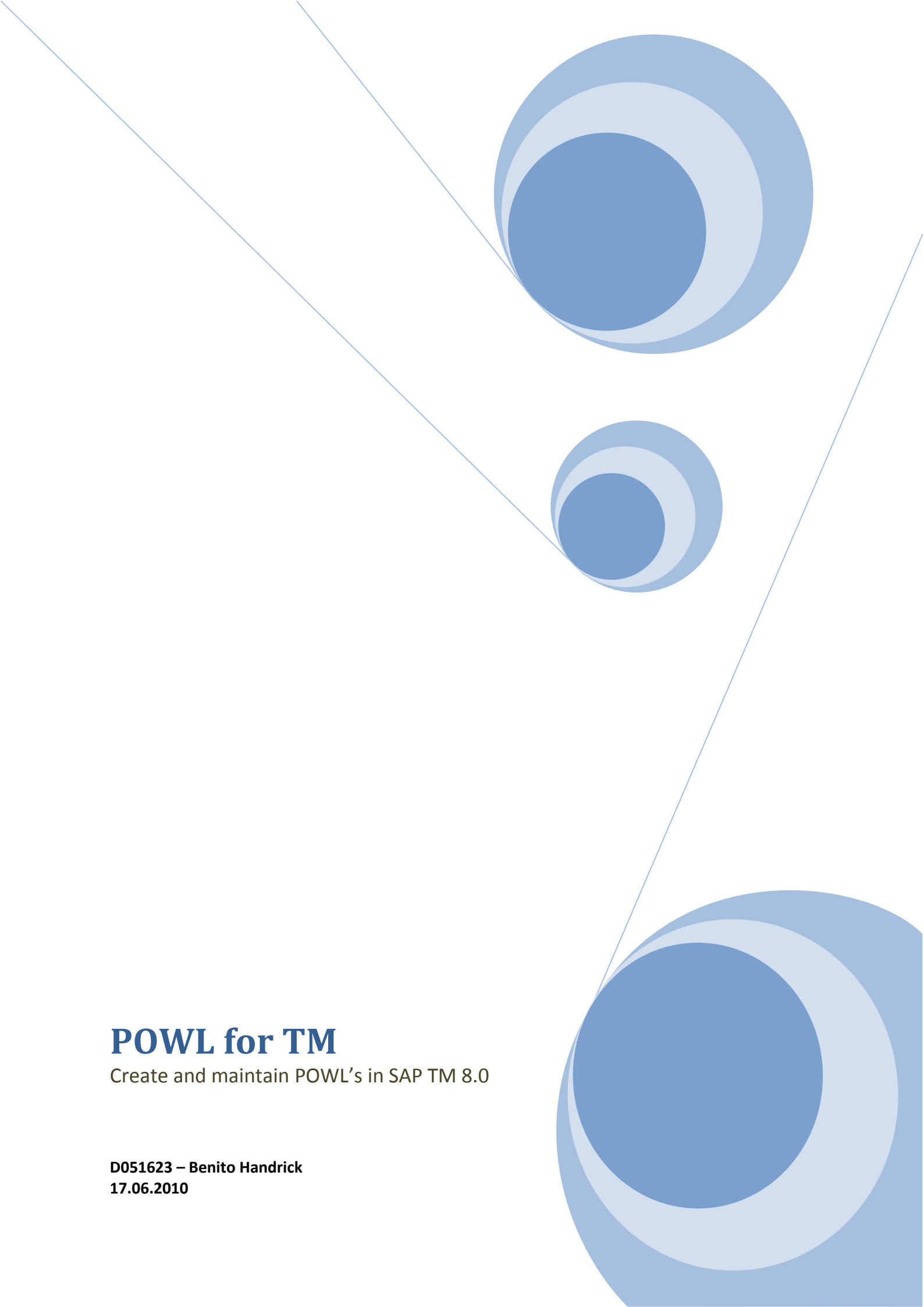


POWL for TM

Create and maintain POWL's in SAP TM 8.0

D051623 – Benito Handrick
17.06.2010

Contents

Introduction

This document describes the creation and the maintenance of POWL for SAP TM 8.0.

After reading this document you are able to define new POWL feeder classes, types and queries. Additionally, you are able to redefine the output formats of this query by redefining the standard methods for the field catalog and the selection criteria. You will also get familiar with the possibility to bring new BO actions in form of buttons and drop down elements in the POWL toolbar.

This document is not an introduction in the POWL technology. It describes only the implementation of the POWL in SAP TM 8.0.

For further details, please take a look into the POWL wiki.

https://wiki.wdf.sap.corp/wiki/display/ERPOPSBNG/POWL+Framework

Creation of a TM specific POWL

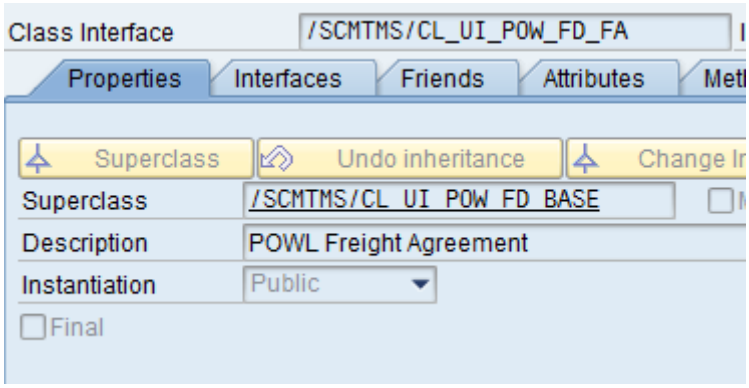
The POWL Feeder Class

The POWL Feeder Class is the main access point for the POWL creation. The feeder class is responsible for:

- field catalog
- selection criteria
- action definition

The 3 methods and their redefinition will be described in the following chapters.

The first step is creating a new class with a “self-speaking” name. The POWL Feeder Class inherits from the superclass „/SCMTMS/CL_UI_POW_FD_BASE“.



In the next step it is necessary that the class constructor is redefined. The following code example shows the redefinition of the Freight Agreement Feeder Class Constructor.

```
METHOD constructor.
super->constructor( ).

ms_pow_profile-bo_name
= /scmtms/if_fag_c=>sc_bo_name.


ms_pow_profile-bo_query_key
= /scmtms/if_fag_c=>sc_query-root-select_by_elements_powl.


ms_pow_profile-bo_altid_attribute_name
= /scmtms/if_fag_c=>sc_node_attribute-root-fagrmntid044.


ms_pow_profile-bo_category_attribute_name
= /scmtms/if_fag_c=>sc_node_attribute-root-FAGUSAGEID105.


ms_pow_profile-pow_output_structure_name
= /scmtms/if_ui_pow_const=>co_output_structure_name-fa.
```

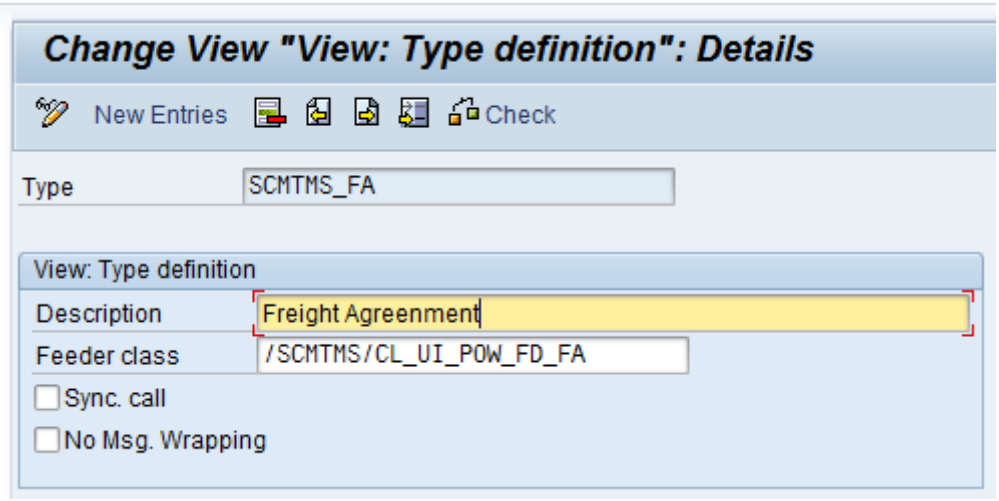
```
ms_pow_profile-pow_selcrit_structure_name
= /scmtms/if_ui_pow_const=>co_selcrit_structure_name-fa.


ms_pow_profile-action_class_name
= /scmtms/if_ui_pow_const=>co_action_class_name-fa.
init( ).
ENDMETHOD.
```

bo_name	The relevant BO name
bo_query_key	BO Query
bo_altid_attribute_name	describes the attribute name of the result structure, which is the alternative identifier of the Business Object. His attribute will be displayed as the first column. Furthermore this column will be displayed as link and also be fixed
pow_output_structure_name	is the output structure for the field catalog (based on the result structure of the output query or the data structure of the root node). All Standard POWL result structure start with **/SCMTMS/S_UI_POW_R***
pow_selcrit_structure_name	is the selection criteria structure for the selection criteria catalog (based on the querystructure of the BO query). All Standard POWL query structures start with **/SCMTMS/S_UI_POW_S***
action_class_name	(optional), is needed if you use your own action class
bo_category_attribute_name	(optional), describes the attribute name of the result structure that contains a BO Category (needed in TRQ; TOR)
bo_key_attribute_name	describes the attribute name of the result structure being responsible for generic navigation, which starts after having clicked on the generated link which is based on bo_altid_attribute_name
conversion_class_name	(optional) is required if you use your own action class

All standard result's and query structures of TM 80 POWLs are centrally available in the POWL constants interface „/SCMTMS/IF_UI_POW_CONS". If you create a new Standard POWL, you need to ensure that you also use it.

POWL_TYPE

Use this transaction to register the POWL Feeder Class and add a description. The description will be used within the POWL dialog for the user specific query definition.

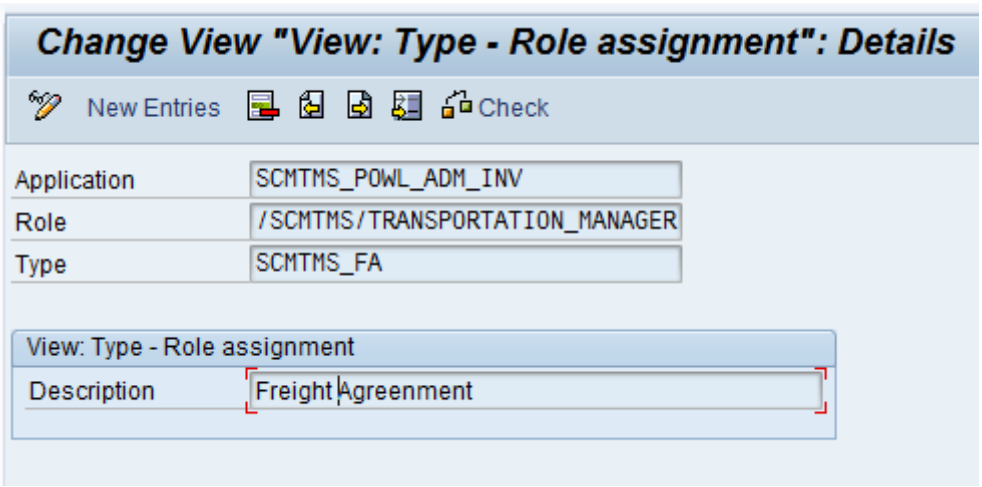


In addition you can set the attribute 'Sync' -> this will force a synchronous refreshing of this type. By default the POWL refreshes asynchronous.

POWL_TYPER and POWL_TYPEU

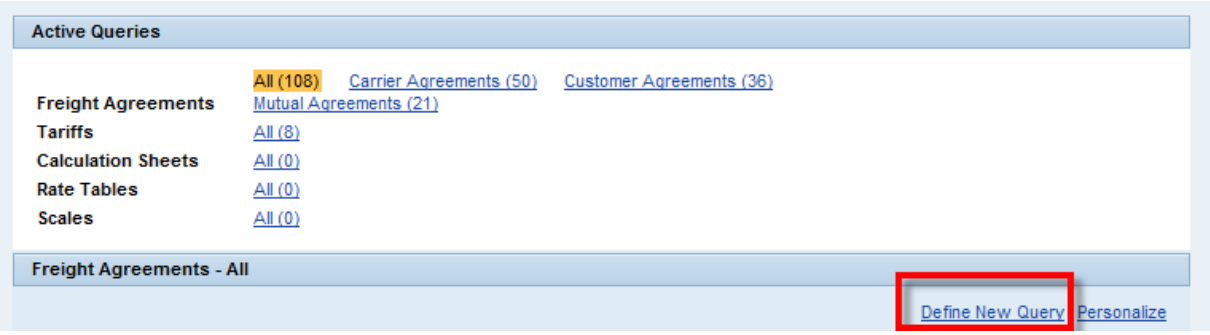
Use transaction:

- POWL_TYPER to assign a POWL type to a PFCG role
- POWL_TYPEU to assign a POWL type to a user

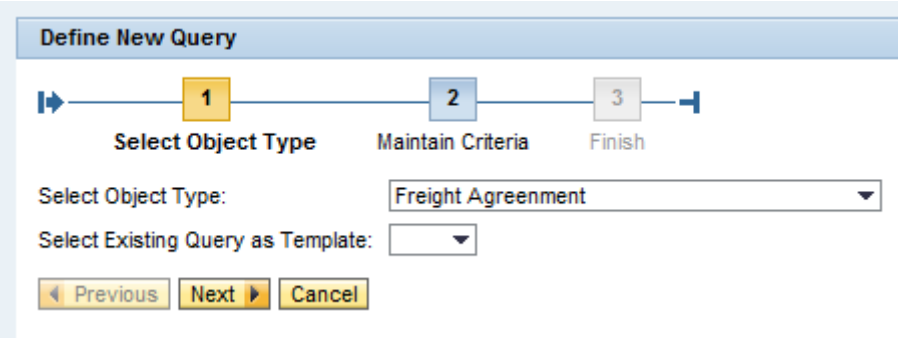


After these two steps you are now able to define a new query in the TM UI.

If you want to define a new query, please go in the POWL UI and press the link "Define New Query"



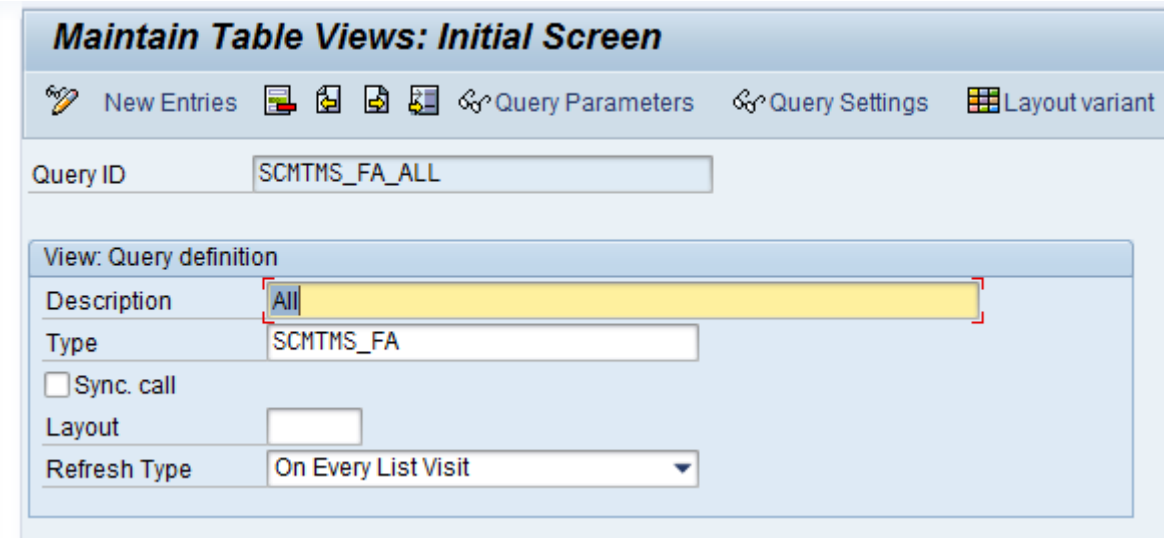
The result is a guided procedure in which you define a complete personalized POWL Query.



The next steps are necessary if you want to create admin work lists which are visible in the standard overviews.

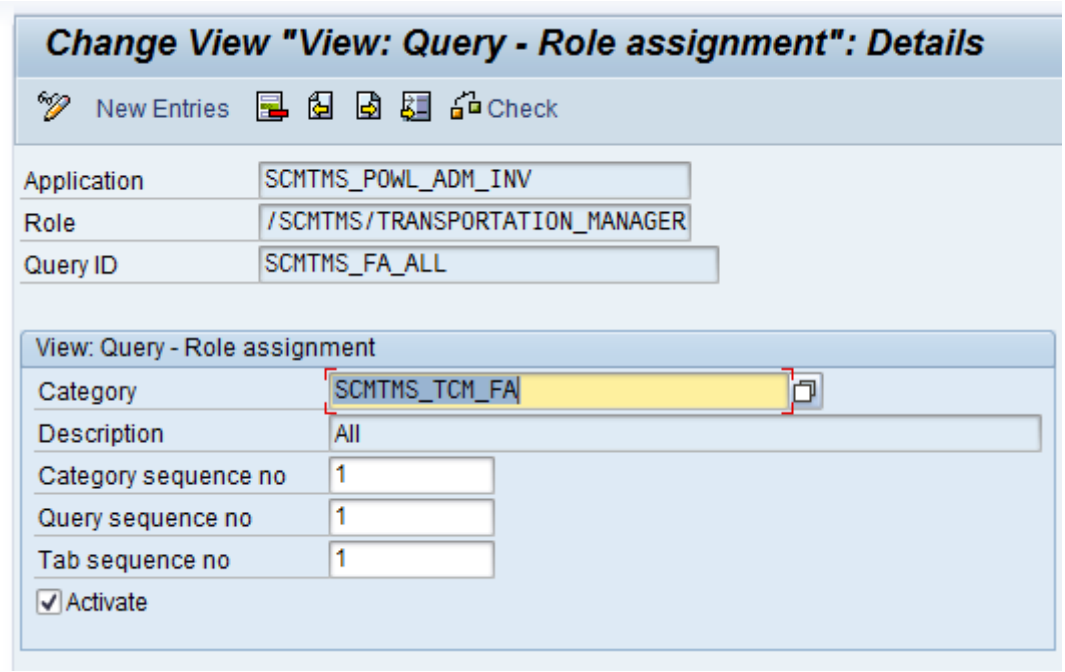
POWL_QUERY

With the POWL_QUERY transaction you can create and define admin work lists. Assign here the created POWL Type to the new Query



POWL_QUERYR and POWL_QUERYU

These transactions are similar to POWL_TYPER and POWL_TYPEU. But here you create an assignment between the Query and the PFCG role or the user.



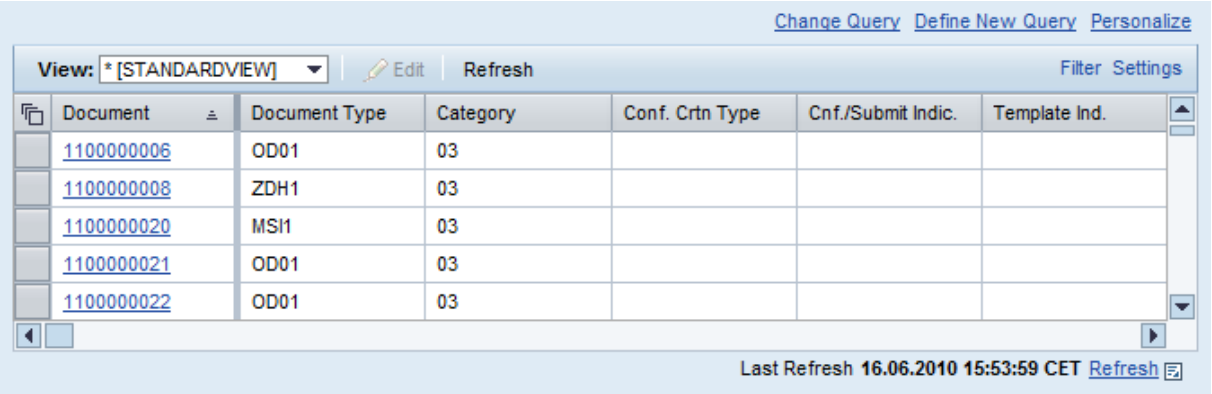
The category sequence number defines in which category you want to display the query. The Query sequence number and the Tab sequence number set the query position within the POWL Category.

Redefinition of Feeder Class Methods

If you want to have some BO specific buttons in the POWL toolbar or would like to fine-tune the layout of the POWL field catalog.

Redefinition GET_FIELD_CATALOG

The field catalog is the collection of the visible POWL columns in the POWL UI.

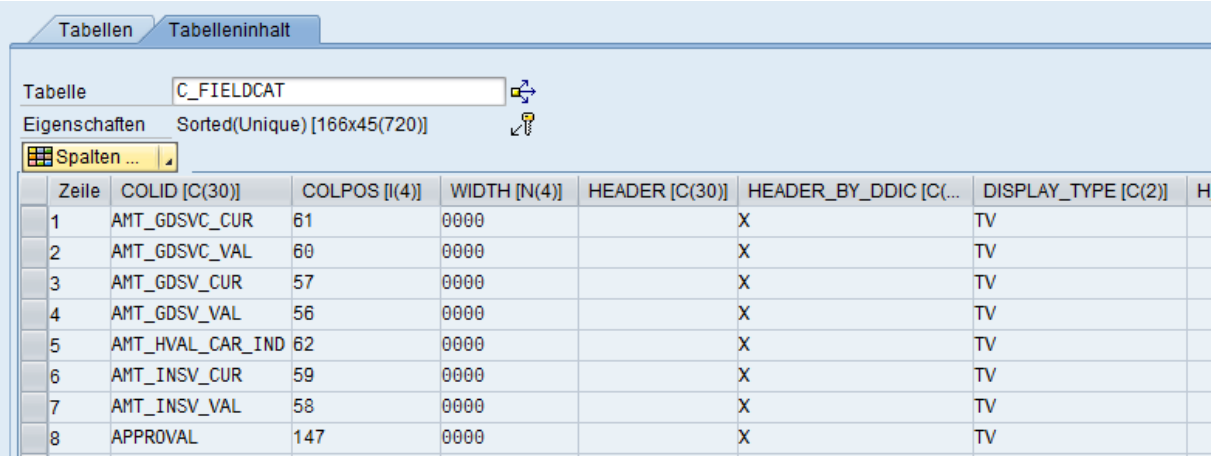


The standard output structure is the structure which is named in the feeder class constructor (*ms_pow_profile-pow_output_structure_name)*

If you want to redefine this output structure, please add the following code in the method:

```
METHOD if_powl_feeder~get_field_catalog.
  super->if_powl_feeder~get_field_catalog(
    EXPORTING
      i_username              = i_username
      i_applid                = i_applid
      i_type                  = i_type
*     i_langu                 = SY-LANGU
      i_selcrit_values        = i_selcrit_values
    importing
      e_fieldcat_changed      = e_fieldcat_changed
*     e_visible_cols_count    =
*     e_visible_rows_count    =
*     e_default_technical_col =
    CHANGING
      c_fieldcat              = c_fieldcat
        ).
```

In the next step you can loop over the table “c_fieldtab” and change every field you want.



Tabellen | Tabelleninhalt

Tabelle C_FIELDCAT

Eigenschaften Sorted(Unique) [166x45(720)]

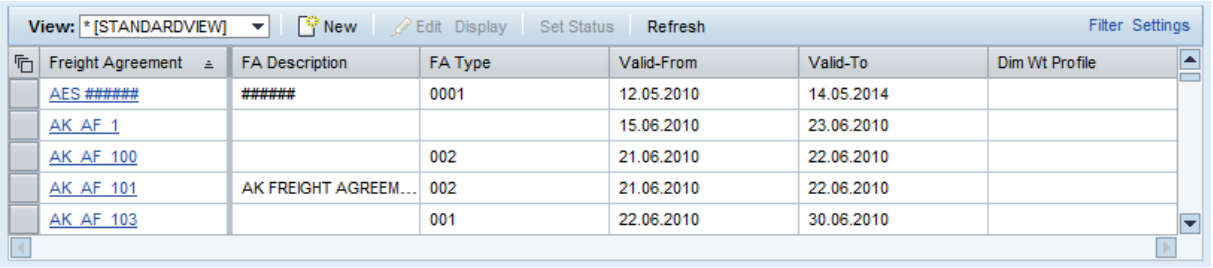
Spalten ...

Zeile	COLID [C(30)]	COLPOS [I(4)]	WIDTH [N(4)]	HEADER [C(30)]	HEADER_BY_DDIC [C(...	DISPLAY_TYPE [C(2)]	H
1	AMT_GDSVC_CUR	61	0000	X		TV	
2	AMT_GDSVC_VAL	60	0000	X		TV	
3	AMT_GDSV_CUR	57	0000	X		TV	
4	AMT_GDSV_VAL	56	0000	X		TV	
5	AMT_HVAL_CAR_IND	62	0000	X		TV	
6	AMT_INSV_CUR	59	0000	X		TV	
7	AMT_INSV_VAL	58	0000	X		TV	
8	APPROVAL	147	0000	X		TV	

You can change different parameters: For example, you can set the position in the output table or change the header and tooltip text. It is also possible to redefine the display type. The following display types are available (check Interface /SCMTMS/IF_UI_POW_CONST=> CO_DISPLAY_TYPE).

- TV - textview
- IM - image
- CK - checkbox
- DK - dropdown by key
- LA - link to action
- IN - input field
- LU - link to url
- BT - button
- PI - progress indicator

For a better understanding we redefine the field catalog of the Freight Agreement.



View: * [STANDARDVIEW] | New | Edit | Display | Set Status | Refresh | Filter Settings

Freight Agreement	FA Description	FA Type	Valid-From	Valid-To	Dim Wt Profile
AES ######	######	0001	12.05.2010	14.05.2014	
AK_AF_1			15.06.2010	23.06.2010	
AK_AF_100		002	21.06.2010	22.06.2010	
AK_AF_101	AK FREIGHT AGREEM...	002	21.06.2010	22.06.2010	
AK_AF_103		001	22.06.2010	30.06.2010	

In the picture you can see the Freight Agreement Query before we changed it.

In our case we want to change the position of the fields “FA Description” and “FA Type”. For our use case, the field “Dim Wt Profile” is obsolete.

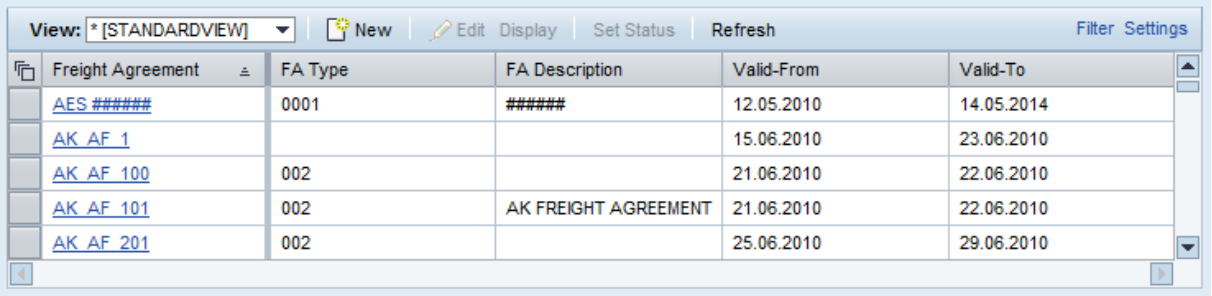
We open the method “GET_FIELD_CATALOG “ in the Feeder Class “/SCMTMS/CL_UI_POW_FD_FA”.

Set the column position of the Freight Agreement Description

```
WHEN 'MNEMONIC'.
  ls_fieldcat-colpos = 3.
```

Delete the Dimension of WT Profile out of the list

```
        when 'DIM_WT_PROFILE'.
          ls_fieldcat-col_visible = abap_false.
```



Now you can see the new POWL after the changes.

Conversion

The conversion is required to convert technical fields for display. Furthermore it is needed to convert selection criterias into internal format for the BO Query.

Conversion of Selection Criteria

There is only one kind of conversion for selection criteria: The date conversion.

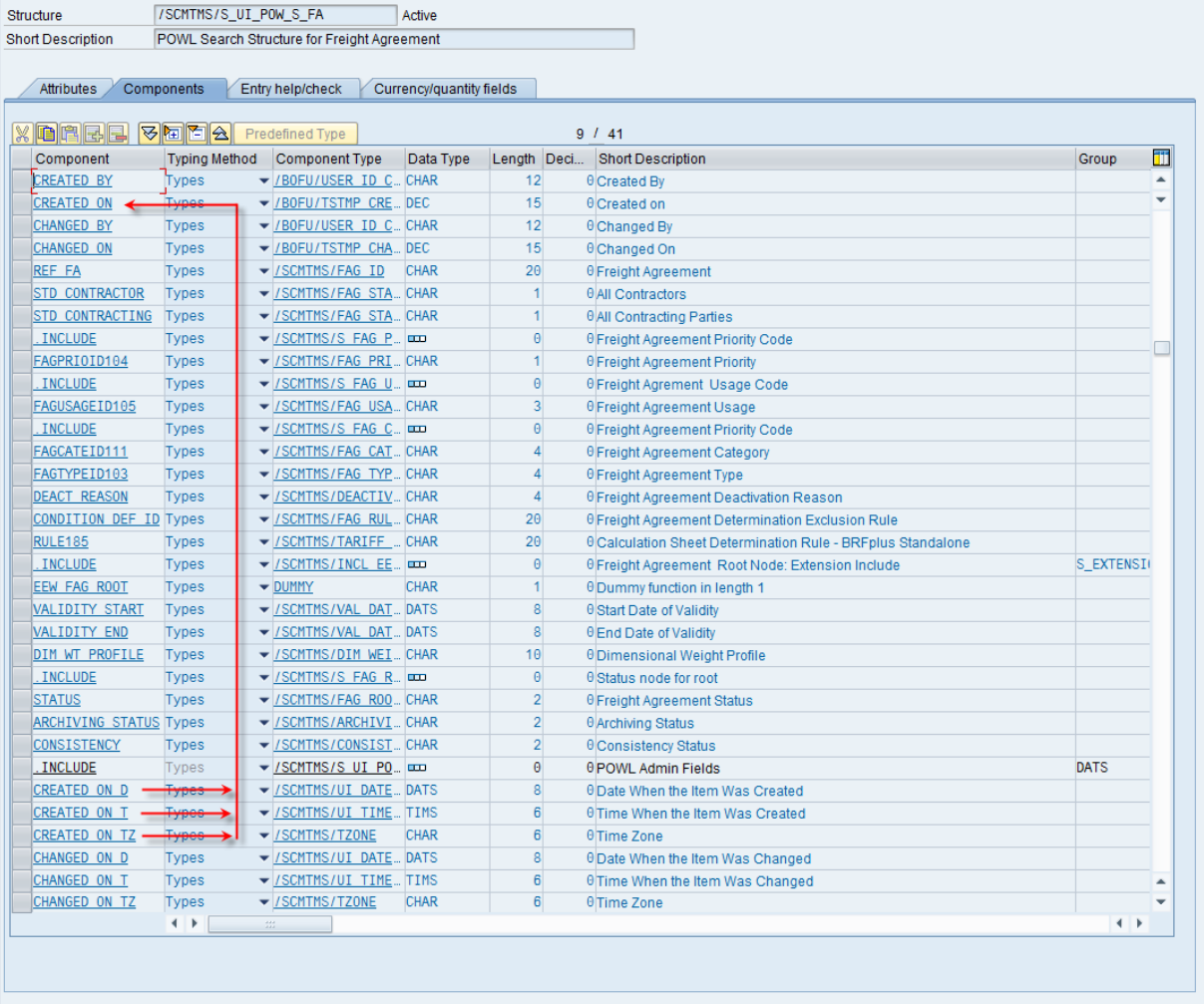
The selection criteria structure of the Freight Agreement consists of following "Includes":

.INCLUDE Types /SCMTMS/S_FAG_ROOT_Q_ELEMENTS

.INCLUDE Types /SCMTMS/S_UI_POW_ADMIN2

The first Include contains the BO query attributes.

The second one is a UI Include which is responsible for the UI fields that have to be converted. This Include has the group name "DATS", which defines the type of conversion (DATS = Date conversion).



The example above shows the conversion of the Creation Date during runtime. The POWL Conversion Method converts the UI attributes CREATED_ON_D + CREATED_ON_T + CREATED_ON_TZ in the second POWL Include structure (group name DATS) to the BO time stamp field CREATED_ON of the first Include structure. It is important that names are equal (CREATED_ON = CREATED_ON***_D/_T/_TZ***).

This example above is a very easy one. If you need more date fields: you have to create (in SAP standard!) a new Includes structure for the date conversion. The name convention is <name of selection criteria structure>_<conversion type>. You can check example for TRQ:

Name of Selection criteria structure is: **/SCMTMS/S_UI_POW_S_TRQ**

Name of Include structure for date conversion: **/SCMTMS/S_UI_POW_S_TRQ_DATS**

Conversion of result list

For the POWL result list two different conversion types exists, the already known DATS conversion and the Code list provider conversion (Group name is here "CODL").

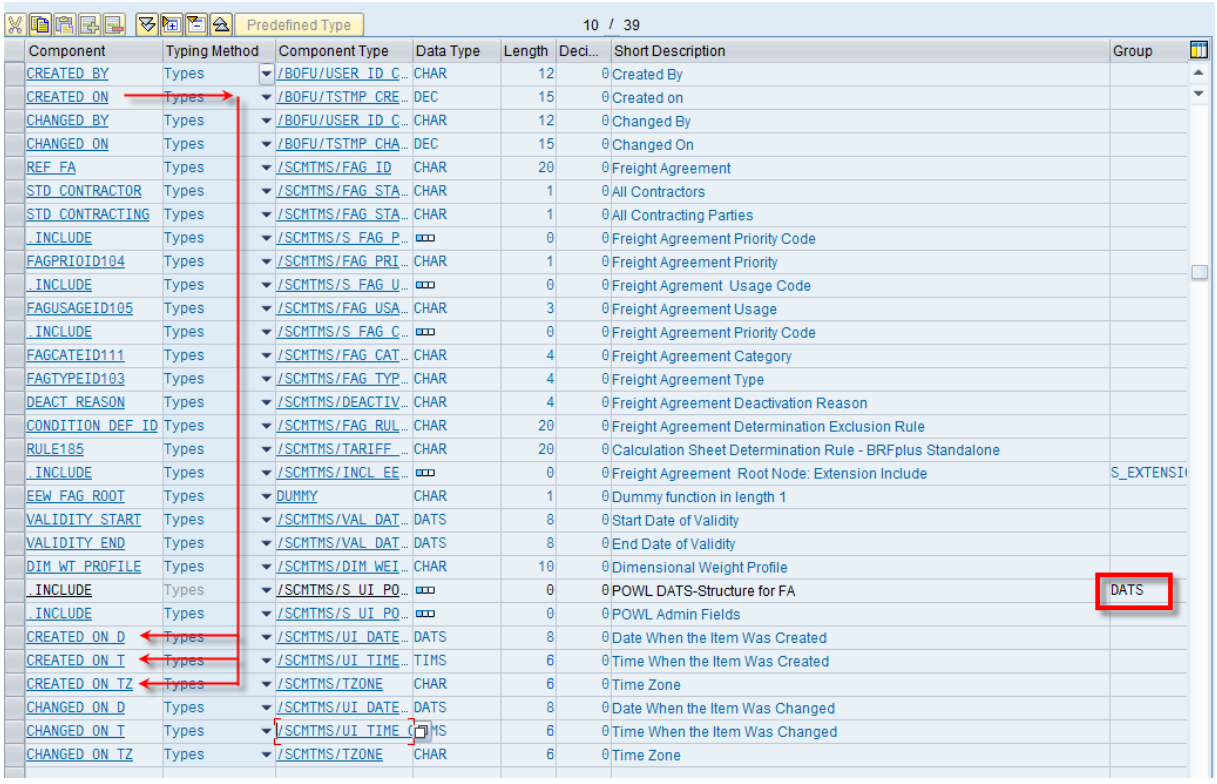
An example which contains both conversions is the TRQ. It contains the following Includes:

.INCLUDE Types /SCMTMS/S_TRQ_Q_RESULT

.INCLUDE Types /SCMTMS/S_UI_POW_R_TRQ_DATS

.INCLUDE Types /SCMTMS/S_UI_POW_R_TRQ_CODL

The first Iinclude contains the BO result list attributes. The second one is a UI Include which is responsible for the UI fields that have to be converted. This Include has the group name "DATS", which defines the type of conversion (DATS = Date conversion). The third one contains the Code list conversion fields with the group name "CODL".



The example above shows the conversion of the Creation Date during runtime. The POWL Conversion Method converts the BO attributes CREATED_ON of the BO Include to the UI attribute

CREATED_ON_D + CREATED_ON_T + CREATED_ON_TZ existing in the second POWL Include structure (group name DATS). It is important that names are equal (CREATED_ON = CREATED_ON***_D/_T/_TZ***).

The CODL conversion converts technical fields to user friendly format (E.G. Life cycle 01 = "New"). To enable the conversion of field BASE_BTD_TCO, an additional attribute BASE_BTD_TCO_TXT has to be added in the CODL Include. During runtime, the conversion class converts the technical value BASE_BTD_TCO into the field BASE_BTD_TCO_TXT. It is important that names are equal (BASE_BTD_TCO = BASE_BTD_TCO***_TXT***).

The Name convention is <name of selection criteria structure>_<conversion type>.

Name result structure is: **/SCMTMS/S_UI_POW_R_TRQ**

Name of Include structure for date conversion: **/SCMTMS/S_UI_POW_R_TRQ_DATS**

Name of Include structure for code list conversion: **/SCMTMS/S_UI_POW_R_TRQ_CODL**

GET_SEL_CRITERIA

With the redefinition of the selection criteria catalog you can give the user a collection of fields with which own queries can be built. These queries are personalized.

The procedure is the same like the redefinition of the field catalog.

Conversion also exists in the Selection Criteria, but it will be only the conversion of dates supported.

GET_ACTIONS

With the get_action method it is possible to bring own action buttons in the POWL toolbar.

If you don't redefine the get_action_method you have the following standard actions:

- New
- Delete
- Edit
- Copy
- Display

Here is a code example which shows the creation of a button.

```
* Create Freight Units
    ls_action_def-placementindx     = lv_index + 2.
    ls_action_def-cardinality       = 'S'.
    ls_action_def-placement         = 'B'.
    ls_action_def-actionid = /scmtms/if_ui_trq_c=>sc_action-create_fu.
    ls_action_def-text = cl_wd_utilities=>get_otr_text_by_alias( 'XY' ).
    ls_action_def-tooltip = cl_wd_utilities=>get_otr_text_by_alias( 'YZ' ).
    ls_action_def-enabled = abap_true.
```

```
insert ls_action_def into table c_action_defs.
```

The placementindex sets the position number of the button in the POWL.

The cardinality has the options ‚S' and ‚I'.

If you set the cardinality S, you must select at least one item to start the action.

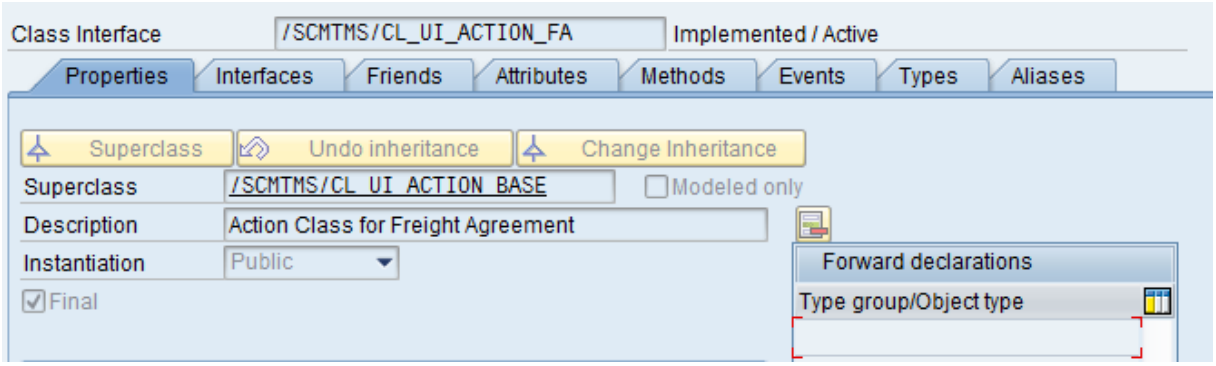
'I' means independent. You can start the action without a selection.

Placement B' sais that the action is represented as a Toolbar Button. 'D' means Drop Down Element.

The Action ID is the BO action which is embedded in the BO constant interface.

Action Handling

If you redefine the get_action method, you must create a action handling method.



This action class must inherit from the superclass "/SCMTMS/CL_UI_ACTION_BASE".

After the creation of the action handling method you must add this method name in the constructor of the special feeder class (ms_pow_profile-action_class_name).

After triggering a self-defined action from the POWL, the "handle_action" method will run and check the action id.

The best way to handle the different action is a CASE statement.

```
METHOD HANDLE_ACTION.
  DATA: lv_key            TYPE /bobf/s_frw_key,
        lv_error          TYPE abap_bool,
        lo_fpm            TYPE REF TO if_fpm,
        lr_event          TYPE REF TO cl_fpm_event.

  IF mo_srvmgr IS BOUND.
    CASE  iv_action_id.
      WHEN /scmtms/if_ui_fa_c=>sc_action-release.
        do_action(
          EXPORTING
            iv_act_key    =  /scmtms/if_fag_c=>sc_action-root-set_released
            iv_save       =  abap_true
            it_key        =  it_keys ).
      WHEN /scmtms/if_ui_fa_c=>sc_action-in_process.
        do_action(
          EXPORTING
            iv_act_key    =  /scmtms/if_fag_c=>sc_action-root-set_in_process
            iv_save       =  abap_true
            it_key        =  it_keys ).
      WHEN /scmtms/if_ui_fa_c=>sc_action-deactivate.
        do_action(
          EXPORTING
            iv_act_key    =  /scmtms/if_fag_c=>sc_action-root-set_deactivated
            iv_save       =  abap_true
            it_key        =  it_keys ).
    ENDCASE.
  ENDIF.
ENDMETHOD.
```

Do something

FAQ

When Do I create a POWL Feeder Class, POWL Type or POWL Query?

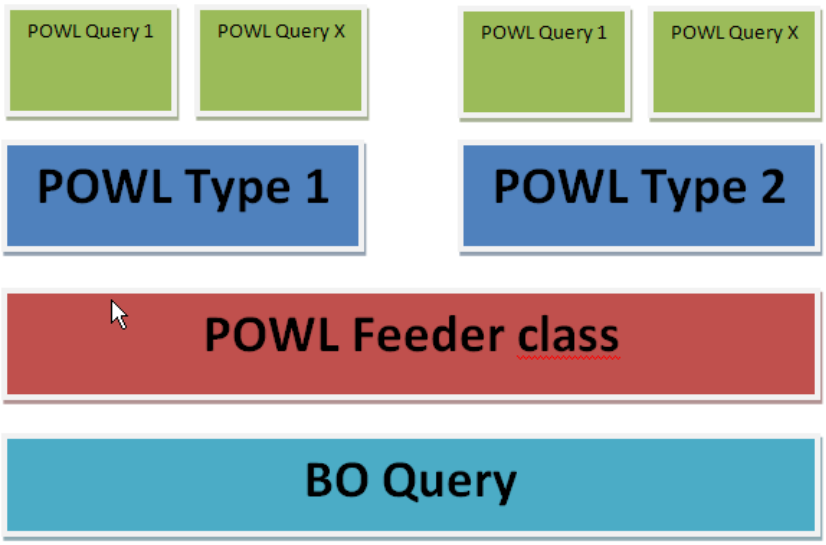


Abbildung 1 Overview POWL Query Objects

To answer this question, let's have a look at the overview. The picture above shows roughly how different objects work together. Base for the POWL Feeder Class is a BO Query. The POWL Feeder class again is base for different POWL Types, which are base for customizable POWL Queries.

POWL Feeder class

In TM, a POWL Feeder Class is based on a query of a Business Object. The relationship is one to one. So whenever there is no POWL Feeder class for an existing BO Query you need a new POWL Feeder class. You should also keep this in mind when you create BO Queries for POWL usage. If you have two complete different requirements which lead to different BO Queries, of course two different POWL-Feeder classes are required. If you have different business categories/usages for POWL which have from technical point of view a lot of common parts, it makes sense, if you design one (technical) BO Query which has one technical POWL Feeder class later. Based on this feeder class, you can separate your different usages/categories with POWL types. (One example is the TRQ Feeder class, with separated POWL types for each BO Category). The next question is now: When do I need a new POWL TYPE? When is a POWL-query enough? To answer this, let's have a closer look at the POWL Type usage in TM.

POWL Type

A POWL Type is always required, if you need adjustment, which requires coding changes or if you would like to enforce separated selection ranges based on coding, which is even not changeable for the POWL admin.

Changes inside coding (or let's say new POWL types are needed):

- If you would like to rename columns header,
- If you would like to rename labels for selection criteria,
- If you would like to change the action toolbar

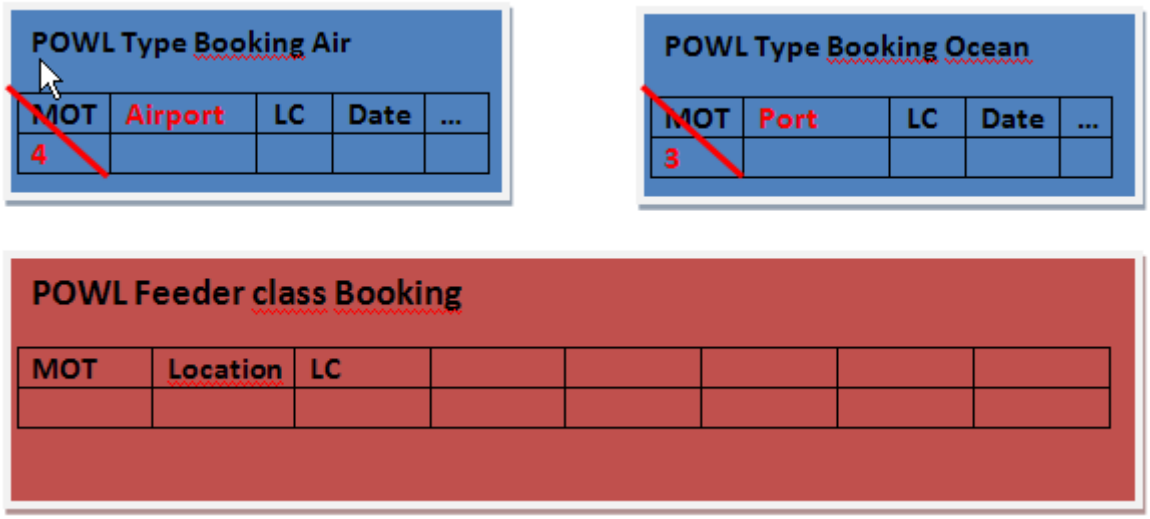


Abbildung 2 POWL Types based on POWL Feeder Class

An example is here the Freight Booking POWLs. Here for each mode of transport exists an extra POWL Type. So it is possible to call one time a technical field "location" one time "harbor" and another time "airport". From technical point of view it is the same BO Query and the same attribute.

An example for separated selection ranges are also the Bookings. Basically the BO Query can select all Bookings independent of the mode of transport. It is possible now to set a default parameter depending on a POWL Type (one time Air Bookings one time Ocean Bookings). This default

parameter is hard coded and later not visible for the POWL admin and user. So it is not possible to create a query based on a POWL Type for Air Booking which selects Ocean Bookings. The POWL Type is the base for different queries and offers a kind of super set with selection attributes and fields for the result list and a fixed defined toolbar.

POWL Query

A POWL Query is customizing and based on a POWL Type. POWL offers the possibility to create based on existing POWL Type different queries. A POWL Query can differ in following things:

- Visibility of Selection Criteria (+Quick Criteria Maintenance),
- Properties (mandatory, read only) of Selection Criteria (+Quick Criteria Maintenance),
- Values of Selection Criteria (+Quick Criteria Maintenance),
- Visible fields in result list,
- Order of result list fields,
- Sort options and all other ALV features,
- Refresh behavior (only manual refresh, every list visit, etc…)

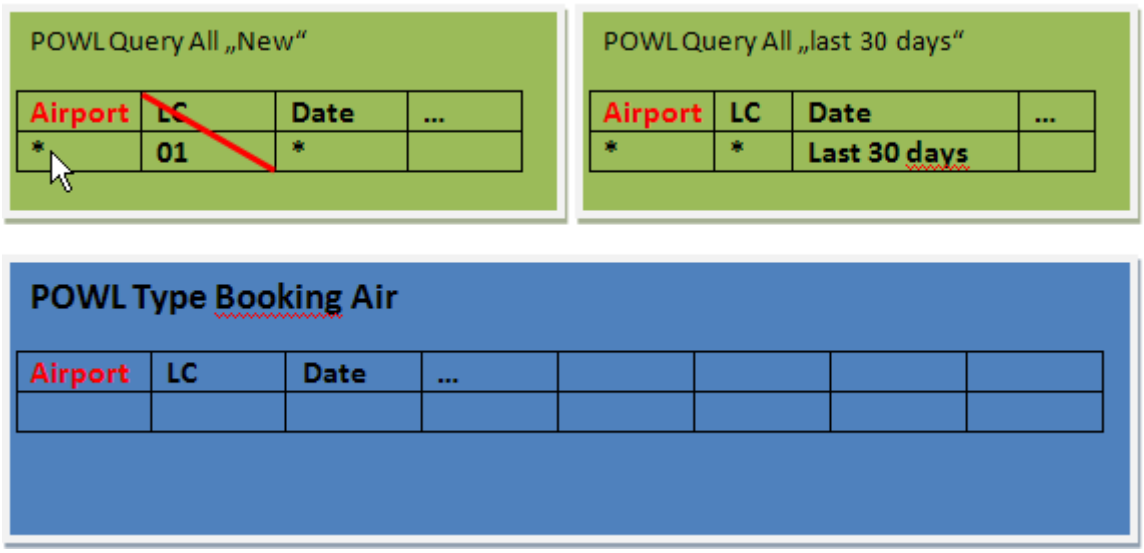


Abbildung 3: POWL Query based on POWL Type

In case of the freight booking POWL a POWL Query might be enough if you would like to see a list of all Air Booking with a Life Cycle Status “New”. Furthermore the life cycle shall be hidden for the customer. Another example for the query would be a list with all Air Bookings which was created in the last 30 days.