Funciones SQL Server

En SQL Server, las funciones son bloques de código SQL reutilizables que realizan una tarea específica y devuelven un resultado. Las funciones permiten encapsular la lógica de negocio y facilitan la modularización del código SQL. Hay dos tipos principales de funciones: funciones definidas por el usuario (UDF) y funciones del sistema.

Funciones Definidas por el Usuario (UDF):

1. **Funciones Escalares:**
    - Una función escalar devuelve un solo valor.
    - Estructura básica:

```
CREATE FUNCTION NombreFuncion
(
    @Parametro1 TipoDato1,
    @Parametro2 TipoDato2
)
RETURNS TipoDatoResultado
AS
BEGIN
    -- Lógica de la función
    RETURN Resultado;
END;
```

    - Ejemplo:

```
CREATE FUNCTION SumarDosNumeros
(
    @Numero1 INT,
    @Numero2 INT
)
RETURNS INT
AS
BEGIN
```

```
    RETURN @Numero1 + @Numero2;
END;
```

2. **Funciones de Tabla:**
   - Una función de tabla devuelve un conjunto de resultados (una tabla).
   - Estructura básica:

```
CREATE FUNCTION NombreFuncion
(
    @Parametro1 TipoDato1,
    @Parametro2 TipoDato2
)
RETURNS TABLE
AS
RETURN
(
    -- Lógica de la función que retorna una tabla
);
```

   - Ejemplo:

```
CREATE FUNCTION ObtenerEmpleadosPorDepartamento
(
    @DepartamentoID INT
)
RETURNS TABLE
AS
RETURN
(
    SELECT *
    FROM Employees
    WHERE DepartmentID = @DepartamentoID
);
```

Funciones del Sistema:

SQL Server proporciona varias funciones del sistema que realizan tareas comunes. Algunas de estas funciones incluyen `LEN` para obtener la longitud de

una cadena, `GETDATE` para obtener la fecha y hora actuales, etc. Estas funciones ya están incorporadas en el sistema y están listas para su uso.

Uso de Funciones:

Una vez que has definido una función, puedes utilizarla en consultas como si fuera una columna o una tabla. Por ejemplo:

```
-- Uso de una función escalar
SELECT dbo.SumarDosNumeros(5, 7) AS Resultado;

-- Uso de una función de tabla
SELECT *
FROM dbo.ObtenerEmpleadosPorDepartamento(1);
```

Función Escalar con Lógica más Compleja:

En este ejemplo, creamos una función escalar que calcula la edad de una persona basándose en su fecha de nacimiento.

```
CREATE FUNCTION CalcularEdad
(
    @FechaNacimiento DATE
)
RETURNS INT
AS
BEGIN
    DECLARE @Edad INT;
    SET @Edad = DATEDIFF(YEAR, @FechaNacimiento, GETDATE())
-
                CASE
                    WHEN (MONTH(@FechaNacimiento) > MONTH(G
ETDATE())) OR
                         (MONTH(@FechaNacimiento) = MONTH(G
ETDATE()) AND DAY(@FechaNacimiento) > DAY(GETDATE()))
                    THEN 1
                    ELSE 0
                END;
```

```
    RETURN @Edad;
END;
```

Puedes utilizar esta función para obtener la edad de una persona:

```
SELECT dbo.CalcularEdad('1990-05-15') AS Edad;
```

Función de Tabla con Parámetros y Joins:

En este ejemplo, creamos una función de tabla que devuelve información sobre pedidos de clientes basándonos en un rango de fechas.

```
CREATE FUNCTION ObtenerPedidosPorFecha
(
    @FechaInicio DATE,
    @FechaFin DATE
)
RETURNS TABLE
AS
RETURN
(
    SELECT
        Orders.OrderID,
        Customers.CustomerName,
        Orders.OrderDate,
        OrderDetails.ProductID,
        OrderDetails.Quantity
    FROM
        Orders
    JOIN
        Customers ON Orders.CustomerID = Customers.Customer
ID
    JOIN
        OrderDetails ON Orders.OrderID = OrderDetails.Order
ID
    WHERE
        Orders.OrderDate BETWEEN @FechaInicio AND @FechaFin
);
```

Puedes usar esta función de tabla en una consulta:

```
SELECT * FROM dbo.ObtenerPedidosPorFecha('2022-01-01', '202
2-12-31');
```