Tarragona Impulsa - Tech Talent

Programación con Java

TAREA 16

Alumno: Alejandro Rodríguez Jiménez
Profesor: Marc Esteve García
Curso 2024

Índice

Introducción

Durante esta práctica aprenderemos mucho mejor la sintaxis de la consulta **SELECT** añadiendo funciones como **COUNT(), SUM()** o **AVG()** y formas para ordenar los datos con **ORDER BY** y **GROUP BY**. Después de profundizar en esta sintaxis junto con **WHERE** o **HAVING** aprenderemos a crear subconsultas y a utilizar la sentencia **JOIN** conjunto varios tipos de este para utilizar unos datos u otros (**INNER**, **LEFT**, **RIGHT**) para combinar ninguna, una o varias columnas de una tabla con otra con un dato relacionado que tienen.

Para la ejecución de las consultas y demás lo primero que tendremos que hacer para cada ejercicio será insertar un mínimo de 10 tuplas para cada tabla.

Ejercicio 01:

```sql
-- INSERCIONES
INSERT INTO fabricantes VALUES
    (0, 'Juan Magan'),
    (1, 'Antonio Sobera'),
    (2, 'Prince Royce'),
    (3, 'Javier Ibarra'),
    (4, 'Luciano Pavarotti'),
    (5, 'Ramon Ramirez'),
    (6, 'Pedro Ruibal'),
    (7, 'Jose Alvarez'),
    (8, 'Joaquin Sanchez'),
    (9, 'Antonio Banderas');
```

```sql
INSERT INTO articulos VALUES
    (0, 'Pollo', 12, 2),
    (1, 'Agua', 200, 1),
    (2, 'Helado', 12, 0),
    (3, 'Jamon', 300, 8),
    (4, 'Almidon', 15, 6),
    (5, 'Aceite', 74, 4),
    (6, 'Gasolina', 750,5),
    (7, 'Peine', 10, 3),
    (8, 'Harina', 400, 9),
    (9, 'Queso', 8, 7);
```

1. Obtener los nombres de los productos de la tienda
   CONSULTA: SELECT nombre FROM articulos;
2. Obtener los nombres y los precios de los productos de la tienda
   CONSULTA: SELECT nombre,precio FROM articulos;
3. Obtener los nombres de los productos cuyo precio sea menor o igual a 200€
   CONSULTA: SELECT nombre,precio FROM articulos WHERE precio<=200;
4. Obtener todos los datos de los artículos cuyo precio esté entre 60€ y 120€
   CONSULTA: SELECT nombre,precio FROM articulos WHERE precio BETWEEN 60 AND 120;
5. Obtener el nombre y el precio en pesetas de los productos de la tienda
   CONSULTA: SELECT nombre,precio*166.386 FROM articulos;

1

	nombre
1	Pollo
2	Agua
3	Helado
4	Jamon
5	Almidon
6	Aceite
7	Gasolina
8	Peine
9	Impresora Laser
10	Queso
11	Portatil
12	Papel
13	Rinonera
14	Altavoces

2

	nombre	precio
1	Pollo	11
2	Agua	170
3	Helado	11
4	Jamon	260
5	Almidon	14
6	Aceite	67
7	Gasolina	665
8	Peine	9
9	Impresora Laser	350
10	Queso	7
11	Portatil	113
12	Papel	14
13	Rinonera	20
14	Altavoces	70

3

	nombre	precio
1	Pollo	11
2	Agua	170
3	Helado	11
4	Almidon	14
5	Aceite	67
6	Peine	9
7	Queso	7
8	Portatil	113
9	Papel	14
10	Rinonera	20
11	Altavoces	70

4

	nombre	precio
1	Aceite	67
2	Portatil	113
3	Altavoces	70

5

	nombre	precio*166.386
1	Pollo	1.830,246
2	Agua	28.285,62
3	Helado	1.830,246
4	Jamon	43.260,36
5	Almidon	2.329,404
6	Aceite	11.147,862
7	Gasolina	110.646,69
8	Peine	1.497,474
9	Impresora Laser	58.235,1
10	Queso	1.164,702
11	Portatil	18.801,618
12	Papel	2.329,404
13	Rinonera	3.327,72
14	Altavoces	11.647,02

6. Seleccionar el precio medio de todos los productos

   CONSULTA: SELECT AVG(precio) FROM articulos;

7. Obtener el precio medio de los artículos cuyo código de fabricante sea 2

   CONSULTA: SELECT AVG(precio) FROM articulos WHERE fabricante=2;

8. Obtener el número de artículos cuyo precio sea mayor o igual a 180€

   CONSULTA: SELECT COUNT(*) FROM articulos WHERE precio>=180;

9. Obtener el nombre y precio de los artículos cuyo precio sea mayor o igual a 180€ y ordenarlos descendentemente por precio, y luego ascendentemente por nombre

   CONSULTA: SELECT nombre,precio FROM articulos WHERE precio>=180 ORDER BY precio DESC, nombre ASC;

```
AVG(precio) AS 'Precio medio'
```

```
COUNT(*) AS 'Conteo productos'
```

6

	Precio medio
1	127,2143

7

	Precio medio
1	40,5

8

	Conteo productos
1	3

9

	nombre	precio
1	Gasolina	665
2	Impresora Laser	350
3	Jamon	260

10. Obtener un listado completo de los artículos, incluyendo por cada artículo los datos del artículo y de su fabricante

    CONSULTA: SELECT * FROM articulos ar INNER JOIN fabricantes fa ON ar.fabricante=fa.codigo;

11. Obtener un listado de artículos, incluyendo el nombre del artículo , su precio, y el nombre de su fabricante

    CONSULTA: SELECT ar.nombre, precio, fa.nombre FROM articulos ar INNER JOIN fabricantes fa ON ar.fabricante=fa.codigo;

10

	codigo	nombre	precio	fabricante	codigo	nombre
1	0	Pollo	11	2	2	Prince Royce
2	1	Agua	170	1	1	Antonio Sobera
3	2	Helado	11	0	0	Juan Magan
4	3	Jamon	260	8	8	Joaquin Sanchez
5	4	Almidon	14	6	6	Pedro Ruibal
6	5	Aceite	67	4	4	Luciano Pavarotti
7	6	Gasolina	665	5	5	Ramon Ramirez
8	7	Peine	9	3	3	Javier Ibarra
9	8	Impresora Laser	350	9	9	Antonio Banderas
10	9	Queso	7	7	7	Jose Alvarez
11	10	Portatil	113	6	6	Pedro Ruibal
12	11	Papel	14	8	8	Joaquin Sanchez
13	12	Rinonera	20	7	7	Jose Alvarez
14	13	Altavoces	70	2	2	Prince Royce

11

	nombre	precio	nombre
1	Pollo	11	Prince Royce
2	Agua	170	Antonio Sobera
3	Helado	11	Juan Magan
4	Jamon	260	Joaquin Sanchez
5	Almidon	14	Pedro Ruibal
6	Aceite	67	Luciano Pavarotti
7	Gasolina	665	Ramon Ramirez
8	Peine	9	Javier Ibarra
9	Impresora Laser	350	Antonio Banderas
10	Queso	7	Jose Alvarez
11	Portatil	113	Pedro Ruibal
12	Papel	14	Joaquin Sanchez
13	Rinonera	20	Jose Alvarez
14	Altavoces	70	Prince Royce

12. Obtener el precio medio de los productos de cada fabricante, mostrando solo los códigos de fabricante

    CONSULTA: SELECT ar.codigo, AVG(precio) FROM articulos ar INNER JOIN fabricantes fa ON ar.fabricante=fa.codigo GROUP BY fabricante;

13. Obtener el precio medio de los productos de cada fabricante, mostrando el nombre del fabricante

    CONSULTA: SELECT ar.codigo, fa.nombre, AVG(precio) FROM articulos ar INNER JOIN fabricantes fa ON ar.fabricante=fa.codigo GROUP BY fabricante;

14. Obtener los nombre de los fabricantes que ofrezcan productos cuyo precio medio sea mayor o igual a 150€

    CONSULTA: SELECT fa.nombre, precio FROM articulos ar INNER JOIN fabricantes fa ON ar.fabricante=fa.codigo GROUP BY fabricante HAVING AVG(precio)>=150;

```
AVG(precio) AS 'Precio medio'
```

12

	codigo	Precio medio
1	2	11,0
2	1	170,0
3	0	40,5
4	7	9,0
5	5	67,0
6	6	665,0
7	4	63,5
8	9	13,5
9	3	137,0
10	8	350,0

13

	codigo	nombre	Precio medio
1	2	Juan Magan	11,0
2	1	Antonio Sobera	170,0
3	0	Prince Royce	40,5
4	7	Javier Ibarra	9,0
5	5	Luciano Pavarotti	67,0
6	6	Ramon Ramirez	665,0
7	4	Pedro Ruibal	63,5
8	9	Jose Alvarez	13,5
9	3	Joaquin Sanchez	137,0
10	8	Antonio Banderas	350,0

14

	nombre	precio
1	Antonio Sobera	170
2	Ramon Ramirez	665
3	Antonio Banderas	350

15. Obtener el nombre y precio del artículo más barato

    CONSULTA: SELECT nombre, precio FROM articulos ORDER BY precio ASC LIMIT 1;

16. Obtener una lista con el nombre y el precio de los artículos más caros de cada proveedor, con su nombre

    CONSULTA: SELECT fa.nombre, ar.nombre, precio FROM articulos ar INNER JOIN fabricantes fa ON ar.fabricante=fa.codigo WHERE precio IN (SELECT MAX(precio) FROM articulos GROUP BY fabricante);

15

	nombre	precio
1	Queso	7

16

	nombre	nombre	precio
1	Juan Magan	Helado	11
2	Antonio Sobera	Agua	170
3	Prince Royce	Pollo	11
4	Prince Royce	Altavoces	70
5	Javier Ibarra	Peine	9
6	Luciano Pavarotti	Aceite	67
7	Ramon Ramirez	Gasolina	665
8	Pedro Ruibal	Portatil	113
9	Jose Alvarez	Rinonera	20
10	Joaquin Sanchez	Jamon	260

17. Añadir un nuevo producto: Altavoces de 70€, fabricante:2

    CONSULTA: INSERT INTO articulos (codigo, nombre, precio, fabricante) VALUES (13, 'Altavoces', 70, 2);

18. Cambiar el nombre del producto 8 a 'Impresora Laser'

    CONSULTA: UPDATE articulos SET nombre='Impresora Laser' WHERE codigo=8;

19. Aplicar un descuento del 10€ a todos los productos

    CONSULTA: UPDATE articulos SET precio=precio*0.9;

20. Aplicar un descuento de 10€ a los productos cuyo precio sea mayor o igual a 120€

    CONSULTA: UPDATE articulos SET precio=precio-10 WHERE precio>=120;

Sin respuesta visual [17, 18, 19 & 20]

Antes

	codigo	nombre	precio	fabricante
1	0	Pollo	12	2
2	1	Agua	200	1
3	2	Helado	12	0
4	3	Jamon	300	8
5	4	Almidon	15	6
6	5	Aceite	74	4
7	6	Gasolina	750	5
8	7	Peine	10	3
9	8	Harina	400	9
10	9	Queso	8	7
11	10	Portatil	113	6
12	11	Papel	14	8
13	12	Rinonera	20	7

Después

	codigo	nombre	precio	fabricante
1	0	Pollo	11	2
2	1	Agua	170	1
3	2	Helado	11	0
4	3	Jamon	260	8
5	4	Almidon	14	6
6	5	Aceite	67	4
7	6	Gasolina	665	5
8	7	Peine	9	3
9	8	Impresora Laser	350	9
10	9	Queso	7	7
11	10	Portatil	102	6
12	11	Papel	13	8
13	12	Rinonera	18	7
14	13	Altavoces	63	2

Ejercicio 02:

```sql
-- INSERCIONES
INSERT INTO departamentos (codigo, nombre, presupuesto) VALUES
    (1, 'Ventas', 50000),
    (2, 'Marketing', 60000),
    (3, 'Recursos Humanos', 45000),
    (4, 'Finanzas', 70000),
    (5, 'Producción', 80000),
    (14, 'Informatica', 65000),
    (7, 'Soporte Tecnico', 55000),
    (37, 'Gramatica', 48000),
    (77, 'Investigacion', 60000),
    (55, 'Arquitectura', 90000),
    (21, 'Administracion', 70000);
```

```sql
INSERT INTO empleados (dni, nombre, apellidos, departamento) VALUES
    ('12345678', 'Juan', 'Gonzalez', 1),
    ('23456789', 'María', 'Perez', 21),
    ('34567890', 'Pedro', 'Rodriguez', 77),
    ('45678901', 'Ana', 'Lopez', 2),
    ('56789012', 'David', 'Lopez', 14),
    ('67890123', 'Laura', 'Diaz', 3),
    ('78901234', 'Carlos', 'Rodriguez', 4),
    ('89012345', 'Sofía', 'Hernandez', 7),
    ('90123456', 'Elena', 'Diaz', 5),
    ('01234567', 'Javier', 'Fernandez', 37),
    ('11234567', 'Luis', 'Gómez', 77),
    ('22345678', 'Ana', 'Martínez', 77),
    ('33456789', 'Mario', 'Sánchez', 4);
```

1. Obtener los apellidos de los empleados

   CONSULTA: SELECT apellidos FROM empleados;

2. Obtener los apellidos de los empleados sin restricciones

   CONSULTA: SELECT apellidos FROM empleados GROUP BY apellidos;

3. Obtener todos los datos de los empleados que se apellidan 'Lopez'

   CONSULTA: SELECT * FROM empleados WHERE apellidos='Lopez';

4. Obtener todos los datos de los empleados que se apellidan 'Lopez' o 'Perez'

   CONSULTA: SELECT * FROM empleados WHERE apellidos='Lopez' OR apellidos='Perez';

5. Obtener todos los datos de los empleados que trabajan para el departamento 14

   CONSULTA: SELECT * FROM empleados WHERE departamento=14;

6. Obtener todos los datos de los empleados que trabajan para el departamento 37 o 77

   CONSULTA: SELECT * FROM empleados WHERE departamento IN (37, 77);

7. Obtener todos los datos de los empleados cuyo apellido comience por 'P'

   CONSULTA: SELECT * FROM empleados WHERE apellidos LIKE 'P%';

1

	apellidos
1	Fernandez
2	Gonzalez
3	Gómez
4	Perez
5	Martínez
6	Rodriguez
7	Sánchez
8	Lopez
9	Lopez
10	Diaz
11	Rodriguez
12	Hernandez
13	Vázquez
14	Diaz

2

	apellidos
1	Diaz
2	Fernandez
3	Gómez
4	Gonzalez
5	Hernandez
6	Lopez
7	Martínez
8	Perez
9	Rodriguez
10	Sánchez
11	Vázquez

3

	dni	nombre	apellidos	departamento
1	45678901	Ana	Lopez	2
2	56789012	David	Lopez	14

4

	dni	nombre	apellidos	departamento
1	23456789	María	Perez	21
2	45678901	Ana	Lopez	2
3	56789012	David	Lopez	14

5

	dni	nombre	apellidos	departamento
1	56789012	David	Lopez	14

6

	dni	nombre	apellidos	departamento
1	01234567	Javier	Fernandez	37
2	13345678	Luis	Gómez	77
3	24456789	Ana	Martínez	77
4	34567890	Pedro	Rodriguez	77

7

	dni	nombre	apellidos	departamento
1	23456789	María	Perez	21

8. Obtener el presupuesto total de todos los departamentos

   CONSULTA: SELECT SUM(presupuesto) FROM departamentos;

9. Obtener el número de empleados en cada departamento

   CONSULTA: SELECT COUNT(departamento), dep.nombre FROM empleados em INNER JOIN departamentos dep ON em.departamento=dep.codigo GROUP BY departamento;

```
SUM(presupuesto) AS 'Suma presupuesto'
```

```
COUNT(departamento) AS 'Conteo departamentos'
```

8

	Suma presupuesto
1	678.700

9

	Conteo departamentos	nombre
1	1	Ventas
2	1	Marketing
3	1	Recursos Humanos
4	2	Finanzas
5	1	Producción
6	1	Soporte Tecnico
7	1	Calidad
8	1	Informatica
9	1	Administracion
10	1	Gramatica
11	3	Investigacion

10. Obtener un listado completo de empleados, incluyendo por cada empleado los datos del empleado y de su departamento

    CONSULTA: SELECT em.*, dep.* FROM empleados em INNER JOIN departamentos dep ON em.departamento=dep.codigo;

11. Obtener un listado completo de empleados, incluyendo el nombre y apellidos del empleado junto al nombre y presupuesto de su departamento

    CONSULTA: SELECT em.nombre, apellidos, dep.nombre, presupuesto FROM empleados em INNER JOIN departamentos dep ON em.departamento=dep.codigo;

10

	dni	nombre	apellidos	departamento	codigo	nombre	presupuesto
1	01234567	Javier	Fernandez	37	37	Gramatica	43.200
2	12345678	Juan	Gonzalez	1	1	Ventas	45.000
3	13345678	Luis	Gómez	77	77	Investigacion	54.000
4	23456789	María	Perez	21	21	Administracion	63.000
5	24456789	Ana	Martínez	77	77	Investigacion	54.000
6	34567890	Pedro	Rodriguez	77	77	Investigacion	54.000
7	35567890	Mario	Sánchez	4	4	Finanzas	63.000
8	45678901	Ana	Lopez	2	2	Marketing	54.000
9	56789012	David	Lopez	14	14	Informatica	58.500
10	67890123	Laura	Diaz	3	3	Recursos Humanos	40.500
11	78901234	Carlos	Rodriguez	4	4	Finanzas	63.000
12	89012345	Sofía	Hernandez	7	7	Soporte Tecnico	49.500
13	89267109	Esther	Vázquez	11	11	Calidad	36.000
14	90123456	Elena	Diaz	5	5	Producción	72.000

11

	nombre	apellidos	nombre	presupuesto
1	Javier	Fernandez	Gramatica	43.200
2	Juan	Gonzalez	Ventas	45.000
3	Luis	Gómez	Investigacion	54.000
4	María	Perez	Administracion	63.000
5	Ana	Martínez	Investigacion	54.000
6	Pedro	Rodriguez	Investigacion	54.000
7	Mario	Sánchez	Finanzas	63.000
8	Ana	Lopez	Marketing	54.000
9	David	Lopez	Informatica	58.500
10	Laura	Diaz	Recursos Humanos	40.500
11	Carlos	Rodriguez	Finanzas	63.000
12	Sofía	Hernandez	Soporte Tecnico	49.500
13	Esther	Vázquez	Calidad	36.000
14	Elena	Diaz	Producción	72.000

12. Obtener los nombres y apellidos de los empleados que trabajen en departamentos cuyo presupuesto sea mayor de 60000€

    CONSULTA: SELECT em.nombre, apellidos FROM empleados em INNER JOIN departamentos dep ON em.departamento=dep.codigo WHERE presupuesto>=60000;

13. Obtener los datos de los departamentos cuyo presupuesto es superior al presupuesto medio de todos los departamentos

    CONSULTA: SELECT * FROM departamentos WHERE presupuesto > (SELECT AVG(presupuesto) FROM departamentos);

14. Obtener los nombres de los departamentos que tienen más de dos empleados

    CONSULTA: SELECT dep.nombre FROM empleados em INNER JOIN departamentos dep ON em.departamento=dep.codigo GROUP BY departamento HAVING COUNT(departamento) > 2;

12

	nombre	apellidos
1	María	Perez
2	Mario	Sánchez
3	Carlos	Rodriguez
4	Elena	Diaz

13

	codigo	nombre	presupuesto
1	4	Finanzas	63.000
2	5	Producción	72.000
3	14	Informatica	58.500
4	21	Administracion	63.000
5	55	Arquitectura	100.000

14

	nombre
1	Investigacion

Sin respuesta visual [15, 16, 17, 18, 19 & 20]

Antes

	nombre	presupuesto
1	Ventas	50.000
2	Marketing	60.000
3	Recursos Humanos	45.000
4	Finanzas	70.000
5	Producción	80.000
6	Soporte Tecnico	55.000
7	Informatica	65.000
8	Administracion	70.000
9	Gramatica	48.000
10	Arquitectura	90.000
11	Investigacion	60.000

	dni	nombre	apellidos	departamento
1	01234567	Javier	Fernandez	37
2	11234567	Luis	Gómez	77
3	12345678	Juan	Gonzalez	1
4	22345678	Ana	Martínez	77
5	23456789	María	Perez	21
6	33456789	Mario	Sánchez	4
7	34567890	Pedro	Rodriguez	77
8	45678901	Ana	Lopez	2
9	56789012	David	Lopez	14
10	67890123	Laura	Diaz	3
11	78901234	Carlos	Rodriguez	4
12	89012345	Sofía	Hernandez	7
13	90123456	Elena	Diaz	5

15. Añadir un nuevo departamento: 'Calidad' con presupuesto de 40000€ y código 11. Tendrá vinculado una empleada: 'Esther Vazquez' con DNI: 89267109

    CONSULTA:

    INSERT INTO departamentos (codigo, nombre, presupuesto) VALUES (11, 'Calidad', 40000);
    INSERT INTO empleados (dni, nombre, apellidos, departamento) VALUES ('8...9', 'Esther', 'Vázquez', 11);

16. Aplicar un recorte presupuestario del 10% a todos los departamentos

    CONSULTA: UPDATE departamentos SET presupuesto=presupuesto*0.9;

17. Reasignar a los empleados del departamento de investigación [77] al de informática [14]

    CONSULTA: UPDATE empleados SET departamento=14 WHERE departamento=77;

Después de los INSERTs [15], UPDATEs [16 & 17]

	nombre	presupuesto
1	Ventas	45.000
2	Marketing	54.000
3	Recursos Humanos	40.500
4	Finanzas	63.000
5	Producción	72.000
6	Soporte Tecnico	49.500
7	Calidad	36.000
8	Informatica	58.500
9	Administracion	63.000
10	Gramatica	43.200
11	Arquitectura	81.000
12	Investigacion	54.000

	dni	nombre	apellidos	departamento
1	01234567	Javier	Fernandez	37
2	11234567	Luis	Gómez	14
3	12345678	Juan	Gonzalez	1
4	22345678	Ana	Martínez	14
5	23456789	María	Perez	21
6	33456789	Mario	Sánchez	4
7	34567890	Pedro	Rodriguez	14
8	45678901	Ana	Lopez	2
9	56789012	David	Lopez	14
10	67890123	Laura	Diaz	3
11	78901234	Carlos	Rodriguez	4
12	89012345	Sofía	Hernandez	7
13	89267109	Esther	Vázquez	11
14	90123456	Elena	Diaz	5

18. Despedir a todos los empleados que trabajen para departamento de informática

CONSULTA: DELETE FROM empleados WHERE departamento=14;

19. Despedir a todos los empleados que trabajen para departamentos cuyo presupuesto sea superior a los 60000€

    CONSULTA: DELETE FROM empleados em INNER JOIN departamentos dep ON em.departamento=dep.codigo WHERE presupuesto>=60000;

Después de los DELETEs [18 & 19]

Antes

	dni	nombre	apellidos	departamento
1	01234567	Javier	Fernandez	37
2	11234567	Luis	Gómez	77
3	12345678	Juan	Gonzalez	1
4	22345678	Ana	Martínez	77
5	23456789	María	Perez	21
6	33456789	Mario	Sánchez	4
7	34567890	Pedro	Rodriguez	77
8	45678901	Ana	Lopez	2
9	56789012	David	Lopez	14
10	67890123	Laura	Diaz	3
11	78901234	Carlos	Rodriguez	4
12	89012345	Sofía	Hernandez	7
13	90123456	Elena	Diaz	5

Después

	dni	nombre	apellidos	departamento
1	01234567	Javier	Fernandez	37
2	12345678	Juan	Gonzalez	1
3	67890123	Laura	Diaz	3
4	89012345	Sofía	Hernandez	7

20. Despedir a todos los empleados

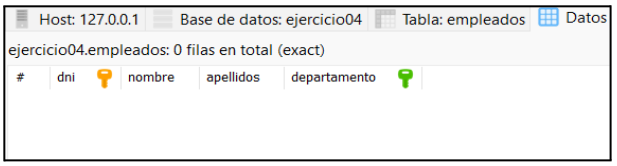
    CONSULTA: DELETE FROM empleados;

DELETE todo [20]

Antes

	dni	nombre	apellidos	departamento
1	01234567	Javier	Fernandez	37
2	11234567	Luis	Gómez	14
3	12345678	Juan	Gonzalez	1
4	22345678	Ana	Martínez	14
5	23456789	María	Perez	21
6	33456789	Mario	Sánchez	4
7	34567890	Pedro	Rodriguez	14
8	45678901	Ana	Lopez	2
9	56789012	David	Lopez	14
10	67890123	Laura	Diaz	3
11	78901234	Carlos	Rodriguez	4
12	89012345	Sofía	Hernandez	7
13	89267109	Esther	Vázquez	11
14	90123456	Elena	Diaz	5

Después



Ejercicio 03:

```sql
-- INSERCIONES
INSERT INTO almacenes (codigo, lugar, capacidad) VALUES
    (1, 'Sevilla', 10),
    (2, 'Bilbao', 45),
    (3, 'Madrid', 10),
    (4, 'Caceres', 12),
    (5, 'Valencia', 7),
    (6, 'Málaga', 2),
    (7, 'Zaragoza', 4),
    (8, 'Palma de Mallorca', 30),
    (9, 'Valladolid', 20),
    (10, 'Granada', 10);
```

```sql
INSERT INTO cajas (num_referencia, contenido, valor, almacen) VALUES
    ('CA001', 'Libros', 50, 1),
    ('CA002', 'Ropa', 20, 10),
    ('CA003', 'Electrónicos', 200, 2),
    ('CA004', 'Juguetes', 180, 2),
    ('CA005', 'Herramientas', 120, 3),
    ('CA006', 'Comestibles', 70, 3),
    ('CA007', 'Cosméticos', 90, 4),
    ('CA008', 'Decoración', 140, 4),
    ('CA009', 'Equipamiento deportivo', 150, 6),
    ('CA010', 'Artículos de oficina', 60, 5),
    ('CA011', 'DVDs', 30, 6),
    ('CA012', 'Instrumentos musicales', 180, 6),
    ('CA013', 'Artículos de cocina', 90, 7),
    ('CA014', 'Joyería', 200, 9),
    ('CA015', 'Material de arte', 130, 8);
```

1. Obtener todos los almacenes

   CONSULTA: SELECT * FROM almacenes;

2. Obtener todas las cajas cuyo contenido tenga un valor superior a 150€

   CONSULTA: SELECT * FROM cajas WHERE valor<150;

3. Obtener los tipos de contenidos de las cajas

   CONSULTA: SELECT num_referencia, contenido FROM cajas;

1

	codigo	lugar	capacidad
1	1	Sevilla	10
2	2	Bilbao	45
3	3	Madrid	10
4	4	Caceres	12
5	5	Valencia	7
6	6	Málaga	2
7	7	Zaragoza	4
8	8	Palma de Mallorca	30
9	9	Valladolid	20
10	10	Granada	10

2

	num_referencia	contenido	valor	almacen
1	CA001	Libros	50	1
2	CA002	Ropa	20	10
3	CA005	Herramientas	120	3
4	CA006	Comestibles	70	3
5	CA007	Cosméticos	90	4
6	CA008	Decoración	140	4
7	CA010	Artículos de oficina	60	5
8	CA011	DVDs	30	6
9	CA013	Artículos de cocina	90	7
10	CA015	Material de arte	130	8

3

	num_referencia	contenido
1	CA001	Libros
2	CA002	Ropa
3	CA003	Electrónicos
4	CA004	Juguetes
5	CA005	Herramientas
6	CA006	Comestibles
7	CA007	Cosméticos
8	CA008	Decoración
9	CA009	Equipamiento deportivo
10	CA010	Artículos de oficina
11	CA011	DVDs
12	CA012	Instrumentos musicales
13	CA013	Artículos de cocina
14	CA014	Joyería
15	CA015	Material de arte

4. Obtener el valor medio de todas las cajas

   CONSULTA: SELECT AVG(valor) FROM cajas;

5. Obtener el valor medio de las cajas de cada almacén

   CONSULTA: SELECT AVG(valor) FROM cajas GROUP BY almacen;

6. Obtener los códigos de los almacenes en los cuales el valor medio de las cajas sea superior a 150€

   CONSULTA: SELECT almacen FROM cajas GROUP BY almacen HAVING AVG(valor)>150;

```
AVG(valor) AS 'Media valores'
```

4

	Media valores
1	114,0

5

	Media valores
1	50,0
2	190,0
3	95,0
4	115,0
5	60,0
6	120,0
7	90,0
8	130,0
9	200,0
10	20,0

6

	almacen
1	2
2	9

7. Obtener el número de referencia de cada caja junto con el nombre de la ciudad en el que se encuentra

   CONSULTA: SELECT num_referencia, lugar FROM cajas ca INNER JOIN almacenes al ON ca.almacen=al.codigo;

8. Obtener el número de cajas que hay en cada almacén

   CONSULTA: SELECT COUNT(num_referencia), codigo, lugar FROM cajas ca INNER JOIN almacenes al ON ca.almacen=al.codigo GROUP BY almacen;

```
COUNT(num_referencia) AS ' Conteo num_ref'
```

7

	num_referencia	lugar
1	CA001	Sevilla
2	CA003	Bilbao
3	CA004	Bilbao
4	CA005	Madrid
5	CA006	Madrid
6	CA007	Caceres
7	CA008	Caceres
8	CA010	Valencia
9	CA009	Málaga
10	CA011	Málaga
11	CA012	Málaga
12	CA013	Zaragoza
13	CA015	Palma de Mallorca
14	CA014	Valladolid
15	CA002	Granada

8

	Conteo num_ref	codigo	lugar
1	1	1	Sevilla
2	2	2	Bilbao
3	2	3	Madrid
4	2	4	Caceres
5	1	5	Valencia
6	3	6	Málaga
7	1	7	Zaragoza
8	1	8	Palma de Mallorca
9	1	9	Valladolid
10	1	10	Granada

9. Obtener los códigos de los almacenes que están saturados

   CONSULTA: SELECT almacen, lugar, capacidad FROM cajas ca INNER JOIN almacenes al ON ca.almacen=al.codigo GROUP BY almacen HAVING capacidad<COUNT(num_referencia);

10. Obtener los numeras de referencia de las cajas que están en Bilbao

    CONSULTA: SELECT num_referencia FROM cajas ca INNER JOIN almacenes al ON ca.almacen=al.codigo WHERE lugar='Bilbao';

9

	almacen	lugar	capacidad
1	6	Málaga	2

10

	num_referencia
1	CA003
2	CA004

Sin respuesta visual [11, 12, 13, 14, 15 & 16]

Antes

	codigo	lugar	capacidad
1	1	Sevilla	10
2	2	Bilbao	45
3	3	Madrid	10
4	4	Caceres	12
5	5	Valencia	7
6	6	Málaga	2
7	7	Zaragoza	4
8	8	Palma de Mallorca	30
9	9	Valladolid	20
10	10	Granada	10

	num_referencia	contenido	valor	almacen
1	CA001	Libros	50	1
2	CA002	Ropa	20	10
3	CA003	Electrónicos	200	2
4	CA004	Juguetes	180	2
5	CA005	Herramientas	120	3
6	CA006	Comestibles	70	3
7	CA007	Cosméticos	90	4
8	CA008	Decoración	140	4
9	CA009	Equipamiento deportivo	150	6
10	CA010	Artículos de oficina	60	5
11	CA011	DVDs	30	6
12	CA012	Instrumentos musicales	180	6
13	CA013	Artículos de cocina	90	7
14	CA014	Joyería	200	9
15	CA015	Material de arte	130	8

11. Nuevo almacén en Barcelona con capacidad para 3 cajas

    CONSULTA: INSERT INTO almacenes (codigo, lugar, capacidad) VALUES (11, 'Barcelona', 3);

12. Nueva caja: número de referencia 'H5RT', contenido: 'Papel', valor: 200, y código: 2

    CONSULTA: INSERT INTO cajas (num_referencia, contenido, valor, almacen) VALUES ('H5RT', 'Papel', 200, 2);

13. Rebajar el valor de todas las cajas un 15%

    CONSULTA: UPDATE cajas SET valor=valor*0.85;

14. Rebajar un 20% el valor a las cajas con valor superior al valor medio de todas

    CONSULTA: UPDATE cajas SET valor=valor*0.8 WHERE valor>(SELECT AVG(valor) FROM cajas);

Después de los INSERTs [15], UPDATEs [16 & 17]

	codigo	lugar	capacidad
1	1	Sevilla	10
2	2	Bilbao	45
3	3	Madrid	10
4	4	Caceres	12
5	5	Valencia	7
6	6	Málaga	2
7	7	Zaragoza	4
8	8	Palma de Mallorca	30
9	9	Valladolid	20
10	10	Granada	10
11	11	Barcelona	3

	num_referencia	contenido	valor	almacen
1	CA001	Libros	43	1
2	CA002	Ropa	17	10
3	CA003	Electrónicos	136	2
4	CA004	Juguetes	122	2
5	CA005	Herramientas	82	3
6	CA006	Comestibles	60	3
7	CA007	Cosméticos	77	4
8	CA008	Decoración	95	4
9	CA009	Equipamiento deportivo	102	6
10	CA010	Artículos de oficina	51	5
11	CA011	DVDs	26	6
12	CA012	Instrumentos musicales	122	6
13	CA013	Artículos de cocina	77	7
14	CA014	Joyería	136	9
15	CA015	Material de arte	89	8
16	H5RT	Papel	136	2

15. Eliminar todas las cajas cuyo valor sea inferior a 100€

    CONSULTA: DELETE FROM cajas WHERE valor < 100;

16. Vaciar el contenido de los almacenes que están saturados

    CONSULTA: DELETE FROM cajas WHERE almacen IN (SELECT codigo FROM almacenes a JOIN (SELECT almacen, COUNT(*) AS 'conteo' FROM cajas GROUP BY almacen) c ON a.codigo = c.almacen WHERE capacidad < conteo);

Después de los DELETEs [15 & 16]

	num_referencia	contenido	valor	almacen
1	CA003	Electrónicos	136	2
2	CA004	Juguetes	122	2
3	CA011	DVDs	30	6
4	CA012	Instrumentos musicales	122	6
5	CA014	Joyería	136	9
6	H5RT	Papel	136	2

Ejercicio 04:

```sql
-- INSERCIONES
INSERT INTO peliculas (codigo, nombre, restriccion_edad) VALUES
    (1, 'La Casa de Papel', 18),
    (2, 'El Señor de los Anillos', 12),
    (5, 'Harry Potter y la Piedra Filosofal', 7),
    (7, 'El Rey León', 0),
    (8, 'El Padrino', 18),
    (9, 'Titanic', 12),
    (10, 'El Lobo de Wall Street', 18);
INSERT INTO peliculas (codigo, nombre) VALUES
    (3, 'Jurassic Park'),
    (4, 'Los Vengadores'),
    (6, 'Piratas del Caribe: La maldición de la Perla Negra');
```

```sql
INSERT INTO salas (codigo, nombre, pelicula) VALUES
    (1, 'Sala 1', 1),
    (2, 'Sala 2', 2),
    (3, 'Sala 3', 3),
    (4, 'Sala 4', 3),
    (5, 'Sala 5', 7),
    (6, 'Sala 6', 6),
    (7, 'Sala 7', 7),
    (8, 'Sala 8', 4);
INSERT INTO salas (codigo, nombre) VALUES
    (9, 'Sala 9'),
    (10, 'Sala 10');
```

1. Mostrar el nombre de todas las películas
   CONSULTA: SELECT nombre FROM peliculas;
2. Mostrar las distintas calificaciones de edad que existen
   CONSULTA: SELECT restriccion_edad FROM peliculas GROUP BY restriccion_edad;

1

	nombre
1	La Casa de Papel
2	El Señor de los Anillos
3	Jurassic Park
4	Los Vengadores
5	Harry Potter y la Piedra Filosofal
6	Piratas del Caribe: La maldición de la Perla Negra
7	El Rey León
8	El Padrino
9	Titanic
10	El Lobo de Wall Street

2

	restriccion_edad
1	(NULL)
2	0
3	7
4	12
5	18

3. Mostrar todas las películas que no han sido calificadas
   CONSULTA: SELECT * FROM peliculas WHERE restriccion_edad IS NULL;
4. Mostrar todas las salas que no proyectan ninguna película
   CONSULTA: SELECT nombre FROM salas WHERE pelicula IS NULL;

3

	codigo	nombre	restriccion_edad
1	3	Jurassic Park	(NULL)
2	4	Los Vengadores	(NULL)
3	6	Piratas del Caribe: La maldición de la Perla Negra	(NULL)

4

	nombre
1	Sala 9
2	Sala 10

5. Mostrar la información de todas las salas y, si se proyecta alguna película en la sala, mostrar también la información de la película

   CONSULTA: SELECT sa.*, peliculas.* FROM salas sa LEFT JOIN peliculas pe ON sa.pelicula=pe.codigo;

6. Mostrar la información de todas las películas y, si se proyecta en alguna sala, mostrar también la información de la sala

   CONSULTA: SELECT pe.*, sa.* FROM peliculas pe LEFT JOIN salas sa ON pe.codigo=sa.pelicula;

5

	codigo	nombre	pelicula	codigo	nombre	restriccion_edad
1	1	Sala 1	1	1	La Casa de Papel	18
2	2	Sala 2	2	2	El Señor de los Anillos	12
3	3	Sala 3	3	3	Jurassic Park	(NULL)
4	4	Sala 4	3	3	Jurassic Park	(NULL)
5	5	Sala 5	7	7	El Rey León	0
6	6	Sala 6	6	6	Piratas del Caribe: La maldición de la Perla Negra	(NULL)
7	7	Sala 7	7	7	El Rey León	0
8	8	Sala 8	4	4	Los Vengadores	(NULL)
9	9	Sala 9	(NULL)	(NULL)	(NULL)	(NULL)
10	10	Sala 10	(NULL)	(NULL)	(NULL)	(NULL)

6

	codigo	nombre	restriccion_edad	codigo	nombre	pelicula
1	1	La Casa de Papel	18	1	Sala 1	1
2	2	El Señor de los Anillos	12	2	Sala 2	2
3	3	Jurassic Park	(NULL)	3	Sala 3	3
4	3	Jurassic Park	(NULL)	4	Sala 4	3
5	4	Los Vengadores	(NULL)	8	Sala 8	4
6	5	Harry Potter y la Piedra Filosofal	7	(NULL)	(NULL)	(NULL)
7	6	Piratas del Caribe: La maldición de la Perla Negra	(NULL)	6	Sala 6	6
8	7	El Rey León	0	5	Sala 5	7
9	7	El Rey León	0	7	Sala 7	7
10	8	El Padrino	18	(NULL)	(NULL)	(NULL)
11	9	Titanic	12	(NULL)	(NULL)	(NULL)
12	10	El Lobo de Wall Street	18	(NULL)	(NULL)	(NULL)

7. Mostrar los nombres de las películas que no se proyectan en ninguna sala

   CONSULTA: SELECT p.nombre FROM peliculas p LEFT JOIN salas s ON p.codigo=s.pelicula WHERE s.codigo IS NULL;

7

	nombre
1	Harry Potter y la Piedra Filosofal
2	El Padrino
3	Titanic
4	El Lobo de Wall Street

Sin respuesta visual [15, 16, 17, 18, 19 & 20]

Antes

	codigo	nombre	restriccion_edad
1	1	La Casa de Papel	18
2	2	El Señor de los Anillos	12
3	3	Jurassic Park	(NULL)
4	4	Los Vengadores	(NULL)
5	5	Harry Potter y la Piedra Filosofal	7
6	6	Piratas del Caribe: La maldición de la Perla Negra	(NULL)
7	7	El Rey León	0
8	8	El Padrino	18
9	9	Titanic	12
10	10	El Lobo de Wall Street	18

	codigo	nombre	pelicula
1	1	Sala 1	1
2	2	Sala 2	2
3	3	Sala 3	3
4	4	Sala 4	3
5	5	Sala 5	7
6	6	Sala 6	6
7	7	Sala 7	7
8	8	Sala 8	4
9	9	Sala 9	(NULL)
10	10	Sala 10	(NULL)

8. Añadir una nueva película 'Uno, Dos, Tres', para mayores de 7 años

   CONSULTA: INSERT INTO peliculas (codigo, nombre, restriccion_edad) VALUES (11, 'Uno, Dos, Tres', 7);

9. Hacer constar que todas las películas no calificadas han sido calificadas 'no recomendables para menores de 13 años'

   CONSULTA: UPDATE peliculas SET restriccion_edad=13 WHERE restriccion_edad IS NULL;

Después del INSERT [8] y del UPDATE [9]

	codigo	nombre	restriccion_edad
1	1	La Casa de Papel	18
2	2	El Señor de los Anillos	12
3	3	Jurassic Park	13
4	4	Los Vengadores	13
5	5	Harry Potter y la Piedra Filosofal	7
6	6	Piratas del Caribe: La maldición de la Perla Negra	13
7	7	El Rey León	0
8	8	El Padrino	18
9	9	Titanic	12
10	10	El Lobo de Wall Street	18
11	11	Uno, Dos, Tres	7

Si ejecutamos el UPDATE anterior ,donde a las películas que no tengan una restricción de edad le asignaremos una de 13 años, antes que el próximo DELETE este no tendrá efecto por cuestiones lógicas. Por lo que para ver un cambio utilizare la base de datos antes de este UPDATE.

TABLA 'peliculas' aplicando solo el INSERT:

	codigo	nombre	restriccion_edad
1	1	La Casa de Papel	18
2	2	El Señor de los Anillos	12
3	3	Jurassic Park	(NULL)
4	4	Los Vengadores	(NULL)
5	5	Harry Potter y la Piedra Filosofal	7
6	6	Piratas del Caribe: La maldición de la Perla Negra	(NULL)
7	7	El Rey León	0
8	8	El Padrino	18
9	9	Titanic	12
10	10	El Lobo de Wall Street	18
11	11	Uno, Dos, Tres	7

10. Eliminar todas las salas que proyectan películas recomendadas para todos los públicos

CONSULTA: DELETE FROM salas WHERE pelicula IS NOT NULL AND pelicula IN (SELECT codigo FROM peliculas WHERE restriccion_edad IS NULL);

Después del DELETE [10]

	codigo	nombre	pelicula
1	1	Sala 1	1
2	2	Sala 2	2
3	5	Sala 5	7
4	7	Sala 7	7
5	9	Sala 9	(NULL)
6	10	Sala 10	(NULL)

Webgrafía

- Funciones SUM(), AVG() y COUNT():
  Funciones → W3Schools
- Reglas de las subconsultas:
  Subconsultas → IBM
- Sintaxis IN ():
  Sintaxis → Microsoft
- El copiloto de confianza como ayuda siempre, importante:
  ChatGPT → Openai