

KARL JOSEF HENSEL

SAP INTERFACE PROGRAMMING WITH RFC AND VBA

*Edit SAP data with MS
Access*





Dear reader,

Thank you for choosing my book.

Each of my books should satisfy you. In order for me to succeed again and again, I depend on feedback. Please let me know what you think about this book. Your critical and friendly suggestions, your wishes and ideas will help me further. I look forward to dialogue with you.

Mail: postmaster@edv-bh.de

The migration of data is an important process when introducing SAP systems or adopting new data from third-party systems. This takes up a large proportion of the time and thus also the costs in migration projects. This book gives you practical tips and examples of how you can use Microsoft products and the VBA programming language to quickly and inexpensively create useful migration tools yourself. It is not necessary to install the software on the SAP systems. This saves the time-consuming procedure of software implementation in SAP systems. Developments with ABAP code, complex documentation, tests in development, quality and productive systems are not necessary and save time and money. The batch input interface and the LSMW and CATT tools are no longer required.

The book is suitable for administrators, technical consultants and developers in the SAP environment. Even power users from the specialist departments will be able to productively extract or upload data from the SAP systems after reading it.

The programs created according to my method can be used for all SAP modules. As an example, I use functionalities of the SAP module PS. By using ready-made SAP functions and BAPIs, functionalities such as creating and changing investment programs, measure requests and projects can be carried out.

Kerpen in 2022

Karl Josef Hensel

SAP interface programming with RFC and VBA

Edit SAP IM/PS data with MS Access

Table of contents

Chapter 1

Chapter 1.1 – The logon to the SAP system

Chapter 2

Chapter 2.1 – Remote Function Call (RFC)

Chapter 2.2 – The function module RFC_READ_TABLE

Chapter 2.3 – The SAP table PRPS

Chapter 2.4 - Download with RFC_READ_TABLE

Chapter 2.5 - Automatically create the download file

Chapter 2.6 - RFC_Read_Table Download with extensive selection options

Chapter 3

Chapter 3.1 – Use functions of the SAP BAPI's

Chapter 3.2 – Determine the status of WBS elements

Chapter 3.3 – Change the status of WBS elements in the system

Chapter 4

Chapter 4.1 – Download data from WBS elements

Chapter 4.1.1 – What is a WBS-Element - small digression

Chapter 4.1.2 – Download WBS-Element data

Chapter 4.1.3 – The call of the function module

Chapter 4.5 – VBA code for data download from WBS elements

Chapter 5

Chapter 5.1 WBS-Elements data upload

Chapter 5.1.1 Create WBS-Element items

Chapter 5.1.2 Change WBS-Elements

Chapter 5.1.3 Customer enhancements to the BAPIs

Chapter 5.1.4 Add WBS-Elements

Chapter 6

Chapter 6.1 WBS-Element Data Mapping

Chapter 6.1.1 What is the meaning of mapping ?

Chapter 6.1.2 Data mapping methods

Chapter 6.1.2.1 1:1 Data mapping with support of a SQL formula

Chapter 6.1.2.2 1:1 Data mapping with the support of a VBA function

Chapter 6.1.3 X:1 Mapping

Chapter 6.1.3.1 Simple mapping with the support of an SQL formula

Chapter 6.1.3.2 Data mapping with the support of a VBA function

Chapter 6.1.4 Data mapping with internal tables

Chapter 6.1.5 Data mapping with external tables

Chapter 7

Chapter 7.1 R/3 Tables overview

Chapter 7.1.1 Overview of all tables in SAP ERP

Chapter 7.1.2 Tables for the project system module PS

Epilogue

Index

Chapter 1

Chapter 1.1 – The logon to the SAP system

You have to decide on a login type in advance when programming.

I definitely recommend using the logon for functions and staying on track.

⇒ Set FunctionCtrl = CreateObject("SAP.Functions")

The advantage is that all SAP functions as well as all BAPI functions can be used, which is not possible the other way around and leads to a constant new logon.

- If no logon parameters are given in the program, the SAP GUI logon is used automatically.

Logon example:

```
!***** RFC LOGON *****  
  
If Not sapConnection Is Nothing Then  
    If sapConnection.IsConnected <> 1 Then  
        If RFC_Connect() = False Then Exit Sub  
    End If  
Else  
    If RFC_Connect() = False Then Exit Sub  
End If  
  
!***** RFC LOGON END *****
```

Requirements:

Create a module in which all RFC functions are collected, e.g.

Module name SAP_RFC

The following objects are declared:

Declarations:

Public FunctionCtrl As Object

Public sapConnection As Object

Private oRowOPTIONS As Object

Private Akt_User As String

Private Akt_PW As String

Private Akt_SAP_System As String

Private Akt_SAPMandant As String

Private Akt_SAPLang As String

Private Akt_ApplicationServer As String

Private Akt_SystemNumber As Integer

Private oRFC_READ_TABLE As Object

Private MyArray(10) As Variant

Required functions:

! *****

' Public Functions

! *****

```
Public Function RFC_Connect(Optional s_User = "", _  
    Optional s_PW = "", _  
    Optional s_SAP_System = "", _  
    Optional s_SAPMandant = "", _  
    Optional s_SAPLang = "", _  
    Optional s_ApplicationServer = "", _  
    Optional i_SystemNumber = 0) As Boolean
```

```
On Error GoTo Error
```

```
Dim Pruefnummer
```

```
Dim TEMP As Variant
```

```
RFC_Connect = False
```

```
' Parameter setzen
```

```
If s_User <> "" Then Akt_User = s_User
```

```
1:
```

```
    Pruefnummer = 1
```

```
    TEMP = FunctionCtrl.Count
```

```
2:
```

```
    Pruefnummer = 2
```

```
    If sapConnection.IsConnected <> 1 Then GoTo Error
```

```
    RFC_Connect = True
```

```
    Exit Function
```

```
Error:
```

```
    Err.Clear
```

```
    Select Case Pruefnummer
```

```
    Case 1 ' FunctionCtrl nicht initiiert.
```

```
        pr_Init
```

```
        GoTo 1
```

```

Case 2 ' Noch keine SAP-Anmeldung
If pr_Logon() = False Then Exit Function
GoTo 2
End Select

End Function

' *****

' Private Functions

' *****

Private Sub pr_Init()
    Set FunctionCtrl = CreateObject("SAP.Functions")
    Set sapConnection = FunctionCtrl.Connection
End Sub

Private Function pr_Logon() As Boolean
    Dim silent As Boolean

    If sapConnection.IsConnected = 1 Then Exit Function

    sapConnection.Client = Akt_SAPMandant
    sapConnection.LANGUAGE = Akt_SAPLang
    sapConnection.User = Akt_User
    sapConnection.Password = Akt_PW
    sapConnection.SYSTEM = Akt_SAP_System
    sapConnection.ApplicationServer = Akt_ApplicationServer
    sapConnection.SystemNumber = Akt_SystemNumber

    silent = Not (Akt_User = "" Or Akt_PW = "")
    sapConnection.RFCWithDialog = False
    'sapConnection.RFCWithDialog = Not silent
    If Not sapConnection.Logon(0, silent) Then
        MsgBox "logon failed !!"
    End If
End Function

```

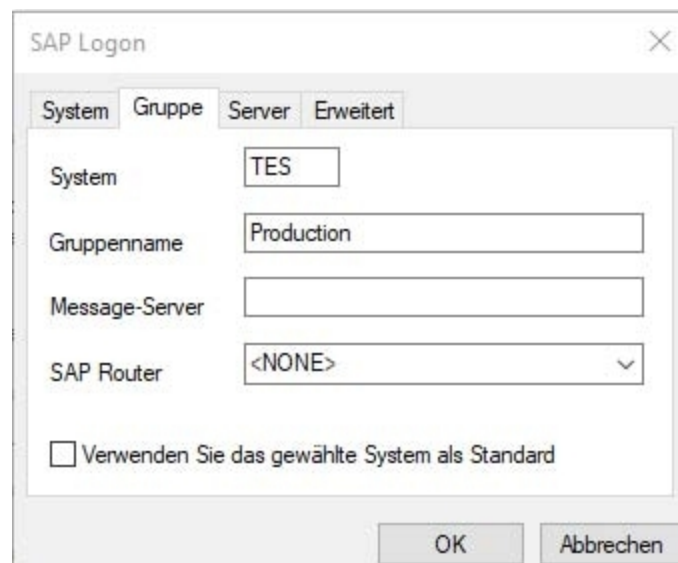
```
pr_Logon = False  
Exit Function  
End If
```

```
pr_Logon = True  
End Function
```

Execution:

The logon routine is ready and now we can test it.

Start the Login procedure.



The screenshot shows the 'SAP Logon' dialog box with the 'System' tab selected. The fields are filled with the following values:

Field	Value
System	TES
Gruppenname	Production
Message-Server	
SAP Router	<NONE>

At the bottom, there is a checkbox labeled 'Verwenden Sie das gewählte System als Standard' which is currently unchecked. The 'OK' and 'Abbrechen' buttons are visible at the bottom right.

The SAP GUI logon screen appears. Enter your system data in the same way as in the SAP GUI. From now on you are connected to the system.

The procedure and the functions are written in VBA and work with all MS Office products. In this book I took MS Access.

Chapter 2

Chapter 2.1 – Remote Function Call (RFC)

Remote Function Call (RFC) is a term from the SAP software environment (especially the SAP ABAP systems, e.g. R/3 and its successors).

On the one hand, RFC is a synonym for the Remote Procedure Call concept (RPC for short). This designates methods with which functions in a remote system are called. On the other hand, RFC is also the umbrella term for SAP's own protocols and interfaces for processing such function calls through to their implementation.

Communication is possible both between SAP systems and between an SAP system and a non-SAP system. The latter typically uses an RFC library provided by SAP for the respective runtime environment or programming language.

A distinction is made between sRFC, which carries out synchronous communication, tRFC (transactional RFC) for asynchronous communication, and qRFC (queued RFC). The qRFC is a further development of the tRFC, which ensures that the order of processing individual units defined by the application is retained.

The possible uses are diverse. Data can be transported into or out of an SAP system and thus made available in almost any environment. An important application is the processing of transactions across system boundaries, e.g. B. for the replication/synchronization of data stocks (see SAP ALE).

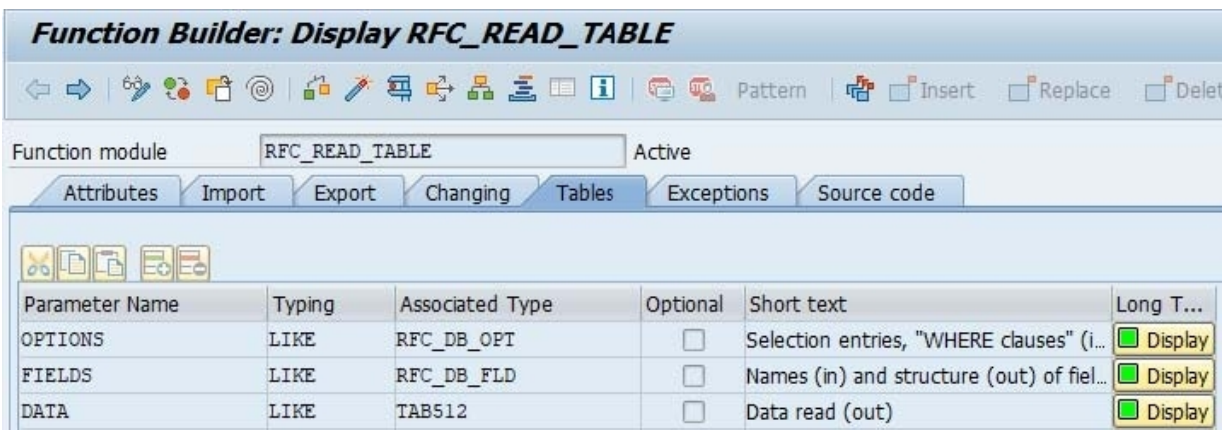
All RFC types are transmitted via the standardized interface CPI-C (Common Programming Interface-Communication) or TCP/IP. The latter is always used nowadays for communication between different systems. In detail: SAP servers accept RFC connections under the TCP ports 33NM (NM is the two-digit instance number/system number of the SAP system; entry in the services file: sapgwNM). That is, the TCP port range from 3300 to 3399 is provided by SAP for this purpose; but there is no IANA registration for this.

Here I present how RFC is used together with Microsoft Access (copyright Microsoft) to get data from a SAP system.

Source: Wikipedia

Chapter 2.2 – The function module RFC_READ_TABLE

The function module RFC_READ_TABLE is delivered by SAP as standard. It reads the definition and content of any table defined in the repository. The output is in the DATA table.



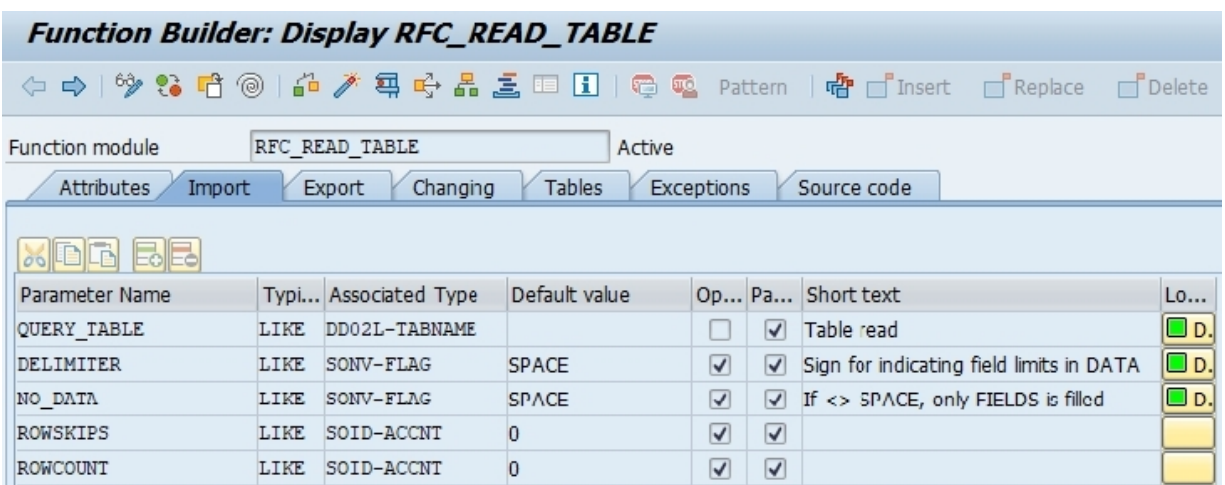
Function Builder: Display RFC_READ_TABLE

Function module: RFC_READ_TABLE Active

Attributes Import Export Changing **Tables** Exceptions Source code

Parameter Name	Typing	Associated Type	Optional	Short text	Long T...
OPTIONS	LIKE	RFC_DB_OPT	☐	Selection entries, "WHERE clauses" (i...	
FIELDS	LIKE	RFC_DB_FLD	☐	Names (in) and structure (out) of fiel...	
DATA	LIKE	TAB512	☐	Data read (out)	

In the import parameter QUERY_TABLE must be given the table name, in the import parameter DELIMITER must be given a field separator. In the FIELDS table, the fields are specified in the desired order.



Function Builder: Display RFC_READ_TABLE

Function module: RFC_READ_TABLE Active

Attributes **Import** Export Changing Tables Exceptions Source code

Parameter Name	Typi...	Associated Type	Default value	Op...	Pa...	Short text	Lo...
QUERY_TABLE	LIKE	DD02L-TABNAME		☐	☒	Table read	
DELIMITER	LIKE	SONV-FLAG	SPACE	☒	☒	Sign for indicating field limits in DATA	
NO_DATA	LIKE	SONV-FLAG	SPACE	☒	☒	If <> SPACE, only FIELDS is filled	
ROWSKIPS	LIKE	SOID-ACCNT	0	☒	☒		
ROWCOUNT	LIKE	SOID-ACCNT	0	☒	☒		

A line from the read table is then supplied in each line of DATA, whereby the field contents read, separated by the specified separator, are written one after the other in CHAR representation in the only field of DATA. A maximum of 512 characters including separators are output. By selecting fields, it is also possible to read in tables that are too wide to fit in one line of DATA.

Selection criteria are passed in the OPTIONS table (maximum 72 characters per data record) in order to limit the number of lines to be read. RFC_READ_TABLE forwards the selection options to ABAP/4-SQL without checking; if the syntax is incorrect, the block triggers an exception. If the 72 characters are not sufficient, a further data record must be added to the Options table, which can then accommodate 72 characters again. In this way, a comprehensive selection can be made. The clearer the selection, the faster the download will be. The function module checks the existence of an authorization for the authorization object S_TABU_DIS. The authorization protection is therefore the same as when displaying tables with transaction SE16.

Chapter 2.3 – The SAP table PRPS

The PRPS table is a standard SAP table in which WBS-Element master data (project structure plan) is stored. From this table we need the internal record number (POSID), the object number (unformatted WBS number) and the superordinate project number (PSPHI)

Chapter 2.4 - Download with RFC_READ_TABLE