

COUNT()	Cuenta el número de filas devueltas por un comando SQL
SUM()	Devuelve la suma de la columna o expresiones especificadas
AVG()	Sirve para calcular la media de la columna o expresión especificada
MAX()	Devuelve el valor más alto de la columna o expresión especificada.
MIN()	Devuelve el valor más bajo de la columna o expresión especificada

13. AGRUPACIÓN DE FILAS

El operador GROUP BY reorganiza en sentido lógico la tabla representada por la cláusula FROM, formando grupos de manera que dentro de un grupo dado todas las filas tengan el mismo valor en el campo de GROUP BY.

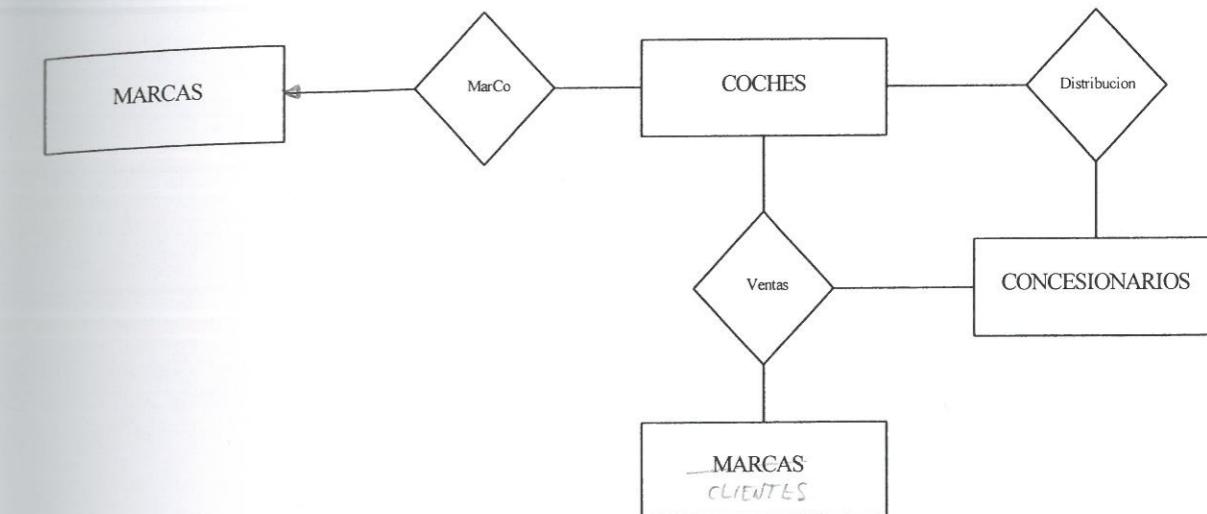
```
SELECT <columna/expresión>
FROM tablas
WHERE <condición>
GROUP BY <columna/expresión> [HAVING <condición>];
```

Cuando se usa la cláusula GROUP BY, los nombres de columnas que aparecen en SELECT deben especificarse también en esta cláusula.

La cláusula HAVING funciona en cada grupo como WHERE en cada fila.

14. TABLAS DE TRABAJO

Para realizar los problemas que se van a resolver a continuación es necesario contar con un caso sobre el cual poder trabajar. Se trata de una base de datos que contenga información sobre MARCAS de COCHES, los diversos MODELOS que tiene cada marca, CONCESIONARIOS que venden esos coches además de las VENTAS realizadas por estos últimos a los clientes. Para ello se propone el siguiente modelo E-R.



Este diseño una vez desarrollado, tal y como se hizo en el capítulo 1, dará origen a las siguientes tablas relationales:

MARCAS (cifm, nombre, ciudad)
COCHES (codcoche, nombre, modelo) *cifm , precio*
CONCESIONARIO (cifc, nombre, ciudad)
CLIENTES (dni, nombre, apellidos, ciudad)
DISTRIBUCION (cifc, codcoche, cantidad)
VENTAS (cifc, dni, codcoche, color)
MARCO (cifm, codcoche)

las cuales han sido cargadas con los siguientes datos

MARCAS

CIFM	NOMBRE	CIUDAD
0001	SEAT	MADRID
0002	RENAULT	BARCELONA
0003	CITROEN	VALENCIA
0004	AUDI	MADRID
0005	OPEL	BILBAO
0006	BMW	BARCELONA

CLIENTES

DNI	NOMBRE	APELLIDO	CIUDAD
0001	LUIS	GARCIA	MADRID
0002	ANTONIO	LOPEZ	VALENCIA
0003	JUAN	MARTIN	MADRID
0004	MARIA	GARCIA	MADRID
0005	JAVIER	GONZALEZ	BARCELONA
0006	ANA	LOPEZ	BARCELONA

VENTAS

CIFC	DNI	CODCOCHE	COLOR
0001	0001	001	BLANCO
0001	0002	005	ROJO
0002	0003	008	BLANCO
0002	0001	006	ROJO
0003	0004	011	ROJO
0004	0005	014	VERDE

COCHES

CODCOCHE	NOMBRE	MODELO
001	IBIZA	GLX
002	IBIZA	GTI
003	IBIZA	GTD
004	TOLEDO	GTD
005	CORDOBA	GTI
006	MEGANE	1.6
007	MEGANE	GTI
008	LAGUNA	GTD
009	LAGUNA	TD
010	ZX	16V
011	ZX	TD
012	XANTIA	GTD
013	A4	1.8
014	A4	2.8
015	ASTRA	CARAVAN
016	ASTRA	GTI
017	CORSA	1.4
018	300	316.i
019	500	525i
020	700	750i

CONCESIONARIO

CIFC	NOMBRE	CIUDAD
0001	ACAR	MADRID
0002	BCAR	MADRID
0003	CCAR	BARCELONA
0004	DCAR	VALENCIA
0005	ECAR	BILBAO

DISTRIBUCION

CIFC	CODCOCHE	CANTIDAD
0001	001	3
0001	005	7
0001	006	7
0002	006	5
0002	008	10
0002	009	10
0003	010	5
0003	011	3
0003	012	5
0004	013	10
0004	014	5
0005	015	10
0005	016	20
0005	017	8
0006	019	3

MARCO

CIFM	CODCOCHE
0001	001
0001	002
0001	003
0001	004
0001	005
0002	006
0002	007
0002	008
0002	009
0003	010
0003	011
0003	012
0004	013
0004	014
0005	015
0005	016
0005	017
0006	018
0006	019
0006	020

15. PROBLEMAS RESUELTOS

1.-Crear la tabla COCHES

```
CREATE TABLE coches
(codcoche          SMALLINT      NOT NULL,
nombre            CHAR(10)     NOT NULL,
modelo           CHAR(10)     NOT NULL,
PRIMARY KEY (codcoche);
```

2.-Crear la tabla DISTRIBUCION

```
CREATE TABLE distribución
(cifc              SMALLINT      NOT NULL,
codcoche          SMALLINT      NOT NULL,
cantidad          SMALLINT      NOT NULL,
PRIMARY KEY (cifc, codcoche),
FOREING KEY (cifc) REFERENCES (concesionario),
FOREING KEY (codcoche) REFERENCES (coches);
```

3.-Crear un índice en la tabla MARCAS del campo CIFM llamado ICIFM

```
CREATE UNIQUE INDEX icifm ON marcas (cifm)
```

4.-Crear un índice en la tabla COCHES de todos los campos llamado ITODOS

```
CREATE UNIQUE INDEX itodos ON coches (codcoche, nombre, modelo)
```

5.-Añadir en la tabla MARCAS un nuevo campo llamado PAIS

```
ALTER TABLE marcas ADD pais CHAR(15);
```

6.-Borrar la tabla DISTRIBUCION

```
DROP TABLE distribucion;
```

7.-Obtener todos los campos de todos los concesionarios

```
SELECT *
FROM concesionario;
```

CIFC	NOMBRE	CIUDAD
0001	ACAR	MADRID
0002	BCAR	MADRID
0003	CCAR	BARCELONA
0004	DCAR	VALENCIA
0005	ECAR	BILBAO

8.-Obtener todos los campos de todos los clientes de 'MADRID'

```
SELECT *
FROM clientes
WHERE ciudad='MADRID';
```

DNI	NOMBRE	APELLIDO	CIUDAD
0001	LUIS	GARCIA	MADRID
0003	JUAN	MARTIN	MADRID
0004	MARIA	GARCIA	MADRID

9.-Obtener los nombres de todas las MARCAS de coches ordenadas alfabéticamente

```
SELECT nombre
FROM marcas
ORDER BY nombre;
```

NOMBRE
AUDI
BMW
CITROEN
OPEL
RENAULT
SEAT

10.-Obtener el cife de todos los concesionarios cuya cantidad en la tabla de DISTRIBUCION es mayor que 18

```
SELECT cife
FROM distribucion
WHERE cantidad > 18;
```

CIFC
0005

11.-Obtener el cife de todos los concesionarios cuya cantidad en la tabla de DISTRIBUCION está comprendida entre 10 y 18 ambos inclusive.

```
SELECT cife
FROM distribucion
WHERE cantidad BETWEEN 10 AND 18;
```

CIFC
0002
0002
0004
0005
0005

12.-Obtener el cifc de todos los concesionarios cuya cantidad en la tabla de DISTRIBUCION está comprendida entre 10 y 18, ambos inclusive. (De otra manera)

```
SELECT cifc
FROM distribucion
WHERE cantidad >=10
AND cantidad <=18;
```

13.-Obtener el cifc de todos los concesionarios que han adquirido más de 10 coches o menos de 5.

```
SELECT cifc
FROM distribucion
WHERE cantidad >10
OR cantidad <5;
```

CIFC
0001
0003
0005
0006

14.-Obtener todas las parejas de cifm de marcas y dni de clientes que sean de la misma ciudad.

Para resolver este problema es necesario reunir las tablas MARCAS y CLIENTES según el campo ciudad y después proyectar sobre los campos pedidos.

```
SELECT cifm, dni
FROM marcas, clientes
WHERE marcas.ciudad=clientes.ciudad;
```

CIFM	DNI
0001	0001
0001	0003
0001	0004
0002	0005
0002	0006
0003	0002
0004	0001
0004	0003
0004	0004
0006	0005
0006	0006
0005	0003
0006	0004

15.-Obtener todas las parejas de dni de clientes y cifm de marcas que NO sean de la misma ciudad.

```
SELECT cifm, dni
FROM marcas, clientes
WHERE NOT (marcas.ciudad=clientes.ciudad);
```

CIFM	DNI
0001	0002
0001	0005
0001	0006
0002	0001
0002	0002
0002	0003
0002	0004
0003	0001
0003	0003
0003	0004
0003	0005
0003	0006

CIFM	DNI
0004	0002
0004	0005
0004	0006
0005	0001
0005	0002
0005	0003
0005	0004
0005	0005
0005	0006
0006	0001
0006	0002
0006	0003
0006	0004

16.-Obtener los codcoche suministrados por algún concesionario de 'BARCELONA'

```
SELECT codcoche
FROM distribucion, concesionario
WHERE distribucion.cifc=concesionario.cifc
AND concesionario.ciudad='BARCELONA';
```

CODCOCHE
010
011
012

17.-Obtener el codcoche de aquellos coches vendidos a clientes de 'MADRID'.

Para resolver este problema, habrá primero que calcular el dni de los clientes que son de Madrid y después ver cuáles de ellos se encuentran en la tabla VENTAS. Esto se puede hacer de varias maneras aunque en este ejercicio lo haremos como una simple reunión entre tablas, es decir;

```
SELECT codcoche
FROM ventas, clientes
WHERE clientes.ciudad='Madrid'
AND clientes.dni=ventas.dni;
```

CODCOCHE
001
005
011
014

18.-Obtener el codcoche de los coches que han sido adquiridos por un cliente de 'MADRID' a un concesionario de 'MADRID'.

Para resolver este problema, será necesario utilizar tres tablas, VENTAS, CONCESIONARIO y CLIENTES. El proceso de resolución será el siguiente; encontrar los clientes de Madrid, encontrar los

concesionarios de Madrid y obtener los codcoche que en la tabla de VENTAS tienen a su lado un cifc de los hallados anteriormente con un dni de los clientes que residen en Madrid. Esto en SQL se puede realizar de diversas maneras aunque en este ejercicio lo haremos como una mera reunión entre tablas.

```
SELECT codcoche
FROM ventas, clientes, concesionario
WHERE concesionario.ciudad='MADRID'
AND clientes.ciudad='MADRID'
AND concesionario.cifc=ventas.cifc
AND clientes.dni=ventas.dni;
```

Básicamente lo que se ha hecho es realizar el producto natural de las tablas CONCESIONARIO, VENTAS y CLIENTES según los campos comunes cifc y dni y luego se han seleccionado aquellas tuplas cuyos campos concesionario.ciudad y clientes.ciudad tuviesen un valor igual a 'MADRID'.

CODCOCHE
001
008
006

19.-Obtener los codcoche de los coches comprados en un concesionario de la misma ciudad que el cliente que lo compra.

La manera de realizar este problema es similar al anterior, aunque ahora lo que hay que hacer es imponer que las ciudades del concesionario y del cliente sean las mismas.

```
SELECT codcoche
FROM ventas, clientes, concesionario
WHERE concesionario.ciudad= clientes.ciudad
AND concesionario.cifc=ventas.cifc
AND clientes.dni=ventas.dni;
```

CODCOCHE
001
008
006

20.-Obtener los codcoche de los coches comprados en un concesionario de distinta ciudad que el cliente que lo compra

```
SELECT codcoche
FROM ventas, clientes, concesionario
WHERE concesionario.ciudad <> clientes.ciudad
AND concesionario.cifc=ventas.cifc
AND clientes.dni=ventas.dni;
```

CODCOCHE
005
011
014

21.-Obtener todas las parejas de nombre de marcas que sean de la misma ciudad.

El proceso de resolución de este problema consistirá en realizar una reunión natural de la tabla MARCAS consigo misma según el campo ciudad, para después seleccionar las tuplas resultantes y proyectar sobre los campos nombre. La dificultad reside en que en SQL una tabla no se puede reunir consigo misma (p.e. marcas.ciudad = marcas.ciudad) ya que esta condición siempre es verdadera. Para ello lo que hay que hacer es darle un alias a la tabla de manera que es como si se trabajara con dos tablas distintas.

En este caso, a la tabla MARCAS se le darán dos alias distintos (A y B) de manera que a efectos formales es como si se tratara de dos tablas totalmente distintas.

```
SELECT A.nombre, B.nombre
FROM marcas A, marcas B
WHERE A.ciudad = B.ciudad
AND A.nombre > B.nombre;
```

A.NOMBRE	B.NOMBRE
AUDI	SEAT
BMW	RENAULT

La condición A.nombre > B.nombre se pone para que no se repitan las parejas como por ejemplo (AUDI, SEAT) y (SEAT, AUDI).

22.-Obtener las parejas de modelos de coches cuyo nombre es el mismo y cuya marca es de 'BILBAO'.

La manera de solucionar este problema es muy parecida al anterior, aunque con la dificultad añadida de que hay que manejar varias tablas. Lo primero, al igual que antes, será reunir la tabla COCHES consigo misma según el campo nombre. Una vez hecho esto, habrá que reunir este resultado con la tabla MARCO para ver cuál es la marca de cada coche, y el resultado global reunirlo con la tabla MARCAS para ver cual es la ciudad de cada marca. Una vez realizado esto sólo quedará seleccionar aquellas tuplas cuya marcas.ciudad sea igual a BILBAO y proyectar sobre los campos modelo.

```
SELECT A.modelo, B.modelo
FROM coches A,coches B, marco, marcas
WHERE A.nombre = B.nombre
AND A.modelo > B.modelo
AND A.codcoche = marco.codcoche
AND marco.cifm = marcas.cifm
AND marcas.ciudad= 'BILBAO';
```

A.MODELO	B.MODELO
CARAVAN	GTI

23.-Obtener todos los codcoche de los coches cuyo nombre empiece por 'C'.

```
SELECT codcoche
FROM coches
WHERE nombre LIKE 'C%';
```

CODCOCHE
005
017

24.-Obtener todos los codcoche de los coches cuyo nombre no contiene ninguna 'A'.

```
SELECT DISTINCT codcoche
FROM coches
WHERE nombre NOT LIKE '%A%';
```

CODCOCHE
004
010
011
018
019
020

25.-Obtener el número total de nombre de marcas de coches que son de MADRID.

```
SELECT COUNT(DISTINCT nombre)
FROM marcas
WHERE ciudad = 'MADRID';
```

COUNT (NOMBRE)
2

26.-Obtener la media de la cantidad de coches que tienen todos los concesionarios.

```
SELECT AVG (cantidad)
FROM distribucion;
```

AVG (CANTIDAD)
7.4

27.-Obtener el dni con numeración más alta de todos los clientes de 'MADRID'.

```
SELECT MAX(dni)
FROM clientes
WHERE ciudad='MADRID';
```

MAX (DNI)
0004

28.-Obtener el dni con numeración mas baja de todos los clientes que han comprado un coche 'BLANCO'.

```
SELECT MIN(dni)
FROM ventas
WHERE color='BLANCO'
```

MIN (DNI)
0001

29.-Obtener el cifc de todos los concesionarios cuyo número de coches en stock no es nulo.

```
SELECT DISTINCT cifc
FROM distribucion
WHERE cantidad IS NOT NULL;
```

CIFC
0001
0002
0003
0004
0005
0006

30.-Obtener el cifm y el nombre de las marcas de coches cuya segunda letra del nombre de la ciudad de origen sea una 'I'.

```
SELECT DISTINCT cifm, nombre
FROM marcas
WHERE ciudad LIKE '_I%';
```

CIFM	NOMBRE
0005	OPEL

31.-Obtener el dni de los clientes que han comprado algún coche a un concesionario de 'MADRID'.

```
SELECT DISTINCT dni
FROM ventas
WHERE cifc IN (SELECT cifc
                FROM concesionario
                WHERE ciudad = 'MADRID');
```

DNI
0001
0002
0003

32.-Obtener el color de los coches vendidos por el concesionario 'ACAR'.

Este, al igual que el problema anterior, puede resolverse tal y como se hizo en los problemas 14-22 es decir, realizando reuniones (productos naturales) entre las tablas.

```
SELECT DISTINCT color
FROM ventas, concesionario
WHERE concesionario.nombre = 'ACAR'
AND concesionario.cifc = ventas.cifc;
```

No obstante éste y los siguientes problemas se resolverán utilizando subconsultas.

```
SELECT DISTINCT color
FROM ventas
WHERE cifc IN (SELECT cifc
                FROM concesionario
                WHERE nombre = 'ACAR');
```

COLOR
BLANCO
ROJO

33.-Obtener el codcoche de los coches vendidos por algún concesionario de 'MADRID'.

```
SELECT DISTINCT codcoche
FROM ventas
WHERE cifc IN (SELECT cifc
                FROM concesionario
                WHERE ciudad = 'MADRID');
```

CODCOCHE
001
005
008
006

34.- Obtener el nombre y el modelo de los coches vendidos por algún concesionario de ‘BARCELONA’.

```
SELECT nombre, modelo
FROM coches
WHERE codcoche IN
    (SELECT codcoche
     FROM ventas
     WHERE cifc IN
         (SELECT cifc
          FROM concesionario
          WHERE ciudad = 'BARCELONA'));
```

NOMBRE	MODELO
ZX	TD

35.- Obtener todos los nombre de los clientes que hayan adquirido algún coche del concesionario ‘DCAR’.

```
SELECT nombre
FROM clientes
WHERE dni IN (SELECT dni
               FROM ventas
               WHERE cifc IN (SELECT cifc
                              FROM concesionario
                              WHERE nombre = 'DCAR'));
```

NOMBRE
JAVIER

36.- Obtener el NOMBRE y el APELLIDO de los clientes que han adquirido un coche modelo ‘GTI’ de color ‘BLANCO’.

```
SELECT nombre, apellido
FROM clientes
WHERE dni IN (SELECT dni
               FROM ventas
               WHERE color = 'BLANCO'
               AND codcoche IN (SELECT codcoche
                                FROM coches
                                WHERE modelo = 'GTI'));
```

NOMBRE	APELLIDO

Es decir, no existe ninguno.

37.- Obtener el nombre y el apellido de los clientes que han adquirido un automóvil a un concesionario que posea actualmente coches en stock del modelo ‘GTI’.

```
SELECT nombre, apellido
FROM clientes
WHERE dni IN
    (SELECT dni
     FROM ventas
     WHERE cifc IN
         (SELECT cifc
          FROM distribucion
          WHERE codcoche IN
              (SELECT codcoche
               FROM coches
               WHERE modelo = 'GTI')));
```

NOMBRE	APELLIDO
LUIS	GARCIA
ANTONIO	LOPEZ

38.- Obtener el nombre y el apellido de los clientes que han adquirido un automóvil a un concesionario de ‘MADRID’ que posea actualmente coches en stock del modelo ‘GTI’.

```
SELECT nombre, apellido
FROM clientes
WHERE dni IN
    (SELECT dni
     FROM ventas
     WHERE cifc IN
```

```
(SELECT cifc
  FROM concesionario
 WHERE ciudad = 'MADRID'
 AND cifc IN
   (SELECT cifc
    FROM distribucion
 WHERE codcoche IN
      (SELECT codcoche
       FROM coches
 WHERE modelo = 'GTI')));
```

NOMBRE	APELLIDO
LUIS	GARCIA

39.- Obtener el nombre y el apellido de los clientes cuyo dni es menor que el del cliente 'JUAN MARTIN'.

```
SELECT nombre, apellido
  FROM clientes
 WHERE dni < (SELECT dni
    FROM clientes
   WHERE nombre = 'JUAN'
 AND apellido = 'MARTIN');
```

NOMBRE	APELLIDO
LUIS	GARCIA
ANTONIO	LOPEZ

40.- Obtener el nombre y el apellido de los clientes cuyo dni es menor que el de los clientes que son de 'BARCELONA'.

```
SELECT nombre, apellido
  FROM clientes
 WHERE dni < ALL (SELECT dni
    FROM clientes
   WHERE ciudad = 'BARCELONA');
```

NOMBRE	APELLIDO
LUIS	GARCIA
ANTONIO	LOPEZ
JUAN	MARTIN
MARIA	GARCIA

41.- Obtener el nombre y el apellido de los clientes cuyo nombre empieza por 'A' y cuyo dni es mayor que el de los clientes que son de 'MADRID'.

```
SELECT nombre, apellido
  FROM clientes
 WHERE dni > ALL (SELECT dni
    FROM clientes
   WHERE ciudad = 'MADRID')
 AND nombre LIKE 'A%';
```

NOMBRE	APELLIDO
ANA	LOPEZ

42.- Obtener el nombre y el apellido de los clientes cuyo nombre empieza por 'A' y cuyo dni es mayor que el de ¡ALGUNO! de los clientes que son de 'MADRID'.

```
SELECT nombre, apellido
  FROM clientes
 WHERE dni > ANY (SELECT dni
    FROM clientes
   WHERE ciudad = 'MADRID')
 AND nombre LIKE 'A%';
```

NOMBRE	APELLIDO
ANTONIO	LOPEZ
ANA	LOPEZ

43.- Obtener el nombre y el apellido de los clientes cuyo nombre empieza por 'A' y cuyo dni es mayor que el de ¡ALGUNO! de los clientes que son de 'MADRID' o menor que el de todos los de 'VALENCIA'.

```
SELECT nombre, apellido
FROM clientes
WHERE (dni > ANY (SELECT dni
                   FROM clientes
                   WHERE ciudad = 'MADRID')
      OR dni < ALL (SELECT dni
                     FROM clientes
                     WHERE ciudad = 'VALENCIA'))
      AND nombre LIKE 'A%';
```

¡ Cuidado con los paréntesis !

NOMBRE	APELLIDO
ANTONIO	LOPEZ
ANA	LOPEZ

44.- Obtener el nombre y el apellido de los clientes que han comprado como mínimo un coche 'BLANCO' y un coche 'ROJO'.

Este problema, aunque a priori parece trivial, presenta una dificultad adicional; supóngase que se realiza la siguiente consulta:

```
SELECT nombre, apellido
FROM clientes
WHERE dni IN (SELECT dni
               FROM ventas
               WHERE color = 'BLANCO'
               AND color = 'ROJO');
```

El resultado de la misma sería el conjunto vacío ya que la variable *color* en cada tupla no puede valer a la vez BLANCO y ROJO. Una de las posibles soluciones es la siguiente:

```
SELECT nombre, apellido
FROM clientes
WHERE dni IN (SELECT dni
               FROM ventas
               WHERE color = 'BLANCO')
      AND dni IN (SELECT dni
                   FROM ventas
                   WHERE color = 'ROJO');
```

Esta sería una de las maneras correctas de realizar la consulta ya que así el operador SELECT seleccionará aquellos nombres y apellidos de aquellas tuplas cuyo atributo COLOR tenga un valor que esté contenido en los dos grupos seleccionados por los otros dos SELECT.

NOMBRE	APELLIDO
LUIS	GARCIA

45.- Obtener el dni de los clientes cuya ciudad sea la última de la lista alfabética de las ciudades donde hay concesionarios.

```
SELECT dni
FROM clientes
WHERE ciudad = (SELECT MAX(ciudad)
                 FROM concesionario);
```

DNI
0002

46.- Obtener la media de los automóviles que cada concesionario tiene actualmente en stock.

Para resolver este tipo de problemas hay que pensar en cómo se haría a mano. Para ello, sería necesario dividir la tabla en grupos, estando compuestos cada uno de ellos por tuplas con el mismo valor del atributo CIFC. Ahora en cada uno de esos grupos se calcularía la media aritmética de su atributo CANTIDAD. Todo eso es posible hacerlo en SQL utilizando las funciones agregadas y GROUP BY.

```
SELECT cifc, AVG(cantidad)
FROM distribucion
GROUP BY cifc;
```

CIFC	AVG(CANTIDAD)
0001	5.66
0002	8.33
0003	4.33
0004	7.5
0005	12.66
0006	3

47.- Obtener el cifc del concesionario que no sea de 'MADRID' cuya media de vehículos en stock sea la mas alta de todas las medias.

```
SELECT cifc
FROM concesionario
WHERE ciudad <> 'MADRID'
AND cifc IN (SELECT cifc
              FROM distribucion
              GROUP BY cifc
              HAVING AVG (cantidad) >= (SELECT AVG(cantidad)
                                              FROM distribucion
                                              GROUP BY cifc));
```

CIFC
0005

48.- Repetir el ejercicio 33 pero utilizando EXISTS en la solución.

```
SELECT DISTINCT codcoche
FROM ventas
WHERE EXISTS ( SELECT *
                FROM concesionario
                WHERE ciudad = 'MADRID'
                AND concesionario.cifc = ventas.cifc);
```

CODCOCHE
001
005
008
006

Como puede verse, en la solución anterior la formulación del problema es muy similar a la del problema 33, aunque conceptualmente es muy distinta ya que la manera de operar es la siguiente:

Para cada tupla de la tabla *ventas* se seleccionará su atributo *codcoche* si existe en la tabla *concesionario* alguna tupla cuyo campo *ciudad* tenga como valor MADRID y cuyo campo *cifc* sea igual al de la tupla de la tabla *ventas* en cuestión.

49.- Utilizando EXISTS obtener el dni de los clientes que hayan adquirido por lo menos alguno de los coches que ha sido vendido por el concesionario cuyo cifc es 0001.

```
SELECT DISTINCT va.dni
FROM ventas va
WHERE EXISTS (SELECT *
               FROM ventas vb
               WHERE va.codcoche = vb.codcoche
               AND vb.cifc = 0001);
```

Como antes, la explicación de este problema es la siguiente: para cada tupla de la tabla *va* (que es un alias de *ventas*) se seleccionará el atributo *dni*, si existe en la tabla *vb* (que es otro alias de ventas) alguna tupla cuyo atributo *cifc* tenga como valor 0001 y cuyo atributo *codcoche* sea el mismo que la tupla de la tabla *va* en cuestión. El motivo de crear los alias es que para resolver este problema es necesario realizar una reunión de la tabla *ventas* consigo misma (*va.codcoche = vb.codcoche*), de manera que si no fuese por los alias esta condición siempre sería cierta.

DNI
0001
0002

50.- Obtener los dni de los clientes que sólo han comprado coches al concesionario 0001.

Este tipo de problemas (“Obtener X de los que sólo...a Y) es mejor intentar reformularlos al revés para que la solución sea más sencilla, es decir, en base a negaciones en vez de afirmaciones. En este caso concreto el problema se podría plantear como: “*Obtener los DNI de los clientes que en la tabla Ventas NO TIENEN a su lado un valor de cifc DISTINTO de 0001*”.

Esta frase, es posible expresarla de una manera más cercana al SQL como:

“*Seleccionar los DNI de las tuplas de los CLIENTES de la tabla VENTAS, de manera que NO EXISTA para esa tupla en el atributo CIFC un valor DISTINTO del 0001*”.

Puesto esto en formato SQL la pregunta quedaría:

```
SELECT dni
  FROM ventas va
 WHERE NOT EXISTS (SELECT *
                      FROM ventas vb
                     WHERE cifc <> 0001
                       AND va.dni = vb.dni);
```

DNI
0002

51.- Obtener los nombres de los clientes que no han comprado ningún coche ‘ROJO’ a ningún concesionario de ‘MADRID’.

Este es otro problema que se resuelve de manera muy sencilla utilizando el comando NOT EXISTS. Para ello el problema se puede plantear como

“*Obtener el nombre de los clientes cuyo dni no aparezca en la tabla VENTAS al lado de un codcoche cuyo color sea ROJO y al lado de un cifc que sea de MADRID*”.

```
SELECT nombre
  FROM clientes
 WHERE NOT EXISTS (SELECT *
                      FROM ventas
                     WHERE ventas.dni = clientes.dni
                       AND color = 'ROJO'
                       AND cifc IN (SELECT cifc
                                     FROM concesionario
                                    WHERE ciudad = 'MADRID'))
```

NOMBRE
JUAN
MARIA
JAVIER
ANA

52.- Obtener el nombre de los clientes que sólo han comprado en el concesionario de cife 0001.

Este problema al igual que los anteriores es mucho mejor formularlo de manera inversa, es decir:

"Obtener el NOMBRE de los clientes cuyo DNI (que está en la tabla VENTAS) no aparece en la tabla VENTAS al lado de algún CIFC distinto de 0001"

```
SELECT nombre
FROM clientes
WHERE dni IN (SELECT dni
               FROM ventas ventas1
               WHERE NOT EXISTS (SELECT *
                                  FROM ventas ventas2
                                  WHERE ventas1.dni=ventas2.dni
                                  AND ventas2.cifc<>0001));
```

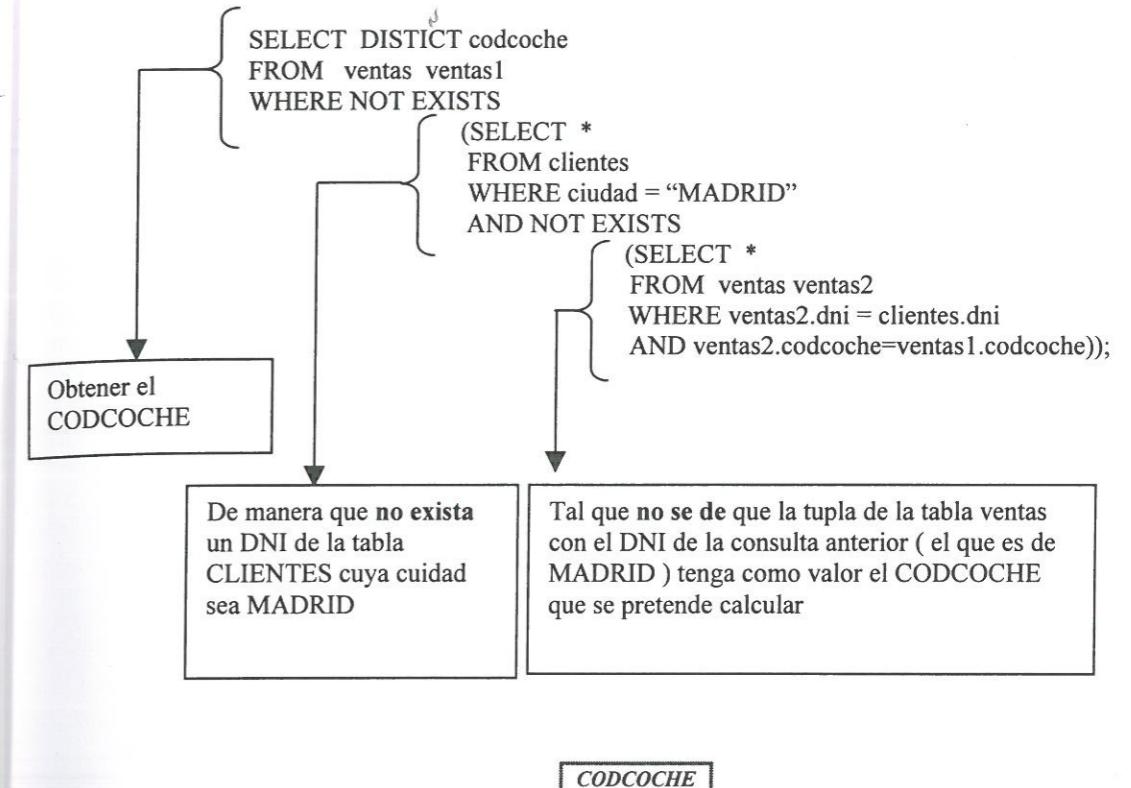
NOMBRE
ANTONIO

53.- Obtener el codcoche de aquellos automóviles que han sido comprados por todos los clientes de 'MADRID'.

Es decir, todos los clientes que son de MADRID deben haber comprado estos coches. En este problema al igual que en los anteriores, es mucho más factible plantearse la pregunta al revés, es decir:

"Obtener los CODCOCHE de los automóviles de manera que no exista un cliente que sea de MADRID que no esté en la tabla VENTAS a su lado (al lado del CODCOCHE)"

Tal y como puede deducirse del planteamiento anterior, el problema es fácilmente abordable utilizando dos negaciones en forma de NOT EXISTS.



54.- Obtener el codcoche de aquellos automóviles de color "ROJO" y de modloe "GTI" que han sido comprados por todos los clientes cuyo aopellido comienza por "G".

Este es un problema exactamente igual al anterior, pero con algún filtro más en cuanto a las condiciones que debe cumplir el coche y el cliente. Por tanto, el problema planteado al revés sería:

"Obtener los CODCOCHE de los automóviles ROJOS y GTI de manera que no exista un cliente cuyo apellido comienza por G que no esté en la tabla VENTAS a su lado (al lado del CODCOCHE)"

```

SELECT DISTINCT codcoche
FROM ventas ventas1
WHERE codcoche IN
    (SELECT codcoche
     FROM coches
     WHERE modelo = "GTI")
AND color = "ROJO"
AND NOT EXISTS
    (SELECT *
     FROM clientes
     WHERE apellido LIKE 'G%'
     AND NOT EXISTS
        (SELECT *
         FROM ventas ventas2
         WHERE ventas2.cifc = clientes.dni
         AND ventas2.codcoche=ventas1.codcoche));

```

CODCOCHE

55.- Obtener el dni de los clientes que han adquirido por lo menos los mismos automóviles que el cliente "LUIS GARCIA".

Este problema es del mismo tipo que los anteriores, por ello para su resolución es mucho mejor formularlo al revés, es decir mediante negaciones:

"Obtener el DNI de los clientes de manera que en la tabla ventas no exista el CODCOCHE de un automóvil adquirido por LUIS GARCIA que no aparezca en una tupla de la tabla ventas con este DNI.

```

SELECT DISTINCT DNI
FROM ventas ventas1
WHERE NOT EXISTS
    (SELECT codcoche
     FROM ventas ventas2
     WHERE dni IN
        (SELECT dni
         FROM clientes
         WHERE nombre = 'LUIS')

```

```

        AND apellido = 'GARCIA')
AND NOT EXISTS
    (SELECT *
     FROM ventas ventas3
     WHERE ventas3.codcoche = ventas2.codcoche
     AND ventas3.dni = ventas1.dni));

```

DNI

400

56.- Obtener el cifc de los concesionarios que han vendido el mismo coche a todos los clientes.

Este probablemente sea el ejercicio en SQL más complicado de los planteados hasta el momento. Al igual que en los problemas anteriores, es recomendable intentar resolverlo formulando las preguntas mediante negaciones. En este caso este problema podría plantearse de la siguiente manera:

"Obtener el CIFC del concesionario de manera que para algún CODCOCHE de la tabla ventas no exista ningún DNI de la tabla clientes, que no estén los tres en la misma tupla en la tabla ventas.

```

SELECT DISTINCT CIFC
FROM ventas V1
WHERE EXISTS
    (SELECT codcoche
     FROM ventas V2
     WHERE NOT EXISTS
        (SELECT dni
         FROM clientes
         WHERE NOT EXISTS
            (SELECT *
             FROM ventas V3
             WHERE V3.cifc=V1.cifc
             AND V3.codcoche=V2.codcoche
             AND V3.dni = clientes.dni)));

```

CIFC

```

SELECT DISTINCT codcoche
FROM ventas ventas1
WHERE codcoche IN
    (SELECT codcoche
     FROM coches
     WHERE modelo = "GTI")
AND color = "ROJO"
AND NOT EXISTS
    (SELECT *
     FROM clientes
     WHERE apellido LIKE 'G%'
     AND NOT EXISTS
        (SELECT *
         FROM ventas ventas2
         WHERE ventas2.cifc = clientes.dni
         AND ventas2.codcoche=ventas1.codcoche));

```

CODCOCHE

55.- Obtener el dni de los clientes que han adquirido por lo menos los mismos automóviles que el cliente "LUIS GARCIA".

Este problema es del mismo tipo que los anteriores, por ello para su resolución es mucho mejor formularlo al revés, es decir mediante negaciones:

"Obtener el DNI de los clientes de manera que en la tabla ventas no exista el CODCOCHE de un automóvil adquirido por LUIS GARCIA que no aparezca en una tupla de la tabla ventas con este DNI.

```

SELECT DISTINCT DNI
FROM ventas ventas1
WHERE NOT EXISTS
    (SELECT codcoche
     FROM ventas ventas2
     WHERE dni IN
        (SELECT dni
         FROM clientes
         WHERE nombre = 'LUIS')

```

```

        AND apellido = 'GARCIA')
AND NOT EXISTS
    (SELECT *
     FROM ventas ventas3
     WHERE ventas3.codcoche = ventas2.codcoche
     AND ventas3.dni = ventas1.dni));

```

DNI

400

56.- Obtener el cifc de los concesionarios que han vendido el mismo coche a todos los clientes.

Este probablemente sea el ejercicio en SQL más complicado de los planteados hasta el momento. Al igual que en los problemas anteriores, es recomendable intentar resolverlo formulando las preguntas mediante negaciones. En este caso este problema podría plantearse de la siguiente manera:

"Obtener el CIFC del concesionario de manera que para algún CODCOCHE de la tabla ventas no exista ningún DNI de la tabla clientes, que no estén los tres en la misma tupla en la tabla ventas.

```

SELECT DISTINCT CIFC
FROM ventas V1
WHERE EXISTS
    (SELECT codcoche
     FROM ventas V2
     WHERE NOT EXISTS
        (SELECT dni
         FROM clientes
         WHERE NOT EXISTS
            (SELECT *
             FROM ventas V3
             WHERE V3.cifc=V1.cifc
             AND V3.codcoche=V2.codcoche
             AND V3.dni = clientes.dni)));

```

CIFC

57.- Convertir la siguiente proposición de SQL a su equivalente en Castellano.

```
SELECT DISTINCT dni
FROM ventas V1
WHERE NOT EXISTS
    (SELECT *
     FROM ventas V2
     WHERE ventas2.dni=ventas1.dni
     AND NOT EXISTS
         (SELECT *
          FROM ventas V3
          WHERE V3.codcoche=V2.codcoche
          AND V3.cifc=0001));
```

“Obtener los DNI de los CLIENTES que hayan comprado sólo automóviles al concesionario de CIFC 0001”

Capítulo VI

QUERY-BY-EXAMPLE

1. INTRODUCCIÓN

Query-by-Example (QBE, consulta mediante ejemplos) es un lenguaje relacional de manipulación de datos, desarrollado por IBM, que forma parte, como una de las interfaces de usuario, del producto de consultas *Query Management Facility* (QMF). Existe una correspondencia muy cercana entre QBE y el *cálculo relacional de dominios*.

En el presente capítulo se proponen y resuelven problemas de QBE y se realiza un recordatorio de los aspectos teóricos más importantes en relación con los problemas.