التوابع المعرفة من قبل المستخدم والمناظير

User Defined Functions and Views

المناظير Views

أبسط طريقة لفهم المناظير هي اعتبارها تعابير Select محفوظة تسمح للمستخدم بالعمل مع النتائج التي تعيدها و هي غالباً ما تستخدم كطريقة لتغليف الاستعلامات أو عمليات الدمج المعقدة بين الجداول و لتطبيق العمليات التجميعية المعقدة و صياغة البيانات المعادة بشكل مريح أكثر. كما أنها تعد من أهم أدوات الأمان لأنها تساعد على منع المستخدمين ذوي الصلاحيات المحدودة من الوصول إلى الجداول الأصلية في قواعد البيانات و منحهم صلاحيات قراءة فقط أو منحهم صلاحيات لرؤية لعض الحقول من الجدول فقط كما تساهم المناظير في اخفاء اسماء الحقول الحقيقية في الجداول الاساسية.

- صيغة إنشاء منظار

```
CREATE VIEW view_name AS
            select_statement
```

نلاحظ أننا يجب أن نعطي المنظار اسما محددا بالواصفة view_name ومن ثم نكتب الاستعلام المغلف بهذا المنظار بعد الكلمة المفتاحية as.

- مثال: لتغليف الاستعلام الذي يعيد اسم وبلد كل زبون في الجدول Customers من القاعدة Northwind فإننا نكتب الأمر التالي:

```
USE northwind
Go
CREATE VIEW vwCustomerCountry
AS
      SELECT CompanyName, Country
       FROM Customers
```

نستطيع الآن الاستعلام عن السجلات الناتجة عن المنظار كما في حالة الجداول باستخدام select.

مثلا لتحديد قائمة الزبائن الذين يحوي اسم بلدهم على الحرف u نستخدم الاستعلام التالي:

```
select * from vwCustomerCountry
      where country like '%u%'
```

- **الفائدة من استخدام المناظير**

❖ يمكن للأشخاص الذين يعملون على تطبيقات التعامل مع استعلامات بسيطة على المنظار الذي يعامل كجدول دون الاضطرار إلى الاستعلامات المعقدة والمزعجة وهذا مفيد في حال الحاجة إلى الاستعلام بشكل متكرر.

❖ تفصيل البيانات بحسب حاجة التطبيقات دون الحاجة إلى خلق بيانات مكررة.

❖ تحديد قيود على الحقول المرئية.

```
CREATE VIEW uk_customers
AS
SELECT * FROM Customers where Country like 'uk'
```

مثلا لعرض قائمة الموظفين الذين تم توظيفهم في العام 1994 يمكننا تغليف الاستعلام الموافق ضمن منظار كما يلي:

```
Use northwind
Go
Alter VIEW vwEmployeesHiredThisYear
AS
 SELECT LastName, FirstName, HireDate
 FROM Employees
 WHERE Year(HireDate) = 1994
```

لاحظ أننا لم نعرض جميع حقول الموظف بل اكتفينا بإرجاع اسم ونسبة وتاريخ توظيف كل منهم مع وضع قيود على السجلات العائدة تمثلت في حالتنا بكون تاريخ التوظيف هو عام 1994.

❖ تغليف الاستعلامات المعقدة و بالأخص لأغراض تصدير التقارير.

مثلا يمكننا تغليف الاستعلام الذي يعيد المبلغ الإجمالي الموافق لطلبيات كل شركة بمنظار لتسهيل التعامل معه

```
use northwind
go
CREATE VIEW vwCustomerOrders
AS
SELECT c.CompanyName , SUM(od.UnitPrice*od.Quantity) AS Total
FROM
      Customers c INNER JOIN Orders o ON c.CustomerID = o.CustomerID
            INNER JOIN [Order Details] od ON  o.OrderID = od.OrderID
GROUP BY
      c.CompanyName
```

- ❖ حماية الجداول من الوصول المباشر: نقوم في هذه الحالة بحرمان المستخدمين من حق الاستعلام في الجدول المطلوب (سنتعرض لهذا الموضوع لاحقا عند الحديث عن أمن البيانات) ونقوم ببناء منظار (أو مناظير) على هذا الجدول المحمي ونتيح استخدام هذه المناظير فقط للمستخدمين.
- ❖ حماية أسماء الحقول من المتطفلين.

مثلا للاستعاضة عن الحقل CompanyName بالقيمة C وعن الحقل ContactName بالقيمة CN يمكننا كتابة المنظار التالي للزبائن:

```
use northwind
go
CREATE VIEW vwObscure
AS
SELECT CompanyName AS C, ContactName AS CN
FROM Customers
```

- **صيغة تعديل منظار**

نستخدم التعليمة التالية

```
Alter VIEW view_name AS
          NEW_select_statement
```

مثلا لتعديل المنظار السابق بحيث نضيف اسم مدينة الزبون نكتب الأمر التالي:

```
USE northwind
Go
Alter VIEW vwCustomerCountry
AS
      SELECT CompanyName, Country, City FROM Customers
```

- **صيغة حذف منظار**

لحذف منظار نستخدم التعليمة التالية

```
DROP VIEW view_name
```

وهذا سيحذف المنظار view_name من القاعدة الحالية.
مثلا لحذف المنظار vwCustomerCountry من القاعدة Northwind نستخدم الأمر التالي:

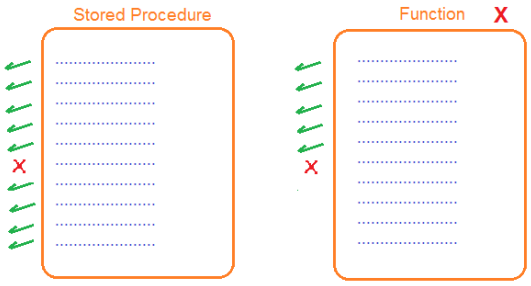
```
drop view vwCustomerCountry
```

التوابع المعرفة من قبل المستخدم:

التوابع هي إجرائيات مخزنة تتكون من سلسلة من التعليمات التي تخزن من اجل استخدام لاحق. يتم إنشاء التوابع بالتعليمة CREATE FUNCTION، ويتم تعديل التوابع بالتعليمة ALTER FUNCTION، أما حذف التوابع فيتم بالتعليمة DROP FUNCTION.

الفروقات بين التوابع المعرفة من قبل المستخدم UDF والإجرائيات المخزنة SP:

التوابع	الاجرائيات المخزنة
يعيد قيمة وحيدة بشكل دائم	يمكن أن لا تعيد أي قيمة أو أن تعيد عدة قيم
يمكن أن يعيد قيمة من نمط جدول	لا يمكن أن يعيد قيمة من نمط جدول
لا نستطيع استدعاء اجرائية مخزنة ضمنه	يمكن استدعاء تابع ضمن الاجرائية
يمكن استخدام التابع ضمن SELECT / where / having	لايمكن استخدام الاجرائية ضمن SELECT / where / having
لا يمكن استخدام ادارة الأخطاء ضمنه بالتالي وجود خطأ سينهي تنفيذ التابع حكما	يمكن استخدام ادارة الأخطاء try-catch ضمنه مما يضمن تنفيذ الاجرائية حتى مع وجود خطأ
لا تستطيع عن طريقه تغيير وسطاء خاصة بالسيرفر	تستطيع تغيير وسطاء خاصة بالسيرفر بشكل عام



بعض النقاط المتعلقة بالتوابع:

- ✓ من أجل إنشاء، تعديل، حذف تابع يجب أن تملك صلاحية CREATE FUNCTION.
- ✓ قبل أن تستطيع استخدام تابع في سلسلة من التعليمات يجب أن تعطى الصلاحية لذلك.
- ✓ يمكن أن تدخل التوابع في تعريف الجدوال (أثناء بناء الجدول) وذلك عن طريق إعطاء قيمة افتراضية DEFAULT أو فرض قيد CHECK CONSTRAINT أو الأعمدة المحسوبة COMPUTED COLUMNS.
- ✓ إن إعطاء المستخدم صلاحية التعديل أو الإنشاء لجدول يحوي مؤشرا Reference إلى تابع لا يمكنه من تعديل الجدول أو إنشاءه. يجب أن يملك المستخدم أيضا صلاحية Reference Permission على التابع.

- ✓ تأخذ التوابع معاملا Parameter أو أكثر وتعيد قيمة عددية أو جدولا. كما يمكن للتوابع أن تكون بدون معاملات.
- ✓ التوابع التي تعيد قيمة عددية يمكن أن تستخدم في أي تعبير Expression أو أي مكان يستخدم قيمة من نفس النمط التي يعيدها التابع.
- ✓ عندما تكون أحد معاملات التابع لها قيمة افتراضية Default فإنه يجب تحديد ذلك أيضا عند استدعاء التابع للحصول على القمية الإفتراضية، وذلك بعكس الإجرائيات حيث تمرر القيمة الإفتراضية إلى الإجرائية بدون ذكر كلمة Default.
- ✓ لا يمكن إعادة الأنماط التالية من تابع:

Timestamp
User defined data types
Cursors

أنماط التوابع المعرفة من قبل المستخدم Types of User-Defined Functions:

- التوابع التي تعيد قيمة Scalar functions.
- التوابع المعرفة ضمن السياق والتي تعيد جدولا Inline Table-Valued Functions:
- التوابع التي تحوي أكثر من تعلمية والتي تعيد جدولا Multi-Statement Table-Valued Functions

لا تحوي التوابع المعرفة ضمن السياق والتي تعيد جدولا على جسم تابع Function Body (جسم التابع هو مجموعة من التعليمات محاطة بـ Begin و End) بل التابع معرف كتعليمة select.

يحوي جسم التابع على أنواع التعلميات التالية فقط:

- تعليمة التصريح عن متحولات داخلية Declare Statement.
- تعلميات الإسناد Assignment Statements.
- التعليمات المتعلقة بالمؤشرات Cursors المعرفة محليا ضمن التابع.
- تعلميات التحكم Control Flow.
- تعلميات Update, Insert, Delete. (للجداول داخل الإجرائية أما الخارجية فلا ..)

```
Invalid use of a side-effecting operator 'INSERT'
```

- تعلميات تنفيذ إجرائية Execute Statement.

من الملاحظ أنه لا يمكن أن تحوي التوابع على تعليمة CREATE. أي أنه لا يمكن إنشاء إغراض جديدة في قاعدة البيانات من ضمن التابع.

للتوابع نمطين من الإستدعاء وذلك حسب القيمة المعادة من التابع:

- استدعاء التوابع التي تعيد قيمة Scalar: في هذه الحالة يجب أن نحدد بالإضافة إلى اسم التابع اسم المالك للتابع. مثال : إذا كان لدينا التابع MyFunction() المالك لهذا التابع هو MyUser فإن استدعاء التابع يجب أن يحوي على MyUser.MyFunction()

لمعرفة المستخدم الحالي :

```
select CURRENT_USER
select schema_name()
```

ملاحظة: في نسخة 2005 ومابعدها يتم إنشاء schema لكل مستخدم بحيث يتم استدعاء التوابع والأغراض بشكل عام بكتابة اسم الـ schema وبعدها اسم الغرض، بحيث لو تم حذف المستخدم يمكن بسهولة نقل الـ schema لمكان آخر مع بقاء صيغة الاستدعاء نفسها دون أن تتأثر.

dbo هو المستخدم الافتراضي ويوجد له schema بنفس الاسم

- استدعاء التوابع التي تعيد جدولا Table: في هذه الحالة يكتفي SQL باستخدام اسم الجدول بدون ذكر اسم المالك عند الإستدعاء.

مثال 1:

في المثال التالي سوف نقوم بإنشاء تابعا يولد قيمة تمثل حجم متوازي مستطيلات. وسوف نقوم باستخدام هذا التابع من أجل حساب قيمة محسوبة في جدول في عمود. اسمه BrickVolume

```
create function CubicVolume
(    @length as int,  @width  int,  @height as int    )
returns int
begin
      return (@length * @width * @height)
end



-- 1. first call of the function
      select dbo.CubicVolume(1,2,3)
-- 2. second call of the function
            declare @x int
            select @x = dbo.cubicVolume(1,2,3)
            print @x

-- 3 create the table that uses the function CubicVolume()
```

```
create table Bricks
(
	BrickPartNumber int primary key,
	BrickColor varchar(20),
	BrickHeight int,
	BrickLength int,
	BrickWidth int,
	BrickVolume as
	(
		dbo.CubicVolume(BrickHeight,BrickLength,BrickWidth)
	)
)

insert into Bricks(BrickPartNumber,BrickColor,BrickHeight,BrickWidth,
BrickLength)
		values (1,'dark red',1,2,3)
select * from Bricks
```

التوابع المعرفة من قبل المستخدم والتي تعيد نمط جدول

أولا: التوابع المعرفة من قبل المستخدم ضمن السياق

Inline User-Defined Functions

التوابع المعرفة من قبل المستخدم ضمن السياق هي مجموعة جزئية من التوابع بشكل عامل. تعيد هذه التوابع جدولا وتتألف من تعلمية اختيار واحدة (استعلام واحد) . تستخدم هذه التوابع كبديل عن المناظير.

مثال:03

إذا طلب منا قائمة بأرقام وأسماء الزبائن من قاعدة البيانات NorthWind والذين يعملون في WA. إن أحد الخيارات هو أنشاء منظار View على الشكل التالي:

```
CREATE VIEW vw_CustomerNamesInWA AS
SELECT
	CustomerID,
	CompanyName
FROM
	Customers
WHERE
	Region = 'WA'
```

نلاحظ في هذا المثال أننا من أجل كل منطقة Region نقوم بإنشاء View. إذا استعضنا عن الحل السابق بتابع يأخذ وسيطا اسم المنطقة فيكون الحل على الشكل التالي:

```
CREATE FUNCTION   fn_CustomerNamesInRegion
	( @RegionParameter nvarchar(30) )
  RETURNS table
AS
RETURN (
		SELECT
```

```
                        CustomerID,
                        CompanyName
                FROM
                    Customers
                WHERE
                 Region = @RegionParameter
        )
---- CALL The function
SELECT * FROM fn_CustomerNamesInRegion('WA')

SELECT * FROM fn_CustomerNamesInRegion('sp')
```

ملاحظات:

- لا يوجد حاجة إلى تعريف اسم المتحول الذي سنستخدمه داخليا في جسم التابع كما في حالة التوابع التي تعيد جدولا ولها جسما.
- لا يوجد حاجة إلى تعريف بينة الجدول المعاد من التابع.
- إن وسطاء التابع يمكن أن تستخدم في فلترة البيانات المعادة من قبل التابع.

ثانيا: التوابع المعرفة من قبل المستخدم بأكثر من تعليمة

Multi-Statement Table-Valued Functions

التوابع المعرفة من قبل المستخدم والتي تعيد جدولا يمكن أن تكون بديلا ممتازا عن المناظير **Views**. يمكن أن تستخدم التوابع التي تعيد جدولا في أي مكان يمكن استخدام جدول أن أو منظار. بينما تحوي المناظير على تعلمية اختيار واحدة، يمكن للتوابع أن تحوي تعليمات أخرى تمكننا من تنفيذ ألية عمل في التابع.

التوابع المعرفة من قبل المستخدم والتي تعيد جدولا يمكن أن تكون بديلا عن الإجرائيات التي تعيد نتيجة واحدة على شكل جدول **Single Result Set**، في هذه الحالة يمكن للتوابع أن تستخدم في فقرة **From** من تعليمة الإختيار **Select** بينما الإجرائيات المخزنة لا يمكن أن تستخدم في هذه الفقرة.

ملاحظة:

تعرف التوابع التي تعيد جدولا متحولا داخليا من نمط جدول. يمكن استخدام هذا المتحول الداخلي لعمليات الإضافة والحذف والتعديل. ناتج العمليات السابقة سوف يكون خرج التابع.

مثال 02

من قاعدة البيانات **Northwind**، نريد قائمة بأرقام وأسماء الشاحنين **Shippers** وأرقام الطلبيات وتاريخ الشحن وكلفة الشحن وذلك فقط للطلبيات التي يزيد كلفة شحنها عن مقدار محدد. علما أن عمود كلفة الشحن موجود في جدول الطلبيات واسمه **orders.Freight**

```
use Northwind
go
```

```
CREATE FUNCTION LargeOrderShippers ( @FreightParm money )
      RETURNS @OrderShipperTab TABLE
                        (  ShipperID     int,
                             ShipperName   nvarchar(80),
                             OrderID       int,
                             ShippedDate   datetime,
                             Freight       money
                        )
AS
BEGIN
   INSERT @OrderShipperTab
        SELECT S.ShipperID, S.CompanyName,
               O.OrderID, O.ShippedDate, O.Freight
        FROM Shippers AS S INNER JOIN Orders AS O
              ON S.ShipperID = O.ShipVia
        WHERE O.Freight > @FreightParm
   RETURN
END
-- Call the Function in From STATEMENT
select * from LargeOrderShippers($2000)
```

مثال آخر:

```
create function book_pric(@mypric int)
      returns   @mytable table
                              ( bookname varchar(80),
                                booktype varchar(80),
                                myprice int
                                     )
begin
            insert @mytable
                  select title , type , price
                  from titles
                  where price > @mypric
      return
                                                              end
```

التوابع المحددة وغير المحددة

Deterministic and Nondeterministic Functions

جميع التوابع إما أن تكون محددة أو غير محددة.

- **التوابع المحددة Deterministic functions: تعيد هذه المجموعة من التوابع نفس الخرج من أجل نفس الدخل مهما كان زمن الأستدعاء.**
- **التوابع الغير محددة Nondeterministic functions : تعيد هذه المجموعة من التوابع خرجا مختلفا من أجل الدخل نفسه وذلك باختلاف زمن الإستدعاء.**

مثال: التابع DATEADD هو تابع محدد لأنه يعطي نفس النتيجة من أجل الدخل نفسه. بينما التابع GETDATE هو تابع غير محدد لأنه في كل لحظة يعطي خرجا مختلفا.

تحوي القائمة التالية مجموعة التوابع المحددة في Microsoft SQL Server 2000:

ABS	DATEDIFF	PARSENAME
ACOS	DAY	POWER
ASIN	DEGREES	RADIANS
ATAN	EXP	ROUND
ATN2	FLOOR	SIGN
CEILING	ISNULL	SIN
COALESCE	ISNUMERIC	SQUARE
COS	LOG	SQRT
COT	LOG10	TAN
DATALENGTH	MONTH	YEAR
DATEADD	NULLIF	

إعادة كتابة الإجرائيات المخزنة كتوابع

Rewriting Stored Procedures as Functions

في الحالة العامة: إذا كانت الإجرائية تعيد قيمة Scalar Value يجب إعادة كتابة الإجرائية كتابع يعيد القيمة المحسوبة من قبل الإجرائية. وإذا كانت الإجرائية تعيد جدولا وحيدا Single Result Set وأردنا استخدام هذا الجدول في فقرة From من تعليمة Select فإنه يجب إعادة كتابة الإجرائية كتابع يعيد جدولا.

انتهت المحاضرة