

### **CASE 3: Conversion (UPLOAD) Creation**

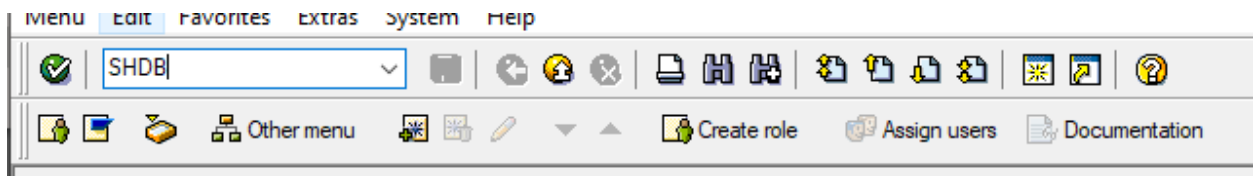
#### **Requirement**

Create an ABAP program that will let you upload and create Domains from an excel file.

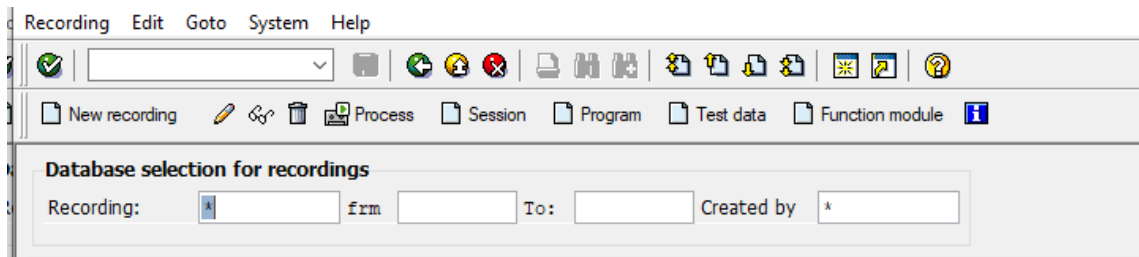
#### **Process**

1. Create an ABAP Program to read an excel file.
2. The ABAP program will mass create Domain in SE11 and save it a local package (\$TMP)
3. Fields in the excel must contain the following:
  - a. Domain Name
  - b. Description
  - c. Data Type
  - d. No. of Characters
  - e. Decimal Places
4. You may record using the SHDB transaction for the BDC (Batch Input Session)
5. In each loop in the records show a log after. You can use WRITE commands or show the report in other output methods.

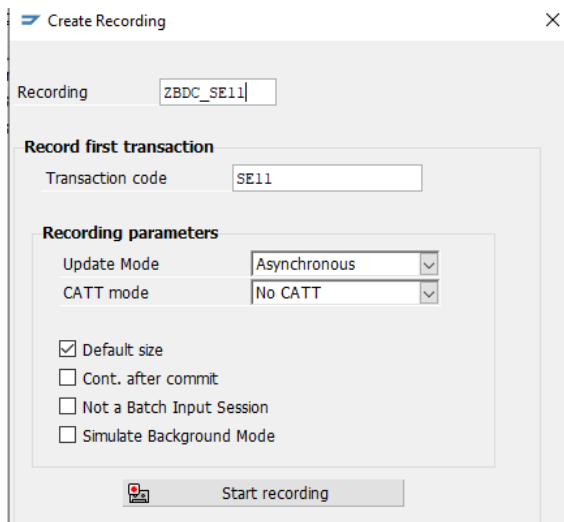
To Start BDC, go to SHDB for recording



Click on Recording



Put the recording name and the transaction code and hit start recording



Record how you declare Domain in SE11. Put a name and hit Create

The screenshot shows the 'Create Object' dialog box in SAP SE11. The 'Domain' radio button is selected. The text 'domainsample1' is entered in the domain name field. At the bottom, there are buttons for 'Display', 'Change', and 'Create'.

<input type="radio"/> Database table	
<input type="radio"/> View	
<input type="radio"/> Data type	
<input type="radio"/> Type Group	
<input checked="" type="radio"/> Domain	domainsample1
<input type="radio"/> Search help	
<input type="radio"/> Lock object	

Buttons: Display, Change, Create

Put a short description, Data type, No. Characters, Decimal Places, and output length.

The screenshot shows the 'Domain Sample 1' properties tab in SE11. The 'Format' section has 'Data Type' set to 'CHAR', 'No. Characters' set to '10', and 'Decimal Places' set to '0'. The 'Output Characteristics' section has 'Output Length' set to '10', 'Output' set to 'Normal', and 'Routine' set to an empty field. There are checkboxes for 'Sign' and 'Case-sensitive', both of which are unchecked.

Domain: DOMAINSAMPLE1 New(Revised)	
Short Description: Domain Sample 1	
Properties   Definition   Value Range	
<b>Format</b>	
Data Type	CHAR
No. Characters	10
Decimal Places	0
<b>Output Characteristics</b>	
Output Length	10
Output	Normal
Routine	
<input type="checkbox"/> Sign	
<input type="checkbox"/> Case-sensitive	

Save it on \$TMP Package. click SAVE icon

The screenshot shows the 'Create Object Directory Entry' dialog box in SE11. The 'Object' field contains 'R3TR DOMA ZDOMAINSAMPLE1'. The 'Attributes' section has 'Package' set to '\$TMP', 'Person Responsible' set to 'ABAPUSER01', 'Original System' set to 'SM2', 'Original language' set to 'EN English', and 'Created On' set to an empty field. At the bottom, there are icons for 'Local Object', 'Lock Overview', and a 'Save' icon.

Create Object Directory Entry	
Object: R3TR DOMA ZDOMAINSAMPLE1	
<b>Attributes</b>	
Package	\$TMP
Person Responsible	ABAPUSER01
Original System	SM2
Original language	EN English
Created On	















Buttons: Local Object, Lock Overview, Save











The Recording will appear at the database

Database selection for recordings					
Recording:	ZBC_SAMPLE	frm		To:	
Recording	CreatedBy	Date	Time	Transact.	Sc
ZBC_SAMPLE	ABAPUSER01	04/26/2023	17:40:30	1	10

From the toolbar you will click the "Program"

Recording Edit Goto System Help



             

 New recording     Process  Session  Program  Test data  Function module 

Database selection for recordings

Recording: ZBC\_SAMPLE frm To: Created by \*

Put the program name and select the "Transfer From Recording"

 Generate Program for Recording ZBC\_SAMPLE 

Program Name

**Field contents**



☐ Read from file

☒

**Test data**

☐ Create

File name

Put a title on the program. Click the "Source code"

The screenshot shows the 'ABAP: Program Attributes Z\_DOMAINSAMPLE1 Change' dialog box. It contains the following fields and options:

- Title: [Empty text field]
- Original language: [EN]
- Created: [ABAPUSER01] [04/26/2023]
- Last Changed: [Empty text field]
- Status: [New (Revised)]
- Attributes**
  - Type: [Executable program]
  - Status: [Unclassified]
  - Authorization Group: [Empty text field]
  - Application: [Empty dropdown menu]
  - LDB name: [Empty text field]
  - Selection screen: [Empty text field]
  - ABAP Language Version: [Standard ABAP (Unicode)]
  - ☒ Fixed point arithmetic
  - ☐ Editor lock
  - ☐ Start using variant

At the bottom left, there is a 'Source code' button with a magnifying glass icon.

Click on save

The screenshot shows the 'Create Object Directory Entry' dialog box. It contains the following fields and options:

- Object: [R3TR] [PROG] [Z\_DOMAINSAMPLE1]
- Attributes**
  - Package: [\$TMP]
  - Person Responsible: [ABAPUSER01]
  - Original System: [SM2]
  - Original language: [EN] English
  - Created On: [Empty text field]

At the bottom, there is a toolbar with the following icons and labels:

- Local Object (floppy disk icon)
- Lock Overview (lock icon)
- [Empty icon]
- [X icon]

It will redirect to the ABAP Editor with the code generated from the BDC Recording

```
1 report Z_DOMAINSAMPLE1
2     no standard page heading line-size 255.
3
4 * Include bdcrcxl_s:
5 * The call transaction using is called WITH AUTHORITY-CHECK!
6 * If you have own auth.-checks you can use include bdcrcxl instead.
7 include bdcrcxl_s.
8
9 start-of-selection.
10
11 perform open_group.
12
13 perform bdc_dynpro      using 'SAPLSD_ENTRY' '1000'.
14 perform bdc_field      using 'BDC_CURSOR'
15                             'RSRDI-DOMA_VAL'.
16 perform bdc_field      using 'BDC_OKCODE'
17                             '=WB_CREATE'.
18 perform bdc_field      using 'RSRDI-DOMA'
19                             'X'.
20 perform bdc_field      using 'RSRDI-DOMA_VAL'
21                             'ZDOMAINSAMPLE1'.
22 perform bdc_dynpro      using 'SAPLSD11' '1200'.
23 perform bdc_field      using 'BDC_OKCODE'
24                             '/00'.
25 perform bdc_field      using 'DD01D-DDTEXT'
26                             'Sample 1'.
27 perform bdc_field      using 'BDC_CURSOR'
28                             'DD01D-OUTPUTLEN'.
29 perform bdc_field      using 'DD01D-DATATYPE'
30                             'CHAR'.
31 perform bdc_field      using 'DD01D-LENG'
32                             ' 10'.
33 perform bdc_field      using 'DD01D-DECIMALS'
34                             ' 0'.
35 perform bdc_field      using 'DD01D-OUTPUTLEN'
36                             ' 10'.
37 perform bdc_dynpro      using 'SAPLSD11' '1200'.
38 perform bdc_field      using 'BDC_OKCODE'
39                             '=WB_SAVE'.
40 perform bdc_field      using 'DD01D-DDTEXT'
41                             'Sample 1'.
42 perform bdc_field      using 'BDC_CURSOR'
43                             'DD01D-OUTPUTLEN'.
44 perform bdc_field      using 'DD01D-DATATYPE'
45                             'CHAR'.
46 perform bdc_field      using 'DD01D-LENG'
47                             ' 10'.
48 perform bdc_field      using 'DD01D-OUTPUTLEN'
49                             ' 10'.
50 perform bdc_dynpro      using 'SAPLSTRD' '0100'.
51 perform bdc_field      using 'BDC_CURSOR'
52                             'K0007-L DEVCLASS'.
```

Declaring Truxs for Type-pools for our Conversion of XLS to Internal Table

```
7
8 TYPE-POOLS: truxs.
9
```

Declare the Data for our table that will be populated by excel that we will upload

```
⊞ TYPES: BEGIN OF t_datatab,
        col1(30) TYPE c,
        col2(30) TYPE c,
        col3(30) TYPE c,
        col4(30) TYPE c,
        col5(30) TYPE c,
        col6(480) TYPE c,
        END OF t_datatab.
```

Declare the internal table where data from excel will be saved

```
21  
22 DATA: it_datatab TYPE STANDARD TABLE OF t_datatab.  
23  
24 DATA: it_raw TYPE truxs_t_text_data.  
25
```

Data for the Batch Input Data of single transaction

```
28  
29 DATA: bdcdata TYPE STANDARD TABLE OF bdcdata, "OCCURS 0 WITH HEADER LINE.  
30 wa_bdcdata TYPE bdcdata.  
31
```

table for the Messages(Error) of call transaction

```
31  
32 DATA: messtab TYPE STANDARD TABLE OF bdcmsgcoll. " OCCURS 0 WITH HEADER LINE.  
33  
34
```

Data for Display structure

First is the Processing mode for call transaction N for Normal

Second is the Update mode for call transaction a for automatic

And v\_message that we will use later for our message table

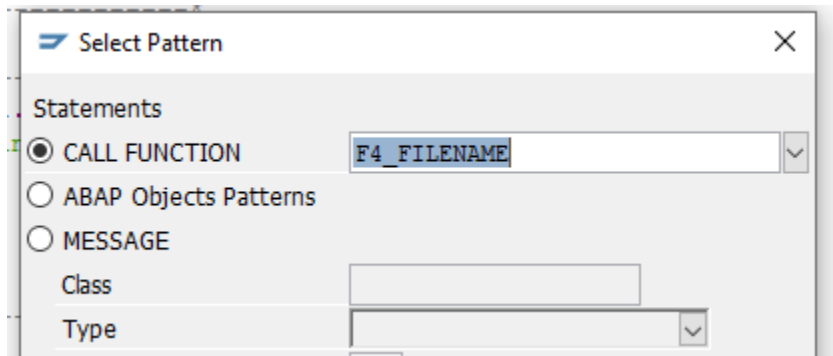
```
34  
35 DATA: ctumode LIKE ctu_params-dismode VALUE 'N'.  
36 DATA: cupdate LIKE ctu_params-updmode VALUE 'A'.  
37 DATA: v_message(480).  
38
```

Next is the declaration of SELECTION-SCREEN for our File Parameter.

You can change the text on selection screen by clicking F5 and going to selection text tab.

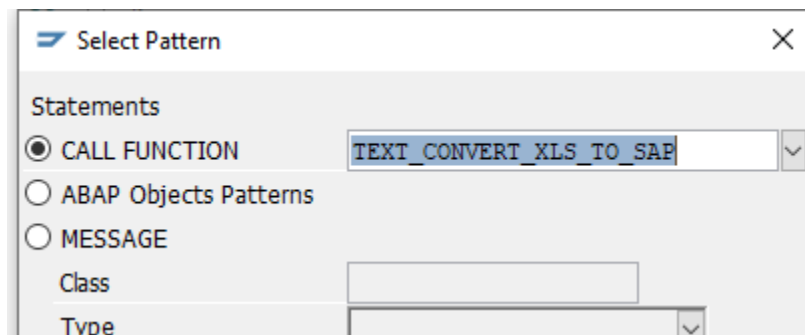
```
42 ☐ SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME TITLE TEXT-001.  
43 | PARAMETERS: p_file TYPE rlgrap-filename DEFAULT 'C:\Domain_Upload.xlsx'.  
44 | SELECTION-SCREEN END OF BLOCK b1.  
45
```

AT-SELECTION SCREEN VALUE REQUEST FOR file, you will go to PATTERN and call F4\_FILENAME



```
51 AT SELECTION-SCREEN ON VALUE-REQUEST FOR p_file.  
52 CALL FUNCTION 'F4_FILENAME'  
53 EXPORTING  
54     field_name = 'P_FILE'  
55 IMPORTING  
56     file_name  = p_file.  
57
```

On our START-OF-SELECTION FUNCTION go to Pattern and Call 'TEXT\_CONVERT\_XLS\_TO\_SAP'





We are using the it\_raw for the raw table data

P\_file for the file parameter

It\_datatab for our the destination of the converted data

```
63      START-OF-SELECTION.
64
65      CALL FUNCTION 'TEXT_CONVERT_XLS_TO_SAP'
66      EXPORTING
67      *      I_FIELD_SEPERATOR      =
68      i_line_header      = 'X'
69      i_tab_raw_data      = it_raw      " WORK TABLE
70      i_filename          = p_file
71      TABLES
72      i_tab_converted_data = it_datatab[] "ACTUAL DATA
73      EXCEPTIONS
74      conversion_failed    = 1
75      OTHERS               = 2.
76
77  IF sy-subrc <> 0.
78      MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
79      WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
80  ENDIF.
81
```

---

At the END-OF-SELECTION

We will the table using WRITE and vline for vertical lines and data from the it\_datatab

```
87      END-OF-SELECTION.
88      *Write Header Column Name
89      ULINE 1(255).
90      WRITE:/1 sy-vline, 'Domain Name'(c01), 15 sy-vline,
91      16 'Description'(c02), 50 sy-vline,
92      51 'Data Type'(c03), 60 sy-vline,
93      61 'No Characters'(c04), 75 sy-vline,
94      76 'Decimal Places'(c05), 90 sy-vline,
95      91 'Log Details'(c06), 255 sy-vline.
96      *SKIP.
97      ULINE (255).
98  LOOP AT it_datatab INTO DATA(wa_datatab).
99  |
```

Clear first the message table named messtab[]

The batch input data is stored in the bdcdata internal table, which is later used to call the transaction SE11 to create domains.

The processing mode we have declare above which is the ctumode, that has normal listing, and update that has automatic update

Messages from the message table

```
157      CLEAR messtab[].
158      CALL TRANSACTION 'SE11' WITH AUTHORITY-CHECK USING bdcdata
159              MODE      ctumode
160              UPDATE     cupdate
161              MESSAGES INTO messtab.
```

We will display message from wa\_messtab and save it in v\_message

```
183      LOOP AT messtab INTO DATA(wa_messtab).
184          MESSAGE ID      wa_messtab-msgid
185          TYPE            wa_messtab-msgtyp
186          NUMBER          wa_messtab-msgnr
187          INTO v_message
188          WITH wa_messtab-msgv1
189              wa_messtab-msgv2
190              wa_messtab-msgv3
191              wa_messtab-msgv4.
192
193      IF sy-tabix = 1.
194          wa_datatab-col6 = v_message.
195      ELSE.
196          CONCATENATE wa_datatab-col6 ',' v_message INTO wa_datatab-col6.
197      ENDIF.
198
199      CLEAR v_message.
200  ENDLOOP.
201  CLEAR bdcdata[].
202
```

To display the output of the program we use write and get the data from our work area data table which is wa\_datatab and separated by vertical line from sy-vline.

```
203      *      ULINE 2(255).
204      WRITE:/1 sy-vline, wa_datatab-col1, 15 sy-vline,
205              16 wa_datatab-col2, 50 sy-vline,
206              51 wa_datatab-col3, 60 sy-vline,
207              61 wa_datatab-col4, 75 sy-vline,
208              76 wa_datatab-col5, 90 sy-vline,
209              91 wa_datatab-col6, 255 sy-vline.
210      ULINE AT /1(255).
211  ENDLOOP.
212
```

```
|Column 1|Column 2|...
```

```
219  □ FORM bdc_dynpro USING program dynpro.
220      CLEAR wa_bdcdata.
221      wa_bdcdata-program = program.
222      wa_bdcdata-dynpro  = dynpro.
223      wa_bdcdata-dynbegin = 'X'.
224      APPEND wa_bdcdata TO bdcdata.
225  ENDFORM.
```

```
230  □ FORM bdc_field USING fnam fval.
231      * IF fval <> nodata.
232          CLEAR wa_bdcdata.
233          wa_bdcdata-fnam = fnam.
234          wa_bdcdata-fval = fval.
235          APPEND wa_bdcdata TO bdcdata.
236      * ENDIF.
237  ENDFORM.
238
```

File Selection:

Select File:

[illegible]

Result:

Domain Conversion and Upload					
Domain Name	Description	Data Type	No Characters	Decimal Places	Log Details
ZCHAR	Test CHAR Domain	CHAR	10		The object will be created in the original language English (EN),The object R
ZINT1	Test INT1 Domain	INT1	10		The object will be created in the original language English (EN),The object R
ZDATS	Test DATE Domain	DATS	8		The object will be created in the original language English (EN),The object R
ZDEC	Test DECIMAL Domain	DEC	10	3	The object will be created in the original language English (EN),The object R