Advanced SQL

Sample manual - first two chapters

Wise Owl Training

TABLE OF CONTENTS (1 of 5)

TABLE OF CONTENTS (2 of 5)

TABLE OF CONTENTS (3 of 5)

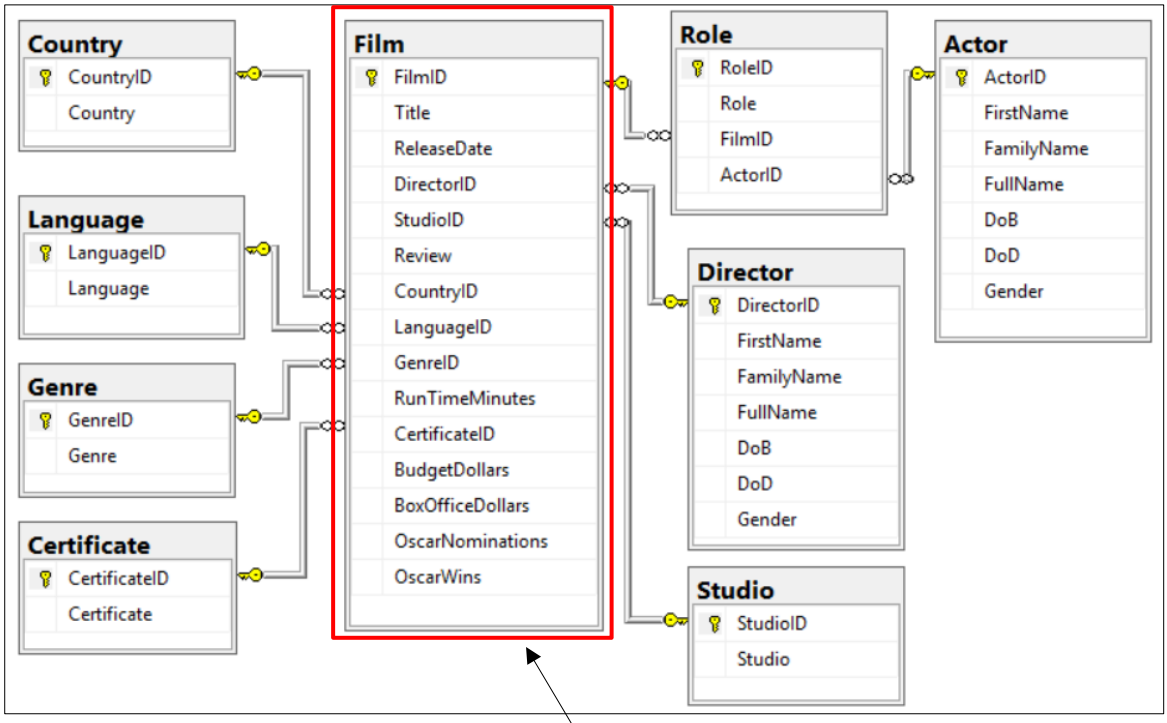
TABLE OF CONTENTS (4 of 5)

TABLE OF CONTENTS (5 of 5)

CHAPTER 1 - THE MOVIES DATABASE

1.1 Our Example Database

The database used throughout this manual contains 1,200 films, with associated details:



For each film there is an associated country, language, genre, certificate, director and studio. In addition there is a table of actors, and a **Role** table which links films and actors together (as explained below).

The **Role** table looks like this:

Each row contains the details of one actor who played a role in a film.

RoleID	Role	FilmID	ActorID
1	Ray Ferrier	33	1
2	Dr. Alan Grant	1	2
3	Dr. Ellie Sattler	1	3
4	Dr. Ian Malcolm	1	4

```
-- who played which roles
SELECT
  f.Title AS 'Film',
  a.FullName AS 'Actor',
  r.Role AS 'Role'
FROM
  Film AS f
  INNER JOIN Role AS r
    ON f.FilmID = r.FilmID
  INNER JOIN Actor AS a ON
    r.ActorID = a.ActorID
ORDER BY
  r.RoleID
```

You could run a query like this to list out all of the roles in the database, with who played this part and in which film:

Film	Actor	Role
War of the Worlds	Tom Cruise	Ray Ferrier
Jurassic Park	Sam Neill	Dr. Alan Grant
Jurassic Park	Laura Dern	Dr. Ellie Sattler
Jurassic Park	Jeff Goldblum	Dr. Ian Malcolm

CHAPTER 2 - STORED PROCEDURES

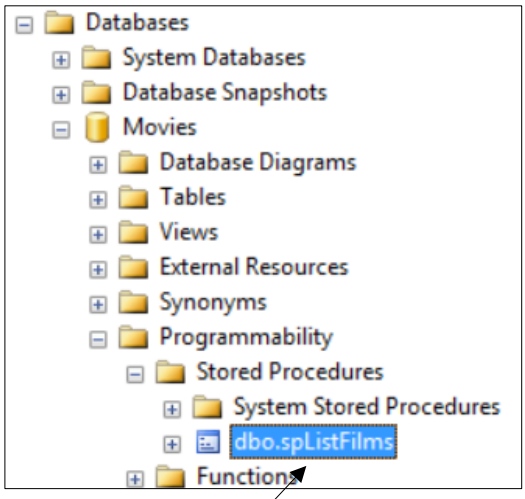
2.1 Overview

What is a Stored Procedure?

A stored procedure is a set of SQL instructions (often just a single **SELECT** statement) which is saved within your database:

```
CREATE PROC spListFilms
AS

-- list out all of the films
SELECT
  f.Title
  , f.OscarWins AS Oscars
  , f.RunTimeMinutes AS RunTime
FROM
  Film AS f
```

This is the code to create a stored procedure, here called **spListFilms**.



Here is the created procedure, in the **Programmability** section of your database.

Advantages and Disadvantages

Stored procedures have many advantages:

Advantage	Notes
Range of commands	Whereas a query can only select data, a stored procedure can also insert, update and delete rows (not to mention creating and dropping tables).
Debugging	You can step through a stored procedure line by line to see what it's doing (although this strangely isn't that useful).
Parameters	Above all, you can pass parameters to a stored procedure (although we won't do this until a later chapter). For example, you could write a procedure to list all the films made between any two given dates, winning at least N Oscars.

Against all this is one potential disadvantage: because stored procedures are so powerful, not all IT departments are that keen on giving people the authority to create and execute them!

One common misconception about stored procedures is that they run faster than simple queries. They don't, since SQL Server will create an optimised execution plan in either case.

2.2 Creating Stored Procedures

Typing in a Stored Procedure

The best way to create a procedure is to press **Ctrl** + **N** to create a query, then use this syntax:

a) Begin a stored procedure with **CREATE PROC** then give your procedure a name.

b) Although it serves no purpose, you need the keyword **AS** to separate the instruction to create a procedure from what it does.

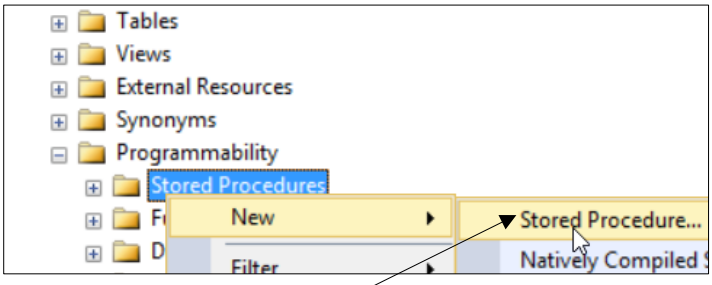
c) Finally, you need to say what your stored procedure does. This can run to hundreds of lines of code – creating tables and manipulating data – but for this chapter we'll just stick to selecting a set of rows from a table.

```
USE Movies
GO

-- this must be the first statement
-- in the batch (we needed that GO)
CREATE PROC spListFilms

-- the word AS is a necessary but
-- meaningless link word
AS

-- list out all of the films
SELECT
  f.Title
  , f.OscarWins AS Oscars
  , f.RunTimeMinutes AS RunTime
FROM
  Film AS f
```

*It's a common convention to begin procedure names with **sp**, as above. However, avoid using **sp_** as a prefix, since this is reserved for system stored procedures (and Microsoft may create one in the future which clashes with your name!).*

Creating a Stored Procedure using a Template

This is Microsoft trying to be helpful, but failing!



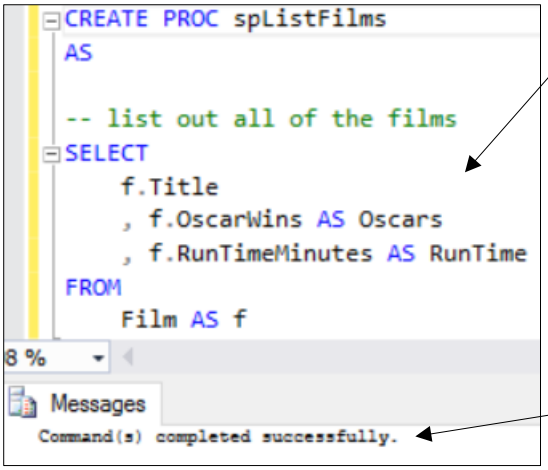
a) Expand your database to choose **Programmability → Stored Procedures → New → Stored Procedure …**

b) Edit this template. The trouble is that while you're a stored procedure newbie it's more baffling than helpful, and when you know how to create procedures it's quicker to type them in yourself!

```
-- values below.
--
-- This block of comments will not be included in
-- the definition of the procedure.
-- ============================================
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- ============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- ============================================
CREATE PROCEDURE <Procedure_Name, sysname, ProcedureN
    -- Add the parameters for the stored procedure he
    <@Param1, sysname, @p1> <Datatype_For_Param1, , i
    <@Param2, sysname, @p2> <Datatype_For_Param2, , i
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result s
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    SELECT <@Param1, sysname, @p1>, <@Param2, sysname
END
```

Executing the Query to Create your Stored Procedure

Once you've typed in SQL to create a stored procedure, it's time to run this:

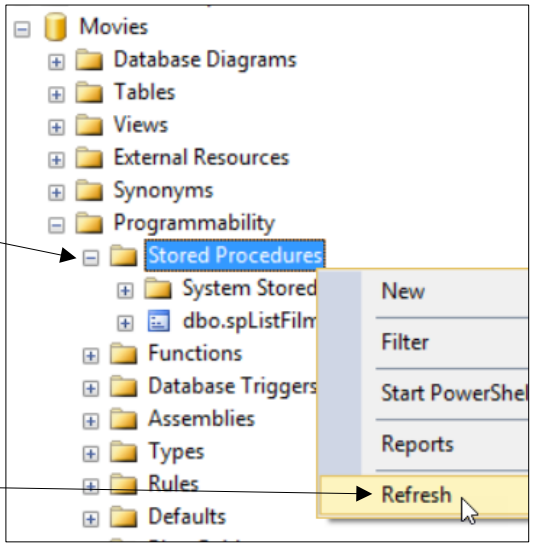


a) As for a query, click anywhere in the stored procedure and press **F5** to execute it.

b) If all goes well (and you haven't made any mistakes) you'll see this message, to show you that the stored procedure has been successfully created.

Viewing your Stored Procedure

To check SSMS has created your stored procedure, expand your database as shown here:



a) In the **Programmability** section, you should be able to expand **Stored Procedures** to see the one you've created.

b) If you can't see the procedure you've just created, right-click on **Stored Procedures** and choose **Refresh** as shown here to bring the list up to date.

*If you still can't see your stored procedure, by far the most likely reason is that you've created it in one database (probably the **master** one), but are looking in another!*

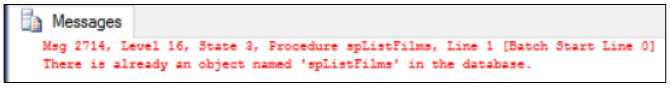
2.3 Altering a Stored Procedure

If you want to change what a stored procedure does, in the strange world of Management Studio you need to write *script* to alter it.

Altering an Open Stored Procedure

If you've just been working with a stored procedure, it's easy to change it:

a) Change the word **CREATE** to **ALTER**. If you don't do this, you'll see this message when you run your script:



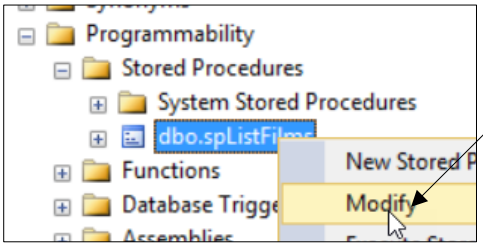
b) Make any other changes to your procedure (here we've tacked on an **ORDER BY** clause to sort the films by title).

```
ALTER PROC spListFilms
AS

-- list out all of the films
SELECT
  f.Title
  , f.OscarWins AS Oscars
  , f.RunTimeMinutes AS RunTime
FROM
  Film AS f
ORDER BY
  f.Title
```

When you run the script you'll again see the message **Command(s) completed successfully**. This means SSMS has deleted the old version of your procedure and replaced it with your new one.

Altering a Procedure in a Database

If your procedure isn't open, follow these steps to make changes to it (you can then execute the script to change what the procedure does, as shown above):



a) Right-click on the procedure that you want to change, and choose to modify it. SSMS will generate a new query containing the script shown below.

b) Although you don't have to, it's a good idea to delete these added lines of SQL to remove clutter. Here's what they do, and why you won't miss them:

Line	Notes
`USE [Movies]`	You're already using this database!
`SET ANSI_NULLS ON`	Obscure changes to the way nulls and quotation marks are treated, which are of no consequence or relevance.
`SET QUOTED_IDENTIFIER ON`	

If you find this explanation a bit lacking, go to http://bit.ly/2kQ1mfx for more details (but you're not missing anything, honest!).

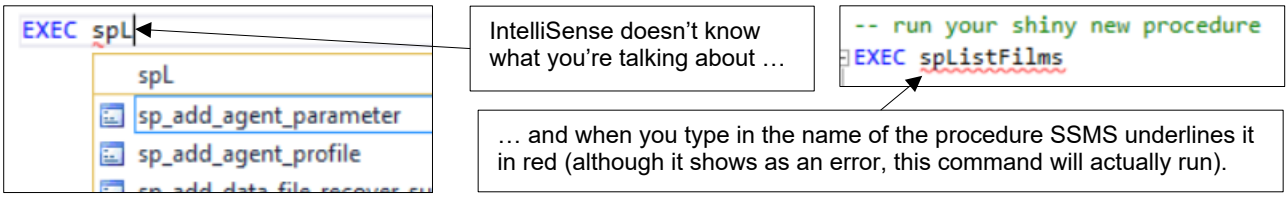
```
USE [Movies]
GO
/****** Object:  StoredProcedure [
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROC [dbo].[spListFilms]
AS

-- list out all of the films
SELECT
    f.Title
    , f.OscarWins AS Oscars
    , f.RunTimeMinutes AS RunTime
FROM
    Film AS f
```

2.4 Executing Stored Procedures

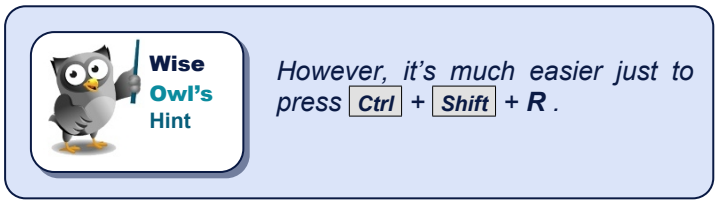
Before running a procedure, it's first a good idea to persuade SSMS your procedure exists!

Refreshing your Local Cache

You can (as we'll see in a moment) run a stored procedure using the **EXEC** command, but you have to persuade Management Studio that your stored procedure actually exists:

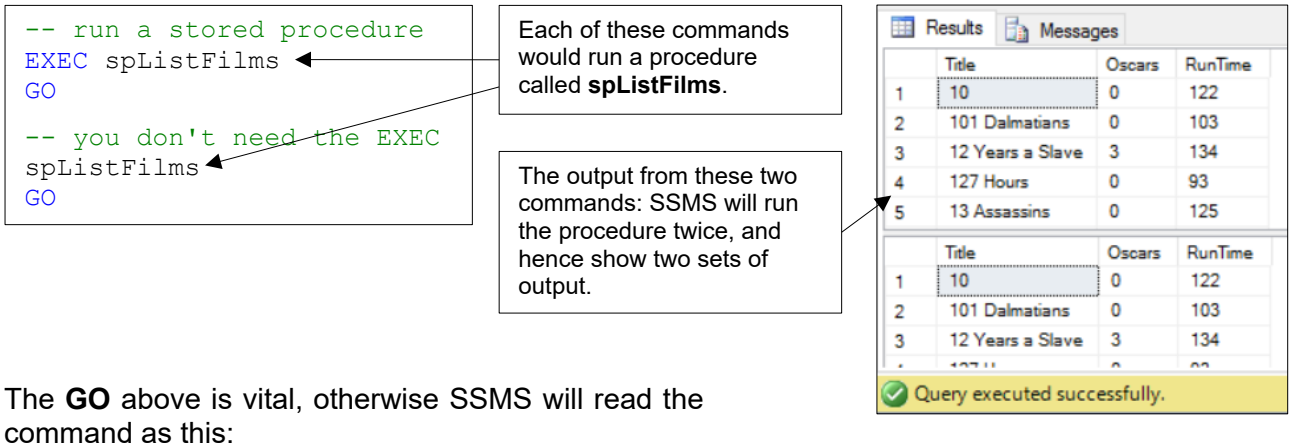


The easy way to get SSMS to acknowledge your new procedure exists is to update its memory of what's in your database. To do this select: **Edit ➔ IntelliSense ➔ Refresh Local Cache** .

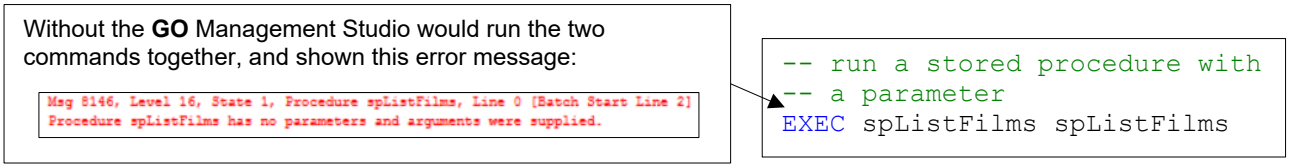


Executing a Procedure

The commands shown here would run your procedure:



The **GO** above is vital, otherwise SSMS will read the command as this:



Altering and Executing a Stored Procedure Together

A common way to run a procedure is immediately after creating or changing it:

```
ALTER PROC spListFilms
AS

-- list out all of the films
SELECT
  f.Title
  , f.OscarWins AS Oscars
  , f.RunTimeMinutes AS RunTime
FROM
  Film AS f
ORDER BY
  f.Title

-- finish creating or altering
-- the procedure!
GO

-- NOW we can run it
spListFilms
```

This part of the script alters the existing procedure, replacing whatever it used to do with new code. Don't worry if the old code and the new code are actually exactly the same!

After finishing the previous batch of statements, to modify what the stored procedure does, we now execute it.

*You need the **GO** above because otherwise you would create a script which tried to run itself, which SSMS wouldn't be happy with!*

Selecting a Stored Procedure Name to Run It

For a simple stored procedure (one which you can run without specifying any parameters), the easiest way to run it is often just to select it and press **F5** .

```
ALTER PROC spListFilms
AS

-- list out all of the
films
SELECT
  f.Title
```

a) Double-click on the name of the procedure to select it, then press **F5** .

b) SSMS will run your procedure and show its output.

Results | Messages

	Title	Oscars	RunTime
1	10	0	122
2	101 Dalmatians	0	103
3	12 Years a Slave	3	134
4	127 Hours	0	93
5	13 Assassins	0	125

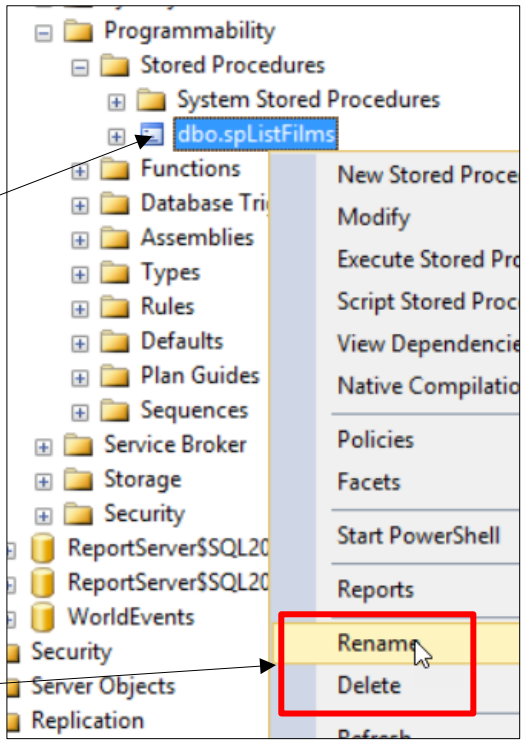
2.5 Renaming and Deleting Stored Procedures

Renaming/Deleting a Procedure with the Menu

To change the name of a stored procedure, or delete it, right-click on it:

a) Find the procedure that you want to rename or delete, and right-click on it.

b) Choose one of these options to either change its name or delete it.



Deleting a Procedure in Script

To delete a procedure, you *drop* it:

```
-- delete a procedure
DROP PROC spListFilms
```

Run this command to permanently delete the stored procedure called **spListFilms**.

Renaming a Procedure in Script

To change the name of a procedure in script, create a new version with the new name and then delete the old one:

a) Create script to modify the procedure as shown in the previous pages.

```
ALTER PROC spListFilms
AS

-- list out all films
SELECT
  f.Title
```

b) Change **ALTER** to **CREATE**, type in a new name for the procedure then execute this.

```
CREATE PROC spNewName
AS

-- list out all films
SELECT
  f.Title
```

c) Change the command to drop the original procedure,

```
DROP PROC
  spListFilms
```

2.6 System Stored Procedures

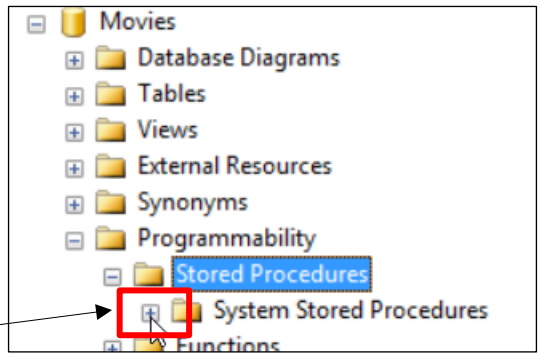
Listing System Stored Procedures

SQL Server comes with many built-in system stored procedures (1,390 in the version being used to write this courseware). Here are two ways to show these:

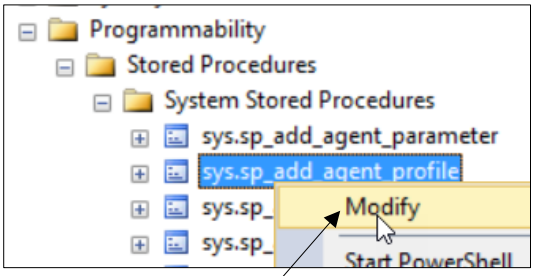
```
-- list system stored procedures
SELECT o.Name AS 'Procedure'
FROM sys.system_objects as o
WHERE o.type = 'P'
ORDER BY o.Name
```

Either run this script to show all the system objects which are procedures …

… or click on the ⊞ symbol to list them all.



You can then choose to look at any of them, although you may regret it …



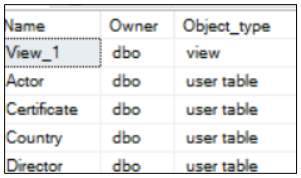
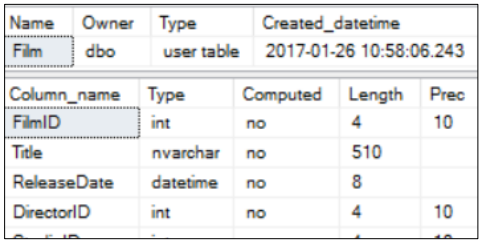
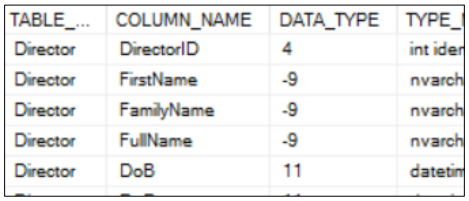
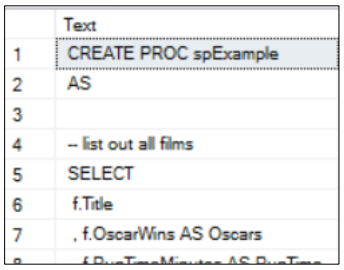
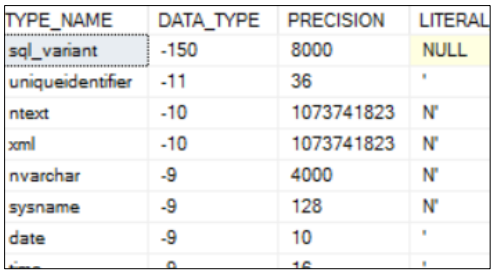
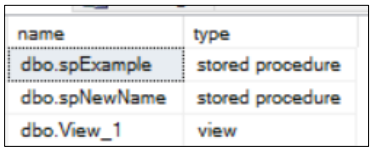
You can right-click on any of the system stored procedures to change them …

```
/*
** The system profile of the same type of agen
** the parameters in this new user profile.
*/
ALTER procedure [sys].[sp_add_agent_profile] (
    @profile_id             int = NULL OUTPUT,
    @profile_name           sysname,
    @agent_type             int,            --
```

… but the contents won't be easy to read (Wise Owl have absolutely no idea what this procedure does, for example!).

Useful System Stored Procedures

Here are some stored procedures which you might like to try:

Procedure	What it does	Example results
sp_help	Lists out all of the tables, views, etc in your database (you can also press **Alt** + **F1** to do this).	
sp_help 'Table'	Lists out all the details of (and columns in) any specified table (eg **sp_help 'Film'**)	
sp_columns 'Table'	Another way to list all the columns included in a particular table (eg **sp_columns 'Director'**).	
sp_helptext 'Procedure'	Returns the lines in a stored procedure, view or function as a table (what you do with this is not obvious!).	
sp_datatype_info	Shows information on the data types in SQL, to jog your memory.	
sp_depends	Shows where a particular table is used in your database (for example, **sp_depends 'Film'**) or which tables and columns a procedure references (eg **sp_depends 'spExample'**).	

You can see more examples of the above at this blog:

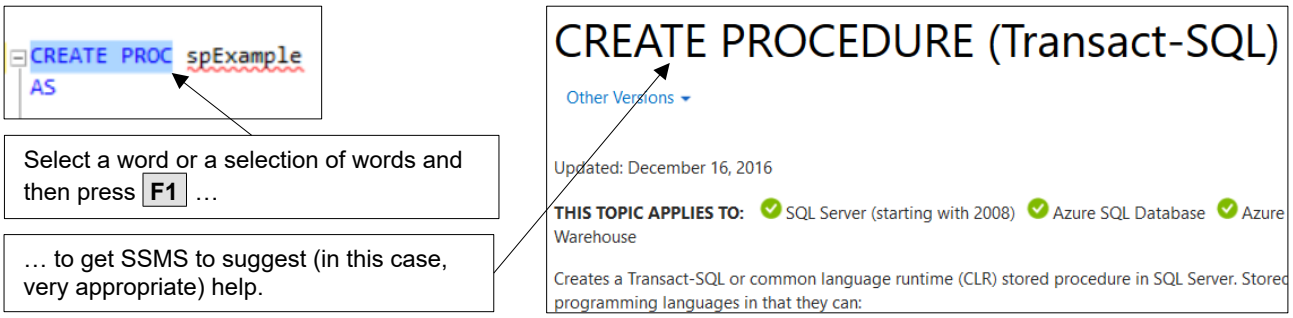
http://www.wiseowl.co.uk/blog/s2522/system_stored_procedures.htm

2.7 Getting Help on SQL

Although every programmer will have their own way to get help, here are couple of general tips.

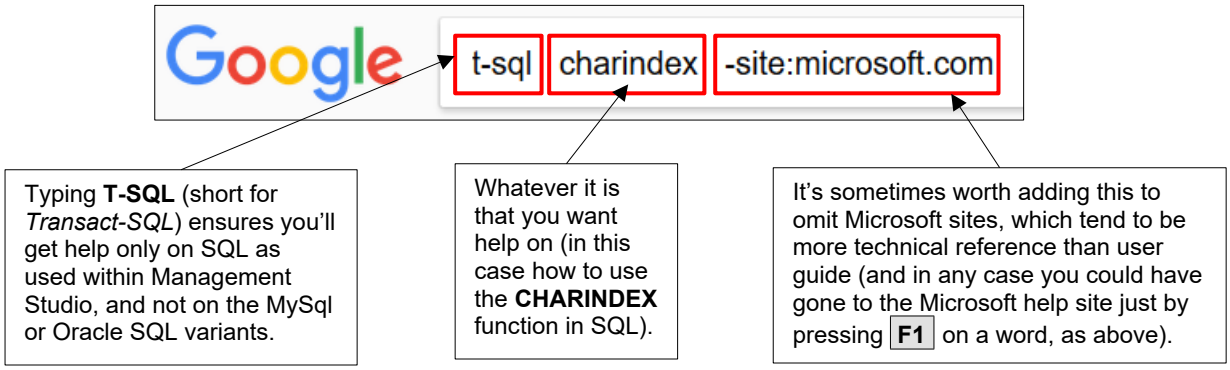
Context-Sensitive Help

You can press **F1** on any keyword (or collection of keywords) to show help in your web browser:



Tips on Googling

If you're reading this, you probably don't need much help on using search engines. Here's our advice for how to get help on any SQL topic:




Wise Owl
Training


Wise Owl
Training


Wise Owl
Training


Wise Owl
Training


Wise Owl
Training


Wise Owl
Training


Wise Owl
Training


Wise Owl
Training


Wise Owl
Training


Wise Owl
Training

What we do!

		Basic training	Advanced training	Systems / consultancy
Office	Microsoft Excel			
	VBA macros			
	Office Scripts			
	Microsoft Access			
Power BI, etc	Power BI and DAX			
	Power Apps			
	Power Automate (both)			
SQL Server	SQL			
	Reporting Services			
	Report Builder			
	Integration Services			
	Analysis Services			
Coding	Visual C#			
	VB programming			
	MySQL			
	Python			


Wise Owl
Training