Kopieren der QPilot SQL DBs.

Aufgabe:  
Jede Nacht um 5 Uhr sollen von diversen Servern die Datenbanken auf einen anderen kopiert werden.

Hintergrund:  
Das Printserver Team möchte im Störungsfall eines primären PrintDB Servers unabhängig von anderen Teams einen Schwenk auf einen sekundären Server machen. Dies wird durch einen anderen Prozess gesteuert.

Lösung:  
Es wurde ein Powershellskript entwickelt welches die Anforderungen umsetzt. Das Skript arbeitet derzeit noch seriell die Datenbanken ab. Die Gesamtdauer betrifft jetzt noch gerade mal rund 45Sekunden. Das Skript erledigt die folgenden Punkte vollautomatisiert.

* Backup der DB via sqlcmd über einen UNC Pfad auf den Zielserver. Die Freigabe auf dem Zielserver muss vorher eingerichtet und die entsprechenden Rechte sowohl im NTFS als auch auf Netzwerk Ebene eingerichtet sein.
* Restore auf dem Zielserver
* Setzten der DB\_owner
* Leeren der Tabellen Printjobs und Scnajobs
* Löschen der Backup Datei
* Versenden einer Mail im Fehlerfall

Skriptsteuerung:  
Das Skript selber liegt auf dem Management Server vom MS SQL.OP

Server: DE-DACSQLPWC906 (Clusterinstanz)  
Pfad: D:\SystemDBs\Skripte\Copy\_Qpilot\_DBs.ps1

Gesteuert wird das Skript über den SQL Agent in der MGT Instanz von MS SQL.OP auf dem oben benannten Server. Hier ist ein Job eingerichtet der immer um 5:00Uhr läuft und das Skript startet. Der Job heißt „QPilot\_KopierenAllerDBs“.

Jobname: QPilot\_KopierenAllerDBs  
 Uhrzeit: täglich um 5:00 Uhr

Zum Skript selbst:

Im Kopf muss nur die eine Zeile mit den Quell und Zielinformationen angepasst werden wenn neue Server dazu kommen, oder welche abgebaut werden.

Hier das komplette Skript:

#cls

# WICHTIG! Die Paarungen werden durch ";" getrennt. Die Werte innerhalb durch ein "," und alles OHNE Leerzeichen

# Im Beispiel sind dieser nur zur besseren Darstellung.

# Auf dem Zielpfad muss der Ordner Backup als Backup freigegeben werden und die SQL SA Konnten sowohl NTFS als auch im Share Lese/Schreibrechte haben.

#Beispiel Paarungen: "Quellserver , Zielserver , DB ; nächste Paarung"

# Anzupassende Parameter

$Paarung = "DE-DACPRNPWV002\DE\_QPILOT\_P\_01,DE-DACPRNPWV005\DE\_QPILOT\_P\_04,QPilot\_02;DE-DACPRNPWV003\DE\_QPILOT\_P\_02,DE-DACPRNPWV005\DE\_QPILOT\_P\_04,QPilot\_03;DE-DACPRNPWV004\DE\_QPILOT\_P\_03,DE-DACPRNPWV005\DE\_QPILOT\_P\_04,QPilot\_04;DE-DACPRNPWV006\DE\_QPILOT\_S\_01,DE-DACPRNPWV005\DE\_QPILOT\_P\_04,QPilot\_06"

#zum testen

#$Paarung = "DE-DACPRNPWV002\DE\_QPILOT\_P\_01,DE-DACPRNPWV005\DE\_QPILOT\_P\_04,QPilot\_01;DE-DACPRNPWV003\DE\_QPILOT\_P\_02,DE-DACPRNPWV005\DE\_QPILOT\_P\_04,QPilot\_03;DE-DACPRNPWV004\DE\_QPILOT\_P\_04,DE-DACPRNPWV005\DE\_QPILOT\_P\_04,QPilot\_04;DE-DACPRNPWV006\DE\_QPILOT\_S\_01,DE-DACPRNPWV005\DE\_QPILOT\_P\_04,QPilot\_06"

# Optional wenn sich die SQL Version ändert

$SQL = "12" #SQL2014

$debug = "" # 1 für debuging = Ausgabe in der Konsole

$\_from\_mail = "SQL MGT Server <mssql\_reportingservice@de.pwc.com>"

$\_to\_mail = "Print Services <print-services@de.pwc.com>"

$\_to\_mail\_cc = "SQL MGT Server <mssql\_reportingservice@de.pwc.com>"

#$\_to\_mail = "<tobias.kolberg@de.pwc.com>"

# Diese Schleife läuft seriell für jede Paarung

foreach($P in $($Paarung.Split(";"))){

#Paarungen aufplitten und Variablen füllen

$P\_Split = $P.Split(",")

$quellserver=$P\_Split[0]

$zielserver=$P\_Split[1]

$DB=$P\_Split[2]

$PfadTeil1= $($P\_Split[1]).Split("\")[0]

$PfadTeil2= $($P\_Split[1]).Split("\")[1]

$remoteBackupPfad="\\$PfadTeil1\Backup\$DB.bak"

$localBackupPfad="D:\UserDBs\MSSQL$SQL.$PfadTeil2\MSSQL\Data\Backup\$DB.bak"

$RestoreDB="D:\UserDBs\MSSQL$SQL.$PfadTeil2\MSSQL\Data\$DB.mdf"

$DBLogName="$DB"+"\_log"

$Logfile= "$DBLogName.ldf"

$RestoreLog="D:\UserLogs\MSSQL$SQL.$PfadTeil2\MSSQL\Data\$Logfile"

$BackupQuerry ="BACKUP DATABASE [$DB] TO DISK = N'$remoteBackupPfad' WITH COPY\_ONLY, NOFORMAT, NOINIT, NAME = N'$DB-copyjob', SKIP, NOREWIND, NOUNLOAD, STATS = 10"

$RestoreQuerry ="USE [master];ALTER DATABASE [$DB] SET SINGLE\_USER WITH ROLLBACK IMMEDIATE;RESTORE DATABASE [$DB] FROM DISK = N'$localBackupPfad' WITH FILE = 1, MOVE N'$DB' TO N'$RestoreDB', MOVE N'$DBLogName' TO N'$RestoreLog', NOUNLOAD, REPLACE, STATS = 5;ALTER DATABASE [$DB] SET MULTI\_USER"

$scriptPath = split-path -parent $MyInvocation.MyCommand.Definition

$scriptLogPath = "$scriptPath\Logs"

$Datum=(get-date -DisplayHint Date -Format yyyy\_MM\_dd)

$scriptLogfile = "$scriptLogPath\$Datum-Copy\_QPilot\_log.txt"

if ($debug) { Write-host " ## next Server: $quellserver ## " -ForegroundColor Yellow }

# Backup DB

if ($debug) { Write-host "Backup DB: $DB from Server: $quellserver path: $remoteBackupPfad" -BackgroundColor Cyan -ForegroundColor Black }

$BCKout = $(sqlcmd -b -S $quellserver -Q $BackupQuerry)

if ($BCKout -like "BACKUP DATABASE successfully\*")

{

if ($debug) { Write-host "Backup OK" -ForegroundColor Green }

# Restore DB

if ($debug) { Write-host "restore DB: $DB on Server: $zielserver" -BackgroundColor Cyan -ForegroundColor Black }

$RESTout = $(sqlcmd -b -S $zielserver -Q $RestoreQuerry)

if ($RESTout -like "RESTORE DATABASE successfully\*")

{

if ($debug) { Write-host "Restore OK" -ForegroundColor Green }

if ($debug) { Write-host "Set DB Owner and delete data from tables" -BackgroundColor Cyan -ForegroundColor Black }

$setOwner = "USE [$DB]; EXEC dbo.sp\_changedbowner @loginame = N'pwcsysop', @map = false"

$out = $(sqlcmd -b -S $zielserver -Q $setOwner)

$DelTables = "USE [$DB]; DELETE FROM dbo.PrintJob; DBCC CHECKIDENT ('$DB.dbo.PrintJob',RESEED, 0); DELETE FROM dbo.ScanJob; DBCC CHECKIDENT ('$DB.dbo.ScanJob',RESEED, 0)"

$out = $(sqlcmd -b -S $zielserver -Q $DelTables)

if ($debug) { Write-host "remove backup $remoteBackupPfad" -BackgroundColor Cyan -ForegroundColor Black }

Remove-Item -Path $remoteBackupPfad -Force

}

else

{

#Errorlog für Mail

$fehleraufgetreten = "1"

$zeit = $(Get-Date -DisplayHint Time -Format HH:MM:ss)

"$zeit [ERROR] Restore - Quelle: $quellserver - Ziel: $zielserver - DB: $DB" | Out-File -filepath "$scriptLogfile" -Encoding default -Append

$RESTout | Out-File -filepath $scriptLogfile -Encoding default -Append

" ####### " | Out-File -filepath $scriptLogfile -Encoding default -Append

}

if ($debug)

{

Write-host " ####### " -ForegroundColor Red

Write-host "Restore error" -Foregroun Red

$RESTout

Write-host " ####### " -ForegroundColor Red

}

}

else

{

#Errorlog für Mail

$fehleraufgetreten = "1"

$zeit = $(Get-Date -DisplayHint Time -Format HH:MM:ss)

"$zeit [ERROR] Backup - Quelle: $quellserver - Ziel: $zielserver - DB: $DB" | Out-File -filepath "$scriptLogfile" -Encoding default -Append

if ($BCKout)

{

$BCKout | Out-File -filepath $scriptLogfile -Encoding default -Append

}

else

{

"Es ist ein anderer Fehler aufgetreten. Evtl. war ein Server nicht erreichbar." | Out-File -filepath "$scriptLogfile" -Encoding default -Append

}

" ####### " | Out-File -filepath $scriptLogfile -Encoding default -Append

}

if ($debug)

{

Write-host " ####### " -ForegroundColor Red

Write-host "Backup error" -Foregroun Red

$BCKout

Write-host " ####### " -ForegroundColor Red

}

$BCKout = ""

$RESTout=""

}

# Send Mail on error

if ($fehleraufgetreten)

{

$sub = "[ERROR] QPilot Copyjob mit Fehler"

$body = "Der QPilot CopyJob hat einen Fehler festgestellt. Mehr Infos im angehängten Log File. `r `n ##### LOGFILE ############### `r $(foreach ($line in $(Get-Content $scriptLogfile)) { "$line `r" })"

Send-MailMessage -SmtpServer smtp-prod -to $\_to\_mail -from $\_from\_mail -Cc $\_to\_mail\_cc -Subject $sub -body $body -Encoding ([System.Text.Encoding]::UTF8)

Remove-Item $scriptLogfile -Force

}