

Excel and Access

Getting data FROM Excel

1. External Data – Import - Excel file – into New Table
2. Same thing programmatically

`DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "BOM info", "C:\Documents and Settings\username\My Documents\Presentations\Excel\BOM INFO.xls", True`

`DoCmd.TransferSpreadsheet [TransferType], [SpreadsheetType], [TableName], [FileName], [HasFieldNames], [Range]`

`TransferType = acExport, acImport, acLink`

`SpreadsheetType = acSpreadsheetType` most common are

`acSpreadsheetTypeExcel9` (Excel 2000-2003 .xls) and `acSpreadsheetTypeExcel12` (Excel 2007 .xlsx)

`HasFieldNames = True` (has column headers)

`Range =` If other than first worksheet put in worksheet and or cell range like

`"Sheet2" or "Sheet2!A6:D15"`

Running a saved import

`DoCmd.RunSavedImportExport "Import-BOM INFO"`

`DoCmd.RunSavedImportExport [Import Specification]`

`CurrentProject.ImportExportSpecifications.Count` lists how many Import Specifications you have saved in this project, you could sift through them

3. External Data – Import - Excel file – Link to a table

Creates a READ ONLY table but still useful if the info needs to get updated from outside the DB.

Putting data INTO Excel

1. Select table or query that you want to export. External Data – Export - Excel file – into named spreadsheet.
2. Same thing programmatically

```
DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel12, "BOM", "C:\Documents and Settings\username\My Documents\Presentations\Excel\qryBOM.xlsx", True
```

Running a saved export

```
DoCmd.RunSavedImportExport "Export-qryBOM"
```

3. If want to place on other than 1st worksheet, format the info, or apply to a different range. You must either manipulate the resulting workbook or make through VBA code.

```
Public oBook As Excel.Workbook
```

```
Public Sub TrendAnalysis()
```

```
Dim bRunningAlready As Boolean
```

```
Dim oApp As Object
```

```
On Error GoTo TrendAnalysis_Error
```

```
DoCmd.OpenForm "Report Date Range", acNormal, , , acDialog, "Trend Analysis"
```

```
bRunningAlready = IsExcelRunning()
```

```
If bRunningAlready Then
```

```
Set oApp = GetObject(, "Excel.Application")
```

```
Else
```

```
Set oApp = CreateObject("Excel.Application")
```

```
oApp.Visible = True
```

```
End If
```

```
oApp.ScreenUpdating = False ' this prevents screen flashing and speeds up processing
```

```
Set oBook = oApp.Workbooks.Add
```

```
AddTrans 1, "UnitsOrdered"
```

```
AddTrans 2, "UnitsReceived"
```

```
AddTrans 3, "UnitsAdded"
```

```
AddTrans 4, "UnitsUsed"
```

```
AddTrans 5, "UnitsSold"
```

```
AddTrans 6, "UnitsShrinkage"
```

```
AddTrans 7, "UnitsCRWaste"
```

```
AddTrans 8, "UnitsCRConsignment"
```

```
AddTrans 9, "UnitsInternalUsage"
```

```
oApp.ScreenUpdating = True
```

```
TrendAnalysis_Error:
```

```
DoCmd.Close acForm, "Report Date Range"
```

```
End Sub
```

```

Private Sub AddTrans(WkShtNum As Integer, wkshtName As String)
Dim oWkSht As Excel.Worksheet
Dim sSQL As String
Dim rs As New ADODB.Recordset
Dim i As Integer
Dim Row As Integer

```

```

On Error GoTo TrendAnalysis_Error
If WkShtNum > oBook.Worksheets.Count Then
    oBook.Worksheets.Add After:=oBook.Worksheets(WkShtNum - 1)
End If
Set oWkSht = oBook.Worksheets(WkShtNum)

```

```

sSQL = "TRANSFORM Sum(qryInventoryTransactions.[SumOf" & wkshtName & "]) AS SumOfSumOf" & wkshtName & _
" SELECT qryInventoryTransactions.[ProductName], Sum(qryInventoryTransactions.[SumOf" & _
wkshtName & "]) & _AS Total" & wkshtName & " FROM qryInventoryTransactions WHERE" & _
"(((qryInventoryTransactions.TransDate) >= " & _
Format(Forms("Report Date Range")!BeginDate, "yyyy/mm") & _
" And (qryInventoryTransactions.TransDate) <= " & _
Format(Forms("Report Date Range")!EndDate, "yyyy/mm") & "))) " & _
"GROUP BY qryInventoryTransactions.[ProductName] ORDER BY " & _
"qryInventoryTransactions.ProductName PIVOT qryInventoryTransactions.[TransDate];"

```

```

With oWkSht
    .Name = wkshtName
    rs.Open sSQL, CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    If Not rs.EOF Then
        For i = 0 To (rs.Fields.Count - 1) ' loop through the query field names to fill your header row
            .Cells(1, i + 1).Value = rs.Fields(i).Name
        Next
    End If
    Row = 2
    While Not rs.EOF ' loop through your records and fill each row
        For i = 0 To (rs.Fields.Count - 1)
            .Cells(Row, i + 1).Value = rs.Fields(i).Value
        Next
        rs.MoveNext
        Row = Row + 1
    Wend
    rs.Close
    Set rs = Nothing
    .Columns("A:A").ColumnWidth = 13 ' select column A and set width of column to 13 pixels
    .Columns("B:B").ColumnWidth = 17
    .Columns("A:A").HorizontalAlignment = xlLeft ' select column A and align to left
End With
TrendAnalysis_Error:
End Sub

```

```

Public Function IsExcelRunning()
    Dim xlApp
    On Error Resume Next
    Set xlApp = GetObject("Excel.Application")
    IsExcelRunning = (Err.Number = 0)
    Set xlApp = Nothing
    Err.Clear
End Function

```

Other helpful things to know:

```

'open an existing workbook
Set oBook = oApp.Workbooks.Open(CurrentProject.Path & "/IOUF Template.xls")

```

```

'save a workbook with a new name
oBook.SaveAs CurrentProject.Path & "/Monthly Shipped " & sMonth

```

```

'close your spreadsheet if you do not want to leave it displayed
oBook.Close

```

```

'close Excel ONLY if you opened it
If Not bRunningAlready Then oApp.Quit 'Don't close Excel unless you opened it!

```

```

'activate the worksheet
.Activate

```

```

'set the value of an individual cell
.Cells(Row, 1).Value = rs.Fields("MinIND").Value

```

```

'insert a new row and copy formatting
Row = Row + 1
.Rows(Row).Copy
.Rows(Row + 1).PasteSpecial

```

```

' find the last used row in column 1
iEnd = oWkSht.Cells(65000, 1).End(xlUp).Row

```

Quickest way to figure out what Excel methods to use is to record a macro, then copy the code in from last Macro. Add With oWkSht and End With around copied code.

Other tips – be careful using ActiveSheet & ActiveWorkbook because they might have other Workbooks open.

This has to be the first FUNCTION called in your program. Place a call to it first thing in the LOAD event of your startup form!

```
Sub CheckReferences()  
Dim intX As Long  
  
intX = 1  
  
' Just use Application.References.AddFromGuid "{00020813-0000-0000-C000-000000000046}", 0, 0  
'... The 0,0 should pick the latest version installed on that machine  
  
Do Until intX > References.Count  
  'Debug.Print " Case:" & References.Item(intX).Name & ": References.AddFromGuid :" & References.Item(intX).Guid  
  If References.Item(intX).IsBroken Then  
    Select Case References.Item(intX).Name  
  
      Case "VBA"  
        References.AddFromGuid "{000204EF-0000-0000-C000-000000000046}", 0, 0  
      Case "Access"  
        References.AddFromGuid "{4AFFC9A0-5F99-101B-AF4E-00AA003F0F07}", 0, 0  
      Case "stdole"  
        References.AddFromGuid "{00020430-0000-0000-C000-000000000046}", 0, 0  
      Case "DAO"  
        References.AddFromGuid "{00025E01-0000-0000-C000-000000000046}", 0, 0  
      Case "MDACVer"  
        References.AddFromGuid "{54AF9343-1923-11D3-9CA4-00C04F72C514}", 0, 0  
      Case "ADODB"  
        References.AddFromGuid "{00000201-0000-0010-8000-00AA006D2EA4}", 0, 0  
      Case "Word"  
        References.AddFromGuid "{00020905-0000-0000-C000-000000000046}", 0, 0  
      Case "Office"  
        References.AddFromGuid "{2DF8D04C-5BFA-101B-BDE5-00AA0044DE52}", 0, 0  
      Case "Excel"  
        References.AddFromGuid "{00020813-0000-0000-C000-000000000046}", 0, 0  
      Case "ADOX"  
        References.AddFromGuid "{00000600-0000-0010-8000-00AA006D2EA4}", 0, 0  
      Case "PowerPoint"  
        References.AddFromGuid "{91493440-5A91-11CF-8700-00AA0060263B}", 0, 0  
  
      Case "ACRODISTXLib"  
        References.AddFromGuid "{317DA881-ECC5-11D1-B976-00600802DB86}", 0, 0  
      Case "Acrobat"  
        References.AddFromGuid "{E64169B3-3592-47D2-816E-602C5C13F328}", 0, 0  
      Case "AcrobatAccessLib"  
        References.AddFromGuid "{C523F390-9C83-11D3-9094-00104BD0D535}", 0, 0  
  
    End Select  
  End If  
  intX = intX + 1  
Loop  
  
End Sub
```