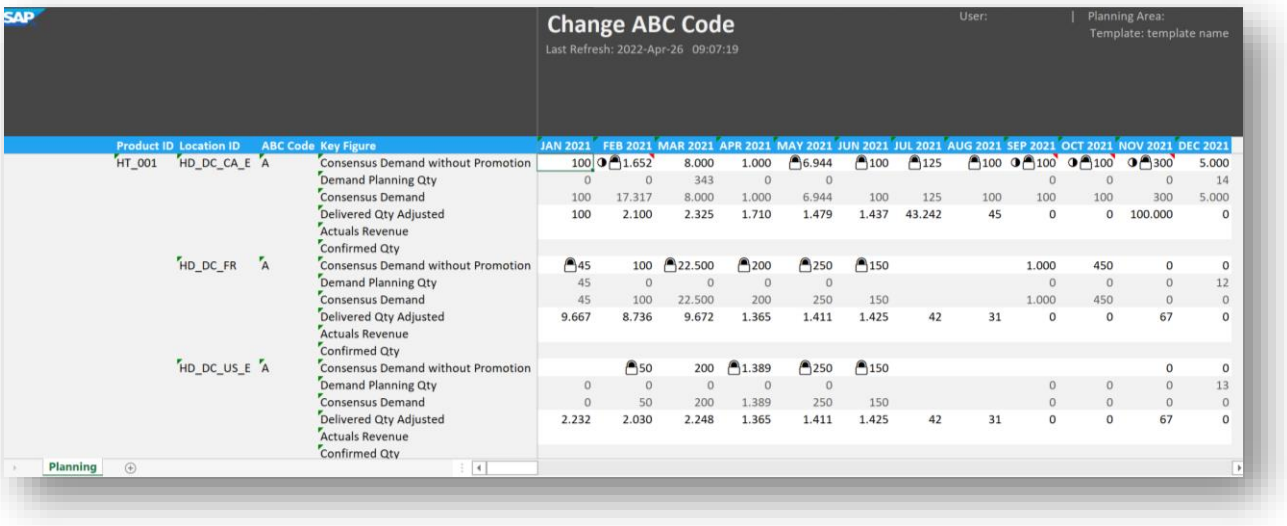
Change Master Data Attribute Value

With the newly provided API UpdateSingleMasterData it is possible to change the value of a specific master data attribute by using VBA code. If your business users must change one specific master data attribute several times per day you might include a specific logic in your template to make it possible to change the attribute value without the need of choosing ***Manage*** > ***Single...*** in the ***Master Data*** group in the ***SAP IBP*** ribbon.

We explain in our example how to implement changing the ABC code for a specific product ID by double-clicking the cell of the ABC code in the planning view. The same code can be used for other attributes then ABC code as well, so as always feel free to adjust the code to your needs.

HOW TO START

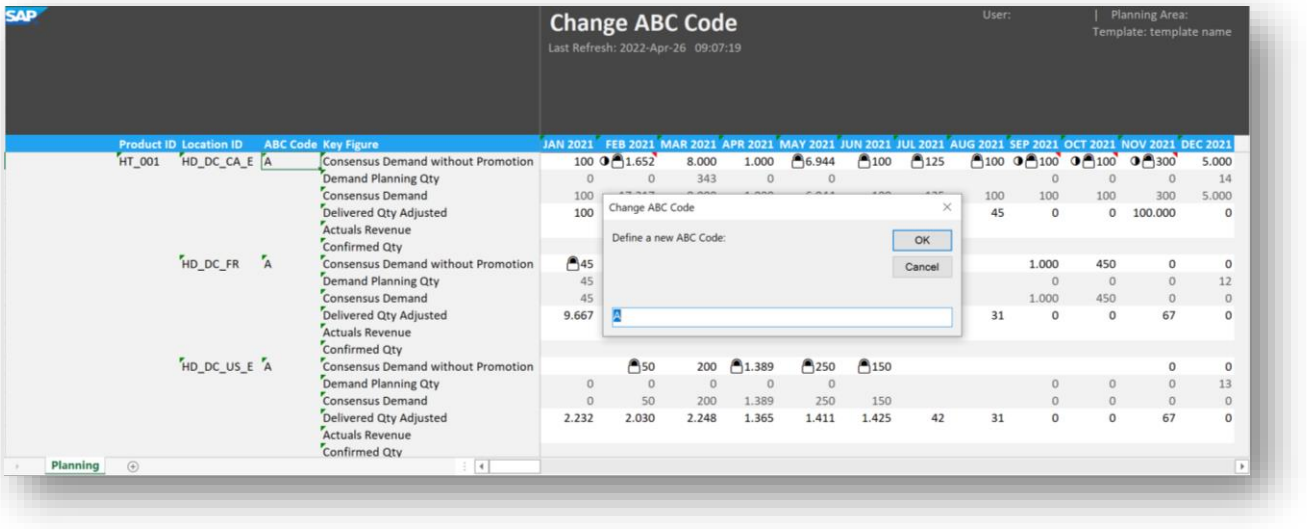
As a first step you need to create a planning view for which you want to include a specific logic to update master data attribute values. To use our sample code, the ABC Code needs to be selected as attribute, a part of others.



Product ID	Location ID	ABC Code	Key Figure	JAN 2021	FEB 2021	MAR 2021	APR 2021	MAY 2021	JUN 2021	JUL 2021	AUG 2021	SEP 2021	OCT 2021	NOV 2021	DEC 2021
HT_001	HD_DC_CA_E	A	Consensus Demand without Promotion	100	1.652	8.000	1.000	6.944	100	125	100	100	100	300	5.000
			Demand Planning Qty	0	0	343	0	0				0	0	0	14
			Consensus Demand	100	17.317	8.000	1.000	6.944	100	125	100	100	100	300	5.000
			Delivered Qty Adjusted	100	2.100	2.325	1.710	1.479	1.437	43.242	45	0	0	100.000	0
			Actuals Revenue												
			Confirmed Qty												
	HD_DC_FR	A	Consensus Demand without Promotion	45	100	22.500	200	250	150			1.000	450	0	0
			Demand Planning Qty	45	0	0	0	0				0	0	0	12
			Consensus Demand	45	100	22.500	200	250	150			1.000	450	0	0
			Delivered Qty Adjusted	9.667	8.736	9.672	1.365	1.411	1.425	42	31	0	0	67	0
			Actuals Revenue												
			Confirmed Qty												
	HD_DC_US_E	A	Consensus Demand without Promotion		50	200	1.389	250	150					0	0
			Demand Planning Qty	0	0	0	0	0				0	0	0	13
			Consensus Demand	0	50	200	1.389	250	150			0	0	0	0
			Delivered Qty Adjusted	2.232	2.030	2.248	1.365	1.411	1.425	42	31	0	0	67	0
			Actuals Revenue												
			Confirmed Qty												

The trigger for the change of the ABC Code is a double-click on the cell of the ABC code of a specific product ID. To implement this we use the Worksheet_BeforeDoubleClick VBA event provided by Microsoft in combination with the SAP IBP APIs GetAttributeValues and UpdateSingleMasterData.

If a cell is double-clicked, we check whether the cell includes the ABC code. If yes, we determine the Product ID of the cell which was double-clicked with the API GetAttributeValues. Then we open an inputbox, to allow the user to enter a new value for ABC code.



Once the user confirmed the input by clicking ok, we are validating the input and calling the API UpdateSingleMasterData passing the master data type “Product”, the Product ID and the new ABC code.

The code for this use case, is part of the worksheet “Planning” and looks the following:

```
Private IBPAutomationObject As Object

Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)

Dim ABCcode As String
Dim attributes() As String
Dim formula
Dim newABC As Variant

On Error GoTo ErrorHandling:
    If IBPAutomationObject Is Nothing Then Set IBPAutomationObject =
Application.COMAddIns("IBPXLClient.Connect").Object

    'Check if the cell includes the ABC code
    formula = Target.Formula2
    If InStr(1, formula, "=@ EPMOlapMemberO(""[ABCID].[].[") = 0 Then
        Exit Sub
    End If

    'Determine the Product ID
    attributes = IBPAutomationObject.GetAttributeValues(Target.Offset(0, 4))
    attributes = Filter(attributes, "[PRDID].[].")
    If UBound(attributes) < 0 Then
        MsgBox "The Product ID couldn't be determined.", vbOKOnly, "Microsoft Excel: Custom VBA code"
        Exit Sub
    End If

    'Show a inputbox for entering a new ABC code
    ABCcode = Target.Value2
    newABC = InputBox("Define a new ABC Code:", "Change ABC Code", ABCcode)

    'Check the input
    If newABC = "A" Or newABC = "B" Or newABC = "C" Then
        'Update ABCCode for the specific Product ID
        ABCcode = "[ABCID].[].[" + newABC + "]"
        Call IBPAutomationObject.UpdateSingleMasterData("Product", attributes, Array(ABCcode))
        Call IBPAutomationObject.Refresh
    ElseIf StrPtr(newABC) = 0 Then
        Exit Sub
```

```
    Else
        MsgBox "The input you entered is not valid.", vbOKOnly, "Microsoft Excel: Custom VBA code"
    End If
    Exit Sub

ErrorHandling:
'Implement an error handling to help the user to understand what went wrong
MsgBox Err.Description, vbOKOnly, "Microsoft Excel: Custom VBA code"

End Sub
```

Code explained row by row:

Please note: Code lines which already have been explained in the tutorial for use cases provided with release 2202 are not explained in detail again, so please check those samples as well.

```
If IBPAutomationObject Is Nothing Then Set IBPAutomationObject =
Application.COMAddIns("IBPXLClient.Connect").Object
```
If the SAP IBP automation object was not set yet, we retrieve it from the SAP IBP, add-in for Microsoft Excel (Excel add-in).

```
formula = Target.Formula2
If InStr(1, formula, "=@ EPMOlapMemberO(""[ABCID].[].[") = 0 Then
    Exit Sub
End If
```
Check whether the formula of the Target cell which was double-clicked contains the ABC code.

```
attributes = IBPAutomationObject.GetAttributeValues(Target.Offset(0, 4))
```
Determine the attribute values of the Target cell which was double-clicked.
As the range which is provided as parameter for GetAttributeValues needs to be part of the data area of the planning view, we need to shift the Target range by several columns (to the right). The exact number of shifted columns depends on the planning view that you have created (number of attributes and where the ABC code column is situated).

```
attributes = Filter(attributes, "[PRDID].[].")
```
Filter out the entry for Product ID from the array of attribute values, by using the Filter function (VBA function provided by Microsoft).

```
If UBound(attributes) < 0 Then
    MsgBox "The Product ID couldn't be determined.", vbOKOnly, "Microsoft Excel: Custom VBA code"
    Exit Sub
End If
```
If the filtered array contains no entries, show the error message “The Product ID couldn't be determined." and exit the sub, else continue with the next steps.

```
ABCcode = Target.Value2
```
Save the current value of the Target cell which was double-clicked as ABCcode using the Property Value2 provided by Microsoft.

```
newABC = InputBox("Define a new ABC Code:", "Change ABC Code", ABCcode)
```
Open an inputbox with prompt “Define a new ABC Code:”, title “Change ABC Code” and default value ABCcode and save the entered value as newABC.

```
If newABC = "A" Or newABC = "B" Or newABC = "C" Then
    …
ElseIf StrPtr(newABC) = 0 Then
    Exit Sub
Else
    MsgBox "The input you entered is not valid.", vbOKOnly, "Microsoft Excel: Custom VBA code"
End If
```
Check the input. In our case only “A”, “B” or “C” are allowed. If the user clicks cancel (StrPtr(newABC) = 0) we just close the window, else the error message "The input you entered is not valid." shows up.

ABCcode = "[ABCID].[].[" + newABC + "]"
To pass the non-key attribute values that you want to change (parameter nonKeyValues of API UpdateSingeMasterData), the attribute values need to have the format [attribute ID].[].[attribute value] in this case for example [ABCID].[].[A].

Call IBPAutomationObject.UpdateSingleMasterData("Product", attributes, Array(ABCcode))
Call the API UpdateSingleMasterData, passing “Product” as masterDataType, the attributes array with the Product ID as keyValues and Array(ABCCode) as nonKeyValues.
Both parameters keyValues and nonKeyValues are from type String[], therefore you need to pass a one-dimensional string array, even if only one key attribute value or non-key attribute value is passed.
Please make sure to only pass non-key attribute values that are different from the actual saved value, else an error message will be shown, see UpdateSingleMasterData on the SAP Help Portal for further information.

Call IBPAutomationObject.Refresh
Call the API Refresh to refresh the active worksheet. With this the changed value will get visible for the user in the planning view.

Further comments to the code:

- Adjust the Offset of the Target cell to be used for GetAttributeValues according to your planning view.
- If you want to implement a generic logic that every attribute can be double-clicked and changed some more effort and VBA code is needed, similar applies as already described in sample “1.2 Show Master Data Details”. To change master data attribute values via the API UpdateSingle-MasterData the ID or name of the master data type and the key attribute values are needed, please check the other tutorial for further information/ideas.
- The API UpdateSingleMasterData is not meant for mass changes of master data attributes. If you want to change a specific attribute value as shown in our example this is a good option.

Follow us

www.sap.com/contactsap