SQL Server Backup Size vs. Database Size Script

Kathegorie
SQL Server

Simon
Schlaipfer

Problem

In der sehr fehleranfälligen Welt der IT sind Vorbereitungen auf gewisse Katastrophenszenarios durch Sicherungen und Backups unerlässlich und gehören, gerade im Bereich der Datanbankadministration, zu einer der wichtigsten Vorkehrungen.

Es gibt jedoch Fälle, in denen eine einfache Sicherungsdatei nicht ausreicht, da möglicherweise im Vorhinein gar nicht bekannt ist, wie viel Speicherplatz tatsächlich für die Dateien der Datenbank (also .mfd- und .idf-Dateien) benötigt wird, bis diese erfolgreich wiederhergestellt wurden.

In einem anderen durchaus denkbaren Szenario könnte sich die Sicherungsdatei an einem Remotestandort befinden. Besteht nun kein Zugriff mehr auf den Datenbankserver, auf dem sich diese befindet, ist die aktuelle Größe der Dateien nicht länger einsehbar.

Lösung

In den folgenden Abschnitten werden wir nun eine Reihe von T-SQL-Skripten vorstellen, mit denen nützliche Informationen für vollständige Sicherungen sowie die tatsächliche Dateigröße nachverfolgt werden können.

Ebenfalls wird das Szenario behandelt, in dem zwar eine Sicherungsdatei zur Verfügung steht, jedoch der Zugriff auf die Quellinstanz nicht möglich ist.

Zur Veranschaulichung der vorgestellten Skripte werden wir gemeinsam mit Ihnen eine leere Datenbank mit dem Namen "LargeDB" erstellen, welche eine primäre .mfd-Datei (80 GB) und eine .Idf-Datei (9 GB) enthält.

Es folgen nun 3 verschiedene Ansätze, die beim Einsehen dieser Informationen helfen können, und das Problem aus verschiedenen Blickwinkeln betrachten.

Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz

info@madafa.de
madafa.de

+49 6131 3331612

Ansatz 1 – Auflisten der Größe der SQL Server-Datenbankdateien und des Backups

Hierfür gehen Sie wie folgt vor:

- Als Erstes erstellen wir eine temporäre Tabelle #FreeSpace, um die Menge an freiem Speicherplatz in jeder Datenbank zu speichern. Diese Information ergibt sich aus der Subtraktion der gesamten Größe der Datenbank (in der Tabelle sys.master_files) und des von der Datenbank verwendeten Speicherplatzes (zurückgeben von der Funktion FILEPROPERTY).
- Da die Funktion FILEPROPERTY nur Informationen aus der Datenbank, zu der eine akute Verbindung besteht, zurückgibt, wird die Logik in dem Befehl sp_MSforeachdb gekapselt, damit die Informationen aller Datenbanken in der Tabelle #FreeSpace für spätere Verwendung gespeichert werden können.
- Nach der Ausführung der gespeicherten Prozedur sp_MSforeachdb erfolgt die Hauptabfrage mithilfe der folgenden Tabellen: sys.databases, sys.master_files und eine Unterabfrage aus dem msdb.dbo.backupset (Zur Erfassung der Sicherungsgröße der letzten vollständigen Sicherung, falls vorhanden)
- Abschließend wird die #FreeSpace Tabelle wieder gelöscht.

Die Ausgabe der Abfrage sollte so aussehen:

	DatabaseName	DataSizeInMB	LogSizeInMB	FreeSpaceInMB	CompressedBackupSizeInMB	State
1	LargeDB	80000.00	9000.00	88996	84.09	ONLINE
2	master	5.38	2.00	2	NULL	ONLINE
3	model	8.00	8.00	13	NULL	ONLINE
4	msdb	19.63	28.81	28	NULL	ONLINE

Es ist sofort ersichtlich, dass die Sicherungsdatei der "LargeDB"-Datenbank 84 MB groß ist, die Größe der enthaltenen Daten jedoch 80 GB und die Protokolldatei 9 GB betragen. Die Sicherung einer 84 MB-großen Datenbank kann also schnell 90 GB Speicherplatz belegen, ohne dass dies direkt ersichtlich ist.

T-SQL Skript zum ersten Ansatz:

```
IF OBJECT_ID('tempdb..#FreeSpace') IS NOT NULL DROP TABLE #FreeSpace

CREATE TABLE #FreeSpace([database] VARCHAR(64) NOT NULL,amount INT NOT NULL)

DECLARE @sqlCommand varchar(2048)

SELECT @sqlCommand = 'USE [?]
          DECLARE @freeSpace INT
          SELECT @freeSpace = SUM(size/128 -(FILEPROPERTY(name,
''SpaceUsed'')/128)) FROM sys.master_files
          INSERT INTO #FreeSpace VALUES(''?'', @freeSpace)
         '

EXEC sp_MSforeachdb @sqlCommand

SELECT DISTINCT
         d.name AS 'DatabaseName',
         (SELECT CONVERT( DECIMAL(10,2),SUM(size)*8.0/1024)
          FROM sys.master_files
          WHERE type_desc = 'ROWS'
            AND database_id = mf.database_id
          GROUP BY database_id) AS 'DataSizeInMB',
         (SELECT CONVERT(DECIMAL(10,2),SUM(size)*8.0/1024)
          FROM sys.master_files
          WHERE type_desc = 'LOG'
             AND database_id = mf.database_id
          GROUP BY database_id) AS 'LogSizeInMB',
         (SELECT amount
          FROM #FreeSpace
          WHERE [database] = d.name) AS 'FreeSpaceInMB',
         CONVERT(DECIMAL(10,2),b.compressed_backup_size/1024.0/1024.0) AS
CompressedBackupSizeInMB,
         d.state_desc AS 'State',
         suser_sname(d.owner_sid) AS 'Owner',
```


Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

```
        d.compatibility_level AS 'CompatibilityLevel',
        d.create_date AS 'DBCreatedDate'
FROM    sys.databases d
JOIN    sys.master_files mf ON d.database_id = mf.database_id
LEFT JOIN (
        SELECT bs.compressed_backup_size,bs.database_name
        FROM msdb.dbo.backupset bs
        WHERE bs.backup_set_id IN (SELECT backup_set_id FROM msdb.
dbo.backupset WHERE backup_start_date = (SELECT MAX(backup_start_
date) FROM msdb.dbo.backupset WHERE database_name = bs.database_
name))
      ) AS b ON b.database_name = d.name
WHERE   d.name NOT IN ('tempdb')
ORDER BY  d.name

DROP TABLE #FreeSpace/
```

Ansatz 2

Der zweite Ansatz verfolgt das selbe Ziel des Ersten, also das Auflisten der Größe der SQL-Server Datenbankdateien und des Backups, jedoch aus der "Perspektive" des Backups.

Der Vorgang:

- Das Ziel der "Common Table Expression" (CTE) mit dem Namen "MostRecentBackups" besteht darin, eine Ergebnismenge zu erstellen, die die neuesten vollständigen Sicherungen enthält, die in der msd, mit dem jeweiligen Datum protokolliert sind, an dem diese Sicherung tatsächlich abgeschlossen wurde.
- Mit diesen Informationen kann nun eine neue CTE mit dem Namen BackupsWithSize erstellt werden, sodass der Rest der Sicherungsinformationen hinzugefügt werden kann (z.B. Größe der Sicherung oder der Pfad, in dem sich die Sicherung befindet, etc.)
- Nachdem nun die Ergebnismenge, die die Sicherungsinformationen enthält, erstellt wurde, können die Informationen zur Größe der Datendatei und der Protokolldatei in die endgültige Ergebnismenge eingefügt werden.

Die Ausgabe der Abfrage sollte so aussehen:

	Database	State	LastFull	TimeSinceLastFullInDays	FullBackupSizeInMB	FullBackupSecondsToComplete
1	LargeDB	ONLINE	2020-02-28 13:37:33.000	0	84.09	1
2	master	ONLINE	NULL	NULL	NULL	NULL
3	model	ONLINE	NULL	NULL	NULL	NULL
4	msdb	ONLINE	NULL	NULL	NULL	NULL

FullBackupLocalPath	DataFileSize	LogFileSize
E:\Program Files\Microsoft SQL Server\MSSQL14.SQ...	80000.00	9000.00
NULL	5.38	2.00
NULL	8.00	8.00
NULL	19.63	28.81

(Aufgrund der Breite der Ausgabe, ist diese hier auf zwei Screenshots aufgeteilt)

Die Ergebnismenge konzentriert sich auf relevante Informationen für die vollständigen Sicherungen, wird jedoch durch die tatsächliche Datengröße, welche die Sicherungsdatei tatsächlich darstellt, ergänzt. Für die "LargeDB" - Datenbank werden also genau die gleichen 80 GB für die Quelldatei und 9 GB für die Protokolldatei angezeigt (die gleichen wie bei Ansatz 1).

T-SQL Skript zum 2. Ansatz:

```
WITH
   MostRecentBackups
   AS(
      SELECT
         database_name AS [Database],
         MAX(bus.backup_finish_date) AS LastBackupTime,
         CASE bus.type
```


Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

```
            WHEN 'D' THEN 'Full'
         END AS Type
      FROM msdb.dbo.backupset bus
      WHERE bus.type <> 'F'
      GROUP BY bus.database_name,bus.type
   ),
   BackupsWithSize
   AS(
      SELECT
               mrb.*,
               (SELECT TOP 1 CONVERT(DECIMAL(10,2), b.compressed_
backup_size/1024/1024) AS backup_size FROM msdb.dbo.backupset b
WHERE [Database] = b.database_name AND LastBackupTime = b.backup_
finish_date) AS [Backup Size],
               (SELECT TOP 1 DATEDIFF(s, b.backup_start_date,
b.backup_finish_date) FROM msdb.dbo.backupset b WHERE [Database]
= b.database_name AND LastBackupTime = b.backup_finish_date) AS
[Seconds],
        (SELECT TOP 1 b.media_set_id FROM msdb.dbo.backupset b
WHERE [Database] = b.database_name AND LastBackupTime = b.backup_
finish_date) AS media_set_id
      FROM MostRecentBackups mrb
   )

SELECT
    d.name AS [Database],
    d.state_desc AS State,
    bf.LastBackupTime AS [LastFull],
    DATEDIFF(DAY,bf.LastBackupTime,GETDATE()) AS
[TimeSinceLastFullInDays],
    bf.[Backup Size] AS [FullBackupSizeInMB],
    bf.Seconds AS [FullBackupSecondsToComplete],
    CASE WHEN DATEDIFF(DAY,bf.LastBackupTime,GETDATE()) > 14 THEN
NULL ELSE (SELECT TOP 1 bmf.physical_device_name FROM msdb.dbo.
backupmediafamily bmf WHERE bmf.media_set_id = bf.media_set_id AND
bmf.device_type = 2) END AS [FullBackupLocalPath],
               (SELECT CONVERT(DECIMAL(10,2),SUM(size)*8.0/1024)
AS size FROM sys.master_files WHERE type = 0 AND d.name = DB_
NAME(database_id)) AS DataFileSize,
    (SELECT CONVERT(DECIMAL(10,2),SUM(size)*8.0/1024) AS size FROM
sys.master_files WHERE type = 1 AND d.name = DB_NAME(database_id))
AS LogFileSize
FROM sys.databases d
LEFT JOIN BackupsWithSize bf ON (d.name = bf.[Database] AND (bf.
Type = 'Full' OR bf.Type IS NULL))
WHERE d.name <> 'tempdb' AND d.source_database_id IS NULL
ORDER BY d.name/
```

Ansatz 3 – Abrufen der Größe der SQL Server-Datenbankdateien aus der Sicherungsdatei

Der dritte und letzte Ansatz basiert ausschließlich auf der Sicherungsdatei:

- Zur Annäherung des Problems verwenden wir den Befehl "RESTORE FILELISTONLY", um so die Größeninformationen direkt aus der Datei zu extrahieren.
- "RESTORE FILELISTONLY" gibt eine Ergebnismenge mit den gewünschten Informationen zurück. Um diese jedoch in einer temporären Tabelle zu speichern und mit den Daten arbeiten zu können, wird der Befehl als Parameter eines EXEC-Aufrufs ausgeführt.
- Hierbei gibt es ein wichtiges Detail zu beachten:
  Ab SQL Server 2016 hat Microsoft der vom FILELISTONLY-Befehl zurückgegebenen Ergebnismenge eine zusätzliche Spalte hinzugefügt.

Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

T-SQL Skript zum 3. Ansatz:

```
-- enter the path and file name of the backup
DECLARE @filename nvarchar(500) = 'E:\Program Files\Microsoft SQL
Server\MSSQL14.SQL_2017_1\MSSQL\Backup\test.bak'

IF NOT EXISTS (SELECT * FROM dbo.sysobjects where id = object_
id(N'RestoreFilelistOnly') and OBJECTPROPERTY(id, N'IsTable') = 1)
BEGIN
SET NOCOUNT ON;
DECLARE @sqlCommand NVARCHAR(2048);

IF(
   (SELECT
      CASE
         WHEN CONVERT(VARCHAR(128), SERVERPROPERTY
('PRODUCTVERSION')) LIKE '8%'    THEN 0
         WHEN CONVERT(VARCHAR(128), SERVERPROPERTY
('PRODUCTVERSION')) LIKE '9%'    THEN 0
         WHEN CONVERT(VARCHAR(128), SERVERPROPERTY
('PRODUCTVERSION')) LIKE '10.0%' THEN 0
         WHEN CONVERT(VARCHAR(128), SERVERPROPERTY
('PRODUCTVERSION')) LIKE '10.5%' THEN 0
         WHEN CONVERT(VARCHAR(128), SERVERPROPERTY
('PRODUCTVERSION')) LIKE '11%'   THEN 0
         WHEN CONVERT(VARCHAR(128), SERVERPROPERTY
('PRODUCTVERSION')) LIKE '12%'   THEN 0
         WHEN CONVERT(VARCHAR(128), SERVERPROPERTY
('PRODUCTVERSION')) LIKE '13%'   THEN 1
         WHEN CONVERT(VARCHAR(128), SERVERPROPERTY
('PRODUCTVERSION')) LIKE '14%'   THEN 1
         WHEN CONVERT(VARCHAR(128), SERVERPROPERTY
('PRODUCTVERSION')) LIKE '15%'   THEN 1
         ELSE 1
      END
   ) = 0
)

SET @sqlCommand = '
CREATE TABLE ##RestoreFilelistOnly (
    [LogicalName]           NVARCHAR(128),
    [PhysicalName]          NVARCHAR(260),
    [Type]                  CHAR(1),
    [FileGroupName]         NVARCHAR(128),
    [Size]                  NUMERIC(20,0),
    [MaxSize]               NUMERIC(20,0),
    [FileID]                BIGINT,
    [CreateLSN]             NUMERIC(25,0),
    [DropLSN]               NUMERIC(25,0),
    [UniqueID]              UNIQUEIDENTIFIER,
    [ReadOnlyLSN]           NUMERIC(25,0),
    [ReadWriteLSN]          NUMERIC(25,0),
    [BackupSizeInBytes]     BIGINT,
    [SourceBlockSize]       INT,
    [FileGroupID]           INT,
    [LogGroupGUID]          UNIQUEIDENTIFIER,
    [DifferentialBaseLSN]   NUMERIC(25,0),
    [DifferentialBaseGUID]  UNIQUEIDENTIFIER,
    [IsReadOnly]            BIT,
    [IsPresent]             BIT,
    [TDEThumbprint]         VARBINARY(32)
)'

ELSE
SET @sqlCommand = '
CREATE TABLE ##RestoreFilelistOnly (
    [LogicalName]           NVARCHAR(128),
    [PhysicalName]          NVARCHAR(260),
    [Type]                  CHAR(1),
    [FileGroupName]         NVARCHAR(128),
```


Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

```
    [Size]                  NUMERIC(20,0),
    [MaxSize]               NUMERIC(20,0),
    [FileID]                BIGINT,
    [CreateLSN]             NUMERIC(25,0),
    [DropLSN]               NUMERIC(25,0),
    [UniqueID]              UNIQUEIDENTIFIER,
    [ReadOnlyLSN]           NUMERIC(25,0),
    [ReadWriteLSN]          NUMERIC(25,0),
    [BackupSizeInBytes]     BIGINT,
    [SourceBlockSize]       INT,
    [FileGroupID]           INT,
    [LogGroupGUID]          UNIQUEIDENTIFIER,
    [DifferentialBaseLSN]   NUMERIC(25,0),
    [DifferentialBaseGUID]  UNIQUEIDENTIFIER,
    [IsReadOnly]            BIT,
    [IsPresent]             BIT,
    [TDEThumbprint]         VARBINARY(32),
    [SnapshotUrl]           NVARCHAR(360)
)'

EXEC sp_executesql @sqlCommand;

INSERT INTO ##RestoreFilelistOnly EXEC('RESTORE FILELISTONLY FROM
DISK = ''' + @filename + '''')

SELECT PhysicalName, CONVERT(DECIMAL(10,3),(Size/1024/1024)) as
FileSizeMB, CONVERT(DECIMAL(10,3),(BackupSizeInBytes/1024/1024)) as
BackupSizeMB
FROM ##RestoreFilelistOnly

DROP TABLE ##RestoreFilelistOnly

END
```

Hier haben wir eine leere Datenbank namens ”test“ mit einer Quelldatei (.mdf-Datei) von 6 GB und einer Protokolldatei (.ldf-Datei) von 10 MB erstellt. Nach dem Ausführen einer vollständigen Sicherung zum Testen des obigen Skripts sehen Sie hier die Ausgabe:

	PhysicalName	FileSizeMB	BackupSizeMB
1	E:\Program Files\Microsoft SQL Server\MSSQL14.SQ...	6000.000	8.000
2	E:\Program Files\Microsoft SQL Server\MSSQL14.SQ...	10.000	0.000

Und tatsächlich stimmen die Informationen genau mit der Größe unserer ”test“-Datenbank überein.

Nächste Schritte

- Allgemein ist es am sinnvollsten, sich für Ansatz 1 oder 2 zu entscheiden und deren Ergebnisse für eine spätere Analyse in einer Tabelle zu speichern. Ansatz 3 zielt auf die Lösung des Szenarios ab, in dem nur die Sicherungsdatei zur Verfügung steht und ansonsten nichts weiter über die Datenbank bekannt und kein Zugriff auf die SQL Server-Quellinstanz möglich ist. Nur dann sollte dieser gewählt werden.
- Mithilfe einer Automatisierung durch PowerShell können Ergebnismengen für eine Reihe von Instanzen erfasst und zur Nachanalyse in einer zentralen Datenbank gespeichert werden. So auch die Ergebnisse aus Ansatz 1 und 2.
- Der Code aus Ansatz 3 sollte vor der Durchführung einer Wiederherstellung auf einem anderen Server ausgeführt werden. Auch wenn die Sicherungsdatei klein aussieht, kann die wiederhergestellte Datenbank aufgrund von Komprimierung wesentlich größer sein.

backup, database size, size, sql backup size, sql server, sql server backup

Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612