Funciones de Sistema

Funciones Matemáticas

Las siguientes funciones escalares realizan un cálculo, normalmente basado en valores de entrada proporcionados como argumentos, y devuelven un valor numérico:

ABS	ROUND	CEILING
SIGN	FLOOR	

Sintaxis

ABS

Función matemática que devuelve el valor absoluto positivo de una expresión numérica específica. (ABS cambia los valores negativos por valores positivos. ABS no tiene ningún efecto en los valores cero o positivos).

```
SELECT ABS(1), ABS(-1), ABS(0)
--------- --------- -------------------------
1         1         0.00
```

ROUND

Devuelve un valor numérico, redondeado a la longitud o precisión especificadas.

```
SELECT ROUND(4.55,1),  ROUND(4.54,1)
--------- ---------
4.60      4.50
```

CEILING

Devuelve el número entero más pequeño que sea mayor o igual que la expresión numérica especificada.

```
SELECT ROUND(4.55,2),  ROUND(4.54,2)
--------- --------- -------------------------
124.00    -123.00   0.00
```

SIGN

Devuelve el signo positivo (+1), cero (0) o negativo (-1) de la expresión especificada.

```
SELECT SIGN(-125), SIGN(0), SIGN(564);
--------- --------- -------------------------
-1        0         1
```

FLOOR
Devuelve el entero más grande que sea menor o igual que la expresión numérica especificada.

```
SELECT FLOOR(123.45), FLOOR(-123.45);
--------- ---------
123       -124
```

Funciones Fechas
En las secciones de este tema se describen todos los tipos de datos de las principales funciones de fecha y hora de Transact-SQL.

DATENAME	DATEPART	DAY
MONTH	YEAR	GETDATE
DATEDIFF	DATEADD	

Sintaxis

DATENAME
Esta función devuelve una cadena de caracteres que representa el parámetro *datepart* especificado del argumento *date* especificado.

```
SELECT DATENAME(YEAR, '2018-06-21')
       , DATENAME(MONTH, '2018-06-21')
       , DATENAME(DAY, '2018-06-21')
       , DATENAME(WEEKDAY, '2018-06-21');
--------- --------- --------- -------------------------
2018      Junio     21        Jueves
```

DATEPART
Devuelve un entero que representa el parámetro *datepart* especificado del parámetro *date* especificado.

```
SELECT DATEPART (YEAR, '2007-05-10')
      , DATEPART(MONTH, '2007-05-10')
      , DATEPART(DAY, '2007-05-10');
--------- --------- -------------------------
2007      5         21
```

DAY-MONTH-YEAR
Devuelve un entero que representa el *datepart* del argumento especificado.

```
SELECT YEAR('1995-08-19')
       , MONTH('1995-08-19')
       , DAY('1995-08-19');
--------- --------- -------------------------
1995      8         19
```

GETDATE

Devuelve la marca de tiempo del sistema de base de datos actual como un valor datetime.

```
SELECT GETDATE();
-----------------------
2018-06-05 01:44:08.640
```

DATEDIFF

Esta función devuelve el recuento (como un valor entero con firma) de los límites datepart que se han cruzado entre los valores *startdate* y *enddate* especificados.

```
SELECT DATEDIFF(YEAR, '2005-01-01', '2018-01-01');
-----------------------
13
```

DATEADD

Esta función agrega un valor especificado según el *datepart* determinado a una fecha y devuelve el valor modificado. El siguiente ejemplo resta un mes a la fecha.

```
SELECT DATEADD(MONTH, -1, '2006-08-01');
-----------------------
2006-07-01
```

Funciones de Caracteres

Las siguientes funciones escalares realizan una operación sobre un valor de cadena de entrada y devuelven un valor de cadena o un valor numérico:

RIGHT	LEFT	CHARINDEX
SUBSTRING	REPLACE	LEN
LOWER	UPPER	

Sintaxis

RIGHT

Devuelve la parte derecha de una cadena de caracteres con el número de caracteres especificado.

```
SELECT LastName, RIGHT(LastName,2)
FROM Person.Person
WHERE BusinessEntityID=1;
--------- ---------
Sanchez   ez
```

LEFT

```
SELECT LastName, LEFT(LastName,2)
FROM Person.Person
WHERE BusinessEntityID=1;
--------- ---------
Sanchez   Sa
```

SUBSTRING

Devuelve parte de una expresión de caracteres. El siguiente ejemplo se posiciona en el tercer carácter y extrae los dos caracteres siguientes

```
SELECT SUBSTRING('AABBCC',3,2);
---------
BB
```

CHARINDEX

Esta función busca una expresión de caracteres dentro de una segunda expresión de caracteres, y devuelve la posición inicial de la primera expresión si se encuentra.

```
SELECT CHARINDEX('B','ABC');
---------
2

SELECT CHARINDEX('D','ABC');
--------- ---------
```

```
0
```

REPLACE
Reemplaza todas las instancias de un valor de cadena especificado por otro valor de cadena. El siguiente ejemplo reemplaza las letras B por Z

```
SELECT REPLACE('AABB','B','Z');
---------
AAZZ
```

LEN
Devuelve el número de caracteres de la expresión de cadena especificada, excluidos los espacios en blanco finales.

```
SELECT REPLACE('AABB');
---------
4
```

LOWER-UPPER
Devuelve una expresión de caracteres después de convertir en minúsculas los datos de caracteres en mayúsculas y viceversa.

```
SELECT LOWER('AABB');
---------
aabb

SELECT UPPER('aabb');
---------
AABB
```

Funciones definidas por el Usuario

Una Función puede tener cero o más parámetros de entrada y puede devolver tanto un valor escalar o tabular.
La adición de funciones al lenguaje del SQL solucionara los problemas de reutilización del código y dando mayor flexibilidad al programar las consultas de SQL.
SQL Server admite las funciones definidas por el usuario y las funciones del sistema integradas. Al igual que las funciones en los lenguajes de programación, las funciones definidas por el usuario

son rutinas que aceptan parámetros, realizan una acción, como un cálculo complejo y devuelve el resultado de esa acción como un valor. El valor devuelto puede ser un valor escalar único o un conjunto de resultados.
SQL Server provee diferentes tipos de funciones:

Funciones Escalares	Devuelven un solo valor del tipo definido en la cláusula RETURNS. Este tipo de funciones es sintácticamente similar a las funciones del Sistema tales como COUNT() o MAX().
Funciones Tabulares En Línea	Devuelven una tabla que es el resultado de una sola sentencia SELECT. Es similar a una Vista, pero ofrecen más flexibilidad que una Vista porque se le pueden suministrar parámetros a la Función.
Funciones Tabulares Multi-sentencia	Devuelve una Tabla construida por una o más sentencias Transact-SQL. Es similar a un Procedimiento Almacenado, pero a diferencia de éste último, la Vista puede referenciarse en la cláusula FROM de una sentencia SELECT como si se tratase de una Tabla.

Nota:
Para modificar o eliminar una Función, se usa una sintaxis similar a la usada con cualquier otro objeto de la base de datos. Use ALTER FUNCTION para modificar sus funciones. Luego de creadas, use DROP FUNCTION para eliminar Funciones de la base de datos.
Una Función Escalar puede ser invocada en cualquier lugar del código donde se necesite una expresión del mismo tipo de datos.

La siguiente tabla contiene ejemplos de dónde puede usar Funciones Escalares:

ÁREA	EJEMPLO
Consultas	- Como una expresión en la lista de columnas de una sentencia SELECT. - Como una expresión en la cláusula WHERE o HAVING. - Como una expresión en una cláusula GROUP BY. - Como una expresión en una cláusula ORDER BY. - Como una expresión en la cláusula SET de una sentencia UPDATE. - Como una expresión en la cláusula VALUES de una sentencia INSERT.
Definición de Tablas	- En restricciones (constraints) CHECK - En definiciones DEFAULT - En columnas calculadas
Sentencias Transact-SQL	- En operadores de asignación - En expresiones boolean de sentencias de control de flujo - En expresiones CASE - En sentencias PRINT (sólo si devuelve string)
Procedimientos y otras Funciones	- Como argumentos - En la sentencia RETURN de un Procedimiento (Sólo si devuelve INT) - En la sentencia RETURN de una Función (de tipo compatible con la expresión donde se usa)

Mostrar la definición de funciones definidas por el usuario de Transact-SQL

Sintaxis

```
SELECT definition, type
FROM sys.sql_modules AS m
JOIN sys.objects AS o ON m.object_id = o.object_id
    AND type IN ('FN', 'IF', 'TF');
GO
```

CREATE FUNCTION

Crea una función definida por el usuario en SQL Server y Base de datos SQL de Azure. Una función definida por el usuario es una rutina de Transact-SQL o Common Language Runtime (CLR) que acepta parámetros, realiza una acción, como un cálculo complejo, y devuelve el resultado de esa acción como un valor. El valor devuelto puede ser un valor escalar (único) o una tabla. Utilice esta instrucción para crear una rutina reutilizable que se pueda utilizar de estas formas:

- En instrucciones Transact-SQL como SELECT
- En las aplicaciones que llaman a la función
- En la definición de otra función definida por el usuario
- Para parametrizar una vista o mejorar la funcionalidad de una vista indizada
- Para definir una columna en una tabla
- Para definir una restricción CHECK en una columna
- Para reemplazar un procedimiento almacenado
- Usar una función insertada como predicado de filtro de la directiva de seguridad

Tipos de Funciones

Funciones escalares

Una Función Escalar devuelve un solo valor. El valor que devuelve es un tipo de dato escalar, tal como int, money, varchar, real, etc.
Las funciones escalares son las más parecidas a las funciones que se utilizan dentro de los lenguajes de programación. Pueden ser utilizadas en cualquier lugar incluso incorporado dentro de sentencias SQL.
El cuerpo de la Función, es definido por un bloque BEGIN…END , contiene las sentencias Transact-SQL que devuelven el valor.

El siguiente ejemplo crea una Función Escalar que totaliza todas las ventas de un producto en la base AdventureWorks y retorna el total como un INT.

Sintaxis

```
IF OBJECT_ID ('Sales.fnTotalVentas', 'FN' ) IS NOT NULL
    DROP FUNCTION Sales.fnTotalVentas;
GO

-- CREACIÓN DE FUNCIÓN ESCALAR
```

```
CREATE FUNCTION Sales.fnTotalVentas
(
      @ProductID INT
)
RETURNS INT
AS
BEGIN

        DECLARE @ret int;

        SELECT
                  @ret = SUM(OrderQty)
        FROM
                  Sales.SalesOrderDetail
        WHERE ProductID = @ProductID;

        RETURN @ret;
END
GO

SELECT Sales.fnTotalVentas(712) AS Producto;
```

Funciones Tabulares En Línea

Podemos usar Funciones Tabulares En Línea para obtener la funcionalidad de Vistas parametrizadas.

Las funciones de tabla en línea son las funciones que devuelven un ROWSET . Osea que la salida de una de estas funciones es una simple declaración SELECT.

Una de las limitaciones de las vistas, es que no podemos incluir parámetros en la vista que estamos creando. Usualmente lo resolvemos con un filtro WHERE cuando invocamos a la Vista.

Sin embargo, esto puede requerir la construcción de una cadena de caracteres que constituye la consulta SELECT a ejecutar, aumentando la complejidad de la solución. Usted puede conseguir la funcionalidad de una Vista parametrizada, usando una Función Tabular.

Considere las siguientes características de las Funciones Tabulares En Línea:

- La cláusula RETURNS específica TABLE como el tipo de dato retornado.
- El conjunto de resultados de la sentencia SELECT define el formato de salida.
- La cláusula RETURN contiene un solo SELECT entre paréntesis.
- El cuerpo de la Función no necesita estar encerrado en un bloque BEGIN…END.
- El siguiente ejemplo crea una Función Tabular En Línea, que devuelve los nombres de los empleados para un Administrador en particular, en la base AdventureWorks.

Use una Función Tabular En Línea donde normalmente usaría una Vista, tal como en la cláusula FROM de una sentencia SELECT.

Características:

- La salida se puede utilizar a dentro de joins o querys como si fuera una tabla estándar.
- El tipo de datos que devuelve es TABLE.
- La cláusula RETURNS especifica una TABLA sin información adicional, ésta debe ser una simple sentencia SELECT.

Sintaxis

```
IF OBJECT_ID ('Sales.ifTotalVentas', 'IF' ) IS NOT NULL
    DROP FUNCTION [Sales].[ifTotalVentas];

GO

-- CREACION DE FUNCION TABULAR
CREATE FUNCTION Sales.ifTotalVentas
(
      @ProductID INT
)
RETURNS TABLE
AS
RETURN
(
            SELECT *
            FROM  Sales.SalesOrderDetail
            WHERE ProductID = @ProductID
);
GO

SELECT * FROM  [Sales].[ifTotalVentas](712);
GO
```

Funciones Tabulares Multi-Sentencia

Una Función Tabular Multi-Sentencia es una combinación de una Vista y un Procedimiento. Este tipo de Función Tabular puede usar lógica compleja y múltiples sentencias Transact-SQL para construir una Tabla.
Del mismo modo que usaría una Vista, puede usar estas Funciones en la cláusula FROM de Transact-SQL.

Considere las siguientes características de las funciones tabulares multi-sentencias:

- La sentencia RETURNS específica TABLE como tipo de dato a retornar y define un nombre y un formato para la tabla de salida.
- Un bloque BEGIN…END delimita el cuerpo de la función.

Características:

- Las funciones de tabla de multi sentencias son similares a los procedimientos almacenados excepto que devuelven una tabla.
- Este tipo de función se usa en situaciones donde se requiere más lógica y proceso.

Sintaxis

```
IF OBJECT_ID ('SALES.tfTotalVentas', 'TF' ) IS NOT NULL
    DROP FUNCTION SALES.tfTotalVentas;
GO

CREATE FUNCTION [Sales].tfTotalVentas
(
      @format NVARCHAR(9)
)
RETURNS @Empleados TABLE
(
    EmpleadoID    INT, Nombre   NVARCHAR(100)
)
AS
BEGIN
      IF (@format = 'SHORTNAME')
      BEGIN
            INSERT @Empleados
            SELECT BusinessEntityID, LastName
            FROM
                  HumanResources.vEmployee;
       END
      ELSE
      BEGIN
            INSERT @Empleados
            SELECT BusinessEntityID, (FirstName + ' ' + LastName)
            FROM
                  HumanResources.vEmployee;
      END

      RETURN
END
GO

SELECT * FROM Sales.tfTotalVentaS('SHORTNAME');
SELECT * FROM Sales.tfTotalVentaS('LONGNAME');
```