|  |
| --- |
|  |
| Cursos SQL Server 2008 R2 |
|  |
| *Funciones Escalares* |

Cursos SQL Server 2008 R2

# Ejemplos

#### 5.1 Escalares-6.SQL

DIFERENTES TIPOS DE LLAMADAS A LAS FUNCIONES CREADAS EN LOS GUIONES ANTERIORES

USE AdventureWorks

GO

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Uso de la función Quien()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- dbo.Quien

SELECT dbo.Quien() AS [Quién y de dónde]

GO

-- Creación de una tabla que usa "Quien" como una restricción DEFAULT

IF OBJECT\_ID('TestQuien') IS NOT NULL

DROP TABLE TestQuien

GO

CREATE TABLE TestQuien(

ID int IDENTITY (1,1)

PRIMARY KEY

, Nombre nvarchar(40)

, QuienDonde nvarchar(256)

DEFAULT dbo.Quien()

)

GO

INSERT INTO TestQuien (Nombre) VALUES ('Nuevo registro')

SELECT \* FROM TestQuien

GO

-- Creamos un trigger para saber quién y desde dónde se hace un cambio

CREATE TRIGGER trQuien

ON TestQuien

AFTER INSERT, UPDATE

AS

UPDATE TestQuien

SET QuienDonde = dbo.Quien()

FROM TestQuien T

JOIN Inserted I

ON T.ID = I.ID

GO

-- Probamos la funcionalidad del trigger

INSERT INTO TestQuien (Nombre, QuienDonde)

VALUES ('Más registros', 'Nadie de ningún sitio')

-- Coprobamos que lo ha hecho

SELECT \* FROM TestQuien

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Uso de la función ValorFuturo()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- dbo.ValorFuturo

SELECT 0.05 AS Interes

, 48 AS NPagos

, 1000 AS Pago

, 10000 AS Vactual

, 0 AS Tipo

, dbo.ValorFuturo(0.07, 36, 1000, 10000, 0) AS FV

#### 5.1 Escalares-7.SQL

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USO DE COLUMNAS COMO PARÁMETROS EN UNA LLAMADA

A UNA FUNCIÓN DEFINIDA POR EL USUARIO

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

USE AdventureWorks

GO

-- Utilización de la función ImporteTotal para obtener

-- información de la tabla "Sales.SalesOrderDetail"

SELECT SalesOrderID

, ProductID

, dbo.ImporteTotal(OrderQty, UnitPrice, UnitPriceDiscount) AS Total

FROM Sales.SalesOrderDetail

WHERE ProductID = 760

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

LLAMADA A FUNCIONES CON EXECUTE

En las llamadas se pueden omitir aquellos parámetros que tengan

un valor predeterminado y modificar el orden de los parámetros,

siempre que la llamada a la función sea con el nombre de los

parámetros (nominada)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

USE AdventureWorks

GO

--Declaramos una variable para guardar el resultado de la UDF

DECLARE @Total money

-- Utilizamos EXECUTE y suministramos valores para todos los parámetros

EXECUTE @Total = dbo.ImporteTotal 12, 25.4, 0.0

SELECT @Total

-- Usamos EXECUTE y omitimos el parámetro @Descuento

-- ya que tiene valor predeterminado

EXECUTE @Total = dbo.ImporteTotal 12, 25.4

SELECT @Total

-- Usamos EXECUTE y omitimos el nombre del propietario de la UDF

-- ya que está predeterminado para dbo.

EXECUTE @Total = ImporteTotal 12, 25.4

SELECT @Total

-- Usamos EXECUTE y suministramos valores para cada parámetro

-- indicando los nombres de parámetro

EXECUTE @Total = ImporteTotal

@Cantidad = 12

, @PrecioUnitario = 25.4

, @Descuento = 0.2

SELECT @Total

-- Usamos EXECUTE y suministramos valores para todos los parámetros

-- especificando los parámetros en orden y con nombre en cualquier orden

EXECUTE @Total = ImporteTotal

12

, @Descuento = 0.2

, @PrecioUnitario = 25.4

SELECT @Total

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

COMENTARIOS

Al llamar a una UDF con EXECUTE no hace falta cualificar la función

con el nombre del propietario.

Sin embargo, siempre es más eficiente cualificar los objetos usados,

ya que de este modo SQL Server no tiene que buscar primero entre los

objetos que pertenecen al usuario actual, antes de hacerlo con los

del usuario dbo.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# Más información

|  |  |
| --- | --- |
| C/ Miracruz, 10 (Bº de Gros) 20001 Donostia  Telf.: 943 275819  email: [seim@centroseim.com](mailto:seim@centroseim.com) |  |

Logotipo SEIM