



SUBCONSULTAS

Curso de Bases de datos.
Jesús Reyes Carvajal

Una subconsulta es una instrucción SELECT anidada dentro de una instrucción SELECT, INSERT...INTO, DELETE, o UPDATE o dentro de otra subconsulta.

Sintaxis básica

- Primero se ejecuta la subconsulta y luego la consulta principal.
- La subconsulta arroja un resultado, el cual sera cotejado por la consulta principal.

Consulta principal

subconsulta

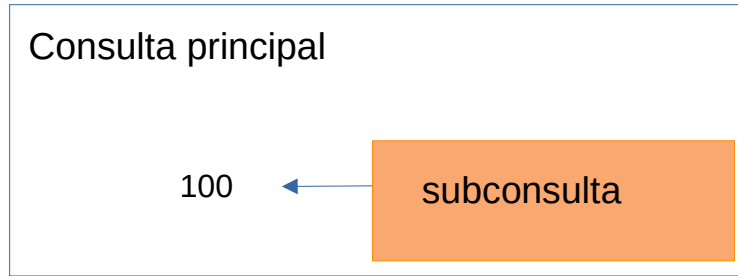
```
SELECT column
FROM   from_item
WHERE  condition (SELECT column
                  FROM   from_item
                  WHERE  condition);
```

CONSIDERACIONES PARA USAR SUBCONSULTAS

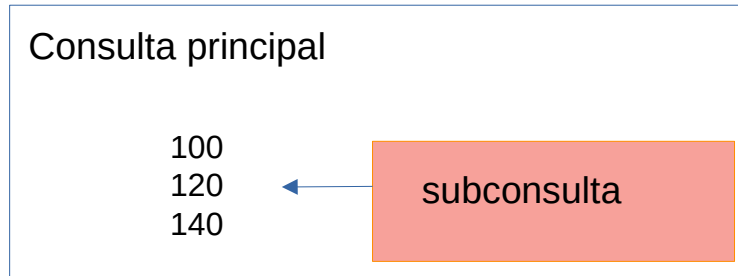
- La lista de selección de una subconsulta que va luego de un operador de comparación puede incluir sólo una expresión o campo (excepto si se emplea "exists" y "in").
- Si el "where" de la consulta exterior incluye un campo, este debe ser compatible con el campo en la lista de selección de la subconsulta.
- Las subconsultas luego de un operador de comparación (que no es seguido por "any" o "all") no pueden incluir cláusulas "group by" ni "having".
- "distinct" no puede usarse con subconsultas que incluyan "group by".
- Una subconsulta puede estar anidada dentro del "where" o "having" de una consulta externa o dentro de otra subconsulta.
- Si una tabla se nombra solamente en un subconsulta y no en la consulta externa, los campos no serán incluidos en la salida (en la lista de selección de la consulta externa).

TIPOS DE SUBCONSULTAS

- Retorna un valor escalar o un registro.



- Retorna múltiples registros o valores.



Las subconsultas con retorno de un escalar utilizan los operadores relaciones

OPERADOR	SIGNIFICADO
<	Menor que
<=	Menor o igual
>	Mayor
>=	Mayor o igual
=	Igual
<>	Diferente

SUBCONSULTAS CON RETORNO DE UN VALOR o REGISTRO

Las subconsultas simples son aquellas que devuelven una fila o un valor.

Consultar los estudiantes donde la edad es menor que la edad de Adelaida y mayor que la edad de Carlos Alberto.

```
select nomb_est,apel_est from estudiantes
where edad > (select edad from estudiantes where nomb_est='Carlos Alberto')
and edad < (select edad from estudiantes where nomb_est='Adelaida');
```

Consultar la diferencia de edad con respecto al estudiante de mayor edad.

```
SELECT nomb_est,(SELECT MAX(edad) FROM estudiantes) - edad AS
diferencia FROM estudiantes;
```

ESTUDIANTES

Id	nomb_est	apel_est	edad
1	Carlos Alberto	Carranza	23
2	Oscar Andres	torres	27
3	Andres Felipe	Borda	37
4	Lucero	Cortez	55
5	Adelaida	Jimenez	48

nomb_est	apel_est
Oscar Andres	torres
Andres Felipe	Borda

nomb_est	diferencia
Carlos Alberto	32
Oscar Andres	28
Andres Felipe	18
Lucero	0
Adelaida	7

SUBCONSULTAS CON RETORNO DE MÚLTIPLES REGISTROS

- Retornan mas de un registro.
- Utilizan los operadores de comparación.
- Se puede utilizar el operador NOT en cualquier de los siguiente operadores.

OPERADOR	SIGNIFICADO
IN	Compara si uno o mas de los elementos retornados por la subconsulta coinciden con los de la lista de la consulta principal.
NOT IN	Compara si un valor no se encuentra en una subconsulta.
ANY	Compara el valor de la consulta principal con cualquiera de los valores retornados por la subconsulta.
ALL	Compara el valor de la consulta principal con todos los valores retornados por la subconsulta.

SUBCONSULTA CON IN

ESTUDIANTES

id	nomb_est	apel_est	edad	cod_ciu
5	Adelaida	Jimenez	48	1
1	Carlos Alberto	Carranza	23	1
2	Oscar Andres	torres	27	1
3	Andres Felipe	Borda	37	2
4	Lucero	Cortez	55	3

CIUDADES

cod_ciu	nomb_ciu	poblacion
1	Villavicencio	3000
2	Restrepo	1500
3	Cumaral	1600
4	Bogotá	12000
5	Cali	8000

Nombre de las ciudades que están referenciadas en la tabla estudiantes.

```
SELECT nomb_ciu FROM ciudades  
WHERE cod_ciu IN (SELECT cod_ciu FROM estudiantes);
```

Consulta principal

nomb_ciud	cod_ciu
Villavicencio	1
Restrepo	2
Cumaral	3
Bogotá	4
Cali	5

Retorno de subconsulta

cod_ciu
1
1
1
2
3

nomb_ciu
Villavicencio
Restrepo
Cumaral



Otra forma seria, pero ya no es subconsulta:

```
SELECT DISTINCT nomb_ciu FROM ciudades c,estudiantes e  
WHERE c.cod_ciu=e.cod_ciu;
```

SUBCONSULTA CON NOT IN

ESTUDIANTES

id	nomb_est	apel_est	edad	cod_ciu
5	Adelaida	Jimenez	48	1
1	Carlos Alberto	Carranza	23	1
2	Oscar Andres	torres	27	1
3	Andres Felipe	Borda	37	2
4	Lucero	Cortez	55	3

CIUDADES

cod_ciu	nomb_ciu	poblacion
1	Villavicencio	3000
2	Restrepo	1500
3	Cumaral	1600
4	Bogotá	12000
5	Cali	8000

Nombre de las ciudades que no aparecen como sitio de vivienda de los estudiantes.

```
SELECT nomb_ciu FROM ciudades
WHERE cod_ciu NOT IN (SELECT cod_ciu FROM estudiantes);
```

Resultado final

nomb_ciu
Bogota
Cali

Consulta principal

nomb_ciud	cod_ciu
Villavicencio	1
Restrepo	2
Cumaral	3
Bogotá	4
Cali	5

Retorno de subconsulta

cod_ciu
1
1
1
2
3

SUBCONSULTA CON ALL

Listar el nombre del estudiante que tiene mayor edad.

```
SELECT nomb_est, edad FROM estudiantes  
WHERE edad >= ALL (SELECT edad FROM estudiantes);
```

ESTUDIANTES

id	nomb_est	apel_est	edad	cod_ciu
5	Adelaida	Jimenez	48	1
1	Carlos Alberto	Carranza	23	1
2	Oscar Andres	torres	27	1
3	Andres Felipe	Borda	37	2
4	Lucero	Cortez	55	3

Resultado final

nomb_est	edad
Lucero	55

Consulta principal

nomb_est	edad
Adelaida	48
Carlos Alberto	23
Oscar Andres	27
Andres Felipe	37
Lucero	55

Retorno de subconsulta

edad
48
23
27
37
55

Utilizar "<> ALL" es lo mismo que emplear "NOT IN".

Lista los empleados que tienen un salario menor que todos los empleados de IT_PROG.

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary < ALL (SELECT salary
                     FROM   employees
                     WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```

employee_id	last_name	job_id	salary
115	Khoo	PU_CLERK	3100
116	Baida	PU_CLERK	2900
117	Tobias	PU_CLERK	2800
118	Himuro	PU_CLERK	2600
119	Colmenares	PU_CLERK	2500
125	Nayer	ST_CLERK	3200
126	Mikkilineni	ST_CLERK	2700
127	Landry	ST_CLERK	2400
128	Markle	ST_CLERK	2200
...			

(44 rows)

salary
9000
6000
4800
4800
4200
(5 rows)

En la consulta principal se deben excluir los empleados del puesto IT_PROG(para que no haga cotejo con el mismo).

SUBCONSULTA CON ANY

ESTUDIANTES

id	nomb_est	apel_est	edad	cod_ciu
5	Adelaida	Jimenez	48	1
1	Carlos Alberto	Carranza	23	1
2	Oscar Andres	torres	27	1
3	Andres Felipe	Borda	37	2
4	Lucero	Cortez	55	3
6	Pedro	Nel	23	3
7	Carolina	Ospina	21	3

CIUDADES

cod_ciu	nomb_ciu	poblacion
1	Villavicencio	3000
2	Restrepo	1500
3	Cumaral	1600
4	Bogotá	12000
5	Cali	8000

Nombre de los estudiantes de Cumaral que son menores de alguno de los estudiantes de Villavicencio.

```
SELECT nomb_est FROM estudiantes e  
JOIN ciudades c ON c.cod_ciu=e.cod_ciu  
WHERE edad < ANY
```

```
(SELECT edad FROM estudiantes e JOIN ciudades c1 ON e.cod_ciu=c1.cod_ciu WHERE c1.nomb_ciu='Villavicencio')  
AND c.nomb_ciu='Cumaral';
```

Resultado final

nomb_est
Pedro
Carolina



Consulta principal

nomb_est	edad
Lucero	55
Pedro	23
Carolina	21

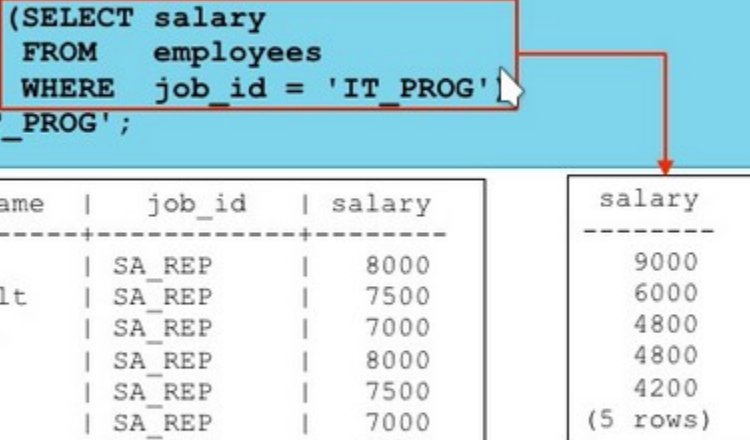
Retorno de subconsulta

edad
48
23
27

Utilizar "**= ANY**" es lo mismo que emplear "**IN**".

Lista los empleados que tienen un salario menor que cualquier empleado de IT_PROG.

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary < ANY (SELECT salary
                     FROM   employees
                     WHERE  job_id = 'IT_PROG')
AND    job_id <> 'IT_PROG';
```



employee_id	last_name	job_id	salary
153	Olsen	SA_REP	8000
154	Cambrault	SA_REP	7500
155	Tuvault	SA_REP	7000
159	Smith	SA_REP	8000
160	Doran	SA_REP	7500
161	Sewall	SA_REP	7000
164	Marvins	SA_REP	7200
165	Lee	SA_REP	6800
166	Ande	SA_REP	6400
...			

(76 rows)

salary
9000
6000
4800
4800
4200
(5 rows)

En la consulta principal se deben excluir los empleados del puesto IT_PROG.

SUBCONSULTA CON EXISTS

Nombre de los estudiantes que viven en la ciudad de Villavicencio.

```
SELECT e.nomb_est FROM estudiantes e
WHERE EXISTS (SELECT * FROM ciudades c WHERE
e.cod_ciu=c.cod_ciu AND nomb_ciu='Villavicencio');
```

Otra forma seria, pero ya no es una subconsulta:

```
SELECT e.nomb_est FROM estudiantes e,ciudades c
WHERE e.cod_ciu=c.cod_ciu AND nomb_ciu='Villavicencio';
```

Oh con JOIN seria, pero ya no es una subconsulta :

```
SELECT e.nomb_est FROM estudiantes e JOIN ciudades c ON
e.cod_ciu=c.cod_ciu WHERE nomb_ciu='Villavicencio';
```

nomb_est
Carlos Alberto
Oscar Andres
Adelaida

ESTUDIANTES

id	nomb_est	apel_est	edad	cod_ciu
5	Adelaida	Jimenez	48	1
1	Carlos Alberto	Carranza	23	1
2	Oscar Andres	torres	27	1
3	Andres Felipe	Borda	37	2
4	Lucero	Cortez	55	3

CIUDADES

cod_ciu	nomb_ciud	poblacion
1	Villavicencio	3000
2	Restrepo	1500
3	Cumaral	1600
4	Bogota	12000
5	Cali	8000

SUBCONSULTA CON NOT EXISTS

Nombre de los estudiantes que no viven en la ciudad de Villavicencio.

```
SELECT e.nomb_est FROM estudiantes e
WHERE NOT EXISTS (SELECT * FROM ciudades c WHERE e.cod_ciu=c.cod_ciu
AND nomb_ciu='Villavicencio');
```

Otra forma seria:

```
SELECT e.nomb_est FROM estudiantes e JOIN ciudades c ON e.cod_ciu=c.cod_ciu
WHERE nomb_ciu NOT LIKE 'Villavicencio';
```

Con operador <>:

```
SELECT e.nomb_est FROM estudiantes e JOIN ciudades c ON e.cod_ciu=c.cod_ciu
WHERE nomb_ciu <> 'Villavicencio';
```

nomb_est Andres Felipe Lucero
--

ESTUDIANTES

id	nomb_est	apel_est	edad	cod_ciu
5	Adelaida	Jimenez	48	1
1	Carlos Alberto	Carranza	23	1
2	Oscar Andres	torres	27	1
3	Andres Felipe	Borda	37	2
4	Lucero	Cortez	55	3

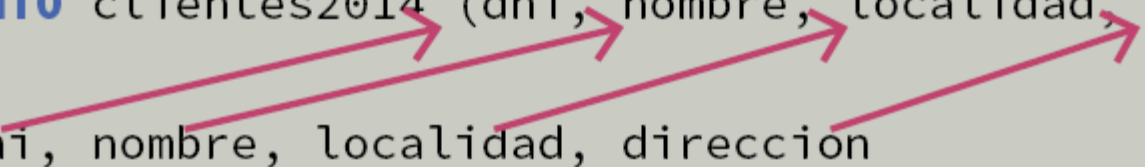
CIUDADES

cod_ciu	nomb_ciud	poblacion
1	Villavicencio	3000
2	Restrepo	1500
3	Cumara	1600
4	Bogota	12000
5	Cali	8000

SUBCONSULTAS EN INSTRUCCIONES DDL Y DML

Con INSERT: Relleno de registros a partir de filas de una consulta

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



Con UPDATE: Aumenta un 10% el sueldo de los empleados de la sección llamada Producción.

```
UPDATE empleados
SET sueldo=sueldo*1.10
WHERE id_seccion =(SELECT id_seccion FROM secciones
                    WHERE nom_seccion='Producción');
```

Con DELETE: Elimina los empleados cuyo identificador esté dentro de la tabla errores graves.

```
DELETE empleados  
WHERE id_empleado IN  
(SELECT id_empleado FROM errores_graves);
```