

**11.2** Rellena la siguiente tabla, o bien construye el enunciado para la sentencia SQL o bien escribe la solución del enunciado:

ENUNCIADO	SOLUCIÓN
¿Qué día es hoy?	
¿En qué minuto estamos?	SELECT YEAR(CURDATE());
Concatena las cadenas 'hola' y 'adios'	
¿En qué año naciste?	SELECT ROUND((100+200+300)/3, 2);
Visualiza palabra 'Hola' añadiendo guiones a la derecha hasta formar 10 posiciones	SELECT RPAD('HOLA', 10, '-');
A partir de la cadena 'BLANCO Y NEGRO', devuelve la posición donde empieza la letra O	SELECT REPLACE('BLANCO Y NEGRO', 'O', 'A');
Obtener los nombres de alumnos de la tabla NOTAS_ALUMNOS que tengan alguna nota con valor nulo	SELECT CONCAT('Nombre:', Nombre_alumno) FROM notas_alumnos;
Obtener en una columna la concatenación de las columnas APELLIDOS y DIRECCION de la tabla ALUM2006 convertidas a mayúsculas.	SELECT * FROM notas_alumnos WHERE nota3 <> 7 AND nota1 <> 7 AND nota2 <> 7 AND asignatura = 'LENGUA';  SELECT * FROM libros WHERE YEAR(fecha_publicacion) NOT IN (2000, 2001);  SELECT CONCAT(nombre, CONCAT(apellidos, dirección)) FROM alum2006;  SELECT dni, IFNULL(fecha_nac, 'Fecha Nula') FROM alum2006;

# 12

# Consultas más complejas

## Introducción

En este capítulo seguiremos viendo la potencia de la orden SELECT. Hasta ahora se han realizado consultas sencillas que manejaban una tabla, pero también podemos realizar consultas más complejas para obtener datos de varias tablas, para obtener datos referentes a un grupo de filas utilizando funciones resumen; o incluso dentro de una orden SELECT podemos incluir otra orden SELECT para realizar subconsultas.

## Contenido

- 12.1. Consultas multitable**
- 12.2. Consultas resumen**
- 12.3. Consultas agrupadas**
- 12.4. Combinación externa**
- 12.5. Subconsultas**

### Ejercicios propuestos

## Objetivos

- Construir sentencias SQL usando varias tablas
- Realizar consultas combinando varias tablas de la base de datos
- Aplicar las diferentes funciones para obtener información de varias filas
- Realizar consultas agrupando filas
- Entender el concepto de combinación externa
- Aplicar la combinación externa para consultar dos tablas
- Realizar subconsultas a la base de datos
- Utilizar manuales en línea para obtener información adicional

## 12.1 Consultas multitabla

En las consultas que hemos realizado hasta ahora, sólo se ha utilizado una tabla, que se indicaba a la derecha de la cláusula **FROM**; pero hay veces que una consulta necesita columnas de varias tablas. En este caso, los nombres de las tablas se expresarán a la derecha de la palabra **FROM**. La sintaxis general es la siguiente:

```
SELECT columnas de las tablas citadas en la cláusula "FROM"
FROM tabla1, tabla2, ...
WHERE condición de combinación de las tablas;
```

### 12.1.1. Consideraciones para consultas multitabla

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

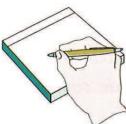
- Se pueden unir tantas tablas como deseemos.
- En la cláusula **SELECT** se pueden citar columnas de todas las tablas.
- Si hay columnas con el mismo nombre en las distintas tablas de la cláusula **FROM**, se deben identificar, especificando **NombreTabla.NombreColumna**.
- Si el nombre de una columna 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.



#### Actividad de Aplicación 12.1

Abre MySQL Query Browser y crea un nuevo esquema de base de datos de nombre UNIDAD12 para cargar el script que crea las tablas con las que trabajaremos en esta unidad.

A continuación carga el script TABLAS\_UNIDAD12.SQL que está en la página web del libro en el Editor de Scripts y ejecútalo.



#### Ejercicio Resuelto 12.1

Partimos de las tablas **EMPLEADOS** y **DEPARTAMENTOS**. Su descripción se muestra en las Figuras 12.1 y 12.2.

Field	Type	Null	Key	Default
emp_no	smallint(4) unsigned	NO		
apellido	varchar(10)	YES		
oficio	varchar(10)	YES		
dir	smallint(6)	YES		
fecha_alt	date	YES		
salario	float(6,2)	YES		
comision	float(6,2)	YES		
dept_no	tinyint(2)	NO		

Figura 12.1. Descripción de la tabla EMPLEADOS

Field	Type	Null	Key	Default
dept_no	tinyint(2)	NO		
dnombre	varchar(15)	YES		
loc	varchar(15)	YES		

Figura 12.2. Descripción de la tabla DEPARTAMENTOS

El contenido de las tablas se muestra en las Figuras 12.3 y 12.4.

emp_no	apellido	oficio	dir	fecha_alt	salario	comision	dept_no
7363	SÁNCHEZ	EMPLEADO	7902	1990-12-17	1040.00		20
7499	ARROYO	VENDEDOR	7698	1990-02-20	1500.00	390.00	30
7521	SALA	VENDEDOR	7698	1991-02-22	1625.00	650.00	30
7566	JIMÉNEZ	DIRECTOR	7833	1991-04-02	2800.00		20
7654	MARTÍN	VENDEDOR	7698	1991-09-29	1600.00	1020.00	30
7658	NEGRO	DIRECTOR	7833	1991-05-01	3005.00		30
7782	CEREZO	DIRECTOR	7833	1991-06-09	2885.00		10
7788	GIL	ANALISTA	7566	1991-11-09	3000.00		20
7833	REY	PRESIDENTE		1991-11-17	4100.00		10
7844	TOVAR	VENDEDOR	7698	1991-09-08	1350.00	0.00	30
7876	ALONSO	EMPLEADO	7788	1991-09-23	1430.00		20
7900	JIMENO	EMPLEADO	7698	1991-12-03	1335.00		30
7902	FERNÁNDEZ	ANALISTA	7566	1991-12-03	3000.00		20
7934	MUÑOZ	EMPLEADO	7782	1992-01-23	1690.00		10

Figura 12.3. Contenido de la tabla EMPLEADOS

dept_no	dnombre	loc
10	CONTABILIDAD	SEVILLA
20	INVESTIGACIÓN	MADRID
30	VENTAS	BARCELONA
40	PRODUCCIÓN	BILBAO

Figura 12.4. Contenido de la tabla DEPARTAMENTOS

La tabla **EMPLEADOS** contiene los datos de los empleados: el número del empleado, columna **EMP\_NO**; el **APELLIDO**, el **OFICIO**; el número del empleado jefe, columna **DIR**; la fecha de alta, columna **FECHA\_ALT**, el **SALARIO** del empleado, la **COMISION** y el número del departamento del empleado, columna **DEPT\_NO**.

La tabla **DEPARTAMENTOS** contiene los datos de los departamentos de los empleados: el número del departamento, columna **DEPT\_NO**; el nombre del departamento, columna **DNOMBRE** y la localidad donde está el departamento, columna **LOC**. Estas dos tablas se relacionan por la columna **DEPT\_NO**.

1. A partir de estas dos tablas obtenemos los siguientes datos de los empleados: **APELLIDO**, **OFICIO**, número de empleado (columna **EMP\_NO**), nombre de departamento (columna **DNOMBRE**) y localidad (columna **LOC**). Estas tablas tienen en común la columna **DEPT\_NO**, que es la columna por la que se combinan las tablas (véase la Figura 12.5):

```
SELECT apellido, oficio, emp_no, dnombre, loc
FROM empleados, departamentos
WHERE empleados.dept_no = departamentos.dept_no;
```

apellido	oficio	emp_no	dnombre	loc
SÁNCHEZ	EMPLEADO	7363	INVESTIGACIÓN	MADRID
ARROYO	VENDEDOR	7499	VENTAS	BARCELONA
SALA	VENDEDOR	7521	VENTAS	BARCELONA
JIMÉNEZ	DIRECTOR	7566	INVESTIGACIÓN	MADRID
MARTÍN	VENDEDOR	7654	VENTAS	BARCELONA
NEGRO	DIRECTOR	7658	VENTAS	BARCELONA
CEREZO	DIRECTOR	7782	CONTABILIDAD	SEVILLA
GIL	ANALISTA	7788	INVESTIGACIÓN	MADRID
REY	PRESIDENTE	7833	CONTABILIDAD	SEVILLA
TOVAR	VENDEDOR	7844	VENTAS	BARCELONA
ALONSO	EMPLEADO	7876	INVESTIGACIÓN	MADRID
JIMENO	EMPLEADO	7900	VENTAS	BARCELONA
FERNÁNDEZ	ANALISTA	7902	INVESTIGACIÓN	MADRID
MUÑOZ	EMPLEADO	7934	CONTABILIDAD	SEVILLA

Figura 12.5. Combinación de EMPLEADOS y DEPARTAMENTOS

En el ejemplo anterior, si no especificamos delante de la columna DEPT\_NO el nombre de la tabla, MySQL devuelve un mensaje error indicando que la columna es ambigua, ya que existe en ambas tablas, véase la Figura 12.6. Cuando ocurre esto, en la sentencia SQL se indicará de qué tabla se obtiene la columna.

Figura 12.6. Error de columnas ambiguas

En la combinación anterior, 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, resultaría 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 EMPLEADOS por 4 filas de la tabla DEPARTAMENTOS = 56 filas. Prueba la siguiente consulta y analiza el resultado:

```
SELECT apellido, oficio, emp_no, dnombre, loc
FROM empleados, departamentos;
```



### Actividad de Aplicación 12.2

A partir de las tablas EMPLEADOS Y DEPARTAMENTOS obtenemos las siguientes columnas: APELLIDO, OFICIO, DNOMBRE y columna DEPT\_NO. Como la columna DEPT\_NO está en ambas tablas, es necesario indicar de qué tabla se obtendrá; por tanto, delante de la columna se especificará el nombre de la tabla.

¿Qué obtiene esta consulta?

```
SELECT apellido, oficio, dnombre
FROM empleados, departamentos
WHERE empleados.dept_no = departamentos.dept_no
AND empleados.dept_no IN (10,20);
```

También se puede combinar una tabla consigo misma. En este caso es necesario usar alias de tablas y de columnas. El siguiente ejemplo muestra el apellido de los directores y su número de empleado:

```
SELECT DISTINCT e1.apellido , e1.emp_no
FROM empleados e1, empleados e2 WHERE e1.emp_no = e2.dir;
```

### Ejercicio Resuelto 12.2

Partimos de tres tablas: ALUMNOS, ASIGNATURAS y NOTAS. La descripción de las tablas se muestra en las Figuras 12.7, 12.8 y 12.9.

La tabla ALUMNOS contiene datos de los alumnos: el NIF, apellidos y nombre (columna APENOM), dirección, población y teléfono.

La tabla ASIGNATURAS contiene datos de las asignaturas en que pueden estar matriculados los alumnos: el código de la asignatura (columna COD) y el nombre.

La tabla NOTAS contiene las notas de los alumnos en las asignaturas: el NIF del alumno, el código de la asignatura y la nota.

Las relaciones de las tablas se muestran en la Figura 12.10. La tabla NOTAS se relaciona con la tabla ALUMNOS por la columna NIF; y con la tabla ASIGNATURAS por la columna COD.

Field	Type	Null	Key	Default
nif	varchar(10)	NO		
apenom	varchar(30)	YES	HULL	
direc	varchar(30)	YES	HULL	
pobla	varchar(15)	YES	HULL	
telef	varchar(10)	YES	HULL	

Figura 12.7. Tabla ALUMNOS

Field	Type	Null	Key	Default
cod	tinyint(3) unsigned	NO		
nombre	varchar(25)	YES	HULL	

Figura 12.8. Tabla ASIGNATURAS

Field	Type	Null	Key	Default
nif	varchar(10)	NO		
cod	tinyint(3) unsigned	NO		
nota	float(3,1)	YES	HULL	

Figura 12.9. Tabla NOTAS

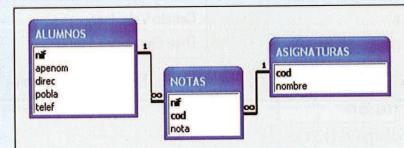


Figura 12.10. Tablas ALUMNOS, ASIGNATURAS, NOTAS

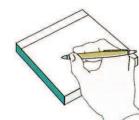
- Realizamos una consulta para obtener el nombre de alumno (columna APENOM) que está en la tabla ALUMNOS, el nombre de la asignatura del alumno (columna NOMBRE) que está en la tabla ASIGNATURAS y la nota (columna NOTA) que está en la tabla NOTAS. El resultado se muestra en la Figura 12.11.

Las columnas de la sentencia SELECT son las siguientes: APENOM, NOMBRE, NOTA.

Ahora bien, como los datos se obtienen de tres tablas, es necesario especificar en la cláusula FROM las tres tablas: FROM alumnos, asignaturas, notas.

Por último, cuando se combinan tablas es necesario especificar en la cláusula WHERE la condición de combinación; en este ejemplo la tabla NOTAS se relaciona con la tabla ALUMNOS por la columna NIF, lo que origina una condición: alumnos.nif = notas.nif. Y también se relaciona con la tabla ASIGNATURAS por la columna COD, lo que origina otra condición: asignaturas.cod = notas.cod. Por lo tanto, la cláusula WHERE quedaría así: WHERE alumnos.nif = notas.nif AND asignaturas.cod = notas.cod. La sentencia SELECT es la siguiente:

```
SELECT apenom, nombre, nota
FROM alumnos, asignaturas, notas
WHERE alumnos.nif = notas.nif AND asignaturas.cod = notas.cod;
```



apenom	nombre	nota
Alcalde García, Elena	Prog. Leng. Estr.	6.0
Alcalde García, Elena	Sist. Informáticos	5.0
Alcalde García, Elena	Análisis	6.0
Cerrato Vela, Luis	FOL	6.0
Cerrato Vela, Luis	RET	8.0
Cerrato Vela, Luis	Entornos Gráficos	4.0
Cerrato Vela, Luis	Aplic. Entornos 4ºGen	5.0
Díaz Fernández, María	FOL	8.0
Díaz Fernández, María	RET	7.0
Díaz Fernández, María	Entornos Gráficos	8.0
Díaz Fernández, María	Aplic. Entornos 4ºGen	9.0

Figura 12.11. Combinación de 3 tablas

2. Obtenemos la consulta anterior, pero ahora, sólo aquellas filas cuyo nombre de asignatura (columna NOMBRE) es "FOL".

```
SELECT apenom, nombre, nota
FROM alumnos, asignaturas, notas
WHERE alumnos.nif = notas.nif AND asignaturas.cod = notas.cod
AND nombre = "FOL";
```

apenom	nombre	nota
Cerrato Vela, Luis	FOL	6.0
Díaz Fernández, María	FOL	8.0

Figura 12.12. Alumnos matriculados en FOL



### Actividad de Aplicación 12.3

A partir de la última consulta, obtener sólo los alumnos que tengan un 8 en la asignatura de nombre "FOL".

## 12.2 Consultas resumen

SQL proporciona un conjunto de funciones que nos permiten resumir datos de la base de datos. Estas funciones nos permitirán calcular, por ejemplo: el salario medio de los empleados, el número de empleados que hay en un departamento, el salario máximo y mínimo, etc. Actúan sobre un grupo de filas para obtener un valor. Estas funciones se muestran en la Tabla 1.

Función	Propósito
AVG(n)	Calcula el valor medio de "n" ignorando los valores nulos.
COUNT (*   expresión)	Cuenta el número de veces que la expresión evalúa algún dato con valor no nulo. La opción "*" cuenta todas las filas seleccionadas.
MAX(expresión)	Calcula el máximo valor de la "expresión".
MIN(expresión)	Calcula el mínimo valor de la "expresión".
SUM(expresión)	Obtiene la suma de valores de la "expresión" distintos de nulos.

Tabla 1. Funciones de grupos de valores

Los valores nulos son ignorados por las funciones de grupos de valores, y los cálculos se realizan sin contar con ellos. El argumento de estas funciones suele ser un nombre de columna.

### Ejercicio Resuelto 12.3

Probamos las funciones de grupos de valores con la tabla EMPLEADOS.

1. Cálculo del salario medio de los empleados:

```
SELECT AVG(salario) "Salario Medio" FROM empleados;
```

2. Cálculo del salario medio de los empleados del departamento 10:

```
SELECT AVG(salario) FROM empleados WHERE dept_no = 10;
```

3. Cálculo del número de filas de la tabla:

```
SELECT COUNT(*) "Filas" FROM empleados;
```

4. Cálculo del número de filas de la tabla donde la *comisión* no es nula:

```
SELECT COUNT(comision) FROM empleados;
```

5. Cálculo del máximo salario:

```
SELECT MAX(salario) FROM empleados;
```

6. Obtener el apellido mínimo (alfabéticamente) de los empleados del departamento 20:

```
SELECT MIN(apellido) FROM empleados WHERE dept_no = 20;
```

7. Cálculo de la suma de salarios de todos los empleados:

```
SELECT SUM(salario) FROM empleados;
```

Figura 12.13. Consulta 1

Salario Medio
2175.714286

Figura 12.14. Consulta 2

AVG(salario)
2891.666667

Figura 12.15. Consulta 3

Filas
14

Figura 12.16. Consulta 4

COUNT(comision)
4

Figura 12.17. Consulta 5

MAX(salario)
4100.00

Figura 12.18. Consulta 6

MIN(apellido)
ALONSO

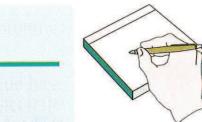
Figura 12.19. Consulta 7



### Actividad de Aplicación 12.4

¿Cuántos empleados hay en el departamento 30?

¿Cuál es la suma de los salarios de los empleados del departamento 30?



realiza una selección de filas cuyos valores en la columna especificada no estén duplicados. La cláusula ALL recoge todas las filas aunque sus valores estén duplicados.

El formato de COUNT, incluyendo DISTINCT y ALL es el siguiente:

```
COUNT ( * | [DISTINCT | ALL] expresión)
```

Si COUNT recibe como argumento una expresión o columna, ésta podrá ir precedida de las cláusulas ALL o DISTINCT.



### Actividad de Aplicación 12.5

Ejecuta las dos sentencias SQL siguientes y analiza los resultados de su ejecución. ¿Qué obtiene cada una?

```
SELECT COUNT(oficio) "OFICIOS" FROM empleados;
SELECT COUNT(DISTINCT oficio) "OFICIOS" FROM empleados;
```

## 12.3 Consultas agrupadas

Las consultas descritas en el apartado anterior obtenían un único dato resumen como si fuese un total final. Pero, a veces, nos interesa consultar los datos según grupos determinados para calcular subtotales. Así, por ejemplo, para saber cuál es el salario medio de cada departamento de la tabla EMPLEADOS, las cláusulas que conocemos hasta ahora no son suficientes, porque necesitamos realizar un agrupamiento por departamento.

Para realizar agrupamientos usaremos la cláusula **GROUP BY**, que sirve para calcular propiedades de uno o más conjuntos de filas. Así, para saber cuál es el salario medio de cada departamento de la tabla EMPLEADOS es necesario agrupar por la columna DEPT\_NO. La columna o columnas por las que se quiere realizar el agrupamiento se indicarán a la derecha de GROUP BY. La consulta anterior se escribiría (el resultado se muestra en la Figura 12.20):

```
SELECT dept_no, AVG(salario) FROM empleados GROUP BY dept_no;
```

Los datos seleccionados en la sentencia SELECT que lleva el GROUP BY deben ser: una constante, una función de grupo (SUM, COUNT, AVG, ...) o una columna expresada en el GROUP BY.

Del mismo modo que existe la condición de búsqueda WHERE para filas individuales, también hay una condición de búsqueda para grupos de filas. La cláusula **HAVING** se emplea para controlar cuál de los conjuntos de filas se visualiza. Se evalúa sobre las filas que devuelve GROUP BY. No puede existir sin GROUP BY.

El orden en que aparecen las cláusulas SELECT vistas hasta ahora es el siguiente:

```
SELECT ...
FROM ...
WHERE ...
GROUP BY column1, column2, column3, ...
HAVING condición
ORDER BY columnas;
```

dept_no	AVG(salario)
10	2891.666667
20	2274.000000
30	1735.833333

Figura 12.20. Salario medio por departamento

### Ejercicio Resuelto 12.4

A partir de la tabla EMPLEADOS realizamos las siguientes consultas:

- Obtener el número de empleados que hay en cada departamento visualizando la columna DEPT\_NO y el número de empleados.

Para hacer esta consulta, hay que agrupar las filas por la columna DEPT\_NO, ya que hay que obtener algún dato referente a cada departamento. Por tanto, a la derecha de GROUP BY incluimos dicha columna.

Para calcular el número de empleados se contarán las filas usando la función COUNT(\*). El resultado es el siguiente:

```
SELECT dept_no, COUNT(*) FROM empleados
GROUP BY dept_no;
```

dept_no	COUNT(*)
10	3
20	5
30	6

Figura 12.21. Consulta 1. Ejercicio Resuelto 12.4

- Partimos de la consulta anterior. Obtenemos aquellos departamentos cuyo número de empleados es mayor que 3.

En este caso es necesario controlar qué grupo de filas se va a visualizar: aquél cuyo número de empleados sea mayor que 3. Por tanto, es necesario incluir la cláusula HAVING que comprobará si el número de empleados, calculado con la función COUNT(\*), es mayor que 3. El resultado es el siguiente:

```
SELECT dept_no, COUNT(*) FROM empleados
GROUP BY dept_no HAVING COUNT(*) > 3;
```

dept_no	COUNT(*)
20	5
30	6

Figura 12.22. Consulta 2. Ejercicio Resuelto 12.4

- Calculamos la consulta anterior ordenando descendenteamente por número de empleados:

```
SELECT dept_no, COUNT(*) FROM empleados
GROUP BY dept_no HAVING COUNT(*) > 3
ORDER BY COUNT(*) DESC;
```

dept_no	COUNT(*)
30	6
20	5

Figura 12.23. Consulta 3. Ejercicio Resuelto 12.4

- Obtenemos el máximo salario que hay en cada departamento visualizando la columna DEPT\_NO y el máximo salario.

```
SELECT dept_no, MAX(salario)
FROM empleados
GROUP BY dept_no;
```

dept_no	MAX(salario)
10	4100.00
20	3000.00
30	3005.00

Figura 12.24. Consulta 4. Ejercicio Resuelto 12.4

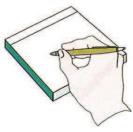


### Actividad de Aplicación 12.6

Obtener el máximo apellido (alfabéticamente) que hay en cada departamento visualizando las columnas DEPT\_NO y máximo apellido, ordenando descendenteamente por apellido.

¿Qué obtiene esta consulta?:

```
SELECT dept_no, MIN(apellido)
FROM empleados GROUP BY dept_no;
```



## Ejercicio Resuelto 12.5

A partir de las tablas EMPLEADOS y DEPARTAMENTOS realizamos las siguientes consultas con agrupamientos y combinación de tablas.

- Obtener el número de empleados que hay en cada departamento visualizando las columnas: número de departamento, nombre de departamento y número de empleados. En esta consulta es necesario combinar las tablas ya que el nombre de departamento (columna DNOMBRE) no existe en la tabla EMPLEADOS. Por tanto, se aplica combinación de tablas y agrupamientos:

```
SELECT empleados.dept_no, dnombre, COUNT(*)
FROM empleados, departamentos
WHERE empleados.dept_no = departamentos.dept_no
GROUP BY empleados.dept_no, dnombre;
```

dept_no	dnombre	COUNT(*)
10	CONTABILIDAD	3
20	INVESTIGACIÓN	5
30	VENTAS	6

Figura 12.25. Consulta 1. Ejercicio Resuelto 12.5

MySQL ha extendido el uso de GROUP BY permitiendo utilizar columnas o cálculos en las expresiones SELECT que no aparecen en la parte GROUP BY; con esto, se pretende evitar ordenaciones y agrupaciones innecesarias. La cláusula GROUP BY anterior se puede escribir así:

```
GROUP BY empleados.dept_no;
```

- Obtener la media de salarios por cada departamento visualizando las columnas: nombre de departamento y media de salarios:

```
SELECT dnombre, AVG(salario)
FROM empleados, departamentos
WHERE empleados.dept_no = departamentos.dept_no
GROUP BY dnombre;
```

dnombre	AVG(salario)
CONTABILIDAD	2891.666667
INVESTIGACIÓN	2274.000000
VENTAS	1735.833333

Figura 12.26. Consulta 2. Ejercicio Resuelto 12.5

## 12.4 Combinación externa

La combinación de tablas vista anteriormente se conoce como combinación interna; en ésta, se combina información procedente de dos tablas formando pares de filas relacionadas según la condición de búsqueda. Existe una variedad de combinación de tablas que se denomina combinación externa o JOIN EXTERNO, y permite seleccionar algunas filas de una tabla aunque éstas no tengan correspondencia con las filas de la otra tabla con la que se combina.

La mejor forma de entender la combinación externa es mediante ejemplos.

## Ejercicio Resuelto 12.6

Partimos de las tablas EMPLEADOS y DEPARTAMENTOS cuyo contenido se muestra en las Figuras 12.3 y 12.4. Si nos fijamos en la tabla EMPLEADOS, vemos que el departamento 40 no existe; sin embargo, en DEPARTAMENTOS sí existe una fila con los datos para ese departamento.

- Queremos realizar una consulta donde se visualice el número de departamento, el nombre y el número de empleados que tiene. Para hacer esta consulta combinamos las dos tablas. El resultado se muestra en la Figura 12.27:

```
SELECT d.dept_no, dnombre, COUNT(e.emp_no)
FROM empleados e, departamentos d
WHERE e.dept_no = d.dept_no
GROUP BY d.dept_no, dnombre;
```

dept_no	dnombre	COUNT(e.emp_no)
10	CONTABILIDAD	3
20	INVESTIGACIÓN	5
30	VENTAS	6

Figura 12.27. Consulta 1. Ejercicio Resuelto 12.6

Observamos que con esta sentencia sólo aparecen los departamentos que tienen empleados. El departamento 40 se pierde. Para que aparezca este departamento que no existe en ninguna fila de la tabla EMPLEADOS, usamos la combinación externa.

La combinación externa se puede expresar de dos formas:

- Combinación externa a la izquierda: **LEFT JOIN**. En esta combinación, por cada fila de la primera tabla (colocada a la izquierda de LEFT JOIN) que no se haya combinado con ninguna fila de la segunda tabla (colocada a la derecha de LEFT JOIN), se añade una fila al resultado utilizando los valores de las columnas de la primera tabla y suponiendo un valor NULL para todas las columnas de la segunda tabla correspondientes en el emparejamiento.

En el ejemplo anterior, para que aparezca una fila con los datos del departamento 40, que no existe en la tabla EMPLEADOS, se escribirá la tabla DEPARTAMENTOS a la izquierda de la cláusula LEFT JOIN. La consulta es la siguiente:

```
SELECT d.dept_no, dnombre, COUNT(e.emp_no)
FROM departamentos d LEFT JOIN empleados e
ON e.dept_no = d.dept_no
GROUP BY d.dept_no, dnombre;
```

El resultado de la combinación externa se muestra en la Figura 12.28.

dept_no	dnombre	COUNT(e.emp...)
10	CONTABILIDAD	3
20	INVESTIGACIÓN	5
30	VENTAS	6
40	PRODUCCIÓN	0

Figura 12.28. Consulta 2. Ejercicio Resuelto 12.5

3. Combinación externa a la derecha: **RIGHT JOIN**. En esta combinación, por cada fila de la segunda tabla (colocada a la derecha de RIGHT JOIN) que no se haya combinado con ninguna fila de la primera tabla (colocada a la izquierda de RIGHT JOIN), se añade una fila al resultado utilizando los valores de las columnas de la segunda tabla y suponiendo un valor NULL para todas las columnas de la primera tabla correspondientes en el emparejamiento.

En este caso, para que aparezca una fila con los datos del departamento 40, se escribirá la tabla DEPARTAMENTOS a la derecha de la cláusula RIGHT JOIN:

```
SELECT d.dept_no, dnombre, COUNT(e.emp_no)
FROM empleados e RIGHT JOIN departamentos d
ON e.dept_no = d.dept_no
GROUP BY d.dept_no, dnombre;
```

El resultado de esta combinación es idéntico al mostrado en la Figura 12.28.



### Actividad de Aplicación 12.7

Prueba las siguientes sentencias SQL y analiza los resultados:

```
SELECT e.dept_no, dnombre, COUNT(e.emp_no)
FROM departamentos d LEFT JOIN empleados e
ON e.dept_no = d.dept_no
GROUP BY e.dept_no, dnombre;

SELECT e.dept_no, dnombre, COUNT(e.emp_no)
FROM empleados e RIGHT JOIN departamentos d
ON d.dept_no = e.dept_no
GROUP BY e.dept_no, dnombre;
```

## 12.5 Subconsultas

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 'GIL', para comenzar haremos de averiguar el oficio de 'GIL' (primera consulta, consulta más interna). Una vez conocido este dato, podemos averiguar qué empleados tienen el mismo oficio que 'GIL' (segunda consulta, consulta más externa).

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 (o HAVING) 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 al siguiente:

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

La subconsulta (el comando SELECT entre paréntesis o consulta más interna) se ejecutará primero y, posteriormente, el valor extraído es "introducido" en la consulta principal o consulta más externa. Por ejemplo, a partir de la tabla EMPLEADOS, obtenemos 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'. En la Figura 12.29 se ve el resultado de la consulta: el oficio es ANALISTA:

```
SELECT oficio FROM empleados WHERE apellido = 'GIL';
```

A continuación, visualizamos el APELLIDO de los empleados con el mismo OFICIO que 'GIL', en la Figura 12.30 se ve el resultado de la consulta:

```
SELECT apellido FROM empleados WHERE oficio='ANALISTA';
```

Resumiendo las dos consultas anteriores en una única consulta obtendremos la subconsulta; en la cláusula WHERE de esta consulta se incluirá la sentencia SELECT que averigua el oficio de 'GIL':

```
SELECT apellido FROM empleados WHERE oficio=
(SELECT oficio FROM empleados WHERE apellido = 'GIL');
```

oficio
ANALISTA

Figura 12.29. Oficio de GIL

apellido
GIL
FERNÁNDEZ

Figura 12.30. Apellidos con el mismo oficio que GIL



### Actividad de Aplicación 12.8

A partir de la tabla EMPLEADOS, obtener el *APELLIDO*, *OFICIO* y *SALARIO* de los empleados con el mismo *SALARIO* que 'SALA'.

#### 12.5.1. Subconsultas que generan valores simples

Se trata de aquellas subconsultas que devuelven una fila o un valor simple; en estas subconsultas se utilizan los operadores de comparación ( $>$ ,  $<$ ,  $\neq$ ,  $\leq$ ,  $\geq$ ,  $=$ ). Si la subconsulta obtiene más de una fila, se produce un mensaje de error, véase la Figura 12.31. Por ejemplo, con la siguiente consulta se pretende obtener los apellidos de los empleados cuyo OFICIO coincide con algún OFICIO del departamento 20:

```
SELECT apellido FROM empleados WHERE oficio =
(SELECT oficio FROM empleados WHERE dept_no = 20);
```

! Description	ErrorNr.
Subquery returns more than 1 row	1242

Figura 12.31. Error: la subconsulta devuelve varias filas

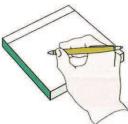
En el departamento 20 hay varios oficios; por tanto, la subconsulta devuelve varias filas. Por ello, cuando una subconsulta devuelve más de una fila, no se puede recurrir a los operadores de comparación.

#### 12.5.2. 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 (o NOT IN) en la cláusula WHERE. La solución al ejemplo anterior sería la siguiente:

```
SELECT apellido FROM empleados WHERE oficio IN
(SELECT oficio FROM empleados 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.



## Ejercicio Resuelto 12.7

Usamos las tablas EMPLEADOS y DEPARTAMENTOS.

1. Obtener el APELLIDO, el SALARIO y el departamento (columna DEPT\_NO) de los empleados con el mismo departamento que 'ARROYO' y con el salario menor que el salario de 'ARROYO'. Para hacer la consulta se compara el departamento de los empleados con el departamento de 'ARROYO' y el salario de los empleados con el salario de 'ARROYO'. La Figura 12.32 muestra el resultado:

```
SELECT apellido, salario, dept_no FROM empleados
WHERE dept_no =
      (SELECT dept_no FROM empleados WHERE apellido = 'ARROYO')
AND salario <
      (SELECT salario FROM empleados WHERE apellido = 'ARROYO');
```

apellido	salario	dept_no
TOVAR	1350.00	30
JIMENO	1335.00	30

Figura 12.32. Consulta 1. Ejercicio Resuelto 12.7

2. Obtener los datos de los empleados que trabajen en el departamento de 'VENTAS'. El nombre de departamento se obtiene de la tabla DEPARTAMENTOS; por tanto, hemos de relacionar las tablas EMPLEADOS y DEPARTAMENTOS por el número de departamento. Para ello, descomponemos esa consulta en varias: primero tenemos que localizar el número de departamento cuyo nombre (columna DNOMBRE) sea 'VENTAS'. La Figura 12.33 muestra el departamento:

dept_no
30

Figura 12.33. Consulta 2, parte 1. Ejercicio Resuelto 12.7

```
SELECT dept_no FROM departamentos WHERE dnombre = 'VENTAS';
```

A continuación, seleccionamos los datos de los empleados que estén en ese departamento:

```
SELECT * FROM empleados WHERE dept_no = 30;
```

Por último, reunimos estas dos sentencias SELECT utilizando una subconsulta. La Figura 12.34 muestra el resultado:

```
SELECT * FROM empleados WHERE dept_no =
      (SELECT dept_no FROM departamentos WHERE dnombre = 'VENTAS');
```

emp_no	apellido	oficio	dir	fecha_alt	salario	comision	dept_no
7493	ARROYO	VENDEDOR	7698	1990-02-20	1500.00	390.00	30
7521	SALA	VENDEDOR	7698	1991-02-22	1625.00	650.00	30
7654	MARTÍN	VENDEDOR	7698	1991-09-29	1600.00	1020.00	30
7698	NEGRO	DIRECTOR	7639	1991-05-01	3005.00	NULL	30
7844	TOVAR	VENDEDOR	7638	1991-09-08	1350.00	0.00	30
7900	JIMENO	EMPLEADO	7698	1991-12-03	1335.00	NULL	30

Figura 12.34. Consulta 2 completa. Ejercicio Resuelto 12.7

3. Obtener los datos de los empleados (APELLIDO, OFICIO, SALARIO, DEPT\_NO) que trabajen en los departamentos de 'VENTAS' o 'CONTABILIDAD'; la Figura 12.35 muestra el resultado. En este caso se usa el operador IN porque la subconsulta devuelve más de un departamento:

```
SELECT apellido, oficio, salario, dept_no FROM empleados
WHERE dept_no IN (SELECT dept_no FROM departamentos
WHERE dnombre IN ('VENTAS', 'CONTABILIDAD'));
```

apellido	oficio	salario	dept_no
ARROYO	VENDEDOR	1500.00	30
SALA	VENDEDOR	1625.00	30
MARTÍN	VENDEDOR	1600.00	30
NEGRO	DIRECTOR	3005.00	30
CEREZO	DIRECTOR	2885.00	10
REY	PRESIDENTE	4100.00	10
TOVAR	VENDEDOR	1350.00	30
JIMENO	EMPLEADO	1335.00	30
MUÑOZ	EMPLEADO	1690.00	10

Figura 12.35. Consulta 3. Ejercicio Resuelto 12.7

4. Obtener los datos de los empleados (APELLIDO, OFICIO, SALARIO, DEPT\_NO) cuyo salario supere la media de salarios de todos los empleados. El resultado se muestra en la Figura 12.36:

```
SELECT apellido, oficio, salario, dept_no FROM empleados
WHERE salario > (SELECT AVG(salario) FROM empleados);
```

apellido	oficio	salario	dept_no
JIMÉNEZ	DIRECTOR	2900.00	20
NEGRO	DIRECTOR	3005.00	30
CEREZO	DIRECTOR	2885.00	10
GIL	ANALISTA	3000.00	20
REY	PRESIDENTE	4100.00	10
FERNÁNDEZ	ANALISTA	3000.00	20

Figura 12.36. Consulta 4. Ejercicio Resuelto 12.7

5. Obtener los datos de los empleados (APELLIDO, OFICIO, SALARIO, DEPT\_NO) cuyo apellido sea alfabéticamente el más pequeño. El resultado se muestra en la Figura 12.37:

```
SELECT apellido, oficio, salario, dept_no FROM empleados
WHERE apellido = (SELECT MIN(apellido) FROM empleados);
```

apellido	oficio	salario	dept_no
ALONSO	EMPLEADO	1430.00	20

Figura 12.37. Consulta 5. Ejercicio Resuelto 12.7

6. Obtener los datos de los departamentos que NO tengan empleados. El resultado se muestra en la Figura 12.38:

```
SELECT * FROM departamentos WHERE dept_no NOT IN
      (SELECT DISTINCT dept_no FROM empleados);
```

dept_no	dnombre	loc
40	PRODUCCIÓN	BILBAO

Figura 12.38. Consulta 6. Ejercicio Resuelto 12.7

7. Obtener los datos de los departamentos que tengan empleados. El resultado se muestra en la Figura 12.39:

```
SELECT * FROM departamentos WHERE dept_no IN
(SELECT DISTINCT dept_no FROM empleados);
```

dept_no	dnombre	loc
10	CONTABILIDAD	SEVILLA
20	INVESTIGACIÓN	MADRID
30	VENTAS	BARCELONA

Figura 12.39. Consulta 7. Ejercicio Resuelto 12.7

8. Obtener los datos de los empleados (APELLIDO, OFICIO, SALARIO, DEPT\_NO) que trabajen en el departamento de ‘CONTABILIDAD’. Se puede realizar esta consulta mediante subconsultas o mediante combinación de tablas. El resultado se muestra en la Figura 12.40:

```
SELECT apellido, oficio, salario, dept_no FROM empleados
WHERE dept_no =
(SELECT dept_no FROM departamentos WHERE dnombre = 'CONTABILIDAD');
SELECT apellido, oficio, salario, e.dept_no
FROM empleados e, departamentos d
WHERE e.dept_no = d.dept_no
AND dnombre = 'CONTABILIDAD';
```

apellido	oficio	salario	dept_no
CEREZO	DIRECTOR	2885.00	10
REY	PRESIDENTE	4100.00	10
MUÑOZ	EMPLEADO	1690.00	10

Figura 12.40. Consulta 8. Ejercicio Resuelto 12.7



### Actividad de Aplicación 12.9

- Obtener el apellido, el salario y el número de departamento de los empleados cuyo salario sea mayor que el máximo salario del departamento 20.
- Igual que la consulta anterior, pero además de visualizar el apellido, el salario y el número de departamento, se ha de visualizar el nombre del departamento (columna DNOMBRE); para ello, será necesario realizar también una combinación de tablas (EMPLEADOS y DEPARTAMENTOS).



### Direcciones Web de interés:

Zona de descarga de documentación sobre MySQL:  
<http://dev.mysql.com/doc/>

Manual de referencia de MySQL en castellano:  
<http://dev.mysql.com/doc/refman/5.0/es/index.html>

Fuente de consulta y referencia para el aprendizaje de MySQL: artículos, noticias, eventos, etc.  
<http://www.mysql-hispano.org/>

Tutorial básico de MySQL:  
[http://www.programacion.com/bbdd/tutorial/mysql\\_basico/](http://www.programacion.com/bbdd/tutorial/mysql_basico/)



### Tablas EMPLEADOS y DEPARTAMENTOS

- 12.1 Seleccionar las columnas APELLIDO, OFICIO y nombre de departamento (columna DNOMBRE) de aquellos empleados cuyo oficio sea “ANALISTA”. La salida se muestra en la Figura 12.41.

apellido	oficio	dnombre
GIL	ANALISTA	INVESTIGACIÓN
FERNÁNDEZ	ANALISTA	INVESTIGACIÓN

Figura 12.41. Ejercicio propuesto 12.1

- 12.2 Seleccionar las columnas APELLIDO, OFICIO, nombre de departamento (columna DNOMBRE) y localidad del departamento (columna LOC) de aquellos empleados cuyo oficio no sea ni “ANALISTA” ni “DIRECTOR”. Mostrar los resultados ordenados por la columna APELLIDO. La salida se muestra en la Figura 12.40.

apellido	oficio	dnombre	loc
ALONSO	EMPLEADO	INVESTIGACIÓN	MADRID
ARROYO	VENDEDOR	VENTAS	BARCELONA
JIMENO	EMPLEADO	VENTAS	BARCELONA
MARTÍN	VENDEDOR	VENTAS	BARCELONA
MUÑOZ	EMPLEADO	CONTABILIDAD	SEVILLA
REY	PRESIDENTE	CONTABILIDAD	SEVILLA
SALA	VENDEDOR	VENTAS	BARCELONA
SÁNCHEZ	EMPLEADO	INVESTIGACIÓN	MADRID
TOVAR	VENDEDOR	VENTAS	BARCELONA

Figura 12.42. Ejercicio propuesto 12.2

- 12.3 Obtener el máximo salario que hay en cada departamento visualizando las columnas DEPT\_NO, máximo salario y nombre del departamento (columna DNOMBRE). La salida se muestra en la Figura 12.43.

dept_no	max(salario)	dnombre
10	4100.00	CONTABILIDAD
20	3000.00	INVESTIGACIÓN
30	3005.00	VENTAS

Figura 12.43. Ejercicio propuesto 12.3

- 12.4** Obtener la suma de salarios que hay en cada departamento visualizando las columnas DEPT\_NO, suma de salarios y nombre del departamento (columna DNAME). Si el departamento no tiene empleados, visualizar como suma de salarios el valor 0; utilizar combinación externa y la función IFNULL. La salida se muestra en la Figura 12.44.

dept_no	Suma Salario	dnombre
10	8675.00	CONTABILIDAD
20	11370.00	INVESTIGACIÓN
30	10415.00	VENTAS
40	0.00	PRODUCCIÓN

Figura 12.44. Ejercicio propuesto 12.4

#### Tablas ALUMNOS, ASIGNATURAS y NOTAS

- 12.5** Visualizar los nombres de alumnos (columna APENOM) de “Madrid” que tengan en alguna asignatura una nota menor que 5. La salida se muestra en la Figura 12.45.

apenom
Cerato Vela, Luis

Figura 12.45. Ejercicio propuesto 12.5

- 12.6** Visualizar los nombres de asignaturas (columna NOMBRE) que contengan tres letras “o” en su interior y tengan alumnos matriculados de “Madrid”. La salida se muestra en la Figura 12.46.

nombre
Entornos Gráficos

Figura 12.46. Ejercicio propuesto 12.6

- 12.7** Obtener los datos de las asignaturas que no tengan alumnos. La salida se muestra en la Figura 12.47.

cod	nombre
8	Idioma

Figura 12.47. Ejercicio propuesto 12.7

- 12.8** Obtener el nombre y apellido de los alumnos (columna APENOM) que no tengan nota en la asignatura con código 1. La salida se muestra en la Figura 12.48.

apenom
Cerato Vela, Luis
Díaz Fernández, María

Figura 12.48. Ejercicio propuesto 12.8

- 12.9** Obtener el nombre y apellido de los alumnos (columna APENOM) y la nota de aquellos alumnos que tengan la máxima nota en alguna asignatura. La salida se muestra en la Figura 12.49.

apenom	nota
Díaz Fernández, María	9.0

Figura 12.49. Ejercicio propuesto 12.9

**Tablas EDITORIALES y LIBROS.** Recordemos que se relacionan por la columna EDITORIAL. Las columnas de cada tabla son:

TABLA EDITORIALES:		TABLA LIBROS:	
editorial	INTEGER (2)	codigo	INTEGER (4)
nombre	VARCHAR (30)	titulo	VARCHAR (40)
localidad	VARCHAR (15)	autor	VARCHAR (50)
		editorial	INTEGER (2)
		unidades_vendidas	INTEGER (5)
		fecha_publicacion	DATE

- 12.10** Obtener las columnas CODIGO, TITULO y AUTOR de los libros de la editorial “PARANINFO”. La salida se muestra en la Figura 12.50.

codigo	titulo	autor
2232	SERVICIOS DE INFORMACIÓN ELECTRÓNICA	M.GONZÁLEZ, R.CHAMORRO y otros
4897	RIEDES DE ÁREA LOCAL	J.M.HUIDOBRO y otros
5044	MANTO DE PORTALES DE INFORMACIÓN	E.QUERO, A.GARCÍA y otros
7254	WINDOWS 2000 PROF.	A.GONZÁLEZ MANGAS
8609	INSTALACIONES ELÉCTRICAS	J.L.SANZ SERRANO
9732	SQL PARA USUARIOS Y PROGRAMADORES	E.RIVERO, L.MARTÍNEZ, y otros

Figura 12.50. Ejercicio propuesto 12.10

- 12.11** Obtener por cada editorial los libros que tiene, visualizando las columnas EDITORIAL, NOMBRE y número de libros. Si la editorial no tiene libros se mostrará 0. La salida se muestra en la Figura 12.51.

editorial	nombre	Num Libros
10	PARANINFO	6
20	RAMA	4
30	MCGRAW-HILL	5
40	DONOSTIARRA	0

Figura 12.51. Ejercicio propuesto 12.11

- 12.12** Obtener los datos de la editorial que no tiene libros. La salida se muestra en la Figura 12.52.

editorial	nombre	localidad
40	DONOSTIARRA	SAN SEBASTIÁN

Figura 12.52. Ejercicio propuesto 12.12

- 12.13** ¿Qué obtiene la siguiente consulta?

```
SELECT e.editorial, nombre, titulo
FROM   libros l, editoriales e ORDER BY 1;
```