



SQL ESTÁNDAR BÁSICO

Presentación #12. Subconsultas

A veces tenemos que realizar operaciones más complejas con los datos, operaciones en las que nos interesaría ayudarnos de una primera consulta auxiliar que extrajera la información en la que nos queremos basar. Esta consulta auxiliar recibe el nombre de "subconsulta" o "subquery".

Ejecutemos el siguiente bloque de instrucciones SQL:

```
drop table if exists facturas;  
drop table if exists detalles;
```

```
create table facturas(  
    numero int not null,  
    fecha date,  
    cliente varchar(30),  
    primary key(numero)  
);
```

```
create table detalles(  
    numerofactura int not null,  
    numeroitem int not null,  
    articulo varchar(30),  
    precio decimal(5,2),  
    cantidad int,  
    primary key(numerofactura,numeroitem)  
);
```

```
insert into facturas values(1200,'2018-01-15','Juan Lopez');  
insert into facturas values(1201,'2018-01-15','Luis Torres');  
insert into facturas values(1202,'2018-01-15','Ana Garcia');  
insert into facturas values(1300,'2018-01-20','Juan Lopez');
```

```
insert into detalles values(1200,1,'lapiz',1,100);  
insert into detalles values(1200,2,'goma',0.5,150);  
insert into detalles values(1201,1,'regla',1.5,80);  
insert into detalles values(1201,2,'goma',0.5,200);  
insert into detalles values(1201,3,'cuaderno',4,90);  
insert into detalles values(1202,1,'lapiz',1,200);  
insert into detalles values(1202,2,'escuadra',2,100);  
insert into detalles values(1300,1,'lapiz',1,300);
```



SUBCONSULTAS

De Traslado de Valor

Sucede cuando una consulta facilita uno o más valores en algún criterio de otra consulta.

Ejemplo:

```
select
numeroitem,
articulo,
precio,
(select avg(precio) from detalles) as promedio
from detalles
where
precio > (select avg(precio) from detalles);
```

De Exists o Not Exists

Sucede cuando se comparan conjuntos de datos entre dos consultas para evaluar si hay relación o no de subconjuntos en los registros.

Ejemplos:

```
select cliente,numero
from facturas as f
where exists
(select * from detalles as d
where f.numero=d.numeroofactura
and d.articulo='lapiz');
```

```
select cliente,numero
from facturas as f
where not exists
(select * from detalles as d
where f.numero=d.numeroofactura
and d.articulo='lapiz');
```

De In o Not In

Sucede cuando se comparan los valores en uno de los campos de una consulta contra una lista de valores obtenidos en otra consulta.

Ejemplos:



```
select cliente,numero
from facturas
where numero in
(select distinct numerofactura from detalles);
```

```
select cliente,numero
from facturas
where numero not in
(select distinct numerofactura from detalles);
```

De **SOME**, **ANY**, **ALL**

Sucede cuando se comparan los valores de una consulta en pruebas relacionales con los valores de otra consulta.

La palabra clave **ANY**, que debe seguir a un operador de comparación (relacional), significa “return TRUE si la comparación es TRUE para **ANY** (cualquiera) de los valores en la columna que retorna la subconsulta”.

Por ejemplo:

```
SELECT s1 FROM t1 WHERE s1 > ANY (SELECT s1 FROM t2);
```

Suponga que hay un registro en una tabla **t1** que contiene (10). La expresión es TRUE si la tabla **t2** contiene (21, 14, 7) ya que hay un valor 7 en **t2** que es menor que 10. La expresión es FALSE si la tabla **t2** contiene (20, 10), o si la tabla **t2** está vacía. La expresión es UNKNOWN si la tabla **t2** contiene (NULL, NULL, NULL).

Ejemplo:

```
SELECT cliente, numero FROM facturas WHERE numero = ANY (SELECT numerofactura FROM detalles
WHERE articulo='escuadra');
```

La palabra **SOME** es un alias para **ANY**. Por lo tanto, estos dos comandos son el mismo:

```
SELECT s1 FROM t1 WHERE s1 <> ANY (SELECT s1 FROM t2);
```

```
SELECT s1 FROM t1 WHERE s1 <> SOME (SELECT s1 FROM t2);
```

El uso de la palabra **SOME** es raro, pero este ejemplo muestra cómo puede ser útil. Para la mayoría de gente, la frase en inglés “a is not equal to any b” significa “there is no b which is equal to a,” pero eso no es lo que quiere decir la sintaxis SQL. La sintaxis significa “there is some b to which a is not equal.” Usando **<> SOME** en su lugar ayuda a asegurar que todo el mundo entiende el significado de la consulta.



Ejemplo:

```
SELECT cliente, numero FROM facturas WHERE numero <> SOME (SELECT numerofactura FROM  
detalles WHERE articulo='lapiz');
```

La palabra ALL, que debe seguir a un operador de comparación, significa “return TRUE si la comparación es TRUE para ALL todos los valores en la columna que retorna la subconsulta”. Por ejemplo:

```
SELECT s1 FROM t1 WHERE s1 > ALL (SELECT s1 FROM t2);
```

Suponga que hay un registro en la tabla t1 que contiene (10). La expresión es TRUE si la tabla t2 contiene (-5, 0, +5) ya que 10 es mayor que los otros tres valores en t2. La expresión es FALSE si la tabla t2 contiene (12, 6, NULL, -100) ya que hay un único valor 12 en la tabla t2 mayor que 10. La expresión es UNKNOWN si la tabla t2 contiene (0, NULL, 1).

Finalmente, si la tabla t2 está vacía, el resultado es TRUE. Puede pensar que el resultado debería ser UNKNOWN, pero lo sentimos, es TRUE. Así, aunque extraño, el siguiente comando es TRUE cuando la tabla t2 está vacía:

```
SELECT * FROM t1 WHERE 1 > ALL (SELECT s1 FROM t2);
```

Ejemplo:

```
SELECT cliente, numero FROM facturas WHERE numero > ALL (SELECT numerofactura FROM detalles  
WHERE numerofactura<1300);
```