Installing SQL Storage Compress

Before we learn more about how to use SQL Storage Compress, we need to take a quick look at how you install it. Fortunately, it is a quick and easy procedure.

SQL Storage Compress comes in two different files: the "Hyperbac Installer" and the "SQL Storage Compress" setup program. The "Hyperbac Installer" is executed first, and then the "SQL Storage Compress" setup program is run. Installing both only takes a minute or two and does not require a reboot. Once they are installed, and assuming you have purchased a license, all you need to do is to enter the activation code, and you are on your way.

SQL Storage Compress works with the Developer, Standard, and Enterprise editions of SQL Server 2000 (SP 3 or later), SQL Server 2005, SQL Server 2008, and SQL Server 2008 R2. Operating systems supported include XP Professional (SP1 & SP2), Vista, Windows 7, Windows 2000 (SP 4), Windows Server 2003 (SP 1 & SP2), Windows Server 2008, and Windows Server 2008 R2.

Compressing your database

Once SQL Storage Compress has been installed, you have two options for compressing a database's MDF and NDF files. The option you choose depends upon whether you are creating a new database from scratch, or you want to convert an existing database to a SQL Storage Compress compressed database. Let's look at each option in turn.

Creating a new database

Creating a new SQL Storage Compress compressed database uses exactly the same CREATE DATABASE syntax as when creating any new database in SQL Server. The only difference is the file extension you give to the MDF or NDF files.

For example, let's say I want to create a 1 GB database with a 100 MB log file: I execute the following Transact-SQL code:

```
USE master ;
GO
CREATE DATABASE SQL_Storage_Compress_Demo ON
( NAME = ssc_demo_dat,
    FILENAME = 'E:\ssc_demo.mdfx',
    SIZE = 1000MB) LOG ON
( NAME = ssc_demo_log,
    FILENAME = 'E:\ssc_demo.ldf',
    SIZE = 100MB) ;
GO
```

As soon as the database is created, I can run the standard Disk Usage report from SQL Server Management Studio (SSMS), and see that my database was created as I expected.

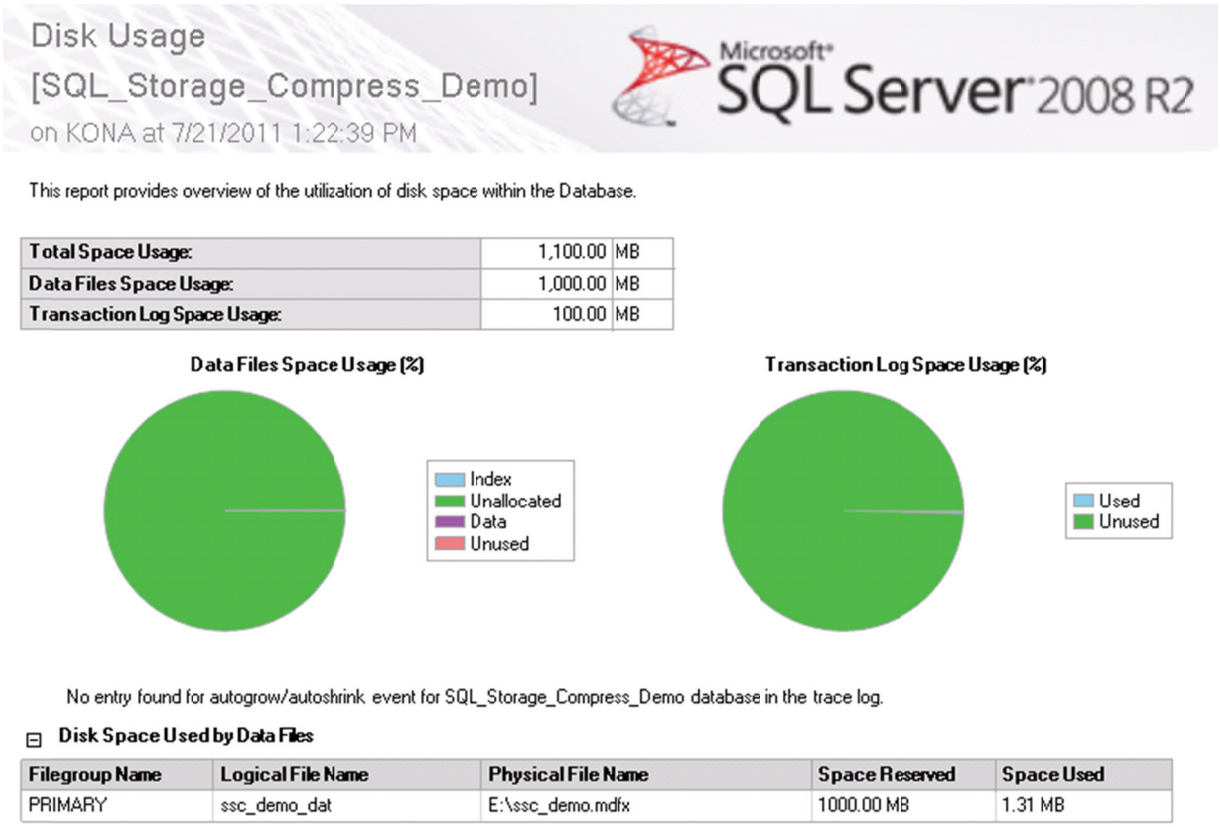

Figure 3: The Disk Usage report tells us that SQL Server thinks it has created a 1 GB database with a 100 MB log file

So where's the compression? Wasn't SQL Storage Compress supposed to compress the MDF file? Yes, and the MDF file was, indeed, compressed. To prove it, let's look at the MDF and LDF files from Windows Explorer.

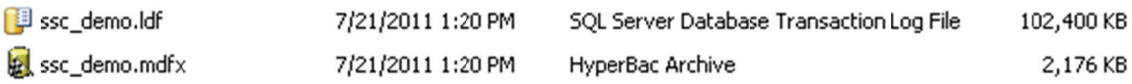

Figure 4: Showing that the new MDF file we created was compressed

So, if SQL Storage Compress compressed the MDF file, why doesn't this show up in the Disk Usage Report? Remember, SQL Server doesn't know about the compression. Since the details of compression are hidden from SQL Server, the Disk Usage Reports show us what SQL Server thinks it sees. Of course, we know better, because we can see the size of the actual MDF file. In fact, if you do the math, the compression of the MDF file before adding any data is about 99.8%. Naturally, when data is added to the database, the compression ratio will go down, but all the data added to the new database will be automatically compressed. The amount of compression will depend on the amount of free space in the MDF and NDF files, and the compressibility of the data.

I still have a little more explaining to do, and that involves discussing the significance of the .mdfx and .ndfx extensions. SQL Storage Compress has been configured so that when it sees a file using these extensions, it knows that the data stored in these SQL Server databases are to be compressed. That's all, nothing fancy. And as you probably know, SQL Server doesn't care what the file extensions of database files are, so any file extension can be used. With SQL Storage Compress we simply add the letter "x" after MDF and NDF.

Within SSMS, whenever you work with system views or DMVs on a database compressed by SQL Storage Compress, don't forget that SQL Server always displays the disk sizes it thinks it sees, not the actual size of the data files. Use Windows Explorer to view the actual sizes of the compressed MDF and NDF files, or the new graphical user interface (GUI) included with SQL Storage Compress 6.0.

Compressing an existing database

In many cases, you will want to compress an existing database using SQL Storage Compress. This is a simple, one-time step; essentially, all you have to do is to back up your original database and then restore it. Once this step has been completed, the original database can be deleted, as it is no longer needed.

For example, let's say that you want to use SQL Storage Compress to compress the AdventureWorks database. The first step is to back up the database. You can back it up using the native SQL Server BACKUP command; or, if you are running SQL Server 2008 Enterprise Edition, or if you have the Standard or Enterprise Edition SQL Server 2008 R2, you can use the native backup compression. You can also back up the database using SQL HyperBac, SQL Backup Pro, or even some other third-party backup tools.

Once you have a backup, you restore the database using the native Transact-SQL RESTORE command (or the normal syntax used for restoring using third-party backup tools, such as stored procedure calls). The syntax of the command must include the MOVE clause and the file extensions must be changed from MDF and NDF to MDFX and NDFX, respectively. For example, the following Transact-SQL code creates a SQL Storage Compress version of the AdventureWorks database that was created using the native SQL Server BACKUP command:

```
RESTORE DATABASE [AdventureWorks_Compressed] FROM DISK =
    'D:\AdventureWorks.bak' WITH MOVE 'AdventureWorks_Data' TO
    'C:\AdventureWorks_Data.mdfx', MOVE 'AdventureWorks_Log' TO
    'C:\AdventureWorks_Log.ldf'
```

Notice that the RESTORE syntax is standard Transact-SQL. The only difference between this code and any RESTORE code you normally might write is the name of the MDF file extension. You can restore the database to any SQL Server instance you want that has SQL Storage Compress installed on it.

As the database is restored, it is automatically compressed by SQL Storage Compress. When the RESTORE is complete, the MDF and NDF files will have been compressed and the database will work just like it did before, except that it will now take up much less storage space.