### Introduction

As you know that SQL Server 2008 supports Windows PowerShell and is a powerful scripting shell that helps administrators to automate and built robust server administration scripts. In my previous articles I’ve written about Using SQL Server 2008 PBM, [how to monitor Free Space of data file(s)](http://www.sqlservercentral.com/articles/PBM/68602/) and How to Evaluate Policies on Multiple Instances using PowerShell scripts(Note for Editor. Marked words need to be made as hyperlink with my 29th Mar publishing article). In this article I’m going to walk you through how to create policies using PowerShell script.

In SQL 2008, policies can be created in two ways. One is using SSMS (SQL Server Management Studio) wizard and second is using PowerShell script. In my scenario I’ve been requested to create this script to deploy/create policies as a POST installation step. Suppose if you need to use this script for unattended SQL installations then make sure that this script called only after a complete SQL installation done on a server, because you need a SQL policy store to create your policies.

To simplify the code for readers I’ve divided the script in to five parts and described them as below

1. Load required assemblies if you need to run script from windows command line.

When you run/call the PowerShell script from Windows command line prompt and if the script using any assemblies that need to be loaded before it used. But this step is not required if you run/call in the PowerShell command window.

1. Policy & Condition declaration

In the script the Policy and Condition name definitions are passed through parameters that would be lot easier to update them later if needed.

1. Create Policy Store Connection

Before you create a policy or condition on SQL Server of course you need to connect to the policy store, in this part of code connecting the SQLDBOps instance to create your policy.

1. Create Conditions

Through a policy if you want to do an audit/validate a target in SQL server you need to specify the condition which achieves your goal.

1. Create Objectset

Objectset is a condition specifies a set of objects to which the policy should be applied. Which means in a SQL server there are numerous objects but through the condition you have to narrow down to which object needs to be audited/evaluated. Here you are evaluating data files and a SQL server may have many databases and each database may have several files and file groups. Here I’m using an object set to evaluate this policy only on user databases.

1. Create Policy

This script will create the policy "DataFile\_FreeSpace\_Check" on SQLDBOps server.

Here is the script.

#0. Load required assemblies if you need to run script from windows command line.

[System.Reflection.Assembly]::LoadWithPartialName('Microsoft.SQLServer.Management.Sdk.Sfc')

[System.Reflection.Assembly]::LoadWithPartialName('Microsoft.SqlServer.Dmf')

# 1.Policy & Condition declaration

$policyName = "DataFile\_FreeSpace\_Check"

$conditionName = "DataFile\_Free\_Space"

$conditionName2 = "UserDatabasesOnly"

$ObjectSetName = "UserDatabasesOnly\_ObjSet"

$ManagementServer="SQLDBOps"

#2. Create Policy Store Connection

$conn = New-Object Microsoft.SQlServer.Management.Sdk.Sfc.SqlStoreConnection("server=$ManagementServer;Trusted\_Connection=true");

$PolicyStore = New-Object Microsoft.SqlServer.Management.DMF.PolicyStore($conn);

#3. Create Conditions

 $condition = New-Object ('Microsoft.SqlServer.Management.Dmf.Condition')($conn, $conditionName)

 #Make sure there is space between words, otherwise you have to break your head (Error 1)

 $condition.Facet = 'DataFile'

 $condition.ExpressionNode = '@AvailableSpace >= Multiply(@Size, 0.1)'

 $Condition.Create()

 $condition = New-Object ('Microsoft.SqlServer.Management.Dmf.Condition')($conn, $conditionName2)

 $condition.Facet = 'Database'

 $condition.ExpressionNode = '@ID >= 4'

 $Condition.Create()

#4. Create Objectset

 $ObjectSet = New-Object ('Microsoft.SQLServer.Management.Dmf.ObjectSet')($Conn, $ObjectSetName)

 $ObjectSet.Facet = 'DataFile'

 $targetSet = $ObjectSet.TargetSets["Server/Database/FileGroup/File"]

 $targetSet.Enabled = 1

 $targetSet.SetLevelCondition($targetSet.GetLevel("Server/Database"),$conditionName2)

 $objectSet.Create()

#5. Create Policy

 $Policy = New-Object Microsoft.SQLServer.Management.Dmf.Policy ($Conn, $policyName)

 $Policy.Condition=$conditionName

 $Policy.AutomatedPolicyEvaluationMode="None"

 $Policy.Objectset = $ObjectSetName

 $Policy.Create()

Save the above script in a .ps1 (PowerShell) file and run it on PowerShell window as below example.

.\CreatePolicy.ps1

If you want to call the script on windows command line then call the script file as below example.

PowerShell –File “C:\Foldername\CreatePolicy.ps1”

Also make sure that load assembly script exists in your script file otherwise you will get the below errors.

New-Object : Cannot find type [Microsoft.SQlServer.Management.Sdk.Sfc.SqlStoreConnection]: make sure the assembly containing this type is loaded.

New-Object : Cannot find type [Microsoft.SqlServer.Management.DMF.PolicyStore]: make sure the assembly containing this type is loaded.

 [Special thanks to Sethu.Srinivasan@microsoft.com to figure out the above issue]

**Conclusion**

I hope this script will help DBA’s to create policies from PowerShell prompt / windows command line. Based on the above method you can create other polices with conditions for your environment.

**References**

<http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.management.dmf.aspx>

MAK’s [Microsoft SQL Server 2008 Administration with Windows PowerShell](http://books.internet.com/books/0470477288)

Sethu’s <http://blogs.msdn.com/sethus/archive/2008/06/16/sql-2008-powershell-script-for-creating-a-policy-and-saving-to-file.aspx>

BOB’s <http://www.sqlskills.com/BLOGS/BOBB/category/Policy-Based-Management.aspx#p3>