Documentation

OTRS Business Solution™ Manual

Build Date:

2014-12-05

OTRS Business Solution™ Manual

Table of Contents

Chapter 1. Introduction

General information about **OTRS Business Solution**™...

Chapter 2. Upgrading to OTRS Business Solution™

How to upgrade, update, and downgrade...

Chapter 3. Features of OTRS Business Solution™

1. Time Line View for Tickets

…

1.1. Description

…

1.2. Configuration

…

1.3. Usage

…

2. Attachment View for Tickets

…

2.1. Description

…

2.2. Configuration

…

2.3. Usage

…

3. Chat

…

3.1. Description

…

3.2. Configuration

…

3.3. Usage

…

4. SLA Selection Dialog

…

4.1. Description

…

4.2. Configuration

…

4.3. Usage

…

5. DynamicField "Contact With Data"

This feature allows to add contacts with data to tickets.

5.1. Description

5.1.1. Definition of data sources

Using a new DynamicField type any number of DynamicFields can be created, each of which is an individual data source for contacts with data.

Within any data source a number of available fields with key and value can be defined and a sort order, required fields and searchable fields can be configured.

5.1.2. Contact management

Via a new menu module any number of contacts can be created and managed for every data source.

For these contacts the configured attributes within the respective data source can be used.

5.1.3. Adding contacts to tickets

For every configured DynamicField one of the created contacts can be selected and assigned to a ticket.

The contact attributes will be shown in the ticket zoom.

5.1.4. Usage of contacts in search and statistics

Tickets can be confined in search and statistics via the required attribute 'Name'.

The name can also be returned as a field.

5.2. Configuration

The following list of configuration options can be changed for this feature.

5.2.1. `AdminContactWithData::RunInitialWildcardSearch`

Group: OTRSBusiness, Subgroup: Frontend::Admin::AdminContactWithData

Defines if a search with placeholder should be executed when initially calling the contact management.

5.2.2. `Frontend::Module###AdminContactWithData`

Group: OTRSBusiness, Subgroup: Frontend::Agent::ModuleRegistration

Allows to distribute contact and contact detail management permissions via group definition.

5.3. Usage

An exemplary usage of contacts with data is as follows:

- Creating a dynamic field of type 'Contact with data' and configuring the field for desired ticket modules.
- Setting possible contact attributes for the new field as well as noting which fields should be required, which fields should be usable for searching of contacts and the display order of the fields. The attributes 'Name' and 'ValidID' are always necessary.
- Adding and if necessary modifying contacs of the new field via 'Tickets' -> 'Edit contacts with data'

- Calling a ticket mask which is configured for the new field.
- Selecting an existing contact using autocomplete and choosing a contact.
- The assigned contact and its attributes will be shown in the ticket zoom.
- Should contact attributes be modified, the link below the attributes can be used (if permissions are set).
- If necessary, a different contact can be assigned to the ticket via ticket masks.

6. DynamicField "Database"

This feature implements a dynamic field of the type 'Database'.

6.1. Description

This feature implements a generic dynamic fieldtype, which offers the possibility to gather data from external databases. Such connected datasets can be searched and filtered using additional masks.

Related found and marked datasets can be saved to the particular tickets through the dynamic field.

'Database' dynamic fields can be created the same way default dynamic fields are created.

6.2. Configuration

The following list of configuration options can be used for this feature.

6.2.1. Add a dynamic field 'Database'

This feature implements a configuration interface to create dynamic fields of the type 'Database'.

'Database' dynamic fields can be created the same way default dynamic fields are created. For this switch to the ADMIN -> Ticket Settings -> Dynamic Fields view. In this mask you can select the 'Database' field from the ticket drop down box on the left side. Currently it's not possible to use the 'Database' dynamic field in the article context.

Configuration "General - Name": Dynamic fields of the type 'Database' need a unique name just like other dynamic fields, too. This name has to contain only alphanumeric values. This name will be used for internal handling of the field but will not be displayed.

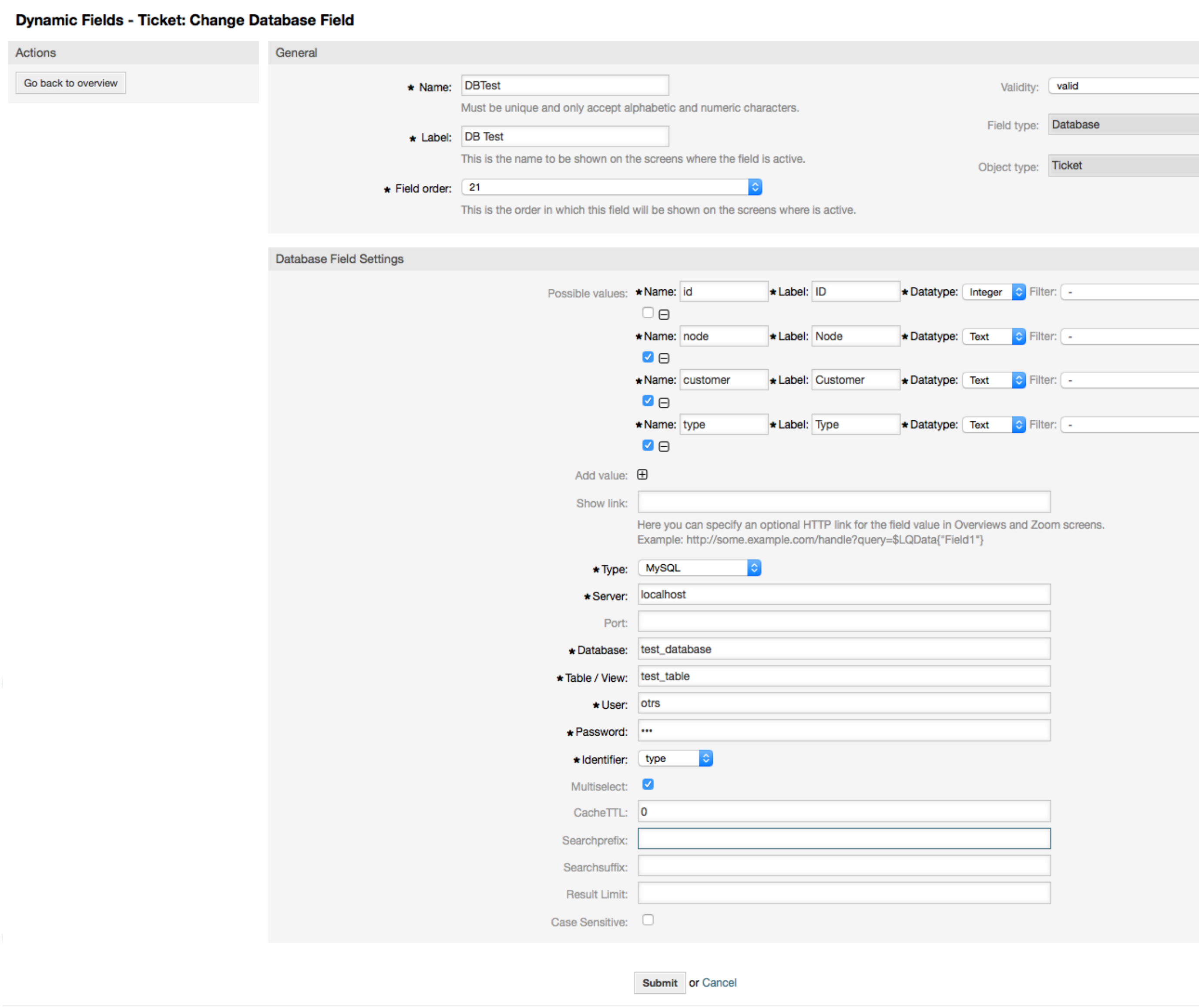
Configuration "General - Label": The label can be individually set and can contain white spaces etc. It will be used as the field label in the different views.

Configuration "General - Field order": The field order allows a administrator to change the order of created dynamic fields. If this configuration get changed the general field order will be adjusted and other dynamic fields will be moved one position back.

Configuration "General - Valid": To use the dynamic field in the configured OTRS 'views' it has to be set as valid. If the field is set to 'invalid' it will disappear from all the configured views but no data will be lost.

6.2.2. Configuration of the external source (Database)

Before an external database can be searched and the results be saved at the ticket through the dynamic field, the credentials have to be stored in the configuration of the dynamic field.



Dynamic Fields - Ticket: Change Database Field

Actions

Go back to overview

General

★ Name: DBTest
Must be unique and only accept alphabetic and numeric characters.

★ Label: DB Test
This is the name to be shown on the screens where the field is active.

★ Field order: 21
This is the order in which this field will be shown on the screens where is active.

Validity: valid

Field type: Database

Object type: Ticket

Database Field Settings

Possible values:

★Name: id	★Label: ID	★Datatype: Integer	Filter: -
★Name: node	★Label: Node	★Datatype: Text	Filter: -
★Name: customer	★Label: Customer	★Datatype: Text	Filter: -
★Name: type	★Label: Type	★Datatype: Text	Filter: -

Add value:

Show link:
Here you can specify an optional HTTP link for the field value in Overviews and Zoom screens.
Example: http://some.example.com/handle?query=$LQData{"Field1"}

★Type:	MySQL
★Server:	localhost
Port:	
★Database:	test_database
★Table / View:	test_table
★User:	otrs
★Password:	•••
★Identifier:	type
Multiselect:	✓
CacheTTL:	0
Searchprefix:	
Searchsuffix:	
Result Limit:	
Case Sensitive:	

Submit or Cancel

Type: The type of the desired database can be selected here. The field supports the default OTRS database types mysql, oracle, postgres or mssql.

SID: This option is only available for Oracle connections and will be shown or hidden automatically. Within this option you have to enter the SID of your Oracle connection.

Driver: This option is only available for ODBC connections and will be shown or hidden automatically. Within this option you enter the previously in the hostsystem configured ODBC driver to connect to the desired MSSQL database.

Server: The database host (hostname or IP-Address).

Port: The port of the database server

Database: Defines the desired target database of the DBMS. This database will be used for queries.

Table / View: This table or view will be used for the queries.

User: The username for the database connection.

Password: The user password for the database connection.

Identifier: This select box will be automatically filled through "Possible Values (description below)". This field represents the value which will be stored in the dynamic field.

Multiselect: If this field is selected, it will be possible to store more than one value to the dynamic field. Those values will be stored comma seperated.

CacheTTL: This value defines the period of validity of the database cache in seconds. Equal queries to the database will be answered through the cache (local filesystem) within this period instead of asking the database again.

Searchprefix: This value will be put in the front of every search term while using the autocompletion to search the database. Wildcard characters are supported as well. The searchprefix will be ignored during the detailed search, but it is still possible to use wilcard characters in those masks.

Searchsuffix: This value will be put in the end of every search term while using the autocompletion to search the database. Wildcard characters are supported as well. The searchsuffix will be ignored during the detailed search, but it is still possible to use wilcard characters in those masks.

Result-Limit: The entered integer value defines the maximum amount of allowed results during a database search. This includes the autocompletion search as well as the detailed search.

Case Sensitive: If this field is selected, case-sensitivity will take effect on searches.

Possible values: As already explained the possible values will fill up the identifier field automatically, which defines the value that will be stored in the dynamic field. Possible values can be created as much as needed (or at least as many table columns as the database table has.). The possible values defines the database columns to search in. It is possible to set the column name, a description (label) the fields should have, the needed datatype and if the field should be a search- or listfield.

Name: The exact name of the database column which will be requested through the database queries.

Description: The label of the field which will be displayed in the detailed search.

Datatype: The datatype which will be stored in the dynamic field. Possible values: TEXT, INTEGER or DATE.

Filter: With the filter field, it is possible to choose a ticket attribute or a dynamic field as a filter for the related column. If the dynamic field is bound to a related ticket, the attributes will be used for the filter mechanism, otherwise the filters will be ignored. If filter will be configured to a table column, only search results matching to the search term and the related ticket attribute on exactly the configured column will be displayed.

Searchfield: Indicates if a field should be included in the search requests.

Listfield: Indicates if a field should be displayed in the results.

6.2.3. Configure dynamic fields of type 'Database' to be displayed in the agent interface

Dynamic fields of type 'Database' have to be activated for the several masks in which they should be displayed like the other types of dynamic fields.

This can be done through ADMIN -> System Administration -> SysConfig, in which "Ticket" must be selected on the left hand site.

For every interface area (Frontend), in which the dynamic field of type 'Database' should be displayed, the admin has to configure it to fit his needs. Examples:

Frontend::Agent::Ticket::ViewZoom for the ticket zoom view.

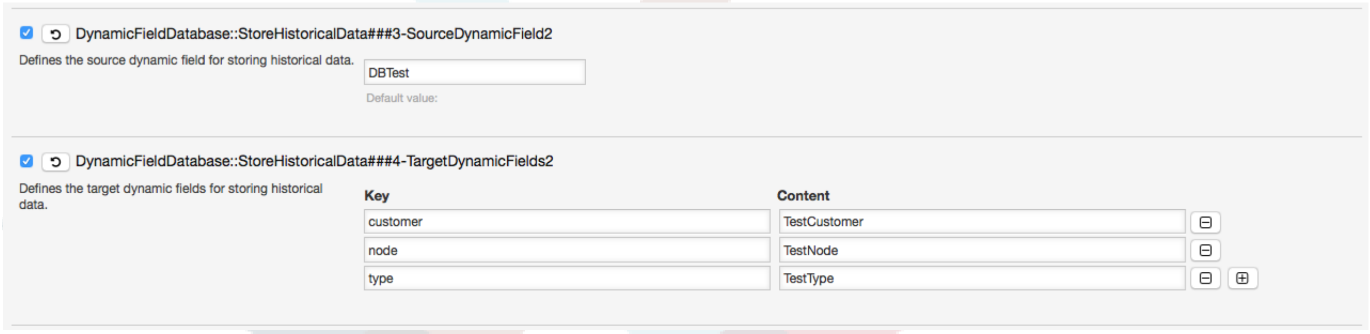
Frontend::Agent::Ticket::ViewPhoneNew for new phone tickets

Frontend::Agent::Ticket::ViewEmailNew for new email tickets.

In each of these view configurations is a entry called "Ticket::Frontend::AgentTicket*###DynamicField". This configuration defines which dynamic field(s) should get displayed in this view. To add a dynamic field the internal name of the field has to be filled in the 'Key' field. The field 'Value' can take the values 0 (deactivated), 1 (active) and 2 (active and mandatory).

6.2.4. Storage of historical data

This feature offers a functionality to store historical data. For this to work it´s necessary to activate and set SysConfig options, as visibly in the following screenshot:



DynamicFieldDatabase::StoreHistoricalData###3-SourceDynamicField2

Defines the source dynamic field for storing historical data.

DBTest

Default value:

DynamicFieldDatabase::StoreHistoricalData###4-TargetDynamicFields2

Defines the target dynamic fields for storing historical data.

Key	Content
customer	TestCustomer
node	TestNode
type	TestType

In the configuration option for the "SourceDynamicField" it´s needed to fill in the already created dynamic (Database) field name, which will be used to gather the historical data. In the related option "TargetDynamicField" the field(s) "Key" have to be filled with the table columns of the connected external database, which will be readout. For every column the related target dynamic field has to be configured in the field "content". The gathered data will be saved in these dynamic fields.

If the configuration is ready and active, the configured fields will be readout of the external database, since the source field gets a new value via the configured masks. The data will be searched by it´s stored identifier via an event module and the found values will be stored in the target dynamic fields.

6.2.5. Sysconfig settings

6.2.5.1. AutoComplete::Agent###DynamicFieldDatabaseSearch.

Group: Framework, Subgroup: Frontend::Agent.

Defines the config options for the autocompletion feature.

6.2.5.2. Ticket::EventModulePost###950-StoreHistoricalData.

Group: Ticket, Subgroup: Core::Ticket.

Update dynamic fields, if configured ones will be updated.

6.2.5.3. DynamicFieldDatabase::StoreHistoricalData###1-SourceDynamicField1.

Group: OTRSBusiness, Subgroup: Core.

Defines the source dynamic field for storing historical data.

6.2.5.4. DynamicFieldDatabase::StoreHistoricalData###2-Target-DynamicFields1.

Group: OTRSBusiness, Subgroup: Core.

Defines the target dynamic fields for storing historical data.

6.2.5.5. DynamicFieldDatabase::StoreHistoricalData###3-Source-DynamicField2.

Group: OTRSBusiness, Subgroup: Core.

Defines the source dynamic field for storing historical data.

6.2.5.6. DynamicFieldDatabase::StoreHistoricalData###4-Target-DynamicFields2.

Group: OTRSBusiness, Subgroup: Core.

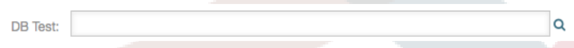
Defines the target dynamic fields for storing historical data.

6.3. Usage

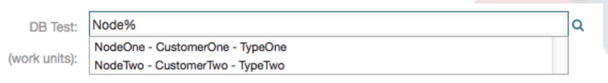
An exemplary usage of DynamicField Database is as follows:

6.3.1. Searching and saving datasets - Autocompletion

After the created dynamic fields are activated in the well known masks (like ViewPhoneNew, ViewEmailNew) a new textfield appears with the name, the dynamic field got in the configuration. In this field it is possible to input searchterms and therefore execute a search over all configured database fields or otherwise do a click on the link 'Detailed search' and start a detailled search in which the fields to search in are selected explicitly.



Since search terms are typed in into the textfield, a database search will be started over the configured columns and the result will displayed via an autocompletion below the textfield. The more exact the search term is, the more exact will be the result (less result entries).

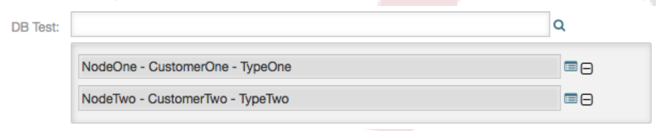

DB Test: Node%

(work units):

NodeOne - CustomerOne - TypeOne

NodeTwo - CustomerTwo - TypeTwo

If the wished value will be displayed in the results, it can be selected via a mouse click or via the keyboard and therefore be added to the dynamic field results.


DB Test:

NodeOne - CustomerOne - TypeOne

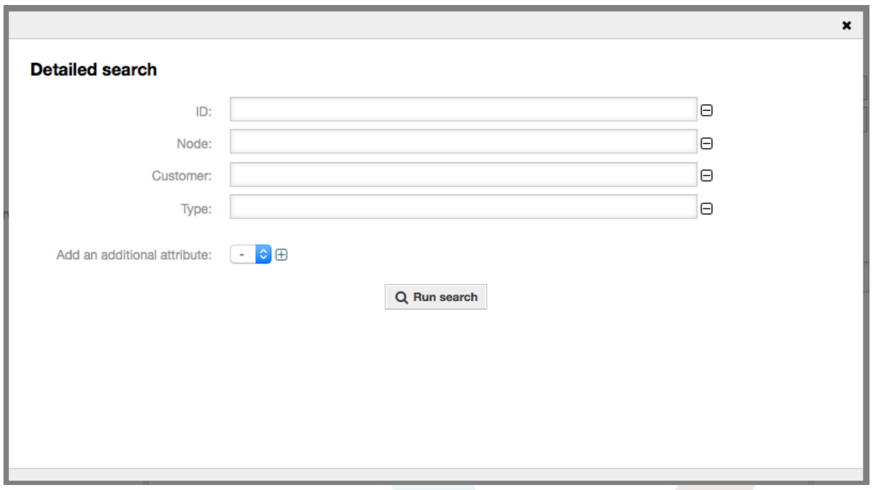
NodeTwo - CustomerTwo - TypeTwo

Via the link 'Details' a pop can be accessed, which offers detailed information about the whole result row. This information includes the line headers and the data. This information can be used to get an overview about the rest (of the not configured) columns or to compare data.

The added result entries can be removed via the minus button.

6.3.2. Searching and saving datasets - Detailed search

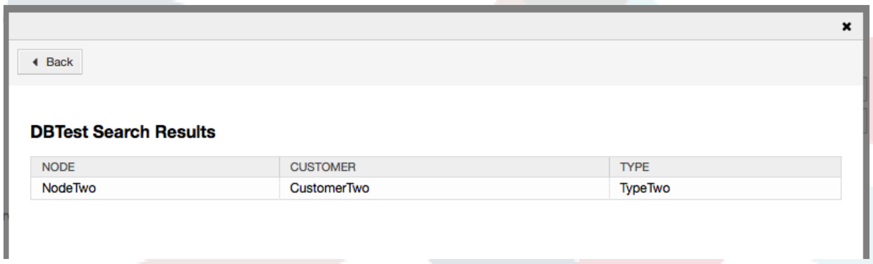
The link 'Detailed search' opens a new modal dialog to start a new database search. In this mask it is possible to select the fields to search on explicitly.


Detailed search

ID:	
Node:	
Customer:	
Type:	
Add an additional attribute:	-

By default the first available field is activated, but it´s also possible to remove available fields or add additional ones. Only activated and filled fields are considered for the search. Wildcard characters '*' are allowed in every single field.

The database search will be executed via the button "Start search" and the results will be tabular displayed. If the search was successful, the results will be listed and one of the entries can be selected via a mouse click. The value will be added to the list of saved values afterwards.



DBTest Search Results

NODE	CUSTOMER	TYPE
NodeTwo	CustomerTwo	TypeTwo

Independent of using the autocompletion or the detailled search, every single result can just selected ones. It an agent tries to select a value multiple times, a related warning message is displayed.