**كيفية تسجيل ملفات في قاعدة بيانات Sql Server**

**Mohammad Elsheimy**

2009/07/29

## نظرة خاطفة

درسنا اليوم يتكلم عن طريقة تسجيل البيانات Binary Data في قاعدة بيانات SQL Server. فمثلا بدلا من تسجيل النصوص والأرقام فقط، درسنا اليوم يتكلم عن طريقة أخرى لتسجيل البيانات وهي تسجيل ملفات أو بالأصح بيانات ثنائية Binary Data مثل ملفات الصور والفيديو والـ Word وملفات الكتابة وجميع الملفات. فجميع الملفات كما نعرف هي ملفات ثنائية Binary. يشرح الدرس أولا مقدمة عن هذه الملفات وكيفية التعامل معها في SQL Server ثم ندخل في طريقة تسجيل هذه البيانات واسترجاعها.

مع الدرس مثال يشرح كيفية تسجيل الملفات واسترجاعها من قاعدة بيانات SQL Server.

## مقدمة

جميع الملفات على الجهاز هي كما نعرف ملفات ثنائية أو Binary، فكما تعرف أن جميع البيانات يتم تمثيلها داخل الجهاز بنظام ثنائي فمثلا الرقم 5 يتم تمثيلها هكذا 101 والحرف a له الكود 65 فلهذا يتم تمثيله هكذا 1000001.

> *يمكنك استخدام الآلة الحاسبة Calculator في التحويلات. فقط قم بالتغيير إلى الوضع "علمي Scientific".*

وهذه البيانات يطلق عليها لفظ BLOBs أي Binary Large Objects أي العناصر الثنائية الكبيرة. ونرمز بمصطلح BLOB إلى جميع البينات من ملفات كتابة وصور وفيديو وألعاب وبرامج وغيرها فهي جميعها ملفات (أو عناصر) ثنائية كبيرة الحجم.

لماذا يختلف التعامل مع بيانات BLOB عن غيرها من البيانات الأخرى في SQL Server؟ بالطبع البيانات BLOB يتم تسجيلها بطريقة مختلفة عن طريقة تسجيل البينات من الأنواع الأخرى مثل النصوص والأرقام وذلك بسبب شيئين:

1. في الغالب أنت لا تعرف حجم البيانات التي يتم تسجيلها. بعكس مثلا إذا كنت تحاول تسجيل نص مثلا اسم شخص في قاعدة البيانات فأنت تطلب من المستخدم إدخال اسمه والذي يجب أن لا يزيد عن 30 حرف مثلا.
2. السبب الثاني وهو أن البيانات هذه تكون أحيانا حجمها كبير جدا فمثلا تخيل أنك تحاول أن تقوم بتسجيل ملف فيديو حجمه 50 ميجا بايت! فماذا الحل؟ بالطبع أولا، سوف يزيد حجم قاعدة البيانات بحجم هذا الملف المراد تسجيله. وثانيا عند التعامل معه بالطبع أنت لا يمكنك التعامل مع 50 ميجا بايت مرة واحدة فهذا سوف يسبب بطء وفي الغالب توقف برنامجك، ولهذا سوف تحتاج إلى تسجيله واسترجاعه دفعة Chunk بدفعة أخرى. فمثلا بدلا من تسجيل 50 ميجا بايت مرة واحدة يمكنك عمل دوارة Loop تقوم بتسجيل 100 ك.ب. في المرة مثلا. (وهذا أيضا يعتبر إلى حد ما رقم كبير جدا).

*للأسف Access لا يمكنه تسجيل بيانات من هذا النوع.*

في شرحنا في هذا الدرس سوف نشرح على قاعدة بيانات SQL Server بسيطة جدا يمكنك إنشائها بنفسك وتسمى FileStore. هذه القاعدة لا تحتوي إلا على جدول واحد اسمه MyFiles وهذا الجدول يحتوي على فقط عمودين الأول اسمه Filename ويحتوي على اسم الملف وهو من نوع نصي أي nvarchar بالحجم 260 وبالطبع هو المفتاح الأساسي (PK)Primary Key لهذا الجدول. أما العمود الآخر فهو يسمى Data وهو من نوع image ويستخدم هذا العمود لتسجيل بيانات الملف. **لاحظ أنه النوع image للعمود يستخدم لتسجيل البيانات binary أي بيانات BLOB.**

## تسجيل بيانات BLOB

الكود التالي هو كود بسيط جدا عبارة عن دالة لها مدخل Argument واحد وهو عبارة عن اسم الملف المراد تسجيله. تقوم هذه الدالة بقراءة هذا الملف وتسجيله في قاعدة البيانات. لاحظ الكود.

*بالطبع لا تنسى إضافة Reference للمكتبة dll.Data.System وبالطبع يتبعها dll.Xml.System. ولا تنسى أيضا إضافة جملتي using (أو Imports في NET.VB) للـ Namespaces المطلوبة وهي SqlClient.Data.System و IO.System.*

**كود**

```
// C# Code
   static void StoreFile(string filename)
   {  SqlConnection connection = new SqlConnection
     ("Server=(local); Initial Catalog = FileStore; Integrated Security =
```

```
SSPI");

     SqlCommand command = new SqlCommand
          ("INSERT INTO MyFiles VALUES (@Filename, @Data)", connection);
     command.Parameters.AddWithValue("@Filename",
                                     Path.GetFileName(filename));
     command.Parameters.AddWithValue("@Data",
                                      File.ReadAllBytes(filename));

       connection.Open();
       command.ExecuteNonQuery();
       connection.Close();
   }
```

**كود**

```
   ' VB.NET Code
   Sub StoreFile(ByVal filename As String)
       Dim connection As New SqlConnection( _
           "Server=(local); Initial Catalog = FileStore; Integrated
Security = SSPI")

       Dim command As New SqlCommand( _
           "INSERT INTO MyFiles VALUES (@Filename, @Data)", connection)

       command.Parameters.AddWithValue("@Filename",
Path.GetFileName(filename))
       command.Parameters.AddWithValue("@Data", File.ReadAllBytes(filename))

    connection.Open()

    command.ExecuteNonQuery()

    connection.Close()
  End Sub
```

شرح الكود

قمنا أولا بإنشاء Connection لقاعدة البيانات ثم قمنا بعد ذلك بإنشاء العنصر SqlCommand والذي سيحوي أمر SQL المراد تنفيذه لإضافة البيانات.

بالطبع، ولأن هذا هو التطبيق الأفضل، قمنا بإستخدام مدخلات Parameters لأمر SQL بدلا من كتابتها في الأمر نفسه وذلك أأمن وأفضل وأسرع ولهذا استخدمنا عناصر من نوع SqlParameter.

أخيرا وبعد فتح الاتصال مع القاعدة وتنفيذ الأوامر قمنا بإغلاق الاتصال مع القاعدة.

يفضل إغلاق جميع الاتصالات Connections وجميع الأوامر Commands التي تم إنشائها وبالطبع جميع العناصر التي تتعامل مع قواعد البيانات وتوفر لنا أمر للإغلاق مثل أمر Close() أو Dispose() وذلك توفيرا لموراد الجهاز والحفاظ على أداء برنامجك.

## استرجاع بيانات BLOB

الكود التالي يوضح كيفية استرجاع البينات التي قمنا بإدخالها مسبقا. لاحظ أن الكود يأخذ اسم الملف كمدخلات ويقوم بإرجاع بيانات هذا الملف في هيئة مصفوفة Array من نوع Byte.

**كود**

```
// C# Code
   static byte[] RetrieveFile(string filename)
   {
       SqlConnection connection = new SqlConnection(
          "Server=(local); Initial Catalog = FileStore; Integrated Security
= SSPI");

       SqlCommand command = new SqlCommand(
          "SELECT * FROM MyFiles WHERE Filename=@Filename", connection);

       command.Parameters.AddWithValue("@Filename", filename);

       connection.Open();

       SqlDataReader reader =
command.ExecuteReader(System.Data.CommandBehavior.SequentialAccess);

       reader.Read();

       MemoryStream memory = new MemoryStream();

       long startIndex = 0;
       const int ChunkSize = 256;
       while (true)
       {
         byte[] buffer = new byte[ChunkSize];
         long retrievedBytes = reader.GetBytes(1, startIndex,
                                                buffer, 0, ChunkSize);

           memory.Write(buffer, 0, (int)retrievedBytes);
           startIndex += retrievedBytes;

           if (retrievedBytes != ChunkSize)
               break;
       }

       connection.Close();
       byte[] data = memory.ToArray();
       memory.Dispose();

       return data;
   }
```

**كود**

```
  ' VB.NET Code
  Function RetrieveFile(ByVal filename As String) As Byte()
       Dim connection As New SqlConnection( _
           "Server=(local); Initial Catalog = FileStore; Integrated
Security = SSPI")

       Dim command As New SqlCommand( _
           "SELECT * FROM MyFiles WHERE Filename=@Filename", connection)

       command.Parameters.AddWithValue("@Filename", filename)

       connection.Open()

       Dim reader As SqlDataReader = command.ExecuteReader _
           (System.Data.CommandBehavior.SequentialAccess)

       reader.Read()
       Dim memory As New MemoryStream()
       Dim startIndex As Long = 0
       Const ChunkSize As Integer = 256
       While (True)
        Dim buffer(ChunkSize) As Byte
        Dim retrievedBytes As Long = reader.GetBytes(1, startIndex, buffer,
0, ChunkSize)

       memory.Write(buffer, 0, CInt(retrievedBytes))

       startIndex += retrievedBytes

       If (retrievedBytes <> ChunkSize) Then
           Exit While
           End If
       End While

       connection.Close()
       Dim data() As Byte = memory.ToArray()
       memory.Dispose()
       Return data
   End Function
```

شرح الكود

بعد فتح الوصلة Connection إلى قاعدة البيانات وكتابة الأمر SQL نقوم بتنفيذه عن طريق الدالة ExecuteReader وهذه الدالة تقوم بتنفيذ الأوامر وإرجاع عنصر SqlDataReader يتم عن طريقه استرجاع البيانات كما قلنا Chunk by Chunk او دفعة بدفعة. لاحظ أنه يجب تحديد الأمر SequentialAccess.CommandBehavior للدالة ExecuteReader وهذا الأمر يسمح لنا بعمل Sequential Access أو وصول تتابعي للبيانات أي أننا لا نأخذها كلها مرة واحدة بل دفعة بدفعة بشكل تتابعي.

قمنا بعد ذلك بالنداء على دالة Read الخاصة بالعنصر SqlDataReader وذلك لنجعلها تقرأ أول صف في الصفوف وهكذا كل نداء على هذه الدالة يجعلها تقرأ صف بصف وهكذا. تقوم هذه الدالة بإرجاع True إذا كان هناك صف موجود أو False إذا كان آخر صف قرأناه هو آخر صف في البينات المسترجعة.

ثم نقوم بعد ذلك باسترجاع البينات دفعة بدفعة عن طريق الدالة GetBytes الخاصة بالعنصر SqlDataReader. وهذه الدالة تأخذ خمس مدخلات Arguments وهي على الترتيب:

1. رقم العمود. وفي هذا المثال الرقم هو 1 أي العمود الثاني.
2. المكان الذي سوف يتم القراءة من بدايته. وهو المكان داخل البيانات Binary المخزنة وهو بالبايت. فعلى سبيل المثال إذا قمنا بكتابة المكان 1024 معنى هذا أننا سوف نقوم بالقراءة من أول الكيلو بايت الثاني وهكذا. بالطبع قمنا باستخدام العنصر startIndex لمعرفة مكان القراءة لأننا نقرأ كل مرة دفعة بدفعة.
3. العنصر الذي سوف يتم تسجيل البينات المدخلة فيه. قمنا باستخدام مصفوفة من النوع Byte حجمها حجم الدفعة Chunk التي نقوم باسترجاعها في المرة.
4. حجم البيانات أي الدفعة لاسترجاعها. في هذا المثال قمنا بتحديد 256 بايت كحجم للدفعة Chunk.

تقوم هذه الدالة بإرجاع عدد البايتات التي تم استرجاعها ونقوم بتسجيل هذا العدد في المتغير vedBytesretrie ويساعدنا هذا العدد في معرفة إذا كنا في آخر البيانات أم لا لأن إذا كان العدد المسترجع غير العدد المطلوب (وهو 256 بايت كما حددنا) إذا فهذه آخر بيانات في الملف.

قمنا أيضا باستخدام عنصر من نوع MemoryStream لتسجيل البيانات المسترجعة فيه وهذا العنصر يقوم بتسجيل البيانات في الذاكرة وبالطبع هذا ليس حلا مثاليا لأنه ربما تكون هناك بيانات كبيرة الحجم جدا فلذلك يفضل التسجيل على ملف على الجهاز أفضل ولكن لنجعل المثال أبسط قمنا باستخدام MemoryStream.

أخيرا قمنا باسترجاع البيانات من الذاكرة عن طريق الدالة ToArray().

*بالطبع جميع العناصر التي لها الدالة Close أو Dispose يجب النداء على هذه الدوال متى تنتهي من استخدامها وذلك لأن هذه العناصر تستخدم موارد كبيرة للجهاز فلذلك يجب إلغاء هذه الموارد متى يتم الانتهاء من التعامل معها. تمسى هذه العناصر Disposable Objects وهي تقوم بتطبيق الـ Interface المسماة بـ*

*IDisposable. من هذه العناصر معظم (ربما جميع) العناصر المستخدمة مع قواعد البيانات وأيضا معظم (ربما جميع أيضا) العناصر المستخدمة مع الملفات.*

## المثال

مع هذا الدرس مثال يوضح كيفية تسجيل الملفات واسترجاعها من قاعدة البيانات. هذا المثال بالـ C# ويعمل على قاعدة بيانات تصميمها كالتالي:

File

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| Filename | nvarchar(260) | ☐ |
| Size | bigint | ☐ |
| Data | image | ☐ |
| | | ☐ |

لإنشاء قاعدة البيانات إما أن تقوم بإنشائها يدويا أو باستخدام ملف الأوامر الموجود مع المثال.

## خاتمة

تعلمنا في هذا المثال كيفية تسجيل ملفات في قاعدة بيانات SQL Server. كما تعرف أن حجم قاعدة البيانات سوف يزيد بحجم البيانات المسجلة فيها فلهذا يفضل أن تقوم بضغط الملفات قبل تسجيلها وأيضا يفضل ألا تقوم بتسجيل ملفات إلا إذا كانت ذا أهمية كبيرة. ويفضل أيضا وضح حد أقصى لحجم الملفات المسجلة. كما يفضل تسجيل الملفات في جدول خاص بها وألا تسجل في نفس الجدول الذي يحوي البيانات المتعلقة بها. فعلى سبيل المثال، إذا كنا نقوم ببرمجة برنامج للموارد البشرية HR لشركة يفضل ألا يتم تسجيل بيانات الشخص المتقدم للوظيفة مع السيرة الذاتية CV الخاصة به. بل يفضل وضع السير الذاتية في جدول والبيانات بجدول آخر وربط الجدولين ببعضهم وذلك حفاظا على أنه إذا كنا نريد مثلا تفحص البيانات وليست السير الذاتية لأشخاص معينين نضمن بهذا سرعة استرجاع البيانات وذلك لأن بيانات الملفات تأخذ مساحة ووقت أكبر. وهكذا...

**قم بتحميل المثال**

http://www.4shared.com/file/93798138/8ec84823/FileStore_Sample.html