

# SQL Programming

## Módulo 7 - Cursores

# Cursores

# Declare cursor

Define los atributos de un cursor de servidor de Transact-SQL, como su comportamiento de desplazamiento y la consulta utilizada para generar el conjunto de resultados sobre el que opera el cursor.

DECLARE CURSOR acepta tanto una sintaxis basada en el estándar ISO como una sintaxis que utiliza un conjunto de extensiones Transact-SQL.



# @@FETCH\_STATUS

Esta función devuelve el estado de la última instrucción FETCH del cursor emitida contra cualquier cursor abierto actualmente por la conexión.

## Argumentos

- **NEXT**

Devuelve la fila de resultados inmediatamente posterior a la fila actual, y aumenta la fila actual a la fila devuelta. Si FETCH NEXT es la primera operación de captura en un cursor, se devuelve la primera fila del conjunto de resultados. NEXT es la opción predeterminada para la captura de cursores.

- **PRIOR**

Devuelve la fila de resultados inmediatamente anterior a la fila actual, y reduce la fila actual a la fila devuelta. Si FETCH PRIOR es la primera operación de captura en un cursor, no se devuelve ninguna fila y el cursor queda posicionado delante de la primera fila.

- **FIRST**

Devuelve la primera fila del cursor y la convierte en la fila actual.

- **LAST**

Devuelve la última fila del cursor y la convierte en la fila actual.

# Procedimientos almacenados del sistema

Nombre	Descripción
<b>sp_cursor_list</b>	Devuelve la lista de los cursores que están visibles actualmente en la conexión y sus atributos.
<b>sp_describe_cursor</b>	Describe los atributos de un cursor, por ejemplo si es de solo avance o de desplazamiento.
<b>sp_describe_cursor_columns</b>	Describe los atributos de las columnas en el conjunto de resultados del cursor.
<b>sp_describe_cursor_tables</b>	Describe las tablas base a las que tiene acceso el cursor.

# Procedimientos almacenados del sistema



## Sintaxis

```
IF OBJECT_ID (N'tempdb..#pvt', N'U') IS NOT NULL
    DROP TABLE #pvt;

CREATE TABLE #pvt (ID INT, Monto MONEY, Anio INT);

DECLARE @Id CHAR(12);
DECLARE @Monto MONEY;
DECLARE @Cantidad INT;
DECLARE @Anio INT;
DECLARE Cursor_Titulos CURSOR
FOR
    SELECT SalesOrderID, OrderQty, LineTotal, YEAR(ModifiedDate)
    FROM Sales.SalesOrderDetail

OPEN Cursor_Titulos
FETCH NEXT FROM Cursor_Titulos
INTO @Id, @Cantidad, @Monto, @Anio
WHILE @@FETCH_STATUS = 0
BEGIN
    IF @Cantidad > 10
    BEGIN
        INSERT INTO #pvt
        SELECT @Id, @Monto, @Anio
    END

    FETCH NEXT FROM Cursor_Titulos
    INTO @Id, @Cantidad, @Monto, @Anio
END

CLOSE Cursor_Titulos;
DEALLOCATE Cursor_Titulos;

SELECT * FROM #pvt
```

# PIVOT & UNPIVOT

Se pueden usar los operadores relacionales PIVOT y UNPIVOT para modificar una expresión con valores de tabla en otra tabla. PIVOT gira una expresión con valores de tabla convirtiendo los valores únicos de una columna de la expresión en varias columnas en la salida y realiza agregaciones donde son necesarias en cualquier valor de columna restante que se quiera en la salida final. UNPIVOT realiza la operación contraria a PIVOT girando las columnas de una expresión con valores de tabla a valores de columna.

La sintaxis de PIVOT es más sencilla y legible que la sintaxis que se puede especificar en una serie compleja de instrucciones SELECT...CASE.

Para obtener una descripción completa de la sintaxis de PIVOT, vea [FROM \(Transact-SQL\)](#).



## PIVOT & UNPIVOT

La sintaxis siguiente resume cómo se usa el operador PIVOT.

En el ejemplo toma la tabla cargada en el cursor y transpone el resultado.

### Sintaxis

```
SELECT TOP 10 ID, 2010, 2011, 2012, 2013, 2014, 2015
FROM
(
    SELECT ID, Monto, Anio
    FROM #pvt
) AS SOURCE
PIVOT
(
    SUM(Monto)
    FOR Anio IN ( 2010, 2011, 2012, 2013, 2014, 2015)
) AS REP
```

ID	2011	2012	2013	2014
44294	89,3475	NULL	NULL	NULL
47033	NULL	401,8424	NULL	NULL
51124	NULL	NULL	29679,2729	NULL
69532	NULL	NULL	NULL	658,256



# ¡Muchas gracias!

¡Sigamos trabajando!