**Another way to Identify Database Snapshots**

Posted on [February 21, 2012](http://www.maxtblog.com/2012/02/tsql-powershell-another-way-to-identify-database-snapshots/) by [admin](http://www.maxtblog.com/author/admin/)

FROM: <http://www.maxtblog.com/2012/02/tsql-powershell-another-way-to-identify-database-snapshots/>

Just a quick blog on spotting your Database Snaphots. I just couldn’t believe that I’ve been missing creating SQL database snapshots but sometimes having so much work make you blind.  I’ve been using a lot Hyper-V Snapshot features and recently (Thanks to [Chad Miller](http://sev17.com/)) I got the chance to create and test a few.

So, first we need to create a new db snapshot of my AdventureWorks database using T-SQL Script:

|  |  |
| --- | --- |
| 1  2  3  4  5 | CREATE DATABASE AdventureWorks\_dbSnapShot\_0001 ON  ( NAME = AdventureWorks\_Data, FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL11.MSQLDENALICTP3\MSSQL\DATA\AdventureWorks\_Data\_0001.ss' ) AS SNAPSHOT OF AdventureWorks;  GO |

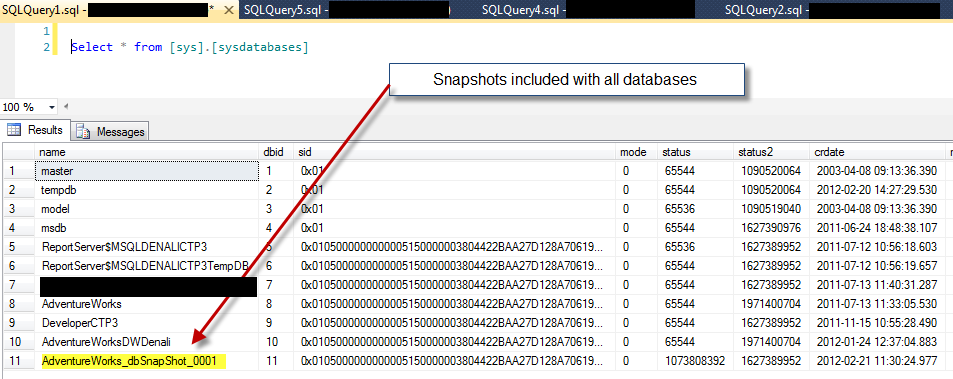
Now, I go to SSMS and verify my new database snapshot exist by going into the ‘Object Explorer’ and looking under ‘Database Shashots’ folder.

Using T-SQL

If I use the following T-SQL command to list all my databases:

[?](http://www.maxtblog.com/2012/02/tsql-powershell-another-way-to-identify-database-snapshots/)

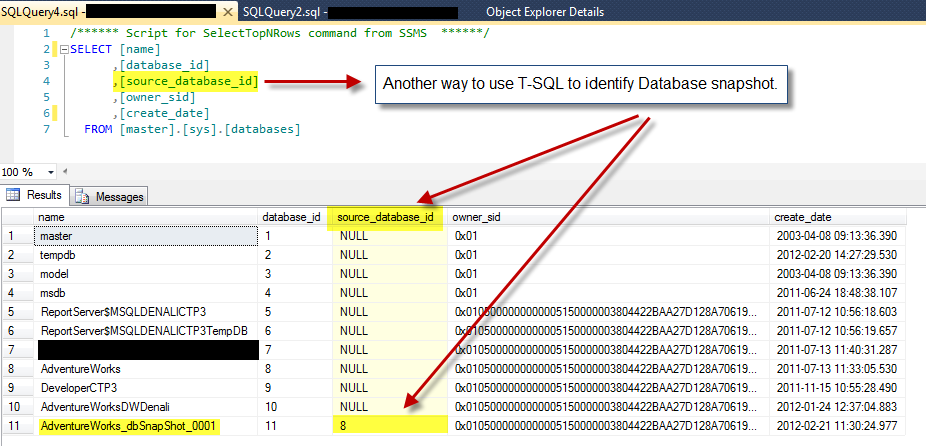
|  |  |
| --- | --- |
| 1 | Select \* from [sys].[sysdatabases] |

[](http://www.maxtblog.com/wp-content/uploads/2012/02/fix_ListwSysdatabases.png)

I will get all of them listed including the snapshots. So, here’s another way to use T-SQL to identify all database snapshots. For that, I’m going to do a select and use from the Master db the view named “[sys].[databases]”:

[?](http://www.maxtblog.com/2012/02/tsql-powershell-another-way-to-identify-database-snapshots/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | SELECT [name]        ,[database\_id]        ,[source\_database\_id]        ,[owner\_sid]        ,[create\_date]    FROM [master].[sys].[databases] |

[](http://www.maxtblog.com/wp-content/uploads/2012/02/fix_TSQL_IdentifyDBSnapshot.png)

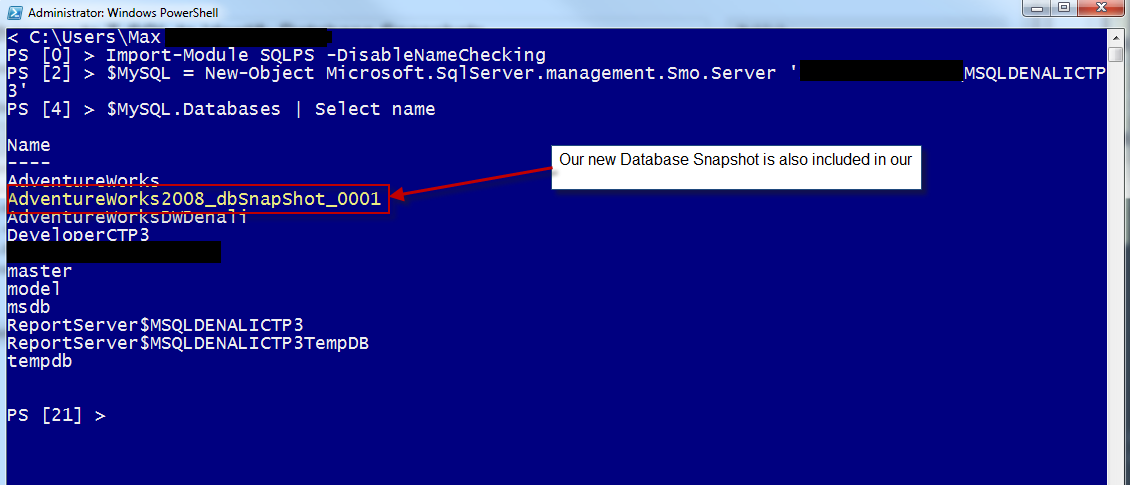
In this result, pay attention to the “[source\_database\_id]” column. Notice that this column is mostly “null” except when if it has a value. This value will match the “[Database\_ID]” column. So, you can use this to identify database snapshots.

Using PowerShell

Now, let’s take a look on how to use PowerShell to identify database snapshots using SMO. In the following code I’m listing all databases but snapshots are also included.

[?](http://www.maxtblog.com/2012/02/tsql-powershell-another-way-to-identify-database-snapshots/)

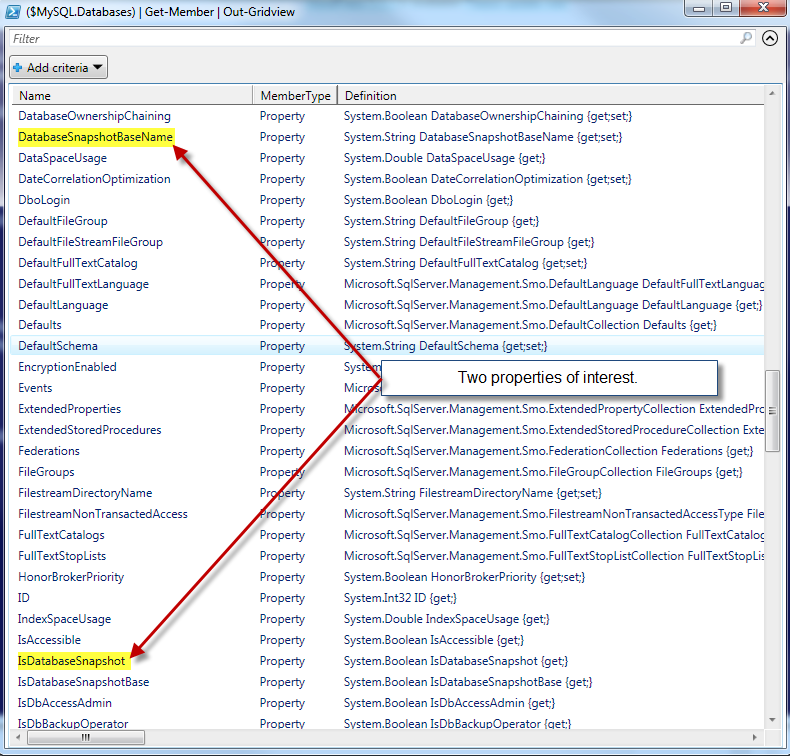
|  |  |
| --- | --- |
| 1  2  3 | Import-Module SQLPS -DisableNameChecking $MySQL = New-Object Microsoft.SqlServer.management.Smo.Server 'YourServername\InstanceName' $MySQL.Databases | Select name |

[](http://www.maxtblog.com/wp-content/uploads/2012/02/fIX_PS_DBSnapshots_listed1.png)

So, in order to identify the database snapshot I need to look deeper into our .NET database object. Using the “**Get-Member**” command I can see what’s object are available.

[?](http://www.maxtblog.com/2012/02/tsql-powershell-another-way-to-identify-database-snapshots/)

|  |  |
| --- | --- |
| 1 | ($MySQL.Databases) | Get-Member | Out-Gridview |

[](http://www.maxtblog.com/wp-content/uploads/2012/02/PS_GetMember.png)

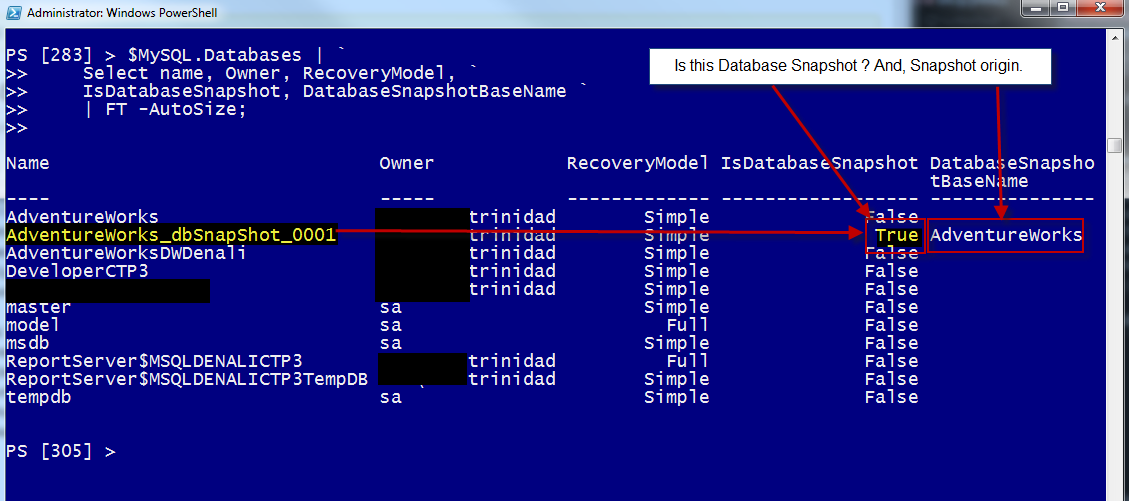
Using the “**Out-Gridview**” command to view my results a separate popup window, I found two properties of interest:

1. ‘*IsDatabaseSnapshot*‘ – which show “*true*” us if is a snapshot.
2. ‘*DatabaseSnapshotBaseName*‘ – which give us the origne of database name of the snapshot.

So, now I can use the following PowerShell commands to show all my databases and identify the snapshots:

[?](http://www.maxtblog.com/2012/02/tsql-powershell-another-way-to-identify-database-snapshots/)

|  |  |
| --- | --- |
| 1  2  3  4 | $MySQL.Databases | `     Select name, Owner, RecoveryModel, `     IsDatabaseSnapshot, DatabaseSnapshotBaseName `     FT -AutoSize; |

[](http://www.maxtblog.com/wp-content/uploads/2012/02/PS_IdentifyDBSnapshot.png)

Conclusion

Using both T-SQL and PowerShell examples, now you have a startup point to spot and take some control over your database snapshots.

Happy PowerShelling!