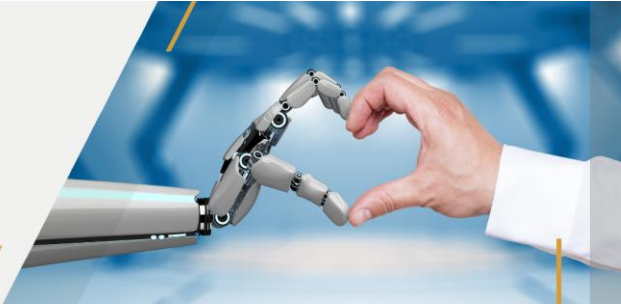




PERFORMER SUITE

Customer Function Modules (BW/4HANA)

SCENARIO VERSION 25.1

SETTINGS VARIANT IT STANDARD DOCUMENTATION

COMMENT VARIANT ---

RESPONSIBLE PERSON ALEXANDER LUDWIG

CREATION DATE 18.08.2025

Scenario: CUSTOMER FUNCTION MODULES V24.2 BW4 - Function Modules for Customers, with Sourcecode

Table of contents

1	Docu Performer General Settings.....	3
2	General Information	3
3	Customer-specific Function Modules	4
4	 Z RFC_READ_REPORT	4
4.1	Comment.....	4
4.1.1	Objective.....	4
4.1.2	Product	4
4.2	Technical Documentation.....	4
4.2.1	General Information	4
4.2.2	Import Parameter	4
4.2.3	Export Parameter.....	4
4.2.4	Tables.....	4
4.2.5	Exceptions	4
4.2.6	Source Code Function module.....	4
5	 Z RFC_GET_DTP_DETAILS.....	5
5.1	Comment.....	5
5.1.1	Objective.....	5
5.1.2	Product	5
5.2	Technical Documentation.....	5
5.2.1	General Information	5
5.2.2	Import Parameter	5
5.2.3	Export Parameter.....	5
5.2.4	Tables.....	5
5.2.5	Exceptions	6
5.2.6	Source Code Function module.....	6
6	 Z RFC_AUTH_CHECK.....	7
6.1	Comment.....	7
6.1.1	Objective.....	7
6.1.2	Product	7
6.2	Technical Documentation.....	7
6.2.1	General Information	7
6.2.2	Import Parameter	8
6.2.3	Tables.....	8
6.2.4	Exceptions	8
6.2.5	Source Code Function module.....	8
7	 Z RFC_USAGE_ANALYSIS.....	9
7.1	Comment.....	9
7.1.1	Objective.....	9
7.1.2	Product	9
7.2	Technical Documentation.....	9
7.2.1	General Information	9
7.2.2	Import Parameter	10
7.2.3	Tables.....	10
7.2.4	Exceptions	10
7.2.5	Source Code Function module.....	10
8	 Z RFC_CODESCAN.....	20
8.1	Comment.....	20
8.1.1	Objective.....	20
8.1.2	Product	21
8.2	Technical Documentation.....	21
8.2.1	General Information	21
8.2.2	Import Parameter	21
8.2.3	Tables.....	21
8.2.4	Exceptions	21
8.2.5	Source Code Function module.....	21
8.2.6	Includes	48
9	 Z RFC_TRANSLATION.....	48
9.1	Comment.....	48
9.1.1	Objective.....	48
9.1.2	Product	48
9.2	Technical Documentation.....	48
9.2.1	General Information	48
9.2.2	Import Parameter	48
9.2.3	Tables.....	49
9.2.4	Exceptions	49
9.2.5	Source Code Function module.....	49

10	F Z RFC_ADSO_GETDTL_XML	50
10.1	Comment	50
10.1.1	Objective	50
10.1.2	Product	50
10.2	Technical Documentation	50
10.2.1	General Information	50
10.2.2	Import Parameter	51
10.2.3	Export Parameter	51
10.2.4	Exceptions	51
10.2.5	Source Code Function module	51
11	F Z RFC_ADSO_CREATE_XML	53
11.1	Comment	53
11.1.1	Objective	53
11.1.2	Product	53
11.2	Technical Documentation	53
11.2.1	General Information	53
11.2.2	Import Parameter	53
11.2.3	Exceptions	54
11.2.4	Source Code Function module	54
12	F Z RFC_FUNCTION_DELETE	56
12.1	Comment	56
12.1.1	Objective	56
12.1.2	Product	56
12.2	Technical Documentation	56
12.2.1	General Information	56
12.2.2	Import Parameter	57
12.2.3	Exceptions	57
12.2.4	Source Code Function module	57
13	F Z RFC_CHECK_ACT_TR	57
13.1	Comment	57
13.1.1	Objective	57
13.1.2	Product	58
13.2	Technical Documentation	58
13.2.1	General Information	58
13.2.2	Import Parameter	58
13.2.3	Export Parameter	58
13.2.4	Exceptions	58
13.2.5	Source Code Function module	58
14	F Z RFC_HCPR_CREATE	64
14.1	Comment	64
14.1.1	Objective	64
14.1.2	Product	64
14.2	Technical Documentation	64
14.2.1	General Information	64
14.2.2	Import Parameter	64
14.2.3	Export Parameter	64
14.2.4	Tables	64
14.2.5	Exceptions	64
14.2.6	Source Code Function module	64

1 Docu Performer General Settings

Performer Suite Settings	
Settings Variant	IT standard documentation
Comment Variant	
Word Template	20220318_EN_Performer_Suite_Scenario_Template.dotx

2 General Information

Created By	Alexej
Created On	07/01/2025 13:29:36
Last Changed By	Admin
Change Date	18/08/2025 14:07:48

Timestamp of documentation	18/08/2025 14:08:21
----------------------------	---------------------

3 Customer-specific Function Modules

4 Z_RFC_READ_REPORT

4.1 Comment

4.1.1 Objective

Read Source code of Includes, Reports and Function Modules.

4.1.2 Product

Docu Performer, System Scout

4.2 Technical Documentation

4.2.1 General Information

System	A4H
Function Module	Z_RFC_READ_REPORT,
Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG
Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:08:23 PM

4.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
PROGRAM	LIKE	SY-REPID			X	

4.2.3 Export Parameter

Parameter		Associated Type	Pass	Short text
SYSTEM	LIKE	SY-SYSID	X	
TRDIR	LIKE	TRDIR	X	

4.2.4 Tables

Parameter		Associated Type	Opt.	Short text
QTAB	LIKE	BAPITGB		

4.2.5 Exceptions

Exception	Short text
RELEASE_1_1	
NOT_AUTHORIZED	

4.2.6 Source Code Function module

FUNCTION Z_RFC_READ_REPORT.

```

*-----
**"Local Interface:
**  IMPORTING
**      VALUE(PROGRAM) LIKE  SY-REPID
**  EXPORTING
**      VALUE(SYSTEM) LIKE  SY-SYSID
**      VALUE(TRDIR) LIKE  TRDIR STRUCTURE  TRDIR
**  TABLES
**      QTAB STRUCTURE  BAPITGB

```

```

""" EXCEPTIONS
"""     RELEASE_1_1
"""     NOT_AUTHORIZED
"""-----

refresh qtab.
read report program into qtab.
select single * from trdir where name = program. "#EC CI_ALL_FIELDS_NEEDED
system = sy-sysid.
*****New ADÜ_20250430
IF sy-subrc <> 0.
"handled later in the Performer Suite
ENDIF.
*****End ADÜ_20250430

endfunction.

```

5 Z RFC_GET_DTP_DETAILS

5.1 Comment

5.1.1 Objective

Read DTP Filter and Semantic Groups.

If the Function Module is not installed, the DTPs are documented without Filters and Semantic Groups.

5.1.2 Product

Docu Performer

5.2 Technical Documentation

5.2.1 General Information

System	A4H
Function Module	Z RFC_GET_DTP_DETAILS,
Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG
Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:08:27 PM

5.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
I_DTP	TYPE	RSBKDTPNM			X	

5.2.3 Export Parameter

Parameter		Associated Type	Pass	Short text
E_MAXSIZE	TYPE	RSBKMAXSIZE	X	
DATA_VAR	TYPE	MCH_T_VAR_SELECT	X	

5.2.4 Tables

Parameter		Associated Type	Opt.	Short text
DATA_SEL	LIKE	RSSELECT		
DATA_RULE	LIKE	MCH_S_SOURCECOD E		
DATA_GROUP	LIKE	RSBK_S_FIELDS_KEY FL		

5.2.5 Exceptions

Exception	Short text
RELEASE_1_5	
NOT_AUTHORIZED	
NOT_GET_FILTER_DETAILS	

5.2.6 Source Code Function module

```

FUNCTION Z_RFC_GET_DTP_DETAILS.
*"-----
**"Local Interface:
**  IMPORTING
**    VALUE(I_DTP) TYPE  RSBKDTPNM
**  EXPORTING
**    VALUE(E_MAXSIZE) TYPE  RSBKMAXSIZE
**    VALUE(DATA_VAR) TYPE  MCH_T_VAR_SELECT
**  TABLES
**    DATA_SEL STRUCTURE  RSSELECT
**    DATA_RULE STRUCTURE  MCH_S_SOURCECODE
**    DATA_GROUP STRUCTURE  RSBK_S_FIELDS_KEYFL
**  EXCEPTIONS
**    RELEASE_1_5
**    NOT_AUTHORIZED
**    NOT_GET_FILTER_DETAILS
**-----
*****End ADÜ_20250430

data l_r_rsbk_dtp type ref to cl_rsbk_dtp.
data l_r_rsbk_filter type ref to cl_rsbk_filter.
data l_r_rsbk_error_handler_tpl
    type ref to cl_rsbk_error_handler_tpl.

data: l_t_groupfields type rsbk_tx_fields_keyfl,
      l_th_groupfields type hashed table of rsbk_sx_fields_keyfl
          with unique key segid.
data ls_group          type rsbk_s_fields_keyfl.
data ls_seltab         type rsbk_s_select.
data ls_ruletab        type mch_s_sourcecode.
data ls_groupfields    type rsbk_sx_fields_keyfl.
data lv_sel            type rsselect.
data lv_rule           type mch_s_sourcecode.
data lv_group          type rsbk_s_fields_keyfl.
data lv_maxsize        type rsbkmaxsize.

* ==== Get Filter details ====
try.
*BEGIN New ADÜ_20250328
    authority-check object 'S_RS_DTP'
        id 'RSTLDTPSRC' field '*'
        id 'RSSTDTPSRC' field '*'
        id 'RSONDTPSRC' field '*'
        id 'RSTLDTPGT' field '*'
        id 'RSSTDTPGT' field '*'
        id 'RSONDTPGT' field '*'
        id 'ACTVT' field '03'.
    if sy-subrc <> 0.
        raise not_authorized.
    else.
*END New ADÜ_20250328
        l_r_rsbk_dtp = cl_rsbk_dtp=>factory( i_dtp ).
        l_r_rsbk_filter =
            l_r_rsbk_dtp->if_rsbk_dtp_display~get_obj_ref_filter( ).
*BEGIN New ADÜ_20250328
    endif.
*END New ADÜ_20250328
    catch cx_rs_access_error.
*****New ADÜ_20250430
        RAISE not_get_filter_details.
*****End ADÜ_20250430
    endtry.

* ==== Get Semantic Groups details ====
    call method l_r_rsbk_dtp->if_rsbk_dtp_display~get_groupfields

```

```

importing
  e_t_groupfields = l_t_groupfields.
insert lines of l_t_groupfields into table l_th_groupfields.

* Fill result table for selections:
loop at l_r_rsbrc_filter->n_t_seltab into ls_seltab.

  move ls_seltab-field to lv_sel-fieldnm.
  move ls_seltab-sign to lv_sel-sign.
  move ls_seltab-option to lv_sel-option.
  move ls_seltab-low to lv_sel-low.
  move ls_seltab-high to lv_sel-high.
  append lv_sel to data_sel.

endloop.

* Fill result table for ABAP-Routines in selections
loop at l_r_rsbrc_filter->n_t_dtprule into ls_ruletab.

  move-corresponding ls_ruletab to lv_rule.
  append lv_rule to data_rule.

endloop.

* Fill result table for Variables in selections
move-corresponding l_r_rsbrc_filter->n_t_varseltab to data_var.

* Fill result table for Semantic Groups
loop at l_th_groupfields into ls_groupfields.
  loop at ls_groupfields-t_fields into ls_group.

    move ls_group-fieldname to lv_group-fieldname.
    move ls_group-txtlg to lv_group-txtlg.
    append lv_group to data_group.

  endloop.
endloop.

* Get Max. Package Size
try.
  call method l_r_rsbk_dtp->get_maxsize
    receiving
      r_maxsize = lv_maxsize.
catch cx_rs_not_authorized.
  message id sy-msgid type sy-msgty number sy-msgno
    with sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
endtry.

e_maxsize = lv_maxsize.

endfunction.

```

6 Z_RFC_AUTH_CHECK

6.1 Comment

6.1.1 Objective

Identify users who are authorized to Queries.

If the Function Module is not installed, the function will work with less performance.

6.1.2 Product

Docu Performer

6.2 Technical Documentation

6.2.1 General Information

System	A4H
Function Module	Z_RFC_AUTH_CHECK,

Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG
Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:08:33 PM

6.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
I_COMPID	TYPE	XUVAL			X	
I_PROVIDER	TYPE	XUVAL			X	
I_INFOAREA	TYPE	XUVAL		X	X	
I_ACTIVITY	TYPE	XUVAL			X	
I_OWNER	TYPE	XUVAL			X	

6.2.3 Tables

Parameter		Associated Type	Opt.	Short text
T_USER	LIKE	BAPITGB		

6.2.4 Exceptions

Exception	Short text
RELEASE_1_2	

6.2.5 Source Code Function module

```

FUNCTION Z_RFC_AUTH_CHECK.
*"-----
***"Local Interface:
*"  IMPORTING
*"    VALUE(I_COMPID) TYPE  XUVAL
*"    VALUE(I_PROVIDER) TYPE  XUVAL
*"    VALUE(I_INFOAREA) TYPE  XUVAL OPTIONAL
*"    VALUE(I_ACTIVITY) TYPE  XUVAL
*"    VALUE(I_OWNER) TYPE  XUVAL
*"  TABLES
*"    T_USER STRUCTURE  BAPITGB
*"  EXCEPTIONS
*"    RELEASE_1_2
*"-----

TYPES: BEGIN OF ty_user,
        bname TYPE xubname,
        name_text TYPE ad_namtext,
      END OF ty_user.

FIELD-SYMBOLS <fs_user> TYPE ty_user.

DATA: lt_user TYPE TABLE OF ty_user,
      chk_comp TYPE c LENGTH 1,
      chk_compl TYPE c LENGTH 1.

REFRESH t_user.

SELECT  bname name_text FROM user_addrp INTO TABLE lt_user."#EC CI_NOWHERE
SORT lt_user.

LOOP AT lt_user ASSIGNING <fs_user>.

  CLEAR: chk_comp, chk_compl.

  CALL FUNCTION 'AUTHORITY_CHECK'
    EXPORTING
      user          = <fs_user>-bname
      object        = 'S_RS_COMP'
      field1        = 'RSZCOMPID'
      value1        = i_compid
      field2        = 'RSZCOMPTP'
      value2        = 'REP'
      field3        = 'RSINFOCUBE'

```

```

        value3          = i_provider
        field4          = 'ACTVT'
        value4          = i_activity
        field5          = 'RSINFOAREA'
        value5          = i_infoarea
    EXCEPTIONS
        user_dont_exist = 1
        user_is_authorized = 2
        user_not_authorized = 3
        user_is_locked = 4
        OTHERS = 5.
    IF sy-subrc = 2.
        chk_comp = 'X'.
    ENDIF.

    CALL FUNCTION 'AUTHORITY_CHECK'
        EXPORTING
            user          = <fs_user>-bname
            object        = 'S_RS_COMP1'
            field1        = 'RSZCOMPID'
            value1        = i_compid
            field2        = 'RSZCOMPTP'
            value2        = 'REP'
            field3        = 'RSZOWNER'
            value3        = i_owner
            field4        = 'ACTVT'
            value4        = i_activity
            field5        = ''
            value5        = ''
        EXCEPTIONS
            user_dont_exist = 1
            user_is_authorized = 2
            user_not_authorized = 3
            user_is_locked = 4
            OTHERS = 5.

    IF sy-subrc = 2.
        chk_compl = 'X'.
    ENDIF.

    IF chk_comp = 'X' AND chk_compl = 'X'.
        APPEND <fs_user> TO t_user.
    ENDIF.

ENDLOOP.

```

ENDFUNCTION.

7 Z RFC_USAGE_ANALYSIS

7.1 Comment

7.1.1 Objective

Mandatory for the Where-Used analysis of InfoObjects and reporting entities.
The Where-Used analysis of InfoProvider works without this Function Module.

7.1.2 Product

System Scout

7.2 Technical Documentation

7.2.1 General Information

System	A4H
Function Module	Z RFC_USAGE_ANALYSIS,
Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG

Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:08:40 PM

7.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
I_VERSION	TYPE	RSOBJVERS	'A'	X	X	
I_IOBJNM	TYPE	RSIOBJNM		X	X	
I_ATTRINM	TYPE	RSATTRINM		X	X	
I_VALUE	TYPE	RSCHAVL		X	X	
I_COMPID	TYPE	RSZCOMPID		X	X	
I_UID	TYPE	SYSUUID_25		X	X	
I_EXTENDED	TYPE	SONV-FLAG		X	X	
I_LEVEL_MAX	TYPE	INT4	99	X	X	

7.2.3 Tables

Parameter		Associated Type	Opt.	Short text
T_CHAVL	LIKE	RSA_S_CHAVL		
T_DATA	LIKE	V_COMPDIR_COMPIC		
T_DATA2	LIKE	TAB512		
T_DATA_RANGE	LIKE	TAB512		

7.2.4 Exceptions

Exception	Short text
RELEASE_2_1	

7.2.5 Source Code Function module

FUNCTION Z RFC_USAGE_ANALYSIS.

```

*-----
**"Local Interface:
**  IMPORTING
**    VALUE(I_VERSION) TYPE  RSOBJVERS DEFAULT 'A'
**    VALUE(I_IOBJNM) TYPE  RSIOBJNM OPTIONAL
**    VALUE(I_ATTRINM) TYPE  RSATTRINM OPTIONAL
**    VALUE(I_VALUE) TYPE  RSCHAVL OPTIONAL
**    VALUE(I_COMPID) TYPE  RSZCOMPID OPTIONAL
**    VALUE(I_UID) TYPE  SYSUUID_25 OPTIONAL
**    VALUE(I_EXTENDED) TYPE  SONV-FLAG OPTIONAL
**    VALUE(I_LEVEL_MAX) TYPE  INT4 DEFAULT 99
**  TABLES
**    T_CHAVL STRUCTURE  RSA_S_CHAVL
**    T_DATA STRUCTURE  V_COMPDIR_COMPIC
**    T_DATA2 STRUCTURE  TAB512
**    T_DATA_RANGE STRUCTURE  TAB512
**  EXCEPTIONS
**    RELEASE_2_1
**-----

```

* A) Elemente in Query-Selektionen -> RSZRANGE

* B) Berechnete Kennzahlen -> RSZCALC

```

*-----

```

TYPES:

```

  BEGIN OF ty_uid_deftp,
    compuid TYPE sysuuid_25,
    deftp   TYPE rszdeftp,
  END OF ty_uid_deftp.

```

DATA:

```

  l_subrc      TYPE rs_bool,
  l_iobjnm     TYPE rsiobjnm,
  l_uid        TYPE sysuuid_25,
  l_iobjtp     TYPE rsiobjtp,
  l_s_uid_deftp TYPE ty_uid_deftp,
  l_t_data     TYPE ty_t_data_scan,

```

```

l_t_data_range TYPE ty_t_data_range,
l_s_data       TYPE v_compdire_compic, "Struktur für Export Parameter
l_s_data2      TYPE char512,          "Struktur für Export Parameter
l_s_data_range TYPE char512.          "Struktur für Export Parameter

FIELD-SYMBOLS:
<s_data>       TYPE ty_s_data_scan,
<s_data_range> TYPE ty_s_data_range,
<fs_chavl>     TYPE rsa_s_chavl.

LOOP AT t_chavl ASSIGNING <fs_chavl>.
  ls_chavl-sign   = 'I'.
  ls_chavl-option = 'EQ'.
  ls_chavl-low    = <fs_chavl>-chavl.
  APPEND ls_chavl TO gt_chavl.
ENDLOOP.

"Globalen Parameter setzen
p_level_max = i_level_max.

"Wiederverwendbare Elemente puffern
SELECT * FROM rszcompdire INTO TABLE g_t_rszcompdire "#EC CI_NOFIRST
  WHERE objvers = 'A'.

IF i_iobjnm IS NOT INITIAL.
  "-----
  " Where-used analysis for InfoObjects
  "-----

  "Check if a display attribute is entered:
  IF i_attrnm IS NOT INITIAL.
    PERFORM scan_attr_dis USING i_iobjnm i_attrnm i_version
      CHANGING l_t_data.
  ELSE.
    SELECT SINGLE iobjtp FROM rsdiobj INTO l_iobjtp "#EC CI_NOORDER
      WHERE iobjnm = i_iobjnm."#EC CI_GENBUFF
    *****New ADÜ_20250430
    IF sy-subrc <> 0.
      "handled later in the Performer Suite
    ENDIF.
    *****End ADÜ_20250430
    IF l_iobjtp = 'KYF'.
      PERFORM scan_kyfnm USING i_iobjnm i_version CHANGING l_t_data.
    ELSEIF i_value IS NOT INITIAL OR gt_chavl IS NOT INITIAL.
      PERFORM scan_value_sel
        USING i_value i_iobjnm i_version gt_chavl
        CHANGING l_t_data l_t_data_range.
    ELSE.
      *      l_seltp = rsd_c_metaobj-charact.
      PERFORM scan_char USING i_iobjnm i_version CHANGING l_t_data.
    ENDIF.
  ENDIF.

ELSEIF i_compid IS NOT INITIAL.
  "-----
  " Analysis for Reusable Query Elements
  "-----

  "Get type and uid from view V_COMPDIR_ELTDIRE
  SELECT SINGLE compid deftp FROM v_compdire_eltdire
    INTO l_s_uid_deftp WHERE compid = i_compid."#EC CI_NOORDER
  IF sy-subrc = 0.
    l_uid = l_s_uid_deftp-compid.
    CLEAR l_iobjnm.
    *      CASE l_deftp.
    CASE l_s_uid_deftp-deftp.
      WHEN 'VAR'.
        "Get iobjnm from view rszglobr
        SELECT SINGLE iobjnm FROM rszglobr "#EC CI_NOORDER "#EC CI_NOFIRST
          INTO l_iobjnm WHERE vnam = i_compid." "#EC CI_NOFIELD
    *****New ADÜ_20250430
    IF sy-subrc <> 0.
      "handled later in the Performer Suite
    ENDIF.
    *****End ADÜ_20250430

  ENDCASE.
  PERFORM scan_compid USING i_compid i_version l_iobjnm l_uid

```

```

                                CHANGING l_t_data.
ENDIF.

ELSEIF i_uid IS NOT INITIAL.
"-----
" Analysis for UID
"-----
IF i_extended IS INITIAL.
    SELECT SINGLE deftp FROM v_compdireltdir "#EC CI_NOORDER
        INTO CORRESPONDING FIELDS OF l_s_uid_deftp
        WHERE compuid = i_uid.
ELSE.
    SELECT SINGLE deftp FROM rszeltdir "#EC CI_NOORDER
        INTO CORRESPONDING FIELDS OF l_s_uid_deftp
        WHERE eltuid = i_uid.
ENDIF.
IF sy-subrc = 0.
    l_uid = i_uid.
    PERFORM scan_compid USING i_compid i_version l_iobjnm l_uid
        CHANGING l_t_data.
ENDIF.

ENDIF.

"-----
" Exportparameter füllen
"-----
"Tabelle T_DATA + T_DATA2
LOOP AT l_t_data ASSIGNING <s_data>.
    AT FIRST.
        "Header Zeile für Exportparameter T_DATA2 erzeugen
        l_s_data2 = 'COMPUID;LEVEL;DEFT;COMPID;COMPUID_PARENT'.
        APPEND l_s_data2 TO t_data2[].
    ENDAT.

    "Parameter T_DATA
    MOVE-CORRESPONDING <s_data> TO l_s_data.
    APPEND l_s_data TO t_data[].

    "Parameter T_DATA2
    CLEAR l_s_data2.
    CONCATENATE
        <s_data>-compuid
        <s_data>-level
        <s_data>-deftp
        <s_data>-compid
        <s_data>-compuid_parent
        INTO l_s_data2 SEPARATED BY ';'.
    APPEND l_s_data2 TO t_data2[].
ENDLOOP.

"Doppelte Einträge löschen
SORT t_data[] BY compuid.
DELETE ADJACENT DUPLICATES FROM t_data[].

"Tabelle T_DATA_RANGE
LOOP AT l_t_data_range ASSIGNING <s_data_range>.
    AT FIRST. "#EC CI_SORTED
        "Header Zeile für Exportparameter T_DATA_RANGE erzeugen
        l_s_data_range =
            'COMPUID;IOBJNM;SIGN;OPT;LOW;LOWFLAG;HIGH;HIGHFLAG;HIENM'.
        APPEND l_s_data_range TO t_data_range[].
    ENDAT.

    READ TABLE l_t_data ASSIGNING <s_data>
        WITH KEY compuid_parent = <s_data_range>-compuid.
    IF sy-subrc = 0.
        <s_data_range>-compuid = <s_data>-compuid.
    ENDIF.

    CLEAR l_s_data_range.
    CONCATENATE
        <s_data_range>-compuid
        <s_data_range>-iobjnm
        <s_data_range>-sign
        <s_data_range>-opt
        <s_data_range>-low
        <s_data_range>-lowflag

```

```

        <s_data_range>-high
        <s_data_range>-highflag
        <s_data_range>-hienm
    INTO l_s_data_range SEPARATED BY ';'.
    APPEND l_s_data_range TO t_data_range[].
ENDLOOP.

ENDFUNCTION.

*&-----*
*&      Form  scan_attr_dis
*&-----*
FORM scan_attr_dis
    USING
        i_iobjnm
        i_attrinm
        i_version
    CHANGING
        c_t_data.

    DATA:
        l_t_eltuid  TYPE ty_t_eltuid,
        l_eltuid    TYPE sysuuid_25,
        l_s_data    TYPE ty_s_data_scan.

    FIELD-SYMBOLS:
        <s_data>    TYPE rsz_x_eltdir,
        <s_eltuid>   TYPE ty_s_eltuid.

    FREE l_t_eltuid.

    SELECT DISTINCT eltuid FROM rszeltattr INTO TABLE l_t_eltuid  "#EC CI_NOFIRST
    WHERE objvers = i_version
    AND iobjnm = i_iobjnm
    AND attrinm = i_attrinm.

    "Elemente anfügen und an den Elementen hochhangeln
    LOOP AT l_t_eltuid ASSIGNING <s_eltuid>.
        l_eltuid = <s_eltuid>-eltuid.

        CLEAR l_s_data.
        l_s_data-compuid_parent = l_eltuid.

        PERFORM scan_query_elem_parent
            USING
                i_iobjnm
                l_eltuid
                i_version
            CHANGING
                c_t_data
                l_s_data.
    ENDLOOP.

ENDFORM.          "scan_attr_dis

*&-----*
*&      Form  scan_kyfnm
*&-----*
FORM scan_kyfnm
    USING
        i_iobjnm
        i_version
    CHANGING
        c_t_data.

    DATA:
        l_t_eltuid  TYPE ty_t_eltuid,
        l_eltuid    TYPE sysuuid_25,
        l_s_data    TYPE ty_s_data_scan.

    FIELD-SYMBOLS:
        <s_data>    TYPE rsz_x_eltdir,
        <s_eltuid>   TYPE ty_s_eltuid.

    "-----
    " A] Kennzahl in Tabelle RSZRANGE suchen...

```

```

"      (Eingeschränkte Kennzahl oder direkt in Query)
"-----
FREE l_t_eltuid.

SELECT DISTINCT eltuid FROM rszrange INTO TABLE l_t_eltuid "#EC CI_NOFIRST
WHERE objvers = i_version
  AND iobjnm   = rsd_c_metaioobj-keyfigure
  AND seltp    = rzd1_c_seltp-keyfig
  AND low      = i_iobjnm.

"Elemente anfügen und an den Elementen hochhangeln
LOOP AT l_t_eltuid ASSIGNING <s_eltuid>.
  l_eltuid = <s_eltuid>-eltuid.

  CLEAR l_s_data.
  l_s_data-compuid_parent = l_eltuid.

  PERFORM scan_query_elem_parent
    USING
      i_iobjnm
      l_eltuid
      i_version
    CHANGING
      c_t_data
      l_s_data.
ENDLOOP.

"-----
" B] Kennzahl in Tabelle RSZCALC suchen...
"      (Formel und berechnete Kennzahl)
"-----
FREE l_t_eltuid.
SELECT DISTINCT eltuid FROM rszcalc INTO TABLE l_t_eltuid "#EC CI_NOFIRST
WHERE ( objvers = i_version
      )
  AND ( oper1   = i_iobjnm OR oper2 = i_iobjnm ).

"Elemente anfügen und an den Elementen hochhangeln
LOOP AT l_t_eltuid ASSIGNING <s_eltuid>.
  l_eltuid = <s_eltuid>-eltuid.

  CLEAR l_s_data.
  l_s_data-compuid_parent = l_eltuid.

  PERFORM scan_query_elem_parent
    USING
      i_iobjnm
      l_eltuid
      i_version
    CHANGING
      c_t_data
      l_s_data.
ENDLOOP.

ENDFORM.                                "scan_kyfnm

*&-----*
*&      Form   scan_value_sel
*&-----*
*      Search for characteristics with special selection
*-----*
FORM scan_value_sel
  USING
    i_value
    i_iobjnm
    i_version
    lt_chavl LIKE gt_chavl
  CHANGING
    c_t_data
    c_t_data_range TYPE ty_t_data_range.

DATA:
  l_value      TYPE rschavl,
  l_eltuid     TYPE sysuuid_25,
  l_s_data     TYPE ty_s_data_scan,
  l_t_data_range TYPE ty_t_data_range.

FIELD-SYMBOLS:

```

```

<s_data_range> TYPE ty_s_data_range.

FREE l_t_data_range.

IF i_value CS '*' OR i_value CS '%'.
  l_value = i_value.
  REPLACE ALL OCCURRENCES OF '*' IN l_value WITH '%'.
  SELECT DISTINCT "#EC CI_NOFIRST
    eltuid
    iobjnm
    sign
    opt
    low
    lowflag
    high
    highflag
    hienm
  FROM rszrange INTO CORRESPONDING FIELDS OF TABLE l_t_data_range
  WHERE objvers = i_version
    AND iobjnm LIKE i_iobjnm "1_8; support Nav-Attributes
    AND ( ( opt = 'EQ' AND low LIKE l_value AND lowflag = 1 )
      OR ( opt = 'BT' AND low <= l_value AND lowflag = 1
        AND high >= l_value AND highflag = 1 ) ).
ELSEIF i_value IS NOT INITIAL.
  SELECT DISTINCT "#EC CI_NOFIRST
    eltuid
    iobjnm
    sign
    opt
    low
    lowflag
    high
    highflag
    hienm
  FROM rszrange INTO CORRESPONDING FIELDS OF TABLE l_t_data_range
  WHERE objvers = i_version
    AND iobjnm LIKE i_iobjnm "1_8; support Nav-Attributes
    AND ( ( opt = 'EQ' AND low = i_value )
      OR ( opt = 'BT' AND low <= i_value AND lowflag = 1
        AND high >= i_value AND highflag = 1 ) ).
ELSEIF lt_chavl IS NOT INITIAL. "Several values are passed (in table)
  SELECT DISTINCT "#EC CI_NOFIRST
    eltuid
    iobjnm
    sign
    opt
    low
    lowflag
    high
    highflag
    hienm
  FROM rszrange INTO CORRESPONDING FIELDS OF TABLE l_t_data_range
  WHERE objvers = i_version
    AND iobjnm LIKE i_iobjnm "1_8; support Nav-Attributes
    AND low IN lt_chavl.
ENDIF.

"Elemente anfügen und an den Elementen hochhangeln
LOOP AT l_t_data_range ASSIGNING <s_data_range>.
  l_eltuid = <s_data_range>-eltuid.

  CLEAR l_s_data.
  l_s_data-compuid = l_eltuid.
  l_s_data-compuid_parent = l_eltuid.

  <s_data_range>-compuid = l_eltuid.
  APPEND <s_data_range> TO c_t_data_range.

  PERFORM scan_query_elem_parent
    USING
      i_iobjnm
      l_eltuid
      i_version
    CHANGING
      c_t_data
      l_s_data.
ENDLOOP.

```

```

ENDFORM.                                "scan_value_sel

*&-----*
*&      Form  scan_char
*&-----*
*-----*
* A] Merkmale in Queries                -> RSZELTDIR
* B] Merkmale in Queries                -> RSZSELECT
* C] Merkmale in Ausnahmeaggregation -> RSZCALC
* D] Merkmale in Variablen             -> RSZGLOBV
* E] Merkmale als Attribut             -> RSZELTATTR
*-----*

FORM scan_char
  USING
    i_iobjnm
    i_version
  CHANGING
    c_t_data.

  DATA:
    l_t_eltuid      TYPE ty_t_eltuid,
    l_eltuid        TYPE sysuuid_25,
    l_s_rszeltxref  TYPE ty_s_rszeltxref,
    l_s_rszelttxt   TYPE rszelttxt,
    l_subrc         TYPE rs_bool,
    l_srch_attr     TYPE string,
    l_s_data        TYPE ty_s_data_scan.

  FIELD-SYMBOLS:
    <s_data>        TYPE ty_s_data_scan,
    <s_eltuid>       TYPE ty_s_eltuid,
    <s_rszcompdir>   TYPE rszcompdir.

  CONCATENATE '%__' i_iobjnm INTO l_srch_attr.

  "-----*
  " A] RSZELTDIR
  "-----*

  FREE l_t_eltuid.
  SELECT DISTINCT eltuid FROM rszeltdir
    INTO CORRESPONDING FIELDS OF TABLE l_t_eltuid
    WHERE ( objvers      = i_version
           AND ( defaulthint = i_iobjnm OR defaulthint LIKE l_srch_attr ) ).

  "Elemente anfügen und an den Elementen hochhangeln
  LOOP AT l_t_eltuid ASSIGNING <s_eltuid>.
    l_eltuid = <s_eltuid>-eltuid.

    CLEAR l_s_data.
    l_s_data-compuid_parent = l_eltuid.

    PERFORM scan_query_elem_parent
      USING
        i_iobjnm
        l_eltuid
        i_version
      CHANGING
        c_t_data
        l_s_data.
  ENDLOOP.

  "-----*
  " B] RSZSELECT
  "-----*

  FREE l_t_eltuid.
  SELECT DISTINCT eltuid FROM rszselect  "#EC CI_NOFIRST
    INTO CORRESPONDING FIELDS OF TABLE l_t_eltuid
    WHERE objvers = i_version
      AND ( ( iobjnm = i_iobjnm OR coniobjnm = i_iobjnm )
          OR ( iobjnm LIKE l_srch_attr OR coniobjnm LIKE l_srch_attr ) ).

  "Elemente anfügen und an den Elementen hochhangeln
  LOOP AT l_t_eltuid ASSIGNING <s_eltuid>.
    l_eltuid = <s_eltuid>-eltuid.

    CLEAR l_s_data.
    l_s_data-compuid_parent = l_eltuid.

```

```

PERFORM scan_query_elem_parent
  USING
    i_iobjnm
    l_eltuid
    i_version
  CHANGING
    c_t_data
    l_s_data.
ENDLOOP.

"-----
" C] RSZCALC
"-----

FREE l_t_eltuid.
SELECT DISTINCT eltuid FROM rszcalc INTO TABLE l_t_eltuid "#EC CI_NOFIRST
  WHERE objvers = i_version
    AND ( aggrcha = i_iobjnm
      OR aggrcha LIKE l_srch_attr ).

"Elemente anfügen und an den Elementen hochhangeln
LOOP AT l_t_eltuid ASSIGNING <s_eltuid>.
  l_eltuid = <s_eltuid>-eltuid.

  CLEAR l_s_data.
  l_s_data-compuid_parent = l_eltuid.

  PERFORM scan_query_elem_parent
    USING
      i_iobjnm
      l_eltuid
      i_version
    CHANGING
      c_t_data
      l_s_data.
ENDLOOP.

"-----
" D] RSZGLOBV
"-----

FREE l_t_eltuid.
SELECT DISTINCT varuniid FROM rszglobov INTO TABLE l_t_eltuid
  WHERE objvers = i_version
    AND ( iobjnm = i_iobjnm
      OR iobjnm LIKE l_srch_attr ).

"Elemente anfügen und an den Elementen hochhangeln
LOOP AT l_t_eltuid ASSIGNING <s_eltuid>.
  l_eltuid = <s_eltuid>-eltuid.

  CLEAR l_s_data.
  l_s_data-compuid_parent = l_eltuid.

  PERFORM scan_query_elem_parent
    USING
      i_iobjnm
      l_eltuid
      i_version
    CHANGING
      c_t_data
      l_s_data.
ENDLOOP.

"-----
" E] RSZELTATTR
"-----

"In case of 7.x Queries: Attributes are yet detected by selection
"on table RSZELTDIR, but in case of 3.x Queries they have to be
"detected by reading table RSZELTATTR
FREE l_t_eltuid.
SELECT DISTINCT eltuid FROM rszeltattr INTO TABLE l_t_eltuid "#EC CI_NOFIRST
  WHERE objvers = i_version
    AND attrnm = i_iobjnm.

"Elemente anfügen und an den Elementen hochhangeln
LOOP AT l_t_eltuid ASSIGNING <s_eltuid>.
  l_eltuid = <s_eltuid>-eltuid.

  CLEAR l_s_data.

```

```

        l_s_data-compuid_parent = l_eltuid.

    PERFORM scan_query_elem_parent
        USING
            i_iobjnm
            l_eltuid
            i_version
        CHANGING
            c_t_data
            l_s_data.
    ENDLOOP.

ENDFORM.                                "scan_char

*&-----*
*&      Form  scan_compid
*&-----*
FORM scan_compid
    USING
        i_compid
        i_version
        i_iobjnm
        i_uid
    CHANGING
        c_t_data.

DATA:
    l_eltuid          TYPE sysuuid_25,
    l_s_rszeltxref    TYPE ty_s_rszeltxref,
    l_t_rszeltxref    TYPE TABLE OF ty_s_rszeltxref,
    l_s_data          TYPE ty_s_data_scan.

SELECT DISTINCT seltuid teltuid laytp FROM rszeltxref
    INTO TABLE l_t_rszeltxref
    WHERE objvers = i_version
        AND teltuid = i_uid.

"Elemente anfügen und an den Elementen hochhangeln
LOOP AT l_t_rszeltxref INTO l_s_rszeltxref.
    l_eltuid = l_s_rszeltxref-seltuid.

    CLEAR l_s_data.
    l_s_data-compuid_parent = l_eltuid.

    PERFORM scan_query_elem_parent
        USING
            i_iobjnm
            l_eltuid
            i_version
        CHANGING
            c_t_data
            l_s_data.

    ENDLOOP.

ENDFORM.                                "scan_compid

*&-----*
*&      Form  SCAN_QUERY_ELEM_PARENT
*&-----*
FORM scan_query_elem_parent
    USING
        i_iobjnm
        i_teltuid
        i_version
    CHANGING
        c_t_data          TYPE ty_t_data_scan
        c_s_data_parent   TYPE ty_s_data_scan.

DATA:
    l_s_rszeltxref    TYPE ty_s_rszeltxref,
    l_t_rszeltxref    TYPE ty_t_rszeltxref,
    l_level           TYPE c LENGTH 4,
    l_compid          TYPE sysuuid_25.

"Handelt es sich um ein wiederverw. Query Element? => hinzufügen

```

```

PERFORM add_record
  USING
    i_iobjnm
    i_teltuid
    i_version
  CHANGING
    c_t_data
    c_s_data_parent.

l_level    = c_s_data_parent-level.
l_compuuid = c_s_data_parent-compuuid.

"Übergeordnete Elemente ermitteln (SELTUID)
SELECT seltuid teltuid laytp FROM rszeltxref
  INTO TABLE l_t_rszeltxref
  WHERE objvers = 'A'
    AND teltuid = i_teltuid.

LOOP AT l_t_rszeltxref INTO l_s_rszeltxref.

  c_s_data_parent-level    = l_level.
  c_s_data_parent-compuuid = l_compuuid.

  "Rekursion beenden?
  CHECK l_level < p_level_max.

  "Eine Stufe hoch (die oberste Ebene entspricht der Query)
  PERFORM scan_query_elem_parent
    USING
      i_iobjnm
      l_s_rszeltxref-seltuid
      i_version
    CHANGING
      c_t_data
      c_s_data_parent.

ENDLOOP.

ENDFORM.                                "SCAN_QUERY_ELEM_PARENT

*&-----*
*&      Form  add_record
*&-----*
FORM add_record
  USING
    i_iobjnm      TYPE rsiobjnm
    i_eltuid      TYPE sysuuid_25
    i_version     TYPE rsobjvers
  CHANGING
    c_t_data      TYPE ty_t_data_scan
    c_s_data_parent TYPE ty_s_data_scan.

DATA:
  l_s_data      TYPE ty_s_data_scan,
  l_s_rszglobv  TYPE rszglobv,
  l_level       TYPE c LENGTH 4.

FIELD-SYMBOLS:
  <s_rszcompdir> TYPE rszcompdir.

"-----"
" Wiederverwendbares Element?
"-----"
READ TABLE g_t_rszcompdir ASSIGNING <s_rszcompdir>
  WITH TABLE KEY compuuid = i_eltuid.
IF sy-subrc = 0.

  CLEAR l_s_data.
  MOVE <s_rszcompdir>-compuuid TO l_s_data-compuuid.
  MOVE <s_rszcompdir>-compid  TO l_s_data-compid.
  MOVE <s_rszcompdir>-version TO l_s_data-version.
  MOVE <s_rszcompdir>-tstpnm  TO l_s_data-tstpnm.
  MOVE <s_rszcompdir>-timestamp TO l_s_data-timestamp.

  l_level = c_s_data_parent-level + '0001'.
  MOVE l_level TO l_s_data-level.
  MOVE c_s_data_parent-compuuid TO l_s_data-compuuid_parent.

```

```

"Datensatz schon vorhanden?
READ TABLE c_t_data TRANSPORTING NO FIELDS
  WITH KEY
    compuid      = <s_rszcompdir>-compuid
    compuid_parent = l_s_data-compuid_parent.

IF sy-subrc NE 0.

*   IF l_s_data-compuid NE l_s_data-compuid_parent.

      IF l_s_data-compuid EQ l_s_data-compuid_parent.
        CLEAR l_s_data-compuid_parent.
      ENDIF.

      "Zusätzl. Infos aus RSZCOMPIC
      SELECT SINGLE infocube FROM rszcompic INTO l_s_data-infocube
        WHERE compuid = i_eltuid
          AND objvers = i_version. "#EC CI_NOORDER

      "Zusätzl. Infos lesen aus RSZELTDIR
      SELECT SINGLE deftp FROM rszeltldir INTO (l_s_data-deftp)
        WHERE eltuid = i_eltuid
          AND objvers = i_version.

      "Zusätzl. Infos aus V_CMP_JOIN
      SELECT SINGLE deftp FROM v_cmp_join INTO (l_s_data-deftp)
        WHERE compuid = i_eltuid
          AND objvers = i_version. "#EC CI_NOORDER

      "Text lesen aus RSZELTTXT
      SELECT SINGLE txtlg FROM rszelttxt INTO l_s_data-txtlg
        WHERE eltuid = l_s_data-compuid
          AND objvers = i_version
          AND langu = sy-langu.

      INSERT l_s_data INTO TABLE c_t_data.

*   ENDIF.

      "Letzten Satz für Level und Vorgänger merken
      c_s_data_parent = l_s_data.

    ENDIF.

  RETURN.

ENDIF.

*   "-----
*   " Variable?
*   "-----
  SELECT SINGLE * FROM rszglobv INTO l_s_rszglobv
    WHERE varuniid = i_eltuid
      AND objvers = 'A'. "#EC CI_ALL_FIELDS_NEEDED

  IF sy-subrc = 0.
    l_s_data-compuid = l_s_rszglobv-varuniid.
    l_s_data-compid = l_s_rszglobv-vnam.
    INSERT l_s_data INTO TABLE c_t_data.
  ENDIF.

ENDFORM.          " ADD_RECORD

```

8 Z_RFC_CODESCAN

8.1 Comment

8.1.1 Objective

Mandatory to scan the ABAP source code (coding in Transformations, Includes, Methods, DTP Routines, etc.)

8.1.2 Product

System Scout

8.2 Technical Documentation

8.2.1 General Information

System	A4H
Function Module	Z_RFC_CODESCAN,
Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG
Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:09:05 PM

8.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
I_CASE	TYPE	RS_BOOL	"	X	X	
I_SKIP_COMMENT	TYPE	RS_BOOL	'X'	X	X	
I_REGEX	TYPE	RS_BOOL	"	X	X	
I_UPDRU	TYPE	RS_BOOL	'X'	X	X	
I_TRANR	TYPE	RS_BOOL	'X'	X	X	
I_TRANS	TYPE	RS_BOOL	'X'	X	X	
I_IOBJROUT	TYPE	RS_BOOL	'X'	X	X	
I_IP	TYPE	RS_BOOL		X	X	
I_DTP	TYPE	RS_BOOL		X	X	
I_REPS	TYPE	RS_BOOL		X	X	
I_CLAS	TYPE	RS_BOOL		X	X	
I_FUNC	TYPE	RS_BOOL		X	X	
I_PLSE	TYPE	RS_BOOL		X	X	

8.2.3 Tables

Parameter		Associated Type	Opt.	Short text
T_STRING_RANGE	LIKE	RSRANGE		
T_DEVCL_RANGE	LIKE	RSRANGE		
T_REPS_RANGE	LIKE	RSRANGE		
T_SUBC_RANGE	LIKE	RSRANGE		
T_CLAS_RANGE	LIKE	RSRANGE		
T_FUGR_RANGE	LIKE	RSRANGE		
DATA	LIKE	TAB512		

8.2.4 Exceptions

Exception	Short text
RELEASE_2_3	
INPUT_CANNOT_BE_TREATED	
NOT_AUTHORIZED	

8.2.5 Source Code Function module

FUNCTION Z_RFC_CODESCAN.

```

*-----
*""Local Interface:
*  IMPORTING
*    VALUE(I_CASE) TYPE RS_BOOL DEFAULT ''
*    VALUE(I_SKIP_COMMENT) TYPE RS_BOOL DEFAULT 'X'
*    VALUE(I_REGEX) TYPE RS_BOOL DEFAULT ''
*    VALUE(I_UPDRU) TYPE RS_BOOL DEFAULT 'X'

```

```

*"      VALUE(I_TRANR) TYPE  RS_BOOL DEFAULT 'X'
*"      VALUE(I_TRANS) TYPE  RS_BOOL DEFAULT 'X'
*"      VALUE(I_IOBJROUT) TYPE  RS_BOOL DEFAULT 'X'
*"      VALUE(I_IP) TYPE  RS_BOOL OPTIONAL
*"      VALUE(I_DTP) TYPE  RS_BOOL OPTIONAL
*"      VALUE(I_REPS) TYPE  RS_BOOL OPTIONAL
*"      VALUE(I_CLAS) TYPE  RS_BOOL OPTIONAL
*"      VALUE(I_FUNC) TYPE  RS_BOOL OPTIONAL
*"      VALUE(I_PLSE) TYPE  RS_BOOL OPTIONAL
*"  TABLES
*"      T_STRING_RANGE STRUCTURE  RSRANGE
*"      T_DEVCL_RANGE STRUCTURE  RSRANGE
*"      T_REPS_RANGE STRUCTURE  RSRANGE
*"      T_SUBC_RANGE STRUCTURE  RSRANGE
*"      T_CLAS_RANGE STRUCTURE  RSRANGE
*"      T_FUGR_RANGE STRUCTURE  RSRANGE
*"      DATA STRUCTURE  TAB512
*"  EXCEPTIONS
*"      RELEASE_2_3
*"      INPUT_CANNOT_BE_TREATED
*"      NOT_AUTHORIZED
*"-----
data:
  ls_data          like line of data,
  ls_result        type s_result,
  ls_line          type c length 255,
  lv_namespacegen  type namespace,
  lv_name_w_o_prefix type rs_char30,
  lt_string        type t_string,
  ls_string        type s_string.

field-symbols:
  <string> type s_string,
  <s_range> type rsrange.

*-----

clear:
  gt_rsaabap,
  gt_object,
  gt_result.

loop at t_string_range assigning <s_range>.

  if ( <s_range>-high = 'ODSO' or <s_range>-high = 'CUBE'
    or <s_range>-high = 'CHAR' or <s_range>-high = 'IOBJ')
    or <s_range>-high = 'ADSO'. "ADÜ_20189424 - DP-96
    call function 'RSD_NAMESPACE_PAR_GET_FROM_NAME'
      exporting
        i_objnm          = <s_range>-low
      importing
*        E_NAMESPACE      =
*        e_namespacegen   = lv_namespacegen
*        E_SYSTP          =
*        e_name_w_o_prefix = lv_name_w_o_prefix
*        E_SOBJNM_W_O_PREFIX =
      exceptions
        name_error       = 1
        others            = 2.

    if sy-subrc = 0.
      "regular expressions: the . is a placeholder for a single sign
      if <s_range>-high = 'ODSO'.
        "sign after prefix is A
        concatenate lv_namespacegen 'A' lv_name_w_o_prefix '.0'
          into ls_string.
        "Start Change ADÜ_20189424 - DP-96
      elseif <s_range>-high = 'ADSO'.
        "sign after prefix is A
        concatenate lv_namespacegen 'A' lv_name_w_o_prefix '.'
          into ls_string.
        "End Change ADÜ_20189424 - DP-96
      elseif <s_range>-high = 'CUBE'.
        "sign after prefix can be D,E or F
        concatenate lv_namespacegen '.' lv_name_w_o_prefix
          into ls_string.
      elseif ( <s_range>-high = 'CHAR' or <s_range>-high = 'IOBJ' ).
        concatenate lv_namespacegen '.' lv_name_w_o_prefix

```

```

        into ls_string.

        "Start Change ADÜ_20180426 - DP-1945
data: ls_string2 type s_string,
      ls_string3 type s_string,
      ls_fieldnm type rsdiobjfieldnm.

select single fieldnm
  from rsdiobj
  into ls_fieldnm
  where iobjnm = <s_range>-low
  and objvers = 'A'.
*****New ADÜ_20250430
  IF sy-subrc <> 0.
    "handled later in the Performer Suite
  ENDIF.
*****End ADÜ_20250430

  if ls_fieldnm is not initial.

    concatenate '-' ls_fieldnm '\' into ls_string2.
    append ls_string2 to lt_string.
    concatenate '-' ls_fieldnm '\s' into ls_string3.
    append ls_string3 to lt_string.

  endif.
  "End Change ADÜ_20180426 - DP-1945

  endif.
endif.
else.
  ls_string = <s_range>-low.
endif.

append ls_string to lt_string.

endloop.

if i_updru = 'X' or i_tranr = 'X' or i_trans = 'X'
  or i_iobjrout = 'X'.
  perform scan_rsaabap
    tables lt_string
    using i_skip_comment i_case i_regex i_updru
        i_tranr i_trans i_iobjrout.
endif.

if i_dtp = 'X'.
*BEGIN New ADÜ_20250328
  authority-check object 'S_RS_DTP'
    id 'RSTLDTPSRC' field '*'
    id 'RSSTDTPSRC' field '*'
    id 'RSONDTPSRC' field '*'
    id 'RSTLDTPGT' field '*'
    id 'RSSTDTPGT' field '*'
    id 'RSONDTPGT' field '*'
    id 'ACTVT' field '03'.
  if sy-subrc <> 0.
    raise not_authorized.
  else.
*END New ADÜ_20250328
    perform scan_dtp
      tables lt_string t_devcl_range
      using i_skip_comment i_case i_regex.
*BEGIN New ADÜ_20250328
  endif.
*END New ADÜ_20250328
endif.

if i_ip = 'X'.
  perform scan_ip
    tables lt_string t_devcl_range
    using i_skip_comment i_case i_regex.
endif.

if i_reps = 'X'.
  perform get_reps
    tables t_devcl_range t_reps_range t_subc_range.

```

```

endif.

if i_clas = 'X'.
    perform get_clas
    tables t_devcl_range t_clas_range.
endif.

if i_func = 'X'.
    perform get_func
    tables t_devcl_range t_fugr_range.
endif.

if i_plse = 'X'.
    perform get_plse
    tables lt_string t_devcl_range
    using i_skip_comment i_case i_regex.
endif.

if not gt_object is initial.
    perform scan_include_code
    tables lt_string
    using i_skip_comment i_case i_regex.
endif.

"Fill result table
loop at gt_result into ls_result.
    if i_skip_comment = 'X'.
        if ls_result-line(1) = '*'.
            continue.
        else.
            ls_line = ls_result-line.
            shift ls_line left deleting leading space.
            "Start change ADü_20180618 - DP-1210
            "AMDP Code
            if ls_result-routinetype = 'RT20'
                or ls_result-routinetype = 'RT21'
                or ls_result-routinetype = 'RT22'
                or ls_result-routinetype = 'RT23'
                or ls_result-routinetype = 'RT24'
                or ls_result-routinetype = 'RT30'
                or ls_result-routinetype = 'RT31'
                or ls_result-routinetype = 'RT32'
                or ls_result-routinetype = 'RT33'.
                if ls_line(2) = '--'.
                    continue.
                endif.
            "ABAP Code
        else.
            "End change ADü_20180618 - DP-1210
            if ls_line(1) = '"'.
                continue.
            endif.
            "Start change ADü_20180618 - DP-1210
        endif.
        "End change ADü_20180618 - DP-1210
    endif.
    ls_data = ls_result.
    append ls_data to data.

endloop.

endfunction.

*&-----*
*&      Form  scan_rsaabap
*&-----*
*      text
*-----*
form scan_rsaabap
    tables lt_string
    using i_skip_comment i_case i_regex i_updru i_tranr
        i_trans i_iobjrout.

types:
    begin of s_lookup,
```

```

    targetname      type rstran-targetname,
    targettype      type rstran-targettype,
    targetsubtype   type rstran-targetsubtype,
    sourcename      type rstran-sourcename,
    sourcetype      type rstran-sourcetype,
    sourcesubtype   type rstran-sourcesubtype,
    codeid          type rsaabap-codeid,
    tranid          type rstran-tranid,
    set_runtime     type char1,
    iobjnm          type rsupdrout-iciobjnm,
    routinetype     type c length 4,
    routine         type n length 4,
end of s_lookup,
t_lookup type standard table of s_lookup.

data:
  l_tabix          like sy-tabix,
  lt_rsaabap       type t_rsaabap,
  ls_rsaabap       type s_rsaabap,
  ls_match_result  type match_result,
  lt_lookup_finder type t_lookup,
  ls_lookup_temp   type s_lookup,
  lt_lookup_temp   type t_lookup,
  ls_lookup        type s_lookup,
  l_updtype        like rsupddat-updtype,
  ls_result        type s_result,
  lv_txtlg         type rstxtlg.

field-symbols:
  <string> type s_string,
  <s_range> type rsrange.

clear gt_rsaabap.
* Restrict on version M, because RSAABAP contains deleted coding
* that exists only in version A but not in version M!
select a~codeid codetp line_no line
  from          rsaabap as a
        inner join rsarout as r
on a~codeid = r~codeid and a~objvers = r~objvers
into corresponding fields of table gt_rsaabap
where a~objvers = 'A'
and a~codeid in (
  select codeid from rsarout
  where objvers = 'M').

"#EC *

"Start Change ADü_20180618 - DP-1210
data: lv_rstranscript      type string value 'RSTRANSCRIPT',
      lv_rstranstepscript type string value 'RSTRANSTEPSCRIPT',
      lv_rstransteprout   type string value 'RSTRANSTEPROUT',
      lv_bwrelease        type saprelease,
      lv_bw4hana          type rs_bool value ''.

types: begin of lty_sscript,
      tranid type rstranid,
      script type string,
      ruleid type rstran_ruleid,
      stepid type rstran_stepid,
      codeid type rscodeid,
      line_no type rsaabap-line_no,
end of lty_sscript,

begin of lty_script,
  tranid type rstranid,
  procnm type char30,
  codeid type rscodeid,
  script type string,
  line_no type rsaabap-line_no,
end of lty_script,

begin of lty_rstransteprout,
  tranid type rstranid,
  ruleid type rstran_ruleid,
  stepid type rstran_stepid,
  codeid type rscodeid,
  on_hana type rs_bool,
  line_no type rsaabap-line_no,
  kind type char7,

```

```

        code      type string,
        code_inv type string,
    end of lty_rstransteprou.

data: ls_script      type lty_script,
      ls_g_script    type lty_script,
      ls_sscript     type lty_sscript,
      ls_g_sscript   type lty_sscript,
      lt_script      type table of lty_script,
      lt_g_script    type table of lty_script,
      lt_sscript     type table of lty_sscript,
      lt_g_sscript   type table of lty_sscript,
      ls_rstransteprou type lty_rstransteprou,
      ls_g_rstransteprou type lty_rstransteprou,
      lt_rstransteprou type table of lty_rstransteprou,
      lt_g_rstransteprou type table of lty_rstransteprou,
      lt_cvers type table of cvers.

*****New ALU 20250701
SELECT component release
  FROM cvers
  INTO TABLE lt_cvers.
* Versuch SAP_BW zu finden
CLEAR lv_bwrelease.
LOOP AT lt_cvers INTO DATA(ls_cvers).
  IF ls_cvers-component = 'SAP_BW'.
    lv_bwrelease = ls_cvers-release.
    EXIT. "#EC CI_NOORDER
  ENDIF.
ENDLOOP.

* Wenn nichts gefunden → BW/4HANA prüfen
IF lv_bwrelease IS INITIAL.
  lv_bw4hana = rs_c_true.

  LOOP AT lt_cvers INTO ls_cvers.
    IF ls_cvers-component = 'DW4CORE'.
      lv_bwrelease = ls_cvers-release.
      EXIT. "#EC CI_NOORDER
    ENDIF.
  ENDLOOP.
ENDIF.
*****End ALU 20250701 Onapsis Change

if lv_bw4hana = rs_c_false.
  if lv_bwrelease < 740. "#EC NUMBER_OK
    "nothing
  elseif lv_bwrelease >= 740 and lv_bwrelease < 750. "#EC NUMBER_OK

    select tranid procnm script
      from (lv_rstranscript)
      into corresponding fields of table lt_script
      where objvers = 'A'.

  elseif lv_bwrelease >= 750. "#EC NUMBER_OK

    select tranid procnm script
      from (lv_rstranscript)
      into corresponding fields of table lt_script
      where objvers = 'A'.

    select tranid ruleid stepid codeid script
      from (lv_rstranstepscrip)
      into corresponding fields of table lt_sscript
      where objvers = 'A'.

  endif.
elseif lv_bw4hana = rs_c_true.
  select tranid ruleid stepid codeid code code_inv
    from (lv_rstransteprou)
    into corresponding fields of table lt_rstransteprou
    where objvers = 'A'.
endif.
"End Change ADü_20180618 - DP-1210

"Start Change ADü_20180426 - DP-1945
data: ls_complstring type string.
loop at lt_string assigning <string>.
  if sy-tabix = 1.

```

```

        ls_complstring = <string>.
    else.
        concatenate ls_complstring '|' <string> into ls_complstring.
    endif.
endloop.
"End Change ADü_20180426 - DP-1945

sort gt_rsaabap by codeid.
loop at gt_rsaabap into ls_rsaabap.
    l_tabix = sy-tabix.
    "LOOP AT lt_string ASSIGNING <string>. "ADü_20180426 - DP-1945
    clear ls_match_result.
    if i_case = ' ' and i_regex = 'X'.
        find first occurrence of regex ls_complstring "<string> "ADü_20180426 - DP-1945
        in ls_rsaabap-line
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = 'X'.
        find first occurrence of regex ls_complstring "<string> "ADü_20180426 - DP-1945
        in ls_rsaabap-line
        in character mode
        respecting case
        results ls_match_result.
    elseif i_case = ' ' and i_regex = ''.
        find first occurrence of ls_complstring "<string> "ADü_20180426 - DP-1945
        in ls_rsaabap-line
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = ''.
        find first occurrence of ls_complstring "<string> "ADü_20180426 - DP-1945
        in ls_rsaabap-line
        in character mode
        respecting case
        results ls_match_result.
    endif.
    if sy-subrc <> 0.
        delete gt_rsaabap index l_tabix.
    endif.
    "ENDLOOP. "ADü_20180426 - DP-1945
endloop.

"Start Change ADü_20180618 - DP-1210
"get the full scripts with comments:
class cl_oo include naming definition load.
data lo_clif_incl_naming type ref to if_oo_clif_incl_naming.
data lo_class_incl_naming type ref to if_oo_class_incl_naming.
data l_t_method_w_include type seop_methods_w_include.
field-symbols: <s_method_w_include> type seop_method_w_include.
data: l_clskey type seoclskey,
      lt_abap type abaptxt255_tab,
      ls_abap type line of abaptxt255_tab,
      length type i,
      last type char30.

"HANA Script
loop at lt_script into ls_script.

    concatenate '/BIC/' ls_script-procnm+3 into l_clskey.

    call method cl_oo_include_naming=>get_instance_by_cifkey
        exporting
            cifkey          = l_clskey
        receiving
            cifref          = lo_clif_incl_naming
        exceptions
            no_objecttype   = 1
            internal_error  = 2
            others          = 3.

    if sy-subrc = 0.
        lo_class_incl_naming ?= lo_clif_incl_naming.
        l_t_method_w_include =
            lo_class_incl_naming->get_all_method_includes( ).
        loop at l_t_method_w_include assigning <s_method_w_include>.
            clear: length, last.
            length = strlen( <s_method_w_include>-cpdkey ) - 9.

```

```

last = <s_method_w_include>-cpdkey+length.
if last = 'PROCEDURE'.
  clear lt_abap.
  read report <s_method_w_include>-incname into lt_abap.
  loop at lt_abap into ls_abap.
    ls_g_script-tranid      = ls_script-tranid.
    ls_g_script-procnm     = ls_script-procnm.
    ls_g_script-codeid     = ls_script-procnm+3.
    ls_g_script-script     = ls_abap.
    ls_g_script-line_no    = sy-tabix.
    append ls_g_script to lt_g_script.
  endloop.
endif.
endloop.
endif.
endloop.

```

"AMDP

```

loop at lt_sscript into ls_sscript.

  concatenate '/BIC/' ls_sscript-codeid into l_clskey.

  call method cl_oo_include_naming=>get_instance_by_cifkey
    exporting
      cifkey          = l_clskey
    receiving
      cifref          = lo_clif_incl_naming
    exceptions
      no_objecttype   = 1
      internal_error  = 2
      others          = 3.

  if sy-subrc = 0.
    lo_class_incl_naming ?= lo_clif_incl_naming.
    l_t_method_w_include =
      lo_class_incl_naming->get_all_method_includes( ).
    loop at l_t_method_w_include assigning <s_method_w_include>.
      clear: length, last.
      length = strlen( <s_method_w_include>-cpdkey ) - 9.
      last = <s_method_w_include>-cpdkey+length.
      if last = 'PROCEDURE'.
        clear lt_abap.
        read report <s_method_w_include>-incname into lt_abap.
        loop at lt_abap into ls_abap.
          ls_g_sscript-tranid      = ls_sscript-tranid.
          ls_g_sscript-codeid     = ls_sscript-codeid.
          ls_g_sscript-script     = ls_abap.
          ls_g_sscript-line_no    = sy-tabix.
          append ls_g_sscript to lt_g_sscript.
        endloop.
      endif.
    endloop.
  endif.
endif.
endloop.

```

"BW4HANA

```

loop at lt_rstransteprount into ls_rstransteprount.
  clear lt_abap.
  split ls_rstransteprount-code at cl_abap_char_utilities=>newline into table lt_abap.
  loop at lt_abap into ls_abap.
    ls_g_rstransteprount-tranid      = ls_rstransteprount-tranid.
    ls_g_rstransteprount-codeid     = ls_rstransteprount-codeid.
    ls_g_rstransteprount-code       = ls_abap.
    ls_g_rstransteprount-line_no    = sy-tabix.
    append ls_g_rstransteprount to lt_g_rstransteprount.
  endloop.
endloop.

```

"Search code in new tables:

"RSTRANSRIPT

```

loop at lt_g_script into ls_g_script.
  l_tabix = sy-tabix.
  clear ls_match_result.
  if i_case = ' ' and i_regex = 'X'.
    find first occurrence of regex ls_complstring
    in ls_g_script-script
    in character mode
    ignoring case
  endif.
endloop.

```

```

        results ls_match_result.
    elseif i_case = 'X' and i_regex = 'X'.
        find first occurrence of regex ls_complstring
        in ls_g_script-script
        in character mode
        respecting case
        results ls_match_result.
    elseif i_case = ' ' and i_regex = ''.
        find first occurrence of ls_complstring
        in ls_g_script-script
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = ''.
        find first occurrence of ls_complstring
        in ls_g_script-script
        in character mode
        respecting case
        results ls_match_result.
    endif.
    if sy-subrc <> 0.
        delete lt_g_script index l_tabix.
    endif.
endloop.

```

"RSTRANSTEPSCRIPT

```

loop at lt_g_sscript into ls_g_sscript.
    l_tabix = sy-tabix.
    clear ls_match_result.
    if i_case = ' ' and i_regex = 'X'.
        find first occurrence of regex ls_complstring
        in ls_g_sscript-script
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = 'X'.
        find first occurrence of regex ls_complstring
        in ls_g_sscript-script
        in character mode
        respecting case
        results ls_match_result.
    elseif i_case = ' ' and i_regex = ''.
        find first occurrence of ls_complstring
        in ls_g_sscript-script
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = ''.
        find first occurrence of ls_complstring
        in ls_g_sscript-script
        in character mode
        respecting case
        results ls_match_result.
    endif.
    if sy-subrc <> 0.
        delete lt_g_sscript index l_tabix.
    endif.
endloop.

```

"End Change ADÜ_20180618 - DP-1210

"RSTRANSTEPROUT

```

sort lt_g_rstransteprou by codeid.
loop at lt_g_rstransteprou into ls_g_rstransteprou.
    l_tabix = sy-tabix.
    clear ls_match_result.
    if i_case = ' ' and i_regex = 'X'.
        find first occurrence of regex ls_complstring
        in ls_g_rstransteprou-code
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = 'X'.
        find first occurrence of regex ls_complstring
        in ls_g_rstransteprou-code
        in character mode
        respecting case
        results ls_match_result.
    elseif i_case = ' ' and i_regex = ''.

```

```

        find first occurrence of ls_complstring
        in ls_g_rstransteprou-code
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = ''.
        find first occurrence of ls_complstring
        in ls_g_rstransteprou-code
        in character mode
        respecting case
        results ls_match_result.
    endif.
    if sy-subrc <> 0.
        delete lt_g_rstransteprou index l_tabix.
    endif.
endloop.

if gt_rsaabap is initial.
    "Start Change ADÜ_20180618 - DP-1210
    if lt_g_script is initial and lt_g_sscript is initial.
        "End Change ADÜ_20180618 - DP-1210
        if lt_g_rstransteprou is initial.
            return.
        endif.
        "Start Change ADÜ_20180618 - DP-1210
    endif.
    "End Change ADÜ_20180618 - DP-1210
endif.

"-----
" Scan routines of Update Rules (3.x)
"-----
if i_updru = 'X'.

    types:
        begin of s_rsupd,
            updid    like rsupdinfo-updid,
            infocube like rsupdinfo-infocube,
            isource  like rsupdinfo-isource,
            codeid   like rsupdrout-codeid,
            routine  like rsupdrout-routine,
            iciobjnm like rsupdrout-iciobjnm,
        end of s_rsupd,
        t_rsupd type standard table of s_rsupd.

    data: lt_rsupd type t_rsupd.

    field-symbols <s_rsupd> type s_rsupd.

    clear lt_rsaabap.
    loop at gt_rsaabap into ls_rsaabap where codetp = 'UR'.
        append ls_rsaabap to lt_rsaabap.
    endloop.
    if not lt_rsaabap is initial.

        clear lt_lookup_temp.
        select a~updid infocube isource codeid routine iciobjnm
        into table lt_rsupd
        from rsupdinfo as a
        inner join rsupdrout as b on a~updid = b~updid
        for all entries in lt_rsaabap
        where
            b~codeid = lt_rsaabap-codeid and
            a~objvers = 'A' and
            b~objvers = 'A'.

        loop at lt_rsupd assigning <s_rsupd>.
            ls_lookup_temp-targetname = <s_rsupd>-infocube.
            ls_lookup_temp-sourcename = <s_rsupd>-isource.
            ls_lookup_temp-codeid = <s_rsupd>-codeid.
            if <s_rsupd>-routine = '9998'.
                ls_lookup_temp-routinetype = 'RT07'.
            elseif <s_rsupd>-routine = '9999'.
                ls_lookup_temp-routinetype = 'RT09'.
            else.
                ls_lookup_temp-routinetype = 'RT08'.
            endif.
        endloop.
    endif.
endif.

```

```

ls_lookup_temp-routine = <s_rsupd>-routine.
ls_lookup_temp-tranid = <s_rsupd>-updid.
ls_lookup_temp-iobjnm = <s_rsupd>-iciobjnm.
append ls_lookup_temp to lt_lookup_temp.
endloop.

sort lt_lookup_temp by codeid.
loop at lt_rsaabap into ls_rsaabap.

clear ls_lookup.
read table lt_lookup_temp into ls_lookup
with key codeid = ls_rsaabap-codeid binary search.
if sy-subrc = 0.
clear l_updtype.
select single updtype into l_updtype from rsupddat
where
updid = ls_lookup-tranid and
objvers = 'A' and
routine = ls_lookup-routine.          "#EC CI_NOORDER

"no update
if l_updtype = 'NOP'.
delete lt_rsaabap.
else.
clear ls_result.
move-corresponding ls_lookup to ls_result.
move-corresponding ls_rsaabap to ls_result.
"Derive description of the Update Rule
concatenate ls_lookup-targetname ls_lookup-sourcename
into ls_result-txtlg separated by space.
append ls_result to gt_result.
endif.
endif.

endloop.

endif.
endif.

"-----
" Scan routines of Transfer Rules (3.x)
"-----
if i_tranr = 'X'.

clear lt_rsaabap.
loop at gt_rsaabap into ls_rsaabap where codetp = 'TR'.
append ls_rsaabap to lt_rsaabap.
endloop.
if not lt_rsaabap is initial.

clear lt_lookup_temp.

sort lt_lookup_temp by codeid.
loop at lt_rsaabap into ls_rsaabap.
clear ls_lookup.
read table lt_lookup_temp into ls_lookup
with key codeid = ls_rsaabap-codeid binary search.
if sy-subrc = 0.
clear ls_result.
if ls_lookup-iobjnm is not initial.
ls_lookup-routinetype = 'RT11'.
endif.
move-corresponding ls_lookup to ls_result.
move-corresponding ls_rsaabap to ls_result.

append ls_result to gt_result.
endif.
endloop.

endif.
endif.

"-----
" Scan routines of Transformations (7.x)
"-----
if i_trans = 'X'.

clear lt_rsaabap.

```

```

loop at gt_rsaabap into ls_rsaabap where codetp = 'TF'.
  append ls_rsaabap to lt_rsaabap.
endloop.
if not lt_rsaabap is initial.

  "----- Transformations (rules)
  clear lt_lookup_temp.
  select targetname targettype targetsubtype sourcename sourcetype
    sourcesubtype codeid a~tranid fieldnm
    into table lt_lookup_temp
    from rstran as a
    inner join rstranroutmap as b on a~tranid = b~tranid
    for all entries in lt_rsaabap
    where
      expert      = ''                and
      b~codeid    = lt_rsaabap-codeid and
      a~objvers   = 'A'              and
      b~objvers   = 'A'              and
      param_id    = 1                and
      paramtype   = 1.                "#EC *

  loop at lt_lookup_temp into ls_lookup.
    ls_lookup-routinetype = 'RT04'.
    append ls_lookup to lt_lookup_finder.
  endloop.

  "----- Transformations (start routine)
  clear lt_lookup_temp.
  select targetname targettype targetsubtype sourcename sourcetype
    sourcesubtype startroutine tranid
    into table lt_lookup_temp from rstran
    for all entries in lt_rsaabap
    where
      startroutine = lt_rsaabap-codeid and
      objvers      = 'A'.                "#EC *

  loop at lt_lookup_temp into ls_lookup.
    ls_lookup-routinetype = 'RT01'.
    append ls_lookup to lt_lookup_finder.
  endloop.

  "----- Transformations (end routine)
  clear lt_lookup_temp.
  select targetname targettype targetsubtype sourcename sourcetype
    sourcesubtype endroutine tranid
    into table lt_lookup_temp from rstran
    for all entries in lt_rsaabap
    where
      endroutine = lt_rsaabap-codeid and
      objvers    = 'A'.                "#EC *

  loop at lt_lookup_temp into ls_lookup.
    ls_lookup-routinetype = 'RT02'.
    append ls_lookup to lt_lookup_finder.
  endloop.

  "----- Transformations (expert routine)
  clear lt_lookup_temp.
  select targetname targettype targetsubtype sourcename sourcetype
    sourcesubtype expert tranid
    into table lt_lookup_temp from rstran
    for all entries in lt_rsaabap
    where
      expert      = lt_rsaabap-codeid and
      objvers     = 'A'.                "#EC *

  loop at lt_lookup_temp into ls_lookup.
    ls_lookup-routinetype = 'RT03'.
    append ls_lookup to lt_lookup_finder.
  endloop.

  "----- Transformations (Global declaration part 1)
  clear lt_lookup_temp.
  select targetname targettype targetsubtype sourcename sourcetype
    sourcesubtype glbcode tranid
    into table lt_lookup_temp from rstran
    for all entries in lt_rsaabap
    where

```

```

        glbcode = lt_rsaabap-codeid and
        objvers = 'A'.                                     "#EC *

loop at lt_lookup_temp into ls_lookup.
    ls_lookup-routinetype = 'RT05'.
    append ls_lookup to lt_lookup_finder.
endloop.

"----- Transformations (Global declaration part 2)
clear lt_lookup_temp.
select targetname targettype targetsubtype sourcename sourcetype
    sourcesubtype glbcode2 tranid
into table lt_lookup_temp from rstran
for all entries in lt_rsaabap
where
    glbcode2 = lt_rsaabap-codeid and
    objvers = 'A'.                                     "#EC *

loop at lt_lookup_temp into ls_lookup.
    ls_lookup-routinetype = 'RT06'.
    append ls_lookup to lt_lookup_finder.
endloop.
"-----
sort lt_lookup_finder by codeid.
loop at lt_rsaabap into ls_rsaabap.
    clear ls_lookup.
    read table lt_lookup_finder into ls_lookup
        with key codeid = ls_rsaabap-codeid binary search.
    if sy-subrc = 0.
        clear ls_result.
        if ls_lookup-sourcetype = 'RSDS'.
            condense ls_lookup-sourcename.
        endif.
        move-corresponding ls_lookup to ls_result.
        move-corresponding ls_rsaabap to ls_result.
        "Get description of Transformation
        clear lv_txtlg.
*****New ALU 20250630
        PERFORM get_txtlg USING ls_lookup-tranid CHANGING lv_txtlg.
*****End ALU 20250630
        if sy-subrc <> 0.
            "If no text found derive description of the Transformation
            concatenate ls_lookup-sourcetype ls_lookup-sourcename '->'
                ls_lookup-targettype ls_lookup-targetname
                into lv_txtlg separated by space.

        endif.
        ls_result-txtlg = lv_txtlg.

        append ls_result to gt_result.
    endif.
endloop.
endif.

"Start Change ADü 20180618 - DP-1210
"handle AMDP Scripts
"RT20 -> HANA Script (Expert Routine)
"RT21 -> AMDP Start Routine
"RT22 -> AMDP End Routine
"RT23 -> AMDP Expert Routine
"RT24 -> AMDP InfoObject Routine

"HANA Script
if lt_g_script is not initial.
    clear lt_lookup_finder.
    "----- Transformations - HANA Script (Expert Routine)
    clear lt_lookup_temp.
    select targetname targettype targetsubtype sourcename sourcetype
        sourcesubtype tranid
    into corresponding fields of table lt_lookup_temp
    from rstran
    for all entries in lt_g_script
    where tranid = lt_g_script-tranid
    and objvers = 'A'.                                     "#EC *

loop at lt_lookup_temp into ls_lookup.
    ls_lookup-routinetype = 'RT20'.
    append ls_lookup to lt_lookup_finder.
endloop.

```

```

"-----
sort lt_lookup_finder by codeid.
loop at lt_g_script into ls_g_script.
  clear ls_lookup.
  read table lt_lookup_finder into ls_lookup
  with key tranid = ls_g_script-tranid binary search.
  if sy-subrc = 0.
    clear ls_result.
    if ls_lookup-sourcetype = 'RSDS'.
      condense ls_lookup-sourcename.
    endif.
    move-corresponding ls_lookup to ls_result.
    move-corresponding ls_g_script to ls_result.
    ls_result-line = ls_g_script-script.
    "Get description of Transformation
    clear lv_txtlg.
*****New ALU_20250630
    PERFORM get_txtlg USING ls_lookup-tranid CHANGING lv_txtlg.
*****End ALU_20250630
    if sy-subrc <> 0.
      "If no text found derive description of the Transformation
      concatenate ls_lookup-sourcetype ls_lookup-sourcename '-'>
        ls_lookup-targettype ls_lookup-targetname
        into lv_txtlg separated by space.
    endif.
    ls_result-txtlg = lv_txtlg.
    ls_result-codetp = 'TF'.

    append ls_result to gt_result.
  endif.
endloop.

endif.

"AMDP
if lt_g_sscript is not initial.
  clear lt_lookup_finder.
  "----- Transformations - AMDP Start Routine
  clear lt_lookup_temp.
  select targetname targettype targetsubtype sourcename sourcetype
    sourcesubtype startroutine tranid
    into table lt_lookup_temp from rstran
  for all entries in lt_g_sscript
  where
    startroutine = lt_g_sscript-codeid and
    objvers      = 'A'.
                                     "#EC *

  loop at lt_lookup_temp into ls_lookup.
    ls_lookup-routinetype = 'RT21'.
    append ls_lookup to lt_lookup_finder.
  endloop.

  "----- Transformations - AMDP End Routine
  clear lt_lookup_temp.
  select targetname targettype targetsubtype sourcename sourcetype
    sourcesubtype endroutine tranid
    into table lt_lookup_temp from rstran
  for all entries in lt_g_sscript
  where
    endroutine = lt_g_sscript-codeid and
    objvers    = 'A'.
                                     "#EC *

  loop at lt_lookup_temp into ls_lookup.
    ls_lookup-routinetype = 'RT22'.
    append ls_lookup to lt_lookup_finder.
  endloop.

  "----- Transformations - AMDP Expert Routine
  clear lt_lookup_temp.
  select targetname targettype targetsubtype sourcename sourcetype
    sourcesubtype expert tranid
    into table lt_lookup_temp from rstran
  for all entries in lt_g_sscript
  where
    expert = lt_g_sscript-codeid and
    objvers = 'A'.
                                     "#EC *

  loop at lt_lookup_temp into ls_lookup.

```

```

ls_lookup-routinetype = 'RT23'.
append ls_lookup to lt_lookup_finder.
endloop.

"----- Transformations - AMDP InfoObject Routine
clear lt_lookup_temp.
select targetname targettype targetsubtype sourcename sourcetype
sourcesubtype codeid a~tranid fieldnm
into table lt_lookup_temp
from rstran as a
inner join rstranroutmap as b on a~tranid = b~tranid
for all entries in lt_g_sscript
where
  expert      = ''                and
  b~codeid    = lt_g_sscript-codeid and
  a~objvers   = 'A'              and
  b~objvers   = 'A'              and
  param_id    = 1                and
  paramtype   = 1.                "#EC *

loop at lt_lookup_temp into ls_lookup.
  ls_lookup-routinetype = 'RT24'.
  append ls_lookup to lt_lookup_finder.
endloop.

"-----
sort lt_lookup_finder by codeid.
loop at lt_g_sscript into ls_g_sscript.
  clear ls_lookup.
  read table lt_lookup_finder into ls_lookup
  with key codeid = ls_g_sscript-codeid binary search.
  if sy-subrc = 0.
    clear ls_result.
    if ls_lookup-sourcetype = 'RSDS'.
      condense ls_lookup-sourcename.
    endif.
    move-corresponding ls_lookup to ls_result.
    move-corresponding ls_g_sscript to ls_result.
    ls_result-line = ls_g_sscript-script.
    "Get description of Transformation
    clear lv_txtlg.
    *****New ALU_20250630
    PERFORM get_txtlg USING ls_lookup-tranid CHANGING lv_txtlg.
    *****End ALU_20250630
    if sy-subrc <> 0.
      "If no text found derive description of the Transformation
      concatenate ls_lookup-sourcetype ls_lookup-sourcename '-'>
        ls_lookup-targettype ls_lookup-targetname
        into lv_txtlg separated by space.
    endif.
    ls_result-txtlg = lv_txtlg.
    ls_result-codetp = 'TF'.

    append ls_result to gt_result.
  endif.
endloop.
endif.
"End Change ADÜ_20180618 - DP-1210

"BW4/HANA
if lt_g_rstransteprout is not initial.
  clear lt_lookup_finder.
  clear lt_lookup_temp.
  "----- Transformations - Start Routine
  select ('targetname targettype targetsubtype sourcename sourcetype sourcesubtype
startroutine tranid set_runtime')
  into table lt_lookup_temp from rstran
  for all entries in lt_g_rstransteprout
  where
    startroutine = lt_g_rstransteprout-codeid and
    objvers      = 'A'.                "#EC *

loop at lt_lookup_temp into ls_lookup.
  if ls_lookup-set_runtime = 'O'.
    ls_lookup-routinetype = 'RT34'.
  else.
    ls_lookup-routinetype = 'RT30'.
  endif.

```

```

        append ls_lookup to lt_lookup_finder.
    endloop.
    "----- Transformations - End Routine
    clear lt_lookup_temp.
    select ('targetname targettype targetsubtype sourcename sourcetype sourcesubtype endroutine
    tranid set_runtime')
        into table lt_lookup_temp from rstran
        for all entries in lt_g_rstranstestrout
        where
            endroutine = lt_g_rstranstestrout-codeid and
            objvers = 'A'.
    "EC *

    loop at lt_lookup_temp into ls_lookup.
        if ls_lookup-set_runtime = 'O'.
            ls_lookup-routinetype = 'RT35'.
        else.
            ls_lookup-routinetype = 'RT31'.
        endif.
        append ls_lookup to lt_lookup_finder.
    endloop.
    "----- Transformations - Expert Routine
    clear lt_lookup_temp.
    select ('targetname targettype targetsubtype sourcename sourcetype sourcesubtype expert
    tranid set_runtime')
        into table lt_lookup_temp from rstran
        for all entries in lt_g_rstranstestrout
        where
            expert = lt_g_rstranstestrout-codeid and
            objvers = 'A'.
    "EC *

    loop at lt_lookup_temp into ls_lookup.
        if ls_lookup-set_runtime = 'O'.
            ls_lookup-routinetype = 'RT37'.
        else.
            ls_lookup-routinetype = 'RT33'.
        endif.
        append ls_lookup to lt_lookup_finder.
    endloop.
    "----- Transformations - InfoObject Routine
    clear lt_lookup_temp.
    select ('targetname targettype targetsubtype sourcename sourcetype sourcesubtype codeid
    a~tranid a~set_runtime fieldnm')
        into table lt_lookup_temp
        from rstran as a
        inner join rstranroutmap as b on a~tranid = b~tranid
        for all entries in lt_g_rstranstestrout
        where
            expert = '' and
            b~codeid = lt_g_rstranstestrout-codeid and
            a~objvers = 'A' and
            b~objvers = 'A' and
            param_id = 1 and
            paramtype = 1.
    "EC *

    loop at lt_lookup_temp into ls_lookup.
        if ls_lookup-set_runtime = 'O'.
            ls_lookup-routinetype = 'RT36'.
        else.
            ls_lookup-routinetype = 'RT32'.
        endif.
        append ls_lookup to lt_lookup_finder.
    endloop.
    "----- Transformations (Global declaration part 1)
    clear lt_lookup_temp.
    select targetname targettype targetsubtype sourcename sourcetype
        sourcesubtype glbcode tranid
        into table lt_lookup_temp from rstran
        for all entries in lt_g_rstranstestrout
        where
            glbcode = lt_g_rstranstestrout-codeid and
            objvers = 'A'.
    "EC *

    loop at lt_lookup_temp into ls_lookup.
        ls_lookup-routinetype = 'RT05'.
        append ls_lookup to lt_lookup_finder.
    endloop.
    "----- Transformations (Global declaration part 2)
    clear lt_lookup_temp.

```

```

select targetname targettype targetsubtype sourcename sourcetype
  sourcesubtype glbcode2 tranid
into table lt_lookup_temp from rstran
for all entries in lt_g_rstransteprout
where
  glbcode2 = lt_g_rstransteprout-codeid and
  objvers = 'A'.                                     "#EC *

loop at lt_lookup_temp into ls_lookup.
  ls_lookup-routinetype = 'RT06'.
  append ls_lookup to lt_lookup_finder.
endloop.
"-----
sort lt_lookup_finder by codeid.
loop at lt_g_rstransteprout into ls_g_rstransteprout.
  clear ls_lookup.
  read table lt_lookup_finder into ls_lookup
  with key codeid = ls_g_rstransteprout-codeid binary search.
  if sy-subrc = 0.
    clear ls_result.
    if ls_lookup-sourcetype = 'RSDS'.
      condense ls_lookup-sourcename.
    endif.
    move-corresponding ls_lookup to ls_result.
    move-corresponding ls_g_rstransteprout to ls_result.
    ls_result-line = ls_g_rstransteprout-code.
    "Get description of Transformation
    clear lv_txtlg.
*****New ALU_20250630
    PERFORM get_txtlg USING ls_lookup-tranid CHANGING lv_txtlg.
*****End ALU_20250630
    if sy-subrc <> 0.
      "If no text found derive description of the Transformation
      concatenate ls_lookup-sourcetype ls_lookup-sourcename '->'
        ls_lookup-targettype ls_lookup-targetname
        into lv_txtlg separated by space.

    endif.
    ls_result-txtlg = lv_txtlg.
    ls_result-codetp = 'TF'.

    append ls_result to gt_result.
  endif.
endloop.
endif.

endif.

"-----
" Scan InfoObject (Char.) Transfer-Routines
"-----
if i_iobjrout = 'X'.

  types:
    begin of s_chabas,
      chabasnm like rsdchabas-chabasnm,
      iobjrout like rsdchabas-iobjrout,
    end of s_chabas,
    t_chabas type standard table of s_chabas.

  data lt_chabas type t_chabas.
  field-symbols <s_chabas> type s_chabas.

  clear lt_rsaabap.
  loop at gt_rsaabap into ls_rsaabap where codetp = 'IC'.
    append ls_rsaabap to lt_rsaabap.
  endloop.
  if not lt_rsaabap is initial.

    clear: lt_lookup_temp, lt_chabas.
    select chabasnm iobjrout
      into table lt_chabas
      from rsdchabas
      for all entries in lt_rsaabap
      where
        iobjrout = lt_rsaabap-codeid and
        objvers = 'A'.                                     "#EC *

    loop at lt_chabas assigning <s_chabas>.

```

```

        clear ls_lookup.
        ls_lookup-iobjnm = <s_chabas>-chabasnm.
        ls_lookup-codeid = <s_chabas>-iobjrout.
        append ls_lookup to lt_lookup_temp.
    endloop.

    sort lt_lookup_temp by codeid.
    loop at lt_rsaabap into ls_rsaabap.
        clear ls_lookup.
        read table lt_lookup_temp into ls_lookup
            with key codeid = ls_rsaabap-codeid binary search.
        if sy-subrc = 0.
            clear ls_result.
            ls_lookup-routinetype = 'RT15'.
            move-corresponding ls_lookup to ls_result.
            move-corresponding ls_rsaabap to ls_result.
            append ls_result to gt_result.
        endif.
    endloop.

endif.
endif.

endform.                                "scan_rsaabap

*&-----*
*&      Form  scan_dtp
*&-----*
*      text
*-----*
form scan_dtp
    tables lt_string t_devcl_range
        using i_skip_comment i_case i_regex.

data:
    l_t_dtp_range      type t_devcl_range,
    l_s_dtp_range      type s_devcl_range,
    lt_obj              type standard table of tadir-obj_name,
    ls_match_result     type match_result,
    l_dtpnm            type rsbkdtprm,
    ls_rsbkdtpt        like rsbkdtpt,
    lt_rsbkdtpt        like table of rsbkdtpt,
    lr_cl_rsbk_dtp      type ref to cl_rsbk_dtp,
    lr_cl_rsbk_filter   type ref to cl_rsbk_filter,
    ls_dtprule         type mch_s_sourcecode,
    lt_dtprule         type mch_t_sourcecode,
    ls_result          type s_result.

field-symbols:
    <string> type s_string,
    <s_range> type rsrange,
    <s_obj>   like line of lt_obj.
"-----

select obj_name into table lt_obj from tadir
where
    pgmid      = 'R3TR'          and
    object     = 'DTPA'.        "#EC *

select * into table lt_rsbkdtpt from rsbkdtpt
where
    langu      = sy-langu and
    objvers    = 'A'.

sort lt_rsbkdtpt by dtp.

loop at lt_obj assigning <s_obj>.

    clear:
        lr_cl_rsbk_dtp,
        lr_cl_rsbk_filter.

    "==== Get Filter-Routines =====
    l_dtpnm = <s_obj>.
    lr_cl_rsbk_dtp = cl_rsbk_dtp=>factory( l_dtpnm ).
    try.

```

```

    lr_cl_rsbcb_filter = lr_cl_rsbcb_dtp->get_obj_ref_filter( ).
    catch cx_rs_access_error.
        continue.
    endtry.

loop at lr_cl_rsbcb_filter->n_t_dtprule into ls_dtprule.

loop at lt_string assigning <string>.
    clear ls_match_result.
    if i_case = ' ' and i_regex = ' '.
        find first occurrence of <string> in ls_dtprule-line
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = ' '.
        find first occurrence of <string> in ls_dtprule-line
        in character mode
        respecting case
        results ls_match_result.
    elseif i_case = ' ' and i_regex = 'X'.
        find first occurrence of regex <string> in ls_dtprule-line
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = 'X'.
        find first occurrence of regex <string> in ls_dtprule-line
        in character mode
        respecting case
        results ls_match_result.
    endif.
    if ls_match_result-offset <> 0.
        clear ls_result.
        ls_result-codetp      = 'DTP'.
        ls_result-targetname = <s_obj>.
        ls_result-routinetype = 'RT13'.
        ls_result-iobjnm     = ls_dtprule-field.
        ls_result-line_no    = ls_dtprule-line_no.
        ls_result-line       = ls_dtprule-line.
        "Get description
        read table lt_rsbkdtpt into ls_rsbkdtpt
        with key dtp = <s_obj> binary search.
        if sy-subrc = 0.
            ls_result-txtlg = ls_rsbkdtpt-txtlg.
        endif.
        append ls_result to gt_result.
    endif.
endloop.

endloop.

endloop.

endform.                                "scan_dtp

*&-----*
*&      Form  scan_ip
*&-----*
form scan_ip
    tables lt_string t_devcl_range
    using i_skip_comment i_case i_regex.
    data:
        l_t_devcl_range type t_devcl_range,
        l_s_devcl_range type s_devcl_range,
        lt_obj          type standard table of tadir-obj_name,
        ls_rslldprule   like rslldprule,
        lt_rslldprule   like table of ls_rslldprule,
        ls_match_result type match_result,
        ls_rslldpiot    like rslldpiot,
        lt_rslldpiot    like table of rslldpiot,
        ls_result       type s_result.

    field-symbols:
        <string> type s_string,
        <s_range> type rsrange,
        <s_obj>   like line of lt_obj.

```

```

loop at t_devcl_range assigning <s_range>.
  l_s_devcl_range-sign    = <s_range>-sign.
  l_s_devcl_range-option  = <s_range>-option.
  l_s_devcl_range-low     = <s_range>-low.
  l_s_devcl_range-high    = <s_range>-high.
  append l_s_devcl_range to l_t_devcl_range.
endloop.

"-----

select obj_name into table lt_obj from tadir
  where
    pgmid      = 'R3TR'          and
    object     = 'ISIP'.        "#EC *

select * into table lt_rsldpiot from rsldpiot
  where
    langu = sy-langu and
    objvers = 'A'.

sort lt_rsldpiot by logdpid.

loop at lt_obj assigning <s_obj>.

  clear lt_rsl DPRule.
  select
    r~logdpid
    r~objvers
    r~fieldname
    r~lnr
    r~iobjnm
    line
  into table lt_rsl DPRule
  from rsl Dpsel as s inner join rsl DPRule as r
  on s~logdpid = r~logdpid and
    s~objvers = r~objvers and
    s~fieldname = r~fieldname
  where
    s~logdpid = <s_obj> and
    s~objvers = 'A'.

loop at lt_rsl DPRule into ls_rsl DPRule.

  loop at lt_string assigning <string>.

    clear ls_match_result.
    if i_case = 'I' and i_regex = ' '.
      find first occurrence of <string> in ls_rsl DPRule-line
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = ' '.
      find first occurrence of <string> in ls_rsl DPRule-line
        in character mode
        respecting case
        results ls_match_result.
    elseif i_case = ' ' and i_regex = 'X'.
      find first occurrence of regex <string> in ls_rsl DPRule-line
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = 'X'.
      find first occurrence of regex <string> in ls_rsl DPRule-line
        in character mode
        respecting case
        results ls_match_result.
    endif.

    if ls_match_result-offset <> 0.
      clear ls_result.
      ls_result-codetp      = 'IP'.
      ls_result-targetname  = <s_obj>.
      ls_result-routinetype = 'RT14'.
      ls_result-txtlg       = ls_rsl DPRule-fieldname.
      ls_result-iobjnm      = ls_rsl DPRule-iobjnm.
      ls_result-line_no     = sy-tabix.
      ls_result-line       = ls_rsl DPRule-line.

```

```

        read table lt_rslldpiot into ls_rslldpiot
        with key logdpid = ls_rslldprule-logdpid binary search.
        if sy-subrc = 0.
            ls_result-txtlg = ls_rslldpiot-text.
        endif.
        append ls_result to gt_result.
    endif.

endloop.

endloop.

endloop.
endform.                                "scan_ip

*&-----*
*&      Form  Get_REPS
*&-----*
*      text
*-----*
form get_reps
    tables t_devcl_range t_reps_range t_subc_range.

data:
    l_t_devcl_range type t_devcl_range,
    l_s_devcl_range type s_devcl_range,
    l_t_name_range  type t_name_range,
    l_s_name_range  type s_name_range,
    l_t_subc_range  type t_subc_range,
    l_s_subc_range  type s_subc_range.

field-symbols:
    <s_range>      type rsrange.

"-----"
" Range Importparam. in Selektionstab. mit passendem Typ übertragen
"-----"
loop at t_devcl_range assigning <s_range>.
    l_s_devcl_range-sign = <s_range>-sign.
    l_s_devcl_range-option = <s_range>-option.
    l_s_devcl_range-low   = <s_range>-low.
    l_s_devcl_range-high  = <s_range>-high.
    append l_s_devcl_range to l_t_devcl_range.
endloop.

loop at t_reps_range assigning <s_range>.
    l_s_name_range-sign = <s_range>-sign.
    l_s_name_range-option = <s_range>-option.
    l_s_name_range-low   = <s_range>-low.
    l_s_name_range-high  = <s_range>-high.
    append l_s_name_range to l_t_name_range.
endloop.

loop at t_subc_range assigning <s_range>.
    l_s_subc_range-sign = <s_range>-sign.
    l_s_subc_range-option = <s_range>-option.
    l_s_subc_range-low   = <s_range>-low.
    l_s_subc_range-high  = <s_range>-high.
    append l_s_subc_range to l_t_subc_range.
endloop.
"-----"

select object obj_name into table gt_object from tadir
as a inner join trdir as r on a~obj_name = r~name
where
    pgmid    = 'R3TR'    and
    object   = 'PROG'    and
    obj_name in l_t_name_range and
    devclass in l_t_devcl_range and
    subc     in l_t_subc_range.                                "#EC *

endform.                                "Get_REPS

*&-----*
*&      Form  get_FUNC

```

```

*&-----*
*          text
*-----*
*      -->I_T_DEVCL_RANGE  text
*      -->I_T_FUGR_RANGE  text
*-----*
form get_func
  tables t_devcl_range t_fugr_range.

data:
  l_t_devcl_range type t_devcl_range,
  l_s_devcl_range type s_devcl_range,
  l_t_fugr_range  type t_fugr_range,
  l_s_fugr_range  type s_fugr_range,
  lt_obj          type standard table of s_object,
  l_fgroup        type rs381-area,
  l_program       type progname.

field-symbols:
  <s_range> type rsrange,
  <s_obj>    like line of lt_obj.

"-----"
" Range Importparam. in Selektionstab. mit passendem Typ übertragen
"-----"
loop at t_devcl_range assigning <s_range>.
  l_s_devcl_range-sign = <s_range>-sign.
  l_s_devcl_range-option = <s_range>-option.
  l_s_devcl_range-low = <s_range>-low.
  l_s_devcl_range-high = <s_range>-high.
  append l_s_devcl_range to l_t_devcl_range.
endloop.

loop at t_fugr_range assigning <s_range>.
  l_s_fugr_range-sign = <s_range>-sign.
  l_s_fugr_range-option = <s_range>-option.
  l_s_fugr_range-low = <s_range>-low.
  l_s_fugr_range-high = <s_range>-high.
  append l_s_fugr_range to l_t_fugr_range.
endloop.
"-----"

select object obj_name into table lt_obj from tadir
  where
    pgmid = 'R3TR' and
    object = 'FUGR' and
    obj_name in l_t_fugr_range and
    devclass in l_t_devcl_range. "#EC *

loop at lt_obj assigning <s_obj>.
  l_fgroup = <s_obj>-obj_name.
  clear l_program.

  call function 'FUNCTION_INCLUDE_CONCATENATE'
    changing
      program = l_program
      complete_area = l_fgroup
    exceptions
      not_enough_input = 1
      no_function_pool = 2
      delimiter_wrong_position = 3
      others = 4.

  check sy-subrc is initial and l_program is not initial.
  <s_obj>-object = 'FUGR'.
  <s_obj>-incl_name = l_program.
  append <s_obj> to gt_object.
endloop.

endform. "get_FUNC

*&-----*
*& Form get_plse
*&-----*
* Search in Planning Functions
*-----*

```

```

form get_plse
  tables lt_string t_devcl_range
  using i_skip_comment i_case i_regex.

data:
  lt_obj          type standard table of tadir-obj_name,
  ls_rsplf_srv_p  type rsplf_srv_p,
  lt_rsplf_srv_p  type table of rsplf_srv_p,
  ls_match_result type match_result,
  ls_rsplf_srvt   type rsplf_srvt,
  lt_rsplf_srvt   type table of rsplf_srvt,
  ls_result       type s_result.

field-symbols:
  <string> type s_string,
  <s_range> type rsrange,
  <s_obj>   like line of lt_obj.
"-----

select obj_name into table lt_obj from tadir
  where
    pgmid   = 'R3TR' and
    object  = 'PLSE'.                                "#EC *

select * into table lt_rsplf_srvt from rsplf_srvt
  where
    langu = sy-langu and
    objvers = 'A'.                                    "#EC *

sort lt_rsplf_srvt by srvnm.

loop at lt_obj assigning <s_obj>.

  clear lt_rsplf_srv_p.
  select * from rsplf_srv_p
    into table lt_rsplf_srv_p
    where
      srvnm = <s_obj> and
      objvers = 'A' and
      parnm = 'FLINE'.                                "#EC *

**new ADÜ_20170613
  data: lv_string type string,
        ltSplitted type table of string,
        lsSplitted type string,
        lv_index type i value 0.

  clear: lv_string, lv_index.

  loop at lt_rsplf_srv_p into ls_rsplf_srv_p.
    concatenate lv_string ls_rsplf_srv_p-value into lv_string.
  endloop.

  split lv_string at cl_abap_char_utilities=>cr_lf into table ltSplitted.

  read table lt_rsplf_srv_p into ls_rsplf_srv_p index 1. "#EC CI_NOORDER

  loop at ltSplitted into lsSplitted.
    lv_index = lv_index + 1.

    "LOOP AT lt_rsplf_srv_p INTO ls_rsplf_srv_p.

**end new ADÜ_20170613

  loop at lt_string assigning <string>.

    clear ls_match_result.
    if i_case = 'I' and i_regex = ' '.
      find first occurrence of <string> in lsSplitted "ls_rsplf_srv_p-value
        in character mode
        ignoring case
        results ls_match_result.
    elseif i_case = 'X' and i_regex = ' '.
      find first occurrence of <string> in lsSplitted "ls_rsplf_srv_p-value
        in character mode
        respecting case
        results ls_match_result.

```

```

elseif i_case = ' ' and i_regex = 'X'.
    find first occurrence of regex <string>
    in ls_splitted "ls_rsplf_srv_p-value"
    in character mode
    ignoring case
    results ls_match_result.
elseif i_case = 'X' and i_regex = 'X'.
    find first occurrence of regex <string>
    in ls_splitted "ls_rsplf_srv_p-value"
    in character mode
    respecting case
    results ls_match_result.
endif.

if ls_match_result-offset <> 0.
    clear ls_result.
    ls_result-codetp      = 'PF'.
    ls_result-targetname  = <s_obj>.
    ls_result-routinetype = 'RT19'.
    ls_result-txtlg       = ''.
    ls_result-iobjnm      = ''.
    ls_result-line_no     = lv_index. "ls_rsplf_srv_p-indx.
    ls_result-line        = ls_splitted. "ls_rsplf_srv_p-value.
    read table lt_rsplf_srvt into ls_rsplf_srvt
    with key srvnm = ls_rsplf_srv_p-srvnm binary search.
    if sy-subrc = 0.
        ls_result-txtlg = ls_rsplf_srvt-txtlg.
    endif.
    append ls_result to gt_result.
endif.

endloop.

endloop.

endloop.

endform.                                "get_plse

*&-----*
*&      Form  get_clas
*&-----*
*      text
*-----*
form get_clas
    tables t_devcl_range t_clas_range.

    data:
        l_t_devcl_range type t_devcl_range,
        l_s_devcl_range type s_devcl_range,
        l_t_clas_range  type t_clas_range,
        l_s_clas_range  type s_clas_range,
        lt_obj           type standard table of s_object,
        l_clskey         type seoclskey,
        l_obj            type tadir-obj_name,
        lt_includes      type seop_methods_w_include.

    field-symbols:
        <s_range> type rsrange,
        <s_obj>    like line of lt_obj,
        <s_include> like line of lt_includes.

    "-----"
    " Range Importparam. in Selektionstab. mit passendem Typ übertragen
    "-----"

    loop at t_devcl_range assigning <s_range>.
        l_s_devcl_range-sign = <s_range>-sign.
        l_s_devcl_range-option = <s_range>-option.
        l_s_devcl_range-low = <s_range>-low.
        l_s_devcl_range-high = <s_range>-high.
        append l_s_devcl_range to l_t_devcl_range.
    endloop.

    loop at t_clas_range assigning <s_range>.
        l_s_clas_range-sign = <s_range>-sign.
        l_s_clas_range-option = <s_range>-option.

```

```

l_s_clas_range-low      = <s_range>-low.
l_s_clas_range-high     = <s_range>-high.
append l_s_clas_range to l_t_clas_range.
endloop.
"-----

select object obj_name into table lt_obj from tadir
where
  pgmid      = 'R3TR'          and
  object     = 'CLAS'          and
  obj_name in l_t_clas_range and
  devclass in l_t_devcl_range.                                "#EC *

loop at lt_obj assigning <s_obj>.
  l_clskey = <s_obj>-obj_name.
* ts 28.10.2014 - Start comment
*   CALL FUNCTION 'SEO_CLASS_GET_METHOD_INCLUDES'
*     EXPORTING
*       clskey                = l_clskey
*     IMPORTING
*       includes              = lt_includes
*     EXCEPTIONS
*       _internal_class_not_existing = 1
*       OTHERS                 = 2.
*
*   LOOP AT lt_includes ASSIGNING <s_include>.
*     <s_obj>-object = 'METH'.
*     <s_obj>-incl_name = <s_include>-incname.
*     <s_obj>-obj_name = l_clskey.
*     <s_obj>-meth_name = <s_include>-cpdkey+30.
*     APPEND <s_obj> TO gt_object.
*   ENDLOOP.
* ts 28.10.2014 - End comment

* ts 28.10.2014 - Start insert
class cl_oo_include_naming definition load.
data lo_clif_incl_naming type ref to if_oo_clif_incl_naming.
data lo_class_incl_naming type ref to if_oo_class_incl_naming.
data l_t_method_w_include type seop_methods_w_include.
field-symbols: <s_method_w_include> type seop_method_w_include.

call method cl_oo_include_naming=>get_instance_by_cifkey
  exporting
    cifkey                = l_clskey
  receiving
    cifref                = lo_clif_incl_naming
  exceptions
    no_objecttype         = 1
    internal_error        = 2
    others                 = 3.

if sy-subrc = 0.
  lo_class_incl_naming ?= lo_clif_incl_naming.
  l_t_method_w_include =
    lo_class_incl_naming->get_all_method_includes( ).
  loop at l_t_method_w_include assigning <s_method_w_include>.
    <s_obj>-object = 'METH'.
    <s_obj>-incl_name = <s_method_w_include>-incname.
    <s_obj>-obj_name = l_clskey.
    <s_obj>-meth_name = <s_method_w_include>-cpdkey+30.
    append <s_obj> to gt_object.
  endloop.
  "ADÜ - Start insert 20170621
  "Public Section
  <s_obj>-object      = 'METH'.
  <s_obj>-incl_name = lo_class_incl_naming->public_section.
  <s_obj>-obj_name   = l_clskey.
  <s_obj>-meth_name = 'PUBLIC SECTION'.
  append <s_obj> to gt_object.
  "Private Section
  <s_obj>-object      = 'METH'.
  <s_obj>-incl_name = lo_class_incl_naming->private_section.
  <s_obj>-obj_name   = l_clskey.
  <s_obj>-meth_name = 'PRIVATE SECTION'.
  append <s_obj> to gt_object.
  "Protected Section
  <s_obj>-object      = 'METH'.
  <s_obj>-incl_name = lo_class_incl_naming->protected_section.

```

```

        <s_obj>-obj_name = l_clskey.
        <s_obj>-meth_name = 'PROTECTED SECTION'.
        append <s_obj> to gt_object.
        "Local Implementations
        <s_obj>-object = 'METH'.
        <s_obj>-incl_name = lo_class_incl_naming->locals_imp.
        <s_obj>-obj_name = l_clskey.
        <s_obj>-meth_name = 'LOCAL IMPLEMENTATIONS'.
        append <s_obj> to gt_object.
        "ADü - End insert 20170621
    endif.
* ts 28.10.2014 - End insert

endloop.

endform.                                "get_clas

*&-----*
*&      Form  scan_include_code
*&-----*
*      text
*-----*
form scan_include_code
    tables lt_string
        using i_skip_comment i_case i_regex.

data:
    lt_object      type standard table of s_object,
    ls_object      type s_object, "TS, 20140509
    lt_include     type standard table of tadir-obj_name,
    lv_program     type sy-repid,
    lt_abap        type abaptxt255_tab,
    lt_match_results type match_result_tab,
    ls_match_result like line of lt_match_results,
    ls_result      type s_result,
    lv_func        type rs381-name,
    lv_func_incl   type rs381-name.

field-symbols:
    <string> type s_string,
    <s_range> type rsrange,
    <obj> type s_object,
    <include> type tadir-obj_name,
    <abap> like line of lt_abap.

*-----*
"Get includes
clear lt_object.
"for reports and classes we already have the includes
loop at gt_object assigning <obj> where object = 'FUGR'.
    clear lt_include.
    lv_program = <obj>-incl_name.

    call function 'RS_GET_ALL_INCLUDES'
        exporting
            program      = lv_program
        tables
            includetab   = lt_include
        exceptions
            not_existent = 1
            no_program   = 2
            others       = 3.

    check sy-subrc is initial.

    loop at lt_include assigning <include>.
        ls_object-object = 'FUNC'.
        ls_object-incl_name = <include>.
        append ls_object to gt_object.
    endloop.

endloop.

sort lt_object.
delete adjacent duplicates from lt_object.

```

```

append lines of lt_object to gt_object.

"Source scan
loop at gt_object assigning <obj>.

    if <obj>-object = 'PROG'.
        <obj>-incl_name = <obj>-obj_name.
    endif.
    read report <obj>-incl_name into lt_abap.
    if sy-subrc is not initial.
        continue.
    endif.

    loop at lt_string assigning <string>.

        clear lt_match_results.
        if i_case = '' and i_regex = ''.
            find all occurrences of <string> in table lt_abap
            in character mode
            ignoring case
            results lt_match_results.
        elseif i_case = 'X' and i_regex = ''.
            find all occurrences of <string> in table lt_abap
            in character mode
            respecting case
            results lt_match_results.
        elseif i_case = '' and i_regex = 'X'.
            find all occurrences of regex <string> in table lt_abap
            in character mode
            ignoring case
            results lt_match_results.
        elseif i_case = 'X' and i_regex = 'X'.
            find all occurrences of regex <string> in table lt_abap
            in character mode
            respecting case
            results lt_match_results.
        endif.

        loop at lt_match_results into ls_match_result.
            read table lt_abap
            index ls_match_result-line assigning <abap>.
            if sy-subrc = 0.
                clear ls_result.
                ls_result-codetp = <obj>-object(2).
                "if FUNC: Get name of FM
                if <obj>-object = 'FUNC' or <obj>-object = 'FUGR'.
                    clear lv_func.

                    call function 'FUNCTION_INCLUDE_INFO'
                    "IMPORTING
                    "FUNCTAB      =
                    "NAMESPACE    =
                    "PNAME        =
                    changing
                        funcname      = lv_func
                        "GROUP        =
                        include        = <obj>-incl_name
                    exceptions
                        function_not_exists = 1
                        include_not_exists  = 2
                        group_not_exists    = 3
                        no_selections       = 4
                        no_function_include = 5
                        others              = 6.
                    if sy-subrc = 0.
                        <obj>-obj_name = lv_func.
                    endif.

                endif.

            if <obj>-object = 'METH'.
                ls_result-txtlg = <obj>-meth_name.
            endif.
            ls_result-targetname = <obj>-obj_name.
            ls_result-sourcename = <obj>-incl_name.
            ls_result-line_no    = ls_match_result-line.
            ls_result-line       = <abap>.

```

```

        case ls_result-codetp.
            when 'PR'.
                ls_result-routinetype = 'RT16'.
            when 'ME'.
                ls_result-routinetype = 'RT17'.
            when 'FU'.
                ls_result-routinetype = 'RT18'.
            endcase.
        append ls_result to gt_result.
    endif.
endloop.

endloop.

endloop.

endform.                                "scan_include_code

*&-----*
*&      Form  get transformation text
*&-----*
*      text
*-----*
FORM get_txtlg USING      p_tranid TYPE rstrant-tranid
                        CHANGING p_txtlg TYPE rstrant-txtlg.

SELECT SINGLE txtlg INTO p_txtlg FROM rstrant
WHERE langu = sy-langu
      AND tranid = p_tranid
      AND objvers = 'A'.

ENDFORM.

```

8.2.6 Includes

9 Z_RFC_TRANSLATION

9.1 Comment

9.1.1 Objective

Write descriptions of entities to relevant tables of the SAP BW-System.

9.1.2 Product

Migration Booster, Translation Steward

9.2 Technical Documentation

9.2.1 General Information

System	A4H
Function Module	Z_RFC_TRANSLATION,
Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG
Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:10:05 PM

9.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
I_DELIMITER	LIKE	SONV-FLAG	" "		X	

9.2.3 Tables

Parameter		Associated Type	Opt.	Short text
T_VALUES	LIKE	TAB512		

9.2.4 Exceptions

Exception	Short text
RELEASE_1_3	
NOT_AUTHORIZED	

9.2.5 Source Code Function module

FUNCTION Z_RFC_TRANSLATION.

```

*****
**"Local Interface:
**  IMPORTING
**    VALUE(I_DELIMITER) LIKE  SONV-FLAG DEFAULT ''
**  TABLES
**    T_VALUES STRUCTURE  TAB512
**  EXCEPTIONS
**    RELEASE_1_3
**    NOT_AUTHORIZED
**"-----

field-symbols: <fs_values>      type tab512,
                <fieldname>     type string,
                <target_struc>   type any,
                <fieldvalue>     type any.

data:
  ls_tab          type tab512,
  lt_fieldnames   type standard table of string,
  lt_values       type standard table of string,
  lr_target_struc type ref to data,
  lv_tabname      type tabname,
  lv_value        type string,
  lv_fieldname    type string.

"get fields from first dataset
read table t_values[] into ls_tab index 1.
if sy-subrc <> 0.
  "empty table passed - not valid
  return.
endif.

loop at t_values[] assigning <fs_values>.

  clear: lv_value, lv_tabname.
  free: lt_fieldnames, lt_values.
  split <fs_values> at i_delimiter into table lt_fieldnames.

  "Get table
  loop at lt_fieldnames[] assigning <fieldname>.

    if <fieldname>(5) eq 'TABLE'.

      split <fieldname> at '===' into table lt_values.
      read table lt_values into lv_value index 2.
      if sy-subrc = 0.
        lv_tabname = lv_value.
        exit.
      endif.

    endif.
  endloop.

  "create table structure dynamically
  free lr_target_struc.
  create data lr_target_struc type (lv_tabname).
  assign lr_target_struc->* to <target_struc>.

  "Write fields
  loop at lt_fieldnames[] assigning <fieldname>.

```

```

clear: lv_value.
free: lt_values.

if <fieldname>(5) eq 'TABLE'.
  continue.
endif.

split <fieldname> at '===' into table lt_values.
read table lt_values into lv_fieldname index 1.
if sy-subrc <> 0.
  continue.
endif.
read table lt_values into lv_value index 2.
if sy-subrc <> 0.
  continue.
endif.

assign component lv_fieldname of structure <target_struct>
to <fieldvalue>.
if sy-subrc <> 0.
  write:/ 'FIELD ', <fieldname>, 'NOT FOUND'.
  continue.
endif.

<fieldvalue> = lv_value.

endloop.
*BEGIN New ADÜ_20250328
authority-check object 'S_TABU_NAM'
id 'ACTVT' field '02'
id 'TABLE' field lv_tabname.
if sy-subrc <> 0.
  raise not_authorized.
else.
*END New ADÜ_20250328
  modify (lv_tabname) from <target_struct>.
*BEGIN New ADÜ_20250328
endif.
*END New ADÜ_20250328

endloop.

call function 'DB_COMMIT'.

endfunction.

```

10 Z_RFC_ADSO_GETDTL_XML

10.1 Comment

10.1.1 Objective

Reading information regarding an ADSO.

[Supported since SAP BW Release 7.40]

10.1.2 Product

Migration Booster

10.2 Technical Documentation

10.2.1 General Information

System	A4H
Function Module	Z_RFC_ADSO_GETDTL_XML,
Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG
Last Change	8/14/2025

Timestamp of documentation	8/18/2025 2:10:11 PM
----------------------------	----------------------

10.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
I_ADSONM	TYPE	RSOADSONM			X	

10.2.3 Export Parameter

Parameter		Associated Type	Pass	Short text
E_TEXT	TYPE	RSOADSODESCR	X	
E_INFOAREA	TYPE	RSINFOAREA	X	
EV_ADISOFLAGS_XML	TYPE	STRING	X	
EV_OBJECT_XML	TYPE	STRING	X	
EV_DIMENSION_XML	TYPE	STRING	X	
EV_KEY_XML	TYPE	STRING	X	
EV_HASH_XML	TYPE	STRING	X	
EV_INDEX_XML	TYPE	STRING	X	
EV_PARTITION_XML	TYPE	STRING	X	
EV_CHA_CONST_XML	TYPE	STRING	X	
EV_VALIDITY_XML	TYPE	STRING	X	

10.2.4 Exceptions

Exception	Short text
RELEASE_1_2	
NOT_AUTHORIZED	

10.2.5 Source Code Function module

FUNCTION Z_RFC_ADSO_GETDTL_XML.

```

*****-----
**"Local Interface:
**  IMPORTING
**    VALUE(I_ADSONM) TYPE  RSOADSONM
**  EXPORTING
**    VALUE(E_TEXT) TYPE  RSOADSODESCR
**    VALUE(E_INFOAREA) TYPE  RSINFOAREA
**    VALUE(EV_ADISOFLAGS_XML) TYPE  STRING
**    VALUE(EV_OBJECT_XML) TYPE  STRING
**    VALUE(EV_DIMENSION_XML) TYPE  STRING
**    VALUE(EV_KEY_XML) TYPE  STRING
**    VALUE(EV_HASH_XML) TYPE  STRING
**    VALUE(EV_INDEX_XML) TYPE  STRING
**    VALUE(EV_PARTITION_XML) TYPE  STRING
**    VALUE(EV_CHA_CONST_XML) TYPE  STRING
**    VALUE(EV_VALIDITY_XML) TYPE  STRING
**  EXCEPTIONS
**    RELEASE_1_2
**    NOT_AUTHORIZED
*****-----

data: lt_parameters type abap_parmbind_tab.
data: ls_parameter type abap_parmbind.
data: lo_ref type ref to data.
*****20210422 ADü_Begin Bug DP-5283
data: objvers_a type rsobjvers value 'A'.
*****20210422 ADü_End

field-symbols: <any> type any.

" get method parameter types
select
sconame,
type
from seosubcodf
into table @data(lt_method_param_types)

```

```

where clsname = 'CL_RSO_ADSO_API'
  and cmpname = 'READ'
  and type <> ''
order by editororder.

loop at lt_method_param_types assigning field-symbol(<ls_method_param_type>).
  try.
    create data lo_ref type (<ls_method_param_type>-type).
    catch cx_sy_create_data_error.
      " probably type of class
      data(lv_type) = |CL_RSO_ADSO_API=>{ <ls_method_param_type>-type }|.
      try.
        create data lo_ref type (lv_type).
        catch cx_sy_create_data_error..
          " no plan c
          exit.
        endtry.
      endtry.
    endtry.

  case <ls_method_param_type>-sconame+0(1).
    when 'E'.
      data(lv_kind) = cl_abap_objectdescr=>importing.
    when 'I'.
      lv_kind = cl_abap_objectdescr=>exporting.
    when others.
      exit.
    endcase.

    ls_parameter = value abap_parmbind(
      name = <ls_method_param_type>-sconame
      kind = lv_kind
      value = lo_ref ).
    *****20210422 ADÜ_Begin Bug DP-5283
    if ls_parameter-name = 'I_OBJVERS'.
      assign lo_ref->* to <any>.
      <any> = objvers_a.
    endif.
    *****20210422 ADÜ_End
    insert ls_parameter into table lt_parameters.
    free: ls_parameter.
    *****20210422 ADÜ_Begin Bug DP-5283
    unassign <any>.
    *****20210422 ADÜ_End

  endloop.

read table lt_parameters assigning field-symbol(<ls_parameter>) with key name = 'I_ADSO_NM'.
if sy-subrc = 0.
  assign <ls_parameter>-value->* to <any>.
  <any> = i_adsonm.
endif.

try.
*BEGIN New ADÜ_20250328
  authority-check object 'S_RS_ADSO'
    id 'RSINFOAREA' field '*'
    id 'RSOADSONM' field '*'
    id 'RSOADSOPAR' field '*'
    id 'ACTVT' field '03'.
  if sy-subrc <> 0.
    raise not_authorized.
  else.
*END New ADÜ_20250328
    call method ('CL_RSO_ADSO_API')=>read parameter-table lt_parameters.
*BEGIN New ADÜ_20250328
  endif.
*END New ADÜ_20250328
  catch cx_root. " catch all errors and exit -> no exceptions defined
  return.
endtry.

" Transform to XML for Export
loop at lt_parameters assigning <ls_parameter>.

  assign <ls_parameter>-value->* to <any>.
  " For easier consumption in the create Module switch the naming in XML to I_
  " for the importing Parameters of the create method
  if <ls_parameter>-name+0(1) = 'E'.

```

```

        data(lv_name) = |I{ <ls_parameter>-name+1(29) }|.
    else.
        lv_name = <ls_parameter>-name.
    endif.

    data(lt_srcbind) = value abap_trans_srcbind_tab( (
        name = lv_name
        value = <ls_parameter>-value ) ).

    case <ls_parameter>-name.
        when 'E_TEXT'.
            e_text = <any>.
        when 'E_INFOAREA'.
            e_infoarea = <any>..
        when 'E_S_ADSOFLAGS'.
            call transformation ( `ID` ) source (lt_srcbind) result xml ev_adsoflags_xml.
        when 'E_T_OBJECT'.
            call transformation ( `ID` ) source (lt_srcbind) result xml ev_object_xml.
        when 'E_T_DIMENSION'.
            call transformation ( `ID` ) source (lt_srcbind) result xml ev_dimension_xml.
        when 'E_T_KEY'.
            call transformation ( `ID` ) source (lt_srcbind) result xml ev_key_xml.
        when 'E_T_HASH'.
            call transformation ( `ID` ) source (lt_srcbind) result xml ev_hash_xml.
        when 'E_T_INDEX'.
            call transformation ( `ID` ) source (lt_srcbind) result xml ev_index_xml.
        when 'E_T_PARTITION'.
            call transformation ( `ID` ) source (lt_srcbind) result xml ev_partition_xml.
        when 'E_T_CHA_CONST'.
            call transformation ( `ID` ) source (lt_srcbind) result xml ev_cha_const_xml.
        when 'E_T_VALIDITY'.
            call transformation ( `ID` ) source (lt_srcbind) result xml ev_validity_xml.
    endcase.
endloop.
endfunction.

```

11 Z RFC_ADSO_CREATE_XML

11.1 Comment

11.1.1 Objective

Creating and activating ADSOs in an SAP BW-System.

[Supported since SAP BW Release 7.40]

11.1.2 Product

Migration Booster

11.2 Technical Documentation

11.2.1 General Information

System	A4H
Function Module	Z RFC_ADSO_CREATE_XML,
Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG
Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:10:18 PM

11.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
I_ADSO_NM	TYPE	RSOADS_ONM			X	
I_TEXT	TYPE	RSOADS_ODESCR		X	X	
I_INFOAREA	TYPE	RSINFOAREA		X	X	

Parameter		Associated Type	Default	Opt.	Pass	Short text
IV_ADSOFLAGS	TYPE	STRING			X	
IV_OBJECT	TYPE	STRING			X	
IV_DIMENSION	TYPE	STRING			X	
IV_KEY	TYPE	STRING			X	
IV_HASH	TYPE	STRING			X	
IV_INDEX	TYPE	STRING			X	
IV_PARTITION	TYPE	STRING			X	
IV_CHA_CONST	TYPE	STRING			X	
IV_VALIDITY	TYPE	STRING			X	

11.2.3 Exceptions

Exception	Short text
NOT_ACTIVE	
NOT_CREATED	
RELEASE_1_3	
NOT_AUTHORIZED	

11.2.4 Source Code Function module

FUNCTION Z_RFC_ADSO_CREATE_XML.

```

*-----
**"Local Interface:
**  IMPORTING
**    VALUE(I_ADSO) TYPE  RSOADSONM
**    VALUE(I_TEXT) TYPE  RSOADSODESCR OPTIONAL
**    VALUE(I_INFOAREA) TYPE  RSINFOAREA OPTIONAL
**    VALUE(IV_ADSOFLAGS) TYPE  STRING
**    VALUE(IV_OBJECT) TYPE  STRING
**    VALUE(IV_DIMENSION) TYPE  STRING
**    VALUE(IV_KEY) TYPE  STRING
**    VALUE(IV_HASH) TYPE  STRING
**    VALUE(IV_INDEX) TYPE  STRING
**    VALUE(IV_PARTITION) TYPE  STRING
**    VALUE(IV_CHA_CONST) TYPE  STRING
**    VALUE(IV_VALIDITY) TYPE  STRING
**  EXCEPTIONS
**    NOT_ACTIVE
**    NOT_CREATED
**    RELEASE_1_3
**    NOT_AUTHORIZED
**
DATA: lt_resbind      TYPE abap_trans_resbind_tab.
DATA: ls_resbind      TYPE abap_trans_resbind.
DATA: lt_parameters   TYPE abap_parmbind_tab.
DATA: ls_parameter    TYPE abap_parmbind.
DATA: lo_ref          TYPE REF TO data.

FIELD-SYMBOLS: <any> TYPE any.

" get method parameter types
SELECT
  sconame,
  type
FROM seosubcodf
INTO TABLE @DATA(lt_method_param_types)
WHERE clsname = 'CL_RSO_ADSO_API'
  AND cmpname = 'CREATE'
  AND type <> ''
ORDER BY editorder.

LOOP AT lt_method_param_types ASSIGNING FIELD-SYMBOL(<ls_method_param_type>).
  FREE: ls_parameter, ls_resbind, lt_resbind.

  TRY. " create data reference for parameter
      CREATE DATA lo_ref TYPE (<ls_method_param_type>-type).
    CATCH cx_sy_create_data_error.
      " probably type of class
      DATA(lv_type) = |CL_RSO_ADSO_API=>{ <ls_method_param_type>-type }|.
      TRY.

```

```

        CREATE DATA lo_ref TYPE (lv_type).
        CATCH cx_sy_create_data_error..
            " no plan c
            EXIT.
        ENDTRY.
    ENDTRY.

ls_resbind = VALUE abap_trans_resbind(
    name = <ls_method_param_type>-sconame
    value = lo_ref ).
INSERT ls_resbind INTO TABLE lt_resbind.

" determine kind of parameter for dynamic method call
CASE <ls_method_param_type>-sconame+0(1).
    WHEN 'E'.
        DATA(lv_kind) = cl_abap_objectdescr=>importing.
    WHEN 'I'.
        lv_kind = cl_abap_objectdescr=>exporting.
    WHEN OTHERS.
        EXIT.
ENDCASE.

CASE <ls_method_param_type>-sconame.
    WHEN 'I_ADSONM'.
        GET REFERENCE OF i_adsonm INTO DATA(lo_adsonmref).
        ls_parameter = VALUE abap_parmbind(
            name = <ls_method_param_type>-sconame
            kind = lv_kind
            value = lo_adsonmref ).
        INSERT ls_parameter INTO TABLE lt_parameters.
    WHEN 'I_TEXT'.
        GET REFERENCE OF i_text INTO DATA(lo_text).
        ls_parameter = VALUE abap_parmbind(
            name = <ls_method_param_type>-sconame
            kind = lv_kind
            value = lo_text ).
        INSERT ls_parameter INTO TABLE lt_parameters.
        CONTINUE.
    WHEN 'I_INFOAREA'.
        GET REFERENCE OF i_infoarea INTO DATA(lo_infoarea).
        ls_parameter = VALUE abap_parmbind(
            name = <ls_method_param_type>-sconame
            kind = lv_kind
            value = lo_infoarea ).
        INSERT ls_parameter INTO TABLE lt_parameters.
        CONTINUE.
    WHEN 'I_S_ADISOFLAGS'.
        CALL TRANSFORMATION ('ID') SOURCE XML iv_adsoflags RESULT (lt_resbind).
    WHEN 'I_T_OBJECT'.
        CALL TRANSFORMATION ('ID') SOURCE XML iv_object RESULT (lt_resbind).
    WHEN 'I_T_DIMENSION'.
        CALL TRANSFORMATION ('ID') SOURCE XML iv_dimension RESULT (lt_resbind).
    WHEN 'I_T_KEY'.
        CALL TRANSFORMATION ('ID') SOURCE XML iv_key RESULT (lt_resbind).
    WHEN 'I_T_HASH'.
        CALL TRANSFORMATION ('ID') SOURCE XML iv_hash RESULT (lt_resbind).
    WHEN 'I_T_INDEX'.
        CALL TRANSFORMATION ('ID') SOURCE XML iv_index RESULT (lt_resbind).
    WHEN 'I_T_PARTITION'.
        CALL TRANSFORMATION ('ID') SOURCE XML iv_partition RESULT (lt_resbind).
    WHEN 'I_T_CHA_CONST'.
        CALL TRANSFORMATION ('ID') SOURCE XML iv_cha_const RESULT (lt_resbind).
    WHEN 'I_T_VALIDITY'.
        CALL TRANSFORMATION ('ID') SOURCE XML iv_validity RESULT (lt_resbind).
ENDCASE.

" build parameter table for method call
ls_parameter = VALUE abap_parmbind(
    name = <ls_method_param_type>-sconame
    kind = lv_kind
    value = lt_resbind[ 1 ]-value ).
INSERT ls_parameter INTO TABLE lt_parameters.

ENDLOOP.

TRY.
    *BEGIN New ADü_20250328
    authority-check object 'S_RS_ADISO'

```

```

        id 'RSINFOAREA' field ''
        id 'RSOADSONM' field '*'
        id 'RSOADSOPAR' field '*'
        id 'ACTVT' field '03'.
        if sy-subrc <> 0.
            raise not_authorized.
        else.
*END New ADü_20250328
        CALL METHOD ('CL_RSO_ADSO_API')=>create PARAMETER-TABLE lt_parameters.
*BEGIN New ADü_20250328
        endif.
*END New ADü_20250328
        CATCH cx_root. " catch all errors and exit
        ENDTRY.

        "check if object is created, active and dequeue it
        DATA: lv_exists TYPE i.
        SELECT COUNT( * )
        FROM rsoadso "#EC CI_BYPASS
        INTO lv_exists
        WHERE adsonm = i_adsonm
        AND ( objvers = 'A'
        OR objvers = 'M' ).

        IF NOT lv_exists >= 1.
            RAISE not_created.
        ENDIF.

        DATA: lo_adso TYPE REF TO cl_rso_adso.

        CALL METHOD cl_rso_adso=>factory
        EXPORTING
            i_adsonm = i_adsonm
        RECEIVING
            r_r_adso = lo_adso.

        "dequeue
        CALL METHOD lo_adso->if_rso_tlogo_maintain~dequeue.

        "is active
        DATA: is_active TYPE rs_bool.

        CALL METHOD lo_adso->if_rso_tlogo_maintain~is_active
        RECEIVING
            r_is_active = is_active.

        IF is_active = rs_c_false.
            RAISE not_active.
        ENDIF.
    ENDFUNCTION.

```

12 Z_RFC_FUNCTION_DELETE

12.1 Comment

12.1.1 Objective

Ability to delete Function Modules in an SAP BW-System.
Required to update outdated Function Modules.

12.1.2 Product

General

12.2 Technical Documentation

12.2.1 General Information

System	A4H
Function Module	Z_RFC_FUNCTION_DELETE,
Function Pool	Z_PS_FG
Remote	Yes

Last Changed By	AKOLMOG
Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:10:26 PM

12.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
FUNCNAME	TYPE	FUNCNAME			X	

12.2.3 Exceptions

Exception	Short text
ERROR_MESSAGE	
RELEASE_1_1	
NOT_AUTHORIZED	
NOT_ALLOWED_TO_DELETE	

12.2.4 Source Code Function module

```

FUNCTION Z_RFC_FUNCTION_DELETE.
"-----
***Local Interface:
"   IMPORTING
"       VALUE (FUNCNAME) TYPE  FUNCNAME
"   EXCEPTIONS
"       ERROR_MESSAGE
"       RELEASE_1_1
"       NOT_AUTHORIZED
"       NOT_ALLOWED_TO_DELETE
"-----
*BEGIN New ADü_20250328
  authority-check object 'S_DEVELOP' "#EC AUTFLD_MIS
    id 'DEVCLASS' field '*'
    id 'OBJTYPE' field 'FUNC'
    id 'OBJNAME' field funcname
    id 'ACTVT' field '06'.
  if sy-subrc <> 0.
    raise not_authorized.
  endif.
  " Check if FM which should be deleted is in the same FUGR as this FM
  data: lv_pnameOfFuncname type pname.
  " get pname of FM which should be delete
  select single pname from tfdir into lv_pnameOfFuncname where funcname = funcname.
  " compare pnames
  if lv_pnameOfFuncname <> sy-repid.
    raise not_allowed_to_delete.
  endif.
*END New ADü_20250328

  call function 'FUNCTION_DELETE'
    exporting
      funcname      = funcname
    exceptions
      error_message = 1
      others        = 2.
  if sy-subrc = 1.
    raise error_message.
  endif.
endfunction.

```

13 Z_RFC_CHECK_ACT_TR

13.1 Comment

13.1.1 Objective

Check, activate and assign transports of all entity types for our product **Translation Steward**.

13.1.2 Product**Translation Steward****13.2 Technical Documentation****13.2.1 General Information**

System	A4H
Function Module	Z_RFC_CHECK_ACT_TR,
Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG
Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:10:30 PM

13.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
IV_MODE	TYPE	CHAR1	'1'		X	
IV_XML_OBJECTS	TYPE	STRING			X	
IV_TRANSPORT	TYPE	TRKORR		X	X	

13.2.3 Export Parameter

Parameter		Associated Type	Pass	Short text
EV_XML_STATUS	TYPE	STRING	X	

13.2.4 Exceptions

Exception	Short text
RELEASE_1_2	
NOT_AUTHORIZED	

13.2.5 Source Code Function module

FUNCTION Z_RFC_CHECK_ACT_TR.

```

*-----
*""Local Interface:
*  IMPORTING
*    VALUE(IV_MODE) TYPE  CHAR1 DEFAULT '1'
*    VALUE(IV_XML_OBJECTS) TYPE  STRING
*    VALUE(IV_TRANSPORT) TYPE  TRKORR OPTIONAL
*  EXPORTING
*    VALUE(EV_XML_STATUS) TYPE  STRING
*  EXCEPTIONS
*    RELEASE_1_2
*    NOT_AUTHORIZED
*-----

types:
  begin of ty_message,
    message_type type sy-msgty,
    message      type string,
  end of ty_message.
types: tt_messages type standard table of ty_message with default key.

types:
  begin of ty_status,
    tlogo                type rstlogo,
    objnm                type sobj_name,
    actflg               type rsawbn_obj_activflg,
    objstat              type rsobjstat,
    saved                type rs_bool,
    activflg             type rsawbn_obj_activflg,
    locked               type rs_bool,
    tr_lock              type rs_bool,
    no_handler           type rs_bool,
    written_to_tr        type rs_bool,
    activation_successful type rs_bool,
    messages             type tt_messages,
  end of ty_status.

```

```

data: lt_objects          type rso_t_tlogo.
data: lt_status           type table of ty_status.
data: lv_objvers          type rs_objvers.
data: lo_tlogo            type ref to if_rso_tlogo.
data: lv_objstat          type rsobjstat.
data: lo_msg              type ref to cl_rso_msg.
data: lt_object_details   type rso_t_cto_tlogo_prop.
data: lv_request          type trkorr.
data: lv_error            type rs_bool.
data: lt_tr_msg           type bapirettab.
data: lo_awbn_factory      type ref to cl_rsawbn_obj_factory.
data: lo_awbn_object       type ref to cl_rsawbn_obj.
data: lt_enq              type table of seqg3.
data: lv_garg             type eqegraarg.
data: lv_logsys           type rsrlogsys.
data: ls_tlogoprop        type rstlogoprop.
data: lt_msg              type rs_t_msg.
data: lv_repaired         type rs_bool.
data: lv_subrc            type sysubrc.

field-symbols: <ls_enq>   type seqg3.
field-symbols: <message>  type ty_message.
field-symbols: <ls_status> type ty_status.
field-symbols: <ls_object> type rso_s_tlogo.
field-symbols: <ls_msg>    type rs_s_msg.
field-symbols: <ls_tr_obj> type trexreqob.

call transformation (`ID`) source xml iv_xml_objects result xml_objects = lt_objects.

" only accept 1 - check / 2 - activate / 3 - transport mode
assert iv_mode = 1 or iv_mode = 2 or iv_mode = 3.

if lt_objects is initial.
    return. " no objects -> nothing to do
endif.

" create AWBN Factory for later usage
create object lo_awbn_factory.

loop at lt_objects assigning <ls_object>.

    " Prepare Status Output
    append initial line to lt_status assigning <ls_status>.

    " Fill general fields
    <ls_status>-tlogo = <ls_object>-tlogo.
    <ls_status>-objnm = <ls_object>-objnm.

*BEGIN New ADü_20250328
    authority-check object 'S_RS_ADMWB'
    id 'RSADMWBOBJ' field '*'
    id 'ACTVT' field '03'.
    if sy-subrc <> 0.
        raise not_authorized.
    else.
*BEND New ADü_20250328
    " get logsys
    lv_logsys = cl_rso_repository=>get_logical_system_self( ).
*BEGIN New ADü_20250328
    endif.
*BEND New ADü_20250328
    " get tlogoprops
    cl_rso_repository=>get_tlogoprop( exporting i_tlogo = <ls_object>-tlogo importing
e_s_tlogoprop = ls_tlogoprop ).

    try .
        " Check A Version
        lv_objvers = 'A'.
        call method cl_rso_object_collection_cache=>get_object_properties
            exporting
                i_tlogo          = <ls_object>-tlogo
                i_objnm          = <ls_object>-objnm
                i_objvers        = lv_objvers
                i_objlogsys      = lv_logsys

```

```

        i_bypassing_buffer = abap_true
        i_no_message       = abap_true
    importing
        e_objstat          = lv_objstat
    exceptions
        object_not_found   = 1.
catch cx_root.
    " Check M Version
    lv_objvers = 'M'.
    call method cl_rso_object_collection_cache=>get_object_properties
    exporting
        i_tlogo           = <ls_object>-tlogo
        i_objnm           = <ls_object>-objnm
        i_objvers         = lv_objvers
        i_objlogsys       = lv_logsys
        i_bypassing_buffer = abap_true
        i_no_message      = abap_true
    importing
        e_objstat         = lv_objstat
    exceptions
        object_not_found   = 1.
endtry.

" fill Object Status
<ls_status>-objstat = lv_objstat.

" Check if tlogo class is available
if ls_tlogoprop-class is not initial.
    call method (ls_tlogoprop-class)=>if_rso_tlogo~get_instance
    exporting
        i_objnm = <ls_object>-objnm
    receiving
        r_r_tlogo = lo_tlogo.
else.
    <ls_status>-no_handler = abap_true.
endif.

" saved?
if lo_tlogo is bound.
    <ls_status>-saved = lo_tlogo->if_rso_tlogo_maintain~is_saved( ).
endif.

if iv_mode = '1'. " Check Mode

    " Check for M / A Version
    lo_awbn_factory->get_instance(
    exporting
        i_tlogo           = <ls_object>-tlogo
        i_objnm           = <ls_object>-objnm
    receiving
        re_r_awbobj       = lo_awbn_object ).

    <ls_status>-activflg = lo_awbn_object->get_activflg( ). " U -> M <> A Version / X -> M = A
Version / ' ' no information
    "Special case Analysis Auth.
    if <ls_object>-tlogo = 'ASOB'.
        data: lt_objstat type table of rsobjstat.
        select objstat
        from rsecbiau
        into table lt_objstat
        where auth = <ls_object>-objnm.

        find 'INA' in table lt_objstat.                                "#EC CI_NOORDER

        if sy-subrc = 0.
            <ls_status>-activflg = 'U'.
        endif.
        "special case 7.x InfoSources
    elseif <ls_object>-tlogo = 'TRCS'.
        data: lv_timestmp_a type rstimestamp,
              lv_timestmp_m type rstimestamp.
        select single timestamp
        from rksnew
        into lv_timestmp_a
        where isource = <ls_object>-objnm
        and objvers = 'A'.

        select single timestamp

```

```

        from rsksnw
        into lv_tstmp_m
        where isource = <ls_object>-objnm
        and objvers = 'M'.

    if lv_tstmp_m > lv_tstmp_a.
        <ls_status>-activflg = 'U'.
    endif.
endif.

" check for lock
call function 'ENQUEUE_READ'
    exporting
        gclient = sy-mandt
        guname = ''
    tables
        enq      = lt_enq.

concatenate <ls_object>-objnm '*' into lv_garg.

loop at lt_enq assigning <ls_enq> where garg cp lv_garg. " AND gobj = ls_tlogoprop-
enqobject.
endloop.
if sy-subrc = 0.
    <ls_status>-locked = abap_true.
endif.

" transport locks
data: ls_e071      type e071.
data: lv_result    type trpari-s_checked.
data: ls_tadir     type tadir.
data: ls_lock_key  type tlock_int.
data: lt_tlock     type table of tlock.
data: lv_lockflag  type trpari-s_lockflag.

ls_e071-pgmid      = 'R3TR'.
ls_e071-object     = <ls_object>-tlogo.
ls_e071-obj_name   = <ls_object>-objnm.

call function 'TR_CHECK_TYPE'
    exporting
        wi_e071      = ls_e071
    importing
        pe_result    = lv_result
        we_lock_key  = ls_lock_key
        we_tadir     = ls_tadir.

" determine lock key for complete object (TADIR)
if ls_e071-pgmid    = ls_tadir-pgmid
and ls_e071-object  = ls_tadir-object
and ls_e071-obj_name = ls_tadir-obj_name.
    " object is already complete (TADIR object): lock key is already known
else.
    move-corresponding ls_tadir to ls_e071.
    call function 'TR_CHECK_TYPE'
        exporting
            wi_e071      = ls_e071
        importing
            pe_result    = lv_result
            we_lock_key  = ls_lock_key
            we_tadir     = ls_tadir.
endif.

" proceed only for lockable objects
if lv_result = 'L'.
    " determine locks referring to complete (TADIR) object
    call function 'TRINT_CHECK_LOCKS'
        exporting
            wi_lock_key = ls_lock_key
        importing
            we_lockflag = lv_lockflag
        tables
            wt_tlock    = lt_tlock
        exceptions
            empty_key   = 1
            others      = 2 ##FM_SUBRC_OK.
    if lv_lockflag = abap_true.
        <ls_status>-tr_lock = abap_true.
    else.

```

```

        <ls_status>-tr_lock = abap_false.
      endif.

    else.
      <ls_status>-tr_lock = abap_false.
    endif.

    if lo_tlogo is bound.
      lo_tlogo->if_rso_tlogo_maintain~check(
        exporting
          i_objvers          = lv_objvers
        importing
          e_r_msg            = lo_msg
          e_is_repaired      = lv_repaired
          e_subrc            = lv_subrc ).

      if lo_msg is bound.
        lt_msg = lo_msg->get_all_msg( ).

        loop at lt_msg assigning <ls_msg>.
          append initial line to <ls_status>-messages assigning <message>.
          <message>-message_type = <ls_msg>-msgty.
          message id <ls_msg>-msgid type <ls_msg>-msgty number <ls_msg>-msgno
            with <ls_msg>-msgv1 <ls_msg>-msgv2 <ls_msg>-msgv3 <ls_msg>-msgv4
              into <message>-message.
        endloop.
      endif.
    endif.

  endif.

  if iv_mode = '2'. " Activation Mode

    if lo_tlogo is not bound.
      exit.
    endif.

    " Run Check before activation.
    if lo_tlogo is bound.
      lo_tlogo->if_rso_tlogo_maintain~check(
        exporting
          i_objvers          = lv_objvers
        importing
          e_r_msg            = lo_msg
          e_is_repaired      = lv_repaired
          e_subrc            = lv_subrc ).
      if lo_msg is bound.
        lt_msg = lo_msg->get_all_msg( ).

        delete lt_msg where not ( msgty eq 'E' or msgty eq 'W' ). " only look for errors and
warnings
      if lt_msg is not initial. " Error messages found? return messages - skip activation
        loop at lt_msg assigning <ls_msg>.
          append initial line to <ls_status>-messages assigning <message>.
          <message>-message_type = <ls_msg>-msgty.
          message id <ls_msg>-msgid type <ls_msg>-msgty number <ls_msg>-msgno
            with <ls_msg>-msgv1 <ls_msg>-msgv2 <ls_msg>-msgv3 <ls_msg>-msgv4
              into <message>-message.
        endloop.
      endif.
    endif.
  endif.
  try.
    lo_tlogo->if_rso_tlogo_maintain~prepare(
      exporting
        i_with_enqueue      = rs_c_false    " = 'X': with Enqueue Lock
        i_with_cto_check    = rs_c_false    " = 'X': with CTO Check
        i_with_authority    = rs_c_false    " = 'X': with Authorization
        i_with_rebuild      = rs_c_false    " = 'X': Database Must Reread the Object
        i_actvt             = rssb_c_auth_actvt-maintain " Activity
        i_objvers           = rs_c_objvers-active " Object version
        i_detlevel          = '3'          " Application Log: Level of Detail
      ).
    catch cx_rs_cancelled.
    catch cx_rs_not_authorized.
    catch cx_rs_display_only.
  endtry.

```

```

lo_tlogo->if_rso_tlogo_maintain~activate(
  exporting
    i_objvers          = lv_objvers    " Object Version (A / M)
    i_force_activation  = rs_c_true     " = 'X': activate in case the object is already
active
    i_show_check_protocol = rs_c_false  " = 'X': Display Consistency Log as Popup if
Warnings Arise
    i_with_cto          = rs_c_false
  importing
    e_subrc            = lv_subrc ).

if lv_subrc = 0.
  <ls_status>-activation_successful = abap_true.
  append initial line to <ls_status>-messages assigning <message>.
  <message>-message_type = 'S'.
  <message>-message = 'Activation successful' ##NO_TEXT.

else.
  <ls_status>-activation_successful = abap_false.
  append initial line to <ls_status>-messages assigning <message>.
  <message>-message_type = 'E'.
  <message>-message = 'Activation failed' ##NO_TEXT.
endif.

endif.

if iv_mode = '3'. " Transport Mode
  check iv_transport is not initial.

  data: ls_wi_e071 like e071.
  ls_wi_e071-pgmid = 'R3TR'.
  ls_wi_e071-object = <ls_object>-tlogo.
  ls_wi_e071-obj_name = <ls_object>-objnm.

  call function 'TR_APPEND_TO_COMM'
    exporting
      pi_korrnum          = iv_transport
      wi_e071             = ls_wi_e071
    exceptions
      no_authorization    = 1
      no_systemname       = 2
      no_systemtype       = 3
      tr_check_keysyntax_error = 4
      tr_check_obj_error  = 5
      tr_enqueue_failed   = 6
      tr_ill_korrnum       = 7
      tr_key_without_header = 8
      tr_lockmod_failed   = 9
      tr_lock_enqueue_failed = 10
      tr_modif_only_in_modif_order = 11
      tr_not_owner        = 12
      tr_no_append_of_corr_entry = 13
      tr_no_append_of_c_member = 14
      tr_no_shared_repairs = 15
      tr_order_not_exist   = 16
      tr_order_released    = 17
      tr_order_update_error = 18
      tr_repair_only_in_repair_order = 19
      tr_wrong_order_type  = 20
      wrong_client         = 21
      others               = 22.

  if sy-subrc <> 0.
    <ls_status>-written_to_tr = abap_false.
  else.
    <ls_status>-written_to_tr = abap_true.
  endif.
  free: ls_wi_e071.

endif.

endloop.

call transformation ('ID') source status = lt_status result xml ev_xml_status.

endfunction.

```

14 Z RFC_HCPR_CREATE

14.1 Comment

14.1.1 Objective

Create and activate HCPRs in the SAP BW-System.

14.1.2 Product

Migration Booster

14.2 Technical Documentation

14.2.1 General Information

System	A4H
Function Module	Z RFC_HCPR_CREATE,
Function Pool	Z_PS_FG
Remote	Yes
Last Changed By	AKOLMOG
Last Change	8/14/2025
Timestamp of documentation	8/18/2025 2:10:46 PM

14.2.2 Import Parameter

Parameter		Associated Type	Default	Opt.	Pass	Short text
I_HCPRNM	TYPE	RSOHCPRNM	"		X	
I_DESCRIPTION	TYPE	SSTRING	"	X	X	
I_INFOAREA	TYPE	RSINFOAREA	"		X	

14.2.3 Export Parameter

Parameter		Associated Type	Pass	Short text
E_SUCCESS	TYPE	RS_BOOL	X	

14.2.4 Tables

Parameter		Associated Type	Opt.	Short text
T_PARTPROVIDER	LIKE	RSD_S_PROV		

14.2.5 Exceptions

Exception	Short text
RELEASE_1_1	
NOT_AUTHORIZED	

14.2.6 Source Code Function module

```
FUNCTION Z RFC_HCPR_CREATE.
```

```

*-----
*""Local Interface:
*  IMPORTING
*    VALUE(I_HCPRNM) TYPE  RSOHCPRNM DEFAULT ''
*    VALUE(I_DESCRIPTION) TYPE  SSTRING DEFAULT ''
*    VALUE(I_INFOAREA) TYPE  RSINFOAREA DEFAULT ''
*  EXPORTING
*    VALUE(E_SUCCESS) TYPE  RS_BOOL
*  TABLES
*    T_PARTPROVIDER STRUCTURE  RSD_S_PROV
*  EXCEPTIONS
*    RELEASE_1_1
*    NOT_AUTHORIZED
*-----

```

```

data: lt_partprovider type table of rsinfoprov,
      ls_partprovider type rsinfoprov,
      ls_temp         type rsd_s_prov.

```

```
loop at t_partprovider into ls_temp.  
  ls_partprovider = ls_temp-infoprov.  
  append ls_partprovider to lt_partprovider.  
endloop.  
  
*BEGIN New ADü_20250328  
authority-check object 'S_RS_HCPR'  
  id 'RSINFOAREA' field i_infoarea  
  id 'RSHCPR' field i_hcprnm  
  id 'ACTVT' field '03'  
  id 'RSHCPROBJ' field '*'.  
if sy-subrc <> 0.  
  raise not_authorized.  
else.  
*END New ADü_20250328  
  call method cl_rso_hcpr_metadata_api=>create_union  
    exporting  
      i_hcprnm           = i_hcprnm  
      i_description      = i_description  
      i_infoarea         = i_infoarea  
      i_t_part_provider = lt_partprovider  
    importing  
      "e_r_msg           = E_R_MSG  
      e_success          = e_success.  
*BEGIN New ADü_20250328  
  endif.  
*END New ADü_20250328  
endfunction.
```