

## [9.1] introducción a las subconsultas

El uso de subconsultas es una técnica que permite utilizar el resultado de una tabla SELECT en otra consulta SELECT. Permite solucionar consultas complejas mediante el uso de resultados previos conseguidos a través de otra consulta.

El SELECT que se coloca en el interior de otro SELECT se conoce con el término de *SUBSELECT*. Ese SUBSELECT se puede colocar dentro de las cláusulas **WHERE**, **HAVING**, **FROM** o **JOIN**.

## [9.2] uso de subconsultas en instrucciones SELECT

### [9.2.1] uso de subconsultas simples

Las subconsultas simples son aquellas que devuelven una única fila. Si además devuelven una única columna, se las llama **subconsultas escalares**, ya que devuelven un único valor.

La sintaxis es:

```
SELECT listaExpresiones
FROM tabla
WHERE expresión OPERADOR
      (SELECT listaExpresiones
      FROM tabla);
```

El operador puede ser **>**, **<**, **>=**, **<=**, **!=**, **=** o **IN**.

Ejemplo:

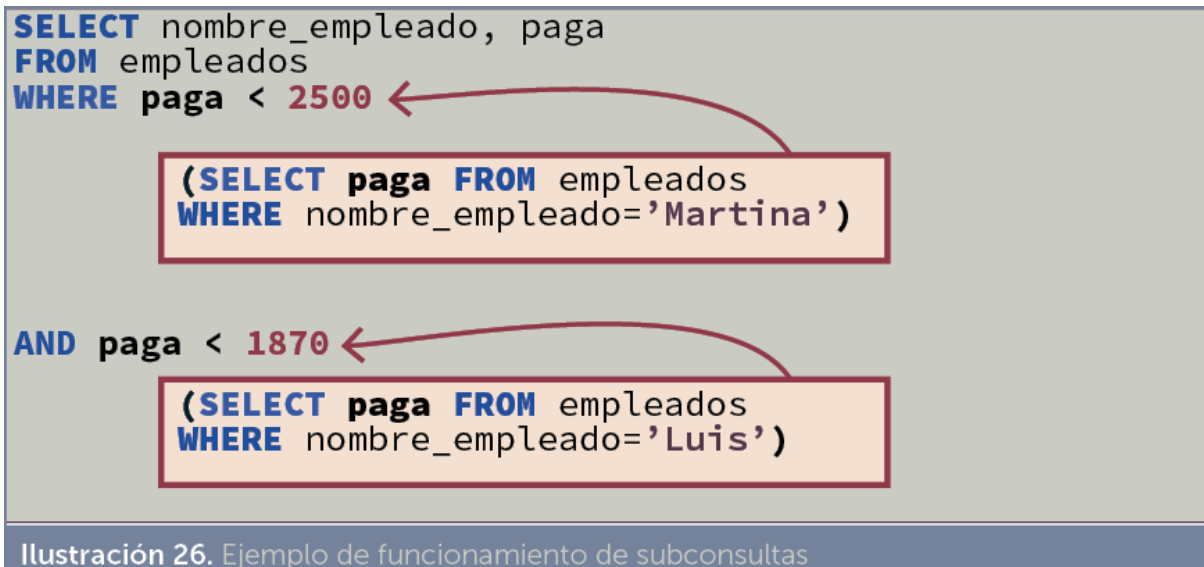
```
SELECT nombre_empleado, paga
FROM empleados
WHERE paga <
    (SELECT paga FROM empleados
     WHERE nombre_empleado='Martina')
;
```

Esa consulta muestra el *nombre* y *paga* de los empleados cuya paga es menor que la de la empleada *Martina*. Para que funcione esta consulta, la subconsulta solo puede devolver un valor (solo puede haber una empleada que se llame Martina).

Se pueden usar subconsultas las veces que haga falta:

```
SELECT nombre_empleado, paga
FROM empleados
WHERE paga <
    (SELECT paga FROM empleados
     WHERE nombre_empleado='Martina')
AND paga >
    (SELECT paga FROM empleado
     WHERE nombre_empleado='Luis');
```

En realidad lo primero que hace la base de datos es calcular el resultado de la subconsulta:



La última consulta obtiene los empleados cuyas pagas estén entre lo que gana Luís (1870 euros) y lo que gana Martina (2500) .

Las subconsultas siempre se deben encerrar entre paréntesis y se deberían (aunque no es obligatorio, sí altamente recomendable) colocar a la derecha del operador relacional.

**Una subconsulta que utilice los valores >, <, >=, ... tiene que devolver un único valor**, de otro modo ocurre un error.

Además **tienen que devolver el mismo tipo y número de datos** para relacionar la subconsulta con la consulta que la utiliza (no puede ocurrir que la subconsulta tenga dos columnas y ese resultado se compare usando una sola columna en la consulta general).

## [9.2.2] uso de subconsultas de múltiples filas

En el apartado anterior se comentaba que las subconsultas sólo pueden devolver una fila. Pero a veces se necesitan consultas del tipo: *mostrar el sueldo y nombre de los empleados cuyo sueldo supera al de cualquier empleado del departamento de ventas.*

La subconsulta necesaria para ese resultado mostraría **todos** los sueldos del departamento de ventas. Pero no podremos utilizar un operador de comparación directamente ya que esa subconsulta devuelve más de una fila. La solución a esto es utilizar instrucciones especiales entre el operador y la consulta, que permiten el uso de subconsultas de varias filas.

Esas instrucciones son:

Instrucción	Significado
<b>ANY</b> o <b>SOME</b>	Compara con cualquier registro de la subconsulta. La instrucción es válida si hay un registro en la subconsulta que permite que la comparación sea cierta. Se suele utilizar la palabra ANY (SOME es un sinónimo)
<b>ALL</b>	Compara con todos los registros de la consulta. La instrucción resulta cierta si es cierta toda comparación con los registros de la subconsulta
<b>IN</b>	No usa comparador, ya que sirve para comprobar si un valor se encuentra en el resultado de la subconsulta
<b>NOT IN</b>	Comprueba si un valor no se encuentra en una subconsulta

Ejemplo:

```
SELECT nombre, sueldo
FROM empleados
WHERE sueldo >= ALL (SELECT sueldo FROM
empleados) ;
```

La consulta anterior obtiene el empleado que más cobra. Otro ejemplo:

```
SELECT nombre FROM empleados
WHERE dni IN (SELECT dni FROM directivos);
```

En ese caso se obtienen los nombres de los empleados cuyos *dni* están en la tabla de directivos.

Si se necesita comparar dos columnas en una consulta IN, se hace de esta forma:

```
SELECT nombre FROM empleados
WHERE (cod1,cod2) IN (SELECT cod1,cod2 FROM
directivos);
```

### [9.2.3]consultas correlacionadas

En las subconsultas a veces se puede desear poder utilizar datos procedentes de la consulta principal. Eso es posible utilizando el alias de la tabla que queremos usar de la consulta principal.

Por ejemplo, supongamos que deseamos obtener de una base de datos geográfica, el nombre y la población de las localidades que sean las más pobladas de su provincia. Es decir, las localidades cuya población es la mayor de su provincia. Para ello necesitamos comparar la población de cada localidad con la de todas las localidades de su provincia. Supongamos que la tabla de las localidades almacena el nombre, población y el número de la provincia a la que pertenecen.

La consulta sería:

```

SELECT l.nombre, poblacion
FROM localidades l
WHERE poblacion >= ALL(
    SELECT poblacion
    FROM localidades l2
    WHERE l2.n_provincia = l.n_provincia
)a

```

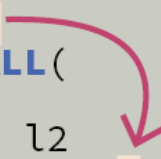


Ilustración 27. Ejemplo de funcionamiento de subconsultas correlacionadas

En el código anterior se observa que dentro de la subconsulta usamos el alias / correspondiente a la tabla de localidades de la consulta principal (por eso se le ha puesto como alias /2 a la tabla localidades en la subconsulta).

## [9.2.4] consultas EXISTS

Este operador devuelve verdadero si la consulta que le sigue devuelve algún valor. Si no, devuelve falso. Se utiliza normalmente mediante consultas correlacionadas. Ejemplo:

```

SELECT tipo, modelo, precio_venta
FROM piezas p
WHERE EXISTS (
    SELECT tipo, modelo FROM existencias
    WHERE tipo = p.tipo AND
    modelo = p.modelo) ;

```

Esta consulta devuelve las piezas que se encuentran en la tabla de existencias (es igual al ejemplo comentado en el apartado subconsultas sobre múltiples valores).

La consulta contraria es :

```
SELECT tipo, modelo, precio_venta
FROM piezas p
WHERE NOT EXISTS (
    SELECT tipo, modelo FROM existencias
    WHERE tipo=p.tipo AND
    modelo=p.modelo) ;
```

Normalmente las consultas EXISTS se pueden realizar de alguna otra forma con otros operadores.

## [9.3] uso de subconsultas

# SELECT en instrucciones DML y DDL

A pesar del poco ilustrativo título de este apartado, la idea es sencilla. Se trata de cómo utilizar instrucciones SELECT dentro de las instrucciones DML (**INSERT**, **DELETE** o **UPDATE**) o incluso dentro de otros apartados.

### [9.3.1] relleno de registros a partir de filas de una consulta

Hay un tipo de consulta, llamada de adición de datos, que permite rellenar datos de una tabla copiando el resultado de una consulta. Se hace mediante la instrucción **INSERT** y, en definitiva, permite copiar datos de una tabla a otra.

Ese relleno se basa en una consulta SELECT que poseerá los datos a añadir. El orden y tipo de las columnas que resultan del SELECT debe de coincidir con el orden y tipo de las columnas de la instrucción INSERT.

Sintaxis:

```
INSERT INTO tabla (columna1, columna2,...)
SELECT expresioCompatibleColumna1,
expresionCompatibleColumna2,...
FROM listaDeTablas
[...otras cláusulas del SELECT...]
```

Ejemplo:

```
INSERT INTO clientes2014 (dni, nombre, localidad, direccion)
SELECT dni, nombre, localidad, direccion
FROM clientes
WHERE problemas=0;
```

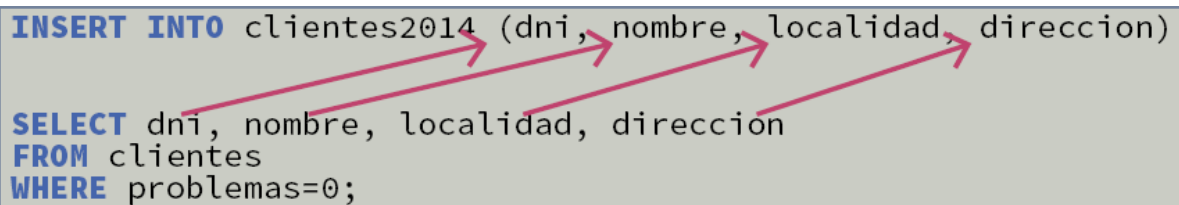


Ilustración 28. Ejemplo de funcionamiento de la instrucción INSERT SELECT

Lógicamente las columnas del SELECT se tienen que corresponder (en cuanto al tipo de datos y posición) con las columnas a rellenar mediante INSERT (observar las flechas).

## [9.3.2] subconsultas en la instrucción UPDATE

### subconsultas en el apartado WHERE

La instrucción **UPDATE** permite modificar filas. Es muy habitual el uso de la cláusula **WHERE** para indicar las filas que se modificarán. Esta cláusula se puede utilizar con las mismas posibilidades que en el caso del SELECT, por lo que es posible utilizar subconsultas. Por ejemplo:

```
UPDATE empleados
SET sueldo=sueldo*1.10
WHERE id_seccion =(SELECT id_seccion FROM
secciones
```



```
WHERE  
nom_seccion='Producción' );
```

Esta instrucción aumenta un 10% el sueldo de los empleados de la sección llamada *Producción*.

Es posible utilizar subconsultas correlacionadas.