



# BASE DE DATOS AVANZADO I

---

Unidad 4: Programación Transact SQL  
Tema 9: Manejo de Funciones de usuario





# Tema 09: Manejo de Funciones de usuario

---

## 4.2. Tema 9: Manejo de Funciones de usuario

### 4.2.1. Funciones del sistema

### 4.2.2. Funciones de usuario

- Funciones escalares
- Funciones de tabla
- Funciones multisentencia

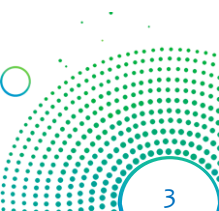




# Capacidades

---

1. Implementa procedimientos almacenados, funciones, en todos sus tipos, y triggers para garantizar el alto rendimiento de la base de datos, al hacer las transacciones





# 1. Funciones del sistema

*¿Qué son las funciones?*

- Son rutinas de programación que realizan operaciones y tratamiento con los datos, optimizando el rendimiento de los procesos y consultas.
- Las funciones son rutinas que permiten encapsular sentencias TRANSACT-SQL que se ejecutan frecuentemente.
- Existen 02 categorías de funciones en SQL Server:
  - Funciones Incorporadas del Sistema.
  - Funciones Definidas por el usuario





# 1. Funciones del sistema

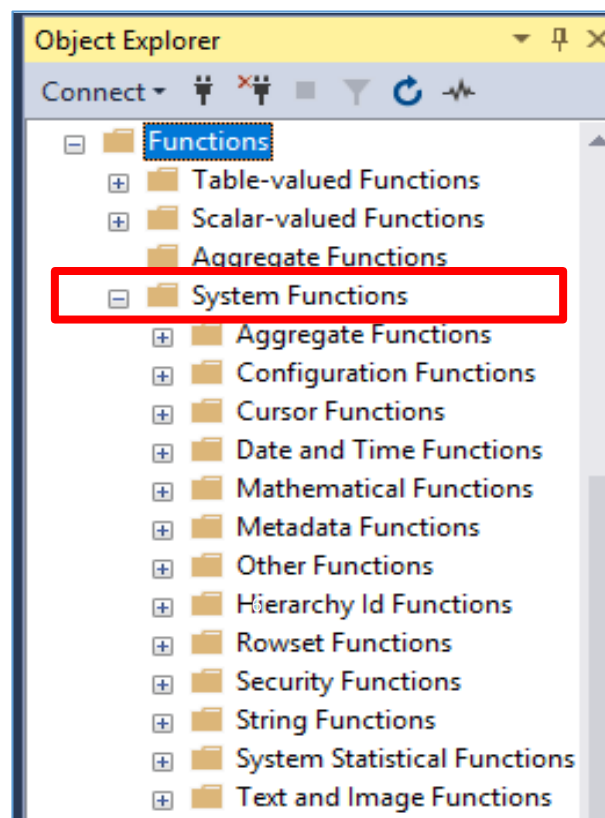
- Estas funciones se clasifican a su vez en:
  - Funciones de Texto.
  - Funciones Numéricas.
  - Funciones de Conversión.
  - Funciones de Fecha y Hora.
  - Funciones de Configuración.
  - Funciones de Cursor.





# 1. Funciones del sistema

- Las funciones integradas del sistema se pueden ubicar en el contenedor o carpeta programación de la Base de Datos (contenedor de objetos).





## 2. Funciones definidas por el usuario

- SQL Server proporciona al usuario la posibilidad de definir sus propias funciones, conocidas como UDF (user defined functions).
- Las funciones definidas por el usuario, en tiempo de ejecución de lenguaje TRANSACT-SQL o común (CLR), acepta parámetros, realiza una acción, como un cálculo complejo, y devuelve el resultado de esa acción como un valor.
- El valor de retorno puede ser un escalar (único) valor o una tabla.





## 2. Funciones definidas por el usuario

- Limitaciones y restricciones de las Funciones definidas por el usuario:
  - No se pueden utilizar para realizar acciones que modifican el estado de la base de datos.
  - No pueden tener una cláusula OUTPUT INTO que tenga una tabla como destino.
  - No pueden devolver varios conjuntos de resultados. Utilice un procedimiento almacenado si necesita devolver varios conjuntos de resultados.
  - El control de errores está restringido en una función definida por el usuario. Una UDF no admite TRY...CATCH, @ERROR o RAISERROR.
  - No pueden llamar a un procedimiento almacenado.







## 2. Funciones definidas por el usuario

- Las funciones de usuario, según el tipo de retorno se clasifican en:
  - Funciones escalares
  - Funciones con valores de tabla de varias instrucciones
  - Funciones con valores de tabla en línea





### 3. Funciones escalares

- Son aquellas funciones donde retornan un valor único: tipo de datos como int, Money, varchar, real, etc. Pueden ser utilizadas en cualquier lugar, incluso, incorporada dentro de las sentencias SQL.

- Sintaxis:

```
CREATE FUNCTION [propietario.] nombre_funcion  
  ([{ @parameter tipo de dato [= default]} [,..n]])  
    RETURNS valor_escalar  
  [AS]  
  BEGIN  
    cuerpo de la funcion  
    RETURN expresion_escalar  
  END
```





## 3. Funciones escalares

### Ejemplo 1

- Crear una función de parámetro vacío que retorne el precio promedio de todos los productos.

```
CREATE FUNCTION dbo.PrecioPromedio()  
    RETURNS decimal  
AS  
BEGIN  
    Declare @prom decimal  
    Select @prom=AVG(precioUNidad) From Compras.productos  
    RETURN @prom  
END  
GO
```






## 3. Funciones escalares

### Ejemplo 1

- Usando la función escalar.

```
--Usando la función  
PRINT 'El Precio promedio de los productos es: ' + str(dbo.PrecioPromedio(),3)  
GO
```

 Messages

El Precio promedio de los productos es: 29



## 3. Funciones escalares

### Ejemplo 1

- Usando la función en una consulta.

```
Select *  
From Compras.productos  
Where PrecioUnidad > dbo.PrecioPromedio()  
go
```

	IdProducto	NomProducto	IdProveedor	IdCategoria	CantxUnidad	PrecioUnidad	UnidadesEnExistencia	UnidadesEnPedido
1	7	Peras secas organicas del tio Bob	3	7	12 - paq. 1 kg	30	15	0
2	8	Salsa de arandanos Northwoods	3	2	12 - frascos 12l	40	6	0
3	9	Buey Mishi Kobe	4	6	18 - paq. 500 g	107	29	0
4	10	Pez espada	4	8	12 - frascos 200 ml	34	31	0
5	12	Queso Manchego La Pastora	5	4	10 - paq. 500 g	42	86	0
6	17	Cordero Alice Springs	7	6	20 - latas 1 kg	39	0	0



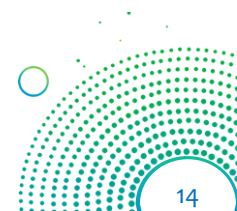


## 3. Funciones escalares

### Ejemplo 2

- Implemente un función escalar de devuelva la cantidad de pedidos efectuados a un determinado Cliente.

```
CREATE FUNCTION Ventas.fnCantPedidosXCliente
(@p_idCliente Char(5))
    RETURNS Smallint
AS
BEGIN
    Declare @v_ValorAretornar smallint=0
    Select @v_ValorAretornar = Count(*)
    From Ventas.pedidoscabe
    Where IdCliente=@p_idCliente
    RETURN @v_ValorAretornar
END
go
```






## 3. Funciones escalares

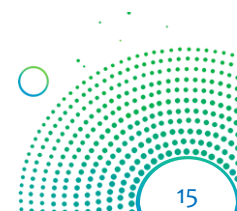
### Ejemplo 2

- Usando la función para conocer la cantidad de pedidos del cliente de código ALFKI.

```
Print 'La cantidad de Pedidos de ALFKI es: ' +  
  str(Ventas.fnCantPedidosXCliente ('ALFKI'), 3)  
go
```

 Messages

La cantidad de Pedidos de ALFKI es: 6





### 3. Funciones escalares

#### Ejemplo 2

- Usando la función en una consulta.

```
SELECT IdCliente [Id del Cliente],  
       NomCliente [Cliente],  
       DirCliente [Dirección del Cliente],  
       Ventas.fnCantPedidosXCliente(IdCliente) [Cantidad de Pedidos]  
FROM Ventas.clientes  
ORDER BY 4 Desc  
go
```

	Id del Cliente	Cliente	Dirección del Cliente	Cantidad de Pedidos
1	SAVEA	Save-a-lot Markets	187 Suffolk Ln.	31
2	ERNSH	Ernst Handel	Kirchgasse 6	30
3	QUICK	QUICK-Stop	Taucherstraße 10	28
4	HUNGO	Hungry Owl All-Night Grocers	8 Johnstown Road	19
5	FOLKO	Folk och fä HB	Åkergatan 24	19

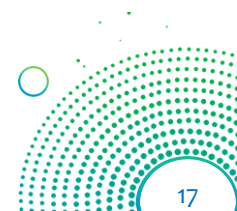




## 4. Funciones de tabla en línea

- Las funciones de tabla en línea son funciones que devuelven la salida de una simple declaración SELECT.
- La salida se puede utilizar con JOINS o queries como si fuera una tabla de estándar.
- Sintaxis:

```
CREATE FUNCTION [propietario.] nombre_funcion  
( [{ @parameter tipo de dato [= default]} [,..n]])  
RETURNS TABLE  
[AS]  
RETURN [(] Sentencia SQL [)]
```

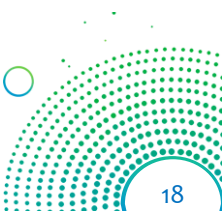




## 4. Funciones de tabla en línea

---

- *Ejemplo 1*
- Defina una función que liste los registros de los pedidos por un determinado año, incluya el nombre del producto, el precio que fue vendido y la cantidad vendida.



## 4. Funciones de tabla en línea

### Ejemplo 1

```
CREATE FUNCTION dbo.PedidosxAño
(@p_year int) RETURNS TABLE
AS
RETURN (Select  pc.idpedido      as 'Pedido',
                pc.FechaPedido  as 'Fecha de Pedido',
                p.NomProducto   as 'Producto',
                pd.PrecioUnidad as 'Precio de venta',
                pd.Cantidad     as 'Cantidad pedida'
            From  Ventas.pedidoscabe pc JOIN Ventas.pedidosdeta pd
            ON    pc.idpedido = pd.idpedido JOIN Compras.productos p
            ON    pd.idproducto = p.idproducto
            WHERE Year(FechaPedido) = @p_year)
go
```



## 4. Funciones de tabla en línea

### Ejemplo 1

Consultando la función de tabla en línea creada.

```
--Consultando la función de tabla en línea  
Select * from dbo.PedidosxAño(2010)  
go
```

Results		Messages			
	Pedido	Fecha de Pedido	Producto	Precio de venta	Cantidad pedida
1	11068	2010-09-04 00:00:00.000	Col fermentada Rössle	45	8
2	11068	2010-09-04 00:00:00.000	Cafe de Malasia	46	36
3	11068	2010-09-04 00:00:00.000	Salsa verde original Frankfurter	13	28
4	11069	2010-10-04 00:00:00.000	Licor verde Chartreuse	18	20
5	11070	2010-10-05 00:00:00.000	Te Dharamsala	18	40
6	11070	2010-10-05 00:00:00.000	Cerveza tibetana Barley	19	20
7	11070	2010-10-05 00:00:00.000	Postre de merengue Pavlova	17	30
8	11070	2010-10-05 00:00:00.000	Queso gorgonzola Telino	12	20



## 5. Funciones multisentencia

- Las funciones en línea de múltiples sentencias son similares a las funciones en línea excepto que el conjunto de resultados que devuelven puede estar compuesto por la ejecución de varias consultas SELECT.
- 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.



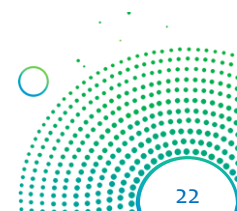


## 5. Funciones multisentencia

- Sintaxis:

```
CREATE FUNCTION [propietario.] nombre funcion
( [{ @parameter tipo de dato [= default]} [,..n]])
RETURNS @nombreVariableTabla TABLE
                                (campo1      tipo,
                                campo2      tipo,...
                                campon      tipo)

[AS]
BEGIN
    Cuerpo de la función
    RETURN
END
```





## 5. Funciones multisentencia

---

- *Ejemplo 1*
- Defina una función de multi sentencias de parámetro vacío que retorne los productos registrados en la base de datos (Id del producto, nombre, precio y unidades en existencias).



## 5. Funciones multisentencia

### Ejemplo 1

```
CREATE FUNCTION dbo.Inventario()  
RETURNS @v_tabla TABLE( v_idproducto int,  
                          v_nombre varchar(50),  
                          v_precio decimal, v_stock int)  
  
AS  
BEGIN  
    INSERT INTO @v_tabla  
    SELECT idproducto, nomProducto,  
           precioUnidad, UnidadesEnExistencia  
    FROM Compras.productos  
    RETURN  
END  
go
```





## 5. Funciones multisentencia

### Ejemplo 1

Ahora consultamos la función multi sentencias.

```
--Consultando la función de Multisentencias  
Select * from dbo.Inventario()  
go
```

Results		Messages		
	v_idproducto	v_nombre	v_precio	v_stock
1	1	Te Dharamsala	18	39
2	2	Cerveza tibetana Barley	19	17
3	3	Sirope de regaliz	10	13
4	4	Especias Cajun del chef Anton	22	53
5	5	Mezcla Gumbo del chef Anton	21	0
6	6	Memelada de grosellas de la abuela	25	120

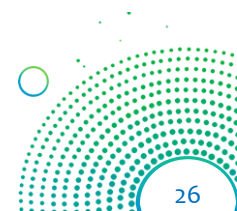


## 5. Funciones multisentencia

### *Ejemplo 2*

Defina una función que permita generar un reporte de ventas por empleado, por año.

En este proceso la función debe retornar: los datos del empleado, la cantidad de pedidos registrados y el monto total por empleado.



## 5. Funciones multisentencia

### Ejemplo 2

```
CREATE FUNCTION dbo.ReporteVentas(@p_year int)
RETURNS @v_tabla TABLE( v_id int, v_nombre varchar(50),
                        v_cantidad int, v_monto decimal)
AS
BEGIN
    INSERT INTO @v_tabla
    SELECT e.idempleado, e.ApeEmpleado, COUNT(*), SUM(precioUNidad*cantidad)
    FROM Ventas.pedidoscabe pc JOIN Ventas.pedidosdeta pd
    ON pc.idpedido = pd.idpedido JOIN RRHH.empleados e
    ON e.idempleado = pc.idempleado
    WHERE YEAR(pc.FechaPedido) = @p_year
    GROUP BY e.idempleado, e.ApeEmpleado
    RETURN
END
GO
```



## 5. Funciones multisentencia

### Ejemplo 2

Ahora consultamos la función de multi sentencias.

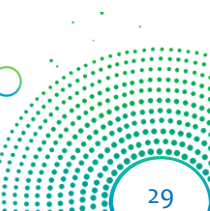
```
--Mostrar el reporte de ventas del año 2010  
Select * from dbo.ReporteVentas(2010)  
go
```

Results		Messages		
	v_id	v_nombre	v_cantidad	v_monto
1	1	Davolio	3	870
2	2	Fuller	6	2140
3	4	Peacock	4	5154
4	8	Callahan	3	2380



# Ejercicio 1

- En el departamento de sistemas, se solicita implementar una función escalar que permita generar un nuevo código a partir de la segunda letra del apellido, año de nacimiento, 2 primeras letra del nombre y las 3 últimas letras del apellido.



# Ejercicio 1: Solución

- Se creará la función escalar de la siguiente manera

```
CREATE FUNCTION dbo.fnGenCode
(
    @p_nombre      varchar(40),
    @p_apellido    varchar(40),
    @p_fechanac    date
)
Returns Varchar(15)
As
Begin
    Declare @v_code Varchar(10)
    set @v_code = Upper(SUBSTRING(@p_apellido,2,1)+
                        Ltrim(Str(year(@p_fechanac)))+
                        Left(@p_nombre,2)+Right(@p_apellido,3))
    Return @v_code
End
go
```

# Ejercicio 1: Solución

- Comprobando:

```
--Comprobar  
Print 'Código generado: ' +  
      dbo.fnGenCode('Luis','Alvarez','10/12/80')  
go
```

Messages  
Código generado: L1980LUREZ

# Ejercicio 1: Solución

- Comprobando en una consulta:

```
--Usando función en consulta
Select IdEmpleado, NomEmpleado, ApeEmpleado, FecNac,
       dbo.fnGenCode(NomEmpleado, ApeEmpleado, FecNac) as [NewCode]
from rrhh.empleados
go
```

Results		Messages			
	IdEmpleado	NomEmpleado	ApeEmpleado	FecNac	NewCode
1	1	Nancy	Davolio	1968-12-08 00:00:00.000	A1968NALIO
2	2	Andrew	Fuller	1952-02-19 00:00:00.000	U1952ANLER
3	3	Janet	Leverling	1963-08-30 00:00:00.000	E1963JAING
4	4	Margaret	Peacock	1958-09-19 00:00:00.000	E1958MAOCK
5	5	Steven	Buchanan	1955-03-04 00:00:00.000	U1955STNAN



## Ejercicio 2

- Implemente una función de tabla en línea, que muestre los 02 productos mas consumidos en un determinado país. Deberá recibir como parámetro el nombre del país.

## Ejercicio 2: Solución

```
B Create Function dbo.ConsumoXpais ( @p_nomPais varchar (max) )
    Returns Table
As
    Return (Select top 2 PD.IdProducto, PD.NomProducto,
                Sum (D.Cantidad) [Cantidad Consumida] --
            From Ventas .países PA Join Ventas .clientes C
                On PA.Idpais = C.idpais Join Ventas .pedidoscabe P
                On C.IdCliente = P.IdCliente Join Ventas .pedidosdeta D
                On P.IdPedido = D.IdPedido Join Compras .productos PD
                On D.IdProducto = PD.IdProducto
            Where PA.NombrePais = @p_nomPais
            Group By PD.IdProducto, PD.NomProducto
            Order By 3 dese
        )

go
```

## Ejercicio 2: Solución

- Comprobando la función:

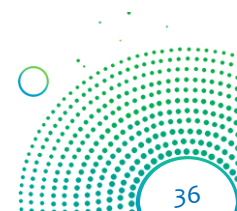
```
Select * from dbo.ConsumoXpais ('Argentina')  
go
```

Results		Messages	
	IdProducto	NomProducto	Cantidad Consumida
1	71	Crema de queso Fløtemys	135
2	36	Escabeche de arenque	127



# Conclusiones

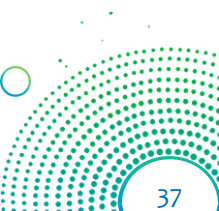
- Las funciones definidas por el usuario permiten optimizar el tiempo de respuestas en las operaciones con los datos.
- Existen algunas limitaciones que pueden resolverse con el uso
- de procedimientos almacenados.
  - Funciones Vs Procedimientos, las funciones garantizan un mejor rendimiento, pero solo devuelven un resultado, mientras que los procedimientos pueden retornar mas de un valor.
  - El uso de las funciones se pueden dar en consultas.





# Bibliografía

Microsoft (2017) CREATE FUNCTION (Transact-SQL). Recuperado de: <https://docs.microsoft.com/en-us/sql/t-sql/statements/create-function-transact-sql?view=sql-server-2017>



# GRACIAS

**SEDE LIMA CENTRO**

Av. Uruguay 514  
Cercado – Lima  
Teléfono: 419-2900

**SEDE INDEPENDENCIA**

Av. Carlos Izaguirre 233  
Independencia – Lima  
Teléfono: 633-5555

**SEDE BREÑA**

Av. Brasil 714 – 792  
(CC La Rambla – Piso 3)  
Breña – Lima  
Teléfono: 633-5555

**SEDE TRUJILLO**

Calle Borgoño 361  
Trujillo  
Teléfono: (044) 60-2000

**SEDE SAN JUAN DE LURIGANCHO**

Av. Próceres de la Independencia 3023-3043  
San Juan de Lurigancho – Lima  
Teléfono: 633-5555

**SEDE BELLAVISTA**

Av. Mariscal Oscar R. Benavides 3866 – 4070  
(CC Mall Aventura Plaza)  
Bellavista – Callao  
Teléfono: 633-5555

**SEDE AREQUIPA**

Av. Porongoche 500  
(CC Mall Aventura Plaza)  
Paucarpata - Arequipa  
Teléfono: (054) 60-3535