Using Dynamic Cell References in ActiveFactory Workbook

Legacy Tech Note

535

SUMMARY

This *Tech Note* describes the process of including dynamic cell references within ActiveFactory Workbook's Direct Query functionality by using Microsoft Excel's built-in **=CONCATENATE()** function.

SITUATION

Application Versions

- ActiveFactory 8.x, 9.x
- IndustrialSQL Server Historian 8.x, 9.x

Note: This Tech Note assumes familiarity with Wonderware ActiveFactory, ActiveFactory Workbook, and Microsoft Excel.

Summary

ActiveFactory Workbook's **=wwQuery()** function (aka **Direct Query**) is great for making flexible Transact-SQL queries that may not be easily accomplished through the standard menu options. The second parameter of this function is a reference to an Excel cell containing text representing the Transact-SQL (T-SQL) script to be executed.

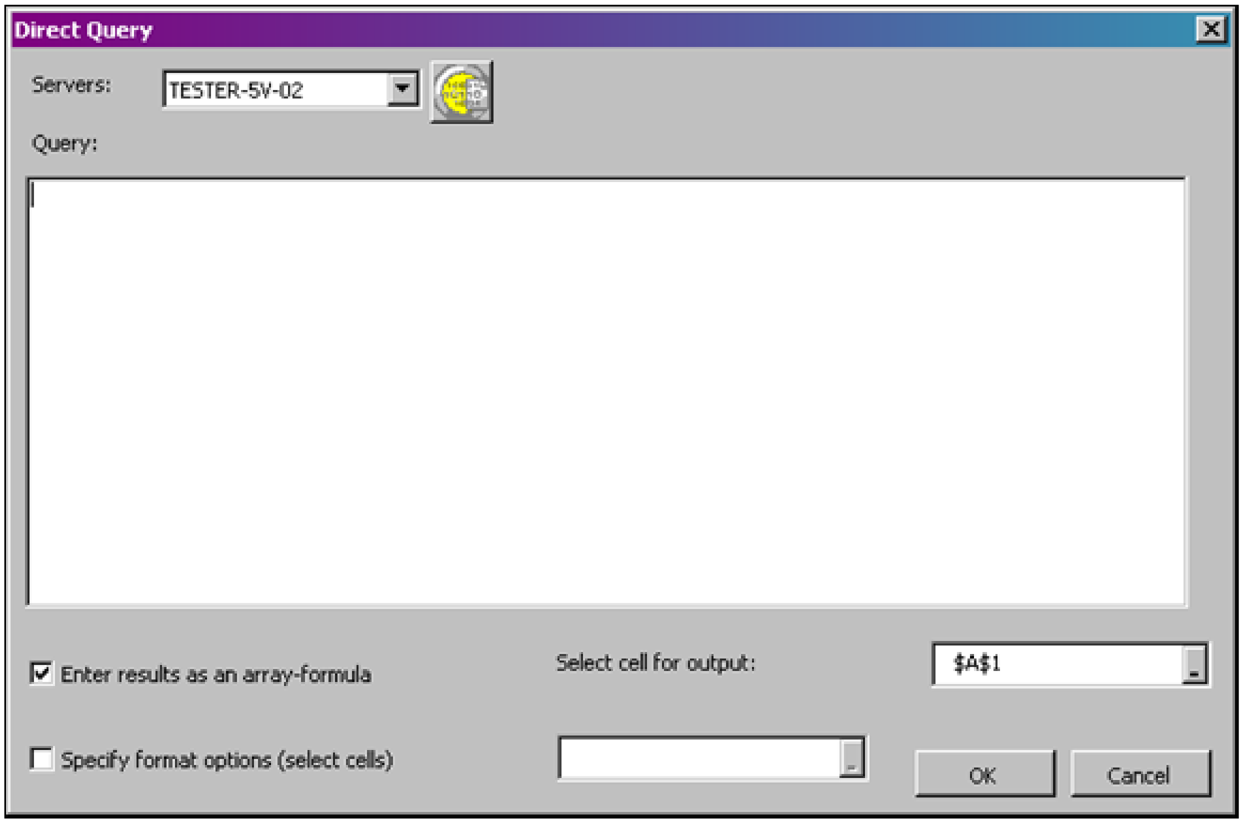
Microsoft Excel includes a function called =CONCATENATE() which combines cell references and literal text values into a single entity. Excel also has a function called =TEXT() which can be used to translate datetime values from other cells into literal text values.

If you point the second parameter of the =wwQuery() function to a cell containing a =CONCATENATE() function, you can apply dynamic information from other cell references quite easily.

Basic Use of wwQuery()

This example will generate a typical =wwQuery() function using the Direct Query function wizard, and then later you can manipulate the result to extend its functionality.

1. On the ActiveFactory menu of Excel, click **Direct Query**. The **Direct Query** editor appears.


Direct Query
Servers:
Query:
Enter results as an array-formula
Specify format options (select cells)
OK
Cancel

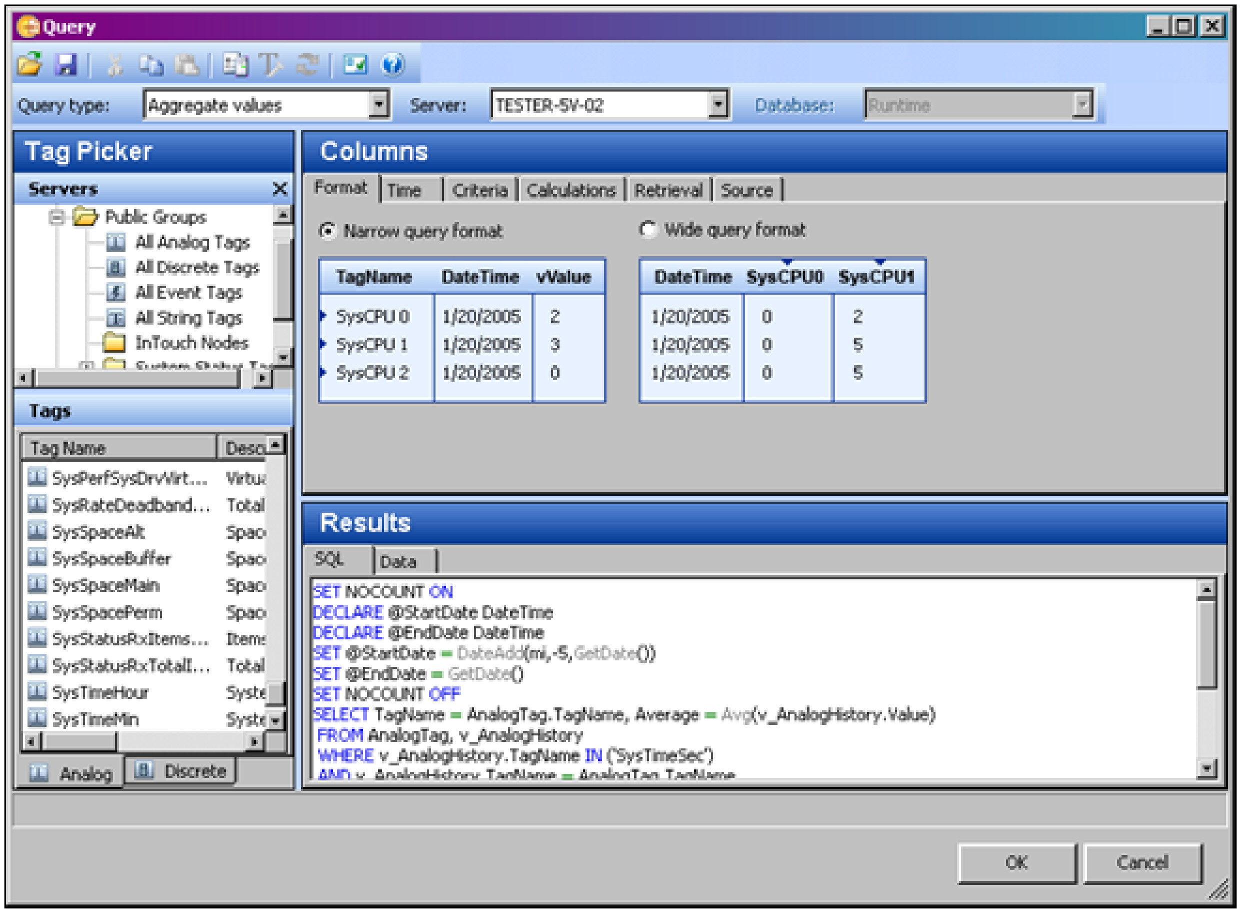
Servers: TESTER-5V-02

Select cell for output: A1

Figure 1: Direct Query Editor Window

2. In the Query pane, type in a custom query, or create a query using the ActiveFactory Query client tool.

   Clicking on the **Query** button opens the Query client to help design the query:



Query type: Aggregate values Server: TESTER-SV-02 Database: Runtime

(●) Narrow query format ( ) Wide query format

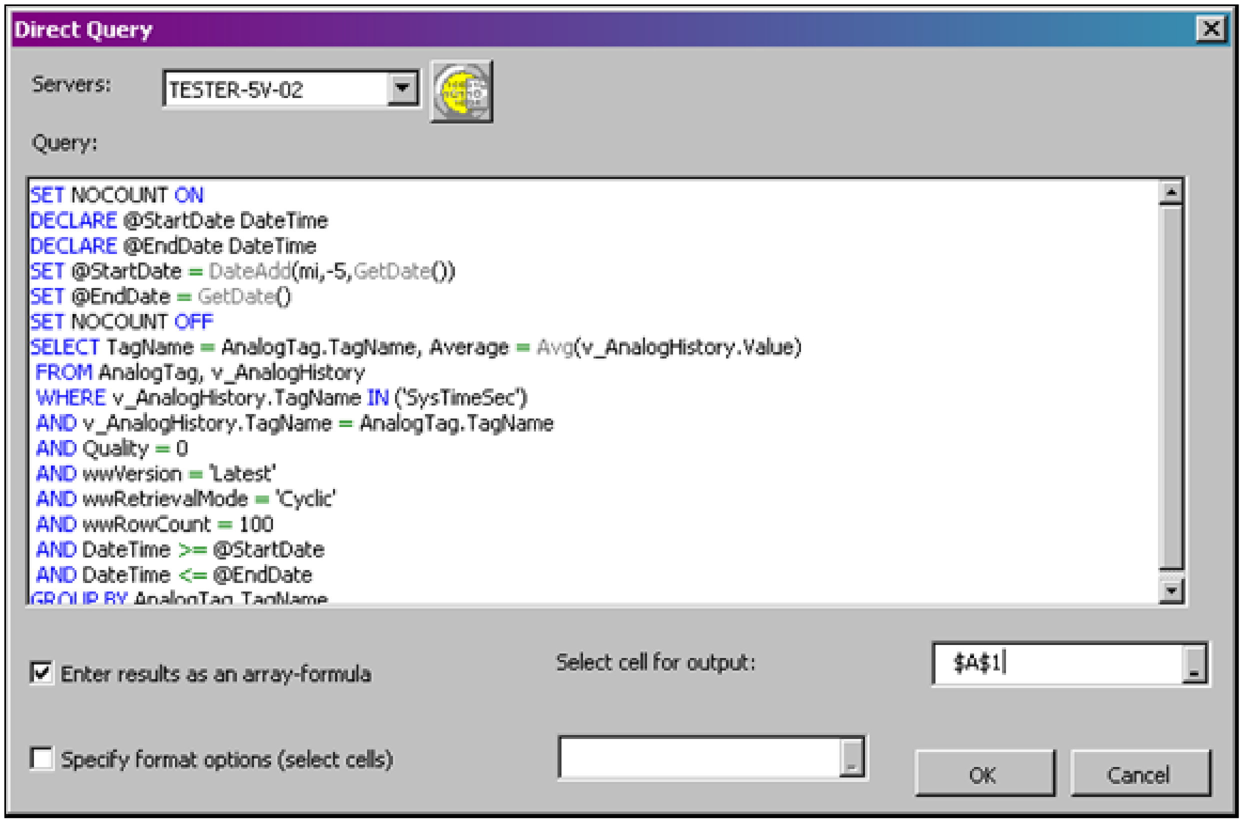
TagName	DateTime	vValue
SysCPU 0	1/20/2005	2
SysCPU 1	1/20/2005	3
SysCPU 2	1/20/2005	0

DateTime	SysCPU0	SysCPU1
1/20/2005	0	2
1/20/2005	0	5
1/20/2005	0	5

```
SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = DateAdd(mi,-5,GetDate())
SET @EndDate = GetDate()
SET NOCOUNT OFF
SELECT TagName = AnalogTag.TagName, Average = Avg(v_AnalogHistory.Value)
 FROM AnalogTag, v_AnalogHistory
 WHERE v_AnalogHistory.TagName IN ('SysTimeSec')
```

Figure 2: ActiveFactory Query

3. Enter a cell reference for the results in the **Select cell for output** field.



Servers: TESTER-SV-02

Query:

```
SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = DateAdd(mi,-5,GetDate())
SET @EndDate = GetDate()
SET NOCOUNT OFF
SELECT TagName = AnalogTag.TagName, Average = Avg(v_AnalogHistory.Value)
 FROM AnalogTag, v_AnalogHistory
 WHERE v_AnalogHistory.TagName IN ('SysTimeSec')
 AND v_AnalogHistory.TagName = AnalogTag.TagName
 AND Quality = 0
 AND wwVersion = 'Latest'
 AND wwRetrievalMode = 'Cyclic'
 AND wwRowCount = 100
 AND DateTime >= @StartDate
 AND DateTime <= @EndDate
GROUP BY AnalogTag.TagName
```

[✓] Enter results as an array-formula

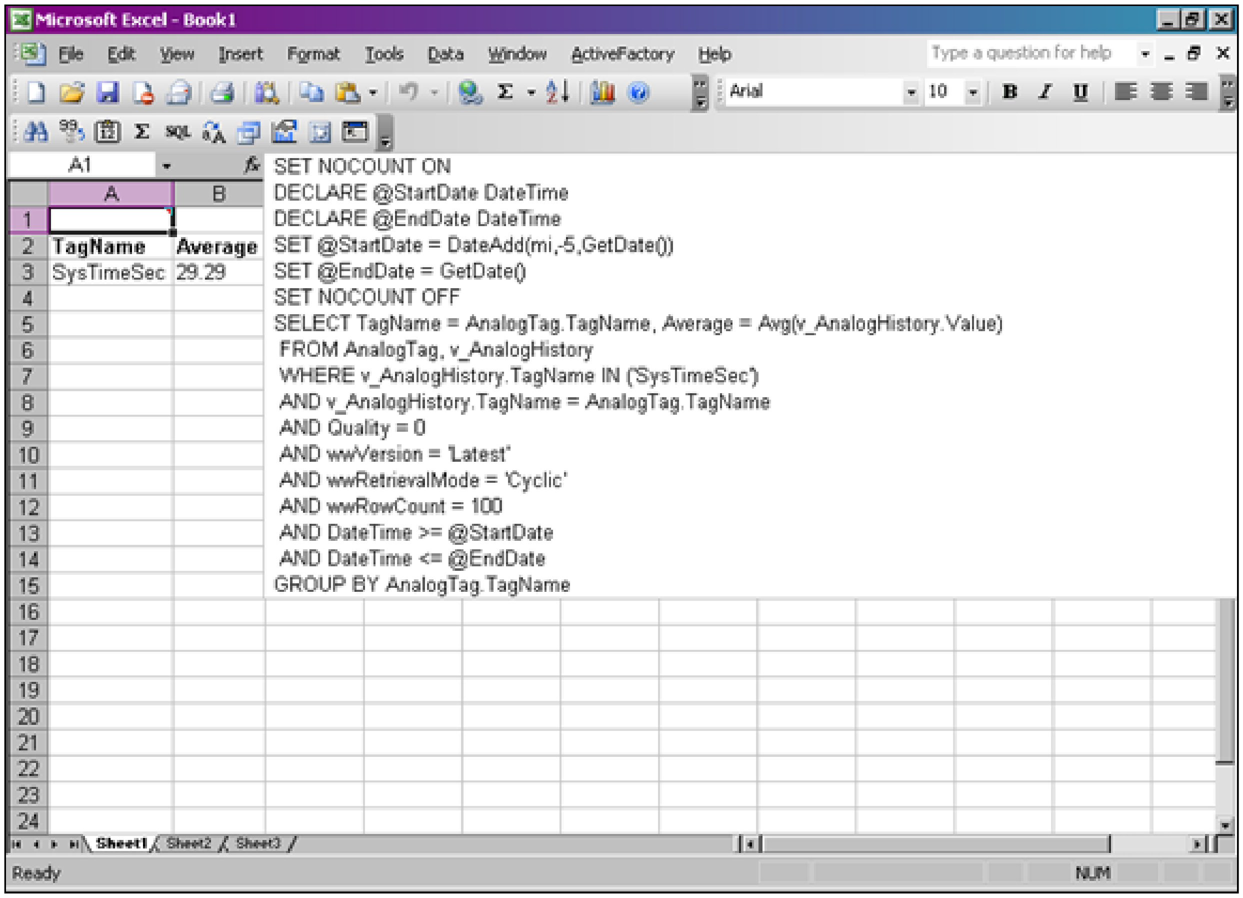
Select cell for output: A1

[ ] Specify format options (select cells)

Figure 3: Cell Reference and Ouput Field

4. Click **OK**.
5. The results are placed two cells below the output cell because the function **=wwQuery("InSQLServerName", A1)** indicates that the T-SQL query is embedded in cell A1.
6. The text of the A1 cell is automatically formatted as a white font in order to appear invisible on a white background; there is also a comment indicator in the cell containing the query text. Clicking cell A1 will show the query text in the formula bar (Figure 4 below).

   The contents of Cell A1 can be edited at any time and the function refreshes automatically.



```
SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = DateAdd(mi,-5,GetDate())
SET @EndDate = GetDate()
SET NOCOUNT OFF
SELECT TagName = AnalogTag.TagName, Average = Avg(v_AnalogHistory.Value)
 FROM AnalogTag, v_AnalogHistory
 WHERE v_AnalogHistory.TagName IN ('SysTimeSec')
 AND v_AnalogHistory.TagName = AnalogTag.TagName
 AND Quality = 0
 AND wwVersion = 'Latest'
 AND wwRetrievalMode = 'Cyclic'
 AND wwRowCount = 100
 AND DateTime >= @StartDate
 AND DateTime <= @EndDate
GROUP BY AnalogTag.TagName
```

	A	B
1		
2	TagName	Average
3	SysTimeSec	29.29

Figure 4: Hidden Cell Contents Displaying T-SQL Query

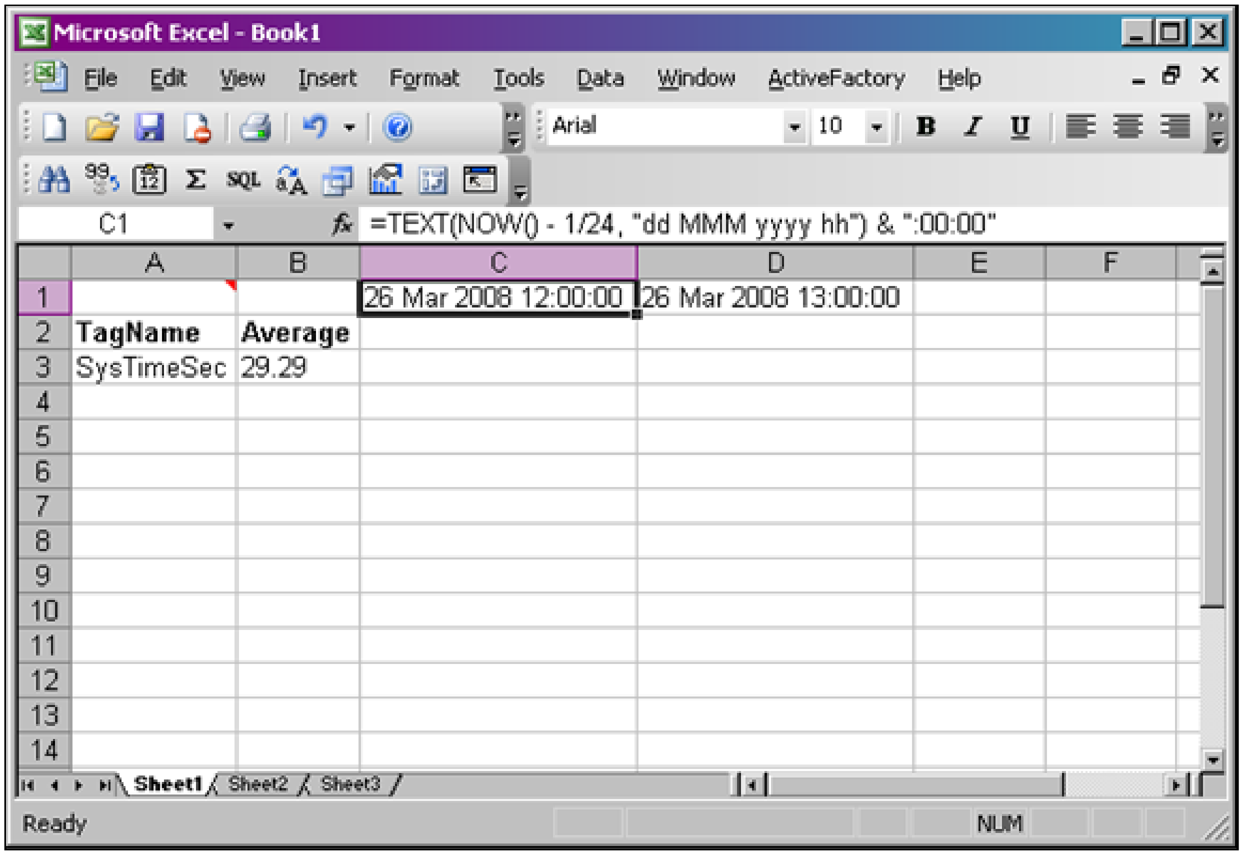
Editing wwQuery() to Use Concatenation

Now let's modify cell A1 to build the query based on contents of other cells containing the start and end dates. Once the query has been modified, you will only need to type in the start and end times to get new results.

1. Start by inserting start and end times in cells **C1** and **D1**. In this example these are built using the **=TEXT()** function to specify the top of the current hour for the end time and the top of the previous hour as the start time.

   You can also use cell references in place of the NOW() function when you need to enter dates manually.

   Start Time: =TEXT(NOW()-1/24,"dd MMM yyyy hh") & ":00:00"
   End Time: = TEXT(NOW(),"dd MMM yyyy hh") & ":00:00"


Microsoft Excel - Book1
File Edit View Insert Format Tools Data Window ActiveFactory Help
Arial 10
Sheet1 Sheet2 Sheet3
Ready
NUM

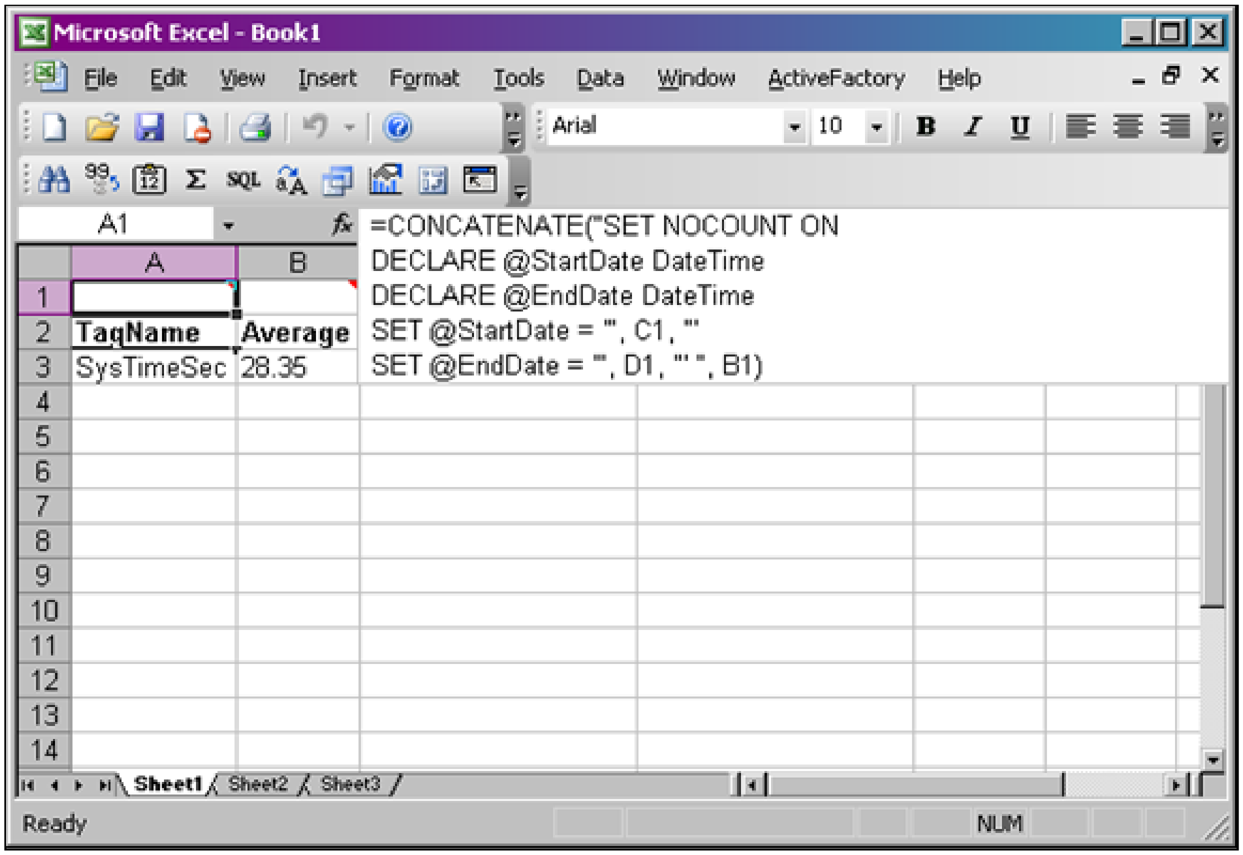
C1 =TEXT(NOW() - 1/24, "dd MMM yyyy hh") & ":00:00"

	A	B	C	D
1			26 Mar 2008 12:00:00	26 Mar 2008 13:00:00
2	TagName	Average		
3	SysTimeSec	29.29		

Figure 5: Start Time and End Time Formula Examples

2. Modify the contents of cell A1 to use a =CONCATENATE() function to build the query using the times in cells C1 and D1. In this example, the query itself is too long to fit within Excel's limitations for the function, so we move the last part of the original query into cell B1. The resulting function in A1 becomes:

   ```
   =CONCATENATE("SET NOCOUNT ON
   DECLARE @StartDate DateTime
   DECLARE @EndDate DateTime
   SET @StartDate = '", C1, "'
   SET @EndDate = '", D1, "' ", B1)
   ```

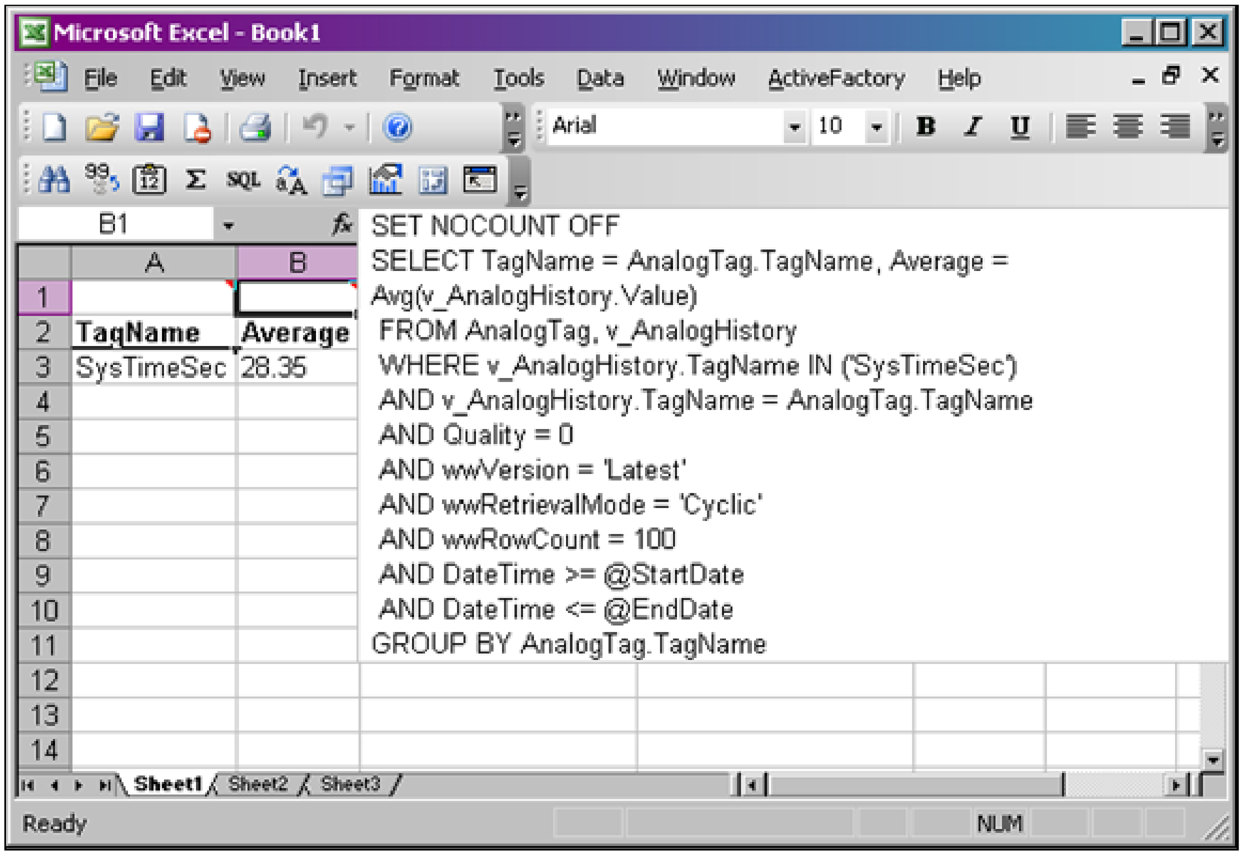


```
=CONCATENATE("SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = "', C1, "'
SET @EndDate = "', D1, "' ", B1)
```

	A	B
1		
2	TagName	Average
3	SysTimeSec	28.35

Figure 6: Cell A1 Edited with CONCATENATE Function

For cell B1, the query looks like the following:

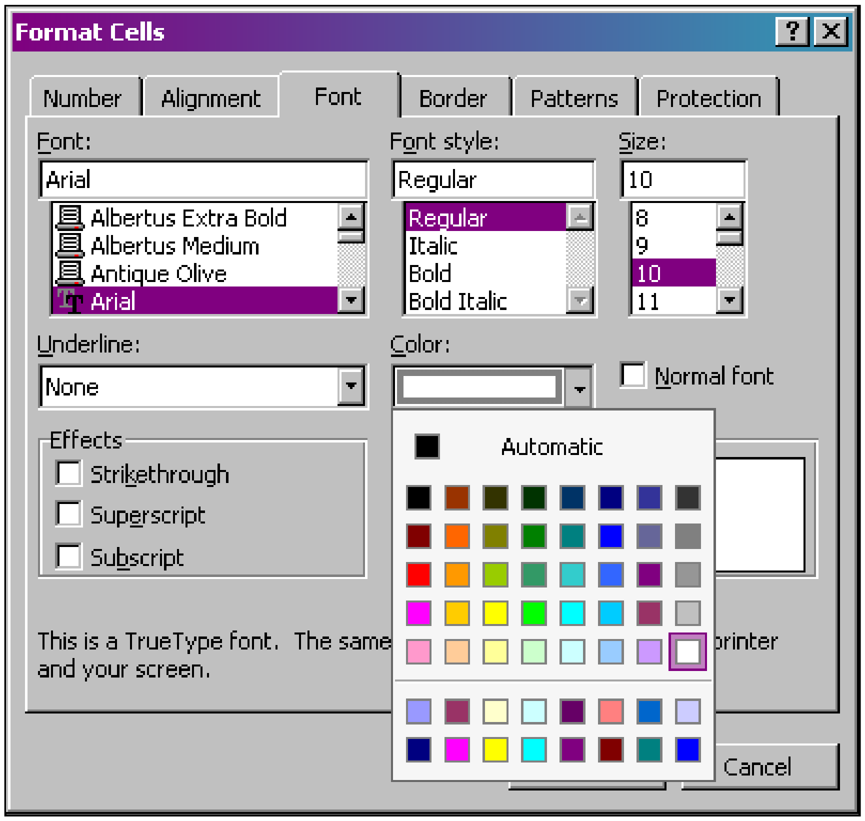


```
SET NOCOUNT OFF
SELECT TagName = AnalogTag.TagName, Average =
Avg(v_AnalogHistory.Value)
 FROM AnalogTag, v_AnalogHistory
 WHERE v_AnalogHistory.TagName IN ('SysTimeSec')
 AND v_AnalogHistory.TagName = AnalogTag.TagName
 AND Quality = 0
 AND wwVersion = 'Latest'
 AND wwRetrievalMode = 'Cyclic'
 AND wwRowCount = 100
 AND DateTime >= @StartDate
 AND DateTime <= @EndDate
GROUP BY AnalogTag.TagName
```

	A	B
1		
2	TagName	Average
3	SysTimeSec	28.35

Figure 7: Cell B1 Query

Note: The text in cell B1 can also be hidden by formatting the text color to white (or whatever color matches your background).



Font: Arial
Font style: Regular
Size: 10
Underline: None
Color:
☐ Normal font

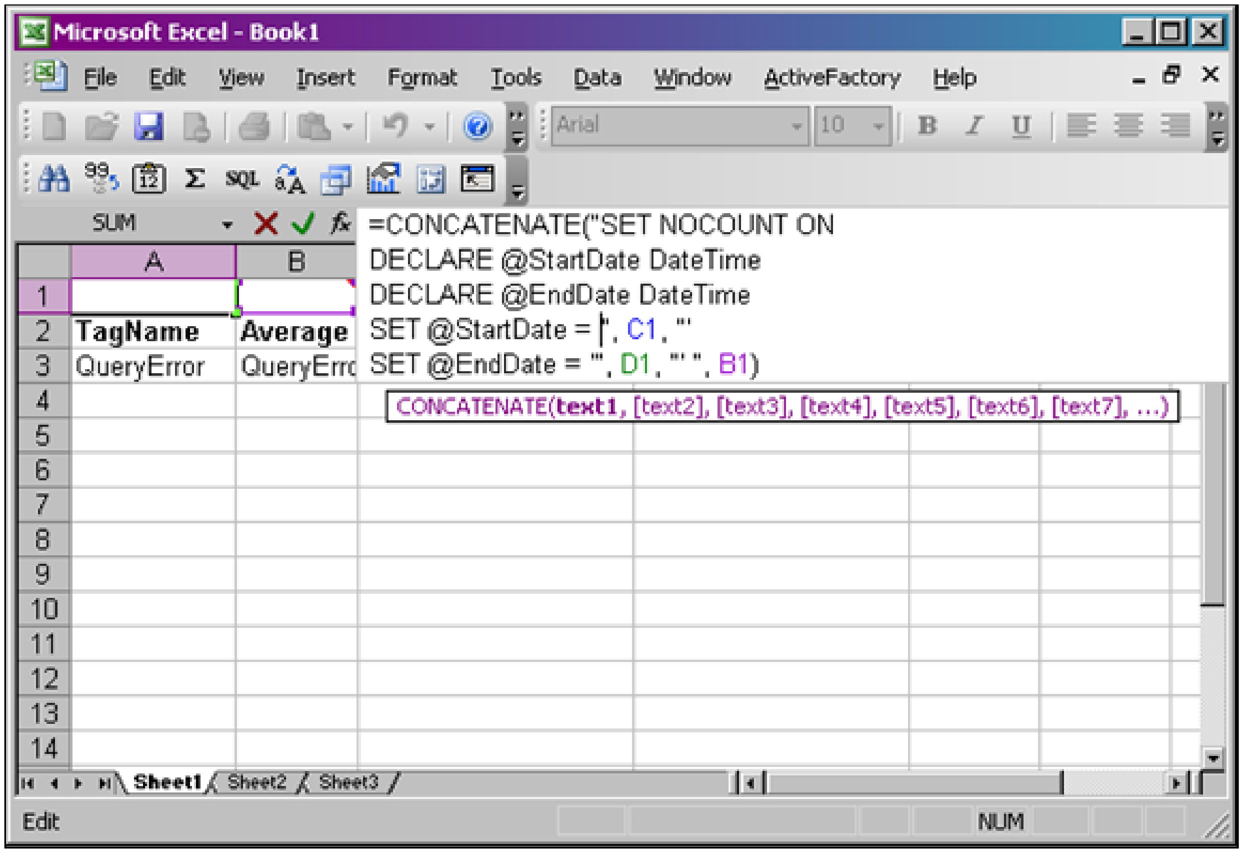
This is a TrueType font. The same ... printer and your screen.

Figure 8: Format Cells to Hide Query

Troubleshooting Query Errors

If you make a mistake with the =CONCATENATE() function, the **QueryError** message appears in the cell.

1. To debug an error, remove one of the single quotes in the first string, just before the reference to C1:


Microsoft Excel - Book1
File Edit View Insert Format Tools Data Window ActiveFactory Help
SUM
Edit
NUM

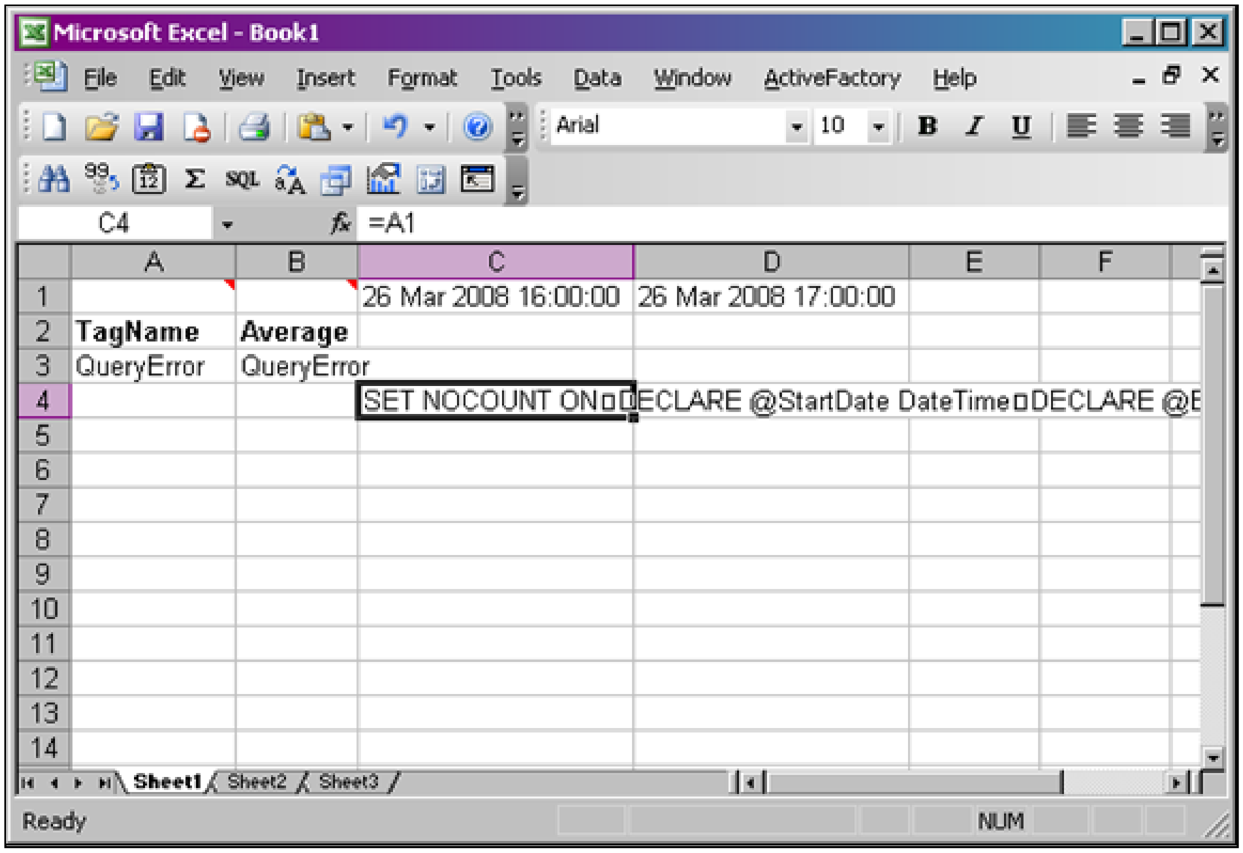
```
=CONCATENATE("SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = '", C1, "'
SET @EndDate = '", D1, "' ", B1)
```

```
CONCATENATE(text1, [text2], [text3], [text4], [text5], [text6], [text7], ...)
```

	A	B
1		
2	TagName	Average
3	QueryError	QueryErrc

Figure 9: Troubleshooting

2. To test the query, type **=A1** in a cell to see what the concatenate function actually produces:


Microsoft Excel - Book1
File Edit View Insert Format Tools Data Window ActiveFactory Help
Arial
10
Ready
NUM

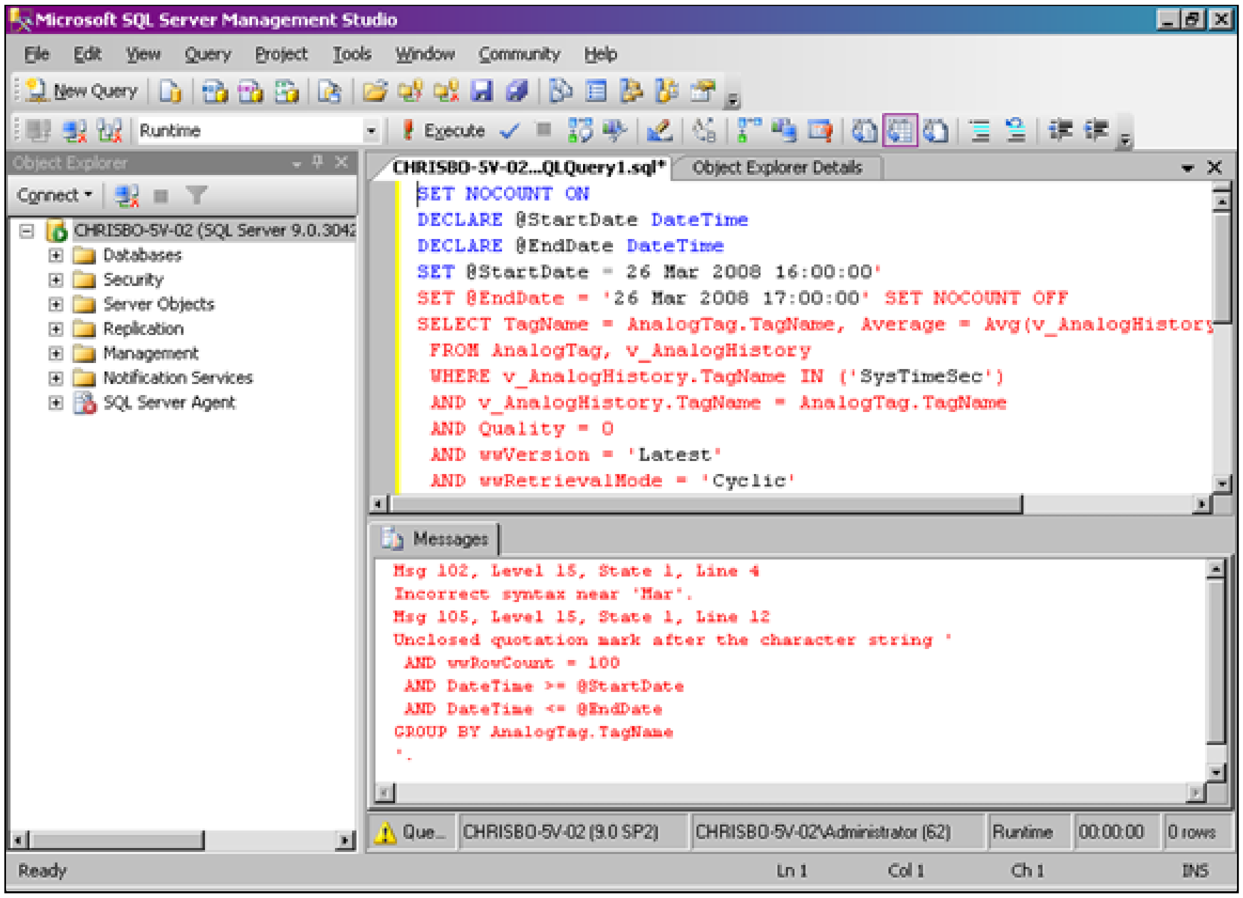
C4 =A1

	A	B	C	D	E	F
1			26 Mar 2008 16:00:00	26 Mar 2008 17:00:00		
2	TagName	Average				
3	QueryError	QueryError				
4			SET NOCOUNT ON□DECLARE @StartDate DateTime□DECLARE @E			

Figure 10: Query Production

3. Right-click and copy the C4 test cell, then paste it into Microsoft Query Analyzer or SQL Server

Management Studio. Remove the double-quotes (") that get inserted at the beginning and end, then execute the query.

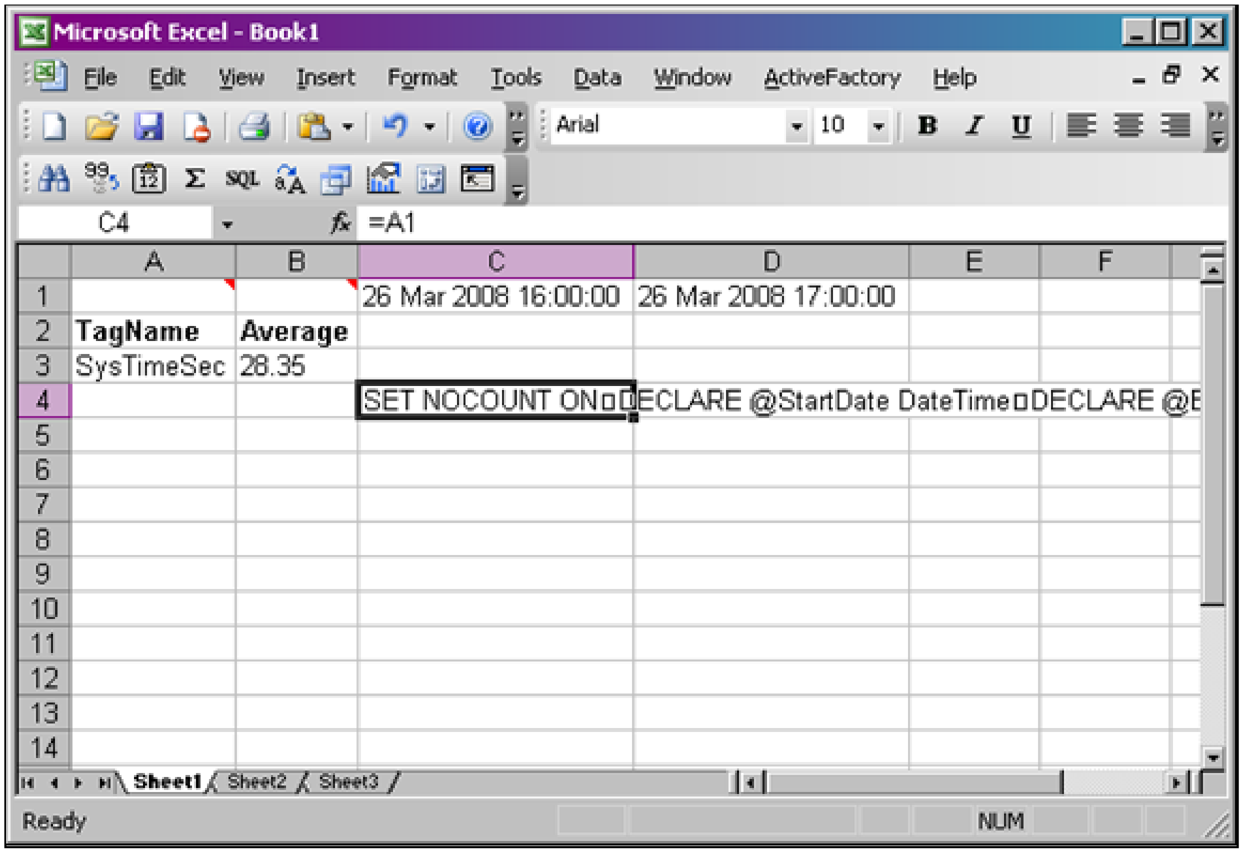


```
SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = 26 Mar 2008 16:00:00'
SET @EndDate = '26 Mar 2008 17:00:00' SET NOCOUNT OFF
SELECT TagName = AnalogTag.TagName, Average = Avg(v_AnalogHistory
 FROM AnalogTag, v_AnalogHistory
 WHERE v_AnalogHistory.TagName IN ('SysTimeSec')
 AND v_AnalogHistory.TagName = AnalogTag.TagName
 AND Quality = 0
 AND wwVersion = 'Latest'
 AND wwRetrievalMode = 'Cyclic'
```

```
Msg 102, Level 15, State 1, Line 4
Incorrect syntax near 'Mar'.
Msg 105, Level 15, State 1, Line 12
Unclosed quotation mark after the character string '
 AND wwRowCount = 100
 AND DateTime >= @StartDate
 AND DateTime <= @EndDate
GROUP BY AnalogTag.TagName
'.
```

Figure 11: Testing the Query String

4. Now you can see that a single-quote (') is missing in the **SET @StartDate=** statement. Make the fix in the =CONCATENATE() function; results are returned rather than an error message.


Microsoft Excel - Book1
File Edit View Insert Format Tools Data Window ActiveFactory Help
Arial 10
Sheet1 Sheet2 Sheet3
Ready NUM

C4 =A1

	A	B	C	D	E	F
1			26 Mar 2008 16:00:00	26 Mar 2008 17:00:00		
2	TagName	Average				
3	SysTimeSec	28.35				
4			SET NOCOUNT ON□DECLARE @StartDate DateTime□DECLARE @E			

Figure 12: Returning Results, not an Error