SQL Programming

Módulo 3 - Tablas temporales y globales

Tablas temporales y globales

Tabla temporal local

Las **tablas temporales** son tablas que las crea el usuario (desarrollador / programador) temporalmente para resolver problemas complejos donde se necesita almacenar la información en un estado intermedio.

Hay dos tipos de tablas temporales: Locales y Globales. Las **tablas locales** son visibles solamente en la sesión actual mientras que las globales pueden ser accedidas desde otra sesión. Las tablas temporales (ambas: locales y globales) se eliminan automáticamente al finalizar la sesión o la función o el procedimiento almacenado en el cual fueron definidas (creadas). También se pueden eliminar con "Drop table".

Por qué usar tablas temporales

A pesar de que todo el mundo sabe que este tipo de estructuras ralentizan el funcionamiento de nuestras consultas, los programadores no pueden evitar recurrir a ellas porque muchas veces facilitan la resolución de problemas.

La solución más importante es que nos permiten almacenar datos para usarlos posteriormente, guardar resultados parciales y analizar grandes cantidades de filas.

Características

- Almacenar datos en forma temporales.
- Cuando se reinicia el servicio de SQL, estas tablas desaparecen.
- Facilitan la resolución de problemas.
- Almacenar datos para usarlos posteriormente.
- Se utilizan para guardar resultados parciales.
- Las tablas temporales se crean en la base tempdb
- Al crearlas es necesario que se realicen accesos de escritura al disco.
- Al leer datos de la tabla temporal hay que recurrir de nuevo al disco.
- Pueden ser locales (son visibles sólo en la sesión actual)

Por qué usar tablas temporales

Sintaxis

```
CREATE TABLE #Provincias(
ProvinciaID int NOT NULL,
Ciudad varchar(40) NOT NULL
);
```

Tabla temporal global

Para crear tablas temporales globales se emplea la misma sintaxis que para crear cualquier tabla, excepto que se coloca un signo numeral doble (##) precediendo el nombre.

El (o los) numerales son parte del nombre.
Así que puede crearse una tabla permanente llamada "libros", otra tabla temporal local llamada "#libros" y una tercera tabla temporal global denominada "##libros".

Características

- Utilizan el símbolo **##** como primer carácter en su nombre.
- Son visibles por cualquier usuario conectado al SQL Server.

Expresión de tabla común (CTE)

WITH common_table_expression

Especifica un conjunto de resultados temporal con nombre, conocido como expresión de tabla común (CTE). Se deriva de una consulta simple y se define en el ámbito de ejecución de una sola instrucción SELECT, INSERT, UPDATE o DELETE. Esta cláusula también se puede utilizar en una instrucción CREATE VIEW como parte de la instrucción SELECT que la define. Una expresión de tabla común puede incluir referencias a ella misma. Ésto se conoce como expresión de tabla común recursiva.

Instrucciones para crear y utilizar expresiones de tablas comunes

Las instrucciones siguientes se aplican a expresiones de tabla comunes no recursivas. Para obtener instrucciones que se aplican a expresiones de tabla comunes recursivas, vea *"Instrucciones para definir y usar expresiones de tabla comunes recursivas"* más adelante.

- Una expresión CTE debe ir seguida de una única instrucción SELECT, INSERT, UPDATE o DELETE que haga referencia a una parte o a la totalidad de sus columnas. Una expresión CTE también se puede especificar en una instrucción CREATE VIEW como parte de la instrucción SELECT de definición de la vista.
- Se pueden especificar varias definiciones de consulta de CTE en una CTE no recursiva. Las definiciones deben combinarse mediante uno de estos operadores de conjuntos: UNION ALL, UNION, INTERSECT y EXCEPT.

Instrucciones para crear y utilizar expresiones de tablas comunes

- Una expresión CTE puede hacer referencia a ella misma y a otras expresiones CTE previamente definidas en la misma cláusula WITH. No se permite la referencia adelantada.
- No se permite especificar más de una cláusula WITH en una expresión CTE. En la definición de CTE no se pueden usar las siguientes cláusulas:
  - ORDER BY (excepto cuando se especifica una cláusula TOP)
  - INTO
  - Cláusula OPTION con sugerencias de consulta
  - FOR BROWSE
- Cuando se utiliza una expresión CTE en una instrucción que forma parte de un lote, la instrucción que la precede debe ir seguida de punto y coma.
- Una consulta que haga referencia a una CTE se puede utilizar para definir un cursor.

Instrucciones para crear y utilizar expresiones de tablas comunes

- En la expresión CTE se puede hacer referencia a tablas de servidores remotos.
- Cuando se ejecuta una CTE, todas las sugerencias que hagan referencia a ella pueden entrar en conflicto con otras sugerencias detectadas cuando la CTE tiene acceso a sus tablas subyacentes, de la misma manera que las sugerencias que hacen referencia a vistas en las consultas. En ese caso, la consulta devuelve un error.

Sintaxis

```
WITH Sales_CTE (SalesPersonID, SalesOrderID, SalesYear)
AS
(
  SELECT SalesPersonID, SalesOrderID, YEAR(OrderDate) AS
SalesYear
  FROM Sales.SalesOrderHeader
  WHERE SalesOrderID=43659
)

SELECT * FROM Sales_CTE;
-----------------------------------------------
SalesPersonIDSalesOrderID SalesYear
279                 43659         2011
```

Inserción masiva de datos

BULK INSERT

Select Into

La instrucción SELECT INTO crea una nueva tabla y la llena con el conjunto de resultados de la instrucción SELECT.

SELECT INTO se puede emplear para combinar datos de varias tablas o vistas en una tabla.

También se puede utilizar para crear una nueva tabla que contenga datos seleccionados de un servidor vinculado.

La estructura de la nueva tabla se define con los atributos de las expresiones de la lista de selección.

Consideraciones

La cláusula SELECT INTO se puede utilizar si se encuentra habilitada la opción de select into/bulkcopy en las opciones de la base de datos.

Se puede utilizar SELECT INTO para grabar resultados en una tabla temporal.

No debe existir ninguna tabla con el mismo nombre que se utilizará en el SELECT INTO.

Las nuevas columnas que se crearán junto con la tabla no pueden contener espacios.

BULK INSERT

Sintaxis

```
SELECT TerritoryID, Name
INTO #Territorios FROM Sales.SalesTerritory WHERE CountryRegionCode='US';

SELECT * FROM #Territorios
ORDER BY TerritoryID;
----------------------
TerritoryID        Name
1                Northwest
2                Northeast
3                Central
4                Southwest
5                Southeast
```

¡Muchas gracias!

¡Sigamos trabajando!