راه آهن جمهوری اسلامی ایران

بنام خدا

مقاله علمی با موضوع:

آشنایی با قابلیت FileStream در SQL server

ارائه دهنده:

مهدی عبدالکریمی

کارشناس واحد برنامه ریزی

اداره کل واگنهای راه آهن جمهوری اسلامی ایران

پاییز ۱۳۹۶

معایب ذخیره فایلها در دیتابیس:

الف) اختصاص یافتن قسمتی از بافر SQL Server به این امر.
ب) با توجه به قرار گرفتن داده‌های BLOB در دیتابیس ، transaction logقابل توجهی تولید خواهد ش .
ج) بیش از ۲ GBرا نمی‌توان در فیلدهایی از نوع varbinary(max) ذخیره کرد.
د) به روز رسانی BLOB ها سبب ایجاد fragmentation می‌شود.

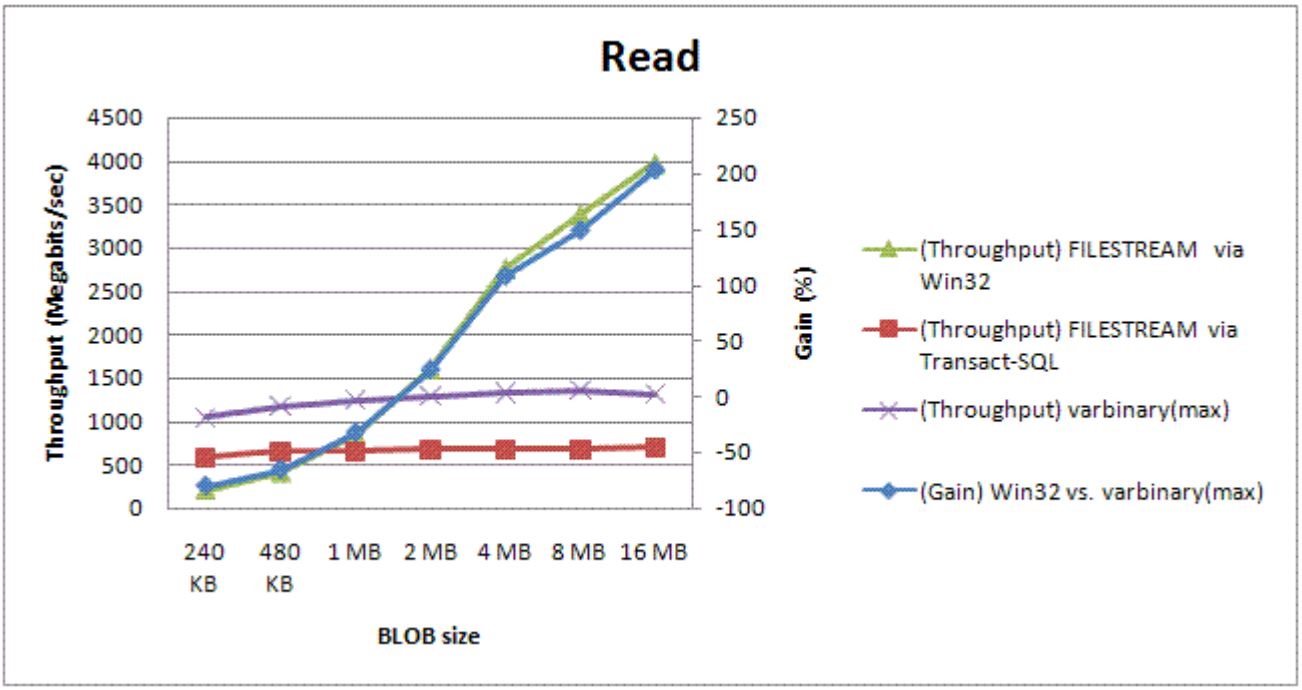
مایکروسافت برای رفع این مشکلات در SQL Server 2008 قابلیت جدیدی را ارائه داده است به نام FileStream که در طی مقالاتی به بررسی آن خواهیم پرداخت.

FILESTREAMموتور دیتابیس اس کیوال سرور را با سیستم فایل NTFS یکپارچه می‌کند؛ به این صورت که داده‌های BLOB از نوع varbinary(max)را به صورت فایل بر روی سیستم ذخیره خواهد کرد. سپس با استفاده از دستورات T-SQL می‌توان این فایل‌ها را ثبت، حذف، به روز رسانی، جستجو و بک آپ گیری کرد. این قابلیت نیز از فیلدهای varbinary(max) استفاده می‌کند؛ اما اکنون ویژگی و برچسب FILESTREAM به این نوع فیلدها الصاق خواهد شد .باید در FILESTREAM data FILESTREAM filegroups ذخیره شوند FILESTREAM filegroups .در حقیقت همان پوشه‌های فایل سیستم می‌باشند. به آن‌ها data containers نیز گفته می‌شوند که مرزی هستند بین ذخیره سازی داده‌ها در فایل سیستم و در دیتابیس.

مزایای سیستم FileStream چیست؟
الف) سیستم transaction مختص به خود را داشته، به همین جهت سبب رشد غیر منطقی حجم فایل transaction log دیتابیس اصلی نمی‌شوند.
ب) هنگام به روز رسانی فیلدهایی از این دست، صرفا ایجاد یا حذف یک فایل مد نظر است؛ بنابراین fragmentation ایجاد شده در این حالت بسیار کمتر از روش استفاده از فیلدهایی از نوع varbinary(max) می‌باشد.
ج) استفاده از NT system cache جهت کش کردن اطلاعات که سبب بالا بردن بازدهی بانک اطلاعاتی خواهد شد.
د) از buffer pool اس کیوال سرور در این حالت استفاده نشده (مطابق قسمت ج) و این حافظه جهت امور روزمره‌ی اس کیوال سرور کاملا مهیا خواهد بود.
ه) محدودیت ۲ GBفیلدهایی از نوع varbinary(max) با توجه به ذخیره سازی این نوع BLOBs در فایل سیستم، دیگر وجود نخواهد داشت.

چه زمانی بهتر است از FileStream استفاده شود؟
الف) فایل‌هایی که ذخیره می‌شوند به طور متوسط بیش از یک مگابایت حجم داشته باشند. (برای کمتر از این مقدار varbinary(max) BLOBsکارآیی بهتری را ارائه می‌دهند). هر چند این مرز یک مگابایت مطابق اطلاعات books online است اما تجربیات کاری نشان می‌دهند که این سقف را باید ۲۵۶ کیلوبایت درنظر گرفت.
ب) قابلیت خواندن سریع اطلاعات فایل‌ها مد نظر باشد (بررسی کارآیی مطابق تصویر زیر از .(MSDN سیستم NTFS نسبت به SQL Server در خواندن فایل‌های حجیم سریعتر عمل می‌کند.
ج) اگر از یک معماری middle tier در برنامه‌های خود در حال استفاده‌اید.

د) زمانیکه نیاز باشد تا اطلاعات relational و non-relational در یک تراکنش مورد استفاده قرار گیرند.



نکاتی را که باید هنگام ذخیره سازی اطلاعات در FileStream در نظر داشت

الف) هنگامی که یک جدول حاوی فیلدی از نوع FileStream می‌باشد، باید دارای فیلد ID منحصربفرد نیز باشد.

ب data containers (ایی که پیش از این در مورد آن‌ها صحبت شد، نباید تو در تو باشند.

ج FILESTREAM filegroups (بر روی درایوهای فشرده شده نیز می‌توانند قرار داشته باشند.

FileStreamاز دیدگاه امنیت

امنیت داده‌های FileStream در اس کیوال سرور دقیقا همانند امنیت سایر اطلاعات ذخیره شده در دیتابیس است (دسترسی در حد جدول و یا فیلد). اگر کاربری دسترسی به فیلد FileStream در یک جدول داشته باشد، می‌تواند آن فایل را گشوده و استفاده کند. رمزنگاری بر روی این ستون‌ها پشتیبانی نمی‌شود. تنها اکانتی که اس کیوال سرور تحت آن در حال اجرا است دسترسی به FILESTREAM container دارد. همچنین توصیه شده است که به هیچ اکانت دیگری این دسترسی داده نشود. زمانیکه یک دیتابیس آغاز و مشغول به کار می‌شود، اس کیوال سرور دسترسی به FILESTREAM data container را محدود خواهد کرد و دسترسی به این اطلاعات تنها از طریق دستورات T-SQL و یا OpenSqlFilestream APIمیسر خواهد بود. بدیهی است زمانیکه اس کیوال سرور متوقف شود، این اطلاعات بدون هیچگونه محدودیتی قابل دسترسی بوده و تنها محدودیت‌های سیستمی به آن‌ها اعمال خواهند شد (که این مورد باید مد نظر باشد).

نگهداری FileStream

FileStreamبه صورت فیلدهای varbinary(max) یکپارچه با دیتابیس ذخیره می‌شود؛ بنابراین نحوه‌ی تهیه پشتیبان از آن‌ها همانند

روش‌های متداول است بدون هیچگونه تغییری (و این اطلاعات در بک آپ دیتابیس لحاظ می‌شوند). اگر نیاز بود هنگام تهیه پشتیبان از این نوع داده‌ها بک آپ گرفته نشود، می‌توان از partial backup با پارامترهای مربوطه استفاده کرد.

در این قسمت نحوه‌ی فعال سازی قابلیت FileStream را بررسی خواهیم کرد و در قسمت بعدی نحوه‌ی دسترسی به آن‌را از طریق برنامه نویسی مرور می‌نمائیم.

فعال سازی قابلیت FileStream

همانند اکثر قابلیت‌های اس کیوال سرور، فعال سازی FileStream نیز حداقل به دو صورت استفاده از GUI و قابلیت‌های management studio میسر است و یا استفاده از دستورات T-SQL و البته کتابخانه‌ی SMO یا همان محصور کننده‌ی توانایی‌های management studio نیز قابل استفاده است.

روش اول) استفاده از management studio

قابلیت FileStream به صورت پیش فرض غیرفعال است. برای فعال سازی آن به مسیر زیر مراجعه نمائید:

Start > All Programs > Microsoft SqlServer 2008 > Configuration Tools > SQL Server Configuration Manager

سپس در قسمت SQL Server services ، وهله مربوط به SQL Server را یافته، کلیک راست و به برگه خواص آن مراجعه کرده (شکل زیر) و قابلیت FileStream را فعال کنید:

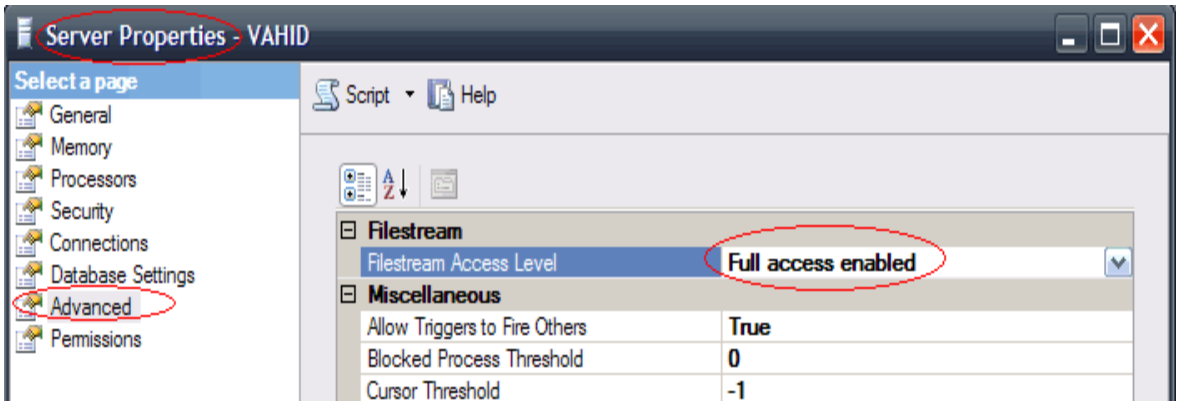
گزینه‌های مختلف آن به شرح زیر هستند:

• Enable FileStream for transact-sql access : امکان استفاده از دستورات T-SQL را جهت دسترسی به فایل‌ها فعال می‌سازد (یا برعکس)

• Enable FileStream for File I/O streaming access : امکان دسترسی به فایل‌ها با استفاده از Win32 streaming access

• All remote clients to have streaming access to file stream data : اجازه‌ی دسترسی به کلاینت‌های راه دور جهت استفاده از قابلیت FileStream

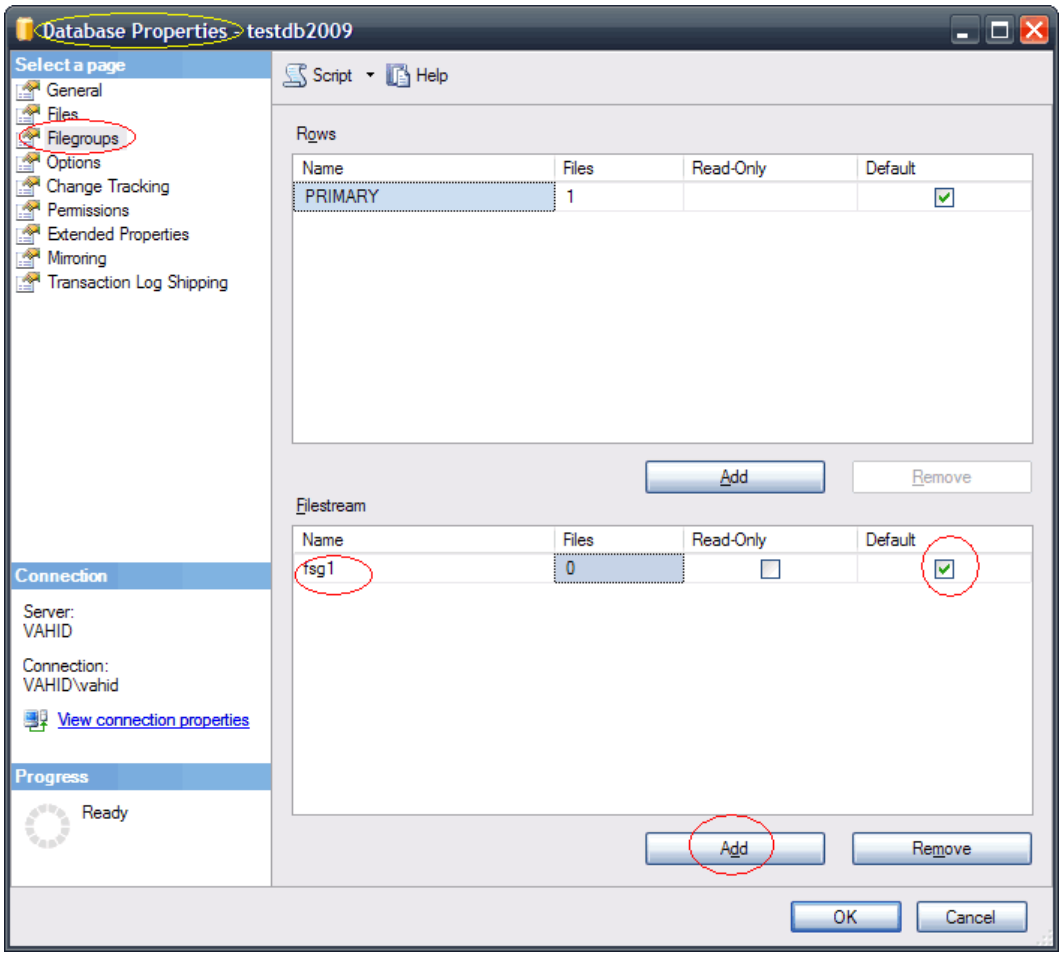
مرحله بعد، فعال سازی سطح دسترسی به سرور است. به management studio مراجعه نمائید. سپس بر روی وهله سرور مورد نظر کلیک راست نموده و به خواص آن مراجعه کنید (شکل زیر). سپس در قسمت advanced سطح دسترسی را بر روی Full قرار دهید.



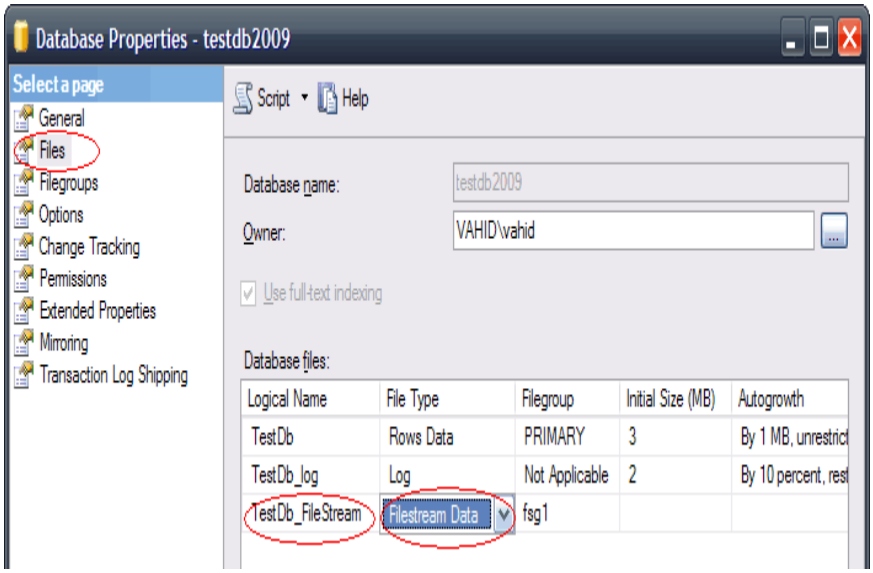
پس از این تنظیم به شما پیغام داده خواهد شد که باید دیتابیس سرور را یکبار راه اندازی مجدد نمائید تا تنظیمات مورد نظر، اعمال شوند.

در ادامه باید دیتابیسی را که نیاز است نوع داده FileStream را بپذیرد، تنظیم نمود.

بر روی دیتابیس مورد نظر کلیک راست کرده و در برگه خواص آن به گزینه‌ی Filegroups مراجعه کنید. سپس در اینجا یک گروه جدید را اضافه کرده ، نامی دلخواه را وارد نموده و سپس تیک مربوط به default بودن آن‌را نیز قرار دهید (شکل زیر):



سپس در همین برگه‌ی خواص دیتابیس که باز است، به گزینه‌ی Files مراجعه کنید. در اینجا سه کار را باید انجام دهید. ابتدا بر روی دکمه Add کلیک کرده و در قسمت logical name ردیف اضافه شده، نامی دلخواه را وارد کنید. سپس file type آن را بر روی FileStream قرار دهید. در ادامه به قسمت path در همین ردیف مراجعه نموده و مسیر ذخیره سازی را مشخص کنید. در پایان بر روی دکمه‌ی OK کلیک نمائید تا کار تنظیم دیتابیس به پایان رسد (شکل زیر):



روش دوم) استفاده از دستورات T-SQL

منهای قسمت تنظیمات SQL Server Configuration Manager که باید از طریق روش عنوان شده صورت گیرد، سایر موارد فوق را با استفاده از دستورات T-SQL نیز می‌توان انجام داد:

الف) تنظیم سطح دسترسی بر روی سرور

```
EXEC sp_configure filestream_access_level, 2 -- 0 : Disable , 1 : Transact Sql Access , 2 : Win IO Access
GO
RECONFIGURE
GO
```

ب) تنظیمات دیتابیس

اگر نیاز باشد دیتابیس جدیدی ایجاد شود: (ایجاد گروه فایل مربوطه و سپس تنظیمات مسیر آن)

```
CREATE DATABASE Test_Db
ON
PRIMARY ( NAME = TestDb1,
   FILENAME = 'C:\DATA\Test_Db.mdf'),
FILEGROUP FileStreamGroup1 CONTAINS FILESTREAM( NAME = Testfsg1,
   FILENAME = 'C:\DATA\Learning_DbStream')
LOG ON  ( NAME = TestDbLog1,
    FILENAME = 'C:\DATA\Test_Db.ldf')
GO
```

و یا ایجاد تغییرات بر روی دیتابیسی موجود: (ایجاد گروه فایل مخصوص و سپس افزودن فایل مربوطه و تنظیمات آن)

```
--add filegroup
```

```
alter database TestDb
 Add FileGroup FileStreamFileGroup1 contains FileStream
go

--Add FileGroup To DB
alter database TestDB
 add file
 (
   name = 'UserDocuments'  ,
   filename = 'C:\FileStream\UserDocuments'
 ) to filegroup FileStreamFileGroup1
```

تعریف جدولی آزمایشی به همراه فیلدی از نوع: FileStream

تا اینجا سرور و همچنین دیتابیس جهت پذیرش این نوع داده آماده شدند. اکنون نوبت به استفاده از آن است:

```
CREATE TABLE [tblFiles]
(
 FileId      UNIQUEIDENTIFIER NOT NULL ROWGUIDCOL UNIQUE DEFAULT(NEWID()),
 Title       NVARCHAR(255) NOT NULL,
 SystemFile  VARBINARY(MAX) FILESTREAM NULL
)
ON [PRIMARY] FILESTREAM_ON [fsg1]
```

توسط دستور T-SQL فوق جدولی که از نوع داده FileStream استفاده می‌کند، ایجاد خواهد شد. این جدول همانطور که مشخص است حتما باید دارای یک فیلد منحصربفرد باشد (ر.ک. مقاله قبل) و همچنین برچسب فایل استریم به فیلدی از نوع VARBINARY(MAX) نیز الصاق شده است. به علاوه گروه فایل آن نیز باید به صورت صریح مشخص گردد؛ که در مثال ما مطابق تصاویر به fsg1 تنظیم شده بود.

در انتهای قسمت قبل، نحوه‌ی ایجاد یک جدول جدید با فیلدی از نوع فایل استریم بررسی شد، حال اگر جدولی از پیش وجود داشت، نحوه‌ی افزودن فیلد ویژه مورد نظر به آن، به صورت زیر است:

```
alter table tbl_files set(filestream_on ='default')

go
alter table tbl_files
 add

   [systemfile] varbinary(max) filestream  null  ,
   FileId  uniqueidentifier  not null rowguidcol unique default (newid())
go
```

در ادامه جدول tblFiles قسمت قبل را در نظر بگیرید:

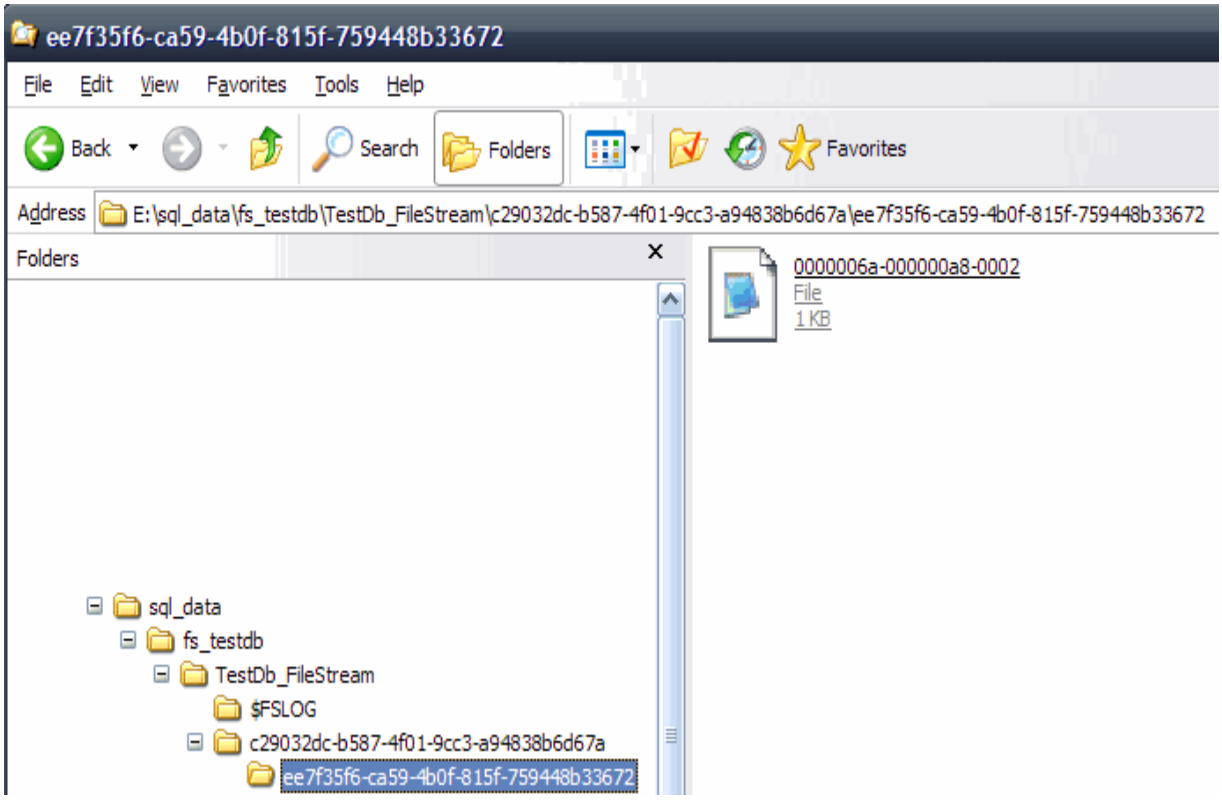
```
CREATE TABLE [tblFiles](
 [FileId] [uniqueidentifier] ROWGUIDCOL  NOT NULL,
 [Title] [nvarchar](255) NOT NULL,
 [SystemFile] [varbinary](max) FILESTREAM  NULL,
UNIQUE NONCLUSTERED
(
 [FileId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON,
ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY] FILESTREAM_ON [fsg1]

ALTER TABLE [dbo].[tblFiles] ADD  DEFAULT (newid()) FOR [FileId]
GO
```

نحوه‌ی افزودن رکوردی جدید به جدول: tblFiles

```
INSERT INTO [tblFiles]
 (
   [Title],
   [SystemFile]
 )
VALUES
 (
   'file-1',
   CAST('data data data' AS VARBINARY(MAX))
 )
```

در اینجا سعی کرده‌ایم یک رشته ساده را در فیلدی از نوع فایل استریم ذخیره کنیم که روش کار به صورت فوق است. از آنجائیکه مقدار پیش فرض FileIdرا هنگام تعریف جدول به NEWID تنظیم کرده‌ایم، نیازی به ذکر آن نیست و به صورت خودکار محاسبه و ذخیره خواهد شد.
اگر کنجکاو باشید که این فایل اکنون کجا ذخیره شده و نحوه‌ی مدیریت آن توسط اس کیوال سرور به چه صورتی است، فقط کافی است به مسیری که هنگام افزودن گروه فایل‌ها و فایل مربوطه در تنظیمات خواص دیتابیس در قسمت قبل مشخص کردیم، مراجعه کرد (شکل زیر).



بدیهی است افزودن یک رشته به این صورت کاربرد عملی ندارد و صرفا جهت یک مثال ارائه شد. در ادامه، نحوه‌ی ثبت محتویات یک فایل را در فیلدی از نوع فایل استریم و سپس خواندن اطلاعات آن‌را از طریق برنامه نویسی بررسی خواهیم کرد:

```
using System;
using System.IO;
using System.Data.SqlClient;
using System.Data;

namespace FileStreamTest
{
   class CFS
   {
       /// <summary>
       /// افزودن رکورد به جدول حاوی ستونی از نوع فایل استریم
       /// </summary>
       /// <param name="filePath">مسیر فایل</param>
       /// <param name="title">عنوانی دلخواه</param>
       public static void AddNewRecord(string filePath, string title)
```

```
{
    //آیا فایل وجود دارد؟
    if (!File.Exists(filePath))
        throw new FileNotFoundException(
            "لطفا مسیر فایل معتبری را مشخص نمائید", filePath);

    //خواندن اطلاعات فایل در آرایه‌ای از بایت‌ها
    byte[] buffer = File.ReadAllBytes(filePath);

    using (SqlConnection objSqlCon = new SqlConnection())
    {
        //todo: کانکشن استرینگ باید از یک فایل کانفیگ خوانده شود
        objSqlCon.ConnectionString =
            "Data Source=(local);Initial Catalog=testdb2009;Integrated Security = true";
        objSqlCon.Open();

        //شروع یک تراکنش
        using (SqlTransaction objSqlTran = objSqlCon.BeginTransaction())
        {
            //ساخت عبارت افزودن پارامتری
            using (SqlCommand objSqlCmd = new SqlCommand(
                "INSERT INTO [tblFiles]([Title],[SystemFile]) VALUES(@title , @file)",
                objSqlCon, objSqlTran))
            {
                objSqlCmd.CommandType = CommandType.Text;

                //تعریف وضعیت پارامترها و مقدار دهی آن‌ها
                objSqlCmd.Parameters.AddWithValue("@title", title);
                objSqlCmd.Parameters.AddWithValue("@file", buffer);

                //اجرای فرامین
                objSqlCmd.ExecuteNonQuery();
            }

            //پایان تراکنش
            objSqlTran.Commit();
        }
    }
}

/// <summary>
/// دریافت اطلاعات فایل ذخیره شده به صورت آرایه‌ای از بایت‌ها
/// </summary>
/// <param name="fileId">کلید مورد استفاده</param>
```

```
        /// <returns></returns>
        public static byte[] GetDataFromDb(string fileId)
        {
            byte[] data = null;

            using (SqlConnection objConn = new SqlConnection())
            {
                //کوئری اس کیوال پارامتری جهت دریافت محتویات فایل
                string cmdText = "SELECT SystemFile FROM tblFiles WHERE FileId=@id";
                using (SqlCommand objCmd = new SqlCommand(cmdText, objConn))
                {
                    //todo: کانکشن استرینگ باید از یک فایل کانفیگ خوانده شود
                    objConn.ConnectionString =
                        "Data Source=(local);Initial Catalog=testdb2009;Integrated Security = true";
                    objConn.Open();

                    //تنظیم کردن وضعیت و مقدار پارامتر تعریف شده در کوئری
                    objCmd.Parameters.AddWithValue("@id", fileId);

                    //اجرای فرامین و دریافت فایل
                    using (SqlDataReader objread = objCmd.ExecuteReader())
                    {
                        if (objread != null)
                            if (objread.Read())
                            {
                                if (objread["SystemFile"] != DBNull.Value)
                                    data = (byte[])objread["SystemFile"];
                            }
                    }
                }
            }

            return data;
        }
    }
}
```

مثالی در مورد روش استفاده از کلاس فوق:

```
using System.IO;

namespace FileStreamTest
{
```

```
    class Program
    {
        static void Main(string[] args)
        {
            CFS.AddNewRecord(@"C:\filest05.PNG", "test1");

            //آی دی رکورد ذخیره شده در دیتابیس برای مثال
            byte[] data = CFS.GetDataFromDb("BB848D45-382C-4D95-BF4E-۵۲۳۵۰۹۴۰۷۳۴");
            if (data != null)
            {
                File.WriteAllBytes(@"C:\tst.PNG", data);
            }
        }
    }
}
```

روش فوق با روش متداول افزودن یک فایل به دیتابیس اس کیوال سرور هیچ تفاوتی ندارد و این‌جا هم بدون مشکل کار می‌کند. اطلاعات نهایی به صورت فایل‌هایی بر روی سیستم که توسط اس کیوال سرور مدیریت خواهند شد و با جدول شما یکپارچه‌اند، ذخیره می‌شوند.

در روش دیگری که در اکثر مقالات مرتبط مورد استفاده است، از شیء SqlFileStream کمک گرفته شده و نحوه‌ی انجام آن نیز به صورت زیر می‌باشد.

در ابتدا دو رویه ذخیره شده زیر را ایجاد می‌کنیم:

```
CREATE PROCEDURE [AddFile](@Title NVARCHAR(255), @filepath VARCHAR(MAX) OUTPUT)
AS
BEGIN
   SET NOCOUNT ON;

   DECLARE @ID UNIQUEIDENTIFIER
   SET @ID = NEWID()

   INSERT INTO [tblFiles]
     (
       [FileId],
       [title],
       [SystemFile]
     )
   VALUES
     (
       @ID,
       @Title,
       CAST('' AS VARBINARY(MAX))
     )
```

```
    SELECT @filepath = SystemFile.PathName()
    FROM   tblFiles
    WHERE  FileId = @ID
END
GO

CREATE PROCEDURE [GetFilePath](@Id VARCHAR(50))
AS
BEGIN
    SET NOCOUNT ON;

    SELECT SystemFile.PathName()
    FROM   tblFiles
    WHERE  FileId = @ID
END
```

در رویه ذخیره شده AddFile ، ابتدا رکوردی بر اساس عنوان دلخواه ورودی با یک فایل خالی ایجاد می‌شود. سپس مسیر سیستمی این فایل را در آرگومان خروجی filepath قرار می‌دهیم SystemFile.PathName .از اس کیوال سرور ۲۰۰۸ جهت فیلدهای فایل استریم به اس کیوال سرور اضافه شده است. از این مسیر در برنامه خود جهت نوشتن بایت‌های فایل مورد نظر در آن توسط شیء SqlFileStream استفاده خواهیم کرد.

رویه ذخیره شده GetFilePath نیز تنها مسیر سیستمی فایل استریم ذخیره شده را بر می گرداند.

به این ترتیب کدهای برنامه به صورت زیر تغییر خواهند کرد:

```
using System.Data.SqlClient;
using System.Data;
using System.Data.SqlTypes;
using System.IO;

namespace FileStreamTest
{
    class CFSqlFileStream
    {
        /// <summary>
        /// افزودن رکورد به جدول حاوی ستونی از نوع فایل استریم
        /// </summary>
        /// <param name="filePath">مسیر فایل</param>
        /// <param name="title">عنوانی دلخواه</param>
        public static void AddNewRecord(string filePath, string title)
        {
            //آیا فایل وجود دارد؟
```

```
            if (!File.Exists(filePath))
                throw new FileNotFoundException(
                    "لطفا مسیر فایل معتبری را مشخص نمائید", filePath);

            //خواندن اطلاعات فایل در آرایه ای از بایتها
            byte[] buffer = File.ReadAllBytes(filePath);

            using (SqlConnection objSqlCon = new SqlConnection())
            {
                //todo: کانکشن استرینگ باید از یک فایل کانفیگ خوانده شود
                objSqlCon.ConnectionString =
                    "Data Source=(local);Initial Catalog=testdb2009;Integrated Security = true";
                objSqlCon.Open();

                //شروع یک تراکنش
                using (SqlTransaction objSqlTran = objSqlCon.BeginTransaction())
                {
                    //استفاده از رویه ذخیره شده افزودن فایل
                    using (SqlCommand objSqlCmd = new SqlCommand(
                                "AddFile", objSqlCon, objSqlTran))
                    {
                        objSqlCmd.CommandType = CommandType.StoredProcedure;

                        //مشخص ساختن وضعیت و مقدار پارامتر عنوان
                        SqlParameter objSqlParam1 = new SqlParameter("@Title", SqlDbType.NVarChar,
255);
                        objSqlParam1.Value = title;

                        //مشخص ساختن پارامتر خروجی رویه ذخیره شده
                        SqlParameter objSqlParamOutput = new SqlParameter("@filepath",
SqlDbType.VarChar, -1);
                        objSqlParamOutput.Direction = ParameterDirection.Output;

                        //افزودن پارامترها به شیء کامند
                        objSqlCmd.Parameters.Add(objSqlParam1);
                        objSqlCmd.Parameters.Add(objSqlParamOutput);

                        //اجرای رویه ذخیره شده
                        objSqlCmd.ExecuteNonQuery();

                        //و سپس دریافت خروجی آن
                        string Path = objSqlCmd.Parameters["@filepath"].Value.ToString();

                        // زمینه تراکنش فایل استریم موجود را دریافت کرده و از آن برای نوشتن
محتویات فایل استفاده خواهیم کرد
```

```
                //این مورد نیز یکی از تازه‌های اس کیوال سرور ۲۰۰۸ است
                using (SqlCommand objCmd = new SqlCommand(
                    "SELECT GET_FILESTREAM_TRANSACTION_CONTEXT()", objSqlCon, objSqlTran))
                {
                    byte[] objContext = (byte[])objCmd.ExecuteScalar();
                    using (SqlFileStream objSqlFileStream =
                        new SqlFileStream(Path, objContext, FileAccess.Write))
                    {
                        objSqlFileStream.Write(buffer, 0, buffer.Length);
                    }
                }
            }

            objSqlTran.Commit();
        }
    }
}

/// <summary>
/// دریافت اطلاعات فایل ذخیره شده به صورت آرایه‌ای از بایت‌ها
/// </summary>
/// <param name="fileId">کلید مورد استفاده</param>
/// <returns></returns>
public static byte[] GetDataFromDb(string fileId)
{
    byte[] buffer = null;

    using (SqlConnection objSqlCon = new SqlConnection())
    {
        //todo: کانکشن استرینگ باید از یک فایل کانفیگ خوانده شود
        objSqlCon.ConnectionString =
            "Data Source=(local);Initial Catalog=testdb2009;Integrated Security = true";
        objSqlCon.Open();

        //شروع یک تراکنش
        using (SqlTransaction objSqlTran = objSqlCon.BeginTransaction())
        {
            //استفاده از رویه ذخیره شده دریافت مسیر فایل
            using (SqlCommand objSqlCmd =
                new SqlCommand("GetFilePath", objSqlCon, objSqlTran))
            {
                objSqlCmd.CommandType = CommandType.StoredProcedure;

                //مشخص ساختن پارامتر ورودی رویه ذخیره شده و مقدار دهی آن
```

```
                    SqlParameter objSqlParam1 = new SqlParameter("@ID", SqlDbType.VarChar, 50);
                    objSqlParam1.Value = fileId;
                    objSqlCmd.Parameters.Add(objSqlParam1);

                    //اجرای رویه ذخیره شده و دریافت مسیر سیستمی فایل استریم
                    string path = string.Empty;
                    using (SqlDataReader sdr = objSqlCmd.ExecuteReader())
                    {
                        sdr.Read();
                        path = sdr[0].ToString();
                    }

                    // زمینه تراکنش فایل استریم موجود را دریافت کرده و از آن برای خواندن
محتویات فایل استفاده خواهیم کرد
                    //این مورد نیز یکی از تازه‌های اس کیوال سرور ۲۰۰۸ است
                    using (SqlCommand objCmd = new SqlCommand(
                        "SELECT GET_FILESTREAM_TRANSACTION_CONTEXT()", objSqlCon, objSqlTran))
                    {
                        byte[] objContext = (byte[])objCmd.ExecuteScalar();

                        using (SqlFileStream objSqlFileStream =
                            new SqlFileStream(path, objContext, FileAccess.Read))
                        {
                            buffer = new byte[(int)objSqlFileStream.Length];
                            objSqlFileStream.Read(buffer, 0, buffer.Length);
                        }
                    }
                }

                objSqlTran.Commit();
            }
        }

        return buffer;
    }
  }
}
```