

# SQL SERVER<sup>TM</sup> MANAGEMENT STUDIO TIPS AND TRICKS

DR GREG LOW

Third Edition 2025



# SQL Server™ Management Studio

## Tips and Tricks

Dr Greg Low  
SQL Down Under Pty Ltd  
<https://bsky.app/profile/greglow.bsky.social>

<http://ssmsbook.sqldownunder.com>

First edition April 2018  
Second edition March 2019  
Third edition September 2025

SQL Server is a trademark of Microsoft Corporation

Cover Image by the amazing **Glen Carrie** (c/- Unsplash <https://unsplash.com/photos/JiSkHnWLo2o>)

This eBook is copyright material and must not be copied, reproduced, transferred, distributed, leased, licensed or publicly performed or used in any way except as specifically permitted in writing by the publishers, as allowed under the terms and conditions under which it was purchased or as strictly permitted by applicable copyright law. Any unauthorized distribution or use of this text may be a direct infringement of the author's and publisher's rights and those responsible may be liable in law accordingly.

The SSMS product described in this eBook is itself subject to constant change. The tips and tricks described could change at any point in time. We've done our best to make this eBook as error free as possible at the time of publication, but we don't promise that it is error free or that anything we describe will work for you.

### **Note from the author:**

I've worked with SQL Server™ for decades and have compiled this list of tips and tricks over that time. This eBook is a compilation of a series of blog posts that I have either made or are scheduled to be made as part of my Thursday "Shortcut" series, for shortcuts that apply to SSMS. (On each heading, I've added the publication date and depending upon when you read this, some might be in the future).

While I'd love to remember who first showed me each one of these, the original source of any particular tip or trick is now too difficult to determine, and often may have come from multiple sources. Though, I need to thank the Data Platform MVP community for endless inspiration, and for no doubt being the ones who showed me many of these items over the years. For that reason, I consider myself the editor more than the author.

We intend to keep enhancing and upgrading this book. If you have feedback for it, please send that to [ssmsbook@sqldownunder.com](mailto:ssmsbook@sqldownunder.com)



Need to learn about SQL Server? SQL Down Under offer online on-demand courses that you can take whenever you want.

You can learn with Greg right now!

We're rapidly expanding our list of courses. And we have courses on topics that others don't even consider.

Check us out now at <http://sqldownunder.com>

# Contents

---

<b>Foreward .....</b>	<b>6</b>
<b>1    Object Explorer .....</b>	<b>8</b>
1.1    Dragging all column names from Object Explorer .....	8
1.2    Control the use of square brackets (quoting) when dragging columns .....	10
1.3    Script Multiple Objects at Once in SSMS .....	11
1.4    Add columns to Object Explorer Details window (ie: rowcounts of tables) .....	13
1.5    Extended Properties for objects .....	16
1.6    Filters in Object Explorer .....	18
1.7    Navigate as you type in sorted OED pane .....	20
1.8    Generate insert scripts (script data) .....	22
1.9    Turn off option to prevent saving changes that require table recreation .....	33
1.10    Dependency tracking .....	34
1.11    Built-in standard reports .....	37
1.12    Custom report creation .....	41
1.13    Database Scoped Configurations UI .....	52
1.14    Pane options in Edit N Rows .....	53
<b>2    Appearance .....</b>	<b>55</b>
2.1    Environment Fonts.....	55
2.2    Changing displayed status bar values.....	57
2.3    Import and Export settings.....	58
2.4    Presentation Mode .....	61
2.5    Screen and Printing Colors.....	63
2.6    Cleaning up the Scroll Bar .....	65
2.7    Making sense of the colors in the SSMS scroll bar .....	68
2.8    Scroll bar map mode .....	69
2.9    Adding multi-level undo, redo .....	73
2.10    Toggle full screen (Alt shift enter).....	77
2.11    Use visual glyphs for word wrap.....	78
2.12    Using zoom features.....	81
2.13    Using environment color themes .....	82
2.14    Grid Results Border Colors.....	84
<b>3    Query Editing .....</b>	<b>86</b>
3.1    Using Line Numbers and GOTO Line Number.....	86
3.2    Useful keyboard shortcuts .....	88
3.3    Apply cut or copy commands to blank lines when there is no selection .....	90
3.4    The magical F1 key – help on syntax and metadata.....	91
3.5    Fixing or improving the online documentation .....	94
3.6    Manually prompting for and Refreshing Intellisense .....	99
3.7    Replace Tabs with Spaces and do Macro-like work using Regular Expressions .....	101
3.8    Selecting and modifying rectangular regions in SSMS .....	103
3.9    Using the Clipboard Ring in SSMS.....	105
3.10    Using Code Outlining.....	107



3.11	Using Bookmarks.....	110
3.12	Using Query Templates.....	111
3.13	Creating T-SQL Templates.....	114
3.14	Using Snippets.....	119
3.15	Using Snippets to Improve the Drop Database Statement .....	121
3.16	Using "Surround with" snippets .....	127
3.17	Intellisense casing for function names .....	130
3.18	Scrolling Sensitivity.....	131
<b>4</b>	<b>Executing Queries .....</b>	<b>132</b>
4.1	Adding additional parameters to connections .....	132
4.2	Using Colors to Avoid Running Scripts Against the Wrong Server .....	134
4.3	Change connection.....	137
4.4	Configuring registered servers .....	139
4.5	Multi-server Queries .....	143
4.6	Using a Count with the GO Batch Separator in T-SQL .....	147
4.7	Viewing Client statistics .....	150
4.8	Set shortcuts for favorite stored procedures .....	152
4.9	Set SQLCMD mode for all new windows .....	153
4.10	Activity Monitor .....	156
4.11	Encryption Status in Query Status Bar .....	160
4.12	Avoiding deadlocks when working interactively .....	161
<b>5</b>	<b>Results .....</b>	<b>163</b>
5.1	Changing the number of rows selected or edited in Object Explorer .....	163
5.2	Viewing and configuring spatial data output.....	164
5.3	Using the XML editor and Increasing the XML output size .....	166
5.4	Find error locations within scripts .....	169
5.5	Determining when a query finished .....	170
5.6	Play a sound when a query completes.....	171
5.7	Query and results in separate tab .....	172
5.8	Turning off completion times .....	173
5.9	Closing Idle Connections.....	176
<b>6</b>	<b>Execution Plans .....</b>	<b>178</b>
6.1	Compare query plans .....	178
6.2	Missing index details .....	182
6.3	Saving and sharing query plans .....	184
6.4	Saving and sharing deadlock graphs .....	186
6.5	Zooming and navigating query plans.....	188
<b>7</b>	<b>Projects/Solutions .....</b>	<b>190</b>
7.1	Change default text in new query windows .....	190
7.2	Pinning and clearing the connection entries.....	191
7.3	Configure autorecover time, and recover unsaved queries .....	193
7.4	Accessing script files and folders in SSMS .....	194
7.5	Using quick launch .....	195
7.6	Run SSMS as someone else.....	196

7.7	Using script projects and solutions .....	199
7.8	Start faster by disabling CRL checking in constrained environments .....	203
7.9	Connecting to Azure Storage and other services .....	204
7.10	Setting startup options .....	206
7.11	Providing feedback directly to the SSMS team .....	207
7.12	Accessing Preview Features .....	209
7.13	Working in both English and another language.....	210
7.14	Adding support for Analysis Services, Integration Services, and Reporting Services ...	212
7.15	Disabling the Open Transaction Check when Closing SSMS .....	214
7.16	Using encodings when opening or saving files .....	215
7.17	Git Integration .....	218
7.18	Opening shortcuts.....	221
7.19	Comparing Scripts and Other files.....	222
<b>8</b>	<b>Window Tabs .....</b>	<b>224</b>
8.1	Pinned Tabs.....	224
8.2	Reset window layout.....	226
8.3	Using Split Screens .....	228
8.4	Document Groups.....	231
8.5	Undock tabs and windows and using multiple screens .....	233
8.6	Using the PowerShell terminal .....	234
8.7	Renaming tabs that have not been saved .....	236

## Foreward

---

If you work with SQL databases, you've spent hundreds – maybe thousands – of hours working in SQL Server Management Studio (SSMS). It's not just an application; it's the entry point for nearly everything you do with a SQL database. Whether you're writing queries, troubleshooting performance issues, trying out a new feature, or navigating a database in Object Explorer, SSMS is where it happens. Yet for as long as SSMS has been around, many of us have only scratched the surface of what it can do.

As software users, we figure out what we need to do, in ways that work for us. These habits and patterns become ingrained in our daily use of the product. New features often fly under the radar. There's no blinking sign that tells you how to speed up your workflow, find hidden functionality, or make the interface work for you instead of against you. Those gems are scattered across documentation, blog posts, and conference demos, or passed from one DBA to another during late-night troubleshooting sessions.

Thankfully, this book brings all those tucked-away treasures together. It's rare to find someone who not only masters a tool but also makes it more approachable for everyone, yet that's exactly what Dr. Greg Low has done for SSMS. Greg has spent decades helping people get the most out of SQL Server, and it shows in these pages. This book reflects the experience of someone who has lived in SSMS day in and day out, and who knows that the best tools don't just function, they empower. Greg doesn't simply know SSMS; he knows how people actually use it (and misuse it), and he knows the difference the right shortcut can make at the right time.

Introduced in 2005 as the successor to Enterprise Manager and Query Analyzer, the first versions of SSMS provided the basics: a query editor, object exploration, and a few tools to monitor performance. Over the next decade and a half, each release added new capabilities such as IntelliSense, monitoring for Availability Groups, Query Store reports, setup for Managed Instance Link, and Always Encrypted configuration. Even with these improvements, it's fair to say that in recent years the community felt SSMS had been left behind, and many believed it was on the path to retirement. The release of SSMS 21 marked a turning point. It wasn't an update; it was a revival. Rebuilt from the ground up and based on the latest release of Visual Studio, SSMS has reestablished its role and its relevance.

The SSMS 21 journey officially kicked off in early 2024, but the move to Visual Studio 2022 had long been a discussion topic for the SSMS team. It takes time to change direction on a big ship, and it takes effort to coordinate across organizations. The engineering teams dove in, creating space and reconstructing SSMS piece by piece through a new pipeline. SSMS 21 brings numerous features that users requested for years, including a modern installer, automatic updates, 64-bit support, Git integration, initial dark theme support, and query editor improvements.

All these changes, in recent years and over the life of SSMS, have introduced complexity and as a result, you're probably not using SSMS to its full potential.

That's why this book is such a gift; it contains a wealth of information you may not even know you need. It doesn't teach you how to use SSMS - it shows you how to use it better. It's full of practical, relevant tips that will help you work faster and smarter...and maybe even enjoy your time in SSMS a little more.

Further, this book meets you where you are. New to SSMS? You'll discover features that will flatten the learning curve. Been using it for years? I guarantee you'll still find shortcuts or settings you never knew existed, perhaps finally understanding what that odd little button in the toolbar does. While I used to spend a lot of time in execution plans (chapter 5), you'll find many of my favorite tips in chapter 3, as they saved hours of time over the years when working in the query editor.

Whether you're a developer, a DBA, a data analyst, or someone who just wants to get more out of their everyday work with SQL, this book is your guide to unlocking the best of what SSMS has to offer. Keep it handy. Share it with teammates. You're about to learn how to make tomorrow's work easier than today's.

Let's get started.

Erin Stellato  
Principal Program Manager  
Microsoft

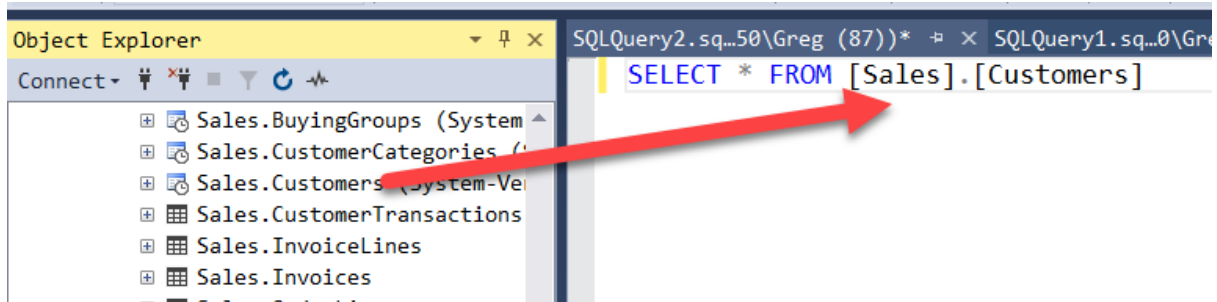


## 1 Object Explorer

### 1.1 Dragging all column names from Object Explorer

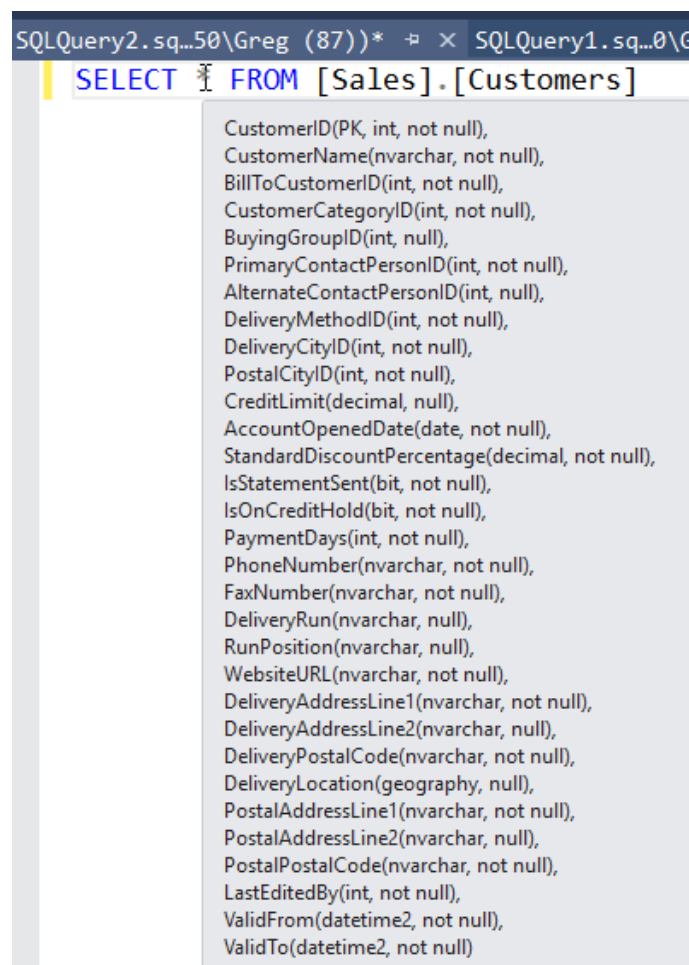
This is a popular shortcut in SQL Server Management Studio (SSMS) but I continue to be amazed how many people aren't aware of it.

Object Explorer is a very useful part of SSMS and you can drag pretty much any name that you see in it across to a query window.

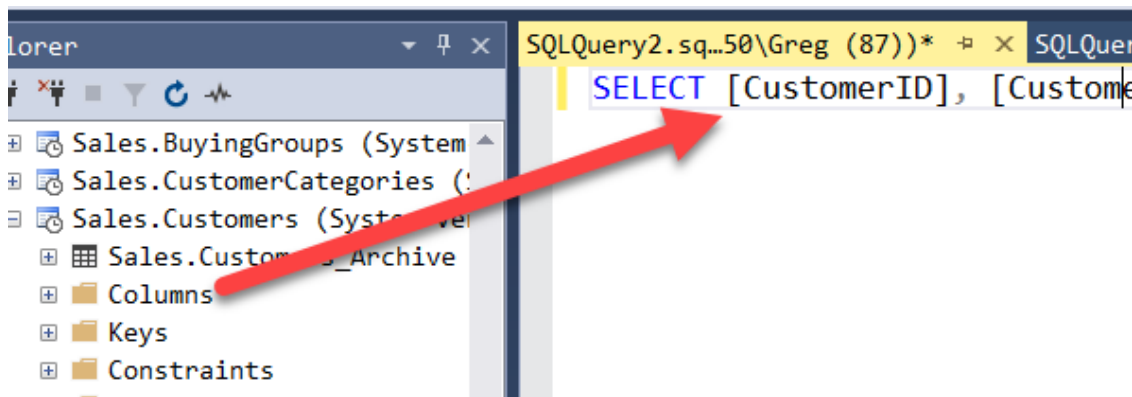


You could do the same for each column in the Columns list.

You might also realize that you can hover over the asterisk and see a list of columns:

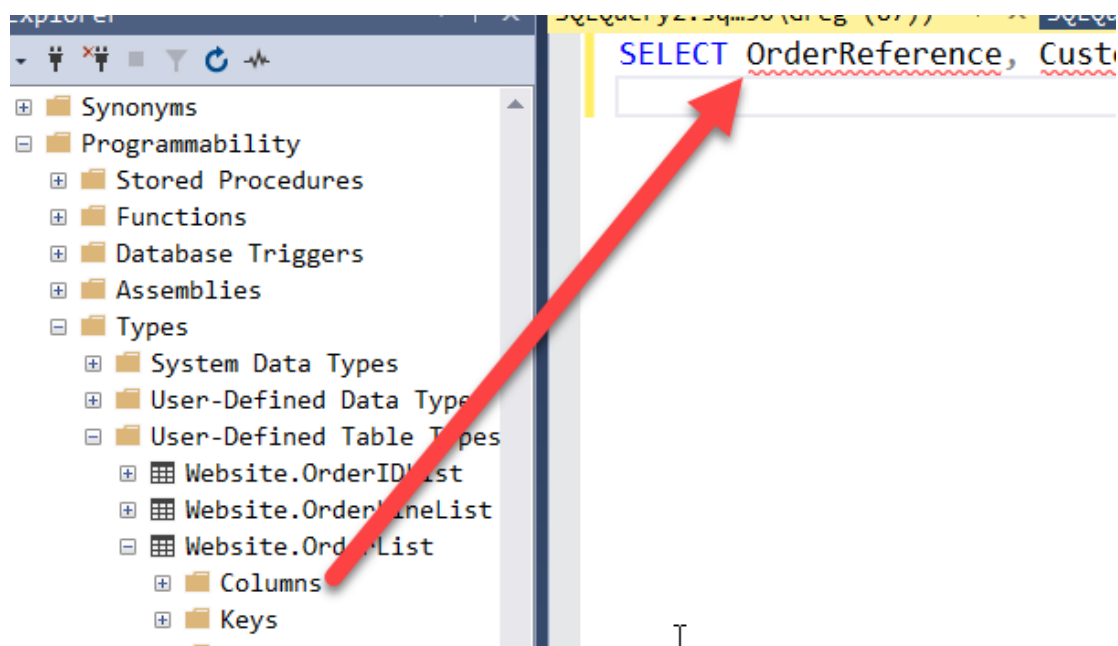


But the one that many people aren't aware of is that you can drag the word **Columns** across and get all the columns:



The final thing that seemed missing about this for me, was that I often wanted to do the same with the column list in a user defined table type.

I asked [@sqltoolsguy](#) and the SSMS team nicely, and only a few weeks later, that works too:



In earlier versions, the quoting wasn't consistent. Version 17.6 and later fixed that.

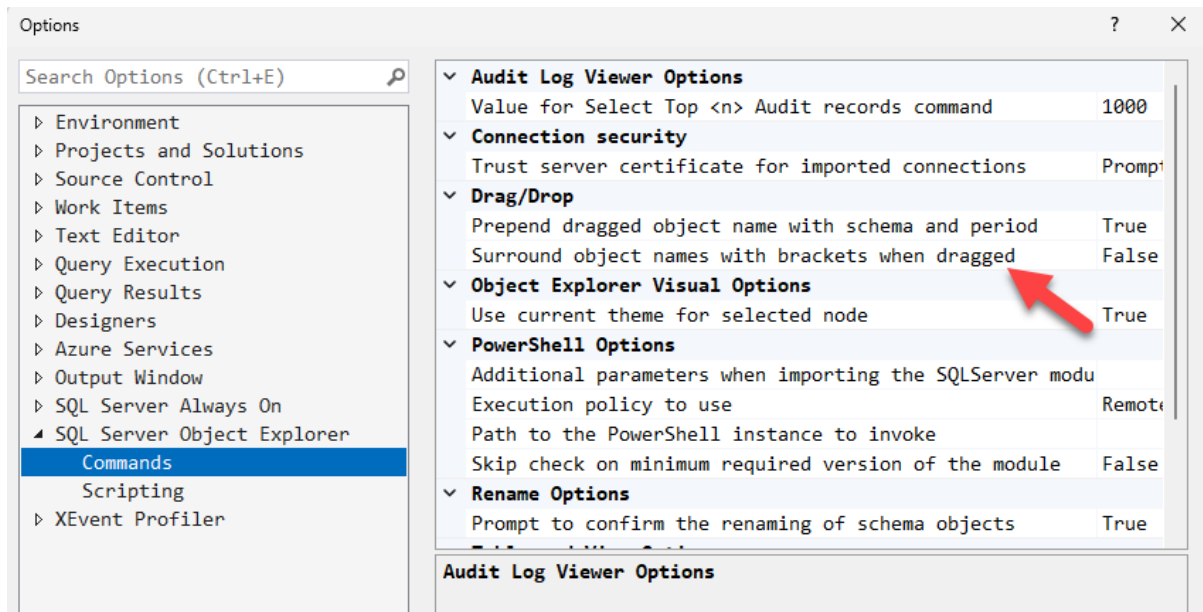
## 1.2 Control the use of square brackets (quoting) when dragging columns

Previously I showed how useful it is to be able to drag columns from either a table or from user-defined table type onto a query window.

But one of the first comments I hear from people who did that is:

*I hate all those square brackets. How do I stop that?*

The option to do that was added to **Tools> Options > SQL Server Object Explorer > Commands**:

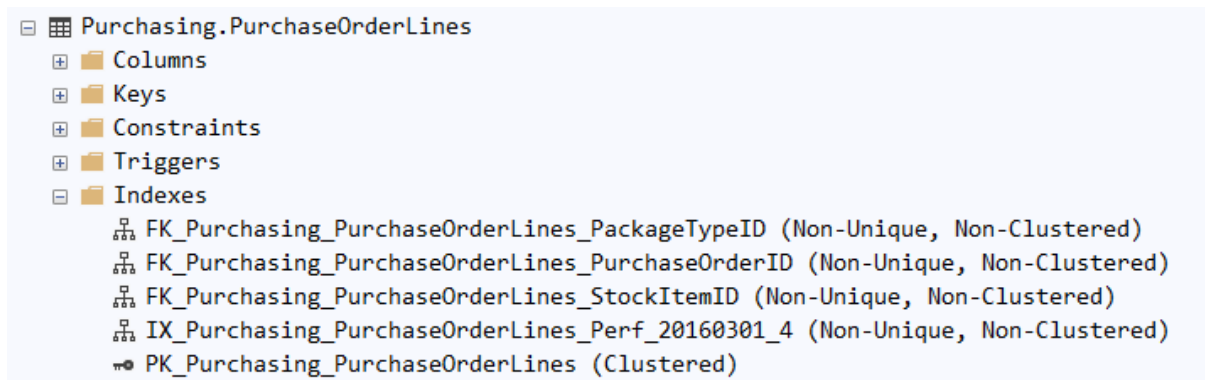


Bonus points for the team because they also provided us with an option to decide whether schema names are dragged as well. That's mostly used for tables but also applies to other schema-bound objects like stored procedures.

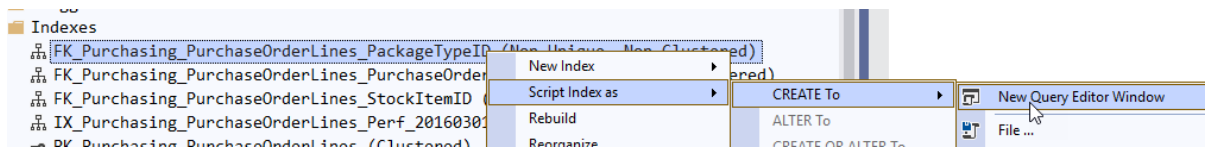
### 1.3 Script Multiple Objects at Once in SSMS

If I want to script all the indexes on a table (or all the tables, all the stored procedures, etc, etc.), you can do that the long way by using the **Generate Scripts** right-click option for a database, but there's a better way.

Let's use the **Purchasing.PurchaseOrderLines** table from **WideWorldImporters** as an example. Here are the indexes on it:

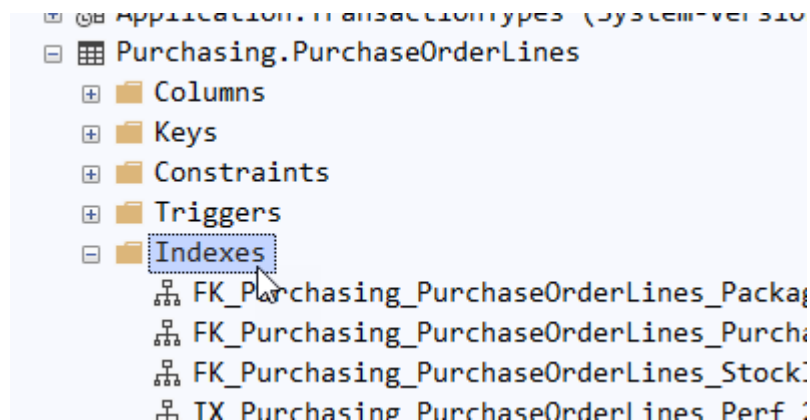


The scripting options are well-known. You right-click the object, and can navigate to the scripting option:

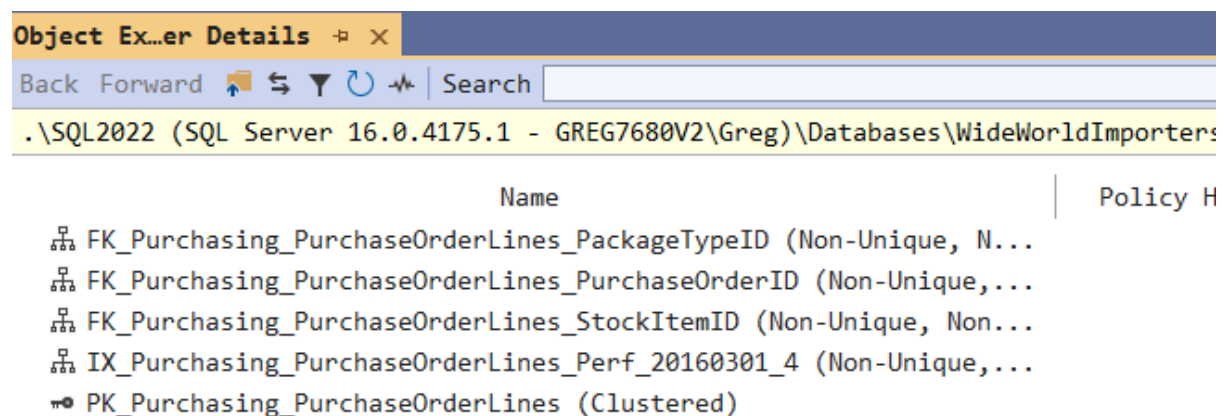




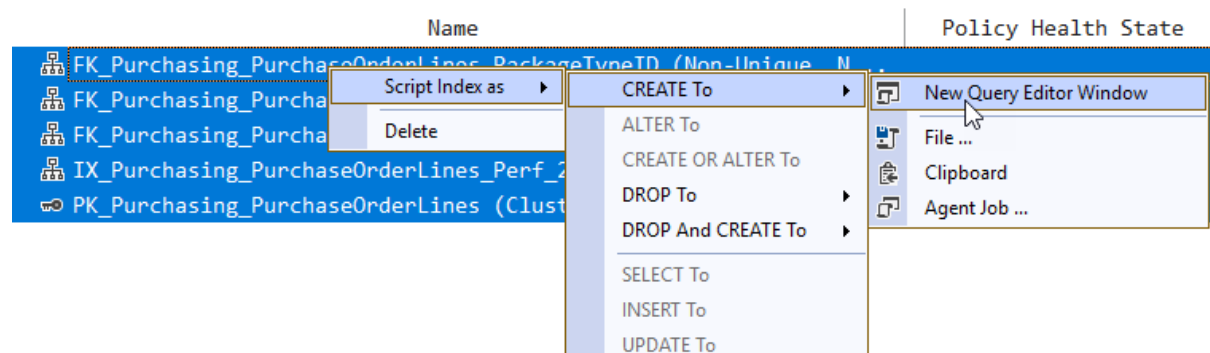
But this is tedious when you want to script a whole set of them. Instead, start by clicking on the **Indexes** node itself:



When it's selected, click **F7** to open the **Object Explorer Details** pane:



Then click the top item, shift-click the bottom item (to select them all), and finally right-click to see that you can script all of them at once:



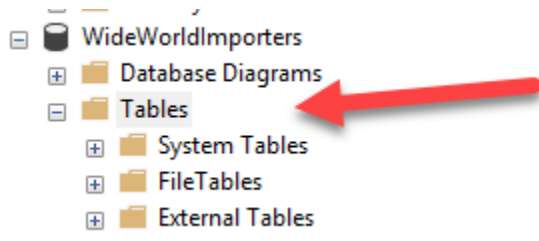
You could also use Control-click to select just the ones you want. This is also an easy way to drop more than one at the same time as well.

#### 1.4 Add columns to Object Explorer Details window (ie: rowcounts of tables)

I've mentioned in another section about scripting multiple objects at once, how useful the Object Explorer Details window is, and how little understood it is.

Another useful option in it, is that the displayed columns can be changed. In particular, you can add columns that would be useful. Let's look at an example.

In Object Explorer, I've expanded the WideWorldImporters database and clicked on the word Tables:

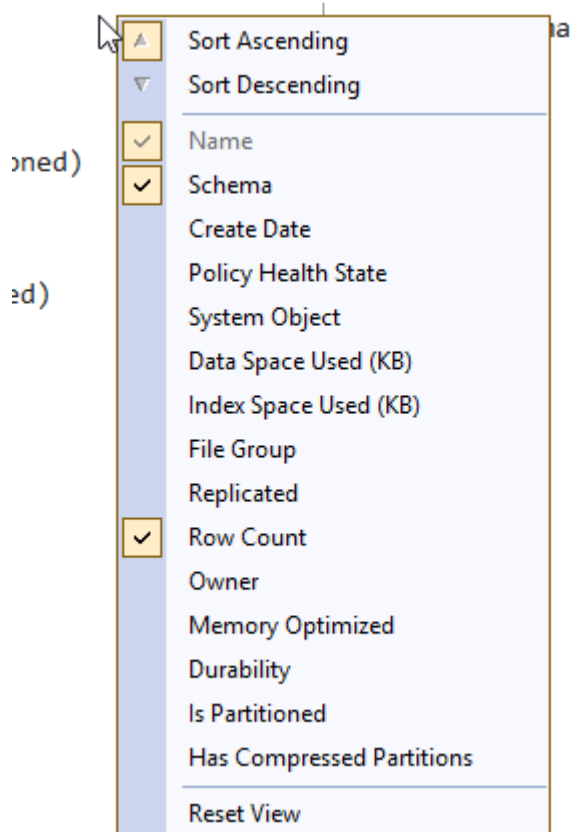


Next, I hit the **F7** key, and the **Object Explorer Details** pane opens showing this:















Name	Schema	Create Date	Policy Health State
System Tables			
FileTables			
External Tables			
Cities (System-Versioned)	Application	2/06/2016 10:04 AM	
Countries (System-Versioned)	Application	2/06/2016 10:04 AM	
DeliveryMethods (System-Versioned)	Application	2/06/2016 10:04 AM	
PaymentMethods (System-Versioned)	Application	2/06/2016 10:04 AM	
People (System-Versioned)	Application	2/06/2016 10:04 AM	
StateProvinces (System-Versioned)	Application	2/06/2016 10:04 AM	
SystemParameters	Application	2/06/2016 10:04 AM	
TransactionTypes (System-Versioned)	Application	2/06/2016 10:04 AM	
PurchaseOrderLines	Purchasing	2/06/2016 10:04 AM	
PurchaseOrders	Purchasing	2/06/2016 10:04 AM	
SupplierCategories (System-Versioned)	Purchasing	2/06/2016 10:04 AM	
Suppliers (System-Versioned)	Purchasing	2/06/2016 10:04 AM	
SupplierTransactions	Purchasing	2/06/2016 10:04 AM	
BuyingGroups (System-Versioned)	Sales	2/06/2016 10:04 AM	
CustomerCategories (System-Versioned)	Sales	2/06/2016 10:04 AM	
Customers (System-Versioned)	Sales	2/06/2016 10:04 AM	
CustomerTransactions	Sales	2/06/2016 10:04 AM	
InvoiceLines	Sales	2/06/2016 10:04 AM	
Invoices	Sales	2/06/2016 10:04 AM	
OrderLines	Sales	2/06/2016 10:04 AM	
Orders	Sales	2/06/2016 10:04 AM	
SpecialDeals	Sales	2/06/2016 10:04 AM	
ColdRoomTemperatures (System-Versioned)	Warehouse	6/06/2016 11:06 AM	
Colors (System-Versioned)	Warehouse	2/06/2016 10:04 AM	
PackageTypes (System-Versioned)	Warehouse	2/06/2016 10:04 AM	
StockGroups (System-Versioned)	Warehouse	2/06/2016 10:04 AM	
StockItemHoldings	Warehouse	2/06/2016 10:04 AM	
StockItems (System-Versioned)	Warehouse	2/06/2016 10:04 AM	
StockItemStockGroups	Warehouse	2/06/2016 10:04 AM	
StockItemTransactions	Warehouse	2/06/2016 10:04 AM	
VehicleTemperatures	Warehouse	6/06/2016 11:06 AM	

I get a list of tables showing Name, Schema, Create Date, and Policy Health State. Policy was an interesting concept that I thought never got fully baked into the product, so it's not of great interest to me, and really is just clutter. However, the number of rows in the table would be much more interesting.

If I right-click on the heading row, I get options for what is displayed. I've excluded the ones I don't want, and I've included the Row Count.



I'm left with a much more useful result:

Name	Schema	Row Count
 BuyingGroups (System-Versioned)	Sales	2
 Cities (System-Versioned)	Application	37,940
 ColdRoomTemperatures (System-Versioned)	Warehouse	4
 Colors (System-Versioned)	Warehouse	36
 Countries (System-Versioned)	Application	190
 CustomerCategories (System-Versioned)	Sales	8
 Customers (System-Versioned)	Sales	663
 CustomerTransactions	Sales	97,147
 DeliveryMethods (System-Versioned)	Application	10
 Dropped Ledger Tables		
 External Tables		
 FileTables		
 Graph Tables		
 InvoiceLines	Sales	228,265
 Invoices	Sales	70,510
 OrderLines	Sales	231,412
 Orders	Sales	73,595
 PackageTypes (System-Versioned)	Warehouse	14
 PaymentMethods (System-Versioned)	Application	4
 People (System-Versioned)	Application	1,111
 PurchaseOrderLines	Purchasing	8,367
 PurchaseOrders	Purchasing	2,074
 SpecialDeals	Sales	2
		--

And I hear you ask: *Do I have to do this every time?*

The answer is no. If you close the Object Explorer Details Window and hit F7 on the Tables node again, the same output appears.



## 1.5 Extended Properties for objects

I started working with SQL Server in 1992, but all through the 1980's and 1990's, I was also working with Progress 4GL. I thought it was the best of the character-based 4GLs but unfortunately, they did a poor job of migrating to Windows and we decided to stop using the product.

One thing that I used to love with Progress though is that the metadata for each column in the database was much richer than what is present in SQL Server. In fact, Microsoft Access was probably closer to it in that regard. It's something I really missed when moving to SQL Server. In Progress, when I defined a column, I could also define things like:

- The name that would be displayed as a prompt on an input screen for this column
- The format that data in this column would be displayed in
- Any non-default input format or masking
- And so on

Having this type of information in the database and stored in a consistent form can greatly reduce the amount of boilerplate code that needs to be written in applications.

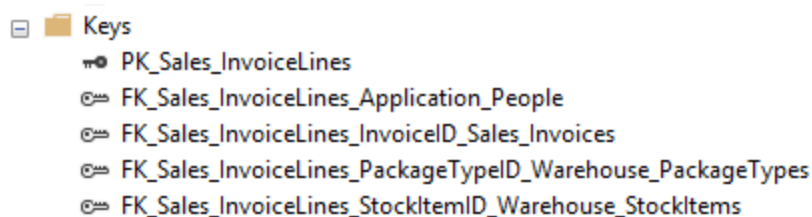
SQL Server does have a concept of extended properties but what I think is missing is a set of "well-known" properties such as "Description". The SQL Server team is starting to use these properties now for things like classifying (or labelling) data for security purposes.

I'd like to see them used more often. Here's an example:

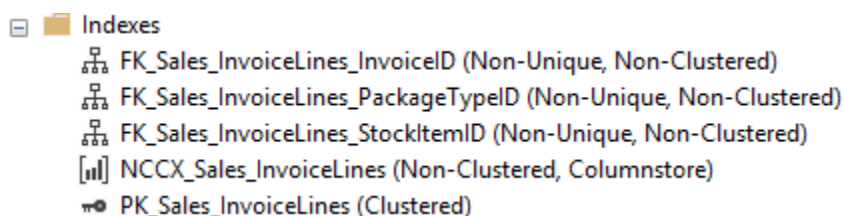
I often come across indexes in databases and when I ask why the index was created, no-one knows why. That also means that they are scared to remove the index. How much easier would this type of thing be if we had standard properties for each index that described why it was added?

We can already do this, but I just wish there were agreed standards for this.

As an example though, I tend to name indexes that are just present to support foreign keys, with the same name as the foreign key. That then matches how primary key indexes are named. In the WideWorldImporters database, you can see the naming convention. For the Sales.InvoiceLines table, here are the keys:

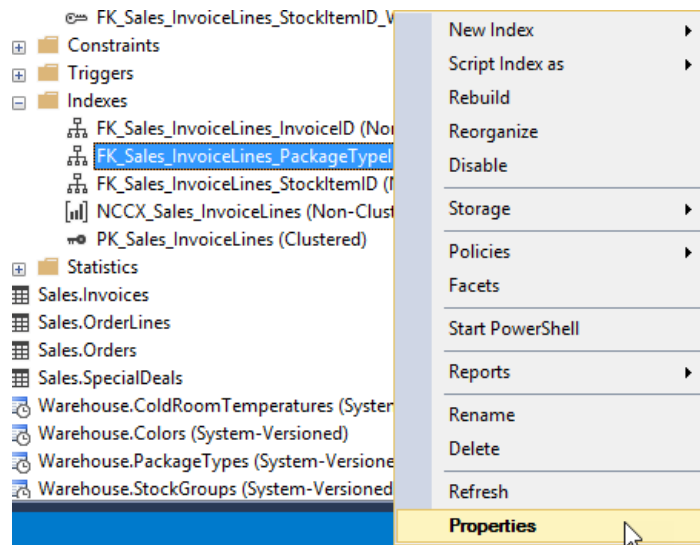


And here are the indexes:

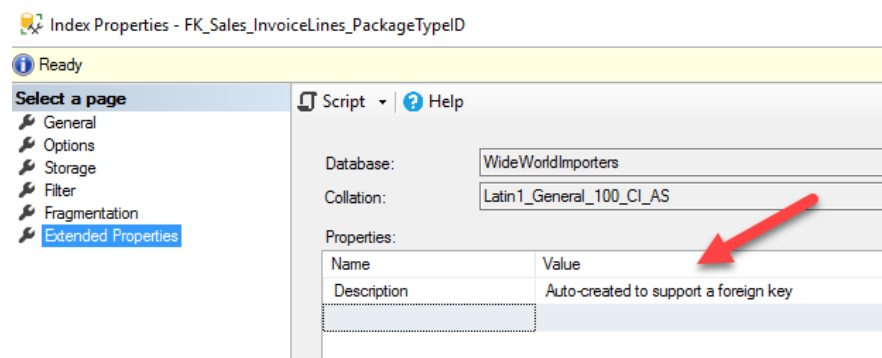


It's pretty obvious which indexes are there to support foreign keys. (*Note we made a conscious decision not to index the Application.People related foreign key*).

But for other indexes, how would you know why they are there? We included that info in our code generation, by using extended properties. To see this, right-click one of these indexes, and choose Properties:



On the Extended Properties page, you can see a description of why this index was created:

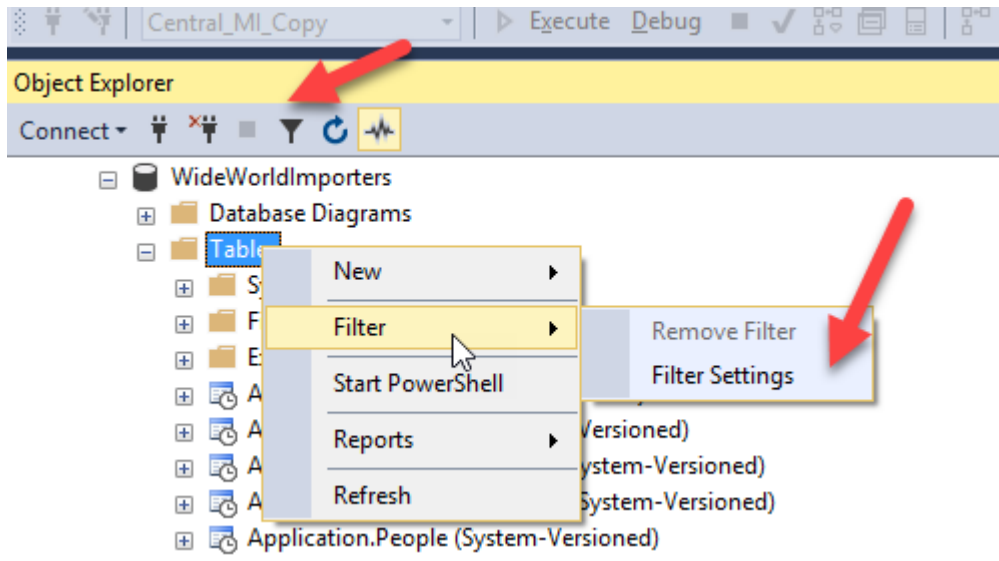


I'd like to see much richer metadata for each object in SQL Server and I suspect we might have to do that with a set of extended properties. I'd like to see a standard set of these defined. Even better would be a richer metadata store within the database engine.

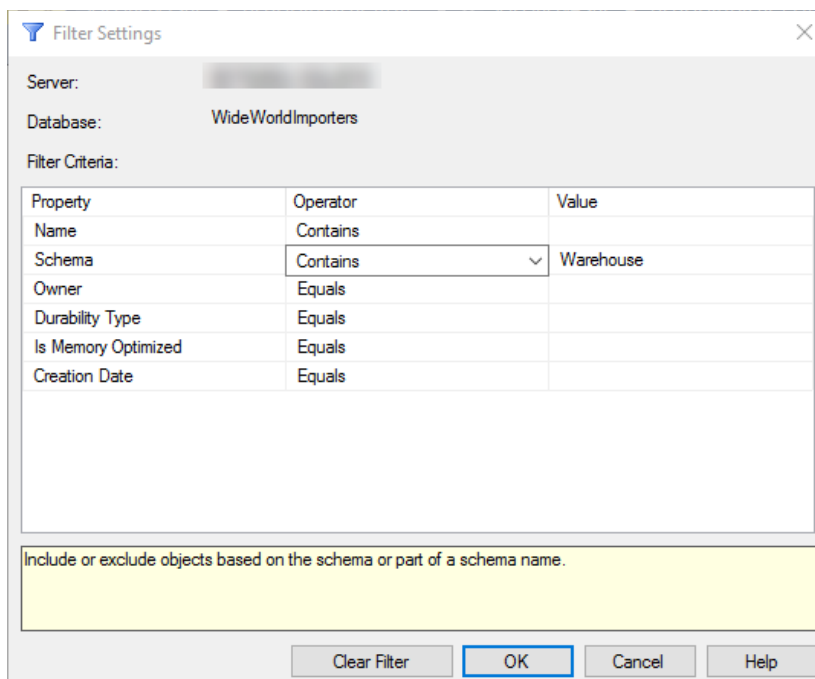
## 1.6 Filters in Object Explorer

If you are working with databases with large numbers of objects, the contents of Object Explorer can start to become a bit overwhelming. I wish it offered an option to group by schema. That would be helpful.

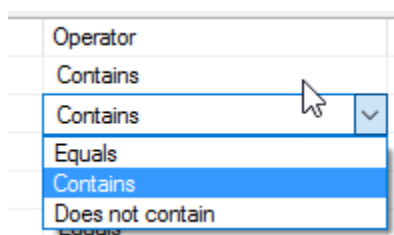
But you can at least filter by things like schema, when you need to work with a specific set of objects. You'll notice that if you click on the database name, that the filter in the toolbar is grayed out, but if you click on a node below that like Tables, you can click on the toolbar filter icon. You can also right-click the node and choose to filter:



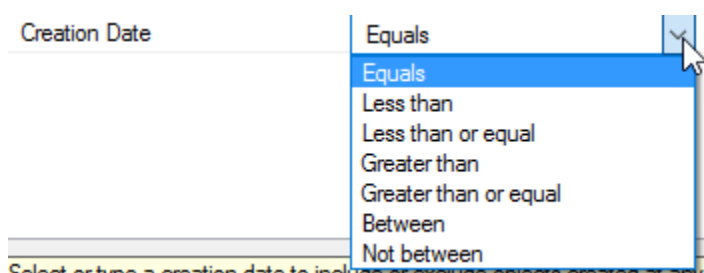
You then are presented with these options:



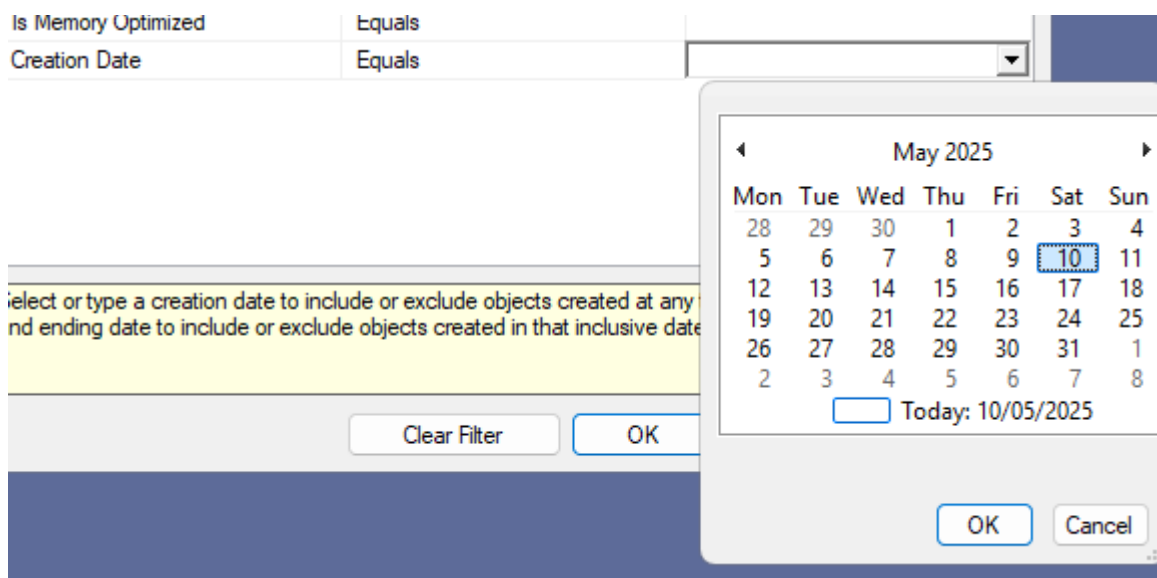
That will work to give me just the tables in the Warehouse schema but notice that the operator is Contains. There are other options:



So, I could see all the tables except those that are in this schema. Note that Creation Date has even more options:



I can then also use this for other interesting queries such as “Just show me tables that have been created since 10<sup>th</sup> May:

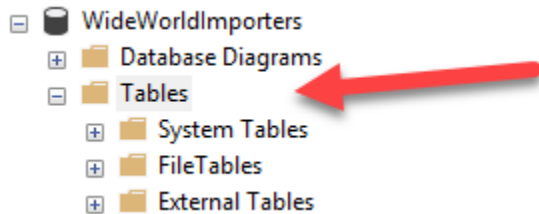


## 1.7 Navigate as you type in sorted OED pane

I've mentioned a number of times how useful I think the Object Explorer Details panel is in SQL Server Management Studio.

One option in that panel that might not be so obvious is the sorted navigation. Here's an example.





I've opened the WideWorldImporters database in Object Explorer, and clicked on the Tables node:



I then hit F7 to open the Object Explorer Details pane and clicked the Name heading to sort the table list. I like it ascending so you might need to click it twice:

Name	Schema
BuyingGroups (System-Versioned)	Sales
Cities (System-Versioned)	Application
ColdRoomTemperatures (System-Versioned)	Warehouse
Colors (System-Versioned)	Warehouse
Countries (System-Versioned)	Application
CustomerCategories (System-Versioned)	Sales
Customers (System-Versioned)	Sales
CustomerTransactions	Sales
DeliveryMethods (System-Versioned)	Application
External Tables	
FileTables	
InvoiceLines	Sales
Invoices	Sales
OrderLines	Sales
Orders	Sales
PackageTypes (System-Versioned)	Warehouse
PaymentMethods (System-Versioned)	Application
People (System-Versioned)	Application
PurchaseOrderLines	Purchasing
PurchaseOrders	Purchasing
SpecialDeals	Sales
StateProvinces (System-Versioned)	Application
StockGroups (System-Versioned)	Warehouse
StockItemHoldings	Warehouse
StockItems (System-Versioned)	Warehouse
StockItemStockGroups	Warehouse
StockItemTransactions	Warehouse
SupplierCategories (System-Versioned)	Purchasing

Then if I type (say the letters Pe), you'll notice that I'm positioned immediately to the first table starting with Pe.

 PackageTypes (System-Versioned)	Warehouse
 PaymentMethods (System-Versioned)	Application
 People (System-Versioned)	Application
 PurchaseOrderLines	Purchasing

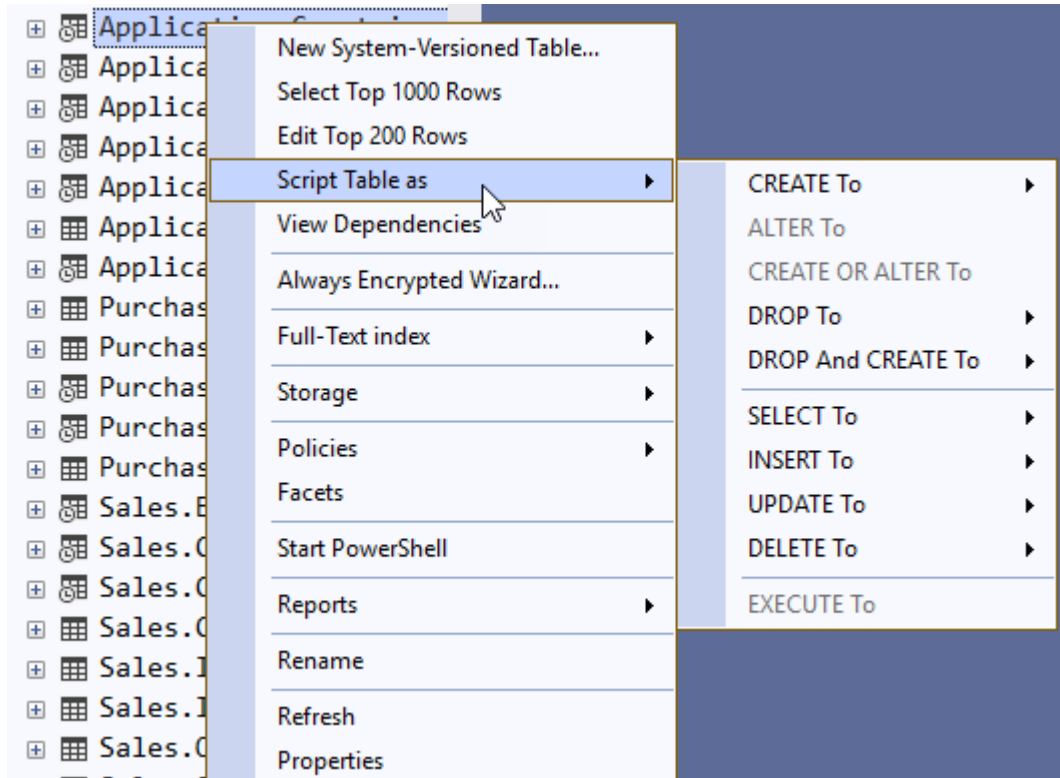
When you have a small number of tables, this is no big deal, but when you have a lot of tables, this is very useful.

## 1.8 Generate insert scripts (script data)

Over the years, I've had a surprising number of questions on how to output all the data in a table as a series of INSERT statements.

SQL Server Management Studio has had the ability to do this for a long time. Here's an example.

In Object Explorer, I've expanded the WideWorldImporters database, then expanded Tables. Where people come unstuck is they right-click the table, and look at the scripting options:



But if you use the option to script INSERT to a new query window, you get this:

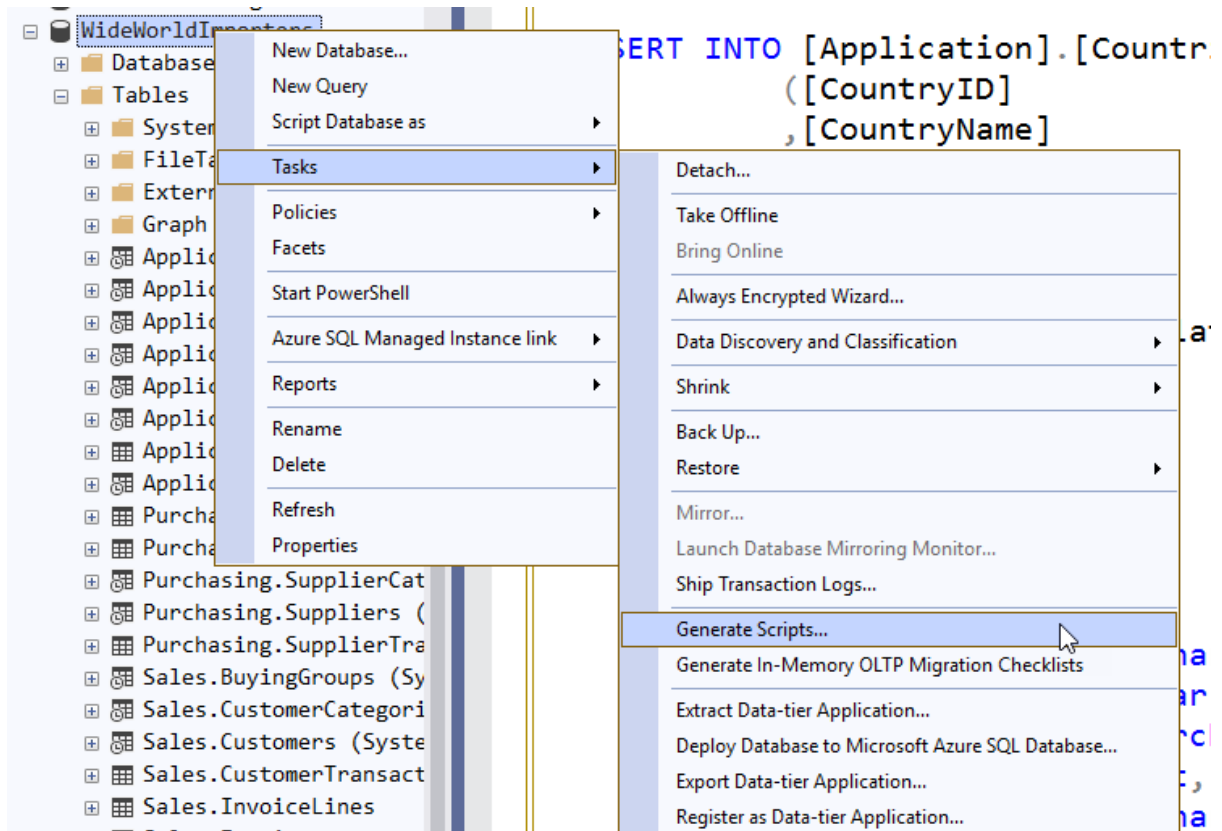
```
USE [WideWorldImporters]
GO

INSERT INTO [Application].[Countries]
    ([CountryID]
    ,[CountryName]
    ,[FormalName]
    ,[IsoAlpha3Code]
    ,[IsoNumericCode]
    ,[CountryType]
    ,[LatestRecordedPopulation]
    ,[Continent]
    ,[Region]
    ,[Subregion]
    ,[Border]
    ,[LastEditedBy])
VALUES
    (<CountryID, int,>
    ,<CountryName, nvarchar(60),>
    ,<FormalName, nvarchar(60),>
    ,<IsoAlpha3Code, nvarchar(3),>
    ,<IsoNumericCode, int,>
    ,<CountryType, nvarchar(20),>
    ,<LatestRecordedPopulation, bigint,>
    ,<Continent, nvarchar(30),>
    ,<Region, nvarchar(30),>
    ,<Subregion, nvarchar(30),>
    ,<Border, geography,>
    ,<LastEditedBy, int,>))
GO
```

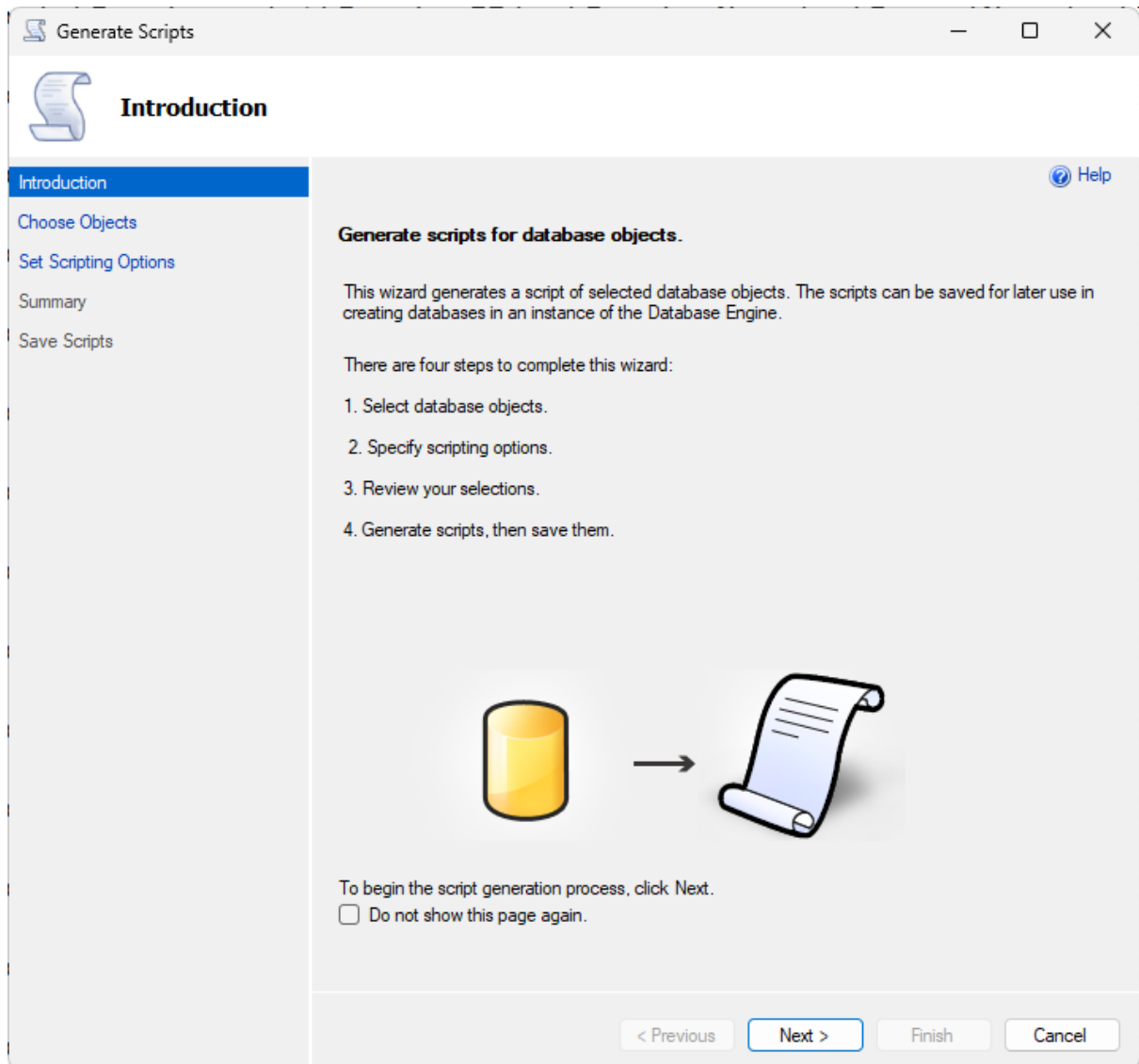
It's scripting an INSERT statement, not the data as a set of INSERT statements.



Now, I think that option should actually be present here, but the way to get to it, is a bit more roundabout. You need to right-click the database, then choose **Tasks**, then **Generate Scripts**.



On the first window, click **Next**.



On the second screen, choose **Select specific database objects**, then expand **Tables**, and pick the table you're after, then click **Next**:

**Select the database objects to script.**

☐ Script entire database and all database objects

☒ Select specific database objects

**Tables**

- ☐ Application.Cities
- ☐ Application.Cities\_Archive
- ☒ Application.Countries
- ☐ Application.Countries\_Archive
- ☐ Application.DeliveryMethods
- ☐ Application.DeliveryMethods\_Archive
- ☐ Application.PaymentMethods
- ☐ Application.PaymentMethods\_Archive
- ☐ Application.People
- ☐ Application.People\_Archive
- ☐ Application.StateProvinces
- ☐ Application.StateProvinces\_Archive
- ☐ Application.SystemParameters
- ☐ Application.TransactionTypes
- ☐ Application.TransactionTypes\_Archive
- ☐ Purchasing.PurchaseOrderLines
- ☐ Purchasing.PurchaseOrders
- ☐ Purchasing.SupplierCategories
- ☐ Purchasing.SupplierCategories\_Archive

Select All   Deselect All

On the third screen, click the **Advanced** button.

**Specify how scripts should be saved.**

☐ Save as notebook Advanced

File name:  ...

☐ Save as script file

Files to generate: ☒ Single script file  
☐ One script file per object

File name:  ...

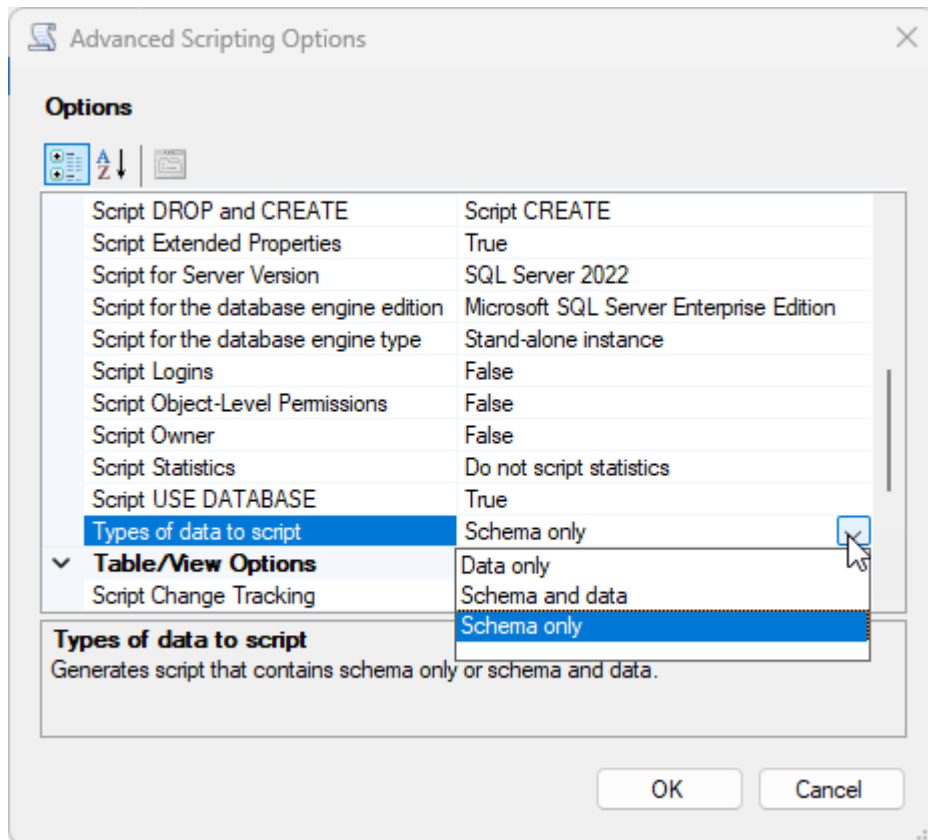
☒ Overwrite existing file

Save as: ☒ Unicode text  
☐ ANSI text

☐ Save to clipboard

☐ Open in new query window

In the **Advanced Scripting Options** window, scroll down to find **Types of data to script** and note the options:



Choose **Data only** if that's all you want, then **OK**. I've then chosen **Open in new query window**, and **Next**.

**Specify how scripts should be saved.**

☐ Save as notebook Advanced

File name:  ...

☐ Save as script file

Files to generate: ☒ Single script file  
☐ One script file per object

File name:  ...

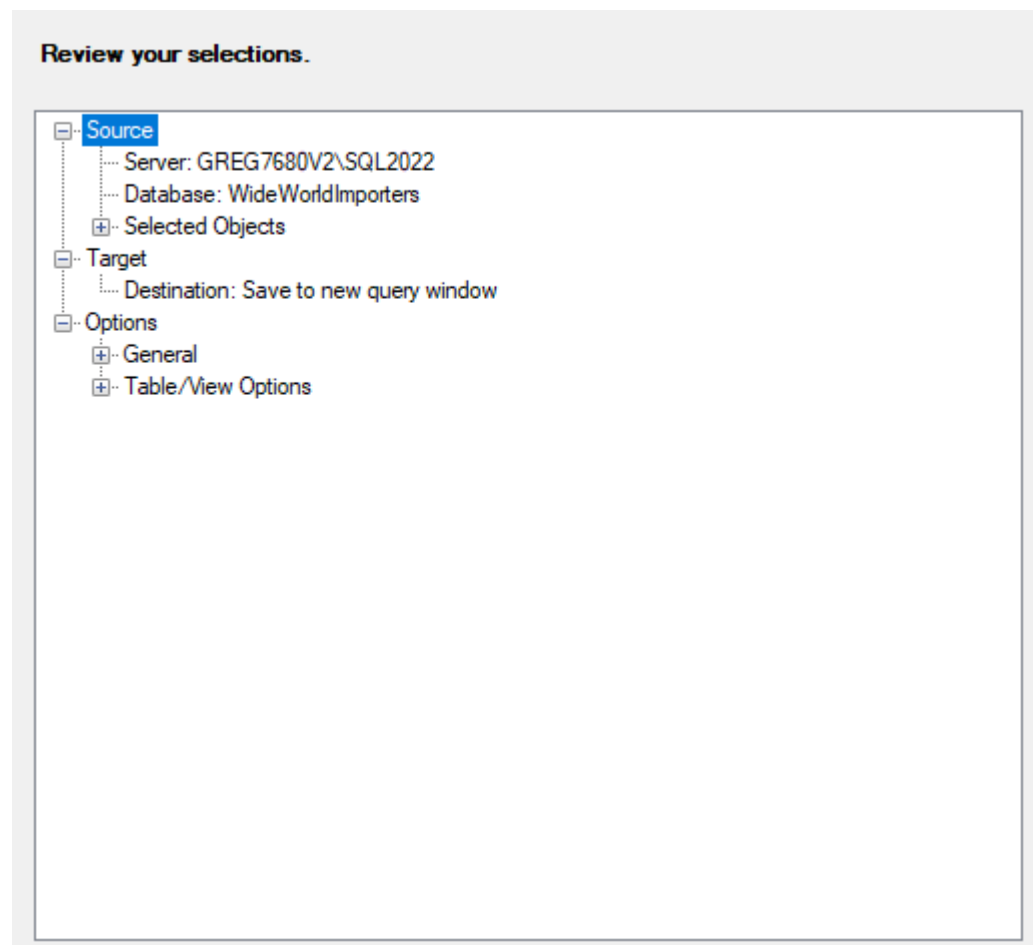
☒ Overwrite existing file

Save as: ☒ Unicode text  
☐ ANSI text

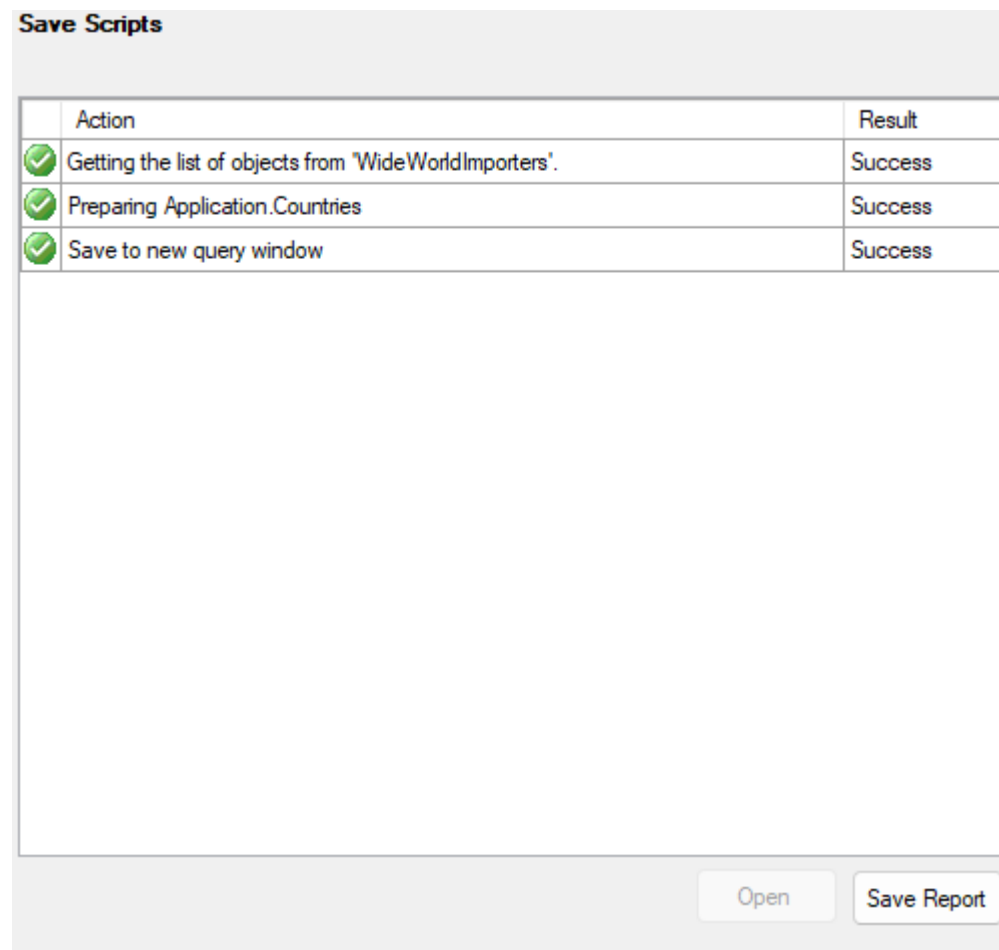
☐ Save to clipboard

☒ Open in new query window

I've then reviewed what's going to happen and clicked **Next** yet again.



On the final screen, I've noted the generation happening, then clicked **Finish**.



Then the window I was after has appeared:





I wish this was much simpler to do as it's a fairly common operation, certainly more common than many of the operations that are on the right-click context menu for a table. It's also not going to be suitable for large amounts of data but often that's not what you need to script.

And you might then want to use a SQL formatting tool to clean up the output window.

## 1.9 Turn off option to prevent saving changes that require table recreation

I don't use the table designer in SQL Server Management Studio. Sorry, but I just don't like it, or the options that it chooses for me. I'd rather use T-SQL every time, but I'm also the first to admit that there are plenty of people who would use that designer. And when they do, many run into an error that says:

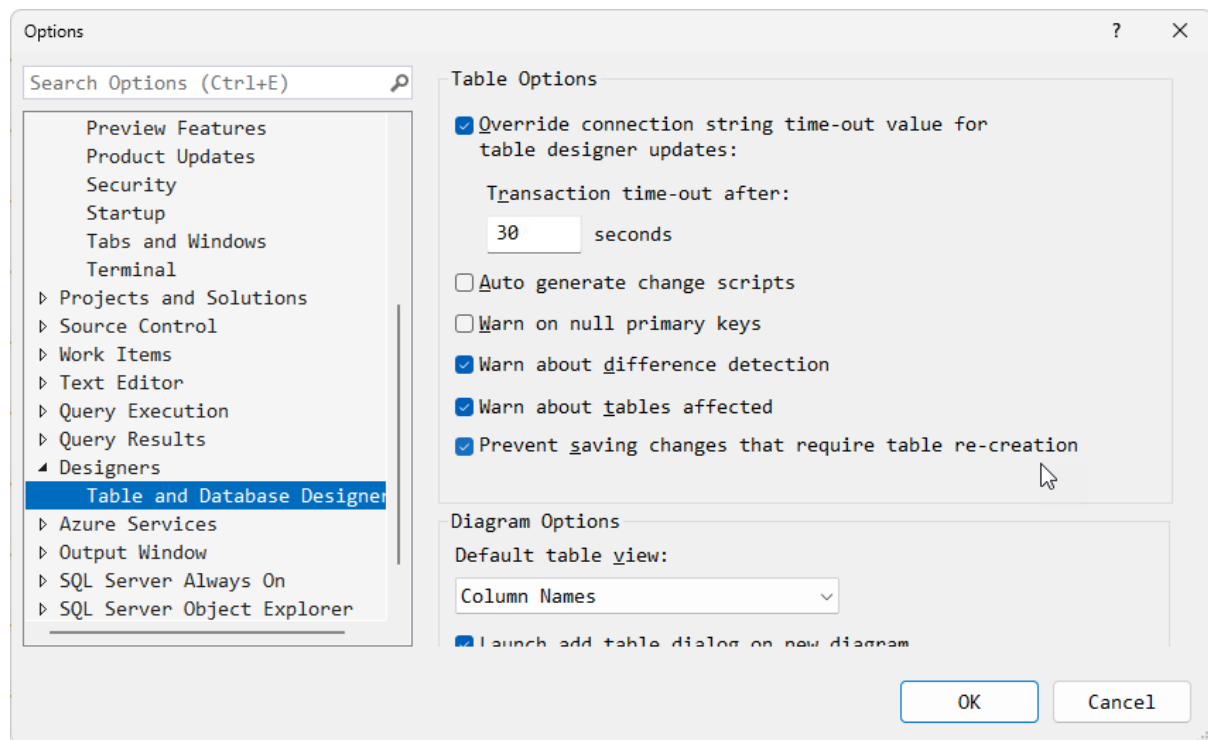
**Saving changes is not permitted. The changes that you have made require the following tables to be dropped and re-created. You have either made changes to a table that can't be re-created or enabled the option Prevent saving changes that require the table to be re-created.**

And they throw up their hands in horror. According to the online documentation, this is typically because they've done one of these types of things:

- Change the Allow Nulls setting for a column
- Reorder columns in the table
- Change the column data type
- Add a new column

Having to recreate a table for things like changing a column's data type seems way too heavy-handed. SQL Server Management Studio takes a safe option here. You can override it by doing this:

In Tools > Options > Designers > Table and Database Designers



While you can turn that off, be very careful if you decide to do so. You can cause issues, particularly if you are also using Change Tracking. (Existing tracking information is deleted when the table is recreated). You could also affect DDL triggers and many other things. For simple situations, this might help.

**As I mentioned, if it was me, I'd use T-SQL statements that do just the actions that I need.**

## 1.10 Dependency tracking

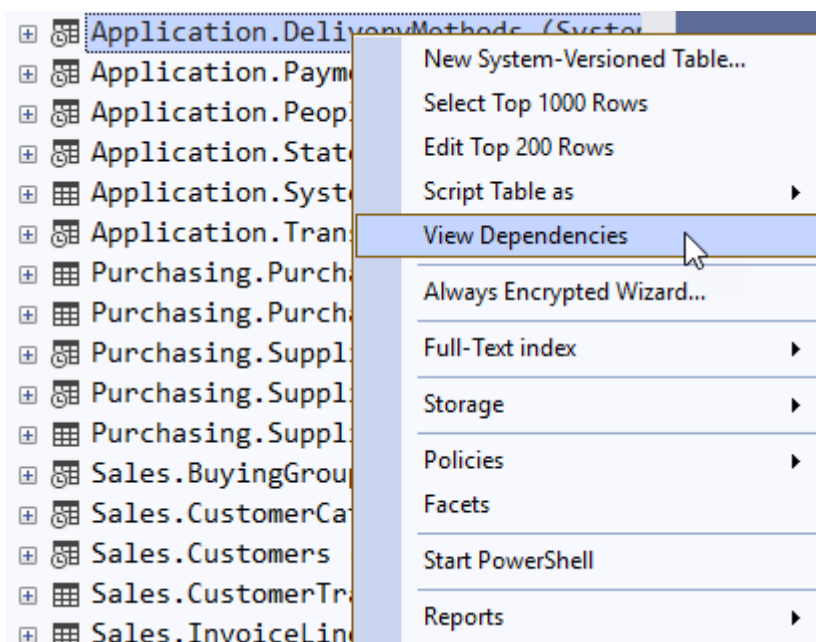
In early versions of SQL Server, the only way to try to track dependencies between tables, procedures, functions, etc. was to use the **sp\_depends** stored procedure.

**And everyone thought it lied.**

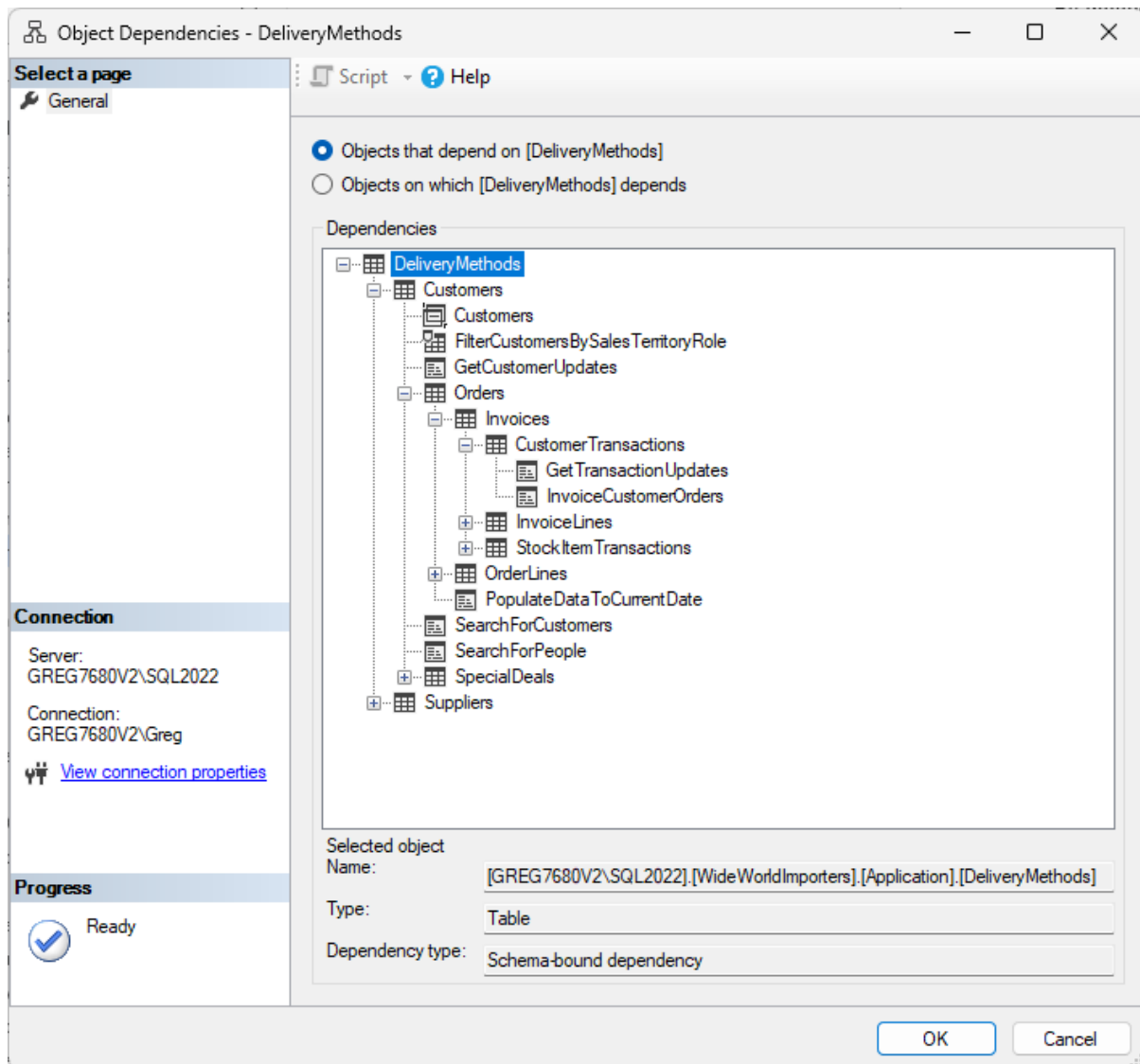
The real problem was that it didn't understand partial dependencies and deferred resolution of objects. For example, it got confused if you created a procedure that mentioned a table, then later created the table.

SQL Server 2012 introduced far superior dependency views, and SQL Server Management Studio (SSMS) shows dependencies using those views under the covers.

Here's an example. If I right-click the Application.DeliveryMethods table in the WideWorldImporters database, I can choose to **View Dependencies**:

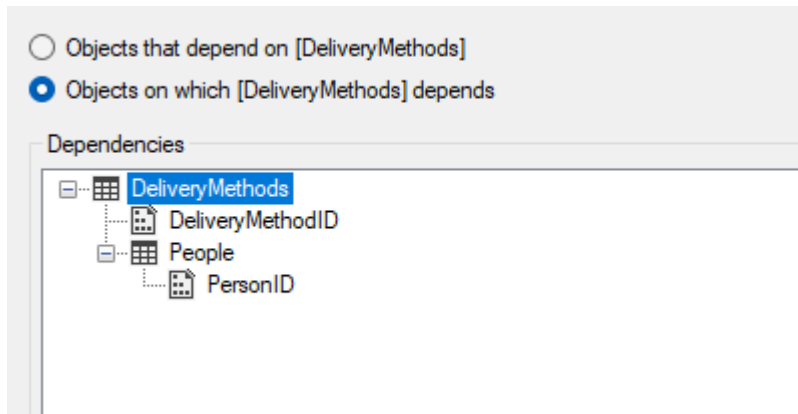


By default, you are shown the objects that depend upon the selected object (in this case the table that we right-clicked):



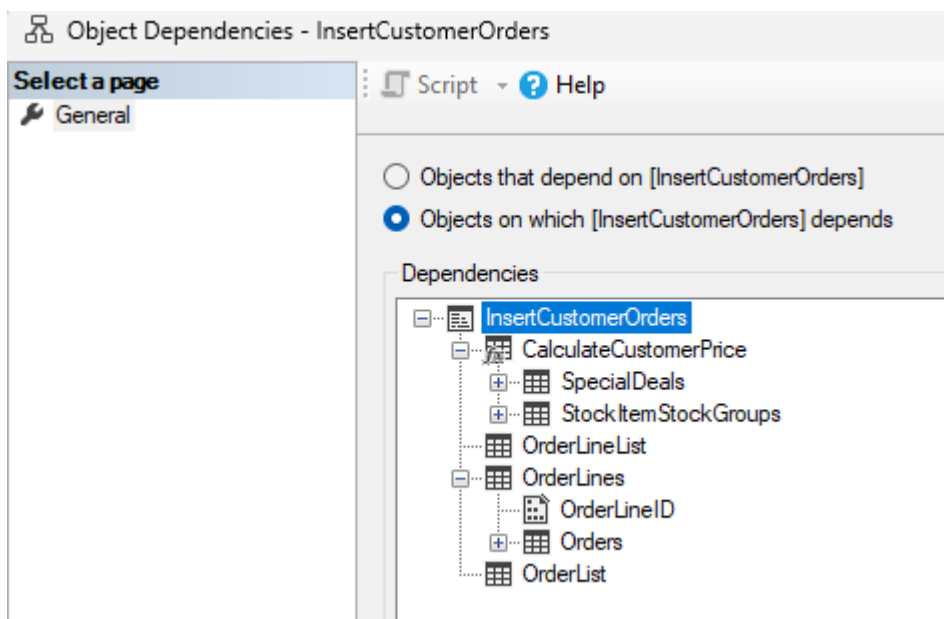
Note that this is a multi-layer dependency tree. We can see that the Customers table depends upon this table, as does the GetCustomerUpdates stored procedure. The Orders table also depends upon the DeliveryMethods table, and through that, the CustomerTransactions table, and from there, onto the GetTransactionUpdates procedure.

We can also see objects that the DeliveryMethods table depends upon:



In this case, we can see that the table depends upon its own primary key (ie: DeliveryMethodID), and on the People table because there is a foreign key to the PersonID column in that table, for the last person who modified the rows in the table.

We can also see dependencies for other types of objects. Here is the tree for the InsertCustomerOrders stored procedure:

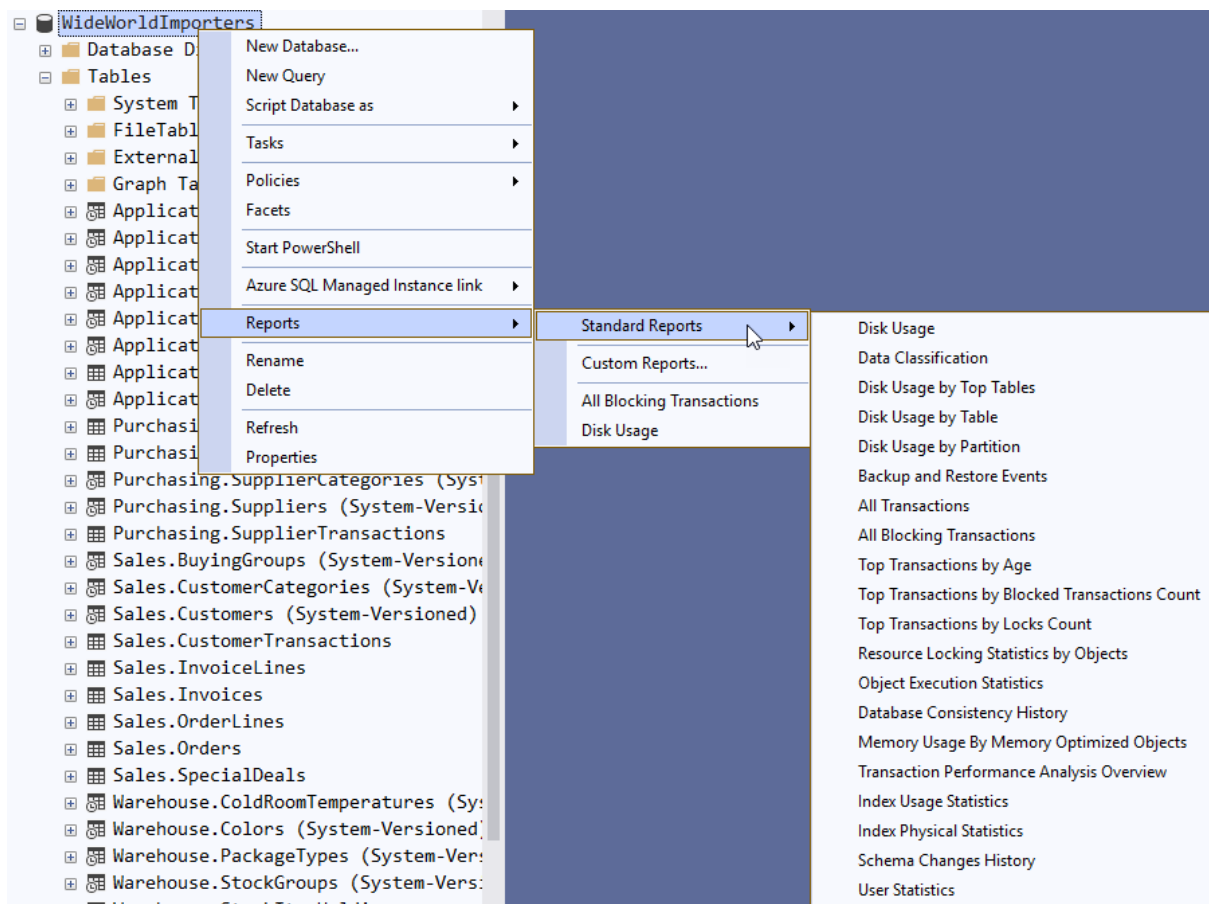


I'm really pleased that such a good dependency system is available within SSMS.

## 1.11 Built-in standard reports

SQL Server Management Studio provides a large amount of information about databases by letting the user navigate around Object Explorer and view the properties of objects.

What I'm often surprised by though, is the number of users who haven't ever explored the reports that are available. In another section, we'll look at creating custom reports, but there is a wonderful set of built-in standard reports that you should explore. For example, if I right-click the WideWorldImporters database, I can see these reports:



In this section, I just want to highlight a few of the most useful reports.

One of the common queries when databases get larger is about what's taking up all the space. That one is easy to answer by using Disk Usage by Top Tables. (Note: this report shows the top 1000 by usage, whereas Disk Usage by Table shows all tables).

## Disk Usage by Top Tables

[WideWorldImporters]

SQL Server

on GREG7680V2\SQL2022 at 19/05/2025 9:06:35 PM

This report provides detailed data on the utilization of disk space by top 1000 tables within the Database. The report does not provide data for memory optimized tables.

Table Name	# Records	Reserved (KB)	Data (KB)	Indexes (KB)	Unused (KB)
Sales.Invoices	70,510	105,360	90,856	14,128	376
Warehouse.ColdRoomTemperatures_Archive	3,654,736	78,408	76,040	592	1,776
Sales.OrderLines	231,412	71,928	38,240	33,200	488
Sales.InvoiceLines	228,265	53,288	41,552	11,504	232
Warehouse.StockItemTransactions	236,667	52,672	2,720	37,648	12,304
Sales.CustomerTransactions	97,147	26,248	8,944	12,536	4,768
Sales.Orders	73,595	11,752	5,504	6,104	144
Application.Cities	37,940	4,880	3,960	896	24
Purchasing.SupplierTransactions	2,438	4,296	248	688	3,360
Purchasing.PurchaseOrderLines	8,367	2,216	1,248	832	136
Application.Countries	190	1,952	1,744	72	136
Application.People_Archive	961	1,728	232	184	1,312
Warehouse.StockItems_Archive	444	1,080	168	112	800
Application.People	1,111	1,064	624	312	128
Sales.Customers	663	976	304	344	328
Application.StateProvinces	53	680	384	64	232
Application.Countries_Archive	36	656	480	16	160
Purchasing.Suppliers	13	576	8	120	448

These reports are all sortable by clicking the column headings. From this, we can see that Sales.Invoices contains the most data, and by clicking the # Records heading, we can see that Warehouse.ColdRoomTemperatures\_Archive is holding the most rows:

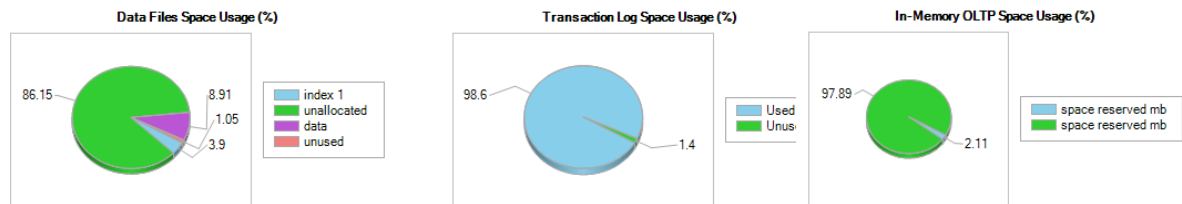
Table Name	# Records	Reserved (KB)	Data (KB)	Indexes (KB)	Unused (KB)
Warehouse.ColdRoomTemperatures_Archive	3,654,736	78,408	76,040	592	1,776
Warehouse.StockItemTransactions	236,667	52,672	2,720	37,648	12,304
Sales.OrderLines	231,412	71,928	38,240	33,200	488
Sales.InvoiceLines	228,265	53,288	41,552	11,504	232
Sales.CustomerTransactions	97,147	26,248	8,944	12,536	4,768
Sales.Orders	73,595	11,752	5,504	6,104	144
Sales.Invoices	70,510	105,360	90,856	14,128	376
Application.Cities	37,940	4,880	3,960	896	24
Purchasing.PurchaseOrderLines	8,367	2,216	1,248	832	136
Purchasing.SupplierTransactions	2,438	4,296	248	688	3,360

Another useful query is about how much free space there is, and if filegrowth events have been occurring. The standard Disk Usage report shows this:

Disk Usage		SQL S
[WideWorldImporters]		
on GREG7680V2\SQL2022 at 19/05/2025 9:10:41 PM		

This report provides overview of the utilization of disk space within the Database.

Total Space Reserved	6.43 GB
Data Files Space Reserved	3,072.00 MB
Transaction Log Space Reserved	2,084.00 MB
In-Memory OLTP Space Reserved	1.40 GB

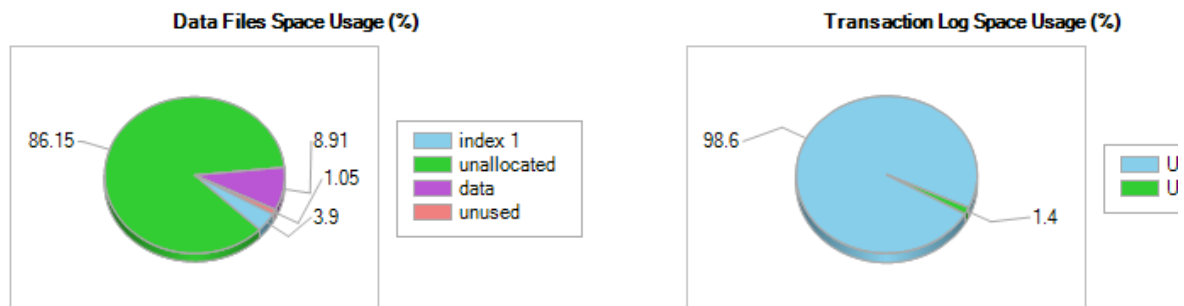


☐ Data/Log Files Autogrow/Autoshrink Events

☐ Disk Space Used by Data Files

☐ Disk Space Used by In-Memory OLTP Files

And you can expand the first line under the graphs to see autogrow or autoshrink events:



☐ Data/Log Files Autogrow/Autoshrink Events

Event	Logical File Name	Start Time	Duration (ms.)	Change In Size (MB)
Log File Auto Growth	WWI_Log	28/04/2025 11:59:29 AM	210	64.00

The Index Usage Statistics report puts the output of the sys.dm\_db\_index\_usage\_stats DMV into an easier-to-consume form.



Wondering who just made that recent schema change? The Schema Changes History report might help.

## Schema Changes History

[WideWorldImporters]

on GREG7680V2\SQL2022 at 19/05/2025 9:14:18 PM

This report provides a history of all committed DDL statement executions within the Database recorded by the default tr

### Schema Change History (Since 17/05/2025 5:41:20 PM ).

Shows changes made in the schema of the objects by DDL operations.

Object Name		Type		
[-] TestSchema		Schema		
DDL Operation	Time	Login Name	User Name	
DROP	19/05/2025 9:14:11 PM	GREG7680V2\Greg	Greg	
CREATE	19/05/2025 9:14:02 PM	GREG7680V2\Greg	Greg	

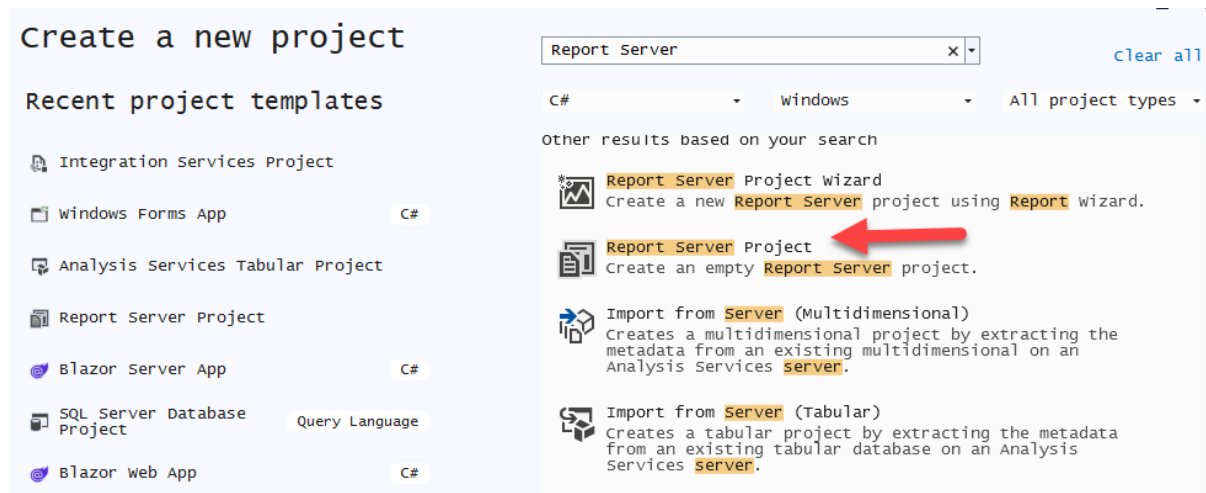
I'd encourage you to spend some time investigating what these reports offer.

## 1.12 Custom report creation

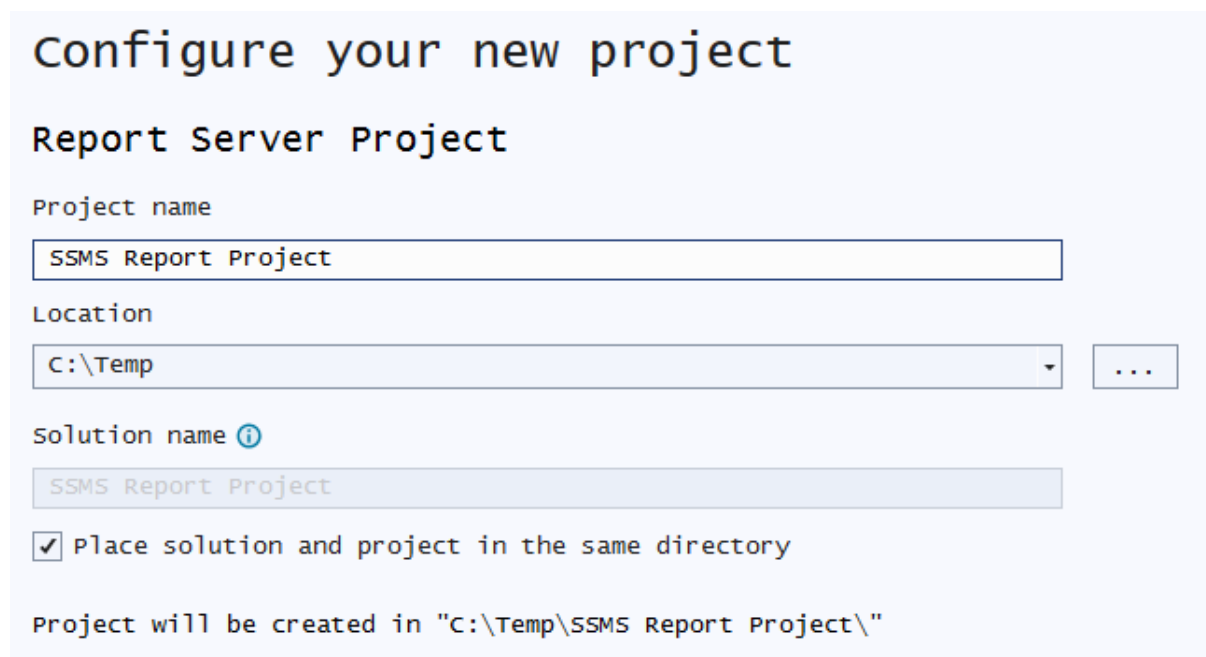
The built-in reports in SQL Server Management Studio are great but you can add your own as well. SQL Server 2005 and later have an option for Custom Reports.

Let's create a report that shows the use of deprecated data types. We'll use a stored procedure from our free SDU Tools to do that.

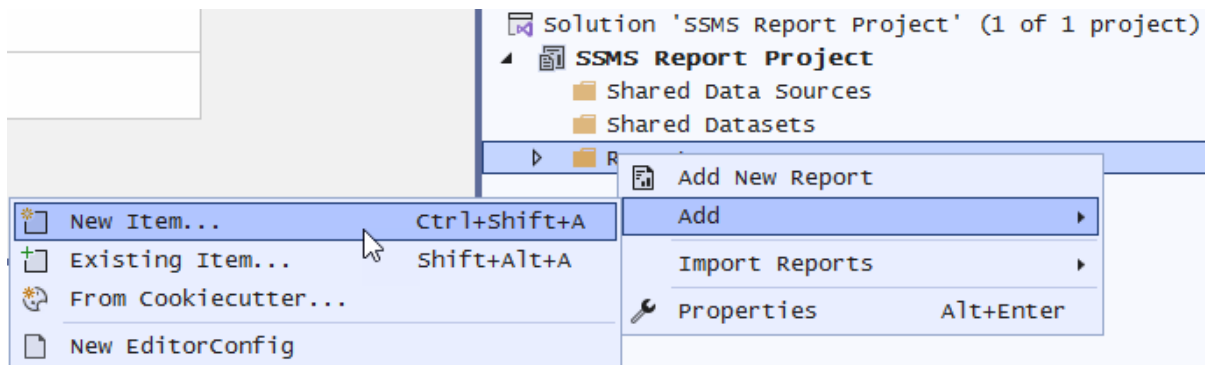
In SQL Server Data Tools, create a new Report Server project. (Note: not the Report Server Project Wizard)



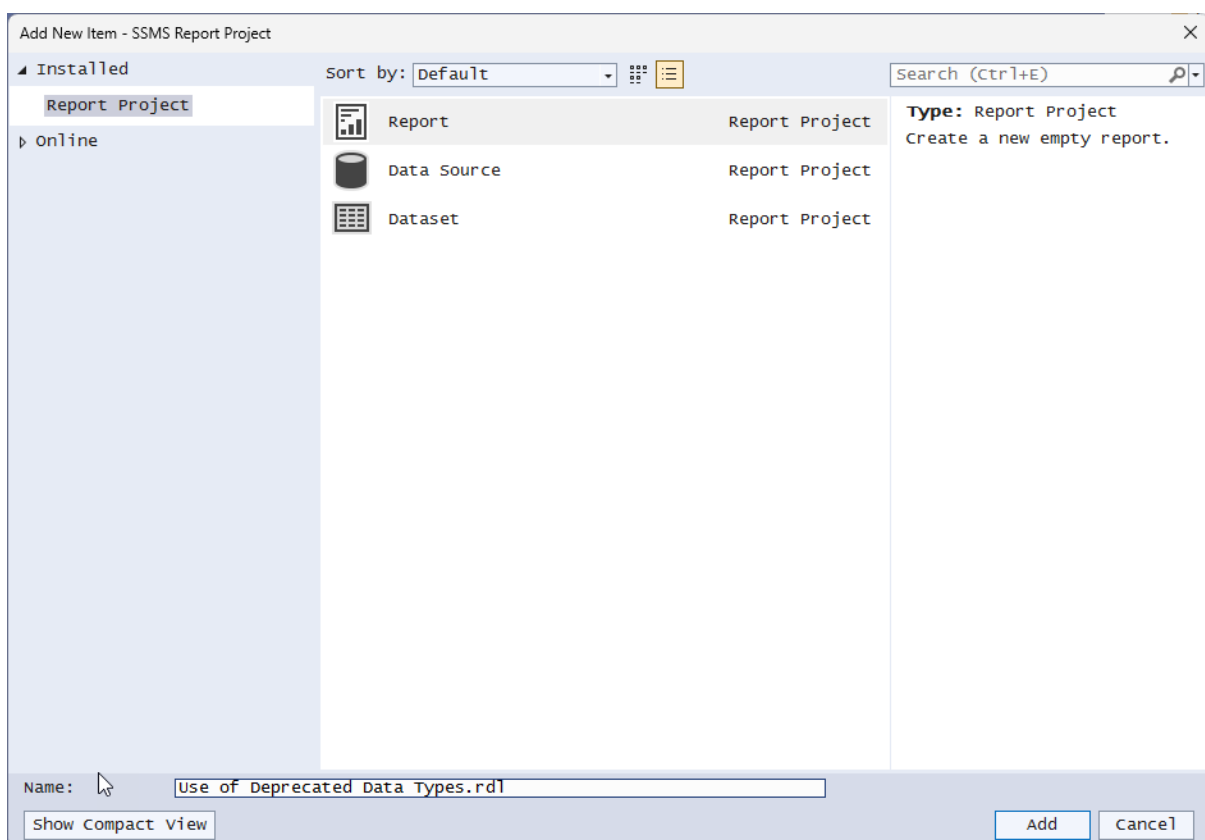
Give the project an appropriate name and location, then click Create.



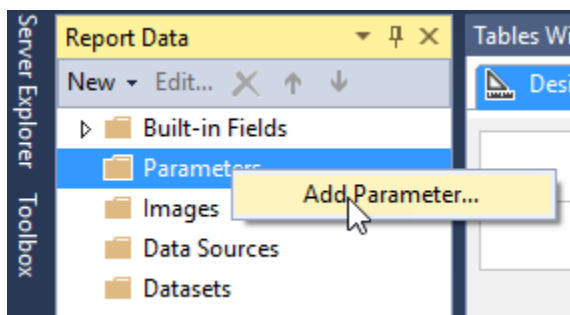
In Solution Explorer, right-click Reports, click Add, then click New Item. (Note: NOT Add New Report)



From the list of templates, choose to create a Report and give it a name then click Add.



There are a few standard parameters that are passed to your report by Object Explorer. In this case, we're only interested in the DatabaseName but we'll create them all anyway. In the Report Data pane, right-click Parameters and click Add Parameter.



Complete the parameter details as follows. Don't choose Allow null value. We're going to require a database name.

Report Parameter Properties

General

Available Values

Default Values

Advanced

Change name, data type, and other options.

Name: DatabaseName

Prompt: DatabaseName

Data type: Text

☒ Allow blank value ("")

☐ Allow null value

☐ Allow multiple values

Select parameter visibility:

☒ Visible

☐ Hidden

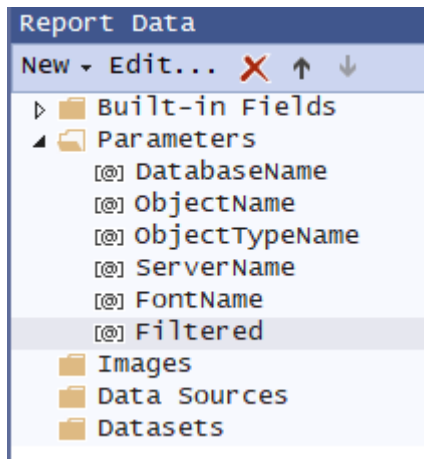
☐ Internal

Help OK Cancel

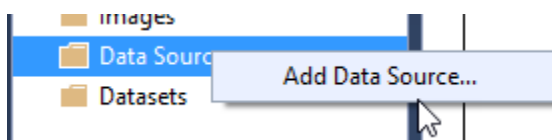
Create parameters for each of the following, ensuring that you do select Allow null value:

- ObjectName (Text)
- ObjectTypeName (Text)
- ServerName (Text)
- FontName (Text)
- Filtered (Boolean)

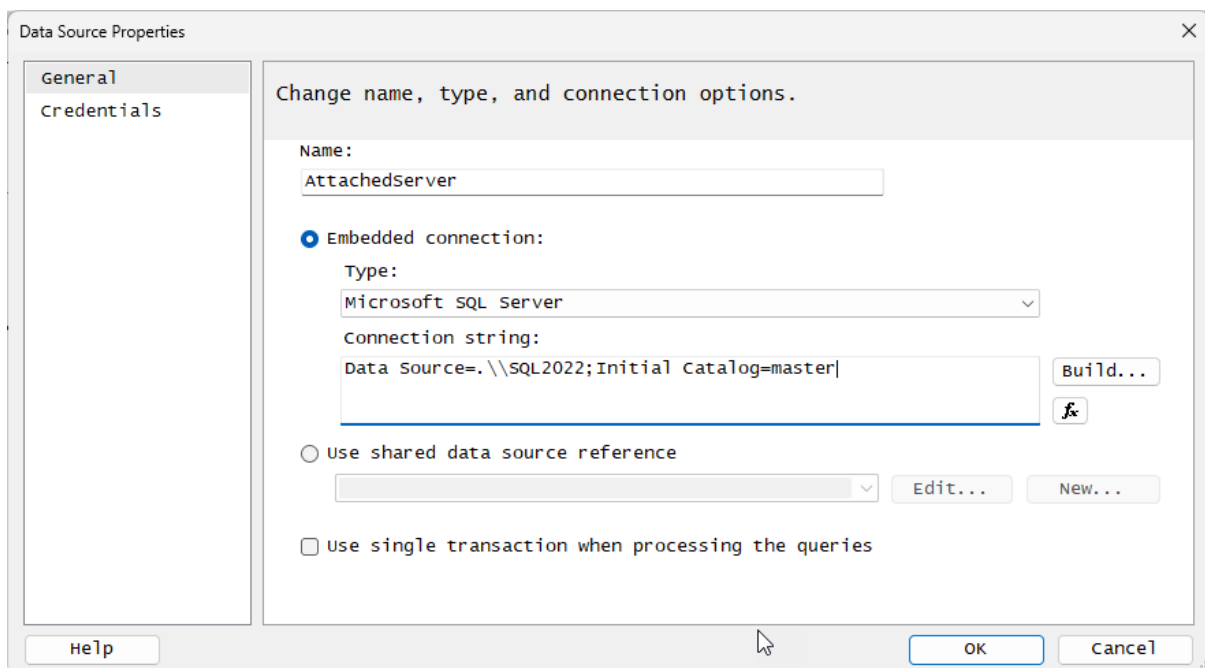
Your list should look something like this:



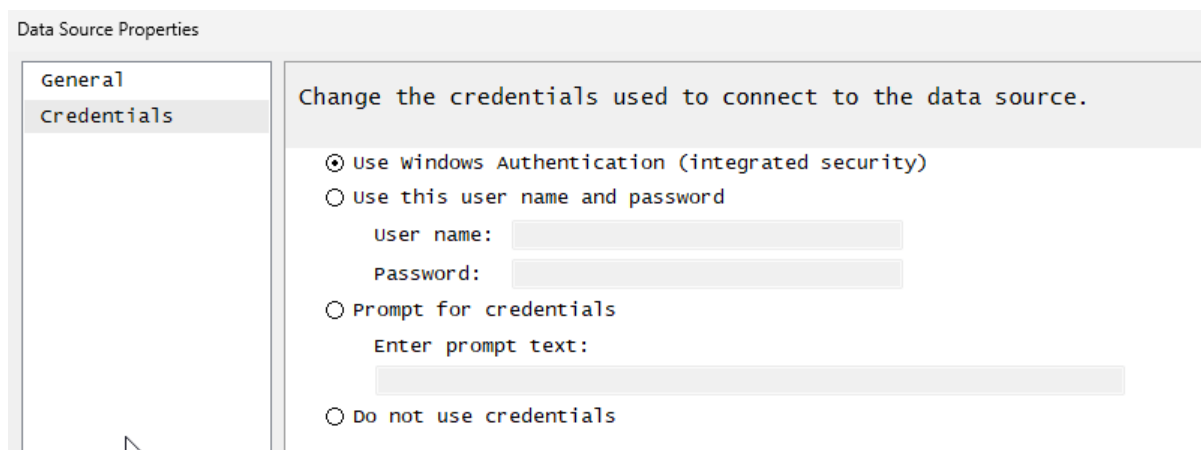
In the Report Data pane, right-click Data Sources, and click Add Data Source.



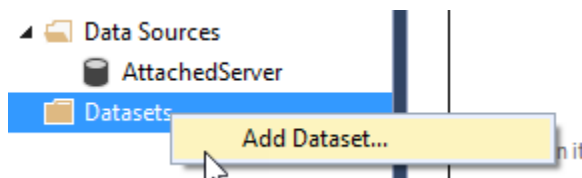
Configure the data source as follows (but note that your server connection might need to be different – mine was connected to a SQL Server 2022 server that I referred to as **.\\SQL2022** and is shown with a doubled backslash): Click OK to save it.



On the Credentials tab, choose the **Use Windows** Authentication option.



In the Report Data pane, right-click Datasets, and click Add Dataset.



Configure the dataset properties as follows:

Dataset Properties

Choose a data source and create a query.

Name:  
ListUseOfDeprecatedDataTypes

☐ Use a shared dataset.  
☒ Use a dataset embedded in my report.

Data source:  
AttachedServer New...

Query type:  
☒ Text ☐ Table ☐ Stored Procedure

Query:  
EXEC DATABASE\_NAME\_HERE.SDU\_Tools.ListUseOfDeprecatedDataTypes  
@DatabaseName = @DatabaseName;

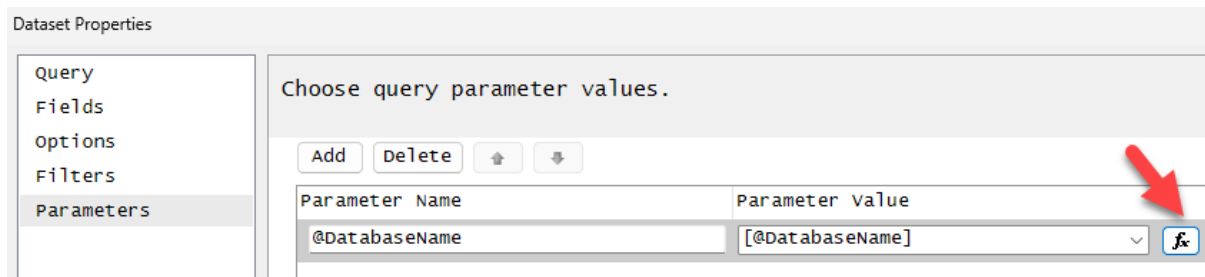
Query Designer... Import... Refresh Fields

Time out (in seconds):  
0

Help OK Cancel

Note that you will need to use whatever query you want to execute. In my case, the SDU Tool called ListUseOfDeprecatedDataTypes is stored in a database called DATABASE\_NAME\_HERE. (Yes that's the name of the database).

Also, for the dataset, make sure the Parameters collection is set like this:




Dataset Properties

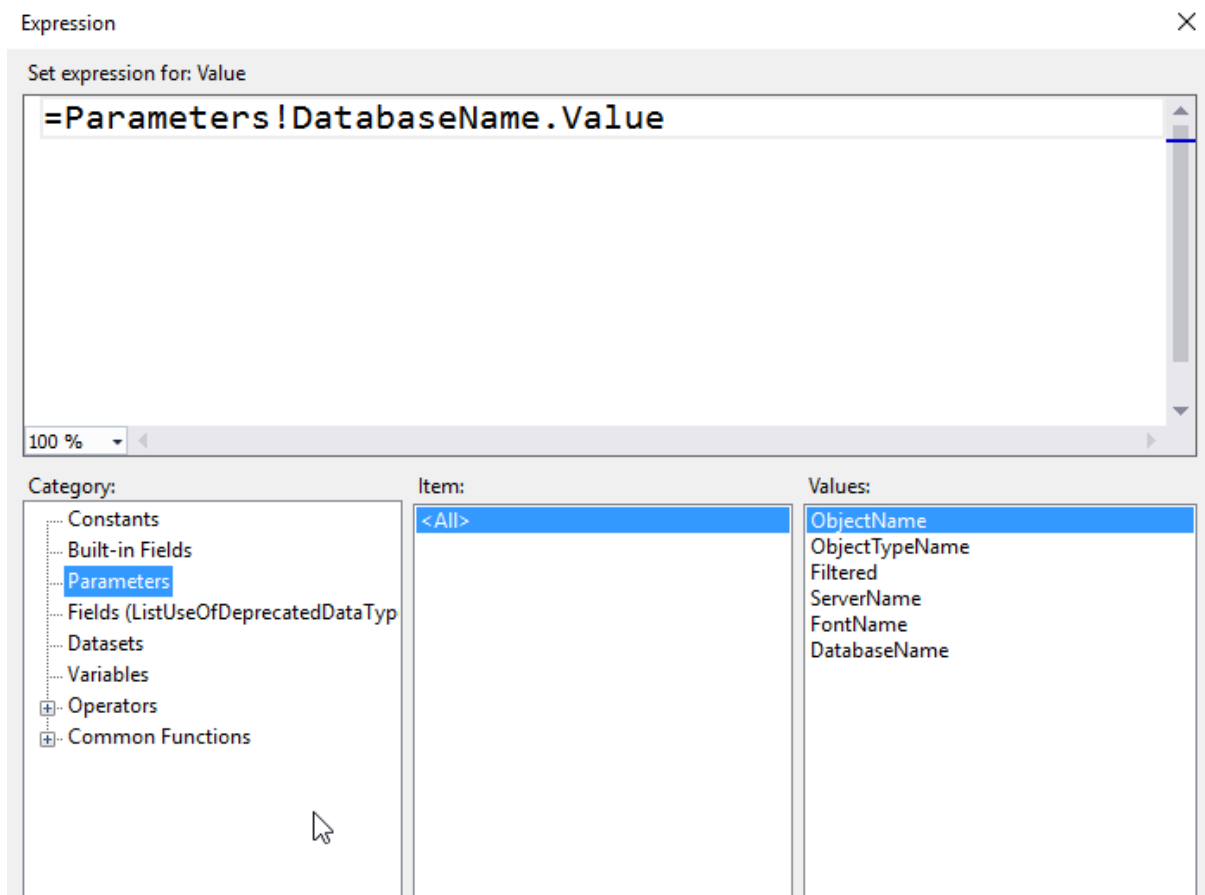
Query  
Fields  
Options  
Filters  
Parameters

Choose query parameter values.

Add Delete Up Down

Parameter Name	Parameter Value
@DatabaseName	[@DatabaseName] 

If you need to change it, use the function icon and do this:



Expression

Set expression for: Value

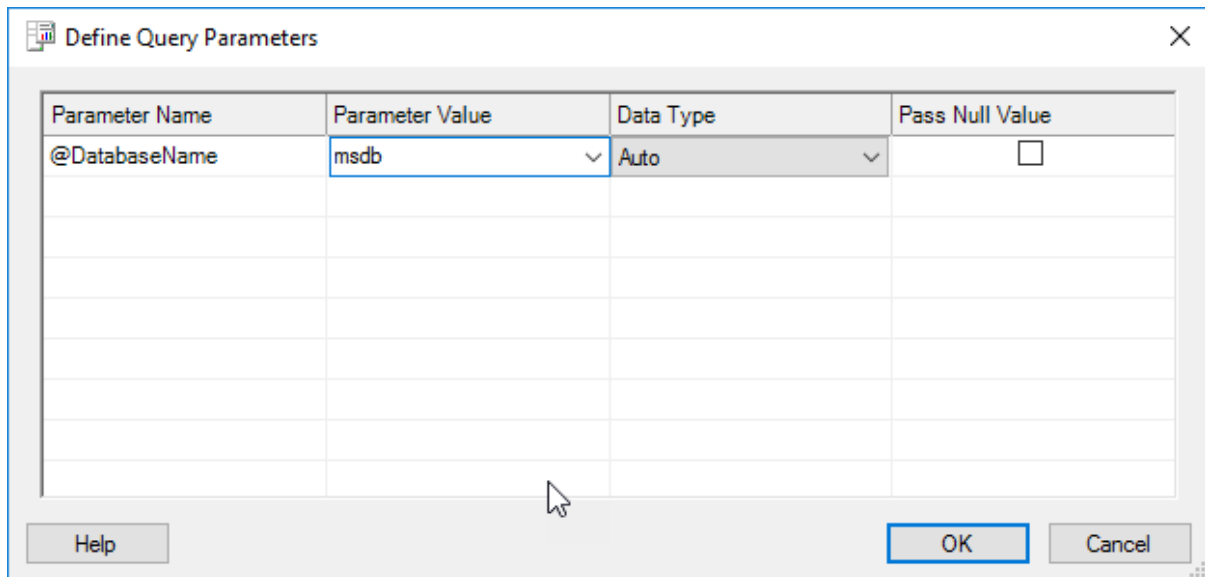
=Parameters!DatabaseName.Value

100 %

Category:	Item:	Values:
Constants	<All>	ObjectName
Built-in Fields		ObjectTypeName
Parameters		Filtered
Fields (ListUseOfDeprecatedDataType)		ServerName
Datasets		FontName
Variables		DatabaseName
Operators		
Common Functions		



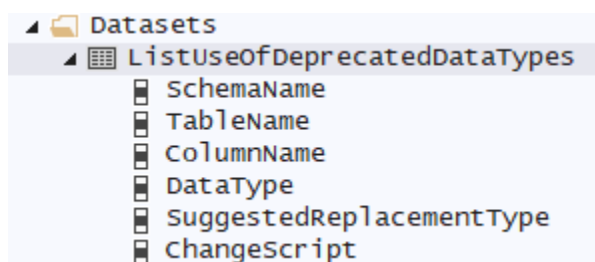
If you try to save it, you'll be asked for a parameter value so that the fields can be refreshed. Type msdb for the database name parameter.



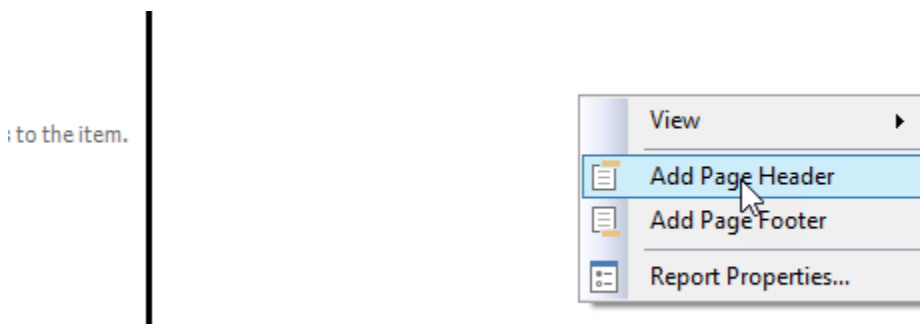
The 'Define Query Parameters' dialog box is shown. It contains a table with the following columns: Parameter Name, Parameter Value, Data Type, and Pass Null Value. The first row has the following values: @DatabaseName, msdb, Auto, and an unchecked checkbox. Below the table are buttons for Help, OK, and Cancel.

Parameter Name	Parameter Value	Data Type	Pass Null Value
@DatabaseName	msdb	Auto	<input type="checkbox"/>

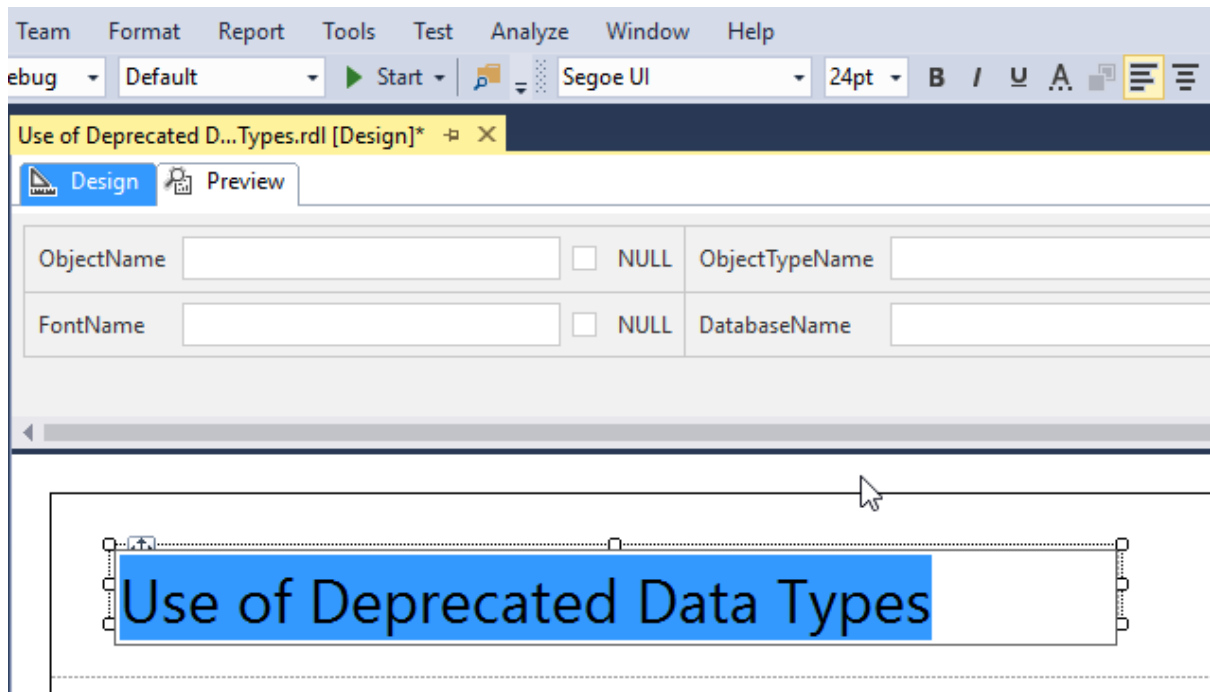
A list of fields should now be present if you expand the Dataset in the Report Data pane. The fields that are listed will depend upon your query.



Right-click in the open space of the report and click Add Page Header.



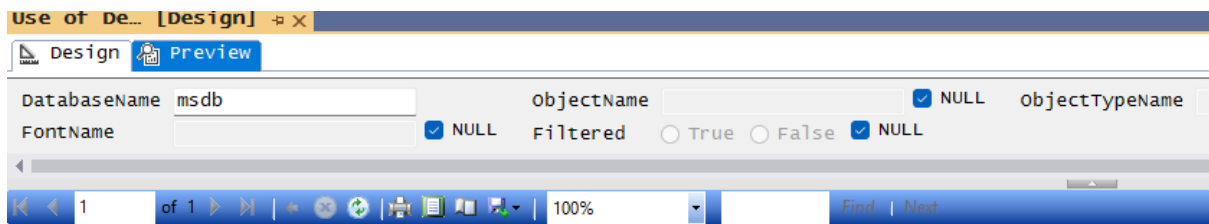
Drag a textbox from the Toolbox into the new header, enter a name, and change its size:



From the Toolbox, drag a table onto the body of the report, then drag the columns out, and resize the table. Bold the header line by selecting it and using the Font setting in the Properties window.

Use of Deprecated Data Types				
Schema Name	Table Name	Column Name	Data Type	Suggested Replacement Type
[SchemaName]	[TableName]	[ColumnName]	[DataType]	[SuggestedReplacementType]

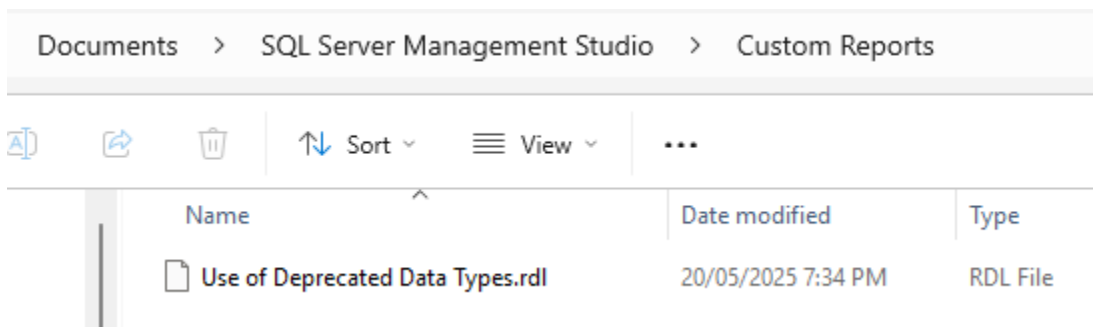
You should now be able to preview the report. Click the Preview tab, enter msdb for the database name, and click View Report.



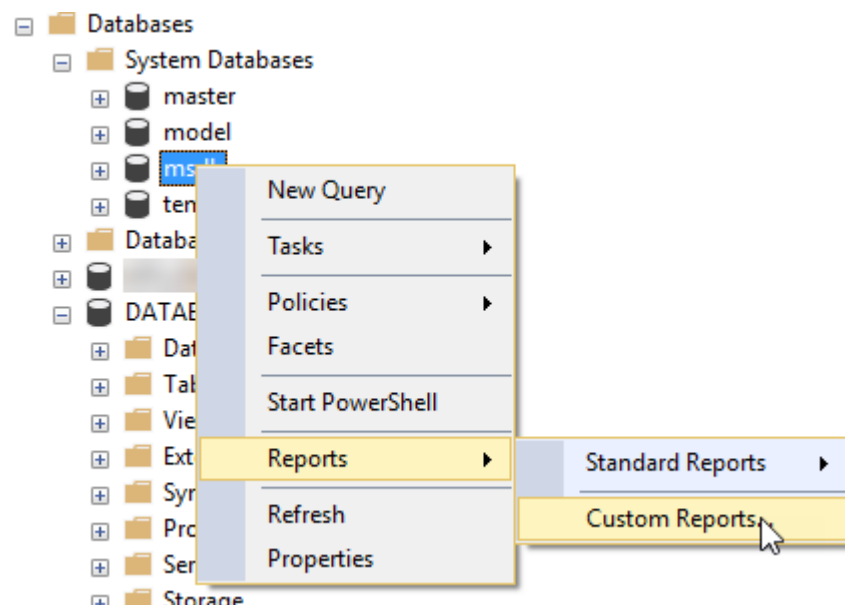
## Use of Deprecated Data Types

Schema Name	Table Name	Column Name	Data Type	Suggested Replacement Type
dbo	syslog	databytes	image	varbinary(max)
dbo	syslogpackages	packagedata	image	varbinary(max)

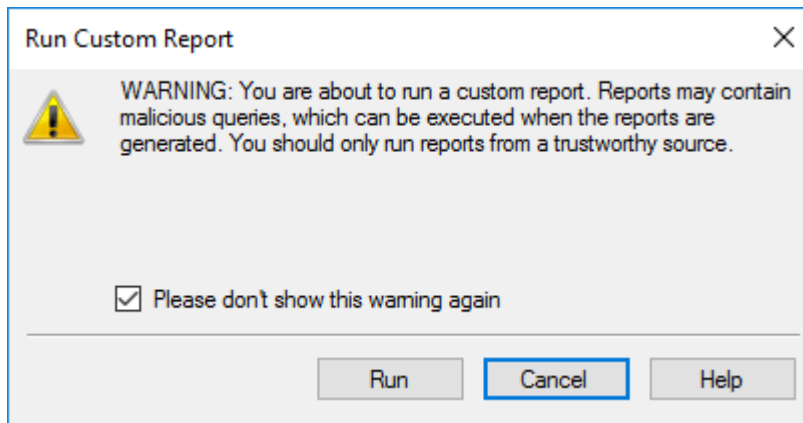
Now we're ready to push the report out. Copy the rdl file to the appropriate SSMS folder in your Documents.



Back in SSMS, right-click the msdb database, then click Reports, then Custom Reports.



You'll get a warning which you can choose to not show again.



Then your report should appear.

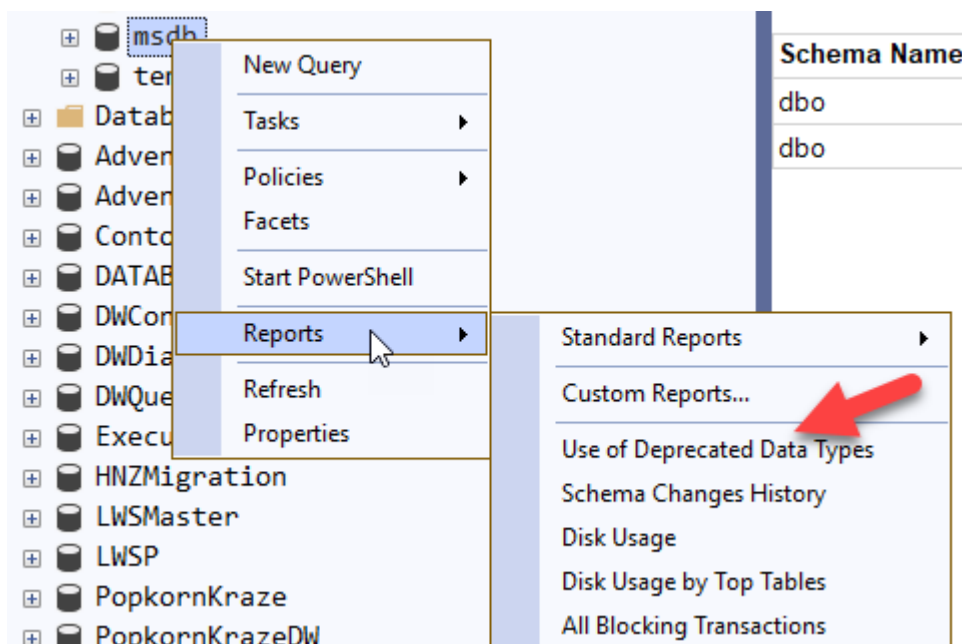


## Use of Deprecated Data Types

Schema Name	Table Name	Column Name	Data Type	Suggested Replacement Type
dbo	sysssislog	databytes	image	varbinary(max)
dbo	sysssispackages	packagedata	image	varbinary(max)

Try it on different databases and you'll see it works well. Note though that we haven't displayed the database name. That should be added to the page header but is left as an exercise for you to try.

Another thing to notice is that if you try to run it again, it appears in the drop-down list, for any object of the same type.



Custom Reports were a useful addition to SQL Server Management Studio.

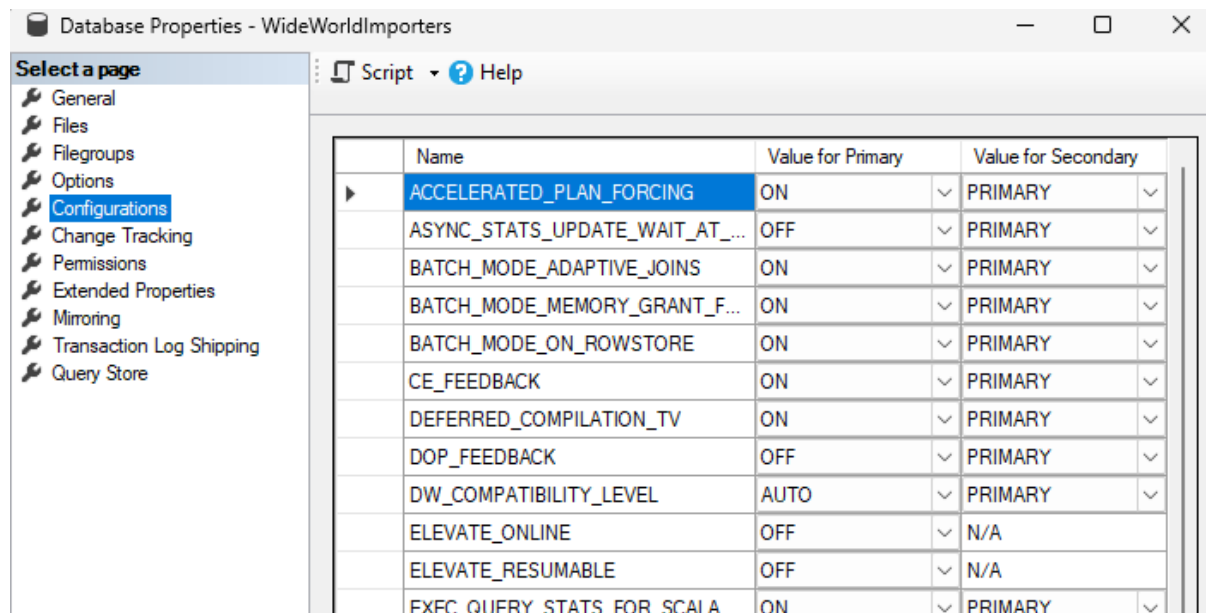
### 1.13 Database Scoped Configurations UI

The SQL Server team has been working towards making databases more standalone, and less dependent on the server, for quite a while.

SQL Server 2012 was the first version that had a concept of a contained database. The thinking was that you should be able to move databases around to different servers and, in that case, users would follow with the database because they didn't have a dependency on logins, which was the case up until then.

But users were only an early part of this story. Whether you want users contained or not, there are so many configurations at the server level that people wanted to be able to configure at each database, instead of only at the server. So, we saw the introduction of Database Scoped Configurations.

What was missing from this was a UI for setting them. SSMS now has a UI for configurations at the database level in the database properties from Object Explorer:



And because databases can be part of availability groups, there are options for setting values on the primary replica, and on secondaries.

All are scriptable as well. For example, if I change the CE\_FEEDBACK option on the primary to OFF, and instead of clicking OK to save it, I use the Script option at the top, to script to a new query window, I'll see the following. Can't say I love the generated script, but it would work.

```
USE [master]
GO

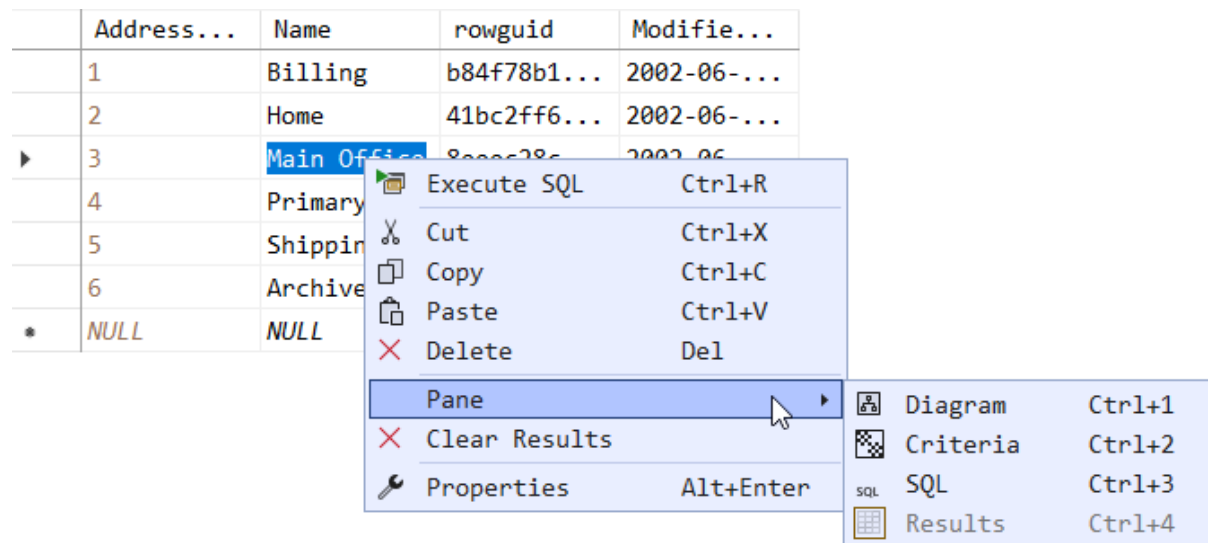
GO
USE [WideWorldImporters]
GO
ALTER DATABASE SCOPED CONFIGURATION SET CE_FEEDBACK = OFF;
GO
```

## 1.14 Pane options in Edit N Rows

(Thanks to Klaus Oberdalhoff for this one)

Klaus noted that when **EDIT TOP n** is used, only the result is displayed. That's different to the **SELECT TOP n** option where the query appears as well and can be edited.

However, after executing **EDIT Top n**, in the table displayed, right-click and you'll see a menu item for **Pane**.

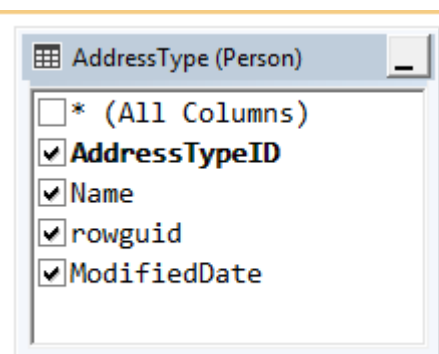


The sub-menu has the following options:

- **Diagram** - Shows the underlying table as a database diagram
- **Criteria** - Shows the graphical query editor
- **SQL** - The SQL script that was used to open the editor
- **Results** - Hides or shows the editable table

### Diagram

This option leads to a database diagram like this:



## Criteria

This option leads to a graphical query editor like this:

Column	Alias	Table	Output	Sort Type	Sort Order	Filter	Or...	Or...	Or...
Address...		Addr...	<input checked="" type="checkbox"/>						
Name		Addr...	<input checked="" type="checkbox"/>						
rowguid		Addr...	<input checked="" type="checkbox"/>						
Modifie...		Addr...	<input checked="" type="checkbox"/>						

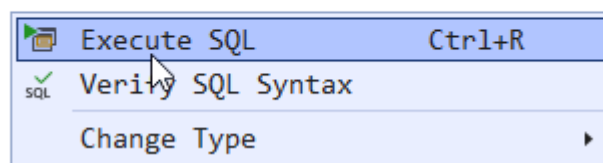
## SQL

No surprise that this one leads to a SQL query pane:

```
SELECT TOP (200) AddressTypeID, Name, rowguid, ModifiedDate
FROM Person.AddressType
```

After changing the SQL statement, you need to re-execute the SQL query. You can do that by right-clicking in the query pane, and clicking **Execute SQL**.

```
SELECT TOP (100) AddressTypeID, Name, rowguid, ModifiedDate
FROM Person.AddressType
```



## 2 Appearance

---

### 2.1 Environment Fonts

I've been very lucky over the years because I haven't needed to wear glasses. Every now and then I purchase a pair because I thought it might help with reading. Once I get them though, I find them more inconvenient than helpful and stop using them. I've am long-sighted in one eye and short-sighted in the other. That's turned out to be a really useful thing in day to day life.

However, where this comes unstuck is on modern laptops. There seems to be a current trend to pushing more and more pixels into the same size laptop screens, but the applications aren't helping to deal with that.



Image by Kevin

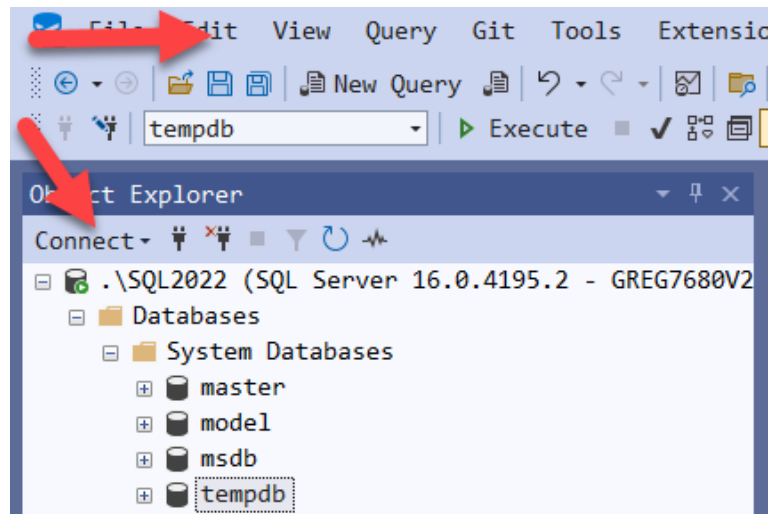
I'm sorry, but even a full 1080p screen (1920x1080) is crazy small on a 13-inch laptop, pretty bad on a 14-inch one, and OK on a 15-inch one.

Back in Windows 8, Microsoft tried to fix things by **automagically** scaling everything. I hated that with a passion. I'd move things from one screen to another and things would radically change size because even though the screens had the same resolution, they had different DPI (dots per inch) settings and Windows decided to auto-scale it for me. So, I found the option to kill that.

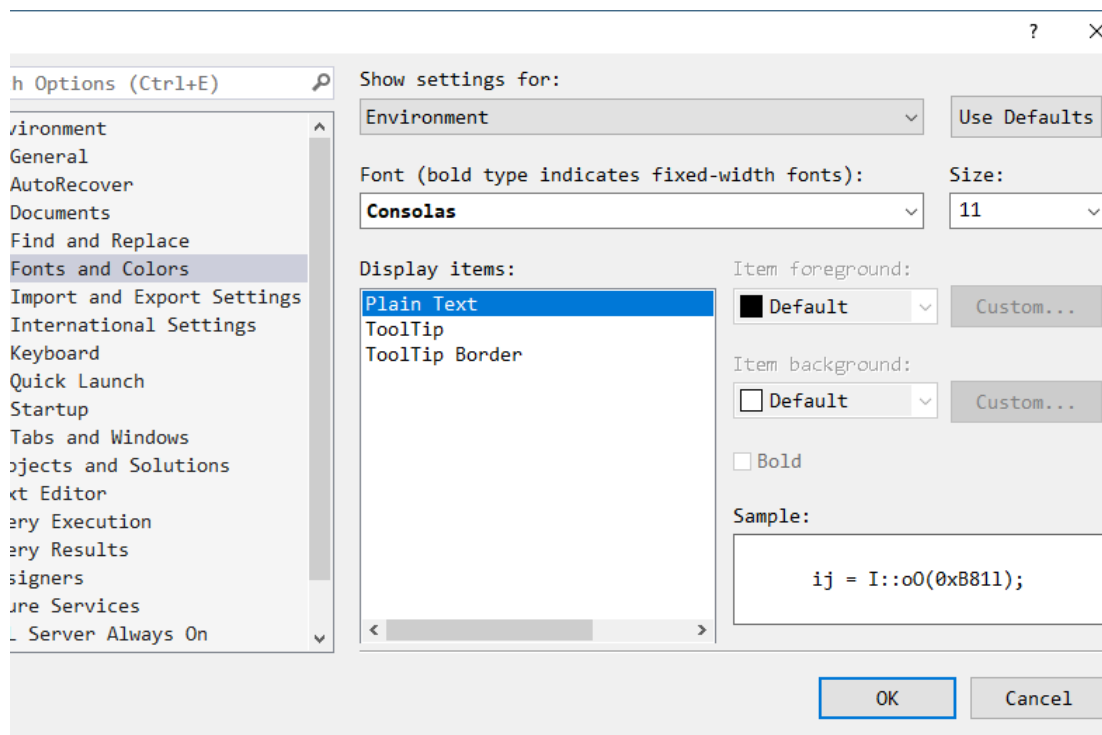
Ever since Windows 10, **SQL Server Management Studio** has been pretty usable for me. I can set fonts or use zoom as needed. The one thing that used to be a problem on small laptops though was the text used in areas like Object Explorer.



For some versions now, you've been able to fix that. Notice that on my screen, Object Explorer text is much more readable:



I've found that many people don't realize that in recent versions of SSMS, this can be changed. In Tools>Options>Fonts and Colors, you choose **Environment**:



(Notice that the text in this tool window is also larger and more readable)

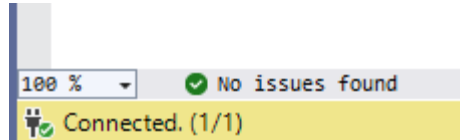
What you'll find though, is that you can't just directly change the size. You need to change the font first. In this case, I chose **Consolas**. Then, you can set the size.

I hope that helps someone else avoid glasses for just a little longer. A bonus is that it makes presentation screens look better to the audience as well.

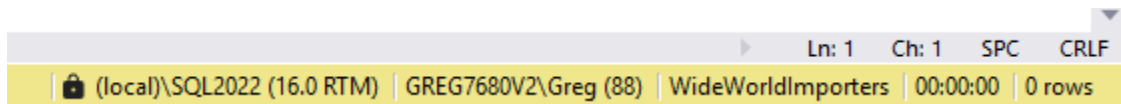
## 2.2 Changing displayed status bar values

The status bar at the bottom of a query window in SQL Server Management Studio contains a wealth of information in its default configuration.

The bottom left shows the connection state:



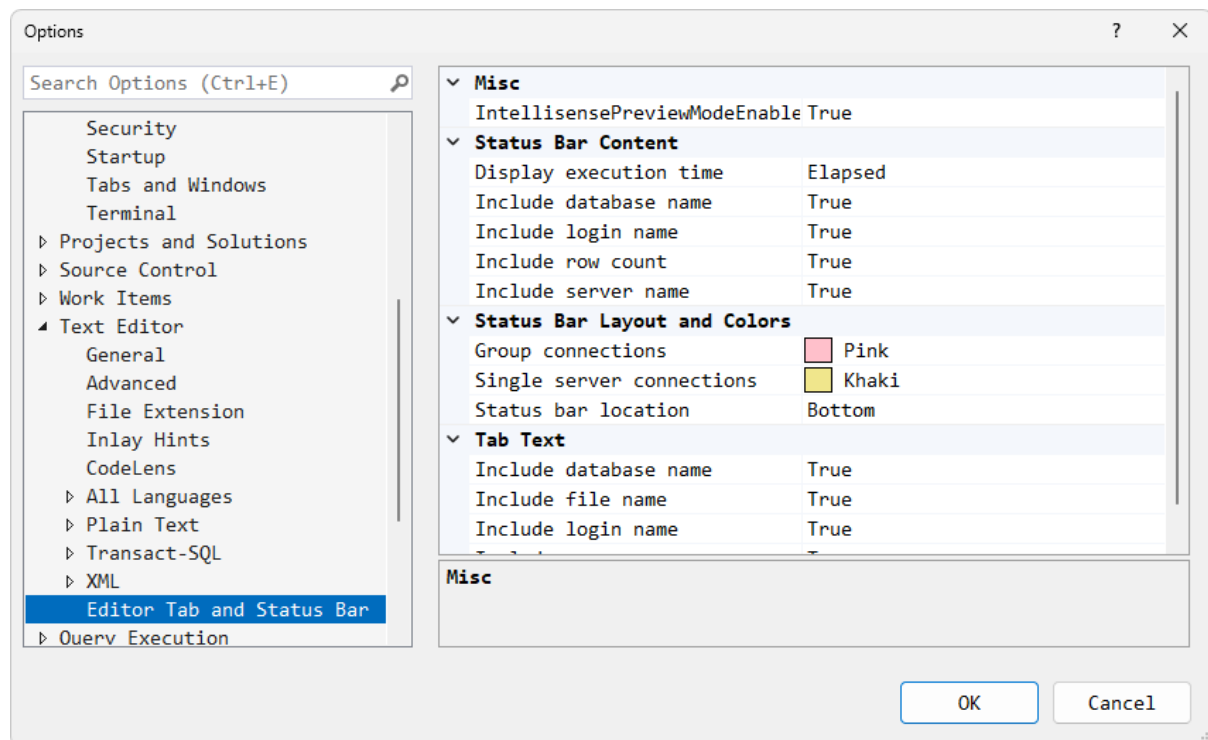
And the bottom right shows quite a bit:



In this case, it's showing me that I'm connected to a server (local)\SQL2022; it's running v16 of SQL Server (ie: SQL Server 2022); I'm logged on as GREG7680\Greg; my SPID (or session ID) is 88; and I'm connected to the WideWorldImporters database.

If I had a query running, the time would be counting up, and if a query had completed, it would show me the number of rows.

While this is useful, it's also configurable. The options are in Tools > Options> Text Editor > Editor Tab and Status Bar:



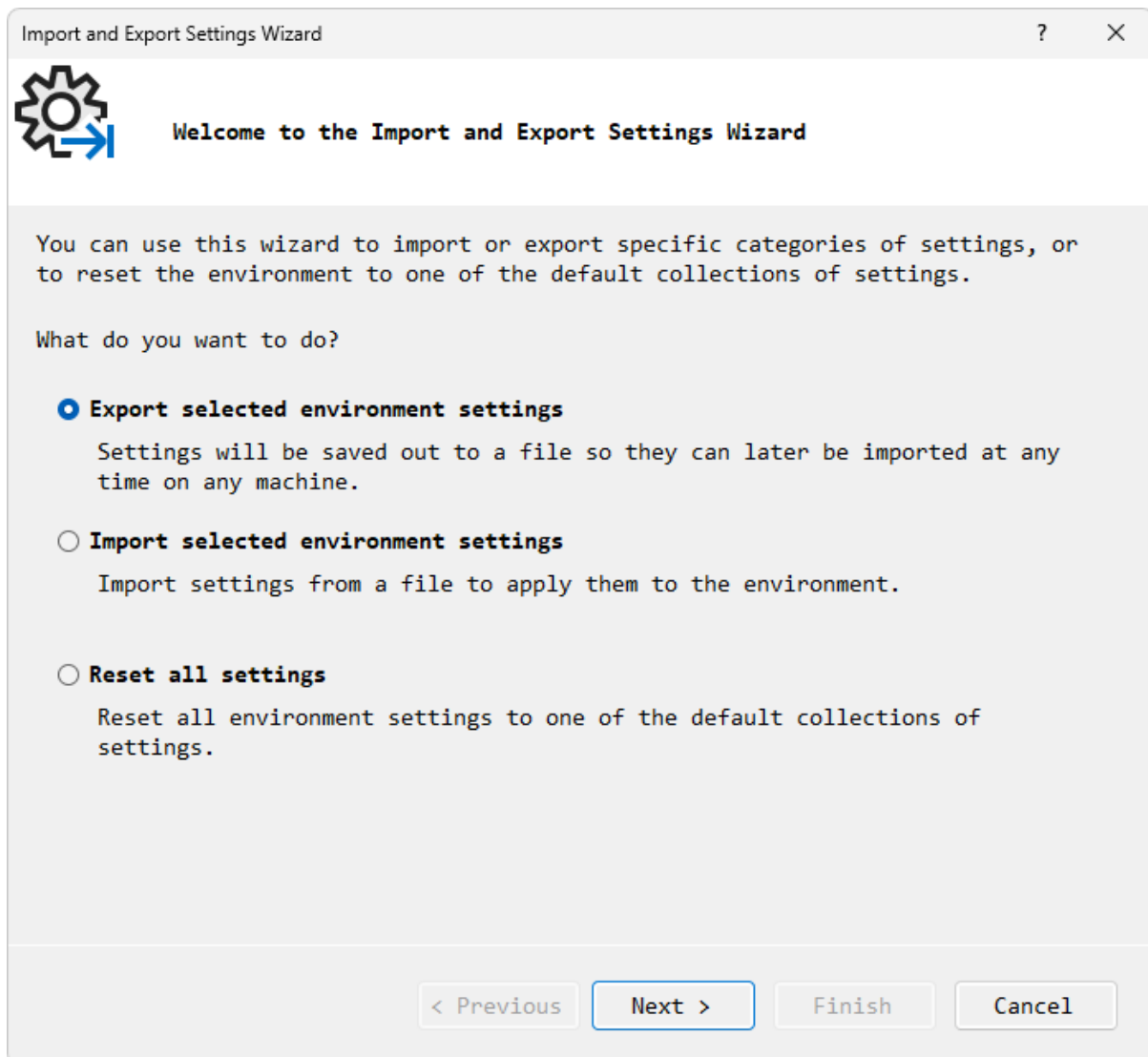
You can choose whether the displayed time is the elapsed time or the end time. You can also remove the display of the database, login, row count, and/or server name. You can set the color used for group and/or single server connections and choose where the status bar is located. You can move it to the top.

## 2.3 Import and Export settings

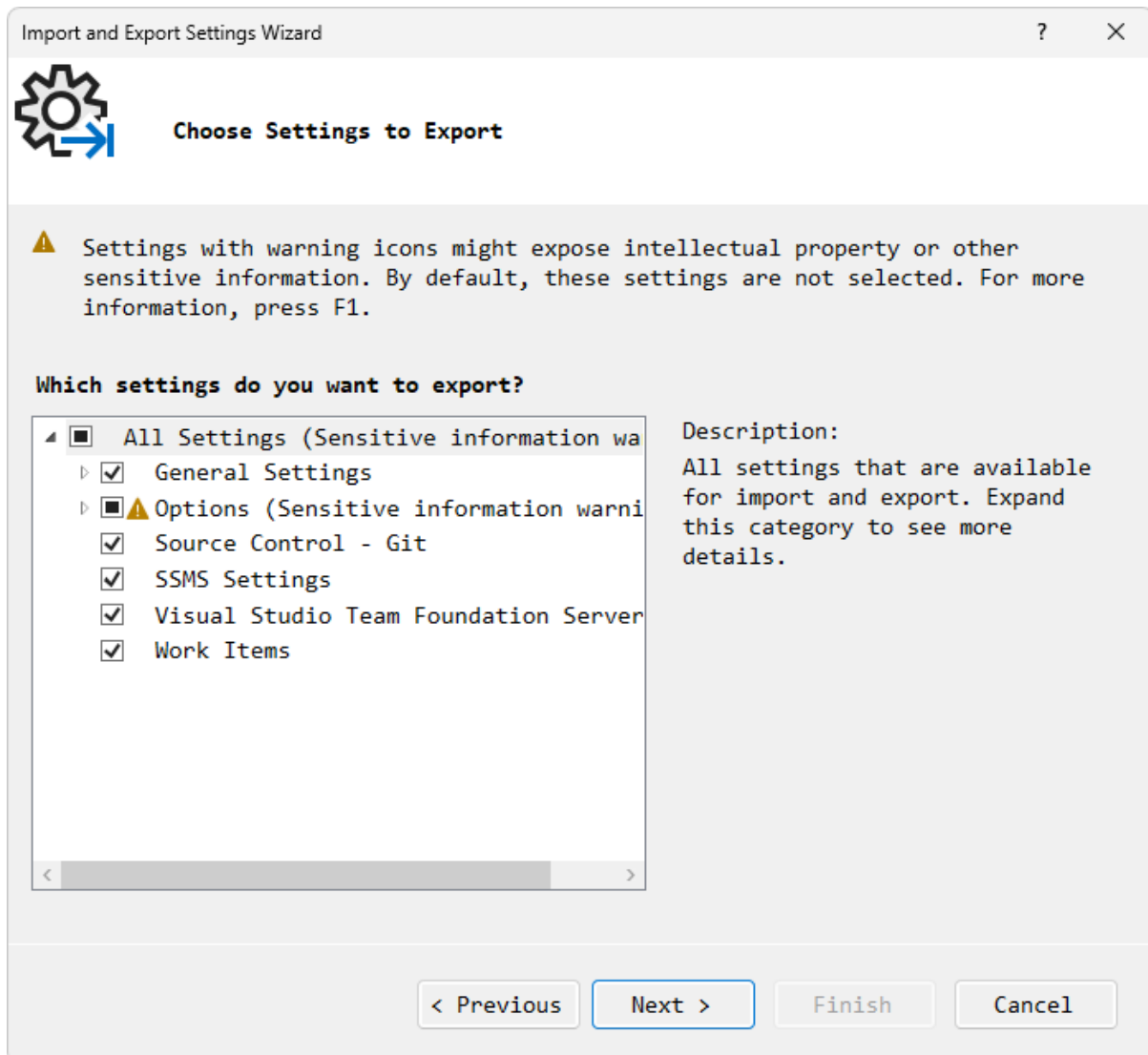
Whenever I need to work on a new laptop or server, or whenever I change versions of SQL Server Management Studio, I kick myself for not remembering to export my settings, so I can import them again.

I spend quite a bit of effort getting SSMS configured the way I want, so it only makes sense to save the settings. Saving them isn't perfect but it's far better than not having done it.

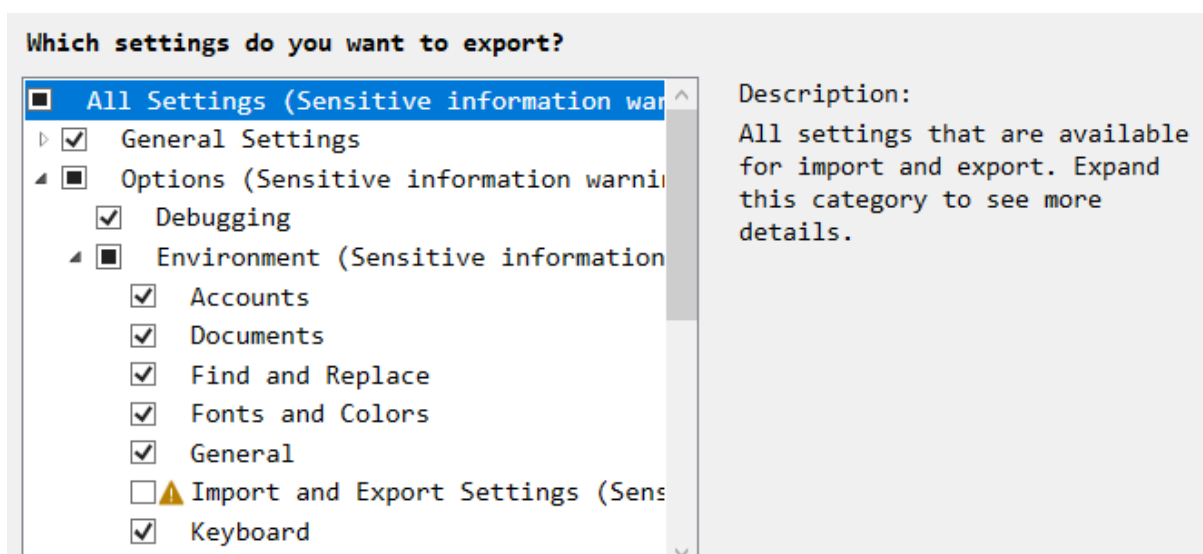
From the Tools menu, choose Import and Export Settings:



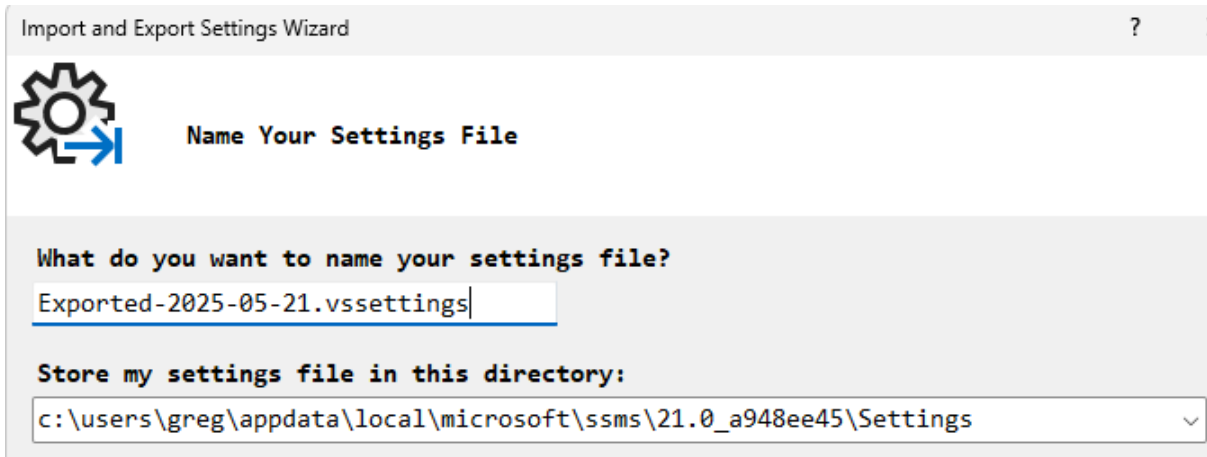
As an example, let's export the settings. I'll choose Next:




Notice that it's not an all or nothing export. I can choose details of which settings or groups of settings to export.



In this case, I wanted all of them, so I just need to pick a name and a location:



Import and Export Settings Wizard ?

 **Name Your Settings File**

**What do you want to name your settings file?**  
Exported-2025-05-21.vssettings

**Store my settings file in this directory:**  
c:\users\greg\appdata\local\microsoft\ssms\21.0\_a948ee45\Settings

And next time I change to a different machine or new version, I can just import them and pat myself on the back for thinking ahead.

## 2.4 Presentation Mode

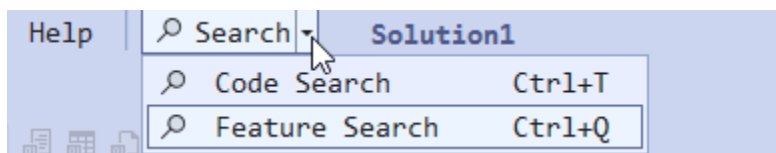
I spend a lot of time delivering presentations of various types. Many of those presentations involve showing code in either SQL Server Management Studio (SSMS) or Visual Studio (VS).

I've become quite fast at taking a default setup of SSMS and changing it to the fonts, etc. that I want to use for a presentation. Given how large these fonts are, I don't want to use them for day-to-day work.

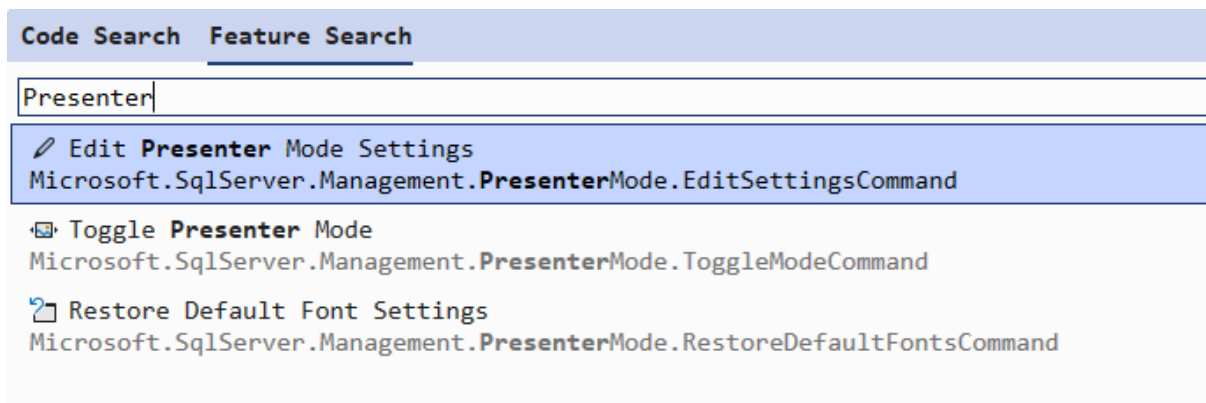
**The best solution that I've found for this is to create another user (let's call it DemoUser) on my laptop, and then configuring fonts, etc. for presentations for that user, quite separate to my normal work fonts.**

In recent builds of SSMS, the team realized the importance of this and added a Presentation mode.

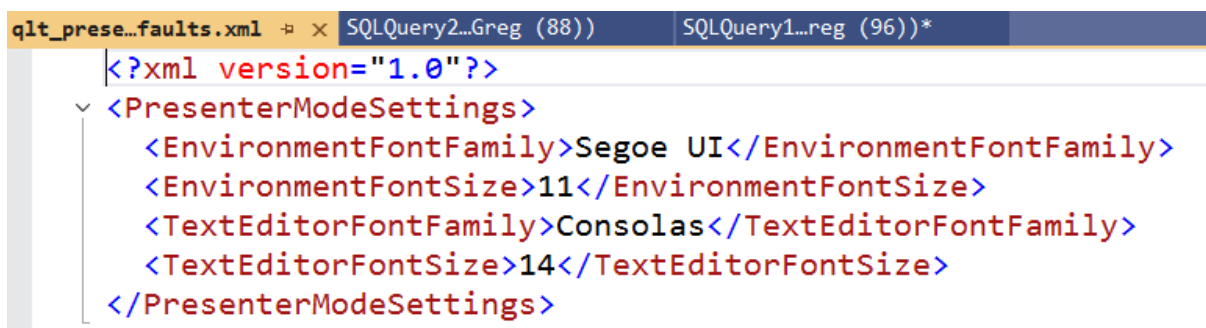
You can find it as part of the Feature search option.



When you open that, you get the Search screen that has both Code and Feature searches available, and it will have preselected Feature Search. I then entered the word Presenter.



After selecting the option to Edit Presenter Mode Settings, I see:

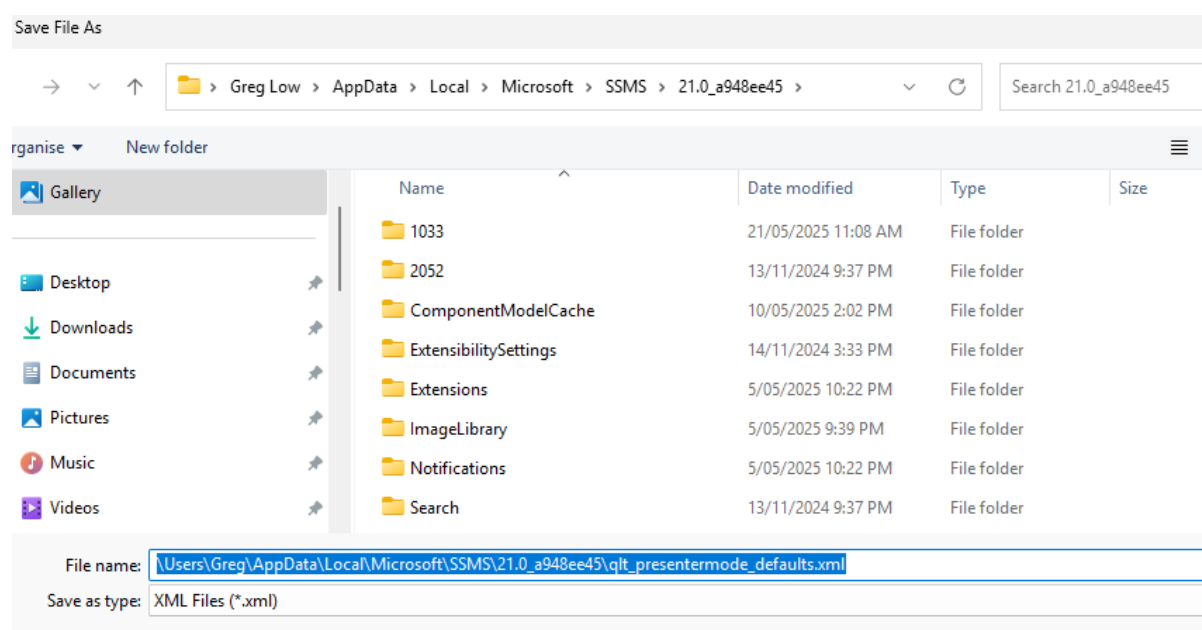


You'll note that an XML file appears. It would be helpful if this was a GUI instead of an XML file because you'll need to know what the names of the other settings are, if you want to change them.

For now, let's just change the `TextEditorFontSize` and the `EnvironmentFontSize` using the perfectly-fine XML editor in SSMS. The first entry changes the size of the text when you're editing queries. The second one affects the size of text in Object Explorer, menus, etc.

```
<?xml version="1.0"?>
<PresenterModeSettings>
  <EnvironmentFontFamily>Segoe UI</Envir
  <EnvironmentFontSize>12</EnvironmentFo
  <TextEditorFontFamily>Consolas</TextEd
  <TextEditorFontSize>16</TextEditorFont
</PresenterModeSettings>
```

If I click File, then Save As, note where this is saved:



It's under your **AppData** folder. Once this is saved though, we can test it. Once this is saved though, we can test it. In the same Feature Search screen, just double-click the option to toggle the presenter mode, and you should see the change. Do it again to set it back.

If you change output grid or text sizes, you'll still need to restart SSMS to see the outcome. The only difference is that you won't get the warning that you normally do.

The option to toggle is great. Previously there was only an option to set it back to the default values, which wasn't helpful..

## 2.5 Screen and Printing Colors

SQL Server Management Studio (SSMS) is a highly configurable tool. One of the areas that's often ignored but which can be quite important is color configuration.

SSMS color codes SQL scripts (and other types of files that it understands) as you type.

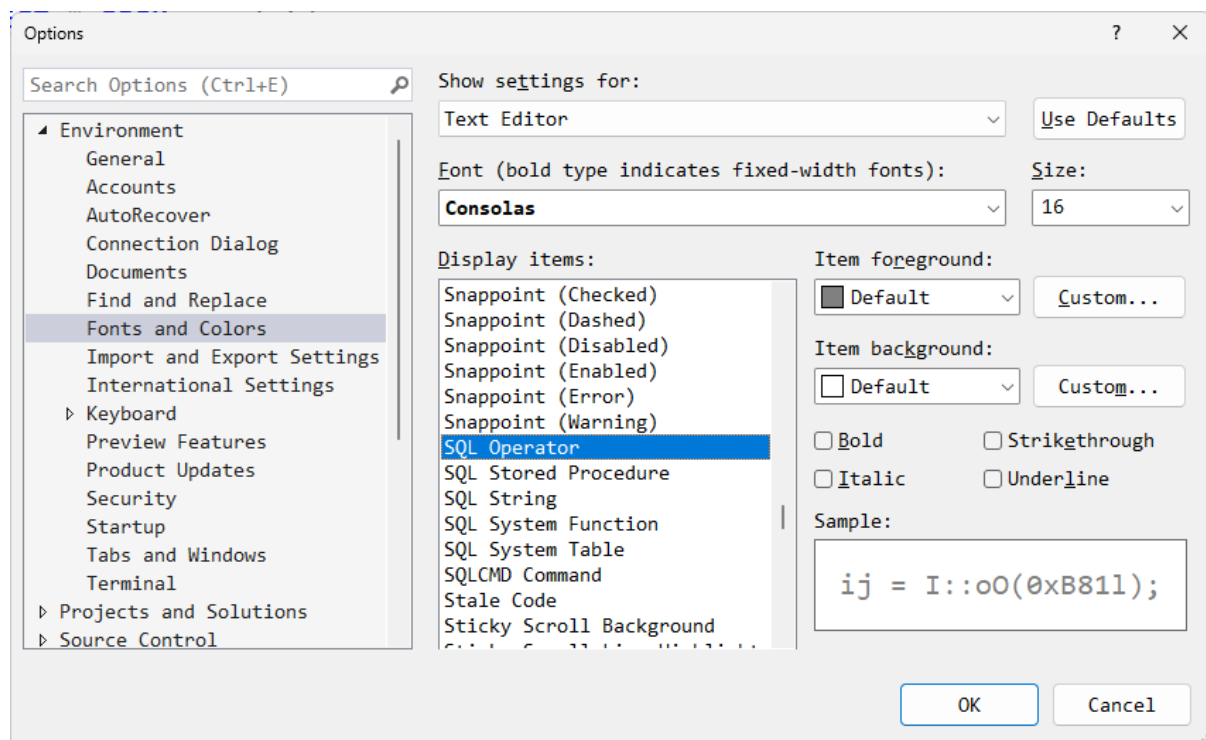
```
/*  
  
SELECT SDU_Tools.CalculateAge('1968-11-20', SYSDATETIME());  
SELECT SDU_Tools.CalculateAge('1942-09-16', '2017-12-31');  
  
*/  
RETURN (CAST(CONVERT(char(8), @CalculationDate, 112) AS int) I  
        - CAST(CONVERT(char(8), @StartingDate, 112) AS int)) / 10000;  
END;
```

This is useful but I've found on some systems that some of the color selections aren't great. Here's an example:

```
SELECT @@VERSION;  
SELECT * FROM sys.tables ;
```

On many systems that I work with, depending upon the version, the color for **sys.tables** in the query above is quite a fluoro green and almost unreadable. The default from v21 onwards is much better but if you don't like it, you can change it.

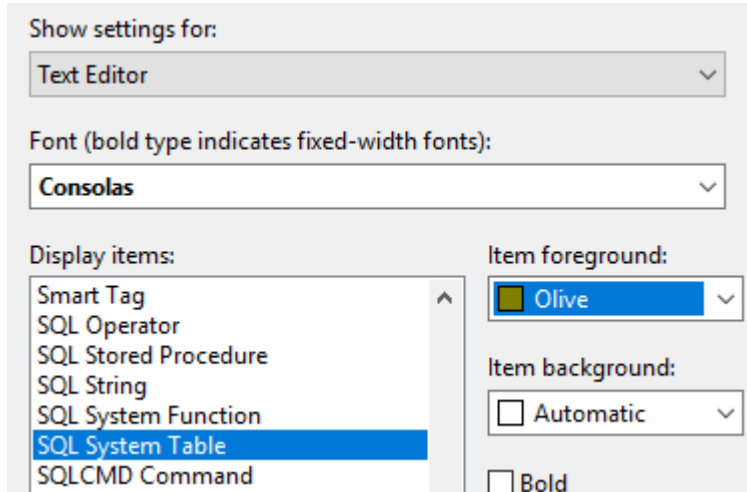
In **Tools**, then **Options**, then **Environment** then **Fonts and Colors**, select **Text Editor**, then look in the list of **Display items**:





Note that when **Text Editor** is selected, several SQL options appear in the **Display items** list. They are not there when you select other settings such as **Printer**.

I could then change the nasty SQL System Table color option to something easier to work with:



I noticed that **SQL System Table** was a standard **Green** so I've chosen **Olive** here, and then on my screen, they look much better:

```
SELECT @@VERSION;  
SELECT * FROM sys.tables ;
```

If they aren't dark enough, I could also **Bold** them.

It's worth noting that this can help for people with different visual challenges or color blindness in general. Often, reds and greens are a specific problem.

While there is a separate set of colors for Printer and Cut/Copy, it unfortunately doesn't include the list of SQL language elements. (That does seem odd as it has other language elements).

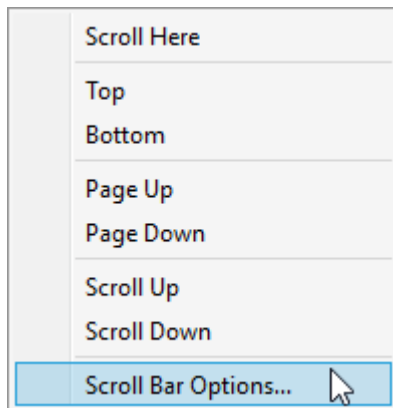
## 2.6 Cleaning up the Scroll Bar

It's great that SQL Server Management Studio has moved into the latest Visual Studio shell. Unfortunately, there are one or two things that are a little harder at first for people who want to use SSMS to write T-SQL. One that was driving me crazy was the scroll bar.

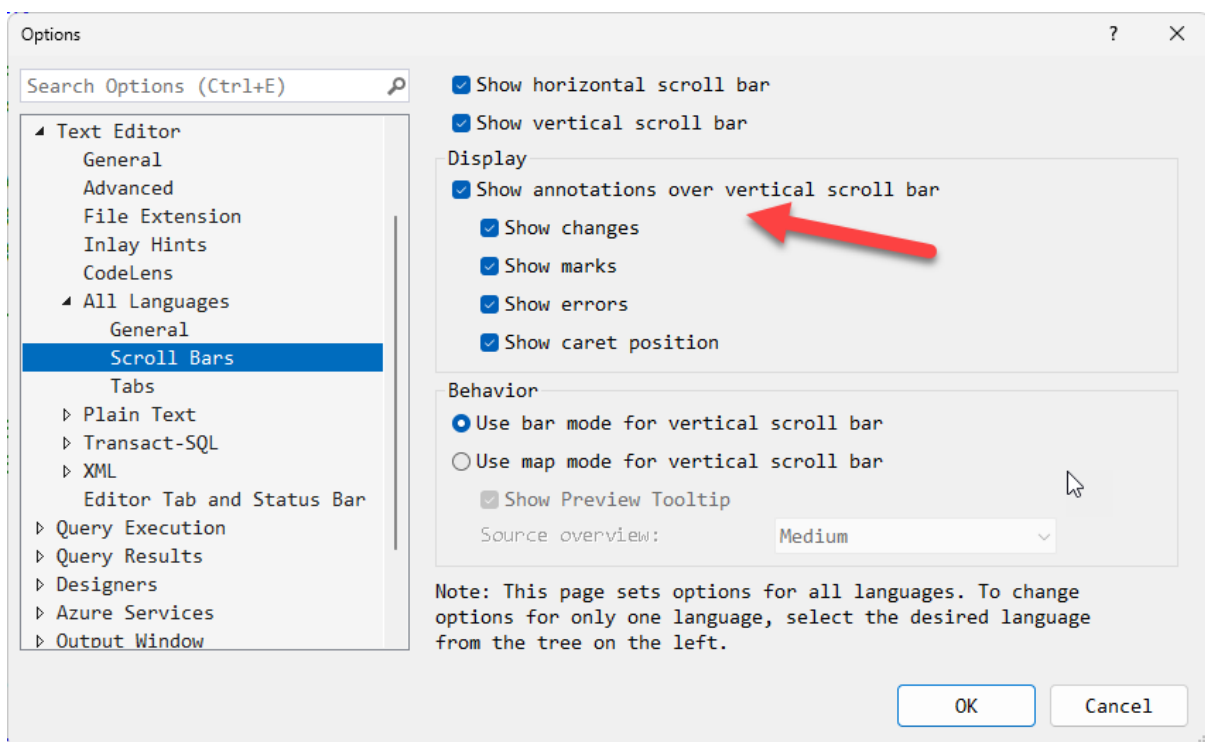
Visual Studio tries to give so much information on that bar, about what's changed, where the insertion carat is, etc. The problem with this is that I often now can't even find the handle when I want to scroll the window. For example, how do you grab the handle with your mouse and slide the window when it looks like this?



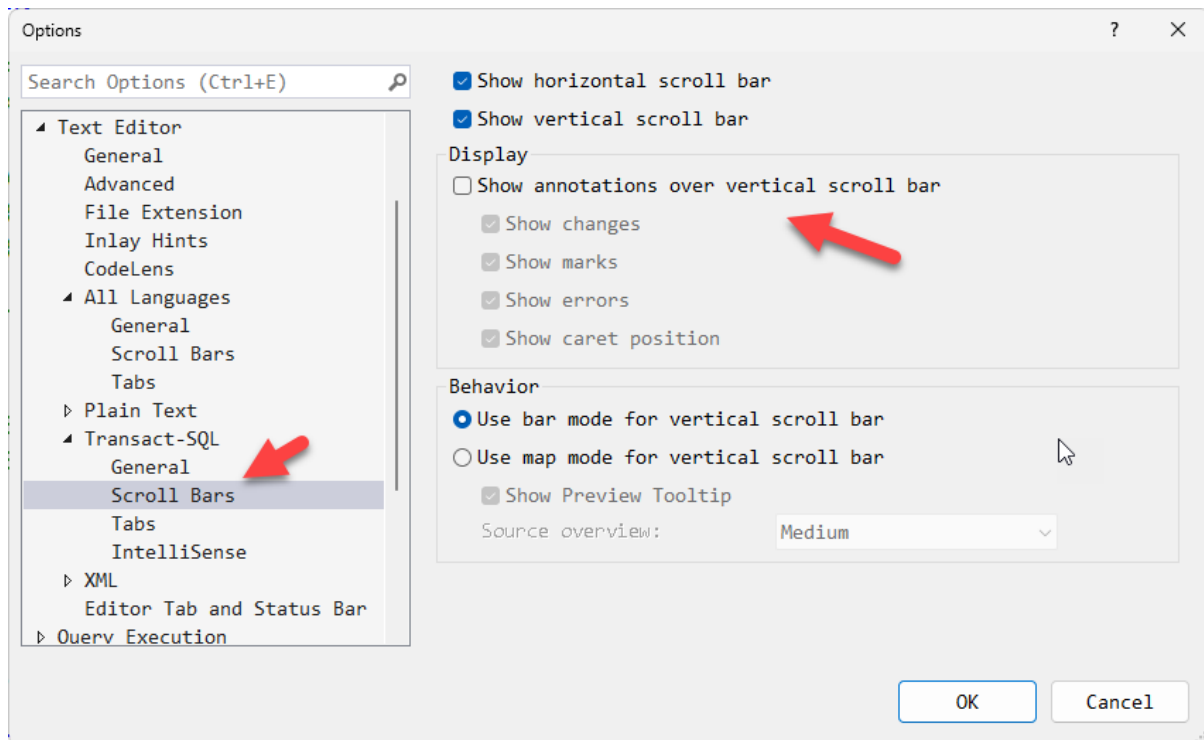
I was starting to get resigned to this when I asked in the MVP email list. Erik Jenson pointed out that the scroll bar itself had properties. I should have thought of that. If you right-click the scroll bar, you get these options:



Choosing “Scroll Bar Options” then leads to this:



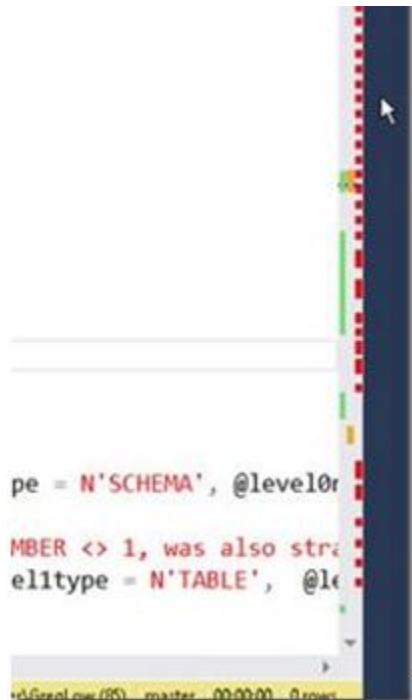
For me, the ones that I've highlighted are the real offenders. However, note the warning at the bottom. You really don't want to remove these for all languages. Some might be helpful to you if you use other languages. So, if you do decide to change them, click on the option further down the list, to set them for T-SQL only:



I hope that helps you make SQL Server Management Studio editing a little easier.

## 2.7 Making sense of the colors in the SSMS scroll bar

In an earlier post, I described how I didn't particularly like all the colors that are shown in the scroll bar now in SQL Server Management Studio (SSMS):



In that post, I described how to turn them all off, or at least how to kill off some of them. But, of course they are there for a reason.

Instead of turning them all off, you might decide to make sense of what they are there for.

The colors that are displayed indicate the following:

- **Red** - this is showing where syntax errors appear in your code
- **Blue** - this shows where the cursor currently is. That's helpful when you have scrolled but haven't moved the cursor. However, given this is the most useful one for me, I have to say that when all the other colors are present, it's the one that I find hard to locate.
- **Yellow** - this is indicating changes that you have made but have not yet saved.
- **Green** - this is showing saved changes.
- **Maroon** - this shows the location of "marks" - for us this means breakpoints.
- **Black** - this shows bookmarks.

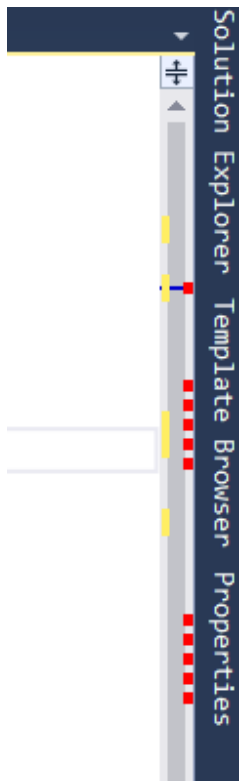
And remember that you can change which ones are displayed by right clicking the scroll bar and changing its settings.

## 2.8 Scroll bar map mode

In another section, I've described ways to configure the scroll bar in SQL Server Management Studio (SSMS).

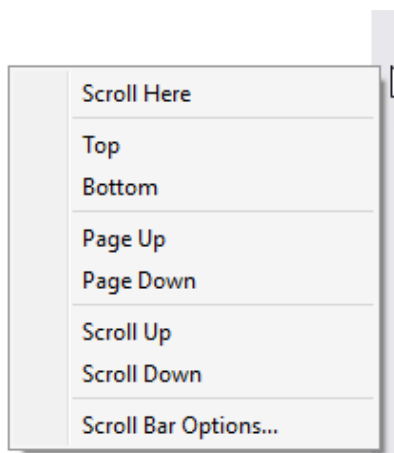
There is another key option that I haven't discussed previously: map mode.

By default, the scroll bar shows the changes, syntax errors, etc:



If you have a long script though, it can be hard to visualize what's in the other parts of the script. Map mode can help with this.

Right-click the scroll bar and choose Scroll Bar Options:



Below the display options that we have previously looked at is another set of options:

**Display**

- ☒ Show annotations over vertical scroll bar
  - ☒ Show changes
  - ☒ Show marks
  - ☒ Show errors
  - ☒ Show caret position

**Behavior**

- ☐ Use bar mode for vertical scroll bar
- ☒ Use map mode for vertical scroll bar
- ☒ Show Preview Tooltip
- Source overview: Medium ▼
  - Off
  - Narrow
  - Medium
  - Wide

Note: This page sets options for only one language from the tree on the left.

ange  
guage

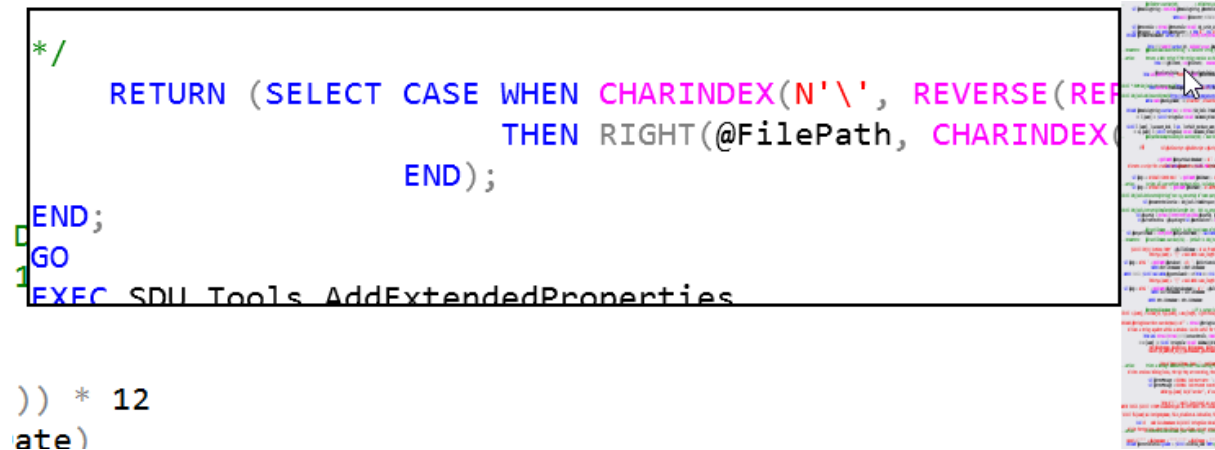
When you choose to use map mode instead of bar mode (the default), the appearance of the scroll bar changes:



You are presented with a tiny preview of the script beside the colored indicator. Obviously you can't read that but it can help to visualize the position of the code you are working on within the script.



If you hover over any code section though, it then shows you that code in a preview window:



```
)) * 12  
ate)
```

## 2.9 Adding multi-level undo, redo

Years ago, I had the privilege of presenting "what's new in SQL Server" sessions at many locations around the world. When presenting sessions, you can sometimes learn as much as you teach. What I learned while delivering those sessions is that the product group sees what's important in a release very differently to what the attendees do.

Each time there's a new version of SQL Server, there will normally be three or four marketing pillars (groups of functionality), and each will have about eight to ten bullet points.

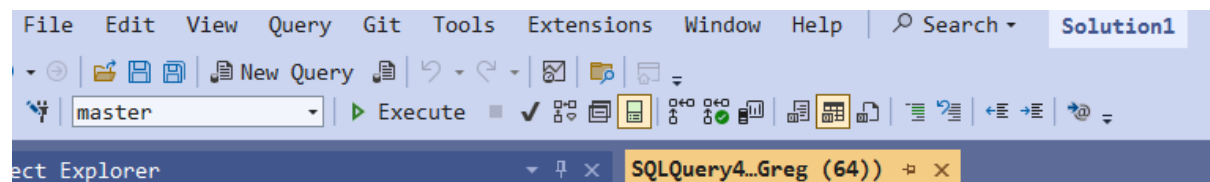
So, when SQL Server 2012 was released, what got the biggest reaction from the audiences? Was it the new availability groups? Was it the new tabular data model in SSAS? Was it the new project model in SSIS?

**It might (or might not) surprise you to hear that the biggest reaction world-wide to the release came when I showed the audience that the undo/redo options in the SSIS designer now actually worked.**

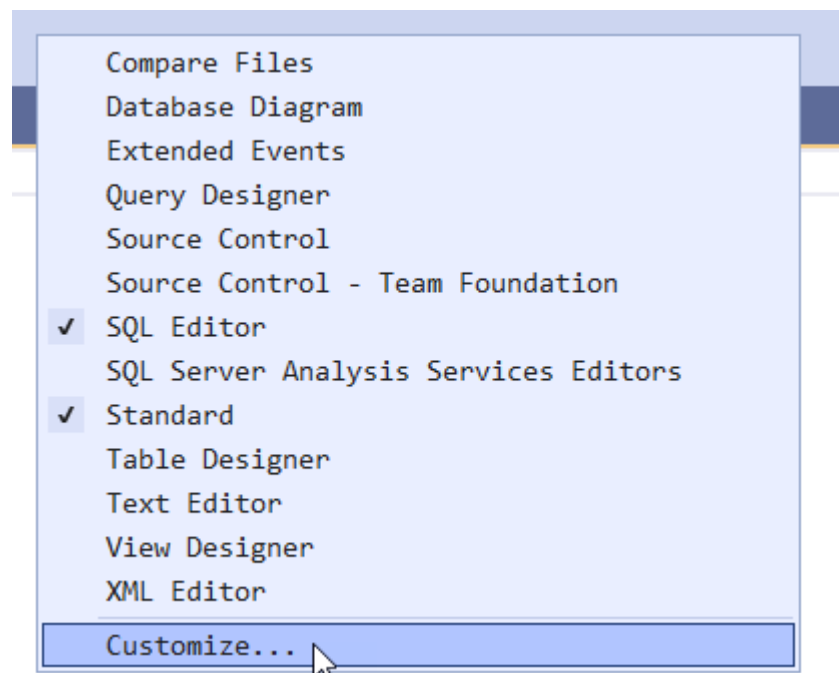
Even better, they were multi-level undo and redo.

SQL Server Management Studio can also use the same option. By default, it only has a single-level undo and redo. It can go back quite a distance in undo but it does so one step at a time. Adding a multi-level undo and redo can really improve your editing experience. I have no idea why it's not there on the toolbar by default. So let's fix that !

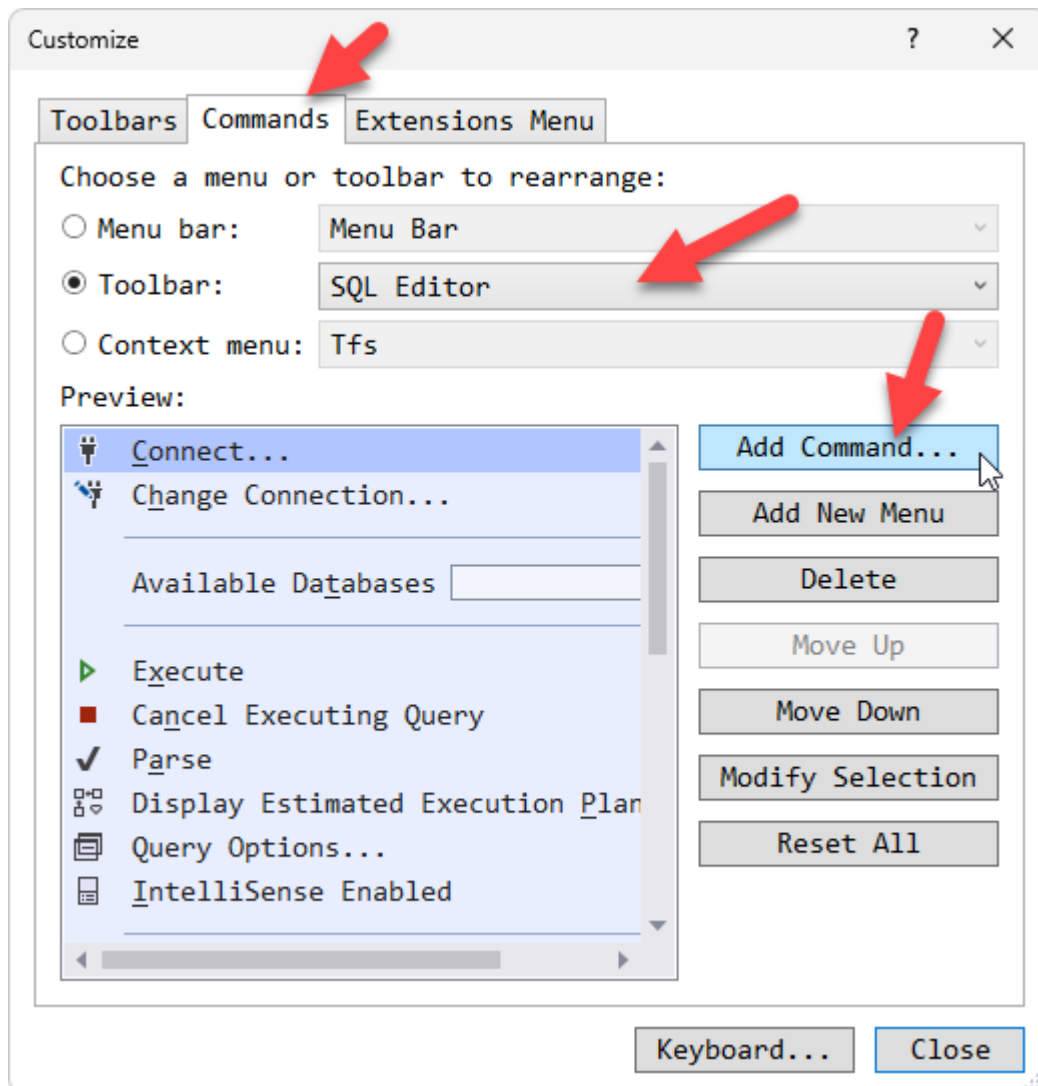
First let's see the toolbar as it started.



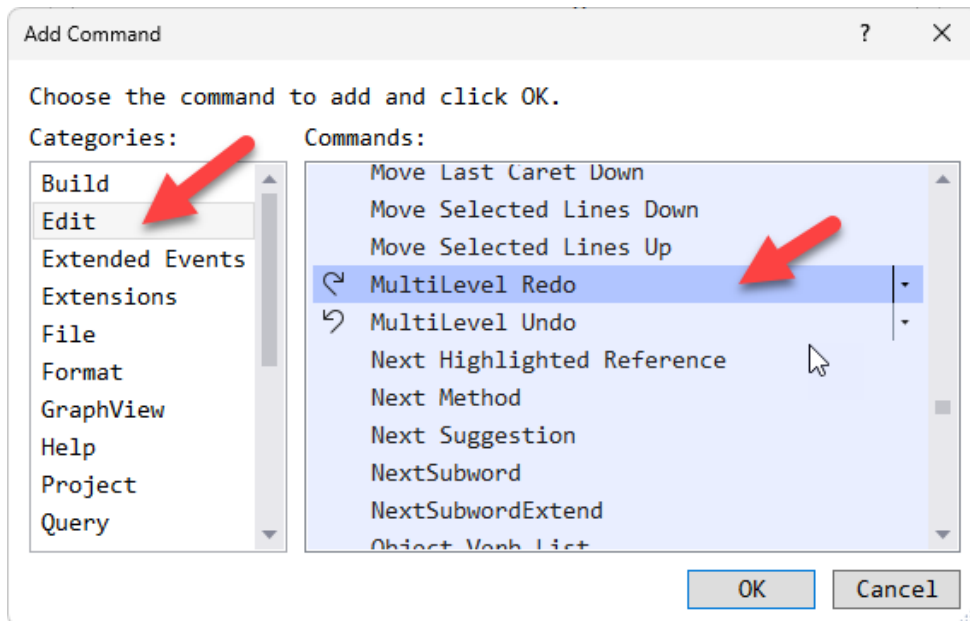
Right-click a blank area in the tool bar then choose Customize.



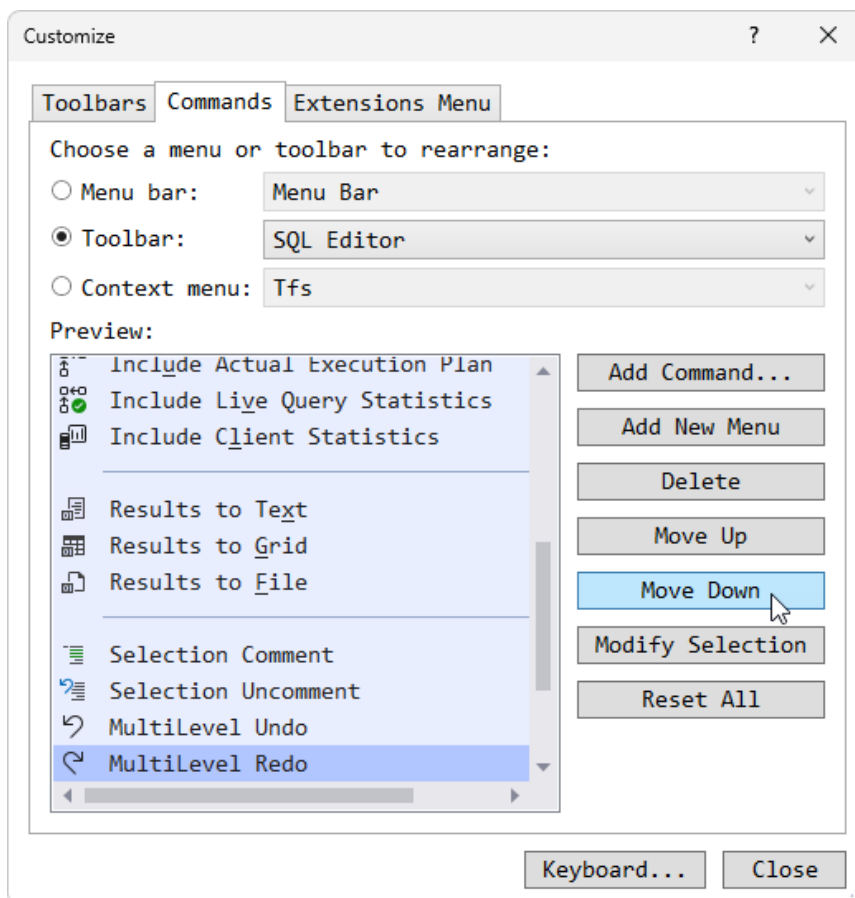
Then choose the Commands tab, the SQL Editor for the Toolbar, and click Add Command:



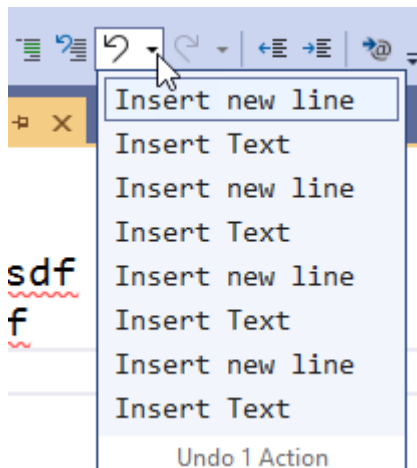
From the Categories, choose Edit. When the commands appear, scroll down to the MultiLevel Redo and click OK.



Do the same to add MultiLevel Undo, then move them down to where you want them using the Move Down button. I like to have them just after the comment / uncomment buttons.



Once they are in place, note that unlike the normal undo (or Control Z), you can click the button, see a list of your previous actions, and choose how many to apply.

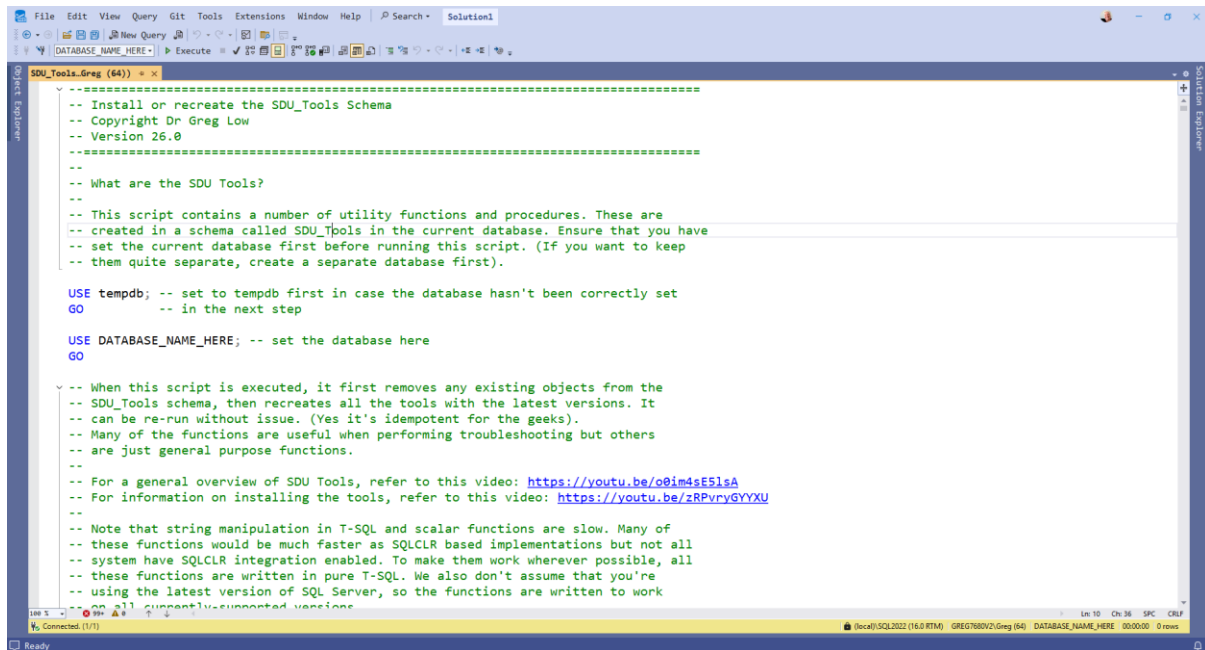


Once you undo some, the same applies to Redo.

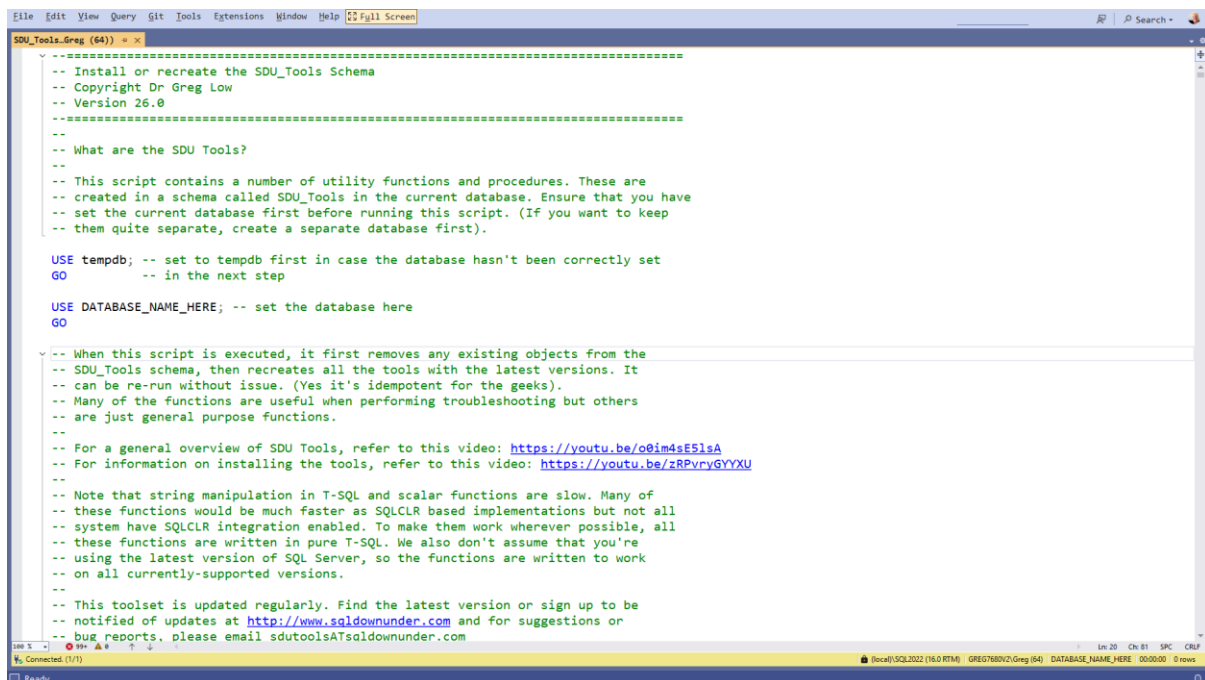
## 2.10 Toggle full screen (Alt shift enter)

SSMS is a great tool and it has lots of helpful menu items and toolbar items. Unfortunately, all these items take up screen real estate.

Even though you can get more screen real estate for editing, by unpinning the windows on each side, you can see that the default screen layout could be considered a bit cluttered if you really just want to focus on the particular query that you're working on.



A keyboard shortcut can help here. **Alt-Shift-Enter** toggles full screen mode in SSMS. Note how it gives you much more screen real estate to work with:



And the same shortcut toggles it back.

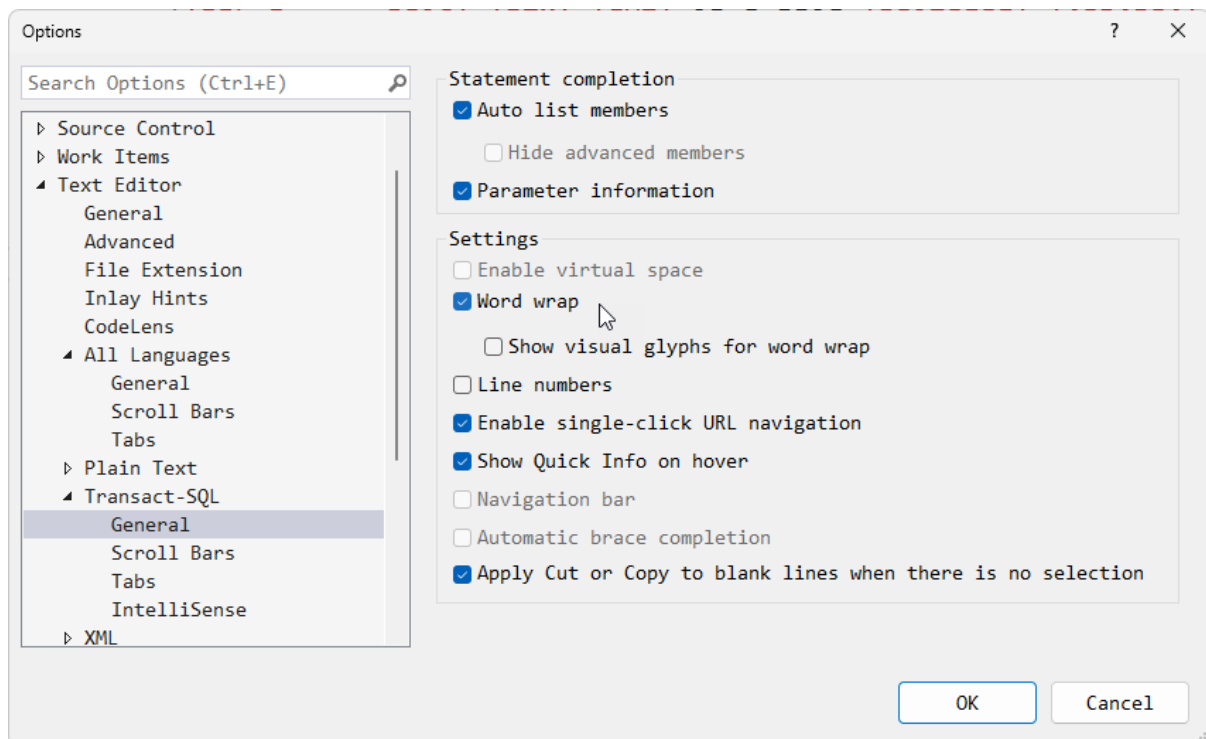
## 2.11 Use visual glyphs for word wrap

Code quality has always been an important topic ever since coding began. Code complexity is an important part of this. One of the topics that came up many years ago was a discussion on what length procedures or functions should be, before they became too difficult to follow.

**I remember one guy commenting that he thought as soon as all the code didn't fit on your screen any more, you were much more likely to have bugs in it.** At the time, screens weren't all that big.

Even today when writing T-SQL, if you've ever worked on a 4000-line stored procedure, you'll know how hard that is. But if you must work on a long script, it makes it even harder if it's also a wide script. Then you end up scrolling around in both directions. For this reason, most formatting tools like SQL Prompt have an option to limit the width of lines.

Breaking code into separate lines however, isn't always possible. And if you've decided that you really don't want to do that, you can enable word wrap in SQL Server Management Studio (SSMS).



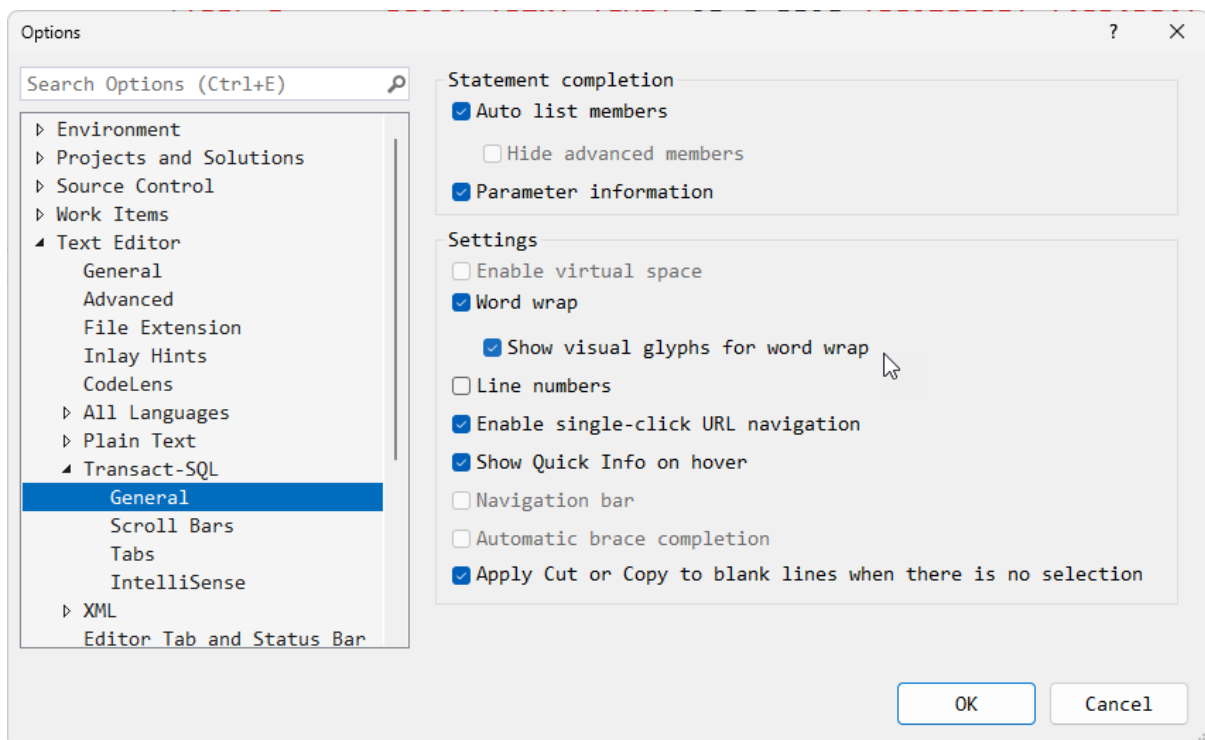
Once you do that, you no longer have long lines.

```
( 'SQL Server 2016', 'SP1', 'CU8', 13, 0, 4474, '20180319', '4077064', NULL ),  
( 'SQL Server 2016', 'SP1', 'CU9', 13, 0, 4502, '20180530', '4100997', NULL ),  
( 'SQL Server 2016', 'SP1', 'CU10', 13, 0, 4514, '20180716', '4341569', NULL ),  
( 'SQL Server  
2016', 'SP1', 'CU11', 13, 0, 4528, '20180918', '4459676', '4338204,4133191,4457953,4458880,4131251,4459535,  
4463125,4463328' ),  
( 'SQL Server  
2016', 'SP1', 'CU12', 13, 0, 4541, '20181014', '4464343', '4019799,4294694,4459220,4459981,4460116,4465443,  
4465745,4465747,4465867,4468103,4469349,4469554,4469600,4469815,4470057,4470546' ),  
( 'SQL Server
```

One challenge with this is that it's hard to work out at a glance, which lines have been wrapped and which haven't.

The answer to this is to enable a visual glyph for the word wrap.


From the Tools menu, choose Options, then Text Editor, then Transact-SQL (or All Languages if you prefer), then General. Note the option for showing the glyphs:





Now when the word wrap occurs, you can see it quite clearly:

```
7, '20171121', '4037354', NULL),  
5, '20180104', '4057119', NULL),  
4, '20180319', '4077064', NULL),  
2, '20180530', '4100997', NULL),  
14, '20180716', '4341569', NULL),  
  
3, '4459676', '4338204,4133191,4457953,4458880,4131251,4459535',  
  
4, '4464343', '4019799,4294694,4459220,4459981,4460116,4465443',  
349,4469554,4469600,4469815,4470057,4470546'),  
  
4, '4475775', '4055674,4090032,4346803,4460112,4465745,4475322',  
  
9, '4488535', '4463320,4488809,4490138,4490435,4491696,4493329',  
  
7, '4495257', '4338636,4489150,4492604,4497225,4497230,4497701',
```



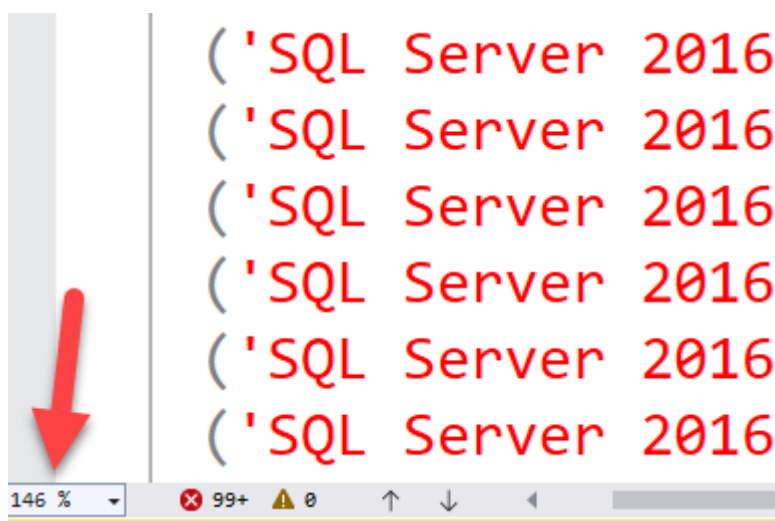
## 2.12 Using zoom features

When working with SQL Server Management Studio, sometimes you need to show someone else what you're working on, and the fonts that you're using are just too small for someone looking over your shoulder or looking at a screen that you've shared with them.

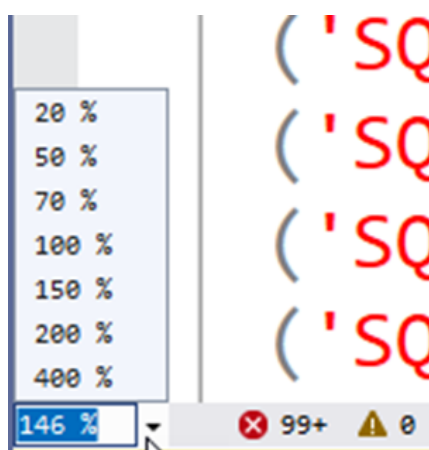
What I often see someone do then, is to go into Tools and Options and start to change the font and color settings. The pain with this is that you then need to set them back later.

There are several options to allow you to zoom in, without needing to change the settings.

First, like many Windows programs, you can hold down the Control key and use the mouse scroll bar to increase or decrease the font size. Note that as you do that, the percentage of zoom is also displayed in the lower left-hand side of the query window.



You could also, of course, just use that drop-down to set the zoom level too:

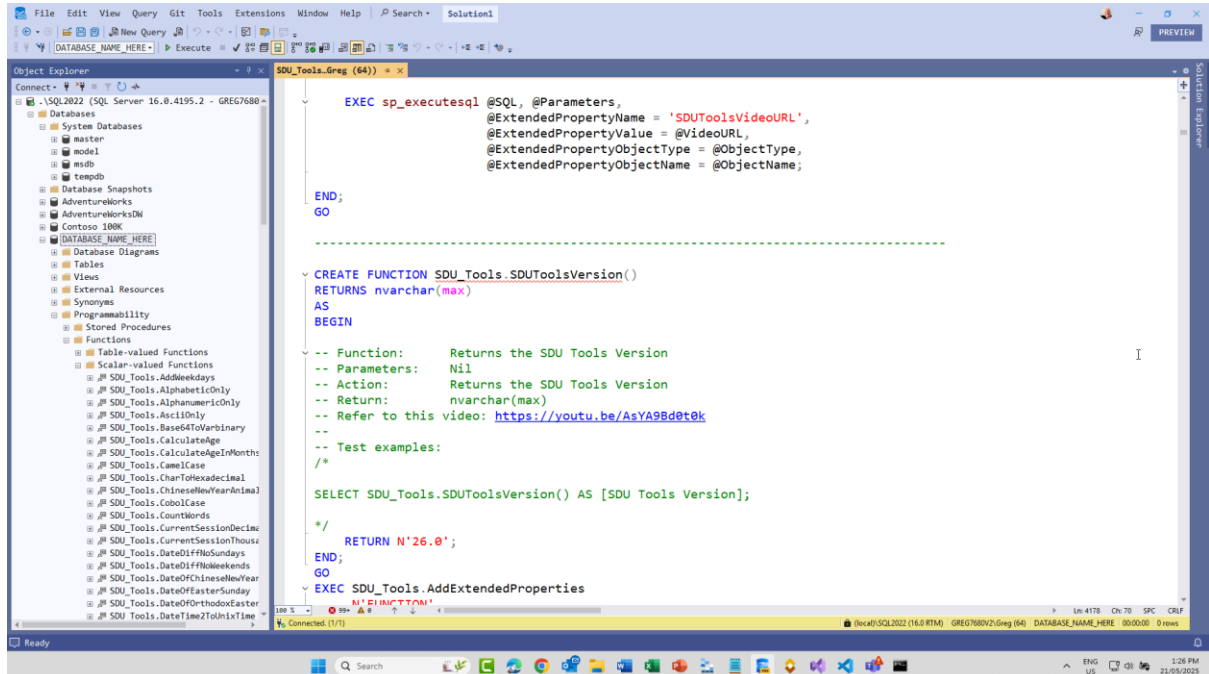


When you make this change, it applies to all current query windows, and all other query windows that you open until you change it back.

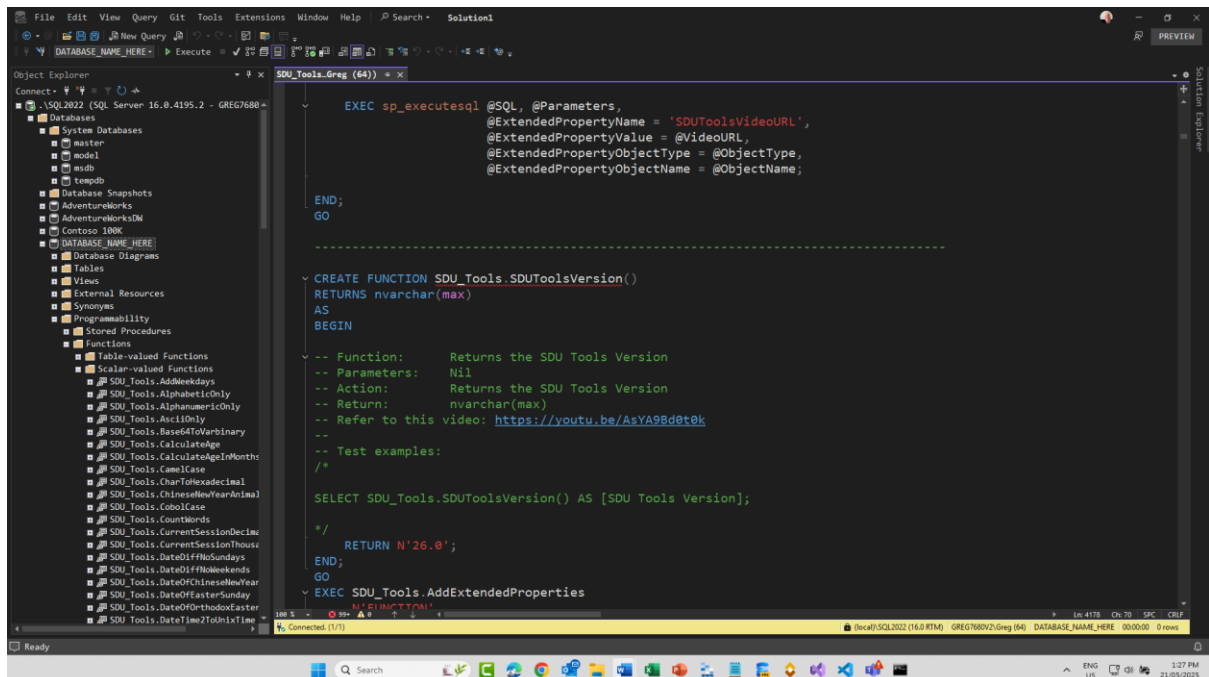
## 2.13 Using environment color themes

One feature that many developers have been asking the SSMS team for, is the ability to use a dark mode.

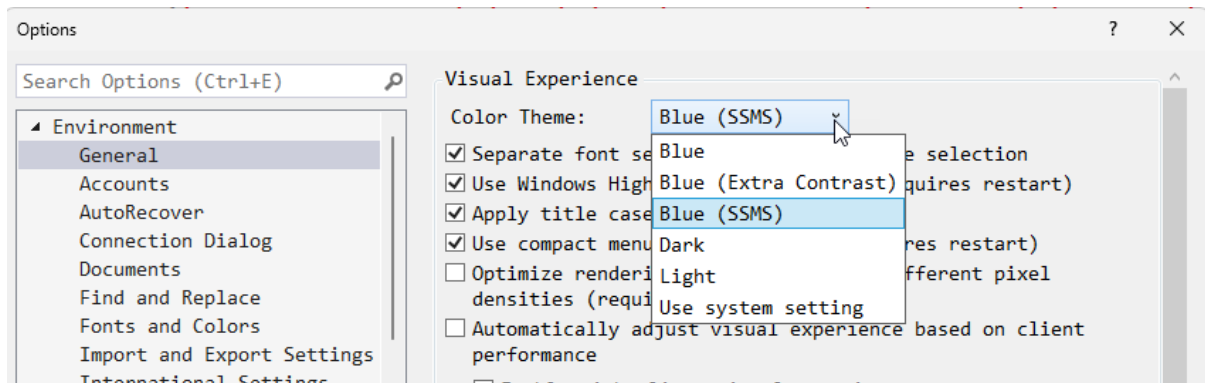
Instead of a screen that looks like this:



They wanted a screen that looks like this:



This capability is now present in SSMS. In **Tools** then **Options** then **Environment** then **General**, you can now choose a Color Theme:

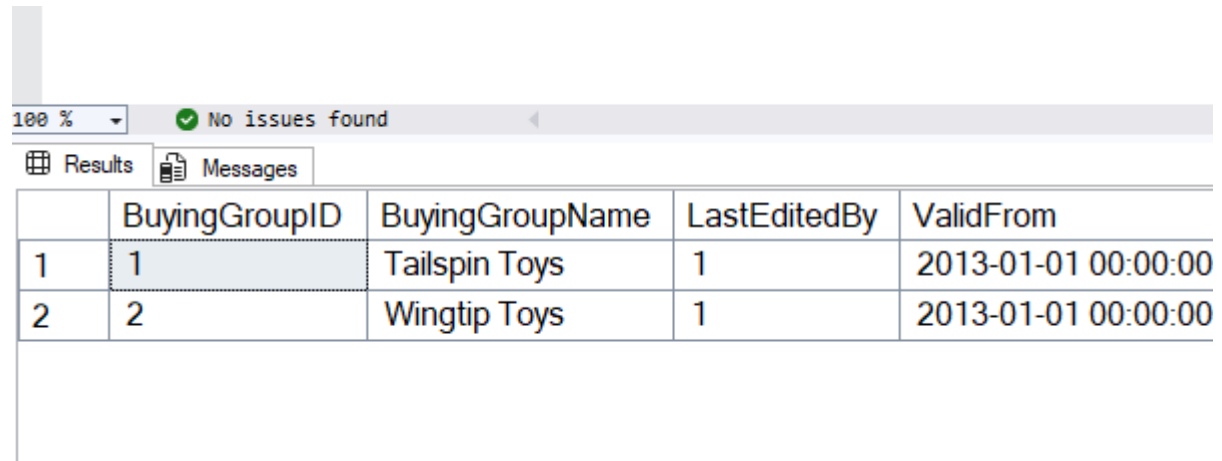


Note that there are several themes now available, including a new extra contrast option.

Also note that at present, I'm sure this is still a work in progress. The team might not have updated all UI dialogs. It's a big job.

## 2.14 Grid Results Border Colors

As soon as I started working with a recent version of SSMS, I realized that I didn't like how heavy the grid lines were in the output:

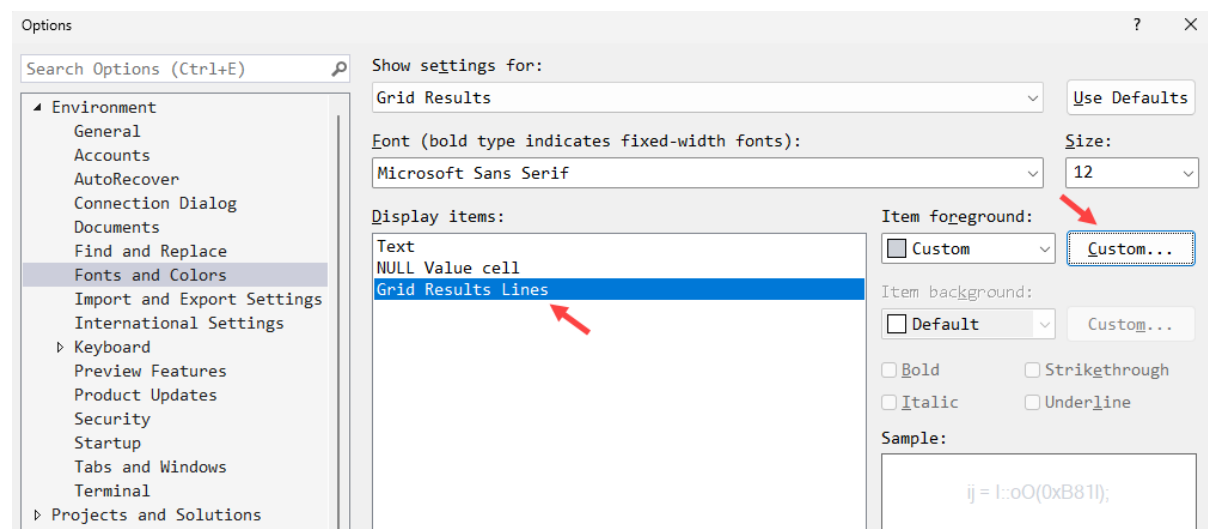


The screenshot shows a SQL Server Enterprise Manager window with a query results grid. The grid has thick border lines, making the data cells look like separate boxes. The status bar at the top indicates '100 %' zoom and 'No issues found'. The grid has two tabs: 'Results' and 'Messages'. The data is as follows:

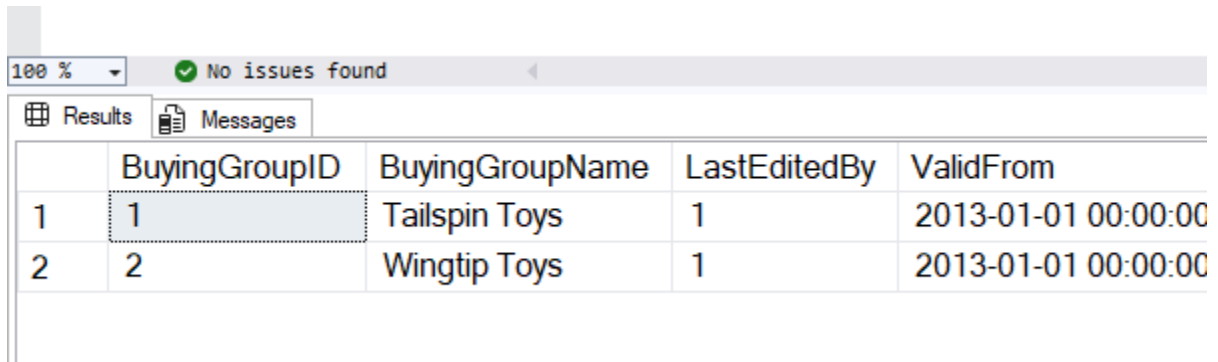
	BuyingGroupID	BuyingGroupName	LastEditedBy	ValidFrom
1	1	Tailspin Toys	1	2013-01-01 00:00:00
2	2	Wingtip Toys	1	2013-01-01 00:00:00

I really wanted the data to be the focus, not the lines. They seemed much darker than on the previous versions. Note that this is a personal preference. I can imagine some people preferring them as they now are.

Fortunately, though, you can now change the color of the lines. In Tools, Options, Fonts and Colors, in the Grid Output section, there is now an entry for Grid Results Lines.



I was able to set it using a custom color, and make them much lighter. I'm much happier with the outcome:



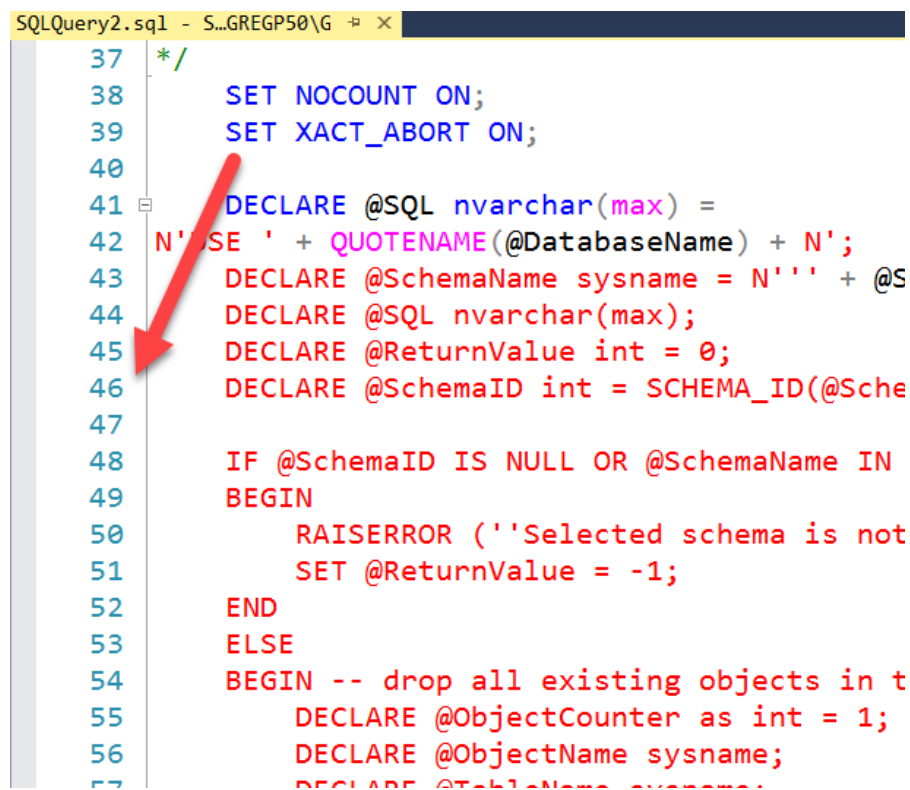
100 %    ✓ No issues found

Results    Messages

	BuyingGroupID	BuyingGroupName	LastEditedBy	ValidFrom
1	1	Tailspin Toys	1	2013-01-01 00:00:00
2	2	Wingtip Toys	1	2013-01-01 00:00:00



Once that's enabled, the query windows look like this:

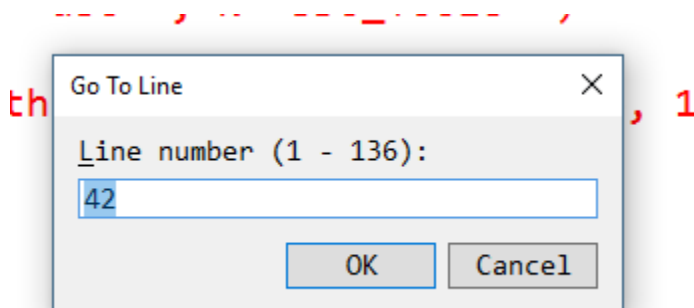


```
SQLQuery2.sql - S...GREGP50\G
37 */
38 SET NOCOUNT ON;
39 SET XACT_ABORT ON;
40
41 DECLARE @SQL nvarchar(max) =
42 N'USE ' + QUOTENAME(@DatabaseName) + N';
43 DECLARE @SchemaName sysname = N''' + @S
44 DECLARE @SQL nvarchar(max);
45 DECLARE @ReturnValue int = 0;
46 DECLARE @SchemaID int = SCHEMA_ID(@Sche
47
48 IF @SchemaID IS NULL OR @SchemaName IN
49 BEGIN
50     RAISERROR ('Selected schema is not
51     SET @ReturnValue = -1;
52 END
53 ELSE
54 BEGIN -- drop all existing objects in t
55     DECLARE @ObjectCounter as int = 1;
56     DECLARE @ObjectName sysname;
57     DECLARE @TableName sysname;
```

The line numbers go in a separate margin. Note that all lines are numbered, unrelated to whether they contain an individual SQL statement.

If you have a long script and you need to quickly reposition the cursor, note that you can also go directly to a line number. On the Edit menu, you can see that Ctrl-G is the shortcut for Go To.

Hitting Ctrl-G pops up a dialog and asks me for a line number:



Then takes me straight there. Note that in the dialog, the maximum line number is shown.



### 3.2 Useful keyboard shortcuts

Visual Studio is a very configurable tool, and particularly in the area of keyboard shortcuts. Because SQL Server Management Studio (SSMS) is based on Visual Studio, it inherits many of these configuration options.

SSMS has a very rich set of keyboard shortcuts. Without trying to cover most of them, I do want to highlight a few that I think are really important to know how to use.

Let's start with an easy set of commands.

You might need to change the case of some values. If I have this code:

```
DECLARE @A_Constant_Value int = 12;  
DECLARE @AnotherImportantValue int = 15;
```

I might decide that our new coding standards say that variables that are treated like constants, must be in all capitals and with words separated by underscores. (This is often called SHOUTY\_SNAKE\_CASE).

I can just highlight the name (or double-click it), then hit Ctrl-Shift-U to make it upper case.

```
DECLARE @A_CONSTANT_VALUE int = 12;  
DECLARE @AnotherImportantValue int = 15;
```

And Ctrl-Shift-L would make it lower case.

Now, what if I want to rearrange the order of these declaration lines. If I needed to move the @A\_CONSTANT\_VALUE line below the other line, I often see people highlight the whole line, then cut it, then paste it below.

A great keyboard shortcut for doing that, is to hit Alt-Shift-T. No matter where you are on the line, it just moves the line down by one.

```
DECLARE @AnotherImportantValue int = 15;  
DECLARE @A_CONSTANT_VALUE int = 12;
```

If I wanted to then add a line in between these two lines, what I typically see people do is to put the cursor on the D in the second line (at the beginning of the line), then hit Enter to move it down one, then use the up arrow to go back to the newly emptied line.

What you can do instead is put the cursor anywhere on the first line, and hit Ctrl-Shift-Enter.

```
DECLARE @AnotherImportantValue int = 15;  
  
DECLARE @A_CONSTANT_VALUE int = 12;
```

I'm sure that I'm somewhat anal, and because of this, I also often spend time cleaning up lines. So if I come across a line like the second one here, the spacing grates on me:

```
DECLARE @AnotherImportantValue int = 15;

DECLARE      @A_CONSTANT_VALUE                int =      12;
```

The quickest way to clean up the second line is to highlight the whole line, then hit Ctrl-K followed by Ctrl-Backslash.

```
DECLARE @AnotherImportantValue int = 15;

DECLARE @A_CONSTANT_VALUE int = 12;
```

Notice that it automatically removed all the messy whitespace. It can do multiple lines at once, but it won't rearrange what's on each line.

Finally, if I needed to comment out this code, I'd highlight it and hit Ctrl-K then Ctrl-C.

```
--DECLARE @AnotherImportantValue int = 15;

--DECLARE @A_CONSTANT_VALUE int = 12;
```

And Ctrl-K followed by Ctrl-U will uncomment it.

Two more easy shortcuts: **Ctrl-Home** takes you to the top of the script. **Ctrl-End** takes you to the bottom of the script.

The final most useful keyboard shortcut is **Ctrl-R**. It hides or shows the results pane.

You'll find a detailed list of keyboard shortcuts here:

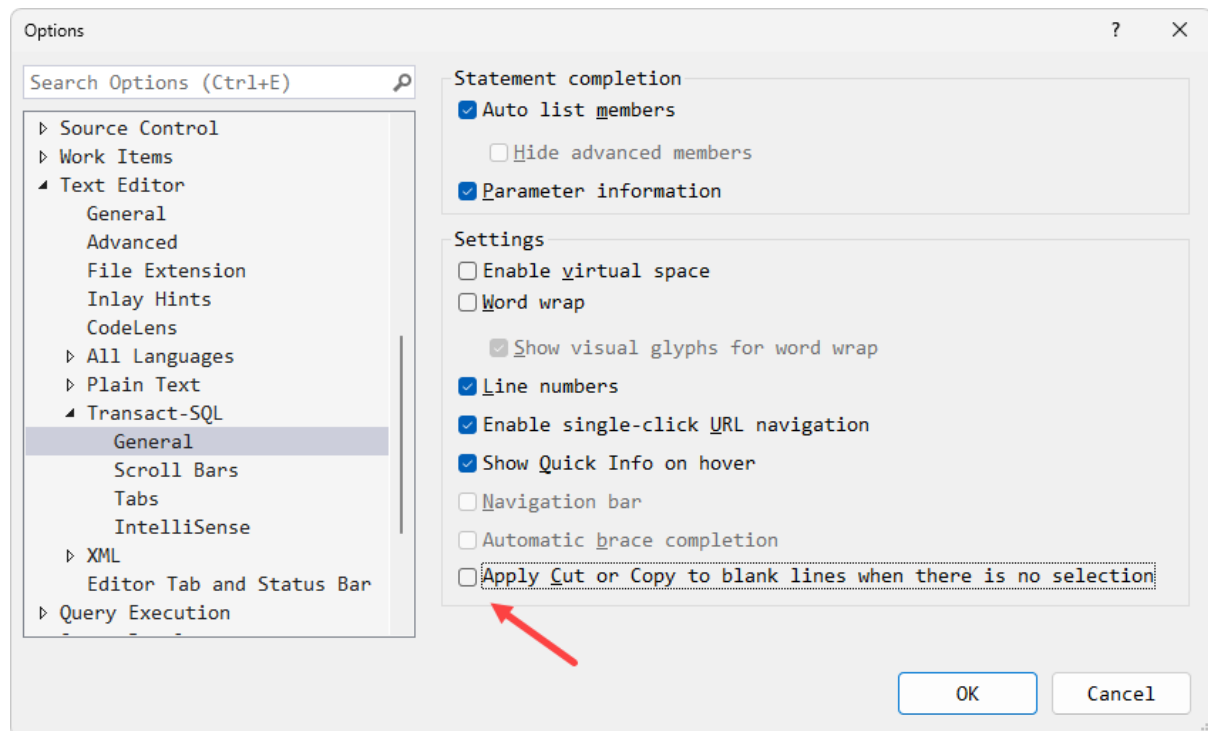
<https://learn.microsoft.com/en-us/ssms/sql-server-management-studio-keyboard-shortcuts>

### 3.3 Apply cut or copy commands to blank lines when there is no selection

When I'm doing a lot of query editing, I often get a bit mesmerized, particularly if there's a lot of manual copy and paste or cut and paste going on.

One thing that often drives me crazy is when I use Ctrl-C (ie: copy) when I meant to use Ctrl-V (ie: paste). Invariably, I do this when I have nothing highlighted at all. So not only did I not get the value pasted, I just copied an empty value into the clipboard.

But SQL Server Management Studio (SSMS) has your back on this. (And so does Visual Studio)



In Tools, then Options, then Text Editor, then Transact-SQL, under the General tab, there's an option for **Apply Cut of Copy commands to blank lines when there is no selection**.

The default is that it works as expected (like you don't want it to), but if you uncheck it, you might have just saved yourself some annoyance.

If I'm working with this code:

```
DECLARE @AnotherImportantValue int = 15;  
  
DECLARE @A_CONSTANT_VALUE int = 12;
```

and I highlight the word DECLARE and hit Ctrl-C, then move to the blank line and hit Ctrl-C instead of Ctrl-V, I'd have just lost my first clipboard item. With this option unchecked, I can just smile, and hit Ctrl-V again, and it will still paste:

```
DECLARE @AnotherImportantValue int = 15;  
DECLARE  
DECLARE @A_CONSTANT_VALUE int = 12;
```

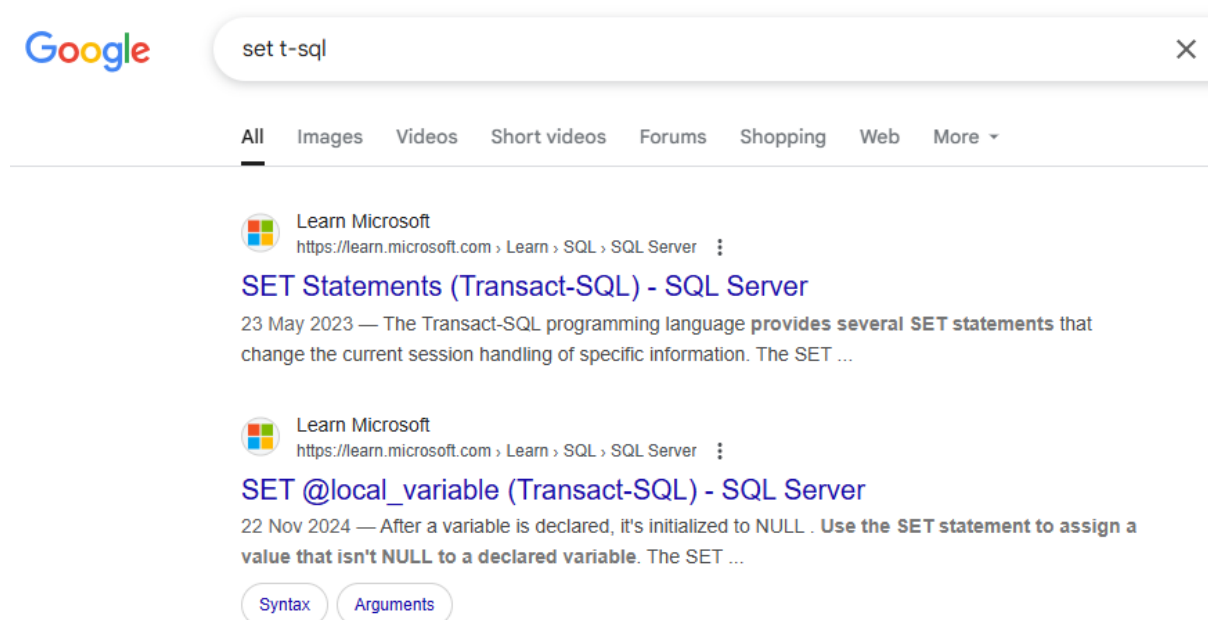
### 3.4 The magical F1 key – help on syntax and metadata

Years ago, I used to always recommend that people install Books Online (BOL) on their systems. It's ironic that it was called "Online", given we're really talking about "Offline", but back when we first were talking about it, we were comparing it to a physical book, not to a live reference on a computer screen.

Nowadays though, I find that the version online is so far superior to the one that you can install locally, that I think it's better to just use the online version. I particularly like the way that the online books are now cross-version ie: each page covers all supported versions, instead of having a separate page for each version.

Given there is a lot of information online, what's the quickest way then to find the page that you're after?

For T-SQL commands, I find the best option is to type the command name followed by t-sql into the search bar. For example, here's a search for the SET command:



Invariably, the command that you're after will be either the first or second entry. Note that with the current version of Google search, you get better results with "t-sql" than with "tsql".

But there's an even quicker way when you're editing in SQL Server Management Studio (SSMS). In this script that I'm editing:

```
SET @ObjectCounter += 1;
END;
END;';
EXECUTE (@SQL);
END;
```

If I double-click EXECUTE to highlight it, then hit F1, I'm taken directly to the correct page.

Version

SQL Server 2025 Preview

Filter by title

EXECUTE

PRINT

RAISERROR

CHECKPOINT

KILL

KILL QUERY NOTIFICATION SUBSCRIPTION

Learn / SQL / SQL Server /

## EXECUTE (Transact-SQL)

12/24/2024

Applies to: SQL Server Azure SQL Database Azure SQL Managed Analytics Platform System (PDW) SQL analytics endpoint in Microsoft Fabric

Executes a command string or character string within a Transact-SQL batch, or stored procedure, user-defined stored procedure, CLR stored procedure, scalar

That's awesome but the F1 key can do more. If I was looking at or editing this script from WideWorldImporters:

```
BEGIN TRAN;

INSERT [Application].People
(PersonID, FullName, PreferredName, IsPermittedToLogon, Logon
IsExternalLogonProvider, HashedPassword, IsSystemUser, IsEmp
IsSalesperson, UserPreferences, PhoneNumber, FaxNumber,
EmailAddress, LastEditedBy, ValidFrom, ValidTo)
VALUES
(@PrimaryContactPersonID, @PrimaryContactFullName, @PrimaryCo
0, NULL, 0, 0,
0, NULL, N'(' + CAST(@AreaCode AS nvarchar(20)) + N') 555-01
```

and I'm wondering about the People table, I can highlight the full table name **[Application].People**, then hit **Alt-F1**. I'm then returned all sorts of information about the table:

1	INSERT [Application].People									
100 % 1 0 ↑ ↓										
Results Messages										
	Name	Owner	Type	Created_datetime						
1	People	dbo	user table	2016-06-02 10:04:03.167						
	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullIns	
1	PersonID	int	no	4	10	0	no	(n/a)	(n/a)	
2	FullName	nvarchar	no	100			no	(n/a)	(n/a)	
3	PreferredName	nvarchar	no	100			no	(n/a)	(n/a)	
4	SearchName	nvarchar	yes	202			no	(n/a)	(n/a)	
5	IsPermittedToLogon	bit	no	1			no	(n/a)	(n/a)	
6	LogonName	nvarchar	no	100			yes	(n/a)	(n/a)	
7	IsExternalLogonPr...	bit	no	1			no	(n/a)	(n/a)	
8	HashedPassword	varbin...	no	-1			yes	no	yes	
	Identity		Seed	Increment	Not For Replication					
1	No identity column defined.		NULL	NULL	NULL					
	RowGuidCol									
1	No rowguidcol column defined.									
	Data_located_on_filegroup									
1	USERDATA									
	index_name		index_description				index_keys			
1	IX_Application_People_FullName		nonclustered located on USERDATA				FullName			
2	IX_Application_People_IsEmployee		nonclustered located on USERDATA				IsEmployee			
3	IX_Application_People_IsSalespe...		nonclustered located on USERDATA				IsSalespe...			
4	IX_Application_People_Perf_201...		nonclustered located on USERDATA				IsPermitte...			
5	PK_Application_People		clustered, unique, primary key locat...				PersonID			
	constraint_type		constraint_name				delete_action	update_action	status	
1	DEFAULT on column PersonID		DF_Application_People_PersonID				(n/a)	(n/a)	(n/a)	
2	FOREIGN KEY		FK_Application_People_Application_People				No Action	No Action	Enab	
3										
4	PRIMARY KEY (clustered)		PK_Application_People				(n/a)	(n/a)	(n/a)	
Table is referenced by foreign key										

What SSMS is doing is running the **sp\_help** command for that object. In another post, we'll talk about how you could change that if needed or change what happens with other function keys.

### 3.5 Fixing or improving the online documentation

I mentioned in an earlier post that I think the online documentation is now superior to any version that you can install locally.

I particularly like the way that the online version is cross-version ie: each page covers all supported versions, instead of having a separate page for each version.

But one of the really big bonuses is that you also have the opportunity to change the documentation if you think it's incorrect or you think it could be improved. Microsoft have placed all the documentation in a GitHub repository and you can change it. Doing so is easier than you might expect.

Let's look at an example.

I've searched for the LEN function in T-SQL and found the page:

[Learn](#) / [SQL](#) / [SQL Server](#) /

[Ask Learn](#)

## LEN (Transact-SQL)

09/04/2024

**Applies to:** SQL Server Azure SQL Database Azure SQL Managed Instance Azure Synapse Analytics Analytics Platform System (PDW) SQL analytics endpoint in Microsoft Fabric Warehouse in Microsoft Fabric

Returns the number of characters of the specified string expression, excluding trailing spaces.

#### Note

To return the number of bytes used to represent an expression, use the [DATALENGTH](#) function.

[Transact-SQL syntax conventions](#)

## Syntax

syntasql

Copy

```
LEN ( string_expression )
```

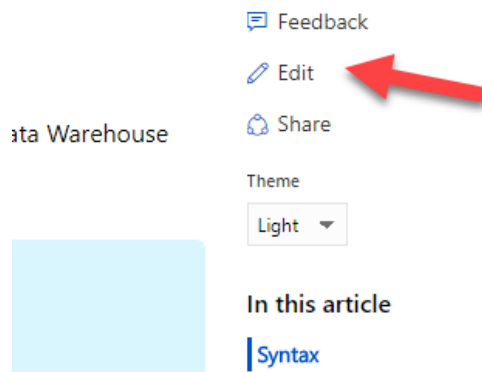
Now while I'm reading the page, I see this:

## Examples

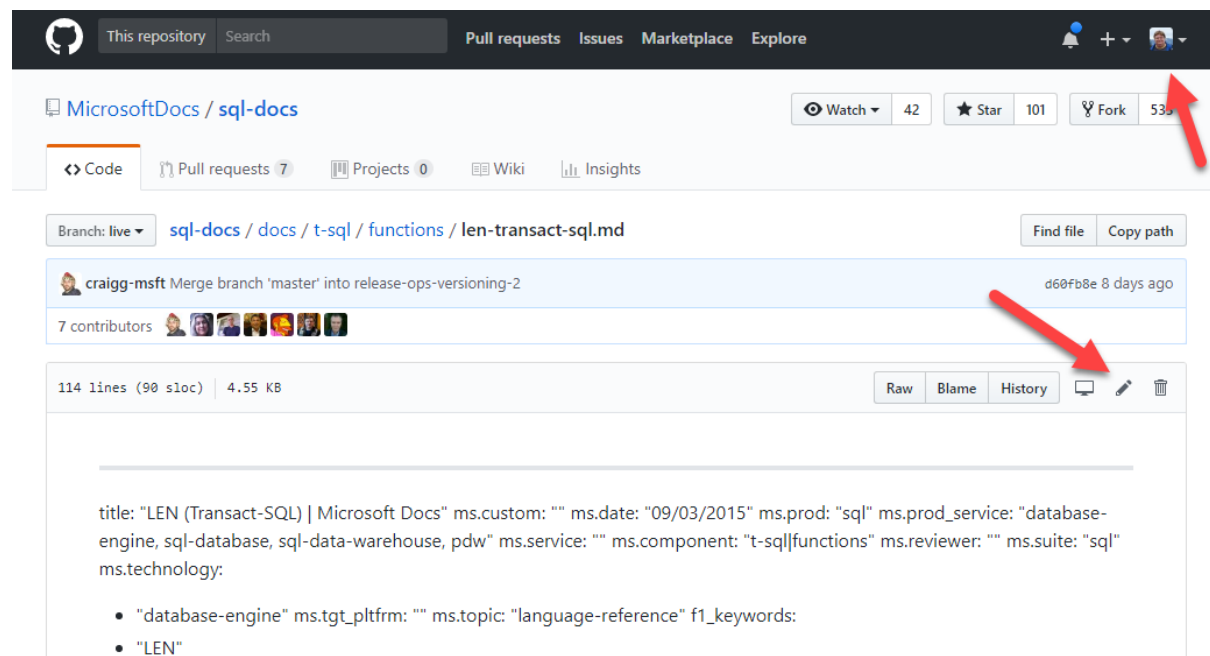
The following example selects the number of characters and the data in `FirstName` AdventureWorks2012 database.

```
SELECT LEN(FirstName) AS Length, FirstName, LastName
FROM Sales.vIndividualCustomer
WHERE CountryRegionName = 'Australia';
GO
```

I'm puzzled why that example is specific to AdventureWorks2012. That code would work on all AdventureWorks versions. So, let's try to change it. At the top of the page, there's an **Edit** link.



I'll click this and I'm taken to the page in Git:



Note in the top right-hand corner that I've already logged onto GitHub. Also notice the Edit pencil. Now I can't just directly change this info, so what I do is click this to "fork" the project so that I have my own copy to edit.



MicrosoftDocs / sql-docs

Watch 42 Star 101 Fork 533

Code Pull requests 7 Projects 0 Wiki Insights

You're editing a file in a project you don't have write access to. Submitting a change to this file will write it to a new branch in your fork greglow/sql-docs, so you can send a pull request.

sql-docs / docs / t-sql / functions / len-transact-sql.md or cancel

Edit file Preview changes Spaces 2 Soft wrap

```

1  ---
2  title: "LEN (Transact-SQL) | Microsoft Docs"
3  ms.custom: ""
4  ms.date: "09/03/2015"
5  ms.prod: "sql"
6  ms.prod_service: "database-engine, sql-database, sql-data-warehouse, pdw"

```

Now I can make the change that I want to:


```


70
71  ## Examples
72  The following example selects the number of characters and the data in `FirstName` for people located in the AdventureWorks database.
73
74  ```
75  SELECT LEN(FirstName), AS Length, FirstName, LastName
76  FROM Sales.vIndividualCustomer
77  WHERE CountryRegionName = 'Australia';
78  GO
79  ```
80

```

And at the bottom of the page, I explain why:

113  
114

 **Propose file change**

 **ProTip!** Great commit summaries contain fewer than 50 characters. Place extra information in the extended description.

Removed specific reference to 2012 version of AdventureWorks

This example runs on all versions of AdventureWorks. It's misleading to have just AdventureWorks2012 shown.

Propose file change Cancel

Then I click **Propose file change**, and I'm taken to a page that asks me to **create a pull request**.

[Code](#) [Pull requests 7](#) [Projects 0](#) [Wiki](#) [Insights](#)

## Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

base fork: MicrosoftDocs/sql-docs

base: live

head fork: greglow/sql-docs

compare: patch-49

✓ **Able to merge.** These branches can be automatically merged.

[Create pull request](#) Discuss and review the changes in this comparison with others.

1 commit

1 file changed

0 commit comments

Commits on Apr 21, 2018

greglow

Removed specific reference to 2012 version of AdventureWorks

Showing 1 changed file with 1 addition and 1 deletion.

docs/t-sql/functions/len-transact-sql.md

@@ -69,7 +69,7 @@	SELECT LEN(@v2) AS [nvarchar LEN], DATALENGTH(@v2) AS [nvarchar DATALENGTH];
69 69	...
70 70	
71 71	## Examples
72	- The following example selects the number of characters and the data in `FirstName` for people located uses the <code>!!INCLUDE[ssSampleDBnormal](../../includes/sssampledbnormal-md.md)</code> database.
72	+ The following example selects the number of characters and the data in `FirstName` for people located uses the <code>AdventureWorks</code> database.
73 73	
74 74	...
75 75	SELECT LEN(FirstName) AS Length, FirstName, LastName

GitHub runs some automated checks and makes sure that I'm not suggesting a change that can't easily be merged.

If I'm happy with the differences shown, I just click **Create pull request** and again another time. This page is then sent off to the person who is responsible for maintaining the page:

The screenshot shows a GitHub pull request interface for the repository `MicrosoftDocs/sql-docs`. The pull request title is "Removed specific reference to 2012 version of AdventureWorks #561". It was created by `greglow` and is targeting the `MicrosoftDocs:live` branch from the `greglow:patch-49` branch. The interface includes tabs for Code, Pull requests (8), Projects (0), Wiki, and Insights. Below the title, there are statistics for Conversation (0), Commits (1), and Files changed (1). A comment from `greglow` states: "This example runs on all versions of AdventureWorks. It's misleading to have just AdventureWorks2012 shown." Below the comment, a commit is listed: "Removed specific reference to 2012 version of AdventureWorks" with a "Verified" status and a green checkmark. A message indicates that all checks have passed and the branch has no conflicts with the base branch. At the bottom, there is a "Write" and "Preview" section for adding a description.

If they agree, your change will be merged in. Either way, you'll receive emails telling you what's going on.

If you have added a lot of changes or code, you'll also receive another email asking you to agree to be a contributor.

This is a great option and once you're used to it, it's very easy to do. And if you look at the published page today, you'll see this change was made!

### 3.6 Manually prompting for and Refreshing Intellisense

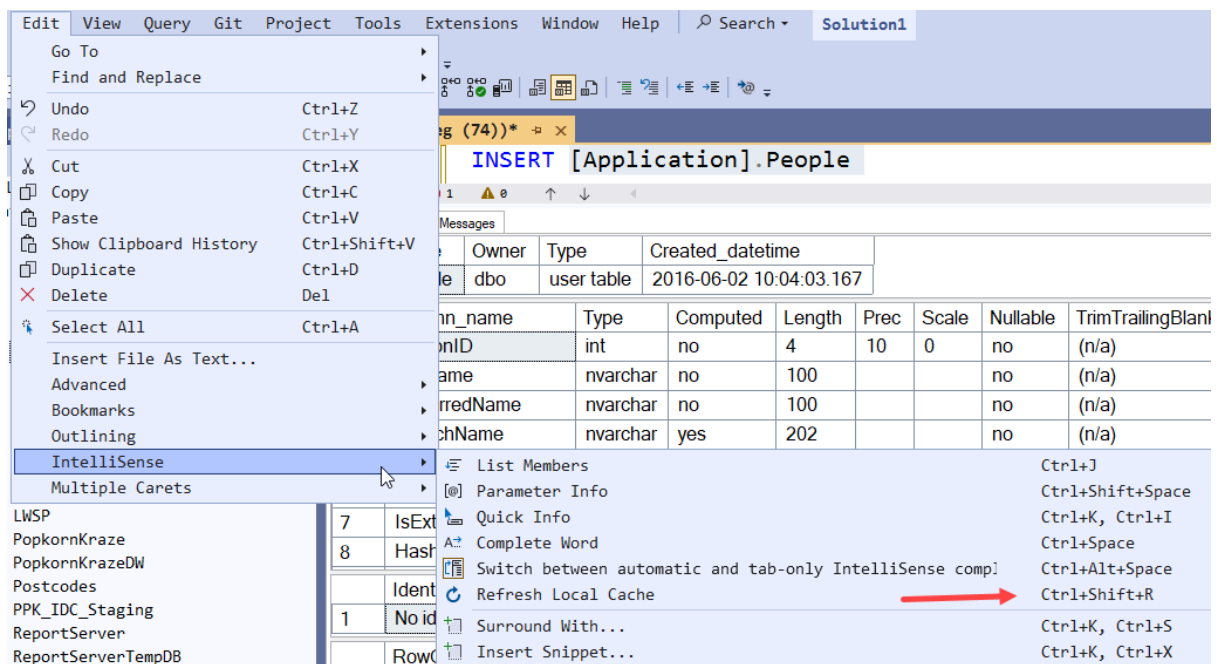
Intellisense is one of the best things that's ever been added to Visual Studio or to SQL Server Management Studio (SSMS). It's hard to remember back to before it was added, or how we worked then.

I had a young friend from the United Kingdom who had just completed a Computer Science degree and one of the things that he was most proud of, is that he knew so many HTML tags and which attributes went with which tags. When I showed him HTML Intellisense in Visual Studio, I think he was about to cry.

While Intellisense in SSMS pretty much works as expected, there are a few things that can go wrong to confuse it.

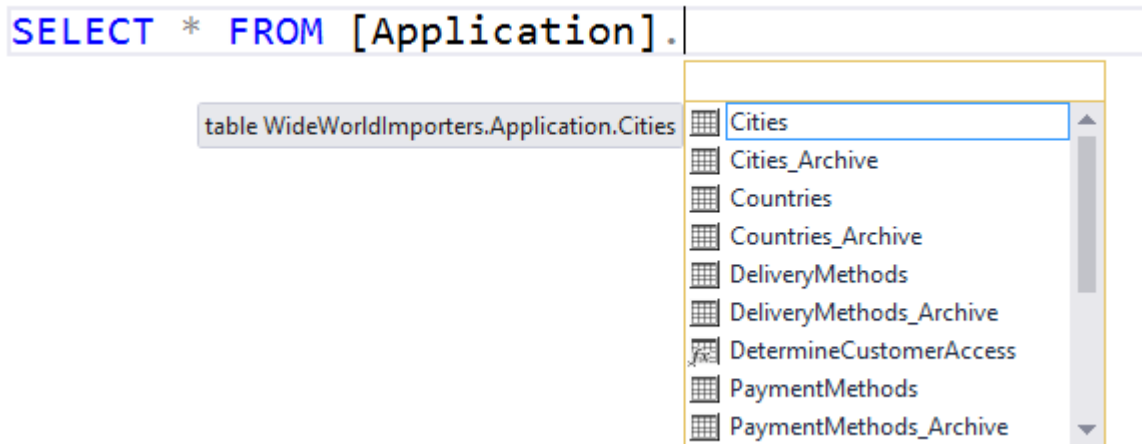
The first problem is that it caches the list of objects to make sure it can perform quite well. But the cache can get out of sync with reality. A common cause of that is if I execute T-SQL commands in one query window, and I'm using those same objects in another window.

I've seen people quite puzzled about this but it's easy to fix. On the Edit menu, you can see that the shortcut key to refresh the local cache is Ctrl-Shift-R:



So if you see a bunch of unexpected "red squiggles", the first thing to do is to hit **Ctrl-Shift-R**.

Another thing that can happen is that you get into a situation where the prompted values won't appear. If I type the following code:



note that the Intellisense has appeared. But if I hit the Escape key, it will disappear again. So users wonder how to get it back. Now one option is to backspace over the period, then type the period again. The standard option though, is to hit **Alt-RightArrow**.

An alternative to this is to hit **Ctrl-Space**, and that's easier to hit anyway.

### 3.7 Replace Tabs with Spaces and do Macro-like work using Regular Expressions

A request that I hear all the time, is “I don’t like tabs but **insert name of annoying colleague here** decided he likes to use them. How do I remove them?” Similar thing happens when installing SSMS and leaving the default options.

#### Tabs vs Spaces

Whether to use tabs or spaces leads to near religious level arguments amongst SQL developers. I see strong arguments on both sides. In the past, I’ve always ended up using spaces because I’ve run into issues with tabs in some of the tooling that I needed to use.

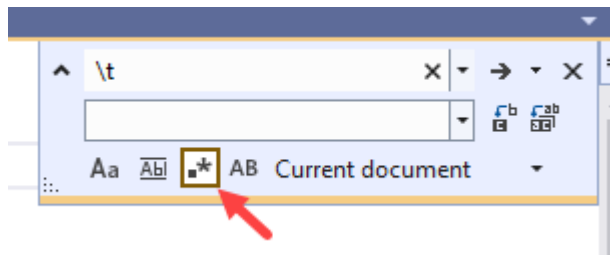
I’m quite swayed though, by the argument that says that tabs are best for accessibility. For someone with a visual impairment, being able to change the size of the tabs can be really helpful. It’s also useful for people switching between different size screens.

In this item, I’m not taking sides, but showing you how to change from tabs to spaces, if that’s what you want to do.

#### SSMS options

Now SSMS allows you to choose to use spaces instead of tabs (in Tools > Options) but that doesn’t “fix” tabs that are already in the code.

The regular expression functions in SSMS let you do this. Hit Ctrl-H to open the Find and Replace dialog (or use the Edit > Find and Replace > Quick Replace menus), then configure the window as follows:

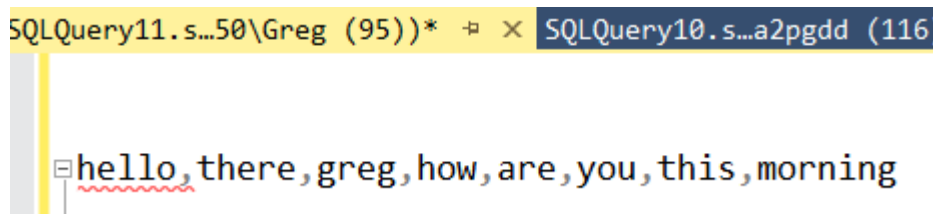


Note that I’ve clicked the little square and asterisk thing (which says to use regular expressions), then put the change from text to \t which is a tab, and set the change to text to four spaces (or however many you want), then I click the Replace All button and magically all my tabs are replaced.

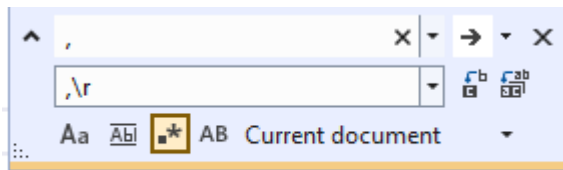
#### Other regex options

But changing tabs and spaces is just the start of what we can do.

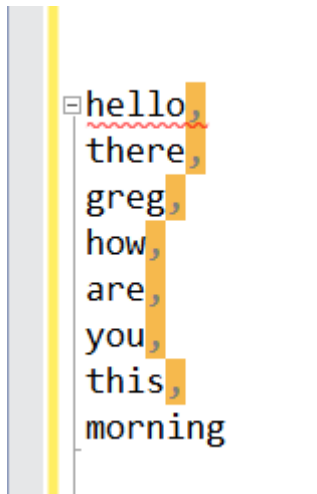
For example, if I have a long row of comma-delimited values like this:



Now I could go to each comma and hit enter after each one. That's what I see most people doing. But you could do this:



I'm changing a comma to a comma followed by a carriage return. And magically, this is the outcome:

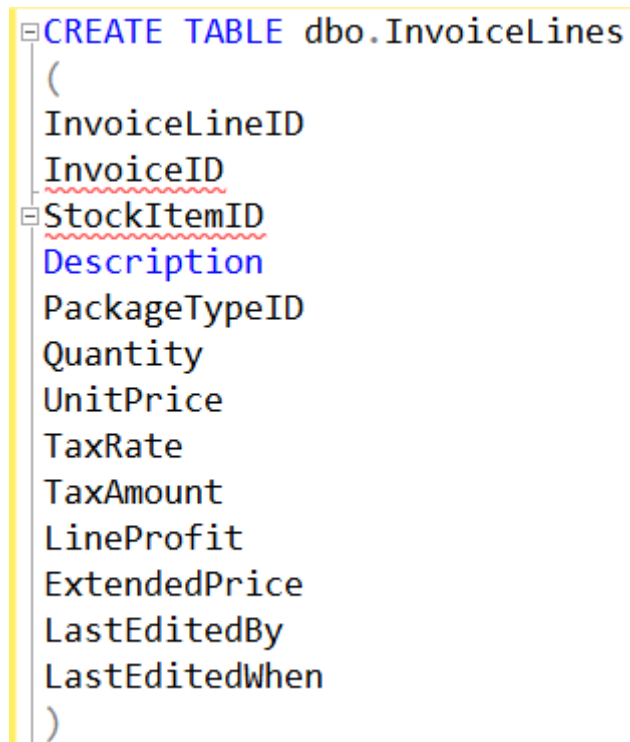


There are so many things you can use this regular expression functionality for within SSMS. I encourage you to try it out.

### 3.8 Selecting and modifying rectangular regions in SSMS

I often see people doing very repetitive editing tasks that could easily be carried out by using the selecting and changing rectangular regions of code.

The simplest example of doing this is to insert some text on a number of rows. Take the following code as an example:



```
CREATE TABLE dbo.InvoiceLines
(
    InvoiceLineID
    InvoiceID
    StockItemID
    Description
    PackageTypeID
    Quantity
    UnitPrice
    TaxRate
    TaxAmount
    LineProfit
    ExtendedPrice
    LastEditedBy
    LastEditedWhen
)
```

I've got the skeleton of a list of columns in a CREATE TABLE statement but let's assume that I'm a *comma in front* person and want to put a few spaces and a comma, etc. in front of each column after the second.

There are several ways to do this:

I **could** just put the cursor on the InvoiceID line and type what I wanted, then do the same again on the next line, and so on. That's tiring.

Another option would be to do it on the first line, then select and copy it, and insert it into the front of every other line.

What I should do, however, is to put the cursor in front of InvoiceID, and while holding Alt-shift, use the down arrow to select the beginning of every line, then just type what I want.



Similarly, if I want to change text on a bunch of lines, I can do the same by using both down arrow and side arrow to select a rectangular region, then just type to replace. In this example, I have a table name that I want to change from InvoiceLines to OrderLines:

```
Somedatabase.InvoiceLines.InvoiceID  
Somedatabase.InvoiceLines.StockItemID  
Somedatabase.InvoiceLines.Description  
Somedatabase.InvoiceLines.PackageTypeID  
Somedatabase.InvoiceLines.Quantity  
Somedatabase.InvoiceLines.UnitPrice  
Somedatabase.InvoiceLines.TaxRate  
Somedatabase.InvoiceLines.TaxAmount  
Somedatabase.InvoiceLines.LineProfit  
Somedatabase.InvoiceLines.ExtendedPrice  
Somedatabase.InvoiceLines.LastEditedBy  
Somedatabase.InvoiceLines.LastEditedWhen
```

So I select the starting point, Alt-shift and arrows to select the rectangle, then just type OrderLines.

### 3.9 Using the Clipboard Ring in SSMS

Two key combinations used by SQL Server T-SQL developers every day are Ctrl-C and Ctrl-V for copy and paste.

But many users of SQL Server Management Studio (SSMS) don't realize that it has a clipboard ring and can deal with several objects in the clipboard at the same time.

Let's see an example.

In this screen shot, I've opened a query window with the source code of the AnalyzeTableColumns procedure from SDU\_Tools.

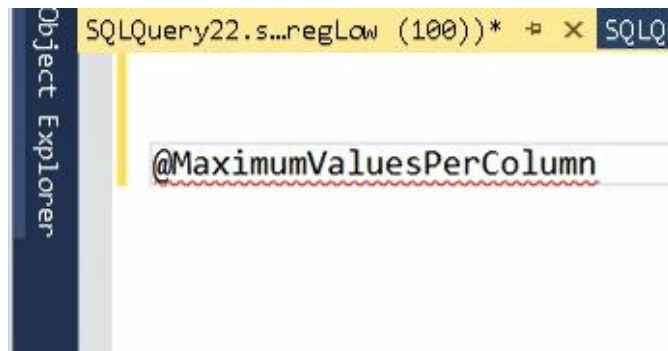
```
ALTER PROCEDURE [SDU_Tools].[AnalyzeTableColumns]
@DatabaseName sysname = NULL,
@SchemaName sysname = N'dbo',
@TableName sysname,
@OrderByColumnName bit = 1,
@OutputSampleValues bit = 1,
@MaximumValuesPerColumn int = 100
AS
BEGIN

-- Function:      Analyze a table's columns
-- Parameters:    @DatabaseName sysname          -> (default current database)
--               @SchemaName sysname            -> (default dbo) schema for t
```

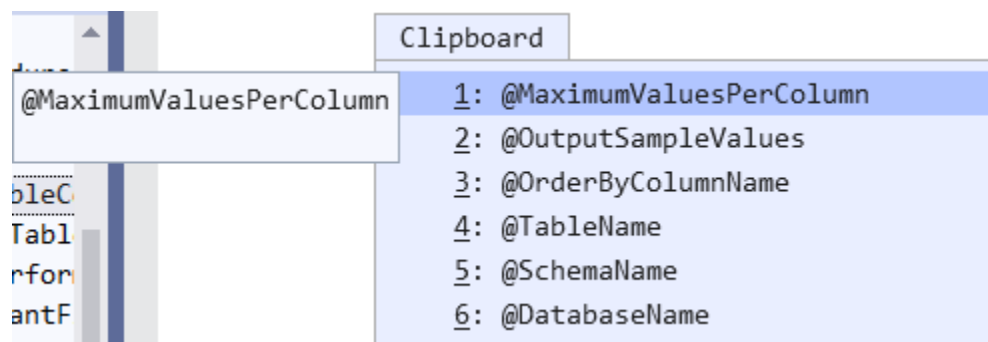
I might want to work with the parameters for that procedure, so I double-click and hit Ctrl-C for each of the parameter names.

```
ALTER PROCEDURE [SDU_Tools].[AnalyzeTableColumns]
@DatabaseName sysname = NULL,
@SchemaName sysname = N'dbo',
@TableName sysname,
@OrderByColumnName bit = 1,
@OutputSampleValues bit = 1,
@MaximumValuesPerColumn int = 100
AS
BEGIN
```

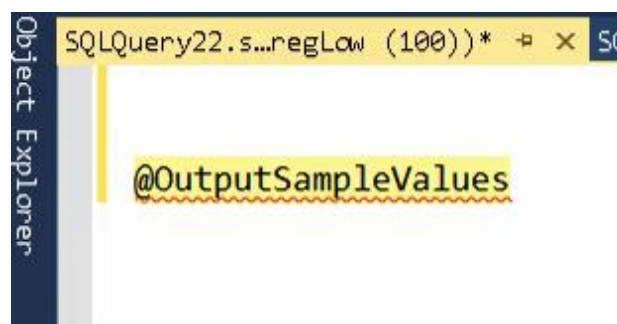
I've then opened a new query window where I want to work. If I hit Ctrl-V, I just get the last value that I copied, as expected:



However, instead of using Ctrl-V, if I use Ctrl-Shift-V, I see this:



I see the previous clipboard entries, and I can select one of them:



This is one of those things that once you get used to it, you'll wonder how you ever worked without it.

### Clearing the clipboard ring

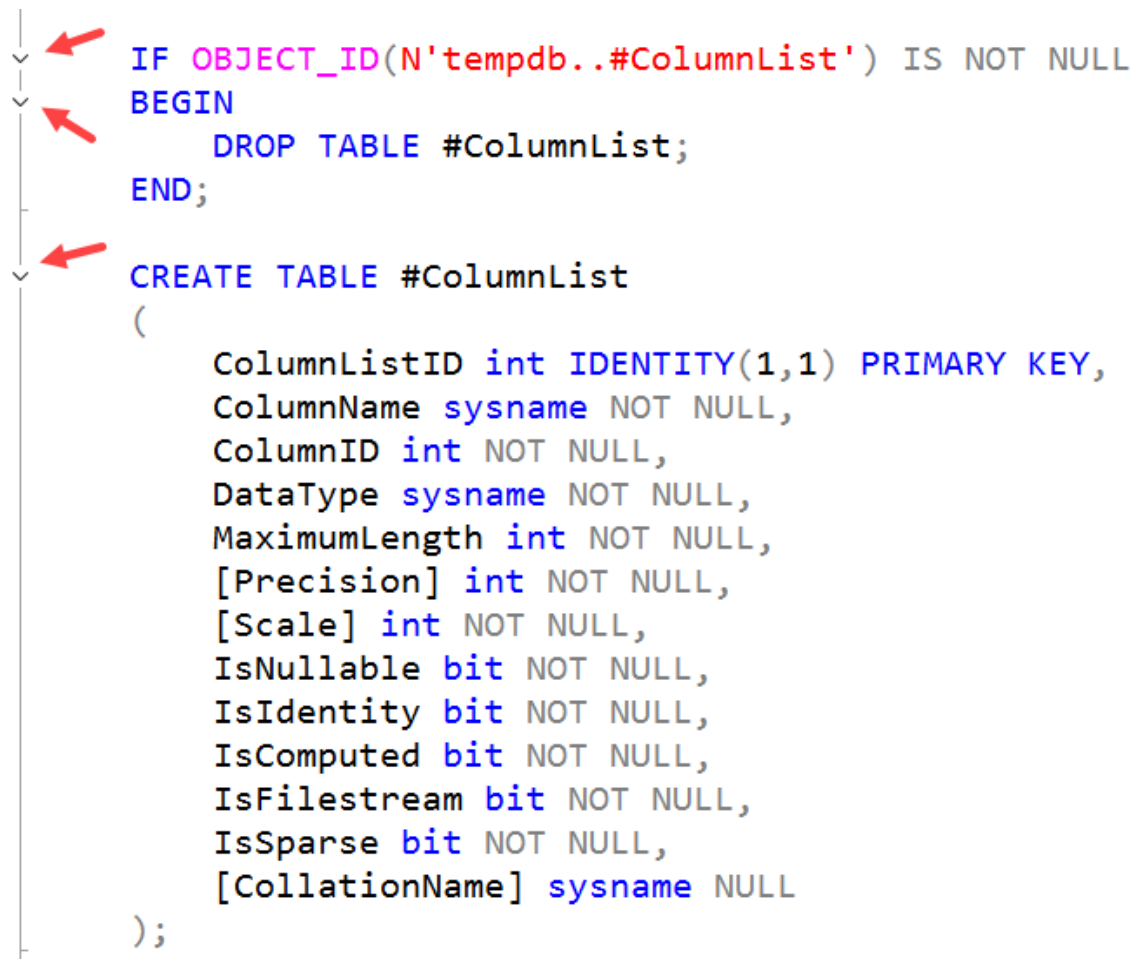
At present, there is no simple shortcut for clearing the clipboard ring. You could close SSMS and reopen it but that's a bit heavy-handed. I've seen someone suggest selecting a space, and using Ctrl-C 20 times but that seems silly too.

### 3.10 Using Code Outlining

For some years now, SQL Server Management Studio (SSMS) has had the ability to use code outlining, the same way that other Visual Studio languages can.

This can be very useful when you are trying to navigate around a large script file.

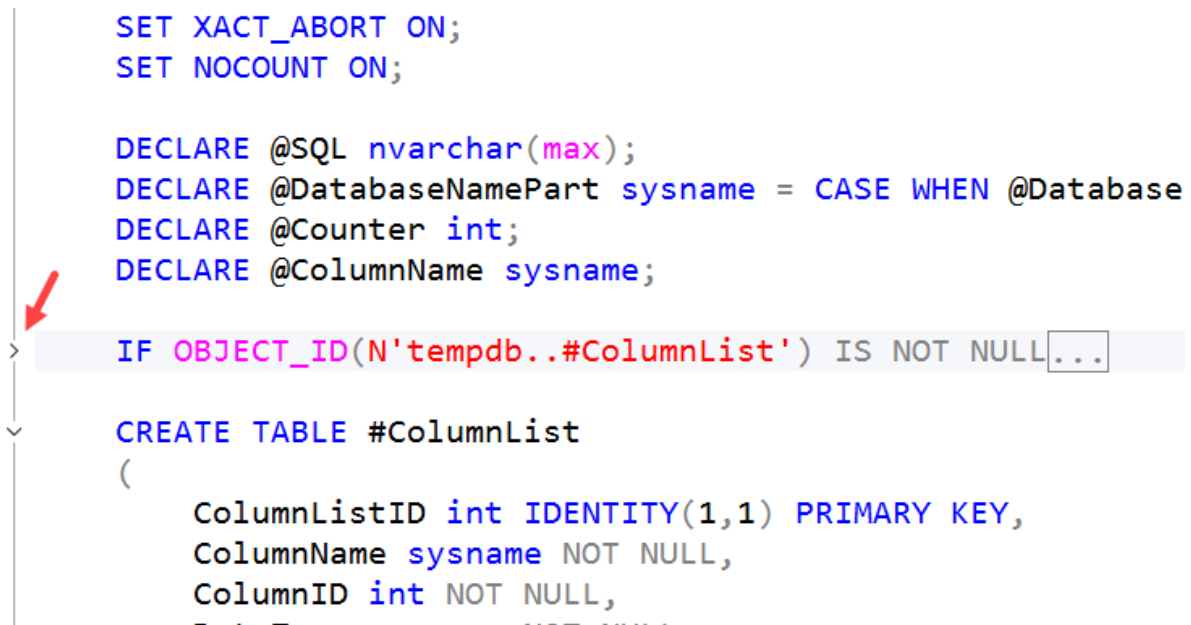
The simplest usage is to collapse or expand a region of code. Note that in the following script, code regions have been automatically added by SSMS:



```
IF OBJECT_ID(N'tempdb..#ColumnList') IS NOT NULL
BEGIN
    DROP TABLE #ColumnList;
END;

CREATE TABLE #ColumnList
(
    ColumnListID int IDENTITY(1,1) PRIMARY KEY,
    ColumnName sysname NOT NULL,
    ColumnID int NOT NULL,
    DataType sysname NOT NULL,
    MaximumLength int NOT NULL,
    [Precision] int NOT NULL,
    [Scale] int NOT NULL,
    IsNullable bit NOT NULL,
    IsIdentity bit NOT NULL,
    IsComputed bit NOT NULL,
    IsFilestream bit NOT NULL,
    IsSparse bit NOT NULL,
    [CollationName] sysname NULL
);
```

This allows us to click on the outline handles, and collapse the code:



```
SET XACT_ABORT ON;
SET NOCOUNT ON;

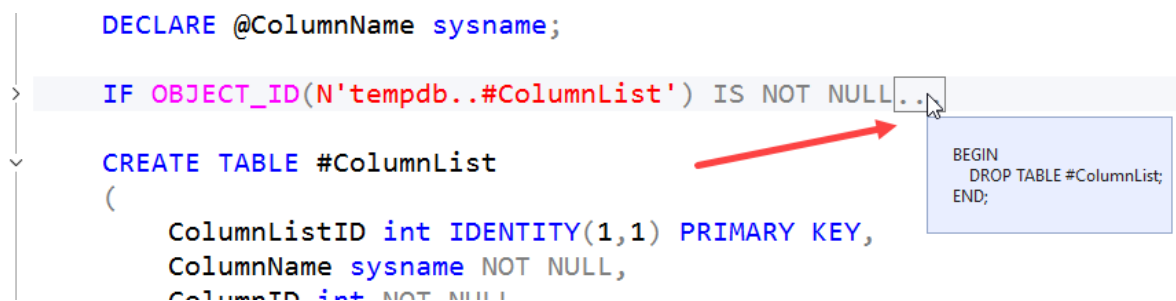
DECLARE @SQL nvarchar(max);
DECLARE @DatabaseNamePart sysname = CASE WHEN @DatabaseNamePart IS NULL THEN @DatabaseName ELSE @DatabaseNamePart END;
DECLARE @Counter int;
DECLARE @ColumnName sysname;

IF OBJECT_ID(N'tempdb..#ColumnList') IS NOT NULL...

CREATE TABLE #ColumnList
(
    ColumnListID int IDENTITY(1,1) PRIMARY KEY,
    ColumnName sysname NOT NULL,
    ColumnID int NOT NULL,
    ...
);
```

When the region of code is collapsed, the name of the region is shown as the first line of the code within the region, truncated.

If you hover over the ellipsis (the dot dot dot) at the beginning of the code region, you'll be shown what's contained within the region:



```
DECLARE @ColumnName sysname;

IF OBJECT_ID(N'tempdb..#ColumnList') IS NOT NULL...

CREATE TABLE #ColumnList
(
    ColumnListID int IDENTITY(1,1) PRIMARY KEY,
    ColumnName sysname NOT NULL,
    ColumnID int NOT NULL,
    ...
);
```

BEGIN  
DROP TABLE #ColumnList;  
END;

Now, what's missing? I'd love to be able to just drag regions around.

## Region Naming

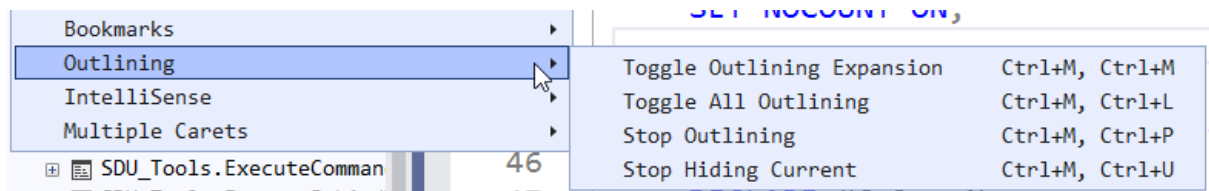
I'd also love to be able to name the regions better. It's not too bad if the regions are procedures or functions but for other chunks of code, there's really no good option.

Note that if I add a comment immediately above the code, it's not part of the same region. It might be better if it was like that, or if a specific comment could be treated as a region heading:

```
-- Drop the table if it already exists
IF OBJECT_ID(N'tempdb..#ColumnList') IS NOT NULL...

CREATE TABLE #ColumnList
(
```

In the Edit menu, the Outlining submenu doesn't show anything else useful at this point, apart from bulk operations:



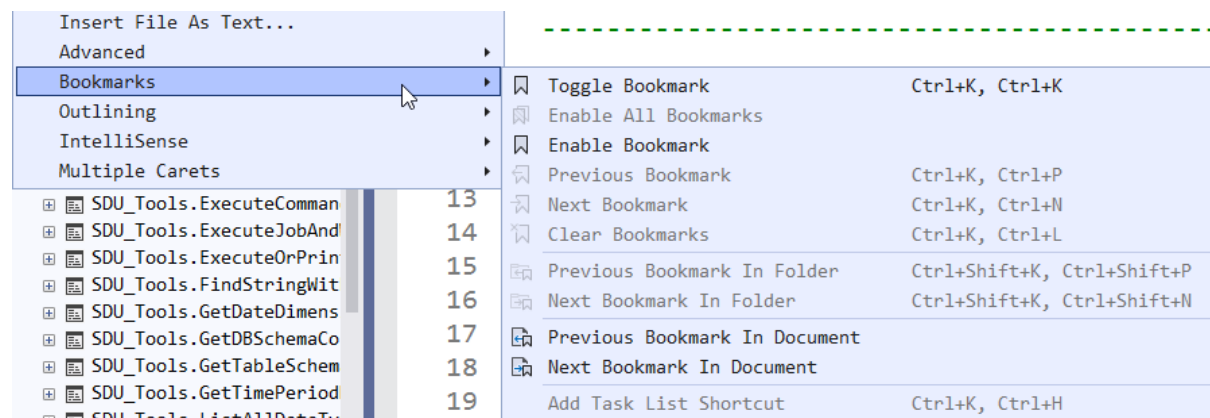
### 3.11 Using Bookmarks

In another section, I was discussing how outlining can be helpful with navigating around within a large T-SQL script file.

If you were trying to do that within a Microsoft Word document, the most common thing to use is **bookmarks**, and SQL Server Management Studio (SSMS) has them as well.

Bookmarks are simply placeholders within a script. (They can also apply to other types of document within SSMS). Where I find them very useful is when I'm working in two or three places within a long script at the same time. Perhaps I'm working on a function, and on the code that calls the function. By using bookmarks, I'm not flipping endlessly around the script file, and can jump directly from placeholder to placeholder.

A quick check of the Bookmarks submenu (under the Edit menu), shows what's available:



You toggle (enable or disable) a bookmark at a point, by using **Ctrl-K** and **Ctrl-K**. You can then navigate forwards or backwards using **Ctrl-K and Ctrl-N** (next), or **Ctrl-K and Ctrl-P** (previous).

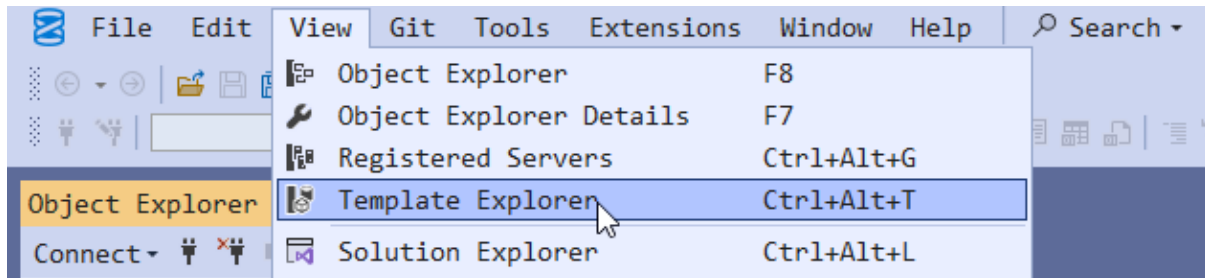
Note that there are options available for both the document and the folder. The folder option can be particularly powerful.

Bookmarks are often ignored but are a highly useful part of using SSMS. If you don't currently use them, you might want to consider them.

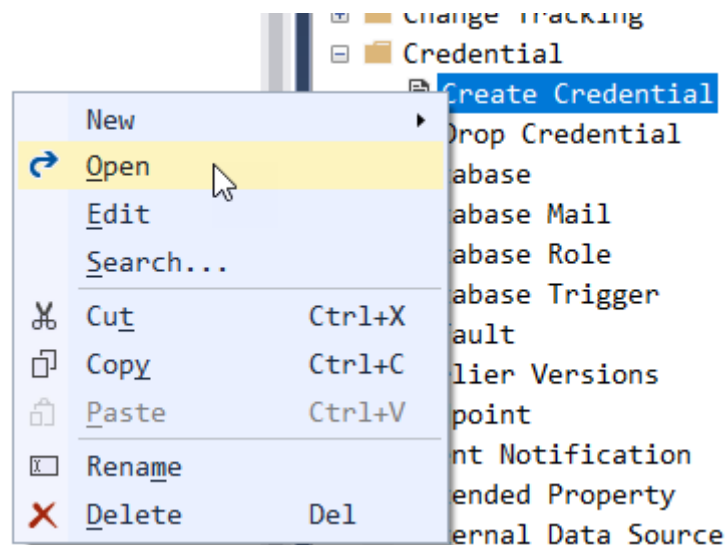
### 3.12 Using Query Templates

Sometimes you might need help to remember the T-SQL syntax required to create different types of objects. Templates can do this. Because they don't appear on the screen by default, many people don't even realize they are there.

From the View menu, you can choose to view Template Explorer:



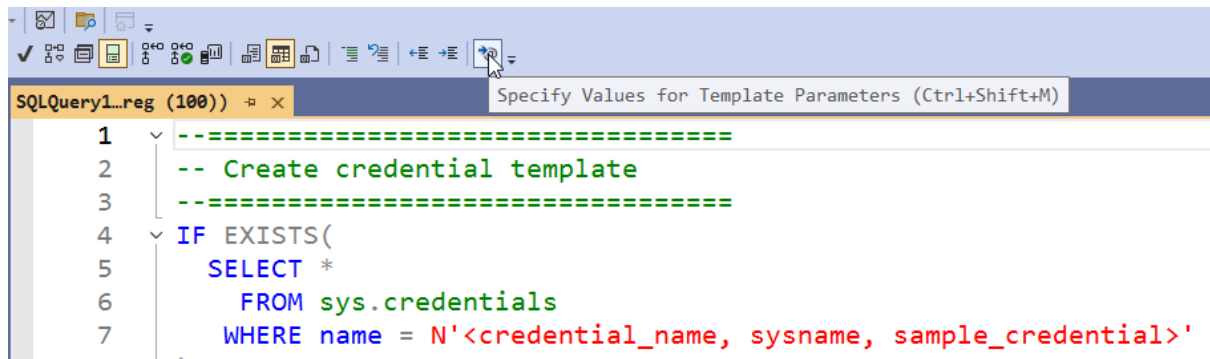
When Template Explorer opens, you see a list of predefined templates, plus any that you have created yourself. (One advantage of templates is that they are quite easy to create). In this case I've expanded the **Credential** folder, right-clicked the **Create Credential** template, then clicked **Open**.



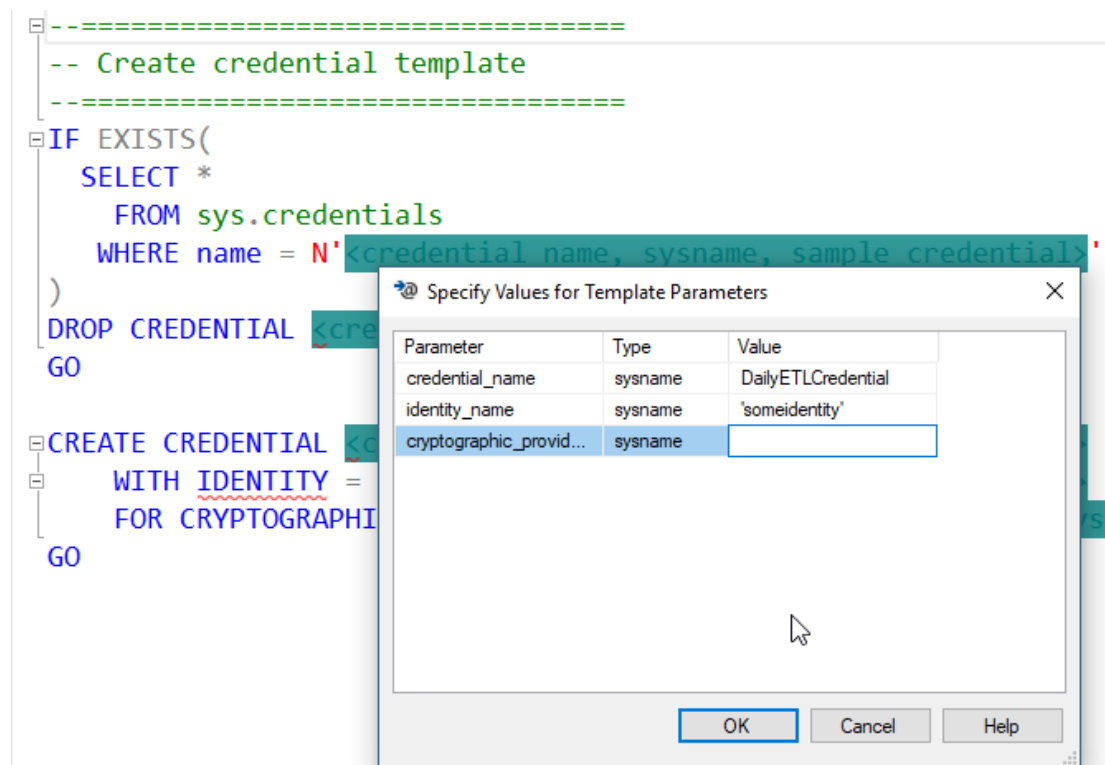
This then opens a new query window with the template for creating a credential inserted.



At this point, it's tempting to start editing the template manually, but instead click the **Specify Values for Template Parameters** button on the toolbar.



This finds all the declared parameter value, and opens a window where you can enter the values.



By completing the values in this window, you avoid the need to have to put the same value in many places. In this example, the name of the credential is used in several places.

```
--=====
-- Create credential template
--=====
IF EXISTS(
    SELECT *
    FROM sys.credentials
    WHERE name = N'DailyETLCredential'
)
DROP CREDENTIAL DailyETLCredential
GO

CREATE CREDENTIAL DailyETLCredential
    WITH IDENTITY = 'someidentity'
```

Templates are a great way to work with predefined code layouts, more than you can do with snippets. SQL Server Management Studio comes with a large number of T-SQL templates but this feature becomes even more powerful when you create your own. Soon, we'll discuss how to do that.

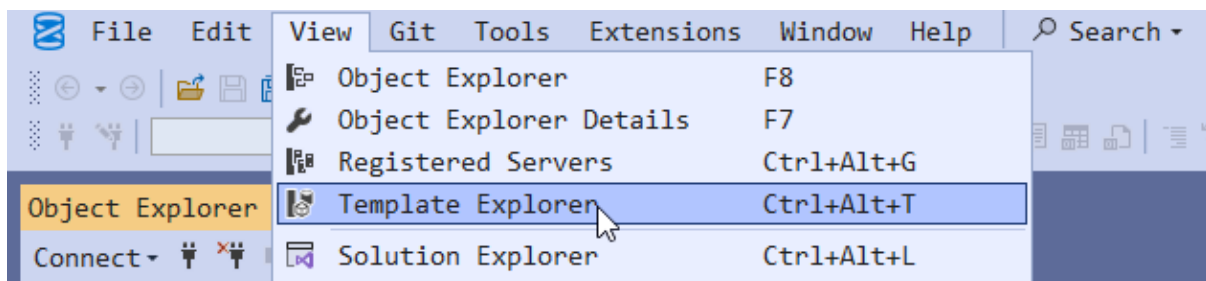
### 3.13 Creating T-SQL Templates

In another section, I mentioned how useful templates can be. I said that I'd discuss how to create them later. Well, that's today.

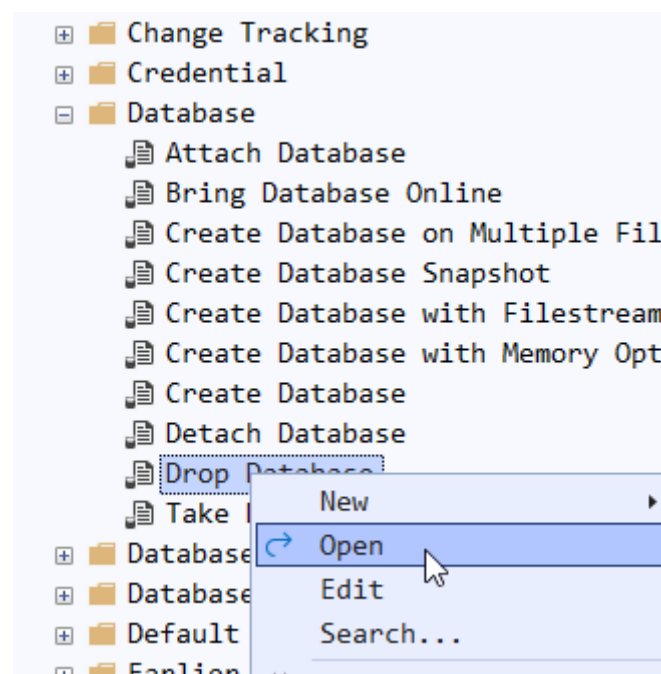
I thought I'd take dropping a database as an example. The supplied template doesn't work for me, so let's create a new one.

**Note:** SQL Server 2016 SP1 introduced **DROP DATABASE IF EXISTS** but I find that option quite useless. It fails if anyone is connected to the database. And to disconnect people beforehand, you need to first check if it exists, so the statement is pointless.

Let's start by opening the Template Explorer (from the View menu, click Template Explorer as it's not open by default).



Next, we look for a DROP DATABASE option, and right-click it.



When we click Open, we see the supplied template:

```
-- =====
-- Drop Database Template
-- =====

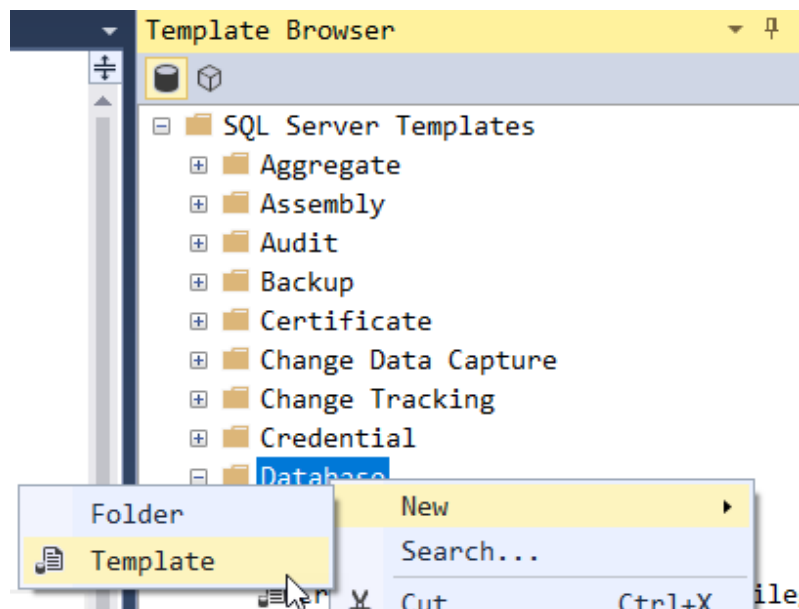
USE master
GO

IF EXISTS (
    SELECT name
      FROM sys.databases
     WHERE name = N'<Database_Name, sysname, Database_Name>'
)
DROP DATABASE <Database_Name, sysname, Database_Name>
GO
```

Importantly, note the text here that's shown between the less than and greater than operators. This is an example of a parameter to the template. Parameters look like:

**<ParameterName, DataType, DefaultValue>**

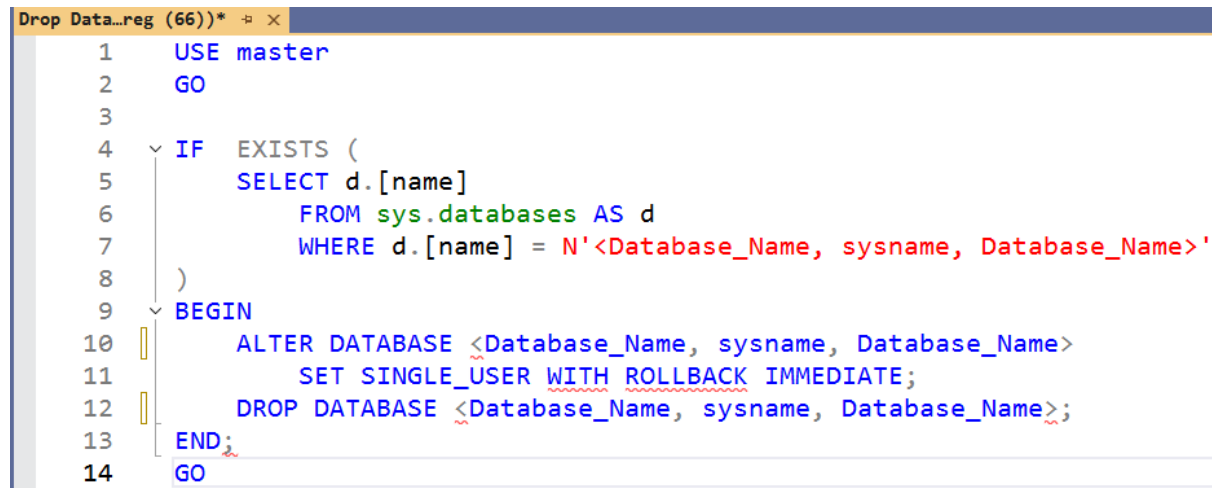
To create a new template, we could right-click the Database node in Template Explorer, and click New, then Template:



But to modify an existing one, it's easier to right-click the one we want to modify (ie: Drop Database), and click Copy. Then right-click where we want it (ie: the Database folder) and click Paste. That'll give us a copy of the template as a starting point.

Rename the new template to Drop Database - GL, right-click it, and click Edit.

I've changed the template as follows:



```
1  USE master
2  GO
3
4  IF EXISTS (
5      SELECT d.[name]
6      FROM sys.databases AS d
7      WHERE d.[name] = N'<Database_Name, sysname, Database_Name>'
8  )
9  BEGIN
10     ALTER DATABASE <Database_Name, sysname, Database_Name>
11         SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
12     DROP DATABASE <Database_Name, sysname, Database_Name>;
13 END;
14 GO
```

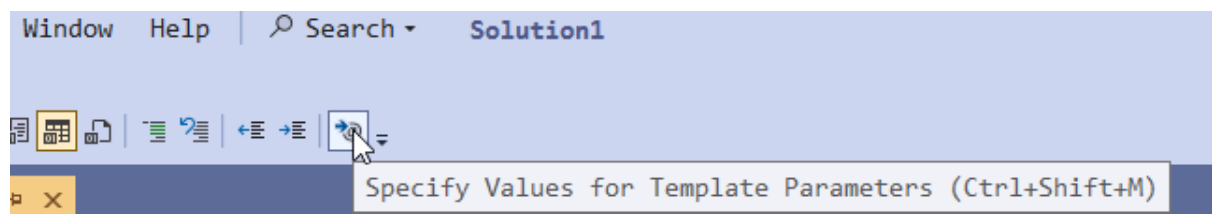
First up, I've used semicolons as statement terminators. We've been asked to use them since SQL Server 2005 so I have no idea why the existing templates (and much sample code from the SQL Server team) doesn't include them.

I've removed the comment headings. Otherwise, I'd need to do that every time. I've quoted the [name] column as I like to see the column names colorized properly, and this name would otherwise be blue. I've also aliased the table.

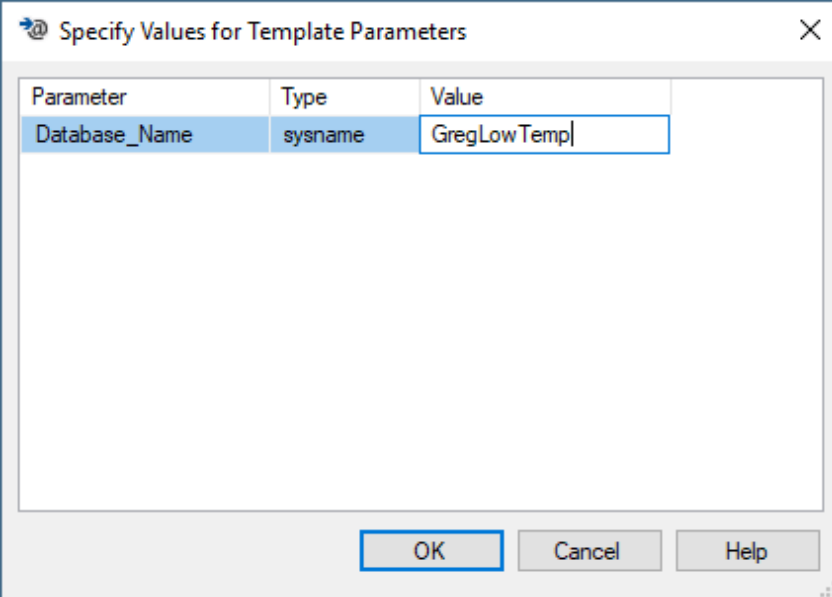
Next, to try to get around dropping a database with people already connected, I've altered it to single user, with an option to kick off all but a single user. This isn't even necessarily perfect, but it will be far better than the supplied template.

From the File menu, I click Save, then close the template window.

Now let's try it. From Template Explorer, I right-click **Drop Database - GL**, and click Open, and my new template appears. Then, on the toolbar, I click the option to let me enter the parameters:



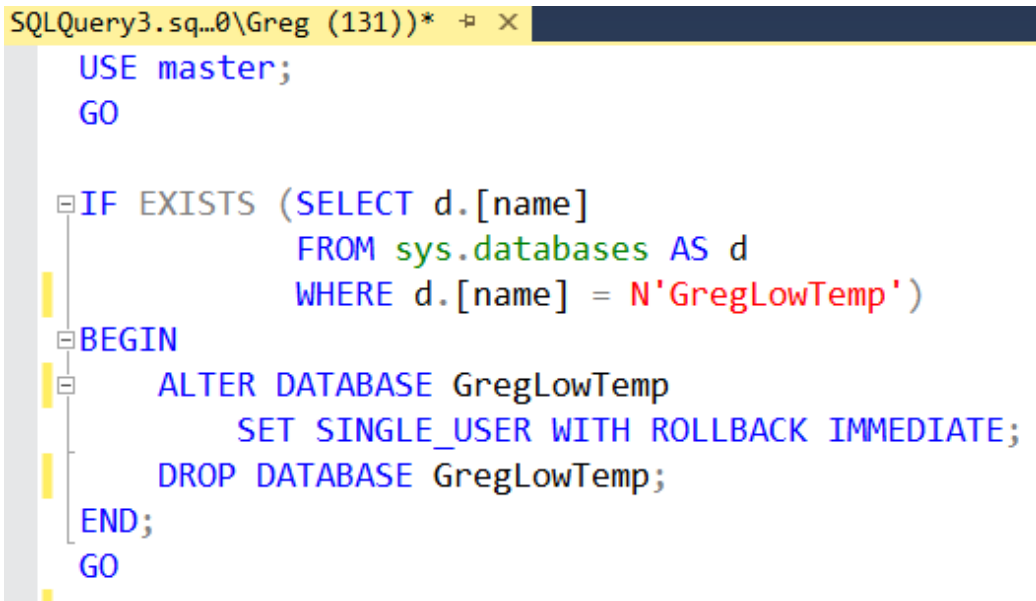
I enter the parameter value:



A dialog box titled "Specify Values for Template Parameters" with a close button (X) in the top right corner. It contains a table with three columns: "Parameter", "Type", and "Value". The first row is selected, showing "Database\_Name" as the parameter, "sysname" as the type, and "GregLowTemp" as the value. At the bottom of the dialog are three buttons: "OK", "Cancel", and "Help".

Parameter	Type	Value
Database_Name	sysname	GregLowTemp

After I click OK, my template is ready for use:



The image shows a SQL query editor window with a tab titled "SQLQuery3.sql...0\Greg (131))\*". The query text is as follows:

```
USE master;
GO

IF EXISTS (SELECT d.[name]
           FROM sys.databases AS d
           WHERE d.[name] = N'GregLowTemp')
BEGIN
    ALTER DATABASE GregLowTemp
        SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
    DROP DATABASE GregLowTemp;
END;
GO
```

The parameter value has been replaced wherever it had been used.

Once I see it, I note that it would be better if I had quoted the database name. Templates will often need to be refined over time.

As a side note, this is still not 100% bulletproof anyway. Note that the command is executed in the master database, and so there is no guarantee that the single user will be you. You could use RESTRICTED\_USER but again, that might have the same issue. What is really needed is this:

```
DROP DATABASE IF EXISTS GregLowTemp WITH ROLLBACK IMMEDIATE;
```

But sadly, **that command doesn't exist**. There really should be a way to reliably drop a database by standard T-SQL. Many people need to write **drop and recreate** scripts.

Finally, I'll now point out that this code would be better as a snippet than as a template. Templates work well when they provide an entire script. This code is likely to be used as part of another script. In a later section, I'll show you how to make this a snippet instead.

### 3.14 Using Snippets

Have you ever started to create an object using T-SQL in SQL Server, and thought: "what's the right syntax for this?"

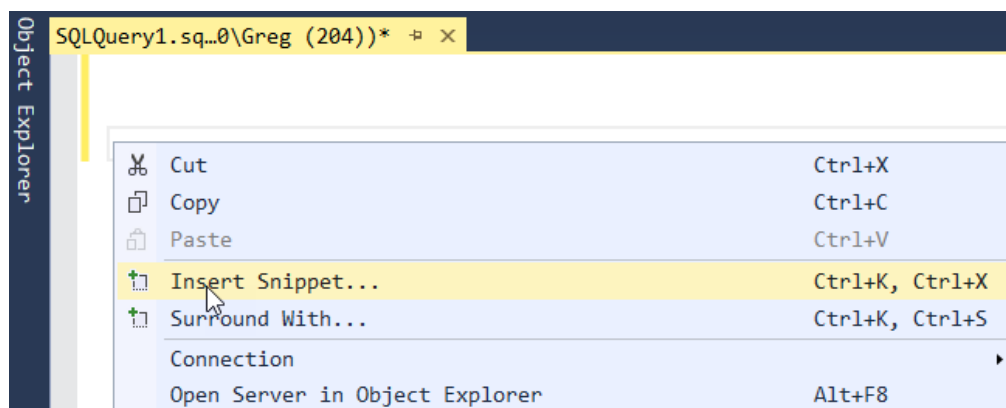
I've worked with SQL Server since 1992 (version 4.2) and yet almost every time I go to create a function, I must spend a few moments thinking about what the correct syntax is, because there are different types of functions (scalar vs table-valued, inline vs multi-statement).

SSMS has had templates for a long time, and they are useful. In fact, I've written about how you can create your own.

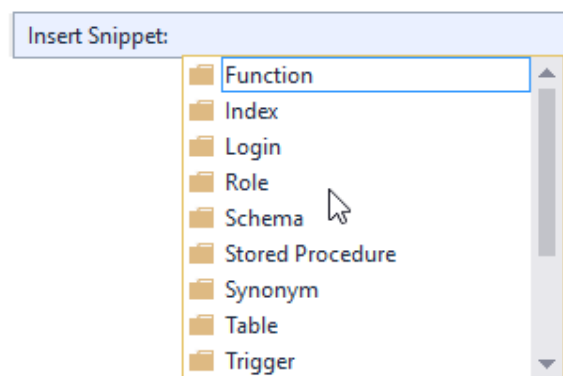
But what I wanted to focus on here, is another option that was added a while back, yet most people seem to be unaware is there. And that's **snippets**.

One of the easiest ways to get the basic syntax for creating an object is to use a snippet.

In a query window, right-click a blank area and note the option for inserting a snippet (you can also do this by Ctrl-K, then Ctrl-X):

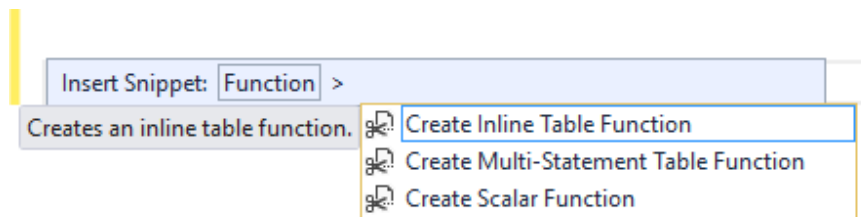


Then select the type of object you want:





In this case, let's choose **Function**, and then we'll see the types of functions that are available:



Let's choose **Create Inline Table Function**, and SSMS fills in a skeleton for us:

```
CREATE FUNCTION [dbo].[FunctionName]
(
    @param1 int,
    @param2 char(5)
)
RETURNS TABLE AS RETURN
(
    SELECT @param1 AS c1,
           @param2 AS c2
)
```

Great. But don't get carried away with your mouse just yet. Notice that the template has placeholders that you can replace and in this case, the schema is pre-highlighted. So I can just type the schema name, and hit tab, then type the function name, and hit tab, and so on until all placeholders are complete. Note that it has also stubbed out a SELECT statement for us, and as we change the parameter names, it changes the names in the SELECT as well.

```
CREATE FUNCTION [Sales].[GetTaxRatesByCategory]
(
    @ProductCategory int,
    @CountryCode nvarchar(10)
)
RETURNS TABLE AS RETURN
(
    SELECT @ProductCategory AS c1,
           @CountryCode AS c2
)
```

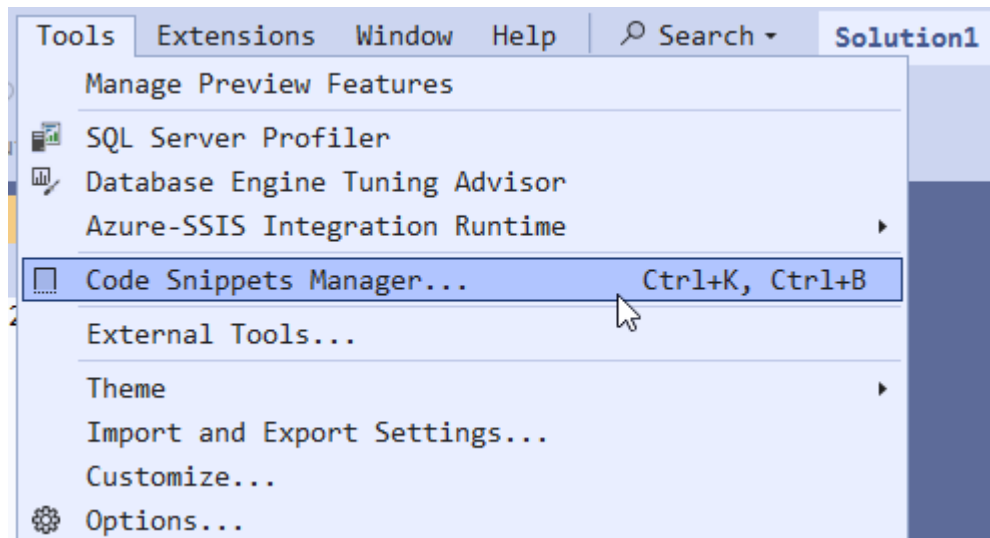
I've no doubt that the product team thinks I'm a bit pedantic about these things, but given we've been encouraging people to use semicolons as statement terminators since 2005, I wish they'd include that in the statement that they've provided. I must ping them about that. But regardless, snippets are wonderful and help to avoid that "what's the syntax for this?" issue.

### 3.15 Using Snippets to Improve the Drop Database Statement

Earlier, I showed how to create a DROP DATABASE template in SSMS. I mentioned that a template wasn't the best option because a command like this is normally inserted into a script; it's not the whole script.

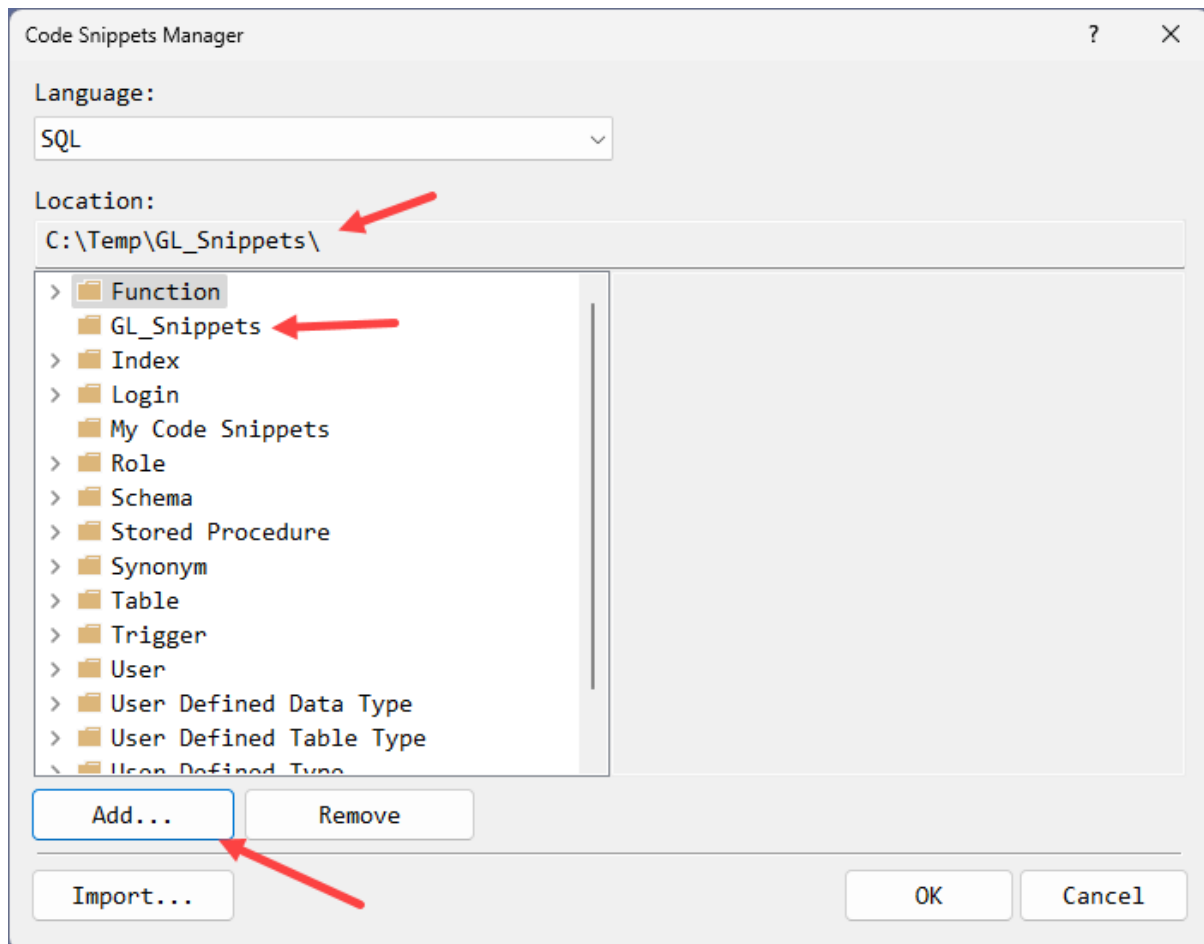
That's where snippets shine. Let's create a snippet for it.

First let's open **Code Snippets Manager** (Tools > Code Snippets Manager):

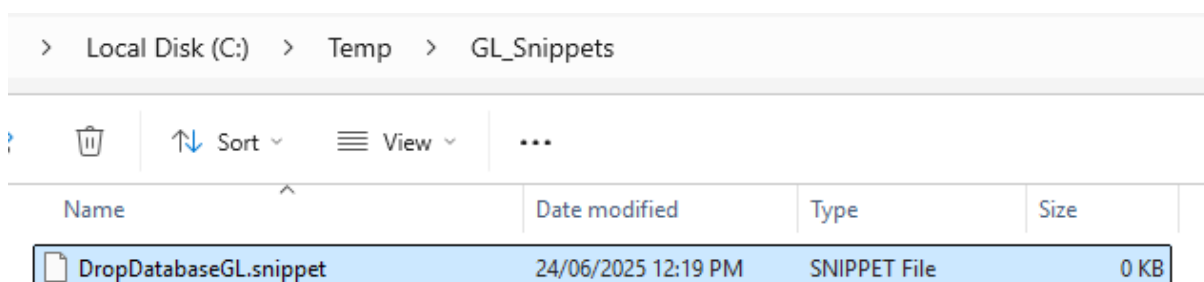


You'll see the existing snippet folders.

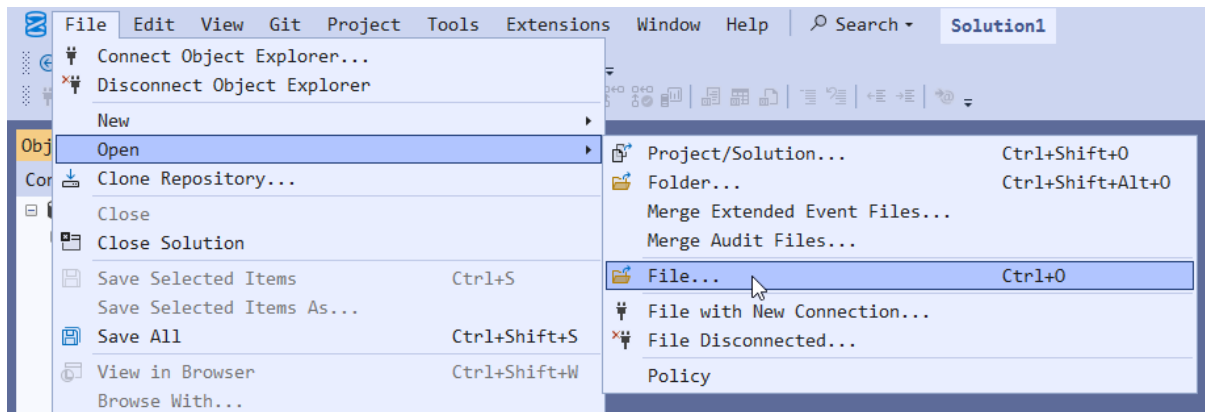
I've clicked **Add**, then created a new folder called **GL\_Snippets**. I created a new folder and pointed to it. For this temporary one, I've created it in C:\Temp but you would normally use an appropriate folder.



Next I've created a file called **DropDatabase.snippet** in that folder:



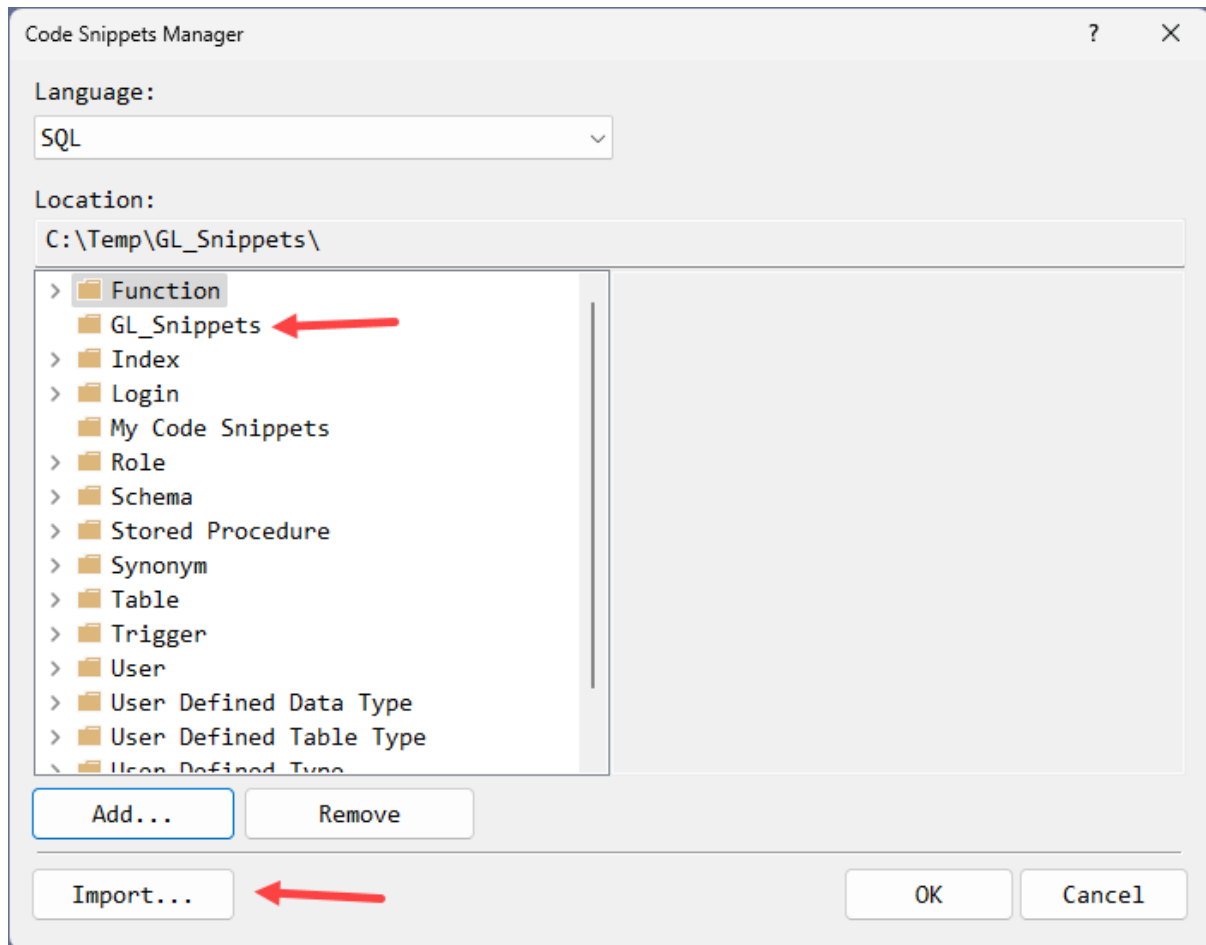
I then opened that file using SSMS:



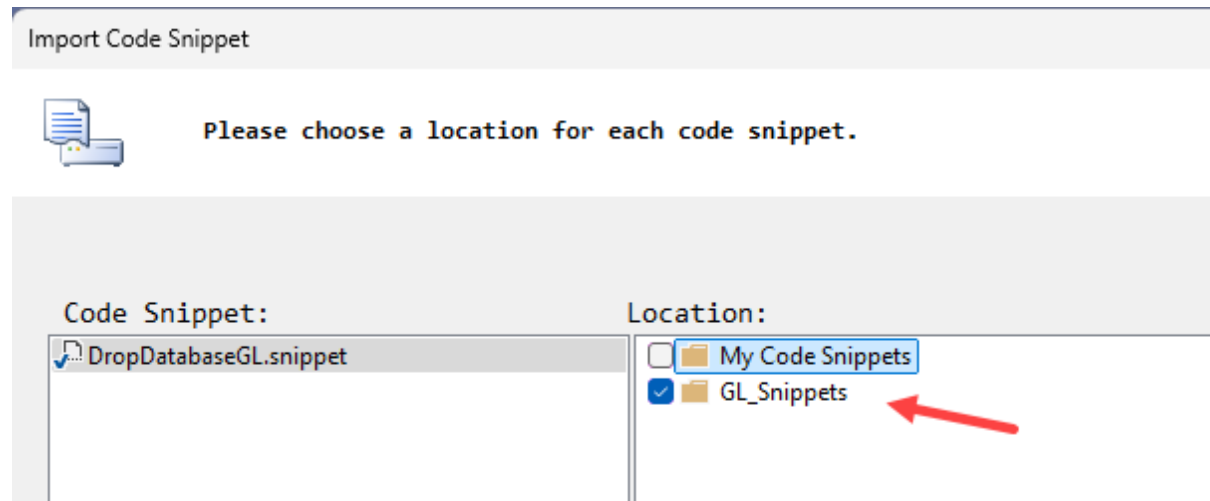
Note that SSMS has a perfectly good XML editor. I've then used it to create the snippet file and saved it:

```
DropDatab.L.snippet* x
1  <?xml version="1.0" encoding="utf-8" ?>
2  <CodeSnippets xmlns="http://schemas.microsoft.com/VisualStudio/2005/CodeSnippet">
3    <CodeSnippet Format="1.0.0">
4      <Header>
5        <Title>DropDatabaseGL</Title>
6        <Description>Drop a database if it exists GL</Description>
7        <Author>Dr Greg Low</Author>
8      </Header>
9      <Snippet>
10        <Declarations>
11          <Literal>
12            <ID>DatabaseName</ID>
13            <ToolTip>Name of the database to drop if it exists</ToolTip>
14            <Default>GregLowTemp</Default>
15          </Literal>
16        </Declarations>
17        <Code Language="SQL"><![CDATA[
18  USE master;
19  GO
20
21  IF EXISTS (SELECT 1 FROM sys.databases AS d WHERE d.[name] = N'$DatabaseName$')
22  BEGIN
23    ALTER DATABASE [$DatabaseName$] SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
24    DROP DATABASE [$DatabaseName$];
25  END;
26  GO
27  ]]>
28        </Code>
29      </Snippet>
30    </CodeSnippet>
31  </CodeSnippets>
```

Back in Code Snippets Manager, I've clicked **Import**:



I located the file to import, deselected the **My Code Snippets** folder, and selected the **GL\_Snippets** folder, then clicked **Finish**:

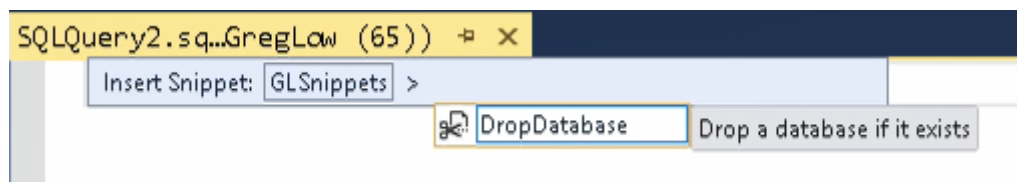


If you are prompted about the file already existing, just choose the Overwrite option.

Now let's try the new snippet. I opened a new query window, and right-clicked in the empty space. We can then see the option for **Insert Snippet**(note we could have used Ctrl+K, Ctrl+X):



Then select our new DropDatabase snippet:



The snippet will then appear, with the literal parameter highlighted, ready for replacement:

```
USE master;
GO

/ IF EXISTS (SELECT 1 FROM sys.databases AS d WHERE d.[name] = N'GregLowTemp')
/ BEGIN
    ALTER DATABASE [GregLowTemp] SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
    DROP DATABASE [GregLowTemp];
END;
GO
```

I typed **SomeDatabase** and then clicked elsewhere, and it's all magically updated:

```
USE master;
GO

/ IF EXISTS (SELECT 1 FROM sys.databases AS d WHERE d.[name] = N'SomeDatabase')
/ BEGIN
    ALTER DATABASE [SomeDatabase] SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
    DROP DATABASE [SomeDatabase];
END;
GO
```

---

This is really useful for any code that you find yourself typing all the time.

### 3.16 Using "Surround with" snippets

In the previous sections, I've been talking about how to use snippets in SSMS and how to create your own. There are several types of snippets and one of the special types of snippets that I want to mention are the "surround with" snippets.

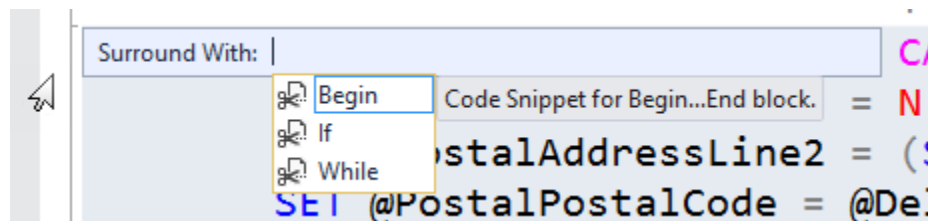
If you look at the following block of code:

```
SET @DeliveryAddressLine2 = CAST(CEILING(RAND() * 2000) + 1 AS nvarchar
                                + (SELECT TOP(1) PreferredName FROM [Applica
                                + N' ' + @StreetSuffix;

SET @DeliveryPostalCode = CAST(CEILING(RAND() * 800) + 90000 AS nvarchar
SET @PostalAddressLine1 = N'PO Box ' + CAST(CEILING(RAND() * 10000) +
SET @PostalAddressLine2 = (SELECT TOP(1) PreferredName FROM [Applicati
SET @PostalPostalCode = @DeliveryPostalCode;

SET @PrimaryContactPersonID = NEXT VALUE FOR Sequences.PersonID;
```

Imagine that you want to execute the four highlighted lines only when a condition is true. If I hit Ctrl-K and Ctrl-S while they are highlighted, I'm prompted with this:



Note that I get an option to surround them with a BEGIN/END, or an IF or WHILE. Let's choose an IF. I just double-click the If option and this happens:

```
SET @DeliveryAddressLine2 = CAST(CEILING(RAND() * 200
                                + (SELECT TOP(1) PreferredN
                                + N' ' + @StreetSuffix;

IF( Condition )
BEGIN

    SET @DeliveryPostalCode = CAST(CEILING(RAND() * 800)
    SET @PostalAddressLine1 = N'PO Box ' + CAST(CEILING(R
    SET @PostalAddressLine2 = (SELECT TOP(1) PreferredNam
    SET @PostalPostalCode = @DeliveryPostalCode;

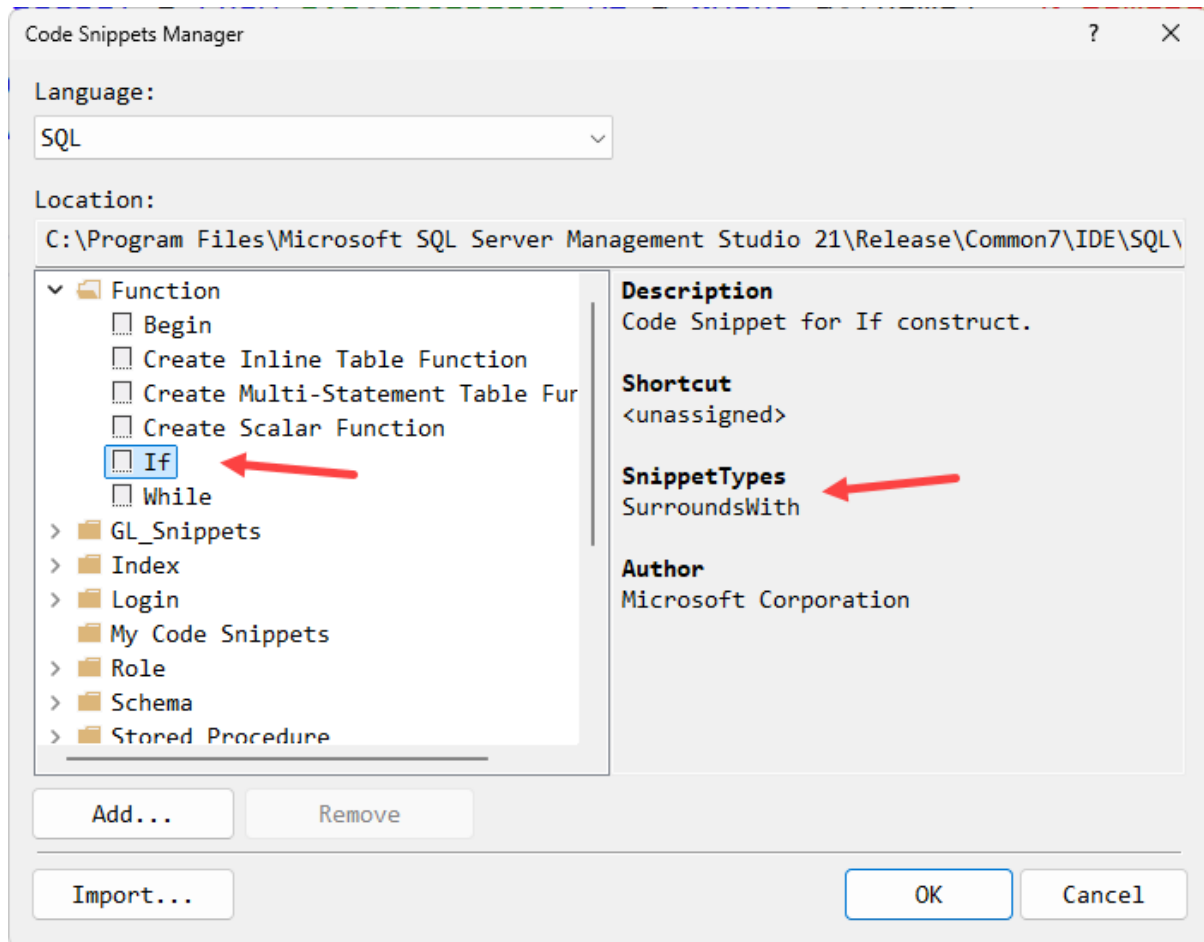
END
```

We can then just type the condition. Notice that because I was using multiple lines, it put them all in a BEGIN/END block for me. How convenient.



I still might want to create my own instead, and I can do that. An example of why I might want to do that, is that I might want a statement terminator after the END.

If we go into the Code Snippets Manager (from the Tools menu), and expand the Function category, we can see this:



I can't say that I think of IF as a function but none-the-less, note that the type is a SurroundsWith snippet. Let's see how it's defined. The Location is shown above.

If I open that file path in SSMS, I see the following:



```
<?xml version="1.0" encoding="utf-8" ?>
<CodeSnippets xmlns="http://schemas.microsoft.com/VisualStudio/2005/CodeSnippet">
  <_locDefinition xmlns="urn:locstudio">
    <_locDefault _loc="locNone" />
    <_locTag _loc="locData">Title</_locTag>
    <_locTag _loc="locData">Description</_locTag>
    <_locTag _loc="locData">Author</_locTag>
    <_locTag _loc="locData">ToolTip</_locTag>
    <_locTag _loc="locData">Default</_locTag>
  </_locDefinition>
  <CodeSnippet Format="1.0.0">
    <Header>
      <Title>If</Title>
      <Shortcut></Shortcut>
      <Description>Code Snippet for If construct.</Description>
      <Author>Microsoft Corporation</Author>
      <SnippetTypes>
        <SnippetType>SurroundsWith</SnippetType>
      </SnippetTypes>
    </Header>
    <Snippet>
      <Declarations>
        <Literal>
          <ID>Condition</ID>
          <ToolTip>Condition to evaluate</ToolTip>
          <Default>Condition</Default>
        </Literal>
      </Declarations>
      <Code Language="SQL"><![CDATA[
IF( $Condition$ )
BEGIN

$selected$ $end$

END
]]>
      </Code>
    </Snippet>
  </CodeSnippet>
</CodeSnippets>
```

Note how the value called **\$selected\$** is enclosed within the snippet.

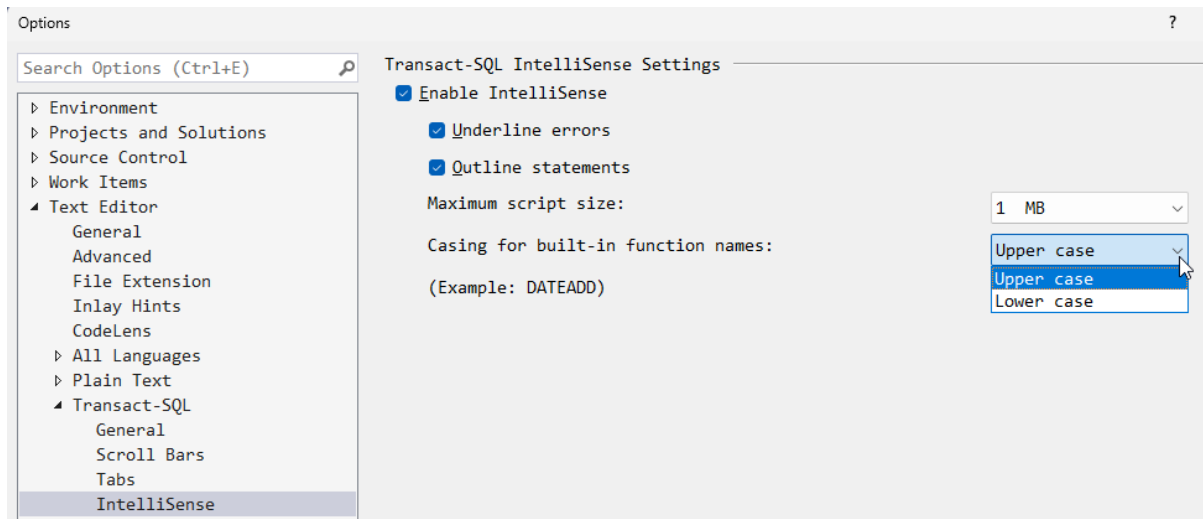
I could then either modify this one (probably not best) or use it as the basis of a new snippet for myself.

### 3.17 Intellisense casing for function names

When I write T-SQL, my standard is to use upper-case names for built-in system functions. But not everyone likes that.

I've done a lot of work on systems where all these names are lower-case. I don't have any great objection to that, but it's painful if Intellisense keeps making them upper-case.

But there is a solution. In Text Editor, Transact-SQL, then Intellisense, there is an option for this:

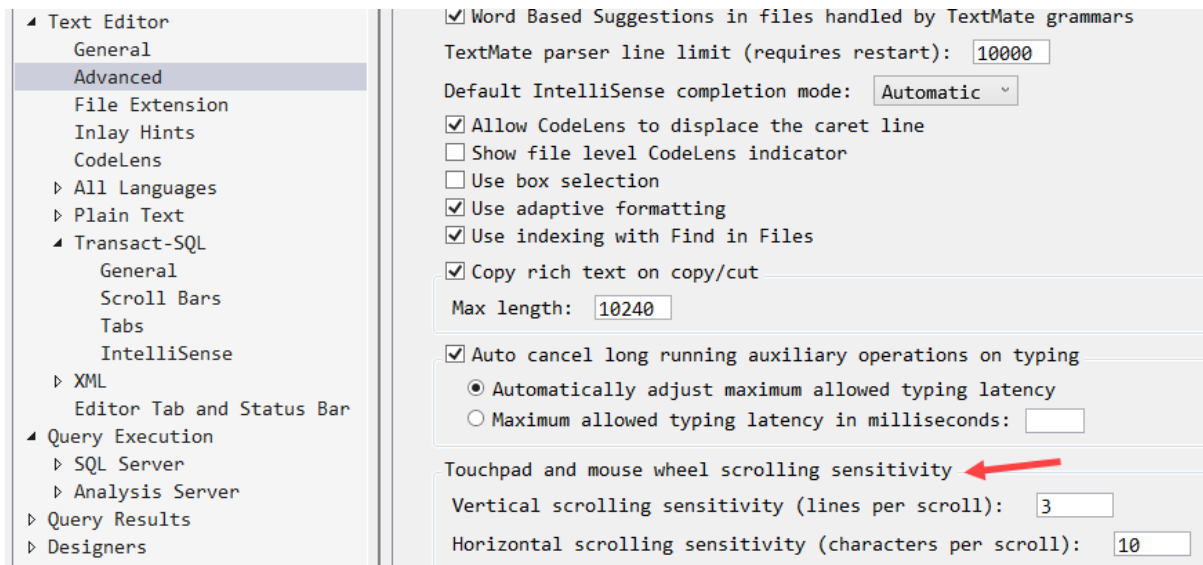


### 3.18 Scrolling Sensitivity

When I'm editing large script files in SSMS, I sometimes find that the scrolling speed of my mouse wheel is too fast or too slow.

If you don't like the current speed, SSMS has an option to adjust that.

It's in the Text Editor, Advanced, section. It allows you to control how many lines you get per scroll (for vertical scrolling) or how many characters (for horizontal scrolling):



## 4 Executing Queries

### 4.1 Adding additional parameters to connections

When I am writing my own code using a .NET (or other) language, I have a great deal of control of how the connection string that my application uses to connect to SQL Server is configured.

In particular, I might need to add another parameter or two.

As a simple example, I might be working with a multi-subnet Availability Group, spread across a production site and a disaster recovery site. It's common to then have an Availability Group Listener in both subnets.

If you add the parameter **MultiSubnetFailover=true** to your connection string, when SQL Server attempts to connect to the listener, it will send a request to each IP address concurrently, not just to one at a time. It will then connect to whichever server responds first.

This is great, but how do we do that with SQL Server Management Studio (SSMS) connections?

The answer is that in the database server connection dialog, we can choose **Advanced**:

Connection Properties

Server Name: .\SQL2022

Authentication: Windows Authentication

User Name: GREG7680V2\Greg

Password:

☒ Remember Password

Database Name: <default>

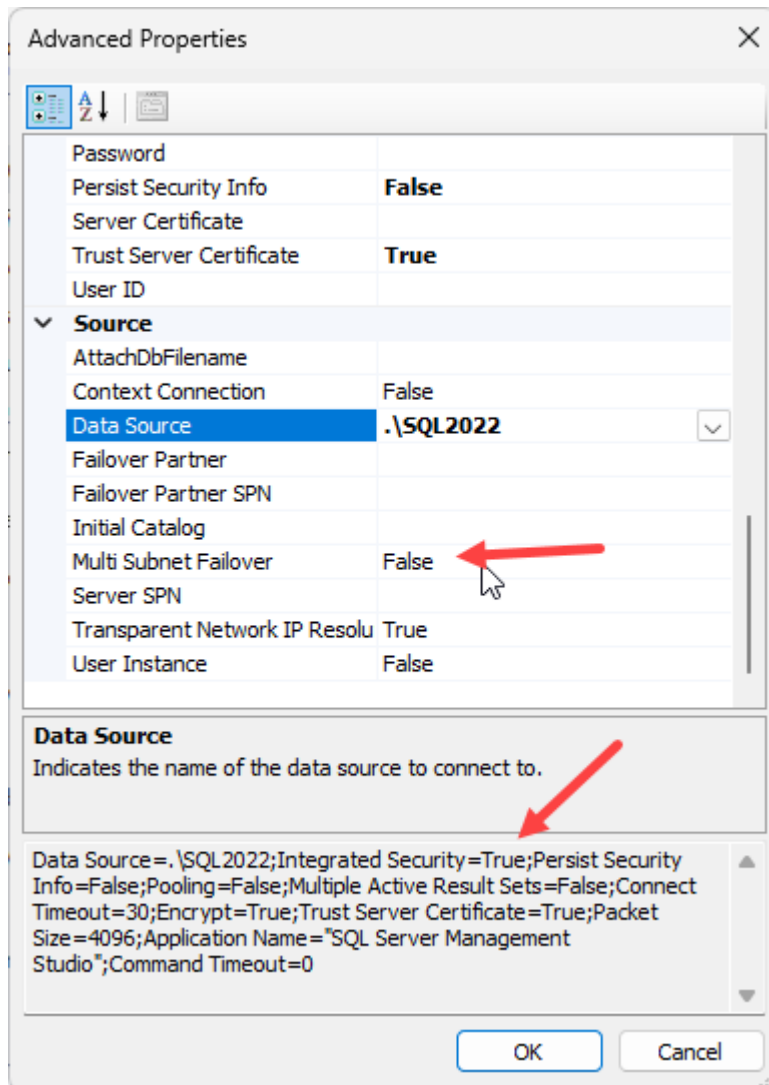
Encrypt: Mandatory

☒ Trust Server Certificate

Color: <default> Custom... Advanced...

Connect Cancel Help

In the dialog that appears, there are many more options that can be configured for the connection:



## 4.2 Using Colors to Avoid Running Scripts Against the Wrong Server

Everyone who's worked with SQL Server for any length of time, has had the experience of executing a T-SQL script, and then noticing, with horror, that they've just executed the script against the wrong server.

You know the feeling. It even happens at Christmas time, just when you were hoping to get away from work for a few days, or when you are the unlucky one who's doing on call work.



Many of these surprises would be avoided if there was something that gave you a visual clue that you were connected to the wrong server.

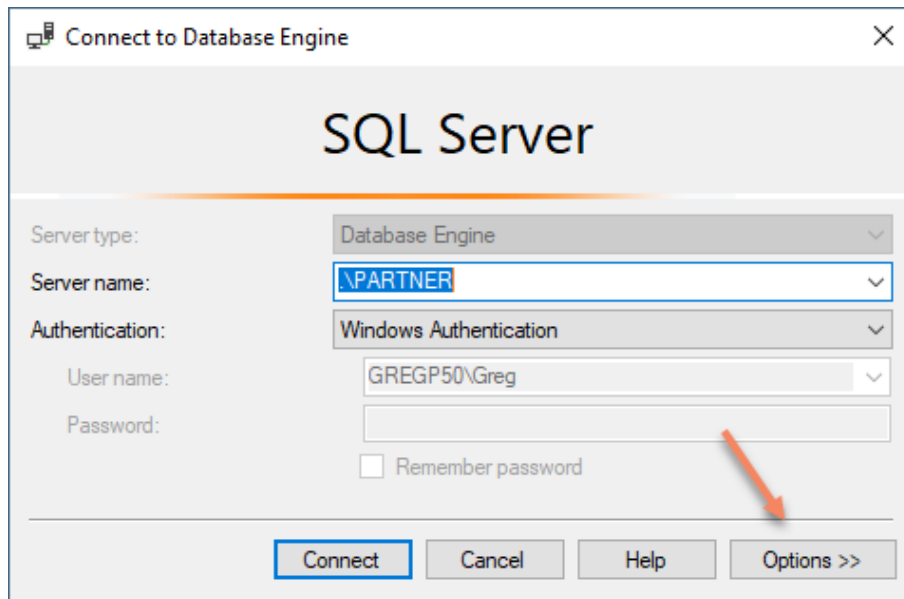
SSMS has had an option to color code connections to servers for quite a while. The solution isn't perfect and isn't as good as Mladen Prajdić's SSMS Tools Pack:

<https://www.ssmstoolspack.com/>

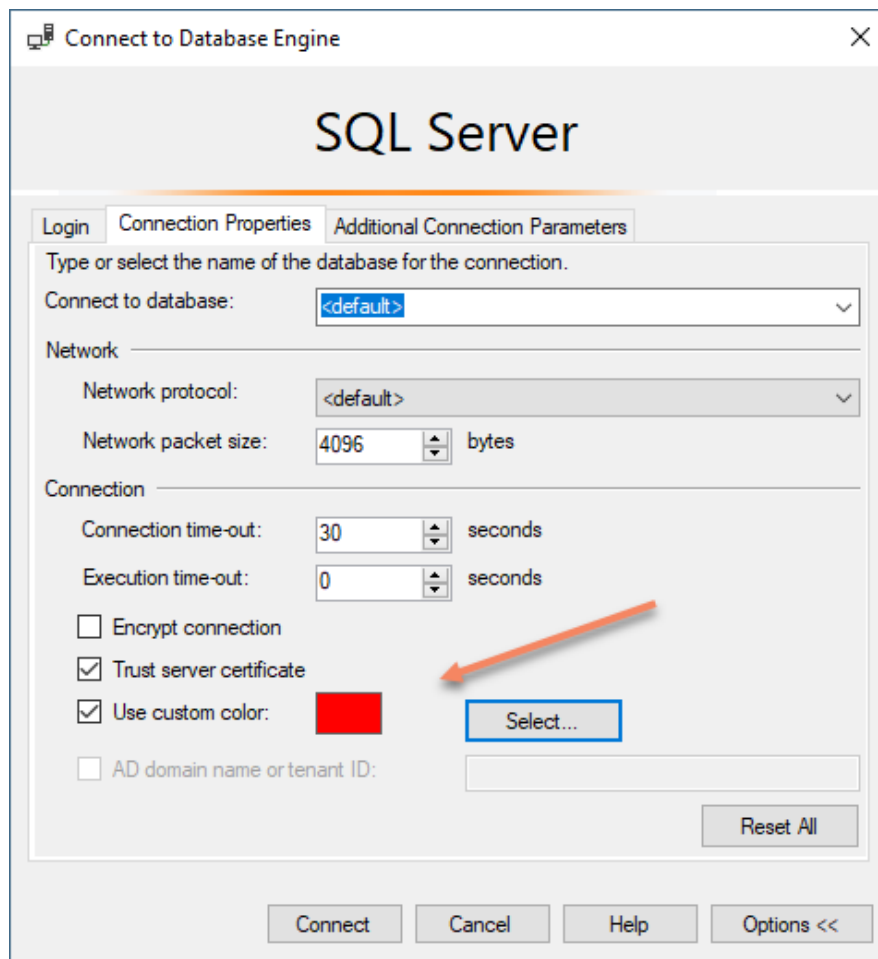
(If you're keen to pay for an add-in, I do recommend that one)

For many people though, the colorizing provided by SSMS is just enough. And it's easy to use.

In previous versions of SSMS, you needed to click **\*\*Options\*\*** when you opened a new database connection.

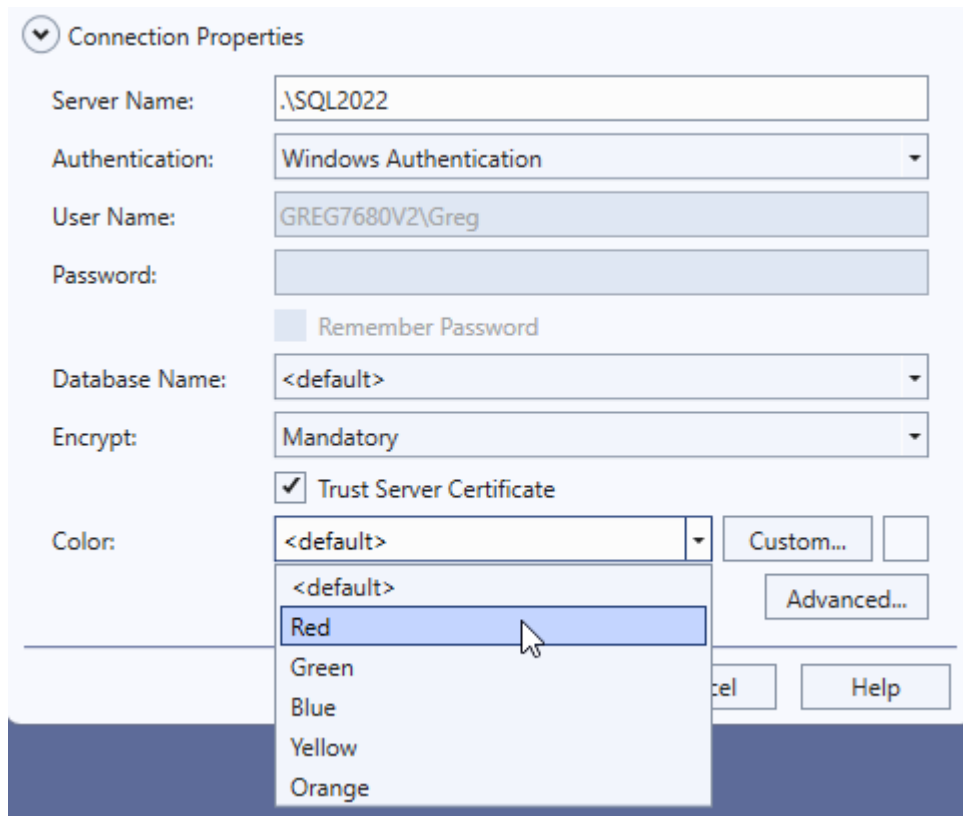


That opened a larger dialog that allowed you to specify a color for the connection:



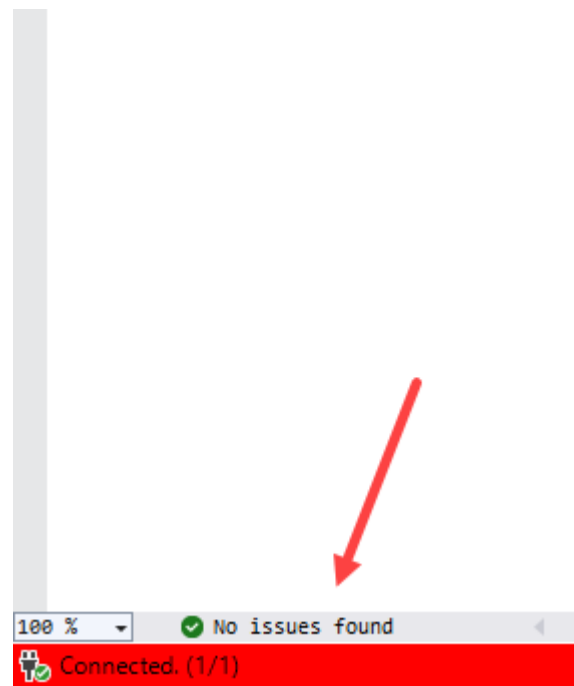


In the new connection dialog, the color option is now right on the front page for the connection:



I've chosen Red to warn me that this is a production server. You can even use the Custom button to pick a custom color.

Then, once you open a query window, you'll see the color bar at the bottom:



### 4.3 Change connection

I commonly run into a few connection-related scenarios:

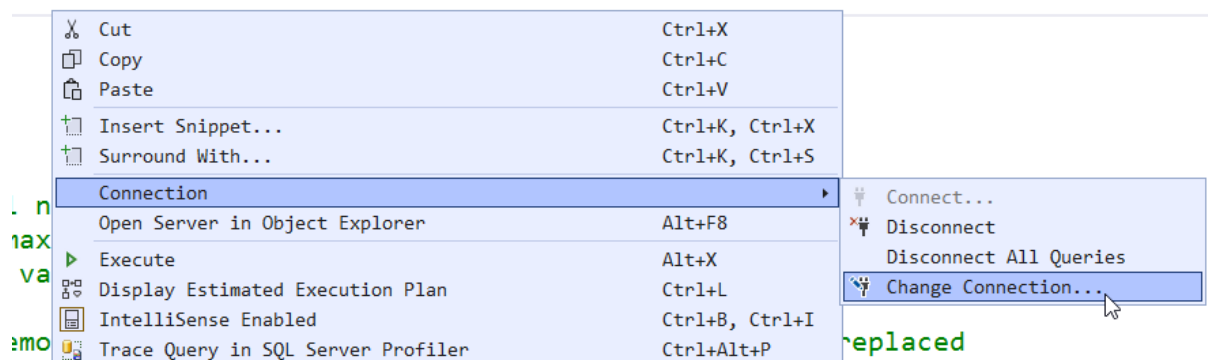
- I'm working on a large query and need to run it against several servers, not concurrently, but one after the other.
- I've just made a database connection, and got my query ready, only to discover that I've connected to the wrong server.

Either way, what I've seen people do in these scenarios is to:

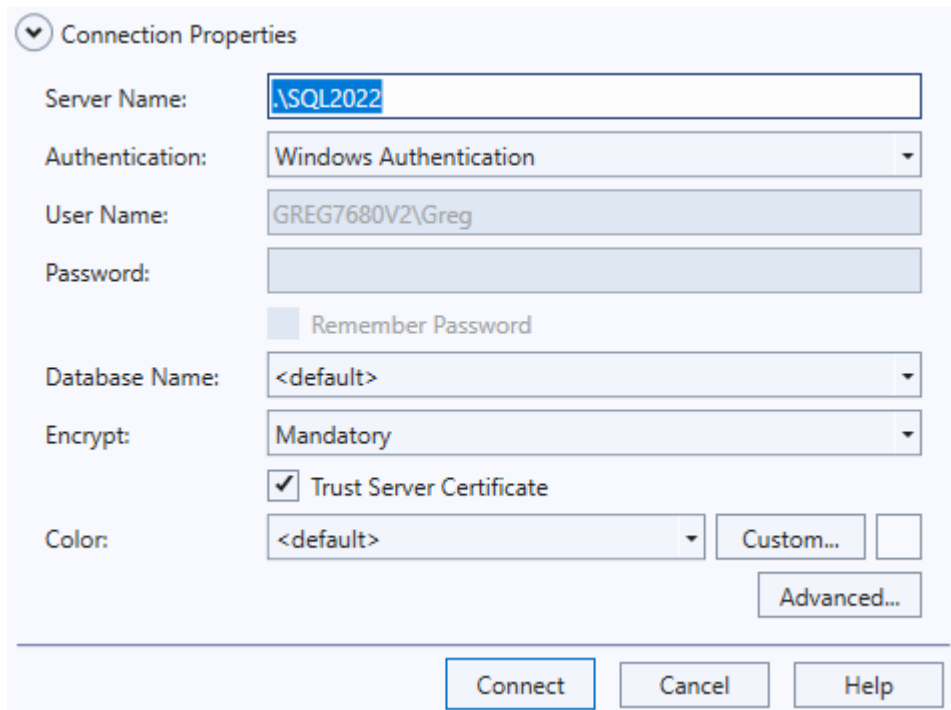
- Select all the text in the current query
- Copy it
- Open a new query window
- Paste the code

That's all good but SSMS has had a simpler way of doing this for quite a while.

If you right-click in the middle of a query window, you can choose to change the connection. Here's an example. I opened a new query window and entered a query, only to find that it connected to the Azure server that I was working with earlier. The easiest option is to right-click and choose to change the connection:



After you click **Change Connection**, you can log on again:



Connection Properties

Server Name:

Authentication:

User Name:

Password:

☐ Remember Password

Database Name:

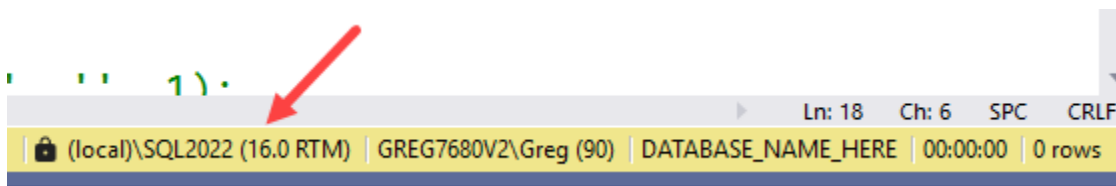
Encrypt:

☒ Trust Server Certificate

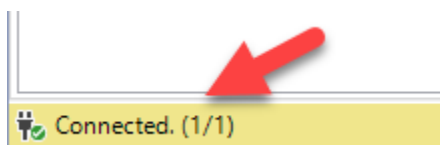
Color:

And then I can continue in the same query window as before. This is particularly useful if I need to run some code against a test server, and once I've decided that it was correct, I can just change the connection and connect to the "real" server.

At the bottom-right of your query window, you can always see which server you are connected to:



And at the bottom-left, you can see your connection state:

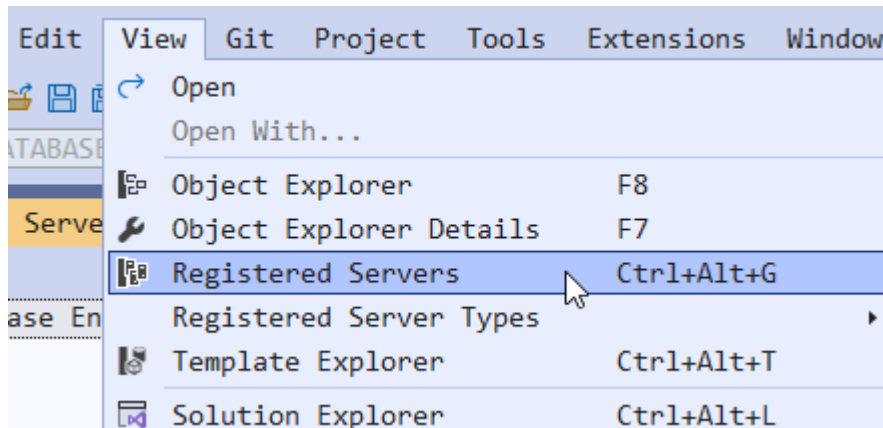


#### 4.4 Configuring registered servers

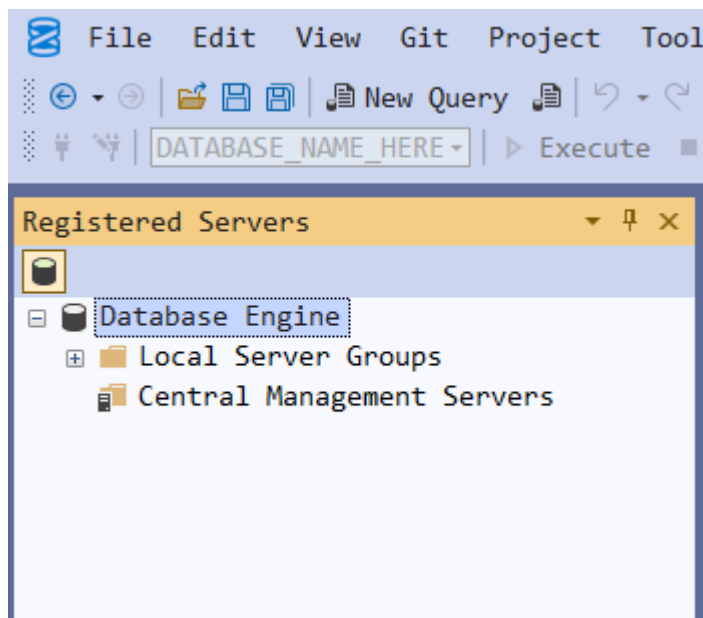
When working with SQL Server systems, it can be hard to remember the names of all the servers, to remember connection details for the ones that need SQL logins (instead of Windows authentication), and to remember other details of those servers, such as which environments they are part of (eg: production, UAT, test)

SSMS has a facility to help you to do this. It allows you to register server details in a single place.

By default, the window isn't shown, but from the **View** menu, you can choose **Registered Servers**.



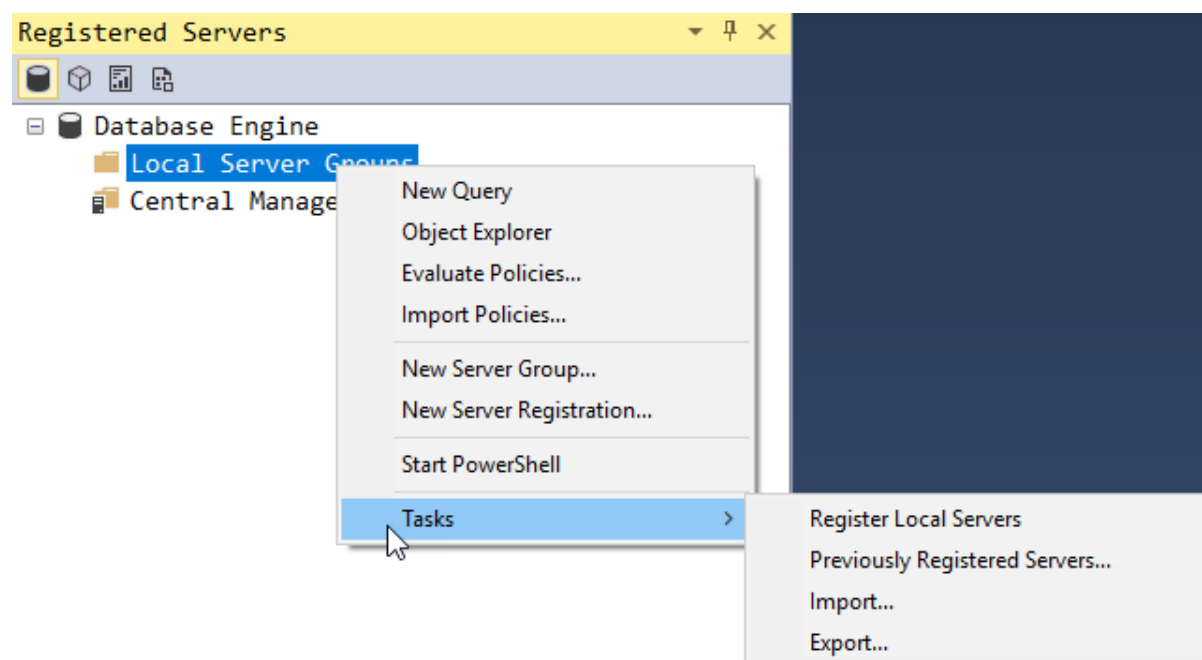
When the window opens, you can see this:



The first decision that you need to take is to decide where the details will be stored. **Local Server Groups** are stored on your local system ie: the system that is running SSMS. If you move to a different system to work, you won't have those connection details. Alternately, a Central Management Server can be configured. This is a server that agrees to hold connection details in the msdb database.

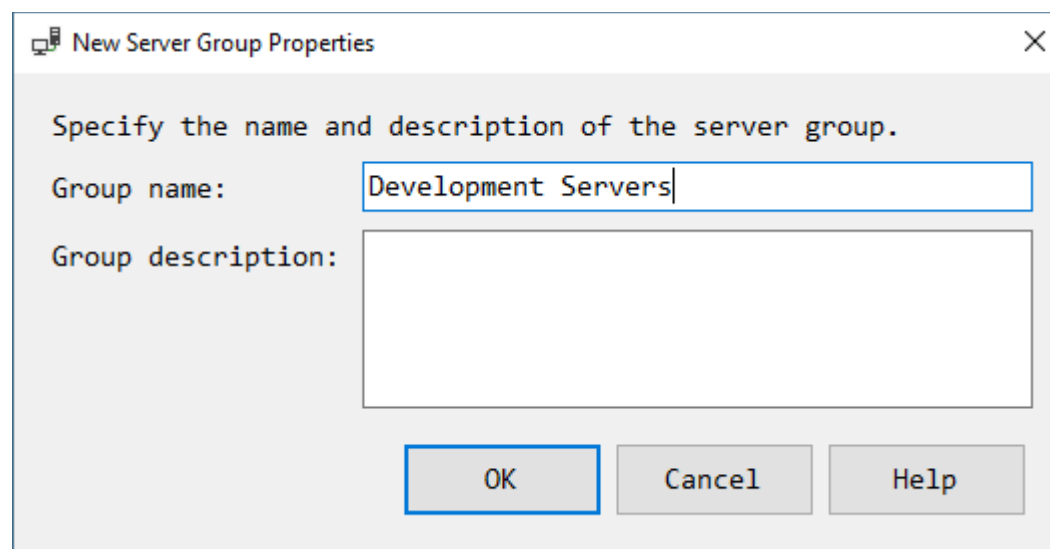
While this seems a great idea (because the details would be held in a single place), one of the downsides of this arrangement is that only Windows authentication can then be used. Local Server Groups can also work with SQL logins.

Let's create a server group as an example. If I right-click Local Server Groups, here are the available options:



Note that there is an option to Import (and Export) these details. This at least allows you to move details between systems.

Let's create a new Server Group:



It just needs a name and an optional description, then OK.

When it's created, right-click it, and choose New Server Registration:

New Server Registration

General Connection Properties Always Encrypted Additional Connection Properties

Login

Server type: Database Engine

Server name: .\SQL2022

Authentication: Windows Authentication

User name: GREG7680V2\Greg

Password:

☐ Remember password

Connection Security

Encryption: Mandatory

☒ Trust server certificate

Host name in certificate:

Registered server

You can replace the registered server name with a new name and optional server description.

Registered server name: SDUDEV

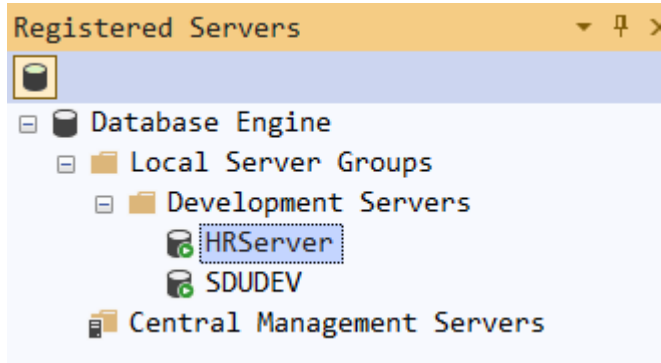
Registered server description:

Test Save Cancel Help

I've connected to the server .\SQL2022 and I've given the registered server the name SDUDEV. Note that you can call it whatever you want. I could have called it PayrollServer or some other more meaningful name. You'll also notice that there are tabs for configuring other connection properties.

I've then created a second server called HRServer and under the covers, I've pointed it to another server.

Now I have all my servers in groups, in an appropriate location. I can right-click them to open new queries to them, and to do much more.

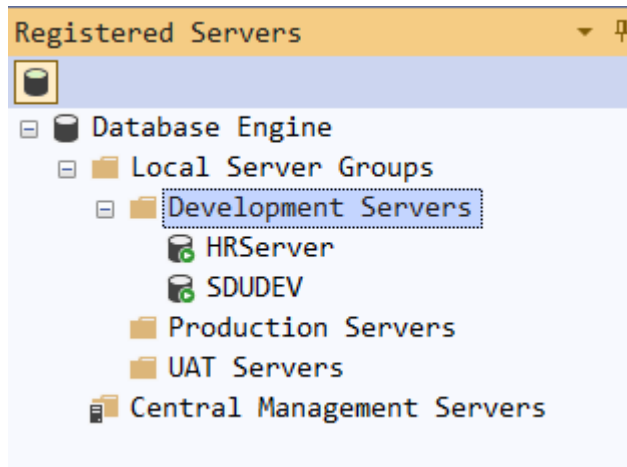


## 4.5 Multi-server Queries

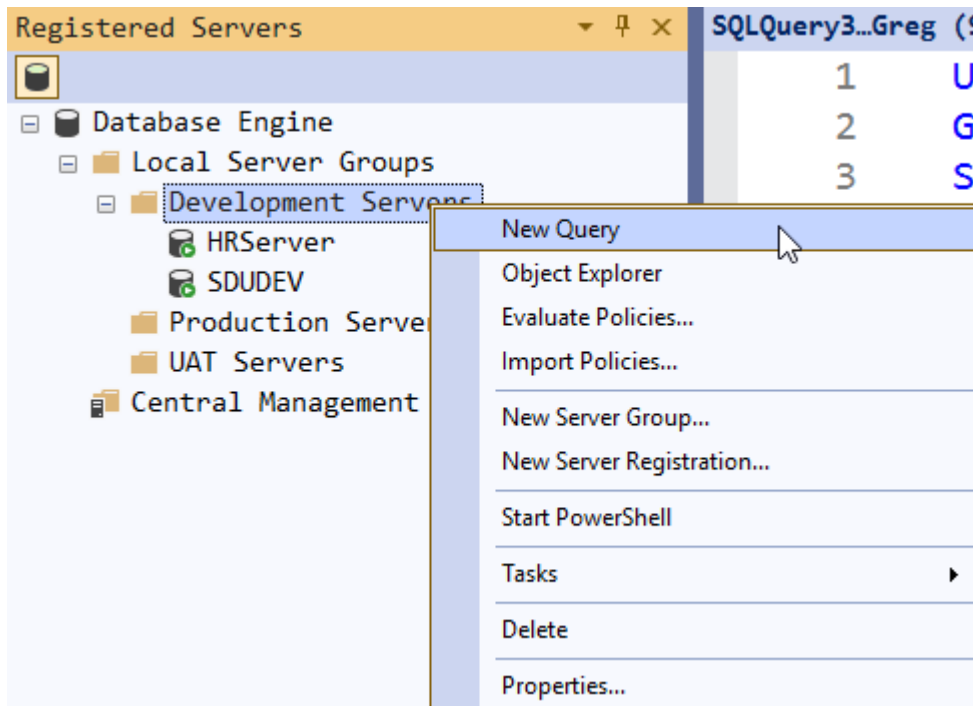
In an earlier post, I mentioned that you can create a registered list of servers, either in Local Server Groups or stored in a Central Management Server.

What I didn't really talk about though, is what you can do with these groups of servers, rather than just executing queries on an individual server.

I've created three local server groups, for my development, UAT, and production servers.



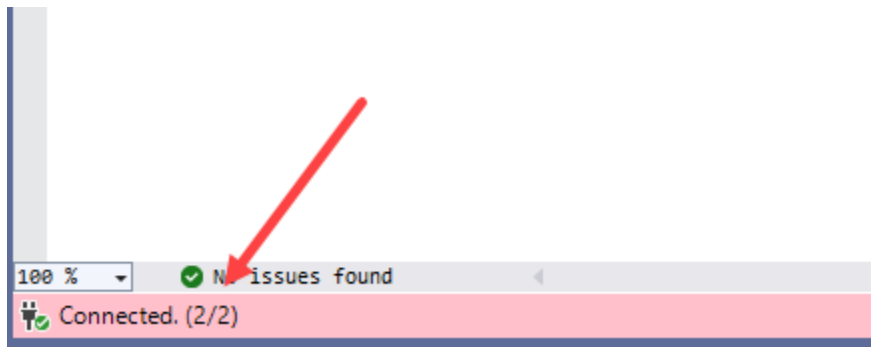
The Development Servers group has two database servers in it. If I right-click the group, rather than any individual server, we get these options:



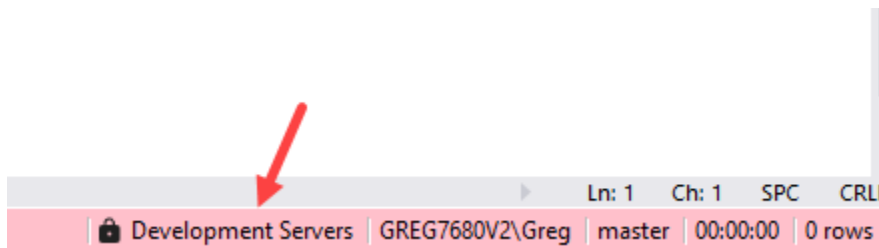
Note that you can import or check (evaluate) policies across a whole group of servers.



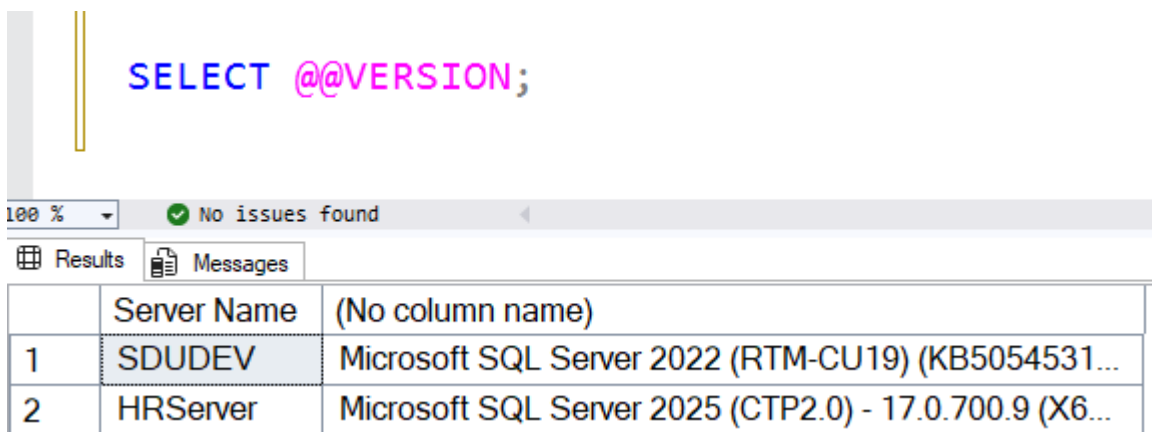
But the option that interests me today is the **New Query** option. When you click this, it opens a query window for the group of servers.



The window color at the bottom has changed from the default, and in the bottom right, we can see that the window is connected to the local server group:

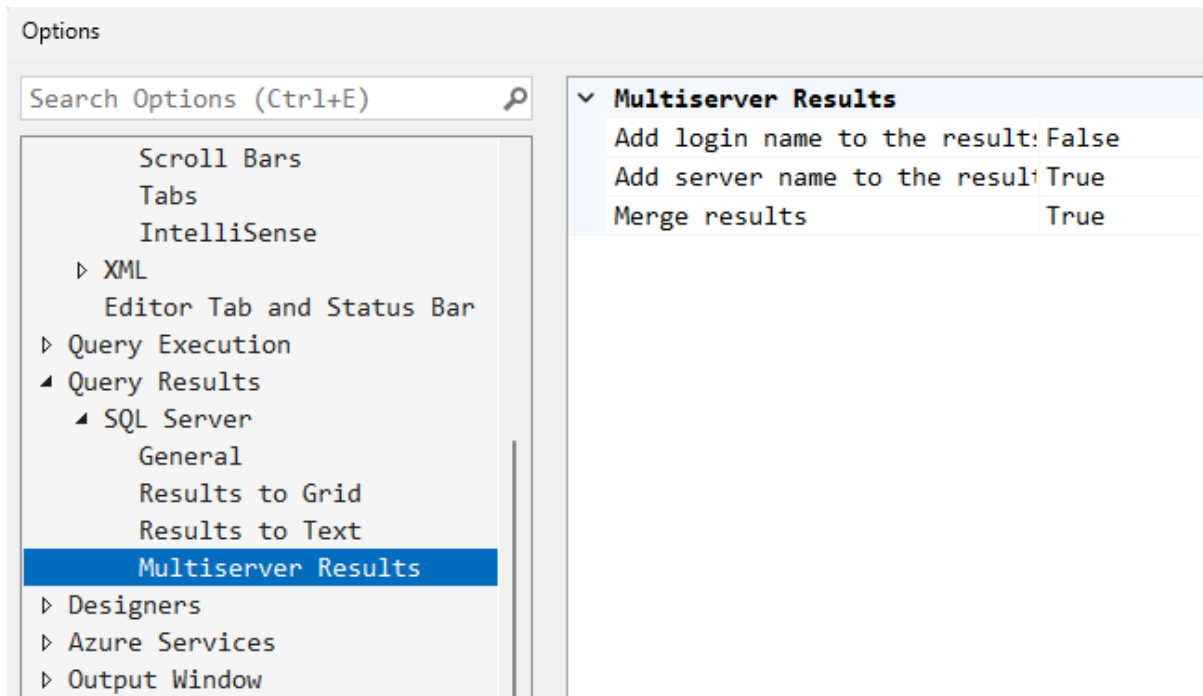


If I type the query **SELECT @@VERSION;** and click **Execute**, I see this:



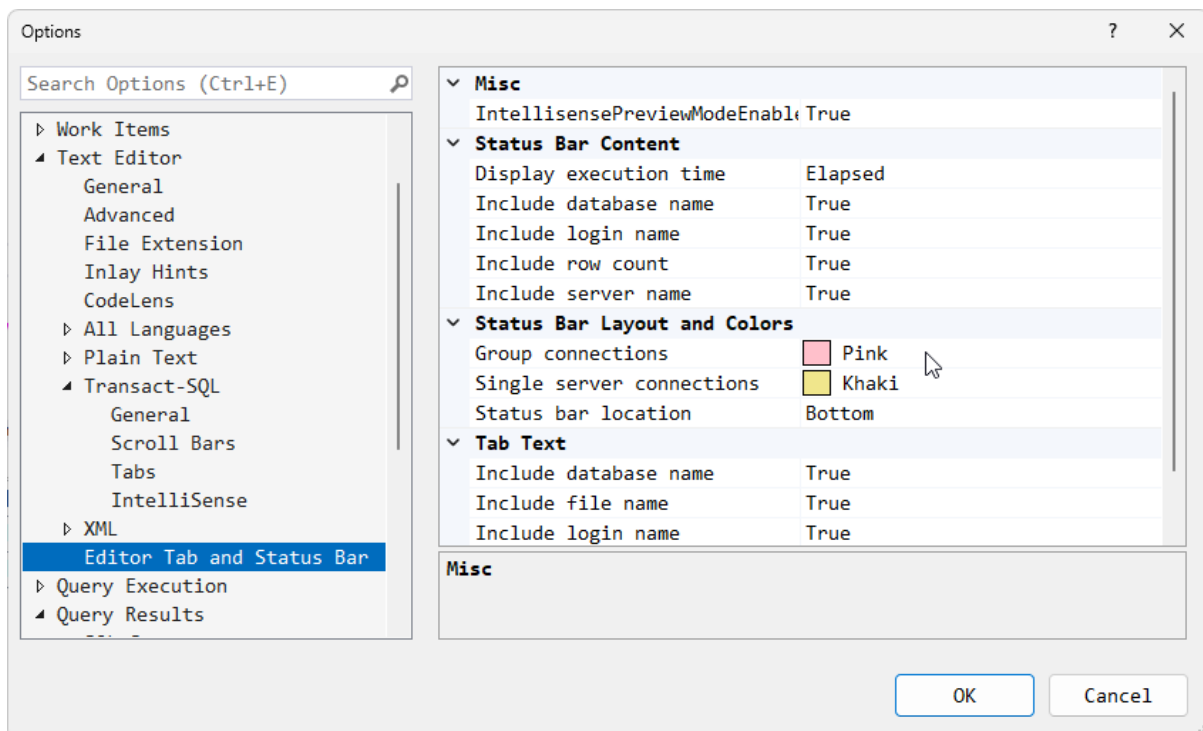
Even though this looks like a single result set, this is just an SSMS trick. Under the covers, it executed the query individually against each server. It has just presented the results to us as though they are a single set of results.

We can configure several things about how this happens. In Tools, Options, Query Results, SQL Server, Multi-server Results, we have these options:



We could add a column that shows our login name for each server to the results. We could remove the server name if required, although that doesn't seem very useful. And we could choose to not have the results merged. If we do that, SSMS returns a separate result set for each server.

You can also change the color of the bar below, in this location:



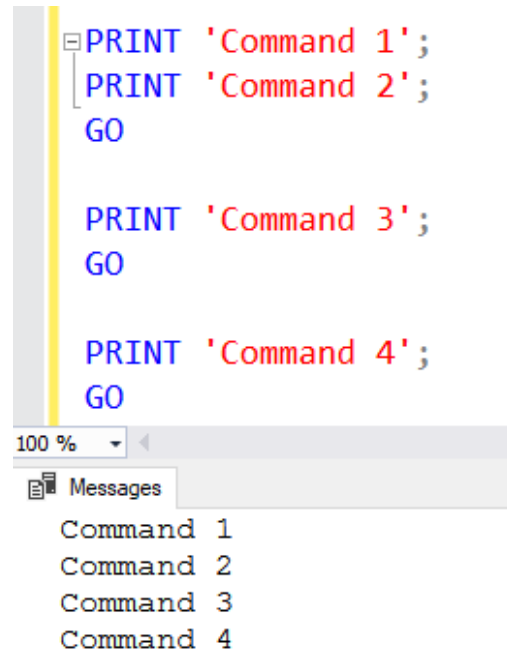
Multi-server queries were an interesting addition to SSMS. They are useful for a relatively small number of servers. As the number of servers increases, they would become more fragile, and you might want to consider using a 3<sup>rd</sup> party tool that works out which servers have or haven't had the query executed, retry options, etc.

Even though the results aren't a single result-set, it can be useful to include the server name, and to save or copy the results to Excel or another location where you will process them.

#### 4.6 Using a Count with the GO Batch Separator in T-SQL

In T-SQL, a script is a set of one or more batches.

For example, if we have the following script and click Execute, it looks like all the commands were sent to the server and executed all at once:



```
PRINT 'Command 1';
PRINT 'Command 2';
GO

PRINT 'Command 3';
GO

PRINT 'Command 4';
GO
```

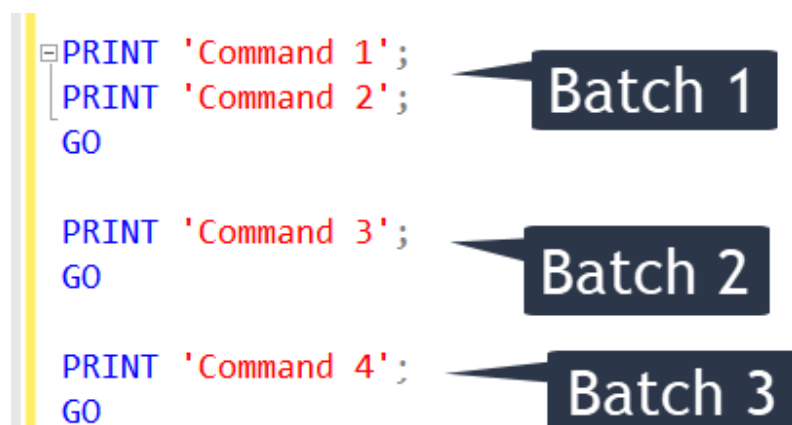
100 %

Messages

Command 1  
Command 2  
Command 3  
Command 4

But that isn't what happened.

What did happen is that SSMS found the word **GO** and broke the script into a series of batches. In this case, there were three batches. First, it sent the commands shown here as Batch 1 to the server, waited for them to execute, then sent Batch 2, waited for it to execute, then sent Batch 3, and waited for it to execute.

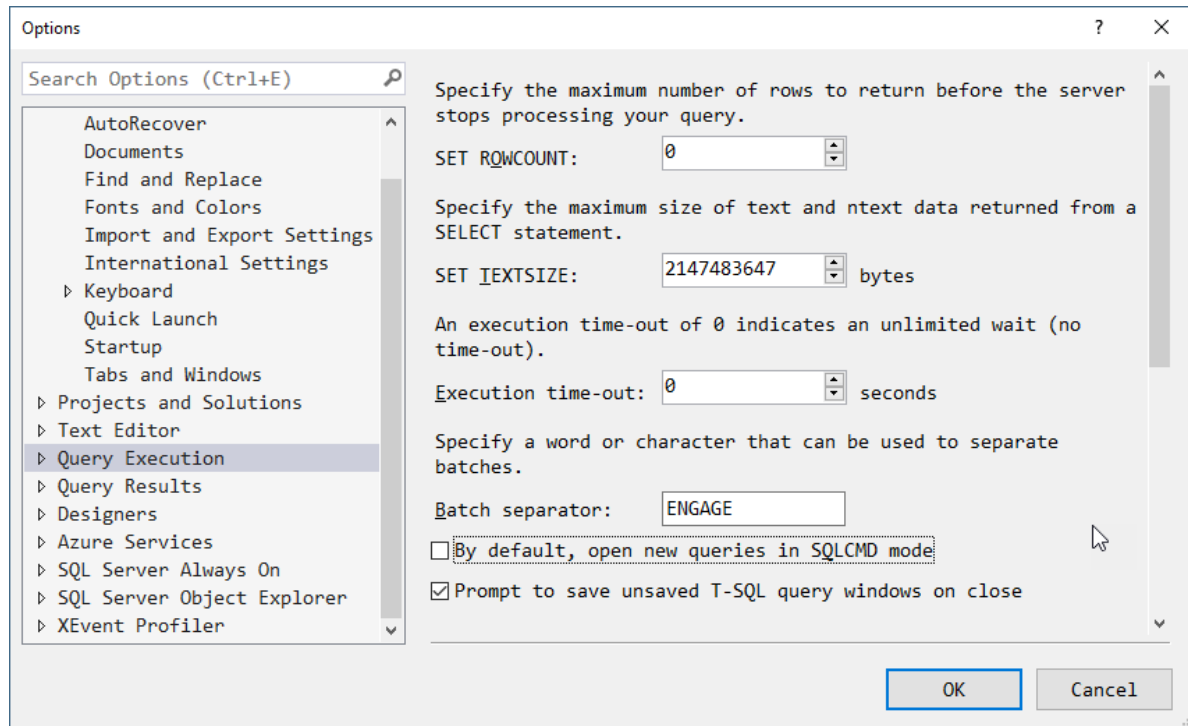


It looks like it just happened all at once, but it didn't.

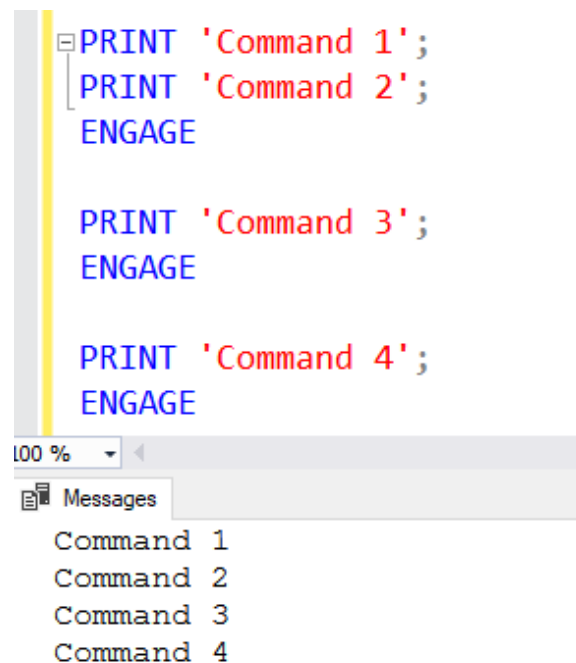
The key thing to understand about GO is that it's not a T-SQL command. It's a batch separator. We use GO to separate batches of commands that are part of a single script. The word GO is never sent to SQL Server. It only has meaning to the client tool (i.e., in this case SSMS).

In fact, you can change it. One of my friends that's a Star Trek fan has all his scripts with **ENGAGE** rather than **GO**.

You can see that I've changed it in Tools -> Options :



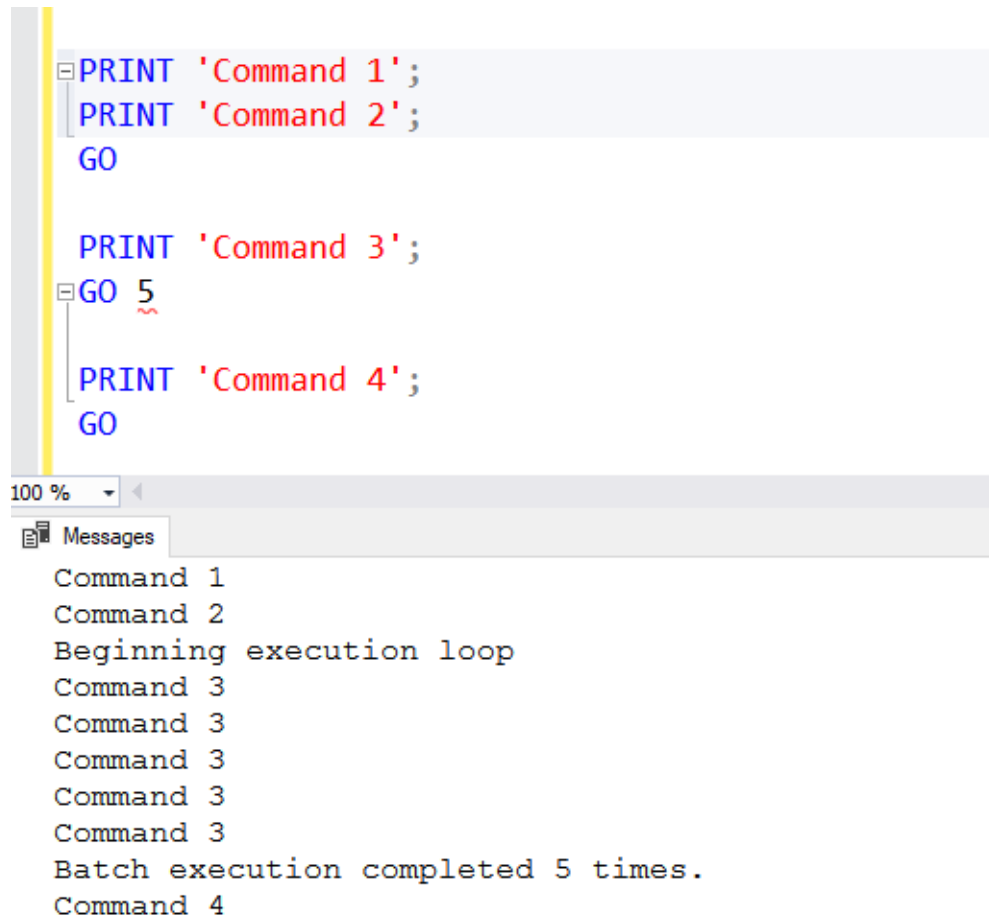
And now if I open a new query window, that works just fine:



I can't recommend doing that as your scripts won't be much use to anyone else.

So, keep in mind that it's the client that understands GO, not the server. And that leads us to our shortcut of the day.

You can add a count after the word GO, and then SSMS will send your commands from that batch to the server more than once:



```
PRINT 'Command 1';
PRINT 'Command 2';
GO

PRINT 'Command 3';
GO 5
PRINT 'Command 4';
GO
```

100 %

Messages

Command 1  
Command 2  
Beginning execution loop  
Command 3  
Command 3  
Command 3  
Command 3  
Command 3  
Batch execution completed 5 times.  
Command 4

Notice that Intellisense in SSMS doesn't understand the syntax (there is a red squiggle) but it works just fine. I have reported that and hope it will be fixed one day.

I find this shortcut quite useful if I want to execute a command a substantial number of times. An example is when I want to insert a bunch of default rows into a table; another is for executing commands while testing. I could even add a delay between each iteration by inserting a WAITFOR DELAY statement within the batch.

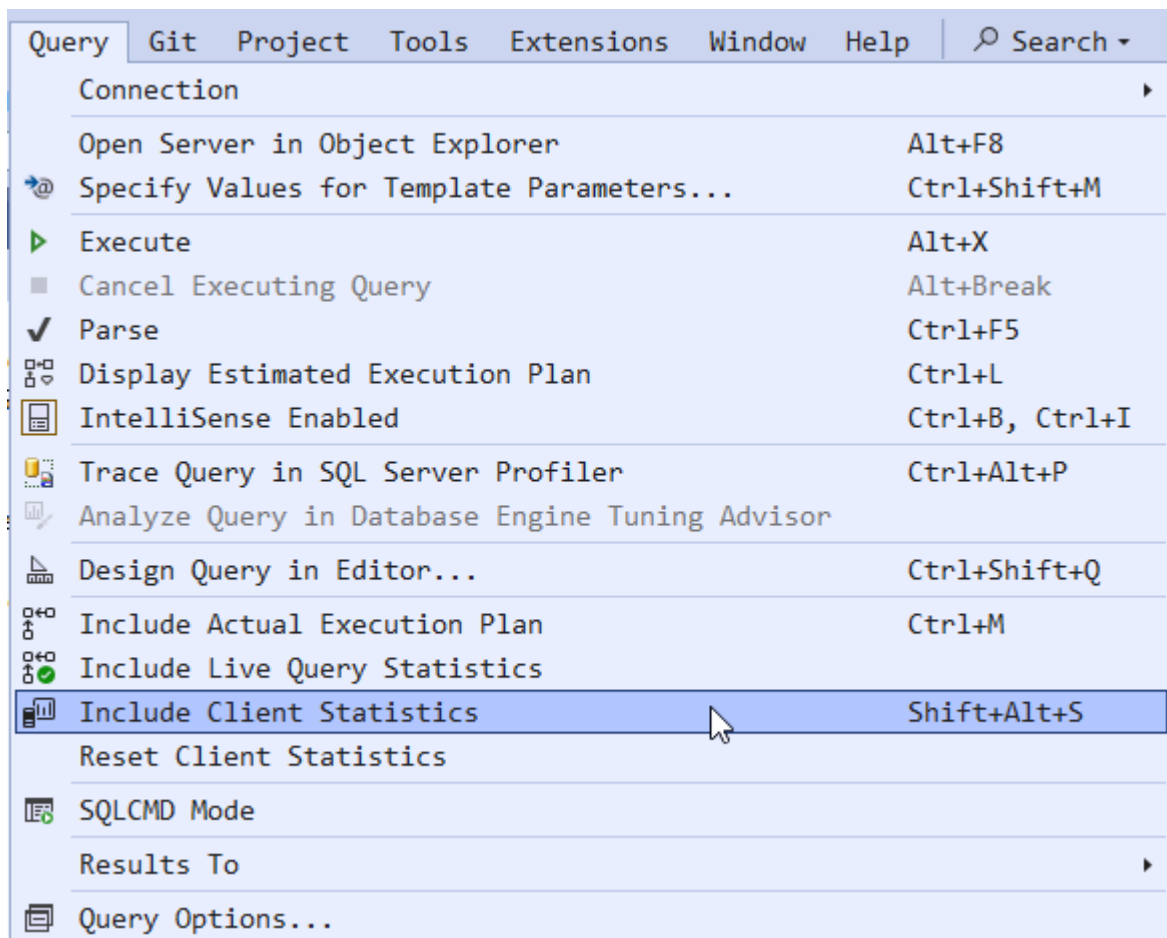
## 4.7 Viewing Client statistics

While SQL Server is quite fast at executing queries, when you are connecting from a client application like SSMS, you might wonder how much time SQL Server spent executing the query, as opposed to how long the communication with the server took.

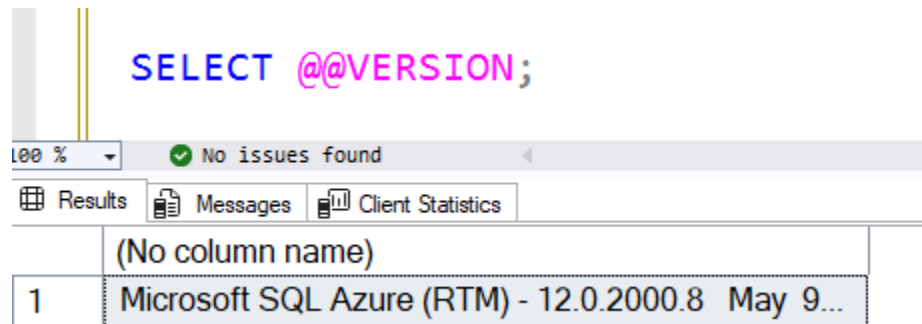
This type of information is available in the **Client Statistics**.

Let's see an example. If I connect to a server in an Azure data center, I'll have higher latency than for one in my own site. That will affect the wait time for a server response. I'll connect to a server that I have aliased as SDUAzure. The server is in the Australia South East data center.

Let's execute a simple query against it, but before doing so, on the **Query** menu, choose **Include Client Statistics**.



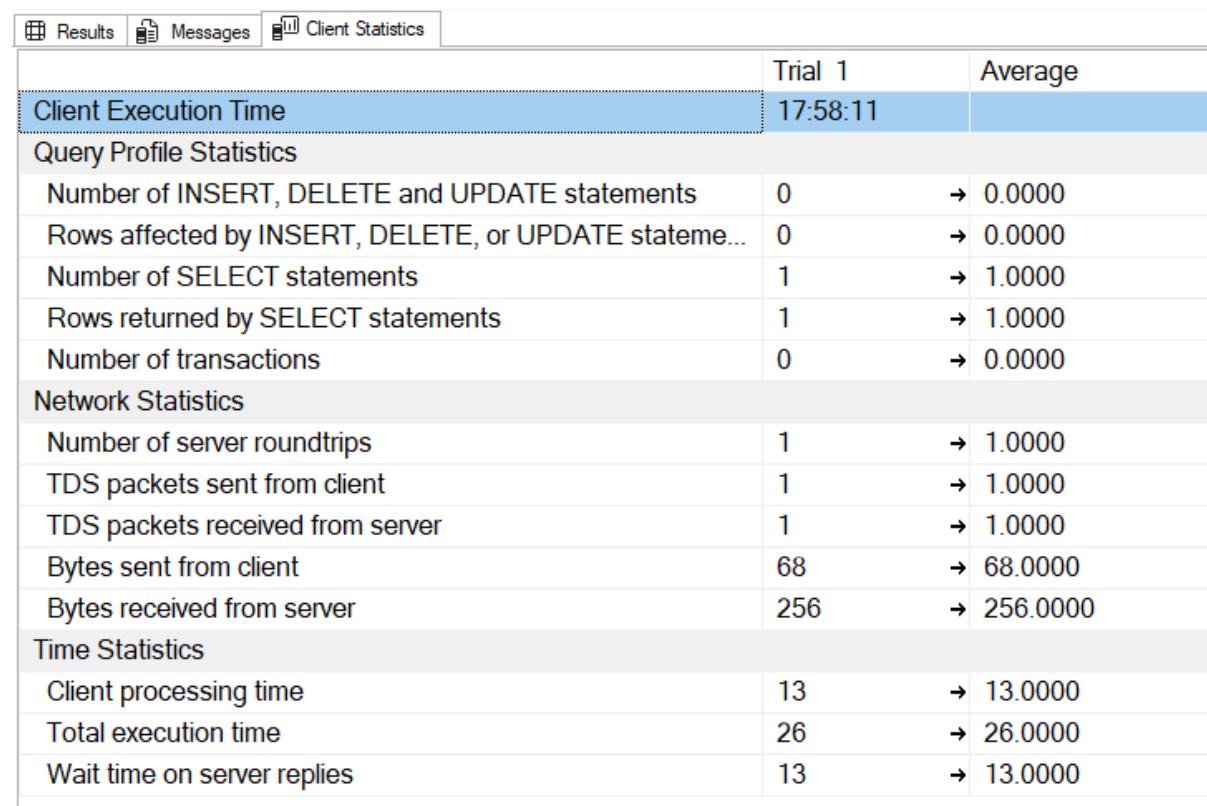
I'll just query the SQL Server version:



The screenshot shows a SQL query window with the text `SELECT @@VERSION;`. Below the query, the results pane displays a single row with the value "Microsoft SQL Azure (RTM) - 12.0.2000.8 May 9...". The interface includes tabs for Results, Messages, and Client Statistics, and a status bar indicating "No issues found".

	(No column name)
1	Microsoft SQL Azure (RTM) - 12.0.2000.8 May 9...

Notice that an extra tab of data is returned.



The screenshot shows the Client Statistics tab in SQL Server Enterprise Manager. It displays a table with three columns: Client Execution Time, Trial 1, and Average. The Client Execution Time is 17:58:11. Below this, there are sections for Query Profile Statistics, Network Statistics, and Time Statistics, each with a list of metrics and their values.

	Trial 1	Average
Client Execution Time	17:58:11	
Query Profile Statistics		
Number of INSERT, DELETE and UPDATE statements	0	→ 0.0000
Rows affected by INSERT, DELETE, or UPDATE statements	0	→ 0.0000
Number of SELECT statements	1	→ 1.0000
Rows returned by SELECT statements	1	→ 1.0000
Number of transactions	0	→ 0.0000
Network Statistics		
Number of server roundtrips	1	→ 1.0000
TDS packets sent from client	1	→ 1.0000
TDS packets received from server	1	→ 1.0000
Bytes sent from client	68	→ 68.0000
Bytes received from server	256	→ 256.0000
Time Statistics		
Client processing time	13	→ 13.0000
Total execution time	26	→ 26.0000
Wait time on server replies	13	→ 13.0000

From the bottom section of this tab, we can see where the time was spent. In this case, out of a total of 26 milliseconds for the query, 13 milliseconds was spent waiting for the server.



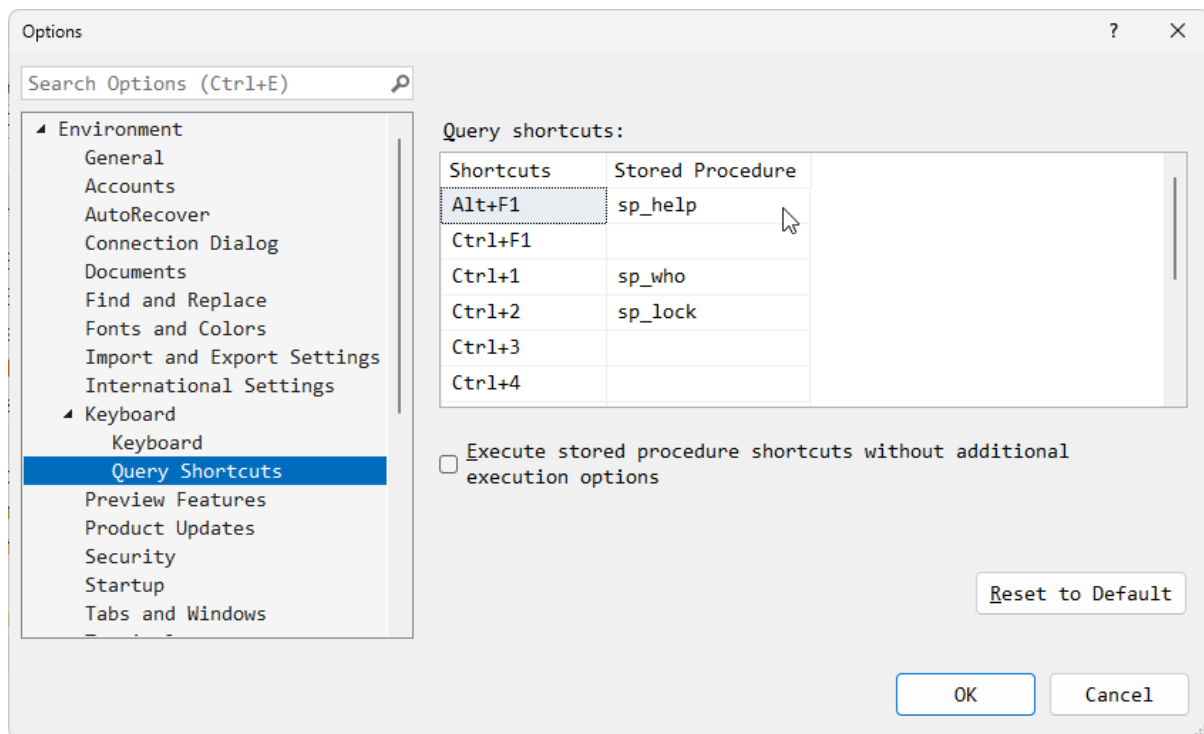
#### 4.8 Set shortcuts for favorite stored procedures

In an earlier entry, I mentioned how useful the F1 key is. On its own, it provides syntax help, but when you highlight an object and hit Alt-F1, you get to see metadata about the object.

Under the covers, this just runs the **sp\_help** system stored procedure. Alt-F1 has been mapped to that.

You can see where this is configured, change it if required, and/or configure other procedures as well.

In **Tools, Options, Environment, Keyboard**, then **Query Shortcuts**, you can see this:



Here we can see how the Alt-F1 command is configured. By default, only three shortcuts are configured. It's easy to configure another one for your own use.

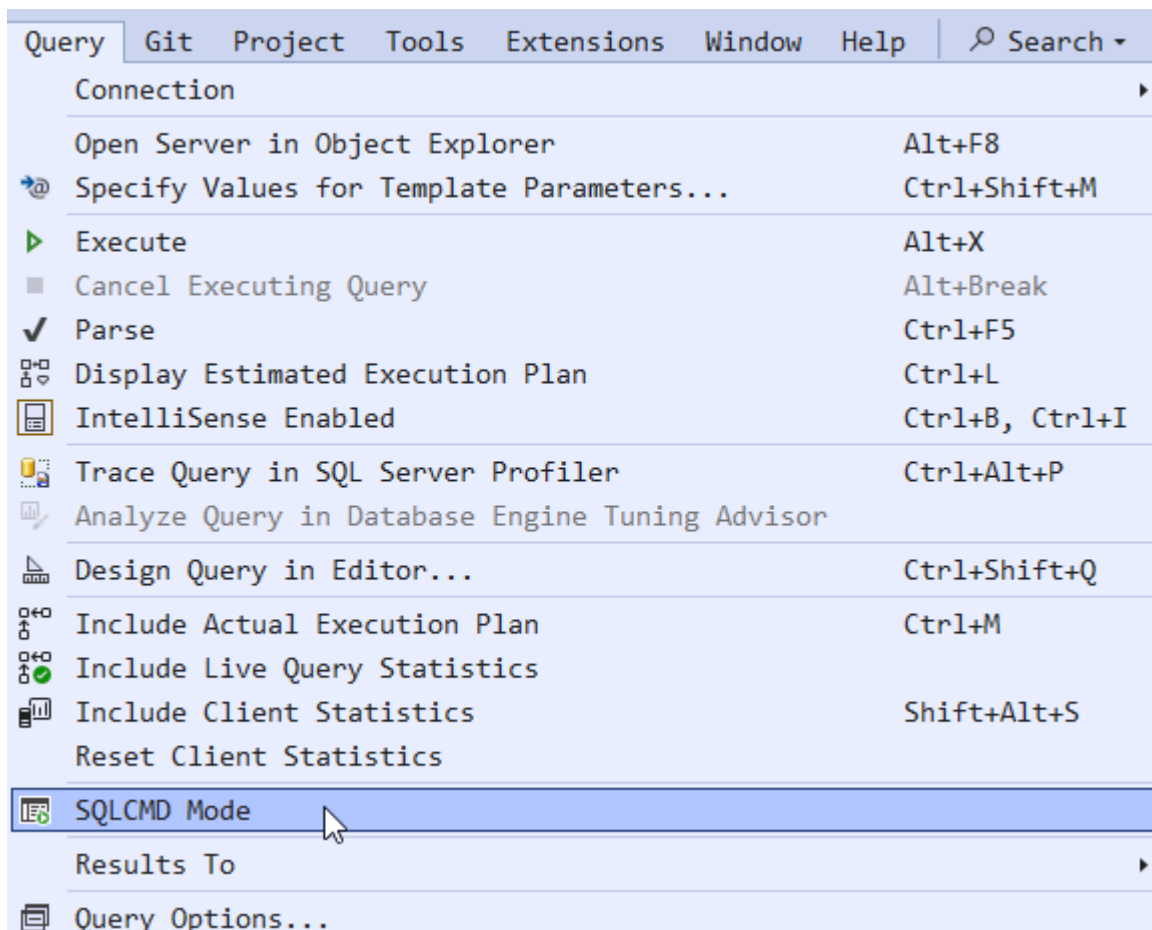
#### 4.9 Set SQLCMD mode for all new windows

SQLCMD mode changes how queries are executed in SSMS. When using this mode, you can work with options that aren't normally part of SQL Server T-SQL scripts.

Some installation scripts also require SQLCMD mode and will fail if it's not enabled.

Let's look at an example executing a query against 2 servers within the same script.

First, we open a new query window, then on the Query menu, we choose SQLCMD Mode:



Then I execute the following query:

```
:CONNECT .\SQL2022

SELECT @@SERVERNAME;
GO

:CONNECT .\SQL2025

SELECT @@SERVERNAME;
GO
```

00 % No issues found

Results Messages

	(No column name)
1	GREG7680V2\SQL2022

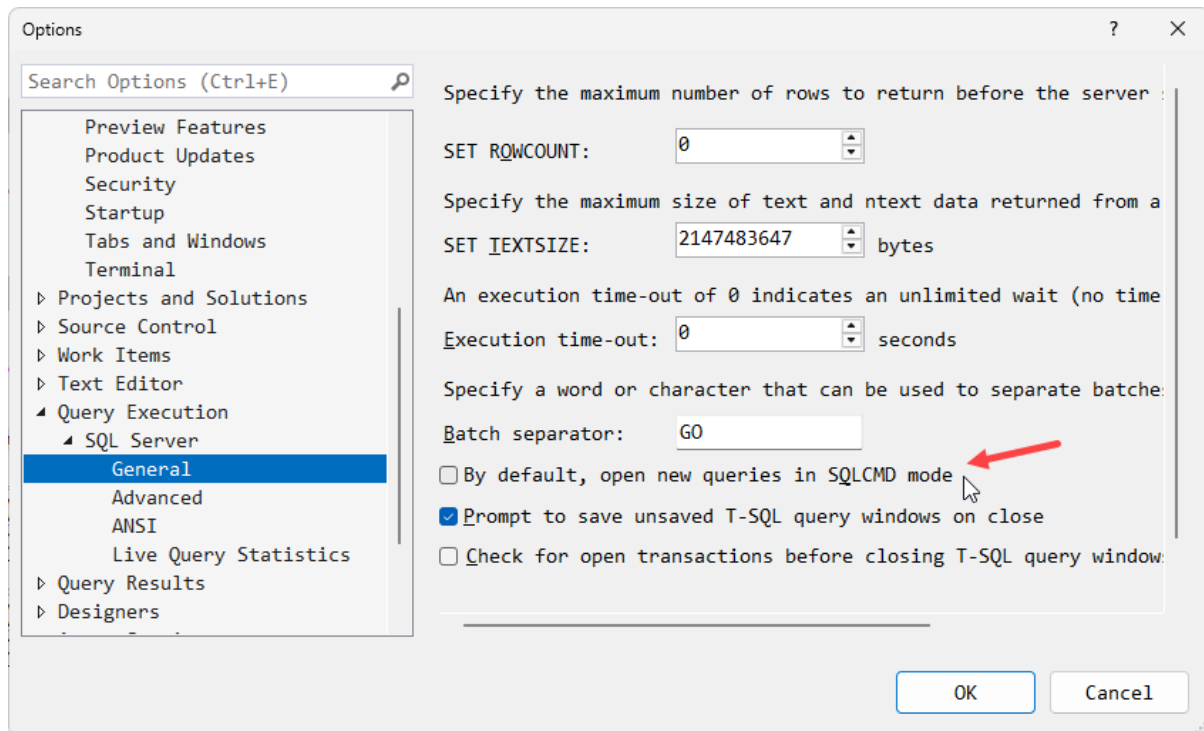
  

	(No column name)
1	GREG7680V2\SQL2025

Note the **:CONNECT** command is used to connect to another server.

Because everything else works pretty much the same, and you get a whole lot of additional options, you might choose to open all your new queries in SQLCMD mode. That's easy to do.

In Tools, Options, Query Execution, SQL Server, then General, there is a checkbox to enable this.

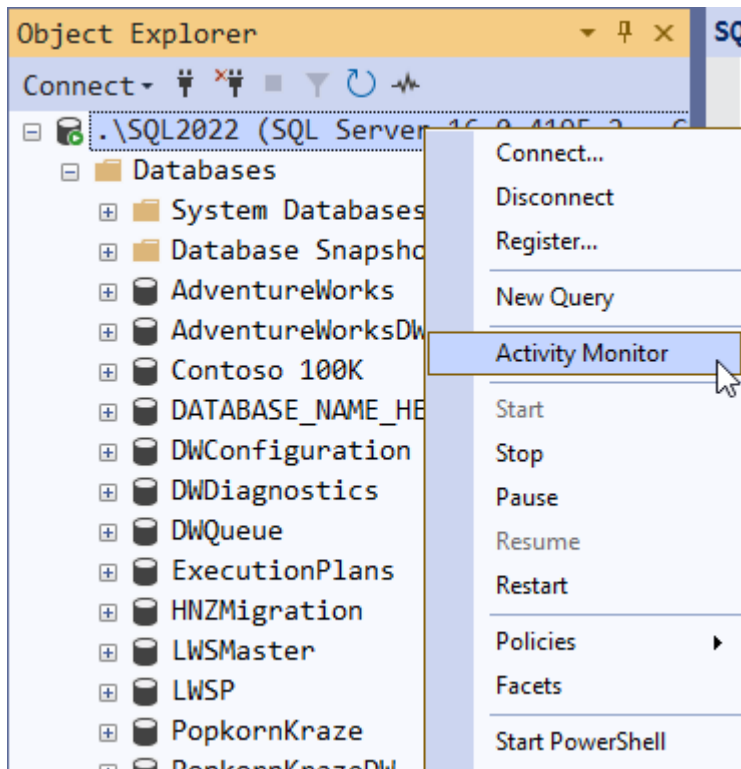


## 4.10 Activity Monitor

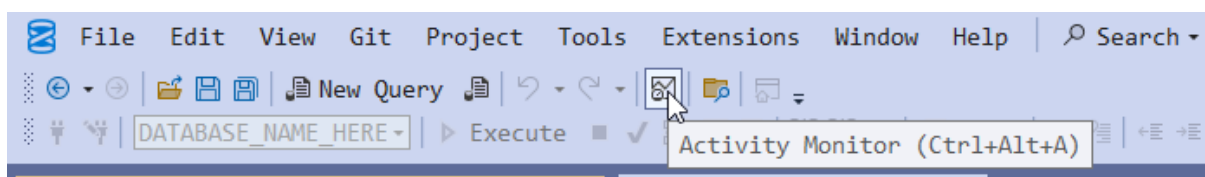
This is a quick tip but an important one. I see many people using SSMS and they aren't aware of Activity Monitor.

While there are many clever things that we can do with queries, to interrogate the health of the system, don't forget that there is quite a bit of useful information in Activity Monitor, and it's easy to get to.

There are two basic ways to launch Activity Monitor. The first is to right-click the server in Object Explorer:



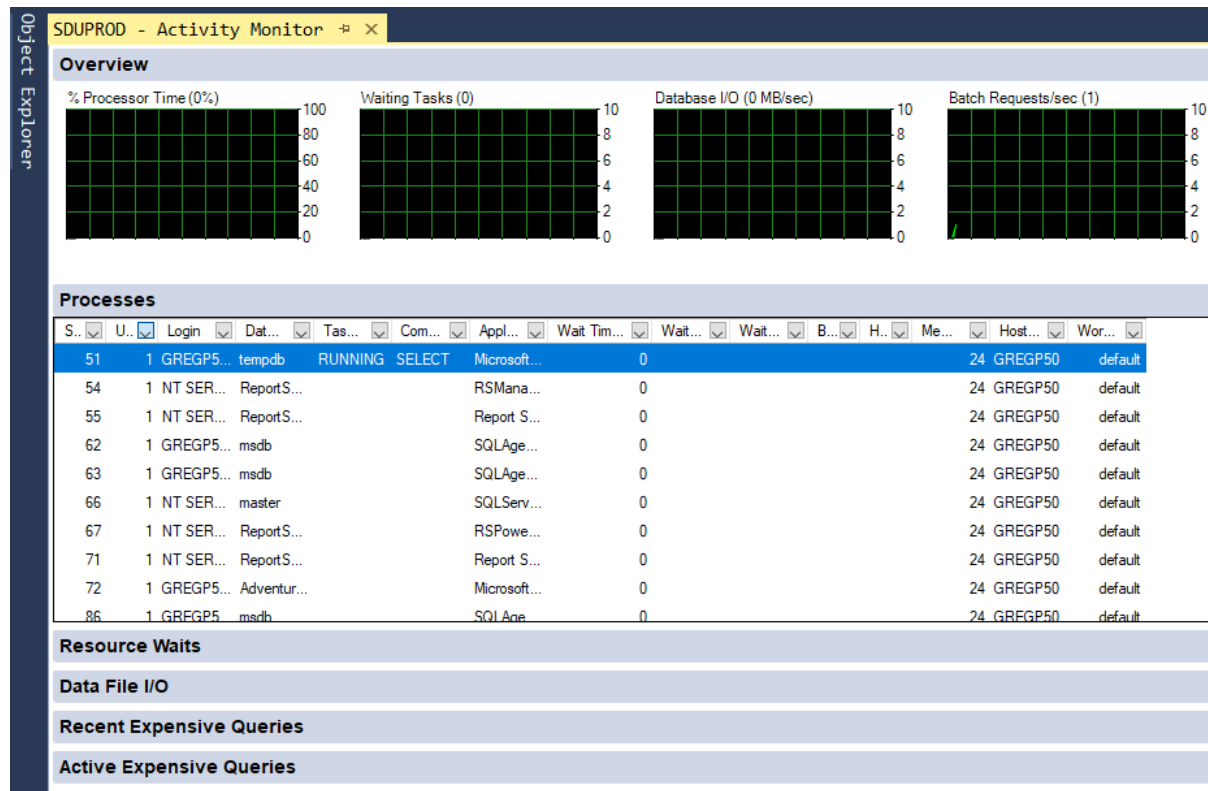
The other common way to launch it is from the Toolbar:



Note that if you connect to more than one server in Object Explorer, Activity Monitor will connect to whichever one you have selected any object from.

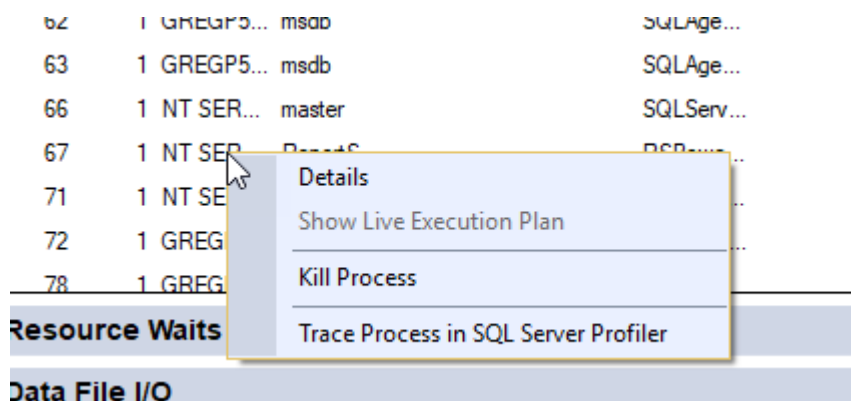
**Activity Monitor puts a bit of a load on the server that it's connected to but I generally don't find it too bad. However, please don't leave it running and go on using other tabs. I've been to sites where there are many copies of it running all the time from several users. Don't do that.**

I don't find most of the graphs at the top very useful, apart from perhaps the processor time.



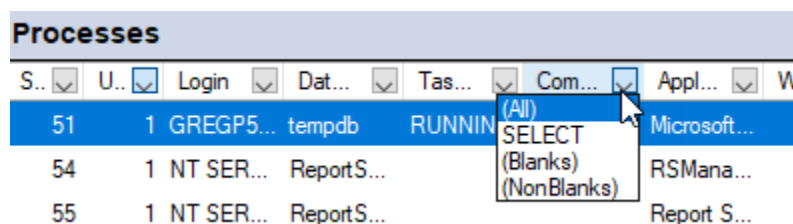
It will show you if the server is running flat out.

The list of Processes is more interesting. If you right-click any session, you get these options:



The Details link will show you the last command executed on that connection. Take note that this doesn't mean it's still running. You can also kill the process (obviously carefully), and you can connect SQL Server Profiler to the server and filter the session immediately, to see what it's doing.

The columns are filterable.



They show you a list of values currently in that column, plus an All, and a choice of Blanks (rows with no value in this column) or NonBlanks (rows with anything in this column). They start as All.

For a simple example of using this though, we could pick sessions that have any type of command running, by choosing Task State of RUNNING.

Processes									
S...	U...	Login	Dat...	Task State	Com...	App...	Wait Tim...	Wait...	Wait...
51	1	GREGP5...	tempdb	RUNNING	SELECT	Microsoft...	0		
								24	GREGP50
									default

One that I often use this view for is to look for blocking issues. Every process that's blocked by another process will tell you that. Generally, what I'm looking for is the head of a blocking chain ie: who's the main culprit that's blocking everyone.

For that, I look for a value of 1 in the Head Blocker column. Unfortunately, the way it's designed, you can't select that value until there is a row with that value.

The Application Name, Database Name, and Login can all be pretty useful as well.

The Resource Waits section is only mildly interesting.

Processes					
Resource Waits					
Wait Category	Wait Time (ms/sec)	Recent Wait Time (ms/sec)	Average Waiter Count	Cumulative Wait Time (sec)	
Buffer I/O	0	0	0.0	3	
Buffer Latch	0	0	0.0	0	
Latch	0	0	0.0	0	
Lock	0	0	0.0	24	
Logging	0	0	0.0	24	
Memory	0	0	0.0	0	
Network I/O	0	0	0.0	0	
Other	0	0	0.0	0	

The information there is at a bit of a coarse level to be really useful to me. Note that on this system, Buffer I/O is top of the list, but the cumulative wait time (since the server restarted) is small. Over time, if the system has been up for a long time, you can start to get a feel for the main waits in here, but be aware that there are a lot of values that can appear in here, without actually being an issue.

The Data File I/O list is a little more interesting:

Data File I/O					
Database	File Name	MB/sec Read	MB/sec Written	Response Time (ms)	
tempdb	C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MS...	0.0	0.0	8	
AdventureWorks	C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MS...	0.0	0.0	0	
AdventureWorks	C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MS...	0.0	0.0	0	
AdventureWorks...	C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MS...	0.0	0.0	0	
AdventureWorks...	C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MS...	0.0	0.0	0	
AdventureWorks...	C:\Temp\AdventureWorksDW2016_Data.mdf	0.0	0.0	0	
AdventureWorks...	C:\Temp\AdventureWorksDW2016_Log.ldf	0.0	0.0	0	
DATABASE_NA...	C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MS...	0.0	0.0	0	
DATABASE_NA...	C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MS...	0.0	0.0	0	

This will show you how busy each data and log file is, for all databases. I generally sort it by Response Time (ms) descending. The value here is then basically the latency for the I/O on that file. In this example, it's 8 milliseconds. That's ok.

The Recent Expensive Queries list is interesting. The information is available from the system DMVs but this puts some useful data in an easy-to-get location:

Recent Expensive Queries									
Query	Executio...	CPU (ms...	Physical ...	Logical ...	Logical ...	Average...	Plan Co...	Dat...	
delete top(20) s_output deleted_SessionID, dele...		0	0	0	0	0	0	1	ReportS...
update [ReportServerTempDB].dbo.SnapshotD...		0	0	0	0	0	0	1	ReportS...
delete top (@PermanentChunkCount) SC o...		0	0	0	0	0	0	1	ReportS...
SELECT TOP 1 @previous_collection_time = c...		6	0	0	0	0	0	1	tempdb
Update [Notifications] set [ProcessStart] = NUL...		0	0	0	0	0	0	1	ReportS...
DELETE FROM #am_resource_mon_snap WH...		12	0	0	0	0	0	1	tempdb
select top 8 -- Notificatio...		2	0	0	0	0	0	1	ReportS...
delete top(@TemporaryMappingCount) CSM ...		0	0	0	0	0	0	1	ReportS...
DELETE [Policies]WHERE [PoliciesID]Poli...		0	0	0	0	0	0	1	ReportS...
UPDATE SNSET_PermanentRefcou		0	0	0	0	0	0	1	ReportS...
Active Expensive Queries									

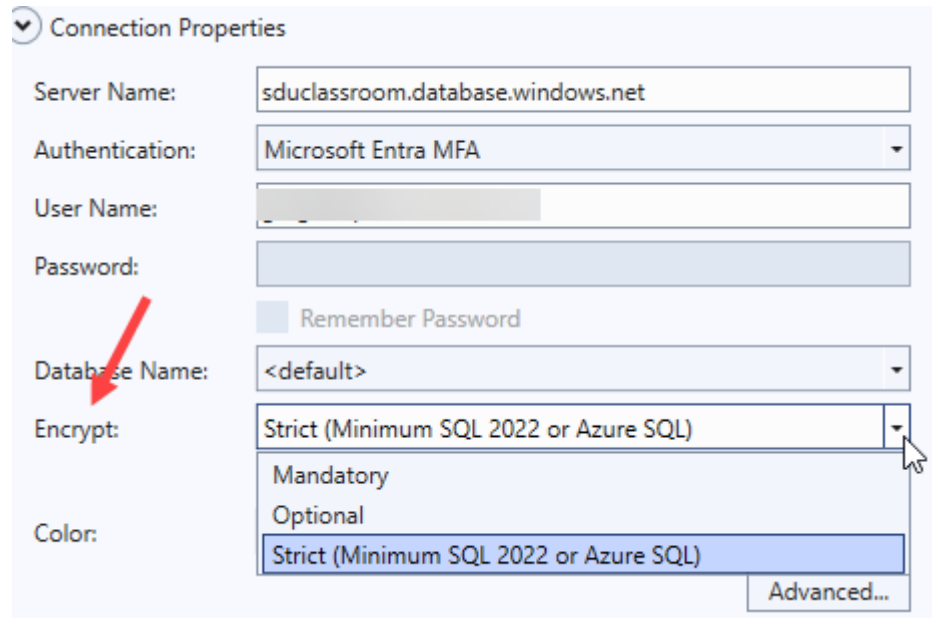
It keeps updating this over time. Note that this won't be showing you queries currently running, just ones that were expensive and finished recently. If you right-click one, you can either look at the query text or check out the execution plan that was being used.

The final section with Active Expensive Queries will only have data if you're using Live Query Statistics.



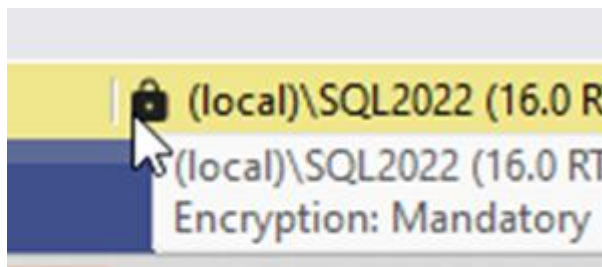
## 4.11 Encryption Status in Query Status Bar

One of the bigger changes in recent versions of SSMS has been the inclusion of connection encryption details on the main connection page:



Previously, this was on another page of the logon dialog.

Once you make the connection though, there previously was no indication of what type of encryption was used. Now that appears in the query status bar as a lock. When you hover over the lock, it shows you the type of encryption that had been selected.



## 4.12 Avoiding deadlocks when working interactively

Several times, I've seen situations where a user who's working interactively in SSMS ends up causing deadlocks and causing issues for an application that's in use. This is even more likely for users who hold locks for long periods of time, and who work directly with production systems.

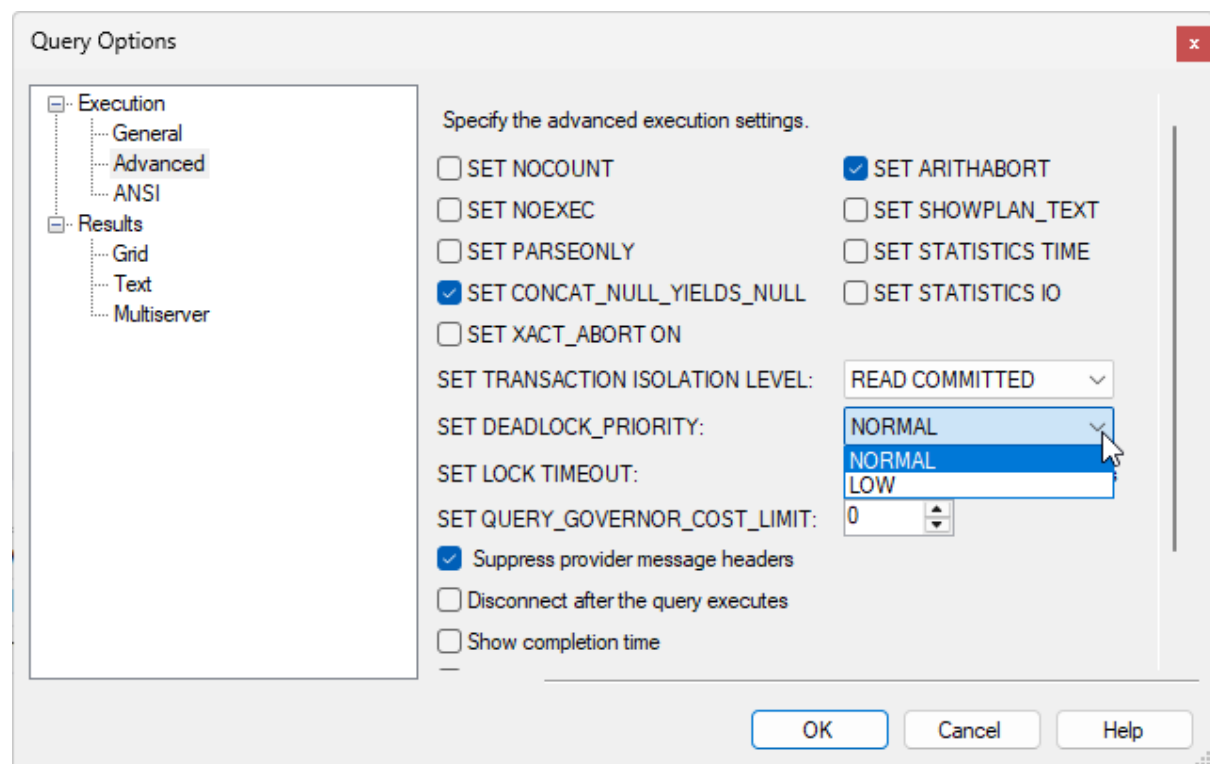
At best, they might just cause an application to hang. At worst, they might cause a poorly-designed application to terminate.

Why would a user hold locks for a long time? Many users work in what's called **chained mode**, where they automatically start a transaction when they make any sort of update. Some other database engines default to that behavior, but you can choose that as a session option in SQL Server as well.

Another class of user that might do this is the **nervous administrator** who manually starts a transaction before doing any sort of work, because they're not confident of the outcome, and want to be able to roll the actions back.

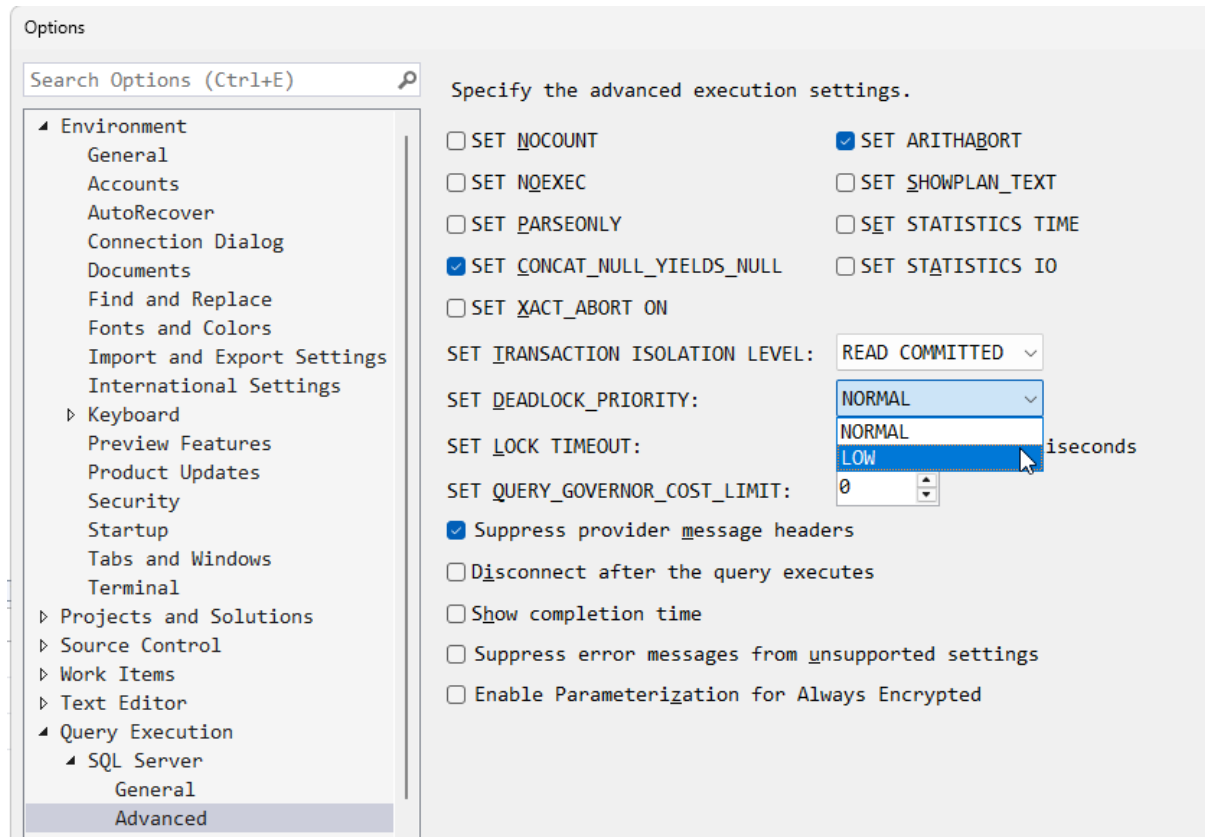
Regardless of why they're holding locks for long periods of time, the problem is compounded if they start causing deadlocks as well.

One option that's worth considering is setting the deadlock priority for your session to LOW:



If you do that, you're saying **If a deadlock is going to happen, I'm willing to be the victim**. At least then you'll get your query terminated instead of the application that's also using the database.

If you wanted to make this the default for all query windows that you open, you can set it in Tools Options:

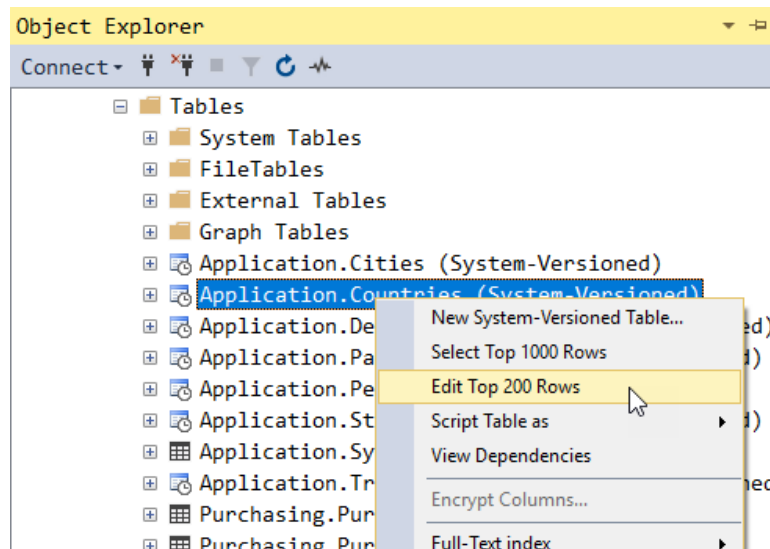


Another option you might consider is adding it to your default query template, as discussed in section 7.1. Then you get to see it in the query window and decide if you want it or not for the query session.

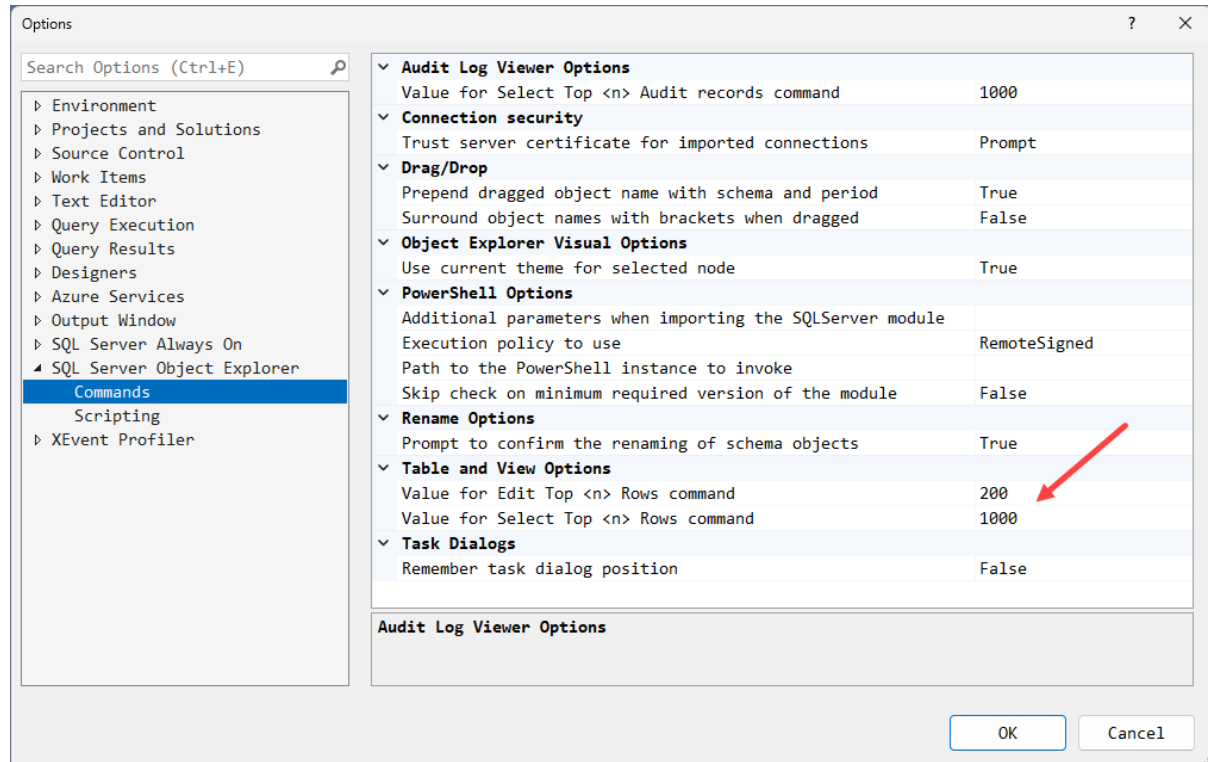
## 5 Results

### 5.1 Changing the number of rows selected or edited in Object Explorer

When you right-click a table in SQL Server Management Studio, you get options for selecting or editing but the number of rows is limited:



Those values can be changed. By default, these numbers are both 200, but I've decided to change the default number of rows selected to 1000. In Tools, Options, SQL Server Object Explorer, then Commands, you can set the values to whatever suits you:



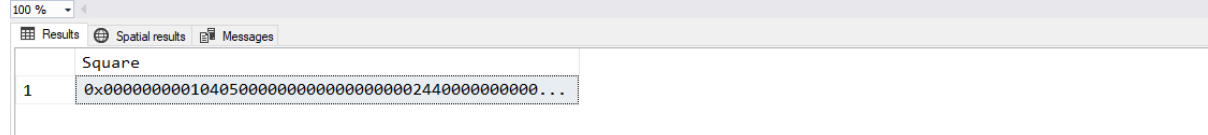
I don't tend to ever use the Edit option but I'd suggest not making it too large.

## 5.2 Viewing and configuring spatial data output

SQL Server 2008 added the ability to work with spatial data by the additional of the geometry and geography data types. When they first were added, there was no tools support for working with them, and all we had was direct manipulation of their internal binary storage.

Here's an example:

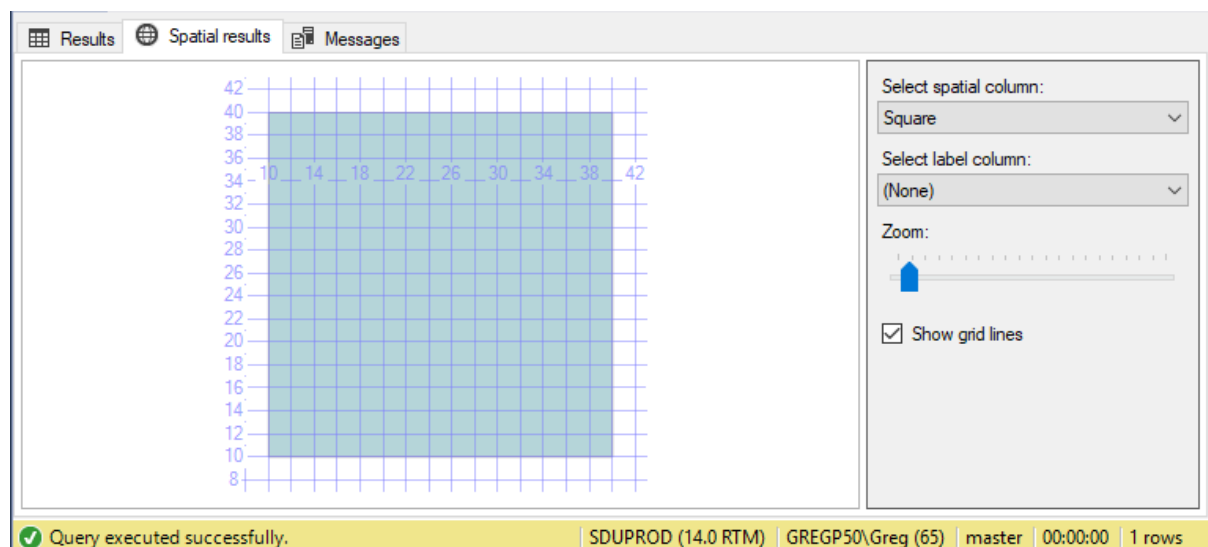
```
-- Draw a square
DECLARE @Shape GEOMETRY;
SET @Shape = GEOMETRY::STGeomFromText('POLYGON ((10 10, 10 40, 40 40, 40 10, 10 10))',0);
SELECT @Shape AS Square;
GO
```



I've defined a variable named @Shape of type GEOMETRY. I've then assigned a shape to it, based on a polygon formed by a set of points. If you look carefully, you'll notice that it's a square.

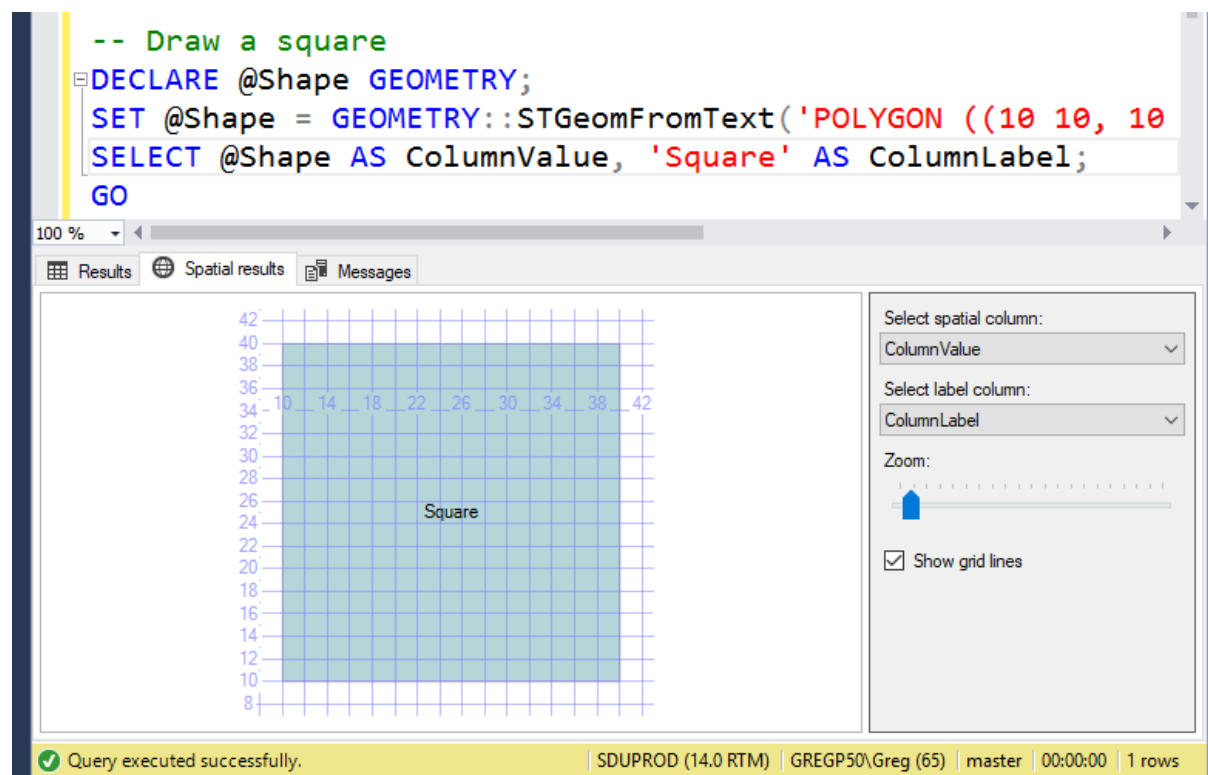
But when I select the value, what is returned is the internal value of the data type, as a binary string, **written in hexadecimal for ease of reading** 😊

And at first, that's all we had. But notice that there is now another results tab:



As soon as SSMS sees any spatial data in the results, it adds a tab to try to visualize it. Under the covers, it's using the mapping control that Microsoft purchased from Dundas and put into Reporting Services.

On the right-hand side, we can pick the column to be displayed, because we might have more than one, and we can overlay a label if another column is holding the label. In this case, let's modify the query to have two columns, one with the value, one with the label:



The mapping tool works out where to put the label so it's somewhere on the image. SQL Server has a function call to try to help with that. (It's easy for a square, but hard for say a donut).

If more than one row of data is returned, a shape is displayed in a different color for each value. This can lead to some awesome outcomes. Note what happens if we select all of the Application.Countries table from WideWorldImporters, choose the CountryName as the label, and zoom in slightly:



This is an awesome tool for visualizing data.

### 5.3 Using the XML editor and Increasing the XML output size

Most people use SSMS to edit SQL queries. No big surprise there. The files will have a file type of **.sql**.

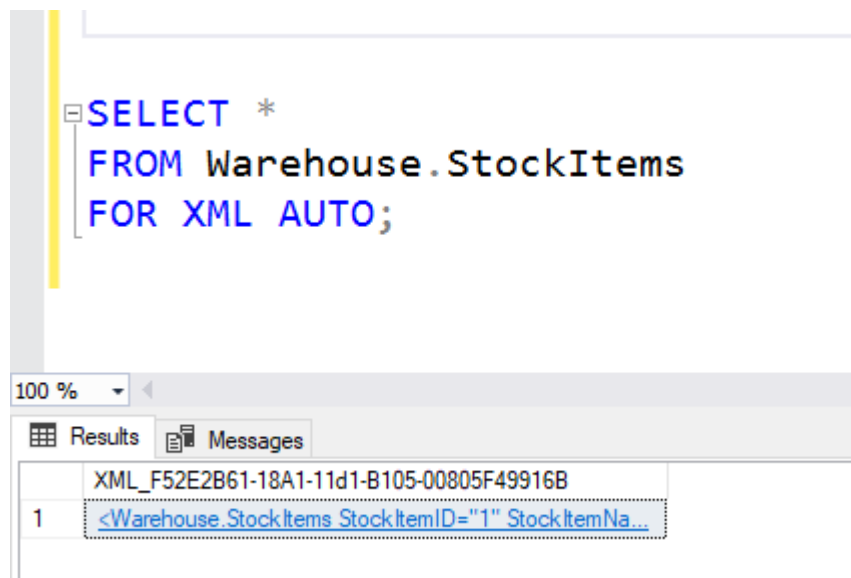
But what many people don't understand is that SSMS inherits many of its underlying Visual Studio's abilities to edit other document types.

For example, if you open a **.txt** text file, you can edit it just fine, and you can also include files like this in SSMS script projects. That can be useful for additional notes and documentation.

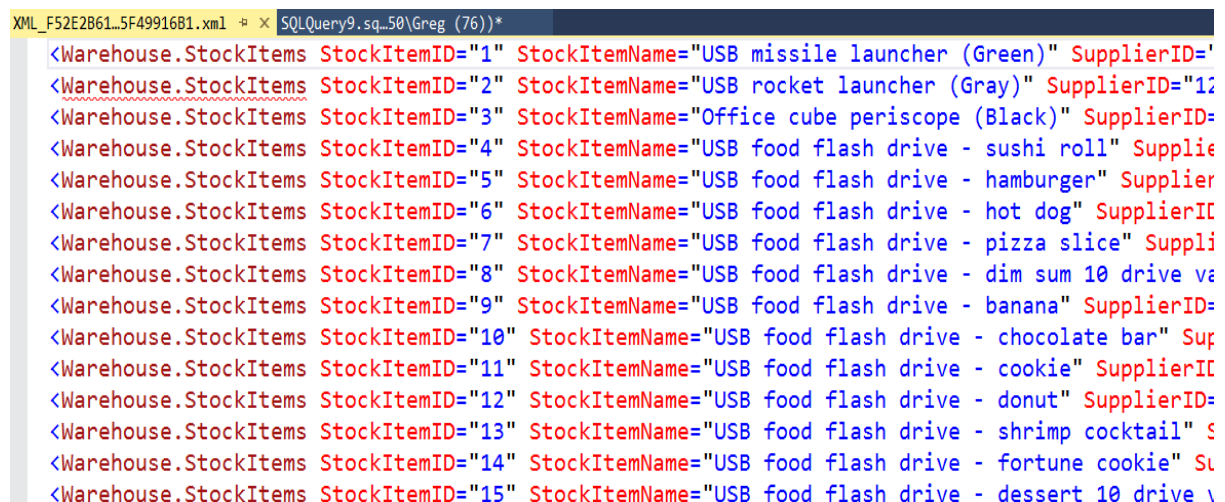
SSMS also knows how to open related file types like **.sqlplan** (for query plans) and **.xdl** files (for deadlocks), and more.

Most of these other file types though, are actually XML files with specific schemas constraining their contents. **SSMS also contains a perfectly acceptable XML editor.**

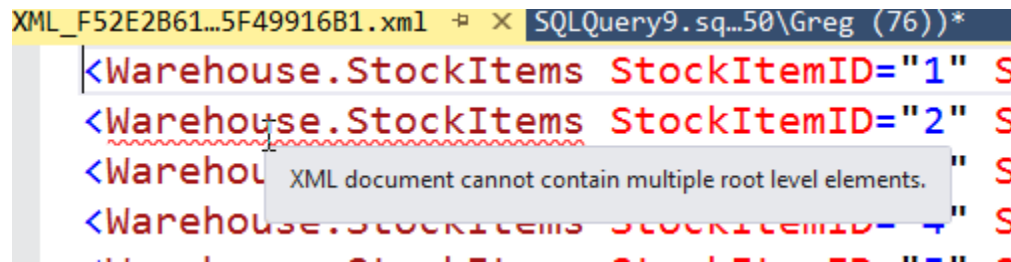
Here's an example:



If I execute the above query, the outcome is some XML. Note that SSMS recognizes that the output data type is XML and then provides a hyperlink for opening it. If I click on the link, I see this:

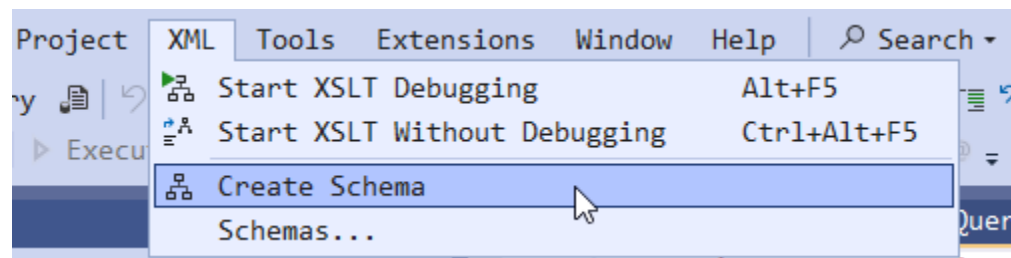


The important item to notice here though is the red squiggly on the second line. If we hover over that, we'll see this:

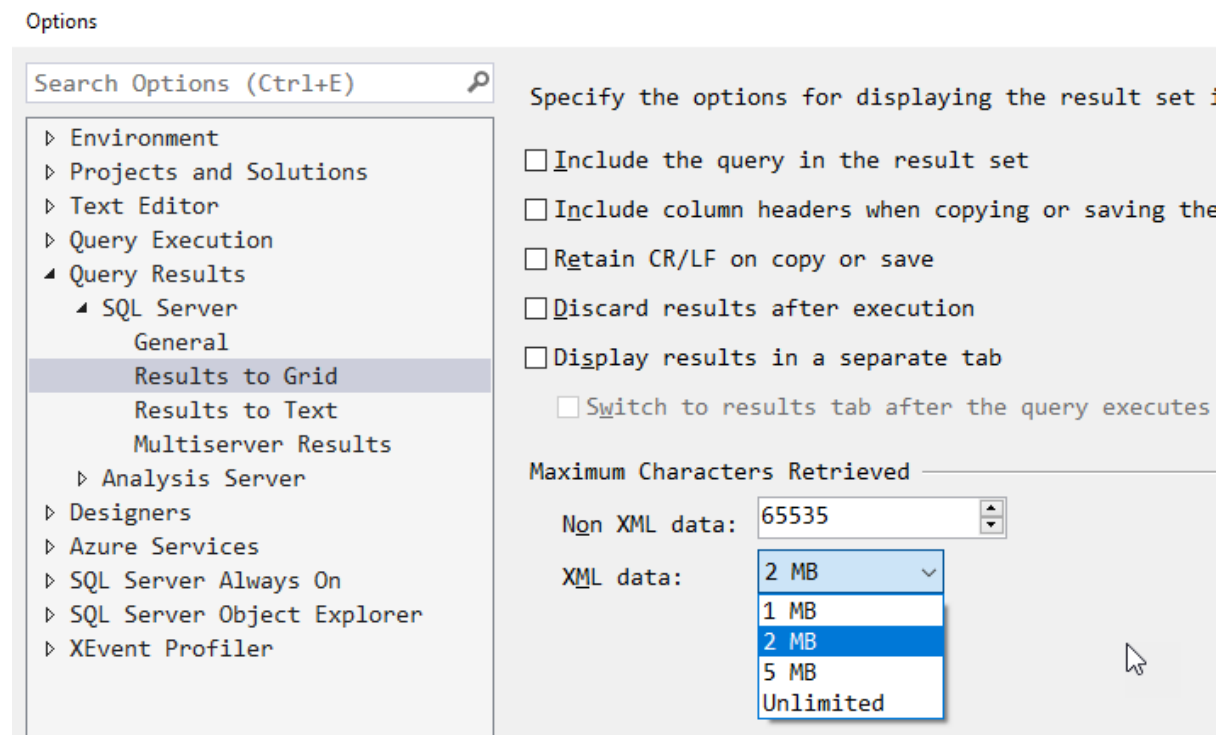


XML documents can only have a single root element. This XML is actually a fragment, not a complete document, and so it thinks that all the stock item lines are all root elements.

The important thing is that this is an XML editor, not just an XML viewer. Notice that when an XML file is open, an XML menu also appears:



Now, while it's not a bad XML editor, it has limits on the size of the data that you can work with, but you can control that too. In Tools, Options, Query Results, SQL Server, Results to Grid, you can see this:





By default, you are limited to 2MB of XML data. You can increase this to unlimited but previously this wasn't a good idea as SSMS was only a 32 bit application. Now that it's a 64 bit application, this should be safer to use.

## 5.4 Find error locations within scripts

This is probably one of the simplest tips that I've published, yet I'm endlessly surprised how many people do not realize that it's available.

When you have a script loaded in SSMS, and you execute the script, you might run into an error like this:

```
SQLQuery10.s...50\Greg (66))*
c.CustomerName,
ct.CityName,
c.PhoneNumber,
c.FaxNumber,
p.FullName AS PrimaryContactFullName,
p.PreferredName AS PrimaryContactPreferredName
FROM Sales.Customers AS c
INNER JOIN FREETEXTTABLE(Sales.Customers, CustomerName, @Sear
ON c.CustomerID = ft.[KEY]
INNER JOIN [Application].Cities AS ct
ON c.DeliveryCityID = ct.CityID
LEFT OUTER JOIN column DeliveryCityID(int, not null) ple AS p
ON c.PrimaryContactPersonID = p.PersonID
ORDER BY ft.[RANK]
FOR JSON AUTO, ROOT(N'Customers');
END;
```

100 %

Messages

Msg 343, Level 15, State 1, Line 9  
Unknown object type 'THIS' used in a CREATE, DROP, or ALTER statement.  
Msg 156, Level 15, State 1, Line 13  
Incorrect syntax near the keyword 'AS'.  
Msg 137, Level 15, State 2, Line 23  
Must declare the scalar variable "@SearchText".

To find where the error is, just double-click the error down in the Messages tab. I double-clicked it, and it took me directly to the error and highlighted it.

```
SET QUOTED_IDENTIFIER ON
GO

ALTER THIS PROCEDURE [Website].[SearchForCustomers]
@SearchText nvarchar(1000),
@MaximumRowsToReturn int
WITH EXECUTE AS OWNER
AS
BEGIN
    SELECT c.CustomerID,
           c.CustomerName,
           ct.CityName,
```

100 %

Messages

Msg 343, Level 15, State 1, Line 9  
Unknown object type 'THIS' used in a CREATE, DROP, or ALTER statement.  
Msg 156, Level 15, State 1, Line 13  
Incorrect syntax near the keyword 'AS'

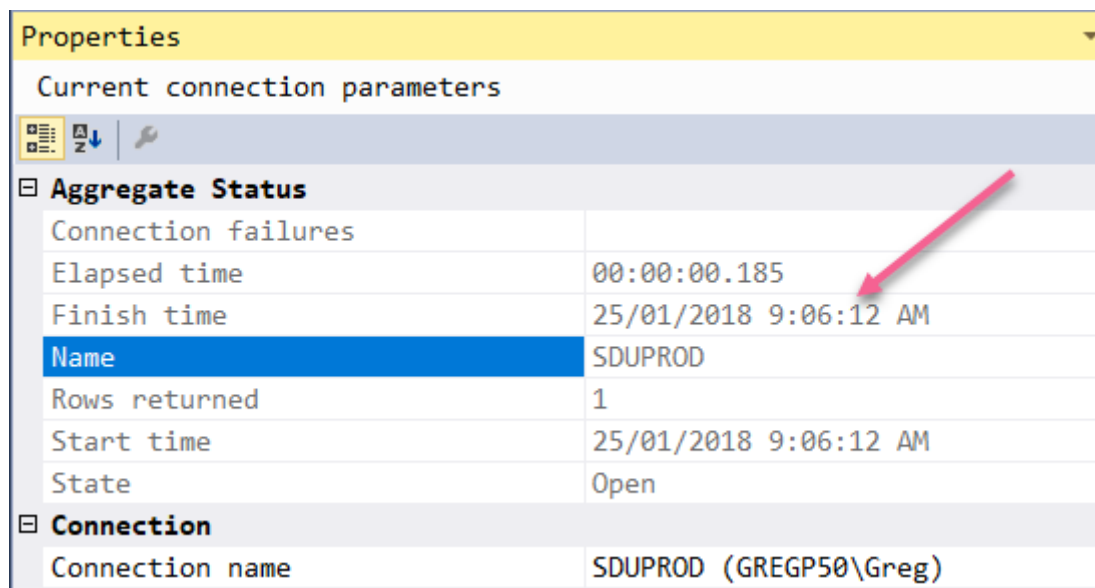
## 5.5 Determining when a query finished


It's likely that everyone who uses SSMS knows how to tell how long a query ran for. You can see it in the bottom right of the status bar when a query finishes.

But one question that often comes up with a long-running query is **when did my query finish?**

That's not in the status bar and many people don't seem to be aware that you can determine that.

It's part of the data in the Properties window. So, when you come to a query window where the query has finished, and you're wondering when it finished, hit F4 (or right-click in the window and click Properties), and you'll see this info shown:



Properties	
Current connection parameters	
	
[-] <b>Aggregate Status</b>	
Connection failures	
Elapsed time	00:00:00.185
Finish time	25/01/2018 9:06:12 AM
Name	SDUPROD
Rows returned	1
Start time	25/01/2018 9:06:12 AM
State	Open
[-] <b>Connection</b>	
Connection name	SDUPROD (GREGP50\Greg)

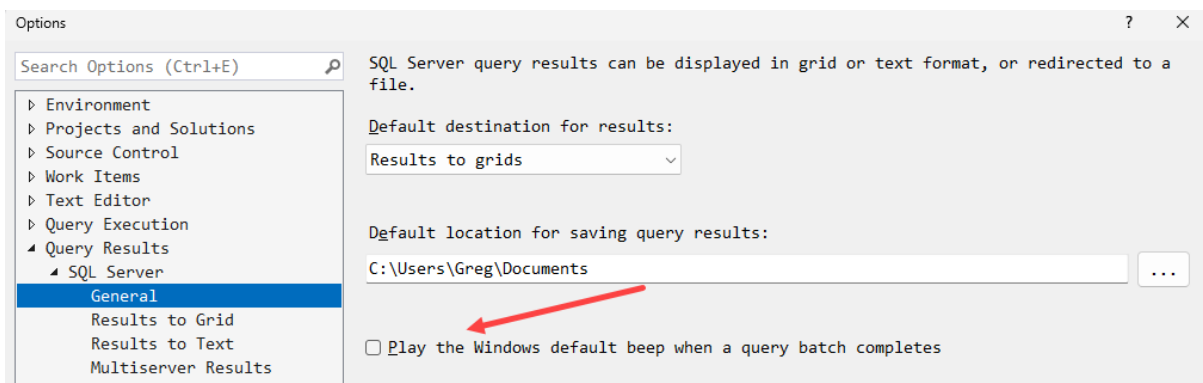
## 5.6 Play a sound when a query completes

In another section, I mentioned that when a long running query completes, I might not have been waiting around for it, and so I wanted to know when it completed.

But sometimes I do wait around for a query to complete, yet I'm distracted by other things and don't realize that the query has actually completed. That's not surprising because if a query takes a long time, I'm probably going to go on with other work while that's running.

So I want to get a prompt when the query finishes.

SSMS does provide an option for this. In Tools, Options, Query Results, there is an option to **Play the Windows default beep when a query batch completes**.



I do wish it was a stronger option than this but at least it's a start.

What I'd particularly like would be:

- Ability to play a different sound, not just the default beep.
- Ability to enable/disable this on a specific query window once a query is already running.

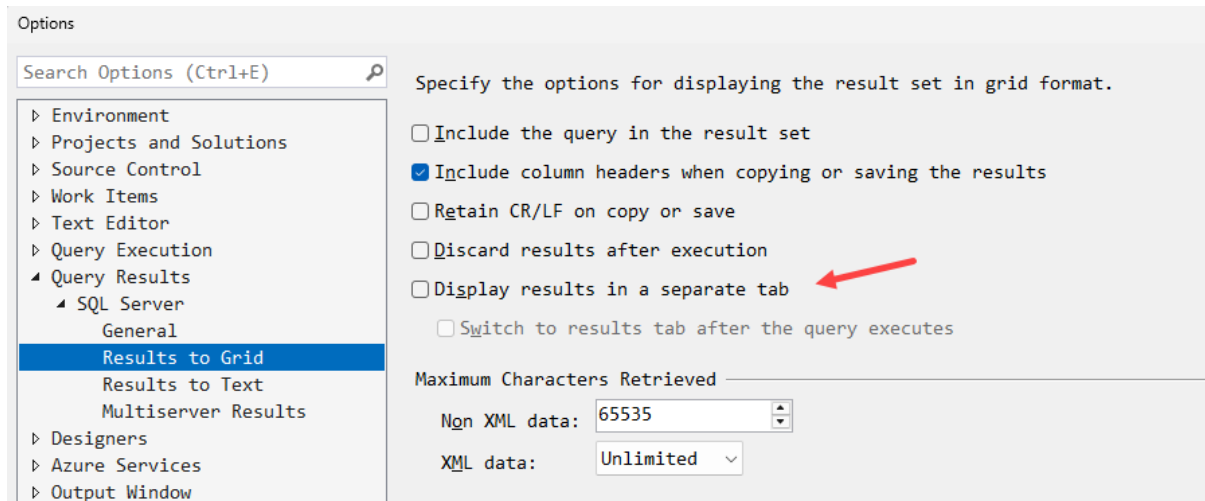
Having this on all the time would be quite annoying, so I'd be pretty selective about using it in its current form.

## 5.7 Query and results in separate tab

In SSMS query results are normally shown at the bottom of the query window.

This can greatly reduce the screen real estate both for the query, and for viewing the results.

In **Tools, Options, Query Results, SQL Server, Results to Grid**, there is an option to **Display results in a separate tab**. This can be very useful and generally you will also want to choose the extra option to **Switch to results tab after the query executes**.



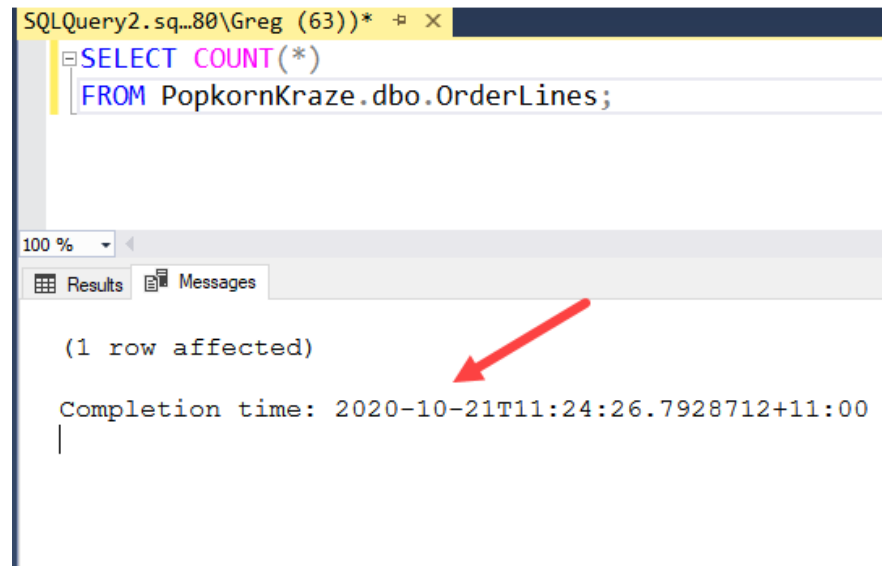
There is a similar (but separate) option for working with results to text rather than grid.

One further option on this dialog that's worth mentioning is the **Include the query in the result set**. I find that useful when working with text output, not so much with grid output.

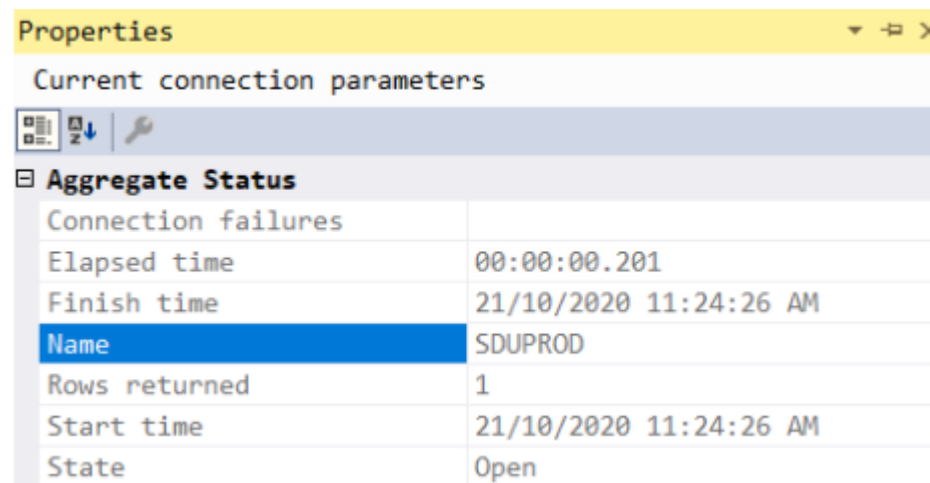
## 5.8 Turning off completion times

I've had many developers ask me *"how can I turn off those annoying completion time messages in SSMS?"*

A while back, the product team decided that we'd all like completion times shown in the Messages output tab. I don't share their enthusiasm for them.



For a start, a completion time was already shown here in the Properties window for the query, along with much more info:

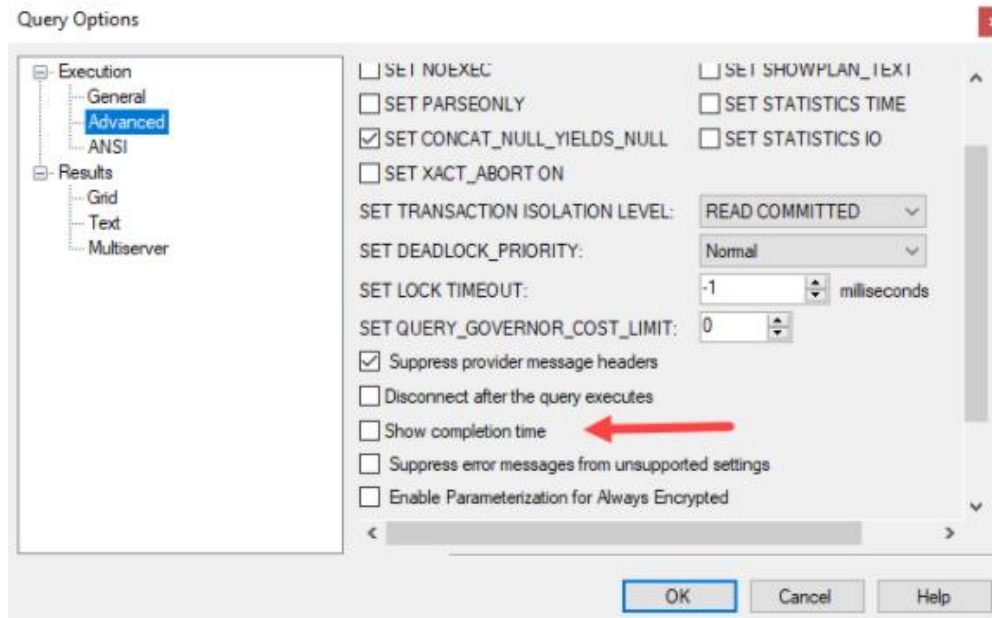


So, for me, it really was quite irrelevant. Worse, I often use the output of the Messages tab as text for scripts, etc. The last thing I wanted were a bunch of completion times sprinkled through those.

Initially, you couldn't turn it off either. Fortunately, the team listened to us all complaining about it, and they fixed it.

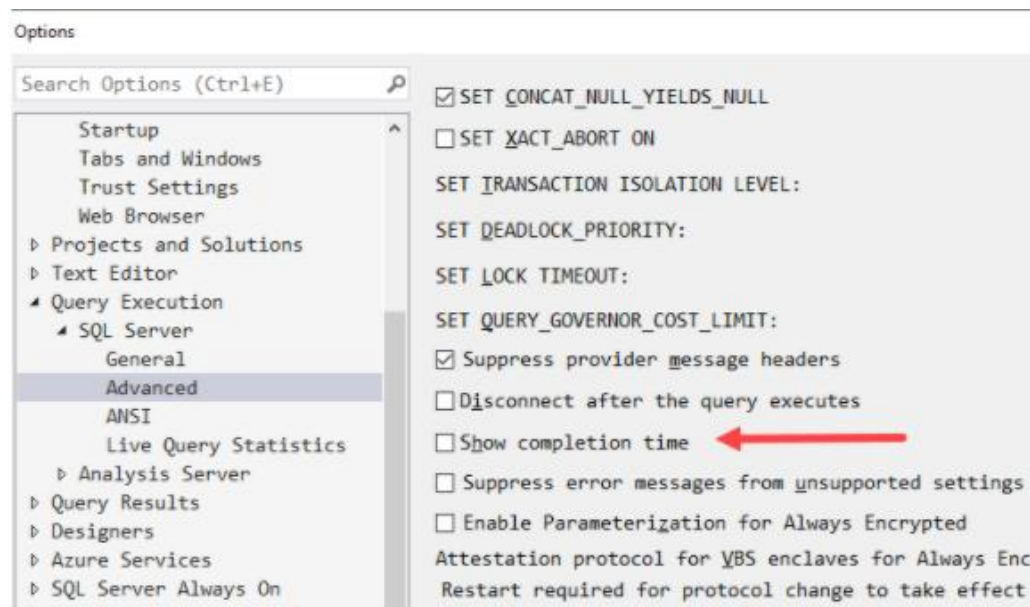
## For One Query Window

If you want to disable these messages for a single query window, you can do it here from the **Query** menu, then **Query Options**, then **Advanced**.



## As the Default

If you want this disabled by default, you can do that from the **Tools** menu, then **Options**, then **Query Execution**, then **SQL Server**, then **Advanced**:



And say goodbye to those messages.





## 5.9 Closing Idle Connections

One challenge that I find with T-SQL is that there's no 100% reliable way to drop a database. I wish I was joking.

If you execute DROP DATABASE, the command will fail if anyone is connected to the database. The way that we normally drop databases is as follows:

```
USE master;
```

```
GO
```

```
IF EXISTS (SELECT 1 FROM sys.databases AS d WHERE d.[name] = N'somedb')
```

```
BEGIN
```

```
    ALTER DATABASE somedb SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
```

```
    DROP DATABASE somedb;
```

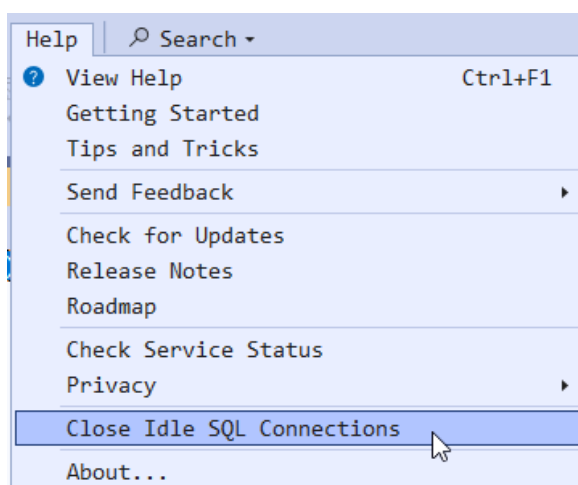
```
END;
```

That mostly works, but the problem is that I need to execute the command from the **master** database. That means that when I set the database to single user, I don't know that I'm the single user. What I've seen happen sometimes, is that the Intellisense system in SSMS is reading further down my script, where I'm perhaps recreating the database, and it's maintaining a connection to the DB.

So, my DROP fails.

The Intellisense connection is a pooled connection, and stays around for at least a few minutes after you last use it. So, even if you seem to have nothing connected to the DB, it can still be connected.

To try to help with this, the SSMS team have a **Close Idle Connections** option. It closes any idle pooled connections.



It's a pity that in the name of the menu option, it doesn't mention that it only affects pooled connections. Your normal query window connections are not pooled and are unaffected by this option.

I do wish there was an option to close any connection that has been idle for more than a certain number (e.g., 5) of minutes. But that's not what it does.

There is one other strange thing that was added to T-SQL a while back. While the team was adding all the DROP ... IF EXISTS options, they added one for databases. It's not very useful. It also fails if anyone is connected.

### **What's needed?**

What's really needed is a way to combine the dropping with the disconnecting such as:

```
DROP DATABASE IF EXISTS somedb WITH ROLLBACK IMMEDIATE;
```

Then we wouldn't need most of these messy options.

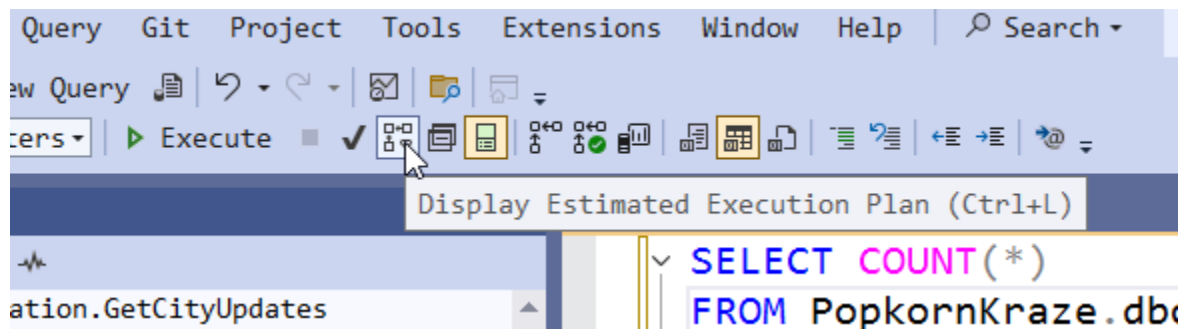
## 6 Execution Plans

### 6.1 Compare query plans

One of the advantages of SSMS is that it can be used to analyze queries, not just to execute them.

There are two basic types of query plan: estimated execution plans, and actual execution plans.

For a typical query, I can obtain the estimated execution plan, by hitting Ctrl-L, choosing the option in the Query menu, or clicking on the toolbar icon:

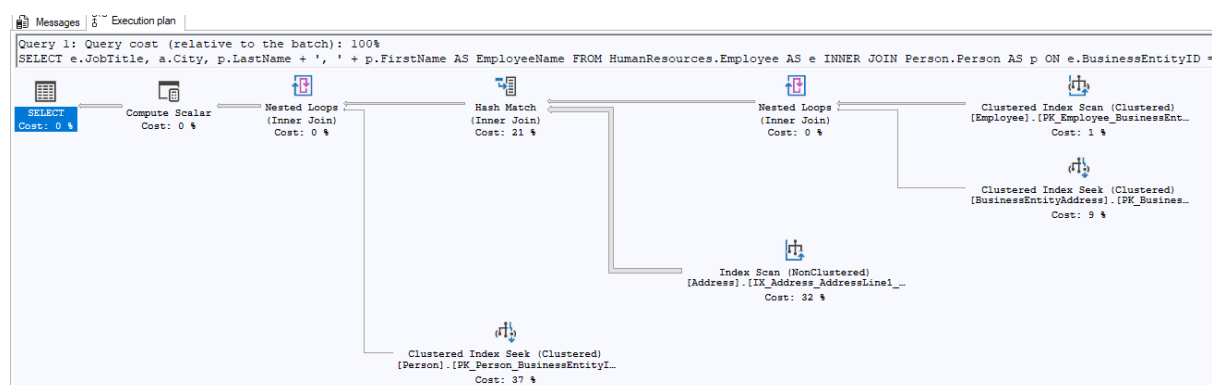


Let's do this for the following query:

```
-- Hash Match
```

```
SELECT e.JobTitle, a.City, p.LastName + ', ' + p.FirstName AS EmployeeName
FROM HumanResources.Employee AS e
INNER JOIN Person.Person AS p
ON e.BusinessEntityID = p.BusinessEntityID
INNER JOIN Person.BusinessEntityAddress AS bea
ON e.BusinessEntityID = bea.BusinessEntityID
INNER JOIN Person.Address AS a
ON bea.AddressID = a.AddressID;
```

The following query plan is returned:



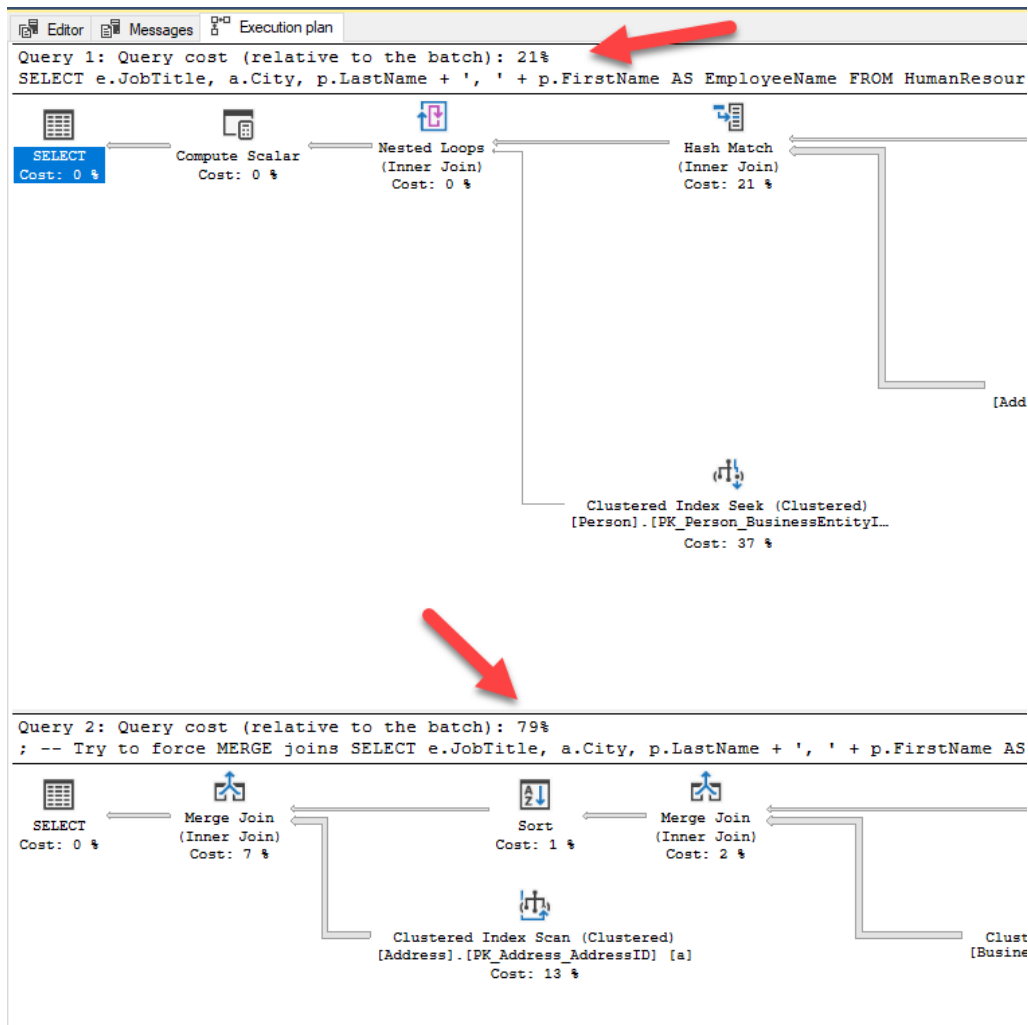
Now we might look at this plan and wonder if we exerted control over how the joins were performed, if we'd improve things. Perhaps we'd heard that merge joins were more efficient and thought SQL Server should have used those.

Now we can change the query like this:

```
SELECT e.JobTitle,  
       a.City,  
       p.LastName + ', ' + p.FirstName AS EmployeeName  
FROM HumanResources.Employee AS e  
INNER MERGE JOIN Person.Person AS p  
ON e.BusinessEntityID = p.BusinessEntityID  
INNER MERGE JOIN Person.BusinessEntityAddress AS bea  
ON bea.BusinessEntityID = e.BusinessEntityID  
INNER MERGE JOIN Person.[Address] AS a  
ON a.AddressID = bea.AddressID;
```

Notice I've added the word MERGE between INNER and JOIN in this query. But how do we know what SQL Server thinks? We can get another estimated query plan but what we really want is to compare them.

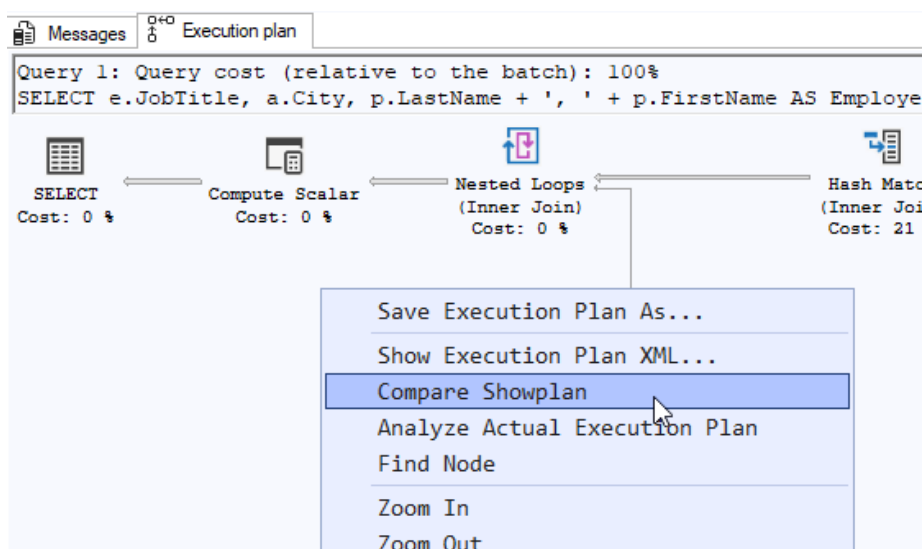
When we obtain an estimated plan for multiple queries at once, SSMS shows us a comparison of the two, by showing us the proportion of the overall query for each part. (Perhaps we shouldn't force that change 😊)



## Comparing to a Saved Plan

As well as comparing two queries directly, you can also compare the query plan for a query, with an existing saved query plan.

The option to do this is a right-click option in the displayed query plan:



When you open an existing plan to compare, a detailed comparison of the plans is shown.

**Execution plan**  
 SELECT e.JobTitle, a.City, p.LastName + ', ' + p.FirstName AS EmployeeName FROM HumanResources.Employee AS e INNER MERGE JOIN  
 Person.Person AS n ON e.BusinessEntityID = n.BusinessEntityID INNER MERGE JOIN Person.BusinessEntityAddress AS bea ON

**C:\Users\Greg\Desktop\Plan1.sqlplan**  
 SELECT e.JobTitle, a.City, p.LastName + ', ' + p.FirstName AS EmployeeName FROM HumanResources.Employee AS e INNER JOIN  
 Person.Person AS n ON e.BusinessEntityID = n.BusinessEntityID INNER JOIN Person.BusinessEntityAddress AS bea ON

**Properties**  
 Top Plan: SELECT  
 Bottom Plan: SELECT  
 Actual Num: 0  
 Cached plan: 48 KB  
 Cardinality: 160  
 CompileCP: 6  
 CompileMe: 456  
 CompileTm: 7  
 Estimated: 0  
 Estimated: 275.573  
 Estimated: 0 (0%)  
 Estimated: 2.07509  
 MemoryGr: Optimizatio FULL  
 OptimizerH: Optimizer5  
 OptimizerQ: QueryPlan: 0xF7FE9FBC  
 QueryPlan: 0x1E0C486D  
 Retrieved: false  
 SecurityPo: False  
 Set Option: ANSI\_NULLS: True  
 Statement: SELECT e.Jo

**Showplan Analysis**  
 Statement Options: Multi Statement: Scenarios  
 Highlight similar operations: ☒  
 List of similar areas in compared plans:  
 ..... Clustered Index Scan (Clustered) [Employee].[PK\_Employee\_BusinessEntityID] [e]  
 Highlight operators not matching similar segments: ☐  
 Ignore database name when comparing operators: ☒

This is very powerful.

## 6.2 Missing index details

I've mentioned before that SSMS is a good tool for analyzing queries, as much as for executing them.

Way back in SQL Server 2005, query plans had missing index details added. When a query plan was created, SQL Server recorded that it thought it could have executed the query better, if only you'd provided it with appropriate indexes. But at that point, the suggestions weren't very good, and the tools didn't show them.

In SQL Server 2008, the suggestions got better (eg: we weren't endlessly having suggestions to create non-clustered indexes for every column in the table), and SSMS now showed them clearly.

Here's an example query with an issue:

The screenshot shows the SQL Server Enterprise Manager interface. At the top, a query is displayed in a text editor:

```
SELECT InvoiceID
FROM Sales.CustomerTransactions
WHERE IsFinalized = 1
```

Below the query, the 'Execution plan' tab is selected. The top of the execution plan shows the query cost and a message:

Query 1: Query cost (relative to the batch): 100%

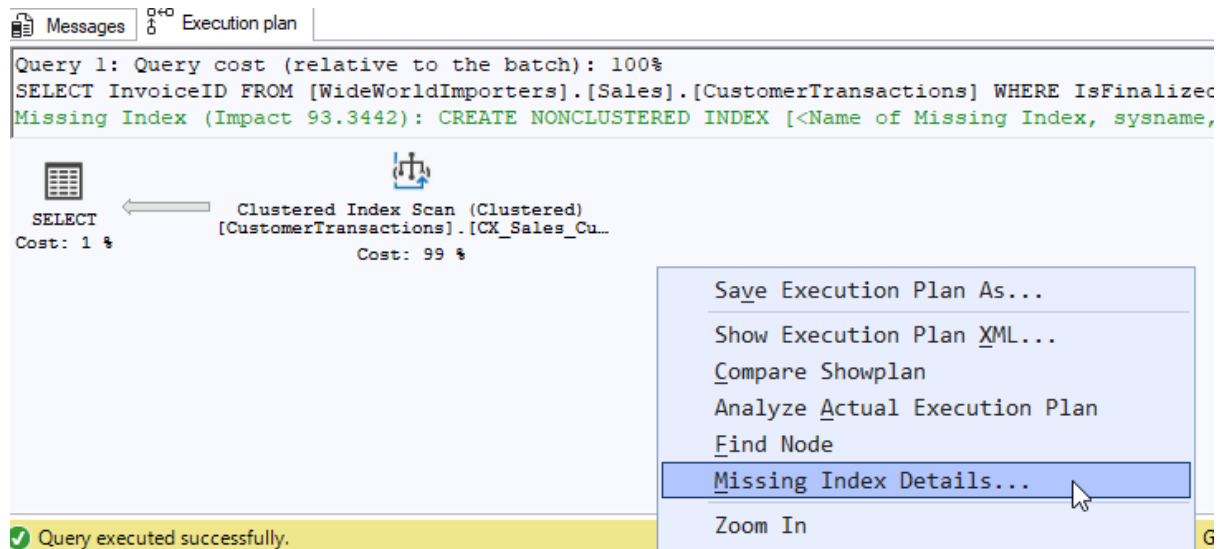
SELECT InvoiceID FROM [WideWorldImporters].[Sales].[CustomerTransactions]

Missing Index (Impact 93.3442): CREATE NONCLUSTERED INDEX [<Name of Missi

The execution plan itself shows a 'SELECT' operator with a cost of 1% and a 'Clustered Index Scan (Clustered)' operator with a cost of 99%. An arrow points from the 'Missing Index' message to the 'Clustered Index Scan' operator, indicating that the missing index would significantly improve the query performance.

SQL Server thinks it's doing this work the hard way. Note that it suggests an impact of over 97% on running this query, just because an index is missing.

To find out what it wants, we right-click in the white area of the query plan:



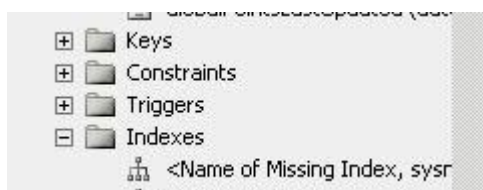
The option of interest is **Missing Index Details**. When we click that to open it, the following appears:

```
/*
Missing Index Details from SQLQuery3.sql - (local)\SQL2022.Wide
The Query Processor estimates that implementing the following i
*/

/*
USE [WideWorldImporters]
GO
CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>]
ON [Sales].[CustomerTransactions] ([IsFinalized])
INCLUDE ([InvoiceID])
GO
*/
```

Overall, the suggestions that this system makes aren't bad. They are far from perfect but if you don't know much about SQL Server indexing, these suggestions might make a good start. Just don't blindly follow large numbers of them. Someone with strong SQL Server skills can often come up with better suggestions than those currently offered by the tools.

Also, make sure you rename the index. I have come across indexes at customer sites where the index name is **[<Name of Missing Index, sysname,>]**. I wish I was joking.

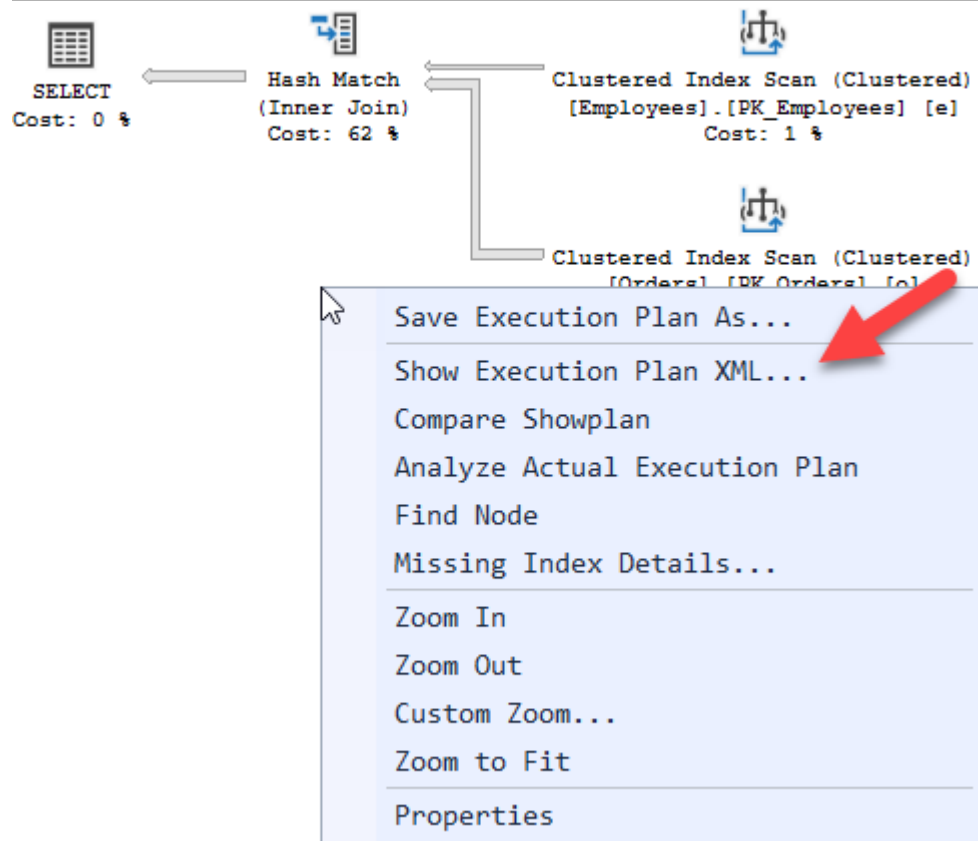




### 6.3 Saving and sharing query plans

SQL Server query plans are stored as XML. You can see what they look like by right-clicking in any query plan in SSMS, and clicking Show Execution Plan XML:

Query 1: Query cost (relative to the batch): 100%  
SELECT e.EmployeeID,e.FullName,e.BadgeNumber,e.BirthDate, o.  
Missing Index (Impact 97.7213): CREATE NONCLUSTERED INDEX [C]



That will return a whole bunch of XML like this:

It's important to understand that when SSMS is showing a graphical execution plan, it's just graphically rendering some XML like the plan above.

The Properties window in SSMS is also showing details extracted from that same XML.

Prior to SQL Server 2005, it was very difficult to share a graphical query plan with anyone else. You'd have to pop up the details of each operator one by one and take screenshots. Nasty.

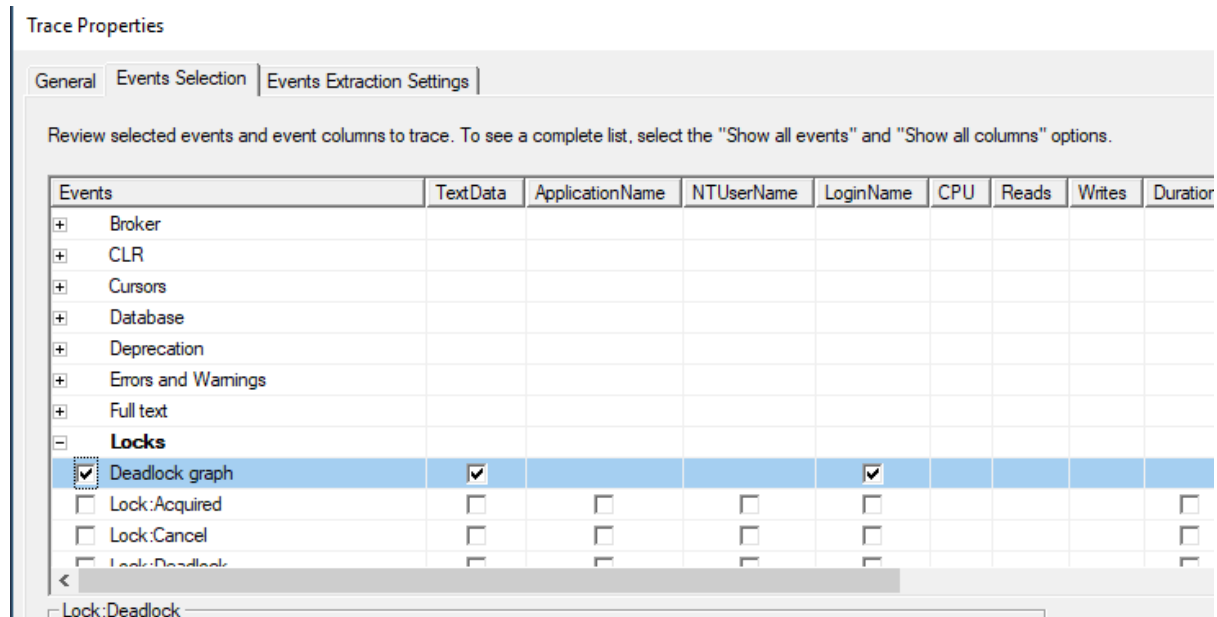
Since SQL Server 2005, query plans can be stored as XML files, with a **.sqlplan** file extension. If you save a plan, that's the default file extension. And because SSMS is set in Windows as being associated with this file type, you can just open a **.sqlplan** file and see the full graphical plan in SSMS, including having all the popups working.

## 6.4 Saving and sharing deadlock graphs

In an earlier entry, I described how query plans could be saved as **.sqlplan** file, shared, and loaded again in SSMS. It's also possible to extract them out of SQL Server Profiler or Extended Events Profiler.

This is useful, but the same applies to deadlock graphs. SQL Server 2005 added **Deadlock graph** as a type of event in SQL Server Profiler. (It's also part of Extended Events Profiler).

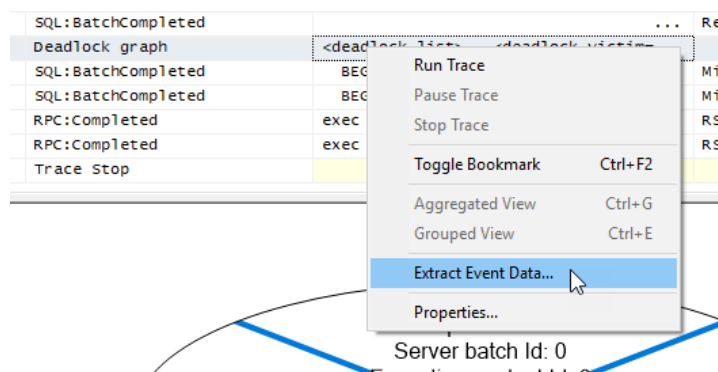
If I open a new trace in Profiler, I can add Deadlock graph to the list of events:



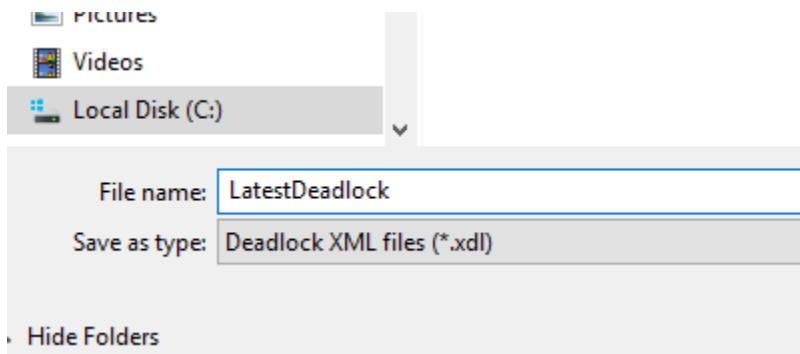
Then when a deadlock occurs, we can see an entry for it in the trace:



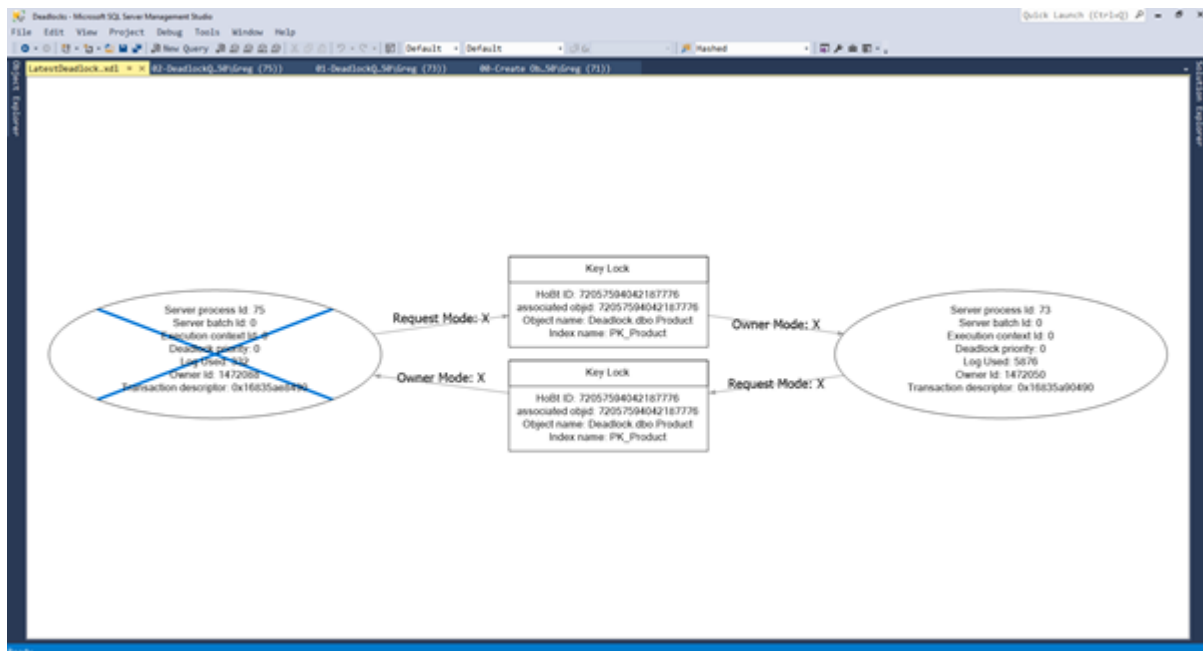
Now if you want to share that with someone else, you can right-click the deadlock graph event and extract the event data:



When you do this for a deadlock, the output is saved as a **.xdl** file extension.



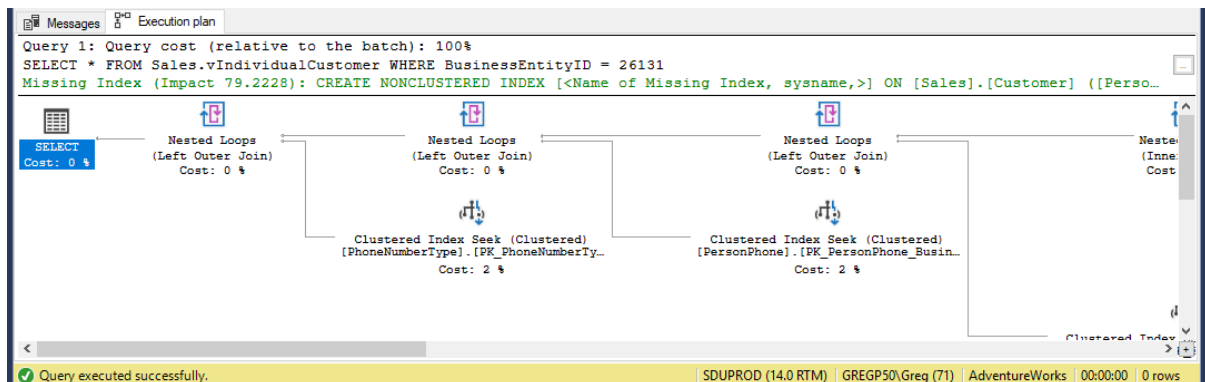
You can then send it to someone else and they can load it in SSMS:



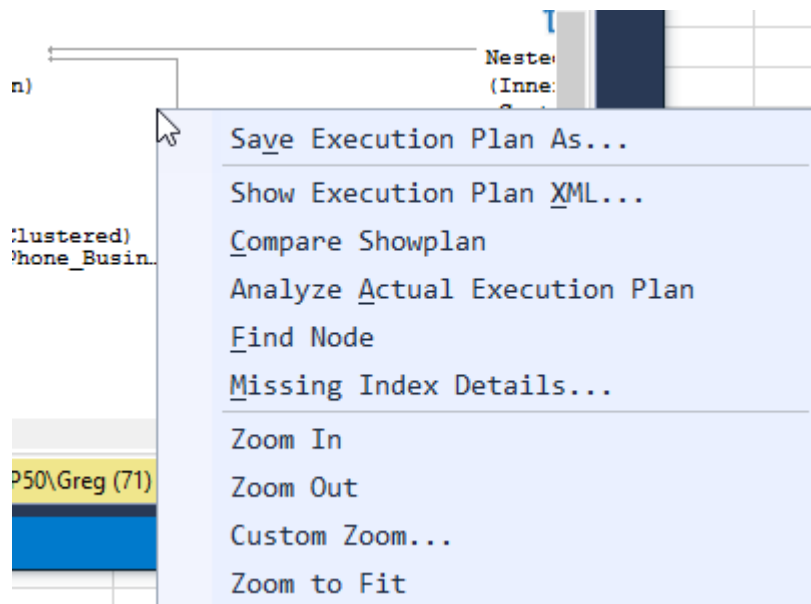
One important thing to note is that a deadlock graph is actually a list of deadlocks, not just one deadlock. When a deadlock graph is opened in SSMS, it only shows the first deadlock in the graph.

## 6.5 Zooming and navigating query plans

SQL Server execution plans can become quite large. That makes them hard to navigate because you are endlessly scrolling around the results pane in SSMS.

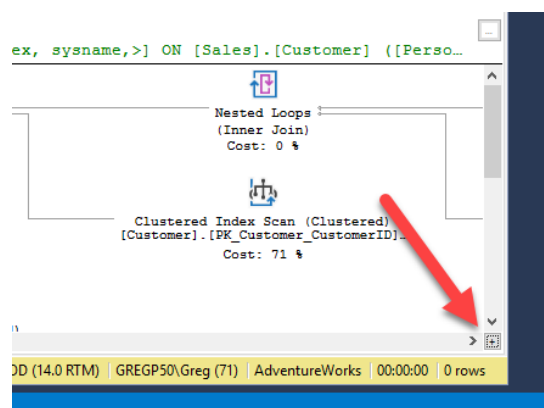


The pane does have some zoom features. Note that if I right-click in the whitespace, I get these options:

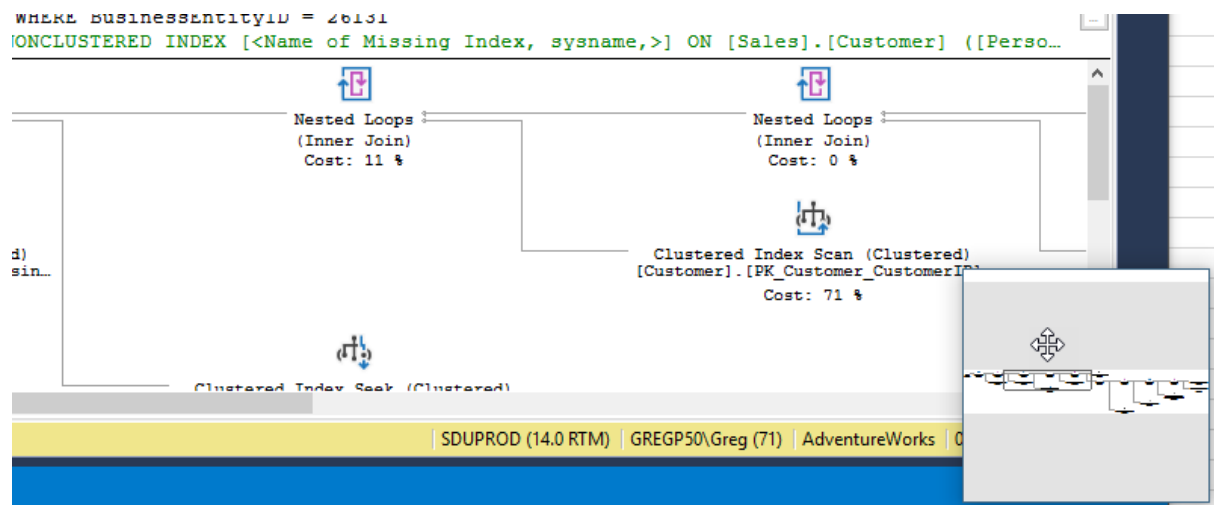


So I can zoom in and out, set a custom zoom level, or zoom until the entire plan fits. Generally though, that would make the plan too small to read, as soon as you have a complicated plan.

But in **one of the least discoverable UI features in SSMS**, there is an option to pan around the plan.



If you click and hold that little + sign, you'll find you can pan around within the plan:



That's very nice but I think it needs to be a little easier to find.

## 7 Projects/Solutions

---

### 7.1 Change default text in new query windows

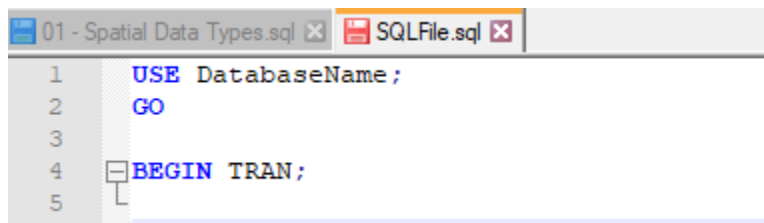
In SSMS, when you click New Query, a new query window is opened, but because it's blank, what might not be immediately obvious is that it's based on a template.

The location of the template depends upon the version, but for SSMS 17.6, you'll find it in this folder:

**C:\Program Files\Microsoft SQL Server Management Studio  
21\Release\Common7\IDE\NewFileItems\SQLFile.sql**

The file is called **SQLFile.sql**. If it's not in the location above, just search for **SQLFile.sql**.

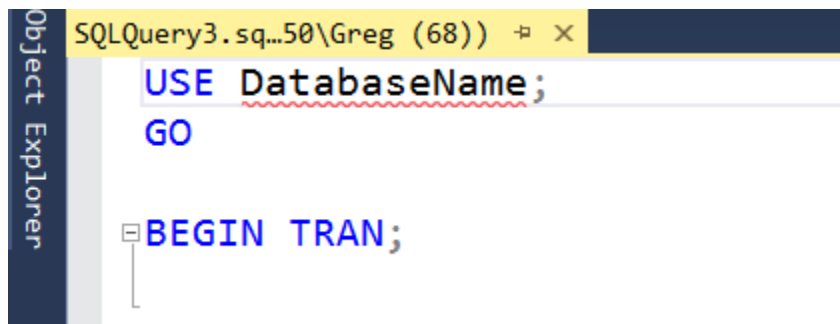
One of the things that I often forget to do is to change my connection to the correct database. Let's add a USE statement to make that obvious.



```
01 - Spatial Data Types.sql x SQLFile.sql x
1  USE DatabaseName;
2  GO
3
4  BEGIN TRAN;
5  [
```

I've also put a BEGIN TRAN that I might use before doing any ad-hoc modifications.

So, I save that file, and the next time I click **New Query** in SSMS, I see this:

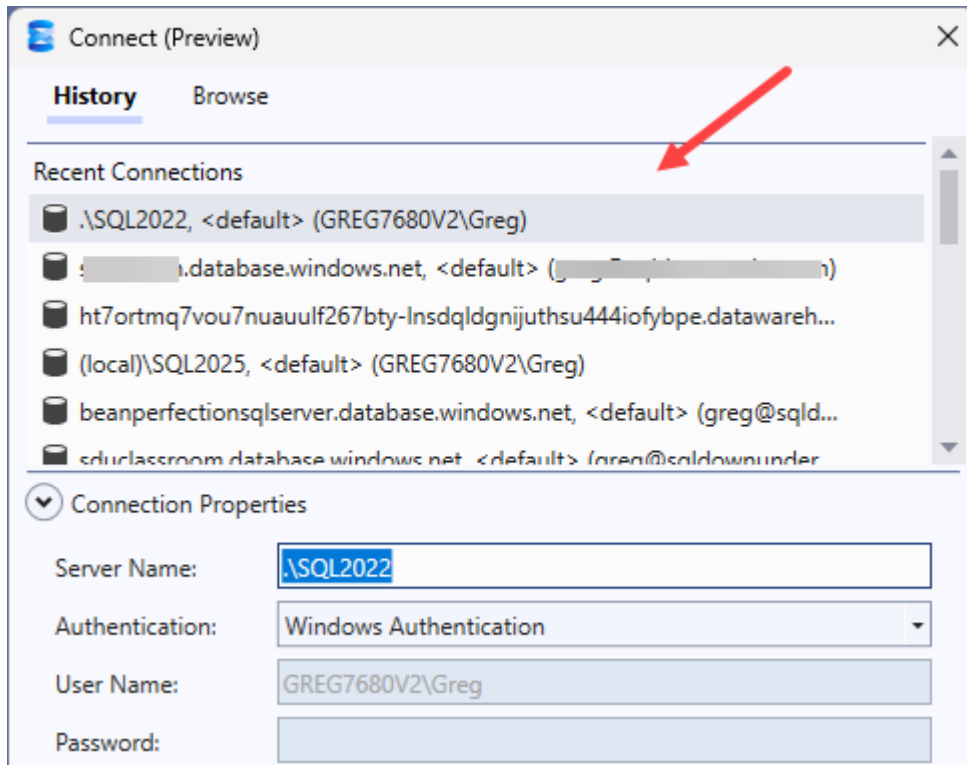


```
SQLQuery3.sql...50\Greg (68)) x
USE DatabaseName;
GO
BEGIN TRAN;
[
```

If I was really keen, I might also add templated values to be completed.

## 7.2 Pinning and clearing the connection entries

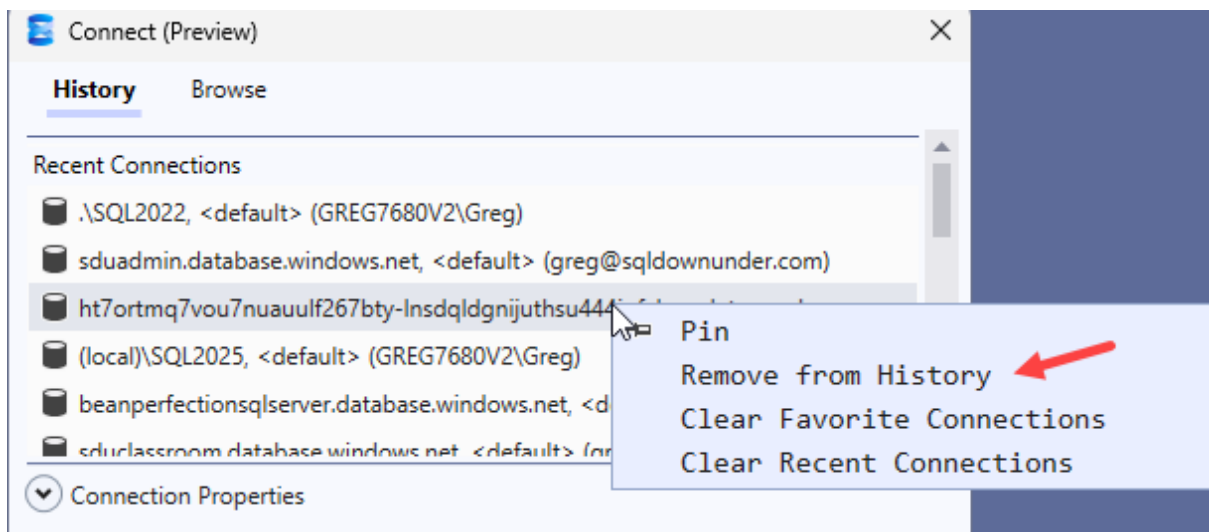
SSMS keeps a list of the server names that you have connected to, and prompts you with those when you drop-down the list while making a connection:



Eventually, that list can either become messy, it can include servers that don't exist anymore, and so on. You might want to clear up the list.

To do this in early versions of SSMS, you needed to locate the SqlStudio.bin file from the Documents and Settings area in your user profile. Fortunately, that's no longer required. In more recent versions, all you needed to do was to open this dialog, arrow down to the ones that you want to remove, and hit the Delete key.

Now from version 21, there's a menu:





Note that it also provides an option to remove more than just that single entry, and an option to pin a particular entry to the list, so that it's always there at the top.



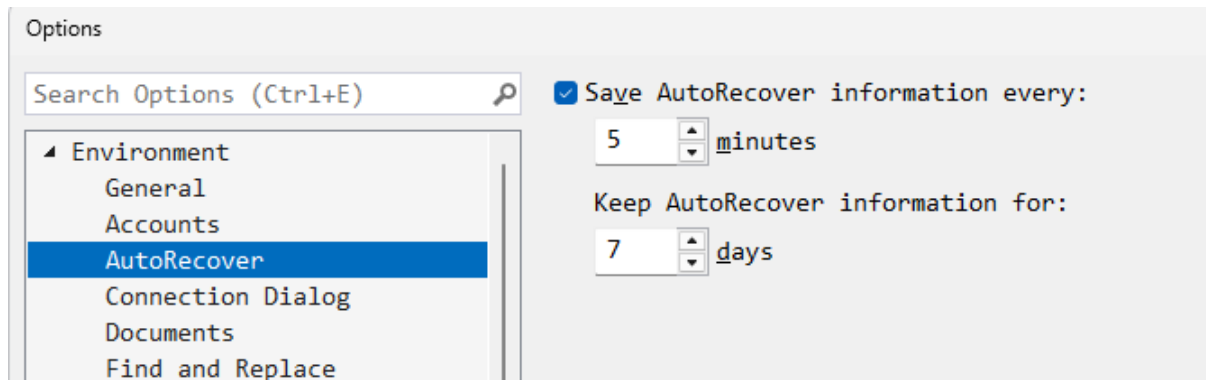
It then appears in a separate Favorites section which also has an option for removing all entries.

### 7.3 Configure autorecover time, and recover unsaved queries

Every now and again, I come back to my laptop and find that it has rebooted for some reason, while I wasn't expecting it. A prime cause of that is Windows Updates. I really, really wish that wasn't so, but someone at Microsoft has decided that I must apply these updates. I have very little control over the time when that occurs. For example, if I'm on the road delivering presentations, there's no "wait till I get home" option for Windows Updates.

Either way, it's really helpful that SSMS creates periodic backups of unsaved queries.

You can control how often it does this, and how long it keeps the files for, by adjusting the values in Tools, Options here:



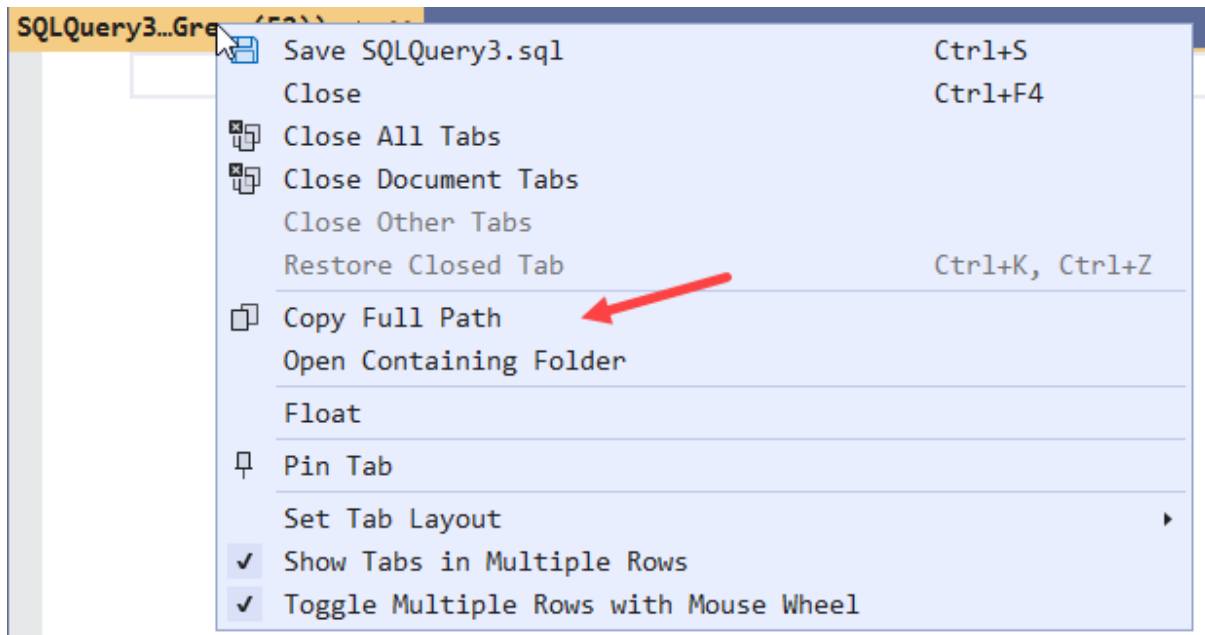
If SSMS crashes, you'll be prompted to recover the unsaved files when it restarts. But if you want to just go and find these files after another type of shutdown that you weren't planning on, you'll find them in your Documents folder in Windows, under the subfolders, SQL Server Management Studio, then Backup Files, then the name of a solution (likely Solution1 if you didn't create a scripts project).

## 7.4 Accessing script files and folders in SSMS

This one is a very simple and quick tip.

When working in SSMS, I often need to open Windows File Explorer in the folder where the script file is stored. Turns out there is an easy way to do that.

There are two interesting options when you right-click the tab at the top of a query window. (Note: not on the File menu).



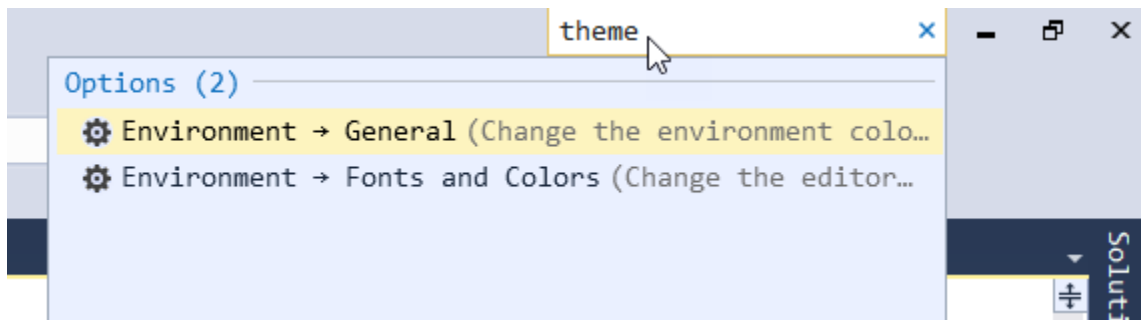
Note that you can open the containing folder for the script. You can also copy the path to the script into the clipboard.

## 7.5 Using quick launch

Back when SSMS for SQL Server 2016 was released, a search tool called **Quick Launch** was added. It was this bar up the top of previous versions:

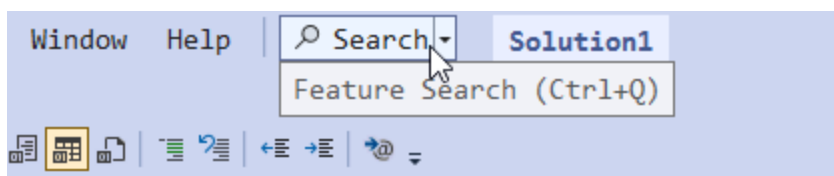


Note there was also another bar underneath it to the left. That was the **Find** bar. While the Find bar was useful for searching for text within queries, etc., the Quick Launch bar was useful for searching within SSMS itself. This was great because it means you don't have to remember where all the options for various things are set. Here was an example:

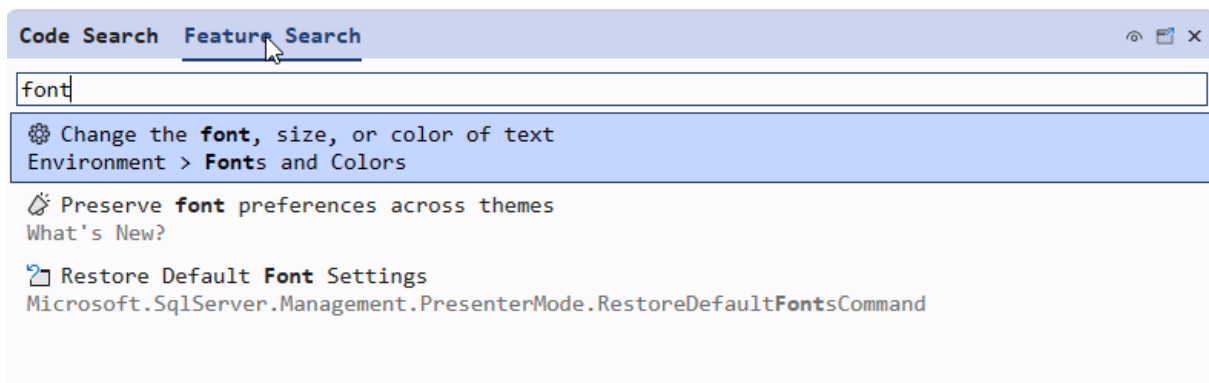


### SSMS Feature Search

From v21, those bars are gone, and have been replaced by a Feature Search:



When I open this and type **font**, I see this:



This shows the menu items that contain that word. Note there is also a **Code Search** option.

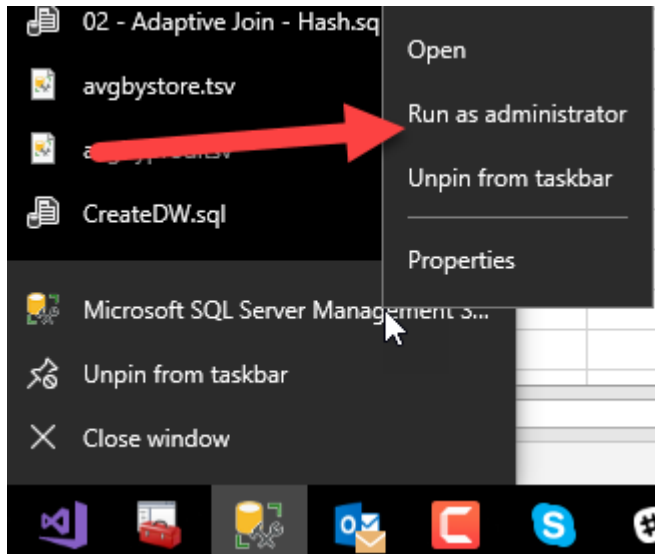
## 7.6 Run SSMS as someone else

You don't always want to run SSMS as your current login for Windows.

Now if all you want to do is to use a SQL Server login, then that's easy. When you connect to a server in Object Explorer, or when you start a new Database Engine query, you can just choose SQL authentication instead.

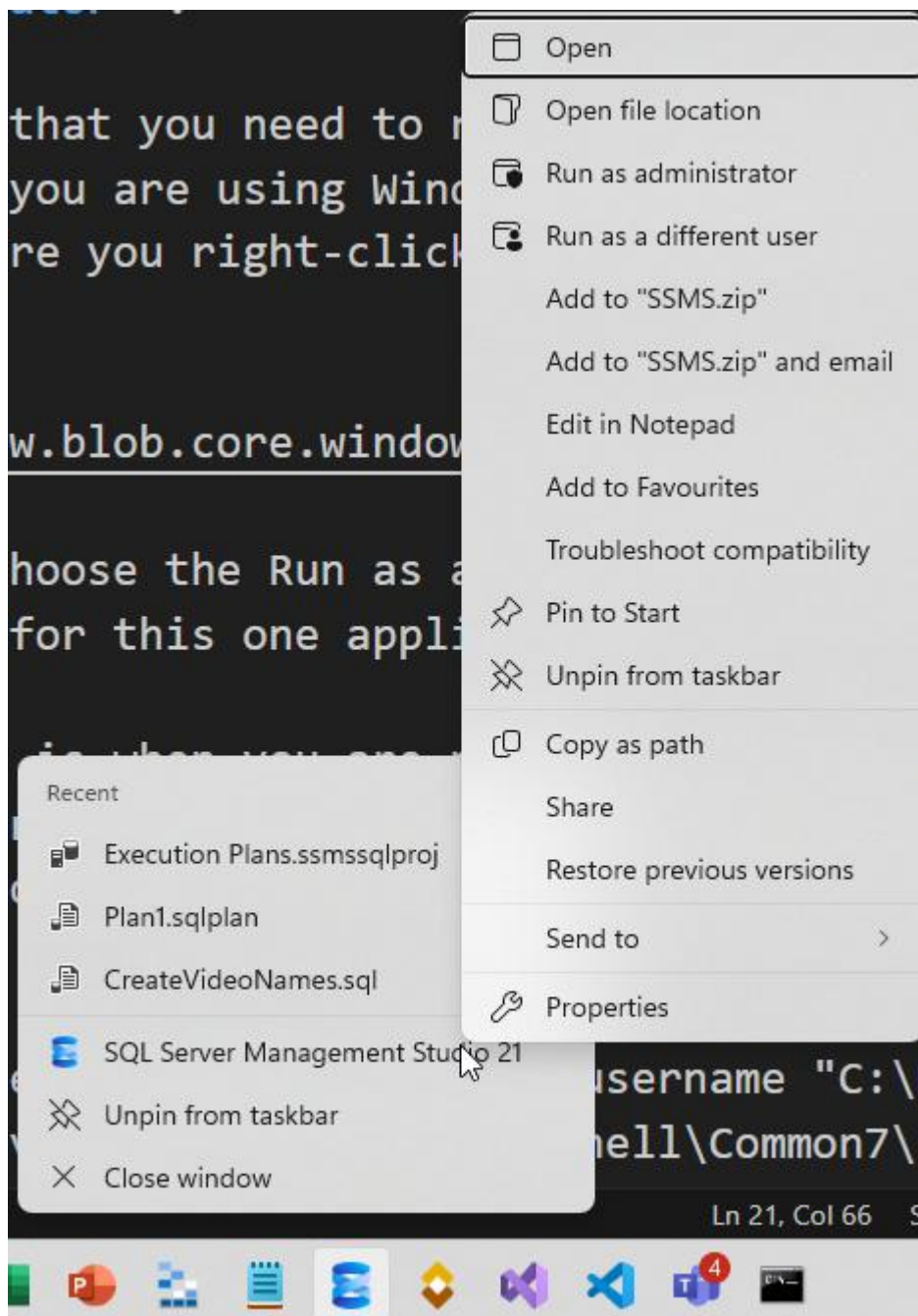
But three other scenarios commonly occur.

If you need to run SSMS as an administrator on a machine with UAC, you can do this:



You right-click the link to SSMS,, then right-click the menu entry, and choose **Run as administrator**.

Another option is that you need to run as another user ie: not just an administrator. If you are using Windows 10 or Windows 11, the way to do this is to hold the Shift key before you right-click the SSMS link. Then you see this instead:



And you can then choose the Run as a different user option to log on as someone else just for this one application.

The third scenario is when you are needing to make a Windows authenticated connection to a server, but from a system that is not part of the domain. In that case, you need to run SSMS from the command line like this:

**runas /netonly /user:targetdomain\targetusername "C:\Program Files\Microsoft SQL Server Management Studio 21\Release\Common7\IDE\SSMS.exe"**

You will need to modify that command to point to the current location of Ssms.exe for the version that you are running.

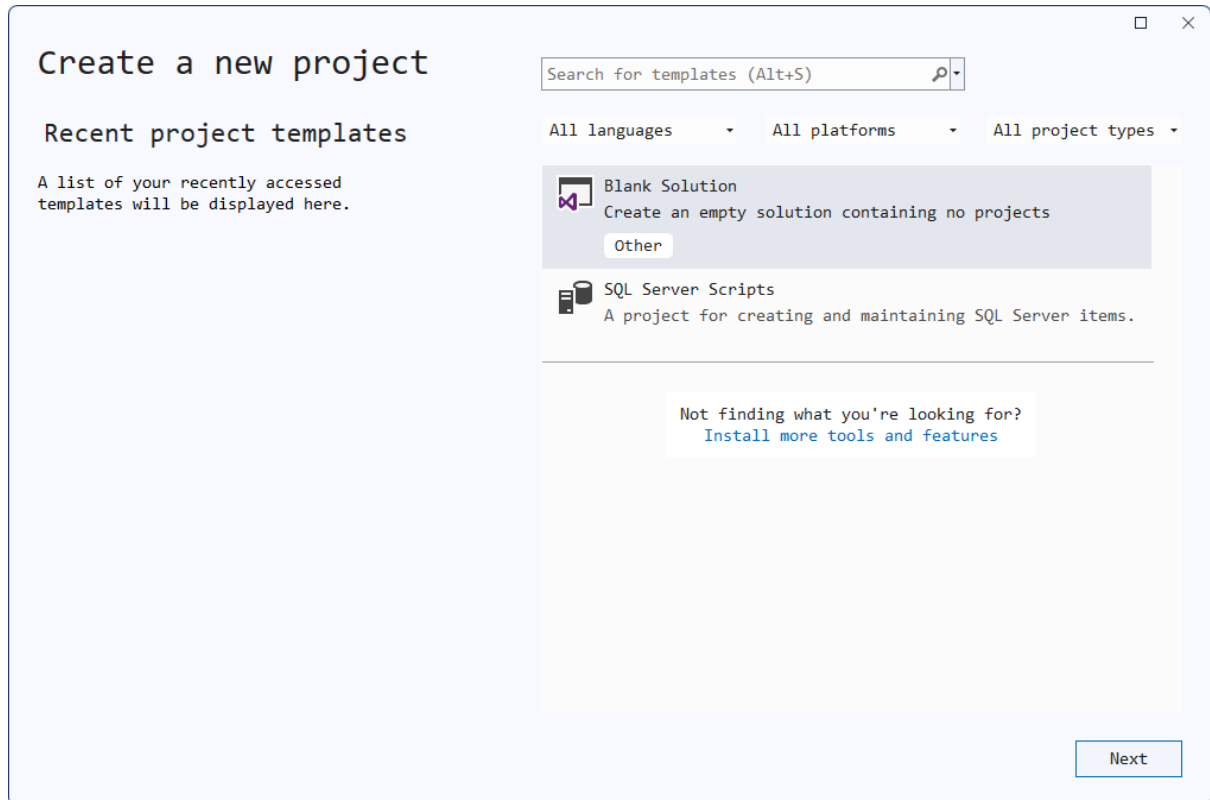
You will then be prompted to log on to the domain.

## 7.7 Using script projects and solutions

I'm puzzled that so few people use script projects and solutions when working with SSMS.

They are easy to use. Let's see an example:

Instead of just starting to create scripts, from the File menu, click New, then Project. You are greeted with the new Project dialog which also allows you to create a solution.





I then select SQL Server Scripts as the project template and Next:

Configure your new project

SQL Server Scripts

Project name

SQL Server Scripts1

Location

C:\Users\Greg\Documents\SQL Server Management Studio ...

Solution

Create new solution

Solution name ⓘ

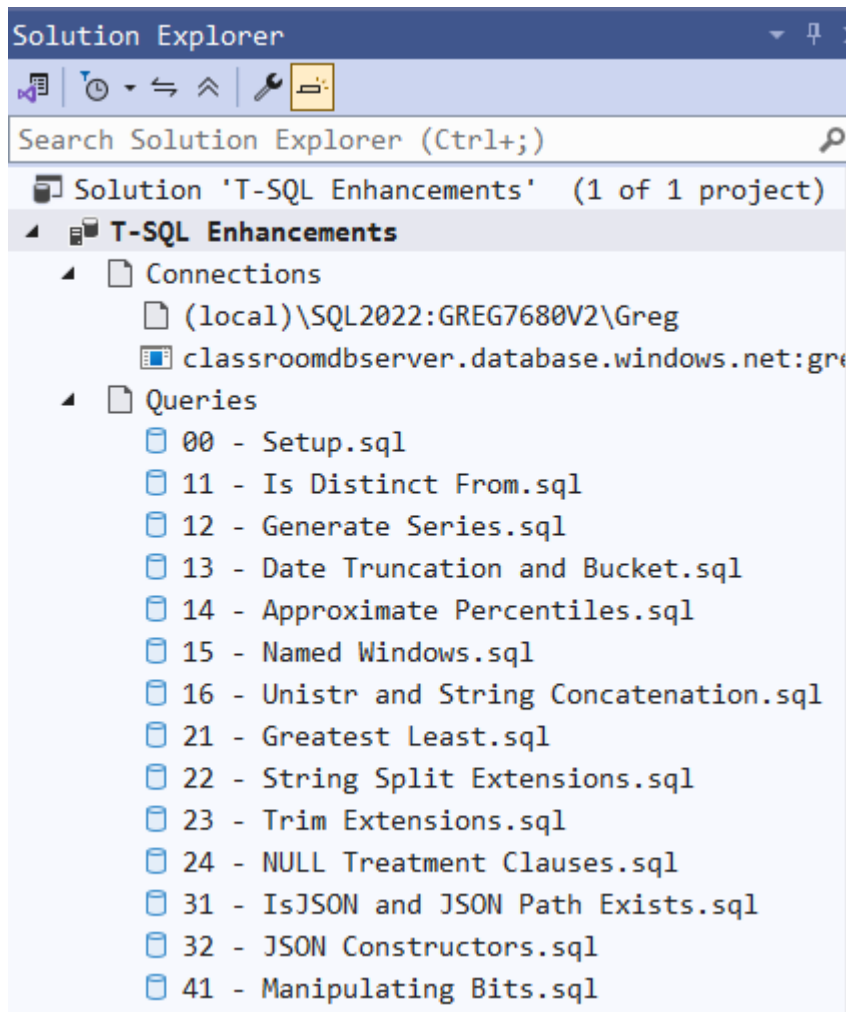
SQL Server Scripts1

☐ Place solution and project in the same directory

Project will be created in "C:\Users\Greg\Documents\SQL Server Management Studio\SQL Server Scripts1\SQL Server Scripts1\"

Back Create

To get to the point faster, here's one that I created earlier:

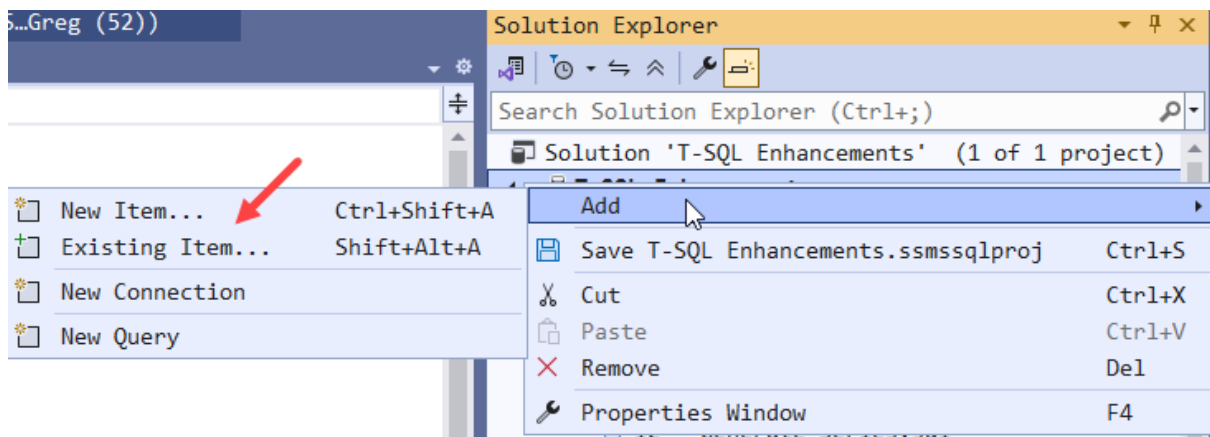


There are three sections of interest:

**Connections** – keeps details of all connections used by the project. A big advantage of this is that if I need to change the connection, I do it once, and all scripts here will use it when I open them.

**Queries** – as the name says. A nice addition in recent versions is that it automatically keeps them shown in alphabetical order. It didn't used to do that so we had to remove them all and put them back in.

**Miscellaneous** (not shown above but appears below Queries)– you'll find if you right-click this that there's nothing of interest. So how does something get there I hear you ask? Well, it's when you right-click the project and ask to add an existing item.



If you pick any file apart from those with a **.sql** extension (such as a text file), they'll be placed into the Miscellaneous folder.

These projects are a great free built-in way to manage your scripts.

## 7.8 Start faster by disabling CRL checking in constrained environments

If you have ever started SSMS in an isolated environment (ie: one with no external Internet connectivity), you'll find that it's slower to start.

That's because SQL Server uses signed assemblies, and whenever an application with signed assemblies starts, it needs to check whether or not the certificate that they were signed with has been revoked. It's not good enough to just check if it's a valid certificate.

Certificates include a CRL (Certificate Revocation List) and this tells an application that's trusting the certificate where to check for a list of revoked certificates.

The problem is that when you try to locate a server in an isolated environment, you might see a delay of around 40 seconds as the DNS timeout occurs.

**If you have an environment like this, you might decide that it's safe to turn off this revocation checking. That's a call you need to make, and if in doubt or don't understand the issues, leave it on.**

I often run across this though as I have isolated virtual machines running in Hyper-V on my laptop. SSMS isn't going to be able to look these details up, nor is any other application running within the virtual machine.

Turning this off is a registry setting but, depending upon the OS build, there are other ways to do it, such as:

1. Control Panel → Internet Options → Advanced
2. Scroll down to the Security section
  - Uncheck the box next to "Check for publisher's certificate revocation"
  - Uncheck the box next to "Check for server certificate revocation"
  - Uncheck the box next to "Check for signatures on downloaded programs"
3. Click OK
4. Restart your computer

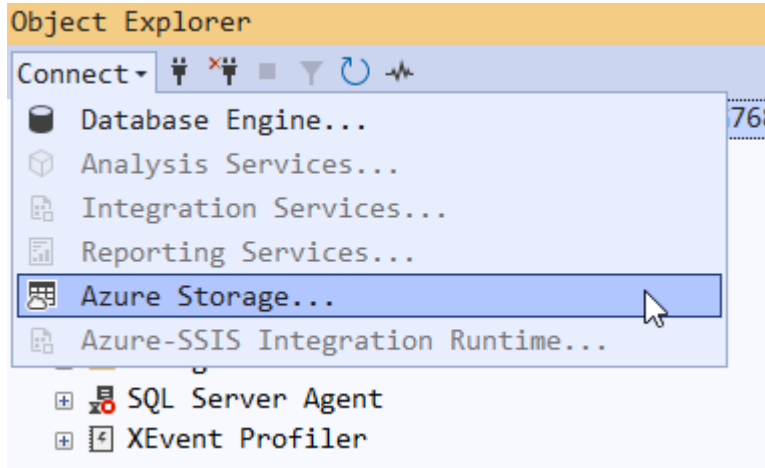
You might also be able to do it via your browser security settings.

**Keep in mind that if you disable this, it applies to all checking of certificates on the machine. As I said, if in doubt, don't do it.**

## 7.9 Connecting to Azure Storage and other services

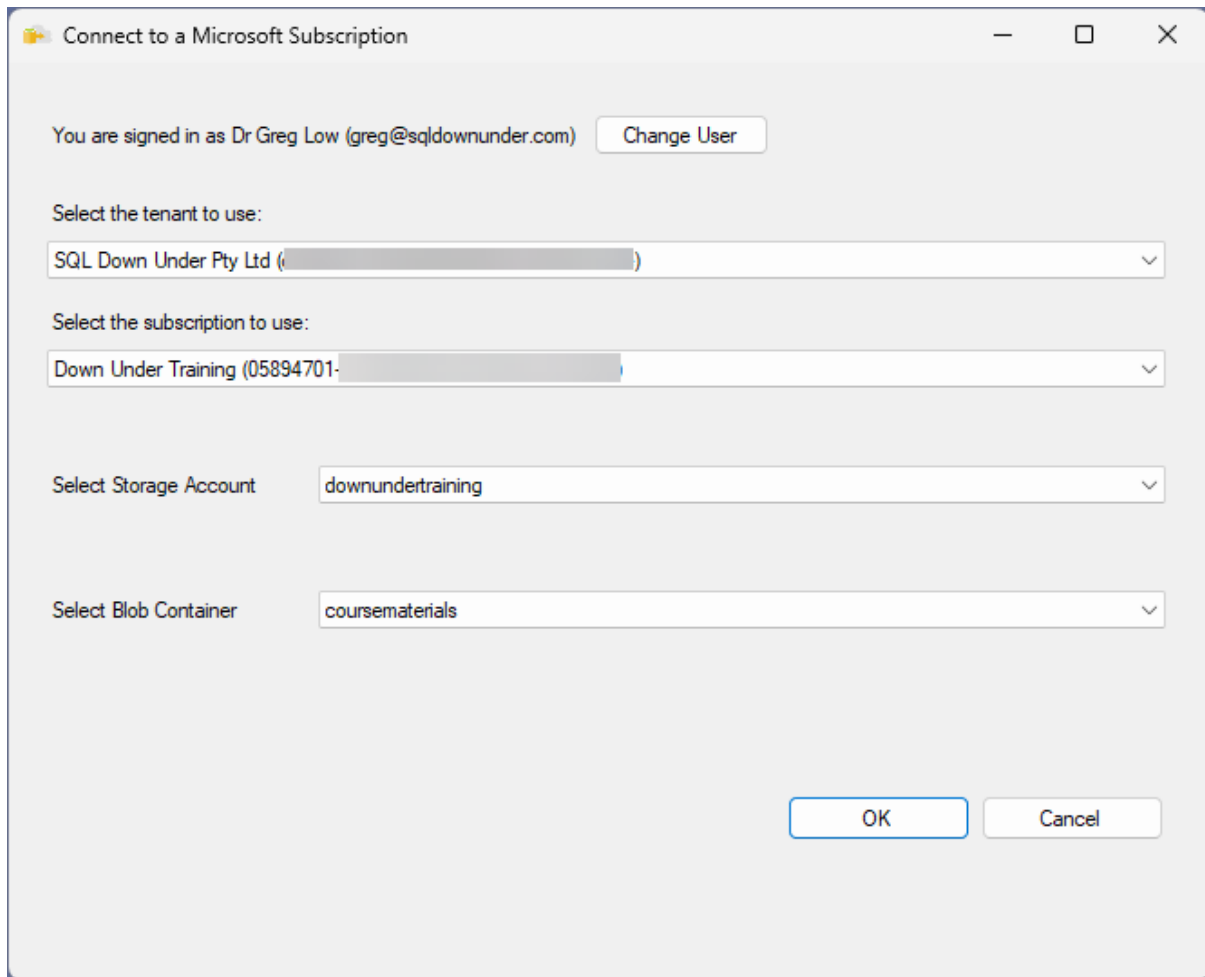
SSMS is a great tool for working with SQL Server relational databases but it can do much more than that.

In Object Explorer, note that you can easily connect to other types of services:

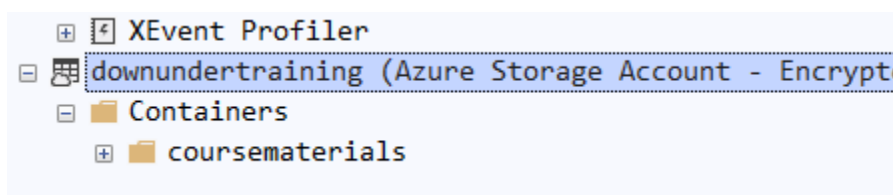


For a long time, it has been able to connect to Analysis Services to manage SSAS databases, both tabular and multi-dimensional. It can connect to Integration Services but that's to the older style interface for SSIS. Nowadays, you should use the SSIS Catalog instead. There are a few items that you can configure via the Reporting Services connection as well.

One option that is often quite unexpected though, is that you can connect to Azure Storage. Here's an example. After I click Azure Storage, I'm prompted for the subscription and container details.



From there, once the connection is made, you can drill into the containers and their contents:

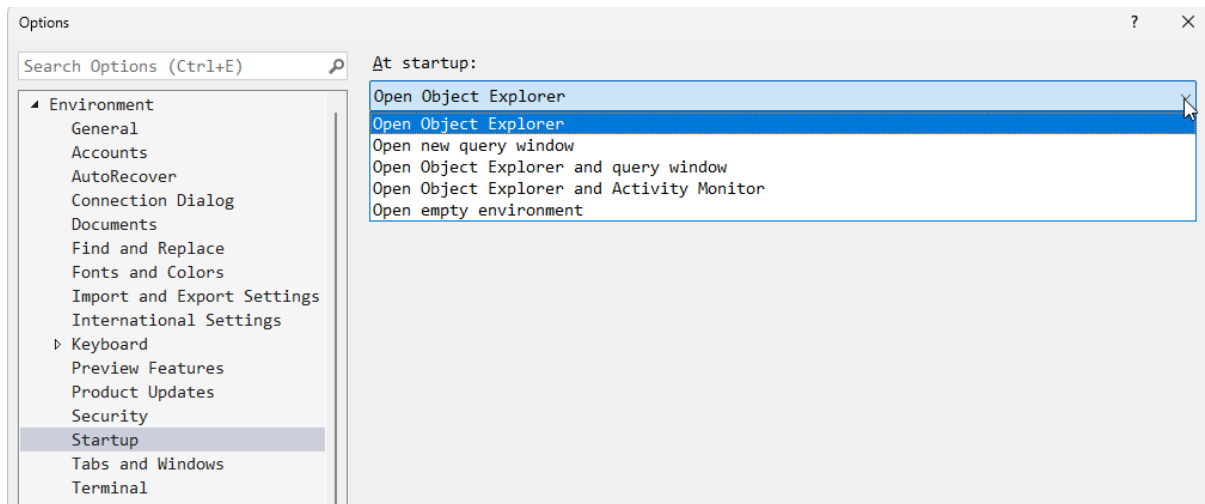


Generally, I would use **Azure Storage Explorer** to manipulate these storage accounts, but this option can be useful if you are using BACKUP TO URL and you need to just check the backup files that you are creating.

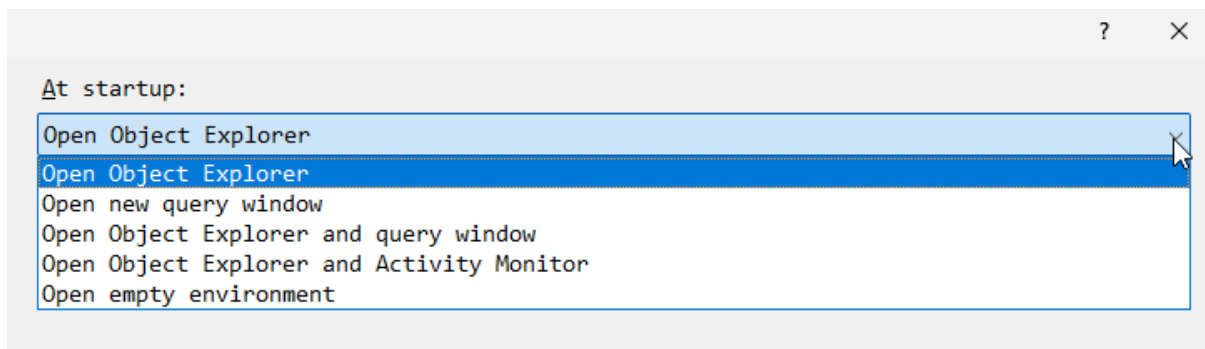
## 7.10 Setting startup options

When you start SSMS, the default action at startup is to open Object Explorer. But you can change that behavior.

The options to do that are in Tools. Options, then Environment, and Startup.



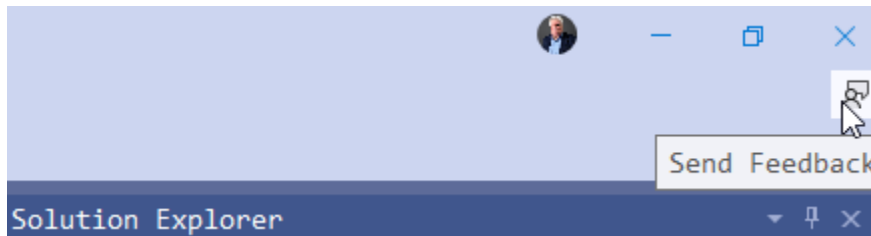
These are the options that are provided:



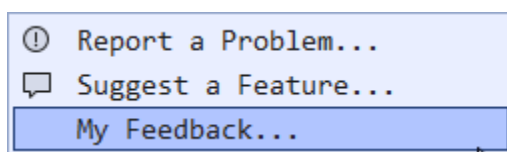
One that is often surprisingly useful is to open an empty environment. You might want to use SSMS to edit files without any connection to a database and not want to waste time waiting for SSMS to open the connection dialog, just for you to close it again..

## 7.11 Providing feedback directly to the SSMS team

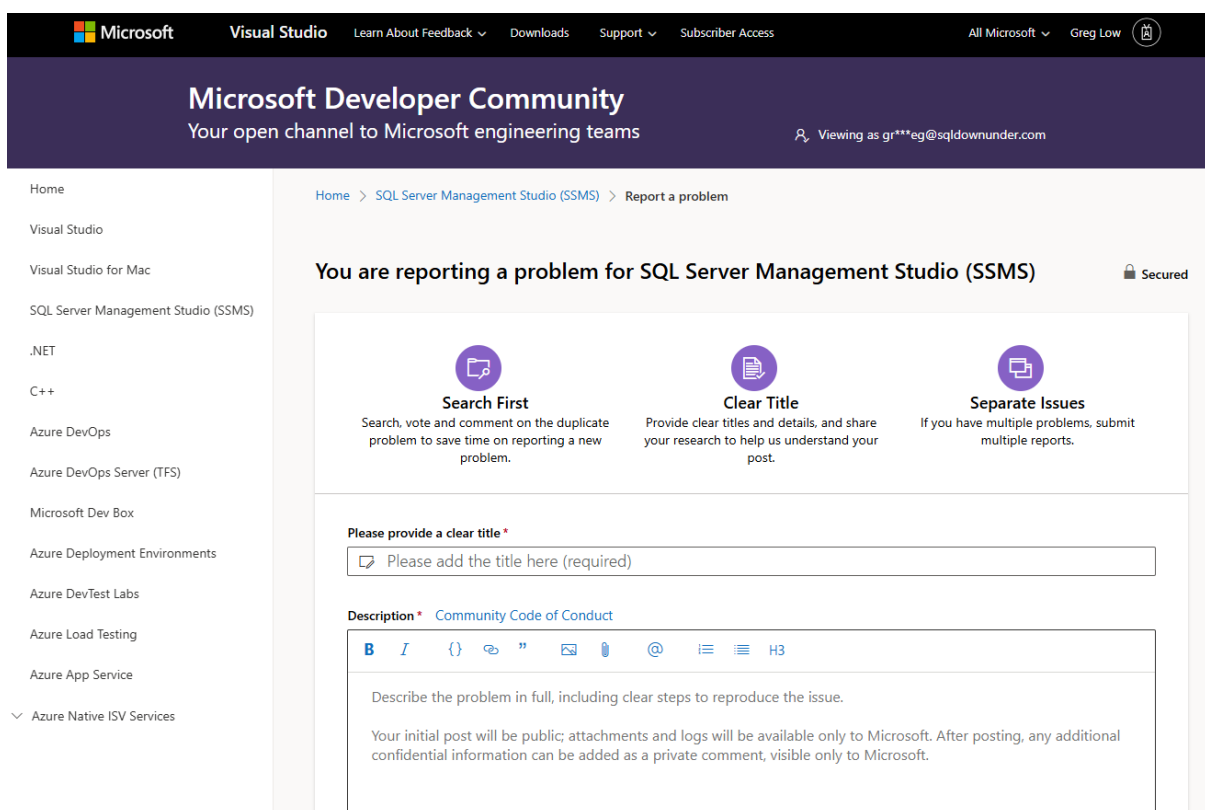
In the current version of SSMS, there is an option to provide feedback directly to the SSMS team. While you can just browse to the website, you can click this link in the top right-hand side.



At that point, you are given three options:



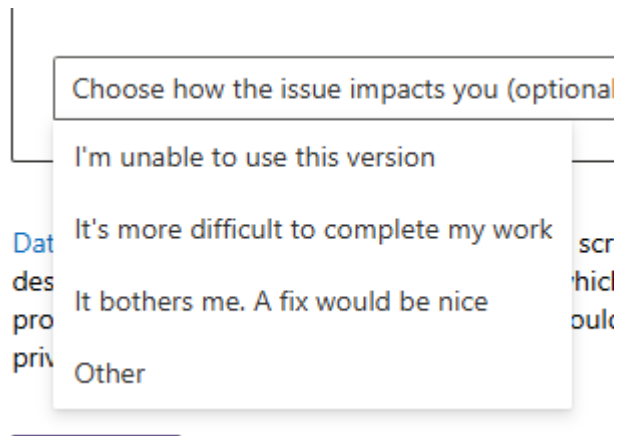
The first two end up in very similar locations, just with a different preselected context:



Make sure you search first to see if the problem has already been reported. As soon as you provide a title for your issue, the site does a search and shows you issues that are potentially related. If you see the same issue, you can click on it to go to the discussion and add your own comments there.



If the issue hasn't already been reported, you can create a new issue. Make sure you provide a clear title, then detailed information about how to reproduce the issue. Finally, before submitting, make sure you indicate the impact you are experiencing:



The image shows a screenshot of a GitHub issue form. A dropdown menu is open, displaying the title 'Choose how the issue impacts you (optional)' and four options: 'I'm unable to use this version', 'It's more difficult to complete my work', 'It bothers me. A fix would be nice', and 'Other'. The background shows parts of the form fields for 'Data', 'des', 'pro', and 'priv'.

Choose how the issue impacts you (optional)

- I'm unable to use this version
- It's more difficult to complete my work
- It bothers me. A fix would be nice
- Other

## 7.12 Accessing Preview Features

SSMS has a concept of channels that are used to control releases of the product. You can find them described here <https://learn.microsoft.com/en-us/ssms/install/channels-release>

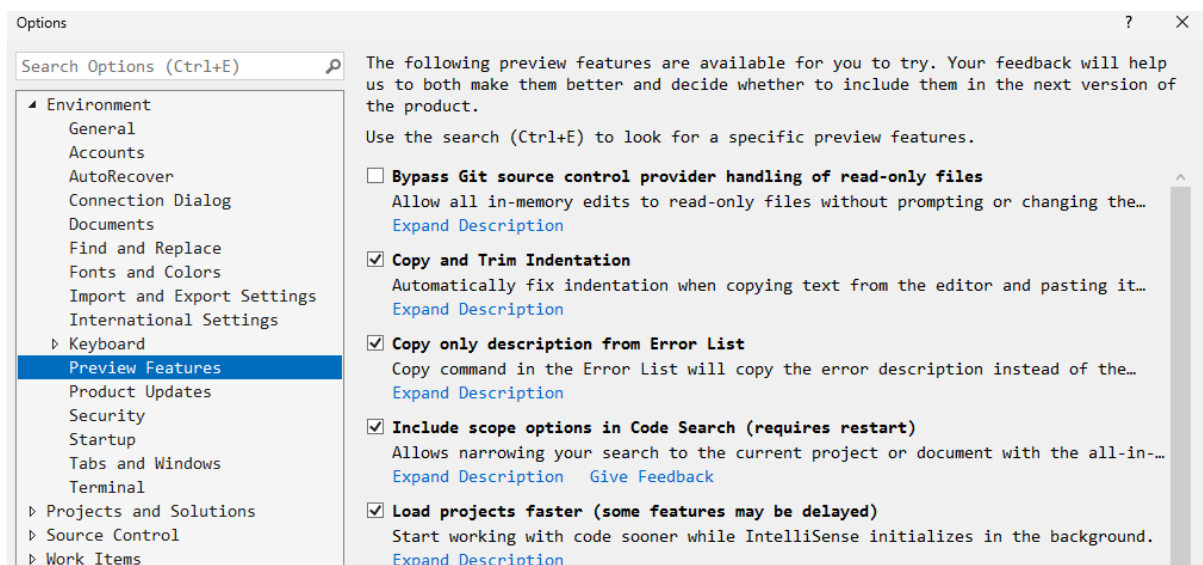
You can have both Release and Preview channels installed on the same system.

Most people who are using the product in production scenarios want to have a supported stable release. They should use the Release Channel.

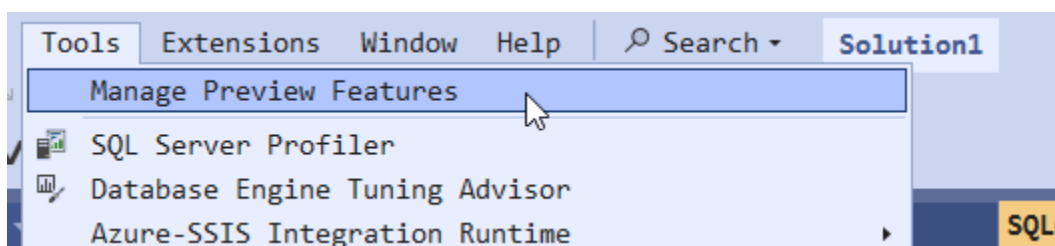
Other people, however, are keen to try the latest features as soon as they are available, even if they aren't at a Release quality bar yet. They can use the Preview Channel.

### Preview channel vs preview features

Users who have installed SSMS using the Release channel still have the option to try out Preview features. The options for this can be found in the Tools, Options, Environment, Preview Features settings. At the time of writing, some of the options available are shown:TODO



A shortcut way to get to these settings is also available from the Tools menu:



## 7.13 Working in both English and another language

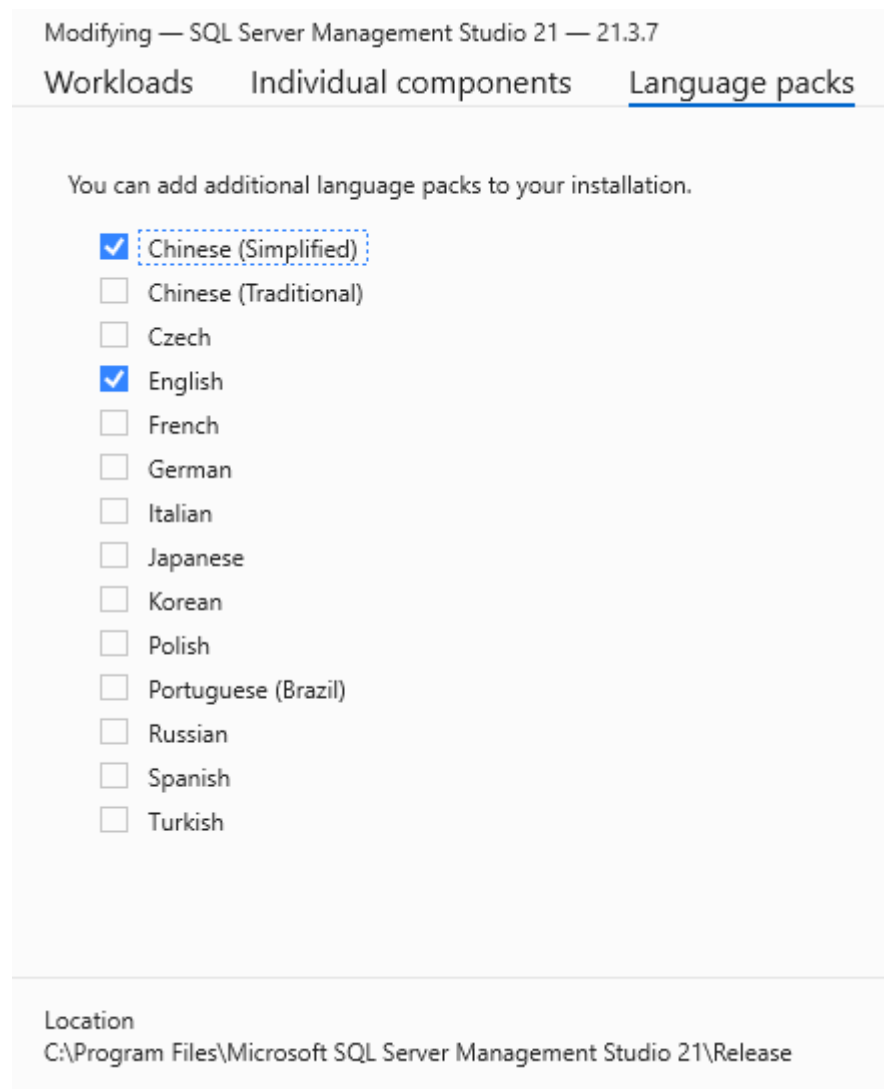
I've been learning Mandarin (Chinese) for many, many years. I'm determined to become as fluent in speaking as I can, but to also be reasonable at reading and writing.

One important aspect of doing this, is to make sure you use the language you are learning as much as possible.

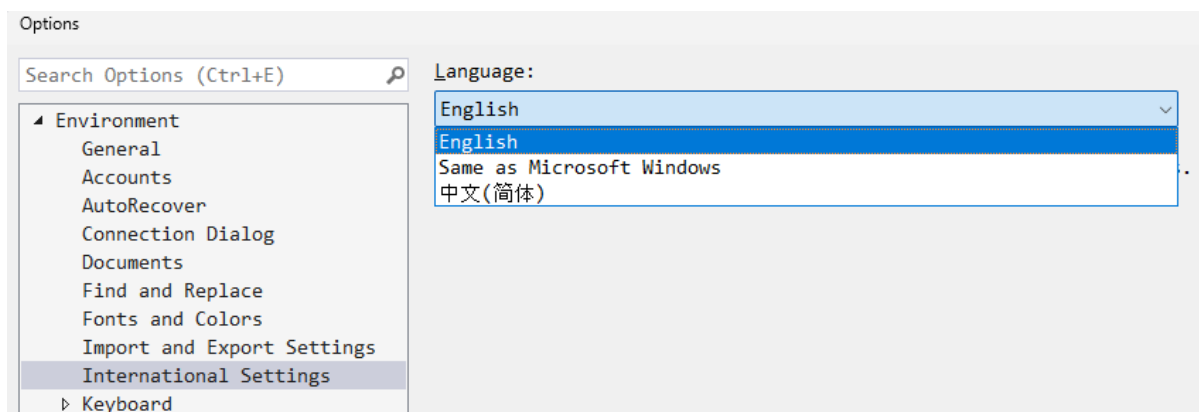
A great advantage of SSMS is that it is available for use in many languages. Yet when you do this, SSMS ends up supporting both the target language and English. That means that I can work part of the time in Chinese (to force myself to learn) and the rest of the time in English.

In earlier versions, the way you did that was to install from a different language-specific installer file.

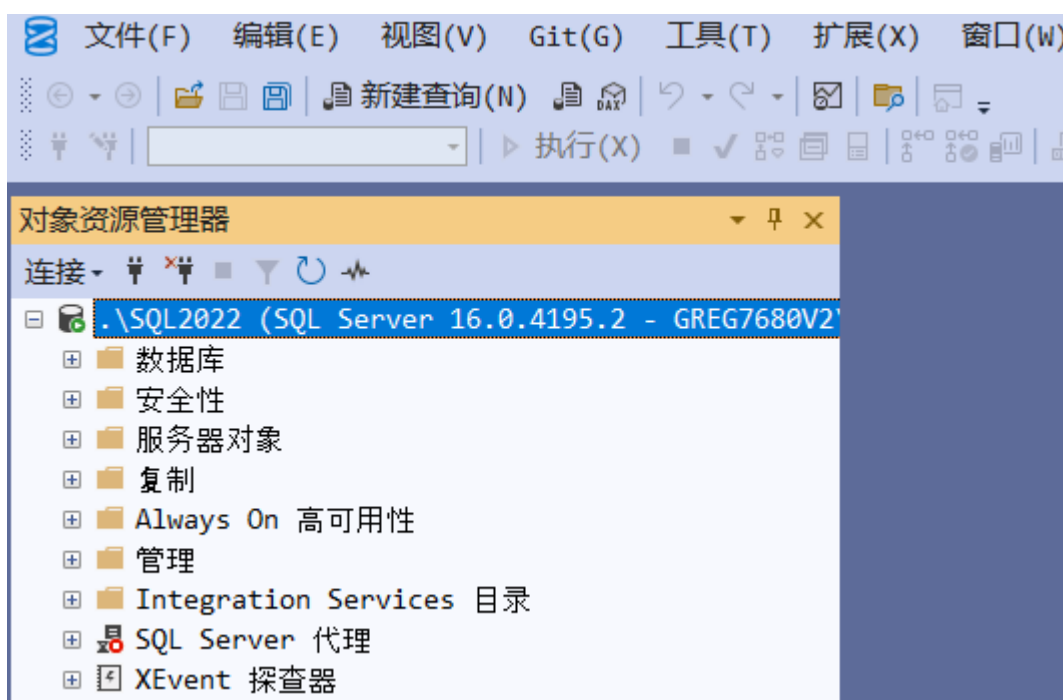
From v21 onwards, to install SSMS in an additional language, you need to choose the Language packs tab on the Visual Studio Installer.



Once you restart SSMS, you can choose the language you want to have displayed, from the Tools, Options, Environment, International Settings menu:



And once I choose Simplified Chinese and restart SSMS, I can now work in that language but choose to return to English at any time:

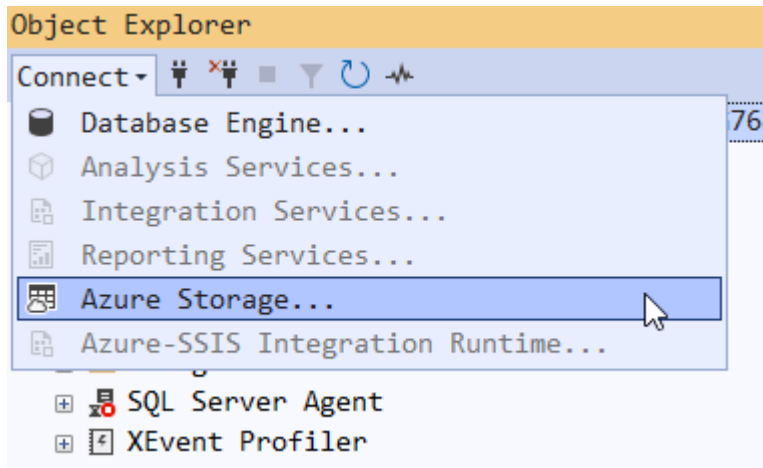


A beautiful aspect of this now is that I can install many languages concurrently, not just two.

One other thing to note is that in the Tools Options list, all the options are presented alphabetically. So, you might be surprised that the order of the options is different in a different language. I know I was. It took me a moment to realize why I couldn't find things where I expected them to be, based on their location.

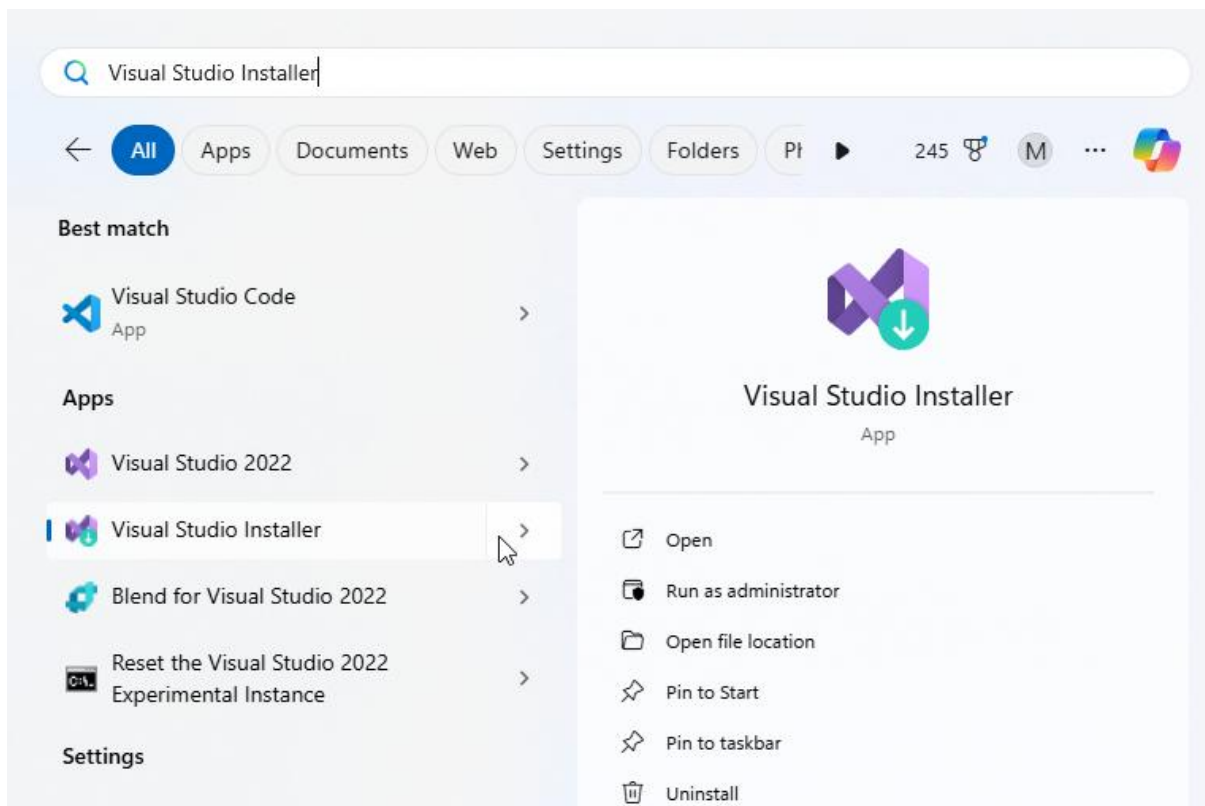
## 7.14 Adding support for Analysis Services, Integration Services, and Reporting Services

When I first installed v21 of SSMS, I was puzzled that many of the connection options in Object Explorer were greyed out:

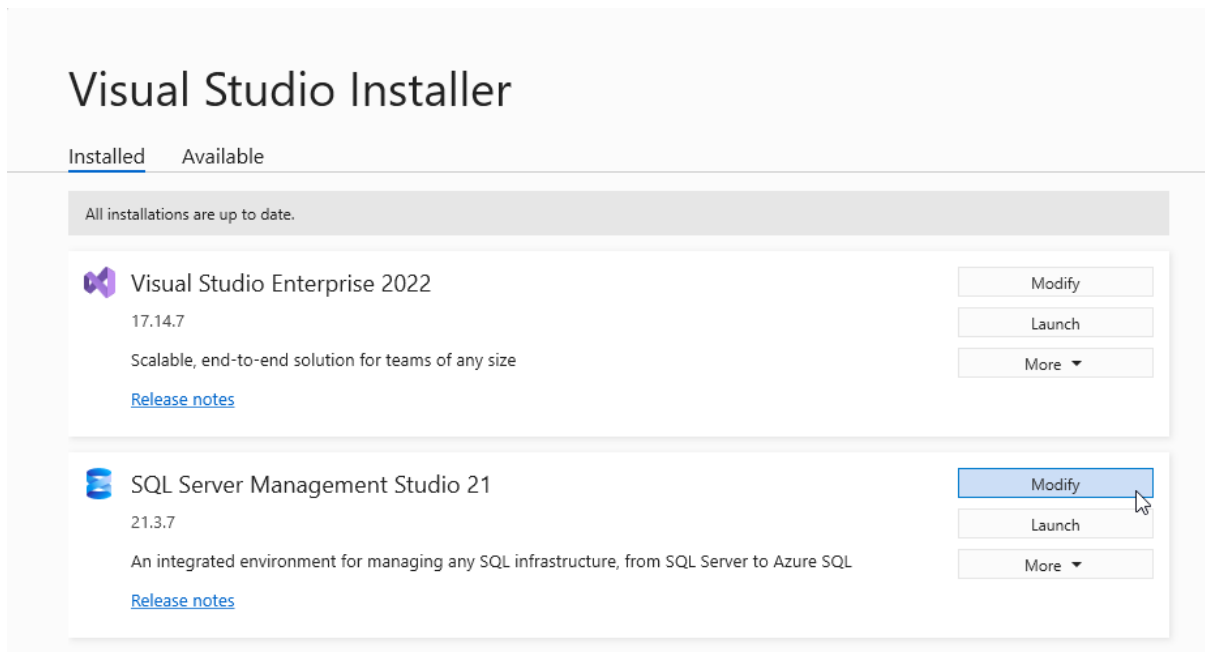


Because I had been working on a preview version, I thought it just wasn't there yet. But when I checked the release notes, I saw it should be there. The issue is that I hadn't added the Business Intelligence workload.

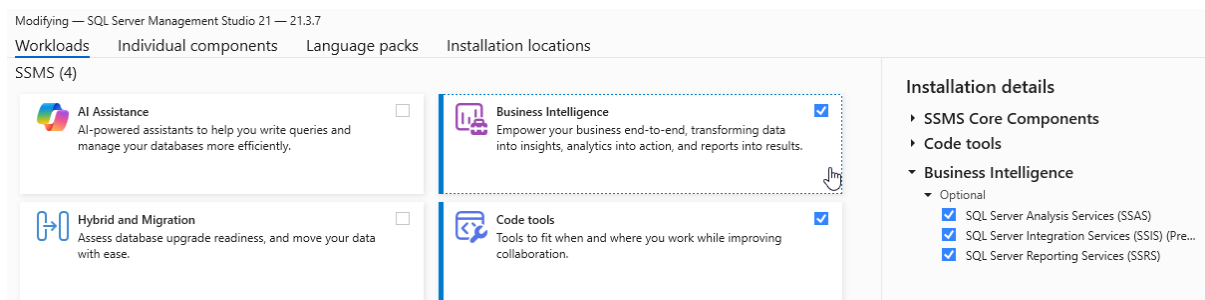
You don't do that from within SSMS. Instead, you need to run the Visual Studio Installer:



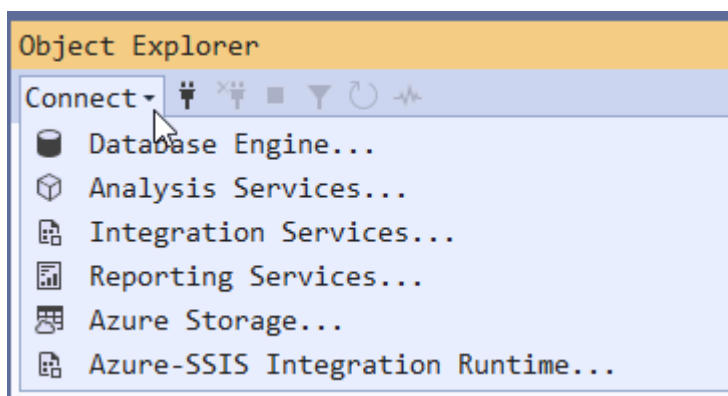
When it opens, you need to modify the SSMS installation:



Make sure the Business Intelligence workload is installed:



And on restart, we're back to the options we expected:



## 7.15 Disabling the Open Transaction Check when Closing SSMS

When you close SSMS, the default action is that it checks each open window and makes sure there are no open transactions before it closes.

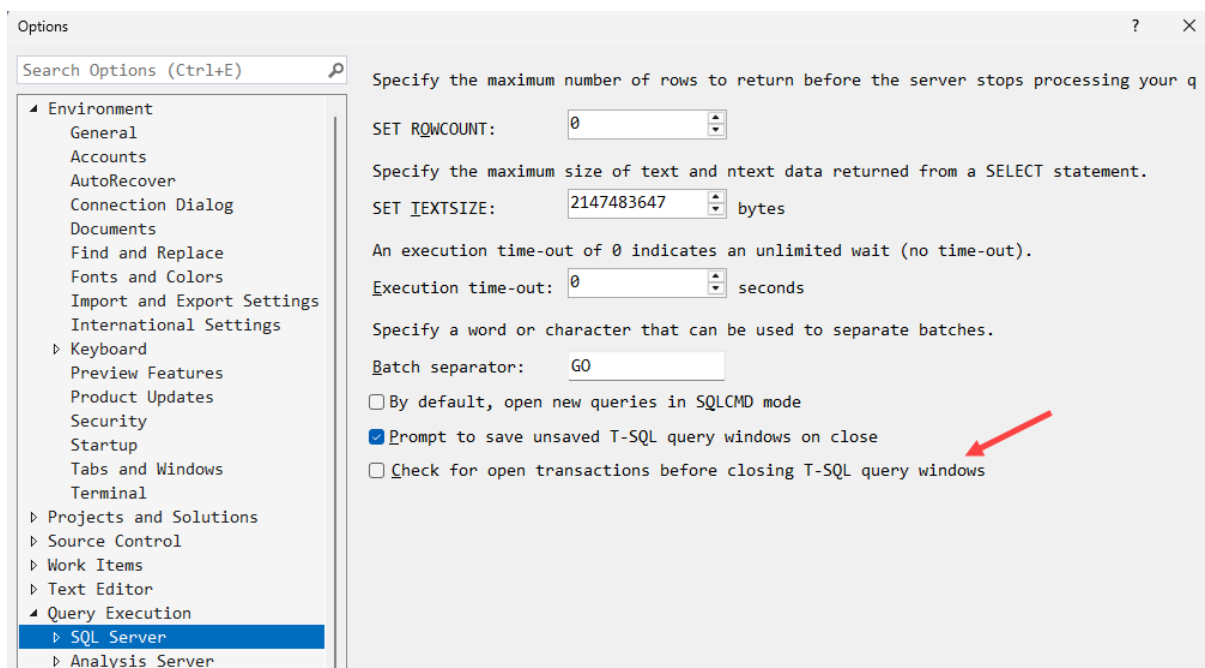
This is to allow you to decide what to do if you do have an open transaction, and based on that, to avoid losing any work.

While this sounds like a great idea, I mostly now work with Azure SQL Database, and my SSMS windows have been idle for long enough that I've lost the connection. So, when I close SSMS, it tries to check the transaction status against connections that are already closed under the covers.

This makes SSMS appear to hang for a while before closing, and sometimes, to even pop-up windows or throw errors.

I'd rather it didn't do this at all.

And I can disable it. In Query Execution, SQL Server, there is an option to avoid this check.



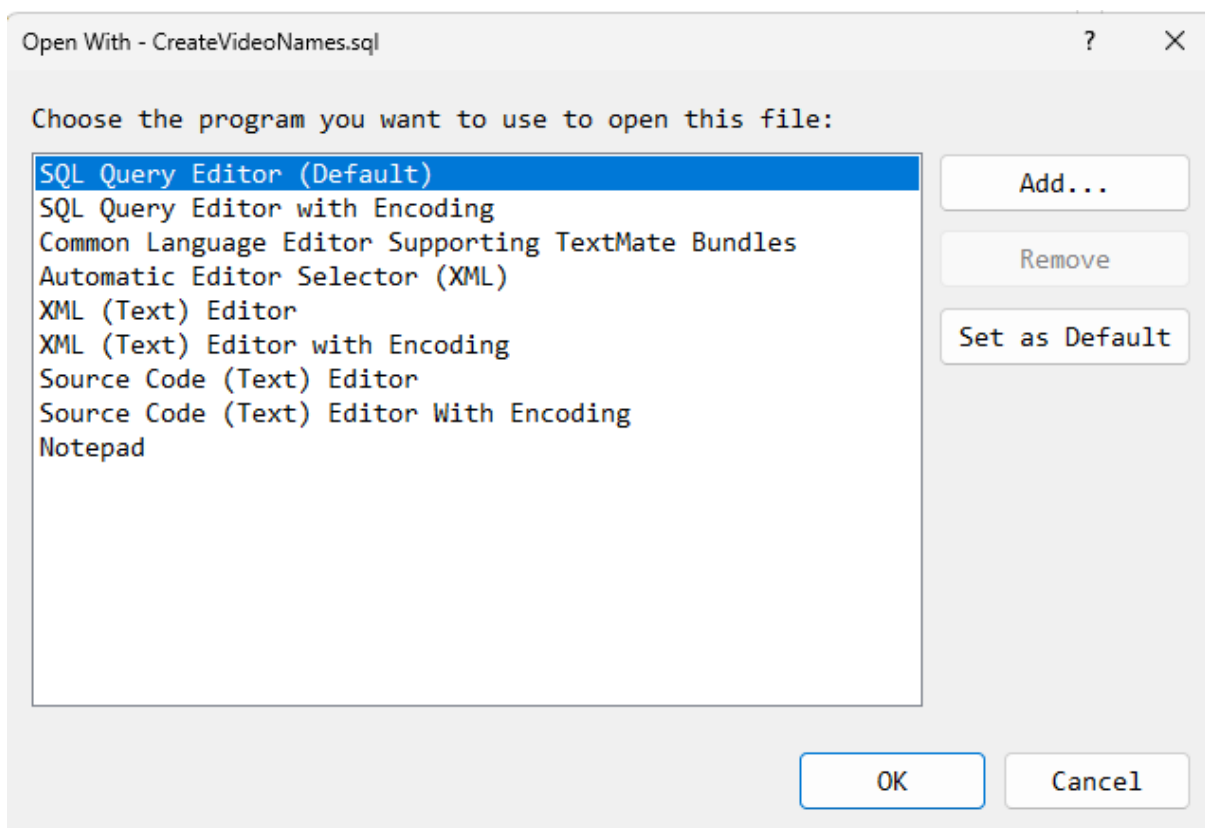
## 7.16 Using encodings when opening or saving files

SSMS has the ability to work with different encodings when you are opening or saving files.

When you use File and Open, note that beside the Open button is a drop-down arrow:

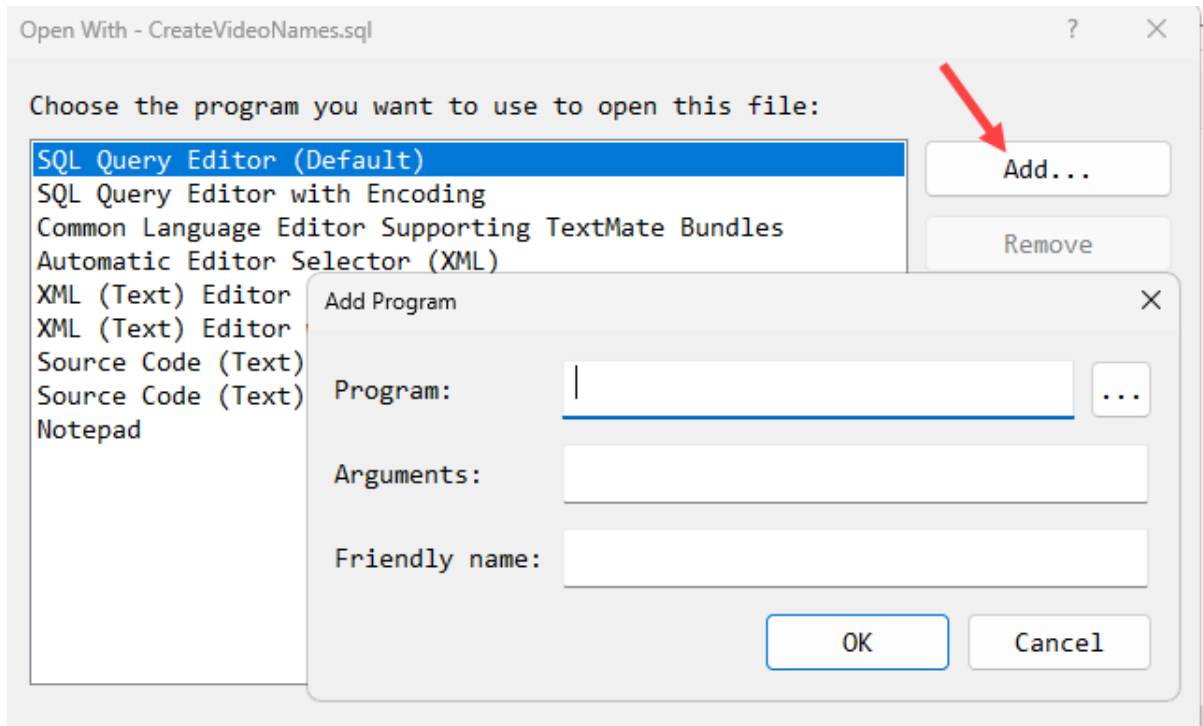


The Open With option leads to this:



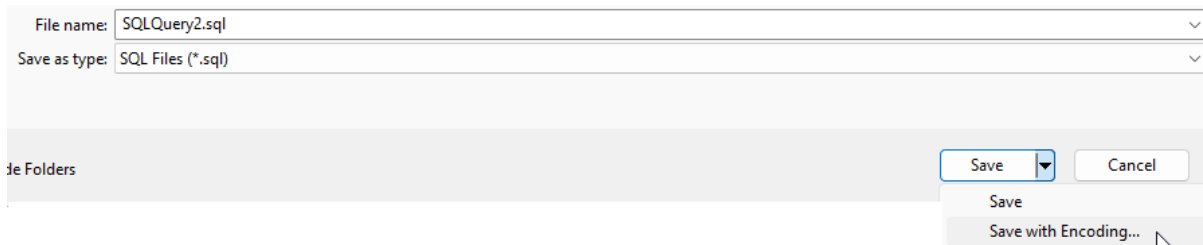


In that screen, you can choose an existing value or click Add to add another program that you want to use for opening files of that type:

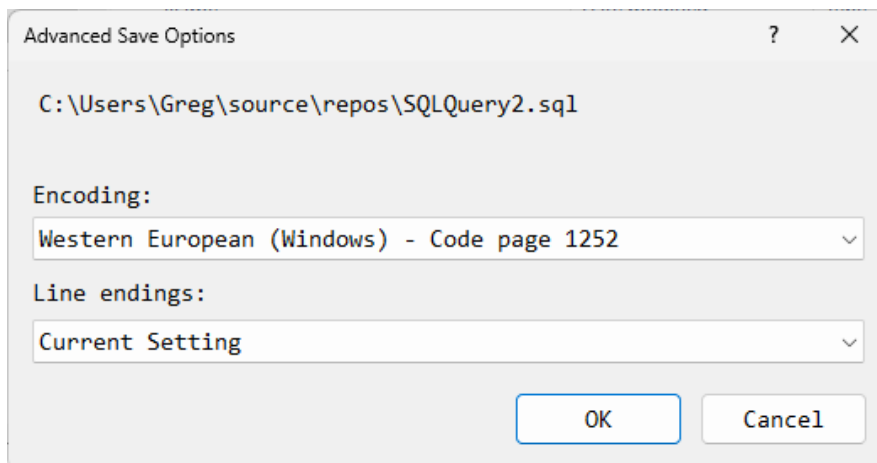


As an example, you might want to use a different XML or JSON editor.

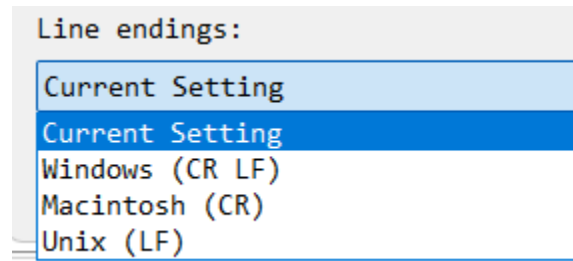
Similarly, when you want to save a file, there is another drop-down beside the Save button:



It leads to this:



In that window, you can choose the encoding to use, and you can also choose the line endings to apply:



You might want to save your file with just Unix line endings (i.e., just a line feed, and no carriage return).

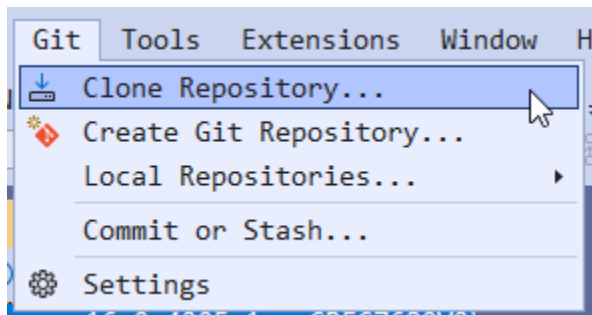
## 7.17 Git Integration

Early versions of SSMS included the ability to work with source control systems. The way that worked was that SSMS implemented an SCCI (Source Code Control Interface). It would let you connect to any source control system that implemented that interface.

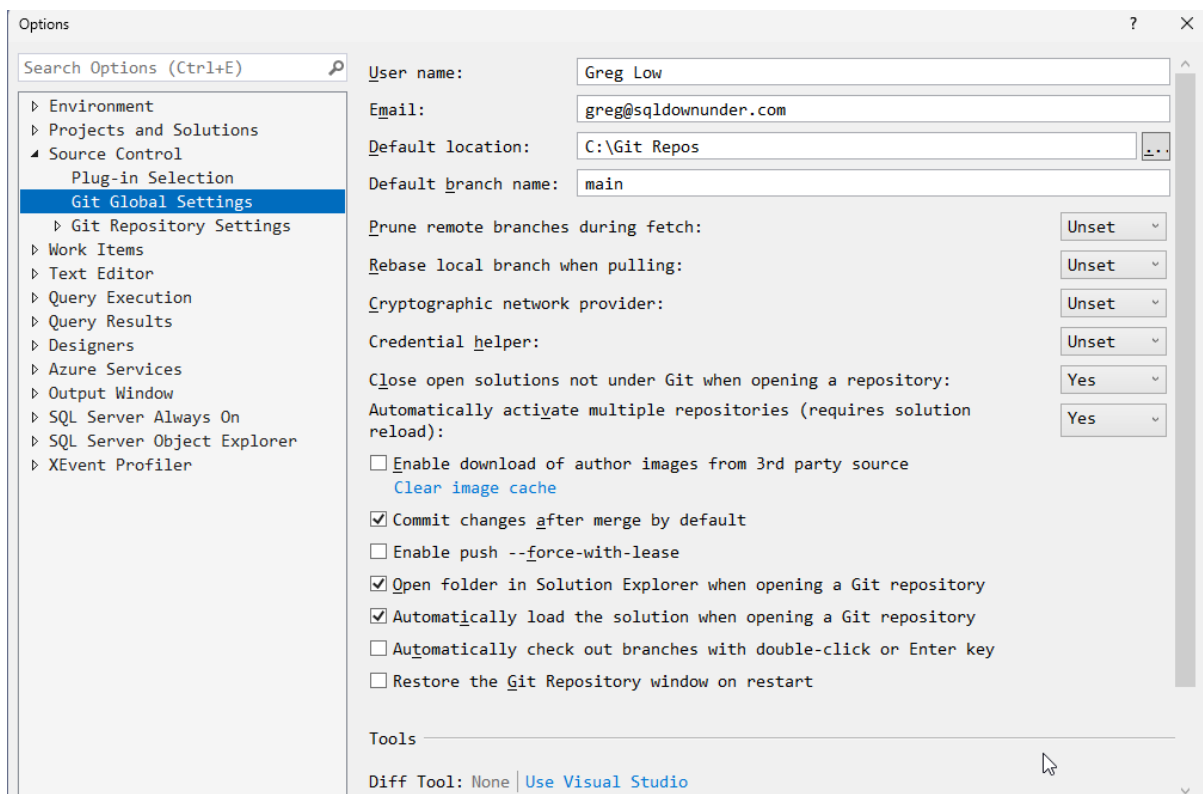
For a while, SSMS lost any ability to talk to source control. I thought that was a pity as I make extensive use of projects within SSMS and source control is where I want them stored.

Since then, the world has moved to Git. And, overall, that's a good thing.

SSMS now supports the Git integration that comes with the underlying Visual Studio implementation. To me, the options look pretty much identical to the VS options. There is a top-level Git menu:

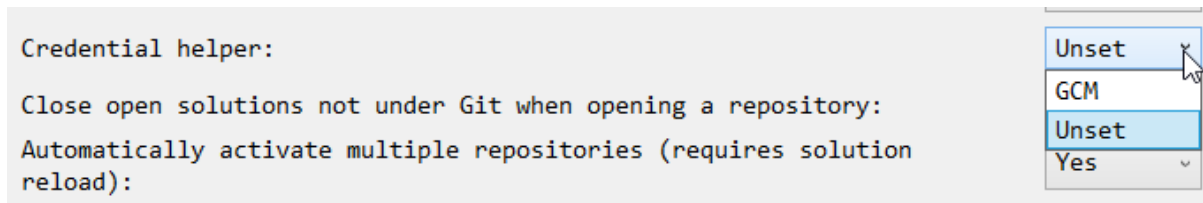


Before you start trying to work with it though, you need to go into the Settings area and configure your global Git options. You could also get to that from the Tools Options menus.



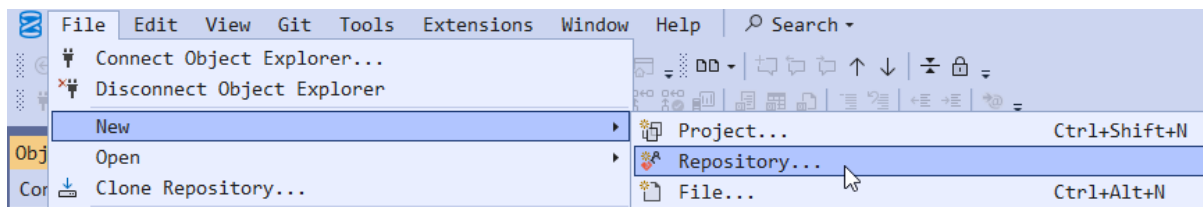
I found that I didn't need to set them up, as it just took the entries I already had in Visual Studio. If you haven't been previously using Visual Studio with Git, you'll need to set them up.

Most entries should be ok with default values. For some users, you might need to configure the GCM (Git Credential Manager) as your Credential helper:

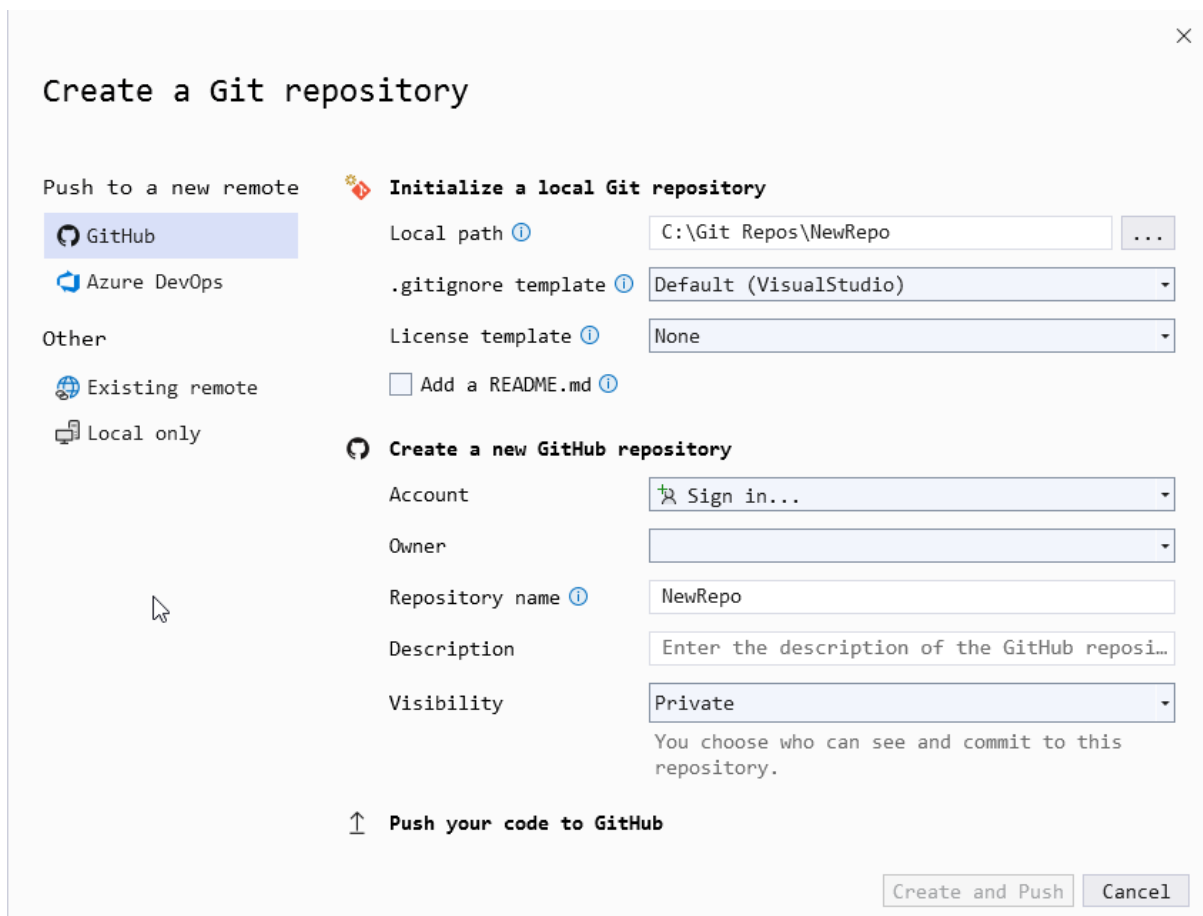


The configuration entries look like they'd work well with most flavors of Git.

There is also now a menu item on the File menu, for creating a new repository and for cloning a repository:



When you are creating or opening repositories, you'll notice that there is support for both GitHub and Azure DevOps, but you should also be able to get this to work with other flavors of Git:



If this is new to you, I'd suggest starting with Azure DevOps. They have a free license for up to 5 users. It is somewhat easier to use than GitHub once you get into CI/CD (continuous integration/continuous delivery). New users seem to find Azure Pipelines (part of Azure DevOps) easier to get started with than GitHub Actions. But both work fine. For repositories, there really isn't much difference between the two.

Generally, I tend to create repositories in GitHub or Azure Repos (part of Azure DevOps) first, and then clone them into SSMS and/or other tools. That lets me configure them the way I want, more than the options that appear from the client end.

## 7.18 Opening shortcuts

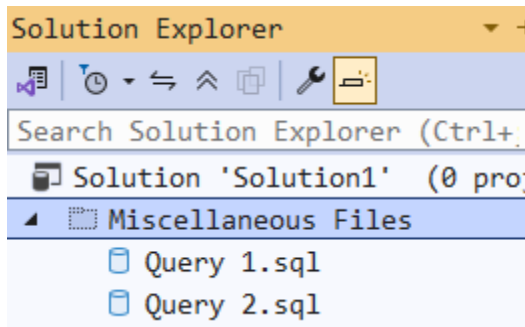
(Thanks to César F. for this one)

If you are working with many script files and they are all over your disk/storage, it can be painful to need to keep navigating whenever you want to open one.

However, when you use the File > Open > File option in SSMS, the dialog that opens up is capable of opening operating system shortcuts, and not just files. This means that you can have a folder of shortcuts and every time you need to open one of these files, you can just have it open the same location.

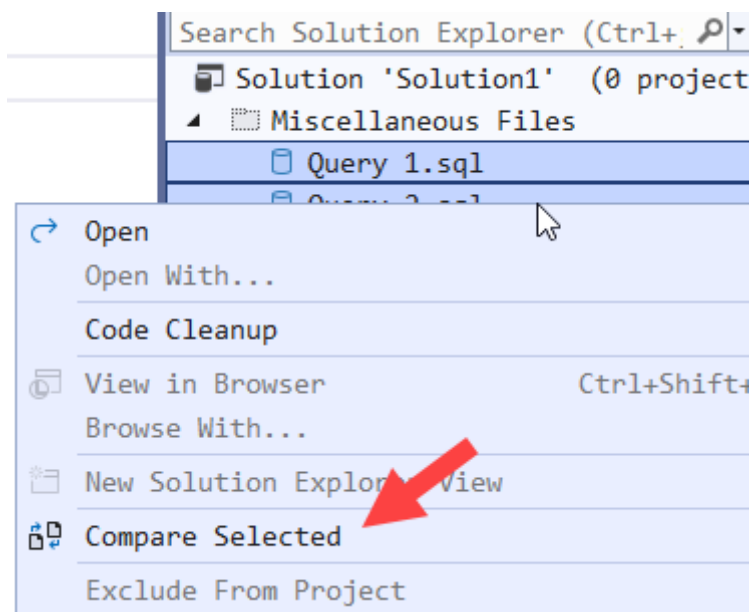
## 7.19 Comparing Scripts and Other files

When you create a new query file in SSMS, it automatically becomes part of a solution. You can view it in **Solution Explorer**. In this example, I have created two query files, and renamed the tabs without saving them, as **Query 1** and **Query 2**:

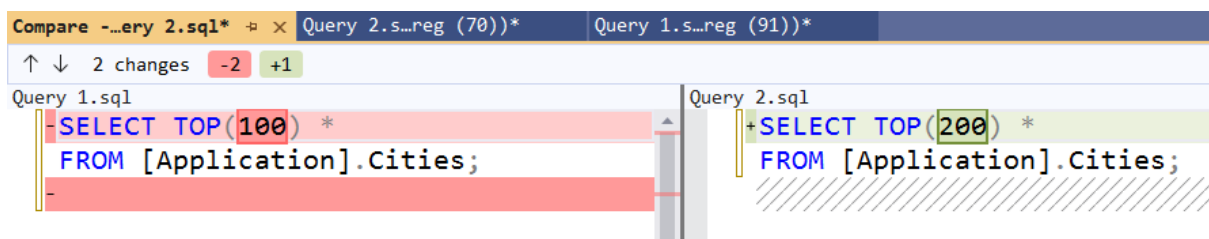


They are in the **Miscellaneous Files** section as they weren't part of a script project.

SSMS can now compare multiple files. If I highlight both the files, and right-click, I get this option:



And when I select it, I now get a comparison of the two files:





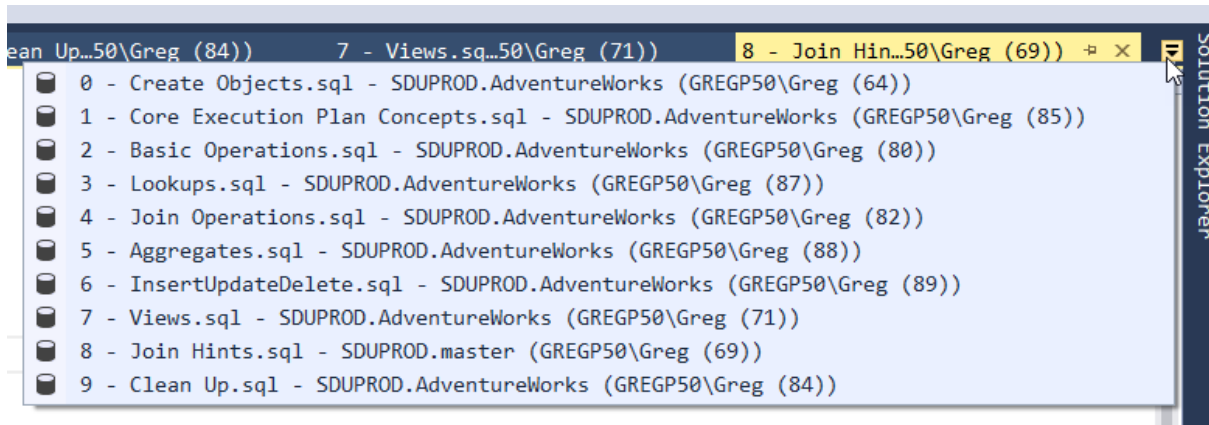


## 8 Window Tabs

### 8.1 Pinned Tabs

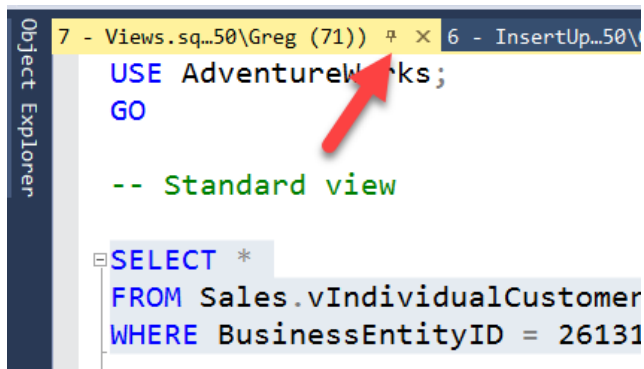
When you get to a large number of query windows or other documents open as tabs in SSMS, it can start to be difficult to keep track of them, and to find them when needed.

It's not too bad when you can immediately find the tab that you want in the drop-down list:

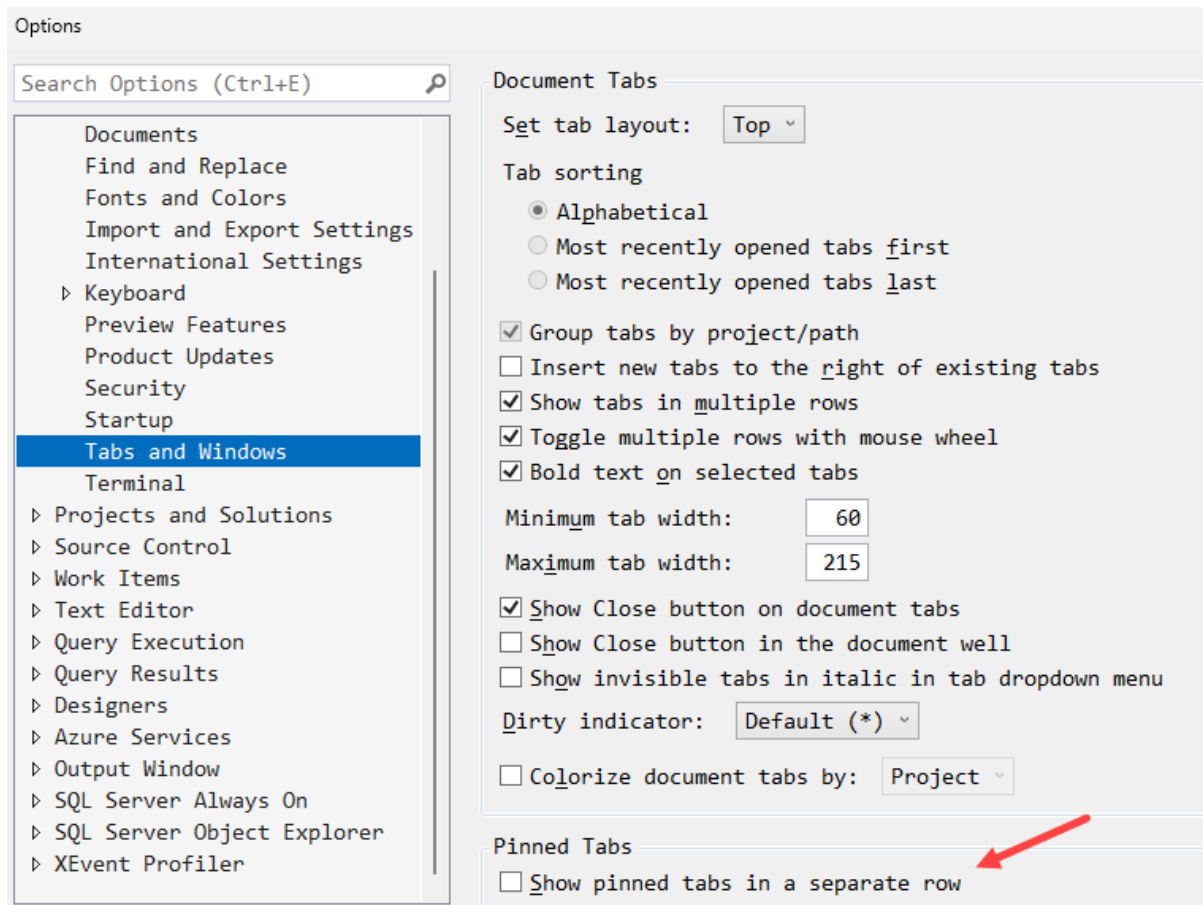


But if you have more tabs than are shown in this drop-down list or if, like me, you often end up with many of them without names (as they are temporary), it can get very hard to find the few that you are mainly referring to.

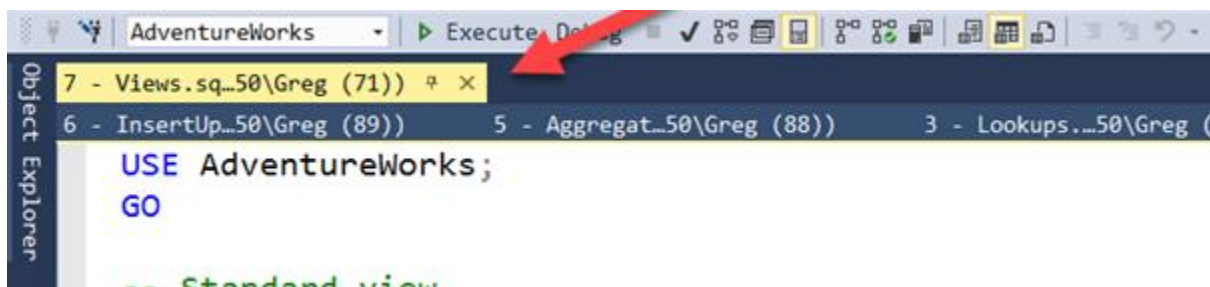
Just like you can with tool windows, you can pin tabs:



Once you do this, they stay against the left-hand side (by default). Now that's not bad but again if you have a few of them, there's another option that can help.



Once you configure that, another row of tabs appears in SSMS:



## 8.2 Reset window layout

One of the problems with applications that have highly-configurable user interfaces (UI) is that users can end up configuring them in ways they hadn't intended, and then don't know how to get back to where they were.

I remember the first time that I was at a session with a presenter from Microsoft showing the (at the time) new personalization options in ASP.NET. You could build a website and let the user determine how the site should be laid out, to suit themselves.

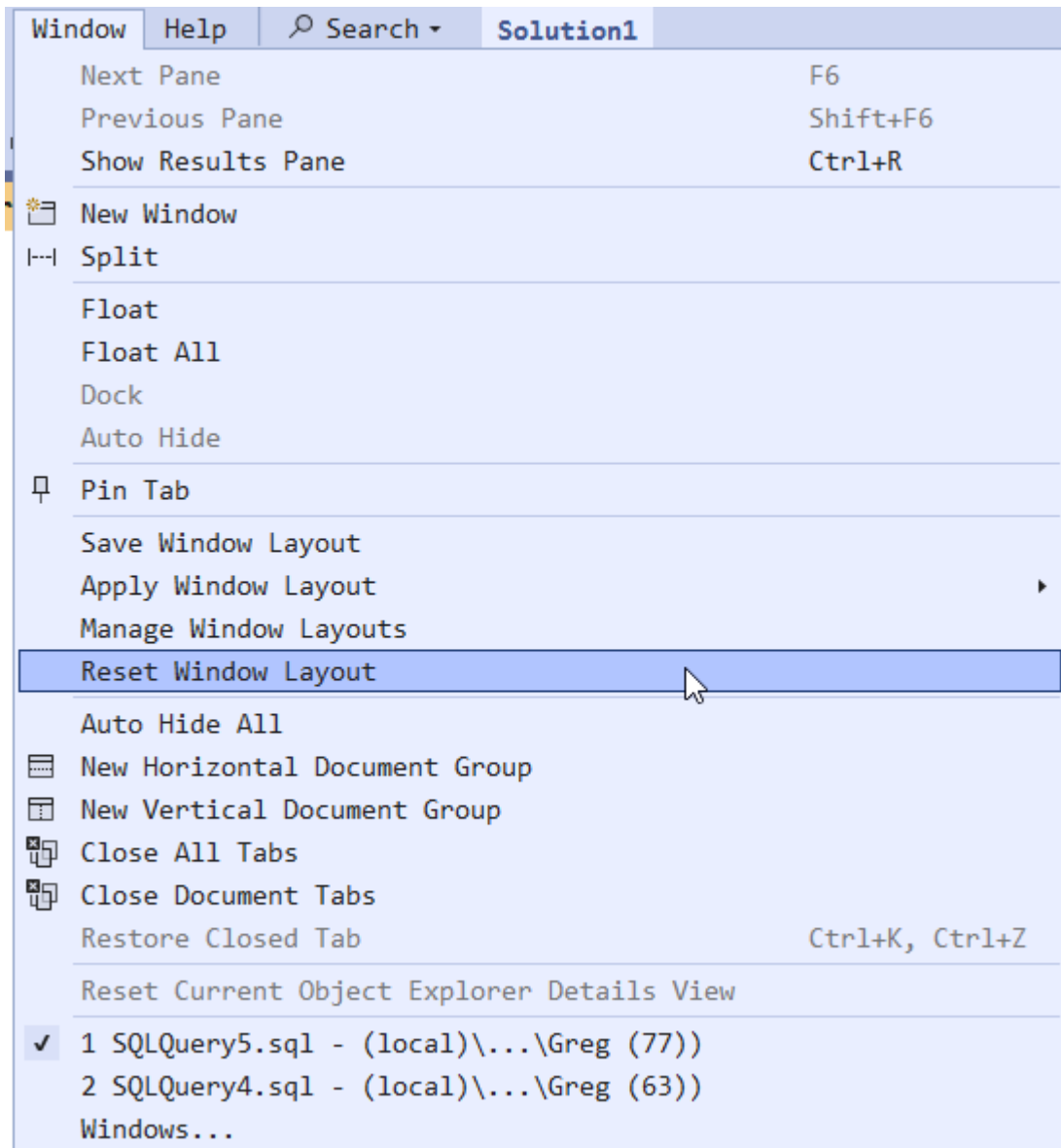
Overall, I can't say that I really like working with websites like that but I can understand the potential appeal. I can also easily see how end users could get really messed up.

I remember asking the presenter if there was a simple button that put the site back the way it was initially developed and removed the user's modifications, so that a user could always just get back to square one.

He told me **"ah no, there isn't an option like that"**.

I'm glad that the SSMS team don't think that way. While SSMS is very configurable, I have seen people get really messed up with the window management in it. They ended up dragging a window when they meant to drag something else, or did another action that changed their UI and it stuck. Then they don't know how to "fix" it.

In SSMS, there's a wonderful option in the Window menu, that does just what's needed:

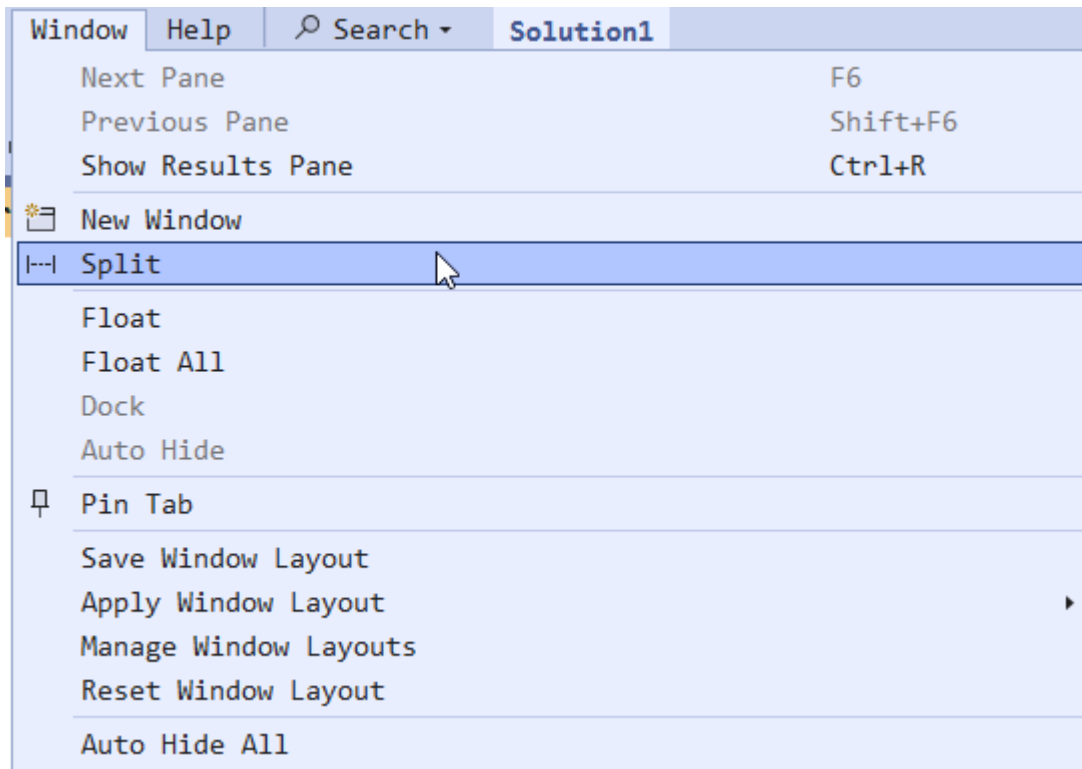


**Reset Window Layout** is the "get me back to where I was" menu item.

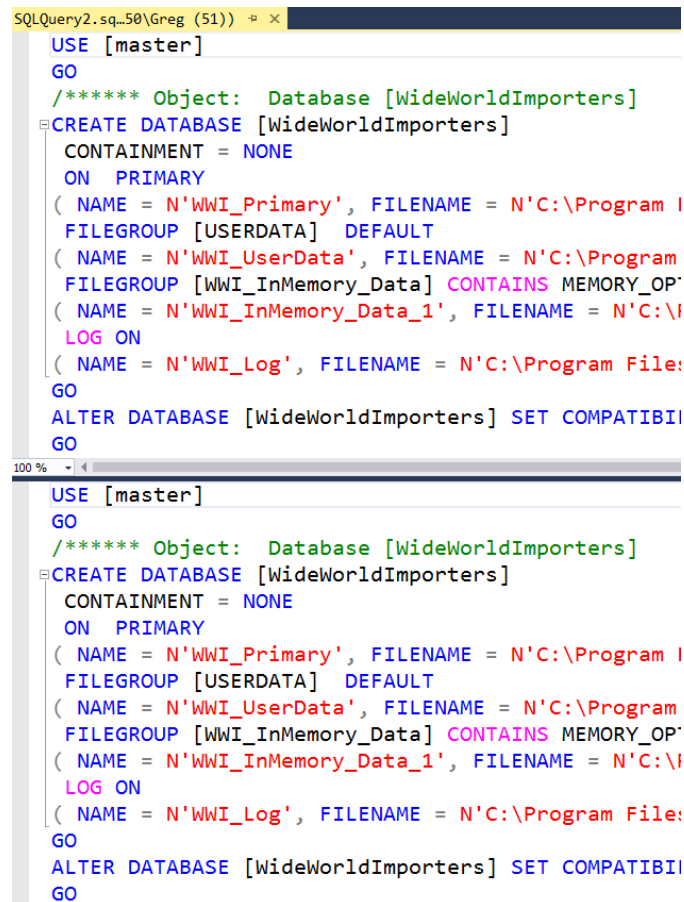
### 8.3 Using Split Screens

If you are working with really long script files in SSMS, you might need to work on more than one part of the script at the same time. Perhaps you need to work on a function, and also on the code that calls the function.

On the Window menu, there is a Split option.



When you first do this, you'll see a split window with the same query at top and bottom:



The screenshot shows a SQL Server Enterprise Manager window titled 'SQLQuery2.sql...50\Greg (51)'. It displays a split view with two identical panes. Each pane contains the following SQL script:

```
USE [master]
GO
/***** Object: Database [WideWorldImporters]
CREATE DATABASE [WideWorldImporters]
    CONTAINMENT = NONE
    ON PRIMARY
    ( NAME = N'WWI_Primary', FILENAME = N'C:\Program I
    FILEGROUP [USERDATA] DEFAULT
    ( NAME = N'WWI_UserData', FILENAME = N'C:\Program
    FILEGROUP [WWI_InMemory_Data] CONTAINS MEMORY_OP
    ( NAME = N'WWI_InMemory_Data_1', FILENAME = N'C:\I
    LOG ON
    ( NAME = N'WWI_Log', FILENAME = N'C:\Program File:
GO
ALTER DATABASE [WideWorldImporters] SET COMPATIBI
GO
```

You can then scroll each vertically and resize them independently, and work on different parts of the same script:

```
SQLQuery2.sql...50\Greg (51)
USE [master]
GO
/***** Object: Database [WideWorldImp
CREATE DATABASE [WideWorldImporters]
    CONTAINMENT = NONE
    ON PRIMARY
    ( NAME = N'WWI_Primary', FILENAME = N'C:
    FILEGROUP [USERDATA] DEFAULT
    ( NAME = N'WWI_UserData', FILENAME = N'
    FILEGROUP [WWI_InMemory_Data] CONTAINS
    ( NAME = N'WWI_InMemory_Data_1', FILENA
    LOG ON
    ( NAME = N'WWI_Log', FILENAME = N'C:\Pr
GO
ALTER DATABASE [WideWorldImporters] SET
GO

INCREMENT BY 1
MINVALUE -2147483648
MAXVALUE 2147483647
CACHE
GO
USE [WideWorldImporters]
GO
/***** Object: Sequence [Sequences].[
CREATE SEQUENCE [Sequences].[CityID]
    AS [int]
    START WITH 38187
    INCREMENT BY 1
    MINVALUE -2147483648
    MAXVALUE 2147483647
    CACHE
GO
```

The easiest way that I've found to close this, is to double-click on the dark bar in the middle, but there is also a Remove Split option in the Window menu.

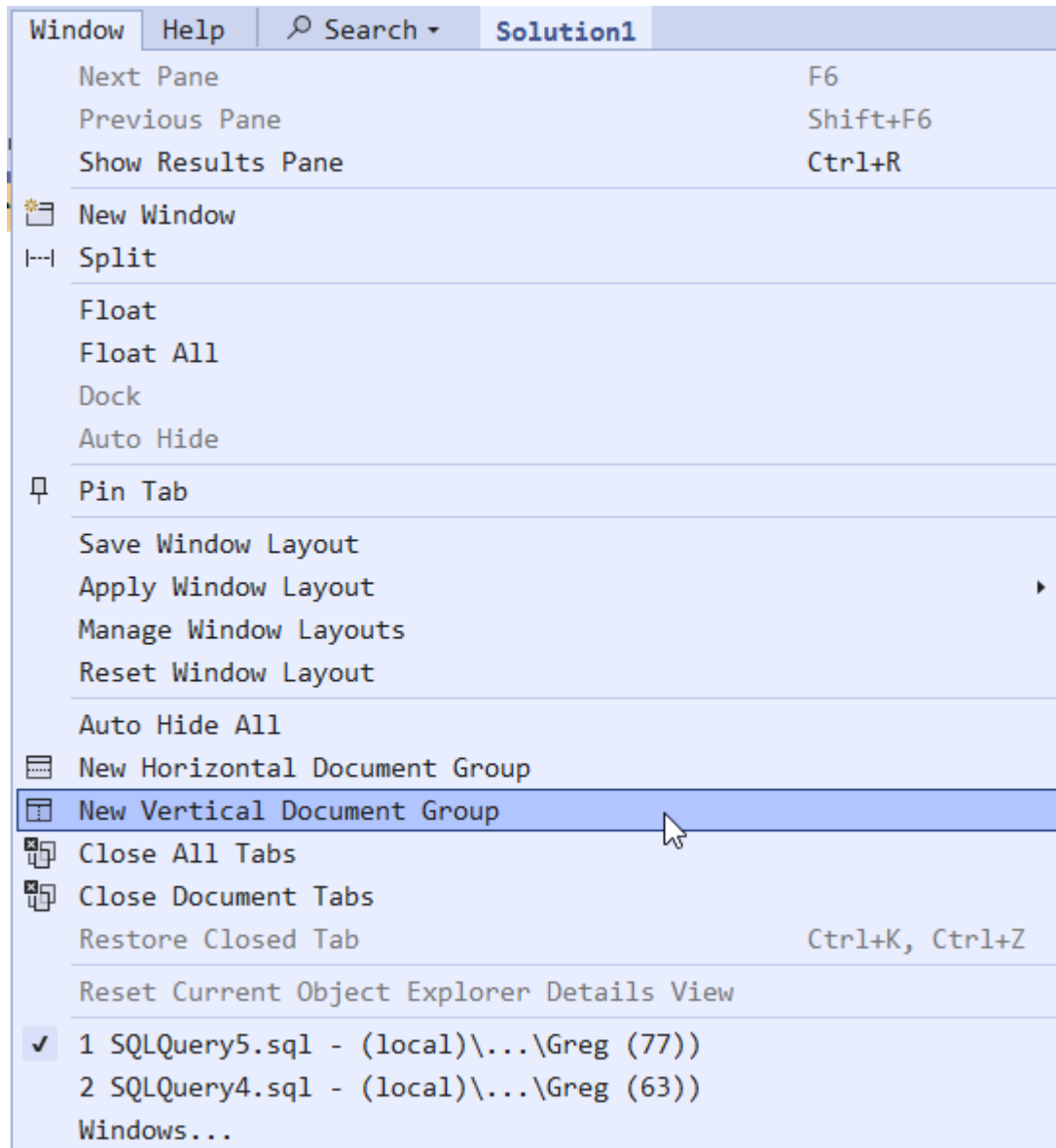
## 8.4 Document Groups

I previously showed how you might use split windows to allow you to work on different parts of a single query at the same time.

But what if you need to work on two queries and see parts of both of them?

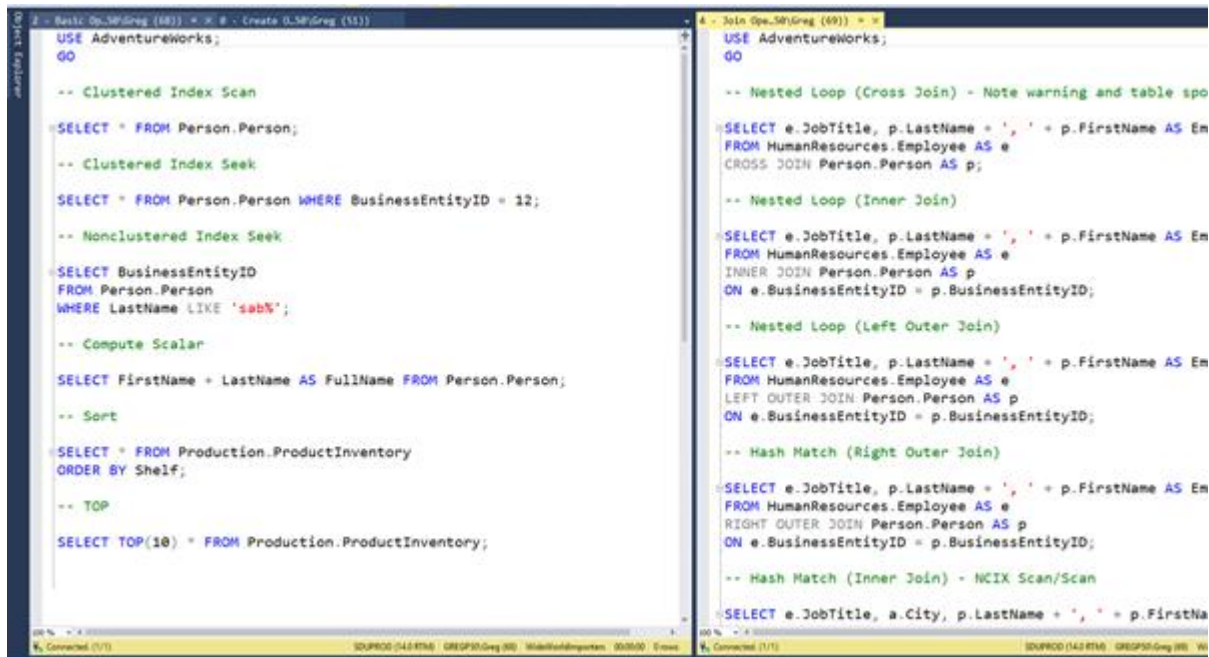
That's where document groups can help you. You can create both vertical and horizontal groups. For me, the most useful is typically side-by-side vertically, for when I'm comparing two sections of code.

From the Window menu, I choose New Vertical Document Group:



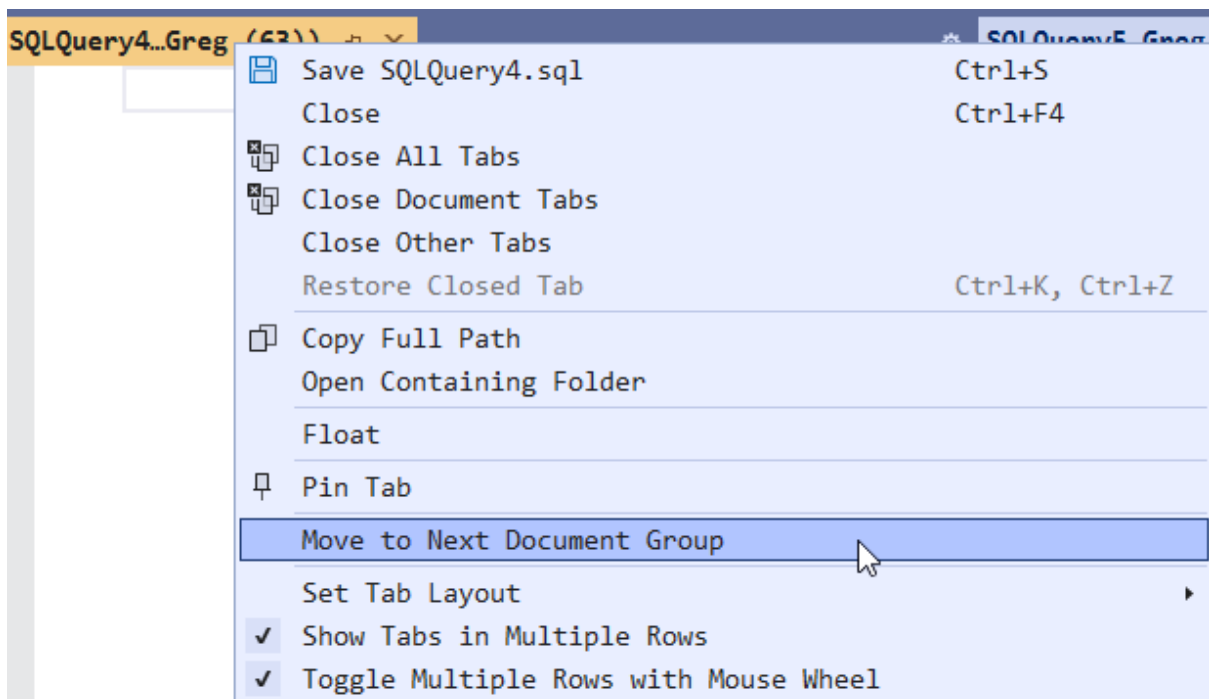


After I click that, as long as there are two or more queries open, I'll see this:



I then have two queries that I can work with, side-by-side. A horizontal document group places them one above the other.

In the image above, I had two queries in the left group, and one in the right group. You can also move queries between these groups. If I right-click the tab heading for one of the queries, I see this:



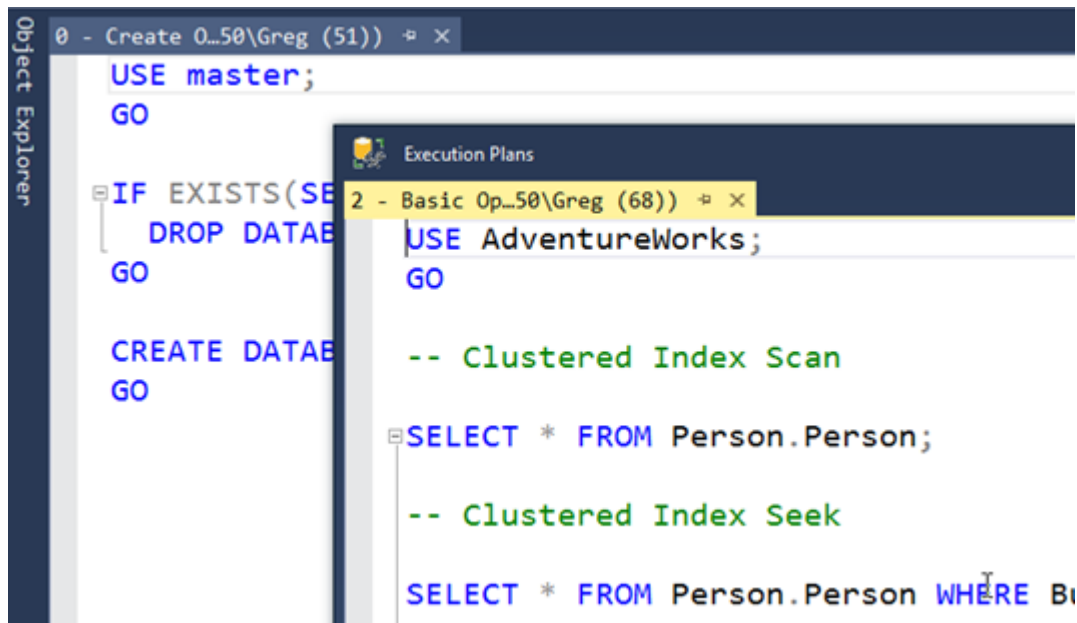
I can create yet another document group based on this tab, or move it to the next document group, or even close this document group by moving all tabs within it to the next document group.

## 8.5 Undock tabs and windows and using multiple screens

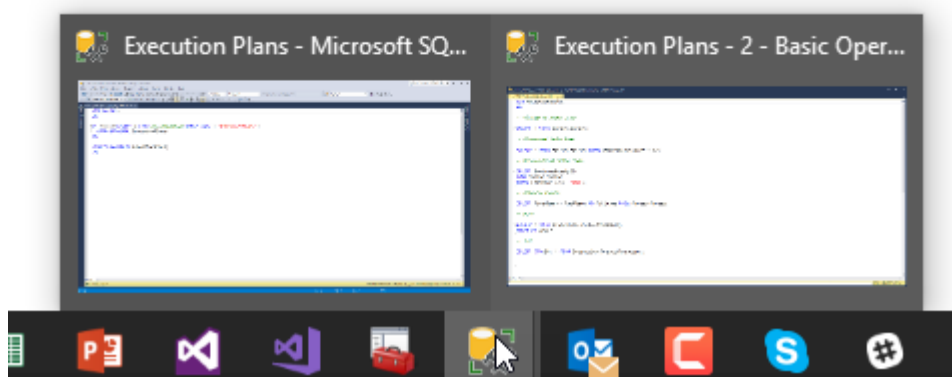
Like Visual Studio that it's based upon, SSMS is very flexible when working with query windows and tabs.

Most people realize that you can undock and move tabs and windows around. Usually they discover that by accident and then realize that the **Reset Window Layout** option in the Window menu is helpful.

But one option I've found that many people don't seem to realize is that you can undock just a single query window and move it outside the bounds of SSMS. You can even place it across on another screen if you have multiple screens.



It then also appears separately in your taskbar in Windows:

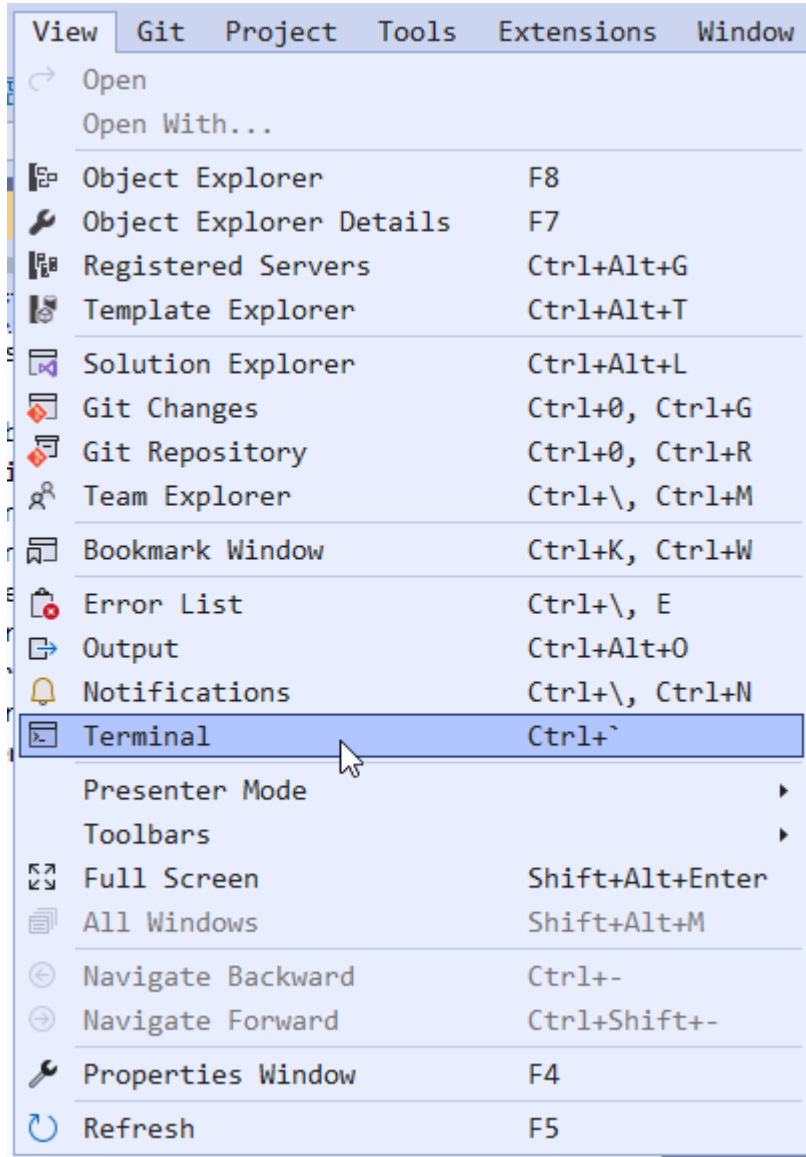


You can later drag it back to redock it.

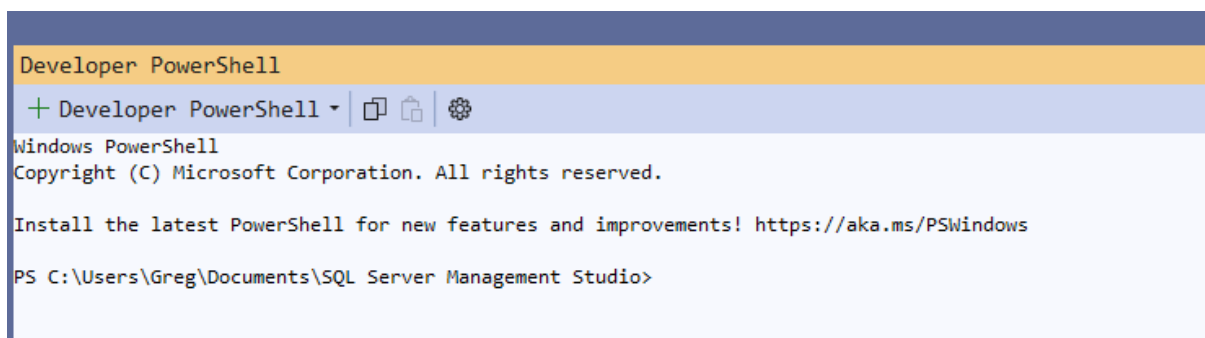
## 8.6 Using the PowerShell terminal

SSMS used to have a built-in web browser. That's now gone.

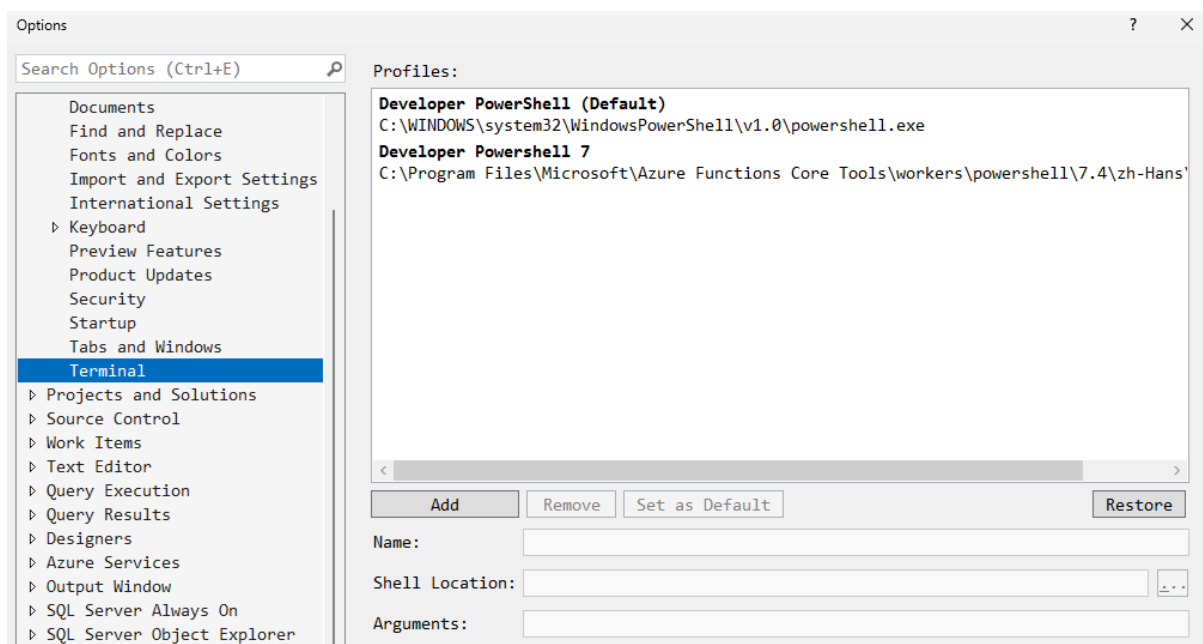
What was added, though, is a Developer PowerShell window. On the View menu, you can choose Terminal.



This will then open a Developer PowerShell window:



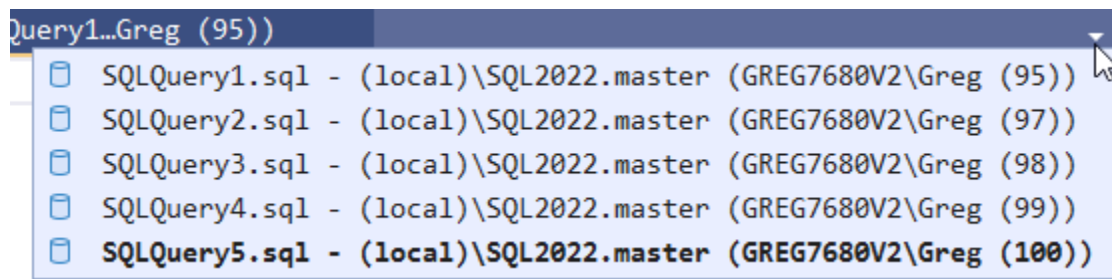
Note that you can change the version of PowerShell that's launched, from the drop-down. But the Settings option will take you to the Tools Options page where the Terminal can be configured.



That will let you set how it's configured when it's launched.

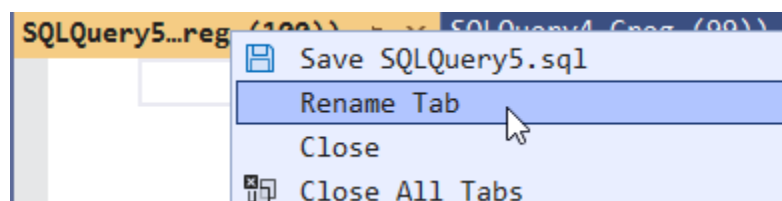
## 8.7 Renaming tabs that have not been saved

Anyone who has worked with SSMS for long, will have the experience of opening a number of query windows for short-term work, not wanting to save the scripts, and then being unable to find the one they want in the drop-down list of scripts:

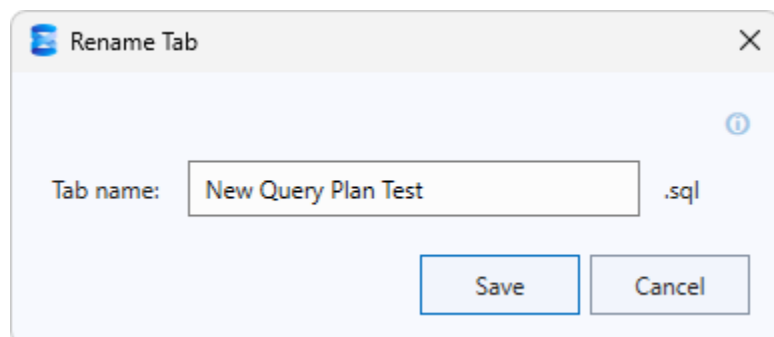


The alternative to this was to save them all somewhere so you could give them a name. Then you need to clean them all up later.

What we've wanted for a long time was the ability to name these tabs, without saving them. And that's now possible. Each tab has a rename option in its context menu:



That lets me assign a name:



And then in the drop-down, I can find the scripts that I want:

