

BBDD Actividad 5

SENTENCIAS SQL

AVANZADAS Y

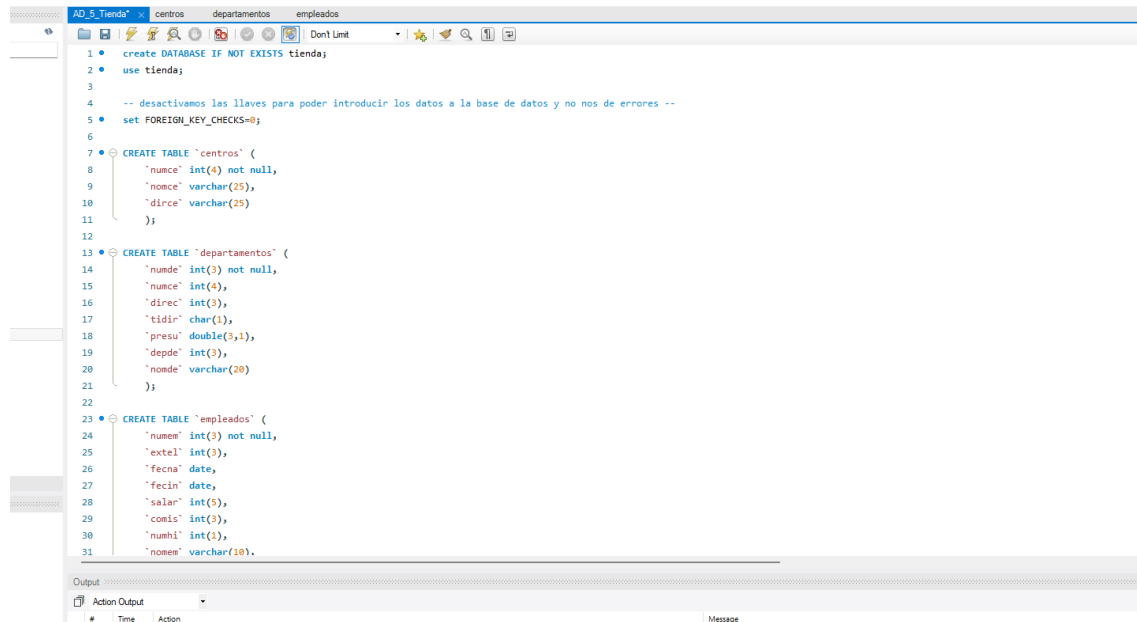
VISTAS

- **GRUPO 13:**

- Gerard Perujo Buxeda
- María Isabel Martín Simal
- Noelia Villahermosa García

Creación Base de Datos: Gerard Perujo

- 1- Creamos la base de datos de Tienda. Al insertar el primer registro si no se desactivan las llaves da error, ya que no se pueden introducir datos con las llaves activadas. Es necesario desactivarlas antes de introducir datos.



```
1 create DATABASE IF NOT EXISTS tienda;
2 use tienda;
3
4 -- desactivamos las llaves para poder introducir los datos a la base de datos y no nos de errores --
5 set FOREIGN_KEY_CHECKS=0;
6
7 CREATE TABLE 'centros' (
8     'numce' int(4) not null,
9     'nomce' varchar(25),
10    'dirce' varchar(25)
11 );
12
13 CREATE TABLE 'departamentos' (
14     'numde' int(3) not null,
15     'numce' int(4),
16     'direc' int(3),
17     'tidir' char(1),
18     'presu' double(3,1),
19     'depde' int(3),
20     'nomde' varchar(20)
21 );
22
23 CREATE TABLE 'empleados' (
24     'numem' int(3) not null,
25     'extel' int(3),
26     'fecna' date,
27     'fecin' date,
28     'salan' int(5),
29     'comis' int(3),
30     'numhi' int(1),
31     'nomem' varchar(10);
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
```

```
34
35 -- una vez creadas las tablas insertamos los indices ---
36
37 -- tabla centros--
38
39 ALTER TABLE 'centros'
40     add primary key ('numce');
41
42 -- tabla departamentos--
43
44 ALTER TABLE 'departamentos'
45     add primary key ('numde'),
46     add key 'numce' ('numce'),
47     add key 'numce' ('numce'),
48     add key 'depde' ('depde');
49
50 -- tabla empleados--
51
52 ALTER TABLE 'empleados'
53     add primary key ('numem'),
54     add key 'numem' ('numem'),
55     add key 'numde' ('numde');
56
57 -- ahora creamos los filtros para las diferentes tablas --
58
59 ALTER TABLE 'departamentos'
60     add constraint 'departamentos_ibfk1' foreign key ('numce') references 'centros' ('numce') on delete no action on update no action,
61     add constraint 'departamentos_ibfk2' foreign key ('depde') references 'departamentos' ('numde') on delete no action on update no action;
62
63 ALTER TABLE 'empleados'
64     add constraint 'empleados_ibfk1' foreign key ('numde') references 'departamentos' ('numde') on delete no action on update no action;
```

```
67 -- ahora ya empezariamos a cargar los datos --
68
69 -- Insertamos los campos de departamentos --
70
71 • insert into departamentos (numde,numce,direc,tidir,presu,depde,nomde)
72 values (100, 10, 260, 'P', 72, NULL, 'DIRECCIÓN GENERAL');
73
74 • insert into departamentos (numde,numce,direc,tidir,presu,depde,nomde)
75 values (110, 20, 180, 'P', 90, 100, 'DIRECC_COMERCIAL');
76
77 • insert into departamentos (numde,numce,direc,tidir,presu,depde,nomde)
78 values (111, 20, 180, 'F', 66, 110, 'SECTOR INDUSTRIAL');
79
80 • insert into departamentos (numde,numce,direc,tidir,presu,depde,nomde)
81 values (112, 20, 270, 'P', 54, 110, 'SECTOR SERVICIOS');
82
83 • insert into departamentos (numde,numce,direc,tidir,presu,depde,nomde)
84 values (120, 10, 150, 'F', 18, 100, 'ORGANIZACIÓN');
85
86 • insert into departamentos (numde,numce,direc,tidir,presu,depde,nomde)
87 values (121, 10, 150, 'P', 12, 120, 'PERSONAL');
88
89 • insert into departamentos (numde,numce,direc,tidir,presu,depde,nomde)
90 values (122, 10, 350, 'P', 36, 120, 'PROCESO DE DATOS');
91
92 • insert into departamentos (numde,numce,direc,tidir,presu,depde,nomde)
93 values (130, 10, 310, 'P', 12, 100, 'FINANZAS');
94
95 -- Insertamos los campos de centros --
96
97 • insert into centros (numce,nomce,dirce)
```

```
97 • insert into centros (numce,nomce,dirce)
98 values (10, 'cede central', 'C/ Atocha, 820, MADRID');
99
100 • insert into centros (numce,nomce,dirce)
101 values (20, 'relación con clientes', 'C/ Atocha, 405, MADRID');
102
103 -- insertamos los campos de empleados utilizando el bulk insert --
104
105 • insert into empleados (numen, extel, fecna, fecin, salar, comis, numhi, nomem, numde)
106 values (110, 350, '1978-11-10', '1985-02-15', 1800, NULL, 3, 'CESAR', 121),
107 (120, 840, '1968-06-09', '1988-10-01', 1900, 110, 1, 'MARIO', 112),
108 (130, 810, '1695-09-09', '1981-02-01', 1500, 110, 2, 'LUCIANO', 112),
109 (150, 340, '1975-08-10', '1997-05-15', 2600, NULL, 0, 'JULIO', 121),
110 (160, 740, '1980-07-09', '2005-11-11', 1800, 110, 2, 'AUREO', 111),
111 (180, 500, '1974-10-18', '1996-03-18', 2800, 50, 2, 'MARCOS', 120),
112 (190, 350, '1972-05-12', '1992-02-11', 1750, NULL, 4, 'JULIANA', 121),
113 (210, 200, '1970-09-28', '1999-01-22', 1910, NULL, 2, 'PILAR', 100),
114 (240, 760, '1967-02-26', '1989-02-24', 1700, 100, 3, 'LAVINIA', 111),
115 (250, 250, '1976-10-27', '1997-03-01', 2700, NULL, 0, 'ADRIANA', 100),
116 (260, 220, '1973-12-03', '2001-07-12', 720, NULL, 6, 'ANTONIO', 100),
117 (270, 800, '1975-05-21', '2003-09-10', 1910, 80, 3, 'OCTAVIO', 112),
118 (280, 410, '1978-01-10', '2010-10-08', 1500, NULL, 5, 'DOROTEA', 130),
119 (285, 620, '1979-10-25', '2011-02-15', 1910, NULL, 0, 'OTILIA', 122),
120 (290, 910, '1967-11-30', '1988-02-14', 1790, NULL, 3, 'GLORIA', 120),
121 (310, 480, '1976-11-21', '2011-01-15', 1950, NULL, 0, 'AUGUSTO', 130),
122 (320, 620, '1977-12-25', '2003-02-05', 2400, NULL, 2, 'CORNELIO', 122),
123 (330, 850, '1958-08-19', '1980-03-01', 1700, 90, 0, 'AMELIA', 112),
124 (350, 610, '1979-04-13', '1999-09-10', 2700, NULL, 1, 'AURELIO', 122),
125 (360, 750, '1978-10-29', '1998-10-10', 1800, 100, 2, 'DORINDA', 111),
126 (370, 360, '1977-06-22', '2000-01-20', 1860, NULL, 1, 'FABIOLA', 121),
127 (380, 800, '1978-03-30', '1999-01-01', 110, NULL, 0, 'MICAELA', 112).
```

Output

Action Output

The screenshot shows a SQL editor with a dark theme. The main window displays a large SQL INSERT statement with 139 lines of code. The statement inserts data into a table, with values for various columns including names and IDs. The data is organized into groups of 10 rows, with the last group containing 9 rows. The statement is as follows:

```
120 (290, 910, '1967-11-30', '1988-02-14', 1790, NULL, 3, 'GLORIA', 120),
121 (310, 480, '1976-11-21', '2011-01-15', 1950, NULL, 0, 'AUGUSTO', 130),
122 (320, 620, '1977-12-25', '2003-02-05', 2400, NULL, 2, 'CORNELIO', 122),
123 (330, 850, '1958-08-19', '1980-03-01', 1700, 90, 0, 'AMELIA', 112),
124 (350, 610, '1979-04-13', '1999-09-10', 2700, NULL, 1, 'AURELIO', 122),
125 (360, 750, '1978-10-29', '1998-10-10', 1000, 100, 2, 'DORINDA', 111),
126 (370, 360, '1977-06-22', '2000-01-20', 1000, NULL, 1, 'FABIOLA', 121),
127 (380, 880, '1978-03-30', '1999-01-01', 110, NULL, 0, 'MICHAELA', 112),
128 (390, 500, '1976-02-19', '2010-10-08', 1290, NULL, 1, 'CARPEN', 110),
129 (400, 780, '1979-08-18', '2011-11-01', 1150, NULL, 0, 'LUCRECIA', 111),
130 (410, 660, '1968-07-14', '1989-10-13', 1010, NULL, 0, 'AZUCENA', 122),
131 (420, 450, '1966-10-22', '1988-11-19', 2400, NULL, 0, 'CLAUDIA', 130),
132 (430, 650, '1967-10-26', '1988-11-19', 1260, NULL, 1, 'VALERIANA', 122),
133 (440, 760, '1966-09-26', '1986-02-28', 1260, 100, 0, 'LIVIA', 111),
134 (450, 800, '1966-10-21', '1986-02-28', 1260, 100, 0, 'SABINA', 112),
135 (480, 760, '1965-04-04', '1986-02-28', 1260, 100, 1, 'DIANA', 111),
136 (490, 880, '1964-06-06', '1988-01-01', 1090, 100, 0, 'HORACIA', 112),
137 (500, 750, '1965-10-08', '1987-01-01', 1200, 100, 0, 'HONORIA', 111),
138 (510, 550, '1966-05-04', '1986-11-01', 1200, NULL, 1, 'ROMULO', 110),
139 (550, 780, '1970-01-10', '1998-01-21', 600, 120, 0, 'SANCHO', 111);
```

Below the main window, there is a comment and two SQL statements:

```
140
141
142 -- Al finalizar la carga de todos los datos volvemos activar las llaves --
143 set FOREIGN_KEY_CHECKS=1;
144 commit;
```

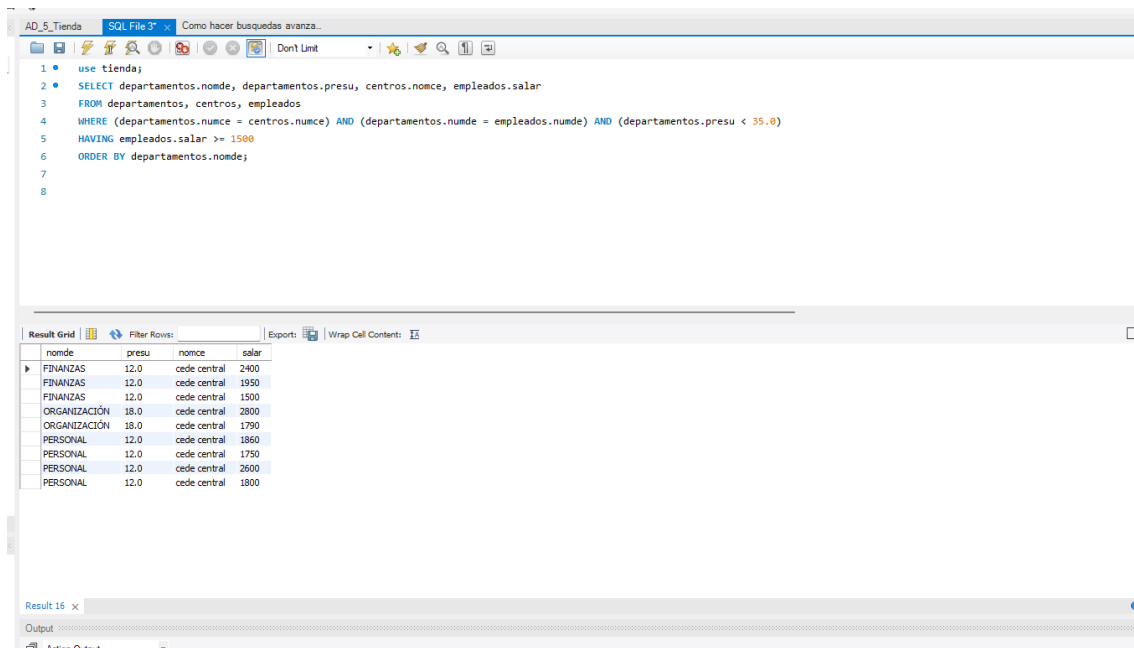
The right sidebar shows a tab labeled "My Snippets". The bottom status bar indicates "Output" and "Action Output".

SENTENCIAS

Queries 1 al 6: Gerard Perujo

1. Para cada departamento con presupuesto inferior a 35.000 €, hallar el nombre del Centro donde está ubicado y el máximo salario de sus empleados (si dicho máximo excede de 1.500 €). Clasificar

```
SELECT departamentos.nomde, departamentos.presu, centros.nomce, empleados.salar
FROM departamentos, centros, empleados
WHERE (departamentos.numce = centros.numce) AND (departamentos.numde =
empleados.numde) AND (departamentos.presu < 35.0)
HAVING empleados.salar >= 1500
ORDER BY departamentos.nomde;
```



The screenshot shows a SQL IDE window with a query editor and a results grid. The query editor contains the following SQL code:

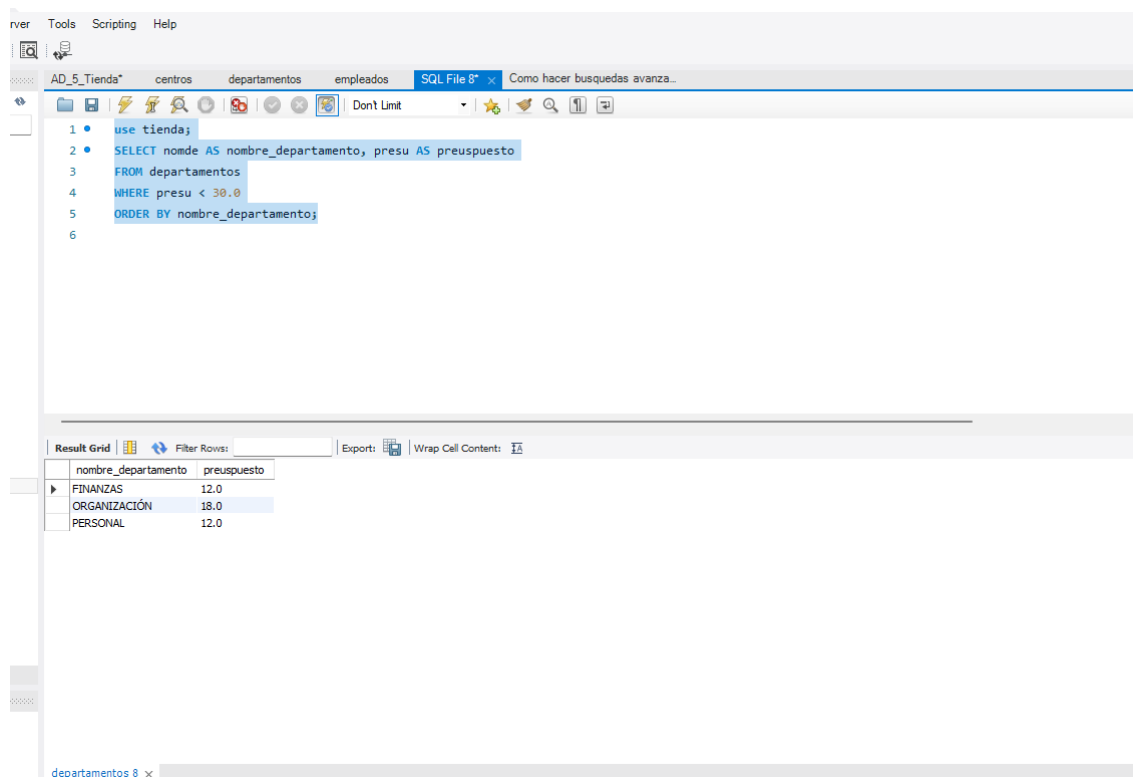
```
1 use tienda;
2 SELECT departamentos.nomde, departamentos.presu, centros.nomce, empleados.salar
3 FROM departamentos, centros, empleados
4 WHERE (departamentos.numce = centros.numce) AND (departamentos.numde = empleados.numde) AND (departamentos.presu < 35.0)
5 HAVING empleados.salar >= 1500
6 ORDER BY departamentos.nomde;
7
8
```

The results grid displays the following data:

nomde	presu	nomce	salar
FINANZAS	12.0	cede central	2400
FINANZAS	12.0	cede central	1950
FINANZAS	12.0	cede central	1500
ORGANIZACIÓN	18.0	cede central	2800
ORGANIZACIÓN	18.0	cede central	1790
PERSONAL	12.0	cede central	1860
PERSONAL	12.0	cede central	1750
PERSONAL	12.0	cede central	2600
PERSONAL	12.0	cede central	1800

- 2- Hallar por orden alfabético los nombres de los departamentos que dependen de los que tienen un presupuesto inferior a 30.000 €. También queremos conocer el nombre del departamento del que dependen y su presupuesto.

```
SELECT nomde AS nombre_departamento, presu AS preuspuesto
FROM departamentos
WHERE presu < 30.0
ORDER BY nombre_departamento;
```



The screenshot shows a SQL IDE interface. The top menu bar includes 'Server', 'Tools', 'Scripting', and 'Help'. Below the menu is a toolbar with various icons. The main editor area displays a SQL script with the following lines:

```
1 use tienda;
2 SELECT nomde AS nombre_departamento, presu AS preuspuesto
3 FROM departamentos
4 WHERE presu < 30.0
5 ORDER BY nombre_departamento;
6
```

Below the editor is a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The results are displayed in a table with two columns: 'nombre_departamento' and 'presupuesto'.

nombre_departamento	presupuesto
FINANZAS	12.0
ORGANIZACIÓN	18.0
PERSONAL	12.0

The status bar at the bottom shows 'departamentos 8'.

- 3- Obtener los nombres y los salarios medios de los departamentos cuyo salario medio supera al salario medio de la empresa.

```
SELECT departamentos.nomde, AVG(empleados.salar)
FROM empleados
LEFT JOIN departamentos ON empleados.numde = departamentos.numde
GROUP BY departamentos.nomde
HAVING AVG(empleados.salar) > (SELECT AVG(empleados.salar)
                                FROM empleados);
```

The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1 SELECT departamentos.nomde, AVG(empleados.salar)
2 FROM empleados
3 LEFT JOIN departamentos ON empleados.numde = departamentos.numde
4 GROUP BY departamentos.nomde
5 HAVING AVG(empleados.salar) > (SELECT AVG(empleados.salar)
6                                FROM empleados);
7
8
9
```

The results grid displays the following data:

nomde	AVG(empleados.salar)
DIRECCIÓN GENERAL	1776.6667
FINANZAS	1950.0000
ORGANIZACIÓN	2295.0000
PERSONAL	2002.5000
PROCESO_DE_DATOS	1856.0000

The bottom of the screenshot shows the 'Output' pane with the following message:

```
251 15:25:35 SELECT AVG(salar) FROM empleados WHERE salar = 300
Message
1 row(s) returned
```

- 4- Para los departamentos cuyo director lo sea en funciones, hallar el número de empleados y la suma de sus salarios, comisiones y número de hijos.

```
SELECT departamentos.nomde, count(empleados.numem) as numero_empleados,  
Sum(empleados.salar) as suma_salario, SUM(empleados.comis) as suma_comision,  
empleados.numhi  
FROM departamentos, empleados  
WHERE departamentos.numde = empleados.numde AND departamentos.tidir = 'F'  
GROUP BY departamentos.nomde;
```

The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1  
2 use tienda;  
3 SELECT DISTINCT count(empleados.numem)AS cantidad_empleados, count(empleados.numhi) AS Cantidad_hijos,  
4 SUM(empleados.salar)AS suma_salarios, SUM(empleados.comis)AS suma_comisiones, departamentos.nomde AS Nombre_departamento  
5 FROM empleados  
6 left join departamentos on departamentos.numde = empleados.numde  
7 WHERE departamentos.tidir = 'F'  
8 GROUP BY departamentos.nomde;
```

The results grid displays the following data:

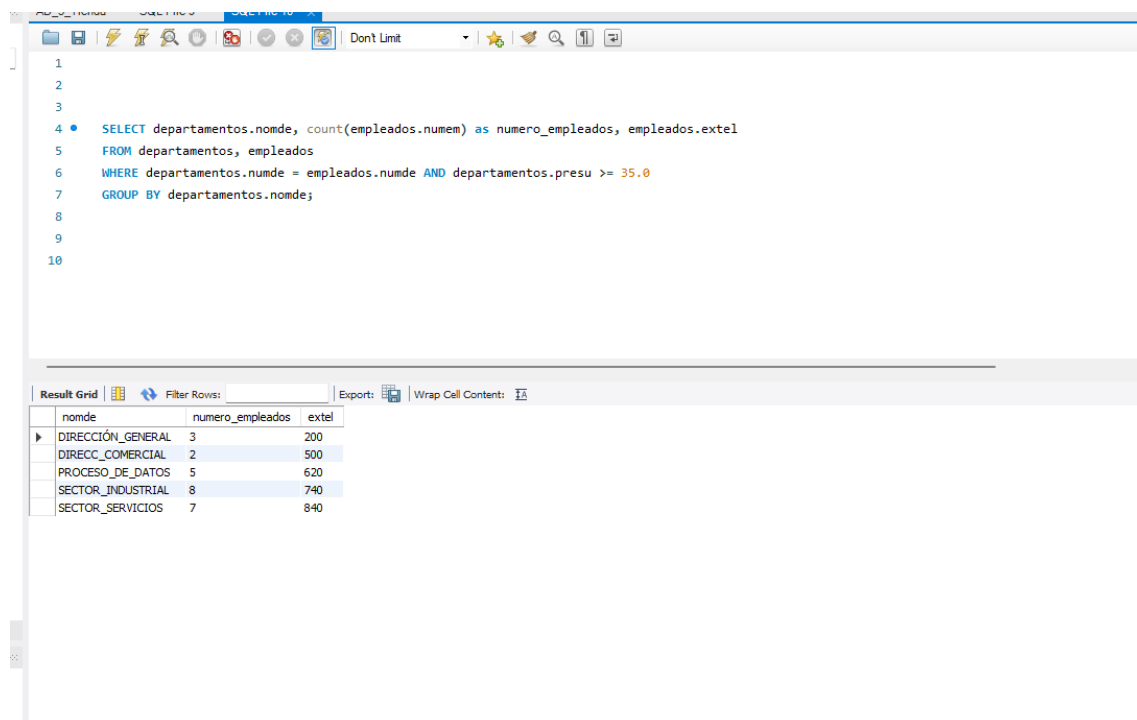
cantidad_empleados	Cantidad_hijos	suma_salarios	suma_comisiones	Nombre_departamento
2	2	4590	50	ORGANIZACIÓN
8	8	10770	730	SECTOR_INDUSTRIAL

The output pane shows the execution log with the following messages:

```
112 13:04:07 use tienda 0 row(s) affected  
113 13:04:07 SELECT DISTINCT count(empleados.numem)AS cantidad_empleados, departamentos.direc AS numero_director, count(empleados.numhi) AS C... 1 row(s) returned  
114 13:12:04 use tienda 0 row(s) affected  
115 13:12:04 SELECT DISTINCT count(empleados.numem)AS cantidad_empleados, count(empleados.numhi) AS Cantidad_hijos, SUM(empleados.salar)AS s... 1 row(s) returned  
116 13:13:00 SELECT DISTINCT count(empleados.numem)AS cantidad_empleados, count(empleados.numhi) AS Cantidad_hijos, SUM(empleados.salar)AS s... 2 row(s) returned
```


- 5- Para los departamentos cuyo presupuesto anual supera los 35.000 €, hallar cuantos empleados hay por cada extensión telefónica.

```
SELECT departamentos.nomde, count(empleados.numem) as  
numero_empleados, empleados.extel  
FROM departamentos, empleados  
WHERE departamentos.numde = empleados.numde AND  
departamentos.presu >= 35.0  
GROUP BY departamentos.nomde;
```



1
2
3
4 • SELECT departamentos.nomde, count(empleados.numem) as numero_empleados, empleados.extel
5 FROM departamentos, empleados
6 WHERE departamentos.numde = empleados.numde AND departamentos.presu >= 35.0
7 GROUP BY departamentos.nomde;
8
9
10

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [1/1](#)

nomde	numero_empleados	extel
DIRECCIÓN_GENERAL	3	200
DIRECC_COMERCIAL	2	500
PROCESO_DE_DATOS	5	620
SECTOR_INDUSTRIAL	8	740
SECTOR_SERVICIOS	7	840

- 6- Hallar por orden alfabético los nombres de los empleados y su número de hijos para aquellos que son directores en funciones.

```
SELECT nomem, numhi
FROM empleados
WHERE numde in (SELECT numde
                FROM departamentos
                WHERE tidir ='F')
GROUP BY nomem asc;
```

The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1
2
3 • use tienda;
4 • SELECT nomem, numhi
5   FROM empleados
6  WHERE numde in (SELECT NUMDE
7                  FROM departamentos
8                  WHERE tidir ='F')
9  GROUP BY nomem asc;
10
11
12
```

The results grid displays the following data:

nomem	numhi
AUREO	2
DIANA	1
DORINDA	2
GLORIA	3
HONORIA	0
LAVINIA	3
LIVIA	0
LUCRECIA	0
MARCOS	2
SANCHO	0

The interface also shows a toolbar with various icons, a filter row, and an export button. The bottom panel shows the output of the query, which is the same data as the results grid.

QUERIES 7 AL 14: Noelia Villahermosa

7. Hallar si hay algún departamento (suponemos que sería de reciente creación) que aún no tenga empleados asignados ni director en propiedad.

```
SELECT departamentos.nomde AS NOMBRE_DEPARTAMENTO, empleados.nomem AS  
NOMBRE_EMPL, departamentos.tidir AS TIPO_DIRECTOR  
FROM departamentos  
LEFT JOIN empleados ON departamentos.numde = empleados.numde  
WHERE empleados.numde IS NULL AND departamentos.tidir <> 'F';
```

The screenshot shows a SQL IDE interface with a query editor and a result grid. The query editor contains the following SQL code:

```
1 • USE tienda;  
2  
3 -- Hallar si hay algún departamento que aún no tenga empleados asignados ni director en propiedad.  
4  
5 • SELECT departamentos.nomde AS NOMBRE_DEPARTAMENTO, empleados.nomem AS NOMBRE_EMPL, departamentos.tidir AS TIPO_DIRECTOR  
6 FROM departamentos  
7 LEFT JOIN empleados ON departamentos.numde = empleados.numde  
8 WHERE empleados.numde IS NULL AND departamentos.tidir <> 'F';  
9  
10  
11
```

The result grid is displayed below the query editor, showing the following columns:

NOMBRE_DEPARTAMENTO	NOMBRE_EMPL	TIPO_DIRECTOR
---------------------	-------------	---------------

The result grid is currently empty, indicating that no results were found for the query. The interface also includes a toolbar with various icons for file operations, a 'Limit to 1000 rows' dropdown, and a 'Read Only' status indicator at the bottom right.

8. Añadir un nuevo departamento de nombre NUEVO y con director en funciones.

```
SET FOREIGN_KEY_CHECKS = 0;  
SET AUTOCOMMIT = 0;  
INSERT INTO departamentos (numde, numce, direc, tidir, presu, depde, nomde)  
VALUES (140, 10, 160, 'F', 72, 100, 'NUEVO');  
SET FOREIGN_KEY_CHECKS = 1;  
ROLLBACK;
```

```
SELECT *  
FROM departamentos;
```

The screenshot shows a SQL IDE interface with a script editor and a result grid. The script editor contains the following SQL commands:

```
1  -- Añadir un nuevo departamento de nombre NUEVO y con director en funciones.  
2  
3  • USE tienda;  
4  • SELECT *  
5  FROM departamentos;  
6  
7  • INSERT INTO departamentos (numde,numce,direc,tidir,presu,depde,nomde)  
8  VALUES (140, 10, 160, 'F', 72, 100, 'NUEVO');  
9  
10 • SELECT *  
11 FROM departamentos;  
12
```

The result grid displays the data from the 'departamentos' table after the execution of the SQL commands. The grid has 7 columns: numde, numce, direc, tidir, presu, depde, and nomde. The data is as follows:

	numde	numce	direc	tidir	presu	depde	nomde
111	20	180	F	66.0	110		SECTOR_INDUSTRIAL
112	20	270	P	54.0	110		SECTOR_SERVICIOS
120	10	150	F	18.0	100		ORGANIZACIÓN
121	10	150	P	12.0	120		PERSONAL
122	10	350	P	36.0	120		PROCESO_DE_DATOS
130	10	310	P	12.0	100		FINANZAS
140	10	160	F	72.0	100		NUEVO

The interface also shows a toolbar with various icons for file operations, a 'Limit to 1000 rows' dropdown, and a 'Result Grid' button on the right side. The status bar at the bottom indicates 'departamentos 2' and 'Apply'/'Revert' buttons.

9. Añadir un nuevo empleado de nombre NORBERTO y sin departamento asignado.
Inventar el resto de los datos.

```
SET FOREIGN_KEY_CHECKS = 0;
SET AUTOCOMMIT = 0;
INSERT INTO empleados (numem, extel, fecna, fecin, salar, comis, numhi, nomem, numde)
VALUES (570, 926, '1998-02-15', '2019-02-15', 1900, NULL, 4, 'NORBERTO', NULL);
SET FOREIGN_KEY_CHECKS = 1;
ROLLBACK;

SELECT *
FROM empleados;
```

The screenshot shows a SQL IDE interface with a script editor and a result grid. The script editor contains the following SQL code:

```
14
15 -- Añadir un nuevo empleado de nombre NORBERTO y sin departamento asignado. Inventar el resto de datos.
16
17 • insert into empleados (numem, extel, fecna, fecin, salar, comis, numhi, nomem, numde)
18 values (570, 926, '1998-02-15', '2019-02-15', 1900, NULL, 4, 'NORBERTO', NULL);
19
20 • SELECT *
21 FROM empleados;
22
23
24
25
```

The result grid displays the data after the SQL script execution. The columns are: numem, extel, fecna, fecin, salar, comis, numhi, nomem, numde. The data is as follows:

	numem	extel	fecna	fecin	salar	comis	numhi	nomem	numde
	490	880	1964-06-06	1988-01-01	1090	100	0	HORACIO	112
	500	750	1965-10-08	1987-01-01	1200	100	0	HONORIA	111
	510	550	1966-05-04	1986-11-01	1200	NULL	1	ROMULO	110
	550	780	1970-01-10	1998-01-21	600	120	0	SANCHO	111
▶	560	926	0000-00-00	2019-02-15	1900	NULL	4	NORBERTO	NULL
	570	926	1998-02-15	2019-02-15	1900	NULL	4	NORBERTO	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

The result grid also includes a toolbar with options like Filter Rows, Edit, Export/Import, and Wrap Cell Content. The status bar at the bottom shows the table name 'empleados3' and buttons for Apply and Revert.

9B. Como por error he creado dos empleados NORBERTO, borro de la tabla el empleado NORBERTO con fecha de nacimiento 0000-00-00

```
SET FOREIGN_KEY_CHECKS = 0;  
DELETE FROM empleados  
WHERE numem = 560;  
SET FOREIGN_KEY_CHECKS = 1;
```

The screenshot shows a SQL IDE window with a script editor and a result grid. The script editor contains the following SQL code:

```
15 -- Añadir un nuevo empleado de nombre NORBERTO y sin departamento asignado. Invenstar el resto de datos.  
16  
17 • insert into empleados (numem, extel, fecna, fecin, salar, comis, numhi, nomem, numde)  
18 values (570, 926, '1998-02-15', '2019-02-15', 1900, NULL, 4, 'NORBERTO', NULL);  
19  
20 • SELECT *  
21 FROM empleados;  
22  
23 • DELETE FROM empleados  
24 WHERE numem = 560;  
25  
26
```

The result grid displays the following data:

	numem	extel	fecna	fecin	salar	comis	numhi	nomem	numde
	480	760	1965-04-04	1986-02-28	1260	100	1	DIANA	111
	490	880	1964-06-06	1988-01-01	1090	100	0	HORACIO	112
	500	750	1965-10-08	1987-01-01	1200	100	0	HONORIA	111
	510	550	1966-05-04	1986-11-01	1200	NULL	1	ROMULO	110
	550	780	1970-01-10	1998-01-21	600	120	0	SANCHO	111
	570	926	1998-02-15	2019-02-15	1900	NULL	4	NORBERTO	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

The IDE interface includes a toolbar with various icons, a status bar at the bottom with 'empleados 4 x', and buttons for 'Apply' and 'Revert'.

10. Muestra los departamentos que no tienen empleados.

```
SELECT departamentos.numde AS DEPARTAMENTO, empleados.numde AS EMPLEADOS  
FROM departamentos  
LEFT JOIN empleados ON departamentos.numde = empleados.numde  
WHERE empleados.numde IS NULL;
```

The screenshot shows a SQL IDE window with the title 'EXPLICACION 23-01-2023 UF5'. The active tab is 'SQL File 5*'. The query editor contains the following SQL code:

```
1  -- MUESTRA LOS DEPARTAMENTOS QUE NO TIENEN EMPLEADOS  
2  •  USE tienda;  
3  •  SELECT departamentos.numde AS DEPARTAMENTO, empleados.numde AS EMPLEADOS  
4  FROM departamentos  
5  LEFT JOIN empleados ON departamentos.numde = empleados.numde  
6  WHERE empleados.numde IS NULL;
```

The result grid at the bottom shows the following data:

	DEPARTAMENTO	EMPLEADOS
▶ 140		NULL

11. Muestra los nombres de departamentos que no tienen empleados haciendo uso la combinación externa LEFT JOIN. Muestra una segunda columna con los nombres de empleados para asegurarnos que realmente está a NULL.

```
SELECT departamentos.numde AS DEPARTAMENTO, departamentos.nomde AS  
NOMBRE_DEPARTAMENTO, empleados.numde AS ID_EMPLEADO, empleados.nomem  
AS NOMBRE_EMPLEADO  
FROM departamentos  
LEFT JOIN empleados ON departamentos.numde = empleados.numde  
WHERE empleados.numde IS NULL;
```

The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1 • USE tienda;  
2  
3 -- Muestra los nombres de departamentos que no tienen empleados haciendo uso la combinación externa LEFT JOIN.  
4 -- Muestra una segunda columna con los nombres de empleados para asegurarnos que realmente está a NULL.  
5 • SELECT departamentos.numde AS DEPARTAMENTO, departamentos.nomde AS NOMBRE_DEPARTAMENTO, empleados.numde AS ID_EMPLEADO,  
6 empleados.nomem AS NOMBRE_EMPLEADO  
7 FROM departamentos  
8 LEFT JOIN empleados ON departamentos.numde = empleados.numde  
9 WHERE empleados.numde IS NULL;
```

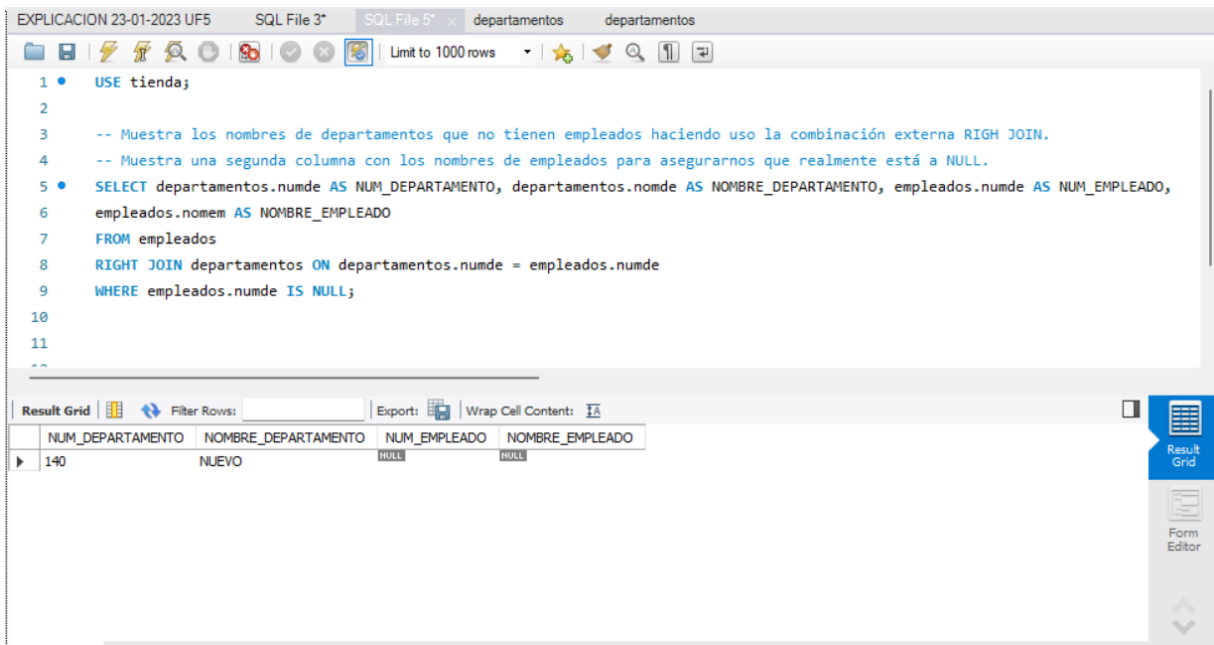
The results grid shows the following data:

	DEPARTAMENTO	NOMBRE_DEPARTAMENTO	ID_EMPLEADO	NOMBRE_EMPLEADO
140		NUEVO	NULL	NULL

The interface also includes a toolbar with various icons, a filter row section, and a sidebar with options like 'Result Grid', 'Form Editor', and 'Read Only'.

12. Muestra los nombres de departamentos que no tienen empleados haciendo uso la combinación externa RIGHT JOIN. Muestra una segunda columna con los nombres de empleados para asegurarnos que realmente está a NULL.

```
SELECT departamentos.numde AS NUM_DEPARTAMENTO, departamentos.nomde AS  
NOMBRE_DEPARTAMENTO, empleados.numde AS NUM_EMPLEADO, empleados.nomem  
AS NOMBRE_EMPLEADO  
FROM empleados  
RIGHT JOIN departamentos ON departamentos.numde = empleados.numde  
WHERE empleados.numde IS NULL;
```



The screenshot shows a SQL IDE interface with a query editor and a result grid. The query editor contains the following SQL code:

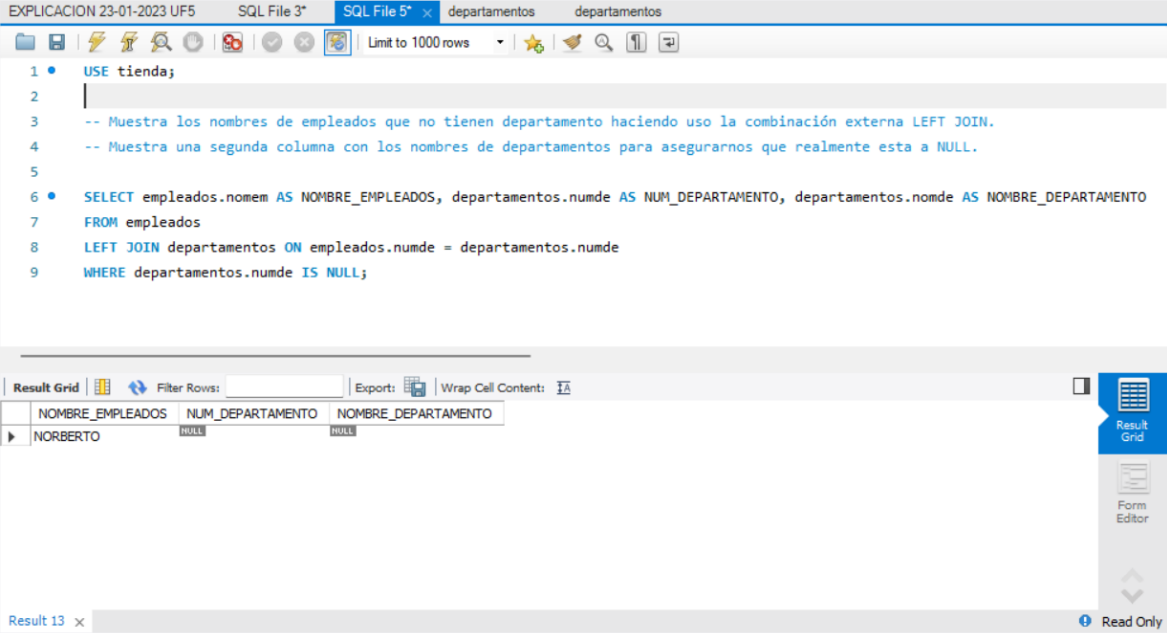
```
1 • USE tienda;  
2  
3 -- Muestra los nombres de departamentos que no tienen empleados haciendo uso la combinación externa RIGHT JOIN.  
4 -- Muestra una segunda columna con los nombres de empleados para asegurarnos que realmente está a NULL.  
5 • SELECT departamentos.numde AS NUM_DEPARTAMENTO, departamentos.nomde AS NOMBRE_DEPARTAMENTO, empleados.numde AS NUM_EMPLEADO,  
6 empleados.nomem AS NOMBRE_EMPLEADO  
7 FROM empleados  
8 RIGHT JOIN departamentos ON departamentos.numde = empleados.numde  
9 WHERE empleados.numde IS NULL;  
10  
11  
12
```

The result grid shows the following data:

	NUM_DEPARTAMENTO	NOMBRE_DEPARTAMENTO	NUM_EMPLEADO	NOMBRE_EMPLEADO
140		NUEVO	NULL	NULL

13. Muestra los nombres de empleados que no tienen departamento haciendo uso la combinación externa LEFT JOIN. Muestra una segunda columna con los nombres de departamentos para asegurarnos que realmente está a NULL.

```
SELECT departamentos.numde AS NUM_DEPARTAMENTO, departamentos.nomde AS  
NOMBRE_DEPARTAMENTO, empleados.nomem AS NOMBRE_EMPLEADO  
FROM empleados  
LEFT JOIN departamentos ON empleados.numde = departamentos.numde  
WHERE departamentos.numde IS NULL;
```



The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1 • USE tienda;  
2  
3 -- Muestra los nombres de empleados que no tienen departamento haciendo uso la combinación externa LEFT JOIN.  
4 -- Muestra una segunda columna con los nombres de departamentos para asegurarnos que realmente esta a NULL.  
5  
6 • SELECT empleados.nomem AS NOMBRE_EMPLEADOS, departamentos.numde AS NUM_DEPARTAMENTO, departamentos.nomde AS NOMBRE_DEPARTAMENTO  
7 FROM empleados  
8 LEFT JOIN departamentos ON empleados.numde = departamentos.numde  
9 WHERE departamentos.numde IS NULL;
```

The results grid displays the following data:

NOMBRE_EMPLEADOS	NUM_DEPARTAMENTO	NOMBRE_DEPARTAMENTO
NORBERTO	NULL	NULL

The interface includes a toolbar with various icons, a status bar at the bottom indicating "Result 13" and "Read Only", and a sidebar with "Result Grid" and "Form Editor" options.

14. Muestra los nombres de empleados que no tienen departamento haciendo uso la combinación externa RIGHT JOIN. Muestra una segunda columna con los nombres de empleados para asegurarnos que realmente está a NULL.

```
SELECT empleados.nombre AS NOMBRE_EMPLEADOS, departamentos.numde AS  
NUM_DEPARTAMENTO, departamentos.nomde AS NOMBRE_DEPARTAMENTO  
FROM departamentos  
RIGHT JOIN empleados ON departamentos.numde = empleados.numde  
WHERE empleados.numde IS NULL;
```

The screenshot shows a SQL IDE interface. The query editor at the top contains the following SQL code:

```
275  
276 /* 14. Muestra los nombres de empleados que no tienen departamento haciendo uso la combinación externa RIGHT JOIN.  
277 Muestra una segunda columna con los nombres de empleados para asegurarnos que realmente está a NULL. */  
278  
279 SELECT empleados.nombre AS NOMBRE_EMPLEADOS, departamentos.numde AS NUM_DEPARTAMENTO, departamentos.nomde AS NOMBRE_DEPARTAMENTO  
280 FROM departamentos  
281 RIGHT JOIN empleados ON departamentos.numde = empleados.numde  
282 WHERE empleados.numde IS NULL;  
283
```

Below the query editor, the 'Result Grid' shows the results of the query. It has three columns: NOMBRE_EMPLEADOS, NUM_DEPARTAMENTO, and NOMBRE_DEPARTAMENTO. The first row shows the name 'NORBERTO' and two NULL values.

NOMBRE_EMPLEADOS	NUM_DEPARTAMENTO	NOMBRE_DEPARTAMENTO
NORBERTO	NULL	NULL

At the bottom, the 'Output' pane shows a log of database actions. The last action is a SELECT query that returned 35 rows.

#	Time	Action	Message
24	22:30:26	delete from empleados where numde in (SELECT numde FROM departamentos WHERE numde = 'NUEVO')	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version f
25	22:32:28	delete from departamentos WHERE numde in (SELECT numde FROM empleados WHERE nombre = 'NORBERTO')	0 row(s) affected
26	22:33:10	SELECT * FROM tienda.empleados	35 row(s) returned
27	22:33:10	SQL FTT * FROM tienda.departamentos	0 row(s) returned

Queries 15 al 20: María Isabel Martín

15. Muestra los departamentos que no tienen empleados y los empleados que no tiene departamento haciendo uso la combinación externa FULL JOIN.

/*

Por problemas de compatibilidad, en muchos casos MySQL no acepta el full join, cuyo esquema es el siguiente:

SELECT columns

FROM table1

FULL [OUTER] JOIN table2

ON table1.column = table2.column;

Para solucionarlo, interesa aplicar la unión de un left join con un right join, como se va a desarrollar a continuación:

*/

SELECT *

FROM departamentos LEFT JOIN empleados

ON departamentos.numde = empleados.numde

UNION

SELECT *

FROM departamentos RIGHT JOIN empleados

partamentos.numde = empleados.numde;

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree with 'departamentos' and 'empleados' tables. The main editor shows a SQL query for Query 15, which uses a combination of LEFT and RIGHT JOINs to find departments without employees and employees without departments. The 'Result Grid' at the bottom shows the output of the query, listing columns like numde, numce, direc, idr, presu, depte, nomde, numem, extel, fecna, fecin, salar, coms, numh, nomem, and numde. The results are organized into two sections: one for departments without employees and another for employees without departments. The 'Output' pane at the bottom shows the execution of the query, including the SQL statement and the number of rows returned.

numde	numce	direc	idr	presu	depte	nomde	numem	extel	fecna	fecin	salar	coms	numh	nomem	numde
121	10	100	P	12.0	120	PERSONAL	100	340	1975-09-10	1997-05-13	2000	OTR	0	JAJJO	121
121	10	150	P	12.0	120	PERSONAL	100	250	1975-09-12	1992-02-11	1750	OTR	4	JAJJANA	121
121	10	100	P	12.0	120	PERSONAL	370	360	1977-06-22	2000-01-20	1860	OTR	1	FABOLA	121
122	10	300	P	36.0	120	PROCESO_DE_DATOS	285	620	1979-05-25	2011-02-15	1910	OTR	0	OTILLA	122
122	10	300	P	36.0	120	PROCESO_DE_DATOS	320	630	1975-12-25	2003-02-05	2400	OTR	2	CONVELDO	122
122	10	300	P	36.0	120	PROCESO_DE_DATOS	390	630	1979-04-13	1999-09-10	2700	OTR	1	ALFREDO	122
122	10	300	P	36.0	120	PROCESO_DE_DATOS	410	660	1968-03-14	1999-10-15	1910	OTR	0	ARZENA	122
122	10	300	P	36.0	120	PROCESO_DE_DATOS	430	680	1967-10-26	1988-11-19	1200	OTR	1	VALERIANA	122
130	10	310	P	12.0	100	FINANZAS	280	410	1978-01-10	2010-05-08	1900	OTR	5	DOROTEA	130
130	10	310	P	12.0	100	FINANZAS	310	460	1978-11-21	2011-01-10	2900	OTR	0	AUGUSTO	130
140	10	310	P	12.0	100	FINANZAS	420	480	1968-10-22	1988-11-19	2400	OTR	0	CLAUSIA	130
140	100	P	72.0	330	NAUEVO		970	926	1968-02-15	2019-02-15	1900	OTR	4	NORBERTO	

16. Muestra los empleados y sus respectivos departamentos haciendo uso de la combinación interna INNER JOIN. ¿Aparecen el departamento NUEVO y el empleado NORBERTO? ¿Por qué?

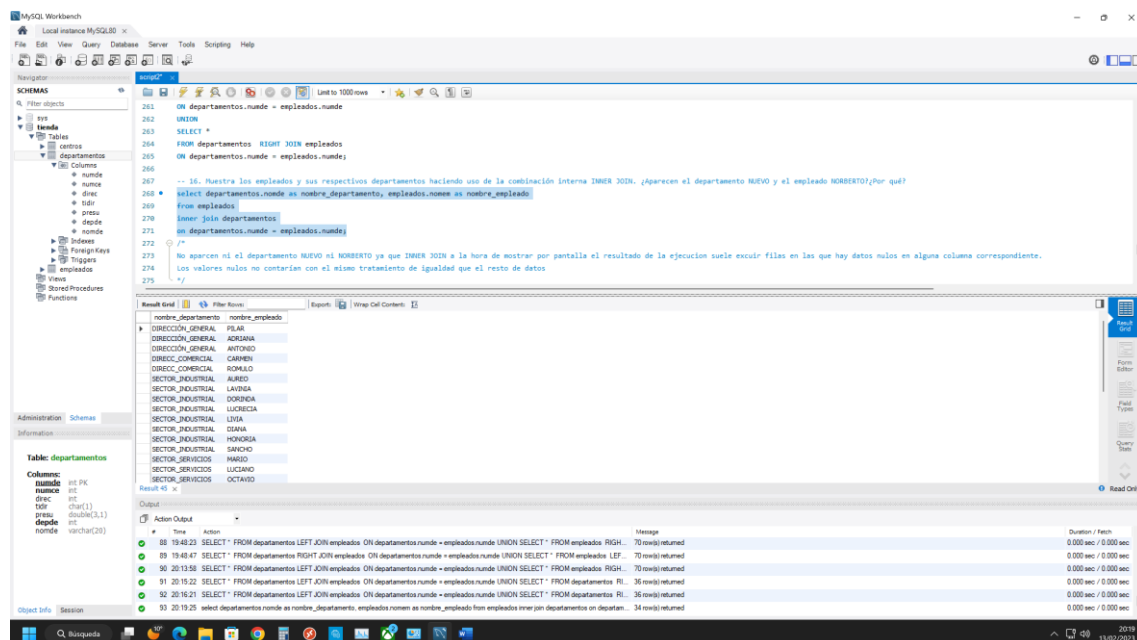
```
SELECT departamentos.nomde AS nombre_departamento, empleados.nomem AS  
nombre_empleado  
FROM empleados  
INNER JOIN departamentos  
ON departamentos.numde = empleados.numde;
```

/*

No aparecen ni el departamento NUEVO ni NORBERTO ya que INNER JOIN a la hora de mostrar por pantalla el resultado de la ejecución suele excluir filas en las que hay datos nulos en alguna columna correspondiente.

Los valores nulos no contarían con el mismo tratamiento de igualdad que el resto de datos.

*/



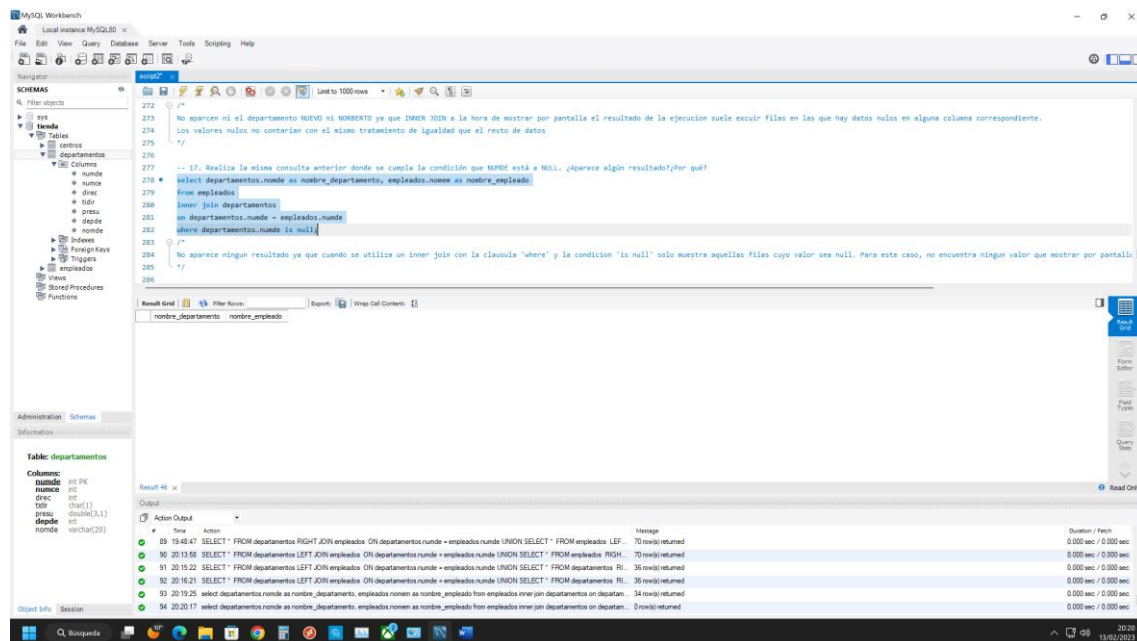
17. Realiza la misma consulta anterior donde se cumpla la condición que NUMDE está a NULL. ¿Aparece algún resultado? ¿Por qué?

```
SELECT departamentos.nomde AS nombre_departamento, empleados.nomem AS  
nombre_empleado  
FROM empleados  
INNER JOIN departamentos  
ON departamentos.numde = empleados.numde  
WHERE departamentos.numde is null;
```

/*

No aparece ningún resultado ya que cuando se utiliza un inner join con la cláusula 'where' y la condición 'is null' solo muestra aquellas filas cuyo valor sea null. Para este caso, no encuentra ningún valor que mostrar por pantalla.

*/



18. Muestra los empleados y sus respectivos departamentos haciendo uso de la combinación interna NATURAL JOIN.

```
SELECT *  
FROM empleados  
NATURAL JOIN departamentos;
```

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
-- 17. Realiza la misma consulta anterior donde se cumple la condición que NUNQUE está a NULL. ¿Aparece algún resultado? ¿Por qué?  
select departamentos.nombre as nombre_departamento, empleados.nombre as nombre_empleado  
from empleados  
inner join departamentos  
on departamentos.nume = empleados.nume  
where departamentos.nume is null;  
  
-- 18. Muestra los empleados y sus respectivos departamentos haciendo uso de la combinación interna NATURAL JOIN.  
select *  
from empleados  
natural join departamentos;
```

The results grid shows the output of the NATURAL JOIN query. It displays 15 columns: nume, nume, enal, feces, feces, idre, nume, nume, direc, hde, presu, depte, and nombre. The data is as follows:

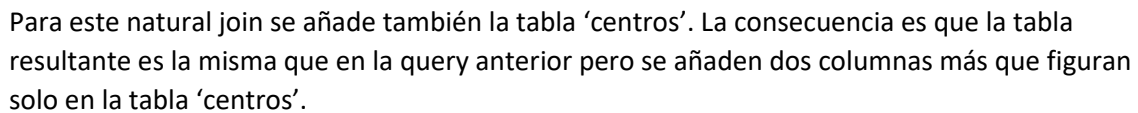
nume	nume	enal	feces	feces	idre	nume	nume	direc	hde	presu	depte	nombre	
100	210	200	1979-09-28	1999-01-22	1910	0000	2	PILAR	10	260	P	72.0	DIRECCIÓN GENERAL
100	230	230	1979-10-27	1997-03-01	2700	0000	0	ADRIANA	10	260	P	72.0	DIRECCIÓN GENERAL
100	260	230	1979-12-03	2001-07-02	730	0000	6	ANTONIO	10	260	P	72.0	DIRECCIÓN GENERAL
110	390	300	1976-02-19	2010-10-08	1290	0000	1	CARMEN	20	180	P	90.0	DIRECCIÓN COMERCIAL
110	510	510	1966-05-04	1986-11-01	1200	0000	1	RODOLFO	20	180	P	90.0	DIRECCIÓN COMERCIAL
111	160	740	1968-07-09	2005-11-01	1800	110	2	ALBERTO	20	180	F	66.0	SECTOR INDUSTRIAL
111	240	760	1967-02-26	1989-02-24	1700	100	3	LAUREN	20	180	F	66.0	SECTOR INDUSTRIAL
111	260	750	1979-10-29	1999-10-10	1800	100	2	DORINDA	20	180	F	66.0	SECTOR INDUSTRIAL
111	400	780	1979-08-18	2011-11-01	1100	0000	0	LUCRECIA	20	180	F	66.0	SECTOR INDUSTRIAL
111	440	760	1966-09-26	1986-02-28	1260	100	0	LINA	20	180	F	66.0	SECTOR INDUSTRIAL
111	480	760	1965-04-04	1986-02-28	1260	100	1	DIANA	20	180	F	66.0	SECTOR INDUSTRIAL
111	500	750	1965-10-08	1987-01-01	1200	100	0	HONORATA	20	180	F	66.0	SECTOR INDUSTRIAL
111	550	780	1970-01-10	1998-01-21	800	120	0	SANCHO	20	180	F	66.0	SECTOR INDUSTRIAL
112	120	840	1968-06-09	1989-10-01	1900	110	1	HAROLD	20	270	P	54.0	SECTOR SERVICIOS
112	130	810	1995-09-09	1981-02-01	1000	110	2	LUCIANO	20	270	P	54.0	SECTOR SERVICIOS
112	270	800	1975-09-21	2003-09-10	1910	80	3	OCTAVIO	20	270	P	54.0	SECTOR SERVICIOS

The bottom part of the screenshot shows the 'Action Output' tab with the following messages:

- 90 20:13:58 SELECT * FROM departamentos LEFT JOIN empleados ON departamentos.nume = empleados.nume UNION SELECT * FROM empleados RIGHT JOIN departamentos ON departamentos.nume = empleados.nume UNION SELECT * FROM departamentos NATURAL JOIN departamentos; 70 rows returned
- 91 20:15:22 SELECT * FROM departamentos LEFT JOIN empleados ON departamentos.nume = empleados.nume UNION SELECT * FROM departamentos RIGHT JOIN departamentos ON departamentos.nume = empleados.nume UNION SELECT * FROM departamentos NATURAL JOIN departamentos; 36 rows returned
- 92 20:15:21 SELECT * FROM departamentos LEFT JOIN empleados ON departamentos.nume = empleados.nume UNION SELECT * FROM departamentos RIGHT JOIN departamentos ON departamentos.nume = empleados.nume UNION SELECT * FROM departamentos NATURAL JOIN departamentos; 36 rows returned
- 93 20:15:25 select departamentos.nume as nombre_departamento, empleados.nombre as nombre_empleado from empleados inner join departamentos on departamentos.nume = empleados.nume; 34 rows returned
- 94 20:20:17 select departamentos.nume as nombre_departamento, empleados.nombre as nombre_empleado from empleados inner join departamentos on departamentos.nume = empleados.nume; 0 rows returned
- 95 20:21:39 select * from empleados natural join departamentos LIMIT 0, 1000; 34 rows returned

Con el natural join se consigue que en una misma tabla resultante se incorporen todas las columnas de las tablas escogidas. En el caso de que alguna se repite solo se muestra una. En este caso se cuentan 15 columnas.

```
SELECT*
FROM empleados
NATURAL JOIN departamentos
NATURAL JOIN centros;
```



20. Borra los registros dados de alta para el departamento NUEVO y el empleado introducido en el apartado anterior.

```
SET FOREIGN_KEY_CHECKS = 0;
DELETE
FROM empleados
WHERE nomem = 'NORBERTO';
SET FOREIGN_KEY_CHECKS = 1;
```

```
SET FOREIGN_KEY_CHECKS = 0;
DELETE
FROM departamentos
WHERE nomde = 'NUEVO';
SET FOREIGN_KEY_CHECKS = 1;
```

```
SELECT *
FROM empleados, departamentos
WHERE empleados.numde = departamentos.numde
GROUP BY departamentos.nomde;
```

</

Tras realizar los 'delete' se comprueba con select que, efectivamente, se han eliminado esos registros.