عنوان: **کاهش حجم لاگ فایل‌های اس‌کیوال سرور 2005 و 2008**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۸/۰۷ ۰۱:۲۴:۱۱
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

نمی‌دونم تابحال به صورت جدی با SharePoint مایکروسافت کار کردید یا نه؟ این برنامه که عمده کارهای خودش رو با SQL server انجام میده در طول یک روز ممکن است تا 80 گیگ log file اس‌کیوال سرور تولید کند و بعد از چند روز اگر به همین صورت به حال خود رها شود (که عموما هم به همین صورت است!) ممکن است دیگر قادر به استفاده از سرور به دلیل پر شدن درایوی که لاگ فایل‌ها در آن ذخیره می‌شوند نباشید.

همچنین رشد tempdb نیز توسط این برنامه بسیار چشم‌گیر است. بنابراین همیشه به‌خاطر داشته باشید محل قرارگیری tempdb و همچنین محل قرارگیری لاگ فایل‌ها (که هر دو قابل تنظیم هستند) را در درایوهایی قرار دهید که حداقل 100 گیگ فضای خالی در آنها موجود باشد.

با استفاده از اسکریپت زیر می‌شود حجم لاگ فایل‌های اس کیوال سرور را به حداقل رساند و نفس راحتی کشید! این مساله اگر جدی گرفته نشود واقعا تبدیل به یک کابوس می‌شود!

اسکریپت زیر کلیه دیتابیس‌های موجود را یافته و shrink می‌کند. قسمت offline و online کردن آن هم به این خاطر است که ارتباط تمام کاربران متصل را به صورت آنی قطع می‌کند (یکی از چندین روش موجود برای kill کردن کاربران است). (یک stored procedure از آن درست کنید و با تعریف یک job جدید در اس کیوال سرور ، این stored procedure را برای مثال هر روز ساعت 3 بامداد به صورت خودکار اجرا کنید)

```
Declare @database nvarchar(1000)
Declare @tsql nvarchar(4000)
Declare DatabaseCursor Cursor
Local
Static
For
select name from master.dbo.sysdatabases
open DatabaseCursor
fetch next from DatabaseCursor into @database

while @@fetch_status = 0
begin
 print 'database:' + @database
 if @database not in ('tempdb','master','model','msdb')
 begin
  SET @tsql = 'use master;
     alter database ['+@database+'] set offline with rollback immediate;
     alter database ['+@database+'] set online;
   DECLARE @dbLogName nvarchar(500) ;
   Use  ['+@database+']  ;
   select @dbLogName = rtrim(ltrim(name)) from sysfiles WHERE FILEID=2;
   ALTER DATABASE ['+@database+'] SET SINGLE_USER ;
   DBCC SHRINKFILE(@dbLogName , 2) ;
   BACKUP LOG  ['+@database+']  WITH TRUNCATE_ONLY ;
   DBCC SHRINKFILE(@dbLogName , 2) ;
   ALTER DATABASE ['+@database+']  SET MULTI_USER ;'
   exec(@tsql)
 end
 fetch next from DatabaseCursor into @database
end

close DatabaseCursor

deallocate DatabaseCursor
```

اسکریپت فوق با SQL Server 2005 سازگار است اما در SQL Server 2008 منسوخ شده است! (قسمت truncate کردن)
نسخه سازگار با SQL server 2008 آن به صورت زیر است:

```
Declare @database nvarchar(1000)
Declare @tsql nvarchar(4000)
Declare DatabaseCursor Cursor
Local
```

```
Static
For
select name from master.dbo.sysdatabases
open DatabaseCursor
fetch next from DatabaseCursor into @database

while @@fetch_status = 0
begin
 print 'database:' + @database
 if @database not in ('tempdb','master','model','msdb')
 begin
  SET @tsql = 'use master;
     alter database ['+@database+'] set offline with rollback immediate;
     alter database ['+@database+'] set online;
   DECLARE @dbLogName nvarchar(500) ;
   Use  ['+@database+']  ;
   select @dbLogName = rtrim(ltrim(name)) from sysfiles WHERE FILEID=2;
   ALTER DATABASE ['+@database+'] SET RECOVERY SIMPLE;
   ALTER DATABASE ['+@database+'] SET SINGLE_USER ;
   DBCC SHRINKFILE(@dbLogName , 2) ;
   ALTER DATABASE ['+@database+'] SET MULTI_USER ;
   ALTER DATABASE ['+@database+'] SET RECOVERY FULL;'
   exec(@tsql)
 end
 fetch next from DatabaseCursor into @database
end

close DatabaseCursor

deallocate DatabaseCursor
```

ماخذ اصلی مورد استفاده:

http://go.microsoft.com/fwlink/?LinkId=111531&clcid=0x409

نظرات خوانندگان

نویسنده: Salar
تاریخ: ۱۳۸۷/۰۸/۰۶ ۲۲:۴۶:۰۰

جناب نصیری، با اجازه به شما لینک دادم

نویسنده: میثم
تاریخ: ۱۳۸۸/۰۷/۰۴ ۱۷:۰۴:۴۶

خب چرا تو Propeties دیتابیس shrink رو Auto نمیکنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۰۴ ۱۷:۲۰:۲۸

زمانیکه صحبت از دیتابیس شیرپوینت می‌شود یعنی دیتابیس بالای 50 گیگ.
و به 2 دلیل در اینجا نباید از auto shrink استفاده کرد:
- کاهش حجم قابل ملاحظه ذکر شده توسط auto shrink رخ نمی‌دهد.
- auto shrink یکی از مواردی است که به شدت روی کارآیی دیتابیسی حجیم تاثیر منفی دارد و همچنین سبب ایجاد fragmentation می‌شود و در اکثر مقالات استفاده از آن به شدت نهی شده‌است.

عنوان: **گزارشگیری از تاریخچه‌ی پشتیبان‌گیری‌ها در اس کیوال سرور**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۸/۱۵ ۰۹:۰۷:۴۶
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

آیا می‌دانید آخرین باری که از یک دیتابیس مفروض بک‌آپ گرفته شده، چه زمانی بوده است؟ (یا اینکه اصلا چه اهمیتی داره؟!)

```
USE master;
SELECT B.name AS Database_Name,
   ISNULL(STR(ABS(DATEDIFF(day, GetDate(),
   MAX(Backup_finish_date)))), 'NEVER') AS DaysSinceLastBackup,
   ISNULL(CONVERT(char(10), MAX(backup_finish_date), 101), 'NEVER') AS
   LastBackupDate
FROM   master.dbo.sysdatabases B
LEFT OUTER JOIN   msdb.dbo.backupset A
ON   A.database_name = B.name
AND   A.type = 'D'
GROUP BY
   B.Name
ORDER BY
   B.name
```

Results | Messages

	Database_Name	DaysSinceLastBackup	LastBackupDate
22	Northwind	NEVER	NEVER
23	outtime	5	10/31/2008

همانطور که مطلع هستید در SQL Server 2008 امکان فشرده سازی خودکار بک آپ‌ها هم فراهم شده است. با استفاده از اسکریپت زیر می‌توان از درصد فشرده سازی بک‌آپ‌ها گزارش گرفت:

```
SELECT bs.server_name,
   bs.database_name AS 'Database Name',
   CONVERT (BIGINT, bs.backup_size / 1048576 ) AS
   'Uncompressed Backup Size (MB)',
   CONVERT (BIGINT, bs.compressed_backup_size / 1048576 ) AS
   'Compressed Backup Size (MB)',
   CONVERT (NUMERIC (20,2), (CONVERT (FLOAT, bs.backup_size) /
   CONVERT (FLOAT, bs.compressed_backup_size))) AS 'Compression Ratio',
   DATEDIFF (SECOND, bs.backup_start_date, bs.backup_finish_date) AS
   'Backup Elapsed Time (sec)',
   bs.backup_finish_date AS 'Backup Finish Date'
FROM   msdb.dbo.backupset AS bs
WHERE  DATEDIFF (SECOND, bs.backup_start_date, bs.backup_finish_date) > 0
AND  bs.backup_size > 0
AND  bs.type = 'D' -- Change to L if you want Log backups
ORDER BY
   bs.backup_finish_date DESC
```

Results | Messages

	server_name	Database Name	Uncompressed Backup Size (MB)	Compressed Backup Size (MB)	Compression Ratio	Backup Elapsed Time (sec)	Backup Finish Date
19	VAHID	RSSReader	42169	9871	4.27	929	2008-10-24 23:15:33.000

برای فعال سازی گزینه فشرده سازی بک‌آپ‌ها با استفاده از دستورات T-SQL می‌توان به صورت زیر عمل کرد (در SQL server 2008):

```
USE master;
EXEC sp_configure 'backup compression default', '1';
RECONFIGURE WITH OVERRIDE;
```

ماخذ مورد استفاده:

http://www.sqlservercentral.com

عنوان: **گزارشگیری از تاریخچه‌ی اجرای کوئری‌ها در SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۸/۱۸ ۰۹:۴۹:۲۸
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

چند روز قبل مشکلی رخ داده بود به این شرح!
سروری که SQL server بر روی آن نصب بود بخاطر SQL server ، بیش از 50 درصد CPU usage مداوم پیدا کرده بود. عموما مصرف CPU اس کیوال سرور روی سرورهای قوی بالا نیست و تداوم این حالت به این شدت یعنی بروز مشکل.
اینجا است که این سؤال پیش میاد، SQL Server الان داره چکار میکنه که تا این حد به صورت مداوم مصرف CPU آن بالا رفته؟
(حدودا 2 ساعت تمام به صورت مداوم مصرف CPU بالای 50 درصد بود)
با استفاده از کوئری زیر می‌شود، عبارات SQL ایی را که هم اکنون در حال اجرا هستند را به صورت زنده مشاهده کرد: (در اس کیوال سرور 2005 به بعد)

```
USE master;
SELECT   st.text, r.session_id, r.status, r.command, r.cpu_time, r.total_elapsed_time
FROM sys.dm_exec_requests r
CROSS APPLY sys.dm_exec_sql_text(sql_handle) AS st
```

مشکل از بروز یک loop بود که به این صورت دقیقا کوئری مربوطه تشخیص داده شد و برطرف شد.

همچنین با استفاده از کوئری زیر می‌توان آخرین 50 کوئری اجرا شده در SQL server را به همراه زمان اجرای آنها گزارشگیری کرد:

```
SELECT TOP 50
deqs.last_execution_time AS [Time],
      dest.text AS [Query]
FROM   sys.dm_exec_query_stats AS deqs
CROSS apply
sys.dm_exec_sql_text(deqs.sql_handle) AS dest
ORDER BY
      deqs.last_execution_time DESC
```

اگر علاقمند باشید در مورد این نوع کوئری‌ها اطلاعات بیشتری کسب کنید، مطالعه مقاله زیر توصیه می‌شود:
http://www.sqlteam.com/article/dynamic-management-views

عنوان: تشخیص کمبود ایندکس‌ها در SQL server
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۸/۲۲ ۲۲:۴۳:۳۶
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

در مطالب قبلی به اختصار در مورد dynamic management views که از SQL server 2005 به بعد ارائه شده‌اند مثال‌هایی کاربردی ارائه گشتند. یکی دیگر از قابلیت‌های فوق العاده مهم این DMV ها، پیشنهاد ایجاد ایندکس بر روی جداول است. این پیشنهادات بر اساس آمارهای جمع آوری شده توسط موتور بهینه ساز اجرای کوئری‌ها در اس کیوال سرور به شما ارائه خواهند شد. برای مثال کوئری زیر را در management studio اجر نمائید:

```
USE master;
SELECT d.database_id,
      d.object_id,
      d.index_handle,
      d.equality_columns,
      d.inequality_columns,
      d.included_columns,
      d.statement AS fully_qualified_object,
      gs.*
FROM   sys.dm_db_missing_index_groups g
JOIN   sys.dm_db_missing_index_group_stats gs
 ON   gs.group_handle = g.index_group_handle
JOIN   sys.dm_db_missing_index_details d
 ON   g.index_handle = d.index_handle
```

Results | Messages

	database_id	object_id	index_handle	equality_columns	inequality_columns	included_columns	fully_qualified_object
1	13	1237579447	5	NULL	[IsRead]	[pubDate]	[RSSReader].[dbo].[tblExtractedData]
2	13	1237579447	3	NULL	[IsRead]	[AddDate]	[RSSReader].[dbo].[tblExtractedData]
3	13	2137058649	1	[IsCanceled]	NULL	NULL	[RSSReader].[dbo].[tblBlogs]

خروجی حاصل لیستی است که بر اساس تفاسیر موتور بهینه ساز اجرای کوئری‌ها بدست آمده است. equality_columns بر اساس حالت‌هایی مانند table.column = constant_value پیش بینی شده‌است. inequality_columns بر اساس حالت‌هایی مانند table.column > constant_value و included_columns برای حالت‌هایی است که می‌خواهیم ایندکس ایجاد شده محدودیت اندازه 900 بایت را نداشته باشد، یا نوع داده‌ای مورد استفاده برای مثال nvrachar max و امثال آن باشد (text و ntext مجاز نیست) و مواردی از این دست.

fully_qualified_object هم مشخص می‌کند که این ایندکس دقیقا باید بر روی چه دیتابیس و جدولی ایجاد شود.

تذکر: این آمارهای جمع‌آوری شده پس از هر بار ری‌استارت سرور، صفر خواهند شد.

اکنون این سؤال مطرح می‌شود که چگونه از این اطلاعات استفاده کنیم؟

دقیقا بر اساس INEQUALITY_COLUMNS ، EQUALITY_COLUMNS و INCLUDED_COLUMNS گزارش فوق، می‌توان به صورت زیر عمل کرد:

```
CREATE NONCLUSTERED INDEX <unique index name>
ON <FULL_TABLE_NAME> (<EQUALITY_COLUMNS>,<INEQUALITY_COLUMNS>) -- exclude INEQUALITY_COLUMNS if NULL
INCLUDE (<INCLUDED_COLUMNS>); -- exclude INCLUDED_COLUMNS if NULL
```

خوب، پس از گزارشگیری، ممکن است لیست بلند بالایی تهیه شود. کوئری زیر عبارات create index مورد نظر را بر اساس این قابلیت جدید تولید خواهد کرد:

```
SELECT mig.index_group_handle,
      mid.index_handle,
      migs.avg_total_user_cost AS AvgTotalUserCostThatCouldbeReduced,
      migs.avg_user_impact AS AvgPercentageBenefit,
      'CREATE INDEX missing_index_' + CONVERT (varchar, mig.index_group_handle)
      + '_' + CONVERT (varchar, mid.index_handle)

      + ' ON ' + mid.statement

      + ' (' + ISNULL (mid.equality_columns,'')

      + CASE
            WHEN mid.equality_columns IS NOT NULL AND mid.inequality_columns
            IS NOT NULL THEN ','
            ELSE ''
        END

        + ISNULL (mid.inequality_columns, '')

 + ')'

 + ISNULL (' INCLUDE (' + mid.included_columns + ')', '') AS
 create_index_statement

   FROM sys.dm_db_missing_index_groups mig

INNER JOIN sys.dm_db_missing_index_group_stats migs ON migs.group_handle = mig.index_group_handle

INNER JOIN sys.dm_db_missing_index_details mid ON mig.index_handle = mid.index_handle
```

Results | Messages

	AvgPercentageBenefit	create_index_statement
1	78.51	CREATE INDEX missing_index_6_5 ON [RSSReader].[dbo].[tblExtractedData] ([IsRead]) INCLUDE ([pubDate])
2	76.67	CREATE INDEX missing_index_4_3 ON [RSSReader].[dbo].[tblExtractedData] ([IsRead]) INCLUDE ([AddDate])
3	89.32	CREATE INDEX missing_index_2_1 ON [RSSReader].[dbo].[tblBlogs] ([IsCanceled])

مزایای

ایجاد ایندکس‌های صحیح بر اساس نیازهای واقعی کاری:

سریعتر شدن اجرای کوئری‌های جستجو در تعداد رکوردهای بالا

مرتب سازی سریعتر نتایج (sorting)

کوئری‌هایی که بر اساس عبارت GROUP BY ایجاد شده‌اند، سریعتر اجرا خواهند شد

نظرات خوانندگان

نویسنده: Alex
تاریخ: ۱۳۸۷/۰۸/۲۶ ۰۸:۳۲:۰۰

ممنون خیلی مفید بود. البته بنظر بنده ی حقیر تمام پستهای شما واسه من مفید بوده و هست. امیدوارم که بقیه اش هم اینجوری باشه.
همیشه موفق و سلامت باشید.

عنوان: **پیدا کردن لیست SQL server های نصب شده در یک شبکه**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۸/۲۸ ۱۴:۰۴:۳۸
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

با آمدن SQL server 2008 استفاده از کتابخانه SQL-DMO برای انجام یک سری از امور بر روی اس کیوال سرور با استفاده از برنامه نویسی منسوخ شد. یکی از توانایی‌های این کتابخانه لیست کردن سرورهای اس کیوال (قابل دسترسی) موجود در شبکه بود.
برای مثال توسط این کتابخانه به صورت زیر می‌توان اینکار را انجام داد:
در قطعه کد زیر فرض بر این است که ارجاعی به کتابخانه sqldmo را در برگه com مربوط به project->add reference اضافه کرده‌اید:

```
using SQLDMO;
   using System.Collections.Generic;

   public static List<string> GetSQLServersList2()
   {
       List<string> result = new List<string>();
       ApplicationClass sqlApp = new ApplicationClass();
       NameList lst = sqlApp.ListAvailableSQLServers();
       for (int i = 1; i <= lst.Count; i++)
           result.Add(lst.Item(i));
       lst = null;
       sqlApp = null;

       return result;

   }
```

با منسوخ شدن این کتابخانه COM (که تنها تا اس کیوال سرور 2005 پشتیبانی می‌شود)، در نگارش‌های جدید (و قدیم) اس کیوال سرور، با استفاده از قطعه کد زیر می‌توان لیست تمام SQL server های نصب شده در یک شبکه به همراه instance های آنها را بدست آورد.

```
using System.Collections.Generic;
using System.Data;
using System.Data.Sql;

public class CListServers
{
   public static List<string> GetSQLServersList()
   {
       List<string> result = new List<string>();

       // Retrieve the enumerator instance and then the data.
       var instance = SqlDataSourceEnumerator.Instance;
       var table = instance.GetDataSources();

       // Display the contents of the table.
       foreach (DataRow row in table.Rows)
       {
           result.Add(string.Format("{0}\\{1}", row[0], row[1]));
       }

       return result;
   }
}
```

راه دیگر:
کتابخانه COM یاد شده (SQL-DMO) در SQL server 2008 با کتابخانه SMO جایگزین شده است.

در این حالت خواهیم داشت:

```
using System.Collections.Generic;
using System.Data;
using Microsoft.SqlServer.Management.Smo;

public class CListServers
{
   public static List<string> GetSQLServersListSMO()
   {
       List<string> result = new List<string>();
       DataTable dt = SmoApplication.EnumAvailableSqlServers(false);
       if (dt.Rows.Count > 0)
       {
           foreach (DataRow dr in dt.Rows)
           {
               result.Add(dr["Name"].ToString());
           }
       }
       return result;
   }
}
```

تقریبا کلیه اعمالی که از طریق management studio قابل انجام هستند با کمک این کتابخانه نیز از طریق برنامه نویسی می‌توان به آن‌ها پرداخت. برای مثال تهیه اسکریپت کلیه جداول ، تریگرها و غیره.

عنوان: **چرا نباید از کوئری‌های select * استفاده کرد؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۰۵ ۱۷:۱۷:۲۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

عموما اولین پاسخی که به این سؤال داده می‌شود این است که این نوع کوئری‌ها اطلاعات زیادی را باز می‌گردانند و در نتیجه ترافیک شبکه بی‌جهت افزایش خواهد یافت. اما اگر طراحی دیتابیس صحیح بوده و اصول نرمال سازی در آن پیاده سازی شده باشد، این پاسخ آنچنان صادق نخواهد بود (زیرا جداول اینگونه دیتابیس‌ها از تعداد فیلدهای بسیاری تشکیل نخواهند شد). برای مثال به نتیجه اجرای کوئری‌های زیر بر روی دیتابیس AdventureWorks با 89 هزار رکورد، دقت بفرمائید:

```
SELECT * FROM Production.TransactionHistoryArchive
WHERE ReferenceOrderID < 100

SELECT ReferenceOrderLineID FROM Production.TransactionHistoryArchive
WHERE ReferenceOrderID < 100
```

اختلاف ترافیک شبکه در این مثال تنها 15K یا حدودا 10 درصد است (180K در مقابل 165K). هر چند ارزش بررسی و برطرف کردن را دارد اما تفاوت حاصل آنچنان قابل ملاحظه نیست.
مهم‌ترین دلیلی که اینجا باید به آن دقت داشت، تفاوت چشمگیر execution plan این دو کوئری (Ctrl-L) و بحث index coverage است. اس کیوال سرور برای اجرای بهینه کوئری‌ها از ایندکس‌های موجود استفاده خواهد کرد. اگر ایندکس تعریف شده از تمامی فیلدهای درخواستی شما تشکیل شده باشد، دیگر حتی به سراغ جدول هم نخواهد رفت (به این مفهوم، پوشش ایندکسی گفته می‌شود).

برای تولید تصویر فوق، کلیدهای Ctrl+L را در management studio فشار دهید.
این دیتابیس را از آدرس زیر می‌توانید دریافت کنید:
http://www.codeplex.com/MSFTDBProdSamples

کوئری اول از مزایای پوشش ایندکسی برخوردار نخواهد بود (از روش جستجوی Clustered Index استفاده می‌کند) و در حالت دوم از Index Seek استفاده می‌گردد. حالت Index Seek یک‌صد بار بهینه‌تر از استفاده از Clustered Index عمل می‌کند زیرا در

حالت کوئری اول باید تمامی رکوردهای جدول بررسی شوند (این عدد از مقایسه نتایج execution plan بدست آمده است).
تنها در صورتیکه بر روی تمامی فیلدهای جدول ایندکس تعریف کرده باشید (که اصلا توصیه نمی‌شود)، کوئری اول توسط ایندکس‌ها پوشش داده شده و سریع اجرا خواهد شد.
بنابراین اگر از کندی اجرای کوئری‌ها با تعداد رکورد بالا شکایت دارید بهتر است نگاهی به نحوه تعریف آنها داشته باشید و تنها فیلدهایی را در کوئری تعریف کنید که به آنها نیاز دارید. در این حالت از مزایای پوشش ایندکسی برخوردار شده ، کوئری‌های سریعتری را خواهید داشت و همچنین در این حالت میزان مصرف CPU و حافظه نیز بر روی سرور کمتر خواهد بود.
همچنین در حالت کوئری‌هایی از نوع دوم ذکر شده، موتور بهینه ساز اس کیوال سرور پیشنهادات بهتری را برای ایجاد ایندکس‌های جدید و گوشزد نمودن کمبود آنها با ارائه included columns مناسب، ارائه می‌دهد.
بعلاوه مشخص ساختن تعداد دقیق فیلدهای مورد نیاز، نگهداری برنامه را ساده‌تر ساخته و فیلدهای اضافه شده آتی سبب تغییر رفتار کوئری‌ها برنامه نخواهند شد و استفاده نکردن از آن نشانه این است که هیچ برآوردی از ابعاد واقعی کار در دست نیست.

مآخذ:

Speed Up Your Site! 8 ASP.NET Performance Tips
The real reason SELECT * queries are bad: index coverage

نظرات خوانندگان

نویسنده: حمیدرضا
تاریخ: ۱۳۸۷/۰۹/۰۵ ۱۹:۰۲:۰۰

چند وقتیه که وبلاگ شما رو می‌خونم، استفاده می‌کنم و لذت می‌برم، گفتم حتی شده به قیمت یه نظر نابجا و بی‌ربط به موضوع هم شده، ازتون تشکر بکنم و براتون آرزوی موفقیت.
موفق باشید.

عنوان: **منسوخ شده‌ها در نگارش‌های جدید SQL server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۱۱ ۱۸:۳۶:۰۹
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

با تکامل SQL server و بهبودهای حاصل شده، یک سری از ویژگی‌های موجود صرفا جهت حفظ سازگاری با نگارش‌های قبلی ارائه می‌شوند. لیست کامل آنها را در آدرس زیر می‌توان مشاهده نمود:

Deprecated Database Engine Features in SQL Server 2008

لیست بلند بالایی است. اما در یک محیط کاری، نوع‌های زیر از سایر موارد ذکر شده بیشتر مورد استفاده قرار می‌گیرند:
منسوخ شده‌ها: ntext ، text و image . جایگزین‌ها : nvarchar ، varchar و varbinary از نوع max دار

عموما علت استفاده از نوع‌های text یا ntext (نمونه یونیکد text) ، مشخص نبودن تعداد کاراکتری است که کاربر قرار است وارد کند. برای مثال یک سایت خبری ایجاد کرده‌اید و طول محتوای خبر ثبت شده در بانک اطلاعاتی از یک خبر به خبر دیگر کاملا متفاوت است. در اینجا برای حل این مشکل از نوع‌های text یا ntext استفاده می‌شد (این مورد تا اس‌کیوال سرور 2000 توصیه می‌شود).
varchar max تا 2,147,483,648 کاراکتر را می‌تواند ذخیره کند، یعنی تا 2 GB و nvarchar max تا نصف این مقدار را. در اس کیوال سرور 2000 محدودیت 8000 کاراکتر برای نوع vrachar وجود داشت (و نوع nvrachar تا 4000 کاراکتر).

مزایای استفاده از نوع‌های max دار (از اس کیوال سرور 2005 به بعد) :
بهبود کارآیی کوئری‌های جستجو نسبت به نوع‌های Text
اگر مطلب تشخیص کمبود ایندکس‌ها را دنبال کرده باشید، در آنجا ذکر شد که در قسمت included columns نمی‌توان از text و ntext استفاده کرد اما نوع‌های max دار متنی مجازند.
امکان استفاده از فیلدهای max دار برای مرتب سازی کوئری مجاز است. (به شخصه با این مورد زیاد برخورد داشتم. برای مثال امکان سورت کردن یک گرید را در ASP.Net فراهم کرده‌اید و کاربر با کلیک بر روی سر ستون فیلدی از نوع ntext با یک خطا متوقف خواهد شد)
امکان استفاده از نوع‌های Text به‌عنوان متغیر در رویه‌های ذخیره شده یا توابع T-SQL مهیا نیست اما این محدودیت در نوع‌های max دار برطرف شده است.
نوع‌های text را در توابع CHARINDEX ، REPLACE و SUBSTRINGنمی‌توان بکار برد (برخلاف نوع‌های متنی max دار).
بعد از این توضیحات شاید علاقمند شده باشید که نوع‌های فیلدهای قدیمی را به نوع‌های جدید تبدیل کنید (مراجعه به management studio ، تغییر نوع فیلد و کلیک بر روی دکمه ذخیره در نوار ابزار آن). اگر تعداد رکوردها بالا باشد، بدون شک با خطای زیر متوقف خواهید شد:

```
Timeout expired.  The timeout period elapsed prior to completion of the operation or the server is not responding.
```

برای حل این مشکل می‌توان مقدار پیش فرض timeout را مطابق تصویر زیر تغییر داد (منوی tools گزینه options) :

بعد از این تغییر به سادگی می‌توان عملیات ارتقاء را بدون نگرانی از بروز خطای فوق انجام داد.

عنوان: پیدا کردن وابستگی‌های اشیاء در SQL Server
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۲۳ ۲۱:۴۵:۰۷
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

با بالا رفتن تعداد اشیاء تعریف شده در SQL server ، نگهداری آنها نیز مشکل‌تر می‌شود. در این حالت تغییر کوچکی در یکی از اشیاء ممکن است باعث از کار افتادن قسمتی از سیستم شود. بنابراین قبل از هر گونه تغییری در یک شیء، ابتدا باید سایر اشیاء وابسته به آن را یافت و در نظر داشت (dependencies). برای این منظور (impact analysis) راه‌کارهای مختلفی در SQL server وجود دارد که در ادامه به آن‌ها خواهیم پرداخت:

الف) استفاده از امکانات management studio (اس کیوال سرور 2005 به بعد)

ساده‌ترین راه ممکن که گزارش مفصلی را نیز ارائه می‌دهد، کلیک بر روی یک شیء در management studio و انتخاب گزینه view dependencies است (شکل زیر).

در صفحه ظاهر شده می‌توان اشیایی را که شیء مورد نظر به آنها وابسته است، مشاهده نمود یا برعکس (اشیایی که عملکرد آنها وابسته به شیء انتخابی است را نیز می‌توان ملاحظه کرد).

ب) کوئری گرفتن از جداول سیستمی

امکانات قسمت قبل را با استفاده از اطلاعات جدول syscomments نیز می‌توان شبیه سازی کرد. در این جدول اطلاعات تعاریف کلیه trigger ، view ، رویه‌های ذخیره شده و غیره نگهداری می‌شود. برای مثال فرض کنید قصد داریم در جدول Orders دیتابیس Northwind ، نام فیلد OrderDate را تغییر دهیم. قبل از این‌کار بهتر است کوئری زیر را اجرا کنیم تا نام اشیاء وابسته را بدست آوریم:

```
SELECT NAME
FROM   syscomments c
   JOIN sysobjects o
        ON  c.id = o.id
```

```
WHERE  TEXT LIKE '%OrderDate%'
   AND TEXT LIKE '%Orders%'
```

Results | Messages

	name
1	Orders Qry
2	Quarterly Orders
3	Invoices
4	Sales by Category
5	CustOrdersOrders
6	SalesByCategory

این روش انعطاف پذیری بیشتری را نسبت به امکانات قسمت الف ، ارائه می‌دهد. برای نمونه فرض کنید می‌خواهید در یک دیتابیس کلیه اشیایی که عملیات delete را انجام می‌دهند پیدا کنید (جستجوی اشیاء حاوی یک عبارت خاص). در این مورد خواهیم داشت:

```
SELECT NAME
FROM   syscomments c
   JOIN Northwind.dbo.sysobjects o
        ON  c.id = o.id
WHERE  TEXT LIKE '%delete%'
```

ج) استفاده از رویه ذخیره شده سیستمی sp_depends

جدول سیستمی دیگری در اس کیوال سرور به نام sysdepends وجود دارد که اطلاعات وابستگی‌های اشیاء در آن‌ها نگهداری می‌شود. برای دسترسی به اطلاعات این جدول ، اس کیوال سرور رویه ذخیره شده سیستمی sp_depends را ارائه داده است. برای مثال فرض کنید می‌خواهیم لیست اشیایی را که به جدول Oredres دیتابیس Northwind وابسته هستند، پیدا کنیم. در این حالت داریم:

```
USE Northwind
EXEC sp_depends 'Orders'
```

Results | Messages

	name	type
1	dbo.CustOrderHist	stored procedure
2	dbo.CustOrdersOrders	stored procedure
3	dbo.Employee Sales by Country	stored procedure
4	dbo.Invoices	view
5	dbo.Orders Qry	view
6	dbo.Product Sales for 1997	view
7	dbo.Quarterly Orders	view
8	dbo.Sales by Category	view
9	dbo.Sales by Year	stored procedure
10	dbo.Sales Totals by Amount	view
11	dbo.SalesByCategory	stored procedure
12	dbo.Summary of Sales by Quarter	view
13	dbo.Summary of Sales by Year	view

د) استفاده از schema view

با استفاده از view سیستمی INFORMATION_SCHEMA.ROUTINES ، که از ترکیب جداول syscolumns و sysobjects ایجاد شده است نیز می‌توان عملکرد sp_depends را شبیه سازی کرد اما جداول و view ها از گزارش آن حذف شده‌اند.

```
SELECT routine_name,
   routine_type
FROM   INFORMATION_SCHEMA.ROUTINES
WHERE  routine_definition LIKE '%Orders%'
```

در جدول زیر مقایسه‌ای از امکانات و گزارش حاصل از این چهار روش با هم مقایسه شده‌اند:

Method	Type	Tables	Views	UDF	Stored Procedures	Triggers	CHECK Constraints
View Dependencies	X	X	X	X	X	X	X
syscomments query			X	X	X	X	X
schema view	X			X	X		
sp_depends	X		X	X	X		X

ه) استفاده از برنامه SQL Dependency Tracker

نسخه آزمایشی برنامه ذکر شده را از این آدرس می‌توان دریافت کرد.

همانطور که ملاحظه می‌کنید این جستجوها بر روی اطلاعات ذخیره شده در اس کیوال سرور صورت می‌گیرند و اگر در کدهای خود در خارج از اس کیوال سرور مخلوطی از عبارات اس کیوال را داشته باشید، نگهداری آنها بسیار مشکل خواهد بود. بنابراین تا حد ممکن باید عملیات مرتبط را در دیتابیس و توسط اشیاء اس کیوال سرور مانند رویه‌های ذخیره شده، view ها و امثال آن‌ها انجام داد تا این جدا سازی به‌خوبی صورت گرفته و در زمان نیاز به انجام تغییرات، ردگیری اشیاء وابسته به‌سادگی صورت گیرد.

عنوان: SQL Server 2005 SP3
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۲۷ ۰۰:۴۳:۲۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

خبر! نگارش نهایی سرویس پک 3 اس کیوال سرور 2005 ارائه شد.

دریافت

لیست باگ‌های فیکس شده

نظرات خوانندگان

نویسنده: مهدی پایروند
تاریخ: ۱۳۸۷/۱۲/۱۸ ۱۰:۴۳:۰۰

سلام، اگه که سرویس پک های قبلی رو نصب نکرده باشیم هم میتونیم سرویس پک 3 رو نصب کنیم یا نه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۱۸ ۱۲:۲۸:۰۰

بله.مطابق مستندات رسمی آن:

SP3 can be used to upgrade from any previous instance of SQL Server 2005

http://blogs.technet.com/dataplatforminsider/archive/2008/12/10/sql-server-2005-sp3-now-available-for-download.aspx

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۱۸ ۱۲:۲۹:۰۰

ضمنا برای sp3 هم آپدیت اومده که باید نصب شود:

http://support.microsoft.com/kb/959195

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۲۸ ۱۴:۰۲:۰۰

SQL Server 2005 SP4

عنوان: **به روز رسانی View ها و رویه‌های ذخیره شده در SQL server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۳۰ ۱۶:۲۸:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

یکی دیگر از معایب کوئری‌های select * در SQL server این است که تغییرات حاصل در فیلدهای جداول یک بانک اطلاعاتی را در view های ساخته شده از این نوع کوئری‌ها منعکس نمی‌کند.
برای مثال جدول tblTreeItems را با سه فیلد parent ، id و title در نظر بگیرید. فرض کنید بر این اساس view زیر را ساخته‌ایم:

```
CREATE VIEW GetData
as
SELECT * FROM tblTreeItems
```

اکنون به جدول فوق ، فیلد جدید isActive را اضافه می‌کنیم. پس از این عملیات اگر کوئری ساده SELECT * FROM GetData را اجرا کنیم، فیلد جدید isActive را در آن نخواهیم دید (برخلاف انتظار که می‌بایست کوئری select * رکوردهای تمام فیلدهای جدول را بر می‌گرداند. در این‌جا ممکن است مدتی وقت صرف دیباگ کردن سیستم شود که چرا تغییرات جدید اعمال نشده و چرا سیستمی که تا چند لحظه پیش داشت کار می‌کرد الان از کار افتاد!).
باید در نظر داشت که هنگام ایجاد یک view ، تصویری از تمامی فیلدهای مورد استفاده در آن زمان، جهت بالابردن کارآیی کوئری و عدم محاسبه مجدد فیلدها در جداول سیستمی ذخیره می‌گردد (* با نام فیلدهای همان زمان ایجاد (نه زمان فعلی)، جایگزین خواهد شد). این تصویر ایستا است و با تغییر فیلدهای یک جدول به روز نخواهد شد.
برای به روز کردن view ها و stored procedures پس از تغییرات ساختاری در جداول، باید مجددا آنها را کامپایل کرد. برای این منظور راه‌های زیادی وجود دارد، برای مثال drop کردن یک view و ایجاد مجدد آن. یا باز کردن آن view در management studio (حالت alter query) و سپس فشردن دکمه F5 جهت اجرای مجدد کوئری که این‌بار بر اساس اطلاعات جدید به روز خواهد شد.
یا استفاده از رویه‌های سیستمی sp_refreshview و sp_recompile که برای کامپایل مجدد view ها و رویه‌های ذخیره شده بکار می‌روند.

برای مدیریت ساده‌تر این موارد ، اسکریپت زیر تمامی view ها و رویه‌های ذخیره شده یک دیتابیس را به صورت خودکار یافته و آنها را مجددا کامپایل می‌کند: (جهت مشاهده آن نیاز به ثبت نام دارد و رایگان است)
Refreshing Views and Recompiling Stored Procs

عنوان: **نحوه راه اندازی مجدد یک دیتابیس اس کیوال سرور پس از پر شدن هارد دیسک**
نویسنده: وحید نصیری
تاریخ: ۰۹:۵۵:۲۳ ۱۳۸۷/۱۰/۰۴
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

امروز یکی از برنامه‌ها (برنامه ASP.Net) با مشکل زیر مواجه شده بود:

پیغام خطا:
اتصال با سرور اس کیوال قطع شده است. لطفا با مسئول مربوطه هماهنگ نمائید.
SQLErr: 4060

این خطا به معنای عدم امکان باز کردن دیتابیس است.

در طی این مدت با موارد زیادی از این دست (مشکلات مختلف عدم امکان برقراری ارتباط با اس کیوال سرور) برخورد داشتم که خلاصه تمام آن‌ها تابع زیر شده است:

```
public void CheckSQLServerStat(Exception ex)
{
    try
    {
        SqlException ar = (SqlException) ex;
        switch (ar.Number)
        {
            case 2:
            case 11:
            case 17:
            case 40:
            case 4060:
            case 1326:
            case 17142:
            case 18456:
                HttpContext.Current.Response.Write("<br/>" + ".اتصال با سرور اس کیوال قطع شده است
لطفا با مسئول مربوطه هماهنگ نمائید." + "<br/> SQLErr:" + ar.Number + "<br/>");
                break;
        }
    }catch{}
}
```

هنگام رخ‌دادن یک خطا (در توابعی که با اس کیوال سرور کار می‌کنند)، exception حاصل را به این تابع پاس کرده و خطای حاصل را به صورت مختصر و مفید به کاربر نشان می‌دهم. کاربر متوجه می‌شود که یک مشکل اساسی در سیستم رخ داده است. برنامه نویس هم با جستجو در مورد شماره خطا می‌تواند مشکل یابی کند. تابع Response.Write هم بالاتر از هر المان دیگری در صفحه، این خطا را به شکل واضحی نمایش می‌دهد. (کلا ریخت صفحه را به هم می‌ریزد که از لحاظ روانی لازم است! چون عملا در این حالت سیستم از کار افتاده است)

به management studio اس کیوال سرور که مراجعه کردم، علامت خاصی کنار نام دیتابیس نبود فقط برخلاف سایر دیتابیس‌ها که آیکون + مربوط به باز شدن tree آن وجود دارد، این یک مورد آن‌را نداشت. بر روی نام دیتابیس کلیک راست کردم و انتخاب خواص، خطای زیر نمایش داده شد:

```
------------------------------
An exception occurred while executing a Transact-SQL statement or batch.
(Microsoft.SqlServer.ConnectionInfo)
------------------------------
Database 'dbName' cannot be opened due to inaccessible files or insufficient memory or disk space.  See
the SQL Server errorlog for details. (Microsoft SQL Server, Error: 945)
------------------------------
```

بله! دیتابیس قابل باز شدن نبود چون درایو مربوطه پر شده بود. بعد از خالی کردن درایو و باز کردن فضای لازم، باز هم دیتابیس به حالت اجرایی و قابل استفاده برنگشت. یک راه این است که کل سرویس مربوط به اس کیوال سرور را استاپ و استارت کرد. که البته این مورد سبب قطع ارتباط کل مجموعه می‌شود. یا راه دیگر اجرای چند سطر زیر است که دیتابیس را مجددا راه اندازی خواهد کرد (بدون نیاز به ری استارت سرویس)

```
use master;
alter database dbName set OFFLINE;
alter database dbName set online;
```

به این صورت دیتابیس مربوطه به حالت عادی بازگشت و قابل استفاده شد.

عنوان: **مشکل اتصال به اس کیوال سرور 2000 از طریق management studio 2008**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۱۵ ۱۱:۱۱:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

هر چند مدتی است که اس کیوال سرور 2008 ارائه شده یا سرویس پک 3 اس کیوال سرور 2005 اخیرا منتشر گردیده، اما هنوز هم هاست‌های ما زحمت ارتقاء را به خود نداده‌اند. (چرا باید پول خرج کنند؟!)
اس کیوال سروری را که الان برای برنامه نویسی از آن استفاده می‌کنم 2008 است و چند روز قبل قصد داشتم به اس کیوال سرور 2000 هاست محترم از طریق management studio 2008 متصل شوم تا تغییرات جدید را به دیتابیس سایت اعمال کنم.
اولین تلاش برای اتصال با خطای زیر متوقف شد!

```
TITLE: Microsoft SQL Server Management Studio
```

```
------------------------------
```

```
Failed to retrieve data for this request. (Microsoft.SqlServer.Management.Sdk.Sfc)
```

```
------------------------------
```

```
ADDITIONAL INFORMATION:
```

```
An exception occurred while executing a Transact-SQL statement or batch.
(Microsoft.SqlServer.ConnectionInfo)
```

```
------------------------------
```

(Server user 'hostXYZ' is not a valid user in database 'hostZXY'. (Microsoft SQL Server, Error: 916

```
------------------------------
```

فقط هم management studio 2008 این مشکل را دارد. با 2005 تست کردم مشکلی نبود.
پس از مدتی جستجو ، نتیجه کار به صورت زیر است:
به منوی View گزینه Object Explorer Details مراجعه کنید (در management studio اس کیوال سرور 2008). روی هدر صفحه‌ای که ظاهر می‌شود کلیک راست کرده و تیک collation را بردارید (شکل زیر). ممکن است collation دیتابیس آفلاین، نال گزارش شود و این مورد مشکل درست می‌کند. تیک collation را که بردارید این مورد بررسی نشده و لیست دیتابیس‌ها نمایش داده می‌شود.

Object Explorer Details
Search
sql.[illegible].com (SQL Server 8.0.2050 - host[illegible])\Databases
Name
Policy Health State
System Databases
Sort Ascending
Sort Descending
Name
Policy Health State
Recovery Model
Compatibility Level
Collation
Owner

نظرات خوانندگان

نویسنده: msj
تاریخ: ۱۳۸۹/۰۷/۲۹ ۱۵:۰۱:۲۲

فکر می کنم که این مشکل در Service Pack 1 که برای SQL 2008 اومد حل شد.

عنوان: SQL Server 2008: Script Data
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۱۷ ۱۲:۵۷:۰۳
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

افزونه‌ای برای SQL server 2005 به نام Database Publishing Wizard وجود داشت/دارد که توسعه‌ی آن به ظاهر برای SQL server 2008 متوقف شده است. توسط این افزونه می‌توان رکوردهای یک دیتابیس را به صورت عبارات T-SQL درآورد (هر رکورد را به صورت خودکار تبدیل به یک دستور insert می‌کند). به این صورت کار انتقال دیتا خصوصا به هاست‌هایی که دسترسی مستقیم restore کردن داده را نمی‌دهند، به سادگی صورت می‌گیرد. تنها کافی است خروجی کار یکبار بر روی دیتابیس مقصد اجرا شود تا رکوردهایی دقیقا با همان اطلاعات دیتابیس منبع در آن ایجاد گردند.
این قابلیت اکنون جزئی از management studio اس کیوال سرور 2008 است.
برای استفاده از آن بر روی دیتابیس مورد نظر در management studio 2008 کلیک راست کرده و گزینه زیر را انتخاب کنید:

```
Tasks -> Generate scripts…
```

در صفحه ویزاردی که ظاهر می‌شود، بر روی next کلیک کرده و در صفحه‌ی بعدی گزینه script data را یافته و مقدار آن‌را به true تنظیم نمائید (شکل زیر).

سپس بر روی next کلیک کرده و در صفحه بعد گزینه tables را انتخاب کنید. در ادامه با کلیک بر روی دکمه next ، در صفحه‌ی بعدی می‌توان تمامی جداول و یا تنها جداول مورد نظر را جهت تهیه اسکریپت داده‌های آن‌ها انتخاب نمود و در آخر عملیات تهیه اسکریپت insert داده‌ها به صورت خودکار صورت خواهد گرفت.

برای مشاهده تصاویر بیشتری از این عملیات ساده می‌توان به این مقاله رجوع نمود.

شایان ذکر است این قابلیت با نگارش‌های پائین‌تر اس کیوال سرور نیز کار می‌کند (برای مثال اتصال به اس کیوال سرور 2000 از طریق management studio 2008).

عنوان: **مشکل ی و ک فارسی و عربی در یک دیتابیس اس کیوال سرور**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۲۴ ۱۷:۵۹:۳۸
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

دیروز به من اطلاع دادند که در یکی از برنامه‌ها دو تا گروه "تاسیسات مکانیکی" پیدا شده!!
تاسیسات مکانیکی
تاسیسات مکانیکی

استاندارد این شرکت، استفاده از kbdfa.dll مخصوص و نسبتا قدیمی است. بنابراین استاندارد مورد استفاده همان ی و ک عربی است. (کاری ندارم خوب است یا بد، یا باید اینطور باشد یا نه، بحث این است که فعلا اینطور است و قرار نیست چیزی عوض بشود!)
در مثال فوق، ی و ک عبارت دوم فارسی است. یعنی نصب kbdfa.dll روی ویندوز تازه نصب شده، فراموش شده بوده.

راه حل‌ها:
الف) قبل از ثبت، یکسان سازی صورت گیرد. یعنی اجرای متدی شبیه به متد زیر بر روی هر ورودی متنی فارسی:

```
public string SafeFarsiStr(string input)
   {
       return input.Replace("ی" ,"ی").Replace("ک" ,"ک");
   }
```

ب) خوب، الان که این یکسان سازی صورت نگرفته چه باید کرد؟
اسکریپتی را تهیه کرده‌ام (مخصوص SQL Server 2005 به بعد) به صورت زیر که این تبدیل را برای شما انجام می‌دهد.
به صورت خودکار تمامی فیلدهای متنی کلیه جداول دیتابیس جاری شما را یافته و ی و ک آن‌ها را یکسان می‌کند. البته همانطور که عرض شد، مطابق استاندارد این شرکت و استفاده از فایل kbdfa.dll قدیمی مورد استفاده، تمام ی و ک های فارسی به عربی تبدیل می‌شوند.

```
--اسکریپتی برای یک دست سازی ی و ک در تمامی رکوردهای تمامی جداول دیتابیس جاری
-- اسکریپت زیر ی و ک فارسی را به عربی تبدیل می‌کند
-- در صورت نیاز به حالت عکس ، جای مقادیر عددی یونیکد را تعویض نمائید

USE TestDb;

DECLARE @Table NVARCHAR(MAX),
       @Col NVARCHAR(MAX)

DECLARE Table_Cursor CURSOR
FOR
   --پیدا کردن تمام فیلدهای متنی تمام جداول دیتابیس جاری
   SELECT a.name, --table
          b.name --col
   FROM   sysobjects a,
          syscolumns b
   WHERE  a.id = b.id
          AND a.xtype = 'u' --User table
          AND (
                 b.xtype = 99 --ntext
                 OR b.xtype = 35 -- text
                 OR b.xtype = 231 --nvarchar
                 OR b.xtype = 167 --varchar
                 OR b.xtype = 175 --char
                 OR b.xtype = 239 --nchar
              )

OPEN Table_Cursor FETCH NEXT FROM  Table_Cursor INTO @Table,@Col
WHILE (@@FETCH_STATUS = 0)
BEGIN
```

```
   EXEC (
            'update [' + @Table + '] set [' + @Col +
            ']= REPLACE(REPLACE(CAST([' + @Col +
            '] as nvarchar(max)) , NCHAR(1740), NCHAR(1610)),NCHAR(1705),NCHAR(1603)) '
        )

   FETCH NEXT FROM Table_Cursor INTO @Table,@Col
END CLOSE Table_Cursor DEALLOCATE Table_Cursor
```

توضیحات و نکاتی در مورد اسکریپت فوق:

الف) برای آشنایی با انواع XType Datatype مورد استفاده در کوئری فوق به این آدرس مراجعه نمائید.

ب) همانطور که در مطالب قبلی این وبلاگ نیز ذکر شد ، امکان استفاده از تابع replace بر روی فیلدهای text و ntext وجود ندارد. هیچ اشکالی ندارد! تمام آن‌ها به nvarchar از نوع max دار cast شده و این مشکل به این صورت حل می‌شود.

ج) اس کیوال سرور اجازه تعریف یک جدول یا فیلد را به صورت متغیر بکار رفته در یک کوئری T-SQL نمی‌دهد. برای حل این موضوع باید عبارت SQL مورد نظر را به صورت یک رشته درآورد و سپس exec کرد.

د) مجبور شدم از معادل‌های عددی برای دقت بیشتر کار استفاده کنم

(در کل از تابع UNICODE اس کیوال سرور برای بدست آوردن این اعداد می‌توان استفاده کرد)

تذکر: این اسکریپت بر روی یک دیتابیس کاری تست شده و نتیجه رضایت بخش بوده؛ اما اگر شما روزی خواستید از آن استفاده کنید، حتما full backup را قبل از اجرای آن فراموش نکنید و پس از اجرا، تابع SafeFarsiStr فوق را برای ادامه کار حتما لحاظ نمائید یا از یک kbdfa.dll هماهنگ استفاده کنید.

نظرات خوانندگان

نویسنده: Alex
تاریخ: ۰۷:۵۲:۰۰ ۱۳۸۷/۱۰/۲۵

خیلی به درد بخور بود.

نویسنده: افشار
تاریخ: ۰۸:۱۵:۰۰ ۱۳۸۷/۱۰/۲۵

بعضی نسخه‌های kbdfa.dll بر اساس استاندارد ۲۹۰۱ هستند. استاندارد ۲۹۰۱ توسط استاندارد جدیدتر ۹۱۴۷ باطل اعلام گردید. البته اختلاف این دو تا استاندارد خیلی توی چشم نمی‌زند.
یک درایور خوب هم برای استاندارد ۹۱۴۷ وجود دارد که من در ویندوز ویستا و چندین ویندوز ایکس پی از آن استفاده می‌کنم بدون این که مشکلی برایم به وجود آمده باشد.

آدرس درایور: http://prdownloads.sourceforge.net/farsitools/persiankeyboard.zip?download

مطالعه استاندارد: http://fa.farsiweb.ir/mediawiki-fa/images/a/a9/Isiri-9147.pdf

لینک استاندارد در سایت سازمان استاندارد: http://www.isiri.org/asp/account/checklog.asp?ID=9147.pdf

بیاید از زبان فارسی حمایت کنیم.

نویسنده: hajloo
تاریخ: ۱۴:۳۶:۰۰ ۱۳۸۷/۱۰/۲۶

مطلب خوبی بود اما من هم با افشار موافقم که باید از فارسی حمایت کرد و بهتره که بجای جایگزینی حروف فارسی با عربی ، بهتره که از فونتهای استاندارد استفاده بشه و حروف عربی به فارسی تبدیل بشن
-
از مطلب خوبتون هم واقعا ممنونم

نویسنده: وحید نصیری
تاریخ: ۱۶:۰۴:۰۰ ۱۳۸۷/۱۰/۲۶

اگر این مورد به نحوی در خود برنامه‌های پایگاه داده حل بشود، این مسایل دیگر وجود نخواهد داشت.
مثلا اس کیوال سرور ی و ی را به یک صورت پردازش کند و جدا از هم درنظر نگیرد.
این مورد الان در جستجوی گوگل هم مشهود است و مشکل زا. شما با ی یک سری جواب می‌گیرید و با ی یک سری دیگر. این مورد مهم است!

نویسنده: معنوی
تاریخ: ۱۶:۵۸:۰۰ ۱۳۸۷/۱۱/۰۷

تو برنامه تون (دات نتی) با استفاده از MessageFilter می تونید کلیه ورودی ها رو تحت کنترل داشته باشید.

نویسنده: as
تاریخ: ۰۰:۱۷:۰۹ ۱۳۸۹/۰۵/۲۷

مرسی از کمکت عالی بود...

نویسنده: msj
تاریخ: ۱۳۸۹/۰۷/۲۹ ۱۴:۵۳:۳۶

با سلام و تشکر از مطالب مفیدتون.
استاندارد سیستم من فارسی هست و میخوام همون هم ذخیره بشه.
من این اسکریپت رو اجرا کردم (البته با تغییر در replace و جابه جا کردن فارسی و عربی) و دیتابیس رو درست کرد.
فقط الان یه مشکلی دارم(البته فکر می کنم قبل از این اسکریپت هم بوده): هر insert که می زنم ، خود SQL "ی" فارسی رو به "ی" عربی تبدیل می کنه و ذخیره می کنه. یعنی من 1740 می فرستم ولی 1610 ذخیره میشه. این مشکل در مورد ک وجود نداره و همون 1705 فرستاده و ذخیره میشه.
من از SQL 2005 express و Collation Arabic_CI_AS استفاده می کنم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۹ ۱۶:۵۲:۱۲

مشکل از SQL Server نیست. مشکل از درایور صفحه کلید شما است که به نظر ایراد دارد. باید ورودی کدهای خودتون رو در تمام قسمت‌هایی که insert دارد بررسی کنید (قبل از ورود به دیتابیس این تصحیح باید انجام شود مانند استفاده از تابع SafeFarsiStr فوق).

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۹ ۱۷:۰۶:۰۲

اثبات این مدعا هم که مشکل از SQL Server نیست ساده است (فیلد با collation عربی ایجاد شده و ی فارسی در آن ثبت شده است و سپس گزارشگیری):

```
(DECLARE @tbl AS TABLE (f1 NVARCHAR(50) COLLATE Arabic_CI_AS NULL
((INSERT INTO @tbl(f1) VALUES(NCHAR(1740
SELECT * FROM @tbl
```

نویسنده: ساماني
تاریخ: ۱۳۹۱/۰۴/۲۷ ۱۱:۲۵

واقعا عالی بود

عنوان: **مونیتور کردن میزان مصرف CPU در اس کیوال سرور**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۳۰ ۱۵:۱۰:۳۶
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

در این مقاله قصد داریم نحوه مونیتور کردن میزان مصرف CPU توسط اس کیوال سرور را بررسی کنیم. برای بدست آوردن میزان CPU مصرفی اس کیوال سرور می‌توان به اسکریپت زیر رجوع کرد:

```
DECLARE @CPU_BUSY int, @IDLE int
SELECT @CPU_BUSY = @@CPU_BUSY, @IDLE = @@IDLE WAITFOR DELAY '000:00:01'
SELECT (@@CPU_BUSY - @CPU_BUSY)/((@@IDLE - @IDLE + @@CPU_BUSY - @CPU_BUSY) *1.00) *100 AS 'CPU
Utilization by sqlsrvr.exe'
```

ماخذ

در ادامه قصد داریم، هر 5 دقیقه به صورت خودکار بررسی کنیم که آیا میزان مصرف CPU در اس کیوال سرور بالای 50 درصد است؟ و اگر بله، ایمیلی را به مسؤول مربوطه جهت بررسی ارسال کنیم.

بنابراین اولین کاری که باید صورت گیرد، فعال سازی Database Mail در اس کیوال سرور است که به صورت پیش فرض غیرفعال است. برای این منظور تنها کافی است اسکریپت زیر را بر روی سرور اجرا کنید:

```
USE [master]
GO
sp_configure 'show advanced options',1
GO
RECONFIGURE WITH OVERRIDE
GO
sp_configure 'Database Mail XPs',1
GO
--RECONFIGURE
GO
-- Create a New Mail Profile for Notifications
EXECUTE msdb.dbo.sysmail_add_profile_sp
     @profile_name = 'DBA_Notifications',
     @description = 'Profile for sending Automated DBA Notifications'
GO
-- Set the New Profile as the Default
EXECUTE msdb.dbo.sysmail_add_principalprofile_sp
  @profile_name = 'DBA_Notifications',
  @principal_name = 'public',
  @is_default = 1 ;
GO
-- Create an Account for the Notifications
EXECUTE msdb.dbo.sysmail_add_account_sp
  @account_name = 'SQLMonitor',
  @description = 'Account for Automated DBA Notifications',
  @email_address = 'nasiri@site.net',  -- Change This
  @display_name = 'SQL Monitor',
  @mailserver_name = 'mail.site.net'  -- Change This
GO
-- Add the Account to the Profile
EXECUTE msdb.dbo.sysmail_add_profileaccount_sp
  @profile_name = 'DBA_Notifications',
  @account_name = 'SQLMonitor',
  @sequence_number = 1

GO
```

ماخذ

این اسکریپت برای اس کیوال سرورهای 2005 به بعد طراحی شده و تنها دو سطر آن‌را پیش از اجرا باید ویرایش کنید. سطر مربوط به email_address و mailserver_name . آدرس ایمیل درحقیقت آدرس ایمیل قسمت from پیغام ارسالی را تشکیل می‌دهد. نام سرور میل هم، منظور آدرس smtp server شما در شبکه است.

یا اگر علاقمند بودید که این‌کار را توسط ویزاردهای management studio انجام دهید (که در نهایت هیچ تفاوتی با اسکریپت فوق نخواهد داشت)، می‌توان به این مقاله رجوع کرد.

پس از اجرای اسکریپت فوق، برای بررسی صحت عملکرد فوق می‌توان دستور زیر را اجرا کرد:

```
--test
EXECUTE msdb.dbo.sp_send_dbmail
 @recipients = 'nasiri@site.net',  -- Change This
 @Subject = 'Test Message generated from SQL Server DatabaseMail',
 @Body = 'This is a test message from SQL Server DatabaseMail'
```

سایر پارامترهای این دستور را در MSDN می‌توان ملاحظه نمود.

تا اینجا اس کیوال سرور برای ارسال ایمیل آماده شد. در ادامه قصد داریم یک job جدید در اس کیوال سرور ایجاد کنیم تا تمام موارد فوق را لحاظ کند.

مطابق تصویر فوق ابتدا یک job جدید را آغاز خواهیم کرد.

در ادامه اسکریپت زیر را جهت اجرا به آن معرفی می‌کنیم. توسط این اسکریپت، میزان جاری مصرف CPU اس کیوال سرور محاسبه شده و اگر این میزان بیشتر از 50 بود، یک ایمیل به مسؤول مربوطه با ذکر میزان CPU usage ارسال می‌گردد.

```
DECLARE @CPUUsage INT
DECLARE @CPU_BUSY INT,
      @IDLE INT

SELECT @CPU_BUSY = @@CPU_BUSY,
     @IDLE = @@IDLE

WAITFOR DELAY '000:00:01'
SELECT @CPUUsage = (@@CPU_BUSY - @CPU_BUSY) / ((@@IDLE - @IDLE + @@CPU_BUSY - @CPU_BUSY) * 1.00)
     * 100 -- CPU Utilization by sqlsrvr.exe


IF @CPUUsage > 50
BEGIN
  DECLARE @msg NVARCHAR(1000)
  SET @msg = 'Please check SQL server, CPU usage is ' + CAST(@CPUUsage AS NVARCHAR(50))
      + '%.'

  EXECUTE msdb.dbo.sp_send_dbmail
  @recipients = 'nasiri@site.net', -- Change This
  @copy_recipients = 'nasiri@site.net', -- Change This
  @Subject = 'CPU overload',
  @Body = @msg
  ,@importance = 'High'
END
```

و در آخر زمان اجرای آن را به هر روز، هر 5 دقیقه یکبار تنظیم خواهیم کرد.

اگر نیاز به راه حلی پخته‌تر و بررسی متوسط چندین مقدار قبلی ، مقایسه آن‌ها و سپس ارسال ایمیل داشتید، می‌توان به فصل 14 کتاب Super SQL Server Systems مراجعه کرد.

عنوان: **تنظیمات پیشنهادی حداکثر حافظه‌ی مصرفی اس کیوال سرور**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۱/۰۶ ۱۶:۳۶:۱۱
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

به صورت پیش فرض تنظیمات حافظه‌ی اس کیوال سرور به صورتی است که به آن اجازه می‌دهد تمامی حافظه‌ی مهیای سیستم عامل را مصرف کند! اگر شخصی با این مساله آشنایی نداشته باشد احتمالا تصور خواهد کرد که اس کیوال سرور نشتی حافظه دارد یا کلا مشکلی در سیستم روی داده است که تا این حد مصرف حافظه بالا رفته است.
برای مشاهده‌ی این تنظیمات روی instance سرور مورد نظر کلیک راست کرده و گزینه خواص را انتخاب کنید. سپس در صفحه‌ی باز شده گزینه memory را مشاهده نمائید.
البته این‌کار را با استفاده از دستورات T-SQL هم می‌توان انجام داد:

```
EXEC sp_configure 'show advanced options', 1
RECONFIGURE WITH OVERRIDE

EXEC sp_configure 'max server memory (MB)'

EXEC sp_configure 'show advanced options', 0
RECONFIGURE WITH OVERRIDE
```

نتیجه‌ی آن بر روی سیستم برنامه نویسی من به صورت زیر است:

که البته عمدا آن‌را بر روی یک گیگ تنظیم کرده‌ام تا عملکرد سیستم مختل نشود. (چون اگر غیر از این باشد، تعارفی نداشته و هر آنچه را که صلاح بداند مصرف می‌کند!)
کمی در بلاگ‌های رسمی برنامه نویس‌های اس کیوال سرور در سایت MSDN در این‌باره جستجو کردم و نتیجه آن به صورت زیر است:

تنظیمات حداکثر حافظه‌ی مصرفی اس کیوال سرورهای 2005 و 2008
برای یک سیستم عامل 64 بیتی:

MaxMem Setting	Physical RAM
1500	2GB
3200	4GB
4800	6GB
6700	8GB
10600	12GB
14500	16GB
22400	24GB
30000	32GB
45000	48GB
59000	64GB

برای سروری 32 بیتی این اعداد حداکثر را تقریبا منهای 300 مگابایت نمائید. (و البته همانطور که مطلع هستید در سیستم عامل‌های سرور 32 بیتی، اگر بیشتر از 2 گیگا بایت رم داشتید سوئیچ 3GB و اگر بیشتر از 4 گیگابایت رم مهیا بود، سوئیچ PAE باید در boot.ini تنظیم شوند. در غیر اینصورت هزینه‌ی سخت افزاری بیهوده‌ای را متحمل شده‌اید، زیرا استفاده‌ی بهینه‌ای از آن در یک سیستم عامل 32 بیتی نخواهد شد)

و این اعداد را همانطور که ذکر شد، در تنظیمات سرور می‌توان وارد نمود (از طریق management studio) و یا با استفاده از دستورات T-SQL نیز این امر میسر است: (در مثال زیر حداکثر حافظه مجاز مصرفی به 2300 مگابایت تنظیم می‌شود)

```
EXEC sp_configure 'show advanced options', 1
RECONFIGURE WITH OVERRIDE

EXEC sp_configure 'max server memory (MB)', 2300

EXEC sp_configure 'show advanced options', 0
RECONFIGURE WITH OVERRIDE
```

و البته لازم به ذکر است که اعداد فوق برای حالتی پیشنهاد شده است که سرور مورد نظر فقط یک کار و آن‌هم اجرا و سرویس دهی اس کیوال سرور را به عهده داشته باشد. بدیهی است اگر از این سرور استفاده‌های دیگری هم می‌شود این اعداد را باید کمتر نیز در نظر گرفت.

برای مطالعه‌ی بیشتر
http://sqlnerd.blogspot.com/2006/07/memory-use-in-sql-server.html
http://blogs.msdn.com/axperf/archive/2008/03/10/welcome-database-configuration-checklist-part-1.aspx
http://technet.microsoft.com/en-us/library/bb124810.aspx

نظرات خوانندگان

نویسنده: Meysam
تاریخ: ۱۳۸۹/۰۷/۲۲ ۲۲:۰۷:۰۳

اگر این سرور وظیفه ی وب سروری رو هم داشته باشه(IIS) چه درصدی رو پیشنهاد میکنید؟
البته DNS هم هست طبیعتا!!

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۲ ۲۳:۰۵:۳۱

برای IIS حدود حداقل یک گیگ و اگر تعداد کاربر زیاد است تا یک و نیم گیگ را کنار بگذارید. DNS Server زیاد حافظه مصرف نمی‌کند. یا حداقل به اندازه‌ی IIS نیست. تا 300 مگ برای آن زیاد هم هست.
بنابراین اگر مثلا سرور شما 4 گیگ رم دارد و IIS و DNS و SQL Server و غیره را با هم دارد، حداکثر حافظه مصرفی SQL Server را به حدود 2.3 گیگ محدود کنید و گرنه خیلی زود صدای کاربران سایت(ها) در خواهد آمد که باز هم کند شد ... باز هم کند شد ...

عنوان: Microsoft SQL Server 2008 Management Objects
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۱/۱۳ ۱۲:۳۵:۱۲
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

یک سرویس کوچک ویندوز ان اتی نوشته‌ام که کارش این است که در پایان هر هفته، تمام دیتابیس‌های اس کیوال سرور موجود را یافته و اسکریپت تمام اشیاء آن‌ها را به صورت خودکار تولید می‌کند (از جداول گرفته تا تریگرها، رویه‌های ذخیره شده و غیره)، سپس کل مجموعه را فشرده کرده و سپس ایمیل می‌زند. این‌کار برای نگهداری تغییرات انجام شده در طول یک هفته لازم است.
برنامه با استفاده از امکانات SMO تهیه شده است و اگر علاقمند بودید که این‌کار را انجام دهید، می‌توانید به مقاله‌های زیر رجوع کنید:

Making a database clone using SMO

Using the SqlServer.Management.Smo

SQL Server: SMO Scripting Basics

با آمدن اس کیوال سرور 2008، اشیاء SMO هم به روز شده‌اند و اگر با این اشیاء برنامه نویسی کرده باشید، برنامه بر روی سروری با اس کیوال سرور 2005 اجرا نخواهد شد و پیغام خطای زیر را دریافت خواهید کرد:

```
Could not load file or assembly 'Microsoft.SqlServer.Management.Sdk.Sfc, Version=10.0.0.0,
Culture=neutral, PublicKeyToken=89845dcd8080cc91' or one of its dependencies. The system cannot find
the file specified.
```

خوشبختانه مایکروسافت این کتابخانه‌ها را به صورت مجزا هم برای دریافت قرار داده است و می‌توان آن‌ها را نصب نمود تا برنامه بدون اشکال اجرا شود. به صفحه زیر و قسمت Microsoft SQL Server 2008 Management Objects مراجعه نمائید:
اینجا کلیک نمائید

البته همانطور که در صفحه ذکر شده نیز عنوان گردیده است، به MSXML 6.0 هم نیاز می‌باشد که لینک دریافت آن در ابتدای صفحه فوق موجود است.

عنوان: **مونیتور کردن میزان فضای خالی باقیمانده در سرور توسط اس کیوال سرور**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۱/۲۶ ۱۵:۳۲:۵۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

با توجه به در حال اجرا بودن 24 ساعته‌ی سرویس SQL server agent، استفاده‌های ارزنده‌ای از آن می‌توان کرد. برای مثال هر از گاهی بررسی کند که آیا هارد سرور پر شده یا نه؟ و اگر بله (کمبود میزان فضای خالی به حد خطرناکی رسیده)، یک ایمیل خودکار به مسؤول مربوطه ارسال کند.

عمده‌ی مطالبی که در این مقاله بررسی خواهند شد همانند مطلب مونیتور کردن میزان مصرف CPU توسط اس کیوال سرور است و از تکرار آن‌ها در این‌جا صرفنظر خواهد شد (راه اندازی دیتابیس میل و همچنین تعریف یک job جدید که در مورد آن‌ها صحبت شد، همانند قبل است). تنها مطلب جدیدی که به آن اشاره خواهد شد، اسکریپت بررسی میزان فضای خالی و سپس ارسال ایمیل است که در یک job جدید همانند مقاله‌ی قبل باید به سرور اضافه شود. این اسکریپت به شرح زیر است:

```
DECLARE @DriveBenchmark INT
DECLARE @MachineName NVARCHAR(1000)
DECLARE @DiskFreeSpace INT
DECLARE @DriveLetter CHAR(1)
DECLARE @AlertMessage NVARCHAR(MAX)
DECLARE @MailSubject NVARCHAR(MAX)
DECLARE @NewLine CHAR(2)

SET @NewLine = CHAR(13) + CHAR(10)

SET @DriveBenchmark = 2048 -- 2GB
SET @MailSubject = 'Free space is low on ' + @@SERVERNAME
SET @AlertMessage = ''

IF EXISTS (
     SELECT *
     FROM   tempdb..sysobjects
     WHERE  id = OBJECT_ID(N'[tempdb]..[#disk_free_space]')
 )
  DROP TABLE #disk_free_space

CREATE TABLE #disk_free_space
(
 DriveLetter  CHAR(1) NOT NULL,
 FreeMB       INTEGER NOT NULL
)

/* Populate #disk_free_space with data */
INSERT INTO #disk_free_space
EXEC MASTER..xp_fixeddrives


DECLARE DriveSpace CURSOR FAST_FORWARD
FOR
  SELECT DriveLetter,
         FreeMB
  FROM   #disk_free_space

 OPEN DriveSpace
 FETCH NEXT FROM DriveSpace INTO @DriveLetter, @DiskFreeSpace

WHILE (@@FETCH_STATUS = 0)
BEGIN
  IF @DiskFreeSpace < @DriveBenchmark
  BEGIN
      SET @AlertMessage = @AlertMessage + 'Drive ' + @DriveLetter + ' on ' + @@SERVERNAME
          + ' has only ' + CAST(@DiskFreeSpace AS VARCHAR) + ' MB left.' + @NewLine
  END

  FETCH NEXT FROM DriveSpace INTO @DriveLetter, @DiskFreeSpace
END
 CLOSE DriveSpace
 DEALLOCATE DriveSpace

 DROP TABLE #disk_free_space
```

```
IF @AlertMessage <> ''
BEGIN
  EXECUTE msdb.dbo.sp_send_dbmail
  @recipients = 'nasiri@site.net', -- Change This
  @copy_recipients = 'Administrator@site.net', -- Change This
  @Subject = @MailSubject,
  @Body = @AlertMessage
  ,@importance = 'High'
END
```

بررسی اسکریپت فوق:

همه چیز از رویه‌ی سیستمی xp_fixeddrives شروع می‌شود. حاصل اجرای این رویه، دریافت میزان فضای خالی هر درایو موجود در سرور خواهد بود. همانطور که در اسکریپت نیز مشخص است، برای ذخیره سازی خروجی این رویه، یک جدول موقتی (disk_free_space) ایجاد شده و خروجی آن به درون این جدول اضافه خواهد شد. سپس یک cursor ایجاد شده و تک تک رکوردهای حاصل با مقدار متغیر DriveBenchmark که در اینجا 2 گیگابایت در نظر گرفته شده است، مقایسه می‌گردند. سپس هر کدام از رکوردها که کمتر از 2 گیگابایت بود، متغیر AlertMessage ما را مقدار دهی خواهد کرد. در پایان اگر این متغیر مقدار دهی شده بود، یعنی مشکل حاصل شده و نتیجه‌ی بررسی به صورت یک ایمیل ارسال می‌گردد. بدیهی است که در صورت نیاز مقدار متغیر DriveBenchmark و آرگومان‌های recipients و copy_recipients کد فوق باید اصلاح شوند.

برای استفاده از آن یک job جدید تعریف کنید که مثلا هر سه ساعت یکبار اجرا شده و این اسکریپت را فراخوانی نماید.

عنوان: **آیا از وضعیت رویه‌های ذخیره شده‌ی دیتابیس‌های اس کیوال سرور خود خبر دارید؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۰۷ ۱۳:۴۹:۵۶
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

به لطف امکانات سیستمی اس کیوال سرورهای 2005 به بعد و DMV های آن‌ها، آمارگیری از ریز اتفاقات رخ داده در یک اس کیوال سرور این روزها بسیار ساده شده است و نیازی به ابزارهای جانبی برای انجام این نوع عملیات نیست (یا کمتر هست). در ادامه مروری خواهیم داشت بر یک سری کوئری که اطلاعات جالبی را در مورد وضعیت رویه‌های ذخیره شده‌ی دیتابیس‌های شما ارائه می‌دهند. لازم به ذکر است که اکثر این آمارها با هر بار ری استارت سرور، صفر خواهند شد.

آیا می‌دانید در یک دیتابیس خاص کدامیک از رویه‌های ذخیره شده‌ی شما بیش از سایرین مورد استفاده بود و آماری از این دست؟

```
use dbName;
SELECT TOP(100) qt.text AS 'SP Name',
     qs.execution_count AS 'Execution Count',
     qs.execution_count / DATEDIFF(Second, qs.creation_time, GETDATE()) AS
     'Calls/Second',
     qs.total_worker_time / qs.execution_count AS 'AvgWorkerTime',
     qs.total_worker_time AS 'TotalWorkerTime',
     qs.total_elapsed_time / qs.execution_count AS 'AvgElapsedTime',
     qs.max_logical_reads,
     qs.max_logical_writes,
     qs.total_physical_reads,
     DATEDIFF(Minute, qs.creation_time, GETDATE()) AS 'Age in Cache'
FROM   sys.dm_exec_query_stats AS qs
     CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) AS qt
WHERE  qt.dbid = DB_ID() -- Filter by current database
ORDER BY
     qs.execution_count DESC
```

البته مرتب سازی پیش فرض این کوئری بر اساس تعداد بار اجرا است (رویه‌های ذخیره شده‌ی محبوب!)، می‌شود آن‌را بر اساس total_worker_time (فشار بر روی CPU سیستم)، total_logical_reads (فشار بر روی حافظه)، total_physical_reads (فشار I/O کوئری‌ها)، total_logical_writes نیز مرتب کرد و نتایج جالب توجهی را بدست آورد.

آیا می‌دانید کدامیک از رویه‌های ذخیره شده‌ی شما بیش از سایرین کامپایل مجدد شده است؟

```
select top 50
  sql_text.text,
  sql_handle,
  plan_generation_num,
  execution_count,
  dbid,
  objectid
from
  sys.dm_exec_query_stats a
  cross apply sys.dm_exec_sql_text(sql_handle) as sql_text
where
  plan_generation_num >1
order by plan_generation_num desc
```

آیا می‌دانید آخرین باری که رویه‌های ذخیره شده‌ی شما ویرایش شده‌اند چه زمانی بوده است؟

```
SELECT NAME,
     create_date,
     modify_date
FROM   sys.objects
```

```
WHERE  TYPE = 'P'
ORDER BY
     Modify_Date DESC,
     NAME
```

نظرات خوانندگان

نویسنده: مهرداد قاسمی
تاریخ: ۱۳۸۷/۱۲/۰۷ ۲۱:۱۵:۰۰

بسیار جالب و مفید ...
ممنون استاد

عنوان: **تعیین اعتبار کردن یک عبارت SQL**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۱۱ ۲۰:۴۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

سؤال: چگونه می‌توان یک کوئری را تعیین اعتبار کرد بدون اینکه نتیجه‌ی اجرای آن برای ما اهمیتی داشته باشد یا دیتابیس را به هم بریزد؟

روش انجام این‌کار در اس کیوال سرور به صورت زیر است:

```
SET NOEXEC ON
GO
عبارات اس کیوال ما در اینجا قرار خواهند گرفت
GO
SET NOEXEC OFF
GO
```

مطابق راهنمای اس کیوال سرور، SET NOEXEC سبب کامپایل کوئری‌های پس از آن شده ولی آن‌هارا اجرا نخواهد کرد. همچنین SET PARSEONLY ON فقط syntax یک کوئری را بررسی کرده و خطاهای حاصل را نمایش می‌دهد (بدون اجرا کوئری، بدون کامپایل آن).

نظرات خوانندگان

نویسنده: Alex's Blog
تاریخ: ۱۳۸۷/۱۲/۱۲ ۰۷:۲۷:۰۰

این مورد خیلی بدردبخوره. ممنون

عنوان: **تعیین اعتبار کردن یک عبارت SQL - قسمت دوم**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۱۲ ۱۲:۳۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

مطلبی را روز قبل نوشتم در مورد تعیین اعتبار یک کوئری. این مورد از آنجایی حائز اهمیت می‌شود که برای مثال تغییری در ساختار یکی از جداول حاصل شود. اکنون می‌خواهیم بررسی کنیم آیا سیستم از کار افتاده یا نه!؟
شما می‌توانید نام یک فیلد را تغییر دهید (حتی اگر این فیلد در یک رویه ذخیره شده استفاده شده باشد) و هیچ خطایی هم نخواهید گرفت و این منشاء دردسرهای زیادی خواهد بود.
در حالت استفاده از SET NOEXEC ON ، کوئری مورد نظر فقط کامپایل می‌شود و همچنین از لحاظ نحوی بررسی خواهد شد، اما این کافی نیست.
مثال زیر را در نظر بگیرید:

```
Create PROCEDURE Test1
AS
SELECT * FROM tblPIDs1
```

جدول tblPIDs1 در دیتابیس مورد نظر وجود ندارد.
این کوئری قابل اجرا است. دکمه‌ی F5 را فشار دهید، بلافاصله رویه ذخیره شده‌ی Test1 برای شما ایجاد خواهد شد.
سپس کوئری زیر را اجرا کنید:

```
USE testdb
SET NOEXEC ON;
exec test1 ;
SET NOEXEC OFF;
```

بدون مشکل و بروز خطایی، پیغام زیر را نشان می‌دهد:

```
Command(s) completed successfully
```

ایرادی هم وارد نیست چون فقط عملیات parsing و compile صورت گرفته و نه اجرای واقعی رویه ذخیره شده. اینجا از لحاظ دستوری مشکلی وجود ندارد.

در این نوع موارد می‌توان از SET FMTONLY ON استفاده کرد. این مورد اجرای غیر واقعی یک کوئری را سبب می‌شود (تاثیری روی دیتابیس موجود نخواهد داشت، برای مثال اگر در رویه ذخیره شما عبارت insert وجود داشت، دیتایی insert نخواهد شد) و تنها متادیتای حاصل را بازگشت می‌دهد. مثلا نام ستون‌های یک کوئری را و همچنین در این حین اگر خطایی رخ داده باشد، آن‌را نیز ارائه خواهد داد.

```
USE testdb
SET FMTONLY ON;
exec test1 ;
SET FMTONLY OFF;
```

با اجرای کوئری فوق خطای زیر ظاهر می‌شود:

```
Msg 208, Level 16, State 1, Procedure test1, Line 3
Invalid object name 'tblPIDs1'.
```

برای اتوماسیون این توانایی می‌توان از کوئری زیر استفاده کرد:

```
USE  testdb;

SET NOCOUNT ON;

DECLARE @name NVARCHAR(MAX),
       @sql NVARCHAR(MAX),
       @type CHAR(2), -- object type
       @type_desc NVARCHAR(60), -- object type description
       @params NVARCHAR(MAX) -- parameters

DECLARE @tblInvalid TABLE (
           -- invalid objects
           [type_desc] NVARCHAR(60),
           [name] NVARCHAR(MAX),
           [error_number] INT,
           [error_message] NVARCHAR(MAX),
           [type] CHAR(2)
       );

DECLARE testSPs CURSOR FAST_FORWARD
FOR
   SELECT [name] = OBJECT_NAME(SM.[object_id]),
          [type] = SO.[type],
          SO.[type_desc],
          [params] = (
              SELECT (
                          SELECT CONVERT(
                                     XML,
                                     (
                                         SELECT STUFF(
                                                     (
                                                         SELECT ', ' + [name] +
                                                                '=NULL' AS
                                                                [text()]
                                                         FROM   sys.parameters
                                                         WHERE  [object_id] = SM.[object_id]
                                                                FOR XML PATH('')
                                                     ),
                                                     1,
                                                     1,
                                                     ''
                                                 )
                                     )
                                 )
                                 FOR XML RAW,
                                 TYPE
                      ).value('/row[1]', 'varchar(max)')
          )
   FROM   sys.sql_modules SM
          JOIN sys.objects SO
               ON  SO.[object_id] = SM.[object_id]
   WHERE  SO.[is_ms_shipped] = 0
          AND SO.[type] = 'P'


OPEN testSPs
FETCH NEXT FROM testSPs INTO @name, @type, @type_desc, @params

WHILE (@@FETCH_STATUS = 0)
BEGIN
   BEGIN TRY
       SET @sql = 'SET FMTONLY ON; exec ' + @name + ' ' + @params +
           '; SET FMTONLY OFF;'
       --PRINT @sql;
       EXEC (@sql) ;
   END TRY
   BEGIN CATCH
       PRINT @type_desc + ', ' + @name + ', Error: ' + CAST(ERROR_NUMBER() AS VARCHAR)
       + ', ' + ERROR_MESSAGE();
       INSERT INTO @tblInvalid
       SELECT @type_desc,
              @name,
              ERROR_NUMBER(),
              ERROR_MESSAGE(),
              @type

       ;
   END CATCH
```

```
   FETCH NEXT FROM testSPs INTO @name, @type, @type_desc, @params
END
CLOSE testSPs
DEALLOCATE testSPs


SELECT [type_desc],
      [name],
      [error_number],
      [error_message]
FROM   @tblInvalid
ORDER BY
      CHARINDEX([type], ' U V PK UQ F TR FN TF P SQ '),
     [name];
```

توضیحات:

این کوئری، در دیتابیس جاری که در قسمت use dbname مشخص می‌شود، تمامی رویه‌های ذخیره شده را به صورت خودکار پیدا می‌کند. سپس لیست آرگومان‌های آن‌ها را نیز یافته و عبارت exec مربوطه را تشکیل می‌دهد. سپس با استفاده از SET FMTONLY ON سعی در شبیه سازی اجرای تک تک رویه‌های ذخیره شده می‌کند. اگر خطایی در این بین رخ داد، آن‌ها را در یک جدول موقتی ذخیره کرده و در آخر نتیجه را نمایش می‌دهد.

ارزش این کوئری زمانی مشخص می‌شود که تعداد زیادی رویه ذخیره شده داشته باشید اما نمی‌دانید کدامیک از آن‌ها بر اساس آخرین تغییرات صورت گرفته، هنوز معتبر هستند یا نه. آیا به قول معروف، سیستم اومد پایین یا خیر!؟

نکته:

قسمتی که از XML استفاده شده جهت concatenating نتیجه حاصل از کوئری، مورد استفاده قرار گرفته و این روزها بحث رایجی است که در بسیاری از سایت‌ها در مورد آن می‌توان مطالب مفیدی را یافت. راه دیگر انجام آن استفاده از COALESCE می‌باشد.

مآخذ:

Check Validity of SQL Server Stored Procedures

Which of your Stored Procedures are no longer Valid

SET FMTONLY ON

نظرات خوانندگان

نویسنده: Alex's Blog
تاریخ: ۱۳۸۷/۱۲/۱۲ ۱۴:۱۲:۰۰

در مورد مطلب دیروزتون که فکر می کردم دنبال موردی بودم که امروز بهش اشاره کردین
منظورم SET FMTONLY ON بود. این مورد بیشتر بکار میاد البته بنظر من.
ممنون

عنوان: **نگهداری ایندکس‌ها در اس‌کیوال سرور**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۲/۲۱ ۰۹:۵۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

پس از مدتی که از شروع به کار یک سیستم می‌گذرد، همانطور که تعریف ایندکس‌های مفید سرعت جستجوها را بالا می‌برد، ایجاد fragmentation در آن‌ها نیز تاثیر منفی در کارآیی خواهد داشت. به همین منظور نیاز است هر از چندگاهی بررسی شود میزان fragmentation ایندکس‌ها چقدر است. اگر این میزان بیش از 30 درصد بود توصیه شده است که از دستور DBCC INDEXDEFRAG استفاده شود یا بازسازی مجدد (rebuild) ایندکس‌ها صورت گیرد.

یکی دیگر از امکانات dmv های اس کیوال سرورهای 2005 به بعد، ارائه آمار میزان fragmentation ایندکس‌ها است که کوئری آن به صورت زیر می‌تواند باشد:

```
USE dbName;
SELECT OBJECT_NAME(DMV.object_id) AS TABLE_NAME,
      SI.NAME AS INDEX_NAME,
      avg_fragmentation_in_percent AS FRAGMENT_PERCENT,
      DMV.record_count
FROM   sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, 'SAMPLED') AS
      DMV
      LEFT OUTER JOIN SYS.INDEXES AS SI
           ON  DMV.OBJECT_ID = SI.OBJECT_ID
           AND DMV.INDEX_ID = SI.INDEX_ID
WHERE  avg_fragmentation_in_percent > 10
      AND index_type_desc IN ('CLUSTERED INDEX', 'NONCLUSTERED INDEX')
      AND DMV.record_count >= 2000
ORDER BY
      TABLE_NAME DESC
```

باید در نظر داشت که اجرای این کوئری بر روی یک دیتابیس حجیم زمان‌بر بوده و احتمالا عملکرد سیستم را تحت تاثیر قرار می‌دهد. بنابراین استفاده از آن در خارج از ساعات کاری باید مد نظر باشد. بازسازی ایندکس‌ها نیز به همین صورت است.

برای بازسازی تمامی ایندکس‌های یک دیتابیس مفروض می‌توان از کوئری زیر استفاده کرد:

```
DECLARE @TableName VARCHAR(255)
DECLARE @sql NVARCHAR(500)
DECLARE @fillfactor INT
SET @fillfactor = 80
DECLARE TableCursor CURSOR
FOR
   SELECT OBJECT_SCHEMA_NAME([object_id]) + '.' + NAME AS TableName
   FROM   sys.tables

OPEN TableCursor
FETCH NEXT FROM TableCursor INTO @TableName
WHILE @@FETCH_STATUS = 0
BEGIN
   SET @sql = 'ALTER INDEX ALL ON ' + @TableName +
       ' REBUILD WITH (FILLFACTOR = ' + CONVERT(VARCHAR(3), @fillfactor) + ')'

   EXEC (@sql)
   FETCH NEXT FROM TableCursor INTO @TableName
END
CLOSE TableCursor
DEALLOCATE TableCursor
```

عنوان: **حذف سریع تمام رکوردها در SQL server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۱۰ ۱۵:۴۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

فرض کنید یک دیتابیس آزمایشی دارید که می‌خواهید تمام رکوردهای آن‌را حذف کنید. اگر در این دیتابیس انواع و اقسام کلیدهای خارجی و تریگر و امثال آن وجود داشته باشند، صرفا با یک دستور delete ساده کار به پایان نمی‌رسد و موفق به حذف رکوردها نخواهید شد (چون این قید و بندها به همین جهت طراحی شده‌اند تا یکپارچگی دیتابیس حفظ شود).
اما اگر واقعا این قیود در این لحظه مهم نبودند و نیاز بود تا تمام رکوردها را حذف کنیم، سریعترین راه حل موجود چیست؟

```
--Disable Constraints & Triggers
exec sp_MSforeachtable 'ALTER TABLE ? NOCHECK CONSTRAINT ALL'
exec sp_MSforeachtable 'ALTER TABLE ? DISABLE TRIGGER ALL'
--Perform delete operation on all table for cleanup
exec sp_MSforeachtable 'DELETE ?'
--Enable Constraints & Triggers again
exec sp_MSforeachtable 'ALTER TABLE ? CHECK CONSTRAINT ALL'
exec sp_MSforeachtable 'ALTER TABLE ? ENABLE TRIGGER ALL'
--Reset Identity on tables with identity column
exec sp_MSforeachtable 'IF OBJECTPROPERTY(OBJECT_ID(''?''), ''TableHasIdentity'') = 1 BEGIN DBCC
CHECKIDENT (''?'',RESEED,0) END'
```

اسکریپت فوق تمامی رکوردهای دیتابیس جاری را حذف کرده و همچنین فیلدهای identity را نیز مجددا به حالت اول باز می‌گرداند.

توضیحات:
sp_Msforeachtable یکی از رویه‌های ذخیره شده‌ی سیستمی اس کیوال سرور می‌باشد که مستند نشده است. اگر نیاز داشتید کدی را بدون نوشتن یک کرسر و امثال آن، بر روی تمامی جداول اجرا کنید می‌توان از آن استفاده نمود.
امضای این رویه ذخیره شده به صورت زیر است:

```
exec @RETURN_VALUE=sp_MSforeachtable @command1, @replacechar, @command2,
 @command3, @whereand, @precommand, @postcommand
```

که در آن:
RETURN_VALUE مقدار بازگشتی است.
Command1 اولین دستوری است که اجرا خواهد شد (به همین ترتیب سپس Command2 و بعد از آن Command3 اجرا خواهد گردید)
Replacechar کاراکتری است که در دستور T-SQL مورد نظر جایگزین نام جدول خواهد شد. مقدار پیش فرض آن ? است.
Precommand پیش از پردازش عملیات روی هر جدولی اجرا می‌شود.
Postcommand پس از اجرای کلیه دستورات روی تمامی جداول، اجرا خواهد شد.

چند مثال:
نمایش تمامی جداول دیتابیس جاری

```
EXEC sp_MSforeachtable "print '?'"
```

نمایش اندازه‌ی جداول یک دیتابیس

```
EXEC sp_msforeachtable 'sp_spaceused ''?'''
```

نظرات خوانندگان

نویسنده: مهدی
تاریخ: ۱۳۸۸/۰۵/۲۳ ۲۰:۵۷:۵۷

ممنون، مطلب خوبی بود.

عنوان: **مثال‌هایی در مورد استفاده از SQL Server 2005 Reporting Services**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۱۶ ۱۷:۲۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

SQL Server 2005 Reporting Services راه حلی مبتنی بر سرور جهت گزارشگیری‌های جامع با قابلیت ارائه گزارش‌های paper-based ، ad hoc و Web-based است. (جهت آشنایی بیشتر می‌توان به مقاله " SQL Server Reporting Service چیست؟ " آقای رمضانی مراجعه کرد)

اخیرا مجموعه‌ای مثال در مورد کار با Reporting Services ارائه شده است که از آدرس زیر قابل دریافت است:

SQL Server 2005 Report Packs

این مجموعه شامل 7 بسته آموزشی و مثال‌های مرتبط می‌باشد:

```
SQL Server 2005 Integration Services Log Reports
SQL Server 2005 Report Pack for Microsoft Dynamics Axapta 3.0
SQL Server 2005 Report Pack for Microsoft Dynamics Navision 4.0
SQL Server 2005 Report Pack for Microsoft Dynamics Great Plains 8.0
SQL Server 2005 Report Pack for Microsoft Dynamics Great Plains 9.0
SQL Server 2005 Report Pack for Microsoft Internet Information Services (IIS)
SQL Server 2005 Report Pack for Financial Reporting
```

نظرات خوانندگان

نویسنده: Alex's Blog
تاریخ: ۱۳۸۸/۰۱/۱۷ ۰۸:۳۵:۰۰

سلام
امروز بعد از 16 روز تعطیلی و نبود اینترنت ، به وبلاگتون که سر زدم بمباران اطلاعاتی شدم :)
ممنون.

عنوان: **بررسی صحت پشتیبان‌های تهیه شده در SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۱۸ ۱۸:۳۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

اولین و اساسی‌ترین قدم در نگهداری یک سیستم مبتنی بر داده، تهیه پشتیبان‌های منظم و همچنین قابل اطمینان می‌باشد.
دستور T-SQL زیر بدون ری‌استور کردن یک فایل بک آپ اس کیوال سرور، سعی در تعیین اعتبار آن می‌کند:

```
RESTORE VERIFYONLY
FROM DISK = 'C:\SQL_Backup\Test1'
WITH FILE = 1,
LOADHISTORY
```

این دستور
وضعیت کامل بودن پشتیبان و همچنین قابل خواندن بودن اطلاعات آن‌را بررسی می‌کند و در صورت سالم بودن بک آپ، پیغام زیر را نمایش خواهد داد:

```
The backup set on file 1 is valid.
```

عموما عملیات تهیه پشتیبان در یک مکان مشخص در سرور صورت می‌گیرد (خصوصا اگر یک job مختص به آن تعریف شده باشد که این‌کار را به صورت خودکار انجام دهد). بنابراین می‌توان عملیات اعتبار سنجی فوق را مکانیزه کرد. اسکریپت زیر مسیر آخرین بک آپ‌های گرفته شده در سرور را بر می‌گرداند:

```
SELECT DISTINCT physical_device_name
FROM   msdb.dbo.backupmediafamily
ORDER BY
      physical_device_name
```

اکنون می‌توان مسیرهای فوق را در یک cursor جهت بررسی صحت تک تک موارد استفاده نمود:

```
DECLARE @path NVARCHAR(1000),
       @msg NVARCHAR(MAX),
       @NewLine CHAR(2),
       @sql NVARCHAR(2000)

SET @NewLine = CHAR(13) + CHAR(10)
SET @msg = ''

DECLARE DATABASES_CURSOR CURSOR
FOR
   SELECT DISTINCT physical_device_name
   FROM   msdb.dbo.backupmediafamily
   ORDER BY
          physical_device_name

OPEN DATABASES_CURSOR

FETCH NEXT FROM DATABASES_CURSOR INTO @path

WHILE @@FETCH_STATUS = 0
BEGIN
   PRINT 'Verifying: ' + @path
   SET @sql = 'RESTORE VERIFYONLY FROM DISK = ''' + @path
       + ''' WITH FILE = 1, LOADHISTORY'

   EXEC sp_executesql @sql
   IF @@ERROR <> 0
   BEGIN
       SET @msg = @msg + 'Failed to verify: ' + @path + @NewLine
   END

   FETCH NEXT FROM DATABASES_CURSOR INTO @path
END
```

```
CLOSE DATABASES_CURSOR
DEALLOCATE DATABASES_CURSOR

IF @msg <> ''
BEGIN
   PRINT @msg
   -- send email
   EXECUTE msdb.dbo.sp_send_dbmail
   @recipients = 'nasiri@site.net', -- Change This
   @copy_recipients = 'Administrator@site.net', -- Change This
   @Subject = 'backup verification info.',
   @Body = @msg
   ,@importance = 'High'

END
```

اسکریپت فوق بر روی تمامی مسیرهای ثبت شده موجود که در آن‌ها پیشتر پشتیبان تهیه شده است، دستور RESTORE VERIFYONLY را اجرا می‌کند و در آخر اگر پیغامی حاصل شد، یعنی مشکلی پدید آمده و ایمیلی را به اشخاص مورد نظر ارسال می‌کند.

می‌توان بر روی این اسکریپت یک job تهیه کرد که هر روز پس از تهیه بک آپ خودکار، کار بررسی صحت عملیات را نیز انجام دهد.

عنوان: **سرویس پک یک SQL Server 2008**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۲۰ ۰۰:۳۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

نگارش نهایی سرویس پک یک SQL Server 2008 ارائه شد. (به بیان دیگر اکنون می‌توانید با خیال راحت از آن در محیط کاری استفاده کنید!)

دریافت

توضیحات بیشتری در مورد این نگارش

لیست باگ‌های فیکس شده در سرویس پک یک

این به روز رسانی به همراه Report Builder 2.0 و Feature Pack 2009 نیز می‌باشد.

عنوان: **یافتن لیست اسمبلی‌های ارجاعی**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۲۵ ۲۱:۱۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

اگر برنامه‌ی شما برای مثال با SMO مربوط به اس کیوال سرور 2008 کامپایل شود، روی سروری با SQL Server 2005 کار نخواهد کرد و پیغام می‌دهد که نگارش 10 اسمبلی Microsoft.SqlServer.Management.Sdk.Sfc یافت نشد.

یک راه حل آن، نصب Microsoft SQL Server 2008 Management Objects بر روی سرور است، یا راه حل دوم، پیدا کردن اسمبلی‌هایی که برنامه به آن‌ها ارجاع دارد و کپی کردن آن‌ها کنار فایل اجرایی برنامه در سرور. (درست کردن یک برنامه پرتابل دات نتی، یا نسبتا پرتابل!)

برای این منظور کلاس زیر تهیه شده است که مسیر فایل اجرایی یا dll یک پروژه را دریافت کرده و لیست تمام ارجاعات به آن‌را به صورت بازگشتی پیدا می‌کند. (البته در قسمت یافتن مسیر اسمبلی‌ها، اسمبلی‌های سیستمی که با خود دات نت فریم ورک نصب می‌شوند، حذف شده است)

```
using System.Collections.Generic;
using System.Reflection;
using System.IO;

namespace App
{
   class CFindRef
   {
       #region Fields (2)

       /// <summary>
       /// لیستی جهت نگهداری نام اسمبلی‌ها
       /// </summary>
       private readonly List<string> _assemblies = new List<string>();
       /// <summary>
       /// لیستی جهت نگهداری مسیر اسمبلی‌ها
       /// </summary>
       private readonly List<string> _filePath = new List<string>();

       #endregion Fields

       #region Constructors (1)

       /// <summary>
       /// سازنده کلاس
       /// </summary>
       /// <param name="fileName">مسیر اولیه اسمبلی مورد نظر</param>
       public CFindRef(string fileName)
       {
           ListReferences(fileName);
       }

       #endregion Constructors

       #region Properties (2)

       /// <summary>
       /// لیست مسیر اسمبلی‌هایی که به آن‌ها ارجاعی وجود دارد منهای موارد سیستمی
       /// </summary>
       public List<string> ReferencedFiles
       {
           get
           {
               _filePath.Sort();
               return _filePath;
           }
       }

       /// <summary>
       /// لیست کامل اسمبلی‌هایی که اسمبلی ما به آن‌ها وابسته است
       /// </summary>
       public List<string> ReferencedNames
       {
           get
```

```
            {
                _assemblies.Sort();
                return _assemblies;
            }
        }

        #endregion Properties

        #region Methods (1)

        // Private Methods (1)

        /// <summary>
        /// متدی بازگشتی جهت یافتن لیست ارجاعات
        /// </summary>
        /// <param name="path">مسیر یا نام اسمبلی</param>
        private void ListReferences(string path)
        {
            //آیا تکراری است؟
            if (_assemblies.Contains(path))
                return;

            Assembly asm;
            // آیا فایل است یا نام کامل اسمبلی
            if (File.Exists(path))
            {
                // load the assembly from a path
                asm = Assembly.LoadFrom(path);
            }
            else
            {
                // سعی در بارگذاری اسمبلی
                try
                {
                    asm = Assembly.Load(path);
                }
                catch
                {
                    asm = null; //جای بهبود دارد
                }
            }

            if (asm == null) return;

            // افزودن به لیست نام‌ها
            _assemblies.Add(path);
            string asmLocation = asm.Location;
            //حذف موارد سیستمی از لیست مسیر فایل‌ها
            if (asmLocation != null && !asmLocation.Contains("\\System.")
                && !asmLocation.Contains("\\mscorlib"))
                _filePath.Add(asmLocation);

            // پیدا کردن ارجاع‌ها
            AssemblyName[] imports = asm.GetReferencedAssemblies();
            // iterate
            foreach (AssemblyName asmName in imports)
            {
                // فراخوانی بازگشتی جهت یافتن تمامی ارجاعات
                ListReferences(asmName.FullName);
            }
        }

        #endregion Methods
    }
}
```

مثالی در مورد نحوه‌ی استفاده از آن:

```
CFindRef cfr = new CFindRef(@"C:\App\test.exe");
foreach (var asmRef in cfr.ReferencedFiles)
{
 textBox1.Text += asmRef + Environment.NewLine;
 //Application.DoEvents();
}
```

برای نمونه، برنامه‌ای که از SMO مربوط به اس کیوال سرور 2008 استفاده می‌کند، این لیست ارجاعات را دارد:

```
C:\WINDOWS\assembly\GAC_MSIL\Microsoft.SqlServer.ConnectionInfo\10.0.0.0__89845dcd8080cc91\Microsoft.Sq
lServer.ConnectionInfo.dll
C:\WINDOWS\assembly\GAC_MSIL\Microsoft.SqlServer.Dmf\10.0.0.0__89845dcd8080cc91\Microsoft.SqlServer.Dmf
.dll
C:\WINDOWS\assembly\GAC_MSIL\Microsoft.SqlServer.Management.Sdk.Sfc\10.0.0.0__89845dcd8080cc91\Microsof
t.SqlServer.Management.Sdk.Sfc.dll
C:\WINDOWS\assembly\GAC_MSIL\Microsoft.SqlServer.ServiceBrokerEnum\10.0.0.0__89845dcd8080cc91\Microsoft
.SqlServer.ServiceBrokerEnum.dll
C:\WINDOWS\assembly\GAC_MSIL\Microsoft.SqlServer.Smo\10.0.0.0__89845dcd8080cc91\Microsoft.SqlServer.Smo
.dll
C:\WINDOWS\assembly\GAC_MSIL\Microsoft.SqlServer.SqlClrProvider\10.0.0.0__89845dcd8080cc91\Microsoft.Sq
lServer.SqlClrProvider.dll
C:\WINDOWS\assembly\GAC_MSIL\Microsoft.SqlServer.SqlEnum\10.0.0.0__89845dcd8080cc91\Microsoft.SqlServer
.SqlEnum.dll
```

با کپی کردن همین فایل‌ها کنار فایل اجرایی برنامه‌ای که قرار است روی سروری با SQL Server 2005 کار کند، مشکل برطرف می‌شود و برنامه بدون مشکل کار خواهد کرد.

برنامه آماده هم در این زمینه موجود است، برای مثال CheckAsm

مشاهده سایت

نظرات خوانندگان

نویسنده: Sirasad
تاریخ: ۱۳۸۸/۰۱/۲۵ ۲۲:۳۷:۰۰

ببخشید آقای نصیری که این رو اینجا مطرح کردم. من به سفارش شما تو شرکت از SVN استفاده کردم . بدین نحو که اول خود SVN رو نصب کردم بعد TortoiseSVN . بعدش هم که .برنامه رو سرور مثل سعت کار میکنه , ولی رو کلاینت ها مورد داره . پروژه رو هم رو سرور share کردم و کلاینت هم اون رو Map کردند . حالا وقتی Solution رو باز می کنیم برای هر پروژه این warning رو میده: Project Location Is not Trusted .

وقتی هم که روی یک button کلیک میکنیم این error رو میده : Soure file f:\solution\project1 does not belong to the project being debogged.

وقتی هم میخاییم solution رو commit کنیم error میده که :

Unable to open an ra_local url.unable to open repository.

البته خیلی من رو ببخشید که اندازه یه تاپیک پست دادم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۲۵ ۲۳:۴۹:۰۰

سلام

نیازی نبود سورسی را Map کنید یا اصلا نباید این‌کار را می‌کردید. (اگر منظور مپ کردن سورس بوده نه فایل اجرایی برنامه)

SVN وظیفه مدیریت و به اشتراک گذاشتن پروژه رو داره، نه شبکه ویندوزی یا لینوکسی. (حتما از visual svn server استفاده کنید تا این موارد را برای شما ساده کند)

کلاینت‌ها هر کدام نسخه‌ی کامل و لوکال خودشون رو باید داشته باشند (از طریق check out مخزن کد این پروژه لوکال باید تشکیل شود نه کپی دستی). سپس مثل اینکه دارند لوکال کار می‌کنند (نه از روی شبکه در حالت مپ شده). کاملا حالت معمولی و قطع از شبکه. SVN برای مدیریت پروژه روی اینترنت هم بکار میره. نه چیزی map میشه و نه لازم هست کاربر همیشه به شبکه وصل باشه.

نسخه کد شما که روی سرور هست و توسط SVN نگهداری می‌شود، مخزن اصلی است که تغییرات با آن هماهنگ می‌شود. برای انتقال کد به مخزن، باید عملیات check in صورت گیرد.

بعد هر کدام از اعضای تیم زمانیکه check out می‌کنند ، یک نسخه‌ی محلی دریافت می‌کنند و این فولدر تحت کنترل SVN قرار میگیره، حالا مباحث update و commit و غیره کار می‌کند. فقط هر بار که می‌خواهند commit کنند باید اول update کنند ببینند کسی چیزی را تغییر داده، تصادمی هست یا نه؟ بعد commit کنند به سرور. (یعنی ارتباط با شبکه فقط در همین چند لحظه کوتاه است و بسیار سریع هم خواهد بود)

فصل دوم کتابچه‌ای را که من تهیه کردم لطفا مطالعه کنید، گردش کاری آن توضیح داده شده است.

http://www.dotnettips.info/2008/10/subversion.html

در فصل یک هم توضیح دادم که چه پورتی را باید روی سرور باز کنید تا SVN توسط فایروال بلاک نشود.

حتما توصیه می‌کنم اگر با VS.Net کار می‌کنید افزونه Visual SVN را نصب کنید تا راحت و بدون دردسر کار کنید .

خطایی رو که توضیح دادید مربوط هست به اشتراک گذاشتن فایل اجرایی یک برنامه دات نتی روی شبکه که این خطا رو می‌گیرید:

Project Location Is not Trusted

این خطا رو با دادن full trust به برنامه می‌تونید حل کنید که اینجا قدم به قدم توضیح داده شده:

http://msdn.microsoft.com/en-us/library/bs2bkwxc(VS.80).aspx

همچنین کامنت‌های اون رو لطفا بخونید. مثلا ممکن هست فایل بلاک شده باشه که با کلیک راست و unblock کردن مشکل حل میشه.

خطای زیر بیشتر مربوط به حالتی است که الف) هنوز مخزن کد ایجاد نشده و ب) عملیات initial import یا check in صورت نگرفته

Unable to open an ra_local url. unable to open repository.

حتما کتابچه فوق را مطالعه نمایید.

نویسنده: SirAsad
تاریخ: ۱۳۸۸/۰۱/۲۶ ۰۹:۰۰:۰۰

خیلی ممنون آقای نصیری ... ولی یه قضیه برام حل نشد . شما گفتید که: "کلاینت‌ها هر کدام نسخه‌ی کامل و لوکال خودشون رو باید داشته باشند (از طریق check out مخزن کد این پروژه لوکال باید تشکیل شود نه کپی دستی). سپس مثل اینکه دارند لوکال کار می‌کنند (نه از روی شبکه در حالت مپ شده). "
چجوری باید بدون اینکه اونا سرور رو مپ کنند و یا اینکه فایل ها رو خودشون کپی کنند به سورس ها دسترسی داشته باشند ؟

من الان دارم Viusal SVN Server رو دانلود میکنم , اما قبل این هم از Ankh-SVN-2.0.6347 استفاده میکردم که کار هم باهاش بسیار آسان بود ... اگر لطفی بکنید و در مورد ساخت نسخه مخصوص کلاینت ها یه توضیح بدید ممنون میشم .

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۲۶ ۱۱:۱۴:۰۰

اولا قرار نیست همه به سورس‌ها دسترسی داشته باشند. هر شخصی به پروژه‌ای که شما بهش دسترسی خواهید داد باید دسترسی داشته باشد. مدیریت اینکار با visual svn server و یکپارچه بودنش با سیستم windows authentication بسیار ساده و در حد چند کلیک است.
یک repository در این سرور درست می‌کنید. بعد نسخه‌ی اولیه پروژه در آن check in و import می شود. حالا سرور راه اندازی شده و آماده سرویس دهی است.
ثانیا توزیع کد از طریق svn انجام میشه. کلاینت‌ها مثلا با استفاده از TortoiseSVN یا افزونه‌ای که شما نصب می‌کنید با سرور ارتباط خواهند داشت و همین. نیازی نیست چیزی مپ بشه.
Viusal SVN Server را با افزونه‌ای مثل Ankh-SVN اشتباه نگیرید. Ankh-SVN یک کلاینت SVN است و به خودی خود کاری ازش ساخته نیست. این افزونه به سرور وصل می‌شود اطلاعات جدید را می‌گیرد یا اطلاعات موجود را با سرور هماهنگ می‌کند.
برای توضیحات بیشتر لطفا کتابچه ذکر شده را مطالعه کنید. کمی تست کنید بعد پیاده سازی
http://www.dotnettips.info/2008/10/subversion.html

نویسنده: Sirasad
تاریخ: ۱۳۸۸/۰۱/۲۷ ۲۲:۴۳:۰۰

آقای خیلی ممنون . آقا مشکل ما حل شد .

دست گل تون درد نکنه

عنوان: **اس کیوال سرور 2008 و عملگرهای C مانند**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۳۱ ۲۱:۳۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

اگر با زبان C و مشتقات آن آشنایی داشته باشید، حتما با عملگرهای ترکیبی آن‌ها که جهت خلاصه نویسی بکار می‌روند، نیز کار کرده‌اید. برای مثال:

```
int i =5;
i += 15; // i = i + 15;
```

اس کیوال سرور 2008 نیز از اینگونه عملگرها پشتیبانی به عمل می‌آورد . برای نمونه:

```
DECLARE @x1 int = 27;
SET @x1 += 2 ;
SELECT @x1 AS Added_2;
```

در دستورات T-SQL فوق دو نکته قابل توجه است:

الف) امکان تعریف و مقدار دهی همزمان یک متغیر (مقدار دهی همزمان با تعریف، تا قبل از اس کیوال سرور 2008 پشتیبانی نمی‌شد)

ب) امکان استفاده از عملگرهای C مانند در عبارات T-SQL

لیست این عملگرهای جدید به شرح زیر است:

```
+= (Add EQUALS) (Transact-SQL)
-= (Subtract EQUALS) (Transact-SQL)
*= (Multiply EQUALS) (Transact-SQL)
/= (Divide EQUALS) (Transact-SQL)
%= (Modulo EQUALS) (Transact-SQL)
&= (Bitwise AND EQUALS) (Transact-SQL)
^= (Bitwise Exclusive OR EQUALS) (Transact-SQL)
|= (Bitwise OR EQUALS) (Transact-SQL)
```

عنوان: **مقایسه ساختاری دو دیتابیس SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۰۱ ۱۹:۰۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

یکی از مواردی که در محیط کاری زیاد پیش می‌آید بحث همگام نبودن دیتابیس توسعه با دیتابیس کاری است.
منظور از دیتابیس توسعه، همان دیتابیسی است که برای برنامه نویسی و آزمایش از آن استفاده می‌شود و دیتابیس کاری هم مشخص است (برای مثال بر روی یک سرور در اینترانت داخلی یک شرکت و یا بر روی یک سرور اینترنتی قرار دارد). عادت‌های مختلفی هم این‌جا ممکن است وجود داشته باشد، برای مثال تغییرات جدید بر روی دیتابیس کاری اعمال شود و سپس فراموش شود که همان‌ها نیز باید به دیتابیس توسعه هم اعمال شوند تا در تغییرات بعدی برای آزمایش دچار مشکل نشویم و برعکس. بعد از یک مدت هم تبدیل به کابوس می‌شود؛ نمی‌دانیم الان دیتابیس کاری جدیدتر است یا دیتابیس توسعه؛ و یا اینکه کلا دو دیتابیس مفروض چه تفاوت‌های ساختاری با هم دارند (بدیهی است بحث دیتا در اینجا در درجه‌ی اول اهمیت قرار ندارد). فرصت این هم وجود ندارد که تک تک جداول، ویووها، رویه‌های ذخیره شده و خلاصه تمامی اشیاء مرتبط را بررسی کنیم که چه اختلافی با هم دارند. اینجا مستندات هم کمکی نخواهند کرد چون صحبت از یک جدول با 5 فیلد در میان نیست که موارد را سریع و به صورت دستی تطابق دهیم. همچنین این مشکل عموما زمانی رخ می‌دهد که یکی از دو طرف در حال حاضر مستندات کامل و به روزی ندارد. اکنون چه باید کرد؟
اولین فکری که به ذهن خطور می‌کند مراجعه به ابزارهای جانبی است (مثلا Red Gate's SQL Compare چند صد دلاری) غافل از اینکه خود Visual studio 2008 (نگارش‌های تیمی و دیتابیسی) این قابلیت را نیز ارائه می‌دهد (شکل زیر).

پس از انتخاب new schema comparison ، در صفحه‌ای که ظاهر می‌شود، بر روی new connection کلیک کرده و دیتابیس‌های مبداء و مقصد را جهت مقایسه ساختاری انتخاب نمائید و سپس بر روی دکمه Ok کلیک کنید.

اگر اس کیوال سرور 2008 را نصب کرده باشید، با پیغام زیر روبرو خواهید شد:

برای رفع این مشکل باید بسته به روز رسانی زیر را نصب کرد تا این نگارش نیز پشتیبانی شود:

Microsoft® Visual Studio Team System 2008 Database Edition GDR

(برای نصب حتما باید SP1 مربوط به VS.Net 2008 پیشتر نصب شده باشد)

پس از کلیک بر روی دکمه Ok، کار آنالیز دو دیتابیس شروع شده و تفاوت‌ها گزارش داده می‌شوند:

SchemaCompare1* | Start Page

The schema comparison is complete.

Status	vahid.testdb.dbo (Source Data...	Update Action	vahid.testdb2009.dbo (Target Database)
Equal	<unnamed>	Skip	<unnamed>
Tables			
Different definition	dbo.tblDates	Update	dbo.tblDates
Equal	dbo.tblFarsiTest	Skip	dbo.tblFarsiTest
Equal	dbo.tblMax	Skip	dbo.tblMax
Different definition	dbo.tblPIDs	Update	dbo.tblPIDs
Equal	dbo.tblTest	Skip	dbo.tblTest
Equal	dbo.tblTimes	Skip	dbo.tblTimes

Object Definitions

Source Object:

```
1 CREATE TABLE [dbo].[tblPIDs] (
2     [p_id] NVARCHAR (50) NULL
3 );
4
5
```

Target Object:

```
1 CREATE TABLE [dbo].[tblPIDs] (
2     [p_id] NVARCHAR (50) NULL,
3     [f1]   INT           NULL
4 );
5
6
```

New Text | Different Text | Missing Text

همچنین جهت سهولت کار، اسکریپت T-SQL ایی را نیز به نام schema update script تولید می‌کند که با اجرای آن به سادگی کار به روز رسانی دیتابیس مقصد صورت خواهد گرفت.

Schema Update Script - [vahid.testdb2009.dbo]

```
67
68 GO
69 ALTER TABLE [dbo].[tblPIDs] DROP COLUMN [f1];
70
71
72 GO
73 PRINT N'Refreshing dbo.crossJoin...';
74
```

در پایان یا می‌توان اسکریپت تولید شده را ذخیره کرد و در زمانی دلخواه اجرا و اعمال نمود و یا می‌توان بلافاصله بر روی دکمه write updates که در نوار ابزار ظاهر شده است کلیک کرد تا دیتابیس مقصد از لحاظ ساختاری با دیتابیس مبداء یکی شود.

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۲/۰۱ ۲۱:۳۹:۰۰

من هر روز مطالب شما را پیگیری می کنم.مثل همیشه عالی بود.

نویسنده: salarblog
تاریخ: ۱۳۸۸/۰۲/۰۲ ۰۱:۳۰:۰۰

بسیار نکته کاربردی بود
همانطور که گفتید از reg gate استفاده می کردم، غافل از این امکان!

نویسنده: مهدی یوسفی
تاریخ: ۱۳۸۸/۰۲/۰۲ ۰۸:۰۸:۰۰

من هم برنامه های زیادی رو تست کردم
برنامه رایگان در این زمینه وب سایت کلور کامپوننت هست:
http://www.clevercomponents.com/downloads/dbcomparer/dbcdownload.asp
البته این برنامه هم اشکالاتی دارد ولی به نظر من که نسخه تیم سوییت رو ندارم خوب بود.

نویسنده: reza
تاریخ: ۱۳۸۸/۰۲/۰۲ ۱۹:۴۷:۰۰

salam
man az in estedage mikonam
http://opendbiff.codeplex.com

نویسنده: مهدی
تاریخ: ۱۳۸۸/۰۲/۰۵ ۱۳:۱۶:۰۰

سلام
ولی من Redgate رو ترجیح میدم؛ چون کرکش دم دسته!

عنوان: **sp_send_dbmail و ارسال ایمیل فارسی**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۰۷ ۱۳:۵۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

نکته‌ی کوچکی در مورد ارسال ایمیل فارسی توسط رویه ذخیره شده سیستمی sp_send_dbmail اس کیوال سرور وجود دارد که شبیه به insert داده‌های فارسی در دیتابیسی است که پس از ثبت، به صورت ؟؟؟ ذخیره می‌شوند. (این مورد با تنظیم collation تقریبا قابل حل است)

اگر هنگام ثبت، collation عربی یا فارسی (در اس کیوال سرور 2008) انتخاب شود، مشکلی در ثبت نخواهد بود.

اگر به collation اهمیت نمی‌دهید باید اس کیوال سرور را مجبور کرد که داده را یونیکد ذخیره کند و اینکار با اضافه کردن یک N به ابتدای رشته صورت می‌گیرد و همچنین انتخاب نوع داده‌های n دار مانند nvarchar و امثال آن (n در اینجا به معنای national و اجبار آن می‌باشد):

```
Insert into tblTest(f1,f2) values(1,N'متن فارسی')
```

دقیقا همین نکته هم درباره‌ی ارسال ایمیل از طریق اس کیوال سرور صادق است. اگر N به ابتدای رشته اضافه نشود، رشته ارسالی را با فرمت ANSI ارسال می‌کند و داده‌های یونیکد متن تخریب خواهند شد؛ مثلا چیزی شبیه به حالت زیر:

```
<div align="center"><table border="1" width="95%" dir="rtl" cellspacing="0" cellpadding="0"
style="font-family: Tahoma; font-size: 8pt" bordercolor="#660066"><tr><td
bgcolor="#FFF9FF"><blockquote><p align="justify"><br>????? ????? ?<br>???? ???? ? ????? ?????? ??? ??
????? ????? ?????. ???? ??? ???? ???? ????? ????? ????? ????? ????? ??? ??? ???? ???? ?????? ? ???? ??
????? ???? ???? ??? ????? ??????.<br>???? ??? ????? ???? ?? ?????? ???? ????? ?????? ????? ? ?? ????
???????? ????? ???? ???? ???? ?????? ???? ???? ??? ?? ????? ????.</blockquote></td></tr></table></div>
```

این مشکل به صورت زیر قابل حل است:

```
DECLARE @msg NVARCHAR(max)
SET @msg=N'متن فارسی'
```

برای ردگیری وضعیت ایمیل‌های ارسالی هم می‌توان از کوئری‌های زیر استفاده نمود:

```
SELECT * from sysmail_allitems
SELECT * from sysmail_faileditems
SELECT * from sysmail_event_log
```

عنوان: **محدود کردن دسترسی به اس کیوال سرور بر اساس IP**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۱۰ ۲۱:۰۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

عموما محدود کردن دسترسی بر اساس IP بهتر است بر اساس راه حل‌هایی مانند فایروال ، IPSec و یا RRAS IP Filter صورت گیرد که جزو بهینه‌ترین و امن‌ترین راه حل‌های ممکن هستند.
در ادامه قصد داریم این محدودیت را با استفاده از امکانات خود اس کیوال سرور انجام دهیم (بلاک کردن کاربران بر اساس IP های غیرمجاز). مواردی که در ادامه ذکر خواهند شد در مورد اس کیوال سرور 2005 ، سرویس پک 2 به بعد و یا اس کیوال سرور 2008 صادق است.
اس کیوال سرور این قابلیت را دارد که می‌توان بر روی کلیه لاگین‌های صورت گرفته در سطح سرور تریگر تعریف کرد . به این صورت می‌توان تمامی لاگین‌ها را برای مثال لاگ کرد (جهت بررسی مسایل امنیتی) و یا می‌توان هر لاگینی را که صلاح ندانستیم rollback نمائیم (ایجاد محدودیت روی لاگین در سطح سرور).

لاگ کردن کلیه لاگین‌های صورت گرفته به سرور

ایجاد جدولی برای ذخیره سازی اطلاعات لاگین‌ها:

```
USE [master]
GO

SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[Logging](
 [id] [int] IDENTITY(1,1) NOT NULL,
 [LogonTime] [datetime] NULL,
 [LoginName] [nvarchar](max) NULL,
 [ClientHost] [varchar](50) NULL,
 [LoginType] [varchar](100) NULL,
 [AppName] [nvarchar](500) NULL,
 [FullLog] [xml] NULL,
CONSTRAINT [PK_IP_Log] PRIMARY KEY CLUSTERED
(
 [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON,
ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]

GO

SET ANSI_PADDING OFF
GO

ALTER TABLE [dbo].[Logging] ADD  CONSTRAINT [DF_IP_Log_LogonTime]  DEFAULT (getdate()) FOR [LogonTime]
GO
```

در ادامه یک تریگر لاگین را جهت ذخیره سازی اطلاعات کلیه لاگین‌ها به سرور ایجاد می‌نمائیم:

```
USE [master]
GO

CREATE TRIGGER LogonTrigger
ON ALL SERVER
FOR LOGON
```

```
AS
BEGIN
   DECLARE @data XML
   SET @data = EVENTDATA()

   INSERT INTO [Logging]
     (
       [LoginName],
       [ClientHost],
       [LoginType],
       [AppName],
       [FullLog]
     )
   VALUES
     (
       @data.value('(/EVENT_INSTANCE/LoginName)[1]', 'nvarchar(max)'),
       @data.value('(/EVENT_INSTANCE/ClientHost)[1]', 'varchar(50)'),
       @data.value('(/EVENT_INSTANCE/LoginType)[1]', 'varchar(100)'),
       APP_NAME(),
       @data
     )
END
```

اکنون برای مثال از آخرین 100 لاگین انجام شده، به صورت زیر می‌توان گزارشگیری کرد:

```
SELECT TOP 100 *  FROM [master].[dbo].[Logging] ORDER BY id desc
```

و بدیهی است در تریگر فوق می‌توان روی هر کدام از آیتم‌های دریافتی مانند ClientHost و غیره فیلتر ایجاد کرد و تنها موارد مورد نظر را ثبت نمود.

محدود کردن کاربران بر اساس IP

ClientHost ایی که در رخداد لاگین فوق بازگشت داده می‌شود همان IP کاربر راه دور است. برای فیلتر کردن IP های غیرمجاز، ابتدا در دیتابیس مستر یک جدول برای ذخیره سازی IP های مجاز ایجاد می‌کنیم و IP های کلیه کلاینت‌های معتبر خود را در آن وارد می‌کنیم:

```
USE [master]
GO
CREATE TABLE [IP_RESTRICTION](
 [ValidIP] [varchar](15) NOT NULL,
CONSTRAINT [PK_IP_RESTRICTION] PRIMARY KEY CLUSTERED
(
 [ValidIP] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON,
ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

سپس تریگر لاگین ما برای منع کاربران غیرمجاز بر اساس IP ، به صورت زیر خواهد بود:

```
USE [master]
GO

CREATE TRIGGER [LOGIN_IP_RESTRICTION]

ON ALL SERVER
FOR LOGON
AS
BEGIN
   DECLARE @host NVARCHAR(255);
   SET @host = EVENTDATA().value('(/EVENT_INSTANCE/ClientHost)[1]', 'nvarchar(max)');

   IF (
          NOT EXISTS(
              SELECT *
              FROM   MASTER.dbo.IP_RESTRICTION
```

```
            WHERE  ValidIP = @host
        )
    )
   BEGIN
        ROLLBACK;
   END
END;
```

اخطار مهم!

تریگر فوق خطرناک است! ممکن است خودتان هم دیگر نتوانید لاگین کنید!! (حتی با اکانت ادمین)

بنابراین قبل از لاگین حتما IP لوکال و یا ClientHost لوکال را هم وارد کنید.

اگر گیر افتادید به صورت زیر می‌شود رفع مشکل کرد:

تنها حالتی که تریگر لاگین را فعال نمی‌کند Dedicated Administrator Connection است یا DAC هم به آن گفته می‌شود. به صورت پیش فرض برای ایجاد این اتصال اختصاصی باید به کامپیوتری که اس کیوال سرور بر روی آن نصب است به صورت لوکال لاگین کرد و سپس در خط فرمان دستور زیر را صادر کنید (حرف A آن باید بزرگ باشد):

```
C:\>sqlcmd -A -d master -q "insert into IP_RESTRICTION(validip) values('<local machine>')"
```

به این صورت local machine به جدول IP های مجاز اضافه شده و می‌توانید لاگین کنید!

این نوع تریگرها در قسمت server objects در management studio ظاهر می‌شوند.

نظرات خوانندگان

نویسنده: مجید
تاریخ: ۱۳۹۱/۰۶/۲۹ ۱۱:۵۶

سلام جناب نصیری
مقاله بسیار جالبی بود.(حتی با اینکه متعلق به سه سال پیش بود)
بر اساس مقاله، تریگری برای لاگ کردن تمامی لاگین‌ها به سرور ایجاد کردم. اما به یک مشکل برخوردم:اگر لاگین sysAdmin باشد مشکلی نخواهیم داشت اما اگر تنها public باشد در هنگام لاگین با خطا مواجه خواهیم شد:

```
Logon failed for login 'm' due to trigger execution.
```

```
Changed database context to 'master'.
```

```
Changed language setting to us_english. (Microsoft SQL Server, Error: 17892)
```

آیا راه حلی وجود دارد؟متاسفانه من نتوانستم راه حلی پیدا کنم.
با تشکر...

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۹ ۱۲:۲۶

باید دسترسی به سرور شما داشت تا بشود اظهار نظر کرد. دسترسی‌ها چی هست. یوزرها به چه بانک‌های اطلاعاتی دسترسی دارند و مسایلی از این دست که از راه دور قابل بررسی نیست.
ولی درکل در اینجا از دیتابیس سیستمی master برای ساخت جدول Logging استفاده شده. این رو تغییر بدید به یک دیتابیس دیگر با سطح دسترسی عمومی.

```
USE [dbName]
go
INSERT INTO [dbName].schemaName.[Logging]
...
```

عنوان: **مقایسه نتایج الگوریتم‌های هش کردن اطلاعات در اس کیوال سرور و دات نت**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۱۴ ۲۱:۵۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

از اس کیوال سرور 2005 به بعد تابع HashBytes نیز به مجموعه توابع قابل استفاده در دستورات T-SQL اس کیوال سرور اضافه شده است که الگوریتم‌های MD2 | MD4 | MD5 | SHA | SHA1 را پشتیبانی می‌کند. برای مثال:

```
DECLARE @str1 VARCHAR(4),
    @str2 NVARCHAR(4)

--متن یونیکد اینجا ناقص ذخیره می‌شود
SET @str1 = 'وحید'

SET @str2 = N'وحید'

SELECT hashbytes('md5', @str1) --C82A7D721AAE517AD76EF1B871BC33CE

SELECT hashbytes('md5', @str2) --7D883091B80F3CD20B872CADBFDDACDF
```

اگر این نتایج را بخواهیم با استفاده از فضای نام استاندارد System.Security.Cryptography تولید کنیم، باید به encoding رشته دریافتی حتما دقت داشت؛ در غیر اینصورت نتایج یکسان نخواهند بود.
مهم‌ترین encoding های پشتیبانی شده در دات نت در جدول زیر برشمرده شده‌اند:

Encoding	تعداد بیت هر کاراکتر
ASCII	هر کاراکتر آن 7 بیت است
UTF7	هر کاراکتر آن 7 بیت است
UTF8	هر کاراکتر آن 8 بیت و یا یک بایت است
Unicode (UTF-16)	هر کاراکتر آن 16 بیت و یا دو بایت است
UTF32	هر کاراکتر آن 32 بیت و یا 4 بایت است

نوع nvarchar در اس کیوال سرور همانند حالت Encoding.Unicode دات نت است و هر کاراکتر آن 2 بایت می‌باشد.
این نکته هنگام استفاده از این توابع بسیار حائز اهمیت است. برای مثال اگر تابع HashBytes اس کیوال سرور را بخواهیم در دات نت پیاده سازی کنیم، به کلاس زیر خواهیم رسید:

```
using System.Text;
using System.Security.Cryptography;

class CHash
{
public static string GetMD5Hash(string input, Encoding encoding)
{
    byte[] bytes = new MD5CryptoServiceProvider().ComputeHash(encoding.GetBytes(input));
    StringBuilder chars = new StringBuilder();
    foreach (byte chr in bytes)
    {
        chars.Append(chr.ToString("x2"));
    }
    return chars.ToString();
}
}
```

در اینجا تنها حالت زیر با هش تولید شده یک فیلد یا متغیر از نوع nvarchar توسط تابع HashBytes اس کیوال سرور معادل است:

```
string result = CHash.GetMD5Hash("وحید", Encoding.Unicode);
```

پ.ن.

احتمالا عده‌ای را دیده‌اید که هر چقدر تلاش می‌کنند با سی شارپ متون ایران سیستم تحت داس را به نمونه‌های ویندوزی تبدیل کنند، کمتر موفق می‌شوند؛ علت را با توجه به جدول encoding فوق و عدم اطلاع از آن بهتر می‌توان بررسی کرد.

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۲/۱۵ ۰۰:۱۴:۰۰

با عرض سلام
آقای نصیری حدودا 2 3 سال پیش با خوندن کتاب شما و آقای هاشمیان با دات نت آشنا شدم بعد از مدتها وبلاگ شما رو بصورت اتفاقی پیدا کردم(ضمنا همچنا دنبال وبلاگ و یا... از آقای هاشمیان هستم) شما 2 نفر حق معلمی گزدن من دارید امیدوارم بتونم یه روز جبران کنم
قبل از هر چیز از شما بخاطر بروز نگه داشتن وبلاگتون تشکر میکنم
و اما سوال و درخواستی دارم
سوال
1:برای هش کردن پسورد از همین تابع ی که معرفی کردید (در قالب تریگر)استفاده کنم یا از توابع دات نت

پیشنهاد

2:اگه امکانش هست درباره وب سرویس ها هم تاپیک داشه باشید مخصوصا تکنولوژی جدید دات نت WCF

ممنون از توجهتون

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۱۵ ۰۰:۵۲:۰۰

سلام،
- شما لطف دارید.
- بر روی select نمی‌شود تریگر تعریف کرد بنابراین ...
و در کل فرقی نمی‌کند. یا یک رویه ذخیره شده بنویسید و کل عملیات را در آن پیاده کنید و سپس مثلا یک 0 یا 1 یا true یا false بازگشت دهید که شخص یوزر نیم و پسورد درستی وارد کرده (در این حالت می‌شود از روش اس کیوال سرور استفاده کرد برای هش کردن پسورد وارد شده و سپس مقایسه)
یا پسورد را هش کنید (توسط کلاینت دات نتی) و سپس با نمونه موجود در دیتابیس مقایسه کنید.
هر دو راه قابل انجام است و تفاوتی هم ندارد.
روش رویه ذخیره شده بهینه‌تر است و بدلیل کامپایل شدن سریعتر هم خواهد بود.

نویسنده: Mohammad Shams Javi
تاریخ: ۱۳۸۸/۰۲/۱۵ ۰۹:۰۹:۰۰

متشکرم
جالب بود

نویسنده: فاتحی
تاریخ: ۱۳۸۸/۰۲/۱۵ ۱۲:۱۲:۰۰

البته در استفاده از رویه ذخیره شده سمت sql این امکان وجود دارد که رمزعبور توسط SQL Profiler رصد شود!
و کلا پسورد هش نشده به دیتابیس پاس می شود.
جا داره منم از زحمات و مطالب شما تشکر و سپاس گذاری کنم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۱۵ ۱۴:۳۷:۰۰

البته کسی که دسترسی به SQL Profiler دارد یعنی دسترسی در حد برنامه نویس یا مدیر سیستم به او اعطاء شده و کلا باید به این دو اطمینان کرد چون در غیر اینصورت واقعا کارکردن مشکل خواهد شد.

نویسنده: حمیدرضا
تاریخ: ۱۳۸۸/۰۲/۱۵ ۱۷:۵۷:۰۰

اگه اشتباه نفهمیده باشم تعداد بایتهای یک کاراکتر توی استاندارد UTF-8 حداقل یک بایته (متغیره)، توی همون لینکی دادید نوشته 1 تا 4 بایت.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۱۵ ۱۹:۴۸:۰۰

درسته. جدول حداقل و حداکثر رو میشه در سایت زیر هم دید:

http://unicode.org/faq/utf_bom.html

ولی برای این حداقل‌ها و حداکثرها، اما و اگرهای زیادی هست (در مورد عدم تداخل با یکدیگر) که در لینک‌های زیر توضیح داده شده:

http://en.wikipedia.org/wiki/UTF-8

http://tools.ietf.org/html/rfc3629

رشته‌ها در دات نت فریم ورک از نوع UTF-16 هستند و برای اینکه به صورت صحیحی تبدیل به آرایه‌ای از بایت‌ها شده و در الگوریتم‌های مورد نظر استفاده شوند باید به این نکته دقت داشت.

عنوان: **تنظیم درجه سازگاری یک دیتابیس اس کیوال سرور**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۲۲ ۱۸:۳۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

اس کیوال سرور خود را به 2008 ارتقاء داده‌اید؟ آیا می‌دانید که مجاز هستید از تمامی امکانات جدید آن در دیتابیس‌های موجود خود استفاده کنید یا خیر؟

به همین منظور ابتدا کوئری زیر را اجرا نمائید:

```
Use master;
SELECT name, compatibility_level FROM sys.databases
```

لیست دیتابیس‌های موجود به همراه درجه سازگاری آن‌ها نمایش داده می‌شود. هر کدام که درجه سازگاری مساوی با 100 نداشت یعنی مجاز به استفاده از تمامی امکانات سرور جدید نیست و با آن همانند یک دیتابیس قدیمی بر اساس درجه سازگاری آن رفتار می‌شود.

تفسیر این اعداد مطابق اطلاعات زیر است:

```
60 = SQL Server 6.0
65 = SQL Server 6.5
70 = SQL Server 7.0
80 = SQL Server 2000
90 = SQL Server 2005
100= SQL Server 2008
```

برای تغییر این درجه سازگاری می‌توان از یکی از دستورات T-SQL زیر استفاده کرد:

```
-- Old way:
EXEC sp_dbcmptlevel 'AdventureWorks', 100

-- New way:
ALTER DATABASE 'AdventureWorks' SET COMPATIBILITY_LEVEL = 100 ;
```

که روش اول منسوخ شده در نظر گرفته شده و روش دوم پیشنهاد می‌شود.

برای مثال کوئری زیر عبارات Alter مورد نظر را جهت ارتقاء دیتابیس‌های موجود که درجه سازگاری آن‌ها 100 نیست (هنوز به اس کیوال سرور 2008 ارتقاء پیدا نکرده‌اند)، تولید می‌کند:

```
SELECT 'ALTER DATABASE ' + NAME + ' SET COMPATIBILITY_LEVEL = 100;' TSQL
FROM   sys.databases
WHERE  compatibility_level <> 100
```

عنوان: **انتقال فایل‌های دیتابیس اس کیوال سرور 2008**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۲۳ ۲۱:۵۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

روز قبل نیاز بود تا فایل‌های mdf و ldf دیتابیس‌ها جابجا شوند (یک هارد بزرگتر و از این مسایل).
برای جابجا کردن این فایل‌ها هم روش معمول detach و سپس attach است. ابتدا روی دیتابیس کلیک راست کرده و detach .
حالا فایل‌ها را جابجا می‌کنید و سپس attach . یا می‌شود بک آپ کامل گرفت و بعد ری استور کرد.
عموما هم نمی‌توان دیتابیس در حال استفاده را detach کرد. باید دیتابیس ابتدا single user شود و بعد می‌توان این‌کار را انجام داد.
تا اینجای کار متداول است. همه چیز به خوبی انجام شد. سپس در لحظه attach ، دیتابیس‌ها به صورت read only اتچ شدند با آیکونی سیاه رنگ در management studio . (و رنگ من هم بلافاصله به همین رنگ متمایل شد!)
بعد از مدتی جستجو مشخص شد که در اس کیوال سرور 2008 برای کاهش سطح حمله به سرور، از یک سری یوزر با دسترسی کم برای نصب اس کیوال سرور استفاده می‌شود (بسیار هم خوب) و اس کیوال سرور 2008 ، یک سری یوزر مخصوص را هم در حین نصب ایجاد می‌کند که به صورت خودکار بر روی پوشه دیتای شما دسترسی full control دارد برای اینکه بتواند کارش را انجام دهد.
حال اگر شما در جای دیگری پوشه‌ای درست کردید و این دیتابیس‌ها را منتقل نمودید، مجددا پیش از هر کاری باید این دسترسی را برقرار کنید و گرنه اس کیوال سرور مجوز write نخواهد داشت؛ به همین جهت دیتابیس به صورت read only در management studio با رنگ مشکی ظاهر می‌شود.
نام این کاربر مخصوص به صورت زیر است:

```
SQLServerMSSQLUser$ComputerName$MSSQLSERVER
```

پس از برقراری دسترسی هم مشکل برطرف نمی‌شود. باید دستور زیر را نیز اجرا نمود:

```
ALTER DATABASE myDB SET READ_WRITE
```

اجرای این دستور نیز، نیاز به حالت single user دارد.

پ.ن.
می‌توان دسترسی یوزر سرویس اس کیوال سرور 2008 را نیز مانند نگارش‌های قبلی به حالت local system تغییر داد (یا هر اکانت دیگری با دسترسی بالا) تا این مشکلات نباشد؛ ولی بدیهی است سطح حمله به سرور نیز به همین اندازه افزایش می‌یابد.

نظرات خوانندگان

نویسنده: SirAsad
تاریخ: ۱۳۸۸/۰۲/۲۴ ۰۹:۰۲:۰۰

من به شخصه برای انتقال دیتابیس اول میام سرویس دیتابیس رو stop می کنم بعد منتقل میکنم و بعد دوباره سرویس رو Start می کنم . آیا این روش موردی داره ؟
حتی تو یه برنامه مجبور شدم این کار رو توسط برنامه نویسی هم انجام بدم ...چونکه مجبور از فایل دیتابیس کپی بگیرم .

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۲۴ ۱۰:۲۰:۰۰

- البته منظور از انتقال در اینجا بحث move و cut بود نه copy . بنابراین در حالت move ، سرور به نحوی باید از محل جدید مطلع بشه و نیاز به attach هست.
- ضمنا اگر به دنبال امنیت هستید، کپی کردن mdf و ldf ممکن است کار دست شما بده و یک وقت خیلی ساده attach نشه. ولی بک آپ این مشکل رو نداره.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۲۴ ۱۴:۱۶:۰۰

این عدم دسترسی ممکن است هنگام attach سبب بروز خطاهای زیر هم شود (جهت تکمیل بحث):

```
Database cannot be upgraded because it is read-only or has read-only files. Make the database or files -
writeable, and rerun recovery

CREATE FILE encountered operating system error 5 (failed to retrieve text for this error. Reason: 1815) while -
attempting to open or create the physical file
```

عنوان: **مقایسه رکوردهای دو جدول**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۰۴ ۲۱:۱۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

گاهی از اوقات یک سری از امکانات جدید در دسترس هستند اما فراموش می‌شوند. برای مثال روش یافتن رکوردهای غیر یکسان دو جدول یکسان. مثلا یک دیتابیس قدیمی دارید دقیقا مشابه دیتابیس کاری فعلی با همان ساختار (ری استور شده از یک بک آپ). اکنون می‌خواهید بدانید در طول این مدت چه رکوردهایی به دیتابیس کاری اضافه شده که در دیتابیس قدیمی ری استور شده موجود نیست و کلا کدام رکوردها با هم متفاوتند. چه باید کرد؟

مثال:
دو جدول موقتی یکسان زیر را در نظر بگیرید.

```
CREATE TABLE #tableA
(
column1  INT,
column2  INT
)

INSERT INTO #tableA
VALUES
(1,1)
,(1, 2)
,(1, 3)
,(2, 1)

SELECT column1,
     column2
FROM   #tableA

CREATE TABLE #tableB
(
column1  INT,
column2  INT
)

INSERT INTO #tableB
VALUES
(1,1)
,(1, 3)
,(2, 2)

SELECT column1,
     column2
FROM   #tableB
```

یک سری دیتای دلخواه به آن‌ها اضافه شده است. (از روش اضافه کردن چندین رکورد توسط یک عبارت insert که در اس کیوال سرور 2008 معرفی شده، استفاده گردیده است)

```
#tableA
column1   column2
-----------  -----------
1               1
1               2
1               3
2               1


#tableB
column1   column2
-----------  -----------
1              1
1              3
2              2
```

اکنون می‌خواهیم رکوردهایی از جدول A را که در جدول B نیستند، پیدا کنیم. روش متداول انجام این‌کار در اس کیوال سرور 2000 به صورت زیر است:

```
SELECT column1,
     column2
FROM   #tableA
WHERE  NOT EXISTS (
         SELECT *
         FROM   #tableB
         WHERE  #tableA.column1 = #tableB.column1
                AND #tableA.column2 = #tableB.column2
     )
```

```
column1   column2
-----------  -----------
1               2
2               1
```

و یا روش زیباتر انجام این‌کار که از اس‌کیوال سرور 2005 به بعد معرفی شده، به صورت زیر می‌باشد:

```
SELECT column1, column2 FROM #tableA
EXCEPT
SELECT column1, column2 FROM #tableB
```

```
column1   column2
-----------  -----------
1               2
2               1
```

Except

رکوردهای منحصربفردی از کوئری سمت چپ را که در کوئری سمت راست وجود ندارند، بر می‌گرداند.

در این حالت تعداد ستون‌های در نظر گرفته شده برای مقایسه باید یکسان و یک نوع باشند.

همچنین اگر می‌خواهید رکوردهایی از جدول A را که در جدول B وجود دارند بیابید، می‌توان از intersect استفاده کرد.

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۳/۰۵ ۰۵:۵۴:۰۴

جالب بود ولی این ها فقط برای SQL Server نیستند. بلکه دستورات کلی SQL هستند!

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۰۵ ۰۹:۴۳:۵۰

من عموما مطالبی رو که توسط یک ناشناس طرح میشه رو حذف می‌کنم. محض اطلاع.
+
اپراتور except جزو استاندارد ANSI SQL است ولی تا اس کیوال سرور 2005 در این محصول پشتیبانی نمی‌شد. (درک ضمنی این مطلب از مقاله فوق کار مشکلی نیست)

عنوان: **فشرده سازی اطلاعات در SQL server 2008**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۱۳ ۲۰:۵۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

علاوه بر فشرده سازی خودکار بک آپ‌ها که پیشتر در مورد آن‌ها صحبت شد ، اس کیوال سرور 2008 دو نوع فشرده سازی دیگر را نیز پشتیبانی می‌کند:

Row Compression :

حالت row compression نحوه‌ی ذخیره سازی فیزیکی داده‌ها را تغییر می‌دهد. فعال سازی آن اثرات زیر را خواهد داشت:
الف) متادیتای هر رکورد را حداقل می‌کند (منظور از متادیتا اطلاعاتی مانند اطلاعات ستون‌ها، طول و آفست و غیره است)
ب) داده‌های عددی و رشته‌هایی با طول ثابت، به صورت اطلاعاتی با طول متغیر ذخیره خواهند شد، درست مانند varchar ها.

برای ایجاد جدولی که row compression در آن به صورت پیش‌فرض فعال است، می‌توان مانند مثال زیر عمل کرد:

```
CREATE TABLE MyTable
(
      ID int identity Primary key,
      Name char(100),
      Email char(100)
)
WITH (DATA_COMPRESSION = Row);

GO
```

و اگر جدول موجودی را می‌خواهید تغییر داده و این خاصیت را بر روی آن فعال نمائید، روش زیر را اعمال کنید:

```
Alter TABLE MyTable REBUILD WITH (DATA_COMPRESSION=Row, MAXDOP=2);
```

در اینجا MAXDOP مشخص می‌کند که از چند CPU باید برای فشرده سازی استفاده شود. (اگر جدولی حجیم دارید، به این صورت می‌توان عملیات فشرده سازی را سریعتر به پایان رساند)

Page Compression :

در روش دوم فشرده سازی اطلاعات در اس‌کیوال سرور 2008 ، که مهم‌ترین حالت موجود نیز می‌باشد، اطلاعات مشترک، بین سطرهای یک صفحه به اشتراک گذاشته می‌شوند. این روش از فناوری‌های زیر استفاده می‌کند:
الف) روش row compression که در مورد آن صحبت شد جزئی از این روش است.
ب) Prefix Compression : به ازای هر ستون در یک صفحه، Prefix های تکراری یافت شده و در هدر مخصوص فشرده سازی ذخیره می‌شوند (محل این هدر پس از هدر صفحه است). سپس هرجایی که به این Prefix ها اشاره شده‌باشد، عدد منحصربفرد شناسایی کننده آن‌ها نسبت داده می‌شود.
ج) Dictionary Compression : در این حالت مقادیر تکراری یک صفحه جستجو شده و در هدر فشرده سازی صفحه ذخیره می‌شوند. حالت Prefix Compression فقط به یک ستون منحصر می‌شود اما Dictionary Compression به کل صفحه اعمال می‌گردد.

برای فعال سازی آن در یک جدول جدید به روش زیر می‌توان عمل نمود:

```
CREATE TABLE MyTable
(
      ID int identity Primary key,
```

```
      Name char(100),
      Email char(100)
)
WITH (DATA_COMPRESSION = Page);
```

و برای اعمال آن به جدولی موجود از روش زیر می‌توان استفاده کرد:

```
Alter TABLE MyTable REBUILD WITH (DATA_COMPRESSION=Page, MAXDOP=2);
```

یک سری رویه‌های ذخیره شده سیستمی جدید نیز برای محاسبه حجم جداول، پیش و پس از فشرده سازی (بدون فشرده سازی واقعی) نیز در این نگارش گنجانده شده‌اند که به شرح زیر هستند:

```
-- بررسی اینکه چه میزان فضا با اعمال فشرده سازی صفحات قابل صرفه جویی خواهد بود
EXEC sp_estimate_data_compression_savings 'schemaname', 'TableName', NULL, NULL, 'PAGE';

-- بررسی اینکه چه میزان فضا با اعمال فشرده سازی ردیف‌ها قابل صرفه جویی خواهد بود
EXEC sp_estimate_data_compression_savings 'schemaname', 'TableName', NULL, NULL, 'ROW';
```

بنابراین قبل از اینکه فشرده سازی را فعال نمائید، ابتدا بررسی کنید آیا واقعا میزان قابل توجهی اطلاعات فشرده خواهند شد و نتیجه حاصل رضایت بخش است یا خیر. همچنین باید درنظر داشت که جداول و یا ایندکس‌هایی که read و write بالایی دارند برای این منظور مناسب نیستند. برای یافتن آن‌ها کوئری زیر را اجرا کنید:

```
USE dbName;
SELECT objectname = OBJECT_NAME(s.object_id),
      indexname = i.name,
      i.index_id,
      reads = range_scan_count + singleton_lookup_count,
      'leaf_writes' = leaf_insert_count + leaf_update_count + leaf_delete_count,
      'leaf_page_splits' = leaf_allocation_count,
      'nonleaf_writes' = nonleaf_insert_count + nonleaf_update_count +
      nonleaf_delete_count,
      'nonleaf_page_splits' = nonleaf_allocation_count
FROM   sys.dm_db_index_operational_stats (DB_ID(), NULL, NULL, NULL) AS s
      INNER JOIN sys.indexes AS i
           ON  i.object_id = s.object_id
WHERE  OBJECTPROPERTY(s.object_id, 'IsUserTable') = 1
      AND i.index_id = s.index_id
ORDER BY
      leaf_writes DESC,
      nonleaf_writes DESC
```

و جهت تکمیل مبحث می‌توان به مقاله بسیار جامع زیر که اخیرا توسط مایکروسافت منتشر شده است رجوع نمود:

Data Compression: Strategy, Capacity Planning and Best Practices

عنوان: **تهیه بک آپ‌های خودکار از SQL Server Express**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۲۱ ۲۰:۴۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

SQL Server express edition

نگارش مجانی و ساده شده‌ی اس کیوال سرور است. این نگارش مجانی فاقد SQL Server agent برای زمان بندی انجام امور تکراری، برای مثال تهیه بک آپ‌های خودکار است. این مورد در کل ایرادی محسوب نمی‌شود زیرا می‌توان این عملیات را با استفاده از سیستم استاندارد scheduled tasks ویندوز نیز پیاده سازی کرد.

برنامه خط فرمان سورس بازی به نام ExpressMaint موجود است که می‌تواند از دیتابیس‌های اس کیوال سرور اکسپرس (و غیر اکسپرس) بک آپ تهیه کند. فقط کافی است این برنامه را به عنوان یک scheduled task ویندوز معرفی کنیم تا در زمان‌های تعیین شده در مکان‌هایی مشخص، بک آپ تهیه کند. همچنین این برنامه فایل‌های بک آپ تهیه شده را نیز تعیین اعتبار می‌کند.

با پارامترهای خط فرمان آن در این‌جا می‌توانید آشنا شوید. خلاصه کاربردی آن را به صورت چند دستور در ادامه مرور خواهیم کرد.

الف) یک فایل bat را با محتوای زیر درست کنید :

```
C:\backup\expressmaint.exe -S (local)\sqlexpress -D ALL_USER -T DB -R C:\backup -RU WEEKS -RV 2 -B
C:\backup -BU DAYS -BV 2 -V -C
```

توضیحات: در این فایل bat ، مسیر فایل اجرایی برنامه حتما باید دقیقا ذکر شود و گرنه scheduled task ویندوز درست کار نخواهد کرد. همچنین instance اس کیوال سرور اکسپرس در اینجا (local)\sqlexpress فرض شده است. این دستور از تمامی دیتابیس‌های غیرسیستمی در مسیر C:\backup بک آپ می‌گیرد (به ازای هر دیتابیس یک پوشه مجزا درست خواهد کرد و هر فایل را بر اساس تاریخ و ساعت مشخص می‌سازد). همچنین لاگ عملیات را نیز در همان پوشه تهیه می‌کند (نتیجه اعتبار سنجی صورت گرفته بر روی بک آپ‌های تهیه شده در این فایل‌ها ثبت می‌شود). مطابق پارامترهای بکار گرفته شده، بک آپ‌های قدیمی‌تر از دو روز به صورت خودکار حذف شده و لاگ فایل‌ها به مدت 2 هفته نگهداری می‌شوند.

ب) برای اجرای زمان بندی شده‌ی این فایل bat تهیه شده، دستورات زیر را در خط فرمان اجرا کنید (فرض بر این است که فایل bat تهیه شده در مسیر مشخص شده C:\backup\backup.bat قرار دارد) :

```
AT 23:30 /EVERY:m,t,w,th,f,s,su C:\backup\backup.bat
AT 11:30 /EVERY:m,t,w,th,f,s,su C:\backup\backup.bat
```

به این صورت ویندوز هر روز، دوبار در طول روز از کلیه دیتابیس‌ها به صورت خودکار بک آپ تهیه می‌کند.

روشی که در این‌جا ذکر شد منحصر به نگارش express نیست و با کلیه نگارش‌های SQL Server سازگار است.

عنوان: **بازسازی msdb تخریب شده**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۲۸ ۱۲:۲۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

حاصل قطع برق و یا یک ری استارت دستی ناصحیح را در نظر بگیرید:

```
Database 'msdb' cannot be opened. It has been marked SUSPECT by recovery. See the SQL Server errorlog
for more information. (Microsoft SQL Server, Error: 926)
```

Msdb از نوع دیتابیس‌های سیستمی است و نمی‌شود مطابق روال متداول دیتابیس‌های SUSPECT شده آن‌را بازیابی کرد. این روش متداول به صورت زیر است:

```
ALTER DATABASE DBName SET EMERGENCY
DBCC checkdb('DBname')
ALTER DATABASE DBName SET SINGLE_USER WITH ROLLBACK IMMEDIATE
DBCC CheckDB ('DBName', REPAIR_ALLOW_DATA_LOSS)
ALTER DATABASE DBName SET MULTI_USER
```

در ابتدای کار دیتابیس در حالت اورژانسی قرار می‌گیرد. بعد وضعیت و میزان تخریب نمایش داده شده، سپس تک کاربره می‌شود. در ادامه به اس کیوال سرور اجازه داده می‌شود که دیتابیس را با هر وضعی (حتی به قیمت از دست رفتن تعدادی رکورد) ترمیم کند و در آخر دیتابیس مجددا به حالت چند کاربره بازگشت داده می‌شود.
این روشی است که سال قبل با قطعی‌های مکرر برق زیاد کاربرد داشت.

اما دیتابیس سیستمی msdb را نمی‌شود در حالت اورژانسی قرار داد؛ بنابراین باید به دنبال راه چاره‌ی دیگری بود. پس از مدتی جستجو در وبلاگ‌های msdn ، راه حل زیر یافت شد و کاملا عملی است (تست شده!) :

روش زیر در مورد اس کیوال سرور 2008 ، 2005 و حتی 2000 نیز قابل استفاده است.

ابتدا خونسردی خودتان را حفظ کنید! الان فقط دیگر با management studio نمی‌توانید دیتابیس‌ها را مرور کنید و همچنین تمام job های تعریف شده شما نابود شده‌اند! اما سرور به کار عادی خودش می‌تواند ادامه دهد. سپس :

الف) تمام سرویس‌های مربوط به اس کیوال سرور را stop کنید. به کنسول سرویس‌ها مراجعه کرده و هر آنچه که در نام آن sql را مشاهده می‌کنید، stop کنید.

ب) با استفاده از خط فرمان، ابتدا به مسیر زیر وارد شوید:

```
cd "C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\Binn\"
```

و سپس دستور زیر را اجرا نمائید:

```
start sqlservr.exe -c -m -T3608
```

به این ترتیب اس کیوال سرور در یک حالت حداقل که بتوان دیتابیس msdb تخریب شده را detach کرد راه اندازی می‌شود. (پرچم 3608 مجوز detach کردن این دیتابیس را می‌دهد)

ج) management studio را اجرا کنید. زمانیکه پنجره کانکت ظاهر می‌شود آن‌را کنسل کرده و در نوار ابزار بالای صفحه روی دکمه new query کیک کنید (چون حالت راه اندازی سرور در حالت تک کاربره است نمی‌خواهیم اتصال دیگری برقرار شود و در کار اخلال کند). با کلیک بر روی new query پنجره connect to server ظاهر می‌شود. در همین پنجره بر روی دکمه options کلیک کرده در برگه connection properties در قسمت connect to database نام master را وارد نمود و اکنون بر روی دکمه connect کلیک نمائید.

ج) سپس دستور زیر را وارد کنید تا دیتابیس msdb را بتوان detach کرد.

```
Use master;
sp_detach_db 'msdb'
```

مراحلی که عنوان شد مهم است. اگر به این صورت عمل نکنید با پیغام خطای زیر مواجه خواهید شد:

```
Cannot detach an opened database when the server is in minimally configured mode
```

اگر به این خطا برخوردید، یکبار دیگر از صفر شروع کنید. تمام سرویس‌های مرتبط با sql را استاپ کنید (حتی در صورت نیاز کارت شبکه سرور را نیز غیرفعال کنید). و از مرحله الف مجددا شروع نمائید تا حتما حالت تک کاربره‌ی اتصال برقرار شود. (همچنین پنجره‌ی کوئری جدیدی را نیز باز نکنید چون در این حالت فقط و فقط یک اتصال مجاز است)

تا اینجا موفق شدیدم که دیتابیس msdb را detach کنیم. اکنون به پوشه دیتابیس‌ها مراجعه کرده و mdf و ldf این دیتابیس تخریب شده را rename کنید (به هر اسمی که مایل بودید).

د) اکنون نوبت بازسازی مجدد این دیتابیس است.

محتویات فایل instmsdb.sql را که در مسیر C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\install قرار دارد، در پنجره‌ی کوئری تک کاربره‌ای که در مرحله قبل بازکرده‌ایم، copy/paste کرده و دکمه F5 را فشار دهید. پس از مدتی دیتابیس msdb باز سازی شده و مشکل برطرف می‌شود.

ه) اکنون سرور را stop و start کنید یا کلا کامپیوتر سرور را restart کنید تا تمامی سرویس‌های stop شده راه اندازی مجدد شوند.

نظرات خوانندگان

نویسنده: نیما
تاریخ: ۱۳۸۸/۰۴/۲۸ ۱۶:۲۴:۴۲

سلام استاد نصیری
باز هم یک نکته بی نظیر. از اینکه ما رو هم در دانسته هاتون شریک میکنین ممنونم

نویسنده: mojtaba
تاریخ: ۱۳۸۸/۰۷/۱۲ ۱۷:۳۵:۰۷

سلام
من در همان مرحله اول با این پیغام مواجه میشم:
error getting instance ID from name

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۱۲ ۱۸:۱۳:۵۵

سلام
دو علت می‌تونه داشته باشه
1- یوزری که اس کیوال سرور تحت آن اجرا می‌شود دسترسی به مسیر رجیستری زیر ندارد:
HKLM\Software\Microsoft\Microsoft SQL Server\Instance Names
2- نام وهله (InstanceName) مورد نظر را باید دقیقا ذکر کنید:
start sqlservr.exe -c –SServername\InstanceName -T3608

عنوان: **مقایسه حساس به حروف کوچک و بزرگ در SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۰۷ ۲۱:۳۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

چندین روش برای انجام مقایسه حساس به حروف کوچک و بزرگ (case sensitive) در SQL Server وجود دارد که در ادامه آن‌ها را مرور خواهیم کرد:

ابتدا جدول موقتی زیر را جهت آزمایشات بعدی در نظر بگیرید

```
CREATE TABLE #tblTest
(
 f1 NVARCHAR(50)
)

INSERT INTO #tblTest  (f1) VALUES('Test1')
INSERT INTO #tblTest  (f1) VALUES('TEST1')
```

الف) استفاده از collation صحیح
عموما هنگام نصب اس کیوال سرور از collation غیرحساس به کوچکی و بزرگی حروف استفاده می‌شود و این مورد سبب می‌شود که پیش فرض ایجاد دیتابیس‌ها نیز به همین صورت باشد (هر چند کاملا قابل کنترل و تنظیم است). به صورت پویا می‌توان این collation را در کوئری‌ها نیز اعمال نمود. برای مثال:

```
SELECT f1 FROM  #tblTest WHERE f1 COLLATE SQL_Latin1_General_CP1_CS_AS = 'Test1'
```

در این مثال اگر از collation حساس به کوچکی و بزرگی حروف استفاده نمی‌شد، خروجی هر دو رکورد ثبت شده می‌بود.

ب) استفاده از تابع BINARY_CHECKSUM اس کیوال سرور (نوعی الگوریتم ویژه، شبیه به امضای دیجیتال و هش کردن اطلاعات است)

```
SELECT f1 FROM  #tblTest WHERE BINARY_CHECKSUM(f1) = BINARY_CHECKSUM('Test1')
```

و یا استفاده از امکانات هش کردن اطلاعات در اس کیوال سرور: (امضای دیجیتال دو رشته با هم مقایسه می‌شوند)

```
SELECT f1 FROM  #tblTest WHERE  hashbytes('md5',f1) = hashbytes('md5',N'Test1')
```

ج) مقایسه در حالت باینری

```
SELECT f1 FROM  #tblTest WHERE  convert(varbinary(50),f1) = convert(varbinary(50),N'Test1')
```

عنوان: **مزیت‌های استفاده از رویه‌های ذخیره شده؛ واقعیت یا توهم؟!**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۳ ۲۱:۱۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

متن زیر یک سری نکات و یا شاید توهماتی را مطرح می‌کند که در مورد رویه‌های ذخیره شده در اس کیوال سرور رایج هستند.

1- رویه‌های ذخیره شده در مقابل SQL Injection مقاوم هستند. کوئری‌های Ad hoc همیشه این آسیب پذیری را به همراه دارند.
نادرست است! رویه‌های ذخیره شده‌ای که رشته‌ها را به صورت پارامتر دریافت کرده و آن‌ها را به صورت یک عبارت sql اجرا می‌کنند، آسیب پذیر هستند. اگر هنگام استفاده از کوئری‌های Ad hoc از پارامترها استفاده شود، در برابر حملات SQL Injection مصون خواهید بود.

2- execution plan رویه‌های ذخیره شده کش می‌شوند اما این Plan برای کوئری‌های Ad hoc هر بار محاسبه و تولید می‌گردد.
نادرست است! اس کیوال سرور تا این اندازه بی هوش نیست! اگر execution plan ایی موجود باشد حتما استفاده خواهد شد و برای موتور اس کیوال سرور اصلا اهمیتی ندارد که کوئری در حال اجرا از یک رویه ذخیره شده صادر شده است یا از یک کوئری Ad hoc . رویه‌های ذخیره شده پیش کامپایل شده نیستند و مانند تمامی کوئری‌های دیگر در زمان اجرا کامپایل می‌شوند.

3- زمانیکه از رویه ذخیره شده استفاده می‌کنید همه چیز را در یک مکان به صورت متمرکز و مجتمع خواهید داشت (مدیریت بهتر)
نادرست است! در یک مکان متمرکز در اختیار شما نیستند. برنامه جای خود را دارد و رویه‌های ذخیره شده در دیتابیس در جای دیگری قرار دارند و برای مثال اگر قرار باشد یک پارامتر را به رویه ذخیره شده خود اضافه کنید، کدهای شما نیز باید تغییر کنند.

4- می‌توان از یک رویه ذخیره شده استفاده مجدد کرد (در نقاط مختلف یک کد) و اعمال تغییرات تنها در یک مکان (دیتابیس) باید صورت گیرد.
هر چند این مورد درست است، اما باید دقت داشت که اگر چندین برنامه از این رویه ذخیره شده استفاده می‌کنند نباید تغییرات شما باعث از کار افتادن سایر برنامه‌ها شوند.

5- می‌توان رویه ذخیره شده را بدون نیاز به توزیع مجدد برنامه تغییر داد.
این مورد تا حدودی صحیح است. اگر تنها بحث بهینه سازی و امثال آن مطرح باشد صحیح است اما اگر واقعا نیاز به تغییر یک کوئری در رویه ذخیره شده وجود داشته باشد به احتمال زیاد برنامه نیز باید دستخوش تغییراتی گردد تا این دو با هم هماهنگ شوند.

نظر شما چیست؟

نظرات خوانندگان

نویسنده: نیما
تاریخ: ۱۳۸۸/۰۵/۱۳ ۲۱:۵۳:۳۱

سلام استاد نصیری
در رابطه با نکته دوم من فکر کنم که هر بار هر کوئری میاد پارس بشه و بعد براش Execution Plan در نظر گرفته بشه. یادمه خودم امتحان کردم استور سریعتر بود. به شخصه ترجیح میدم که داخل برنامم از دستورات اس کیو ال استفاده نکنم چون مدیریتش سخت تره
ممنون از مطلبتون

نویسنده: محمد
تاریخ: ۱۳۸۸/۰۵/۱۳ ۲۳:۱۴:۳۳

مرسی از مطلبتون،
منظورتون از کوئری‌های Ad Hoc کوئری‌هایی ست که در زمان اجرا ساخته می‌شوند یا در سمت کلاینت قرار دارند هست؟ یا تعریف دیگری دارد؟
در مورد نکته‌ی دوم من این مطلب رو جایی خوندم که کوئری‌های Ad Hoc هر بار کامپایل میشن و Exec Plan شون بدست میاد و.... البته اون مطلب در مورد SQL Server 2000 بود.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۳ ۲۳:۴۵:۲۴

بله. Ad hoc query به کوئری‌هایی گفته می‌شود که توسط برنامه بر اساس نیاز آنی کاربر ساخته می‌شوند.
http://it.toolbox.com/wiki/index.php/Ad_hoc_query

البته اینجا بیشتر منظور همان کوئری‌های مخلوط با سورس کدهای شما ست.

نویسنده: افشار محبی
تاریخ: ۱۳۸۸/۰۵/۱۴ ۰۷:۴۶:۰۴

من شخصاً ترجیح می‌دهم به جای استفاده از SP و امثالهم از یک ORM خوب مثل NHibernate استفاده کنم. این طوری می‌توانم بخش زیادی از منطق را در داخل زبان مورد استفاده مثل #C پیاده‌سازی کنم.

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۵/۱۴ ۱۰:۴۹:۳۲

منم با آقای محبی موافقم.با این تفاوت که LINQ رو NHibernate ترجیح میدم.

با SP ها درسته که تقرایبا کار ها مدیریت بهتری دارن ولی مثلا زمانی که لازم باشه SP رو تغییر بدید باید خیلی چیز ها باز بینی بشن.

این مشکل وقتی با DataSet ها کار می کنی بیشتر هم میشه.

موفق باشید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۵ ۱۳:۳۱:۱۸

چند مورد را در مورد نکته دوم جهت تکمیل بحث اضافه کنم:

در اس کیوال سرور 2000 اگر از ad hoc کوئری استفاده شود احتمال recompile شدن بسیار زیاد است اما این مورد از اس کیوال سرور 2005 به بعد به شدت بهبود یافته. برای مطالعه بیشتر
Execution Plan Caching and Reuse
http://technet.microsoft.com/en-us/library/ms181055%28SQL.90%29.aspx

نکته مهمی که در ad hoc کوئری‌ها ممکن است سبب recompile شدن plan اجرایی آن شود بحث تغییر نوع یا اندازه‌ی پارامترهای مورد استفاده است. اگر این موارد به صورت صریح ذکر شوند و از پارامترهایADO.Net استفاده شوند، تشخیص نوع و اندازه پارامترها برای اس کیوال سرور بسیار ساده شده و حتما از کش بجای recompile استفاده خواهد کرد. برای مطالعه بیشتر:
Parameters and Execution Plan Reuse
http://technet.microsoft.com/en-us/library/ms175580%28SQL.90%29.aspx
دقیقا همین نکته را اگر از Nhibernate استفاده می‌کنید نیز باید رعایت کنید:
NHibernate queries & sql server execution plans
/http://testdrivendevelopment.wordpress.com/2009/03/10/nhibernate-queries-sql-server-execution-plans
خلاصه این مقاله به این صورت است که اس کیوال سرور را وادار نکنیم که از recompile به جهت مشخص نبودن طول یا اندازه پارامترها، هر بار استفاده نماید.

نویسنده: Saeed Zarinfam
تاریخ: ۱۳۸۸/۰۵/۱۸ ۰۸:۳۵:۱۰

سلام
به چه مبحث خوبی اشاره کردید.
یکی دیگر از نکاتی که مهم است ، از یکپارچگی درآمدن تیم پیاده سازی است که از لحاظ مدیریت پروژه بسیار مهم است. بدین معنی که اعضای تیم پروژه باید مفهوم جدیدی به نام sp را آموزش ببینند که این امر باعث کاهش سرعت توسعه می شود در حالی که با استفاده از یک ORM برنامه نویسان در لایه DAL با همان زبان برنامه نویسی که در لایه BLL برنامه می نویسند کار می کنند.
از طرف دیگر استفاده از sp ممکن است شما را به استفاده از پایگاه داده خاصی محدود کند.

عنوان: **استفاده از قابلیت Script Data اس کیوال سرور 2008 از طریق برنامه نویسی**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۶ ۱۲:۵۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

همانطور که مطلع هستید قابلیت تهیه عبارات Insert از جداول یک دیتابیس، به صورت استاندارد به management studio 2008 اضافه شده است. برای استفاده از این قابلیت از طریق برنامه نویسی به صورت زیر می‌توان عمل نمود:

الف) سه ارجاع را به اسمبلی‌های زیر اضافه نمائید:

Microsoft.SqlServer.ConnectionInfo

Microsoft.SqlServer.Management.Sdk.Sfc

Microsoft.SqlServer.Smo

ب) اکنون کدی که عملیات Script Data را با استفاده از قابلیت‌های SMO ارائه می‌دهد به صورت زیر خواهد بود:

```
using System;
using System.Text;
using Microsoft.SqlServer.Management.Smo;
using Microsoft.SqlServer.Management.Common;
using Microsoft.SqlServer.Management.Sdk.Sfc;

        /// <summary>
        /// تهیه اسکریپت رکوردها
        /// </summary>
        /// <param name="dbName">نام دیتابیس مورد نظر</param>
        /// <param name="table">نام جدولی که باید اسکریپت شود</param>
        /// <param name="instance">وهله سرور</param>
        /// <param name="userName">نام کاربری جهت اتصال</param>
        /// <param name="pass">کلمه عبور جهت اتصال به سرور</param>
        /// <returns>اسکریپت تهیه شده</returns>
        public static string ScriptData(string dbName, string table, string instance, string userName,
string pass)
        {
            try
            {
                ServerConnection serverConnection = new ServerConnection(instance, userName, pass);
                Server server = new Server(serverConnection);
                Database database = server.Databases[dbName];
                if (database == null) return string.Empty;
                Table tb = database.Tables[table];
                Scripter scripter = new Scripter(server) {Options = {ScriptData = true}};
                if (tb == null) return string.Empty;
                StringBuilder sb = new StringBuilder();
                foreach (string s in scripter.EnumScript(new Urn[] {tb.Urn}))
                    sb.AppendLine(s);
                return sb.ToString();
            }
            catch(Exception ex)
            {
                //todo: log ...
                return string.Empty;
            }
        }
```

نکته:

اسمبلی‌های SMO به همراه مجموعه SQL Server 2008 نصب می‌شوند. بنابراین متد و برنامه‌ی فوق بر روی سروری با این مشخصات بدون مشکل اجرا خواهد شد. اما اگر قصد توزیع برنامه خود را دارید باید Microsoft SQL Server 2008 Management Objects را نیز به همراه برنامه خود نصب نمائید.

عنوان: **روش‌هایی برای حذف رکوردهای تکراری**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۲۷ ۲۲:۲۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

هر برنامه نویسی در طول عمر کاری خود حداقل یکبار با این مساله مواجه خواهد شد: "چگونه یک سری رکورد تکراری موجود را باید حذف کرد؟"

ابتدا جدول موقتی زیر را که در آن یک سری رکورد تکراری ثبت شده است در نظر بگیرید:

```
CREATE TABLE #Employee
(
ID          INT,
FIRST_NAME  NVARCHAR(100),
LAST_NAME   NVARCHAR(300)
)

INSERT INTO #Employee VALUES ( 1, 'Vahid', 'Nasiri' );
INSERT INTO #Employee VALUES ( 2, 'name1', 'lname1' );
INSERT INTO #Employee VALUES ( 3, 'name2', 'lname2' );
INSERT INTO #Employee VALUES ( 2, 'name1', 'lname1' );
INSERT INTO #Employee VALUES ( 3, 'name2', 'lname2' );
INSERT INTO #Employee VALUES ( 4, 'name3', 'lname3' );
```

روش‌های حذف رکوردهای تکراری جدول موقتی Employee

الف) استفاده از یک جدول موقتی دیگر بر اساس تمامی فیلدهای موجود

```
SELECT DISTINCT *
FROM   #Employee

SELECT * INTO #DuplicateEmployee
FROM   #Employee

INSERT #DuplicateEmployee
SELECT DISTINCT *
FROM   #Employee

BEGIN TRAN
DELETE #Employee
INSERT #Employee
SELECT *
FROM   #DuplicateEmployee

   COMMIT TRAN

  DROP TABLE #DuplicateEmployee

SELECT DISTINCT *
FROM   #Employee
```

ب) استفاده از یک جدول موقتی دیگر بر اساس یک مجموعه از فیلدهای موجود

```
SELECT DISTINCT * FROM #Employee

SELECT * INTO #DuplicateEmployee FROM #Employee

INSERT #DuplicateEmployee
SELECT ID,
    FIRST_NAME,
    LAST_NAME
```

```
FROM   #Employee
GROUP BY
    ID,
    FIRST_NAME,
    LAST_NAME
HAVING COUNT(*) > 1

BEGIN TRAN
DELETE #Employee
FROM   #DuplicateEmployee
WHERE  #Employee.ID = #DuplicateEmployee.ID
   AND #Employee.FIRST_NAME = #DuplicateEmployee.FIRST_NAME
   AND #Employee.LAST_NAME = #DuplicateEmployee.LAST_NAME

INSERT #Employee
SELECT *
FROM   #DuplicateEmployee

COMMIT TRAN
DROP TABLE #DuplicateEmployee

SELECT DISTINCT * FROM   #Employee
```

ج) استفاده از rowcount بر اساس یک مجموعه از فیلدهای موجود

```
SELECT DISTINCT *
FROM   #Employee

SET ROWCOUNT 1
SELECT 1
WHILE @@rowcount > 0
   DELETE #Employee
   WHERE  1 < (
               SELECT COUNT(*)
               FROM   #Employee a2
               WHERE  #Employee.ID = a2.ID
                      AND #Employee.FIRST_NAME = a2.FIRST_NAME
                      AND #Employee.LAST_NAME = a2.LAST_NAME
          )

SET ROWCOUNT 0

SELECT DISTINCT *
FROM   #Employee
```

د) استفاده از Analytical Functions بر اساس یک مجموعه از فیلدهای موجود

```
SELECT DISTINCT *
FROM   #Employee;

WITH #DeleteEmployee AS (
                         SELECT ROW_NUMBER()
                                OVER(PARTITION BY ID, First_Name, Last_Name ORDER BY ID) AS
                                RNUM
                         FROM   #Employee
                       )

DELETE
FROM   #DeleteEmployee
WHERE  RNUM > 1

SELECT DISTINCT *
FROM   #Employee
```

ه) استفاده از یک فیلد identity جدید بر اساس یک مجموعه از فیلدهای موجود

```
SELECT DISTINCT *
```

```
FROM   #Employee;

ALTER TABLE #Employee ADD UNIQ_ID INT IDENTITY(1, 1)

DELETE
FROM   #Employee
WHERE  UNIQ_ID < (
        SELECT MAX(UNIQ_ID)
        FROM   #Employee a2
        WHERE  #Employee.ID = a2.ID
               AND #Employee.FIRST_NAME = a2.FIRST_NAME
               AND #Employee.LAST_NAME = a2.LAST_NAME
    )

ALTER TABLE #Employee DROP COLUMN UNIQ_ID

SELECT DISTINCT *
FROM   #Employee
```

و در آخر

```
DROP TABLE #Employee
```

نظرات خوانندگان

نویسنده: salarblog
تاریخ: ۱۳۸۸/۰۵/۲۸ ۱۵:۵۶:۵۹

بسیار عالی، تشکر.

نویسنده: مهران زند
تاریخ: ۱۳۸۸/۰۵/۳۱ ۰۹:۱۱:۴۶

جناب نصیری من دقیقا کد های شما را نخواندمو اما من از تیکه کد زیر برای حدف رکورد های تکراری استفاده میکنم :

```
DELETE
FROM MyTable
WHERE ID NOT IN
)
(SELECT MAX(ID
FROM MyTable
GROUP BY COLUMN1, COLUMN2
```

که در اینجا ستون ها ستون هایی هستند که امکان یونیک بودن داده ها در آن وجود دارد.

با سپاس

عنوان: **ذخیره سازی فایل‌ها در دیتابیس یا استفاده از فایل سیستم متداول؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۱۰ ۲۰:۴۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

اگر به ساز و کار شیرپوینت مایکروسافت دقت کنید، همه چیز را داخل دیتابیس ذخیره می‌کند (از اطلاعات رکوردها گرفته تا فایل‌ها و غیره). حال شاید این سؤال مطرح شود که برای ذخیره سازی فایل‌هایی با تعداد بیش از یک میلیون عدد، استفاده از دیتابیس مناسب است یا فایل سیستم متداول. برای پاسخ به این سؤال باید به نکات ذیل توجه داشت:

- هر نوع عملیاتی که بر روی فایل‌ها صورت گیرد، بستن، بازکردن و غیره، نیازمند اعمالی در سطح سیستم عامل است (برای مثال بررسی سطح دسترسی لازم برای انجام این‌کارها).
- هر گونه عملیاتی بر روی فایل‌ها نیازمند یک حداقل قفل گذاری بر روی آن‌ها است که این نیز مصرف CPU قابل توجهی را سبب خواهد شد.
- تمامی اعمال ذکر شده کل سرور و تمامی سرویس‌های در حال اجرا را تحت تاثیر قرار داده و بازدهی آن‌هارا کاهش می‌دهند.
- حتی سیستم عامل‌ها نیز از یک file system database جهت مدیریت اعمال خود استفاده می‌کنند اما این روش برای مدیریت میلیون‌ها و میلیاردها فایل بهینه سازی نشده است.
- ذخیره سازی میلیون‌ها و میلیاردها فایل به تدریج سبب ایجاد fragmentation قابل توجهی شده و این مورد نیز بر روی کارآیی تاثیر منفی خواهد گذاشت (همچنین این مورد بر روی طول عمر تجهیزات ذخیره سازی داده‌ها تاثیر منفی دارند).
- تهیه پشتیبان و بازگرداندن میلیون‌ها فایل بسیار زمانگیر است (برای مثال جابجایی یک فایل یک مگابایتی بسیار سریعتر است از جابجایی 100 فایل 10 کیلوبایتی).
- مدیریت تغییرات و همچنین بررسی اینکه چه شخصی چه فایلی را قرار داده، حذف کرده یا تغییر داده است در حالت استفاده از file system مشکل است.
- به صورت پیش فرض عموما مباحث replication و امثال آن توسط روش استفاده از file system خصوصا با تعداد بالای فایل، پشتیبانی نمی‌شود.
- در حالت استفاده از file system ، برنامه‌های وب باید دسترسی write بر روی یک سری پوشه داشته باشند که این مورد همیشه از دیدگاه امنیتی مساله ساز بوده و مشکل آفرین.
- کرش file system مساوی است با کرش سیستم عامل و بازگشت این‌ها زمان‌بر خواهد بود.

با توجه به این نکات استفاده از دیتابیس برای ذخیره سازی تعداد زیادی فایل، مزایای زیر را به همراه خواهد داشت:

- اکثر سیستم‌های دیتابیسی امروزی برای کار با حجم عظیمی از داده‌ها به حد بلوغ خود رسیده‌اند.
- هنگام استفاده از دیتابیس برای ذخیره سازی فایل‌ها دیگر سر و کار ما با میلیون‌ها فایل نخواهد بود و حداکثر چند فایل دیتابیس و ملحقات آن مانند لاگ فایل، کل سیستم را تشکیل می‌دهند.
- فایل‌های دیتابیس برای مثال SQL Server ، همیشه توسط SQL Server در حالت باز قرار داشته و مباحث قفل‌گذاری بر روی فایل‌های دیتابیس و بررسی سطح دسترسی و غیره توسط سیستم عامل در این‌جا به حداقل خود می‌رسد.
- در این حالت بار سیستم عامل شما تنها سیستمی است که مشغول سرویس دهی اطلاعات دیتابیس‌های شما است.
- جستجوی فایل‌ها، حتی جستجو در محتوای این فایل‌های ذخیره شده در یک دیتابیس بسیار سریعتر از روش file system می‌باشد. امکان استفاده از کوئری‌های SQL انعطاف پذیری خاصی را به این سیستم‌ها خواهند داد (برای مثال قابلیت full text search مربوط به SQL server امکان جستجو بر روی رکوردهایی با محتوای pdf را نیز پس از انجام اندکی تنظیمات، دارا می‌باشد).
- هنگام کار با دیتابیس مباحث تراکنشی نقش بسیار حائز اهمیتی را بازی می‌کنند اما عموما سیستم عامل‌ها در این زمینه نیازمند کار و برنامه نویسی قابل توجهی هستند (این قابلیت به ویندوز ویستا اضافه شده است).
- کرش یک دیتابیس عموما سبب کرش سیستم عامل یا حتی کرش سایر دیتابیس‌های موجود نخواهد شد.
- امکان تهیه پشتیبان از دیتابیس‌ها و بازیابی آن‌ها ساده است. (حداقل از بازیابی میلیون‌ها فایل ساده‌تر است)
- امکانات replication به صورت پیش فرض در اکثر سیستم‌های دیتابیسی امروزی مهیا است.

- امکان ثبت وقایع و مدیریت اطلاعات افزوده شده به دیتابیس، از طریق نرم افزارهایی که برای این کار نوشته خواهند شد (یا حتی امکانات توکار این برنامه‌ها) از هر لحاظ نسبت به روش file system برتری دارد.
- امکانات سوئیچ کردن به دیتابیسی دیگر در شبکه در صورت کرش یک نود، مهیا است و پیش بینی شده است.
- برای استفاده از یک دیتابیس توسط یک برنامه وب، نیازی به داشتن دسترسی write بر روی هیچ فولدری وجود ندارد که این خود یک مزیت امنیتی مهم است و همچنین امکان محدود کردن سطوح دسترسی به فایل‌های ذخیره شده در دیتابیس با برنامه‌های نوشته شده نیز ساده‌تر است. (البته در این‌جا مسلما منظور از دیتابیس، دیتابیس Access نیست و SQL Server یا MySQL مد نظر هستند)

نظرات خوانندگان

نویسنده: salarblog
تاریخ: ۱۳۸۸/۰۶/۱۰ ۲۲:۱۶:۳۴

مطلب جالبی بود.
این سوال برای من هم پیش اومده بود که چرا از SQL Server برای Team Foundation Server استفاده کردن، درحالی که این حجم زیاد تراکنش و فایل وجود داره. ولی باید اقرار کنم که خیلی سریع تر SourceControl2005 هست.

با این حال استفاده از دیتابیش مسئله قابلیت حمل رو پیش میاره.

نویسنده: macromediax
تاریخ: ۱۳۸۸/۰۶/۱۹ ۱۷:۱۲:۵۳

اما معایب استفاده از دیتابیس برای فایل ها رو ننوشتی که فکر میکنم در انتهای مقاله این مورد رو هم باید بررسی میکردی . مثلا فرض کن یک سرویس نوشتی که فایل های آهنگ و عکس های کاربرانشو میگیره و باید ذخیره کنه و چندین بار پخش کنه یا نمایش بده . مسلما نمایش هر عکس اگر بخواد از دیتابیس لود بشه، فشار زیادی را روی dbms میذاره .. بخصوص وقتی تعداد عکس ها و درخواست نمایش بالاست ..

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۱۹ ۱۷:۴۷:۴۰

مقاله کلا در این مورد بود که اگر تعداد عکس‌ها بالا باشد روش سنتی ذخیره سازی روی هارد به دلایلی که ذکر شده مشکل زا خواهد شد و اس کیوال سرور اینقدر توانمند و پخته است که بتواند این مساله را حل و فصل کند. (نمونه استفاده در مقیاس سازمانی از آن شیرپوینت است که همه اطلاعات را داخل دیتابیس قرار داده و سپس لود می‌کند)
یا برای نمونه YouTube نیز تمام ویدیوهای خودش را درون دیتابیس ذخیره کرده و صد البته برای مقیاس پذیری آن روش‌های کش کردن اطلاعات پیشرفته‌ای را نیز توسعه داده‌اند.
شیرپوینت هم برای این منظور چیزی به نام BlobCache دارد.

نویسنده: Saeed
تاریخ: ۱۳۸۹/۰۳/۲۲ ۱۷:۰۳:۴۸

خیلی ممنون از مطلب خوبتون.
خیلی وقت بود که به دنبال جواب این سوال بودم که برای ذخیره کردن فایلها در سمت سرور از کدوم روش استفاده کنم. البته به این نتیجه رسیده بودم که فایلها رو در بانک اطلاعاتی ذخیره کنم. بعد از انجام این کار متوجه شدم که برای ذخیره کردن حدود 20 مگابایت عکس در سمت سرور ، به اندازه فایلهای بانک اطلاعاتی من در حدود 80 مگابایت افزوده می شود. که این امر من رو برای ذخیره و نگهداری حجم و تعداد زیاد فایل نگران کرده بود.

عنوان: **به روز رسانی فیلدهای XML در SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۱۳ ۱۸:۱۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

از SQL server 2005 به بعد، پشتیبانی کاملی از XML توسط این محصول صورت می‌گیرد. در ادامه مروری خواهیم داشت بر نحوه‌ی به روز رسانی مقادیر فیلدهایی از نوع XML در SQL Server .
در ابتدا جدول موقتی زیر را که شامل یک رکورد از نوع XML است، در نظر بگیرید:

```
DECLARE @tblTest AS TABLE (xmlField XML)

INSERT INTO @tblTest
 (
   xmlField
 )
VALUES
 (
   '<Sample>
       <Node1>Value1</Node1>
       <Node2>Value2</Node2>
       <Node3>OldValue</Node3>
  </Sample>'
 )
```

می‌خواهیم OldValue را به مقداری دیگر تغییر دهیم.

سعی اول:

```
DECLARE @newValue VARCHAR(50)
SELECT @newValue = 'NewValue'
UPDATE @tblTest
SET    xmlField.modify('replace value of (/Sample/Node3)[1] with ' + @newValue)
```

این سعی با خطای زیر متوقف می‌شود:

```
The argument 1 of the XML data type method "modify" must be a string literal.
```

بنابراین از روش String concatenation برای معرفی مقدار متغیر مورد نظر در اینجا نمی‌شود استفاده کرد.

سعی دوم:

```
DECLARE @newValue VARCHAR(50)
SELECT @newValue = 'NewValue'

UPDATE @tblTest
SET    xmlField.modify(
           'replace value of (/Sample/Node3)[1] with sql:variable("@newValue")'
       )
```

روش معرفی صحیح یک متغیر را در اینجا می‌توان مشاهده کرد. اما این سعی نیز با خطای زیر متوقف می‌شود:

```
XQuery [@tblTest.xmlField.modify()]: The target of 'replace value of' must be a non-metadata attribute
or an element with simple typed content, found 'element(NodeThree,xdt:untyped) ?'
```

سعی سوم:

```
DECLARE @newValue VARCHAR(50)
SELECT @newValue = 'NewValue'

UPDATE @tblTest
SET    xmlField.modify(
          'replace value of (/Sample/Node3/text())[1]
           with sql:variable("@newValue")'
      )

SELECT xmlField.value('(/Sample/Node3)[1]','varchar(50)') FROM @tblTest
```

و بله. کار می‌کنه!

XML ایی را که در ابتدا استفاده کردیم از نوع un-typed XML محسوب شده و هیچ schema ایی را برای آن در نظر نگرفته‌ایم، به همین جهت باید دقیقا مشخص کنیم که قصد داریم text این node را ویرایش نمائیم.

مشکل بعدی!

در ابتدا مثال زیر را در نظر بگیرید:

```
DECLARE @tblTest AS TABLE (xmlField XML)

INSERT INTO @tblTest
 (
   xmlField
 )
VALUES
 (
   '<Sample>
       <Node1>Value1</Node1>
       <Node2>Value2</Node2>
       <Node3></Node3>
  </Sample>'
 )

DECLARE @newValue VARCHAR(50)
SELECT @newValue = 'NewValue'

UPDATE @tblTest
SET    xmlField.modify(
          'replace value of (/Sample/Node3/text())[1]
           with sql:variable("@newValue")'
      )

SELECT xmlField.value('(/Sample/Node3)[1]','varchar(50)') FROM @tblTest
```

این عبارات T-SQL ، خلاصه بحث ما تا به اینجا هستند اما با یک تفاوت. نود 3 در اینجا خالی است.

اگر اسکریپت را اجرا کنید، هیچ تغییری را مشاهده نخواهید کرد. به عبارت دیگر به روز رسانی صورت نمی‌گیرد. در اینجا چون text این نود خالی است ، فرض SQL Server بر این خواهد بود که وجود ندارد، بنابراین این نود را به روز رسانی نخواهد کرد. به همین منظور باید برای به روز رسانی این نود، عبارت جدید را در جایی که text ندارد insert کرد (و نه replace).

```
DECLARE @newValue VARCHAR(50)
SELECT @newValue = 'NewValue'

UPDATE @tblTest
SET    xmlField.modify(
          'replace value of (/Sample/Node3/text())[1]
           with sql:variable("@newValue")'
      )

UPDATE @tblTest
SET    xmlField.modify(
          'insert text{sql:variable("@newValue")} into
           (/Sample/Node3)[1] [not(text())]'
      )
```

```
SELECT xmlField.value('(/Sample/Node3)[1]','varchar(50)') FROM @tblTest
```

نظرات خوانندگان

نویسنده: Alex's Blog
تاریخ: ۱۳۸۸/۰۶/۲۲ ۰۸:۰۸:۵۸

سلام
عالی بود. ممنون

عنوان: **Optimize for unknown**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۲۱ ۲۱:۰۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

مفهومی در SQL Server وجود دارد به نام parameter sniffing که شرح آن به صورت زیر است.
ابتدا رویه ذخیره شده زیر را در نظر بگیرید:

```
create procedure test (@pid int)
as
select * from Sales.SalesOrderDetail
where ProductID = @pid
```

استفاده از کوئری‌های پارامتری یکی از بهترین تمرین‌های کاری با SQL server است؛ از آنجائیکه در این حالت plan تهیه شده مجددا مورد استفاده قرار گرفته، همچنین از SQL injection نیز جلوگیری خواهد کرد، زیرا برای نمونه در مثال فوق تنها pid از نوع int پذیرفته می‌شود و نه هر ورودی خطرناک دیگری.
اما این نوع کوئری‌ها یک مشکل را نیز به همراه خود دارند. این plan تهیه شده به ازای اولین ورودی رویه ذخیره شده تهیه می‌شود (parameter sniffing) و الزامی ندارد که برای دومین ورودی و فراخوانی‌های بعدی، بهترین plan باشد.

برای حل این مشکل راه‌های زیادی هست:
الف) انتساب پارامترهای یک رویه ذخیره شده به متغیری محلی

```
create procedure test (@pid int)
as
Declare @mpid int
Set @mpid = @pid
select * from Sales.SalesOrderDetail
where ProductID = @mpid
```

در اینجا پارامتر ورودی مستقیما در کوئری استفاده نشده است و SQL Server این متغیر محلی را sniff نخواهد کرد.

ب) استفاده از گزینه RECOMPILE که سبب خواهد شد به ازای هر ورودی یک plan بهینه تهیه شود. این مورد مصرف CPU بالایی را به همراه خواهد داشت.

ج) راه حل ارائه شده در SQL Server 2005
استفاده از روش الف به علاوه اضافه کردن گزینه کمکی زیر به انتهای اسکریپت فوق

```
OPTION (OPTIMIZE FOR(@pid = 544))
```

در اینجا فرض بر این است که می‌دانیم pid=544 بسیار مورد استفاده قرار خواهد گرفت، بنابراین این معرفی را به موتور بهینه ساز SQL Server ارائه خواهیم کرد.

د) راه حل ارائه شده در SQL Server 2008
با استفاده از Optimize for unknown که در اس کیوال سرور 2008 معرفی شده است، مزیت استفاده از کوئری‌های پارامتری همانند استفاده مجدد از plan تهیه شده، حفظ گشته اما این plan تهیه شده اولیه بر اساس اولین مقدار پاس شده، تهیه نگردیده و حالت عمومی‌تر و بهینه‌تری را برای اکثر مقادیر پاس شده خواهد داشت.

```
create procedure test (@pid int)
```

Optimize for unknown

```
as
select * from Sales.SalesOrderDetail
where ProductID = @pid

OPTION(OPTIMIZE FOR (@pid UNKNOWN))
```

جهت مطالعه بیشتر (+ و + و +)

نظرات خوانندگان

نویسنده: Amin
تاریخ: ۱۳۸۹/۰۲/۰۵ ۰۹:۳۳:۳۶

این مورد خیلی منو اذیت کرده بود که حلش کرده بودم اما امروز این پستتون رو دیدم!

عنوان: FxCop برای SQL Server
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۲۴ ۲۲:۱۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

حتما با FxCop که برای آنالیز اسمبلی‌های برنامه‌های دات نتی بکار می‌رود آشنایی دارید. شبیه به این مورد به صورت افزونه‌ای برای Visual studio 2008 team system نیز موجود است. فقط کافی است Microsoft® Visual Studio Team System 2008 Database Edition GDR R2 را نصب کرده و یک پروژه دیتابیس جدید را شروع کنید (نوع database wizard که یک دیتابیس کامل را import می‌کند). سپس در برگه build تیک مربوط به code analysis را قرار دهید (شکل 1) و یکبار پروژه را build کنید. به این صورت در پنجره خروجی، اشکالات کدهای T-SQL شما گوشزد می‌شود (شکل 3). این‌کار را با استفاده از منوی Data نیز می‌توان انجام داد (شکل 2).

شکل 1- فعال سازی تحلیل و بررسی کد

شکل 2- اجرای تحلیل و بررسی کد

شکل 3- یک نمونه خروجی حاصل از تحلیل و بررسی کد

عنوان: آشنایی با قابلیت FileStream اس کیوال سرور 2008 - قسمت اول
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۰۴ ۱۱:۳۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

مطلبی چندی قبل در مورد " ذخیره سازی فایل‌ها در دیتابیس یا استفاده از فایل سیستم متداول؟ " منتشر گردید، جهت برشمردن فواید ذخیره سازی فایل‌ها در دیتابیس (+). اما معایب این نوع ذخیره سازی بررسی نشدند:

الف) اختصاص یافتن قسمتی از بافر SQL Server به این امر.
ب) با توجه به قرار گرفتن داده‌های BLOB در دیتابیس ، transaction log قابل توجهی تولید خواهد شد. (+)
ج) بیش از 2GB را 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.

FILESTREAM موتور 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. FILESTREAM data باید در 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 2GB فیلدهایی 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 BLOBs varbinary(max) کارآیی 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 256 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.

FileStream از 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 API میسر 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

FileStream به 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.

ادامه دارد ...

نظرات خوانندگان

نویسنده: fahimi
تاریخ: ۱۳۹۱/۰۶/۰۷ ۲۲:۲

سلام
از آموزش بسیار عالی و کاربردی جنابعالی کمال تشکر را دارم
من آموزش شما را در delphi & C# پیاده کردم و به خوبی اجرا شد
سوال:آیا دستورات Update , Delete
علاوه بر تیبل فایل‌ها را هم تاثیر دار
به هر حال من دستور delete را اجرا کردم در تیبل رکورد حذف شد ولی فایل مرتبط با رکورد حذف نشد
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۰۷ ۲۲:۱۰

همینطوره. این فایل‌ها سر فرصت توسط سیستم FILESTREAM garbage collection حذف خواهند شد.

نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۱/۰۶/۰۷ ۲۲:۵۷

سلام؛
اگر از Entity Framework در حالت CodeFirst استفاده نماییم میتوان از قابلیت‌های FileStream در آن استفاده کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۰۷ ۲۳:۴۴

بله. خاصیت مورد نظر را به صورت byte array تعریف کنید:

```
public class Image
{
    public int Id { get; set; }
    public byte[] Data { get; set; }
}
```

عنوان: **آشنایی با قابلیت FileStream اس کیوال سرور 2008 - قسمت دوم**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۰۵ ۱۲:۴۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

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.

ادامه دارد ...

نظرات خوانندگان

نویسنده: Yahya
تاریخ: ۱۳۸۸/۰۹/۰۹ ۱۱:۲۲:۲۷

از اینکه به این خوبی توضیح داده اید بسیار سپاسگذارم

عنوان: آشنایی با قابلیت FileStream اس کیوال سرور 2008 - قسمت سوم
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۰۶ ۰۸:۴۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

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 FileId را 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-52C3509407D4");
            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 2008 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 می‌گرداند.
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 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())
                {
                    //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 2008 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 2008 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;
        }
    }
}
```

در پایان برای تکمیل بحث می‌توان به مقاله‌ی مرجع زیر مراجعه کرد:

FILESTREAM Storage in SQL Server 2008

نظرات خوانندگان

نویسنده: Yahya
تاریخ: ۱۳۸۸/۰۹/۰۹ ۱۱:۲۲:۰۳

تشکر فراوان از مطلب خوب شما

نویسنده: حمید
تاریخ: ۱۳۹۱/۰۵/۰۶ ۲۰:۱۳

سلام
ممنون از مطلب مفیدتون
یک سؤالی که برام پیش اومده اینه که ، این فایل‌ها که در مسیر مشخص شده ذخیره شدن ، موندگار هستن یا موقتی اند ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۶ ۲۰:۱۵

ماندگار هستند.

نویسنده: حمید
تاریخ: ۱۳۹۱/۰۵/۰۶ ۲۱:۹

خب ، اگر این فایل‌ها به هر دلیلی حذف بشن یا ویروسی بشن روی دیتا اصلی تأثیری داره ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۶ ۲۱:۵۲

دیتای اصلی همین فایل‌ها هستند. نحوه دسترسی به آن‌ها اینبار از طریق استریم‌ها است که سربار کمی دارند نسبت به حالت معمولی که کل فایل، داخل بانک اطلاعاتی ذخیره می‌شود و برای دریافت آن buffer pool موتور بانک اطلاعاتی و حافظه سیستم کاملا درگیر و مصرف می‌شوند. در دو قسمت قبل این بحث فایل استریم به مباحث تئوری آن پرداخته شده. لطفا آن‌ها را مطالعه کنید.

عنوان: **با رویه‌های ذخیره شده خود، وب سرویس ایجاد کنید**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۱۲ ۲۰:۱۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

قابلیت جالبی از SQL Server 2005 به بعد به این محصول اضافه شده است که امکان ایجاد یک وب سرویس بومی را بر اساس رویه‌های ذخیره شده و یا توابع تعریف شده در دیتابیس‌های موجود، فراهم می‌سازد. این قابلیت نیازی به IIS یا هر هاست دیگری برای اجرا ندارد و توسط خود اس کیوال سرور راه اندازی و مدیریت می‌شود.
توضیحات مفصل آن‌را در MSDN می‌توانید ملاحظه کنید و در اینجا یک مثال عملی از آن را با هم مرور خواهیم کرد:

الف) ایجاد یک جدول آزمایشی به همراه تعدادی رکورد دلخواه در آن

```
CREATE TABLE [tblWSTest](
 [id] [int] IDENTITY(1,1) NOT NULL,
 [f1] [nvarchar](50) NULL,
 [f2] [nvarchar](500) NULL,
CONSTRAINT [PK_tblWSTest] PRIMARY KEY CLUSTERED
(
 [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON,
ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]

SET IDENTITY_INSERT [tblWSTest] ON
INSERT [tblWSTest] ([id], [f1], [f2]) VALUES (1, N'a1', N'a2')
INSERT [tblWSTest] ([id], [f1], [f2]) VALUES (2, N'b1', N'b2')
INSERT [tblWSTest] ([id], [f1], [f2]) VALUES (3, N'c1', N'c2')
INSERT [tblWSTest] ([id], [f1], [f2]) VALUES (4, N'd1', N'd2')
INSERT [tblWSTest] ([id], [f1], [f2]) VALUES (5, N'e1', N'e2')
SET IDENTITY_INSERT [dbo].[tblWSTest] OFF
```

ب) ایجاد یک رویه ذخیره شده در دیتابیس جاری

```
CREATE PROCEDURE GetAllData
AS
 SELECT f1,
        f2
 FROM   tblWSTest
```

ج) ایجاد یک HTTP Endpoint

```
CREATE ENDPOINT GetDataService
   STATE = STARTED
AS HTTP(
          PATH = '/GetData',
          AUTHENTICATION = (INTEGRATED),
          PORTS = (CLEAR),
          CLEAR_PORT = 8080,
          SITE = '*'
      )
FOR SOAP(
           WEBMETHOD 'GetAllData'
           (NAME = 'testdb2009.dbo.GetAllData'),
           WSDL = DEFAULT,
           DATABASE = 'testdb2009',
           NAMESPACE = DEFAULT
       )
```

توضیحات:
Ports در حالت clear و یا ssl می‌تواند باشد. همچنین برای اینکه با IIS موجود بر روی سیستم هم تداخل نکند CLEAR_PORT به

8080 تنظیم شده است. سایر پارامترهای آن بسیار واضح هستند. برای مثال تعیین دیتابیسی که این رویه ذخیره شده در آن قرار دارد و همچنین مسیر کامل دسترسی به آن دقیقا مشخص می‌گردند.

این وب سرویس هم اکنون آغاز به کار کرده است. برای مشاهده wsdl آن، آدرس زیر را در مرورگر وب خود وارد نمائید (PATH و CLEAR_PORT معرفی شده در endPoint اینجا بکار می‌رود):

```
http://localhost:8080/GetData?wsdl
```

د) استفاده از این وب سرویس در یک برنامه ویندوزی

یک برنامه ساده winForms را شروع کنید. سپس یک DataGridView را بر روی فرم قرار دهید (بدیهی است این مورد می‌تواند یک برنامه ASP.Net هم باشد و موارد مشابه دیگر). سپس از منوی پروژه، یک service reference را در VS2008 بر اساس آدرس wdsl فوق اضافه کنید (شکل زیر):

با رویه‌های ذخیره شده خود، وب سرویس ایجاد کنید

برای اینکه این مثال در VS2008 درست کار کند باید فایل app.config ایجاد شده را کمی ویرایش کرد. قسمت security آن را یافته و تغییرات زیر را با توجه به AUTHENTICATION مورد نیاز تغییر دهید:

```
<security mode="TransportCredentialOnly">
          <transport clientCredentialType="Windows" proxyCredentialType="None"
              realm="" />
          <message clientCredentialType="UserName" algorithmSuite="Default" />
        </security>
```

سپس کد برنامه ما به صورت زیر خواهد بود:

```
using System;
using System.Data;
using System.Windows.Forms;

namespace WebServiceTest
{
   public partial class Form1 : Form
   {
       public Form1()
       {
           InitializeComponent();
       }
```

```
        private void Form1_Load(object sender, EventArgs e)
        {
            ServiceReference1.GetDataServiceSoapClient data =
                new ServiceReference1.GetDataServiceSoapClient();
            dataGridView1.DataSource = (data.GetAllData()[0] as DataSet).Tables[0];
        }
    }
}
```

نظرات خوانندگان

نویسنده: DotNetCoders
تاریخ: ۱۳۸۸/۰۷/۱۲ ۲۲:۱۵:۵۳

خیلی جالب بود و بسیار کارا !

فقط سوال اینجاست که میشه نوع خروجی رو مشخص کرد ؟

مثلا Json ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۱۲ ۲۲:۴۰:۲۳

سلام
نه مسلما در حد WCF service های کامل نمی‌تونید ازش انتظار داشته باشید و خروجی فقط SOAP messages است.
البته می‌تونید FORMAT = NONE را در تعریف end point لحاظ کنید و بعد خروجی را آن طور که دوست دارید فرمت کرده و ارائه دهید. (باید کد نویسی کنید)

ولی کلا در این حد که نیاز به یک هاست را برطرف می‌کند و خیلی سریع بدون کد نویسی آنچنانی یک وب سرویس در اختیار شما خواهد بود، واقعا کارآمد است.

نویسنده: Alex's Blog
تاریخ: ۱۳۸۸/۰۷/۱۶ ۱۴:۱۹:۵۴

واقعا عالیه. خیلی بدرد بخوره.
ما اگه شما رو نداشتیم چیکار می کردیم :)

عنوان: **به روز رسانی فیلدهای XML در SQL Server - قسمت دوم**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۳۰ ۰۰:۰۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

قسمت اول را در این آدرس می‌توانید مطالعه نمائید.

در ادامه قسمت اول، اگر بخواهیم نود جدیدی را به فیلد XML موجود اضافه کنیم، روش انجام آن به صورت زیر است (یکی از روش‌ها البته):

```
DECLARE @tblTest AS TABLE (xmlField XML)

INSERT INTO @tblTest
(
 xmlField
)
VALUES
(
 '<Sample>
    <Node1>Value1</Node1>
    <Node2>Value2</Node2>
    <Node3>OldValue</Node3>
</Sample>'
)

DECLARE @Name NVARCHAR(50)
SELECT @Name = 'Vahid'

UPDATE @tblTest
SET    xmlField.modify(
        'insert element Node4 {sql:variable("@Name")} as last into
 (/Sample)[1]'
    )

SELECT tt.xmlField
FROM   @tblTest tt
```

که حاصل آن (افزوده شدن یک المان جدید به نام Node4 بر اساس مقدار متغیر Name در انتهای لیست) به صورت زیر می‌باشد:

```
<Sample>
<Node1>Value1</Node1>
<Node2>Value2</Node2>
<Node3>OldValue</Node3>
<Node4>Vahid</Node4>
</Sample>
```

سؤال 1 :
اگر بخواهیم نام Node4 نیز متغیر باشد به چه صورتی باید مساله را حل کرد؟
در این حالت باید از کوئری‌های داینامیک استفاده کرد. باید یک رشته را ایجاد (کل عبارت update باید یک رشته شود) و سپس از دستور exec کمک گرفت و البته باید دقت داشت در این حالت کار encoding کارکترهای غیرمجاز در XML را باید خودمان انجام دهیم.

سؤال 2:
اگر بخواهیم نام نودها و مقادیر آن‌ها را به صورت یک جدول معمولی بازگشت دهیم به چه صورتی باید عمل کرد؟

```
DECLARE @XML AS XML
```

```
SELECT @XML = tt.xmlField
FROM   @tblTest tt

SELECT t.c.value('local-name(..)', 'varchar(max)') AS ParentNodeName,
    t.c.value('local-name(.)', 'varchar(max)') AS NodeName,
    t.c.value('text()[1]', 'varchar(max)') AS NodeText

FROM   @XML.nodes('/*/*') AS t(c)
```

که پس از اجرای آن خواهیم داشت:

```
ParentNodeName - NodeName - NodeText
Sample           Node1     Value1
Sample           Node2     Value2
Sample           Node3     OldValue
Sample           Node4     Vahid
```

عنوان: **مرجعی در مورد نگارش‌های مختلف SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۸/۱۳ ۱۶:۳۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

آیا می‌دانید که تا این تاریخ پس از ارائه سرویس پک یک اس کیوال سرور 2008، دقیقا 5 به روز رسانی دیگر نیز در مورد آن منتشر شده‌اند؟
آیا می‌دانید پس از ارائه سرویس پک سه مربوط به اس کیوال سرور 2005 ، دقیقا 10 مورد به روز رسانی دیگر آن نیز منتشر گردیده‌اند؟
آیا می‌دانید پس از سرویس پک 4 اس کیوال سرور 2000 دقیقا چند مورد به روز رسانی مرتبط با آن منتشر شده‌اند؟ (72 مورد!)
آیا می‌دانید دقیقا از چه نگارشی از SQL Server با کدام به روز رسانی‌ها استفاده می‌کنید؟

پاسخ دقیق به تمام این سؤالات را به صورت طبقه بندی شده و بسیار منظم، در وبلاگ زیر می‌توانید مشاهده نمائید:

http://sqlserverbuilds.blogspot.com

نظرات خوانندگان

نویسنده: Kianoosh
تاریخ: ۱۳۸۸/۰۸/۱۶ ۰۰:۳۳:۲۸

باتشکر از شما آقای مهندس نصیری

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۸/۱۵ ۱۳:۳۳:۵۷

این سایت اطلاعات نگارش‌های مختلف را نگهداری می‌کند: (+)

عنوان: **یافتن تداخلات Collations در SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۸/۲۸ ۱۶:۲۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

اگر دیتابیس خود را در طی چند سال از یک نگارش به نگارشی دیگر یا از یک سرور به سروری دیگر منتقل کرده باشید، به احتمال زیاد به مشکلات Collations هم برخورده‌اید. یکی از فیلدها Arabic _CI_AS است (بجا مانده از دوران قبل از SQL Server 2008) در یک جدول و در جدولی دیگر فیلدی تازه‌ای با Collation از نوع Persian _100_CI_AS تعریف شده است. Collations نحوه ذخیره سازی و مقایسه رشته‌ها را کنترل می‌کنند. زمانیکه یک جدول جدید را در SQL Server ایجاد می‌کنیم، اگر Collation فیلدها به صورت صریح ذکر نگردند، بر مبنای همان Collation پیش فرض دیتابیس تعریف خواهند شد.
بنابراین اگر پس از استفاده از SQL Server 2008 و تنظیم Collation پیش فرض دیتابیس به Persian_100_CI_AS ، به این موارد دقت نکنیم، دیر یا زود دچار مشکل خواهیم شد.
عملیات مرتب سازی با وجود تداخلات Collations مشکل ساز نمی‌شود (خطایی دریافت نمی‌کنید)، اما ممکن است الزاما صحیح عمل نکند. مشکل از آنجایی آغاز می‌شود که قصد داشته باشیم داده‌ها را مقایسه کنیم یا join ایی بین این دو جدول با فیلدهای ناهمگون از لحاظ Collation ایجاد نمائیم. در این حالت حتما خطاهای تداخل Collation را دریافت کرده و کوئری‌های ما اجرا نخواهند شد.

Cannot resolve collation conflict for equal to operation

یک راه حل این است که در حین join به صورت صریح collation هر دو فیلد ذکر شده را به صورت یکسان ذکر کنیم که بیشتر یک مرهم موقتی است تا راه حل اصولی. برای مثال:

```
SELECT ID
FROM ItemsTable
INNER JOIN AccountsTable
WHERE ItemsTable.Collation1Col COLLATE DATABASE_DEFAULT
= AccountsTable.Collation2Col COLLATE DATABASE_DEFAULT
```

راه دیگر این است که مشخص کنیم که Collation کدام فیلدها در دیتابیس با Collation پیش فرض دیتابیس تطابق ندارند. سپس بر اساس این لیست شروع به تغییر Collations نمائیم.
اسکریپت زیر تمام فیلدهای ناهمگون از لحاظ Collation دیتابیس جاری را برای شما لیست خواهد کرد:

```
DECLARE @defaultCollation NVARCHAR(1000)
SET @defaultCollation = CAST(
      DATABASEPROPERTYEX(DB_NAME(), 'Collation') AS NVARCHAR(1000)
  )

SELECT C.Table_Name,
     Column_Name,
     Collation_Name,
     @defaultCollation DefaultCollation
FROM   Information_Schema.Columns C
     INNER JOIN Information_Schema.Tables T
          ON  C.Table_Name = T.Table_Name
WHERE  T.Table_Type = 'Base Table'
     AND RTRIM(LTRIM(Collation_Name)) <> RTRIM(LTRIM(@defaultCollation))
     AND COLUMNPROPERTY(OBJECT_ID(C.Table_Name), Column_Name, 'IsComputed') = 0
ORDER BY
     C.Table_Name,
     C.Column_Name
```

برای مثال جهت تغییر Collation فیلد Serial از جدول tblArchive از نوع nvarchar با طول 200 به Persian_100_CI_AS می‌توان از دستور T-SQL زیر استفاده کرد:

```
ALTER TABLE [tblArchive] ALTER COLUMN [Serial] nvarchar(200) COLLATE Persian_100_CI_AS not null
```

نظرات خوانندگان

نویسنده: reza
تاریخ: ۱۳۸۸/۰۹/۰۱ ۱۷:۲۲:۱۵

آریان

میشه اینطوری هم انجام داد

```
, SELECT sys.tables.name AS TableName , sys.columns.name AS ColumnName
sys.columns.is_nullable , sys.columns.collation_name
FROM sys.columns
INNER JOIN sys.tables ON sys.columns.object_id=sys.tables.object_id
(((WHERE sys.columns.collation_name<>(CAST(DATABASEPROPERTYEX(DB_NAME() , 'Collation') AS NVARCHAR(1000
```

عنوان: **عدم کاهش حجم لاگ فایل SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۹/۲۶ ۱۳:۴۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

در مورد روش‌های کاهش حجم لاگ فایل‌های SQL Server در این مطلب بحث شد.
اما یکی از دیتابیس‌های قدیمی shrink نمی‌شد و پیغام خطای زیر را صادر می‌کرد:

```
Cannot shrink log file 2 because of minimum log space required.
```

یکی از علت‌هایی که اگر مطابق روش ذکر شده در مقاله یاده شده رفتار شود، سبب کاهش حجم لاگ فایل یک دیتابیس نمی‌شود، وجود تراکنش‌های کامل نشده است. جهت مشاهده‌ی وضعیت تراکنش‌های یک دیتابیس می‌توان دستور زیر را صادر کرد:

```
DBCC OPENTRAN
```

که نتیجه به صورت زیر بود:

```
Replicated Transaction Information:
```

```
Oldest distributed LSN     : (0:0:0)
```

Oldest non-distributed LSN : (5291:25:1)

وجود سطر مربوط به Oldest non-distributed LSN به این معنا است که هنوز یک replication نا تمام بر روی این دیتابیس موجود است. البته چون این دیتابیس از یک سرور دیگر به اینجا منتقل شده بود و هیچ نوع replication ایی هم در این سرور بر روی آن تنظیم نشده بود؛ بنابراین ابتدا این replication حذف شد:

```
exec sp_removedbreplication 'dbName', 'both';
```

سپس مجددا دستور زیر جهت مشاهده‌ی وضعیت تراکنش‌های ناتمام صادر شد:

```
DBCC OPENTRAN
```

که این‌بار دیگر هیچ خروجی نداشت.
اکنون با استفاده از روش ذکر شده، لاگ فایل 70 گیگابایتی این دیتابیس به سادگی به چند مگابایت shrink شد.

نظرات خوانندگان

نویسنده: www.Abshar-System.ir
تاریخ: ۱۳۹۰/۰۱/۲۲ ۱۱:۰۴:۲۶

آقای نصیری تشکر

عنوان: **مقایسه امنیت Oracle11g و SQL server 2008 از دید آمار در سال 2009**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۹/۲۸ ۱۸:۱۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

جدول زیر تعداد باگ‌های امنیتی Oracle11g و SQL server 2008 را تا ماه نوامبر 2009 نمایش می‌دهد:

Vulnerabilities	Advisories	Product
0	0	SQL Server 2008
239	7	Oracle11g

و به صورت خلاصه مایکروسافت در 6 سال گذشته تنها 59 باگ امنیتی وصله شده مربوط به نگارش‌های مختلف SQL Server داشته است (از نگارش 2000 به بعد). در طی همین مدت اوراکل (نگارش‌های 8 تا 10) تعداد 233 وصله امنیتی را ارائه داده است. در سال 2006 ، اس کیوال سرور 2000 با سرویس پک 4 ، به عنوان امن‌ترین بانک اطلاعاتی موجود در بازار شناخته شد (به همراه PostgreSQL). در همین زمان Oracle10g در قعر این جدول قرار گرفت.

اعداد و آمار از سایت secunia.com استخراج شده است: + و +

نظرات خوانندگان

نویسنده: behnam
تاریخ: ۲۳:۵۷:۵۶ ۱۳۸۸/۰۹/۲۸

سایت secunia.com تو این زمینه معتبره؟
من تا امروز بر عکس این جریان فکر می کردم!

نویسنده: وحید نصیری
تاریخ: ۰۹:۱۸:۴۳ ۱۳۸۸/۰۹/۲۹

بله. سایت بی طرفی است و یکی از کارهایش جمع کردن آمارهای اینگونه است.

نویسنده: سار
تاریخ: ۰۹:۵۴:۱۲ ۱۳۸۸/۰۹/۲۹

مدیر بخش نرم افزار ما که یک پزشک هست اگر این رو ببینه سکته می کنه :D

نویسنده: وحید نصیری
تاریخ: ۱۳:۱۱:۴۰ ۱۳۸۸/۰۹/۲۹

کم شدن باگ‌های امنیتی مایکروسافت در چند سال اخیر سه دلیل عمده داشته:
- استفاده بیشتر از دات نت فریم ورک در محصولات عمده خودش (اس کیوال سرور، SharePoint، exchange server و ...).
برای مثال دات نت فریم ورک به حملات سرریز بافر (اگر کد خالص دات نتی باشد و از API ویندوز به صورت ناشیانه استفاده نکرده باشد) مقاوم است و خیلی مسایل دیگر که به صورت توکار در دات نت لحاظ شده.
- وضع کردن یک سری قوانین سفت و سخت در مورد کتابخانه‌ها و توابع C مورد استفاده در محصولات نوشته شده با سی و CPP
http://www.dotnettips.info/2009/05/bannedh.html
- تهیه برنامه‌هایی که کار آنالیز خودکار فایل‌های باینری و همچنین سورس کدهای سازمان آن‌را انجام می‌دهند. به این صورت قبل از اینکه دیگران از سیستم‌های آن‌ها باگ‌های متداول را درآورند، خودشان نسبت به اینکار اقدام می‌کنند.
http://blogs.msdn.com/bharry/archive/2009/10/02/two-free-security-tools-from-microsoft-sdl-team.aspx
http://www.dotnettips.info/2009/11/blog-post.html

نویسنده: Hamidreza
تاریخ: ۲۳:۳۲:۴۴ ۱۳۸۸/۱۰/۰۲

یه موضوعی که تو این آمار بهش اشاره نشده تعداد و میزان حرفه ای بودن پروژه هایی هست که دارن از این دیتابیس ها استفاده میکنن. ممکنه این آمار اینجوری بدست اومده باشه که مثلا 10 تا سرور اوراکل تپل رو اومده مقایسه کرده باشه با 2تا سرور اس کیو ال معمولی و در این حالت باگ ها رو شمرده باشه. طبیعتا باگ های امنیتی اونی که در پروژه های بیشتری استفاده شده و سطح پروژه هم اینترپرایزتر هست ملموس تر هست. مثله قضیه ویندوز و لینوکس منظورم هست که اگه فراگیری ویندوز رو لینوکس هم داشت مطمئنا مورد حمله های بیشتری به نسبت ویندوز قرار می گرفت.

نویسنده: وحید نصیری
تاریخ: ۰۰:۳۵:۵۷ ۱۳۸۸/۱۰/۰۳

- این آمار فقط بر اساس تعداد وصله‌های ارائه شده است نه بر اساس تعداد پروژه‌ها یا تعداد سرورها و غیره.
- کسانی که در این زمینه فعالیت می‌کنند و از محصولات باگ در می‌آورند عموما در شرکت‌های امنیتی کاری به سرورهای شما یا پروژه‌های شما ندارند. مطالعات و مهندسی معکوس خودشون رو در آزمایشگاه‌های مجازی تهیه شده انجام می‌دن.
- علت استقبال از اوراکل در یک سری از شرکت‌ها داشتن نسخه‌ی لینوکسی آن است.

مایکروسافت مقایسه‌ای رو بین آخرین نگارش‌های این محصولات اینجا انجام داده که فارق از شرکت تهیه کننده آن آمارهای جالبی را ارائه می‌دهد:

http://www.microsoft.com/sqlserver/2008/en/us/compare-oracle.aspx

عنوان: **دریافت خطای database is not accessible و نحوه‌ی رفع مشکل**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۰/۱۳ ۱۳:۲۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

ممکن است هنگام تلاش جهت اتصال به دیتابیس اس کیوال سرور 2005 به بعد از طریق management studio با پیغام خطای زیر مواجه شوید:

```
The database XYZ is not accessible. (ObjectExplorer)
```

و یا اگر بر روی همین دیتابیس کلیک راست کرده و به خواص آن مراجعه کنیم، خطای 952 زیر صادر شود:

```
Database 'XYZ' is in transition. Try the statement later. (Microsoft SQL Server, Error: 952)
```

اصلا نگران نباشید؛ هیچ مشکلی نیست!
ابتدا رویه‌ی ذخیره شده‌ی sp_who2 را اجرا کنید. یک لیست از کانکشن‌های باز به دیتابیس‌های موجود را به شما خواهد داد.
در این لیست به دنبال کانکشن‌های موجود به دیتابیسی که این خطا را می‌دهد بگردید. Pid این کانکشن‌ها را یافته و سپس با دستور kill pid آن‌ها را از بین ببرید. مشکل حل خواهد شد.
عموما نبستن خود management studio سبب این مشکل می‌شود. بنابراین حتی یکبار باز و بسته کردن آن نیز باید این مشکل را برطرف کند (یا تمام management studio های متصل، که البته راه ساده‌تر همان kill کردن pid آن‌ها است).

نظرات خوانندگان

نویسنده: Mohsen
تاریخ: ۱۳۸۸/۱۰/۱۳ ۱۴:۰۵:۳۲

جالب و مفید بود .مرسی

راستی به من هم سر بزن تازه سایتم رو راه انداختم

http://www.msalehi.net

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۰/۱۳ ۱۴:۱۹:۲۴

سلام
موفق باشید. یک مطلب هم در مورد معرفی مطالب وبلاگ‌ها اینجا نوشتم
http://www.dotnettips.info/2009/09/blog-post_04.html
جهت یافتن و کشف وبلاگ‌های جدید.

عنوان: **تهیه گزارش از منسوخ شده‌های مورد استفاده در SQL Server 2008**
نویسنده: وحید نصیری
تاریخ: ۲۱:۴۱:۰۰ ۱۳۸۸/۱۰/۲۷
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

مطلب " منسوخ شده‌ها در نگارش‌های جدید SQL server " را احتمالا به خاطر دارید. جهت تکمیل آن، کوئری زیر را هم می‌توان ذکر کرد:

```
SELECT instance_name,
      cntr_value
FROM   sys.dm_os_performance_counters
WHERE  OBJECT_NAME = 'SQLServer:Deprecated Features'
      AND cntr_value > 0
ORDER BY
      cntr_value DESC
```

توسط این کوئری گزارشی از منسوخ شده‌های مورد استفاده در دیتابیس‌های شما ارائه می‌شود. برای مثال چندبار از text و ntext استفاده کرده‌اید، آیا هنوز compatibility level دیتابیس‌های خود را تغییر نداده‌اید و مثال‌هایی از این دست .

برای مثال جهت یافتن سریع فیلدهای منسوخ شده text و image دیتابیس جاری از کوئری زیر می‌توان کمک گرفت:

```
SELECT O.Name,
      col.name AS ColName,
      systypes.name
FROM   syscolumns col
      INNER JOIN sysobjects O
           ON  col.id = O.id
      INNER JOIN systypes
           ON  col.xtype = systypes.xtype
WHERE  O.Type = 'U'
      AND OBJECTPROPERTY(o.ID, N'IsMSShipped') = 0
      AND systypes.name IN ('text', 'ntext', 'image')
ORDER BY
      O.Name,
      Col.Name
```

عنوان: **سرورهای متصل شده‌ی SQL Server و مبحث تراکنش‌ها**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۱/۰۱ ۱۷:۳۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

یکی از قابلیت‌های جالب SQL Server در یک شبکه محلی امکان link و اتصال آن‌ها به یکدیگر است. به این صورت امکان کوئری گرفتن (و یا اعمال متداول SQL ایی) از دو یا چند سرور مختلف با دستورات T-SQL میسر می‌شود؛ به نحوی که حس یکپارچگی دیتابیس‌های این سرورها را حین کوئری نوشتن خواهیم داشت.
برای مثال فرض کنید دو سرور SQL1 و SQL2 را در شبکه داریم. می‌خواهیم در سرور SQL1 اتصالی را به سرور SQL2 ایجاد کنیم.

```
USE master

EXEC sp_addlinkedserver
 'SQL2',
 N'SQL Server'

sp_addlinkedsrvlogin @useself='false ', @rmtsrvname = 'SQL2',
@rmtuser = 'sa',
@rmtpassword = 'pass#'
```

دستورات T-SQL فوق کار ثبت یک liked server جدید و اعمال مشخصات کاربری که توسط آن قرار است به سرور SQL2 دسترسی داشت، انجام می‌دهند.
اکنون جهت بررسی این اتصال در سرور SQL1 کوئری زیر را اجرا می‌کنیم:

```
select * from sql2.faxManager.dbo.tblErja
```

که نحوه‌ی فراخوانی جدول مورد نظر باید به صورت Server.DatabaseName.dbo.TableName در آن رعایت شود.
تا اینجا همه چیز خوب است. مشکل از زمانی شروع می‌شود که بخواهیم تراکنش‌ها را نیز دخالت دهیم و اصولی کار کنیم. برای مثال:

```
begin distributed tran
select * from sql2.faxManager.dbo.tblErja
commit tran
```

خطایی که در ویندوز سرور 2003 با آخرین به روز رسانی‌ها ظاهر می‌شود به صورت زیر است:

```
The operation could not be performed because OLE DB provider for linked server was unable to begin a
distributed transaction.
OLE DB provider  for linked server  returned message "The partner transaction manager has disabled its
support for remote/network transactions.".
```

به صورت پیش فرض این نوع تراکنش‌های توزیع شده غیرفعال هستند مگر اینکه فعال شوند و روش حل مشکل نیز به صورت زیر می‌باشد:
قبل از هر کاری به کنسول سرویس‌های ویندوز مراجعه کرده و از در حال اجرا بودن سرویس Distribute Transaction Coordinator اطمینان حاصل کنید.

سپس به قسمت زیر مراجعه نمائید:

```
Control Panel > Administrative Tools > Component Services
```

نود مربوط به Component Service را گشوده و سپس بر روی My Computer کلیک راست کرده و گزینه‌ی خواص را انتخاب کنید.
در صفحه‌ی بازه شده به برگه‌ی MSTDC مراجعه کرده و بر روی دکمه‌ی Security Configuration کلیک نمائید.
اکنون تنظیمات آن‌را مطابق شکل زیر تغییر دهید.

این تنظیم باید بر روی هر دو سرور SQL1 و SQL2 انجام شود.

پس از این تغییرات که شامل راه اندازی مجدد سرویس Distribute Transaction Coordinator نیز خواهد شد، مشکل خطای فوق برطرف شده و امکان استفاده از تراکنش‌ها در linked servers نیز میسر می‌شود.

مشکل دیگری که به آن برخوردم خطای زیر است:

```
Unable to start a nested transaction for OLE DB provider for linked server . A nested transaction was
required because the XACT_ABORT option was set to OFF.
OLE DB provider for linked server returned message "Cannot start more transactions on this session.".
```

برای حل این مشکل یک سطر زیر را باید به ابتدای کوئری خود اضافه کرد که جزو الزامات تراکنش‌های توزیع شده است و به این صورت از rollback کامل تمامی دستورات موجود فراخوانی شده T-SQL در صورت بروز کوچکترین خطایی اطمینان حاصل می‌کند:

```
SET XACT_ABORT ON
```

برای مطالعه بیشتر:

MSDTC Troubleshooting

عنوان: **چک لیست نصب SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۲ ۱۷:۰۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

عموما هنگام نصب SQL Server ، پیش و پس از آن، بهتر است موارد زیر جهت بالا بردن کیفیت و کارآیی سرور، رعایت شوند:

1- پیش فرض‌های نصب SQL Server در مورد محل قرارگیری فایل‌های دیتا و لاگ و غیره صحیح نیست. هر کدام باید در یک درایو مجزا مسیر دهی شوند برای مثال:

```
Data drive D:
Transaction Log drive E:
TempDB drive F:
Backup drive G:
```

این مورد TempDB را کسانی که با SharePoint کار کرده باشند به خوبی علتش را درک خواهند کرد. پیش فرض نصب افراد تازه کار، نصب SQL Server و تمام مخلفات آن در همان درایو ویندوز است (یعنی همان چندبار کلیک بر روی دکمه‌ی Next برای نصب). SharePoint هم به نحو مطلوبی تمام کارهایش مبتنی بر transactions است. یعنی استفاده‌ی کامل از TempDB . نتیجه؟ پس از مراجعه به درایو ویندوز مشاهده خواهید کرد که فقط چند مگ فضای خالی باقی مانده! حالا اینجا است که بدو این مقاله و اون مقاله رو بخون که چطور TempDB را باید از درایو C به جای دیگری منتقل کرد. چیزی که همان زمان نصب اولیه SQL Server باید در مورد آن فکر می‌شد و نه الان که سیستم از کار افتاده.
همچنین وجود این مسیرهای مشخص و پیش فرض و آگاهی از سطوح دسترسی مورد نیاز آن‌ها، از سر دردهای بعدی جلوگیری خواهد کرد. برای مثال : انتقال فایل‌های دیتابیس اس کیوال سرور 2008

2- پس از رعایت مورد 1 ، نوبت به تنظیمات آنتی ویروس نصب شده روی سرور است. این پوشه‌های ویژه را که جهت فایل‌های دیتا و لاگ و غیره بر روی درایوهای مختلف معرفی کرده‌اید یا خواهید نمود، باید از تنظیمات آنتی ویروس شما Exclude شوند. همچنین در حالت کلی فایل‌هایی با پسوندهای LDF/MDF/NDF باید جزو فایل‌های صرفنظر شونده از دید آنتی ویروس شما معرفی گردند.
این مورد علاوه بر بالا بردن کارآیی SQL Server ، در حین Boot سیستم نیز تاثیر گذار است. گاها دیده شده است که آنتی ویروس‌ها این فایل‌های حجیم را در حین راه اندازی اولیه سیستم، پیش از SQL Server ، جهت بررسی گشوده و به علت حجم بالای آن‌ها این قفل‌ها تا مدتی رها نخواهند شد. در نتیجه آغاز سرویس SQL Server را با مشکلات جدی مواجه خواهند کرد که عموما عیب یابی آن کار ساده‌ای نیست.

3- پیش فرض میزان حافظه‌ی مصرفی SQL Server صحیح نیست. این مورد باید دقیقا بلافاصله پس از پایان عملیات نصب اولیه اصلاح شود. برای مطالعه بیشتر: تنظیمات پیشنهادی حداکثر حافظه‌ی مصرفی اس کیوال سرور

4- آیا مطمئن هستید که از تمام امکانات نگارش جدید SQL Server ایی که نصب کرده‌اید در حال استفاده می‌باشید؟
برای مطالعه بیشتر: تنظیم درجه سازگاری یک دیتابیس اس کیوال سرور

5- بهتر است فشرده سازی خودکار بک آپ‌ها در SQL Server 2008 فعال شوند.
برای مطالعه بیشتر: +

6- از paging بیش از حد اطلاعات، از حافظه‌ی فیزیکی سرور به virtual memory و انتقال آن به سخت دیسک سیستم جلوگیری کنید. برای این منظور:
در قسمت Run ویندوز تایپ کنید : GPEDIT.MSC و پس از اجرای آن با مراجعه به Group policy editor ظاهر شده به مسیر زیر مراجعه کنید:

```
windows settings -> security settings -> local policies -> user rights assignment -> lock pages in
memory
```

در اینجا به یوزر اکانت سرویس SQL Server دسترسی lock pages in memory را بدهید.
علاوه بر آن در همین قسمت (user rights assignment) گزینه‌ی "Perform Volume Maintenance tasks" را نیز یافته و دسترسی لازم را به یوزر اکانت سرویس SQL Server بدهید.

7- به روز رسانی اطلاعات آماری SQL Server را به حالت غیرهمزمان تنظیم کنید.
اگر مطالب مرتبط با SQL Server این سایت را مرور کرده باشید حتما با یک سری DMV که دقیقا به شما خواهند گفت بر اساس اطلاعات آماری جمع شده برای مثال بهتر است روی چه فیلدهایی Index درست کنید، آشنا شده‌اید. حالت پیش فرض به روز رسانی این اطلاعات آماری، synchronous است یا همزمان. به این معنا که تا اطلاعات آماری یک کوئری ذخیره نشود، حاصل کوئری به کاربر بازگشت داده نخواهد شد که این امر می‌تواند بر روی کارآیی سیستم تاثیر گذار باشد. اما امکان تنظیم آن به حالت غیر همزمان نیز مطابق کوئری‌های زیر وجود دارد (این مورد از SQL Server 2005 به بعد اضافه شده است):

```
ALTER DATABASE dbName SET AUTO_UPDATE_STATISTICS ON
ALTER DATABASE dbName SET AUTO_UPDATE_STATISTICS_ASYNC ON
```

8- نصب آخرین سرویس پک موجود فراموش نشود. برای مثال این سایت آمار تمام به روز رسانی‌ها را نگهداری می‌کند.

9- حتما رویه‌ای را برای تهیه بک آپ‌های خودکار پیش بینی کنید. برای مثال : +

10- میزان فضای خالی باقیمانده درایوهای سرور را مونیتور کنید. اطلاعات بیشتر: +

11- با نصب سرور جدید و تنظیم collation آن به فارسی، به نکات " یافتن تداخلات Collations در SQL Server " دقت داشته باشید.

نظرات خوانندگان

نویسنده: Mohammad Shams Javi
تاریخ: ۱۳۸۹/۰۷/۲۳ ۲۰:۲۱:۲۷

سلام

خیلی مطلب خوب و مفیدی بود آقای نصیری، خصوصا موارد 2 و 6 و 7 جالب و آموزنده بودند.

نویسنده: Amir
تاریخ: ۱۳۸۹/۰۷/۲۸ ۱۴:۵۰:۴۸

مطلب بسیار مفید و کاربردی بود. از اهتمامی که در تهیه این مطالب دارید سپاسگزارم.

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۷/۱۰ ۱۶:۱۸

در اینجا به یوزر اکانت سرویس SQL Server دسترسی lock pages in memory را بدهید. دقیقا منظورتون کدوم یوزر هست.

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۷/۱۰ ۱۶:۳۳

یوزری که سرویس MSSQLSERVER با اون بالا میاد از قسمت sql server configuration manager میتونی چکش کنی...

عنوان: **درک نمودار**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۳/۱۲ ۱۲:۳۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

سؤال: از نمودار زیر چه چیزی را برداشت می‌کنید؟!

منحنی که بالا رفته یعنی چی؟ یعنی بده، خوبه؟!
منحنی‌های پایین‌تر یعنی چی؟ این‌ها بهترند یا بالایی‌های آن‌ها؟
با بالا رفتن حجم فایل‌ها، کدام یک کارآیی بهتری دارد؟ بالایی‌ها یا پایینی‌ها؟

ماخذ این نمودار:(+). البته قبل از مراجعه به ماخذ و مطالعه آن، سعی کنید به سؤالات فوق پاسخ دهید.

نظرات خوانندگان

نویسنده: alireza
تاریخ: ۱۳۹۰/۰۳/۱۴ ۰۷:۱۸:۳۰

حرف حسابش این هست که استفاده از فایل استریم با استفاده از ویندوز آی پی آی خیلی بهتر از استفاده از این شکل دیتا با استفاده از ترنسزاکت هست. یعنی اگر میخواهی با دستورات اس کیو ال با این شکل دیتا کار کنی پرفورمنس خوبی نخواهد داشت. و حتی در این حالت استفاده از واربایندری جواب بهتری میدهد . همچنین همانطور که مشخص است این شکل داده ها از 2 مگابایت بالاتر تازه جواب میدهد

عنوان: بررسی مقدار دهی اولیه متغیرها در T-SQL
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۰۲ ۱۳:۲۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

یکی از موارد مشکل ساز حین استفاده از T-SQL ، مقدار دهی اولیه متغیرها به نال است و اگر اسکریپت تهیه شده کمی طولانی باشد، خطایابی مشکلات مرتبط با آن بسیار مشکل می‌شود. برای مثال:

```
Declare
@x int,
@y int

Set @x = 1
If (@x + @y = 1)
BEGIN
  print 'yes!'
End

Set @y = (select sum(id) from Account)
If @x + @y = 1
BEGIN
  print 'yes!'
End
```

کد فوق بدون هیچگونه خطایی اجرا می‌شود و هیچ وقت هم yes را چاپ نمی‌کند. مشکل هم همینجا است. خطایابی قسمت دوم این اسکریپت کمی مشکل‌تر از حالت قبل است. چون در اینجا به نظر متغیر y صریحا مقدار دهی شده است؛ اما در عمل ممکن است برای مثال به دلیل عدم وجود رکوردی در جدول Account ، باز هم null به آن نسبت داده شود.

بنابراین سؤال این است که چگونه این نوع مشکلات را در یک پروژه با تعداد زیادی رویه ذخیره شده، تابع و غیره می‌توان تشخیص داد؟

پاسخ:

در این مورد قبلا مطلبی در این سایت منتشر شده [+] (البته اگر از نگارش کامل VS 2010 استفاده می‌کنید نیازی به نصب چیزی نخواهید داشت) و نکته‌ی آن بررسی SR0007 است.

نظرات خوانندگان

نویسنده: سپیده
تاریخ: ۱۳۹۱/۰۴/۰۱ ۲۲:۱۴

خیلی جالب بود،مرسی!

عنوان: **اگر نصب سرویس پک اس کیوال سرور Fail شد ...**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۷/۱۹ ۲۲:۵۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

همانطور که مطلع هستید سرویس پک سه SQL Server چند روزی است که منتشر شده . این به روز رسانی بر روی یک سرور بدون مشکل نصب شد؛ در سرور دیگر به علت داشتن یک سری برنامه امنیتی مزاحم (که مثلا دسترسی به رجیستری را مونیتور و سد می‌کنند) با شکست مواجه و در آخر پیغام Fail نمایش داده شد. مجددا آنرا اجرا کردم، سریع تمام مراحل را تمام کرد باز هم Fail را نمایش داد.
خوب؛ گفتم احتمالا مشکلی نیست. سعی کردم به سرور وصل شوم ... پیغام «این سرور دسترسی از راه دور را نمی‌پذیرد» و از این حرف‌های متداول ظاهر شد. به لاگ موجود در Event log ویندوز که مراجعه کردم پیغام خطای زیر نمایان بود:

```
Script level upgrade for database '
```

master

```
' failed because upgrade step 'sqlagent100_msdb_upgrade.sql' encountered error 5597, state 1, severity
16. This is a serious error condition which might interfere with regular operation and the database
will be taken offline. If the error happened during upgrade of the 'master' database, it will prevent
the entire SQL Server instance from starting. Examine the previous errorlog entries for errors, take
the appropriate corrective actions and re-start the database so that the script upgrade steps run to
completion.
```

اوه! اوه! اوه! در این لحظه‌ی عرفانی ، دیتابیس master نابود شده! نمی‌شود وصل شد. سروری که داشت تا مدتی قبل بدون هیچ مشکلی کار می‌کرد، الان دیگر حتی نمی‌شود به آن وصل شد. به کنسول سرویس‌های ویندوز مراجعه کردم (services.msc)، سعی کردم سرویس اس کیوال را که از کار افتاده دستی اجرا کنم، پیغام زیر مجددا در event log ظاهر شد:

FILESTREAM feature could not be initialized. The Windows Administrator must enable FILESTREAM on the instance using Configuration Manager before enabling through sp_configure.

قابلیت FILESTREAM را نمی‌تواند آغاز کند. پس از مدتی جستجو مشخص شد که این مورد را می‌شود در رجیستری ویندوز غیرفعال کرد؛ به صورت زیر:

```
1) Open up Registry Editor
2) Go To HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Microsoft SQL
Server\MSSQL10.MSSQLServer\MSSQLSERVER\FileStream
3) Edit the value "EnableLevel" and set it to 0
4) Restart SQL Server.
```

پس از انجام اینکار، سرویس اس کیوال استارت شد (از طریق کنسول سرویس‌های ویندوز). در ادامه، امکان اتصال به آن نبود (حتی با اکانت sa):

```
Login failed for user 'sa'. Reason: Server is in
```

script upgrade mode

```
. Only administrator can connect at this time. (Microsoft SQL Server, Error: 18401)
```

باز هم پس از مدتی جستجو معلوم گردید که «کمی باید صبر کرد». آن پیغام اول کار مبتنی بر تخریب دیتابیس master هم بی‌مورد است. پس از fail شدن نصب سرویس پک، هنوز برنامه نصاب آن در پشت صحنه مشغول به کار است. این مورد به وضوح در task manager ویندوز مشخص است. سرور به مدت 15 دقیقه به حال خود رها شد. پس از آن بدون مشکل اتصال برقرار گردید و همه چیز مجددا شروع به کار کرد.

بنابراین اگر در حین نصب سرویس پک SQL Server مشکلی پیش آمد، نگران نباشید. باید به نصاب آن زمان داد (برنامه mscorsw.exe در پشت صحنه مشغول به کار است). برنامه نصاب آن هم هیچ نوع خطای مفهومی را گزارش نمی‌دهد. تمام مراحل، بجای نمایش در برنامه تمام صفحه نصاب آن، در event log ویندوز ثبت می‌شود. این برنامه تمام صفحه فقط کارش نمایش یک progress bar است!

اگر ... هیچکدام از این موارد جواب نداد، امکان بازسازی دیتابیس master نیز وجود دارد: [^ , ^]
ولی دست نگه دارید و سریع اقدام نکنید. ابتدا به task manager مراجعه کنید. آیا برنامه mscorsw.exe در حال اجرا است؟ اگر بله، یعنی هنوز کار نصب تمام نشده. حداقل یک ربع باید صبر کنید.

نظرات خوانندگان

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۰/۰۷/۱۹ ۲۳:۰۶:۵۸

ممنون از به اشتراک گذاشتن این نکته

عنوان: **Microsoft® SQL Server® 2012**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۲/۱۹ ۰۹:۳۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

نگارش نهایی Microsoft® SQL Server® 2012 چند روزی هست که ارائه شده . فعلا نسخه آزمایشی RTM آن در اختیار عموم است.
در ادامه جمع آوری لینک‌های مرتبط به این ارائه را مشاهده خواهید نمود:

Microsoft® SQL Server® 2012 Express and LocalDB
Microsoft® SQL Server® 2012 Evaluation
Microsoft® SQL Server® StreamInsight 2.0
Microsoft® SQL Server® 2012 Feature Pack
Microsoft® SQL Server® 2012 Semantic Language Statistics
Microsoft ®SQL Server® 2012 FileStream Driver - Windows Logo Certification
Microsoft® SQL Server® 2012 Performance Dashboard Reports
Microsoft® SQL Server® 2012 Report Builder

و یک جدول مقایسه‌ای بین امکانات نگارش‌های رایگان SQL Server 2012 در اینجا

نظرات خوانندگان

نویسنده: علی قمشلویی
تاریخ: ۱۳۹۰/۱۲/۱۹ ۲۲:۱۰:۱۷

سلام آقای نصیری خیلی داریم استفاده میکنیم لطفا بیشتر بنویسید با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۲/۱۹ ۲۲:۲۴:۲۷

ایمیل‌های خودکار خلاصه اشتراک‌های روزانه من هنوز برقرار است. جهت اطلاع : (^)

عنوان: **نحوه تبدیل نگارش SQL Server 2012 RTM مدت دار، به نگارش کامل**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۲/۲۱ ۲۲:۴۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

Microsoft® SQL Server® 2012 Evaluation از این آدرس قابل دریافت است. همچنین اگر به سایت‌های وارز مراجعه کنید، به ازای هر نگارش SQL Server 2012، یک بسته دریافتی 4 گیگابایتی را به شما ارائه می‌دهند. یعنی اگر کسی بخواهد نسخه developer و نسخه enterprise را دریافت کند بیش از 8 گیگ را باید دریافت نماید!

اما واقعیت این است که نیازی به دریافت هیچکدام نیست. یک فایل ISO مربوط به SQL Server 2012 بیشتر وجود خارجی ندارد. تمام این نگارش‌ها هم فقط براساس Product key است که مشخص می‌شوند. اگر سریال مرتبط با نگارش developer را وارد کنید، این نگارش نصب خواهد شد. اگر سریال نگارش enterprise را وارد کنید، نگارش سازمانی نصب خواهد شد؛ و تمام این‌ها هم فقط با همان یک فایل ISO اصلی ارائه شده توسط مایکروسافت میسر می‌شوند.

این فایل ISO اصلی را از اینجا می‌توان دریافت کرد. بدیهی است Product key توکار و پیش فرض آن که در اختیار عموم است، مدت دار می‌باشد. بنابراین حین نصب تنها نیاز به سریال معتبر وجود دارد.

نظرات خوانندگان

نویسنده: Ahmad Moghadas khoo
تاریخ: ۱۳۹۰/۱۲/۲۲ ۱۷:۲۲:۳۵

کلا لنگه نداری.
وحید جان

نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۰/۱۲/۲۳ ۰۰:۰۱:۳۵

این نگارش نهایی 2012 هست یا باید منتظر نگارش نهایی باشیم؟!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۲/۲۳ ۰۰:۰۷:۴۴

What is RTM

نویسنده: مرتضی
تاریخ: ۱۳۹۰/۱۲/۲۵ ۱۰:۲۲:۰۹

درود بر شما آقای نصیری!

نویسنده: Mohsen
تاریخ: ۱۳۹۰/۱۲/۲۷ ۰۹:۱۸:۰۷

زنده باد مهندس نصیری و سال نو پیشاپیش مبارک!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۲/۲۷ ۲۰:۳۰:۳۱

سلامت باشید؛ سال نوی شما هم مبارک.

عنوان: **ایجاد بک آپ برای دیتابیس با استفاده از CMD**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۴/۲۳ ۱۴:۵۶
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, cmd, SMO

در پست قبلی مهندس نصیری با استفاده کتابخانه ExpressMaint و Schedule ویندوز یه سیستم بک آپ گیری را پیاده کرده اند، در این بین با توجه به وجود SMO میتوان از طریق خط فرمان ویندوز و کمک گیری از کتابخانه 7ZIP و نوشتن فرامین مربوط و ایجاد بچ فایل و افزودن به Schedule ویندوز، نیز میتوان بصورت زیر نیز این پیاده سازی را انجام داد، که بنا بر نیاز اسکریپت بک آپگیری را نوشت:

```
echo off
cls
rem --------------- Variables ---------------
set packName=DbName
set connection=-S Server -U User -P Password
set path=PathForSaveBackup
rem --------------- Variables ---------------

echo create backup
sqlcmd %connection% -Q "BACKUP DATABASE %packName% TO DISK = '%path%\%packName%.bak'"

echo compress backup file
..\tools\7za.exe a %path%\%packName%.7z %path%\%packName%.bak

echo delete backup file
del %path%\%packName%.bak
```

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۴ ۱۳:۷

در SQL Server 2008 اگر سرور رو تنظیم کنید، بک‌آپ‌های تولیدی آن به صورت خودکار فشرده شده تولید می‌شوند: (^) و نیازی به ابزار اضافی برای اینکار ندارند.

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۵/۰۲ ۹:۵۶

برای استفاده این امکان توکار و بی نیاز شدن از 7Zip باید اسکریپت را به این شکل تغییر دهیم:

```
sqlcmd %connection% -Q "BACKUP DATABASE %packName% TO DISK = '%path%\%packName%.bak', COMPRESSION"
```

عنوان: **انتخاب Sub Query درون پرانتزها به کمک [Ctrl+Shift+**
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۴/۲۶ ۱۵:۴۸
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

بوسیله [Ctrl+Shift+ شما می‌توانید از دورن Queryهای تودرتو، Sub Query مورد نظر را انتخاب نمایید، فقط کافیست، کرسر ماوس را ابتدا یا انتهای پرانتز قرار دهید.

امکان فوق از Sql Server 2005 به بعد گذاشته شده است.

منبع:

http://blog.sqlauthority.com

عنوان: **روشی جهت یافتن فیلدهای استفاده شده درون Function ، Stored Procedure و View**
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۲:۲۶
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

گاهی اوقات بدلیل تغییرات در جداول و غیرو...، ممکن است لازم شود، از استفاده قرار گرفتن فیلدهای جداول درون Function ها،Stored Procedureها و یا Viewها ، مطلع شوید. در این زمان شما می‌توانید از روش زیر استفاده نماید.
قبل از هر چیز در ابتدا Script زیر را اجرا نمایید.که شامل یک جدول به نام FindField ، یک Stored Procedure به نام PROCEDURE_FindField و یک Function به نام FUNCTION_FindField می‌باشد.

```
Create Table FindField
(ID int,
Firstname varchar(255),
Lastname varchar(255))
Go
CREATE PROCEDURE PROCEDURE_FindField
AS
    SELECT Firstname FROM FindField;
GO
CREATE FUNCTION FUNCTION_FindField (@id int)
RETURNS int
WITH EXECUTE AS CALLER
AS
begin
declare @id1 int;
   set @id1= (Select Max(Firstname) from dbo.FindField) ;
   RETURN (@id1);
END;
```

در ادامه Script زیر را اجرا نمایید:

```
SELECT OBJECT_NAME(object_id) as 'Procedure/Function/View Name',
definition
FROM sys.sql_modules
WHERE definition LIKE '%' + 'Firstname' + '%'
```

خروجی بصورت زیر خواهد بود:

Results | Messages

	Procedure/Function/View Name	definition
1	PROCEDURE_FindField	CREATE PROCEDURE PROCEDURE_FindField AS ...
2	FUNCTION_FindField	CREATE FUNCTION FUNCTION_FindField (@id int) RE...

همانطور که در شکل مشاهده می‌نمایید، فیلد Firstnameدر Function و Stored Procedure مورد استفاده واقع شده است.

جهت کسب اطلاعات بیشتر به آدرس زیر مراجعه نمایید:

http://msdn.microsoft.com/en-us/library/ms175081.aspx

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۲:۵۶

ضمن تشکر. ابزار (افزونه) رایگانی هم برای انجام اینکار وجود دارد:
SQL Search

نویسنده: فرهاد یزدان پناه
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۳:۱۹

با تشکر از مطلب جالبتون.
به نظر من یک کار دیگه. استفاده از WITH SCHEMABINDING در زمان ایجاد اشیای وابسته است.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۳:۵۲

این روش خوبی است ولی فقط برای viewها و توابع کاربرد دارد و نه برای SPها.

نویسنده: فرهاد یزدان پناه
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۴:۸

بله همینطوره.
به نظر من ایجاد اشیایی همانند Function و Stored Proc با وجود ORMهای امروزی، تا حدودی مناسب نیست (خارج از مواردی که مسائل Performance مطرح است).
ایجاد و بازسازی این اشیا در صورت وجود در هنگام Seed پایگاه داده (چه ایجاد و چه تغییر) می‌تواند بسیاری از مشکلات موجود را حل کند.

نویسنده: مرتضی
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۹:۵۶

سلام
با این کوئری که شما گفتید اگر View, Function , Stored Procedure رمز شده باشند
دیگه نمی‌تونه جواب بده
ولی با این کوئری امکانش هست

```
SELECT TABLE_NAME,
       column_Name
FROM   INFORMATION_SCHEMA.COLUMNS
WHERE  column_Name LIKE '%' + 'Name' + '%'
```

نویسنده: مرتضی
تاریخ: ۱۳۹۱/۰۵/۰۴ ۲۰:۲

اون افزونه رایگانی که مهندس نصیری معرفی کردن
از هر دو استفاده می‌کنه
یعنی هم متن درون view و Function , Procedure رو جستجو می‌کنه
که اگر رمز شده باشند دیگه نتیجه ای نداره
و هم عناوین ستون‌های اون شئ رو جستجو میکنه

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۵/۰۴ ۲۰:۲۴

مرسی از افزونه ای که معرفی نمودید،

عنوان: آشنایی با Row_Number،Rank،Dense_Rank،NTILE
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۵/۰۸ ۲۲:۱۴
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

توابعی که در این بررسی عنوان می‌شود در زمان انتشار نسخه SQL Server2005 ارائه شده است.

قبل از بررسی توابع، Script زیر را اجرا می‌نماییم، که شامل جدولی به نام Testو درج چند رکورد درون آن می‌باشد:

```
CREATE TABLE Test (ID INT, Product VARCHAR(100), Price INT, Color VARCHAR(100))
GO
INSERT INTO Test
SELECT 1, 'Toy', 100, 'Black'
UNION ALL
SELECT 2, 'Pen', 100, 'Black'
UNION ALL
SELECT 3, 'Pencil', 100, 'Blue'
UNION ALL
SELECT 4, 'Pencil', 100, 'Red'
UNION ALL
SELECT 5, 'Pencil', 200, 'Yellow'
UNION ALL
SELECT 6, 'Cup', 300, 'Orange'
UNION ALL
SELECT 7, 'Cup', 400, 'Brown'
GO
```

اولین تابعی را که بررسی می‌نماییم، **Row_Number** می‌باشد و Syntax آن بصورت زیر است:

```
ROW_NUMBER () OVER ([<partition_by_clause>] <order_by_clause>)
```

بوسیله تابع Row_Number می‌توان اعداد توالی (ترتیبی) را به رکوردهای یک جدول نسبت داد. برای روشن‌تر شدن مطلب فوق مثالی را بررسی می‌نماییم.
در ابتدا Query زیر را اجرا نمایید:

```
Select *, ROW_NUMBER() OVER ( ORDER BY Price DESC) AS RN from Test
```

خروجی آن بصورت زیر می‌باشد:

Results | Messages

	ID	Product	Price	Color	RN
1	7	Cup	400	Brown	1
2	6	Cup	300	Orange	2
3	5	Pencil	200	Yellow	3
4	1	Toy	100	Black	4
5	2	Pen	100	Black	5
6	3	Pencil	100	Blue	6
7	4	Pencil	100	Red	7

همانطور که در شکل مشاهده می‌نمایید، یک عدد ترتیبی تولید شده و به هریک از رکوردهای جدول نسبت داده شده است، در مطلب بعدی یک مثال کاربردی از Row_Number خواهم زد.

لازم به یادآوری است که استفاده از Order by در Syntax تابع Row_Number الزامی میباشد.
اگر به Syntax تابع Row_Number توجه نمایید، با کلمه Partition مواجه می‌شویم،که جهت گروه بندی استفاده می‌شود،به عبارت دیگر ممکن است شما بخواهید، ابتدا جدول خود را براساس فیلد یا فیلدهایی Group by نمایید و سپس روی آنها Row_Number را اعمال کنید، که در این حالت از Partition استفاده می‌شود.
برای درک بیشتر Query زیر را اجرا نمایید:

```
Select *,ROW_NUMBER() OVER (PARTITION BY Product ORDER BY Price DESC) AS RN from Test
```

خروجی بصورت زیر خواهد بود:

Results | Messages

	ID	Product	Price	Color	RN
1	7	Cup	400	Brown	1
2	6	Cup	300	Orange	2
3	2	Pen	100	Black	1
4	5	Pencil	200	Yellow	1
5	3	Pencil	100	Blue	2
6	4	Pencil	100	Red	3
7	1	Toy	100	Black	1

همانطور که در شکل مشاهده می‌نمایید، در ابتدا، جدول براساس فیلد Product، دسته بندی (Group by) شده است و سپس اعداد ترتیبی روی هر Group by بصورت جداگانه اعمال شده است.

تابع RANK()

از تابع فوق در جهت رتبه بندی نمودن فیلدهای یک جدول استفاده می‌شود و Syntax آن بصورت زیر میباشد:

```
RANK () OVER ([<partition_by_clause>] <order_by_clause>)
```

برای درک مطلب فوق نیز مثالی می‌زنیم:
ابتدا Query زیر را اجرا می‌نماییم:

```
Select *,RANK() over (ORDER BY Price ) AS RANK from Test
```

خروجی بصورت زیر خواهد بود:

Results | Messages

	ID	Product	Price	Color	RANK
1	1	Toy	100	Black	1
2	2	Pen	100	Black	1
3	3	Pencil	100	Blue	1
4	4	Pencil	100	Red	1
5	5	Pencil	200	Yellow	5
6	6	Cup	300	Orange	6
7	7	Cup	400	Brown	7

یادآوری: زمانی که دورن Order by ترتیب صعودی یا نزولی بودن را تعیین نکنیم، Order by بصورت پیش فرض صعودی میباشد.

همانطور که در شکل مشاهده می‌نمایید،رتبه بندی انجام شده به ترتیب نمی‌باشد، و برای مقادیر تکراری فیلد Price از Rank یکسانی استفاده شده است. نکته دیگر این که بین اعداد مشاهده شده در فیلد Rank نیز gap ایجاد می‌شود. به عبارت دیگر عمده تفاوت تابع Rank با تابع Row_Number همین مواردی است که بیان شده است.

در Syntax تابع Rank نیز کلمه Partition هم وجود دارد، که در جهت Group by فیلد یا فیلدهای خاصی استفاده می‌شود، و رتبه بندی نیز در این حالت روی Group by انجام می‌گردد.

برای درک بهتر Query زیر را اجرا نمایی:

```
Select *,RANK() over (Partition by Product ORDER BY Price Desc) AS RANK from Test
```

خروجی بصورت زیر خواهد بود:

Results | Messages

	ID	Product	Price	Color	RANK
1	7	Cup	400	Brown	1
2	6	Cup	300	Orange	2
3	2	Pen	100	Black	1
4	5	Pencil	200	Yellow	1
5	3	Pencil	100	Blue	2
6	4	Pencil	100	Red	2
7	1	Toy	100	Black	1

همانطور که در شکل مشاهده می‌نمایید، رتبه بندی روی هر Group by بصورت جداگانه اعمال شده است.

تابع Dense_Rank

این تابع نیز همانند تابع Rank عمل می‌کند، با این تفاوت که هیچ gap ی بین اعداد آن رخ نمی‌دهد.

با جرای Query زیر خواهیم داشت:

```
Select *,dense_RANK() over (ORDER BY Price ) AS dense_RANK from Test
```

خروجی بصورت زیر خواهد بود:

Results | Messages

	ID	Product	Price	Color	dense_RANK
1	1	Toy	100	Black	1
2	2	Pen	100	Black	1
3	3	Pencil	100	Blue	1
4	4	Pencil	100	Red	1
5	5	Pencil	200	Yellow	2
6	6	Cup	300	Orange	3
7	7	Cup	400	Brown	4

همانطور که ملاحظه می‌نماییدهیچ gap ی بین اعداد Rank ایجاد نشده است.

و برای استفاده از Partition، درتابع Dense_Rank همانند تابع‌های دیگر میباشد.

تابع NTILE:

این تابع نیز مانند توابع بالا در جهت رتبه بندی استفاده می‌شود، و بوسیله تابع فوق شما می‌توانید رکوردهای جدول خود را به تعداد گروه‌های دلخواه تقسیم نمایید.و Syntaxآن بصورت زیر میباشد:

```
NTILE (integer_expression) OVER ([<partition_by_clause>] <order_by_clause>)
```

برای درک مطلب فوق مثالی می‌زنیم:

```
Select * ,NTILE(4) over ( ORDER BY Price desc) from Test
```

خروجی بصورت زیر خواهد بود:

Results | Messages

	ID	Product	Price	Color	NTILE
1	7	Cup	400	Brown	1
2	6	Cup	300	Orange	1
3	5	Pencil	200	Yellow	2
4	1	Toy	100	Black	2
5	2	Pen	100	Black	3
6	3	Pencil	100	Blue	3
7	4	Pencil	100	Red	4

در Syntax تابع فوق اشاره به Integer_Expressionشده است.که یک مقدار عددی دریافت می‌کند و بیانگر تعداد گروه بندی دلخواه میباشد.

حال سئوال اینجاست که رتبه بندی جدول به چه صورت انجام شده است:

همانطور که مشاهده می‌نمایید، جدول فوق شامل 7 رکورد می‌باشد،و ما در مثال خود،تمایل داشتیم که رکوردهای جدول به چهار گروه تقسیم و سپس رتبه بندی شوند، بنابراین 7 تقسیم بر 4 شده است و باقی مانده آن می‌شود 3

پس خواهیم داشت4*1+3=7

در ابتدا چهار گروه ایجاد می‌شودو در هر خانه یک رکورد قرار می‌گیرد

سپس 3 رکورد باقی می‌ماند که از اولین گروه رو به پایین ، برای هر گروه فقط یک رکورد درج می‌شود، یعنی یک رکورد به گروه

یک،یک رکورد به گروه 2 و هم چنین یک رکورد به گروه 3 بنابراین خواهیم داشت:

نکته مهم: **اگر تعداد رکورد باقی مانده بعد از تقسیم بیش از یک عدد باشد، در زمان اختصاص دادن به گروه ها، به هر گروه از بالا به پایین فقط یک رکورد اختصاص داده می‌شود.**

مثالی دیگر:

```
Select *,NTILE(3) over ( ORDER BY Price desc) AS NTILE from Test
```

خروجی:

Results | Messages

	ID	Product	Price	Color	NTILE
1	7	Cup	400	Brown	1
2	6	Cup	300	Orange	1
3	5	Pencil	200	Yellow	1
4	1	Toy	100	Black	2
5	2	Pen	100	Black	2
6	3	Pencil	100	Blue	3
7	4	Pencil	100	Red	3

در این حالت 3*2+1=7

امیدوارم مطلب فوق مفید واقع شده باشد.

نظرات خوانندگان

نویسنده: حمیدرضا
تاریخ: ۱۳۹۱/۰۵/۰۹ ۷:۵۹

مطلب بسیار مفیدی بود. تشکر می‌کنم.

عنوان: **بازگردانی پایگاه داده بدون فایل لاگ**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۵/۱۰ ۱۰:۲
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

در بعضی مواقع ممکن است که در حین کار و با تراکنش‌های باز، دیتابیس SQL Server دچار مشکل شود و از دسترس خارج شود و این فرض را هم در نظر بگیرید که بک آپ دیتابیس مربوط به زمانی بیش از حد انتظار است. آیا ممکن است که دیتابیس را با وجود از دست دادن فایل لاگ آن بازگردانی کرد؟

جواب: بله ولی بدون عواقب نیست.

بطور معمول در زمانی که تراکنش‌های باز بر روی سرور دیتابیس وجود دارد و بانک کرش میکند، کرش ریکاوری، تراکنش‌های باز را رول بک میکند. این امر مانع از اثرات پراکنده از تراکنش‌های فعلی در پایگاه داده می‌باشد.

اگر لاگ در دسترس نباشد هنگامی که کار سرور SQL شروع می‌شود، پایگاه داده در حالت SUSPECT قرار داده می‌شود. در این مواقع تنها راه آن لاین نمودن بانک (البته منظور آماده بکار نمودن بانک نیست) استفاده از حالت قابلیت تعمیر اورژانسی است که از نسخه 2005 افزده شده است که با ساخت یک فایل لاگ جدید و سپس اجرا نمودن DBCC CHECKDB با استفاده از REPAIR_ALLOW_DATA_LOSS انجام می‌شود.

مشکل اینجاست که اگر شما به همین طریق بخواهید ادامه بدهید و از این قابلیت اورژانسی استفاده کنید، مسلما امکان وجود دسته‌ای از تغییرات در بانک که ممکن است در میانه اعمال بروز رسانی چند رکورد فروش در یک جدول نیمی از تراکنش‌های آن‌ها روی بانک اعمال شده و بعد از راه اندازی دوباره با لاگ جدید، امکان برگرداندن آن‌ها وجود داشته باشد، مواجه شوید. به این معنی که در بهترین حالت، پایگاه داده بدست آمده با برنامه هماهنگ نیست و یا اینکه به بک آپی که وجود دارد رضایت بدهید.

حالت تعمیر اورژانسی، زمانیکه همه راه‌حل‌های بازگردانی بانک مغلوب شوند آخرین روش مانده است. این حالت، حالتی بین دو شر، ریکاور کردن بانک به حالتی نا هماهنگ با نرم افزار و یا برگرداندن به زمانی خیلی عقب‌تر است که در نهایت اقدام به انجام هر دو و هماهنگ‌تر کردن بانک برای کار با برنامه میشوید که بسیار وقت گیر و مشکل ساز است.

ولی باز ممکن است این وضعیت پیش بیاد چون شما از اتفاقی که در زمان کرش در بانک افتاده اطلاعی ندارید. برای جلوگیری از این موارد در آینده سازوکار بک آپ گیری از بانک را باید تغییر بدهید و دفعات بک آپ گیری را افزایش بدهید و نیز از روش‌های جدید که قابلیت دسترسی بالا دارند استفاده کنید همانند mirroring و SQL Server 2012 Availability Groups است.

منبع: ^

لینکهای مرتبط:

1. Is the recovery interval guaranteed? ^
2. Scaling and Sharding Deep Freeze ^
3. Why Database Operations Hit the Wall ^
4. Avoiding logging for user operations ^
5. Resolving the Database Performance Problem Blame Game ^

نظرات خوانندگان

نویسنده: ali
تاریخ: ۱۳۹۱/۰۵/۱۰ ۱۱:۹

لطفا کاملتر توضیح بدید.
من دیتابیسی دارم که فایل لاگ آن پاک شده باید چیکار کنم؟

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۵/۱۰ ۱۲:۲۱

سوالتون کلی هست لطفا توضیح بیشتری بدید و البته باید توجه داشته باشید که با توجه به اطلاعات بالا اگه میخواهید از تعمیر اورژانسی استفاده کنید نسخه بانک باید از 2005 به بالا باشه

نویسنده: ali
تاریخ: ۱۳۹۱/۰۵/۱۲ ۰:۷

سوال این هست که من یک دیتابیس دارم که فایل LDF آن (فایل log) پاک شده حالا من بدون فایل LDF چطور این دیتابیس را attach کنم.فایل MDF را واسه Attach انتخاب میکنم Error میده! چون فایل LDF را پیدا نمیکنه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۲ ۰:۱۲

سه راه حل اینجا ذکر شده . آزمایش کنید کدام یکی برای وضعیت شما مفید است.

نویسنده: ali
تاریخ: ۱۳۹۱/۰۵/۱۲ ۰:۲۰

آقای نصیری ممنون با همون روش اول حل شد.

```
USE [master]
GO
-- Method 1: I use this method
EXEC sp_attach_single_file_db @dbname='TestDb',
@physname=N'C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\DATA\TestDb.mdf'
GO
```

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۱۱/۱۱ ۲۲:۵۳

سلام
دیتابیسی هست که به حالت InRecovery رفته. در SQL Server Management Studio در لیست دیتابیس‌ها وجود داره ولی جلوی اسم اون عبارت InRecovery رو نوشته. در این حالت که هست نمیشه که ساختار این دیتابیس اعم از جدول ها، داده‌های داخل اون جداول، stored procedureها و ... رو ببینیم. از اطلاعات اون هم پشتیبان گیری نشده و به داده‌های داخل اون نیاز هست. شما می‌دونید مشکل چیه و چطور حل میشه ؟
با تشکر

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۱۱ ۲۳:۱

چیز مهمی نیست. احتمالا فضا کم اومده روی هارد یا لاگ پر شده. به علاوه فقط باید بهش زمان بدی تا کارش تموم شه.

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۱۱/۱۲ ۱۴:۲

این اسکریپت رو پیدا کردم، منتهی اجراش که میکنم زمان باقی مونده رو اعلام نمیکنه.

```
DECLARE @DBName VARCHAR(64) = 'databasename'

DECLARE @ErrorLog AS TABLE([LogDate] CHAR(24), [ProcessInfo] VARCHAR(64), [TEXT] VARCHAR(MAX))

INSERT INTO @ErrorLog
EXEC sys.xp_readerrorlog 0, 1, 'Recovery of database', @DBName

SELECT TOP 5
 [LogDate]
,SUBSTRING([TEXT], CHARINDEX(') is ', [TEXT]) + 4,CHARINDEX(' complete (', [TEXT]) - CHARINDEX(') is ',
[TEXT]) - 4) AS PercentComplete
,CAST(SUBSTRING([TEXT], CHARINDEX('approximately', [TEXT]) + 13,CHARINDEX(' seconds remain', [TEXT]) -
CHARINDEX('approximately', [TEXT]) - 13) AS FLOAT)/60.0 AS MinutesRemaining
,CAST(SUBSTRING([TEXT], CHARINDEX('approximately', [TEXT]) + 13,CHARINDEX(' seconds remain', [TEXT]) -
CHARINDEX('approximately', [TEXT]) - 13) AS FLOAT)/60.0/60.0 AS HoursRemaining
,[TEXT]

FROM @ErrorLog ORDER BY [LogDate] DESC
```

در ضمن دیتابیس روی درایو D هست و فضای چند ده گیگ خالی داره. حجم فایل ldf هم تا حدود 400 مگ هست. وقتی هم یک کوئری بخوام روی دیتابیس اجرا کنم خطای زیر رو میده:

```
Database 'Manager' is being recovered. Waiting until recovery is finished.
```

عنوان: **LocalDB چیست؟**
نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۱/۰۶/۱۶ ۱:۲۴
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, LocalDB, SQL Server 2012

LocalDB نسخه‌ای جدید از Sql server express است که به توسعه دهندگان این اجازه را می‌دهد تا با نصب آن، از نصب کامل دیگر نسخه‌های Sql server جلوگیری نمایند. LocalDB برای برنامه‌هایی که به صورت Local و بر روی یک سیستم اجرا می‌شوند مورد استفاده قرار می‌گیرد.

مزایای استفاده از این نسخه

فایل نصب با حجم بسیار کم. (28.2MB برای نسخه 32 بیتی و 33.7MB برای نسخه 64بیتی)

سادگی (بدون نیاز به انجام تنظیمات خاص بر روی سیستم)

اجرا در محیطهایی که کاربر جاری دسترسی مدیریتی ندارد.(برای اجرای آن نیاز به Permissionهای مدیریتی نیست و یک کاربر سطح پایین هم می‌تواند آن را اجرا کند)

سادگی نصب

همانند Sql server Express سازگاری کاملی با T-Sql دارد. همچنین از Stored Procedureها ، داده‌های جغرافیایی و مکانی (geometry and geography ها) ، Triggers و Viewها پشتیبانی می‌کند.
سازگاری با Provider معمولی Sql server

عدم اجرای سرویس خاصی در حافظه برای مدیریت دیتابیس. پروسس‌های LocalDb هر زمان که نیاز باشد اجرا می‌شوند و هر زمان که به آنها نیاز نداشته باشیم به صورت اتوماتیک متوقف می‌شوند.

پشتیبانی از خصوصیت AttachDbFileName در کانکشن استرینگ جهت استفاده از فایل بانک اطلاعات به صورت مستقیم

سرویس پک‌های جدید جهت LocalDB به راحتی برروی نسخه موجود نصب میشوند و نسخه قبلی را به روز رسانی میکنند.

نصب یک LocalDB برای همه کاربران یک کامپیوتر

پشتیبانی کامل از Silent Installation

امکان استفاده از آن توسط Asp.net

پشتیبانی از XQuery (XML و XPath) و BLOB

پشتیبانی از Ado.net sync framework

پشتیبانی از LINQ

پشتیبانی از Distributed transactions

کانکشن‌های نامحدود (البته به صورت Local)

نیازمندی‌های نصب

نیاز به نصب Sql server 2012 native client . این مورد به همراه LocalDB روی سیستم نصب نمیشود

نیاز به دسترسی مدیریتی جهت نصب

140MB فضای خالی دیسک سخت

به روز رسانی دات نت فریم ورک 4 به 4.0.2 و یا نسخه‌های بالاتر

محدودیت ها

عدم پشتیبانی از Window server 2003 ، Windows xp و Windows 2000

عدم امکان نصب نسخه 32 بیتی بر روی ویندوز 64 بیتی (حتما باید نسخه 64 بیتی آن را نصب کنید)

فقط می‌توان به صورت Local از آن استفاده کرد. امکان استفاده تحت شبکه وجود ندارد و فقط به کانکشن‌های Local پاسخ می‌دهد.

فقط توسط Sql server 2012 management studio در دسترس می‌باشد. LocalDB را نمی‌توان از طریق Management studioهای قدیمی مدیریت کرد.

عدم پشتیبانی Visual Studio 2010 از LocalDB

عدم اجرا بر روی موبایل‌های هوشمند

محدودیت سایز بانک اطلاعات : 10GB

عدم پشتیبانی از قابلیت FileStream

محدودیت استفاده از فقط یک CPU

عدم امکان Debuging دستورات Sql در هنگام اتصال به LocalDB

نحوه نصب

ابتدا Sql server LocalDB را دانلود نمایید. سپس برای نصب آن بر روی سیستم فقط کافی است که فایل نصاب برنامه را اجرا نموده و License مربوطه را قبول نمایید.همچنین در صورت نیاز به Silent Installation کافی است که از دستور زیر در خط فرمان استفاده نمایید:

```
msiexec /i SqlLocalDB.msi /qn IACCEPTSQLLOCALDBLICENSETERMS=YES
```

همچنین می‌توانید مراحل نصب را توسط فایل نصاب انجام دهید:

Microsoft SQL Server 2012 Express LocalDB RC0
Welcome to the Installation Wizard for SQL Server 2012 Express LocalDB RC0
Setup helps you install, modify or remove SQL Server 2012 Express LocalDB RC0. To continue, click Next.
WARNING: This program is protected by copyright law and international treaties.
< Back
Next >
Cancel

Microsoft SQL Server 2012 Express LocalDB RC0
License Agreement
Please read the following license agreement carefully.
MICROSOFT PRE-RELEASE SOFTWARE LICENSE TERMS
MICROSOFT SQL SERVER 2012 EXPRESS LOCALDB RELEASE CANDIDATE 0 (RC0)
These license terms are an agreement between Microsoft Corporation (or based on where you live, one of its affiliates) and you. Please read them. They apply to the pre-release software named above, which includes the media on which you received it, if any. The terms also apply to any Microsoft
I accept the terms in the license agreement
I do not accept the terms in the license agreement
< Back
Next >
Cancel

Microsoft SQL Server 2012 Express LocalDB RC0
Ready to Install the Program
Setup is ready to begin installation.
Click Install to begin the installation.
If you want to review or change any of your installation settings, click Back. Click Cancel to exit Setup.
< Back
Install
Cancel

Microsoft SQL Server 2012 Express LocalDB RC0
Installing SQL Server 2012 Express LocalDB RC0
The program features you selected are being installed.
Please wait while the Installation Wizard installs SQL Server 2012 Express LocalDB RC0. This may take several minutes.
Status:
< Back
Next >
Cancel

نحوه اتصال به LocalDB توسط Sql Server Management Studio

اگر net framework. خود را از نسخه 4 به 4.0.2 و یا نسخه‌های بعد از آن به روز رسانی کرده باشید می‌توان توسط Sql Server Management Studio 2012 به Sql server LocalDB وصل شد. عبارت (local)\v11.0 را به عنوان نام سرور وارد نمایید.

مجددا لازم به ذکر است که امکان اتصال توسط Management Studioهای قبلی به بانک LocalDB امکان پذیر نمی‌باشد.

برای مطالعه بیشتر

نظرات خوانندگان

نویسنده: ramin_rp
تاریخ: ۱۳۹۱/۰۶/۱۶ ۹:۳۷

سلام
تا جایی که من اطلاع دارم نسخه rtm از vs 2010 ساپورت نمیشه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۶ ۱۲:۷

هیچ بانک اطلاعاتی خاصی الزاما نیازی به VS.NET ندارد. کانکشن استرینگ را تنظیم می‌کنید و بعد می‌تونید با کد نویسی حتی جدول، فیلد و تمام تنظیمات مورد نظر را اضافه کنید.
ضمن اینکه خیلی از توانایی‌های آن با VS.NET 2010 هم کار می‌کند (^).

نویسنده: علی یزدی
تاریخ: ۱۳۹۱/۰۶/۱۷ ۲:۲۲

واقعاً که عالی بود. ممنون

نویسنده: کیان
تاریخ: ۱۳۹۱/۰۶/۲۸ ۱۵:۵۹

ایا LocalDB جایگزینی برای SqlExpress خواهد بود در ادامه؟ (اینجا انتهای پاراگراف اول منظورش چیه دقیقا؟ *SQL Server Express LocalDB should be used in place of the SQL Server Express user instance feature which is deprecated*)

هیچ شکلی نمی‌توان تحت شبکه کار کرد با این نسخه؟ مثلا جایی که قراره 3 تا سیستم بصورت کلاینت/سرور کار کنن نمی‌شه فایل دیتابیس روی یک سیستم باشه و از رو 2تای دیگه بهش کانکت شد؟ سناریویی که خیلی وقتا ممکنه استفاده بشه
بهترین راه برای سناریوهایی مثل مورد 2 که مثال زدم چیه؟ (یک محیط و یک برنامه معمولی که قرار نیست sql server بصورت full نصب بشه رو سرور) آیا می‌شه با خیال راحت از **SqlServer Express Edition** استفاده کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۸ ۱۶:۱۸

- خیر. نام کامل این محصول « SQL Server Express LocalDB » است + نسخه «Microsoft® SQL Server® 2012 Express» از اینجا قابل دریافت است. بنابراین جایگزین یا حذف نشده.
- هدف اصلی از LocalDb ارائه یک « embedded database » جدید از طرف مایکروسافت است.
بنابراین هدف آن استفاده تحت شبکه نیست. جاهایی استفاده می‌شود که تک کاربر نهایی دانش آنچنانی در نصب و نگهداری بانک‌های اطلاعاتی ندارد و برنامه و سیستم بانک اطلاعاتی او یکپارچه به نظر می‌رسند. از این نمونه بانک‌های اطلاعاتی embedded باز هم هستند. مانند SQL CE، SQLite ، Firebird Embedded و غیره.
- نسخه Express تحت شبکه قابل استفاده است؛ البته نیاز به تنظیم دارد.

نویسنده: حسین
تاریخ: ۱۳۹۱/۰۸/۰۲ ۲۱:۱۰

من با Management Studio ی SQL Server 2008 همین (localdb)\v11.0 رو وارد کردم و متصل شد. مطمئن هستید که فقط با SQL 2012 میشه بهش وصل شد؟

نویسنده: رضا بزرگی
تاریخ: ۱۳۹۱/۰۸/۰۶ ۲۰:۳۵

- میشه لطفا بگین منظور شما از تحت شبکه دقیقا چیه؟
- برنامه MVC4 که با localdb هست را چگونه میشود روی هاست‌های اشتراکی استفاده کرد؟ آیا نیاز به تنظیم خاص در کانکشن‌استرینگ دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۰۶ ۲۰:۴۹

- یعنی حالت سرور داشته باشد و بتوان به آن مستقیما از راه دور متصل شد. مثل SQL Server کامل.
- هاست اشتراکی باید این بانک اطلاعاتی را پیشتر نصب کرده باشد.

مانند SQL Server CE نیست که با توزیع چند DLL مشکل آن برطرف شود و نیازی به نصب نداشته باشد.

نویسنده: میثم
تاریخ: ۱۳۹۲/۰۱/۱۱ ۱۰:۲۴

سلام. دیتابیس برنامه من LocalDB هستش و از روش EF Code First هم برای برقراری ارتباط با دیتابیس استفاده میکنم. میخوام فایل‌های مرتبط با دیتابیسم mdf. و ldf. توی فولدر Bin باشه. ولی وقتی از کانکشن استرینگ زیر استفاده میکنم دیتابیس رو توی C:\Users\... قرار میده. در حالی که با SQL Compact چنین مشکلی نداشتم و فایل sdf. رو در فولدر Bin قرار میداد.

```
<entityFramework>
    <defaultConnectionFactory type="System.Data.Entity.Infrastructure.SqlConnectionFactory,
EntityFramework">
      <parameters>
        <parameter value="Data Source=(localdb)\v11.0; Integrated Security=True;
MultipleActiveResultSets=True;" />
      </parameters>
    </defaultConnectionFactory>
  </entityFramework>
```

```
<connectionStrings>
    <clear/>
    <add name="MyContext"
         connectionString="Data Source=(localdb)\v11.0; Integrated Security=True;
MultipleActiveResultSets=True; AttachDBFilename=MyDatabase.mdf"
         providerName="System.Data.SqlClient"
           />
  </connectionStrings>
```

ممنون میشم راهنماییم کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۱۱ ۱۰:۳۸

این یک روش عمومی است و در تمام رشته‌های اتصالی دات نتی کار می‌کند:
از DataDirectory استفاده کنید. مثلا:

```
AttachDBFilename=|DataDirectory|\database.mdf
```

مقدار آن در برنامه‌های ASP.NET به صورت خودکار به پوشه استاندارد App_Data مپ می‌شود. برای سایر حالات می‌تونید اون رو در زمان آغاز برنامه دستی مقدار دهی کنید:

```
AppDomain.CurrentDomain.SetData("DataDirectory", "C:\myDB");
```

ضمنا روش مسیردهی کامل هم همیشه کار می‌کند

```
AttachDbFilename='Full\Path\To.MDF'
```

نویسنده: یزدان
تاریخ: ۱۳۹۲/۰۱/۱۱ ۱۲:۱۳

سلام؛ AppDomain چیه ؟
اینکه میگن توی یک برنامه ممکنه چند AppDomain داشته باشیم چه معنی میده ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۱۱ ۱۲:۲۷

AppDomain روشی است برای ایزوله سازی اطلاعات برنامه‌ها از یکدیگر. به این ترتیب در یک پروسه هم می‌شود چندین برنامه دیگر را به صورت کاملا ایزوله از هم بارگذاری و اجرا کرد.

نویسنده: رحیم
تاریخ: ۱۳۹۲/۰۱/۱۵ ۰:۳۰

اگر از ویندوز XP رو هم پشتیبانی می‌کرد خیلی خوب می‌شد، بدون این گزینه نمی‌شه رفت سراغش مخصوصا برای برنامه‌های شرکتی که باید روی اکثر سیستم عامل‌ها اجرا بشه (و فعلا xp استفاده زیادی داره)

نویسنده: حسین
تاریخ: ۱۳۹۲/۱۱/۱۸ ۱:۰

با تشکر
من 2 تا مشکلی که در هنگام استفاده از local db در ویژوال استدیو 2012 برخورد کردم این بود که
-1هر از گاهی پیغام کم آوردن خطای حافظه رو میداد (مثلا هنگامی که میخواستم محتویات جداول رو ببینم) و تقریبا توی اکثر اوقات چاره ای به جز بستن ویژوال استدیو و باز کردن مجدد آن نداشتم
-2اسکریپت تهیه کردن از جدول و اطلاعات داخل آن واقعا عذاب آوره و هر جدول رو باید جداگانه اسکریپت تبدیل میکردم
خواستم ببینم راهی برای این مشکل وجود دارد؟

نویسنده: محسن درپرستی
تاریخ: ۱۳۹۳/۰۱/۰۹ ۲۰:۳۷

امروز هرچقدر سعی می‌کنم یک دیتابیس Sql server ce رو روی هاست اشتراکی منتقل کنم مدام با خطای زیر مواجه می‌شم . راه حل‌های مختلفی رو هم امتحان کردم مثل تغییر سطح تراست یا تغییر نسخه کامپایل به x86 یا x64 یا این و این
نهایتا به جایی نرسیدم .

```
Unable to load the native components of SQL Server Compact corresponding to the ADO.NET provider of
version 8854. Install the correct version of SQL Server Compact. Refer to KB article 974247 for more
details.
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۰۹ ۲۰:۵۵

روی این هاست قبلا یک نسخه‌ی SQL Ce نصب شده. الان شماره درایور دات نت SQL Ce که در برنامه‌ی شما در حال استفاده است، تطابقی با نگارش نصب شده‌ی روی سرور ندارد.
بسته کامل آن‌را از نیوگت دریافت کنید. از اینجا
علاوه بر ارائه درایور دات نتی، حاوی دو پوشه amd64 و x86 کپی شده در پوشه bin برنامه هم خواهد بود (فایل‌های native اصل بانک اطلاعاتی). همین پوشه‌ها و فایل‌ها را باید به همراه برنامه نیز ارائه کنید.

نویسنده: علی پناهی
تاریخ: ۱۳۹۳/۰۴/۱۱ ۸:۵۰

در کد زیر اگر کانکشن استرینگ را کامنت کنم بانک ساخته می‌شود و مشکلی ندارد در غیر این صورت پیغام خطای زیر را می‌دهد:

```
A file activation error occurred. The physical file name 'tax.mdf' may be incorrect. Diagnose and
correct additional errors, and retry the operation.

CREATE DATABASE failed. Some file names listed could not be created. Check related errors.
```

فایل app.config :

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>

    <section name="entityFramework"
type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />
  <!-- For more information on Entity Framework configuration, visit
http://go.microsoft.com/fwlink/?LinkID=237468 --></configSections>

  <connectionStrings>
    <clear/>
    <add name="Context"
         connectionString="Data Source=(localdb)\v11.0; Integrated Security=True;
MultipleActiveResultSets=True; AttachDBFilename=tax.mdf"
         providerName="System.Data.SqlClient"
           />
  </connectionStrings>



  <entityFramework>

     <defaultConnectionFactory  type="System.Data.Entity.Infrastructure.LocalDbConnectionFactory,
EntityFramework">
      <parameters >
        <parameter value="v11.0"/>
      </parameters>
    </defaultConnectionFactory>
  </entityFramework>

<startup><supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" /></startup></configuration>
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۹:۵۰

کمی بالاتر در مورد تنظیمات data directory بحث شده.

عنوان: **نحوه ایجاد Sequence و استفاده آن در Sql Server 2012**
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۷/۲۲ ۱۵:۸
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server 2012, SQL Server

قبل از ایجاد Sequence در Sql Server 2012، توضیح مختصری را درباره آن می‌دهم.
در واقع Sequence روشی برای تولید اعداد ترتیبی با قابلیت افزایش یا کاهش عددهای دلخواه می‌باشد که توسط کاربر یا برنامه نویس ایجاد می‌شود. بنابراین Sequenceها User-Defined می‌باشند.
در اینجا ممکن است سئوالی پیش بیاید که اینکار توسط Identity هم قابل انجام است، اما چرا استفاده از Sequence توسط مایکروسافت پیشنهاد می‌شود.
بدلایل زیر استفاده از Sequence بهتر می‌باشد:

ممکن است Application شما قبل از درج رکورد، درون یک جدول نیاز به عدد منحصربفردی داشته باشد.
عدد تولید شده بوسیله Sequence را می‌توانید بین جداول یا ستونهای مختلف یک جدول به اشتراک بگذارید.
می‌توانید روند تولید اعداد ترتیبی را Restart نمایید. به عبارت دیگر قابلیت Restart نمودن Sequence وجود دارد.
می‌توانید Sequence خود را براساس Sort یک یا چند فیلد، تنظیم نمایید.

Syntax آن به شرح ذیل می‌باشد:

```
CREATE SEQUENCE [schema_name . ] sequence_name
    [ AS [ built_in_integer_type | user-defined_integer_type ] ]
    [ START WITH <constant> ]
    [ INCREMENT BY <constant> ]
    [ { MINVALUE [ <constant> ] } | { NO MINVALUE } ]
    [ { MAXVALUE [ <constant> ] } | { NO MAXVALUE } ]
    [ CYCLE | { NO CYCLE } ]
    [ { CACHE [ <constant> ] } | { NO CACHE } ]
    [ ; ]
```

شرح Syntax :
در زمان ایجاد Sequence، نوع آن می‌بایست عددی باشد، چنانچه Type، آن را مشخص ننمایید، SQL Server، نوع آن را bigint در نظر می‌گیرد.
Start With: بدین مفهوم می‌باشد، که Sequence ایجاد شده از چه عددی آغاز شود.
INCREMENT BY: مفهومش این است که Sequence به چه مقداری افزایش یا کاهش یابد. به عبارت دیگری عدد تولید شده براساس مقدار Increment by تولید می‌شود.
Minvalue: کمترین مقداری که Sequence می‌تواند ایجاد نماید.
Maxvalue: بیشترین مقداری که Sequence می‌تواند ایجاد نماید.
Cycle: مقداری را که برای Cycle تعیین می‌نماییم، بدین مفهوم است که Sequence پس از چه عددی می‌بایست Restart شود.
Cache: عددی که برای Cache در نظر می‌گیریم، مفهومش این است که چه تعداد از اعداد تولید شده توسط Sequence، قبل از استفاده، می‌تواند در Cache قرار گیرد.

در ادامه با یک مثال ساده، یک Sequence ایجاد می‌نماییم:

```
CREATE SEQUENCE [dbo].[SequenceTest]
 AS [int]
 START WITH 1
 INCREMENT BY 1
 MINVALUE 1
 MAXVALUE 30
 CYCLE
 CACHE
GO
```

در مثال بالا Start with برابر یک است، یعنی اولین عددی که تولید می‌شود، برابر یک است،INCREMENT BY برابر یک است یعنی

در هر بار فراخوانی Sequence یک عدد به عدد تولید شده قبلی افزوده می‌شود. مقدار Minvalue برابر یک است، یعنی کمترین مقداری که Sequence می‌تواند تولید نماید برابر یک است. مقدار Maxvalue برابر 30 است، یعنی بیشترین مقداری که Sequence می‌تواند تولید نماید برابر 30 می‌باشد. و Cycle هم برای Sequence فوق در نظر گرفته شده است، بدین مفهوم ، که Sequence به مقدار 30 برسد، Restart شده و از مقدار یک شروع به تولید اعداد می‌نماید.

برای اینکه بتوانیم مقدار Sequence را بدست آوریم، کافیست از Syntax زیر استفاده نمایید:

```
NEXT VALUE FOR [ database_name . ] [ schema_name . ]  sequence_name
   [ OVER (<over_order_by_clause>) ]
```

به عنوان مثال داریم:

اگر Select بالا را تا 30 بار انجام دهید، برای دفعه 31 مقدار آن یک می‌شود، چون در زمان تعریف Sequence، Cycle را انتخاب کرده بودیم. در غیر اینصورت برای دفعه 31 با خطا زیر مواجه می‌شوید.

```
Msg 11728, Level 16, State 1, Line 1
The sequence object 'SequenceTest' has reached its minimum or maximum value. Restart the sequence
object to allow new values to be generated.
```

یکی از امکانات جالب Sequence این است که شما می‌توانید Sequence را روی یک فیلد Sort شده تنظیم نمایید، برای روش شدن مطلب به مثال زیر توجه نمایید:

در ابتدا مطابق Script زیر جدولی را ایجاد و مقادیری را درون آن درج می‌نماییم:

```
create table Kids
( ID int,
Name varchar(50)
);
Go
insert Kids
 values
  (1,'Emma')
, (1,'Tabitha')
, (2,'Kendall')
, (3,'Delaney')
, (4,'Kyle')
, (5,'Jessica')
, (6,'Josh')
, (7,'Kirsten')
, (8,'Amanda')
, (9,'Jimmy')
;
```

سپس یک Schema به نام Sample ایجاد می‌کنیم:

```
CREATE SCHEMA Samples ;
GO
```

در ادامه یک Sequence به نام Test ایجاد می‌کنیم:

```
CREATE SEQUENCE Samples.Test
    AS tinyint
    START WITH 1
    INCREMENT BY 1 ;
GO
```

حال Query زیر را اجرا می‌نماییم:

```
SELECT NEXT VALUE FOR Samples.Test OVER (ORDER BY Name) AS NutID, ID, Name FROM  test1.Kids
WHERE Name LIKE '%e%' ;
```

در Query بالا روی فیلد Name به صورت صعودی عملیات Sort انجام می‌شود، و سپس Sequence روی آن اعمال می‌گردد، برای روشن‌تر شدن مطلب خروجی را مشاهده نمایید که تعداد رکورد آن 6 میباشد.

امیدوارم مطلب فوق مفید واقع شده باشد.

نظرات خوانندگان

نویسنده: کاربر
تاریخ: ۱۳۹۲/۰۴/۲۹ ۱۷:۵۳

سلام
مشکلی بر IDENTITY وارد هست بوجود آمدن Gap بین داده هاست مثلا با حذف پی در پی سطرهای جدول بین مقادیر تولید شده Gap بوجود می‌آید ولی ما این را نمیخواهیم، نیاز داریم که همیشه مقادیر identity پشت سر هم و متوالی باشند یعنی با حذف یک سطر میانی تمام مقادیر شیفت پیدا کنند. و برای رفع این مساله هیچ راهی وجود ندارد. حتی اگر بخواهیم با تابع row_number مقادیر IDENTITY را بعد متوالی کردن بروز رسانی کنیم باز دستور set insert_identity table_name on این اجازه را نمی‌دهد یعنی این دستور

```
set insert_identity tbl on
update t
set id = new_id
from (select new_id = row_number() over(order by id), *) t
set insert_identity tbl off
```

در کتابی خواندم که این مشکل با sequence حل میشود.
میخوام بدانم چگونه؟

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۲/۰۵/۰۱ ۱۶:۳۰

سلام
شما قادر نیستید یک فیلد Identity را بروز رسانی نمایید، دستور set insert_identity Tablename on به شما اجازه Insert به جدول بدون Identity را می‌دهد، برای اینکه بتوانید Gap مرتبط به فیلد Identity را در جدول برطرف کنید، در ابتدا از جدول مورد نظر خود یک کپی تهیه و جدول اصلی را Truncate کنید، سپس یک Sequencer ایجاد و محتویات جدول کپی را بوسیله Sequencer در جدول اصلی کپی نمایید.
فرض کنیم جدول اصلی Table_3 باشد، ابتدا آن را کپی می‌کنیم در جدولی به نام T

```
Select * into T from table_3
```

سپس دستور Truncate را روی جدول Table_3 اجرا کنید:

```
truncate table dbo.table_3
```

حال یک Sequence ایجاد کنید:

```
CREATE SEQUENCE testEventCounter
    AS int
    START WITH 1
    INCREMENT BY 1 ;
```

در ادامه محتویات جدول کپی را به جدول اصلی منتقل نمایید:

```
SET IDENTITY_INSERT table_3 on
INSERT INTO table_3 (ID, Descritp)
SELECT
      NEXT VALUE FOR testEventCounter AS id
    , Descritp
```

```
FROM T
```

راه دیگر این است که به جای استفاده از Identity از Sequence در فیلد خود استفاده نمایید، بصورت زیر :

```
CREATE TABLE Table3
(
    ID int PRIMARY KEY CLUSTERED
        DEFAULT (NEXT VALUE FOR SequenceTest),
    De nvarchar(300) NULL
) ;
GO
```

در هنگام ایجاد جدول Sequence را به فیلد ID ست کردیم.

حال هر زمانی که بخواهید می‌توانید فیلد ID را مطابق Sequence خود بروز رسانی کنید:

```
Update table3 set id=(NEXT VALUE FOR testEventCounter )
```

موفق باشید و امیدوارم مفید واقع شده باشد

نویسنده: کاربر
تاریخ: ۱۳۹۲/۰۵/۰۱ ۲۰:۸

سلام

خیلی ممنون از پاسخی که دادین. مفید واقع شد.

البته فکر کنم برای گارانتی کردن ترتیب sequence بر اساس ستون id می‌تونیم از OVER(ORDER BY نیز استفاده کنیم.

به این شکل:

```
SET IDENTITY_INSERT table_3 on
INSERT INTO table_3 (ID, Descritp)
SELECT
      NEXT VALUE FOR testEventCounter (OVER ORDER BY ID) AS id
    , Descritp
FROM T
```

به هر شکل با مفهوم جدیدی نیز آشنا شدیم :)

عنوان: **آشنایی با تابع PATINDEX در SQL Server**
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۷/۲۳ ۲۱:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

قبل از مطالعه باید بگویم سطح مقاله مبتدی می‌باشد.
گاهی اوقات در زمان Migration یک دیتابیس شما با جداولی برخورد می‌نمایید که محتویات بعضی از فیلد هایشان ترکیبی از عدد و حروف می‌باشد، و شما برای انجام یکسری از عملیات نیاز دارید. که حروف را از اعداد متمایز نمایید، یا اینکه مکان اولین کاراکتر غیر عددی را بعد از هر عدد، بیابید. برای انجام چنین کاری می‌توان از تابعی به نام Patindex استفاده نمود.
تابع PATINDEX به شما امکان، مکان یابی یک یا چند حرف در بین رشته‌های متنی را می‌دهد.
Syntax تابع PATINDEX بصورت زیر میباشد:

```
PATINDEX ( '%pattern%' , expression )
```

تابع PATINDEX شامل دو آرگومان می‌باشد که هر کدام را به اختصار توضیح می‌دهیم:
1- آرگومان اولPattern نامگذاری شده است، Pattern در واقع یک الگوی اختصاصی میباشد که توسط کاربر، جهت جستجو در یک متن تعیین می‌شود. به بیان ساده‌تر اگر شما دنبال مکان حرف یا کلمه خاصی در یک رشته متنی می‌گردید، می‌بایست آن را در آرگومان Pattern قرار دهید.
لازم است در آرگومان اول حداقل یک % وجود داشته باشد.

حداکثر تعداد کاراکترهایی را که می‌توان در آرگومان اول قرار داد 8000 میباشد.

2- آرگومان دوم یا همان Expression : متنی که عملیات جستجو روی آن اعمال می‌گردد، در این آرگومان قرار می‌گیرد.
اگر تعداد کارکترهای آرگومان دوم Varchar(Max) یا nVarchar(Max) باشد، در آن صورت Type خروجی تابع PATINDEX از نوع bigint می‌باشد، در غیر این صورت Type خروجی تابع PATINDEX از نوع Int است.
اگر مقدار آرگومان دوم Null باشد، تابع PATINDEX مقدار Null بر می‌گرداند.

یک مثال کاربردی از تابع PATINDEX
یافتن اولین کاراکتر غیر عددی در رکودهای یک جدول
ابتدا در دیتابیس tempdb یک جدول به نام UsingPATINDEX مطابق Script زیر ایجاد،و چندین رکورد درون آن درج می‌نماییم:

```
USE tempdb
GO
CREATE TABLE UsingPATINDEX (ID INT, Words VARCHAR (100))
GO
INSERT INTO UsingPATINDEX (ID, Words)
SELECT 1, '1one'
UNION ALL
SELECT 2, '11eleven'
UNION ALL
SELECT 3, '2two'
UNION ALL
SELECT 4, '22twentytwo'
UNION ALL
SELECT 5, '111oneeleven'
GO
```

در ادامه QUERY زیر را اجرا نمایید:

```
SELECT PATINDEX('%[a-z]%',Words) 'مکان اولین کاراکتر غیر عددی',
SUBSTRING(Words,PATINDEX('%[a-z]%',Words),1) 'نام اولین کاراکتر غیر عددی بعد از عدد',
Words 'متن اصلی'
FROM  UsingPATINDEX
```

خروجی آن به شکل زیر است:

Results | Messages

	مکان اولین کاراکتر غیر عددی	نام اولین کاراکتر غیر عددی بعد از عدد	متن اصلی
1	2	o	1one
2	3	e	11eleven
3	2	t	2two
4	3	t	22twentytwo
5	4	o	111oneeleven

توضیح درباره QUERY :

قطعه کد زیر دنبال تمامی حروف a تا z ، درون فیلد Words می‌گردد و به اولین کاراکتر غیر عددی که می‌رسد، مکان آن را بر می‌گرداند.

```
PATINDEX('%[a-z]%',Words) 'مکان اولین کاراکتر غیر عددی'
```

قطعه کد زیر با توجه به مکان کاراکتر، خود کاراکتر را بر می‌گرداند:

```
SUBSTRING(Words,PATINDEX('%[a-z]%',Words),1) 'نام اولین کاراکتر غیر عددی بعد از عدد'
```

مثالی دیگر: فرض کنید،دنبال کلمه ای همانند ensure می گردید، بطوریکه دو حرف اول و دو حرف آخر آن را بخاطر می‌آورید و حرف میانی آن را بخاطر نمی‌آورید، در آن صورت نیز می‌توانید از تابع PATINDEX استفاده نمایید، بدین شکل که به جای حرفی که بخاطر نمی‌آورید از _ استفاده کنید، همانند QUERY زیر:

```
SELECT PATINDEX('%en_ure%', 'please ensure the door is locked');
```

خروجی عدد 8 میباشد، که مکان حرف e است.

در تابع PATINDEX می‌توانید براساس Collation دلخواه عملیات جستجو را انجام دهید، برای روش‌تر شدن مطلب شکل زیر را مشاهده نمایید:

```
Select Patindex( '%ن%','استاد' Collate Arabic_CI_AS)
```

100 %

Results | Messages

	(No column name)
1	3

همانطور که درابتدای مطلبم گفتم در آرگومان اول می‌توان از یک % استفاده نمود، به مثال زیر توجه نمایید:

```
Select PATINDEX('a%', 'abc')
```

خروجی آن مقدار یک است.

مثالی دیگر:

```
Select PATINDEX('%a', 'cba')
```

خروجی آن مقدار 3 می‌باشد.

باید متذکر شوم، با دیدن دو مثال آخر،این تصور ایجاد نشود که تابع PATINDEX شبیه به تابع LIKE می‌باشد، چرا که تابع PATINDEX موقعیت کاراکتر را بر می‌گرداند، نه خود کاراکتر را،عملکرد تابع PATINDEX شبیه به عملکرد تابع CHARINDEX می‌باشد.

عنوان: **اجرای Stored Procedure با چند نوع مقدار برگشتی توسط EF CodeFirst**
نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۱/۰۷/۲۵ ۱۶:۲۹
آدرس: www.dotnettips.info
گروه‌ها: C#, Entity framework, EF Code First, SQL Server

فرض کنید Stored Procedure ی با چند مقدار برگشتی را می‌خواهیم در EF CodeFirst مورد استفاده قرار دهیم. برای مثال Stored Procedure زیر را در نظر بگیرید:

```
CREATE PROCEDURE [dbo].[GetAllBlogsAndPosts]
AS
    SELECT * FROM dbo.Blogs
    SELECT * FROM dbo.Posts
```

Stord Procedure ی که توسط این دستور ساخته می‌شود تمام رکوردهای جدول Blogs و تمامی رکوردهای جدول Posts را واکشی کرده و به عنوان خروجی برمیگرداند (دو خروجی متفاوت). روش فراخوانی و استفاده از داده‌های این StoredProcedure در EF CodeFirst به صورت زیر است :
تعریف دو کلاس مدل Blog و Post به ترتیب برای نگهداری اطلاعات وبلاگ‌ها و پست‌ها در زیر آمده است. در ادامه نیز تعریف کلاس BloggingContext را مشاهده می‌کنید.

```
public class Blog
    {
        public int BlogId { get; set; }
        public string Name { get; set; }

        public virtual List<Post> Posts { get; set; }
    }

    public class Post
    {
        public int PostId { get; set; }
        public string Title { get; set; }
        public string Content { get; set; }

        public int BlogId { get; set; }
        public virtual Blog Blog { get; set; }
    }

    public class BloggingContext : DbContext
    {
        public DbSet<Blog> Blogs { get; set; }
        public DbSet<Post> Posts { get; set; }
    }



using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Data.Objects;

namespace Sproc.Demo
{
    class Program
    {
        static void Main(string[] args)
        {
            using (var db = new BloggingContext())
            {
                 db.Database.Initialize(force: false);

                var cmd = db.Database.Connection.CreateCommand();
                cmd.CommandText = "[dbo].[GetAllBlogsAndPosts]";

                try
                {
                    // اجرای پروسیجر
                    db.Database.Connection.Open();
                    var reader = cmd.ExecuteReader();

                    // خواند رکوردهای blogs
```

```
                    var blogs = ((IObjectContextAdapter)db)
                        .ObjectContext
                        .Translate<Blog>(reader, "Blogs", MergeOption.AppendOnly);

                    foreach (var item in blogs)
                    {
                        Console.WriteLine(item.Name);
                    }

                    // پرش به نتایج بعدی (همان Posts)
                    reader.NextResult();
                    var posts = ((IObjectContextAdapter)db)
                        .ObjectContext
                        .Translate<Post>(reader, "Posts", MergeOption.AppendOnly);

                    foreach (var item in posts)
                    {
                        Console.WriteLine(item.Title);
                    }
                }
                finally
                {
                    db.Database.Connection.Close();
                }
            }
        }
    }
```

در کدهای بالا ابتدا یک Connection به بانک اطلاعاتی باز می‌شود:

```
db.Database.Connection.Open();
```

و پس از آن نوبت به اجرای Stored Procedure می‌رسد:

```
var reader = cmd.ExecuteReader();
```

در کد بالا پس از اجرای Stored Procudure نتایج بدست آمده در یک reader ذخیره می‌شود. شئ reader از نوع DBDataReader می‌باشد. پس از اجرای Stored Procedure و دریافت نتایج و ذخیره سازی در شئی reader ، نوبت به جداسازی رکوردها می‌رسد. همانطور که در تعریف Stored procedure مشخص است این Stored Procedure دارای دو نوع خروجی از نوع‌های Blog و Post می‌باشد و این دو نوع باید از هم جدا شوند.برای انجام این کار از متد Translate شئی Context استفاده می‌شود. این متد قابلیت کپی کردن نتایج موجود از یک شئی DBDataReader به یک شئی از نوع مدل را دارد. برای مثال :

```
var blogs = ((IObjectContextAdapter)db)
           .ObjectContext
           .Translate<Blog>(reader, "Blogs", MergeOption.AppendOnly);
```

در کدهای بالا تمامی رکوردهایی از نوع Blog از شئی reader خوانده شده و پس از تبدیل به نوع Blog درون شئی Blogs ذخیره می‌شود.
پس از آن توسط حلقه foreach محتویات Blogs پیمایش شده و مقدار موجود در فیلد Name نمایش داده می‌شود.

```
foreach (var item in blogs)
  {
            Console.WriteLine(item.Name);
  }
```

با توجه به اینکه حاصل اجرای این Stored Procedure دو خروجی متفاوت بوده است ، پس از پیمایش رکوردهای Blogs باید به سراغ نتایج بعدی که همان رکوردهای Post می‌باشد برویم. برای اینکار از متد NextResult شئی reader استفاده می‌شود:

```
reader.NextResult();
```

در ادامه برای خواندن رکوردهایی از نوع Post نیز به همان روشی که برای Blog انجام شد عمل می‌شود. مطالعه بیشتر

نظرات خوانندگان

نویسنده: ashi mashi
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۴:۳۶

لطفا نحوه اجرا کردن پروسیجرها با مقادیر مختلف در ورودی هم کمی توضیح می‌دادید ممنون می‌شدم.

با تشکر از توضیحات خوبتون،

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۴:۵۴

استفاده مستقیم از عبارات SQL در EF Code first و مثال void runSp آن.

نویسنده: بهمن آبادی
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۶:۱۷

منظور من روش مشابه parameterAttributeها در mapping کردن ورودی‌های پروسیجرها در linq می‌باشد.
نه اینکه صرفا از دستورات sql بصورت command استفاده شود.

مانند استفاده از پروسیجرها با چند ورودی و multiResult بودن آن در linq

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۶:۲۷

در مورد نحوه اجرای رویه‌های ذخیره شده با ورودی‌های مختلف پرسیدید، روش اجراش تا EF 5.0 به همین صورتی است که ملاحظه می‌کنید .

قرار است در EF 6.0 این مساله با Fluent API به نحو دیگری پوشش داده شود.

نویسنده: بهمن آبادی
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۶:۴۱

دقیقا منظورم همون لینکه EF 6.0 که گفتین هستش.
Road Map برای انتشار نسخه جدید EF 6.0 وجود داره که قراره نهایتا تا کی انتشار پیدا بکنه ؟

بازم ممنون.

نویسنده: سید مهدی فاطمی
تاریخ: ۱۳۹۲/۰۷/۰۴ ۲۱:۴۷

تشکر دوست عزیز

اما در مدل من که از دیتابیس گرفتم شی رویه ذخیره شده هم به عنوان یک تابع شبیه سازی شده و مثل شما عمل نکردم اما نمی‌تونم مقادیر رو از این شی بگیرم.

```
var sp1=_db.splogin(user,pas,value1,value2);
```

در توضیح این کد هم بگم که این اس پی 4 پارامتر داره و قراره دو مقدار رو برای من برگردونه .

نویسنده: سانای رحیمی
تاریخ: ۱۳۹۳/۰۴/۲۲ ۱۹:۳۹

سلام
اگر SP ما بدون پارامتر out باشد و ما در آخر دستورات آن از دستور SCOPE_IDENTITY() استفاده کرده باشیم چگونه می‌توانیم مقدار آن را به دست بیاوریم؟

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۴/۲۳ ۱۱:۰

در مطلب استفاده مستقیم از عبارات SQL در EF Code first بحث شده. از متد جنریک db.Database.SqlQuery باید استفاده کنید.

عنوان: آشنایی با Column Store Index در SQL Server 2012
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۷/۲۹ ۱۶:۹
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SQL Server 2012

Column Store Index یکی از ویژگیهای جدید SQL Server 2012 می باشد، که کارایی Query های قایل اجرا روی دیتابیس‌های با حجم داده ای بسیار بالا را (که اصطلاحا به آنها Data Warehouse یا انبار داده گویند)، چندین برابر بهبود بخشیده است .

قبل از توضیح در مورد Column Store مختصری در مورد نحوه ذخیره سازی داده‌ها در SQL Server می پردازیم. می‌توان گفت در SQL Server دو روش ذخیره سازی وجود دارد،یکی بصورت ردیفی که اصطلاحا به آن Row Store یا Row-Wise گویند، و دیگری بصورت ستونی که اصطلاحا به آن Column Store گویند .

در روش ذخیره سازی Row Store، مقادیر ستونها در یک سطر بصورت متوالی ذخیره می‌شوند، در این روش ذخیره سازی از ساختار B-Tree یا Heap استفاده می‌شود.

یادآوری: در ساختار B-Tree، یک گره Root وجود دارد، و گره بعد از Root گره ای است که آدرس گره راست بعدی و آدرس گره چپ بعدی را در خود نگه می‌دارد.

شکل زیر نمای یک درخت B-Tree می‌باشد:

جهت کسب اطلاعات بیشتر درمورد ساختار B-Tree

یادآوری: وقتی در یک جدول، ایندکسی از نوع Clustered ایجاد نماییم، SQL Server، در ابتدا یک کپی از جدول ایجاد و داده‌های جدول را از نو مرتب می‌نماید، و ساختار صفحه ریشه و دیگر صفحات را ایجاد می‌کند و سپس جدول اصلی را حذف می‌نماید. به جدولی که Clustered Index ندارد، اصطلاحا Heap گویند.

برخلاف ذخیره سازی Row Store، در ذخیره سازی Column Store، داده‌ها بصورت ستونی ذخیره می‌شوند،در این روش داده ها، فشرده سازی می‌شوند و اینکار باعث می‌شود،در زمان درخواست یک Query، نیاز به Disk I/o به حداقل برسد، در نتیجه، زمان و سرعت پاسخگویی به پرس و جوها بسیار افزایش می‌یابد.

شکل زیر نحوه ذخیره سازی داده ها،بصورت Row Store را نمایش می‌دهد:

شکل بالا ذخیره سازی داده ها، در ساختار B-Tree یا Heap را نمایش می‌دهد، در شکل فوق یک جدول چهار ستونی با N سطر (Row) در نظر گرفته شده است.بطوریکه ستونهای هر Row بطور متوالی در یک صفحه (Page) یکسان ذخیره می‌شوند.

شکل زیر نحوه ذخیره سازی داده ها،بصورت Column Store را نمایش می‌دهد:

مطابق شکل،ستونهای مربوط به هر Row،همگی در یک صفحه (Page) یکسان ذخیره شده اند. به عنوان مثال ستون C1 که مربوط به سطر اول (Row1) می‌باشد، با ستون C1 که مربوط به سطر دوم (Row2) می‌باشد، در یک ستون و در یک صفحه (Page1) ذخیره شده اند، و الی آخر ...

سئوال: یکبار دیگر به هردو شکل با دقت نگاهی بیاندازید، عمده تفاوت آنها در چیست؟

جواب: درست حدس زدید، تفاوت بارز بین دو روش Column Store و Row Store در نحوه ذخیره سازی داده‌ها می‌باشد. بطور مثال، فرض کنید،در روش ذخیره سازی Row Store، به دنبال مقادیری از ستون C2 می‌باشید، SQL Server می‌بایست کل رکوردهای جدول (منظور همه Rowها در همه Page ها)را Scan نماید، تا مقادیر مربوط به ستون C2 را بدست آورد.درحالیکه در روش ذخیره سازی Column Store، جهت یافتن مقادیر ستون C2، نیازی به Scan نمودن کل جدول نیست،بلکه SQL Server فقط به Scan نمودن ستون دوم (C2) یا Page2 بسنده می‌نماید.همین امر باعث افزایش چندین برابری، زمان پاسخگویی به هر Query می‌شود.

سئوال: در روش ذخیره سازی Column Store، چگونه مصرف حافظه بهینه می‌شود؟

جواب: واضح است، که در روش Row Store، SQL Server مجبور است، برای بدست آوردن داده‌های مورد نظرتان،کل اطلاعات جدول را وارد حافظه نماید(اطلاعات اضافه ای که به هیچ وجه بدرد، نتیجه پرس و جوی شما نمی‌خورد)، و شروع به Scan داده‌های مد نظر شما می‌نماید.بطوریکه در روش Column Store، SQL Server، فقط ستون داده‌های مورد پرس و جو را در حافظه قرار می‌دهد.(در واقع فقط داده هایی را در حافظه قرار می‌دهد، که شما به آن نیاز دارید)،بنابراین،طبیعی است که در روش Column Store مقدار حافظه کمتری نسبت به روش Row Store در هنگام اجرای Query استفاده می‌شود. به عبارت دیگر می‌توان گفت که در روش Column Store به دلیل، به حداقل رساندن استفاده از Disk I/o سرعت و زمان پاسخگویی به پرس و جوها چندین برابر می‌شود.

برای درک بیشتر Row Store و Column Store مثالی می‌زنیم:

فرض کنید،قصد بدست آوردن ستونهای C1 و C2 از جدول A را داریم، بنابراین خواهیم داشت:

```
Select C1, C2 from A
```

روش Row Store:

در این روش همه صفحات دیسک (مربوط به جدول A) درون حافظه قرار داده می‌شود، یعنی علاوه بر ستونهای C1 و C2، اطلاعات مربوط به ستونهای C3 و C4 نیز درون حافظه قرار می‌گیرد،بطوریکه مقادیر ستونهای C3 و C4 به هیچ وجه مورد قبول ما نیست، و در خروجی پرس و جوی ما تاثیری ندارد، و فقط بی جهت حافظه اشغال می‌نماید.

روش Column Store:

در این روش فقط صفحات مروبط به ستون C1 و C2 در حافظه قرار می‌گیرد.(منظور Page1 و Page2 می‌باشد) بنابراین فقط اطلاعات مورد نیاز در خروجی، در حافظه قرار می‌گیرد.
از دیگر مزایای استفاده از روش Column Store، فشرده سازی داده می‌باشد،برای درک بیشتر توضیح می‌دهم:

همانطور که در اوایل مطلبم به عرض رساندم، در روش Row Store ، داده‌ها در یک سطر و در یک Page ذخیره می‌شوند، بنابراین امکان وجود داده‌های تکراری در یک سطر به حداقل می‌رسد، چرا که، اگر فرض کنیم چهار ستون به نام‌های ID،FirstName،LastName و City، داشته باشیم،در آن صورت بطور حتم،در یک سطر، داده تکراری وجود نخواهد داشت، اما ممکن است در تعداد سطرهای زیاد داده‌های تکراری مانند Firstname یا City و غیرو بوجود بیاید، این موضوع را بیان کردم، چون می‌خواستم عنوان کنم،بسیاری از الگوریتم‌های فشرده سازی از الگوی تکراری بودن داده، جهت فشرده سازی داده‌ها استفاده می‌کنند، به همین جهت فشرده سازی در روش Row Store به حداقل می‌رسد و فضای اشغال شده در حافظه دراین روش بسیار زیاد خواهد بود. اما در روش Column Store ، امکان تکراری بودن مقادیر یک ستون بسیار زیاد است، بطور مثال ممکن است تعداد افرادی را که نام شهر آنها "تهران" باشد مثلا 20 بار تکرار شده باشد، و چون در روش Column Store، ستون‌ها در یک Page ذخیره می‌شوند، بنابراین امکان استفاده از الگوریتمهای فشرده سازی در این روش بسیار بالا می‌باشد، در نتیجه مقدار فضایی را که در حافظه یا دیسک سخت توسط این روش اشغال می‌شود، بسیار کمتر از روش Row Store است.
چه موقع می‌توانیم از Column Store استفاده نماییم:

در تعریف Column Store گفته بودم، روش فوق، جهت بهبود بخشیدن به زمان و سرعت پاسخگویی به Queryهای اجرا شده روی دیتابیس‌های با حجم داده ای بسیار بالا(Data Warehouse) می‌باشد، به بیان ساده‌تر Column Store را روی دیتابیس‌های offline یا دیتابیسهایی که صرفا جهت گزارش گیری مورد استفاده قرار می‌گیرند، تنظیم می‌نمایند.در واقع با تنظیم Column Store Index روی Databaseهای بزرگ مانند Databaseهای بانک‌ها که حجم داده ای میلیونی در جداول آنها وجود دارد، سرعت پاسخگویی Query ها، چندین برابر افزایش می‌یابد.
در یک جدول می‌توانید، هم Column Store Index داشته باشید و هم یک Row Store Index (منظور یک Clustered Index می باشد)

Syntax برای ایجاد Column Store Index به شرح ذیل می‌باشد:

```
CREATE [ NONCLUSTERED ] COLUMNSTORE INDEX index_name
    ON <object> ( column  [ ,...n ] )
    [ WITH ( <column_index_option> [ ,...n ] ) ]
    [ ON {
           { partition_scheme_name ( column_name ) }
           | filegroup_name
           | "default"
         }
    ]
[ ; ]

<object> ::=
{
    [database_name. [schema_name ] . | schema_name . ]
     table_name
```

```
{

<column_index_option> ::=
{
      DROP_EXISTING = { ON | OFF }
    | MAXDOP = max_degree_of_parallelism
 }
```

یک Column Store Index می‌بایست از نوع NONCLUSTERED باشد.

مثال از یک Column Store Index :

```
CREATE NONCLUSTERED COLUMNSTORE INDEX [IX_MyFirstName_ColumnStore]
ON [Test]
(Firstname)
```

در قطعه کد بالا، یک Column Store Index به نام Ix_MyFirstname_ColumnStore روی فیلد Firstname از جدول Test ایجاد شده است.

محدودیت‌های استفاده از Column Store Index به اختصار به شرح ذیل می‌باشد:

زمانی که در یک جدول، یک Column Store Index ایجاد نماییم، جدول ما در حالت Read-only قرار می‌گیرد، بطوریکه از آن پس اختیار Delete،Update و Insert روی جدول فوق را نخواهیم داشت. برای اینکه بتوانید عملیات Update، Insert یا Delete را انجام دهید، میبایست Column Store Index جدول مربوطه را Disable نمایید، و برای فعال نمودن Column Store Index، می‌بایست آن را Rebuild نمایید، با کلیک راست روی ایندکس ایجاد شده در SQL Server2012 موارد Disable و Rebuild قابل مشاهده می‌باشد.

یا بوسیله Scriptهای زیر می‌توانید، عملیات Disable یا Rebuild را روی Column Store Index انجام دهید:

```
ALTER INDEX [IX_MyFirstName_ColumnStore] ON [Test] DISABLE
ALTER INDEX [IX_MyFirstName_ColumnStore] ON [Test] Rebuild
```

بیشتر از یک Column Store Index نمی‌توانید روی یک جدول ایجاد نمایید.
در صورتی که تمایل داشته باشید بوسیله Alter ، نوع فیلدی (Type)، را که Column Store Index روی آنها اعمال گردیده است، تغییر دهید، در ابتدا می‌بایست Column Store Index، خود را Drop یا حذف نمایید، سپس عملیات Alter را اعمال کنید، در غیر اینصورت با خطای SQL Server مواجه می‌شوید.
یک Column Store Index می‌تواند روی 1024 ستون در یک جدول اعمال گردد.
یک Column Store Index نمی توانند، Unique باشد و نمی‌توان از آن به عنوان Primary Key یا Foreign Key استفاده نمود.

یاد آوری: با توجه به مزایای استفاده از Column Store Index، باید بگویم که در حجم‌های داده ای کم استفاده از Row Store Index بهتر می‌باشد. پیشنهاد مایکروسافت برای استفاده از Column Store Index برای دیتابیس‌های با حجم داده ای بسیار بالا می‌باشد.
موفق باشید

منابع: ”Columnstore Indexes: A New Feature in SQL Server known as Project “Apollo
Columnstore Indexes
Fundamentals of Columnstore Index
SQL Server 2012 Column Store Index Example

نظرات خوانندگان

نویسنده: صالح
تاریخ: ۱۳۹۱/۰۷/۳۰ ۰:۵۷

درود بر شما
بسیار بهره جستم از مطلبتون. عالی بود.

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۷/۳۰ ۱۱:۱۶

ممنون از لطفا شما

نویسنده: علیرضا جهانشاهلو
تاریخ: ۱۳۹۱/۰۷/۳۰ ۲۲:۵۷

بسیار بهره بردیم از مطلب خوبتون
با تشکر

نویسنده: libertad
تاریخ: ۱۳۹۱/۰۸/۰۳ ۱۰:۳۴

آیا چنین امکانی در oracle هم وجود دارد؟

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۸/۰۳ ۱۲:۳۹

بله، چنین امکانی موجود است،برای اطلاعات بیشتر می‌توانید به آدرس زیر مراجعه کنید:
Compressing Columns

نویسنده: مهران زند
تاریخ: ۱۳۹۱/۰۸/۰۵ ۱۶:۲۸

بسیار عالی و ممنون. نتیجتا این گونه ایندکس گذاری برای دیتابیس‌های که درد حال رشد هستند یعنی مداوم در حال,update insert.. هستند به درد نخواهد خورد.

نویسنده: علی قمشلویی
تاریخ: ۱۳۹۱/۰۸/۰۵ ۲۰:۵۵

با سلام و تشکر به خاطر مطلب عالی و فوق العاده قابل فهمتون
یکی از سئوالاتی که همیشه برام هنگام کوئری گرفتن از دیتابیس پیش می‌آمد این بود که :

```
select c1 from A1 where c1=x

select * from A1 where c1=x
```

کدامیک دارای کارایی بیشتری می‌باشند که دراین مقاله به طور عالی توضیح داده شده بود.
البته جوابشو نمیدم تا دوستان این مقاله زیبا را از دست ندهند.

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۸/۰۵ ۲۲:۵۸

البته موردی که در این مقاله عنوان شده یکی از مشکلات Select * هست...

نویسنده: ali
تاریخ: ۱۳۹۲/۰۹/۰۵ ۲۳:۴۵

با سلام یک جایی مطلب شما گنگ ماند " یعنی علاوه بر ستونهای C1 و C2، اطلاعات مربوط به ستونهای C3 و C4 نیز درون حافظه قرار می‌گیرد " پس چرا میگن تا آنجا که میشه برای بالا بردن performance در دستور select حتما تعداد فیلدهای آن مشخص باشد با آنچه شما گفتید c3 وc4 درون حاظه لود میشوند اینطور نیست ممنون

عنوان: آشنایی با Contained Databases در SQL Server 2012
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۸/۱۰ ۲۲:۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SQL Server 2012

مقدمه: بعضی وقت‌ها به هر دلیلی لازم است پایگاه داده برنامه هایمان را به سرور دیگری انتقال دهیم . برخلاف Oracle که انتقال پایگاه داده به سرور دیگر کار مشکل و تخصصی می‌باشد در SQL Server براحتی با یک Detach و Attach مجدد این انتقال انجام خواهد شد. اما اطلاعات Logins در پایگاه داده Master سرور ذخیره شده است و با فایل پایگاه داده انتقال نمی‌یابد و به هیمن خاطر برخی مواقع پس از Attach پایگاه داده با پیغام The database is not accessible یا Cannot open user default database مواجع خواهیم شد که مشکل بخاطر همین نبود کاربران سرور قبلی و Owner متفاوت پایگاه داده می‌باشد که با استفاده از رویه ذخیره شده sp_change_users_login این مشکل قابل حل می‌باشد اما در SQL Server 2012 راحل بهتری فراهم شده است.

Contained Databases در Sql Server 2012

یکی از امکانات اضافه شده در SQL Server 2012 امکان Contained Databases هست که برای حل همین مشکل و رفع کامل وابستی پایگاه داده با Instance نصب شده و خاصیت قابل حمل کامل databaseها می‌باشد. در ادامه نحوه استفاده از این امکان را بررسی خواهیم کرد:

فعال کردن خاصیت Contained Databases

قبل از استفاده از Contained Databases می بایست این امکان را فعال کرد. برای این کار می‌توانید از SQL Server Management Studio یا T-SQL commands استفاده نمایید. بر روی نام Instance راست کلیت کنید و گزینه Properties را انتخاب نمایید. از گزینه Advanced که در شکل زیر مشاهده می‌نمایید خاصیت Enable Contained Databases را بر روی True قرار دهید.

یا می‌توانید از sp_configure این کار را انجام دهید.دستورات زیر این موضوع را نشان می‌دهد.

```
sp_configure 'show advanced options',1
GO
RECONFIGURE WITH OVERRIDE
GO
sp_configure 'contained database authentication',1
GO
RECONFIGURE WITH OVERRIDE
```

```
GO
```

ایجاد یا تغییر یک پایگاه داده از نوع Contained Databases

برای ایجاد یک پایگاه داده با این خاصیت یا تغییر پایگاه داده موجود کافیست مقدار گزینه Containment type را بر روی Partial قرار دهید. برای پایگاه داده موجود از پنجره Properties پایگاه داده صفحه Options را انتخاب کنید.

ایجاد یک کاربر برای پایگاه داده Contained Databases

برای تعریف یک کاربر در سطح پایگاه داده پوشه Security پایگاه داده خود را باز کنید بر روی پوشه Users راست کیلک و گزینه New User را انتخاب نمایید از گزینه User type که در شکل زیر نشان داده شده است SQL user with password را انتخاب نمایید و نام کاربر و رمز عبور و تکرار آن را وارد نمایید. کاربر ایجاد شده در سطح پایگاه داده می‌باشد و با انتقال به سرور دیگر نیر قابل دسترسی می‌باشد.

اتصال به پایگاه داده Contained Databases

برای اتصال به پایگاه داده کافیست در حالت SQL Server Authentication نام کاربری و رمز عبور جدید را وارد و گزینه Options را انتخاب و از برگه Additional Connection Parameters نام پایگاه داده مورد نظر را مانند شکل زیر وارد نمایید پس از ورود تنها پایگاه داده خود را مشاهده می‌نمایید. یکی از کاربرهای این قابلیت برای مدیران سرور پایگاه داده می‌باشد که بدون استفاده از مجوز sysadmin به کاربران اجازه دسترسی را می‌دهد.

Connect to Server
Microsoft SQL Server 2012
Login
Connection Properties
Additional Connection Parameters
Enter additional connection string parameters (will be sent in clear text):
Initial Catalog = < Database Name >
(Note: Connection string parameters override graphical selections on other panels)
Connect
Cancel
Help
Options <<

نظرات خوانندگان

نویسنده: ابراهیم شادافروغ
تاریخ: ۱۳۹۱/۰۸/۱۱ ۱۱:۵۳

مقاله عالی بود من با این مشکل در یکی از پروژه‌های کاری ام برخورد کردم و با مطالعه و تمرین و خطا موفق به پیدا کردن راه حل این مشکل شدم
خیلی خوبه که راه کار حل موضوعاتی این چنین در وب فارسی وجود داشته باشه

عنوان: آشنایی با SQL Server Data Tools
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۸/۲۴ ۱۵:۱۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, VisualStudio.NET 2012, SQL Server 2012, Data Tools

مقدمه

یکی از امکانات اضافه شده در Sql Server 2012 ابزار Sql Server Data Tools یا به اختصار SSDT می‌باشد. این ابزار در واقع جایگزین ابزار Business Intelligence Development Studio می باشد که همه امکانات قبلی را داشته و همچنین قابلیت‌های جدیدی نیز به آن اضافه شد است. اما کاربرد این ابزار ایجاد محیطی یکپارچه برای طراحی و توسعه تمامی نسخه‌های پایگاه داده Sql Server با استفاده از Visual Studio می‌باشد. در این مقاله من به بیشتر بر روی امکانات زیر تمرکز دارم:

Installation

Creating a SQL Server Database Project

Modify Database Schema

Schema Compare and Update

Snapshot Project

Publish

نصب Sql Server Data Tools

آخرین نسخه این ابزار را از این آدرس دانلود کنید یا زمان نصب Sql Server 2012 گزینه آن را انتخاب کنید

ایجاد یک پروژه جدید از نوع SQL Server Database Project

پس از نصب SSDT شما از طریق Visual Studio 2012 Shell که همراه SqlServer 2012 نصب می‌شود یا با Visual Stadio 2012 یک پروژه جدید از نوع SQL Server Database Project ایجاد کنید.

Server Database Project Project به شما امکان توسعه پایگاه داده Sql Server را با استفاده از محیط یکپارچه Visual Studio با در اختیار گذاشتن ابزارهای همچون navigation, intellisense, validation, debugging, declarative editing و غیره را می‌دهد. شی پایگاه داده شما در پروژه و فایل‌های مجزا ذخیره می‌شود مثل آن که شما در حال توسعه برنامه C# یا VB.NET هستید.

ورود پایگاه داده موجود به پروژه SQL Server Database

شما می‌توانید شمای پایگاه داده موجود یا SQL Server DAC Package File (.dacpac) یا هر T-SQL دیگری را با راست کلیک بر روی پروژه و انتخاب گزینه Import به پروژه خود اضافه و تغییرات لازم را اعمال نماید شکل زیر پنجره Import Database را نشان می‌دهد:

پنجره Solution Explorer امکانات زیر را در اختیار شما قرارمی دهد:

- اضافه کردن اشیای همچون Table، View و غیره با راست کیلک کردن بر روی پروژه و انتخاب گزینه Add
- ویرایش اشیا موجود با دبل کیلک کردن بر روی اشیا
- مقایسه شمای پایگاه داده با پایگاه داده دیگر و یا Microsoft SQL Server DAC Package File کافیست بر روی پروژه راست کیلک و گزینه Schema Compare را انتخاب نمایید.
- ایجاد یک Snapshot از شمای پایگاه داده در یک Microsoft SQL Server DAC Package File . یک snapshot برای ایجاد یک پایگاه داده یا ورود در پروژه ای دیگر یا در مقایسه دو پایگاه داده کاربرد دارد.
- انتشار پایگاه داده که امکان ایجاد پایگاه داده یر روی یک سرور SQL Server را فراهم می‌نماید.

مقایسه شمای دو پایگاه داده با هم

SSDT ابزاری برای مقایسه دو پایگاه داده و بروز رسانی پایگاه داده مقصد از روی شمای منبع از طریق ایجاد یک Script یا به صورت مستقیم را می‌دهد.

انتشار پروژه با استفاده از SQL Server Data Tools

این ابزار امکان گسترش پروژه شما بر روی SQL Server 2005, 2008, 2008 R2, 2012, یا SQL Azure instance را می‌دهد. برای انتشار کافیت از پنجره Solution Explorer بر روی پروژه راست کیلک و گزینه Publish را انتخاب کنید.

پنجره SQL Server Object Explorer

این پنجره با اتصال به سرور SQL Server امکان دسترسی و مدیریت پایگاه داده را به ما می‌دهد. علاوه بر دسترسی به اجزای یک پایگاه داده امکان مدیریت پوشه‌های Server Objects , Security نیز فراهم می‌باشد. همچنین امکان اتصال به Sql Server Express , localdb نیز وجود دارد.

عنوان: **مشکل ارتباط با SQL Server در لوکال**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۹/۰۸ ۱۱:۴۷
آدرس: www.dotnettips.info
برچسب‌ها: Debugging, SQL Server, T-SQL

در حین کار با SQL Server نیاز به دیباگ اسکریپتی طولانی و اورژانسی T-SQL بود که در محیط Management Studio با مشکل زیر برخورد کردم:

در این مورد نظرات و پیشنهادات زیادی از جمله ریستارت سرویس SQL Server و تعویض کلمه عبور لاگین و یا پاک کردن کلمات عبور کش شده در سیستم و حتی ریستارت کامپیوتر :) از دوستان همکار در فروم‌های موجود در اینترنت گذاشته شده بود و در گوشه ای هم اشاره به '.' شده بود که طبق عادت همیشگی برای لاگین به بانک استفاده میکردم و خواسته شده بود که برای لاگین لوکال به SQL Server از نام کامپیوتر بجای '.' استفاده شود که مفید فایده بود.

عنوان: استفاده از Sparse Columns در SQL Server 2012
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۹/۱۱ ۱۶:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, SQL Server 2012, T-SQL, DML

مقدمه

مقدار null به معنی پوچ و هیچ می‌باشد اما زمانی که در مقدار دهی جداول از آن استفاده می‌نمایم با توجه به نوع آن ستون فضای متفاوتی اشتغال می‌نماید. شاید در پایگاه داده‌های کوچک زیاد مطرح نباشد اما زمانی که حداقل چند گیگ حجم آن باشد و فرضا 20 تا 30 درصد آن از مقادیر null پر شده باشد فضای زیادی از پوچ گرفته شده است این در حالی است که خیلی از توسعه دهندگان اصلا به اهمیت استفاده از null توجهی نمی‌کنند و از مقادیری غیر معتبری مثل 0 یا 1- در آن ستون به جای null استفاده می‌کنند.

SQL Server Sparce Columns

sparse column یا ستون‌های تنک قابلیتی از که از SQL Server 2008 اضافه شده و به ستون‌های عادی امکان استفاده بهینه از فضای ذخیره شده برای مقادیر null را می‌دهد. در واقع sparse column فضای مورد نیاز برای مقادیر null نسبت به مقادیر غیر null را کاهش می‌دهد. با استفاده از sparse column فضای ذخیره شده حداقل 20 تا 40 درصد کمتر خواهد شد.

ویژگی‌های Sparse Columns

SQL Server Database Engine از کلمه کلیدی SPARSE برای تعریف یک ستون که مقادیر آن می‌بایست بهینه شود استفاده می‌نماید.

نمای Catalog جداول با ستون sparse شبیه جداول معمولی می‌باشد.

مقدار برگشتی از تابع COLUMNS_UPDATED با ستون sparce متفاوت از ستون معمولی است.

در نوع داده‌های زیر امکان استفاده از sparce columns را ندارند:

text	geography
timestamp	geometry
user-defined data types	image
	ntext

sparse column فضای بیشتری برای ذخیره داده‌های غیر null نسبت به داده‌های نشانه گذاری نشده با SPARSE لازم دارد و این فضا4 بایت بیشتر از ستون معمولی است. برآورد فضای ذخیره شده براساس نوع داده با طول ثابت در جدول زیر آورده شده است:

نوع داده	بایت بدون sparse	بایت sparse	درصد null
bit	0.125	5	98%
tinyint	1	5	86%
smallint	2	6	76%
int	4	8	64%
bigint	8	12	52%
real	4	8	64%
float	8	12	52%
smallmoney	4	8	64%
money	8	12	52%

نوع داده	بایت بدون sparse	بایت sparse	درصد null
smalldatetime	4	8	64%
datetime	8	12	52%
uniqueidentifier	16	20	43%
date	3	7	69%

نوع داده با دقت - وابسته به طول

نوع داده	بایت بدون sparse	یابت sparse	درصد null
(datetime(2	6	10	57%
(datetime(2	8	12	52%
(time(0	3	7	69%
(time(7	5	9	60%
(datetimetoffset(0	8	12	52%
(datetimetoffset (7	10	14	49%
decimal/numeric(1,s)	5	9	60%
decimal/numeric(38,s)	17	21	42%
vardecimal(p,s)			

نوع داده - داده وابسته به طول

نوع داده	بایت بدون sparse	یابت sparse	درصد null
sql_variant	2*	2*	60%
varchar or char	2*	4*+	60%
nvarchar or nchar	2*	4*	60%
varbinary or binary	2*	4*	60%
xml	2*	4*	60%
hierarchyid	2*	4*	60%

محدویت‌های استفاده از Sparse columns

sparse column می بایست nullable باشد و نمی‌تواند ROWGUIDCOL یا IDENTITY باشد.

sparse column مقدار پیش فرض نمی‌تواند داشته باشد

ستون محاسبه ای نمی‌تواند sparse باشد

sparse column نمی‌تواند بخشی از clustered index یا unique primary key index باشد

sparse column نمی تواند بخشی از user-defined table باشد

مثالی از کاربرد Sparse columns

```
CREATE TABLE Employees_sparse (
   EMP_ID INT IDENTITY(5001,1) PRIMARY KEY,
   SSN CHAR(9) NOT NULL,
   TITLE CHAR(10) SPARSE NULL,
```

```
   FIRSTNAME VARCHAR(50) NOT NULL,
   MIDDLEINIT CHAR(1) SPARSE NULL,
   LASTNAME VARCHAR(50) NOT NULL,
   EMAIL CHAR(50) SPARSE NULL)
GO
```

```
CREATE TABLE Employees (
   EMP_ID INT IDENTITY(5001,1) PRIMARY KEY,
   SSN CHAR(9) NOT NULL,
   TITLE CHAR(10) NULL,
   FIRSTNAME VARCHAR(50) NOT NULL,
   MIDDLEINIT CHAR(1) NULL,
   LASTNAME VARCHAR(50) NOT NULL,
   EMAIL CHAR(50) NULL)
GO
```

در این دو جدول یکی با سه ستون Sparse و دیگری بدون Sparse ایجاد شده و با 50000 ردیف داده پر شده است حال با رویه ذخیره شده sp_spaceused می‌توان فضای ذخیره شده دو جدول را باهم مقایسه نمود.

```
sp_spaceused 'Employees'
GO
sp_spaceused 'Employees_sparse'
```

Results | Messages

	name	rows	reserved	data	index_size	unused
1	Employees	50000	5064 KB	5008 KB	32 KB	24 KB

	name	rows	reserved	data	index_size	unused
1	Employees_sparse	50000	3080 KB	3064 KB	16 KB	0 KB

البته ذکر این نکته گفتی است که بهتر است از این تکنیک برای جداولی که تعداد زیادی ستون null دارند استفاده شود.

عنوان: **تغییر نام پایگاه داده و فایل هایش در SQL Server 2012**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۹/۱۲ ۲۱:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, SQL Server 2012, T-SQL

بعضی وقت‌ها به هر علتی لازم است پایگاه داده و فایل هایش را تغییر نام دهیم. اگر در اینترنت جستجو کنیم روش‌های مختلفی برای تغییر نام مثل تغییر با Management Studio یا T-SQL یا روش‌های دیگری یافت می‌شود. اما اکثرا در بین انجام به مشکلی غیر قابل پیش بینی بر می‌خوریم. پایگاه داده در حالت آفلاین یا Pending قرار گرفته و به خطاهای نا مفهومی بر می‌خوریم. حالا باید دوباره کلی جستجو کنیم تا مشکل بوجود آمده را حل نمائیم.

بهترین روش تغییر نام پایگاه داده

بهترین روش استفاده از قابلیت Copy Database خود SQL Server است است که به راحتی این کار را برای ما انجام می‌دهد.
بر روی پایگاه داده مورد نظر راست کلیک کرده و از گزینه Tasks گزینه Copy Database را انتخاب کنید.
پس از ظاهر شدن پنجره کپی گزینه Next را انتخاب و در مرحله مبدا و مقصد، سرور جاری را انتخاب کنید و به مرحله بعد بروید.
این برای زمانی است که شما می‌خواهید پایگاه داده را در سرور دیگری کپی نماید
در پنجره Transfer Method دو روش Detach and Attach و استفاده از SQL Management Object وجود دارد که با همان روش اول به مرحله بعد بروید

در مرحله بعد نام پایگاه داده شما انتخاب شده به مرحله بعد بروید.
مرحله بعد پیکربندی پایگاه داده مقصد می‌باشد که نام و مسیر پایگاه داده جدید را می‌توانید مشخص نمایید.

این عملیات با SQL Server Agent صورت می‌پذیرد به همین خاطر Agent می‌بایست نصب و Start شده باشد.
با انتخاب گزینه Next مراحل بعد را رد کرده تا عملیات آغاز شود.
در مرحله آخر پایگاه داده قبلی را حذف نمایید.

نظرات خوانندگان

نویسنده: اردلان شاه قلی
تاریخ: ۱۳۹۲/۰۴/۰۶ ۱۱:۵۲

خیلی متشکرم.
البته یک سوال هم دارم. تصور کنید حجم پایگاه داده‌ی من 400 گیگابایت می‌باشد و کل فضای خالی موجودم 350 گیگابایت می‌باشد در این صورت امکان استفاده از این روش وجود ندارد؟

عنوان: **آشنایی با Window Function ها در SQL Server بخش اول**
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۹/۱۷ ۲۳:۲۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SQL Server 2012, T-SQL

Window Functionها برای اولین بار در نسخه SQL Server 2005 ارائه گردیدند، و در ورژن‌های جدیدتر SQL Server، به تعداد این فانکشنها افزوده شده است.

تعریف Window Function :

معمولا از این نوع فانکشنها روی مجموعه ای از ROWهای یک جدول، در جهت اعمال عملیاتهای محاسباتی ،ارزیابی داده ها، رتبه بندی و غیرو... استفاده می‌گردد، به بیان ساده‌تر بوسیله Window Functionها می‌توان، ROWهای یک جدول را گروه بندی نمود. و روی گروه‌ها از توابع جمعی (Aggregate Functions) استفاده کرد. این نوع فانکشنها از قابلیت و انعطاف پذیری زیادی برخوردار می‌باشند، و بوسیله آنها می‌توان نتایج (خروجی) بسیار مفیدی از Query ها، بدست آورد، معمولا از این نوع فانکشنها در Data Mining (داده کاوی) و گزارشگیری‌ها استفاده می‌گردد. و آگاهی و روش استفاده از Window Functionها برای برنامه نویسان و DBA ها، می‌تواند بسیار مفید باشد.
مفهوم Window Function مطابق استاندارد ISO و ANSI می‌باشد، و دیتابیس هایی همچون Oracle،DB2،Sybase از آن پشتیبانی می‌نمایند.برای اطلاعات بیشتر می‌توانید به سایت‌های زیر مراجعه کنید: SQL:2003
و SQL:2008

کلمه "Window" در Window Function، به مجموعه ROW هایی اشاره می‌کند، که محاسبات و ارزیابی و غیرو... روی آنها اعمال می‌گردد.
Window Functionها برای ارائه قابلیت‌های خود، از Over Clause استفاده می‌کنند. اگر مقاله آشنایی با Row_Number،Rank،Dense_Rank،NTILE را مطالعه کرده باشید، می‌توان هریک از آنها را یک Window Function دانست.
برای شروع، به بررسی Over Clause می‌پردازیم، و Syntax آن به شرح ذیل می‌باشد:

```
OVER (
       [ <PARTITION BY clause> ]
       [ <ORDER BY clause> ]
       [ <ROW or RANGE clause> ]
      )

<PARTITION BY clause> ::=
PARTITION BY value_expression , ... [ n ]

<ORDER BY clause> ::=
ORDER BY order_by_expression
    [ COLLATE collation_name ]
    [ ASC | DESC ]
    [ ,...n ]

<ROW or RANGE clause> ::=
{ ROWS | RANGE } <window frame extent>

<window frame extent> ::=
{   <window frame preceding>
  | <window frame between>
}

<window frame between> ::=
  BETWEEN <window frame bound> AND <window frame bound>

<window frame bound> ::=
{   <window frame preceding>
  | <window frame following>
}

<window frame preceding> ::=
{
    UNBOUNDED PRECEDING
  | <unsigned_value_specification> PRECEDING
  | CURRENT ROW
}
```

```
<window frame following> ::=
{
    UNBOUNDED FOLLOWING
  | <unsigned_value_specification> FOLLOWING
  | CURRENT ROW
}

<unsigned value specification> ::=
{  <unsigned integer literal> }
```

OVER دارای سه آرگومان اختیاری است که هر کدام را به تفصیل بررسی می‌کنیم:

1- PARTITION BY clause : بوسیله این پارامتر می‌توانیم Rowهای یک جدول را گروه بندی نماییم. این پارامتر یک value_expression می پذیرد. یک Value_expression می‌تواند نام یک ستون ، یک Scalar Function ، Scalar Subquery و غیرو باشد.

2- ORDER BY clause : از نامش مشخص است و برای Sort استفاده می‌شود، و ویژگی‌های Order By در آن اعمال می‌گردد. به جز Offset.

3- ROW or RANGE clause :این پارامتر بیشتر برای محدود نمودن Row در یک Partition (گروه) مورد استفاده قرار می‌گیرد، به عنوان مثال نقطه شروع و پایان را می‌توان بوسیله پارامتر فوق تعیین نمود.

Row و Range نسبت به هم یک تفاوت عمده دارند،و آن این است که، اگر از ROW Clause استفاده نمایید، ارتباط ROWهای قبلی یا بعدی، نسبت به Row جاری،بصورت فیزیکی (physical association) سنجیده می‌شود، بطوریکه با استفاده از Range Clause ارتباط سطرهای قبلی و بعدی، نسبت به سطر جاری بصورت منطقی (logical association) در نظر گرفته می‌شود. ممکن است درک این مطلب کمی سخت باشد، در ادامه با مثالهایی که بررسی می‌نماییم، براحتی تفاوت این دو را متوجه می‌شوید.

Row یا Range در قالب‌های متفاوتی مقدار می‌پذیرند، که هر کدام را بررسی می‌کنیم:

UNBOUNDED PRECEDING : بیانگر اولین سطر Partition می‌باشد. UNBOUNDED PRECEDING فقط نقطه شروع را مشخص می‌نماید.

UNBOUNDED FOLLOWING : بیانگر آخرین سطر Partition می‌باشد. UNBOUNDED FOLLOWING فقط نقطه پایانی را مشخص می‌نماید.

CURRENT ROW : اولین سطر جاری یا آخرین سطر جاری را مشخص می‌نماید.

n PRECEDING یا <unsigned value specification> PRECEDING : تعداد سطرهای قبل از سطر جاری را تعیین می‌کند، n یا <unsigned value specification> تعداد سطرهای قبل از سطر جاری را تعیین می‌نماید. از n PRECEDING نمی توان برای Range استفاده نمود.

n FOLLOWING یا <unsigned value specification> FOLLOWING : تعداد سطرهای بعد از سطر جاری را تعیین می‌کند، n یا<unsigned value specification> تعداد سطر های بعد از سطر جاری را تعیین می‌نماید. از n FOLLOWING نمی توان برای Range استفاده نمود.

BETWEEN <window frame bound > AND <window frame bound> : از چارچوب فوق برای Range و Row می‌توان استفاده نمود، و نقطه آغازین و نقطه پایانی توسط قالب فوق تعیین می‌گردد. نکته قابل توجه آن است که نقطه پایانی نمی‌تواند، کوچکتر از نقطه آغازین گردد.

در ادامه برای درک هرچه بیشتر تعاریف بیان شده، چندین مثال می‌زنیم و هر کدام را بررسی می‌نماییم:

در ابتدا Script زیر را اجرا نمایید، که شامل جدولی به نام Revenue (سود،درآمد) و درج چند درکورد در آن:

```
CREATE TABLE REVENUE
(
[DepartmentID] int,
[Revenue] int,
[Year] int
);

insert into REVENUE
values (1,10030,1998),(2,20000,1998),(3,40000,1998),
 (1,20000,1999),(2,60000,1999),(3,50000,1999),
 (1,40000,2000),(2,40000,2000),(3,60000,2000),
 (1,30000,2001),(2,30000,2001),(3,70000,2001)
```

مثال اول : می‌خواهیم براساس فیلد DepartmentID جدول Revenue را Partition بندی نماییم و از توابع جمعی AVG و SUM روی فیلد درآمد(Revenue) استفاده کنیم.

ابتدا Script زیر را اجرا می‌کنیم:

```
select *,
 avg(Revenue) OVER (PARTITION by DepartmentID) as AverageRevenue,
 sum(Revenue) OVER (PARTITION by DepartmentID) as TotalRevenue
from REVENUE
order by departmentID, year;
```

خروجی بصورت زیر خواهد بود:

	DepartmentID	Revenue	Year	AverageRevenue	TotalRevenue
1	1	10030	1998	25007	100030
2	1	20000	1999	25007	100030
3	1	40000	2000	25007	100030
4	1	30000	2001	25007	100030
5	2	20000	1998	37500	150000
6	2	60000	1999	37500	150000
7	2	40000	2000	37500	150000
8	2	30000	2001	37500	150000
9	3	40000	1998	55000	220000
10	3	50000	1999	55000	220000
11	3	60000	2000	55000	220000
12	3	70000	2001	55000	220000

مطابق شکل، جدول براساس فیلد DepartmentID به سه Partition تقسیم شده است، و عملیات میانگین و جمع روی فیلد Revenue انجام شده است و عملیات Sort روی هرگروه بطور مستقل انجام گرفته است. چنین کاری را نمی‌توانستیم بوسیله Group By انجام دهیم.

مثال دوم : نحوه استفاده از ROWS PRECEDING،در این مثال قصد داریم عملیات جمع را روی فیلدRevenue انجام دهیم. بطوریکه جمع هر مقدار برابر است با سه مقدار قبلی + مقدار جاری:

لطفا رکوردهای زیر را به جدول فوق درج نمایید:

```
insert into REVENUE
 values(1,90000,2002),(2,20000,2002),(3,80000,2002),
 (1,10300,2003),(2,1000,2003), (3,90000,2003),
 (1,10000,2004),(2,10000,2004),(3,10000,2004),
 (1,20000,2005),(2,20000,2005),(3,20000,2005),
 (1,40000,2006),(2,30000,2006),(3,30000,2006),
 (1,70000,2007),(2,40000,2007),(3,40000,2007),
 (1,50000,2008),(2,50000,2008),(3,50000,2008),
 (1,20000,2009),(2,60000,2009),(3,60000,2009),
 (1,30000,2010),(2,70000,2010),(3,70000,2010),
 (1,80000,2011),(2,80000,2011),(3,80000,2011),
 (1,10000,2012),(2,90000,2012),(3,90000,2012)
```

سپس Script زیر را اجرا می‌نماییم:

```
select Year, DepartmentID, Revenue,
sum(Revenue) OVER (PARTITION by DepartmentID ORDER BY [YEAR]
             ROWS BETWEEN 3 PRECEDING AND CURRENT ROW) as Prev3
From REVENUE order by departmentID, year;
```

خروجی :

	Year	DepartmentID	Revenue	Prev3
1	1998	1	10030	10030
2	1999	1	20000	30030
3	2000	1	40000	70030
4	2001	1	30000	100030
5	2002	1	90000	180000
6	2003	1	10300	170300
7	2004	1	10000	140300
8	2005	1	20000	130300
9	2006	1	40000	80300

در Script بالا، جدول را براساس فیلد DepartmentID گروه بندی می‌کنیم، که سه گروه ایجاد می‌شود، هر گروه را بطور مستقل، روی فیلد Year بصورت صعودی مرتب می‌نماییم. حال برای آنکه بتوانیم سیاست جمع، روی فیلد Revenue، را پیاده سازی نماییم ، قطعه کد زیر را در Script بالا اضافه کردیم.

```
ROWS BETWEEN 3 PRECEDING AND CURRENT ROW) as Prev3
```

برای شرح چگونگی استفاده از PRECEDING،فقط به شرح گروه اول بسنده می‌کنیم. مقدار جمع فیلد Revenue سطر اول، که قبل از آن سطری وجود ندارد، برابر است با مقدار خود، یعنی 10030، مقدار جمع فیلد Revenue سطر دوم برابر است با حاصل جمع مقدار فیلدRevenue سطر اول و دوم ، یعنی 30030 . این روند تا سطر چهار ادامه دارد، اما برای بدست آوردن مقدار جمع فیلدRevenue سطر پنجم، مقدار جمع فیلد Revenue سطر دوم،سوم،چهارم و پنجم در نظر گرفته می‌شود، و مقدار فیلدRevenue سطر اول در حاصل جمع در نظر گرفته نمی‌شود،بنابراین مقدار جمع فیلد Revenue سطر پنجم برابر است با 180000. در صورت مسئله گفته بودیم، مقدار جمع فیلد Revenue هر سطر جاری برابر است با حاصل جمع مقدارسطر جاری و مقادیر سه سطر ماقبل خود.

مثال سوم: نحوه استفاده از ROWS FOLLOWING ، این مثال عکس مثال دوم است، یعنی حاصل جمع مقدار فیلد Revenue هر سطر برابر است با حاصل جمع سطر جاری با سه سطر بعد از خود. بنابراین Script زیر را اجرا نمایید:

```
select Year, DepartmentID, Revenue,
 sum(Revenue) OVER (PARTITION by DepartmentID ORDER BY [YEAR]
              ROWS BETWEEN CURRENT ROW AND 3 FOLLOWING) as Next3
From REVENUE order by departmentID, year;
```

خروجی :

	Year	DepartmentID	Revenue	Next3
1	1998	1	10030	100030
2	1999	1	20000	180000
3	2000	1	40000	170300
4	2001	1	30000	140300
5	2002	1	90000	130300
6	2003	1	10300	80300
7	2004	1	10000	140000
8	2005	1	20000	180000

مطابق شکل مقدار جمع فیلد اول برابراست با حاصل جمع مقدار سطر جاری و سه سطر بعد از آن.

نکته ای که در مثالهای دوم و سوم،می بایست به آن توجه نمود، این است که در زمان استفاده از Row یا Range ، استفاده از Order by در Partition الزامی است، در غیر این صورت با خطا مواجه می‌شوید.

```
select Year, DepartmentID, Revenue,
  sum(Revenue) OVER (PARTITION by DepartmentID --ORDER BY [YEAR]
                     ROWS BETWEEN CURRENT ROW AND 3 FOLLOWING) as Next3
From REVENUE order by departmentID, year;
```

Messages

```
Msg 10756, Level 15, State 1, Line 2
Window frame with ROWS or RANGE must have an ORDER BY clause.
```

نحوه استفاده از UNBOUNDED PRECEDING ، این امکان در T-SQL Server 2012 افزوده شده است.

مثال چهار: در این مثال می‌خواهیم کمترین سود بدست آمده در چند سال را بدست آوریم:

ابتدا Script زیر را اجرا نمایید:

```
select Year, DepartmentID, Revenue,
       min(Revenue) OVER (PARTITION by DepartmentID ORDER BY [YEAR]
                          ROWS UNBOUNDED PRECEDING) as MinRevenueToDate
From REVENUE order by departmentID, year;
```

خروجی:

Results | Messages

	Year	DepartmentID	Revenue	MinRevenueToDate
1	1998	1	10030	10030
2	1999	1	20000	10030
3	2000	1	40000	10030
4	2001	1	30000	10030
5	2002	1	90000	10030
6	2004	1	10000	10000
7	2005	1	20000	10000
8	2006	1	40000	10000
9	2007	1	70000	10000
10	2008	1	50000	10000
11	2009	1	20000	10000
12	2010	1	30000	10000
13	2011	1	80000	10000
14	2012	1	10000	10000
15	1998	2	20000	20000
16	1999	2	60000	20000

طبق تعریف UNBOUNDED PRECEDING اولین سطر هر Partition را مشخص می‌نماید، و چون از PRECEDING استفاده کرده ایم، بنابراین مقایسه همیشه بین سطر جاری و سطرهای قبل از آن انجام می‌پذیرد. بنابراین خواهیم داشت، کمترین مقدار فیلد Revenue در سطر اول، برابر با مقدار خود می‌باشد، چون هیچ سطری ماقبل از آن وجود ندارد. در سطر دوم مقایسه کمترین مقدار، بین 20000 و 10030 انجام می‌گیرد، که برابر است با 10030، در سطر سوم، مقایسه بین مقادیر سطر اول،دوم و سطر سوم صورت می‌گیرد، یعنی کمترین مقدار بین 20000،40000 و 10030، بنابراین کمترین مقدار سطر سوم برابر است با 10030.
به بیان ساده‌تر برای بدست آوردن کمترین مقدار هر سطر، مقدار سطر جاری با مقادیر همه سطرهای ماقبل خود مقایسه می‌گردد.
برای بدست آوردن کمترین مقدار در سطر ششم، مقایسه بین مقادیر سطرهای اول،دوم،سوم،چهارم،پنجم و ششم صورت می‌گیرد که عدد 10000 بدست می‌آید و الی آخر...
نکته: اگر در Over Clause شرط Order by را اعمال نماییم، اما از Row یا Range استفاده نکنیم، SQL Server بصورت پیش فرض از قالب زیر استفاده می‌نماید:

```
RANGE UNBOUNDED PRECEDING AND CURRENT ROW
```

برای روشن‌تر شدن مطلب فوق مثالی می‌زنیم:
ابتدا Script زیر را اجرا نمایید، که شامل ایجاد یک جدول و درج چند رکورد در آن می‌باشد:

```
CREATE TABLE Employees (
    EmployeeId INT IDENTITY PRIMARY KEY,
    Name VARCHAR(50),
    HireDate DATE NOT NULL,
    Salary INT NOT NULL
)
GO
INSERT INTO Employees (Name, HireDate, Salary)
VALUES
```

```
    ('Alice', '2011-01-01', 20000),
    ('Brent', '2011-01-15', 19000),
    ('Carlos', '2011-02-01', 22000),
    ('Donna', '2011-03-01', 25000),
    ('Evan', '2011-04-01', 18500)
GO
```

سپس Script زیر را اجرا نمایید:

```
SELECT
    Name,
    Salary,
    AVG(Salary) OVER(ORDER BY HireDate) AS avgSalary
FROM Employees
GO
```

خروجی :

	Name	Salary	avgSalary
1	Alice	20000	20000
2	Brent	19000	19500
3	Carlos	22000	20333
4	Donna	25000	21500
5	Evan	18500	20900

حال اگر Script زیر را نیز اجرا نمایید، خروجی آن مطابق شکل بالا خواهد بود:

```
SELECT
    Name,
    Salary,
    AVG(Salary) OVER(ORDER BY HireDate
                  RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS avgSalary
FROM Employees
GO
```

توضیح درباره Script بالا، در این روش برای بدست آوردن میانگین هر سطر، مقدار سطر جاری با مقادیر سطرهای ماقبل خود جمع و تقسیم بر تعداد سطر می‌شود.

سطر دوم 20000 + 19000 تقسیم بر دو برابر است با 19500

میانگین سطر پنجم، حاصل جمع فیلد Salary همه مقادیر سطرها تقسیم بر 5

*** اگر بخواهید بوسیله Over Clause ، میانگین همه سطرها یکسان باشد می‌توانید از Script زیر استفاده نمایید:

```
SELECT
    Name,
    Salary,
    AVG(Salary) OVER(ORDER BY HireDate
                        RANGE
                        BETWEEN UNBOUNDED PRECEDING
                        AND UNBOUNDED FOLLOWING
                    ) AS avgSalary
FROM Employees
GO
```

خروجی :

	Name	Salary	avgSalary
1	Alice	20000	20900
2	Brent	19000	20900
3	Carlos	22000	20900
4	Donna	25000	20900
5	Evan	18500	20900

منظور از ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING یعنی در محاسبه میانگین برای هر سطر تمامی مقادیر سطرهای دیگر در نظر گرفته شود.

پایان بخش اول

امیدوارم مفید واقع شده باشد.

نظرات خوانندگان

نویسنده: ناصر
تاریخ: ۱۳۹۱/۰۹/۱۸ ۰:۲۳

بسیار عالی. فقط خواستم تشکری کرده باشم :)

نویسنده: reza
تاریخ: ۱۳۹۱/۰۹/۱۸ ۰:۳۳

SQLQuery2.sql - (l...Er-PC\na3Er (51))* | SQLQuery1.sql - (l...Er-PC\na3Er (54))* | NA3ER-PC.learn - dbo.REVENUE

```
select Year, DepartmentID, Revenue,
 sum(Revenue) OVER (PARTITION by DepartmentID ORDER BY [YEAR]
              ROWS BETWEEN 3 PRECEDING AND CURRENT ROW) as Prev3
From REVENUE order by departmentID, year;
```

Messages

```
Msg 102, Level 15, State 1, Line 3
Incorrect syntax near 'ROWS'.
```

علت این خطا چیست ؟ آیا این دستور مال 2012 هستش ؟ جون من تو 2008 تست میکنم

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۹/۱۸ ۶:۸

سلام

Syntax ارائه شده Over Clause در SQL Server 2008 شامل دو پارامتر است :

```
Ranking Window Functions
< OVER_CLAUSE > :: =
    OVER ( [ PARTITION BY value_expression , ... [ n ] ]
           <ORDER BY_Clause> )

Aggregate Window Functions
< OVER_CLAUSE > :: =
    OVER ( [ PARTITION BY value_expression , ... [ n ] ] )
```

در SQL Server 2012 پارامتر <ROW or RANGE clause> به Syntax فوق افزوده شد. بنابراین، Query هایی که از Row یا Range

استفاده کرده اند، در SQL Server 2008 با خطا مواجه می‌شوند.

نویسنده: معتمدی
تاریخ: ۱۳۹۱/۰۹/۱۸ ۱۲:۵۹

با سلام

آیا استفاده از این queryها برای محیط‌های transactional هم مناسب است؟ یا بیشتر در databaseهای آماری جهت تهیه گزارشات کاربرد دارد؟

از نظر زمان و هزینه‌ی اجرا و نیز تخصیص منابع سرور به آنها می پرسم.

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۹/۱۸ ۱۴:۵۷

سلام
سرعت و قابلیت اجرایی Over Clause به مراتب از Group by بهتر است. بطوریکه اگر یک عملیات یکسان را،بطور جداگانه،هم با Over Clause و هم با Group By انجام دهید. و در Execution Plan مشاهده نمایید، تفاوت را حس خواهید نمود. سایت زیر یک مثال ساده در این رابطه قرار داده است: Windowing functions - Underappreciated

در مورد اینکه برای محیط های Transactional مناسب است یا نه، عوامل زیادی در آن دخیل است و بسته به حجم داده ای مورد انتظار شما در خروجی دارد،بطور مثال اگر بخواهید یک گزارش 400 صفحه ای ایجاد نمایید، بطور حتم در چنین محیط هایی هیچ Scriptی مناسب، نیست، اما بطور قاطع می‌توان گفت که Window Functionها از کارایی بسیار خوبی برخوردار هستند، و انجام عملیات‌های پیچیده محاسباتی را برای ما آسانتر نموده اند.

نویسنده: فاطمه
تاریخ: ۱۳۹۱/۱۰/۲۳ ۱۷:۲۲

بسیار عالی بود
با تشکر

نویسنده: zarei
تاریخ: ۱۳۹۲/۰۲/۲۱ ۲۲:۱۴

سلام
ممنون از آموزش مفیدتون . سوال من اینه که اگه بخواهیم رکوردهای تکراری حذف بشن باید چیکار کنیم ؟ مثلا من میخام مجموع مبلغ بدهکار برای یک کد کل و در یک سند را در یک ردیف و همین مورد برای مجموع مبلغ بستانکار را نیز در یک رکورد یا ردیف دیگر بدهد . در صورتی که اگر از توابع Over ()Sum استفاده کنیم به ازای هر کد کل درآن سند یک رکورد در خروجی داریم (چه بدهکار و چه بستانکار)

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۲/۰۲/۲۲ ۸:۱۹

سلام
اگر سئوال شما را درست متوجه شده باشم، فکر می‌کنم، می‌بایست از Pivot استفاده نمایید،مقاله زیر می‌تواند در درک Pivot به شما کمک نماید. Pivot

موفق باشید.

نویسنده: محمد شهریاری
تاریخ: ۱۳۹۳/۰۲/۲۱ ۹:۰

با سلام
آیا امکان استفاده از scaler functionها هم هست ؟

ممنون

نویسنده: فرهاد
تاریخ: ۱۳۹۳/۰۲/۲۲ ۱۵:۱۶

سلام
معمولا از توابع Aggregate Functions می‌توان در Window Function استفاده نمود: ^

عنوان: **گرفتن خروجی XML از جداول در SQL Server 2012**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۹/۱۹ ۰:۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, xml, SQL Server 2012, T-SQL

فرض کنید که می‌خواهیم خروجی از جدول خود را به صورت XML نمایش یا از طریق وب سرویس در برنامه مان استفاده نماییم. اولین راهی که به ذهنمان می‌رسد خودمان رشته xml را با حلقه ای ایجاد نماید یا استفاده از فضای نام System.Xml و کلاس‌های نوشته شده برای این کار . اما خود Sql Server امکانات ویژه ای برای کار با ساختار xml مهیا نموده که براحتی می‌توانید خروجی xml از داده هایتان ایجاد نمایید.

برای این کار از عبارت For XML در Select می‌توان استفاده نمود. برای مثال برای بدست آوردن ساختار ساده از For Xml Auto استفاده نمایید

```
SELECT BusinessEntityID, PersonType, Title, FirstName, MiddleName, LastName
FROM Person
WHERE BusinessEntityID = 10001
FOR XML AUTO
```

که خروجی بصورت **node attribute** زیر می‌باشد:

```
<Person.Person BusinessEntityID="10001" PersonType="IN" FirstName="Carolyn" LastName="Alonso" />
```

اما اگر بخواهیم خروجی به صورت node Elements باشد کافیست از پارامتر Elements استفاده نمایید

```
SELECT BusinessEntityID, PersonType, Title, FirstName, MiddleName, LastName
FROM Person
WHERE BusinessEntityID = 10001
FOR XML AUTO, ELEMENTS
```

خروجی بصورت زیر می‌باشد:

```
<Person.Person>
  <BusinessEntityID>10001</BusinessEntityID>
  <PersonType>IN</PersonType>
  <FirstName>Carolyn</FirstName>
  <LastName>Alonso</LastName>
</Person.Person>
```

اگر بخواهیم node attributes و node elements با هم ترکیب کنیم بصورت زیر عمل می‌کنیم:

```
SELECT BusinessEntityID AS '@ID', PersonType, Title,  FirstName, MiddleName, LastName
FROM Person
WHERE BusinessEntityID = 10001
FOR XML ELEMENTS
```

خروجی بصورت زیر است:

```
<Person ID="10001">
  <PersonType>IN</PersonType>
  <FirstName>Carolyn</FirstName>
  <LastName>Alonso</LastName>
</Person>
```

حال می‌خواهیم همه nodeها را یک node ریشه قرار دهیم برای این کار از پارامتر ROOT در کنار AUTO به صورت زیر استفاده نمایید:

```
SELECT *
FROM Person
WHERE BusinessEntityID = 15291
FOR XML AUTO , ROOT('Persons')
```

اما اگر بخواهیم نام جدول را با نام دلخواه خود تغییر دهیم از پارامتر PATH به جای AUTO به صورت زیر استفاده نمایید:

```
SELECT *
FROM Person
WHERE BusinessEntityID = 15291
FOR XML PATH('P') , ROOT('Persons')
```

نظرات خوانندگان

نویسنده: احسان میرسعیدی
تاریخ: ۱۳۹۱/۰۹/۱۹ ۰:۲۶

بسیار تکنیک کارامدی بود. واقعا متشکرم

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۹/۱۹ ۷:۲۹

سلام
مرسی از مطلب مفید تان

نویسنده: بهراد
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۱:۸

بسیار عالی بود ، خیلی برام مفید واقع شد
راه مشابه ای برای خروجی Json نیست؟
ممنون

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۱:۵۷

ممنون .هنوز به صورت native نه اما از تیم microsoft در این لینک خواسته شده که For Json را هم اضافه کند.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۱:۵۹

« استفاده از JSON در SQL Server »

نویسنده: مهدی ناظم السادات
تاریخ: ۱۳۹۲/۰۹/۱۷ ۱۸:۵۸

حالا دستور Insert نداریم که بشه یه فایل xml رو مثل فایل backup روی دیتابیس restore کنیم؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۹/۱۸ ۰:۵۹

- می‌تونی با کدنویسی اینکار رو انجام بدی:

```
var reportData = new DataSet();
reportData.ReadXml("yourfile.xml");
var connection = new SqlConnection("DB ConnectionString");
var sbc = new SqlBulkCopy(connection);
sbc.DestinationTableName = "yourXMLTable";
```

- یا می‌تونی از import و export خود SQL Server استفاده کنی.
- و یا از OPENXML میشه استفاده کرد:

```
INSERT Customers
SELECT *
FROM OPENXML ...
```

نویسنده: ناصرنیازی
تاریخ: ۱۳۹۲/۱۱/۲۹ ۲۱:۵۱

تشکر فراوان از این مطلب فوق العاده کاربردی. یه نکته کوچک به ذهنم رسید. سومین قطعه کدی که نوشتید در جدول من کار نکرد به نظرم اومد که شاید اینچنین بوده باشه :
کد شما

```
FOR XML ELEMENTS
```

کدی که من مد نظرم هست و در جدول من کار می‌کنه :

```
for xml auto, ELEMENTS
```

عنوان: آشنایی با Window Function ها در SQL Server بخش دوم
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۹/۱۹ ۹:۳۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SQL Server 2012, T-SQL

قبل از مطالعه این بخش لطفا آشنایی با Window Functionها در SQL Server بخش اول را مطالعه نمایید.

در بخش اول ،در مورد Syntax مربوط به Over Clause صحبت کردیم، و برای درک استفاده از Over Clause، مثالهایی را بررسی نمودیم، در این بخش نیز،به تفاوت Row Clause و Range Clause می پردازیم.

مثال: با ایجاد یک Script،عملیات جمع روی یک فیلد خاص، بوسیله Row Clause و Range Clause انجام می‌دهیم. تا تفاوت آنها را درک نماییم.

در ادامه Script زیر را اجرا نمایید:

```
DECLARE @Test TABLE
(
RowID INT IDENTITY,
FName VARCHAR(20),
Salary SMALLINT
);
INSERT INTO @Test (FName, Salary)
VALUES ('George', 800),
('Sam', 950),
('Diane', 1100),
('Nicholas', 1250),
('Samuel', 1250),
('Patricia', 1300),
('Brian', 3000),
('Thomas', 1600),
('Fran', 2450),
('Debbie', 2850),
('Mark', 2975),
('James', 3000),
('Cynthia', 3000),
('Christopher', 5000);

SELECT RowID,FName,Salary,
       SumByRows = SUM(Salary) OVER (ORDER BY Salary ROWS UNBOUNDED PRECEDING),
   SumByRange = SUM(Salary) OVER (ORDER BY Salary RANGE UNBOUNDED PRECEDING)
FROM @Test
ORDER BY RowID;
```

خروجی بصورت زیر خواهد بود:

	RowID	FName	Salary	SumByRows	SumByRange
1	1	George	800	800	800
2	2	Sam	950	1750	1750
3	3	Diane	1100	2850	2850
4	4	Nicholas	1250	4100	5350
5	5	Samuel	1250	5350	5350
6	6	Patricia	1300	6650	6650
7	7	Brian	1500	8150	8150
8	8	Thomas	1600	9750	9750
9	9	Fran	2450	12200	12200
10	10	Debbie	2850	15050	15050
11	11	Mark	2975	18025	18025
12	12	James	3000	21025	24025
13	13	Cynthia	3000	24025	24025
14	14	Christopher	5000	29025	29025

با مشاهده شکل بالا، به وضوح می‌توان تفاوت Row و Range را تشخیص داد. در Script بالا از UNBOUNDED PRECEDING استفاده کردیم ، و مفهوم قالب آن به شرح ذیل می‌باشد:

مقدار فیلد Salary سطر جاری = جمع مقادیر فیلد Salary همه سطرهای ماقبل،سطر جاری + مقدار فیلد Salary سطر جاری

Row Clause بصورت فیزیکی به سطرها می‌نگرد و قالب بیان شده در Script را،روی تمامی سطرها،نسبت به جایگاه آنها در جدول، به ترتیب اعمال می‌نماید.و در شکل نیز قابل مشاهده می باشد، یعنی به چیدمان سطرها در خروجی که بصورت فیزیکی نمایش داده شده است، توجه می کند، و حاصل جمع هر سطر برابر است با حاصل جمع سطرهای ماقبل + سطر جاری

اما Range Clause:به چیدمان فیزیکی سطرها توجه نمی‌کند، بلکه بصورت منطقی به مقدار فیلد Salary سطرها توجه می‌نماید، یعنی مقادیری که در یک محدوده(Range) قرار دارند، حاصل جمع آنها،یکی است.

مقدار فیلد Salary سطر چهار و پنج برابر است با 1250 بنابراین حاصل جمع آنها برابر هم می‌باشد. و بصورت زیر محاسبه می‌شود:

5350= 1250 + 1250 + 1100 + 950 + 800

روش بیان شده، در مورد سطرهای 12 و 13 نیز صادق است.

امیدوارم با مثالهایی که در بخش اول و بخش دوم بررسی نمودیم، روش استفاده از Over Clause را درک کرده باشیم.

Window Functionها را به چهار بخش تقسیم بندی شده اند، که به شرح ذیل می‌باشد:

1- Ranking functions (توابع رتبه بندی)، که بررسی نمودیم.

2- NEXT VALUE FOR ، که در بحث ایجاد Sequence آن را بررسی نمودیم.

3- Aggregate Functions (توابع جمعی)، اکثرا با اینگونه توابع آشنا هستیم.

4- Analytic Functions (توابع تحلیلی) که در بخش بعدی آن را بررسی می‌نماییم.

یکی از منابع بسیار مفید در مورد Window Function ها کتاب Microsoft SQL Server 2012 High-Performance T-SQL Using Window Functions ، می باشد،که بطور کامل به Window Functionها اختصاص دارد و تکنیک‌های بسیار مفیدی را بیان می‌کند. مطالعه آن به علاقمندان، پیشنهاد می‌گردد.

موفق باشید.

نظرات خوانندگان

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۲:۲۱

دوست عزیز ممنون...
من قسمت Range و Row رو که شما توضیح دادی درست متوجه نشدم سرچی که زدم متوجه شدم این قابلیت تقریبا شبیه قسمت WITH TIES تو Select هست .برای مثال اگه بخواهیم 3 شاگرد برتر کلاس رو کوئری بزنیم اگه تو کلاس 3 نفر معدل 18 داشته باشن(با توجه به اینکه یک معدل 20 و 19 داریم) 2 نفر از شاگردها که معدل 18 دارن تو این کوئری نمیان(TOP 3) و...
برداشت منم ازRange اینه که بواسط ی برابر بودن تاریخ‌ها این 2 مقدار به هم گره خوردن و هنگام محاسبه مقدار یکسانی را تولید میکنن.

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۴:۲۱

سلام
اگر سئوال شما رو درست متوجه شده باشم،با یک مثال مفهوم Range رو بررسی می‌کنیم:

```
CREATE TABLE #Transactions
(
AccountId INTEGER,
TranDate DATE,
TranAmt NUMERIC(8, 2)
);
INSERT INTO #Transactions
SELECT *
FROM ( VALUES ( 1, '2011-01-15', 50),( 1, '2011-01-17', 500),( 1, '2011-01-17', 500),
  ( 1, '2011-01-16', 500),( 1, '2011-01-24', 75),( 1, '2011-01-26', 125),
  ( 1, '2011-02-28', 500),( 2, '2011-01-01', 500),( 2, '2011-01-15', 50),
              ( 2, '2011-01-22', 25),( 2, '2011-01-23', 125),( 2, '2011-01-26', 200),
              ( 2, '2011-01-29', 250),( 3, '2011-01-01', 500),( 3, '2011-01-15', 50 ),
              ( 3, '2011-01-22', 5000),( 3, '2011-01-25', 550),( 3, '2011-01-27', 95 ),
              ( 3, '2011-01-30', 2500)
) dt (AccountId, TranDate, TranAmt);
```

روی جدول فوق دو نوع Script اجرا می‌کنیم، مثال اول، براساس AccountID جدول را گروه بندی می‌نماییم. سپس هر گروه را براساس تاریخ Sort می‌کنیم، و در هر گروه مقدار Sum آن را بدست می‌آوریم:

```
SELECT
    AccountId,
    TranDate,
  TranAmt,
    Sum(TranAmt) OVER(partition by Accountid ORDER BY TranDate RANGE UNBOUNDED PRECEDING) AS SumAmt
FROM  #Transactions
GO
```

خروجی :

	AccountId	TranDate	TranAmt	SumAmt
1	1	2011-01-15	50.00	50.00
2	1	2011-01-16	500.00	550.00
3	1	2011-01-17	500.00	1550.00
4	1	2011-01-17	500.00	1550.00
5	1	2011-01-24	75.00	1625.00
6	1	2011-01-26	125.00	1750.00
7	1	2011-02-28	500.00	2250.00

مطابق شکل Sort براساس TranDate است، که چهار مقدار 500 در سه بازه تاریخی دیده می‌شود، حال محاسبه جمع هر سطر بصورت زیر است:

سطر دوم با وجود اینکه مقدار آن 500 است و در بازه تاریخی 2011-01-16 قرار دارد: مقدار آن برابر است با 500 + 50=550

سطر سوم و چهارم که در بازه تاریخی 2011-01-17 می‌باشد(به عبارتی در یک محدوده می‌باشند): برابر است با :

500+500+500+50=1550

در اینجا چیزی حذف نشده، حاصل جمع سطر سوم و چهارم ، چون در یک محدوده (Range) می‌باشد، برابر است با حاصل جمع سطرهای ما قبل یعنی سطر اول و دوم (500+50=550) + حاصل جمع تمامی سطرهای آن محدوده(Range)، یعنی سطر سوم و چهارم (500+500=1000)

مثال دیگر، در این حالت Sort روی فیلد TranAMT انجام می‌شود، و جدول همچنان روی فیلد Accountid گروه بندی می‌شود، بنابراین خواهیم داشت:

```
SELECT
    AccountId,
    TranDate,
  TranAmt,
    Sum(TranAmt) OVER(partition by Accountid ORDER BY TranAmt RANGE UNBOUNDED PRECEDING) AS SumAmt
FROM  #Transactions
GO
```

خروجی :

	AccountId	TranDate	TranAmt	SumAmt
1	1	2011-01-15	50.00	50.00
2	1	2011-01-24	75.00	125.00
3	1	2011-01-26	125.00	250.00
4	1	2011-02-28	500.00	2250.00
5	1	2011-01-17	500.00	2250.00
6	1	2011-01-17	500.00	2250.00
7	1	2011-01-16	500.00	2250.00
8	2	2011-01-22	25.00	25.00
9	2	2011-01-15	50.00	75.00
10	2	2011-01-23	125.00	200.00

در شکل، مقدار جمع هیچ سطری حذف نشده است، و Top ی هم در کار نیست.
حال اگر مثال فوق را روی میانگین در نظر بگیرید، باز هم تمام مقادیر، در محاسبه میانگین تاثیر گذار میباشند.

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۴:۵۹

ممنون...
« سطرهای فیزیکی » و «سطرهای منطقی» که شما گفتید برا من گیج کننده بود من با مثال Top ی که زدم فرق رو متوجه شدم
گفتم عنوان کنم دیگران هم استفاده کنن.البته این پست شما خیلی بهتر موضوع رو توضیح داد.

نویسنده: محمد سلم ابادی
تاریخ: ۱۳۹۱/۱۰/۳۰ ۱۲:۱۱

سلام،
ممنون از مطالب مفیدتون.
آیا دو دستور زیر با هم یکسان هستند یا خیر؟
range unbounded preceding
range between unbounded preceding and current row

و کوئری اولتون باید مساله running total باشه، که به سادگی توسط Over Clause حل شده.
کوئری زیر روشی بوده که قبل از نسخه 2012 برای حل اینگونه مسائل مورد استفاده قرار میگرفته

```
SELECT AccountId,
       TranDate,
       TranAmt,
       D.sumAmt AS older_method
       Sum(TranAmt) OVER(partition by Accountid
                         ORDER BY TranDate
                         RANGE UNBOUNDED PRECEDING) AS SumAmt
FROM  #Transactions AS T1
CROSS APPLY (SELECT SUM(T2.TranAmt)
   FROM #Transactions AS T2
  WHERE T2.AccountId = T1.AccountId
    AND T2.TranDate <= T1.TranDate) AS D(SumAmt)
ORDER BY T1.AccountId, T1.TranDate;
```

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۱۰/۳۰ ۱۳:۲۵

سلام
در جواب سؤال شما باید بگویم هر دو دستور یکی میباشند و هر دو از اولین سطر تا سطر جاری را در نظر می‌گیرند.

نویسنده: zarei
تاریخ: ۱۳۹۲/۰۲/۲۲ ۱۹:۳

سلام
من یه کوئری توسط Over Sum() .. نوشتم که تو در تو هست که ترتیب جمع دستور بیرونی برام مهمه .

```
;WITH cteBed ([Counter], id_doc , [Year] ,id_Total , date_duc ,Number_Temp , number_fix , sumbed ,
sumbes , row_no ) AS (
SELECT [Counter], d.id_doc , d.[Year] ,r.id_Total , d.date_duc ,d.Number_Temp ,d.number_fix ,
SUM( r.Mablagh_bed) OVER(PARTITION BY d.[Year] ,r.id_Total , d.Number_Temp) AS sumbed ,
 sumbes= 0,
```

```
ROW_NUMBER() OVER (PARTITION BY d.[Year] ,r.id_Total , d.date_duc , d.Number_Temp , d.number_fix  ORDER
BY  d.date_duc )AS  row_no
FROM tbl_Records r JOIN tbl_Documents d ON d.id_doc = r.id_doc  ) ,

 cteBes ([Counter], id_doc , [Year] ,id_Total , date_duc ,Number_Temp , number_fix , sumbed , sumbes  ,
row_no) AS (
SELECT   [Counter], d.id_doc , d.[Year] ,r.id_Total , d.date_duc ,d.Number_Temp ,d.number_fix , sumbed
= 0 ,
SUM( r.Mablagh_bes ) OVER(PARTITION BY d.[Year] ,r.id_Total , d.Number_Temp ) AS sumbes,
ROW_NUMBER() OVER (PARTITION BY d.[Year] ,r.id_Total ,  d.date_duc ,d.Number_Temp , d.number_fix ORDER
BY  d.date_duc )AS row_no
FROM tbl_Records r JOIN tbl_Documents d ON d.id_doc = r.id_doc )

SELECT [Counter], id_doc , [Year] ,id_Total , date_duc ,Number_Temp , number_fix , sumbed , sumbes ,
amountBed ,amountBes
,SUM(amountBed)OVER(  ORDER BY [Year] ,id_Total , date_duc , number_Temp, number_Fix ROWS BETWEEN
UNBOUNDED PRECEDING AND CURRENT ROW ) AS bed
,SUM(amountBes)OVER(  ORDER BY [Year] ,id_Total , date_duc , number_Temp, number_Fix ROWS BETWEEN
UNBOUNDED PRECEDING AND CURRENT ROW ) AS bes
FROM (
SELECT [Counter], id_doc , [Year] ,id_Total , date_duc ,Number_Temp , number_fix , sumbed , sumbes ,
amountBed = CASE WHEN id_Total  LIKE '1%' OR  Id_Total LIKE '2%' OR  Id_Total LIKE '7%' OR  Id_Total
LIKE '8%' THEN (tt.sumbed-tt.sumbes) ELSE 0 END ,
amountBes=CASE WHEN Id_Total LIKE '3%' OR Id_Total LIKE '4%' OR Id_Total LIKE '5%' OR Id_Total LIKE
'6%' OR Id_Total LIKE '9%' THEN (tt.sumbes-tt.sumbed)ELSE 0 END ,
ROW_NUMBER() OVER (PARTITION BY [Year] ,id_Total , date_duc , Number_Temp , number_fix  ORDER BY
date_duc )AS  row_no
FROM (
SELECT * FROM cteBed cb WHERE cb.row_no = 1
UNION ALL
SELECT * FROM cteBes cb WHERE cb.row_no = 1
) AS tt ([Counter], id_doc , [Year] ,id_Total , date_duc ,Number_Temp , number_fix , sumbed ,
sumbes,row_no ) WHERE not(sumbed = 0 AND sumbes = 0)
) AS rr
```

اگه تو یه دستور Select از Row_Number استفاده کرده باشم ، اول خروجی رو بدست میاره بعد خروجی رو بر حسب نوع مرتب سازی مربوط به Row_Number مرتب میکنه ؟ و دیگه اینکه خروجی دستور اول که شامل Row_Number هست بعد از مرتب شدن به همون صورت به دست دستور دوم (یا همون Select بیرونی) میرسه یا باز باید روی اون نیز مرتب سازی انجام بدم ؟ اصلا جای ستونی که مربوط به Row_Number هست اول یا آخر فرق میکنه ؟

اینارو به این خاطر پرسیدم ، چون هر بار داده هام جواب متفاوتی میداد و نتونستم تشخیص بدم . ممنون

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۲/۰۲/۲۵ ۸:۱۴

سلام

جواب سوال اول: در Syntax تابع Row_Number عملیات order by اجباری است، بنابراین عملیات سورت در ابتدا انجام می‌شود و سپس Row_Number (اعداد ترتیبی) روی رکوردها اعمال می‌گردد.

در سایت مایکروسافت به خوبی اشاره شده است که هیچ تضمینی وجود ندارد، خروجی یک Query با استفاده از Row_number در هر بار اجرا، با اجرای قبلی یکی باشد مگر آنکه موارد زیر را رعایت کرده باشید:

1- مقادیر ستونی که برای قسمت Partition در نظر گرفته اید، منحصربفرد باشد.

2- مقادیری که برای قسمت Order by در نظر گرفته اید منحصربفرد باشد.

3- ترکیب مقادیر Partition و Order by نیز مقدار منحصربفردی را ایجاد نماید.

جواب سوال دوم: جای ستون Row_number در زمان نمایش اهمیتی ندارد.

پیشنهاد دوستانه:

1- تاجایی که امکان دارد از OR در Queryهای خود استفاده ننمایید، باعث افزایش زمان اجرای Query شما می‌شود و هزینه بالایی دارد.

2- از Like نیز در نوشتن Queryهای خود اجتناب کنید.

برای اطلاعات بیشتر در مورد Row_Number به آدرس زیر مراجعه نمایید: Row_Number()

موفق باشید.

نویسنده: zarei
تاریخ: ۱۳۹۲/۰۲/۲۷ ۱:۳۸

ممنون از راهنماییتون . ولی برای رسیدن به پاسخ راه دیگه ای به ذهنم نرسید (استفاده از OR و Like)
میشه خواهش کنم راه جایگزین رو بهم بگید ؟ اموزشی در خصوص بررسی مفهومی Planهای تولیدی SQL سراغ دارید ؟
دیگه اینکه از کجا میتونم به طور دقیق و مفهومی هزینه استفاده از دستورات SQL رو مثل OR و ... رو بخونم ؟
ممنون

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۲/۰۲/۲۷ ۲۲:۱۹

سلام
مقاله زیر به خوبی طرز استفاده از Execution Plan را آموزش می‌دهد. How to read SQL Server graphical query execution plans

دو کتاب زیر، جهت مطالعه و بهینه سازی در ایجاد Query مفید است: SQL Server 2012 T-SQL Recipes
SQL Server 2012 Query Performance Tuning

موفق باشید.

عنوان: **گرفتن خروجی JSON از جداول در SQL Server 2012**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۳:۱۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, JSON, SQL Server 2012, T-SQL

در مطلب قبلی با استفاده از دستور For XML خروجی xml تولید کردیم اما با همین دستور می‌توان تا حدودی خروجی Json نیز تولید نمود. البته به صورت native هنور در sql server این امکان وجود ندارد که با رای دادن به این لینک از تیم ماکروسافت بخواهید که این امکان را در نسخه بعدی اضافه کند.
برای این کار یک جدول موقت ایجاد کرده و چند رکورد در آن درج می‌کنیم:

```
declare @t table(id int, name nvarchar(max), active bit)
insert @t values (1, 'Group 1', 1), (2, 'Group 2', 0)
```

حال با استفاده از همان for xml و پارامتر type که نوع خروجی xml را خودمان می‌توانیم تعیین نماییم و پارمتر Path این کار را بصورت زیر انجام می‌دهیم:

```
select '[' + STUFF((
        select
            ',{"id":' + cast(id as varchar(max))
            + ',"name":"' + name + '"'
            + ',"active":' + cast(active as varchar(max))
            +'}'

        from @t t1
        for xml path(''), type
    ).value('.', 'varchar(max)'), 1, 1, '') + ']'
```

توجه کنید در این جا از پارامتر path بدون نام استفاده شده است و از تابع STUFF برای در یک رشته در رشته دیگر استفاده شده است. خروجی در زیر آورده شده است:

```
[{"id":1,"name":"Group 1","active":1},{"id":2,"name":"Group 2","active":0}]
```

حالت پیشرفته‌تر آن است که بتوانیم یک join را بصورت فرزندان آن در json نمایش دهیم قطعه کد زیر را مشاهده فرمایید:

```
declare @group table(id int, name nvarchar(max), active bit)
insert @group values (1, 'Group 1', 1), (2, 'Group 2', 0)

declare @member table(id int, groupid int,name nvarchar(max))
insert @member values (1, 1,'Ali'), (2, 1,'Mojtaba'),(3,2,'Hamid')

select '[' + STUFF((
        select
            ',{"id":' + cast(g.id as varchar(max))
            + ',"name":"' + g.name + '"'
+ ',"members": { "children": [' +
(select + STUFF((
select
 ',{"id":' + cast(m.id as varchar(max))
+ ',"name":"' + m.name + '"}'
from @member m
where m.groupid = g.id
for xml path(''), type
 ).value('.', 'varchar(max)'), 1, 1, '')

 + ']}'
            + ',"active":' + cast(g.active as varchar(max))
            +'}')

        from @group g
        for xml path(''), type
    ).value('.', 'varchar(max)'), 1, 1, '') + ']'
```

خروجی json بصورت زیر است:

```
[{"id":1,"name":"Group 1","members":
     { "children": [{"id":1,"name":"Ali"},{"id":2,"name":"Mojtaba"}]}
,"active":1},
{"id":2,"name":"Group 2","members":
      { "children": [{"id":3,"name":"Hamid"}]}
,"active":0}]
```

حالت‌های خاص و پیشرفته‌تر را با امکانات t-sql خودتان می‌توانید به همین شکل تولید نمایید.

عنوان: **آشنایی با Window Function ها در SQL Server بخش سوم**
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۹/۲۴ ۱۹:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, SQL Server 2012, T-SQL

آشنایی با Window Functionها در SQL Server بخش اول
آشنایی با Window Functionها در SQL Server بخش دوم

در این بخش به دو Function از Analytic Functionها (توابع تحلیلی)، یعنی Lead Function و LAG Function می پردازیم.
قبل از اینکه به توابع ذکرشده بپردازیم، باید عرض کنم، شرح عملکرد اینگونه توابع کمی مشکل می‌باشد، بنابراین با ذکر مثال و توضیح آنها،سعی می‌کنیم،قابلیت هریک را بررسی و درک نماییم.

Lead Function:

این فانکشن در SQL Server 2012 ارائه شده است، و امکان دسترسی، به Dataهای سطر بعدی نسبت به سطر جاری را در نتیجه یک پرس و جو (Query)، ارائه می‌دهد. بدون آنکه از Self-join استفاده نمایید،
Syntax تابع فوق بصورت زیر است:

```
LEAD ( scalar_expression [ ,offset ] , [ default ] )
    OVER ( [ partition_by_clause ] order_by_clause )
```

شرح Syntax:

Scalar_expression: در Scalar_expression، نام یک فیلد یا ستون درج می‌شود، و مقدار برگشتی فیلد مورد نظر، به مقدار تعیین شده offset نیز بستگی دارد. خروجی Scalar_expression فقط یک مقدار است.
offset: منظور از Offset در این Syntax همانند عملکرد Offset در Syntax مربوط به Over می‌باشد. یعنی هر عددی برای offset در نظر گرفته شود، بیانگر نقطه آغازین سطر بعدی یا قبلی نسبت به سطر جاری است. به بیان دیگر، عدد تعیین شده در Offset به Sql server می‌فهماند چه تعداد سطر را در محاسبه در نظر نگیرد.
Default: زمانی که برای Offset مقداری را تعیین می‌نمایید، SQL Server به تعداد تعیین شده در Offset، سطرها را در نظر نمی‌گیرد، بنابراین مقدار خروجی Scalar_expression بطور پیش فرض Null در نظر گرفته می‌شود، چنانچه بخواهید، مقداری غیر از Null درج نمایید، می‌توانید مقدار دلخواه را در قسمت Default وارد کنید.
OVER **(** [partition_by_clause] order_by_clause) : در بخش اول بطور کامل توضیح داده شده است.

برای درک بهتر Lead Function چند مثال را بررسی می‌نماییم:
ابتدا Script زیر را اجرا می‌نماییم، که شامل ایجاد یک جدول و درج 18 رکورد در آن:

```
Create Table TestLead_LAG
(SalesOrderID int not null,
 SalesOrderDetailID int not null ,
 OrderQty smallint not null);
 GO
Insert Into TestLead_LAG
       Values (43662,49,1),(43662,50,3),(43662,51,1),
          (43663,52,1),(43664,53,1),(43664,54,1),
  (43667,77,3),(43667,78,1),(43667,79,1),
  (43667,80,1),(43668,81,3),(43669,110,1),
  (43670,111,1),(43670,112,2),(43670,113,2),
  (43670,114,1),(43671,115,1),(43671,116,2)
```

مثال:قصد داریم در هر سطر مقدار بعدی فیلد SalesOrderDetailID در فیلد دیگری به نام LeadValue نمایش دهیم، بنابراین Script زیر را ایجاد می‌کنیم:

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
LEAD(SalesOrderDetailID) OVER (ORDER BY SalesOrderDetailID) LeadValue
FROM TestLead_LAG s
WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
```

```
ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
```

خروجی بصورت زیر خواهد بود:

	SalesOrderID	SalesOrderDetailID	OrderQty	LeadValue
1	43663	52	1	77
2	43667	77	3	78
3	43667	78	1	79
4	43667	79	1	80
5	43667	80	1	110
6	43669	110	1	111
7	43670	111	1	112
8	43670	112	2	113
9	43670	113	2	114
10	43670	114	1	NULL

مطابق شکل، براحتی واضح است، که در هر سطر مقدار بعدی فیلد SalesOrderDetailID در فیلد LeadValue درج و نمایش داده می‌شود. فقط در سطر 10، چون مقدار بعدی برای فیلد SalesOrderDetailID وجود ندارد، SQL Server مقدار فیلد LeadValue را، Null در نظر می‌گیرد.

در این مثال فقط از آرگومان Scalar_expression، استفاده کردیم، و Offset و Default را مقدار دهی ننمودیم، بنابراین SQL Server بطور پیش فرض هیچ سطری را حذف نمی‌کند و مقدار Default را Null در نظر می‌گیرد.

مثال دوم: قصد داریم در هر سطر مقدار دو سطر بعدی فیلد SalesOrderDetailID را در فیلد LeadValue نمایش دهیم، و در صورت وجود نداشتن مقدار فیلد SalesOrderDetailID، مقدار پیش فرض صفر ،در فیلد LeadValue قرار دهیم،بنابراین Script آن بصورت زیر خواهد شد:

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
LEAD(SalesOrderDetailID,2,0) OVER (ORDER BY SalesOrderDetailID) LeadValue
FROM TestLead_LAG s
WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
```

خروجی:

	SalesOrderID	SalesOrderDetailID	OrderQty	LeadValue
1	43663	52	1	78
2	43667	77	3	79
3	43667	78	1	80
4	43667	79	1	110
5	43667	80	1	111
6	43669	110	1	112
7	43670	111	1	113
8	43670	112	2	114
9	43670	113	2	0
10	43670	114	1	0

در صورت مسئله بیان کرده بودیم، در هر سطر،مقدار فیلد SalesOrderDetailID دو سطر بعدی، را نمایش دهیم، بنابراین مقداری که برای Offset در نظر می‌گیریم، برابر دو خواهد بود، سپس گفته بودیم، چنانچه در هر سطر مقدار فیلد SalesOrderDetailID وجود نداشت،بجای مقدار پیش فرض Null،از مقدار صفر استفاده شود، بنابراین به Default مقدار صفر را نسبت دادیم.

```
LEAD(SalesOrderDetailID,2,0)
```

در شکل، مطابق صورت مسئله، مقدار فیلد LeadValue سطر اول برابر است با 78،
به بیان ساده‌تر برای بدست آوردن مقدار فیلد LaedValue هر سطر، می‌بایست هر سطر را به علاوه 2 (Offset) نماییم، تا سطر بعدی بدست آید، سپس مقدار SalesOrderDetailID را در فیلد LeadValue قرار می‌دهیم.
به سطر 9 و 10 توجه نمایید، که مقدار فیلد LeadValue آنها برابر با صفر است، واضح است، سطر 10 + 2 برابر است با 12(12=2+10)، چنین سطری در خروجی نداریم، بنابراین بطور پیش فرض مقدار LeadVaule توسط Sql Server برابر Null در نظر گرفته می‌شود، اما نمی‌خواستیم، که این مقدار Null باشد، بنابراین به آرگومان Default مقدار صفر را نسبت دادیم، تا SQL Server ، به جای استفاده از Null، مقدار در نظر گرفته شده صفر را استفاده نماید.
اگر چنین فانکشنی وجود نداشت، برای شبیه سازی آن می‌بایست از Join روی خود جدول استفاده می‌نمودیم، و یکسری محاسبات دیگر، که کار را سخت می‌نمود، مثال دوم را با Script زیر می‌توان شبیه سازی نمود:

```
WITH cteLead
AS
(
SELECT SalesOrderID,SalesOrderDetailID,OrderQty,
       ROW_NUMBER() OVER (ORDER BY SalesOrderDetailID) AS sn
FROM TestLead_LAG
WHERE
SalesOrderID IN (43670, 43669, 43667, 43663)
)
SELECT m.SalesOrderID, m.SalesOrderDetailID, m.OrderQty,
       case  when sLead.SalesOrderDetailID is null Then 0 Else sLead.SalesOrderDetailID END as
leadvalue
FROM cteLead AS m
LEFT OUTER JOIN cteLead AS sLead ON sLead.sn = m.sn+2
ORDER BY m.SalesOrderID, m.SalesOrderDetailID, m.OrderQty
```

جدول موقتی ایجاد نمودیم، که ROW_Number را در آن اضافه کردیم، سپس جدول ایجاد شده را با خود Join کردیم، و گفتیم، که مقدار فیلدLeadValue هر سطر برابر است با مقدار فیلد SalesOrderDetailID دو سطر بعد از آن. و با Case نیز مقدار پیش فرض را صفر در نظر گرفتیم.

LAG Function:

این فانکشن نیز در SQL Server 2012 ارائه شده است، و امکان دسترسی، به Dataهای سطر قبلی نسبت به سطر جاری را در نتیجه یک پرس و جو (Query)، ارائه می‌دهد. بدون آنکه از Self-join استفاده نمایید،

Syntax آن شبیه به فانکشن Lead میباشد و بصورت زیر است:

```
LAG (scalar_expression [,offset] [,default])
    OVER ( [ partition_by_clause ] order_by_clause )
```

Syntax مربوط به فانکشن LAG را شرح نمی‌دهم، بدلیل آنکه شبیه به فانکشن Lead می‌باشد، فقط تفاوت آن در Offset است، Offset در فانکشن LAG روی سطرهای ماقبل سطر جاری اعمال می‌گردد.

مثال دوم را برای حالت LAG Function شبیه سازی می‌نماییم:

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
LAG(SalesOrderDetailID,2,0) OVER (ORDER BY SalesOrderDetailID) LAGValue
FROM TestLead_LAG s
WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
go
```

خروجی :

	SalesOrderID	SalesOrderDetailID	OrderQty	LAGValue
1	43663	52	1	0
2	43667	77	3	0
3	43667	78	1	52
4	43667	79	1	77
5	43667	80	1	78
6	43669	110	1	79
7	43670	111	1	80
8	43670	112	2	110
9	43670	113	2	111
10	43670	114	1	112

همانطور که گفتیم، LAG Function عکس LEAD Function میباشد. یعنی مقدار فیلد LAGValue سطر جاری برابر است با مقدار SalesOrderDetailID دو سطر ما قبل خود.

مقدار فیلد LAGValue دو سطر اول و دوم نیز برابر صفر است، چون دو سطر ماقبل آنها وجود ندارد، و مقدار صفر نیز بدلیل این است که Default را برابر صفر در نظر گرفته بودیم.

مثال: در این مثال از Laed Function و LAG Function بطور همزمان استفاده می‌کنیم، با این تفاوت، که از گروه بندی نیز استفاده شده است:

Script زیر را اجرا نمایید:

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
       Lead(SalesOrderDetailID) OVER (PARTITION BY SalesOrderID ORDER BY SalesOrderDetailID) LeadValue,
       LAG(SalesOrderDetailID) OVER (PARTITION BY SalesOrderID ORDER BY SalesOrderDetailID) LAGValue
FROM TestLead_LAG s
WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
go
```

خروجی:

	SalesOrderID	SalesOrderDetailID	OrderQty	LeadValue	LAGValue
1	43663	52	1	NULL	NULL
2	43667	77	3	78	NULL
3	43667	78	1	79	77
4	43667	79	1	80	78
5	43667	80	1	NULL	79
6	43669	110	1	NULL	NULL
7	43670	111	1	112	NULL
8	43670	112	2	113	111
9	43670	113	2	114	112
10	43670	114	1	NULL	113

با بررسی هایی که در مثالهای قبل نمودیم،خروجی زیر را می‌توان براحتی تشخیص داد، و توضیح بیشتری نمی‌دهم.
موفق باشید.

نظرات خوانندگان

نویسنده: محمد
تاریخ: ۱۳۹۱/۱۰/۲۸ ۱۸:۵۳

سلام،
این توابع واقعا کار رو آسون کردن، ما رو از بکارگیری چندین بار self join بی نیاز کردن.
بطور نمونه اگه بخواهیم مقدار SalesOrderDetailID سطر قبلی، دو سطرقبلی، سطربعدی و دو سطر بعدی را بدست بیاریم در نسخه 2008 ساده‌ترین و مناسب‌ترین کوئری این هست:

```
WITH cteLead
AS
(
SELECT SalesOrderID,SalesOrderDetailID,OrderQty,
       ROW_NUMBER() OVER (PARTITION BY SalesOrderID
                          ORDER BY SalesOrderDetailID) AS sn
FROM TestLead_LAG
WHERE
SalesOrderID IN (43670, 43669, 43667, 43663)
)
SELECT m.SalesOrderID, m.SalesOrderDetailID, m.OrderQty,
       COALESCE(sLead1.SalesOrderDetailID, 0) as leadvalue1,
       COALESCE(sLead2.SalesOrderDetailID, 0) as leadvalue2,
       COALESCE(sLag1.SalesOrderDetailID, 0) as lagvalue2,
       COALESCE(sLag2.SalesOrderDetailID, 0) as lagvalue2
FROM cteLead AS m
LEFT OUTER JOIN cteLead AS sLead1
ON m.sn = sLead1.sn - 1
AND m.SalesOrderID = sLead1.SalesOrderID
LEFT OUTER JOIN cteLead AS sLead2
ON m.sn = sLead2.sn - 2
AND m.SalesOrderID = sLead2.SalesOrderID
LEFT OUTER JOIN cteLead AS sLag1
ON m.sn = sLag1.sn + 1
AND m.SalesOrderID = sLag1.SalesOrderID
LEFT OUTER JOIN cteLead AS sLag2
ON m.sn = sLag2.sn + 2
AND m.SalesOrderID = sLag2.SalesOrderID
ORDER BY m.SalesOrderID, m.SalesOrderDetailID, m.OrderQty;
```

در حالی که با دو تابعی که شما در اینجا پوشش دادین میشه کوئری فوق را فوق العاده ساده‌تر نمود:

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
       Lead(SalesOrderDetailID, 1, 0) OVER (PARTITION BY SalesOrderID ORDER BY SalesOrderDetailID)
LeadValue1,
       LAG(SalesOrderDetailID, 1, 0) OVER (PARTITION BY SalesOrderID ORDER BY SalesOrderDetailID)
LAGValue1,
       Lead(SalesOrderDetailID, 2, 0) OVER (PARTITION BY SalesOrderID ORDER BY SalesOrderDetailID)
LeadValue2,
       LAG(SalesOrderDetailID, 2, 0) OVER (PARTITION BY SalesOrderID ORDER BY SalesOrderDetailID)
LAGValue2,
FROM TestLead_LAG s
WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
```

نویسنده: حسین
تاریخ: ۱۳۹۱/۱۰/۳۰ ۱۴:۵۸

جالب بود مرسی

عنوان: **آشنایی با Window Function ها در SQL Server بخش چهارم**
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۱۰/۰۳ ۲۳:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, SQL Server 2012, T-SQL

برای مطالعه این بخش لازم است، به Syntax مربوط به Over آشنا باشیم، در بخش اول بطور کامل به Syntax مربوط به Over پرداختیم.

در این بخش دو فانکشن دیگر از توابع تحلیلی (Analytic functions) به نامهای First_Value و Last_Value را بررسی می‌نماییم.

First_Value

این فانکشن نیز همانند دیگر فانکشنهای تحلیلی در نسخه SQL Server 2012 ارائه گردیده است. و اولین مقدار از یک مجموعه مقادیر را بر می‌گرداند. و Syntax آن بصورت ذیل می‌باشد:

```
FIRST_VALUE ( [scalar_expression )
    OVER ( [ partition_by_clause ] order_by_clause [ rows_range_clause ] )
```

شرح Syntax:

1- Scalar_expression : مقدار آن می‌تواند نام یک فیلد یا Subquery باشد.

2- Over : در بخش اول بطور مفصل آن را بررسی نمودیم.

قبل از بررسی تابع First_Value،ابتدا Script زیر را اجرا نمایید، که شامل یک جدول و درج چند رکورد در آن است.

```
Create Table Test_First_Last_Value
(SalesOrderID int not null,
 SalesOrderDetailID int not null ,
 OrderQty smallint not null);
 GO
Insert Into Test_First_Last_Value
       Values (43662,49,1),(43662,50,3),(43662,51,1),
          (43663,52,1),(43664,53,1),(43664,54,1),
  (43667,77,3),(43667,78,1),(43667,79,1),
          (43667,80,1),(43668,81,3),(43669,110,1),
          (43670,111,1),(43670,112,2),(43670,113,2),
          (43670,114,1),(43671,115,1),(43671,116,2)
```

مثال: ابتدا Scriptی ایجاد می‌نماییم،بطوریکه جدول Test_Firts_Last_Value را براساس فیلد SalesOrderID گروه بندی نموده و اولین مقدار فیلد SalesOrderDetailID در هرگروه را مشخص نماید.

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
       FIRST_VALUE(SalesOrderDetailID) OVER (PARTITION BY SalesOrderID
       ORDER BY SalesOrderDetailID) FstValue
FROM Test_First_Last_Value s
     WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
     ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
```

خروجی:

	SalesOrderID	SalesOrderDetailID	OrderQty	FstValue
1	43663	52	1	52
2	43667	77	3	77
3	43667	78	1	77
4	43667	79	1	77
5	43667	80	1	77
6	43669	110	1	110
7	43670	111	1	111
8	43670	112	2	111
9	43670	113	2	111
10	43670	114	1	111

مطابق Script چهار گروه در خروجی ایجاد شده است و در فیلد FstValue ، اولین مقدار هر گروه نمایش داده می‌شود. اگر بخش‌های قبلی Window Functionها را مطالعه کرده باشید، تحلیل این تابع کار بسیار ساده ای است.

Last_Value

این تابع نیز در نسخه SQL Server 2012 ارائه گردیده است. و آخرین مقدار از یک مجموعه مقادیر را بر می‌گرداند، به عبارتی فانکشن Last_Value عکس فانکشن First_Value عمل می‌نماید و Syntax آن به شرح ذیل میباشد:

```
LAST_VALUE ( [scalar_expression ]
    OVER ( [ partition_by_clause ] order_by_clause rows_range_clause )
```

شرح Syntax تابع Last_Value شبیه به تابع First_Value می‌باشد.

مثال: همانند مثال قبل Scriptی ایجاد می‌نماییم،بطوریکه جدول Test_Firts_Last_Value را براساس فیلد SalesOrderID گروه بندی نموده و آخرین مقدار فیلد SalesOrderDetailID در هرگروه را مشخص نماید.

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
       LAST_VALUE(SalesOrderDetailID) OVER (PARTITION BY SalesOrderID
       ORDER BY SalesOrderDetailID RANGE BETWEEN UNBOUNDED PRECEDING  AND UNBOUNDED FOLLOWING)
LstValue
FROM Test_First_Last_Value s
     WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
     ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
```

خروجی:

	SalesOrderID	SalesOrderDetailID	OrderQty	LstValue
1	43663	52	1	52
2	43667	77	3	80
3	43667	78	1	80
4	43667	79	1	80
5	43667	80	1	80
6	43669	110	1	110
7	43670	111	1	114
8	43670	112	2	114
9	43670	113	2	114
10	43670	114	1	114

خروجی جدول،به چهار گروه تقسیم،و آخرین مقدار هر گروه،در فیلد LstValue نمایش داده شده است. در این مثال نیز تحلیلی نخواهیم داشت، چون فرض بر آن است که بخش‌های قبلی را مطالعه نموده ایم.

موفق باشید.

نظرات خوانندگان

نویسنده: محمد
تاریخ: ۱۳۹۱/۱۰/۲۸ ۱۰:۳۹

سلام،
مطلب اول: قسمت order by در ماده over در هر دو کوئری به چه جهت آمده است؟
مطلب دوم: first_value چه مزیتی نسبت به min() over داره، منظورم اینه که میشه خروجی کوئری اولتون رو با این کوئری بدست آورد:

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
       MIN(SalesOrderDetailID) OVER (PARTITION BY SalesOrderID) FstValue
FROM Test_First_Last_Value s
WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
```

نویسنده: محمد
تاریخ: ۱۳۹۱/۱۰/۲۸ ۱۲:۱۱

سلام،
من SQL Server 2012 ندارم، ولی تا اونجایی که متوجه شدم بر اساس شواهد دو کوئری زیر باید یک نتیجه رو برگردانند. منظورم اینکه که با first_value میشه last_value هم شبیه سازی کرد، فقط کافیه که در ماده order by از کلید واژه DESC استفاده بشه.
اگه من اشتباه میکنم لطفا راهنمایی بفرمایید.

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
       LAST_VALUE(SalesOrderDetailID) OVER (PARTITION BY SalesOrderID
       ORDER BY SalesOrderDetailID)  LstValue
FROM Test_First_Last_Value s
     WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
     ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty

SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
       FITST_VALUE(SalesOrderDetailID) OVER (PARTITION BY SalesOrderID
       ORDER BY SalesOrderDetailID DESC) FstValue
FROM Test_First_Last_Value s
     WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
     ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
```

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۱۰/۲۹ ۱۱:۵۳

سلام
جواب سئوال اول: در Syntax تابع First_value استفاده از Order by اجباری می‌باشد.
جواب سئوال دوم:
First_Value اولین مقدار یا اولین Row در یک گروه را مشخص می‌کند و به مفهوم کوچکترین مقدار نمی‌باشد، شاید، مثالی که در مقاله زدم شما را به اشتباه انداخت، در زیر با یک مثال، First_value و Min را مقایسه می‌کنیم.
ابتدا یک جدول و چند رکورد، در آن درج می‌کنیم:

```
CREATE TABLE Employees (
    EmployeeId INT IDENTITY PRIMARY KEY,
    Name VARCHAR(50),
    HireDate DATE NOT NULL,
    Salary INT NOT NULL
    )
GO
```

```
INSERT INTO Employees (Name, HireDate, Salary)
VALUES
    ('Alice', '2011-01-01', 20000),
    ('Brent', '2011-01-15', 19000),
    ('Carlos', '2011-02-01', 22000),
    ('Donna', '2011-03-01', 25000),
    ('Evan', '2011-04-01', 18500)
GO
```

در ادامه Script زیر را اجرا می‌کنیم:

```
Select EmployeeId,Name,Salary,HireDate,
   First_VALUE(HireDate) OVER(ORDER BY Salary RANGE BETWEEN UNBOUNDED PRECEDING
                              AND UNBOUNDED FOLLOWING) AS First,
   Min(HireDate) OVER(ORDER BY Salary
                      RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS Min
FROM Employees
ORDER BY EmployeeId
GO
```

خروجی بصورت زیر می‌شود:

	EmployeeId	Name	Salary	HireDate	First	Min
1	1	Alice	20000	2011-01-01	2011-04-01	2011-01-01
2	2	Brent	19000	2011-01-15	2011-04-01	2011-01-01
3	3	Carlos	22000	2011-02-01	2011-04-01	2011-01-01
4	4	Donna	25000	2011-03-01	2011-04-01	2011-01-01
5	5	Evan	18500	2011-04-01	2011-04-01	2011-01-01

در شکل بالا تفاوت Min و First_Value بطور کامل مشخص است، اگر به Query دقت نمایید، Sort براساس Salary انجام شده است، برای حالت First_value ،مقدار فیلد HireDate در اولین رکورد، برابر است با 2011-04-01 ، بنابراین سورت روی نمایش First_value تاثیر گذار است، بطوریکه Sort برای حالت Min، تاثیر گذار نمی‌باشد، و تابع Min ، کوچکترین مقدار، از مقادیر ستون HireDate را بدست می‌آورد، به بیان ساده‌تر در حالت استفاده از Min، عملیات Sort بیهوده می‌باشد. چون تابع MIN روی کل مقادیر یک گروه یا ستون تاثیر می‌گذارد.

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۱۰/۲۹ ۱۲:۵۱

سلام

شما می‌توانید، با دستکاری Queryها خروجی‌های یکسانی را ایجاد نمایید، دو Query که ایجاد نمودید، خروجی یکسانی ندارند. Query دوم شما با خروجی Last_Value،مقاله یکسان است، اما باید بگویم که مفهوم Last_Value این است که آخرین سطر در یک گروه را بر می‌گرداند. بهتر است بخش اول را مطالعه نمایید.

علت یکسان نبودن نتیجه دو Query شما در نحوه Sort و مفهوم First_value و Last_Value می‌باشد:

نکته: اگر در Over Clause شرط Order by را اعمال نماییم، اما از Row یا Range استفاده نکنیم، SQL Server بصورت پیش فرض از قالب زیر استفاده می‌نماید:

```
RANGE UNBOUNDED PRECEDING AND CURRENT ROW
```

نویسنده: محمد
تاریخ: ۱۳۹۱/۱۰/۲۹ ۱۳:۲۳

ممنون از پاسختون، الان متوجه تفاوتشون شدم.

ستون older_method باید مقادیرش دقیقا مشابه با first_value شما باشد:

```
Select EmployeeId,Name,Salary,HireDate,
       First_VALUE(HireDate) OVER(ORDER BY Salary RANGE BETWEEN UNBOUNDED PRECEDING
                                  AND UNBOUNDED FOLLOWING) AS First,
   D.HireDate AS older_method
FROM Employees
CROSS APPLY (SELECT TOP 1 HireDate
   FROM Employees
  --WHERE E1.EmployeeId = E2.EmployeeId
  ORDER BY Salary ASC) AS D
ORDER BY EmployeeId;
```

نویسنده: محمد
تاریخ: ۱۳۹۱/۱۰/۲۹ ۱۳:۳۱

ممنون از شما، من مطالب بخش اول رو مطالعه کردم.

عبارت RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING به معنای تمام سطرهای جدول هست دیگه درسته.

یعنی تمام سطرهای جدول از اولین گرفته، جاری گرفته و آخرین رو پوشش میده.

با این توضیحات باید دو کوئری زیر اینبار جواب یکسانی بدهند:

```
SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
       FIRST_VALUE(SalesOrderDetailID) OVER (PARTITION BY SalesOrderID
       ORDER BY SalesOrderDetailID RANGE BETWEEN UNBOUNDED PRECEDING  AND UNBOUNDED FOLLOWING)
LstValue
FROM Test_First_Last_Value s
     WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
     ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty

SELECT s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty,
       LAST_VALUE(SalesOrderDetailID) OVER (PARTITION BY SalesOrderID
       ORDER BY SalesOrderDetailID DESC RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING)
LstValue
FROM Test_First_Last_Value s
     WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
     ORDER BY s.SalesOrderID,s.SalesOrderDetailID,s.OrderQty
```

دو کوئری کاملا یکسان هستند به غیر از اینکه در کوئری دوم یک DESC اضافه شده و نام تابع از first به last تغییر کرده است.

عنوان: **استفاده از SQLDom برای آنالیز عبارات T-SQL**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۰۶ ۲۲:۱۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, T-SQL

به همراه بسته Features pack اس کیوال سرور 2012، دو بسته SqlDom.msi نیز وجود دارند (نسخه‌های X86 و X64). این بسته حاوی اسمبلی Microsoft.SqlServer.TransactSql.ScriptDom.dll می‌باشد که نهایتا در آدرس Program Files\Microsoft SQL Server\110\SDK\Assemblies کپی خواهد شد.

به کمک آن می‌توان عبارات پیچیده T-SQL را Parse و آنالیز کرد. البته باید در نظر داشت هرچند این بسته جهت SQL Server 2012 ارائه شده اما این اسمبلی با نگارش‌های 2005 به بعد اس کیوال سرور کاملا سازگار است و اساسا نیازی هم به SQL Server ندارد. در ادامه مروری خواهیم داشت بر نحوه استفاده از آن.

یافتن کوئری‌های Select * در بین انبوهی از اسکریپت‌ها به کمک SQLDom

در مورد مضرات کوئری‌های select * پیشتر مطلبی را در این سایت خوانده‌اید. در ادامه قصد داریم به کمک امکانات اسمبلی Microsoft.SqlServer.TransactSql.ScriptDom.dll، تعدادی عبارت T-SQL را آنالیز کرده و مشخص کنیم که آیا حاوی select * هستند یا خیر. کد کامل آن‌را در ذیل مشاهده می‌کنید:

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;
using Microsoft.SqlServer.TransactSql.ScriptDom;

namespace DbCop
{
    // Microsoft® SQL Server® 2012 Transact-SQL ScriptDom
    // SQL Server 2012 managed parser, Supports SQL Server 2005+
    // SQLDom.msi (redist x86/x64)
    // http://www.microsoft.com/en-us/download/details.aspx?id=29065
    // X86: http://go.microsoft.com/fwlink/?LinkID=239634&clcid=0x409
    // X64: http://go.microsoft.com/fwlink/?LinkID=239635&clcid=0x409
    // Program Files\Microsoft SQL
Server\110\SDK\Assemblies\Microsoft.SqlServer.TransactSql.ScriptDom.dll

    class Program
    {
        static void Main()
        {
            const string tSql = @"
                -- select * in PROCEDURE
                CREATE PROCEDURE dbo.SelectStarTest
                AS
                SELECT * FROM dbo.tbl1
                go

                -- select * in PROCEDURE with TableVar
                Create PRocedure SelectAll
                AS
                Declare @X table(Id integer)
                Select * from @x
                go

                -- select * in PROCEDURE with ctex
                CREATE PROCEDURE dbo.SelectAllCte
                AS
                WITH ctex
                AS (
                SELECT * FROM sys.objects
                )
                SELECT * FROM ctex
                go

                -- normal select *
                select * from tbl1;
                select * from dbo.tbl2;
            ";
```

```
            IList<ParseError> errors;
            TSqlScript sqlFragment;
            using (var reader = new StringReader(tSql))
            {
                var parser = new TSql110Parser(initialQuotedIdentifiers: true);
                sqlFragment = (TSqlScript)parser.Parse(reader, out errors);
            }

            if (errors != null && errors.Any())
            {
                var sb = new StringBuilder();
                foreach (var error in errors)
                    sb.AppendLine(error.Message);

                throw new InvalidOperationException(sb.ToString());
            }

            var i = 0;
            foreach (var batch in sqlFragment.Batches)
            {
                Console.WriteLine("Batch: {0}, Statement(s): {1}", ++i, batch.Statements.Count);
                foreach (var statement in batch.Statements)
                {
                    processStatement(statement);
                }
                Console.WriteLine();
            }

            Console.WriteLine("\nPress a key...");
            Console.Read();
        }

        private static void processStatement(TSqlStatement statement)
        {
            var createProcedureStatement = statement as CreateProcedureStatement;
            if (createProcedureStatement != null)
            {
                var statementList = createProcedureStatement.StatementList;
                foreach (var procedureStatement in statementList.Statements)
                {
                    processStatement(procedureStatement);
                }
            }

            var selectStatement = statement as SelectStatement;
            if (selectStatement != null)
            {
                var query = selectStatement.QueryExpression;
                var selectElements = ((QuerySpecification)query).SelectElements;
                foreach (var selectElement in selectElements)
                {
                    var expression = selectElement as SelectStarExpression;
                    if (expression == null) continue;
                    Console.WriteLine(
                        "`Select *` detected @StartOffset:{0}, Line:{1}, T-SQL: {2}",
                        expression.StartOffset,
                        expression.StartLine,
                        statementToString(selectStatement));
                }
            }
        }

        private static string statementToString(TSqlFragment selectStatement)
        {
            var text = new StringBuilder();
            for (var i = selectStatement.FirstTokenIndex; i <= selectStatement.LastTokenIndex; i++)
            {
                text.Append(selectStatement.ScriptTokenStream[i].Text);
            }
            return text.ToString();
        }
    }
}
```

توضیحات:

پس از نصب SQLDom.msi، ارجاعی را به اسمبلی زیر اضافه نمائید تا بتوانید کد فوق را کامپایل کنید:

Program Files\Microsoft SQL Server\110\SDK\Assemblies\Microsoft.SqlServer.TransactSql.ScriptDom.dll

کار با ایجاد وهله‌ای از TSql110Parser شروع می‌شود. متد Parse آن، آرگومانی از نوع TextReader را قبول می‌کند. برای مثال با استفاده از StringReader می‌توان محتوای یک متغیر رشته‌ای را به آن ارسال کرد و یا توسط StreamReader یک فایل sql را.

پس از فراخوانی متد Parse، بهتر است بررسی شود که آیا عبارت T-SQL دریافتی معتبر بوده است یا خیر. اینکار را توسط لیستی از ParseErrorهای دریافتی می‌توان انجام داد.

خروجی متد Parse، حاوی یک سری Batch آنالیز شده است. هر عبارت Go در اینجا یک Batch را تشکیل می‌دهد. سپس در داخل هر batch به دنبال batch.Statements خواهیم گشت تا بتوان به عبارات T-SQL آن‌ها دسترسی یافت.

در ادامه کار اصلی توسط متد processStatement صورت می‌گیرد. عبارات دریافتی، در حالت کلی از نوع TSqlStatement هستند اما در اصل می‌توانند یکی از مشتقات آن نیز باشند. در اینجا فقط دو مورد CreateProcedureStatement و SelectStatement بررسی شده‌اند (مطابق رشته tSql ابتدای مثال). هر دو عبارت، از کلاس TSqlStatement مشتق شده‌اند.

در متد processStatement عبارات select معمولی و همچنین آن‌هایی که داخل رویه‌های ذخیره شده تعریف شده‌اند، استخراج شده و در نهایت بررسی می‌شوند که آیا از نوع SelectStarExpression هستند یا خیر (همان select * صورت مساله).

خروجی مثال فوق به شرح زیر است:

```
Batch: 1, Statement(s): 1
`Select *` detected @StartOffset:140, Line:5, T-SQL: SELECT * FROM dbo.tbl1

Batch: 2, Statement(s): 1
`Select *` detected @StartOffset:368, Line:12, T-SQL: Select * from @x

Batch: 3, Statement(s): 1
`Select *` detected @StartOffset:659, Line:22, T-SQL: WITH ctex
               AS (
               SELECT * FROM sys.objects
               )
               SELECT * FROM ctex

Batch: 4, Statement(s): 2
`Select *` detected @StartOffset:753, Line:26, T-SQL: select * from tbl1;
`Select *` detected @StartOffset:791, Line:27, T-SQL: select * from dbo.tbl2;
```

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۳۰ ۱۳:۱۸

یک نکته‌ی تکمیلی

بسته‌ی ScriptDom را از نیوگت نیز می‌توانید دریافت کنید:

```
PM> Install-Package Microsoft.SqlServer.TransactSql.ScriptDom
```

عنوان: **دسته بندی سطر های جدول**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۰/۳۰ ۱۰:۵
آدرس: www.dotnettips.info
برچسب‌ها: T-SQL, SQL Server

فرض کنید میخواهیم سطرهای جدول را 6 تا 6 تا سوا کنیم و به هر کدام یک عددی انتساب دهیم و هر قسم تولید شده را نیز 2 تا 2 تا سوا کنیم و بهش عدد انتساب دهیم.
به تصویر زیر توجه بفرمایید. ابتدا داده‌ها به دو دسته ششتایی تقسیم شدن(ستون ntl)، سپس هر کدام از این دسته‌ها نیز به سه دسته دوتایی تقسیم شدن(ستون grp) هدف ما تولید دو ستون ntl و grp توسط query می‌باشد.

	nbr	ntl	grp
1	1	1	1
2	2	1	1
3	3	1	2
4	4	1	2
5	5	1	3
6	6	1	3
7	7	2	1
8	8	2	1
9	9	2	2
10	10	2	2
11	11	2	3
12	12	2	3

برای بدست آوردن مقادیر دو ستون مذکور روش‌های متنوعی وجود دارد که برخی از آنها را در انیجا پوشش میدم.
قبل از هر چیزی ابتدا جدول را ایجاد و 12 سطر زیر را در آن انتشار دهید:

```
CREATE TABLE T (nbr INT NOT NULL);
INSERT T VALUES (1), (2), (3), (4), (5), (6),
                (7), (8), (9), (10), (11), (12);
```

روش اول:
این روش، تعمیم پذیری و پویایی ندارد و برای هر سناریویی مناسب نخواهد بود. ولی از آنجایی که دیدم کوئری زیر میتواند یک نمونه از کاربرد Ntile باشه آن را مطرح کردم.

```
SELECT nbr, ntl, NTILE(3) OVER(PARTITION BY ntl ORDER BY nbr) AS grp
FROM (
         SELECT nbr, NTILE(2) OVER(ORDER BY nbr) ntl
         FROM T
     ) AS D;
```

تابع ntile داخلی سطرهای جدول را به دو قسم تقسیم می‌کند و برای قسم اول عدد 1 و برای قسم دوم عدد 2 را در نظر می‌گیرد.
تابع ntile بیرونی بر اساس دو عدد 1و 2 گروه بندی انجام داده و هر گروه را به 3 قسمت تقسیم می‌کند. قسمت اول 1، دوم 2 و

سوم 3 خواهد بود.
لازم به ذکر است که باید خارج قسمت تقسیم تعدادسطرها بر عدد ntile یک عدد صحیح باشد تا خروجی مناسب داشته باشیم.
و همچنین بایستی بدانیم که تعداد سطرهای جدول چنتاست تا آن را به گونه ای تقسیم کنیم که خارج قسمت برابر شود با عدد مورد نظر ما یعنی 6.

روش دوم:
در این روش بر خلاف روش قبل که همه چیز توسط تابع بدست می‌آمد باید خودمان دست به کار شویم و فرمولی را بدست آوریم که نتیجه مورد نظر را تولید کند.
برای حل این مساله ابتدا باید سطرهای جدول را 6 تا 6 تا سوا کنیم و عناصر هر دسته را شماره گذاری کنیم (از 1 تا 6 بر اساس ترتیب مقدار nbr) سپس با کمک سایر فرمول‌ها دسته‌ها را دوتا دوتا شماره گذاری میکنیم.

به تصویر زیر توجه بفرمایید:

	nbr	mk1	mk2	grp2	grp2
1	1	1	1	1	1
2	2	2	2	1	1
3	3	3	3	2	2
4	4	4	4	2	2
5	5	5	5	3	3
6	6	6	6	3	3
7	7	1	1	1	1
8	8	2	2	1	1
9	9	3	3	2	2
10	10	4	4	2	2
11	11	5	5	3	3
12	12	6	6	3	3

در کادر نارنجی رنگ همانطور که اشاره شد ما سطرهای شمارگذاری شده ای داریم که در رنج 1 تا 6 هستند. و در کادر بنفش ستون مورد نظر ما قرار دارد. ستون بنفش با کمک ستون نارنجی بدست آمده است.
اگر تقسیم صحیح را div و باقیمانده صحیح را mod بگیریم فرول‌های مورد نظر به این شرح خواهد بود:

$$(nbr - 1) mod\ 6 + 1$$

$$\left(\left((nbr - 1) mod\ 6 + 1\right) + 1\right) div\ 2$$

طبق کوئری زیر ستون نارنجی(rnk1/rnk2) را به دو طریق می‌توان ایجاد نمود و ستون بنفش(grp1/grp2) را نیز به دو طریق میتوان ایجاد نمود.

```
SELECT nbr, rnk1, rnk2,
       (rnk1 + 1) / 2 AS grp2,
       (rnk1 - 1) / 2 + 1 AS grp2
FROM
(
SELECT nbr,
       ROW_NUMBER() OVER(PARTITION BY (nbr + 5) / 6 ORDER BY nbr) rnk1,
       (nbr - 1) % 6 + 1 AS rnk2
FROM t
)d
```

عنوان: **حذف نمودن کاراکتر های ناخواسته توسط Recursive CTE قسمت اول**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۰۱ ۱۵:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, T-SQL, recursive cte, replace

شاید برایتان تا حالا پیش آمده باشد که بخواهید یکسری کاراکترهای ناخواسته و اضافه را از یک رشته حذف کنید. بطور مثال تمام کاراکتر هایی غیر عددی را باید از یک رشته حذف نمود تا آن رشته قابلیت تبدیل به نوع integer را بدست بیاورد.

اگر تعداد کاراکترهای ناخواسته محدود و مشخص هستند می‌توانید با دستور REPLACE آنها را حذف کنید، مثلا میخواهیم هر سه کاراکتر ~!@ از رشته حذف شوند:

```
DECLARE @s VARCHAR(50) = '~~~~~~!@@@@@@ salam';
SET @s = REPLACE(REPLACE(REPLACE(@s, '~', ''), '!', ''), '@', '');
SELECT @s AS new_string
```

ولی هنگامی که کاراکترها نامحدود بوده امکان نوشتن تابع REPLACE به کرات بی معنا است در این حالت باید دنبال روشی پویا و تعمیم پذیر بود.
با جستجویی که در اینترنت انجام دادم متوجه شدم تکنیک WHILE یا همون loop یکی از روش‌های رایج برای انجام اینکار هست، که احتمالا به دلیل سهولت در بکارگیری و سادگی آن بوده که عمومیت پیدا کرده است.
مستقل از این صحبتها هدف معرفی یک روش مجموعه گرا (set-based) برای این مساله می‌باشد.

حذف کاراکترها ناخواسته با تکنیک Recursive CTE

راه حل بر اساس جدول زیر است:

```
CREATE TABLE test_string
(id integer not null primary key,
 string_value varchar(500) not null);

INSERT INTO test_string
VALUES (1, '@@@@ #### salam 12345'),
       (2, 'good $$$$$ &&&&&& bye 00000');
```

حالا فرض کنید می‌خواهیم هر کاراکتری غیر از حروف الفبای انگلیسی و فاصله(space) از رشته حذف شود. پس دو داده فوق به صورت salam و good bye در انتها در خواهند آمد.
برای حذف کاراکترهای ناخواسته فوق query زیر را اجرا کنید.

```
WITH CTE (ID, MyString, Ix) AS
(
    SELECT id,
           string_value,
           PATINDEX('%[^a-z ]%', string_value)
    FROM   test_string

    UNION ALL

    SELECT id,
           CAST(REPLACE(MyString, SUBSTRING(MyString,Ix , 1), '') AS VARCHAR(500)),
           PATINDEX('%[^a-z ]%', REPLACE(MyString, SUBSTRING(MyString,Ix , 1), ''))
    FROM   CTE
    WHERE  Ix > 0
)
SELECT *
FROM   cte
--WHERE  Ix = 0;
ORDER BY id, CASE WHEN Ix = 0 THEN 1 ELSE 0 END, Ix;
```

توضیح query:

در قسمت anchor اندیس اولین کاراکتر ناخواسته (خارج از رنج حروف الفبا و فاصله) بدست می‌آید. سپس در قسمت recursive هر کاراکتری که برابر باشد با کاراکتر ناخواسته ای که در مرحله قبل بدست آمده از رشته حذف می‌شود این عملیات توسط تابع replace صورت می‌گیرد و اندیس کاراکتر ناخواسته بعدی بعد از حذف کاراکتر ناخواسته قبلی بدست می‌آید که به مرحله بعد منتقل می‌شود. این مراحل تا آنجایی پیش می‌رود که دیگر کاراکتر ناخواسته ای در رشته وجود نداشته باشد.

به جدول زیر توجه بفرمایید (خروجی query فوق)

	ID	MyString	Ix
1	1	@@@@ #### salam 12345	1
2	1	#### salam 12345	2
3	1	salam 12345	9
4	1	salam 2345	9
5	1	salam 345	9
6	1	salam 45	9
7	1	salam 5	9
8	1	salam	0
9	2	good $$$$$ &&&&&& bye 00000	6
10	2	good &&&&&& bye 00000	7
11	2	good bye 00000	12
12	2	good bye	0

نتیجه مطلوب ما آن دو سطری است که در کادر بنفش هستند. که اگر به ستون Ix اشان توجه کنید مقدارش برابر با 0 است.

لطفا به سطر اول جدول توجه بفرمایید مشاهده می‌شود که هر 4 کاراکتر @ یکباره از رشته حذف شدند که بدلیل استفاده از تابع REPLACE میباشد.

نظرات خوانندگان

نویسنده: سید حمزه
تاریخ: ۱۳۹۲/۰۶/۲۷ ۱۶:۵۳

سلام
خیلی جامع بود
فقط من متوجه نشدم دقیقا کجای کوئریم باید اینو بنویسم.
و همینطور میخواستم اگر بشه یک فانکشن بسازم و از اون هم توی select استفاده کنم.
ممنون

عنوان: **جستجوی کاراکترهای wildcards توسط ماده اختیاری ESCAPE**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۰۴ ۱۶:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, T-SQL, like, pattern matching

syntax کامل LIKE:

```
match_expression [ NOT ] LIKE pattern [ ESCAPE escape_character ]
```

چهار کاراکتر وجود دارد که الگوهای جستجو را توسط آنها تعریف می‌کنند. این کاراکترها wildcard نام دارند.
زمانی که کاراکترهای wildcard موجود در الگوی ما به عنوان کاراکترهای عادی در نظر گرفته شده اند نه wildcard آنگاه باید به SQL اطلاع دهیم که آنها را با کاراکترهای wildcard اشتباه نگیرد، برای این منظور چندین راه حل وجود دارد که در ادامه شرح خواهم داد.

فرض کنید الگوی شما این مقدار باشد:
"تنها دو کاراکتر % کاراکترهای wildcard هستند"

```
%[abcde_ ]%
```

روش اول: ESCAPE

```
WHERE string_value LIKE '%@[abcde@_ ]%' ESCAPE '@'
```

دو کاراکتر] و _ علامت گذاری (بوسیله @) و توسط ماده ESCAPE به عنوان کاراکترهای معمولی شناخته شدند.

روش دوم: Bracket

```
WHERE string_value LIKE '%[[]abcde[_] ]%'
```

دو کاراکتر] و _ را داخل [] قرار دادیم.

روش سوم: CHARINDEX

```
WHERE CHARINDEX('[abcde_ ]', string_value) > 0
```

عبارت را بدون هیچ گونه تغییری در تابع استفاده کردیم. این روش تنها زمانی مورد استفاده می‌تواند قرار گیرد که کاراکتر % در ابتدا یا انتهای (یا هر دو) آن قرار گرفته باشد.

تمرین: عباراتی را پیدا کنید که الگوی زیر در آن پیدا شود، فقط آخرین کاراکتر wildcard می‌باشد یعنی کاراکتر %

```
[[[%^ab'c'de]]]%
```

برای تست راه حلتان از متغیر جدولی زیر استفاده کنید:

```
declare @t table (s varchar(50))
insert @t values ('[[[%^ab''c''de]]]%'),
('[[[ %^ab''cde]]%'), ('[[%^ab''c''de]]%'),
('[[[%^abc''de]]]%');
```

عنوان: **ستون محاسباتی (computed column)**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۰۶ ۱۸:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: computed column, clustered index, execution plan, index seek, index scan, SQL Server

برخی از داده‌ها از ترکیب و ادغام شدن چند داده دیگر بدست می‌آیند. مثلا شماره دانشجویی از ترکیب چند صفت مختلف بوجود می‌آید (مثل نیمسال ورودی، کددانشگاه، کدرشته تحصیلی...).
برای پیاده سازی اینگونه ستون‌ها SQL Server یک قابلیتی به نام computed column ارائه داده است. برای تعریف این چنین ستون هایی بعد از نام ستون از کلمه AS استفاده می‌کنیم. عبارتی که ستون محاسباتی را تشکیل می‌دهد می‌تواند شامل این موارد باشد:
تابع، نام ستون غیر محاسباتی و مقادیر ثابت ولی امکان استفاده از subquery وجود ندارد.
ستون‌های محاسباتی بطور پیشفرض مجازی هستند (بطور فیزیکی بر روی دیسک ذخیره نشده اند). یعنی هر موقع که query اجرا می‌شود آنها نیز مجدد محاسبه شده و نمایش داده می‌شوند.
برای اینکه نوع ذخیره سازی را از مجازی به فیزیکی تبدیل کنیم باید در هنگام ساخت جدول (یا تغییر آن) از کلید واژه PERSISTED استفاده کنیم. وقتی بطور فیزیکی ذخیره شده باشد با هر بار ویرایش یکی از ستون‌های تشکلیل دهنده ستون محاسباتی هم ویرایش می‌شود.
ستون محاسباتی بعد از تبدیل شدن از مجازی به فیزیکی می‌تواند به عنوان کلید اولیه و ایندکس در نظر گرفته شود.

به مثال زیر توجه کنید:
جدولی داریم با دو ستون، قرار هست بر اساس ترکیب مقادیر دو ستون جستجویی انجام دهیم. ضمن اینکه ترکیب دو ستون باید منحصر بفرد باشد. برای این منظور یک unique index روی دو ستون لحاظ می‌کنیم.

```
create table t1
(
col1 char(1),
col2 char(1)
)

create unique nonclustered index ix_uq on t1 (col1 , col2);

insert t1
values('A', 'B'), ('B', 'C'), ('C', 'D'), ('D', 'E'), ('E', 'F'),
('G', 'H'), ('I', 'J'), ('K', 'L'), ('M', 'N'), ('O', 'P');
```

اکنون به دنبال سطری میگردیم که ترکیب مقادیر دو ستون آن برابر با OP باشد. پس query زیر را اجرا میکنیم

```
select col1 + col2
from t1
where col1 + col2 = 'OP'
```

اما همانطور که در تصویر زیر مشاهده می‌شود عمل Index Seek صورت نگرفته است. زمانی که از ستون به عنوان یک عبارت استفاده شود Index Seek نخواهیم داشت. منظور عبارت، الحاق مقداری با ستون، قرار گرفتن ستون در یک تابع و ... می‌باشد.

برای اینکه Index Seek داشته باشیم بایستی مقادیر را جداگانه مقایسه کنیم(ستون‌ها به صورت عبارت محاسباتی نباشند)

```
select col1 + col2
from t1
where col1  = 'O' and col2 = 'P'
```

ولی ما می‌خواهیم شرط بر اساس ترکیب دو ستون باشد. خب اینجا هست که Computed Columns مطرح میشوند.

```
alter table t1 add col3 as col1 + col2 persisted

create clustered index ix1 on t1 (col3)
```

با دستور اول یک ستون محاسباتی از نوع persisted به جدول اضافه نمودیم. و با دستور دوم یک Index روی ستون محاسباتی ایجاد نمودیم.

حال مجددا عمل جستجو را انجام میدهیم ولی به کمک ستون محاسباتی که اخیرا ایجاد نمودیم:

```
select *
from t1
where col3 = 'OP';
```

حالا مشاهده می‌شود که شاخص ix1 اسکن نشده است. و از آنجایی که شاخص از نوع Clustered است مشکل Covering هم نخواهیم داشت.

عنوان: **گذری بر مفاهیم relationship**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۰۶ ۲۲:۵
آدرس: www.dotnettips.info
گروه‌ها: Referential Integrity, foreign key, relationship, table, SQL Server

متاسفانه کاربران زیادی وجود دارند که هنوز درک صحیحی از جامعیت داده‌های ارجاعی (referential Integrity) ندارند. نمی‌دانند که relationship چیزی جز قید کلید خارجی (foreign key) نیست. در ادامه مفاهیم زیر را در حد آشنایی توضیح خواهم داد:

کلید خارجی ترکیبی (composite foreign key)

خود ارجاعی (self referencing)

اعمال تغییرات به صورت آبشاری (cascade)

چندین مسیر برای اعمال (multiple cascading path)

جدول اتصال (junction table)- ارتباط یک به یک

توسط دستور create table به دو شکل می‌توانیم بر روی ستون‌ها قید (کلید اولیه، check، کلید خارجی، کلید یونیک...) تعریف نمود:

قید ستونی

قید جدولی

syntax مربوط به قید کلید خارجی در مدل ستونی به صورت زیر است:

```
<column_constraint> ::=
[ CONSTRAINT constraint_name ]
{    ...

  | [ FOREIGN KEY ]
        REFERENCES [ schema_name . ] referenced_table_name [ ( ref_column ) ]
        [ ON DELETE { NO ACTION | CASCADE | SET NULL | SET DEFAULT } ]
        [ ON UPDATE { NO ACTION | CASCADE | SET NULL | SET DEFAULT } ]
        [ NOT FOR REPLICATION ]

  ...
}
```

نکته: بطور پیش فرض برای کلید خارجی اعمال update و delete روی وضعیت no action تنظیم شده است. به این معنا که اگر سعی کنیم کلید اولیه جدول مرجع را بروز رسانی یا حذف کنیم ممانعت به عمل خواهد آمد. برای رفع این مشکل هم میتوانید از طریق design اقدام کنید و هم در هنگام ساخت جدول توسط DDL (همانطور که در دستورات فوق مشاهده میشود).

کلید خارجی ترکیبی

زمانی که در جدول والد (parent) کلید اولیه ترکیبی باشد، هر جدولی که بخواهد به کلید جدول والد ارجاعی داشته باشد باید از ترکیب دو ستون برای ساخت کلید خارجی استفاده کند.

فرض کنید جدول parent به این صورت است (ترکیب دو ستون col1 و col2 کلید اولیه است)

```
create table parent
(
col1 int not null,
col2 int not null,
col3 char(1) null,
-- Composite Primary Key
primary key(col1, col2)
);
```

در اینجا چون ترکیب دو ستون کلید اولیه هست باید توسط "قید جدولی" اقدام به تعریف کلید کرد

و جدول child که دارای قید کلید خارجی ترکیبی به نام fk_comp است و به جدول parent ارجاع داده است:

```
create table child
(
col0 int primary key,
col1 int null,
col2 int null,
-- Composite Foreing Key Constraint
constraint fk_comp
foreign key (col1, col2)
references parent(col1, col2)
);
```

در این DDL هم از قید جدولی برای تعریف کلید خارجی ترکیبی استفاده شده است.

نمودار این دو جدول:

پس به عنوان نتیجه گیری، هرگاه جدول اصلی دارای کلید ترکیبی بود در جداول child نیز باید از کلید خارجی ترکیبی برای ایجاد relationship استفاده نمود.

اما این دو جدول را به یک شیوه دیگر نیز می‌توان طراحی نمود. در جدول parent ترکیب دو ستون col1 و col2 را منحصربفرد (unique) گرفته و ستونی دیگر (مثلا از نوع identity) را به عنوان کلید اولیه در نظر گرفت (یا یک ستون از نوع محاسباتی تعریف کرده و آن را کلید قرار داد)

```
create table parent
(
col0 int not null primary key identity,
col1 int not null,
col2 int not null,
col3 char(1) null,
-- Composite Unique Key
unique(col1, col2)
);

create table child
(
col0 int primary key,
col1 int null references parent
);
```

خود ارجاعی و multiple cascading path

فرض کنید بخش‌های مختلف یک سازمان که بصورت چارت است را توسط جدول پیاده سازی کردیم. ستون‌های جدول به این شرح هستند:

کد بخش
نام بخش
کد بخش بالایی

ستون "کد بخش بالایی" نیز خود یک بخش است. برای پیاده سازی این چنین ساختارهایی از جدول زیر کمک گرفته می‌شود:

```
create table chart
(
chart_nbr int not null primary key,
parent_nbr int null references chart,
chart_name varchar(5) null
);
```

تصویر نمودار جدول chart

حالا فرض کنید میخواهیم اطلاعات نامه هایی که بین بخش‌ها رد و بدل میشود را در یک جدول ذخیره کنیم. جدول دارای ستون‌های زیر خواهد بود:

شماره نامه
کد بخش فرستنده
کد بخش گیرنده
ستون شماره نامه کلید اولیه و دو ستون دیگه کلیدهای خارجی هستند که به جدول chart مراجعه می‌کنند:

```
create table letters
(
letter_nbr int primary key,
sec_sender int not null references chart,
sec_reciver int not null references chart
);
```

نمودار جدول نامه‌ها و چارت:

نکته ای که در اینجا وجود دارد این است که اگر کلید جدول chart بروز شود آنگاه SQL Server از دو راه می‌تواند جدول letters را بروز رسانی کند، به این علت پیغام خطایی با عنوان multiple cascading paths صادر می‌شود. برای رفع این مشکل باید از trigger کمک گرفت.

جدول اتصال (junction table)

برای پیاده سازی رابطه N-N از جدول واسط کمک گرفته می‌شود. برای این منظور رابطه N-N را باید به دو رابطه N-1 تجزیه کرد.

فرض کنید یک جدول مربوط به خلبانان و جدول دیگر مربوط به مسیرهای پروازی (مثل مسیر ایران-ترکیه، ایران-عربستان...) است. یک خلبان ممکن است در چند مسیر پروازی هواپیما را هدایت کرده باشد و یا بالعکس یک مسیر پروازی ممکن است توسط N خلبان طی شده باشد.

برای پیاده سازی اینگونه سیستم هایی باید یک جدول ایجاد نمود که دارای دو کلید خارجی باشد یکی آنها به جدول خلبانان و دیگری به مسیرهای پروازی مرتبط است.

می‌توان ترکیب دو کلید خارجی جدول واسط را کلید اولیه در نظر گرفت.

پس خواهیم داشت:

```
create table pilot
(
pilot_code int primary key,
pilot_name varchar(20)
);

create table paths
(
path_code int primary key,
path_name varchar(20)
);
```

```
create table junction
(
pilot_code int references pilot,
path_code int references paths,
primary key (pilot_code, path_code)
);
```

و نمودار آن:

رابطه یک به یک

زمانی که نمونه‌های محدودی از یک موجودیت دارای مقدار برای یکسری خصیصه هستند بهتر است جدول به دو جدول تجزیه شود تا فضای اضافی صرف جدول نشود. مثلا در مدرسه تنها 10 درصد دانش آموزان جزء تیم فوتبال هستند حال اگر بخواهیم اطلاعات مربوط به تیم فوتبال مثل تعداد گل زده، تعداد بازی ... در جدول اصلی ذخیره کنیم برای 90 درصد دانش آموزان مقداری نخواهیم داشت. برای حل این مساله ارتباط یک به یک پیشنهاد می‌شود.

```
create table student
(
std_code int primary key,
std_name varchar(25) not null
);

create table football
(
std_code int primary key
  constraint one_to_one_fk
  references student,
std_cnt_goal int not null
  default (0)
);
```

توجه داشته باشید که ستون std_code هم کلید اولیه هست و هم کلید خارجی که به جدول student ارجاع داده شده است.

نتیجه گیری

یک ستون همزمان می‌تواند کلید اولیه باشد و هم کلید خارجی (مثلا در ارتباط یک به یک)

همانطور که کلید اولیه ترکیبی داریم به همان شکل هم کلید خارجی ترکیبی داریم.

یک جدول می‌تواند به خودش ارجاع دهد که به آن اصطلاحا self-referencing می‌گویند

relationship چیزی جز کلید خارجی نیست و کلید خارجی نیز چیزی جز یک قید برای جامعیت داده‌ها نیست

جامعیت داده ارجاعی را می‌توان توسط trigger پیاده سازی کرد

اگر SQL Server بیش از یک مسیر برای تغییر جدول child داشته باشد با مشکل مواجه خواهید شد

نظرات خوانندگان

نویسنده: Mohsen
تاریخ: ۱۳۹۱/۱۱/۰۷ ۸:۱۵

بسیار عالی بود.مفاهیم پایگاه داده برای جامعه نرم افزار ایران بسیار حیاتی هستند.

نویسنده: mohammad
تاریخ: ۱۳۹۱/۱۱/۰۸ ۲۰:۴۰

با سلام.ممنون از مطلبتون. لطفا توضیحی راجع به اینکه چطور میتوان با دادن یک id تمام ریشه‌های اون رو درحالت خودارجاعی درآورد بدید.

نویسنده: mohammad
تاریخ: ۱۳۹۱/۱۱/۰۸ ۲۰:۵۵

ببخشید منظورم برگ‌ها بود که اشتباه نوشتم ریشه. با تشکر

نویسنده: محمد سلم ابادی
تاریخ: ۱۳۹۱/۱۱/۰۸ ۲۱:۵۶

سلام،
برای این منظور باید از recursive cte کمک گرفت.
فرض کنید درختی به شکل زیر داریم:

```
/*         A
         /   \
        B     C
        |    /|\
        D   E F G
        |
        H

*/
```

و هدف بدست آوردن تمام زیر شاخته‌های گره A است.
ابتدا باید تمام گره هایی که مقدار گره پدرشان برابر با A است را بدست بیاریم یعنی گره‌های B و C
حالا باید تمام گره هایی که گره پدرشان B و یا C است را بدست بیاریم یعنی گره‌های D E F G
و در مرحله بعد باید تمام گره هایی را بدست بیاریم که گره پدرشان برابر با یکی از مقادیر بدست آمده در مرحله قبل (یعنی D E F G) یعنی H

این الگوریتم را توسط Recursive CTE پیاده میکنیم:

```
declare @t table
(
   id char(1) primary key not null,
   pid char(1) null --references @t
);

insert @t
values ('A', null), ('B','A'),('C','A'),
('D','B'), ('H','D'),('E','C'),('F','C'),('G','C');

with cte as
(
select id
from @t
where pid = 'A'
```

```
union all

select t.id
from cte c
join @t t
on t.pid = c.id
)
select * from cte
```

موفق باشید

نویسنده: mohammad
تاریخ: ۱۳۹۱/۱۱/۰۹ ۱۰:۵۳

ممنون خیلی هم خوب.
اما یک سوال؟ آیا این روش فشار زیادی روی سرور ایجاد نمیکنه. مثلا در یک برنامه تحت وب با تعداد زیاد کاربر؟

نویسنده: محمد سلم ابادی
تاریخ: ۱۳۹۱/۱۱/۰۹ ۱۲:۱۵

این یک روش معمولی و رایج هست که همگان دارن ازش استفاده می‌کنند. بعید می‌دانم که در عملکرد مشکلی داشته باشد.
اگر فرضا سرعت اجرای query اتان پایین بود یک اندیس (index) روی ستون "کد والد" تعریف کنید
به این شکل

```
create nonclustered index ix_parent on table_name (parent_id)
```

عنوان: **حذف کاراکتر های ناخواسته توسط Recursive CTE قسمت دوم**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۰۸ ۸:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, recursive cte

مقدمه

در قسمت پیشین نشان داده شد که چگونه کاراکترهای خارج از رنج حروف الفبای انگلیسی از عبارات موجود در یک جدول حذف شدند.

اکنون شرایط کمی تغییر کرده است کاراکترهای ناخواسته در قالب یک مجموعه (جدول) به ما ارائه داده می‌شوند. ما بایستی تمام کاراکترهای داده شده را از عبارات (موجود در جدول) در صورت تطابق حذف کنیم.

جدول کاراکترهای ناخواسته Unwanted و جدول داده‌ها Data نامگذاری شده اند.

```
CREATE TABLE Data
(
id INTEGER NOT NULL PRIMARY KEY IDENTITY,
data VARCHAR(50) NOT NULL
);

INSERT INTO Data
VALUES ('~!hasan @#$%^&*(reza)[ali]^^^^^^^^'),
('(Ja[][][]va~!@#$*d-mohammad)'),
('Mohammad'), ('Maryam');

CREATE TABLE Unwanted
(
id INT NOT NULL PRIMARY KEY IDENTITY,
chars CHAR(1) NOT NULL UNIQUE
);

INSERT Unwanted
VALUES ('~'),('!'),('@'),('#'),('$'),('%'),
('['),(']'),('^'),('&'),('*');
```

الگوریتم

قبل از هر چیزی فرض میگیریم که داده‌ها ستون Id جدول Unwanted کاملا متوالی نیستند و gap بین مقادیر id وجود دارد. پس برای داشتن مقادیر متوالی از تابع row_number استفاده شده است.

بعد از آن باید عباراتی که حداقل با یک کاراکتر ناخواسته تطابق پیدا میکنند انتخاب شوند. و عملیات پاکسازی روی این عبارات به تعداد کاراکترهای موجود در جدول unwanted انجام می‌شود. یعنی در مرحله اول شمارنده برابر است با تعداد کاراکترهای ناخواسته سپس در هر فراخوانی یک مقدار از این شمارنده کم شده تا اینکه به 0 برسد در اینجا کار به اتمام می‌رسد.

بعد از پایان یافتن فراخوانی‌های query بازگشتی باید سطرهایی که شمارنده آنها برابر با 0 است انتخاب شده و عباراتی که در مرحله اول به دلیل عدم تطابق انتخاب نشدن بایستی به نتیجه اضافه شوند. این کار را توسط Union all انجام شده است.

```
WITH pre AS
(
SELECT ROW_NUMBER() OVER(ORDER BY id) AS id, chars
  FROM Unwanted
),
 cte AS
(
SELECT  data.id,
        nbr,
        CAST(Data AS VARCHAR(50)) AS Data
  FROM Data
       CROSS JOIN (SELECT COUNT(*) AS nbr FROM Unwanted)t
 WHERE EXISTS
       (SELECT *
          FROM Unwanted
         WHERE Data LIKE '%#' + chars + '%' ESCAPE '#')

 UNION ALL

 SELECT C.id,
        nbr - 1,
        CAST(REPLACE(C.data, U.chars, '') AS VARCHAR(50))
   FROM cte AS C
```

```
        INNER JOIN pre U
        ON C.nbr = U.id
  WHERE nbr > 0
)
SELECT data
  FROM cte
 WHERE nbr = 0

 UNION ALL

SELECT data
  FROM Data
 WHERE NOT EXISTS
       (SELECT id
          FROM cte
         WHERE cte.id = data.id);
```

عنوان: **از کار افتادن SQL Server Agent**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۱۲ ۸:۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server

SQL Server Agent مربوط به SQL Server 2008 از کار افتاده بود و راه اندازی نمی‌شد. خطای مرتبط با آن در لاگ‌های ویندوز به نحو زیر بود:

```
SQLServerAgent could not be started (reason: Unable to connect to server '(local)'; SQLServerAgent
cannot start).
```

پس از مدتی جستجو، عنوان شده بود که مسیر رجیستری زیر را یافته:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Microsoft SQL Server\MSSQL10.MSSQLSERVER\SQLServerAgent
```

و در آن ServerHost را به نام سرور ویرایش کنید. سپس سرور را ری استارت نمائید. این تغییر انجام شد اما باز هم SQL Server Agent راه اندازی نمی‌شد.
لاگ‌های آن در مسیر ذیل ثبت می‌شوند:

```
C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\Log\SQLAGENT.OUT
```

در اینجا خطاهای زیر ثبت شده بودند:

```
2013-01-28 20:02:34 - ! [298] SQLServer Error: 780, SQL Server Network Interfaces: The logon attempt
failed [SQLSTATE HY000]
2013-01-28 20:02:34 - ! [298] SQLServer Error: 780, Cannot generate SSPI context [SQLSTATE HY000]
2013-01-28 20:02:34 - ! [000] Unable to connect to server ; SQLServerAgent cannot start
2013-01-28 20:02:34 - ! [298] SQLServer Error: 780, SQL Server Network Interfaces: The logon attempt
failed [SQLSTATE HY000]
2013-01-28 20:02:34 - ! [298] SQLServer Error: 780, Cannot generate SSPI context [SQLSTATE HY000]
2013-01-28 20:02:34 - ! [382] Logon to server failed (DisableAgentXPs)
2013-01-28 20:02:34 - ? [098] SQLServerAgent terminated (normally)
```

همانطور که مشخص است مشکل اصلی در عدم توانایی لاگین اکانت SQL Server Agent ذکر شده است.
برای تغییر آن مسیر زیر را طی کنید:

```
SQL Configuration manager -> SQL Server Agent -> Logon User -> NT/Local Service
```

به عبارتی در قسمت SQL Configuration manager، تنظیمات SQL Server Agent را یافته و نوع اکانت آن‌را به Local Service تغییر دهید.
پس از آن سرویس بدون مشکل استارت شد.

عنوان: **بدست آوردن برگهای یک درخت توسط Recursive CTE**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۱۶ ۱۷:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, recursive cte, query, not exists, dense_rank

امروز در یک تالار سوالی مطرح شد با این عنوان "چگونه می‌توانم گره‌های برگ یک شاخه را بدست بیاورم". خب راه حلی که فورا به ذهنم رسید استفاده از یک query بازگشتی (recursive) بود.
به ساختار سلسله مراتبی زیر توجه بفرمایید:

گره هایی که با رنگ سبز علامت گذاری شده اند را گره‌های برگ می‌نامیم چون که آن گره‌ها بدون زیر شاخه هستند.
فرض کنید از ما خواسته شده است با داشتن گره A تمام برگهای این شاخه را بدست بیاوریم.
دو مرحله را باید طی کنیم ابتدا تمام گره هایی که زیر شاخه گره A هستند را باید بدست آورد سپس توسط یک گزاره گره‌های برگ را فیلتر کنیم.

در واقع گره هایی برگ هستند که پدر هیچ گره‌ی دیگری نباشند.

```
declare @t table
(id char(1) primary key,
parent char(1));

insert @t values
('A',null),                                  --Level 1
('B', 'A'), ('C', 'A'),                      --Level 2
('D', 'B'), ('E', 'B'),('R','B'), ('F', 'C'), --Level 3
('G', 'D'),  --Level 4
('H', 'G'), ('I', 'G');                      --Level 5
```

```
;with cte as
(
select id, rnk=0 from @t
where parent = 'A'
```

```
union all

select t.id, rnk+1
from cte join @t t
on cte.id = t.parent
)
select *
  from cte
 where not exists
       (select *
          from @t
         where parent = cte.id);
```

و حالا به درخت زیر توجه بفرمایید:

هدف پیدا کردن برگ هایی از شاخه مورد نظر است که در پایین‌ترین سطح قرار گرفته باشند. برای این منظور از همان query بازگشتی استفاده کرده و با کمک تابع dense_ranke گره‌های مورد نظر را بدست میاوریم.

```
;with cte as
(
select id, rnk=0 from @t
where parent = 'A'

union all

select t.id, rnk+1
from cte join @t t
on cte.id = t.parent
)
select *
from
(
   select *, dense_rank() over(order by rnk desc) rk
     from cte
)t
where rk = 1
```

نظرات خوانندگان

نویسنده: بهمن خلفی
تاریخ: ۱۳۹۱/۱۱/۱۷ ۱۴:۴۱

ممنون از مطلب مفیدتون

اما یک سوال : چگونه میتوان به نتیجه زیر دست پیدا کرد

```
a , ab , ac , abd , abe , abr , abdg , abdgh , ....
```

نویسنده: محمد سلم ابادی
تاریخ: ۱۳۹۱/۱۱/۱۷ ۱۵:۳۰

من دقیقا متوجه نشدم نتیجه مورد نظر شما چیست.

آیا نتیجه مورد نظر شما به صورت الحاق یافته (concatenated) هست یا نه؟

در هر صورت باید یکی از دو query زیر نتیجه مورد نظر شما را تولید کند.

```
declare @t table
(id char(1) primary key,
parent char(1));

insert @t values
('A',null),                                  --Level 1
('B', 'A'), ('C', 'A'),                      --Level 2
('D', 'B'), ('E', 'B'),('R','B'), ('F', 'C'), --Level 3
('G', 'D'),                                  --Level 4
('H', 'G'), ('I', 'G');                      --Level 5

;with cte as
(
select id, rnk=0,
       concats = cast(id as varchar(10))
from @t
where parent is null

union all

select t.id, rnk+1,
       cast(cte.concats + t.id as varchar(10))
from cte join @t t
on cte.id = t.parent
)
select * from cte
/*
id   rnk         concats
---- ----------- ----------
A    0           A
B    1           AB
C    1           AC
F    2           ACF
D    2           ABD
E    2           ABE
R    2           ABR
G    3           ABDG
H    4           ABDGH
I    4           ABDGI
*/
;with cte as
(
select id, rnk=0,
       concats = cast(id as varchar(10))
from @t
where parent is null

union all

select t.id, rnk+1,
       cast(cte.concats + t.id as varchar(10))
from cte join @t t
on cte.id = t.parent
)
```

```
select stuff(d.list,1,1,'') as concats
from (select ','+concats
      from cte
  for xml path(''))d(list)
/*
concats
----------------------------------------
A,AB,AC,ACF,ABD,ABE,ABR,ABDG,ABDGH,ABDGI
*/
```

موفق باشید

عنوان: **بررسی مساله متداول Top N در نسخه های مختلف SQL Server**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۲۱ ۲۳:۴۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, querying, Top N

مقدمه (شرح مساله)

چندی پیش در تالار T-SQL سوالی مطرح شد راجع به مساله ای که معروف است به top N per group.
تنها موضوعی که باعث شد من مطلبی راجع به آن بنویسم محدودیتی بود که کاربر مورد نظر داشت؛ که آن محدودیت چیزی نبود جز: query بایستی در نسخه 2000 جوابگو باشد.

قطعا شده است که بخواهید مثلا به ازای هر مشتری آخرین سفارش آن را انتخاب کنید. این مساله Top N نامیده می‌شود.

فرض کنید جدولی داریم که حاوی سفارشات مشتریان می‌باشد. هر مشتری می‌تواند چندین سفارش داشته باشد؛ هر سفارش دارای حداقل دو مقدار "تاریخ سفارش" و "مبلغ سفارش است". هدف پیدا کردن آخرین سفارشات هر مشتری می‌باشد.
نکته: اگر چند تاریخ برای آخرین سفارش مشتری وجود داشت آنگاه بایستی بر اساس مبلغ سفارش مرتب سازی نزولی صورت بگیرد. یا به عبارت دیگر ابتدا باید مرتب سازی نزولی بر اساس ستون تاریخ سفارش انجام شود و سپس مرتب سازی نزولی بر اساس ستون مبلغ سفارش.

فرض می‌گیریم داده‌های جدول ما چیزیست شبیه به این:

Row_id	Customer_id	Order_date	Order_value
6	1	10	15
9	1	10	5
4	1	9	10
3	1	8	20
10	2	12	15
7	2	10	15
11	3	15	5
5	3	10	20
8	3	10	10
1	3	5	15
2	3	5	15

سطرهایی از جدول که رنگی شده اند سطرهای مورد نظر ما هستند که باید در خروجی ظاهر شوند.
داده‌های جدول با کمک قابلیت Sort نرم افزار word مرتب سازی شده اند، این تصویر را به این خاطر در اینجا قرار دادم چون که دیدم می‌تواند در شفاف سازی مساله به من کمک کند.
ابتدا مرتب سازی نزولی بر اساس ستون order_date انجام گرفته و سپس مرتب سازی نزولی بر اساس ستون order_value. و در پایان اولین سطر مربوط به هر مشتری به عنوان خروجی مورد نظر انتخاب می‌شوند.

راه حل ها

خب پر واضح است که در نسخه 2005 و بعد از آن ساده‌ترین و بهینه‌ترین راه حل استفاده از تابع row_number می‌باشد.

```
SELECT row_id, customer_id, order_date, order_value
  FROM (SELECT *,
               ROW_NUMBER() OVER(PARTITION BY customer_id
                                 ORDER BY order_date DESC, order_value DESC) AS rnk
          FROM table_name
       )t
 WHERE rnk = 1;
```

اما با محدودیتی که در نسخه 2000 وجود دارد راه حلی بهتر از این پیدا نخواهیم کرد:

```
SELECT *
  FROM table_name t
 WHERE row_id = (SELECT TOP 1 row_id
                   FROM table_name
                  WHERE customer_id = t.customer_id
                  ORDER BY order_date DESC, order_value DESC);
```

حالا چه میشود راه حلی بخواهیم مستقل از هر یک از نسخه‌های SQL Server:

```
SELECT MIN(row_id) AS row_id, customer_id, order_date, order_value
    FROM table_name t
   WHERE order_date =
```

```
        (SELECT MAX(order_date)
           FROM table_name
          WHERE customer_id = t.customer_id)
    AND order_value =
        (SELECT MAX(order_value)
           FROM table_name
          WHERE customer_id = t.customer_id
            AND order_date =
                (SELECT MAX(order_date)
                   FROM table_name
                  WHERE customer_id = t.customer_id))
                  GROUP BY customer_id, order_date, order_value;
```

عنوان: **فعال و غیر فعال کردن قیود**
نویسنده: محمد سلیم ابادی
تاریخ: ۱۳۹۱/۱۱/۲۴ ۲۰:۵۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, foreign key, constraint

SQL Server قابلیت فعال(enable) و غیر فعال(disable) کردن دو قید کلید خارجی و check را برای ما مهیا کرده است.
ما می‌توانیم بعد از ساخت جدول و انتشار مقداری داده در آن قیدهایی را ایجاد کنیم. بطور پیشفرض اگر شرط قید ما بر قرار بود قید به طور صحیح ساخته می‌شود و اگر شرط قید ما بر قرار نباشد قید با خطای conflict مواجه خواهد شد.
بطور کلی غیر فعال کردن قیدها کار درستی نیست. ولی در برخی مواقع برای تسریع در اجرای کد می‌توانیم قید را غیر فعال کنیم.
بطور مثال اگر یک میلیون داده قرار است در جدول درج شود و مطمئن هستیم که این داده‌ها جامعیت داده‌ها را حفظ می‌کنند آنگاه می‌توانیم قید را برای تسریع در عمل درج بطور موفق غیر فعال کنیم.

فعال و غیر فعال کردن از طریق DDL

با غیر فعال کردن قیود داده‌ها را در وضعیت نامناسبی قرار می‌دهیم ولی همان طور که اشاره شد بطور موفق اشکالی پیش نخواهد آمد.
در ادامه ابتدا طریقه غیرفعال کردن و مجددا فعال کردن قیود را توسط دستور alter table نشان خواهم داد سپس به سراغ امکانات ویزاردی می‌رویم. ابتدا یک جدول تک ستونه ایجاد می‌کنیم:

```
CREATE TABLE testTable
(column1 integer not  null);
```

الان هیچ قیدی روی جدول لحاظ نشده است. پس هر داده که در رنج domain ستون باشد را میتوانیم درج کنیم. پس بطور نمونه این داده‌ها را درج میکنیم:

```
INSERT INTO testTable
VALUES (-10), (0), (10), (20), (30), (40)
```

حالا تصمیم داریم قیدی روی ستون column1 بگذاریم که توسط آن تنها اعداد مثبت در جدول درج شوند. پس داریم:

```
ALTER TABLE testTable
WITH CHECK
ADD CONSTRAINT NoNegative
CHECK (column1 > 0);
```

ولی چون داده هایی در جدول از قبل وجود داشته اند که قید ما را نقض کنند این قید ساخته نخواهد شد و با پیغام زیر مواجه خواهیم شد:

```
The ALTER TABLE statement conflicted with the CHECK constraint "NoNegative".
```

برای ساخت این قید روی این داده‌ها تنها راه استفاده از کلید واژه‌های WITH NOCHECK است یعنی:

```
ALTER TABLE testTable
WITH NOCHECK
ADD CONSTRAINT NoNegative
CHECK (column1 > 0);
```

و اکنون سعی میکنیم یک مقدار منفی در جدول درج کنیم:

```
INSERT INTO testTable VALUES (-5)
  /*
 The INSERT statement conflicted with the CHECK constraint "NoNegative".
 */
```

اما قیدی که ساخته بودیم در جدول در حال اعمال شدن است. برای درج مقدار منفی باید غیر را غیر فعال کنیم.

```
ALTER TABLE TestTable
NOCHECK CONSTRAINT NoNegative
```

و حالا مقدار منفی را درج میکنیم. و برای برگرداندن وضعیت NOCHECK به وضعیت CHECK باید از کلید واژه‌های WITH NOCHECK استفاده کنیم. چرا که داده هایی در جدول درج شده اند که قید مورد نظر ما را نقض می‌کنند.

```
ALTER TABLE TestTable
WITH NOCHECK
CHECK CONSTRAINT NoNegative
```

فعال و غیر فعال کردن از طریق design

در قسمت object explorer قید مورد نظر را پیدا کرده و روی آن راست کلیک کرده و گزینه Modify را انتخاب کنید. سپس در پنجره باز شده در قسمت Table Designer تغییرات مورد نظر خود را اعمال کنید.

عنوان: مروری بر طراحی Schema less بانک اطلاعاتی SisoDb
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۲۵ ۱۵:۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, NoSQL, SisoDb

اس کیوال سرور، از سال 2005 به بعد، به صورت توکار امکان تعریف و ذخیره سازی اطلاعات schema less و یا schema free را به کمک فیلدهایی از نوع XML ارائه داده است؛ به همراه یکپارچگی آن با زبان XQuery برای تهیه کوئری‌های سریع سمت سرور. در فیلدهای XML می‌توان اطلاعات انواع و اقسام اشیاء را بدون اینکه نیازی به تعریف تک تک فیلدهای مورد نیاز، در بانک اطلاعاتی وجود داشته باشد، ذخیره کرد. یک نمونه از کاربرد چنین امکانی، نوشتن برنامه‌های «فرم ساز» است. برنامه‌هایی که کاربران آن می‌توانند فیلد اضافه و کم کرده و نهایتا اطلاعات را ذخیره و از آن‌ها کوئری بگیرند.
خوب، این فیلد کمتر بحث شده XML، فقط در اس کیوال سرور و نگارش‌های اخیر آن وجود دارد. اگر نیاز به کار با بانک‌های اطلاعاتی سبک‌تری وجود داشت چطور؟ یک راه حل عمومی برای این مساله مراجعه به روش‌های NoSQL است. یعنی بطور کلی بانک‌های اطلاعاتی رابطه‌ای کنار گذاشته شده و به یک سکوی کاری دیگر سوئیچ کرد. در این بین، SisoDb راه حل میانه‌ای را ارائه داده است. با کمک SisoDb می‌توان اطلاعات را به صورت schema less و بدون نیاز به تعریف فیلدهای متناظر آن‌ها، در انواع و اقسام بانک‌های اطلاعاتی SQL Server با فرمت JSON ذخیره و بازیابی کرد. این انواع و اقسام، شامل SQL Server CE نیز می‌شود.

دریافت و نصب SisoDb

دریافت و نصب SisoDb بسیار ساده است. به کمک package manager و امکانات NuGet، کلمه Sisodb را جستجو کنید. در بین مداخل ظاهر شده، پروایدر مورد علاقه خود را انتخاب و نصب نمائید. برای مثال اگر قصد دارید با SQL Server CE کار کنید، SisoDb.SqlCe4 را انتخاب و یا اگر SQL Server 2008 مدنظر شما است، SisoDb.Sql2008 را انتخاب و نصب نمائید.

ثبت و بازیابی اطلاعات به کمک SisoDb

کار با SisoDb بسیار روان است. نیازی به تعاریف نگاشت‌ها و ORM خاصی نیست. یک مثال مقدماتی آن‌را در ادامه ملاحظه می‌کنید:

```
using SisoDb.Sql2008;

namespace SisoDbTests
{
    public class Customer
    {
        public int Id { get; set; }
        public int CustomerNo { get; set; }
        public string Name { get; set; }
    }

    class Program
    {
        static void Main(string[] args)
        {
            /*var cnInfo = new SqlCe4ConnectionInfo(@"Data source=sisodb2013.sdf;");
            var db = new SqlCe4DbFactory().CreateDatabase(cnInfo);
            db.EnsureNewDatabase();*/

            var cnInfo = new Sql2008ConnectionInfo(@"Data Source=(local);Initial
Catalog=sisodb2013;Integrated Security = true");
            var db = new Sql2008DbFactory().CreateDatabase(cnInfo);
            db.EnsureNewDatabase();

            var customer = new Customer
            {
                CustomerNo = 20,
                Name = "Vahid"
            };
            db.UseOnceTo().Insert(customer);

            using (var session = db.BeginSession())
            {
                var info = session.Query<Customer>().Where(c => c.CustomerNo == 20).FirstOrDefault();

                var info2 = session.Query<Customer>().Where(c => c.CustomerNo == 20 &&
c.Name=="Vahid").FirstOrDefault();
```

```
            }
        }
    }
}
```

در این مثال، ابتدا اتصال به بانک اطلاعاتی برقرار شده و سپس بانک اطلاعاتی جدید تهیه می‌شود. سپس یک وهله از شیء مشتری ایجاد و ذخیره می‌گردد. در ادامه دو کوئری بر روی بانک اطلاعاتی انجام شده است.

ساختار داخلی SisoDb

SisoDb به ازای هر کلاس، حداقل 9 جدول را ایجاد می‌کند. در ادامه نحوه ذخیره سازی شیء مشتری ایجاد شده و مقادیر خواص آن‌را نیز مشاهده می‌نمائید:

```
1 SELECT TOP 1000 [RowId]
2        ,[StructureId]
3        ,[MemberPath]
4        ,[Value]
5        ,[StringValue]
6   FROM [sisodb2013].[dbo].[CustomerIntegers]
```

Results | Messages

	RowId	StructureId	MemberPath	Value	StringValue
1	1	1	Id	1	1
2	2	1	CustomerNo	20	20

ذخیره سازی جداگانه خواص عددی

```
1 SELECT TOP 1000 [RowId]
2        ,[StructureId]
3        ,[MemberPath]
4        ,[Value]
5   FROM [sisodb2013].[dbo].[CustomerStrings]
```

Results | Messages

	RowId	StructureId	MemberPath	Value
1	1	1	Name	Vahid

ذخیره سازی جداگانه خواص رشته‌ای

```
1 SELECT TOP 1000 [RowId]
2        ,[StructureId]
3        ,[Json]
4   FROM [sisodb2013].[dbo].[CustomerStructure]
```

Results | Messages

	RowId	StructureId	Json
1	1	1	{"Id":1,"CustomerNo":20,"Name":"Vahid"}

ذخیره سازی کلی شیء مشتری با فرمت JSON به صورت یک رشته

همانطور که ملاحظه می‌کنید، یک جدول کلی SisoDbIdentities ایجاد شده است که اطلاعات نام اشیاء را در خود نگهداری می‌کند. سپس اطلاعات خواص اشیاء یکبار به صورت JSON ذخیره می‌شوند؛ با تمام اطلاعات تو در توی ذخیره شده در آن‌ها و همچنین یکبار هم هر خاصیت را به صورت یک رکورد جداگانه، بر اساس نوع کلی آن‌ها، در جداول رشته‌ای، عددی و امثال آن ذخیره می‌کند.

شاید بپرسید که چرا به همان فیلد رشته‌ای JSON اکتفاء نشده است؟ از این جهت که پردازشگر سمت بانک اطلاعاتی آن همانند فیلدهای XML در SQL Server و نگارش‌های مختلف آن وجود ندارد (برای مثال به کمک زبان T-SQL می‌توان از زبان XQuery در خود بانک اطلاعاتی، بدون نیاز به واکشی کل اطلاعات در سمت کلاینت، به صورت یکپارچه استفاده کرد). به همین جهت برای کوئری گرفتن و یا تهیه ایندکس، نیاز است این موارد جداگانه ذخیره شوند.

به این ترتیب زمانیکه کوئری تهیه می‌شود، برای مثال:

```
var info = session.Query<Customer>().Where(c => c.CustomerNo == 20).FirstOrDefault();
```

به کوئری زیر ترجمه می‌گردد:

```
SELECT DISTINCT TOP(1) (s.[StructureId]),
                       s.[Json]
                FROM   [CustomerStructure] s
                       LEFT JOIN [CustomerIntegers] mem0
                            ON  mem0.[StructureId] = s.[StructureId]
                            AND mem0.[MemberPath] = 'CustomerNo'
                WHERE  (mem0.[Value] = 20);
```

و یا کوئری ذیل:

```
var info2 = session.Query<Customer>().Where(c => c.CustomerNo == 20 &&
c.Name=="Vahid").FirstOrDefault();
```

معادل زیر را خواهد داشت:

```
SELECT DISTINCT TOP(1) (s.[StructureId]),
                       s.[Json]
                FROM   [CustomerStructure] s
                       LEFT JOIN [CustomerIntegers] mem0
                            ON  mem0.[StructureId] = s.[StructureId]
                            AND mem0.[MemberPath] = 'CustomerNo'
                       LEFT JOIN [CustomerStrings] mem1
                            ON  mem1.[StructureId] = s.[StructureId]
                            AND mem1.[MemberPath] = 'Name'
                WHERE  ((mem0.[Value] = 20) AND (mem1.[Value] = 'Vahid'));
```

در هر دو حالت از جداول کمکی تعریف شده برای تهیه کوئری استفاده کرده و نهایتا فیلد JSON اصلی را برای نگاشت نهایی به اشیاء تعریف شده در برنامه بازگشت می‌دهد.

در کل این هم یک *روش تفکر* و طراحی Schema less است که با بسیاری از بانک‌های اطلاعاتی موجود سازگاری دارد.
برای مشاهده اطلاعات بیشتری در مورد جزئیات این روش می‌توان به Wiki آن مراجعه کرد.

عنوان: **اجرای یک Script حاوی دستورات Go در سی شارپ**
نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۰۳ ۱۲:۱۰
آدرس: www.dotnettips.info
برچسب‌ها: C#, SQL Server

سلام ؛

سال نو مبارک ! امیدوارم سال بسیار خوبی در پیش داشته باشید :)

از زمانی که استفاده از ORMهای Code First رایج شده ، اجرای اسکریپت‌های طولانی جهت ایجاد دیتابیس خیلی استفاده ندارد، اما حالت خاص همیشه پیش می‌آید.
مثلا قصد داریم پیش از آغاز برنامه پس از ایجاد دیتابیس توسط Entity Framework به یک سری جداول فیلدی با نوع داده‌ی Geometry اضافه کنیم. یا باید به دیتابیس یک سری Stored Procedure و View اضافه کرد.
Scriptها ی Generate شده توسط SQL Server حاوی دستور Go هستند. ADO.NET اجرای Script که حاوی Go باشد را پشتیبانی نمی‌کند.

اما روش‌ها و ترفندهای زیادی برای اجرای یک فایل Script طولانی حاوی دستور Go روی دیتابیس وجود دارد :

```
private static string GetScript()
        {
            string path = AppDomain.CurrentDomain.BaseDirectory +
                          @"Scripts\script.sql";
            var file = new FileInfo(path);

            string script = file.OpenText().ReadToEnd();
            return script;
        }

        private static void ExecuteScript()
        {
            string script = GetScript();

            //split the script on "GO" commands
            var splitter = new[] {"\r\nGO\r\n"};
            string[] commandTexts = script.Split(splitter,
                                                 StringSplitOptions.RemoveEmptyEntries);
            foreach (string commandText in commandTexts)
            {
                using (var ctx = new DbContext())
                {
                    if (!string.IsNullOrEmpty(commandText))
                    {
                        ctx.Database.ExecuteSqlCommand(commandText);
                    }
                }
            }
        }
```

در اینجا به جای آنکه تلاش کنیم یک فایل را روی دیتابیس یک جا اجرا کنیم دستورات را جدا کرده و تک به تک اجرا می‌کنیم.
بروزرسانی: روش بهتر

نظرات خوانندگان

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۱/۰۳ ۱۷:۱۷

ممنون. میشه برای بهبود سرعت insert های زیاد، از SQL Bulk Copy هم استفاده کرد.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۰۳ ۱۹:۲۲

بله درسته،
در سناریویی که برای خودم پیش آمده بود کوئری‌ها فقط Insert نبودند و همچنین فقط قرار بود یک بار در ابتدای برنامه اجرا شوند برای همین روی سرعت حساسیت نداشتم.

نویسنده: مهدی موسوی
تاریخ: ۱۳۹۲/۰۱/۰۳ ۱۹:۵۳

سلام؛ سال نو مبارک :)
آقای کیاست، بهترین روش برای اجرای Script هایی که حاوی کلمه کلیدی GO هستند، استفاده از SMO است. به کد زیر دقت کنید:

```
Server server = new Server(new ServerConnection() { ConnectionString = cnnStr });
server.ConnectionContext.ExecuteNonQuery(sql);
```

اینجا همه چیز توسط SMO کنترل میشه و دیگه نیازی به آنالیز Script اصلی بر اساس عبارت GO نیست.

موفق باشید.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۰۴ ۰:۱۴

سلام قربان ؛
خیلی ممنون.
در استفاده از SMO در .NET 4 مشکل داشتم در نتیجه از این روش که نیازمند به Assemblyهای SMO نیست استفاده کردم.
کامنت شما را در انتهای مطلب قرار دادم.

نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۲/۰۱/۰۵ ۱۷:۱۸

شاهین جان
ممنون از مطلب مفیدت
من این مورد رو قبلا نوشتم. به همین روشی که شما نوشتید. منتها تو یه Query با مشکل مواجه شد و اجرا نشد. واسه همین از روش زیر استفاده کردم که همه جا جواب میده:

```
var commandTexts=Regex.Split(script,@"\s*GO\s+");
foreach(var commandText in commandTexts)
{
    // Execution Code
}
```

نویسنده: مهدی موسوی

تاریخ: ۱۳۹۲/۰۱/۰۶ ۱۵:۳۶

سلام.
خیر، همه جا جواب نمیده! عرض کردم، نیازی به جداسازی Script بر اساس GOهای موجود در اون ندارید. Script زیر رو در نظر بگیرید (بعنوان مثالی نقض) تا متوجه بشید چرا کد شما نیز ایراد داره:

```
USE [MyDB]
GO

SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[GO AND EAT](
[Id] [int] IDENTITY(1,1) NOT NULL,
[MyVal] [decimal](18, 0) NOT NULL,
) ON [PRIMARY]

GO
```

لطفا اینو در نظر بگیرید که منظورم این نیست که با تصحیح Regex Pattern می‌تونید این مشکل رو نیز برطرف کنید، منظورم اینه که با استفاده از SMO می‌تونید از نوشتن کدهای Error-Prone دوری کنید. فقط همین. :)

موفق باشید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۰۶ ۱۷:۱۸

پشت صحنه SMO برای علاقمندان در مورد نحوه جدا سازی batchها از یکدیگر در عمل.

نویسنده: علی
تاریخ: ۱۳۹۲/۰۲/۲۴ ۱۵:۴۷

برای اجرای چنین دستوراتی نیازی به تکه تکه کردن نداریم فقط کافیه go رو از اسکریپت حدف کنیم خودش همرو اجرا می‌کنه قبلا خودم چنین کاری رو انجام دادم

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۲۴ ۱۶:۵۸

کل این بحث در مورد نحوه یافتن صحیح این Go بدون تخریب اطلاعات جانبی بود. حالا می‌خوای حذفش کن یا تکه تکه‌اش کن!

عنوان: آشنایی با Window Function ها در SQL Server بخش پنجم
نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۲/۰۱/۲۹ ۱۸:۵۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, SQL Server 2012, T-SQL

در این بخش فانکشن دیگری از توابع تحلیلی به نام CUME_DIST را بررسی می‌نماییم.

CUME_DIST:

بوسیله تابع CUME_DIST می‌توان ارزیابی نمود، در یک گروه، چه درصد از مقادیر،مساوی یا کوچکتر از مقدار سطر جاری می‌باشند، به این تابع cumulative distribution نیز گفته می‌شود.

Syntax تابع CUME_DIST به صورت زیر است:

```
CUME_DIST( )
    OVER ( [ partition_by_clause ] order_by_clause )
```

شرح Syntax:

1- Partition By Clause : بوسیله پارامتر فوق می‌توانید، نتیجه پرس جو (Query)،خود را دسته بندی نمایید.

2- order by clause : همانطور که از نامش مشخص است، جهت مرتب نمودن خروجی Query می‌باشد.

معمولا شرح عملکرد توابع تحلیلی، کمی مشکل است. بنابراین برای درک، عملکرد تابع CUME_DIST چند مثال را بررسی می‌کنیم.

در ابتدا بوسیله Script زیر یک جدول ایجاد و 10 رکورد در آن درج می‌کنیم:

```
Create Table TestCUME_DIST
(SalesOrderID int not null,
OrderQty smallint not null,
ProductID int not null
 );
 GO
Insert Into TestCUME_DIST
       Values (43663,1,760),(43667,3,710),(43667,1,773),
  (43667,1,775),(43667,1,778),(43669,1,747),
  (43670,1,709),(43670,2,710),(43670,2,773),(43670,1,776)
```

مثال اول: Script زیر را اجرا می‌کنیم، سپس خروجی آن را بررسی می‌نماییم:

```
SELECT SalesOrderID, OrderQty,
       CUME_DIST() OVER(ORDER BY SalesOrderID) AS [CUME_DIST]
FROM TestCUME_DIST ORDER BY [CUME_DIST] DESC
```

پس از اجرا خروجی بصورت زیر خواهد بود:

	SalesOrderID	OrderQty	CUME_DIST
1	43670	1	1
2	43670	2	1
3	43670	2	1
4	43670	1	1
5	43669	1	0.6
6	43667	3	0.5
7	43667	1	0.5
8	43667	1	0.5
9	43667	1	0.5
10	43663	1	0.1

در ادامه اجازه دهید،مقادیری که در فیلد CUME_DIST بدست آمده است را بصورت تصویری بررسی کنیم.
مقادیر سطر اول تا چهارم:

	SalesOrderID	OrderQty	CUME_DIST
1	43670	1	1
2	43670	2	1
3	43670	2	1
4	43670	1	1
5	43669	1	0.6
6	43667	3	0.5
7	43667	1	0.5
8	43667	1	0.5
9	43667	1	0.5
10	43663	1	0.1

سوال اول:چند سطر، SalesOrderID آن برابر ۴۳۶۷۰ می باشد؟
جواب: چهار سطر(Row)، فرض می کنیم C1=4

سوال دوم: چند سطر (Row)، زیر SalesOrderID=43670 قرار دارد؟
جواب: شش سطر (Row)، فرض می کنیم C2=6

سوال سوم: تعداد کل سطرها (Rows)، چند عدد می باشد؟
جواب: ۱۰ سطر (Rows) ، فرض می کنیم C3=10

برای بدست آوردن CUME_DIST چهار سطر اول از فرمول زیر پیروی می نماییم:
Rows= (C1+C2)/C3= (4+6)/10 =1

*** برای بدست آوردن CUME_DIST سطر پنجم نیز خواهیم داشت:

Rows=(c1+c2)/c3 بنابراین خواهیم داشت: Rows=(1+5)/10=6/0

مثال دوم : ابتدا Script زیر را اجرا نمایید:

```
SELECT SalesOrderID, OrderQty, ProductID,
       CUME_DIST() OVER(PARTITION BY SalesOrderID ORDER BY ProductID ) AS [CUME_DIST]
FROM TestCUME_DIST
WHERE SalesOrderID IN (43670, 43669, 43667, 43663)
ORDER BY SalesOrderID DESC, [CUME_DIST] DESC
```

خروجی :

	SalesOrderID	OrderQty	ProductID	CUME_DIST
1	43670	1	776	1
2	43670	2	773	0.75
3	43670	2	710	0.5
4	43670	1	709	0.25
5	43669	1	747	1
6	43667	1	778	1
7	43667	1	775	0.75
8	43667	1	773	0.5
9	43667	3	710	0.25
10	43663	1	760	1

همانگونه که ملاحظه می‌کنید، در این مثال، خروجی، براساس SalesOrderID به چهار گروه تقسیم می‌شود و عملیات مرتب سازی روی فیلد ProductID انجام می گیرد، بنابراین CUME_DIST، روی هر گروه بر روی فیلد ProductID محاسبه می‌شود.

گروه اول : نحوه محاسبه Cume_DIST سطر اول:

سوال:چه تعداد از مقادیر ProductID آن برابر 776 میباشد؟

جواب: فقط مقدار سطر اول، بنابراین خواهیم داشت C1=1

سوال: چه تعداد از مقادیر کوچکتر از ProductID=776 می‌باشد؟

جواب: مقدار سه سطر، در واقع مقادیر سطر دوم،سوم و چهارم کوچکتر از مقدار سطر اول می‌باشند، c2=3

سوال: تعداد کل سطرهای گروه اول چه مقدار می‌باشد؟

جواب: 4سطر

بنابراین برای بدست آوردن CUME_DIST سطر اول خواهیم داشت:

Rows=(1+3)/4=1

محاسبه سطر دوم از گروه اول بدون شرح:

Rows=(1+2)/4=0/75

امیدوارم مفید واقع شده باشد.

عنوان: **افزودن یک DataType جدید برای نگه‌داری تاریخ خورشیدی - 1**
نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۰۹ ۲۳:۳۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SQL Server 2012, C#.NET, Persian, CLR

ثبت و نگه‌داری تاریخ خورشیدی در SQL Server از دیرباز یکی از نگرانی‌های برنامه‌نویسان و طراحان پایگاه داده‌ها بوده است. در این نوشتار، راه‌کار تعریف یک DataType در SQL Server 2012 به روش CLR آموزش داده خواهد شد.

در ویژوال استودیو یک پروژه‌ی جدید از نوع SQL Server Database Project به شکل زیر ایجاد کنید:

نام پروژه را به یاد تقویم خیام، prgJalaliDate می‌گذارم. در Solution Explorer روی نام پروژه راست‌کلیک کرده، سپس روی Add New Item کلیک کنید. در پنجره‌ی بازشده مطابق شکل SQL CLR C# User Defined Type را برگزینید؛ سپس نام JalaliDateType را برای آن انتخاب کنید.

متن موجود در صفحه‌ی بازشده را کاملاً حذف کرده و با کد زیر جای‌گزین کنید.

(در کد زیر همه‌ی توابع لازم برای مقداردهی به سال، ماه، روز، ساعت، دقیقه و ثانیه و البته گرفتن مقدار از آن‌ها، تبدیل تاریخ خورشیدی به میلادی، گرفتن تاریخ به تنهایی، گرفتن زمان به تنهایی، افزایش یا کاهش زمان برپایه‌ی یکی از متغیرهای زمان و بررسی و اعتبارسنجی انواع بخش‌های زمان گنجانده شده است. در صورت پرسش یا پیشنهاد روی هر کدام در قسمت نظرات، پیام خود را بنویسید.)

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

[Serializable()]
[SqlUserDefinedType(Format.Native)]
public struct JalaliDate : INullable
{
    private Int16 m_Year;
    private byte m_Month;
    private byte m_Day;
    private byte m_Hour;
    private byte m_Minute;
    private byte m_Second;
    private bool is_Null;


    public Int16 Year
    {
        get
        {
            return (this.m_Year);
        }
        set
        {
            m_Year = value;
        }
    }

    public byte Month
    {
        get
        {
            return (this.m_Month);
```

```
        }
        set
        {
            m_Month = value;
        }
    }

    public byte Day
    {
        get
        {
            return (this.m_Day);
        }
        set
        {
            m_Day = value;
        }
    }

    public byte Hour
    {
        get
        {
            return (this.m_Hour);
        }
        set
        {
            m_Hour = value;
        }
    }

    public byte Minute
    {
        get
        {
            return (this.m_Minute);
        }
        set
        {
            m_Minute = value;
        }
    }

    public byte Second
    {
        get
        {
            return (this.m_Second);
        }
        set
        {
            m_Second = value;
        }
    }

    public bool IsNull
    {
        get
        {
            return is_Null;
        }
    }

    public static JalaliDate Null
    {
        get
        {
            JalaliDate jl = new JalaliDate();
            jl.is_Null = true;
            return (jl);
        }
    }


    public override string ToString()
    {
        if (this.IsNull)
        {
            return "NULL";
        }
        else
```

```
        {
            return this.m_Year.ToString("D4") + "/" + this.m_Month.ToString("D2") + "/" +
this.m_Day.ToString("D2") + " " + this.Hour.ToString("D2") + ":" + this.Minute.ToString("D2") + ":" +
this.Second.ToString("D2");
        }
    }


    public static JalaliDate Parse(SqlString s)
    {
        if (s.IsNull)
        {
            return Null;
        }

        System.Globalization.PersianCalendar pers = new System.Globalization.PersianCalendar();
        string str = Convert.ToString(s);
        string[] JDate = str.Split(' ')[0].Split('/');

        JalaliDate jl = new JalaliDate();

        jl.Year = Convert.ToInt16(JDate[0]);
        byte MonthsInYear = (byte)pers.GetMonthsInYear(jl.Year);
        jl.Month = (byte.Parse(JDate[1]) <= MonthsInYear ? (byte.Parse(JDate[1]) > 0 ?
byte.Parse(JDate[1]) : (byte)1) : MonthsInYear);
        byte DaysInMonth = (byte)pers.GetDaysInMonth(jl.Year, jl.Month); ;
        jl.Day = (byte.Parse(JDate[2]) <= DaysInMonth ? (byte.Parse(JDate[2]) > 0 ?
byte.Parse(JDate[2]) : (byte)1) : DaysInMonth);
        if (str.Split(' ').Length > 1)
        {
            string[] JTime = str.Split(' ')[1].Split(':');
            jl.Hour = (JTime.Length >= 1 ? (byte.Parse(JTime[0]) < 23 && byte.Parse(JTime[0]) >=
(byte)0 ? byte.Parse(JTime[0]) : (byte)0) : (byte)0);
            jl.Minute = (JTime.Length >= 2 ? (byte.Parse(JTime[1]) < 59 && byte.Parse(JTime[1]) >=
(byte)0 ? byte.Parse(JTime[1]) : (byte)0) : (byte)0);
            jl.Second = (JTime.Length >= 3 ? (byte.Parse(JTime[2]) < 59 && byte.Parse(JTime[2]) >=
(byte)0 ? byte.Parse(JTime[2]) : (byte)0) : (byte)0);
        }
        else { jl.Hour = 0; jl.Minute = 0; jl.Second = 0; }

        return (jl);
    }

    public SqlString GetDate()
    {
        return this.m_Year.ToString("D4") + "/" + this.m_Month.ToString("D2") + "/" +
this.m_Day.ToString("D2");
    }

    public SqlString GetTime()
    {
        return this.Hour.ToString("D2") + ":" + this.Minute.ToString("D2") + ":" +
this.Second.ToString("D2");
    }

    public SqlDateTime ToGregorianTime()
    {
        System.Globalization.PersianCalendar pers = new System.Globalization.PersianCalendar();
        return SqlDateTime.Parse(pers.ToDateTime(this.Year, this.Month, this.Day, this.Hour,
this.Minute, this.Second, 0).ToString());
    }

    public SqlString JalaliDateAdd(SqlString interval, int increment)
    {
        System.Globalization.PersianCalendar pers = new System.Globalization.PersianCalendar();
        DateTime dt = pers.ToDateTime(this.Year, this.Month, this.Day, this.Hour, this.Minute,
this.Second, 0);
        string CInterval = interval.ToString();
        bool isConvert = true;
        switch (CInterval)
        {
            case "Year":
                dt = pers.AddYears(dt, increment);
                break;
            case "Month":
                dt = pers.AddMonths(dt, increment);
                break;
            case "Day":
                dt = pers.AddDays(dt, increment);
                break;
            case "Hour":
```

```
                dt = pers.AddHours(dt, increment);
                break;
            case "Minute":
                dt = pers.AddMinutes(dt, increment);
                break;
            case "Second":
                dt = pers.AddSeconds(dt, increment);
                break;
            default:
                isConvert = false;
                break;
        }

        if (isConvert == true)
        {
            this.Year = (Int16)pers.GetYear(dt);
            this.Month = (byte)pers.GetMonth(dt);
            this.Day = (byte)pers.GetDayOfMonth(dt);
            this.Hour = (byte)pers.GetHour(dt);
            this.Minute = (byte)pers.GetMinute(dt);
            this.Second = (byte)pers.GetSecond(dt);
        }


        return this.m_Year.ToString("D4") + "/" + this.m_Month.ToString("D2") + "/" +
this.m_Day.ToString("D2") + " " + this.Hour.ToString("D2") + ":" + this.Minute.ToString("D2") + ":" +
this.Second.ToString("D2");
    }
}
```

از منوهای بالا روی منوی Bulild و سپس گزینه‌ی Publish prgJalaliDate کلیک کنید:

در پنجره‌ی بازشده روی دکمه‌ی Edit کلیک کنید سپس تنظیمات مربوط به اتصال به پایگاه داده را انجام دهید.

روی دکمه‌ی OK کلیک کنید و سپس در پنجره‌ی اولیه، روی دکمه‌ی Publish کلیک کنید:

به همین سادگی، DataType مربوطه در SQL Server 2012 ساخته می‌شود. خبر خوش این‌که شما می‌توانید با راست‌کلیک روی نام پروژه و انتخاب گزینه‌ی Properties در قسمت Project Setting تنظیمات مربوط به نگارش SQL Server را انجام دهید. (از نگارش 2005 به بعد در VS 2012 پشتیبانی می‌شود.)

اکنون زمان آن رسیده است که DataType ایجادشده را در SQL Server 2012 بیازماییم. SQL Server را باز کنید و دستور زیر را در آن اجرا کنید.

```
USE Northwind

GO

CREATE TABLE dbo.TestTable
(
Id int NOT NULL IDENTITY (1, 1),
TestDate dbo.JalaliDate NULL
)  ON [PRIMARY]
GO
```

همین‌طور که مشاهده می‌کنید؛ امکان به‌کارگیری DataType تعریف‌شده وجود دارد.
اکنون چند رکورد درون این جدول درج می‌کنیم:

```
Insert into TestTable (TestDate) Values ('1392/02/09'),('1392/02/09 22:40'),('1392/12/30 22:40')
```

پس از اجرای این دستور خطای زیر در پایین صفحه‌ی SQL Server نمایان می‌شود:

```
Messages
Msg 6263, Level 16, State 1, Line 1
Execution of user code in the .NET Framework is disabled. Enable "clr enabled" configuration option.
```

این خطا به این خاطر است که CLR را در SQL Server فعال نکرده ایم. جهت فعال‌کردن CLR دستور زیر را اجرا کنید:

```
sp_configure 'clr enabled', 1
Reconfigure
```

بار دیگر دستور درج را اجرا می‌کنیم:

```
Insert into TestTable (TestDate) Values ('1392/02/09'),('1392/02/09 22:40'),('1392/12/30 22:40')
```

	Id	TestDate
▶	1	1392/02/09 00:00:00
	2	1392/02/09 22:40:00
	3	1392/12/29 22:40:00
*	NULL	NULL

ملاحظه می‌کنید که داده‌ها در جدول مربوطه ذخیره شده است. در رکورد نخست چون ساعت، دقیقه و ثانیه تعریف نشده است؛ به طور هوشمند صفر درج شده است. در رکورد دوم، ساعت و دقیقه مقدار دارد ولی ثانیه صفر ثبت شده است. و در رکورد سوم چون سال 1392 کبیسه نیست؛ به صورت هوشمند آخرین روز ماه به جای روز ثبت شده است. هرچند می‌توان با دست‌کاری در توابع سی‌شارپ، این قوانین را عوض کرد.

اکنون زمان آن رسیده است که توسط یک پرس‌وجو، همه‌ی توابعی که در سی‌شارپ برای این نوع داده نوشتیم، بیازماییم.
پرس‌وجوی زیر را اجرا کنید:

```
Select TestDate.ToString() as JalaliDateTime,
          TestDate.GetDate() as JalaliDate, TestDate.GetTime() as JalaliTime,
          TestDate.ToGregorianTime() as GregorianTime,
          TestDate.JalaliDateAdd('Day',1) JalaliTomorrow,
          TestDate.Month as JalaliMonth from TestTable
```

خروجی این پرس‌وجو به شکل زیر خواهد بود:

	JalaliDateTime	JalaliDate	JalaliTime	GregorianTime	JalaliTomorrow	JalaliMonth
1	1392/02/09 00:00:00	1392/02/09	00:00:00	2013-04-29 00:00:00.000	1392/02/10 00:00:00	2
2	1392/02/09 22:40:00	1392/02/09	22:40:00	2013-04-29 22:40:00.000	1392/02/10 22:40:00	2
3	1392/12/29 22:40:00	1392/12/29	22:40:00	2014-03-20 22:40:00.000	1393/01/01 22:40:00	12

البته درباره‌ی ستون پنجم و ششم شما می‌توانید روی همه‌ی اجزای تاریخ افزایش و کاهش داشته باشید و هم‌چنین می‌توانید با تابع مربوطه هر کدام از اجزای زمان را جداگانه به دست بیاورید که در این مثال عدد ماه نشان داده شده است.

نیازی به گفتن نیست که می‌توانید به سادگی از توابع مربوط به DateTime در SQL Server بهره ببرید. برای مثال برای به دست

آوردن فاصله‌ی میان دو روز از پرس‌وجوی زیر استفاده کنید:

```
Declare @a JalaliDate  = '1392/02/07 00:00:00'
Declare @b JalaliDate = '1392/02/05 00:00:00'

SELECT DATEDIFF("DAY",@b.ToGregorianTime(),@a.ToGregorianTime()) AS DiffDate
```

شاد و پیروز باشید.

نظرات خوانندگان

نویسنده: ali
تاریخ: ۱۳۹۲/۰۲/۱۰ ۰:۵

Nice article

Thanks

نویسنده: سام ناصری
تاریخ: ۱۳۹۲/۰۲/۱۰ ۴:۲۲

بسیار خوب.
فقط من تو مقاله شما دلیلی برای اینکه چرا زمان و تاریخ را میخواهی به این صورت ذخیره کنی متوجه نشدم؟ چرا به همان شکل استانداردش ذخیره نکنیم؟
دیگر اینکه نوع داده جدید به چه شکل در دیتابیس ذخیره میشود. Sql Server از کجا میداند که باید چگونه لیترالها را پارس کند(چگونه متود Parse برگزیده میشود)؟ عملگرهای مقایسه چگونه کار خواهند کرد؟

نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۱۰ ۷:۸

با درود
به پرسش‌های شما در بخش دوم پاسخ خواهم داد.

نویسنده: قاسم
تاریخ: ۱۳۹۲/۰۲/۱۰ ۹:۴۲

سلام، میشه مراحل انجام کار توی VS2010 هم بنویسید، فقط مراحل کار. خیلی ممنون میشم

نویسنده: فرشید علی اکبری
تاریخ: ۱۳۹۲/۰۲/۱۰ ۹:۵۵

سلام
کدهای شمارو درست همونطوریکه گفتین کپی کردم ولی موقعی که Publish رو میزنم پیغام زیر رو میده :
Publish cannot begin until your project(s) build successfully
و پنجره publish ظاهر نمی‌شه و هیچ جایی هم برای تعریف کانکشن وجود نداره... مشکل از کجاست؟
دوم اینکه clr رو هربار که خواستیم روی یک دیتابیس جدید این نوع رو تعریف کنیم باید فعال کنیم... مثلا توی ایجاد سال مالی جدید که سیستم یک دیتابیس خام ایجاد میکنه تا مانده حسابها رو بهش انتقال بدیم قبلش باید دستور فعال سازی clr رو هم مجددا بدیم یافقط توی زمان طراحی کفایت میکنه ؟
سوم اینکه : در مورد استفاده اون توی EF Code First هم اگه نکته‌ی خاصی وجود داره محبت کنین ممنون میشم.
شاد وپیروز باشید.

نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۱۰ ۱۱:۴

با درود
New Project -> Database -> SQL Server -> Visual C# SQL CLR Database Project
- تنظیمات اتصال به پایگاه داده ها
- انتخاب دکمه Yes

- Add New Item -> User Defined Type
- کپی کدها
- استفاده از Deploy در منوی Build یا استفاده از روشی که در بخش دوم نوشتار آموزش داده خواهد شد.

نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۱۰ ۱۱:۲۶

با درود
1- لطفاً پروژه را پیش از Publish یک بار Rebuild کنید. احتمالاً به خاطر یک ارور خاص Publish نمی‌شود. اگر به راحتی Rebuild شده ولی باز هم Publish کار نمی‌کند؛ می‌توانیم با روش دیگری که در بخش دوم آموزش می‌دهم DLL را به SQL معرفی کنیم.
2- کافی است یک بار پس ساخت Database این پرس‌وجو را اجرا کنید.
3- تست نکردم ولی به نظر می‌رسد این نوع داده از سمت EF شبیه به Hierarchy باشد.

نویسنده: ش.د
تاریخ: ۱۳۹۲/۰۲/۱۰ ۱۴:۳۵

آیا فقط در sql2012 قابل اجرا می‌باشد؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۰ ۱۴:۵۴

در متن نوشتن از نگارش 2005 به بعد اس کیوال سرور این قابلیت استفاده از افزونه‌های CLR اضافه شدن.

نویسنده: سید امیر سجادی
تاریخ: ۱۳۹۲/۰۲/۱۰ ۱۸:۳۸

با تشکر از مطلب مفیدتون.
چند تا سوال برام پیش اومده.
اول اینکه آیا به صورت یک DLL به بانک اضافه میشه؟
دوم اینکه اگه از بانک بک آپ بگیریم و جایی دیگه خواستیم اون رو ریستور کنیم چی میشه؟
آپلود بانک روی هاست (بک آپ یا اتچ) ؟

نویسنده: امیر بختیاری
تاریخ: ۱۳۹۲/۰۲/۱۱ ۹:۴۳

به صورت اسمبلی به بانک اضافه میشه
وقتی بک آپ بگیرید و ریستور کنید همراه بک آپ این اسمبلی هم انقال داده می‌شود
در اتچ هم به همین شکل
فقط زمانی که از این اسمبلی در توابع و پروسیجرها استفاده می‌کنید نمی‌تونید حذفش کنید و در صورت تغیرات باید اسمبلی را به روز کنید.
یک مورد دیگه که من زیاد تو هاست‌های شیر شده بهش برخوردم اینه که این امکان توشون فعال نیست و ادمین سرورها هم به سختی این امکان را فعال می‌کنند پس اگر خواستید از این امکان استفاده کنید ابتدا از فعال بودن آن مطمئن شوید

نویسنده: rahim
تاریخ: ۱۳۹۳/۰۵/۱۱ ۱۲:۱۴

با تشکر از مطلب مفیدتون
من زمانی که از این روش استفاده میکنم در هنگام درج رکورد جدید با پیغام خطای زیر مواجه میشم.
:"A .NET Framework error occurred during execution of user-defined routine or aggregate "JalaliDate

.System.OverflowException: Value was either too large or too small for an unsigned byte

عنوان: افزودن یک DataType جدید برای نگه‌داری تاریخ خورشیدی - 2
نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۱۰ ۲۱:۱۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, SQL Server 2012, Persian, CLR, SQL

پیش از هرچیز به شما پیش‌نهاد می‌کنم؛ بار دیگر کد سی‌شارپ درس نخست را در پروژه‌ی خود کپی کنید و سپس Publish را بزنید. پس از ارسال آن مطلب، تغییراتی در جهت بهینه‌سازی کد دادم که به نظرم بهتر است شما نیز در پروژه‌ی خود به کار برید.

چرا از این نوع داده استفاده کنیم؟

نخستین پرسشی که ممکن است برای شما پیش بیاید این است که چرا بهتر است از این نوع داده استفاده کنیم. برای پاسخ به این پرسش باید راه‌کارهای گذشته را بررسی کنیم. معمولاً طراحان پایگاه داده‌ها برای استفاده از تاریخ خورشیدی، زمان را به صورت میلادی ثبت می‌کنند؛ سپس با یک scalar-valued function زمان درج شده را به خورشیدی تبدیل می‌کنند. در این صورت می‌توان با یک تابع کوچک دیگر بخش مربوط به ساعت را نیز از همان ستون به دست آورد. در این صورت می‌توانیم از کلیه‌ی متدهای مربوط به DateTime در SQL از جمله افزایش و کاهش و تفاضل دو تاریخ بهره برد. برخی دیگر از طراحان، ستونی از نوع char(10) در نظر می‌گیرند و تاریخ خورشیدی را به صورت ده‌کاراکتری در آن ذخیره می‌کنند. این روش هرچند نیاز به تبدیل به خورشیدی را ندارد ولی کلیه‌ی مزایایی که در استفاده از DateTime به آن‌ها دسترسی داریم از دست می‌دهیم. افزون بر این جهت نگه‌داری زمان باید یک فیلد دیگر از نوع کاراکتری و یا در نگارش‌های نوین‌تر از نوع time تعریف کنیم. برخی دیگر از هر دو را در کنار هم استفاده می‌کنند و در واقع جهت سرعت بالاتر نمایش و بررسی داده‌ها از طریق محیط SQL Server از فیلد کاراکتری تاریخ خورشیدی و برای مقایسه و بدست آوردن ساعت از فیلد نوع DateTime استفاده می‌کنند.

از نظر فضای اشغال‌شده نوع DataTime، هشت بایت، smalldatetime (در صورت استفاده) 4 بایت و فیلد 10 کاراکتری تاریخ 10 بایت فضا اشغال می‌کند در صورتی که نوع JalaliDate با درنظر گرفتن همه‌ی مزایای انواع داده‌ی استفاده‌شده برای تاریخ، فقط 8 بایت فضا اشغال می‌کند. با استفاده از این نوع به راحتی داده‌ی تاریخ را بر اساس تقویم ایرانی اعتبارسنجی می‌کنید و بخش‌های مختلف زمان از سال تا ثانیه را با یک متد به دست می‌آورید. می‌توانید به راحتی به تاریخ خود زمانی را بیفزایید یا بکاهید و در گزارش‌ها بدون نگرانی از تبدیل درست استفاده کنید. چون کدباز است می‌توانید با کمی حوصله امکانات دیگر مد نظر خود را به آن بیفزایید و از آن در SQL بهره ببرید.

چگونه این نوع داده را حذف کنم!؟

شما می‌توانید به سادگی نوع داده‌ی ایجادشده توسط CLR را در مسیر زیر بیابید و اقدام به حذف آن نمایید:

همان‌طور که مشاهده می‌شود؛ حتی نوع داده‌ی سیستمی hierarchyid که جهت ساختار سلسله‌مراتبی مانند چارت سازمانی یا درخت تجهیزات استفاده می‌شود؛ نیز یک نوع داده‌ی CLR است.

آیا راه دیگری نیز برای افزودن این نوع داده به SQL به جز Publish کردن وجود دارد؟

مانند بسیاری دیگر از گونه‌های پروژه، در اینجا نیز شما یک فایل DLL خواهید داشت. این فایل برپایه‌ی تنظیماتی که شما در قسمت Properties پروژه‌ی خود انجام می‌دهید ساخته می‌شود. پس از تغییر مسیر فایل DLL در دستور زیر توسط یک New Query از Database خود، آن را اجرا کنید:

```
CREATE ASSEMBLY JalaliDate
FROM 'F:\prgJalaliDate.dll'
WITH PERMISSION_SET = SAFE;
```

هم‌چنین در صورت ویرایش‌های دوباره پروژه از دستور زیر استفاده کنید:

```
ALTER ASSEMBLY JalaliDate
FROM 'F:\prgJalaliDate.dll'
```

با استفاده از دستورهای زیر می‌توانید از چگونگی درج فایل‌های افزوده شده آگاه شوید:

```
select * from sys.assemblies
select * from sys.assembly_files
```

Results | Messages

	name	principal_id	assembly_id	clr_name	permission...	permission_set_desc	is_visible	create_date	modify_date	is_user_defined
1	Microsoft.SqlServer.Types	4	1	microsoft.sqlserver.types, ver...	3	UNSAFE_ACCESS	1	2012-02-10 20:...	2012-02-10 ...	0
2	prgJalaliDate	1	65536	prgjalalidate, version=0.0.0.0,...	1	SAFE_ACCESS	1	2013-04-29 22:...	2013-04-29 ...	1

	assembly_id	name	file_id	content
1	1	microsoft.sqlserver.types.dll	1	0x4D5A900003000000040000 00FFFF0000B80000000000000...
2	65536	prgJalaliDate	1	0x4D5A900003000000040000 00FFFF0000B80000000000000...
3	65536	prgJalaliDate.pdb	2	0x4D6963726F736F667420432F432B2B204D534620372E30...

تا اینجا SQL Server، دی‌ال‌ال مربوط به پروژه را شناخته است. برای تعریف نوع داده از دستور زیر بهره ببرید:

```
CREATE TYPE dbo.JalaliDate
EXTERNAL NAME JalaliDate.[JalaliDate];
```

این کار همانند استفاده از گزینه‌ی Publish در Visual Studio است.

هم‌چنین چنان‌چه در SQL Server 2012 از منوی راست‌کلیک پایگاه داده‌ها روی گزینه Tasks و سپس Generate Scripts را انتخاب کنیم، از مشاهده‌ی سند ساخته شده، درخواهیم یافت که حتی دستورهای مربوط به ساخت اسمبلی CLR با تبدیل فایل به کد در Scripts وجود دارد و با اجرای آن در سروری دیگر، انتقال می‌یابد.

```
GO
/****** Object:  SqlAssembly [prgJalaliDate]    Script Date: 2013/04/30 08:27:00 ب.ظ ******/
CREATE ASSEMBLY [prgJalaliDate]
FROM 0x4D5A900003000000040000 00FFFF0000B8000000000000 ..... بقیه‌ی کدها حذف شده
WITH PERMISSION_SET = SAFE

GO
ALTER ASSEMBLY [prgJalaliDate]
ADD FILE FROM 0x4D6963726F736F667420432F432B2B204D534620372E30300D0A1A44530..... بقیه‌ی کدها حذف شده
AS N'prgJalaliDate.pdb'

GO
/****** Object:  UserDefinedType [dbo].[JalaliDate]    Script Date: 2013/04/30 08:27:00 ب.ظ ******/
CREATE TYPE [dbo].[JalaliDate]
EXTERNAL NAME [prgJalaliDate].[JalaliDate]

GO
```

دنباله دارد ...

نظرات خوانندگان

نویسنده: محمد عادل
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱:۴۳

در این حالت ، چطور میتونیم در EF Code First از این DataType استفاده کنیم ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۱ ۸:۳۴

از برنامه Entity Framework Power Tools برای مهندسی معکوس ساختار موجود استفاده کنید و بعد مشاهده کنید که چه کدی رو تولید می‌کنه. کار شما در این حالت code first نیست.

نویسنده: حامد حسین نژاد
تاریخ: ۱۳۹۲/۰۲/۱۱ ۹:۲۰

البته باید این را هم در نظر گرفت که کلا CLR Integration باعث کندی دیتابیس می‌شود و انواع داده CLR بسیار کندتر از بقیه اجرا می‌شوند. این موضوع ممکن است که در دیتابیس‌های کوچک چندان مهم بنظر نیاید ولی در دیتابیس‌های بزرگ باعث بروز مشکل خواهد شد. علاوه بر این استفاده از انواع داده CLR و یا توابع CLR در دیتابیس باعث می‌شود که امکان پارتیشن بندی جداول آن دیتابیس وجود نداشته باشد.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۱:۳۴

مقاله‌ای هست اینجا در مورد کارآیی CLR در SQL Server. به نظر می‌رسه سریعتر است حدود 11 درصد نسبت به T-SQL معمولی. برای پارتشین بندی می‌تونید اینکار رو انجام بدید فقط این نوع خاص قابل انتخاب نیست. مابقی فیلدها رو می‌تونید انتخاب کنید.

نویسنده: حامد حسین نژاد
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۸:۳۱

البته کارایی CLR، بسته به مورد استفاده، متفاوته (این لینک). در مواردی مثل همین مثال، اگه تعداد سطور جدول زیاد باشه، کارایی رو به شدت کاهش میده. مخصوصا اگه بخواین از Data Warehousing هم استفاده کنید.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۹:۲۸

این به شدت رو می‌تونید در موردش عدد و رقم ارائه بدید؟ در همون لینکی که دادید عنوان شده پیاده سازی RegEx روی سطور بالا خیلی سریع‌تر هست با CLR. در هر حال بهتره با عدد و رقم و محاسبات بحث کرد.

نویسنده: حامد حسین نژاد
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۴:۲۸

مثلا فرض کنید جدولی دارید که اطلاعات Taskهای یک شرکت رو نگه میداره که شامل تاریخ شروع و تاریخ پایان هر Task هم هست. اگه این جدول یک میلیون سطر داده داشته باشه و بخوایم Task هایی که مدت زمان انجام اونها کمتر از 5 روز بوده رو انتخاب کنیم تفاوت فاحشی با datetime خواهیم داشت.
البته، همونطور که گفتم بسته به نوع استفاده داره و ممکنه از دیتابیسی به دیتابیس دیگه فرق بکنه.

عنوان: **افزودن یک DataType جدید برای نگه‌داری تاریخ خورشیدی - 3**
نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۱۳ ۱۳:۲۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, SQL Server 2012, Persian, CLR

در دو درس گذشته با DataType ای که با هم نوشتیم؛ می‌توانستیم این پرس‌وجوها را داشته باشیم:

```
Declare @MyDate JalaliDate = '1392/02/11 21:38:24'
Select @MyDate.ToString() as MyDateTime
           , @MyDate.GetDate() as MyDate
           , @MyDate.GetTime() as MyTime
           , @MyDate.Year as MyYear
           , @MyDate.Month as MyMonth
          , @MyDate.Day as MyDay
          , @MyDate.Hour as MyHour
          , @MyDate.Minute as MyMinute
          , @MyDate.Second as MySecond
          , @MyDate.JalaliDateAdd('Year',1) as NextYear
          , @MyDate.JalaliDateAdd('Month',1) as NextMonth
          , @MyDate.JalaliDateAdd('Day',1) as NextDay
          , @MyDate.JalaliDateAdd('Hour',1) as NextHour
          , @MyDate.JalaliDateAdd('Minute',1) as NextMinute
          , @MyDate.JalaliDateAdd('Second',1) as NextSecond
```

که در نهایت خروجی کار به شکل زیر خواهد بود:

Results | Messages

	MyDateTime	MyDate	MyTime	MyYear	MyMonth	MyDay	MyHour	MyMinute	MySecond
1	1392/02/11 21:38:24	1392/02/11	21:38:24	1392	2	11	21	38	24

NextYear	NextMonth	NextDay	NextHour	NextMinute	NextSecond
1393/02/11 21:38:24	1392/03/11 21:38:24	1392/02/12 21:38:24	1392/02/11 22:38:24	1392/02/11 21:39:24	1392/02/11 21:38:25

با این همه، همان‌سان که در پرس‌وجوی بالا هم می‌بینید؛ ناگزیر شدم تاریخ و زمان را خودم وارد کنم و هرآینه در واقعیت شما جهت پرس‌وجو روی زمان کنونی، ناگزیر به استفاده از یک تابع برای تبدیل تقویم میلادی به خورشیدی هستیم. به نظر شما بهتر نیست دست به کار شویم و تابعی برای تبدیل تاریخ میلادی به خورشیدی بنویسیم؟

برای این‌کار پروژه‌ای را که در دو درس 1 ساختیم باز کنید و سپس روی نام پروژه در Solution Explorer راست‌کلیک کرده و Add New Item را انتخاب کنید.

محتویات فایل بازشده را حذف کنید و دستورهای زیر را جای‌گزین کنید:

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

public partial class UserDefinedFunctions
{
    [Microsoft.SqlServer.Server.SqlFunction]
    public static JalaliDate GetCurrentDateTime()
    {
        System.Globalization.PersianCalendar pers = new System.Globalization.PersianCalendar();
        DateTime CurrentDate = DateTime.Now;
        JalaliDate jl;
        jl.Year = (Int16)pers.GetYear(CurrentDate);
        jl.Month = (byte)pers.GetMonth(CurrentDate);
        jl.Day = (byte)pers.GetDayOfMonth(CurrentDate);
        jl.Hour = (byte)pers.GetHour(CurrentDate);
        jl.Minute = (byte)pers.GetMinute(CurrentDate);
        jl.Second = (byte)pers.GetSecond(CurrentDate);
        return jl;
    }
}
```

بار دیگر پروژه را به روشی که در درس یک آموزش داده شد؛ Publish کنید. اکنون می‌توانیم تاریخ و زمان کنونی را در متغیری از نوع JalaliDate وارد کنیم.

```
Declare @MyDate JalaliDate = dbo.GetCurrentDateTime()
```

یا به این شکل تاریخ و زمان جاری را مشاهده کنیم:

```
Select dbo.GetCurrentDateTime().ToString()
```

دنباله دارد...

نظرات خوانندگان

نویسنده: فرشید علی اکبری
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۰:۳۴

از دوستان اگه کسی نحوه استفاده این تکنیک رو توی EF Code First میدونه لطفا مطلب رو Share کنه تا امکان استفاده این دسته از عزیزان هم از زحمتی که آقای قنادی کشیدن باشه.
با تشکر.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۰:۴۲

```
public DateTime AddDate { set; get; }

[NotMapped]  // فیلد محاسباتی سمت برنامه که در بانک اطلاعاتی قرار نخواهد گرفت
public string PersianDate
{
    get { return MyDateConverter(AddDate); }
}
```

در code first از ویژگی NotMapped استفاده کنید تا بتونید با استفاده از تابع کمکی تبدیل تاریخ خودتون راحت این تبدیلات رو انجام بدید. در بانک اطلاعاتی DateTime استاندارد رو ذخیره کنید، در سمت کدها برای مسایل نمایشی از خاصیت PersianDate که NotMapped تعریف شده، میشه استفاده کرد. به علاوه روی DateTime استاندارد راحت میشه کوئری‌های LINQ رو اجرا کرد بدون محدودیت. زمانیکه مثلا یک List تهیه شد، قسمت بعدی، کار نمایشی است که از خاصیت NotMapped میشه کمک گرفت. این روش با تمام بانک‌های اطلاعاتی کار می‌کنه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۱:۴۱

```
[Column(TypeName = "xml")]
public string XmlValue { get; set; }

[NotMapped]
public XElement XmlValueWrapper
{
    get { return XElement.Parse(XmlValue); }
    set { XmlValue = value.ToString(); }
}
```

روش عمومی کار با نوع‌های خاصی که در EF تعریف نشدن، استفاده از ویژگی Column و مشخص کردن Type آن است؛ مانند مثالی که در بالا ملاحظه می‌کنید. البته این نوع خاص، در سمت کدها باید به صورت رشته تعریف شود. مثلا از سال 2005 به این طرف فیلد XML به SQL Server اضافه شده. اما نمی‌شود ازش در EF به همون شکل XML استفاده کرد. باید تبدیلش کنی به String تا قابل استفاده بشه. یک نمونه دیگرش نوع خاص Spatial هست که در نگارش‌های اخیر SQL Server اضافه شده (geography و geometry). این مورد فقط از EF 5.0 به بعد پشتیبانی توکاری ازش ارائه شده. یا برای hierarchyID در EF معادلی وجود نداره. برای تعریف این مورد نیز در یک مدل باید از string استفاده کرد.

بعد اگر این نوع خاص (که الان به صورت رشته دریافت شده) قابل نگاشت به نوعی مشخص در سمت کدهای برنامه بود (یعنی صرفا یک رشته ساده نبود) مثلا می‌شود از ویژگی NotMapped برای تبدیل آن و تعریف آن به شکل یک فیلد محاسباتی استفاده کرد.

نویسنده: پرنیا
تاریخ: ۱۳۹۲/۰۷/۲۱ ۲۳:۱۶

با سلام
پروژه ای که شرح دادین رو ایجاد و در Publish ، SQL server 2012 کردم و در جدول هم مقادیر تستی درج کردم.
زمانی که جدول را در Object Browser با Mode ویرایش باز می‌کنم هیچ مشکلی وجود ندارد و داده‌ها درست نمایش داده می‌شوند اما زمانی که با دستورات T-SQL کار میکنم مقادیر را به صورت یک رشته از کاراکترها نمایش می‌دهد که نامفهوم می‌باشد.
تصویر اجرای کوئری‌ها را میذارم لطفا راهنمایی کنید.

	Id	TestDate
▶	۱	۱۳۹۲/۰۲/۰۹ ۰۰:۰۰:۰۰
	۲	۱۳۹۲/۰۲/۰۹ ۲۲:۴۰:۰۰
	۳	۱۳۹۲/۱۲/۲۹ ۲۲:۴۰:۰۰
*	NULL	NULL

```
select *
from TestTable
```

۱۰۰ %

Results | Messages

	Id	TestDate
۱	۱	۰x۸۵۷۰۰۲۰۹۰۰۰۰۰۰۰۰
۲	۲	۰x۸۵۷۰۰۲۰۹۱۶۲۸۰۰۰۰
۳	۳	۰x۸۵۷۰۰C۱D۱۶۲۸۰۰۰۰

```
select dbo.GetCurrentDateTime()
```

۱۰۰ %

Results | Messages

	(No column name)
۱	۰x۸۵۷۰۰۷۱۵۱۷۰A۳A۰۰

نویسنده: وحید نصیری

تاریخ: ۱۳۹۲/۰۷/۲۲ ۰:۴

SQLCLR types به صورت پیش فرض با فرمت serialized binary value بازگشت داده می‌شوند.

```
SELECT id, TestDate.ToString() FROM TestTable;
```

در ابزارهای کوئری گرفتن ad-hoc مثل SSMS باید یک متد ToString را هم به انتهای نام ستون اضافه کنید تا مقدار نمایشی واضحی حاصل شود.

عنوان: **آشنایی با FileTable در SQL Server 2012 بخش 1**
نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۱۶ ۲۰:۲۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SQL Server 2012

پیش از آشنایی با FileTable نیاز است که پیشینه‌ای از شیوه‌های ذخیره‌سازی فایل و یا بهتر بگویم BLOB در SQL Server را داشته باشیم. نخستین شیوه‌ی نگه‌داری فایل استفاده از Image است که در SQL Server 2000 کاربرد داشت و هم‌اکنون استفاده از آن به دلیل کاهش بسیار کارآیی منسوخ‌شده است. به دلایل مشکلات بسیار فراوان Image هم‌زمان بسیاری از طراحان پایگاه داده‌ها، جهت کاهش حجم جدول‌ها و پیروی آن حجم پایگاه داده‌ها، فایل را در سیستم‌فایل نگه‌داری می‌کردند و تنها مسیر آن را در فیلدی از نوع کاراکتری در پایگاه‌داده‌ها ذخیره می‌کردند. این روش هرچند از حجم پایگاه داده‌ها می‌کاست ولی به دلیل عدم دخالت SQL Server در مدیریت فایل‌ها مشکلات دیگری را به وجود آورد.

از SQL Server 2005 نوع داده‌ی varbinary(max) معرفی شد که برخی از چالش‌های به‌کاربری Image را کاست و درباره‌ی بسیاری از موارد مانند ذخیره‌ی عکس پرسنلی هنوز هم کاربرد دارد؛ ولی توجه داشته باشید که استفاده از این فیلد فقط برای فایل‌های کم‌تر از 256 کیلوبایت سفارش شده است و برای بالاتر از آن، کارآیی کاهش فراوانی خواهد یافت.

در SQL Server 2008 نوع داده‌ی جدیدی به نام FileStream به وجود آمد به این شکل که یک FileGroup از نوع Data FileStream به پایگاه‌داده افزوده می‌شود و در واقع با یک پوشه در سیستم فایل در پیوند است. از این پس هنگام ساخت یک جدول به جای استفاده از نوع داده‌ی varbinary از نوع FileStream استفاده می‌کنیم با مد نظر داشتن این نکته که حتماً باید یک فیلد از نوع Uniqueidentifier هم در آن جدول تعریف شده باشد. شیوه‌ی کار نیز به این صورت خواهد بود که خود رکورد در جدول ذخیره می‌شود و فقط محتوای فایل در آن مسیری از NTFS ذخیره می‌شود. برخلاف روش درج مسیر فایل در جدول که پس از حذف رکورد، فایل هم‌چنان در سیستم فایل می‌ماند؛ این بار با حذف رکورد فایل مربوطه نیز حذف خواهد شد. افزون بر این مدیریت پشتیبانی از فایل‌ها نیز برعهده‌ی پایگاه داده‌ها خواهد بود. اندازه‌ی فایل‌ها در FileStream محدودیت‌های پیشین را نخواهد داشت و شما به اندازه‌ی حجم درایو هارددیسک می‌توانید فایل در آن ذخیره کنید. نکته‌ی دیگر درباره‌ی فایل‌های با حجم سنگین که می‌توانید Stream مربوط به یک فایل را به صورت بخش‌بخش در سمت مشتری بارگذاری کنید و به او نشان دهید. در FileStream امنیت و تراکنش فایل‌ها برعهده‌ی SQL Server است و از این دیدگاه بسیار ساده‌تر و کارآتر از FileSystem است.

(برای آشنایی بیشتر با FileStream، این نوشتار از مهندس وحید نصیری را مطالعه کنید.)

گونه‌ی FileTable از ویژگی‌های نوین SQL Server 2012 است که تکمیل‌کننده‌ی FileStream است. FileTable آمیزشی از FileStream با hierarchyid و سیستم فایل ویندوز برای ارائه‌ی توانایی‌های نوین مدیریت BLOB در SQL Server است. FileTable همان‌گونه که از دو واژه‌ی تشکیل‌دهنده‌اش پیداست؛ هم‌زمان یک جدول و یک سیستم فایل معمولی است.

FileTable به هر روی یک جدول از پایگاه‌داده‌های SQL Server است با یک تفاوت که ساختار آن از پیش تعریف‌شده است. ستون‌های FileTable و نوع داده‌ی آن از پیش توسط SQL Server مشخص شده است. ستون‌های تشکیل‌دهنده‌ی FileTable دربرگیرنده‌ی جدول زیر است:

نام ستون	DataType	شرح
stream_id	Uniqueidentifier ROWGUIDCOL	شناسه‌ی یکتای ردیف
file_stream	varbinary(max) FILESTREAM	محتوای فایل، اگر پوشه باشد؛ Null است.
name	nvarchar(255)	نام فایل یا پوشه
path_locator	hierarchyid	جای فایل یا پوشه در سیستم‌فایل
creation_time	datetimeoffset(7)	زمان ایجاد
last_write_time	datetimeoffset(7)	زمان واپسین ویرایش
ast_access_time	datetimeoffset(7)	زمان واپسین دسترسی
is_directory	bit	۰ = فایل / ۱ = پوشه
is_offline is_hidden is_readonly is_archive is_system is_temporary	bit	

هر ردیف از FileTable نماینده‌ی یک فایل یا پوشه در File System است. ستون path_locator که از نوع hierarchyid است نشان‌دهنده‌ی مسیر یک فایل یا پوشه است. hierarchyid که از SQL Server 2008 معرفی شده است؛ بهترین نوع داده برای نگه‌داری ارتباط ساختار سلسله‌مراتبی مانند چارت سازمانی، درخت تجهیزات یک کارخانه و یا در همین نمونه درخت فایل‌ها و پوشه‌ها است. پس می‌توانیم از همه‌ی امکانات hierarchyid در FileTable نیز برخوردار شویم. این‌که این فایل به ترتیب در چه پوشه‌هایی قرار گرفته است یا این‌که این پوشه شامل چه فایل‌ها یا پوشه‌هایی خواهد بود. این‌که پوشه‌های هم‌فرزند پوشه‌ی جاری کدام است و یا یا توابع مربوط به جابه‌جایی فایل‌ها و پوشه‌ها.

دنباله دارد...

نظرات خوانندگان

نویسنده: امیر بختیاری
تاریخ: ۱۳۹۲/۰۲/۱۷ ۱۶:۲۶

با سلام و خسته نباشید
مطلب خیلی خوبی بود فقط اگر امکان دارد در یک بخش درباری hierarchyid نیز کامل به همراه مثال توضیح دهید
با تشکر

نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۱۷ ۱۷:۲۷

درود بر شما
پیشنهاد خوبی است. پیش از این نوشتاری در این باره نوشته بودم (هرچند منتشر نکرده ام.) ولی با یادداشت شما به این اندیشه افتادم که مروری بر این بحث در SQL Server 2012 داشته باشم و با ویرایشی نوین، در این تارنما منتشر کنم.

نویسنده: ali.rezayee
تاریخ: ۱۳۹۲/۰۲/۱۷ ۱۹:۳۱

با سلام و تشکر از این مطلب خوب.
آیا نوع فیلدها قابل تغییر است؟
برای مثال در فیلد Name چون varchar است نمیتوان نام فارسی برای فایل ثبت کرد، آیا امکان تغییر آن به Nvarchar هست؟
ممنون از شما.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۷ ۲۲:۵۸

در اصل nvarchar هست که به نظر در جدول فوق باید اصلاح بشه.

نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۱۸ ۶:۵۳

بله. اشتباه از من بود. این فیلدها غیرقابل تغییر هستند ولی هیچ مشکلی هم با زبان پارسی ندارند.

نویسنده: سید امیر سجادی
تاریخ: ۱۳۹۲/۰۹/۱۴ ۲۰:۱۹

سلام. ممنون از مطلب مفیدتون. فقط یک مشکلی هست و اون عدم پشتیبانی توسط Linq2Sql هست. فکر کنم فقط با DataSet مشکلی نداشته باشه و اون هم بخاطر این هست که نوع hierarchyid به نوع nvachar(867) تبدیل میکنه.
لطفا اگه برای این مشکل راه حلی هست بنویسید که دوستان هنگام استفاده به مشکل برنخورند

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۹/۱۴ ۲۰:۳۱

Linq2Sql که خیلی وقت هست با EF جایگزین شده. EF هم از نوع‌های خاص دیتابیس‌ها عموما پشتیبانی نمی‌کنه. بحث EF هم نیست. اکثر ORMهای موجود همینطور هستند.

نویسنده: سید امیر سجادی

تاریخ: ۱۳۹۲/۰۹/۱۴ ۲۳:۸

یعنی فقط باید از DataSet استفاده کرد !؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۹/۱۵ ۱:۱

ADO.NET می‌تونه خروجی از نوع DataSet هم بده؛ ولی اسمش DataSet نیست. تمام ORMها هم برفراز ADO.NET کار می‌کنند. بنابراین اگر تصور کنیم که ابدا نمی‌شود با این‌ها کار کرد، خیر. اینطور نیست. شما با ORMها هم می‌تونی مستقیما SQL بنویسی ef-code-first-13 تا به قابلیت‌هایی دسترسی پیدا کنی که در ORM پیش بینی نشده.

نویسنده: سعید
تاریخ: ۱۳۹۲/۱۰/۱۴ ۲۳:۵۱

سلام
چطور امنیت فایل‌ها رو برقرار کنیم ؟ منظورم اینه که کسی نتونه اونا رو پاک یا مشاهده و یا اضافه کنه ، مگه اینکه ما بهش اجازه بدیم
راهی برای این کار هست ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۱۵ ۰:۴۵

مگه کاربران شما مستقیما با فایل سرور کار می‌کنند؟ روال عادی آن کار از طریق یک برنامه مثلا وب هست. امنیت رو در اونجا پیاده سازی کنید.

نویسنده: سعید
تاریخ: ۱۳۹۲/۱۰/۲۲ ۱۱:۱۱

من دارم از اپلیکیشن استفاده میکنم . از داخل برنامه درست هست ، یعنی همونی که شما میگی . ولی اگه کسی از بیرون برنامه بخواد این کارو کنه چی ؟
مستقیم بره تو پوشه و فایل‌ها رو تغییر بده . من میخوام بدونم sql میتونه جلوی این کارو هم بگیره ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۲ ۱۲:۱۲

- به چه نحوی از SQL Server استفاده می‌کنید؟ آیا سرور و برنامه دسکتاپ شما روی یک کامپیوتر هستند؟ برای اینکار بهتر است از SQL CE یا SQLite استفاده کنید؛ یا حتی LocalDB. هدف از SQL Server نصب آن روی یک سرور و خدمات دهی به چندین و چند کامپیوتر تحت شبکه است. برای استفاده روی یک کامپیوتر یعنی کسب و کار کوچک و عملا نیازی به SQL Server 2012 ندارد اینکار. زندگی مصرف کننده را سخت نکنید. نصب و نگهداری یک سرور کار هر شخصی نیست و برای سازمان‌ها طراحی شده و نه مصارف کوچک تک کاربره دسکتاپ.

- با این توضیح اگر کسی به سرور شما دسترسی دارد، آیا نمی‌تواند مثلا اگر فایل‌ها در دیتابیس ذخیره می‌شدند، اون‌ها رو دستی با یک کوئری حذف کند؟ امنیت کار با سرور این‌جا است که مطرح می‌شود و همچنین اطمینان به ادمین‌ها.

- در مورد امنیت file table مراجعه کنید به مستندات مایکروسافت. مثلا: FileTables are secured by SQL Server security only

عنوان: آشنایی با FileTable در SQL Server 2012 بخش 2
نویسنده: حامد قنادی
تاریخ: ۱۳۹۲/۰۲/۱۸ ۱۵:۳۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SQL Server 2012

ستون دیگر **stream_id** نام دارد که از نوع uniqueidentifier ROWGUIDCOL است. همان‌گونه که در یاد دارید، در FileStream نیز ناگزیر به تعریف چنین ستونی بودیم. بنابراین FileTable استثناء نیست و در این‌جا نیست چنین فیلدی توسط SQL Server تعریف می‌شود. اگر فایل‌ها و پوشه‌ها جابه‌جا نمی‌شدند می‌توانستید از هر دو ستون path_locator یا stream_id برای شناسایی یک رکورد از جدول بهره ببرید. ولی با جابه‌جایی یک فایل و یا به عبارت دیگر تغییر پدر آن در ساختار سلسله‌مراتبی، مقدار path_locator نیز تغییر می‌کند، پس ناگزیر به استفاده از این ستون برای ارجاع به یک ردیف در جدول هستیم.
هر ردیف از جدول نمایان‌گر یک فایل یا پوشه است، بنابراین به ستونی نیاز داریم که بتوانیم این موضوع را نشان دهیم. بر این پایه از ستون **is_directory** بهره می‌بریم که 1 بودن آن نشان‌دهنده‌ی این است که این ردیف از جدول به یک پوشه ارجاع دارد.
نام فایل یا پوشه در ستونی به نام **name** نگه‌داری می‌شود که رشته‌ای از نوع nvarchar(255) است. افزون بر این ستون، ستون‌های دیگری نیز در این جدول وجود دارد که ویژگی‌های یک فایل مانند پنهان‌بودن، فقط‌خواندنی و ... توسط آن توسط آن به دست می‌آید. ستون پسین **file_stream** نام دارد که برای پوشه‌ها، محتوای آن Null است. علت آن این است که محتوای واقعی فایل در این ستون نگه‌داری می‌شود. در واقع یک varbinary(max) با ویژگی‌های fileStream است که محتوای باینری آن در سیستم فایل NTFS ذخیره می‌شود. مدیریت پشت صحنه‌ی این ستون برعهده‌ی SQL Server است.
افزون بر این 14 ستون، هر FileTable شامل سه ستون محاسباتی به شرح زیر است:

نام ستون	DataType	شرح
parent_path_locator	hierarchyid	مسیر به‌دست‌آمده از path_locator
file_type	nvarchar(255)	گونه‌ی فایل به دست‌آمده از name
cached_file_size	bigint	اندازه‌ی فایل به دست‌آمده از file_stream

ستون parent_path_locator نتیجه‌ی فراخوانی تابع GetAncestor(1) در ستون path_locator است که جهت به دست آوردن پوشه‌ی پدر یک فایل و پوشه استفاده می‌شود. ستون file_type که از مقدار رشته‌ای ستون name تجزیه شده است، پسوند فایل را برمی‌گرداند. و ستون cached_file_size اندازه‌ی بایت ذخیره‌شده ستون file_stream را برمی‌گرداند. با این ساختار ثابت در اینجا، هر FileTable هر آن‌چه از File System نیاز دارید در یک پوشه‌ی اشتراکی به شما می‌دهد.

این یعنی نمایش بی‌واسطه FileTable به هر کاربر یا برنامه. به طوری که برای نمایش یا به‌روزرسانی جدول می‌توانید از روش استاندارد I/O مانند کشیدن و رهاکردن با Windows Explorer یا برنامه‌نویسی با System.IO.FileStream و APIهای ویندوز استفاده کنید. این‌چنین:

- ایجاد یک فایل یا پوشه در سیستم فایل -> افزودن یک ردیف به جدول

- افزودن یک ردیف به جدول -> ایجاد یک فایل یا پوشه در سیستم فایل

با کپی فایل‌ها در مسیر بالا، به صورت خودکار رکوردهای زیر در جدول PhotoTable در پایگاه‌داده‌ها افزوده می‌شود:

	stream_id	file_stream	name	path_locator	parent_pa
1	FABBA0DA-A1B7-E211-...	0x89504E470D0A1A0A0000000D49484452...	Pic13920218.png	0xFF1E127B0A0A9B2FDDD353FFA0403...	NULL
2	A0CB07E7-A1B7-E211-...	0xFFD8FFE000104A4649460001020100600...	Hydrangeas.jpg	0xFECCB4A9F110A38FD391534845521...	NULL
3	A2CB07E7-A1B7-E211-...	0xFFD8FFE000104A4649460001020100600...	Chrysanthemum.jpg	0xFD64A656CE5A91CFDB125242D522...	NULL
4	A4CB07E7-A1B7-E211-...	0xFFD8FFE000104A4649460001020100600...	Desert.jpg	0xFF6A01070D9AB3EFC61352DE20FB1...	NULL
5	3DCCC316-A2B7-E211-...	NULL	Archives-92-02	0xFD00916BA6D2930FCFD2351E24281...	NULL
6	63B6DF2A-A2B7-E211-...	0xFFD8FFE000104A4649460001020100600...	Tulips.jpg	0xFD00916BA6D2930FCFD2351E24281...	0xFD009

به طور خلاصه پیش از این برای افزودن به FileStream دو راه کار پیش رو داشتید. یکی استفاده از T-SQL و دیگر sqlFileStream اکنون SQL Server 2012 راه کار سوم را پیشنهاد می‌کند. استفاده از File System در این روش FileStream به طور خودکار پر می‌شود.

پیش از ساخت یک FileTable بیان این نکته دارای اهمیت است که با کپی فایل‌ها و پوشه‌ها هیچ چیز جدیدی به NTFS افزوده نمی‌شود بلکه محتوای فایل به FileStream افزوده می‌شود و SQL Server با بررسی همزمان FileStream و FileTable نمایشی از ردیف‌های FileTable به صورت یک پوشه‌ی اشتراکی نشان می‌دهد. این نکته پاسخی به این پرسش خواهد بود که آیا با استفاده از FileTable حجم پایگاه‌داده‌ها دو برابر خواهد شد و در نتیجه دشواری‌ها و چالش‌های نگه‌داری و پشتیبانی را پیش رو خواهیم داشت!؟ که پاسخ "خیر" خواهد بود.

ایجاد یک FileTable

پیش از این در همین تارنما، روش فعال کردن FileStream در SQL Server را آموزش دیده اید. اگر درست به خاطر داشته باشید، چیزی شبیه به دستورهای زیر بود:

```
CREATE DATABASE MyFileArchive
ON PRIMARY
(NAME = MyFileArchive_data,
FILENAME = 'C:\Demo\MyFileArchive_data.mdf'),
FILEGROUP FileStreamGroup CONTAINS FILESTREAM
(NAME = PhotoFileLibrary_blobs,
FILENAME = 'C:\Demo\MyFiles')
LOG ON
(NAME = PhotoFileLibrary_log,
FILENAME = 'C:\Demo\MyFileArchive_log.ldf')
```

FileTable به FileStream متکی است؛ بر این پایه پیش از ایجاد یک FileTable باید FileStream را روی پایگاه‌داده‌ها فعال کنیم. این کار با یک تعریف درست توسط بند FILEGROUP...CONTAINS FILESTREAM انجام می‌شود.

برای ایجاد FileTable تنها کافی است که بند WITH FILESTREAM را به دستور CREATE DATABASE بیفزایید. (یا برای فعال‌کردن FileTable روی یک پایگاه‌داده‌ی ساخته شده بند SET FILESTREAM را در دستور ALTER DATABASE بنویسید.) در این بند، از DIRECTORY_NAME برای نام‌گذاری یک پوشه برای پایگاه‌داده‌ها استفاده می‌کنیم. این پوشه در یک پوشه ریشه به نام SQL Server instance نمایش داده خواهد شد. بخش دوم بند NON_TRANSACTED_ACCESS=FULL است که دسترسی غیرتراکنشی را فعال می‌کند. با این کار برای هر FileTable در پایگاه داده یک زیرپوشه درون پوشه‌ای که به نام DIRECTORY_NAME نام‌گذاری شده است؛ ساخته می‌شود.

با توجه به آنچه گفته شد برای ایجاد یک پایگاه‌داده با امکان ساخت FileTable دستورهای زیر را اجرا کنید:

```
CREATE DATABASE MyFileArchive
ON PRIMARY
(NAME = MyFileArchive_data,
FILENAME = 'C:\Demo\MyFileArchive_data.mdf'),
FILEGROUP FileStreamGroup CONTAINS FILESTREAM
(NAME = PhotoFileLibrary_blobs,
FILENAME = 'C:\Demo\MyFiles')
LOG ON
(NAME = PhotoFileLibrary_log,
FILENAME = 'C:\Demo\MyFileArchive_log.ldf')
WITH FILESTREAM
(DIRECTORY_NAME='FilesLibrary',
NON_TRANSACTED_ACCESS=FULL)
```

اکنون برای ساخت یک FileTable درون این پایگاه‌داده‌ها از دستور زیر استفاده کنید:

```
USE MyFileArchive
GO
CREATE TABLE PhotoTable AS FileTable
GO
```

توجه داشته باشید که چون ستون‌های FileTable از پیش تعریف شده است؛ ایجاد آن فقط با نوشتن دستور امکان پذیر است و مانند یک Table عادی از محیط کاربری SQL Server نمی‌توان بهره برد.
در Object Explorer از گره‌ی Tables، گره‌ی FileTables را باز کنید و روی جدولی که هم‌اکنون ساختیم راست‌کلیک کنید. با انتخاب گزینه‌ی Explore FileTable Directory پنجره‌ی زیر بازمی‌شود:

\\041-computer-35\mssqlserver\FilesLibrary
Search FilesLibrary
Organize
New folder
DVD Drive (G:)
CD Drive (H:) CD_ROM
Network
041-COMPUTER-35
mssqlserver
FilesLibrary
PhotoTable
Archives-92-02
SQLFileStream
Name
Date modified
Type
Size
PhotoTable
File folder
1 item

دنباله دارد ...

نظرات خوانندگان

نویسنده: ali.rezayee
تاریخ: ۱۳۹۲/۰۳/۰۴ ۱۶:۳۶

با سلام
موضوع بسیار جالب و مهمی است، لطفاً سریع‌تر ادامه دهید.

نویسنده: علی رضایی
تاریخ: ۱۳۹۳/۰۴/۲۷ ۱۹:۵۸

بهروش‌های استفاده از FILESTREAM

نویسنده: M.Shakeri
تاریخ: ۱۳۹۳/۰۵/۰۸ ۱۰:۱۴

فکر می‌کنم برای افزودن این قابلیت به دیتابیس دسترسی به دیتابیس master نیاز باشه که هاست‌ها این امکان رو نمیدن.

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۵/۰۸ ۱۲:۸

خیلی از قابلیت‌ها مخصوص سازمان‌ها طراحی می‌شن.

نویسنده: Hamidbp
تاریخ: ۱۳۹۳/۰۷/۱۴ ۱۴:۴۰

می خواستم بدونم چطور میشه یک فایل رو که در Filetable ذخیره شده، به اصطلاح Fetch کنم و توسط برنامه مربوطه (مثلا adobe acrobat یا word یا ...) نشونش بدم؟

عنوان: نحوه انتقال اطلاعات استخراج شده از وب سرویس به SQL Server به کمک SSIS
نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۲/۰۲/۳۰ ۲۲:۱۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Web Service, SSIS, data mining

ممکن است در مواقعی نیاز به اطلاعات استخراج شده از وب سرویسی داشته باشید که در همان مقطع زمانی به آن دسترسی ندارید . مسلما برای این منظور باید آن اطلاعات را ذخیره کرده تا در صورت نیاز بتوان به آنها رجوع کرد . یکی از راه حل‌ها ذخیره آن در پایگاه داده (در اینجا Sql Server) است که در این پست به کمک امکانات BIDS در پکیج‌های SSIS و کوئری‌های SQL این مشکل را برطرف میکنیم. برای مشاهده نحوه استخراج اطلاعات از وب سرویس به اینجا مراجعه کنید .

تنها تفاوتی که در این پست در کار با سرویس با پست اشاره شده در بالا وجود دارد ذخیره اطلاعات استخراج شده است که در آن پست در یک فایل xml ذخیره شدند ولی در اینجا ما نیاز داریم تا اطلاعات را در یک متغیر با حوزه کاری Package ذخیره کنیم (به این معنی که مختص به همان flow نباشد و در تمام پکیج دیده شود)

به دلیل اینکه هدر xml خروجی از سرویس دارای چندین namespace هست هنگام کار با آنها به مشکل خواهیم خورد. (هم هنگام کار با xml taskها و هم هنگام کار با xml در sql)

```
▼<ArrayOfUserInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  ▼<UserInfo>
```

به همین دلیل باید این قسمت از محتوا را حذف کرد . برای همین پس از گرفتن اطلاعات از سرویس آن را به کمک یک Script task حذف می‌کنیم

```
public void Main()
{
    //xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://tempuri.org/"
    object o = "null";
    byte[] emptyBytes = new byte[0];
    try
    {
        Dts.VariableDispenser.LockForRead("User::XMLContent");
        Variable xmlc = Dts.Variables["User::XMLContent"];
        o = xmlc.Value;
        if (o != null)
        {
            string s = o.ToString();
            s = s.Replace("xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"", "");
            s = s.Replace("xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\"", "");
            s = s.Replace("xmlns=\"http://tempuri.org/\"", "");
            Dts.VariableDispenser.Reset();
            Dts.VariableDispenser.LockForWrite("User::XMLContent");
            Dts.Variables["User::XMLContent"].Value = s;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error SSIS : " + Environment.NewLine + Environment.NewLine + ex.Message, "Error on Package...");
        Dts.TaskResult = (int)ScriptResults.Failure;
    }
    finally
    {
        Dts.VariableDispenser.Reset();
    }

    Dts.Log("nameSpace Removed Successfully...", 0, emptyBytes);
    Dts.TaskResult = (int)ScriptResults.Success;
}
```

در این مرحله اطلاعات استخراج شده را باید در SQL درج کنیم . برای همین ساختاری که باید اطلاعات را در SQL نگه دارد را در دیتابیس ایجاد می‌کنیم :

جدول person برای نگهداری اطلاعات سرویس و XmlContainer برای نگهداری xmlهای سرویس .(برای داشتن History)

برای درج هم از SP استفاده می‌کنیم :

```
ALTER PROCEDURE [dbo].[Add_step1]

    @xml XML
AS
BEGIN
    SET NOCOUNT ON;

INSERT INTO dbo.XmlContainer
values( @xml, GETDATE())

DECLARE @id INT
SET @id = SCOPE_IDENTITY() ;

SELECT  @id AS Result


END
```

و پیکربندی SQL Task :

نکته اول ایجاد کانکشن به پایگاه داده است که در اینجا از نمایش جزییات آن صرف نظر شده است . نکته دیگری پارامتر SP است که چون یک پارامتر دارد یک علامت سوال قرار میگیرد . اگر چند پارامتر بود به صورت علامت‌های سوال با ویرگول از هم جدا می‌شوند . ?,?,?,?

سپس در بخش parameter mapping به ترتیب مقدار دهی می‌شوند :

و مقدار خروجی SP که شناسه آیتم درج شده است را در یک متغیر نگهداری می‌کنیم :

برای قدم بعد می‌خواهیم اطلاعات موجود در XML استخراج شده در پایگاه داده را در جدول مربوطه ذخیره کنیم . برای این کار این SP را می‌نویسیم :

```
ALTER PROCEDURE [dbo].[transform_Step2]
 @id int
AS
BEGIN

    SET NOCOUNT ON;
    DECLARE @trans CHAR(50),@xml XML
    SET @trans = 'MyTrans'
    BEGIN TRY
    BEGIN TRANSACTION @trans

    DELETE FROM Person -- empty table
    SELECT @xml = xc.XmlContent FROM XmlContainer xc
    WHERE xc.ID = @id
    DECLARE  @dt DATETIME , @hoc int;
    EXEC sp_xml_preparedocument @hoc OUTPUT, @xml
    INSERT INTO Person (ID,FirstName,LastName)
    SELECT ID, FirstName, LastName
    FROM OPENXML(@hoc, 'ArrayOfUserInfo/UserInfo')
    WITH
    (
    ID [nvarchar](50) 'ID',
    FirstName [nvarchar](250) 'FirstName',
    LastName [nvarchar](250) 'LastName'
    )
    EXEC sp_xml_removedocument @hoc -- empty cache
    IF (@@ERROR <> 0 )
    BEGIN
        ROLLBACK TRANSACTION @trans
    END
    ELSE
    BEGIN
    COMMIT TRANSACTION @trans
     SELECT 'OK' AS  'ERROR'
    END

    END TRY
    begin CATCH
        ROLLBACK TRANSACTION @trans
        SELECT  ERROR_MESSAGE()  AS  'ERROR'
    END CATCH
END
```

نکات مهم موارد زیر هستند :

1 - استفاده از OpenXml برای parse کردن xml

2 - استفاده از sp سیستمی sp_xml_preparedocument و sp_xml_removedocument (بیشتر)

3- اطلاعات شناسه و نام و نام خانوادگی inner textهای نودهای xml هستند . اگر این موارد به صورت attribute باشند باید قبل از نام آنها علامت @ قرار بگیرند.

پس از ایجاد این Sp باید آن را در package فراخوانی کنیم :

و پیکربندی این SQL Task :

و پارامترهای این SP :

و ذخیره نتیجه تراکنش در متغیری در پکیج :

و اکنون پکیج ما ظاهری شبیه به این مورد خواهد داشت :

در نهایت به عنوان یک facility می‌توانیم وضعیت تراکنش را به کاربر نمایش دهیم (به کمک Script Task) :

```
public void Main()
{

    const string _id = "User::SqlResultId";
    const string _err = "User::FinalState";

    Dts.VariableDispenser.LockForRead(_id);
    Dts.VariableDispenser.LockForRead(_err);


    string id = Convert.ToString( Dts.Variables[_id].Value);
    string error = Convert.ToString(Dts.Variables[_err].Value);

    string msg = "با موفقیت با شناسه "+id+" درج گردید";
    Dts.VariableDispenser.Reset();

    if (error.Equals("OK"))
    {
        MessageBox.Show(msg,"موفقیت در عملیات");
        Dts.TaskResult = (int)ScriptResults.Success;
    }
    else
    {
        MessageBox.Show("Error : "+Environment.NewLine+error, "خطا در انجام عملیات");
        Dts.TaskResult = (int)ScriptResults.Failure;
    }


    // TODO: Add your code here
}
```

و تمام ...

Results | Messages

	ID	XmlContent	DateOfInsertion
1	7	<ArrayOfUserInfo><UserInfo><ID>bd36d158-f2bd-40d...	2013-05-16 21:36:44.84
2	8	<ArrayOfUserInfo><UserInfo><ID>5848557b-6482-4d8...	2013-05-16 21:44:21.49
3	9	<ArrayOfUserInfo><UserInfo><ID>5f465898-bce8-4153...	2013-05-16 21:45:02.61
4	10	<ArrayOfUserInfo><UserInfo><ID>0836a55b-3d8f-4ce...	2013-05-16 21:53:19.83
5	11	<ArrayOfUserInfo><UserInfo><ID>73670cd5-1d09-4a9...	2013-05-16 22:35:52.70
6	12	<ArrayOfUserInfo><UserInfo><ID>316e06a0-4063-479...	2013-05-16 22:36:30.89
7	13	<ArrayOfUserInfo><UserInfo><ID>c56ab672-9415-418...	2013-05-16 22:38:56.86
8	14	<ArrayOfUserInfo><UserInfo><ID>c991b4ee-a389-427...	2013-05-17 10:55:12.27

موفق باشید

نظرات خوانندگان

نویسنده: Amir
تاریخ: ۱۳۹۲/۱۰/۱۸ ۰:۴۴

با سلام وسپاس از مطالب مفیدتون

سوالی دارم ممنون میشم اگه جواب بدید

برای انتقال داده‌های هشت سرور که از نوع پارادوکس به SQL Server 2012 هستند آیا به غیر از روش SSIS راه دیگری وجود دارد ؟ ناگفته نماند که داده‌ها حجم قابل توجهی دارند و با روش SSIS سرعت شبکه را کاهش می‌دهد.

نکته دیگر، داده‌ها به صورت کامل به SQL Server منتقل نمیشوند و بعضی جداول منتقل شده یا خالی هستند یا جداول تکراری داریم در SQL Server که در نتیجه باعث قابل اطمینان نبودن Data Warehouse میشود.

با سپاس

نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۲/۱۰/۱۸ ۱۲:۲۹

سلام
اجازه بدید اول به این نکته اشاره کنم که SSIS روش نیست بلکه یک ابزاره. روش مد نظر برای Migration داده ، ETL نام داره (Extract , Transform , Load) نه SSIS.
حالا برای انجام عمل ETL ابزارهای دیگری هم میشه پیدا کرد مثل informatica یا DB2 Infosphere که می‌تونید تعدادی از اون‌های رو اینجا ببینید
در مورد مشکلاتی که در باره داده‌ها یا شبکه می‌فرمایید هم باید یک plan مناسب برای این مهاجرت داده ای ایجاد کنید . جهت اطلاع عرض کنم که این پروسه ممکنه گاها تا یک ماه طول بکشه بسته به پلنی که طراحی شده و شرایط و محدودیت هایی که موجوده. اطلاعات بیشتر رو می‌تونید از اینجا کسب کنید
موفق باشید

نویسنده: Amiir
تاریخ: ۱۳۹۲/۱۰/۱۸ ۱۳:۴۵

ممنون از راهنمایتون

عنوان: **Full Text Search و Rank فیلدهای بازیابی شده**
نویسنده: دل محسن
تاریخ: ۱۳۹۲/۰۳/۱۰ ۱۰:۵۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Full Text Search

حالتی را در نظر بگیرید که بخواهید تعداد زیادی رکورد را که حجم هر رکورد هم قابل ملاحظه هست، نگهداری کنید(مثلا چندین هزار مقاله) و همچنین قابلیت جستجو را در این رکوردها لحاظ کنید به صورتی که بر اساس رکوردهایی که بیشترین تعداد تکرار کلمات مدنظر را دارند مرتب شوند.
شاید اولین راه حل، مطلب آقای سلیم آبادی در اینجا باشه، که تعداد تکرار یک کلمه را در فیلدی در جدول بیان کردند و درست هم هست اما با 2 شرط:
1) رکوردهای ما حجم کمی داشته باشند چرا که دستور LIKE پاسخ سریعی را با حجم بالای اطلاعات به ما نمی‌دهد.
2) رکوردهای ما از خانواده‌ی char باشند. برای مثال اگر مقالات ما به صورت pdf باشند این کد جواب نمی‌دهد.

اما راه دوم استفاده از Ful Text Search و دستور CONTAINSTABLE هست که 2 شرط لازم را برای راه حل اول احتیاج نداره. در اینجا فقط نحوه‌ی استفاده از CONTAINSTABLE رو مطرح میکنیم.

CONTAINSTABLE جدولی از موارد یافت شده را بر اساس معیارهایی که ما به اون معرفی می‌کنیم، ایجاد می‌کند. این جدول حاوی دو فیلد KEY(کلید فیلد مورد نظر) و RANK(مقداری بین 0 تا 1000) است که میزان همسانی رکورد با معیار ما را مشخص می‌کند و ما با استفاده از این فیلد می‌توانیم رکوردهایمان را مرتب کنیم.

به این کد توجه کنید:

```
SELECT t.Title, p.[RANK]
    FROM Articles AS t
        INNER JOIN CONTAINSTABLE(Articles, Data, 'management' ) AS p
            ON t.Id = p.[KEY]
ORDER BY p.RANK
```

در اینجا کار جستجو انجام شده و بر حسب میزان نزدیکی محتویات رکورد با معیار ما مرتب شده است.
نکته: هیچ فرقی نمی‌کند که محتویات فیلد مورد نظر شما یک متن ساده، یک فایل word یا حتی pdf باشد. فقط باید تنظیمات Full Text Search درست انجام شود.

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۳/۱۰ ۲۲:۴۹

برای PDF که عنوان کردید، آیا جستجوی فارسی کار می‌کند؟ از چه IFilter ایی برای اینکار استفاده کردید؟

نویسنده: دل محسن
تاریخ: ۱۳۹۲/۰۳/۱۱ ۱۴:۵۰

بله کار میکنه. من با Adob IFilter(رایگان) و Foxit IFilter(تجاری هست) کار کردم.

نویسنده: علی
تاریخ: ۱۳۹۲/۰۶/۰۶ ۱۲:۲۲

سلام.
من با rank کار می‌کنم توی پروژه هام.
منتها خیلی وقت‌ها رکوردهایی رنک بالایی می‌گیرند در حالی که رکوردهای مشابه دیگر رنک پایینی می‌گیرند. البته کمتر از 5 درصد اتفاق می‌افته
در مورد خود رنک مطلب خوبی پیدا نکردم که اصلا خود مفهوم رنک رو برام توضیح بده

نویسنده: علی
تاریخ: ۱۳۹۲/۱۰/۲۴ ۱۹:۲۷

منم توی این رنک بدجور گیر کردم. به جابجایی کامات حساسه . مثلا «خواجه علی» و «علی خواجه ».

عنوان: بررسی دستور Truncate Table و Delete
نویسنده: فرید طاهری
تاریخ: ۱۳۹۲/۰۳/۱۰ ۱۳:۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, T-SQL, Delete, Truncate Table

قبل از اینکه این موضوع را بررسی کنیم باید با دستور Truncate و Delete آشنا شویم.

بررسی دستور Delete :

همانگونه که می‌دانیم از این دستور برای حذف رکوردها استفاده می‌کنند. با اجرای دستور Delete به راحتی می‌توانید تعدادی از رکوردهای یک جدول را حذف کنید. ساده‌ترین شکل استفاده از دستور Delete به صورت زیر می‌باشد.

```
DELETE FROM table_name
WHERE some_column=some_value
```

برای مثال در صورتیکه بخواهیم مشتریانی را حذف کنیم که کشور (Country) آنها USA است باید از دستور زیر استفاده کنیم.

```
DELETE FROM Customers
WHERE Country='USA'
GO
```

با اجرای این دستور هر رکوردی که در شرط مربوطه صدق کند حذف خواهد شد. (خوب این رو که همه می‌دانند)
اما نکته مهمی که دستور Delete دارد این است کار این دستور به شکل Transactional می‌باشد. یعنی یا کلیه رکوردهایی که Country آنها USA است حذف می‌شود و یا هیچکدام از آنها. پس اگر شما 200000 رکورد داشته باشید که در این شرط صدق کند اگر وسط کار Delete (البته اگر عملیات حذف طولانی باشد) منصرف شوید می‌توانید با Cancel کردن این دستور عملیات Rollback Transaction را به خودکار توسط SQL Server داشته باشید. در صورتیکه عملیات Cancel را انجام دهید SQL Server از Log File برای بازگرداندن مقادیر حذف شده استفاده خواهد کرد.

اما نکته دیگری که دستور Delete دارد این است که این دستور Log کلیه رکوردهایی را که قرار است حذف کند در Log File می‌نویسد. این Log شامل اصل رکورد، تاریخ و زمان حذف، نام کاربر و... می‌باشد. شاید الان متوجه شوید که دستور Delete چرا در برخی از مواقع که قرار است حجم زیادی از اطلاعات را حذف نماید به کُندی این کار را انجام می‌دهد. (چون باید Log رکوردهای حذف شده در Log File نوشته شود.)

بررسی دستور Truncate Table:

Truncate در لغت به معنی بریدن و کوتاه کردن می‌باشد. با استفاده دستور Truncate Table می‌توانید محتوای کلیه رکوردهای موجود در یک جدول را در کسری از ثانیه حذف کنید.
نکته مهمی که باید درباره دستور Truncate Table بدانید این است که تاثیر استفاده از این دستور بر روی کلیه رکوردها بوده و به

هیچ عنوان نمی‌توان برای این دستور شرط (Where Clause) اعمال نمود.

شکل کلی استفاده از این دستور به صورت زیر می‌باشد.

```
TRUNCATE TABLE  table_name
GO
```

برای مثال اگر بخواهیم کلیه رکوردهای موجود در جدول Customers را حذف نمایید کافی است با استفاده از این دستور اینکار را انجام دهید

```
TRUNCATE TABLE Customers
GO
```

با اجرای این دستور در کسری از ثانیه کلیه رکوردهای جدول Customers حذف خواهد شد. (بهتر است از این دستور زمانی استفاده کنید که بخواهید ساختار جدول شما باقی بماند)

اما در مورد دستور Truncate Table و Delete باید به نکات زیر توجه کنید. 1- دستور Truncate Table فاقد قسمت شرط (Where Clause) می‌باشد در صورتیکه دستور Delete دارای قسمت شرط (Where Clause) است

2- دستور Truncate Table در Log File آدرس Page و مقدار فضای آزاد شده (کمترین میزان Log) را می‌نویسد اما در صورتیکه دستور Delete در Log هر رکوردی را که
قرار است حذف شود را در Log File ثبت می‌نماید.

3- دستور Truncate Table باعث می‌شود که Pageهای متعلق به جدول deallocate شوند. deallocate شدن Pageها این معنی را می‌دهد که رکوردهای موجود در جدول واقعاً حذف نشوند بلکه Extentهای مربوط به آن Pageها علامت Empty خورده تا دفعات بعد مورد استفاده قرار گیرند اما دستور Delete به طور فیزیکی محتوای Pageها مربوط به جدول را خالی می‌کند.

نکته : پس از Truncate شدن رکوردها امکان بازگشت آنها وجود ندارد.

4- در صورتیکه جدول شما دارای ایندکس باشد. دستور Truncate Table آزاد کردن فضای مربوط به ایندکس را در یک مرحله انجام می‌دهد(مطابق بند 3) همچنین Log مربوط به این حذف به شکل حداقل (مطابق بند 2) در Log File ثبت می‌شود. اما دستور Delete هر رکوردی را که از ایندکس حذف می‌کند در Log File ثبت می‌کند.

5- Trigger مربوط به دستور Delete به هیچ عنوان هنگام اجرای دستور Truncate Table فعال نمی‌شود. در صورتیکه با اجرای دستور Delete تریگر آن فعال خواهد شد.

6- در صورتیکه جدول شما دارایReference (Relation) باشد امکان استفاده از دستور Truncate Table وجود ندارد. لازم به ذکر است حتی اگر Reference را غیر فعال کنید
باز هم امکان استفاده از دستور Truncate Table وجود نخواهد داشت و تلاش برای اجرای دستور Truncate Table باعث نمایش خطای زیر خواهد شد.

```
TRUNCATE TABLE CUSTOMERS
GO
```

100 %

Messages

```
Msg 4712, Level 16, State 1, Line 1
Cannot truncate table 'Customers' because it is being referenced by a FOREIGN KEY constraint.
```

در صورتیکه در دستور Delete امکان حذف رکوردها به ازای جداولی که دارای Relation هستند وجود دارد. فقط باید به این نکته توجه کنید که ترتیب حذف رکوردها از جداول Master و Detail را رعایت کنید.

7- دستور Truncate Table مقدار Identity را Reset کرده و آن را به Seed (هسته/مقدار اولیه) بر می‌گرداند. در صورتیکه دستور Delete تاثیری بر روی مقدار Identity ندارد

8- دستور Truncate Table تنها توسط کاربرانی قابل اجرا است که نقش DB_Owner و یا SysAdmin را داشته باشند در صورتیکه دستور Delete توسط هر کاربری که مجوز Delete بر روی جدول را داشته باشد قابل اجرا می‌باشد.

9- پس از اجرای دستور Truncate Table تعداد رکوردهای حذف شده نمایش داده نمی‌شود. در صورتیکه هنگام اجرای دستور Delete تعداد رکوردهای حذف شده نمایش داده می‌شود.

```
TRUNCATE TABLE Customers2
GO
```

100 %

Messages

```
Command(s) completed successfully.
```

```
DELETE FROM Customers2
GO
```

100 %

Messages

```
(91 row(s) affected)
```

نظرات خوانندگان

نویسنده: محمد مهدی ملائیان
تاریخ: ۱۳۹۲/۰۴/۰۲ ۱۵:۳۰

مطلب مفیدی بود. منتظر مطالب شما هستم.

عنوان: **Identity و مباحث مربوط به آن (قسمت اول) - آشنایی با Identity**
نویسنده: فرید طاهری
تاریخ: ۱۳۹۲/۰۳/۲۱ ۱۴:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, T-SQL, Identity

Identity یکی از Attributeهایی که در SQL Server به ازای Columnهای عددی می‌توان در نظر گرفت. به طور خیلی ساده هنگامی که این Attribute به ازای یک فیلد عددی تنظیم گردد. چنانچه رکوردی در جدول مربوط به Identity درج شود فیلد Identity مقداری را به طور اتوماتیک دریافت خواهد نمود.
نحوه دریافت مقدار به ازای فیلد Identity با توجه به آخرین مقدار آن و گام افزایش است که در هنگام ایجاد identity تعریف می‌گردد.
برای ایجاد یک فیلد از نوع Identity می‌توانید زمانیکه جدول خود را ایجاد می‌کنید این Attribute را به فیلد مورد نظر خود تخصیص دهید.
مثال 1 : این مثال نحوه ایجاد یک فیلد از نوع Identity را نمایش می‌دهد.

```
USE tempdb
GO
CREATE TABLE Customers1
(
 ID INT IDENTITY,-- ID INT IDENTITY(1,1)
 Name NVARCHAR(100),
 [Address] NVARCHAR(200)
)
GO
```

همانطور که در مثال 1 مشاهده می‌کنید فیلد ID از نوع Identity تعریف شده است. در این حالت (ID int IDENTITY) مقدار شروع و گام افزایش به ازای این فیلد 1 در نظر گرفته خواهد شد. در این صورت اگر چند رکورد زیر را به ازای این جدول درج کنید. مقدار Identity به صورت زیر خواهد بود.

```
INSERT INTO Customers1 (Name,[Address]) VALUES
(N'مسعود',N'میانه'),
(N'فرید',N'میانه'),
(N'احمد',N'میانه')
GO
SELECT * FROM Customers1
```

Results | Messages

	ID	Name	Address
1	1	مسعود	میانه
2	2	فرید	میانه
3	3	احمد	میانه

مثال 2 : این مثال نحوه ایجاد یک فیلد از نوع Identity به همراه مقدار شروع و گام افزایش را مشخص می‌کند.

```
USE tempdb
GO
CREATE TABLE Customers2
(
ID INT IDENTITY(100,2),
Name NVARCHAR(100),
```

```
[Address] NVARCHAR(200)
)
GO
```

همانطور که در مثال 2 مشاهده می‌کنید فیلد ID از نوع Identity تعریف شده است و مقدار شروع آن از 100 و همچنین گام افزایش 2 در نظر گرفته شده است. در این صورت اگر چند رکورد زیر را به ازای این جدول درج کنید. مقدار Identity به صورت زیر خواهد بود.

```
INSERT INTO Customers2 (Name,[Address]) VALUES
(N'مسعود',N'میانه'),
(N'فرید',N'میانه'),
(N'احمد',N'میانه')
GO
SELECT * FROM Customers2
```

Results | Messages

	ID	Name	Address
1	100	مسعود	میانه
2	102	فرید	میانه
3	104	احمد	میانه

مثال 3 : این مثال نحوه تنظیم یک فیلد به صورت Identity را در محیط SSMS (SQL Server Management Studio) آموزش می‌دهد.

1- برای شروع کار همانند تصویر زیر بر روی قسمت Table کلیک راست کنید و گزینه New Table... را انتخاب کنید.

2- پس از نمایش پنجره زیر فیلدی را که می‌خواهید از نوع Identity باشد را انتخاب کرده و در قسمت Column Properties

خصیصه Is Identity را برابر Yes قرار دهید تا فیلد مورد نظر شما از نوع Identity در نظر گرفته شود. لازم به ذکر است که Identity Seed مقدار شروع و Identity Increment گام افزایش را مشخص می‌نماید.

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۳/۲۱ ۱۷:۱

آیا فیلد Identity به خودی خود زمانیکه primary key و unique نباشه (مانند مثال‌های بالا) ارزش و کاربردی داره؟

نویسنده: فرید طاهری
تاریخ: ۱۳۹۲/۰۳/۲۲ ۹:۲۳

این موضوع بستگی به سناریو شما داره
اما معمولا در بیشتر مواقع Identity را به شکل Unique در نظر می‌گیرند ذکر این نکته هم ضروری است که
1- در SQL Server معمولا Primary Key بوسیله یک Unique Clustered Index هندل می‌شود
(هر چند می‌شود اون رو به صورت یک Unique Non Clustered Index در نظر گرفت)
2- Clustered Index ترتیب و چینش فیزیکی رکوردها را مشخص می‌کند یعنی اگر Identity به عنوان کلاستر ایندکس باشد چینش و ترتیب فیزیکی رکوردها بر اساس Identity خواهد بود (سطح leaf Level مربوط به ایندکس که در کلاستر ایندکس همان Data Level است)

عنوان: **Identity و مباحث مربوطه (قسمت دوم) نحوه بدست آوردن مقادیر Identity**
نویسنده: فرید طاهری
تاریخ: ۱۳۹۲/۰۳/۲۵ ۱۷:۲۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, T-SQL, Identity

همانگونه که می‌دانید مقدار Identity پس از درج به آن تخصیص می‌یابد چنانچه بخواهید به این مقدار دسترسی پیدا کنید چندین روش به ازای اینکار وجود دارد که ما در این مقاله سه روش معمول را بررسی خواهیم نمود.

1- استفاده از متغییر سیستمی @@Identity

2- استفاده از تابع Scope_Identity ()

3- استفاده از تابع Ident_Current

هر سه این توابع مقدار Identity ایجاد شده برای جداول را نمایش می‌دهند. اما تفاوت هایی باهم دارند که در ادامه مقاله این تفاوت‌ها بررسی شده است.

1- متغییر سیستمی @@Identity : این متغییر سیستمی حاوی آخرین Identity ایجاد شده به ازای Session جاری شما است. لازم به ذکر است اگر به واسته Insert شما، Identity دیگری در یک حوزه دیگر (مانند یک Trigger) ایجاد شود مقدار موجود در این متغییر حاوی آخرین Identity ایجاد شده است. (یعنی Identity ایجاد شده توسط آن تریگر و نه خود جدول). لازم به ذکر است این موضوع به طور کامل در ادامه مقاله شرح داده شده است.

2- استفاده از تابع() Scope_Identity : با استفاده از این تابع می‌توانیم آخرین Identify ایجا دشده به ازای Session جاری را بدست آوریم. لازم به ذکر است مقادیر Identity ایجاد شده توسط سایر حوزه‌ها تاثیر در مقدار بازگشتی توسط این تابع ندارد. در ادامه مقاله این موضوع به طور کامل بررسی شده است.

3- استفاده از تابع ident_Current : این تابع آخرین مقدار Identity موجود در یک جدول را نمایش می‌دهد. ذکر این نکته ضروری است که Identity ایجاد شده توسط سایر Sessionها هم روی خروجی این تابع تاثیرگذار است. چون این تابع آخرین Identity موجود در جدول را به شما نمایش می‌دهد و نه Identity ایجاد شده به ازای یکSession را.

برای بدست آوردن یک Identity کافی است که پس از درج رکورد در جدول مورد نظر متغییر سیستمی @@Identity و یا توابع Scope_Identity و یا Ident_Current را همانند مثال زیر Select کنید.

```
USE TEMPDB
GO
IF OBJECT_ID(N'Employees', N'U') IS NOT NULL
    DROP TABLE Employees1;
GO
CREATE TABLE Employees
(
ID int IDENTITY,
FirstName NVARCHAR(50),
LastName NVARCHAR(50)
)
GO
INSERT INTO Employees (FirstName,LastName) VALUES (N'مسعود',N'طاهری')
GO
SELECT @@IDENTITY AS [@@IDENTITY]
SELECT SCOPE_IDENTITY() AS [SCOPE_IDENTITY()]
SELECT IDENT_CURRENT('Employees1') AS [IDENT_CURRENT('Employees1')]
GO
```

خروجی دستورات بالا پس از درج رکورد مورد نظر به صورت زیر است.

اما ممکن است از خودتان این سوال را بپرسید که آیا این توابع در سطح شبکه آخرین مقدار Identity درج شده توسط سایر Sessionها را نمایش می‌دهند و یا Session جاری را؟ (منظور Sessionی که درخواست مقدار موجود در identity را نموده است).

برای دریافت پاسخ این سوال مطابق مراحل اسکریپ‌های زیر را اجرا نمایید.

1-ایجاد جدول Employees1

```
USE TEMPDB GO
IF OBJECT_ID(N'Employees1', N'U') IS NOT NULL
    DROP TABLE Employees1;
GO
CREATE TABLE Employees1
(
ID int IDENTITY(1,1),
FirstName NVARCHAR(50),
LastName NVARCHAR(50)
)
GO
```

همانطور که مشاهده می‌کنید مقدار شروع برای Identity برابر 1 و گام افزایش هم برابر 1 در نظر گرفته شده است(Identity(1,1)) .

2-در Sessionی جدید دستورات زیر را اجرا نمایید. (درج رکورد جدید در جدول Employees1 و واکشی مقدار Identity)

USE tempdb

```
GO
INSERT INTO Employees1(FirstName,LastName) VALUES (N'فرید',N'طاهری')
GO
SELECT @@IDENTITY AS [@@IDENTITY]
SELECT SCOPE_IDENTITY() AS [SCOPE_IDENTITY()]
SELECT IDENT_CURRENT('Employees1') AS [IDENT_CURRENT('Employees1')]
GO
```

SPID مربوط به
Connection جاری

SQLQuery2.sql - (lo...Administrator (55))* | SQLQuery1.sql - (lo...Administrator (53))* | Ider

```
USE tempdb
GO
INSERT INTO Employees1(FirstName,LastName) VALUES (N'فرید',N'ظاهری'
GO
SELECT @@IDENTITY AS [@@IDENTITY]
SELECT SCOPE_IDENTITY() AS [SCOPE_IDENTITY()]
SELECT IDENT_CURRENT('Employees1') AS [IDENT_CURRENT('Employees1')]
GO
```

100 %

Results | Messages

	@@IDENTITY
1	1

	SCOPE_IDENTITY()
1	1

	IDENT_CURRENT('Employees1')
1	1

همانگونه که ملاحظه می‌کنید @@Scope_Identity ، Identity() و Ident_Current هر سه مقدار Identity (عدد 1) ایجاد شده بوسیله دستور Insert را به شما نمایش می‌دهند.

1- و در انتها در یک Session دیگر دستورات زیر را اجرا نمایید. (واکشی مقدار Identity)

```
USE tempdb
GO
SELECT @@IDENTITY AS [@@IDENTITY]
SELECT SCOPE_IDENTITY() AS [SCOPE_IDENTITY()]
SELECT IDENT_CURRENT('Employees1') AS [IDENT_CURRENT('Employees1')]
GO
```

همانطور که مشاهده می‌کنید در این Seesion ما از SQL خواسته‌ایم آخرین مقدار Identity را به ما نشان داده شود. باید به این نکته توجه کنید با توجه به اینکه در این Session عملیات درجی هنوز انجام نگرفته است که ما Identity ایجاد شده را مشاهده نماییم. بنابراین صرفاً تابع Iden_Current مقدار Identity موجود در جدول را به ما نمایش می‌دهد.

پس می‌توان به این نکته رسید که

Idnetity@@ و Identity : **Scope_Identity** ایجاد به ازای Session جاری را نمایش داده و به مقادیر تولید شده توسط سایر Session های دیگر دسترسی ندارد.

Ident_Current : آخرین Identity موجود در جدول را به شما نمایش می‌دهد. بنابراین باید این نکته را در نظر داشته باشید که Identity ها ایجاد شده توسط سایر Session ها روی مقدار بازگشتی این تابع تاثیرگذار است.

اما یکی دیگر از مباحث مهم درباره Identity تاثیر Scope بر مقدار Identity است (یعنی چه!) . برای اینکه با مفهوم این موضوع آشنا شوید اسکریپت‌های مربوط به مثال زیر را بدقت اجرا کنید.

1- ایجاد جدول Employees1

```
USE TEMPDB
GO
IF OBJECT_ID(N'Employees1', N'U') IS NOT NULL
    DROP TABLE Employees1;
GO
CREATE TABLE Employees1
(
ID int IDENTITY(1,1),
FirstName NVARCHAR(50),
LastName NVARCHAR(50)
)
GO
```

همانطور که مشاهده می‌کنید مقدار شروع برای Identity برابر 1 و گام افزایش هم برابر 1 در نظر گرفته شده است(Identity(1,1)) .

2- ایجاد جدول Employees2

```
USE TEMPDB
GO
IF OBJECT_ID(N'Employees2', N'U') IS NOT NULL
    DROP TABLE Employees2;
GO
CREATE TABLE Employees2
(
ID int IDENTITY(100,1),
FirstName NVARCHAR(50),
LastName NVARCHAR(50)
)
GO
```

همانطور که مشاهده می‌کنید مقدار شروع برای Identity برابر 100 و گام افزایش هم برابر 1 در نظر گرفته شده است(Identity(100,1)).

3- ایجاد یک Trigger به ازای جدول Employees1

```
USE tempdb
GO
CREATE TRIGGER Employees1_Insert ON Employees1 FOR INSERT
AS
BEGIN
   INSERT Employees2(FirstName,LastName)
SELECT FirstName,LastName FROM INSERTED
END;
GO
```

Trigger ایجاد شده به ازای جدول Employees1 به ازای عملیات Insert اجرا می‌شود. همچنین مقادیر درج شده در جدول Employees1 بوسیله جدول Inserted در دسترس است. لازم به ذکر است جدول Inserted یک جدول موقت بوده که توسط Trigger ایجاد شده و داخل خود آن معتبر است.

هدف ما از ایجاد این Trigger تهیه یک کپی از رکوردهایی که در جدول Employees1 درج می‌شوند است. این کپی قرار است با استفاده از دستور Insert...Select در جدول Employees2 ایجاد گردد.

4- درج یک رکورد در جدول Employees1 و واکشی مقدار Identity

```
USE tempdb
GO
INSERT INTO Employees1(FirstName,LastName) VALUES (N'مسعود',N'طاهری')
GO
SELECT @@IDENTITY AS [@@IDENTITY]
SELECT SCOPE_IDENTITY() AS [SCOPE_IDENTITY()]
SELECT IDENT_CURRENT('Employees1') AS [IDENT_CURRENT('Employees1')]
SELECT IDENT_CURRENT('Employees2') AS [IDENT_CURRENT('Employees2')]
GO
```

Results | Messages

	@@IDENTITY
1	100

	SCOPE_IDENTITY()
1	1

	IDENT_CURRENT('Employees1')
1	1

	IDENT_CURRENT('Employees2')
1	100

مقادیر استخراج شده به ازای Identity به شرح زیر است

1- @@Identity : پس از درج رکورد در جدول Employees1 متغییر سیستمی @@Identity مقدار 100 را نمایش داده است دلیل این موضوع بر می‌گردد به Trigger موجود در جدول Employees1 .

با توجه به اینکه جدول Employees1 دارای یک فیلد Identity بوده است هنگام درج رکورد در جدول مقدار @@Identity=1 است اما چون این جدول دارای Trigger ی است که این Trigger خود با جدولی دیگری درگیر است که دارای Identity است مقدار متغییر @@identity=100 خواهد شد.

2- Scope_Identity() : مقدار نمایش داده شده توسط تابع Scope_Identity() برابر با مقدار Identity تخصیص (عدد 1) داده شده به ازای رکورد شما می‌باشد که این موضوع در اغلب موارد مد نظر برنامه‌نویسان می‌باشد.

3- Ident_Current('Employees1') : مقدار نمایش شده توسط تابع Ident_Current آخرین مقدار Identity (عدد 1) موجود در جدول Employees1 است.

4- Ident_Current('Employees2') : مقدار نمایش شده توسط تابع Ident_Current آخرین مقدار Identity (عدد 100) موجود در جدول Employees2 است.

چند نکته مهم

1- مقدار بازگردانده شده توسط تابع Ident_Current آخرین مقدار Identity موجود در جدول مورد نظر شما بوده است و عملیات درج سایر کاربران در این مقدار تاثیر گذار است.

2- برای بدست آوردن مقدار Identity درست بهتر است از تابع Scope_Identity() استفاده نماییم. معمولاً در بیشتر مواقع مقدار بازگردانده شده توسط این تابع مد نظر برنامه نویسان است.

3- EntityFramework و Nhibernate هم برای بدست آوردن Identity از تابع Scope_Identity استفاده می‌کند.

نظرات خوانندگان

نویسنده: کاربر
تاریخ: ۱۳۹۲/۰۴/۳۱ ۴:۵۰

ببین دوست من مطلبتون رو خوندم هم اینو و هم قبلی رو، ازش خوشم اومد اما چیزی راجب درج صریح یا بروز رسانی مقادیر Identity ننوشته بودین. یا اینکه نمیشه در یک جدول دو identity property داشت.
من بلدم با set identity_insert table_name on/off کاری کنم که خودم دستی مقداری را برای خصیصه identity لحاظ کنم. ولی متاسفانه نتونستم مقدار یک ستون با خصیصه Identity رو بروز رسانی (یا همون update) کنم. لطفا بهم بگید که اصلا این کار ممکنه یا من بلد نیستم. البته براساس query زیر بمن SQL Server گفته که نمیشه این ستون را update کرد که ظاهرا هم همین طور(ستون id همانطور که در پیام آمده از نوع identity هست)

```
update t
set id = new_id
from (select id, row_number() over(order by id) new_id from #temp)t

--Cannot update identity column 'id'.
```

اصلا اجازه بدین یه جور دیگه سوال رو مطرح کنم من نیاز دارم تمام مقادیر identity رو بروز رسانی کنم تا کاملا پشت سر هم و متوالی بشن این کار را میتونم با یک تابع row_number و یک derived table انجام بدم (اگر بذارن!) همانطور که قبلا نشان دادم، یا با روش زیر این کار را بکنم که البته اجرا نمیشه به این دلیل که در یک جدول نمیشه دو identity property داشت. با فرض اجرا شدن دستور select into باز هم در دستور update با مشکل بر می‌خوردیم (چون نمیشه ستون id را بروز رسانی کرد)

```
select id, identity(int, 1,1) new_id
into #temptable
from #temp
order by id asc

/*
cannot add identity column, using the SELECT INTO statement, to table '#temptable',
 which already has column 'id' that inherits the identity property.
*/
update t
set id = new_id
from #temp t
join #temptable d
on t.id = d.id;
```

البته یک راهی برای حل این مساله هست اونم اینه که ابتدا بیاییم تمام داده‌ها جدول را در جدول دیگه ای درج کنیم سپس تمام داده‌های جدول را حذف کنیم سپس داده‌های حذف شده را با id جدید و مرتب شده در جدول اول درج کنیم. به این شکل

```
declare @t table(id int)

insert into @t
select id from #temp

delete from #temp

set identity_insert #temp on
insert #temp (id)
select row_number() over(order by id) from @t
set identity_insert #temp off
```

اما مشکلی که وجود داره اینه که اگر جدول ما parent باشه با مشکل واجه میشیم تمام سطرهای جداول child یتیم میشن.

من قصد ندارم صورت مساله نقد و بررسی بشه و اصولی بودن یا صحیح بودنش مورد ارزیابی قرار بگیره فقط برام این یک سوال شده.

مساله عمومی که راجب این ستون وجود داره استفاده کردن از Gapهای حاصل شده در این ستون برای درج‌های بعدی است. که query آن نیز بسیار ساده و در دسترس است.
آیا شما میدانید که چگونه این مشکل با sequence ای که در نسخه 2012 معرفی شده است حل می‌شود؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۳۱ ۱۴:۵۵

- خیر. چندین نوع استراتژی برای تعیین PK وجود دارند که یکی از آن‌ها فیلدهای Identity است و این تنها روش و الزاما بهترین روش نیست.
- مثلا زمانیکه با ORMها کار می‌کنید استفاده از فیلدهای Identity در حین ثبت تعداد بالایی از رکوردها مشکل ساز می‌شوند. چون این فیلدها تحت کنترل دیتابیس هستند و نه برنامه، ORM نیاز دارد پس از هربار Insert یکبار آخرین Id را از بانک اطلاعاتی واکشی کند. همین مساله یعنی افت سرعت در تعداد بالای Insertها (چون یکبار کوئری Insert باید ارسال شود و یکبار هم یک Select اضافی دوم برای دریافت Id تولیدی توسط دیتابیس).
- روش دوم تعیین PK استفاده از نوع Guid است. در این حالت، هم مشکل حذف رکوردها و خالی شدن یک شماره را در این بین ندارید و هم چون عموما تحت کنترل برنامه است، سرعت کار کردن با آن بالاتر است. فقط تنها مشکل آن زیبا نبودنش است در مقایسه با یک عدد ساده فیلدهای Identity.

در مورد فیلدهای Identity، تغییر شماره Id به صلاح نیست چون:
الف) همانطور که عنوان کردید روابط بین جداول را به هم خواهد ریخت.
ب) در یک وب سایت و یا هر برنامه‌ای، کلا آدرس‌ها و ارجاعات قدیمی را از بین می‌برد. مثلا فرض کنید شماره این مطلب 1381 است و شما آن‌را یادداشت کرده‌اید. در روزی بعد، برنامه نویس شماره Idها را کلا ریست کرده. در نتیجه یک هفته بعد شما به شماره 1381 ایی خواهید رسید که تطابقی با مطلب مدنظر شما ندارد (حالا فرض کنید که این عدد شماره پرونده یک شخص بوده یا شماره کاربری او و نتایج و خسارات حاصل را درنظر بگیرید).
ج) این خوب است که در بین اطلاعات یک ردیف خالی وجود دارد. چون بر این اساس می‌توان بررسی کرد که آیا واقعا رکوردی حذف شده یا خیر. گاهی از اوقات کاربران ادعا می‌کنند که اطلاعات ارسالی آن‌ها نیست در حالیکه نبود این رکوردها به دلیل حذف بوده و نه عدم ثبت آن‌ها. با بررسی این Idها می‌شود با کاربران در این مورد بحث کرد و پاسخ مناسبی را ارائه داد.

و اگر شماره‌ای که به کاربر نمایش می‌دهید فقط یک شماره ردیف است (و از این لحاظ می‌خواهید که حتما پشت سرهم باشد)، بهتر است یک View جدید ایجاد کنید تا این Id خود افزاینده را تولید کند (بدون استفاده از pk جدول).

پ.ن.
هدف من از این توضیحات صرفا عنوان این بود که به PK به شکل یک فیلد read only نگاه کنید. این دقیقا برخوردی است که Entity framework با این مفهوم دارد و صحیح است و اصولی. اگر در یک کشور هر روزه عده‌ای به رحمت ایزدی می‌روند به این معنا نیست که سازمان ثبت احوال باید شماره شناسنامه‌ها را هر ماه ریست کند!

نویسنده: فرید طاهری
تاریخ: ۱۳۹۲/۰۴/۳۱ ۲۰:۰

با تشکر از آقای نصیری و پاسخ مناسبی که ارائه کرده اند
در مورد استفاده از GUID به جای identity باید به یک نکته هم اشاره کنم که در بیشتر مواقع اگر مقدار GUIDی که به ازای یک فیلد UNIQUEIDENTIFIER تنظیم می‌کنید به صورت SEQUNTIAL نباشد باعث Fragment شدن ایندکس خواهد شد.
برای مقایسه بهتر بین Fragmentation ایندکس مربوط به Identity و GUID به مثال زیر دقت کنید. هر دو مثال فیلد ID خود را به شکل Clustered Index دارند بعد از درج تعدادی رکورد مساوی در دو جدول Fragmentation مربوط به جدولی که دارای GUID است به شدت بالا است که این موضوع باعث کاهش کارایی خواهد شد

```
USE TEMPDB
GO
```

```
IF OBJECT_ID('TABLE_GUID')>0
DROP TABLE TABLE_GUID
GO
CREATE TABLE TABLE_GUID
(
ID UNIQUEIDENTIFIER PRIMARY KEY,
FirstName NVARCHAR(1000),
LastName NVARCHAR(1000)
)
GO
IF OBJECT_ID('TABLE_IDENTITY')>0
DROP TABLE TABLE_IDENTITY
GO
CREATE TABLE TABLE_IDENTITY
(
ID INT IDENTITY PRIMARY KEY,
FirstName NVARCHAR(1000),
LastName NVARCHAR(1000)
)
GO
INSERT INTO TABLE_GUID(ID,FirstName,LastName) VALUES
(NEWID(),REPLICATE('FARID*',100),REPLICATE('Taheri*',100))
GO 10000

INSERT INTO TABLE_IDENTITY(FirstName,LastName) VALUES
(REPLICATE('FARID*',100),REPLICATE('Taheri*',100))
GO 10000

--Fragmentation بررسی وضعیت
SELECT * FROM sys.dm_db_index_physical_stats(DB_ID(),OBJECT_ID('TABLE_GUID'),NULL,NULL,'DETAILED')
DBCC SHOWCONTIG(TABLE_GUID)
GO
SELECT * FROM sys.dm_db_index_physical_stats(DB_ID(),OBJECT_ID('TABLE_IDENTITY'),NULL,NULL,'DETAILED')
DBCC SHOWCONTIG(TABLE_IDENTITY)
GO
```

خوب برای اینکه Fragmentation این نوع جداول را رفع کنید چند راه داریم

1- تولید GUID به صورت Sequential (لازم می‌دانم اشاره کنم این قابلیت در SQL Server وجود دارد ولی مقدار تولید شده باید به شکل یک Default Constraint باشد که این موضوع نیازمند این است که شما اگر در سورس به این GUID نیاز پیدا کنید مجبور به زدن Select و... شوید. اگر بخواهید در سورس این کار را انجام دهید باید از Extentionهایی که برای اینکار وجود دارند استفاده کنید فکر کنم Nhibernate این حالت رو پشتیبانی کنه در مورد EF دقیقا اطلاع ندارم باید اهل فن نظر بدن)

2- تنظیم مقدار Fillfactor به ازای ایندکس

3-Rebuild و یا Reorganize دوره ای ایندکس

نویسنده: برنامه نویس
تاریخ: ۱۳۹۳/۰۷/۰۴ ۱۹:۱

باسلام؛ اگر همزمان 10 کاربر در شبکه بخوان درج کنن و هر بار identity درست بعد از درج نشان داده شود تابع Scope_Identity درست عمل میکند؟ تداخل به وجود نمی‌آید؟

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۷/۰۵ ۱۳:۳۲

در مطلب بررسی Transactions و Locks در SQL Server پاسخ خود را خواهید یافت.

عنوان: **ویدئوی آموزش مقدمات CodeFirst در قالب یک کلاس آموزشی به همراه مثال**
نویسنده: سیدمجتبی حسینی
تاریخ: ۱۳۹۲/۰۴/۰۳ ۱:۲۰
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, SQL Server, EF Code First

این ویدئو
به مدت حدوداً یکساعت و حجمی حدود 50 مگابایت، مربوط به یک کلاس آموزشی است که در ضمن آن به بررسی مقدماتی چگونگی بکاربردن روش CodeFirst برای تولید دیتابیس و جداول آن، پرداخته‌ام. پیشاپیش از نواقص و نارسایی‌های احتمالی آن، پوزش می‌طلبم.

سرفصل مطالبی که بطور مختصر مطرح شده‌اند، عبارتند از:

معرفی EF CodeFirst و کاربرد آن

استفاده از Nuget Package Manager برای افزودن EntityFrameWork

ایجاد کلاس نمونه User و معرفی DbContext جهت معرفی کلاس User به عنوان جدولی از دیتابیس

ایجاد ConnectionString و نکات مربوط به آن برای ایجاد صحیح جداول در SQL Server

چگونگی ایجاد فیلد کلیدی

روش ذخیره سازی اطلاعات در جدول

روش ایجاد رابطه یک به چند با ایجاد دو جدول کمکی Log و Work و مرتبط با جدول User

روش جستجو در جداول بدون استفاده مستقیم از SQL Query

نظرات خوانندگان

نویسنده: صادق
تاریخ: ۱۳۹۲/۰۴/۰۶ ۱۷:۵۶

با سلام و عرض ادب
می‌خواستم بدونم ادامه این ویدئو را چطوری می‌تونم تهیه کنم . با تشکر

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۴/۰۶ ۱۸:۳۱

روی لینک «این ویدیو» در ابتدای مطلب کلیک کنید.

نویسنده: sorosh
تاریخ: ۱۳۹۲/۰۴/۰۶ ۱۸:۴۹

با سلام؛ می‌خواستم بدونم ویدئو دیگه‌ای آیا این مجموعه داره یا همینه فقط ؟ مرسی

نویسنده: سیدمجتبی حسینی
تاریخ: ۱۳۹۲/۰۴/۰۷ ۱۳:۰

سلام و درود.
در بینابین جلسات آموزش سی شارپ پیشرفته به جهت تبیین یکی از کاربردهای نوین کدنویسی شیءگرا و همچنین تقویت انگیزه یادگیرندگان در یادگیری همه مطالب ، چه انتزاعی و چه کاربردی، به طرح این بحث فقط در یک جلسه اکتفا کردم.
باتشکر

نویسنده: آگاه
تاریخ: ۱۳۹۲/۱۱/۲۱ ۱۵:۳۶

نمیشه آموزش ویدیویی این بحث رو ادامه بدین؟

خیلی شیوا کارامد و مفید بود.. مخصوصا اینکه کلی جزئیات هم بینش گفته میشد.. :(

عنوان: مفهوم READ_COMMITTED_SNAPSHOT در EF 6
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۱۴ ۸:۰
آدرس: www.dotnettips.info
برچسب‌ها: Entity framework, SQL Server

مدتی است که حالت READ_COMMITTED_SNAPSHOT بسیار مورد توجه واقع شده:
- در سایت Stack overflow از آن استفاده می‌شود (^).
- در SQL Server Azure حالت پیش فرض ایجاد دیتابیس‌ها و تراکنش‌های جدید است (^).
- در Entity framework 6 حالت پیش فرض تراکنش‌های ایجاد شده، قرار گرفته است (^).

و ... در Oracle، تنها حالت مدیریت مسایل همزمانی است! (البته به نام MVCC ، اما با همین عملکرد)

اما READ_COMMITTED_SNAPSHOT در SQL Server چیست و کاربرد آن کجا است؟

اگر استفاده گسترده و سنگینی از SQL Server داشته باشید، حتما به پیغام‌های خطای deadlock آن برخورده‌اید:

```
Transaction (Process ID 54) was deadlocked on lock resources with another process and has been chosen
as the deadlock victim.
Rerun the transaction.
```

روش پیش فرض مدیریت مسایل همزمانی در SQL Server، حالت READ COMMITTED است. به این معنا که اگر در طی یک تراکنش مشغول به تغییر اطلاعاتی باشیم، سایر کاربران از خواندن نتیجه آن (اصطلاحا به آن Dirty read گفته می‌شود) منع خواهند شد؛ تا زمانیکه این تراکنش با موفقیت به پایان برسد. هرچند در این حالت سایر تراکنش‌ها امکان ویرایش یا حذف اطلاعات را خواهند داشت. به علاوه اگر در طی این تراکنش، اطلاعاتی خوانده شوند، سایر تراکنش‌ها تا پایان تراکنش جاری، قادر به تغییر این اطلاعات خوانده شده نخواهند بود (منشاء بروز خطاهای deadlock یاد شده در سیستم‌های پرترافیک).
در SQL Server 2005 برای بهبود مقیاس پذیری SQL Server و کاهش خطاهای deadlock، مکانیزم READ_COMMITTED_SNAPSHOT معرفی گشت.
به صورت خلاصه زمانیکه که تراکنش مورد نظر تحت حالت READ COMMITTED SNAPSHOT انجام می‌شود، optimistic reads and pessimistic writes خواهیم داشت (خواندن‌های خوشبینانه و نوشتن‌های بدبینانه). در این حالت تضمین می‌شود که خواندن اطلاعات داخل یک تراکنش، شامل اطلاعات تغییر یافته توسط سایر تراکنش‌های همزمان نخواهد بود. همچنین زمانیکه در این بین، اطلاعاتی خوانده می‌شود، بر روی این اطلاعات برخلاف حالت READ COMMITTED قفل قرار داده نمی‌شود. بنابراین تراکنش‌هایی که درحال خواندن اطلاعات هستند، تراکنش‌های همزمانی را که در حال نوشتن اطلاعات می‌باشند، قفل نخواهد کرد و برعکس.

نحوه فعال سازی READ_COMMITTED_SNAPSHOT

فعال سازی READ_COMMITTED_SNAPSHOT باید ابتدا در سطح یک بانک اطلاعاتی SQL Server انجام شود:

```
ALTER DATABASE testDatabase SET ALLOW_SNAPSHOT_ISOLATION ON;
ALTER DATABASE testDatabase SET READ_COMMITTED_SNAPSHOT ON;
```

کاری که در اینجا انجام خواهد شد، ایجاد یک snapshot یا یک کپی فقط خواندنی، از بانک اطلاعاتی کاری شما می‌باشد. بنابراین در این حالت، زمانیکه یک عبارت Select را فراخوانی می‌کنید، این خواندن، از بانک اطلاعاتی فقط خواندنی تشکیل شده، صورت خواهد گرفت. اما تغییرات بر روی دیتابیس اصلی کاری درج شده و سپس این snapshot به روز می‌شود.
حالت READ_COMMITTED_SNAPSHOT خصوصا برای برنامه‌های وبی که تعداد بالایی Read در مقابل تعداد کمی Write دارند، به شدت بر روی کارآیی و بالا رفتن سرعت و مقیاس پذیری آن‌ها تاثیر خواهد داشت؛ به همراه حداقل تعداد deadlockهای حاصل شده.

در Entity framework وضعیت به چه صورتی است؟

EF از حالت پیش فرض مدیریت مسایل همزمانی در SQL Server یا همان حالت READ COMMITTED در زمان فراخوانی متد SaveChanges استفاده می‌کند.

در EF 6 این حالت پیش فرض به READ_COMMITTED_SNAPSHOT تغییر کرده است. البته همانطور که عنوان شد، پیشتر باید بانک اطلاعاتی را نیز جهت پذیرش این نوع تراکنش‌ها آماده ساخت.

اگر از نگارش‌های پایین‌تر از EF 6 استفاده می‌کنید، برای استفاده از حالت READ_COMMITTED_SNAPSHOT باید صراحتا IsolationLevel را مشخص ساخت:

```
using (var transactionScope =
  new TransactionScope(TransactionScopeOption.Required,
  new TransactionOptions { IsolationLevel= IsolationLevel.Snapshot }))
{
   // update some tables using entity framework
   context.SaveChanges();
   transactionScope.Complete();
}
```

نظرات خوانندگان

نویسنده: ali.rezayee
تاریخ: ۱۳۹۲/۰۴/۱۴ ۱۵:۴۱

با سلام و تشکر بخاطر این مطلب عالی.
امکان دارد در خصوص بخش « READ_COMMITTED_SNAPSHOT در SQL Server چیست و کاربرد آن کجا است؟ » یک مثال عملی بزنید، مطلب کمی گنگ است.
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۱۴ ۱۸:۱۳

فرض کنید یک جدول نظرات دارید با این تعریف

```
CREATE TABLE [BlogComments](
[Id] [int] IDENTITY(1,1) NOT NULL,
[Body] [nvarchar](max) NULL,
[Date1] [datetime] NOT NULL,
 CONSTRAINT [PK_BlogComments] PRIMARY KEY CLUSTERED
(
[Id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,
 ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

بعد در management studio دو پنجره اجرای کوئری جدید را ایجاد کنید. در پنجره اول بنویسید:

```
-- در پنجره اول
BEGIN TRAN
UPDATE [BlogComments] SET Body='Test' WHERE id=1
```

در پنجره دوم بنویسید

```
-- در پنجره دوم
SELECT TOP 1000 [Id]  ,[Body] ,[Date1] FROM [BlogComments]
```

- ابتدا عبارت پنجره اول را اجرا کنید. این پنجره حاوی یک تراکنش نا تمام است. شروع دارد اما به عمد پایان آن‌را ذکر نکردیم.
- الان پنجره دوم را اجرا کنید.

مشاهده خواهید کرد که ... به جواب نمی‌رسید. کوئری اجرا نمی‌شود و سیستم قفل شده چون تراکنش اول commit نشده (مثلا یک تراکنش طولانی را اینجا شبیه سازی کردیم؛ یا حتی یک اشتباه در تعاریف T-SQL انجام شده).

در ادامه، عملیات این پنجره‌ها را دستی متوقف کنید. بعد مطابق دستوراتی که پیشتر ذکر شد، READ_COMMITTED_SNAPSHOT را روی دیتابیس فعال کنید.
مجددا دو مرحله قبل را اجرا کنید. در این حالت کوئری دوم اجرا خواهد شد، چون اطلاعات را از کپی فقط خواندنی بانک اطلاعاتی شما دریافت می‌کند؛ بر اساس آخرین اطلاعات commit شده در سیستم.

نویسنده: مرادی
تاریخ: ۱۳۹۲/۰۴/۱۵ ۱۰:۵۳

با سلام، در قسمتی از مطلبتان، آورده اید " یک snapshot یا یک کپی فقط خواندنی، "

که این پیش فرض اشتباه را در ذهن خواننده ایجاد می‌کند که از دیتابیس یک کپی گرفته می‌شود، در حالی که از دیتابیس کپی

گرفته نمی‌شود، بلکه Snapshot حاوی تغییرات دیتابیس از لحظه ایجاد Snapshot تا به حال است، مثلا می‌گوید در جدول People، سه رکورد درج شده است، از تفاضل دیتابیس و این Snapshot می‌توان به وضعیت دیتابیس قبل از ایجاد Snapshot پی برد، به همین دلیل است که ایجاد یک Snapshot ولو روی یک دیتابیس چند گیگابایتی نیز در کسری از ثانیه انجام می‌پذیرد.
علاوه بر این امکان استفاده از این امکان در SQL 2000 نیست، ولو با ADO.NET، که البته چیزی رو از ارزش‌های این روش کم نمی‌کند، با سپاس از مطلب خوبتان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۱۵ ۱۱:۴۸

بحث اصلی هم همین نحوه و محل ذخیره سازی snapshot است.
- Sanpshot مطابق واژه نامه مایکروسافت معنای «نگارش» را می‌دهد. در این حالت کلیه کوئری‌های داخل یک تراکنش، یک نگارش یا snapshot از دیتابیس را مشاهده خواهند کرد. این نگارش یا Row version، در *tempdb* نگه داری می‌شود. با فعال سازی SNAPSHOT isolation، هر زمانیکه یک ردیف به روز رسانی می‌شود، موتور SQL Server یک نسخه از اطلاعات اولیه این ردیف را در tempdb ذخیره می‌کند (اینجا بود که عنوان شد با یک کپی فقط خواندنی از اطلاعات در حین واکشی اطلاعات سر و کار خواهید داشت).
خلاصه الگوریتم کاری آن :
الف) با آغاز یک تراکنش، یک عدد متوالی منحصربفرد تراکنش (شماره نگارش) ایجاد شده و به آن نسبت داده می‌شود.
ب) در حین این تراکنش، موتور SQL Server، به tempdb مراجعه کرده و شماره نگارشی نزدیک و کمتر از شماره نگارش تراکنش جاری را پیدا می‌کند. همچنین SQL Server بررسی می‌کند که این شماره یافت شده حتما جزو تراکنش‌های پایان یافته سیستم باشد.
ج) بر اساس این شماره یافت شده، نگارش معتبری از اطلاعات از tempdb استخراج می‌شود.
به این ترتیب یک تراکنش، کلیه اطلاعات موجود در ابتدای کار خود را بدون قرار دادن قفلی بر روی جداول مرتبط، دریافت خواهد کرد.
اطلاعات بیشتر

- در متن ذکر گردید که از SQL Server 2005 به بعد قابلیت فوق اضافه شده.
- همچنین SQL Server 2000 دیگر پشتیبانی رسمی ندارد و استفاده از آن حداقل از لحاظ امنیتی معقول نیست.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۲۳ ۱۲:۵

خبری در اینباره از تیم SQL Server بعدی

```
The Hekaton team also found that multi-version concurrency control (MVCC) proved robust
in scenarios with higher workloads and higher contention
```

اطلاعات بیشتر

نویسنده: amir ranjbarian
تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۸:۳۷

با سلام؛ در صورتی که بخواهیم این مورد را در دیتابیسی که از filestream استفاده میکنه فعال کنیم با این خطا ALTER DATABASE failed because the READ_COMMITTED_SNAPSHOT and the ALLOW_SNAPSHOT_ISOLATION options cannot be set to ON when a database has FILESTREAM filegroups. To set READ_COMMITTED_SNAPSHOT or ALLOW_SNAPSHOT_ISOLATION to ON, you must remove the FILESTREAM filegroups from the database.
مواجه می‌شویم من در دیتابیس از filestreame استفاده کردم برای ذخیره فایلهای مورد نیاز در نرم افزار خودم، می‌خواستم بدونم آیا استفاده از این روش (filestreame) اصولا خوب هست یا نه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۸:۴۵

این محدودیت از نگارش **R2** اس کیوال سرور 2008 به بعد برطرف شده .

عنوان: آشنایی با SQL Server Common Table Expressions - CTE
نویسنده: عمران
تاریخ: ۱۳۹۲/۰۶/۰۷ ۱۴:۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, T-SQL

مقدمه

تکنولوژی CTE از نسخه SQL Server 2005 رسمیت یافته است و شامل یک result set موقتی[1] است که دارای نام مشخص بوده و می-توان از آن در دستورات SELECT, INSERT, UPDATE, DELETE استفاده کرد. همچنین از CTE می‌توان در دستور CREATE VIEW و دستور SELECT مربوط به آن استفاده کرد. در نسخه SQL Server 2008 نیز امکان استفاده از CTE در دستور MERGE فراهم شده است.

در SQL Serverاز دو نوع CTE بازگشتی[2] و غیر بازگشتی[3] پشتیبانی می-شود. در این مقاله سعی شده است نحوه تعریف و استفاده از هر دو نوع آن آموزش داده شود.

انواع روش-های ایجاد جداول موقت

برای استفاده از جداول موقتی در سرور اسکیوال، سه راه زیر وجود دارد.

روش اول: استفاده از دستوری مانند زیر است که سبب ایجاد جدول موقتی در بانک سیستمی tempdb می-شود. زمانی-که شما ارتباط خود را با سرور SQL قطع می-کنید به صورت اتوماتیک جداول موقت شما از بانک tempdb حذف می-شوند. این روش در برنامه نویسی پیشنهاد نمی-شود و فقط در کارهای موقتی و آزمایشی مناسب است.

```
SELECT * INTO #temptable FROM [Northwind].[dbo].[Products]
UPDATE #temptable SET [UnitPrice] = [UnitPrice] + 10
```

روش دوم: استفاده از متغیر نوع Table است، که نمونه آن در مثال زیر دیده می-شود. زمانیکه از محدوده[4] جاری کد[5] خودتان خارج شوید آن متغیر نیز از حافظه پاک میشود. از این روش، عموما در کدهای Stored Procedureها و UserDefined Functionها استفاده می-شود.

```
DECLARE @tempTable TABLE
(
   [ProductID] [int] NOT NULL,
   [ProductName] [nvarchar](40) NOT NULL,
   [UnitPrice] [money] NULL
)

INSERT INTO @tempTable
SELECT
   [ProductID],
   [ProductName],
   [UnitPrice]
FROM [Northwind].[dbo].[Products]

UPDATE @temptable SET [UnitPrice] = [UnitPrice] + 10
```

روش سوم: استفاده از CTE است که مزیت-هایی نسبت به دو روش قبلی دارد و در بخش بعدی به نحوه تعریف و استفاده از آن خواهیم پرداخت.

کار با CTE

ساده -ترین شکل تعریف یک CTE به صورت زیر است:

```
WITH yourName [(Column1, Column2, ...)]
AS
(
```

```
   your query
)
```

با کلمه WITH شروع شده و یک نام اختیاری به آن داده می-شود. سپس فهرست فیلدهای جدول موقت را درون زوج پرانتز، مشخص می-کنید. تعریف این فیلدها اختیاری است و اگر حذف شود، فیلدهای جدول موقت، مانند فیلدهای کوئری مربوطه خواهد بود.

your query شامل دستوری است که سبب تولید یک result set می-شود. قواعد تعریف این کوئری مشابه قواعد تعریف کوئری است که در دستور CREATE VIEW کاربرد دارد.

شکل کلی دستور

همانطور که از این تصویر مشخص است می-توان چندین بلوک از این ساختار را به دنبال هم تعریف نمود که با کاما از هم جدا می-شوند. در واقع یکی از کاربردهای CTE ایجاد قطعات کوچکی است که امکان استفاده مجدد را به شما داده و می-تواند سبب خواناتر شدن کدهای پیچیده شود.
یکی دیگر از کاربردهای CTE آنجایی است که شما نمی‌خواهید یک شی Viewی عمومی تعریف کنید و در عین حال می‌خواهید از مزایای Viewها بهرمند شوید.
و همچنین از کاربردهای دیگر CTE تعریف جدول موقت و استفاده از آن جدول به صورت همزمان در یک دستور است.

بعد از آنکه CTE یا CTEهای خودتان را تعریف کردید آنگاه می-توانید مانند جداول معمولی از آنها استفاده کنید. استفاده از این جداول توسط دستوری خواهد بود که دقیقا بعد از تعریف CTE نوشته می-شود.

ایجاد یک CTE غیر بازگشتی[6]

مثال اول، یک CTE غیر بازگشتی ساده را نشان میدهد.

```
WITH temp
AS
(
    SELECT
         [ProductName],
         [UnitPrice]
    FROM [Northwind].[dbo].[Products]
)
SELECT * FROM temp
ORDER BY [UnitPrice] DESC
```

مثال دوم نمونه-ای دیگر از یک CTE غیر بازگشتی است.

```
WITH orderSales (OrderID, Total)
AS
(
   SELECT
      [OrderID],
     SUM([UnitPrice]*[Quantity]) AS Total
   FROM [Northwind].[dbo].[Order Details]
   GROUP BY [OrderID]
)
SELECT
   O.[ShipCountry],
   SUM(OS.[Total]) AS TotalSales
FROM [Northwind].[dbo].[Orders] AS O INNER JOIN [orderSales] AS OS
ON O.[OrderID] = OS.[OrderID]
GROUP BY O.[ShipCountry]
ORDER BY TotalSales DESC
```

هدف این کوئری، محاسبه کل میزان فروش کالاها، به ازای هر کشور می-باشد. ابتدا از جدول Order Details مجموع فروش هر سفارش محاسبه شده و نتیجه آن در یک CTE به نام orderSales قرار می-گیرد و از JOIN این جدول موقت با جدول Orders محاسبه نهایی انجام شده و نتیجه-ای مانند این تصویر حاصل می‌شود.

Results | Messages

	ShipCountry	TotalSales
1	USA	263566.98
2	Germany	244640.63
3	Austria	139496.63
4	Brazil	114968.48
5	France	85498.76
6	Venezuela	60814.89
7	UK	60616.51
8	Sweden	59523.70
9	Ireland	57317.39
10	Canada	55334.10
11	Belgium	35134.98
12	Denmark	34782.25

نتیجه خروجی

مثال سوم استفاده از دو CTE را به صورت همزمان نشان می-دهد:

```
WITH customerList
AS
(
   SELECT
      [CustomerID],
      [ContactName]
   FROM [Northwind].[dbo].[Customers]
   WHERE [Country] ='UK'
)
,orderList
AS
(
   SELECT
      [CustomerID],
      [OrderDate]
   FROM [Northwind].[dbo].[Orders]
   WHERE YEAR([OrderDate])< 2000
)
SELECT
   cl.[ContactName],
   YEAR(ol.[OrderDate]) AS SalesYear
FROM customerList AS cl JOIN orderList AS ol
ON cl.[CustomerID] = ol.[CustomerID]
```

مثال چهارم استفاده مجدد از یک CTE را نشان می-دهد. فرض کنید جدولی به نام digits داریم که فقط یک فیلد digit دارد و دارای 10 رکورد با مقادیر 0 تا 9 است. مانند تصویر زیر

	digit
	0
	1
	2
	3
	4
	5
	6
▶	7
	8
	9
*	NULL

نتیجه خروجی

حال می-خواهیم از طریق CROSS JOIN اعداد 1 تا 100 را با استفاده از مقادیر این جدول تولید کنیم. کد زیر آنرا نشان می-دهد:

```
WITH digitList
AS
(
   SELECT [digit] from [digits]
)
SELECT
   a.[digit] * 10 + b.[digit] + 1 AS [Digit]
FROM [digitList] AS a CROSSJOIN [digitList] AS b
```

در این کد یک CTE تعریف شده و دو بار مورد استفاده قرار گرفته است. مثلا اگر بخواهید اعداد 1 تا 1000 را تولید کنید می-توانید سه بار از آن استفاده کنید. حاصل این دستور result setی مانند زیر است.

Results | Messages

	Digit
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13

نتیجه

87	87
88	88
89	89
90	90
91	91
92	92
93	93
94	94
95	95
96	96
97	97
98	98
99	99
100	100

نتیجه

حتی می-توان از یک CTE در کوئری CTE بعدی مانند کد زیر استفاده کرد.

```
WITH CTE_1 AS
(
   ....
),
CTE_2 AS
(
   SELECT  ... FROM CTE_1 JOIN ...
)
SELECT   *
FROM FOO
LEFTJOIN  CTE_1
LEFTJOIN  CTE_2
```

ایجاد یک CTE بازگشتی[7]

از CTE بازگشتی برای پیمایش جداولی استفاده می-شود که رکوردهای آن دارای رابطه سلسله مراتبی یا درختی است. نمونه این جداول، جدول کارمندان است که مدیر هر کارمند نیز مشخص شده است یا جدولی که ساختار سازمانی را نشان می-دهد یا جدولی که موضوعات درختی را در خود ذخیره کرده است. یکی از مزایای استفاده از CTE بازگشتی، سرعت کار آن در مقایسه با روش-های پردازشی دیگر است.

ساختار کلی یک دستور CTE بازگشتی به صورت زیر است.

```
WITH cteName AS
(
   query1
   UNION ALL
   query2
)
```

در بدنه CTE حداقل دو عضو[8] (کوئری) وجود دارد که بایستی با یکی از عبارت-های زیر به هم متصل شوند.

UNION

UNION ALL

INTERSECT

EXCEPT

query1 شامل دستوری است که اولین سری از رکوردهای result set نهایی را تولید می-کند. اصطلاحا به این کوئری anchor memberمی-گویند.

بعد از دستور query1، حتما بایستی از UNION ALL و امثال آنها استفاده شود.

سپس query2 ذکر می-شود. اصطلاحا به این کوئری recursive member گفته می-شود. این کوئری شامل دستوری است که سطوح بعدی درخت را تولید خواهد کرد. این کوئری دارای شرایط زیر است.

حتما بایستی به CTE که همان cteName است اشاره کرده و در جایی از آن استفاده شده باشد. به عبارت دیگر از رکوردهای موجود در جدول موقت استفاده کند تا بتواند رکوردهای بعدی را تشخیص دهد.

حتما بایستی مطمئن شوید که شرایط کافی برای پایان حلقه پیمایش رکوردها را داشته باشد در غیر این صورت سبب تولید حلقه بی پایان[9] خواهد شد.

بدنه CTE می-تواند حاوی چندین anchor member و چندین recursive member باشد ولی فقط recursive memberها هستند که به CTE اشاره می-کنند.

برای آنکه نکات فوق روشن شود به مثال-های زیر توجه کنید.

فرض کنید جدولی از کارمندان و مدیران آنها داریم که به صورت زیر تعریف و مقداردهی اولیه شده است.

```
IFOBJECT_ID('Employees','U')ISNOTNULL
DROPTABLE dbo.Employees
GO

CREATETABLE dbo.Employees
(
  EmployeeID intNOTNULLPRIMARYKEY,
  FirstName varchar(50)NOTNULL,
  LastName varchar(50)NOTNULL,
  ManagerID intNULL
)

GO


INSERTINTO Employees VALUES (101,'Alireza','Nematollahi',NULL)
INSERTINTO Employees VALUES (102,'Ahmad','Mofarrahzadeh', 101)
INSERTINTO Employees VALUES (103,'Mohammad','BozorgGhommi', 102)
INSERTINTO Employees VALUES (104,'Masoud','Narimani', 103)
INSERTINTO Employees VALUES (105,'Mohsen','Hashemi', 103)
INSERTINTO Employees VALUES (106,'Aref','Partovi', 102)
INSERTINTO Employees VALUES (107,'Hosain','Mahmoudi', 106)
INSERTINTO Employees VALUES (108,'Naser','Pourali', 106)
INSERTINTO Employees VALUES (109,'Reza','Bagheri', 102)
INSERTINTO Employees VALUES (110,'Abbas','Najafian', 102)
```

مثال اول: می-خواهیم فهرست کارمندان را به همراه نام مدیر آنها و شماره سطح درخت نمایش دهیم. کوئری زیر نمونه‌ای از یک کوئری بر اساس CTE بازگشتی می-باشد.

```
WITHcteReports(EmpID, FirstName, LastName, MgrID, EmpLevel)
AS
(
SELECT EmployeeID, FirstName, LastName, ManagerID, 1
FROM Employees
WHERE ManagerID ISNULL
UNIONALL
SELECT e.EmployeeID, e.FirstName, e.LastName, e.ManagerID,r.EmpLevel + 1
FROM Employees e INNERJOINcteReports r
ON e.ManagerID = r.EmpID
)
SELECT
FirstName +' '+ LastName AS FullName,
EmpLevel,
(SELECT FirstName +' '+ LastName FROM Employees
WHERE EmployeeID = cteReports.MgrID)AS Manager
FROMcteReports
ORDERBY EmpLevel, MgrID
```

کوئری اول در بدنه CTE رکورد مدیری را می-دهد که ریشه درخت بوده و بالاسری ندارد و شماره سطح این رکورد را 1 در نظر می-گیرد.

کوئری دوم در بدنه CTE از یک JOIN بین Employees و cteReports استفاده کرده و کارمندان زیر دست هر کارمند قبلی (فرزندان) را بدست آورده و مقدار شماره سطح آنرا به صورت Level+1 تنظیم می-کند.

در نهایت با استفاده از CTE و یک subquery جهت بدست آوردن نام مدیر هر کارمند، نتیجه نهایی تولید می-شود.

مثال دوم: می-خواهیم شناسه یک کارمند را بدهیم و نام او و نام مدیران وی را به عنوان جواب در خروجی بگیریم.

```
WITHcteReports(EmpID, FirstName, LastName, MgrID, EmpLevel)
AS
(
SELECT EmployeeID, FirstName, LastName, ManagerID, 1
FROM Employees
WHERE EmployeeID = 110
UNIONALL
SELECTe.EmployeeID, e.FirstName, e.LastName, e.ManagerID,r.EmpLevel + 1
FROM Employees e INNERJOINcteReports r
ON e.EmployeeID = r.MgrID
```

```
)
SELECT
FirstName +' '+ LastName AS FullName,
EmpLevel
FROMcteReports
ORDERBY EmpLevel
```

اگر دقت کنید اولین تفاوت در خط اول مشاهده می-شود. در اینجا مشخص می-کند که اولین سری از رکوردها چگونه انتخاب شود. مثلا کارمندی را می-خواهیم که شناسه آن 110 باشد.
دومین تفاوت اصلی این کوئری با مثال قبلی، در قسمت دوم دیده می-شود. شما می‌خواهید مدیر (پدر) کارمندی که در آخرین پردازش در جدول موقت قرار گرفته است را استخراج کنید.

[1] a temporary named result set

[2] recursive

[3] nonrecursive

[4] Scope

[5]مثلا محدوده کدهای یک روال یا یک تابع

[6] nonrecursive

[7] recursive

[8] member

[9] Infinite loop

نظرات خوانندگان

نویسنده: همراز
تاریخ: ۱۳۹۳/۰۱/۲۶ ۱۱:۷

ضمن تشکر از پست مفید جناب عمران یکی از استفاده‌های CTE افزودن شماره ردیف به ساختار خروجی و محدود کردن نتیجه باتوجه به شماره ردیف است.
مثلاً ردیف 20 تا 30 ... که البته با پارامتر پاس می‌شوند.

```
WITH RCTE AS
(
SELECT TOP (100)
ROW_NUMBER() OVER (ORDER BY Invoice.InsertDate ASC) AS RowNumber,
Invoice.ID,
Invoice.PreInvoiceNo,
Invoice.InvoiceNo,
Invoice.IssueDate,
Invoice.CustomerID,  ...
FROM
Invoice
WHERE
Invoice.HistorySequence = 1
)
SELECT DISTINCT
RCTE.ID,
RCTE.PreInvoiceNo,
RCTE.InvoiceNo,
(dbo.fnc_Calendar_Gregorian_to_Persian(RCTE.IssueDate) + 'T' + CONVERT(CHAR(8), RCTE.IssueDate, 14)) AS
IssueDate,
RCTE.CustomerID,
Customer.NameEn AS CustomerNameEn,
Customer.NameFa AS CustomerNameFa,
FROM
RCTE
INNER JOIN Customer ON RCTE.CustomerID = Customer.ID
WHERE
RowNumber BETWEEN @StartFrom AND (@RowsCount + @StartFrom - 1)
```

عنوان: NoSQL و مایکروسافت
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۰۸ ۱۵:۴۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, NoSQL

روشی را که مایکروسافت برای پرداختن به مقوله NoSQL تاکنون انتخاب کرده است، قرار دادن ویژگی‌هایی خاصی از دنیای NoSQL مانند امکان تعریف اسکیمای متغیر، داخل مهم‌ترین بانک اطلاعاتی رابطه‌ای آن، یعنی SQL Server است، که در ادامه به آن خواهیم پرداخت. همچنین در سمت محصولات پردازش ابری آن نیز امکان دسترسی به محصولات NoSQL کاملی وجود دارد.

1) Azure table storage

Azure table storage در حقیقت یک Key-value store ابری است و برای کار با آن از اینترفیس پروتکل استاندارد OData استفاده می‌شود. علت استفاده و طراحی یک سیستم Key-value store در اینجا، مناسب بودن اینگونه سیستم‌ها جهت مقاصد عمومی است و به این ترتیب می‌توان به بازه بیشتری از مصرف کنندگان، خدمات ارائه داد.

پیش از ارائه Azure table storage، مایکروسافت سرویس خاصی را به نام SQL Server Data Services که به آن SQL Azure نیز گفته می‌شود، معرفی کرد. این سرویس نیز یک Key-Value store است؛ هرچند از SQL Server به عنوان مخزن نگهداری اطلاعات آن استفاده می‌کند.

2) SQL Azure XML Columns

فیلدهای XML از سال 2005 به امکانات توکار SQL Server اضافه شدند و این نوع فیلدها، بسیاری از مزایای دنیای NoSQL را درون SQL Server رابطه‌ای مهیا می‌سازند. برای مثال با تعریف یک فیلد به صورت XML، می‌توان از هر ردیف به ردیفی دیگر، اطلاعات متفاوتی را ذخیره کرد؛ به این ترتیب امکان کار با یک فیلد که می‌تواند اطلاعات یک شیء را قبول کند و در حقیقت امکان تعریف اسکیمای پویا و متغیر را در کنار امکانات یک بانک اطلاعاتی رابطه‌ای که از اسکیمای ثابت پشتیبانی می‌کند، میسر می‌شود. در این حالت در هر ردیف می‌توان تعدادی ستون ثابت را با یک ستون XML با اسکیمای کاملا پویا ترکیب کرد.

همچنین SQL Server در این حالت قابلیتی را ارائه می‌دهد که در بسیاری از بانک‌های اطلاعاتی NoSQL میسر نیست. در اینجا در صورت نیاز و لزوم می‌توان اسکیمای کاملا مشخصی را به یک فیلد XML نیز انتساب داد؛ هر چند این مورد اختیاری است و می‌توان یک un typed XML را نیز بکار برد. به علاوه امکانات کوئری گرفتن توکار از این اطلاعات را به کمک XPath ترکیب شده با T-SQL، نیز فراموش نکنید.

بنابراین اگر یکی از اهداف اصلی گرایش شما به سمت دنیای NoSQL، استفاده از امکان تعریف اطلاعاتی با اسکیمای متغیر و پویا است، فیلدهای نوع XML اس کیوال سرور را مدنظر داشته باشید.

یک مثال عملی: فناوری Azure Dev Fabric's Table Storage (نسخه Developer ویندوز Azure که روی ویندوزهای معمولی اجرا می‌شود؛ یک شبیه ساز خانگی) به کمک SQL Server و فیلدهای XML آن طراحی شده است.

3) SQL Azure Federations

در اینجا منظور از Federations در حقیقت همان پیاده سازی قابلیت Sharding بانک‌های اطلاعاتی NoSQL توسط SQL Azure است که برای توزیع اطلاعات بر روی سرورهای مختلف طراحی شده است. به این ترتیب دو قابلیت Partitioning و همچنین Replication به صورت خودکار در دسترس خواهند بود. هر Partition در اینجا، یک SQL Azure کامل است. بنابراین چندین بانک اطلاعاتی فیزیکی، یک بانک اطلاعاتی کلی را تشکیل خواهند داد.

هرچند در اینجا Sharding (که به آن Federation member گفته می‌شود) و در پی آن مفهوم «عاقبت یک دست شدن اطلاعات» وجود دارد، اما درون یک Shard یا یک Federation member، مفهوم ACID پیاده سازی شده است. از این جهت که هر Shard واقعا یک بانک اطلاعاتی رابطه‌ای است. اینجا است که مفهوم برنامه‌های Multi-tenancy را برای درک آن باید درنظر داشت. برای نمونه یک برنامه وب را درنظر بگیرید که قسمت اصلی اطلاعات کاربران آن بر روی یک Shard قرار دارد و سایر اطلاعات بر روی سایر Shards پراکنده شده‌اند. در این حالت است که یک برنامه وب با وجود مفهوم ACID در یک Shard می‌تواند سریع پاسخ دهد که آیا کاربری پیشتر در سایت ثبت نام کرده است یا خیر و از ثبت نام‌های غیرمجاز جلوگیری به عمل آورد.

در اینجا تنها موردی که پشتیبانی نشده‌است، کوئری‌های Fan-out می‌باشد که پیشتر در مورد آن بحث شد. از این جهت که با نحوه خاصی که Sharding آن طراحی شده است، نیازی به تهیه کوئری‌هایی که به صورت موازی بر روی کلیه Shards برای جمع آوری اطلاعات اجرا می‌شوند، نیست. هر چند از هر shard با استفاده از برنامه‌های دات نت، می‌توان به صورت جداگانه نیز کوئری

گرفت.

4) OData

اگر به CouchDB و امکان دسترسی به امکانات آن از طریق وب دقت کنید، در محصولات مایکروسافت نیز این دسترسی REST API پیاده سازی شده‌اند.

OData یک RESTful API است برای دسترسی به اطلاعاتی که به شکل XML یا JSON بازگشت داده می‌شوند. انواع و اقسام کلاینت‌هایی برای کار با آن از جاوا اسکریپت گرفته تا سیستم‌های موبایل، دات نت و جاوا، وجود دارند. از این API نه فقط برای خواندن اطلاعات، بلکه برای ثبت و به روز رسانی داده‌ها نیز استفاده می‌شود. در سیستم‌های جاری مایکروسافت، بسیاری از فناوری‌ها، اطلاعات خود را به صورت OData دراختیار مصرف کنندگان قرار می‌دهند مانند Azure table storage، کار با SQL Azure از طریق WCF Data Services (جایی که OData از آن نشات گرفته شده)، Azure Data Market (برای ارائه فیدهایی از اطلاعات خصوصا رایگان)، ابزارهای گزارشگیری مانند SQL Server reporting services، لیست‌های شیرپوینت و غیره.

به این ترتیب به بسیاری از قابلیت‌های دنیای NoSQL مانند کار با اطلاعات JSON بدون ترک دنیای رابطه‌ای می‌توان دسترسی داشت.

5) امکان اجرای MongoDB و امثال آن روی سکوی کاری Azure

امکان توزیع MongoDB بر روی یک Worker role سکوی کاری Azure وجود دارد. در این حالت بانک‌های اطلاعاتی این سیستم‌ها بر روی Azure Blob Storage قرار می‌گیرند که به آن‌ها Azure drive نیز گفته می‌شود. همین روش برای سایر بانک‌های اطلاعاتی NoSQL نیز قابل اجرا است.

به علاوه امکان اجرای Hadoop نیز بر روی Azure وجود دارد. مایکروسافت به کمک شرکتی به نام HortonWorks نسخه ویندوزی Hadoop را توسعه داده‌اند. HortonWorks را افرادی تشکیل داده‌اند که پیشتر در شرکت یاهو بر روی پروژه Hadoop کار می‌کرده‌اند.

6) قابلیت‌های فرا رابطه‌ای SQL Server

الف) فیلدهای XML (که در ابتدای این مطلب به آن پرداخته شد). به این ترتیب می‌توان به یک اسکیمای انعطاف پذیر، بدون از دست دادن ضمانت ACID رسید.

ب) فیلد HierarchyId برای ذخیره سازی اطلاعات چند سطحی. برای مثال در بانک‌های اطلاعاتی NoSQL سندگرا، یک سند می‌تواند سند دیگری را در خود ذخیره کند و الی آخر.

ج) Sparse columns؛ ستون‌های اسپارس تقریبا شبیه به Key-value stores عمل می‌کنند و یا حتی Wide column stores نیز با آن قابل مقایسه است. در اینجا هنوز اسکیما وجود دارد، اما برای نمونه علت استفاده از Wide column stores این نیست که واقعا نمی‌دانید ساختار داده‌های مورد استفاده چیست، بلکه در این حالت می‌دانیم که در هر ردیف تنها از تعداد معدودی از فیلدها استفاده خواهیم کرد. به همین جهت در هر ردیف تمام فیلدها قرار نمی‌گیرند، چون در اینصورت تعدادی از آن‌ها همواره خالی باقی می‌ماندند. مایکروسافت این مشکل را با ستون‌های اسپارس حل کرده است؛ در اینجا هر چند ساختار کلی مشخص است، اما مواردی که هر بار استفاده می‌شوند، تعداد محدودی می‌باشند. به این صورت SQL Server تنها برای ستون‌های دارای مقدار، فضایی را اختصاص می‌دهد. به این ترتیب از لحاظ فیزیکی و ذخیره سازی نهایی، به همان مزیت Wide column stores خواهیم رسید.

د) FileStreams در اس کیوال سرور بسیار شبیه به پیوست‌های سندهای بانک‌های اطلاعاتی NoSQL سندگرا هستند. در اینجا نیز اطلاعات در فایل سیستم ذخیره می‌شوند اما ارجاعی به آن‌ها در جداول مرتبط وجود خواهند داشت.

7) SQL Server Parallel Data Warehouse Edition

SQL PDW، نگارش خاصی از SQL Server است که در آن یک شبکه از SQL Serverها به صورت یک وهله منطقی SQL Server در اختیار برنامه نویس‌ها قرار می‌گیرد.

این نگارش، از فناوری خاصی به نام MPP یا massively parallel processing برای پردازش کوئری‌ها استفاده می‌کند. در اینجا همانند بانک‌های اطلاعاتی NoSQL، یک کوئری به نود اصلی ارسال شده و به صورت موازی بر روی تمام نودها پردازش گردیده (همان مفهوم Map Reduce که پیشتر در مورد آن بحث شد) و نتیجه در اختیار مصرف کننده قرار خواهد گرفت. نکته مهم آن نیز در عدم نیاز به نوشتن کدی جهت رخ دادن این عملیات از طرف برنامه نویس‌ها است و موتور پردازشی آن جزئی از سیستم اصلی است. تنها کافی است یک کوئری SQL صادر گردد تا نتیجه نهایی از تمام سرورها جمع آوری و بازگردانده شود.

این نگارش ویژه تنها به صورت یک Appliance به فروش می‌رسد (به صورت سخت افزار و نرم افزار باهم) که در آن CPUها،

فضاهای ذخیره سازی اطلاعات و جزئیات شبکه به دقت از پیش تنظیم شده‌اند.

عنوان: **نحوه تهیه گزارش در SSRS و انتشار آن روی وب سرور**
نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۲/۰۶/۱۶ ۰:۳۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Reporting, SSRS, Business Intelligence, Report

یکی از محصولات پرکاربرد و حرفه ای مایکروسافت در زمینه تولید گزارش SQL Server Reporting Services یا به اختصار SSRS می‌باشد . در این پست نحوه ایجاد یک گزارش ساده به صورت والد و فرزندی ، انتشار گزارش روی وب سرور و مدیریت نمایش ستون‌ها با استفاده از Expressionها را در محیط BIDS بیان می‌کنم

برای انجام این پروسه ، از ابزار BIDS استفاده خواهیم کرد . همچنین برای اطلاعات و داده‌ها از دیتابیس آزمایشی Adventure works استفاده می‌کنیم (دانلود)

صورت مسئله : گزارش از دپارتمان‌های کاری و تعداد کارمندان مرد و زن هر دپارتمان . که با کلیک روی هر سطر گزارش (بسته به جنسیت) بتوان لیست تمام افراد آن دپارتمان را دید .

بخش اول :

برای شروع BIDS را باز کرده و یک پروژه از نوع Report Server Project استارت بزنید .

نام این پروژه را در اینجا introductionPrj1 قرار دادم .

همانطور که ملاحظه می‌کنید ، سه پوشه در Solution explorer قرار دارد که برای تعریف پایگاه داده با پوشه Shared Data Sources و برای تعریف گزارشات از پوشه Reports استفاده خواهیم کرد

برای این منظور ابتدا یک data source تعریف می‌کنیم :

و سپس شروع به ساختن یک گزارش می‌کنیم :

مراحل را تا رسیدن به مرحله تعریف Query پی میگیریم .

انتخاب دیتا سورس :

در اینجا می‌خواهیم قسمت اول گزارش یعنی فهرست کردن تعداد کارمندان هر دپارتمان را به تفکیک جنسیت مشاهده کنیم :

```
SELECT DEPARTMENTNAME, GENDER, COUNT(1) AS COUNT FROM DBO.DIMEMPLOYEE GROUP BY DEPARTMENTNAME,GENDER
ORDER BY DEPARTMENTNAME,GENDER
```

برای مشاهده صحت دستور می‌توانید از Query Builder کمک بگیرید :

ادامه تنظیمات را مانند تصویر پی بگیرید (تعریف Tabular بودن گزارش و طراحی جدول و theme و نام گذاری گزارش)

Select the Report Type

Select the type of report that you want to create.

Tabular

Matrix

Design the Table

Choose how to group the data in the table.

Choose the Table Style
Choose a style for the table.
Slate
Forest
Corporate
Bold
Ocean
Generic
XXXXXX
XXXX

به این ترتیب بخش اول گزارش ایجاد شد . حال باید زیر گزارش مربوطه را ایجاد کنیم :

مجددا مراحل را برای ساخت یک گزارش جدید پیگیری کنید و برای دستور کوئری از دستور زیر استفاده کنید :

```
SELECT EMPLOYEEKEY,FIRSTNAME,LASTNAME, MIDDLENAME,TITLE,HIREDATE, BIRTHDATE,EMAILADDRESS,PHONE,GENDER
FROM DBO.DIMEMPLOYEE WHERE DEPARTMENTNAME=@DEPARTMENTNAME AND GENDER=@GENDER
```

Report Wizard
Design the Query
Specify a query to execute to get the data for the report.
Use a query builder to design your query.
Query Builder...
Query string:
SELECT EmployeeKey, FirstName, LastName, MiddleName, Title, HireDate, BirthDate, EmailAddress, Phone, Gender
FROM DimEmployee
WHERE (DepartmentName = @DEPARTMENTNAME) AND (Gender = @GENDER)

و تست گزارش :

و بقیه قسمت‌ها مانند قبل :

Report Wizard
Design the Table
Choose how to group the data in the table.
Available fields:
Displayed fields:
Page>
Gender
Group>
Details>
EmployeeKey
FirstName
LastName
MiddleName
Title
HireDate
BirthDate
EmailAddress
Phone
< Remove

تا به این مرحله data source و گزارش‌ها ایجاد شدند :

اکنون باید ارتباط بین دو گزارش را برقرار کنیم :

گزارش والد را باز کرده و روی ستون COUNT کلیک راست نموده و گزینه Popperties را انتخاب نمایید :

سپس در تب action گزینه Go to Report را انتخاب نموده و گزارش فرزند را انتخاب نمایید .

در انتها هم باید پارامترها را تعریف کنید . خروجی مانند زیر خواهد بود :

ToolTip از تب General قابل اعمال است .

و در نهایت با کلیک روی این سطر می‌توانید گزارش مرتبط را مشاهده کنید :

Employee Key	First Name	Last Name	Middle Name	Title	Hire Date	Birth Date	Email Address	Phone
56	Tengiz	Kharatishvili	N	Control Specialist	1/17/1999 12:00:00 AM	5/29/1980 12:00:00 AM	tengiz0@adventure-works.com	910-555-0116
92	Zainal	Arifin	T	Document Control Manager	2/5/1999 12:00:00 AM	3/2/1966 12:00:00 AM	zainal0@adventure-works.com	204-555-0115
130	Sean	Chai	N	Document Control Assistant	2/23/1999 12:00:00 AM	4/12/1977 12:00:00 AM	sean1@adventure-works.com	205-555-0132
216	Chris	Norred	K	Control Specialist	4/7/1999 12:00:00 AM	6/26/1977 12:00:00 AM	chris1@adventure-works.com	575-555-0126

تا به این مرحله گزارش تکمیل شد که البته برای ظاهر آن هم باید فکری کرد که در این پست اشاره ای نمی‌شود .

بخش دوم :

گزارش جاری فقط قابل استفاده از طریق BIDS است و با توجه به محدودیت دسترسی باید آن را در جایی قرار داد تا کاربران بتوانند از آن استفاده کنند . برای این منظور باید تنظیمات SSRS Web Application انجام شود تا بتواند روی سرور عملیاتی قرار گیرد .

در صورتی که تنظیمات SSRS برای قرار گرفتن روی وب سرور انجام نشده باشد و ما بخواهیم گزارش را Deploy کنیم خطا دریافت خواهیم کرد .

پس در ادامه نحوه تنظیم وب سرور را بیان می‌کنم و پس از آن گزارش را روی وب سرور قرار می‌دهیم :
برای این منظور باید برنامه Reporting Services Configuration Manager که در مسیر نصب SQL Server است برویم

پس از اتصال به سرور به تب Report Manager Url بروید :

در این مرحله باید سرور را تنظیم کنید تا بتوانیم پروژه را روی آن Deploy کنیم . از باز بودن پورت اطیمنان حاصل کنید . سپس وب سرویس را تنظیم کنید که هر دو فقط شامل نام Virtual Directory و Credential آن می‌شود . (مگر اینکه تنظیمات خاصی داشته باشید).

در صورت اجرا کردن مسیر URL باید بتوانید صفحه خانگی آن را مشاهده کنید :

که البته هنوز هیچ گزارشی روی آن قرار نگرفته است .سپس به گزارش خود باز می‌گردیم تا تنظیمات سرور را روی BIDS تکمیل کنیم :

برای این منظور روی پروژه کلیک راست کنید و ابتدا روی Properties کلیک کنید .

Solution Explorer - Introd...
WSPBuilder
Build
Rebuild
Deploy
2
Clean
Add
Import Reports
Set as StartUp Project
Debug
Cut
Paste
Remove
Rename
Unload Project
Find Code Issues
Find Symbols External to Scope
Find Code Dependent on Module
Optimize references...
Project Hierarchy
Refactor
Cleanup Code...
Run Unit Tests
Debug Unit Tests
1
Collapse All
Properties
lution 'IntroductionPrj1' (1 pro
IntroductionPrj1
Shared Data Sources
DataSource_Adventure
Shared Datasets
Reports
MyChildReport.rdl
MyParentReport.rdl

تنظمات مهم شامل مسیر سرور که برابر با همان سرویسی است که تنظیم کرده اید و مسیر پوشه ای که گزارش‌ها روی سرور در آن قرار خواهند گرفت ، می‌باشد.دقت کنید که باید حتما پوشه موجود باشد . اگر به حالت پیشفرش رها کنید ، گزارشات در Root قرار خواهند گرفت .

روی OK کلیک کنید و پروژه را Deploy کنید .

Output

Show output from: Build

```
------ Build started: Project: IntroductionPrj1, Configuration: Debug ------
Skipping 'DataSource_AdventureWorks.rds'. Item is up to date.
Skipping 'MyChildReport.rdl'. Item is up to date.
Skipping 'MyParentReport.rdl'. Item is up to date.
Build complete -- 0 errors, 0 warnings
------ Deploy started: Project: IntroductionPrj1, Configuration: Debug ------
Deploying to http://mbs-lt/ssrs_srv
Deploying data source '/Data Sources/DataSource_AdventureWorks'.
Deploying report '/IntroductionPrj1/MyChildReport'.
Deploying report '/IntroductionPrj1/MyParentReport'.
Deploy complete -- 0 errors, 0 warnings
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
========== Deploy: 1 succeeded, 0 failed, 0 skipped ==========
```

Error List | Output

اگر به سایت برگردید ، گزارشات را می‌توانید مشاهده کنید

SQL Server Reporting Services

Home

New Folder | New Data Source | Report Builder | Folder Settings | Upload File

Data Sources

IntroductionPrj1

Home > IntroductionPrj1 > MyParentReport

1 of 1 100% Find

MyParentReport

Department Name	Gender	COUNT
Document Control	F	1
Document Control	M	4
Engineering	F	3
Engineering	M	[illegible]
Executive	F	2
Executive	M	1
Facilities and Maintenance	F	2
Facilities and	M	5

مشاهده جزییات

Home > IntroductionPrj1 > MyChildReport

1 of 1 100% Find | Next

MyChildReport

M

Employee Key	First Name	Last Name	Middle Name	Title	Hire Date	Birth Date	Email Addre
56	Tengiz	Kharatishvili	N	Control Specialist	1/17/1999 12:00:00 AM	5/29/1980 12:00:00 AM	tengiz0 enture-works.c
92	Zainal	Arifin	T	Document Control Manager	2/5/1999 12:00:00 AM	3/2/1966 12:00:00 AM	zainal0@ nture-works.c
130	Sean	Chai	N	Document Control Assistant	2/23/1999 12:00:00 AM	4/12/1977 12:00:00 AM	sean1@ nture-works.c
216	Chris	Norred	K	Control Specialist	4/7/1999 12:00:00 AM	6/26/1977 12:00:00 AM	chris1@ nture-works.c

بخش سوم :

در این مرحله می‌خواهیم یکی از ویژگی هایی که در گزارش گیری کاربرد زیادی دارد ، یعنی نمایش ستون‌های دلخواه در گزارش را به کمک SSRS و BIDS به کار ببریم.

برای این منظور به پنجره Report Data مراجعه کرده و روی Parameters کلی راست کرده و گزینه Add Parameter را انتخاب کنید :

فیلدها را مانند زیر پر کنید : (در اینجا می‌خواهیم یک dropdown به کاربر نشان داده شود تا انتخاب کند که ستون نمایش داده شود یا خیر . همچنین مقدار پیشفرض بله است)

Report Parameter Properties

General
Available Values
Default Values
Advanced

Change name, data type, and ot

Name:
ShowBirthDate

Prompt:
آیا ستون تاریخ تولد نمایش داده شود؟

Data type:
Text

Allow blank value ("")
Allow null value
Allow multiple values

Select parameter visibility:
Visible
Hidden
Internal

گزینه‌های مورد نظر را انتخاب نمایید

و تعیین مقدار پیش فرض

و نتیجه در BIDS :

اکنون باید تغییر مقدار این ستون را بر روی گزارش اعمال کنیم :

برای همین منظور روی ستون BirthDate در حالت Design کلیک راست کرده و گزینه Column Visibility را انتخاب کنید :

سپس باید تنظیم کنیم که در نمایش و عدم نمایش این ستون باید بر اساس یک عبارت یا expression باشد .

عبارت زیر را وارد کنید (به این معنی که اگر مقدار 1 بود نمایش داده شود در غیر این صورت نمایش داده نشود) برای اطلاعات بیشتر در مورد دستورات Expression به اینجا و اینجا و اینجا مراجعه کنید

```
=(iif(Parameters!ShowBirthDate.Value=1,true,false
```

اکنون می‌توانید گزارش را deploy کنید و تنظیمات سطوح دسترسی کاربران را انجام دهید

موفق باشید

نظرات خوانندگان

نویسنده: محمد دلیری
تاریخ: ۱۳۹۲/۰۶/۱۶ ۱۵:۱۴

سلام ممنونم از مطلبتون جالب بود ! .چجور میشه این نمونه گزارش گیری رو در Windows Application استفاده کرد؟

نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۲/۰۶/۱۶ ۱۶:۵۴

با استفاده از SOAP APIهای SSRS می‌تونید (+ و + و +)
و در بخش Reporting موجود در VS Toolbox کنترل Report Viewer بهتون کمک می‌کنه.
موفق باشید

نویسنده: bahman
تاریخ: ۱۳۹۲/۰۶/۱۹ ۱۵:۲۳

سلام؛ مقاله خوبی بود. فقط من یه سوال دارم. چطوری از پروسیجر ای استفاده کنیم که پرامتر میگیره. ممنون

نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۲/۰۶/۱۹ ۱۶:۵۸

سلام
باید از **Dataset** استفاده کنید (+ و +)

نویسنده: arefeh
تاریخ: ۱۳۹۳/۰۱/۱۸ ۱۲:۴۰

سلام
ممنون از اموزش شما .
مراحل گفته شده را اجرا کردم . زمانی که پروژه را deploy می‌کنم خطای زیر را می‌دهد
could not connect to the report server .verify that the target server url is valid and that you have the correct permissions to connect to the report server

نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۳/۰۱/۱۸ ۱۳:۰

سلام
از صحت آدرس‌های تعریف شده اطمینان پیدا کنید.

در قسمت Reporting Services Configuration Manager
در زیر مجموعه‌های Report Manager URL و Web Service URL
و همچنین
در قسمت Properties قسمت TargetServerRUL
احتمال دوم، عدم وجود دسترسی کافی برای تنظیمات فوق است
با کاربر مدیر ارشد این کار را انجام دهید

نویسنده: علیرضا
تاریخ: ۲۱:۴۳ ۱۳۹۳/۰۱/۲۰

مهمترین مشکل ابزارهای قدرتمند گزارشگیری اینه که اکثرا فقط با دستورات Sql کار می‌کنند. برای اینکه SSRS با مجموعه ای از موجودیتهای دامنه Domain Entities کار کنند چه راهی وجود داره؟ مثلا چطور میشه SSRS رو به Entity Framework متصل کرد؟

نویسنده: محمد باقر سیف اللهی
تاریخ: ۲۲:۴۰ ۱۳۹۳/۰۱/۲۰

به هیچ وجه ، انطباق خوب یک ابزار با یک پایگاه داده ازطریق SQL یک اشکال محسوب نمیشه !
منظورتون از برقراری ارتباط بین EF و SSRS اینه که data source انتخابی برای SSRS رو EF انتخاب کنیم ؟ اگر **بله** ؛ دلیله این کار چیه که از یک سو EF به SQL وصل بشه و از طرفی هم به SSRS... مستقیم این ارتباط رو برقرار می‌کنیم... (یعنی SSRS به SQL و با همان دستورات بدون نیاز به وجود واسطه ای به نام EF) هر ابزاری را با هدف خاصی پیاده سازی می‌کنند و اصلا دلیلی نداره که انتظار برقراری از طریق EF با SSRS وجود داشته باشه ! EF برای یک مقیاس و کاربرد استفاده میشه و SSRS برای یک مقیاس و کاربرد دیگر.

و اگر **نه** ، ممکنه بیشتر توضیح بدید؟

نویسنده: علیرضا
تاریخ: ۲۱:۰ ۱۳۹۳/۰۱/۲۳

راهی وجود داره که SSRS از Domain Objectها گزارش تهیه کنه؟

نویسنده: وحید نصیری
تاریخ: ۲۱:۴۲ ۱۳۹۳/۰۱/۲۳

- EF -> WCF Data Service -> SSRS via XML
- Data Processing Extension
- Using SSRS In ASP.NET MVC Application
- Microsoft.Reporting.WebForms.ReportDataSource

عنوان: **مدیریت اطلاعات وابسته به زمان در بانک‌های اطلاعاتی رابطه‌ای**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۱۶ ۱۴:۲۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, NoSQL, database design

در یک برنامه فروشگاه، جداول مشتری و خریدهای او را درنظر بگیرید. خرید 3 سال قبل مشتری خاصی به آدرس قبلی او ارسال شده‌است. خرید امروز او به آدرس جدید او ارسال خواهد شد. سؤال: آیا با وارد کردن و به روز رسانی آدرس جدید مشتری، باید سابقه اطلاعاتی قبلی او حذف شود؟ اجناس ارسالی پیشین او، واقعا به آدرس دیگری ارسال شده‌اند و نه به آدرس جدید او. چگونه باید اینگونه اطلاعاتی را که در طول زمان تغییر می‌کنند، در بانک‌های اطلاعاتی رابطه‌ای نرمال شده مدیریت کرد؟ از این نمونه‌ها در دنیای کاری واقعی بسیارند. برای مثال قیمت اجناس نیز چنین وضعی را دارند. یک بستنی مگنوم، سال قبل 300 تومان بود؛ امسال شده است 1500 تومان. یک سطل ماست 2500 تومان بود؛ امروز همان سطل ماست 6500 تومان است. چطور باید سابقه فروش این اجناس را نگهداری کرد؟

منابع مطالعاتی مرتبط

این موضوع اینقدر مهم است که تابحال چندین کتاب در مورد آن تالیف شده است:

Temporal Data & the Relational Model
Trees and Hierarchies in SQL
Developing Time-Oriented Database Applications in SQL
Temporal Data: Time and Relational Databases
Temporal Database Entries for the Springer Encyclopedia of Database Systems
Temporal Database Management

نکته مهمی که در این مآخذ وجود دارند، واژه کلیدی « Temporal data » است که می‌تواند در جستجوهای اینترنتی بسیار مفید واقع شود.

بررسی ابعاد زمان

فرض کنید کارمندی را استخدام کرده‌اید که ساعتی 2000 تومان از ابتدای فروردین ماه حقوق دریافت می‌کند. حقوق این شخص از ابتدای مهرماه قرار است به ساعتی 2400 تومان افزایش یابد. اگر مامور مالیات در بهمن ماه در مورد حقوق این شخص سؤال پرسید، ما چه پاسخی را باید ارائه دهیم؟ قطعا در بهمن ماه عنوان می‌کنیم که حقوقش ساعتی 2400 تومان است؛ اما واقعیت این است که این عدد از ابتدای استخدام او ثابت نبوده است و باید تاریخچه تغییرات آن، در نحوه محاسبه مالیات سال جاری لحاظ شود.
بنابراین در مدل سازی این سیستم به دو زمان نیاز داریم:
الف) actual time یا زمان رخ دادن واقعه‌ای. برای مثال حقوق شخصی در تاریخ ابتدای مهر ماه تغییر کرده است. به این تاریخ در منابع مختلف Valid time نیز گفته می‌شود.
ب) record time یا زمان ثبت یک واقعه؛ مثلا زمان پرداخت حقوق. به آن Transaction time هم گفته شده است.
یک مثال:

```
record date  actual date  حقوق دریافتی
1392/01/01  1392/01/01  2000/روز
1392/02/01  1392/01/01  2000/روز
...
1392/07/01  1392/07/01  2400/روز
...
1392/17/01  1392/07/01  2400/روز
```

در این لیست، ریز حقوق پرداختی به یک شخص را ملاحظه می‌کنید. actual dateها، زمان‌هایی هستند که حقوق پایه شخص در

آن‌ها تغییر کرده و record dateها زمان‌هایی هستند که به شخص حقوق داده شده‌است.
به ترکیب Valid Time و Transaction Time، اصطلاحا Bitemporal data می‌گویند.

مشکلات طراحی‌های متداول اطلاعات وابسته به زمان

در طراحی‌های متداول، عموما یک جدول کارمندان وجود دارد و یک جدول لیست حقوق‌های پرداختی. رکوردهای لیست حقوق‌های پرداختی نیز توسط یک کلید خارجی به اطلاعات هر کارمند متصل است؛ از این جهت که نمی‌خواهیم اطلاعاتی تکراری را در جدول لیست حقوقی ثبت کنیم و طراحی نرمال سازی شده‌ای مدنظر می‌باشد.
خوب؛ اول مهرماه حقوق شخصی تغییر کرده است. بنابراین کارمند بخش مالی اطلاعات شخص را به روز می‌کند. با این کار، کل سابقه حقوق‌های پرداختی شخص نیز از بین خواهد رفت. چون وجود این کلید خارجی به معنای استفاده از آخرین اطلاعات به روز شده یک کارمند در جدول لیست حقوقی است. الان اگر از جدول لیست حقوقی گزارش بگیریم، کارمندان همواره از آخرین حقوق به روز شده خودشان استفاده خواهند کرد.

راه حل‌های متفاوت مدل سازی اطلاعات وابسته به زمان
برای رفع این مشکل مهم، راه حل‌های متفاوتی وجود دارند که در ادامه آن‌ها را بررسی خواهیم کرد.

الف) نگهداری اطلاعات وابسته به زمان در جداول نهایی مرتبط
اگر حقوق پایه شخص در زمان‌های مختلف تغییر می‌کند، بهتر است عدد نهایی این حقوق پرداختی نیز در یک فیلد مشخص، در همان جدول لیست حقوقی ثبت شود. این مورد به معنای داشتن «داده‌ای تکراری» نیست. از این جهت که داده‌ای تکراری است که اطلاعات آن در تمام زمان‌ها، دارای یک مقدار و مفهوم باشد و اطلاعات حقوق یک شخص اینچنین نیست.

ب) نگهداری اطلاعات تغییرات حقوقی در یک جداول جداگانه
یک جدول ثانویه حقوق جاری کارمندان مرتبط با جدول اصلی کارمندان باید ایجاد شود. در این جدول هر رکورد آن باید دارای بازه زمانی (valid_start_time و valid_end_time) مشخصی باشد. مثلا از تاریخ X تا تاریخ Y، حقوق کارمند شماره 11 ، 2000 تومان در ساعت بوده است. از تاریخ H تا تاریخ Z اطلاعات دیگری ثبت خواهند شد. به این ترتیب با گزارشگیری از جدول لیست حقوق‌های پرداخت شده، سابقه گذشته اشخاص محو نشده و هر رکورد بر اساس قرارگیری در یک بازه زمانی ثبت شده در جدول ثانویه حقوق جاری کارمندان تفسیر می‌شود.
در این حالت باید دقت داشت که بازه‌های زمانی تعریف شده، با هم تداخل نکنند و برنامه ثبت کننده اطلاعات باید این مساله را به ازای هر کارمند کنترل کند و یا با ثبت record_date، اجازه ثبت بازه‌های تکراری را نیز بدهد (توضیحات در قسمت بعد).
به این جدول، یک Temporal table نیز گفته می‌شود. نمونه دیگر آن، نگهداری قیمت یک کالا است از یک تاریخ تا تاریخی مشخص. به این ترتیب می‌توان کوئری گرفت که بستنی مگنوم فروخته شده در ماه آبان سال قبل، بر مبنای قیمت آن زمان، دقیقا چقدر فروش کرده است و نه اینکه صرفا بر اساس آخرین قیمت روز این کالا گزارشگیری کنیم که در این حالت اطلاعات نهایی استخراج شده صحیح نیستند.
حال اگر به این طراحی در جدولی دیگر Transaction time یا زمان ثبت یک رکورد یا زمان ثبت یک فروش را هم اضافه کنیم، به جداول حاصل Bitemporal Tables می‌گویند.

مدیریت به روز رسانی‌ها در جداول Temporal
در جداول Temporal، حذف فیزیکی اطلاعات مطلقا ممنوع است؛ چون سابقه سیستم را تخریب می‌کند. اگر اطلاعاتی در این جداول دیگر معتبر نیست باید تنها تاریخ پایان دوره آن به روز شوند یا یک رکورد جدید بر اساس بازه‌ای جدید ثبت گردد.
همچنین به روز رسانی‌ها در این جداول نیز معادل هستند با یک Insert جدید به همراه فیلد record_date و نه به روز رسانی واقعی یک رکورد قبلی (شبیه به سیستم‌های حسابداری باید عمل کرد).
یک مثال:
فرض کنید حقوق کارمندی که مثال زده شد، در مهرماه به ساعتی 2400 تومان افزایش یافته است و حقوق نهایی نیز پرداخته شده است. بعد از یک ماه مشخص می‌شود که مدیر عامل سیستم گفته بوده است که ساعتی 2500 تومان و نه ساعتی 2400 تومان! (از این نوع مسایل در دنیای واقعی زیاد رخ می‌دهند!) خوب؛ اکنون چه باید کرد؟ آیا باید رفت و رکورد ساعتی 2400 تومان را به روز کرد؟ خیر. چون سابقه پرداخت واقعی صورت گرفته را تخریب می‌کند. به روز رسانی شما ابدا به این معنا نخواهد بود که دریافتی

واقعی شخص در آن تاریخ خاص، ساعتی 2500 بوده است.
بنابراین در جداول Temporal، تنها «تغییرات افزودنی» مجاز هستند و این تغییرات همواره به عنوان آخرین رکورد جدول ثبت می‌شوند. به این ترتیب می‌توان اصطلاحا «مابه التفاوت» حقوق پرداخت نشده را به شخص خاصی، محاسبه و پرداخت کرد (می‌دانیم در یک بازه زمانی خاص به او چقد حقوق داده‌ایم. همچنین می‌دانیم که این بازه در یک record_date دیگر لغو و با عددی دیگر، جایگزین شده‌است).

برای مطالعه بیشتر

Bitemporal Database Table Design - The Basics

Temporal Data Techniques in SQL

Database Design: A Point in Time Architecture

Temporal database

Temporal Patterns

راه حلی دیگر؛ استفاده از بانک‌های اطلاعاتی NoSQL

بانک‌های اطلاعاتی NoSQL برخلاف بانک‌های اطلاعاتی رابطه‌ای برای اعمال Read بهینه سازی می‌شوند و نه برای Write. در چند دهه قبل که بانک‌های اطلاعاتی رابطه‌ای پدیدار شدند، یک سخت دیسک 10 مگابایتی حدود 4000 دلار قیمت داشته است. به همین جهت مباحث نرمال سازی اطلاعات و ذخیره نکردن اطلاعات تکراری تا این حد در این نوع بانک‌های اطلاعاتی مهم بوده است. اما در بانک‌های اطلاعاتی NoSQL امروزی، اگر قرار است فیش حقوقی شخصی ثبت شود، می‌توان کل اطلاعات جاری او را یکجا داخل یک سند ثبت کرد (از اطلاعات شخص در آن تاریخ تا اطلاعات تمام اجزای فیش حقوقی در قالب یک شیء تو در توی JSON). به همین جهت بسیار سریع هستند برای اعمال Read و گزارشگیری. همچنین این نوع سیستم‌ها برای نگهداری نگارش‌های مختلف یک سند بهینه سازی شده‌اند و جزو ساختار توکار آن‌ها است. بنابراین در این نوع سیستم‌ها اگر نیاز است از یک سند خاصی گزارش بگیریم، دقیقا اطلاعات همان تاریخ خاص را دارا است و اگر اطلاعات پایه سیستم را به روز کنیم، از امروز به بعد در سندهای جدید ثبت خواهد شد. این نوع سیستم‌ها رابطه‌ای نیستند و بسیاری از مباحث نرمال سازی اطلاعات در آن‌ها ضرورتی ندارد. قرار است یک فیش حقوقی شخص را نمایش دهیم؟ خوب، چرا تمام اطلاعات مورد نیاز او را در قالب یک شیء JSON تو در توی حاضر و آماده نداشته باشیم؟

نظرات خوانندگان

نویسنده: ناصر
تاریخ: ۱۳۹۲/۰۷/۱۷ ۰:۵۴

من هم قبلا از این روش برای قیمت‌های جدید و قدیم یک کالا در یک سیستم فروشگاهی استفاده کردم.
با نوشتن یک تریگر که به محض تغییر روی قیمت کالا ، بلافاصله داخل یک جدول دیگه این تغییرات درج میشد.
و موقع نمایش ، قیمت جدید و قدیم هر دو با هم به مشتری نمایش داده میشد.

نویسنده: سیروس
تاریخ: ۱۳۹۲/۰۷/۱۸ ۱۱:۱۱

به نظر من در نگهداری به این روش نیازی به تاریخ پایان نیست، مثلا هنگام تغییر قیمت کالا، رکوردی با تاریخ روز در جدول temporal ثبت می‌کنیم و در تغییر دوباره رکورد جدید دیگری ثبت می‌شود.
کارکردن به این روش آسانتر به نظر می‌رسد و یک فیلد کمتر داریم و نیازی هم به چک کردن درست بودن بازه‌ی تاریخی نیست.

Date	Price	ProdcutId
1392/01/01	1000	1
1392/03/05	1500	1
1392/06/27	1780	1

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۳/۰۴/۰۲ ۱۱:۴۵

البته روش شما برای حالتی مناسب است که بازه‌های تاریخی به هم متصل باشند.

عنوان: Embed کردن SQL Server Express 2008 در یک برنامه
نویسنده: جوادی
تاریخ: ۱۳۹۲/۰۷/۲۲ ۰:۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server 2008, SQL Server

مقدمه

نصب Microsoft Sql Server یکی از عملیات مشکل برای کاربر نهایی می‌باشد. برای رفع این مشکل، باید آنرا بصورت اتوماتیک و بدون درگیر کردن کاربر آن را نصب کنیم.برای اینکار دو روش موجود است:

1) استفاده از فایل Configure.ini

2) ارسال پارامتر به فایل Setup.exe از طریق Command Prompt

قابلیت‌های نسخه‌های مختلف Sql Server Express 2008

نسخه / قابلیت	Database Engine	Management Studio Basic	Full-Text Search	Reporting Services
Management Studio Basic		X		
Runtime Only	X			
with Tools	X	X		
with Advanced Services	X	X	X	X

SQL Server 2008 Management Studio Express (SSMSE)

ابزارهای مدیریتی گرافیکی رایگان برای پیکربندی، مدیریت و اداره کردن برنامه‌های SQL Server Express 2008 استفاده برای مدیریت چندین نمونه از موتور پایگاه داده SQL Server که توسط نسخه‌های مختلف SQL Server 2008 ساخته شده اند.

SQL Server 2008 Express (Runtime Only)

موتور پایگاه داده SQL Server برای ساخت، ذخیره سازی، بروز رسانی و واکشی داده

SQL Server 2008 Express with Tools

موتور پایگاه داده SQL Server برای ساخت، ذخیره سازی، بروز رسانی و واکشی داده

SQL Server Management Studio Basic - یک ابزار مدیریتی ویژوال برای ساخت، ویرایش و مدیریت پایگاه‌های داده

SQL Server 2008 Express with Advanced Services

موتور پایگاه داده SQL Server برای ساخت، ذخیره سازی، بروز رسانی و واکشی داده

SQL Server Management Studio Basic - یک ابزار مدیریتی ویژوال برای ساخت، ویرایش و مدیریت پایگاه‌های داده

Full-Text Search - یک موتور قدرتمند و پرسرعت برای جستجوی داده‌های متن-فشرده

دو حالت برای نصب SQL Server وجود دارد:

نصب یک نمونه جدید

آپگرید

نصب SQL Server Express 2008 از طریق Command Prompt

1) پس از دانلود SQL Server Express 2008 با پارامتر X/ انرا از حالت فشرده خارج کنید.

برای این کار Command Prompt (یا همان cmd) را باز کنید و با دستور cd به مسیری که فایل SQL Server Express هست بروید.

حالا نام فایل نصب را همراه با پارامتر X/ در cmd تایپ کنید. مثلا: SQLExpr32_x86_enu.exe /x

نکته: اگر فایل یا سی دی نصب ویژوال استودیو 2010 را دارید می‌توانید فایل نصب SQL Server Express را در مسیر \WCU\SSE پیدا کنید.

2) همانطور که می‌بینید یک مسیر برای Extract از شما خواسته شده. یک مسیر وراد کنید.

3) در مسیر ساخته شده، یک فایل با پسوند bat. بسازید و آنرا یا یک ویرایشگر متنی باز کنید و دستورات زیر را در آن تایپ

کنید:

```
Setup.exe /q /Action=Install /Hideconsole /Features=SQL,Tools
/InstanceName=SQLExpress /SQLSYSADMINACCOUNTS="Builtin\Administrators"
/SQLSVCACCOUNT="<DomainName\UserName>" /SQLSVCPASSWORD="<StrongPassword>
```

توضیح پارامترهای فوق بشرح زیر است:

/q (اختیاری): اینکه Setup به حالت خاموش (quiet mode) و بدون رابط کاربری اجرا شود.

/Action (الزامی): عملیاتی که باید انجام شود. این پارامتر مقدار install و upgrade را قبول می‌کند.

/Features (الزامی): اینکه کدام قابلیت‌های SQL Server باید نصب شوند.

/HideConsole (اختیاری): اگر از این پارامتر استفاده شود، پنجره کنسول نمایش داده نخواهد شد.

/InstanceName (الزامی): نام نمونه ایی که باید نصب شود.

/SQLSYSADMINACCOUNTS (الزامی): مقررات لوگین برای اعضای با نقش "مدیر سیستم".

/SQLSVCACCOUNT (الزامی): تعیین اکانتی که سرویس SQL Server را در Startup ویندوز اجرا کند.

/SQLSVCACCOUNT (اگر از یک اکانت لوکال یا تحت Domain استفاده کنید الزامی است): تعیین پسورد اکانت پارامتر SQLSVCACCOUNT.

باتوجه به سیاست‌ها نصب می‌توانید از پارامترها دیگری نیز استفاده کنید. بعنوان مثال پارامترهای زیر برای نصب روی سیستمی که نام کاربری و پسورد انرا نداریم مناسب است:

```
setup.exe /q /Action=Install /Features=SQL /InstanceName=SQLExpress /SECURITYMODE=SQL /SAPWD="1234567"
/SQLSYSADMINACCOUNTS="Builtin\Administrators" /SQLSVCACCOUNT="NT AUTHORITY\SYSTEM"
/SQLSVCSTARTUPTYPE="Automatic" /TCPENABLED=1
```

نکته: اگر مقدار پارامتر SECURITYMODE برابر SQL باشد حتما باید پارامتر SAPWD مقداردهی شود.

نکته: اگر مقدار TCPENABLED برابر 1 باشد پروتکل TCP/IP فعال می‌شود. اگر هیچ نمونه‌ی دیگری روی سیستم نصب نباشد مقدار TCP Port برابر 1433 است، درغیر اینصورت یک مقدار تصادفی تولید می‌شود.

نکته: برای خوانایی بیشتر، پارامترهای فوق در چند خط نوشته شده اند. برای اجرای صحیح در یک فایل bat، همه انها باید در یک خط باشند.

برای اولین بهتر است از پارامتر HideConsel/ استفاده نکنید تا خطای احتمالی رو مشاهده کنید. برای آپگرید کردن نیز می‌توانید از دستور زیر استفاده کنید:

```
Setup.exe /q /Hideconsole /ACTION=upgrade /INSTANCENAME=SQLExpress
```

برای مشاهده دیگر پارامترها به مستندات MSDN مراجعه کنید. همچنین می‌توان نصب از طریق فایل Configuration را نیز انجام داد.

امیدوارم مفید واقع شده باشد.

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۷/۲۲ ۹:۱

ممنون از شما. برای کارهای سبک استفاده از SQL Server CE شاید مناسب‌تر باشد یا حتی LocalDB .

نویسنده: جوادی
تاریخ: ۱۳۹۲/۰۷/۲۳ ۰:۳

خواهش می‌کنم.
بله، بسته به سناریو می‌شود از پایگاه داده‌های سبکتر نیز استفاده کرد.
LocalDB متاسفانه محدودیت‌های زیادی دارد، مثلا در XP ساپورت نمی‌شود، فقط توسط Sql server 2012 management studio در دسترس هست که برای محصولات عمومی مناسب نیست. البته محل استفاده خاص خودش را دارد. باز هم همه چیز به نحوه و محل استفاده بستگی دارد.

نویسنده: افشار محبی
تاریخ: ۱۳۹۲/۰۸/۰۸ ۱۵:۴۳

شما که زحمت نوشتن این مقاله را کشیده اید کاش از نسخه ۲۰۱۲ استفاده می‌کردید.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۸/۰۸ ۱۷:۳

به نظر ایشون بیشتر نگارش‌هایی مدنظرشون هست که روی XP هم قابل اجرا باشند که نگارش 2012 جزو آن‌ها نیست .

نویسنده: افشار محبی
تاریخ: ۱۳۹۲/۰۸/۰۸ ۱۹:۲

بله درسته. من اصلاً حواسم به پشتیبانی از xp و کامپیوترهای قدیمی نبود. ویندوز xp هم معضلی شده برای خودش. از یک طرف نمی‌توان پشتیبانی را قطع کرد، چون تعداد زیادی کامپیوتر در داخل کشور هنوز در همین حد هستند از طرف دیگر پشتیبانی از فناوری به این حد قدیمی چندان عاقلانه نیست.

البته یک راه دیگر هم هست. استفاده از db engineهای کوچکتر که معمولا embeded هستند. مثل sqlite و شاید sql ce و بقیه.

نویسنده: جوادی
تاریخ: ۱۳۹۲/۰۸/۰۸ ۱۹:۵۹

من نسخه 2012 رو تست نکردم اما گمان نکنم تفاوتی داشته باشه؛ یعنی بعد از اکسترکت کردن با استفاده از Command Prompt و ارسال پارامتر به اون میشه یک Instance نصب کرد.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۸/۱۳ ۱۸:۵۵

یک مقایسه خوب هم در اینجا هست که می‌تونه به انتخاب نگارش مناسب کمک کنه:

SQL Express v LocalDB v SQL Compact Edition

نویسنده: یاسین
تاریخ: ۱۳۹۲/۱۲/۰۲ ۱۵:۲۶

سلام، ببینید من ورژن‌های متعددی که برای دانلود نسخه 2012 قرار داده شده رو به همراه حجمشون پایین نوشتم. به عنوان مبتدی می‌خواستم بدونم چه وقت از کدام یک از اینها استفاده میشه؟
یک حالتی هست که ویژوال استودیو از من می‌خواد نسخه اکسپرس رو نصب کنم، خوب نسخه ای با حجم و امکانات بالا رو نصب می‌کنم. یک حالتی هم هست که بصورت توکار در یک برنامه قرار داده میشه که کاربر نهایی در اجرای برنامه به مشکل برنخوره، حالا چطور متوجه بشیم که کدوم نسخه برای کاربر نهایی مناسبه.
تشکر

SQL Server Express with Tools (with LocalDB, Includes the database engine and SQL Server Management Studio Express) 1.4G

```
SQL Server Management Studio (Tools only) 939MB
```

```
SQL Server Express LocalDB (MSI installer) 24MB
```

```
SQL Server Express with Advanced Services (contains the database engine, Express Tools, Reporting Services, and Full Text Search) 1.7G
```

```
SQL Server Express (Containing only the database engine) 133MB
```

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۲/۰۲ ۱۸:۵۴

نگارش‌های اکسپرس و local db رایگان هستند. نگارش CE یا Compact edition نیز رایگان است. سایر نگارش‌ها خیر. بنابراین خیلی‌ها در سراسر دنیا بر این اساس تصمیم گیری می‌کنند. نگارش CE داخل پروسه برنامه اجرا میشه و تا 4 گیگ دیتا داخلش می‌تونید ذخیره کنید. نگارش اکسپرس تا 12 گیگ محدودیت داره. سایر نگارش‌ها محدودیتی ندارند. خلاصه باید به کسب و کار خودتون نگاه کنید. برنامه حسابداری هست که تعداد رکوردهاش تا آخر سال مثلا میشه 10 هزار رکورد؟ نگارش CE براش کافی هست.

نویسنده: یاسین
تاریخ: ۱۳۹۲/۱۲/۰۳ ۱۱:۱۳

ممنون، برای کاربران نهایی که از ویندوز XP استفاده می‌کنند، آیا نسخه 2008 مناسب هست یا اینکه تا نسخه 2005 باید تنزل کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۳ ۱۱:۱۶

نگارش 2008 با XP سازگار است. نگارش‌های پس از آن خیر.

نویسنده: جمال
تاریخ: ۱۳۹۲/۱۲/۰۳ ۱۸:۱۸

میشه لطفا روشی را برای نصب خودکار بانک اطلاعاتی بهنگام نصب برنامه‌ی اصلی معرفی نمایید.
چون در روال عادی برنامه‌های دات نت این کار باید به صورت دستی انجام بشه.

نویسنده: شهریار
تاریخ: ۱۳۹۲/۱۲/۰۳ ۲۱:۴۱

بهترین کار برای نصب خودکار فایلهای مورد نیاز جهت اجرای برنامه‌های مبتنی بر دات نت ، استفاده از نرم افزارهای ساخت

Setup می‌باشد. از قبیل

1- InstallShield

2- InstallAware

3- Advanced Installer

مورد اول ، از قدیمی‌ترین نرم افزارها و حرفه ای می‌باشد. ولی متاسفانه در تهیه نسخه به روز آن کمی دچار مشکل خواهید شد.

مورد دوم ادعای مقایسه با اینستال شیلد را دارد .از عیوبی که من در استفاده از این نرم افزار دیدم میتوان به حجم بالای نرم افزار اصلی اشاره کرد که بیش از دو گیگابایت است و هر دفعه نسخه جدید اومد شما باید مجدداً این حجم را دانلود کنید.

مورد سوم که بهترین گزینه نیز می‌باشد ، بسیار خوش دست و سبک می‌باشد. و به راحتی تمام موارد مورد نیاز جهت اجرای برنامه‌های شما را نصب می‌نماید.

عنوان: **تبدیل اعداد صحیح و اعشاری به حروف در T-SQL با استفاده از Join**
نویسنده: حمیدرضا عابدینی
تاریخ: ۱۳۹۲/۰۸/۰۱ ۸:۵۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, T-SQL, SQL Server 2008, Number To Word

استفاده شده از SQL 2008

روش کار : 1- دریافت پارامتر ورودی به صورت رشته
2- درج عناوین اعداد، ارزش مکانی اعداد صحیح و اعشاری هرکدام در یک جدول
3- جدا کردن ارقام صحیح و اعشاری
4- جداکردن سه رقم سه رقم اعداد صحیح و انتقال آنها به جدول مربوطه
5- Join جداول عناوین و ارقام جدا شده
6- ارسال ارقام اعشاری به همین تابع
7- مشخص کردن ارزش مکانی رقم اعشار
8- اتصال رشته حروف صحیح و اعشاری

در آخر این مطلب کد این تابع را به صورت کامل، برای دانلود قرار داده ام.

بررسی قسمت‌های مختلف کد

برای اینکه محدودیتی در تعداد ارقام صحیح و اعشاری نداشته باشیم، پارامتر ورودی را از نوع VARCHAR می‌گیریم. پس باید ورودی را بررسی کنیم تا رشته عددی باشد.

بررسی رشته ورودی:

```
-- @pNumber   پارامتر ورودی

IF LEN(ISNULL(@pNumber, '')) = 0  RETURN NULL

IF (PATINDEX('%[^0-9.-]%', @pNumber) > 0)
   OR (LEN(@pNumber) -LEN(REPLACE(@pNumber, '-', '')) > 1)
   OR (LEN(@pNumber) -LEN(REPLACE(@pNumber, '.', '')) > 1)
   OR (CHARINDEX('-', @pNumber) > 1)
RETURN 'خطا'

IF PATINDEX('%[^0]%', @pNumber) = 0  RETURN 'صفر'
IF (CHARINDEX('.', @pNumber) = 1) SET @pNumber='0'+@pNumber

DECLARE @Negative AS VARCHAR(5) = '';
IF LEFT(@pNumber, 1) = '-'
BEGIN
    SET @pNumber = SUBSTRING(@pNumber, 2, 100)
    SET @Negative = 'منفی '
END
```

- بررسی NULL ، خالی بودن و یا داشتن فاصله در رشته، با دانستن اینکه تابع LEN فاصله‌های آخر یک رشته را درنظر نمی‌گیرد.
- بررسی رشته ورودی برای پیدا کردن کاراکتر غیر عددی، نقطه و منفی. بررسی تعداد علامت منفی و نقطه که بیشتر از یک مورد نباشند، و در نهایت بررسی اینکه علامت منفی در ابتدای رشته ورودی باشد.
- بررسی صفر بودن ورودی(0)، مقدار ورودی شروع شونده با ممیز(0213.) و مقدار عددی منفی(-21210.0021).
چیز دیگری به ذهنم نرسید!

درج عناوین در جداول مربوطه:

فکر کنم اینجا به علت وجود کاراکترهای فارسی و انگلیسی کد کمی بهم ریخته نمایش داده می‌شود.

```
DECLARE @NumberTitle TABLE (val  INT,Title NVARCHAR(100));
```

```
INSERT INTO @NumberTitle (val,Title)
VALUES(0, ''),(1, 'چهار' ,4),('سه' ,3),('دو' ,2), ('یک')
,(5, 'هشت' ,8),('هفت' ,7),('شش' ,6),('پنج')
,(9, 'دوازده' ,12),('یازده' ,11),('ده' ,10),('نه')
,(13, 'شانزده' ,16),('پانزده' ,15),('چهارده' ,14),('سیزده')
,(17, 'بیست' ,20),('نوزده' ,19),('هجده' ,18),('هفده')
,(30, 'شصت' ,60),('پنجاه' ,50),('چهل' ,40),('سی')
,(70, 'صد' ,100),('نود' ,90),('هشتاد' ,80),('هفتاد')
,(200, 'پانصد' ,500),('چهارصد' ,400),('سیصد' ,300),('دویست')
,(600, 'نهصد' ,900),('هشتصد' ,800),('هفتصد' ,700),('ششصد')

DECLARE @PositionTitle TABLE (id  INT,Title NVARCHAR(100));
INSERT INTO @PositionTitle (id,title)
VALUES (1, ''),(2, 'تریلیون' ,5),('میلیارد' ,4),('میلیون' ,3),('هزار')
,(6, 'سپتیلیون' ,9),('سیکستیلون' ,8),('کوینتیلیون' ,7),('کوادریلیون')
,(10, 'دسیلیون' ,12),('نونیلیون' ,11),('اکتیلیون')
,(13, 'تریدسیلیون' ,15),('دودسیلیون' ,14),('آندسیلیون')
,(16, 'سیکسدسیلیون' ,18),('کویندسیلیون' ,17),('کواتردسیلیون')
,(19, 'نومدسیلیون' ,21),('اکتودسیلیوم' ,20),('سپتندسیلیون')

DECLARE @DecimalTitle TABLE (id  INT,Title NVARCHAR(100));
INSERT INTO @DecimalTitle (id,Title)
VALUES( 1 ,'هزارم' , 3),('صدم' , 2),( 'دهم')
,(4 , 'میلیون ام' , 6),('صد-هزارم' , 5),('ده-هزارم')
,(7 , 'میلیاردم' , 9),('صد-میلیون ام' , 8),('ده-میلیون ام')
,(10 , 'ده-میلیاردم')
```

جداسازی رقم صحیح و اعشاری: عدد ورودی ممکن است حالت‌های مختلفی داشته باشد مثل: .00002 , 0.000000 , .123 , 234.434400000000

بنابراین براساس ممیز، قسمت صحیح را از اعشاری جدا می‌کنیم. برای ورودی که با ممیز شروع شود، در ابتدا تابع بررسی می‌کنیم و عدد صفر را به رشته اضافه می‌کنیم.

بعد از ممیز و اعداد بزرگتر از یک، **با صفرهای بی ارزش چه کنیم؟** شاید اولین چیزی که به ذهن برسد استفاده از حلقه (WHILE) برای حذف صفرهای بی ارزش قسمت ممیز باشد؛ ولی من ترجیح می‌دهم که از روش دیگری استفاده کنم :

برعکس کردن رشته قسمت اعشاری، پیدا کردن مکان اولین عدد غیر صفر منهای یک ، و کم کردن عدد بدست آمده از طول رشته اعشاری، قسمت مورد نظر ما را برخواهد گرداند:

```
SUBSTRING(@DecimalNumber,1, len(@DecimalNumber )-PATINDEX('%[^0]%', REVERSE (@DecimalNumber))-1)
```

اما اگر عدد ورودی **20.0** باشد همچنان صفر بی ارزش بعداز ممیز را خواهیم داشت. برای رفع این مشکل کافی است که **کاراکتری غیر از صفر** را به اول رشته اعشاری اضافه کنیم. من از علامت **'?'** استفاده کردم. پس به علت اضافه کردن کاراکتر، استارت را از 2 شروع کرده و دیگر نیازی به -1 نخواهیم داشت. با کد زیر قسمت صحیح و اعشاری را بدست می‌آوریم:

```
DECLARE @IntegerNumber NVARCHAR(100),
@DecimalNumber NVARCHAR(100),
@PointPosition INT =case CHARINDEX('.', @pNumber) WHEN 0 THEN LEN(@pNumber)+1 ELSE CHARINDEX('.',
@pNumber) END

SET @IntegerNumber= LEFT(@pNumber, @PointPosition - 1)
SET @DecimalNumber= '?' + SUBSTRING(@pNumber, @PointPosition + 1, LEN(@pNumber))
SET @DecimalNumber=  SUBSTRING(@DecimalNumber,2, len(@DecimalNumber )-PATINDEX('%[^0]%', REVERSE
(@DecimalNumber)))

SET @pNumber= @IntegerNumber
```

جداد کردن سه رقم سه رقم : - بدست آوردن یکان، دهگان و صدگان

- برای قسمت دهگان، اگر عددی بین 10 تا 19 باشد به صورت کامل (مثلا 15) و در غیر این صورت فقط رقم دهگان. برای بدست آوردن یکان اگر دو رقم آخر بین 10 و 19 بود صفر و در غیر این صورت یکان برگردانده می‌شود و در جدول MyNumbers درج می‌گردد.

```
DECLARE @Number AS INT
DECLARE @MyNumbers TABLE (id INT IDENTITY(1, 1), Val1 INT, Val2 INT, Val3 INT)

WHILE (@pNumber) <> '0'
BEGIN
    SET @number = CAST(SUBSTRING(@pNumber, LEN(@pNumber) -2, 3)AS INT)

INSERT INTO @MyNumbers
SELECT (@Number % 1000) -(@Number % 100),
CASE
WHEN @Number % 100 BETWEEN 10 AND 19 THEN @Number % 100
ELSE (@Number % 100) -(@Number % 10)
END,
CASE
WHEN @Number % 100 BETWEEN 10 AND 19 THEN 0
ELSE @Number % 10
END

    IF LEN(@pNumber) > 2
        SET @pNumber = LEFT(@pNumber, LEN(@pNumber) -3)
    ELSE
        SET @pNumber = '0'
END
```

سطری که تمام مقادیر آن صفر باشد برای ما بی ارزش محسوب می‌شود، مانند سطر یک در عکس زیر (جدول MyNumbers) برای عدد **1200955000** :

	id	Val1	Val2	Val3
1	1	0	0	0
2	2	900	50	5
3	3	200	0	0
4	4	0	0	1

استفاده از JOIN : JOIN کردن جدول اعداد با عناوین عددی براساس ارزش آن‌ها و JOIN جدول اعداد با جدول ارزش مکانی براساس ID به صورت نزولی(شماره سطر).

```
DECLARE @Str AS NVARCHAR(2000) = '';
SELECT @Str += REPLACE(REPLACE(LTRIM(RTRIM(nt1.Title + ' ' + nt2.Title + ' ' + nt3.title)),'  ',' '),'
', ' و ')
       + ' ' + pt.title + ' و '
FROM   @MyNumbers  AS mn
       INNER JOIN @PositionTitle pt
            ON  pt.id = mn.id
       INNER JOIN @NumberTitle nt1
            ON  nt1.val = mn.Val1
       INNER JOIN @NumberTitle nt2
            ON  nt2.val = mn.Val2
       INNER JOIN @NumberTitle nt3
            ON  nt3.val = mn.Val3
WHERE  (nt1.val + nt2.val + nt3.val > 0)
ORDER BY pt.id DESC
```

Replace داخلی: جایگزین کردن "دو فاصله‌ی خالی" با "یک فاصله‌ی خالی"

Replace بیرونی: جایگزینی فاصله‌های خالی با ' و '

همانطور که در بالا اشاره کردم سطرهایی که val2,val1 و val3 آن صفر باشد برای ما بی ارزش هستند، پس آنها را با شرط نوشته شده حذف می‌کنیم.

بدست آوردن مقدار اعشاری: خوب! حالا نوبت به عدد اعشاری می‌رسد. برای بدست آوردن حروف، مقدار اعشاری بدست آمده

را به همین تابع ارسال می‌کنیم و برای بدست آوردن عنوان ارزش مکانی، براساس طول اعشار (ID) آن را در جدول مربوطه پیدا می‌کنیم.

اگر عدد ورودی مثلا 0.355 باشد، تابع باید صفر اول را شناسایی و قسمت عناوین اعشاری را به آن اضافه کند، که این کار با شرط ذیل انجام می‌شود.

اگر رشته اعشار بدون مقدار باشد، تابع مقدار NULL بر می‌گرداند (قسمت بررسی رشته ورودی) و هر رشته ای که با NULL جمع شود برابر با NULL خواهد بود. در این صورت با توجه به کد زیر مقداری به رشته Str به عنوان قسمت اعشاری، اضافه نمی‌گردد.

```
IF @IntegerNumber='0'
SET @Str=CASE WHEN PATINDEX('%[^0]%', @DecimalNumber) > 0 THEN @Negative ELSE '' END + 'صفر'
ELSE
SET @Str = @Negative  + LEFT (@Str, LEN(@Str) -2)

DECLARE @PTitle NVARCHAR(100)=ISNULL((SELECT Title FROM @DecimalTitle WHERE id=LEN(@DecimalNumber)),'')
SET @Str += ISNULL(' ممیز '+[dbo].[fnNumberToWord_Persian](@DecimalNumber) +' '+@PTitle,'')
RETURN @str
```

مثال: رشته ' **5445789240.54678000000000** '

پنج میلیارد و چهارصد و چهل و پنج میلیون و هفتصد و هشتاد و نه هزار و دویست و چهل ممیز پنجاه و چهار هزار و ششصد و هفتاد و هشت صد-هزارم

دانلود فایل

نظرات خوانندگان

نویسنده: میثم چگینی
تاریخ: ۱۳۹۲/۰۸/۰۱ ۱۱:۴

با تشکر
یک مورد کوچک داره که برای نمایش درست نیاز به اضافه کردن N قبل از تمام کلمات فارسی می‌باشد
N'صفر'

نویسنده: حمیدرضا عابدینی
تاریخ: ۱۳۹۲/۰۸/۰۳ ۱۰:۵۳

حق با شماست!
من چون از Collation نوع Persian_100_CI_AI استفاده کردم همه‌ی Nها را حذف کردم. ولی برای دیگر موارد برای اینکه به صورت علامت ؟ ظاهر نشود، همانطور که شما گفتید باید از N استفاده کرد.

ممنون

نویسنده: مهیار
تاریخ: ۱۳۹۲/۱۱/۱۵ ۲۰:۳۸

سلام،
ممنون از راه حلی که ارائه دادید.

برای حذف صفرهای غیرضروری موجود در اعشار پیشنهاد من کد زیر است که ساده‌تر و خواناتر است:

```
declare @value varchar(50) ='0.1010000000'
select replace(rtrim(replace(@value,'0',' ')),' ','0');
```

توضیح کد: ابتدا تمام صفرهای موجود در رشته را تبدیل به کاراکتر Space میکنیم توسط تابع Replace سپس با تابع Rtrim تمام کاراکترهای Space آخر رشته را Remove می‌کنیم و مجدد کاراکترهای Space را برمیگردانیم به صفر.

نویسنده: مهیار
تاریخ: ۱۳۹۲/۱۱/۱۵ ۲۱:۳۷

کد مربوط به حلقه While اتان هم کمی ساده‌تر کردم مخصوصا اولین دستور حلقه که مربوط میشه به انتخاب سه رقم آخر رشته. ضمنا نیازی نیست که صراحتا متغیر را به integer تبدیل کنید. با صفر جمع یا با یک ضرب کنید تا بصورت Implicit تبدیل صورت بگیره:

```
WHILE (@pNumber) <> '0'
BEGIN
    SET @number = RIGHT(@pNumber, 3) + 0

INSERT INTO @MyNumbers
SELECT
     @Number / 100 * 100,
 CASE
   WHEN nbr BETWEEN 10 AND 19 THEN nbr
   ELSE nbr / 10 * 10
 END,
 CASE
   WHEN nbr BETWEEN 10 AND 19 THEN 0
   ELSE nbr % 10
 END
FROM (SELECT @Number % 100)S(nbr);

    IF LEN(@pNumber) > 2
```

```
        SET @pNumber = LEFT(@pNumber, LEN(@pNumber) -3)
    ELSE
        SET @pNumber = '0'
END
```

نویسنده: حمیدرضا عابدینی
تاریخ: ۱۳۹۲/۱۱/۲۰ ۲۳:۳۲

سلام مهیار جان!

کد بسیار خوب و خوانایی گذاشتید، ممنونم!

در نتیجه:

```
DECLARE @IntegerNumber NVARCHAR(100),
@DecimalNumber NVARCHAR(100),
@PointPosition INT =case CHARINDEX('.', @pNumber) WHEN 0 THEN LEN(@pNumber)+1 ELSE CHARINDEX('.',
@pNumber) END

SET @pNumber=replace(rtrim(replace(@pNumber,'0',' ')),' ','0');
SET @IntegerNumber= LEFT(@pNumber, @PointPosition - 1)
SET @DecimalNumber= SUBSTRING(@pNumber, @PointPosition+1 , LEN(@pNumber))

SET @pNumber= @IntegerNumber
```

عنوان: **SQL Instance**
نویسنده: م منفرد
تاریخ: ۱۳۹۲/۱۰/۱۲ ۱۹:۲۵
آدرس: www.dotnettips.info
گروه‌ها: C#, SQL Server, Registry

ممکن است کاربر بر روی سیستم خود نسخه‌های مختلفی از SQL Server را نصب کرده باشد. برای مثال SQL Express, SQL 2005, SQL 2008. و یا نسخه ای خاص (مثلا 2012) را چند بار روی سیستم خود نصب کرده باشد. SQL برای تفکیک این نسخه‌ها و نصب‌ها از مفهومی با عنوان Instance استفاده می‌کند. یعنی به هر نسخه نصب شده نامی یکتا می‌دهد تا بتوان به تفکیک به آنها دسترسی داشت.

برای اتصال به این نسخه‌ها باید در بخش آدرس سرور، از ترکیب نام سیستم و نام Instance به این شکل استفاده کرد:

SystemName\Instance

بعضی مواقع لازم است که لیست Instanceهای نصب شده روی سیستم کاربر را به دست آوریم. ADO.NET کلاسی به همین منظور تعبیه کرده که شبکه را جستجو کرده و SQL Instanceهای مختلف را که قابل دسترسی هستند را برای شما لیست می‌کند. استفاده از این کلاس بسیار ساده است:

```
using System.Data.Sql;

class Program
{
  static void Main()
  {
    // Retrieve the enumerator instance and then the data.
    SqlDataSourceEnumerator instance =
      SqlDataSourceEnumerator.Instance;
    System.Data.DataTable table = instance.GetDataSources();

    // Display the contents of the table.
    DisplayData(table);

    Console.WriteLine("Press any key to continue.");
    Console.ReadKey();
  }

  private static void DisplayData(System.Data.DataTable table)
  {
    foreach (System.Data.DataRow row in table.Rows)
    {
      foreach (System.Data.DataColumn col in table.Columns)
      {
        Console.WriteLine("{0} = {1}", col.ColumnName, row[col]);
      }
      Console.WriteLine("============================");
    }
  }
}
```

البته با توجه به اینکه شبکه را جستجو می‌کند در نرم افزار شما وقفه خواهد انداخت. خوب اگه بخواهیم Instanceهای نصب شده روی سیستم کاربر را پیدا کنیم چی؟ ساده‌ترین و سریعترین راه استفاده از رجیستری سیستم است. نام Instanceها در رجیستری ویندوز در آدرس زیر قابل دسترسی است:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Microsoft SQL Server\Instance Names
```

برای استفاده از این کلید در c# می‌توان از کد زیر کمک بگیرید:

```
var key = Registry.LocalMachine.OpenSubKey(@"SOFTWARE\Microsoft\Microsoft SQL Server\Instance Names");

            foreach (string sk in key.GetSubKeyNames())
            {
                var rkey = key.OpenSubKey(sk);
                foreach (string s in rkey.GetValueNames())
                {
                    MessageBox.Show("Sql instance name:" + s);
                }
            }
```

فقط دو نکته قابل توجه است. برنامه باید در Any CPU کامپایل شود تا در سیستم‌های 64 بیتی بتوانید به محل درست رجیستری دسترسی پیدا کنید. چون نرم افزارهای 32 بیت در ویندوز 64 بیت در سیستم wow64 اجرا می‌شود که دسترسی به رجیستری آن در آدرس wow64 هر قسمت رجیستری است. بنابراین کد فوق در حالت Any CPU و غیر فعال بودن Prefer 32-bit قسمت Build در Properties برنامه به درستی اجرا می‌شود.

نکته: Default Instance در SQL مقدار MSSQLSERVER می‌باشد.

عنوان: **خواندن سریع اطلاعات فایل اکسل و ذخیره در بانک SQL**
نویسنده: مجتبی فخاری
تاریخ: ۱۳۹۲/۱۱/۱۷ ۱۸:۳۵
آدرس: www.dotnettips.info
گروه‌ها: C#, SQL Server, Excel

یکی از زمانبرترین عملیاتها در نرم افزارهای اتوماسیون، خواندن اطلاعات از فایل‌های اکسل با حجم بالا است. در صورتی که این کار را می‌توان با استفاده از کلاس SqlBulkCopy به سرعت انجام داد. در ادامه نحوه استفاده از این کلاس، همراه نمونه کدها آورده شده است.

توضیحات به صورت Comment است.

```
try
            {
//انتخاب فایل اکسل
                OpenFileDialog ofd = new OpenFileDialog();
                ofd.Title = "انتخاب فایل اکسل حاوی اطلاعات";
                ofd.Filter = "2003 فایل اکسل (*.xls)|*.xls|فایل اکسل 2007 به بعد(*.xlsx)|*.xlsx";
                if (ofd.ShowDialog() == DialogResult.OK)
                {
                    string excelConnectionString = "";
                    string SourceFilePath = ofd.FileName;
//ایجاد کانکشن استرینگ برا خواندن کل اطلاعات از فایل اکسل و ریختن آنها در یک دیتاتیبل به نام dt
                    if (System.IO.Path.GetExtension(ofd.FileName) == ".xlsx")
//تشخیص نوع فایل اکسل برای ایجاد کانکشن استرینگ برای نسخه‌های مختلف اکسل
                        excelConnectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" +
SourceFilePath + ";Extended Properties=Excel 12.0";
                    else if (System.IO.Path.GetExtension(ofd.FileName) == ".xls")
                        excelConnectionString = "Provider=Microsoft.Jet.Oledb.4.0;Data Source=" +
SourceFilePath + ";Extended Properties=Excel 8.0";

                    DataTable dt = new DataTable("tblinfos");


                    using (System.Data.OleDb.OleDbConnection connection = new
System.Data.OleDb.OleDbConnection(excelConnectionString))
                    {

                        connection.Open();

                        System.Data.OleDb.OleDbDataAdapter da = new
System.Data.OleDb.OleDbDataAdapter("SELECT * FROM [List$]", connection);//List نام شیت در فایل اکسل
است

                        da.Fill(dt);
                        connection.Close();

                    }

                    using (System.Data.OleDb.OleDbConnection connection = new
System.Data.OleDb.OleDbConnection(excelConnectionString))
                    {
                        this.Cursor = Cursors.WaitCursor;

                        connection.Open();
//ایجاد ارتباط با بانک اس کیو ال
                        string sqlConnectionString = @"Data
Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\BankDPR.mdf;Max Pool Size=6000; Connection
Timeout=50;Integrated Security=True;User Instance=True";


                        Dataaccess db = new Dataaccess();
                        DataTable dtr = db.select("Select Top(1) * From tblinfos");//بدست آورن نام
ستون‌های جدول مورد نظر برای تطبیق با ستونهای فایل اکسل
                        //*******************   Fast Copy
                        using (System.Data.SqlClient.SqlBulkCopy bulkCopy = new
System.Data.SqlClient.SqlBulkCopy(sqlConnectionString,
System.Data.SqlClient.SqlBulkCopyOptions.KeepIdentity))//تعریف یک شی از کلاس SqlBulkCopy
                        {
                            for (int i = 1; i < dtr.Columns.Count; i++)
                            {
                                bulkCopy.ColumnMappings.Add(dt.Columns[i - 1].Caption,
dtr.Columns[i].Caption);//نام ستونهای فایل اکسل با نام ستونهای جدول ColumnMappings با استفاده از خاصیت
تطبیق داده می‌شود sql مورد نظر بانک
                            }

                            bulkCopy.DestinationTableName = "tblinfos";//مشخص نمودن نام جدول که قرار
```

```
است اطلاعات درون ان کپی گردد
                            bulkCopy.WriteToServer(dt);//به dt انجام عملیات کپی اطلاعات از دیتاتیبل با نام
بانک

                        }

                        this.Cursor = Cursors.Arrow;
                        MessageBox.Show("اطلاعات از فایل شما خوانده شد");

                    }
                }
            }
            catch (Exception ex)
            {
                this.Cursor = Cursors.Arrow;

                MessageBox.Show(ex.Message, "error");
            }
```

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۱۷ ۲۱:۲۲

احتمالا این قطعه کد مستقیما از داخل سورس یکی از پروژه‌های شما بیرون آمده (مثال نیست؛ واقعی هست). می‌شد کمی اون رو refactor کرد مثلا یک متد از داخلش بیرون آورد که این متد داخلش مسیج باکس نباشه یا باز کردن یک صفحه دیالوگ و تغییر کرسر. try و catch هم نداشته باشه چون باید در یک سطح بالاتر catch بشه مشکلاتش. اون selectها مثلا می‌شدند چند پارامتر، برای اینکه این کد قابلیت استفاده مجدد بهتری پیدا کنه. یا مثلا اون sqlConnectionString از داخل کدها بیرون می‌اومد و می‌شد یک پارامتر جدید. نمایش کرسر هم داخل این متد قرار نمی‌گرفت. نام جدول نهایی هم مثلا یک پارامتر دیگر می‌شد برای سهولت استفاده مجدد و همچنین تست بهتر یک قطعه کوچک از کار. خود متد اصلی هم می‌شد دو متد کوچک‌تر؛ یکی کار load رو انجام می‌داد و دیگری کار insert سریع.

نویسنده: پالادین
تاریخ: ۱۳۹۲/۱۱/۱۹ ۱۳:۳۸

من هم با دوستمون محسن خان موافقم. اتفاق‌ها زیادی در مورد این کد باید بیافته تا به یک کد خوب تبدیل بشه. شما کاملا به صورت Smart UI کد زدید. امیدوارم وقت بزارید و این کد رو بهینه کنید. توی همین سایت مثال‌های خوبی واسه‌ی یادگیری هست..

نویسنده: مجتبی فخاری
تاریخ: ۱۳۹۲/۱۱/۲۰ ۱۱:۳۶

بله این کد رو من دقیقا از یکی از برنامه هام آوردم و مال چندین سال پیشه.
اما خوبیش اینه که بدون خطا جواب میده و فقط کافیه اسم جداول رو عوض کنید.

نویسنده: محمدی
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۰:۲۹

با سلام و تشکر. این برنامه پیغام میدهد List وجود ندارد.

```
SELECT * FROM [List$]
```

لطفا راهنمایی بفرمایید من چطور میتوانم نام شیت‌ها را بدست بیاورم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۰:۴۷

در کد فوق اکثر قسمت‌ها بر اساس یک سری پیش فرض مشخص تهیه شده‌اند و آن‌ها را تبدیل به پارامتر متغیر نکرده‌اند. نام شیت، همان نام برگه جاری است:

نویسنده: احمد زواری
تاریخ: ۱۳۹۳/۰۴/۱۱ ۲۳:۴۰

سلام. من از این روش برای خواندن فایل موجود در آدرس http://www.site.com/market/export.aspx?type=dtod&date=20120405 استفاده کردم، ولی متاسفانه فیلدهای عددی را NULL مقدار میگیرد. مشکل از چیست و چه جوری برطرف میشود؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۲ ۰:۲۴

این فایل استاندارد اکسل نیست. خروجی آن را با نوت پد باز کنید؛ یک فایل HTML معمولی است.

نویسنده: احمد زواری
تاریخ: ۱۳۹۳/۰۴/۱۲ ۳:۳۰

الان من نیاز شدید دارم این فایل رو تبدیل به SQL کنم، و چون هر روز این فایل تغییر میکنه نمیتونم توی فایل تغییری بدم، چه راه حلی هست برای این کار؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۲ ۹:۱۹

فایل‌های HTML را با استفاده از کتابخانه‌ی HTML Agility Pack پردازش می‌کنند.

عنوان: **افزودن اکانت مدیریتی فراموش شده به SQL Server**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۹ ۱:۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Security

فرض کنید
- تمام اکانت‌های مدیریتی توکار SQL Server را حذف کرده‌اید (یا برایتان حذف کرده‌اند).
- بجز کاربر SA، تمام کاربران را از نقش SYSADMIN حذف کرده‌اید؛ شامل تمام اکانت‌های ویندوزی و همچنین خود SQL Server.
- پسورد SA را هم فراموش کرده‌اید یا ندارید.

خوب، الان چکار می‌خواهید بکنید؟!
احتمالا نصب مجدد سرور را پیشنهاد دهید یا بانک اطلاعاتی Master را بازسازی کنید که در هر دو حالت تمام تنظیمات سرور را از دست خواهید داد. روش دیگری هم بدون از دست دادن تنظیمات سرور وجود دارد که در ادامه آن‌را بررسی خواهیم کرد.

افزودن یک اکانت مدیریتی جدید به SQL Server

اگر دسترسی کامل مدیریتی خود را به SQL Server از دست داده‌اید باید ابتدا به آن سرور لاگین کنید؛ با این فرض که کاربر وارد شده به سیستم، جزو local administrators group است.

```
C:\>sqlcmd -S .
1> create login [MachineName\TestUser] from windows;
2> go
1> exec sp_addsrvrolemember 'MachineName\TestUser','sysadmin'
2> go
1> select is_srvrolemember('sysadmin', 'MachineName\TestUser')
2> go
-----------
1
(1 rows affected)
1>
```

سپس خلاصه مواردی را که ملاحظه می‌کنید، اجرای سه دستور است:
الف) اجرای sqlcmd با پارامتر S و مشخص سازی وهله‌ی مورد نظر
البته حالت کامل آن در صورتیکه از وهله‌ی پیش فرض استفاده نمی‌کنید SQLCMD –S Server_Name\Instance_Name است. S نیز در اینجا باید با حروف بزرگ نوشته شود.
ب) create login را بر روی یکی از اکانت‌های محلی سیستم اجرا کنید. مثلا MachineName\administrator یا هر اکانت موجود دیگری که لازم است. نام آن نیز باید حتما به شکل server_name\user_name باشد.
در حین استفاده از sqlcmd، هر فرمان زمانی اجرا می‌شود که در سطر پس از آن، یک go نوشته شده و enter شود.
ج) سپس توسط sp_addsrvrolemember به این اکانت اضافه شده، دسترسی sysadmin بدهید.

برای آزمایش آن فقط کافی است از متد is_srvrolemember برای کوئری گرفتن استفاده کنید و یا سعی کنید توسط اکانت اضافه شده، به SQL Server لاگین کنید. این اکانت اکنون در قسمت Security/logins قابل مشاهده است.

اگر نمی‌خواهید از اکانت‌های ویندوزی استفاده کنید، create login را به نحو ذیل مقدار دهی کنید:

```
C:\>sqlcmd -S .
1> use master
2> go
Changed database context to 'master'.
1> create login test_user with password='123#123'
2> go
1> exec sp_addsrvrolemember 'test_user','sysadmin'
2> go
1>
```

سپس به این کاربر اضافه شده با کلمه‌ی عبور مشخص، توسط sp_addsrvrolemember دسترسی sysadmin بدهید.

نظرات خوانندگان

نویسنده: وحید
تاریخ: ۱۳۹۲/۱۲/۱۱ ۱۳:۱۹

با سلام؛ دستور خط اول را که می‌نویسم ارور زیر را می‌دهد

```
Sqlcmd: Error: Microsoft SQL Server Native Client 11.0 : Login failed for user 'NARM-2-PC\Narm-2'..
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۱ ۱۴:۸

در متن قید شده «... با این فرض که کاربر وارد شده به سیستم، جزو local administrators group است ...». یعنی:

عنوان: **اندازه گیری کارآیی پرس و جوها با استفاده از SET STATISTICS TIME**
نویسنده: امید شکری
تاریخ: ۱۳۹۲/۱۲/۱۰ ۲۱:۵۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SQL, SQL Server Performance Tuning

یکی از وظایف اصلی مدیر و یا توسعه دهنده یک بانک اطلاعاتی، نوشتن کدهای T-SQL و اندازه‌گیری عملکرد آنها می‌باشد. ابزارهای مختلفی برای انجام این کار وجود دارد، چه آنهایی که در خود SQL Server بصورت محلی وجود دارند و چه آنهایی که توسط شرکت‌های ثالث ارائه می‌شوند. اما مسئله مهمی که باید در نظر بگیرید چگونگی نوشتن یک پرس و جو (Query) و اندازه گیری کارآیی آن می‌باشد و اینکه باید روی چه مواردی متمرکز شد. در اکثر مواقع گرفتن زمان اجرای یک پرس و جو تا اندازه‌ای خوب می‌باشد. یکی از مواردی که باید روی آن متمرکز شد منابع استفاده شده توسط سرور می‌باشد، درحالیکه زمان اجرای پرس وجو به پارامترهای دیگری همچون بار سرور نیز بستگی دارد. علاوه بر استفاده از پروفایلر و نقشه اجرای کوئری (Execution Plan) ، می‌توانید از **SET STATISTICS TIME** نیز استفاده نمایید.

SET STATISTICS TIME تنظیمی است که برای اندازه گیری منابع مورد نیاز اجرای یک پرس و جو به شما کمک می‌کند. SET STATISTICS TIME آمار مربوط به زمان تجزیه و تحلیل (Parse)، کامپایل و اجرای هر دستور در یک پرس و جو را نمایش می‌دهد. راه‌های مختلف اندکی برای مقایسه آماری دو پرس و جو و انتخاب بهترین آنها برای استفاده وجود دارند.

برای روشن کردن این تنظیم دو راه وجود دارد. ابتدا اینکه از دستور Set برای روشن و خاموش کردن استفاده نمایید و یا اینکه از طریق Query Analyzer اقدام به انجام این کار نمایید.

```
SET STATISTICS TIME ON
```

برای اینکه بتوانید آمارهای این تنظیمات را مشاهده کنید می‌بایست قبل از اجرای پرس و جو تنظیم مورد نظر را روشن نمایید. در نظر داشته باشید که با روشن کردن این تنظیم، برای تمامی پرس و جوهای مربوط به آن جلسه (Session) روشن خواهد ماند تا زمانیکه آنرا خاموش نمایید.

```
SELECT ProductID, StartDate, EndDate, StandardCost
FROM Production.ProductCostHistory
WHERE StandardCost < 500.00;
```

با اجرای دستورات بالا خروجی آن بصورت زیر می‌باشد:

```
SQL Server parse and compile time:
   CPU time = 0 ms, elapsed time = 1 ms.

(269 row(s) affected)

SQL Server Execution Times:
   CPU time = 1 ms,  elapsed time = 2 ms.
```

زمان صرف شده برای اجرای یک کوئری به دو بخش تقسیم می‌شود:

زمان کامپایل و تجزیه و تحلیل (parse and compile time) : زمانیکه یک کوئری را برای اجرا به SQL Server ارائه می‌دهید، SQL Server آنرا از نظر خطای نحوی بررسی می‌نماید و بهینه ساز یک نقشه بهینه را برای اجرا تولید می‌نماید. اگر به خروجی نگاه کنید، زمان پردازش (CPU time) و زمان سپری شده (elapsed time) را نشان می دهد. منظور از زمان پردازش زمان واقعی صرف شده روی پردازنده می‌باشد و زمان سپری شده اشاره به زمان تکمیل شدن عملیات کامپایل و تجزیه و تحلیل می‌باشد. تفاوت بین زمان پردازش و زمان سپری شده ممکن است زمان انتظار در صف برای گرفتن پردازنده و یا انتظار برای تکمیل عملیات IO باشد.

زمان اجرا (Execution Times) **:** این زمان اشاره به زمان سپری شده برای تکمیل اجرای نقشه کامپایل شده دارد. زمان پردازش اشاره به زمان واقعی صرف شده روی پردازنده دارد و زمان سپری شده نیز مجموع زمان صرف شده تا تکمیل اجرای دستور که شامل زمان انتظار برای تکمیل عملیات IO و زمان صرف شده برای انتقال خروجی به کلاینت می‌باشد، دارد. زمان پردازش می‌تواند به عنوان مبنایی برای اندازه‌گیری کارآیی مورد استفاده قرار بگیرد. این مقدار در اجراهای مختلف تفاوت چندانی با هم ندارند جز اینکه کوئری و یا داده‌ها تغییر نمایند. توجه نمایید که زمان براساس میلی ثانیه می‌باشد. زمان سپری شده نیز به فاکتورهایی مانند

بارگذاری روی سرور، بارگذاری IO، و پهنای باند بین سرور و کلاینت وابسته می‌باشد. بنابراین **همیشه زمان پردازش به عنوان مبنایی برای اندازه‌گیری کارایی استفاده می‌شود .**

در این بخش به بررسی **SET STATISTICS TIME** در SQL Server پرداختیم. در بخش بعدی به بررسی **SET STATISTICS IO** برای اندازه گیری کارایی پرس و جوها از نظر میزان استفاده IO خواهیم پرداخت.

عنوان: **ساخت کاربر ویندوز توسط SQL Server**
نویسنده: یوسف رضایی
تاریخ: ۱۳۹۲/۱۲/۲۷ ۹:۴۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server

چندی پیش لازم بود که از طریق ویندوز، یک یوزر ادمین برای sql بسازم که خدا را شکر میسر شد. اما خوب است بدانید با SQL Server نیز می‌توانید یک کاربر جدید ویندوز بسازید. این مورد را در ادامه‌ی بحث بررسی خواهیم کرد.

البته باید دقت داشت که در حقیقت این امکان را توسط خط فرمان فعال شده‌ی ویندوز، به نام xp_cmdshell در اختیار خواهیم داشت. توسط این رویه ذخیره شده، هر کاری را که در ویندوز توسط خط فرمان استاندارد آن می‌توانید انجام دهید، اینجا نیز قادر به انجام آن خواهید بود. برای مثال یک کاربر را ادمین کنید و امثال آن.
برای اینکار ابتدا باید این امکان را جهت کاربر sa، یا هر کاربر دیگری که در SQL Server نقش sysadmin دارد، فعال کنید. برای این منظور کوئری‌های ذیل را اجرا کنید:

```
-- To allow advanced options to be changed.
EXEC sp_configure 'show advanced options', 1
GO
-- To update the currently configured value for advanced options.
RECONFIGURE
GO
-- To enable the feature.
EXEC sp_configure 'xp_cmdshell', 1
GO
-- To update the currently configured value for this feature.
RECONFIGURE
GO
```

بعد برای اینکه کاربری را اضافه کنید، کد زیر را کپی و اجرا کنید:

```
xp_cmdshell 'net user nadeema /Add'
```

و در آخر جهت تغییر گروه این کاربر و افزودن آن به مجموعه‌ی ادمین‌های محلی، می‌توانید کد زیر را نوشته و اجرا کنید:

```
xp_cmdshell  'net localgroup Administrator nadeema /add'
```

عنوان: Resource Governor در SQL Server 2008
نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۲/۰۲ ۱۷:۵۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Database, SQL Server 2008

مقدمه

Resource Governor، اجازه می‌دهد تا انواع مختلف Session را بر روی Server طبقه بندی کنید که به نوبه خود چگونگی کنترل تخصیص منابع سرور به فعالیت داده شده را به شما اعطا می‌کند. این قابلیت کمک می‌کند که ادامه فرآیندهای OLTP تضمین شود و یک عملکرد قابل پیش بینی فراهم می‌کند تا توسط فرآیندهای غیر قابل پیش بینی، تحت تاثیر منفی قرار نگیرد. با استفاده از Resource Governor، قادر خواهید بود نحوه دستیابی به Session را به منظور محدود کردن منابع خاص برای SQL Server مشخص کنید. به عنوان مثال می‌توانید مشخص کنید که بیش از 20 درصد از پردازنده یا منابع حافظه به گزارش‌های در حال اجرا اختصاص داده نشود. هنگامیکه این ویژگی فعال باشد، مهم نیست چه تعداد گزارش در حال اجرا است، آنها هرگز نمی‌توانند از تخصیص منابع تعیین شده تجاوز کنند. البته این موضوع عملکرد گزارش گیری را کاهش می‌دهد ولی عملکرد فرآیندهای OLTP حداقل توسط گزارش ها، دیگر تحت تاثیر منفی قرار نمی‌گیرد.

1- بررسی اجمالی Resource Governor:

Resource Governor، با کنترل تخصیص منابع بر حسب Workload کار می‌کند. هنگامی که یک درخواست اتصال به موتور بانک اطلاعاتی ارسال می‌شود درخواست براساس یک تابع رده بندی (Classification function) طبقه بندی می‌شود. تابع رده بندی یک تابع اسکالر است که از طریق T-SQL تعریف می‌شود. تابع رده بندی، اطلاعات را درباره یک اتصال (برای مثال، login ID، application name، hostname، server role) ارزیابی می‌کند، به منظور تشخیص اینکه چگونه آنها را دسته بندی کند. پس از دسته بندی درخواست اتصال، آنها به گروه‌های حجم کاری (Workload Group) که برای رده بندی تعریف شده اند، شکسته می‌شوند. هر Workload Group مرتبط با یک مخزن منابع (Resource Pool) است.

یک Resource Pool، منابع فیزیکی SQL Server را نمایش می‌دهد (در حال حاضر در SQL Server 2008، تنها منابع فیزیکی موجود برای پیکربندی پردازنده و حافظه است) و مقدار حداکثر پردازنده و یا منابع حافظه را که به نوع خاصی از Workload اختصاص داده می‌شود، تعیین می‌کند. هنگامی که یک اتصال طبقه بندی شده و در Workload Group صحیح خود قرار می‌گیرد به این اتصال، پردازنده و منابع حافظه به اندازه نسبت داده شده به آن تخصیص داده می‌شود و سپس Query به Query Optimizer برای اجرا داده می‌شود.

2- اجزای Resource Governor:

Resource Governor، از سه قسمت اصلی تشکیل شده است: Classification، Workload Groups و Resource Pools. درک این سه قسمت و چگونگی تعامل آنها به درک و استفاده از Resource Governor کمک می‌کند.

2-1- Classification:

Classification، فرآیند ارزیابی اتصالات ورودی کاربر و اختصاص آن به یک Workload Group است که توسط منطق موجود در یک تابع تعریف شده توسط کاربر (user-defined function) انجام می‌شود. تابع نام یک Workload Group را برمی گرداند که Resource Governor از آن برای مسیر دهی Session به Workload Group مناسب استفاده می‌کند.

هنگامی که Resource Governor پیکربندی می‌شود فرآیند ورود به سیستم برای یک Session شامل گام‌های زیر است:

- Login authentication
- LOGON trigger execution
- Classification

2-2- Workload Groups:

Workload Groups، ظروفی برای اتصالات مشابه هستند که با توجه به معیارهای طبقه بندی برای هر اتصال گروه بندی می‌شوند. Workload Groups همچنین مکانیسمی برای تجمیع نظارت بر روی منابع مصرفی فراهم می‌کند.

Resource Governor دو Workload Group از پیش تعریف شده دارد: یک گروه داخلی (internal group) و یک گروه پیش فرض (default group).

Internal Workload Group ، تنها توسط فرآیندهای داخلی موتور بانک اطلاعاتی استفاده می‌شود. معیارهای طبقه بندی را برای گروه‌های داخلی نمی‌توانید تغییر دهید و همچنین هیچ یک از درخواست‌های کاربران را برای انتقال به گروه داخلی نمی‌توانید رده بندی کنید، با این حال بر گروه داخلی می‌توانید نظارت کنید.

درخواست‌های اتصال به طور خودکار هنگامی که شرایط زیر وجود دارد، به *Default Workload Group* رده بندی می‌شوند:

- معیاری برای طبقه بندی درخواست وجود ندارد.
- کوششی برای رده بندی درخواستی به گروهی که وجود ندارد.
- خرابی کلی Classification

Resource Governor، در مجموع 20 عدد Workload Group را پشتیبانی می‌کند. از آنجائی که دو عدد از آنها برای Workload Groupهای داخلی و پیش فرض ذخیره شده اند در مجموع 18 عدد Workload Group تعریف شده توسط کاربر (user-defined) می‌توان تعریف نمود.

2-3- Resource pools:

Resource Pool (مخزن منابع)، نشان دهنده تخصیص منابع فیزیکی به SQL Server است. یک Resource Pool از دو بخش تشکیل شده است:

- در بخش نخستین حداقل رزرو منابع را مشخص می‌کنیم، این بخش از مخزن منابع با مخازن دیگر همپوشانی نمی‌کند.
- در بخش دیگر حداکثر ممکن رزرو منابع را برای مخزن مشخص می‌کنیم، تخصیص منابع با مخازن دیگر مشترک است.

در SQL Server 2008 مخزن منابع با تعیین حداقل و حداکثر تخصیص CPU و حداقل و حداکثر تخصیص حافظه تنظیم می‌گردد. با تنظیم حداقل، در دسترس بودن منبع از مخزن تضمین می‌شود. از آنجائی که در هر رزرو حداقل منابع تداخلی نمی‌تواند وجود داشته باشد، مجموع مقادیر حداقل در تمام مخازن از 100% کل منابع Server نمی‌تواند تجاوز کند.

مقدار حداکثر در محدوده بین حداقل و شامل 100% مقدار می‌تواند تنظیم گردد. تنظیم حداکثر نشان دهنده مقدار حداکثری است که یک Session می‌تواند مصرف کند، مادامی که منابع در دسترس باشند و توسط مخزن دیگر که با حداقل مقدار غیر صفر پیکربندی شده، استفاده نشود. هنگامی که یک مخزن با حداقل مقدار غیر صفر تعریف شده، مقدار حداکثر موثر از مخزن‌های دیگر دوباره تنظیم می‌شوند، در صورت لزوم حداکثر مقدار موجود از جمع کل حداقل منابع مخازن دیگر کسر می‌گردد.

برای مثال، دو مخزن تعریف شده توسط کاربر (user-defined) را در نظر بگیرید. مخزن اول Pool1 با مقدار حداقل 20% و مقدار حداکثر 100% تعریف شده، مخزن دیگری Pool2 با مقدار حداقل 50% و مقدار حداکثر 70% تعریف شده است. حداکثر مقدار موثر برایPool1 برابر 50% است (100% منهای مقدار حداقل 50% مخزن Pool2) و حداکثر مقدار موثر برای 70% ،Pool2 است زیرا حداکثر مقداری است که پیکربندی شده است، گر چه 80% باقی می‌ماند.

بخش مشترکی از مخزن (مقدارش بین مقدار حداقل و مقدار حداکثر موثر است) که برای تعیین مقدار منابع مورد استفاده است، توسط مخزن می‌تواند مصرف شود اگر منابعی موجود باشد و توسط مخازن دیگر مصرف نشده باشد. هنگامی که منابعی توسط یک مخزن مصرف می‌شوند، آنها به یک مخزن مشخص نسبت داده می‌شوند، به بیان دیگر اشتراکی نیستند تا زمانی که فرآیند در آن مخزن به اتمام برسد.

برای توضیح بیشتر یک سناریو که در آن سه مخزن تعریف شده توسط کاربر (user-defined) وجود دارد، را در نظر بگیرید:

PoolA با حداقل مقدار 10% و حداکثر مقدار 100% تعریف می‌شود.

PoolB با حداقل مقدار 35% و حداکثر مقدار 90% تعریف می‌شود.

PoolC با حداقل مقدار 30% و حداکثر مقدار 80% تعریف می‌شود.

مقدار موثر PoolA و مجموع در صد منابع به اشتراک گذاشته PoolA به شرح زیر محاسبه خواهد شد:

(حداکثر مقدار PoolA) - (حداقل مقدار PoolB) - (حداقل مقدار PoolC) = (حداکثر مقدار موثر PoolA)

(حداکثر مقدار موثر PoolA) – (حداقل مقدار PoolA) = (اشتراک PoolA)

جدول زیر مقدار حداکثر موثر و اشتراکی را برای هر مخزن در این پیکربندی نمایش می‌دهد:

Resource Pool	MIN %	MAX %	Effective MAX %	Shared %
Internal	0	100	100	100
Default	0	100	25	25
PoolA	10	100	35	25
PoolB	35	90	50	15
PoolC	30	80	35	5

Internal Pool، منابع مصرف شده توسط فرآیندهای داخلی موتور بانک اطلاعاتی را نشان می‌دهد. این مخزن تنها شامل گروه‌های داخلی است و به هیچ وجه قابل تغییر نیست. مخزن داخلی مقدار ثابت حداقل صفر و حداکثر 100% را دارد و مصرف منابع توسط مخزن داخلی، از طریق تنظیمات در هر مخزن دیگر محدود یا کاسته نمی‌شود.

به عبارت دیگر حداکثر مقدار موثر مخزن داخلی همیشه 100% است. هر workloads در مخزن داخلی برای عملکرد Server حیاتی در نظر گرفته می‌شود و Resource Governor در صورت لزوم اجازه می‌دهد تا مخازن داخلی 100% منابع موجود را مصرف کند حتی اگر به معنی نقض نیازمندیهای منابع از سایر مخازن باشد.

Default Pool، اولین مخزن تعریف شده کاربر است. قبل از هرگونه پیکربندی، Default Pool تنها حاوی Default group است. Default Pool نمی‌تواند ایجاد یا حذف شود اما می‌تواند تغییر کند. Default Pool علاوه بر Default group می‌تواند شامل گروه‌های تعریف شده توسط کاربر (user-defined) نیز باشد.

3- پیکر بندی Resource Governor :

پیکربندی Resource Governor شامل مراحل زیر است:

- فعال کردن Resource Governor
- ایجاد مخازن منابع (Resource Pools) تعریف شده توسط کاربر (user-defined)
- تعریف Workload Groups و نسبت دادن آن به مخازن
- ایجاد Classification function
- ثبت Classification function به Resource Governor

3-1- فعال کردن Resource Governor

پیش از اینکه بتوانید یک Resource Pool را ایجاد کنید، نیاز است تا نخست Resource Governor را فعال کنید.

3-2- تعریف Resource Pool

ویژگی‌های موجود برای یک Resource Pool عبارتند از:

%Name، Minimum CPU %، Maximum CPU%، Min Memory%، Max Memory

3-3- تعریف Workload Group

پس از اینکه Resource Pool را تعریف کردید، گام بعدی ایجاد یک Workload Group و اختصاص آن به Resource Pool مناسب است. چندین workgroup را می‌توان به مخزن (Pool) یکسان نسبت داد اما یک workgroup را به چندین Resource Pool نمی‌توان نسبت داد. خواص انتخابی موجود برای Workload Groups به شما اجازه می‌دهد سطح بهتری از کنترل را روی اجرای دستورات یک Workload Group تنظیم کنید. انتخاب‌های موجود عبارتند از:

3-3-1- Importance :

اهمیت نسبی (کم، متوسط یا بالا) Workload Group درون Resource Pool را تعیین می‌کند. اگر چندین Workload Group را در یک Resource Pool تعریف کنید این تنظیمات تعیین می‌کند که درخواست‌ها در عرض یک Workload Group در اولویت بالاتر یا پائین‌تری از Workload Groupهای دیگر درون همان Resource Pool اجرا شوند، مقدار متوسط تنظیم پیش فرض است. در حال حاضر فاکتورهای وزنی برای هر تنظیم کم برابر 1، متوسط برابر3 و زیاد برابر 9 است. به این معنی که زمانبند به اجرای Sessionهای درون workgroup هائی با اهمیت بالا، سه برابر بیشتر از workgroupهای با اهمیت متوسط و نه برابر بیشتر از

workgroupهای کم اهمیت، مبادرت خواهد کرد.

3-3-2- Maximum Request :

حداکثر تعداد درخواست‌های همزمان که اجازه دارند در یک Workload Group اجرا شوند را مشخص می‌کند. تنظیم پیش فرض، صفر، تعداد نامحدود دستور را اجازه می‌دهد.

3-3-3- CPU Time :

حداکثر مقدار زمان پردازنده در ثانیه را مشخص می‌کند که یک درخواست درون Workload Group می‌تواند استفاده کند. تنظیم پیش فرض، صفر، به معنی نامحدود است.

3-3-4- Memory Grant %:

به صورت در صد، حداکثر مقدار اعطا حافظه برای اجرا (Execution grant memory)، که یک تک دستور از Resource Pool می‌تواند اخذ کند را مشخص می‌کند. این درصد نسبی است از مقدار حافظه ای که به Resource Pool نسبت داده می‌شود. محدوده مجاز مقادیر از 0 تا 100 است. تنظیم پیش فرض 25 است.

Execution grant memory، مقدار حافظه ای است که برای اجرای query استفاده می‌شود (نه برای Buffer کردن یا cache کردن) که می‌تواند صرفه نظر از Resource Pool یا Workload Group توسط تعدادی از Sessionها به اشتراک گذاشته شود. توجه شود که تنظیم این مقدار به صفر از اجرای عملیات Hash Join و دستورات مرتب سازی در Workload Groupهای تعریف شده توسط کاربر (user-defined)جلوگیری می‌کند. همچنین این مقدار توصیه نمی‌شود بیشتر از 70 باشد زیرا ممکن است Server قادر نباشد، اگر Queryهای همزمان در حال اجرا باشند، حافظه آزاد کافی اختصاص دهد.

3-3-5- Grant Time-out :

حداکثر زمان، به ثانیه، که یک query برای یک منبع منتظر می‌ماند تا در دسترس شود را مشخص می‌کند. اگر منبع در دسترس نباشد، فرآیند ممکن است با یک خطای time-out مواجه شود. تنظیم پیش فرض، صفر، به معنی این است که سرور time-out را با استفاده از محاسبات داخلی بر مبنای هزینه پرس و جو (query cost) با تعیین حداکثر زمان برآورد می‌کند.

3-3-6- Degree of Parallelism :

حداکثر درجه موازی سازی (DOP) را برای پرس و جوهای موازی تعیین می‌کند. محدوده مجاز مقادیر از 0 تا 64 است. تنظیم پیش فرض، صفر، به معنی این است که فرآیندها از تنظیمات عمومی استفاده می‌کنند.

3-4- ایجاد یک Classification function

پس از تعریف Resource Pool و Workload Group، به یک Classification function نیاز است که شامل منطق ارزیابی اتصالات و نسبت دادن آنها به Workload Group مناسب است. Classification function برای هر اتصال Session جدید به SQL Server بکار می‌رود. هر Session در Workload Group نسبت داده شده به آن باقی می‌ماند تا زمانی که به پایان برسد، مگر اینکه صراحتاً به یک گروه متفاوت دوباره نسبت داده شود. فقط یک Classification function فعال در هر زمان می‌تواند وجود داشته باشد. در صورت عدم تعریف شدن یا عدم فعال بودن Classification function همه اتصالات به Workload Group Default نسبت داده می‌شوند. Classification function یک نام workgroup که نوع آن SYSNAME است (که یک نام مستعار برای دیتا تایپ nvarchar 128 است.) برمی گرداند. اگر تابع تعریف شده مقدار 'Default'، NULL یا نام گروهی که وجود ندارد را برگرداند، Session به Workload Group Default نسبت داده می‌شود. همچنین اگر به هر دلیلی تابع با موفقیت خاتمه نیابد Session به Workload Group Default نسبت داده می‌شود.

منطق Classification function معمولاً مبتنی بر ویژگی‌های اتصال است و اغلب از طریق مقدار بازگشتی توابع سیستمی از قبیل:

HOST_NAME()، IS_SERVERROLEMEMBER()، IS_MEMBER()، SUSER_SNAME()، SUSER_NAME() و یا APP_NAME()، نام Workload Group اتصال مشخص می‌شود. علاوه بر این توابع می‌توانید از ویژگی‌های توابع دیگر برای ساخت منطق رده بندی استفاده کنید. تابع LOGINPROPERTY() شامل دو ویژگی (DefaultDatabase و DefaultLanguage) می‌باشد که می‌تواند برای Classification function استفاده شود. بعلاوه تابع CONNECTIONPROPERTY() پروتکل‌ها و دسترسی به نقل و انتقالات در شبکه، همچنین جزئیات طرح احراز هویت، Local IP address و TCP Port و Client’s IP Address را برای استفاده اتصالات فراهم می‌کند. برای مثال می‌توانید برای یک اتصال، یک Workload Group نسبت دهید، مبتنی بر اینکه subnet یک اتصال ازکجا می‌آید.

نکته: اگر قصد دارید از هر یک از توابع HOST_NAME() و یا APP_NAME() در تابع رده بندی تان استفاده کنید، توجه داشته باشید این امکان وجود دارد مقادیر بازگردانده شده توسط این توابع توسط کاربران تغییر داده شوند، گر چه به طور کلی گرایش به استفاده از تابع APP_NAME() برای رده بندی اتصالات بیشتر است.

4- بررسی نمونه ای از پیکربندی Resource Governor

برای سادگی، در این قسمت مثالی ارائه می‌شود که از تابع SUSER_NAME() استفاده می‌کند: در گام نخست، دو Resource Pool ایجاد می‌شود (ReportPool و OLTPPool)

```
CREATE RESOURCE POOL [ReportPool] WITH(
min_cpu_percent=0,
        max_cpu_percent=20,
        min_memory_percent=0,
        max_memory_percent=30)
GO
CREATE RESOURCE POOL [OLTPPool] WITH(
min_cpu_percent=80,
        max_cpu_percent=100,
        min_memory_percent=75,
        max_memory_percent=100)
GO
```

در گام بعدی، دو Workload Group ایجاد می‌شود (ReportWG1 و OLTPWG1)

```
CREATE WORKLOAD GROUP [ReportWG1] WITH(
group_max_requests=0,
        importance=Medium,
        request_max_cpu_time_sec=0,
        request_max_memory_grant_percent=25,
        request_memory_grant_timeout_sec=0,
        max_dop=0) USING [ReportPool]
GO
CREATE WORKLOAD GROUP [OLTPWG1] WITH(
group_max_requests=0,
        importance=High,
        request_max_cpu_time_sec=0,
        request_max_memory_grant_percent=25,
        request_memory_grant_timeout_sec=0,
        max_dop=0) USING [OLTPPool]
GO
```

سپس دو Login ایجاد می‌شود (report_user و oltp_user) که در تابع رده بندی استفاده خواهند شد برای مشخص کردن این که اتصالات Seesion به کدام Workload Group نسبت داده شوند. پس از اضافه کردن Loginها به عنوان Userها به Database مورد نظر مان، در بانک اطلاعاتی Master تابع رده بندی (Classification function) را ایجاد می‌کنیم:

```
use master
go
CREATE FUNCTION dbo.WorkgroupClassifier ()
RETURNS SYSNAME WITH SCHEMABINDING
AS
BEGIN
DECLARE @WorkloadGroup SYSNAME = N'Unidentified';
SET @WorkloadGroup = CASE suser_name()
        WHEN N'report_user' THEN N'ReportWG1'
        WHEN N'oltp_user' THEN N'OLTPWG1'
        ELSE N'Unidentified'
        END;
RETURN @WorkloadGroup;
END;
Go
GRANT EXECUTE on dbo.WorkgroupClassifier to public
Go
```

می توان تابع WorkgroupClassifier() را در محیط SSMS با اجرای دستور زیر برای Loginهای متفاوت تست نمود:

```
select dbo.WorkgroupClassifier()
```

در ادامه دستور زیر برای پیکربندی تابع رده بندی به Resource Governor استفاده می‌شود:

```
ALTER RESOURCE GOVERNOR
WITH (CLASSIFIER_FUNCTION = dbo.WorkgroupClassifier);
ALTER RESOURCE GOVERNOR RECONFIGURE;
```

5- اصلاح پیکربندی Resource Governor:

می‌توانید درمحیط SSMS تنظیمات Resource Pool و Workload Group را تغییر دهید (برای مثال حداکثر استفاده CPU برای یک Resource Pool و یا درجه اهمیت یک Workload Group). متناوباً می‌توان از دستورات T-SQL استفاده نمود.

نکته: پس از اجرای دستورات ALTER RESOURCE POOL یا ALTER WORKLOAD GROUP، برای اعمال کردن تغییرات اجرای دستور ALTER RESOURCE GOVERNOR RECONFIGURE نیاز می‌باشد.

5-1- حذف Workload Group :

یک Workload Group را اگر هر نوع Session فعال نسبت داده شده به آن وجود داشته باشد، نمی‌توان حذف نمود. اگر یک Workload Group شامل Sessionهای فعال باشد، حذف Workload Groupو یا جابجائی آن به یک Resource Pool متفاوت، هنگامی که دستور ALTER RESOURCE GOVERNOR RECONFIGURE برای اعمال نمودن تغییرات فراخوانی می‌شود، با خطا مواجه خواهد شد.

5-2- حذف Resource Pools:

یک Resource Pool را اگر هر نوع Workload Group نسبت داده شده به آن وجود داشته باشد، نمی‌توان حذف نمود. نخست نیاز

دارید Workload Group حذف شود و یا به Resource Pool دیگری جابجا گردد.

5-3- اصلاح Classification function:

اگر نیاز دارید تغییراتی در تابع رده بندی ایجاد نمائید، مهم است توجه داشته باشید که تابع رده بندی تا زمانی که مشخص شده (marked) برای Resource Governor است، نمی‌توان آنرا حذف و یا تغییر داد. پیش از اینکه بتوان تابع رده بندی را اصلاح و یا حذف نمود نخست نیاز دارید Resource Governor را غیر فعال نمائید. متناوباً می‌توان تابع رده بندی را جایگزین کرد با اجرای دستور ALTER RESOURCE GOVERNOR و فرستادن (passing) یک اسم متفاوت برای CLASSIFIER_FUNCTION،همچنین می‌توان با اجرای دستور زیر تابع رده بندی جاری را غیر فعال نمود:

```
ALTER RESOURCE GOVERNOR
WITH (CLASSIFIER_FUNCTION = NULL);
ALTER RESOURCE GOVERNOR RECONFIGURE;
```

تابع رده بندی می‌توان تعریف کرد که نام Workload Group را از جداول یک بانک اطلاعاتی جستجو کند به جای اینکه نام Workload Group به صورت hard-coding و مطابق با ضوابط درون تابع باشد. عملکرد، در موقع دسترسی به جدول برای جستجو کردن نام Workload Group، نباید تا حد زیادی تحت تاثیر قرار گیرد.

6- نظارت بر Resource Governor

با استفاده از Performance Monitor، events و Dynamic Management View (DMV) می‌توان Workload Group و Resource Pool را نظارت (Monitor) کرد. دو شی Performance برای این کار موجود است: SQL Server:Workload Group Stats و SQL Server:Resource Pool Stats

شکل زیر مربوط به پیکر بندی مثال مورد نظرمان می‌باشد:

7- نتیجه گیری

Resource Governor چندین مزیت بالقوه ارائه می‌دهد، در درجه اول قابلیت اولویت بندی منابع Server برای کاربران و برنامه‌های کاربردی (applications) بحرانی، جلوگیری از "runaway" یا درخواست‌های غیر منتظره ای که به شدت و بطور قابل توجهی روی کارائی Server تاثیر منفی می‌گذارند.

ضمناً Resource Governor چندین مشکل بالقوه نیز عرضه می‌کند، برای مثال پیکربندی اشتباه Resource Governor تنها به عملکرد کلی Server آسیب نمی‌رساند بلکه به طور بالقوه روی سرور قفل (Lock) می‌تواند ایجاد کند و نیاز به استفاده از اتصال اختصاصی Administrator برای متصل شدن به SQL Server به منظور اشکال یابی و رفع مشکل می‌باشد. بنابراین توصیه شده است که تنها در صورتی که DBA با تجربه ای هستید و درک خوب و آشنائی خوبی با Workload هائی که روی بانک اطلاعاتی اجرا می‌شوند دارید، Resource Governor را بکار برید. حتی در این صورت، ضروری است که پیکربندی تان را روی یک Server تستی پیش از اینکه روی محیط تولیدی بگسترانید، تست نمائید.

Resource Governor به عنوان یک ویژگی با نام تجاری جدید در SQL Server 2008، با تعدادی محدودیت همراه است که احتمالاً در نسخه‌های بعدی SQL Server حذف خواهد شد، از محدودیت های بارز :

- محدودیت منابع (Resource)، که به CPU و حافظه محدود می‌شوند. I/O Disk و منابع شبکه را در SQL Server 2008 نمی‌توان محدود کرد.
- استفاده از منابع برای Analysis Service، Reporting Service و Integeration Service را نمی‌توان محدود کرد . در این نسخه محدودیت‌های منابع تنها روی هسته موتور بانک اطلاعاتی بکار برده می‌شود.
- محدودیت‌های Resource Governor روی یک SQL Instance تعریف و بکار برده می‌شود.

عنوان: **بررسی Transactions و Locks در SQL Server**
نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۲/۰۳ ۹:۳۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, T-SQL, Database, development

مقدمه SQL Server، با هر تقاضا به عنوان یک واحد مستقل رفتار می‌کند. در وضعیت‌های پیچیده ای که فعالیت‌ها توسط مجموعه ای از دستورات SQL انجام می‌شود، به طوری که یا همه باید اجرا شوند یا هیچکدام اجرا نشوند، این روش مناسب نیست. در چنین وضعیت هایی، نه تنها تقاضاهای موجود در یک دنباله به یکدیگر بستگی دارند، بلکه شکست یکی از تقاضاهای موجود در دنباله، به معنای این است که کل تقاضاهای موجود در دنباله باید لغو شوند، و تغییرات حاصل از تقاضاهای اجراشده در آن دنباله خنثی شوند تا بانک اطلاعاتی به حالت قبلی برگردد.

1- تراکنش چیست؟ تراکنش شامل مجموعه ای از یک یا چند دستور SQL است که به عنوان یک واحد عمل می‌کنند. اگر یک دستور SQL در این واحد با موفقیت اجرا نشود، کل آن واحد خنثی می‌شود و داده هایی که در اجرای آن واحد تغییر کرده اند، به حالت اول برگردانده می‌شود. بنابراین تراکنش وقتی موفق است که هر یک از دستورات آن با موفقیت اجرا شوند. برای درک مفهوم تراکنش مثال زیر را در نظر بگیرید: سهامدار A در معامله ای 400 سهم از شرکتی را به سهامدار B می‌فروشد. در این سیستم، معامله وقتی کامل می‌شود که حساب سهامدار A به اندازه 400 بدهکار و حساب سهامدار B همزمان به اندازه 400 بستانکار شود. اگر هر کدام از این مراحل با شکست مواجه شود، معامله انجام نمی‌شود.

2- خواص تراکنش هر تراکنش دارای چهار خاصیت است (معروف به ACID) که به شرح زیر می‌باشند:

2-1- خاصیت یکپارچگی (Atomicity) یکپارچگی به معنای این است که تراکنش باید به عنوان یک واحد منسجم (غیر قابل تفکیک) در نظر گرفته شود. در مثال مربوط به مبادله سهام، یکپارچگی به معنای این است که فروش سهام توسط سهامدار A و خرید آن سهام توسط سهامدار B، مستقل از هم قابل انجام نیستند و برای این که تراکنش کامل شود، هر دو عمل باید با موفقیت انجام شوند.
اجرای یکپارچه، یک عمل "همه یا هیچ" است. در عملیات یکپارچه، اگر هر کدام از دستورات موجود در تراکنش با شکست مواجه شوند، اجرای تمام دستورات قبلی خنثی می‌شود تا به جامعیت بانک اطلاعاتی آسیب نرسد.

2-2- خاصیت سازگاری (Consistency) سازگاری زمانی وجود دارد که هر تراکنش، سیستم را در یک حالت سازگار قرار دهد (چه تراکنش به طور کامل انجام شود و چه در اثر وجود خطایی خنثی گردد). در مثال مبادله سهام، سازگاری به معنای آن است که هر بدهکاری مربوط به حساب فروشنده، موجب همان میزان بستانکاری در حساب خریدار می‌شود.
در SQL Server، سازگاری با راهکار ثبت فایل سابقه انجام می‌گیرد که تمام تغییرات را در بانک اطلاعاتی ذخیره می‌کند و جزییات را برای ترمیم تراکنش ثبت می‌نماید. اگر سیستم در اثنای اجرای تراکنش خراب شود، فرآیند ترمیم SQL Server با استفاده از این اطلاعات، تعیین می‌کند که آیا تراکنش با موفقیت انجام شده است یا خیر، و در صورت عدم موفقیت آن را خنثی می‌کند. خاصیت سازگاری تضمین می‌کند که بانک اطلاعاتی هیچگاه تراکنش‌های ناقص را نشان نمی‌دهد.

2-3- خاصیت تفکیک (Isolation) تفکیک موجب می‌شود هر تراکنش در فضای خودش و جدا از سایر تراکنش‌های دیگری که در سیستم انجام می‌گیرد، اجرا شود و نتایج هر تراکنش فقط در صورت کامل شدن آن قابل مشاهده است. اگر چندین تراکنش همزمان در سیستم در حال اجرا باشند، اصل تفکیک تضمین می‌کند که اثرات یک تراکنش تا کامل شدن آن، قابل مشاهده نیست. در مثال مربوط به مبادله سهام، اصل تفکیک به معنای این است که تراکنش بین دو سهامدار، مستقل از تمام تراکنش‌های دیگری است که در سیستم به مبادله سهام می‌پردازند و اثر آن وقتی برای افراد قابل مشاهده است که آن تراکنش کامل شده باشد. این اصل در مواردی که سیستم همزمان از چندین کاربر پشتیبانی می‌کند، مفید است.

2-4- پایداری (Durability) پایداری به معنای این است که تغییرات حاصل از نهایی شدن تراکنش، حتی در صورت خرابی سیستم نیز پایدار می‌ماند. اغلب سیستم‌های مدیریت بانک اطلاعاتی رابطه ای، از طریق ثبت تمام فعالیت‌های تغییر دهنده‌ی داده‌ها در بانک اطلاعاتی، پایداری را تضمین می‌کنند. در صورت خرابی سیستم یا رسانه ذخیره سازی داده ها، سیستم قادر است آخرین

بهنگام سازی موفق را هنگام راه اندازی مجدد، بازیابی کند. در مثال مربوط به مبادله سهام، پایداری به معنای این است که وقتی انتقال سهام از سهامدار A به B با موفقیت انجام گردید، حتی اگر سیستم بعداً خراب شد، باید نتیجه‌ی آن را منعکس سازد.

3- مشکلات همزمانی(Concurrency Effects):

3-1- Dirty Read: زمانی روی می‌دهد که تراکنشی رکوردی را می‌خواند، که بخشی از تراکنشی است که هنوز تکمیل نشده است، اگر آن تراکنش Rollback شود اطلاعاتی از بانک اطلاعاتی دارید که هرگز روی نداده است.
اگر سطح جداسازی تراکنش (پیش فرض) Read Committed باشد، این مشکل بوجود نمی‌آید.

3-2- Non-Repeatable Read: زمانی ایجاد می‌شود که رکوردی را دو بار در یک تراکنش می‌خوانید و در این اثنا یک تراکنش مجزای دیگر داده‌ها را تغییر می‌دهد. برای پیشگیری از این مسئله باید سطح جداسازی تراکنش برابر با Repeatable Read یا Serializable باشد.

3-3- Phantoms: با رکوردهای مرموزی سروکار داریم که گویی تحت تاثیر عبارات Update و Delete صادر شده قرار نگرفته اند. به طور خلاصه شخصی عبارت Insert را درست در زمانی که Update مان در حال اجرا بوده انجام داده است، و با توجه به اینکه ردیف جدیدی بوده و قفلی وجود نداشته، به خوبی انجام شده است. تنها چاره این مشکل تنظیم سطح Serializable است و در این صورت بهنگام رسانی‌های جداول نباید درون بخش Where قرار گیرد، در غیر این صورت Lock خواهند شد.

3-4- Lost Update: زمانی روی می‌دهد که یک Update به طور موفقیت آمیزی در بانک اطلاعاتی نوشته می‌شود، اما به طور اتفاقی توسط تراکنش دیگری بازنویسی می‌شود. راه حل این مشکل بستگی به کد شما دارد و بایست به نحوی تشخیص دهید، بین زمانی که داده‌ها را می‌خوانید و زمانی که می‌خواهید آنرا بهنگام کنید، اتصال دیگری رکورد شما را بهنگام کرده است.

4- منابع قابل قفل شدن 6 منبع قابل قفل شدن برای SQL Server وجود دارد و آن‌ها سلسله مراتبی را تشکیل می‌دهند. هر چه سطح قفل بالاتر باشد، Granularity کمتری دارد. در ترتیب آبشاری Granularity عبارتند از:

- **Database:** کل بانک اطلاعاتی قفل شده است، معمولاً طی تغییرات Schema بانک اطلاعاتی روی می‌دهد.
- **Table:** کل جدول قفل شده است، شامل همه اشیای مرتبط با جدول.
- **Extent:** کل Extent (متشکل از هشت Page) قفل شده است.
- **Page:** همه داده‌ها یا کلیدهای Index در آن Page قفل شده اند.
- **Key:** قفلی در کلید مشخصی یا مجموعه کلید هایی Index وجود دارد. ممکن است سایر کلیدها در همان Index Page تحت تاثیر قرار نگیرند.
- **Row or Row Identifier (RID):** هر چند قفل از لحاظ فنی در Row Identifier قرار می‌گیرد ولی اساساً کل ردیف را قفل می‌کند.

5- تسریع قفل (Lock Escalation) و تاثیرات قفل روی عملکرد اگر تعداد آیتم‌های قفل شده کم باشد نگهداری سطح بهتری از Granularity (مثلاً RID به جای Page) معنی دار است. هرچند با افزایش تعداد آیتم‌های قفل شده، سربار مرتبط با نگهداری آن قفل‌ها در واقع باعث کاهش عملکرد می‌شود، و می‌تواند باعث شود قفل به مدت طولانی‌تری در محل باشد(هر چه قفل به مدت طولانی‌تری در محل باشد، احتمال این که شخصی آن رکورد خاص را بخواهد بیشتر است).
هنگامی که تعداد قفل نگهداری شده به آستانه خاصی برسد آن گاه قفل به بالاترین سطح بعدی افزایش می‌یابد و قفل‌های سطح پایین‌تر نباید به شدت مدیریت شوند (آزاد کردن منابع و کمک به سرعت در مجادله).
توجه شود که تسریع مبتنی بر تعداد قفل هاست و نه تعداد کاربران.

6- حالات قفل (Lock Modes): همانطور که دامنه وسیعی از منابع برای قفل شدن وجود دارد، دامنه ای از حالات قفل نیز وجود دارد.

6-1- Shared Locks (S): زمانی استفاده می‌شود، که فقط باید داده‌ها را بخوانید، یعنی هیچ تغییری ایجاد نخواهید کرد. Shared Lock با سایر Shared Lockهای دیگر سازگار است، البته قفل‌های دیگری هستند که با Shared Lock سازگار نیستند. یکی از کارهایی که Shared Lock انجام می‌دهد، ممانعت از انجام Dirty Read از طرف کاربران است.

6-2- Exclusive Locks (X): این قفل‌ها با هیچ قفل دیگری سازگار نیستند. اگر قفل دیگری وجود داشته باشد، نمی‌توان به Exclusive Lock دست یافت و همچنین در حالی که Exclusive Lock فعال باشد، به هر قفل جدیدی از هر شکل اجازه ایجاد شدن در منبع را نمی‌دهند.
این قفل از اینکه دو نفر همزمان به حذف کردن، بهنگام رسانی و یا هر کار دیگری مبادرت ورزند، پیشگیری می‌کند.

6-3- Update Locks (U): این قفل ها نوعی پیوند میان Shared Locks و Exclusive Locks هستند.
برای انجام Update باید بخش Where را (در صورت وجود) تایید اعتبار کنید، تا دریابید فقط چه ردیف هایی را می‌خواهید بهنگام رسانی کنید. این بدان معنی است که فقط به Shared Lock نیاز دارید، تا زمانی که واقعاً بهنگام رسانی فیزیکی را انجام دهید. در زمان بهنگام سازی فیزیکی نیاز به Exclusive Lock دارید.
Update Lock نشان دهنده این واقعیت است که دو مرحله مجزا در بهنگام رسانی وجود دارد، Shared Lock ای دارید که در حال تبدیل شدن به Exclusive Lock است. Update Lock تمامی Update Lockهای دیگر را از تولید شدن باز می‌دارند، و همچنین فقط با Shared Lock و Intent Shared Lockها سازگار هستند.

6-4- Intent Locks: با سلسله مراتب شی سر و کار دارد. بدون Intent Lock، اشیای سطح بالاتر نمی‌دانند چه قفلی را در سطح پایین‌تر داشته اید. این قفل‌ها کارایی را افزایش می‌دهند و 3 نوع هستند:

6-4-1- Intent Shared Lock (IS: Shared Lock) در نقطه پایین‌تری در سلسله مراتب، تولید شده یا در شرف تولید است. این نوع قفل تنها به Table و Page اعمال می‌شود.

6-4-2- Intent Exclusive Lock (IX): همانند Intent Shared Lock است اما در شرف قرار گرفتن در آیتم سطح پایین‌تر است.

6-4-3- Shared With Intent Exclusive (SIX: Shared Lock) در پایین سلسله مراتب شی تولید شده یا در شرف تولید است اما Intent Lock قصد اصلاح داده‌ها را دارد بنابراین در نقطه مشخصی تبدیل به Intent Exclusive Lock می‌شود.

6-5- Schema Locks: به دو شکل هستند:

6-5-1- Schema Modification Lock (Sch-M): تغییر Schema به شی اعمال شده است. هیچ پرس و جویی یا سایر عبارت‌های Create، Alter و Drop نمی‌توانند در مورد این شی در مدت قفل Sch-M اجرا شوند. با همه حالات قفل ناسازگار است.

6-5-2- Schema Stability Lock (Sch-S): بسیار شبیه به Shared Lock است، هدف اصلی این قفل پیشگیری از Sch-M است وقتی که قبلاً قفل هایی برای سایر پرس و جو-ها (یا عبارت‌های Create، Alter و Drop) در شی فعال شده اند. این قفل با تمامی انواع دیگر قفل سازگار است به جز با Sch-M.

6-6- Bulk Update Locks (BU): این قفل‌ها بارگذاری موازی داده‌ها را امکان پذیر می‌کنند، یعنی جدول در مورد هر فعالیت نرمال (عبارات T-SQL) قفل می‌شود، اما چندین عمل bcp یا Bulk Insert را می‌توان در همان زمان انجام داد. این قفل فقط با Sch-S و سایر قفل هایBU سازگار است.

7- سطوح جداسازی (Isolation Level):

7-1- Read Committed (وضعیت پیش فرض): با Read Committed همه Shared Lockهای ایجاد شده، به محض اینکه عبارت ایجاد کننده آنها تکمیل شود، به طور خودکار آزاد می‌شوند. به طور خلاصه قفل‌های مرتبط با عبارت Select به محض تکمیل عبارت Select آزاد می‌شوند و SQL Server منتظر پایان تراکنش نمی‌ماند. اگر تراکنش پرس و جویی را انجام می‌دهد که داده‌ها را اصلاح می‌کند (Insert، Delete و Update) قفل‌ها برای مدت تراکنش نگه داشته می‌شوند.
با این سطح پیش فرض، می‌توانید مطمئن شوید جامعیت کافی برای پیشگیری از Dirty Read دارید، اما همچنان Phantoms و Non-Repeatable Read می‌تواند روی دهد.

7-2- Read Uncommitted: خطرناک‌ترین گزینه از میان تمامی گزینه‌ها است، اما بالاترین عملکرد را به لحاظ سرعت دارد. در واقع با این تنظیم سطح تجربه همه مسائل متعدد هم زمانی مانند Dirty Read امکان پذیر است. در واقع با تنظیم این سطح به SQL Server اعلام می‌کنیم هیچ قفلی را تنظیم نکرده و به هیچ قفلی اعتنا نکند، بنابراین هیچ تراکنش دیگری را مسدود نمی‌کنیم.
می‌توانید همین اثر Read Uncommitted را با اضافه کردن نکته بهینه ساز **NOLOCK** در پرس و جوها بدست آورید.

7-3- Repeatable Read: سطح جداسازی را تا حدودی افزایش می‌دهد و سطح اضافی محافظت همزمانی را با پیشگیری از Dirty Read و همچنین Non-Repeatable Read فراهم می‌کند.
پیشگیری از Non-Repeatable Read بسیار مفید است اما حتی نگه داشتن Shared Lock تا زمان پایان تراکنش می‌تواند دسترسی کاربران به اشیا را مسدود کند، بنابراین به بهره وری لطمه وارد می‌کند.
نکته بهینه ساز برای این سطح **REPEATEABLEREAD** است.

7-4- Serializable: این سطح از تمام مسائل هم زمانی پیشگیری می‌کند به جز برای Lost Update.
این تنظیم سطح به واقع بالاترین سطح آنچه را که سازگاری نامیده می‌شود، برای پایگاه داده فراهم می‌کند. در واقع فرآیند بهنگام رسانی برای کاربران مختلف به طور یکسان عمل می‌کند به گونه ای که اگر همه کاربران یک تراکنش را در یک زمان اجرا می‌کردند، این گونه می‌شد « پردازش امور به طور سریالی».
با استفاده از نکته بهینه ساز **SERIALIZABLE** یا **HOLDLOCK** در پرس و جو شبیه سازی می‌شود.

7-5- Snapshot: جدترین سطح جداسازی است که در نسخه 2005 اضافه شد، که شبیه ترکیبی از Read Committed و Read Uncommitted است. به طور پیش فرض در دسترس نیست، در صورتی در دسترس است که گزینه ALLOW_SNAPSHOT_ISOLATION برای بانک اطلاعاتی فعال شده باشد.(برای هر بانک اطلاعاتی موجود در تراکنش)
Snapshot مشابه Read Uncommitted هیچ قفلی ایجاد نمی‌کند. تفاوت اصلی آن‌ها در این است که تغییرات صورت گرفته در بانک اطلاعاتی را در زمان‌های متفاوت تشخیص می‌دهند. هر تغییر در بانک اطلاعاتی بدون توجه به زمان یا Commit شدن آن، توسط پرس و جو هایی که سطح جداسازی Read Uncommitted را اجرا می‌کنند، دیده می‌شود. با Snapshot فقط تغییراتی که قبل از شروع تراکنش، Commit شده اند، مشاهده می‌شود.
از شروع تراکنش Snapshot، تمامی داده‌ها دقیقاً مشاهده می‌شوند، زیرا در شروع تراکنش Commit شده اند.
نکته: در حالی که Snapshot توجهی به قفل‌ها و تنظیمات آنها ندارد، یک حالت خاص وجود دارد. چنانچه هنگام انجام Snapshot یک عمل Rollback (بازیافت) بانک اطلاعاتی در جریان باشد، تراکنش Snapshot قفل‌های خاصی را برای عمل کردن به عنوان یک مکان نگهدار و سپس انتظار برای تکمیل Rollback تنظیم می‌کند. به محض تکمیل Rollback، قفل حذف شده و Snapshot به طور طبیعی به جلو حرکت خواهد کرد.

Isolation level	Dirty read	Non-Repeatable read	Phantom
Read uncommitted	Yes	Yes	Yes
Read committed	No	Yes	Yes
Repeatable read	No	No	Yes
Snapshot	No	No	No
Serializable	No	No	No

نظرات خوانندگان

نویسنده: محمد
تاریخ: ۱۳۹۳/۰۲/۰۵ ۱۲:۳۷

سلام ، خسته نباشید. ممنون از زحماتی که برای این مقاله کشیدید.
من یه سوال داشتم درباره تراکنش (Transaction) : من توی پروژه ام از sqlTransaction و isolation سریالایز استفاده کردم در این بلاک، 9 جدول من مورد عملیات درج و یا ویرایش قرار می‌گیره (و البته دو وب سرویس نیز صدا زده می‌شه) و معمولا 2 یا 3 ثانیه برای ثبت این تراکنش زمان صرف میشه . تا اینجا مشکلی نیست ولی اگر چند کاربر مثلا 5 نفر این عملیات انجام بدن در بعضی از مواقع (به ندرت) به ترتیب عملیات برای هر نفر انجام میشه (همین توقع از سریالایز می‌ره). ولی در اکثر مواقع خطا deadlock می‌ده و یا خطا timeout و همچنین جاهای مختلف برنامه که به یکی از این 9 جدول select می‌شه به انها هم همین خطاها رو می‌ده . لطفا در صورت امکان راه حلی برای مشکل من بگین . ممنون

نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۲/۰۵ ۱۵:۴۵

با سلام و سپاس از محبت تان، چند نکته به ذهن من میرسد، که شاید راهگشا باشد:
- تعداد دستورات درون بلاک Transaction تا جایی که میسر است، کمینه باشد.
- عملیات مربوط به وب سرویس، خارج از Transaction انجام گیرد. (به عنوان یک نکته در کار با Transactionباید عملیات Input و Output خارج از Transaction انجام گیرد.)
- طبیعتا هر چه درجه Isolation بالاتر باشد، Lock سختگیرانه‌تر برخورد می‌کند.
- برای مشکل بن بست، اگر می‌توانید ترتیب در اختیار گرفتن جداول را (پس از شناسائی محل بن بست) تغییر دهید و یا از جداول Temp استفاده کنید.
- چنانچه در قسمت هایی از برنامه همانطور که اشاره کردید فقط نیاز به خواندن مقادیر جداول دارید، از بهینه ساز پرس و جوی nolock استفاده کنید. برای مثال:

```
select * from tbl with (NOLOCK)
```

عنوان: معرفی و استفاده از DDL Triggers در SQL Server
نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۲/۰۳ ۱۱:۵۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Database

استفاده از DDL Trigger امکان ایجاد Trigger برای عملیات DDL(Data Definition Language) از SQL Server 2005 فراهم گردید. عملیاتی مانند ایجاد یک جدول جدید در بانک اطلاعاتی، اضافه شدن یک Login جدید و یا ایجاد یک بانک اطلاعاتی جدید را به وسیله این نوع Triggerها می‌توان کنترل نمود. در حقیقت DDL Trigger به شما اجازه می‌دهد که از تاثیر تعدادی از دستورات DDL جلوگیری کنید. بدین ترتیب که تقریباً هر دستور DDL به طور خودکار، تراکنشی (Transactional) اجرا می‌شود . می‌توان با دستور ROLLBACK TRANSACTION اجرای دستور DDL را لغو نمود. توجه شود همه دستورات DDL به صورت تراکنشی اجرا نمی‌شوند، به عنوان مثال دستور ALTER DATABASE ممکن است Database را تغییر دهد. در این صورت ساختار فایلی Database را تغییر می‌دهد، از آنجائی که سیستم عامل ویندوز به صورت تراکنشی عمل نمی‌کند بنابراین شما نمی‌توانید این عمل فایل سیستمی را لغو نمائید. به هر حال شما می‌توانید Trigger را با ALTER DATABASE فعال (fire) کنید برای عملیات Auditing، ولی نمی‌توان از انجام عمل ALTER DATABASE جلوگیری کرد.
برای نمونه می‌خواهیم از حذف و یا تغییر جداول یک بانک اطلاعاتی که به صورت عملیاتی در حال سرویس دهی است جلوگیری کنیم، برای اینکار از دستورهای زیر استفاده می‌کنیم:

```
create trigger Prevent_AlterDrop
on database
for drop_table, alter_table
as
  print 'table can not be dropped or altered'
  rollback transaction
```

از عبارت ON برای مشخص کردن محدوده Trigger در سطح SQL Instance (در این صورت ON All SERVER نوشته می‌شود) و یا در سطح Database (در این حالت ON DATABASE نوشته می‌شود) استفاده می‌شود و از عبارت FOR برای مشخص کردن رویداد یا گروه رویدادی که سبب فراخوانی Trigger می‌شود، استفاده خواهد شد.

1- معرفی تابع EVENTDATA() این تابع، یک تابع سیستمی مهم است که در DDL Trigger استفاده می‌شود. در حالیکه DDL Trigger در هر سطحی فعال (fire) شود تابع سیستمی EVENTDATA() فراخوانی (raise) می‌شود. خروجی تابع در قالب XML است . می‌توان اطلاعات را از تابع EVENTDATA دریافت کرد و آنها را در یک جدول با فیلدی از جنس XML و یا با استفاده از XPath Query ثبت کرد (Logging). عناصر کلیدی (Key Elements) تابع EVENTDATA به شرح زیر است:

- **EventType:** نوع رویدادی که باعث فراخوانی Trigger شده است.
- **PostTime:** زمانی که رویداد رخ می‌دهد.
- **SPID:** SPID کاربری که باعث ایجاد رویداد شده است.
- **ServerName:** نام SQL Instance که رویداد در آن رخ داده است.
- **LoginName:** نام Login که عمل مربوط به وقوع رویداد را اجرا می‌کند.
- **UserName:** نام User که عمل مربوط به وقوع رویداد را اجرا می‌کند.
- **DatabaseName:** نام Database که رویداد در آن رخ می‌دهد.
- **ObjectType:** نوع Object که اصلاح، حذف و یا ایجاد شده است.
- **ObjectName:** نام Object که اصلاح، حذف و یا ایجاد شده است.
- **TSQLCommand:** دستور T-SQL که اجرا شده و باعث اجرا شدن Trigger شده است.

2- بررسی یک سناریو نمونه

برای نمونه در دستورات زیر جدولی با نام ddl_log

```
CREATE TABLE ddl_log
  (
     EventType nvarchar(100),
 PostTime datetime,
```

```
 SPID nvarchar(100),
 ServerName nvarchar(100),
 LoginName nvarchar(100),
 UserName  nvarchar(100),
 DatabaseName  nvarchar(100),
 ObjectName  nvarchar(100),
 ObjectType  nvarchar(100),
 DefaultSchema nvarchar(100),
 [SID]  nvarchar(100),
 TSQLCommand nvarchar(2000));
```

و یک Trigger با نام log برای رویدادهایی که در سطح Database رخ می‌دهد، ایجاد می‌کنیم.

```
CREATE TRIGGER [Log] ON DATABASE
 FOR DDL_DATABASE_LEVEL_EVENTS
 AS
 DECLARE @data XML
 SET @data = EVENTDATA()
 INSERT INTO ddl_log
 VALUES (
   @data.value('(/EVENT_INSTANCE/EventType)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/PostTime)[1]',   'datetime'),
   @data.value('(/EVENT_INSTANCE/SPID)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/ServerName)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/LoginName)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/UserName)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/DatabaseName)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/ObjectName)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/ObjectType)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/DefaultSchema)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/SID)[1]',   'nvarchar(100)'),
   @data.value('(/EVENT_INSTANCE/TSQLCommand)[1]',   'nvarchar(2000)'));
```

نمونه ای از مقادیر ذخیره شده در جدول ddl_log به شکل زیر خواهد بود:

EventType	PostTime	SPID	ServerName	LoginName	UserName	DatabaseName	ObjectName
CREATE_TABLE	11/7/2010 10:36:26 AM	55	1936-603102405\SQL2008	sa	dbo	Test	Tmp_TestTable
ALTER_TABLE	11/7/2010 10:36:26 AM	55	1936-603102405\SQL2008	sa	dbo	Test	Tmp_TestTable
DROP_TABLE	11/7/2010 10:36:26 AM	55	1936-603102405\SQL2008	sa	dbo	Test	TestTable
RENAME	11/7/2010 10:36:26 AM	55	1936-603102405\SQL2008	sa	dbo	Test	Tmp_TestTable
DROP_TABLE	11/7/2010 10:38:26 AM	52	1936-603102405\SQL2008	sa	dbo	Test	TestTable

DefaultSchema	SID	TSQLCommand
NULL	NULL	CREATE TABLE dbo.Tmp_TestTable (a nvarchar(50) NULL) ON [PRIMARY]
NULL	NULL	ALTER TABLE dbo.Tmp_TestTable SET (LOCK_ESCALATION = TABLE)
NULL	NULL	DROP TABLE dbo.TestTable
NULL	NULL	EXECUTE sp_rename N'dbo.Tmp_TestTable', N'TestTable', 'OBJECT'
NULL	NULL	DROP TABLE TestTable ;

3- ملاحظات

در صورت فعال شدن Trigger می‌توان برخی موارد مانند محدودیت زمانی، کاربر اجرا کننده و ... را اضافه نمود. برای مثال در دستور زیر اجازه تغییرات در این زمان (بین 7:00 .A.M تا 8:00 .P.M) امکان پذیر نیست و در صورت اقدام پیغام خطا دریافت می‌کنید و دستورات Create لغو خواهند شد و اگر خارج از زمان فوق دستورات DDL را اجرا کنید دستورات به طور موفقیت آمیز اجرا می‌شود و البته تغییرات نیز Log می‌شوند.

```
IF DATEPART(hh,GETDATE()) > 7 AND DATEPART(hh,GETDATE()) < 20
BEGIN
    RAISERROR ('You can only perform this change between 8PM and 7AM. Please tr
    this change again or contact Production support for an override.', 16, 1)
ROLLBACK
END
```

این Trigger تاثیرات کمی بر روی کارایی دارد به این دلیل که معمولاً رویدادهای DDL به ندرت رخ می‌دهد. می‌توانید هنگامی که قصد دارید دستورات DDL را اجرا کنید موقتاً Trigger را با دستورات زیر غیر فعال نمائید:

```
DISABLE TRIGGER ALL ON DATABASE
DISABLE TRIGGER ALL ON ALL SERVER
```

پس از Overrdie کردن می‌توانید مجدداً Trigger را فعال کنید:

```
ENABLE TRIGGER ALL ON DATABASE
ENABLE TRIGGER ALL ON ALL SERVER
```

4- معرفی DDL Event Groups:

برای مشاهده جزئیات بیشتر می‌توانید به این لینک مراجعه کنید.

Server Level

DDL_SERVER_LEVEL_EVENTS

DDL_LINKED_SERVER_EVENTS

DDL_LINKED_SERVER_LOGIN_EVENTS

DDL_REMOTE_SERVER_EVENTS

DDL_EXTENDED_PROCEDURE_EVENTS

DDL_MESSAGE_EVENTS

DDL_ENDPOINT_EVENTS

DDL_SERVER_SECURITY_EVENTS

DDL_LOGIN_EVENTS

DDL_GDR_SERVER_EVENTS

DDL_AUTHORIZATION_SERVER_EVENT Database Level

Database Level

```
DDL_DATABASE_LEVEL_EVENTS
```

DDL_TABLE_VIEW_EVENTS

DDL_TABLE-EVENTS

DDL_VIEW_EVENTS

DDL_INDEX_EVENTS

DDL_STATISTICS_EVENTS

DDL_DATABASE_SECURITY_EVENTS

DDL_CERTIFICATE_EVENTS

DDL_USER_EVENTS

DDL_ROLE_EVENTS

DDL_APPLICATION_ROLE_EVENTS

DDL_SCHEMA_EVENTS

DDL_GDR_DATABASE_EVENTS

DDL_AUTHORIZATION_DATABASE_EVENTS

DDL_FUNCTION_EVENTS

DDL_PROCEDUER_EVENTS

DDL_TRIGGER_EVENTS

DDL_PARTITION_EVENTS

DDL_PARTITION_FUNCTION_EVENTS

DDL_PARTITION_SCHEME_EVENTS

DDL_SSB_EVENTS

DDL_MESSAGE_TYPE_EVENTS

DDL_CONTRACT_EVENTS

DDL_QUEUE_EVENTS

DDL_SERVER_EVENTS

DDL_ROUTE_EVENTS

DDL_REMOTE_SERVICE_BINDING_EVENTS

DDL_XML_SCHEMA_COLEECTION_EVENTS

DDL_FULLTEXT_CATALOG_EVENTS

DDL_DEFAULT_EVENTS

DDL_EXTENDED_PROPERTY_EVENTS

DDL_PLAN_GUIDE_EVENTS

DDL_RULE_EVENTS

DDL_SYNONYM_EVENTS

DDL_EVENT_NOTIFICATION_EVENTS

DDL_ASSEMBLY_EVENTS

DDL_TYPE_EVENTS

نظرات خوانندگان

نویسنده: محمد
تاریخ: ۱۳۹۳/۰۶/۲۵ ۲۰:۴۱

با سلام
اول از همه تشکر میکنم از شما دوست عزیز بابت زحماتی که کشیدید.
سئوال بنده اینه اگر بخواهیم خودمون یک نوع TRIGGER جدید به Database اضافه کنیم به چه صورت امکان پذیره؟
به طور مثال من می‌خوام یک تریگر فقط برای کلیه فعالیت‌ها در حوزه‌ی بکاپ و ریستور بنویسم.
مثل تریگر Alter table یا Drop_table که ماهیتشون از قبل تعریف شده است.

با تشکر

نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۷/۰۲ ۱۰:۳۴

با سلام و احترام؛ همانطور که در متن به عرض رسانده شده:
" از عبارت ON برای مشخص کردن محدوده Trigger در سطح SQL Instance (در این صورت ON All SERVER نوشته می‌شود) و یا در سطح Database (در این حالت ON DATABASE نوشته می‌شود) استفاده می‌شود و از عبارت FOR برای مشخص کردن رویداد یا گروه رویدادی که سبب فراخوانی Trigger می‌شود، استفاده خواهد شد. "
در خصوص مثالی که اشاره کردید، به نظرم می‌رسد از Trigger برای این منظور استفاده نمی‌شود (در حوزه‌ی بکاپ و ریستور)، شاید اگر قصدتان به منظور ثبت log و ... بایست از Auditing استفاده کنید. به این منظور در Auditing با توجه به جدول زیر می‌توان اقدام به ثبت موارد نمود:

Server-Level Audit Action Groups

Description	Event Class	Action group name
یک دستور Backup یا Restore صادر شود	Audit Backup/Restore	BACKUP_RESTORE_GROUP

به طور مختصر Auditing به شرح زیر است:

بررسی SQL Server Audit

بازبینی (Auditing) شامل پیگیری و ثبت رویدادهایی است که در سطح SQL Instance و یا Databaseهای روی یک سیستم اتفاق می‌افتد. چندین سطح برای Auditing در SQL Server وجود دارد که به صلاحدید و نیازمندی‌های نصب شما وابسته است. شما می‌توانید گروه اقدامات بازبینی سرویس دهنده (server audit action groups) را به ازای هر SQL Instance و گروه اقدامات بازبینی بانک اطلاعاتی (database audit action groups) را به ازای هر بانک اطلاعاتی ثبت کنید. رویداد Audit هر زمان عملی که مورد رسیدگی قرار گرفته اتفاق افتد، رخ می‌دهد.
تا پیش از SQL SERVER 2008، شما باید از خصیصه‌های متعددی برای انجام یک مجموعه کامل بازبینی (Auditing) برای نمونه DDL Trigger، DML Trigger و SQL Trace، بر روی یک SQL Instance استفاده می‌کردید.
SQL SERVER 2008، همه قابلیت‌های Auditing را روی یک audit specification ترکیب می‌کند. Audit Specification با تعریف یک شی بازبینی (audit object) در سطح سرویس دهنده برای ثبت (logging) یک دنباله بازبینی (audit trial) آغاز می‌شود.
توجه شود که بایست یک شیء بازبینی ایجاد کنید پیش از اینکه یک Server Audit Specification و یا Database Audit Specification ایجاد کنید.
Server Audit Specification، گروه اقدامات در سطح سرویس دهنده را جمع آوری می‌کند که با رویدادهای وسیعی فعال می‌شوند، این گروه اقدامات تحت عنوان Server-Level Audit Action Groups تشریح شده اند. شما می‌توانید یک Server Audit Specification را به ازای هر Audit ایجاد کنید چرا که هر دو در محدوده یک SQL Instance ایجاد می‌شوند.
Database Audit Specification، گروه اقدامات در سطح بانک اطلاعاتی را جمع آوری می‌کند که با رویدادهای وسیعی فعال می‌شود. این گروه اقدامات تحت عنوان‌های Database-Level Audit Action Groups و Database-Level Audit Actions تشریح

شده اند.می توانید یک Database Audit Specification را به ازای هر Audit در بانک اطلاعاتی SQL Server ایجاد کنید.
همچنین می‌توانید هر گروه اقدامات بازبینی(audit action groups) یا رویدادهای بازبینی(audit events) را به یک Database Audit Specification اضافه کنید. گروه اقدامات بازبینی، گروه اقدامات از پیش تعریف شده ای هستند و رویدادهای بازبینی اقدامات تجزیه ناپذیری هستند که توسط موتور بانک اطلاعاتی مورد رسیدگی قرار می‌گیرند، هر دو در محدوده بانک اطلاعاتی (Database) هستند. این اقدامات برای Audit فرستاده می‌شوند تا در Target (که می‌تواند یک فایل، Windows Security Log و یا Windows Application Log باشد) ذخیره شوند. برای نوشتن در Windows Security Log لازم است که Service Account سرویس دهنده شما به Policy، Generate security audits اضافه شده باشد، به صورت پیش فرض Local System، Local Service و Network Service بخشی از این Policy می‌باشند. پس از اینکه Audit را ایجاد و فعال کردید، Target ورودی‌ها را دریافت خواهد کرد.

Server-Level Audit Action Groups (گروه اقدامات بازبینی در سطح سرویس دهنده)

این گروه اقدامات به گروه رویداد Security Audit شبیه هستند. به طور خلاصه این گروه اقدامات، اقداماتی را که در یک SQL Instance شامل می‌شوند، در بر می‌گیرد. برای مثال اگر گروه اقدام مناسب با Server Audit Specification اضافه شده باشد هر شیء در هر Schema که مورد دستیابی قرار می‌گیرد، ثبت می‌شود. اقدامات در سطح سرویس دهنده به شما اجازه نمی‌دهد که جزئیات اقدامات در سطح بانک اطلاعاتی را فیلتر کنید. یک بازبینی در سطح بانک اطلاعاتی برای انجام به جزئیات دقیق فیلتر کردن نیاز دارد، برای مثال اجرای دستور Select روی جدول Customers برای login هایی که در گروه Employee هستند.

Database-Level Audit Action Groups (گروه اقدامات بازبینی در سطح بانک اطلاعاتی)

این گروه اعمال به کلاس‌های رویداد Security Audit شبیه هستند.

Database-Level Audit Actions

اقدامات در سطح بانک اطلاعاتی، اقدامات بازبینی خاصی را به طور مستقیم روی Database، Schema و اشیاء Schema (از قبیل جداول، View ها، رویه‌های ذخیره شده، توابع و ...) فراهم می‌کند. این اقدامات برای فیلدها (Columns) صدق نمی‌کنند.

Audit-Level Audit Action Groups

شما می‌توانید اقداماتی را که در فرآیند Auditing هستند، بازبینی کنید که می‌تواند در محدوده سرویس دهنده یا بانک اطلاعاتی باشد. در محدوده بانک اطلاعاتی تنها برای database audit specification رخ می‌دهد.
جهت بررسی بیشتر به این لینک مراجعه شود.

نویسنده: MF
تاریخ: ۱۳۹۳/۰۷/۰۲ ۱۷:۷

DDL Triggerها , حالت Befor ندارند(انجام یک سری کارها قبل از عملیات درج,حذف و ...).برای حل این مساله باید از INSTEAD OF در تریگر استفاده نمود ؟

نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۷/۰۳ ۱۲:۱

مواردی که اشاره کردید مربوط به DML Triggers می‌باشند.
برای اطلاعات بیشتر به این لینک Understanding DDL Triggers vs. DML Triggers مراجعه شود.

نویسنده: مجید فاضلی
تاریخ: ۱۳۹۳/۰۷/۰۶ ۱۲:۴۶

باید از از حالت INSTEAD OF استفاده کنیم در **DML** Trigger ای که قراره نوشته بشه.
می‌توانیم در یک جدول از دیتابیس مان بر اساس یک شرط خاص, عملیات Insert,Delete,Update را مدیریت کنیم.
بعنوان مثال در قطعه کد زیر ما قبل از عملیات Insert در جدول tblTest چک میکنیم که اگر مقدار ستون FirstName برابر با null بود عملیات Insert آن رکورد در دیتابیس لغو شود.

```
ALTER TRIGGER [dbo].[Prevent_Befor_Insert_Null]
   ON  [dbTest].[dbo].[tblTest]
   INSTEAD OF INSERT
AS
BEGIN
```

```
SET NOCOUNT ON
IF OBJECT_ID(N'dbTest.dbo.tblTest.FirstName') is null
BEGIN
DECLARE @Id int
SET @Id = (select Id from inserted)
RAISERROR ('16,1,'مقدار فیلد نام نباید خالی باشد)
ROLLBACK
END
END
```

از دو طریق می‌توان به مقادیر فیلدهای رکورد جاری دسترس داشت:

1- استفاده از OBJECT_ID و ذکر نام فیلد مورد نظر

2- گرفتن فیلد مورد نظر از جدول INSERTED یا DELETED

DML Triggerها دارای دو جدول خاص بنام‌های INSERTED و DELETED هستند که توسط خود SQL Server مدیریت می‌شوند.در حقیقت در پشت صحنه, ما با این دو جدول در هنگام تغییر مقادیر داده‌های جداول دیتابیس کار می‌کنیم و نمی‌توانیم بصورت مستقیم داده‌های جداول موجود در دیتا بیس مان را تغییر دهیم.

جدول INSERTED و DELETED حاوی رکورد جاری است که تحت تاثیر عمل درج, ویرایش و حذف در دیتابیس قرار گرفته است.

اطلاعات بیشتر در اینجا و اینجا

نویسنده: محمد
تاریخ: ۱۳۹۳/۰۷/۲۸ ۰:۱۵

سلام؛ تشکر از توضیحات شما. اجازه بدید من طور دیگری سئوالم رو مطرح کنم. به طور مثال ما برای کار با تاریخ شمسی در SQl چندین روش پیش رو داریم که وارد جزئیات آن نمیشوم ولی یکی از این روش‌ها که به خوبی جواب میدهد استفاده از CLR است که ما با توسط این قابلیت می‌توانیم یک نوع دیتا تایپ جدید ، با ماهیت جدید در اس کیو ال اضافه کنیم. حالا منظور بنده این است که آیا برای تریگرها هم می‌شود این کار را انجام داد یا خیر؟ مثلا توسط CLR یا هر روش دیگری که وجود دارد ، ما بیاییم و یک نوع تریگر کاملا جدید و Customize شده برای خودمان درست کنیم. به طور مثال : زمانی که کاربر از دیتابیس بخواهد بکاپ تهیه کند یا آن را ریستور کند ، یکسری فعالیتها به آن فعالیت اضافه شود یا در راستای آن انجام شود. با تشکر

عنوان: بررسی دو نکته (ترفند) کاربردی در SQL Server
نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۲/۰۳ ۱۶:۲۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Database

1- اندازه گیری تعداد Transactionها در واحد زمان روی یک Database خاص در SQL Server جهت بدست آوردن تعداد Transactionها در واحد زمان(Transactions Per Second) روی یک Database خاص در یک سیستم عملیاتی، جهت ارتقاء سخت افزاری ، تست فشار و ... می‌توانید از یک DMV با نام sys.dm_os_performance_counters به طریق زیر استفاده نمائید:

```
declare @cntr_value bigint

Select @cntr_value=cntr_value
from sys.dm_os_performance_counters
where instance_name='AdventureWorks' and
counter_name='Write Transactions/sec'

/* ایجاد یک تاخیر مثلاً یک ثانیه */
waitfor delay '00:00:01'

Select cntr_value -@cntr_value
from sys.dm_os_performance_counters
where instance_name='AdventureWorks' and
counter_name='Write Transactions/sec'
```

View معرفی شده تمامی شمارنده‌های عملکردی را برای یک Instance خاص شامل می‌شود، ستون instance_name برابر نام بانک اطلاعاتی مورد نظر می‌باشد.

2- sys.sp_MSforeachtable

از رویه‌های ذخیره شده UnDocumented در SQL Server می‌باشد و این قابلیت را دارا است که برای هر یک از جداول موجود در یک بانک اطلاعاتی، یک رویه‌ای را اجرا کند. برای مثال با استفاده از دستور زیر، می‌توانید تعداد سطرها، اندازه‌ی داده‌ها و ایندکس‌های یک جدول را بدست آورید

```
EXEC sys.sp_MSforeachtable 'sp_spaceused ''?''';
```

به عنوان یک مثال کاربردی، با اجرای دستور زیر می‌توان جداول بانک اطلاعاتی مورد نظرتان را از لحاظ معیارهایی که پیشتر ذکر آن رفت، مورد بررسی قرار دهید.

```
USE [AdventureWorksDW2008R2]
GO

CREATE TABLE #TableSpaceUsed(
[name] [nvarchar](120) NULL,
[rows] [nvarchar](120) NULL,
[reserved] [nvarchar](120) NULL,
[data] [nvarchar](120) NULL,
[index_size] [nvarchar](120) NULL,
[unused] [nvarchar](120) NULL
) ON [PRIMARY]

Insert Into #TableSpaceUsed
EXEC sys.sp_MSforeachtable 'sp_spaceused ''?''';

Select * from #TableSpaceUsed
Order by CAST([rows] as int) desc

Drop table #TableSpaceUsed
```

خروجی مثال فوق به شکل زیر است.

	name	rows	reserved	data	index_size	unused
1	FactInternetSalesReason	64515	2208 KB	2048 KB	48 KB	112 KB
2	FactResellerSales	60855	25736 KB	11864 KB	13008 KB	864 KB
3	FactInternetSales	60398	20464 KB	8200 KB	11312 KB	952 KB
4	FactFinance	39409	2184 KB	2152 KB	8 KB	24 KB
5	DimCustomer	18484	8832 KB	7800 KB	904 KB	128 KB
6	FactAdditionalInternationalProductDescription	15168	4216 KB	4008 KB	64 KB	144 KB
7	FactCurrencyRate	14264	584 KB	472 KB	16 KB	96 KB
8	FactSurveyResponse	2727	352 KB	192 KB	160 KB	0 KB
9	ProspectiveBuyer	2059	728 KB	632 KB	88 KB	8 KB
10	DimDate	1188	232 KB	152 KB	48 KB	32 KB
11	DimReseller	701	280 KB	192 KB	88 KB	0 KB
12	DimGeography	655	136 KB	104 KB	16 KB	16 KB
13	DimProduct	606	6120 KB	5936 KB	88 KB	96 KB
14	DimEmployee	296	176 KB	120 KB	48 KB	8 KB
15	FactSalesQuota	163	48 KB	8 KB	40 KB	0 KB
16	FactCallCenter	120	48 KB	16 KB	32 KB	0 KB
17	DatabaseLog	115	408 KB	384 KB	8 KB	16 KB
18	DimCurrency	105	32 KB	8 KB	24 KB	0 KB
19	DimAccount	99	48 KB	16 KB	32 KB	0 KB
20	DimProductSubcategory	37	32 KB	8 KB	24 KB	0 KB
21	DimPromotion	16	32 KB	8 KB	24 KB	0 KB
22	DimOrganization	14	16 KB	8 KB	8 KB	0 KB
23	DimSalesTerritory	11	32 KB	8 KB	24 KB	0 KB
24	DimSalesReason	10	16 KB	8 KB	8 KB	0 KB
25	DimDepartmentGroup	7	16 KB	8 KB	8 KB	0 KB
26	DimProductCategory	4	32 KB	8 KB	24 KB	0 KB
27	DimScenario	3	16 KB	8 KB	8 KB	0 KB
28	AdventureWorksDWBuildVersion	1	72 KB	8 KB	8 KB	56 KB

نظرات خوانندگان

نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۶/۱۸ ۱۲:۵۴

یک مورد استفاده کاربردی، از رویه **sp_MSforeachtable** به هنگام تجمیع بانک‌های اطلاعاتی است (برای مثال تجمیع دو DB کوچک در یک DB بزرگتر). برای این منظور می‌توان در ابتدا تمامی Constraintهای جداول را موقتاً غیر فعال کرد، عملیات Load داده در جداول را انجام داد و مجدداً آنها را فعال نمود.

چنانچه از SSIS Package برای اینکار استفاده شود، با فرض اینکه در مرحله قبل Schema تمامی اشیاء بانک منتقل شده است، Control Flow این Package شامل Stepهای زیر میباشد:

گام اول - غیر فعال کردن تمامی Constraintهای جداول

برای این منظور از Object موسوم به Execute SQL Task به این صورت استفاده میشود که در بخش SQL Statement دستور زیر نوشته شود:

```
exec sp_MSforeachtable 'ALTER TABLE ? NOCHECK CONSTRAINT ALL'
```

گام دوم - انتقال داده‌ها به شکل Data Only

برای این منظور از Object موسوم به Transfer SQL Server Objects Task به صورت زیر استفاده میشود :

در بخش Object تغییرات زیر اعمال شود:

- در قسمت Destination ، پروپرتی CopyData با مقدار True و ExistingData با مقدار Append تنظیم شود.
- در قسمت Destination Copy & Option ، پروپرتی CopyAllTables در بخش ObjectsToCopy با مقدارTrue تنظیم شود.

گام سوم - فعال کردن مجدد Constraintهای جداول

برای این منظور از Object موسوم به Execute SQL Task به صورت زیر استفاده میشود :

در بخش SQL Statement دستور زیر نوشته شود:

```
exec sp_MSforeachtable 'ALTER TABLE ? CHECK CONSTRAINT ALL'
```

نکته: در صورت وجود Trigger برای جداول بانک اطلاعاتی نیز می‌توان به همین شکل عمل نمود. (ابتدا آنها را غیر فعال و مجدا فعال کرد)

برای مثال شکل SSIS Package فوق به صورت تصویر فوق است.

عنوان: بررسی ابزار SQL Server Profiler
نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۲/۰۶ ۱۰:۱۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Database, Performance

مقدمه Profiler یک ابزار گرافیکی برای ردیابی و نظارت بر کارآئی SQL Server است. امکان ردیابی اطلاعاتی در خصوص رویدادهای مختلف و ثبت این داده‌ها در یک فایل (با پسوند trc) یا جدول برای تحلیل‌های آتی نیز وجود دارد. برای اجرای این ابزار مراحل زیر را انجام دهید:

```
Start > Programs> Microsoft SQL Server > Performance Tools> SQL Server Profiler
```

و یا در محیط Management Studio از منوی Tools گزینه SQL Server Profiler را انتخاب نمائید.

1- اصطلاحات

1-1- رویداد (Event): یک رویداد، کاری است که توسط موتور بانک اطلاعاتی (Database Engine) انجام می‌شود. برای مثال هر یک از موارد زیر یک رویداد هستند.

- متصل شدن کاربران (login connections) قطع شدن ارتباط یک login
- اجرای دستورات T-SQL، شروع و پایان اجرای یک رویه، شروع و پایان یک دستور در طول اجرای یک رویه، اجرای رویه‌های دور Remote Procedure Call
- باز شدن یک Cursor
- بررسی و کنترل مجوزهای امنیتی

1-2- کلاس رویداد (Event Class): برای بکارگیری رویدادها در Profiler، از یک Event Class استفاده می‌کنیم. یک Event Class رویدادی است که قابلیت ردیابی دارد. برای مثال بررسی ورود و اتصال کاربران با استفاده از کلاس Audit Login قابل پیاده سازی است. هر یک از موارد زیر یک Event Class هستند.

- SQL:BatchCompleted
- Audit Login
- Audit Logout
- Lock: Acquired
- Lock: Released

1-3- گروه رویداد (Event Category): یک گروه رویداد شامل رویدادهایی است که به صورت مفهومی دسته بندی شده اند. برای مثال، کلیه رویدادهای مربوط به قفل‌ها از جمله Lock: Acquired (بدست آوردن قفل) و Lock: Released (رها کردن قفل) در گروه رویداد Locks قرار دارند.

1-4- ستون داده ای (Data Column): یک ستون داده ای، خصوصیت و جزئیات یک رویداد را شامل می‌شود. برای مثال در یک Trace که رویدادهای Lock: Acquired را نظارت می‌کند، ستون Binary Data شامل شناسه (ID) یک صفحه و یا یک سطر قفل شده است و یا اینکه ستون Duration مدت زمان اجرای یک رویه را نمایش می‌دهد.

1-5- الگو (Template): یک الگو، مشخص کننده تنظیمات پیش گزیده برای یک Trace است، این تنظیمات شامل رویدادهایی است که نیاز دارید بر آنها نظارت داشته باشید. هنگامیکه یک Trace براساس یک الگو اجرا شود، رویدادهای مشخص شده، نظارت می‌شوند و نتیجه به صورت یک فایل یا جدول قابل مشاهده خواهد بود.

1-6- ردیاب (Trace): یک Trace داده‌ها را براساس رویدادهای انتخاب شده، جمع آوری می‌کند. امکان اجرای بلافاصله یک Trace برای جمع آوری اطلاعات با توجه به رویدادهای انتخاب شده و ذخیره کردن آن برای اجرای آتی وجود دارد.

1-7- فیلتر (Filter): هنگامی که یک Trace یا الگو ایجاد می‌شود، امکان تعریف شرایطی برای فیلتر کردن داده‌های جمع آوری شده نیز وجود دارد. این کار باعث کاهش حجم داده‌های گزارش شده می‌شود. برای مثال اطلاعات مربوط به یک کاربر خاص جمع آوری می‌شود و یا اینکه رشد یک بانک اطلاعاتی مشخص بررسی می‌شود.

2- انتخاب الگو (Profiler Trace Templates) از آنجائیکه اصولاً انتخاب Eventهای مناسب، کار سخت و تخصصی می‌باشد برای راحتی کار تعدادی Templateهای آماده وجود دارد، برای مثال TSQL_Duration تاکیدش روی مدت انجام کار است و یا SP_Counts در مواردی که بخواهیم رویه‌های ذخیره شده را بهینه کنیم استفاده می‌شود در جدول زیر به شرح هر یک پرداخته شده است:

الگو	هدف
Blank	ایجاد یک Trace کلی
SP_Counts	ثبت اجرای هر رویه ذخیره شده برای تشخیص اینکه هر رویه چند بار اجرا شده است
Standard	ثبت آمارهای کارائی برای هر رویه ذخیره شده و Queryهای عادی SQL که اجرا می‌شوند و عملیات ورود و خروج هر Login (پیش فرض)
TSQL	ثبت یک لیست از همه رویه‌های ذخیره شده و Queryهای عادی SQL که اجرا می‌شوند ولی آمارهای کارائی را شامل نمی‌شود
TSQL_Duration	ثبت مدت زمان اجرای هر رویه ذخیره شده و هر Query عادی SQL
TSQL_Grouped	ثبت تمام loginها و logoutها در طول اجرای رویه‌های ذخیره شده و هر Query عادی SQL، شامل اطلاعاتی برای شناسائی برنامه و کاربری که درخواست را اجرا می‌کند
TSQL_Locks	ثبت اطلاعات انسداد (blocking) و بن بست (deadlock) از قبیل deadlock graphs، deadlock chains، blocked processes,... . این الگو همچنین درخواست‌های تمام رویه‌های ذخیره شده و تمامی دستورات هر رویه و هر Query عادی SQL را دریافت می‌کند
TSQL_Replay	ثبت اجرای رویه‌های ذخیره شده و Queryهای SQL در یک SQL Instance و مهیا کردن امکان اجرای دوباره عملیات در سیستمی دیگر
TSQL_SPs	ثبت کارائی برای Queryهای SQL، رویه‌های ذخیره شده و تمامی دستورات درون یک رویه ذخیره شده و نیز عملیات ورود و خروج هر Login
Tuning	ثبت اطلاعات کارائی برای Queryهای عادی SQL و رویه‌های ذخیره شده و یا تمامی دستورات درون یک رویه ذخیره شده

3- انتخاب رویداد (SQL Trace Event Groups) رویدادها در 21 گروه رویداد دسته بندی می‌شوند که در جدول زیر لیست شده اند:

گروه رویداد	هدف
Broker	13 رویداد برای واسطه سرویس (Service Broker)
CLR	1 رویداد برای بارگذاری اسمبلی‌های CLR (Common Language Runtime)
Cursors	7 رویداد برای ایجاد، دستیابی و در اختیار گرفتن Cursor
Database	6 رویداد برای رشد/کاهش (grow/shrink) فایل های Data/Log همچنین تغییرات حالت انعکاس (Mirroring)
Deprecation	2 رویداد برای آگاه کردن وضعیت نابسامان درون یک SQL Instance
Errors and Warnings	16 رویداد برای خطاها، هشدارها و پیغام‌های اطلاعاتی که ثبت شده است
Full Text	3 رویداد برای پیگیری یک شاخص متنی کامل
Locks	9 رویداد برای بدست آوردن، رها کردن قفل و بن بست (Deadlock)
OLEDB	5 رویداد برای درخواست‌های توزیع شده و RPC (اجرای رویه‌های دور)
Objects	3 رویداد برای وقتی که یک شی ایجاد، تغییر یا حذف می‌شود
Performance	14 رویداد برای ثبت نقشه درخواست‌ها (Query Plan) برای استفاده نقشه راهنما (Plan Guide) به منظور بهینه سازی کارائی درخواست ها، همچنین این گروه رویداد در خواست‌های متنی کامل (full text) را ثبت می‌کند
Progress Report	10 رویداد برای ایجاد Online Index
Query Notifications	4 رویداد برای سرویس اطلاع رسان (Notification Service)
Scans	2 رویداد برای وقتی که یک جدول یا شاخص، پویش می‌شود
Security Audit	44 رویداد برای وقتی که مجوزی استفاده شود، جابجائی هویتی رخ دهد، تنظیمات امنیتی اشیائی تغییر کند،یک SQL Instance شروع و متوقف شود و یک Database جایگزین شود یا از آن پشتیبان گرفته شود
Server	3 رویداد برای Mount Tape، تغییر کردن حافظه سرور و بستن یک فایل Trace
Sessions	3 رویداد برای وقتی که Connectionها موجود هستند و یک Trace فعال می‌شود، همچنین یک Trigger و یک تابع دسته بندی(classification functions) مربوط به مدیریت منابع(resource governor) رخ دهد
Stored Procedures	12 رویداد برای اجرای یک رویه ذخیره شده و دستورات درون آن ، کامپایل مجدد و استفاده از حافظه نهانی (Cache)
Transactions	13 رویداد برای شروع، ذخیره ، تائید و لغو یک تراکنش
TSQL	9 رویداد برای اجرای Queryهای SQL و جستجوهای XQUERY (در داده‌های XML)

گروه رویداد	هدف
User Configurable	10 رویداد که شما می‌توانید پیکربندی کنید

به طور معمول بیشتر از گروه رویدادهای Locks، Performance، Security Audit، Stored Procedures و TSQL استفاده می‌شود.

4- انتخاب ستون‌های داده ای (Data Columns) اگرچه می‌توان همه‌ی 64 ستون داده ای ممکن را برای ردیابی انتخاب کرد ولیکن داده‌های Trace شما زمانی مفید خواهند بود که اطلاعات ضروری را ثبت کرده باشید. برای مثال شماره ترتیب تراکنش‌ها را، برای یک رویداد RPC:Completed می‌توانید برگردانید، اما همه رویه‌های ذخیره شده مقادیر را تغییر نمی‌دهند بنابراین شماره ترتیب تراکنش‌ها فضای بیهوده ای را مصرف می‌کند. بعلاوه همه ستون‌های داده ای برای تمامی رویدادهای Trace معتبر نیستند. برای مثال CPU، Write ، Read و Duration برای رویدادهای RPC:Starting و SQL:BatchStarting معتبر نیستند.
ApplicationName، NTUserName، LoginName، ClientProcessID، SPID، HostName، LoginSID، NTDomainName و SessionLoginName ، مشخص می‌کنند چه کسی و از چه منشاء دستور را اجرا کرده است.
ستون SessionLoginName معمولاً نام Login ای که از آن برای متصل شدن به SQL Instance استفاده شده است را نشان می‌دهد. در حالیکه ستون LoginName نام کاربری را که دستور را اجرا می‌کند نشان می‌دهد (EXECUTE AS). ستون ApplicationName خالی است مگر اینکه در ConnectionString برنامه کاربردیمان این خصوصیت (Property) مقداردهی شده باشد. ستون StartTime و EndTime زمان سرحدی برای هر رویداد را ثبت می‌کند این ستون‌ها بویژه در هنگامی که به عملیات Correlate نیاز دارید مفید هستند.

5- بررسی چند سناریو نمونه

• **یافتن درخواست هائی (Queries) که بدترین کارایی را دارا هستند.** برای ردیابی درخواست‌های ناکارا، از رویداد RPC:Completed از دسته Stored Procedure و رویداد SQL:BatchCompleted از دسته TSQL استفاده می‌شود.

• **نظارت بر کارایی رویه ها** برای ردیابی کارائی رویه ها، از رویدادهای SP:Starting، SP:Completed، SP:StmtCompleted و SP:StmtStaring از کلاس Stored Procedure و رویدادهای SQL:BatchCompleted ، SQL:BatchStarting از کلاس TSQL استفاده می‌شود.

• **نظارت بر اجرای دستورات T-SQL توسط هر کاربر** برای ردیابی دستوراتی که توسط یک کاربر خاص اجرا می‌شود، نیاز به ایجاد یک Trace برای نظارت بر رویدادهای کلاس‌های Sessions، ExistingConnection و TSQL داریم همچنین لازم است نام کاربر در قسمت فیلتر و با استفاده از DBUserName مشخص شود.

• **اجرا دوباره ردیاب (Trace Replay)** این الگو معمولاً برای debugging استفاده می‌شود برای این منظور از الگوی Replay استفاده می‌شود. در ضمن امکان اجرای دوباره عملیات در سیستمی دیگر با استفاده از این الگو مهیا می‌شود.

• **ابزار Tuning Advisor (راهنمای تنظیم کارائی)** این ابزاری برای تحلیل کارائی یک یا چند بانک اطلاعاتی و تاثیر عملکرد آنها بر بار کاری (Workload) سرویس دهنده است. یک بار کاری مجموعه ای از دستورات T-SQL است که روی بانک اطلاعاتی اجرا می‌شود. بعد از تحلیل تاثیر بارکاری بر بانک اطلاعاتی، Tuning Advisor توصیه هائی برای اضافه کردن، حذف و یا تغییر طراحی فیزیکی ساختار بانک اطلاعاتی ارائه می‌دهد این تغییرات ساختاری شامل پیشنهاد برای تغییر ساختاری موارد Clustered Indexes، Indexed View، Nonclustered Indexes و Partitioning است.
برای ایجاد بارکاری می‌توان از یک ردیاب تهیه شده در SQL Profiler استفاده کرد برای این منظور از الگوی Tuning استفاده می‌شود و یا رویدادهای RPC:Completed، SQL:BatchCompleted و SP:StmtCompleted را ردیابی نمائید.

• **ترکیب ابزارهای نظارتی (Correlating Performance and Monitoring Data)** یک Trace برای ثبت اطلاعاتی که در یک SQL Instance رخ می‌دهد، استفاده می‌شود. System Monitor برای ثبت شمارنده‌های کارائی(performance counters) استفاده می‌شود و همچنین از منابع سخت افزاری و اجزای دیگر که روی سرور اجرا می‌شوند، تصاویری فراهم می‌کند. توجه شود که در مورد Correlating یک فایل ردیاب (trace file) و یک Counter Log (ابزار Performance)، ستون داده ای StartTime و EndTime باید انتخاب شود، برای این کار از منوی File گزینه Import Performance Data انتخاب می‌شود.

• **جستجوی علت رخ دادن یک بن بست** برای ردیابی علت رخ دادن یک بن بست، از رویدادهای RPC:Starting، SQLBatchStarting از دسته Stored Procedure و رویدادهای Lock:Deadlock، Deadlock graph و Lock:Deadlock Chain از دسته Locks استفاده می‌شود. (در صورتی که نیاز به یک ارائه گرافیکی دارید از Deadlock graph استفاده نمائید، خروجی مطابق تصویر زیر می‌شود).

5-1- ایجاد یک Trace 1- Profiler را اجرا کنید از منوی File گزینه New Trace را انتخاب کنید و به SQL Instance مورد نظرتان متصل شوید.

2- مطابق تصویر زیر برای Trace یک نام و الگو و تنظیمات ذخیره سازی فایل را مشخص کنید.

3- بر روی قسمت Events Selection کلیک نمائید.

4- مطابق تصویر زیر رویدادها و کلاس رویدادها را انتخاب کنید، ستون‌های TextData، NTUserName، LoginName، CPU،Reads،Writes، Duration، SPID، StartTime، EndTime، BinaryData، DataBaseName، ServerName و ObjectName را انتخاب کنید.

5- روی Column Filters کلیک کنید و مطابق تصویر زیر برای DatabaseName فیلتری تنظیم کنید.

6- روی Run کلیک کنید. تعدادی Query و رویه ذخیره شده مرتبط با پایگاه داده AdventureWorks اجرا کنید .

5-2- ایجاد یک Counter Log برای ایجاد یک Counter Log مراحل زیر را انجام دهید:

1- ابزار Performance را اجرا کنید (برای این کار عبارتPerfMon را در قسمت Run بنویسید).

2- در قسمت Counter Logs یک log ایجاد کنید.

3- روی Add Counters کلیک کرده و مطابق تصویر موارد زیر را انتخاب کنید.

Performance Object	Select counters from list
Network Interface	Output Queue Length
Processor	% Processor Time
System	Processor Queue Length
SQLServer	Buffer Manager:Page life expectancy

4- روی Ok کلیک کنید تا Counter Log ذخیره شود سپس روی آن راست کلیک کرده و آنرا Start کنید.

5-3- ترکیب ابزارهای نظارتی (Correlating SQL Trace and System Monitor Data) 1- Profiler را اجرا کنید از منوی File گزینه Open و سپس Trace File را انتخاب کنید فایل trc را که در گام اول ایجاد کردید، باز نمائید.

2- از منوی File گزینه Import Performance Data را انتخاب کنید و فایل counter log را که در مرحله قبل ایجاد کردید انتخاب کنید.

نکته: اطلاعات فایل trc را می‌توان درون یک جدول وارد کرد، بدین ترتیب می‌توان آنالیز بیشتری داشت به عنوان مثال دستورات زیر این عمل را انجام می‌دهند.

```
SELECT * INTO dbo.BaselineTrace
FROM fn_trace_gettable(' c:\performance baseline.trc ', default);
```

با اجرای دستور زیر جدولی با نام BaselineTrace ایجاد و محتویات Trace مان (performance baseline.trc) در آن درج می‌گردد.

عنوان: **اجرای SSIS Package از طریق برنامه کاربردی**
نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۲/۰۷ ۱۷:۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SSIS

مقدمه

در اکثر موارد در یک Landscape عملیاتی، چنانچه به تجمیع و انتقال داده‌ها از بانک‌های اطلاعاتی مختلف نیاز باشد، از SSIS Package اختصار (SQL Server Integration Service) استفاده می‌شود و معمولاً با تعریف یک Job در سطح SQL Server به اجرای Package در زمانهای مشخص می‌پردازند. چنانچه در موقعیتی لازم باشد که از طریق برنامه کاربردی توسعه یافته، به اجرای Package مبادرت ورزیده شود و البته نخواهیم Job تعریف شده را از طریق کد برنامه، اجرا کنیم و در واقع این امکان را داشته باشیم که همانند یک رویه ذخیره شده تعریف شده در سطح بانک اطلاعاتی به اجرای عمل فوق بپردازیم، یک راه حل می‌تواند تعریف یک CLR Stored Procedures باشد. در این مقاله به بررسی این موضوع پرداخته می‌شود، در ابتدا لازم است به بیان تئوری موضوع پرداخته شود (قسمت‌های 1 الی 5) در ادامه به ذکر پیاده سازی روش پیشنهادی پرداخته می‌شود.

1- اجرای Integration Service Package جهت اجرای یک Package از ابزارهای زیر می‌توان استفاده کرد:

- command-line ابزار خط فرمان dtexec.exe
- ابزار اجرائی پکیج dtexecui.exe
- استفاده از SQL Server Agent job

توجه: همچنین یک Package را در زمان طراحی در BIDS (Business Intelligence Development Studio) می‌توان اجرا نمود.

2- استفاده از dtexec جهت اجرای Package با استفاده از ابزار dtexec می‌توان Packageهای ذخیره شده در فایل سیستم، یک SQL Instance و یا Packageهای ذخیره شده در Integration Service را اجرا نمود.

توجه: در سیستم عامل‌های 64 بیتی، ابزار dtexec موجود در Integration Service با نسخه 64 بیتی نصب می‌شود. چنانچه بایست Packageهای معینی را در حالت 32 بیتی اجرا کنید، لازم است ابزار dtexec نسخه 32 بیتی نصب شود. ابزار dtexec دستیابی به تمامی ویژگی‌های پیکربندی و اجرای Package از قبیل اتصالات، مشخصات(Properties)، متغیرها، logging و شاخص‌های پردازشی را فراهم می‌کند.
توجه: زمانی که از نسخه‌ی ابزار dtexec که با SQL Server 2008 ارائه شده استفاده می‌کنید برای اجرای یک SSIS Package نسخه 2005، Integration Service به صورت موقت Package را به نسخه 2008 ارتقا می‌دهد، اما نمی‌توان از ابزار dtexec برای ذخیره این تغییرات استفاده کرد.

2-1- ملاحظات نصب dtexec روی سیستم‌های 64 بیتی به صورت پیش فرض، یک سیستم عامل 64 بیتی که هر دو نسخه 64 بیتی و 32 بیتی ابزار خط فرمان Integration Service را دارد، نسخه 32 بیتی نصب شده را در خط فرمان اجرا خواهد کرد. نسخه 32 بیتی بدین دلیل اجرا می‌شود که در متغیر محیطی (Path environment variable) Path مسیر directory نسخه 32 بیتی قرار گرفته است.به طور معمول:

```
(<drive>:\Program Files(x86)\Microsoft SQL Server\100\DTS\Binn)
```

توجه: اگر از SQL Server Agent برای اجرای Package استفاده می‌کنید، SQL Server Agent به طور خودکار از ابزار نسخه 64 بیتی استفاده می‌کند. SQL Server Agent از Registry و نه از متغیر محیطی Path استفاده می‌کند. برای اطمینان از اینکه نسخه 64 بیتی این ابزار را در خط فرمان اجرا می‌کنید، directory را به directory ای تغییر دهید که شامل نسخه 64 بیتی این ابزار است(drive>:\Program Files\Microsoft SQL Server\100\DTS\Binn>) و ابزار را از این مسیر اجرا کنید و یا برای همیشه مسیر قرار گرفته در متغیر محیطی path را با مسیری که نسخه 64 بیتی قرار دارد، جایگزین کنید.

2-2- تفسیر کدهای خروجی هنگامی که یک Package اجرا می‌شود، dtexec یک کد خروجی (Return Code) بر می‌گرداند:

مقدار	توصیف
0	Package با موفقیت اجرا شده است.
1	Package با خطا مواجه شده است.
3	Package در حال اجرا توسط کاربر لغو شده است.
4	Package پیدا نشده است.
5	Package بارگذاری نشده است.
6	ابزار با یک خطای نحوی یا خطای معنایی در خط فرمان برخورد کرده است.

2-3- قوانین نحوی dtexec تمامی گزینه‌ها (Options) باید با یک علامت Slash (/) و یا Minus (-) شروع شوند.
یک آرگومان باید در یک quotation mark محصور شود چنانچه شامل یک فاصله خالی باشد.
گزینه‌ها و آرگومان‌ها بجز رمزعبور حساس به حروف کوچک و بزرگ نیستند.

Syntax 2-3-1-

```
dtexec /option [value] [/option [value]]…
```

Parameters 2-3-2- نکته: در Integration Service، ابزار خط فرمان dtsrun که برایDTS (Data Transformation Service)های نسخه SQL Server 2000 استفاده می‌شد، با ابزار خط فرمان dtexec جایگزین شده است.

- تعدادی از گزینه‌های خط فرمان dtsrun به طور مستقیم در dtexec معادل دارند برای مثال نام Server و نام Package.
- تعدادی از گزینه‌های dtsrun به طور مستقیم در dtexec معادل ندارند.
- تعدادی گزینه‌های خط فرمان جدید dtsexec وجود دارد که در ویژگی‌های جدید Integration Service پشتیبانی می‌شود.

2-3-3- مثال 1) به منظور اجرای یک SSIS Package که در SQL Server ذخیره شده است، با استفاده از Windows Authentication :

```
dtexec /sq <Package Name> /ser <Server Name>
```

2) به منظور اجرای یک SSIS Package که در پوشه File System در SSIS Package Store ذخیره شده است :

```
dtexec /dts “\File System\<Package File Name>”
```

3) به منظور اجرای یک SSIS Package که در سیستم فایل ذخیره شده است و مشخص کردن گزینه logging:

```
dtexec /f “c:\<Package File Name>” /l “DTS.LogProviderTextFile; <Log File Name>”
```

4) به منظور اجرای یک SSIS Package که در SQL Server ذخیره شده با استفاده از SQL Server Authentication برای نمونه(user:ssis;pwd:ssis@ssis)و رمز (123)Package:

```
dtexec  /server “<Server Name>”  /sql “<Package Name>”  / user “ssis” /Password “ssis@ssis” /De “123”
```

3- تنظیمات سطح حفاظتی یک Package به منظور حفاظت از داده‌ها در Packageهای Integration Service می‌توانید یک سطح

حفاظتی (protection level) را تنظیم کنید که به حفاظت از داده‌های صرفاً حساس یا تمامی داده‌های یک Package کمک نماید. به علاوه می‌توانید این داده‌ها را با یک Password یا یک User Key رمزگذاری نمائید یا به رمزگذاری داده‌ها در بانک اطلاعاتی اعتماد کنید. همچنین سطح حفاظتی که برای یک Package استفاده می‌کنید، الزاماً ایستا (static) نیست و در طول چرخه حیات یک Package می‌تواند تغییر کند. اغلب سطح حفاظتی در طول توسعه یا به محض (deploy) استقرار Package تنظیم می‌شود.
توجه: علاوه بر سطوح حفاظتی که توصیف شد، Packageها در بانک اطلاعاتی msdb ذخیره می‌شوند که همچنین می‌توانند توسط نقش‌های ثابت در سطح بانک اطلاعاتی (fixed database-level roles) حفاظت شوند. Integration Service شامل 3 نقش ثابت بانک اطلاعاتی برای نسبت دادن مجوزها به Package است که عبارتند از db_ssisltduser، db_ssisadmin و db_ssisoperator

3-1- درک سطوح حفاظتی در یک Package اطلاعات زیر به عنوان حساس تعریف می‌شوند:
- بخش password در یک connection string. گرچه، اگر گزینه ای را که همه چیز را رمزگذاری کند، انتخاب کنید تمامی connection string حساس در نظر گرفته می‌شود.
- گره‌های task-generated XML که برچسب (tagged) هایی حساس هستند.
- هر متغییری که به عنوان حساس نشان گذاری شود.

3-1-1- Do not save sensitive هنگامی که Package ذخیره می‌شود از ذخیره مقادیر ویژگی‌های حساس در Package جلوگیری می‌کند. این سطح حفاظتی رمزگذاری نمی‌کند اما در عوض از ذخیره شدن ویژگی هایی که حساس نشان گذاری شده اند به همراه Package جلوگیری می‌کند.

3-1-2- Encrypt all with password به منظور رمزگذاری تمامی Package از یک Password استفاده می‌شود. Package توسط Password ای رمزگذاری می‌شود که کاربر هنگامی که Package را ایجاد یا Export می‌کند، ارائه می‌دهد. به منظور باز کردن Package در SSIS Designer یا اجرای Package توسط ابزار خط فرمان dtexec کاربر بایست رمز Package را ارائه نماید. بدون رمز کاربر قادر به دستیابی و اجرای Package نیست.

3-1-3- Encrypt all with user key به منظور رمزگذاری تمامی Package از یک کلید که مبتنی بر Profile کاربر جاری می‌باشد، استفاده می‌شود. تنها کاربری که Package را ایجاد یا Export می‌کند، می‌تواند Package را در SSIS Designer باز کند و یا Package را توسط ابزار خط فرمان dtexec اجرا کند.

3-1-4- Encrypt sensitive with password به منظور رمزگذاری تنها مقادیر ویژگی‌های حساس در Package از یک Password استفاده می‌شود. برای رمزگذاری از DPAPI استفاده می‌شود. داده‌های حساس به عنوان بخشی از Package ذخیره می‌شوند اما آن داده‌ها با استفاده از Password رمزگذاری می‌شوند. به منظور باز نمودن Package در SSIS Designer کاربر باید رمز Package را ارائه دهد. اگر رمز ارائه نشود، Package بدون داده‌های حساس باز می‌شود و کاربر باید مقادیر جدیدی برای داده‌های حساس فراهم کند. اگر کاربر سعی نماید Package را بدون ارائه رمز اجرا کند، اجرای Package با خطا مواجه می‌شود.

3-1-5- Encrypt sensitive with user key به منظور رمزگذاری تنها مقادیر ویژگی‌های حساس در Package از یک کلید که مبتنی بر Profile کاربر جاری می‌باشد، استفاده می‌شود. تنها کاربری که از همان Profile استفاده می‌کند، Package را می‌تواند بارگذاری (load) کند. اگر کاربر متفاوتی Package را باز نماید، اطلاعات حساس با مقادیر پوچی جایگزین می‌شود و کاربر باید مقادیر جدیدی برای داده‌های حساس فراهم کند. اگر کاربر سعی نماید Package را بدون ارائه رمز اجرا کند، اجرای Package با خطا مواجه می‌شود. برای رمزگذاری از DPAPI استفاده می‌شود.

3-1-6- Rely on server storage for encryption (ServerStorage) با استفاده از نقش‌های بانک اطلاعاتی، SQL Server تمامی Package را حفاظت می‌کند. این گزینه تنها زمانی پشتیبانی می‌شود که Package در بانک اطلاعاتی msdb ذخیره شده است.

4- استفاده از نقش‌های Integration Service برای کنترل کردن دستیابی به SSIS، Package شامل 3 نقش ثابت در سطح بانک اطلاعاتی است. نقش‌ها می‌توانند تنها روی Package هایی که در بانک اطلاعاتی msdb ذخیره شده اند، بکار روند. با استفاده از SSMS می‌توانید نقش‌ها را به Packageها نسبت دهید، این انتساب نقش‌ها در بانک اطلاعاتی msdb ذخیره می‌شود.

Role	Read action	Write action
db_ssisadmin **or** **sysadmin**	Enumerate own packages Enumerate all packages View own packages View all packages Execute own packages Execute all packages Export own packages Export all packages Execute all packages in SQL Server Agent	Import packages Delete own packages Delete all packages Change own package roles Change all package roles * به نکته رجوع شود
db_ssisltduser	Enumerate own packages Enumerate all packages View own packages Execute own packages Export own packages	Import packages Delete own packages Change own package roles
db_ssisoperator	Enumerate all packages View all packages Execute all packages Export all packages Execute all packages in SQL Server Agent	None
Windows administrators	View execution details of all running packages	Stop all currently running packages

*** نکته:** اعضای نقش‌های db_ssisadmin و dc_admin ممکن است قادر باشند مجوزهای خودشان را تا سطح sysadmin ارتقا دهند براساس این ترفیع مجوز امکان اصلاح و اجرای Packageها از طریق SQL Server Agent میسر می‌شود. برای محافظت در برابر این ارتقا، با استفاده از یک (account) حساب Proxy با دسترسی محدود، Job هایی که این Packageها را اجرا می‌کنند، پیکربندی شوند یا تنها اعضای نقش sysadmin به نقش‌های db_ssisadmin و dc_admin افزوده شوند.

همچنین جدول sysssispackages در بانک اطلاعاتی msdb شامل Package هایی است که در SQL Server ذخیره می‌شوند. این جدول شامل ستون هایی که اطلاعاتی درباره نقش هایی که به Packageها نسبت داده شده است، می‌باشد.
به صورت پیش فرض، مجوزهای نقش‌های ثابت بانک اطلاعاتی db_ssisadmin و db_ssisoperator و شناسه منحصر به فرد کاربری (unique security identifier) که Package را ایجاد کرده برای خواندن Package بکار می‌رود، و مجوزهای نقش db_ssisadmin و شناسه منحصر به فرد کاربری که Package را ایجاد کرده برای نوشتن Package به کار می‌رود. یک User باید عضو نقش db_ssisadmin و db_ssisltduser یا db_ssisoperator برای داشتن دسترسی خواندن Package باشد. یک User باید عضو نقش db_ssisadmin برای داشتن دسترسی نوشتن Package باشد.

5- اتصال به صورت Remote به Integration Service زمانی که یک کاربر بدون داشتن دسترسی کافی تلاش کند به یک Integration Service به صورت Remote متصل شود، با پیغام خطای "Access is denied" مواجه می‌شود. برای اجتناب از این پیغام خطا می‌توان تضمین کرد که کاربر مجوز مورد نیاز DCOM را دارد.
به منظور پیکربندی کردن دسترسی کاربر به صورت Remote به سرویس Integration مراحل زیر را دنبال کنید:

- Component Service را باز نمایید (در Run عبارت dcomcnfg را تایپ کنید).

- گره Component Service را باز کنید، گره Computer و سپس My Computer را باز نمایید و روی DCOM Config کلیک نمایید.
- گره DCOM Config را باز کنید و از لیست برنامه هایی که می‌توانند پیکربندی شوند MsDtsServer را انتخاب کنید.
- روی Properties برنامه MsDtsServer رفته و قسمت Security را انتخاب کنید.
- در قسمت Lunch and Activation Permissions، مورد Customize را انتخاب و سپس روی Edit کلیک نمایید تا پنجره Lunch Permission باز شود.
- در پنجره Lunch Permission، کاربران را اضافه و یا حذف کنید و مجوزهای مناسب را به کاربران یا گروه‌های مناسب نسبت دهید. مجوزهای موجود عبارتند از Local Lunch، Remote Lunch، Local Activation و Remote Activation .
- در قسمت Access Permission مراحل فوق را به منظور نسبت دادن مجوزهای مناسب به کاربران یا گروه‌های مناسب انجام دهید.
- سرویس Integration را Restart کنید.

مجوز دسترسی Lunch به منظور شروع و خاتمه سرویس، اعطا یا رد می‌شود و مجوز دسترسیActivation به منظور متصل شدن به سرویس، اعطا (grant) یا رد (deny) می‌شود.

6- پیاده سازی در ابتدا به ایجاد یک CLR Stored Procedures پرداخته می‌شود نام اسمبلی ساخته شده به این نام RunningPackage.dll می‌باشد و حاوی کد زیر است:

```
Partial Public Class StoredProcedures
    '-----------------------------------------------
    'exec dbo.Spc_NtDtexec 'Package','ssis','ssis@ssis','1234512345'
    '-----------------------------------------------
    <Microsoft.SqlServer.Server.SqlProcedure()> _
    Public Shared Sub Spc_NtDtexec(ByVal PackageName As String, _
                                   ByVal UserName As String, _
                                   ByVal Password As String, _
                                   ByVal Decrypt As String)
        Dim p As New System.Diagnostics.Process()
        p.StartInfo.FileName = "C:\Program Files\Microsoft SQL Server\100\DTS\Binn\DTExec.exe"
        p.StartInfo.RedirectStandardOutput = True
        p.StartInfo.Arguments = "/sql " & PackageName & " /User " & UserName & " /Password " & Password
& " /De " & Decrypt
        p.StartInfo.UseShellExecute = False
        p.Start()
        p.WaitForExit()
        Dim output As String
        output = p.StandardOutput.ReadToEnd()
        Microsoft.SqlServer.Server.SqlContext.Pipe.Send(output)
    End Sub
End Class
```

در حقیقت توسط این رویه به اجرای برنامه dtexec.exe و ارسال پارامترهای مورد نیاز جهت اجرا پرداخته می‌شود. با توجه به توضیحات تئوری بیان شده، سطح حفاظتی Package ایجاد شده Encrypt all with password توصیه می‌شود که رمز مذکور در قالب یکی از پارامتر ارسالی به رویه ساخته شده موسوم به Spc_NtDtexec ارسال می‌گردد.

در قدم بعدی نیاز به Register کردن dll ساخته شده در سطح بانک اطلاعاتی SQL Server است، این گام‌ها پس از اتصال به SQL Server Management Studio به شرح زیر است:

1- فعال کردن CLR در سرویس SQL Server

```
SP_CONFIGURE 'clr enabled',1
GO
RECONFIGURE
```

2- فعال کردن ویژگی TRUSTWORTHY در بانک اطلاعاتی مورد نظر

```
ALTER DATABASE <Database Name> SET TRUSTWORTHY ON
GO
RECONFIGURE
```

3- ایجاد Assembly و Stored Procedure در بانک اطلاعاتی مورد نظر

Assembly ساخته شده با نام RunningPacakge.dll در ریشه :C کپی شود. بعد از ثبت نمودن این Assembly لزومی به وجود آن نمی‌باشد.

```
USE <Database Name>
GO
CREATE ASSEMBLY [RunningPackage]
AUTHORIZATION [dbo]
FROM 'C:\RunningPackage.dll'
WITH PERMISSION_SET = UNSAFE
Go
CREATE PROCEDURE [dbo].[Spc_NtDtexec]
@PackageName [nvarchar](50),
@UserName [nvarchar](50),
@Password [nvarchar](50),
@Decrypt [nvarchar](50)
WITH EXECUTE AS CALLER
AS
EXTERNAL NAME [RunningPackage].[RunningPackage.StoredProcedures].[Spc_NtDtexec]
GO
```

توجه: Application User برنامه بایست دسترسی اجرای رویه ذخیره شده Spc_NtDtexec را در بانک اطلاعاتی مورد نظر داشته باشد همچنین بایست عضو نقش db_ssisoperator در بانک اطلاعاتی msdb باشد.(منظور از Application User، لاگین است که در Connection string برنامه قرار داده اید.)

در برنامه کاربردی تان کافی است متدی به شکل زیر ایجاد و با توجه به نیازتان در برنامه به فراخوانی آن و اجرای Package بپردازید.

```
Private Sub ExecutePackage()
        Dim oSqlConnection As SqlClient.SqlConnection
        Dim oSqlCommand As SqlClient.SqlCommand
        Dim strCnt As String = String.Empty
        strCnt = "Data Source=" & txtServer.Text & ";User ID=" & txtUsername.Text & ";Password=" &
txtPassword.Text & ";Initial Catalog=" & cmbDatabaseName.SelectedValue.ToString() & ";"
        Try
            oSqlConnection = New SqlClient.SqlConnection(strCnt)
            oSqlCommand = New SqlClient.SqlCommand
            With oSqlCommand
                .Connection = oSqlConnection
                .CommandType = System.Data.CommandType.StoredProcedure
                .CommandText = "dbo.Spc_NtDtexec"
                .Parameters.Clear()
                .Parameters.Add("@PackageName", System.Data.SqlDbType.VarChar, 50)
                .Parameters.Add("@UserName", System.Data.SqlDbType.VarChar, 50)
                .Parameters.Add("@Password", System.Data.SqlDbType.VarChar, 50)
                .Parameters.Add("@Decrypt", System.Data.SqlDbType.VarChar, 50)
                .Parameters("@PackageName").Value = txtPackageName.Text.Trim()
                .Parameters("@UserName").Value = txtUsername.Text.Trim()
                .Parameters("@Password").Value = txtPassword.Text.Trim()
                .Parameters("@Decrypt").Value = txtDecrypt.Text.Trim()
            End With
            If (oSqlCommand.Connection.State <> System.Data.ConnectionState.Open) Then
                oSqlCommand.Connection.Open()
                oSqlCommand.ExecuteNonQuery()
                System.Windows.Forms.MessageBox.Show("Success")
            End If
            If (oSqlCommand.Connection.State = System.Data.ConnectionState.Open) Then
                oSqlCommand.Connection.Close()
            End If
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub 'ExecutePackage
```

عنوان: معرفی OLTP درون حافظه‌ای در SQL Server 2014
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۱ ۰:۵۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, In-Memory OLTP

OLTP درون حافظه‌ای، مهم‌ترین ویژگی جدید SQL Server 2014 است. موتور بانک اطلاعاتی disk based اس کیوال سرور، حدود 15 تا 20 سال قبل تهیه شده‌است و موتور جدید درون حافظه‌ای OLTP آن، بزرگترین بازنویسی این سیستم از زمان ارائه‌ی آن می‌باشد و شروع این پروژه به 5 سال قبل بر می‌گردد. علت تهیه‌ی آن نیز به نیازهای بالای پردازش‌های همزمان مصرف کنندگان این محصول در سال‌های اخیر، نسبت به 15 سال قبل مرتبط است. با استفاده از امکانات OLTP درون حافظه‌ای، امکان داشتن جداول معمولی disk based و جداول جدید memory optimized با هم در یک بانک اطلاعاتی میسر است؛ به همراه مهیا بودن تمام زیرساخت‌هایی مانند تهیه بک آپ، بازیابی آن‌ها، امنیت و غیره برای آن‌ها.

آیا جداول بهینه سازی شده‌ی برای حافظه، همان DBCC PINTABLE منسوخ شده هستند؟

در نگارش‌های قدیمی‌تر اس کیوال سرور، دستوری وجود داشت به نام DBCC PINTABLE که سبب ثابت نگه داشتن صفحات جداول مبتنی بر دیسک یک دیتابیس، در حافظه می‌شد. به این ترتیب تمام خواندن‌های مرتبط با آن جدول، از حافظه صورت می‌گرفت. مشکل این روش که سبب منسوخ شدن آن گردید، اثرات جانبی آن بود؛ مانند خوانده شدن صفحات جدیدتر (با توجه به اینکه ساختار پردازشی و موتور بانک اطلاعاتی تغییری نکرده بود) و نیاز به حافظه‌ی بیشتر تا حدی که کل کش بافر سیستم را پر می‌کرد و امکان انجام سایر امور آن مختل می‌شدند. همچنین اولین ارجاعی به یک جدول، سبب قرار گرفتن کل آن در حافظه می‌گشت. به علاوه ساختار این سیستم نیز همانند روش مبتنی بر دیسک، بر اساس همان روش‌های قفل گذاری، ذخیره سازی اطلاعات و تهیه ایندکس‌های متداول بود.

اما جداول بهینه سازی شده‌ی برای حافظه، از یک موتور کاملا جدید استفاده می‌کنند؛ با ساختار جدیدی برای ذخیره سازی اطلاعات و تهیه ایندکس‌ها. دسترسی به اطلاعات آن‌ها شامل قفل گذاری‌های متداول نیست و در آن حداقل زمان دسترسی به

اطلاعات درنظر گرفته شده‌است. همچنین در آن‌ها data pages یا index pages و کش بافر نیز وجود ندارد.

نحوه‌ی ذخیره سازی و مدیریت اطلاعات جداول بهینه سازی شده برای حافظه

جداول بهینه سازی شده برای حافظه، فرمت ردیف‌های کاملا جدیدی را نیز به همراه دارند و جهت قرارگرفتن در حافظه ودسترسی سریع به آن‌ها بهینه سازی شده‌اند. برخلاف جداول مبتنی بر دیسک سخت که اطلاعات آن‌ها در یک سری صفحات خاص به نام‌های data or index pages ذخیره می‌شوند، اینگونه جداول، دارای ظروف مبتنی بر صفحه نیستند و از مفهوم چند نگارشی برای ذخیره سازی اطلاعات استفاده می‌کنند؛ به این معنا که ردیف‌ها به ازای هر تغییری، دارای یک نگارش جدید خواهند بود و بلافاصله در همان نگارش اصلی به روز رسانی نمی‌شوند.
در اینجا هر ردیف دارای یک timestamp شروع و یک timestamp پایان است. timestamp شروع بیانگر تراکنشی است که ردیف را ثبت کرده و timestamp پایان برای مشخص سازی تراکنشی بکار می‌رود که ردیف را حذف کرده است. اگر timestamp پایان، دارای مقدار بی‌نهایت باشد، به این معنا است که ردیف متناظر با آن هنوز حذف نشده‌است. به روز رسانی یک ردیف در اینجا، ترکیبی است از حذف یک ردیف موجود و ثبت ردیفی جدید. برای یک عملیات فقط خواندنی، تنها نگارش‌هایی که timestamp معتبری داشته باشند، قابل مشاهده خواهند بود و از مابقی صرفنظر می‌گردد.
در OLTP درون حافظه‌ای که از روش چندنگارشی همزمانی استفاده می‌کند، برای یک ردیف مشخص، ممکن است چندین نگارش وجود داشته باشند؛ بسته به تعداد باری که یک رکورد به روز رسانی شده‌است. در اینجا یک سیستم garbage collection همیشه فعال، نگارش‌هایی را که توسط هیچ تراکنشی مورد استفاده قرار نمی‌گیرند، به صورت خودکار حذف می‌کند؛ تا مشکل کمبود حافظه رخ ندهد.

آیا می‌توان به کارآیی جداول بهینه سازی شده برای حافظه با همان روش متداول مبتنی بر دیسک اما با بکارگیری حافظه‌ی بیشتر و استفاده از یک SSD RAID رسید؟
خیر! حتی اگر کل بانک اطلاعاتی مبتنی بر دیسک را در حافظه قرار دهید به کارآیی روش جداول بهینه سازی شده‌ی برای حافظه نخواهید رسید. زیرا در آن هنوز مفاهیمی مانند data pages و index pages به همراه یک buffer pool پیچیده وجود دارند. در روش‌های مبتنی بر دیسک، ردیف‌ها از طریق page id و row offset آن‌ها قابل دسترسی می‌شوند. اما در جداول بهینه سازی شده‌ی برای حافظه، ردیف‌های جداول با یک B-tree خاص به نام Bw-Tree در دسترس هستند.

میزان حافظه‌ی مورد نیاز برای جداول بهینه سازی شده‌ی برای حافظه

باید درنظر داشت که تمام جداول بهینه سازی شده‌ی برای حافظه، به صورت کامل در حافظه ذخیره خواهند شد. بنابراین بدیهی

است که نیاز به مقدار کافی حافظه در اینجا ضروری است. توصیه صورت گرفته، داشتن حافظه‌ای به میزان دو برابر اندازه‌ی اطلاعات است. البته در اینجا چون با یک سیستم هیبرید سر و کار داریم، حافظه‌ی کافی جهت کار buffer pool مختص به جداول مبتنی بر دیسک را نیز باید درنظر داشت.
همچنین اگر به اندازه‌ی کافی حافظه در سیستم تعبیه نشود، شاهد شکست مداوم تراکنش‌ها خواهید بود. به علاوه امکان بازیابی و restore جداول را نیز از دست خواهید داد.
البته لازم به ذکر است که اگر کل بانک اطلاعاتی شما چند ترابایت است، نیازی نیست به همین اندازه یا بیشتر حافظه تهیه کنید. فقط باید به اندازه‌ی جداولی که قرار است جهت قرار گرفتن در حافظه بهینه سازی شوند، حافظه تهیه کنید که حداکثر آن 256 گیگابایت است.

چه برنامه‌هایی بهتر است از امکانات OLTP درون حافظه‌ای SQL Server 2014 استفاده کنند؟

- برنامه‌هایی که در آن‌ها تعداد زیادی تراکنش کوتاه مدت وجود دارد به همراه درجه‌ی بالایی از تراکنش‌های همزمان توسط تعداد زیادی کاربر.
- اطلاعاتی که توسط برنامه زیاد مورد استفاده قرار می‌گیرند را نیز می‌توان در جداول بهینه سازی شده جهت حافظه قرار داد.
- زمانیکه نیاز به اعمال دارای write بسیار سریع و با تعداد زیاد است. چون در جداول بهینه سازی شده‌ی برای حافظه، صفحات داده‌ها و ایندکس‌ها وجود ندارند، نسبت به حالت مبتنی بر دیسک، بسیار سریعتر هستند. در روش‌های متداول، برای نوشتن اطلاعات در یک صفحه، مباحث همزمانی و قفل‌گذاری آن‌را باید در نظر داشت. در صورتیکه در روش بهینه سازی شده‌ی برای حافظه، به صورت پیش فرض از حالتی همانند snapshot isolation و همزمانی مبتنی بر نگارش‌های مختلف رکورد استفاده می‌شود.
- تنظیم و بهینه سازی جداولی با تعداد Read بالا. برای مثال، جداول پایه سیستم که اطلاعات تعاریف محصولات در آن قرار دارند. این نوع جداول عموما با تعداد Readهای بالا و تعداد Write کم شناخته می‌شوند. چون طراحی جداول مبتنی بر حافظه از hash tables و اشاره‌گرهایی برای دسترسی به رکوردهای موجود استفاده می‌کند، اعمال Read آن نیز بسیار سریعتر از حالت معمول هستند.
- مناسب جهت کارهای data warehouse و ETL Staging Table. در جداول مبتنی بر حافظه امکان عدم ذخیره سازی اطلاعات بر روی دیسک سخت نیز پیش بینی شده‌است. در این حالت فقط اطلاعات ساختار جدول، ذخیره‌ی نهایی می‌گردد و اگر سرور نیز ری استارت گردد، مجددا می‌تواند اطلاعات خود را از منابع اصلی data warehouse تامین کند.

محدودیت‌های جداول بهینه سازی شده‌ی برای حافظه در SQL Server 2014

- تغیر اسکیما و ساختار جداول بهینه سازی شده‌ی برای حافظه مجاز نیست. به بیان دیگر دستور ALTER TABLE برای اینگونه جداول کاربردی ندارد. این مورد جهت ایندکس‌ها نیز صادق است. همان زمانیکه جدول ایجاد می‌شود، باید ایندکس آن نیز تعریف گردد و پس از آن این امکان وجود ندارد.
تنها راه تغییر اسکیمای اینگونه جداول، Drop و سپس ایجاد مجدد آن‌ها است.
البته باید درنظر داشت که SQL Server 2014، اولین نگارش این فناوری را ارائه داده‌است و در نگارش‌های بعدی آن، بسیاری از این محدودیت‌ها قرار است که برطرف شوند.
- جداول بهینه سازی شده‌ی برای حافظه حتما باید دارای یک ایندکس باشند. البته اگر یک primary key را برای آن‌ها تعریف نمائید، کفایت می‌کند.
- از unique indexها پشتیبانی نمی‌کند، مگر اینکه از نوع primary key باشد.
- حداکثر 8 ایندکس را می‌توان بر روی اینگونه جداول تعریف کرد.
- امکان تعریف ستون identity در آن وجود ندارد. اما می‌توان از قابلیت sequence برای رسیدن به آن استفاده کرد.
- DML triggers را پشتیبانی نمی‌کند.
- کلیدهای خارجی و قیود را پشتیبانی نمی‌کند.
- حداکثر اندازه‌ی یک ردیف آن 8060 بایت است. بنابراین از نوع‌های داده‌ای max دار و XML پشتیبانی نمی‌کند.
این مورد در حین ایجاد جدول بررسی شده و اگر اندازه‌ی ردیف محاسبه‌ی شده‌ی آن توسط SQL Server 2014 بیش از 8060 بایت باشد، جدول را ایجاد نخواهد کرد.

اگر سرور را ری استارت کنیم، چه اتفاقی برای اطلاعات جداول بهینه سازی شده‌ی برای حافظه رخ می‌دهد؟

حالت DURABILTY انتخاب شده‌ی در حین ایجاد جدول بهینه سازی شده‌ی برای حافظه، تعیین کننده‌ای این مساله است. اگر SCHEMA_ONLY انتخاب شده باشد، کل اطلاعات شما با ری استارت سرور از دست خواهد رفت؛ البته اطلاعات ساختار جدول حفظ خواهد گردید. اگر حالت SCHEMA_AND_DATA انتخاب شود، اطلاعات شما پس از ری‌استارت سرور نیز در دسترس خواهد بود. این اطلاعات به صورت خودکار از لاگ تراکنش‌ها بازیابی شده و مجددا در حافظه قرار می‌گیرند.
حالت SCHEMA_ONLY برای مصارف برنامه‌های data warehouse بیشتر کاربرد دارد. جایی که اطلاعات قرار است از منابع داده‌ی مختلفی تامین شوند.

برای مطالعه بیشتر

SQL Server 2014: NoSQL Speeds with Relational Capabilities
SQL Server 2014 In-Memory OLTP Architecture and Data Storage
Overview of Applications, Indexes and Limitations for SQL Server 2014 In-Memory OLTP Tables
Microsoft SQL Server 2014: In-Memory OLTP Overview
SQL Server in Memory OLTP for Database Developers
Exploring In-memory OLTP Engine (Hekaton) in SQL Server 2014 CTP1

نظرات خوانندگان

نویسنده: فرید طاهری
تاریخ: ۱۳۹۳/۰۳/۱۱ ۱۰:۳

جناب نصیری با تشکر از مقاله مفیدتون لازم می‌دونم در جهت تکمیل مباحث به چند نکته اشاره کنم

1- « *اگر حالت SCHEMA_AND_DATA انتخاب شود، اطلاعات شما پس از ری‌استارت سرور نیز در دسترس خواهد بود. این اطلاعات به صورت خودکار از لاگ تراکنش‌ها بازیابی شده و مجددا در حافظه قرار می‌گیرند* ».

بازیابی اطلاعات مربوط به تراکنش‌هایی که به ازای In Memory OLTP است بوسیله Data File + Delta File و Log File می‌باشد. در صورتیکه Schema_AND_Data را به ازای این نوع جداول فعال کنید داده‌های شما در Data File و داده‌های حذف شده در Delta File ثبت می‌گردد. مکانیزم Log File برای In Memory OLTP همچنان مانند جداول Disk base وجود دارد اما با بهینه سازی مناسب مانند ثبت Log Record کمتر به ازای عملیات کاربران و...

2- در جایی دیگر در متن اشاره شده که In Memory OLTP اجازه استفاده از Identity را به کاربر نمی‌دهد باید اشاره کنم که این موضوع برای نسخه CTP بوده است در نسخه RTM این قابلیت وجود دارد . لازم می‌دانم اشاره کنم که در Books Online جایی گفته شده که امکان استفاده وجود ندارد و در جایی هم گفته شده وجود دارد .

به مثال زیر دقت کنید

```
CREATE TABLE test(
[ID] BIGINT IDENTITY(1,1) NOT NULL PRIMARY KEY NONCLUSTERED HASH WITH (BUCKET_COUNT=10000),
N1 NVARCHAR(100),
N2 NVARCHAR(100),
N3 NVARCHAR(100)
) WITH (MEMORY_OPTIMIZED=ON,DURABILITY = SCHEMA_AND_DATA)
GO
```

این مثال در SQL Server 2014 RTM Edition قابل اجرا است اما یکسری محدودیت داریم . مثلاً مقدار شروع و گام افزایش باید 1 باشد .

باز هم از مطالب خوب شما متشکرم مقاله مفیدی بود .

عنوان: ایجاد جداول بهینه سازی شده برای حافظه در SQL Server 2014
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۱ ۲۱:۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, In-Memory OLTP

پس از نگاهی به مفاهیم مقدماتی OLTP درون حافظه‌ای در SQL Server 2014 ، در ادامه به نحوه‌ی انجام تنظیمات خاص جداول بهینه سازی شده برای حافظه خواهیم پرداخت.

ایجاد یک بانک اطلاعاتی با پشتیبانی از جداول بهینه سازی شده برای حافظه

برای ایجاد جداول بهینه سازی شده برای حافظه، ابتدا نیاز است تا تنظیمات خاصی را به بانک اطلاعاتی آن اعمال کنیم. برای اینکار می‌توان یک بانک اطلاعاتی جدید را به همراه یک filestream filegroup ایجاد کرد که جهت جداول بهینه سازی شده برای حافظه، ضروری است؛ یا اینکه با تغییر یک بانک اطلاعاتی موجود و افزودن filegroup یاد شده نیز می‌توان به این مقصود رسید.
در اینگونه جداول خاص، اطلاعات در حافظه‌ی سیستم ذخیره می‌شوند و برخلاف جداول مبتنی بر دیسک سخت، صفحات اطلاعات وجود نداشته و نیازی نیست تا به کش بافر وارد شوند. برای مقاصد ذخیره سازی نهایی اطلاعات جداول بهینه سازی شده برای حافظه، موتور OLTP درون حافظه‌ای آن، فایل‌های خاصی را به نام checkpoint در یک filestream filegroup ایجاد می‌کند که از آن‌ها جهت ردیابی اطلاعات استفاده خواهد کرد و نحوی ذخیره سازی اطلاعات در آن‌ها از شیوه‌ی با کارآیی بالایی به نام append only mode پیروی می‌کند.
با توجه به متفاوت بودن نحوه‌ی ذخیره سازی نهایی اطلاعات اینگونه جداول و دسترسی به آن‌ها از طریق استریم‌ها، توصیه شده‌است که filestream filegroupهای تهیه شده را در یک SSD یا Solid State Drive قرار دهید.

پس از اینکه بانک اطلاعاتی خود را به روش‌های معمول ایجاد کردید، به برگه‌ی خواص آن در management studio مراجعه کنید. سپس صفحه‌ی file groups را در آن انتخاب کرده و در پایین برگه‌ی آن، در قسمت جدید memory optimized data، بر روی دکمه‌ی Add کلیک کنید. سپس نام دلخواهی را وارد نمائید.

پس از ایجاد یک گروه فایل جدید، به صفحه‌ی files خواص بانک اطلاعاتی مراجعه کرده و بر روی دکمه‌ی Add کلیک کنید. سپس File type این ردیف اضافه شده را از نوع file stream data و file group آن‌را همان گروه فایلی که پیشتر ایجاد کردیم، تنظیم کنید. در ادامه logical name دلخواهی را وارد کرده و در آخر بر روی دکمه‌ی Ok کلیک کنید تا تنظیمات مورد نیاز جهت تعاریف جدول بهینه سازی شده برای حافظه به پایان برسد.

این مراحل را توسط دو دستور T-SQL ذیل نیز می‌توان سریعتر انجام داد:

```
USE [master]
GO
ALTER DATABASE [testdb2]
      ADD FILEGROUP [InMemory_InMemory] CONTAINS MEMORY_OPTIMIZED_DATA
GO
ALTER DATABASE [testdb2]
      ADD FILE ( NAME = N'InMemory_InMemory', FILENAME =
N'D:\SQL_Data\MSSQL11.MSSQLSERVER\MSSQL\DATA\InMemory_InMemory' )
      TO FILEGROUP [InMemory_InMemory]
GO
```

ساختار گروه فایل بهینه سازی شده برای حافظه

گروه فایل بهینه سازی شده برای حافظه، دارای چندین دربرگیرنده است که هر کدام چندین فایل را در خود جای خواهند داد:
- Root File که در برگیرنده‌ی متادیتای اطلاعات است.
- Data File که شامل ردیف‌های اطلاعات ثبت شده در جداول بهینه سازی شده‌ی برای حافظه هستند. این ردیف‌ها همواره به انتهای data file اضافه می‌شوند و دسترسی به آن‌ها ترتیبی است. کارآیی IO این روش نسبت به روش دسترسی اتفاقی به مراتب بالاتر است. حداکثر اندازه این فایل 128 مگابایت است و پس از آن یک فایل جدید ساخته می‌شود.
- Delta File شامل ردیف‌هایی است که حذف شده‌اند. به ازای هر ردیف، حداقل اطلاعاتی از آن را در خود ذخیره خواهد کرد؛ شامل ID ردیف حذف شده و شماره تراکنش آن. همانطور که پیشتر نیز ذکر شد، این موتور جدید درون حافظه‌ای، برای یافتن راه چاره‌ای جهت به حداقل رسانی قفل گذاری بر روی اطلاعات، چندین نگارش از ردیف‌ها را به همراه timestamp آن‌ها در خود ذخیره می‌کند. به این ترتیب، هر به روز رسانی به همراه یک حذف و سپس ثبت جدید است. به این ترتیب دیگر بانک اطلاعاتی نیازی نخواهد داشت تا به دنبال رکورد موجود برگردد و سپس اطلاعات آن‌را به روز نماید. این موتور جدید فقط اطلاعات به روز شده را در انتهای رکوردهای موجود با فرمت خود ثبت می‌کند.

ایجاد جداول بهینه سازی شده برای حافظه

پس از آماده سازی بانک اطلاعاتی خود و افزودن گروه فایل استریم جدیدی به آن برای ذخیره سازی اطلاعات جداول بهینه سازی شده برای حافظه، اکنون می‌توانیم اینگونه جداول خاص را در کنار سایر جداول متداول موجود، تعریف و استفاده نمائیم:

```
-- It is not a Memory Optimized
CREATE TABLE tblNormal
(
   [CustomerID] int NOT NULL PRIMARY KEY NONCLUSTERED,
```

```
   [Name] nvarchar(250) NOT NULL,
   CustomerSince DATETIME not NULL
      INDEX [ICustomerSince] NONCLUSTERED
)

--  DURABILITY = SCHEMA_AND_DATA
CREATE TABLE tblMemoryOptimized_Schema_And_Data
(
    [CustomerID] INT NOT NULL
PRIMARY KEY NONCLUSTERED HASH WITH (BUCKET_COUNT = 1000000),
    [Name] NVARCHAR(250) NOT NULL,
    [CustomerSince] DATETIME NOT NULL
INDEX [ICustomerSince] NONCLUSTERED
) WITH (MEMORY_OPTIMIZED = ON, DURABILITY = SCHEMA_AND_DATA)


-- DURABILITY = SCHEMA_ONLY
CREATE TABLE tblMemoryOptimized_Schema_Only
(
    [CustomerID] INT NOT NULL
PRIMARY KEY NONCLUSTERED HASH WITH (BUCKET_COUNT = 1000000),
    [Name] NVARCHAR(250) NOT NULL,
    [CustomerSince] DATETIME NOT NULL
INDEX [ICustomerSince] NONCLUSTERED
) WITH (MEMORY_OPTIMIZED = ON, DURABILITY = SCHEMA_ONLY)
```

در اینجا سه جدول را مشاهده می‌کنید که در بانک اطلاعاتی آماده شده در مرحله‌ی قبل، ایجاد خواهند شد. مورد اول یک جدول معمولی است که از آن برای مقایسه سرعت ثبت اطلاعات با سایر جداول ایجاد شده، استفاده خواهد شد.

همانطور که مشخص است، دو جدول بهینه سازی شده برای حافظه، همان سه ستون جدول معمولی مبتنی بر دیسک سخت را دارا هستند؛ اما با این تفاوت‌ها:

- دارای ویژگی **MEMORY_OPTIMIZED = ON** می‌باشند. به این ترتیب اینگونه جداول نسبت به جداول متداول مبتنی به دیسک سخت متمایز خواهند شد.

- دارای ویژگی **DURABILITY** بوده و توسط مقدار SCHEMA_AND_DATA آن مشخص می‌کنیم که آیا قرار است اطلاعات و ساختار جدول، ذخیره شوند یا تنها قرار است ساختار جدول ذخیره گردد (حالت SCHEMA_ONLY).

- بر روی ستون Id آن‌ها یک hash index ایجاد شده‌است که وجود آن ضروری است و در کل بیش از 8 ایندکس را نمی‌توان تعریف کرد.

برخلاف ایندکس‌های B-tree جداول مبتنی بر سخت دیسک، ایندکس‌های جداول بهینه سازی شده برای حافظه، اطلاعات را تکرار نمی‌کنند. این‌ها صرفا اشاره‌گرهایی هستند به ردیف‌های اصلی اطلاعات. به این معنا که این ایندکس‌ها لاگ نشده و همچنین بر روی سخت دیسک ذخیره نمی‌شوند. کار بازسازی مجدد آن‌ها در اولین بار بازیابی بانک اطلاعاتی و آغاز آن به صورت خودکار انجام می‌شود. به همین جهت مباحثی مانند index fragmentation و نگهداری ایندکس‌ها دیگر در اینجا معنا پیدا نمی‌کنند.

دو نوع ایندکس را در اینجا می‌توان تعریف کرد. اولین آن‌ها hash index است و دومین آن‌ها range index. هش ایندکس‌ها برای حالاتی که در کوئری‌ها از عملگر تساوی استفاده می‌شود بسیار مناسب هستند. برای عملگرهای مقایسه‌ای از ایندکس‌های بازه‌ای استفاده می‌شود.

همچنین باید دقت داشت که پس از ایجاد ایندکس‌ها، دیگر امکان تغییر آن‌ها و یا تغییر ساختار جدول ایجاد شده نیست.

همچنین ایندکس‌های تعریف شده در جداول بهینه سازی شده برای حافظه، تنها بر روی ستون‌هایی غیرنال پذیر از نوع BIN2 collation مانند int و datetime قابل تعریف هستند. برای مثال اگر سعی کنیم بر روی ستون Name ایندکسی را تعریف کنیم، به این خطا خواهیم رسید:

```
Indexes on character columns that do not use a *_BIN2 collation are not supported with indexes on
memory optimized tables.
```

- در حین تعریف هش ایندکس‌ها، مقدار BUCKET_COUNT نیز باید تنظیم شود. هر bucket توسط مقداری که حاصل هش کردن یک ستون است مشخص می‌شود. کلیدهای منحصربفرد دارای هش‌های یکسان در bucketهای یکسانی ذخیره می‌شوند. به همین جهت توصیه شده‌است که حداقل مقدار bucket تعیین شده در اینجا مساوی یا بیشتر از مقدار تعداد کلیدهای منحصربفرد یک جدول باشد؛ مقدار پیش فرض 2 برابر توسط مایکروسافت توصیه شده‌است.

- نوع‌های قابل تعریف ستون‌ها نیز در اینجا به موارد ذیل محدود هستند و جمع طول آن‌ها از 8060 نباید بیشتر شود:

```
bit, tinyint, smallint, int, bigint, money, smallmoney, float, real, datetime, smalldatetime,
datetime2,
date, time, numberic, decimal, char(n),  varchar(n) ,nchar(n),  nvarchar(n), sysname, binary(n),
varbinary(n), and Uniqueidentifier
```

همچنین در management studio، گزینه‌ی جدید new -> memory optimized table نیز اضافه شده‌است و انتخاب آن سبب می‌شود تا قالب T-SQL ایی برای تهیه این نوع جداول، به صورت خودکار تولید گردد.

البته این گزینه تنها برای بانک‌های اطلاعاتی که دارای گروه فایل استریم مخصوص جداول بهینه سازی شده برای حافظه هستند،

فعال می‌باشد.

ثبت اطلاعات در جداول معمولی و بهینه سازی شده برای حافظه و مقایسه کارآیی آن‌ها

در مثال زیر، 100 هزار رکورد را در سه جدولی که پیشتر ایجاد کردیم، ثبت کرده و سپس مدت زمان اجرای هر کدام از مجموعه عملیات را بر حسب میلی ثانیه بررسی می‌کنیم:

```
set statistics time off
SET STATISTICS IO Off
set nocount on
go
-----------------------------

Print 'insert into tblNormal'

DECLARE @start datetime = getdate()
declare @insertCount int = 100000
declare @startId int = 1
declare @customerID int = @startId

while @customerID < @startId + @insertCount
begin
    insert into tblNormal values (@customerID, 'Test', '2013-01-01T00:00:00')
    set @customerID +=1
end

Print DATEDIFF(ms,@start,getdate());
go
-----------------------------

Print 'insert into tblMemoryOptimized_Schema_And_Data'

DECLARE @start datetime = getdate()
declare @insertCount int = 100000
declare @startId int = 1
declare @customerID int = @startId

while @customerID < @startId + @insertCount
begin
    insert into tblMemoryOptimized_Schema_And_Data values (@customerID, 'Test', '2013-01-01T00:00:00')
    set @customerID +=1
end
Print DATEDIFF(ms,@start,getdate());
Go
-----------------------------

Print 'insert into tblMemoryOptimized_Schema_Only'

DECLARE @start datetime = getdate()
declare @insertCount int = 100000
declare @startId int = 1
declare @customerID int = @startId

while @customerID < @startId + @insertCount
begin
    insert into tblMemoryOptimized_Schema_Only values (@customerID, 'Test', '2013-01-01T00:00:00')
    set @customerID +=1
end
Print DATEDIFF(ms,@start,getdate());

Go
```

با این خروجی تقریبی که بر اساس توانمندی‌های سخت افزاری سیستم می‌تواند متفاوت باشد:

```
insert into tblNormal
36423

insert into tblMemoryOptimized_Schema_And_Data
30516

insert into tblMemoryOptimized_Schema_Only
3176
```

و برای حالت select خواهیم داشت:

```
set nocount on
print 'tblNormal'
set statistics time on
select count(CustomerID) from tblNormal
set statistics time off
go
print 'tblMemoryOptimized_Schema_And_Data'
set statistics time on
select count(CustomerID) from tblMemoryOptimized_Schema_And_Data
set statistics time off
go
print 'tblMemoryOptimized_Schema_Only'
set statistics time on
select count(CustomerID) from tblMemoryOptimized_Schema_Only
set statistics time off
go
```

با این خروجی

```
tblNormal
 SQL Server Execution Times:
CPU time = 46 ms,  elapsed time = 52 ms.

tblMemoryOptimized_Schema_And_Data
 SQL Server Execution Times:
CPU time = 32 ms,  elapsed time = 33 ms.

tblMemoryOptimized_Schema_Only
 SQL Server Execution Times:
CPU time = 31 ms,  elapsed time = 30 ms.
```

تاثیر جداول بهینه سازی شده برای حافظه را در 350K inserts بهتر می‌توان با نمونه‌های متداول مبتنی بر دیسک مقایسه کرد.

برای مطالعه بیشتر

Getting started with SQL Server 2014 In-Memory OLTP

Introduction to SQL Server 2014 CTP1 Memory-Optimized Tables

Overcoming storage speed limitations with Memory-Optimized Tables for SQL Server

Memory-optimized Table – Day 1 Test

Memory-Optimized Tables – Insert Test

Memory Optimized Table – Insert Test ...Again

نظرات خوانندگان

نویسنده: علی رضایی
تاریخ: ۱۳۹۳/۰۳/۱۲ ۱۹:۲۲

با سلام و تشکر فراوان جهت این آموزش.
میشه لطفاً بررسی کنید چرا نتیجه نهایی برای من متفاوت شده، طوری که زمان جستجو برای جدول نرمال کمتره!
راستی زمان ورود اطلاعات (100 رکورد) شبیه به زمان‌های مثال شما است و من از یک هارد SSD Corsair استفاده میکنم.

```
set nocount on
print 'tblNormal'
set statistics time on
select count(CustomerID) from tblNormal
set statistics time off
go
print 'tblMemoryOptimized_Schema_And_Data'
set statistics time on
select count(CustomerID) from tblMemoryOptimized_Schema_And_Data
set statistics time off
go
print 'tblMemoryOptimized_Schema_Only'
set statistics time on
select count(CustomerID) from tblMemoryOptimized_Schema_Only
set statistics time off
go
```

100 %

Results | Messages

```
tblNormal

 SQL Server Execution Times:
   CPU time = 16 ms,  elapsed time = 6 ms.
tblMemoryOptimized_Schema_And_Data

 SQL Server Execution Times:
   CPU time = 15 ms,  elapsed time = 18 ms.
tblMemoryOptimized_Schema_Only

 SQL Server Execution Times:
   CPU time = 16 ms,  elapsed time = 17 ms.
```

ویرایش:
تست جدید با بیش از 11 میلیون رکورد و خاموش کردن فایروال کومودو و خاموش کردن windows defender

```
set nocount on
print 'tblNormal'
set statistics time on
select count(CustomerID) from tblNormal
set statistics time off
go
print 'tblMemoryOptimized_Schema_And_Data'
set statistics time on
select count(CustomerID) from tblMemoryOptimized_Schema_And_Data
set statistics time off
go
print 'tblMemoryOptimized_Schema_Only'
set statistics time on
select count(CustomerID) from tblMemoryOptimized_Schema_Only
set statistics time off
go
```

100 %

Results | Messages

```
tblNormal
 SQL Server Execution Times:
   CPU time = 1577 ms,  elapsed time = 406 ms.

tblMemoryOptimized_Schema_And_Data
 SQL Server Execution Times:
   CPU time = 2032 ms,  elapsed time = 2040 ms.

tblMemoryOptimized_Schema_Only
 SQL Server Execution Times:
   CPU time = 1968 ms,  elapsed time = 1961 ms.
```

	(No column name)
1	11072240

	(No column name)
1	11000000

	(No column name)
1	11000000

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۳ ۰:۳۷

- بستگی دارد. چقدر سیستم شما RAM دارد. مشخصات CPU آن چیست و خیلی از مسایل جانبی دیگر (مانند تحت نظر بودن پوشه‌های فایل استریم‌ها توسط آنتی ویروس یا خیر).
- هدف اصلی از این تحولات، کارهای همزمان و بررسی تفاوت بهبود در کارهای چند ریسمانی و چند کاربری است. مثال فوق یک مثال تک ریسمانی است.
- یک آزمایش را باید چندبار تکرار کرد و بعد میانگین گرفت. برای تکرار هم نیاز است کش‌های سیستم را هربار حذف کنید:

```
set nocount on
CHECKPOINT
DBCC FREEPROCCACHE()
DBCC DROPCLEANBUFFERS
```

- بعد از پایان Insert تعداد ردیف‌های زیاد در یک جدول درون حافظه‌ای باید اطلاعات آماری آن‌را دستی به روز کرد (^).

```
UPDATE STATISTICS tblNormal WITH FULLSCAN, NORECOMPUTE;
UPDATE STATISTICS tblMemoryOptimized_Schema_And_Data WITH FULLSCAN, NORECOMPUTE;
UPDATE STATISTICS tblMemoryOptimized_Schema_Only WITH FULLSCAN, NORECOMPUTE;
```

این دستورات باید قبل از اجرای سه کوئری آخر قرار گیرند.

- تعداد bucket count هم مهم است. در اینجا فرمولی برای محاسبه آن ارائه شده:

```
Select  POWER(
    2,
    CEILING( LOG( COUNT( 0)) / LOG( 2)))
    AS 'BUCKET_COUNT'
FROM
  (SELECT DISTINCT CustomerId
      FROM tblMemoryOptimized_Schema_And_Data) T
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۴ ۰:۵۹

ارزش واقعی جداول درون حافظه‌ای را باید با اعمال تراکنش‌های همزمان و بررسی میزان پاسخگویی سیستم بررسی کرد و نه صرفا با یک آزمایش ساده تک ریسمانی. برای این منظور برنامه‌ای به نام ostress.exe توسط مایکروسافت تهیه شده‌است که امکان انجام یک چنین آزمایشاتی را میسر می‌کند. برای دریافت آن به آدرس‌های ذیل مراجعه کنید:

RML Utilities X64

RML Utilities X86

که نهایتا در این مسیر C:\Program Files\Microsoft Corporation\RMLUtils نصب خواهد شد.

سپس در خط فرمان این سه دستور را امتحان کنید:

```
-- Insert 10000 records using 20 threads, Repeat Execution 3 times

-- disk-based
"C:\Program Files\Microsoft Corporation\RMLUtils\ostress.exe" -n20 -r3 -S. -E -dTestdb2 -q -Q"set
statistics time off; SET STATISTICS IO Off; set nocount on; DECLARE @start datetime = getdate();
declare @insertCount int = 10000; declare @startId int = 1; while @startId < @insertCount begin insert
into tblNormal values ('Test', '2013-01-01T00:00:00') set @startId +=1 end; Print
DATEDIFF(ms,@start,getdate());" -oc:\temp\output

-- memory-optimized, tblMemoryOptimized_Schema_And_Data
"C:\Program Files\Microsoft Corporation\RMLUtils\ostress.exe" -n20 -r3 -S. -E -dTestdb2 -q -Q"set
statistics time off; SET STATISTICS IO Off; set nocount on; DECLARE @start datetime = getdate();
declare @insertCount int = 10000; declare @startId int = 1; while @startId < @insertCount begin insert
into tblMemoryOptimized_Schema_And_Data values ('Test', '2013-01-01T00:00:00') set @startId +=1 end;
Print DATEDIFF(ms,@start,getdate());" -oc:\temp\output

-- memory-optimized, tblMemoryOptimized_Schema_Only
"C:\Program Files\Microsoft Corporation\RMLUtils\ostress.exe" -n20 -r3 -S. -E -dTestdb2 -q -Q"set
statistics time off; SET STATISTICS IO Off; set nocount on; DECLARE @start datetime = getdate();
declare @insertCount int = 10000; declare @startId int = 1;  while @startId <  @insertCount begin
insert into tblMemoryOptimized_Schema_Only values ('Test', '2013-01-01T00:00:00') set @startId +=1 end;
Print DATEDIFF(ms,@start,getdate());" -oc:\temp\output
```

زمانیکه را که در پایان کار نمایش می‌دهد، مبنای واقعی مقایسه است.

البته برای اجرای این دستورات نیاز است فیلد CustomerID را identity تعریف کنید (در هر سه جدول مطلب جاری).

```
-- It is not Memory Optimized
CREATE TABLE tblNormal
(
       [CustomerID] int identity NOT NULL PRIMARY KEY NONCLUSTERED,
       [Name] nvarchar(250) NOT NULL,
   CustomerSince DATETIME not NULL
      INDEX [ICustomerSince] NONCLUSTERED
)

--  DURABILITY = SCHEMA_AND_DATA
CREATE TABLE tblMemoryOptimized_Schema_And_Data
(
    [CustomerID] INT  identity NOT NULL
PRIMARY KEY NONCLUSTERED HASH WITH (BUCKET_COUNT = 131072),
    [Name] NVARCHAR(250) NOT NULL,
    [CustomerSince] DATETIME NOT NULL
INDEX [ICustomerSince] NONCLUSTERED
) WITH (MEMORY_OPTIMIZED = ON, DURABILITY = SCHEMA_AND_DATA)


-- DURABILITY = SCHEMA_ONLY
CREATE TABLE tblMemoryOptimized_Schema_Only
(
    [CustomerID] INT  identity NOT NULL
PRIMARY KEY NONCLUSTERED HASH WITH (BUCKET_COUNT = 131072),
    [Name] NVARCHAR(250) NOT NULL,
    [CustomerSince] DATETIME NOT NULL
INDEX [ICustomerSince] NONCLUSTERED
) WITH (MEMORY_OPTIMIZED = ON, DURABILITY = SCHEMA_ONLY)
```

عنوان: **ابزارهای مهاجرت به OLTP درون حافظه‌ای در SQL Server 2014**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۲ ۲۱:۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, In-Memory OLTP

در SQL Server 2014، به Management studio آن ابزارهای جدیدی اضافه شده‌اند تا کار تبدیل و مهاجرت جداول معمولی، به جداول بهینه سازی شده‌ی برای حافظه را ساده‌تر کنند. برای مثال امکان جدیدی به نام Transaction performance collector جهت بررسی کارآیی تراکنش‌های جداول و یا رویه‌های ذخیره شده در محیط کاری جاری، طراحی شده‌است. پس از آن، این اطلاعات را آنالیز کرده و بر اساس میزان استفاده از آن‌ها، توصیه‌هایی را در مورد مهاجرت یا عدم نیاز به مهاجرت به سیستم جدید OLTP درون حافظه‌ای ارائه می‌دهد. در ادامه این ابزارهای جدید را بررسی خواهیم کرد.

ابزار Memory Optimization Advisor

Memory Optimization Advisor یک Wizard مانند است که از آن برای گرفتن مشاوره در مورد تبدیل جداول موجود مبتنی بر دیسک سخت، به نمونه‌های بهینه سازی شده برای حافظه می‌توان استفاده کرد. کار آن بررسی ساختار جداولی است که قصد مهاجرت آن‌ها را دارید. برای مثال همانطور که پیشتر نیز عنوان شد ، جداول بهینه سازی شده برای حافظه محدودیت‌هایی دارند؛ مثلا نباید کلید خارجی داشته باشند. این Wizard یک چنین مواردی را آنالیز کرده و گزارشی را ارائه می‌دهد. پس از اینکه مراحل آن‌را به پایان رساندید و مشکلاتی را که گزارش می‌دهد، برطرف نمودید، کد تبدیل جدول را نیز به صورت خودکار تولید می‌کند.
برای دسترسی به آن، فقط کافی است بر روی نام جدول خود کلیک راست کرده و گزینه‌ی memory optimization advisor را انتخاب کنید.

در دو قسمت اول این Wizard، کار بررسی ساختار جدول در حال مهاجرت صورت می‌گیرد. اگر نوع داده‌ای در آن پشتیبانی نشود یا قیود ویژه‌ای در آن تعریف شده باشند، گزارشی را جهت رفع، دریافت خواهید کرد. پس از رفع آن، به صفحه‌ی گزینه‌های مهاجرت می‌رسیم:

همانطور که ملاحظه می‌کنید، گروه فایل ایجاد شده در قسمت قبل ، به صورت خودکار انتخاب شده‌است.
در ادامه می‌توان نام دیگری را برای جدول مبتنی بر دیسک وارد کرد. در اینجا به صورت خودکار کلمه‌ی old به آخر نام جدول اضافه شده‌است. در حین تولید جدول جدید بهینه سازی شده‌ی بر اساس ساختار جدول فعلی، این جدول قدیمی به صورت خودکار تغییر نام خواهد یافت و کلیه اطلاعات آن حفظ می‌شود.
همچنین تخمینی را نیز از مقدار حافظه‌ی مورد نیاز برای نگهداری این جدول جدید درون حافظه‌ای نیز ارائه می‌دهد. در این مثال چون رکوردی در جدول انتخابی وجود نداشته‌است، تخمین آن صفر است. عدد ارائه شده توسط آن بسیار مهم است و باید به همین میزان برای سیستم خود حافظه تهیه نمائید و یا از حافظه‌ی موجود استفاده کنید.
در پایین صفحه می‌توان انتخاب کرد که آیا داده‌های جدول فعلی، به جدول درون حافظه‌ای انتقال یابند یا خیر. به علاوه نوع ماندگاری اطلاعات آن نیز قابل تنظیم است. اگر گزینه‌ی آخر را انتخاب کنید به معنای حالت SCHEMA_ONLY است. حالت پیش فرض آن SCHEMA_AND_DATA می‌باشد که در قسمت‌های قبل بیشتر در مورد آن بحث شد.

در دو صفحه‌ی بعد، کار انتخاب hash index و range index انجام می‌شود:

در اینجا hash index بر روی فیلد ID تولید شده‌است، به همراه تعیین bucket count آن و در صفحه‌ی بعدی range index بر روی فیلد تاریخ تعریف گردیده‌است:

در آخر می‌توان با کلیک بر روی دکمه‌ی Script، صرفا دستورات T-SQL تغییر ساختار جدول را دریافت کرد و یا با کلیک بر روی دکمه‌ی migrate به صورت خودکار کلیه موارد تنظیم شده را اجرا نمود.

خلاصه‌ی این مراحل که توسط دکمه‌ی Script آن تولید می‌شود، به صورت زیر است:

```
USE [testdb2]
GO

EXEC dbo.sp_rename @objname = N'[dbo].[tblNormal]', @newname = N'tblNormal_old', @objtype = N'OBJECT'
GO

USE [testdb2]
GO

SET ANSI_NULLS ON
GO

CREATE TABLE [dbo].[tblNormal]
(
[CustomerID] [int] NOT NULL,
[Name] [nvarchar](250) COLLATE Persian_100_CI_AI NOT NULL,
[CustomerSince] [datetime] NOT NULL,

INDEX [ICustomerSince] NONCLUSTERED
(
[CustomerSince] ASC
),
CONSTRAINT [tblNormal_primaryKey] PRIMARY KEY NONCLUSTERED HASH
(
[CustomerID]
)WITH ( BUCKET_COUNT = 131072)
```

```
)WITH ( MEMORY_OPTIMIZED = ON , DURABILITY = SCHEMA_AND_DATA )

GO

INSERT INTO [testdb2].[dbo].[tblNormal] ([CustomerID], [Name], [CustomerSince]) SELECT [CustomerID],
[Name], [CustomerSince] FROM [testdb2].[dbo].[tblNormal_old]

GO
```

که در آن ابتدا کار تغییر نام جدول قبلی صورت می‌گیرد. سپس یک جدول جدید با ویژگی MEMORY_OPTIMIZED = ON را ایجاد می‌کند. در ساختار این جدول، hash index و range index تعریف شده، قابل مشاهده هستند. در آخر نیز کلیه اطلاعات جدول قدیمی را به جدول جدید منتقل می‌کند.

علاوه بر memory optimization advisor مخصوص جداول، ابزار دیگری نیز به نام **Native compilation advisor** برای آنالیز رویه‌های ذخیره شده تهیه شده‌است:

آیا سیستم فعلی ما واقعا نیازی به ارتقاء به جداول درون حافظه‌ای دارد؟

تا اینجا در مورد نحوه‌ی ایجاد جداول درون حافظه‌ای و یا نحوه‌ی تبدیل جداول موجود را به ساختار جدید بررسی کردیم. ولی آیا واقعا یک چنین تغییراتی برای ما سودمند هستند؟ برای پاسخ دادن به این سؤال ابزاری به نام AMR به management studio 2014 اضافه شده‌است (Analyze, Migrate, Report). کار آن تحت نظر قرار دادن جداول و رویه‌های ذخیره شده‌ی بانک اطلاعاتی است و سپس بر اساس بار سیستم، تعداد درخواست‌های همزمان و میزان استفاده از جداول و تراکنش‌های مرتبط با آن‌ها، گزارشی را ارائه می‌دهد. بر این اساس بهتر می‌توان تصمیم گرفت که کدام جداول بهتر است به جداول درون حافظه‌ای تبدیل شوند.
برای تنظیم آن باید مراحل ذیل طی شوند:
در Management Studio، به برگه‌ی Object Explorer آن مراجعه کنید. سپس پوشه‌ی Management آن‌را یافته و بر روی گزینه‌ی Data Collection کلیک راست نمائید:

در اینجا گزینه‌ی Configure Management Data Warehouse را انتخاب نمائید. در صفحه‌ی باز شده، ابتدا بانک اطلاعاتی مدنظر را انتخاب نمائید. همچنین بهتر است بر روی دکمه‌ی new کلیک کرده و یک بانک اطلاعاتی جدید را برای آن ایجاد نمائید، تا دچار تداخل اطلاعاتی و ساختاری نگردد:

در ادامه نام کاربری را که قرار است کار مدیریت ثبت و جمع آوری اطلاعات را انجام دهد، به همراه نقش‌های آن انتخاب نمائید:

و در آخر در صفحه‌ی بعدی بر روی دکمه‌ی Finish کلیک کنید.

پس از ایجاد و انتخاب بانک اطلاعاتی Management Data Warehouse، نوبت به تنظیم گزینه‌های جمع آوری اطلاعات است:

در اینجا ابتدا سرور جاری را انتخاب کنید. پس از آن به صورت خودکار در لیست بانک‌های اطلاعاتی قابل انتخاب، تنها همان بانک

اطلاعاتی جدیدی را که برای مرحله‌ی قبل ایجاد کردیم، می‌توان مشاهده کرد.

در صفحه‌ی بعد، گزینه‌ی «Transaction Performance Collection Sets» را انتخاب نمائید که دقیقا گزینه‌ی مدنظر ما جهت یافتن آماری از وضعیت تراکنش‌های سیستم است.
در ادامه بر روی گزینه‌های next و finish کلیک کنید تا کار تنظیمات به پایان برسد.

اکنون اگر به لیست وظایف تعریف شده در SQL Server agent مراجعه کنید، می‌توانید، وظایف مرتبط با جمع آوری داده‌ها را نیز مشاهده نمائید:

وظایف Stored Procedure Usage Analysis هر نیم ساعت یکبار و وظایف Table Usage Analysis هر 15 دقیقه یکبار اجرا می‌شوند. البته امکان اجرای دستی این وظایف نیز مانند سایر وظایف SQL Server وجود دارند.

همچنین در پوشه‌ی management، گزینه‌ی Data collection نیز دو زیر شاخه اضافه شده‌اند که نمایانگر آنالیز میزان مصرف جداول و رویه‌های ذخیره شده می‌باشند:

پس از این کارها باید مدتی صبر کنید (مثلا یک ساعت) تا سیستم به صورت معمول کارهای متداول خودش را انجام دهد. پس از آن می‌توان به گزارشات AMR مراجعه کرد.

برای اینکار بر روی بانک اطلاعاتی Management Data Warehouse که در ابتدای عملیات ایجاد شد، کلیک راست نمائید و سپس مراحل ذیل را طی کنید:

Reports > Management Data Warehouse > Transaction Performance Analysis Overview

در گزارش ایجاد شده، ذیل گزینه‌ی usage analysis لینک‌هایی وجود دارند که با مراجعه به آن‌ها، چارت‌هایی از میزان مصرف بانک‌های اطلاعاتی مختلف سیستم ارائه می‌شود. اگر پیام No data available را مشاهده کردید، یعنی هنوز باید مقداری صبر کنید تا کار جمع آوری اطلاعات به پایان برسد.

در این چارت‌ها بانک‌های اطلاعاتی که در سمت راست، بالای تصویر قرار می‌گیرند، انتخاب مناسبی برای تبدیل به بانک‌های اطلاعاتی درون حافظه‌ای هستند. محور افقی آن از چپ به راست بیانگر میزان کاهش سختی انتقال یک جدول به جدول درون حافظه‌ای است (با درنظر گرفتن تمام مسایلی که باید تغییر کنند یا نوع‌های داده‌ای که باید اصلاح شوند) و محور عمودی آن نمایانگر میزان بالا رفتن پاسخ دهی سیستم در جهت انجام کار بیشتر است.

هر زمان هم که کار تصمیم‌گیری شما به پایان رسید، می‌توانید بر روی گزینه‌ی Data collection کلیک راست کرده و آن‌را غیرفعال نمائید.

برای مطالعه بیشتر

SQL Server 2014 Field Benchmarking In-Memory OLTP and Buffer Pool Extension Features
New AMR Tool: Simplifying the Migration to In-Memory OLTP
A Tour of the Hekaton AMR Tool
SQL Server 2014 Memory Optimization Advisor
Getting started with the AMR tool for migration to SQL Server In-memory OLTP Tables
How to Use Microsoft's AMR Tool
SQL Server 2014's Analysis, Migrate, and Report Tool

عنوان: ماندگاری با تاخیر در SQL Server 2014
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۴ ۱۳:۳۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Performance, In-Memory OLTP

به صورت پیش فرض SQL Server از روش write-ahead log - WAL استفاده می‌کند. به این معنا که کلیه تغییرات، پیش از commit نهایی باید در لاگ فایل آن نوشته شوند. این مساله با تعداد بالای تراکنش‌ها تا حدودی بر روی سرعت سیستم می‌تواند تاثیرگذار باشد. برای بهبود این وضعیت، در SQL Server 2014 قابلیتی به نام delayed_durability اضافه شده‌است که با فعال سازی آن، کلیه اعمال مرتبط با لاگ‌های تراکنش‌ها به صورت غیرهمزمان انجام می‌شوند. به این ترتیب تراکنش‌ها زودتر از معمول به پایان خواهد رسید؛ با این فرض که نوشته شدن تغییرات در لاگ فایل‌ها، در آینده‌ای محتمل انجام خواهند شد. این مساله به معنای فدا کردن D در ACID است (Atomicity, Consistency, Isolation, Durability). البته باید دقت داشت که رسیدن به ACID کامل هزینه‌بر است و شاید خیلی از اوقات تمام اجزای آن نیازی نباشند یا حتی بتوان با اندکی تخفیف آن‌ها را اعمال کرد؛ مانند D به تاخیر افتاده.

برای اینکار SQL Server از یک بافر 60 کیلوبایتی برای ذخیره سازی اطلاعات لاگ‌هایی که قرار است به صورت غیرهمزمان با تراکنش‌ها نوشته شوند، استفاده می‌کند. هر زمان که این 60KB پر شد، آن‌را flush کرده و ثبت خواهد نمود. به این ترتیب به دو مزیت خواهیم رسید:

- پردازش تراکنش‌ها بدون منتظر شدن جهت commit نهایی در دیسک سخت ادامه خواهند یافت. صبر کمتر به معنای امکان پردازش تراکنش‌های بیشتری در یک سیستم پر ترافیک است.
- با توجه به بافری که از آن صحبت شد، اینبار اعمال Write به صورت یک سری batch اعمال می‌شوند که کارآیی و سرعت بیشتری نسبت به حالت تکی دارند.

اندکی تاریخچه

ایده یک چنین عملی 28 سال قبل توسط Hal Berenson ارائه شده‌است! اوراکل آن‌را در سال 2006 تحت عنوان Asynchronous Commit پیاده سازی کرد و مایکروسافت در سال 2014 آن‌را ارائه داده‌است.

فعال سازی ماندگاری غیرهمزمان در SQL Server

فعال سازی این قابلیت در سطح بانک اطلاعاتی، در سطح یک تراکنش مشخص و یا در سطح رویه‌های ذخیره شده کامپایل شده مخصوص OLTP درون حافظه‌ای، میسر است.

برای فعال سازی ماندگاری با تاخیر در سطح یک دیتابیس، خواهیم داشت:

```
ALTER DATABASE dbname SET DELAYED_DURABILITY = DISABLED | ALLOWED | FORCED;
```

در اینجا اگر ALLOWED را انتخاب کنید، به این معنا است که لاگ کلیه تراکنش‌های مرتبط با این بانک اطلاعاتی به صورت غیرهمزمان نوشته می‌شوند. حالت FORCED نیز دقیقا به همین معنا است با این تفاوت که اگر حالت ALLOWED انتخاب شود، تراکنش‌های ماندگار (آن‌هایی که به صورت دستی DELAYED_DURABILITY را غیرفعال کرده‌اند)، سبب flush کلیه تراکنش‌هایی با ماندگاری به تاخیر افتاده خواهند شد و سپس اجرا می‌شوند. در حالت Forced تنظیم دسترسی DELAYED_DURABILITY = OFF در سطح تراکنش‌ها تاثیری نخواهد داشت؛ اما در حالت ALLOWED این مساله به صورت دستی در سطح یک تراکنش قابل لغو است.
البته باید توجه داشت، صرفنظر از این تنظیمات، یک سری از تراکنش‌ها همیشه ماندگار هستند و بدون تاخیر؛ مانند تراکنش‌های سیستمی، تراکنش‌های بین دو یا چند بانک اطلاعاتی و کلیه تراکنش‌هایی که با Change Data Capture، FileTable و Change Tracking سر و کار دارند.

در سطح تراکنش‌های می‌توان نوشت:

```
COMMIT TRANSACTION WITH (DELAYED_DURABILITY = ON);
```

و یا در رویه‌های ذخیره شده کامپایل شده مخصوص OLTP درون حافظه‌ای خواهیم داشت:

```
BEGIN ATOMIC WITH (DELAYED_DURABILITY = ON, ...)
```

سؤال: آیا فعال سازی DELAYED_DURABILITY بر روی مباحث locking و isolation levels تاثیر دارند؟
پاسخ: خیر. کلیه تنظیمات قفل گذاری‌ها همانند قبل و بر اساس isolation levels تعیین شده، رخ خواهند داد. تنها تفاوت در اینجا است که با فعال سازی DELAYED_DURABILITY، کار commit بدون صبر کردن برای پایان نوشته شدن اطلاعات در لاگ سیستم صورت می‌گیرد. به این ترتیب قفل‌های انجام شده زودتر آزاد خواهند شد.

سؤال: میزان از دست دادن اطلاعات احتمالی در این روش چقدر است؟
در صورتیکه سرور کرش کند یا ری‌استارت شود، حداکثر به اندازه‌ی 60KB اطلاعات را از دست خواهید داد (اندازه‌ی بافری که برای اینکار درنظر گرفته شده‌است). البته عنوان شده‌است که اگر ری‌استارت یا خاموشی سرور، از پیش تعیین شده باشد، ابتدا

کلیه لاگ‌های flush نشده، ذخیره شده و سپس ادامه‌ی کار صورت خواهد گرفت؛ ولی زیاد به آن اطمینان نکنید. اما همواره با فراخوانی sys.sp_flush_log، می‌توان به صورت دستی بافر لاگ‌های سیستم را flush کرد.

یک آزمایش

در ادامه قصد داریم یک جدول جدید را در بانک اطلاعاتی آزمایشی testdb2 ایجاد کنیم. سپس یکبار تنظیم DELAYED_DURABILITY = FORCED را انجام داده و 10 هزار رکورد را ثبت می‌کنیم و بار دیگر DELAYED_DURABILITY = DISABLED را تنظیم کرده و همین عملیات را تکرار خواهیم کرد:

```
CREATE TABLE tblData(
    ID INT IDENTITY(1, 1),
    Data1 VARCHAR(50),
    Data2 INT
);
CREATE CLUSTERED INDEX PK_tblData ON tblData(ID);
CREATE NONCLUSTERED INDEX IX_tblData_Data2 ON tblData(Data2);

-------------------------

alter database testdb2 SET DELAYED_DURABILITY = FORCED;

-------------------------

SET NOCOUNT ON
Print 'DELAYED_DURABILITY = FORCED'
DECLARE @counter AS INT = 0
DECLARE @start datetime = getdate()
WHILE (@counter < 10000)
BEGIN
      INSERT INTO tblData (Data1, Data2) VALUES('My Data', @counter)
      SET @counter += 1
END
Print DATEDIFF(ms,@start,getdate());
GO

-------------------------

alter database testdb2 SET DELAYED_DURABILITY = DISABLED;
truncate table tblData;
-------------------------

SET NOCOUNT ON
Print 'DELAYED_DURABILITY = DISABLED'
DECLARE @counter AS INT = 0
DECLARE @start datetime = getdate()
WHILE (@counter < 10000)
BEGIN
      INSERT INTO tblData (Data1, Data2) VALUES('My Data', @counter)
      SET @counter += 1
END
Print DATEDIFF(ms,@start,getdate());
GO

------------------------
```

با این خروجی:

```
DELAYED_DURABILITY = FORCED
666
DELAYED_DURABILITY = DISABLED
2883
```

در این آزمایش، سرعت insertها در حالت DELAYED_DURABILITY = FORCED حدود 4 برابر است نسبت به حالت معمولی.

برای مطالعه بیشتر

Control Transaction Durability

SQL Server 2014 Delayed Durability/Lazy Commit

Delayed Durability in SQL Server 2014 – Part 1

Is In-Memory OLTP Always a silver bullet for achieving better transactional speed

Delayed Durability in SQL Server 2014

عنوان: **امکان استفاده از یک هارد SSD بجای RAM در SQL Server 2014**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۱۵ ۱۵:۴۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Performance

Buffer Pool یکی از مصرف کنندگان اصلی حافظه در SQL Server است. برای مثال زمانیکه اطلاعاتی را از بانک اطلاعاتی دریافت می‌کنید، این داده‌ها در Buffer Pool کش می‌شوند. همچنین SQL Server اطلاعات کلیه Execution Plans را نیز در Plan Cache که جزئی از Buffer Pool است، برای استفاده‌ی مجدد نگهداری می‌کند. هر چقدر حافظه‌ی فیزیکی سرور شما بیشتر باشد، مقدار Buffer Pool نیز به همین میزان افزایش خواهد یافت که البته حداکثر آن‌را می‌توان در تنظیمات حافظه‌ی سرور محدود کرد (Max Server Memory setting).

در دنیای واقعی میزان حافظه‌ی فیزیکی سرورها محدود است. در SQL Server 2014 راه حلی برای این مشکل تحت عنوان Buffer Pool Extensions ارائه شده‌است که محل قرارگیری آن‌را در تصویر ذیل مشاهده می‌کنید:

Buffer Pool Extensions از یک فایل ساده که به آن Extension File نیز گفته می‌شود، تشکیل شده‌است و امکان ذخیره سازی آن بر روی هاردهای سریعی مانند SSD Driveها میسر است. این فایل، ساختاری را همانند page file، در سیستم عامل ویندوز دارد. در این حالت بجای اضافه کردن RAM بیشتر به سرور، یک Extension File را می‌توان بکار گرفت. هر زمان که Buffer Pool اصلی تحت فشار قرار گیرد (به میزان حافظه‌ای بیش از حافظه‌ی فیزیکی سرور نیاز باشد)، از این افزونه‌ی فایلی استفاده خواهد

شد.

اطلاعات جزئیات Buffer Pool را توسط کوئری ذیل می‌توان بدست آورد:

```
Select * from sys.dm_os_buffer_descriptors
```

نحوه‌ی فعال سازی و تنظیم Buffer Pool Extensions

قبل از هر کاری بهتر است وضعیت افزونه‌ی Buffer pool را بررسی کرد:

```
select * from sys.dm_os_buffer_pool_extension_configuration
```

SQLQuery1.sql - (l...VahidPC\Vahid (60))*

```
select * from sys.dm_os_buffer_pool_extension_configuration
```

100 %

Results | Messages

	path	file_id	state	state_description	current_size_in_kb
1	NULL	-1	0	BUFFER POOL EXTENSION DISABLED	NULL

همانطور که ملاحظه می‌کنید، در حالت پیش فرض غیرفعال است.

سپس یک فایل یک گیگابایتی را به عنوان افزونه‌ی Buffer pool ایجاد می‌کنیم.

```
ALTER SERVER CONFIGURATION
SET BUFFER POOL EXTENSION ON
 (FILENAME = 'd:\BufferPoolExt.BPE', SIZE = 1GB);
```

توصیه شده‌است که این فایل را در یک درایور پر سرعت SSD قرار دهید؛ ولی محدودیتی از لحاظ محل قرارگیری ندارد (هر چند به نظر فقط در حالتیکه از SSD Drive استفاده شود واقعا کار می‌کند).

اینبار اگر کوئری اول را اجرا کنیم، چنین خروجی قابل مشاهده است:

SQLffQuery1.sql -...VahidPC\Vahid (60))*

```
select * from sys.dm_os_buffer_pool_extension_configuration
```

100 %

Results | Messages

	path	file_id	state	state_description	current_size_in_kb
1	d:\BufferPoolExt.BPE	0	5	BUFFER POOL EXTENSION CLEAN PAGE CACHING ENABLED	1048576

این فایل به صورت خودکار در حین ری‌استارت یا خاموش شدن سرور، حذف شده و با راه اندازی مجدد آن، باز تولید خواهد شد.

تغییر اندازه‌ی افزونه‌ی Buffer pool

اگر سعی کنیم، یک گیگابایت را مثلا به 10 گیگابایت افزایش دهیم:

```
ALTER SERVER CONFIGURATION
SET BUFFER POOL EXTENSION ON
 (FILENAME = 'd:\BufferPoolExt.BPE', SIZE = 10GB);
```

با خطای ذیل مواجه خواهیم شد:

```
Could not change the value of the 'BPoolExtensionPath' property
```

برای رفع این مشکل، ابتدا باید افزونه‌ی Buffer pool را غیرفعال کرد:

```
ALTER SERVER CONFIGURATION
SET BUFFER POOL EXTENSION OFF
```

سپس می‌توان مجددا اندازه و یا مسیر دیگری را مشخص کرد. بهتر است اندازه‌ی این فایل را حدود 16 برابر حداکثر میزان حافظه‌ی سرور (Max Server Memory) تعیین کنید.
همچنین توصیه شده‌است که پس از غیرفعال کردن این افزونه، بهتر است یکبار instance جاری را ری استارت کنید.

چه زمانی بهتر است از افزونه‌ی Buffer pool استفاده شود؟
در محیط‌های read-heavy OLTP، استفاده از یک چنین افزونه‌ای می‌تواند میزان کارآیی و پاسخگویی سیستم را به شدت افزایش دهد (تا 50 درصد).

سؤال: آیا غیرفعال کردن افزونه‌ی Buffer pool سبب از دست رفتن اطلاعات می‌شود؟
خیر. BPE، تنها clean pages را در خود ذخیره می‌کند؛ یعنی تنها اطلاعاتی که Commit شده‌اند در آن حضور خواهند داشت و در این حالت حذف آن یا ری استارت کردن سرور، سبب از دست رفتن اطلاعات نخواهند شد.

برای مطالعه بیشتر

Buffer Pool Extension
SQL Server 2014 Buffer Pool Extensions
Do you require a SSD to use the Buffer Pool Extension feature in SQL Server 2014
Buffer Pool Extensions in SQL Server 2014
SQL Server 2014 – Buffer Pool Extension

عنوان: پردازش داده‌های جغرافیایی به کمک SQL Server و Entity framework
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۳۰ ۱۰:۴۵
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, SQL Server, GIS

پشتیبانی SQL Server از Spatial data

از SQL Server 2008 به بعد، نوع داده جدیدی به نام geography به نوع‌های قابل تعریف ستون‌ها اضافه شده‌است. در این نوع ستون‌ها می‌توان طول و عرض جغرافیایی یک نقطه را ذخیره کرد و سپس به کمک توابع توکاری از آن‌ها کوئری گرفت.

در اینجا نمونه‌ای از نحوه‌ی تعریف و همچنین مقدار دهی این نوع ستون‌ها را مشاهده می‌کنید:

```
CREATE TABLE [Geo](
[id] [int] IDENTITY(1,1) NOT NULL,
[Location] [geography] NULL
)

 insert into Geo( Location , long, lat ) values
( geography::STGeomFromText ('POINT(-121.527200 45.712113)', 4326))
```

متد geography::STGeoFromText یک SQL CLR function است. این متد در مثال فوق، مختصات یک نقطه را دریافت کرده‌است. همچنین نیاز دارد بداند که این نقطه توسط چه نوع سیستم مختصاتی ارائه می‌شود. عدد 4326 در اینجا یک SRID یا Spatial Reference System Identifier استاندارد است. برای نمونه اطلاعات ارائه شده توسط Google و یا Bing توسط این استاندارد ارائه می‌شوند.

در اینجا متدهای توکار دیگری مانند geography::STDistance برای یافتن فاصله مستقیم بین نقاط نیز ارائه شده‌اند. خروجی آن بر حسب متر است.

پشتیبانی از Spatial Data در Entity framework

پشتیبانی از نوع مخصوص geography، در EF 5 توسط نوع داده‌ای DbGeography ارائه شد. این نوع داده‌ای immutable است. به این معنا که پس از نمونه سازی، دیگر مقدار آن قابل تغییر نیست.

در اینجا برای نمونه مدلی را مشاهده می‌کنید که از نوع داده‌ای DbGeography استفاده می‌کند:

```
using System.Data.Entity.Spatial;

namespace EFGeoTests.Models
{
    public class GeoLocation
    {
        public int Id { get; set; }
        public DbGeography Location { get; set; }
        public string Name { get; set; }
        public string Type { get; set; }

        public override string ToString()
        {
            return string.Format("Name:{0}, Location:{1}", Name, Location);
        }
    }
}
```

به همراه یک Context، تا کلاس GeoLocation در معرض دید EF قرار گیرد:

```
using System;
using System.Data.Entity;
using EFGeoTests.Models;

namespace EFGeoTests.Config
{
    public class MyContext : DbContext
    {
        public DbSet<GeoLocation> GeoLocations { get; set; }

        public MyContext()
            : base("Connection1")
        {
            this.Database.Log = sql => Console.Write(sql);
        }
    }
}
```

برای مقدار دهی خاصیت Location از نوع DbGeography می‌توان از متد ذیل استفاده کرد که بسیار شبیه به متد geography::STGeoFromText عمل می‌کند:

```
private static DbGeography createPoint(double longitude, double latitude,  int coordinateSystemId =
4326)
  {
       var text = string.Format(CultureInfo.InvariantCulture.NumberFormat,"POINT({0} {1})", longitude,
latitude);
       return DbGeography.PointFromText(text, coordinateSystemId);
  }
```

تهیه منبع داده‌ی جغرافیایی

برای تدارک یک مثال واقعی جغرافیایی، نیاز به اطلاعاتی دقیق داریم. این نوع اطلاعات عموما توسط یک سری فایل مخصوص به نام Shapefiles که حاوی اطلاعات برداری جغرافیایی هستند ارائه می‌شوند. برای نمونه اطلاعات جغرافیایی به روز ایران را از آدرس ذیل می‌توانید دریافت کنید:

http://download.geofabrik.de/asia/iran.html

http://download.geofabrik.de/asia/iran-latest.shp.zip

پس از دریافت این فایل، به تعدادی فایل با پسوندهای shp، shx و dbf خواهیم رسید.
فایل‌های shp بیانگر فرمت اشکال ذخیره شده هستند. فایل‌های shx یک سری ایندکس بوده و فایل‌های dbf از نوع بانک اطلاعاتی dBase IV می‌باشند.

همچنین اگر فایل‌های prj را باز کنید، یک چنین اطلاعاتی در آن موجودند:

```
GEOGCS["GCS_WGS_1984",DATUM["D_WGS_1984",SPHEROID["WGS_1984",6378137,298.257223563]],PRIMEM["Greenwich"
,0],UNIT["Degree",0.017453292519943295]]
```

نکته‌ی مهمی که در اینجا باید مدنظر داشت، استاندارد GCS_WGS_1984 آن است. این استاندارد معادل است با استاندارد EPSG 4326. عدد 4326 آن جهت ثبت این اطلاعات در یک بانک اطلاعاتی SQL Server حائز اهمیت است (پارامتر coordinateSystemId در متد createPoint) و ممکن است از هر فایلی به فایل دیگر متفاوت باشد.

خواندن فایل‌های shp در دات نت

پس از دریافت فایل‌های shp و بانک‌های اطلاعاتی مرتبط با اطلاعات جغرافیایی ایران، اکنون نوبت به پردازش این فایل‌های مخصوص با فرمت بانک اطلاعاتی فاکس پرو مانند، رسیده‌است. برای این منظور می‌توان از پروژه‌ی سورس باز ذیل استفاده کرد:

C# Esri Shapefile Reader

این پروژه در خواندن فایل‌های shp بدون نقص عمل می‌کند اما توانایی خواندن نام‌های فارسی وارد شده در این نوع بانک‌های اطلاعاتی را ندارد. برای رفع این مشکل، سورس آن را از Codeplex دریافت کنید. سپس فایل Shapefile.cs را گشوده و ابتدای خاصیت Current آن‌را به نحو ذیل تغییر دهید:

```
/// <summary>
        /// Gets the current shape in the collection
        /// </summary>
        public Shape Current
        {
            get
            {
                if (_disposed) throw new ObjectDisposedException("Shapefile");
                if (!_opened) throw new InvalidOperationException("Shapefile not open.");

                // get the metadata
                StringDictionary metadata = null;
                if (!RawMetadataOnly)
                {
                    metadata = new StringDictionary();
                    for (int i = 0; i < _dbReader.FieldCount; i++)
                    {
                        string value = _dbReader.GetValue(i).ToString();
                        if (_dbReader.GetDataTypeName(i) == "DBTYPE_WVARCHAR")
                        {
                            // برای نمایش متون فارسی نیاز است
                            value = Encoding.UTF8.GetString(Encoding.GetEncoding(720).GetBytes(value));
                        }
                        metadata.Add(_dbReader.GetName(i),
                            value);
                    }
                }
```

در اینجا فقط سطر استفاده از Encoding خاصی با شماره 720 و تبدیل آن به UTF8 اضافه شده‌است. پس از آن بدون مشکل می‌توان برچسب‌های فارسی را از فایل‌های dBase IV این نوع بانک‌های اطلاعاتی استخراج کرد (اصلاح شده‌ی آن در فایل پیوست مطلب موجود است).

```
using System.Collections.Generic;
using System.Linq;
using Catfood.Shapefile;

namespace EFGeoTests
{
    public class MapPoint
    {
        public Dictionary<string, string> Metadata { set; get; }
        public double X { set; get; }
        public double Y { set; get; }
    }
```

```
    public static class ShapeReader
    {
        public static IList<MapPoint> ReadShapeFile(string path)
        {
            var results = new List<MapPoint>();

            using (var shapefile = new Shapefile(path))
            {
                foreach (var shape in shapefile)
                {
                    if (shape.Type != ShapeType.Point)
                        continue;

                    var shapePoint = shape as ShapePoint;
                    if (shapePoint == null)
                        continue;


                     var metadataNames = shape.GetMetadataNames();
                    if(!metadataNames.Any())
                        continue;

                    var metadata = new Dictionary<string, string>();
                    foreach (var metadataName in metadataNames)
                    {
                        metadata.Add(metadataName,shape.GetMetadata(metadataName));
                    }

                    results.Add(new MapPoint
                    {
                        Metadata = metadata,
                        X = shapePoint.Point.X,
                        Y = shapePoint.Point.Y
                    });
                }
            }

            return results;
        }
    }
}
```

در کدهای فوق به کمک کتابخانه‌ی C# Esri Shapefile Reader، اطلاعات نقاط بانک اطلاعاتی shape files را خوانده و به صورت لیست‌هایی از MapPoint بازگشت می‌دهیم. نکته‌ی مهم آن، Metadata است که از هر فایلی به فایل دیگر می‌توان متفاوت باشد. به همین جهت این اطلاعات را به شکل ویژگی‌های key/value در این نوع بانک‌های اطلاعاتی ذخیره می‌کنند.

افزودن اطلاعات جغرافیایی به بانک اطلاعاتی SQL Server به کمک Entity framework

فایل places.shp را در مجموعه فایل‌هایی که در ابتدای بحث عنوان شدند، می‌توانید مشاهده کنید. قصد داریم اطلاعات نقاط آن‌را به مدل GeoLocation انتساب داده و سپس ذخیره کنیم:

```
var points = ShapeReader.ReadShapeFile("IranShapeFiles\\places.shp");
            using (var context = new MyContext())
            {
                context.Configuration.AutoDetectChangesEnabled = false;
                context.Configuration.ProxyCreationEnabled = false;
                context.Configuration.ValidateOnSaveEnabled = false;

                if (context.GeoLocations.Any())
                    return;

                foreach (var point in points)
                {
                    context.GeoLocations.Add(new GeoLocation
                    {
                        Name = point.Metadata["name"],
                        Type = point.Metadata["type"],
                        Location = createPoint(point.X, point.Y)
                    });
                }

                context.SaveChanges();
            }
```

تعریف متد createPoint را که بر اساس X و Y نقاط، معادل قابل پذیرش آن‌را جهت SQL Server تهیه می‌کند، در ابتدای بحث مشاهده کردید.
در فایل‌های مرتبط با places.shp، متادیتا name، معادل نام شهرهای ایران است و type آن بیانگر شهر، روستا و امثال آن می‌باشد.
پس از اینکه اطلاعات مکان‌های ایران، در SQL Server ذخیره شدند، نمایش بصری آن‌ها را در management studio نیز می‌توان مشاهده کرد:

کوئری گرفتن از اطلاعات جغرافیایی

فرض کنید می‌خواهیم مکان‌هایی را با فاصله کمتر از 5 کیلومتر از تهران پیدا کنیم:

```
var tehran = createPoint(51.4179604, 35.6884243);

            using (var context = new MyContext())
            {
                // find any locations within 5 kilometers ordered by distance
```

```
                var locations = context.GeoLocations
                    .Where(loc => loc.Location.Distance(tehran) < 5000)
                    .OrderBy(loc => loc.Location.Distance(tehran))
                    .ToList();

                foreach (var location in locations)
                {
                    Console.WriteLine(location.Name);
                }
            }
```

همانطور که پیشتر نیز عنوان شد، متد Distance بر اساس متر کار می‌کند. به همین جهت برای تعریف 5 کیلومتر به نحو فوق عمل شده‌است. همچنین نحوه‌ی مرتب سازی اطلاعات نیز بر اساس فاصله از یک مکان مشخص صورت گرفته‌است.

و یا اگر بخواهیم دقیقا بر اساس مختصات یک نقطه، مکانی را بیابیم، می‌توان از متد SpatialEquals استفاده کرد:

```
var tehran = createPoint(51.4179604, 35.6884243);
            using (var context = new MyContext())
            {
                // find any locations within 5 kilometers ordered by distance
                var tehranLocation = context.GeoLocations.FirstOrDefault(loc =>
loc.Location.SpatialEquals(tehran));
                if (tehranLocation != null)
                {
                    Console.WriteLine(tehranLocation.Type);
                }
            }
```

کدهای کامل این مثال را از اینجا می‌توانید دریافت کنید:

EFGeoTests.zip

نظرات خوانندگان

نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۳/۰۳/۳۱ ۱۰:۴۱

سلام
مشکلی که قبل‌تر در حین کار با فایل‌های shp برخوردم، تغییر مقیاس نقشه و نگاشت مختصات خاص و توزیع شده ای بود که باید روی نقشه نمایش داده می‌شد . (مثلا نمایش مراکز استان‌ها روی نقشه و تغییر scale نقشه و به تبع اون، تغییر مکان مختصات)
برای این کار چه راهکاری هست؟
* دسترسی به فایل های dbf و prj و sbn و sbx و shx وجود دارد.
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۳۱ ۱۱:۱۰

این‌ها بیشتر مسایل نمایشی است و توانمندی ابزار نمایش دهنده‌ی اطلاعات نقشه. نیازی نیست در اصل دیتابیس و اطلاعات، تغییری حاصل شود؛ چون اندازه‌ی نمایشی حتی اگر 10 برابر هم شود، در فاصله‌ی بین تهران و شیراز نهایتا تغییری حاصل نخواهد شد و طول و عرض جغرافیایی مکان‌ها ثابت خواهند ماند. برای نمونه اگر مثال پیوست شده را اجرا کنید، خود management studio امکان تغییر اندازه‌ی نمایشی را دارد:

در برنامه‌های دات نت هم برای مثال از SharpMap می‌شود برای نمایش این نوع اطلاعات به همراه تغییر اندازه و ابعاد خودکار نقشه استفاده کرد. برای برنامه‌های وب هم jVectorMap چنین قابلیت‌هایی را دارد.

نویسنده: بهراد ایزدی
تاریخ: ۱۳۹۳/۰۴/۰۲ ۱۵:۰

سلام
ممنون برای این آموزش خوب
بعضی مناطق فایل shp رو ندارند و فقط دو نوع OSM براشون موجود هست راهی برای خواندن اونها وجود داره ؟

ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۰۲ ۱۵:۱۲

- برنامه‌ی سورس باز ArcGIS Editor for OpenStreetMap یک چنین قابلیتی را دارد.
- همچنین یک سری 4 قسمتی را در اینجا می‌توانید در مورد تبدیل open street maps به داده‌های SQL Server مطالعه کنید.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۳/۰۷/۱۳ ۱۷:۶

آیا ممکن هست به جای نوع داده ی DbGeography از نوع داده‌ی SQLGeometry استفاده کرد؟
-ویرایش
تنها کافی است نوع Property مورد نظر را DbGeometry تعیین کرد.

عنوان: استفاده از قابلیت پارتیشن بندی در آرشیو جداول بانک‌های اطلاعاتی SQL Server
نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۴/۰۸ ۱۴:۲۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Partitioning, Data Archiving

1- مقدمه پارتیشن بندی در بانک اطلاعاتی SQL Server، از ویژگی‌هایی است که از نسخه 2005، به این محصول اضافه شده است. بکارگیری این قابلیت که با Split کردن، محتوای یک جدول و قرار دادن آنها در چندین فایل، برای جداول حجیم، به ویژه جداولی که داده‌های آن حاوی مقادیر تاریخچه‌ای است، بسیار سودمند است.
سادگی در مدیریت داده‌ها و شاخص‌های موجود یک جدول (از قبیل اندازه فضای ذخیره سازی و استراتژی جدید Back up گیری)، اجرای سریعتر کوئری هایی که روی یک محدوده از داده‌ها کار می‌کنند و سهولت در آرشیو داده‌های قدیمی یک جدول، از قابلیت‌هایی است که استفاده از این ویژگی بوجود می‌آورد.
محدوده استفاده از این ویژگی روی یک بانک اطلاعاتی و در یک Instance است. بنابراین مباحث مرتبط با معماری Scalability را پوشش نمی‌دهد و صرفاً Solution ایی است که در یک Instance بانک اطلاعاتی استفاده می‌شود.

2- Data File و Filegroup هر بانک اطلاعاتی در حالت پیش فرض، شامل یک فایل داده‌ای (.MDF) و یک فایل ثبت تراکنشی (.LDF) می‌باشد. می‌توان جهت ذخیره سطرهای داده‌ای از فایل‌های بیشتری تحت نام فایل‌های ثانویه (.NDF) استفاده نمود. به همان طریق که در فایل سیستم، فایل‌ها به پوشه‌ها تخصیص داده می‌شوند، می‌توان Data File را به Filegroup تخصیص داد. چنانچه چندین Data File به یک Filegroup تخصیص داده شوند، داده‌ها در تمامی Data Fileها به طریق Round-Robin توزیع می‌شوند.

3- Partition Function مطابق با مقادیر تعریف شده در بدنه دستور، محدوده داده‌ای (پارتیشن‌ها) با استفاده از Partition Function ایجاد می‌شود. با در نظر گرفتن ستونی که به عنوان Partition Key انتخاب شده، این تابع یک Data Type را به عنوان ورودی دریافت می‌کند. در هنگام تعریف محدوده برای پارتیشن‌ها، به منظور مشخص کردن محدوده هر پارتیشن از Right و Left استفاده می‌شود.
Left نمایش دهنده‌ی حد بالای هر محدوده است و به طور مشابه، Right برای مشخص کردن حد پائین آن محدوده استفاده می‌شود. به منظور درک بهتر، به شکل زیر توجه نمائید:

```
CREATE PARTITION FUNCTION myRangePF1
(int) AS RANGE LEFT FOR VALUES
(1,100,1000)
```

Partition	1	2	3	4
Values	C1 <=1	C1 > 1 AND C1<=100	C1>100 AND C1<=1000	C1>1000

```
CREATE PARTITION FUNCTION myRangePF1
(int) AS RANGE RIGHT FOR VALUES
(1,100,1000)
```

Partition	1	2	3	4
Values	C1 <1	C1 >= 1 AND C1<100	C1>=100 AND C1<1000	C1>=1000

همانطور که مشاهده می‌شود، همواره نیاز به یک Filegroup اضافه‌تری از آنچه مورد نظرتان در تعریف تابع است، می‌باشد. بنابراین اگر Function دارای n مقدار باشد، به n+1 مقدار برای Filegroup نیاز است.
همچنین هیچ محدودیتی برای اولین و آخرین بازه در نظر گرفته نمی‌شود. بنابراین جهت محدود کردن مقادیری که در این بازه‌ها قرار می‌گیرند، می‌توان از Check Constraint استفاده نمود.

Right or Left 3-1-

یک سوال متداول اینکه از کدام مورد استفاده شود؟ در پاسخ باید گفت، به چگونگی تعریف پارتیشن هایتان وابسته است. مطابق شکل، تنها تفاوت این دو، در نقاط مرزی هر یک از پارتیشن‌ها می‌باشد. در بیشتر اوقات هنگام کار با داده‌های عددی می‌توان از Left استفاده نمود و بطور مشابه هنگامیکه نوع داده‌ها از جنس زمان است، می‌توان از Right استفاده کرد.

Partition Schema 4-

گام بعدی پس از ایجاد Partition Function، تعریف Partition Schema است، که به منظور قرار گرفتن هر یک از پارتیشن‌های تعریف شده توسط Function در Filegroupهای مناسب آن استفاده می‌شود.

Partition Table 5-

گام پایانی ایجاد یک جدول، استفاده از Partition Schema است، که داده‌ها را با توجه به رویه درون Partition Function مورد استفاده، ذخیره می‌کند. همانطور که می‌دانید هنگام ایجاد یک جدول، می‌توان مکان ذخیره شدن آنرا مشخص نمود.

```
Create Table <name> (…) ON …
```

دستور بعد از بخش ON، مشخص کننده مکان ذخیره جدول می‌باشد.

در هنگام ایجاد یک جدول، معمولاً جدول در Filegroup پیش فرض که PRIMARY است، قرار می‌گیرد. می‌توان با نوشتن نام Partition Schema و همچنین Partition Key که پیشتر ذکر آن رفت، بعد از بخش ON، برای جدول مشخص نمائیم که داده‌های آن به چه ترتیبی ذخیره شوند. ارتباط این سه به شرح زیر است:

توجه شود زمانیکه یک Primary Key Constraint به یک جدول اضافه می‌شود، یک Unique Clustered Index نیز همراه با آن ساخته می‌شود. چنانچه Primary Key شامل یک Clustered Index باشد، جدول با استفاده از این ستون (ستون‌های) شاخص ذخیره خواهد شد، در حالیکه اگر Primary Key شامل یک Non Clustered Index باشد، یک ساختار ذخیره-سازی اضافی ایجاد خواهد شد که داده‌های جدول در آن قرار خواهند گرفت.

Index & Data Alignment 6-

به عنوان یک Best Practice هنگام ایجاد یک Partition Table به منظور پارتیشن بندی، از ساختار Aligned Index استفاده شود. بدین ترتیب که تعریف Index، شامل Partition Key (ستونی که معیاری برای پارتیشن بندی است) باشد. چنانچه این عمل انجام شود، داده‌های ذخیره شده مرتبط با هر پارتیشن متناظر با همان شاخص، در فایل داده‌ای (.NDF) ذخیره خواهند شد. از این رو چنانچه کوئری درخواست شده از جدول روی یک محدوده باشد

```
Where [OrderDate] Between …
```

تنها از شاخص متناظر با این داده استفاده می‌شود. بدین ترتیب بکارگیری آن برای Execution Plan بسیار سودمند خواهد بود. همچنین می‌توان استراتژی بازیافت سودمندی با Back up گیری از Filegroup ایجاد کرد. هنگامی که Indexها به صورت Aligned هستند می‌توان در کسری از ثانیه، محتوای یک Partition را به یک جدول دیگر منتقل نمود (تنها با تغییر در Meta Data آن).

بدین ترتیب برای بهرمندی از این مزایا، استفاده از Aligned Index توصیه شده است.

7- Operations

از نیازمندی‌های متداول در پارتیشنینگ می‌توان به افزودن، حذف پارتیشن‌ها و جابجایی محتوای یک پارتیشن که برای عملیات آرشیو استفاده می‌شود، اشاره کرد.

7-1- Split Partition

به منظور ایجاد یک محدوده جدید به پارتیشن‌ها استفاده می‌شود. یک نکته مهم مادامی که عملیات انتقال داده‌ها به پارتیشن جدید انجام می‌گیرد، روی جدول یک قفل انحصاری قرار می‌گیرد و بدین ترتیب عملیات ممکن است زمانبر باشد.
به عنوان یک Best Practice همواره یک Partition خالی را Split نمائید و پس از آن اقدام به بارگذاری داده در آن نمائید.
به یاد داشته باشید پیش از انجام عملیات splitting روی Partition Function با تغییر در Partition Schema (و بکارگیری Next Used) مشخص نمائید چه محدوده‌ای در این Filegroup جدید قرار خواهد گرفت.

7-2- Merge Partition

به منظور ادغام پارتیشن‌ها استفاده می‌شود، چنانچه پارتیشن خالی نیست، برای عملیات ادغام مسائل Performance به علت اینکه در طول عملیات از Lock (قفل انحصاری) استفاده می‌شود، در نظر گرفته شود.

7-3- Switch Partition

چنانچه جدول و شاخص‌های آن به صورت Aligned هستند، می‌توانید از Switch in و Switch out استفاده نمائید. عملیات بدین ترتیب انجام می‌شود که بلافاصله محتوای یک پارتیشن یا جدول (Source) در یک پارتیشن خالی جدولی دیگر و یا یک جدول خالی (Target) قرار می‌گیرد. عملیات تنها روی Meta Data انجام می‌گیرد و هیچ داده ای منتقل نمی‌شود.
محدودیت‌های بکارگیری به شرح زیر است:
- جدول یا پارتیشن Target باید حتماً خالی باشد.
- جداول Source و Target حتماً باید در یک Filegroup یکسان قرار داشته باشند.
- جدول Source باید حاوی Aligned Indexهای مورد نیاز Target و همچنین مطابقت در Filegroup را دارا باشد.
- چنانچه Target به عنوان یک پارتیشن است، اگر Source جدول است بایست دارای یک Check Constraint باشد در غیر این صورت چنانچه یک پارتیشن است باید محدوده آن در محدوده Target قرار گیرد.

8- بررسی یک سناریوی نمونه

در ابتدا یک بانک اطلاعاتی را به طریق زیر ایجاد می‌کنیم:
این بانک مطابق تصویر، شامل 3 عدد فایل گروپ (FG1، FG2 و FG3) و 3 عدد دیتا فایل (P1، P2 و P3) می‌باشد. Filegroup پیش فرض Primary است، که چنانچه در تعریف جداول به نام Partition Schema و Partition Key مرتبط اشاره نشود، به طور پیش فرض در Filegroup موسوم به Primary قرار می‌گیرد. چنانچه چک باکس Default انتخاب شود، همانطور که قابل حدس زدن است، آن Filegroup در صورت مشخص نکردن نام Filegroup در تعریف جدول، به عنوان مکان ذخیره سازی انتخاب می‌شود.
چک باکس Read Only نیز همانطور که از نامش پیداست، چنانچه روی یک Filegroup تنظیم گردد، عملیات مربوط به Write روی

داده‌های آن قابل انجام نیست و برای Filegroup هایی که جنبه نگهداری آرشیو را دارند، قابل استفاده است.
چنانچه Filegroup ای را از حالت Read Only دوباره خارج کنیم، می‌توان عملیات Write را دوباره برای آن انجام داد.

پس از ایجاد بانک اطلاعاتی، گام بعدی ایجاد یک Partition Function و پس از آن یک Partition Schema است. همانطور که مشاهده می‌کنید در Partition Function از سه مقدار استفاده شده، بنابراین در Partition Schema باید از چهار Filegroup استفاده شود، که در مثال ما از Filegroup پیش فرض که Primary است، استفاده شده است.

```
USE [PartitionDB]
GO
CREATE PARTITION FUNCTION pfOrderDateRange(DATETIME)
AS
RANGE LEFT FOR VALUES ('2010/12/31','2011/12/31','2012/12/31')
GO
CREATE PARTITION SCHEME psOrderDateRange
AS
PARTITION pfOrderDateRange TO (FG1,FG2,FG3,[PRIMARY])
GO
```

پس از طی گام‌های قبل، به ایجاد یک جدول به صورت Aligned Index مبادرت ورزیده می‌شود.

```
CREATE TABLE Orders
```

```
(
OrderID INT IDENTITY(1,1) NOT NULL,
OrderDate DATETIME NOT NULL,
OrderFreight MONEY NULL,
ProductID INT NULL,
CONSTRAINT PK_Orders PRIMARY KEY CLUSTERED (OrderID ASC, OrderDate ASC)
ON psOrderDateRange (OrderDate)
) ON psOrderDateRange (OrderDate)
GO
```

در ادامه برای بررسی درج اطلاعات در پارتیشن با توجه به محدوده آنها اقدام به افزودن رکوردهایی در جدول ساخته شده می‌نمائیم.

```
SET NOCOUNT ON
DECLARE @OrderDate DATETIME
DECLARE @X INT
SET @OrderDate = '2010/01/01'
SET @X = 0
WHILE @X < 300
BEGIN
INSERT dbo.Orders ( OrderDate, OrderFreight, ProductID)
VALUES( @OrderDate + @X, @X + 10, @X)
 SET @X = @X + 1
END
GO
SET NOCOUNT ON
DECLARE @OrderDate DATETIME
DECLARE @X INT
SET @OrderDate = '2011/01/01'
SET @X = 0
WHILE @X < 300
BEGIN
INSERT dbo.Orders ( OrderDate, OrderFreight, ProductID)
VALUES( @OrderDate + @X, @X + 10, @X)
 SET @X = @X + 1
END
GO
SET NOCOUNT ON
DECLARE @OrderDate DATETIME
DECLARE @X INT
SET @OrderDate = '2012/01/01'
SET @X = 0
WHILE @X < 300
BEGIN
INSERT dbo.Orders ( OrderDate, OrderFreight, ProductID)
VALUES( @OrderDate + @X, @X + 10, @X)
 SET @X = @X + 1
END
GO
```

از طریق دستور Select زیر می‌توان نحوه توزیع داده‌ها را در جدول مشاهده کرد.

```
USE [PartitionDB]
GO
SELECT OBJECT_NAME(i.object_id) AS OBJECT_NAME,
p.partition_number, fg.NAME AS FILEGROUP_NAME, ROWS, au.total_pages,
CASE boundary_value_on_right
WHEN 1 THEN 'Less than'
ELSE 'Less or equal than' END AS 'Comparition',VALUE
FROM sys.partitions p JOIN sys.indexes i
ON p.object_id = i.object_id AND p.index_id = i.index_id
JOIN sys.partition_schemes ps ON ps.data_space_id = i.data_space_id
JOIN sys.partition_functions f ON f.function_id = ps.function_id
LEFT JOIN sys.partition_range_values rv
ON f.function_id = rv.function_id
AND p.partition_number = rv.boundary_id
JOIN sys.destination_data_spaces dds
ON dds.partition_scheme_id = ps.data_space_id
AND dds.destination_id = p.partition_number
JOIN sys.filegroups fg
ON dds.data_space_id = fg.data_space_id
JOIN (SELECT container_id, SUM(total_pages) AS total_pages
FROM sys.allocation_units
GROUP BY container_id) AS au
ON au.container_id = p.partition_id WHERE i.index_id < 2
```

خروجی دستور فوق به شرح زیر است:

	OBJECT_NAME	partition_number	FILEGROUP_NAME	ROWS	total_pages	Comparition	VALUE
1	Orders	1	FG1	300	4	Less or equal than	2010-12-31 00:00:00.000
2	Orders	2	FG2	300	4	Less or equal than	2011-12-31 00:00:00.000
3	Orders	3	FG3	300	4	Less or equal than	2012-12-31 00:00:00.000
4	Orders	4	PRIMARY	0	0	Less or equal than	NULL

در ادامه به ایجاد یک Filegroup جدید می‌پردازیم.

```
/* Query 2-3- Split a partition*/
-- Add FG4:
ALTER DATABASE PartitionDB ADD FILEGROUP FG4
Go
ALTER PARTITION SCHEME [psOrderDateRange] NEXT USED FG4
GO
ALTER PARTITION FUNCTION [pfOrderDateRange]() SPLIT RANGE('2013/12/31')
GO
-- Add Partition 4 (P4) to FG4:
GO
ALTER DATABASE PartitionDB ADD FILE
(
NAME = P4,
FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL10_50.SQLEXPRESS\MSSQL\DATA\P4.NDF'
, SIZE = 1024KB , MAXSIZE = UNLIMITED, FILEGROWTH = 10%)
 TO FILEGROUP [FG4]
--
GO
```

و در ادامه به درج اطلاعاتی برای بررسی نحوه توزیع داده‌ها در Filegroup هایمان می‌پردازیم.

```
SET NOCOUNT ON
DECLARE @OrderDate DATETIME
DECLARE @X INT
SET @OrderDate = '2013/01/01'
SET @X = 0
WHILE @X < 300
BEGIN
INSERT dbo.Orders ( OrderDate, OrderFreight, ProductID)
VALUES( @OrderDate + @X, @X + 10, @X)
 SET @X = @X + 1
END
GO
SET NOCOUNT ON
DECLARE @OrderDate DATETIME
DECLARE @X INT
SET @OrderDate = '2012/01/01'
SET @X = 0
WHILE @X < 300
BEGIN
INSERT dbo.Orders ( OrderDate, OrderFreight, ProductID)
VALUES( @OrderDate + @X, @X + 10, @X)
 SET @X = @X + 1
END
GO
```

خروجی کار تا این مرحله به شکل زیر است:

Results | Messages

	OBJECT_NAME	partition_number	FILEGROUP_NAME	ROWS	total_pages	Comparition	VALUE
1	Orders	1	FG1	300	4	Less or equal than	2010-12-31 00:00:00.000
2	Orders	2	FG2	300	4	Less or equal than	2011-12-31 00:00:00.000
3	Orders	3	FG3	600	5	Less or equal than	2012-12-31 00:00:00.000
4	Orders	5	PRIMARY	0	0	Less or equal than	NULL
5	Orders	4	FG4	300	4	Less or equal than	2013-12-31 00:00:00.000

جهت ادغام پارتیشن‌ها به طریق زیر عمل می‌شود:

```
/* Query 2-4- Merge Partitions */
ALTER PARTITION FUNCTION [pfOrderDateRange]() MERGE RANGE('2010/12/31')
Go
```

پس از اجرای دستور فوق خروجی به شکل زیر خواهد بود:

	OBJECT_NAME	partition_number	FILEGROUP_NAME	ROWS	total_pages	Comparition	VALUE
1	Orders	1	FG2	600	7	Less or equal than	2011-12-31 00:00:00.000
2	Orders	2	FG3	600	5	Less or equal than	2012-12-31 00:00:00.000
3	Orders	4	PRIMARY	0	0	Less or equal than	NULL
4	Orders	3	FG4	300	4	Less or equal than	2013-12-31 00:00:00.000

به منظور آرشیو نمودن اطلاعات به طریق زیر از Switch استفاده می‌کنیم. ابتدا یک جدول موقتی برای ذخیره رکوردهایی که قصد آرشیو آنها را داریم، ایجاد می‌کنیم. همانگونه که در تعریف جدول مشاهده می‌کنید، نام Filegroup ای که برای ساخت این جدول استفاده می‌شود، با Filegroup ای که قصد آرشیو اطلاعات آنرا داریم، یکسان است.

در ادامه می‌توان مثلاً با ایجاد یک Temporary Table به انتقال این اطلاعات بدون توجه به Filegroup آنها پرداخت.

```
/* Query 2-5- Switch Partitions */
USE [PartitionDB]
GO
CREATE TABLE [dbo].[Orders_Temp](
[OrderID] [int] IDENTITY(1,1) NOT NULL,
[OrderDate] [datetime] NOT NULL,
[OrderFreight] [money] NULL,
[ProductID] [int] NULL,
 CONSTRAINT [PK_OrdersTemp] PRIMARY KEY CLUSTERED ([OrderID] ASC,[OrderDate] ASC)ON FG2
) ON FG2
GO
USE [tempdb]
GO
CREATE TABLE [dbo].[Orders_Hist](
[OrderID] [int] NOT NULL,
[OrderDate] [datetime] NOT NULL,
[OrderFreight] [money] NULL,
[ProductID] [int] NULL,
 CONSTRAINT [PK_OrdersTemp] PRIMARY KEY CLUSTERED ([OrderID] ASC,[OrderDate] ASC)
)
GO
USE [PartitionDB]
GO
ALTER TABLE [dbo].[Orders] SWITCH PARTITION 1 TO [dbo].[Orders_Temp]
GO
INSERT INTO [tempdb].[dbo].[Orders_Hist]
SELECT * FROM  [dbo].[Orders_Temp]
GO
DROP TABLE [dbo].[Orders_Temp]
GO
```

```
SELECT * FROM [tempdb].[dbo].[Orders_Hist]
```

پس از اجرای کامل این دستورات، توزیع داده در بانک اطلاعاتی مثال مورد بررسی به شکل زیر است.

	OBJECT_NAME	partition_number	FILEGROUP_NAME	ROWS	total_pages	Comparition	VALUE
1	Orders	2	FG3	600	5	Less or equal than	2012-12-31 00:00:00.000
2	Orders	4	PRIMARY	0	0	Less or equal than	NULL
3	Orders	3	FG4	300	4	Less or equal than	2013-12-31 00:00:00.000
4	Orders	1	FG2	0	0	Less or equal than	2011-12-31 00:00:00.000

نظرات خوانندگان

نویسنده: محمد رضا موسائی
تاریخ: ۱۳۹۳/۰۷/۲۵ ۱۸:۲۳

بسیار امکان زیبایی است و بهتر اینکه اطلاعات قابل دسته بندی در فایل‌های ndf بوده و از همه مهمتر اینکه می‌توان از هر پارتیشن به صورت جداگانه backup تهیه کرد.

عنوان: **بررسی بارگذاری داده ها در انبار های داده و معرفی الگوهای بکار رفته در آن**
نویسنده: محمد رجبی
تاریخ: ۱۳۹۳/۰۴/۱۸ ۱۷:۳۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SSIS, data warehouse, ETL

مقدمه در لینکی که چندی پیش به اشتراک گذاشته بودم؛ به مطلبی تحت این عنوان اشاره شده بود: "آیا از KPI باید به انباره داده و هوش تجاری رسید؟" (بر گرفته از وبلاگ آقای جام سحر) که در آن به موانع پیش روی انجام پروژه‌های BI در ایران پرداخته شده است.

این مقاله بر گرفته از فصل سوم یکی از White Paperهای ماکروسافت با عنوان Microsoft EDW Architecture, Guidance and Deployment Best Practices می‌باشد. که به شرح عملیات Loading در فاز ETL می‌پردازد. از آنجا که به منظور پیاده سازی این نوع پروژه‌ها معمولاً در ایران برون سپاری صورت می‌گیرد و مدیران شرکت‌ها بیشتر درگیر سیستم‌های OLTP هستند و مجری پروژه (شرکت پیمانکار) معمولاً کوتاهترین مسیر را جهت انجام پروژه انتخاب می‌کند(و امروزه نیک میدانیم که "انتخاب مسیرهای کوتاه در زمان کم می‌تواند به پیچیدگی‌های بسیار جدی در دراز مدت منجر شود!") و همچنین از آنجا که متاسفانه به دلیل عدم ثبات مدیریت در ایران معمولاً "مدیریت برای تحویل پروژه تحت فشار است و نه برای مسائل پشتیبانی " و مسائل دیگری از این دست؛ چنانچه در تحویل گیری محصول به درستی تست نرم افزار صورت نگیرد، در نظر گرفتن موارد زیر:

```
Verification: Are we building the product right? ~ Software correctly implements a specific function
```

```
Validation: Are we building the right product? ~   Software is traceable to customer requirements
```

پروژه با شکست مواجه می‌شود و انتظارات مدیران بهره بردار را برآورده نمی‌کند. به هر روی در این مقاله به ترجمه مطالب زیر پرداخته می‌شود، توصیه میکنم در صورتی که با خواندن متن انگلیسی مشکلی ندارید، اصل مقاله مذکور خوانده شود.

1- Full Load vs Incremental Load
2- Detecting Net Changes
2-1- Pulling Net Changes – Last Change Column
2-2- Pulling Net Changes – No Last Change Column
2-3- Pushing Net Changes
3- ETL Patterns
3-1- Destination load Patterns
3-2- Versioned Insert Pattern
3-3- Update Pattern
3-4- Versioned Insert: Net Changes
4- Data Integration Best Practices
4-1- Basic Data Flow Patterns
4-1-1- Update Pattern
4-1-2- Update Pattern – ETL Framework
4-1-3- Versioned Insert Pattern
4-1-4- Update vs. Versioned Insert
4-2- Dimension Patterns
4-3- Fact Table Patterns
4-3-1- Managing Inferred Members

1- Full Load vs Incremental Load نسل‌های اولیه DW (اختصار Data Warehouse) به شکل Full Loads پیاده سازی می‌شدند، به این طریق که هر بار عملیات بارگذاری صورت می‌گرفت، DW از نو دوباره ساخته می‌شد. شکل زیر مراحل مختلف انجام شده در این روش را نمایش می‌دهد:

پروسه Full Load شامل مراحل زیر بود:

Drop Indexes: از آنجا که Indexها زمان بارگذاری را افزایش می‌دادند، این عمل صورت می‌پذیرفت.
Truncate Tables: تمامی رکوردهای موجود در جداول حذف می‌شدند.
Bulk Copy
Load Data
Post Process: شامل عملیاتی نظیر شاخص گذاری روی داده هایی است که اخیراً بارگذاری شده اند و....

روی هم رفته Full Load مسئله ای مشکل ساز بود، زیرا نیاز به زمانی برای بارگذاری مجدد داده‌ها داشت و مسئله‌ی مهم‌تر نداشتن امکان دستیابی به گزارشاتی تاریخچه ای با ماهیت زمان برای مشتریان کسب وکار بود. به این دلیل که همواره یک کپی از آخرین داده‌های موجود در سیستم عملیاتی درون DW قرار می‌گرفت؛ که با بکارگیری Full Load اغلب قادر به ارائه‌ی این نوع از گزارشات نبودیم، بدین ترتیب سازمان‌ها به نسل دوم روی آورند که در این دیدگاه از مفهوم Incremental Load استفاده می‌شود. اشکال زیر مراحلی که در این روش انجام می‌شود را نمایان می‌سازد:

Incremental Load with an Extract In area

Incremental Load without an Extract In area

مراحل Incremental Load شامل:

بارگذاری تغییرات نسبت به آخرین فرآیند بارگذاری انجام شده
درج / بروزرسانی تغییرات درون Production area
درج / بروزرسانی Consumption area نسبت به Production area

تفاوت‌های اصلی میان Full Load و Incremental Load در این است که در Incremental Load:

نیازی به پردازش‌های اضافی جهت حذف شاخص ها، پاک کردن تمامی رکوردهای جداول و ساخت مجدد شاخص‌ها نیست.
البته نیاز به رویه ای جهت شناسایی تغییرات می‌باشد.
و همچنین نیاز به بروزرسانی بعلاوه درج رکوردهای جدید نیز می‌باشد.

ترکیب این عوامل برای ساخت Incremental Load کارآمد تر، منجر به پیچیده‌تر شدن پیاده سازی و نگهداری آن نیز می‌شود.

2- Detecting Net Changes
فرآیند لود افزایشی ETL، بایست قادر به شناسائی رکوردهای تغییریافته در مبداء باشد، که این عمل با استفاده از هر یک از تکنیک‌های Push یا Pull انجام می‌شود.

در تکنیک Pull، فرآیند ETL رکوردهای تغییریافته در مبداء را انتخاب می‌کند:
ایده‌آل وجود داشتن یک ستون Last Changed در سیستم مبداء است؛ که از آن می‌توان جهت انتخاب رکوردهای تغییر یافته استفاده نمود.
چنانچه ستون Last Changed وجود نداشته باشد، تمامی رکوردهای مبداء باید با رکوردهای مقصد مقایسه شود.
در تکنیک Push، مبداء تغییرات را شناسائی می‌کند و آنها را به سمت مقصد Push می‌کند؛ این درخواست می‌تواند توسط فرآیند ETL انجام شود.
از آنجایی که پردازش ETL معمولاً در زمان هایی که Peak کاری وجود ندارد، اجرا می‌شود، استفاده از مکانیسم Pull برای شناسایی تغییرات نسبت به مکانسیم Push ارجحیت دارد.

2-1- Pulling Net Changes – Last Change Column بیشتر جداول در سیستم‌های مبداء حاوی ستون هایی هستند که زمان ایجاد و یا اصلاح رکوردها را ثبت می‌کنند. در نوع دیگری از سیستم‌های مبداء ستونی با مقدار عددی وجود دارد، که هر زمان رکوردی تغییر یافت به آن ستون مقداری اضافه می‌شود. هر دوی این تکنیک‌ها به فرآیند ETL اجازه می‌دهند، بطور کارآمدی رکوردهای تغییریافته را انتخاب کند. (با مقایسه، بیشترین مقدار قرار گرفته در آن ستون؛ که در طول آخرین اجرای فرآیند ETL بدست آمده است).
نمونه ای از جداول سیستم مبداء که دارای تغییرات زمانی است در شکل زیر نمایش داده می‌شود.

همچنین شکل زیر نشان می‌دهد، چگونه یک مقدار عددی می‌تواند به منظور انتخاب رکوردهای تغییریافته استفاده شود.

2-2- Pulling Net Changes – No Last Change Column شکل زیر گردش فرآیند را هنگامی که ستون Last Change وجود ندارد؛ نمایش می‌دهد.

این گردش فرآیند شامل:

Join میان مبداء و مقصد با استفاده از یک دستور Left Outer Join است.
تمامی رکوردهای مبداء که در مقصد وجود ندارند، پردازش می‌شوند.
زمانی که رکوردی در مقصد وجود داشته باشد مقادیر داده‌های مبداء و مقصد مقایسه می‌شوند.
تمامی رکوردهای مبداء که تغییر یافته اند پردازش می‌شوند.
از آنجایی که تمامی رکوردها پردازش می‌شوند، این روش بویژه برای جداول حجیم؛ روش کارآمدی نیست.

2-3- Pushing Net Changes دو متد متداول Push وجود دارد که در تصویر زیر نمایش داده شده است.

تفاوت این دو روش به شرح زیر است:

در سناریو اول (شکل سمت چپ)؛ بانک اطلاعاتی رابطه ای سیستم مبداء Transaction Log را مرتب مانیتور می‌کند تا تغییرات را شناسائی کرده و در ادامه تمامی این تغییرات را در جدولی در مقصد درج می‌کند.
در سناریو دوم؛ توسعه دهندگان Trigger هایی ایجاد می‌کنند تا هر زمان که رکوردی تغییر یافت، تغییرات در جدولی که در مقصد وجود دارد درج گردد.

مسئله ای که در هر دو مورد وجود دارد Load اضافه ای است؛ که روی سیستم مبداء وجود دارد و می‌تواند Performance سیستم‌های OLTP را تحت تاثیر قرار دهد. به هر روی سناریو نخست معمولاً کاراتر از سناریویی است که از Trigger استفاده می‌کند.

3- ETL Patterns

پس از شناسائی رکوردهایی که در مبداء تغییر یافته اند، نیاز داریم تا این تغییرات در مقصد اعمال شود. در این قسمت به معرفی الگوهایی که برای اعمال این تغییرات وجود دارد می‌پردازیم.

3-1- Destination load Patterns

تشخیص چگونگی اضافه نمودن تغییرات در مقصد تابع دو عامل زیر است:

آیا رکورد هم اینک در مقصد وجود دارد؟
الگوی استفاده شده برای جدول مقصد به کدام شکل است؟ (Versioned Insert یا Update)

فلوچارت زیر نشان می‌دهد، به چه شکل جداول مقصد متاثر از چگونگی پردازش رکوردهای مبداء قرار دارند. توجه داشته باشید که عمل بررسی بطور جداگانه و در یک لحظه صورت می‌گیرد.

Versioned Insert Pattern 3-2-

Kimball Type II Slowly Changing Dimension نمونه ای از الگوی Versioned Insert است؛ که در آن نمونه ای از یک موجودیت دارای ورژن‌های متعددی است. مطابق تصویر زیر؛ این الگو به ستون‌های اضافه ای نیاز دارند که وضعیت نمونه ای از یک رکورد را نمایش دهد.

Surrogate Key	Natural Key	Data...	Start Date	End Date	Record Status	Version #

این ستون‌ها به شرح زیر هستند:

Start Date: زمانی که وضعیت آن نمونه از رکورد فعال می‌شود.
End Date: زمانی که وضعیت آن نمونه از رکورد غیر فعال می‌شود.
Record Status: وضعیت‌های یک رکورد را نشان می‌دهد، که حداقل به شکل Active یا Inactive است.
Version #: این ستون که اختیاری می‌باشد، ورژن آن نمونه از رکورد را ثبت می‌کند.

برای مثال شکل زیر؛ بیانگر وضعیت اولیه رکوردی در این الگو است:

Surrogate Key	Natural Key	Data...	Start Date	End Date	Status	Ver #
100	AB549		2001-02-05	NULL	A	1

فرض کنید که این رکورد در تاریخ March 2 , 2010 در سیستم مبداء تغییر می‌کند. فرآیند ETL این تغییر را شناسائی می‌کند و همانند تصویر زیر؛ به شکل نمونه ای ثانویه از این رکورد، اقدام به درج آن می‌کند.

Surrogate Key	Natural Key	Data...	Start Date	End Date	Status	Ver #
100	AB549		2001-02-05	2010-03-02	I	1
537	AB549		2010-03-02	NULL	A	2

توجه داشته باشید زمانی که رکورد دوم در جدول درج می‌شود، به منظور بازتاب این تغییر؛ رکورد اول به شکل زیر بروزرسانی می‌گردد:

End Date: تا این زمان وضعیت این رکورد فعال بوده است.
Record Status:که Active به Inactive تغییر پیدا می‌کند.

در برخی از پیاده سازی‌های DW عمدتاً از الگوی Versioned Insert استفاده می‌شود و هرگز از الگوی Update استفاده نمی‌شود. مزیت این استراتژی در این است که تمامی تاریخچه تغییرات ردیابی و ثبت می‌شود. به هر روی غالباً هزینه ثبت کردن این تغییرات منجر به ایجاد نسخه‌های زیادی از تغییرات می‌شود. تیم DW برای مواردی که تغییرات متاثر از گزارشات تاریخچه ای نیستند، می‌توانند الگوی Update را در نظر گیرند.

Update Pattern 3-3-

الگوی Update روی رکورد موجود، تغییرات سیستم مبداء را بروزرسانی می‌کند. مزیت این روش در این است که همواره یک رکورد وجود دارد و در نتیجه باعث ایجاد Queryهای کارآمدتر می‌شود. تصویر زیر بیانگر ستون هایی است که برای پشتیبانی از الگوی Update بایست ایجاد کرد.

Surrogate Key	Natural Key	Data...	Record Status	Version #

این ستون‌ها به شرح زیر هستند:

Record Status: وضعیت‌های یک رکورد را نشان می‌دهد که حداقل به شکل Active یا Inactive است.
Version #: این ستون که اختیاری می‌باشد، ورژن آن نمونه از رکورد را ثبت می‌کند.

موارد اصلی الگوی Update عبارتند از:

تاریخ ثبت نمی‌شود. ابزاری ارزشمند برای نظارت بر داده ها، تغییرات تاریخی است و زمانی که ممیزی داده رخ می‌دهد؛ می‌تواند مفید واقع شود.
بروزرسانی‌ها یک الگوی مبتنی بر مجموعه هستند. استفاده از بروزرسانی هر بار یک رکورد در ابزار ETL خیلی کارآمد (موجه) نیست.

یک روش دیگر برای در نظر گرفتن موارد فوق؛ اضافه کردن یک جدول برای درج ورژن‌ها به الگوی Update است که در شکل زیر نشان داده شده است.

اضافه نمودن یک جدول تاریخچه، که تمامی تغییرات سیستم مبداء را ثبت می‌کند؛ نظارت و ممیزی داده‌ها را نیز فراهم می‌کند و همچنین بروزرسانی‌های کارآمد مبتنی بر مجموعه را برای جداول DW به ارمغان می‌آورد.

Versioned Insert: Net Changes 3-4-

این الگو غالباً در جداول حجیم Fact که بروزرسانی آنها پر هزینه است استفاده می‌شود. شکل زیر منطق استفاده شده در این الگو را نشان می‌دهد.

توجه داشته باشید در این الگو:
مقادیر مالی و عددی محاسبه شده؛ به عنوان یک Net Change از نمونه قبلی رکورد در جدول Fact ذخیره می‌شود.
هیچ گونه فعالیت Post Processing صورت نمی‌گیرد (از قبیل بروزرسانی جداول Fact پس از کامل شدن Data Flow). هدف استفاده از این الگو اجتناب از بروزرسانی روی جداول بسیار حجیم می‌باشد.
عدم بروزرسانی و همچنین اندازه جدول Fact زمینه ای را فراهم می‌کند که منطق شناسائی رکوردهای تغییریافته پیچیده تر می‌شود. این پیچیدگی از آنجا ناشی می‌شود که نیاز به مقایسه رکوردهای جدول Fact آتی با جدول Fact موجود می‌باشد.

4- Data Integration Best Practices

هم اکنون پس از آشنایی با مفاهیم و الگوهای توزیع داده‌ها به ارائه تعدادی نمونه می‌پردازیم؛ که بتوان این ایده‌ها و الگوها را در عمل پوشش داد.

4-1- Basic Data Flow Patterns

هر یک از الگوهای Update Pattern و Versioned Insert Pattern می‌توانند برای انواعی از جداول بکار روند که معروفترین آن‌ها توسط Kimball ساخته شده اند.

Slowly Changing Dimension Type I (SCD I): از Update Pattern استفاده می‌کند.
Slowly Changing Dimension Type II (SCD II): از Versioned Insert Pattern استفاده می‌کند.
Fact Table: نوع الگویی که استفاده می‌کند به نوع جدول Fact ای که Load خواهد شد بستگی دارد.

4-1-1- Update Pattern

مطابق تصویر زیر جدولی که تنها حاوی ورژن فعلی رکورد هاست؛ از Update Dataflow Pattern استفاده می‌کند.

مواردی که در مورد این گردش کاری باید در نظر داشت به شرح زیر است:

این Data Flow فقط سطرهایی را به یک مقصد اضافه خواهد کرد. SSIS دارای گزینه "Table or view fast load" می‌باشد که بارگذاری‌های انبوه و سریع را پشتیبانی می‌کند.
درون یک Data Flow بروزرسانی رکوردها را می‌توان با استفاده از تبدیل OLE DB Command انجام داد. توجه داشته باشید خروجی‌های این تبدیل در یک دستور Update به ازای هر رکورد بکار می‌رود؛ مفهوم بروزرسانی انبوه در این Data Flow وجود ندارد. بدین ترتیب الگوی فعلی ارائه شده؛ تنها رکوردها را درج می‌کند و هرگز در این Data Flow رکوردها Update نمی‌شوند.
هر جدول دارای یک جدول تاریخچه است که برای ذخیره همه فعالیت‌های مرتبط با آن بکار می‌رود. یک رکورد در جدول تاریخچه زمانی درج خواهد شد؛ که رکورد مبداء در مقصد وجود داشته باشد ولی دارای مقداری متفاوت باشد.
راه دیگر فرستادن تغییرات رکوردها به یک جدول کاری است که پس از پایان یافتن فرآیند Update ، خالی (Truncate) می‌شود.
مزیت نگهداری تمامی رکوردها در یک جدول تاریخچه؛ ایجاد یک دنباله ممیزی است که می‌تواند برای نظارت بر داده‌ها به منظور نمایان ساختن موارد مطرح شده توسط مصرف کننده‌های کسب و کار استفاده شود.
گزینه‌های متفاوتی برای تشخیص تغییرات رکوردها وجود دارد که در ادامه به شرح آنها می‌پردازیم.

شکل زیر نمایش دهنده چگونگی پیاده سازی Update Dataflow Pattern در یک SSIS می‌باشد:

این SSIS شامل عناصر زیر است:

Destination table lookup:

به منظور تشخیص اینکه رکورد در جدول مقصد وجود دارد از "lkpPersonContact" استفاده می‌کنیم.

Change detection logic:

با استفاده از "DidRecordChange" مبداء و مقصد مقایسه می‌شوند. اگر تفاوتی بین مبداء و مقصد وجود نداشت؛ رکورد نادیده گرفته می‌شود. چنانچه بین مبداء و مقصد تفاوت وجود داشت؛ رکورد در جدول تاریخچه درج خواهد شد.

Detection Inserts:

رکوردها در جدول مقصد درج خواهند شد در صورتیکه در آن وجود نداشته باشند.

Destination History Inserts:

رکوردها در جدول تاریخچه مقصد درج خواهند شد، در صورتیکه (در مقصد) وجود داشته باشند.

پس از اتمام Data Flow یک روال Post-processing مسئولیت بروزرسانی رکوردهای جدول اصلی و رکوردهای ذخیره شده در جدول تاریخچه را بر عهده دارد که می‌تواند مطابق تصویر زیر با استفاده از یک Execute Process Task پیاده سازی شود.

PostProcess مسئولیت اجرای تمامی فعالیت‌های زیر را در این الگو برعهده دارد که شامل:

بروزرسانی رکوردهای جداول با استفاده از رکوردهای درج شده در جدول تاریخچه.
درج تمامی رکوردهای جدید (نسخه اولیه و در درون جدول تاریخچه). کلید اصلی جداولی که ستون آنها IDENTITY است مقدار نامشخصی دارد؛ تا زمانی که درج صورت گیرد، این به معنای آن است که پیش از انتقال آنها به جدول تاریخچه نیاز است منتظر درج شدن آنها باشیم.

4-1-2- Update Pattern – ETL Framework

تصویر زیر بیانگر انجام این عملیات با استفاده از ابزارهای ETL است.
در نگاه نخستین ممکن است Data Flow از نوع اصلی خود پیچیده‌تر به نظر آید؛ که در واقع این گونه نیز هست، زیرا در فاز توسعه بیشتر Frameworkها جهت پیاده سازی به یک زمان اضافه‌تری نیاز دارند. به هر روی این زمان جهت اجتناب از هزینه روزانه تطبیق داده‌ها گرفته خواهد شد.
مزایای حاصل شده از افزودن این منطق اضافی عبارت است از:

پشتیبانی از ستون هایی که کارهای ممیزی و نظارت بر داده‌ها را آسانتر می‌کنند.
تعداد سطرها شاخص مناسبی است که می‌تواند بهبود آن Data Flow خاص را فراهم کند. ناظر اطلاعات با استفاده از تعداد رکوردها می‌تواند ناهنجاری‌ها را شناسائی کند.

بهره برداران ETL و ناظران اطلاعات می‌توانند با استفاده از خلاصه تعداد رکوردها درک بیشتری درباره فعالیت‌های آن کسب کنند. پس از آنکه تعداد رکوردها، مشکوک به نظر آمد؛ تحقیقات بیشتری می‌تواند اتفاق افتد. (با عمیق‌تر شدن در جزئیات گزارشات)

Versioned Insert Pattern 4-1-3-

جدولی که به صورت Versioned Insert پر شده است می‌تواند از Versioned Insert Dataflow Pattern استفاده کند. همانند شکل زیر که گردش کار در آن برای کارآئی بیشتر بازنگری شده است.

توجه داشته باشید Data Flow در این روش شامل:

تمامی رکوردهای جدید و تغییر یافته در جدول Versioned Insert قرار می‌گیرند.
این روش دارای Data Flow ساده‌تری نسبت به الگوی Update می‌باشد.

شکل زیر SSIS versioned insert data flow pattern را نشان می‌دهد:

تعدادی نکته در Data Flow فوق وجود دارد که عبارتند از:

در شیء "lkpDimGeography" گزینه "Redirect rows to no match output" با مقدار "Ignore Failures" تنظیم شده است.
شیء "DidRecordChange" بررسی می‌کند چنانچه ستون‌های مبداء و مقصد یکسان باشند، آیا کلید اصلی جدول مقصد Not Null است. اگر این عبارت True ارزیابی شود، رکورد نادیده گرفته می‌شود.
منطق شناسائی تغییرات دربردارنده تغییرات ستون داده ای در مبداء نمی‌باشد.
ستون و تعداد رکوردها مشابه با Data Flow قبلی (ETL Framework) می‌باشد.

4-1-4- Update vs. Versioned Insert

الگوی Versioned Insert نسبت الگوی Update دارای پیاده سازی ساده‌تر و فعالیت‌های I/O کمتری است. از منظر دیگر، جدولی که از الگوی Update استفاده می‌کند، دارای تعداد رکوردهای کمتری است که می‌تواند به معنای Performance بهتر نیز تعبیر شود. ممکن است سوالی مطرح شود، اینکه چرا برای انجام کار به جدول تاریخچه نیاز است؛ این جدول را که نمی‌توان Truncate نمود،

پس چرا به منظور بروزرسانی از جدول اصلی استفاده می‌شود؟ پاسخ این پرسش در این است که جدول تاریخچه، ناظر اطلاعات و ممیزین داده را قادر می‌سازد، تغییرات در طول زمان را پیگیری نمایند.

Dimension Patterns 4-2-

بروزرسانی Dimension موارد زیر را شامل می‌شود:

پیگیری تاریخچه
انجام بروزرسانی
تشخیص رکوردهای جدید
مدیریت surrogate keys

چنانچه با یک Dimension کوچک مواجه هستید (با مقدار هزاران رکورد یا کمتر، که با صدها هزار رکورد یا بیشتر ضدیت دارد)، می‌توانید از تبدیل "Slowly Changing Dimension" که بصورت Built-in در SSIS موجود است، استفاده نمائید. به هر روی با آنکه این تبدیل چندین ویژگی محدودکننده Performance دارد، اغلب کارآمدتر از پروسسه هایی که توسط خودتان ایجاد می‌شود. در واقع فرآیند بارگذاری در جداول Dimension با مقایسه داده‌ها بین مبداء و مقصد انجام می‌شود. به طور معمول مقایسه روی یک ورژن جدید و یا مجموعه ای از سطرهای جدید یک جدول با مجموعه داده‌های موجود در جدول متناظرش صورت می‌گیرد. پس از تشخیص چگونگی تغییر در داده ها، یک سری عملیات درج و بروزرسانی انجام می‌شود. شکل زیر نمونه ای از پردازش سریع در Dimension را نمایش می‌دهد؛ که شامل مراحل اساسی زیر است:

منبع فوقانی سمت چپ، رکوردها را در یک SSIS از یک سیستم مبداء (یا یک سیستم میانی) به شکل Pull دریافت می‌کند. منبع فوقانی سمت راست، داده‌ها را از خود جدول Dimension به شکل Pull دریافت می‌کند.
با استفاده از Merge Join رکوردها از طریق Source Key شان مقایسه می‌شوند. (در شکل بعدی جزئیات این مقایسه نمایش داده شده است.)
با استفاده از یک Conditional Spilt داده‌ها ارزیابی می‌شوند؛ سطرها یا مستقیماً در جدول Dimension درج می‌شوند (منبع تحتانی سمت چپ) و یا در یک جدول عملیاتی (منبع تحتانی سمت راست) جهت انجام بروزرسانی درج می‌شوند.
در گام پایانی (که نمایش داده نشده) مجموعه ای از بروزرسانی بین جدول عملیاتی و جدول Dimension صورت می‌گیرد.

با Merge Join ارتباطی بین رکوردهای مبداء و رکوردهای مقصد برقرار می‌شود. (در این مثال "CustomerAlternateKey"). هنگامی که از این دیدگاه استفاده می‌کنید، خاطر جمع شوید که نوع Join با مقدار "Left outer join" تنظیم شده است؛ بدین ترتیب قادر هستید تا رکوردهای جدید را از مبداء تشخیص دهید؛ از آنجا که هنوز در جدول Dimension قرار نگرفته اند.

گام پایانی به منظور تشخیص اینکه آیا رکورد، جدید یا تغییر یافته است (یا بلاتکلیف است)، مقایسه داده هاست. شکل زیر نمایش می‌دهد چگونه این ارزیابی با استفاده از تبدیل "Conditional Spilt" صورت می‌گیرد.

Conditional Spilt مستقیماً با استفاده از یک Adapter تعریف شده روی مقصد یا یک جدول کاری بروزرسانی که از یک Adapter تعریف شده روی مقصد استفاده می‌کند؛ توسط مجموعه دستور Update زیر، رکوردها را در جدول Dimension قرار می‌دهد. دستور Update زیر مستقیماً با استفاده از روش Join روی جدول Dimension و جدول کاری، مجموعه ای را بصورت انبوه بروزرسانی می‌کند.

```
UPDATE AdventureWorksDW2008R2.dbo.DimCustomer
    SET AddressLine1 = stgDimCustomerUpdates.AddressLine1
    , AddressLine2 = stgDimCustomerUpdates.AddressLine2
    , BirthDate = stgDimCustomerUpdates.BirthDate
    , CommuteDistance = stgDimCustomerUpdates.CommuteDistance
    , DateFirstPurchase = stgDimCustomerUpdates.DateFirstPurchase
    , EmailAddress = stgDimCustomerUpdates.EmailAddress
    , EnglishEducation = stgDimCustomerUpdates.EnglishEducation
    , EnglishOccupation = stgDimCustomerUpdates.EnglishOccupation
    , FirstName = stgDimCustomerUpdates.FirstName
    , Gender = stgDimCustomerUpdates.Gender
    , GeographyKey = stgDimCustomerUpdates.GeographyKey
    , HouseOwnerFlag = stgDimCustomerUpdates.HouseOwnerFlag
    , LastName = stgDimCustomerUpdates.LastName
    , MaritalStatus = stgDimCustomerUpdates.MaritalStatus
    , MiddleName = stgDimCustomerUpdates.MiddleName
```

```
    , NumberCarsOwned = stgDimCustomerUpdates.NumberCarsOwned
    , NumberChildrenAtHome = stgDimCustomerUpdates.NumberChildrenAtHome
    , Phone = stgDimCustomerUpdates.Phone
    , Suffix = stgDimCustomerUpdates.Suffix
    , Title = stgDimCustomerUpdates.Title
    , TotalChildren = stgDimCustomerUpdates.TotalChildren
FROM AdventureWorksDW2008.dbo.DimCustomer DimCustomer
  INNER JOIN dbo.stgDimCustomerUpdates ON
DimCustomer.CustomerAlternateKey = stgDimCustomerUpdates.CustomerAlternateKey
```

4-3- Fact Table Patterns

جداول Fact به پردازش‌های منحصر به فردی نیازمند هستند، نخست به کلیدهای Surrogate جدول Dimension نیاز دارند تا Measureهای محاسبه شدنی را بدست آورند. این اعمال از طریق تبدیلات Lookup، Merge Join و Derived Column صورت می‌گیرد. با بروزرسانی ها، تفاضل رکوردها و یا Snapshot بیشتر این فرآیندهای دشوار انجام می‌شوند.

4-3-1- Inserts

روی اغلب جداول Fact عمل درج صورت می‌گیرد؛ که کار متداولی در جدول Fact می‌باشد. شاید ساده‌ترین کار که در فرآیند ساخت ETL صورت می‌گیرد، عملیات درج روی تنها تعدادی از جدول Fact می‌باشد. درج کردن در صورت لزوم بارگذاری انبوه داده ها، مدیریت شاخص‌ها و مدیریت پارتیشن‌ها را شامل می‌شود.

4-3-2- Updates

بروزرسانی روی جداول Fact معمولاً به یکی از سه طریق زیر انجام می‌گیرد:

از طریق یک تغییر یا بروزرسانی رکورد
از طریق یک دستور Insert خنثی کننده (Via an Insert of a compensating transaction)
با استفاده از یک SQL MERGE

در موردی که تغییرات با فرکانس کمی روی جدول Fact صورت می‌گیرد و یا فرآیند بروزرسانی قابل مدیریت است؛ ساده‌ترین روش انجام یک دستور Update روی جدول Fact می‌باشد. نکته مهمی که هنگام انجام بروزرسانی باید به خاطر داشته باشید، استفاده از روش بروزرسانی مبتنی بر مجموعه است؛ به همان طریق که در قسمت الگوهای Dimension ذکر آن رفت.
در طریقی دیگر (درج compensating) می‌توان اقدام به درج رکورد تغییر یافته نمود، تا ترجیحاً بروزرسانی روی آن صورت گیرد. این استراتژی به سادگی داده‌های جدول Fact میان سیستم مبداء و مقصد را که تغییر یافته اند، به صورت یک رکورد جدید درج خواهد کرد. تصویر زیر مثالی از اجرای موارد فوق را نمایش می‌دهد.

Source ID	Measure Value
12345	80

Source data

Source ID	Measure Value
12345	100

Current fact table data

Source ID	Current Measure Value
12345	100
12345	**- 20**

New fact table data

در آخرین روش از یک دستور SQL MERGE استفاده می‌شود که در آن با استفاده از ادغام و مقایسه، تمامی داده‌های جدید و تغییر یافته جدول Fact، درج و یا بروزرسانی می‌شوند. نمونه ای از استفاده دستور Merge به شرح زیر است:

```
MERGE dbo.FactSalesQuota AS T
USING SSIS_PDS.dbo.stgFactSalesQuota AS S
ON T.EmployeeKey = S.EmployeeKey
AND T.DateKey = S.DateKey
WHEN MATCHED AND BY target
THEN INSERT(EmployeeKey, DateKey, CalendarYear, CalendarQuarter, SalesAmountQuota)
VALUES(S.EmployeeKey, S.DateKey, S.CalendarYear, S.CalendarQuarter, S.SalesAmountQuota)
WHEN MATCHED AND T.SalesAmountQuota != S.SalesAmountQuota
THEN UPDATE SET T.SalesAmountQuota = S.SalesAmountQuota
;
```

اشکال این روش Performance است؛ گرچه این دستور به سادگی عملیات درج و بروزرسانی را انجام می‌دهد ولی به صورت سطر به سطر عملیات انجام می‌شود (در هر زمان یک سطر). در موقعیت هایی که با مقدار زیادی داده مواجه هستید، اغلب بهتر است به صورت انبوه عملیات درج و به صورت مجموعه عملیات بروزرسانی انجام گیرد.

Managing Inferred Members 4-3-3-

زمانیکه یک ارجاع در جدول Fact به یک عضو Dimension که هنوز بارگذاری نشده‌است بوجود آید؛ یک Inferred Member تعبیر می‌شود. به سه طریق می‌توان این Inferred Memberها را مدیریت نمود:

رکوردهای جدول Fact پیش از درج اسکن شوند؛ ایجاد هر Inferred Member در Dimension و سپس بارگذاری رکوردها در جدول Fact

در طول عملیات بارگذاری روی Fact؛ هر رکورد مفقوده شده به یک جدول موقتی ارسال شود، رکوردهای مفقوده شده به Dimension اضافه شود، در ادامه مجدداً آن رکوردهای Fact در جدول Fact بارگذاری شوند.

در یک Data Flow زمانی که یک رکورد مفقود شده، بلاتکلیف تعبیر می‌شود؛ آن زمان یک رکورد به Dimension اضافه شود و

Surrogate Key بدست آمده را برگردانیم؛ سپس Dimension بارگذاری شود.

شکل زیر این موارد را نمایش می‌دهد:

عنوان: **بهبود عملکرد SQL Server Locks در سیستم‌های با تعداد تراکنش بالا در Entity Framework**
نویسنده: یاسر مرادی
تاریخ: ۱۳۹۳/۰۴/۲۷ ۲۳:۵۰
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, SQL Server, Performance, transaction, Entity Framework 6

بر اساس رفتار پیش فرض در دیتابیس SQL Server، در زمان انجام دادن یک دستور که منجر به ایجاد تغییرات در اطلاعات موجود در جدول می‌شود (برای مثال دستور Update)، جدول مربوطه به صورت کامل Lock می‌شود، ولو آن دستور Update، فقط با یکی از رکوردهای آن جدول کار داشته باشد.

در سیستم‌های با تعداد تراکنش بالا و دارای تعداد زیاد کلاینت، این رفتار پیش فرض موجب ایجاد صفی از تراکنش‌های در حال انتظار بر روی جداولی می‌شود که ویرایش‌های زیادی بر روی آنها رخ می‌دهد.
اگر چه که بنظر این مشکل راه حل‌های زیادی دارد ، لکن آن راه حلی که همیشه موثر عمل می‌کند استفاده از SQL Server Table Hints است.
SQL Server Table Hints به تمامی آن دستوراتی گفته می‌شود که هنگام اجرای دستور اصلی (برای مثال Select و یا Update) رفتار پیش فرض SQL Server را بر اساس Hint ارائه شده تغییر می‌دهند.
لیست کامل این Hintها را می‌توانید در اینجا مشاهده کنید .
Hint ای که در اینجا برای ما مفید است، آن است که به SQL Server بگوییم هنگام اجرای دستور Update، به جای Lock کردن کل جدول، فقط رکورد در حال ویرایش را Lock کند، و این باعث می‌شود تا باقی تراکنش ها، که ای بسا با سایر رکوردهای آن جدول کار داشته باشند متوقف نشوند، که البته این مسئله کمی به افزایش مصرف حافظه می‌انجامد، لکن مقدار افزایش بسیار ناچیز است.
این Hint که rowlock نام دارد در تراکنش‌های با Isolation Level تنظیم شده بر روی Snapshot باید با یک Table Hint دیگر با نام updlock ترکیب شود.

توضیحات مفصل‌تر این دو Hint در لینک مربوطه آمده است.
بنابر این، بجای دستور

```
update products
set Name = "Test"
Where Id = 1
```

داریم

```
update products with (nolock,updlock)
set Name = "Test"
where Id = 1
```

تا اینجا مشکل خاصی وجود ندارد، آنچه که از اینجا به بعد اهمیت دارد این است که در هنگام کار با Entity Framework، اساسا ما نویسنده دستورات Update نیستیم که به آنها Hint اضافه کنیم یا نه، بلکه دستورات SQL بوسیله Entity Framework ایجاد می‌شوند.
در Entity Framework، مکانیزمی تعبیه شده است با نام Db Command Interceptor که به شما اجازه می‌دهد دستورات SQL ساخته شده را Log کنید و یا قبل از اجرا تغییر دهید ، که برای اضافه نمودن Table Hintها ما از این روش استفاده می‌کنیم، برای انجام این کار داریم: (توضیحات در ادامه)

```
public class UpdateRowLockHintDbCommandInterceptor : IDbCommandInterceptor
    {
        public void NonQueryExecuting(DbCommand command, DbCommandInterceptionContext<Int32>
interceptionContext)
        {
            if (command.CommandType != CommandType.Text) return; // (1)
            if (!(command is SqlCommand)) return; // (2)
            SqlCommand sqlCommand = (SqlCommand)command;
            String commandText = sqlCommand.CommandText;
            String updateCommandRegularExpression = "(update) ";
```

```
            Boolean isUpdateCommand = Regex.IsMatch(commandText, updateCommandRegularExpression,
RegexOptions.IgnoreCase | RegexOptions.Multiline); // You may use better regular expression pattern
here.
            if (isUpdateCommand)
            {
                Boolean isSnapshotIsolationTransaction = sqlCommand.Transaction != null &&
sqlCommand.Transaction.IsolationLevel == IsolationLevel.Snapshot;
                String tableHintToAdd = isSnapshotIsolationTransaction ? " with (rowlock , updlock) set
" : " with (rowlock) set ";
                commandText = Regex.Replace(commandText, "^(set) ", (match) =>
                {
                    return tableHintToAdd;
                }, RegexOptions.IgnoreCase | RegexOptions.Multiline);
                command.CommandText = commandText;
            }
        }
```

این کد در قسمت (1) ابتدا تشخیص می‌دهد که آیا این یک Command دارای Command Text است یا خیر، برای مثال اگر فراخوانی یک Stored Procedure است، ما با آن کاری نداریم.

در قسمت دوم تشخیص می‌دهیم که آیا با SQL Server در حال تعامل هستیم، یا برای مثال با Oracle و ...، که ما برای Table Hintها فقط با SQL Server کار داریم.

سپس باید تشخیص دهیم که آیا این یک دستور update است یا خیر ؟ برای این منظور از Regular Expressionها استفاده کرده ایم، که خیلی به بحث آموزش این پست مربوط نیست، به صورت کلی از Regular Expressionها برای یافتن و بررسی و جایگزینی عبارات با قاعده در هنگام کار با رشته‌ها استفاده می‌شود.

ممکن است Regular Expression ای که شما می‌نویسید بسیار بهتر از این نمونه باشد، که در این صورت خوشحال می‌شوم در قسمت نظرات آنرا قرار دهید.

در نهایت با بررسی Transaction Isolation Level مربوطه که Snapshot است یا خیر، به درج یک یا هر دو Table Hint مربوطه اقدام می‌نماییم.

عنوان: SQL Antipattern #1
نویسنده: سید مرتضی حسینی
تاریخ: ۱۳۹۳/۰۴/۲۹ ۱۳:۲۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, SQL, Antipatterns

در این سلسله نوشتار قصد دارم ترجمه و خلاصه چندین فصل از کتاب ارزشمند (SQL Antipatterns: Avoiding the Pitfalls of Database Programming (Pragmatic Programmers که حاصل تلاش گروه IT موسسه هدایت فرهیختگان جوان می‌باشد، را ارائه نمایم.

بخش اول : Jaywalking

در این بخش در حال توسعه ویژگی نرم افزاری هستیم که در آن هرکاربر به عنوان یک کاربر اصلی برای یک محصول تخصیص داده می‌شود. در طراحی اصلی ما فقط اجازه می دهیم یک کاربر متعلق به هر محصول باشد، اما امکان چنین تقاضایی وجود دارد

که چند کاربر نیز به یک محصول اختصاص داده شوند.

در این صورت، اگر پایگاه داده را به نحوی تغییر دهیم که شناسه‌ی حساب کاربران را در لیستی که با کاما از یکدیگر جدا شده‌اند ذخیره نماییم، خیلی ساده‌تر به نظر می‌رسد نسبت به اینکه بصورت جداگانه آنها را ثبت نماییم.

برنامه نویسان معمولا برای جلوگیری از ایجاد جدول واسطی [1] که رابطه‌های چند به چند زیادی دارد از یک لیست که با کاما داده‌هایش از هم جدا شده‌اند، استفاده می‌کنند. بدین جهت اسم این بخش antipatten, jaywalking می‌باشد، زیرا jaywalking نیز عملیاتی است که از تقاطع جلوگیری می‌کند.

1.1 هدف: ذخیره کردن چندین صفت

طراحی کردن جدولی که ستون آن فقط یک مقدار دارد، بسیار ساده و آسان می‌باشد. شما می‌توانید نوع داده‌ایی که متعلق به ستون می‌باشد را انتخاب نمایید. مثلا از نوع (int,date...)

چگونه می‌توانیم مجموعه‌ایی از مقادیری که به یکدیگر مرتبط هستند را در یک ستون ذخیره نماییم ؟

در مثال ردیابی خطای پایگاه داده، ما یک محصول را به یک کاربر، با استفاده از ستونی که نوع آن integer است، مرتبط می‌نماییم. هر کاربر ممکن است محصولاتی داشته باشد و هر محصول به یک contact اشاره کند. بنابراین یک ارتباط چند به یک بین محصولات و کاربر برقرار است. برعکس این موضوع نیز صادق است؛ یعنی امکان دارد هر محصول متعلق به چندین کاربر باشد و یک ارتباط یک به چند ایجاد شود. در زیر جدول محصولات را به صورت عادی آورده شده است:

```
CREATE TABLE Products (
product_id SERIAL PRIMARY KEY,
product_name VARCHAR(1000),
account_id BIGINT UNSIGNED,
-- . . .
FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
);

INSERT INTO Products (product_id, product_name, account_id)
VALUES (DEFAULT, 'Visual TurboBuilder' , 12);
```

2.1 Antipattern : قالب بندی لیست هایی که با کاما از یکدیگر جدا شده اند

برای اینکه تغییرات در پایگاه داده را به حداقل برسانید، می‌توانید نوع شناسه‌ی کاربر را از نوع varchar قرار دهید. در این صورت می‌توانید چندین شناسه‌ی کاربر که با کاما از یکدیگر جدا شده‌اند را در یک ستون ذخیره نمایید. پس در جدول محصولات، شناسه‌ی کاربران را از نوع varchar قرار می‌دهیم.

```
CREATE TABLE Products (
product_id SERIAL PRIMARY KEY,
product_name VARCHAR(1000),
account_id VARCHAR(100), -- comma-separated list
-- . . .
);

INSERT INTO Products (product_id, product_name, account_id)
VALUES (DEFAULT, 'Visual TurboBuilder' , '12,34' );
```

اکنون مشکلات کارایی و جامعیت داده‌ها را در این راه حل پیشنهادی بررسی می‌نماییم .

بدست آوردن محصولاتی برای یک کاربر خاص

بدلیل اینکه تمامی شناسه‌ی کاربران که بصورت کلید خارجی جدول Products می‌باشند به صورت رشته در یک فیلد ذخیره

شده‌اند و حالت ایندکس بودن آنها از دست رفته است، بدست آوردن محصولاتی برای یک کاربر خاص سخت می‌باشد. به عنوان مثال بدست آوردن محصولاتی که کاربری با شناسه‌ی 12 خریداری نموده به‌صورت زیر می‌باشد:

```
SELECT * FROM Products WHERE account_id REGEXP '[[:<:]]12[[:>:]]' ;
```

بدست آوردن کاربرانی که یک محصول خاص خریداری نموده اند

Join کردن account_id با Accounts table یعنی جدول مرجع نامناسب و غیر معمول می‌باشد. مساله مهمی که وجود دارد این است که زمانیکه بدین صورت شناسه‌ی کاربران را ذخیره می‌نماییم، ایندکس بودن آنها از بین می‌رود. کد زیر کاربرانی که محصولی با شناسه‌ی 123 خریداری کرده‌اند را محاسبه می‌کند.

```
SELECT * FROM Products AS p JOIN Accounts AS a
ON p.account_id REGEXP '[[:<:]]' || a.account_id || '[[:>:]]'
WHERE p.product_id = 123;
```

ایجاد توابع تجمیعی [2]

مبنای چنین توابعی از قبیل sum(), count(), avg() بدین صورت می‌باشد که بر روی گروهی از سطرها اعمال می‌شوند. با این حال با کمک ترفندهایی می‌توان برخی از این توابع را تولید نماییم هر چند برای همه آن‌ها این مساله صادق نمی‌باشد. اگر بخواهیم تعداد کاربران برای هر محصول را بدست بیاوریم ابتدا باید طول لیست شناسه‌ی کاربران را محاسبه کنیم، سپس کاما را از این لیست حذف کرده و طول جدید را از طول قبلی کم کرده و با یک جمع کنیم. نمونه کد آن در زیر آورده شده است:

```
SELECT product_id, LENGTH(account_id) - LENGTH(REPLACE(account_id, ',' , '' )) + 1
AS contacts_per_product
FROM Products;
```

ویرایش کاربرانی که یک محصول خاص خریداری نموده‌اند

ما به راحتی می‌توانیم یک شناسه‌ی جدید را به انتهای این رشته اضافه نماییم؛ فقط مرتب بودن آن را بهم میریزیم. در نمونه اسکریپت زیر گفته شده که در جدول محصولات برای محصولی با شناسه‌ی 123 بعد از کاما یک کاربر با شناسه‌ی 56 درج شود:

```
UPDATE Products
SET account_id = account_id || ',' || 56
WHERE product_id = 123;
```

برای حذف یک کاربر از لیست کافیست دو دستور sql را اجرا کرد: اولی برای fetch کردن شناسه‌ی کاربر از لیست و بعدی برای ذخیره کردن لیست ویرایش شده. بطور مثال تمامی افرادی که محصولی با شناسه‌ی 123 را خریداری کرده‌اند، از جدول محصولات انتخاب می‌کنیم. برای حذف کاربری با شناسه‌ی 34 از لیست کاربران، بر حسب کاما مقادیر لیست را در آرایه می‌ریزیم، شناسه‌ی مورد نظر را جستجو می‌کنیم و آن را حذف می‌کنیم. دوباره مقادیر درون آرایه را بصورت رشته درمیاوریم و جدول محصولات را با لیست جدید کاربران برای محصولی با شناسه‌ی 123 بروزرسانی می‌کنیم.

```
<?php
$stmt = $pdo->query(
"SELECT account_id FROM Products WHERE product_id = 123");
$row = $stmt->fetch();
$contact_list = $row['account_id' ];


// change list in PHP code
$value_to_remove = "34";
$contact_list = split(",", $contact_list);
$key_to_remove = array_search($value_to_remove, $contact_list);
unset($contact_list[$key_to_remove]);
$contact_list = join(",", $contact_list);
$stmt = $pdo->prepare(
"UPDATE Products SET account_id = ?
WHERE product_id = 123");
$stmt->execute(array($contact_list));
```

اعتبارسنجی شناسه‌ی محصولات

به دلیل آنکه نوع فیلد account_id،varchar می‌باشد احتمال این وجود دارد داده‌ای نامعتبر وارد نماییم چون پایگاه داده‌ها هیچ عکس العملی یا خطایی را نشان نمی‌دهد، فقط از لحاظ معنایی دچار مشکل می‌شویم. در نمونه زیر banana یک داده‌ی غیر معتبر می‌باشد و ارتباطی با شناسه‌ی کاربران ندارد.

```
INSERT INTO Products (product_id, product_name, account_id)
VALUES (DEFAULT, 'Visual TurboBuilder' , '12,34,banana' );
```

انتخاب کردن کاراکتر جداکننده

نکته قابل توجه این است که کاراکتری که بعنوان جداکننده در نظر می‌گیریم باید در هیچکدام از داده‌های ورودی ما امکان بودنش وجود نداشته باشد .

محدودیت طول لیست

در زمان طراحی جدول، برای هر یک از فیلدها باید حداکثر طولی را قرار دهیم. به عنوان نمونه ما اگر varchar(30) در نظر بگیریم بسته به تعداد کاراکترهایی که داده‌های ورودی ما دارند می‌توانیم جدول را پرنماییم. اسکریپت زیر شناسه‌ی کاربرانی که محصولی با شناسه‌ی 123 را خریده‌اند، ویرایش می‌کند. با این تفاوت که با توجه به محدودیت لیست، در نمونه‌ی اولی شناسه‌ی کاربران دو کاراکتری بوده و 10داده بعنوان ورودی پذیرفته است در حالیکه در نمونه‌ی دوم بخاطر اینکه شناسه‌ی کاربران شش کاراکتری می‌باشد فقط 4 شناسه می‌توانیم وارد نماییم.

```
UPDATE Products SET account_id = '10,14,18,22,26,30,34,38,42,46'
WHERE product_id = 123;
```

```
UPDATE Products SET account_id = '101418,222630,343842,467790'
WHERE product_id = 123;
```

3.1 موارد تشخیص این Antipattern:

سؤالات زیر نشان می‌دهند که Jaywalking antipattern مورد استفاده قرار گرفته است:

- حداکثر پذیرش این لیست برای داده ورودی چه میزان است؟
- چه کاراکتری هرگز در داده‌های ورودی این لیست ظاهر نمی‌شود؟ **4.1 مواردی که استفاده از این Antipattern مجاز است:**

دی نرمال کردن کارایی را افزایش می‌دهد. ذخیره کردن شناسه‌ها در یک لیست که با کاما از یکدیگر جدا شده‌اند نمونه بارزی از دی نرمال کردن می‌باشد. ما زمانی می‌توانیم از این مدل استفاده نماییم که به جدا کردن شناسه‌ها از هم نیاز نداشته باشیم. به همین ترتیب، اگر در برنامه، شما یک لیست را از یک منبع دیگر با این فرمت دریافت نمایید، به سادگی آن را در پایگاه داده خود به همان فرمت بصورت کامل می‌توانید ذخیره و بازیابی نمایید و احتیاجی به مقادیر جداگانه ندارید. درهنگام استفاده از پایگاه داده‌های دی‌نرمال دقت کنید. برای شروع از پایگاه داده نرمال استفاده کنید زیرا به کدهای برنامه شما امکان انعطاف بیشتری می‌دهد و کمک می‌کند تا پایگاه داده‌ها تمامیت داده(Data integrity) را حفظ کند. **5.1 راه حل: استفاده کردن از جدول واسط**

در این روش شناسه‌ی کاربران را در جدول جداگانهایی که هر کدام از آنها یک سطر را به خود اختصاص داده اند، ذخیره می‌نماییم.

```
CREATE TABLE Contacts ( product_id BIGINT UNSIGNED NOT NULL,
account_id BIGINT UNSIGNED NOT NULL,
PRIMARY KEY (product_id, account_id),
FOREIGN KEY (product_id) REFERENCES Products(product_id),
FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
);
```

جدول Contacts یک جدول رابطه ایی بین جداول Products,Accounts

بدست آوردن محصولات برای کاربران و موارد مربوط به آن

ما براحتی می‌توانیم تمامی محصولاتی که مختص به یک کاربر هستند را بدست آوریم. در این شیوه خاصیت ایندکس بودن شناسه‌ی کاربران حفظ می‌شود به همین دلیل queryهای آن برای خواندن و بهینه کردن راحت‌تر می‌باشند. در این روش به کاراکتری برای جدا کردن ورودی‌ها از یکدیگر نیاز نداریم چون هر کدام از آنها در یک سطر جداگانه ثبت می‌شوند. برای ویرایش کردن کاربرانی که یک محصول را خریداری نموده اند، کافیست یک سطر از جدول واسط را اضافه یا حذف نماییم. درنمونه کد زیر، ابتدا در جدول Contacts کاربری با شناسه‌ی 34 که محصولی با شناسه‌ی 456 را خریداری کرده، درج شده است و در خط بعد عملیات حذف با شرط آنکه شناسه‌ی کاربر و محصول به ترتیب 34،456 باشد روی جدول Contacts اعمال شده است.

```
INSERT INTO Contacts (product_id, account_id) VALUES (456, 34);

DELETE FROM Contacts WHERE product_id = 456 AND account_id = 34;
```

ایجاد توابع تجمیعی

به عنوان نمونه در مثال زیر براحتی ما می‌توانیم تعداد محصولات در هر حساب کاربری را بدست آوریم:

```
SELECT account_id, COUNT(*) AS products_per_account
FROM Contacts
GROUP BY account_id;
```

اعتبارسنجی شناسه محصولات

از آنجاییکه مقادیری که در جدول قرار دارند کلید خارجی می‌باشند میتوان صحت اعتبار آنها را بررسی نمود. بعنوان مثال Contacts.account_id به Account.account_id اشاره می‌کند. در ضمن برای هر فیلد نوع آن را می‌توان مشخص کرد تا فقط همان نوع داده را بپذیرد.

محدودیت طول لیست

نسبت به روش قبلی تقریبا در این حالت محدودیتی برای تعداد کاراکترهای ورودی نداریم.

مزیت‌های دیگر استفاده از جدول واسط

کارایی روش دوم بهتر از حالت قبلی می‌باشد چون ایندکس بودن شناسه‌ها حفظ شده است. همچنین براحتی میتوانیم فیلدی را به این جدول اضافه نماییم مثلا (time, date...)

Intersection Table [1]

Aggregate [2]

نظرات خوانندگان

نویسنده: م منفرد
تاریخ: ۱۳۹۳/۰۴/۳۰ ۰:۴۸

سلام
منظورتان از گروه it موسسه هدایت فرهیختگان چه بود؟ ایشان کتاب را ترجمه کردند؟ چگونه می‌توان آن را تهیه کرد؟
با تشکر

نویسنده: فرید
تاریخ: ۱۳۹۳/۰۶/۲۶ ۲۰:۱۸

سلام؛ به نظر میرسه این خصیصه مربوط به mysql است. من نتوانستم آن‌را در sqlserver اجرا کنم.

بخش دوم : Naive Trees

فرض کنید یک وب سایت حرفه‌ای خبری یا علمی-پژوهشی داریم که قابلیت دریافت نظرات کاربران را در مورد هر مطلب مندرج در سایت یا نظرات داده شده در مورد آن مطالب را دارا می‌باشد. یعنی هر کاربر علاوه بر توانایی اظهار نظر در مورد یک خبر یا مطلب باید بتواند پاسخ نظرات کاربران دیگر را نیز بدهد. اولین راه حلی که برای طراحی این مطلب در پایگاه داده به ذهن ما می‌رسد، ایجاد یک زنجیره با استفاده از کد sql زیر می‌باشد:

```
CREATE TABLE Comments (
comment_idSERIAL PRIMARY KEY,
parent_idBIGINT UNSIGNED,
comment TEXT NOT NULL,
FOREIGN KEY (parent_id) REFERENCES Comments(comment_id)
);
```

البته همان طور که پیداست بازیابی زنجیره‌ای از پاسخ‌ها در یک پرس‌وجوی sql کار سختی است. این نخ‌ها معمولا عمق نامحدودی دارند و برای به دست آوردن تمام نخ‌های یک زنجیره باید پرس‌وجوهای زیادی را اجرا نمود.

ایده‌ی دیگر می‌تواند بازیابی تمام نظرها و ذخیره‌ی آن‌ها در حافظه‌ی برنامه به صورت درخت باشد. ولی این روش برای ذخیره هزاران نظری که ممکن است در سایت ثبت شود و علاوه بر آن مقالات جدیدی که اضافه می‌شوند، تقریبا غیرعملی است.

1.2 هدف: ذخیره و ایجاد پرس‌وجو در سلسله‌مراتب

وجود سلسله مراتب بین داده‌ها امری عادی محسوب می‌گردد. در ساختار داده‌ای درختی هر ورودی یک گره محسوب می‌گردد. یک گره ممکن است تعدادی فرزند و یک پدر داشته باشد. گره اول پدر ندارد، ریشه و گره فرزند که فرزند ندارد، برگ و گره‌ای دیگر، گره‌های غیربرگ نامیده می‌شوند.

مثال‌هایی که از ساختار درختی داده‌ها وجود دارد شامل موارد زیر است:

Organizational chart : در این ساختار برای مثال در ارتباط بین کارمندان و مدیر، هر کارمند یک مدیر دارد که نشان‌دهنده‌ی پدر یک کارمند در ساختار درختی است. هر مدیر هم یک کارمند محسوب می‌گردد.

Threaded discussion : در این ساختار برای مثال در سیستم نظردهی و پاسخ‌ها، ممکن است زنجیره‌ای از نظرات در پاسخ به نظرات دیگر استفاده گردد. در درخت، فرزندان یک گره‌ی نظر، پاسخ‌های آن گره هستند.

در این فصل ما از مثال ساختار دوم برای نشان دادن Antipattern و راه حل آن بهره می‌گیریم.

2.2 Antipattern : همیشه مبتنی بر یکی از والدین

راه حل ابتدایی و ناکارآمد

اضافه نمودن ستون parent_id . این ستون، به ستون نظر در همان جدول ارجاع داده می‌شود و شما می‌توانید برای اجرای این رابطه از قید کلید خارجی استفاده نمایید. پرس‌وجویی که برای ساخت مثالی که ما در این بحث از آن استفاده می‌کنیم در ادامه آمده است:

```
CREATE TABLE Comments (  comment_idSERIAL PRIMARY KEY,
parent_idBIGINT UNSIGNED,
bug_idBIGINT UNSIGNED NOT NULL,
author BIGINT UNSIGNED NOT NULL,
comment_dateDATETIME NOT NULL,
comment TEXT NOT NULL,
FOREIGN KEY (parent_id)REFERENCES Comments(comment_id),
FOREIGN KEY (bug_id)         REFERENCES Bugs(bug_id),
FOREIGN KEY(author)          REFERENCES Accounts(account_id)
);
```

مثالی از پرس‌وجوی فوق را می‌توانید در زیر ببینید:

comment_id	parent_id	author	comment
1	NULL	Fran	What's the cause of this bug?
2	1	Ollie	I think it's a null pointer.
3	2	Fran	No, I checked for that.
4	1	Kukla	We need to check for invalid input.
5	4	Ollie	Yes, that's a bug.
6	4	Fran	Yes, please add a check.
7	6	Kukla	That fixed it.

لیست مجاورت :

لیست مجاورت خود می‌تواند به عنوان یک antipattern در نظر گرفته شود. البته این مطلب از آنجایی نشأت می‌گیرد که این روش توسط بسیاری از توسعه‌دهندگان مورد استفاده قرار می‌گیرد ولی نتوانسته است به عنوان راه حل برای معمول‌ترین وظیفه‌ی خود، یعنی ایجاد پرس‌وجو بر روی کلیه فرزندان، باشد.

• با استفاده از پرس‌وجوی زیر می‌توان فرزند بلافاصله‌ی یک "نظر" را برگرداند:

```
SELECT c1.*, c2.*
FROM Comments c1 LEFT OUTER JOIN Comments c2
ON c2.parent_id = c1.comment_id;
```

ضعف پرس‌وجوی فوق این است که فقط می‌تواند دو سطح از درخت را برای شما برگرداند. در حالیکه خاصیت درخت این است که شما را قادر می‌سازد بدون هیچ گونه محدودیتی فرزندان و نوه‌های متعدد (سطوح بی‌شمار) برای درخت خود تعریف کنید.
بنابراین به ازای هر سطح اضافه باید یک join به پرس‌جوی خود اضافه نمایید. برای مثال اگر پرس‌وجوی زیر می‌تواند درختی با چهار سطح برای شما برگرداند ولی نه بیش از آن:

```
SELECT c1.*, c2.*, c3.*, c4.*
FROM Comments c1                          -- 1st level
LEFT OUTER JOIN Comments c2
ON c2.parent_id = c1.comment_id  -- 2nd level
LEFT OUTER JOIN Comments c3
ON c3.parent_id = c2.comment_id  -- 3rd level
LEFT OUTER JOIN Comments c4
ON c4.parent_id = c3.comment_id; -- 4th level
```

این پرس‌وجو به این دلیل که با اضافه شدن ستون‌های دیگر، نوه‌ها را سطوح عمیق‌تری برمی‌گرداند، پرس‌وجوی مناسبی نیست.

در واقع استفاده از توابع تجمیعی ، مانند COUNT() مشکل می‌شود.

راه دیگر برای به دست آوردن ساختار یک زیردرخت از لیست مجاورت برای یک برنامه، این است که سطرهای مورد نظر خود را از مجموعه بازیابی نموده و سلسه‌مراتب مورد نظر را در حافظه بازیابی نماییم و از آن به عنوان درخت استفاده نماییم:

```
SELECT * FROM Comments WHERE bug_id = 1234;
```

نگهداری کردن یک درخت با استفاده از لیست مجاورت

البته برخی از عملکردها با لیست مجاورت به خوبی انجام می‌گیرد. برای مثال اضافه نمودن یک گره (نظر)، مکان‌یابی مجدد برای یک گره یا یک زیردرخت .

```
INSERT INTO Comments (bug_id, parent_id, author, comment)
VALUES (1234, 7, 'Kukla' , 'Thanks!' );
```

بازیابی دوباره مکان یک نود یا یک زیردرخت نیز آسان است:

```
UPDATE Comments SET parent_id = 3 WHERE comment_id = 6;
```

با این حال حذف یک گره از یک درخت در این روش پیچیده است. اگر بخواهیم یک زیردرخت را حذف کنید باید چندین پرس‌وجو برای پیدا کردن تمام نوه‌ها بنویسیم و سپس حذف نوه‌ها را از پایین‌ترین سطح شروع کرده و تا جایی که قید کلید خارجی برقرار شود ادامه دهیم. البته می‌توان از کلید خارجی با تنظیم ON DELETE CASCADE استفاده کرد تا این کارها به طور خودکار انجام گیرد.
حال اگر بخواهیم یک نود غیر برگ را حذف کرده یا فرزندان آن را در درخت جابجا کنیم، ابتدا باید parent_id فرزندان آن نود را تغییر داده و سپس نود مورد نظر را حذف می‌کنیم:

```
SELECT parent_id FROM Comments WHERE comment_id = 6; -- returns 4
UPDATE Comments SET parent_id = 4 WHERE parent_id = 6;
DELETE FROM Comments WHERE comment_id = 6;
```

3.2 موارد تشخیص این Antipattern:

سؤالات زیر نشان می‌دهند که Naive Trees antipattern مورد استفاده قرار گرفته است:

چه تعداد سطح برای پشتیبانی در درخت نیاز خواهیم داشت؟

من همیشه از کار با کدی که ساختار داده‌ی درختی را مدیریت می‌کند، می‌ترسم

من باید اسکریپتی را به طور دوره‌ای اجرا نمایم تا سطرهای یتیم موجود در درخت را حذف کند.

4.2 مواردی که استفاده از این Antipattern مجاز است:

قدرت لیست مجاورت در بازیابی پدر یا فرزند مستقیم یک نود می‌باشد. قرار دادن یک سطر هم در لیست مجاورت کار ساده‌ای است. اگر این عملیات، تمام آن چیزی است که برای انجام کارتان مورد نیاز شما است، بنابراین استفاده از لیست مجاورت می‌تواند مناسب باشد.
برخی از برندهای RDBMS از افزونه‌هایی پشتیبانی می‌کنند که قابلیت ذخیره‌ی سلسله مراتب را در لیست مجاورت ممکن می‌سازد. مثلا SQL-99، پرس‌وجوی بازگشتی را تعریف می‌کند که مثال آن در ادامه آمده است:

```
WITH CommentTree (comment_id, bug_id, parent_id, author, comment, depth)
AS (
SELECT *, 0 AS depth FROM Comments
WHERE parent_id IS NULL
UNION ALL
SELECT c.*, ct.depth+1 AS depth FROM CommentTreect
JOIN Comments c ON (ct.comment_id = c.parent_id)
)
SELECT * FROM CommentTree WHERE bug_id = 1234;
```

Oracle 9i، Oracle 11g، IBM DB2، Microsoft SQL Server 2005 و PostgreSQL 8.4 نیز از پرس‌وجوی بازگشتی پشتیبانی می‌کنند. و 10g از عبارت WITH استفاده می‌کنند، ولی نه برای پرس‌وجوهای بازگشتی. در عوض می‌توانید از پرس‌وجوی زیر برای ایجاد پرس‌وجوی بازگشتی استفاده نمایید:

```
SELECT * FROM Comments
START WITH comment_id = 9876
CONNECT BY PRIOR parent_id = comment_id;
```

5.2 راه حل: استفاده از مدل‌های درختی دیگر

جایگزین‌های دیگری برای ذخیره‌سازی داده‌های سلسله مراتبی وجود دارد. البته برخی از این راه حل‌ها ممکن است در لحظه‌ی اول پیچیدتر از لیست مجاورت به نظر آیند، ولی برخی از عملیات درخت که در لیست مجاورت بسیار سخت یا ناکارآمد است، را آسان‌تر می‌کنند. **شمارش مسیر :**

مشکل پرهزینه بودن بازیابی نیاکان یک گره که در روش لیست مجاورت وجود داشت در روش شمارش مسیر به این ترتیب حل شده است: اضافه نمودن یک صفت به هر گره که رشته‌ای از نیکان آن صفت در آن ذخیره شده است.

در جدول Comments به جای استفاده از parent_id، یک ستون به نام path که توع آن varchar است تعریف شده است. رشته‌ای که در این ستون تعریف شده است، ترتیبی از فرزندان این سطر از بالا به پایین درخت است. مانند مسیری که در سیستم عامل UNIX، برای نشان دادن مسیر در سیستم فایل استفاده شده است. شما می‌توانید از / به عنوان کاراکتر جداکننده استفاده نمایید. دقت کنید برای درست کار کردن پرس‌وجوها حتما در آخر مسیر هم این کاراکتر را قرار دهید. پرس‌وجوی تشکیل چنین درختی به شکل زیر است:

```
CREATE TABLE Comments ( comment_id SERIAL PRIMARY KEY,
path VARCHAR(1000),
bug_id BIGINT UNSIGNED NOT NULL,
author BIGINT UNSIGNED NOT NULL,
comment_date DATETIME NOT NULL,
comment TEXT NOT NULL,
FOREIGN KEY (bug_id) REFERENCES Bugs(bug_id),
FOREIGN KEY (author) REFERENCES Accounts(account_id)
```

comment_id	path	author	comment
1	1/	Fran	What's the cause of this bug?
2	1/2/	Ollie	I think it's a null pointer.
3	1/2/3/	Fran	No, I checked for that.
4	1/4/	Kukla	We need to check for invalid input.
5	1/4/5/	Ollie	Yes, that's a bug.
6	1/4/6/	Fran	Yes, please add a check.
7	1/4/6/7/	Kukla	That fixed it.

در این روش، هر گره مسیری دارد که شماره خود آن گره هم در آنتهای آن مسیر قرار دارد. این به دلیل درست جواب دادن پرس‌وجوهای ایجاد شده است.

می‌توان نیاکان را با مقایسه‌ی مسیر سطر کنونی با مسیر سطر دیگر به دست آورد. برای مثال برای یافتن نیاکان گره (نظر) شماره‌ی 7 که مسیر آن 7/6/4/1/ می‌باشد، می‌توان چنین نوشت:

```
SELECT * FROM Comments AS c
WHERE '1/4/6/7/' LIKE c.path || '%' ;
```

این پرس‌وجو الگوهایی را می‌یابد که از مسیرهای 1/6/4/%، 1/4/% و 1/% نشأت می‌گیرد.
همچنین فرزندان (نوه‌های) یک گره، مثلا گره‌ی 4 را که مسیرش 1/4/ است را می‌توان با پرس‌وجوی زیر یافت:

```
SELECT * FROM Comments AS c
WHERE c.path LIKE '1/4/' || '%' ;
```

الگوی 1/4/% با مسیرهای 1/4/5/، 1/4/6/ و 1/4/6/7/ تطابق می‌یابد.
همچنین می‌توان پرس‌وجوهای دیگری را نیز در این مسیر به سادگی انجام داد؛ مانند محاسبه‌ی مجموع هزینه‌ی گره‌ها در یک زیردرخت یا شمارش تعداد گره‌ها.
اضافه نمودن یک گره هم مانند ساختن خود مدل است. می‌توان یک گره‌ی غیر برگ را بدون نیاز به اصلاح هیچ سطری اضافه نمود. برای این کار مسیر را را از گره‌ی پدر کپی کرده و در انتها شماره‌ی خود گره را به آن اضافه می‌کنیم.
از مشکلات این روش می‌توان به عدم توانایی پایگاه داده‌ها در تحمیل این نکته که مسیر یک گره درست ایجاد شده است و یا تضمین وجود گره‌ای در مسیری خاص، اشاره نمود. همچنین نگهداری رشته‌ی مسیر یک گره مبتنی بر کد برنامه است و اعتبارسنجی آن کاری هزینه‌بر است. این رشته اندازه‌ای محدود دارد و درخت‌هایی با عمق نامحدود را پشتیبانی نمی‌کند.

مجموعه‌های تودرتو :

مجموعه‌های تودرتو، اطلاعات را با هر گره‌ای که مربوط به مجموعه‌ای از نوه‌هایش است، به جای این که تنها مربوط به یک فرزند بلافصلش باشد، ذخیره می‌کنند.

این اطلاعات می‌توانند به وسیله‌ی هر گره‌ای که در درخت با دو شماره‌ی nsleft و nsright ذخیره شده، نمایش داده شوند:

```
CREATE TABLE Comments ( comment_id SERIAL PRIMARY KEY,
nsleft INTEGER NOT NULL,
nsright INTEGER NOT NULL,
bug_id BIGINT UNSIGNED NOT NULL,
author BIGINT UNSIGNED NOT NULL,
comment_date DATETIME NOT NULL,
comment TEXT NOT NULL,
FOREIGN KEY (bug_id) REFERENCES Bugs (bug_id),
FOREIGN KEY (author) REFERENCES Accounts(account_id)
);
```

comment_id	nsleft	nsright	author	comment
1	1	14	Fran	What's the cause of this bug?
2	2	5	Ollie	I think it's a null pointer.
3	3	4	Fran	No, I checked for that.
4	6	13	Kukla	We need to check for invalid input.
5	7	8	Ollie	Yes, that's a bug.
6	9	12	Fran	Yes, please add a check.
7	10	11	Kukla	That fixed it.

شماره‌ی سمت چپ یک گره از تمام شماره‌های سمت چپ فرزندان آن گره کوچک‌تر و شماره‌ی سمت راست آن گره از تمام شماره‌های سمت راست آن گره بزرگ‌تر است. این شماره‌ها هیچ ارتباطی به comment_id مربوط به آن گره ندارند.

یک راه حل ساده برای تخصیص این شماره‌ها به گره‌ها این است که از سمت چپ یک گره آغاز می‌کنیم و اولین شماره را اختصاص می‌دهیم و به همین به گره‌ای سمت چپ فرزندان می‌آییم و شماره‌ها را به صورت افزایشی به سمت چپ آن‌ها نیز اختصاص می‌دهیم. سپس در ادامه به سمت راست آخرین نود رفته و از آن جا به سمت بالا می‌آییم و به همین ترتیب به صورت بازگشتی تخصیص شماره‌ها را ادامه می‌دهیم.

با اختصتص شماره‌ها به هر گره، می‌توان از آن‌ها برای یافتن نیاکان و فرزندان آن گره بهره جست. برای مثال برای بازیابی گره‌ی 4 و فرزندان (نوه‌های) آن باید دنبال گره‌هایی باشیم که شماره‌های آن گره‌ها بین nsleft و nsright گره‌ی شماره4 باشد:

```
SELECT c2.* FROM Comments AS c1
JOIN Comments as c2
ON c2.nsleft BETWEEN c1.nsleft AND c1.nsright
WHERE c1.comment_id = 4;
```

همچنین می‌توان گره‌ی شماره‌ی 6 و نیاکان آن را با دنبال نمودن گره‌هایی به دست آورد که شماره‌های آن‌ها در محدوده‌ی شماره‌ی گره‌ی 6 باشد:

```
SELECT c2.*
FROM Comments AS c1
JOIN Comment AS c2
ON c1.nsleft BETWEEN c2.nsleft AND c2.nsright
WHERE c1.comment_id = 6;
```

یک مزیت مهم روش مجموعه‌ای تودرتو، این است که هنگامی که یک گره را حذف می‌کنیم، نوه‌های آن به طور مستقیم به عنوان فرزندان پدر گره‌ی حذف شده تلقی می‌شوند.

برخی از پرس‌وجوهایی که در روش لیست مجاورت ساده بودند، مانند بازیابی فرزند یا پدر بلافصل، در روش مجموعه‌های تودرتو پیچیده‌تر می‌باشند. برای مثال برای یافتن پدر بلافصل گره‌ی شماره‌ی 6 باید چنین نوشت:

```
SELECT parent.* FROM Comment AS c
JOIN Comment AS parent
ON c.nsleft BETWEEN parent.nsleft AND parent.nsright
LEFT OUTER JOIN Comment AS in_between
ON c.nsleft BETWEEN in_between.nsleft AND in_between.nsright
AND in_between.nsleft BETWEEN parent.nsleft AND parent.nsright
WHERE c.comment_id = 6
AND in_between.comment_id IS NULL;
```

دست‌کاری درخت، اضافه، حذف و جابجا نمودن گره‌ها در آن درروش مجموعه‌های تودرتو از مدل‌های دیگر پیچیده‌تر است. هنگامی که یک گره‌ی جدید را اضافه می‌کنیم، باید تمام مقادیر چپ و راست بزرگ‌تر از مقدار سمت چپ گره‌ی جدید را مجددا محاسبه کنیم؛ که این شامل برادر سمت راست گره‌ی جدید، نیاکان آن و برادر سمت راست نیاکان آن می‌باشد. همچنین اگر گره‌ی جدید به عنوان گره‌ی غیربرگ اضافه شده باشد، شامل فرزندان آن هم می‌شود. برای مثال اگر بخواهیم گره‌ی جدیدی به گره‌ی 5 اضافه نماییم، باید چنین بنویسیم:

```
-- make space for NS values 8 and 9
UPDATE Comment
SET nsleft = CASE WHEN nsleft >= 8 THEN nsleft+2 ELSE nsleft END,
nsright = nsright+2
WHERE nsright >= 7;

-- create new child of comment #5, occupying NS values 8 and 9
INSERT INTO Comment (nsleft, nsright, author, comment)
VALUES (8, 9, 'Fran' , 'Me too!' );
```

تنها مزیت این روش نسبت به روش‌های قبلی ساده‌تر و سریع‌تر شدن ایجاد پرس‌وجوها برای پیدا کردن فرزندان یا پدران یک درخت است. اگر هدف استفاده از درخت شامل اضافه نمودن متعدد گره‌ها است، مجموعه‌های تودرتو انتخاب خوبی نیست.

Closure Table

راه حل closure table روشی دیگر برای ذخیره‌ی سلسه‌مراتبی است. این روش علاوه بر ارتباطات مستقیم پدر- فرزندی، تمام مسیرهای موجود در درخت را ذخیره می‌کند.

این روش علاوه بر داشتن یک جدول نظرها، یک جدول دیگر به نام TreePaths با دو ستون دارد که هر کدام از این ستون‌ها یک کلید خارجی به جدولComment هستند:

```
CREATE TABLE Comments ( comment_id SERIAL PRIMARY KEY,
bug_id BIGINT UNSIGNED NOT NULL,
author BIGINT UNSIGNED NOT NULL,
comment_date DATETIME NOT NULL,
comment TEXT NOT NULL,
FOREIGN KEY (bug_id) REFERENCES Bugs(bug_id),
FOREIGN KEY (author) REFERENCES Accounts(account_id)
);
CREATE TABLE TreePaths (
ancestor BIGINT UNSIGNED NOT NULL,
descendant BIGINT UNSIGNED NOT NULL,
PRIMARY KEY(ancestor, descendant),
FOREIGN KEY (ancestor) REFERENCES Comments(comment_id),
FOREIGN KEY (descendant) REFERENCES Comments(comment_id)
);
```

ancestor	descendant	ancestor	descendant	ancestor	descendant
1	1	1	7	4	6
1	2	2	2	4	7
1	3	2	3	5	5
1	4	3	3	6	6
1	5	4	4	6	7
1	6	4	5	7	7

به جای استفاده از جدول Comments برای ذخیره‌ی اطلاعات مربوط به یک درخت از جدول TreePath استفاده می‌کنیم. به ازای هر یک جفت گره در این درخت یک سطر در جدول ذخیره می‌شود که ارتباط پدر فرزندی را نمایش می‌دهد و الزاما نباید این دو پدر فرزند بلافصل باشد. همچنین یک سطر هم به ازای ارتباط هر گره با خودش به جدول اضافه می‌گردد.

پرس‌وجوهای بازیابی نیاکان و فرزندان (گره‌ها) از طریق جدول TreePaths ساده‌تر از روش مجموعه‌های تودرتو است. مثلا برای بازیابی فرزندان (نوه‌های) گره‌ی شماره‌ی 4، سطرهایی که نیاکان آن‌ها 4 است را به دست می‌آوریم:

```
SELECT c.*  FROM Comments AS c
JOIN TreePaths AS t ON c.comment_id = t.descendant
WHERE t.ancestor = 4;
```

برای به دست آوردن نیاکان گره‌ی شماره‌ی 6، سطرهایی از جول TreePaths را به دست می‌آوریم که فرزندان آن‌ها 6 باشد:

```
SELECT c.*
FROM Comments AS c
JOIN TreePaths AS t ON c.comment_id = t.ancestor
WHERE t.descendant = 6;
```

برای اضافه کردن گره‌ی جدید، برای مثال به عنوان فرزند گره‌ی شماره‌ی 5، ابتدا سطری که به خود آن گره برمی‌گردد را اضافه می‌کنیم، سپس یک کپی از سطوری که در جدول TreePaths، به عنوان فرزندان (نوه‌های) گره‌ی شماره5 هستند (که شامل سطری که به خود گره‌ی 5 به عنوان فرزند اشاره می‌کند) به جدول اضافه نموده و فیلد descendant آن را با شماره‌ی گره‌ی جدید جایگزین می‌کنیم:

```
INSERT INTO TreePaths (ancestor, descendant) SELECT t.ancestor, 8
FROM TreePaths AS t
WHERE t.descendant = 5
UNION ALL
SELECT 8, 8;
```

در این جا می‌توان به اهمیت ارجاع یک گره به خودش به عنوان پدر (یا فرزند) پی برد.
برای حذف یک گره، مثلا گره‌ی شماره‌ی 7، تمام سطوری که فیلد descendant آن‌ها در جدول TreePaths برابر با 7 است حذف می‌کنیم:

```
DELETE FROM TreePaths WHERE descendant = 7;
```

برای حذف یک زیردرخت کامل، برای مثال گره‌ی شماره‌ی 4 و فرزندان (نوه‌های) آن، تمام سطوری که در جدول TreePaths دارای فیلد descendant با مقدار 4 هستند، حذف می‌کنیم. علاوه بر این باید نودهایی که به عنوان descendant به فیلد descendant گره‌ی 4، ارجاع داده می‌شوند نیز باید حذف گردد:

```
DELETE FROM TreePaths
WHERE descendant IN (SELECT descendant
FROM TreePaths
WHERE ancestor = 4);
```

دقت کنید وقتی گره‌ای را حذف می‌کنیم، بدان معنی نیست که خود گره (نظر) را حذف می‌کنیم. البته این برای مثال نظر و پاسخ آن مقداری عجیب است ولی در مثال کارمندان در چارت سازمانی امری معمول است. هنگامی که ارتباطات یک کاربر را تغییر می‌دهیم، از حذف در جدول TreePaths استفاده می‌کنیم و این قضیه که ارتباطات کارمندان در جدول جداگانه‌ای ذخیره شده است به ما انعطاف‌پذیری بیشتری می‌دهد.

برای جابجایی یک زیردرخت از مکانی به مکان دیگری در درخت، سطرهایی که ancestor گره‌ی بالایی زیردرخت را برمی‌گردانند و فرزندان آن گره را حذف می‌کنیم. برای مثال برای جابجایی گره‌ی شماره‌ی 6 به عنوان فرزند گره‌ی شماره‌ی 4 و قرار دادن آن به عنوان فرزند گره‌ی شماره‌ی 3، این چنین عمل می‌کنیم. فقط باید حواسمان جمع باشد سطری که گره‌ی شماره‌ی 6 به خودش ارجاع داده است را حذف نکنیم:

```
DELETE FROM TreePaths
WHERE descendant IN (SELECT descendant
                                        FROM TreePaths
                                        WHERE ancestor = 6)
AND ancestor IN (SELECT ancestor
                            FROM TreePaths
                            WHERE descendant = 6
                                AND ancestor != descendant);
```

آن‌گاه این زیردرخت جدا شده را با اضافه کردن سطرهایی که با ancestor مکان جدید و descendant زیردرخت، منطبق هستند، به جدول اضافه می‌کنیم:

```
INSERT INTO TreePaths (ancestor, descendant)
SELECT supertree.ancestor, subtree.descendant
FROM TreePaths AS supertree
CROSS JOIN TreePaths AS subtree
WHERE supertree.descendant = 3
AND subtree.ancestor = 6;
```

روش Closure Table آسان‌تر از روش مجموعه‌های تودرتو است. هر دوی آن‌ها روش‌های سریع و آسانی برای ایجاد پرس‌وجو برای نیاکان و نوه‌ها دارند. ولی Closure Table برای نگهداری اطلاعات سلسله مراتب آسان‌تر است. در هر دو طراحی ایجاد پرس‌وجو در فرزندان و پدر بلافصل سرراست‌تر از روش‌ای لیست مجاورت و شمارش مسیر می‌باشد.
می‌توان عملکرد Closure Table را برای ایجاد پرس‌وجو روی فرزندان و پدر بلافصل را آسان‌تر نیز نمود. اگر فیلد path_length را به جدول TreePaths اضافه نماییم این کار انجام می‌شود. path_length گره‌ای که به خودش ارجاع می‌شود، صفر است. path_length فرزند بلافصل هر گره 1، path_length نوه‌ی آن 2 می‌باشد و به همین ترتیب path_lengthها را در هر سطر مقداردهی می‌کنیم. اکنون یا فتن فرزند گره‌ی شماره‌ی 4 آسان‌تر است:

```
SELECT *
FROM TreePaths
WHERE ancestor = 4 AND path_length = 1;
```

از کدام طراحی استفاده نماییم؟

در این جا این سؤال مطرح است که ما باید از کدام طراحی استفاده نماییم. در پاسخ به این سؤال باید گفت که هر کدام از این روش‌ها نقاط قوت و ضعفی دارند که ما باید نسبت به عملیاتی که می‌خواهیم انجام دهیم از این طراحی‌ها استفاده کنیم. جدولی که در ادامه آمده است، مقایسه‌ای است میان میزان سهولت اجرای این طراحی‌ها در استفاده از پرس‌وجوهای متفاوت.

Design	Tables	Query Child	Query Tree	Insert	Delete	Referential Integration
Adjacency List	1	Easy	Hard	Easy	Easy	Yes
Recursive Query	1	Easy	Easy	Easy	Easy	Yes
Path Enumeration	1	Easy	Easy	Easy	Easy	No
Nested Sets	1	Hard	Easy	Hard	Hard	No
Closure Table	2	Easy	Easy	Easy	Easy	Yes

لازم به ذکر است در اینجا ستون سوم (Query Child) به معنای پرس‌وجوهایی است که با فرزندان کار می‌کند و ستون چهارم (Query Tree) به معنای پرس‌وجوهایی است که با کل درخت کار می‌کنند، می‌باشد.

نظرات خوانندگان

نویسنده: علیرضا صبوری
تاریخ: ۱۳۹۳/۰۵/۰۵ ۱۴:۴۷

فکر میکنم عموما پرس‌وجوی بازگشتی اگر ساپورت بشه توسط دیتابیس بهترین روش همان لیست مجاورت هستش که مدیریت درخت رو برامون ساده میکنه و دیتابیس کنترل بشتری رو هر نود ما داره. البته به غیر از مواردی خاص... ممنون از مطلب مفیدتون ولی سوالی که دارم اینه از نظر Performance مقایسه ای انجام شده که آیا استفاده از لیست بازگشتی چقدر از نظر سرعت در بازیابی اطلاعات با سایر روش‌ها تفاوت داره ؟ مبنعی اگر سراغ دارید ممنون میشم معرفی کنین.

عنوان: **پیاده سازی عملیات صفحه بندی (paging) در sql server**
نویسنده: محمد حسین عزتی
تاریخ: ۱۳۹۳/۰۷/۲۲ ۱۹:۵۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, Paging, Performance

در خیلی مواقع ملاحظه میشود که برای نمایش تعدادی از رکوردهای یک جدول در پایگاه داده، کل مقادیر موجود درآن توسط یک دستور select به دست می‌آید و صفحه‌بندی خروجی، به کنترلهای موجود سپرده میشود. اگر پایگاه داده ما دارای تعداد زیادی رکورد باشد، آن موقع است که دچار مشکل می‌شویم. فرض کنید به طور همزمان ۵ نفر (که تعداد زیادی نیستند) از برنامه ما که شامل ۱۰۰۰۰۰ سطر داده میباشد استفاده کنند و در هر صفحه، ۱۰ رکورد نمایش داده شود و صفحه‌بندی ما از نوع معقولی نباشد. در این صورت به جای اینکه با ۵×۱۰ رکورد داده را بارگزاری کنیم، ۵×۱۰۰۰۰۰ رکورد یعنی ۵۰۰۰۰۰ رکورد را برای به دست آوردن ۵۰ رکورد بارگزاری میکنیم. در زیر روشی شرح داده میشود که توسط آن، این سربار اضافه از روی برنامه و سرورهای مربوطه حذف شود. به stored procedure و توضیحات مربوط به آن توجه فرمایید :

```
CREATE PROCEDURE sp_PagedItems
(
 @Page int,
 @RecsPerPage int
)
AS

-- We don't want to return the # of rows inserted
-- into our temporary table, so turn NOCOUNT ON
SET NOCOUNT ON


--Create a temporary table
CREATE TABLE #TempItems
(
ID int IDENTITY,
Name varchar(50),
Price currency
)


-- Insert the rows from tblItems into the temp. table
INSERT INTO #TempItems (Name, Price)
SELECT Name,Price FROM tblItem ORDER BY Price

-- Find out the first and last record we want
DECLARE @FirstRec int, @LastRec int
SELECT @FirstRec = (@Page - 1) * @RecsPerPage
SELECT @LastRec = (@Page * @RecsPerPage + 1)

-- Now, return the set of paged records, plus, an indiciation of we
-- have more records or not!
SELECT *,
MoreRecords =
(
 SELECT COUNT(*)
 FROM #TempItems TI
 WHERE TI.ID >= @LastRec
)
FROM #TempItems
WHERE ID > @FirstRec AND ID < @LastRec


-- Turn NOCOUNT back OFF
SET NOCOUNT OFF
```

در این کد دو پارامتر از نوع integer تعریف میکنیم. اول پارامتر @Page که مربوط به شماره صفحه‌ای می‌باشد که قصد دارید آن‌را بارگزاری نمایید. دومین پارامتر با نام @RecsPerPage تعداد رکوردهایی است که هر بار میخواهید بارگزاری شوند. مثلا اگر میخواهید هر بار ۱۵ عدد از رکوردها را نمایش دهید، این مقدار را باید برابر ۱۵ قرار دهیم. در مرحله بعد یک جدول موقت با نام #TempItems ساخته شده است که به طور موقت مقادیری را در حافظه نگه میدارد. نکته کلیدی که جلوتر از آن استفاده شده، ستون با نام ID است که از نوع auto-increment بوده و روی جدول موقت تعریف شده است. این ستون شناسه هر سطر را در خود نگه میدارد که به صورت اتوماتیک بالا میرود و جزء لاینفکی از این نوع paging میباشد. پس از آن جدول موقت را توسط

رکوردهای جدول واقعی با نام tblItem توسط دستور select پر میکنیم.

در مرحله بعد شماره اولین و آخرین سطر مورد نظر را بر اساس پارامترهای ورودی محاسبه کرده و در متغیرهای @FirstRec و @LastRec می‌ریزیم.
برای استفاده از این کد فقط کافیست که پارامترهای ورودی را مقداردهی نمایید. مثلا اگر میخواهید در یک کنترل Grid از آن استفاده کنید باید ابتدا یک کوئری داشته باشید که تعداد کل سطرها را به شما بدهد و بر اساس این مقدار تعداد صفحات مورد نظر را به دست آورید. پس از آن با کلیک روی هر کدام از شماره صفحات آن را به عنوان مقدار به پارامتر مورد نظر بفرستید و از آن لذت ببرید.

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۷/۲۲ ۱۹:۵۹

ضمن تشکر از شما. یک اصلاح کوچک: جدول موقتی ایجاد شده در پایان کار رویه ذخیره شده باید drop بشه.

نویسنده: محمد حسین عزتی
تاریخ: ۱۳۹۳/۰۷/۲۲ ۲۰:۱۸

از دقت شما به این نکته ظریف ممنونم
این موضوع در راستای آموزش عنوان مطلبش بود اما نکته شما جهت بالا بردن کیفیت و بهینه کردن کد مورد استفاده قرار میگیرد و عدم drop مشکلی در رسیدن به هدف مورد نظر ایجاد نمی‌کند
متشکرم

نویسنده: امید
تاریخ: ۱۳۹۳/۰۷/۲۲ ۲۱:۲۰

در sql 2012 به بعد جهت صفحه بندی دستورات offset و fetch اضافه شده که از لحاظ Performance بهینه‌تر از باقی روش‌های میباشد . مقایسه صفحه بندی‌های مختلف

نویسنده: محمد حسین عزتی
تاریخ: ۱۳۹۳/۰۷/۲۲ ۲۱:۵۴

ممنونم بخاطر لینک مفیدی که در قسمت نظر ارسال نمودید
تنها نقطه ضعف این مقاله همینطور که خود شما هم متذکر شده اید این است که برای ورژن‌های بانک اطلاعاتی بعد از 2012 قابل استفاده است. هنوز بسیاری از نرم افزارها و سازمانهای ما هنوز با ورژن‌های قدیمی‌تر کار می‌کنند.

متشکرم

نویسنده: حمیدرضا کاظم نادی
تاریخ: ۱۳۹۳/۰۷/۲۳ ۱۱:۲۹

ممنون از مطلب خوبتون
به نظرم اگه جوری Sp را مینوشتید که یک ورودی متنی Query یا یک جدول موقت میگرفت و عمل Paging را روی اون انجام میداد مطلبتون بسیار کامل‌تر بود. مثلا ورودی Sp به این صورت بود که ('select * from Tbl_1',1,10) بازهم ممنون

نویسنده: محمد حسین عزتی
تاریخ: ۱۳۹۳/۰۷/۲۳ ۱۲:۵۴

سلام
ممنونم از نظرتون
دوتا پارامتر داره از ورودی دریافت می‌کنه و هدف نحوه انجام صفحه بندی بوده

متشکرم

نویسنده: رحمت اله رضایی
تاریخ: ۱۳۹۳/۰۷/۲۳ ۱۳:۳۶

روشی بسیار قدیمی‌است و این روزها آنچنان کاربرد ندارد.

برای صفحه‌بندی :

- در SQL Server 2008 از ROW_NUMBER استفاده می‌کنند.
- در SQL Server 2012 به بعد از OFFSET FETCH استفاده می‌کنند .

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۷/۲۳ ۱۴:۱۰

تاریخچه‌ای از روش‌های مختلف صفحه بندی اطلاعات در SQL Server در این مقاله بحث شده به همراه بررسی کارآیی آن‌ها:

Comparing performance for different SQL Server paging methods

عنوان: **فعال‌سازی Multiple Active Result Sets**
نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۳/۰۷/۲۳ ۱۴:۰
آدرس: www.dotnettips.info
گروه‌ها: Entity framework, SQL Server

Multiple Active Result Sets (MARS) یکی از قابلیتهای SQL SERVER است. این قابلیت در واقع این امکان را برای ما فراهم می‌کند تا بر روی یک Connection همزمان چندین کوئری را به صورت موازی ارسال کنیم. در این حالت برای هر کوئری یک سشن مجزا در نظر گرفته می‌شود.

مدل:

```
namespace EnablingMARS.Models
{
    public class Product
    {
        public int Id { get; set; }
        public string Title { get; set; }
        public string Desc { get; set; }
        public float Price { get; set; }
        public Category Category { get; set; }

    }

    public enum Category
    {
        Cate1,
        Cate2,
        Cate3
    }
}
```

کلاس Context:

```
namespace EnablingMARS.Models
{
    public class ProductDbContext : DbContext
    {
        public ProductDbContext() : base("EnablingMARS") {}
        public DbSet<Product> Products { get; set; }

    }
}
```

ابتدا یک سطر جدید را توسط کد زیر به دیتابیس اضافه می‌کنیم:

```
MyContext.Products.Add(new Product()
 {
                Title = "title1",
                Desc = "desc",
                Price = 4500f,
                Category = Category.Cate1
   });
MyContext.SaveChanges();
```

اکنون می‌خواهیم قیمت محصولاتی را که در دسته‌بندی Cate1 قرار دارند، تغییر دهیم:

```
foreach (var product in _dvContext.Products.Where(category => category.Category == Category.Cate1))
{
     product.Price = 50000;
     MyContext.SaveChanges();
}
```

خوب؛ اکنون اگر برنامه را اجرا کنیم با خطای زیر مواجه می‌شویم:

```
There is already an open DataReader associated with this Command which must be closed first.
```

این استثناء زمانی اتفاق می‌افتد که بر روی نتایج حاصل از یک کوئری، یک کوئری دیگر را ارسال کنیم. البته استثنای صادر شده بستگی به کوئری دوم شما دارد ولی در حالت کلی و با مشاهده Stack Trace، پیام فوق نمایش داده می‌شود. همانطور که در کد بالا ملاحظه می‌کنید درون حلقه‌ی forach ما به پراپرتی Price دسترسی پیدا کرده‌ایم، در حالیکه کوئری اصلی ما هنوز فعال (Active) است. MARS در اینجا به ما کمک می‌کند که بر روی یک Connection، بیشتر از یک کوئری فعال داشته باشیم. در حالت عادی Entity Framework Code First این ویژگی را به صورت پیش‌فرض برای ما فعال نمی‌کند. اما اگر خودمان کانکشن‌استرینگ را اصلاح کنیم، این ویژگی SQL SERVER فعال می‌گردد. برای حل این مشکل کافی است به کانکشن‌استرینگ، MultipleActiveResultSets=true را اضافه کنیم:

```
"Data Source=(LocalDB)\v11.0;Initial Catalog=EnablingMARS; MultipleActiveResultSets=true"
```

لازم به ذکر است که این قابلیت از نسخه SQL SERVER 2005 به بالا در دسترس می‌باشد. همچنین در هنگام استفاده از این قابلیت می‌بایستی موارد زیر را در نظر داشته باشید:

وقتی کانکشنی در حالت MARS برقرار می‌شود، یک سشن نیز همراه با یکسری اطلاعات اضافی برای آن ایجاد شده که باعث ایجاد Overhead خواهد شد.

دستورات مارس thread-safe نیستند.

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۷/۲۴ ۱۰:۳۲

با تشکر. یک سؤال: اگر بجای dvContext.Products.Where بنویسیم dvContext.Products. **Include** (x=>x.Category).Where باز هم این مشکل هست؟

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۳/۰۷/۲۴ ۱۰:۴۱

بله می‌شود به صورت eager loading این مشکل را نیز حل کرد. ولی مسئله performance را نیز باید در نظر داشته باشید.
البته می‌بایست به این صورت نوشته شود:

```
foreach (var product in _dvContext.Products.Where(x => x.Category.CateName == "Cate1").Include(x =>
x.Category).ToList())
{
                product.Price = 50000;
                _dvContext.SaveChanges();
}
```

حالت فوق در صورتی است که یک Navigation property با نام Category داشته باشید.

عنوان: **استفاده از SQLDom برای آنالیز عبارات T-SQL، قسمت دوم**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۳۰ ۱۳:۱۵
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, T-SQL, ScriptDom

مدتی قبل مطلبی را در مورد کتابخانه‌ی ویژه SQL Server که یک T-SQL Parser تمام عیار است، در این سایت مطالعه کردید . در این قسمت، همان مطلب را به نحو بهتر و ساده‌تری بازنویسی خواهیم کرد.
مشکلی که در دراز مدت با SQLDom وجود خواهد داشت، مواردی مانند SelectStarExpression و CreateProcedureStatement و امثال آن هستند. این‌ها را از کجا باید تشخیص داد؟ همچنین مراحل بررسی این اجزاء، نسبتا طولانی هستند و نیاز به یک راه حل عمومی‌تر در این زمینه وجود دارد.

راه حلی برای این مشکل در مطلب « XML 'Visualizer' for the TransactSql.ScriptDom parse tree » ارائه شده‌است. در اینجا تمام اجزای TSqlFragment توسط Reflection مورد بررسی و استخراج قرار گرفته و نهایتا یک فایل XML از آن حاصل می‌شود.
اگر نکات ذکر شده در این مقاله را تبدیل به یک برنامه با استفاده مجدد کنیم، به چنین شکلی خواهیم رسید:

این برنامه را از اینجا می‌توانید دریافت کنید:
DomToXml.zip

همانطور که در تصویر مشاهده می‌کنید، اینبار به سادگی، SelectStarExpression قابل تشخیص است و تنها کافی است در T-SQL پردازش شده، به دنبال SelectStarExpressionها بود. برای اینکار جهت ساده شدن آنالیز می‌توان با ارث بری از کلاس پایه TSqlFragmentVisitor شروع کرد:

```
using System;
using System.Linq;
using Microsoft.SqlServer.TransactSql.ScriptDom;

namespace DbCop
{
    public class SelectStarExpressionVisitor : TSqlFragmentVisitor
    {
        public override void ExplicitVisit(SelectStarExpression node)
```

```
        {
            Console.WriteLine(
                  "`Select *` detected @StartOffset:{0}, Line:{1}, T-SQL: {2}",
                  node.StartOffset,
                  node.StartLine,
                  string.Join(string.Empty, node.ScriptTokenStream.Select(x => x.Text)).Trim());

            base.ExplicitVisit(node);
        }
    }
}
```

در کلاس پایه TSqlFragmentVisitor به ازای تمام اشیاء شناخته شده‌ی ScriptDom، یک متد ExplicitVisit قابل بازنویسی درنظر گرفته شده‌است. در اینجا برای مثال نمونه‌ی SelectStarExpression آن را بازنویسی کرده‌ایم.
مرحله‌ی بعد، اجرای این کلاس Visitor است:

```
public static class GenericVisitor
    {
        public static void Start(string tSql, TSqlFragmentVisitor visitor)
        {
            IList<ParseError> errors;
            TSqlScript sqlFragment;
            using (var reader = new StringReader(tSql))
            {
                var parser = new TSql120Parser(initialQuotedIdentifiers: true);
                sqlFragment = (TSqlScript)parser.Parse(reader, out errors);
            }

            if (errors != null && errors.Any())
            {
                var sb = new StringBuilder();
                foreach (var error in errors)
                    sb.AppendLine(error.Message);

                throw new InvalidOperationException(sb.ToString());
            }
            sqlFragment.Accept(visitor);
        }
    }
```

در اینجا متد Accept کلاس TSql120Parser، امکان پذیرش یک Visitor را دارد. به این معنا که Parser در حال کار، هر زمانیکه در حال آنالیز قسمتی از T-SQL دریافتی بود، نتیجه را به اطلاع یکی از متدهای کلاس پایه TSqlFragmentVisitor نیز خواهد رساند. بنابراین دیگر نیازی به نوشتن حلقه و بررسی تک تک اجزای خروجی TSql120Parser نیست. اگر نیاز به بررسی SelectStarExpression داریم، فقط کافی است Visitor آن‌را طراحی کنیم.

مثالی از نحوه‌ی استفاده از کلاس GenericVisitor فوق را در اینجا ملاحظه می‌کنید:

```
var tsql = @"WITH ctex AS (
SELECT * FROM sys.objects
)
SELECT * FROM ctex";
GenericVisitor.Start(tsql, new SelectStarExpressionVisitor());
```