CAPITULO 18. SENTENCIAS PARA MODIFICAR DATOS

En los ejercicios siguientes se supone que disponemos de una tabla llamada TEMPLE2 cuyas columnas tienen igual nombre y características que las de TEMPLE, y que inicialmente está vacía.
(Para resolver los ejercicios marcados con un asterisco es necesario haber leído el capítulo dedicado al manejo de fechas).

18.1). Insertar en TEMPLE2 una fila por cada empleado cuyo salario total (salario más comisión) supere al salario total medio de su departamento.

Interpretación 1:

Empleamos tres sentencias INSERT, una para los empleados con comisión, y las otras para los restantes, distinguiendo en éstos aquellos en cuyo departamento hay alguien con comisión y los que no (se muestran a continuación, separadas por punto y coma):

```
INSERT INTO    TEMPLE2
        SELECT          *
        FROM            TEMPLE E
        WHERE           COMIS IS NOT NULL AND
                        SALAR + COMIS > (SELECT AVG (SALAR) + AVG (COMIS)
                                         FROM   TEMPLE
                                        WHERE  NUMDE = E.NUMDE) ;

INSERT INTO    TEMPLE2
         SELECT         *
        FROM            TEMPLE E
        WHERE           COMIS IS NULL AND
                        SALAR > (SELECT AVG (SALAR) + AVG (COMIS)
                                    FROM    TEMPLE
                                    WHERE  NUMDE = E.NUMDE) AND
                        0 < (SELECT COUNT (DISTINCT COMIS)
                             FROM   TEMPLE
                            WHERE  NUMDE = E.NUMDE) ;

INSERT INTO    TEMPLE2
         SELECT         *
         FROM           TEMPLE E
         WHERE          COMIS IS NULL AND
                        SALAR > (SELECT AVG (SALAR)
                                    FROM    TEMPLE
                                    WHERE  NUMDE = E.NUMDE) AND
                        0 = (SELECT COUNT (DISTINCT COMIS)
                             FROM    TEMPLE
                            WHERE  NUMDE = E.NUMDE) ;
```

Empleando la función COALESCE se podría formular con una sola sentencia:

```
INSERT INTO             TEMPLE2
        SELECT          *
        FROM            TEMPLE E
        WHERE           COALESCE (SALAR + COMIS, SALAR) >
                                (SELECT COALESCE ( AVG (SALAR) + AVG (COMIS),
                                                   AVG (SALAR))
                                 FROM   TEMPLE
                                 WHERE  NUMDE = E.NUMDE)
```

Interpretación 2:

Si la media de la comisión se interpreta con relación al total de empleados de un departamento, en vez de solamente a aquellos que tienen comisión, la formulación podría ser:

```
INSERT INTO     TEMPLE2
       SELECT   *
       FROM     TEMPLE E
       WHERE    COMIS IS NOT NULL AND
                SALAR + COMIS > (SELECT AVG (SALAR) + SUM (COMIS) / COUNT (*)
                                 FROM   TEMPLE
                                 WHERE  NUMDE = E.NUMDE) ;

INSERT INTO     TEMPLE2
       SELECT   *
       FROM     TEMPLE E
       WHERE    COMIS IS NULL AND
                SALAR > (SELECT AVG (SALAR) + SUM (COMIS) / COUNT (*)
                         FROM    TEMPLE
                         WHERE  NUMDE = E.NUMDE) AND
                EXISTS (SELECT *
                        FROM   TEMPLE
                        WHERE  NUMDE = E.NUMDE AND
                               COMIS IS NOT NULL) ;

INSERT INTO     TEMPLE2
       SELECT   *
       FROM     TEMPLE E
       WHERE    COMIS IS NULL AND
                SALAR > (SELECT AVG (SALAR)
                         FROM      TEMPLE
                         WHERE    NUMDE = E.NUMDE) AND
                NOT EXISTS (SELECT *
                            FROM     TEMPLE
                            WHERE    NUMDE = E.NUMDE AND
                                     COMIS IS NOT NULL) ;
```

En esta solución hemos utilizado el predicado EXISTS para hallar si todas las comisiones de un departamento son *Nulas*, en vez de contarlas como en la solución anterior. La evaluación de este predicado será más eficiente que el de la función COUNT, pues ésta obliga a leer todas las filas del departamento. Como antes, la utilización de COALESCE permite expresar la petición en una sola sentencia:

```
INSERT INTO     TEMPLE2
       SELECT   *
       FROM     TEMPLE E
       WHERE    COALESCE (SALAR + COMIS, SALAR) >
                (SELECT AVG (COALESCE (SALAR + COMIS, SALAR))
                 FROM   TEMPLE
                 WHERE  NUMDE = E.NUMDE)
```

18.2). Borrar en TEMPLE2 a los empleados cuyo salario (sin incluir comisión) supere al salario medio de los empleados de su departamento.

Solución:

```
DELETE FROM TEMPLE2 E
WHERE  SALAR > (SELECT AVG (SALAR)
                FROM   TEMPLE
                WHERE  NUMDE = E.NUMDE)
```

18.3). Borrar en TEMPLE2 a los empleados cuyo salario (sin incluir comisión) supere al salario medio de los empleados de su departamento, excluyéndole a él mismo.

Solución:

```
DELETE FROM TEMPLE2 E
```

```
WHERE  SALAR > (SELECT AVG (SALAR)
                FROM   TEMPLE
               WHERE  NUMDE = E.NUMDE AND
                      NUMEM <> E.NUMEM)
```

18.4). Sumar en TEMPLE2 la comisión en los empleados que la tengan al salario y actualizar éste con este nuevo valor, poniendo además *Nulo* en la comisión.

Solución:

```
UPDATE TEMPLE2
SET    SALAR = SALAR + COMIS, COMIS = NULL
WHERE  COMIS IS NOT NULL
```

18.5). Disminuir en TEMPLE2 en un 5 % el salario de los empleados que superan el 50 % del salario máximo de su departamento.

Solución:

```
UPDATE TEMPLE2 E
SET    SALAR = 0.95 * SALAR
WHERE  SALAR > (SELECT 0.5 * MAX (SALAR)
                FROM   TEMPLE
                WHERE  NUMDE = E.NUMDE)
```

18.6). Borrar todas las filas de TEMPLE2.

Solución:

```
DELETE FROM TEMPLE2
```

18.7). Supongamos que la tabla TEMPLE2 está de nuevo vacía y que disponemos de otra tabla, llamada TBORRA, que tiene una sola columna llamada NUMEM. En esta tabla hay una fila con el número de empleado por cada empleado que causa baja en la empresa en este mes. Hay que extraer de TEMPLE todas las filas de estos empleados y almacenarlas en TEMPLE2 para posteriores procesos, y además borrarlas de TEMPLE.

Solución:

Primero insertamos en TEMPLE2 las filas de los empleados que hay que dar de baja y luego las borramos de TEMPLE.

```
INSERT INTO   TEMPLE2
        SELECT E.*
        FROM   TEMPLE E, TBORRA B
        WHERE  E.NUMEM = B.NUMEM ;

DELETE FROM   TEMPLE E
WHERE  EXISTS (SELECT *
                FROM   TBORRA B
                WHERE  E.NUMEM = B.NUMEM) ;
```

18.8). En TEMPLE2 aumentar el salario de los empleados del departamento 111 en un 10% si no tienen ingresos por comisiones y en un 5% si sí los tienen.

Solución:

```
UPDATE TEMPLE2
SET SALAR = CASE
                WHEN COMIS IS NULL THEN SALAR*1.1
                ELSE SALAR*1.05
            END
WHERE NUMDE = 111
```

CAPITULO 19. DEFINICION DE TABLAS

19.1). Suponiendo que las palabras siguientes sean o bien nombres de tablas o bien constantes, decir qué es cada una.

```
ABC
'ABC'
"ABC"
'Abc'
"Abc"
A.BC
A1.E2
1.E2
‘1.E2'
"A1"."E2"
```

Solución:

ABC	Tabla
'ABC'	Constante de hilera de caracteres
"ABC"	Tabla
'Abc'	Constante de hilera de caracteres
"Abc"	Tabla
A.BC	Tabla
A1.E2	Tabla
1.E2	Constante coma flotante
‘1.E2'	Constante de hilera de caracteres
"A1"."E2"	Tabla

19.2). Crear la tabla TEMPLE2 a la que se refieren los ejercicios del capítulo 18.

Solución:

```
CREATE TABLE TEMPLE2
        ( NUMEM     INTEGER          NOT NULL
        , NUMDE     INTEGER          NOT NULL
        , EXTEL     SMALLINT         NOT NULL
        , FECNA     DATE             NOT NULL
        , FECIN     DATE             NOT NULL
        , SALAR     DECIMAL (4, 0)   NOT NULL
        , COMIS     DECIMAL (4, 0)
        , NUMHI     SMALLINT         NOT NULL
        , NOMEM     VARCHAR (20)     NOT NULL
        )
```

Resulta más cómodo crear la tabla con la cláusula LIKE:

```
CREATE TABLE TEMPLE2
LIKE TEMPLE INCLUDING DEFAULTS
```

19.3). (Variante del ejercicio 18.3). Hacer que el contenido de la tabla TEMPLE2 incluya a los empleados cuyo salario supere al salario medio de su departamento. Borrar entonces de TEMPLE2 a los empleados cuyo salario supere al salario medio de los empleados de su departamento que hay en TEMPLE2. Obtener un listado de todos los empleados que permanezcan, con sus departamentos y salarios. Emplear las sentencias de SQL que sean necesarias.

Solución 1 (creando una tabla de trabajo):

Para borrar los que tengan un salario superior a la media de los que están en su mismo departamento podemos usar una tabla de trabajo, que llamaremos TRA1. Ejecutaremos por tanto las sentencias siguientes.

1. Asegurarse de que TEMPLE2 está vacía.

```
DELETE FROM TEMPLE2;
```

2. Incluir los empleados cuyo salario supera a la media de su departamento.

```
INSERT INTO TEMPLE2
        SELECT *
        FROM   TEMPLE E
        WHERE  SALAR > (SELECT AVG (SALAR)
                        FROM   TEMPLE
                       WHERE  NUMDE = E.NUMDE)
```

3. Definir la tabla de trabajo.

```
:       CREATE TABLE TRA1
        ( NUMDE         INTEGER             NOT NULL
        , SALME         DECIMAL (5, 0)      NOT NULL
        )
```

4. Hallar el salario medio de los departamentos en TEMPLE2.

```
INSERT INTO TRA1
        SELECT   NUMDE, AVG (SALAR)
        FROM     TEMPLE2
        GROUP BY NUMDE
```

5. Borrar de TEMPLE2 los empleados cuyo salario supera al de su departamento.

```
DELETE FROM TEMPLE2 E
WHERE  SALAR > (SELECT SALME
                FROM   TRA1
               WHERE  NUMDE = E.NUMDE)
 GROUP BY NUMDE
```

6. Obtener listado.

```
SELECT NUMEM, NUMDE, SALAR
FROM TEMPLE2
```

7. Al terminar el proceso conviene destruir la tabla intermedia. Aunque la sentencia correspondiente no se ha explicado aún en el presente capítulo, la mostramos aquí para completar el ejercicio (lo mismo haremos en algunos de los siguientes):

```
DROP TABLE TRA1
```

Solución 2 (con tabla anidada):

```
DELETE FROM TEMPLE2;

INSERT INTO TEMPLE2
        SELECT *
        FROM   TEMPLE E
        WHERE  SALAR > (SELECT AVG (SALAR)
                        FROM   TEMPLE
                       WHERE  NUMDE = E.NUMDE) ;

DELETE FROM TEMPLE2 E
WHERE  SALAR > (SELECT SALME
                FROM   (SELECT   NUMDE, AVG (SALAR) AS SALME
                                FROM     TEMPLE2
                                GROUP BY NUMDE) AS TRA1
               WHERE  NUMDE = E.NUMDE) ;

SELECT NUMEM, NUMDE, SALAR
FROM TEMPLE2 ;
```

Solución 3 (con tablas temporales locales):

```
DELETE FROM TEMPLE2;

INSERT INTO TEMPLE2
WITH TEMPLE3 AS
        (  SELECT *
           FROM   TEMPLE E
           WHERE  SALAR > (SELECT AVG (SALAR)
                           FROM   TEMPLE
                           WHERE  NUMDE = E.NUMDE)),
     TRA1 AS
        (SELECT   NUMDE, AVG (SALAR) AS SALME
        FROM     TEMPLE3
        GROUP BY NUMDE)
SELECT E.NUMEM, E.NUMDE, E.SALAR
FROM TEMPLE3 E, TRA1 T
WHERE E.NUMDE = T.NUMDE AND
        SALAR <= SALME      ;

SELECT NUMEM, NUMDE, SALAR
FROM TEMPLE2 ;
```

Resultado:

NUMEM	NUMDE	SALAR
130	112	2900
150	121	4400
180	110	4800
240	111	2800
260	100	7200
285	122	3800
320	122	4050
330	112	2800
360	111	2500
420	130	4000

19.4). (Variante del ejercicio 18.5). Disminuir en TEMPLE2 (resultante del ejercicio anterior) en un 5 % el salario de los empleados que superan el 50 % del salario máximo de los empleados de TEMPLE2 que están en su mismo departamento. Obtener un listado con los empleados y el salario resultante.

Solución 1 (con una tabla de trabajo):

Para hallar el salario máximo de un departamento podemos usar una tabla intermedia de trabajo, que llamaremos TRA1. Ejecutaremos por tanto las sentencias siguientes.

1) Definir la tabla de trabajo:

```
CREATE TABLE TRA1
        ( NUMDE         INTEGER             NOT NULL
        , SAMAX         DECIMAL (5, 0)      NOT NULL
        )
```

2) Hallar el salario máximo de los departamentos en TEMPLE2.

```
INSERT INTO TRA1
        SELECT   NUMDE, MAX (SALAR)
        FROM     TEMPLE2
        GROUP BY NUMDE
```

3) Actualizar el salario de los empleados que superan el 50 % del máximo.

```
UPDATE TEMPLE2 E
SET        SALAR = 0.95 * SALAR
WHERE  SALAR > (SELECT 0.5 * SAMAX
                        FROM   TRA1
                       WHERE  NUMDE = E.NUMDE)
```

4) Obtener listado.

```
SELECT NUMEM, SALARIO
FROM TEMPLE2
```

5) Destruir la tabla intermedia.

```
DROP TABLE TRA1
```

Resultado:

NUMEM	SALAR
130	2755
150	4180
180	4560
240	2660
260	6840
285	3610
320	3847
330	2660
360	2375
420	3800

Solución 2 (con tabla anidada):

```
UPDATE TEMPLE2 E
SET        SALAR = 0.95 * SALAR
WHERE  SALAR > (SELECT 0.5 * SAMAX
                        FROM   (SELECT   NUMDE, MAX (SALAR) AS SAMAX
                                    FROM     TEMPLE2
                                    GROUP BY NUMDE) AS TRA1
                       WHERE  NUMDE = E.NUMDE) ;

SELECT NUMEM, SALAR
FROM TEMPLE2 ;
```

19.5). Crear una tabla llamada TRABA1 con columnas llamadas NUMDE1, NUMDE2, NUMNIV, NOMDIR y MASALA. Las dos primeras columnas contienen números de departamento, la tercera un número entero, la cuarta el nombre de un empleado y la quinta una suma de salarios. Las dos últimas pueden ser *Nulas*, así como la segunda.

Solución:

```
CREATE TABLE TRABA1
        ( NUMDE1        INTEGER                 NOT NULL
        , NUMDE2        INTEGER
        , NUMNIV        INTEGER                 NOT NULL
        , NOMDIR        VARCHAR (20)
        , MASALA        DECIMAL (7, 0)
        )
```

19.6). Hallar para cada departamento su masa salarial total (sin incluir comisión) incluyendo todos sus empleados y los de los departamentos que dependen de él a cualquier nivel (hasta cuatro niveles como máximo). Puede usarse la tabla TRABA1 del ejercicio anterior y más de una sentencia SQL si es necesario.

Solución 1(sin SQL recursivo):

1) Asegurarse de que TRABA1 está vacía.

```
DELETE FROM TRABA1
```

2) Masa salarial de los empleados de cada departamento (nivel 0).

```
INSERT INTO TRABA1 (NUMDE1, NUMNIV, MASALA)
 SELECT   D0.NUMDE, 0, SUM (SALAR)
 FROM     TDEPTO D0, TEMPLE EM
WHERE    D0.NUMDE = EM.NUMDE
GROUP BY D0.NUMDE
```

3) Masa salarial de los empleados de los departamentos que dependen directamente de cada departamento (nivel 1).

```
INSERT INTO TRABA1 (NUMDE1, NUMNIV, MASALA)
 SELECT   D0.NUMDE, 1, SUM (SALAR)
 FROM     TDEPTO D0, TDEPTO D1, TEMPLE EM
 WHERE    D0.NUMDE = D1.DEPDE AND
          D1.NUMDE = EM.NUMDE
 GROUP BY D0.NUMDE
```

4) Masa salarial de los empleados de los departamentos que dependen de cada departamento a nivel 2.

```
INSERT INTO TRABA1 (NUMDE1, NUMNIV, MASALA)
 SELECT   D0.NUMDE, 2, SUM (SALAR)
 FROM     TDEPTO D0, TDEPTO D1, TDEPTO D2, TEMPLE EM
 WHERE    D0.NUMDE = D1.DEPDE AND
          D1.NUMDE = D2.DEPDE AND
          D2.NUMDE = EM.NUMDE
 GROUP BY D0.NUMDE
```

5) Masa salarial de los empleados de los departamentos que dependen de cada departamento a nivel 3.

```
INSERT INTO TRABA1 (NUMDE1, NUMNIV, MASALA)
        SELECT   D0.NUMDE, 3, SUM (SALAR)
        FROM     TDEPTO D0, TDEPTO D1, TDEPTO D2,
                 TDEPTO D3, TEMPLE EM
        WHERE    D0.NUMDE = D1.DEPDE AND
                 D1.NUMDE = D2.DEPDE AND
        D2.NUMDE = D3.DEPDE AND
        D3.NUMDE = EM.NUMDE
        GROUP BY D0.NUMDE
```

6) Masa salarial de los empleados de los departamentos que dependen de cada departamento a nivel 4.

```
INSERT INTO TRABA1 (NUMDE1, NUMNIV, MASALA)
        SELECT   D0.NUMDE, 4, SUM (SALAR)
        FROM     TDEPTO D0, TDEPTO D1, TDEPTO D2,
                 TDEPTO D3, TDEPTO D4, TEMPLE EM
        WHERE    D0.NUMDE = D1.DEPDE AND
                 D1.NUMDE = D2.DEPDE AND
                 D2.NUMDE = D3.DEPDE AND
                 D3.NUMDE = D4.DEPDE AND
        D4.NUMDE = EM.NUMDE
```

```
GROUP BY D0.NUMDE
```

7) Hallar la masa salarial total, a cualquier nivel, de cada departamento.

Solución:

```
SELECT      NUMDE1, SUM (MASALA)
FROM        TRABA1
GROUP BY    NUMDE1
ORDER BY    NUMDE1
```

Resultado:

NUMDE1	COL-2
100	103000
110	45100
111	17450
112	18700
120	31300
121	12400
122	16200
130	11100

Solución 2(con SQL recursivo):

```
WITH TRA1  AS
        (SELECT D.NUMDE, DEPDE, SUM (SALAR) AS MASALA
        FROM TDEPTO D, TEMPLE E
        WHERE D.NUMDE = E.NUMDE
        GROUP BY D.NUMDE, DEPDE),
      TRA2 (DEPP, DEPH, NIV, MASALA) AS
        (SELECT NUMDE, NUMDE, 0, MASALA
        FROM TRA1

        UNION ALL

        SELECT P.DEPP, H.NUMDE, NIV + 1, H.MASALA
        FROM TRA2 P, TRA1 H
        WHERE P.DEPH = H.DEPDE AND NIV< 4
)
SELECT DEPP AS DEPTO, SUM (MASALA) AS MASA_SAL
FROM TRA2
GROUP BY DEPP
ORDER BY DEPP
```

19.7). Escribir las sentencias necesarias para hallar cuántos empleados hay cuyos salarios estén en los intervalos siguientes: (0 a 999), (1000 a 1999), (2000 a 2999), (3000 a 3999), (4000 a 4999), (5000 o más). Hallar también el salario medio dentro de cada intervalo.

Solución1 (con una tabla de trabajo):

Utilizaremos una tabla intermedia de trabajo para almacenar en ella la definición de los intervalos. Ejecutaremos las sentencias siguientes:

1. Definir la tabla de trabajo.

```
CREATE TABLE TRA1
        ( NUMIN         INTEGER             NOT NULL
        , LIMIN         DECIMAL (5, 0)      NOT NULL
```

```
, LIMSU          DECIMAL (5, 0)          NOT NULL
)
```

2. Incluir los límites de los intervalos (las distintas sentencias se muestran separadas por punto y coma).

```
INSERT INTO TRA1 VALUES (1, 000000, 000999) ;
INSERT INTO TRA1 VALUES (2, 001000, 001999) ;
INSERT INTO TRA1 VALUES (3, 002000, 002999) ;
INSERT INTO TRA1 VALUES (4, 003000, 003999) ;
INSERT INTO TRA1 VALUES (5, 004000, 004999) ;
INSERT INTO TRA1 VALUES (6, 005000, 099999) ;
```

3. Agrupar por intervalos.

```
SELECT     NUMIN, COUNT (*) AS EMPLES, AVG (SALAR) AS SALMED
FROM       TEMPLE, TRA1
WHERE      SALAR BETWEEN LIMIN AND LIMSU
GROUP BY NUMIN

UNION

SELECT     NUMIN, 0 AS EMPLES, 0 AS SALMED
FROM       TRA1 T
WHERE      NOT EXISTS (SELECT *
                       FROM    TEMPLE
                       WHERE  SALAR BETWEEN T.LIMIN AND
                                    T.LIMSU)
ORDER BY 1
```

Resultado:

NUMIN	EMPLES	SALMED
1	0	0,00
2	6	1683,33
3	13	2396,15
4	7	3442,85
5	7	4350,00
6	1	7200,00

4. Destruir la tabla intermedia:

```
DROP TABLE TRA1
```

Solución 2 (con una tabla temporal local):

```
WITH TRA1 (NUMIN, LIMIN, LIMSU) AS
          ( VALUES  (1, 000000, 000999),
                    (2, 001000, 001999),
                    (3, 002000, 002999),
                    (4, 003000, 003999),
                    (5, 004000, 004999),
                    (6, 005000, 099999)
          )
SELECT NUMIN, COUNT (NUMEN) AS EMPLES, AVG (SALAR) AS SALMED
FROM TRA1 LEFT OUTER JOIN TEMPLE
     ON SALAR BETWEEN LIMIN AND LIMSU
GROUP BY NUMIN
ORDER BY NUMIN
```

***19.8).** Escribir las sentencias de SQL necesarias para hallar cuántos años hay en los que ha habido ingresos de nuevos empleados.

Solución 1 (con tabla de trabajo):

Ejecutaremos las sentencias siguientes:

1. Definir una tabla de trabajo.

```
:       CREATE TABLE TRA1
            ( NUAÑO         INTEGER             NOT NULL
            )
```

2. Hallar los años en que ha habido ingresos.

```
        INSERT INTO TRA1
            SELECT   DISTINCT YEAR (FECIN)
            FROM     TEMPLE
```

3. Contarlos.

```
        SELECT      COUNT (*)
        FROM        TRA1
```

4. Destruir la tabla intermedia.

```
        DROP TABLE TRA1
```

Solución 2 (con tabla anidada):

```
    SELECT      COUNT (*)
    FROM        (SELECT   DISTINCT YEAR (FECIN)
                 FROM     TEMPLE) AS TRA1
```

Resultado:

```
        COL-1
        --------
            16
```

***19.9).** Hallar el salario medio actual de los empleados que han ingresado cada año.

Solución 1 (con tabla de trabajo):

Ejecutaremos las sentencias siguientes:

1. Definir una tabla de trabajo:

```
        CREATE TABLE TRA1
            ( NUAÑO         INTEGER             NOT NULL
            , NUMEM         INTEGER             NOT NULL
            , SALAR         DECIMAL (5, 0)      NOT NULL
            )
```

2. Hallar los empleados que han ingresado cada año.

```
        INSERT INTO TRA1
            SELECT   YEAR (FECIN), NUMEM, SALAR
            FROM     TEMPLE
```

3. Hallar su salario medio.

```
        SELECT      NUAÑO, AVG (SALAR)
        FROM        TRA1
```

```
GROUP BY    NUAÑO
ORDER BY    NUAÑO
```

4. Destruir la tabla intermedia.

```
DROP TABLE TRA1
```

Solución 2 (agrupando con una expresión):

```
SELECT      YEAR (FECIN) AS NUAÑO, AVG (SALAR) AS MEDAÑO
FROM        TEMPLE
GROUP BY    YEAR (FECIN)
ORDER BY    NUAÑO
```

Resultado:

NUAÑO	MEDAÑO
1948	4400.00
1950	3100.00
1956	4800.00
1959	3800.00
1962	3000.00
1966	3300.00
1967	4500.00
1968	3800.00
1969	2900.00
1971	3550.00
1972	2800.00
1978	4050.00
1984	4500.00
1986	2090.00
1987	1916.66
1988	2075.00

***19.10).** Hallar para cada mes de los años 1987 y 1988 el número de empleados que ingresaron y los valores medio, mínimo y máximo de sus salarios actuales.

Solución 1 (con tabla de trabajo):

1. Definir una tabla de trabajo.

```
CREATE TABLE TRA1
    ( NUAÑO     INTEGER     NOT NULL
    , NUMES     INTEGER     NOT NULL
    )
```

2. Hallar los años y meses en que hubo ingresos de nuevos empleados.

```
INSERT INTO TRA1
      SELECT   DISTINCT YEAR (FECIN), MONTH (FECIN)
      FROM     TEMPLE
     WHERE     YEAR (FECIN) IN (1987, 1988)
```

3. Contar los ingresos y hallar los salarios medio, mínimo y máximo.

```
SELECT      NUAÑO, NUMES, COUNT(*) AS EMPLES, DECIMAL(AVG (SALAR),6,2) AS
            MEDSA, MIN (SALAR) AS MINSA, MAX (SALAR) AS MAXSA
FROM        TRA1, TEMPLE
WHERE       NUAÑO = YEAR (FECIN) AND NUMES = MONTH (FECIN)
GROUP BY    NUAÑO, NUMES
ORDER BY    NUAÑO, NUMES
```

4. Destruir la tabla intermedia.

```
DROP TABLE TRA1
```

Resultado:

NUAÑO	NUMES	EMPLES	MEDSA	MINSA	MAXSA
1987	1	2	1950.00	1900	2000
1987	11	1	1850.00	1850	1850
1988	1	3	1533.33	1000	1800
1988	10	1	1750.00	1750	1750
1988	11	2	3050.00	2100	4000

Solución 2 (con tabla temporal local):

```
WITH TRA1 (NUAÑO, NUMES) AS
        (SELECT   DISTINCT YEAR (FECIN), MONTH (FECIN)
                  FROM    TEMPLE
                 WHERE    YEAR (FECIN) IN (1987, 1988)
        )
SELECT  NUAÑO, NUMES, COUNT(*) AS EMPLES, DECIMAL(AVG (SALAR),6,2) AS MEDSA,
MIN (SALAR) AS MINSA, MAX (SALAR) AS MAXSA
        FROM            TRA1, TEMPLE
        WHERE           NUAÑO = YEAR (FECIN) AND NUMES = MONTH (FECIN)
        GROUP BY        NUAÑO, NUMES
        ORDER BY        NUAÑO, NUMES
```

Solución 3 (agrupando con expresiones):

```
SELECT  YEAR (FECIN) AS NUAÑO, MONTH (FECIN) AS NUMES, COUNT(*) AS EMPLES,
        DECIMAL(AVG (SALAR),6,2) AS MEDSA, MIN (SALAR) AS MINSA, MAX (SALAR) AS
        MAXSA
FROM            TEMPLE
WHERE           YEAR (FECIN) IN (1987, 1988)
GROUP BY        YEAR (FECIN), MONTH (FECIN)
ORDER BY        NUAÑO, NUMES
```

***19.11).** Igual que el ejercicio anterior, pero mostrando en el resultado todos los meses, incluso aquellos en los que no haya habido ingresos de nuevos empleados.

Solución 1 (con tabla de trabajo):

1. Definir una tabla de trabajo.

```
CREATE TABLE TRA1
        ( NUAÑO         INTEGER                 NOT NULL
        , NUMES         INTEGER                 NOT NULL
        )
```

2. Preparar valores de años y meses.

```
INSERT INTO TRA1 VALUES (1987, 1)  ;
INSERT INTO TRA1 VALUES (1987, 2)  ;
INSERT INTO TRA1 VALUES (1987, 3)  ;
INSERT INTO TRA1 VALUES (1987, 4)  ;
INSERT INTO TRA1 VALUES (1987, 5)  ;
INSERT INTO TRA1 VALUES (1987, 6)  ;
INSERT INTO TRA1 VALUES (1987, 7)  ;
INSERT INTO TRA1 VALUES (1987, 8)  ;
```

```
INSERT INTO TRA1 VALUES (1987, 9)  ;
INSERT INTO TRA1 VALUES (1987, 10) ;
INSERT INTO TRA1 VALUES (1987, 11) ;
INSERT INTO TRA1 VALUES (1987, 12) ;
INSERT INTO TRA1 VALUES (1988, 1)  ;
INSERT INTO TRA1 VALUES (1988, 2)  ;
INSERT INTO TRA1 VALUES (1988, 3)  ;
INSERT INTO TRA1 VALUES (1988, 4)  ;
INSERT INTO TRA1 VALUES (1988, 5)  ;
INSERT INTO TRA1 VALUES (1988, 6)  ;
INSERT INTO TRA1 VALUES (1988, 7)  ;
INSERT INTO TRA1 VALUES (1988, 8)  ;
INSERT INTO TRA1 VALUES (1988, 9)  ;
INSERT INTO TRA1 VALUES (1988, 10) ;
INSERT INTO TRA1 VALUES (1988, 11) ;
INSERT INTO TRA1 VALUES (1988, 12) ;
```

3. Contar los ingresos y hallar los salarios medio, mínimo y máximo.

```
SELECT      NUAÑO, NUMES, COUNT(*),
            AVG (SALAR), MIN (SALAR), MAX (SALAR)
FROM        TRA1, TEMPLE
WHERE       NUAÑO = YEAR (FECIN) AND
            NUMES = MONTH (FECIN)
GROUP BY    NUAÑO, NUMES
UNION
SELECT      NUAÑO, NUMES, 0, 0, 0, 0
FROM        TRA1
WHERE       NOT EXISTS (SELECT *
                        FROM   TEMPLE
                        WHERE  YEAR (FECIN) = TRA1.NUAÑO AND
                               MONTH (FECIN) = TRA1.NUMES)
ORDER BY    1, 2
```

4. Destruir la tabla intermedia.

```
DROP TABLE TRA1
```

Solución 2 (con una tabla temporal local):

```
WITH TRA1 (NUAÑO, NUMES) AS
        (VALUES ( (1987, 1), (1987, 2), (1987, 3), (1987, 4), (1987, 5), (1987, 6),
                  (1987, 7), (1987, 8), (1987, 9), (1987, 10), (1987, 11), (1987, 12),
                  (1988, 1), (1988, 2), (1988, 3), (1988, 4), (1988, 5), (1988, 6),
                  (1988, 7), (1988, 8) ,(1988, 9), (1988, 10), (1988, 11), (1988, 12)
                  )
         )
SELECT  NUAÑO, NUMES, COUNT (NUMEM) AS EMPLES, AVG (SALAR) AS MEDSA,
        MIN (SALAR) AS MINSA, MAX (SALAR) AS MAXSA
FROM TRA1 LEFT OUTER JOIN TEMPLE
     ON NUAÑO = YEAR (FECIN) AND NUMES = MONTH (FECIN)
GROUP BY NUAÑO, NUMES
ORDER BY NUAÑO, NUMES
```

19.12). (Variante del ejercicio 11.21). Cada director de departamento, tanto en propiedad como en funciones, es responsable de las promociones y sueldos de los empleados del departamento que dirige, excluido él mismo. Además la promoción y sueldo de un director en propiedad es responsabilidad del primero de sus directores en propiedad que se halle ascendiendo en la jerarquía de la organización. Escribir las sentencias SQL necesarias para hallar para cada director su nombre y la masa salarial total de los empleados y directores de cuya promoción es responsable, por orden alfabético.

Solución 1 (sin recursividad):

Ejecutaremos los pasos siguientes.

1. Definir una tabla de trabajo.

```
CREATE TABLE TRA1
        ( NUDEP         INTEGER             NOT NULL
        , NUDIR         INTEGER             NOT NULL
        , NUNIV         INTEGER             NOT NULL
        , MASAL         DECIMAL (7, 0)      NOT NULL
        )
```

2. Masa salarial de los empleados de cada departamento.

```
INSERT INTO TRA1
        SELECT   D0.NUMDE, D0.DIREC, 0, SUM (SALAR)
        FROM     TDEPTO D0, TEMPLE EM
        WHERE    D0.NUMDE = EM.NUMDE AND
                 D0.DIREC <> EM.NUMEM
       GROUP BY D0.NUMDE, D0.DIREC
```

3. Masa salarial de los directores en propiedad de los departamentos que dependen directamente de cada director en propiedad (nivel 1).

```
INSERT INTO TRA1
SELECT   D0.NUMDE, D0.DIREC, 1, SUM (SALAR)
 FROM     TDEPTO D0, TDEPTO D1, TEMPLE EM
 WHERE    D0.NUMDE = D1.DEPDE AND
    D0.TIDIR = 'P' AND D1.TIDIR = 'P' AND
    D1.DIREC = EM.NUMEM
  GROUP BY D0.NUMDE, D0.DIREC
```

4. Masa salarial de los directores en propiedad de los departamentos que dependen de cada director en propiedad a nivel 2.

```
INSERT INTO TRA1
        SELECT   D0.NUMDE, D0.DIREC, 2, SUM (SALAR)
        FROM     TDEPTO D0, TDEPTO D1, TDEPTO D2,  TEMPLE EM
        WHERE    D0.NUMDE = D1.DEPDE AND
                 D1.NUMDE = D2.DEPDE AND
                 D0.TIDIR = 'P' AND D1.TIDIR = 'F' AND
                 D2.TIDIR = 'P' AND
                 D2.DIREC = EM.NUMEM
        GROUP BY D0.NUMDE, D0.DIREC
```

5. Masa salarial de los directores en propiedad de los departamentos que dependen de cada director en propiedad a nivel 3.

```
INSERT INTO TRA1
        SELECT   D0.NUMDE, D0.DIREC, 3, SUM (SALAR)
        FROM     TDEPTO D0, TDEPTO D1, TDEPTO D2,
                 TDEPTO D3, TEMPLE EM
        WHERE    D0.NUMDE = D1.DEPDE AND
                 D1.NUMDE = D2.DEPDE AND
                 D2.NUMDE = D3.DEPDE AND
                 D0.TIDIR = 'P' AND D1.TIDIR = 'F' AND
                 D2.TIDIR = 'F' AND D3.TIDIR = 'P' AND
                 D3.DIREC = EM.NUMEM
        GROUP BY D0.NUMDE, D0.DIREC
```

6. Masa salarial de los directores en propiedad de los departamentos que dependen de cada director en propiedad a nivel 4.

```
INSERT INTO TRA1
        SELECT   D0.NUMDE, D0.DIREC, 4, SUM (SALAR)
        FROM     TDEPTO D0, TDEPTO D1, TDEPTO D2,
                 TDEPTO D3, TDEPTO D4, TEMPLE EM
        WHERE    D0.NUMDE = D1.DEPDE AND
                 D1.NUMDE = D2. DEPDE AND
                 D2.NUMDE = D3. DEPDE AND
                 D3.NUMDE = D4. DEPDE AND
                 D0.TIDIR = 'P' AND D1.TIDIR = 'F' AND
                 D2.TIDIR = 'F' AND D3.TIDIR = 'F' AND
                 D4.TIDIR = 'P' AND
                 D4.DIREC = EM.NUMEM
        GROUP BY D0.NUMDE, D0.DIREC
```

7. Comprobar que no hay en nuestra organización dependencias de departamentos a nivel mayor que 2.

```
SELECT MAX (NUNIV)
FROM   TRA1
```

8. Hallar la masa salarial total, a cualquier nivel, de cada director en propiedad.

```
SELECT          NOMEM, SUM (MASAL)
FROM            TRA1, TEMPLE
WHERE           NUDIR = NUMEM
GROUP BY        NUDIR, NOMEM
ORDER BY        1
```

9. Destruir la tabla intermedia.

```
DROP TABLE TRA1
```

Solución 2 (con recursividad):

```
WITH TRA1 (D1, D2, N, DIR, TIP, SALA) AS
    (SELECT D.NUMDE, E.NUMDE, 0, DIREC, TIDIR, SUM (SALAR)
            FROM TDEPTO D, TEMPLE E
            WHERE D.NUMDE = E.NUMDE AND D.DIREC <> E.NUMEM
            GROUP BY D.NUMDE, E.NUMDE, DIRE, TIDIR),
      TRA (DEPP, DEPH, NIV, NUDIRP, TIDIRH, SALA) AS
            (SELECT *  FROM TRA1

            UNION ALL

            SELECT P.DEPP, H.NUMDE, NIV + 1, NUDIRP, TIDIR, E.SALAR
            FROM TRA P, TDEPTO H, TEMPLE E
            WHERE P.DEPH = H.DEPDE AND
                    ((NIV = 0 AND TIDIRH = 'P') OR (NIV > 0 AND TIDIRH = 'F'))
                    AND H.DIREC = E.NUMEM

            )

    SELECT NOMEM, SUM (SALA) AS MASA_SAL
    FROM TRA, TEMPLE
    WHERE (NIV = 0 OR TIDIRH = 'P') AND NUDIRP = NUMEM
    GROUP BY NOMEM
    ORDER BY NOMEM
```

Resultado:

```
NOMEM                    MASA_SAL
-------------------------  ------------------
CAMPS, AURELIO                 11700
GARCIA, AUGUSTO                 6900
GARCIA, OCTAVIO                14900
LOPEZ, ANTONIO                 26200
PEREZ, JULIO                   10700
PEREZ, MARCOS                  25400
```

19.13). Crear una tabla con dos columnas, una *smallint* y otra *integer*. Dar de alta filas en la tabla de manera que los datos de la columna primera vayan del 1 al 20 y los de la segunda del 20 al 1.

Solución:

```
CREATE TABLE TABLA2 (COL1 SMALLINT, COL2 INTEGER) ;

INSERT INTO TABLA2
        WITH TTEMP (C1,C2) AS
                (VALUES (1,20)
                        UNION ALL
                SELECT (C1+1), (C2-1)
                FROM TTEMP
                WHERE C1< 20)
        SELECT * FROM TTEMP ;
```

CAPITULO 20. USO DE OTROS TIPOS DE DATOS (UDT, LOB) Y FUNCIONES DE USUARIO

20.1). Definir un UDT “KMHORA” que se base en un tipo de datos DEC(4,0). Definir las operaciones básicas de suma y resta sobre el UDT. Crear una tabla que contenga las siguientes columnas (tipos de datos): LOCALIZACIÓN_1 (VARCHAR(20)), LOCALIZACIÓN_2 (VARCHAR(20), TIPO_DE_CARRETERA (CHAR(1)), VELOCIDAD_MAXIMA (KMHORA).

Solución:

```
CREATE DISTINCT TYPE KMHORA AS DECIMAL(4,0) WITH COMPARISONS;

CREATE FUNCTION "+" (KMHORA, KMHORA)
RETURNS KMHORA
SOURCE SYSIBM."+" (DECIMAL(4,0), DECIMAL(4,0));

CREATE FUNCTION "-" (KMHORA, KMHORA)
RETURNS KMHORA
SOURCE SYSIBM."-" (DECIMAL(4,0), DECIMAL(4,0));

CREATE TABLE CARRETERAS
(LOCALIZACION_1 VARCHAR(20),
LOCALIZACION_2 VARCHAR(20),
TIPO_DE_CARRETERA CHAR(1),
VELOCIDAD_MAXIMA KMHORA) ;
```

20.2). Definir las funciones escalares MAX y MIN sobre el UDT anterior. Escribir una sentencia SQL que permita obtener cuales son las dos localidades comunicadas por la carretera en la que se puede viajar a más velocidad.

Solución:

```
CREATE FUNCTION MAX (KMHORA)
RETURNS KMHORA SOURCE SYSIBM.MAX (DECIMAL (4,0));
```

```
CREATE FUNCTION MIN (KMHORA)
RETURNS KMHORA SOURCE SYSIBM.MIN (DECIMAL (4,0));

SELECT LOCALIZACION_1, LOCALIZACION_2
FROM CARRETERAS
WHERE VELOCIDAD_MAXIMA = (SELECT MAX (VELOCIDAD_MAXIMA)
                          FROM CARRETERAS)
```

CAPITULO 21. VISTAS Y AUTORIZACIONES

21.1). Crear una vista donde aparezcan todas las filas de la tabla TDEPTO pero no aparezca la columna PRESU. Hacerla pública para consultas.

Solución:

```
CREATE VIEW VDEPTO
        AS SELECT NUMDE, NUMCE, DIREC,
                  TIDIR, DEPDE, NOMDE
           FROM   TDEPTO ;

GRANT  SELECT ON VDEPTO TO PUBLIC ;
```

21.2). Crear una vista llamada VEMCOM que contenga las columnas NUMEM, NUMDE, EXTEL y NOMEM, de los empleados que trabajan a comisión. Hacerla pública para consultas.

Solución:

```
CREATE VIEW VENCOM AS
            SELECT NUMEM, NUMDE, EXTEL, NOMEM
            FROM   TEMPLE
            WHERE  COMIS IS NOT NULL ;

GRANT  SELECT ON VENCOM TO PUBLIC ;
```

***21.3).** Crear una vista llamada VJUBIL1 en la que aparezcan todos los datos de los empleados que cumplen 65 años de edad este año de modo que puedan ser consultados solamente por el director de Personal, suponiendo que éste tiene como identificador U0150.

Solución:

```
CREATE VIEW VJUBIL1 AS
        SELECT *
        FROM    TEMPLE
        WHERE  YEAR (CURRENT DATE) - YEAR (FECNA) = 65 ;

GRANT  SELECT ON VJUBIL1 TO U0150 ;
```

***21.4).** (Variante del ejercicio anterior). Crear una vista llamada VJUBIL2 en la que aparezcan todos los datos de los empleados que cumplen 65 años o más en este año de modo que puedan ser consultados solamente por el director de Personal, suponiendo que los identificadores de todos los empleados están almacenados en la tabla TIDPER descrita en un ejemplo del párrafo sobre "Utilidad de las vistas" en este capítulo.

Solución:

```
CREATE VIEW VJUBIL2 AS
        SELECT E.*
        FROM    TIDPER I, TDEPTO D, TEMPLE E
        WHERE  IDPEM = USER AND
                I.NUMEM = DIREC AND
                NOMDE LIKE '%PERSONAL%' AND
                YEAR (CURRENT DATE) - YEAR (FECNA) >= 65;

GRANT  SELECT ON VJUBIL2 TO PUBLIC ;
```

Esta vista está vacía para todos los usuarios excepto el director de Personal. Si éste cambia, la vista refleja este cambio automáticamente en cuanto se haya actualizado TDEPTO.

***21.5).** Crear una vista llamada VJUBIL3 en la que aparezcan todos los datos de los empleados que cumplen 65 años o más este año de modo que estos datos puedan ser consultados solamente por los empleados del departamento de Personal.

Solución:

```
CREATE VIEW VJUBIL3  AS
        SELECT J.*
        FROM    TIDPER I, TEMPLE E, TDEPTO D,  TEMPLE J
        WHERE  IDPEM = USER AND
                    I.NUMEM = E.NUMEM AND
                    E.NUMDE = D.NUMDE AND
                    D.NOMDE LIKE '%PERSONAL%' AND
                    YEAR (CURRENT DATE) - YEAR (J.FECNA) >= 65 ;

GRANT  SELECT ON VJUBIL3 TO PUBLIC ;
```

21.6). Crear una vista llamada VINIDE, con columnas llamadas NUMDE1, NIVEL, NUMDE2, NOMDIR y MASALA. Las columnas NUMDE1 y NUMDE2 son números de departamento. NUMDE2 depende administrativamente de NUMDE1, a un nivel que viene dado por el contenido de la columna NIVEL. En NOMDIR está el nombre del director del departamento NUMDE2. En MASALA la masa salarial total de NUMDE2. La vista debe contener los departamentos dependientes administrativamente de otros desde el nivel 0 hasta el 4, como máximo.

Solución 1 (sin recusrsividad):

```
CREATE VIEW VINIDE (NUMDE1, NIVEL, NUMDE2, NOMDIR, MASALA)
AS
         SELECT   D0.NUMDE, 0, D0.NUMDE, ED.NOMEM, SUM (EM.SALAR)
         FROM     TDEPTO D0, TEMPLE ED, TEMPLE EM
         WHERE D0.DIREC = ED.NUMEM AND D0.NUMDE = EM.NUMDE
         GROUP BY D0.NUMDE, ED.NOMEM
UNION
         SELECT   D0.NUMDE, 1, D1.NUMDE, ED.NOMEM, SUM (EM.SALAR)
         FROM     TDEPTO D0, TDEPTO D1, TEMPLE ED, TEMPLE EM
         WHERE    D0.NUMDE = D1.DEPDE AND
                  D1.DIREC = ED.NUMEM AND D1.NUMDE = EM.NUMDE
         GROUP BY D0.NUMDE, D1.NUMDE, ED.NOMEM
UNION
         SELECT   D0.NUMDE, 2, D2.NUMDE, ED.NOMEM, SUM (EM.SALAR)
         FROM     TDEPTO D0, TDEPTO D1, TDEPTO D2, TEMPLE ED,TEMPLE EM
         WHERE    D0.NUMDE = D1.DEPDE AND
                  D1. NUMDE = D2. DEPDE AND
                  D2.DIREC = ED.NUMEM AND D2.NUMDE = EM.NUMDE
         GROUP BY D0.NUMDE, D2.NUMDE, ED.NOMEM
UNION
        SELECT   D0.NUMDE, 3, D3.NUMDE, ED.NOMEM, SUM (EM.SALAR)
        FROM     TDEPTO D0, TDEPTO D1, TDEPTO D2, TDEPTO D3,
                 TEMPLE ED, TEMPLE EM
        WHERE    D0.NUMDE = D1.DEPDE AND
                 D1.NUMDE = D2.DEPDE AND
                 D2.NUMDE = D3.DEPDE AND
                 D3.DIREC = ED.NUMEM AND D3.NUMDE = EM.NUMDE
        GROUP BY D0.NUMDE, D3.NUMDE, ED.NOMEM
UNION
        SELECT   D0.NUMDE, 4, D4.NUMDE, ED.NOMEM,  SUM (EM.SALAR)
        FROM     TDEPTO D0, TDEPTO D1, TDEPTO D2, TDEPTO D3, TDEPTO D4,
                 TEMPLE ED, TEMPLE EM
        WHERE    D0.NUMDE = D1.DEPDE AND
                 D1.NUMDE = D2. DEPDE AND
                 D2.NUMDE = D3. DEPDE AND
```

```
                    D3.NUMDE = D4. DEPDE AND
                    D4.DIREC = ED.NUMEM AND D4.NUMDE = EM.NUMDE
            GROUP BY D0.NUMDE, D4.NUMDE, ED.NOMEM
```

Solución 2 (con recursividad):

```
CREATE VIEW VINIDE (NUMDE1, NIVEL, NUMDE2, NOMDIR, MASALA) AS
    WITH TRA1  AS
        (SELECT D.NUMDE, DEPDE, ED.NOMEM AS NOMBR, SUM (EM.SALAR) AS MASALA
        FROM TDEPTO D, TEMPLE ED, TEMPLE EM
        WHERE D.DIREC = ED.NUMEM AND D.NUMDE = EM.NUMDE
        GROUP BY D.NUMDE, DEPDE, ED.NOMEM),
      TRA2 (DEPP, NIV, DEPH, NOMDIRH, MASALA) AS
        (SELECT NUMDE, 0, NUMDE, NOMBR, MASALA
        FROM TRA1

        UNION ALL

        SELECT P.DEPP, NIV + 1, H.NUMDE, H.NOMBR, H.MASALA
        FROM TRA2 P, TRA1 H
        WHERE P.DEPH = H.DEPDE AND NIV< 4
        )
    SELECT *
    FROM TRA2
```

21.7). Vaciar de filas la tabla TRABA1 creada en los ejercicios del capítulo sobre "Definición de tablas" y llenarla de nuevo con las dependencias entre departamentos hasta cuatro niveles como máximo.

Solución:

```
        DELETE FROM TRABA1 ;

        INSERT INTO TRABA1 SELECT NUMDE1, NUMDE2, 2, NOMDIR, MASALA
                                FROM   VINIDE ;
```

21.8). Crear una vista sobre TRABA1 llamada VIORDE (acrónimo de VIsta de la ORganización de los DEpartamentos) donde aparezcan las columnas NUMDE1, NUMDE2, NUMNIV y NOMDIR de TRABA1 y todas sus filas. Hacer pública para consultas esta vista.

Solución:

```
        CREATE VIEW VIORDE AS
            SELECT NUMDE1, NUMDE2, NUMNIV, NOMDIR
            FROM   TRABA1 ;

        GRANT  SELECT ON VIORDE TO PUBLIC ;
```

21.9). Utilizando la tabla de usuarios TIDPER mencionada más arriba crear una vista llamada VIORDI (VIsta de la Organización para DIrectores) tal que aparezcan en ella todas las columnas de TRABA1, pero sólo las filas de los departamentos de los que el usuario sea director, inmediato o no. Hacer pública para consultas esta vista.

Solución:

```
        CREATE VIEW VIORDI  AS
            SELECT T.*
            FROM   TIDPER U, TDEPTO D, TRABA1 T
            WHERE  IDPEM = USER AND
                        U.NUMEM = D.DIREC AND
                        D.NUMDE = T.NUMDE1 ;

        GRANT  SELECT ON VIORDI TO PUBLIC ;
```

De este modo la vista está vacía para usuarios que no sean directores, y para los que lo sean sólo contiene filas de los departamentos que dirige o que dependen de éstos.

Obsérvese sin embargo que un cambio en TDEPTO o TEMPLE (por ejemplo, si cambia el director de un departamento), no se refleja automáticamente en esta vista. Para que ésta represente la última situación hay que actualizar el contenido de TRABA1 cada vez que haya cambios que la puedan afectar. Si se desea evitar esta situación, se podría hacer de TRABA1 una tabla temporal local definida dentro del CREATE VIEW, como ya se hizo en el ejercicio 21.6 . Dejamos este ejercicio al lector.

21.10). Se desea autorizar a cada director de departamento a ver todos los datos de los empleados de los departamentos que dirige, tanto en propiedad como en funciones. Crear una vista para ello y autorizarla al público para consultas.

Solución:

```
CREATE VIEW VEMPLE AS
        SELECT E.*
        FROM   TIDPER U, TDEPTO D, TEMPLE E
        WHERE  IDPEM = USER AND
                    U.NUMEM = D.DIREC AND
                    D.NUMDE = E.NUMDE ;

GRANT  SELECT ON VEMPLE TO PUBLIC ;
```

Como en el ejercicio anterior, la vista está vacía para usuarios que no sean directores, y para los que lo sean sólo contiene filas de los empleados de los departamentos que dirige. Pero, al contrario que en el ejercicio anterior, los cambios en TDEPTO, TIDPER o TEMPLE, son automáticamente reflejados en la vista puesto que ésta está definida directamente sobre estas tablas (en vez de sobre una tabla intermedia como era el caso en el ejercicio anterior).

21.11). Se desea autorizar a cada director de departamento a ver todos los datos de los empleados de los departamentos que dirige o de los que dependen de ellos a cualquier nivel. Crear una vista para ello y autorizar para que pueda ser consultada.

Solución:

Para acceder a los departamentos dependientes podemos usar la TRABA1 con el contenido almacenado en ella en el ejercicio 21.7 anterior. Puesto que usaremos una tabla intermedia, es aplicable la observación que hacíamos en el ejercicio 21.9, a saber, que las actualizaciones en las tablas que afecten a las dependencias o directores de los departamentos no se reflejan automáticamente en TRABA1, ni por lo tanto tampoco en nuestra vista. Para que la vista estuviera actualizada en todo momento habría que definir TRABA1 como una vista temporal local en el CREATE VIEW, como hicimos en el ejercicio 21.6, lo cual dejamos al lector.

```
CREATE VIEW VEMPLED AS
        SELECT E.*, NUMNIV
        FROM    TIDPER U, TRABA1 D, TEMPLE E
        WHERE  IDPEM = USER AND
                    U.NUMEM = D.NUMDE1 AND
                    D.NUMDE2 = E.NUMDE ;

GRANT  SELECT ON VEMPLED TO PUBLIC ;
```