



Prof. Flavio Bevilacqua

Mail: flabevy88@gmail.com

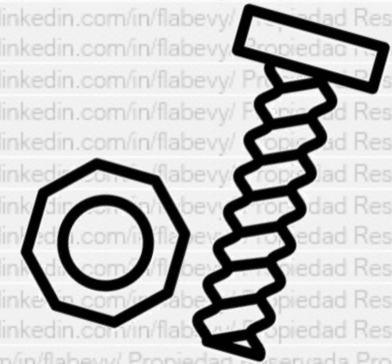
Linkedin: www.linkedin.com/in/flabevy



SUBCONSULTAS

SUBCONSULTAS

- El concepto Subconsulta hace referencia a una sentencia de Consulta contenida dentro de la sentencia principal.
- La subconsulta son aplicables a los comandos WHERE y HAVING.
- Dentro de una sola sentencia SQL se pueden especificar hasta 16 subconsultas y dentro de una subconsulta se pueden especificar subconsultas.
- En el proceso de ejecución del código, el motor de la base de datos resuelve:



SUBCONSULTAS

- En el proceso de ejecución del código, el motor de la base de datos resuelve primero la subconsulta para posteriormente ejecutar la sentencia principal.
- De modo tal que dicho proceso tiene ciertas similitudes con las funciones anidadas.

--lista los campos ProductID, ProductName,UnitsInStock,UnitPrice de todos

--los productos vendidos en una cantidad mayor e igual a 100

```
SELECT ProductID, ProductName,UnitsInStock,UnitPrice
```

```
FROM Products
```

```
WHERE ProductID IN
```

```
( SELECT ProductID
```

```
FROM [ORDER DETAILS]
```

```
WHERE Quantity>=100 ) ORDER BY ProductName
```



OVER PARTITION

OVER PARTITION

- Determina las particiones y el orden de un conjunto de filas antes de que se aplique la función de ventana asociada. Es decir, la cláusula OVER define una ventana o un conjunto de filas definido por el usuario en un conjunto de resultados de la consulta.
- Una función de ventana calcula entonces un valor para cada fila de la ventana.
- Puede utilizar la cláusula OVER con funciones para calcular valores agregados tales como medias móviles, agregados acumulados, totales acumulados o N elementos superiores por resultados del grupo.

ARGUMENTOS

➤ La función de Ventanas podrían tener los siguientes argumentos al utilizarlos con la clausula OVER.

➤ PARTITION BY divide el conjunto de resultados de la consulta en particiones.


➤ ORDER BY que define el orden lógico de las filas dentro de cada partición de conjunto de resultados.

➤ ROWS/RANGE, el cual limita aun mas las filas de la partición especificando los puntos inicial y final. Requiere de los argumentos ORDER BY.

PARTITION BY

➤ Especifica la columna a partir de la cual se particiona el conjunto de filas. *value_expression* solo puede hacer referencia a columnas disponibles a través de la cláusula FROM. *value_expression* no puede hacer referencia a expresiones ni a alias de la lista de selección. *value_expression* puede ser una expresión de columna, una subconsulta escalar, una función escalar o una variable definida por el usuario.

SQL

 Copiar

```
select
    object_id, type
    , [min] = min(object_id) over(partition by type)
    , [max] = max(object_id) over(partition by type)
from sys.objects
```


CASE

CASE

- Las expresiones CASE se admiten en los módulos T-SQL compilados de forma nativa. En el ejemplo siguiente se muestra una forma de usar la expresión CASE en una consulta.
- En los ejemplos de código se usa una variable de tabla para crear un conjunto de resultados único. Esto solo es adecuado al procesar un número limitado de filas, ya que implica la creación de una copia adicional de las filas de datos.

CASE

```
-- Query using a CASE expression in a natively compiled stored procedure.
CREATE PROCEDURE dbo.usp_SOHOnlineOrderResult
WITH NATIVE_COMPILATION, SCHEMABINDING, EXECUTE AS OWNER
AS BEGIN ATOMIC WITH (TRANSACTION ISOLATION LEVEL = SNAPSHOT, LANGUAGE=N
SELECT
    SalesOrderID,
    CASE (OnlineOrderFlag)
    WHEN 1 THEN N'Order placed online by customer'
    ELSE N'Order placed by sales person'
    END
FROM Sales.SalesOrderHeader_inmem
END
GO

EXEC dbo.usp_SOHOnlineOrderResult
GO
```

FIN