


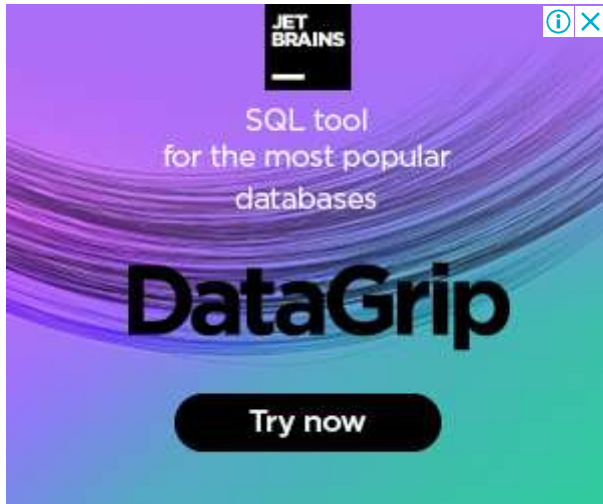




Subconsultas en SQL

Por [Claudio](#)

 23 de enero de 2006  [19 Comentarios](#)  [Lenguaje SQL](#)



Defenimos lo que significa subconsulta y mostramos las diferentes subconsultas que se pueden hacer.

Una subconsulta es una instrucción SELECT anidada dentro de una instrucción SELECT, SELECT...INTO, INSERT...INTO, DELETE, o UPDATE o dentro de otra subconsulta. Puede utilizar tres formas de sintaxis para crear una subconsulta:

comparación [ANY | ALL | SOME] (instrucción sql) expresión [NOT] IN (instrucción sql)
[NOT] EXISTS (instrucción sql)

En donde:

comparación Es una expresión y un operador de comparación que compara la expresión con el resultado de la subconsulta.

expresión Es una expresión por la que se busca el conjunto resultante de la subconsulta.

instrucción SQL Es una instrucción SELECT, que sigue el mismo formato y reglas que cualquier otra instrucción SELECT. Debe ir entre paréntesis.

Se puede utilizar una subconsulta en lugar de una expresión en la lista de campos de una instrucción SELECT o en una cláusula WHERE o HAVING. En una subconsulta, se utiliza una

instrucción SELECT o en una cláusula WHERE o HAVING. En una subconsulta, se utiliza una instrucción SELECT para proporcionar un conjunto de uno o más valores especificados para evaluar en la expresión de la cláusula WHERE o HAVING.

Se puede utilizar el predicado ANY o SOME, los cuales son sinónimos, para recuperar registros de la consulta principal, que satisfagan la comparación con cualquier otro registro recuperado en la subconsulta. El ejemplo siguiente devuelve todos los productos cuyo precio unitario es mayor que el de cualquier producto vendido con un descuento igual o mayor al 25 por ciento:

```
SELECT *
FROM
    Productos
WHERE
    PrecioUnidad
    ANY
    (
        SELECT
            PrecioUnidad
        FROM
            DetallePedido
        WHERE
            Descuento = 0.25
    )
```

El predicado ALL se utiliza para recuperar únicamente aquellos registros de la consulta principal que satisfacen la comparación con todos los registros recuperados en la subconsulta. Si se cambia ANY por ALL en el ejemplo anterior, la consulta devolverá únicamente aquellos productos cuyo precio unitario sea mayor que el de todos los productos vendidos con un descuento igual o mayor al 25 por ciento. Esto es mucho más restrictivo.

El predicado IN se emplea para recuperar únicamente aquellos registros de la consulta principal para los que algunos registros de la subconsulta contienen un valor igual. El ejemplo siguiente devuelve todos los productos vendidos con un descuento igual o mayor al 25 por ciento:

```
SELECT *
FROM
    Productos
WHERE
    IDProducto
    IN
    (
        SELECT
            IDProducto
        FROM
            DetallePedido
        WHERE
            Descuento = 0.25
    )
```

Inversamente se puede utilizar NOT IN para recuperar únicamente aquellos registros de la consulta principal para los que no hay ningún registro de la subconsulta que contenga un valor igual.

El predicado EXISTS (con la palabra reservada NOT opcional) se utiliza en comparaciones de verdad/falso para determinar si la subconsulta devuelve algún registro. Supongamos que deseamos recuperar todos aquellos clientes que hayan realizado al menos un pedido:

```
SELECT
  Clientes.Compañía, Clientes.Teléfono
FROM
  Clientes
WHERE EXISTS (
  SELECT
  FROM
    Pedidos
  WHERE
    Pedidos.IdPedido = Clientes.IdCliente
)
```

Esta consulta es equivalente a esta otra:

```
SELECT
  Clientes.Compañía, Clientes.Teléfono
FROM
  Clientes
WHERE
  IdClientes
  IN
  (
  SELECT
    Pedidos.IdCliente
  FROM
    Pedidos
  )
```

Se puede utilizar también alias del nombre de la tabla en una subconsulta para referirse a tablas listadas en la cláusula FROM fuera de la subconsulta. El ejemplo siguiente devuelve los nombres de los empleados cuyo salario es igual o mayor que el salario medio de todos los empleados con el mismo título. A la tabla Empleados se le ha dado el alias T1:

```
SELECT
  Apellido, Nombre, Titulo, Salario
FROM
  Empleados AS T1
WHERE
  Salario =
```

```

)
SELECT
    Avg(Salario)
FROM
    Empleados
WHERE
    T1.Titulo = Empleados.Titulo
)
ORDER BY Titulo

```

En el ejemplo anterior, la palabra reservada AS es opcional.

```

SELECT
    Apellidos, Nombre, Cargo, Salario
FROM
    Empleados
WHERE
    Cargo LIKE 'Agente Ven*'
    AND
    Salario ALL
    (
        SELECT
            Salario
        FROM
            Empleados
        WHERE
            Cargo LIKE '*Jefe*'
        OR
            Cargo LIKE '*Director*'
    )

```

(Obtiene una lista con el nombre, cargo y salario de todos los agentes de ventas cuyo salario es mayor que el de todos los jefes y directores.)

```

SELECT DISTINCT
    NombreProducto, Precio_Unidad
FROM
    Productos
WHERE
    PrecioUnidad =
    (
        SELECT
            PrecioUnidad
        FROM
            Productos
        WHERE

```

```

NombreProducto = 'Almíbar anisado'
)

```

(Obtiene una lista con el nombre y el precio unitario de todos los productos con el mismo precio que el almíbar anisado.)

```

SELECT DISTINCT
    NombreContacto, NombreCompania, CargoContacto, Telefono
FROM
    Clientes
WHERE
    IdCliente IN (
        SELECT DISTINCT IdCliente
        FROM Pedidos
        WHERE FechaPedido <#07/01/1993#
    )

```

(Obtiene una lista de las compañías y los contactos de todos los clientes que han realizado un pedido en el segundo trimestre de 1993.)

```

SELECT
    Nombre, Apellidos
FROM
    Empleados AS E
WHERE EXISTS
    (
        SELECT *
        FROM
            Pedidos AS O
        WHERE O.IdEmpleado = E.IdEmpleado
    )

```

(Selecciona el nombre de todos los empleados que han reservado al menos un pedido.)

```

SELECT DISTINCT
    Pedidos.Id_Producto, Pedidos.Cantidad,
    (
        SELECT
            Productos.Nombre
        FROM
            Productos
        WHERE
            Productos.IdProducto = Pedidos.IdProducto
    ) AS ElProducto
FROM
    Pedidos

```

```

WHERE
    Pedidos.Cantidad = 150
ORDER BY
    Pedidos.Id_Producto

```

(Recupera el Código del Producto y la Cantidad pedida de la tabla pedidos, extrayendo el nombre del producto de la tabla de productos.)

```

SELECT
    NumVuelo, Plazas
FROM
    Vuelos
WHERE
    Origen = 'Madrid'
    AND Exists (
        SELECT T1.NumVuelo FROM Vuelos AS T1
        WHERE T1.PlazasLibres > 0 AND T1.NumVuelo=Vuelos.NumVuelo)

```

(Recupera números de vuelo y capacidades de aquellos vuelos con destino Madrid y plazas libres

Supongamos ahora que tenemos una tabla con los identificadores de todos nuestros productos y el stock de cada uno de ellos. En otra tabla se encuentran todos los pedidos que tenemos pendientes de servir. Se trata de averiguar que productos no se podemos servir por falta de stock.

```

SELECT
    PedidosPendientes.Nombre
FROM
    PedidosPendientes
GROUP BY
    PedidosPendientes.Nombre
HAVING
    SUM(PedidosPendientes.Cantidad <
    (
        SELECT
            Productos.Stock
        FROM
            Productos
        WHERE
            Productos.IdProducto = PedidosPendientes.IdProducto
    )
    )

```

Supongamos que en nuestra tabla de empleados deseamos buscar todas las mujeres cuya edad sea mayor a la de cualquier hombre:

```

SELECT
    Empleados.Nombre

```

```

empleados.nombre
FROM
    Empleados
WHERE
    Sexo = 'M' AND Edad > ANY
        (SELECT Empleados.Edad FROM Empleados WHERE Sexo ='H')

```

ó lo que sería lo mismo:

```

SELECT
    Empleados.Nombre
FROM
    Empleados
WHERE
    Sexo = 'M' AND Edad >
        (SELECT Max( Empleados.Edad )FROM Empleados WHERE Sexo ='H')

```

La siguiente tabla muestra algún ejemplo del operador ANY y ALL

Valor 1 Operador Valor 2 Resultado

3	> ANY	(2,5,7)	Cierto
3	= ANY	(2,5,7)	Falso
3	= ANY	(2,3,5,7)	Cierto
3	> ALL	(2,5,7)	Falso
3	< ALL	(5,6,7)	Falso

El operacion =ANY es equivalente al operador IN, ambos devuelven el mismo resultado.

--	--	--	--	--

Autor
Claudio

Subir 

Manual

Tutorial de SQL