.

To create a new function an admin user must click on the “Plus” icon available at the top right corner of the page. The action opens the interface shown below. Here you have four tabs that we describe shortly in the following subsections.

1.9.1.1 The General tab

In this tab the user gives the general information about the function as the figure above shows. The admin user must type: the *name* of the function, the *label* with which it is identified uniquely (remember to use only numbers or letters and do not leave spaces between them) and the *type* that is the function category. The *keywords* are tags that can be assigned to a function, searching for a specific tag in the search box will retrieve all the functions that have that tag. In order to add a new keyword you have to write the desired text and then press the submit button on the pc keyboard. The *Description* is where the user can insert a long text or images to be shown when the function is being configured inside documents. In the *Benchmarks* field users can insert information about the function performances.

The screenshot shows the 'General' tab of a function creation form. At the top right are 'CLOSE' and 'SAVE' buttons. Below the tabs are two columns: 'Function Name' (with a sub-label 'Name of the function') and 'Label' (with a sub-label 'Name of the function that will be displayed'). Below these is the 'Owner' field with the value 'kte_admin' and a 'Type' dropdown menu. Further down are three stacked input fields: 'Keywords' (containing 'Keywords'), 'description *' (with a dropdown arrow), and 'Benchmarks' (with a dropdown arrow).

Fig. 1.344: Creating a new function.

1.9.1.2 The Input tab

As shown in the following figure, the function admits two kinds of input: *columns* and *variables*.

The screenshot shows the 'Input' tab of the function configuration. It features two main sections: 'Input Columns' and 'Input Variables'. Each section has a header bar with an 'ADD' button (e.g., 'ADD COLUMN'). Below each header is a large white area with the text 'No input Columns required' and 'No input Variables required' respectively.

Fig. 1.345: Input tab.

In the “Column” instance the function takes input columns that will be referenced inside the script. These columns are generic, the user must only specify their type and the name he later wants to use inside the script to access that specific column.

In the “Variable” case, the user must insert one or more variables and match them with values using the dedicated area.

1.9.1.3 The Script tab

The script tab is where an expert user defines the function through the usage of datamining languages (such as Python), as shown in Figure below.

Inside the script users will have at their disposal a read-only `pandas.Series` variable for each column defined in the input tab. To reference one specific column users must use the placeholder `${column_name}`. Input variables will be accessible with the same syntax.

General • **Input** • Script • Output

Input Columns

ADD COLUMN

Column Name	Column Type
column_1	<div>▼</div> <div>STRING</div> <div>DATE</div> <div>NUMBER</div>

Input Variables

ADD INPUT VARIABLE

Variable Name	Variable Type	Variable Default Value
var_1	<div>▼</div> <div>STRING</div> <div>DATE</div> <div>NUMBER</div>	

No input Variables required

Fig. 1.346: The dataset input of the function settings.

Input Variables

ADD INPUT VARIABLE

Variable Name	Variable Type	Variable Default Value
var_1	<div>▼</div> <div>STRING</div> <div>DATE</div> <div>NUMBER</div>	

Fig. 1.347: The variable input of the function settings.

CLOSE

SAVE

General • Input • **Script** • Output

Language

Python

▼

Script

1

Fig. 1.348: The script tab.

Listing 1.31: Code syntax to recall inputs

```
1  ${column_name}
2  ${variable_name}
```

Warning: Input variables are read only
If you want to manipulate them you should first make a local copy and work on it.

The script will have to produce as output one or more *pandas.Series* variables and will store them inside the corresponding output placeholders. The following is an example of function template.

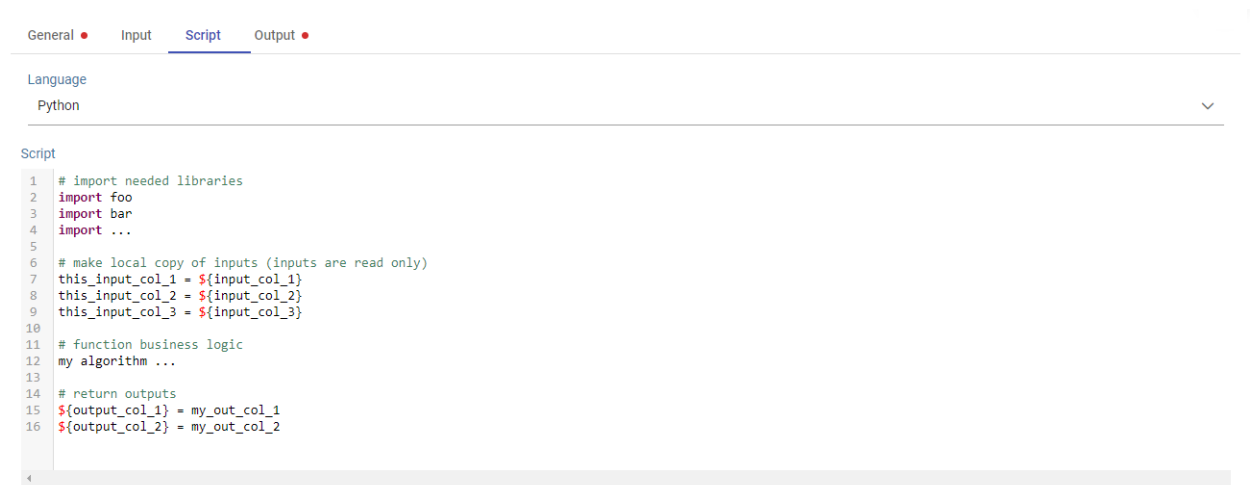


Fig. 1.349: Function template example.

1.9.1.4 The Output tab

Finally it is important to define what kind of outputs the function has produced, according to the script generated in the previous tab. Using the “Output“ tab shown below, you must specify:

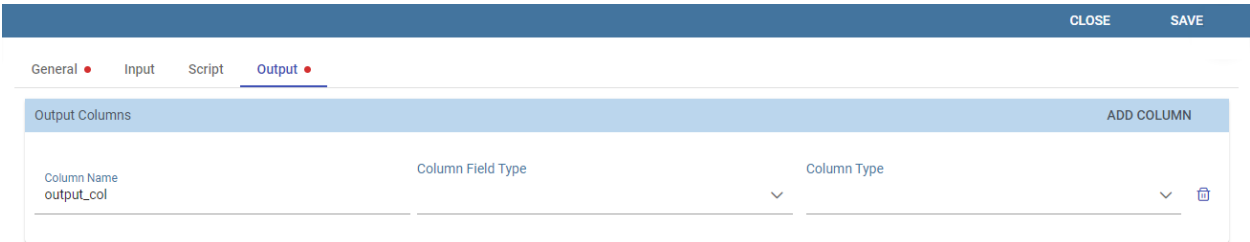


Fig. 1.350: Output tab.

- **Field Type:** it can be *ATTRIBUTE* or *MEASURE*, and defines how the column will behave inside documents;
- **Type:** it depends on the selected Field Type, it can be *String*, *Number (float)* or *Date*;

1.9.2 Engine description

The Catalog Function features leverage on the Python Engine. To understand how to install and configure it, please refer to the [Installation Manual](#)

1.9.3 Use a function inside documents

Now that functions have been created, they must be used inside documents. In this section we will go through all the steps that allow users to execute a function with a specific dataset. This works both for the function preview and for the function used inside cockpits. Depending on the scenario, you will have two different dialogs.

When you are previewing the output of a function, you need to select the dataset you want to use to perform the preview. Therefore on the left card you will be able to select a dataset among the available ones. If the dataset has parameters you will be asked to insert values.

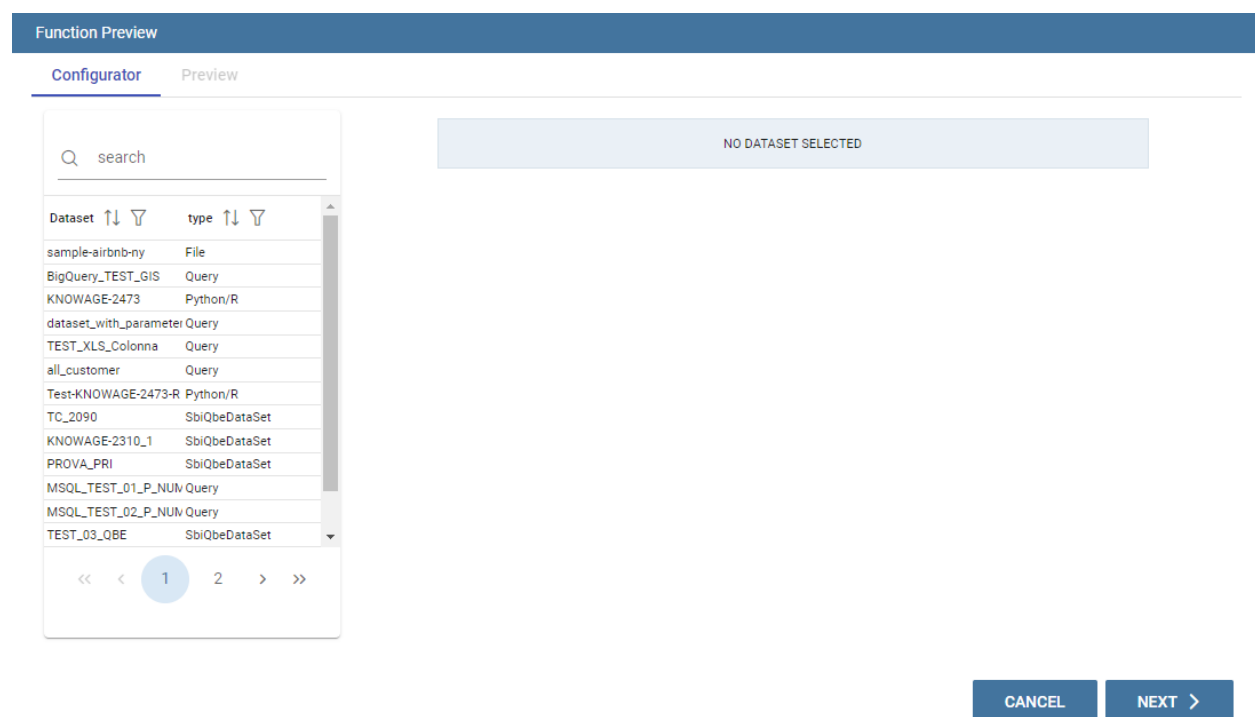


Fig. 1.351: Preview interface.

Instead, when you are creating a new widget that uses a function, you will need to click on the *Use function button* to access the configurator.

Warning: You cannot use more than one function in the same widget

If you try do so you will get an exception.

Note: Functions are available only in some widgets

Table, crosstable, chart and custom chart.

Table Widget Configuration

COLUMNS STYLE CROSS FILTERS

Dataset
toy_dataset (toy_datas... +

Max Rows Number
10

Frontend pagination

TABLE COLUMNS MANAGE COLUMN GROUPS ADD COLUMN ADD CALCULATED FIELD **USE FUNCTION**

Modal selection column

Sorting column

Sorting order

Name	Alias	Type	Data Type	Aggregation	
id	id	MEASURE	# integer	SUM	
name	name	ATTRIBUTE	string		
age	age	MEASURE	# float	SUM	
gender	gender	ATTRIBUTE	string		
state	state	ATTRIBUTE	string		
num_children	num_children	MEASURE	# float	SUM	
num_pets	num_pets	MEASURE	# float	SUM	

CANCEL SAVE

Fig. 1.352: Use a function in a widget.

If you are using a function inside a widget it means that you have already selected the dataset you want to use. Therefore on the left card you will be able to select a function among the available ones.

From this point forward the rest of the configuration is identical for both widget and preview. The first thing that you have to do on the right tab, is bind the input columns of the function with the actual columns coming from the chosen dataset.

By doing this you are providing the **actual data** to the function template. From time to time you can provide different data to the same function just by changing the selected dataset. Depending on this, the same algorithm that is saved inside the function template will work on a different set of data and return different outputs.

If you have defined some input variables, you can also set their values.

The last thing you have to choose is the working environment. You can choose the environment among the available ones. To understand better what is an environment, please refer to the [Installation Manual](#).

After choosing an environment, the list of available libraries installed inside that specific environment appears on screen. You can search or filter libraries and their version in order to find the desired ones, and based on this you can choose the environment that suits your needs the most.

After saving, if you are inside widget configuration you will see that the new output columns generated by the function have been added to the dataset as shown in figure above.

Instead if you were running a preview you will see the output of the dataset execution appearing on screen as shown in the figure below.



Fig. 1.353: Catalog function interface in widgets.

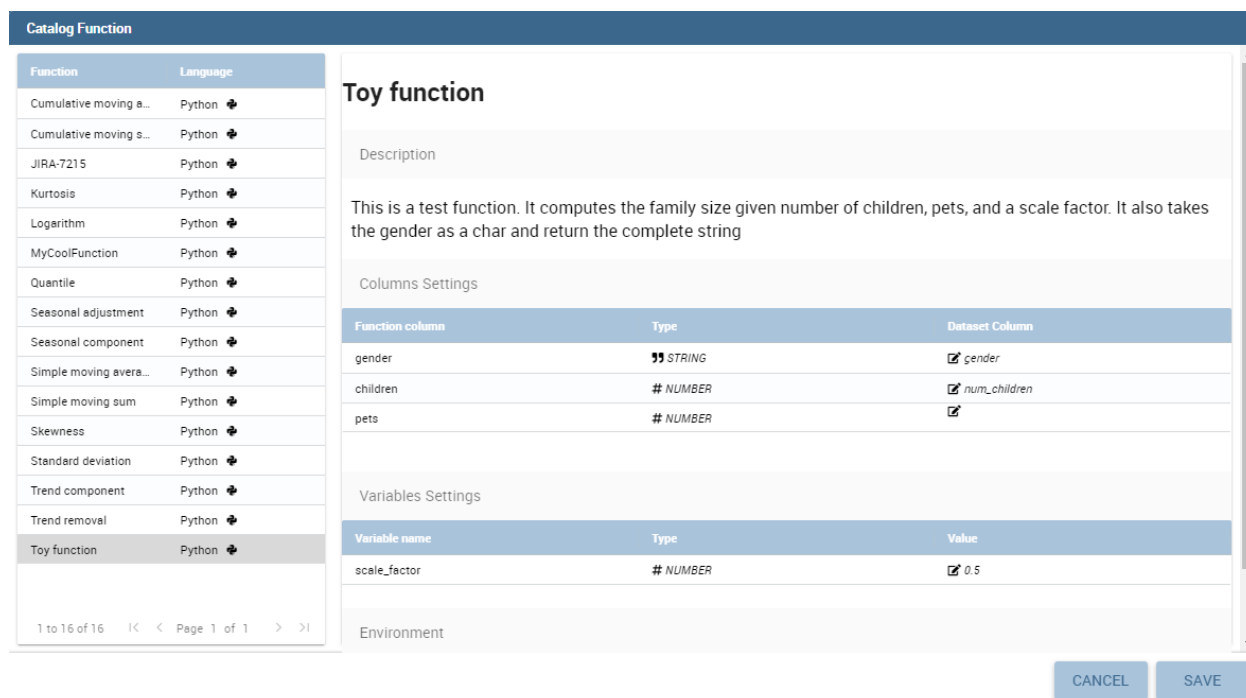


Fig. 1.354: Input columns binding.

Catalog Function

Function	Language
Cumulative moving a...	Python
Cumulative moving s...	Python
JIRA-7215	Python
Kurtosis	Python
Logarithm	Python
MyCoolFunction	Python
Quantile	Python
Seasonal adjustment	Python
Seasonal component	Python
Simple moving avera...	Python
Simple moving sum	Python
Skewness	Python
Standard deviation	Python
Trend component	Python
Trend removal	Python
Toy function	Python

1 to 16 of 16 | < > Page 1 of 1 > >|

Toy function

Description

This is a test function. It computes the family size given number of children, pets, and a scale factor. It also takes the gender as a char and return the complete string

Columns Settings

Function column	Type	Dataset Column
gender	STRING	<input checked="" type="checkbox"/> gender
children	NUMBER	<input checked="" type="checkbox"/> num_children
pets	NUMBER	<input checked="" type="checkbox"/> num_pets

Variables Settings

Variable name	Type	Value
scale_factor	NUMBER	<input checked="" type="checkbox"/> 0.5

Environment

CANCEL SAVE

Fig. 1.355: Input variables binding.

Catalog Function

Standard deviation	Python
Trend component	Python
Trend removal	Python
Toy function	Python

1 to 16 of 16 | < > Page 1 of 1 > >|

Variables Settings

Variable name	Type	Value
scale_factor	NUMBER	<input checked="" type="checkbox"/> 0.5

Environment

python.default.environment.url

Library	Version
tornado	6.0.3
statsmodels	0.12.2
scikit-learn	0.24.2
regex	2021.10.8
PyYAML	5.3
Pillow	8.1.0
MarkupSafe	1.1.1
greenlet	0.4.15

1 to 8 of 58 | < > Page 1 of 8 > >|

CANCEL SAVE

Fig. 1.356: Choose the working environment.

Table Widget Configuration

COLUMNS STYLE CROSS FILTERS

TABLE COLUMNS MANAGE COLUMN GROUPS ADD COLUMN ADD CALCULATED FIELD USE FUNCTION

Modal selection column Sorting column Sorting order

Name	Alias	Type	Data Type	Aggregation	
id	id	MEASURE	# integer	SUM	
name	name	ATTRIBUTE	string		
age	age	MEASURE	# float	SUM	
gender	gender	ATTRIBUTE	string		
state	state	ATTRIBUTE	string		
num_children	num_children	MEASURE	# float	SUM	
num_pets	num_pets	MEASURE	# float	SUM	
gender_str	gender_str	ATTRIBUTE	string		
family_size	family_size	MEASURE	# float		

☐ Summary row ADD

CANCEL SAVE

Fig. 1.357: New columns generated by the functions.

Function Preview

Configurator Preview

search

name	gender	state	id	age	num_children	num_pets	family_size	gender_str
john	M	california	0	23	2	5	4.5	M
mary	F	dc	1	78	0	1	0.5	F
peter	M	california	2	22	0	0	0	M
jeff	M	dc	3	19	3	5	5.5	M
bill	M	california	4	45	2	2	3	M
lisa	F	texas	5	33	1	2	2	F
jose	M	texas	6	20	4	3	5.5	M

<< < 1 > >>

CANCEL < BACK

Fig. 1.358: Output of the function preview.

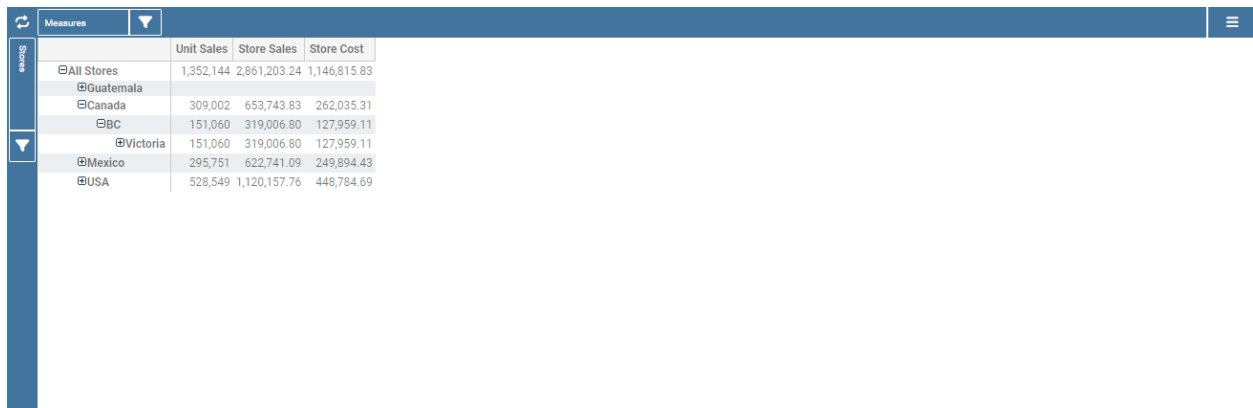
1.10 Create Multi dimensional analisys

1.10.1 Create OLAP

OLAP tools enable users to analyse multidimensional data interactively from multiple perspectives. OLAP consists of basic analytical operations: slice and dice, roll up, drill down and pivot.

1.10.1.1 OLAP user manual step by step

We start our lecture on the OLAP engine by analysing an existing OLAP document. Here you can find an example of OLAP document.



The screenshot shows an OLAP interface with a table of data. The table has four columns: 'Unit Sales', 'Store Sales', and 'Store Cost'. The rows represent different geographical locations, with a hierarchy from 'All Stores' down to 'USA'. The interface includes a 'Measures' dropdown at the top and a sidebar on the left with a 'Sales' section.

	Unit Sales	Store Sales	Store Cost
All Stores	1,352,144	2,861,203.24	1,146,815.83
Guatemala			
Canada	309,002	653,743.83	262,035.31
BC	151,060	319,006.80	127,959.11
Victoria	151,060	319,006.80	127,959.11
Mexico	295,751	622,741.09	249,894.43
USA	528,549	1,120,157.76	448,784.69

Fig. 1.359: Exploring an existing OLAP document.

In the following we will describe the main parts of the OLAP page.

1.10.1.1.1 The filter panel

Once the document is executed, the central area of the window contains the table whose measures are aggregated on dimensions. At the top of this area, panels are available to configure filters on attributes. We see in the following figure that the filter panel is made up of **Filter cards**. Here you can find the cube dimensions and their hierarchies as defined in the OLAP schema by the developer.



Fig. 1.360: The filter panel.

1.10.1.1.2 The filter cards

Filter cards can stay on the filter panel or be placed on column axis. You can move them from the filter panel to the axis or viceversa by dragging and dropping them from one place to the other.

Filter cards are used to:

- inform the user about available dimensions defined in OLAP schema,
- inform the user about dimension's name,

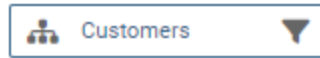


Fig. 1.361: The filter card inside the filter panel.

- perform slices,
- add the dimensions to the cube visualization,
- place hierarchies in different axes,
- filter visible members.

Considering the next figure, we can see that a filter card is made up of:

- an icon for opening hierarchy selection dialog (a),
- the dimension name (b),
- filter icon (if a filter is selected the icon is red) (c)+(d).



Fig. 1.362: Features of a filter card.

1.10.1.1.3 Axes panel

In the panel axes you can:

- drag and drop one or more dimensions,
- organize the dimensions visualization,
- swap axes.

Referring to the following figure, the axes panel consists of the following items:

- columns axis (a),
- row axis (b),
- filter cards (c),
- icon for swap axes (d),
- icon for hierarchy order (e).

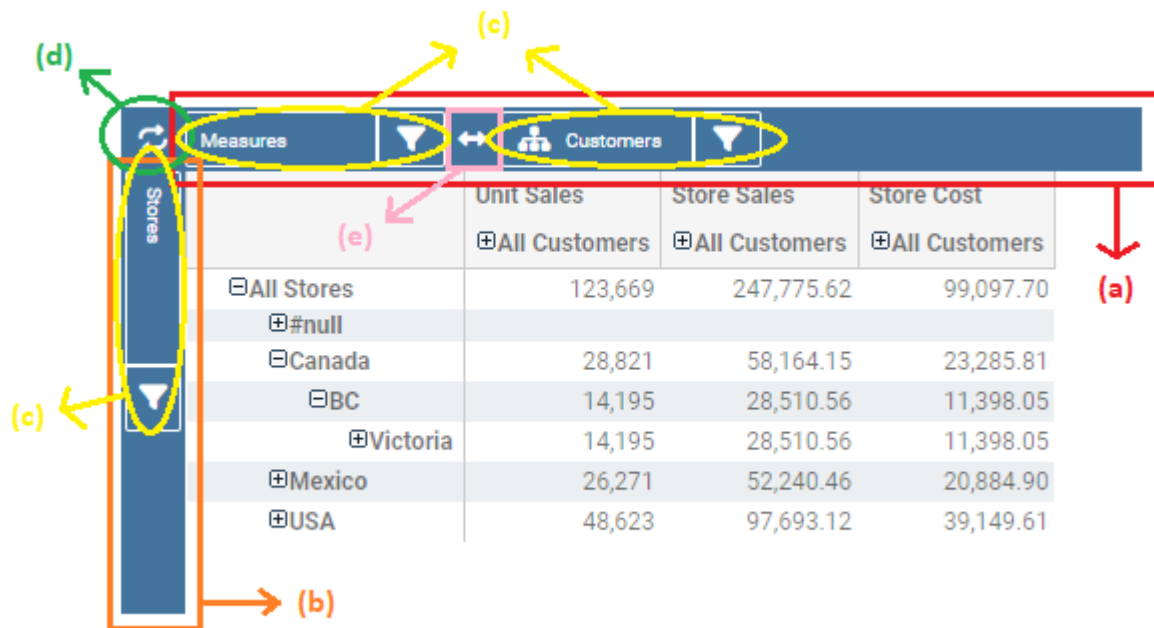


Fig. 1.363: Axes panel features.

	Unit Sales				Store Sales				Store Cost			
	⊖All Customers	⊖Canada	⊖Mexico	⊖USA	⊖All Customers	⊖Canada	⊖Mexico	⊖USA	⊖All Customers	⊖Canada	⊖Mexico	⊖USA
⊖All Stores	123,669	11,350	48,847	63,472	247,775.62	22,784.68	97,736.57	127,254.37	99,097.70	9,117.04	38,999.74	50,980.92
⊖#null												
⊖Canada	28,821	2,508	20,595	5,719	58,164.15	5,049.38	41,709.20	11,405.57	23,285.81	2,052.21	16,628.17	4,605.43
⊖BC	14,195	2,508	11,687		28,510.56	5,049.38	23,461.18		11,398.05	2,052.21	9,345.84	
⊖Victoria	14,195	2,508	11,687		28,510.56	5,049.38	23,461.18		11,398.05	2,052.21	9,345.84	
⊖Mexico	26,271		25,817	455	52,240.46		51,385.70	854.76	20,884.90		20,546.50	338.40
⊖DF	10,496		10,496		21,243.07		21,243.07		8,525.59		8,525.59	
⊖Guerrero												
⊖Veracruz	7,048		6,593	455	13,927.27		13,072.51	854.76	5,569.53		5,231.13	338.40
⊖Yucatan												
⊖Zacatecas	8,728		8,728		17,070.13		17,070.13		6,789.78		6,789.78	
⊖USA	48,623			48,623	97,693.12			97,693.12	39,149.61			39,149.61
⊖CA	11,890			11,890	24,017.64			24,017.64	9,625.53			9,625.53
⊖Los Angeles	6,048			6,048	12,016.47			12,016.47	4,858.08			4,858.08
⊖San Diego	5,842			5,842	12,001.17			12,001.17	4,767.44			4,767.44
⊖OR	14,702			14,702	29,513.05			29,513.05	11,835.00			11,835.00
⊖WA	22,031			22,031	44,162.43			44,162.43	17,689.09			17,689.09

Fig. 1.364: Pivot table.

1.10.1.1.4 Pivot table

The Pivot table is the central part of the OLAP page. In figure below is shown an example.

Pivot table is used to:

- show data based on MDX query sent from the interface,
- drill down/up hierarchies' dimensions,
- drill through,
- show properties of a particular member,
- sort data,
- show calculated fields,
- perform cross navigation to other documents.

Referring to next figure, Pivot table consists of:

- dimensions involved in the analysis (a),
- cells with data (b),
- icons for drill down and drill up (c),
- icons for sorting (only if enabled by the developer) (d),
- links for cross navigation (only if enabled and configured by the developer) (e).

	All Stores	Canada	Mexico	USA
All Products	2,861,203.24	1,146,815.83	653,743.83	262,035.31
Drink	247,775.62	99,097.70	58,164.15	23,285.81
Alcoholic Beverages	71,872.25	28,763.40	17,613.05	7,059.98
Beverages	138,378.34	55,242.97	32,071.88	12,799.34
Dairy	37,525.02	15,091.33	8,479.22	3,426.48
Food	2,048,024.95	821,127.08	466,686.51	187,082.07
Non-Consumable	546,552.43	219,026.94	124,988.02	50,106.33

Fig. 1.365: Pivot table features.

1.10.1.1.5 Side bar

You can open the side bar by clicking on the icon positioned on the top right side of the page (see next figure). Side bar will be shown on the right side (see *Side bar* figure).

Side bar is used to:

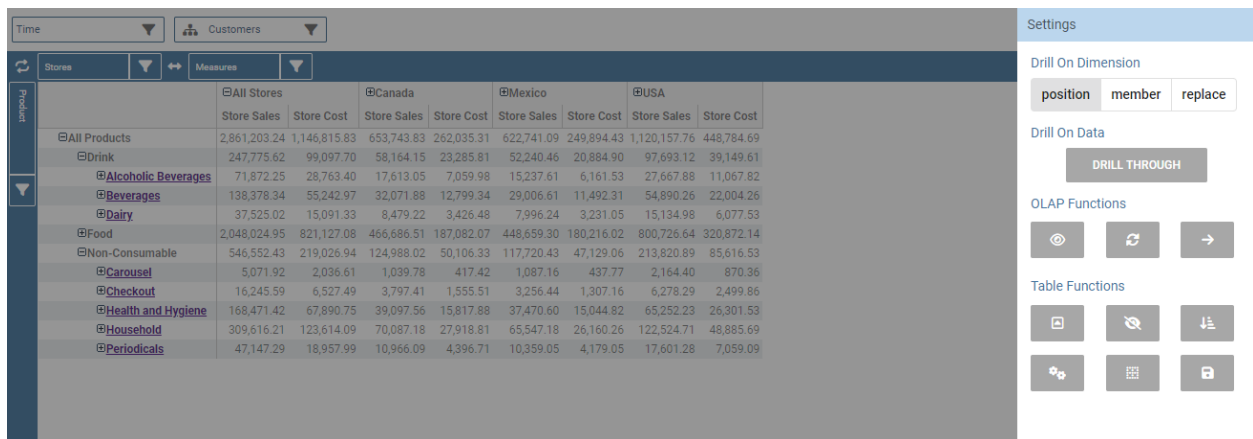
- choose between different data representations,



		All Stores		Canada		Mexico		USA	
		Store Sales	Store Cost	Store Sales	Store Cost	Store Sales	Store Cost	Store Sales	Store Cost
All Products		2,861,203.24	1,146,815.83	653,743.83	262,035.31	622,741.09	249,894.43	1,120,157.76	448,784.69
Drink		247,775.62	99,097.70	58,164.15	23,285.81	52,240.46	20,884.90	97,693.12	39,149.61
Alcoholic Beverages		71,872.25	28,763.40	17,613.05	7,059.98	15,237.61	6,161.53	27,667.88	11,067.82
Beverages		138,378.34	55,242.97	32,071.88	12,799.34	29,006.61	11,492.31	54,890.26	22,004.26
Dairy		37,525.02	15,091.33	8,479.22	3,426.48	7,996.24	3,231.05	15,134.98	6,077.53
Food		2,048,024.95	821,127.08	466,686.51	187,082.07	448,659.30	180,216.02	800,726.64	320,872.14
Non-Consumable		546,552.43	219,026.94	124,988.02	50,106.33	117,720.43	47,129.06	213,820.89	85,616.53
Carousel		5,071.92	2,036.61	1,039.78	417.42	1,087.16	437.77	2,164.40	870.36
Checkout		16,245.59	6,527.49	3,797.41	1,555.51	3,256.44	1,307.16	6,278.29	2,499.86
Health and Hygiene		168,471.42	67,890.75	39,097.56	15,817.88	37,470.60	15,044.82	65,252.23	26,301.53
Household		309,616.21	123,614.09	70,087.18	27,918.81	65,547.18	26,160.26	122,524.71	48,885.69
Periodicals		47,147.29	18,957.99	10,966.09	4,396.71	10,359.05	4,179.05	17,601.28	7,059.09

Fig. 1.366: Open the side bar.

- choose between different drill types,
- call dialogs and functionalities that effect the pivot table.



Time

Customers

Stores

Measures

	All Stores	Canada	Mexico	USA
	Store Sales	Store Cost	Store Sales	Store Cost
All Products	2,861,203.24	1,146,815.83	653,743.83	262,035.31
Drink	247,775.62	99,097.70	58,164.15	23,285.81
Alcoholic Beverages	71,872.25	28,763.40	17,613.05	7,059.98
Beverages	138,378.34	55,242.97	32,071.88	12,799.34
Dairy	37,525.02	15,091.33	8,479.22	3,426.48
Food	2,048,024.95	821,127.08	466,686.51	187,082.07
Non-Consumable	546,552.43	219,026.94	124,988.02	50,106.33
Carousel	5,071.92	2,036.61	1,039.78	417.42
Checkout	16,245.59	6,527.49	3,797.41	1,555.51
Health and Hygiene	168,471.42	67,890.75	39,097.56	15,817.88
Household	309,616.21	123,614.09	70,087.18	27,918.81
Periodicals	47,147.29	18,957.99	10,966.09	4,396.71

Settings

Drill On Dimension

position member replace

Drill On Data

DRILL THROUGH

OLAP Functions

Table Functions

Fig. 1.367: Side bar.

The side bar shows the **Settings**. This area let you customize the Olap layout. As highlighted in the figure below, the Menu is divided in three subsections:

- drill options (a),
- OLAP functions (b),
- table functions (c),
- what if (if it is a what-if document).

We start introducing the interface and leave the description on how they works to the next *Functionalities* paragraph. In particular, referring to next figure, drill types consists of:

- position,
- member,
- replace,
- drill through.

Meanwhile, referring to the following figure, the OLAP functions consist of:

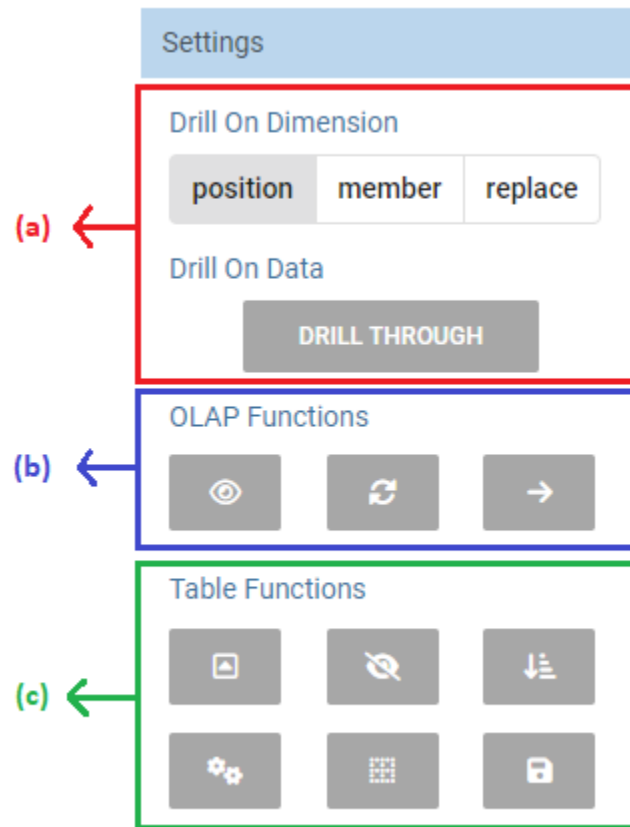


Fig. 1.368: Side bar Menu.

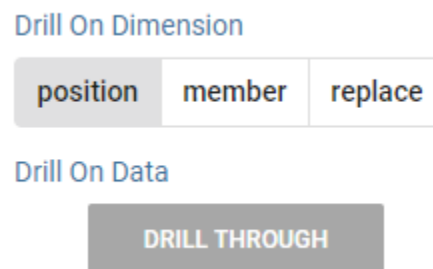


Fig. 1.369: Drill types.

- show MDX Query (a),
- reload model (b),
- enable cross navigation (if enabled and configured by the developer) (c).

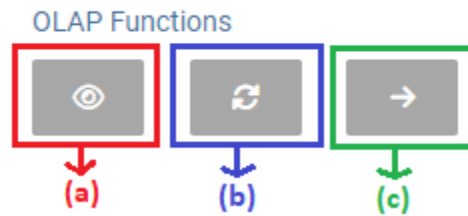


Fig. 1.370: OLAP functions.

Referring to figure below, table functions consist of:

- show parent members (a),
- hide spans (b),
- sorting settings (c),
- show properties (d),
- suppress empty rows/columns (e),
- save customized view (f).

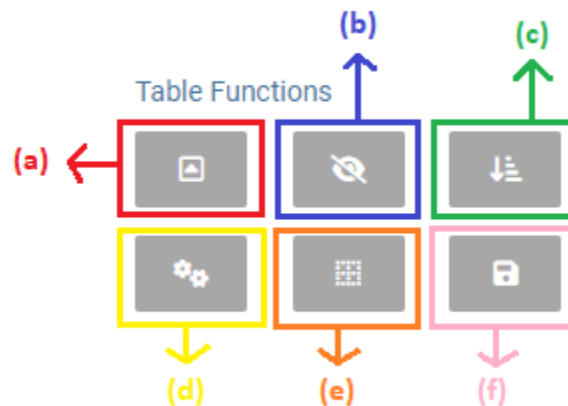


Fig. 1.371: Table functions.

Referring to next figure, if the document is a what-if the what if section consists of:

- lock/unlock model (a),
- save as new version (b),
- undo (c),
- delete versions (d),

- output wizard (e),
- select an algorithm (f).

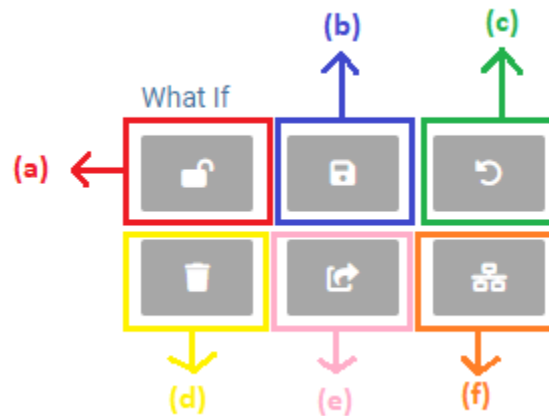


Fig. 1.372: What if.

1.10.1.2 Functionalities

1.10.1.2.1 Placing hierarchies on axes

As we already told, the user can easily move a dimension from the filter bar to the axis or viceversa dragging and dropping it to the desired place.

Let us suppose we want to move a dimension from the filter panel to the columns axis. The steps are summarized in figure below

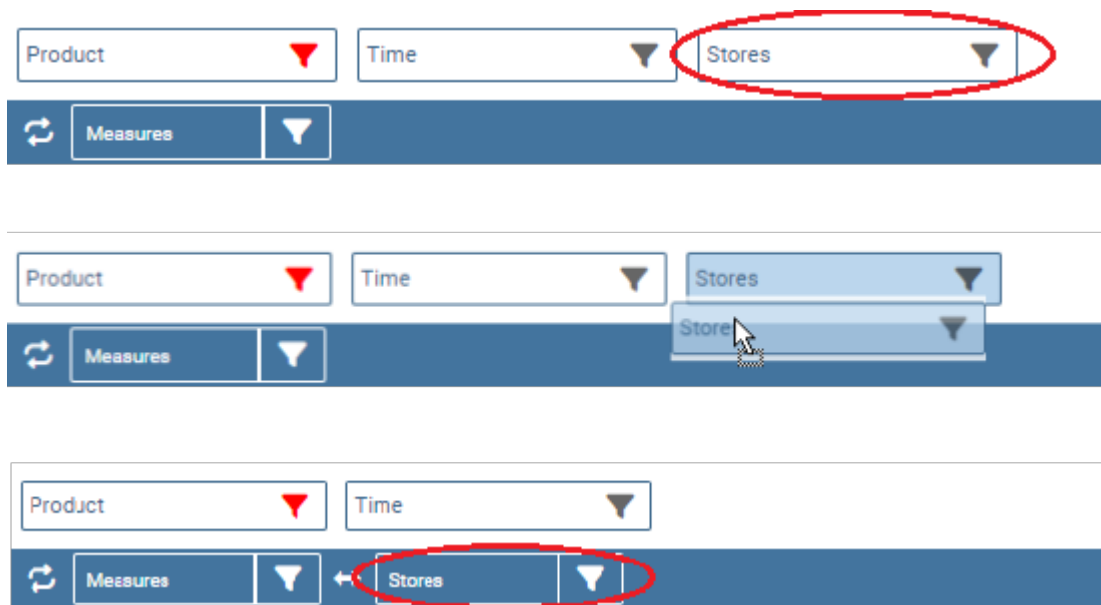


Fig. 1.373: Move a hierarchy to the columns axis.

Vice versa, to move back the dimension from the columns axis to the filter panel the user must simply drag and drop the dimension from one place to the other as in the following figure.

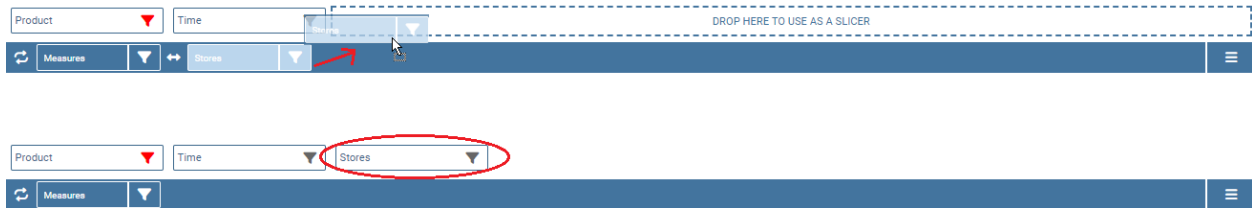


Fig. 1.374: Move a dimension from the columns axis to the filter panel.

Similarly, a dimension can be moved from the filter panel to the rows axis simply dragging and dropping it from one place to the other.

1.10.1.2.2 Swaping axes

To swap axes the user should click on the icon . The user will get the outcome showed in figure below.

The left screenshot shows the pivot table with 'Customers' on the rows axis and 'Store Cost' and 'Store Sales' on the columns axis. The right screenshot shows the pivot table with 'Customers' on the columns axis and 'Store Cost' and 'Store Sales' on the rows axis.

	Store Cost	Store Sales
All Customers	1,146,815.83	2,861,203.24
Canada	103,874.45	258,964.14
Mexico	458,034.06	1,142,058.56
USA	584,907.32	1,460,180.54

	All Customers	Canada	Mexico	USA
Store Cost	1,146,815.83	103,874.45	458,034.06	584,907.32
Store Sales	2,861,203.24	258,964.14	1,142,058.56	1,460,180.54

Fig. 1.375: Swap axes.

1.10.1.2.3 Selecting different hierarchies on dimension

If an OLAP schema is defined, the user can choose different hierarchies of the same dimension. The icon for opening the dialog is positioned on the left of the filter card (if the dimension has more than one hierarchy). Select the hierarchies icon underlined below.

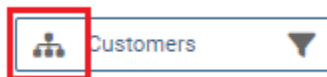


Fig. 1.376: Hierarchies icon.

A pop up will be displayed. The following figure shows its characteristics. Here it is possible to change the hierarchy through the combo-box and then save.

After selecting the hierarchy and saving user's choice, that hierarchy will be used by the pivot table.

If the user re-opens the dialog window, he/she sees the selected hierarchies and has the chance to change it if needed to, as shown below.

We give an example of the output when the hierarchy "All Customers" is selected in first next figure and hierarchy "Customers by segment" in the second next figure.

Customers

The selected hierarchy is **All Customers**. You can change it acting on the form below.

Available Hierarchies

All Customers

CANCEL

SAVE

Fig. 1.377: Hierarchies dialog pop up.

Customers

The selected hierarchy is **All Customers**. You can change it acting on the form below.

Available Hierarchies

All Customers

All Customers

Customer by segment

Fig. 1.378: Changing the hierarchies.

Stores		Measures	
Customers	⊖ All Stores	⊕ Canada	
	Store Cost	Store Sales	Store Cost
	Store Cost	Store Sales	Store Cost
	Store Cost	Store Sales	Store Cost
	Store Cost	Store Sales	Store Cost
Customers	⊖ All Customers	⊕ Mexico	
	1,146,815.83	2,861,203.24	262,035.31
	103,874.45	258,964.14	21,477.44
	458,034.06	1,142,058.56	190,053.85
	584,907.32	1,460,180.54	50,504.02
Customers	⊖ All Customers	⊕ USA	
	1,146,815.83	2,861,203.24	262,035.31
	103,874.45	258,964.14	21,477.44
	458,034.06	1,142,058.56	190,053.85
	584,907.32	1,460,180.54	50,504.02

Fig. 1.379: All Customers hierarchy: the table shows customers by state.

Stores		Measures							
Customers		All Stores		Canada		Mexico		USA	
		Store Cost	Store Sales	Store Cost	Store Sales	Store Cost	Store Sales	Store Cost	Store Sales
	All Customer by segment	1,146,815.83	2,861,203.24	262,035.31	653,743.83	249,894.43	622,741.09	448,784.69	1,120,157.76
	All Clerical	21,042.02	52,453.22	3,562.29	8,898.61	7,199.97	18,005.68	5,283.14	13,108.19
	All Management	164,779.24	411,269.87	35,721.26	89,177.19	36,172.19	90,373.45	61,639.41	153,762.15
	All Manual	277,241.55	692,243.77	64,801.54	161,762.63	58,524.05	145,910.20	113,219.56	282,774.34
	All Professional	371,910.04	928,094.10	91,107.91	227,243.00	77,251.89	192,467.81	149,893.49	374,462.56
	All Skilled Manual	311,842.98	777,142.28	66,842.31	166,662.40	70,746.32	175,983.94	118,749.10	296,050.52

Fig. 1.380: Customers by segment hierarchy: table shows customers by segment.

1.10.1.2.4 Slicing

The slicing operation consists in the analysis of a subset of a multi-dimensional array corresponding to a single value for one or more members of the dimensions. In order to perform this operation you need to drag and drop the dimension of interest in the axis panel. Then clicking on the filter icon choose the new single focus and apply it. Once concluded these steps the cube will show only the selected level of the dimension, while the others have been sliced out.

The following figure shows the slicer option panel where the user can see which items is selected and used in the pivot table.

Customers

Selected items are used as filters in the main table. You can remove some or clicking on add selection the whole tree will be unlocked

- ☒ All Customers
 - ☒ Canada
 - ☒ Mexico
 - ☒ USA

CLEAR

ADD

CANCEL

APPLY

Fig. 1.381: Dialog for slicer choosing.

In order to unlock the whole member tree and to define a new slicing you have to click on the add button. As you can see in the following figure, here it is possible to navigate the member tree, search for a specific value in the member tree, clear all the selections and apply a new one.

In particular, it is possible to search for a member value in two ways:

1. by browsing the member tree;
2. by typing member's name or it's part in the input field. The research will be possible if the user enters at least three letters;

Customers

Selected items will be used as filters in the main table.

search

- ✓ All Customers
 - > ✓ Canada
 - > ✓ Mexico
 - > ✓ USA

CLEAR CANCEL APPLY

Fig. 1.382: Define a new slicing.

Customers

Selected items will be used as filters in the main table.

search

- ✓ All Customers
 - ✓ Canada
 - BC
 - Burnaby
 - Alexandra Wellington
 - Ana Quick
 - Anastacio Butcher
 - Ann Pearce

CLEAR CANCEL APPLY

Fig. 1.383: Browsing the member tree.

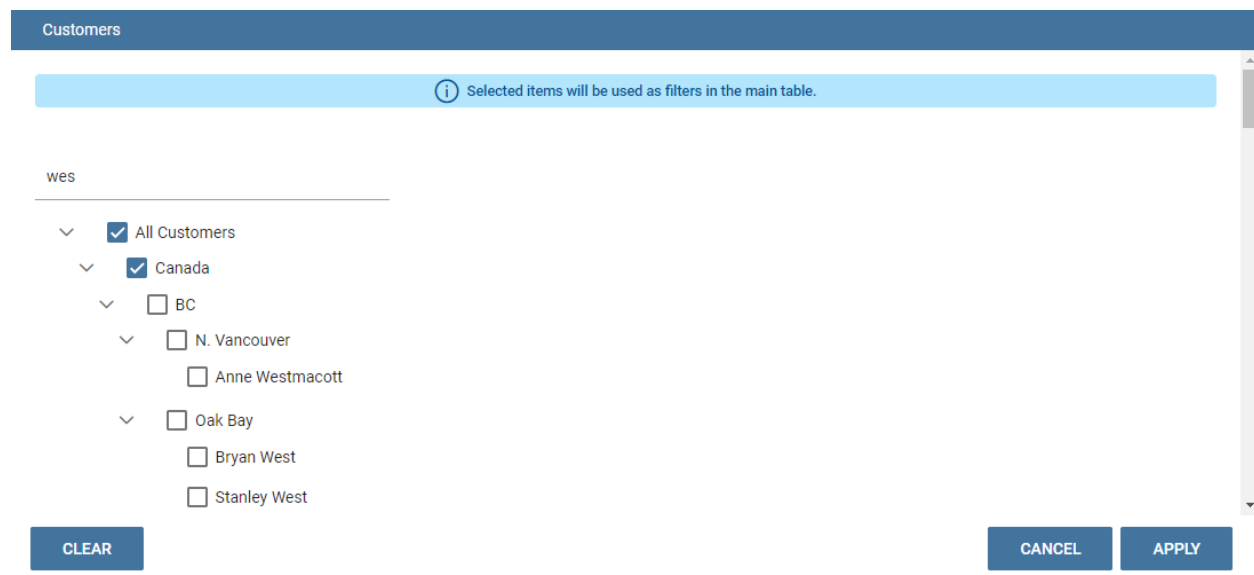


Fig. 1.384: Using the research box.

The check on the checkbox of the selected values and click on the apply button to save the selection. After that, the users choice will affect the pivot table, example is given in the figure below.

	All Stores		Canada		Mexico		USA	
	Store Cost	Store Sales	Store Cost	Store Sales	Store Cost	Store Sales	Store Cost	Store Sales
All Customers	1,146,815.83	2,861,203.24	262,035.31	653,743.83	249,894.43	622,741.09	448,784.69	1,120,157.76
Canada	103,874.45	258,964.14	21,477.44	53,547.19				
Anne Westmacott	34.36	80.83						
Mexico	458,034.06	1,142,058.56	190,053.85	474,145.76	245,527.23	611,768.14		
USA	584,907.32	1,460,180.54	50,504.02	126,050.88	4,367.20	10,972.96	448,784.69	1,120,157.76

Fig. 1.385: Results for slicing operation.

1.10.1.2.5 Filtering

To filter dimension members in a pivot table, the user should click on the funnel icon located on the right side of dimension's filter card placed in the filter area.

The procedure to search for a member using the filter dialog has no meaningful differences with the one described for the slicer chooser dialog. The pop up interface is the one showed below. After selecting a member, the user should click on the apply button in order to filter the values in the pivot. The pivot table will then display the changements. Otherwise click on the cancel button to discard changes.

When a filter is applied on a card the filter button becomes red, as shown in the picture below.

1.10.1.2.6 Drill down and drill up

User can choose between different drill types by clicking on one of the three buttons in the "Drill On Dimensions" section of the side bar. There are three drill types. In the following we give some details on them.

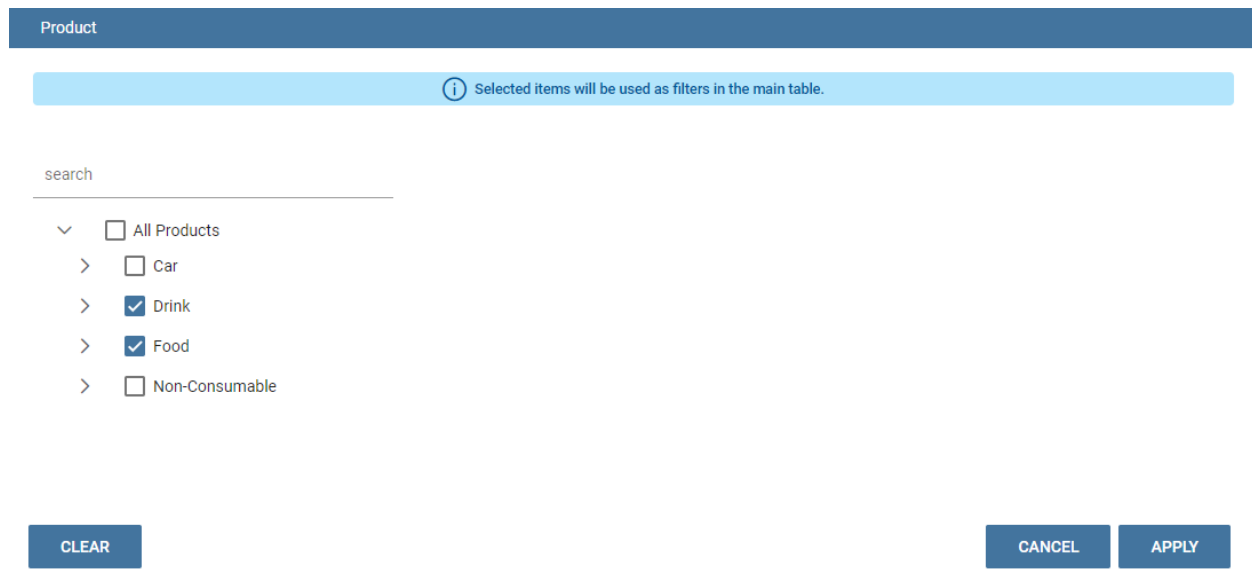


Fig. 1.386: Filter dialog.



Fig. 1.387: Filter icon when a filter is applied.

1. **Position:** this is the default drill type. Clicking on a drill down/drill up command will expand/collapse a pivot table with child members of a member. See below.
2. **Member:** if the user wants to perform drill operation not only on one member per time but on all members of the same name and level at the same time it is needed to select member drill type. See below.
3. **Replace:** this option lets the user replace the parent member with his child member during drill down operation. To drill up the user should click on the arrow icon next to the dimension name on which to perform operation. See figure below.

1.10.1.2.7 Drill through

To perform drill through operation the user needs to click on the corresponding button in the side bar. Then clicking on the magnifying glass button in one of the pivot cell a dialog will open with results (this pop up could take some time to open).

In particular, referring to the next figure, drill though dialog consists of:

- a hierarchy menu,
- a table of values,
- a maximum rows drop down list,
- an apply button,
- an export button,
- a cancel button,

Figure 1.388 shows two screenshots of the Knowage interface illustrating a "Position" drill down. The interface includes a sidebar with "Customers" and "Product" sections, and a main table with columns for "Measures", "Store Cost", and "Store Sales".

Left Screenshot (Initial State):

Measures	Store Cost	Store Sales
All Customers	All Products	1,146,815.83
Canada	All Products	103,815.83
Mexico	All Products	458,024.95
USA	All Products	584,965.05

Right Screenshot (Expanded State):

Measures	Store Cost	Store Sales
All Customers	All Products	1,146,815.83
Canada	All Products	103,815.83
Canada	Car	10,381.58
Canada	Drink	10,381.58
Canada	Food	7,281.19
Canada	Non-Consumable	2,337.48
Mexico	All Products	458,024.95
USA	All Products	584,965.05

Fig. 1.388: "Position" drill down.

Figure 1.389 shows two screenshots of the Knowage interface illustrating a "Member" drill down. The interface includes a sidebar with "Customers" and "Product" sections, and a main table with columns for "Measures", "Store Cost", and "Store Sales".

Left Screenshot (Initial State):

Measures	Store Cost	Store Sales
All Customers	All Products	1,146,815.83
Canada	All Products	103,815.83
Mexico	All Products	458,024.95
USA	All Products	584,965.05

Right Screenshot (Expanded State):

Measures	Store Cost	Store Sales
All Customers	All Products	1,146,815.83
All Customers	Car	10,381.58
All Customers	Drink	10,381.58
All Customers	Food	7,281.19
All Customers	Non-Consumable	2,337.48
Canada	All Products	103,815.83
Canada	Car	10,381.58
Canada	Drink	10,381.58
Canada	Food	7,281.19
Canada	Non-Consumable	2,337.48
Mexico	All Products	458,024.95
Mexico	Car	10,381.58
Mexico	Drink	10,381.58
Mexico	Food	7,281.19
Mexico	Non-Consumable	2,337.48

Fig. 1.389: "Member" drill down.

Figure 1.390 shows two screenshots of the Knowage interface illustrating a "Replace" drill down. The interface includes a sidebar with "Product" and "Measures" sections, and a main table with columns for "Store Cost" and "Store Sales".

Left Screenshot (Initial State):

Product	Store Cost	Store Sales
All Products	1,146,815.83	2,861,203.24

Right Screenshot (Expanded State):

Product	Store Cost	Store Sales
Car	99,097.70	247,775.62
Drink	99,097.70	247,775.62
Food	821,127.08	2,048,024.95
Non-Consumable	219,026.94	546,552.43

Fig. 1.390: "Replace" drill down.

Measures			
Product		Store Cost	Store Sales
	[-] All Products	1,146,815.83	2,861,203.24
	[+] Car		
	[-] Drink	99,097.70	247,775.62
	[-] Alcoholic Beverages	28,763.40	71,872.25
	Beer and Wine	28,763.40	71,872.25
	[+] Beverages	55,242.97	138,378.34
	[+] Dairy	15,091.33	37,525.02
	[+] Food	821,127.08	2,048,024.95
	[+] Non-Consumable	219,026.94	546,552.43

Fig. 1.391: Drill thorough option.

- a clear all button.

Drill Through

Select visible levels:

Measures Product Time Stores All Customers

CLEAR ALL

Q search

Maximum Rows 10

PRODUCT FAMILY ↑↓	PRODUCT DEPARTMENT ↑↓	PRODUCT CATEGORY ↑↓	YEARS ↑↓	MONTHS ↑↓	STORE COST ↑↓	UNIT SALES ↑↓	STORE SALES ↑↓
Drink	Alcoholic Beverages	Beer and Wine	2014	April	3.2631	4.2857	6.9429
Drink	Alcoholic Beverages	Beer and Wine	2014	April	3.7029	5.7143	9.2571
Drink	Alcoholic Beverages	Beer and Wine	2014	April	3.402	4.2857	6.9429
Drink	Alcoholic Beverages	Beer and Wine	2014	April	2.9897	4.2857	9.3429
Drink	Alcoholic Beverages	Beer and Wine	2014	April	1.62	2.8571	4.6286
Drink	Alcoholic Beverages	Beer and Wine	2014	April	4.3509	5.7143	9.2571
Drink	Alcoholic Beverages	Beer and Wine	2014	April	4.3509	5.7143	9.2571
Drink	Alcoholic Beverages	Beer and Wine	2014	April	3.0549	4.2857	6.9429
Drink	Alcoholic Beverages	Beer and Wine	2014	April	3.3326	4.2857	6.9429
Drink	Alcoholic Beverages	Beer and Wine	2014	April	3.6437	4.2857	9.3429

CANCEL EXPORT APPLY

Fig. 1.392: Drill thorough window.

Here the user can choose the level of detail with which data will be displayed thorough the hierachy menu. The steps to follow are:

1. click on a hierarchy in hierarchy menu,
2. check the checkbox of the level,
3. click on the “Apply” button.

The user can also select the maximum rows to load by choosing one of the options in the drop down list. Finally, loaded data can be exported in csv format by clicking on the “Export” button.

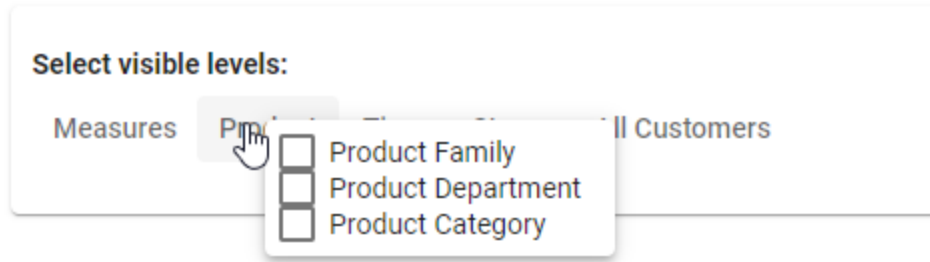


Fig. 1.393: Checkbox of the level.

1.10.1.2.8 Refreshing model

To refresh a loaded model the user needs to click on the “Refresh” button available in the side bar panel. This action will clear the cash, load pivot table and the rest of data again.

1.10.1.2.9 Showing MDX

To show current mdx query user should click on show mdx button in the side bar. Figure below shows an example.

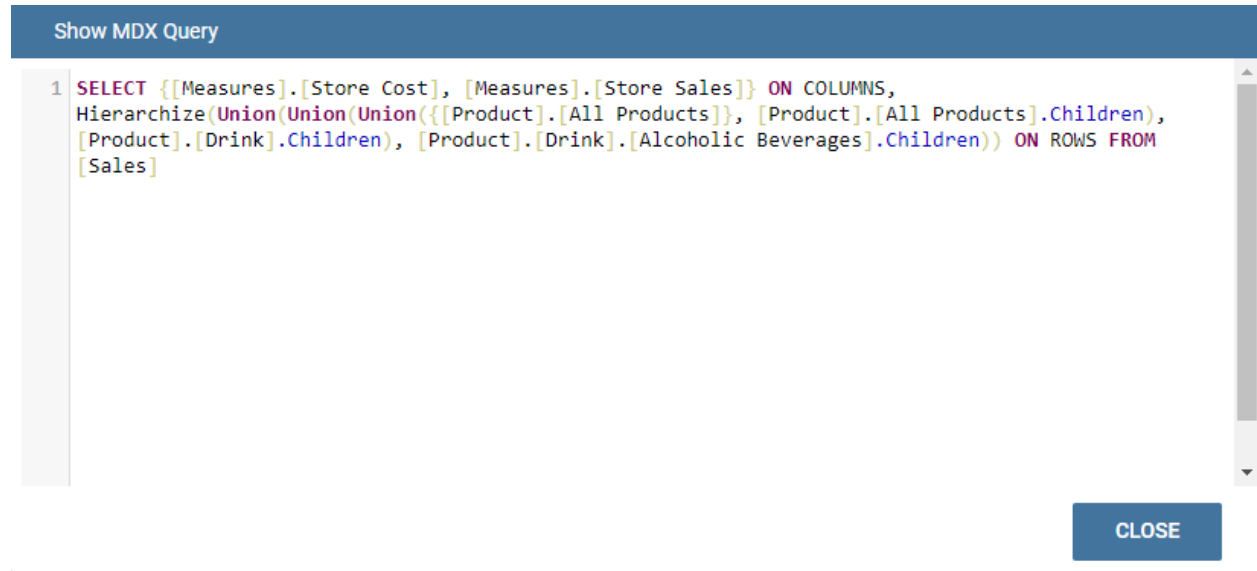


Fig. 1.394: Showing MDX query example.

1.10.1.2.10 Showing parent members

If a user wants to see additional information about members shown in the pivot table (for example: member’s hierarchy, level or parent member) he should click on show parent members button in the side bar panel. The result will be visible in the pivot table. An example is shown in the following two figures.

	Store Sales				Store Cost			
	⊖All Stores	⊕Canada	⊕Mexico	⊕USA	⊖All Stores	⊕Canada	⊕Mexico	⊕USA
⊖All Products	2,861,203.24	653,743.83	622,741.09	1,120,157.76	1,146,815.83	262,035.31	249,894.43	448,784.69
⊖Drink	247,775.62	58,164.15	52,240.46	97,693.12	99,097.70	23,285.81	20,884.90	39,149.61
⊕Alcoholic Beverages	71,872.25	17,613.05	15,237.61	27,667.88	28,763.40	7,059.98	6,161.53	11,067.82
⊕Beverages	138,378.34	32,071.88	29,006.61	54,890.26	55,242.97	12,799.34	11,492.31	22,004.26
⊕Dairy	37,525.02	8,479.22	7,996.24	15,134.98	15,091.33	3,426.48	3,231.05	6,077.53
⊕Food	2,048,024.95	466,686.51	448,659.30	800,726.64	821,127.08	187,082.07	180,216.02	320,872.14
⊕Non-Consumable	546,552.43	124,988.02	117,720.43	213,820.89	219,026.94	50,106.33	47,129.06	85,616.53

Fig. 1.395: Pivot table without the parent members mode.

Product (All)			Measures								
			Store Sales				Store Cost				
			Stores				Stores				
			⊖All Stores	All Stores	⊖Mexico	⊖USA	⊖All Stores	All Stores	⊖Mexico	⊖USA	
⊖All Products			2,861,203.24	653,743.83	622,741.09	1,120,157.76	1,146,815.83	262,035.31	249,894.43	448,784.69	
All Products	⊖Drink		247,775.62	58,164.15	52,240.46	97,693.12	99,097.70	23,285.81	20,884.90	39,149.61	
	Drink	⊖Alcoholic Beverages	71,872.25	17,613.05	15,237.61	27,667.88	28,763.40	7,059.98	6,161.53	11,067.82	
		⊖Beverages	138,378.34	32,071.88	29,006.61	54,890.26	55,242.97	12,799.34	11,492.31	22,004.26	
		⊖Dairy	37,525.02	8,479.22	7,996.24	15,134.98	15,091.33	3,426.48	3,231.05	6,077.53	
	⊖Food		2,048,024.95	466,686.51	448,659.30	800,726.64	821,127.08	187,082.07	180,216.02	320,872.14	
	⊖Non-Consumable		546,552.43	124,988.02	117,720.43	213,820.89	219,026.94	50,106.33	47,129.06	85,616.53	

Fig. 1.396: Pivot table after the parent members selection.

1.10.1.2.11 Hiding/showing spans

To hide or show spans the user should click on show/hide spans button in the side bar. The result will be visible in pivot table as in figure below.

		Store Sales	Store Cost
⊕All Products	⊖All Stores	2,861,203.24	1,146,815.83
	⊕Canada	653,743.83	262,035.31
	⊕Mexico	622,741.09	249,894.43
	⊕DF	253,551.24	101,622.87
	⊕Veracruz	160,319.40	64,320.05
	⊕Zacatecas	208,870.46	83,951.51
	⊕USA	1,120,157.76	448,784.69
	⊕CA	271,429.98	108,882.76
	⊕OR	341,223.87	136,668.76
	⊕WA	507,503.91	203,233.18

		Store Sales	Store Cost
⊕All Products	⊖All Stores	2,861,203.24	1,146,815.83
⊕All Products	⊕Canada	653,743.83	262,035.31
⊕All Products	⊕Mexico	622,741.09	249,894.43
⊕All Products	⊕DF	253,551.24	101,622.87
⊕All Products	⊕Veracruz	160,319.40	64,320.05
⊕All Products	⊕Zacatecas	208,870.46	83,951.51
⊕All Products	⊕USA	1,120,157.76	448,784.69
⊕All Products	⊕CA	271,429.98	108,882.76
⊕All Products	⊕OR	341,223.87	136,668.76
⊕All Products	⊕WA	507,503.91	203,233.18

Fig. 1.397: Hide/show spans.

1.10.1.2.12 Showing properties

In OLAP schema the XML member properties, if configured, is represented as part of pivot table where property values are placed in rows and columns. To get these values, the user needs to click on show properties button in the side bar. Results will be shown in the pivot table;

⊖All Stores	Store Type	Store Manager	Store Sqft	Grocery Sqft	Frozen Sqft	Meat Sqft	Has coffee bar	Street address
⊖Canada	Store Type	Store Manager	Store Sqft	Grocery Sqft	Frozen Sqft	Meat Sqft	Has coffee bar	Street address
⊖BC	Store Type	Store Manager	Store Sqft	Grocery Sqft	Frozen Sqft	Meat Sqft	Has coffee bar	Street address
⊖Vancouver	Store Type	Store Manager	Store Sqft	Grocery Sqft	Frozen Sqft	Meat Sqft	Has coffee bar	Street address
Store 19	Deluxe Supermarket	Ruth	23112	16418	4016	2678	1	6644 Sudance Drive
⊖Victoria	Store Type	Store Manager	Store Sqft	Grocery Sqft	Frozen Sqft	Meat Sqft	Has coffee bar	Street address
⊖Mexico	Store Type	Store Manager	Store Sqft	Grocery Sqft	Frozen Sqft	Meat Sqft	Has coffee bar	Street address
⊖USA	Store Type	Store Manager	Store Sqft	Grocery Sqft	Frozen Sqft	Meat Sqft	Has coffee bar	Street address

Fig. 1.398: Show properties.

1.10.1.2.13 Suppressing empty columns/rows

To hide the empty rows and/or columns, if any, from pivot table the user can click on the “Suppress empty rows/columns” button in the side bar panel. An example is given in Figure below.

	Store Sales		
	⊖All Stores	⊖#null	⊖Canada
⊖All Products	2,861,203.24		653,743.83
⊖Car			
⊖Drink	247,775.62		58,164.15
⊖Alcoholic Beverages	71,872.25		17,613.05
⊖Beverages	138,378.34		32,071.88
⊖Dairy	37,525.02		8,479.22
⊖Food	2,048,024.95		466,686.51
⊖Non-Consumable	546,552.43		124,988.02

	Store Sales	
	⊖All Stores	⊖Canada
⊖All Products	2,861,203.24	653,743.83
⊖Drink	247,775.62	58,164.15
⊖Alcoholic Beverages	71,872.25	17,613.05
⊖Beverages	138,378.34	32,071.88
⊖Dairy	37,525.02	8,479.22
⊖Food	2,048,024.95	466,686.51
⊖Non-Consumable	546,552.43	124,988.02

Fig. 1.399: Suppressing empty columns/rows.

1.10.1.2.14 Sorting

To enable member ordering the user must click on the “Sorting settings” button in the side bar panel. The command for sorting will appear next to the member’s name in the pivot table, as shown below.

	Store Cost				Store Sales			
	⊖All Products	⊖Food	⊖Non-Consumable	⊖Drink	⊖All Products	⊖Food	⊖Drink	⊖Non-Consumable
⊖All Stores	432,574.29	311,639.56	83,427.50	37,498.67	1,079,167.39	777,256.74	93,742.16	208,148.57
⊖USA	220,645.11	158,856.35	42,485.06	19,303.70	550,808.42	396,598.56	48,182.01	106,027.85
⊖Mexico	172,596.61	124,751.27	33,112.69	14,724.09	430,313.51	310,730.21	36,890.31	82,673.07
⊖Canada	39,332.57	28,031.94	7,829.76	3,470.87	98,045.46	69,927.97	8,669.84	19,447.65

Fig. 1.400: Member sorting.

To sort members the user needs to click on the sorting command (two opposite arrows) available next to each member of the pivot table. Note that the sorting criteria is descending at first execution and it represented by a red down arrow. If the user clicks again on the sorting icon, criteria will change to ascending and the icon becomes an upper green arrow. To remove the sorting, the user just have to click on the icon again.

To change sorting mode user should click on sorting settings button in the side bar.

The available types of sorting are:

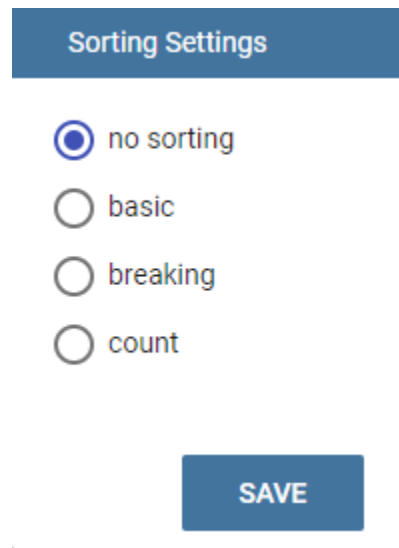
The image shows a 'Sorting Settings' window. It has a title bar at the top with the text 'Sorting Settings'. Below the title bar, there are four radio button options: 'no sorting' (which is selected), 'basic', 'breaking', and 'count'. At the bottom of the window, there is a blue button labeled 'SAVE'.

Fig. 1.401: Sorting settings window.

- no sorting (it is the default);
- basic, it is the standard ascending or descending order according to the column values where the ordering is done;
- breaking, it means that the hierarchy will be broken;
- count, only the top or last members will be shown in the pivot table; the user can change the number of members shown (by default first or last 10) using the number input field that appears clicking on this type of sorting.

1.10.1.3 Creation of an OLAP document

Multidimensional analysis allows the hierarchical inquiry of numerical measures over predefined dimensions. In Cockpit we explained how the user can monitor data on different detail levels and from different perspectives. Here we want to go into details of how a technical user can create an OLAP document. We recall that the main characteristics of OLAP documents are:

- the need for a specific data structure (logical or physical);
- analysis based on dimensions, hierarchies and measures;
- interactive analysis;
- freedom to re-orient analysis;
- different levels of data analysis, through synthetic and detailed views;
- drill-down, slice and dice, drill-through operations.

Considering these items, we will describe the steps to develop an OLAP document.

1.10.1.3.1 About the engine

Knowage performs OLAP documents by relying on the **OLAP engine**. This engine integrates Mondrian OLAP server and two different cube navigation clients to provide multi-dimensional analysis. In general, Mondrian is a Relational Online Analytical Processing (ROLAP) tool that provides the back-end support for the engine. OLAP structures,

such as cubes, dimensions and attributes, are mapped directly onto tables and columns of the data warehouse. This way, Mondrian builds an OLAP cube in cache that can be accessed by client applications. The Knowage OLAP engine provides the front-end tool to interact with Mondrian servers and shows the results via the typical OLAP functionalities, like drill down, slicing and dicing on a multi-dimensional table. Furthermore, it can also interact with XMLA servers. This frontend translates user's navigation actions into MDX queries on the multi-dimensional cube, and show query results on the table he is navigating.

1.10.1.3.2 Development of an OLAP document

The creation of an OLAP analytical document requires the following steps:

- schema modelling;
- catalogue configuration;
- OLAP cube template building;
- analytical document creation.

1.10.1.3.2.1 Schema modelling

The very first step for a multi-dimensional analysis is to identify essential information describing the process/event under analysis and to consider how it is stored and organized in the database. On the basis of these two elements, a mapping process should be performed to create the multi-dimensional model.

Hint: From the relational to the multi-dimensional model

The logical structure of the database has an impact on the mapping approach to be adopted when creating the multidimensional model, as well as on query performances.

If the structure of the relational schema complies with multi-dimensional logics, it will be easier to map the entities of the physical model onto the metadata used in Mondrian schemas. Otherwise, if the structure is highly normalized and scarcely dimensional, the mapping process will probably require to force and approximate the model to obtain a multi-dimensional model. As said above, Mondrian is a ROLAP tool. As such, it maps OLAP structures, such as cubes, dimensions and attributes directly on tables and columns of a relational data base via XMLbased files, called Mondrian schemas. Mondrian schemas are treated by Knowage as resources and organized into catalogues. Hereafter, an example of Mondrian schema:

Listing 1.32: Mondrian schema example

```
1  <?xml version="1.0"?>
2  <Schema name="FoodMart">
3      <!-- Shared dimensions -->
4      <Dimension name="Customers">
5
6          <Hierarchy hasAll="true" allMemberName="All Customers"
7              primaryKey=" customer_id">
8
9              <Table name="customer"/>
10             <Level name="Country" column="country" uniqueMembers="true"/>
11             <Level name="State Province" column="state_province"
12                 uniqueMembers="true"/>
13             <Level name="City" column="city" uniqueMembers="false"/>
14
15         </Hierarchy> ...
16
```

(continues on next page)

(continued from previous page)

```

17     </Dimension> ...
18
19     <!-- Cubes -->
20     <Cube name="Sales">
21
22         <Table name="sales_fact_1998"/>
23
24         <DimensionUsage name="Customers" source="Customers"
25             foreignKey="customer_id" /> ...
26
27         <!-- Private dimensions -->
28
29         <Dimension name="Promotion Media" foreignKey="promotion_id">
30
31             <Hierarchy hasAll="true" allMemberName="All Media"
32                 primaryKey="promotion_id">
33                 <Table name="promotion"/>
34                 <Level name="Media Type" column="media_type" uniqueMembers="true"/>
35             </Hierarchy>
36
37         </Dimension> ...
38
39         <!-- basic measures-->
40
41         <Measure name="Unit Sales" column="unit_sales" aggregator="sum"
42             formatString="#,###.00"/>
43
44         <Measure name="Store Cost" column="store_cost" aggregator="sum"
45             formatString="#,###.00"/>
46
47         <Measure name="Store Sales" column="store_sales" aggregator="sum"
48             formatString="#,###.00"/>
49         ...
50
51         <!-- derived measures-->
52
53         <CalculatedMember name="Profit" dimension="Measures">
54             <Formula>
55                 [Measures].[Store Sales] - [Measures].[Store Cost]
56             </Formula>
57             <CalculatedMemberProperty name="format_string" value="$#,##0.00"/>
58         </CalculatedMember>
59
60     </Cube>
61     ...
62 </Schema>

```

Each mapping file contains one schema only, as well as multiple dimensions and cubes. Cubes include multiple dimensions and measures. Dimensions include multiple hierarchies and levels. Measures can be either primitive, i.e., bound to single columns of the fact table, or calculated, i.e., derived from calculation formulas that are defined in the schema. The schema also contains links between the elements of the OLAP model and the entities of the physical model: for example, <table> sets a link between a cube and its dimensions, while the attributes primaryKey and foreignKey reference integrity constraints of the star schema.

Note: Mondrian

For a detailed explanation of Mondrian schemas, please refer to the documentation available at the official project webpage: <http://mondrian.pentaho.com/documentation>.

1.10.1.3.2.2 Engine catalogue configuration

To reference an OLAP cube, first insert the corresponding Mondrian schema into the catalogue of schemas managed by the engine. In order to do this, go to **Catalogs > Mondrian schemas** in the Knowage menu, as shown below.

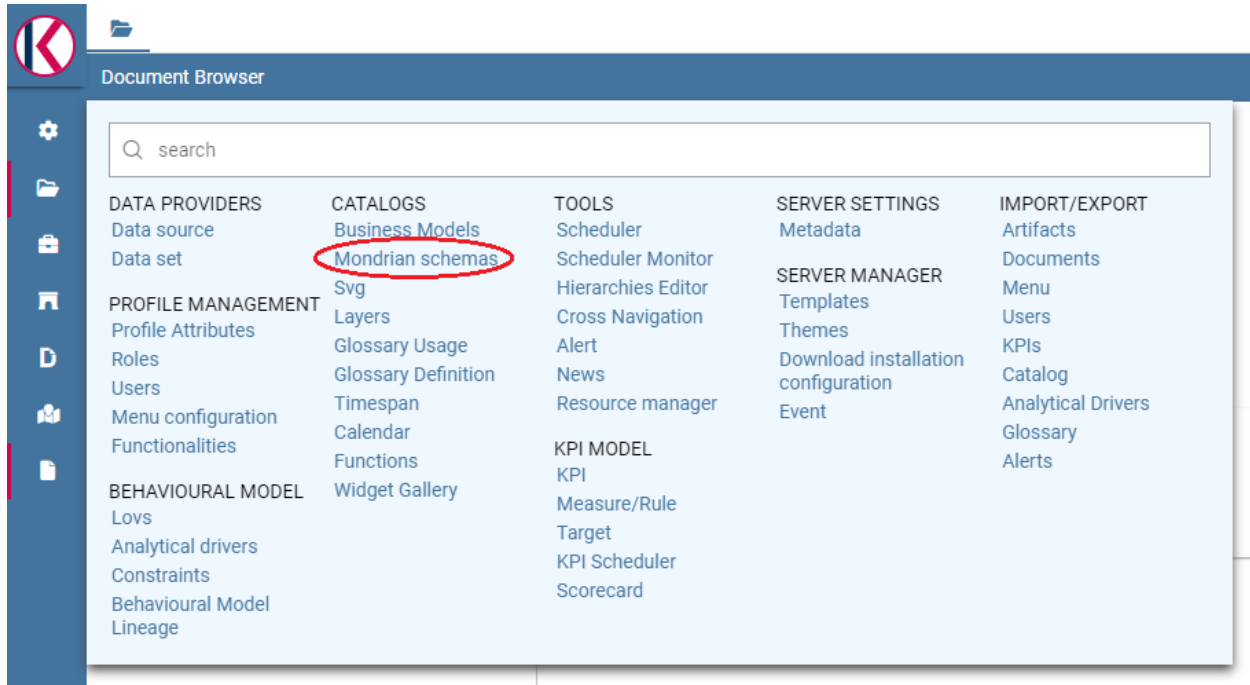


Fig. 1.402: Mondrian schemas menu item.

Here you can find the list of already created mondrian schemas and clicking on the plus button you can define a new one uploading your XML schema file. A new window will open where you have to choose a **Name**, an optional **Description** and to upload your XML file, as you can see in figure below.

When creating a new OLAP template, you will choose among the available cubes defined in the registered schemas.

1.10.1.3.2.3 OLAP template building

Once the cube has been created, you need to build a template which maps the cube to the analytical document. To accomplish this goal the user can manually edit the template or use the guided Knowage designer (look at the “OLAP Designer” section for this functionality). The template is an XML file telling Knowage OLAP engine how to navigate the OLAP cube and has a structure like the one represented in next code:

Listing 1.33: Mapping template example

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <olap>
3    <!-- schema configuration -->
4    <cube reference="FoodMart"/>
5
6    <MDXMondrianQuery>
7      SELECT {[Measures].[Unit Sales]} ON COLUMNS
8      , {[Region].[All Regions]} ON ROWS
9      FROM [Sales]
10     WHERE [Product].[All Products].[Drink]
11    </MDXMondrianQuery>

```

(continues on next page)

NEW_SCHEMA 📄 ✕

Detail

Work Flow

Name *

NEW_SCHEMA

Description

Scegli file

category_management.xml

📁

Saved Versions

🔍 search

Active

File Name

↑↓

Creator

↑↓

Creation Date

No data found

Fig. 1.403: Creating a new mondrian schema.

(continued from previous page)

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

```
<!-- query configuration -->
<MDXquery>
  SELECT {[Measures].[Unit Sales]} ON COLUMNS
    , {[Region].[All Regions]} ON ROWS
  FROM [Sales]
  WHERE [Product].[All Products].[${family}]
  <parameter name="family" as="family"/>
</MDXquery>

<!-- toolbar configuration -->
<TOOLBAR>
  <BUTTON_MDX visible="true" menu="false" />
  <BUTTON_FATHER_MEMBERS visible="true" menu="false"/>
  <BUTTON_HIDE_SPANS visible="true" menu="false"/>
  <BUTTON_SHOW_PROPERTIES visible="true" menu="false"/>
  <BUTTON_HIDE_EMPTY visible="true" menu="false" />
  <BUTTON_FLUSH_CACHE visible="true" menu="false" />
</TOOLBAR>

</olap>
```

An explanation of different sections of Mapping template example follows.

- The CUBE section sets the Mondrian schema. It should reference the exact name of the schema, as registered in the catalogue on the Server.
- The MDXMondrianQuery section contains the original MDX query defining the starting view (columns and rows) of the OLAP document.
- The MDX section contains a variation of the original MDX query, as used by the Knowage Engine. This version includes parameters (if any). The name of the parameter will allow Knowage to link the analytical driver associated to the document via the parameter (on the Server).
- The TOOLBAR section is used to configure visibility options for the side bar in the OLAP document. The exact meaning and functionalities of each toolbar button has been explained in “Functionalities” section. A more

complete list of the available options is shown in Menu configurable options in the Knowage designer.

Listing 1.34: Menu configurable options

```

1 <BUTTON_DRILL_THROUGH visible="true"/>
2 <BUTTON_MDX visible="true"/>
3 <BUTTON_FATHER_MEMBERS visible="true"/>
4 <BUTTON_HIDE_SPANS visible="true"/>
5 <BUTTON_SORTING_SETTINGS visible="true"/>
6 <BUTTON_SHOW_PROPERTIES visible="true"/>
7 <BUTTON_HIDE_EMPTY visible="true"/>
8 <BUTTON_FLUSH_CACHE visible="true"/>
9 <!-- toolbar configuration for what-if documents: -->
10 <BUTTON_SAVE_NEW visible="true"/>
11 <BUTTON_UNDO visible="true"/>
12 <BUTTON_VERSION_MANAGER visible="true"/>
13 <BUTTON_EXPORT_OUTPUT visible="false"/>
14 <BUTTON_SAVE_SUBOBJECT clicked="false" visible="true"/>
15 <BUTTON_EDITABLE_EXCEL_EXPORT clicked="false" visible="true"/>
16 <BUTTON_ALGORITHMS clicked="false" visible="true"/>

```

1.10.1.3.2.4 Creating the analytical document

Once you have the template ready you can create the OLAP document on Knowage Server.

To create a new OLAP document, click on the plus button in the **Document Browser** area and then choose “Generic document”. Filling in the mandatory fields: select a Label and a Name, select **On-Line Analytical Processing** as Type and **OLAP Engine** as Engine, add the Data Source from which the data comes from and the State of the document. Finally, upload the XML template developed in the previous section and click on save.

Fig. 1.404: OLAP document creation interface.

You will see the document in the functionality (folder) you selected.

1.10.1.3.3 OLAP Designer

Knowage Server is also endowed of an efficient OLAP designer which avoid the user to edit manually the XML-based template that we discussed on in Development of an OLAP document. We will therefore describe here all features of this functionality.

The user needs to have a functioning Modrian schema to start the work with. As we have already seen in the previous sections, select **Mondrian Schemas** to check the available Mondrian schemas on server. It is mandatory that the chosen Mondrian schema has no parameters applied.

Warning: Mondrian schema for OLAP designer

If you want to use the designer the Mondrian schema must not be filtered thorough any parameter or profile attribute.

The page as the one in figure below will open.

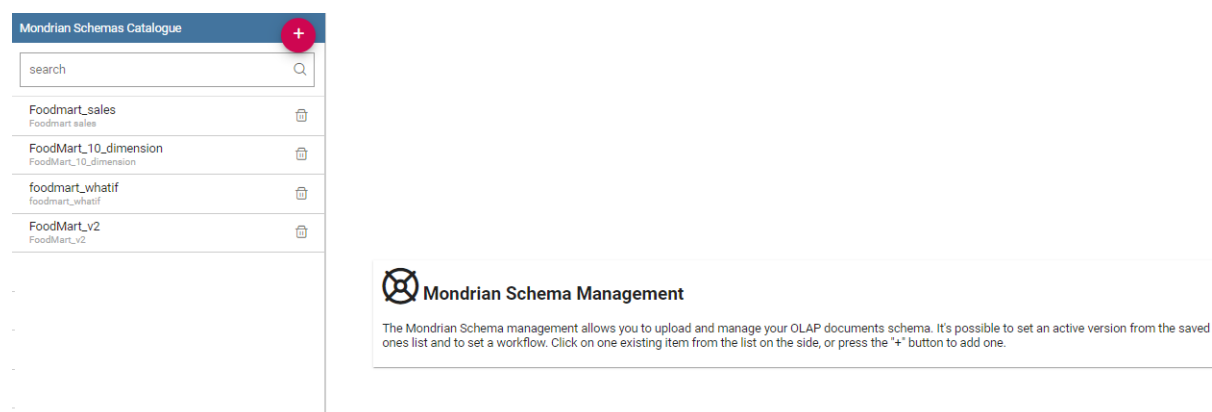


Fig. 1.405: Catalogs list of Schema Mondrian.

Then enter in the **Document Browser**, click on the “Plus” icon at the top right corner of the page and choose “Gneric document”. As described in the previous section, fill in the mandatory boxes as Label and Name of the document, select the **On-Line Analytical Processing** Type of document and the **OLAP Engine**, add the data source and the state. Remember to save to move to the next step: open the Designer. The latter can be opened clicking on the **Open Designer** link.

A new page will be opened, the first thing to choose is the kind of template between XMLA Server and Mondrian, we choose Mondrian one. Then you will be asked to choose the Mondrian Schema and after that to select a cube. Next figure sums up these three steps.

Then clicking on start button you will enter a page like that of the following figure.

Once entered the page the user can freely set the fields as axis or as filter cards, according to requirements. Refer to *Functionalities* Chapter to review the terminology. Make your selection and you can already save the template as shown below.

You can notice that the side panel contains some features (see next figure):

- to set the drill on Position, Member or Replace;
- to show the Mdx query;
- to define the cross navigation;

Olap Designer

BACKSTART

Select Type Of Template

Mondrian

Select Mondrian Schema

FoodMart_v2

Select Cube

Sales

Fig. 1.406: OLAP core configuration.

Time

Stores

Customers

Measures

Unit Sales

Product

All Products

510,011

Fig. 1.407: Defining OLAP template.

Stores

Customers

Measures

Unit Sales

Store Sales

Store Cost

Product

All Products

All Periods

1,352,144

2,861,203.24

1,146,815.83

Product

Drink

All Periods

123,669

247,775.62

99,097.70

Product

Food

All Periods

964,740

2,048,024.95

821,127.08

Product

Non-Consumable

All Periods

256,045

546,552.43

219,026.94

Settings

Drill On Dimension

positionmemberreplace

Template Editing

CLOSE DESIGNER

SAVE TEMPLATE

Fig. 1.408: Defining OLAP template.



Fig. 1.409: Side panel features for the OLAP Designer.

-  to configure buttons visibility.

Refer to Section *Functionalities* to recall the action of the different drills, the one selected in the template will be the default used in the OLAP document.

You can define a cross navigation opening the wizard and clicking on the “Add” button at the top right corner.

Note that the parameter name will be used to configure the (external) cross navigation. In fact, to properly set the cross navigation the user must access the “Cross Navigation” functionalities available in Knowage Server. Here, referring to *Cross Navigation* section of *Analytical document* chapter, you will use the parameter just set as output parameter.

Cross Navigation Definition

ADD NEW

name ↑↓

type ↑↓

No data found

Select The Type Of Cross Navigation Below

From Cell

From Member

CANCEL

NEXT

Cross Navigation Definition

Value

[Promotion Media].[Media Type]

SELECT FROM TABLE

Parameter Name *

Media_type

CANCEL

SAVE

Fig. 1.410: Cross navigation definition.

As shown in figure below, the buttons wizard helps to decide which permissions are granted to the end-user. Some features can only be let visible while others can also be selected by default when a user open the document.

Once the configuration is done click on the **Save template** button and on the **Close designer** button to exit template available at the bottom of the side panel.

name	<input type="checkbox"/> Visible	<input type="checkbox"/> Clicked
Drill Through	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Mdx Query	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Show parent members	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Hide spans	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Sorting settings	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Show properties	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Suppress empty rows/columns	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Save customized view	<input type="checkbox"/>	<input type="checkbox"/>
Reload schema	<input type="checkbox"/>	<input type="checkbox"/>

CANCEL **SAVE**

Fig. 1.411: Buttons wizard.

1.10.1.3.3.1 Profiled access

As for any other analytical document, Knowage provides filtered access to data via its behavioural model. The behavioural model is a very important concept in Knowage. For a full understanding of its meaning and functionalities, please refer to *Data security and access management* section.

Knowage offers the possibility to regulate data visibility based on user profiles. Data visibility can be profiled at the level of the OLAP cube, namely the cube itself is filtered and all queries over that cube share the same data visibility criteria.

To set the filter, which is based on the attribute (or attributes) in the user's profile, the technical user has to type the Mondrian schema. We report Cube level profilation example as a reference guide. Note that data profiling is performed on the cube directly since the filter acts on the data retrieval logics of the Mondrian Server. So the user can only see the data that have been got back by the server according to the filter.

Listing 1.35: Cube level profilation example.

```

1  <?xml version="1.0"?>
2  <Schema name="FoodMartProfiled">
3    ...
4    <Cube name="Sales_profiled"> <Table name="sales_fact_1998"/>
5    ...
6    <!-- profiled dimension -->
7    <Dimension name="Product" foreignKey="product_id">
8      <Hierarchy hasAll="true" allMemberName="All Products" primaryKey="product_id">
9        <View alias="Product">
10          <SQL dialect="generic">
11            SELECT pc.product_family as product_family, p.product_id as
12              product_id,
13              p.product_name as product_name,
14              p.brand_name as brand_name, pc.product_subcategory as
15              product_subcategory, pc.product_category as product_category,
16              pc.product_department as product_department
17            FROM product as p
18            JOIN product_class as pc ON p.product_class_id = pc.
19              product_class_id
20            WHERE and pc.product_family = '${family}'

```

(continues on next page)

(continued from previous page)

```

21      </SQL>
22    </View>
23
24    <Level name="Product Family" column="product_family"
25      uniqueMembers="false" />
26    <Level name="Product Department" column="product_department"
27      uniqueMembers="false"/>
28    <Level name="Product Category" column="product_category"
29      uniqueMembers=" false"/>
30    <Level name="Product Subcategory" column="product_subcategory"
31      uniqueMembers="false"/>
32    <Level name="Brand Name" column="brand_name"
33      uniqueMembers="false"/>
34    <Level name="Product Name" column="product_name"
35      uniqueMembers="true"/>
36  </Hierarchy>
37 </Dimension>
38 </Cube>
39 ...
40 </Schema>

```

In the above example, the filter is implemented within the SQL query that defines the dimension using the usual syntax **pr.product_family = '\${family}'**.

The value of the “family” user profile attribute will replace the `${family}` placeholder in the dimension definition.

You can filter more than one dimensions/cubes and use more profile attributes. The engine substitutes into the query the exact value of the attribute; in case of a multi value attribute to insert in an SQL-IN clause you will have to give the attribute a value like 'value1', 'value2' and insert into the query a condition like **and pc.product_family IN (\${family})**.

Once the OLAP document has been created using the template designer the user can insert parameters to profile the document. To set parameters the user has to download the Mondrian schema and edit it; modify the dimension(s) (that will update according to the value parameter(s)) inserting an SQL query which presents the parametric filtering clause.

Hint: Filter through the interface

Note that for the OLAP instance, it has not proper sense to talk about “general” parameters. In this case we only deal with profile attributes while all the filtering issue is performed through the interface, using the filter panel.

1.10.1.3.4 Cross Navigation

The cross navigation must be implemented at template level but also at analytical document level. The latter has been already wildly described in *Cross Navigation* section. In the following we will see the first case. Observe that both procedures are mandatory.

For OLAP documents it is possible to enable the cross navigation on members or on cells and we will give more details on these two cases in the following.

Generally speaking, the user must modify the template file using the designer to configure the cross navigation in order to declare the output parameters of the document. We remember that the output parameters definition is discussed in *Cross Navigation* section of *Analytical document* chapter of this manual.

1.10.1.3.4.1 Cross navigation on members

To activate the cross navigation on a member means that the user can click on a member of a dimension to send its value and visualize a target document. The first type of navigation can be set by directly editing the OLAP query template or by using the Knowage designer, as described in previous *OLAP designer* section. In the first case you need to add a section called “clickable” inside the MDX query tag. In particular:

- the attribute value is equal to the hierarchy level containing the member(s) that shall be clickable;
- the element represents the parameter that will be passed to the destination document. The name attribute is the URL of the parameter that will be passed to the target document. The value 0 represents the currently selected member, as a convention: this value will be assigned to the parameter whose URL is null.

Figure below gives an example. Note that you can recognize that the cross navigation is activated when elements are shown blue highlighted and underlined.

Stores		Measures							
Product		All Stores		Canada		Mexico		USA	
		Store Cost	Store Sales	Store Cost	Store Sales	Store Cost	Store Sales	Store Cost	Store Sales
	All Products	1,146,815.83	2,861,203.24	262,035.31	653,743.83	249,894.43	622,741.09	448,784.69	1,120,157.76
	Car								
	Drink	99,097.70	247,775.62	23,285.81	58,164.15	20,884.90	52,240.46	39,149.61	97,693.12
	Food	821,127.08	2,048,024.95	187,082.07	466,686.51	180,216.02	448,659.30	320,872.14	800,726.64
	Non-Consumable	219,026.94	546,552.43	50,106.33	124,988.02	47,129.06	117,720.43	85,616.53	213,820.89

Fig. 1.412: Cross navigation on member.

If you open the template file you will read instructions similar to the ones reported in Syntax used to set cross navigation.

Listing 1.36: Syntax used to set cross navigation.

```

1 <MDXquery>
2   select {[Measures].[Unit Sales]} ON COLUMNS,
3   {[([Region].[All Regions], [Product].[All Products])}] ON ROWS from
4   [Sales_V]
5   <clickable name="family" type="From Member" uniqueName="[Product].[Product Family]" >
6     <clickParameter name="family" value="{0}"/>
7   </clickable>
8 </MDXquery>

```

1.10.1.3.4.2 Cross navigation from a cell of the pivot table

This case is similar to the cross navigation on members except that in this case values of all dimensions can be passed to the target document. In other words, the whole dimensional context of a cell can be passed. Now let us suppose the user wishes to click on a cell and pass to the target document the value of the level family of product dimension and year of time dimension. It should creates two parameters: one for family where dimension is product, hierarchy is product, level is product family and one for year parameter where dimension in type, hierarchy is time and level is year. Let see what happens when user clicks on a cell. Depending on the selected cell, the analytical driver family of the target document will have a different value: it will be the name of the context member (of the selected cell) of the “Product” dimension, i.e. the [Product] hierarchy, at [Product].[ProductFamily] level. Look at the following Table for some examples:

Table 1.11: Context member on product dimension

Context member on Product dimension	"Family" analytical driver value
[Product].[All Products]	[no value: it will be prompted to the user]
[Product].[All Products].[Food]	Food
[Product].[All Products].[Drink]	Drink
[Product].[All Products].[Non-Consumable]	Non-Consumable
[Product].[All Products].[Food].[Snacks]	Food
[Product].[All Products].[Food].[Snacks].[Candy]	Food

Let us have a look at the template. Syntax used to set cross navigation shows how to use the cross navigation tag:

Listing 1.37: Syntax used to set cross navigation.

```

1  <CROSS_NAVIGATION>
2  <PARAMETERS>
3  <PARAMETER name="family" dimension="Product" hierarchy="[Product]" level="[Product].[Product_
↔Family]" />
4  <PARAMETER name="year" dimension="Time" hierarchy="[Time]" level="[Time].[Year]" />
5  </PARAMETERS>
6  </CROSS_NAVIGATION>

```

In order to activate cross navigation on cells the user must click on the corresponding button in the side bar, then a green arrow will be displayed in each cells to show that cross navigation is enabled. User can click on that icon to start cross navigation from a cell.

Measures	Time	Unit Sales	Store Sales	Store Cost	Distinct Products	Distinct Customers	Atomic
		⊕All Periods	⊕All Periods	⊕All Periods	⊕All Periods	⊕All Periods	⊕All Periods
⊖All Products		↗ 1,352,144	↗ 2,861,203.24	↗ 1,146,815.83	↗ 1,559	↗ 7,824	↗ 402,476
⊖Car		↗	↗	↗	↗	↗	↗
⊖Drink		↗ 123,669	↗ 247,775.62	↗ 99,097.70	↗ 144	↗ 5,101	↗ 36,666
⊖Food		↗ 964,740	↗ 2,048,024.95	↗ 821,127.08	↗ 1,112	↗ 7,730	↗ 287,440
⊖Non-Consumable		↗ 256,045	↗ 546,552.43	↗ 219,026.94	↗ 294	↗ 6,492	↗ 76,089

Fig. 1.413: Cross navigation on cells.

1.10.2 Create What-If

The **What-if** analysis is the capability to make hypothesis and see how these impacts on the business. In practise user can perform What-if analysis using an extension of the OLAP tool. The process of What-if is composed in three phases:

- data preparation,
- workflow definition,
- hypothesis definition.

We start then focusing on this last phase.

1.10.2.1 Interface details

The workflow has an impact on data visualization. A user can understand the state of the workflow looking at the What-if section of the sidebar. There are three possibilities described in the following.

- The user can perform the What-if analysis: in this case the What-If section contains the buttons to edit values; see figure below to check those buttons.

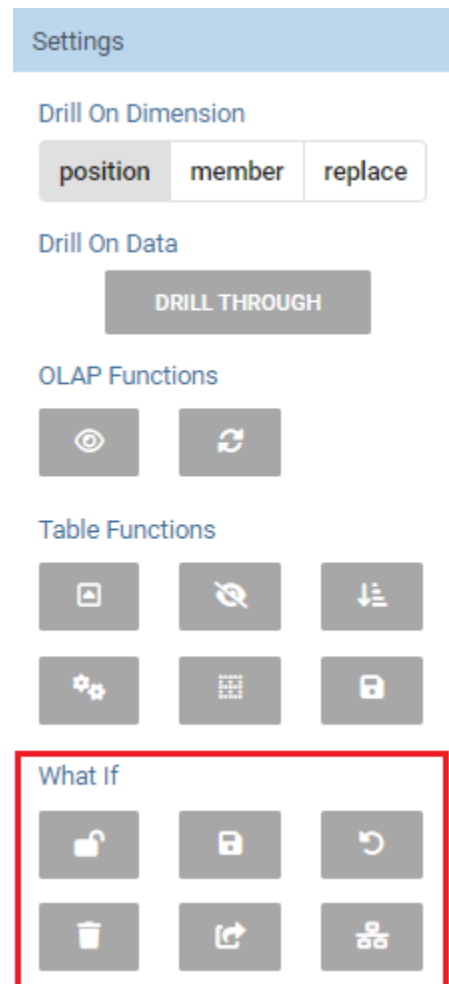


Fig. 1.414: What-If buttons inside the sidebar.

- The schema is locked by another user. In this case a message appears with the name of the active user in the workflow as shown below.
- The workflow is finished.

We briefly recall the functionality of the main buttons:

- **Unlock schema**: it changes the state of the workflow in order to move control to next user.
- **Save as new version**: it persists modification in the database in a new version.
- **Undo**: it undoes last modification.
- **Delete versions**: it opens a wizard user can use to delete versions.
- **Output wizard**: it allows user to export the edit cube in two different formats, table and csv in the specific.



Fig. 1.415: The What-If is used by another user.



Fig. 1.416: The What-If is finished.

- Select an algorithm.

1.10.2.2 Meta-language description

We saw that the What-If engine allows the final user to change data values. Here we see how it is possible to modify a cell value through a formula, unconditionally from the aggregation level of the cell. The formula must be written using a particular language called **meta-language** that is described below. Firstly the available arithmetic symbols are: + - * / () %.

The computation $100 + 10\%$ is a simple example of usage of the operation %. Note that the formula can start with “=”, but this is not mandatory.

To activate the editing of a measure cell that is not shown in the OLAP you must first click on the filter icon of the measure filter card and check the target measure. Then select the version you want to use and change values of figure below shows where are available these objects in the interface.

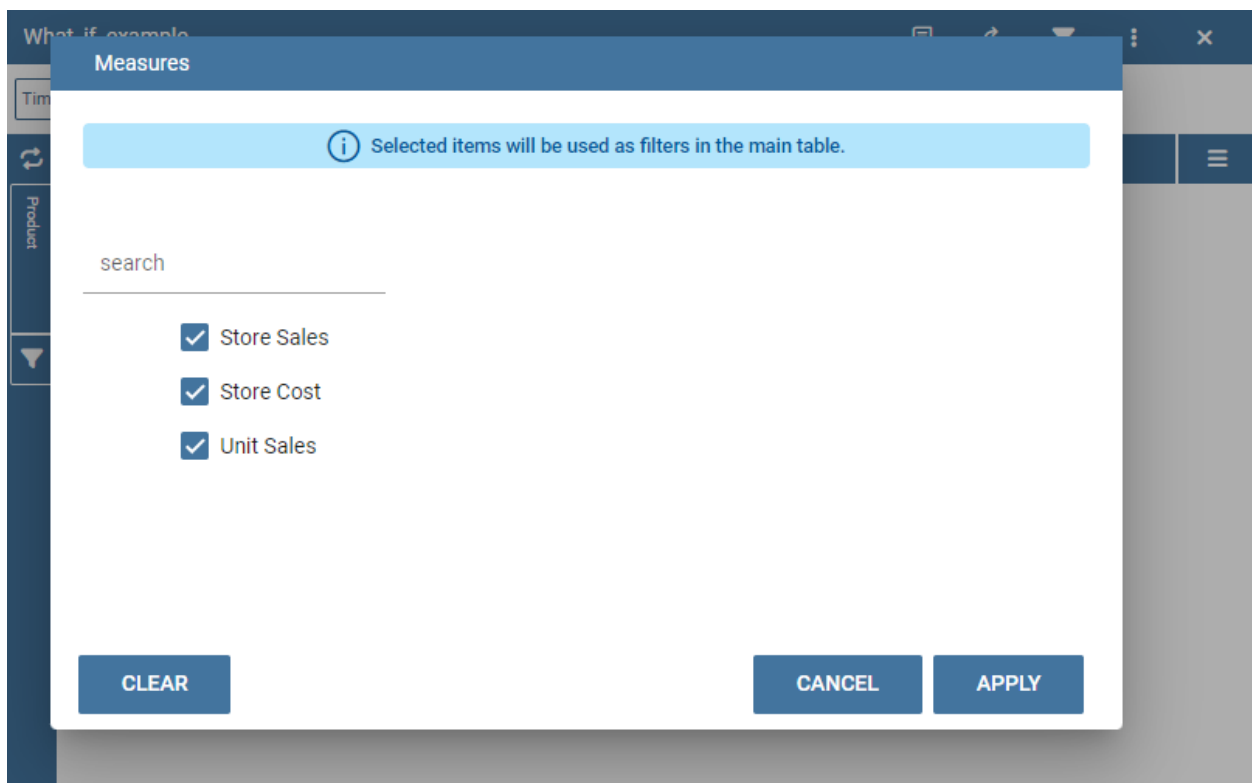


Fig. 1.417: Checking measures.

Then double-click on the target measure cell and a box will appear allowing you to insert a formula. Type the computation syntax and click on the submit button on the keyboard to convalidate it or on the close button to cancel it, as shown below.

We stress that you can also refer to members that are not included in the tuple represented by the cell that is going to be modified. Let's see some examples. For example suppose the cell refers to the following tuple reported in Code below:

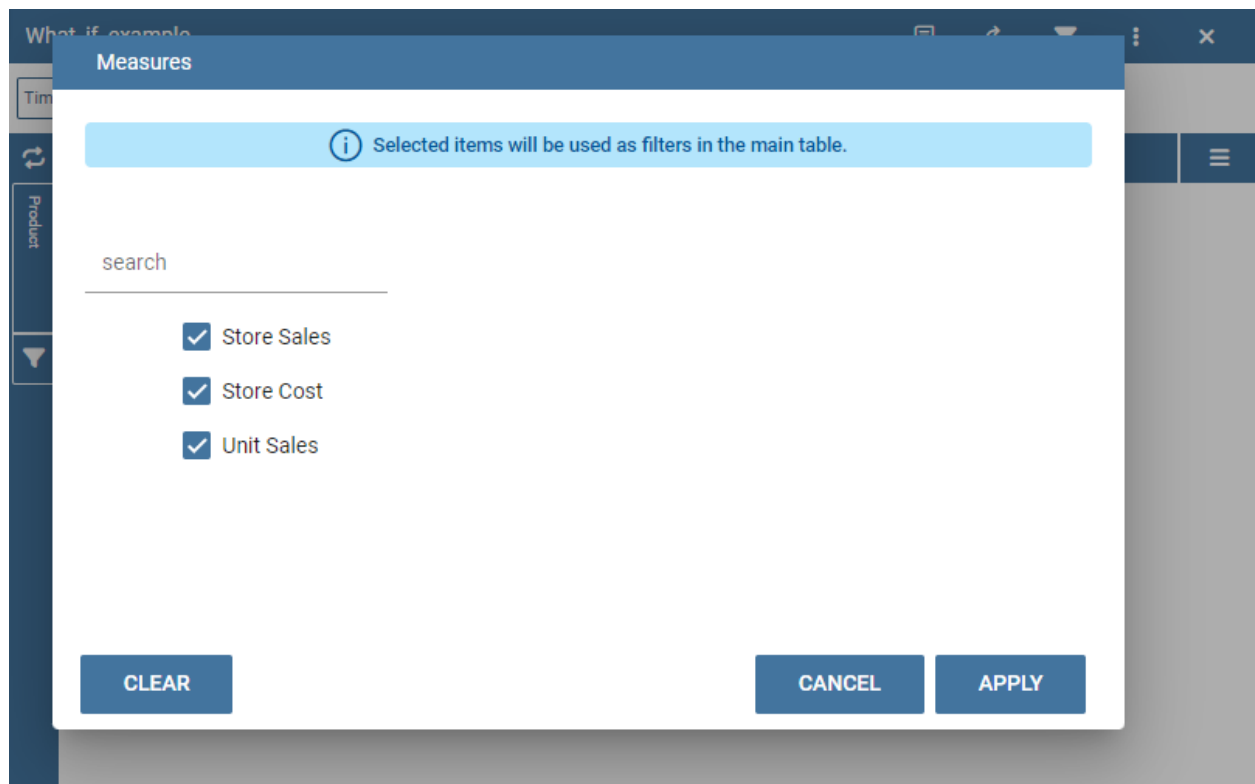


Fig. 1.418: Select the version.

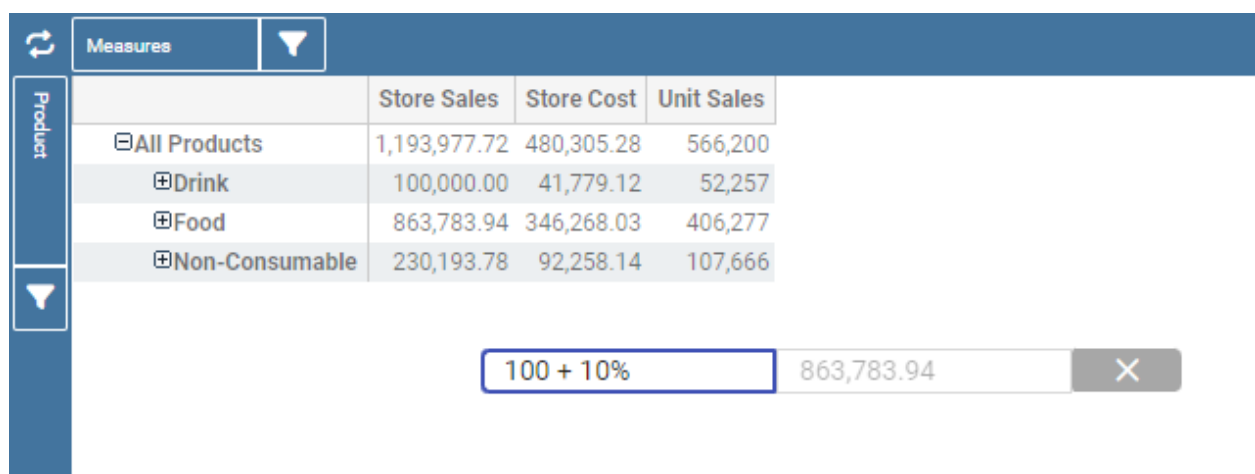


Fig. 1.419: Inserting formula.

Listing 1.38: Product.Deli

```
1 [Measures].[Store Sales], [Product].[Food].[Deli], [Version].[0],
2 [Region].[Mexico Central], [Customers].[All Customers], [Customers].[All Customers]
```

You can refer to the tuple in the next code with just Product.Eggs and at the same time to the tuple in the second code below with just Product.Eggs; Measures.Unit Sales

Listing 1.39: Product.Eggs

```
1 [Measures].[Store Sales], [Product].[Food].[Eggs], [Version].[0],
2 [Region].[Mexico Central], [Customers].[All Customers], [Customers].[All Customers]
```

Listing 1.40: Product.Eggs; Measures.Unit Sales

```
1 [Measures].[Unit Sales], [Product].[Food].[Eggs], [Version].[0],
2 [Region].[Mexico Central], [Customers].[All Customers], [Customers].[All Customers]
```

Note that if you create a formula on a cell and you want to move it along a dimension (for example the cell refers to member Time.2016 and you want to get value for Time.2017) you have to refer to a member of same level. So for example you can get value of the cell for Time.2017, but not for Time.2017.May.

The syntax is as the one shown in *Referring to different members* or, in case you are using another hierarchy, as in the second code below where you can concatenate different members with “;”.

Listing 1.41: Referring to different members.

```
1 <dimension's name>.<member's name>or[<dimension's name>].<member's name>]
```

Listing 1.42: Referring to different members of another hierarchy.

```
1 <dimension's name>.<hierarchy's name>.<member's name>or[<dimension's name>].< hierarchy's name>].<member
  ↳'s name>]
```

You can also refer to members that are on the same level but they are not sibling members: suppose that, for example, the cell's tuple is as in Code below:

Listing 1.43: Example of cell's tuple.

```
1 [Measures].[Store Sales], [Product].[Food].[Deli], [Version].[0],
2 [Region].[Mexico Central], [Customers].[All Customers], [Customers].[All Customers]
```

Note that you can refer to the tuple

Listing 1.44: Example of cell's tuple.

```
1 [Measures].[Store Sales], [Product].[Drink].[Alcoholic Beverages],
2 [Version].[0], [Region].[Mexico Central], [Customers].[All Customers],
3 [Customers].[All Customers]
```

just with:

Listing 1.45: Shorten syntax code.

```
1 [Product].[Drink.Alcoholic Beverages]
```

Another example from Code below

Listing 1.46: Example of cell's tuple.

```
1 [Measures].[Store Sales], [Product].[Food].[Deli].[Meat],
2 [Version].[0], [Region].[Mexico Central], [Customers].[All Customers],
```

to Code below

Listing 1.47: Example of cell's tuple.

```
1 [Measures].[Store Sales], [Product].[Drink].[Alcoholic Beverages].[Beer and Wine], [Version].[0],
2 [Region].[Mexico Central], [Customers].[AllCustomers], [Customers].[All Customers]
```

is as in the following code

Listing 1.48: Used expression.

```
1 [Product].[Drink.Alcoholic Beverages.Beer and Wine]
```

Note that the last part of the expression is the portion of the path to the target member that differs from the path of the cell's member. Some other examples:

Listing 1.49: Further example.

```
1 [Product].[Food]
```

1.10.2.3 What-if analysis implementation

In this chapter we will deal with some technical features of the What-If analysis that can be handled only by expert users.

1.10.2.3.1 Workflow description

When you perform a what-if analysis the schema is shared in order to be used as a data source. Therefore each time a document linked to a schema can be edited only by one user per time. This behaviour is managed by the Workflow of the schema. The administrator can configure a workflow opening the details of the model in Mondrian schemas catalogue, selecting the schema and going on the workflow tab available on the top of the right sided area.

Referring to the next figure, the interface for the definition of the workflow is composed of a double list where

- the **available users** area contains all the users,
- the **workflow** area contains the sequence of users for the workflow.

When an administrator clicks on the user in the list “available users” the user will be added in the workflow.

Administrator can move the users in the sequence using the arrows or remove them clicking again on them. When the workflow is defined, the administrator can start it clicking on the button start. To start a workflow means to enable the first user of the sequence to apply the what-if on that schema. When a workflow is started it can not be edited by anyone else and an icon appears in the row of actual active user so that the administrators can monitor the state of the schema.

1.10.2.3.2 Schema definition

As we foresaid, the What-If analysis requires some changes in the database. The first step is to create a new table in the database to store the named version of the modified data. The user will then change the values of the cube; it is

The screenshot shows the 'Mondrian Schemas Catalogue' interface. On the left, a sidebar lists schemas: 'Foodmart_sales', 'FoodMart_10_dimension', 'foodmart_whatif' (selected), and 'FoodMart_v2'. The main panel is titled 'Foodmart_whatif' and has two tabs: 'Detail' and 'Work Flow' (highlighted with a red box). Under the 'Work Flow' tab, there are fields for 'Name *' (foodmart_whatif) and 'Description' (foodmart_whatif). Below these is a file selection area with a button 'Scegli file', the text 'Nessun file selezionato', and an upload icon. At the bottom, a 'Saved Versions' section contains a table with one entry.

Active	File Name	Creator	Creation Date
<input checked="" type="radio"/>	whatif-sales-segment.xml	kte ADMIN	4/23/18, 2:23 PM

Fig. 1.420: Workflow tab.

The screenshot shows the 'Users Work Flow' interface. It is divided into two main sections. The left section, titled 'Available Users', contains a search bar and a list of users: 'USER_1' (Mario Rossi), 'USER_2' (USER_2), 'USER_3' (USER_3), and 'USER_4' (USER_4). The right section, titled 'Users Work Flow', contains a search bar and a list of users in the workflow: 'kte_admin' (kte ADMIN), which is marked with a green checkmark.

Fig. 1.421: Workflow tab interface.

then mandatory to create a new table with a structure similar to the analysed cube and a new table (wbversion) that will contain the versioning of the definitions set in the analysis.

Therefore the structure of the new fact table should contain:

- all the foreign keys to the dimensions (all the ones visible in the cube),
- all the editable measures,
- a new numeric column that is a foreign key referencing the version table.

In Figure below there is an example where the cube is sales_fact_1998 and the new table is sales_fact_1998_virtual.

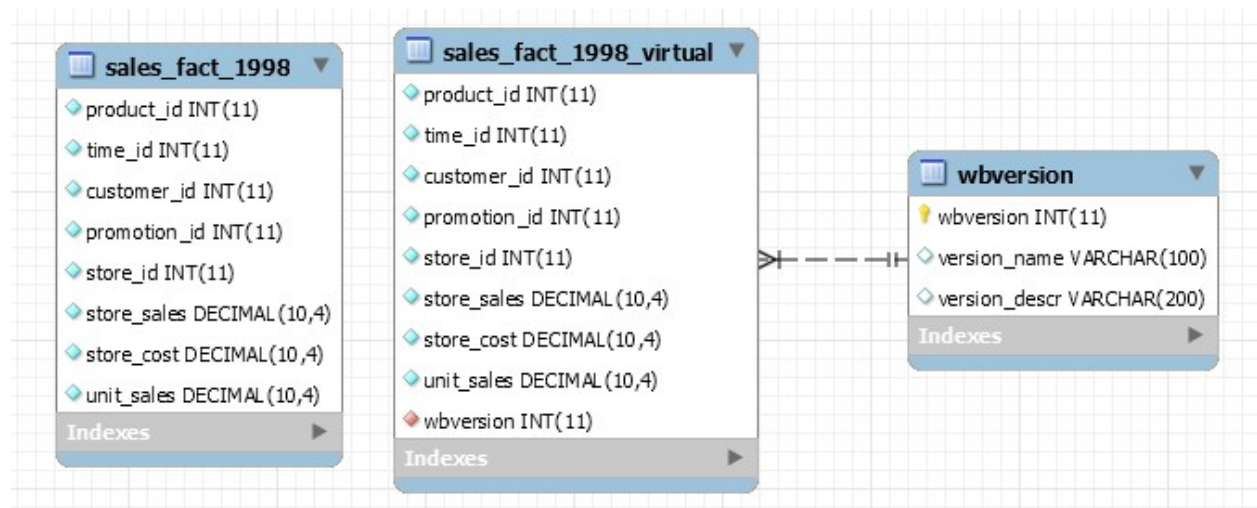


Fig. 1.422: Cube and new virtual table example.

The sales_fact_1998_virtual table should be initialized with the same data contained in sales_fact_1998 plus 0 as version; the wbversion table should be initialized with one record with wbversion = 0 and a name plus a description for the “original values”.

1.10.2.3.3 Changes in the mondrian schema

Now you should map the new tables in the mondrian schema. In order to merge the fact table and the table with the editable measure we create a virtual cube. A virtual cube is a special cube where the values are the result of the join of other cubes. In our case the join keys are the dimensions. The actions to be performed in the mondrian schema are listed right below.

- To create a new “Version” dimension as in *Changing the Mondrian Schema*.

Listing 1.50: Changing the Mondrian Schema.

```

1 <Dimension name="Version">
2   <Hierarchy hasAll="false" primaryKey="wbversion"
3     defaultMember="[Version ].[0]" >
4     <Table name="wbversion"/>
5     <Level name="Version" column="wbversion" uniqueMembers="true"
6       captionColumn="version_name"/>
7   </Hierarchy>
8 </Dimension>

```

- To create the mapping of the editable cube (in our example the table sales_fact_1998_virtual) as shown in *Code Creating the mapping of the editable cube*.

Listing 1.51: Creating the mapping of the editable cube.

```

1 <Cube name="Sales_Edit">
2   <Table name="sales_fact_1998_virtual"/>
3   <DimensionUsage name="Product" source="Product"
4     foreignKey="product_id" />
5   <DimensionUsage name="Region" source="Region"
6     foreignKey="store_id"/>
7   <DimensionUsage name="Customers" source="Customers" foreignKey="customer_id"/>
8   <DimensionUsage name="Version" source="Version"
9     foreignKey="wbversion"/>
10  <Measure name="Store Sales" column="store_sales" aggregator="sum"
11    formatString="#,###.00"/>
12 </Cube>

```

The name of the cube ("Sales_Edit") is the value of the edit Cube attribute of the tag scenario in the template. Note that the name of the dimension Version must be exactly "Version"!!

- To create the virtual cube that will contain the mapping of the columns as in code below.

Listing 1.52: Creating the virtual cube.

```

1 <VirtualCube name="Sales_V">
2   <CubeUsages>
3     <CubeUsage cubeName="Sales_Edit" ignoreUnrelatedDimensions="true"/>
4     <CubeUsage cubeName="Sales" ignoreUnrelatedDimensions="true"/>
5   </CubeUsages>
6
7   <VirtualCubeDimension cubeName="Sales" name="Customers"/>
8   <VirtualCubeDimension cubeName="Sales" name="Product"/>
9   <VirtualCubeDimension cubeName="Sales" name="Region"/>
10  <VirtualCubeDimension cubeName="Sales_Edit" name="Customers"/>
11  <VirtualCubeDimension cubeName="Sales_Edit" name="Product"/>
12  <VirtualCubeDimension cubeName="Sales_Edit" name="Region"/>
13  <VirtualCubeDimension cubeName="Sales_Edit" name="Version"/>
14  <VirtualCubeMeasure cubeName="Sales" name="[Measures].[Unit Sales Original]" visible="false"/>
15  <VirtualCubeMeasure cubeName="Sales" name="[Measures].[Sales Count Original]" visible="false"/>
16  <VirtualCubeMeasure cubeName="Sales_Edit" name="[Measures].[Store Sales]" visible="true"/>
17  <VirtualCubeMeasure cubeName="Sales_Edit" name="[Measures].[Store Cost]" visible="true"/>
18
19  <CalculatedMember name="Sales Count" dimension="Measures">
20    <Formula>VALIDMEASURE([Measures].[Sales Count Original])</Formula>
21  </CalculatedMember>
22
23  <CalculatedMember name="Unit Sales" dimension="Measures">
24    <Formula>VALIDMEASURE([Measures].[Unit Sales Original])</Formula>
25  </CalculatedMember>
26 </VirtualCube>

```

Specifically, in the virtual cube you should specify:

- the list of cubes to be joined (CubeUsages);
- the list of the dimensions of the cube (as you can see it contains all the common dimensions, plus the Version that belongs only to the editable cube);
- the list of the measures. You can perceive that there is a calculated member for the measure Sales Count Original (Sales Count Original is the name of a measure in the Sales cube). This is a trick for the not editable measures. This type of measure lives only in the DWH cube and not in the editable cube. This is due to the fact that the engine doesn't know how to give a value for these measures for the different values of the Version dimension (remember that only the editable cube has the Version dimension). The calculated field solve this problem propagating the same version of the not editable (and versionable) measure for all the version.

Now all the MDX queries can be performed in the virtual cube.

1.10.2.3.4 Changes in the designer

The development of a what-if analytical document is the same as described for the OLAP document but herw you have to choos the **What-If Engine** as the engine instead of the **OLAP Engine**.

All the functionalities described in the *OLAP designer* section are also available for the what-if document. Besides these there are others available only for the what-if documents.

First of all, here you have also to configure the scenario. Opening the side bar in thee designer it will be available a button in order to do this, as shown in figure below.

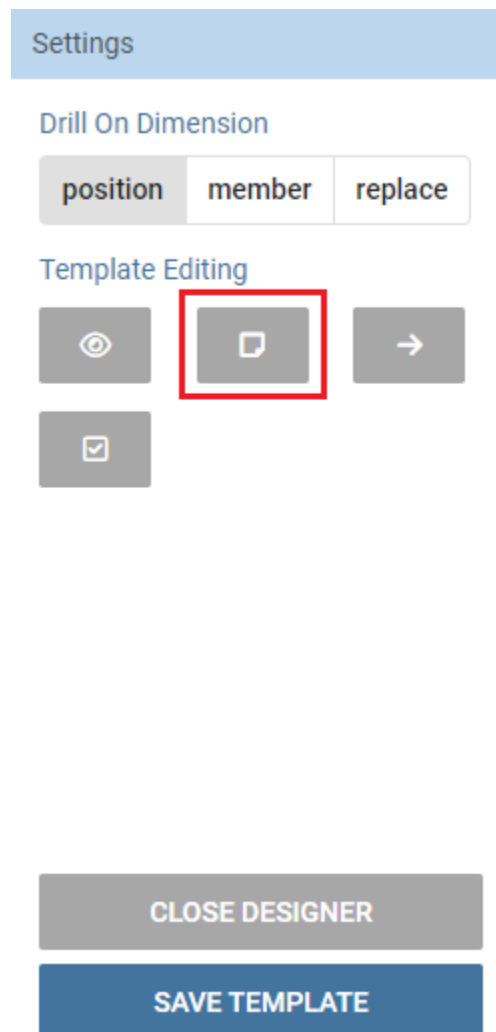


Fig. 1.423: Configure scenario button.

The scenario is used to allow the end-user to edit or not the records contained in the OLAP table. The user is first asked to select the cube in order to get the measures for which the end-user will have the permission to edit and modify. Referring to the following figure, an admin user must simply check the measures using the wizard. At the bottom of the page there is also the possibility to add a parameter that can be used by the end-user when editing the measure, for example if one has a frequent multiplication factor that changes accordingly to the user's needs, the end-user can use that factor to edit measures and ask the admin to update it periodically.

Finally, as far as the buttons wizard the what-if has more available buttons for which the user can only check the

Scenario Wizard ⓘ

Selected Cube
Sales_Edit

Editable Measures

- ☐ name
- ☒ Store Sales
- ☐ Store Cost
- ☐ Unit Sales

Parameters + ⓘ

Parameter name	Parameter value
Click on the + icon to insert data.	

CLEAR DATA CANCEL SAVE

Fig. 1.424: Wizard to configure the scenario.

visibility. Here after the additional available buttons.

1.11 Create a Dossier

A dossier document allows you to obtain a file processed starting from an input template. A typical example of using this feature is the creation of a file with variable content updated at each run. To do this, you need to create a dossier document with a well-configured XML template.

Important: Enterprise Edition

If you purchased Knowage EE, the following features are available only in KnowageER license.

1.11.1 XML Template

1.11.1.1 Tags and properties

XML template is an XML file used to configure parameters needed to the elaboration. This file is uploaded during document creation and can be updated using the HISTORY tab visible in document edit mode.

In particular, tag allowed are:

- **DOSSIER**: main tag used to define the dossier template;
- **PPT_TEMPLATE**: contains properties related to the PPT template (incompatible with DOC_TEMPLATE). You can specify:

Buttons Wizard

name	<input type="checkbox"/> Visible	<input type="checkbox"/> Clicked
Show properties	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Suppress empty rows/columns	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Save new version	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Undo	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Manage versions	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Export output	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Save customized view	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Export excel for edit	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Select algorithm	<input type="checkbox"/>	<input type="checkbox"/>

CANCEL SAVE

Fig. 1.425: Buttons wizard.

- *name*: the name of the template file name (supported file types are PPT, PPTX);
- **DOC_TEMPLATE**: contains properties related to the DOC template (incompatible with PPT_TEMPLATE). You can specify:
 - *name*: the name of the template file name (supported file types are DOCX);
 - *downloadable*: true/false. Enable/disable the download of the template (optional);
 - *uploadable*: true/false. Enable/disable the upload of the template (optional);
- **REPORT**: contains document's properties. You can specify:
 - *label*: the label of the document to be executed and that will substitute the image in the ppt or doc template;
 - *imageName*: the name of the image inside the docx document. If the document is multisheet, imageName value must be composed by the document label ending with the suffix "_sheet_<number>". For example, to use the screenshot of the first sheet, the value will be "documentLabel_sheet_0". This name can be set into title or description using the alternative text menu of the picture;
 - *sheetWidth*: the value of the width of the sheet (in pixels). This value will be used if sheetHeight is also set (optional);
 - *sheetHeight*: the value of the height of the sheet (in pixels). This value will be used if sheetWidth is also set (optional);
 - *deviceScaleFactor*: the value to use as the scaling factor to be applied when capturing the screenshot (optional);
- **REPORTS**: encloses all REPORT tags;
- **PARAMETER**: sets parameter for the document's execution if present. You can specify:
 - *type*: static/dynamic. Defines the way the value is assigned to the parameter. With the dynamic type the user select at each execution the value through the analytical driver associated to the document; with the static type the value is passed once directly in the xml template and the user can't change it. The analytical driver used in the dossier must be the same as the one of the document to be executed;
 - *url_name*: the url name of the analytical driver to be used. It must be the same as the one of the document to be executed;

- *url_name_description*: the description displayed in the driver value input panel for the analytical driver (mandatory if type="static");
- *dossier_url_name*: the url name of the analytical driver set into detail mode of the dossier;
- *value*: the value to be set into the driver (mandatory if type="static");
- **PARAMETERS**: encloses all PARAMETER tags;
- **PLACEHOLDER**: sets the placeholder. You can specify:
 - *value*: the text to be replaced;
- **PLACEHOLDERS**: encloses all PLACEHOLDER tags.

Warning: This feature is compatible with docx created with Microsoft Word 2010 and later.

1.11.2 Image adding (PPT_TEMPLATE)

Suppose you have to create a ppt/pptx file where to place the images relating to one or more reports. You have only to configure XML template defining some placeholders to be use for replacing and execute it. Below is shown an example of an XML template used for this purpose.

Listing 1.53: Example (a) of template for Dossier for Image replacement on docx file.

```

1 <?xml version='1.0' encoding='utf-8'?>
2 <DOSSIER>
3   <PPT_TEMPLATE name="PPT_TEMPLATE.pptx"/>
4   <REPORTS>
5     <REPORT label="Report-multivalue-parameter">
6       <PLACEHOLDERS>
7         <PLACEHOLDER value = "ph1"/>
8       <PLACEHOLDERS>
9       <PARAMETERS>
10        <PARAMETER type="static" dossier_url_name="state" url_name="state"
11        url_name_description="State" value="Canada"/>
12      </PARAMETERS>
13    </REPORT>
14  </REPORTS>
15 </DOSSIER>

```

The example above is using one placeholder and one static analytical driver.

Warning: Please note that the file to be used as a template must be placed in TOMCAT_HOME/resources/<TENANT_NAME>/dossier path.

1.11.3 Image replacing (DOC_TEMPLATE)

Suppose that you have to draw up a document where text is static but images related to need to be updated. With this functionality you will be able to use a docx file as a template and replace images inside it. More precisely, you can configure your XML and docx templates to allow Knowage to replace specific images with new ones obtained by the execution of specified documents.

Below is shown an example of an XML template used for this purpose.

Listing 1.54: Example (a) of template for Dossier for Image replacement on docx file.

```

1  <?xml version='1.0' encoding='utf-8'?>
2  <DOSSIER>
3      <DOC_TEMPLATE name="DOC_TEMPLATE.docx" downloadable="true" uploadable="true" />
4      <REPORTS>
5          <REPORT label="DOC_01" imageName="img_DOC_01" sheetWidth="1366" sheetHeight="650"
↪deviceScaleFactor="1.5">
6              <PARAMETERS>
7                  <PARAMETER type="dynamic" dossier_url_name="family_dossier" url_name="family_
↪document"/>
8                  <PARAMETER type="dynamic" dossier_url_name="category_dossier" url_name="category_
↪document"/>
9              </PARAMETERS>
10             </REPORT>
11             <REPORT label="DOC_02" imageName="img_DOC_02" sheetWidth="1366" sheetHeight="650"
↪deviceScaleFactor="1.5">
12                 <PARAMETERS>
13                     <PARAMETER type="dynamic" dossier_url_name="family_dossier" url_name="family_
↪document"/>
14                     <PARAMETER type="dynamic" dossier_url_name="category_dossier" url_name="category_
↪document"/>
15                 </PARAMETERS>
16                 </REPORT>
17                 <REPORT label="DOC_03" imageName="img_DOC_03" sheetWidth="1366" sheetHeight="650"
↪deviceScaleFactor="1.5" />
18             </REPORTS>
19 </DOSSIER>

```

Note that if the document used to replace an image does not have any parameter, the tag REPORT must be closed inline as you can see for the document with label “DOC_03” in the example above.

Also docx document must be modified to be compatible with the replacer.

In particular: - images in the document must be inserted by copying and pasting from the file system (or using the “insert image” feature); - imageName in the XML template must match the title (alternative text) of the image in the docx; - each image must have a unique name.

Warning: To optimize dossier creation procedure, same document will be executed more than one time if and only if its parameters change. In that scenario sheetHeight, sheetWidth and deviceScaleFactor will be set for every execution. Moreover, if parameters don’t change, document will be executed only one time and sheetHeight, sheetWidth and deviceScaleFactor values will stay the same as the first execution.

1.11.4 My first dossier

You can create a dossier document by using the plus button and choosing “Generic Document”. Proceed by filling in the necessary fields, selecting “Collaboration” as the type and “Dossier engine” as the engine and then choosing the XML template. If the documents to be executed have one or more analytical drivers, these drivers must be added to the dossier document from the DRIVER tab.

After saving the document, you can access the dossier activity page by clicking the play button.

If one or more dynamic analytic drivers are set, the required inputs must be provided in the sliding menu that appears from the right. You will then go to the dossier activity page.

If upload/download are enabled, docx file template can be uploaded/downloaded using the three dot menu on the top right of the “Details” tab.

Document Details

Informations

Upload Template
dossier_template.xml

Label *
DOC-DOSSIER

Name *
DOC-DOSSIER

Description

Preview Image

Type *
Collaboration

Engine *
Dossier Engine

State *
Development

Refresh (Seconds)
0

Select the number of milliseconds after the document will refresh

Visible

Locked

Position

Visibility Restrictions

Defined Restrictions

Select Attribute = Enter Value

Select The Functionalities Where The Document Will Be Visible

- Functionalities
 - Analytical Engines
 - Cockpit
 - Impala
 - Hive
 - SparkSQL
 - MongoDB
 - PostgresSQL

Fig. 1.426: Dossier document creation interface.

DOC-DOSSIER

Details

Activity Name *

LAUNCH ACTIVITY

Launched Activities For This Document

Dossier

No dossier activity has been launched yet. Set a name and click 'Launch activity' to create the first one.

Fig. 1.427: Dossier activity interface.

Warning: This feature is available only for image replacing procedure.

If you want to execute your document, you must enter a name for the activity and click on “LAUNCH ACTIVITY”. A new task will be started in the STARTED state and a new row will be visible in the table below. At the end of the execution of the task, the processed file can be downloaded with the appropriate download icon.

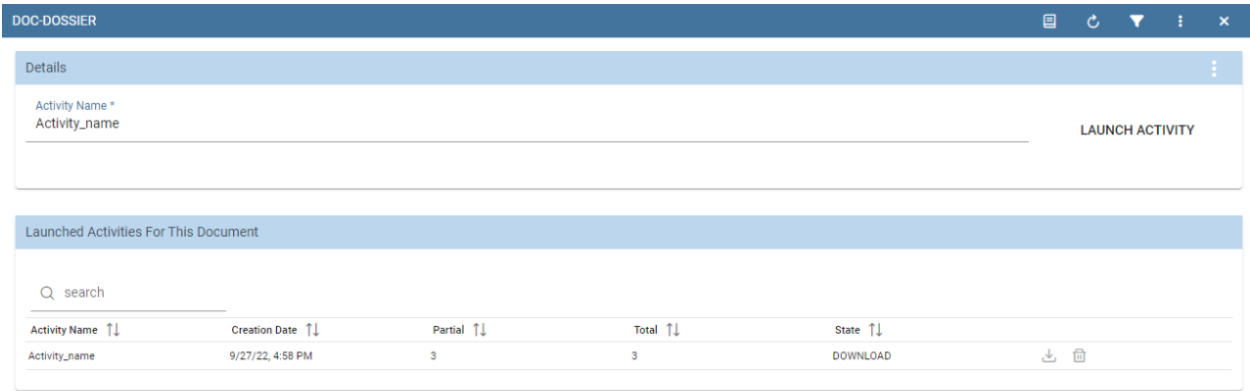


Fig. 1.428: Dossier activity execution finished.

Each line allows you to see useful information on the activity (such as the values of the drivers used for execution) by clicking on the info icon, download the processed file by clicking on the download icon and remove itself by clicking on the trash icon.

1.12 Create Performance Analysis

1.12.1 Create KPI

KPI stands for Key Performance Indicator. It denotes a set of metrics, usually derived from simple measures, which allow managers to take a snapshot on key aspects of their business. The main characteristics of KPIs follow:

- summary indicators,
- complex calculations,
- thresholds supporting results evaluation,
- reference target goals,
- easy to use but requiring more specific skills to design it,
- association with alarms,
- not necessarily used for real-time analysis,
- may refer to a specific time frame.

For these reasons, KPIs are always defined by expert analysts and then used to analyze performances through synthetic views that select and outline only meaningful information.

Knowage allows the configuration of a KPI document thanks to a specific **KPI engine**. The critical value (or values) can be computed and visualized through the functionalities available in the 'KPI model' section of Knowage menu area (see the next figure).

1.12.1.1 KPI development

We introduce the KPI tool by splitting the topic in steps. We briefly sum up here the arguments that we will cover in the following sections in order to have a general overview of the KPI implementation.

- **Measure definition:** it is necessary to define first the measures and attributes, eventually with parameters, to compute the critical value of interest.
- **KPI definition:** here you compute the requested value through a formula using the measures and attributes set in previous step and configure thresholds.
- **Target:** it is possible to monitor the value of one or more KPIs comparing it to an additional fixed value.
- **Scheduling:** you can schedule the execution of one or more KPIs, eventually filtered by conditions.
- **Document creation:** finally, you develop the KPI document.

Therefore, we go into further details.

Measure definition. The first step is to create a new measure or rule. Select **Measure/Rule** from the contextual menu of the main page, as shown below.

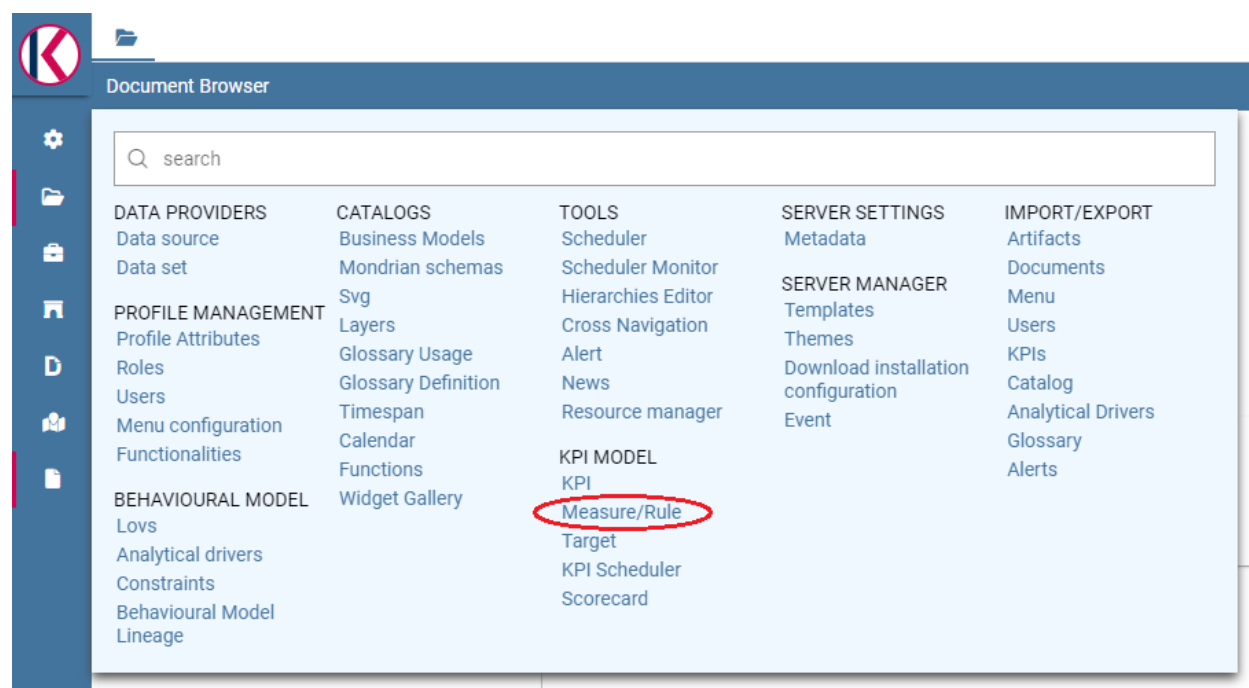


Fig. 1.429: Measure/Rule menu item.

Click on the “Plus” icon to set a new measure/rule. A query editor page is opened. Note that once the data source is selected, pressing simultaneously the CTRL key and the space bar opens a contextual menu containing the available datasource columns and the database keywords. Refer to the following figure to have an example.

Each rule is based on a query to which you can add placeholders. A placeholder represents a filter that you can add to your rule and that can be useful for profiling tasks. It is possible to assign value to a placeholder while configuring the scheduling of the KPI (this procedure will be further examined in “Document creation” paragraph). The syntax to use in queries for setting a placeholder is `columnName=@placeholderName`, as the example in figure below shows.

Generally the rule such a query can return one or more measures and possibly an attribute. An example is given below.

The **Preview** button allows you to check the query execution and have a look on a part of the result set.

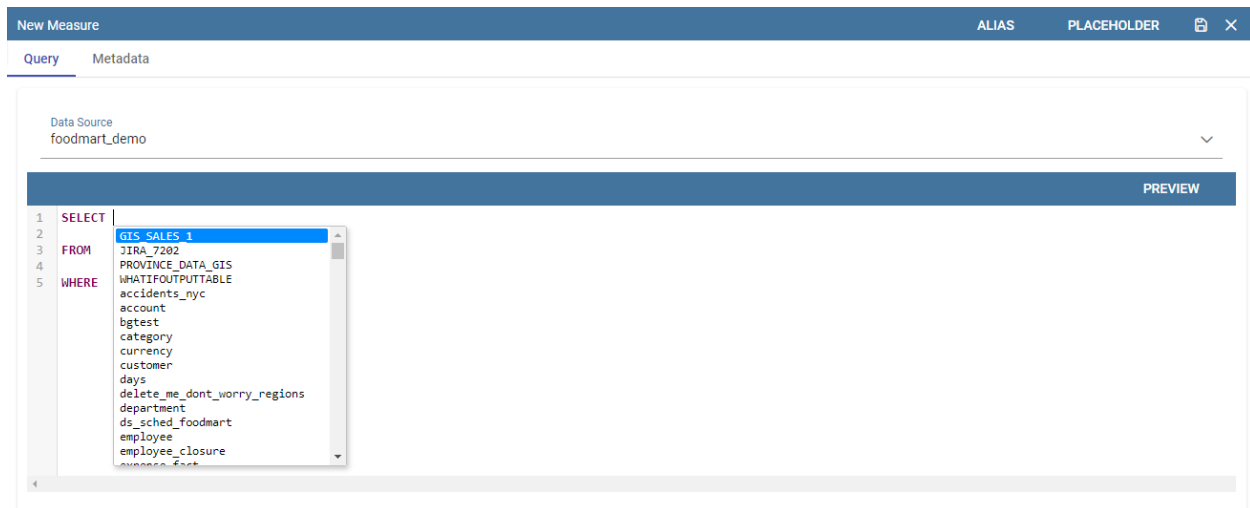


Fig. 1.430: Editing the query when defining a KPI.



Fig. 1.431: Setting placeholder in query definition.



Fig. 1.432: Query definition.

A typology (measure, attribute and temporal attribute) and a category can be assigned to each fields returned by the query using the **Metadata** tab as highlighted in the next figure. The typology is required to associate a type to each field returned by the query. In particular, if the field is a temporal one, it is mandatory to specify to which level you want it to be considered, that is if it corresponds to a day, a month, a year, a century or a millennium. For measures and attribute it is possible to assign also a category to easily look them up in a second moment.

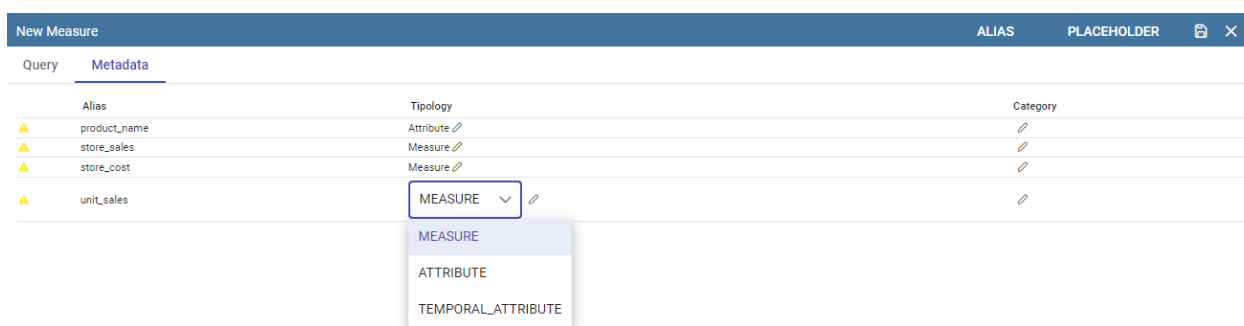


Fig. 1.433: Metadata settings.

We say in advance that, it is important to distinguish these metadata categories from the required field “Category” that occurs while saving the KPI definition (see next figure).

In fact, the category assigned when saving the KPI definition will be added (if it doesn’t exist) in the “KPI categories” list, used to profile KPIs on roles (see Figure below).

Warning: Do not mistake metadata category with the KPI category

The category defined in the metadata tab of the “Measure definition” functionally are not the same categories selected in the tab area of the “Roles management” functionality (see the figure above). The first are used to classify the metadata while the second are needed for the profiling issue.

As we told, a proper categorization exists for the aggregations of type temporal. In fact, when associating “temporal attribute” as metadata typology, the technical user must indicate the temporal level of the data in the category section:

Add KPI Associations

Category *

☐ Enable Versioning

CANCEL

SAVE

Fig. 1.434: Category assigned when saving a KPI definition.

Detail

Authorizations

Business models

Data sets

KPI Categories

KPI Categories

<input type="checkbox"/>	Name
<input type="checkbox"/>	ASDFG
<input checked="" type="checkbox"/>	CUSTOMER
<input type="checkbox"/>	DEMO
<input type="checkbox"/>	EMPLOYEE
<input type="checkbox"/>	FOODMART
<input type="checkbox"/>	FOOD_KPI_2
<input type="checkbox"/>	INVENTORY

Fig. 1.435: KPI category.

day, month, year or others. You can see an example in the following figure. Note that the field set as temporal type must contains numbers (therefore string types are not allowed). For example, if one wants to set a field as “month”, such a field must contain {01,02,03,...,12} that will be considered as {January, February, March,...,December}.

Alias	Tipology	Category
⚠ store_sales	Measure ✎	✎
⚠ store_cost	Measure ✎	✎
⚠ unit_sales	Measure ✎	✎
⚠ month	Temporal attribute ✎	<div> MONTH ✎ <div> MILLENNIUM CENTURY YEAR MONTH DAY </div> </div>

Fig. 1.436: Level for temporal attributes.

Let's now examine extra features available on the right top corner. There you can find the following tab:

- **Alias:** you can see the aliases defined in other rules; note that only the aliases of those columns saved as attribute are stored and showed. This is useful in order to avoid aliases already in use when defining a new rule. Indeed an alias can not be saved twice even if contained in different rules.

The screenshot shows the 'New Measure' interface. The 'Query' tab is active, displaying a SQL query for the 'foodmart_demo' data source. The query selects product name, store sales, store cost, and unit sales from the 'sales_fact' table, joined with the 'product' table. The 'Alias' tab is also visible, showing a list of existing aliases: MONTH, STORE_ID, YEAR, PRODUCT_FAMILY, ANNO, MESE, sales_state_province, store_country, brand_name, and store_city. A search bar is present at the top of the alias list.

Fig. 1.437: Checking aliases.

- **Placeholder:** here you can check the existing placeholders. These are set in the query you're editing or in other ones.
- **Save:** to save the query and other settings just configured.

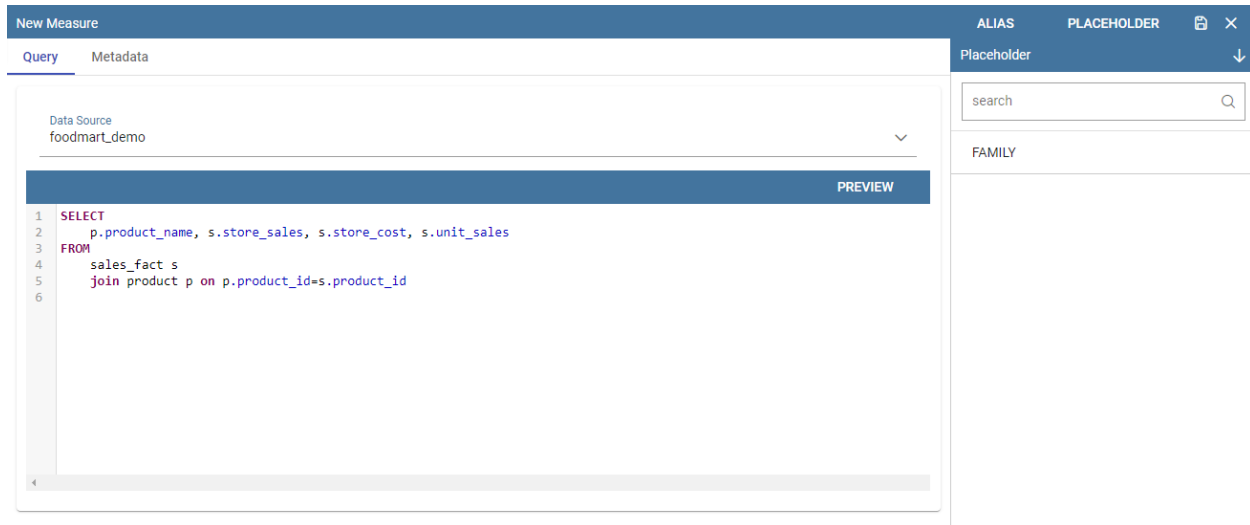


Fig. 1.438: Setting placeholders in a query.

- **Close:** to exit the rule configuration window.

KPI definition. Select the **KPI** item from the contextual menu of the main page of Knowage, as shown in figure below. Click on the “Plus” icon to configure a new KPI.

The window opens a first tag, entitled **Formula** (see figure below), where you must type the formula to enable calculations. Press CTRL key and space bar simultaneously to access all measures defined in the rules, as shown below.

Once a measure is selected, you need to choose which function must act on it. This can be done by clicking on the $f()$ that surrounds the chosen measure. See figure below.

Clicking on the $f()$ the interface opens a pop up where you can select which function apply to the measure, see figure below. Once the selection is made the formula will be autofilled with the proper syntax and you can go on editing it.

Once a complete formula (an example is given in figure below) has been inserted you must set a name for it and then move to the next tab.

The **Cardinality** tab allows you to define the granularity level (namely the grouping level) for the attributes of the defined measures.

Referring to the example below, selecting (with a check) all the measures for the attribute `product_name` the KPI measures are computed for each value of the `product_name`; otherwise no grouping will be done.

Limit values can be set using the Threshold tab (Figure below). It is mandatory to set at least one threshold otherwise the KPI cannot be saved. You can choose a threshold already defined clicking on “Threshold” list or create a new one.

To insert a new threshold it is mandatory to insert a name and assign a type, while the description is optional. Clicking on **Add new threshold item** a new item appears. It is necessary to define the **Label**, **Minimum** and **Maximum** values. It is possible to choose whether to include the minimum and maximum values in the value slot or not. The **Severity** is used to link colors to their meaning and make the thresholds readable by other technical users. Note that the color can be defined through the RGB code, the hexadecimal code or choosing it from the panel.

Warning: “Standard” colors for thresholds

We call standard colors for thresholds the ones listed below (in terms of hexadecimals):

- green: #00FF00,

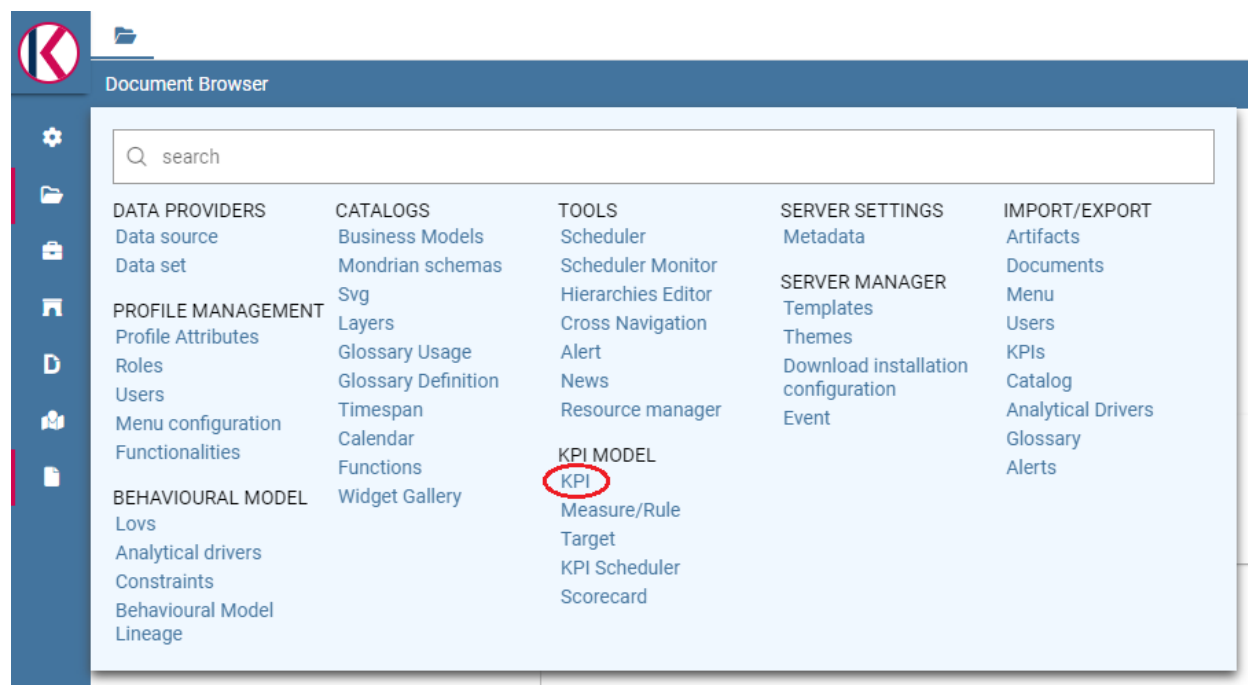


Fig. 1.439: Configure a new KPI.

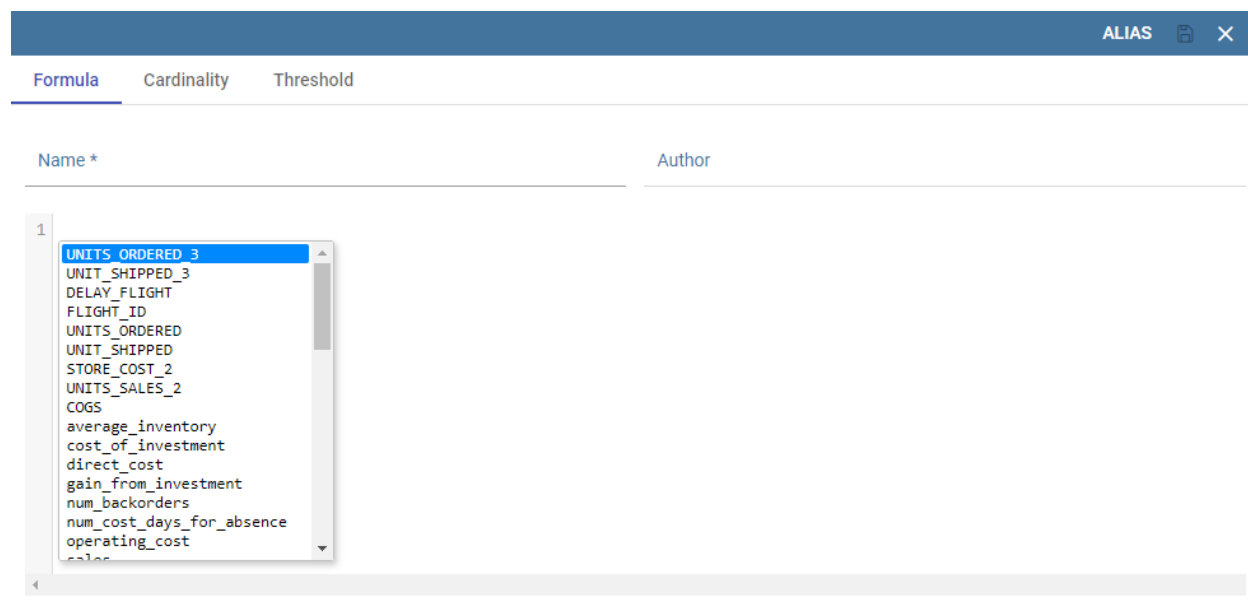


Fig. 1.440: Formula definition tab.

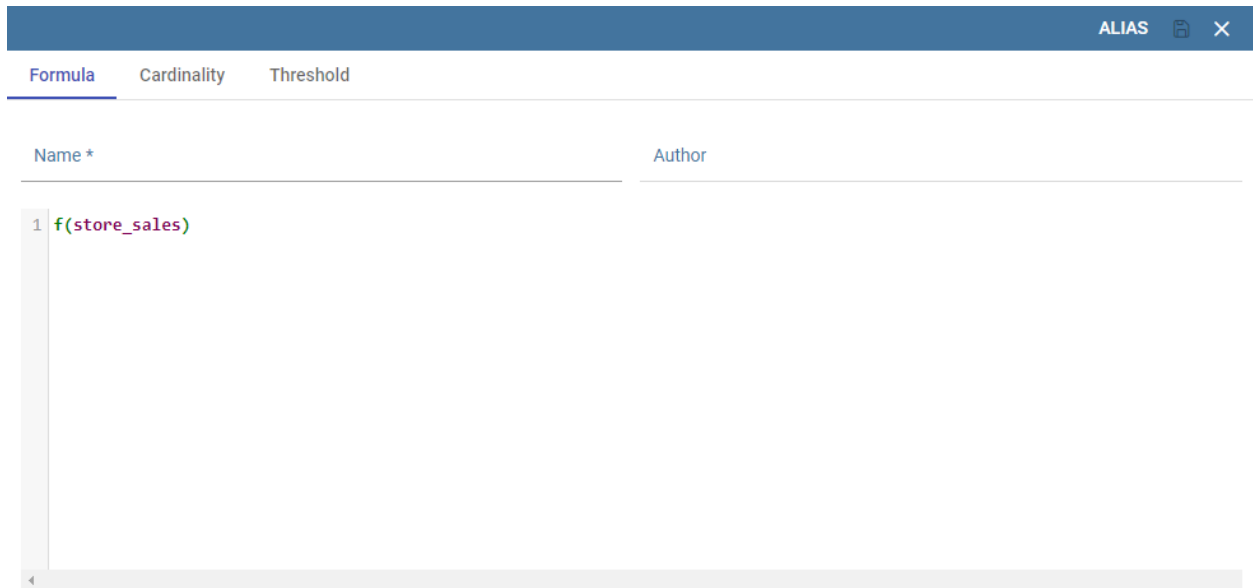


Fig. 1.441: Formula syntax.

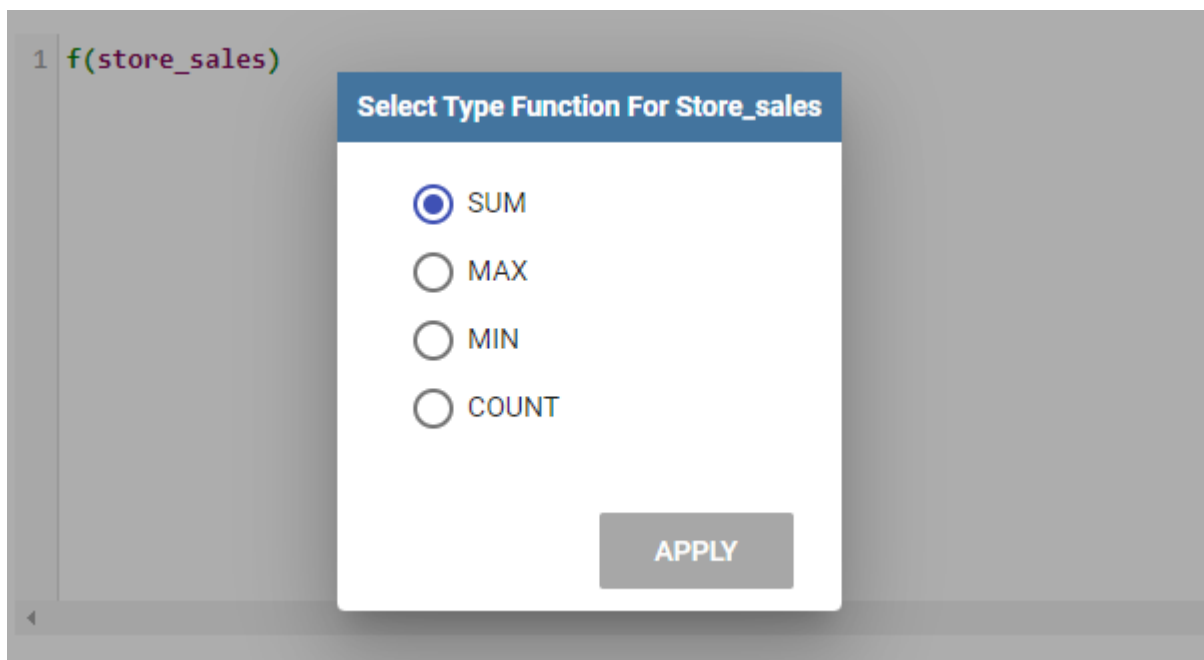


Fig. 1.442: Available functions.

KPI_DEFINITION

ALIAS

×

Formula

Cardinality

Threshold

Name *

KPI_DEFINITION

Author

1

$$\left(\sum(\text{store_sales}) - \sum(\text{cost_store}) \right) / \text{COUNT}(\text{sales_unit})$$

Fig. 1.443: Complete formula example.

KPI_DEFINITION

ALIAS

×

Formula

Cardinality

Threshold

$$\left(\sum(\text{store_sales}) - \sum(\text{cost_store}) \right) / \text{COUNT}(\text{sales_unit})$$

	store_sales	cost_store	sales_unit
product_name	✓	✓	✓

Fig. 1.444: Cardinality settings example.

KPI_DEFINITION
ALIAS

Formula
Cardinality
Threshold

Name *

StoreSalesThreshold

Description

Type

Range

	label	Min	Include Min	Max	Include Max	Severity	Color	
								Thresholds List
	1	0	<input checked="" type="checkbox"/>	50	<input checked="" type="checkbox"/>	LOW	<div style="display: inline-block; width: 15px; height: 15px; background-color: #38c722; border: 1px solid #000;"></div> 38c722	
	2	50	<input type="checkbox"/>	100	<input checked="" type="checkbox"/>	MEDIUM	<div style="display: inline-block; width: 15px; height: 15px; background-color: #ffd900; border: 1px solid #000;"></div> ffd900	
	3	100	<input type="checkbox"/>	150	<input checked="" type="checkbox"/>	HIGH	<div style="display: inline-block; width: 15px; height: 15px; background-color: #ff7700; border: 1px solid #000;"></div> ff7700	
	4	150	<input type="checkbox"/>	200	<input checked="" type="checkbox"/>	URGENT	<div style="display: inline-block; width: 15px; height: 15px; background-color: #c90000; border: 1px solid #000;"></div> c90000	

[Add new threshold item](#)

Fig. 1.445: Setting thresholds.

- yellow: #FFF000,
- red: #FF0000.

Finally the user must save the KPI definition clicking on the “Save” button, available at the right top corner of the page. Once the user clicks on the “Save” button, the “Add KPI associations” wizard opens, as you can see from next figure. Here, it is mandatory to set the KPI category so that only users whose roles have the permissions to this specific category can access the KPI. Remember that it is possible to assign permissions over KPI when defining roles using the “Roles management” functionality. Furthermore, the user can check or uncheck the **Enable Versioning** button if he/she wishes to keep track of the rules/measures/targets that generate the KPI response at each KPI execution.

Target. This step is not mandatory. Enter the **Target** menu item as shown below.

Clicking on the “Plus” icon you can add a new target.

To define a new target you must insert a name, a validity start date/end date and the association to at least one target. It is possible to associate a target clicking on the item **Add KPI association** and selecting the KPI of interest. Once the association is set, the “Value” box gets editable and you can insert the value you wish to send to the selected KPI. An example is given in figure below.

In the KPI visualization phase, a red bold thick will be displayed on the indicated value (see next figure).

Scheduling. Once the KPI has been defined, it is necessary to schedule it before proceeding with the creation of an analytical document. For this purpose, click on the **KPI Scheduler** from the contextual menu that you can see below.

As for the other interfaces it is enough to click on the “Plus” icon to create a new scheduling. The new scheduling window presents several tabs.

- KPI:** it is possible to associate one or more KPI to the scheduling clicking on “Add KPI Association”.
- Filters:** here you assign values to the filters (if configured) associated to the corresponding rule. Note that it is possible to assign values to the filters through a LOV, a fixed list of values or a temporal function. In case the LOV option is chosen, remember that the LOV must return one unique value. This choice can be useful for profiling tasks.

The screenshot displays the 'KPI_DEFINITION' form with three tabs: 'Formula', 'Cardinality', and 'Threshold'. The 'Formula' tab is active, showing a formula editor with the text: `1 (Σ(store_sales) - Σ(cost_store)) / COUNT(sales_unit)`. The form also includes fields for 'Name *' (KPI_DEFINITION) and 'Author' (kte_admin). An 'Add KPI Associations' dialog box is open in the foreground, featuring a 'Category *' dropdown menu set to 'PRODUCT', an 'Enable Versioning' checkbox, and 'CANCEL' and 'SAVE' buttons.

Fig. 1.446: Save the KPI definition and set category.

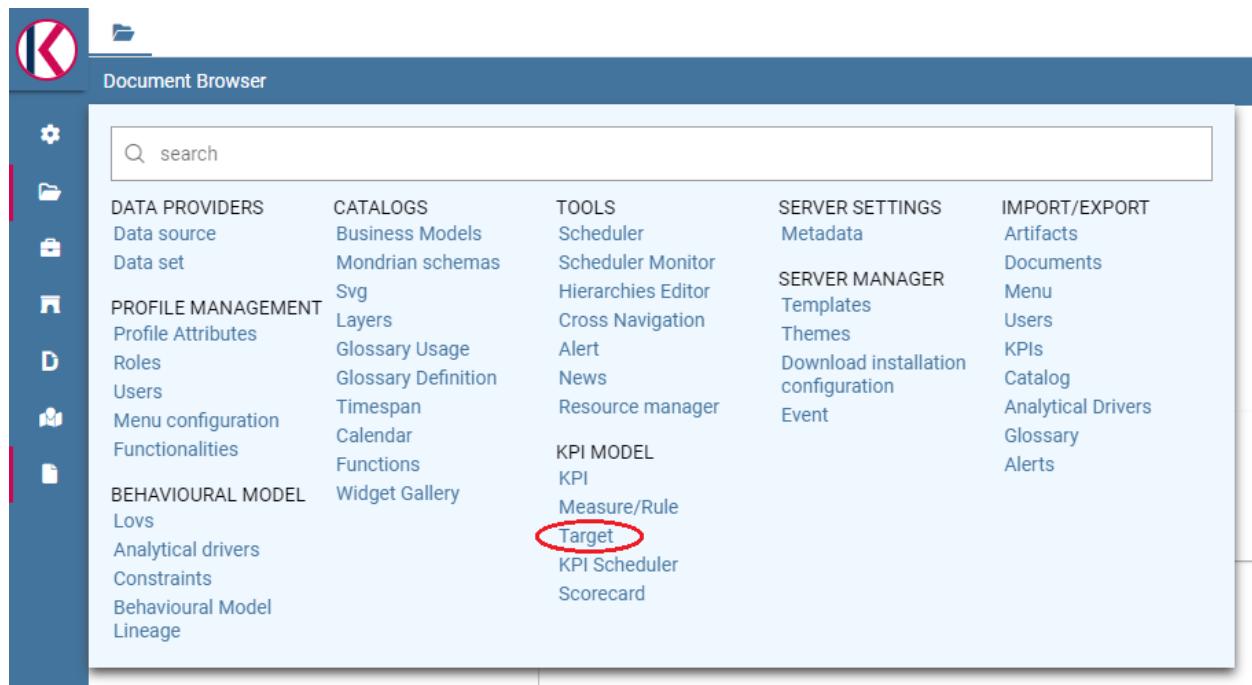


Fig. 1.447: Target Definition menu item.

The screenshot shows a form for defining a KPI target. The form is titled 'KPI target association' and contains the following sections:

- Name ***: TargetIncome
- Category**: (empty dropdown)
- Start Validity Date ***: 9/1/2022
- End Validity Date ***: 11/30/2022
- Apply Target On Kpi**: A table with two columns: 'KPI name' and 'Value'.

KPI name ↑↓	Value ↑↓
KPI_DEFINITION	9.500.000,00

At the bottom of the form is a button labeled 'ADD KPI ASSOCIATION'.

Fig. 1.448: KPI target association.

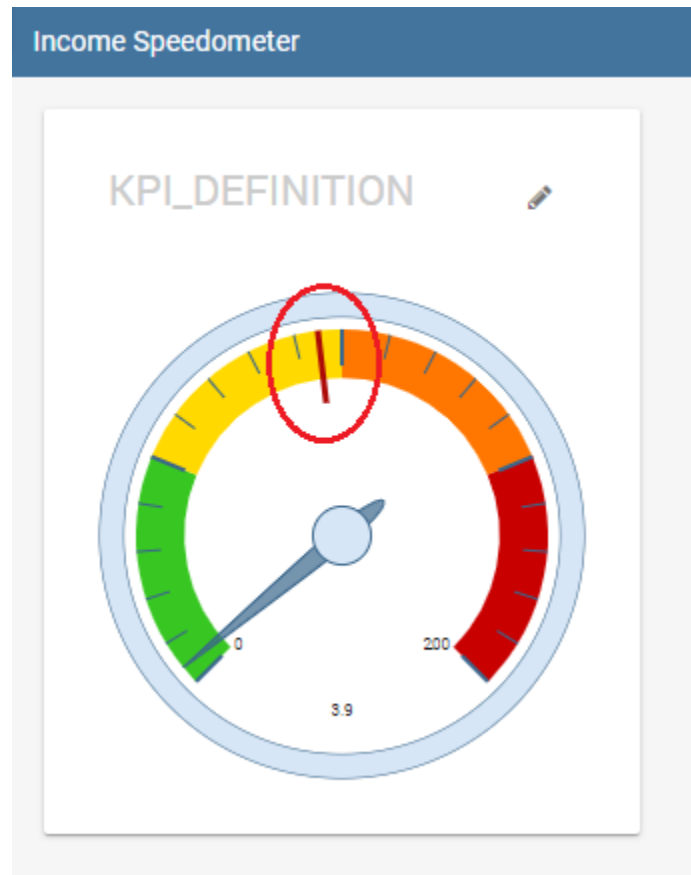


Fig. 1.449: Target mark in KPI scale of values.

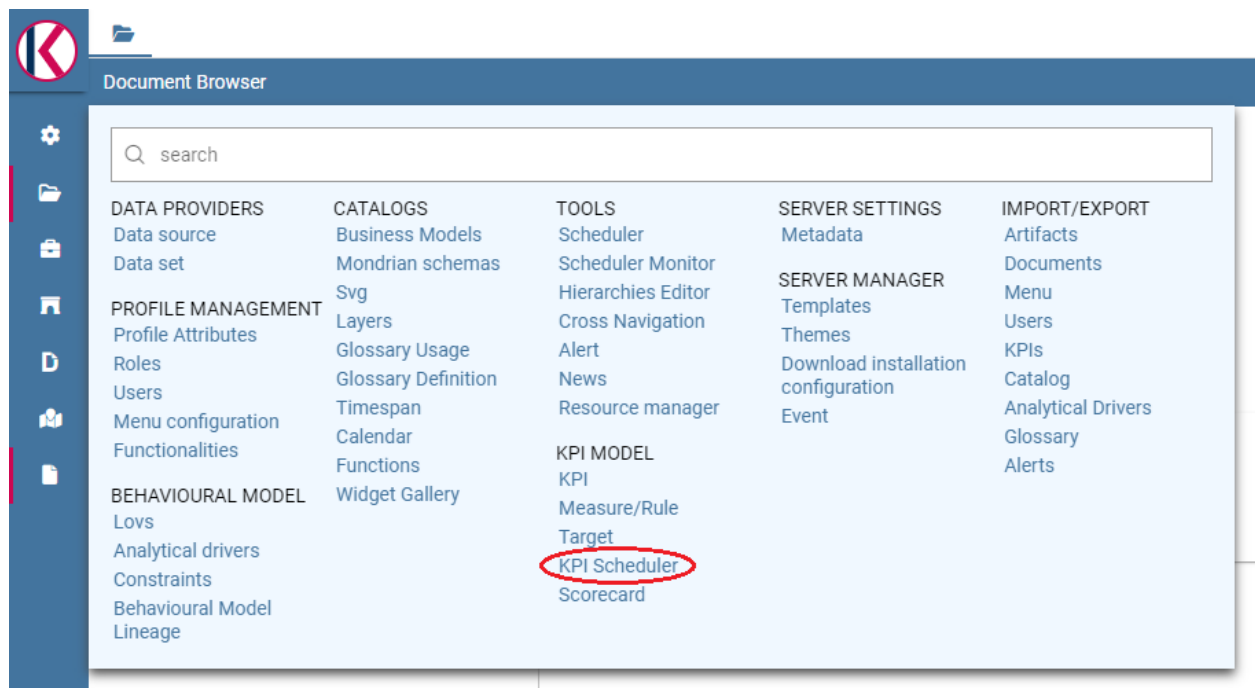


Fig. 1.450: KPI Scheduler menu item.

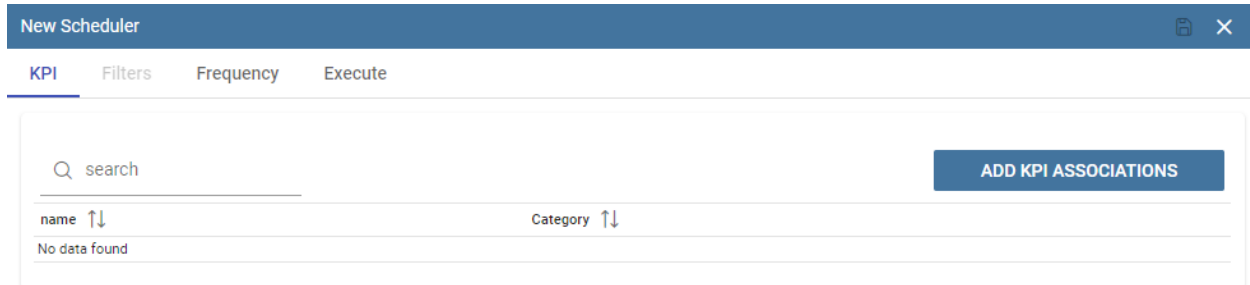


Fig. 1.451: KPI tab window.

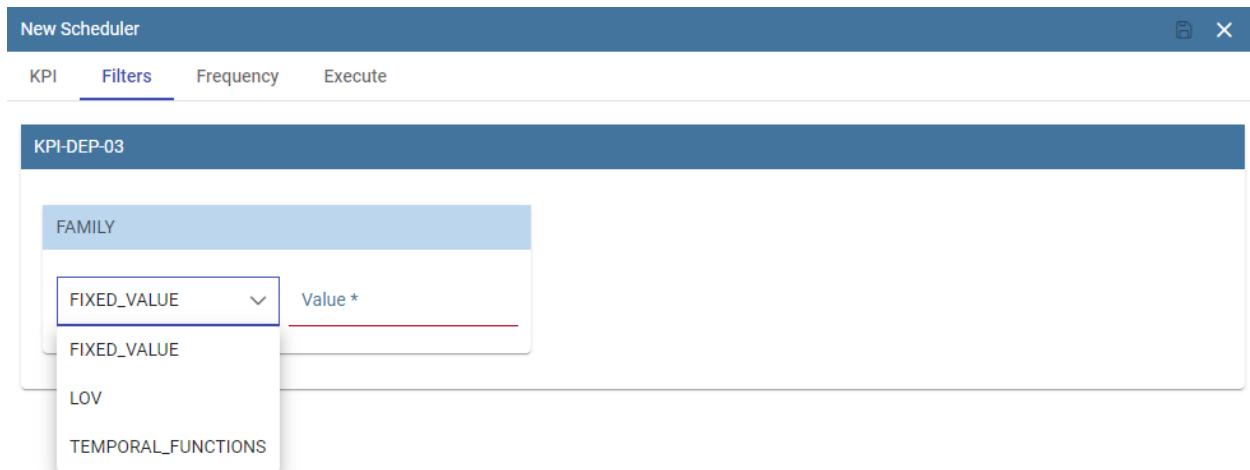


Fig. 1.452: Filters options.

- **Frequency:** here is the place where the scheduling time interval (start and end date) can be set together with its frequency.

New Scheduler

KPI Filters **Frequency** Execute

Start Date: 9/23/2022 Start Time: 17:43

End Date:

Repeat Interval: * Per minute Execution Every 1 Minutes

Fig. 1.453: Frequency tab window.

- **Execute:** here you can select the execution type. The available options distinguish between the storing and the removal of old logged data. In fact, selecting **Insert and update** the scheduler compute the current (accordingly to the frequency choice) KPI values and store them in proper tables without deleting the old measurements and all error log text files are available right beneath. While selecting **Delete and insert** the previous data are deleted.

AMOUNT_COMPUTATION

KPI Filters Frequency **Execute**

Execution Type

☐ Insert and Update

☒ Delete and Insert

Log Execution

Number Of Executions: 10 **LOAD**

TimeRun ↑↓	Error Count ↑↓	Success Count ↑↓	Total Count ↑↓
2018-11-07 14:00:00	0	7950	7950
2018-11-07 13:55:00	0	7950	7950
2018-11-07 13:50:00	0	7950	7950
2018-11-07 13:45:00	0	7950	7950
2018-11-07 13:40:00	0	7950	7950
2018-11-07 13:35:00	0	7950	7950

Fig. 1.454: Execute tab window.

In Figure below we sum up the example case we have referred to since now.

Once the scheduling is completed click on the “Save” button. Remember to give a name to the scheduling as in the following figure.

The figure consists of three screenshots of the 'New Scheduler' dialog box in Knowage, showing different tabs.

Top Screenshot (KPI Tab): The 'KPI' tab is selected. It features a search bar, a table with columns 'name' and 'Category', and an 'ADD KPI ASSOCIATIONS' button. The table contains one row: 'KPI_DEFINITION' and 'NEW_CAT'.

Middle Screenshot (Frequency Tab): The 'Frequency' tab is selected. It shows scheduling parameters: 'Start Date' (9/23/2022), 'End Date' (9/26/2022), 'Start Time' (18:00), 'End Time' (18:00), and 'Repeat Interval' (Per minute Execution, Every 1 Minutes).

Bottom Screenshot (Execute Tab): The 'Execute' tab is selected. It shows the 'Execution Type' section with two radio buttons: 'Insert and Update' (selected) and 'Delete and Insert'.

Fig. 1.455: Overview of the KPI case.

The figure shows the 'INCOME' dialog box with the 'EXECUTE' tab selected. A 'SAVE SCHEDULER' modal is open, showing the 'Name' field with 'Income' entered and a 'SAVE' button.

Fig. 1.456: Creation of a KPI Document.

1.12.1.2 Creation of a KPI document

Now the scheduling has been set and it is possible to visualize the results. The user need at this point to create a new analytical document of type KPI and that uses the KPI engine (Figure below). Add also a Label, a Name and a State for the document and then save clicking on the icon at the top right corner.

The screenshot shows the 'New Document' interface. The 'Document Details' tab is active, with the 'Informations' sub-tab selected. The 'Informations' section contains several input fields: 'Label *' with the value 'KPI_Income', 'Name *' with 'Income', a 'Description' field, 'Type *' set to 'Kpi', 'Engine *' set to 'Kpi Engine', 'State *', and a 'Refresh (Seconds)' field with the value '0'. Below these are two toggle switches: 'Visible' (checked) and 'Locked' (unchecked). The 'Position' section on the right shows 'Visibility Restrictions' and a tree view of 'Functionalities'. The tree is expanded to show 'Analytical Engines', which includes 'Cockpit', 'Impala', 'Hive', 'SparkSQL', 'MongoDB', and 'PostgreSQL'.

Fig. 1.457: Overview of the KPI case.

After saving the document click on **Open designer** link to develop the template. Here you can choose between KPI and Scorecard (refer to Scorecard Chapter for details on the Scorecard option). In the KPI case it is possible to choose between the two following type of document.

- **List:** with this option it is possible to add several KPI that will be shown in the same page with a default visualization.
- **Widget:** with this option it is always possible to add several KPI that will be shown in the same page but in this case you will also be asked to select its visualization: Speedometer or KPI Card; then you have to add the minimum value and the maximum value that the KPI can assume and if you want to add a prefix or a suffix (for example the unit of measure of the value) to the showed value.

Then you must add the KPI association using the KPI List area of the interface. As you can see in figure below you can select the KPI after clicking on the **ADD KPI ASSOCIATION** link. The latter opens a wizard that allows to select one or more KPIs. Once chosen, you need to specify all empty fields of the form, like “View as”. minimum and maximum value for the range and so on (refer to figure below). Note that the “View as” field is were you can decide if the widget will be a Speedometer or a KPI Card.

Moreover, you can set the other properties of the KPI document using the **Options** and the **Style** areas at the bottom of the page.

In particular, it is possible to select the time granularity used by the KPI engine to improve the performances. For this purpose, in the “Options” area (following figure) the user is invited to indicate the level of aggregation choosing among “day”, “week”, “month”, “quarter”, “year”.

Kpi Document Designer

KPI

Scorecard

Type Of Document

List

Widget

KPI List

search

ADD KPI ASSOCIATIONS

name

Category

View as

Range Min Value

Range Max Value

Prefix / Suffix label

Show label as

KPI_DEFINITION

NEW_CAT

Kpi Card

0

300

Prefix

Style

Font-Size

Extra-Small (8px)

Font-Size

Roboto

Font Weight

Normal

Color

Options

Show Value

Show Percentage

Precision

1

Show Target

Show Threshold

Units

Month

Fig. 1.458: Setting the KPI associations using the dedicated area.

ADD KPI ASSOCIATIONS

Range Max Value

Prefix / Suffix label

Show label as

300

Prefix

Options

Show Value

Show Percentage

Precision

1

Day

Week

Month

Quarter

Year

Units

Month

Fig. 1.459: Choose the time granularity.

Finally in the “Style” area the user can customize the size of the widget, the font, the color and size of texts.

Then save and run the document.

In case the document contains KPIs that involves grouping functions upon some attributes, it is possible to filter data returned on those attributes. To easily retrieve the attributes on which measures are grouped, it is sufficient to check the fields listed in the “Cardinality” tab of the KPI definition. We recall it in the picture below.

KPI_DEFINITION ALIAS 🔍 ✕

Formula Cardinality Threshold

$(\Sigma(\text{store_sales}) - \Sigma(\text{cost_store})) / \text{COUNT}(\text{sales_unit})$

	store_sales	cost_store	sales_unit
product_name	✓	✓	✓

Fig. 1.460: Cardinality settings example.

Then to use them to filter the document, first add the proper analytical drivers. Refer to **Analytical Document** section to get more information about how to associate an analytical driver to a document (and therefore to a KPI document). It is mandatory that the URL of the analytical driver *must* coincide with the *attribute aliases* on which you have defined the grouping.

In the following figures you can find examples on the three type of KPIs you can design: Speedometer, KPI Card and KPI List.

1.12.2 Create Scorecard

The **Scorecard** feature, available in Knowage suite as highlighted in the following figure, allows to supervise different KPIs at the same time. This option gives an exclusive complete overview of the KPIs situation when the user is not interested in a single threshold check. This tool is in fact useful when the user is interested in monitoring the overcoming of two or more critical KPI values.

1.12.2.1 Scorecard development

A scorecard is structured in hierarchical levels. Shortly, there is a first level called **Perspectives** composed of KPIs grouped on targets. Otherwise, **Targets** are assigned a threshold depending on the KPIs they are composed of. In the following we will describe in detail a scorecard configuration. When clicking on the Scorecard menu item the window of figure below opens. Here the implemented scorecards are listed and can be opened and edited.

The “Plus” icon available at the right top corner of the page opens a new window where to set a new scorecard, as shown below. Assign a name and click on **Add perspective** (Figure below).

A perspective allows you to organise the monitoring over targets.

An example is given in the following figure.

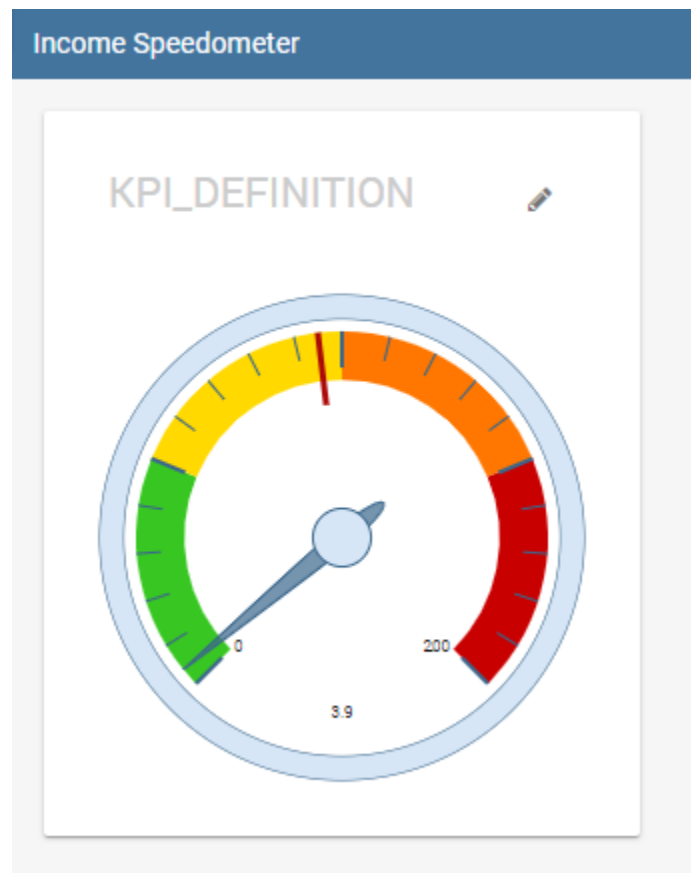


Fig. 1.461: KPI Speedometer.

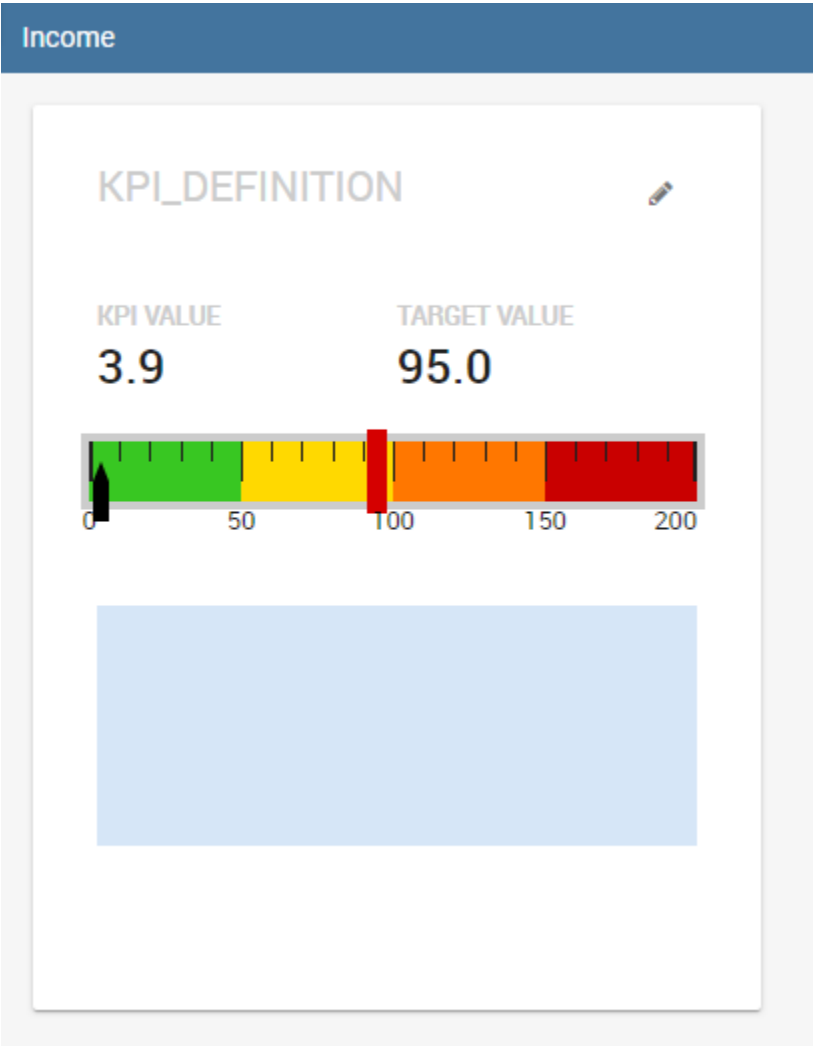


Fig. 1.462: KPI Card.

Income List				
Severity	Name	Value	Thresholds e target	Trend
● LOW	KPI_DEFINITION	3.9	<div></div>	<div></div>

Fig. 1.463: KPI List.

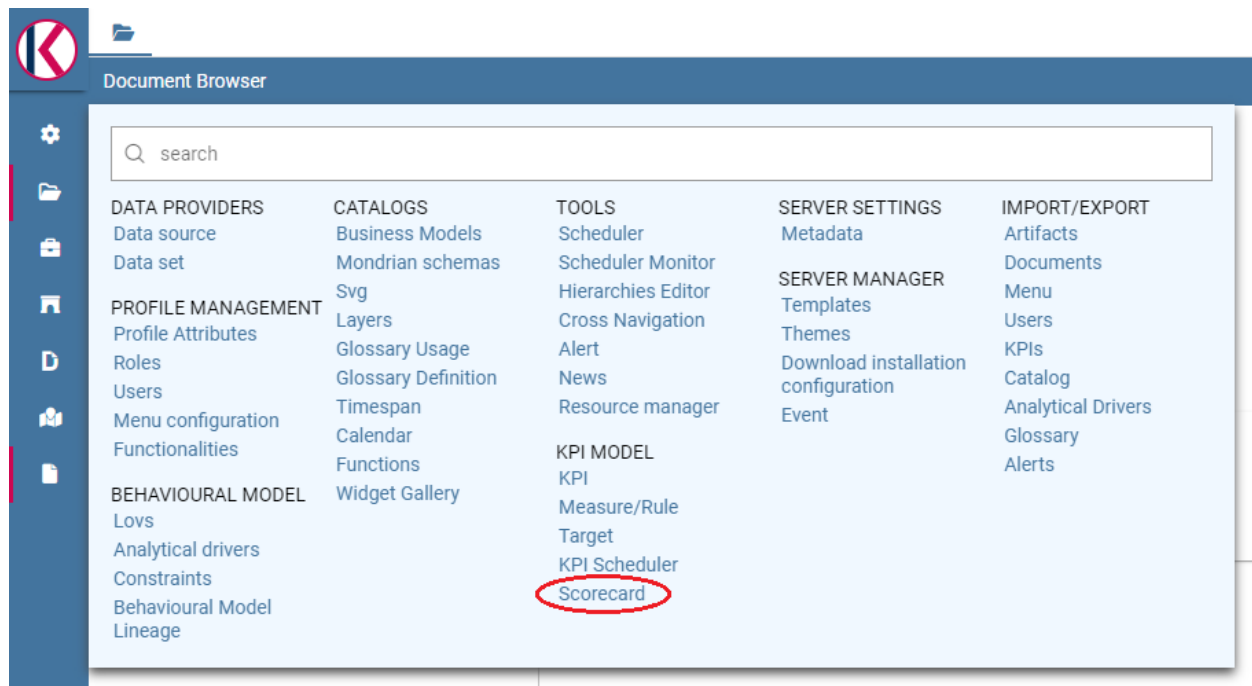


Fig. 1.464: Scorecard from the contextual menu.

Scorecard List			
Q search			
name ↑↓	Creation Date ↑↓	Author ↑↓	
Scorecard_01	11/05/2018	kte_admin	🗑
Scorecard_02	11/05/2018	kte_admin	🗑
scorecard_3	05/05/2022	kte_admin	🗑

Fig. 1.465: Scorecard window.

Scorecard Designer

SAVE

CLOSE


Name *

FirstScorecard

Description

Perspectives

Add Perspective

 **Scorecard Designer**

To create a scorecard at least one perspective (card) should be present. Click on add perspective button to add the first one.

Fig. 1.466: Defining a new scorecard.

Scorecard Designer

SAVE

CLOSE


Name *

FirstScorecard

Description

Perspectives

Add Perspective

>  New Perspective

M

MP

P



 

Fig. 1.467: Add perspective to the scorecard.

Scorecard Designer

SAVECLOSE

Name *

Company Scorecard

Description

Perspectives

Add Perspective

Warehouse	M	MP	P	+	✕
Warehouse	M	MP	P	+	✕
KPI-ARR-01					✕
Profit	M	MP	P	+	✕
Profit	M	MP	P	+	✕
KPI-DEP-02					✕

Fig. 1.468: Perspective list example.

Each perspective manages one or more targets accordingly to the user’s requirements. A target consists of one or more KPIs and it is assigned a threshold color according to the chosen **Evaluation criterion**. The available evaluation criterion are:

- **Policy “Majority” (M)**, the target gets the threshold of the KPI threshold that numerically exceeds the others,
- **Policy “Majority with Priority (MP)”**, the target gets the threshold of a specific KPI,
- **Policy “Priority” (P)**, the target gets the majority threshold of the KPIs in case the primary stated KPI returns the lower threshold, namely the “green” one, while it gets the threshold of a primary stated KPI in case the latter returns the other thresholds, namely the “yellow” or the “red” one.

Warning: Thresholds of selected KPIs must have the right colors

Note that the scorecard shows the right colors accordingly with the selected policy only if the KPIs which compose the targets have **no filters** and **standard colors** (see the **Create KPI** section for definitions) to highlight the threshold.

Warning: “Standard” colors for thresholds

When the targets contain parametric KPIs the target/perspective evaluation cannot be completed for value absence. Therefore the warning lights turn grey. The right visualization of the scorecard must be implemented through a scorecard document. Check the following to have more details on how to develop a scorecard document.

An example is showed below.

The same choice is available at the perspective level (refer to next figure), that is:

- **Policy “Majority”** the perspective gets the threshold of the target threshold that numerically exceeds the others,
- **Policy “Majority with Priority”** the perspective gets the threshold of a specific target,
- **Policy “Priority”** the perspective gets the majority threshold of the targets in case the primary stated target

Scorecard Designer		SAVE	CLOSE
Name * Company Scorecard		Description	
Perspectives			Add Perspective
Warehouse	M	MP	P
Warehouse	M	MP	P
KPI-ARR-01			
Profit	M	MP	P
Profit	M	MP	P
KPI-DEP-02			

Fig. 1.469: Select the KPI with priority.

returns the lower threshold, namely the “green” one, while it gets the threshold of a primary stated target in case the latter returns the other thresholds, namely the “yellow” or the “red” one.

Scorecard Designer		SAVE	CLOSE
Name * Company Scorecard		Description	
Perspectives			Add Perspective
Warehouse	M	MP	P
Warehouse	M	MP	P
KPI-ARR-01			
Profit	M	MP	P
Profit	M	MP	P
KPI-DEP-02			

Fig. 1.470: Perspective policy.

Remember to save once perspectives and targets have been set.

1.12.2.2 Creation of a Scorecard document

Once saved it is possible to develop a scorecard document which can be easily consulted by (authorized) end users. To create a scorecard document click on the “Plus” icon available in the document browser and then “Generic document”.

Here fill in mandatory fields (marked with an asterisk) selecting the **KPI type** and **KPI engine** and save.

Then click on “Open Designer” to develop the template. Here select the “Scorecard” option as in figure below and choose an existing scorecard from the list opened clicking on “Add Scorecard Association”. Make the desired customizations and save.

Figure below gives an example of the scorecard document interface. The arrows point out the perspectives’ achievement of the goals or, on the contrary, the missing of the targets. As well the achievement/failure of the single targets is pinpointed by the arrow signals close to each target.

Note that it is possible to check the policy used for each perspective. In fact, by clicking on one of them a wizard opens showing the policy adopted and the goal got by each KPI.

New Document

Document Details

Informations

Upload Template

Label *
Retail_scorecard

Name *
Retail Scorecard

Description

Preview Image

Type *
Kpi

Engine *
Kpi Engine

State *
Development

Refresh (Seconds)
0

Select the number of milliseconds after the document will refresh

☒ Visible

☐ Locked

Position

Visibility Restrictions

Defined Restrictions

Select Attribute = Enter Value

Select The Functionalities Where The Document Will Be Visible

- Functionalities
 - Analytical Engines
 - Cockpit
 - Impala
 - Hive
 - SparkSQL
 - MongoDB
 - PostgresSQL

Fig. 1.471: Create a generic document of type KPI.

Kpi Document Designer

☐ KPI ☒ Scorecard

Scorecard List

Q search

ADD SCORECARD ASSOCIATION

name	Creation Date
Company Scorecard	09/26/2022

Style

Font-Size
Extra-Small (8px)

Font-Size
Roboto

Font-Weight
Normal

Color
[Black]

Fig. 1.472: Template creation window.

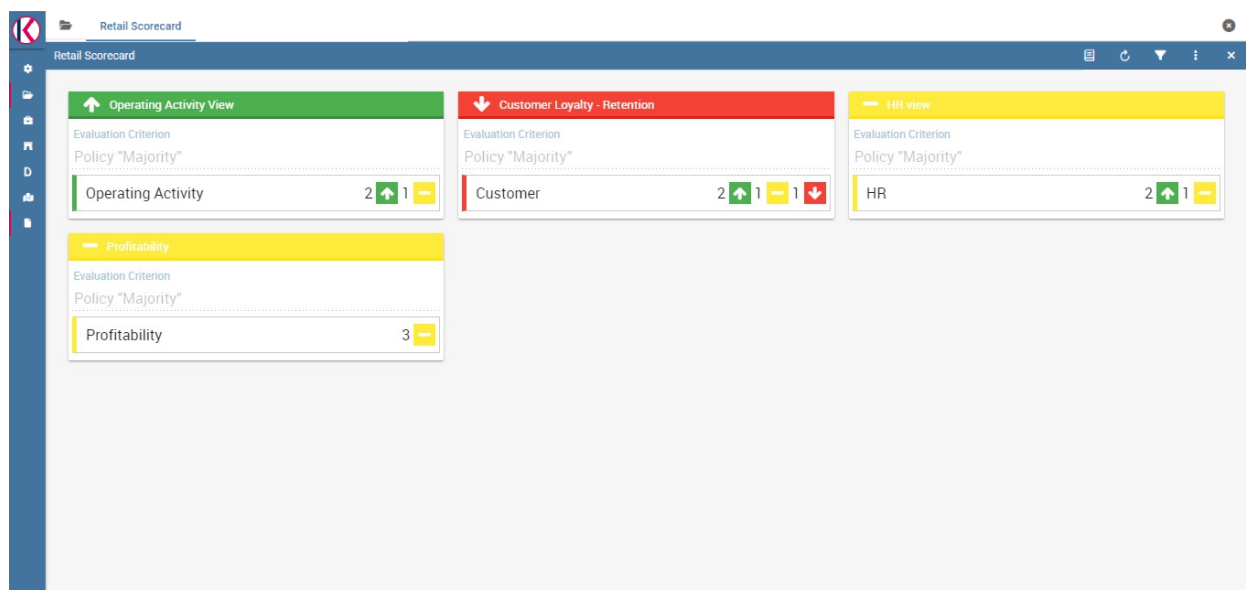


Fig. 1.473: Scorecard document interface.

1.12.3 Create Alert

The **Alert** functionality available under the **Tool** section of Knowage main menu, as shown in Figure below, allows you to implement a control over events. Technically it consists of monitoring the possible overcoming of fixed thresholds and the consequent signal of anomalies by means of an email or by launching an ETL process. In next sections we will see in details how to configure the alarms using the Alert feature.

1.12.3.1 Alert implementation

The Alert definition window (refer to the figure below) displays the list of existing alerts and a search box at the top of the page to facilitate to browse over the items. In the right top corner of the list it is available the “Plus” icon to configure a new alert.

Clicking on the “Plus” icon the page in the following figure opens.

Give a name and select the **KPI listener** from the combobox. Then indicate the start date, the start time and eventually the Alert implementation end date. Specify the time interval and how many times it must be repeated. At this point choose between **Single Execution** and **Event before trigger actions**: in the first case the alert must signal the anomaly just once its occurrence while in the other case it must send the alarms when the irregularity occurs as many times as specified. Note indeed that the box is editable and it must contain a number which indicates the number of irregularities that has to be detected before the alert starts.

At the bottom of the page associate the KPI of interest and select the action by clicking on the **Add action** button. Here you can choose between **Send mail**, **Execute ETL document** or **Context Broker**.

If the “Send mail” is chosen, the user is asked to insert the thresholds that should be monitored: in case the latter are overcome the functionality sends the email to the indicated subjects. Remember to insert all mandatory fields (name, thresholds and subjects of the mail) and, in particular, to select the user to which the email is sent. Note that in the “Mail to” box you can type both mail addresses or usernames (for example “user_admin”), helped by the automatic insertion text. In fact, Knowage server will pick out the mail address from the profile attribute associated to that user. Therefore, we recommend to set the profile attribute previously. In particular, remember that the profile attribute *must* be Alert implementation called “email”.

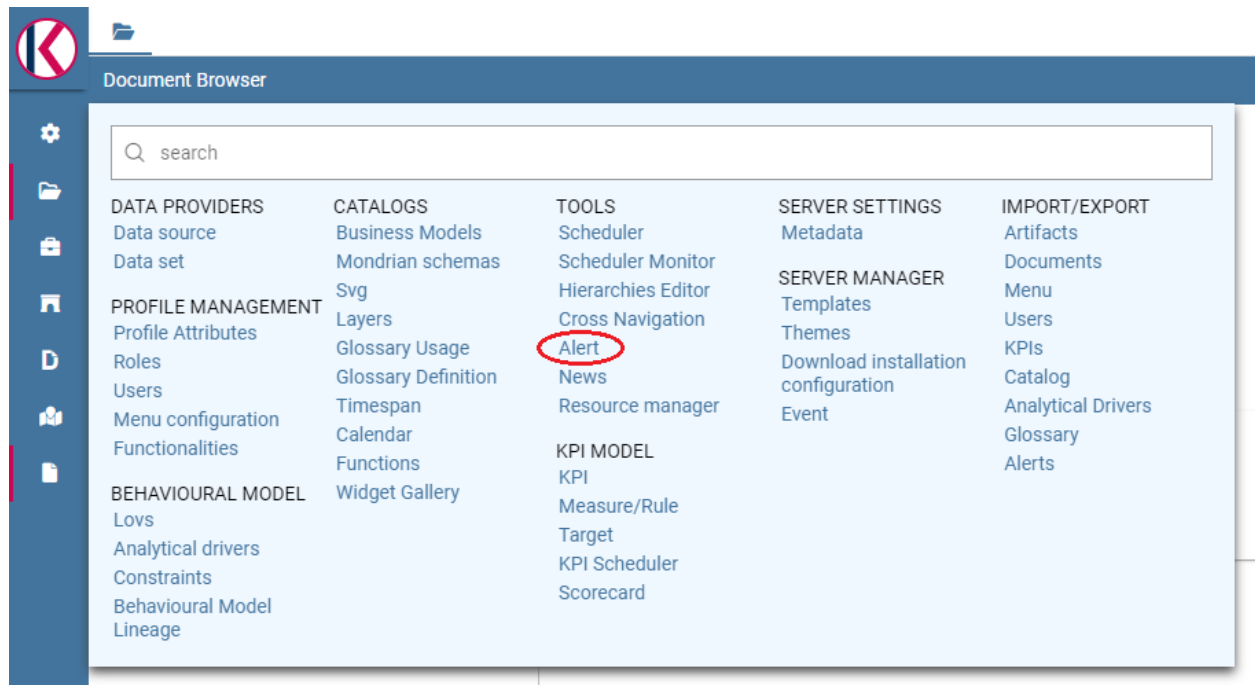


Fig. 1.474: Alert functionality from contextual menu.

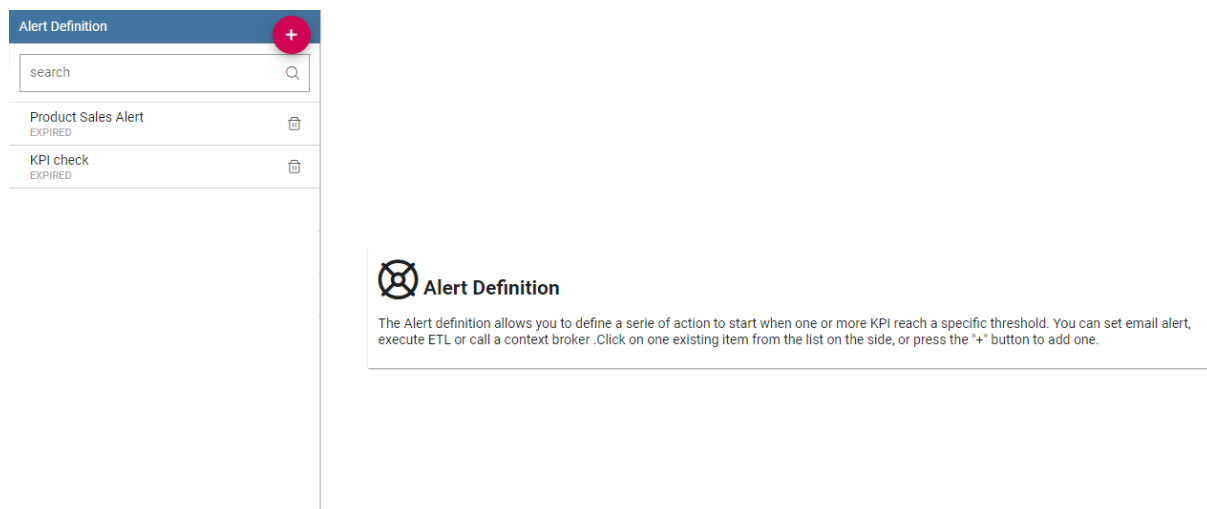


Fig. 1.475: Alert definition window.

New Alert

Name *

New alert

Listener *

KPI Listener

Start Date: 9/26/2022

Start Time: 16:06

End Date:

Repeat Interval: *

Per minute Execution

Every 1 Minutes

Number Of Events Before Triggering Actions

3

☐ Single Execution

Kpi *

KPI-DEP-02

Action List

Add new action

Fig. 1.476: Alert settings.

The screenshot shows the 'Add New Action' dialog box with the following configuration:

- Type ***: Send mail
- Threshold ***: T1 (indicated by a red square icon)
- Mail To**: name.lastname@mail.com
- Mail Subject**: (empty field)
- Mail Body**: A rich text editor containing the text "WARNING! Error detected!". The editor includes a toolbar with options for heading, font face (Sans Serif), bold (B), italic (I), underline (U), text color (A), background color (A), bulleted list, numbered list, indent, link, unlink, source code, and undo.

Fig. 1.477: Send email configuration.

Furthermore, note that the mail body can contain a plain text message or a table showing the registered KPI values which activated the alert. To enable table visualization option you must insert the placeholder `*TABLE_VALUE*` in the mail body as shown in Figure below.

If the “Execute ETL document” option is selected, the alert will launch an ETL process when the chosen thresholds are overcome. Namely, the functionality will launch a process to execute batch actions as the writing of proper tables of a DWH, creation of a file, the call to a log service. Note that the ETL process must exist as a document in Knowage server. An example is given in the following figure.

Save both the action and the alert configuration settings to store your alert. Remember that it is possible to schedule more than one monitoring over the same alert definition.

1.13 Create an ETL Process

Knowage allow the upload of data from source systems according to a common ETL logic, as well as the monitoring of data flows continuously feeding the data warehouse. To this end, Knowage provides the ETL **Knowage Talend Engine**.

1.13.1 KnowageTalendEngine

Knowage Talend Engine integrates the open source tool Talend Open Studio (TOS). Talend Open Studio (TOS) is a graphical designer for defining ETL flows. Each designed flow produces a self-alone Java or Perl package. TOS is based on Eclipse and offers a complete environment including test, debug and support for documentation.

The integration between Talend and Knowage is twofold. TOS includes Knowage as a possible execution target for its job, while Knowage implements the standard context interface for communicating with Talend. Jobs can be directly

Add New Action [Save] [Close]

Type * Send mail Threshold * T1

Mail To
name.lastname@mail.com
Press Enter to confirm the input value

Mail Subject
Alert detected

Heading Sans Serif B I U A [Image] [List] [List] [List] [Link] [Image] [Code] [Text]

Here there are the registered values.

TABLE_VALUE

Fig. 1.478: Send email containing a table with detected values.

Add New Action [Save] [Close]

Type * Execute ETL Document Threshold * T1 T2

search

<input type="checkbox"/>	Name ↑↓	Label ↑↓
<input type="checkbox"/>	CJ	COMMONJ
<input checked="" type="checkbox"/>	TC_TALEND	TC_TALEND

Fig. 1.479: Execute document configuration.

executed from Knowage web interface or possibly scheduled.

Furthermore, the analytical model for monitoring ETL flows can be successfully applied to the analysis of audit and monitoring data generated by a running job. Note that this is not a standard functionality of Knowage suite, but it can be easily realized within a project with Knowage. To create an ETL document, you should perform the following steps:

- Job design (on Talend);
- Job deploy;
- Template building;
- Analytical document building;
- Job execution.

In the remainder of the section, we discuss in detail all steps by providing examples.

1.13.2 Job design

The job is designed directly using Talend.

Designing an ETL job requires to select the proper components from Talend tool palette and connect them to obtain the logic of the ETL flow. Talend will map to appropriate metadata both the structure of any RDBMS and the structure of any possible flow (e.g., TXT, XLS, XML) acting as input or output in the designed ETL.

To design the ETL, several tools are available: from interaction with most RDBMS engines (both proprietary and open source) to support for different file formats; from logging and auditing features to support for several communication and transport protocols (FTP, POP, code, mail); from ETL process control management to data transformation functionalities.

Talend also supports data quality management. Furthermore, it enables the execution of external processes and can interact with other applications, e.g., open source CRM applications, OLAP and reporting tools.

The tMap tool allows the association of sources to targets according to defined rules. The main input is the source table in the data warehouse, while secondary (or lookup) inputs are dimensions to be linked to data. The output (target) is the data structure used for aggregation.

It is also possible to design parametric ETL jobs. We will see how to manage them in the next steps.

1.13.3 Job deploy

Once you have designed the ETL job, you should deploy it on Knowage Server. First of all, configure connections properties to Knowage Server. Select **Preferences > Talend > Import/export** from within Talend. Then set connection options as described below.

Table 1.12: Connection Settings.

Parameter	Value
Engine name	KnowageTalendEngine
Short description	Logical name that will be used by Talend
Host	Host name or IP address of the connection URL to Knowage
Port	Port of the connection URL to Knowage
Host	Host name or IP address of the connection URL to Knowage
Password	Password of the user that will perform the deploy

Once you have set the connection, you can right click on a job and select **Deploy on Knowage**. This will produce the Java code implementing the ETL and make a copy of the corresponding jar file at `\\resources\\talend\\RuntimeRepository\\java\\Talend project name of Knowage Server`. It is possible to deploy multiple jobs at the same time. Exported code is consistent and self-standing. It may include libraries referenced by ETL operators and default values of job parameters, for each context defined in the job. On its side, Knowage manages Talend jobs from an internal repository at `resources/talend/RuntimeRepository`, under the installation root folder.

1.13.4 Template building

As with any other Knowage document, you need to define a template for the ETL document that you wish to create. The ETL template has a very simple structure, as shown in the example below:

Listing 1.55: ETL template.

```

1 <etl>
2   <job    project="Foodmart"
3         jobName="sales_by_month_country_product_familiy"
4         context="Default"
5         version="0.1"
6         language="java"
7   />
8 </etl>

```

The tag job includes all the following configuration attributes:

- project is the name of the Talend project.
- jobName is the label assigned to the job in Talends repository.
- context is the name of the context grouping all job parameters. Typically it is the standard context, denoted with the name **Default**.
- version is the job version.
- language is the chosen language for code generation. The two possible options are: Java and Perl.

Values in the template must be consistent with those defined in Talend, in order to ensure the proper execution of the ETL document on Knowage Server.

1.13.5 Creating the analytical document

Once we have created the template, we can create a new analytical document following the standard procedure. Use the plus button and choose “Generic Document”. Proceed by filling in the necessary fields, uploading the XML template we have just created and selecting “ETL Process” as the type and “Talend engine” as the engine. Select then the corresponding data source and the document state.

If the job has parameters, they should be associated to the corresponding analytical drivers, as usually. In other words, you have to create an analytical driver for each context variable defined in the Talend job.

1.13.6 Job execution

A Talend job can be executed directly from the web interface of Knowage Server and of course from a Talend client. To execute the job on Knowage, click on the play button near the document in the document browser, like with any other analytical document. The execution page will show a message to inform that the process was started.

Informations

Upload Template
etl_template.xml

Label *

DOC-ETL

Name *

DOC-ETL

Description

Preview Image

Type *

ETL Process

Engine *

Talend Engine

Data Source

Foodmart

State *

Released

Refresh (Seconds)

0

Visible

Locked

Fig. 1.480: ETL document creation interface.

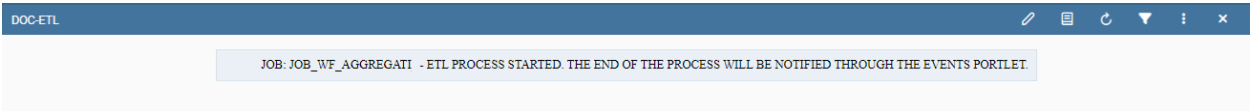


Fig. 1.481: ETL document execution interface.

1.13.7 Job scheduling

Most often it is useful to schedule the execution of ETL jobs instead of directly running them. You can rely on Knowage scheduling functionality to plan the execution of Talend jobs. While defining a scheduled execution, you can set a notification option which will send an email to a set of recipients or a mailing list once the job has completed its execution. To enable this option, check the flag **Send Mail**.

1.14 How use Localization Management

Knowage allows to change the *localization* in order to use different languages. We can distinguish two main categories:

- **System labels:** the labels of the Knowage Graphical User Interface
- **User labels:** labels created by the user in documents or objects created

The user can change the language to use in Knowage by doing the following:



Fig. 1.482: How to change language in Knowage.

1.14.1 System labels

It is also possible to change the default language to use the first time the user logs in; to do this the admin needs to:

- Open the menu and select under Server Settings, Configuration Management
- Change the property SPAGOBI.LANGUAGE_SUPPORTED.LANGUAGE.default and insert the locale under the column *valueCheck* using the format languageCode, nationalVariety (e.g.: en,US)

The currently supported languages for system labels are: - Italian - English - French - Spanish

SPAGOBI.LANGUAGE_SUPPORTED.LANGUAGE.default	default	en,US	LANGUAGE_SUPPORTED
---	---------	-------	--------------------

Fig. 1.483: Change default language.

It is also possible to translate objects created by the user like:

- Analytical Driver name shown in the Parameter Panel
- Name and description of an analytical document
- Name of a menu entry
- Name of a dataset shown in the Workspace
- Name of a business model shown in the Workspace
- Name of a federation shown in the Workspace
- Name of a functionality (folder) in the document browser
- Name of a cross navigation when requested to a final user
- Inside a dashboard: Title of columns in a table widget and crosstab, title, subtitle, no data message, axis and legend title in the charts

1.14.2 User labels

To set *User labels*:

- Check the available languages under the Domains configuration in the Admin menu (see the entry with domain code LANG). The ValueCd must have the ISO code of the language (three letters).
- **Inside the knowage metadata database the table SBI_I18N_MESSAGES contains the labels translations.**
 - LANGUAGE_CD: must be the ID of the language from the SBI_DOMAINS table
 - LABEL: is the label name to be used as placeholder, must begin with i18n
 - MESSAGE: contains the translated string in that particular language that corresponds to that label
 - ORGANIZATION: the tenant name

1.14.3 BIRT labels

Select the report and set the resource file bundle name in the resources tab of the Property Editor; the default message bundle is *messages*. The name of the properties file is *messages_* plus the current locale, e.g. for English USA language: *messages_en_US.properties*

The messages file has a syntax like this for each label, *message_code* = *internationalised message* For example: *title_code* = title of the document

To localize a message in a field within a report template, the user needs to define the code filling the Text key field by clicking on the Localization voice under the field properties tab.

1.15 How create a Data Preparation

1.15.1 What is that for?

Data Preparation is a functionality available since version 8.1 that allows users to create a prepared dataset, starting from an existing one. In this way users can create a prepared dataset and then using it inside a dashboard or simply using it as another dataset. Users have the capability of creating complex and specific datasets quite easily and quickly, in order to use them for any purpose.

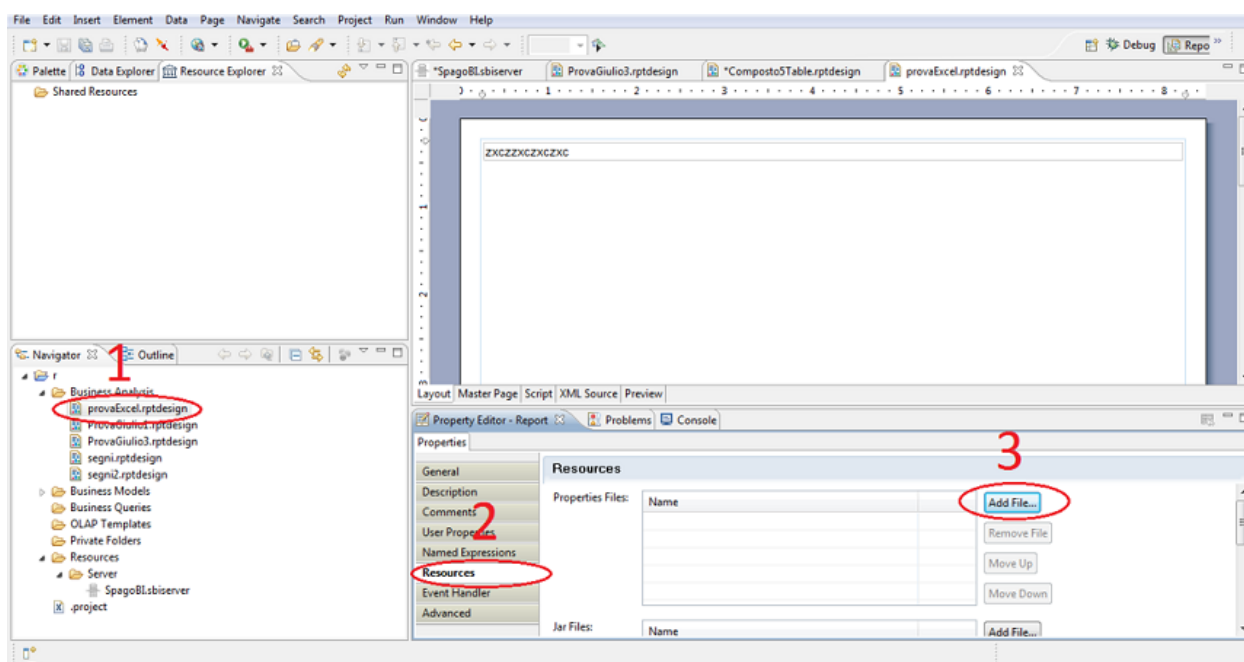


Fig. 1.484: BIRT Property Editor - Resources

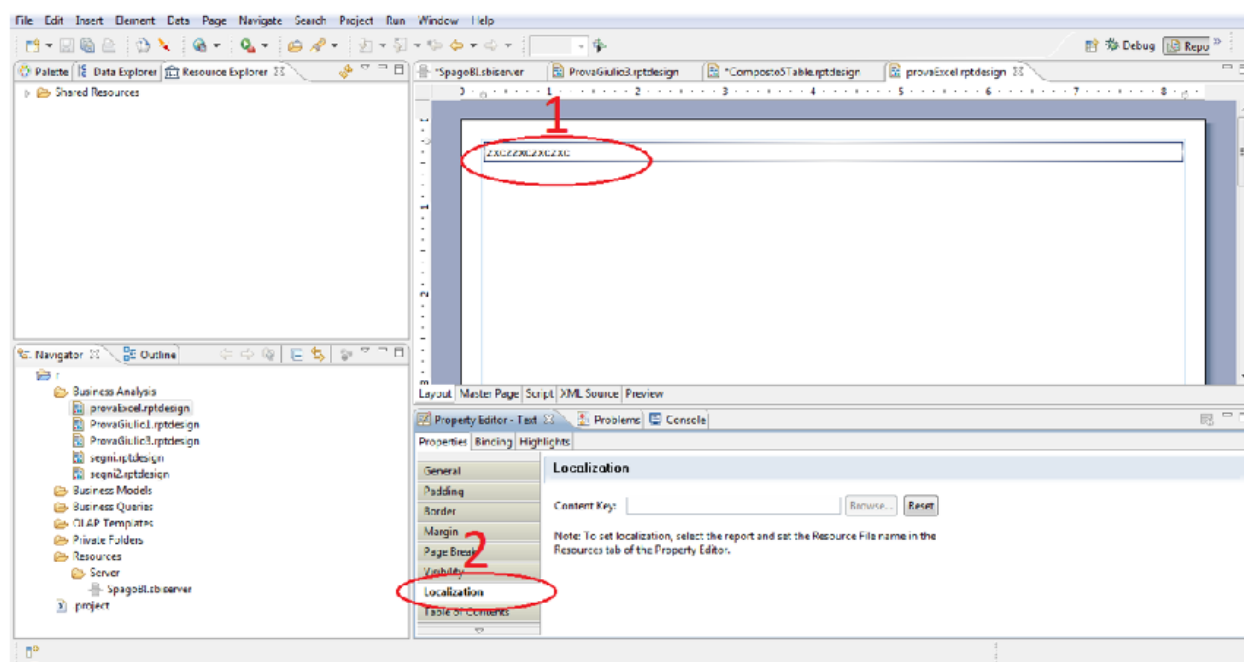


Fig. 1.485: BIRT Property Editor - Localization

1.15.2 Preparing a dataset

The first step is selecting a dataset to work on. You can do it selecting the desired dataset into your workspace section (there should be also a **Data Preparation** entry):

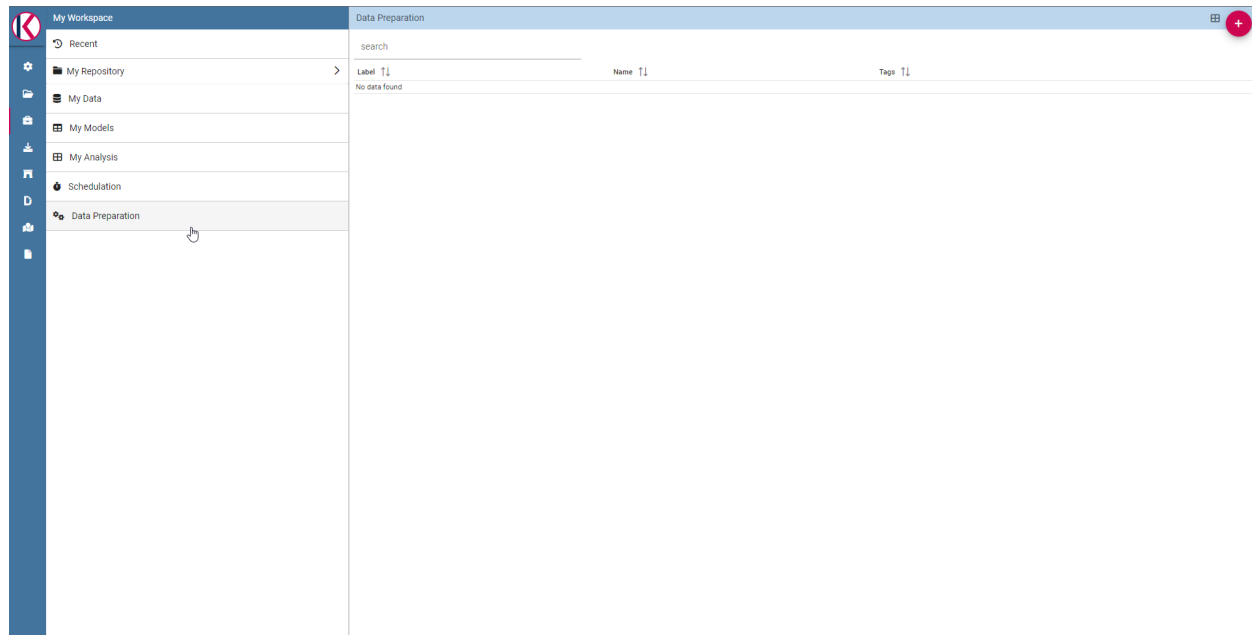


Fig. 1.486: The Data Preparation entry.

There are two ways to prepare and start a data preparation process. The first one is starting from MyData section. Click on MyData and select your desired dataset.

Then click with the right click of mouse and select “Open Data Preparation”:

This operation triggers the avro-file creation process.

The second way is clicking on + button on the top right of Data Preparation section:

And then select your source dataset.

If avro file is created now you should be able to open the data preparation screen, there should be a check icon next to the actions menu:

Data preparation main screen:

1.15.3 Data Preparation Transformations

You can apply transformations to the source dataset just picking the transformation action, step by step, until you reach the desired result.

In the main toolbar menu, there is a set of main transformations (most of them can be applied on many columns in the same time):

- **Add column:** Adds a new column as a calculated field.
- **Merge columns:** Adds a new column merging two selected ones.
- **Split columns:** Adds two columns splitting a selected one.
- **Filter:** Filters a selected column by math conditions (more info later on).

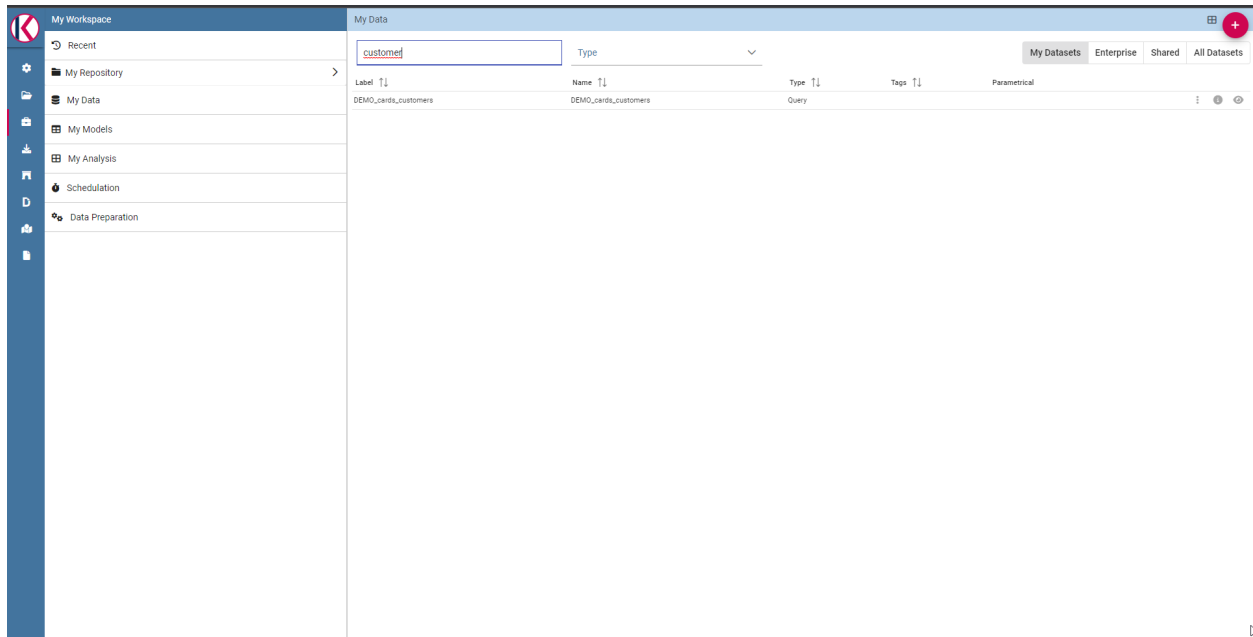


Fig. 1.487: Search for your dataset example.

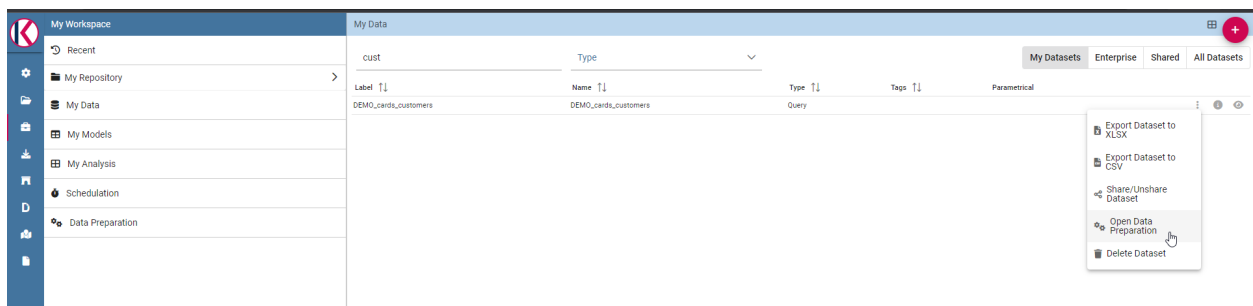


Fig. 1.488: Select “Open Data Preparation”

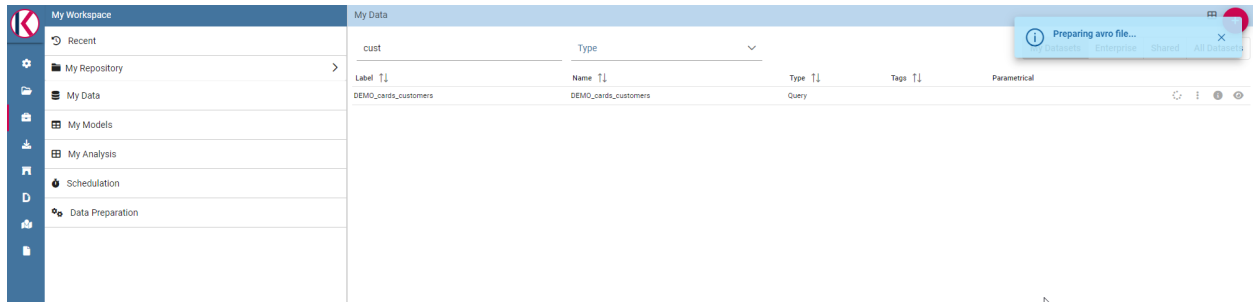


Fig. 1.489: For more info about what “Preparing avro file” means please refer to “Data Preparation technical detail” section.

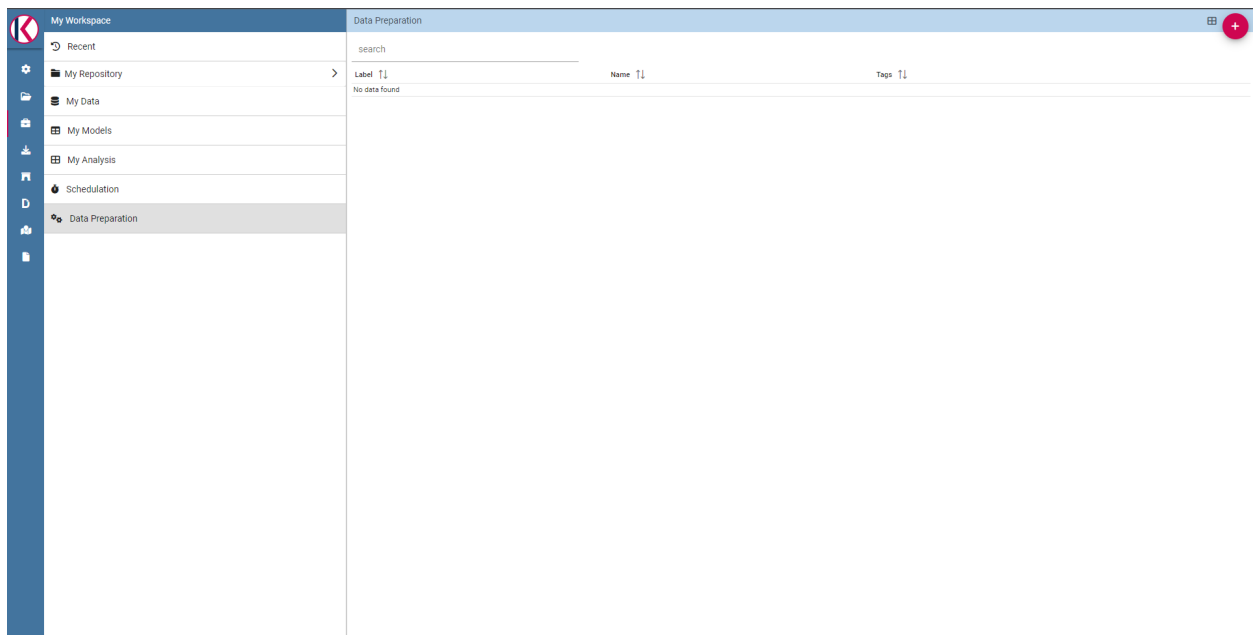


Fig. 1.490: Click on + button on the top right of the page.

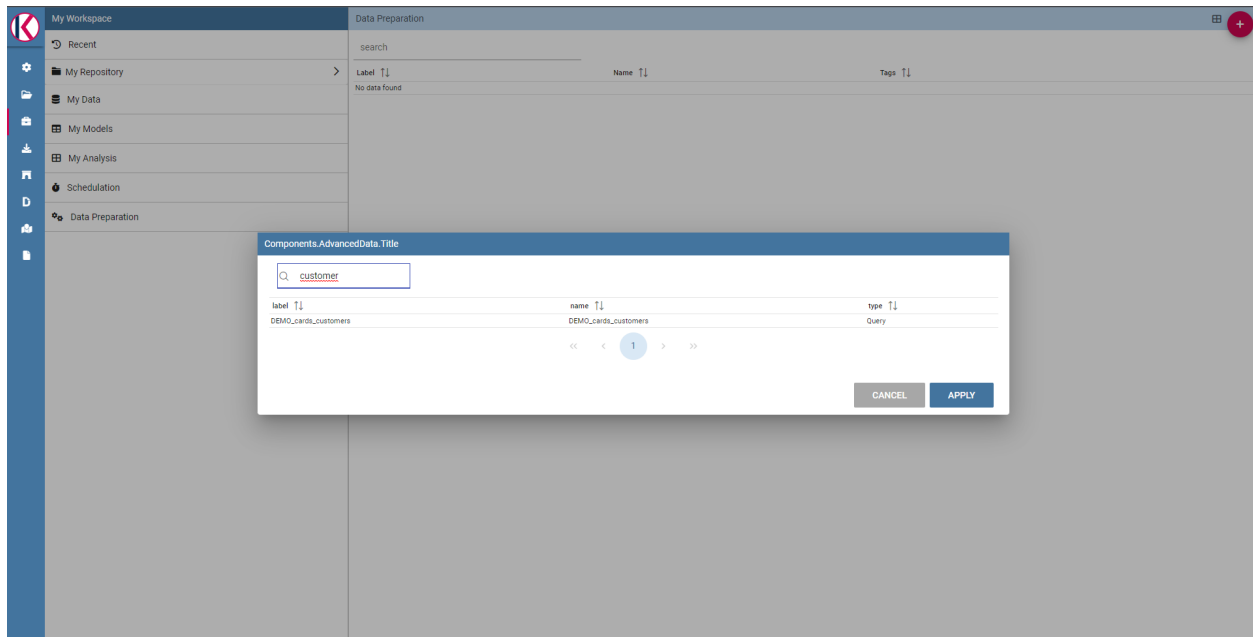


Fig. 1.491: Search for your dataset example.

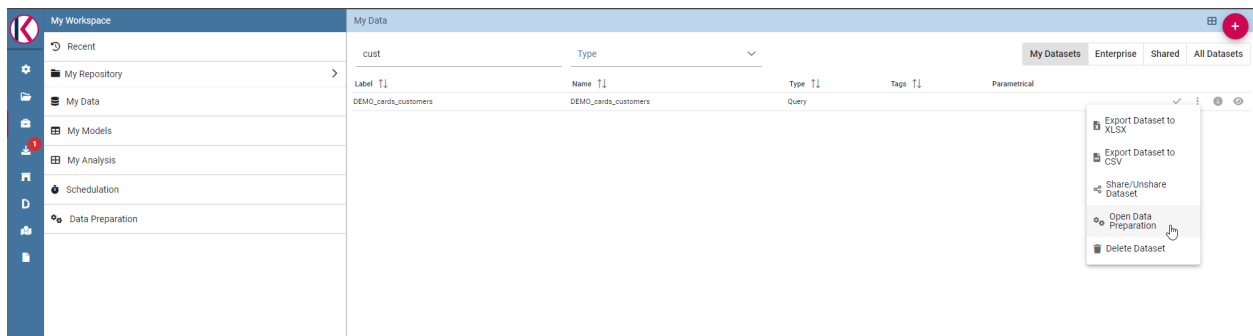
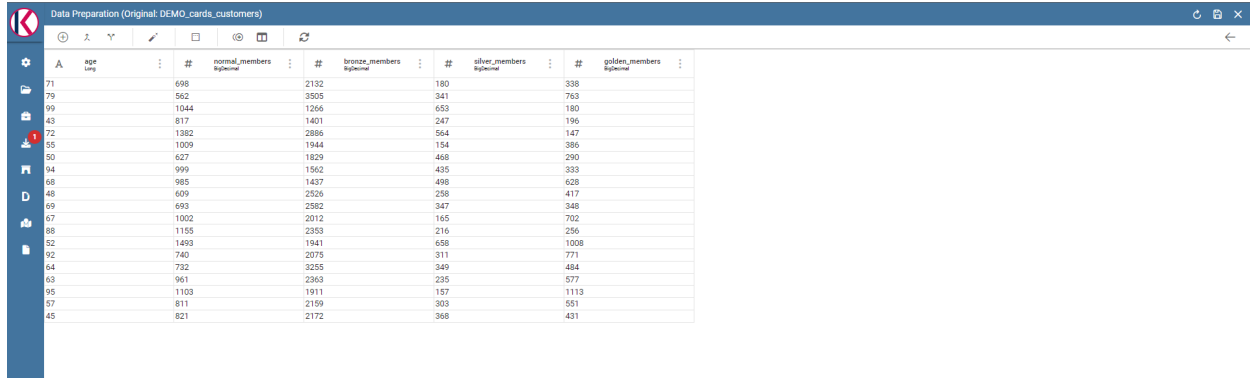


Fig. 1.492: There is a check icon on the left icons menu for your selected dataset.



	age	#	normal_members	#	bronze_members	#	silver_members	#	golden_members
71	698		2132		180		338		
79	562		3905		341		763		
99	1044		1266		653		180		
43	817		1401		247		196		
72	1382		2886		564		147		
55	1009		1944		154		386		
50	627		1829		468		290		
94	999		1562		435		333		
68	985		1437		498		628		
48	609		2326		256		417		
69	693		2582		347		348		
67	1002		2012		165		702		
88	1155		2353		216		256		
52	1493		1941		658		1008		
92	740		2075		311		771		
64	732		3255		349		484		
63	961		2363		235		577		
95	1103		1911		157		1113		
57	811		2159		303		551		
45	821		2172		368		431		

Fig. 1.493: Data Preparation main screen.

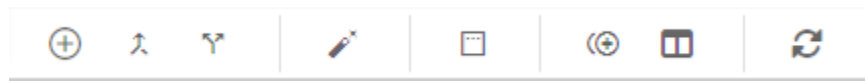


Fig. 1.494: Transformations toolbar icons.

- **Padding:** Adds characters on left or right side of a selected column.
- **Remove duplicates:** Removes duplicates from selected columns.
- **Remove null:** Removes null values from selected columns.
- **Replace:** Replace selected values from specific columns.
- **Trim:** Removes white spaces from specific columns. (Available for single column only)
- **Drop:** Remove a specific columns. (Available for single column only)

The **Add column** transformation allow user to add a **calculated field** of type numeric, string or temporal. These functions are a subset of Spark SQL language functions and are used for calculations or manipulating data. For more info see <https://spark.apache.org/docs/2.4.8/api/sql/index.html>.

Merge columns: Adds a new column merging two selected ones using a separator.

Split columns: Creates two new columns splitting a selected one using a specific condition (ie a character).

Filter: Filters a selected column by special conditions.

Padding: Adds characters on left or right side of a selected column.

Remove duplicates: Removes duplicates from selected columns.

Remove null: Removes null values from selected columns.

Replace: Replace selected values from specific columns. Old char is the old value to be replaced.

Two more transformations are present only by clicking on a specific column: **TRIM** and **DROP** transformations.

Drop column: Removes a specific column from table.

Trim column: Removes white spaces from column.

1.15.4 Data Preparation technical detail

What is an AVRO file?

Avro is a data serialization system.

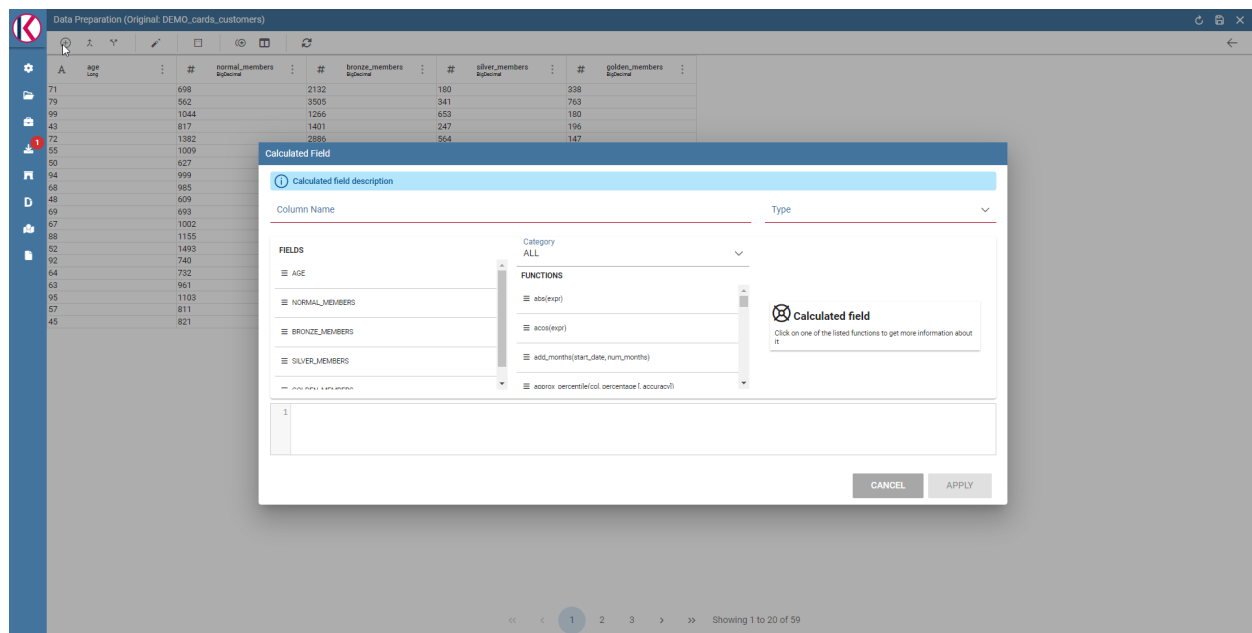


Fig. 1.495: Available functions are a subset of Spark SQL language functions

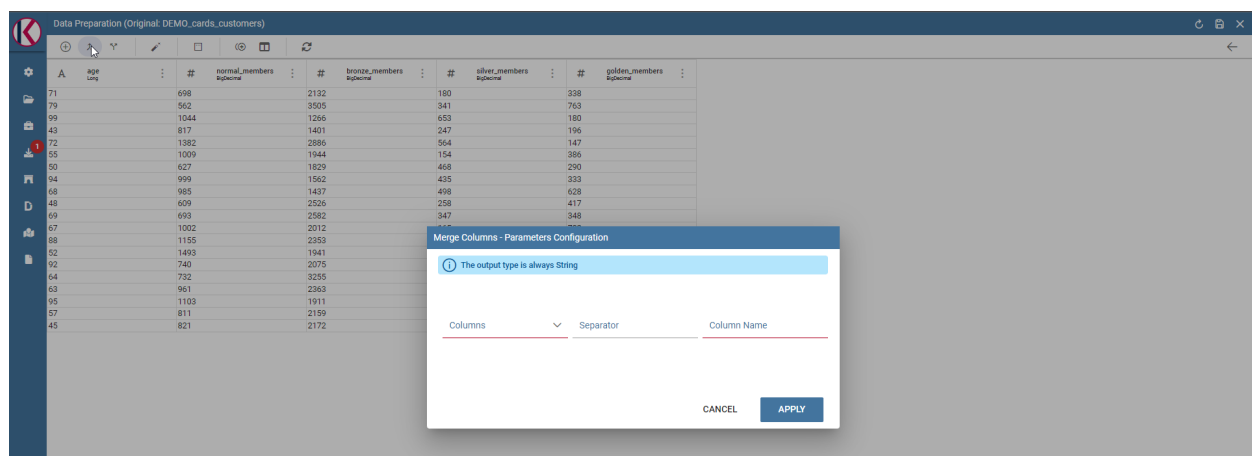


Fig. 1.496: Merge columns dialog example.

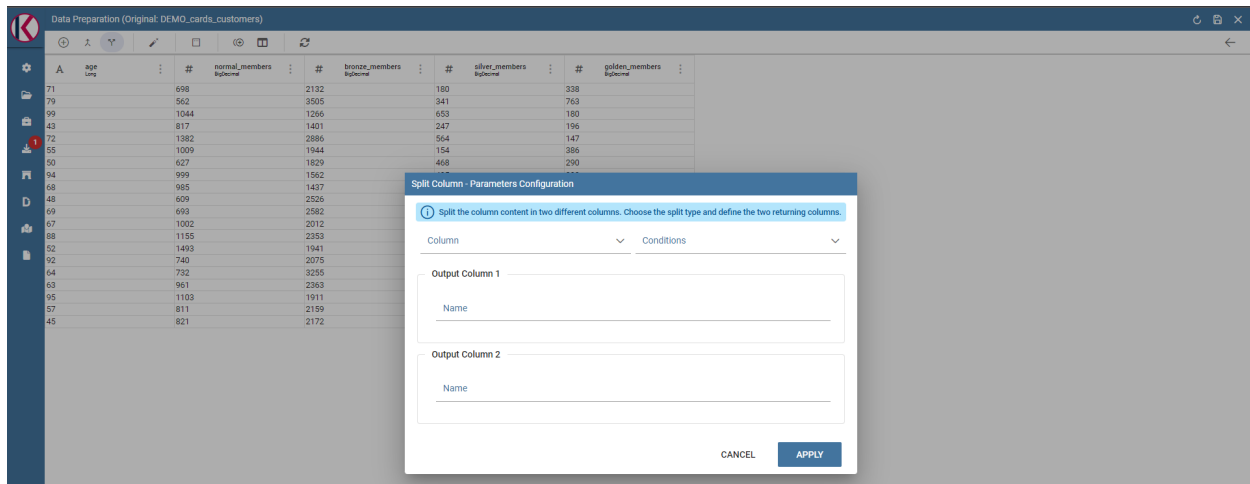
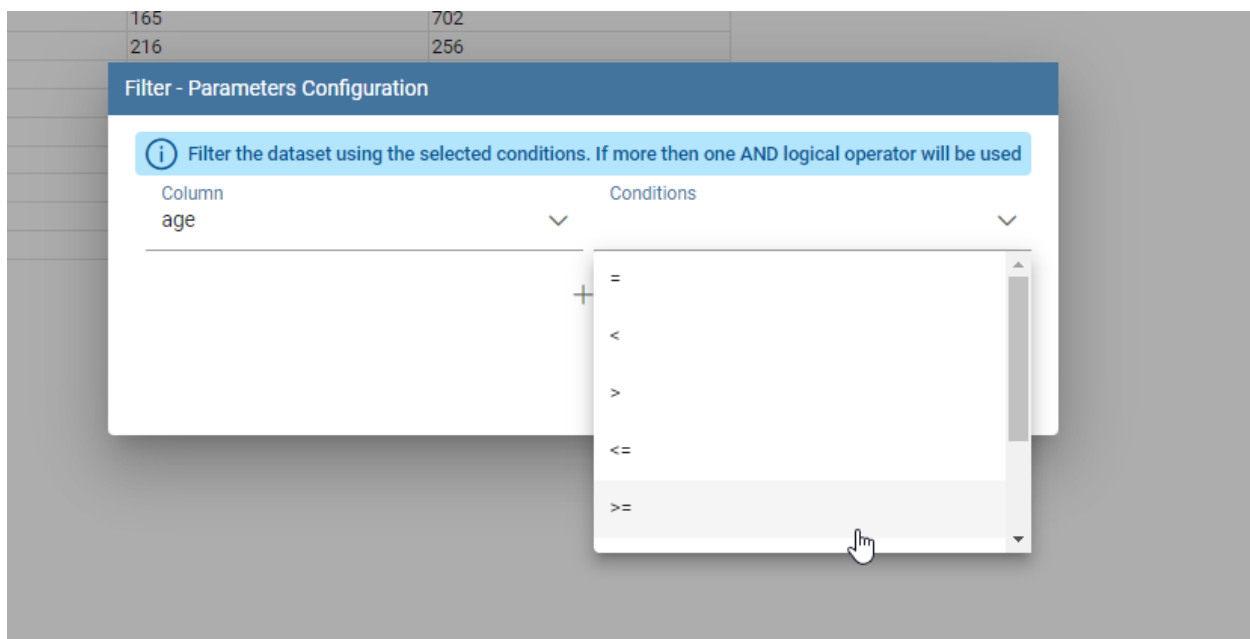
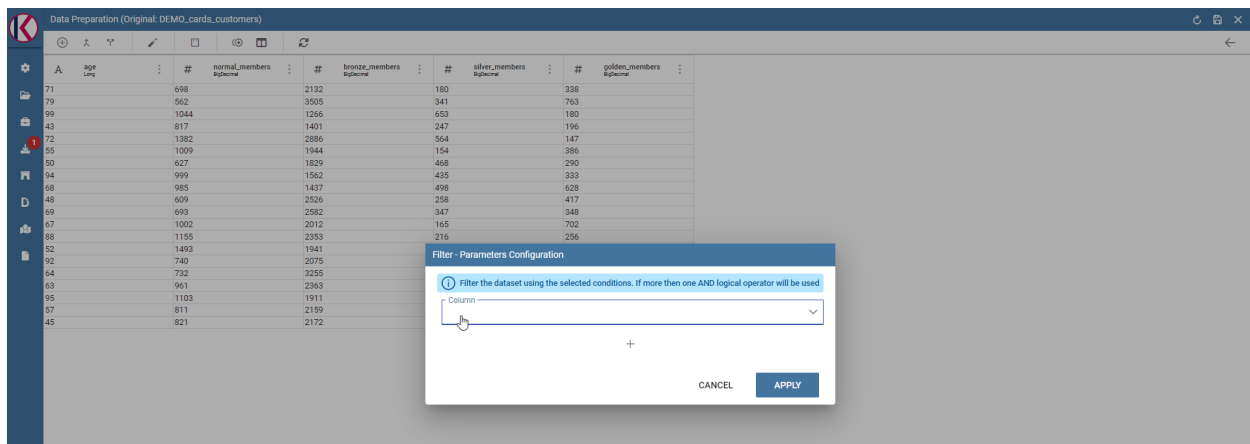


Fig. 1.497: Split columns dialog example.



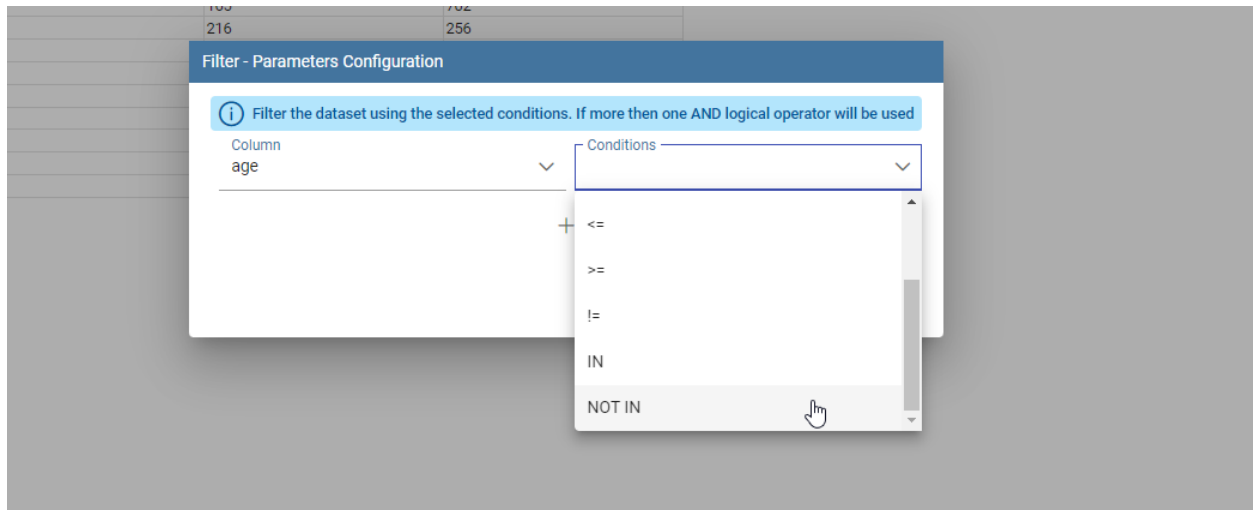


Fig. 1.498: Filter dialog example.

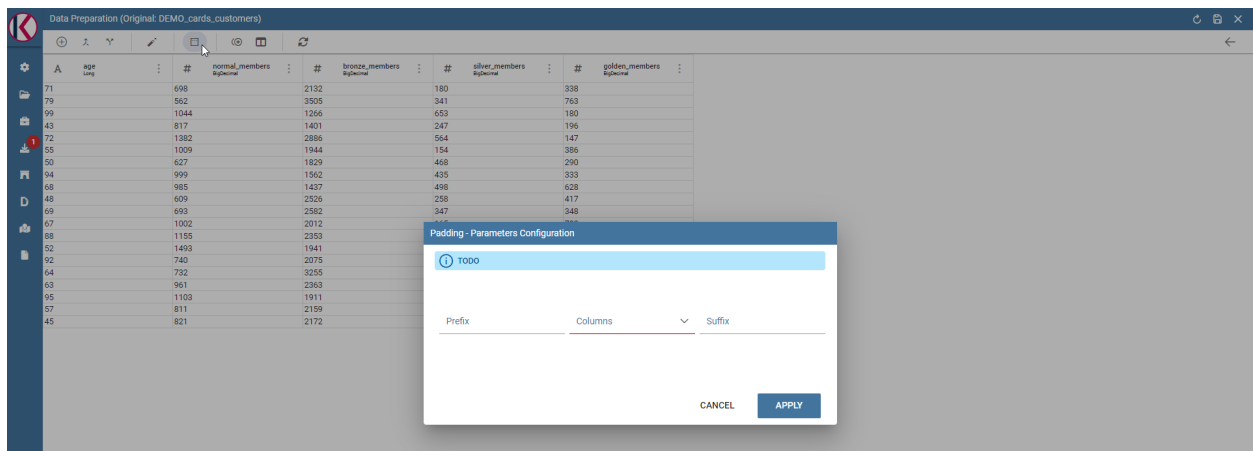


Fig. 1.499: Padding dialog example.

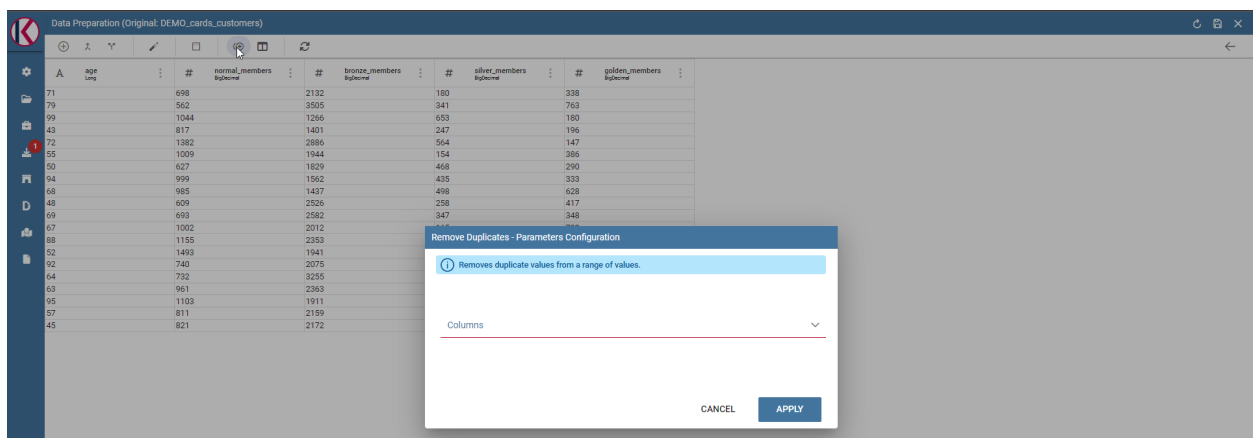


Fig. 1.500: Remove duplicates dialog example.

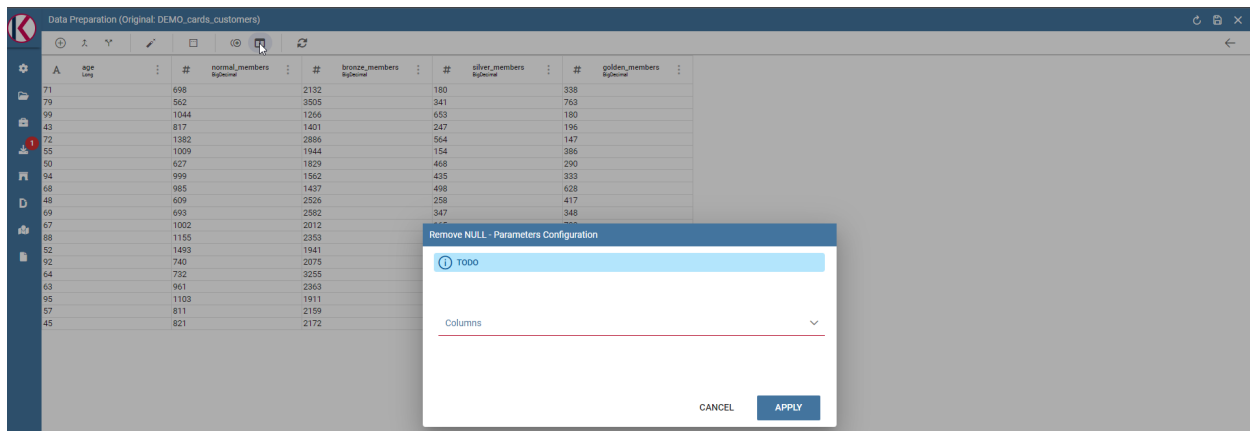


Fig. 1.501: Remove null dialog example.

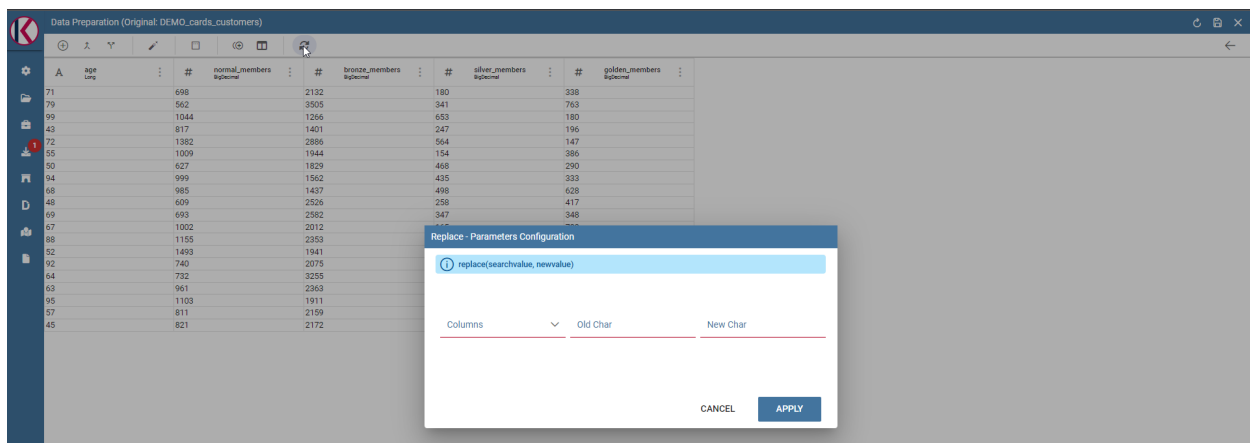
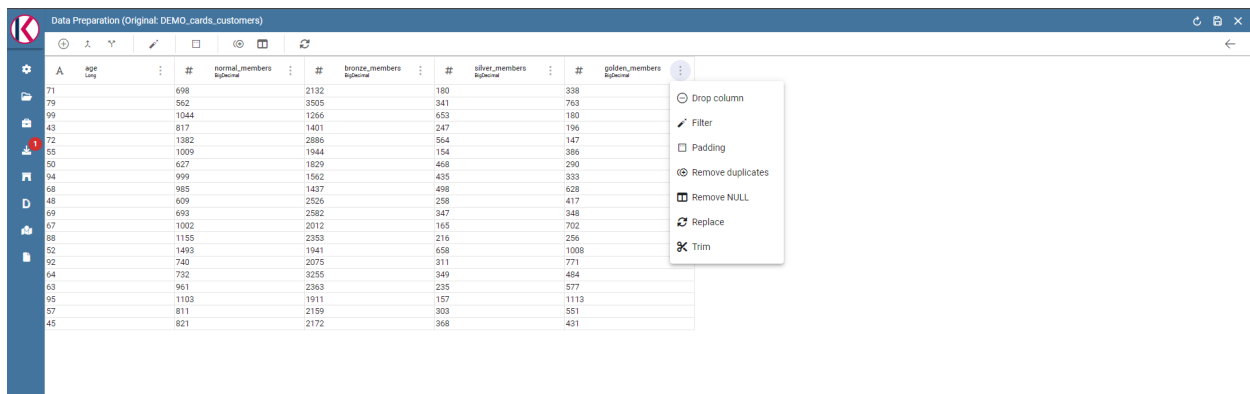


Fig. 1.502: Replace dialog example.



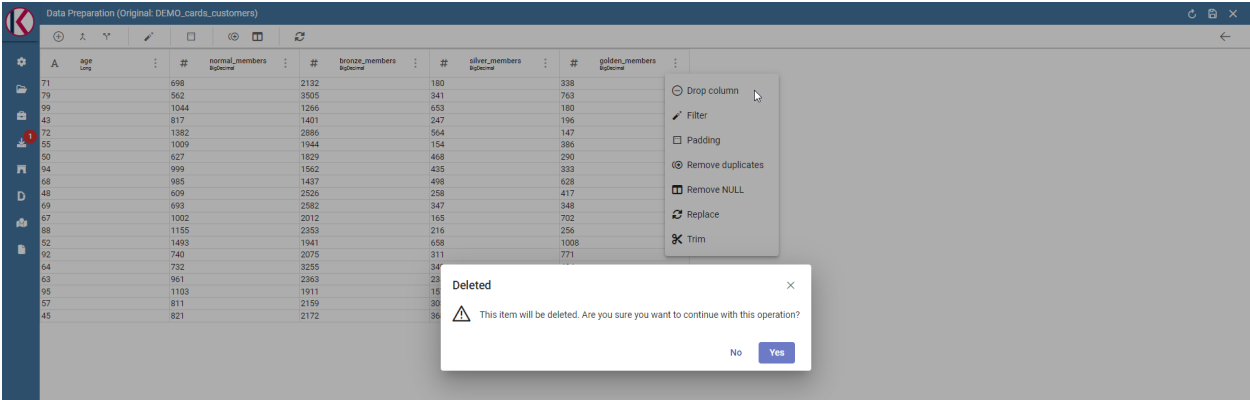


Fig. 1.503: Drop columns dialog warning.

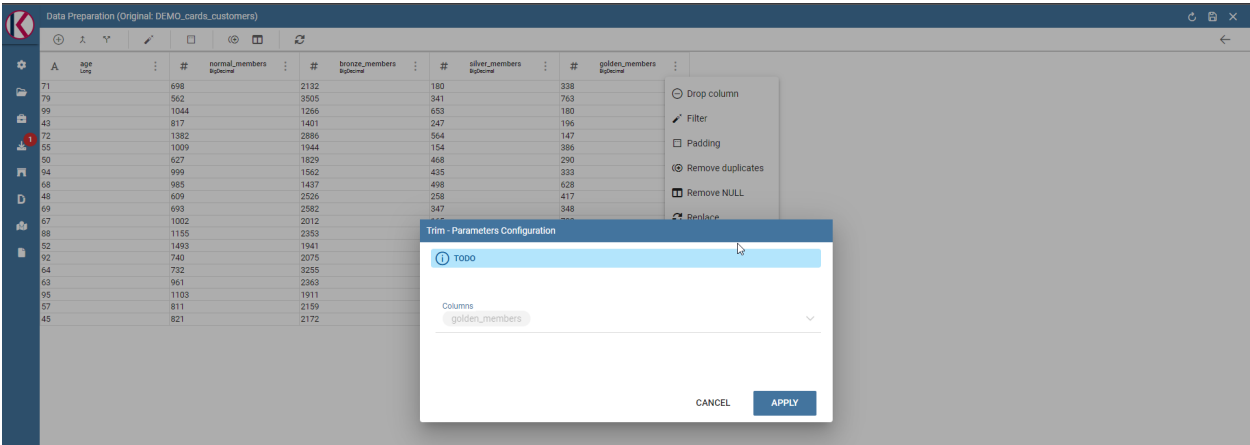


Fig. 1.504: Trim column dialog example.

Avro is a data serialization framework developed within Apache's Hadoop project. It uses JSON for defining data types and protocols, and serializes data in a compact binary format.

Avro relies on schemas. When Avro data is read, the schema used when writing it is always present. This permits each datum to be written with no per-value overheads, making serialization both fast and small. This also facilitates use with dynamic, scripting languages, since data, together with its schema, is fully self-describing.

When Avro data is stored in a file, its schema is stored with it, so that files may be processed later by any program. If the program reading the data expects a different schema this can be easily resolved, since both schemas are present.

Please refer to official documentation for more info: <https://avro.apache.org/>

Avro is used for store Knowage datasets data and schema (with columns metadata) in order to use them as input source for Data Preparation process.

When user open a dataset for data preparation for the first time, an avro file is created. This file is read and then it will be used as data source for data transformations that will be sent to Livy-Spark.

1.15.5 Saving and Using a prepared dataset

Now let's see how to save a prepared dataset. For our documentation example we use two transformations: DROP and then a FILTER on "age" column.

We removed "golden_members" column:

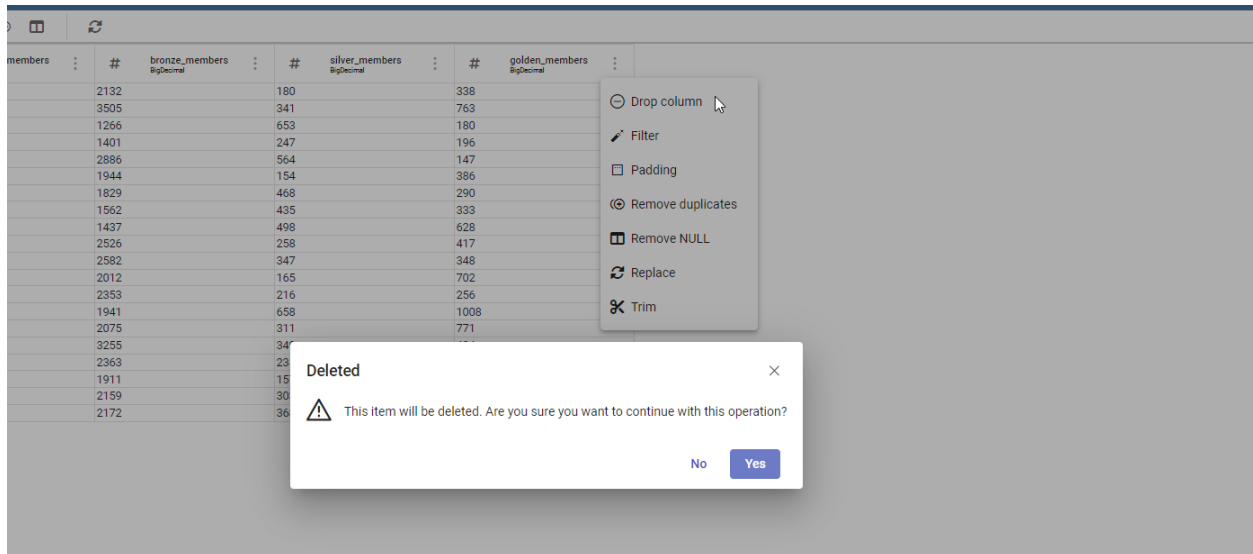


Fig. 1.505: Drop columns dialog example.

And then we filtered by age minor than 60:

The resulting transformations chain can be seen on the right of the page:

As you can see you can remove or preview the last operation (in our case the FILTER transformation).

To see a description of the transformation just click on the eye icon (if present, some transformations don't need it):

You can see how transformation has been configured. Then you can also remove the transformation by clicking on the trash bin:

If you want to save the prepared dataset click on the save icon on the top right of the page:

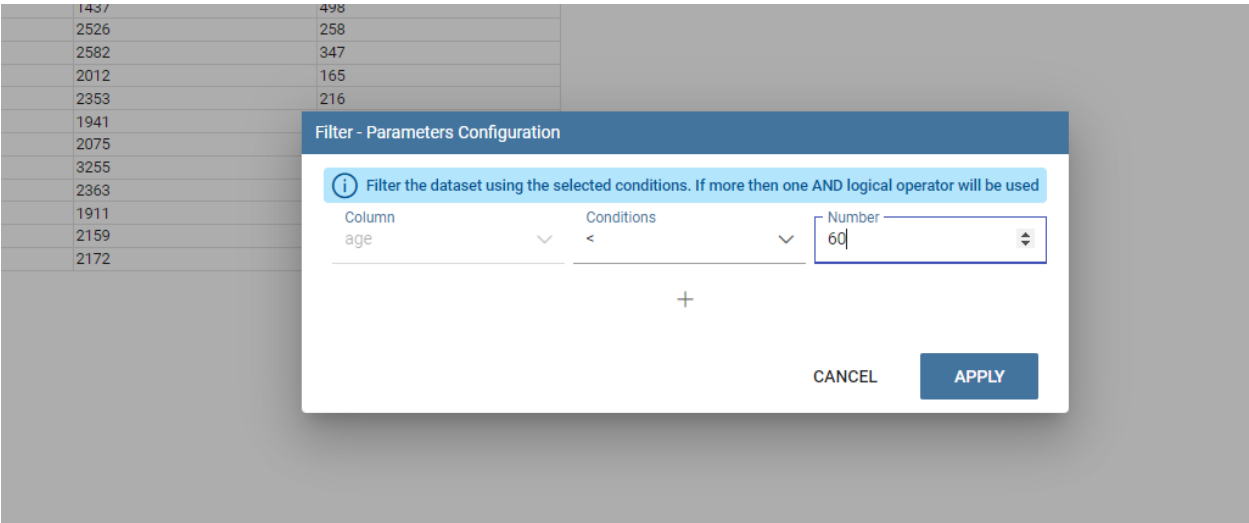


Fig. 1.506: Filter columns dialog example.

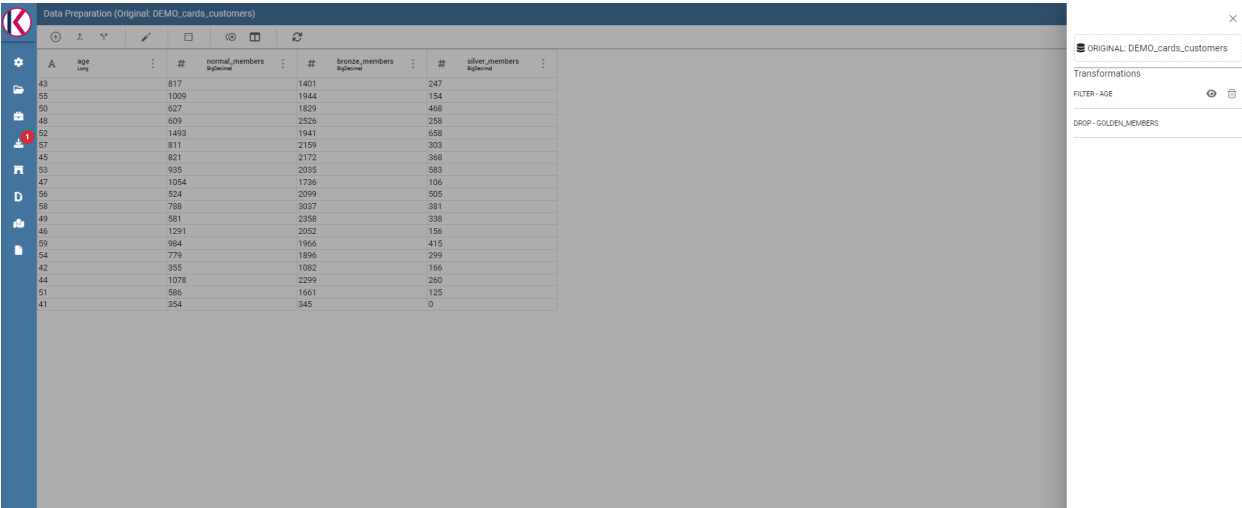


Fig. 1.507: Transformations list is present on the right panel.

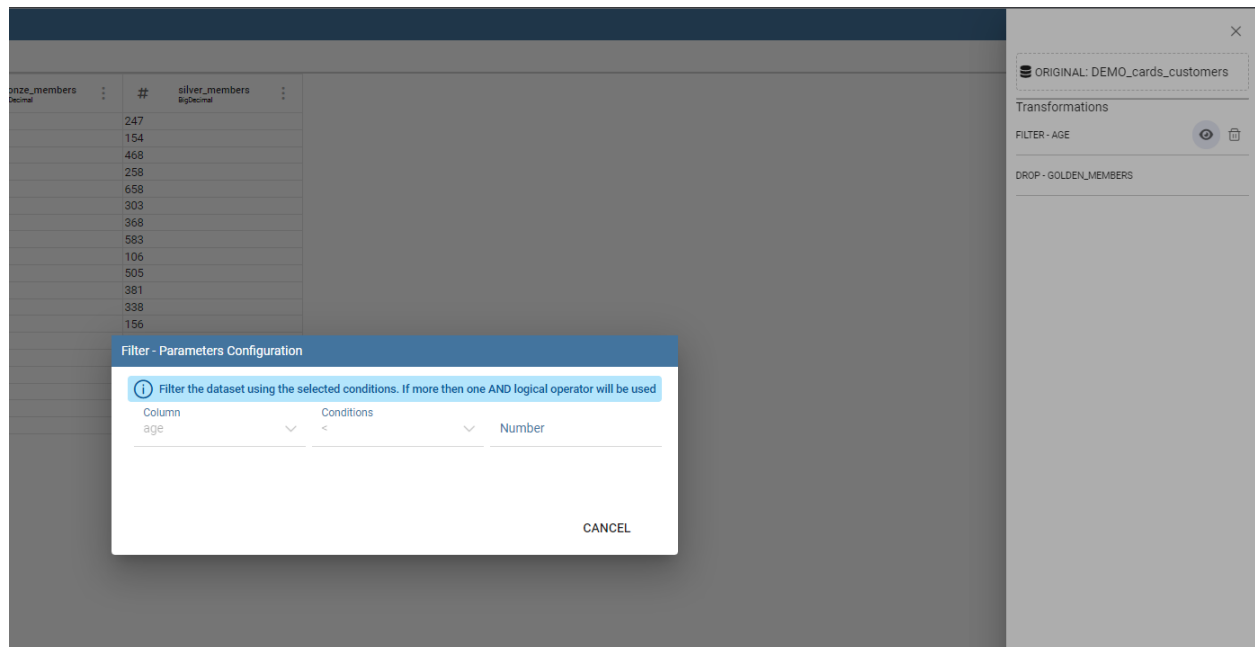


Fig. 1.508: Transformation preview dialog example.

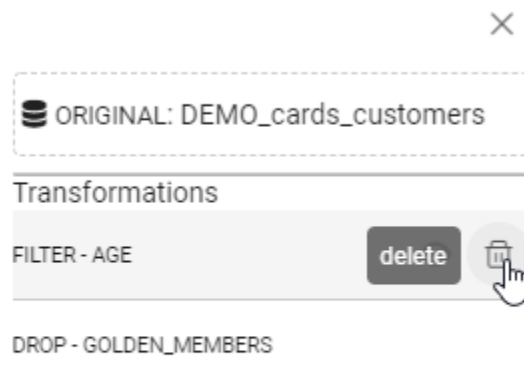


Fig. 1.509: You can delete the last one only.

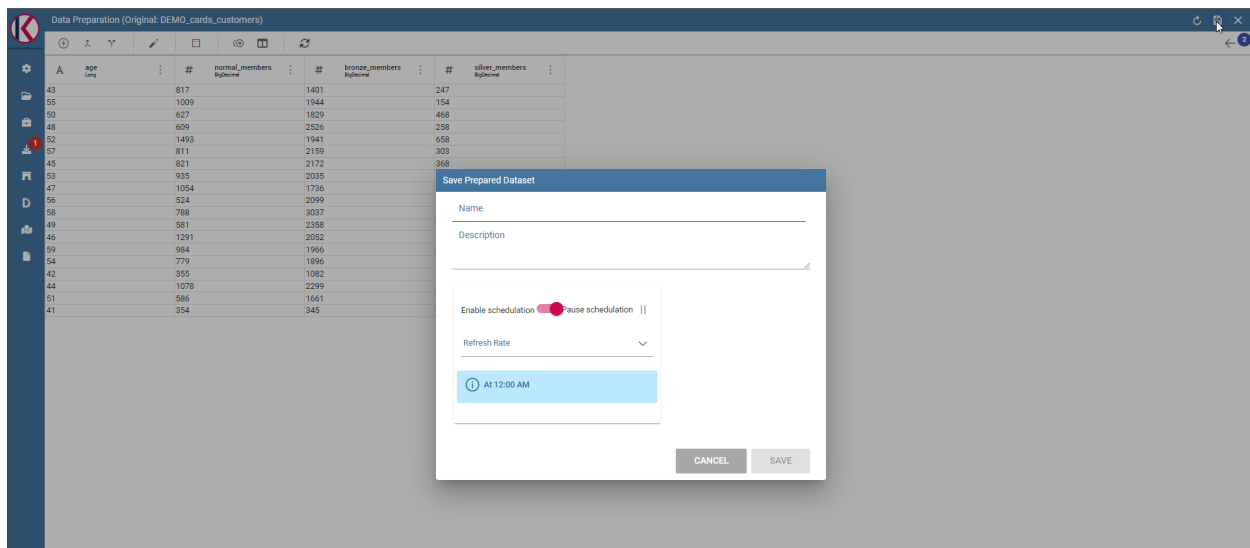


Fig. 1.510: Save panel example.

Here you can choose the name, the description and the scheduling if you want to update the dataset, using the transformation selected, periodically.

After clicking on SAVE button you will see a confirmation message:

After that, waiting for a few moments you will be able to see your data saved on selected datasource clicking on the eye icon on the right into the data preparation section.

If the ingest operation has not finished yet or if there were problems with saving data you will see a warning message telling that the operation is not completed.

You can monitor the process using the monitor section, right click on your saved prepared dataset and clicking on "Monitoring":

You will see a popup with the process results, in case of errors you can download a log file. On the left side you can also change the scheduling of the periodic prepared dataset update.

Now it is possible to see the prepared dataset into the Dataset Management section or into MyData Workspace section, so for example you can use it later for a dashboard.

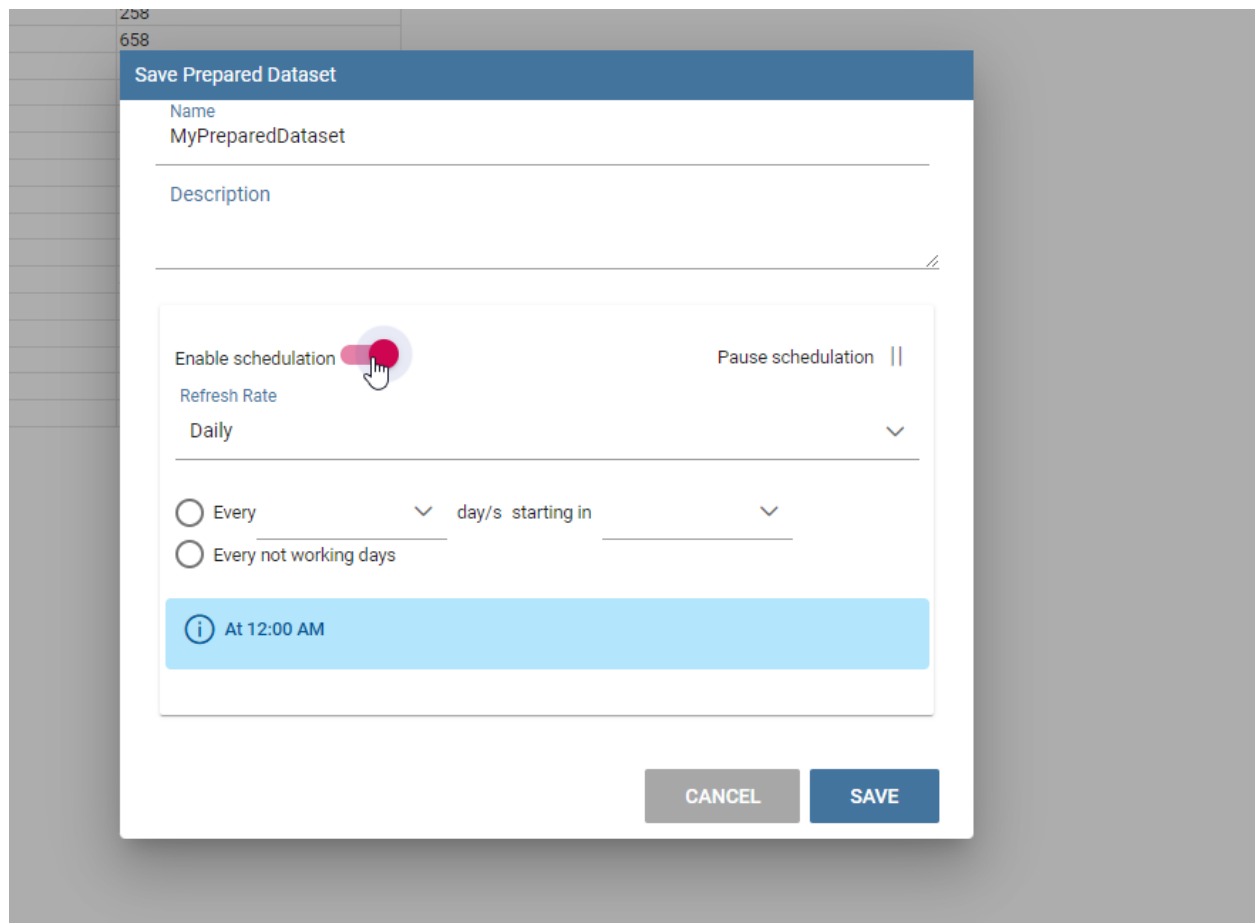


Fig. 1.511: Split columns dialog example.

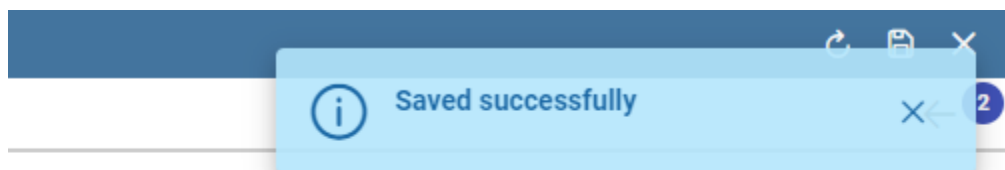
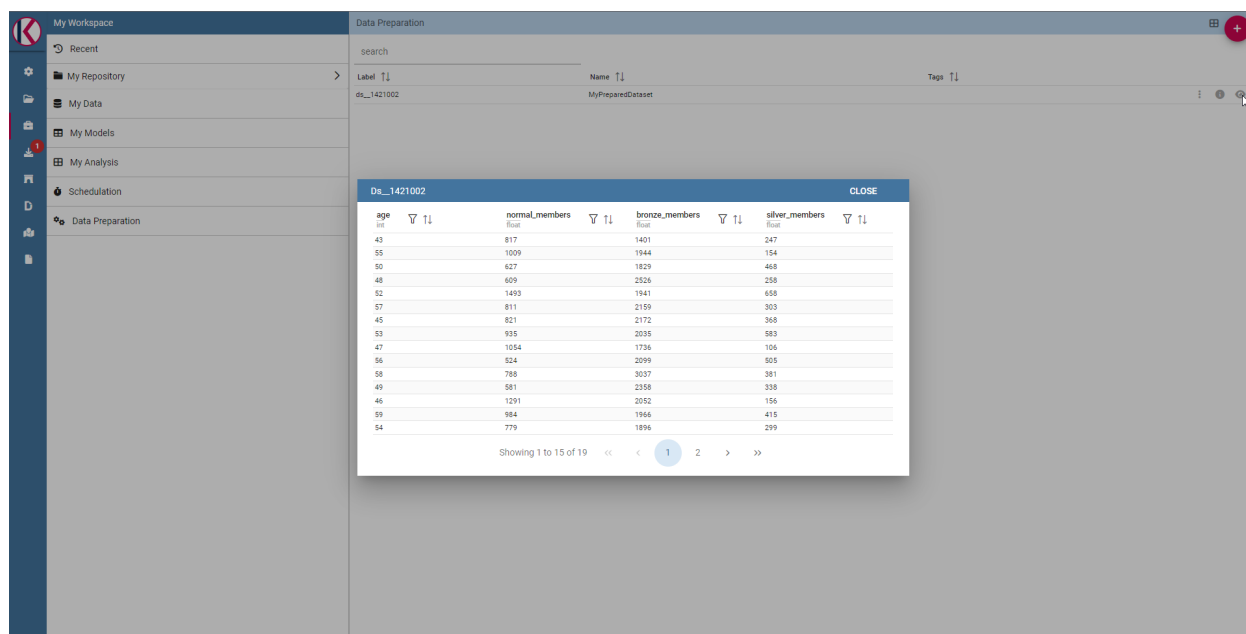


Fig. 1.512: Saving confirmation.



My Workspace

Recent

My Repository

My Data

My Models

My Analysis

Scheduling

Data Preparation

Data Preparation

search

Label []

Name []

Tags []

ds_1421002

MyPreparedDataset

DS_1421002

age

normal_members

bronze_members

silver_members

Showing 1 to 15 of 19

age	normal_members	bronze_members	silver_members
43	817	1401	247
55	1029	1944	154
50	627	1829	468
48	609	2526	258
52	1493	1941	658
57	811	2159	303
45	821	2172	368
53	935	2035	583
47	1054	1736	106
56	524	2099	505
58	788	3037	381
49	581	2358	338
46	1291	2052	156
59	984	1966	415
54	779	1896	299

Fig. 1.513: Prepared data preview panel.

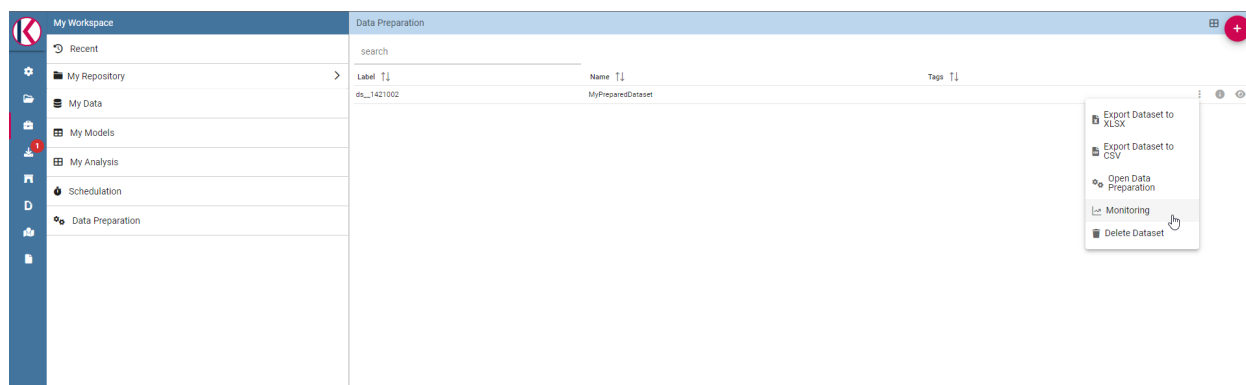


Fig. 1.514: Select monitoring entry.

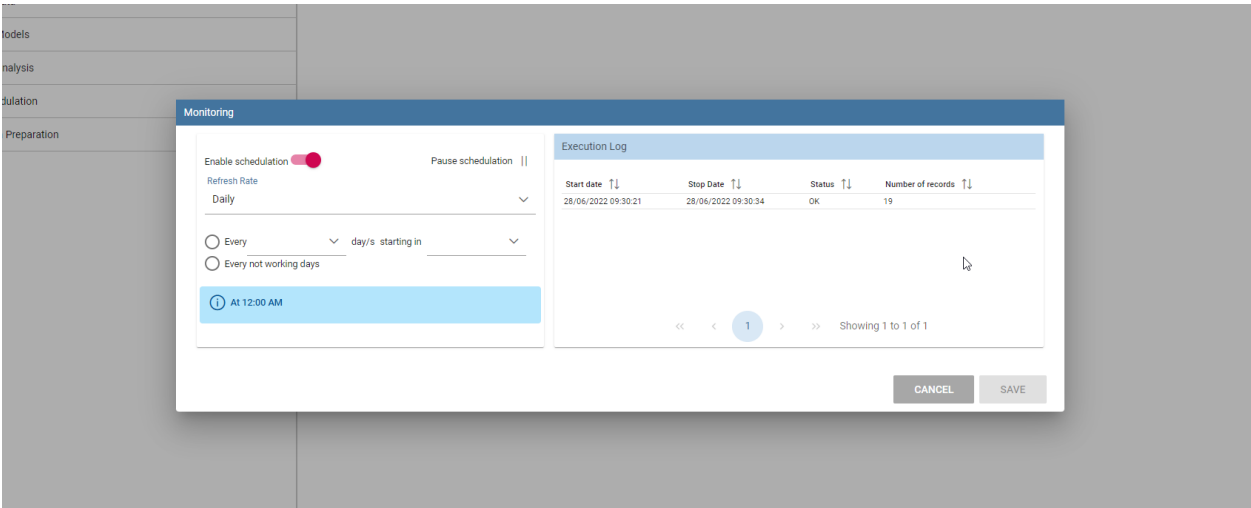


Fig. 1.515: Schedulations and monitoring panel example.

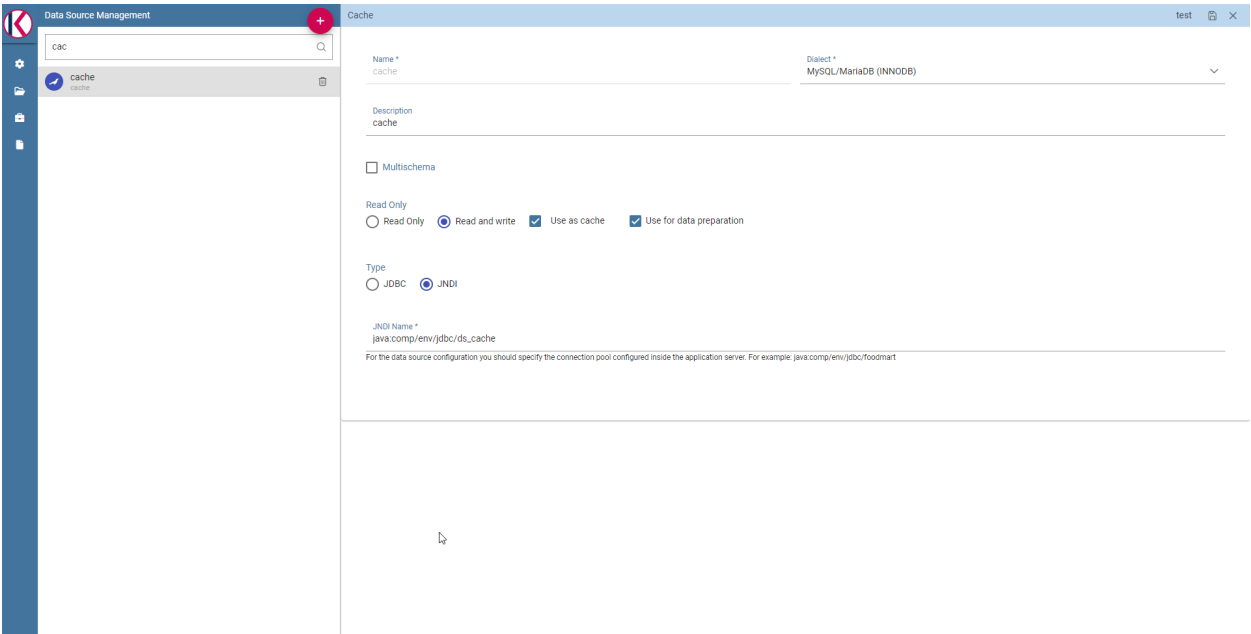


Fig. 1.516: Dataset Management panel example.

2.1 Add new Data Source

To let all the BI tools work properly, you need to configure DB connection. There are two different options available for the configuration, **JNDI** (recommended) and **JDBC**.

2.1.1 Connect to your data

In order to connect to your data, you have to define a new data source connection.

Knowage manages two types of data source connections:

- connections retrieved as JNDI resources, which are managed by the application server Knowage is working on. This allows the application server to optimize data access (e.g. by defining connection pools), reason why they are preferred. By clicking on the following link, you can find information on how creating connection pools in Tomcat : <https://tomcat.apache.org/tomcat-8.0-doc/jndi-datasource-examples-howto.html>
- direct JDBC connections, which are directly managed by Knowage;

Important: How to create connection JNDI on Tomcat

- Create a connection pool on `TOMCAT_HOME/conf/server.xml`
 - Add a ResourceLink on `context.xml`
-

To add a new connection, first add the relative JDBC driver to the folder `TOMCAT_HOME/lib` and restart Knowage. Then, log in as administrator (user: *biadmin*, password: *biadmin* are the default credentials) and select the **Data source** item from the **Data provider** panel in the administrator menu.

By clicking the **Add** button on the top right corner of the left panel, an empty form will be displayed on the right.

Requested information to create a datasource:

- **Name**, identifier of the data source.

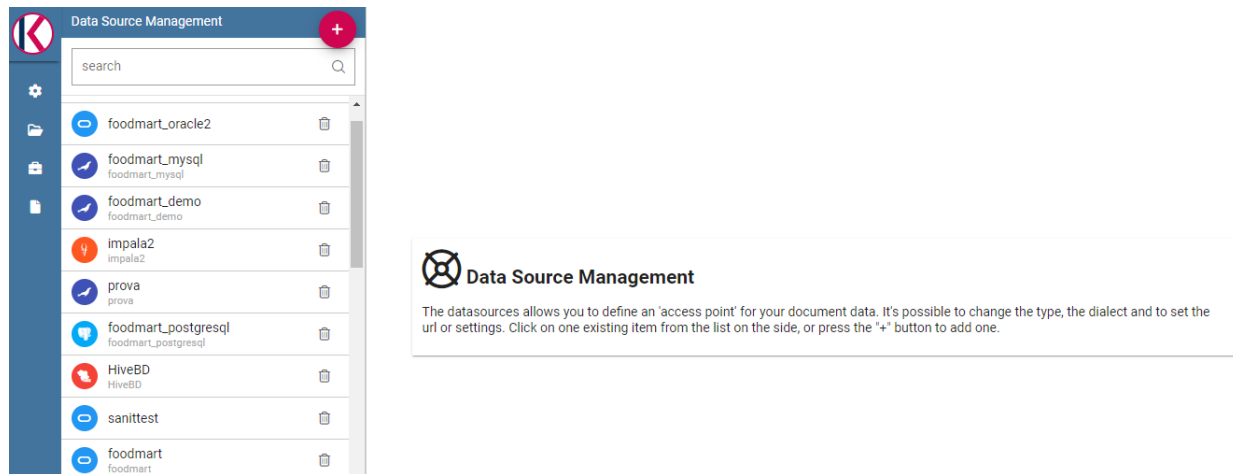


Fig. 2.1: Add a new data source

test

Name *
Dialect *

Description

Read Only
☒ Read Only
☐ Read and write
☐ Use as cache
☐ Use for data preparation

Type
☒ JDBC
☐ JNDI

User
kte_admin

Password (Only If Changed)

URL *

Driver *

Fig. 2.2: Data source details - JDBC.

- **Dialect**, used to access the database. Supported dialects are:

Table 2.1: Certified Data Sources

Certified Data Sources	
Oracle	11, 12
MySQL	5.7, 8
PostgreSQL	8.2, 9.1, 12.3
Maria DB	10.1, 10.2, 10.3
Teradata	15.10.0.7
Vertica	9.0.1-0
Cloudera	5.8.9
Apache Hive 1	1.1.0
Apache Hive 2	2.3.2
Apache Impala	2.6.0
Apache Spark SQL	2.3.0
Apache Cassandra	2.1.3
Mongo DB	3.2.9
Orient DB	3.0.2
Google Big Query	–
Amazon RedShift	(JDBC driver v1)
Azure Synapse	–

- **Read Only/Read and write**, the option *Read Only* is set by default. In case the data source is defined as *Read-and-write*, it will be used by Knowage to write temporary tables.
- **Type, by default set to JDBC**
 - In case of a *JDBC* connection, the fields to fill in are:
 - * **User**, Database username.
 - * **Password**, Database password.
 - * **Driver**, Driver class name. An example for MySQL databases is `com.mysql.jdbc.Driver`.
 - In case of a *JNDI* connection, the fields to fill in are:
 - * **Multischema**, if *checked*, the JNDI resource full name is calculated at runtime by appending a user profile attribute (specified in the *Multischema attribute* field) to the JNDI base name, defined in the `server.xml`.
 - * **JNDI Name**, depends on the application server. For instance, in case of Tomcat 7, the format `java:comp/env/jdbc/<resource_name>` is used. If the data source is multischema, then the format is `java:comp/env/jdbc/<prefix>`.

In case of checking the option *Use as cache*, the datasource will be used as cache in Knowage.

After filling in all the necessary information, test the new data source by clicking on the *Test* button at the top right corner of the page and then *Save* it.

Now you are connected to your data and you can start a new Business Intelligence project with Knowage.

2.1.2 Big Data and NoSQL

In this section we describe how you can connect Knowage to different Big Data data sources.

The screenshot shows a configuration window for a data source. At the top right, there are buttons for 'test', a save icon, and a close icon. The form includes fields for 'Name *' and 'Dialect *'. Below these is a 'Description' field. A checkbox labeled 'Multischema' is present. Under the heading 'Read Only', there are four radio buttons: 'Read Only' (selected), 'Read and write', 'Use as cache', and 'Use for data preparation'. Under the heading 'Type', there are two radio buttons: 'JDBC' and 'JNDI' (selected). At the bottom, there is a 'JNDI Name *' field with a note: 'For the data source configuration you should specify the connection pool configured inside the application server. For example: java:comp/env/jdbc/foodmart'.

Fig. 2.3: Data source details - JNDI.

Important: Enterprise Edition only

Please note that these connections are only available for products KnowageBD and KnowagePM.

2.1.2.1 Hive

Apache Hive is a data warehouse infrastructure built on top of Hadoop for providing data summarization, query and analysis. Apache Hive supports analysis of large datasets stored in Hadoop's HDFS and compatible file systems such as Amazon S3 filesystem. It provides an SQL-like language called HiveQL with schema on read and transparently converts queries to map/reduce, Apache Tez and Spark. All three execution engines can run in Hadoop YARN.

Every distribution of Hadoop provides its JDBC driver for Hive. We suggest you to use or the Apache one or the one specific of your distribution. In general the JDBC driver for Hive is composed by different .jars, and so you should deploy the JDBC driver with all dependencies in your application server. If you are creating a model you should create a new *Data Source Connection* and import the JDBC driver and all the dependencies.

For example suppose you want to connect to Hive using Apache driver you should include these libraries (according to your Hive version) shown in Figure below.

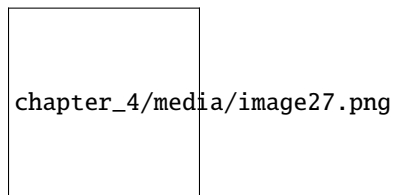


Fig. 2.4: Libraries to include in the apache driver.

If you forget to add one or more libraries, you will likely get a `NoClassDefFoundError` or `ClassNotFoundException`.

The parameters for the Hive connection are:

- **Dialect:** Hive QL;
- **Driver Class:** `org.apache.hive.jdbc.HiveDriver` (if you are not using some specific driver of some distribution. In this case search in the documentation of the distribution);
- **Connection URL:** `jdbc:hive2://<host1>:<port1>,<host2>:<port2>/dbName;sess_var_list?hive_conf_list#hive_var_list`.

Here `<host1>:<port1>,<host2>:<port2>` is a server instance or a comma separated list of server instances to connect to (if dynamic service discovery is enabled). If empty, the embedded server will be used.

A simple example of connection URL is `jdbc:hive2://192.168.0.125:10000`.

2.1.2.2 Spark SQL

Spark SQL reuses the Hive front end and metastore, giving you full compatibility with existing Hive data, queries and UDFs. Simply install it alongside Hive. For the installation of Spark we suggest you to look at the spark website <http://spark.apache.org/>. To create a connection to the Spark SQL Apache Thrift server you should use the same JDBC driver of Hive.

- **Driver Class:** `org.apache.hive.jdbc.HiveDriver` (if you are not using some specific driver of some distro. In this case search in the documentation of the distro);
- **Connection URL:** `jdbc:hive2://<host1>:<port1>,<host2>:<port2>/dbName;sess_var_list?hive_conf_list#hive_var_list`.

Look at the Hive section for the details about parameters. The port in this case is not the port of Hive but the one of Spark SQL thrift server (usually 10001).

2.1.2.3 Impala

Impala (currently an Apache Incubator project) is the open source, analytic MPP database for Apache Hadoop. To create a connection to Impala you should download the jdbc driver from the Cloudera web site and deploy it with all the dependencies, on the application server. The definition of the URL can be different between versions of the driver, please check on the Cloudera web site.

Example parameters for Impala connection are:

- **Dialect:** Hive SQL;
- **Driver Class:** `com.cloudera.impala.jdbc4.Driver`;
- **Connection URL:** `jdbc:impala://dn03:21050/default`.

2.1.2.4 MongoDB

MongoDB is an open-source document database that provides high performance, high availability, and automatic scaling. MongoDB obviates the need for an Object Relational Mapping (ORM) to facilitate development.

MongoDB is different from the other dbs that Knowage can handle, because it does not provide a JDBC driver, but a Java connector. The MongoDB Java driver (at this moment version 3.5.0 is included) is already included inside Knowage so no download is required to add it to the application server.

Example parameters for the connection are:

- **Dialect:** MongoDB;
- **Driver Class:** mongo;
- **Connection URL:** mongodb://localhost:27017/foodmart(please don't include user and password in this URL).

Please keep in mind that the user needs the correct privileges to access to the specified database. For example, on MongoDB you can create a user using this command on the Mongo shell:

Listing 2.1: User creation.

```
1 db.createUser(  
2   {  
3     user: "user",  
4     pwd: "user",  
5     roles: [ { role: "readWrite", db: "foodmart" } ]  
6   }  
7 )
```

Afterwards you must create a role that is able to run functions (this is the way used by Knowage to run the code wrote in the MongoDB's dataset definition) and assign it to the user:

Listing 2.2: Role assignment .

```
1 use admin  
2 db.createRole( { role: "executeFunctions", privileges: [ { resource: { anyResource: true }, actions: [  
3   ↪ "anyAction" ] } ], roles: [ ] } )  
4 use foodmart  
5 db.grantRolesToUser("user", [ { role: "executeFunctions", db: "admin" } ])
```

See also this useful links: - (<https://docs.mongodb.com/manual/tutorial/enable-authentication/>)
- (<https://www.claudiokuenzler.com/blog/555/allow-mongodb-user-execute-command-eval-mongodb-3.x#.W59wiaYzaUI>)

2.1.2.5 Cassandra

Apache Cassandra is an open source distributed database management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. Cassandra offers robust support for clusters spanning multiple datacenters, with asynchronous masterless replication allowing low latency operations for all clients.

There are different ways to connect Knowage to Cassandra.

If you are working with DataStax Enterprise you can use Spark SQL connector and query Cassandra with pseudo standard SQL (https://github.com/datastax/spark-cassandra-connector/blob/master/doc/2_loading.md)

Another solution is to download the JDBC Driver suitable for your Cassandra distribution and query Cassandra using the CQL language. You must deploy the JDBC driver with all dependencies in your application server (copy them into TOMCAT_HOME/lib folder and restart).

Refer to the JDBC driver documentation in order to see how to configure the JDBC connection parameters.

Unless you are using Spark SQL to read from Cassandra, the definition of a business model over Cassandra data using Knowage Meta will be available in the next releases.

2.1.2.6 Google Big Query

Knowage supports Google Big Query datasources trough Simba JDBC Driver: see [official documentation](#).

For example, to create a JDBC connection to a Google Big Query dataset using a service account, you can add the following configuration to `TOMCAT_HOME/conf/server.xml`:

```
<Resource auth="Container" driverClassName="com.simba.googlebigquery.jdbc42.Driver" logAbandoned="true"
  ↪maxActive="20" maxIdle="4"
  ↪maxWait="300" minEvictableIdleTimeMillis="60000" name="jdbc/my-bigquery-ds" removeAbandoned="true"
  ↪removeAbandonedTimeout="3600"
  ↪testOnReturn="true" testWhileIdle="true" timeBetweenEvictionRunsMillis="10000" type="javax.sql.DataSource"
  ↪url="jdbc:bigquery://https://www.googleapis.com/bigquery/v2:443;ProjectId=<<project-id>>;OAuthType=0;
  ↪OAuthServiceAcctEmail=<<service-account-email>>;OAuthPvtKeyPath=<<json-key>>;DefaultDataset=<<default-
  ↪dataset>>;FilterTablesOnDefaultDataset=1;"/>
```

2.1.2.7 Google Cloud Spanner

Knowage supports Google Cloud Spanner datasources via the official open source JDBC driver: see [official documentation](#).

For example, to create a JDBC connection to a Google Cloud Spanner dataset using a service account, you can add the following configuration to `TOMCAT_HOME/conf/server.xml`:

```
<Resource auth="Container" driverClassName="com.google.cloud.spanner.jdbc.JdbcDriver" logAbandoned="true"
  ↪maxActive="20" maxIdle="4"
  ↪maxWait="300" minEvictableIdleTimeMillis="60000" name="jdbc/my-spanner-ds" removeAbandoned="true"
  ↪removeAbandonedTimeout="3600"
  ↪testOnReturn="true" testWhileIdle="true" timeBetweenEvictionRunsMillis="10000" type="javax.sql.DataSource"
  ↪url="jdbc:cloudspanner:/projects/<<project-id>>/instances/<<instance-name>>/databases/<<db-name>>;
  ↪credentials=${catalina.home}/conf/google-cloud-spanner-auth-key.json"/>
```

2.1.2.8 Amazon RedShift

Knowage supports Amazon RedShift datasources through the Official v1 JDBC Driver: see [official reference](#). According to the documentation related to the use of JDBC drivers v1, a RedShift connection configuration can be done exactly like a PostgreSQL configuration. You can test it creating an example db like this one: [official sample testing db](#). To create a JDBC connection to an Amazon RedShift dataset using a RedShift-only connection you can add the following configuration to `TOMCAT_HOME/conf/server.xml`:

```
<Resource auth="Container" driverClassName="com.amazon.redshift.jdbc.Driver" logAbandoned="true" maxActive="10
  ↪maxIdle="1" minEvictableIdleTimeMillis="60000" name="jdbc/redshift" password="password" removeAbandoned=
  ↪"true" removeAbandonedTimeout="3600" testOnReturn="true" testWhileIdle="true" timeBetweenEvictionRunsMillis=
  ↪"10000" type="javax.sql.DataSource" url="jdbc:redshift://examplecluster.abc123xyz789.us-west-2.redshift.
  ↪amazonaws.com:5439/dev" username="user" validationQuery="SELECT 1"/>
```

2.1.2.9 Azure Synapse

Knowage supports connections to Azure Synapse datasources via SQL Server JDBC Driver ([official documentation](#)).

The following example shows how to create a JDBC connection to an Azure Synapse dataset, by adding the following configuration to `TOMCAT_HOME/conf/server.xml`:

```
<Resource auth="Container" driverClassName="com.microsoft.sqlserver.jdbc.SQLServerDriver" logAbandoned="true"
  ↪maxIdle="4" maxTotal="50" maxWait="-1"
  ↪minEvictableIdleTimeMillis="60000" removeAbandoned="true" removeAbandonedTimeout="3600" testOnReturn=
  ↪"true" testWhileIdle="true"
  ↪timeBetweenEvictionRunsMillis="10000" type="javax.sql.DataSource" name="jdbc/synapse" username="<user>
  ↪password="<password>"
  ↪url="jdbc:sqlserver://your-synapse-instance.sql.azuresynapse.net:1433;database=<database>"
  ↪validationQuery="select 1"/>
```

2.2 Scheduler

Knowage scheduler allows to schedule the execution of one or more analytical documents published on the Server. Documents executed by the scheduler can then be distributed along different dispatching channels. In the following we describe how to create an activity, schedule it and dispatch the results.

2.2.1 Create an Activity

In order to define a new scheduled activity the administrator must specify which documents compose the activity and how to execute them. The list of all scheduled activities can be seen selecting **Tools > Scheduler**. To create a new activity click on the “Plus” icon at the top of the page in the left area. In Figure below you can see the main scheduler page and the new activity GUI.

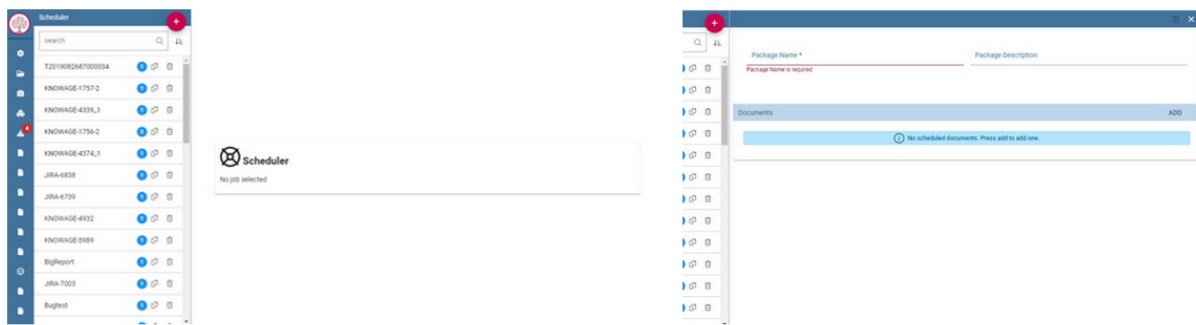


Fig. 2.5: Left: scheduler main page. Right: New activity GUI

Give a name and a description to the new activity. Then select the documents that compose it by clicking on the “Plus” icon and selecting them from the pop up wizard, see Figure below.

Now you need to specify how the scheduler must handle the analytical drivers of each selected document having parameters.

There are two possibilities:

- selecting a value from those available for the specific analytical driver at definition time;
- executing the report one time for each possible values of the analytical driver.

A scheduled activity can be composed by more than one report. Once all desired documents have been added and the management configuration of their parameters has been set up, save the activity by clicking on the save (disk) icon. The new activity is shown in the list and can be modified or deleted using intended specifically icons.

You can manage your activity at any time just clicking on the name of the scheduling item (left side of the window) and all its features will be displayed aside (right half part of the window).

To see and modify the list of the schedules associated to an activity, click on the “ADD” option you find in the “Timing & Output” area in the bottom right side. Similarly, click on the same option to associate schedules to newly created activities.

Timing & Output panel opens (Figure below).

It is composed by two areas: **Timing** and **Output**. The Timing tab let you set all properties of you schedulation. Indeed here you decide its description and timing. A schedulation can be made in a chosen date and time if you choose **Single Execution** as type. Otherwise it can be realized for fixed periods or start on a fixed event.

You can decide it, by choosing the schedulation type.

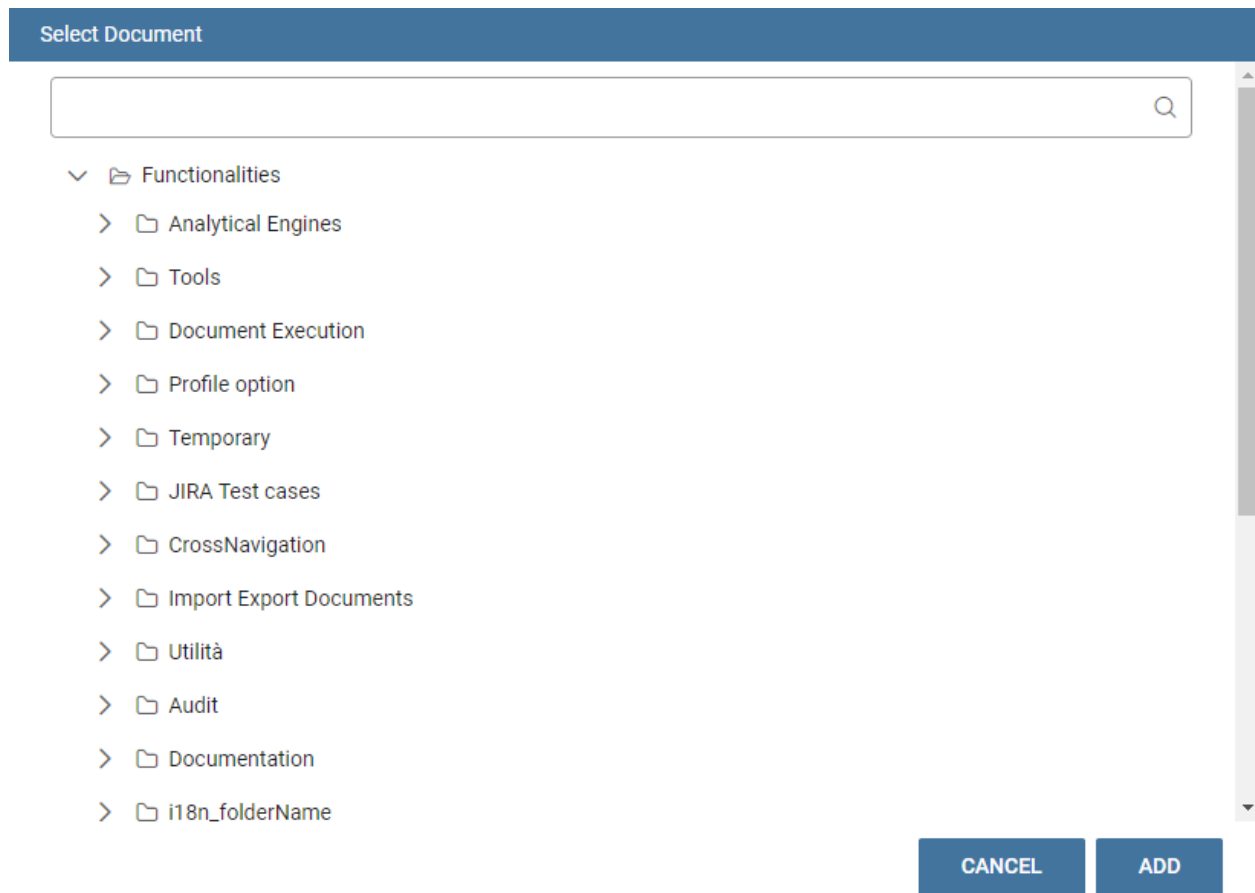


Fig. 2.6: Adding a document to an activity.

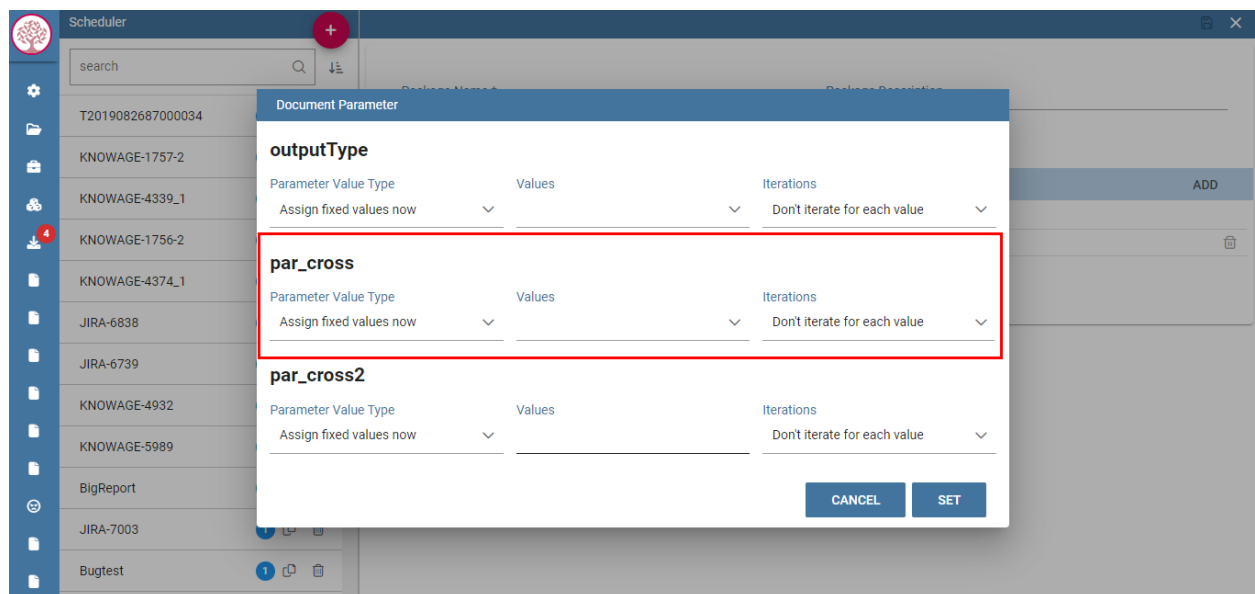


Fig. 2.7: Manage parameters.

The screenshot shows the 'Timing & Output' configuration panel. The 'Timing' tab is active, showing a 'Timing Description *' field and three radio buttons for execution types: 'Single Execution' (selected), 'Periodic Execution', and 'Event Execution'. Below this is a 'Time Window' section with 'Start Date' (1/3/2023) and 'Start Time' (11:56) fields. At the bottom right are 'CANCEL' and 'SAVE' buttons.

Fig. 2.8: Timing & Output panel.

Available schedulation type are:

- Single Execution;
- Periodic Execution;
- Event Execution.

A **Single Execution** is a fixed execution for a date and a time which happens only once. A **Periodic Execution** repeats your schedulation periodically according to your settings. The **Event Execution** starts the scheduling when an event happens.

You can choose among these event types:

- REST WS service,
- Context Broker message,
- Dataset.

If you choose **Dataset**, you have to select a right structured dataset. It has to give as results only true or false. Then set the frequency in seconds. This is the frequency the dataset will be verified. For example if you set it on 10 seconds it means that each 10 seconds the dataset is executed. If the result of its execution is true, the schedulation is triggered otherwise it isn't.

Once you are done, switch to the **Output** tab.

Here you can find the dispatch configurations, that can be different for all the documents that compose the scheduled activity. All documents that compose the activity have their own dispatch configuration and the same document can be distributed along multiple dispatch channels. You can switch among the documents included in your activity by clicking on their name in the upper toolbar. There are many different possible dispatch channels that can be used to distribute the results of the execution of a scheduled activity:

- Save as snapshot,
- Save as file,
- Save as document,
- Send to Java class,
- Send mail

In the following sections we explain them in detail.

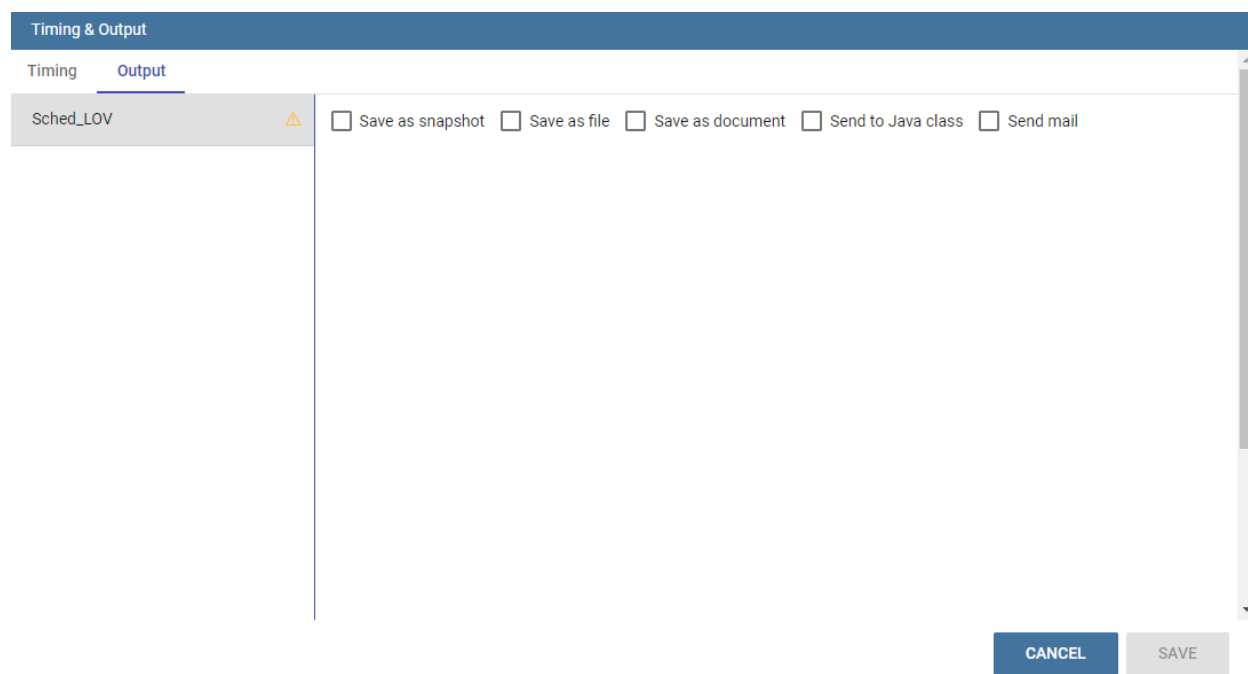


Fig. 2.9: Output panel.

2.2.1.1 Save as snapshot

The executed document can be saved as snapshots in cyclic buffers of configurable size. For example, it is possible to store in the buffer the last 12 snapshots (the **History Length** field) of one report, scheduled to be executed one per month, in order to have a one-year long history.

The list of all snapshots contained in the buffer can be accessed from the **Show scheduled executions** contained in the **Shortcuts** menu. You can find it in the document toolbar at the top right corner. Each snapshot can be opened or deleted from this panel. These steps are shown in the following figure. A snapshot contains data queried from the database at the moment of its execution performed by the scheduler.

2.2.1.2 Save as file

The executed document can be saved as file on the filesystem in the path `/knowage-<version>/resources` (if no destination folder is specified). Otherwise, you can create the relative path of this subfolder by writing your subfolder name. For instance, if you write “MyFirstScheduler” as file name and “Schedulation” as destination folder, after the schedulation execution a subfolder Schedulation containing the file “MyFirstScheduler” is created in `/knowage-<version>/resources`. If the subfolder Schedulation already exist your file is added to this subfolder. You can have a look at the form in Figure below.

If you prefer to generate a .zip file containing the scheduled documents, you can check the dedicated mark.

2.2.1.3 Save as document

The executed document can be saved as an **Ad hoc reporting** document in the Knowage functionality tree. The document execution will be saved in the specified folder and will be visible to all yous that can access that particular folder. For those documents whose execution is iterated over a parameter value, it is also possible to use the value of the parameter to decide to which folder the document shall be dispatched. To do so, define a mapping dataset composed of two columns:

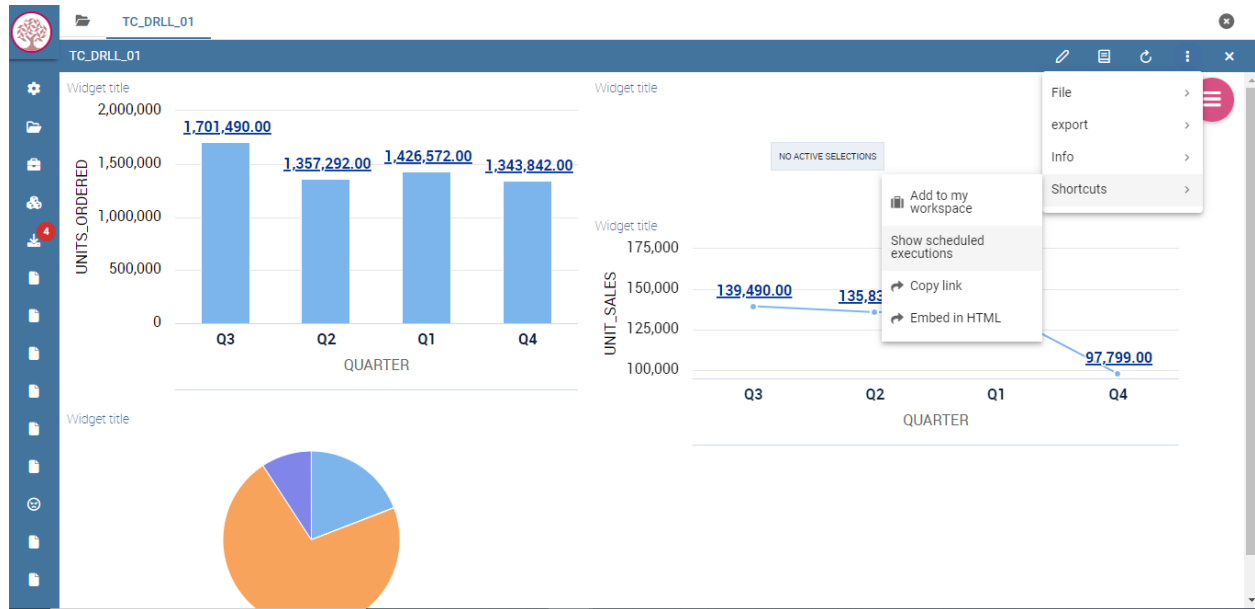


Fig. 2.10: Steps to open saved snapshots

The screenshot shows the 'Timing & Output' dialog box. The 'Output' tab is selected. Under 'Sched_LOV', there are checkboxes for 'Save as snapshot', 'Save as file' (checked), 'Save as document', 'Send to Java class', and 'Send mail'. Below these, there is a 'Save as file' section with a warning icon. It contains a 'File Name *' field, a 'Destination Folder' field, and a 'Save as Zip file' checkbox. At the bottom right, there are 'CANCEL' and 'SAVE' buttons.

Fig. 2.11: Save as File form.

- the first containing a specific parameter value;
- the second containing the label of the folder where the document shall be dispatched when the document is executed with the corresponding parameter value.

Once you have defined the mapping dataset, you can use it in the configuration settings of the document dispatcher. Like in the previous case, the scheduler will execute the report one time for each possible value of the parameter. This time, however, execution results will be dispatched in different folders, according to the mapping defined in the dataset.

2.2.1.4 Send to Java class

The executed document can be sent to a Java class implementing a custom dispatch logic. The custom class must extend the abstract class `JavaClassDestination` that implements the method `execute`. This method is called by the scheduler after document execution. Below an example of Java class.

Listing 2.3: Java Class Code Example.

```

1  package it.eng.spagobi.tools;
2  import it.eng.spagobi.analiticalmodel.document.bo.BIObjct
3  public abstract class JavaClassDestination
4  implements IJavaClassDestination {
5      BIObjct biObj=null;
6      byte[] documentByte=null;
7      public abstract void execute();
8      public byte[] getDocumentByte() {
9          return documentByte;
10     } public void setDocumentByte(byte[] documentByte) {
11         this.documentByte = documentByte;
12     }
13     public BIObjct getBiObj() {
14         return biObj;
15     }
16     public void setBiObj(BIObjct biObj) {
17         this.biObj = biObj;
18     }
19 }
```

The method `getDocumentByte` can be used to get the executed document, while the method `getBiObj` can be used to get all metadata related to the executed document. The following code snippet shows an example of a possible extension of class `JavaClassDestination`.

Listing 2.4: JavaClassDestination example.

```

1  public class FileDestination extends JavaClassDestination {
2      public static final String OUTPUT_FILE_DIR = "D:\\ScheduledRpts\\";
3      public static final String OUTPUT_FILE_NAME = "output.dat";
4      private static transient Logger logger = Logger.getLogger(FileDestination.class);
5      public void execute() {
6          File outputDir;
7          File outputFile;
8          OutputStream out;
9          byte[] content = this.getDocumentByte();
10         String outputFileName;
11         logger.debug("IN");
12         outputFile = null;
13         out = null;
14         try {
15             outputFileName = getFileName();
16             logger.debug("Output dir [" + OUTPUT_FILE_DIR + "]");
17             logger.debug("Output filename [" + outputFileName + "]");
18             outputDir = new File(OUTPUT_FILE_DIR);
```

(continues on next page)

(continued from previous page)

```

19 outputFile = new File(outputDir, outputFileName);
20 if(!outputDir.exists()) {
21     logger.debug("Creating output dir [" + OUTPUT_FILE_DIR + "] ...");
22     if(outputDir.mkdirs()) {
23         logger.debug("Output dir [" + OUTPUT_FILE_DIR + "] succesfully created");
24     } else {
25         throw new SpagoBIRuntimeException( "Impossible to create outputd dir
26 [" + OUTPUT_FILE_DIR + "]" );
27     }
28 } else {
29     if(!outputDir.isDirectory()) {
30         throw new SpagoBIRuntimeException( "Outputd dir [" + OUTPUT_FILE_DIR + "]
31 is not a valid directory");
32     }
33 }
34 try {
35     out = new BufferedOutputStream( new FileOutputStream(outputFile));
36 } catch (FileNotFoundException e) {
37     throw new SpagoBIRuntimeException(
38 "Impossible to open a byte stream to file
39 [" + outputFile.getName() + "]", e);
40 } try {
41     out.write(content);
42 } catch (IOException e) {
43     throw new SpagoBIRuntimeException( "Impossible to write on file
44 [" + outputFile.getName() + "]", e);
45 }
46 } catch(Throwable t) {
47     throw new SpagoBIRuntimeException( "An unexpected error occurs while saving
48 document" + " to file [" + outputFile.getName() + "]", t);
49 } finally {
50     if(out != null) {
51         try {
52             out.flush(); out.close();
53         } catch (IOException e) {
54             throw new SpagoBIRuntimeException( "Impossible to properly close file
55 [" + outputFile.getName() + "]", e);
56         }
57     }
58     logger.debug("OUT");
59 }
60 }
61 private String getFileName() {
62     String filename = "";
63     BIObject analyticalDoc;
64     List analyticalDrivers;
65     BIObjectParameter analyticalDriver;
66     String extension = "pdf";
67     analyticalDoc = getBiObj();
68     analyticalDrivers = analyticalDoc.getBiObjectParameters();
69     for(int i = 0; i < analyticalDrivers.size(); i++) {
70         analyticalDriver = (BIObjectParameter)analyticalDrivers.get(i);
71         String parameterUrlName = analyticalDriver.getParameterUrlName();
72         List values = analyticalDriver.getParameterValues();
73         if(!parameterUrlName.equalsIgnoreCase("outputType")){
74             filename += values.get(0);
75         } else {
76             extension = "" + values.get(0);
77         }
78     }
79     filename = filename.replaceAll("[^a-zA-Z0-9]", "_");
80     filename += "." + extension;
81     return filename;
82 }
83 }

```

The class `FileDestination` copies the executed documents to the local filesystem in a folder named `D:\textbackslashScheduledRpts`. The name of the report file is generated concatenating all the parameter values used by the scheduler during execution. Once implemented and properly compiled, the Java class must be exposed to the classpath of Knowage web application. For example, you can pack the compiled class into a .jar file, copy it into the lib folder of Knowage web application and restart the server. As a last step, it is necessary to assign the fully qualified name of the new class, e.g., `it.eng.spagobi.tools.DestinationFile.`, to the configuration property `classpath`.

2.2.1.5 Send mail

Important: Enterprise Edition only

This feature is available only with KnowageER and KnowageSI, submodules of Knowage Enterprise Edition

The executed document can be sent to one or more mail recipients. The list of mail addresses to be used to forward the executed document can be defined in three different ways:

- statically;
- dynamically, using a mapping dataset;
- dynamically, using a script.

In Figure below you can have a look at the mail form. In the following we will focus on each typology, clicking on the info icon you get detailed information.

The screenshot shows the 'Timing & Output' configuration window. The 'Output' tab is selected. Under 'Sched_LOV', there are checkboxes for 'Save as snapshot', 'Save as file', 'Save as document', 'Send to Java class', and 'Send mail'. The 'Send mail' checkbox is checked. Below this, there is a section for 'Send mail' with a sub-header 'Fixed list of recipients' which is also checked. A text field labeled 'Mail To *' is present with a character count of '0 / 1000'. Other options include 'Use a DataSet as recipients' list', 'Use an expression', 'Send unique mail for all scheduled document (use this tab mail settings)', 'Send zipped file?', and 'Include report name (and timestamp) in mail subject'. At the bottom right, there are 'CANCEL' and 'SAVE' buttons.

Fig. 2.12: Sending mail form.

2.2.1.5.1 Static list

If you want to choose a static list, check the option **Fixed list of recipients** and fill the configuration property **Mail to** with the list of desired mail addresses separated by a comma. An mail for each executed document will be sent to all

the mail addresses contained in the list.

2.2.1.5.2 Dynamic list with mapping dataset

In this case, you have to define a two-column dataset:

- the first containing a specific parameter value;
- the second containing each mail address the executed document should be dispatched to.

You can see an example of dataset in the following Figure.

	parameter_value	mail_address
1	President	namesurname@gmail.com
2	VP Country Manager	name1surname1@gmail.com
3	VP Information System	name1surname1@gmail.com
4	VP Human Resources	name1surname1@gmail.com
5	VP France	name1surname1@gmail.com
6	HQ Information System	name2surname2@gmail.com
7	HQ Marketing	name2surname2@gmail.com
8	HQ Human Resources	name2surname2@gmail.com
9	HQ Finance and Account	name2surname2@gmail.com

Example of mapping dataset for dynamic distribution list

Basically, when the parameter has a given value, the document will be sent to the corresponding email address. Once you have defined the mapping dataset, you can use it in the configuration settings of the document dispatcher. With this configuration, the scheduler will execute the report one time for each possible value of the parameter **Position**, then dispatching the results to different recipients. Specifically, all execution results passing a value of the **Position** parameter to the report starting with VP will be sent to `name1surname1@gmail.com`, the ones starting with HQ will be sent to `name2surname2@gmail.com` and the ones starting with President will be sent to `namesurname@gmail.com`.

2.2.1.5.3 Dynamic List with script

Check the option **Use an expression** and assign a value to the configuration property **Expression** with a parameter-dependent expression like the following:

```
$P{dealer}@eng.it
```

Here dealer is a document parameter label (`$P{dealer}` will be replaced by the parameter value of the scheduled execution).

2.2.2 Scheduling panel

To conclude our overview on the scheduler features, save your settings and go back to the main scheduler page.

Here you can select one of the available scheduled activities to explore details.

Here you find the following information:

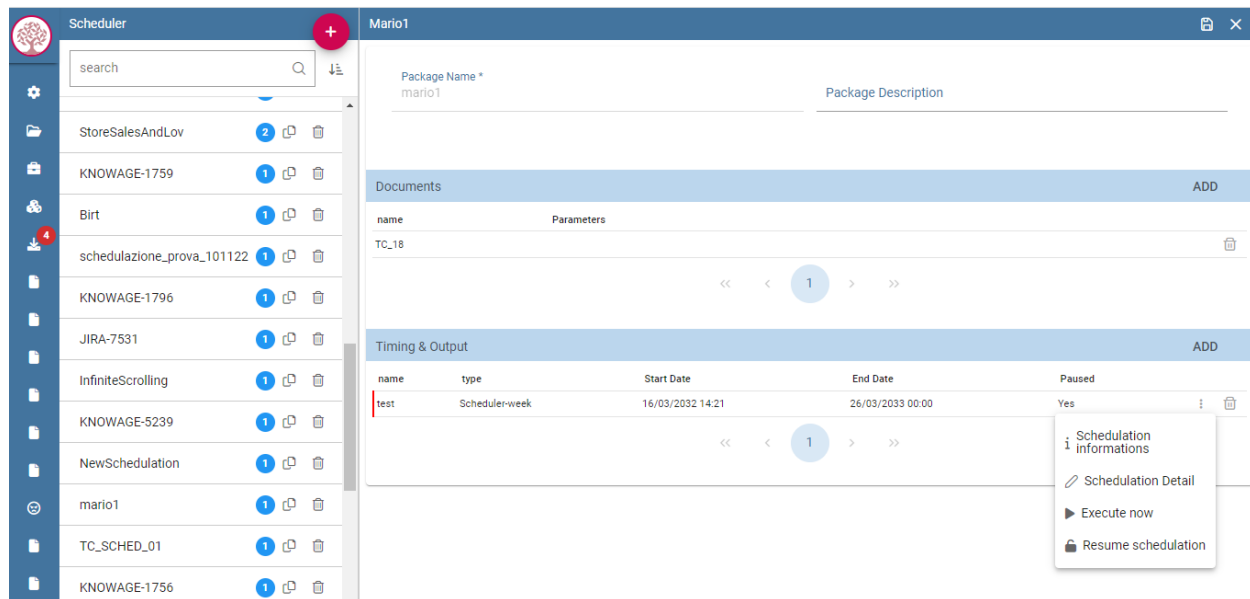


Fig. 2.13: Exploring the detail of a scheduled activity in Timing & Output panel (three dots menu).

- **Scheduling informations**, it give some extra information about your scheduling concerning sending emails
- **Scheduling detail**, it opens the scheduling configuration and let you change them.
- **Execute now**, by clicking it you immediately start the execution of your scheduling.
- **Pause scheduling**, it lets you pause your scheduling.
- **Resume scheduling**, it appears after having paused a scheduling, it enables you to resume it.

In order to delete a scheduling you can use delete (recycle bin) icon, on the right side of a scheduling.

2.2.3 Scheduler Monitor

You can monitor the whole scheduling agenda by entering the **Scheduler Monitor** item from the Knowage Menu. This feature allows you to check which schedulations are active in a certain future time interval and, eventually, to be redirected to the scheduling area in order to modify the selected scheduling.

2.3 Server manager

Important: Enterprise Edition only

Server Manager functionalities are only available with Knowage Enterprise Edition

All the management functionalities can be found under **Server Manager** of the Knowage main menu.

SCHEDULATION INFORMATIONS

TESTBIRT

Mail to:

Attached zip name:

Mail Subject:

Contained File Name:

Mail Text:

CLOSE

Fig. 2.14: Scheduling information pop up example

S Scheduling Agenda

Start Date:

Start Time:

13

:

01

End Date:

End Time:

13

:

01

Package

Document

SEARCH

Scheduling Agenda
Scheduling agenda allows you to search and display scheduled agenda.

Fig. 2.15: Scheduling Agenda tab

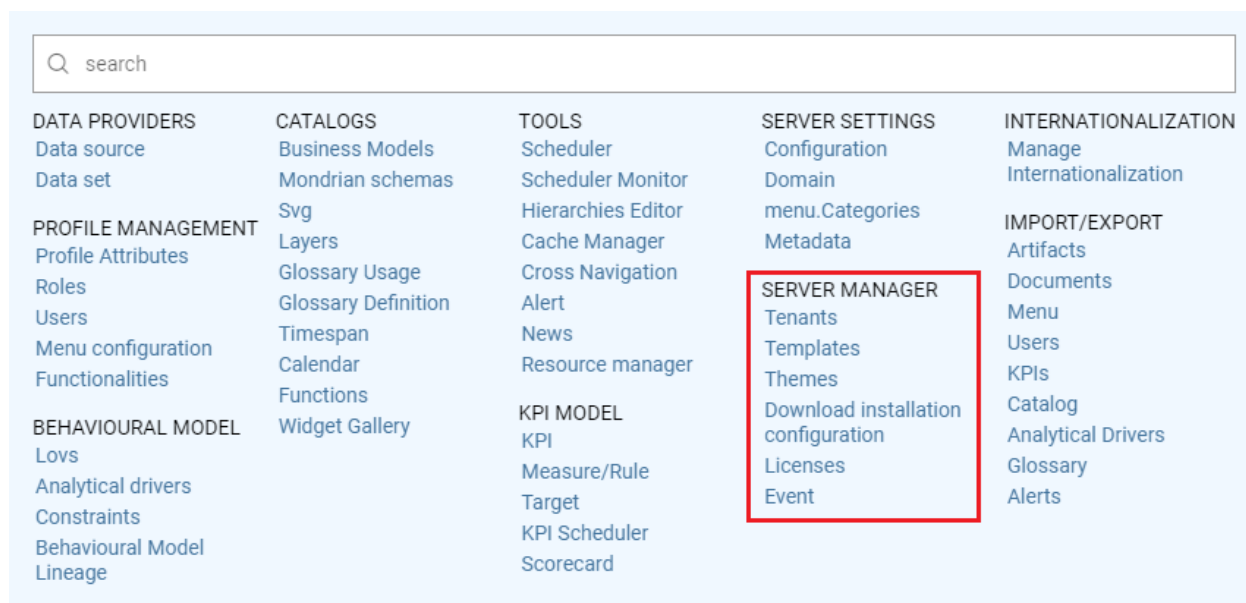


Fig. 2.16: Server Manager Functionalities

2.3.1 Tenants

This feature allows you to create or manage tenants. A single Knowage instance can have one or more tenants. In case of a multi-tenant environment, each tenant owns and manages its own users, documents, configuration and parameters, which are completely independent from those owned by other tenants. The *Tenants* functionality is only available for the Knowage Enterprise Reporting (ER) license and users must have the superadmin role.

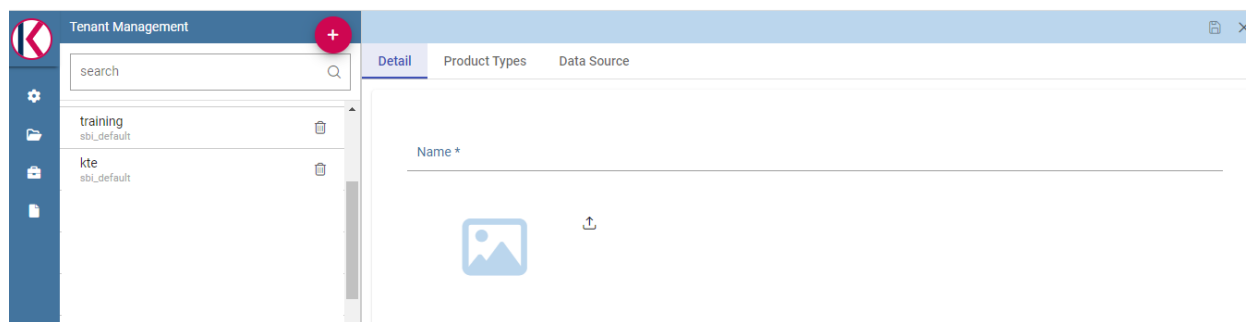


Fig. 2.17: Tenants Management window.

The left side of the image above, presents the list of existing tenants with the availability of a **Search** box, to help users to browse through the existing tenants. The *Plus* icon can be used to create a new tenant.

In a *single-tenant* environment the *admin* role matches with the *superadmin*. In a *multi-tenants* environment only *one* user has the *superadmin* role for each tenant, while there can be more than one user with the *admin* role. Furthermore, the superadmin is the only one who can set up JNDI datasources and access the cache configuration.

2.3.2 Templates

Each Knowage document is associated to a *template*. The template defines the standard layout of a document, including specific information on its appearance and the way contents should be displayed. Templates can be encoded by hand or

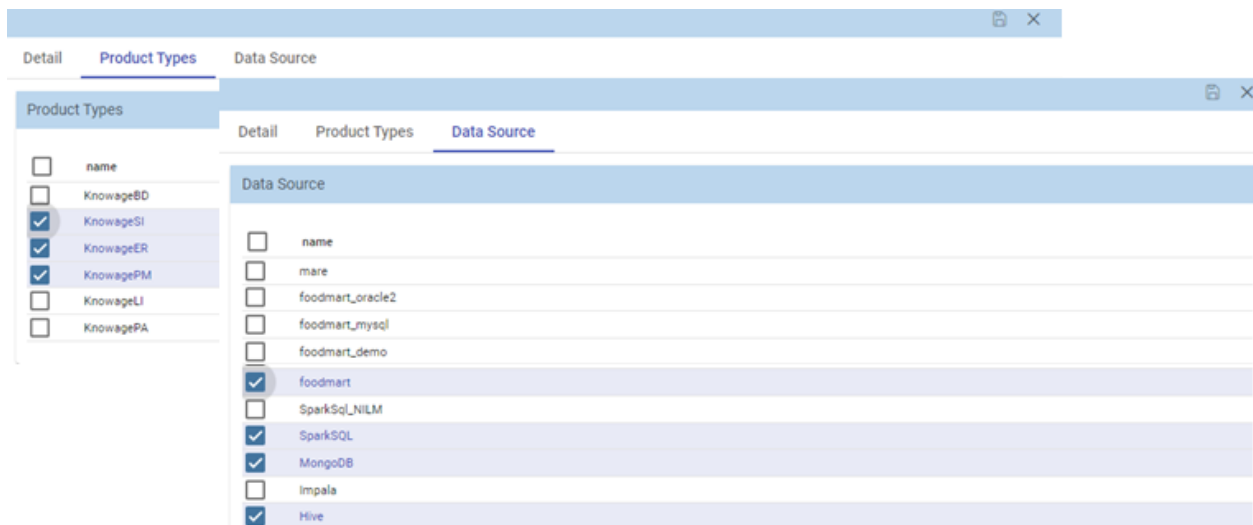


Fig. 2.18: Product types tab and Datasources tabs.

using Knowage Studio designers, when available. For each analytical document the history of templates is maintained. Old templates can be restored if needed. A new version is saved at each deployment, either manual or from Knowage Studio.

The **Templates** functionality allows the deletion of all those templates that have been created before a specific date. This kind of operation allows the administrator to clean the environment and save some space in the Knowage metadata database once a *document life cycle* is completed.

After inserting a date by clicking on the *calendar* icon, click on the *funnel* icon and select the documents of your interest. The *Delete* button deletes the templates (of the selected documents), uploaded before the specified date. If all the templates of a document come first of the specified date, the last template uploaded will be kept, so that no document is accidentally deleted. See figure below.

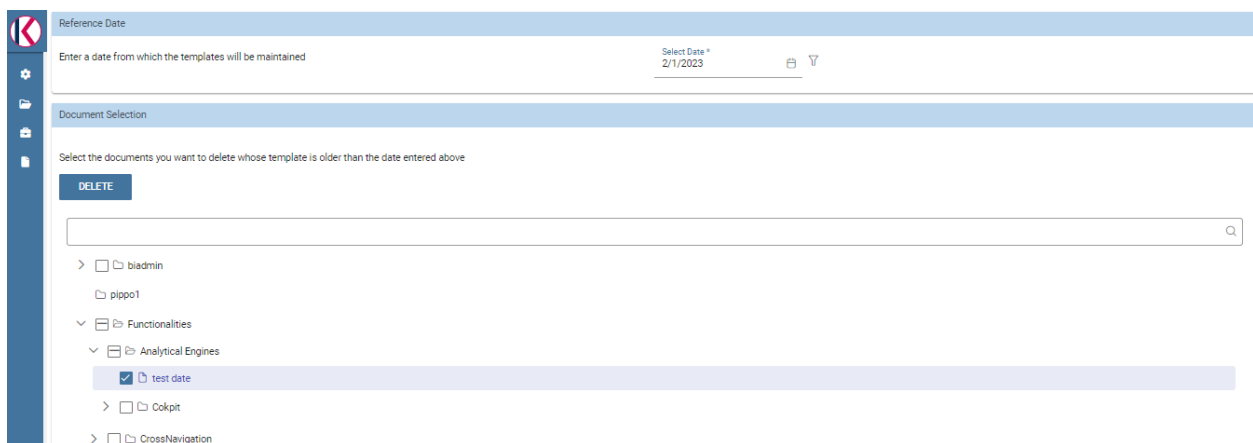


Fig. 2.19: Deleting templates

2.3.3 Themes

This functionality allows the creation/management of themes to be used in the css style of your documents.

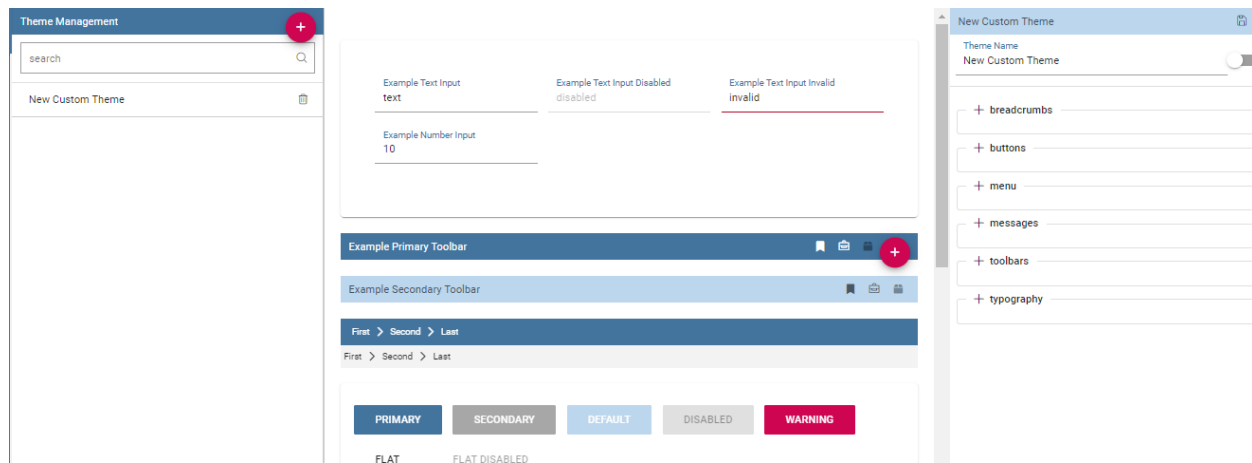


Fig. 2.20: Themes

2.3.4 Download installation configuration

This feature allows the downloading of a *config.zip* file containing the details of the installation configuration.

2.3.5 Licenses

This feature allows the management of Knowage licenses by adding, updating, downloading or removing a license.

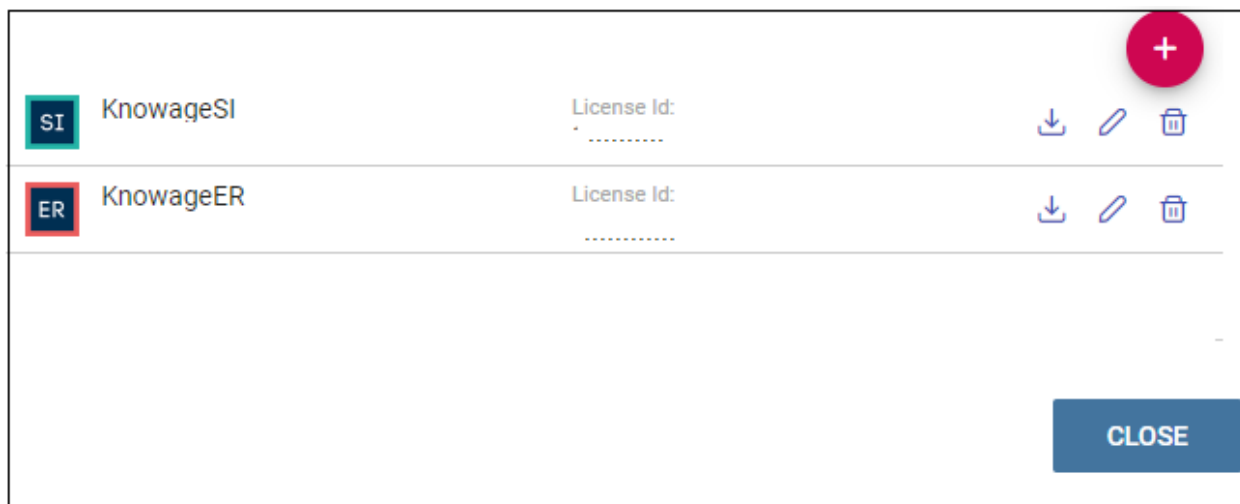


Fig. 2.21: Events

2.3.6 Events

This feature allows to search through all the events (i.e. the scheduling of a report) existing on the Knowage server, specifying a range of dates and the *type* of event. Types of events are:

- Scheduler
- ETL

- CommonJ
- Data_Mining

See below image.

User	Date	type	
Scheduler	28/04/2022 08:20:04	SCHEDULER	
Scheduler	28/04/2022 08:20:00	SCHEDULER	\$(scheduler.startexecsched) Registry Analytical Driver
Scheduler	10/11/2021 10:44:02	SCHEDULER	\$(scheduler.startexecsched) BI-05 - Dashboard Performance ...
Scheduler	16/06/2020 14:19:04	SCHEDULER	\$(scheduler.endexecsched) Hello World
Scheduler	16/06/2020 14:19:00	SCHEDULER	\$(scheduler.startexecsched) Hello World
Scheduler	16/06/2020 14:18:47	SCHEDULER	\$(scheduler.endexecsched) Hello World
Scheduler	16/06/2020 14:18:46	SCHEDULER	\$(scheduler.endexecsched) Hello World
Scheduler	16/06/2020 14:18:46	SCHEDULER	\$(scheduler.endexecsched) Hello World
Scheduler	16/06/2020 14:18:45	SCHEDULER	\$(scheduler.endexecsched) Hello World
Scheduler	16/06/2020 14:18:44	SCHEDULER	\$(scheduler.endexecsched) Hello World
Scheduler	16/06/2020 14:18:44	SCHEDULER	\$(scheduler.endexecsched) Hello World
Scheduler	16/06/2020 14:18:00	SCHEDULER	\$(scheduler.startexecsched) Hello World

Fig. 2.22: Events

2.4 Import/Export

This functionality allows to import/export items belonging to *Gallery*, *Documents*, *Menu*, *Users*, *KPIs* and *Catalogs*. After exporting, the user can import the information previously extracted into a different Knowage installation or tenant.

DATA PROVIDERS	CATALOGS	TOOLS	SERVER SETTINGS	INTERNATIONALIZATION
Data source	Business Models	Scheduler	Configuration	Manage Internationalization
Data set	Mondrian schemas	Scheduler Monitor	Domain	
	Svg	Hierarchies Editor	menu.Categories	IMPORT/EXPORT
PROFILE MANAGEMENT	Layers	Cache Manager	Metadata	Artifacts
Profile Attributes	Glossary Usage	Cross Navigation		Documents
Roles	Glossary Definition	Alert	SERVER MANAGER	Menu
Users	Timespan	News	Tenants	Users
Menu configuration	Calendar	Resource manager	Templates	KPIs
Functionalities	Functions		Themes	Catalog
BEHAVIOURAL MODEL	Widget Gallery	KPI MODEL	Download installation configuration	Analytical Drivers
Lovs		KPI	Licenses	Glossary
Analytical drivers		Measure/Rule	Event	Alerts
Constraints		Target		
Behavioural Model		KPI Scheduler		
Lineage		Scorecard		

Fig. 2.23: Import/Export menu

2.4.1 Artifacts

Export

It is possible to export widgets and functions to a zip file to use afterwards as a *source* to be imported to a different environment. As shown below, select the items of your interest and click on *EXPORT*.

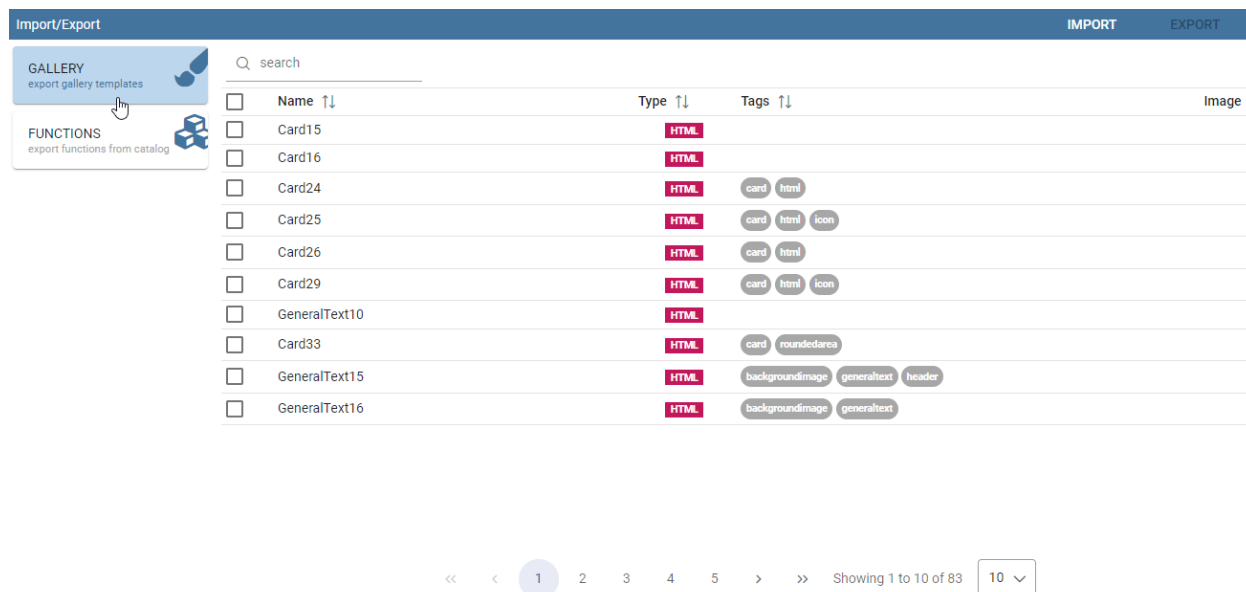


Fig. 2.24: Artifacts Export

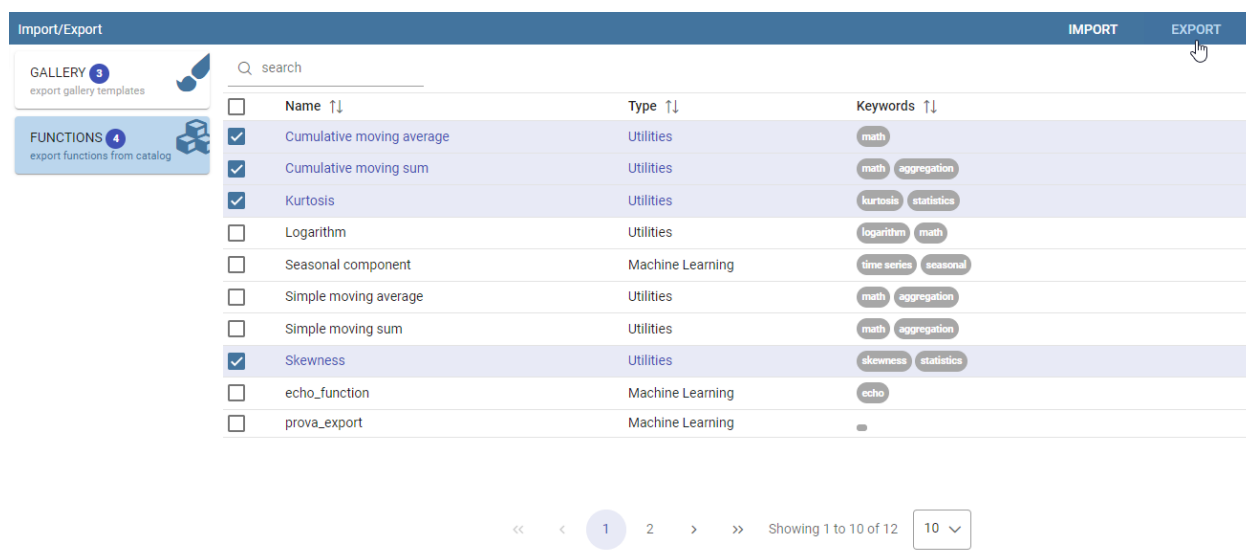


Fig. 2.25: Artifacts Export - Items selection

After the selection, give a name to the file and confirm with *Export* button.

Import

You can import the zip file formerly exported just clicking on *IMPORT*. See below image:

Clicking on *NEXT*, it is possible to see and select the items to be imported..

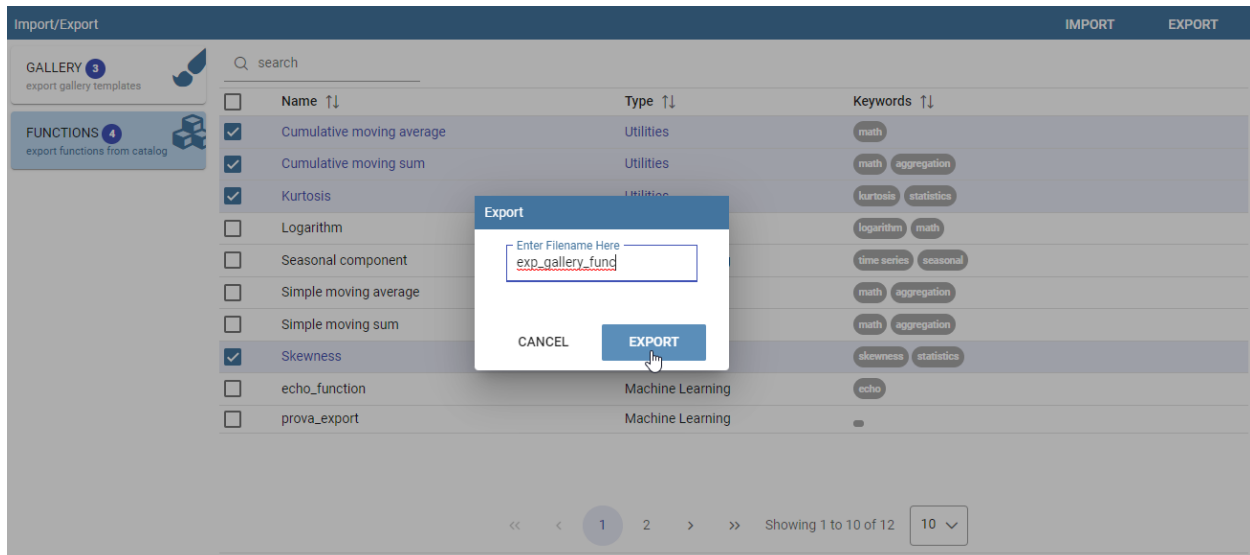


Fig. 2.26: Artifacts Export - setting name for zip file

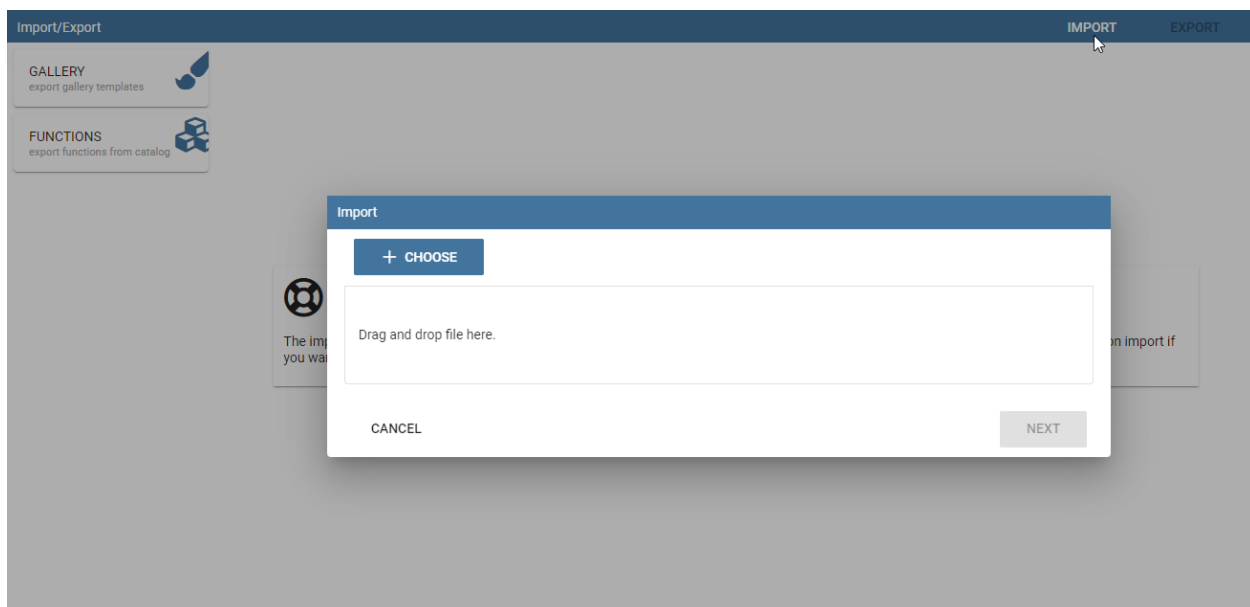


Fig. 2.27: Artifacts Import - Export file selection #1

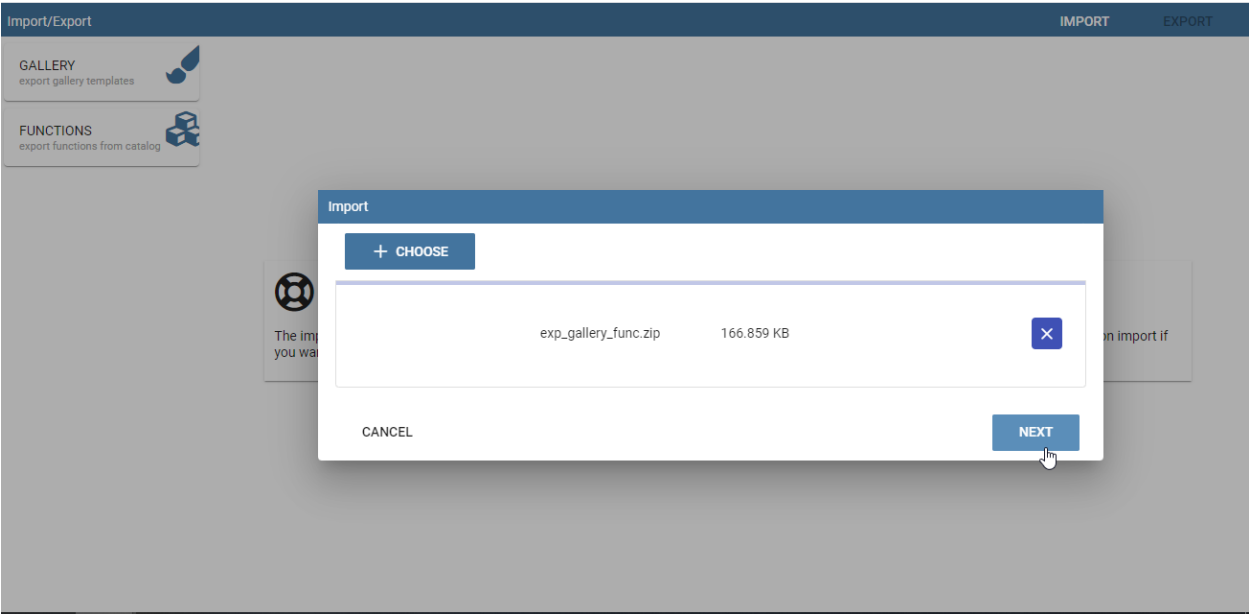


Fig. 2.28: Artifacts Import - Export file selection #2

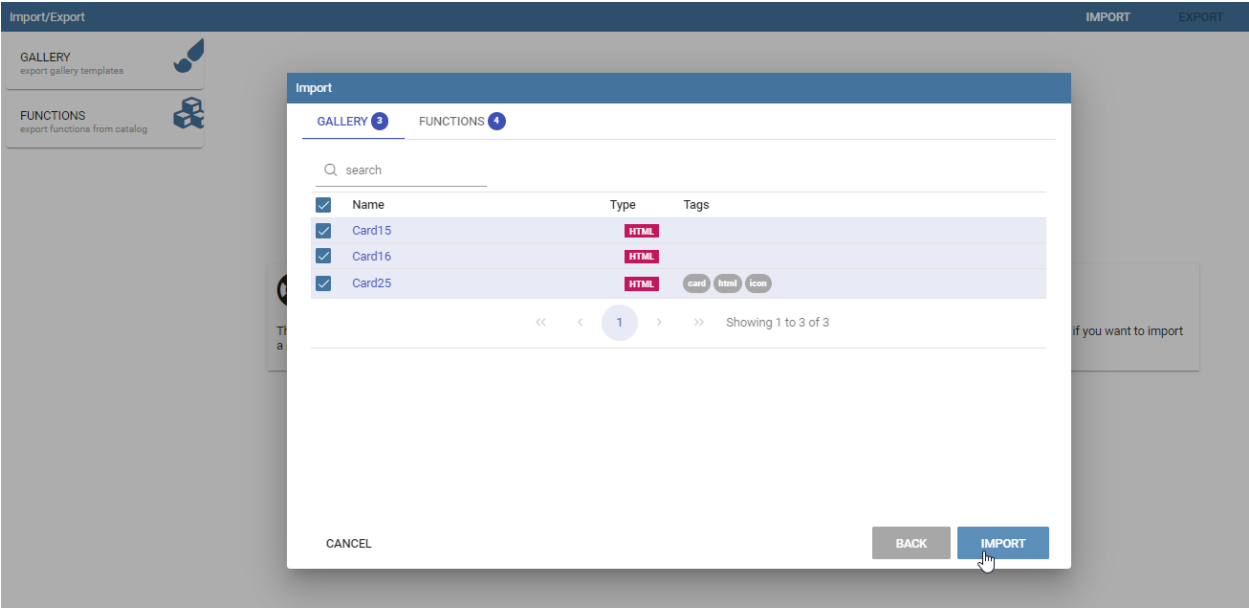


Fig. 2.29: Artifacts Import - items selection

Click on the **IMPORT** button to make selected widgets and functions available in the new environment.

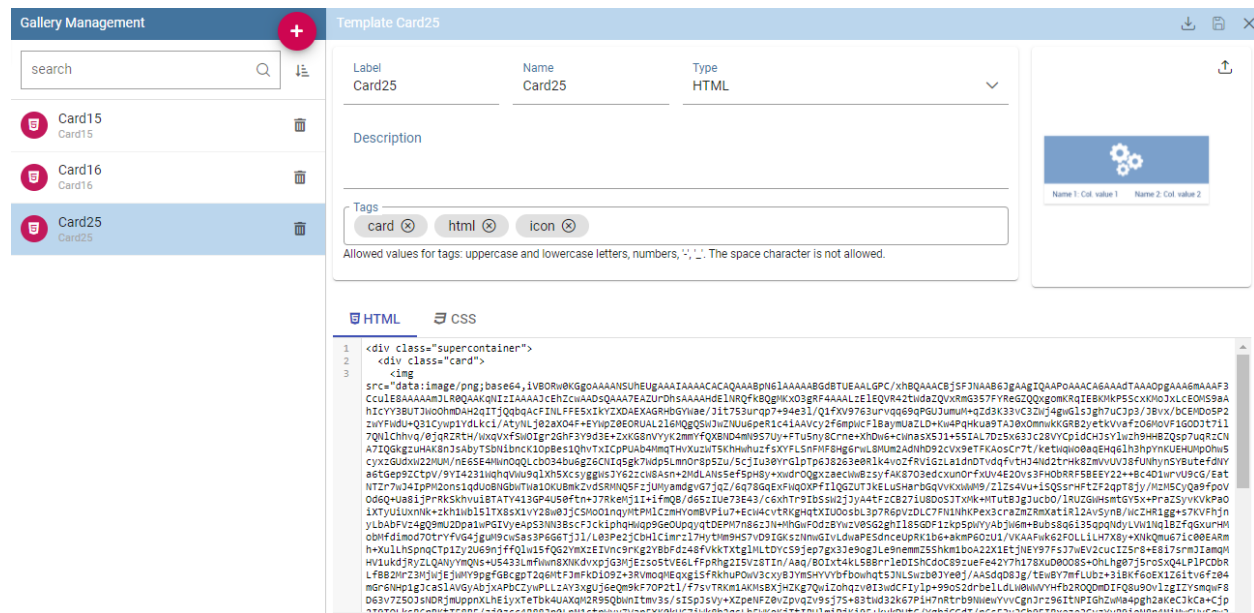


Fig. 2.30: Gallery Management Gui

2.4.2 Documents

This feature allows to download a zip file of the whole or a part of the documents existing in your Knowage installation.

Below, you can see how the export editor looks like.

Fill in the name of your export and select which documents to export. You can browse through folders by clicking on the relative icon. Check the items to include in the file. The export icon changes colour from grey to pink. Before starting to export, you can decide whether to include or not the following options:

- **Olap customized View** By checking this property, the export will include all the customized views saved into the selected OLAP documents. Customized views can be retrieved clicking on the option *Show OLAP custom View* of the Document Menu. See below image for more details:
- **Scheduled documents** By checking this property, the export will include all the scheduled executions saved into the selected documents. You can find the scheduled executions clicking on the option *Show Scheduled Execution* of the Document Menu. See the figure below:
- **BIRT Translation** By checking this property, the export will include all the *translation* added into the *Localization* functionalities of BIRT templates.
- **Schedule configurations** By checking this property, the export will include all the scheduling associated to the selected documents. At the end of the import, the scheduling information will be saved into the Scheduler section.
- **Export the document only in the selected functionality** By checking this property, the export will include documents only if they are inside a selected functionality.
- **Related documents** By checking this property, the export will include all those documents linked through the cross navigation to the selected documents.

You will be asked to map *Roles*, *Engines*, *Datasources* and *Metadata* from *Source* to *Target*.

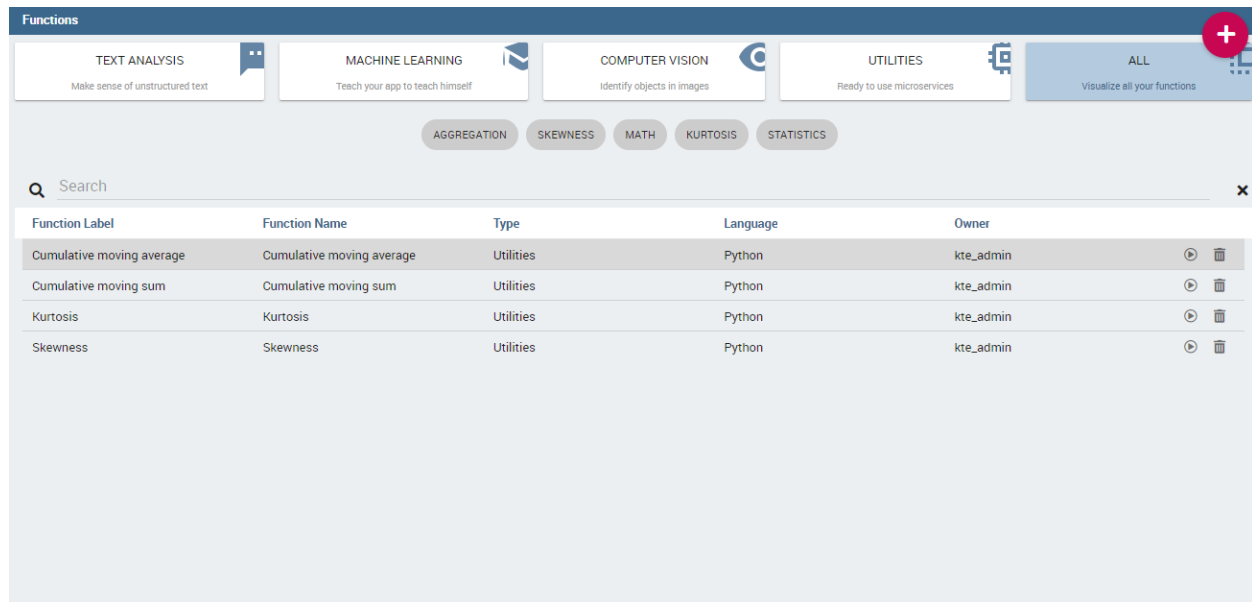


Fig. 2.31: Functions Management Gui

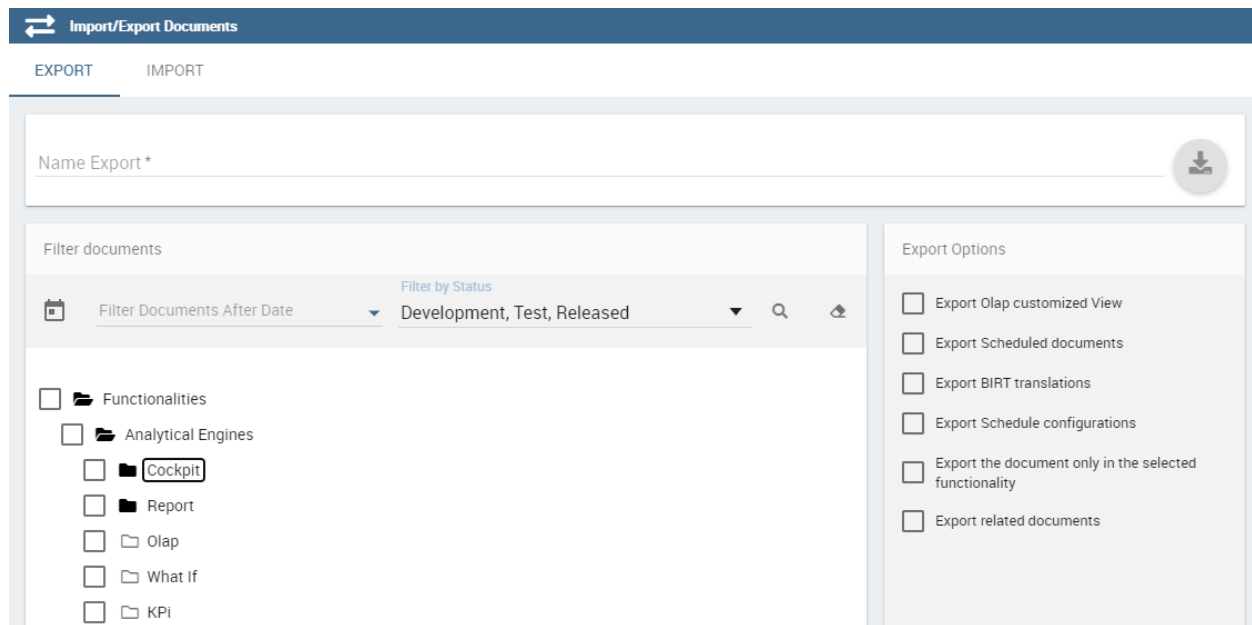


Fig. 2.32: Document Export

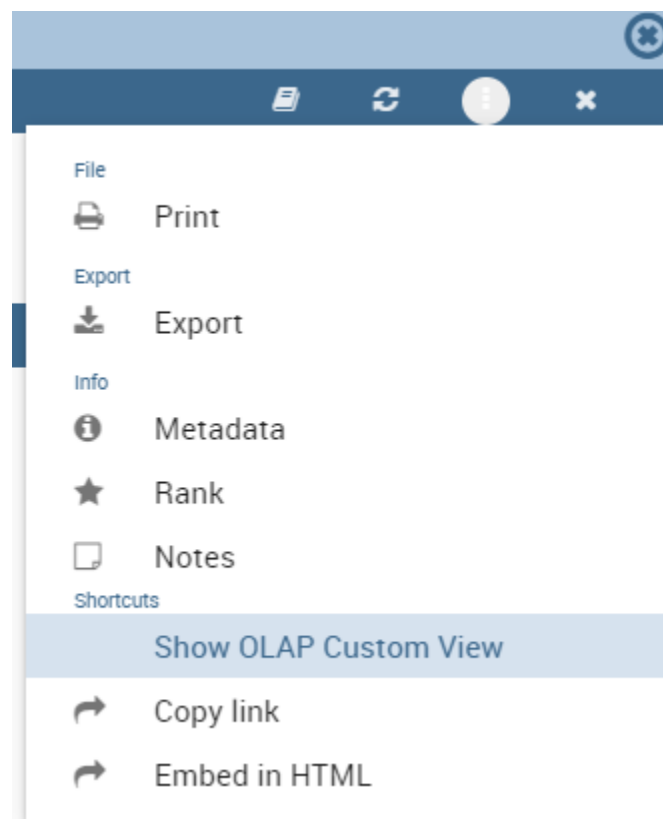


Fig. 2.33: Olap customized view

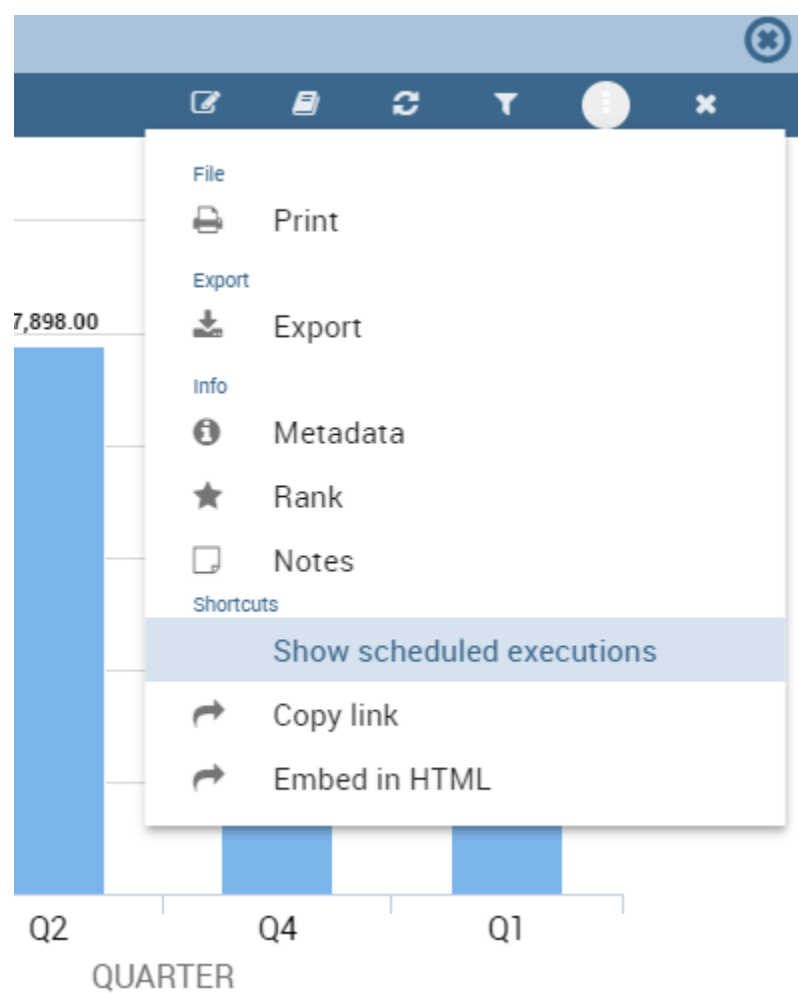


Fig. 2.34: Scheduled documents

The figure shows the 'Import/Export Documents' interface. The 'IMPORT' tab is selected. The 'Upload File' section contains a 'BROWSE' button and a radio button for 'No Associations' (which is selected). Below it is a radio button for 'File Associations'.

Fig. 2.35: Document Import

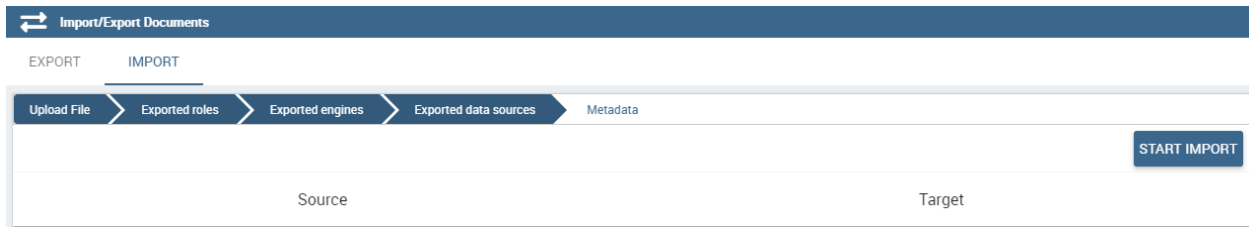


Fig. 2.36: Document Import Wizard

If a role does not match any of the existing ones, the role will be created. **Please bear in mind that all the target metadata with the same label i.e. documents, lovs, drivers will be overwritten when importing.**

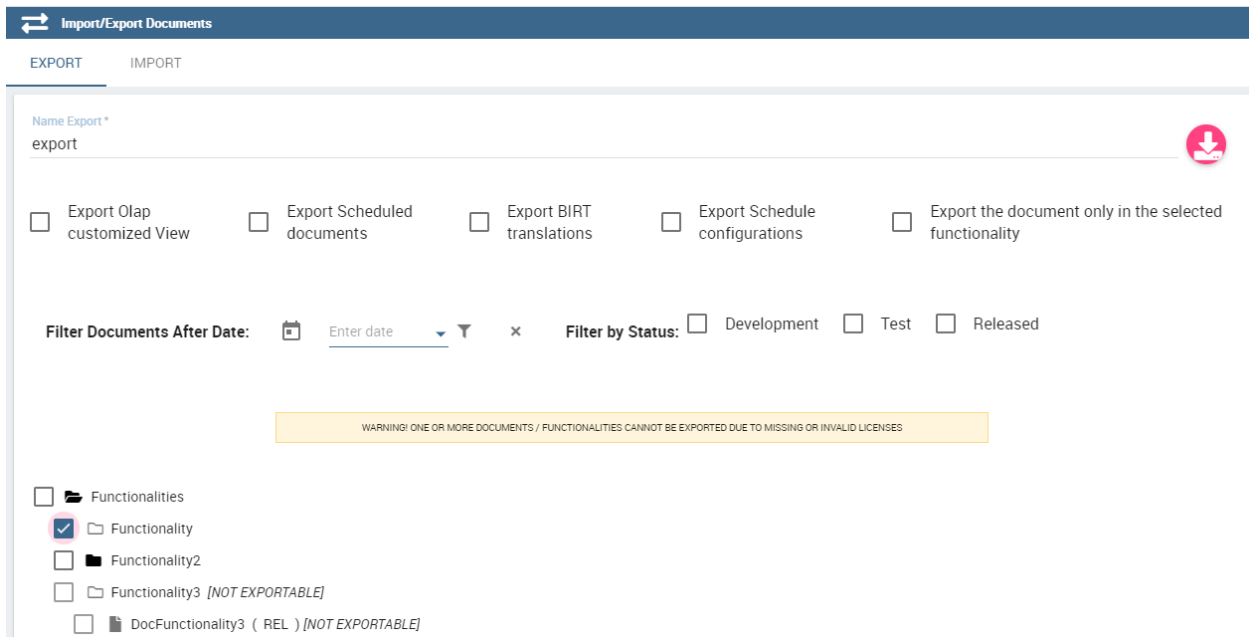


Fig. 2.37: Document Export Missing licenses

Before starting the import procedure, licenses are checked and if one or more are missing or are not valid, the user will be informed with the list of all those documents that will not be imported. See image below.

2.4.3 Menu

This feature lets you import/export the menu structure. The image below refers to the export functionality.

Fill in the name of the file to export. Although the export icon changes color from grey to pink you need to select at least one item from the menu structure.

To upload the zip file generated with the above process, in another installation, just click on **Menu** of the **Import\Export** item, switch to the **IMPORT** tab and click on *Browse* to search the zip file.

Click on *NEXT* as shown below.

After clicking on *NEXT*, you will be asked to map roles from *Source* to *Target*. If a role does not match map any of the existing ones in the target environment, it will be created.

You can click on *START IMPORT*.

Import/Export Documents

EXPORT IMPORT

Upload File Exported roles Exported engines Exported data sources Metadata

START IMPORT

Documents that NOT will be imported

WARNING! ONE OR MORE DOCUMENTS CANNOT BE IMPORTED DUE TO MISSING OR INVALID LICENSES

Source	Target
Cockpit Cockpit	knowagecockpitengine Cockpit Engine Cockpit Engine
leaf leaf	knowagecockpitengine Cockpit Engine Cockpit Engine

Documents that will be imported

Source	Target
lka lka	lka lka

Fig. 2.38: Document Import Missing licenses

Import / Export Menu

EXPORT IMPORT

Name Export *
menu_export

SELECT ALL DESELECT ALL

Current database

- ☐ Menu1
- ☐ Menu2
- ☐ Menu3
- ☐ Menu4
- ☐ Menu5
- ☐ Menu6

Fig. 2.39: Menu Export

Import / Export Menu

EXPORT IMPORT

import

BROWSE new_menu.zip

Fig. 2.40: Menu Import

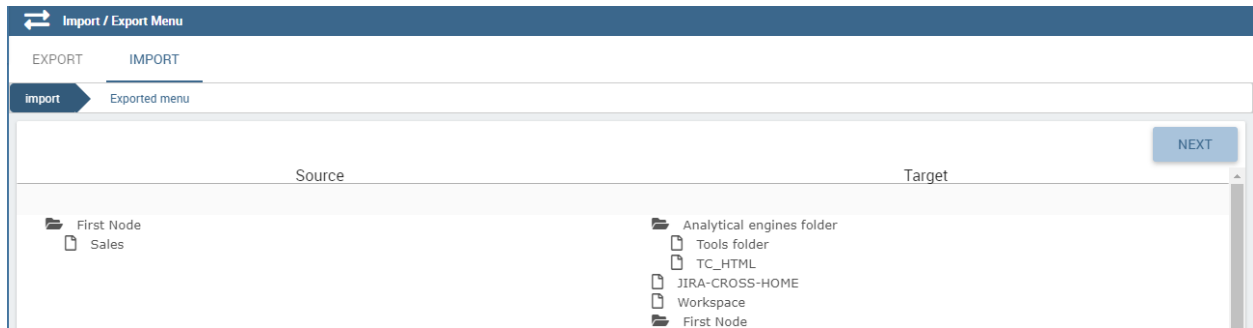


Fig. 2.41: Menu Import

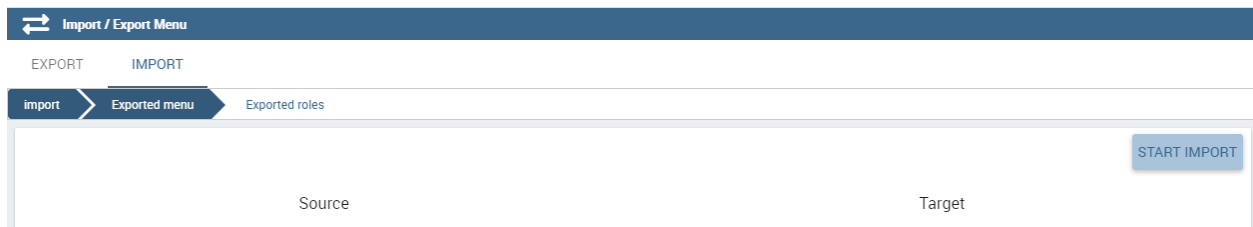


Fig. 2.42: Menu Import Wizard

2.4.4 Users

This functionality allows to export/import users from one installation/tenant to another. See the image below.

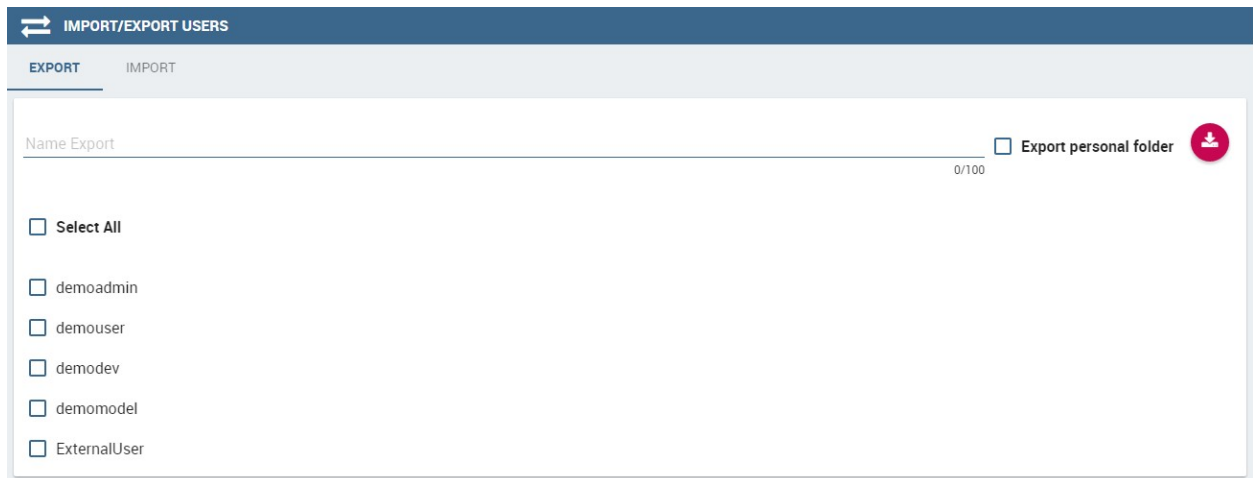


Fig. 2.43: User Export

To generate the zip file, fill in the name of your file and select the users to include. You can also include the personal folder of the users just checking the option **Export Personal folder**.

To import your zip file, log in and select **Users** from **Import/Export**. Switch to the **Import** tab and click on *Browse*. Choose the file and click on the import icon. The list of users contained in your file are uploaded. Make your selection and click on the arrow to move them to the other side. Now click on the *Start import* button and your users will be created. See figure below.

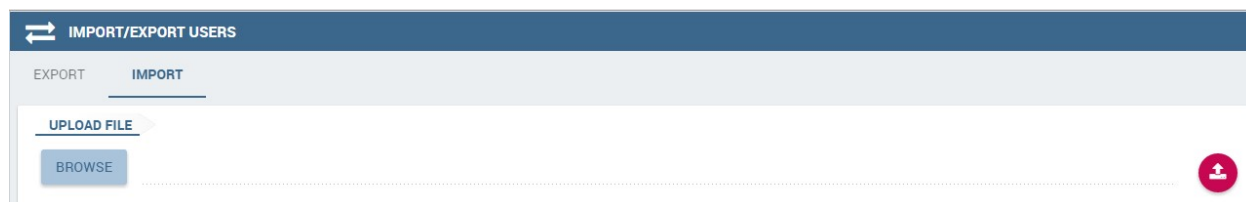


Fig. 2.44: User Import

Important: All users involved in the import procedure will have the password changed with the value set in *Advanced configuration*.

2.4.5 Catalogs

This functionality allows to Export/Import the following elements:

- Datasets,
- Business models,
- Mondrian catalogs,
- Layers,
- SVG files.

The details are shown in the below figure.

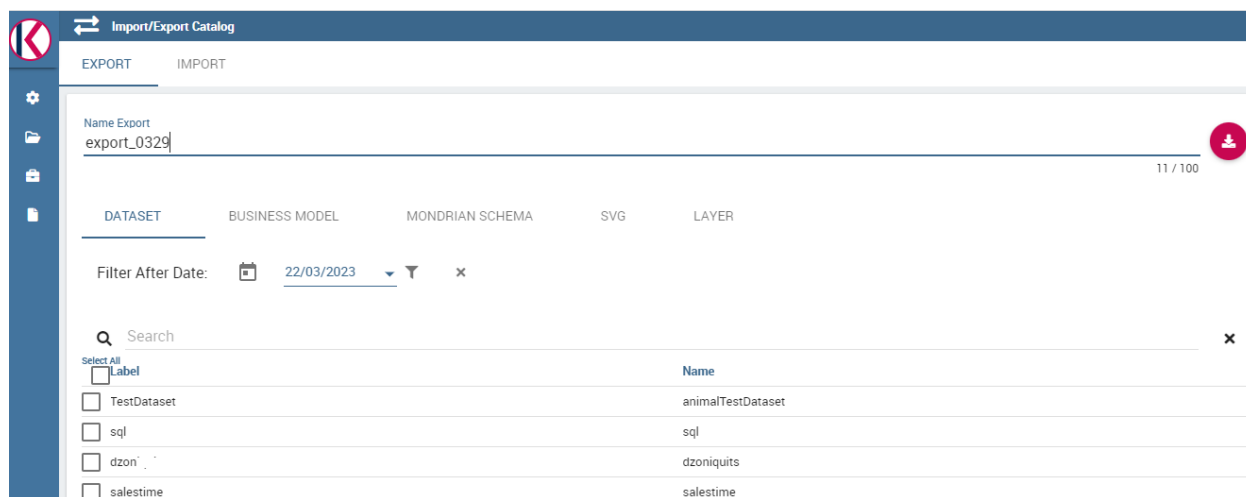
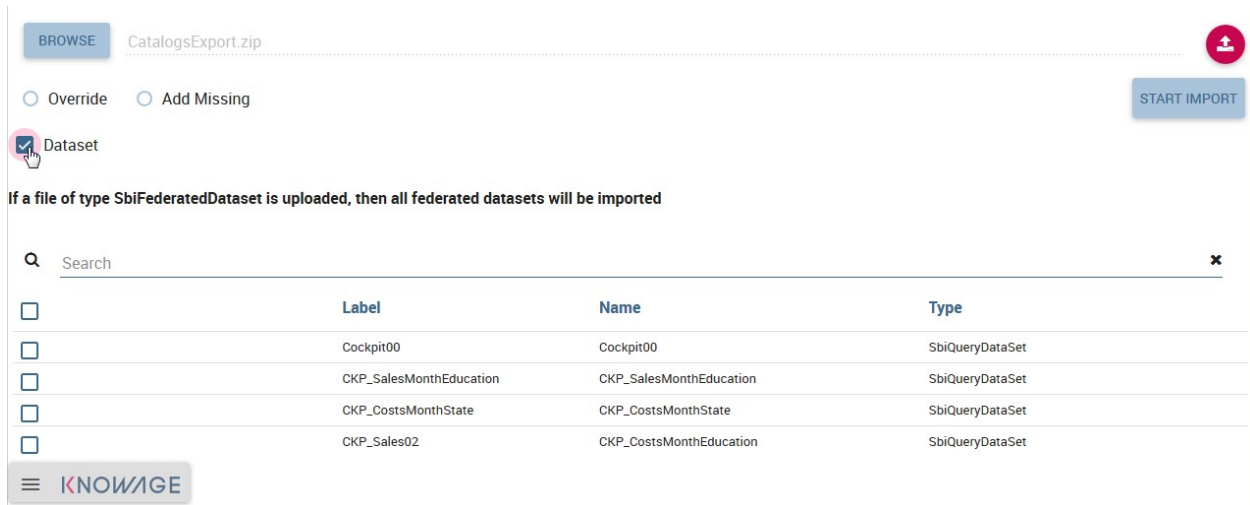


Fig. 2.45: Catalogs Export

To generate the zip file, mark the items to include and fill in the export name. Save the export file in your local system.

To import the zip file, log in to the source tenant, select **Catalogs** from the *Import/Export* menu and switch to the *Import* tab. Click on *Browse* and choose the file previously created through the exportation. Uploading the file, the list of available catalogs are displayed. Choose the ones to be imported, decide if you want to override them or just add the missing ones and then click on *Start import*. Your catalogs will be created in this environment.



The interface shows a 'BROWSE' button next to the file 'CatalogsExport.zip'. Below it are radio buttons for 'Override' and 'Add Missing'. A 'START IMPORT' button is on the right. A 'Dataset' checkbox is checked. A message states: 'If a file of type SbiFederatedDataset is uploaded, then all federated datasets will be imported'. Below this is a search bar and a table of datasets.

	Label	Name	Type
<input type="checkbox"/>	Cockpit00	Cockpit00	SbiQueryDataSet
<input type="checkbox"/>	CKP_SalesMonthEducation	CKP_SalesMonthEducation	SbiQueryDataSet
<input type="checkbox"/>	CKP_CostsMonthState	CKP_CostsMonthState	SbiQueryDataSet
<input type="checkbox"/>	CKP_Sales02	CKP_CostsMonthEducation	SbiQueryDataSet

The 'KNOWAGE' logo is at the bottom left.

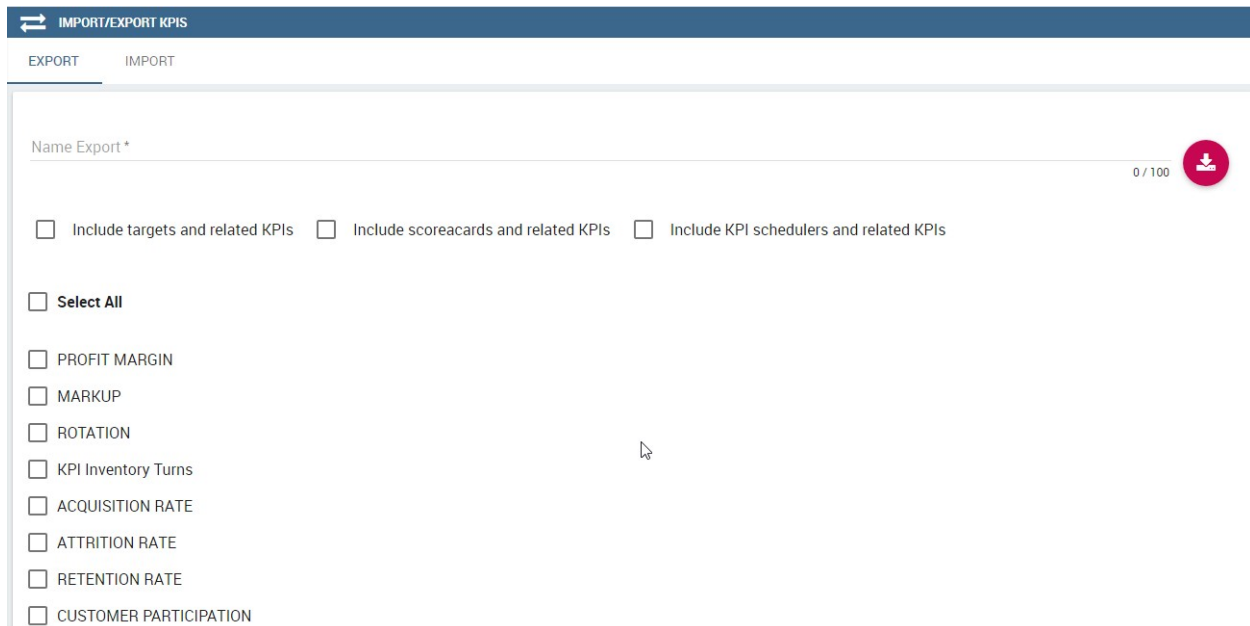
Fig. 2.46: Catalogs Import

2.4.6 KPIs

In this section we describe how to manage the import/export of KPIs between two tenants.

The user must enter Knowage as administrator of the source tenant and click on **KPIs** from the *Import/Export* menu panel.

The page contains the *Export* and the *Import* tab, where the user can select the KPIs for the export/import respectively.



The window has a title bar 'IMPORT/EXPORT KPIs' and two tabs: 'EXPORT' and 'IMPORT'. The 'IMPORT' tab is active. It features a 'Name Export *' field, a progress indicator '0 / 100', and a download icon. Below are three checkboxes for including related items: 'Include targets and related KPIs', 'Include scorecards and related KPIs', and 'Include KPI schedulers and related KPIs'. A 'Select All' checkbox is also present. A list of KPIs follows, each with an unchecked checkbox:

- PROFIT MARGIN
- MARKUP
- ROTATION
- KPI Inventory Turns
- ACQUISITION RATE
- ATTRITION RATE
- RETENTION RATE
- CUSTOMER PARTICIPATION

Fig. 2.47: KPIs Import window

Let's start from the export feature. The user at first selects the KPIs to be exported and in addition can include:

- targets,
- the scorecards related to the selected KPIs,

- schedulations.

Click on the red download button to get a zip file.

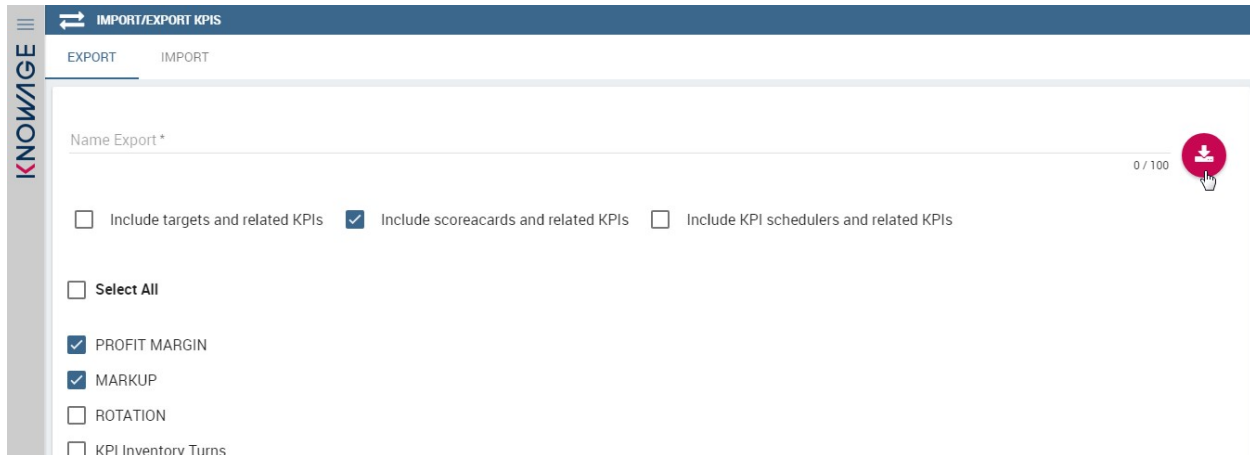


Fig. 2.48: Start export button

Once the zip file is downloaded, the user has to switch to the tenant used for the import and as admin enter the Import/Export KPIS functionality and move to the Import tab.

The user must therefore browse the personal folder to catch the zip file and click on the red upload button, as shown below.

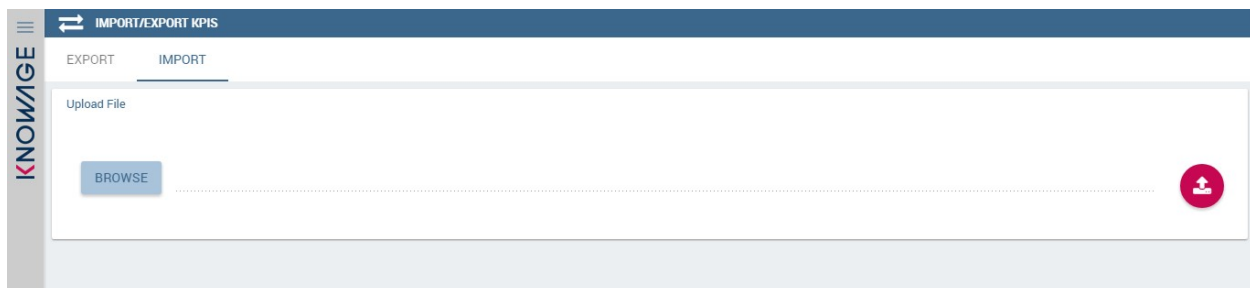


Fig. 2.49: Import tab

Referring to the following image, the user has to specify if:

- overwriting the existing KPIS and their related formulas
- importing targets,
- importing scorecards,
- importing schedulations.

Once the import is started, the GUI leads the user to finalize the import procedure. In particular, the user is asked to map data sources correctly. See figure below.

The process ends successfully when the wizard shows up as following.

2.4.7 Analytical Drivers

This option allows to import/export the analytical drivers and their related LOVs.

IMPORT/EXPORT KPIS

EXPORTIMPORT

Upload FileExported KPIS

☐ Overwrite existing KPIS and rules

☐ Import targets and related KPIS

☐ Import scorecards and related KPIS

☐ Import KPI schedulers and related KPIS

NEXT

Select All☐

Name	Formula	Threshold	Targets
<input type="checkbox"/> PROFIT MARGIN	((SUM(STORE_SALES)-SUM(STORE_COST))/S...	Threshold_1	
<input type="checkbox"/> MARKUP	((SUM(STORE_SALES)-SUM(STORE_COST))/S...	Threshold_1 (Clone)	
<input type="checkbox"/> ROTATION	(SUM(WAREHOUSE_SALES)/SUM(UNIT_ORDE...	Threshold_1 (Clone) (Clone)	
<input type="checkbox"/> KPI Inventory Turns	SUM(WAREHOUSE_SALES_INV)/(SUM(UNIT_O...	Inventory Turns Threshold	
<input type="checkbox"/> ACQUISITION RATE	(SUM(tot_customer_attrited)/SUM(to_cust)	TH_ACQUISITION	
<input type="checkbox"/> ATTRITION RATE	(SUM(tot_customer_attrited)/SUM(to_cust)		
<input type="checkbox"/> RETENTION RATE	(SUM(tot_retained_cust)/SUM(to_cust))*100		
<input type="checkbox"/> CUSTOMER PARTICIPATION	(SUM(tot_loyal_customer)/SUM(to_cust))*100		

Fig. 2.50: Import KPIS settings

IMPORT/EXPORT KPIS

EXPORTIMPORT

Upload FileExported KPISExported Data Sources

START IMPORT

Source	Target
Exported data sources	
Foodmart (jndi) Foodmart java:comp/env/jdbc/foodmart	Foodmart

Fig. 2.51: Mapping data sources

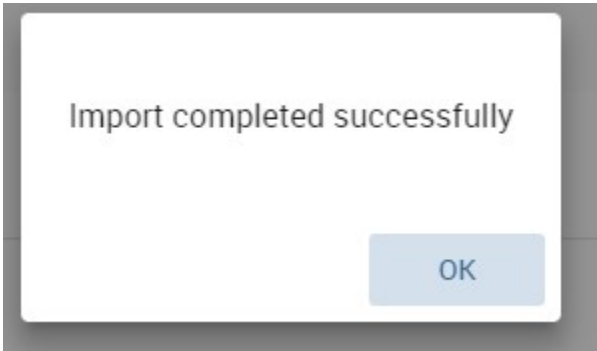


Fig. 2.52: Import KPIS ended successfully

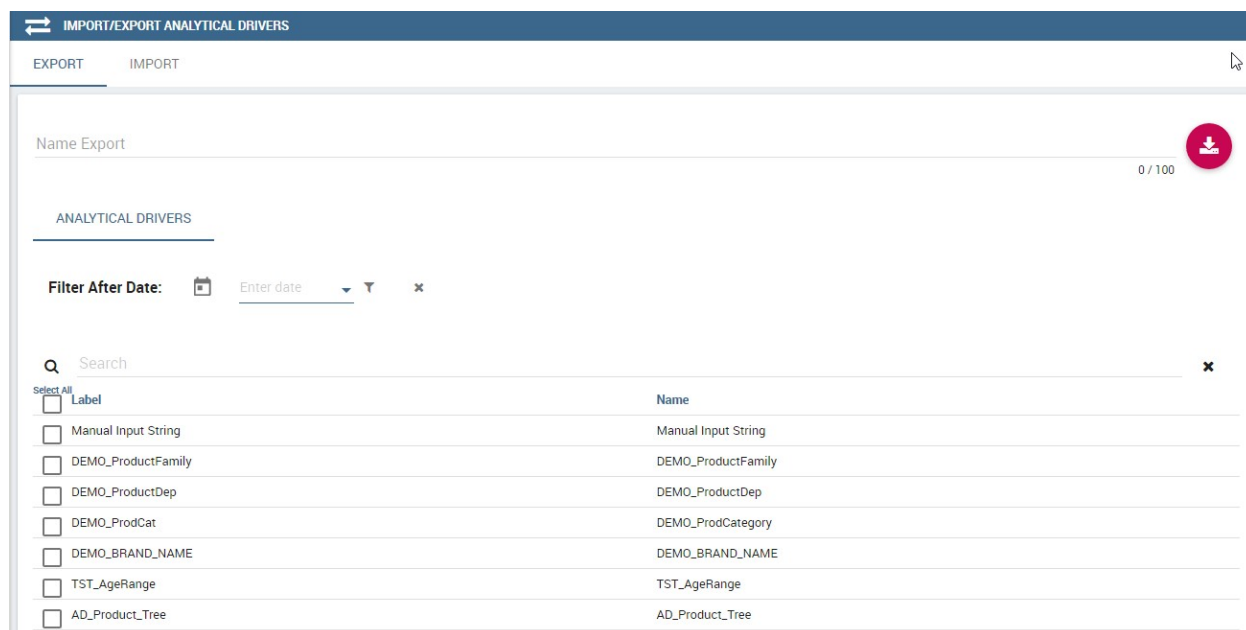


Fig. 2.53: Import/Export of analytical drivers

As shown in figure above, the window contains the Export and the Import tab. Use the Export tab to download the zip file to be used in the import process.

Afterwards:

- log in as administrator to the source tenant,
- assign a name to the export,
- check the analytical drivers of interest and
- click on the red download button, available at the top right corner of the page to get the zip file.

It is possible to narrow the search of the analytical drivers by filtering on their creation date.

Switch to the target tenant and log in as administrator. Use the Import tab to upload the zipped folder and finalize the import.

Use the GUI to upload the zip file and specify if overwriting the existing analytical drivers or just adding the missing ones. Then click on *NEXT* and continue by mapping roles between the source and the target tenants.

The process ends with a message containing the information about the import.

2.4.8 Glossary

The export/import of glossary allows the user to align glossaries among tenants.

In the *Export* tab, the user is asked to select the glossaries to export and type a name that will be assigned to the zip file. The searching functionality can be used for the selection.

Afterwards, connecting as admin to a target tenant, the user selects the *Import* tab from the Export/Import main window.

Arrows are used to indicate the glossaries to consider for the import.

IMPORT/EXPORT ANALYTICAL DRIVERS

EXPORT IMPORT

Upload File

BROWSE dr.zip

☒ Override ☐ Add Missing

NEXT >

Search

Select All

Name	Type
DEMO_ProductDep	

Fig. 2.54: Import of analytical drivers

IMPORT/EXPORT ANALYTICAL DRIVERS

EXPORT IMPORT

Upload File > Exported Roles

NEXT >

Source	Target
/demo/admin	Exported roles
/demo/user	Exported roles

Fig. 2.55: Import of analytical drivers

IMPORT/EXPORT ANALYTICAL DRIVERS

EXPORT IMPORT

Upload File > Exported Roles > Exported Data Sources

START IMPORT

Source	Target
Foodmart (jndi)	Foodmart
java.comp/env/jdbc/foodmart	Foodmart

Fig. 2.56: Import of analytical drivers

The screenshot shows the 'IMPORT EXPORT GLOSSARY' window with the 'EXPORT' tab selected. The window has a dark blue header with a double-headed arrow icon and the title 'IMPORT EXPORT GLOSSARY'. Below the header, there are two tabs: 'EXPORT' and 'IMPORT'. The 'EXPORT' tab is active. The main area contains a text input field labeled 'Name Export *' with the value 'Glossary'. To the right of this field is a red circular button with a download icon and the text '8 / 100'. Below the input field, there is a section titled 'Filter Glossary After Date:' with a calendar icon, a text input field labeled 'Enter date:', a dropdown arrow, and a close icon. At the bottom, there are two checkboxes: 'Select All' (unchecked) and 'MARKET ANALYSIS' (checked).

Fig. 2.57: Export/Import of glossaries window

The screenshot shows the 'IMPORT EXPORT GLOSSARY' window with the 'IMPORT' tab selected. The window has a dark blue header with a double-headed arrow icon and the title 'IMPORT EXPORT GLOSSARY'. Below the header, there are two tabs: 'EXPORT' and 'IMPORT'. The 'IMPORT' tab is active. The main area contains a 'BROWSE' button and a text input field labeled 'Glossary.zip'. To the right of this field is a red circular button with a download icon. Below the input field, there are two radio buttons: 'Override' (unchecked) and 'Add Missing' (checked). To the right of these radio buttons is a grey button labeled 'START IMPORT'. Below the radio buttons, there are two sections: 'Source' and 'Target'. Each section has a search icon, a text input field labeled 'Search', and a close icon. Below the search fields, there are two checkboxes: 'Select All' (unchecked) and 'MARKET ANALYSIS' (checked). In the center of the 'Source' and 'Target' sections, there are five red circular buttons with arrows: a right arrow, a left arrow, a double right arrow, and a double left arrow.

Fig. 2.58: Import of glossaries

2.4.9 Alerts

This functionality allows to Export/Import alerts.

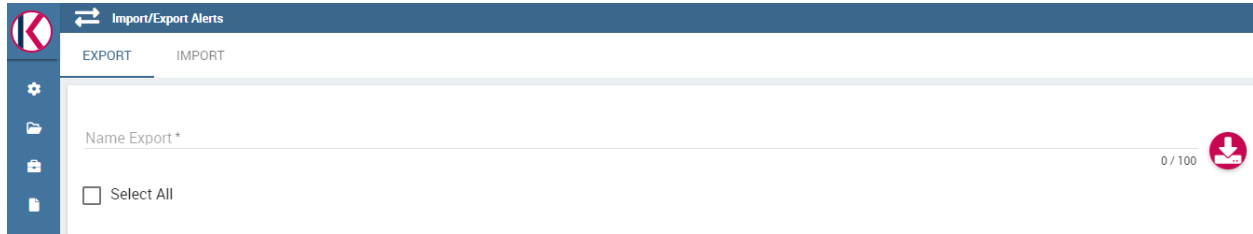


Fig. 2.59: Export alerts

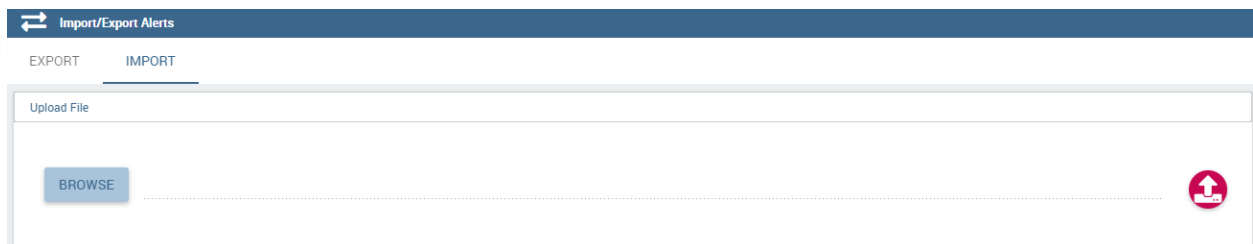


Fig. 2.60: Import alerts

2.5 Time Span

Most of times a technical user develops a business solution for the use and consumption of a third not-skilled end user. In the contest of self-service capability of Knowage, when business models are available to allow end users to query data, it is possible to arrange time or temporal intervals so that end users can use these time periods to filter data when working with the QbE interface.

2.5.1 Create a new Timespan

The Timespan functionality is available under the *CATALOGS* section of the Knowage main menu.

When clicking on **Timespan**, the user sees the list of already defined periods, if any. Possible actions are opening, searching, deleting or cloning a specific timespan. The delete and clone icons are available next to the timespan item name.

To add a new timespan item just click on the *plus* icon on the top right corner of the item list. The technical user is asked to assign a *Name* and to specify the *Type*, i.e. if it is a *Time* or a *Temporal* period. In case of *Time*, the start and the end time should be entered otherwise the start and the end date should be entered. There is also the possibility to associate a *Category* for profiling issues. See the below figure, as an example.

Whatever type is chosen, the technical user defines the start and the end of the interval and has to use the *Save* icon to insert it. Note that it is possible, for each type, to add more than one interval.

Once that the intervals have been defined they can be used inside the QbE interface to filter time or temporal attributes. For this part, please refer to the *Free Inquiry* chapter, in particular when dealing with Filters.

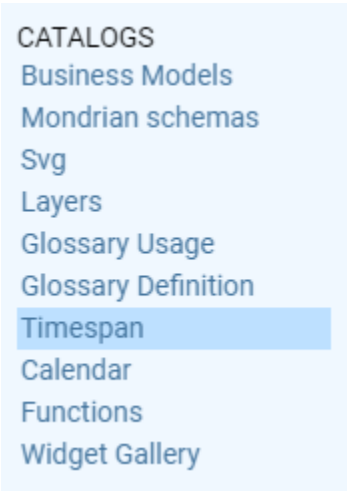


Fig. 2.61: Timespan

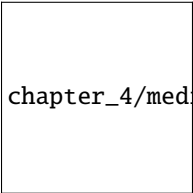


Fig. 2.62: Timespan list page

Name *

Type *
Temporal

Category

From
3/24/2023

To
3/24/2023

ADD

Name *

Type *
Time

Category

From
14:57

To
14:57

ADD

Fig. 2.63: Time and temporal type

1997_JanFeb

Name *

1997_JanFeb

Type *

Temporal

Category

From

3/24/2023

To

3/24/2023

ADD

From

02/01/1997

to

02/28/1997

From

04/01/1997

to

04/30/1997

Fig. 2.64: Adding more than one temporal intervals in one timespan

2.6 News

The **News management** is a helpful functionality to keep the end user constantly informed on any changements or updates on the Knowage platform. In this section we will describe the main steps to insert a news.

2.6.1 How to publish news

To add a news is mandatory to have administration privileges. An admin user can insert news entering the **News Management** functionality from the Knowage main menu.

Q search

DATA PROVIDERS

Data source

Data set

PROFILE MANAGEMENT

Profile Attributes

Roles

Users

Menu configuration

Functionalities

BEHAVIOURAL MODEL

Lovs

Analytical drivers

Constraints

Behavioural Model

Lineage

CATALOGS

Business Models

Mondrian schemas

Svg

Layers

Glossary Usage

Glossary Definition

Timespan

Calendar

Functions

Widget Gallery

TOOLS

Scheduler

Scheduler Monitor

Hierarchies Editor

Cross Navigation

Alert

News

Resource manager

KPI MODEL

KPI

Measure/Rule

Target

KPI Scheduler

Scorecard

SERVER SETTINGS

menu.Categories

Metadata

SERVER MANAGER

Templates

Themes

Download installation configuration

Event

INTERNATIONALIZATION

Manage

Internationalization

IMPORT/EXPORT

Artifacts

Documents

Menu

Users

KPIs

Catalog

Analytical Drivers

Glossary

Alerts

Fig. 2.65: News Management list item.

A new page opens where the left side contains the list of all the already inserted news whereas the right side shows the details of the selected news from the list.

To add a news, just click on the *plus* icon. Below, the information requested for the addition of a news.

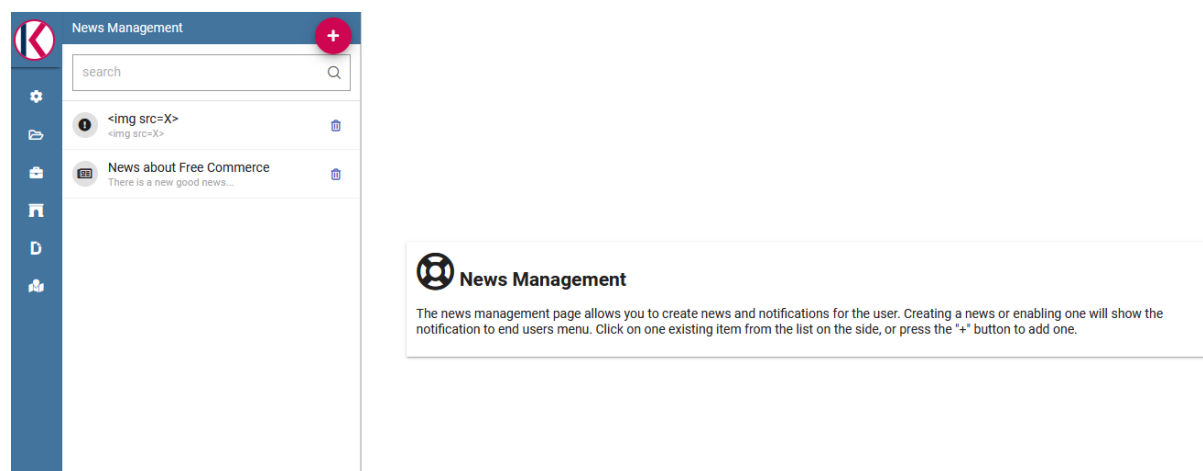


Fig. 2.66: Access news management list.

- **News title**
- **Expiration Date**, after this date the news will no longer be visible to the user
- **News Type**, three possible values between *News*, *Notification* and *Warning*
- **Content of the news**
- **Activation of the news**
- **Roles Permissions**

2.6.2 How the end user can read the news

When some news are added, the **news** icon prompts the number of unread news. ... figure:: media/image04.png

News notification.

Clicking on the icon, the user can access the content of the news.

2.7 Glossary

The **Glossary** functionality offers a way to find documents by browsing an index page.

2.7.1 Glossary management

Once logged in, the user can find the two menu items, **Glossary Definition** and **Glossary Usage** in *CATALOGS* of the Knowage main menu, as shown below.

To create a new glossary, just click on the *Glossary Definition* menu item. As shown in the figure below, the page contains two areas:

- **Words List**: contains the list of all the defined words to be used as labels to attach to analytical objects, as datasets or documents;
- **Glossary**: intended as a hierarchical structure made up of *Words*.

As shown above, next to each word, there are two icons:

News Detail

News Settings

News Title *

Expiration Date *

News Type *

News

Description *

This is the text that will appear on the news notification

0 / 140

Heading

Sans Serif

B

I

U

A

Roles Permissions

Role	<input type="checkbox"/>
/Demo/admin	<input type="checkbox"/>
/Demo/public	<input type="checkbox"/>
/Demo/user	<input type="checkbox"/>
/Kte/admin	<input type="checkbox"/>
/Registry_tenant/admin	<input type="checkbox"/>
Prof_roled	<input type="checkbox"/>

Fig. 2.67: Access news management list.

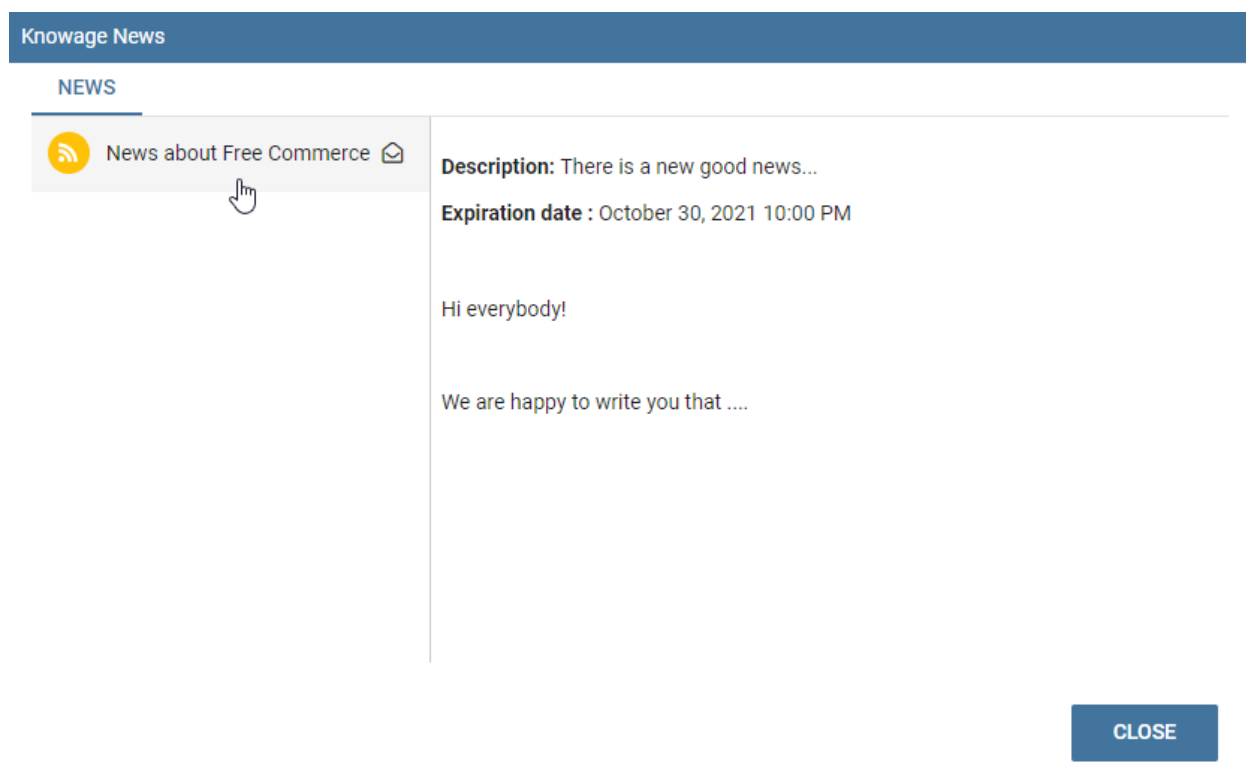


Fig. 2.68: News notification pop up.

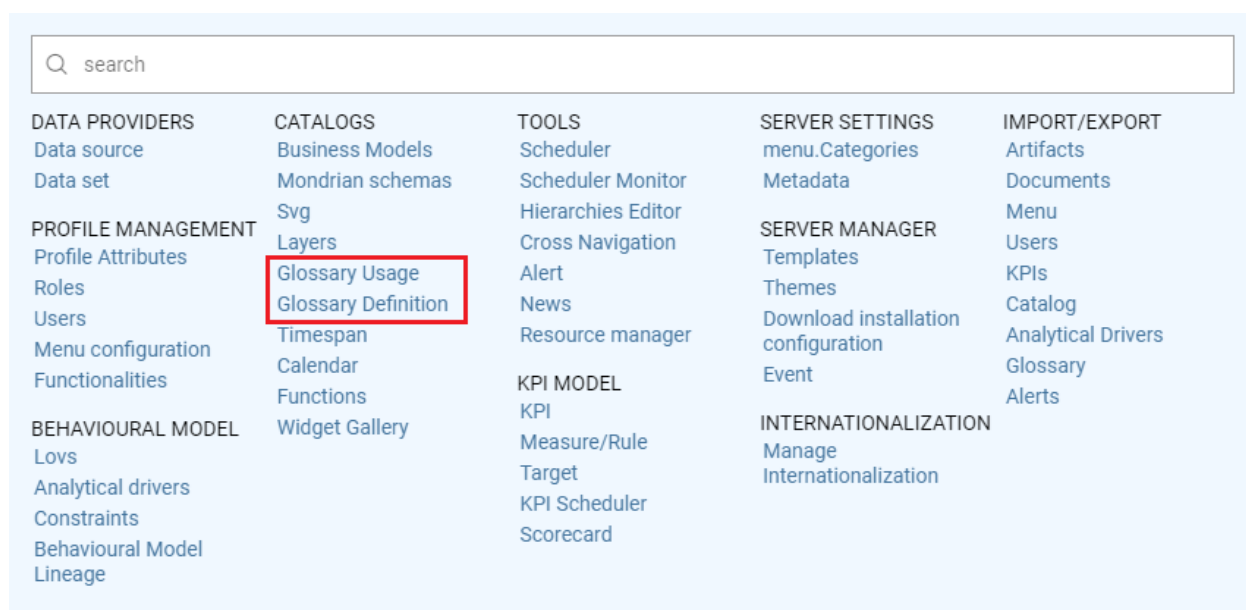


Fig. 2.69: Glossary menu items.

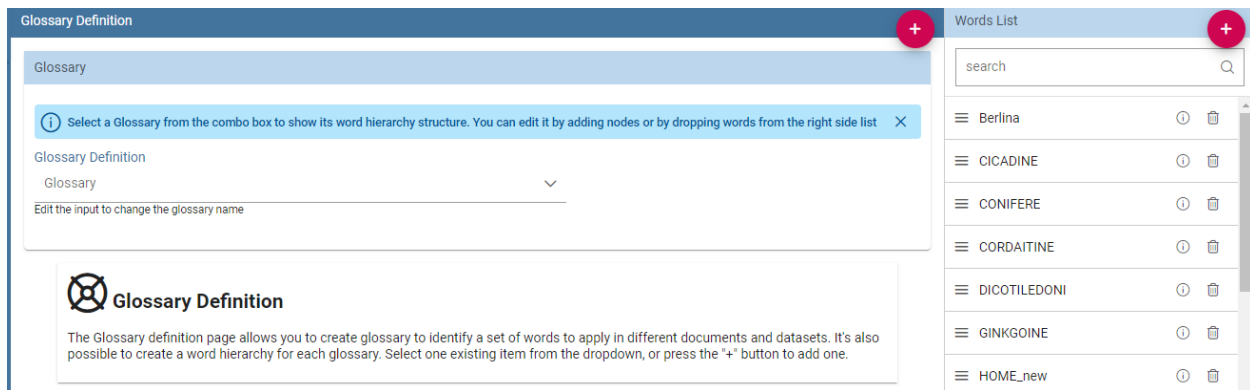


Fig. 2.70: Glossary definition window.

- *i*: prompts the details of the selected word: name of the *Word*, *Status*, *Category*, *Description*, *Formula*, *Links* to other words and *Attributes*
- *delete*: deletes the selected word

In the “Word” area are listed, if any, the words created in a previous moment. To explore the detail of each of them, the user just has to right click on it. A panel containing three features will be shown, as figure below highlights .

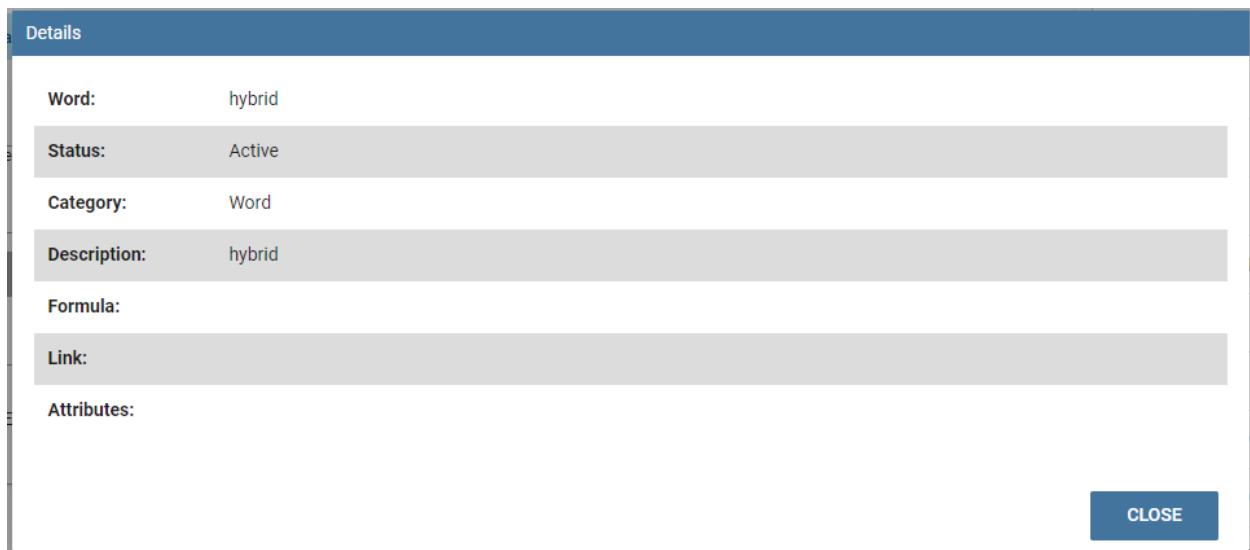


Fig. 2.71: Exploring the details of an existing word.

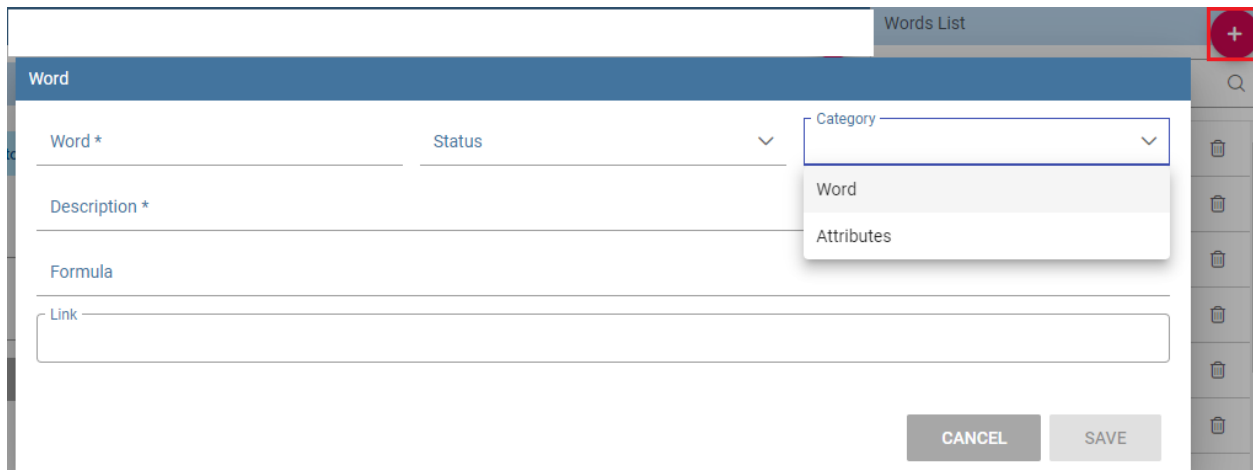
To add a new word, click on the *Plus* icon available next to the *Word List* text item and fill in at least the mandatory information, *Word*, *Category* and *Description*. Use the *Save* button to save it. Once saved, you can search for it or a different word just using the search filter.

The *Glossary* panel contains all the glossaries formerly created. To explore an existing glossary just open the menu of the *Glossary Definition* item and select one of the test options. The figure below shows an example. A hierarchical structure of the glossary appears, where each node has its own words .

To add a new glossary click on the *Plus* icon next to the *Glossary Definition* text item.

After assigning a name to the *Glossary Definition* field, the Glossary is automatically saved.

Click on *ADD NODE*, fill in the node name and save. The image below shows the *ADD Node* functionality and also



Words List

Word

Word *

Status

Category

Word

Attributes

Description *

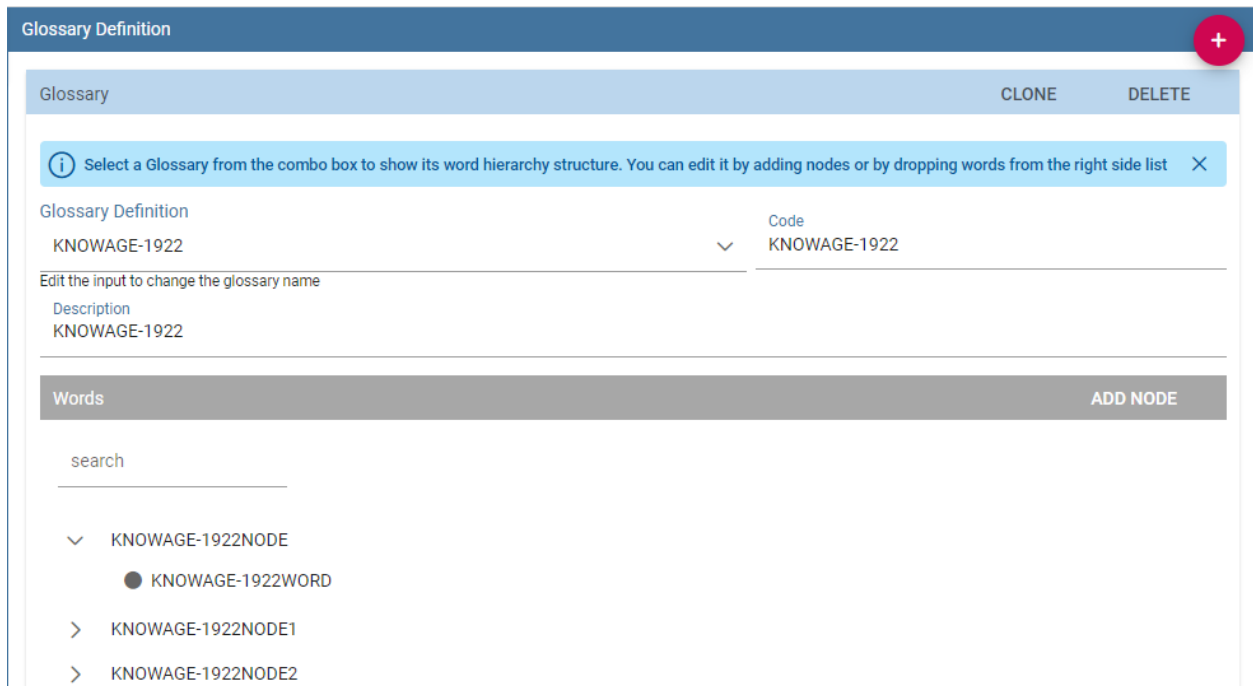
Formula

Link

CANCEL

SAVE

Fig. 2.72: Add a new word.



Glossary Definition

Glossary

CLONE

DELETE

Select a Glossary from the combo box to show its word hierarchy structure. You can edit it by adding nodes or by dropping words from the right side list

Glossary Definition

KNOWAGE-1922

Code

KNOWAGE-1922

Edit the input to change the glossary name

Description

KNOWAGE-1922

Words

ADD NODE

search

KNOWAGE-1922NODE

KNOWAGE-1922WORD

KNOWAGE-1922NODE1

KNOWAGE-1922NODE2

Fig. 2.73: Exploring a glossary from the menu.

Fig. 2.74: New glossary wizard.

how to add words to a node once created.

Fig. 2.75: Add items to the node(s) and words to nodes.

2.7.2 Glossary Usage

This functionality works accordingly to the user role and includes features that allow to create/visualize the associations of the words of a glossary to:

- documents,
- datasets,
- business classes and
- tables (columns).

The details of all the defined glossaries can be displayed just selecting **Glossary Usage** from *CATALOGS* of the Knowage main menu.

Select a glossary from the combobox available and search for a word inside the glossary. The page refreshes showing the links of that word with the components mentioned above.

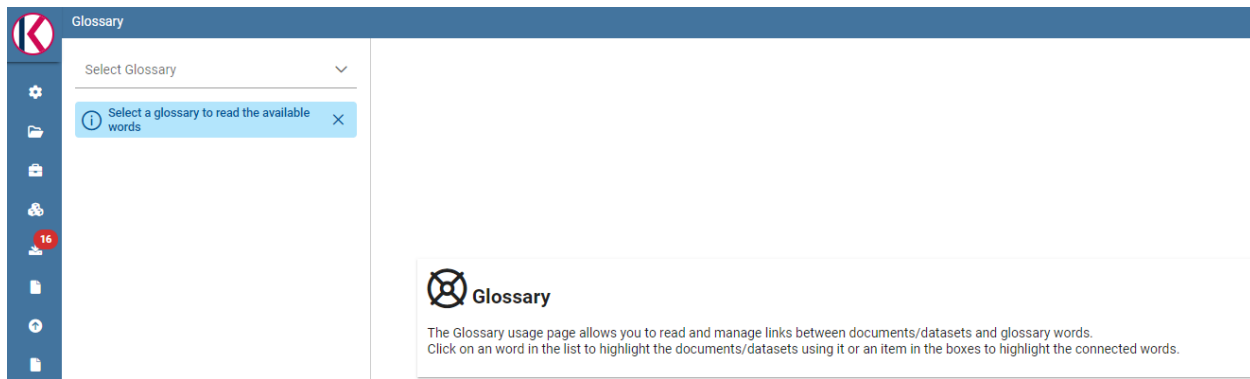


Fig. 2.76: Glossary Usage graphic interface.

In case of no links with i.e. a Table, a message prompts informing that there are no links to the word.

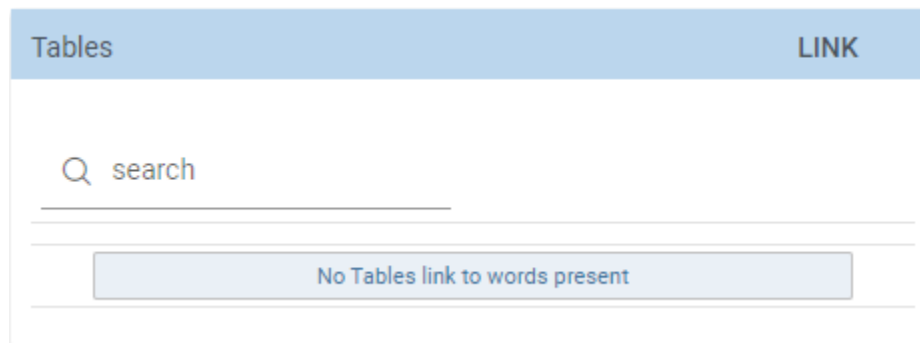


Fig. 2.77: No table links to words.

To associate the glossary to a specific component, just click on the *LINK* functionality next to each element. See figure below as example.

After clicking on *LINK*, a wizard opens. Here you can select the glossary and to associate the word it is enough to drag and drop it in the editor area of the chosen dataset. After closing the wizard, clicking on the *i* icon you can see the list of words associated. See image below:

2.7.3 Help Online functionality

The user can view the association of a specific analytical element (dataset, document or model) by using the **Help Online** functionality from:

- the Document Browser,
- the toolbar of each document, once launched,
- every dataset,
- every entity of the Qbe model,
- Birt reports,
- the cockpit.

As an example, the figure below shows the graphic interface that the user will see once launched a document and used the Help Online functionality.

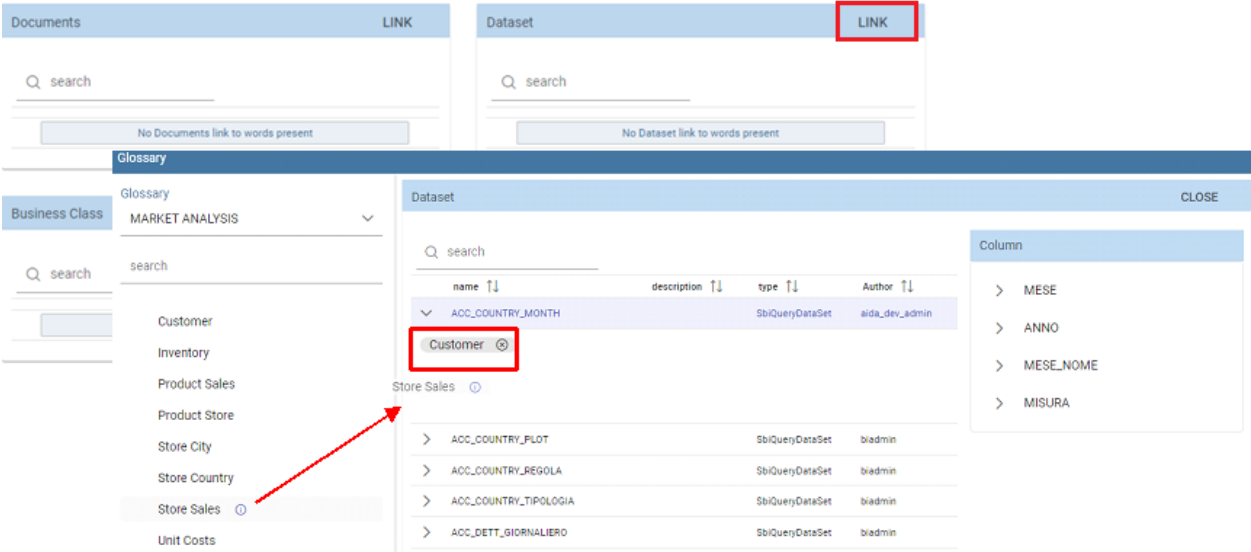


Fig. 2.78: Linking a word to a dataset .

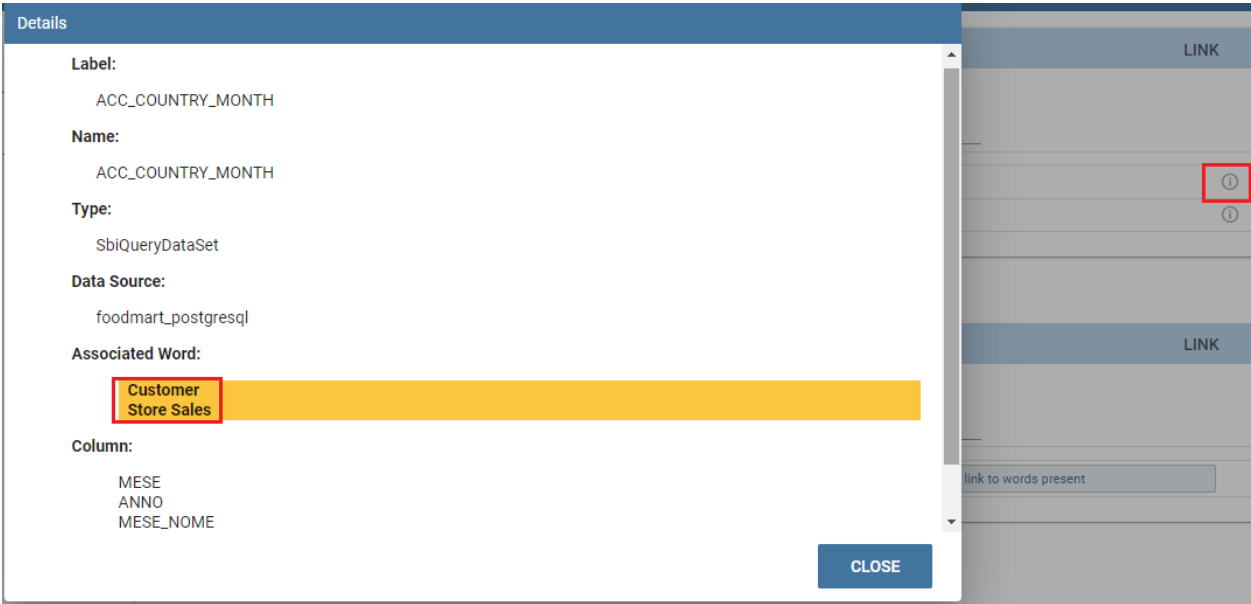


Fig. 2.79: Linking a word to a dataset .

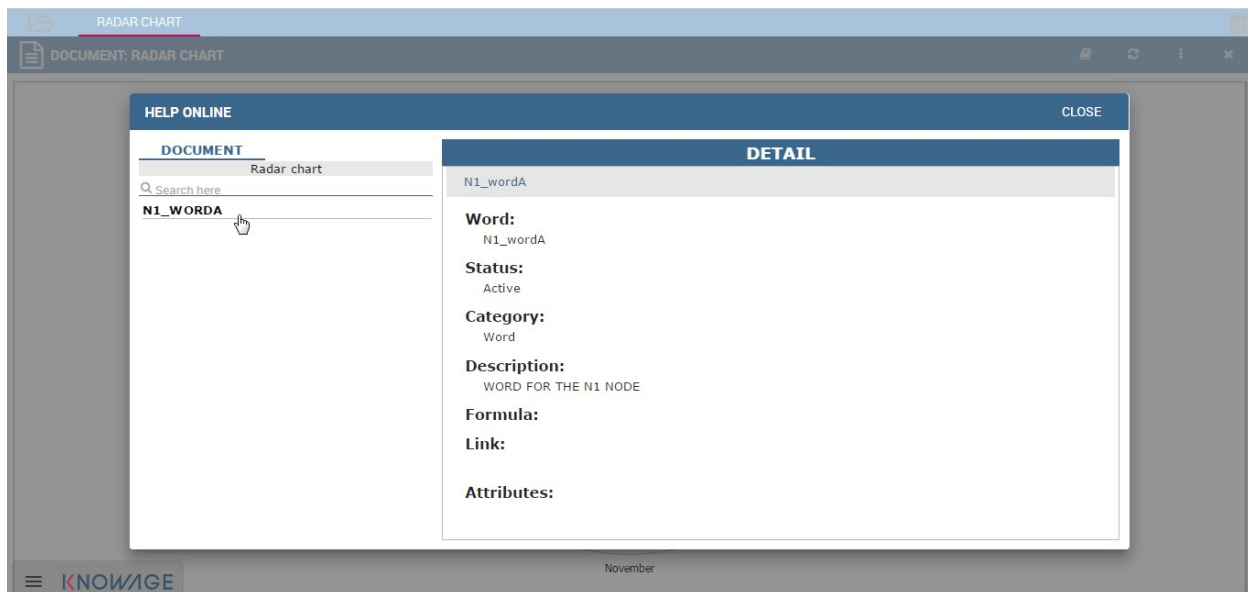


Fig. 2.80: Help Online wizard.

2.8 Resource manager

Important: Enterprise Edition only

The Knowage Community Edition only allows the management of the *models* folder, in order to define metadata for data mining analysis. The Knowage Enterprise Edition allows the management of all the Knowage installation resources.

The **Resource Manager** functionality available under the **TOOLS** section of the Knowage main menu, allows the management of all files within the *Resources* folder of the Knowage installation.

By default, the system shows the files starting from the root directory:

2.8.1 Resource Manager functionalities

The Resource Manager window displays two sections: on the left the tree structure representing the existing resource system; on the right the details of the selected folder.

Tree functionalities

The tree shows exactly the physical structure of the file system from the **resources** folder. On the top bar are present two functionalities: **Refresh tree** and **Create new folder**:

The new folder will be created under the selected folder of the tree (in this case, 'KNOWAGE-5058').

By clicking on the *Download* icon of a specific folder, you can download all files contained:

Select a folder for the download and give a name to the zip file to be available in your local system.

By clicking on the *Delete* icon of a specific folder, the folder can be physically removed from the Knowage server after confirming the operation. Please remember that the deletion of a folder is an irrecoverable operation.

Detail panel

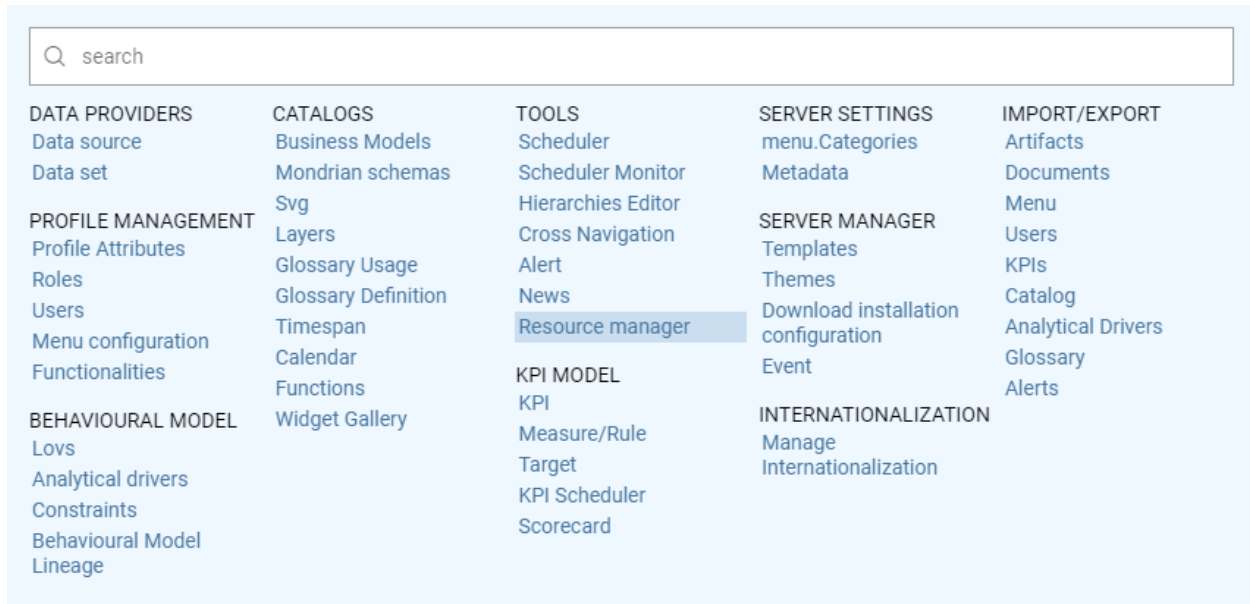


Fig. 2.81: Resource Manager from menu.

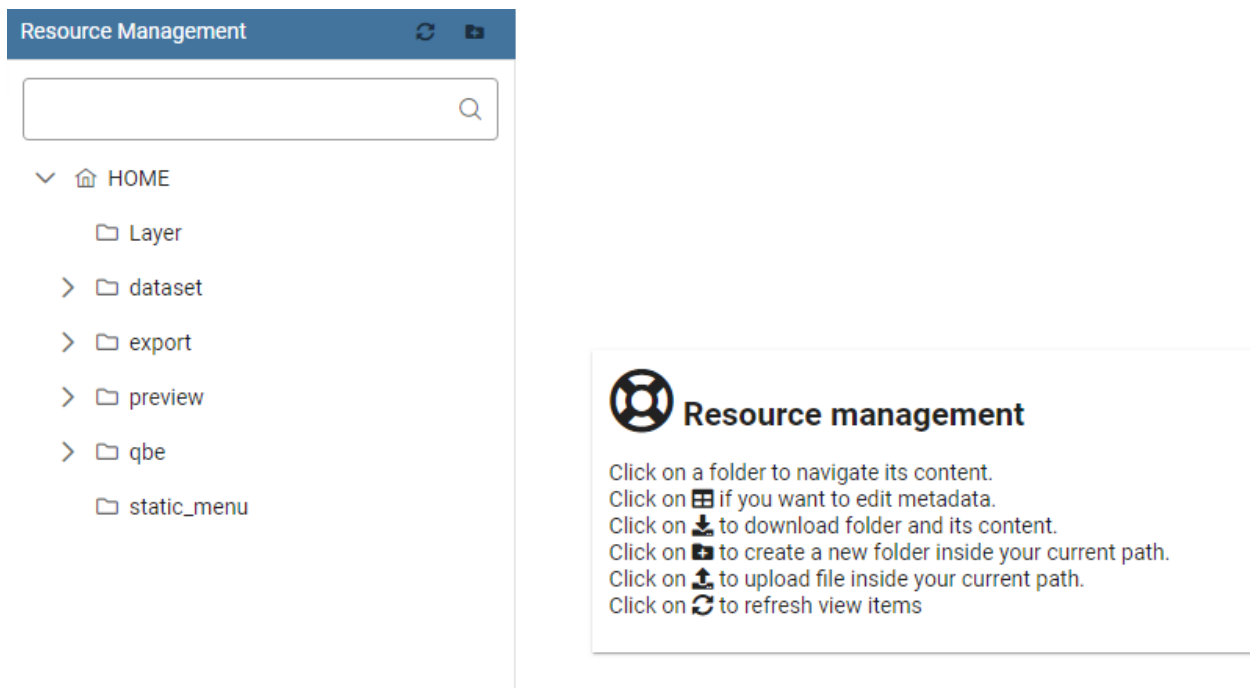


Fig. 2.82: Resource Manager starting view.

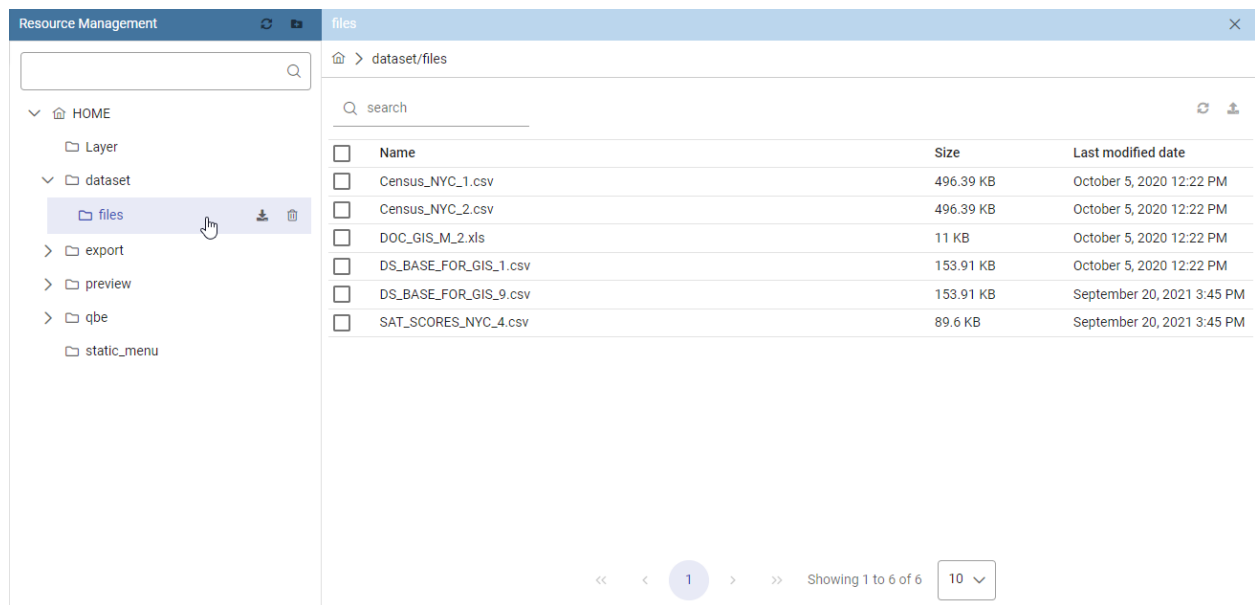


Fig. 2.83: Resource Manager detail view.

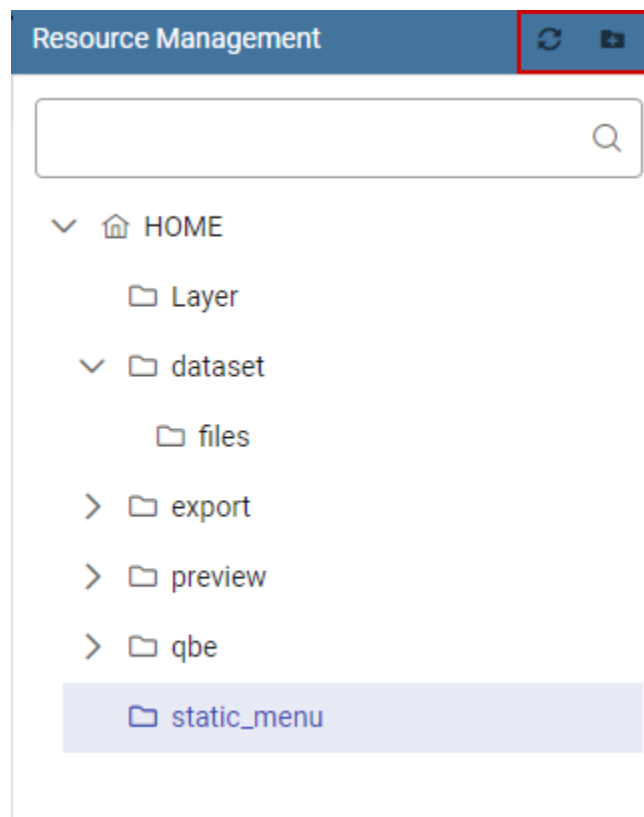


Fig. 2.84: Tree view.

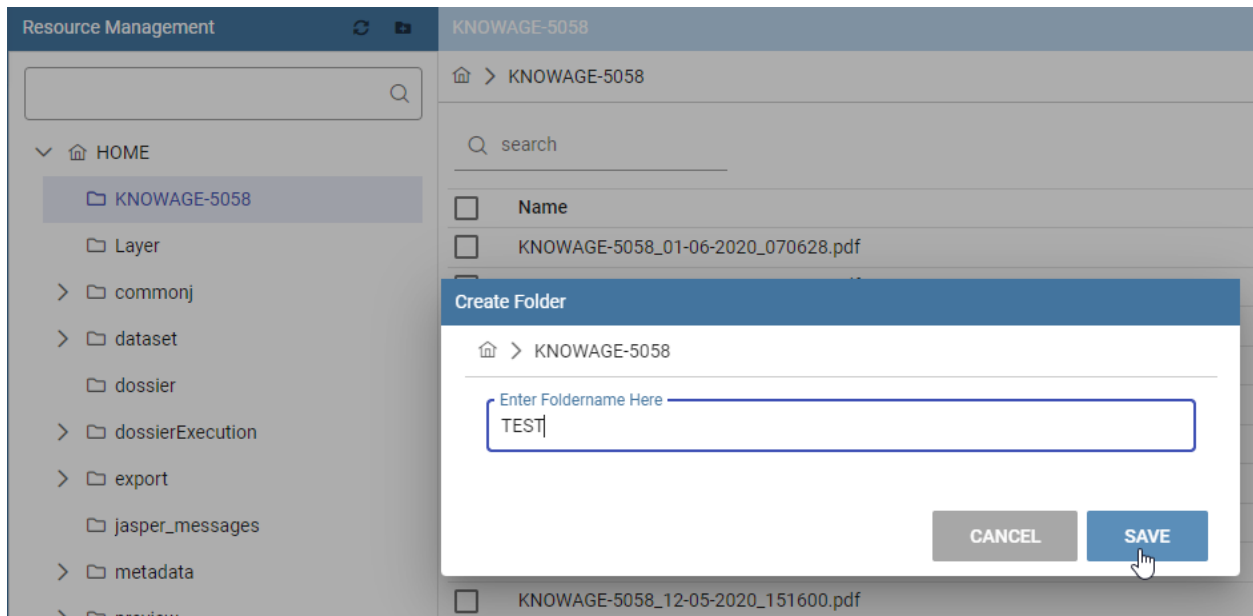


Fig. 2.85: Creation detail.

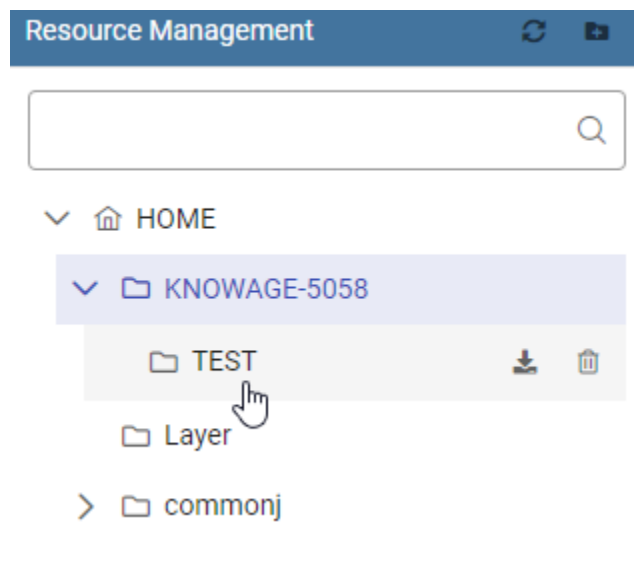


Fig. 2.86: Creation result.

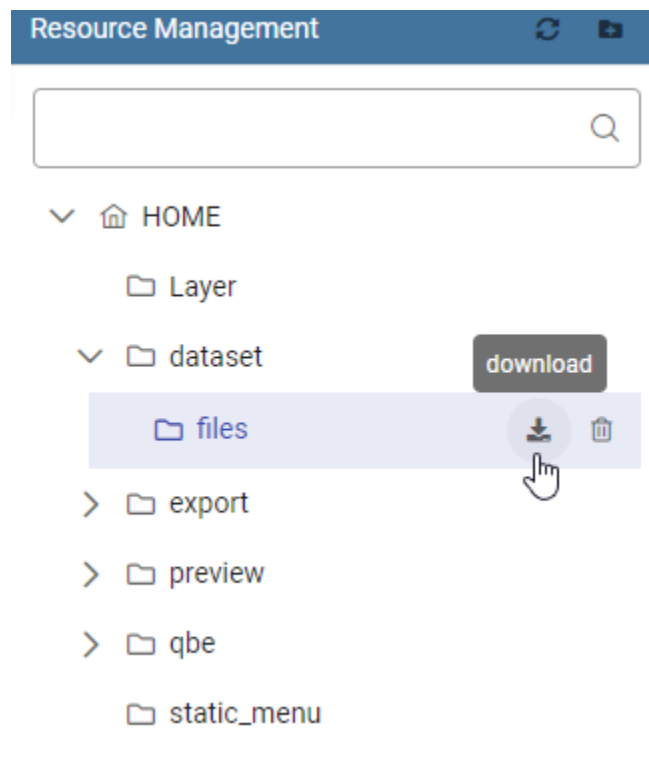


Fig. 2.87: Download functionality icon.

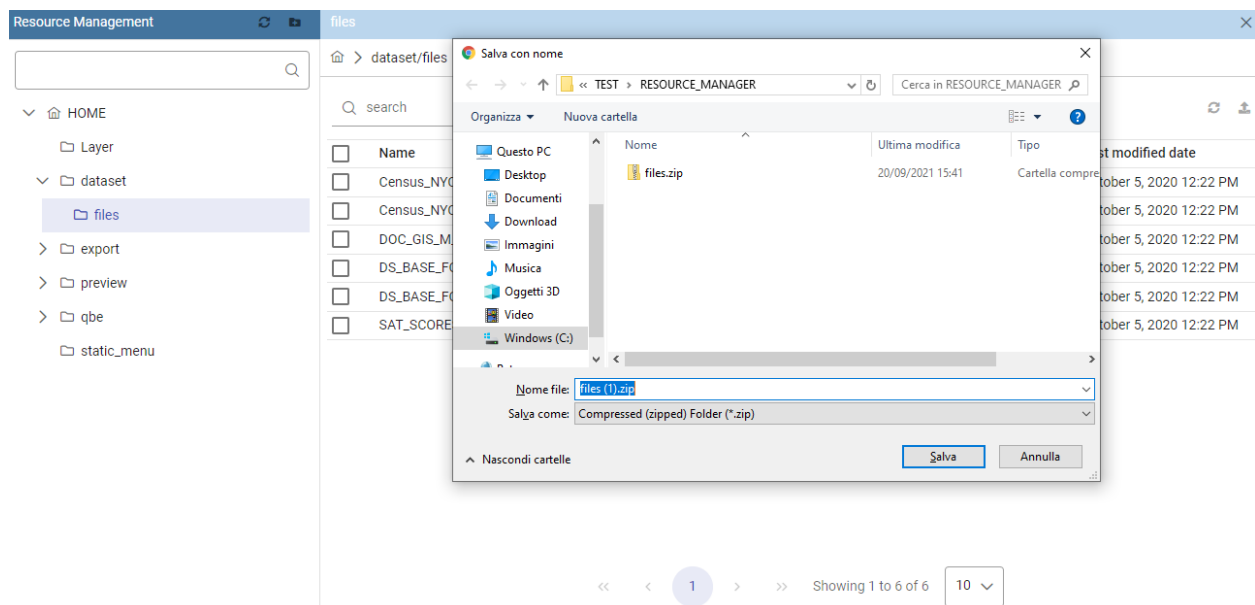


Fig. 2.88: Download functionality.

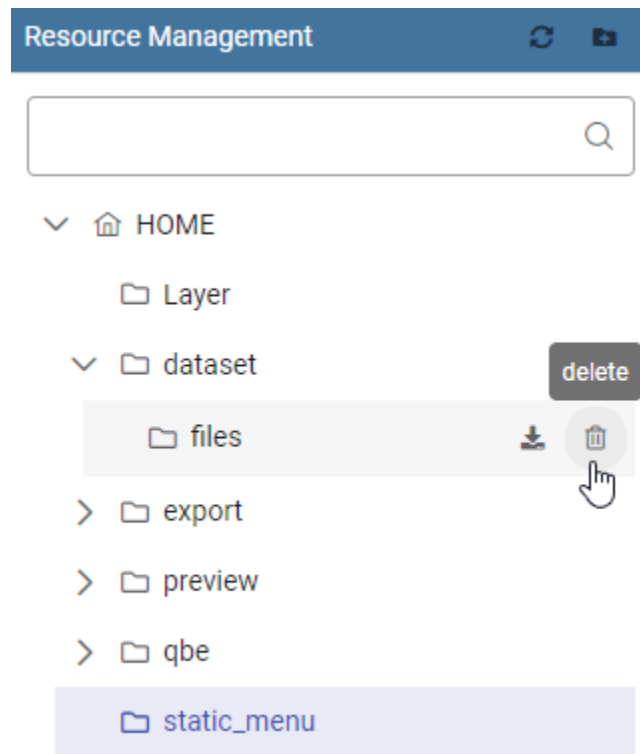


Fig. 2.89: Delete functionality icon.

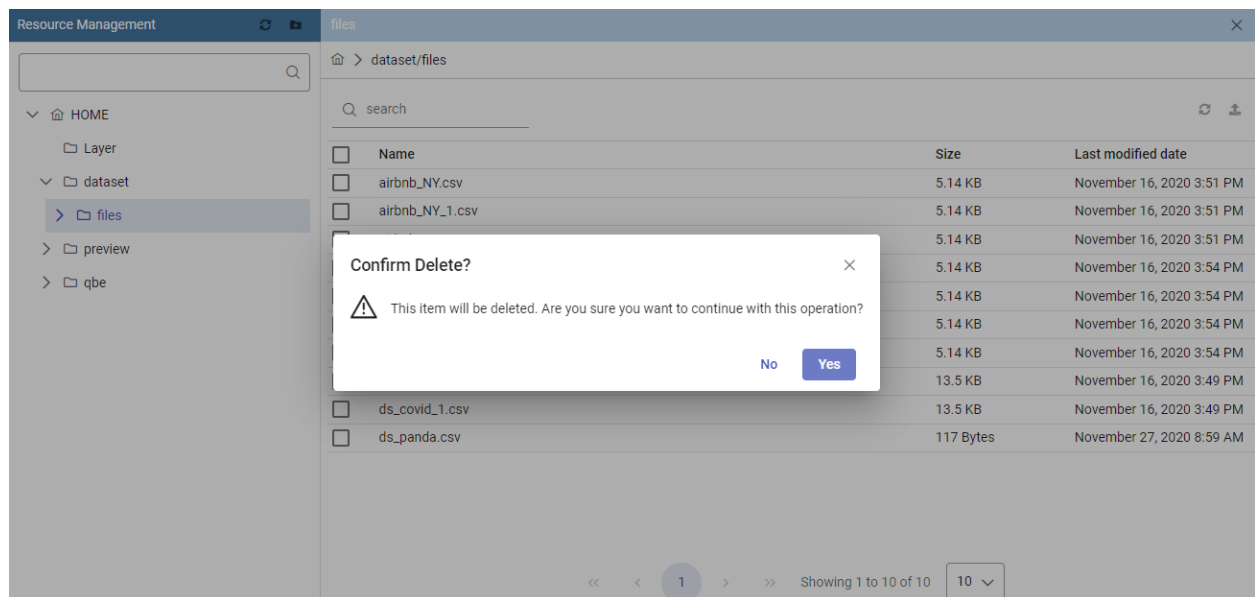


Fig. 2.90: Delete functionality confirmation.

When a folder is selected, the right panel shows the properties *Name*, *Size* and *Last Modified Date* of all contained files.

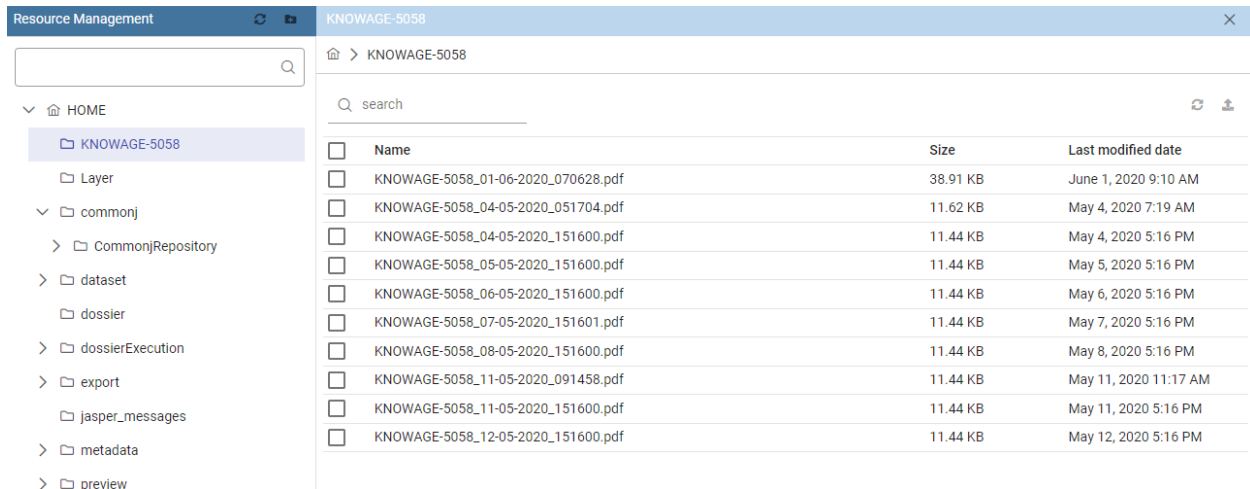


Fig. 2.91: Detail view.

All items are selectable for downloading or removal trough specific icons of the toolbar.

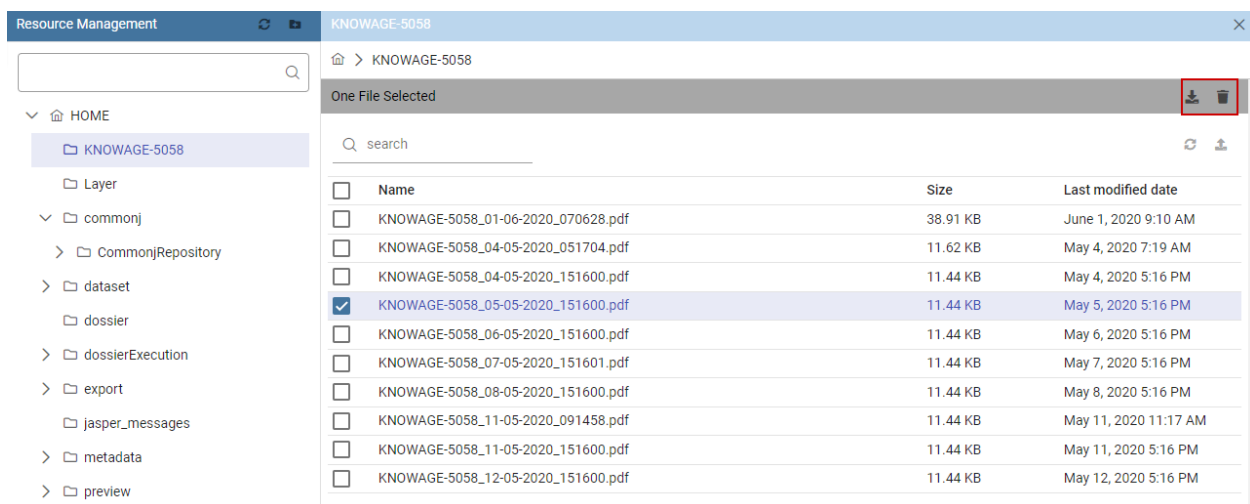


Fig. 2.92: Functionalities available for selected files

An **Upload** functionality is also available through a dedicated icon. See image below.

In case of a zipped file want to be uploaded, it is possible to keep the file zipped.

It is also possible to unzip the file to upload by enabling the *Extract Files* option. See figure below.

Model Metadata Definition

As already told at the beginning, **models** is the unique folder managed by both the Community and the Enterprise Edition. It contains all the data-mining models usable by the Knowage Function Catalog.

For each model it is possible to define its metadata, download and/or delete the model using directly the tree options:

Metadata management

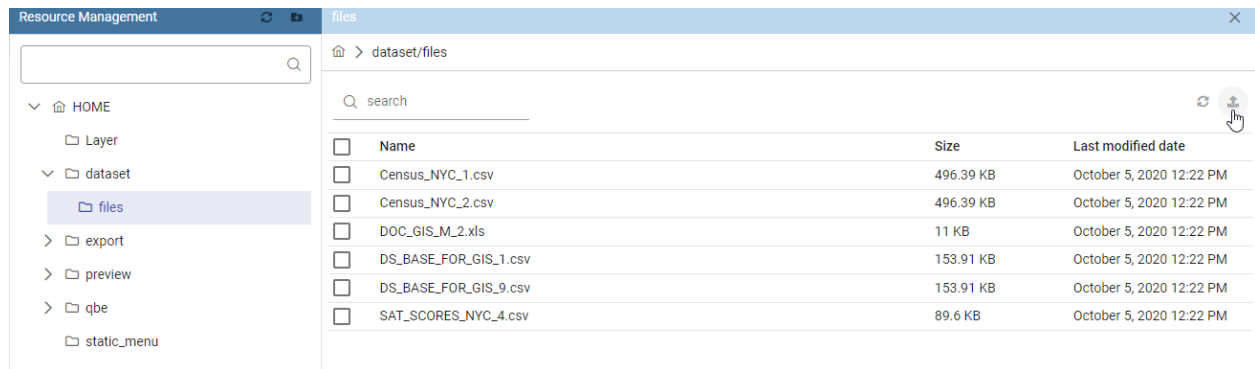


Fig. 2.93: Upload files

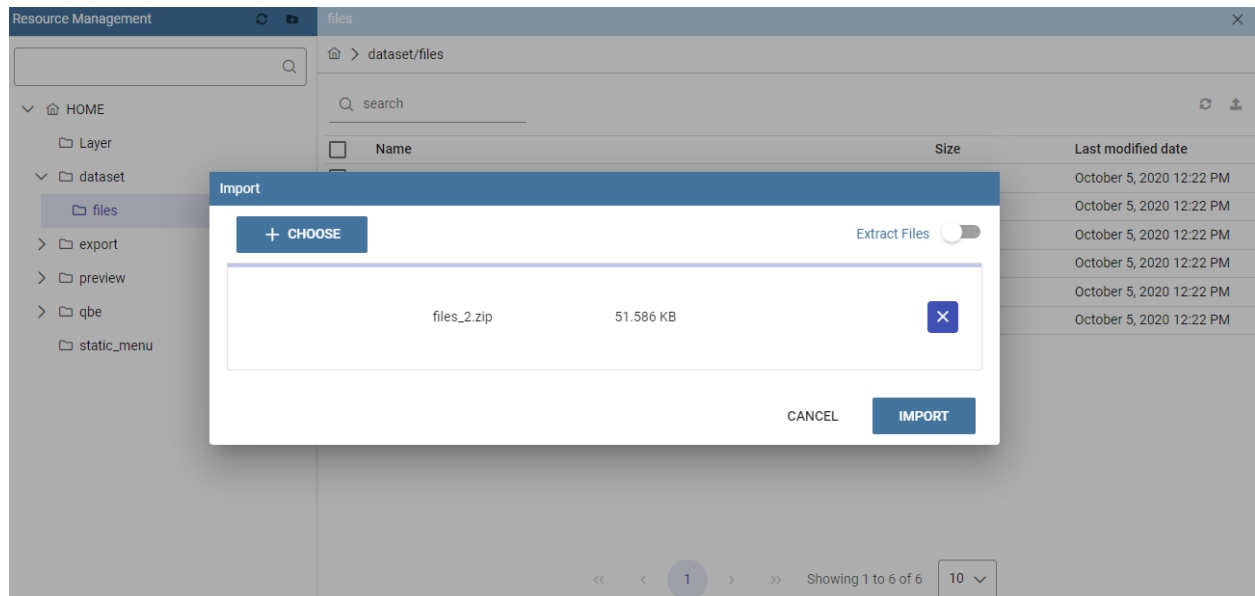


Fig. 2.94: Selection file popup

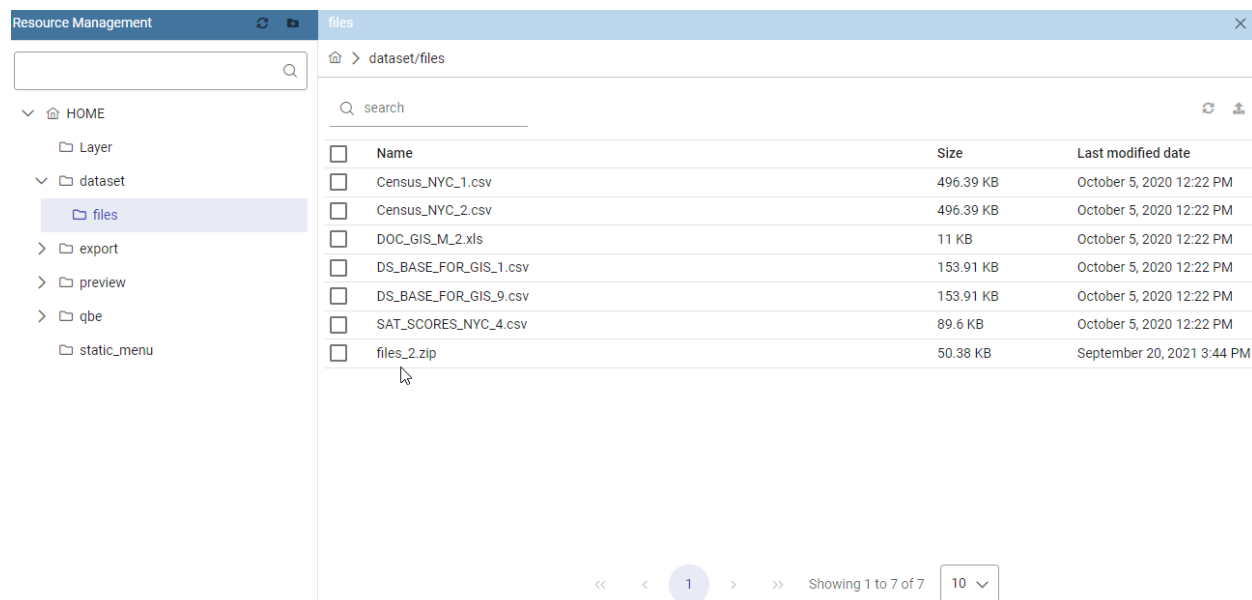


Fig. 2.95: Uploaded zipped file

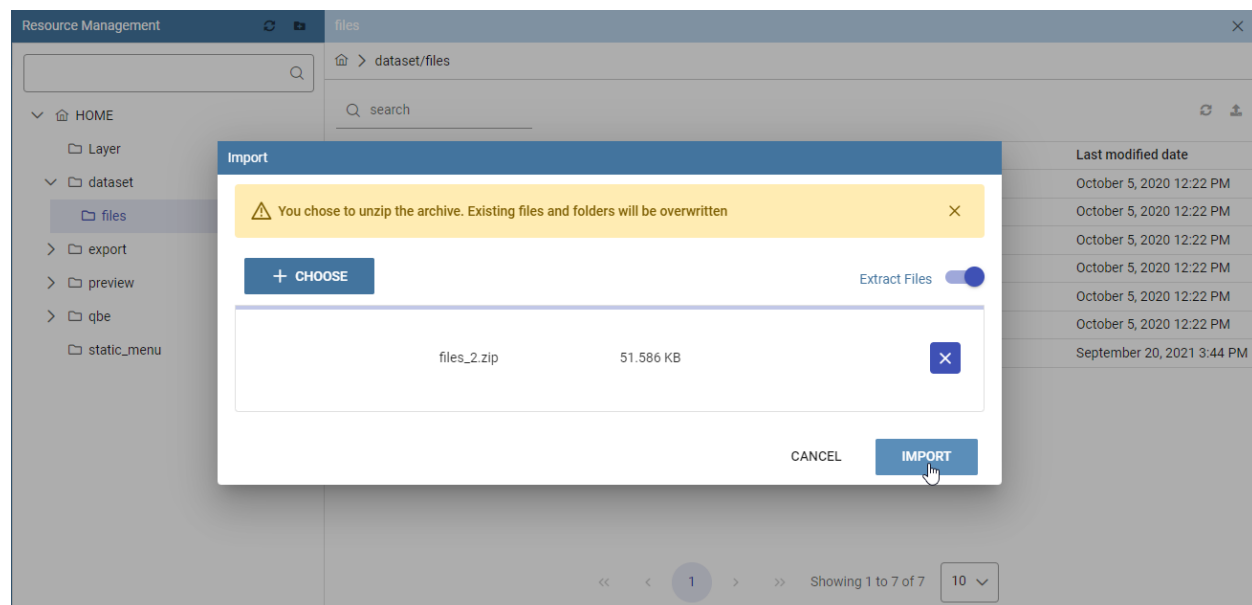


Fig. 2.96: Uploading an unzipped file

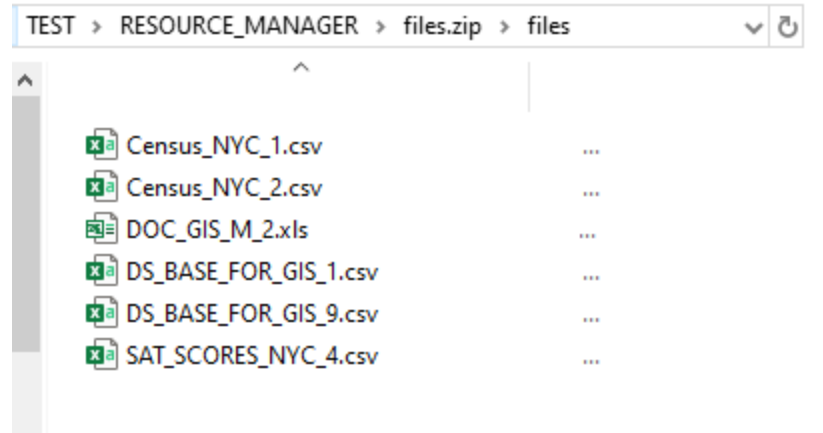


Fig. 2.97: Results of uploading an unzipped file

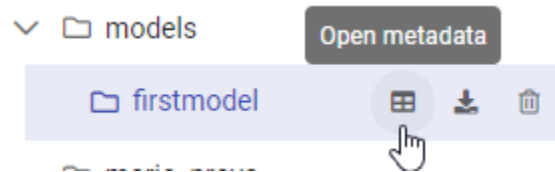


Fig. 2.98: Models folder options

The **Metadata** option opens a GUI where the user can define the metadata information for the model, into the specific see image below:

2.9 Widget gallery

The **Widget Gallery** is a feature available since version 8.0 that allows users and editors to create a template and share it in multiple dashboards. By creating a template and then using it within a dashboard, users will have the ability to create complex dashboard elements quite easily and quickly while maintaining a common basic template.

This functionality is available for the following types of widgets:

- HTML
- Custom Chart
- Python

2.9.1 Gallery management

To open the *Gallery Management*, select **Gallery Management** from the *CATALOGS* option of the Knowage main menu. The first step consists in creating a new template or importing a template not available in the list.

The image below, shows the information to be filled in when adding a new template.

Using import you will see a dialog to choose the template to be imported. Clicking “import” it will be added to the current list.

A **Template** is a json file containing a collection of properties describing the widget and the code composing it.

The following fields will be present:

Metadata

Name	Version	Type Of Analytic To Be Executed	Is Open Source
Metadata about a Model Example	2	Descriptive	<input checked="" type="checkbox"/>

Description

This is an example for the metadata definition of a data-mining model...

Accuracy and performance

In this area, user can describes the accuracy level and performance notes...

Usade of the model

CLOSE

SAVE

Fig. 2.99: Metadata example

Gallery Management

search

+ New template

Import template

Card15

Card16

Card24

Card25

Card26

Card29

Gallery management

The templates allow you to access ready-made elements to proceed more quickly to the creation of dashboards. To view the details of a template, select one of those on the side, or press the "+" button to add or import one.

Fig. 2.100: Gallery management example.

Template

Label

Name

Type

HTML

HTML

Chart

Python

Description

Tags

Allowed values for tags: uppercase and lowercase letters, numbers, '-', '_'. The space character is not allowed.

HTML

CSS

1

Fig. 2.101: Widget - new template.

- **Name:** Mandatory information, representing the name of the widget template
- **Type:** Mandatory information, specifying the widget type. As shown in the above image, there are three available types
- **Output type:** only for Python widgets: HTML or Image values available.
- **Description:** Optional information that will be visible as a tooltip on the dashboard selection.
- **Tags:** Optional information, consisting in a list of unique tags to easily categorize templates. Search functionality using tags too. Allowed values for tags are uppercase and lowercase letters, numbers, '-', '_' whereas the space character is not allowed.
- **Image:** Optional information, representing the image of the widget that will be shown on the dashboard selection. The maximum image size is 200k.
- **Code section:** Mandatory information, representing the code to be written in the editor box.

Editors will look like different, depending on the type of widget: - **HTML **:** HTML, CSS editors - ****Chart:** HTML, CSS and JS editor - **Python:** Python code editor

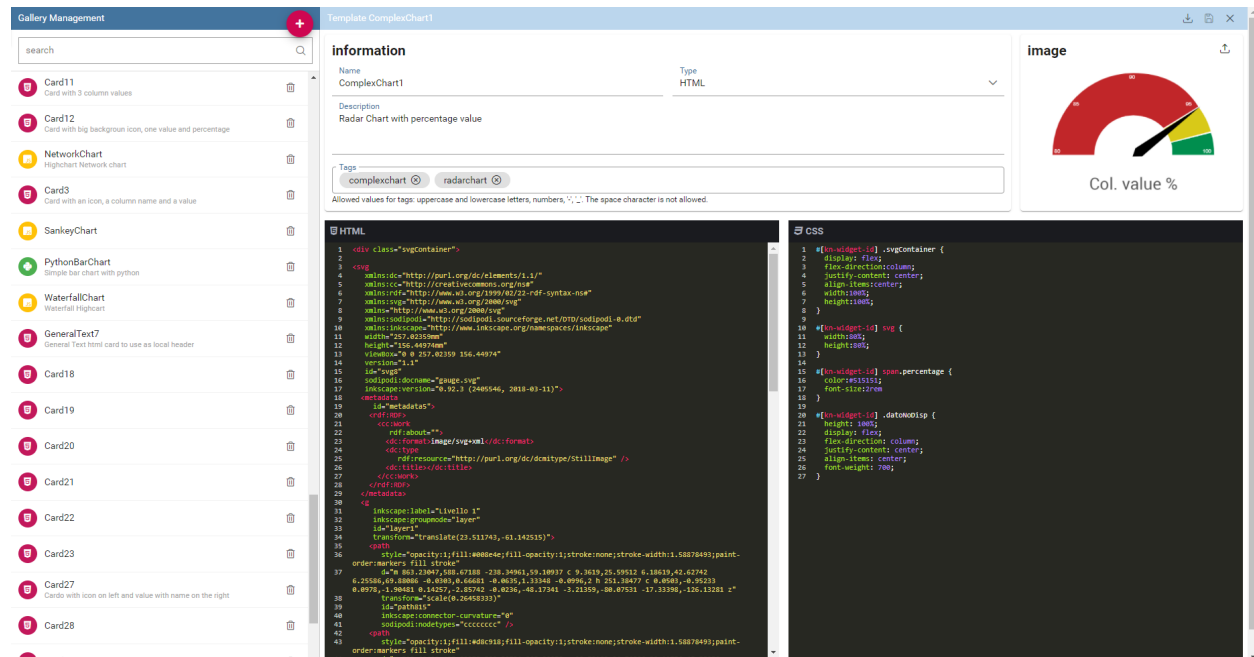


Fig. 2.102: Selected widget template.

Use the *Save* icon to save the template.

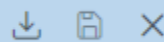


Fig. 2.103: Toolbar icons.

2.9.2 Dashboard gallery

When adding a widget in a dashboard (cockpit document), the wizard shows a *Gallery* tab, containing a first empty template and then a set of available templates showing their image, tags and eventually the description. The empty

template allows users to create a custom widget without starting from an already created template. The *Gallery* tab would not be available if any templates were not already formerly saved and therefore available to be used.

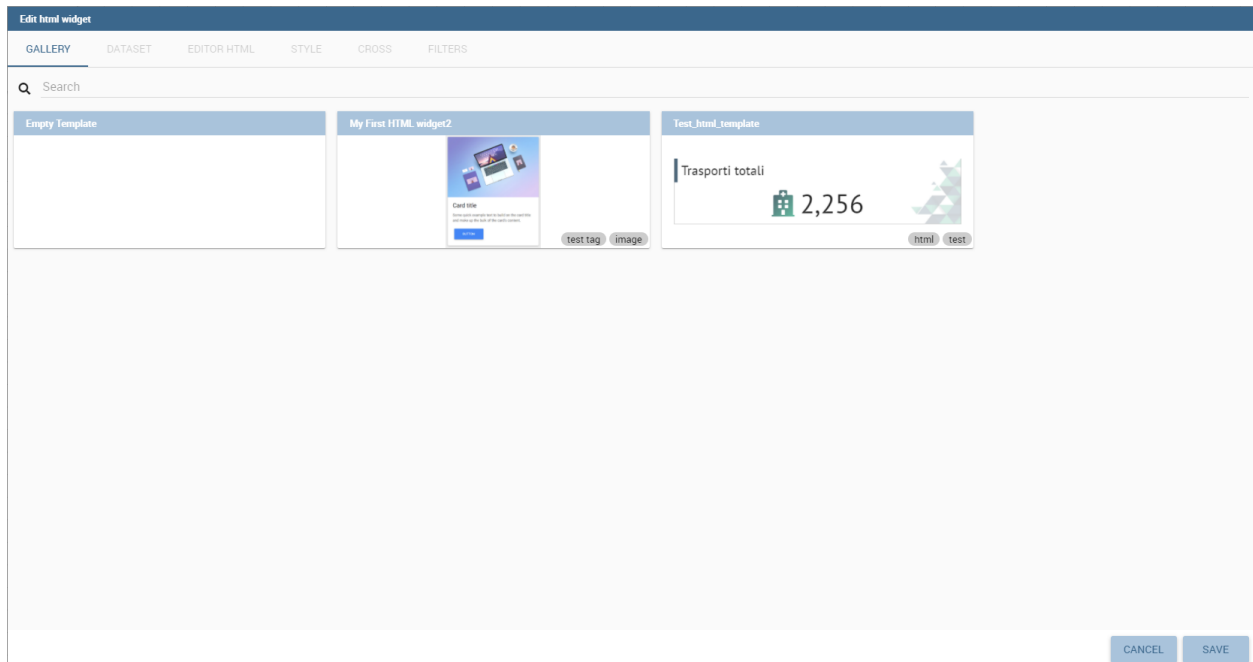


Fig. 2.104: New widget templates list.

Clicking on a given template, the code is automatically copied in the new widget template. The user just needs to make some changes some to the code to customize and create the desired widget.

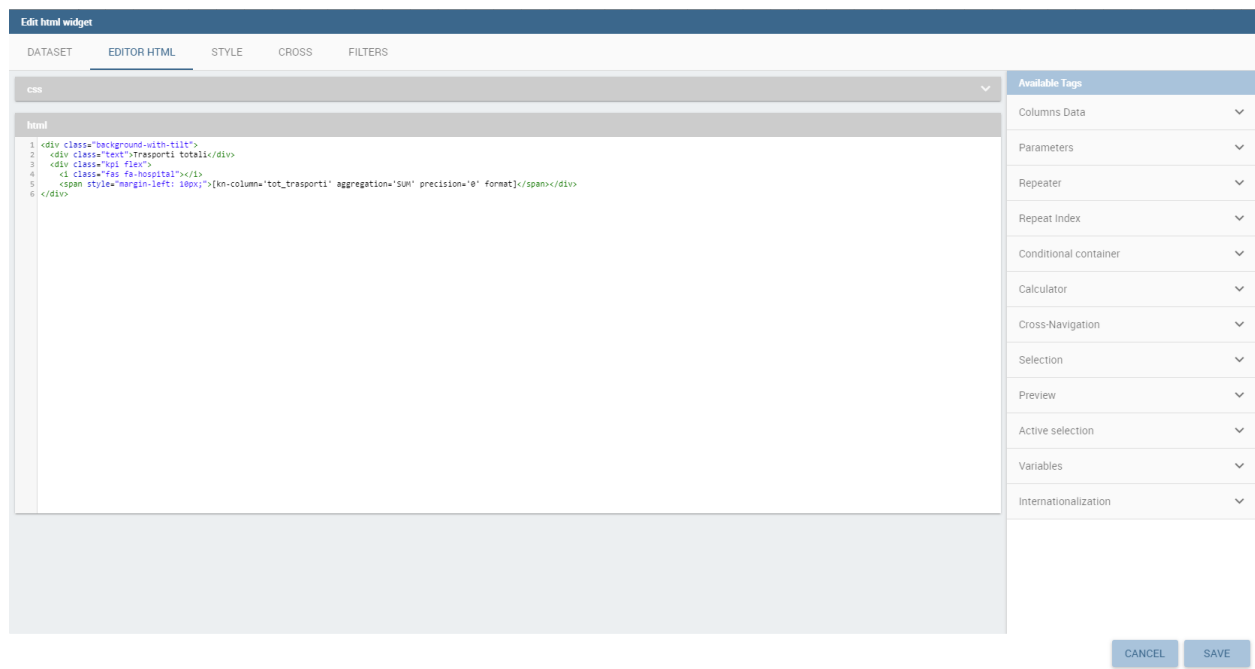


Fig. 2.105: Selected template editor.

Installation and configuration

The present document illustrates how to install and configure the Knowage suite. There will be described the essential steps to succeed with the standard installation on certified environments, which includes the optional use of CAS as a SSO solution and the use of HTTPS protocol.

3.1 Requirements

Before going into details on Knowage installation, it is necessary to check if certain requirements are satisfied. We start to distinguish between the certified environments and the compatible ones. The first are those where check tests take place. The latter are those environments technically compatibles but where integration tests are not executed.

3.1.1 Operating systems

The following Operating Systems (OS) are those ones which suit with Knowage platform.

Table 3.1: Certified environments

Certified Environments	
Operating System	Version
CentOS	7, 8
Windows	7, 10

Table 3.2: Compatible environments

Compatible Environments	
Operating System	Version
RHEL Red Hat Enterprise	7
Ubuntu	22.04 LST
Windows server	2019, 2012, 2008

3.1.2 Disk usage

The Knowage installation requires 2 GB of available space on file system. This space does not include the space relative to the data and the metadata storage.

3.1.3 Java environment

The environment in which Knowage will be installed must include a JDK 1.8 installation. Be sure that the JDK component is successfully installed and that the environment variable `JAVA_HOME` is properly configured. The steps to configure it depend on the OS. Knowage is compatible with Open JDK 1.8.

3.1.3.1 Linux

Define the `JAVA_HOME` variable inside the users' file `.bash_profile` used in the installation process

Listing 3.1: Instructions to set the `JAVA_HOME` variable for Linux environment.

```
export JAVA_HOME=<root path of the Java installation>
export JAVA_HOME=/usr/lib/jvm/jdk1.8.0_60/
export PATH=$JAVA_HOME/bin:$PATH
```

3.1.3.2 Windows

Define the `JAVA_HOME` variable and `PATH` in the section “Environment variables” which can be reached from the “System”.

3.1.4 Application server

The following lists the supported application servers:

Table 3.3: Supported application servers

Support type	Application Server	Version
Certified	Apache Tomcat	9

For each application server installation please refer to its official documentation.

3.1.4.1 Tomcat 9.0

In the following we will refer to Tomcat installation folder as `TOMCAT_HOME`.

3.1.4.1.1 Tomcat on Linux

It is recommended to create a proper user for the execution of Tomcat. We state the main steps to follow for this purpose.

- Create the Tomcat user.

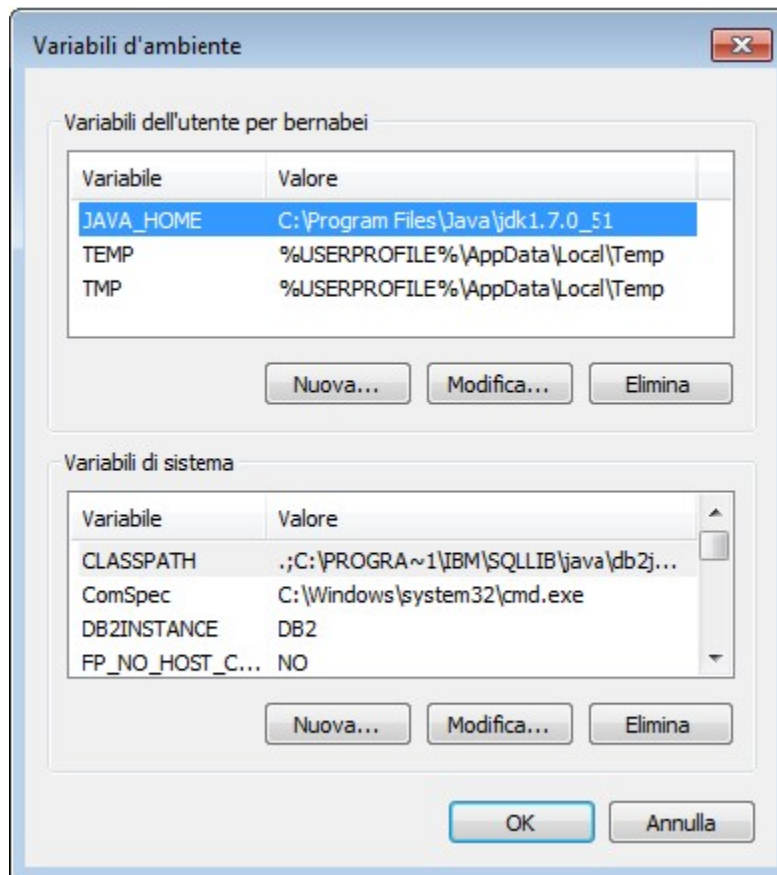


Fig. 3.1: Setting the path for the JAVA_HOME variable for Windows

```
useradd -m tomcat
passwd <password for the tomcat user>
```

- Install the Tomcat using the Tomcat user. Remember to define the TOMCAT_HOME variable.

```
export TOMCAT_HOME=<path of the installation Tomcat root folder >
```

- Be sure that the Tomcat uses the JDK 1.8: usually the Tomcat settings are defined in the TOMCAT_HOME/bin/setenv.sh file, therefore if the TOMCAT_HOME/bin/setenv.sh file does not exist, the user must create it and insert it in the content as shown below. Note that CATALINA_PID contains the ID of the Tomcat process and it kills the process if needed.

```
export CATALINA_PID=<root folder of the Tomcat installation>/logs/tomcat-knowage.pid
export JAVA_HOME=<root folder of the JDK 1.8 installation>
```

- Modify the TOMCAT_HOME/bin/shutdown.sh file to force the shut down of the application in case of hanging:

```
exec "$PRGDIR"/"$EXECUTABLE" stop -f "$@"
```

3.1.4.1.2 Tomcat on Windows

It is recommended to install Tomcat as a service. Documentation is available at <https://tomcat.apache.org/tomcat-9.0-doc/windows-service-howto.html>.

3.1.5 Database schema for metadata

Knowage uses a schema to manage metadata, that is all those information required for its operation. These concern the configuration, the users and the analytical documents. It is possible to use the following DBMSs for the creation of this schema.

Table 3.4: Exploitable DBMSs for the metadata schema creation

Support Type	DBMS	Version
Certified	Oracle	8, 9, 10, 11, 12
Certified	MySQL	5.7, 8.0
Certified	PostgreSQL	8.2, 9.1, 12.3
Certified	MariaDB	10.1, 10.2, 10.3

Therefore, a schema must be available. It can be reached through the JDBC protocol by the Knowage installation server; such a schema will be called *metadata DB* in the following. Observe that Knowage includes all the DDL for table creation.

3.1.6 Database schema for data

A schema for data must be also available. It can be queried through Knowage and can be reached through the JDBC protocol by the Knowage installation server; such a schema will be called *data DB* in the following.

3.1.7 NodeJS requirements

Important: Enterprise Edition only

NodeJS is required only for Enterprise Edition.

Knowage includes some NodeJS scripts that need to be executed with NodeJS 14 or greater: see [NodeJS official documentation](#) for the installation process.

3.1.7.1 CentOS

In CentOS you need to erase older versions of NodeJS, if present:

Listing 3.2: Command to erase older versions of NodeJS

```
yum erase -y nodejs
```

Then you need to clear YUM cache and update all local packages:

Listing 3.3: Cache clearing and system updating

```
yum clean all  
yum update -y
```

Next you can install the official repository of NodeJS:

Listing 3.4: Installation of the repository of NodeJS

```
curl -sL https://rpm.nodesource.com/setup_14.x | bash -
```

Important: If you are behind a corporate proxy, you would need to set `http_proxy` and/or `https_proxy`.

Finally you can install NodeJS:

Listing 3.5: Installation of NodeJS

```
yum install -y nodejs
```

3.1.7.2 Ubuntu

In Ubuntu you need to erase older versions of NodeJS, if present:

Listing 3.6: Command to erase older versions of NodeJS

```
apt-get remove nodejs
```

Then you need to clear APT cache and update all local packages:

Listing 3.7: Cache clearing and system updating

```
apt-get update  
apt-get upgrade -y
```

Next you can install the official repository of NodeJS:

Listing 3.8: Installation of the repository of NodeJS

```
curl -sL https://deb.nodesource.com/setup_14.x | bash -
```

Important: If you are behind a corporate proxy, you would need to set `http_proxy` and/or `https_proxy`.

Finally you can install NodeJS:

Listing 3.9: Installation of NodeJS

```
apt-get install -y nodejs
```

3.1.8 Chromium requirements

Important: Enterprise Edition only

Chromium is required only for Enterprise Edition.

Knowage provides a distribution of Chromium for its functionalities but some other dependencies are needed. In Linux distribution you need to install following Chromium dependencies:

Listing 3.10: Installation of Chromium dependencies

```
# For CentOS 7
yum install -y at-spi2-atk cups-libs expat glib2 glibc.i686 glibc libcanberra-gtk3 libgcc libstdc++ libX11.
↳ libXScrnSaver minizip nspr nss-mdns nss-util nss polycoreutils-python polycoreutils zlib

# For CentOS 8
dnf install -y libX11 libX11-xcb libXcomposite libXcursor libXdamage libXext libXi libXtst nss libXScrnSaver.
↳ libXrandr alsa-lib atk at-spi2-atk pango gtk3 libgbm

# For Debian/Ubuntu
apt-get install -y libgbm1 libxss1 libgtk-3-0 libasound2 libatk-bridge2.0-0 libatk1.0-0 libatspi2.0-0 libc6.
↳ libcairo2 libcups2 libdbus-1-3 libexpat1 libgcc1 libgdk-pixbuf2.0-0 libglib2.0-0 libnspr4 libnss3 libpango-1.
↳ 0-0 libpangocairo-1.0-0 libuuid1 libx11-6 libx11-xcb1 libxcb1 libxcomposite1 libxcursor1 libxdamage1.
↳ libxext6 libxf86dev libxi6 libxrandr2 libxrender1 libxtst6 bash

# For RedHat 7
yum install -y pango.x86_64 libXcomposite.x86_64 libXcursor.x86_64 libXdamage.x86_64 libXext.x86_64 libXi.x86_
↳ 64 libXtst.x86_64 cups-libs.x86_64 libXScrnSaver.x86_64 libXrandr.x86_64 GConf2.x86_64 alsa-lib.x86_64 atk.
↳ x86_64 gtk3.x86_64 ipa-gothic-fonts xorg-x11-fonts-100dpi xorg-x11-fonts-75dpi xorg-x11-utils xorg-x11-fonts-
↳ cyrillic xorg-x11-fonts-Type1 xorg-x11-fonts-misc
```

3.1.9 Support to non-latin languages

Knowage does some of its job at server side and it could need support for non-latin languages. Some operating systems don't provides support to non-latin language by default: see the official documentation to enable the support to those languages.

For example, to install non-latin languages fonts you could use:

Listing 3.11: Installation of non-latin language fonts

```
# For CentOS 7
yum groupinstall fonts

# For Ubuntu
sudo apt-get install language-pack-ja
sudo apt-get install japan*

sudo apt-get install language-pack-zh*
sudo apt-get install chinese*

sudo apt-get install language-pack-ko
sudo apt-get install korean*

etc...
```

3.1.10 Supported browsers

Knowage supports the newest and the second to last version of these browsers:

- Google Chrome
- Firefox
- Microsoft Edge

Important: Internet Explorer

Internet Explorer is no longer supported by Microsoft and it is also vulnerable. Please, use one of the supported browser listed above.

3.1.11 Data Preparation requirements

In order to use data preparation functionality, user should have Apache Livy and Apache Spark installed. Please check Livy and Spark official documentation for more info (<https://livy.apache.org/>, <https://spark.apache.org/>) and manual installation paragraph for technical details.

This functionality is available as an add-on plugin for Smart Intelligence license.

3.2 KNOWAGE Installation

The present document illustrates how to install and configure the Knowage suite. There will be described the essential steps to succeed with the standard installation on certified environments, which includes the optional use of CAS as a SSO solution and the use of HTTPS protocol.

3.2.1 Software release

You can download Knowage through the following links:

- Community Edition: <http://www.knowage-suite.com/site/ce-download/>
- Enterprise Edition: <http://www.knowage-suite.com/portal> (registration required)

A typical release contains three elements:

- Installer, which gives you a step-by-step Knowage installation
- Web Application Archives (WARs) and DDL scripts (to populate the schema used by Knowage for its metadata)

Table 3.5: Supported databases for DDL scripts

Supported databases
MySQL / MariaDB
Oracle
Postgres

3.2.1.1 Optional: using MAC address for license

If you're using a VM on AWS cloud and you want to use Knowage EE with license, you should be sure that hardware id doesn't change every start and stop. To avoid this behaviour you can choose for using MAC address instead of default hardware id calculation. You just have to add this optional row in "setenv.sh" file:

```
export JAVA_OPTS="$JAVA_OPTS -Dmac.address.licensing=true"
```

3.2.2 Manual installation

3.2.2.1 Metadata database initialization

Knowage requires a database schema to store its own metadata (definition of analyses, datasets and so on). For initializing such a schema, the user must execute the creation scripts provided for the DBMS in use. The package which includes the DDL will contain the following SQL files:

Listing 3.12: Scripts for metadata schema

```
XXX_create.sql
XXX_create_quartz_schema.sql
```

where XXX represents the DBMS type (as instance ORA stands for Oracle). The corresponding SQL files for deleting tables are also provided.

3.2.2.2 Using Tomcat

3.2.2.2.1 Dependencies

You must add required libraries into TOMCAT_HOME/lib folder:

- the JDBC connector for the metadata database with its dependencies (if any);
- the JDBC connector for the business data database with its dependencies (if any);
- the commonj library with its dependencies:
 - Apache Geronimo,
 - Concurrency JSR-166,
 - CommonJ.

Important: Enterprise Edition only

To enable the Import/Export capability, please also add the JDBC connector for [HyperSQLDB](#) , taking care of using version 1.8.0.2 .

3.2.2.2.2 File system resources

Create the folder `TOMCAT_HOME/resources`. Such a folder will contain some useful static resources and the indexes for the search engine used by Knowage.

3.2.2.2.3 Connection to metadata database

To define connection towards metadata database, edit the `TOMCAT_HOME/conf/server.xml` and add the information related to the metadata database inside the `GlobalNamingResources` tag. Specify: username, password, driver class name, JDBC URL and validation query (any valid query to be executed to validate connections). Connection's name must be `jdbc/knowage`:

```

1  <Resource auth="Container"
2      driverClassName="JDBC driver"
3      name="jdbc/knowage"
4      password="password"
5      type="javax.sql.DataSource"
6      url="JDBC URL"
7      username="username"
8      validationQuery="validation query"
9      maxTotal="50"
10     maxIdle="50"
11     minIdle="10"
12     validationInterval="34000"
13     removeAbandoned="true"
14     removeAbandonedTimeout="3600"
15     logAbandoned="true"
16     testOnBorrow="true"
17     testWhileIdle="true"
18     timeBetweenEvictionRunsMillis="10000"
19     minEvictableIdleTimeMillis="60000" />

```

3.2.2.2.4 Cache database connection

In some scenarios (for example when defining a cockpit document on top of a file dataset), Knowage requires a database to be used as cache. It is highly recommended to create an empty database schema for this purpose. Then, you need to configure it inside `TOMCAT_HOME/conf/server.xml` as you did for metadata database. Feel free to type a name of your choice, in this example we used `jdbc/ds_cache`:

```

1  <Resource auth="Container"
2      driverClassName="JDBC driver"
3      name="jdbc/ds_cache"
4      password="password"
5      type="javax.sql.DataSource"
6      url="JDBC URL"
7      username="user name"
8      validationQuery="validation query"
9      maxTotal="50"
10     maxIdle="50"
11     minIdle="10"

```

(continues on next page)

(continued from previous page)

```

12     validationInterval="34000"
13     removeAbandoned="true"
14     removeAbandonedTimeout="3600"
15     logAbandoned="true"
16     testOnBorrow="true"
17     testWhileIdle="true"
18     timeBetweenEvictionRunsMillis="10000"
19     minEvictableIdleTimeMillis="60000" />

```

3.2.2.5 Connection to business data

Edit the `TOMCAT_HOME/conf/server.xml` and add the information related to the database containing business data to be analysed by Knowage inside the `GlobalNamingResources` tag, specifying username, password, driver class name, URL and validation query. Feel free to type a name of your choice, in this example we used `jdbc/dwh`:

```

1  <Resource auth="Container"
2      driverClassName="JDBC driver"
3      name="jdbc/dwh"
4      password="password"
5      type="javax.sql.DataSource"
6      url="JDBC URL"
7      username="username"
8      validationQuery="validation query"
9      maxTotal="50"
10     maxIdle="50"
11     minIdle="10"
12     validationInterval="34000"
13     removeAbandoned="true"
14     removeAbandonedTimeout="3600"
15     logAbandoned="true"
16     testOnBorrow="true"
17     testWhileIdle="true"
18     timeBetweenEvictionRunsMillis="10000"
19     minEvictableIdleTimeMillis="60000"
20     factory="org.apache.tomcat.jdbc.pool.DataSourceFactory" />

```

3.2.2.6 Environment variables definition

Edit the file `TOMCAT_HOME/conf/server.xml` in Tomcat and add the following constants in the `GlobalNamingResources` tag.

```

<Environment name="resource_path" type="java.lang.String" value="{catalina.home}/resources" />
<Environment name="sso_class" type="java.lang.String" value="it.eng.spagobi.services.common.JWTSSoService" />
<Environment name="service_url" type="java.lang.String" value="http://localhost:8080/knowage" />
<Environment name="hmacKey" description="HMAC key" type="java.lang.String" value="PUT ANY RANDOM STRING HERE" />
<Environment name="password_encryption_secret" description="File for security encryption location" type="java.lang.String" value="complete_file_path_with_file_name" />

```

Such environment variables have the following meaning:

- `resource_path`: resources folder path,
- `sso_class`: SSO connector class name,
- `service_url`: backend services address, typically set to `http://localhost:8080/knowage`,
- `hmacKey`: secret key to generate JWT tokens used by the default security mechanism. You **must change** it, and **do not distribute** it. You can put any random alphanumeric string in it, and you can change it everytime you want, you just need to restart Tomcat to apply the change,

- **password_encryption_secret**: the complete path of a file to contain the password encryption secret. The file must contain random text of any length. This is a security configuration, so don't use short strings. For example, you can create a file and write text into it. **Do not distribute** it for any reason, create at least a backup copy of the file. **After the first start of Knowage, it will no longer be possible to change the secret key.** In case you lost this secret, look at the paragraph below to see how to update the passwords of existing users.

Important: Again we stress the point that the HMAC key must be a random string. Please DO NOT copy and paste it from this documentation, since this will compromise the security of the application.

Below you can see an example of configuration of the above variables in the server.xml file

```
<Environment name="resource_path" type="java.lang.String" value="${catalina.home}/resources"/>
<Environment name="sso_class" type="java.lang.String" value="it.eng.spagobi.services.common.JWTSSoService"/>
<Environment name="service_url" type="java.lang.String" value="http://mydomain.com/knowage"/>
<Environment name="hmacKey" description="HMAC key" type="java.lang.String" value="a random string"/>
<Environment name="password_encryption_secret" description="File for security encryption location" type="java.
↳lang.String" value="${catalina.home}/conf/knowage.secret"/>
```

3.2.2.2.7 Changing the secret key for password encryption

The password encryption secret key must be set during the installation and cannot be changed **anymore**, otherwise Knowage will no longer be able to authenticate already defined users. *In case the secret key is lost* you must create a new one, configure it into Knowage as described above and update passwords of existing users directly into Knowage metadata database (SBI_USER table). For this reason Knowage provides you a tool to get new encrypted values. This tool is a Java class that is shipped with the knowage-utils library; it accepts 2 input parameters:

- the complete path of the password encryption secret file;
- the password value in plaintext.

Below is an example of invoking the tool by command line using 'mypassword' as the plaintext password to be encrypted (of course TOMCAT_HOME must be replaced by the actual Tomcat base folder path).

```
1 java -cp "<TOMCAT_HOME>/webapps/knowage/WEB-INF/lib/knowage-utils-7.2.0.jar" it.eng.spagobi.security.utils.
↳PasswordEncryptionToolMain <TOMCAT_HOME>/conf/knowage.secret mypassword
```

This procedure must be repeated for all already existing users.

3.2.2.2.8 Mandatory configuration

[LINUX] Edit the TOMCAT_HOME/conf/setenv.sh file in Tomcat by adding the following JVM arguments:

```
1 export JAVA_OPTS="$JAVA_OPTS -Dsymmetric_encryption_key=<generic_random_string>"
```

The symmetric_encryption_key is required to encrypt/decrypt the JDBC data source password. Its value must be a generic ASCII string with at least one character.

[WIN] Edit the TOMCAT_HOME/conf/setenv.bat file in Tomcat by adding the following JVM arguments:

```
1 export JAVA_OPTS="$JAVA_OPTS -Dsymmetric_encryption_key=<generic_random_string>"
```

The symmetric_encryption_key is required to encrypt/decrypt the JDBC data source password. Its value must be a generic ASCII string with at least one character.

3.2.2.2.9 Recommended configuration

[**LINUX**] Edit the `TOMCAT_HOME/conf/setenv.sh` file in Tomcat by adding the following JVM arguments:

```

1  export JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8"
2
3  # We add -Duser.timezone=UTC to solve error when establishing connection to Oracle metadata database:
4  # java.sql.SQLException: ORA-00604: error occurred at recursive SQL level 1
5  # ORA-01882: timezone region not found
6
7  export JAVA_OPTS="$JAVA_OPTS -Duser.timezone=UTC"
8
9  export JAVA_OPTS="$JAVA_OPTS -Djava.awt.headless=true"
10
11 export JAVA_OPTS="$JAVA_OPTS -Djava.security.manager -Djava.security.policy=$CATALINA_HOME/conf/knowage-
↪ default.policy"
```

[**WIN**] Edit the `TOMCAT_HOME/conf/setenv.bat` file in Tomcat by adding the following JVM arguments:

```

1  export JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8"
2
3  # We add -Duser.timezone=UTC to solve error when establishing connection to Oracle metadata database:
4  # java.sql.SQLException: ORA-00604: error occurred at recursive SQL level 1
5  # ORA-01882: timezone region not found
6
7  export JAVA_OPTS="$JAVA_OPTS -Duser.timezone=UTC"
8
9  export JAVA_OPTS="$JAVA_OPTS -Djava.awt.headless=true"
10
11 export JAVA_OPTS="$JAVA_OPTS -Djava.security.manager -Djava.security.policy=%CATALINA_HOME%\conf\knowage-
↪ default.policy"
```

3.2.2.2.10 Applications deploy

To deploy Knowage you have to copy all the WAR files inside the `TOMCAT_HOME/webapps` folder. Once the first start is ended each WAR file will be unzipped. It is also possible to unzip the WAR files manually using the unzip utility.

3.2.2.2.11 Thread pool definition

You must configure `TOMCAT_HOME/conf/server.xml` file and add the settings related to the pool of thread editing the `GlobalNamingResources` tag, as shown follow.

```

1  <Resource auth="Container" factory="de.myfoo.commonj.work.FooWorkManagerFactory" maxThreads="5" name="wm/
↪ SpagoWorkManager" type="commonj.work.WorkManager"/>
```

3.2.2.2.12 Advanced memory settings

It is recommended to increase the memory dimension used by the application server. This can be done by adjusting some properties. The memory required by each application server depends on many factors: number of users, type of analyses, amount of handled data, etc. The minimum requirements are `Xms1024m` and `Xmx2048m`.

[**LINUX**] Insert at the beginning of the `TOMCAT_HOME/bin/setenv.sh` file this command:

```

1  export JAVA_OPTS="$JAVA_OPTS -Xms1024m -Xmx4096m -XX:MaxPermSize=512m"
```

[**WIN**] Insert at the beginning of the `TOMCAT_HOME/bin/setenv.bat` file this command:

```
set JAVA_OPTS= %JAVA_OPTS% -Xms1024m -Xmx4096m -XX:MaxPermSize=512m
```

3.2.2.2.13 Advanced Connector settings

Important: It is highly recommend to add `URIEncoding="UTF-8"` attribute to `server.xml` file connector tags in order to avoid special characters issues.

```
<Connector address="0.0.0.0" port="8009" protocol="AJP/1.3" maxPostSize="2097152000" redirectPort="8443"
  URIEncoding="UTF-8" />
```

3.2.2.3 Datasource link within the applications

You would set up `ResourceLink` for JNDI datasource. To do so, you have to configure each `knowage*/META-INF/context.xml` and set the `ResourceLink` for each JNDI data source previously created. Inside the released packages two links are already defined:

- one for the `jdbc/knowage` resource, which the user must keep
- the other for the `jdbc/foodmart`, which should be renamed with `jdbc/dwh`.

```
<Context docBase="knowage-ee" path="/knowage" reloadable="true">
  <ResourceLink global="jdbc/dwh" name="jdbc/dwh" type="javax.sql.DataSource"/>
  <ResourceLink global="jdbc/knowage" name="jdbc/knowage" type="javax.sql.DataSource"/>
  <ResourceLink global="jdbc/ds_cache" name="jdbc/ds_cache" type="javax.sql.DataSource"/>
  <ResourceLink global="resource_path" name="resource_path" type="java.lang.String" />
  <ResourceLink global="sso_class" name="sso_class" type="java.lang.String" />
  <ResourceLink name="hmacKey" global="hmacKey" type="java.lang.String"/>
  <ResourceLink global="service_url" name="service_url" type="java.lang.String"/>
  <ResourceLink global="wm/SpagoWorkManager" name="wm/SpagoWorkManager" type="commonj.work.WorkManager" />
  <ResourceLink global="password_encryption_secret" name="password_encryption_secret" type="java.lang.String" />
</Context>
```

Due to security reasons, configuration file containing some content to be used for encrypting must be created in Knowage environment. File must not be empty, but there's no minimum length. File can be put everywhere; path and name must be properly configured for each `knowage*/META-INF/context.xml` as shown below.

Important: Context update

The modification of these files will be effective as soon as the web application is reloaded or the application server is restarted.

3.2.2.4 Configuration of the metadata db dialect

Important: This step is not mandatory anymore

Knowage is now able to autonomously determine following configuration. That said, the following is not mandatory anymore however a user can force it to specific value.

Verify that the right dialect has been set inside `hibernate.cfg.xml` files. We list all the possible dialects that can be used:

```
1 <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>,
2 <property name="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</property>
3 <property name="hibernate.dialect">org.hibernate.dialect.Oracle9Dialect</property>
```

You have to configure these following Hibernate configuration files and set the chosen dialect:

```
1 knowagekpiengine/WEB-INF/classes/hibernate.cfg.xml
2 knowagegeoreportengine/WEB-INF/classes/hibernate.cfg.xml
3 knowage/WEB-INF/classes/hsqldb/hibernate.cfg.xml
4 knowage/WEB-INF/classes/hibernate.cfg.xml
5 knowagesvgviewerengine/WEB-INF/classes/hibernate.cfg.xml
6 knowagemeta/WEB-INF/classes/hibernate.cfg.xml
7 knowagecockpitengine/WEB-INF/classes/hibernate.cfg.xml
8 knowagedataminingengine/WEB-INF/classes/hibernate.cfg.xml
```

Important: Context update

The modification of these files will be effective as soon as the web application is reloaded or the application server is restarted.

3.2.2.5 Modification of the Quartz configuration

Important: This step is not mandatory anymore

Knowage is now able to autonomously determine following configuration. That said, the following is not mandatory anymore however a user can force it to specific value.

The scheduler is configured in `knowage/WEB-INF/classes/quartz.properties`. It is essential to enhance in this file the property `org.quartz.jobStore.driverDelegateClass` with the right value, according to the metadata database in use. Following the possible values:

```
1 # Hsqldb delegate class
2 #org.quartz.jobStore.driverDelegateClass=org.quartz.impl.jdbcjobstore.HSQLDBDelegate
3 # Mysql delegate class org.quartz.jobStore.driverDelegateClass=org.quartz.impl.jdbcjobstore.
  ↳ StdJDBCDelegate
4 # Postgres delegate class
5 #org.quartz.jobStore.driverDelegateClass=org.quartz.impl.jdbcjobstore.PostgreSQLDelegate
6 # Oracle delegate class
7 #org.quartz.jobStore.driverDelegateClass=org.quartz.impl.jdbcjobstore.oracle.OracleDelegate
```

3.2.2.5.1 Clustering

When Knowage is installed in cluster with several nodes, it is necessary to activate the Cluster modality, adding these parameters to the `knowage/WEB-INF/classes/quartz.properties` file of every involved machines:

```
1 org.quartz.jobStore.isClustered = true
2 org.quartz.jobStore.clusterCheckinInterval = 20000
3 org.quartz.scheduler.instanceId = AUTO
4 org.quartz.scheduler.instanceName = RHECMClusteredSchedule
```

3.2.2.6 Logging

It is necessary to set up a folder where Knowage and its analytical engines can store their respective log files. From now on, we will call LOG_DIR such folder and LOG_DIR_PATH the path that leads to it. This path is configured in file log4j.properties located inside the \WEB-INF\classes\ available in each web application. Shortly, to configure the Knowage log folder the user must execute the following steps:

- create the LOG_DIR folder on all cluster nodes on which it is intended to deploy Knowage Server and/or one of its analytical engines. The LOG_DIR_PATH string must be the same for every node;
- **[LINUX]** verify that Knowage has write permissions on this folder; set the property log4j.appender.knowage.File inside the WEB-INF/classes/log4j.properties to LOG_DIR_PATH/knowage.log;
- set the property log4j.appender.knowageXXXXXEngine.File inside the WEB-INF/classes/log4j.properties file of each engine to LOG_DIR_PATH/knowageXXXXXEngine.log;
- only for the Birt Engine, to set the property logDirectory inside the WEB-INF/classes/BirtLogConfig.properties file of the knowagebirtreportengine application to LOG_DIR_PATH.

3.2.2.7 Enable Java Security Manager

In Knowage, a user can create datasets, LOVs, etc.. with script languages like JavaScript. That introduces a security concern where a malicious user can execute code that can break the entire system. Java allows a system administrator to enable a [Security Manager](#) and to create a sandbox to limit privileges around the code that execute the script.

The Security Manager can be enabled with the following steps:

- Write a Security Policy for the Security Manager;
- Enable the Security Manager in the JVM.

The Security Policy is a text file read by a Security Manager that specifies all the privileges that a JVM can give to Java code: Tomcat has already a default policy in the file TOMCAT_HOME/conf/catalina.policy but is too much strict for Knowage code that needs to write multiple logs, make network connection and execute external applications. Knowage is already secured and can use a more relaxed policy like:

Listing 3.13: Complete path of the script

```
grant {
    permission java.security.AllPermission;
};
```

This policy can be saved to TOMCAT_HOME/conf/knowage-default.policy.

To enable the Security Manager a system administrator have to add some options to the Java JVM:

[LINUX] Insert at the end of the TOMCAT_HOME/bin/setenv.sh file this command:

```
export JAVA_OPTS="$JAVA_OPTS -Djava.security.manager -Djava.security.policy=$CATALINA_HOME/conf/knowage-
default.policy -Dsymmetric_encryption_key=<generic_random_string>"
```

The symmetric_encryption_key is required to encrypt/decrypt the JDBC data source password. Its value must be a generic ASCII string with at least one character.

[WIN] Insert at the end of the TOMCAT_HOME/bin/setenv.bat file this command:

```
set JAVA_OPTS= %JAVA_OPTS% -Djava.security.manager -Djava.security.policy=%CATALINA_HOME%\conf\knowage-default.
policy -Dsymmetric_encryption_key=<generic_ASCII_string>
```

The symmetric_encryption_key is required to encrypt/decrypt the JDBC data source password. Its value must be a generic ASCII string with at least one character.

Warning: If you are using Oracle provided Java, this configuration may lead to the error “*Illegal key size or default parameters*”. This is a problem with limited Java security policies. See <https://docs.oracle.com/javase/8/docs/technotes/guides/security/crypto/CryptoSpec.html#AppC> for more information.

3.2.2.8 Installation of Chromium Cockpit Export script

Important: Enterprise Edition only

Chromium Cockpit Export script is only available for Enterprise Edition.

Extract archive `knowage-cockpit-export-installation.zip` to `/opt/knowage`:

Listing 3.14: Complete path of the script

```
/opt/knowage/cockpit-export/cockpit-export.js
```

For alternatives path you have to fix `internal.nodejs.chromium.export.path` in Knowage Configuration Management.

3.2.2.9 Configuring environment for Data Preparation

User should have Apache Livy and Apache Spark installed.

This feature is tested on Apache Livy 0.71 and Apache Spark 2.4.8 with Scala 2_11 version.

Please refer to <https://livy.apache.org/> for more details:

To run the Livy server, you will also need an Apache Spark installation. You can get Spark releases at <https://spark.apache.org/downloads.html>. Livy requires at least Spark 1.6 and supports both Scala 2.10 and 2.11 builds of Spark. To run Livy with local sessions, first export these variables:

```
export SPARK_HOME=/usr/lib/spark
```

Then start the server with:

```
./bin/livy-server start
```

Livy uses the Spark configuration under `SPARK_HOME` by default. You can override the Spark configuration by setting the `SPARK_CONF_DIR` environment variable before starting Livy.

Please check Livy and Spark official documentation for more info.

After that it is mandatory to set this variable on Tomcat Server: **KNOWAGE_RESOURCE_PATH** This variable should point to the Tomcat server's resource folder.

Our advice, if you are on Linux environment, is to create a service for Tomcat Server and then let the variable setting available for the system, for example: `KNOWAGE_RESOURCE_PATH = /home/knowage/knowage8_1/apache-tomcat-9/resources`

After that, you should fill this property inside Knowage Configuration: **KNOWAGE.DATAPREP.LIVY_URL** with the right url of Livy server.

You will also need to configure a datasource as “Used for data preparation”, it means that the selected datasource will be used for saving prepared dataset data.

Selecting a **DATA PREPARATION DATASOURCE**:

It is really important to set a datasource for the prepared dataset ingestion output. This one will be the location of the result prepared data. You can do that checking the “Use for data preparation” checkbox using an administration role for Knowage.

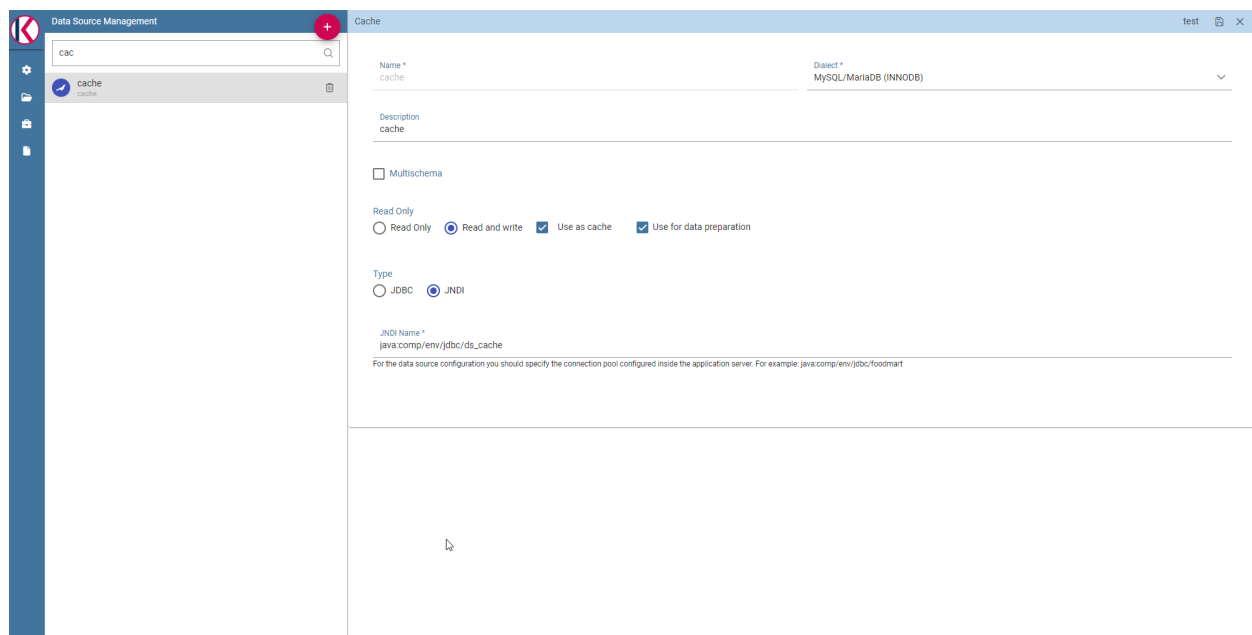


Fig. 3.2: Datasource management section.

In order to allow Spark transformations working, you should provide those libraries on Spark /jars installation folder:

- json-20210307.jar
- livy-api-0.7.1-incubating.jar
- livy-client-http-0.7.1-incubating.jar
- spark-avro_2.11-2.4.8.jar

And:

- knowage-spark-[CURRENT VERSION OF KNOWAGE].jar

This library can be found inside knowage deployed projects jars folder.

3.2.3 Python Engine

These functionalities use a Python standalone webservice, which allows to submit widgets and datasets scripts and get result from a Python environment already installed on the machine where the webservice runs. For this reason, Python environments need to be installed and configured on the same machine of Knowage server or even on a remote one, and the Python webservice has to be running **inside that same environment**. This implies that, in order to use this functionalities, you have to install Python properly (depending on the OS) on the same machine where the service will be running. You can find all information about Python installation at <https://www.python.org>. The official supported version is Python >=3.6.8, but we recommend to use 3.7.x whenever it's possible.

3.2.3.1 Install knowage-python webservice

The knowage-python package contains the source code of the webservice that has to be installed and run on the server. You can download it via pip with the command:

```
pip install knowage-python=={knowage_version_number}
```

or simply you can find it in the Knowage-Server github repository under the Knowage-Python folder.

If you downloaded knowage-python via pip, use “pip show knowage-python” to find the pip package installation location. Then copy the source file from this folder to your own custom folder such as /opt/knowagepython.

You will now have to create a file called hmakey that contains the value of the HMACkey in plaintext, and place it inside the <KNOWAGE_PYTHON_HOME>/ folder. It must be the same value specified in the server.xml file.

3.2.3.2 Run knowage-python webservice

Once you have installed all the requirements, you need to get the python-webservice running. In order to do so, you can rely on a WSGI Server. If you are working on a UNIX environment, take a look at gunicorn (<https://gunicorn.org/>). The service leverages on Flask, for deployment in any other environment take a look at the official documentation (<https://flask.palletsprojects.com/en/1.1.x/deploying/#deployment>). **The entry point for the application is <KNOWAGE_PYTHON_HOME>/knowage-python.py and the default port is 5000.**

To install GUNICORN you can use: .. code-block:: bash

```
pip3 install gunicorn greenlet eventlet gevent wheel pip3 install gunicorn eventlet gevent gthread set-
proctitle pip3 install flask flask_cors bokeh pandas pip3 install matplotlib pip3 install PyJWT pip3 install
pandas
```

Important: Webservice permissions

The knowage-python webservice must have the OS rights to read/write in its own folders.

Following you can find an example that shows you how to run knowage-python with gunicorn. First you need to create a configuration file called gunicorn.conf.py and place it under <KNOWAGE_PYTHON_HOME>/src folder.

```
import multiprocessing

bind = "0.0.0.0:5000"
workers = multiprocessing.cpu_count() * 2 + 1
timeout = 30
keepalive = 2
user = <user>
group = <group>
loglevel = 'info'
accesslog = '/var/log/gunicorn-access.log'
errorlog = '/var/log/gunicorn-error.log'
access_log_format = '%(h)s %(l)s %(u)s %(t)s "%(r)s" %(s)s %(b)s "%(f)s" "%(a)s"'
```

Then to start the service run the following command inside the <KNOWAGE_PYTHON_HOME>/ folder.

You can create service to start/stop Gunicorn.

3.2.3.3 Configure Knowage to enable Python/R functionalities

From the Knowage interface you can now enable the Python/R functionalities.

Go to the **Roles management** section, in the *Authorizations* tab under *Widgets* check the **Edit Python Scripts** option. Now you will be able to see the Python and R Dataset and Widget among the list of available ones.

Go to the **Configuration management** section, and create new variables of category `PYTHON_CONFIGURATION` and `R_CONFIGURATION`. For the label you can use `python.default.environment.url`. The value of this variables will specify the addresses of the Python and R webservices (es. `python.webservice.address.com/domain`). Now you will be able to see the addresses of the so configured environments when creating a Dataset or a Widget.

3.2.4 R Engine

As for Python, also the R functionalities leverage on a standalone webservice, this time written in R. Take a look at the official R Project documentation and find out how to get R (<https://www.r-project.org/>). The official supported version is `R >=3.5.1`, but we recommend to use `3.6.x` whenever it's possible.

3.2.4.1 Install knowage-r webservice

Inside the Knowage-Server github repository, under the Knowage-R folder you can find the sources of the knowage-r webservice.

Once you have downloaded the source code, you will have to create the configuration for the webservice. This configuration will be contained inside a file called `configs.R` and placed inside the Knowage-R folder.

The configuration is indeed really simple since you only need to specify the Knowage HMAC key contained in the `server.xml` file.

In the `constants.R` file you can set the default webservice port and a whitelist of IP addresses that can contact the webservice.

3.2.4.2 Run knowage-r webservice

Once you have installed all the requirements, you need to get the r-webservice running. In order to do so, it's enough to run the main file "knowage-r.R" with the basic R interpreter, via the `RScript` command or an equivalent one.

Important: Webservice permissions

The knowage-r webservice must have the rights to read/write in its own folder.

3.2.5 Knowage CE Installer

Knowage CE installer is an application which steers the user to the installation and the first configuration of the product.

Knowage CE is a web application, meaning it runs centrally on a server, and users interact with it through web browsers from any computer on the same network. Knowage CE Installer lets you easily configure your server.

3.2.5.1 Server-side requirements

3.2.5.1.1 Operating system

Knowage CE Installer runs on **Windows**, **Linux** and **macOS** operating systems.

3.2.5.1.2 Java platform

Knowage CE Installer requires:

- JDK 1.8
- JAVA_HOME environment variable

3.2.5.1.3 Memory

Knowage CE requires **3GB** of available RAM. This configuration is enough for most evaluation purposes.

3.2.5.1.4 Disk usage

Knowage CE requires **2GB** of free space on file system. Optional embedded MariaDB Server 10.2 requires **4GB** of free space on file system.

3.2.5.1.5 Database

Knowage CE Installer requires one of the following **external databases**:

- MySQL Server 5.7 already installed
- MariaDB Server 10.2 already installed

A user with sufficient **permissions to create schemas** must be provided. Knowage CE Installer connects to database using a JDBC driver via TCP/IP connection.

If you are using MySQL Server 5.7 we suggest to set following configuration in file `my.ini`:

- `innodb_buffer_pool_size = 2G` (adjust value here, 50%-70% of total RAM)
- `innodb_log_file_size = 500M`

Knowage CE Installer includes also the option to use one of the following **embedded databases**:

- MariaDB Server 10.2 for Windows 64 bit

Please note that embedded database option is not available for macOS.

3.2.5.1.6 Application server

Knowage CE Installer provides Apache Tomcat 7 out of the box. Don't worry about pre-installing any application server.

3.2.5.1.7 Proxy settings

If proxy is enabled please add property `http.nonProxyHosts` to JVM properties after completing installation, modifying file `<installation directory>\Knowage-Server-CE\bin\setenv.bat` on Windows or `<installation directory>/Knowage-Server-CE/bin/setenv.sh` on Linux/macOS.

```
1 -Dhttp.nonProxyHosts=localhost
```

3.2.5.2 Client-side requirements

3.2.5.2.1 Browser

Enable your browser to execute JavaScript.

3.2.5.2.2 Proxy settings

If proxy is enabled please add hostname to proxy's ignore list.

3.2.5.3 Launching

3.2.5.3.1 Windows

Important: The installer has to be run as administrator.

3.2.5.3.2 Linux/macOS

Extract the installer SH file typing the command in shell:

```
1 unzip Knowage-7_0_0-CE-Installer-Unix-20191022.zip
```

Warning: On macOS the default app used to open ZIP files may fail to extract the installer ZIP file.

Enable the execute permission on the file, typing the command in shell:

```
1 chmod +x Knowage-7_0_0-CE-Installer-Unix-20191022.sh
```

Knowage CE installer can run in GUI or console mode.

- **GUI mode** is available only if a desktop environment is available. Run installer in GUI mode typing the command in shell:

```
1 ./Knowage-7_0_0-CE-Installer-Unix-20191022.sh
```

- **Console mode** is always available and let complete installation using shell. Run installer in Console mode typing the command in shell:

```
1 ./Knowage-7_0_0-CE-Installer-Unix-20191022.sh -c
```

3.2.5.4 Managing Knowage CE

After completing installation, you can start/stop Knowage CE using desktop links, start menu entries or following shell commands.

3.2.5.4.1 Windows

- Start Knowage CE using `<installation directory>\Knowage-Server-CE\bin\startup.bat`
- Stop Knowage CE using `<installation directory>\Knowage-Server-CE\bin\shutdown.bat`

3.2.5.4.2 Windows (embedded MariaDB option)

- Start Knowage CE using `<installation directory>\Knowage-Server-CE\bin\knowage_startup.bat`
- Stop Knowage CE using `<installation directory>\Knowage-Server-CE\bin\knowage_shutdown.bat`

3.2.5.4.3 Linux/macOS

- Start Knowage CE using `<installation directory>/Knowage-Server-CE/bin/startup.sh`
- Stop Knowage CE using `<installation directory>/Knowage-Server-CE/bin/shutdown.sh`

3.3 How to upgrade KNOWAGE

This section describes the main steps to manually update an existing Knowage installation, on top of the certified Apache Tomcat server, to the latest available version. In case you are moving between 2 KNOWAGE versions with different certified Apache Tomcat servers, we recommend to follow all the instructions described on the manual installation section.

Pay attention to the fact that Knowage versions' names adhere to the Semantic Versioning 2.0.0.

In the following we will refer to Tomcat installation folder as `TOMCAT_HOME`.

The upgrade of the following Knowage components is generally needed:

- the applications that reside within `TOMCAT_HOME/webapps` folder: all `knowage*.war` files (`knowage.war`, `knowagebirtreportengine.war`, `knowagecockpitengine.war`, ...)
- the Knowage metadata database, where information about analyses, datasets, etc ... is stored.

The latter component must be upgraded in case you are moving from a different major or minor version (for example: from 6.4.x to 7.2.y, or from 7.1.x to 7.2.y), but there is no need in case you are upgrading to a new patch release of the same major/minor family (for example: from 7.2.0 to 7.2.6, or from 7.2.6 to 7.2.13).

3.3.1 Preliminary operations

Before starting upgrade procedure, you have to:

- download latest packages from the OW2 repository: base location is <https://release.ow2.org/knowage/>, you'll find a folder for each version, each folder contains: Applications with the relevant war files, and Database scripts with the SQL scripts (distributed as zip files) to upgrade Knowage metadata database for supported RDBMS;
- make a backup copy of the old web applications that reside within: `TOMCAT_HOME/webapps`;
- make a backup copy of Knowage metadata database.

3.3.2 Upgrade operations

To upgrade Knowage installation follow these steps:

- stop Apache Tomcat service;
- upgrade the Knowage metadata database by executing the SQL scripts. Each SQL script is conceived for upgrading a Knowage <major.minor> version into the next one, therefore you need to execute all scripts between your current <major.minor> version to the latest one. As an example, suppose RDBMS is Oracle, your current Knowage version is 6.0.x and you want to upgrade into Knowage 6.3.x, then you need to execute the following scripts:

```
1 ORA_upgradescript_6.0_to_6.1.sql
2 ORA_upgradescript_6.1_to_6.2.sql
3 ORA_upgradescript_6.2_to_6.3.sql
```

Note:

Moving into a newest patch version within the same <major.minor> family In case you are moving into a newest patch version that belongs to the same <major.minor> family (for example from Knowage 7.2.0 to 7.2.6, or from 7.2.6 to 7.2.13) there is no need to execute any SQL script.

- move all knowage*.war files and all the knowage* directories from TOMCAT_HOME/webapps into a backup directory;
- delete the following directories: TOMCAT_HOME/temp and TOMCAT_HOME/work;
- copy and paste all the new knowage*.war packages within TOMCAT_HOME/webapps directory;
- in case you are upgrading from a version prior to 7.2.0, create a file containing the password encryption secret (just a random text of any length). This is a security configuration, so don't use short strings. **Do not distribute** it for any reason. Then update TOMCAT_HOME/conf/server.xml file by creating a new environment variable as shown below and set the value property as the complete path of the file with password encryption secret:

```
<Environment name="password_encryption_secret" description="File for security encryption location"
type="java.lang.String" value="${catalina.home}/conf/knowage.secret"/>
```

- in case you are upgrading from a version prior to 8.1.0, you need to add the symmetric_encryption_key system property in file TOMCAT_HOME/bin/setenv.sh or TOMCAT_HOME/bin/setenv.bat; that property is required to encrypt/decrypt the JDBC data source passwords:

```
export JAVA_OPTS="$JAVA_OPTS -Dsymmetric_encryption_key=<any random string>"
```

```
set JAVA_OPTS=%JAVA_OPTS% -Dsymmetric_encryption_key=<any random string>
```

At this point, in some cases you need to deal with some configuration files, in particular when you modified the following files within the previous Knowage installation, then you need to restore those changes (after having unzipped the war files):

- context files TOMCAT_HOME/webapps/knowage*/META-INF/context.xml: they contain links to resources such as datasource connections and environment variables; in case you modified them in order to add a new data-source, you need to restore the changes and check if links to environment variables defined in TOMCAT_HOME/conf/server.xml are all there. In case you defined contexts with relevant files inside TOMCAT_HOME/conf/Catalina/localhost and you are upgrading from a version prior to 7.2.0, then you need to add the link to the password_encryption_secret variable, since that was introduced by 7.2.0 version;
- Hibernate files: they contain the metadata database Hibernate dialect (the hibernate.dialect property): since version 7.2.0, Knowage is able to autodetect the dialect by itself but, in case you customized it to a value

other than `org.hibernate.dialect.MySQLDialect` or `org.hibernate.dialect.PostgreSQLDialect` or `org.hibernate.dialect.Oracle9Dialect`, you have to restore your change: this is the list of Hibernate files to be checked:

```
TOMCAT_HOME/webapps/knowage/WEB-INF/classes/hibernate.cfg.xml
TOMCAT_HOME/webapps/knowagecockpitengine/WEB-INF/classes/hibernate.cfg.xml
TOMCAT_HOME/webapps/knowagedataminingengine/WEB-INF/classes/hibernate.cfg.xml
TOMCAT_HOME/webapps/knowagegeoreportengine/WEB-INF/classes/hibernate.cfg.xml
TOMCAT_HOME/webapps/knowagekpiengine/WEB-INF/classes/hibernate.cfg.xml
TOMCAT_HOME/webapps/knowagemeta/WEB-INF/classes/hibernate.cfg.xml
TOMCAT_HOME/webapps/knowagesvgviewerengine/WEB-INF/classes/hibernate.cfg.xml
```

- Quartz configuration file for metadata database dialect and for cluster configuration (in case of any cluster): again, since version 7.2.0, Knowage is able to autodetect the dialect by itself but, in case you customized the `org.quartz.jobStore.driverDelegateClass` property inside `TOMCAT_HOME/webapps/knowage/WEB-INF/classes/quartz.properties` to a value other than `org.quartz.impl.jdbcjobstore.StdJDBCDelegate` or `org.quartz.impl.jdbcjobstore.PostgreSQLDelegate` or `org.quartz.impl.jdbcjobstore.oracle.OracleDelegate`, you have to restore your change. Regarding cluster configuration, by default it is not enabled on released packages therefore you need to restore it in case you have a clustered installation: add these lines in `TOMCAT_HOME/webapps/knowage/WEB-INF/classes/quartz.properties` (or restore them from the backup copy):

```
org.quartz.jobStore.isClustered = true
org.quartz.jobStore.clusterCheckinInterval = 20000
org.quartz.scheduler.instanceId = AUTO
org.quartz.scheduler.instanceName = RHECMClusteredSchedule
```

Important: Since Knowage 7.2.0, the security level was highly increased. For this reason, users are requested to log in and change their password as a first step after upgrading.

To admin users: it is recommended to check which users didn't change the password and tell them to do it as soon as possible. Run the following query on the Knowage metadata database to extract the list of users who are still using the previous password encryption mechanism:

```
select * from SBI_USER where password like '#SHA#%' order by user_id;
```

3.4 How to configure the KNOWAGE Single Sign On

3.4.1 CAS SSO

CAS is an application that implements a Single-Sign-On (SSO) mechanism. It is a good practise in production environments to install it and configure it so to have secure access to the Knowage server applications. CAS expects the use of the HTTPS protocol.

3.4.1.1 Deploy of the CAS application

Carry out the following steps:

- shut down the server if running,
- deploy CAS application,
- for Tomcat: unzip the `cas.war` file inside the `TOMCAT_HOME/webapps/cas`

- for JBoss: copy the `cas.war` file inside the `JBoss_HOME/standalone/deployments`
- edit `/cas/WEB-INF/classes/cas_spagobi.properties` inserting the connection parameters for the metadata database of Knowage, as following

Listing 3.15: Connection parameters for Knowage metadata db.

```

1      spagobi.datasource.driver=<driver JDBC>
2      spagobi.datasource.url=<URL JDBC>
3      spagobi.datasource.user=<user_name>
4      spagobi.datasource.pwd=<password> encrypt.password=true

```

For further details please refer to the official documents available on CAS website <https://www.apereo.org/projects/cas>.

3.4.1.2 HTTPS certificate

Since CAS application requires the use of the HTTPS protocol, the user must own an SSL certificate: it can be released a Certification Authority (CA) or it can be self-signed (it is recommended the use of the `keytool` utility -<http://docs.oracle.com/javase/7/docs/technotes/tools/solaris/keytool.html>- available in the JDK).

In case the certificate self-signed, it must be inserted in the CA repository of the JDK (usually such a repository is located in the `JDK_HOME/jre/lib/security`) or in an ad-hoc repository, called `truststore`, and conveniently configured in the application server in use. It is sufficient to set the two Java properties `Djavax.net.ssl.trustStore=<truststore path>` and `Djavax.net.ssl.trustStorePassword=<truststore password>`

We suggest to refer to the Java documents for more details. In the following we will restrict on give some useful commands of the `keytool` utility if the user intends to install a self-signed certificate:

- generate a copy of the public/private key-pair into a repository (keystore) called `keystore.jks`, as below:

Listing 3.16: keystore.jks creation.

```

1      $JAVA_HOME/bin/keytool -genkeypair -keystore keystore.jks -storepass {keystore password} -alias
    ↪ {certificate alias} -keyalg RSA -keysize 2048 -validity 5000 -dname CN={server name that hosts Knowage}, OU=
    ↪ {organization unit}, O={organization name}, L={locality name}, ST={state name}, C={country}

```

- export a certificate in a `cert.crt` file, as suggested below:

Listing 3.17: Certificate export.

```

1      $JAVA_HOME/bin/keytool -export -alias {alias of the certificate} -file cert.crt -keystore keystore.jks

```

- set the certificate inside the CA certificates of the JDK to make it accessible (the user will be asked the CA certificates password, the default one is *changeit*)

Listing 3.18: Importing the certificate into JDK CA repository.

```

1      $JAVA_HOME/bin/keytool -import -trustcacerts -alias {alias del certificato} -file cert.crt -keystore
2      $JAVA_HOME/jre/lib/security/cacerts

```

3.4.1.3 Configuration of the HTTPS protocol for Tomcat

To enable the HTTPS protocol it is necessary to operate according to these steps:

- copy the keystore which contains the pair public/private keys (`keystore.jks`) inside the `TOMCAT_HOME/conf`;
- edit the `TOMCAT_HOME/conf/server.xml` file, comment the HTTP connector on 8080 port and uncomment the HTTPS connector on 8443 port and configure it as below:

Listing 3.19: Export of the certificate.

```

1      <Connector acceptCount="100"
2          maxHttpHeaderSize="8192"
3          clientAuth="false"
4          debug="0"
5          disableUploadTimeout="true"
6          enableLookups="false"
7          SSLEnabled="true"
8          keystoreFile="conf/keystore.jks"
9          keystorePass="<keystore password>"
10         maxSpareThreads="75"
11         maxThreads="150"
12         minSpareThreads="25"
13         port="8443"
14         scheme="https"
15         secure="true"
16         sslProtocol="TLS"
17     />

```

3.4.1.4 Knowage configuration

Once the CAS has been installed, it is necessary to modify the Knowage configuration. The user must edit some values of the SBI_CONFIG table using the administrator interface

Listing 3.20: Values of the SBI_CONFIG table to change.

```

1      SPAGOBI_SSO.ACTIVE:
2      set valueCheck to true
3
4      CAS_SSO.VALIDATE-USER.URL:
5      set valueCheck to https://<URL of the CAS application>/cas
6
7      CAS_SSO.VALIDATE-USER.SERVICE:
8      set valueCheck to https://<URL of the Knowage server >:8443/knowage/proxyCallback
9
10     SPAGOBI_SSO.SECURITY_LOGOUT_URL:
11     set valueCheck to https://<URL of the CAS application>/cas/logout

```

Then set the sso_class environment variable as below:

```

1      <Environment name="sso_class" type="java.lang.String" value="it.eng.spagobi.services.cas.
2      ↪CasSsoService3NoProxy"/>

```

This variable is located:

- Tomcat: in the TOMCAT_HOME/conf/server.xml
- JBoss: in the JBOSS_HOME/ standalone/configuration/standalone.xml

Edit all knowage\WEB-INF\web.xml to activate CAS filters.

Listing 3.21: Setting the CAS filters for sso_class variable.

```

1      <filter>
2      <filter-name>CAS Authentication Filter</filter-name>
3      <filter-class>org.jasig.cas.client.authentication.AuthenticationFilter</filter-class>
4      <init-param>
5      <param-name>casServerLoginUrl</param-name>
6      <param-value>https://<nome del server CAS>/cas/login</param-value>
7      </init-param>
8      </init-param>

```

(continues on next page)

(continued from previous page)

```

9      <param-name>serverName</param-name>
10     <param-value><dominio di knowage, incluso il protocollo e la porta, se non standard></param-value>
11   </init-param>
12 </filter>
13
14 <filter>
15   <filter-name>CAS Validation Filter</filter-name>
16   <filter-class>org.jasig.cas.client.validation.Cas20ProxyReceivingTicketValidationFilter</filter-class>
17   <init-param>
18     <param-name>casServerUrlPrefix</param-name>
19     <param-value>https://<nome del server CAS>/cas/</param-value>
20   </init-param>
21   <init-param>
22     <param-name>serverName</param-name>
23     <param-value><dominio di Knowage Server, incluso il protocollo e la porta, se non standard></
↪param-value>
24
25   </init-param>
26   <init-param>
27     <param-name>proxyReceptorUrl</param-name>
28     <param-value>/proxyCallback</param-value>
29   </init-param>
30
31   [Nelle web application knowageXXXengine presente anche questo parametro:
32
33   <init-param> <param-name>proxyCallbackUrl</param-name>
34   <param-value>
35     <dominio di knowage Server, incluso il protocollo e la porta, se non standard></ knowageXXXengine>/
↪proxyCallback </param-value>
36   </init-param>]
37
38 </filter>
39
40 <filter>
41   <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>
42   <filter-class>org.jasig.cas.client.util.HttpServletRequestWrapperFilter</filter-class>
43
44 </filter>...
45
46 <filter-mapping>
47   <filter-name>CAS Authentication Filter</filter-name>
48   <url-pattern>/servlet/*</url-pattern>
49 </filter-mapping>
50
51 <filter-mapping>
52   <filter-name>CAS Validation Filter</filter-name>
53   <url-pattern>/servlet/*</url-pattern>
54 </filter-mapping>
55 <filter-mapping>
56   <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>
57   <url-pattern>/servlet/*</url-pattern>
58 </filter-mapping>
59
60 [Nelle web application knowageXXXengine presente anche questo mapping:
61 <filter-mapping>
62   <filter-name>CAS Validation Filter</filter-name>
63   <url-pattern>/proxyCallback</url-pattern>
64 </filter-mapping>]

```

All web.xml files have CAS filters already configured, but they are commented. The user must uncomment them, looking for the strings START-CAS, END-CAS and adjust the URL as the code above reports.

3.4.2 Azure SSO

Knowage provides integration with Azure Sign-In for authentication purposes, i.e. users can login into Knowage using their Azure accounts.

Remark. Azure Sign-In is exploited only for authentication, not for authorization. The authorization part (that consists of defining roles and profile attributes for users) is still in charge of Knowage: this means that, before an user can enter into Knowage with his Azure account, the admin user has to define that user inside the regular Knowage users catalogue, keeping in mind that the user id must match the email of his Azure account (the provided password will not be considered when user will log in). In case the user is not defined inside Knowage users list, the authentication will fail.

In order to enable Azure Sign-In authentication, please follow these steps:

- start Knowage with default configuration;
- enter Knowage as admin and define users who will be allowed to enter Knowage (use their email address as user id);
- browse to **Configuration management** via the main menu;
- set the value of configuration parameter **SPAGOBI.SECURITY.USER-PROFILE-FACTORY-CLASS.className** to `it.eng.spagobi.security.azure.AzureSecurityServiceSupplier`, then save;
- stop Knowage application;
- follow Azure Sign-In documentation in order to configure your Knowage application and to get information about tenant ID and client ID;
- create a text file wherever you want and paste the tenant ID and the client ID inside it, for example: create file `TOMCAT_HOME/conf/azure.signin.config.properties` with this content:

```
1 enabled=true
2 client_id=<your client ID>
3 tenant_id=<your tenant ID>
```

- add the `azure.signin.config` Java System property that specifies the location of this properties file: in a Unix-like environment, edit `TOMCAT_HOME/bin/setenv.sh` and add

```
1 export JAVA_OPTS="%{JAVA_OPTS} -Dazure.signin.config=<your Tomcat home folder>/conf/azure.signin.config.
  ↪properties"
```

instead, in a Windows environment using Apache Tomcat, edit `TOMCAT_HOME/bin/setenv.bat`:

```
1 set JAVA_OPTS="%JAVA_OPTS% -Dazure.signin.config=<your Tomcat home folder>/conf/azure.signin.config.properties"
```

- start Knowage application.

When users will connect into Knowage, the login page will display the Azure Sing-In button:

When clicking on that button, Azure Sing-In authentication applies; if successful, users will be redirected into their home page inside Knowage. In case the account is not recognized by Knowage (i.e. the provided email address does not match any existing Knowage user id), he will get an error message.

Inside the Azure portal, after you have successfully registered our app, you will have to configure a new Single-page application, and enter “`https://<your-host-name>/knowage/servlet/AdapterHTTP`” as Redirect URI.

The authorization endpoint must be able to issue ID tokens, so make sure that the corresponding checkbox is flagged.

Directly from the application manifest.json file, you need to set “`accessTokenAcceptedVersion`”: 2 in order to use the v2.0 endpoints.



Fig. 3.3: Advanced configuration - Azure Sing-In button.

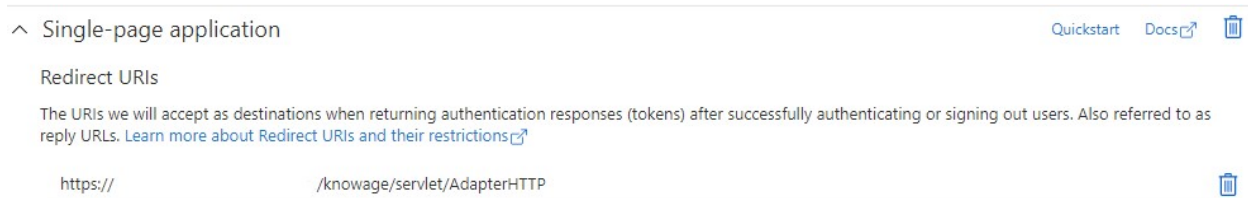


Fig. 3.4: Azure portal configuration - Redirect URI.

Implicit grant and hybrid flows

Request a token directly from the authorization endpoint. If the application has a single-page architecture (SPA) and doesn't use the authorization code flow, or if it invokes a web API via JavaScript, select both access tokens and ID tokens. For ASP.NET Core web apps and other web apps that use hybrid authentication, select only ID tokens. [Learn more about tokens.](#)

Select the tokens you would like to be issued by the authorization endpoint:

- ☒ Access tokens (used for implicit flows)
- ☒ ID tokens (used for implicit and hybrid flows) ←

Fig. 3.5: Azure portal configuration - ID token checkbox.

Home > App registrations > Knowage

Knowage | Manifest

Search (Ctrl+/) Save Discard Upload Download Got feedback?

- Overview
- Quickstart
- Integration assistant

Manage

- Branding
- Authentication
- Certificates & secrets
- Token configuration
- API permissions
- Expose an API
- App roles
- Owners
- Roles and administrators | Preview
- Manifest**

The editor below allows you to update this application by directly modifying its JSON

```
1 {
2   "id":
3   "acceptMappedClaims": null,
4   "accessTokenAcceptedVersion": 2,
5   "addIns": [],
6   "allowPublicClient": null,
7   "appId":
8   "appRoles": [],
9   "oauth2AllowUrlPathMatching": false,
10  "createdDateTime": "2021-05-04T07:46:10Z",
11  "disabledByMicrosoftStatus": null,
12  "groupMembershipClaims": null,
13  "identifierUris": [],
14  "informationalUrls": {
15    "termsOfService": null,
16    "support": null,
17    "privacy": null,
18    "marketing": null
19  },
20  "keyCredentials": [],
21  "knownClientApplications": [],
```




Fig. 3.6: Azure portal configuration - manifest.json file.

3.4.3 LDAP

3.4.3.1 Authentication

Knowage provides integration with a LDAP server for authentication purposes.

Remark. Be sure that the Knowage users have been taken under LDAP census. The LDAP security connectors check the user that is accessing Knowage, but the user must be already defined as a Knowage user. Therefore, the users must coexist in both authentication systems (LDAP and Knowage).

Knowage ships with three LDAP security connectors:

- **LdapSecurityServiceSupplier:** a pure LDAP connector that authenticates every user using an LDAP server,
- **ProfiledLdapSecurityServiceSupplier:** a mixed LDAP/internal connector that authenticates users using an LDAP server or the internal Knowage authentication system according to their profile information. More precisely, the choice of the system to be exploited is based on the **auth_mode** profile attribute: if the user profile attribute **auth_mode** is defined and its value equals to **internal** for the user, then Knowage will use its internal authentication mechanism, otherwise it will try an LDAP authentication,
- **FullLdapSecurityServiceSupplier:** this connector performs both authentication and authorization against LDAP, and is the only connector that does not require users to be present both on LDAP and Knowage. There is an apposite section dedicated to this connector at the bottom of the page.

Warning: The only way to maintain access to Knowage for **users not mapped onto LDAP** is to:

- define the user profile attribute **auth_mode**,
- set **auth_mode = internal** for every user not mapped onto LDAP,
- use the connector **ProfiledLdapSecurityServiceSupplier** (see below).

In order to setup any LDAP security connector, you need to define a `.properties` file that includes the LDAP configuration:

- **INITIAL_CONTEXT_FACTORY:** initial context factory Java class,
- **PROVIDER_URL:** LDAP server IP,
- **SECURITY_AUTHENTICATION:** authentication type,
- **DN_PREFIX:** prefix that will be prepended to the user name to create the DN (distinguished name) of logging user,
- **DN_POSTFIX:** postfix that will be appended to the user name to create the DN (distinguished name) of logging user;

Important: The final concatenation **DN_PREFIX + <Knowage user ID> + DN_POSTFIX** must be equal to the **DN (distinguished name)** of the user as defined in LDAP server. Please check DN examples at <https://ldapwiki.com/wiki/DN%20Syntax>.

Then define a custom JVM property `ldap.config`, setting its value to the path of LDAP properties file.

In a Unix-like environment using Apache Tomcat you can add a custom JVM property to variable `JAVA_OPTS` in a `setenv.sh` file under `bin` folder:

```
export JAVA_OPTS="${JAVA_OPTS} -Dldap.config=/opt/tomcat/conf/ldap.properties"
```

In a Windows environment using Apache Tomcat you can add a custom JVM property to variable `JAVA_OPTS` in a `setenv.bat` file under `bin` folder:

```
set JAVA_OPTS="%JAVA_OPTS% -Dldap.config=C:/Tomcat/conf/ldap.properties"
```

Below there is an example of the `ldap.properties` file configuration for both LDAP connectors:

```
INITIAL_CONTEXT_FACTORY = com.sun.jndi.ldap.LdapCtxFactory
PROVIDER_URL             = ldaps://<LDAP server address>:389
SECURITY_AUTHENTICATION = simple
DN_PREFIX               = CN=
DN_POSTFIX              = ,ou=IT staff,o="Example, Inc",c=US
SEARCH_USER_BEFORE      = false
SEARCH_USER_BEFORE_USER = 
SEARCH_USER_BEFORE_PSW  = 
SEARCH_USER_BEFORE_FILTER = (&((objectclass=person))(uid=%s))

ldap2.INITIAL_CONTEXT_FACTORY = com.sun.jndi.ldap.LdapCtxFactory
ldap2.PROVIDER_URL            = ldaps://<LDAP 2 server address>:389
ldap2.SECURITY_AUTHENTICATION = simple
ldap2.DN_PREFIX              = CN=
ldap2.DN_POSTFIX             = ,ou=IT staff,o="Example, Inc",c=US
ldap2.SEARCH_USER_BEFORE     = true
ldap2.SEARCH_USER_BEFORE_USER = user_before_username
ldap2.SEARCH_USER_BEFORE_PSW = user_before_password
ldap2.SEARCH_USER_BEFORE_FILTER = (&((objectclass=account))(uid=%s))
```

Using the example above, you can configure:

- *auth_mode = internal*: if you want to log in using the Knowage metadata database instead of contacting an LDAP server
- *empty auth_mode value*: if you want to login using the first configuration of the properties file (the one without any prefix)
- *auth_mode = ldap2*: if you want to log in by contacting the LDAP server using the configuration with “`ldap2.`” prefix

In case you want the users to authenticate using their complete distinguish name, set `SEARCH_USER_BEFORE` key to be *false*. In case you want instead the users to authenticate using an LDAP property such as `uid`, then set `SEARCH_USER_BEFORE` key to be *true*; you need also to specify the `SEARCH_USER_BEFORE_FILTER` filter that Knowage will exploit in order to retrieve the user’s information on the LDAP server. **Pay attention that %s place-holder must be present**: it will be replaced by Knowage with the actual username provided by the user when logging in.

The `SEARCH_USER_BEFORE_USER` and `SEARCH_USER_BEFORE_PSW` keys are credentials to authenticate to LDAP server; if the first one is set, the second one will be considered also. *These parameters are used only if anonymous bind is not allowed for LDAP server. For this reason they are optional and can be empty.*

Important: Restart your application server in order to load the custom JVM property.

Warning: After enabling the search functionality you may encounter the following exception: `javax.naming.PartialResultException: Unprocessed Continuation Reference(s)`. To solve this just change LDAP port from 389 to 3268 (credits: <https://stackoverflow.com/questions/16412236/how-to-resolve-javax-naming-partialresultexception>)

The final step is to set the LDAP security connector as follow:

- access Knowage as administrator,

- browse to **Configuration Management** via the main menu,
- set the value of config **SPAGOBISECURITY.USER-PROFILE-FACTORY-CLASS.className** to `it.eng.spagobi.security.LdapSecurityServiceSupplier` or `it.eng.spagobi.security.ProfiledLdapSecurityServiceSupplier`,
- save,
- log out of Knowage.

Warning: To recover the default authentication mechanism please revert manually the config **SPAGOBISECURITY.USER-PROFILE-FACTORY-CLASS.className** to its default value `it.eng.spagobi.security.InternalSecurityServiceSupplierImpl` using a database client.

Knowage is now ready to authenticate the users via LDAP credentials.

3.4.3.2 Authentication + authorization

A third LDAP connector is available, which does not need users to be present in Knowage metadata (but only on LDAP system). This connector allows you to build the UserProfile object based on the information contained in the LDAP system. Only if the user is not found in the LDAP system, the Knowage metadata will be used (this scenario is useful for technical users that are not registered inside the LDAP).

Important: Whenever a user logs into Knowage, the connector will always first query the LDAP system in order to authenticate the user and build the profile object. Only if this operation fails (it means that the user is not present in the LDAP system) Knowage will use its internal metadata to authenticate the user.

As for the above connectors, you must follow some steps in order to setup this connector. First you need to define a `.properties` file that includes the following configurations:

- **INITIAL_CONTEXT_FACTORY**: initial context factory Java class,
- **PROVIDER_URL**: LDAP server IP,
- **SECURITY_AUTHENTICATION**: authentication type,
- **DN_PREFIX**: prefix that will be prepended to the user name to create the DN (distinguished name) of logging user,
- **DN_POSTFIX**: postfix that will be appended to the user name to create the DN (distinguished name) of logging user,
- **AUTHENTICATION_FILTER**: LDAP filter that will be checked before allowing permission to users,
- **USER_ROLES_ATTRIBUTE_NAME**: LDAP attribute that contains the full string with user roles,
- **USER_ROLES_ATTRIBUTE_FIELD**: prefix that anticipates each Knowage role inside the full string,
- **SUPERADMIN_ATTRIBUTE**: role that grants superadmin permissions;

Suppose that the structure of a user, inside LDAP, is as follows:

```
cn      : John
sn      : Doe
mail    : john.doe@yourorganization.com
enabled : true
deprecated : false
memberOf : cn=ADMIN,ou=unit1,dc=yourorganization,dc=com
```

(continues on next page)

(continued from previous page)

```
isMemberOf : cn=DEVELOPER,ou=unit1,dc=yourorganization,dc=com
isMemberOf : cn=USER,ou=unit1,dc=yourorganization,dc=com
```

We want the authentication to succeed only if `enabled=true` and `deprecated=false`. Furthermore, we want to build profile object by using all the `isMemberOf` attributes, and using the Knowage roles specified under the `cn` properties.

Below there is an example of the `ldap.properties` file configuration that satisfies the above requirements:

```
INITIAL_CONTEXT_FACTORY = com.sun.jndi.ldap.LdapCtxFactory
PROVIDER_URL             = ldaps://<LDAP server address>:389
SECURITY_AUTHENTICATION = simple
DN_PREFIX               = CN=
DN_POSTFIX              = ,ou=IT staff,o="Example, Inc",c=US
AUTHENTICATION_FILTER   = (&(enabled=true)(!(deprecated=true)))
USER_ROLES_ATTRIBUTE_NAME = isMemberOf
USER_ROLES_ATTRIBUTE_FIELD = cn
SUPERADMIN_ATTRIBUTE    = ADMIN
```

Then you need to define a custom JVM property `ldap.config`, setting its value to the path of LDAP properties file.

In a Unix-like environment using Apache Tomcat you can add a custom JVM property to variable `JAVA_OPTS` in a `setenv.sh` file under `bin` folder:

```
export JAVA_OPTS="${JAVA_OPTS} -Dldap.config=/opt/tomcat/conf/ldap.properties"
```

In a Windows environment using Apache Tomcat you can add a custom JVM property to variable `JAVA_OPTS` in a `setenv.bat` file under `bin` folder:

```
set JAVA_OPTS="%JAVA_OPTS% -Dldap.config=C:/Tomcat/conf/ldap.properties"
```

The final step is to set the LDAP security connector as follow:

- access Knowage as administrator,
- browse to **Configuration Management** via the main menu,
- set the value of config **SPAGOBI.SECURITY.USER-PROFILE-FACTORY-CLASS.className** to `it.eng.spagobi.security.FullLdapSecurityServiceSupplier`,
- save,
- log out of Knowage.

Warning: To recover the default authentication mechanism please revert manually the config **SPAGOBI.SECURITY.USER-PROFILE-FACTORY-CLASS.className** to its default value `it.eng.spagobi.security.InternalSecurityServiceSupplierImpl` using a database client.

3.4.4 Google SSO

Knowage provides integration with Google Sign-In for authentication purposes, i.e. users can login into Knowage using their Google accounts.

Remark. Google Sign-In is exploited only for authentication, not for authorization. The authorization part (that consists of defining roles and profile attributes for users) is still in charge of Knowage: this means that, before an user can enter into Knowage with his Google account, the admin user has to define that user inside the regular Knowage users catalogue, keeping in mind that the user id must match the email of his Google account (the provided password will

not be considered when user will log in). In case the user is not defined inside Knowage users list, the authentication will fail.

In order to enable Google Sign-In authentication, please follow these steps:

- start Knowage with default configuration;
- enter Knowage as admin and define users who will be allowed to enter Knowage (use their email address as user id);
- browse to **Configuration management** via the main menu;
- set the value of configuration parameter **SPAGOBI.SECURITY.USER-PROFILE-FACTORY-CLASS.className** to `it.eng.spagobi.security.GoogleSecurityServiceSupplier`, then save;
- stop Knowage application;
- follow Google Sign-In documentation in order to configure your Knowage application and to get the OAuth client ID;
- create a text file wherever you want and paste the client ID inside it, for example: create file `TOMCAT_HOME/conf/google.signin.config.properties` with this content:

```
client_id=<your client ID>.apps.googleusercontent.com
```

- add the `google.signin.config` Java System property that specifies the location of this properties file: in a Unix-like environment, edit `TOMCAT_HOME/bin/setenv.sh` and add

```
export JAVA_OPTS="${JAVA_OPTS} -Dgoogle.signin.config=<your Tomcat home folder>/conf/google.signin.config.  
↪properties"
```

instead, in a Windows environment using Apache Tomcat, edit `TOMCAT_HOME/bin/setenv.bat`:

```
set JAVA_OPTS="%JAVA_OPTS% -Dgoogle.signin.config=<your Tomcat home folder>/conf/google.signin.config.  
↪properties"
```

- start Knowage application.

When users will connect into Knowage, the login page will display the Google Sing-In button:

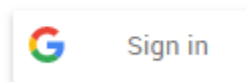


Fig. 3.7: Advanced configuration - Google Sing-In button.

When clicking on that button, Google Sing-In authentication applies; if successful, users will be redirected into their home page inside Knowage. In case the account is not recognized by Knowage (i.e. the provided email address does not match any existing Knowage user id), he will get an error message:



AUTHENTICATION FAILED. PLEASE CHECK IF YOU ARE TO ALLOWED TO ENTER THIS APPLICATION.

Fig. 3.8: Advanced configuration - Authentication error.

3.5 Server settings

In this chapter we will describe all the advanced configuration parameters of Knowage.

3.5.1 Thread manager

For Tomcat: the configuration of the pool of thread is available inside the TOMCAT_HOME/conf/server.xml. Refer to Code below.

Listing 3.22: Configuration of the pool of thread for Tomcat.

```

1 <Resource auth="Container" factory="de.myfoo.commonj.work.FooWorkManagerFactory"
2   maxThreads="5"
3   minThreads="1"
4   queueLength="10"
5   maxDaemons="10"
6   name="wm/SpagoWorkManager"
7   type="commonj.work.WorkManager"/>

```

For JBoss: the configuration of the pool of thread is available inside the JBOSS_HOME/standalone/configuration/s. Refer to Code below.

Listing 3.23: Configuration of the pool of thread for JBoss.

```

1 <object-factory name="java:global/SpagoWorkManager" module="de.myfoo.commonj"
2   class="de.myfoo.commonj.work.MyFooWorkManagerFactory">
3   <environment>
4     <property name="maxThreads" value="5"/>
5     <property name="minThreads" value="1"/>
6     <property name="queueLength" value="10"/>
7     <property name="maxDaemons" value="10"/>
8   </environment>
9 </object-factory>

```

In both cases, the meaning of the configuration parameters is the following:

- minThreads: the minimum number of threads in the thread pool. Default: 2;
- maxThreads: the maximum number of threads in the thread pool. Default: 10;

- **queueLength**: the number of work items that can be queued - 0 means no queuing. Default: 10;
- **maxDaemons**: the maximum number of daemon threads to allow for this work manager. Default: 10.

3.5.2 Cache parameters

First of all, the user must configure the distributed cache. This helps to coordinate the parallel access to the distributed cache, guaranteeing a thread-safe access. It is necessary to configure the `hazelcast.xml` file (available in the `knowage/WEB-INF/classes/`) typing in the "member" tag the IP address or hostname of the machine on which Knowage is installed (for example `<member> 192.168.29.43</member>`). In case of multi-node configuration, it is obviously important to report all cluster members. This operation must be carried out on every node. Furthermore, it is possible to implement a finer tuning of the cache behaviour, changing the Knowage configuration. The user must edit some values of the `SBI_CONFIG` table using the specific administrator interface.

- **SPAGOBI.CACHE.NAMEPREFIX**: It configures the prefix of temporary table in the cache (Default : "sbi-cache")
- **SPAGOBI.CACHE.SPACE_AVAILABLE**: It resizes cache dimension (bytes) (Default : 1024)
- **SPAGOBI.CACHE.LIMIT_FOR_CLEAN**: It configures the maximum cache section (in percentage) that can be cleaned at runtime when the cache has not enough space to store a dataset. (Default : 50)
- **SPAGOBI.CACHE.DS_LAST_ACCESS_TTL**: It configures the Time To Live of a dataset inside the cache. This parameter defines the minimum TTL (in seconds) so to guarantee that a dataset remains in cache. A too-high value can lead the cache to breakdown (in this case, there is no way to insert new datasets), while a too low value can lead to situations when there are no certainties of the stability of the dataset in the cache. (Default 600)
- **SPAGOBI.CACHE.DATABASE_SCHEMA**: Name of the schema on which the tables are created. Such schema is defined by the datasource when it is set as Write-Default. Generally it is not necessary to configure this parameter since it is calculated at runtime. (default <empty>)
- **SPAGOBI.CACHE.LIMIT_FOR_STORE**: It configures the ratio (in percentage) between the dimension of the cache and the maximum dimension of a dataset in the cache. If the dimension of the dataset which the user intends to persist is bigger than the configured percentage, the system blocks the that persistence attempt. (Default : 10)
- **SPAGOBI.CACHE.CREATE_AND_PERSIST_TABLE.TIMEOUT**: It represents the maximum time (in seconds) to create temporary table for the dataset. (Default : 120)
- **SPAGOBI.WORKMANAGER.SQLDBCACHE.TIMEOUT**: It represents the maximum waiting time (in milliseconds) of an asynchronous work. (Default: 180000)
- **SPAGOBI.CACHE.HAZELCAST.TIMEOUT** : It represents the maximum time (in seconds) to get a distributed lock. (Default 120)
- **SPAGOBI.CACHE.HAZELCAST.LEASETIME**: It represents the maximum time (in seconds) for releasing a distributed lock already got. (Default :240)
- **SPAGOBI.CACHE.SCHEDULING_FULL_CLEAN**: It schedules the recurring operation of complete cleaning of the cache. This periodic cleaning delete all dataset in the cache, without considering further parameters. At the end of the cleaning, the cache is empty. The allowable values are:

Option	Description
EVERY_1_MIN("EVERY_1_MIN", "0 0/1 * 1/1 * ? *")	every minute starting the changing of hour
EVERY_10_MINS("EVERY_10_MINS", "0 0/10 * 1/1 * ? *")	every 10 minutes starting the changing of hour
EVERY_15_MINS("EVERY_15_MINS", "0 0/15 * 1/1 * ? *")	every 15 minutes starting the changing of hour
EVERY_20_MINS("EVERY_20_MINS", "0 0/20 * 1/1 * ? *")	every 20 minutes starting the changing of hour
EVERY_30_MINS("EVERY_30_MINS", "0 0/30 * 1/1 * ? *")	every 30 minutes starting the changing of hour
HOURLY("HOURLY", "0 0 0/1 1/1 * ? *")	every hour
DAILY("DAILY", "0 0 0 1/1 * ? *")	every day at midnight
WEEKLY("WEEKLY", "0 0 0 ? * SUN *")	every week at midnight on Sunday
MONTHLY("MONTHLY", "0 0 0 1 1/1 ? *")	at midnight on the first day of the month
YEARLY("YEARLY", "0 0 0 1 1 ? *")	at midnight on the first day of the year

Any value other than those listed above does not enable periodic cleaning. (Default: DAILY)

3.5.3 Logging

Knowage uses the component Log4J to create the log applications. Each web application has its own file inside the folder /knowageXXXX/WEB-INF/classes/log4j.properties. The content of this file change accordingly to the settings: the **appenders** allows to modify the level of the log. As an example, in the following code block, we analyze the log file of Knowage. In the first part we can set the generation mechanism of the log file, while in the second one the level of tracing.

Listing 3.24: Log appender.

```

1 log4j.rootLogger=ERROR, SpagoBI
2
3 # SpagoBI Appender
4 log4j.appender.SpagoBI=org.apache.log4j.RollingFileAppender
5 log4j.appender.SpagoBI.File=${catalina.base}/logs/knowage.log
6 log4j.appender.SpagoBI.MaxFileSize=10000KB
7 log4j.appender.SpagoBI.MaxBackupIndex=0
8 log4j.appender.SpagoBI.layout=org.apache.log4j.PatternLayout
9 log4j.appender.SpagoBI.layout.ConversionPattern=[%t] %d{DATE} %5p %c.%M:%L - %m %n
10
11 log4j.appender.SpagoBI.append=false
12
13 log4j.appender.Quartz=org.apache.log4j.RollingFileAppender
14 log4j.appender.Quartz.File=${catalina.base}/logs/Quartz.log
15 log4j.appender.Quartz.MaxFileSize=10000KB
16 log4j.appender.Quartz.MaxBackupIndex=10
17 log4j.appender.Quartz.layout=org.apache.log4j.PatternLayout
18 log4j.appender.Quartz.layout.ConversionPattern=[%t] %d{DATE} %5p %c.%M:%L - %m %n
19
20 log4j.appender.SpagoBI_Audit=org.apache.log4j.FileAppender
21 log4j.appender.SpagoBI_Audit.File=${catalina.base}/logs/knowage_[1]\_OperatorTrace.log
22
23 log4j.appender.SpagoBI_Audit.layout=org.apache.log4j.PatternLayout
24 log4j.appender.SpagoBI_Audit.layout.ConversionPattern=%m%n
25
26 log4j.appender.CONSOLE = org.apache.log4j.ConsoleAppender
27 log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout
28 log4j.appender.CONSOLE.layout.ConversionPattern=%c.%M: %m%n #
29
30
31 log4j.logger.Spago=ERROR, SpagoBI log4j.additivity.Spago=false
32
33 log4j.logger.it.eng.spagobi=ERROR, SpagoBI, CONSOLE
34 log4j.additivity.it.eng.spagobi=false
35

```

(continues on next page)

(continued from previous page)

```

36 log4j.logger.it.eng.spagobi.commons.utilities.messages=ERROR, SpagoBI
37 log4j.logger.it.eng.spagobi.commons.utilities.urls.WebUrlBuilder=ERROR, SpagoBI
38 log4j.logger.org.quartz=ERROR, Quartz, CONSOLE
39 log4j.logger.org.hibernate=ERROR, SpagoBI
40
41 log4j.logger.audit=INFO, SpagoBI_Audit log4j.additivity.audit=false

```

If the user wishes to enable the tracing of the information to **DEBUG** level it is enough to modify the following line

```
1 log4j.logger.it.eng.spagobi=ERROR, SpagoBI, CONSOLE
```

in

```
1 log4j.logger.it.eng.spagobi=DEBUG, SpagoBI, CONSOLE
```

For further details we refer to the official Log4J documents.

3.5.4 Mail server

Knowage uses in some situations the mail server to send emails. The configuration of this feature can be done right straight through the Knowage GUI, after accessing with administrator credentials.

Selecting the category MAIL the user gets the list of parameters to configure for the following profiles:

- trustedStore;
- scheduler, used by the scheduler to send a report by mail;
- user, used directly by the user when he intends to send a report by mail;
- kpi_alarm, used by the alarm component to send mails.

Etichetta	Nome	Descrizione	Attivo	Valore	Tipo	Categoria
internal.security.encrypt.password	encrypt password	Enable the password encryption	No		NUM	SECURITY
changePwdmodule.len_min	Password Len Min	Minimum length	No		NUM	SECURITY
changePwdmodule.special_char	Special char	Special chars	No		STRING	SECURITY
changePwdmodule.upper_char	Upper char	Minimum a char must be in upper ...	No		STRING	SECURITY
changePwdmodule.lower_char	Lower char	Minimum a char must be in lower ...	No		STRING	SECURITY
changePwdmodule.number	Number	Minimum a char must be a number	No		STRING	SECURITY
changePwdmodule.alphabetical	Alphabetical	Minimum a char must be a letter	No		STRING	SECURITY
changePwdmodule.change	Change from last	The new pwd must be different fro...	No		STRING	SECURITY
changePwd.change_first	Change at first login	The pwd must be changed at first ...	No		STRING	SECURITY
changePwd.disactivation_time	Disactivation time	Number of months before disactiv...	No		NUM	SECURITY
changePwd.expired_time	Expired time	Number of days to the expiration	No	90	NUM	SECURITY
SPAGOB.I.SPA.GOB.I.MODE.mode	SpagoBI mode	Enable the WebApplication or Por...	No	WEB	STRING	GENERIC_CONFIGURATION
SPAGOB.I.HOME.BANNER.view	show the banner	banner	No	true	STRING	GENERIC_CONFIGURATION
SPAGOB.I.HOME.FOOTER.view	show the footer	footer	No	true	STRING	GENERIC_CONFIGURATION
SPAGOB.I.MENU.pathTracked	pathTracked	pathTracked	No	true	STRING	GENERIC_CONFIGURATION
SPAGOB.I.LOOKUP.numberRows	SPAGOB.I LOOKUP numberRows	The number of rows showed in e...	Si	20	STRING	GENERIC_CONFIGURATION
SPAGOB.I.RESOURCE_PATH_JN...	RESOURCE PATH JNDI NAME	The name of the JNDI variable tha...	Si	java:/comp/env/resource_path	STRING	GENERIC_CONFIGURATION
SPAGOB.I.SPA.GOB.I.HOST_JNDI	SPAGOB.I HOST JNDI	HOST JNDI	Si	java:/comp/env/host_url	STRING	GENERIC_CONFIGURATION
SPAGOB.I.SPA.GOB.I.ADAPTERHT...	ADAPTERHTTP URL	ADAPTERHTTP URL	Si	/servlet/AdapterHTTP	STRING	GENERIC_CONFIGURATION
SPAGOB.I.TEMPLATE_MAX_SIZE	TEMPLATE MAX SIZE	TEMPLATE MAX SIZE	Si	5242880	STRING	GENERIC_CONFIGURATION
SPAGOB.I.SPA.GOB.I.CONTEXT	SPAGOB.I CONTEXT	SPAGOB.I CONTEXT	Si	/knowage	STRING	GENERIC_CONFIGURATION
SPAGOB.I.SESSION_EXPIRED_U...	SESSION EXPIRED URL	SESSION EXPIRED URL	Si	/WEB-INF/jsp/commons/sessionE...	STRING	GENERIC_CONFIGURATION
SPAGOB.I.DATASET_maxResult	DATASET maxResult	maxResult	Si	500000	STRING	GENERIC_CONFIGURATION
SPAGOB.I.SESSION_PARAMETER...	SESSION PARAMETERS MANA...	enabled	Si	false	STRING	GENERIC_CONFIGURATION
SPAGOB.I.DATASET_FILE_MAX...	DATASET FILE MAX SIZE	Max size for a file used as a dataset	Si	10485760	STRING	GENERIC_CONFIGURATION
SPAGOB.I.APL.DATASET_MAX_RO...	DATASET MAX ROWS NUMBER ...	Max number of rows returned from...	Si	5000	STRING	GENERIC_CONFIGURATION
SPAGOB.I.EXECUTION.PARAME...	Parameter state persistence enab...	if true the default value for each p...	Si	false	STRING	GENERIC_CONFIGURATION
SPAGOB.I.EXECUTION.PARAME...	Parameter state persistence scope	if equals SESSION the parameter ...	Si	SESSION	STRING	GENERIC_CONFIGURATION
SPAGOB.I.EXECUTION.PARAME...	Parameter memento persistence e...	if true the last N values selected fr...	Si	true	STRING	GENERIC_CONFIGURATION

Fig. 3.9: Mail server configuration.

Moreover, each profile has the following values:

- smtphost: the smtp server,
- Smtpport: the port in use,
- from: the address to which the mail will be associated,
- user: the user of the server connection,
- password: user's password,
- security: the user must choose between NONE, SSL and STARTTLS.

3.5.5 Maximum file size

For security reasons, Knowage has a series of parameters which manage the maximum file size that can be loaded on the server through the web GUI. To modify those parameters, it is required to enter the Knowage server application as administrator and access the "server settings" section and then "configuration management". The parameters at issue are the following:

- **SPAGOBI.TEMPLATE_MAX_SIZE** : TEMPLATE MAX SIZE: it is the maximum template dimension of an analytical document, expressed in bytes; the default value is 5MB;
- **SPAGOBI.DATASET_FILE_MAX_SIZE** : DATASET FILE MAX SIZE: it is the maximum dimension of a file used as a dataset, expressed in bytes; the default value is 10MB;
- **SPAGOBI.DOCUMENTS.MAX_PREVIEW_IMAGE_SIZE** : Max preview image size: it is the maximum dimension of an image used as document preview (in the document browser, for instance), expressed in bytes; the default is 1MB;
- **IMAGE_GALLERY.MAX_IMAGE_SIZE_KB** : Max image size in Kb: it is the maximum size of the images that can be used in a cockpit creation; the default is 1MB;

3.5.6 Date format

Knowage allows the user to visualize the date time in a format that depends on the selected language. To change the visualization of such formats, the user must enter Knowage as administrator and access the "Server Settings" section and, consequently, the "Configuration management". Then finally select "DATE-FORMAT".

Config						
Add Delete		Select Category DATE-FORMAT		Label Enter search Label	Name Enter search Name	
Label	Name	Description	Active	Value check	Type	Category
SPAGOBI.DATE-FORMAT-SERVER.format	SERVE DATE FORMAT	Date format for communications with the server (both on sending a...	Yes	ddMM/yyyy	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT-SERVER.extJsFormat	EXTJS SERVER DAT...	extJsFormat	Yes	dmm/Y	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT.format	DATE FORMAT	Date visual format if language is not found	Yes	ddMM/yyyy	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT.extJsFormat	EXTJS DATE FORMAT	Date visual format if language is not found	Yes	dmm/Y	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT-IT_IT.format	IT DATE FORMAT	Date format used while displaying dates according to user current L...	Yes	ddMM/yyyy	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT-IT_IT.extJsFormat	EXTJS IT DATE FOR...	extJsFormat	Yes	dmm/Y	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT-EN_US.format	US DATE FORMAT	Date format used while displaying dates according to user current L...	Yes	MM/dd/yyyy	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT-EN_US.extJsFormat	EXTJS US DATE FOR...	extJsFormat	Yes	mm/Y	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT-FR_FR.format	FR DATE FORMAT	Date format used while displaying dates according to user current L...	Yes	ddMM/yyyy	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT-FR_FR.extJsFormat	EXTJS RF DATE FOR...	extJsFormat	Yes	dmm/Y	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT-ES_ES.format	ES DATE FORMAT	Date format used while displaying dates according to user current L...	Yes	ddMM/yyyy	STRING	DATE-FORMAT
SPAGOBI.DATE-FORMAT-ES_ES.extJsFormat	EXTJS ES DATE FOR...	extJsFormat	Yes	dmm/Y	STRING	DATE-FORMAT
SPAGOBI.TIMESTAMP-FORMAT.format	TIMESTAMP FORMAT	TimeStamp Format of SpagoBI DB	Yes	ddMM/yyyy HH:mm:ss	STRING	DATE-FORMAT
SPAGOBI.TIMESTAMP-FORMAT.extJsFormat	EXTJS TIMESTAMP F...	TimeStamp Format of SpagoBI DB	Yes	dmm/Y H:is	STRING	DATE-FORMAT
SPAGOBI.DB-TIMESTAMP-FORMAT.format	DB TIMESTAMP FOR...	DB TIMESTAMP FORMAT	Yes	yyyy-MM-dd HH:mm:ss	STRING	DATE-FORMAT
SPAGOBI.DB_LOG.value	DB LOG	DB LOG	Yes	false	STRING	DATE-FORMAT

Fig. 3.10: Date format configuration.

For each available language there are two parameters:

- **SPAGOBI.DATE-FORMAT-<lingua>_<nazione>.format**: it rules the back-end role;
- **SPAGOBI.DATE-FORMAT-<lingua>_<nazione>.extJsFormat**: it rules the front-end role.

We suggest to set the parameters in compliance with each other, according to a local data.

The parameters **SPAGOBI.DATE-FORMAT-SERVER.format** and **SPAGOBI.DATE-FORMAT-SERVER.extJsFormat** control the link between back-end and front-end. The adjustment of these parameters do not affect the web GUI.

3.5.7 Language

Knowage manages the multi-language. The list of all languages is manageable from the “Server Settings” section. Go to “Configuration management” and select the LANGUAGE_SUPPORTED category. Here there are two properties:

- **SPAGOBI.LANGUAGE_SUPPORTED.LANGUAGES**: the list of all supported languages underneath this formalism are: [it,IT],[en,US],[fr,FR],[es,ES];
- **SPAGOBI.LANGUAGE_SUPPORTED.LANGUAGE.default**: the default value is [en,US].

3.5.8 Adding new languages

To add more languages to the list (if not provided) a few steps are required:

- Inside the “Configuration Management” add the language as required in previous chapter. If you don’t know the languages and country code you can get them from this [language code table](#)
- Inside the “Domain Management” add the language as language ISO code as in the following image.




valueCd	valueName	domainCode	domainName "Y"	valueDescription	
ITA	Italian	LANG	Language ISO code	Italian	 
SPA	Spanish	LANG	Language ISO code	Spanish	 
ENG	English	LANG	Language ISO code	English	 
FRA	French	LANG	Language ISO code	French	 
POR	Portoguese	LANG	Language ISO code	Portoguese	 
SWE	swedish	LANG	Language ISO code	swedish	 

Fig. 3.11: domain management example.

- Add inside the project folder Knowage-Server/knowage/src/main/webapp/js/lib/angular-localization/ the desired locale file from the [angular locales](#) and rename it with the lang code from [language code table](#)
- Add inside the project folder Knowage-Server/knowage/src/main/webapp/js/src/messages/ the translations file. The name should be messages_country_LANGUAGE.properties. Ie: messages_country_sw_se.properties.

Warning: use the last step only if no zanata translations are available. Using Zanata should be the primary translations source.

3.5.9 Password constraints settings

User password constraints can be set configuring parameters below:

- **changepwdmodule.len_min**: minimum number of character for the password;
- **changepwdmodule.special_char**: set of allowed special characters;
- **changepwdmodule.upper_char**: if active, the password must contain at least one of the uppercase characters set in the value;
- **changepwdmodule.lower_char**: if active, the password must contain at least one of the lowercase characters set in the value;
- **changepwdmodule.number**: if active, the password must contain at least one of the digit set in the value;

- **changepwdmodule.alphabetical**: if active, the password must contain at least one alphabetical set in the value;
- **changepwdmodule.change**: if true, new password must be different from the latest;
- **changepwd.change_first**: if true, password must be changed at first login;
- **changepwd.disactivation_time**: number of months before deactivation;
- **changepwd.expired_time**: number of days for the expiration.





















Label	Name	Value Check	Category	Active	
changepwdmodule.len_min	Password Len Min	8	SECURITY	false	 
changepwdmodule.special_ch...	Special char	_!@#\$	SECURITY	false	 
changepwdmodule.upper_char	Upper char	ABCDEFGHIJKLMNOPQRSTUVWXYZ...	SECURITY	false	 
changepwdmodule.lower_char	Lower char	abcdefghijklmnopqrstuvwxyz	SECURITY	false	 
changepwdmodule.number	Number	0123456789	SECURITY	false	 
changepwdmodule.alphabetical	Alphabetical	abcdefghijklmnopqrstuvwxyzAB...	SECURITY	false	 
changepwdmodule.change	Change from last		SECURITY	false	 
changepwd.change_first	Change at first login	true	SECURITY	true	 
changepwd.disactivation_time	Disactivation time	6	SECURITY	true	 
changepwd.expired_time	Expired time	90	SECURITY	true	 

Fig. 3.12: Advanced configuration - password constraints settings.

By default, all above configurations are disabled.

3.5.10 Login security settings

Login security configurations can be set filling fields below:

- **internal.security.login.checkForMaxFailedLoginAttempts**: if active and set to true, users will only be able to access Knowage if they have not reached the maximum number of failed login attempts;
- **internal.security.login.maxFailedLoginAttempts**: the maximum number of failed login attempts.





Label	Name	Value Check	Category	Active	
internal.security.login.checkFo...	Max failed login attempts filter	true	SECURITY	true	 
internal.security.login.maxFail...	Max failed login attempts	10	SECURITY	true	 

Fig. 3.13: Advanced configuration - login security settings.

3.5.11 Resource export folder cleaning settings

Resource export folder cleaning configurations can be set filling fields below:

- **KNOWAGE.RESOURCE.EXPORT.FOLDER.CLEANING_PERCENTAGE**: if active, the cleaning procedure will delete the files contained in the export resource folder leaving this percentage of free space (0 - 100). Default 30;
- **KNOWAGE.RESOURCE.EXPORT.FOLDER.MAX_FOLDER_SIZE**: if active, cleaning procedure will start only if the resource export folder will reach this size (byte). Default 10737418240.

3.5.12 Import / Export

3.5.12.1 Users

Specific configurations for users import procedure:

- **IMPORTEXPONENT.USER.DEFAULT_PASSWORD**: password set for all users imported by the import procedure.

3.5.13 Main Menu

Specific settings for the main menu can be set updating the fields below:

- **KNOWAGE.DOWNLOAD.POLLING.TIME**: This field defines the number of milliseconds for the download service polling interval. If set to 0 the service will update just once on page load. Default is 10000 (10 seconds).

3.5.14 Changing the secret key for password encryption

The secret password encryption key must be set during the installation and must **never** be changed. *In case that the secret key is lost* you must create a new one and update database passwords. For this reason Knowage provides you a tool to find out the new encrypted value.

This tool requires:

- knowage-utils-<major.minor.patch>.jar (e.g. knowage-utils-8.0.1.jar) library to be added to the classpath
- the password encryption secret file name with complete path
- password value (plaintext)

Below is an example of invoking the tool using *biadmin* as plaintext password.

```
java -cp "TOMCAT_HOME/webapps/knowage/WEB-INF/lib/knowage-utils-<major.minor.patch>.jar" it.eng.spagobi.  
security.utils.PasswordEncryptionToolMain password/encryption/secret/file/name/with/complete/path biadmin
```

The output value will be the second argument passed in input encrypted with the key present in the file. This procedure must be repeated for all users.

3.5.15 Audit Table Management

The audit table tracks documents changes and other operations made to knowage data at the database level. User can enable audit table tracing (database table name: **SBI_AUDIT**) setting the **KNOWAGE.AUDIT_ENABLED** property to **true** inside **Configuration Management** panel:

Configuration Management					
audit					
Name ↑↓	Label ↑↓	Category ↑↓	Value ↑↓	Is active ↑↓	
KNOWAGE.AUDIT_ENABLED	KNOWAGE.AUDIT_ENABLED	GENERIC_CONFIGURATION	true	true	
KNOWAGE.AUDIT_DATA_RETENTION	KNOWAGE.AUDIT_DATA_RETENTION	GENERIC_CONFIGURATION	180	true	

Fig. 3.14: Enabling audit table tracing.

The **KNOWAGE.AUDIT_DATA_RETENTION** property is used to set the retention period, in order to manage audit table cleanup.