

HOW TO SETUP FILESTREAM FEATURE ON MICROSOFT SQL SERVER 2008/2012/2014

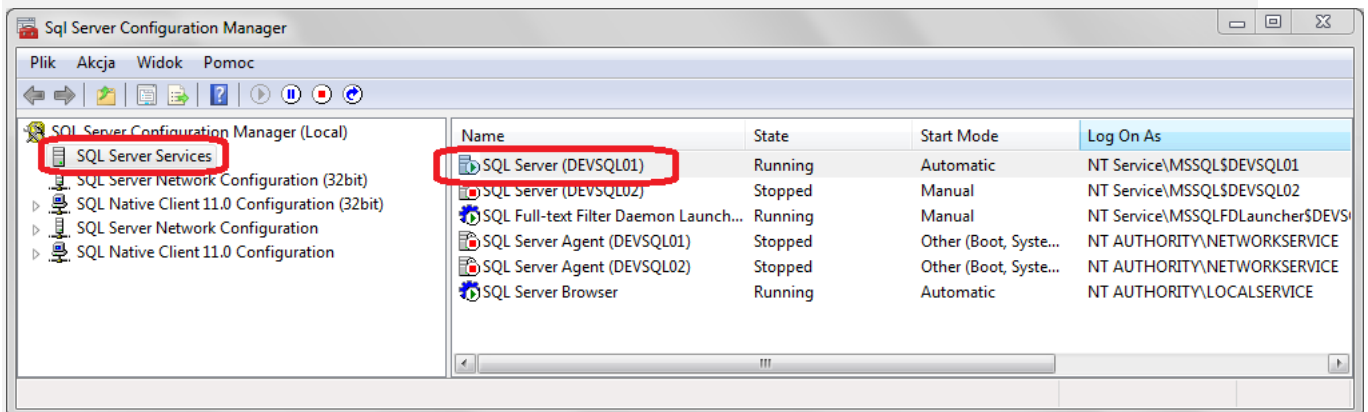
- The **FILESTREAM** feature allows for storing and managing unstructured data in SQL Server.
 - In more technical terms,
 - “The **FILESTREAM** feature allows storing **BLOB** data (example: word documents, image files, music and videos etc) in the NT file system and ensures transactional consistency between the unstructured data stored in the NT file system and the structured data stored in the table.”
- You enable **FILESTREAM** first at the SQL Server level, then at the DB level.

DBA-presents gives a clear demonstration:

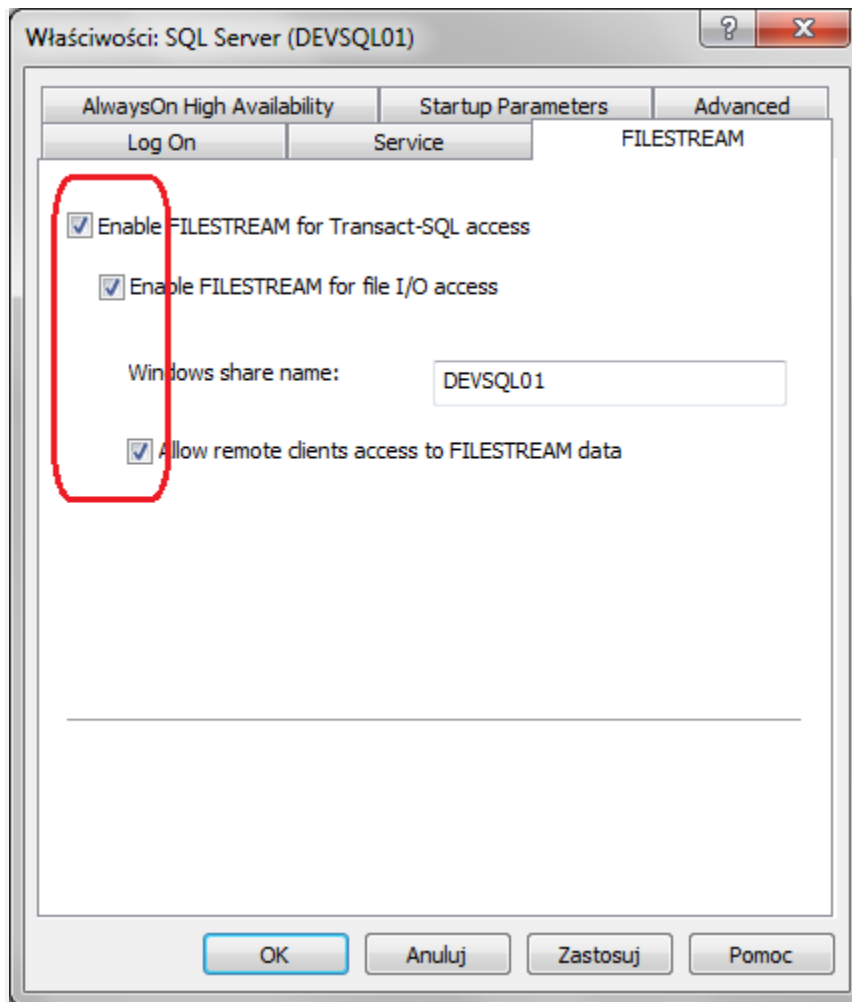
- <http://dba-presents.com/index.php/databases/sql-server/59-introduction-to-filestream>

Enabling FILESTREAM at SQL Server Level

Before **FILESTREAM** can be used, it has to be enabled on the instance. To do this, go to Configuration Manager, select *SQL Server Services* and double click the instance you would like to have **FILESTREAM** enabled.



The properties window should show up. Go to the **FILESTREAM** tab and check *Enable FILESTREAM for Transact-SQL access*, *Enable FILESTREAM for file I/O access* and *Allow remote clients access to FILESTREAM data*. Not all three options are required to be chosen, for more details refer to the documentation.



Note:

- Under “WindowsShare name,” you are not specifying an existing share name; you are also not specifying the name of a folder; you are simply specifying a “name” which SQL Server will use to create a virtual share on the SQL Server instance to perform FILESTREAM operations (Transaction consistency, security, etc).
- The created Virtual Share won’t be mapped directly to the file system (like a normal share); again, it is a virtual device which holds NTFS “Data Containers.”
- When you create a FILESTREAM Filegroup in a DB, the FILEGROUP is what we call the “Data Container” which is mapped to whatever device you choose; here you would specify the folder you wish to store disk files in;
- When creating a table within the FILESTREAM enabled DB (or FILEGROUP), you would specify a column – and this could would serve to hold the unstructured data.
 - Ex: `[ItemImage] VARBINARY(MAX) FILESTREAM NULL`

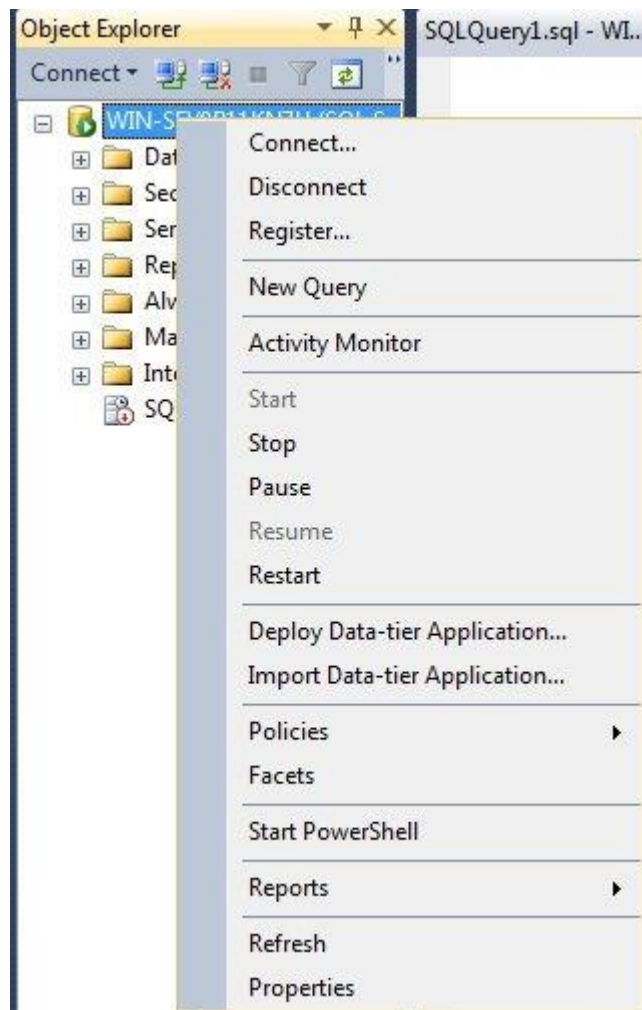
Once it is done, restart the SQL Server service.

- VoiceoftheDBA gives you an illustration of DB level setup:
 - <https://voiceofthedba.wordpress.com/2012/03/07/enabling-filestream-in-sql-server-2012/>

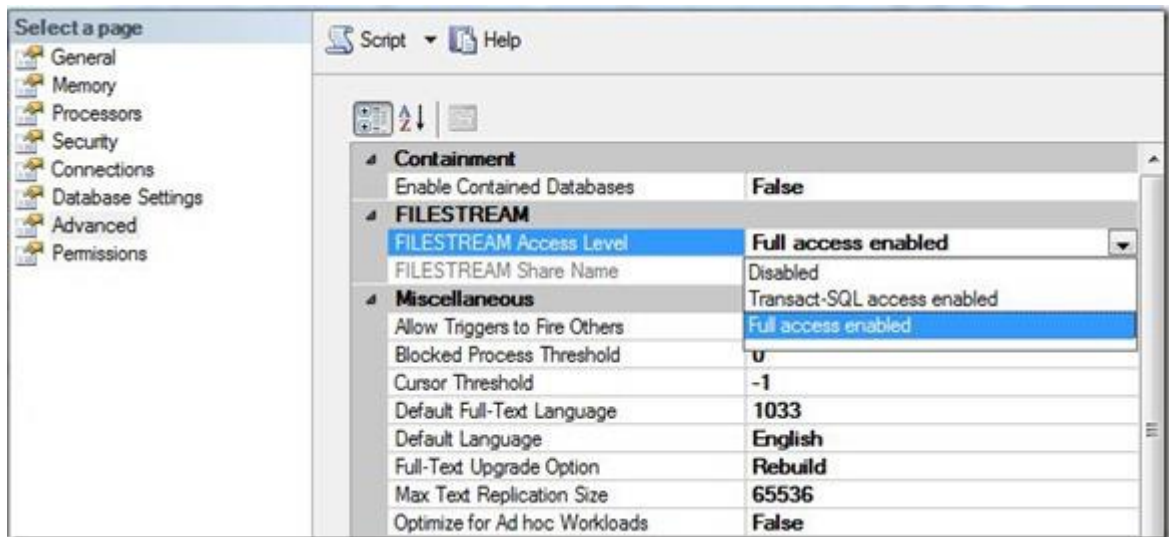
Open SQL Server Management Studio, connect to the instance and execute the following script that sets access level.

```
EXEC sp_configure filestream_access_level, 2  
RECONFIGURE
```

Or via User Interface, Right-click on the “SQL Server” and select “Properties”



If you choose the “Advanced” item on the left, you will get a list of properties for the instance. Near the top, there is the FILESTREAM section. Below I have dropped down the choices. By default, this is disabled, and for FILESTREAM you can select either of the other options, but FileTable needs the full access.



Once this is done, you need to restart the instance to enable the Filestream for the SQL Server. This doesn't set up FILESTREAM in any of your databases; this merely enables it for the instance. You need to still create the FILESTREAM filegroups in any database that will use FILESTREAM data.

- After enabling FILESTREAM feature at the DB level, you would create a DB specifying a FILEGROUP which will bear the location where you are planning to store the disk files.

```
Create database
On Primary
(NAME=TEST_MDF, FILENAME=C:\Temp\Test_Regular_MDF_File.MDF'),
FILEGROUP FileGroupName CONTAINS FILESTREAM
( NAME = FileGroupName, FILENAME = 'C:\Temp\FS'),
LOG ON
(NAME = TEST_LDF, FILENAME = 'c:\Temp\Test_Regular_Log_File.ldf')
```

Folder **c:\Temp\FS** should not exist; **the CREATE statement is creating this folder**. This folder (**C:\temp\FS**) is called a **“Filestream Data Container.”**


- If the DB already exists and you wish to add FileStream feature to a FileGroup, follow these steps:

```
-- Add a FileGroup containing FileStream feature
ALTER DATABASE DB_ForFileStream
ADD FILEGROUP MckLABAK4_FG
```

Folder **c:\Temp** exist; **the CREATE statement is creating this folder** (**C:\temp**) **“Filestream Data**

CONTAINS FILESTREAM

```
-- Add File (Disk folder) to FILESTREAM Filegroup
ALTER DATABASE DB_ForFileStream
ADD FILE
(NAME = 'MckLABAK4_FG', FILENAME='S:\QADB3\FS')
TO FILEGROUP MckLABAK4_FG
```



- If you wish to Remove the FILEGROUP,
ALTER DATABASE DB_ForFileStream
REMOVE FILEGROUP MckLABAK4_FG

- **The term “FILESTREAM” doesn’t refer to a “data type;” it is rather an attribute that can be assigned to a VARBINARY (MAX) Column.**

```
CREATE TABLE TABLE_ForFileStream
(
    fID UNIQUEIDENTIFIER ROWGUIDCOL NOT NULL UNIQUE DEFAULT
    NEWSEQUENTIALID(),
    fData VARBINARY(Max) Filestream null
)
```

```
CREATE TABLE [dbo].[FS_Table](
    [ItemID] UNIQUEIDENTIFIER ROWGUIDCOL NOT NULL UNIQUE DEFAULT NEWSEQUENTIALID(),
    [ItemNumber] VARCHAR(20),
    [ItemDescription] VARCHAR(50),
    [ItemImage] VARBINARY(MAX) FILESTREAM NULL)
```

- - At this point, the column “fItemImage” becomes a “Filestream enabled column.”
 - Any data you store in this column will be stored in the NT File System as a disk file.
 - A pointer to this disk file will be stored in the table.
 - You can access the “disk file” or unstructured data by referencing the table “FS_Table” and/or column “fItemImage.”
 - **Example:**
 - *Select * from FS_Table*
 - **Note:** every table that has a FILESTREAM column should have a **UNIQUEIDENTIFIER** column with **ROWGUIDCOL** and **UNIQUE** attributes.

- **Restrictions on the BLOB?**

- Not many!
- The file can be as big as the space allocated on the disk. SQL Server doesn’t care about the size of the file.
- This disk space, where FILEGROUP Data container is, can also be compressed and even shared (in a cluster environment).

- **History of how unstructured data is stored on a MS SQL Server (*The Filestream feature was introduced in 2008*)**

SQL Server	Approach used:	PROs	CONs
BEFORE SQL SERVER 2008	STORING DATA IN VARBINARY OR IMAGE COLUMN.	*ENSURES TRANSACTIONAL CONSISTENCY. *REDUCES MANAGEMENT COMPLEXITIES.	POOR PERFORMANCE.
BEFORE SQL SERVER 2008	STORE DATA AS FILES ON A DISK AND STORE THE FILE LOCATION ON A TABLE.	GOOD PERFORMANCE	*POOR TRANSACTIONAL CONSISTENCY. *MANAGEMENT (BACKUP & RESTORE, SECURITY) IS NOT EASY.
SQL SERVER 2008/2012/2014	FILESTREAM FEATURE. SQL SERVER & DB ARE FILESTREAM ENABLED.	*TRANSACTIONAL CONSISTENCY. *IMPROVED PERFORMANCE *QUERY LANGUAGE TO SEARCH, RETRIEVE, AND SAVE DISK FILES.	

Using INSERT, UPDATE and DELETE to manage SQL Server FILESTREAM Data

- **Inserting FILESTREAM Data to [TABLE_ForFileStream] Table**

```
Use [DB_ForFileStream]
GO
INSERT INTO [TABLE_ForFileStream] (fData)
SELECT * FROM
OPENROWSET(BULK N'C:\SampleFiles\Image1.JPG' ,SINGLE_BLOB)
AS Document
GO
```

- **Retrieve FILESTREAM Data from [TABLE_ForFileStream] Table**

```
o USE [DB_ForFileStream]
GO
SELECT * FROM [FS_Table]
GO
```

- **Updating FILESTREAM Data stored in [TABLE_ForFileStream] Table**

```
o USE [DB_ForFileStream]
GO
UPDATE [TABLE_ForFileStream]
```

```

SET [fData] = (SELECT *
FROM OPENROWSET (
BULK 'C:\SampleFiles\Image2.JPG',
SINGLE_BLOB) AS Document)
WHERE ID = <guid>
GO

```

- **Deleting FILESTREAM Data stored in [TABLE_ForFileSTream] Table**

```

o USE [DB_ForFileStream]
GO
DELETE [TABLE_ForFileSTream]
WHERE ID = <guid>
GO

```

Resources: <https://www.mssqltips.com/sqlservertip/1850/using-insert-update-and-delete-to-manage-sql-server-filestream-data/>

PERFORMANCE CONSIDERATIONS:

- It's better to access FILESTREAM data via win32 streaming as opposed to t-sql.
 - o When accessing FILESTREAM data via t-sql, SQL Server uses local resources to read the content of the file and hands it to the client.
 - o When accessing FILESTREAM via win32 streaming, SQL Server memory is not used but rather the application server's memory.
 - o Win32 streaming leverages streaming capabilities embedded in NT File system.

FILESTREAM Feature Summary

- FILESTREAM feature is available with all versions of SQL Server 2008, including SQL Server Express.
- SQL Server Express database has a 4 GB limitation; however this limitation does not apply to the FILESTREAM data stored in a SQL Server Express database.
- FILESTREAM Columns can be replicated.
- FILESTREAM enabled databases can be used with LOG Shipping
- FILESTREAM columns can be used in Full Text Indexes
- FILESTREAM works with all recovery models
- FILESTREAM File Groups can be placed on compressed disk volumes
- The maximum size of the file that can be stored into the FILESTREAM data storage is limited by the size of the disk volume only.

Restrictions on existing features

Though the FILESTREAM feature smoothly integrates with many of the existing features of SQL Server, it adds restrictions to or completely disables a few other important features of SQL Server.

- A FILESTREAM enabled database cannot be used for mirroring. This is one of the key restrictions that I dislike. Mirroring is a very interesting feature and it cannot be used with FILESTREAM.
- FILESTREAM data is not available in database snapshots. If you create database snapshots and run a “SELECT * FROM table” query on a table with FILESTREAM columns, you will get an error. All queries that you run on a FILESTREAM enabled table of a database snapshot should exclude FILESTREAM columns.

FILESTREAM Limitations

- FILESTREAM columns cannot be used in Index Keys.
- FILESTREAM columns cannot be used in the INCLUDED columns of a Non Clustered Index
- FILESTREAM columns cannot be used in a Table Valued Parameter
- FILESTREAM columns cannot be used in a memory table
- FILESTREAM columns cannot be used in a global or local temp table
- Statistics cannot be created on FILESTREAM columns
- Computed columns having reference to FILESTREAM columns cannot be indexed
- When FILESTREAM Data is accessed through Win 32 APIs, only READ COMMITTED ISOLATION level is supported.
- FILESTREAM data can be stored only on local disk volumes
- FILESTREAM data is not supported on Database Snapshots.
- Transparent Data Encryption (TDE) does not encrypt FILESTREAM data.

FILESTREAM Feature – Points to Remember

Take note of the following points if you intend to use FILESTREAM in your application.

- You cannot do database mirroring if you use FILESTREAM features

- If you replicate FILESTREAM enabled columns, make sure that all subscribers are running on SQL Server 2008 or later versions.
- If you intend to use LOG shipping, make sure that both primary and secondary servers are running on SQL Server 2008 or later versions.
- If you are on a FAILOVER CLUSTERING environment, make sure that you place the FILESTREAM file groups in a shared disk and FILESTREAM should be enabled on each node.
- You cannot access FILESTREAM data if you create database snapshots. If you try to access a FILESTREAM enabled column, SQL Server will raise an error.
- SQL Server Instance should be configured with Integrated Security if Win 32 access to the FILESTREAM data is required

FILESTREAM Best Practices

- Place each FILESTREAM data container in a separate volume
- Use the correct RAID level depending upon the nature of the application (read intensive, write intensive), expected work load etc
- Do periodical disk defragmentation
- Decide diligently on whether or not to use a compressed disk volume
- Disable 8.3 names in NTFS
- Disable last access time tracking in NTFS
- FILESTREAM data container should not be on a fragmented volume
- Make sure that the data being stored is appropriate for FILESTREAM storage. FILESTREAM storage is only good if the size of the data is more than 1 MB (approx)
- Avoid multiple small appends to the FILESTREAM file
- If FILESTREAM files are large, avoid using TSQL access to the FILESTREAM data.
- If reads require only the first few bytes, then consider using TSQL access using substring() function
- If the entire file is needed Win 32 access is desirable

Resources: <https://www.simple-talk.com/sql/learn-sql-server/an-introduction-to-sql-server-filestream/>