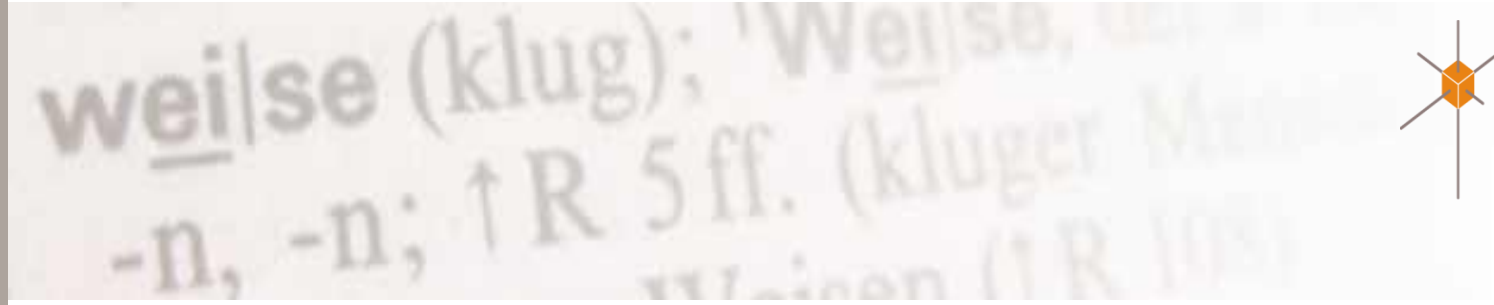


Netz-Weise
Lernen von den Besten.

SQL-Server mit Powershell verwalten

Holger Voges

CCA,MCSE, MCDBA, MCT, MCITP DB Administrator / DB Developer, MCTIP Enterprise Administrator, MCSA Windows Server 2012

Netz-Weise
Freundallee 13a
30173 Hannover
www.netz-weise.de

Cmdlets

Netz-Weise
Lernen von den Besten.

- Bilden die Befehle der Powershell
- Aufbau immer aus Verb und Substantiv
- Können erweitert werden
- Get-command zeigt u.a. installierte Commandlets an
- Get-help zeigt Hilfe zum Commandlet

Aliase

Netz-Weise
Lernen von den Besten.

- Aliase ermöglichen den Zugriff auf Commandlets über alternative Aufrufe
- NT-Shell-Kommandos sind als Aliase eingetragen
- Get-Alias zeigt verfügbare Aliase an
- Set-Alias fügt ein neues Alias ein

Provider

Netz-Weise
Lernen von den Besten.

- Powershell-Provider stellen Zugriff auf Laufwerke, Registry, Variablen… bereit
- Get-PSProvider zeigt die installierten Provider an
- Get-PSDrive zeigt die „Laufwerke“ an
- New-PSDrive legt ein neues Laufwerk an
- Set-Location (Alias: *CD*) wechselt den Provider
- Get-Item, Get-Childitem, Get-Itemproperty

Variablen

Netz-Weise
Lernen von den Besten.

- PSH-Variablen basierten auf .NET-Datentypen
- PSH erkennt den Variablentyp automatisch
- Variablen beginnen mit einem $
- Zuweisung erfolgt mit einem =
  [Datentyp]$Variable legt den Datentyp fest

```
$Var = ‚Dies ist ein Text‘
[Int16]$Zahl
```

Powershell und Objekte

Netz-Weise
Lernen von den Besten.

- PSH arbeitet mit Objekten
- Es kann auf alle Objekteigenschaften und Methoden zugegriffen werden
- Variablen speichern Objekte

```
$Dir = get-childitem c:\Windows\
get-properties –InputObject $Dir
```

Powershell Pipeline

Netz-Weise
Lernen von den Besten.

- Befehle können mit dem "|" Objekte sequentiell bearbeiten
- ForEach-object macht Befehle, die Pipes nicht unterstützen, Pipe-fähig
- Auf Einzelne Objekt wird mit $_ verwiesen

```
get-process -name sidebar | format-list -prop *
```

```
Get-Process | where-object {$_.VirtualMemorySize -gt
1000000} | Sort-Object -Property VirtualMemorySize |
format-table ProcessName,VirtualMemorySize,"CPU(s)"
```

Funktionen

Netz-Weise
Lernen von den Besten.

- Fassen Scriptblöcke zu einem Befehl zusammen
- Können Parametrisiert werden
- Provider Function: zeigt def. Funktionen an

```
Set-location Function:
Get-childitem
```

Funktionen definieren

Netz-Weise
Lernen von den Besten.

```
Function Subtrahiere($Wert1=10, $Wert2=20)
{
  $Wert1 - $Wert2
}
```

```
Function Subtrahiere($Wert1=$(Throw „Wert1 wurde nicht
angegeben!“, $Wert2=20)
{
  $Wert1 - $Wert2
}
```

Powershell und .NET

Netz-Weise
Lernen von den Besten.

- Viele Standard-Assembly stehen in Powershell zur Verfügung
- Nachladen einer .net_Assembly: [System.Reflection.Assembly]::LoadWithPartialName("System.Windows.Forms")
- Direkter Zugriff auf Methoden: [Klasse]::Methode()
- Anlegen eines Objekts mit new-object

Beispiele

Netz-Weise
Lernen von den Besten.

```
$web = New-Object net.webclient
$Seite =
$web.DownloadString(„http://blogs.technet.com/chitpro-de“)
```

```
[System.IO.Path]::GetTempFileName()
```

Powershell-Scripte

Netz-Weise
Lernen von den Besten.

- Funktionieren ähnlich wie Funktionen
- Parameter werden im Kopf mit param() angegeben
- Scriptvariablen gelten nur lokal
- Beim Start von PSH können Profilscripte angegeben werden

Remoting

Netz-Weise
Lernen von den Besten.

- Benötigt WinRM-Dienst
- Commandlets mit Computername-Parameter
- Interaktive Sitzung mit Enter-Pssession
- Invoke-command –computername –scriptblock
- Persistent Connections ermöglichen, mehr als eine Kommando auszuführen
- Rechte werden per wsman-Provider gesetzt

Beispiele Remoting

Netz-Weise
Lernen von den Besten.

```
Enable-PSRemoting –force

$s = new-pssession –computername Server01, Server02
Invoke-command –session –scriptblock {$p = get-process}

invoke-command -computername S1, S2
  -filepath C:\Test\Sample.ps1
```

Background-Jobs

Netz-Weise
Lernen von den Besten.

- Laufen im Hintergrund ab – die Powershell kann weitere Befehle annehmen
- Get-Job ruft Informationen über den laufenden Job auf
- Start-Job, Get-Job, Receive-Job, wait-job

```
$job = start-job –scriptblock {get-process}
$job = Get-Job -id 1
Receive-Job –job $job
```

Module

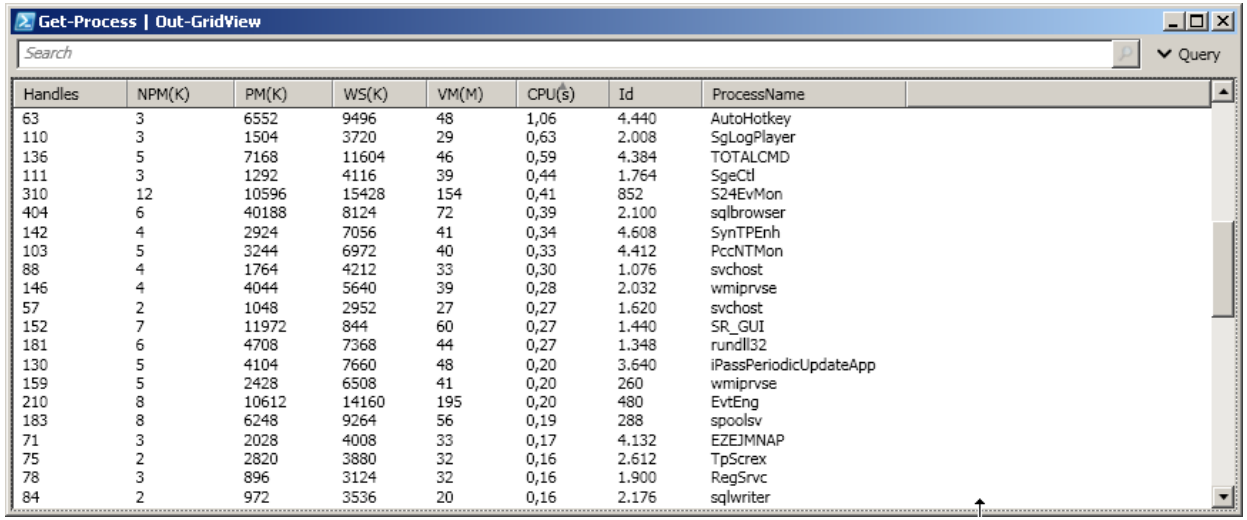
Netz-Weise
Lernen von den Besten.

- Laufen in eigener Laufzeitumgebung
- Modul-lokale Variablen sind private und behalten Ihren Zustand über mehrere Aufrufe
- Script-Module, Binär-Module (dll), Manifest-Module, Dynamische Module
- user\documents\windowspowershell\modules\mymodule (=$PSModulePath)
- windows\system32\windowspowershell\V1.0\modules\mymodule (=$PSModulePath)

Out-Gridview

Netz-Weise
Lernen von den Besten.

- Gibt die Ausgabe in Tabellenform in eigenem Fenster aus
- Ausgabe ist sortier- und durchsuchbar



Handles	NPM(K)	PM(K)	WS(K)	VM(M)	CPU(s)	Id	ProcessName
63	3	6552	9496	48	1,06	4.440	AutoHotkey
110	3	1504	3720	29	0,63	2.008	SgLogPlayer
136	5	7168	11604	46	0,59	4.384	TOTALCMD
111	3	1292	4116	39	0,44	1.764	SgeCtl
310	12	10596	15428	154	0,41	852	S24EvMon
404	6	40188	8124	72	0,39	2.100	sqlbrowser
142	4	2924	7056	41	0,34	4.608	SynTPEnh
103	5	3244	6972	40	0,33	4.412	PccNTMon
88	4	1764	4212	33	0,30	1.076	svchost
146	4	4044	5640	39	0,28	2.032	wmiprvse
57	2	1048	2952	27	0,27	1.620	svchost
152	7	11972	844	60	0,27	1.440	SR_GUI
181	6	4708	7368	44	0,27	1.348	rundll32
130	5	4104	7660	48	0,20	3.640	iPassPeriodicUpdateApp
159	5	2428	6508	41	0,20	260	wmiprvse
210	8	10612	14160	195	0,20	480	EvtEng
183	8	6248	9264	56	0,19	288	spoolsv
71	3	2028	4008	33	0,17	4.132	EZEJMNAP
75	2	2820	3880	32	0,16	2.612	TpScrex
78	3	896	3124	32	0,16	1.900	RegSrvc
84	2	972	3536	20	0,16	2.176	sqlwriter

Titelmasterformat durch Klicken bearbeiten

Powershell und SQL

Sql Server Managed Backup für Azure

Netz-Weise
Lernen von den Besten.

```
cd SQLSERVER:\SQL\Computer\MyInstance
$encryptionOption = New-SqlBackupEncryptionOption -
EncryptionAlgorithm Aes128 -EncryptorType
ServerCertificate -EncryptorName "MyBackupCert"
Get-SqlSmartAdmin | Set-SqlSmartAdmin –BackupEnabled
$True –BackupRetentionPeriodInDays 10 -EncryptionOption
$encryptionOption
```

Server Management Objects

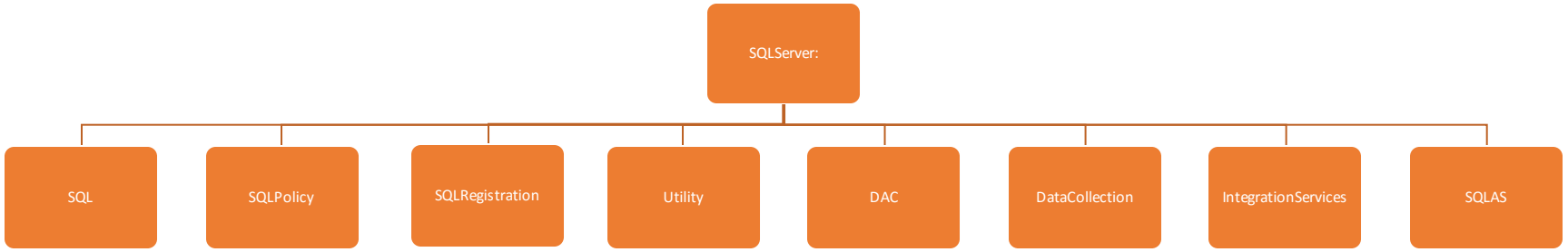
Netz-Weise
Lernen von den Besten.

- Grundlage für Powershell
- Wird von PSH automatisch geladen
- Stellt eine API zum SQL-Server dar
- Liefert jede Menge Diagnoseinformationen

```
$instanceName = „SQL2012"
$server = New-Object -TypeName
Microsoft.SqlServer.Management.Smo.Server -ArgumentList
$instanceName
```

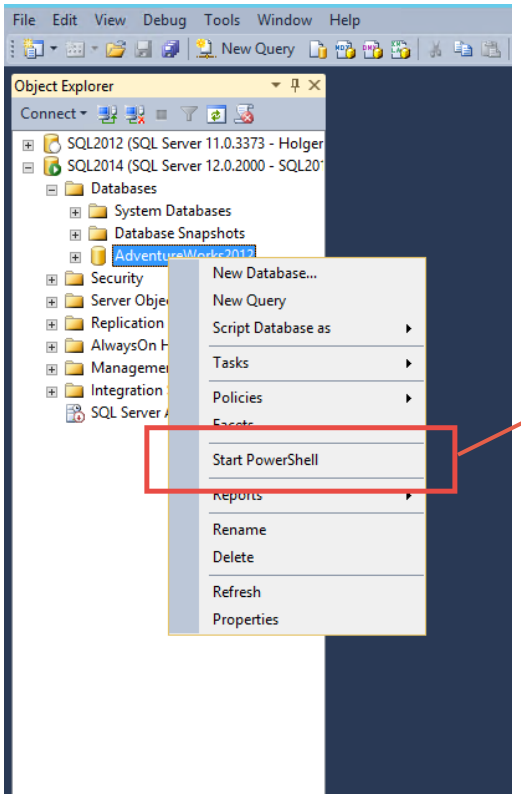
SQL-Server Powershell Provider

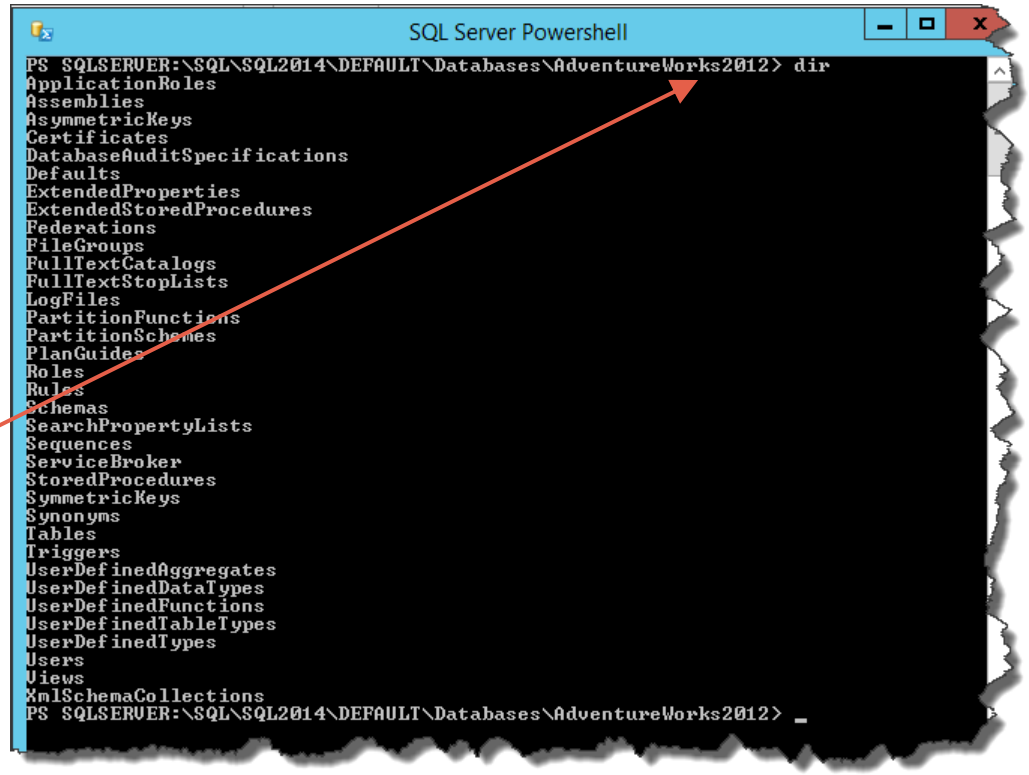
Netz-Weise
Lernen von den Besten.



SQLPS aus dem Management Studio

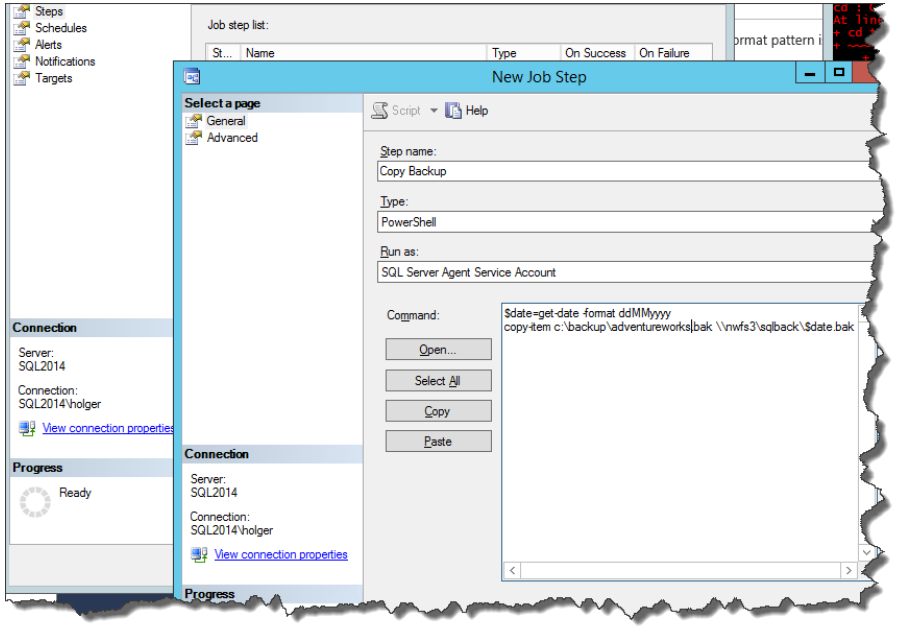
Netz-Weise
Lernen von den Besten.





SQL-Agent Jobs mit Powershell

Netz-Weise
Lernen von den Besten.




```
$RSService = "http://localhost/ReportServer/ReportService2010.asmx?wsdl"
$RSServiceProxy = New-WebServiceProxy -uri $RSService -class RS2 -Namespace ReportingWebservice -Credential Get-Credential
$RSServiceProxy.ListSubscriptions("")
$id = $RSServiceProxy.ListSubscriptions("") | where Report -eq "Sales_Order_Detail_2008R2" | select subscriptionID -ExpandProperty
$RSServiceProxy.FireEvent("TimedSubscription",$id,"")
```

Reports generieren mit dem Agen

Here-Strings verwenden

Netz-Weise
Lernen von den Besten.

- Here-Strings sind ideal um längere SQL-Scripte in Powershell zwischen zu speichern
- Ein Here-String beginnt mit einem @“ und enden mit einem "@
- Alles, was zwischen den @-Zeichen steht, wird als ein String behandelt

```
$Query = „@
SELECT db.name, db.create_date, df.physical_name,
df.max_size FROM master.sys.databases AS db
INNER JOIN master.sys.database_files AS df
ON db.name = df.name
"@
```

SQL mit Powershell ausführen

Netz-Weise
Lernen von den Besten.

- Invoke-sql startet SQL-Befehle
- Wenn sqlps nicht zur Verfügung steht, muß .net herhalten

```
$ConnStr = "Server=SQLDB;Initial Catalog=Nwind;Integrated
Security = True"
$conn= New-Object System.Data.SQLClient.SQLConnection
$conn.ConnectionString = $ConnStr
$Conn.Open()
$command = New-Object System.Data.SqlClient.SqlCommand
$command.Connection = $conn
$statement = "INSERT INTO log (runtime,errors) values
('$($endtime)',$($errors))"
$command.CommandText = $Statement
$command.ExecuteNonQuery()
```

SMO

Netz-Weise
Lernen von den Besten.

- Server Management Objects stellen eine API zum SQL-Server zur Verfügung
- SMO ist über .NET und damit für Powershell nutzbar
- Damit SMO genutzt werden kann, benötigt man die SMO-Erweiterungen:
  - Client SDK im SQL-Server Installationspaket
  - Bei Microsoft als Bestandteil des Feature-Packs verfügbar
- SMO-Assemblies müssen in Powershell mit Add-Type oder der Methode LoadWithPartialName() geladen werden
- Da Add-Type leider einige Macken hat, wird nach wie vor oft auf das (veraltete) LoadWithPartialName() zurückgegriffen http://www.madwithpowershell.com/2013/10/add-type-vs-reflectionassembly-in.html

Demo

Zusätzliche Quellen

Netz-Weise
Lernen von den Besten.

- 10 Tips for the SQL Server Powershell Scripter
  http://blogs.technet.com/b/heyscriptingguy/archive/2013/05/06/10-tips-for-the-sql-server-powershell-scripter.aspx
- Automated Script-generation with Powershell and SMO
  https://www.simple-talk.com/sql/database-administration/automated-script-generation-with-powershell-and-smo/
- SQL Server 2012 with PowerShell V3 Cookbook
  Packt Publishing, 978-1849686464