SUBCONSULTAS

Descripción de tablas : EMPLE y DEPART

```
SQL> DESC EMPLE
  Nombre                        ¿Nulo?   Tipo
  ----------------------------- -------- ---------
  EMP_NO                        NOT NULL NUMBER(4)
  APELLIDO                               VARCHAR2(10)
  OFICIO                                 VARCHAR2(10)
  DIR                                    NUMBER(4)
  FECHA_ALT                              DATE
  SALARIO                                NUMBER(7)
  COMISION                               NUMBER(7)
  DEPT_NO                       NOT NULL NUMBER(2)

SQL> DESC DEPART
  Nombre                        ¿Nulo?   Tipo
  ----------------------------- -------- ---------
  DEPT_NO                       NOT NULL NUMBER(2)
  DNOMBRE                                VARCHAR2(14)
  LOC                                    VARCHAR2(14)
```

A veces, para realizar alguna operación de consulta, necesitamos los datos devueltos por otra consulta; así, si queremos obtener los datos de los empleados que tengan el mismo oficio que 'PEPE', hemos de averiguar el oficio de 'PEPE' (primera consulta). Una vez conocido este dato, podemos averiguar qué empleados tienen el mismo oficio que 'PEPE' (segunda consulta). Este problema se puede resolver usando una subconsulta, que no es más que una sentencia SELECT dentro de otra SELECT. Las **subconsultas** son aquellas sentencias SELECT que forman parte de una cláusula WHERE de una sentencia SELECT anterior. Una subconsulta consistirá en incluir una declaración SELECT como parte de una cláusula WHERE. El formato de una subconsulta es similar a éste:

```
SELECT  ...
FROM    ...
WHERE  columna  operador_comparativo   (SELECT ...
                                       FROM ...
                                       WHERE ... );
```

La subconsulta (el comando SELECT entre paréntesis) se ejecutará primero y, posteriormente, el valor extraído es «introducido» en la consulta principal.

Caso práctico

11 Con la tabla EMPLE, obtén el APELLIDO de los empleados con el mismo OFICIO que 'GIL'. Para ello, descomponemos el enunciado en dos consultas. Primero averiguamos el OFICIO de 'GIL':

```
SQL> SELECT OFICIO FROM EMPLE WHERE APELLIDO ='GIL';

OFICIO
----------
ANALISTA
```

- A continuación, visualizamos el APELLIDO de los empleados con el mismo OFICIO que 'GIL':

```
SQL> SELECT APELLIDO FROM EMPLE WHERE OFICIO='ANALISTA';

APELLIDO
----------
GIL
FERNANDEZ
```

- Es posible resumir estas dos consultas con una sentencia SELECT que forma parte de la cláusula WHERE:

```
SQL> SELECT APELLIDO FROM EMPLE WHERE OFICIO = (SELECT OFICIO FROM EMPLE WHERE APE-
LLIDO ='GIL');

APELLIDO
----------
GIL
FERNANDEZ
```

Actividades propuestas

4 **Muestra los datos (apellido, oficio, salario y fecha de alta) de aquellos empleados que desempeñen el mismo oficio que "JIMENEZ" o que tengan un salario mayor o igual que "FERNANDEZ".**

A. Condiciones de búsquedas en subconsultas

Las **subconsultas** normalmente aparecen como parte de la condición de búsqueda de una cláusula WHERE o HAVING. Las condiciones de búsqueda que nos podemos encontrar en una subconsulta son:

- **Test de comparación en subconsultas (>, <, <>, <=, >=, =).** Compara el valor de una expresión con un valor único producido por una subconsulta. Ejemplo: Obtener aquellos apellidos de empleados cuyo oficio es igual al oficio de 'GIL':

```
SELECT APELLIDO FROM EMPLE WHERE OFICIO = (SELECT OFICIO
FROM EMPLE WHERE APELLIDO ='GIL');
```

- **Test de pertenencia a un conjunto devuelto por una subconsulta (IN).** Comprueba si el valor de una expresión coincide con uno del conjunto de valores producido por una subconsulta. Ejemplo: Obtener aquellos apellidos de empleados cuyo oficio sea alguno de los oficios que hay en el departamento 20:

```
SELECT APELLIDO FROM EMPLE WHERE OFICIO IN (SELECT OFICIO
FROM EMPLE WHERE DEPT_NO=20);
```

- **Test de existencia (EXISTS , NOT EXISTS).** Examina si una subconsulta produce alguna fila de resultados. El test es TRUE si devuelve filas, si no es FALSE. Ejemplo: Listar los departamentos que tengan empleados:

```
SELECT DNOMBRE, DEPT_NO FROM DEPART WHERE EXISTS (SELECT
* FROM EMPLE WHERE EMPLE.DEPT_NO= DEPART.DEPT_NO);
```

  Para calcular los que no tengan empleados se usa el operador NOT EXISTS.

- **Test de comparación cuantificada (ANY y ALL).** Se usan en conjunción con los operadores de comparación (>, <, <>, <=, >=, =).

  ANY compara el valor de una expresión con cada uno del conjunto de valores producido por una subconsulta, si alguna de las comparaciones individuales da como resultado TRUE, ANY devuelve TRUE, si la subconsulta no devuelve nada devolverá FALSE. Ejemplo: obtener los datos de los empleados cuyo salario sea igual a algún salario de los empleados del departamento 30:

```
SELECT * FROM EMPLE WHERE SALARIO = ANY  (SELECT SALARIO
FROM EMPLE WHERE DEPT_NO=30);
```

ALL compara el valor de una expresión con cada uno del conjunto de valores producido por una subconsulta, si todas las comparaciones individuales da como resultado **TRUE**, **ALL** devuelve **TRUE**, en caso contrario devuelve **FALSE**. Ejemplo: Obtener los datos de los empleados cuyo salario sea menor a cualquier salario de los empleados del departamento 30:

```
SELECT * FROM EMPLE WHERE SALARIO < ALL (SELECT SALARIO
FROM EMPLE WHERE DEPT_NO=30);
```

B. Subconsultas que generan valores simples

Se trata de aquellas subconsultas que devuelven una fila o un valor simple; en éstas se utilizan los operadores de comparación (>, <, <>, <=, >=, =). Si la subconsulta obtiene más de una fila, se produce un mensaje de error. Por ejemplo, con la siguiente consulta se pretende obtener los apellidos de los empleados cuyo oficio coincida con algún oficio del departamento 20:

```
SQL>  SELECT APELLIDO FROM EMPLE WHERE OFICIO =
  2   (SELECT OFICIO FROM EMPLE where DEPT_NO=20);
 (SELECT OFICIO FROM EMPLE where DEPT_NO=20)
  *
ERROR en línea 2:
ORA-01427: la subconsulta de una sola fila devuelve más
de una fila
```

En el departamento 20 hay varios oficios por tanto la subconsulta devuelve varias filas. Por ello, cuando una subconsulta devuelve más de una fila no se puede recurrir a los operadores de comparación.

C. Subconsultas que generan listas de valores

Son aquellas subconsultas que devuelven más de una fila o más de un valor. Cuando una subconsulta devuelva más de un valor, usaremos el operador **IN** en la cláusula **WHERE**. La solución al ejemplo anterior sería la siguiente:

```
SQL> SELECT APELLIDO FROM EMPLE WHERE OFICIO IN
(SELECT OFICIO FROM EMPLE where DEPT_NO=20);
```

El tipo de dato de la expresión situada después de **WHERE** debe coincidir con el tipo de dato devuelto por la subconsulta.

Caso práctico

12 **Usamos las tablas EMPLE y DEPART. Queremos consultar los datos de los empleados que trabajen en 'MADRID' o 'BARCELONA'. La localidad de los departamentos se obtiene de la tabla DEPART. Hemos de relacionar las tablas EMPLE y DEPART por el número de departamento. Para ello, descomponemos esa consulta en varias: primero tenemos que localizar los números de departamento que estén en la localidad de 'MADRID' o 'BARCELONA':**

```
SELECT DEPT_NO FROM DEPART WHERE LOC IN ('MADRID', 'BARCELONA');

   DEPT_NO
----------
        20
        30
```

A continuación, seleccionamos los datos de los empleados que estén en esos departamentos:

```
SELECT EMP_NO, APELLIDO, OFICIO, DIR, FECHA_ALT, SALARIO, COMISION, DEPT_NO FROM
EMPLE  WHERE  DEPT_NO  IN(20, 30);
```

Por último, reunimos estas dos sentencias SELECT utilizando una subconsulta:

```
SELECT EMP_NO, APELLIDO, OFICIO, DIR, FECHA_ALT, SALARIO, COMISION, DEPT_NO
FROM  EMPLE  WHERE  DEPT_NO  IN  (SELECT  DEPT_NO  FROM  DEPART  WHERE  LOC  IN
('MADRID','BARCELONA'));
```

- Consulta los apellidos y oficios de todos los empleados del departamento 20 cuyo trabajo sea idéntico al de cualquiera de los empleados del departamento 'VENTAS':

```
SQL> SELECT APELLIDO, OFICIO FROM EMPLE WHERE DEPT_NO=20 AND
  2  OFICIO IN (SELECT OFICIO FROM EMPLE WHERE DEPT_NO =
  3  (SELECT  DEPT_NO  FROM  DEPART  WHERE DNOMBRE='VENTAS'));

APELLIDO     OFICIO
----------   ----------
SANCHEZ      EMPLEADO
JIMENEZ      DIRECTOR
ALONSO       EMPLEADO
```

- Obtén el apellido de los empleados con el mismo oficio y salario que 'GIL'. En esta consulta se introduce una variante: hasta ahora las subconsultas nos devolvían una columna, aunque pueden devolver más de una. En este caso, la subconsulta devuelve dos columnas, el oficio y el salario:

```
SQL> SELECT APELLIDO, SALARIO FROM EMPLE WHERE (OFICIO, SALARIO) =
  2  (SELECT OFICIO, SALARIO FROM EMPLE WHERE APELLIDO ='GIL');

APELLIDO       SALARIO
----------   ----------
GIL              3000
FERNANDEZ        3000
```

(Continuación)

Si se detallan varios campos en la cláusula WHERE, deben encerrarse entre paréntesis, y han de coincidir en número y tipo de datos con los de la sentencia SELECT de la consulta interior. Cada columna de la cláusula WHERE se referirá a la correspondiente columna de la subconsulta. También podíamos haber puesto esto:

```
SQL>SELECT APELLIDO, SALARIO FROM EMPLE
  2 WHERE OFICIO= (SELECT OFICIO FROM EMPLE WHERE APELLIDO ='GIL')
  3 AND SALARIO= (SELECT SALARIO  FROM EMPLE WHERE APELLIDO ='GIL');
```

Actividades propuestas

5 **Presenta los apellidos y oficios de los empleados que tienen el mismo trabajo que "JIMENEZ".**

Muestra en pantalla el APELLIDO, OFICIO y SALARIO de los empleados del departamento de "FERNANDEZ" que tengan su mismo salario.

D. Subconsultas correlacionadas

Una **subconsulta correlacionada** es aquella que hace referencia a una columna o varias de la consulta más externa. A veces la subconsulta hace uso de columnas que tienen el mismo nombre que las columnas de las tablas usadas en la consulta mas externa. Si la subconsulta necesita acceder a esas columnas deberá definirse un alias en la tabla más externa. Por ejemplo, deseamos obtener los datos de los empleados cuyo salario sea el máximo salario de su departamento:

```
SELECT * FROM EMPLE E WHERE SALARIO = (SELECT
MAX(SALARIO) FROM EMPLE WHERE DEPT_NO = E.DEPT_NO);
```

La subconsulta devuelve para cada fila que se recupere de la consulta más externa el máximo salario del departamento que se está recuperando en la consulta externa; para referenciar a dicho departamento se necesita el alias **E** usado en la tabla de la consulta externa.

3.15 Combinación de tablas

Hasta ahora, en las consultas que hemos realizado sólo se ha utilizado una tabla, indicada a la derecha de la palabra FROM; pero hay veces que una consulta necesita columnas de varias tablas. En este caso, las tablas se expresarán a la derecha de la palabra FROM.

Sintaxis general:

```
SELECT    columnas de las tablas citadas en la cláusula
          'from'
FROM      tabla1, tabla2,...
WHERE     tabla1.columna = tabla2.columna;
```

Cuando combinamos varias tablas, hemos de tener en cuenta una serie de reglas:

- Es posible unir tantas tablas como deseemos.
- En la cláusula SELECT se pueden citar columnas de todas las tablas.
- Si hay columnas con el mismo nombre en las distintas tablas de la cláusula FROM, se deben identificar, especificando `NombreTabla.NombreColumna`.
- Si el nombre de una columna existe sólo en una tabla, no será necesario especificarla como `NombreTabla.NombreColumna`. Sin embargo, hacerlo mejoraría la legibilidad de la sentencia SELECT.
- El criterio que se siga para combinar las tablas ha de especificarse en la cláusula WHERE. Si se omite esta cláusula, que especifica la condición de combinación, el resultado será un PRODUCTO CARTESIANO, que emparejará todas las filas de una tabla con cada fila de la otra.

Caso práctico

13 A partir de las tablas EMPLE y DEPART obtenemos los siguientes datos de los empleados: APELLIDO, OFICIO, número de empleado (EMP_NO), nombre de departamento (DNOMBRE) y localidad (LOC). Estas tablas tienen en común el campo DEPT_NO, por el que se combinan las tablas:

```
SQL> SELECT APELLIDO, OFICIO, EMP_NO, DNOMBRE, LOC FROM  EMPLE, DEPART
  2   WHERE EMPLE.DEPT_NO = DEPART.DEPT_NO;
```

APELLIDO	OFICIO	EMP_NO	DNOMBRE	LOC
SANCHEZ	EMPLEADO	7369	INVESTIGACION	MADRID
ARROYO	VENDEDOR	7499	VENTAS	BARCELONA
SALA	VENDEDOR	7521	VENTAS	BARCELONA
JIMENEZ	DIRECTOR	7566	INVESTIGACION	MADRID
MARTIN	VENDEDOR	7654	VENTAS	BARCELONA
NEGRO	DIRECTOR	7698	VENTAS	BARCELONA
CEREZO	DIRECTOR	7782	CONTABILIDAD	SEVILLA
GIL	ANALISTA	7788	INVESTIGACION	MADRID
REY	PRESIDENTE	7839	CONTABILIDAD	SEVILLA
TOVAR	VENDEDOR	7844	VENTAS	BARCELONA
ALONSO	EMPLEADO	7876	INVESTIGACION	MADRID

(Continuación)

```
APELLIDO        OFICIO        EMP_NO      DNOMBRE          LOC
---------       ----------    ------      -------------    ----------
JIMENO          EMPLEADO        7900      VENTAS           BARCELONA
FERNANDEZ       ANALISTA        7902      INVESTIGACION    MADRID
MUÑOZ           EMPLEADO        7934      CONTABILIDAD     SEVILLA

14 filas seleccionadas.
```

Todos los empleados con un número de departamento 10 serán emparejados con los datos del departamento 10; los empleados con un número de departamento 20 se emparejarán con los datos del departamento 20 y, así, sucesivamente. Si se omite la cláusula WHERE, resultará un PRODUCTO CARTESIANO donde se emparejaría cada fila de una tabla con todas las filas de la otra tabla. En este caso: 14 filas de la tabla EMPLE por 4 filas de la tabla DEPART = 56 filas.

- Veamos un ejemplo en el que se combinan las siguientes 3 tablas.

  ALUMNOS: Contiene los datos de los alumnos. La descripción es ésta:

```
SQL> DESC ALUMNOS
     Nombre                          ¿Nulo?        Tipo
     ------------------------------  --------      ------------
     DNI                             NOT NULL      VARCHAR2(10)
     APENOM                                        VARCHAR2(30)
     DIREC                                         VARCHAR2(30)
     POBLA                                         VARCHAR2(15)
     TELEF                                         VARCHAR2(10)
```

  ASIGNATURAS: Contiene los nombres de asignaturas con sus códigos correspondientes. La descripción es la siguiente:

```
SQL> DESC ASIGNATURAS
     Nombre                          ¿Nulo?        Tipo
     ------------------------------  --------      --------
     COD                             NOT NULL      NUMBER(2)
     NOMBRE                                        VARCHAR2(25)
```

  NOTAS: Contiene las notas de cada alumno en cada asignatura. Se relaciona con la tabla ALUMNOS por la columna DNI y con la tabla ASIGNATURAS por la columna COD. La descripción es:

```
SQL> DESC NOTAS
     Nombre                          ¿Nulo?        Tipo
     ------------------------------  --------      --------
     DNI                             NOT NULL      VARCHAR2(10)
     COD                             NOT NULL      NUMBER(2)
     NOTA                                          NUMBER(2)
```

- Realiza una consulta para obtener el nombre de alumno, su asignatura y su nota:

```
SQL> SELECT APENOM, NOMBRE, NOTA FROM ALUMNOS, ASIGNATURAS, NOTAS
  2   WHERE ALUMNOS.DNI=NOTAS.DNI AND NOTAS.COD=ASIGNATURAS.COD;
```

(Continúa)

(Continuación)

```
APENOM                    NOMBRE                      NOTA
------------------------  ----------------------  ------
Alcalde García, Elena     Prog. Leng. Estr.            6
Alcalde García, Elena     Sist. Informáticos           5
Alcalde García, Elena     Análisis                     6
Díaz Fernández, María     FOL                          8
Cerrato Vela, Luis        FOL                          6
Díaz Fernández, María     RET                          7
Cerrato Vela, Luis        RET                          8
Díaz Fernández, María     Entornos Gráficos            8
Cerrato Vela, Luis        Entornos Gráficos            4
Díaz Fernández, María     Aplic. Entornos 4ªGen        9
Cerrato Vela, Luis        Aplic. Entornos 4ªGen        5

11 filas seleccionadas.
```

- Obtén los nombres de alumnos matriculados en 'FOL':

```
SQL> SELECT APENOM FROM ALUMNOS, ASIGNATURAS, NOTAS WHERE
  2  ALUMNOS.DNI=NOTAS.DNI AND NOTAS.COD=ASIGNATURAS.COD
  3  AND NOMBRE='FOL';

     APENOM
     ----------------------
     Cerrato Vela, Luis
     Díaz Fernández, María
```

Actividades propuestas

6 **Visualiza los nombres de alumnos que tengan una nota entre 7 y 8 e la asignatura de "FOL".**

Visualiza los nombres de asignaturas que no tengan suspensos.