



ACADEMY

Programación de Bases de Datos con SQL

10-3

Subconsultas de Varias Filas



ORACLE ACADEMY

Copyright © 2017, Oracle y/o sus filiales. Todos los derechos reservados.

Objetivos

En esta lección se abordan los siguientes objetivos:

- Utilizar correctamente los operadores de comparación IN, ANY y ALL en subconsultas de varias filas
- Crear y ejecutar una subconsulta de varias filas en la cláusula WHERE o en la cláusula HAVING
- Describir qué sucede si una subconsulta de varias filas devuelve un valor nulo
- Comprender cuándo se deben utilizar subconsultas de varias filas y cuándo es seguro utilizar una subconsulta de una sola fila

Objetivos

En esta lección se abordan los siguientes objetivos:

- Distinguir entre subconsultas pareadas y no pareadas
- Crear una consulta con los operadores EXISTS y NOT EXISTS para probar filas devueltas de la subconsulta

Objetivo

- Una subconsulta se diseña para buscar información que no conoce y así poder encontrar la información que desea conocer.
- Sin embargo, las subconsultas de una sola fila solo pueden devolver una fila. ¿Y si necesita buscar información basada en varias filas y varios valores?
- La subconsulta tendrá que devolver varias filas.
- Para ello, utilizaremos subconsultas de varias filas y los tres operadores de comparación: IN, ANY y ALL.

Comparación de Consultas

- ¿De quién es el salario que es igual que el salario de un empleado del departamento 20?

LAST_NAME	DEPT_ID	SALARY
Hartstein	20	13000
Fay	20	6000

- En este ejemplo se devuelve un error porque existe más de un empleado en el departamento 20, la subconsulta devuelve varias filas.
- Esto se denomina subconsulta de varias filas.

```
SELECT first_name, last_name
FROM employees
WHERE salary =
  (SELECT salary
   FROM employees
   WHERE department_id = 20);
```



ORA-01427: single-row subquery returns more than one row

Comparación de Consultas

- El problema es el signo igual (=) en la cláusula WHERE de la consulta externa.
- ¿Cómo puede un valor ser igual (o no ser igual) a más de un valor a la vez?
- Es una pregunta tonta, ¿no?

```
SELECT first_name, last_name
FROM employees
WHERE salary =
  (SELECT salary
   FROM employees
   WHERE department_id = 20);
```



ORA-01427: single-row subquery returns more than one row

Por el mismo motivo, no se puede utilizar <, > o <> en la condición de la cláusula WHERE.

No tiene sentido comparar un valor con varios valores. ¿Es 10000 menor que (13000,6000)?

IN, ANY y ALL

- Las subconsultas que devuelven más de un valor se denominan subconsultas de varias filas.
- Puesto que no podemos utilizar operadores de comparación de una sola fila (=, <, etc.), necesitamos operadores de comparación diferentes para subconsultas de varias filas.
- Los operadores de varias filas son:
 - IN
 - ANY
 - ALL
- El operador NOT se puede utilizar con cualquiera de estos tres operadores.

IN

- El operador IN se utiliza en la cláusula WHERE de la consulta externa para seleccionar solo las filas que están EN la lista de valores devueltos de la consulta interna.
- Por ejemplo, estamos interesados en todos los empleados contratados el mismo año que un empleado del departamento 90.

```
SELECT last_name, hire_date
FROM employees
WHERE EXTRACT(YEAR FROM hire_date) IN
      (SELECT EXTRACT(YEAR FROM hire_date)
       FROM employees
       WHERE department_id=90);
```

LAST_NAME	HIRE_DATE
King	17-Jun-1987
Kochhar	21-Sep-1989
De Haan	13-Jan-1993
Whalen	17-Sep-1987

La función EXTRACT se puede utilizar para extraer los campos YEAR, MONTH o DAY de un tipo de dato DATE.

IN

- La consulta interna devolverá una lista de los años en los que se contrataron empleados del departamento 90.
- A continuación, la consulta externa devolverá cualquier empleado contratado el mismo año que cualquier año de la lista de la consulta interna.


```
SELECT last_name, hire_date
FROM employees
WHERE EXTRACT(YEAR FROM hire_date) IN
      (SELECT EXTRACT(YEAR FROM hire_date)
       FROM employees
       WHERE department_id=90);
```

LAST_NAME	HIRE_DATE
King	17-Jun-1987
Kochhar	21-Sep-1989
De Haan	13-Jan-1993
Whalen	17-Sep-1987

ANY

- El operador ANY se utiliza cuando deseamos que la cláusula WHERE de la consulta externa seleccione las filas que coinciden con los criterios (<, >, =, etc.) de **al menos** un valor en el juego de resultados de la subconsulta.
- En el ejemplo mostrado se devolverá cualquier empleado cuyo año de contratación sea menor que al menos un año de contratación que los empleados del departamento 90.

```
SELECT last_name, hire_date
FROM employees
WHERE EXTRACT(YEAR FROM hire_date) < ANY
      (SELECT EXTRACT(YEAR FROM hire_date)
       FROM employees
       WHERE department_id=90);
```



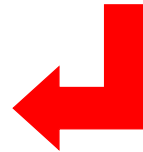
Año de Contratación
1987
1989
1993

LAST_NAME	HIRE_DATE
King	17-Jun-1987
Kochhar	21-Sep-1989
Whalen	17-Sep-1987
Hunold	03-Jan-1990
Ernst	21-May-1991

ALL

- El operador ALL se utiliza cuando deseamos que la cláusula WHERE de la consulta externa seleccione las filas que coinciden con los criterios (<, >, =, etc.) de **todos** los valores en el juego de resultados de la subconsulta.
- El operador ALL compara un valor con todos los valores devueltos por la consulta interna.
- Puesto que no se contrató a ningún empleado antes de 1987, no se devuelve ninguna fila.

```
SELECT last_name, hire_date
FROM employees
WHERE EXTRACT(YEAR FROM hire_date) < ALL
      (SELECT EXTRACT(YEAR FROM hire_date)
       FROM employees
       WHERE department_id=90);
```



Año de Contratación
1987
1989
1993

no se han encontrado datos

=ALL: ¿cómo puede un valor equivaler a todos los de un juego de valores? Por este motivo, =ALL apenas se utiliza.

Valores NULL

- Suponga que uno de los valores devueltos por una subconsulta de varias filas es nulo, pero otros valores no lo son.
- Si se utiliza IN o ANY, la consulta externa devolverá filas que coinciden con los valores no nulos.

```
SELECT last_name, employee_id
FROM employees
WHERE employee_id IN
  (SELECT manager_id
   FROM employees);
```

MANAGER_ID
-
100
100
101
101
205
100
...

LAST_NAME	EMPLOYEE_ID
King	100
Kochhar	101
De Haan	102
Higgins	205
...	



Resultado de la
subconsulta

Valores NULL

- Si se utiliza ALL, la consulta externa no devuelve ninguna fila porque ALL compara la consulta externa con cada valor devuelto por la subconsulta, incluido el valor nulo.
- Y la comparación de cualquier cosa con un valor nulo da como resultado un valor nulo.

```
SELECT last_name, employee_id
FROM employees
WHERE employee_id <= ALL
  (SELECT manager_id
   FROM employees);
```

no se han encontrado datos

GROUP BY y HAVING

- Como puede sospechar, la cláusula GROUP BY y la cláusula HAVING también se puede utilizar con subconsultas de varias filas.
- ¿Y si deseara buscar los departamentos cuyo salario mínimo sea inferior al salario de cualquier empleado que trabaje en el departamento 10 o 20?

LAST_NAME	DEPT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000

DEPARTMENT_ID	MIN(SALARY)
10	4400
20	6000
50	2500
60	4200
80	8600
110	8300
(nulo)	7000

GROUP BY y HAVING

- Necesitamos una subconsulta de varias filas que devuelva los salarios de empleados de los departamentos 10 y 20.
- La consulta externa utilizará una función de grupo (MIN), por lo que necesitamos AGRUPAR (GROUP) la consulta externa POR (BY) department_id.

LAST_NAME	DEPT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000

DEPARTMENT_ID	MIN(SALARY)
10	4400
20	6000
50	2500
60	4200
80	8600
110	8300
(nulo)	7000

GROUP BY y HAVING

- Esta es la sentencia SQL:

```
SELECT department_id, MIN(salary)
FROM employees
GROUP BY department_id
HAVING MIN(salary) < ANY
      (SELECT salary
       FROM employees
       WHERE department_id IN (10,20))
ORDER BY department_id;
```

LAST_NAME	DEPT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000

Resultado de la subconsulta

DEPARTMENT_ID	MIN(SALARY)
10	4400
20	6000
50	2500
60	4200
80	8600
110	8300
-	7000

Subconsultas de Varias Columnas

- Las subconsultas pueden utilizar una o más columnas.
- Si utilizan más de una columna, se denominan subconsultas de varias columnas.
- Una subconsulta de varias columnas pueden ser comparaciones pareadas o no pareadas.

```
SELECT employee_id, manager_id, department_id
FROM employees
WHERE (manager_id, department_id) IN
      (SELECT manager_id, department_id
       FROM employees
       WHERE employee_id IN (149, 174))
AND employee_id NOT IN (149, 174)
```

EMPLOYEE_ID	MANAGER_ID	DEPARTMENT_ID
176	149	80

Subconsultas de Varias Columnas

- En el siguiente ejemplo se muestra una subconsulta pareada de varias columnas con la subconsulta resaltada en rojo y el resultado en la tabla que aparece a continuación.
- La consulta muestra los empleados cuyo jefe y departamentos son los mismos que el jefe y el departamento de los empleados 149 o 174.

```
SELECT employee_id, manager_id, department_id
FROM employees
WHERE (manager_id, department_id) IN
  (SELECT manager_id, department_id
   FROM employees
   WHERE employee_id IN (149, 174))
AND employee_id NOT IN (149, 174)
```

EMPLOYEE_ID	MANAGER_ID	DEPARTMENT_ID
176	149	80

En primer lugar, se ejecuta la subconsulta para recuperar los valores de MANAGER_ID y DEPARTMENT_ID para los empleados con EMPLOYEE_ID 149 o 174.

Estos valores se comparan con las columnas MANAGER_ID y DEPARTMENT_ID de cada fila en la tabla EMPLOYEES. Si coinciden los valores, se muestra la fila.

En la salida, los registros de los empleados con EMPLOYEE_ID 149 o 174 no se mostrarán.

Subconsultas de Varias Columnas

- La consulta de varias columnas pareada también utiliza más de una columna en la subconsulta, pero las compara de una en una, por lo que las comparaciones se producen en distintas subconsultas.

```
SELECT employee_id,  
       manager_id,  
       department_id  
FROM employees  
WHERE manager_id IN  
      (SELECT manager_id  
       FROM employees  
       WHERE employee_id IN  
             (149,174))  
AND department_id IN  
      (SELECT department_id  
       FROM employees  
       WHERE employee_id IN  
             (149,174))  
AND employee_id NOT IN(149,174);
```

EMPLOYEE_ID	MANAGER_ID	DEPARTMENT_ID
176	149	80

Subconsultas de Varias Columnas

- Tendrá que escribir una subconsulta por columna con la que desee comparar al realizar subconsultas de varias columnas no pareadas.
- En el ejemplo de la derecha se muestra una subconsulta no pareada de varias columnas con las subconsultas resaltadas en rojo.

```
SELECT employee_id,  
       manager_id,  
       department_id  
FROM employees  
WHERE manager_id IN  
      (SELECT manager_id  
       FROM employees  
       WHERE employee_id IN  
            (149,174))  
AND department_id IN  
      (SELECT department_id  
       FROM employees  
       WHERE employee_id IN  
            (149,174))  
AND employee_id NOT IN(149,174);
```

EMPLOYEE_ID	MANAGER_ID	DEPARTMENT_ID
176	149	80

Subconsultas de Varias Columnas

- Esta consulta muestra los empleados que tienen el mismo manager_id y department_id que los empleados 149 o 174.

Resultado de la 1ª subconsulta

MANAGER_ID
100
149

Resultado de la 2ª subconsulta

DEPARTMENT_ID
80
80

```
SELECT employee_id,  
       manager_id,  
       department_id  
FROM employees  
WHERE manager_id IN  
      (SELECT manager_id  
       FROM employees  
       WHERE employee_id IN  
         (149,174))  
AND department_id IN  
      (SELECT department_id  
       FROM employees  
       WHERE employee_id IN  
         (149,174))  
AND employee_id NOT IN(149,174);
```

EMPLOYEE_ID	MANAGER_ID	DEPARTMENT_ID
176	149	80

En primer lugar, se ejecuta la subconsulta para recuperar los valores de MANAGER_ID para los empleados con EMPLOYEE_ID 149 o 174. Igualmente, se ejecuta la segunda subconsulta para recuperar los valores de DEPARTMENT_ID para los empleados con EMPLOYEE_ID 149 o 174.

Estos valores recuperados se comparan con las columnas MANAGER_ID y DEPARTMENT_ID de cada fila de la tabla EMPLEADOS.

EXISTS Y NOT EXISTS en Subconsultas

- EXISTS y su opuesto NOT EXISTS son dos cláusulas que se pueden utilizar al comprobar las coincidencias de subconsultas.
- EXISTS comprueba un resultado TRUE o coincidente en la subconsulta.
- Para responder a la pregunta: "¿qué empleados no son jefes?"
 - Primero tiene que preguntar "¿quiénes son los jefes?"
 - Y, a continuación, preguntar "¿quién NO EXISTE en la lista de jefes?"

EXISTS Y NOT EXISTS en Subconsultas

- En este ejemplo, la subconsulta selecciona los empleados que son jefes.
- A continuación, la consulta externa devuelve las filas de la tabla de empleados que NO EXISTEN en la subconsulta.

```
SELECT last_name AS "Not a Manager"
FROM employees emp
WHERE NOT EXISTS
  (SELECT *
   FROM employees mgr
   WHERE mgr.manager_id = emp.employee_id);
```

Not a Manager
Whalen
Gietz
Abel
Taylor
Grant
Rajs
Davies
Matos
Vargas
Ernst
...

EXISTS Y NOT EXISTS en Subconsultas

- Si se ejecuta la misma consulta con NOT IN en lugar de NOT EXISTS, el resultado es muy diferente.
- El resultado de esta consulta sugiere que no hay ningún empleado que también sea jefe, por lo que todos los empleados son jefes, lo que ya sabemos que no es cierto.

```
SELECT last_name AS "Not a Manager"  
FROM employees emp  
WHERE emp.employee_id NOT IN  
  (SELECT mgr.manager_id  
   FROM employees mgr);
```

no se han encontrado datos

EXISTS Y NOT EXISTS en Subconsultas

- La causa del extraño resultado es el valor NULL devuelto por la subconsulta.
- Una de las filas de la tabla employees no tiene un jefe y esto hace que todo el resultado sea incorrecto.
- Las subconsultas pueden devolver tres valores: TRUE, FALSE y UNKNOWN.
- Un valor NULL en el juego de resultados de la subconsulta devolverá un valor UNKNOWN, que Oracle no puede evaluar, por lo que no lo hace.

```
SELECT last_name AS "Not a Manager"  
FROM employees emp  
WHERE emp.employee_id NOT IN  
      (SELECT mgr.manager_id  
       FROM employees mgr);
```

no se han encontrado datos

EXISTS Y NOT EXISTS en Subconsultas

- TENGA CUIDADO con los valores NULL en las subconsultas al utilizar IN o NOT IN.
- Si no está seguro de si una subconsulta incluirá un valor nulo, elimine el valor nulo mediante IS NOT NULL en una cláusula WHERE.
- Por ejemplo : WHERE emp.manager_id IS NOT NULL o utilice NOT EXISTS para estar seguro.

NOT se puede utilizar con IN, ANY y ALL.

Un Último Punto sobre las Subconsultas

- Algunas subconsultas pueden devolver una o varias filas, según los valores de datos de las filas.
- Incluso aunque exista la mínima posibilidad de devolver varias filas, asegúrese de escribir una subconsulta de varias filas.

```
SELECT first_name, last_name, job_id
FROM employees
WHERE job_id =
  (SELECT job_id
   FROM employees
   WHERE last_name = 'Ernst');
```

FIRST_NAME	LAST_NAME	JOB_ID
Alexander	Hunold	IT_PROG
Bruce	Ernst	IT_PROG
Diana	Lorentz	IT_PROG

FIRST_NAME	LAST_NAME	JOB_ID
Bruce	Ernst	IT_PROG

Resultado de la subconsulta

Un Último Punto sobre las Subconsultas

- Por ejemplo: ¿quién tiene el mismo job_id que Ernst?
- Esta subconsulta de una sola fila funciona correctamente porque solo hay un Ernst en la tabla.
- ¿Pero qué sucedería si más adelante la empresa contratara a una nueva empleada llamada Susan Ernst?

```
SELECT first_name, last_name, job_id
FROM employees
WHERE job_id =
  (SELECT job_id
   FROM employees
   WHERE last_name = 'Ernst');
```

FIRST_NAME	LAST_NAME	JOB_ID
Alexander	Hunold	IT_PROG
Bruce	Ernst	IT_PROG
Diana	Lorentz	IT_PROG

FIRST_NAME	LAST_NAME	JOB_ID
Bruce	Ernst	IT_PROG

Resultado de la subconsulta

Un Último Punto sobre las Subconsultas

- Sería mejor escribir una subconsulta de varias filas.
- La sintaxis de la subconsulta de varias filas seguirá funcionando incluso aunque la subconsulta devuelva una sola fila.
- Si tiene dudas, escriba una subconsulta de varias filas.

```
SELECT first_name, last_name, job_id
FROM employees
WHERE job_id IN
  (SELECT job_id
   FROM employees
   WHERE last_name = 'Ernst');
```

FIRST_NAME	LAST_NAME	JOB_ID
Alexander	Hunold	IT_PROG
Bruce	Ernst	IT_PROG
Diana	Lorentz	IT_PROG

FIRST_NAME	LAST_NAME	JOB_ID
Bruce	Ernst	IT_PROG

Resultado de la subconsulta
Hay 2 personas con el apellido "Ernst"

Cuando utilizamos IN, ANY u ALL para comparar con una lista de valores, seguirá funcionando incluso aunque solo haya un valor en la lista.

Terminología

Entre los términos clave utilizados en esta lección se incluyen:

- EXIST y NOT EXIST
- Subconsulta no pareada de varias columnas
- Subconsulta pareada de varias columnas

Resumen

En esta lección, debe haber aprendido lo siguiente:

- Utilizar correctamente los operadores de comparación IN, ANY y ALL en subconsultas de varias filas
- Crear y ejecutar una subconsulta de varias filas en la cláusula WHERE o en la cláusula HAVING
- Describir qué sucede si una subconsulta de varias filas devuelve un valor nulo
- Comprender cuándo se deben utilizar subconsultas de varias filas y cuándo es seguro utilizar una subconsulta de una sola fila

Resumen

En esta lección, debe haber aprendido lo siguiente:

- Distinguir entre subconsultas pareadas y no pareadas
- Crear una consulta con los operadores EXISTS y NOT EXISTS para probar filas devueltas de la subconsulta



 **ACADEMY**