Datenbanken per PowerShell sichern, Teil 1
Versionen: Access 2013, 2010, 2007, 2003 und 2002/XP

Mit der PowerShell von Windows können Sie leistungsfähige Scripts erstellen, die Routineaufgaben per Mausklick oder automatisch zeitgesteuert erledigen. Eine dieser Aufgaben ist beispielsweise die Erstellung von Sicherungskopien Ihrer Datenbanken. Wie Sie eine entsprechende Lösung umsetzen, erläutert unser zweiteiliger Tipp der Woche.

Bei den folgenden Ausführungen gehen wir davon aus, dass die PowerShell auf Ihrem System installiert, für die Ausführung von PowerShell-Scripts eingerichtet ist und Sie mit dem Script-Editor "PowerShell ISE" (ISE= Integrated Script Environment) umgehen können. Tutorials dazu und weitere Informationen zur PowerShell finden Sie bei Bedarf im Microsoft Script Center unter der folgenden Adresse:

[**http://technet.microsoft.com/de-de/scriptcenter/powershell.aspx**](http://technet.microsoft.com/de-de/scriptcenter/powershell.aspx)

Mit den Batch-Dateien (.CMD, .BAT) des Betriebssystems können Sie zwar zum Beispiel per 'copy "D:\Access-DBs\Adressen.accdb" "Q:\DB-Backups\Adressen.accdb"' eine Sicherungskopie erstellen, ab der zweiten Ausführung wird aber die jeweils zuvor erstellte Kopie überschrieben, so dass Sie nicht mehr gezielt beispielsweise auf eine Kopie vom Dienstag der letzten Woche zurückgreifen können. Dieses Problem löst ein PowerShell-Script mit seinen leistungsfähigen Möglichkeiten des Zugriffs auf Funktionen aus dem .NET Framework. Um es anzulegen, starten Sie die PowerShell ISE und kopieren die folgenden Anweisungen in den Editor:

Cls
Write-Host "Datenbanken sichern...`n`n"
foreach($file in (Get-Item D:\Access-DBs\\*.\*db))
{
  Write-Host "Erstelle Backup von $($file.FullName)...";
  $FileNameOnly = [System.IO.Path]::
  GetFileNameWithoutExtension($file);
  $ExtOnly = [System.IO.Path]::GetExtension($file);
  $DestFileName = $FileNameOnly +
                  " - " +
                  [DateTime]::
                  Now.ToString("yyyyMMdd-HHmmss") +
                  $ExtOnly;
  Copy-Item $file.FullName
            "Q:\DB-Backups\$($DestFileName)";
  Write-Host " --> Backup erstellt als $($DestfileName)";
}
Write-Host "`n`nSicherung abgeschlossen..."

Statt des Verzeichnisses "D:\Access-DBs" geben Sie das Verzeichnis an, in dem bei Ihnen die zu sichernden Datenbanken gespeichert sind. Das Verzeichnis "Q:\DB-Backups" ersetzen Sie durch das Verzeichnis, in dem auf Ihrem System die Sicherungskopien abgelegt werden sollen.

Im ersten Schritt löscht das Script den Ausgabebereich der PowerShell-Konsole per "Cls", um Verwirrungen mit zuvor ausgegeben Texten und Meldungen zu vermeiden. Die dann folgende Anweisung gibt den Text "Datenbanken sichern..." aus. Das Steuerzeichen "`n" ("`" = Umschalt+´) entspricht einem "vbCrLf" und sorgt für die Ausgabe von zwei Leerzeilen nach dem Titel. Anschließend wird eine "foreach"-Schleife eingeleitet, die mit der Funktion "Get-Item" alle zur Dateimaske "\*.\*db" passenden Dateien (\*.mdb, \*.accdb) im Verzeichnis "D:\Access-DBs" durchläuft und eine Referenz vom Datentyp "FileInfo" in der Variablen "$file" liefert. Die Eigenschaft "FullName" des FileInfo-Objektes nutzen wir, um eine Statusmeldung "Erstelle Backup von..." in der Konsole auszugeben. Anhand der Zeichenfolge "$()" erkennt der PowerShell-Interpreter, dass der Inhalt der in runden Klammern angegebenen Variablen oder das Ergebnis einer Funktion bzw. eines Ausdrucks einzufügen ist.

Die .NET Framework-Bibliothek "[System.IO.Path]" stellt hilfreiche Funktionen für die Verarbeitung von Pfaden und Dateinamen bereit. Mit "GetFileNameWithoutExtension($file)" kann zum Beispiel der Dateiname aus dem als Parameter übergebenen Pfad extrahiert werden. Analog dazu liefert "GetExtension($file)" die Erweiterung (.MDB, .ACCDB) der jeweiligen Datei. Aus diesen in den Variablen "$FileNameOnly" und "$ExtOnly" gespeicherten Informationen wird im nächsten Schritt ein um einen Zeitstempel ergänzter neuer Dateiname in der Variablen "$DestFileName" zusammengesetzt. Den Zeitstempel erhalten wir aus der .NET Framework-Bibliothek "[DateTime]" über die Funktion "Now()", deren Unterfunktion "ToString()" eine beliebige Formatierung erlaubt. Das Ergebnis ist dann zum Beispiel "20121127-121751" für eine Sicherung, die am 27.11.2012 um 12:17:51 Uhr erstellt wurde.

Weiter geht es mit dem Kopieren der jeweiligen Datei über das Cmdlet "Copy-Item". Als Parameter ist der Quellpfad und der Zielpfad zu übergeben. Den Quellpfad liefert die Eigenschaft "FullName" des FileInfo-Objektes "$file", der Zielpfad wird aus einer konstanten Zeichenkette ""Q:\DB-Backups\"" mit der Einbettung eines Variableninhaltes per "$($DestFileName)" gebildet. Nach dem Kopiervorgang geben wir erneut eine Statusmeldung aus, die über Zielpfad und Dateinamen mit Zeitstempel der erstellten Sicherungskopie informiert. Anschließend geht es dann mit der nächsten über "foreach..Get-Item" gefundenen Datei weiter. Finden sich keine weiteren passenden Dateien mehr, hat das Script seine Aufgabe erledigt und gibt eine Abschlussmeldung "Sicherung abgeschlossen..." aus.

Dieses Script können Sie beispielsweise als "Datenbanken sichern.ps1" auf dem Desktop speichern und dort jederzeit per Doppelklick öffnen, um Sicherungskopien Ihrer Datenbanken anzulegen. Wie Sie ein PowerShell-Script automatisch zeitgesteuert ausführen, erläutern wir im in der nächsten Ausgabe.

Datenbanken per PowerShell sichern, Teil 2
Versionen: Access 2013, 2010, 2007, 2003 und 2002/XP

Mit der PowerShell von Windows können Sie leistungsfähige Scripts erstellen, die Routineaufgaben per Mausklick oder automatisch zeitgesteuert erledigen. Eine dieser Aufgaben ist beispielsweise das Anlegen von Sicherungskopien Ihrer Datenbanken. Wie Sie eine solche Lösung umsetzen, erläutert unser zweiteiliger Tipp der Woche.

Bei den folgenden Ausführungen gehen wir davon aus, dass die PowerShell auf Ihrem System installiert, für die Ausführung von PowerShell-Scripts eingerichtet ist und Sie mit dem Script-Editor "PowerShell ISE" (ISE= Integrated Script Environment) umgehen können. Tutorials dazu und weitere Informationen zur PowerShell finden Sie bei Bedarf im Microsoft Script Center unter der folgenden Adresse:

[**http://technet.microsoft.com/de-de/scriptcenter/powershell.aspx**](http://technet.microsoft.com/de-de/scriptcenter/powershell.aspx)

Nachdem wir im ersten Teil dieses Tipps der Woche erläutert haben, wie Sie ein PowerShell-Script zur Sicherung aller Datenbanken eines Verzeichnisses auf dem Desktop ablegen und bei Bedarf per Doppelklick ausführen, geht es nun um die automatische zeitgesteuerte Ausführung. Dazu nutzen Sie die Aufgabenplanung von Windows 7, 2008 oder Vista folgendermaßen (Windows 2003 und XP siehe unten):

1. Klicken Sie auf *Start*, geben Sie im Suchfeld "Aufgabenplanung" ein und drücken Sie die **Return**-Taste.
2. Klicken Sie rechts unter "Aktionen" auf die Schaltfläche *Aufgabe erstellen*.
3. Geben Sie unter "Name" zum Beispiel "Datensicherung Access-Datenbanken" und unter "Beschreibung" einen passenden Hinweistext ein.
4. Wählen Sie unter "Konfigurieren für" die passende Windows-Version aus.
5. Wechseln Sie auf die Registerkarte "Trigger" und klicken Sie auf die Schaltfläche *Neu*.
6. Stellen Sie die gewünschte Uhrzeit, beispielsweise "22:00:00" ein, aktivieren Sie die Option "Wöchentlich" und die gewünschten Tage, zum Beispiel "Montag" bis "Freitag" und klicken Sie auf die Schaltfläche *Ok*.
7. Wechseln Sie auf die Registerkarte "Aktionen" und klicken Sie auf die Schaltfläche *Neu*.
8. Klicken Sie auf die Schaltfläche *Durchsuchen*, lokalisieren Sie das gewünschte PowerShell-Script wie beispielsweise "DBs sichern.ps1" und übernehmen Sie es mit einem Doppelklick.
9. Klicken Sie auf die Schaltfläche *Ok*, um die Einrichtung der Aufgabe abzuschließen.

Unter Windows 2003 und XP richten Sie ein automatisch auszuführendes PowerShell-Script wie folgt ein:

1. Öffnen Sie in der Systemsteuerung das Kontrollfeld "Geplante Tasks" und doppelklicken Sie auf *Geplanten Task hinzufügen*. Es wird ein Assistent gestartet, der Sie durch die weiteren Schritte führt.
2. Bestätigen Sie mit *Weiter*, klicken Sie auf der folgenden Seite auf die Schaltfläche *Durchsuchen*, lokalisieren Sie das gewünschte PowerShell-Script und wählen Sie es per Doppelklick aus.
3. Auf den dann folgenden Seiten geben Sie eine Bezeichnung für den Job wie beispielsweise "Datensicherung Access-Datenbanken" ein, wählen zum Beispiel "Täglich" für die Ausführung des Jobs und stellen die gewünschte Uhrzeit ein. Klicken Sie dann auf die Schaltfläche *Weiter*.
4. Die nächste Seite dient der Festlegung eines Kontos, das für die Ausführung des Jobs verwendet wird. Geben Sie hier den Benutzernamen und das Kennwort des gewünschten Kontos ein und klicken Sie auf die Schaltfläche *Weiter*.
5. Klicken Sie auf die Schaltfläche *Fertigstellen*.

In Zukunft werden nun regelmäßig und automatisch Datensicherungskopien Ihrer Datenbanken zu den festgelegten Zeitpunkten erstellt.