Spider's web

…share your own experience

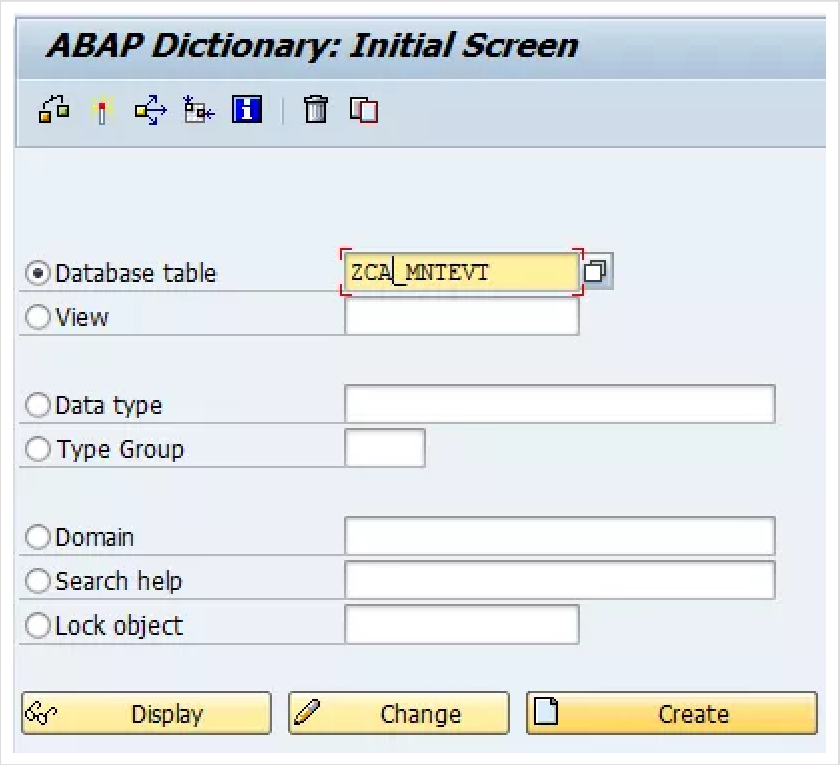
Events in database table maintenance view

Posted on 2013-04-12

I needed to create a database table where there will be no key fields (all fields to be editable during table maintenance) but at the same time I needed some easy way how to identify each table row with unique key. And I found a solution using automatically generated GUID during row addition handled by maintenance view events.

Create table

We will start from scratch so we'll design and create our database table called in our case ZCA_MNTEVT.



ABAP Dictionary: Initial Screen

● Database table ZCA_MNTEVT

○ View

○ Data type

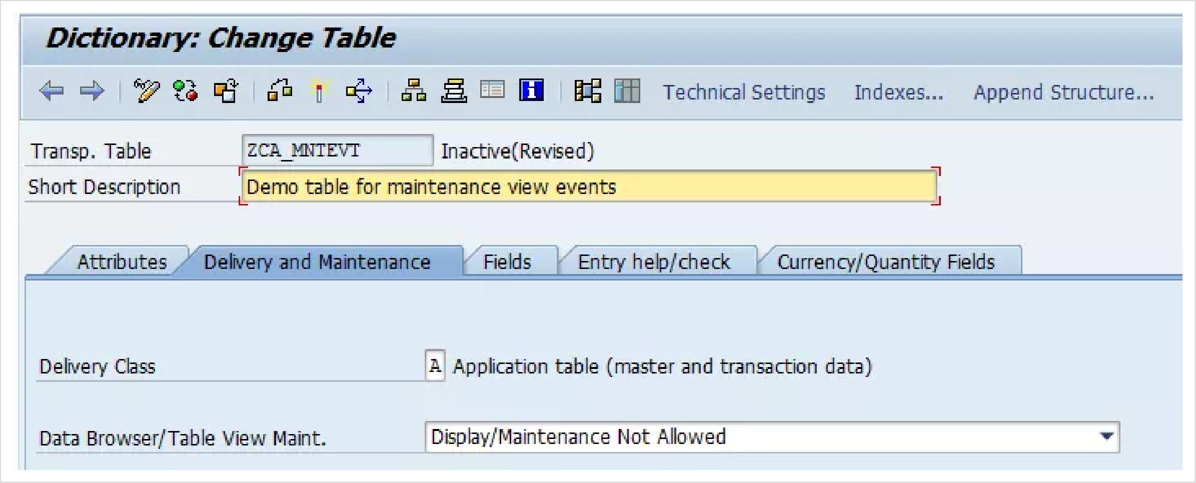
○ Type Group

○ Domain

○ Search help

○ Lock object

We will disallow it’s display and maintenance because these actions will be performed by it’s maintenance view only.



Dictionary: Change Table

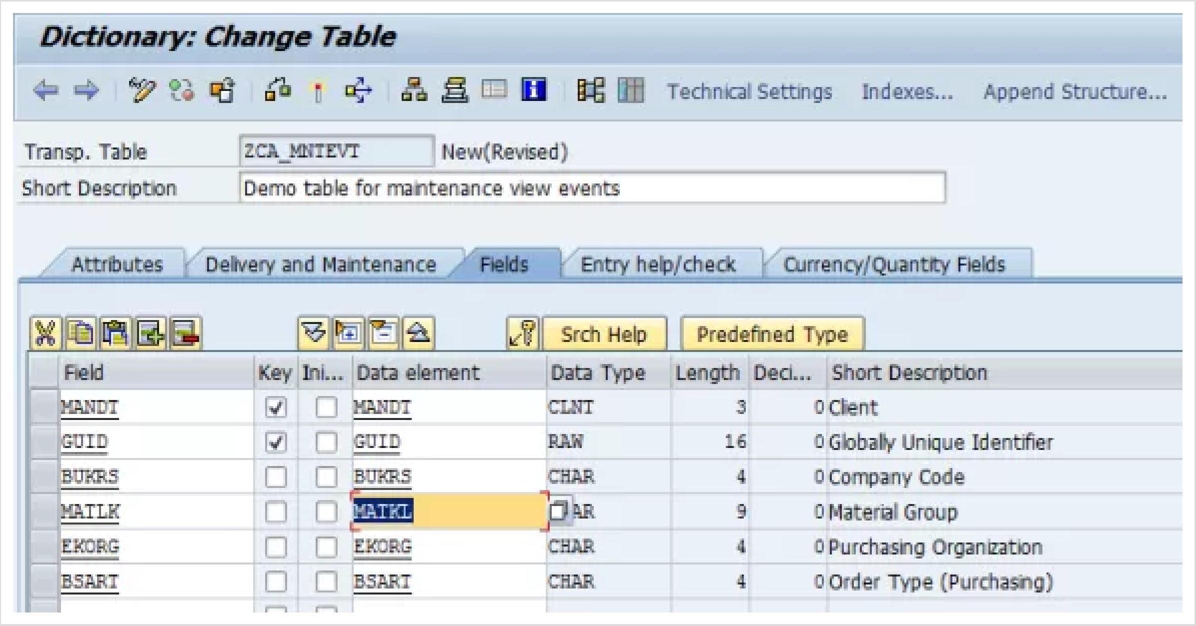
Transp. Table ZCA_MNTEVT Inactive(Revised)

Short Description Demo table for maintenance view events

Delivery Class A Application table (master and transaction data)

Data Browser/Table View Maint. Display/Maintenance Not Allowed

Now we add the necessary fields

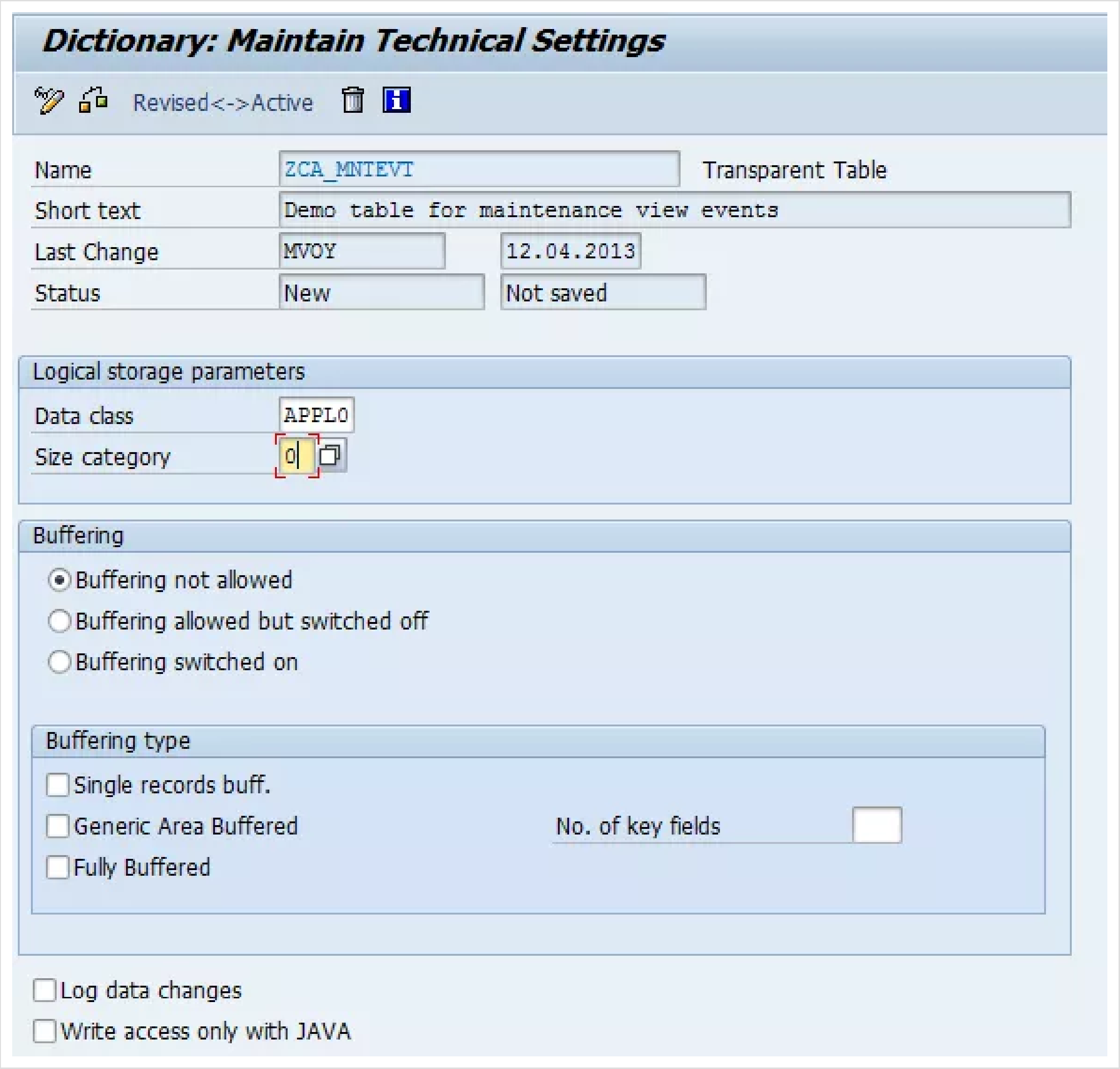


Transp. Table ZCA_MNTEVT New(Revised)

Short Description Demo table for maintenance view events

Field	Key	Ini...	Data element	Data Type	Length	Deci...	Short Description
MANDT	☑	☐	MANDT	CLNT	3	0	Client
GUID	☑	☐	GUID	RAW	16	0	Globally Unique Identifier
BUKRS	☐	☐	BUKRS	CHAR	4	0	Company Code
MATLK	☐	☐	MATKL	AR	9	0	Material Group
EKORG	☐	☐	EKORG	CHAR	4	0	Purchasing Organization
BSART	☐	☐	BSART	CHAR	4	0	Order Type (Purchasing)

… set technical settings



Name	ZCA_MNTEVT	Transparent Table
Short text	Demo table for maintenance view events	
Last Change	MVOY	12.04.2013
Status	New	Not saved

Logical storage parameters

Data class	APPL0
Size category	0

Buffering

● Buffering not allowed

○ Buffering allowed but switched off

○ Buffering switched on

Buffering type

□ Single records buff.

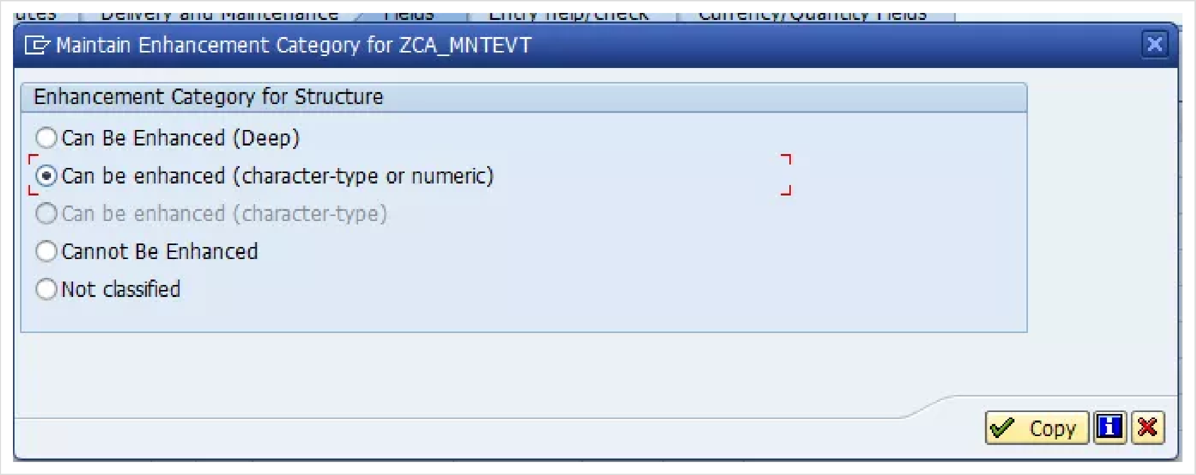
□ Generic Area Buffered   No. of key fields

□ Fully Buffered

□ Log data changes

□ Write access only with JAVA

… and set enhancement category (in top menu – *Extras* – *Enhancement category*)



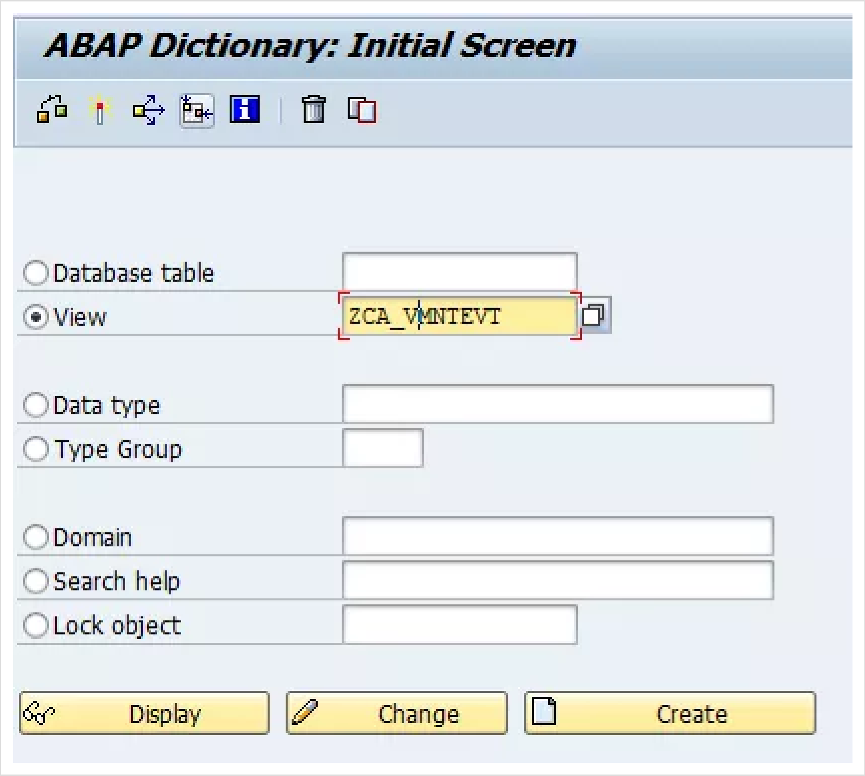
Maintain Enhancement Category for ZCA_MNTEVT

Enhancement Category for Structure

○ Can Be Enhanced (Deep)
● Can be enhanced (character-type or numeric)
○ Can be enhanced (character-type)
○ Cannot Be Enhanced
○ Not classified

Create table view on top of our table

We will name the view similarly to the database table, ie ZCA_VMNTEVT.



ABAP Dictionary: Initial Screen

○ Database table

● View: ZCA_VMNTEVT

○ Data type

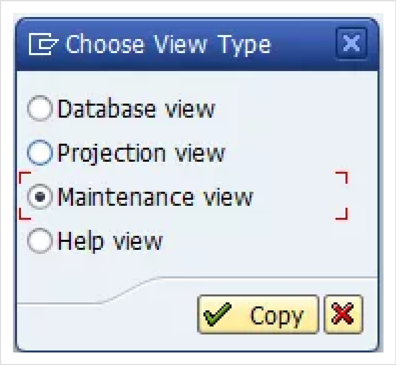
○ Type Group

○ Domain

○ Search help

○ Lock object

… and create it as maintenance view



Choose View Type

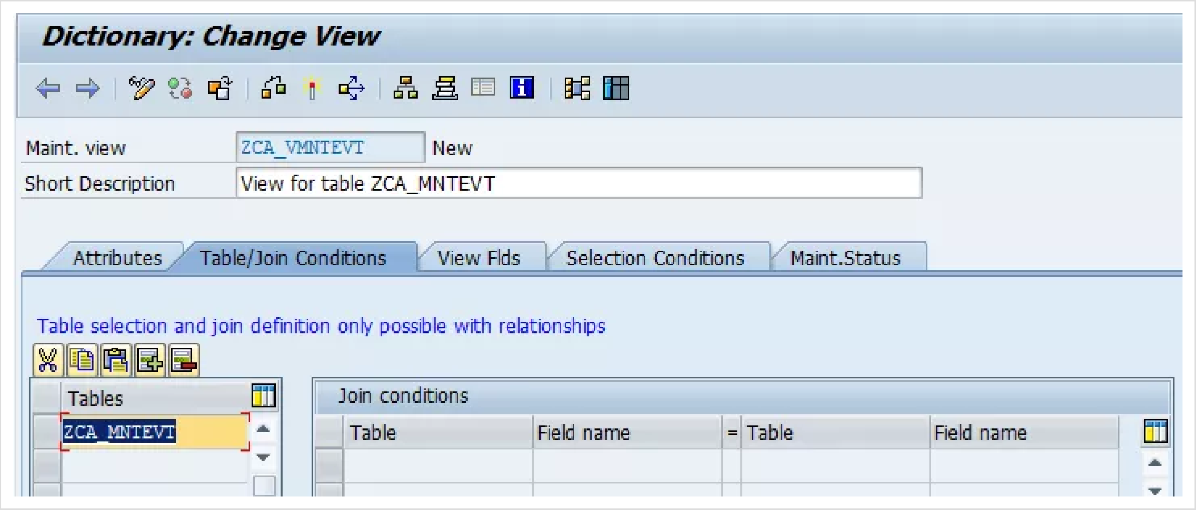
○ Database view

○ Projection view

● Maintenance view

○ Help view

In tab *Table/Join* conditions we enter the name of our database table.



Dictionary: Change View

Maint. view: ZCA_VMNTEVT New

Short Description: View for table ZCA_MNTEVT

Table selection and join definition only possible with relationships

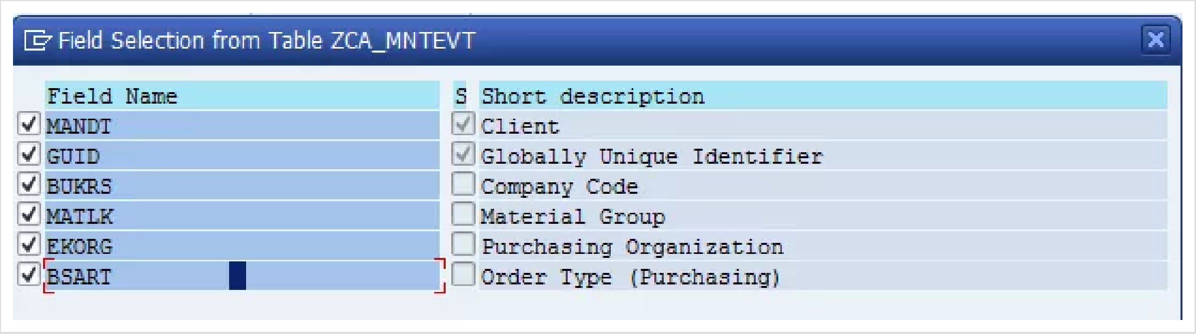
Tables
ZCA_MNTEVT

Join conditions

Table	Field name	=	Table	Field name

Now click on the *View Flds* tab, click on *Table fields* button and select the necessary fields (fileds you want to have in the maintenance view available to be edited)

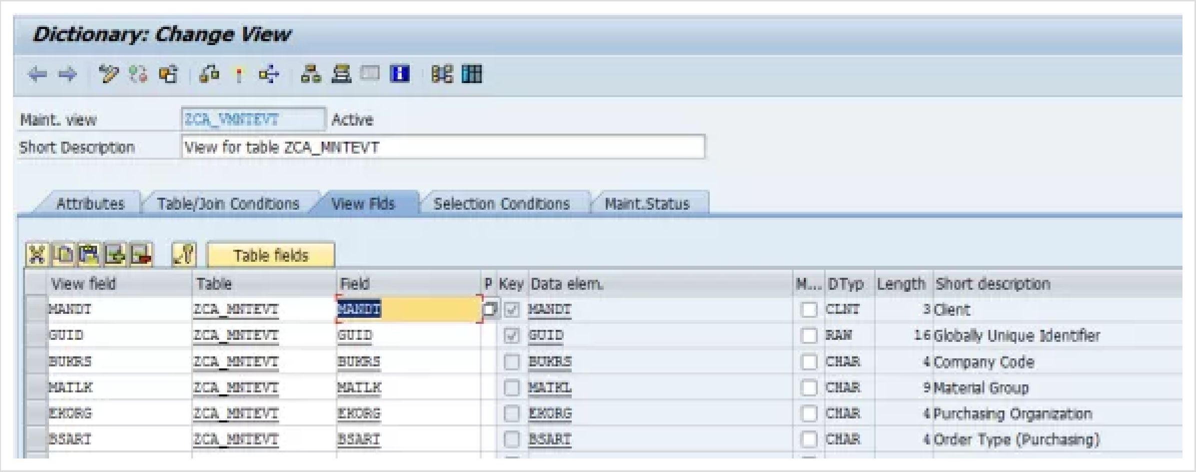
Note: We don't want the MANDT and GUID to be displayed because they will be automatically assigned/generated but we need to have them in the view so we can manipulate with data in the fields in the code.



Field Selection from Table ZCA_MNTEVT

	Field Name	S	Short description
☑	MANDT	☑	Client
☑	GUID	☑	Globally Unique Identifier
☑	BUKRS	□	Company Code
☑	MATLK	□	Material Group
☑	EKORG	□	Purchasing Organization
☑	BSART	□	Order Type (Purchasing)

The final look of available fields will be as on the following picture



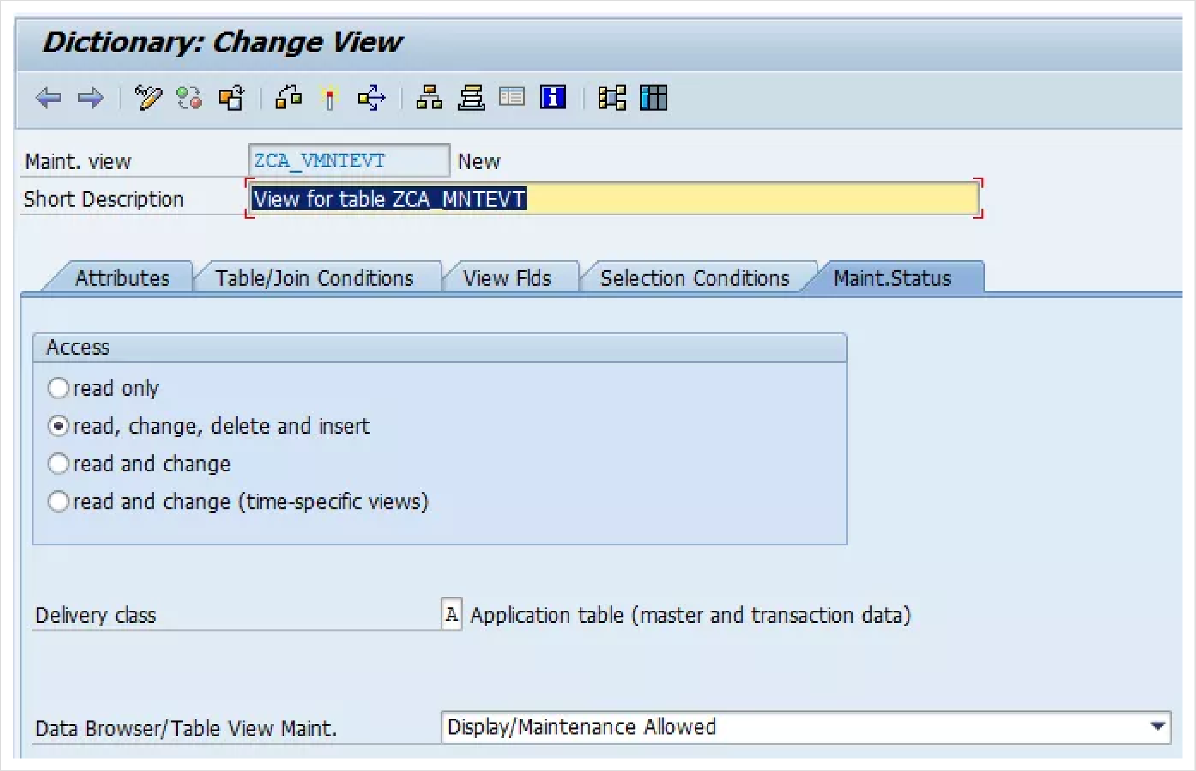
Dictionary: Change View

Maint. view ZCA_VMNTEVT Active

Short Description View for table ZCA_MNTEVT

View field	Table	Field	P	Key	Data elem.	M...	DTyp	Length	Short description
MANDT	ZCA_MNTEVT	MANDT		☑	MANDT	□	CLNT	3	Client
GUID	ZCA_MNTEVT	GUID		☑	GUID	□	RAW	16	Globally Unique Identifier
BUKRS	ZCA_MNTEVT	BUKRS		□	BUKRS	□	CHAR	4	Company Code
MATLK	ZCA_MNTEVT	MATLK		□	MATKL	□	CHAR	9	Material Group
EKORG	ZCA_MNTEVT	EKORG		□	EKORG	□	CHAR	4	Purchasing Organization
BSART	ZCA_MNTEVT	BSART		□	BSART	□	CHAR	4	Order Type (Purchasing)

Now switch to tab *Maint.Status* where we'll set the required access level.



Dictionary: Change View

Maint. view ZCA_VMNTEVT New

Short Description View for table ZCA_MNTEVT

Access

○ read only

● read, change, delete and insert

○ read and change

○ read and change (time-specific views)

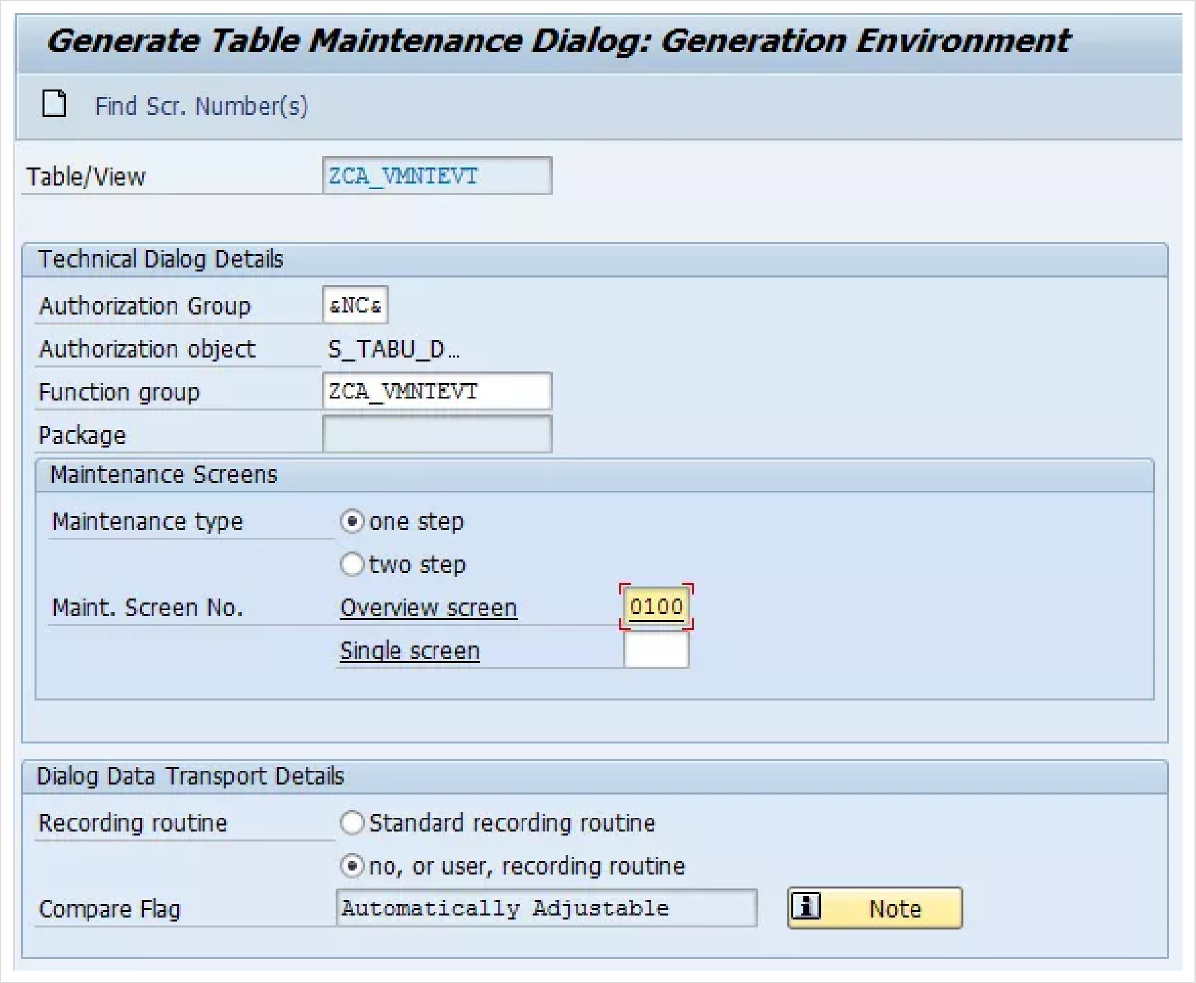
Delivery class A Application table (master and transaction data)

Data Browser/Table View Maint. Display/Maintenance Allowed

At this moment we're done with the view and we can save&activate it.

Create maintenance view for table view

Stay in the maintenance view screen and go to top menu – *Utilities* – *Table Maintenance Generator* where we will create *One step* maintenance type in new function group named with the same as the view: *ZCA_VMNTEVT.*



Generate Table Maintenance Dialog: Generation Environment

Table/View ZCA_VMNTEVT

Technical Dialog Details

Authorization Group &NC&

Authorization object S_TABU_D...

Function group ZCA_VMNTEVT

Package

Maintenance Screens

Maintenance type ● one step

○ two step

Maint. Screen No. Overview screen 0100

Single screen

Dialog Data Transport Details

Recording routine ○ Standard recording routine

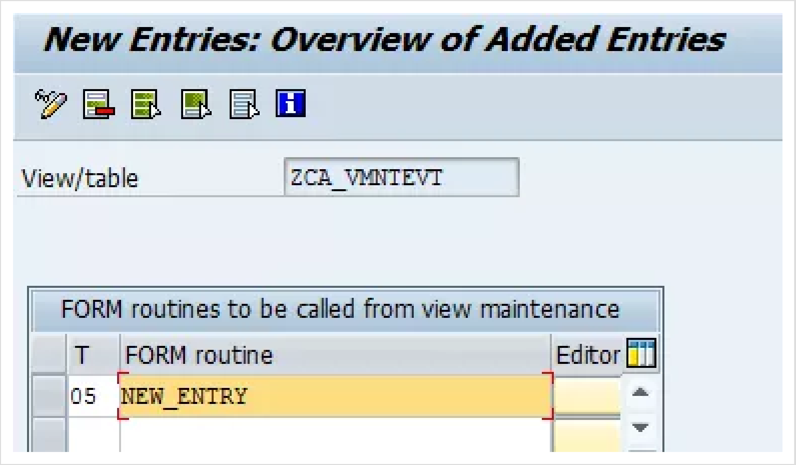
● no, or user, recording routine

Compare Flag Automatically Adjustable

Press F6 (create) to generate code of function group *ZCA_VMNTEVT* for the maintenance view.

Add events

When the generation is completed go to top menu – *Environment* – *Modification* – *Events,* create new entry for event of row creation/addition (event #05) and assign a form name where our code will be placed.
Note: On this picture you can see all available events you can use in case you need them.



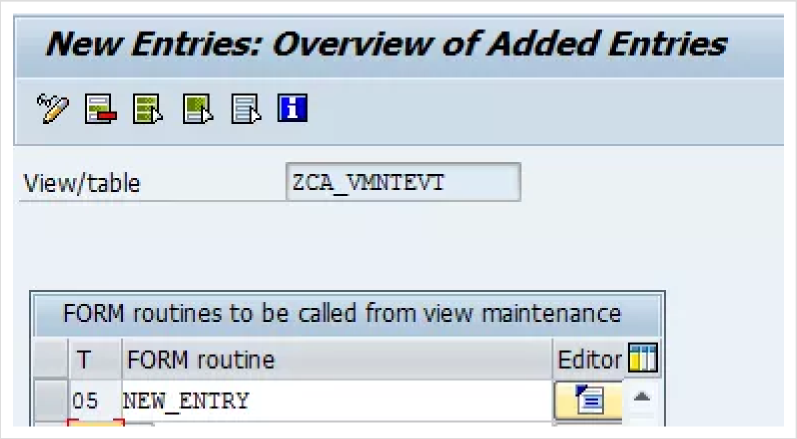
New Entries: Overview of Added Entries

View/table ZCA_VMNTEVT

FORM routines to be called from view maintenance

T	FORM routine	Editor
05	NEW_ENTRY	

Click on the SAVE button and then on the button in the *Editor* column.



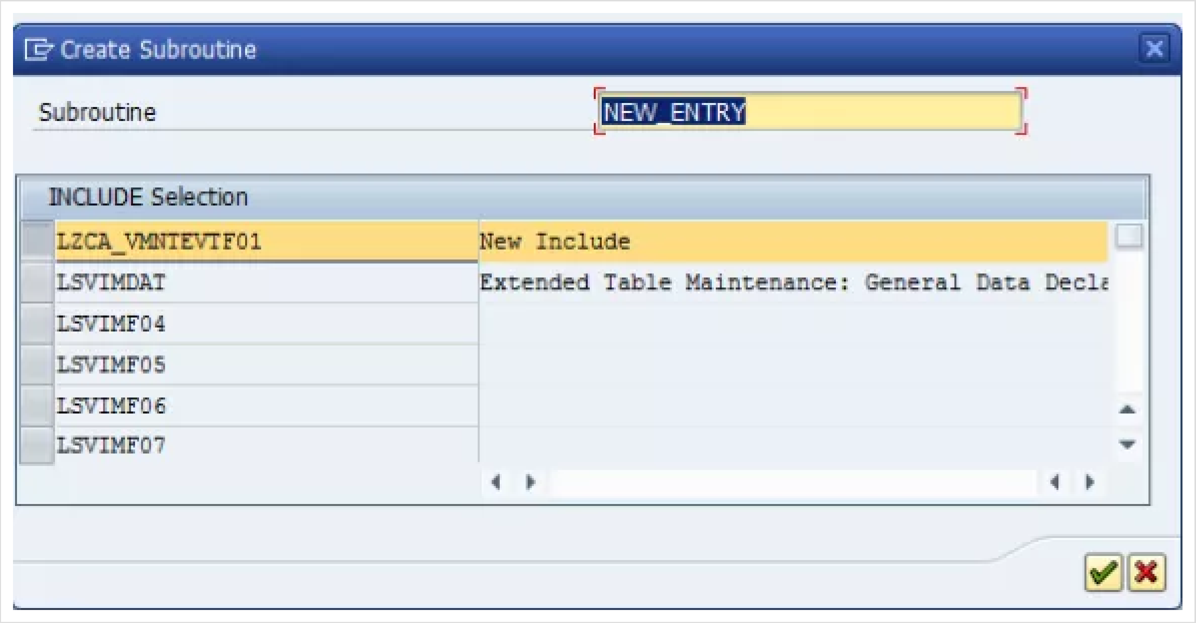
New Entries: Overview of Added Entries

View/table ZCA_VMNTEVT

FORM routines to be called from view maintenance

T	FORM routine	Editor
05	NEW_ENTRY	

You will be asked for a new include name where the code will be placed. Enter your desired name or just confirm the suggested one.



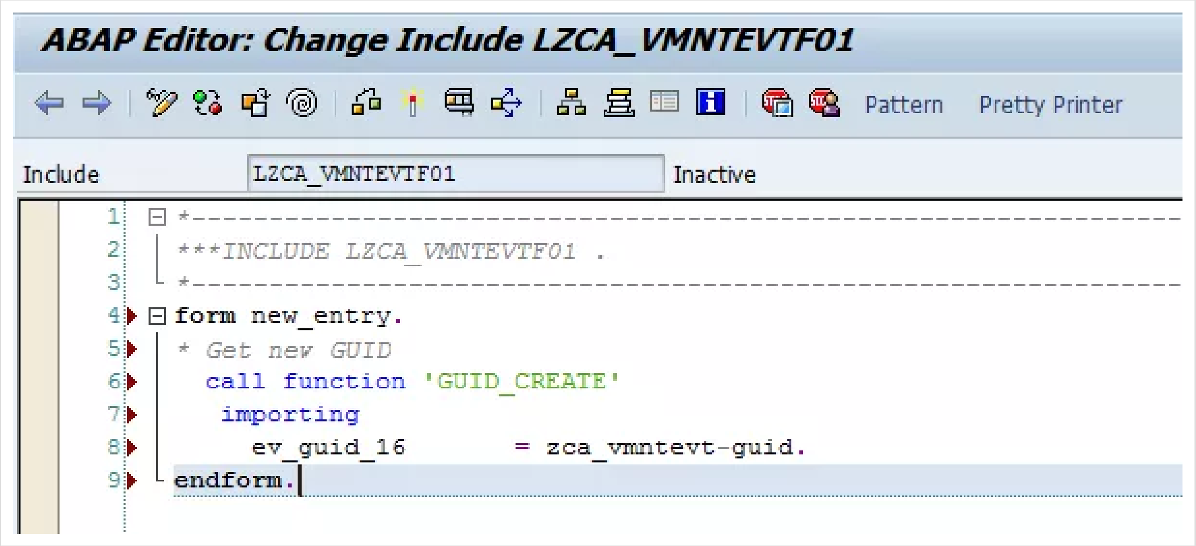
Create Subroutine

Subroutine NEW_ENTRY

INCLUDE Selection

LZCA_VMNTEVTF01	New Include
LSVIMDAT	Extended Table Maintenance: General Data Decla
LSVIMF04	
LSVIMF05	
LSVIMF06	
LSVIMF07	

When the code editor is displayed enter the following piece of code.



ABAP Editor: Change Include LZCA_VMNTEVTF01

Include LZCA_VMNTEVTF01 Inactive

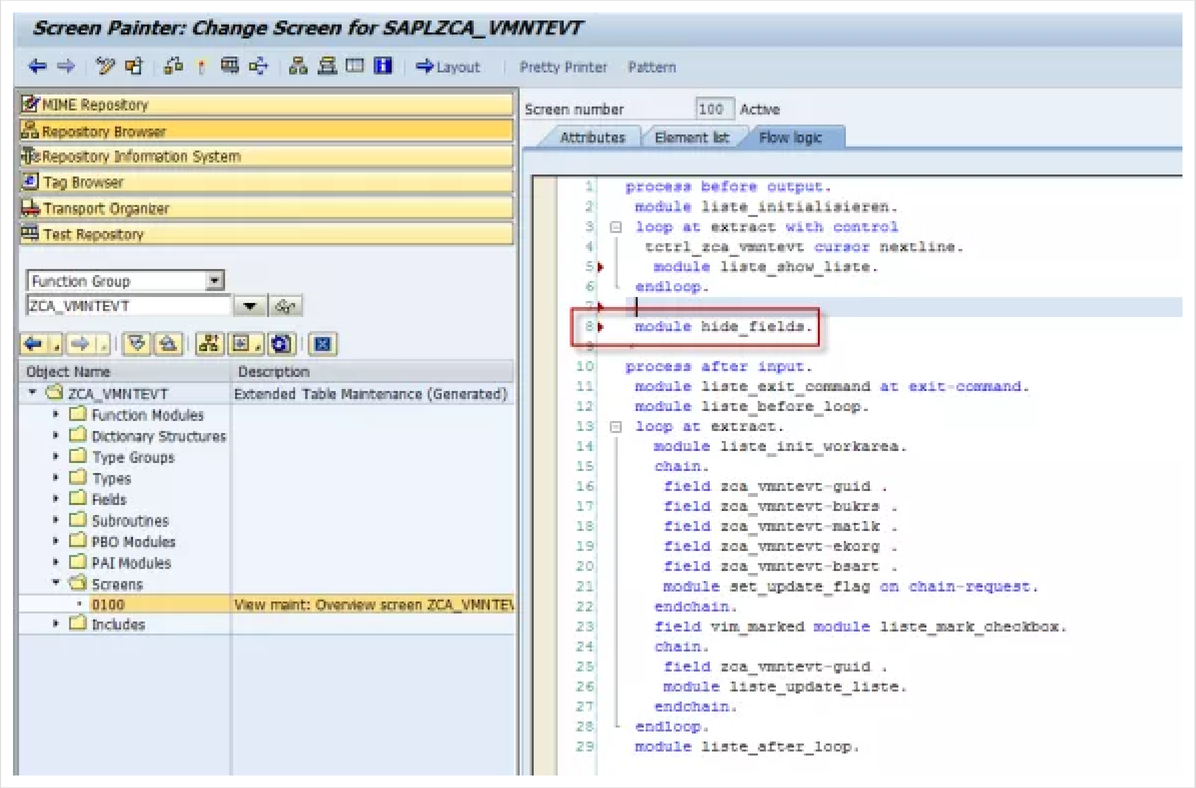
```
*----------------------------------------------------------------------
***INCLUDE LZCA_VMNTEVTF01 .
*----------------------------------------------------------------------
form new_entry.
* Get new GUID
  call function 'GUID_CREATE'
   importing
     ev_guid_16       = zca_vmntevt-guid.
endform.
```

Save and activate all your changes

Modify the maintenance view

Now if you run transaction SM30 and maintain ZCA_VMNTEVT you will see the GUID column is visible and editable. This is not what we've originally wanted. So we have to edit the maintenance view accordingly:

Run transaction SE80 and open function group ZCA_VMNTEVT, expand the *Screens* folder, open screen *0100* and add the highlighted line into the code



Function Group: ZCA_VMNTEVT

Object Name	Description
ZCA_VMNTEVT	Extended Table Maintenance (Generated)
0100	View maint: Overview screen ZCA_VMNTE\

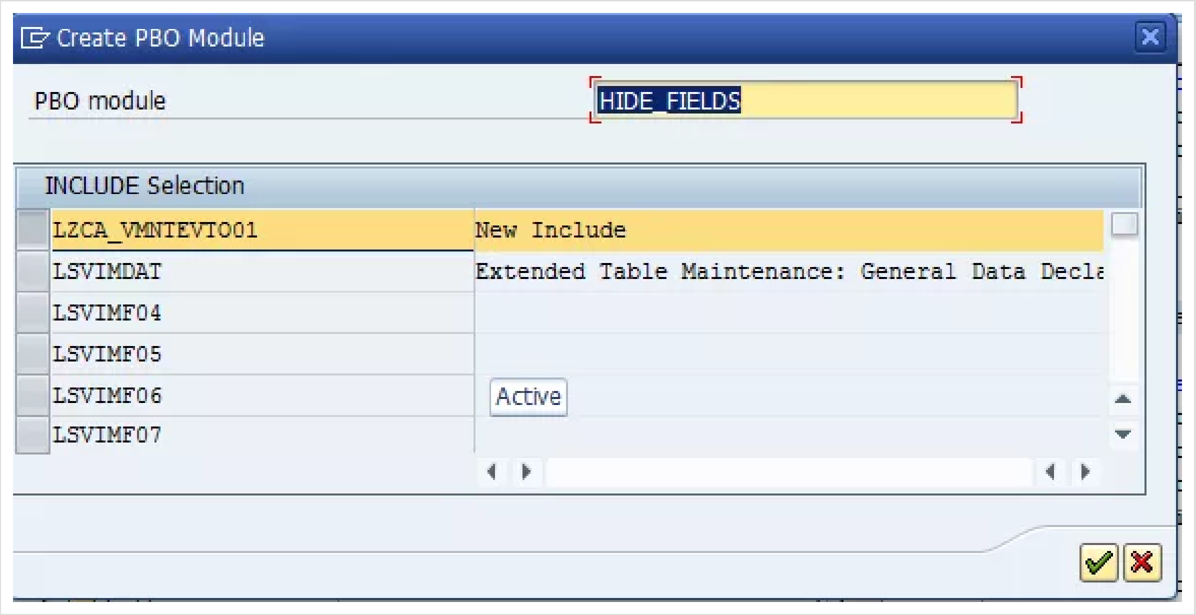
Screen number 100 Active

```
process before output.
 module liste_initialisieren.
 loop at extract with control
  tctrl_zca_vmntevt cursor nextline.
   module liste_show_liste.
 endloop.

 module hide_fields.

process after input.
 module liste_exit_command at exit-command.
 module liste_before_loop.
 loop at extract.
   module liste_init_workarea.
   chain.
    field zca_vmntevt-guid .
    field zca_vmntevt-bukrs .
    field zca_vmntevt-matlk .
    field zca_vmntevt-ekorg .
    field zca_vmntevt-bsart .
    module set_update_flag on chain-request.
   endchain.
   field vim_marked module liste_mark_checkbox.
   chain.
    field zca_vmntevt-guid .
    module liste_update_liste.
   endchain.
 endloop.
 module liste_after_loop.
```

Double click on the module and you'll be prompted for another include name. Enter desired name or just confirm.



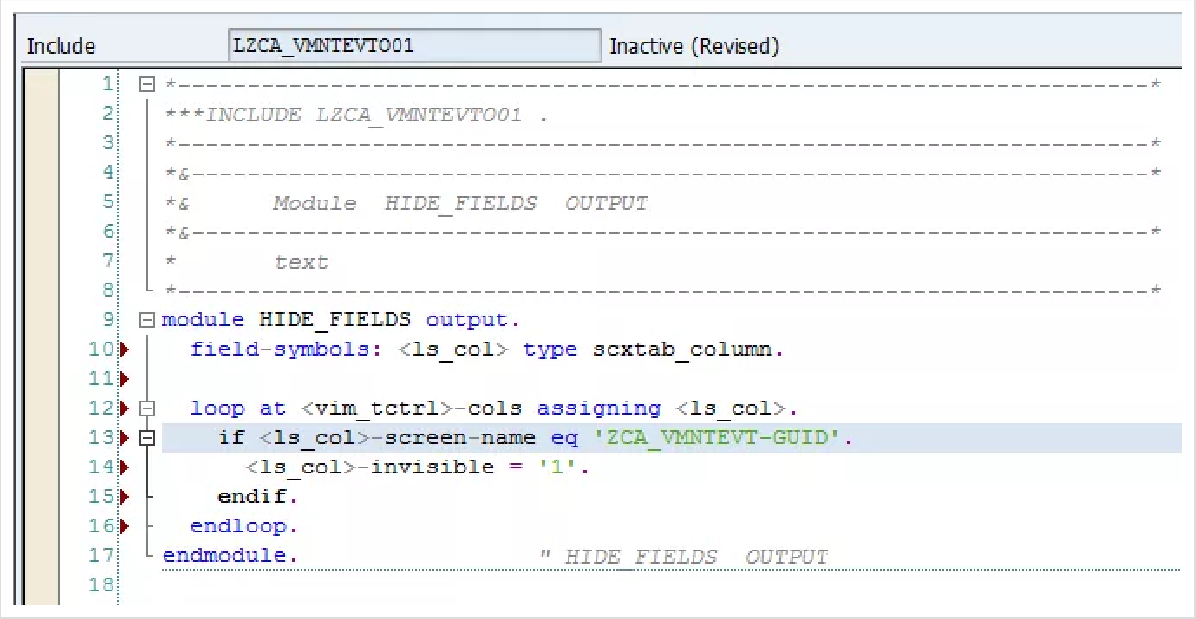
Create PBO Module

PBO module: HIDE_FIELDS

INCLUDE Selection

LZCA_VMNTEVT001	New Include
LSVIMDAT	Extended Table Maintenance: General Data Decla
LSVIMF04	
LSVIMF05	
LSVIMF06	Active
LSVIMF07	

When code editor is displayed enter the following piece of code.

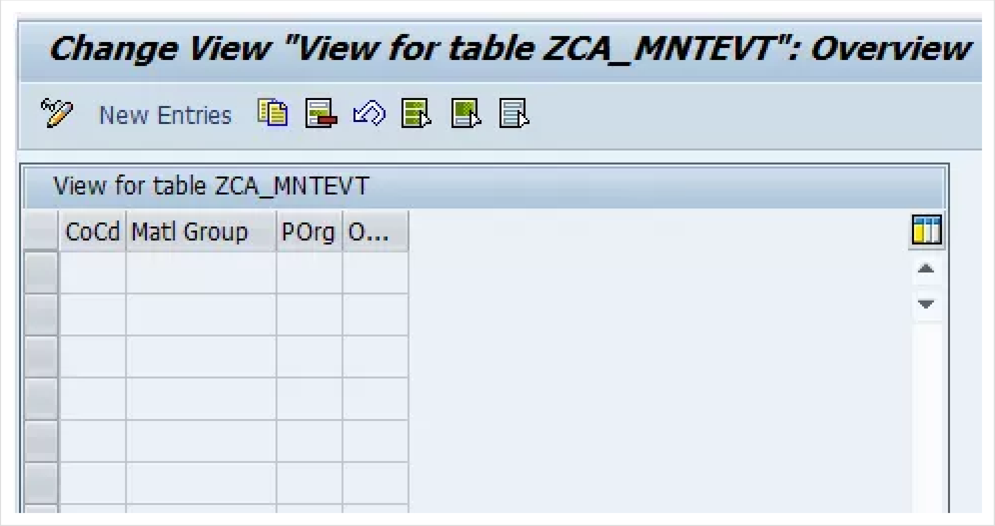


Include LZCA_VMNTEVT001 Inactive (Revised)

```
*----------------------------------------------------------------------*
***INCLUDE LZCA_VMNTEVT001 .
*----------------------------------------------------------------------*
*&---------------------------------------------------------------------*
*&      Module  HIDE_FIELDS  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
module HIDE_FIELDS output.
  field-symbols: <ls_col> type scxtab_column.

  loop at <vim_tctrl>-cols assigning <ls_col>.
    if <ls_col>-screen-name eq 'ZCA_VMNTEVT-GUID'.
      <ls_col>-invisible = '1'.
    endif.
  endloop.
endmodule.                 " HIDE_FIELDS  OUTPUT
```

Now save and activate all the changes, go to transaction *SM30* and maintain *ZCA_VMNTEVT.*
Now you can see the GUID column is not visible.

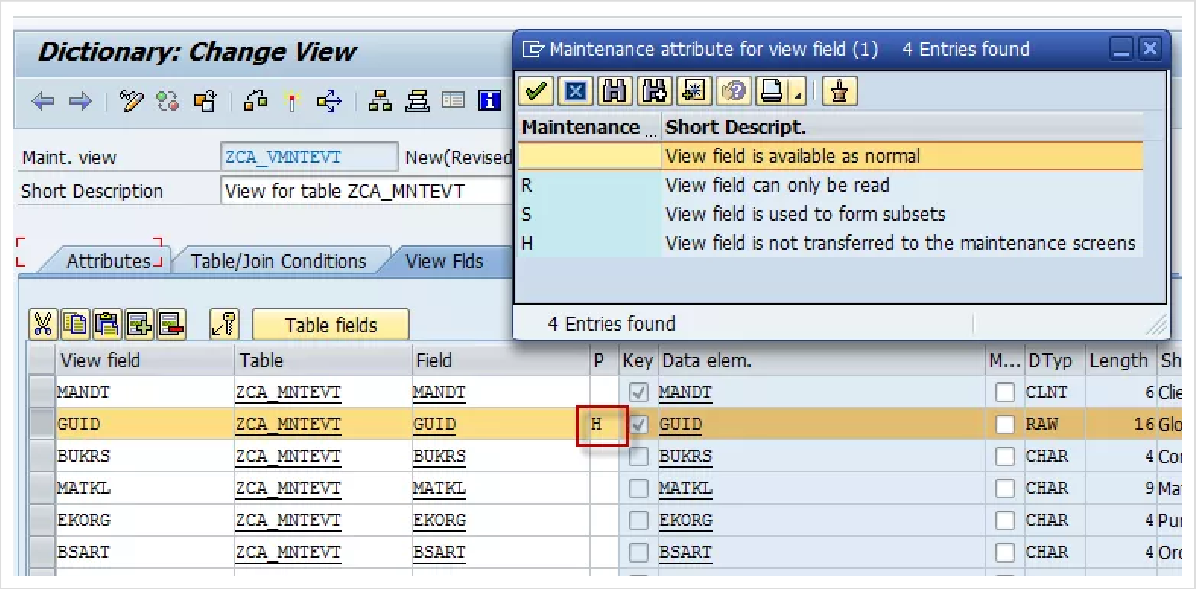


Change View "View for table ZCA_MNTEVT": Overview

View for table ZCA_MNTEVT

CoCd	Matl Group	POrg	O...

The above (hiding a column) was just to show you how you can manipulate data and look&feel of the screen programatically. The same (hiding a column of a view) can be achieved much simpler way – setting it’s visibility directly in the Maintenance view:



Dictionary: Change View

Maint. view ZCA_VMNTEVT New(Revised

Short Description View for table ZCA_MNTEVT

Maintenance attribute for view field (1) 4 Entries found

Maintenance ...	Short Descript.
	View field is available as normal
R	View field can only be read
S	View field is used to form subsets
H	View field is not transferred to the maintenance screens

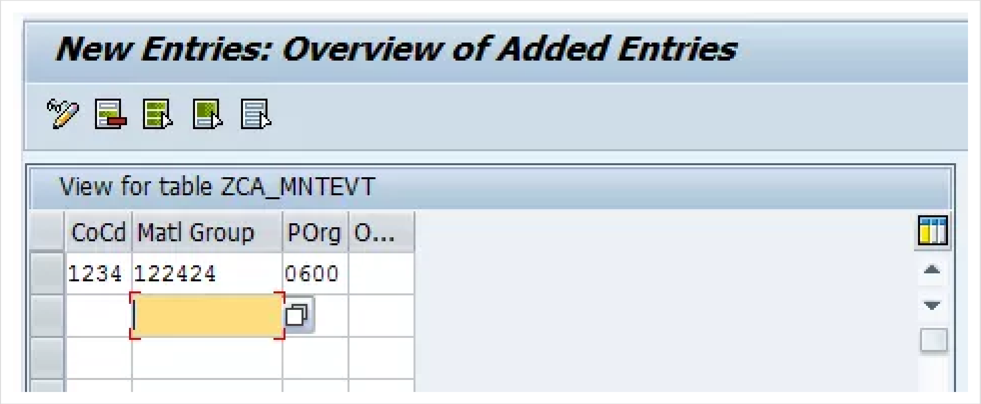
4 Entries found

View field	Table	Field	P	Key	Data elem.	M...	DTyp	Length	Sh
MANDT	ZCA_MNTEVT	MANDT		☑	MANDT	□	CLNT	6	Clie
GUID	ZCA_MNTEVT	GUID	H	☑	GUID	□	RAW	16	Glo
BUKRS	ZCA_MNTEVT	BUKRS		□	BUKRS	□	CHAR	4	Co
MATKL	ZCA_MNTEVT	MATKL		□	MATKL	□	CHAR	9	Ma
EKORG	ZCA_MNTEVT	EKORG		□	EKORG	□	CHAR	4	Pu
BSART	ZCA_MNTEVT	BSART		□	BSART	□	CHAR	4	Or

If you now regenerate the view using Maintenance view generator, you'll achieve the same result as previously with typing the code manually.

In this specific case the second approach (setting the field attribute in the view directly) is definitely better because all manual code you put in the auto-genereated code will be removed when maintenance generator is executed again.

Now try to enter some values and save the changes.

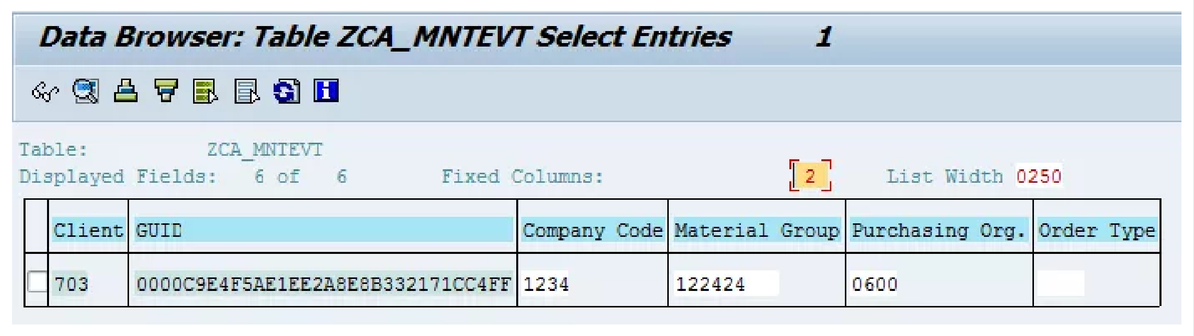


New Entries: Overview of Added Entries

View for table ZCA_MNTEVT

CoCd	Matl Group	POrg	O...
1234	122424	0600	

Go to transaction SE16, open table ZCA_MNTEVT and verify the GUID has been generated for all new rows.



Data Browser: Table ZCA_MNTEVT Select Entries 1

```
Table:          ZCA_MNTEVT
Displayed Fields:    6 of    6         Fixed Columns:          2      List Width 0250
```

	Client	GUID	Company Code	Material Group	Purchasing Org.	Order Type
☐	703	0000C9E4F5AE1EE2A8E8B332171CC4FF	1234	122424	0600	

RELATED

Row level locking of database table
2013-04-04
In "ABAP"

ABAP - Hierarchical View Clusters
2014-10-14
In "ABAP"

ALV tutorial 11 - Edit & Save changes
2013-11-17
In "ABAP"

Article displayed: 7603x
This entry was posted in **ABAP**, **Development** and tagged **ABAP**, **events**, **GUID**, **maintenance**, **maintenance view**, **primary key**, **SAP** by **Spider**. Bookmark the **permalink [https://oprsteny.com/?p=375]** .