

<http://www.sqlservercentral.com/articles/Stored+Procedures/63537/>

Printed 2008/08/13 07:28AM

## Excel with Stored Procedures

By [David Poole](#), 2008/06/25

## Excel with SQL Stored Procs

### Introduction

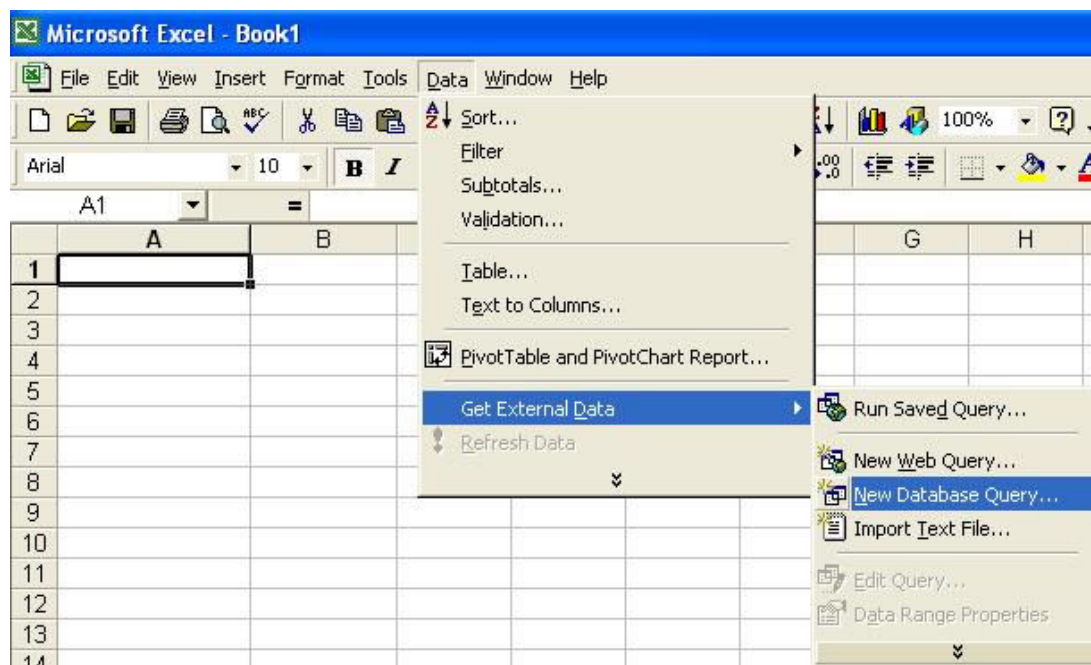
A colleague asked how I felt about making a user a member of db\_datareader on a server used by non-DBAs for reporting data. The server exists precisely to allow people to write ad-hoc queries without threatening live systems.

He then went on to say that they were using Microsoft Excel to query the data and that this was in the requirements for the project he was working on. He was asking the question because his technical specification said that Excel would query the data by calling stored procedures but the end user said that this wasn't possible without writing reams of VBA macros.

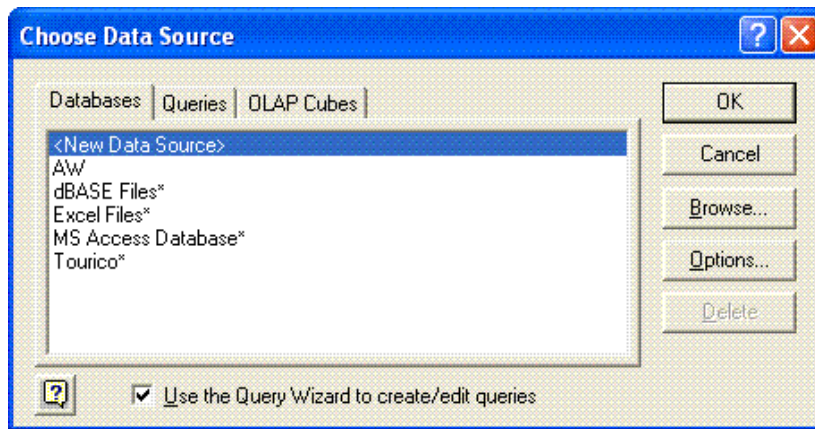
Using Microsoft Excel is a bit off my radar but I thought I would investigate. I'm using Excel 2000, it works, I'm comfortable with it, it does everything I want so why upgrade?

### Basic queries with Excel

SQL queries are classed as external data so our first port of call is to use the commands on the **Data** menu as shown below.



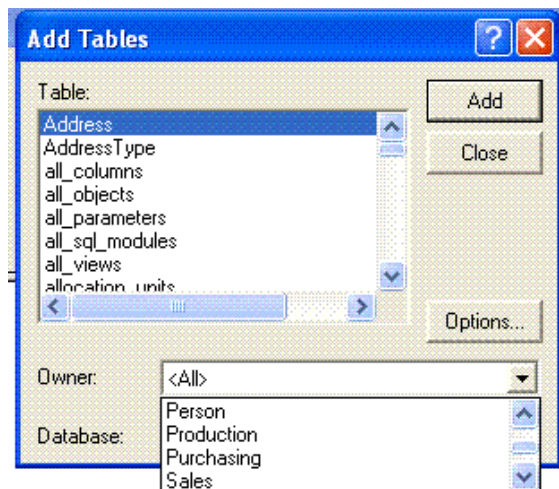
Choosing this option will then present you with a dialogue box that will allow you to set up a new ODBC connection or use an existing one.



I set up my AW data source to look at the AdventureWorks database so I chose that to start my experiments.

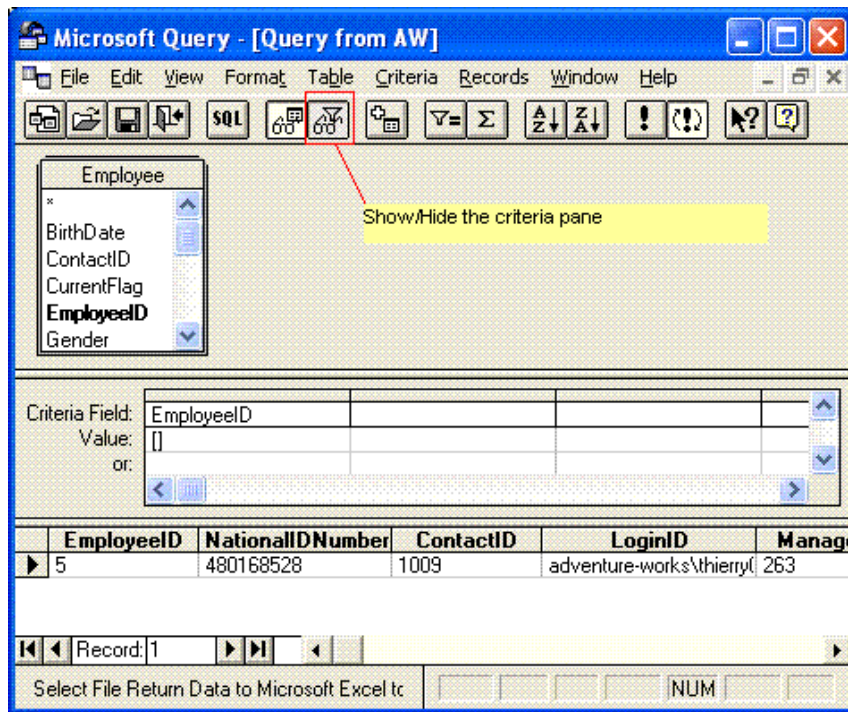
This takes you into a wizard that only shows you objects owned by dbo and of course Adventureworks has most of its objects placed on other schemas so I clicked cancel and got a popup dialogue asking "Do you want to continue editing this query in Microsoft Query" to which I answered YES.

On doing so the Microsoft Query application will open with the dialogue to select the tables. The Owner drop down box will list the schemas.



I chose HumanResources and the Employee table and dragged the \* onto the results grid.

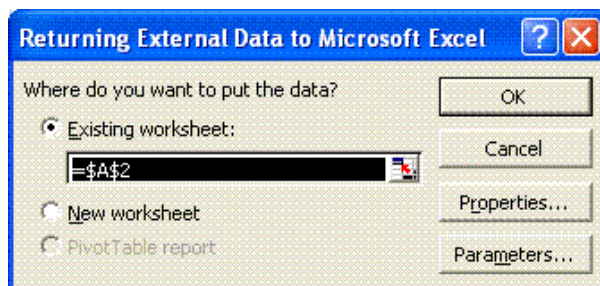
On the criteria grid I dragged EmployeeID and on the value line I added a question mark to indicate that I wanted this to be a parameter. This is shown in the screen shot below



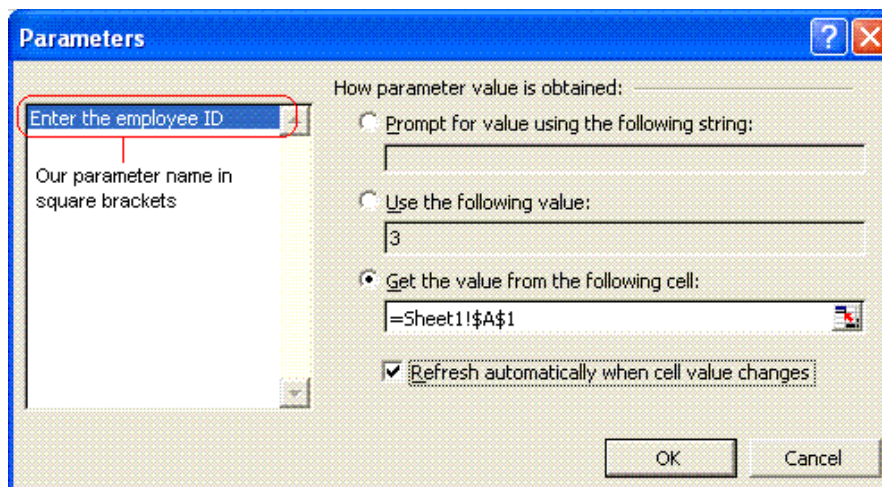
Whenever I run the query I am prompted to enter the value for EmployeeID that I want.

From my MS Access 2.0 days I remembered that could actually put a string within square brackets so I change the value entry to [Enter the employee ID]

From the MS Query **F**ile menu I chose "Save and return data to Microsoft Excel" to get the dialogue box shown below.



Note that I want the data to be returned to the 2<sup>nd</sup> row in my worksheet hence my choice of cell A2 to receive the data. I also want Excel to be able to feed the parameters into the query so I clicked the **P**arameters button to receive the dialogue shown below.

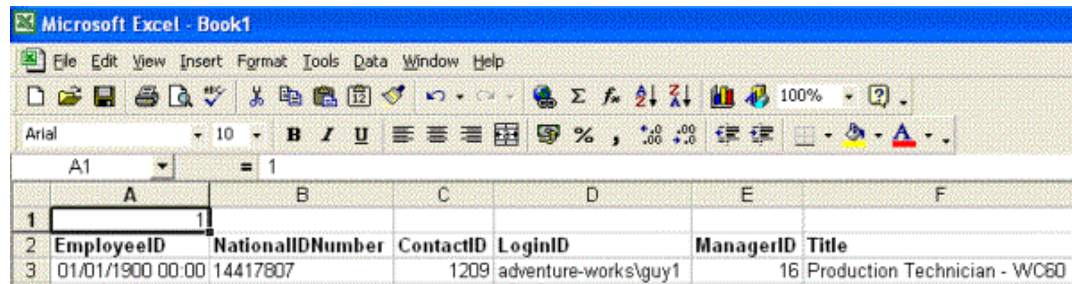




There are three things to note here.

- The text we entered within the square brackets in Microsoft Query appears as the name of the parameter.
- We have the facility to specify the cell that will be used to feed in values into the query
- We have a check box to refresh the query automatically if that cell is changed.

Clicking OK on this dialogue and on the preceding dialogue took me back to Excel and after filling in an employee ID in the top left cell I saw results similar to those shown below



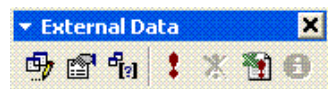
The screenshot shows the Microsoft Excel interface with a table of query results. The table has columns: EmployeeID, NationalIDNumber, ContactID, LoginID, ManagerID, and Title. The first row of data shows values for EmployeeID 1, NationalIDNumber 14417807, ContactID 1209, LoginID adventure-works\guy1, ManagerID 16, and Title Production Technician - WC60.

	A	B	C	D	E	F
1	1					
2	EmployeeID	NationalIDNumber	ContactID	LoginID	ManagerID	Title
3	01/01/1900 00:00	14417807	1209	adventure-works\guy1	16	Production Technician - WC60

Changing the contents of cell \$A\$1 will automatically change the resultset.

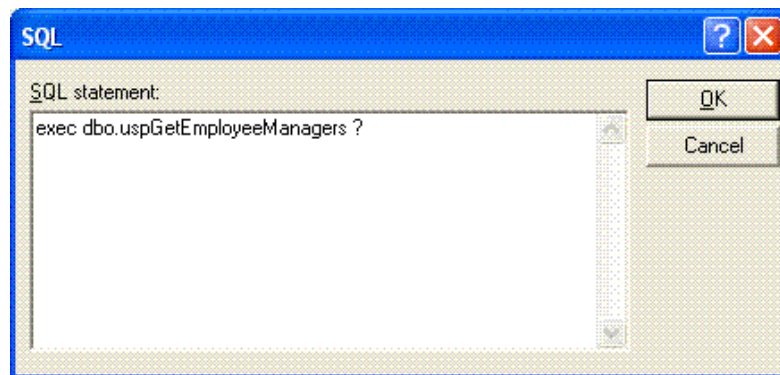
## Making the switch to stored procedures

I remember that Access 2.0 had pass thru queries so surely MS Query would be able to do this as well? A pass thru query is one where the client application does not parse the SQL it just assumes it is correct and passes it through to the server to allow the server to worry about it.

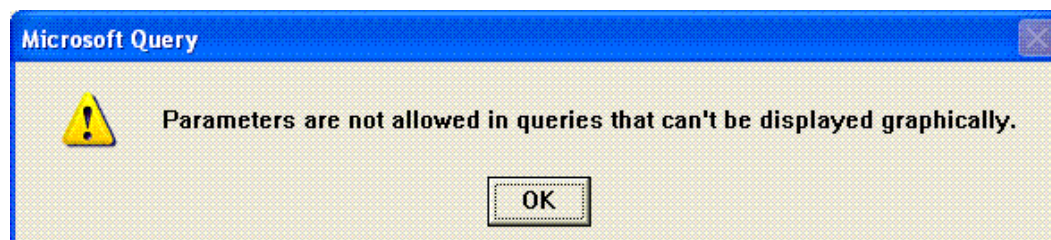


The left button allows us to edit the query. On clicking the button you would be faced with a warning dialogue telling you that the query cannot be edited in the MS Query wizard which is just a confusing distraction. Clicking on OK takes you back into the MS Query.

I couldn't see an option for pass-thru queries so I tried editing the SQL directly. I decided to use the built in AdventureWorks stored procedure dbo.uspGetEmployeeManagers which accepts an EmployeeID parameter.



Click on OK and you are due for a disappointment.



OK so this route is blocked so it looks like some coding is necessary after all but how onerous will it be?

## A little bit of VBA

Back in the days of Office 97 I found that the easiest way of digging into the MS Office object model was to choose to record a new macro, perform whatever tasks I want the macro to do and then edit the resulting VBA Code.

The data that is returned from external data sources is an object called a "QueryTable".

The crude code below shows my eventual edited macro.

```
Sub RefreshSheet()

'
' RefreshSheet Macro
' Macro recorded 05/06/2008 by David Poole
'

    Set qt = Sheets("sheet1").QueryTables(1)
    qt.Sql = "exec uspGetEmployeeManagers ?"
    Set param1 = qt.Parameters("Enter the employee ID")

' The equivalent of the "Get the value from the following cell" portion of the parameters dialogue.
    param1.SetParam xlRange, Range("sheet1!a1")

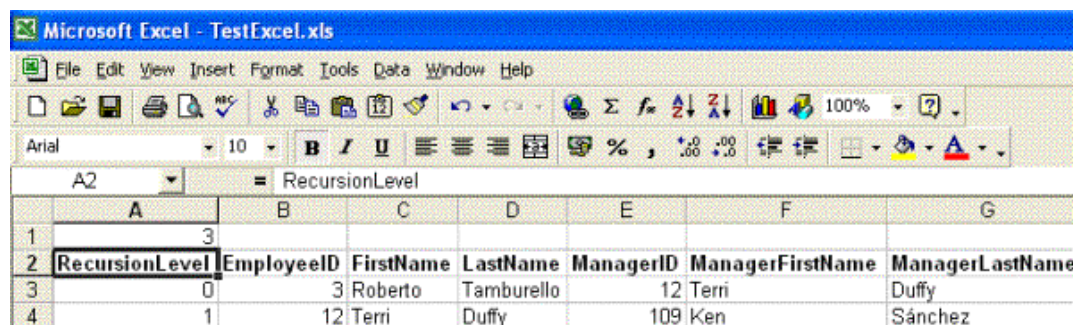
' The equivalent of the check box "Refresh automatically when cell value changes".
    param1.RefreshOnChange = True
    qt.Refresh

' If you instantiate objects destroy them afterwards.
    Set param1 = Nothing
    Set qt = Nothing
End Sub
```

Strictly speaking the macro could be boiled down to a single line of code.

`Sheets("sheet1").QueryTables(1).Sql = "exec uspGetEmployeeManagers ?".`

The resulting spreadsheet looks similar to the one shown below



	A	B	C	D	E	F	G
1	3						
2	<b>RecursionLevel</b>	<b>EmployeeID</b>	<b>FirstName</b>	<b>LastName</b>	<b>ManagerID</b>	<b>ManagerFirstName</b>	<b>ManagerLastName</b>
3	0	3	Roberto	Tamburello	12	Terri	Duffy
4	1	12	Terri	Duffy	109	Ken	Sánchez

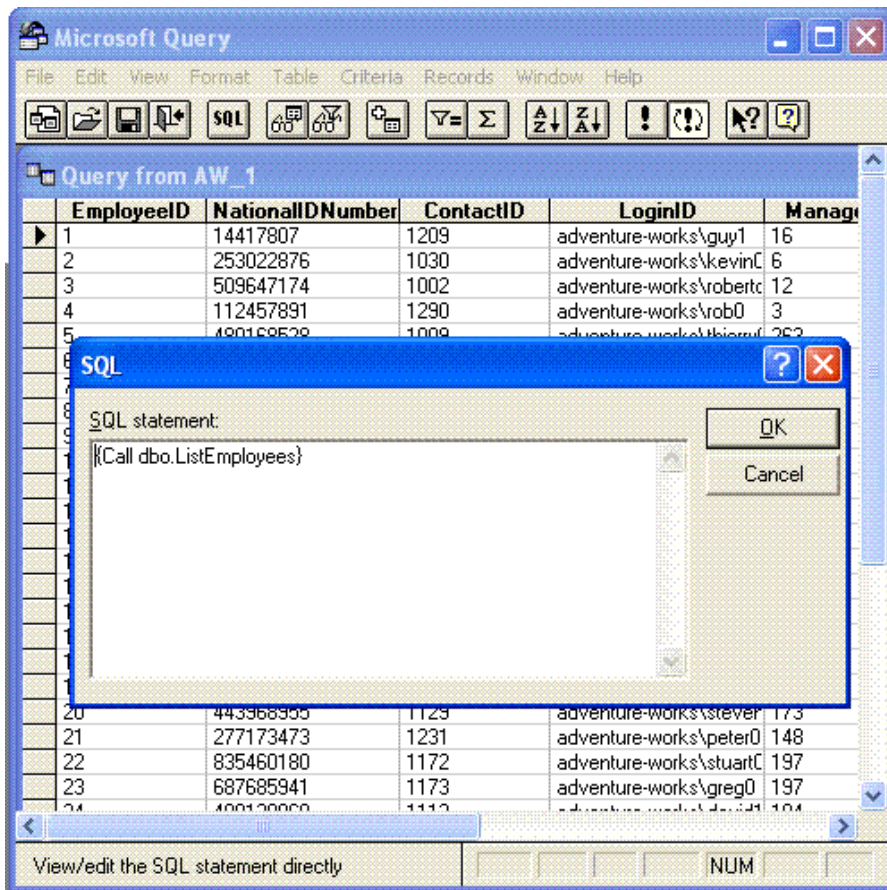
Altering the contents of cell \$A\$1 automatically refreshes the query.

## Stored procedures without VBA

So far I have established that I have to use VBA, albeit a single line of VBA in order to use parameterised stored procedures but what about non-parameterised stored procedures.

I wrapped up my basic `SELECT * FROM HumanResources.Employee` into a stored procedure called `dbo.ListEmployees`.

Microsoft Query can call a non-parameterised stored procedure simply by wrapping up the call to the stored procedure in curly brackets `{CALL dbo.ListEmployees}` as shown below



## Conclusion

I don't know if Microsoft Query remains as crude as it was in Office 2000 but I have proved that using stored procedures with Excel does not require great programming skills. Anyone who can write macros should be able to do it.

I don't want to detract from my end user's flexibility but I would push them towards using stored procedures in future. By all means model the query to start off with, but once your model query is working get it put into a stored procedure.

I would say that stored procedures offer the following benefits to Excel users

- Excel spreadsheets are far more likely to corrupt than SQL Server databases so a lovingly crafted query won't go missing if it is recorded as a stored procedure.
- The complexity of the query is hidden inside the stored procedure. In Adventureworks the `uspGetEmployeeManagers` stored procedure uses a common table expression, which is vastly more complex than anything Microsoft Query could do, and yet the resulting data is available to Excel.
- Microsoft Query generated SQL is verbose and therefore hard to read and maintain. If your network is busy and you are asking to pass a huge block of SQL one way and get a block of data coming back you may get frustrated with the response times you get. A stored procedure call is vastly more compact than the raw SQL.
- A DBA may or may not know Excel VBA macros but they will know SQL. Give them a SQL problem and they will roll up their sleeves and get stuck in leaving you to concentrate on the results.