

Índice

1. Introducción	2
2. Consultas que retornan una única fila	3
3. Consultas que retornan múltiples filas. IN	5
4. Operadores AND y OR	9
5. Múltiples columnas en consultas anidadas	12
6. Operadores ANY y ALL	14
7. Ejercicios propuestos	17

1. INTRODUCCIÓN

Una consulta anidada, como su nombre indica, es aquella que está contenida dentro de otra. Los resultados de una la consulta anidada se utilizan como valores de comparación de la cláusula **WHERE** de la consulta que la anida. Se evaluará antes la sentencia **SELECT** anidada y una vez obtenido el valor o conjunto de valores se evaluará la otra.

El formato es el siguiente, a esto también se le llama “**subselect**”:

```
SELECT col1, col1, ... , coln  
FROM tabla1  
WHERE col <operador lógico>  
(SELECT col FROM tabla2 WHERE condiciones);
```

Se deben tener en cuenta los siguientes aspectos para ejecutar este tipo de sentencias:

- La tabla de la sentencia **SELECT** anidada **no** tiene porque ser la misma que la de la sentencia **SELECT** que la anida.
- El tipo de datos de la 'col' de la sentencia **SELECT** superior y de la 'col' de la anidada debe **ser el mismo**.
- Siempre se seleccionarán el mismo número de columnas de la sentencia **SELECT** anidada que el de columnas que forman parte de la cláusula **WHERE** en la superior.
- Las filas devueltas por la consulta anidada deben corresponderse con el tipo de operador lógico indicado. Si el operador lógico es '=' la sentencia **SELECT** anidada solo podrá devolver una única fila.

2. CONSULTAS ANIDADAS QUE RETORNAN UNA ÚNICA FILA

Las consultas anidadas que retornan una única fila se unen habitualmente por medio del operador lógico '='.

```
SELECT col1, col1, ... , coln
FROM tabla1
WHERE col = (SELECT col FROM tabla1 WHERE condiciones);
```

Si la consulta anidada retorna más de un valor se produce un error en la ejecución de la superior por no poder comparar una sola columna con más de un valor.

Seleccionar aquellos empleados que trabajen en el mismo departamento que 'CLARK'.

```
SELECT last_name, salary, department_id
FROM employee
WHERE department_id = ( SELECT department_id
                        FROM employee
                        WHERE last_name = 'CLARK');
```

	 LAST_NAME	 SALARY	 DEPARTMENT_ID
1	CLARK	2450	10
2	KING	5000	10
3	MILLER	1300	10

Seleccionar aquellos empleados cuyo salario sea superior al que obtiene 'ADAMS'.

```
SELECT last_name, salary
FROM employee
WHERE salary > ( SELECT salary
                  FROM employee
                  WHERE last_name = 'ADAMS');
```

	LAST_NAME	SALARY
1	ALLEN	1600
2	DOYLE	2850
3	DENNIS	2750
4	BAKER	2200
5	WARD	1250
6	PETERS	1250
7	SHAW	1250
8	DUNCAN	1250
9	LANGE	1250
10	JONES	2975
11	ALBERTS	3000
12	PORTER	1250
13	LEWIS	1800
14	MARTIN	1250
15	SOMMERS	1850
16	BLAKE	2850
17	CLARK	2450
18	SCOTT	3000
19	WEST	1500
20	FISHER	3000
21	ROSS	1300
22	KING	5000
23	TURNER	1500
24	FORD	3000
25	ROBERTS	2875
26	MILLER	1300

Seleccionar el nombre y la fecha de entrada del empleado que lleve más tiempo trabajando en la empresa.

```
SELECT last_name, hire_date
FROM employee
WHERE hire_date = ( SELECT MIN (hire_date)
                    FROM employee);
```

	LAST_NAME	HIRE_DATE
1	SMITH	17/12/84



3. CONSULTAS ANIDADAS QUE RETORNAN MULTIPLES FILAS

El operador lógico que deberemos utilizar para crear éstas consultas es 'IN', ya que permite comparar un valor con una lista de valores.

```
SELECT col1, col1, ... , coln  
FROM tabla1  
WHERE col IN (SELECT col FROM tabla1 WHERE condiciones);
```

Seleccionar aquellos empleados cuyo salario sea igual al que obtiene 'ADAMS' o 'SCOTT'.

```
SELECT last_name, salary  
FROM employee WHERE salary IN ( SELECT salary  
                                FROM employee  
                                WHERE last_name IN ('ADAMS', 'SCOTT'));
```

	 LAST_NAME	 SALARY
1	FORD	3000
2	FISHER	3000
3	SCOTT	3000
4	ALBERTS	3000
5	ADAMS	1100



Seleccionar el nombre y la función de aquellos empleados que tengan la misma función que alguno de los empleados que trabajan en el departamento 20.

```
SELECT last_name, job_id
FROM employee
WHERE job_id IN ( SELECT job_id
                  FROM employee WHERE department_id = 20);
```

	LAST_NAME	JOB_ID
1	MURRAY	667
2	JENSEN	667
3	MILLER	667
4	DOUGLAS	667
5	JAMES	667
6	ADAMS	667
7	SMITH	667
8	CLARK	671
9	BLAKE	671
10	ALBERTS	671
11	JONES	671
12	BAKER	671
13	DENNIS	671
14	DOYLE	671
15	ROBERTS	669
16	FORD	669
17	FISHER	669
18	SCOTT	669

Seleccionar el nombre y el departamento de aquellos empleados que trabajan en CHICAGO

```
SELECT last_name, department_id
FROM employee
WHERE department_id IN ( SELECT department_id
                        FROM department
                        WHERE location_id = ( SELECT location_id
                                          FROM location
                                          WHERE regional_group='CHICAGO'));
```

	 LAST_NAME	 DEPARTMENT_ID
1	JAMES	30
2	TURNER	30
3	BLAKE	30
4	MARTIN	30
5	WARD	30
6	ALLEN	30
7	SOMMERS	34

Seleccionar el nombre y el departamento de aquellos empleados que trabajan en CHICAGO o DALLAS

```
SELECT last_name, department_id
FROM employee
WHERE department_id IN ( SELECT department_id
                        FROM department
                        WHERE location_id = ( SELECT location_id
                                          FROM location
                                          WHERE (regional_group='CHICAGO' or regional_group= 'DALLAS')));
```

	LAST_NAME	DEPARTMENT_ID
1	SOMMERS	34
2	JAMES	30
3	TURNER	30
4	BLAKE	30
5	MARTIN	30
6	WARD	30
7	ALLEN	30
8	LEWIS	24
9	MURRAY	23
10	WEST	23
11	LANGE	23
12	DUNCAN	23
13	DENNIS	23
14	FORD	20
15	ADAMS	20
16	SCOTT	20
17	JONES	20
18	SMITH	20

4. OPERADORES AND y OR EN CONSULTAS ANIDADAS

Seleccionar aquellos empleados cuyo salario sea mayor al que obtiene 'ADAMS' o trabajen en el departamento de 'SCOTT'.

```
SELECT last_name, salary, department_id
FROM employee
WHERE salary > ( SELECT salary
                  FROM employee
                  WHERE last_name = 'ADAMS')
OR
department_id = ( SELECT department_id
                  FROM employee
                  WHERE last_name = 'SCOTT');
```

	LAST_NAME	SALARY	DEPARTMENT_ID
1	SMITH	800	20
2	ALLEN	1600	30
3	DOYLE	2850	13
4	DENNIS	2750	23
5	BAKER	2200	14
6	WARD	1250	30
7	PETERS	1250	13
8	SHAW	1250	13
9	DUNCAN	1250	23
10	LANGE	1250	23
11	JONES	2975	20
12	ALBERTS	3000	12
13	PORTER	1250	13
14	LEWIS	1800	24
15	MARTIN	1250	30
16	SOMMERS	1850	34
17	BLAKE	2850	30
18	CLARK	2450	10
19	SCOTT	3000	20
20	WEST	1500	23
21	FISHER	3000	12
22	ROSS	1300	43
23	KING	5000	10
24	TURNER	1500	30
25	ADAMS	1100	20
26	FORD	3000	20
27	ROBERTS	2875	12
28	MILLER	1300	10



Seleccionar el nombre y la función de aquellos empleados que tengan la misma función que alguno de los empleados que trabajan en el departamento 20 o aquellos que entrasen los últimos en la empresa.

```
SELECT last_name, job_id
FROM employee
WHERE job_id IN (SELECT job_id
                 FROM employee
                 WHERE department_id = 20)
OR
hire_date = ( SELECT MAX (hire_date)
              FROM employee);
```

	LAST_NAME	JOB_ID
1	SMITH	667
2	DOYLE	671
3	DENNIS	671
4	BAKER	671
5	JONES	671
6	ALBERTS	671
7	BLAKE	671
8	CLARK	671
9	SCOTT	669
10	FISHER	669
11	ADAMS	667
12	JAMES	667
13	FORD	669
14	ROBERTS	669
15	DOUGLAS	667
16	MILLER	667
17	JENSEN	667
18	MURRAY	667

Seleccionar el nombre y el departamento de aquellos empleados que trabajan en la localidad de código 122 y ganan más que MILLER.

```
SELECT last_name, department_id
FROM employee
WHERE department_id IN ( SELECT department_id
                        FROM department
                        WHERE location_id =122)
AND
Salary > ( SELECT salary
          FROM employee
          WHERE last_name = 'MILLER');
```

	 LAST_NAME	 DEPARTMENT_ID
1	DOYLE	13
2	BAKER	14
3	ALBERTS	12
4	CLARK	10
5	FISHER	12
6	KING	10
7	ROBERTS	12

5. MULTIPLES COLUMNAS EN CONSULTAS ANIDADAS

Dentro de la condición que debe cumplir la consulta anidada, puede haber más de una columna.




```
SELECT col1, col1, ... , coln
FROM tabla1
WHERE (col1, col2, ... ,coln) IN
      (SELECT col1, col2, ... , coln)
      FROM tabla1
      WHERE condiciones).
```

Se tiene que cumplir.

- El número de columnas que forman parte de la condición y el de columnas seleccionadas por la consulta anidada debe ser el mismo.
- Para que se cumpla esta condición tienen que ser igual todos los valores de las columnas devueltas por la consulta anidada a los de las columnas que forman parte de la condición en la superior. No basta con que sea igual el valor de una de las columnas.

Seleccionar el nombre, el departamento y la fecha de entrada de aquellos empleados que llevan trabajando más tiempo en cada departamento.

```
SELECT last_name, department_id, hire_date
FROM employee
WHERE (hire_date, department_id) IN ( SELECT MIN(hire_date), department_id
                                     FROM employee
                                     GROUP BY department_id);
```

	 LAST_NAME	 DEPARTMENT_ID	 HIRE_DATE
1	ALLEN	30	20/02/85
2	SOMMERS	34	19/04/85
3	ROSS	43	01/06/85
4	PETERS	13	31/03/85
5	SMITH	20	17/12/84
6	BAKER	14	10/06/85
7	LEWIS	24	16/04/85
8	WEST	23	04/04/85
9	ALBERTS	12	06/04/85
10	CLARK	10	09/06/85

El anterior ejemplo no hubiese sido válido con la sentencia:

```
SELECT last_name, department_id, hire_date
FROM employee
WHERE (hire_date) IN ( SELECT MIN(hire_date), department_id
                      FROM employee
                      GROUP BY department_id);
```

Ya que debe ser igual, además de la fecha de entrada, el número de departamento. Puede coincidir que un empleado tenga la misma fecha de entrada que el primero que entró en otro departamento, pero que en el suyo haya alguien que entró antes. En este caso la segunda sentencia lo seleccionaría, cuando en realidad no es lo que queremos.



6. OPERADORES ANY y ALL

Se pueden realizar otro tipo de comparaciones de las empleadas hasta el momento cuando la sentencia SELECT anidada devuelve más de un valor, con dos nuevos operadores: **ANY** y **ALL**. Estos actúan sobre todo el conjunto de filas devueltas.

> ANY	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea mayor que algún valor devuelto por la sentencia SELECT anidada.
>= ANY	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea mayor o igual que algún valor devuelto por la sentencia SELECT anidada.
< ANY	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea menor que algún valor devuelto por la sentencia SELECT anidada.
<= ANY	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea menor o igual que algún valor devuelto por la sentencia SELECT anidada.
= ANY	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea igual que algún valor devuelto por la sentencia SELECT anidada. Es equivalente al operador IN.
> ALL	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea mayor que todos los valores devueltos por la sentencia SELECT anidada.
>= ALL	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea mayor o igual que todos los valores devueltos por la sentencia SELECT anidada.
< ALL	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea menor que todos los valores devueltos por la sentencia SELECT anidada.
<= ALL	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea menor o igual que todos los valores devueltos por la sentencia SELECT anidada.
!= ALL	La consulta seleccionará aquellas filas en las que el valor de la columna de la condición sea distinto que todo valor devuelto por la sentencia SELECT anidada. Es equivalente al operador NOT IN.

Seleccionar los empleados que ganan más que todos los que tienen función 668.

```
SELECT last_name, salary
FROM employee
WHERE salary > ALL (SELECT salary
                    FROM employee WHERE job_id = 668)
```

	 LAST_NAME	 SALARY
1	BAKER	2200
2	CLARK	2450
3	DENNIS	2750
4	BLAKE	2850
5	DOYLE	2850
6	ROBERTS	2875
7	JONES	2975
8	FORD	3000
9	FISHER	3000
10	SCOTT	3000
11	ALBERTS	3000
12	KING	5000

Seleccionar los empleados que ganan más que alguno cuya función sea 668.

```
SELECT last_name, salary
FROM employee
WHERE salary > ANY (SELECT salary
                     FROM employee
                     WHERE job_id = 668);
```

	LAST_NAME	SALARY
1	KING	5000
2	FORD	3000
3	FISHER	3000
4	ALBERTS	3000
5	SCOTT	3000
6	JONES	2975
7	ROBERTS	2875
8	BLAKE	2850
9	DOYLE	2850
10	DENNIS	2750
11	CLARK	2450
12	BAKER	2200
13	SOMMERS	1850

7. EJERCICIOS PROPUESTOS

- 1.- Visualiza el nombre del empleado que más cobra en la empresa, sin contar la comisión.
- 2.- Visualiza el nombre del empleado más antiguo de la empresa.
- 3.- Visualiza el nombre de los empleados cuyo salario se encuentra por debajo de la media salarial en la empresa.
- 4.- Visualiza los nombres de productos que nunca han sido vendidos.
- 5.- Visualiza el nombre de empleados que trabajan en el depto RESEARCH.
- 6.- Visualiza el nombre de los empleados cuyo responsable directo es Doyle (last_name).
- 7.- Visualiza cuánto dinero (salario + comisión) total cobran al mes el conjunto de los empleados de los departamentos de operaciones (OPERATIONS).
- 8.- Visualiza el límite de crédito mínimo que corresponde a los clientes que representa el vendedor cuyo apellido es TURNER
- 9.- Visualiza el nombre de los empleados cuya función es vendedor (salesperson)
- 10.- Visualiza el nombre del producto que protagonicen líneas en las que se haya solicitado más de 10 unidades de ellos.
- 11.- Visualiza los nombres de empleados del departamento 20 que cobran más que alguno de los empleados del departamento 13.
- 12.- Visualiza si algún empleado del 20 cobra más que todos los del 13.
- 13.- Visualiza el nombre de los empleados que trabajan en el departamento 20 y tienen un salario superior a la media salarial de su propio departamento.

- 14.- Visualiza el nombre y código de los departamentos con más de 4 empleados.
- 15.- Visualiza el nombre, apellido, ciudad y estado de los clientes cuyos pedidos medios superan los 500\$
- 16.- Visualiza el nombre, apellido, función y departamento de los empleados que tienen el mínimo salario por cada función.
- 17.- Visualiza el nombre, apellido y salario de los empleados que tienen el máximo salario por departamento.
- 18.- Visualiza el nombre de los empleados que trabajen en el mismo depto que Daniel y que cobren lo mismo que él.
- 19.- Visualiza el nombre, dirección, ciudad y estado de los clientes que tienen como crédito el mismo que el mínimo crédito asociado a los clientes que representa el vendedor cuyo apellido es TURNER
- 20.- Visualiza el nombre del producto del que se han vendido más unidades.
- 21.- Visualiza el nombre del cliente que menos ha gastado
- 22.- Visualiza el nombre del departamento con menor sueldo.
- 23.- Visualiza el nombre, apellido y fecha de contratación de los empleados contratados el mes de menor grado de contratación.
- 24.- Visualiza el nombre de los clientes (customer) que son atendidos por vendedores que trabajan en el departamento de ventas de Boston.
- 25.- Visualiza la ciudad donde se ubica el departamento donde se paga más a los empleados.
- 26.- Visualiza cuántos empleados tiene el departamento que mas empleados tiene realizando la función vendedor (SALESPERSON)