

# SQL to VBA & Back Again - Add In

Version 3.42 - 11 July 2024

## Installation Info

This version of the **Sql2Vba** app is supplied as an **Access add-in** so it is available for use with all your Access apps. It is available as an **ACCDE** file for use with either **32-bit** or **64-bit Access** from **version 2010** onwards.

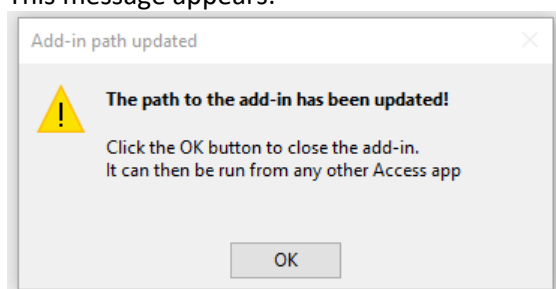
If preferred it can also be used as a standalone app.

To install the add-in:

1. Download the **32-bit** or **64-bit** version of the add-in to match your **Access / Office** bitness
2. Unblock the **zip file** to remove the 'mark of the web'. For more details, see my article: [Unblock downloaded files by removing the Mark of the Web](#)
3. Unzip and **copy** the **ACCDE** file **SQL2VBA\_v342\_AI\_x32.accde** or **SQL2VBA\_v342\_AI\_x64.accde** to your **Access add-ins** folder.  
The folder path is usually: **C:\Users\YourUserNameGoesHere\AppData\Roaming\Microsoft\AddIns**

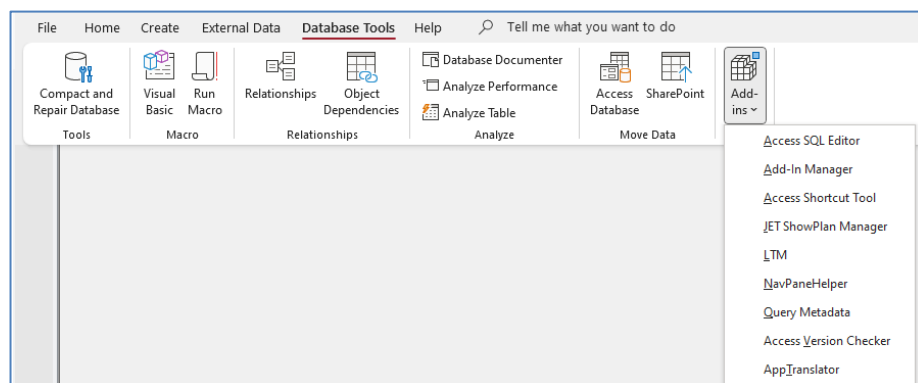
**NOTE:** You will need to tick **Show Hidden Items** in **File Explorer** to see the **AppData** folder

4. Browse to and run the **ACCDE** application from your **AddIns** folder to 'register the file location'.  
This message appears:

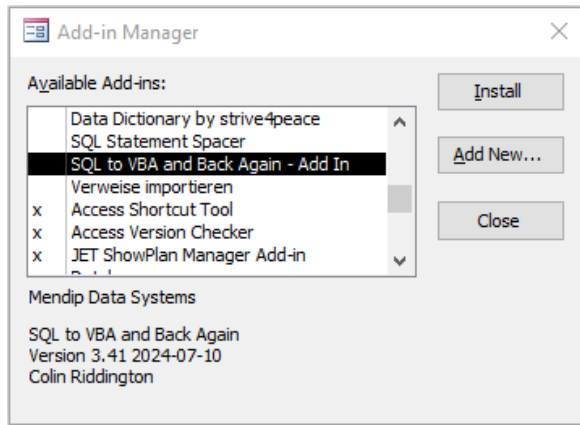


The path settings have been updated in a hidden system table **USysRegInfo**. Click **OK** to close the add-in

5. Open any Access app and go to **Database Tools ... Add-Ins** on the ribbon

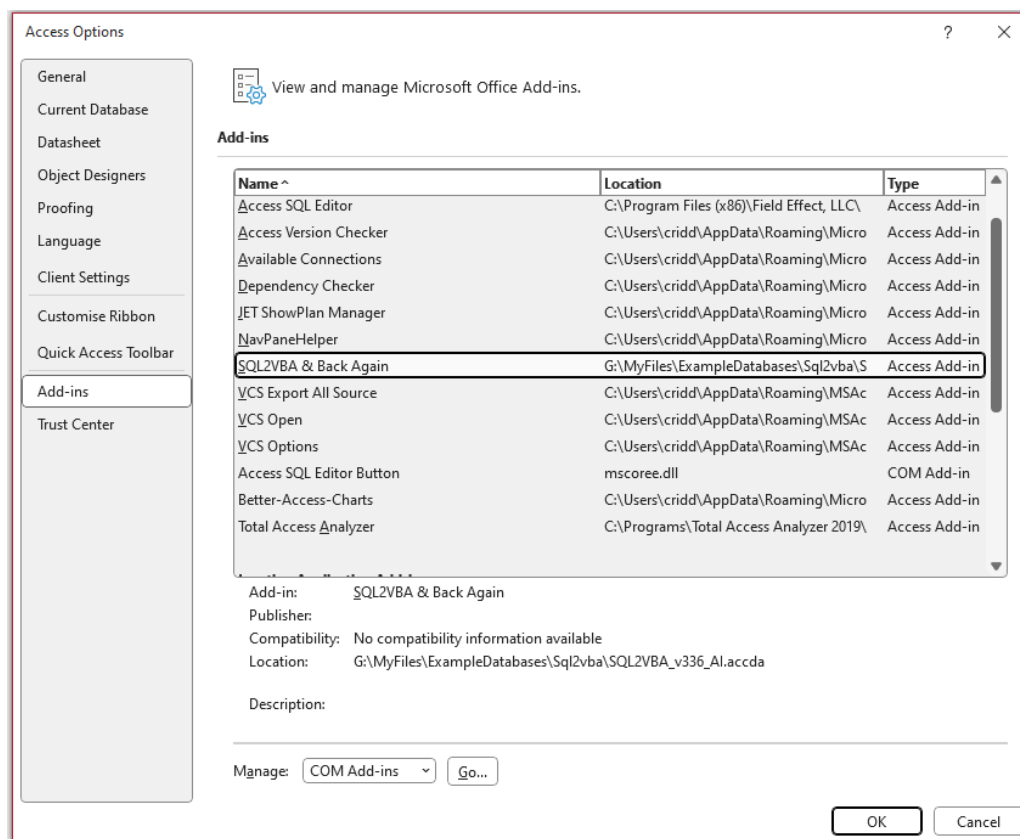


6. Click **Add-in Manager**. This dialog appears showing all the available add-ins on your workstation. **Installed add-ins** are marked with a **X**



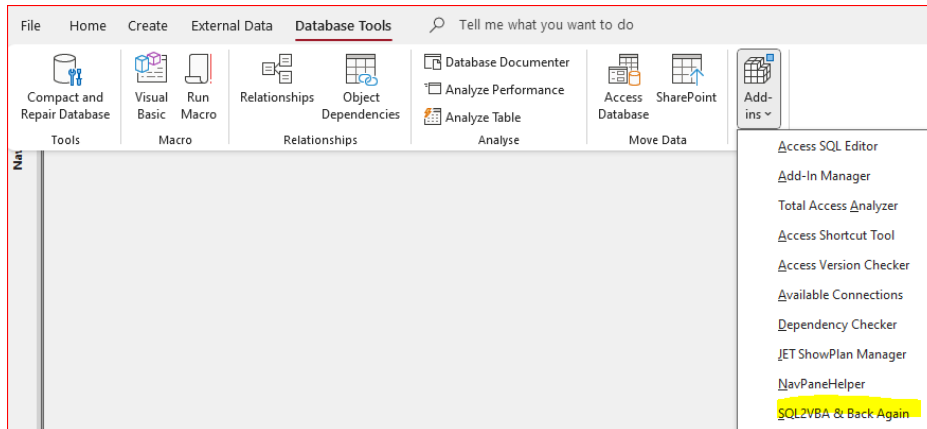
Select **SQL to VBA and Back Again – Add In** and click **Install** then click **Close**  
The **Available Connections** add-in is now available for use with all your Access apps

7. You can also install **Add-Ins** from the **File . . . Options** menu



Click **Access Add-ins** on the dropdown then click **Go**  
The **Access Add-in Manager** will open - same as **step 6**

8. To run the add-in, again go to **Database Tools ... Add-Ins** on the ribbon  
Select **SQL to VBA and Back Again** from the list



The **SQL2VBA** form will open in your application.



9. The form will appear in the centre of the screen but can be moved to any more convenient location.  
It can also be minimized at any time.
10. To use the form, copy a query SQL or VBA statement to the clipboard  
Select the required tab (SQL or VBA), then click the **Paste** button (or use **Ctrl+V**).

Once you have pasted in a query SQL or VBA statement, click the appropriate **Convert** button

**SQL to VBA converter - Add In** Close

☒ SQL
 ☒ VBA
 ☐ Expand SQL
 ☐ Include vbCrLf in VBA

```
UPDATE DISTINCTROW (MSysObjectsEXT INNER JOIN tblObjectTypes ON (MSysObjectsEXT.Flags = tblObjectTypes.Flags) AND (MSysObjectsEXT.Type = tblObjectTypes.Type)) INNER JOIN tblObjectInfo ON (tblObjectTypes.Object = tblObjectInfo.ObjType) AND (MSysObjectsEXT.Name = tblObjectInfo.ObjName) SET tblObjectInfo.TypeValue = [MSysObjectsEXT].[Type], tblObjectInfo.Flags = [MSysObjectsEXT].[Flags], tblObjectInfo.Hidden = [tblObjectTypes].[Hidden], tblObjectInfo.System = [tblObjectTypes].[System] WHERE (((tblObjectInfo.ObjName) NOT LIKE '~*') AND ((tblObjectInfo.ObjType) IS NOT Null) AND ((MSysObjectsEXT.Type) NOT In (2,3,8)) AND ((tblObjectTypes.DeepHidden)=False) AND ((tblObjectTypes.Attached)=False));
```

To use this add-in, COPY a valid query SQL or VBA statement to the clipboard, select the required tab then click the PASTE button (or use Ctrl+V)

Converting SQL to VBA will:

- Add strSQL = " at the start together with line continuations and other related items.
- Change quotes (") to double quotes (") leaving apostrophes (') unchanged e.g. around string values
- Capitalise all keywords e.g. Update => UPDATE, Group By => GROUP BY etc
- The VBA string created can be used in code or to make a temp query for testing

**SQL to VBA converter - Add In** Close

☒ SQL
 ☒ VBA
 ☐ Expand SQL
 ☐ Include vbCrLf in VBA

```
strSQL = "UPDATE DISTINCTROW (MSysObjectsEXT INNER JOIN tblObjectTypes ON (MSysObjectsEXT.Flags = tblObjectTypes.Flags) AND (MSysObjectsEXT.Type = tblObjectTypes.Type)) INNER JOIN tblObjectInfo ON (tblObjectTypes.Object = tblObjectInfo.ObjType) AND (MSysObjectsEXT.Name = tblObjectInfo.ObjName) SET tblObjectInfo.TypeValue = [MSysObjectsEXT].[Type], tblObjectInfo.Flags = [MSysObjectsEXT].[Flags], tblObjectInfo.Hidden = [tblObjectTypes].[Hidden], tblObjectInfo.System = [tblObjectTypes].[System] WHERE (((tblObjectInfo.ObjName) Not Like '~*') AND ((tblObjectInfo.ObjType) Is Not Null) AND ((MSysObjectsEXT.Type) Not In (2,3,8)) AND ((tblObjectTypes.DeepHidden)=False) AND ((tblObjectTypes.Attached)=False));"
```

**NOTE:**  
When running your SQL statements in VBA code:

- Use CurrentDb.Execute or DoCmd.RunSQL with action queries (Append, Update, Delete, Make Table). For example:  
CurrentDb.Execute "UPDATE tblSettings SET tblSettings.Version = 3.2, tblSettings.VersionDate = #6/27/2021#;", dbFailOnError
- For Select queries, you need to create and open a query definition. For example:  
Dim strSQL As String, qdf As DAO.QueryDef  
strSQL = "SELECT ID, Version, VersionDate FROM tblSettings;"  
Set qdf = CurrentDb.CreateQueryDef("MyTempQuery", strSQL) 'create temp QueryDef object  
DoCmd.OpenQuery "MyTempQuery" 'view the query  
CurrentDb.QueryDefs.Delete "MyTempQuery" 'delete the temp query
- Click the Code from VBA button to create a temp query and example VBA code

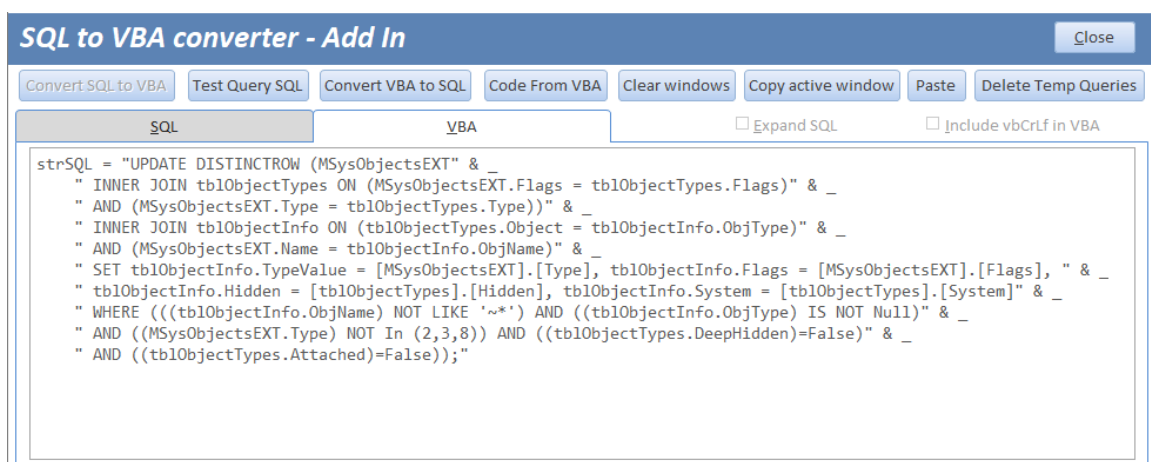
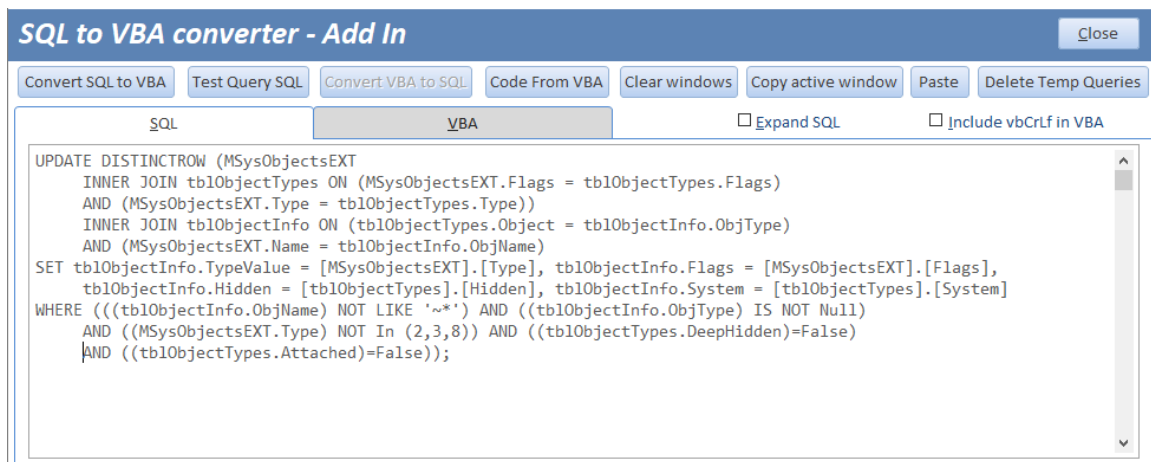
SQL to VBA converter Add-In : Version 3.41 2024-07-10 Mendip Data Systems 2005-2024

Converting VBA to SQL will:

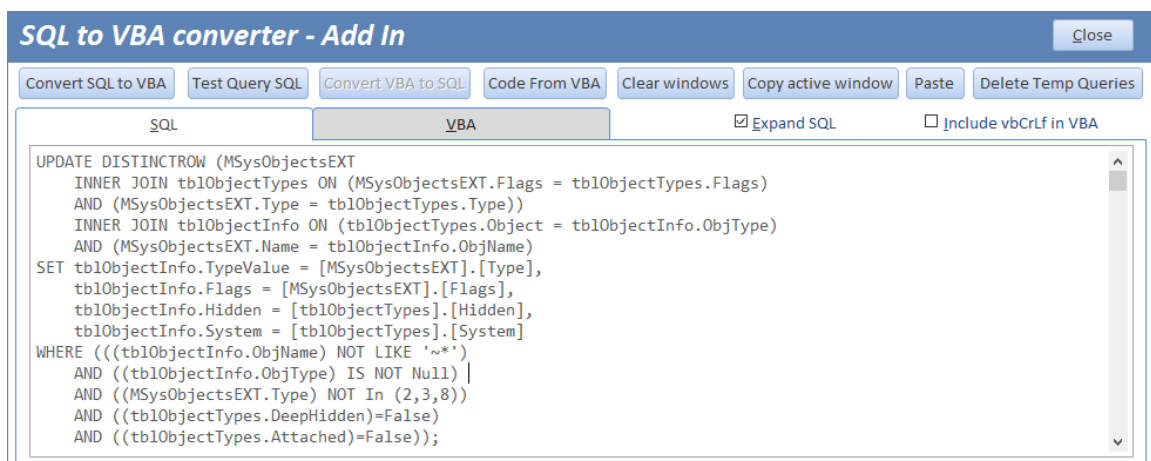
- Strip out all superfluous items e.g. strSQL = , leading and trailing quotes && line continuations
- Convert double quotes (") to quotes (") leaving apostrophes (') unaltered
- The resulting SQL can be used to create a query in SQL or design view

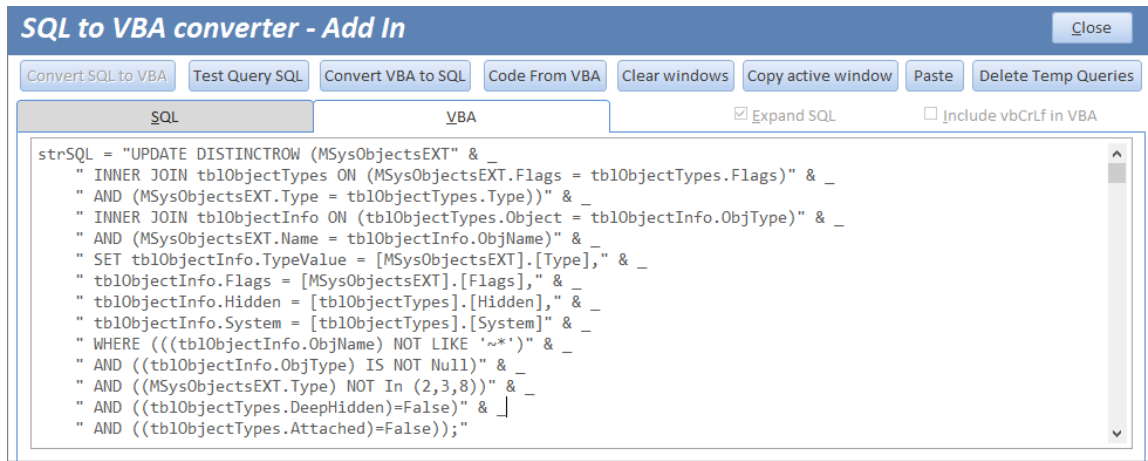
The output from each conversion should ALWAYS be tested using the **Test Query SQL / Code From VBA** buttons

11. You can use **Ctrl+Enter** to add line breaks at selected positions in your **query SQL**.  
The line breaks will be retained on conversion in both directions

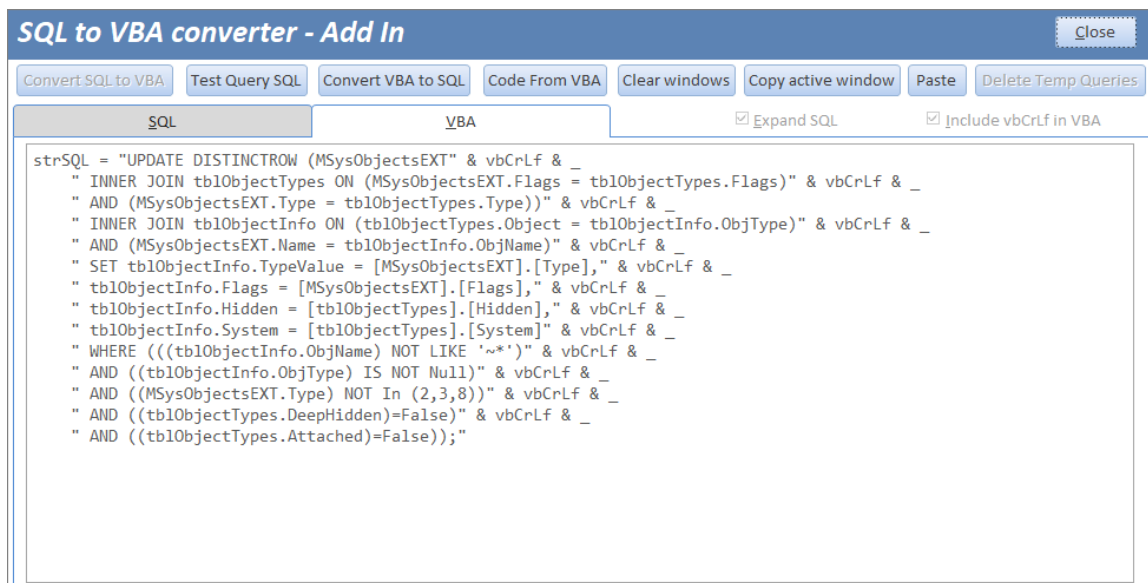


12. Alternatively, when the SQL tab is open, tick the **Expand SQL** checkbox to add a line break after each field.



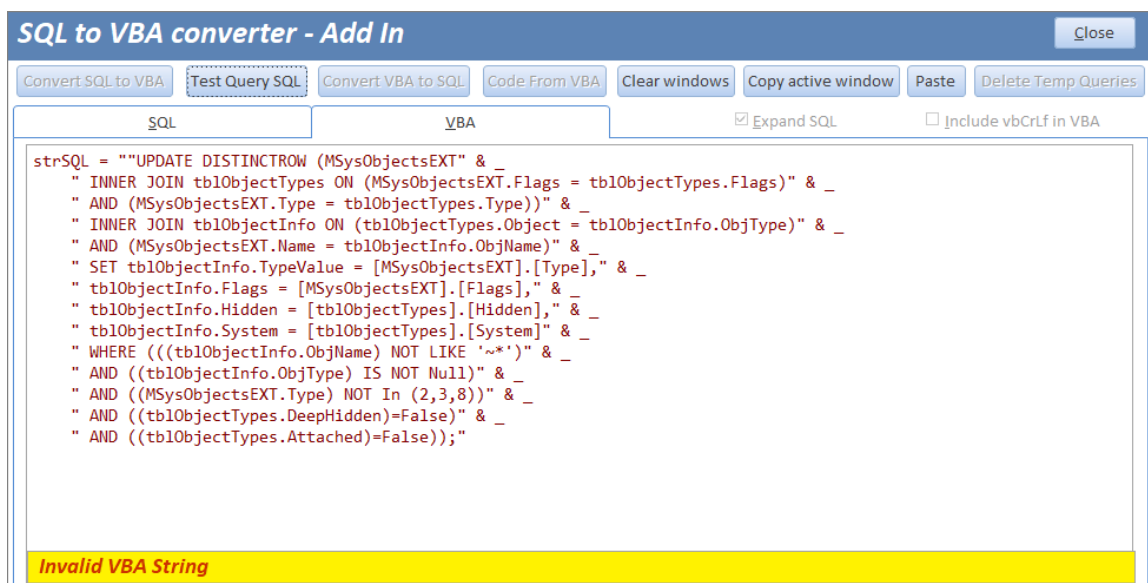


13. You can also opt to **include vbCrLf in VBA** statement before each line break

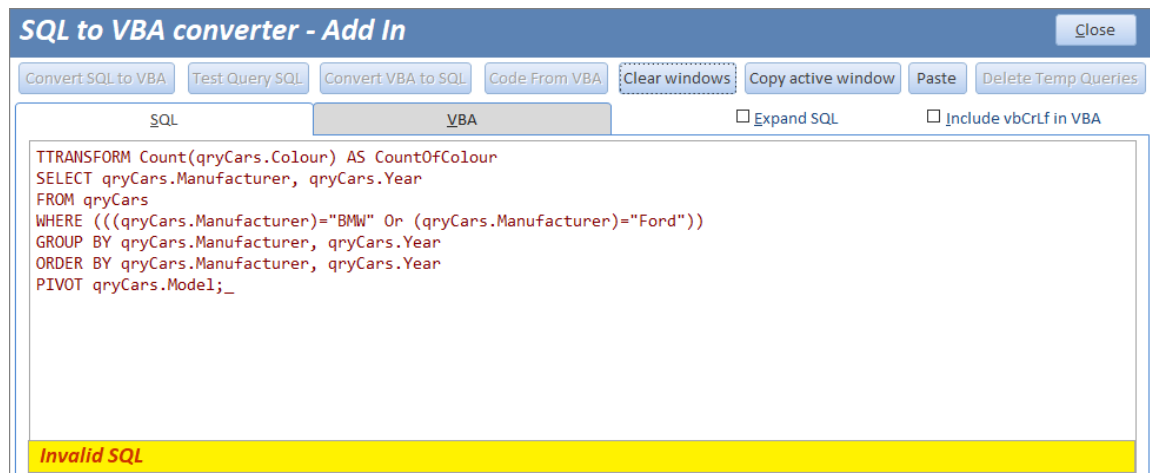


14. If there are SQL or VBA errors, the text string will be marked in RED.  
The errors MUST be corrected before you can test the output or run the conversion

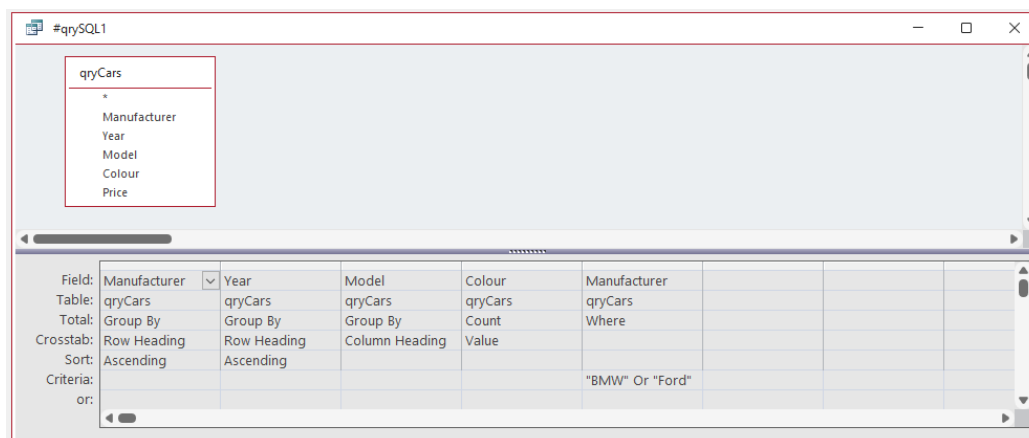
For example, there are 2 initial quotes at the start of the VBA statement below:



In the SQL statement below, the TRANSFORM keyword is spelt incorrectly and there is an extra character after the end semicolon



15. Click the **Test Query SQL** button to create a temp query in your app for testing. The query will open in design view (except for SQL-only queries such as UNION, DDL etc)



For easy identification, the temp queries from SQL are named #qrySQL1, #qrySQL2 etc.

16. If the host application is an **ACCDB** file, click the **Code From VBA** button to create an example procedure in a new module **modExampleVBACode** in the host application.

The procedure will be called e.g. **ExampleCode\_qryVBA1**.

```
Sub ExampleCode_qryVBA1()
```

```
'=====
' Example code created by the SQL2VBA converter add-in
' Author: Colin Riddington (Mendip Data Systems)
' Date: 12/07/2024
' Copy & paste this code into your app. Modify as appropriate
' OR reference it in your own procedure
'=====
```

```
Dim db As DAO.Database
Dim qdf As DAO.QueryDef
Dim strSQL As String
```

```

' define the VBA string
strSQL = "SELECT MSysObjects.Name AS QueryName, tblSysObjectTypes.SubType AS
QueryType, " & _
" GetQueryLastSavedView([Name]) AS LastSavedView, GetDateLastUpdated([Name]) AS
LastSavedDate" & _
" FROM MSysObjects INNER JOIN tblSysObjectTypes ON (MSysObjects.Flags =
tblSysObjectTypes.Flags)" & _
" AND (MSysObjects.Type = tblSysObjectTypes.Type)" & _
" WHERE (((MSysObjects.Flags)<>3) AND ((MSysObjects.Type)=5))" & _
" ORDER BY MSysObjects.Name;"

' Set the database to the current database
Set db = CurrentDb

' Debug.Print Left(strSQL, 6)

' Open/execute temp query
If Left(strSQL, 6) = "SELECT" Then
    ' Close & delete existing temp query if it exists
    On Error Resume Next
    DoCmd.Close acQuery, "~qryVBA1"

    db.QueryDefs.Delete "~qryVBA1"
    On Error GoTo 0

    ' Create a new temp query
    Set qdf = db.CreateQueryDef("~qryVBA1", strSQL)

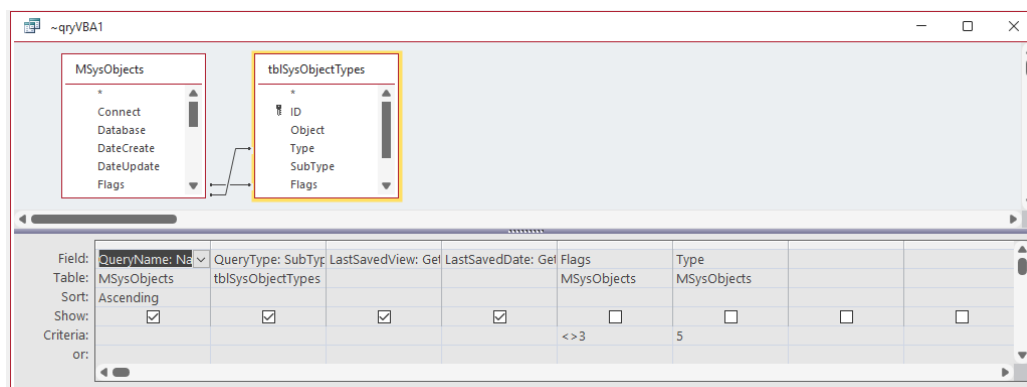
    ' Open the query in design view
    DoCmd.OpenQuery "~qryVBA1", acViewDesign
Else
    db.Execute strSQL, dbFailOnError
End If

' Clean up
Set qdf = Nothing
Set db = Nothing

End Sub

```

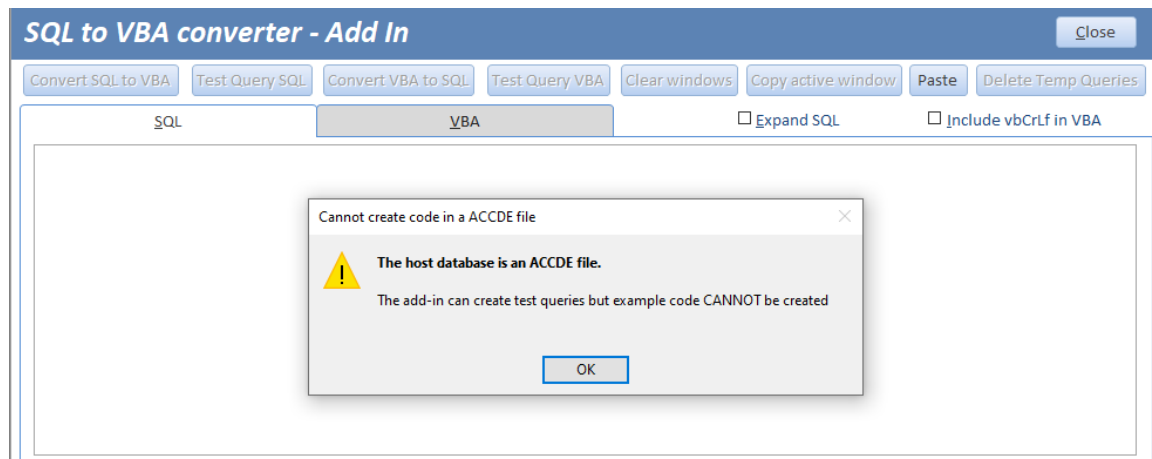
A temp query **~qryVBA1** will also be created.



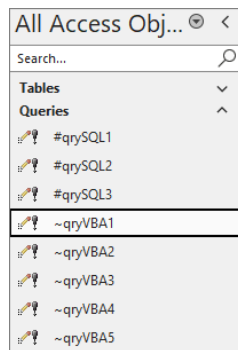
If the code is run, the temp query ~qryVBA1 will be over written.  
Temp queries from VBA are called ~qryVBA1, ~qryVBA2 etc

**NOTE:**

If the host application is an **ACCDE** file, you cannot create **new code**, but you can still create temp queries for testing. The button caption is now labelled **Test Query VBA**.



17. Click the **Delete Temp Queries** button to delete all temp queries when no longer needed



Before doing so, rename any temp queries that you want to keep.

18. You can also **copy the active window** e.g. for pasting the VBA into your own code in the **Visual Basic Editor**
19. Click **Clear Windows** to clear both the SQL and VBA tabs and empty the clipboard
20. If you close the **SQL2VBA** form at any time, just select it from the list of Access add-ins to re-open it.

---

### Acknowledgements:

I would particularly like to thank the following developers who assisted in the development of this utility

- **Allen Browne** – for providing the original version of the [SQL to VBA converter](#)
- **Dale Fye** - for providing code which I used to help convert this utility to an **Access add-in**
- **Michael Villegas** – for testing different versions of the standalone and add-in versions and suggesting various additional features

---

If you find this **Access add-in** useful as an ACCDE file, please consider giving a **donation** to help offset the costs of maintaining my website. Thanks

You can also purchase the **source code** from my website at a reasonable cost.

This will also allow you to make further modifications to the add-in app if you wish to do so.

Colin Riddington

Mendip Data Systems

<https://isladogs.co.uk>

18 July 2024