Ejercicios. Realización de consultas SQL

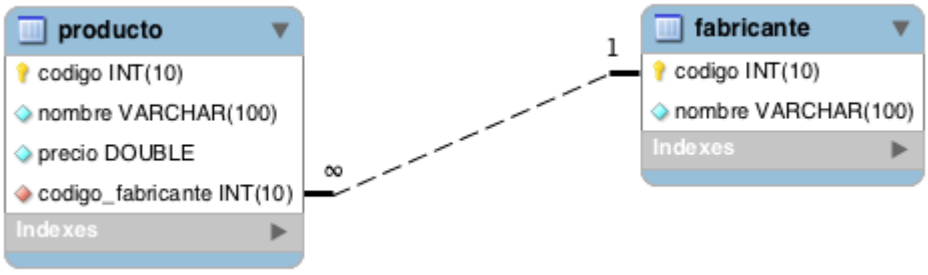
ASIR / DAW - (Gestión de) Bases de datos

IES Celia Viñas (Almería) - 2022/2023

1 Ejercicios. Realización de consultas SQL

1.1 Tienda de informática

1.1.1 Modelo entidad/relación



1.1.2 Base de datos para MySQL

```
DROP DATABASE IF EXISTS tienda;
CREATE DATABASE tienda CHARACTER SET utf8mb4;
USE tienda;
```

```sql
CREATE TABLE fabricante (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(100) NOT NULL
);

CREATE TABLE producto (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(100) NOT NULL,
  precio DOUBLE NOT NULL,
  id_fabricante INT UNSIGNED NOT NULL,
  FOREIGN KEY (id_fabricante) REFERENCES fabricante(id)
);

INSERT INTO fabricante VALUES(1, 'Asus');
INSERT INTO fabricante VALUES(2, 'Lenovo');
INSERT INTO fabricante VALUES(3, 'Hewlett-Packard');
INSERT INTO fabricante VALUES(4, 'Samsung');
INSERT INTO fabricante VALUES(5, 'Seagate');
INSERT INTO fabricante VALUES(6, 'Crucial');
INSERT INTO fabricante VALUES(7, 'Gigabyte');
INSERT INTO fabricante VALUES(8, 'Huawei');
INSERT INTO fabricante VALUES(9, 'Xiaomi');

INSERT INTO producto VALUES(1, 'Disco duro SATA3 1TB', 86.99, 5);
INSERT INTO producto VALUES(2, 'Memoria RAM DDR4 8GB', 120, 6);
INSERT INTO producto VALUES(3, 'Disco SSD 1 TB', 150.99, 4);
INSERT INTO producto VALUES(4, 'GeForce GTX 1050Ti', 185, 7);
INSERT INTO producto VALUES(5, 'GeForce GTX 1080 Xtreme', 755, 6);
INSERT INTO producto VALUES(6, 'Monitor 24 LED Full HD', 202, 1);
INSERT INTO producto VALUES(7, 'Monitor 27 LED Full HD', 245.99, 1);
INSERT INTO producto VALUES(8, 'Portátil Yoga 520', 559, 2);
INSERT INTO producto VALUES(9, 'Portátil Ideapd 320', 444, 2);
INSERT INTO producto VALUES(10, 'Impresora HP Deskjet 3720', 59.99, 3);
INSERT INTO producto VALUES(11, 'Impresora HP Laserjet Pro M26nw', 180, 3);
```

1.1.3 Consultas sobre una tabla

1. Lista el nombre de todos los productos que hay en la tabla `producto`.
2. Lista los nombres y los precios de todos los productos de la tabla `producto`.
3. Lista todas las columnas de la tabla `producto`.
4. Lista el nombre de los productos, el precio en euros y el precio en dólares estadounidenses (USD).

5. Lista el nombre de los productos, el precio en euros y el precio en dólares estadounidenses (USD). Utiliza los siguientes alias para las columnas: `nombre de producto`, `euros`, `dólares`.
6. Lista los nombres y los precios de todos los productos de la tabla `producto`, convirtiendo los nombres a mayúscula.
7. Lista los nombres y los precios de todos los productos de la tabla `producto`, convirtiendo los nombres a minúscula.
8. Lista el nombre de todos los fabricantes en una columna, y en otra columna obtenga en mayúsculas los dos primeros caracteres del nombre del fabricante.
9. Lista los nombres y los precios de todos los productos de la tabla `producto`, redondeando el valor del precio.
10. Lista los nombres y los precios de todos los productos de la tabla `producto`, truncando el valor del precio para mostrarlo sin ninguna cifra decimal.
11. Lista el identificador de los fabricantes que tienen productos en la tabla `producto`.
12. Lista el identificador de los fabricantes que tienen productos en la tabla `producto`, eliminando los identificadores que aparecen repetidos.
13. Lista los nombres de los fabricantes ordenados de forma ascendente.
14. Lista los nombres de los fabricantes ordenados de forma descendente.
15. Lista los nombres de los productos ordenados en primer lugar por el nombre de forma ascendente y en segundo lugar por el precio de forma descendente.
16. Devuelve una lista con las 5 primeras filas de la tabla `fabricante`.
17. Devuelve una lista con 2 filas a partir de la cuarta fila de la tabla `fabricante`. La cuarta fila también se debe incluir en la respuesta.
18. Lista el nombre y el precio del producto más barato. (Utilice solamente las cláusulas `ORDER BY` y `LIMIT`)
19. Lista el nombre y el precio del producto más caro. (Utilice solamente las cláusulas `ORDER BY` y `LIMIT`)
20. Lista el nombre de todos los productos del fabricante cuyo identificador de fabricante es igual a 2.
21. Lista el nombre de los productos que tienen un precio menor o igual a 120€.

22. Lista el nombre de los productos que tienen un precio mayor o igual a 400€.
23. Lista el nombre de los productos que **no tienen** un precio mayor o igual a 400€.
24. Lista todos los productos que tengan un precio entre 80€ y 300€. Sin utilizar el operador `BETWEEN`.
25. Lista todos los productos que tengan un precio entre 60€ y 200€. Utilizando el operador `BETWEEN`.
26. Lista todos los productos que tengan un precio mayor que 200€ y que el identificador de fabricante sea igual a 6.
27. Lista todos los productos donde el identificador de fabricante sea 1, 3 o 5. Sin utilizar el operador `IN`.
28. Lista todos los productos donde el identificador de fabricante sea 1, 3 o 5. Utilizando el operador `IN`.
29. Lista el nombre y el precio de los productos en céntimos (Habrá que multiplicar por 100 el valor del precio). Cree un alias para la columna que contiene el precio que se llame `céntimos`.
30. Lista los nombres de los fabricantes cuyo nombre empiece por la letra `S`.
31. Lista los nombres de los fabricantes cuyo nombre termine por la vocal `e`.
32. Lista los nombres de los fabricantes cuyo nombre contenga el carácter `w`.
33. Lista los nombres de los fabricantes cuyo nombre sea de 4 caracteres.
34. Devuelve una lista con el nombre de todos los productos que contienen la cadena `Portátil` en el nombre.
35. Devuelve una lista con el nombre de todos los productos que contienen la cadena `Monitor` en el nombre y tienen un precio inferior a 215 €.
36. Lista el nombre y el precio de todos los productos que tengan un precio mayor o igual a 180€. Ordene el resultado en primer lugar por el precio (en orden descendente) y en segundo lugar por el nombre (en orden ascendente).

1.1.4 Consultas multitabla (Composición interna)

Resuelva todas las consultas utilizando la sintaxis de `SQL1` y `SQL2`.

1. Devuelve una lista con el nombre del producto, precio y nombre de fabricante de todos los productos de la base de datos.
2. Devuelve una lista con el nombre del producto, precio y nombre de fabricante de todos los productos de la base de datos. Ordene el resultado por el nombre del fabricante, por orden alfabético.
3. Devuelve una lista con el identificador del producto, nombre del producto, identificador del fabricante y nombre del fabricante, de todos los productos de la base de datos.
4. Devuelve el nombre del producto, su precio y el nombre de su fabricante, del producto más barato.
5. Devuelve el nombre del producto, su precio y el nombre de su fabricante, del producto más caro.
6. Devuelve una lista de todos los productos del fabricante `Lenovo`.
7. Devuelve una lista de todos los productos del fabricante `Crucial` que tengan un precio mayor que 200€.
8. Devuelve un listado con todos los productos de los fabricantes `Asus`, `Hewlett-Packard` y `Seagate`. Sin utilizar el operador `IN`.
9. Devuelve un listado con todos los productos de los fabricantes `Asus`, `Hewlett-Packard` y `Seagate`. Utilizando el operador `IN`.
10. Devuelve un listado con el nombre y el precio de todos los productos de los fabricantes cuyo nombre termine por la vocal `e`.
11. Devuelve un listado con el nombre y el precio de todos los productos cuyo nombre de fabricante contenga el carácter `w` en su nombre.
12. Devuelve un listado con el nombre de producto, precio y nombre de fabricante, de todos los productos que tengan un precio mayor o igual a 180€. Ordene el resultado en primer lugar por el precio (en orden descendente) y en segundo lugar por el nombre (en orden ascendente)
13. Devuelve un listado con el identificador y el nombre de fabricante, solamente de aquellos fabricantes que tienen productos asociados en la base de datos.

1.1.5 Consultas multitabla (Composición externa)

Resuelva todas las consultas utilizando las cláusulas `LEFT JOIN` y `RIGHT JOIN`.

1. Devuelve un listado de **todos los fabricantes** que existen en la base de datos, junto con los productos que tiene cada uno de ellos. El listado deberá mostrar también aquellos fabricantes que no tienen productos asociados.
2. Devuelve un listado donde sólo aparezcan aquellos fabricantes que no tienen ningún producto asociado.
3. ¿Pueden existir productos que no estén relacionados con un fabricante? Justifique su respuesta.

1.1.6 Consultas resumen

1. Calcula el número total de productos que hay en la tabla `productos`.
2. Calcula el número total de fabricantes que hay en la tabla `fabricante`.
3. Calcula el número de valores distintos de identificador de fabricante aparecen en la tabla `productos`.
4. Calcula la media del precio de todos los productos.
5. Calcula el precio más barato de todos los productos.
6. Calcula el precio más caro de todos los productos.
7. Lista el nombre y el precio del producto más barato.
8. Lista el nombre y el precio del producto más caro.
9. Calcula la suma de los precios de todos los productos.
10. Calcula el número de productos que tiene el fabricante `Asus`.
11. Calcula la media del precio de todos los productos del fabricante `Asus`.
12. Calcula el precio más barato de todos los productos del fabricante `Asus`.
13. Calcula el precio más caro de todos los productos del fabricante `Asus`.
14. Calcula la suma de todos los productos del fabricante `Asus`.
15. Muestra el precio máximo, precio mínimo, precio medio y el número total de productos que tiene el fabricante `Crucial`.
16. Muestra el número total de productos que tiene cada uno de los fabricantes. El listado también debe incluir los fabricantes que no tienen ningún producto. El

resultado mostrará dos columnas, una con el nombre del fabricante y otra con el número de productos que tiene. Ordene el resultado descendentemente por el número de productos.

17. Muestra el precio máximo, precio mínimo y precio medio de los productos de cada uno de los fabricantes. El resultado mostrará el nombre del fabricante junto con los datos que se solicitan.

18. Muestra el precio máximo, precio mínimo, precio medio y el número total de productos de los fabricantes que tienen un precio medio superior a 200€. No es necesario mostrar el nombre del fabricante, con el identificador del fabricante es suficiente.

19. Muestra el nombre de cada fabricante, junto con el precio máximo, precio mínimo, precio medio y el número total de productos de los fabricantes que tienen un precio medio superior a 200€. Es necesario mostrar el nombre del fabricante.

20. Calcula el número de productos que tienen un precio mayor o igual a 180€.

21. Calcula el número de productos que tiene cada fabricante con un precio mayor o igual a 180€.

22. Lista el precio medio los productos de cada fabricante, mostrando solamente el identificador del fabricante.

23. Lista el precio medio los productos de cada fabricante, mostrando solamente el nombre del fabricante.

24. Lista los nombres de los fabricantes cuyos productos tienen un precio medio mayor o igual a 150€.

25. Devuelve un listado con los nombres de los fabricantes que tienen 2 o más productos.

26. Devuelve un listado con los nombres de los fabricantes y el número de productos que tiene cada uno con un precio superior o igual a 220 €. No es necesario mostrar el nombre de los fabricantes que no tienen productos que cumplan la condición.

Ejemplo del resultado esperado:

nombre	total
Lenovo	2
Asus	1

nombre	total
Crucial	1

27. Devuelve un listado con los nombres de los fabricantes y el número de productos que tiene cada uno con un precio superior o igual a 220 €. El listado debe mostrar el nombre de todos los fabricantes, es decir, si hay algún fabricante que no tiene productos con un precio superior o igual a 220€ deberá aparecer en el listado con un valor igual a 0 en el número de productos.

Ejemplo del resultado esperado:

nombre	total
Lenovo	2
Crucial	1
Asus	1
Huawei	0
Samsung	0
Gigabyte	0
Hewlett-Packard	0
Xiaomi	0
Seagate	0

28. Devuelve un listado con los nombres de los fabricantes donde la suma del precio de todos sus productos es superior a 1000 €.

29. Devuelve un listado con el nombre del producto más caro que tiene cada fabricante. El resultado debe tener tres columnas: nombre del producto, precio y nombre del fabricante. El resultado tiene que estar ordenado alfabéticamente de menor a mayor por el nombre del fabricante.

1.1.7 Subconsultas (En la cláusula `WHERE`)

1.1.7.1 Con operadores básicos de comparación

1. Devuelve todos los productos del fabricante `Lenovo`. (Sin utilizar `INNER JOIN`).

2. Devuelve todos los datos de los productos que tienen el mismo precio que el producto más caro del fabricante `Lenovo`. (Sin utilizar `INNER JOIN`).
3. Lista el nombre del producto más caro del fabricante `Lenovo`.
4. Lista el nombre del producto más barato del fabricante `Hewlett-Packard`.
5. Devuelve todos los productos de la base de datos que tienen un precio mayor o igual al producto más caro del fabricante `Lenovo`.
6. Lista todos los productos del fabricante `Asus` que tienen un precio superior al precio medio de todos sus productos.

1.1.7.2 Subconsultas con `ALL` y `ANY`

8. Devuelve el producto más caro que existe en la tabla `producto` sin hacer uso de `MAX`, `ORDER BY` ni `LIMIT`.
9. Devuelve el producto más barato que existe en la tabla `producto` sin hacer uso de `MIN`, `ORDER BY` ni `LIMIT`.
10. Devuelve los nombres de los fabricantes que tienen productos asociados. (Utilizando `ALL` o `ANY`).
11. Devuelve los nombres de los fabricantes que no tienen productos asociados. (Utilizando `ALL` o `ANY`).

1.1.7.3 Subconsultas con `IN` y `NOT IN`

12. Devuelve los nombres de los fabricantes que tienen productos asociados. (Utilizando `IN` o `NOT IN`).
13. Devuelve los nombres de los fabricantes que no tienen productos asociados. (Utilizando `IN` o `NOT IN`).

1.1.7.4 Subconsultas con `EXISTS` y `NOT EXISTS`

14. Devuelve los nombres de los fabricantes que tienen productos asociados. (Utilizando `EXISTS` o `NOT EXISTS`).
15. Devuelve los nombres de los fabricantes que no tienen productos asociados. (Utilizando `EXISTS` o `NOT EXISTS`).

1.1.7.5 Subconsultas correlacionadas

16. Lista el nombre de cada fabricante con el nombre y el precio de su producto más caro.

17. Devuelve un listado de todos los productos que tienen un precio mayor o igual a la media de todos los productos de su mismo fabricante.

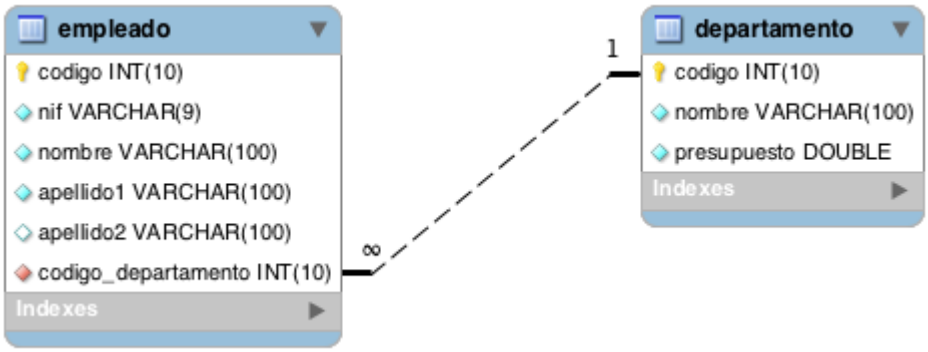
18. Lista el nombre del producto más caro del fabricante `Lenovo`.

1.1.8 Subconsultas (En la cláusula `HAVING`)

7. Devuelve un listado con todos los nombres de los fabricantes que tienen el mismo número de productos que el fabricante `Lenovo`.

1.2 Gestión de empleados

1.2.1 Modelo entidad/relación



1.2.2 Base de datos para MySQL

```
DROP DATABASE IF EXISTS empleados;
CREATE DATABASE empleados CHARACTER SET utf8mb4;
USE empleados;

CREATE TABLE departamento (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(100) NOT NULL,
  presupuesto DOUBLE UNSIGNED NOT NULL,
  gastos DOUBLE UNSIGNED NOT NULL
);

CREATE TABLE empleado (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  nif VARCHAR(9) NOT NULL UNIQUE,
  nombre VARCHAR(100) NOT NULL,
```

```
  apellido1 VARCHAR(100) NOT NULL,
  apellido2 VARCHAR(100),
  id_departamento INT UNSIGNED,
  FOREIGN KEY (id_departamento) REFERENCES departamento(id)
);

INSERT INTO departamento VALUES(1, 'Desarrollo', 120000, 6000);
INSERT INTO departamento VALUES(2, 'Sistemas', 150000, 21000);
INSERT INTO departamento VALUES(3, 'Recursos Humanos', 280000, 25000);
INSERT INTO departamento VALUES(4, 'Contabilidad', 110000, 3000);
INSERT INTO departamento VALUES(5, 'I+D', 375000, 380000);
INSERT INTO departamento VALUES(6, 'Proyectos', 0, 0);
INSERT INTO departamento VALUES(7, 'Publicidad', 0, 1000);

INSERT INTO empleado VALUES(1, '32481596F', 'Aarón', 'Rivero', 'Gómez', 1);
INSERT INTO empleado VALUES(2, 'Y5575632D', 'Adela', 'Salas', 'Díaz', 2);
INSERT INTO empleado VALUES(3, 'R6970642B', 'Adolfo', 'Rubio', 'Flores', 3);
INSERT INTO empleado VALUES(4, '77705545E', 'Adrián', 'Suárez', NULL, 4);
INSERT INTO empleado VALUES(5, '17087203C', 'Marcos', 'Loyola', 'Méndez', 5);
INSERT INTO empleado VALUES(6, '38382980M', 'María', 'Santana', 'Moreno', 1);
INSERT INTO empleado VALUES(7, '80576669X', 'Pilar', 'Ruiz', NULL, 2);
INSERT INTO empleado VALUES(8, '71651431Z', 'Pepe', 'Ruiz', 'Santana', 3);
INSERT INTO empleado VALUES(9, '56399183D', 'Juan', 'Gómez', 'López', 2);
INSERT INTO empleado VALUES(10, '46384486H', 'Diego','Flores', 'Salas', 5);
INSERT INTO empleado VALUES(11, '67389283A', 'Marta','Herrera', 'Gil', 1);
INSERT INTO empleado VALUES(12, '41234836R', 'Irene','Salas', 'Flores', NULL);
INSERT INTO empleado VALUES(13, '82635162B', 'Juan Antonio','Sáez', 'Guerrero', NUL
```

1.2.3 Consultas sobre una tabla

1. Lista el primer apellido de todos los empleados.
2. Lista el primer apellido de los empleados eliminando los apellidos que estén repetidos.
3. Lista todas las columnas de la tabla `empleado` .
4. Lista el nombre y los apellidos de todos los empleados.
5. Lista el identificador de los departamentos de los empleados que aparecen en la tabla `empleado` .
6. Lista el identificador de los departamentos de los empleados que aparecen en la tabla `empleado` , eliminando los identificadores que aparecen repetidos.
7. Lista el nombre y apellidos de los empleados en una única columna.

8. Lista el nombre y apellidos de los empleados en una única columna, convirtiendo todos los caracteres en mayúscula.

9. Lista el nombre y apellidos de los empleados en una única columna, convirtiendo todos los caracteres en minúscula.

10. Lista el identificador de los empleados junto al nif, pero el nif deberá aparecer en dos columnas, una mostrará únicamente los dígitos del nif y la otra la letra.

11. Lista el nombre de cada departamento y el valor del presupuesto actual del que dispone. Para calcular este dato tendrá que restar al valor del presupuesto inicial (columna `presupuesto`) los gastos que se han generado (columna `gastos`). Tenga en cuenta que en algunos casos pueden existir valores negativos. Utilice un alias apropiado para la nueva columna columna que está calculando.

12. Lista el nombre de los departamentos y el valor del presupuesto actual ordenado de forma ascendente.

13. Lista el nombre de todos los departamentos ordenados de forma ascendente.

14. Lista el nombre de todos los departamentos ordenados de forma desscendente.

15. Lista los apellidos y el nombre de todos los empleados, ordenados de forma alfabética tendiendo en cuenta en primer lugar sus apellidos y luego su nombre.

16. Devuelve una lista con el nombre y el presupuesto, de los 3 departamentos que tienen mayor presupuesto.

17. Devuelve una lista con el nombre y el presupuesto, de los 3 departamentos que tienen menor presupuesto.

18. Devuelve una lista con el nombre y el gasto, de los 2 departamentos que tienen mayor gasto.

19. Devuelve una lista con el nombre y el gasto, de los 2 departamentos que tienen menor gasto.

20. Devuelve una lista con 5 filas a partir de la tercera fila de la tabla `empleado`. La tercera fila se debe incluir en la respuesta. La respuesta debe incluir todas las columnas de la tabla `empleado`.

21. Devuelve una lista con el nombre de los departamentos y el presupuesto, de aquellos que tienen un presupuesto mayor o igual a 150000 euros.

22. Devuelve una lista con el nombre de los departamentos y el gasto, de aquellos que tienen menos de 5000 euros de gastos.

23. Devuelve una lista con el nombre de los departamentos y el presupesto, de aquellos que tienen un presupuesto entre 100000 y 200000 euros. Sin utilizar el operador `BETWEEN`.

24. Devuelve una lista con el nombre de los departamentos que **no** tienen un presupuesto entre 100000 y 200000 euros. Sin utilizar el operador `BETWEEN`.

25. Devuelve una lista con el nombre de los departamentos que tienen un presupuesto entre 100000 y 200000 euros. Utilizando el operador `BETWEEN`.

26. Devuelve una lista con el nombre de los departamentos que **no** tienen un presupuesto entre 100000 y 200000 euros. Utilizando el operador `BETWEEN`.

27. Devuelve una lista con el nombre de los departamentos, gastos y presupuesto, de quellos departamentos donde los gastos sean mayores que el presupuesto del que disponen.

28. Devuelve una lista con el nombre de los departamentos, gastos y presupuesto, de aquellos departamentos donde los gastos sean menores que el presupuesto del que disponen.

29. Devuelve una lista con el nombre de los departamentos, gastos y presupuesto, de aquellos departamentos donde los gastos sean iguales al presupuesto del que disponen.

30. Lista todos los datos de los empleados cuyo segundo apellido sea `NULL`.

31. Lista todos los datos de los empleados cuyo segundo apellido **no sea** `NULL`.

32. Lista todos los datos de los empleados cuyo segundo apellido sea `López`.

33. Lista todos los datos de los empleados cuyo segundo apellido sea `Díaz` o `Moreno`. Sin utilizar el operador `IN`.

34. Lista todos los datos de los empleados cuyo segundo apellido sea `Díaz` o `Moreno`. Utilizando el operador `IN`.

35. Lista los nombres, apellidos y nif de los empleados que trabajan en el departamento `3`.

36. Lista los nombres, apellidos y nif de los empleados que trabajan en los departamentos `2`, `4` o `5`.

1.2.4 Consultas multitabla (Composición interna)

Resuelva todas las consultas utilizando la sintaxis de `SQL1` y `SQL2`.

1. Devuelve un listado con los empleados y los datos de los departamentos donde trabaja cada uno.
2. Devuelve un listado con los empleados y los datos de los departamentos donde trabaja cada uno. Ordena el resultado, en primer lugar por el nombre del departamento (en orden alfabético) y en segundo lugar por los apellidos y el nombre de los empleados.
3. Devuelve un listado con el identificador y el nombre del departamento, solamente de aquellos departamentos que tienen empleados.
4. Devuelve un listado con el identificador, el nombre del departamento y el valor del presupuesto actual del que dispone, solamente de aquellos departamentos que tienen empleados. El valor del presupuesto actual lo puede calcular restando al valor del presupuesto inicial (columna `presupuesto`) el valor de los gastos que ha generado (columna `gastos`).
5. Devuelve el nombre del departamento donde trabaja el empleado que tiene el nif `38382980M`.
6. Devuelve el nombre del departamento donde trabaja el empleado `Pepe Ruiz Santana`.
7. Devuelve un listado con los datos de los empleados que trabajan en el departamento de `I+D`. Ordena el resultado alfabéticamente.
8. Devuelve un listado con los datos de los empleados que trabajan en el departamento de `Sistemas`, `Contabilidad` o `I+D`. Ordena el resultado alfabéticamente.
9. Devuelve una lista con el nombre de los empleados que tienen los departamentos que **no** tienen un presupuesto entre 100000 y 200000 euros.
10. Devuelve un listado con el nombre de los departamentos donde existe algún empleado cuyo segundo apellido sea `NULL`. Tenga en cuenta que no debe mostrar nombres de departamentos que estén repetidos.

1.2.5 Consultas multitabla (Composición externa)

Resuelva todas las cláusulas `LEFT JOIN` y `RIGHT JOIN`.

1. Devuelve un listado con **todos los empleados** junto con los datos de los departamentos donde trabajan. Este listado también debe incluir los empleados que no tienen ningún departamento asociado.

2. Devuelve un listado donde sólo aparezcan aquellos empleados que no tienen ningún departamento asociado.

3. Devuelve un listado donde sólo aparezcan aquellos departamentos que no tienen ningún empleado asociado.

4. Devuelve un listado con todos los empleados junto con los datos de los departamentos donde trabajan. El listado debe incluir los empleados que no tienen ningún departamento asociado y los departamentos que no tienen ningún empleado asociado. Ordene el listado alfabéticamente por el nombre del departamento.

5. Devuelve un listado con los empleados que no tienen ningún departamento asociado y los departamentos que no tienen ningún empleado asociado. Ordene el listado alfabéticamente por el nombre del departamento.

1.2.6 Consultas resumen

1. Calcula la suma del presupuesto de todos los departamentos.

2. Calcula la media del presupuesto de todos los departamentos.

3. Calcula el valor mínimo del presupuesto de todos los departamentos.

4. Calcula el nombre del departamento y el presupuesto que tiene asignado, del departamento con menor presupuesto.

5. Calcula el valor máximo del presupuesto de todos los departamentos.

6. Calcula el nombre del departamento y el presupuesto que tiene asignado, del departamento con mayor presupuesto.

7. Calcula el número total de empleados que hay en la tabla `empleado`.

8. Calcula el número de empleados que **no tienen** `NULL` en su segundo apellido.

9. Calcula el número de empleados que hay en cada departamento. Tienes que devolver dos columnas, una con el nombre del departamento y otra con el número de empleados que tiene asignados.

10. Calcula el nombre de los departamentos que tienen más de 2 empleados. El resultado debe tener dos columnas, una con el nombre del departamento y otra con

el número de empleados que tiene asignados.

11. Calcula el número de empleados que trabajan en cada uno de los departamentos. El resultado de esta consulta también tiene que incluir aquellos departamentos que no tienen ningún empleado asociado.

12. Calcula el número de empleados que trabajan en cada unos de los departamentos que tienen un presupuesto mayor a 200000 euros.

1.2.7 Subconsultas

1.2.7.1 Con operadores básicos de comparación

1. Devuelve un listado con todos los empleados que tiene el departamento de `Sistemas`. (Sin utilizar `INNER JOIN`).

2. Devuelve el nombre del departamento con mayor presupuesto y la cantidad que tiene asignada.

3. Devuelve el nombre del departamento con menor presupuesto y la cantidad que tiene asignada.

1.2.7.2 Subconsultas con `ALL` y `ANY`

4. Devuelve el nombre del departamento con mayor presupuesto y la cantidad que tiene asignada. Sin hacer uso de `MAX`, `ORDER BY` ni `LIMIT`.

5. Devuelve el nombre del departamento con menor presupuesto y la cantidad que tiene asignada. Sin hacer uso de `MIN`, `ORDER BY` ni `LIMIT`.

6. Devuelve los nombres de los departamentos que tienen empleados asociados. (Utilizando `ALL` o `ANY`).

7. Devuelve los nombres de los departamentos que no tienen empleados asociados. (Utilizando `ALL` o `ANY`).

1.2.7.3 Subconsultas con `IN` y `NOT IN`

8. Devuelve los nombres de los departamentos que tienen empleados asociados. (Utilizando `IN` o `NOT IN`).

9. Devuelve los nombres de los departamentos que no tienen empleados asociados. (Utilizando `IN` o `NOT IN`).

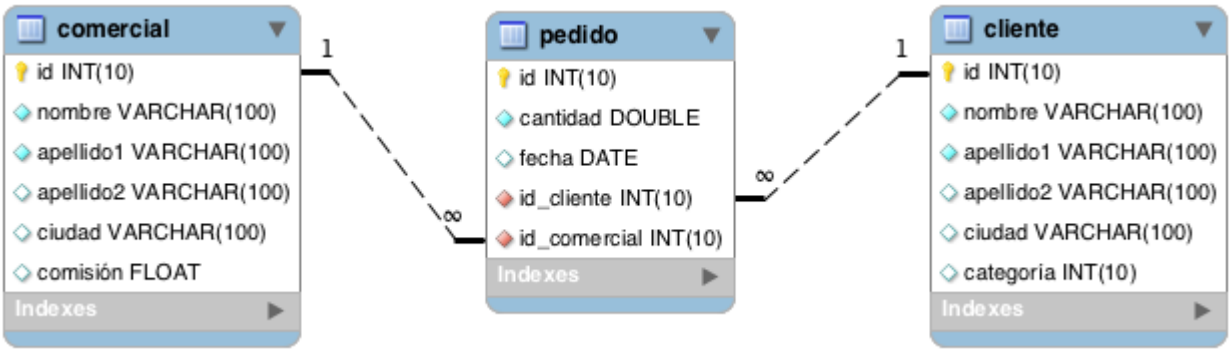
1.2.7.4 Subconsultas con `EXISTS` y `NOT EXISTS`

10. Devuelve los nombres de los departamentos que tienen empleados asociados. (Utilizando `EXISTS` o `NOT EXISTS`).

11. Devuelve los nombres de los departamentos que tienen empleados asociados. (Utilizando `EXISTS` o `NOT EXISTS`).

1.3 Gestión de ventas

1.3.1 Modelo entidad/relación



1.3.2 Base de datos para MySQL

```sql
DROP DATABASE IF EXISTS ventas;
CREATE DATABASE ventas CHARACTER SET utf8mb4;
USE ventas;

CREATE TABLE cliente (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(100) NOT NULL,
  apellido1 VARCHAR(100) NOT NULL,
  apellido2 VARCHAR(100),
  ciudad VARCHAR(100),
  categoría INT UNSIGNED
);

CREATE TABLE comercial (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(100) NOT NULL,
  apellido1 VARCHAR(100) NOT NULL,
  apellido2 VARCHAR(100),
  comisión FLOAT
);

CREATE TABLE pedido (
  id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
```

```
  total DOUBLE NOT NULL,
  fecha DATE,
  id_cliente INT UNSIGNED NOT NULL,
  id_comercial INT UNSIGNED NOT NULL,
  FOREIGN KEY (id_cliente) REFERENCES cliente(id),
  FOREIGN KEY (id_comercial) REFERENCES comercial(id)
);

INSERT INTO cliente VALUES(1, 'Aarón', 'Rivero', 'Gómez', 'Almería', 100);
INSERT INTO cliente VALUES(2, 'Adela', 'Salas', 'Díaz', 'Granada', 200);
INSERT INTO cliente VALUES(3, 'Adolfo', 'Rubio', 'Flores', 'Sevilla', NULL);
INSERT INTO cliente VALUES(4, 'Adrián', 'Suárez', NULL, 'Jaén', 300);
INSERT INTO cliente VALUES(5, 'Marcos', 'Loyola', 'Méndez', 'Almería', 200);
INSERT INTO cliente VALUES(6, 'María', 'Santana', 'Moreno', 'Cádiz', 100);
INSERT INTO cliente VALUES(7, 'Pilar', 'Ruiz', NULL, 'Sevilla', 300);
INSERT INTO cliente VALUES(8, 'Pepe', 'Ruiz', 'Santana', 'Huelva', 200);
INSERT INTO cliente VALUES(9, 'Guillermo', 'López', 'Gómez', 'Granada', 225);
INSERT INTO cliente VALUES(10, 'Daniel', 'Santana', 'Loyola', 'Sevilla', 125);

INSERT INTO comercial VALUES(1, 'Daniel', 'Sáez', 'Vega', 0.15);
INSERT INTO comercial VALUES(2, 'Juan', 'Gómez', 'López', 0.13);
INSERT INTO comercial VALUES(3, 'Diego','Flores', 'Salas', 0.11);
INSERT INTO comercial VALUES(4, 'Marta','Herrera', 'Gil', 0.14);
INSERT INTO comercial VALUES(5, 'Antonio','Carretero', 'Ortega', 0.12);
INSERT INTO comercial VALUES(6, 'Manuel','Domínguez', 'Hernández', 0.13);
INSERT INTO comercial VALUES(7, 'Antonio','Vega', 'Hernández', 0.11);
INSERT INTO comercial VALUES(8, 'Alfredo','Ruiz', 'Flores', 0.05);

INSERT INTO pedido VALUES(1, 150.5, '2017-10-05', 5, 2);
INSERT INTO pedido VALUES(2, 270.65, '2016-09-10', 1, 5);
INSERT INTO pedido VALUES(3, 65.26, '2017-10-05', 2, 1);
INSERT INTO pedido VALUES(4, 110.5, '2016-08-17', 8, 3);
INSERT INTO pedido VALUES(5, 948.5, '2017-09-10', 5, 2);
INSERT INTO pedido VALUES(6, 2400.6, '2016-07-27', 7, 1);
INSERT INTO pedido VALUES(7, 5760, '2015-09-10', 2, 1);
INSERT INTO pedido VALUES(8, 1983.43, '2017-10-10', 4, 6);
INSERT INTO pedido VALUES(9, 2480.4, '2016-10-10', 8, 3);
INSERT INTO pedido VALUES(10, 250.45, '2015-06-27', 8, 2);
INSERT INTO pedido VALUES(11, 75.29, '2016-08-17', 3, 7);
INSERT INTO pedido VALUES(12, 3045.6, '2017-04-25', 2, 1);
INSERT INTO pedido VALUES(13, 545.75, '2019-01-25', 6, 1);
INSERT INTO pedido VALUES(14, 145.82, '2017-02-02', 6, 1);
INSERT INTO pedido VALUES(15, 370.85, '2019-03-11', 1, 5);
INSERT INTO pedido VALUES(16, 2389.23, '2019-03-11', 1, 5);
```

1.3.3 Consultas sobre una tabla

1. Devuelve un listado con todos los pedidos que se han realizado. Los pedidos deben estar ordenados por la fecha de realización, mostrando en primer lugar los pedidos más recientes.

2. Devuelve todos los datos de los dos pedidos de mayor valor.

3. Devuelve un listado con los identificadores de los clientes que han realizado algún pedido. Tenga en cuenta que no debe mostrar identificadores que estén repetidos.

4. Devuelve un listado de todos los pedidos que se realizaron durante el año 2017, cuya cantidad total sea superior a 500€.

5. Devuelve un listado con el nombre y los apellidos de los comerciales que tienen una comisión entre 0.05 y 0.11.

6. Devuelve el valor de la comisión de mayor valor que existe en la tabla `comercial`.

7. Devuelve el identificador, nombre y primer apellido de aquellos clientes cuyo segundo apellido **no** es `NULL`. El listado deberá estar ordenado alfabéticamente por apellidos y nombre.

8. Devuelve un listado de los nombres de los clientes que empiezan por `A` y terminan por `n` y también los nombres que empiezan por `P`. El listado deberá estar ordenado alfabéticamente.

9. Devuelve un listado de los nombres de los clientes que **no** empiezan por `A`. El listado deberá estar ordenado alfabéticamente.

10. Devuelve un listado con los nombres de los comerciales que terminan por `el` o `o`. Tenga en cuenta que se deberán eliminar los nombres repetidos.

1.3.4 Consultas multitabla (Composición interna)

Resuelva todas las consultas utilizando la sintaxis de `SQL1` y `SQL2`.

1. Devuelve un listado con el identificador, nombre y los apellidos de todos los clientes que han realizado algún pedido. El listado debe estar ordenado alfabéticamente y se deben eliminar los elementos repetidos.

2. Devuelve un listado que muestre todos los pedidos que ha realizado cada cliente. El resultado debe mostrar todos los datos de los pedidos y del cliente. El listado debe mostrar los datos de los clientes ordenados alfabéticamente.

3. Devuelve un listado que muestre todos los pedidos en los que ha participado un comercial. El resultado debe mostrar todos los datos de los pedidos y de los

comerciales. El listado debe mostrar los datos de los comerciales ordenados alfabéticamente.

4. Devuelve un listado que muestre todos los clientes, con todos los pedidos que han realizado y con los datos de los comerciales asociados a cada pedido.

5. Devuelve un listado de todos los clientes que realizaron un pedido durante el año `2017`, cuya cantidad esté entre `300 €` y `1000 €`.

6. Devuelve el nombre y los apellidos de todos los comerciales que ha participado en algún pedido realizado por `María Santana Moreno`.

7. Devuelve el nombre de todos los clientes que han realizado algún pedido con el comercial `Daniel Sáez Vega`.

1.3.5 Consultas multitabla (Composición externa)

Resuelva todas las consultas utilizando las cláusulas `LEFT JOIN` y `RIGHT JOIN`.

1. Devuelve un listado con **todos los clientes** junto con los datos de los pedidos que han realizado. Este listado también debe incluir los clientes que no han realizado ningún pedido. El listado debe estar ordenado alfabéticamente por el primer apellido, segundo apellido y nombre de los clientes.

2. Devuelve un listado con **todos los comerciales** junto con los datos de los pedidos que han realizado. Este listado también debe incluir los comerciales que no han realizado ningún pedido. El listado debe estar ordenado alfabéticamente por el primer apellido, segundo apellido y nombre de los comerciales.

3. Devuelve un listado que solamente muestre los clientes que no han realizado ningún pedido.

4. Devuelve un listado que solamente muestre los comerciales que no han realizado ningún pedido.

5. Devuelve un listado con los clientes que no han realizado ningún pedido y de los comerciales que no han participado en ningún pedido. Ordene el listado alfabéticamente por los apellidos y el nombre. En en listado deberá diferenciar de algún modo los clientes y los comerciales.

6. ¿Se podrían realizar las consultas anteriores con `NATURAL LEFT JOIN` o `NATURAL RIGHT JOIN`? Justifique su respuesta.

1.3.6 Consultas resumen

1. Calcula la cantidad total que suman todos los pedidos que aparecen en la tabla `pedido`.
2. Calcula la cantidad media de todos los pedidos que aparecen en la tabla `pedido`.
3. Calcula el número total de comerciales distintos que aparecen en la tabla `pedido`.
4. Calcula el número total de clientes que aparecen en la tabla `cliente`.
5. Calcula cuál es la mayor cantidad que aparece en la tabla `pedido`.
6. Calcula cuál es la menor cantidad que aparece en la tabla `pedido`.
7. Calcula cuál es el valor máximo de categoría para cada una de las ciudades que aparece en la tabla `cliente`.
8. Calcula cuál es el máximo valor de los pedidos realizados durante el mismo día para cada uno de los clientes. Es decir, el mismo cliente puede haber realizado varios pedidos de diferentes cantidades el mismo día. Se pide que se calcule cuál es el pedido de máximo valor para cada uno de los días en los que un cliente ha realizado un pedido. Muestra el identificador del cliente, nombre, apellidos, la fecha y el valor de la cantidad.
9. Calcula cuál es el máximo valor de los pedidos realizados durante el mismo día para cada uno de los clientes, teniendo en cuenta que sólo queremos mostrar aquellos pedidos que superen la cantidad de 2000 €.
10. Calcula el máximo valor de los pedidos realizados para cada uno de los comerciales durante la fecha `2016-08-17`. Muestra el identificador del comercial, nombre, apellidos y total.
11. Devuelve un listado con el identificador de cliente, nombre y apellidos y el número total de pedidos que ha realizado cada uno de clientes. Tenga en cuenta que pueden existir clientes que no han realizado ningún pedido. Estos clientes también deben aparecer en el listado indicando que el número de pedidos realizados es `0`.
12. Devuelve un listado con el identificador de cliente, nombre y apellidos y el número total de pedidos que ha realizado cada uno de clientes **durante el año 2017**.
13. Devuelve un listado que muestre el identificador de cliente, nombre, primer apellido y el valor de la máxima cantidad del pedido realizado por cada uno de los clientes. El resultado debe mostrar aquellos clientes que no han realizado ningún pedido indicando que la máxima cantidad de sus pedidos realizados es `0`. Puede hacer uso de la función `IFNULL`.

14. Devuelve cuál ha sido el pedido de máximo valor que se ha realizado cada año.
15. Devuelve el número total de pedidos que se han realizado cada año.

1.3.7 Subconsultas

1.3.7.1 Con operadores básicos de comparación

1. Devuelve un listado con todos los pedidos que ha realizado `Adela Salas Díaz`. (Sin utilizar `INNER JOIN`).
2. Devuelve el número de pedidos en los que ha participado el comercial `Daniel Sáez Vega`. (Sin utilizar `INNER JOIN`)
3. Devuelve los datos del cliente que realizó el pedido más caro en el año `2019`. (Sin utilizar `INNER JOIN`)
4. Devuelve la fecha y la cantidad del pedido de menor valor realizado por el cliente `Pepe Ruiz Santana`.
5. Devuelve un listado con los datos de los clientes y los pedidos, de todos los clientes que han realizado un pedido durante el año `2017` con un valor mayor o igual al valor medio de los pedidos realizados durante ese mismo año.

1.3.7.2 Subconsultas con `ALL` y `ANY`

6. Devuelve el pedido más caro que existe en la tabla `pedido` si hacer uso de `MAX`, `ORDER BY` ni `LIMIT`.
7. Devuelve un listado de los clientes que no han realizado ningún pedido. (Utilizando `ANY` o `ALL`).
8. Devuelve un listado de los comerciales que no han realizado ningún pedido. (Utilizando `ANY` o `ALL`).

1.3.7.3 Subconsultas con `IN` y `NOT IN`

9. Devuelve un listado de los clientes que no han realizado ningún pedido. (Utilizando `IN` o `NOT IN`).
10. Devuelve un listado de los comerciales que no han realizado ningún pedido. (Utilizando `IN` o `NOT IN`).

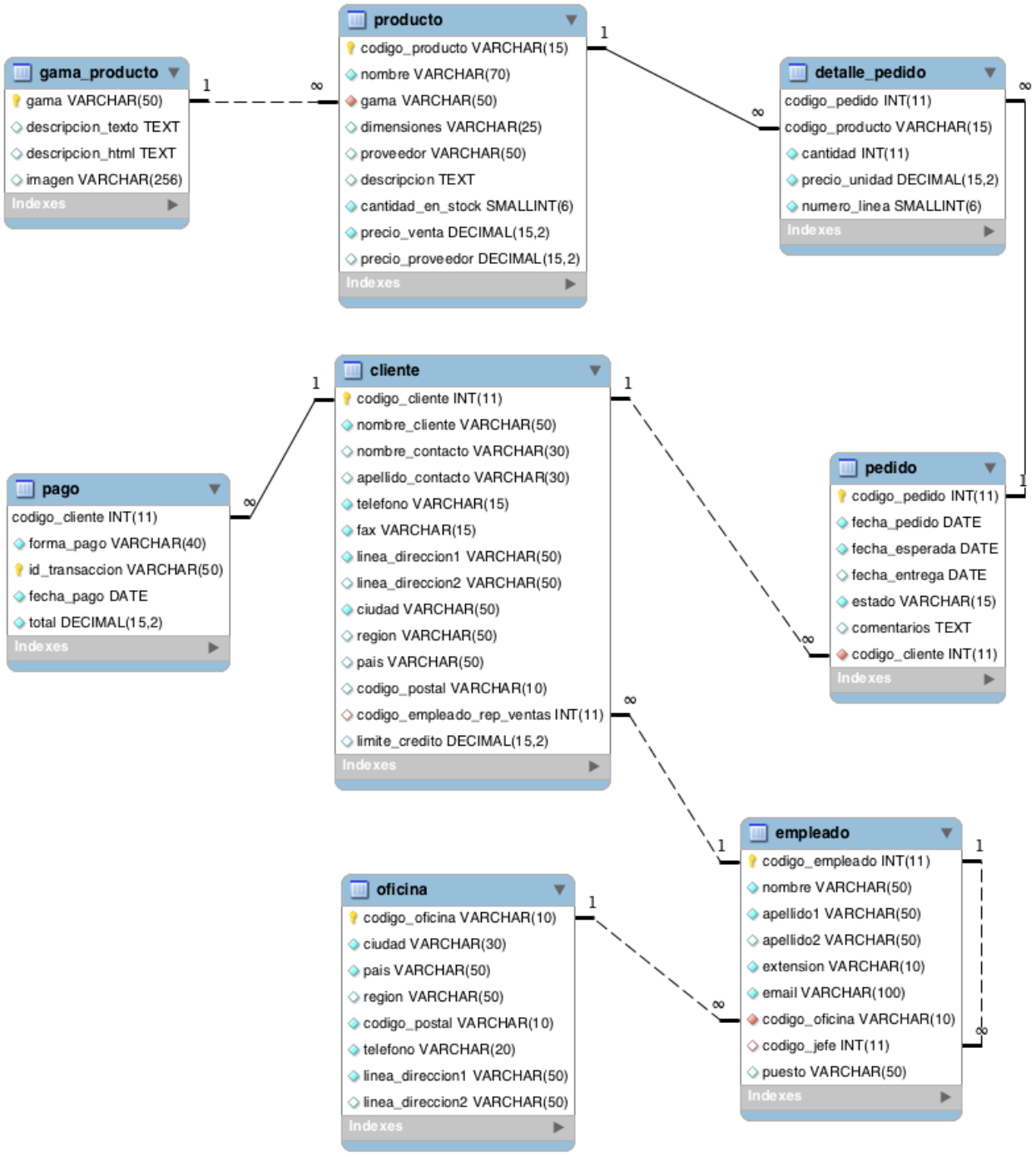
1.3.7.4 Subconsultas con `EXISTS` y `NOT EXISTS`

11. Devuelve un listado de los clientes que no han realizado ningún pedido. (Utilizando `EXISTS` o `NOT EXISTS`).

12. Devuelve un listado de los comerciales que no han realizado ningún pedido. (Utilizando `EXISTS` o `NOT EXISTS`).

1.4 Jardinería

1.4.1 Modelo entidad/relación



1.4.2 Base de datos para MySQL

```sql
DROP DATABASE IF EXISTS jardineria;
CREATE DATABASE jardineria CHARACTER SET utf8mb4;
USE jardineria;

CREATE TABLE oficina (
  codigo_oficina VARCHAR(10) NOT NULL,
  ciudad VARCHAR(30) NOT NULL,
  pais VARCHAR(50) NOT NULL,
  region VARCHAR(50) DEFAULT NULL,
  codigo_postal VARCHAR(10) NOT NULL,
  telefono VARCHAR(20) NOT NULL,
  linea_direccion1 VARCHAR(50) NOT NULL,
  linea_direccion2 VARCHAR(50) DEFAULT NULL,
  PRIMARY KEY (codigo_oficina)
);

CREATE TABLE empleado (
  codigo_empleado INTEGER NOT NULL,
  nombre VARCHAR(50) NOT NULL,
  apellido1 VARCHAR(50) NOT NULL,
  apellido2 VARCHAR(50) DEFAULT NULL,
  extension VARCHAR(10) NOT NULL,
  email VARCHAR(100) NOT NULL,
  codigo_oficina VARCHAR(10) NOT NULL,
  codigo_jefe INTEGER DEFAULT NULL,
  puesto VARCHAR(50) DEFAULT NULL,
  PRIMARY KEY (codigo_empleado),
  FOREIGN KEY (codigo_oficina) REFERENCES oficina (codigo_oficina),
  FOREIGN KEY (codigo_jefe) REFERENCES empleado (codigo_empleado)
);

CREATE TABLE gama_producto (
  gama VARCHAR(50) NOT NULL,
  descripcion_texto TEXT,
  descripcion_html TEXT,
  imagen VARCHAR(256),
  PRIMARY KEY (gama)
);

CREATE TABLE cliente (
  codigo_cliente INTEGER NOT NULL,
  nombre_cliente VARCHAR(50) NOT NULL,
  nombre_contacto VARCHAR(30) DEFAULT NULL,
  apellido_contacto VARCHAR(30) DEFAULT NULL,
  telefono VARCHAR(15) NOT NULL,
  fax VARCHAR(15) NOT NULL,
```

```
  linea_direccion1 VARCHAR(50) NOT NULL,
  linea_direccion2 VARCHAR(50) DEFAULT NULL,
  ciudad VARCHAR(50) NOT NULL,
  region VARCHAR(50) DEFAULT NULL,
  pais VARCHAR(50) DEFAULT NULL,
  codigo_postal VARCHAR(10) DEFAULT NULL,
  codigo_empleado_rep_ventas INTEGER DEFAULT NULL,
  limite_credito NUMERIC(15,2) DEFAULT NULL,
  PRIMARY KEY (codigo_cliente),
  FOREIGN KEY (codigo_empleado_rep_ventas) REFERENCES empleado (codigo_empleado)
);

CREATE TABLE pedido (
  codigo_pedido INTEGER NOT NULL,
  fecha_pedido date NOT NULL,
  fecha_esperada date NOT NULL,
  fecha_entrega date DEFAULT NULL,
  estado VARCHAR(15) NOT NULL,
  comentarios TEXT,
  codigo_cliente INTEGER NOT NULL,
  PRIMARY KEY (codigo_pedido),
  FOREIGN KEY (codigo_cliente) REFERENCES cliente (codigo_cliente)
);

CREATE TABLE producto (
  codigo_producto VARCHAR(15) NOT NULL,
  nombre VARCHAR(70) NOT NULL,
  gama VARCHAR(50) NOT NULL,
  dimensiones VARCHAR(25) NULL,
  proveedor VARCHAR(50) DEFAULT NULL,
  descripcion text NULL,
  cantidad_en_stock SMALLINT NOT NULL,
  precio_venta NUMERIC(15,2) NOT NULL,
  precio_proveedor NUMERIC(15,2) DEFAULT NULL,
  PRIMARY KEY (codigo_producto),
  FOREIGN KEY (gama) REFERENCES gama_producto (gama)
);

CREATE TABLE detalle_pedido (
  codigo_pedido INTEGER NOT NULL,
  codigo_producto VARCHAR(15) NOT NULL,
  cantidad INTEGER NOT NULL,
  precio_unidad NUMERIC(15,2) NOT NULL,
  numero_linea SMALLINT NOT NULL,
  PRIMARY KEY (codigo_pedido, codigo_producto),
  FOREIGN KEY (codigo_pedido) REFERENCES pedido (codigo_pedido),
  FOREIGN KEY (codigo_producto) REFERENCES producto (codigo_producto)
);
```

```sql
CREATE TABLE pago (
  codigo_cliente INTEGER NOT NULL,
  forma_pago VARCHAR(40) NOT NULL,
  id_transaccion VARCHAR(50) NOT NULL,
  fecha_pago date NOT NULL,
  total NUMERIC(15,2) NOT NULL,
  PRIMARY KEY (codigo_cliente, id_transaccion),
  FOREIGN KEY (codigo_cliente) REFERENCES cliente (codigo_cliente)
);
```

1.4.3 Datos

Acceder al *script* SQL para la creación de la base de datos y la inserción de datos: **jardineria.sql**.

1.4.4 Consultas sobre una tabla

1. Devuelve un listado con el código de oficina y la ciudad donde hay oficinas.
2. Devuelve un listado con la ciudad y el teléfono de las oficinas de España.
3. Devuelve un listado con el nombre, apellidos y email de los empleados cuyo jefe tiene un código de jefe igual a 7.
4. Devuelve el nombre del puesto, nombre, apellidos y email del jefe de la empresa.
5. Devuelve un listado con el nombre, apellidos y puesto de aquellos empleados que no sean representantes de ventas.
6. Devuelve un listado con el nombre de los todos los clientes españoles.
7. Devuelve un listado con los distintos estados por los que puede pasar un pedido.
8. Devuelve un listado con el código de cliente de aquellos clientes que realizaron algún pago en 2008. Tenga en cuenta que deberá eliminar aquellos códigos de cliente que aparezcan repetidos. Resuelva la consulta:

- Utilizando la función `YEAR` de MySQL.
- Utilizando la función `DATE_FORMAT` de MySQL.
- Sin utilizar ninguna de las funciones anteriores.

9. Devuelve un listado con el código de pedido, código de cliente, fecha esperada y fecha de entrega de los pedidos que no han sido entregados a tiempo.

10. Devuelve un listado con el código de pedido, código de cliente, fecha esperada y fecha de entrega de los pedidos cuya fecha de entrega ha sido al menos dos días antes de la fecha esperada.

- Utilizando la función `ADDDATE` de MySQL.
- Utilizando la función `DATEDIFF` de MySQL.
- ¿Sería posible resolver esta consulta utilizando el operador de suma `+` o resta `-`?

11. Devuelve un listado de todos los pedidos que fueron **rechazados** en `2009`.

12. Devuelve un listado de todos los pedidos que han sido **entregados** en el mes de enero de cualquier año.

13. Devuelve un listado con todos los pagos que se realizaron en el año `2008` mediante `Paypal`. Ordene el resultado de mayor a menor.

14. Devuelve un listado con todas las formas de pago que aparecen en la tabla `pago`. Tenga en cuenta que no deben aparecer formas de pago repetidas.

15. Devuelve un listado con todos los productos que pertenecen a la gama `Ornamentales` y que tienen más de `100` unidades en stock. El listado deberá estar ordenado por su precio de venta, mostrando en primer lugar los de mayor precio.

16. Devuelve un listado con todos los clientes que sean de la ciudad de `Madrid` y cuyo representante de ventas tenga el código de empleado `11` o `30`.

1.4.5 Consultas multitabla (Composición interna)

Resuelva todas las consultas utilizando la sintaxis de `SQL1` y `SQL2`. Las consultas con sintaxis de `SQL2` se deben resolver con `INNER JOIN` y `NATURAL JOIN`.

1. Obtén un listado con el nombre de cada cliente y el nombre y apellido de su representante de ventas.

2. Muestra el nombre de los clientes que hayan realizado pagos junto con el nombre de sus representantes de ventas.

3. Muestra el nombre de los clientes que **no** hayan realizado pagos junto con el nombre de sus representantes de ventas.

4. Devuelve el nombre de los clientes que han hecho pagos y el nombre de sus representantes junto con la ciudad de la oficina a la que pertenece el representante.

5. Devuelve el nombre de los clientes que **no** hayan hecho pagos y el nombre de sus representantes junto con la ciudad de la oficina a la que pertenece el representante.
6. Lista la dirección de las oficinas que tengan clientes en `Fuenlabrada`.
7. Devuelve el nombre de los clientes y el nombre de sus representantes junto con la ciudad de la oficina a la que pertenece el representante.
8. Devuelve un listado con el nombre de los empleados junto con el nombre de sus jefes.
9. Devuelve un listado que muestre el nombre de cada empleados, el nombre de su jefe y el nombre del jefe de sus jefe.
10. Devuelve el nombre de los clientes a los que no se les ha entregado a tiempo un pedido.
11. Devuelve un listado de las diferentes gamas de producto que ha comprado cada cliente.

1.4.6 Consultas multitabla (Composición externa)

Resuelva todas las consultas utilizando las cláusulas `LEFT JOIN`, `RIGHT JOIN`, `NATURAL LEFT JOIN` y `NATURAL RIGHT JOIN`.

1. Devuelve un listado que muestre solamente los clientes que no han realizado ningún pago.
2. Devuelve un listado que muestre solamente los clientes que no han realizado ningún pedido.
3. Devuelve un listado que muestre los clientes que no han realizado ningún pago y los que no han realizado ningún pedido.
4. Devuelve un listado que muestre solamente los empleados que no tienen una oficina asociada.
5. Devuelve un listado que muestre solamente los empleados que no tienen un cliente asociado.
6. Devuelve un listado que muestre solamente los empleados que no tienen un cliente asociado junto con los datos de la oficina donde trabajan.
7. Devuelve un listado que muestre los empleados que no tienen una oficina asociada y los que no tienen un cliente asociado.

8. Devuelve un listado de los productos que nunca han aparecido en un pedido.
9. Devuelve un listado de los productos que nunca han aparecido en un pedido. El resultado debe mostrar el nombre, la descripción y la imagen del producto.
10. Devuelve las oficinas donde **no trabajan** ninguno de los empleados que hayan sido los representantes de ventas de algún cliente que haya realizado la compra de algún producto de la gama `Frutales`.
11. Devuelve un listado con los clientes que han realizado algún pedido pero no han realizado ningún pago.
12. Devuelve un listado con los datos de los empleados que no tienen clientes asociados y el nombre de su jefe asociado.

1.4.7 Consultas resumen

1. ¿Cuántos empleados hay en la compañía?
2. ¿Cuántos clientes tiene cada país?
3. ¿Cuál fue el pago medio en 2009?
4. ¿Cuántos pedidos hay en cada estado? Ordena el resultado de forma descendente por el número de pedidos.
5. Calcula el precio de venta del producto más caro y más barato en una misma consulta.
6. Calcula el número de clientes que tiene la empresa.
7. ¿Cuántos clientes existen con domicilio en la ciudad de Madrid?
8. ¿Calcula cuántos clientes tiene cada una de las ciudades que empiezan por `M`?
9. Devuelve el nombre de los representantes de ventas y el número de clientes al que atiende cada uno.
10. Calcula el número de clientes que no tiene asignado representante de ventas.
11. Calcula la fecha del primer y último pago realizado por cada uno de los clientes. El listado deberá mostrar el nombre y los apellidos de cada cliente.
12. Calcula el número de productos diferentes que hay en cada uno de los pedidos.

13. Calcula la suma de la cantidad total de todos los productos que aparecen en cada uno de los pedidos.

14. Devuelve un listado de los 20 productos más vendidos y el número total de unidades que se han vendido de cada uno. El listado deberá estar ordenado por el número total de unidades vendidas.

15. La facturación que ha tenido la empresa en toda la historia, indicando la base imponible, el IVA y el total facturado. La base imponible se calcula sumando el coste del producto por el número de unidades vendidas de la tabla `detalle_pedido`. El IVA es el 21 % de la base imponible, y el total la suma de los dos campos anteriores.

16. La misma información que en la pregunta anterior, pero agrupada por código de producto.

17. La misma información que en la pregunta anterior, pero agrupada por código de producto filtrada por los códigos que empiecen por `OR`.

18. Lista las ventas totales de los productos que hayan facturado más de 3000 euros. Se mostrará el nombre, unidades vendidas, total facturado y total facturado con impuestos (21% IVA).

19. Muestre la suma total de todos los pagos que se realizaron para cada uno de los años que aparecen en la tabla `pagos`.

1.4.8 Subconsultas

1.4.8.1 Con operadores básicos de comparación

1. Devuelve el nombre del cliente con mayor límite de crédito.

2. Devuelve el nombre del producto que tenga el precio de venta más caro.

3. Devuelve el nombre del producto del que se han vendido más unidades. (Tenga en cuenta que tendrá que calcular cuál es el número total de unidades que se han vendido de cada producto a partir de los datos de la tabla `detalle_pedido`)

4. Los clientes cuyo límite de crédito sea mayor que los pagos que haya realizado. (Sin utilizar `INNER JOIN`).

5. Devuelve el producto que más unidades tiene en stock.

6. Devuelve el producto que menos unidades tiene en stock.

7. Devuelve el nombre, los apellidos y el email de los empleados que están a cargo de **Alberto Soria**.

1.4.8.2 Subconsultas con ALL y ANY

8. Devuelve el nombre del cliente con mayor límite de crédito.
9. Devuelve el nombre del producto que tenga el precio de venta más caro.
10. Devuelve el producto que menos unidades tiene en stock.

1.4.8.3 Subconsultas con IN y NOT IN

11. Devuelve el nombre, apellido1 y cargo de los empleados que no representen a ningún cliente.
12. Devuelve un listado que muestre solamente los clientes que no han realizado ningún pago.
13. Devuelve un listado que muestre solamente los clientes que sí han realizado algún pago.
14. Devuelve un listado de los productos que nunca han aparecido en un pedido.
15. Devuelve el nombre, apellidos, puesto y teléfono de la oficina de aquellos empleados que no sean representante de ventas de ningún cliente.
16. Devuelve las oficinas donde **no trabajan** ninguno de los empleados que hayan sido los representantes de ventas de algún cliente que haya realizado la compra de algún producto de la gama `Frutales`.
17. Devuelve un listado con los clientes que han realizado algún pedido pero no han realizado ningún pago.

1.4.8.4 Subconsultas con EXISTS y NOT EXISTS

18. Devuelve un listado que muestre solamente los clientes que no han realizado ningún pago.
19. Devuelve un listado que muestre solamente los clientes que sí han realizado algún pago.
20. Devuelve un listado de los productos que nunca han aparecido en un pedido.
21. Devuelve un listado de los productos que han aparecido en un pedido alguna vez.

1.4.8.5 Subconsultas correlacionadas

1.4.9 Consultas variadas

1. Devuelve el listado de clientes indicando el nombre del cliente y cuántos pedidos ha realizado. Tenga en cuenta que pueden existir clientes que no han realizado ningún pedido.
2. Devuelve un listado con los nombres de los clientes y el total pagado por cada uno de ellos. Tenga en cuenta que pueden existir clientes que no han realizado ningún pago.
3. Devuelve el nombre de los clientes que hayan hecho pedidos en 2008 ordenados alfabéticamente de menor a mayor.
4. Devuelve el nombre del cliente, el nombre y primer apellido de su representante de ventas y el número de teléfono de la oficina del representante de ventas, de aquellos clientes que no hayan realizado ningún pago.
5. Devuelve el listado de clientes donde aparezca el nombre del cliente, el nombre y primer apellido de su representante de ventas y la ciudad donde está su oficina.
6. Devuelve el nombre, apellidos, puesto y teléfono de la oficina de aquellos empleados que no sean representante de ventas de ningún cliente.
7. Devuelve un listado indicando todas las ciudades donde hay oficinas y el número de empleados que tiene.

1.5 Universidad (Tipo A)

1.5.1 Modelo entidad/relación

1.5.2 Base de datos para MySQL

```sql
DROP DATABASE IF EXISTS universidad;
CREATE DATABASE universidad CHARACTER SET utf8mb4;
USE universidad;

CREATE TABLE departamento (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(50) NOT NULL
);

CREATE TABLE persona (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    nif VARCHAR(9) UNIQUE,
    nombre VARCHAR(25) NOT NULL,
    apellido1 VARCHAR(50) NOT NULL,
    apellido2 VARCHAR(50),
    ciudad VARCHAR(25) NOT NULL,
    direccion VARCHAR(50) NOT NULL,
    telefono VARCHAR(9),
    fecha_nacimiento DATE NOT NULL,
    sexo ENUM('H', 'M') NOT NULL,
    tipo ENUM('profesor', 'alumno') NOT NULL
);

CREATE TABLE profesor (
    id_profesor INT UNSIGNED PRIMARY KEY,
```

```
    id_departamento INT UNSIGNED NOT NULL,
    FOREIGN KEY (id_profesor) REFERENCES persona(id),
    FOREIGN KEY (id_departamento) REFERENCES departamento(id)
);

 CREATE TABLE grado (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL
);

CREATE TABLE asignatura (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    creditos FLOAT UNSIGNED NOT NULL,
    tipo ENUM('básica', 'obligatoria', 'optativa') NOT NULL,
    curso TINYINT UNSIGNED NOT NULL,
    cuatrimestre TINYINT UNSIGNED NOT NULL,
    id_profesor INT UNSIGNED,
    id_grado INT UNSIGNED NOT NULL,
    FOREIGN KEY(id_profesor) REFERENCES profesor(id_profesor),
    FOREIGN KEY(id_grado) REFERENCES grado(id)
);

CREATE TABLE curso_escolar (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    anyo_inicio YEAR NOT NULL,
    anyo_fin YEAR NOT NULL
);

CREATE TABLE alumno_se_matricula_asignatura (
    id_alumno INT UNSIGNED NOT NULL,
    id_asignatura INT UNSIGNED NOT NULL,
    id_curso_escolar INT UNSIGNED NOT NULL,
    PRIMARY KEY (id_alumno, id_asignatura, id_curso_escolar),
    FOREIGN KEY (id_alumno) REFERENCES persona(id),
    FOREIGN KEY (id_asignatura) REFERENCES asignatura(id),
    FOREIGN KEY (id_curso_escolar) REFERENCES curso_escolar(id)
);
```

1.5.3 Datos

Acceder al *script* SQL para la creación de la base de datos y la inserción de datos: **universidad_a.sql**.

1.5.4 Consultas sobre una tabla

1. Devuelve un listado con el primer apellido, segundo apellido y el nombre de todos los alumnos. El listado deberá estar ordenado alfabéticamente de menor a mayor por el primer apellido, segundo apellido y nombre.
2. Averigua el nombre y los dos apellidos de los alumnos que **no** han dado de alta su número de teléfono en la base de datos.
3. Devuelve el listado de los alumnos que nacieron en `1999`.
4. Devuelve el listado de profesores que **no** han dado de alta su número de teléfono en la base de datos y además su nif termina en `K`.
5. Devuelve el listado de las asignaturas que se imparten en el primer cuatrimestre, en el tercer curso del grado que tiene el identificador `7`.

1.5.5 Consultas multitabla (Composición interna)

1. Devuelve un listado con los datos de todas las **alumnas** que se han matriculado alguna vez en el `Grado en Ingeniería Informática (Plan 2015)`.
2. Devuelve un listado con todas las asignaturas ofertadas en el `Grado en Ingeniería Informática (Plan 2015)`.
3. Devuelve un listado de los profesores junto con el nombre del departamento al que están vinculados. El listado debe devolver cuatro columnas, primer apellido, segundo apellido, nombre y nombre del departamento. El resultado estará ordenado alfabéticamente de menor a mayor por los apellidos y el nombre.
4. Devuelve un listado con el nombre de las asignaturas, año de inicio y año de fin del curso escolar del alumno con nif `26902806M`.
5. Devuelve un listado con el nombre de todos los departamentos que tienen profesores que imparten alguna asignatura en el `Grado en Ingeniería Informática (Plan 2015)`.
6. Devuelve un listado con todos los alumnos que se han matriculado en alguna asignatura durante el curso escolar 2018/2019.

1.5.6 Consultas multitabla (Composición externa)

Resuelva todas las consultas utilizando las cláusulas `LEFT JOIN` y `RIGHT JOIN`.

1. Devuelve un listado con los nombres de **todos** los profesores y los departamentos que tienen vinculados. El listado también debe mostrar aquellos profesores que no

tienen ningún departamento asociado. El listado debe devolver cuatro columnas, nombre del departamento, primer apellido, segundo apellido y nombre del profesor. El resultado estará ordenado alfabéticamente de menor a mayor por el nombre del departamento, apellidos y el nombre.

2. Devuelve un listado con los profesores que no están asociados a un departamento.
3. Devuelve un listado con los departamentos que no tienen profesores asociados.
4. Devuelve un listado con los profesores que no imparten ninguna asignatura.
5. Devuelve un listado con las asignaturas que no tienen un profesor asignado.
6. Devuelve un listado con todos los departamentos que tienen alguna asignatura que no se haya impartido en ningún curso escolar. El resultado debe mostrar el nombre del departamento y el nombre de la asignatura que no se haya impartido nunca.

1.5.7 Consultas resumen

1. Devuelve el número total de **alumnas** que hay.
2. Calcula cuántos alumnos nacieron en `1999`.
3. Calcula cuántos profesores hay en cada departamento. El resultado sólo debe mostrar dos columnas, una con el nombre del departamento y otra con el número de profesores que hay en ese departamento. El resultado sólo debe incluir los departamentos que tienen profesores asociados y deberá estar ordenado de mayor a menor por el número de profesores.
4. Devuelve un listado con todos los departamentos y el número de profesores que hay en cada uno de ellos. Tenga en cuenta que pueden existir departamentos que no tienen profesores asociados. Estos departamentos también tienen que aparecer en el listado.
5. Devuelve un listado con el nombre de todos los grados existentes en la base de datos y el número de asignaturas que tiene cada uno. Tenga en cuenta que pueden existir grados que no tienen asignaturas asociadas. Estos grados también tienen que aparecer en el listado. El resultado deberá estar ordenado de mayor a menor por el número de asignaturas.
6. Devuelve un listado con el nombre de todos los grados existentes en la base de datos y el número de asignaturas que tiene cada uno, de los grados que tengan más de `40` asignaturas asociadas.

7. Devuelve un listado que muestre el nombre de los grados y la suma del número total de créditos que hay para cada tipo de asignatura. El resultado debe tener tres columnas: nombre del grado, tipo de asignatura y la suma de los créditos de todas las asignaturas que hay de ese tipo. Ordene el resultado de mayor a menor por el número total de crédidos.

8. Devuelve un listado que muestre cuántos alumnos se han matriculado de alguna asignatura en cada uno de los cursos escolares. El resultado deberá mostrar dos columnas, una columna con el año de inicio del curso escolar y otra con el número de alumnos matriculados.

9. Devuelve un listado con el número de asignaturas que imparte cada profesor. El listado debe tener en cuenta aquellos profesores que no imparten ninguna asignatura. El resultado mostrará cinco columnas: id, nombre, primer apellido, segundo apellido y número de asignaturas. El resultado estará ordenado de mayor a menor por el número de asignaturas.

1.5.8 Subconsultas

1. Devuelve todos los datos del alumno más joven.

2. Devuelve un listado con los profesores que no están asociados a un departamento.

3. Devuelve un listado con los departamentos que no tienen profesores asociados.

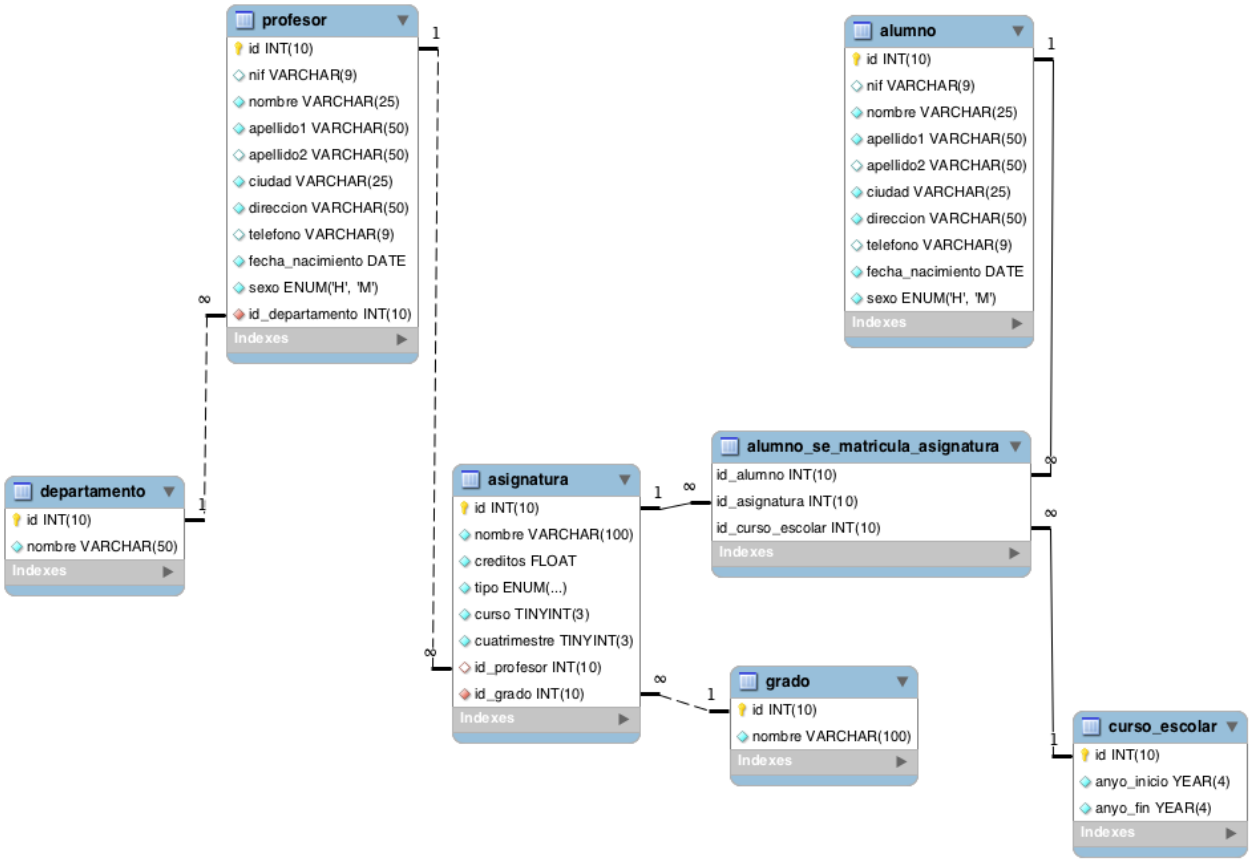
4. Devuelve un listado con los profesores que tienen un departamento asociado y que no imparten ninguna asignatura.

5. Devuelve un listado con las asignaturas que no tienen un profesor asignado.

6. Devuelve un listado con todos los departamentos que no han impartido asignaturas en ningún curso escolar.

1.6 Universidad (Tipo B)

1.6.1 Modelo entidad/relación



1.6.2 Base de datos para MySQL

```sql
DROP DATABASE IF EXISTS universidad;
CREATE DATABASE universidad CHARACTER SET utf8mb4;
USE universidad;

CREATE TABLE departamento (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(50) NOT NULL
);

CREATE TABLE alumno (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    nif VARCHAR(9) UNIQUE,
    nombre VARCHAR(25) NOT NULL,
    apellido1 VARCHAR(50) NOT NULL,
    apellido2 VARCHAR(50),
    ciudad VARCHAR(25) NOT NULL,
    direccion VARCHAR(50) NOT NULL,
    telefono VARCHAR(9),
    fecha_nacimiento DATE NOT NULL,
    sexo ENUM('H', 'M') NOT NULL
);

CREATE TABLE profesor (
```

```
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    nif VARCHAR(9) UNIQUE,
    nombre VARCHAR(25) NOT NULL,
    apellido1 VARCHAR(50) NOT NULL,
    apellido2 VARCHAR(50),
    ciudad VARCHAR(25) NOT NULL,
    direccion VARCHAR(50) NOT NULL,
    telefono VARCHAR(9),
    fecha_nacimiento DATE NOT NULL,
    sexo ENUM('H', 'M') NOT NULL,
    id_departamento INT UNSIGNED NOT NULL,
    FOREIGN KEY (id_departamento) REFERENCES departamento(id)
);

 CREATE TABLE grado (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL
);

CREATE TABLE asignatura (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    creditos FLOAT UNSIGNED NOT NULL,
    tipo ENUM('básica', 'obligatoria', 'optativa') NOT NULL,
    curso TINYINT UNSIGNED NOT NULL,
    cuatrimestre TINYINT UNSIGNED NOT NULL,
    id_profesor INT UNSIGNED,
    id_grado INT UNSIGNED NOT NULL,
    FOREIGN KEY(id_profesor) REFERENCES profesor(id),
    FOREIGN KEY(id_grado) REFERENCES grado(id)
);

CREATE TABLE curso_escolar (
    id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    anyo_inicio YEAR NOT NULL,
    anyo_fin YEAR NOT NULL
);

CREATE TABLE alumno_se_matricula_asignatura (
    id_alumno INT UNSIGNED NOT NULL,
    id_asignatura INT UNSIGNED NOT NULL,
    id_curso_escolar INT UNSIGNED NOT NULL,
    PRIMARY KEY (id_alumno, id_asignatura, id_curso_escolar),
    FOREIGN KEY (id_alumno) REFERENCES alumno(id),
    FOREIGN KEY (id_asignatura) REFERENCES asignatura(id),
    FOREIGN KEY (id_curso_escolar) REFERENCES curso_escolar(id)
);
```

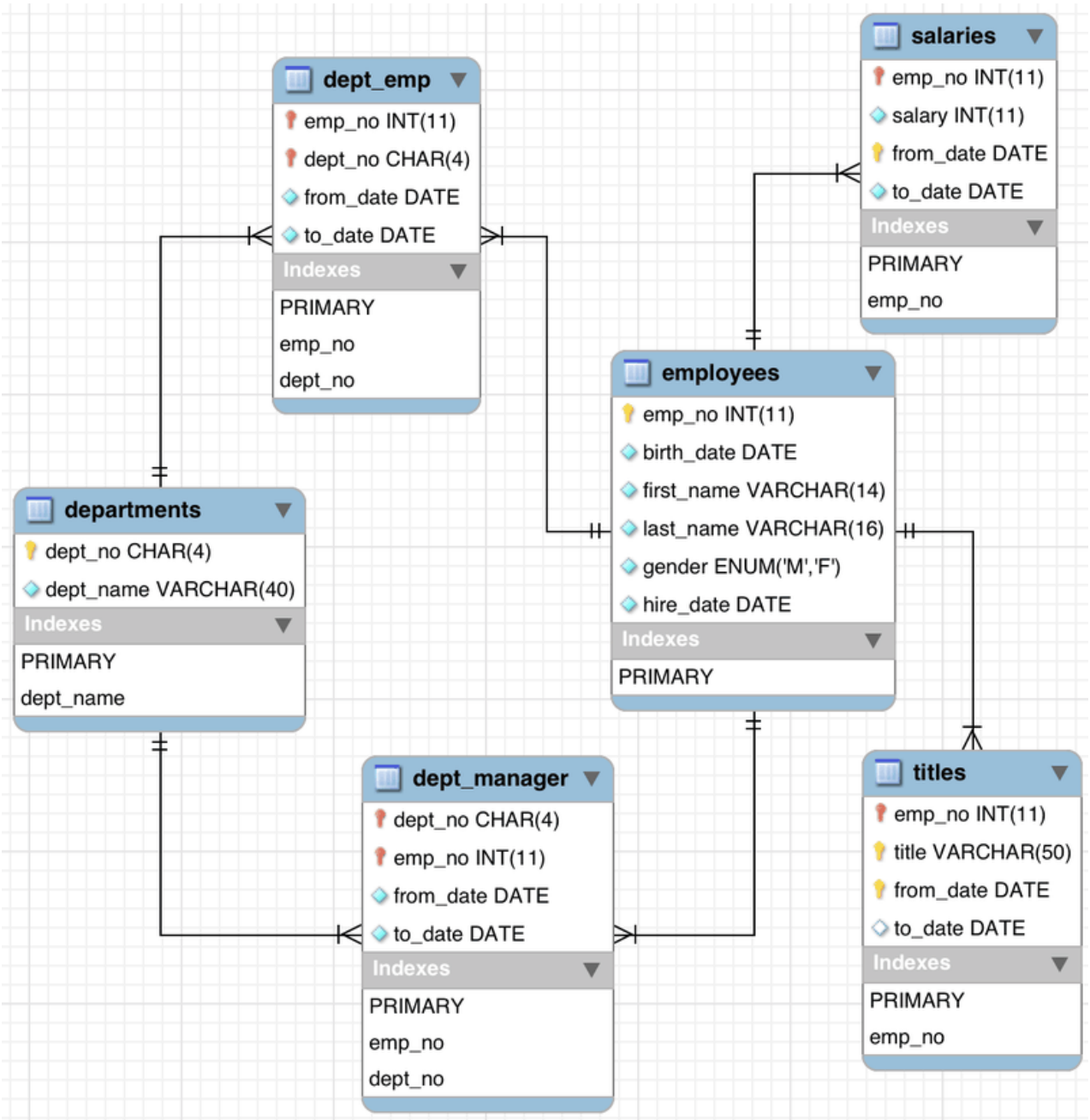
1.6.3 Datos

Acceder al *script* SQL para la creación de la base de datos y la inserción de datos: **universidad_b.sql**.

1.7 Employees

La base de datos `Employees` está disponible en la página web oficial de MySQL. Se trata de una base de datos creada por **Patrick Crews** y **Giuseppe Maxia**.

1.7.1 Modelo entidad/relación



1.7.2 Base de datos para MySQL

La base de datos está disponible en GitHub.

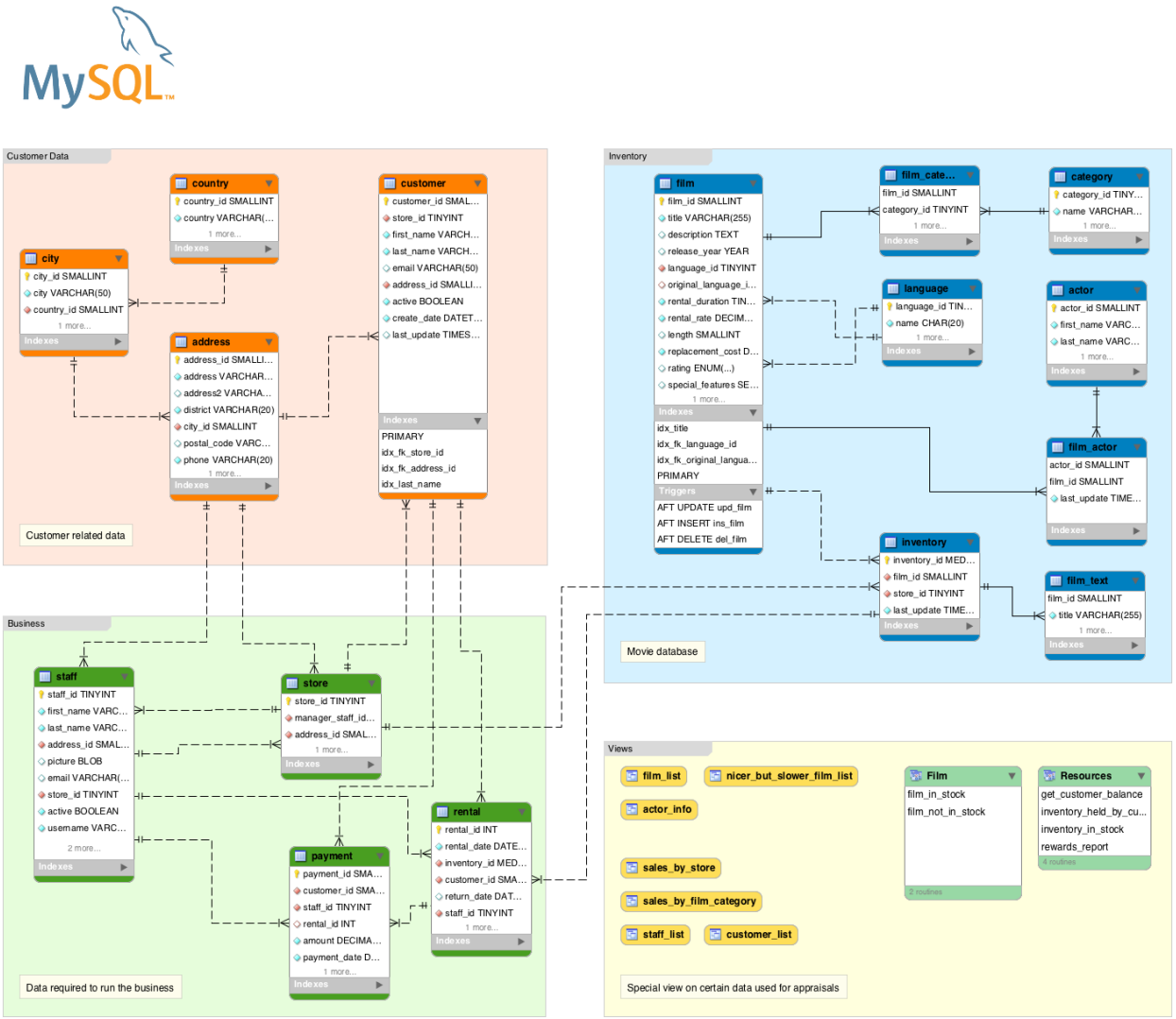
1.7.3 Importar la base datos en MySQL

Cómo importar la base de datos en MySQL.

1.8 Sakila

La base de datos `Sakila` está disponible en la página web oficial de MySQL. Se trata de una base de datos creada por **Mike Hillyers**.

1.8.1 Modelo entidad/relación



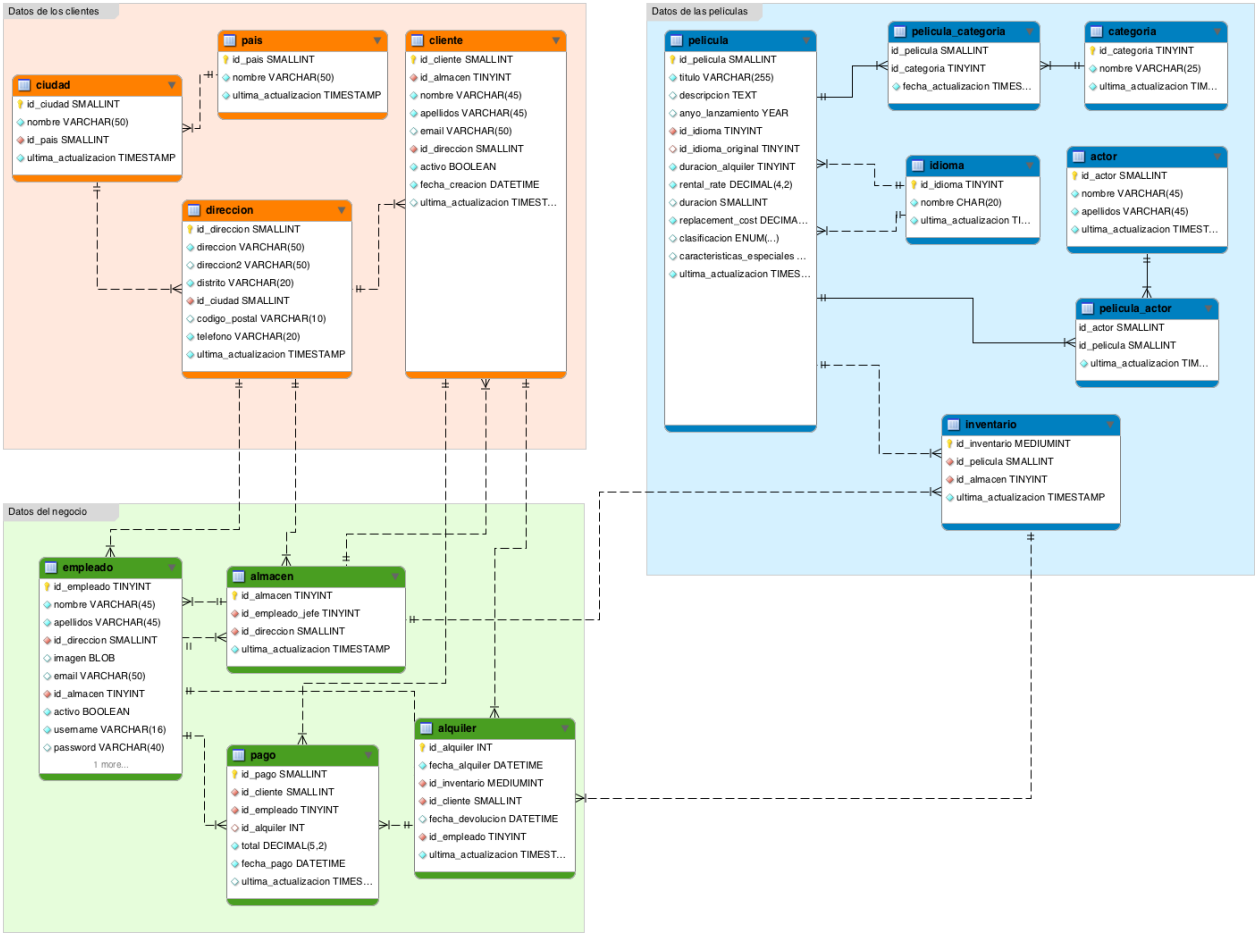
1.8.2 Base de datos para MySQL

La base de datos está disponible en la web oficial de MySQL.

1.9 Sakila (En Español)

Se trata de la base de datos `Sakila` creada por **Mike Hillyers**, con los nombres de las tablas y columnas en español.

1.9.1 Modelo entidad/relación



1.9.2 Base de datos para MySQL

- Esquema.
- Datos.

2 SQL Playground

Puedes practicar todas las consultas que aparecen en esta web en SQL Playground:

- http://sql-playground.com.

SQL Playground es un recurso educativo diseñado para aprender a realizar consultas SQL. La versión actual del proyecto utiliza el SGBD MySQL.

El proyecto fue desarrollado para el alumnado de los módulos **Gestión de Bases de Datos** y **Bases de Datos** de los CFGS del **IES Celia Viñas (Almería)** durante el **curso 2017/2018**.

3 Referencias

- Wikibook SQL Exercises.
- Tutorial SQL de w3resource.
- **Bases de Datos**. 2ª Edición. Grupo editorial Garceta. Iván López Montalbán, Manuel de Castro Vázquez y John Ospino Rivas.
- MySQL Sample Databases. Chua Hock-Chuan.
- SQL Playground.

4 Licencia