EJERCICIO 1

1.1 Obtener los nombres de los productos de la tienda.

SELECT NOMBRE FROM articulos;

1.2 Obtener los nombres y los precios de los productos de la tienda.

SELECT NOMBRE, PRECIO FROM articulos;

1.3 Obtener el nombre de los productos cuyo precio sea menor o igual a 200 €.

SELECT NOMBRE FROM articulos WHERE PRECIO <= 200;

1.4 Obtener todos los datos de los articulos cuyo precio esté entre 60 € y los 120€ (ambas cantidades incluidas).

SELECT NOMBRE FROM articulos WHERE PRECIO BETWEEN 60 AND 120;

1.5 Obtener el nombre y el precio en pesetas (es decir, en euros multiplicado por 166’386).

SELECT NOMBRE, (PRECIO*166.386) FROM articulos;

1.6 Seleccionar el precio medio de todos los productos.

SELECT AVG(PRECIO) FROM articulos;

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

SELECT AVG(PRECIO) FROM articulos where FABRICANTE LIKE 2;

1.8 Obtener el numero de artículos cuyo precio sea mayor o igual a 180€.

SELECT COUNT(PRECIO) FROM articulos where PRECIO >= 180;

Arnau Colomé, Alba Fischer, Daniel Macias

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

SELECT NOMBRE, PRECIO FROM articulos where PRECIO >= 180 order by precio desc, nombre asc;

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

SELECT a.*, f.NOMBRE FROM (articulos a INNER JOIN fabricantes f ON a.fabricante = f.codigo);

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

SELECT a.NOMBRE, a.PRECIO, f.NOMBRE FROM (articulos a INNER JOIN fabricantes f ON a.fabricante = f.codigo);

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

SELECT AVG(PRECIO), FABRICANTE FROM articulos GROUP BY FABRICANTE;

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

SELECT AVG(a.PRECIO), f.NOMBRE FROM (articulos a INNER JOIN fabricantes f ON a.fabricante = f.codigo) GROUP BY f.NOMBRE;

1. 14 Obtener los nombres de los fabricantes que ofrezcan productos cuyo precio medio sea mayor o igual a 150€.

```
SELECT  f.NOMBRE
FROM (articulos a INNER JOIN fabricantes f ON a.fabricante = f.codigo)
where a.PRECIO >= 150
GROUP BY f.NOMBRE
```

1.15 Obtener el nombre y precio del artículo más barato

SELECT PRECIO, NOMBRE FROM articulos WHERE PRECIO = (SELECT MIN(PRECIO) FROM articulos);

1.16 Obtener una lista con el nombre y precio de los artículos más caros de cada proveedor (incluyendo el nombre del proveedor)

```
SELECT articulos.NOMBRE, articulos.PRECIO, fabricantes.NOMBRE
FROM articulos, fabricantes
WHERE articulos.FABRICANTE= fabricantes.CODIGO
AND articulos.PRECIO=(SELECT MAX(PRECIO) FROM articulos WHERE
articulos.FABRICANTE =fabricantes.CODIGO);
```

1.17 Añadir un nuevo producto: Altavoces de 70 € (del fabricante 2)

```
INSERT INTO articulos VALUES (11, 'Altavoces', 70, 2);
```

1.18 Cambia el nombre del producto 8 a ‘Impresora Laser’.

```
UPDATE articulos SET NOMBRE = 'Impresora Laser' WHERE CODIGO = 8;
```

1.19 Aplica un descuento de 10% (multiplicar el precio por 0.9) a todos los productos.

```
UPDATE articulos SET PRECIO = PRECIO * 0.9;
```

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

```
UPDATE articulos SET PRECIO = PRECIO - 10 WHERE PRECIO >= 120;
```

EJERCICIO 2

2.1. Obtener los apellidos de los empleados.

SELECT APELLIDOS FROM empleados;

2.2. Obtener los apellidos de los empleados sin repeticiones.

SELECT APELLIDOS FROM empleados GROUP BY APELLIDOS;

2.3. Obtener todos los datos de los empleados que se apellidan 'López'.

SELECT * FROM empleados WHERE APELLIDOS = 'López';

2.4. Obtener todos los datos de los empleados que se apellidan 'López' y los que se apellidan 'Pérez'.

SELECT * FROM empleados WHERE APELLIDOS = 'López' OR APELLIDOS = 'Pérez';

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

SELECT * FROM empleados WHERE DEPARTAMENTO = 14;

2.6. Obtener todos los datos de los empleados que trabajan para el departamento 37 y para el departamento 77.

SELECT * FROM empleados WHERE DEPARTAMENTO = 37 OR DEPARTAMENTO = 77;

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

SELECT * FROM empleados WHERE APELLIDOS LIKE 'P%';

2.8. Obtener el presupuesto total de todos los departamentos.

SELECT SUM(PRESUPUESTO) FROM departamentos;

2.9. Obtener el numero de empleados en cada departamento.

SELECT COUNT(DEPARTAMENTO), DEPARTAMENTO FROM empleados GROUP BY DEPARTAMENTO;

Arnau Colomé, Alba Fischer, Daniel Macias

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

```
SELECT empleados.*, departamentos.*
FROM empleados
INNER JOIN departamentos
ON empleados.DEPARTAMENTO=departamentos.CODIGO;
```

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

```
SELECT empleados.NOMBRE, empleados.APELLIDOS, departamentos.NOMBRE,
departamentos.PRESUPUESTO
FROM empleados
INNER JOIN departamentos
ON empleados.DEPARTAMENTO=departamentos.CODIGO;
```

2.12. Obtener los nombres y apellidos de los empleados que trabajen en departamentos cuyo presupuesto sea mayor de 60.000 €.

```
SELECT NOMBRE, APELLIDOS FROM empleados WHERE 60000 < (SELECT
PRESUPUESTO FROM departamentos WHERE codigo = DEPARTAMENTO);
```

2.13. Obtener los datos de los departamentos cuyo presupuesto es superior al presupues- to medio de todos los departamentos.

```
SELECT * FROM departamentos WHERE PRESUPUESTO > (SELECT
AVG(PRESUPUESTO) FROM departamentos);
```

2.14. Obtener los nombres (únicamente los nombres) de los departamentos que tienen más de dos empleados.

```
SELECT d.NOMBRE
FROM (empleados e INNER JOIN departamentos d ON e.DEPARTAMENTO = d.CODIGO)
GROUP BY d.NOMBRE
HAVING COUNT(e.DEPARTAMENTO) >2
```

2.15. Añadir un nuevo departamento: 'Calidad', con presupuesto de 40.000 € y código11. Añadir un empleado vinculado al departamento recién creado: Esther Vázquez, DNI: 89267109

```
INSERT INTO departamentos VALUES (11, "Calidad", 40000);
INSERT INTO empleados VALUES (89267109, "Esther", "Vázquez", 11);
```

2.16. Aplicar un recorte presupuestario del 10 % a todos los departamentos.
UPDATE departamentos SET PRESUPUESTO = PRESUPUESTO - (PRESUPUESTO * 0.1);

2.17. Reasignar a los empleados del departamento de investigación (código 77) al depar- tamento de informática (código 14).

UPDATE empleados SET DEPARTAMENTO = 14 WHERE DEPARTAMENTO = 77;

2.18. Despedir a todos los empleados que trabajan para el departamento de informática
(código 14).

DELETE FROM empleados WHERE DEPARTAMENTO = 14;

2.19. Despedir a todos los empleados que trabajen para departamentos cuyo presupuesto sea superior a los 60.000 €.

DELETE FROM empleados WHERE 60000 < (SELECT PRESUPUESTO FROM departamentos WHERE codigo = DEPARTAMENTO);

2.20. Despedir a todos los empleados.

DELETE FROM empleados;

EJERCICIO 3

3.1. Obtener todos los almacenes

SELECT * FROM almacenes;

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

SELECT * FROM cajas WHERE VALOR > 150;

3.3. Obtener los tipos de contenidos de las cajas.

SELECT CONTENIDO FROM Cajas ;

3.4. Obtener el valor medio de todas las cajas.

SELECT AVG(VALOR) FROM cajas;

3.5. Obtener el valor medio de las cajas de cada almacen.

SELECT AVG(VALOR) FROM cajas GROUP BY almacen;

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

SELECT CODIGO FROM almacenes WHERE (SELECT AVG(VALOR) FROM cajas) > 150;

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

SELECT c.NUMREFERENCIA, a.LUGAR
FROM (almacenes a INNER JOIN cajas c ON a.CODIGO = c.ALMACEN)

3.8. Obtener el numero de cajas que hay en cada almacén.

SELECT COUNT(NUMREFERENCIA) FROM cajas GROUP BY almacen;

3.9. Obtener los códigos de los almacenes que están saturados (los almacenes donde el numero de cajas es superior a la capacidad).

SELECT CODIGO FROM almacenes
WHERE CAPACIDAD > (SELECT COUNT(NUMREFERENCIA) FROM cajas WHERE ALMACEN = CODIGO);

Arnau Colomé, Alba Fischer, Daniel Macias

3.10. Obtener los numeros de referencia de las cajas que están en Bilbao.

```
SELECT c.NUMREFERENCIA FROM (almacenes a INNER JOIN cajas c ON a.CODIGO =
c.ALMACEN)
WHERE a.LUGAR LIKE 'Bilbao'
```

3.11. Insertar un nuevo almacén en Barcelona con capacidad para 3 cajas.

```
INSERT INTO almacenes (Lugar, Capacidad) VALUES ('Barcelona',3);
```

3.12. Insertar una nueva caja, con número de referencia 'H5RT', con contenido 'Papel', valor 200, y situada en el almacén 2.

```
INSERT INTO cajas VALUES ('H5RT','Papel', 200, 2);
```

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

```
UPDATE cajas SET VALOR = VALOR*1.15;
```

3.14. Rebajar un 20 % el valor de todas las cajas cuyo valor sea superior al valor medio de todas las cajas.

```
UPDATE cajas SET VALOR = VALOR*0.80
WHERE VALOR > (SELECT VALOR FROM (SELECT AVG(VALOR) FROM cajas) AS C);
```

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

```
DELETE FROM cajas WHERE VALOR<100;
```

3.16. Vaciar el contenido de los almacenes que están saturados

```
DELETE FROM almacenes WHERE CAPACIDAD < (SELECT COUNT(ALMACEN) FROM
cajas WHERE ALMACEN = CODIGO GROUP BY ALMACEN);
```

EJERCICIO 4

4.1. Mostrar el nombre de todas las películas.

SELECT NOMBRE FROM peliculas;

4.2. Mostrar las distintas calificaciones de edad que existen.

SELECT DISTINCT(CALIFICACIONEDAD) FROM peliculas;

4.3. Mostrar todas las películas que no han sido calificadas.

SELECT * FROM peliculas WHERE CALIFICACIONEDAD IS NULL;

4.4. Mostrar todas las salas que no proyectan ninguna película.

SELECT * FROM salas WHERE PELICULA IS NULL;

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

SELECT * FROM salas LEFT JOIN peliculas ON salas.PELICULA = peliculas.CODIGO;

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

SELECT * FROM salas RIGHT JOIN peliculas ON salas.PELICULA = peliculas.CODIGO;

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

SELECT * FROM salas right JOIN peliculas ON salas.PELICULA = peliculas.CODIGO
WHERE salas.CODIGO is null;

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

INSERT INTO peliculas VALUES (10, 'Uno, Dos, Tres', 'PG-7');

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

UPDATE peliculas SET CALIFICACIONEDAD = 'PG-13' WHERE CALIFICACIONEDAD IS NULL;

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

```
DELETE FROM peliculas WHERE CALIFICACIONEDAD <> 'G';
```