بسم الله الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

آموزش مجتمع سازی CLR در SQL Server 2012 قسمت دوم

مدرس : مهندس افشین رفوآ

رمز فایل :tahlildadeh.com

حل کردن مشکلات رایج درDatabase Programming

بخش قبلی، برنامه نویسی بر پایه CLR را در سطحی بالاتر که راجع به T-SQL، extended stored procedures XPها(، و کدی در middle-tier است، قرار داد. در این بخش، ما نگاهی به دسته ای ازtask ها و الگوهایی (pattern) که یک database application developer ممکن است با آن مواجه شود، خواهم انداخت، و چگونگی استفاده (یا عدم استفاده) از CLR integrationجهت حل کردنشان را مورد بحث قرار خواهیم داد. ما چندین code example در اختیار قرار می دهیم، هم در C# و هم در.Visual Basic .NET

اعتبار دهی داده ها با استفاده از.NET Framework Library

CLR integrationدر SQL Server 2005 به کاربران اجازه می دهد از عملکرد غنی (rich functionality) که توسطclass library های .NET Framework فراهم شده، برای حل مشکلات database programmingشان استفاده کنند.

یک مثال، استفاده ازexpression های مرتب است، جهت فراهم کردن الگوهای پیچیده تر نسبت به آنچه که با استفاده از اپراتور LIKE در T-SQL دردسترس است. کد زیر را که چیزی جز یک wrapper ساده اطراف کلاس RegExدر System.Text.RegularExpressions namespaceنیست، ملاحظه فرمایید.

در:Visual Basic .NET

```
Imports System.Data.SqlTypes
Imports Microsoft.SqlServer.Server
Imports System.Text.RegularExpressions

Partial Public Class Validation

    True, IsPrecise:=True)> _
    Public Shared Function RegExMatch(ByVal pattern As String, _
        ByVal matchString As String) As Boolean

    Dim r1 As Regex = New Regex(pattern.TrimEnd(Nothing))
        Return r1.Match(matchString.TrimEnd(Nothing)).Success
    End Function

    True, IsPrecise:=True)> _
    Public Shared Function ExtractAreaCode(ByVal matchString As String)_
        As SqlString

    Dim r1 As Regex = New Regex("\((?[1-9][0-9][0-9])\)")
    Dim m As Match = r1.Match(matchString)

        If m.Success Then
            Return m.Value.Substring(1, 3)
        Else
            Return SqlString.Null
        End If

    End Function
End Class
```

در:C#

```
using System.Data.SqlTypes;
using Microsoft.SqlServer.Server;
using System.Text.RegularExpressions;

public partial class Validation
{
    [SqlFunction(IsDeterministic = true, IsPrecise = true)]
    public static bool RegExMatch(string pattern, string matchString)
    {
        Regex r1 = new Regex(pattern.TrimEnd(null));
        return r1.Match(matchString.TrimEnd(null)).Success;
    }

    [SqlFunction(IsDeterministic = true, IsPrecise = true)]
    public static SqlString ExtractAreaCode(string matchString)
    {

        Regex r1 = new Regex("\\((?[1-9][0-9][0-9])\\)");
        Match m = r1.Match(matchString);
```

```
        if (m.Success)
            return m.Value.Substring(1, 3);
        else return SqlString.Null;
    }
}
```

بیایید فرض کنیم متدهای RegExMatch() و ExtractAreaCode() ، به عنوان تابع های (function) تعریف شده توسط کاربر، در database با آپشن RETURNS NULL ON NULL INPUT ثبت شده اند، به طوری که function ، هنگامی که هر یک از ورودی هایش (input) بی اثر هستند، بی اثر باز می گردد. این، به ما اجازه می دهد از نوشتن هر special null handling code درون function بپرهیزیم.

حالا می توانیم constraint هایی روی ستونهای جدولی که از کد بالا برای اعتبار دهی آدرسهایی ایمیل و شماره تلفنها استفاده می کند، به ترتیب زیر تعریف کنیم:

```
CREATE TABLE contacts
(
    firstName nvarchar(30),
    lastName  nvarchar(30),
    emailAddress nvarchar(30) CHECK
    (dbo.RegExMatch('[a-zA-Z0-9_\-]+@([a-zA-Z0-9_\-]+\.)+(com|org|edu)',
        emailAddress) = 1),
    usPhoneNo nvarchar(30) CHECK
    (dbo.RegExMatch(
        '\([1-9][0-9][0-9]\) [0-9][0-9][0-9]\-[0-9][0-9][0-9][0-9]',
        usPhoneNo)=1),
    areaCode AS dbo.ExtractAreaCode(UsPhoneNo) PERSISTED
)
```

به یاد داشته باشید که ستون areaCode ، یک ستون دایم و محاسبه شده است، که area code را از ستون usPhoneNo با استفاده از تابع ExtractAreaCode() بیرون می کشد. ستون areaCode ممکن است index شود، و query ها را در جدولی که توسط area code دنبال مخاطبین می گردد، تسهیل می کند.

بطور کلی، این مثال نشان می دهد چگونه می توان از library های .NET Framework برای افزایش T-SQL built-in function library با تابع های مفید که در T-SQL به سختی بیان می شوند، استفاده کرد.

بدست آوردن خروجی

نیاز به بدست آوردن خروجی از یک database object) از قبیل یک stored procedure یا یک (view که درون سرور اجرا می شود، یکی از رایج ترین task های database programming است. اگر بتوان خروجی را با استفاده از یک query واحد ایجاد کرد، آنگاه می توان این کار را تنها با استفاده از یک view یا یک inline table-valued function انجام داد. اما، اگر برای ایجاد خروجی، نیاز به عبارات چند گانه و procedural logic باشد، آنگاه دو آپشن وجود دارد stored procedure: ها و table-valued function ها. در حالیکه SQL Server

2000دارایtable-valued function ها است، اما می توان آنها را فقط در T-SQL نوشت. با CLR integrationدرSQL Server 2005 ، می توان اینfunction ها را با استفاده از یک زبان مدیریت شده (managed language)نیز نوشت. در این بخش، ما نگاهی به چگونگی نوشتنstored procedure ها و table-valued functionها با استفاده از CLR می اندازیم.

برگرداندن خروجی نسبی (relational results) ازT-SQL ، چه به عنوان return value یک table-valued function، چه از طریق "caller's pipe" مستقیم و همیشه حاضر، درون یک stored procedure ممکن است: از هرجایی در یک– stored procedure بدون در نظر گرفتن nesting در –execution level یک عبارت SELECTاجرا شده، خروجی را به caller باز می گرداند.

اگر دقیق تر بگوییم، این درست است که عبارتاتSELECT ، مقدار دهی به متغیررا اجرا نمی کنند. عبارات FETCH، READTEXT، PRINT، وRAISERROR ، نیز خروجی را مستقیماً به caller بازمیگردانند.

به یاد داشته باشید که " caller" بدرستی تعریف نشده است؛ بستگی به فراخوانی بافت (invocation context)در stored procedure دارد.

اگر یک stored procedure از هر(client data access API ا زقبیلODBC ، OLEDB، یا (SQLClient فراخوانده شود، caller، APIواقعی و هر) abstraction مثلاً، hstmt، IRowset، یا (SqlDataReader که برای ارائه خروجی مهیا می کند، است. بطور کلی، این بدین معناست که خروجی بدست آمده از درون یکstored procedure، به API فراخواننده بازگردانده خواهد شد، و همهframe های T-SQL روی stack را کنار می گذارد، مانند مثال زیر:

```
CREATE PROC proc1 AS
   SELECT col1 FROM dbo.table1;
CREATE PROC proc2 AS
   EXEC proc1;
```

هنگام اجرای رویهproc2 ، خروجی بدست آمده ازproc1 ، به caller of proc2 می روند. فقط یک راه هست که در آن proc2 می تواند خروجی بدست آمده را capture کند: از طریق stream کردن خروجی بهdisk ، با استفاده از INSERT یاEXEC ، به یک جدول دایم یا موقت، یا یک جدول متغیر.

```
CREATE PROC proc2 AS
   DECLARE @t TABLE(col1 INT);
   INSERT @t (col1) EXEC proc1;
   -- do something with results
```

Stored procedureهایSQL Server 2005 CLR ، نوع جدیدی از caller را معرفی می کنند. وقتی یک queryبا استفاده از ADO.NET provider از درون یک frame مدیریت شده اجرا می شود، خروجی از طریق شی SqlDataReader قابل دسترسی می شوند و می توانند درون stored procedure مصرف شوند.

در:Visual Basic .NET

```
...
Using conn As New SqlConnection("context connection = true")
    conn.Open()
Dim cmd As SqlCommand = New SqlCommand( _
    "SELECT col1 FROM dbo.table1", conn)
Dim reader As SqlDataReader = cmd.ExecuteReader()

    Do While reader.Read()
' Do something with current row
    Loop
End Using
```

در:C#...

```
...
using (SqlConnection conn= new SqlConnection("contect connection = true"))
{
    ...
    SqlCommand cmd = new SqlCommand(
        "SELECT col1 FROM dbo.table1", conn);
    SqlDataReader reader = cmd.ExecuteReader();
    while (reader.Read())
    {
        // do something with current row
    }
    ...
}
```

...مسئله ای که باقی می ماند، این است که یک managed routine چگونه خروجیش را به caller باز می گرداند. این کار در stored procedure ها و table-valued function ها به طور متفاوتی انجام می شود. یک stored procedure از یک مثال static که از کلاس SqlContext قابل دسترسی است، جهت بازگرداندن data استفاده می کند، در حالیکه یک table-valued function ، یک interface را پیاده سازی می کند که به SQL Server اجازه بازیابی خروجی را می دهد. هر دوی اینها در قسمت بعدی مورد بحث قرار می گیرند.

Stored Procedure های CLR و SqlPipe

در میان متدهای دردسترس در کلاس SqlPipe ، ساده ترینشان، ExecuteAndSend() است که یک command object را یک پارامترورودی در نظر می گیرد. این متد، command را اجرا می کند، اما به جای اینکه خروجی برای managed frame قابل دسترس کند، خروجی را به فراخواننده stored procedure ارسال می کند. این، معادل قرار دادن یک عبارت درون یک T-SQL stored procedure است، و در حالیکه ناخوشایندتر است، همراه با معادل T-SQL ، روی par است.

یک stored procedure ساده برای اجرای یک SELECT در T-SQL:

```
CREATE PROC proc1 AS
```

```
   SELECT col1 FROM dbo.table1;
```

معادلش در:C#

```
...
[Microsoft.SqlServer.Server.SqlProcedure]
public static void proc1()
{
    using (SqlConnection conn =
        new SqlConnection("context connection = true"))
    {
        conn.Open();
        SqlCommand cmd = new SqlCommand(
            "SELECT col1 FROM dbo.table1", conn);
        SqlContext.Pipe.ExecuteAndSend(cmd);
        conn.Close();
    }
}

...
```

و در:Visual Basic .NET

```
...
_
Public Shared Sub VBproc1()
    Using conn As New SqlConnection("context connection=true")
        conn.Open()
        Dim cmd As SqlCommand = New SqlCommand( _
            "SELECT col1 FROM dbo.table1", conn)
        SqlContext.Pipe.ExecuteAndSend(cmd)
        conn.Close()
    End Using
End Sub

...
```

SqlPipe.ExecuteAndSend()، در سناریوهایی که داده های در شرف بازگشت، مستقیماً توسط یک query اجرا شده بدست می آیند، خوب جواب می دهد. اما، ممکن است مواردی وجود داشته باشد که در آنها، تغییر (manipulation) داده ها قبل از ارسال آنها، یا ارسال داده هایی که از source های خارج از local SQL Server instance بدست آمده، مطلوب باشد.

SqlPipe، گروهی از متدهایی را که با هم کار می کنند تا application ها را قادر به باز گرداندن خروجی اختیاری به caller ، (SendResultsStart()، SendResultsRow()، و SendResultsEnd()) کنند، در اختیار می گذارد. در اکثر موارد، این API ها، شبیه API های srv_describe و srv_sendrow هستند که برای extended stored procedure ها دردسترس هستند.

SendResultsStart()، یک SqlDataRecord را یک پارامتر ارسالی (argument) در نظر می گیرد و شروع یک نتیجه جدید را نشان می دهد. این API ، اطلاعات metadata را از record object می خواند و آن را به

caller ارسال می کند. متعاقباً، می توان Row ها را با فراخواندن SendResultsRow() ، یکبار برای هر row که در شرف ارسال است، باز گرداند. بعد از اینکه همه row های دلخواه ارسال شدند، فراخوانی SendResultsEnd() جهت نشان دادن پایان خروجی، لازم است.

مثال: بازگرداندن یک RSS Feed در یک CLR Stored Procedure

C# code fragment زیر، بخشی از یک stored procedure نشان می دهد که یک XML document را از Web می خواند- یک Really Simple Syndication (RSS) feed از MSDN- و کلاسهای System.Xml برای parse کردن آن استفاده می کند، و اطلاعات را به شکل نسبی باز می گرداند. به یاد داشته باشید که کد باید در یک EXTERNAL_ACCESS یا UNSAFE assembly نمایش داده شود، زیرا permission های Code Access Security (CAS) که برای دسترسی به اینترنت ضروری هستند، فقط در این دسته از permission ها قابل دسترس هستند.

```
...
using (SqlConnection conn =
    new SqlConnection("context connection = true"))
{
    // Retrieve the RSS feed
    XPathDocument doc = new
        PathDocument("http://msdn.microsoft.com/sql/rss.xml");
    XPathNavigator nav = doc.CreateNavigator();
    XPathNodeIterator i = nav.Select("//item");

    // create metadata for four columns
    // three of them are string types and one of the is a datetime
    SqlMetaData[] rss_results = new SqlMetaData[4];
    rss_results[0] = new SqlMetaData("Title", SqlDbType.NVarChar, 250);
    rss_results[1] = new SqlMetaData("Publication Date",
        SqlDbType.DateTime);
    rss_results[2] = new SqlMetaData("Description",
        SqlDbType.NVarChar, 2000);
    rss_results[3] = new SqlMetaData("Link", SqlDbType.NVarChar, 1000);

    // construct the record which holds metadata and data buffers
    SqlDataRecord record = new SqlDataRecord(rss_results);

    // cache a SqlPipe instance to avoid repeated calls to
    // SqlContext.GetPipe()
    SqlPipe sqlpipe = SqlContext.Pipe;

    // send the metadata, do not send the values in the data record
    sqlpipe.SendResultsStart(record);

    // for each xml node returned, extract four pieces
    // of information and send back each item as a row
    while (i.MoveNext())
    {
```

```
        record.SetString(0, (string)

        i.Current.Evaluate("string(title[1]/text())"));
            record.SetDateTime(1, DateTime.Parse((string)

        i.Current.Evaluate("string(pubDate[1]/text())")));
            record.SetString(2, (string)

        i.Current.Evaluate("string(description[1]/text())"));
            record.SetString(3, (string)

        i.Current.Evaluate("string(link[1]/text())"));
        sqlpipe.SendResultsRow(record);
    }

    // signal end of results
    sqlpipe.SendResultsEnd();
}
```

...

به خاطر داشته باشید که بین call ها به SendResultsStart() و SendResultsEnd() ، عبارت SqlPipe در یک حالت busy تنظیم می شود، وفراخوانی هر متد Send ، غیر از SendResultsRow() ، منجر به یک error خواهد شد SendingResults property . ، در حالیکه SqlPipe در حالت busy است، true تنظیم می شود.

Functions های Table-Valued

CLR integration، از function های table-valued (TVF) که در managed language ها نوشته شده اند، نیز ساپورت می کند. مانند معادل T-SQL ، CLR TVF ها نیز ابتداً برای بازگرداندن خروجی جدولی استفاده می شدند. واضح ترین تفاوت این است که T-SQL table-valued function ، خروجی را موقتاً در یک جدول ذخیره می کنند، در حالیکه CLR TVF ، قابلیت stream کردن خروجی بدست آمده را دارد، بدین معنی که این خروجی نباید قبل از بازگشت از function ، materialize شوند.

مثال: یک Table-Valued Function برای بازیابی یک RSS Feed

در اینجا، ما RSS retrieval recast را به عنوان یک table-valued function در C# نشان می دهیم. به تطابق بین SqlFunction annotation در متد RSS_TVF() و امضاء (signature) متد FillTVFRow() ، دقت کنید.

```
using System;
using System.Data;
using System.Data.Sql;
using System.Data.SqlTypes;
using Microsoft.SqlServer.Server;
using System.Xml.XPath;
using System.Collections;
```

```
public partial class UserDefinedFunctions
{
    [SqlFunction(FillRowMethodName = "FillTVFRow",
        TableDefinition = "Title nvarchar(250), " +
            "PublicationDate datetime, " +
            "Description nvarchar(2000), " +
            "Link nvarchar(1000)")
    ]
    public static IEnumerable RSS_TVF()
    {
        return new RssReader();
    }

    public static void FillTVFRow(object row, out SqlString str,
        out SqlDateTime date, out SqlString desc, out SqlString link)
    {
        // split each object array
        object[] rowarr = (object[])row;
        str = (SqlString)(rowarr[0]);
        date = (SqlDateTime)(rowarr[1]);
        desc = (SqlString)(rowarr[2]);
        link = (SqlString)(rowarr[3]);
    }
}

public class RssReader : IEnumerable
{
    XPathDocument doc;
    XPathNavigator nav;

    // Construct helper class, initializing metadata for the results
    // reading from the RSS feed, creating the iterator
    public RssReader(string site)
    {
        // Retrieve the RSS feed
        //doc = new XPathDocument("http://msdn.microsoft.com/sql/rss.xml");
        doc = new XPathDocument(site);
        nav = doc.CreateNavigator();
    }

    public IEnumerator GetEnumerator()
    {
        return new RSSEnumerator(this);
    }

    private class RSSEnumerator : IEnumerator
    {
        XPathNodeIterator i;
        Object[] current;
        RssReader reader;

        public RSSEnumerator(RssReader reader)
        {
            this.reader = reader;
            Reset();
```

```
        }

        public void Reset()
        {
            i = reader.nav.Select("//item");
        }

        public bool MoveNext()
        {
            if (i.MoveNext())
            {
                current = new Object[4];
                current[0] = new SqlString((string)
                    i.Current.Evaluate("string(title[1]/text())"));
                current[1] = new SqlDateTime(DateTime.Parse((string)
                    i.Current.Evaluate("string(pubDate[1]/text())")));
                current[2] = new SqlString((string)
                    i.Current.Evaluate("string(description[1]/text())"));
                current[3] = new SqlString((string)
                i.Current.Evaluate("string(link[1]/text())"));
                return true;
            }
            else return false;

        }

        public Object Current
        {
            get
            {
                return current;
            }
        }
    }
}
```

یک query ساده برای consume کردن خروجی از این جدول، اینگونه خواهد بود:

```
SELECT *
FROM RSS_TVF()
```

طبیعتاً، queryهای قویتر را می توان روی شکل TVF این داده، بیان کرد. با فرض اینکه ما یک تابع CanonicalURL()داریم که نسخه کانونیك یک URL را باز می گرداند، داده هایی از یک RSS feed را می توان با استفاده ازURL های کانونیک به آسانی بازگرداند:

```
select title, publicationDate, description, dbo.CanonicalURL(link)
from dbo.RSS_TVF()
order by publicationDate
```

توجه داشته باشید که در مثال بالا، ما از قابلیتهای stream کردن TVF استفاده نمی کنیم، زیرا ما کل RSS feedرا consume می کنیم، یک navigator در بالا درست می کنیم، و سپس آیتمهای مجزا را هنگامی که MoveNext()فراخوانده می شود، تکرار می کند. اما، consumeکردن خروجی از یک Web source و تکرار XMLتولید شده با یکXmlReader ، با استفاده از streaming API ممکن است. قابل ذکر است که با فرض وجود تفاوت مدل اجرا بین table-valued functions در CLR وT-SQ ، تفاوتی بزرگ ممکن است مشاهده شود، مخصوصاً در سناریوهایی که در آنها stream کردن خروجی ممکن است.

مثال crack :کردن Scalarها در Rowها

معمولاً در برنامه نیاز داریم که چندین پارامتر ورودی را به صورت همزمان ارسال کنیم. مثلاً، امکان دارد یک order processing systemنیاز به یک stored procedure داشته باشد که یک order را در جدولorder ها قرار می دهد. امکان دارد یک پارامتر ورودی مطلوب یکstored procedure ، درorder ، line-itemsباشد؛ اما، این موضوع با این محدودیت روبرو می شود که T-SQL از پارامترهای چندگانه ساپورت نمی کند، و فاقد collectionها وarray ها است.

یک راه حل، encodeکردن collection به عنوان یک scalar value است – مثلاً به عنوان یک nvarchar یا xml-و ارسال آن به عنوان یک پارامتر ورودی به stored procedure. Stored procedure می تواند از-table valued functionکه scalar input را می گیرد و آن را تبدیل به دسته ای ازrow ها می کند، استفاده می کند، و سپس می تواند در یک جدول line-items یا تغییر یافته (manipulated) قرار گیرد.

در حالیکه table-valued function را می توان در T-SQL نوشت، اما اگر در CLR نوشته شود، performanceبهتری خواهد داشت. زیرا می تواند ازstring manipulation function ها درSystem.Text namespaceاستفاده بهینه ببرد، همچنین انجامش هم بسیار ساده است.

مثال زیر، پیاده سازی یک table-valued function را نشان می دهد که یک input string را که توسط یک نقطه ویرگول جدا شده را می گیرد و قطعه ها را به شکل دسته ای ازrow ها باز می گرداند.

در:Visual Basic .NET

```
Imports System.Collections
Imports System.Data.SqlTypes
Imports Microsoft.SqlServer.Server

Partial Public Class UserDefinedFunctions

    ' This needs to return an IEnumerable, but since an array
    ' does, in this case we do not need to define a new class
    ' that implements it: we can simply return the array.
    "FillRow", _
        TableDefinition:="value nvarchar(60)")> _
    Public Shared Function GetStrings(ByVal str As SqlString) _
        As IEnumerable
```

```
        Return str.Value.Split(";"c)
    End Function

    ' This method does the decoding of the row object. Since the
    ' row is already a string, this method is trivial. Note that
    ' this method is pointed to by the annotation on the
    ' GetString method.
    Public Shared Sub FillRow(ByVal row As Object, _
        ByRef str As String)

        str = CType(row, String)
    End Sub

End Class
```

در:C#

```
using System.Collections;
using System.Data.SqlTypes;
using Microsoft.SqlServer.Server;

public partial class UserDefinedFunctions
{
    /*
     * This needs to return an IEnumerable, but since an array
     * does, in this case we do not need to define a new class
     * that implements it: we can simply return the array.
     */
    [SqlFunction(FillRowMethodName = "FillRow",
        TableDefinition = "value nvarchar(60)")]
    public static IEnumerable GetStrings(SqlString str)
    {
        return str.Value.Split(';');
    }

    /*
     * This method does the decoding of the row object. Since the
     * row is already a string, this method is trivial. Note that
     * this method is pointed to by the annotation on the
     * GetString method.
     */
    public static void FillRow(object row, out string str)
    {
        str = (string)row;
    }
}
```