CASO PRÁCTICO 1

Para crear las tablas en la base de datos **Empresa**, vamos utilizar **SQL** en un **SGDB** como MySQL Workbench. A continuación, muestro los comandos **SQL** para la creación de las tablas con sus claves primarias y foráneas:

```sql
CREATE DATABASE IF NOT EXISTS Empresa;
USE Empresa;
```

A continuacion creamos la tabla cliente:

```sql
CREATE TABLE Clientes (
    ID_Cliente INT AUTO_INCREMENT PRIMARY KEY,
    Nombre VARCHAR(20) NOT NULL,
    Apellidos VARCHAR(50) NOT NULL,
    DNI VARCHAR(9) NOT NULL UNIQUE,
    Dirección VARCHAR(40) NOT NULL,
    CodigoPostal VARCHAR(6) NOT NULL,
    Teléfono VARCHAR(12) NOT NULL
);
```

Tabla Articulos:

```sql
CREATE TABLE Articulos (
    ID_Articulo INT AUTO_INCREMENT PRIMARY KEY,
    Nombre VARCHAR(20) NOT NULL,
    Precio DECIMAL(10,2) NOT NULL
);
```

Tabla Ventas:

```sql
CREATE TABLE Ventas (
    ID_Venta INT AUTO_INCREMENT PRIMARY KEY,
    ID_Cliente INT NOT NULL,
    ID_Articulo INT NOT NULL,
    Fecha DATE NOT NULL,
    Abonado BOOLEAN NOT NULL,
    Cantidad INT NOT NULL CHECK (Cantidad > 0),
    FOREIGN KEY (ID_Cliente) REFERENCES Clientes(ID_Cliente) ON DELETE CASCADE,
    FOREIGN KEY (ID_Articulo) REFERENCES Articulos(ID_Articulo) ON DELETE CASCADE
);
```

Explicación del Código

1. **Creamos la base de datos** si no existe y la utilizamos.

2. **Definimos la tabla** Clientes con su clave primaria (ID_Cliente) y un campo DNI único.
3. **Definimos la tabla** Articulos con su clave primaria (ID_Articulo) y un campo Precio en formato decimal.
4. **Definimos la tabla** Ventas con su clave primaria (ID_Venta) y claves foráneas (ID_Cliente y ID_Articulo) que referencian a las otras tablas.
5. **Restringimos la cantidad** para que sea mayor a 0 con CHECK (Cantidad > 0).
6. **Implementamos** ON DELETE CASCADE, lo que significa que si se borra un cliente o un artículo, sus ventas asociadas también se eliminarán automáticamente.

2. Crea las siguientes relaciones entre las tablas:

Para crear las relaciones entre las tablas en **MySQL**, debo establecer claves foráneas en la tabla **ventas**, ya que esta tabla se relaciona con **clientes** y **artículos**.

A continuación, muestro el **código SQL** para definir estas relaciones:

-- Agregamos las claves foráneas en la tabla Ventas:

```
27 •  ALTER TABLE Ventas
28    ADD CONSTRAINT FK_Ventas_Clientes
29    FOREIGN KEY (ID_Cliente) REFERENCES Clientes(ID_Cliente)
30    ON DELETE CASCADE ON UPDATE CASCADE;
```

```
31 •  ALTER TABLE Ventas
32    ADD CONSTRAINT FK_Ventas_Articulos
33    FOREIGN KEY (ID_Articulo) REFERENCES Articulos(ID_Articulo)
34    ON DELETE CASCADE ON UPDATE CASCADE;
```

Explicación del Código

1. **Claves foráneas en** ventas:
   - ID_Cliente hace referencia a ID_Cliente en clientes.
   - ID_Articulo hace referencia a ID_Articulo en artículos.
2. **Restricciones de integridad referencial**:
   - ON DELETE CASCADE: Si se elimina un cliente o un artículo, las ventas asociadas también se eliminan automáticamente.
   - ON UPDATE CASCADE: Si cambia el ID_Cliente o ID_Articulo, se actualiza en ventas.

Diagrama Relacional

El diagrama de la imagen refleja estas relaciones:

- **Uno a muchos** entre clientes y ventas (un cliente puede tener varias ventas).
- **Uno a muchos** entre artículos y ventas (un artículo puede estar en varias ventas).

Con esto ya hemos creado correctamente las relaciones en la base de datos.

3. Introduce los siguientes valores en cada una de las tablas:

Para introducir los valores en cada una de las tablas **clientes**, **artículos** y **ventas**, utilizo el comando INSERT INTO en **SQL**.

Aquí muestro el código SQL para insertar los datos según la imagen:

-- Insertar datos en la tabla Clientes:

```
35 •  INSERT INTO Clientes (ID_Cliente, Nombre, Apellidos, DNI, Dirección, CodigoPostal, Teléfono) VALUES
36    (1, 'Sergio', 'González Ruiz', '123456789', 'calle cristo', '29700', '658'),
37    (2, 'Francisco', 'Garcia Perez', '4587965', 'canalejas', '29700', '658954855'),
38    (3, 'Felipe', 'Ruiz Pascual', '456789123', 'poeta', '56845', '654879256');
```

-- Insertar datos en la tabla Artículos:

```
39 •  INSERT INTO Articulos (ID_Articulo, Nombre, Precio) VALUES
40    (1, 'tomates', 0.1),
41    (2, 'pimientos', 3),
42    (3, 'sandías', 1),
43    (4, 'guindas', 7);
```

-- Insertar datos en la tabla Ventas:

```
105 •  INSERT INTO Ventas (ID_Venta, ID_Cliente, ID_Articulo, Fecha, Abonado, Cantidad) VALUES
106    (1, 1, 1, '2012-01-01', 0, 8),
107    (2, 1, 2, '2011-04-05', 1, 10),
108    (3, 2, 1, '2010-06-01', 0, 75),
109    (4, 1, 3, '2011-04-05', 1, 120),
110    (5, 3, 3, '2014-01-02', 0, 800);
```

Explicación del Código

1. **Insertamos clientes** con los datos de la imagen.
2. **Insertamos artículos** con nombres y precios.
3. **Insertamos ventas**:
    - ID_Cliente y ID_Articulo coinciden con los datos insertados en las otras tablas.

- Fecha en formato YYYY-MM-DD.
- Abonado como 1 (TRUE) o 0 (FALSE).
- Cantidad representa la cantidad vendida.

Consideraciones

- **El orden de inserción es importante**:
  1 **Clientes** → 2 **Artículos** → 3 **Ventas**
  Para evitar errores de integridad referencial.
- **Si las claves primarias son** AUTO_INCREMENT, no es necesario especificar los ID_Cliente, ID_Articulo ni ID_Venta en el INSERT.

Con esto ya tenemos los datos cargados correctamente en la base de datos!

4. Resuelve las siguientes consultas:

1. Seleccionar el nombre y apellidos de clientes que tienen las ventas abonadas

```
111 •   SELECT DISTINCT c.Nombre, c.Apellidos
112     FROM Clientes c
113     JOIN Ventas v ON c.ID_Cliente = v.ID_Cliente
114     WHERE v.Abonado = 1;
```

Result Grid | Filter Rows: | Export: | Wrap Ce

Nombre	Apellidos
Sergio	González Ruiz

Explicación:

- JOIN para conectar Clientes con Ventas.
- WHERE v.Abonado = 1 filtra solo las ventas abonadas.
- DISTINCT evita nombres duplicados si un cliente tiene varias ventas abonadas.

2. Seleccionar todos los registros de todas las tablas

SELECT * FROM Clientes;

```
63 •   SELECT * FROM Clientes;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

ID_Cliente	Nombre	Apellidos	DNI	Dirección	CodigoPostal	Teléfono
1	Sergio	González Ruiz	123456789	calle cristo	29700	658
2	Francisco	Garcia Perez	4587965	canalejas	29700	658954855
3	Felipe	Ruiz Pascual	456789123	poeta	56845	654879256
NULL	NULL	NULL	NULL	NULL	NULL	NULL

SELECT * FROM Articulos;

```
64 •  SELECT * FROM Articulos;
```

Result Grid | Filter Rows:

ID_Articulo	Nombre	Precio
1	tomates	0.10
2	pimientos	3.00
3	sandías	1.00
4	guindas	7.00
NULL	NULL	NULL

SELECT * FROM Ventas;

```
116 •  SELECT * FROM Ventas;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

ID_Venta	ID_Cliente	ID_Articulo	Fecha	Abonado	Cantidad
1	1	1	2012-01-01	0	8
2	1	2	2011-04-05	1	10
3	2	1	2010-06-01	0	75
4	1	3	2011-04-05	1	120
5	3	3	2014-01-02	0	800

Explicación:

- SELECT * devuelve todos los registros de cada tabla.
- Se ejecutan tres consultas separadas.

3. Nombre y apellidos de clientes, nombre de artículos que compran tomates o pimientos

```
67      FROM Clientes c
68      JOIN Ventas v ON c.ID_Cliente = v.ID_Cliente
69      JOIN Articulos a ON v.ID_Articulo = a.ID_Articulo
70      WHERE a.Nombre IN ('tomates', 'pimientos');
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Nombre	Apellidos	Nombre_Articulo
Sergio	González Ruiz	tomates
Felipe	Ruiz Pascual	tomates
Sergio	González Ruiz	pimientos

Explicación:

- JOIN para conectar las tablas Clientes, Ventas y Articulos.

- WHERE a.Nombre IN ('tomates', 'pimientos') filtra solo las compras de esos productos.

4. Nombre y apellidos de clientes, nombre de artículo, que compran tomates o pimientos y cuyo nombre empieza por "S"

```
71 •  SELECT c.Nombre, c.Apellidos, a.Nombre AS Nombre_Articulo
72    FROM Clientes c
73    JOIN Ventas v ON c.ID_Cliente = v.ID_Cliente
74    JOIN Articulos a ON v.ID_Articulo = a.ID_Articulo
75    WHERE a.Nombre IN ('tomates', 'pimientos')
76    AND c.Nombre LIKE 'S%';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Nombre	Apellidos	Nombre_Articulo
Sergio	González Ruiz	tomates
Sergio	González Ruiz	pimientos

5. Todos los datos de los clientes, nombre de artículo y cantidad, de los clientes que compran más de 10 tomates

```
128 •  SELECT c.*, a.Nombre AS Nombre_Articulo, v.Cantidad
129    FROM Clientes c
130    JOIN Ventas v ON c.ID_Cliente = v.ID_Cliente
131    JOIN Articulos a ON v.ID_Articulo = a.ID_Articulo
132    WHERE a.Nombre = 'tomates'
133    AND v.Cantidad > 10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ID_Cliente	Nombre	Apellidos	DNI	Dirección	CodigoPostal	Teléfono	Nombre_Articulo	Cantidad
2	Francisco	Garcia Perez	4587965	canalejas	29700	658954855	tomates	75

Explicación:

- Se seleccionan **todos los datos de** Clientes **(**c.***)**.
- Se unen las tablas con JOIN.
- Se filtran las ventas donde:
  - El artículo sea **"tomates"**.
  - La cantidad comprada sea **mayor a 10**.

Tabla Resumen de las Consultas SQL

Consulta	Explicación	Código SQL
Clientes con ventas abonadas	Devuelve los nombres de los clientes que han pagado sus compras.	SELECT DISTINCT c.Nombre, c.Apellidos FROM Clientes c JOIN Ventas v ON c.ID_Cliente = v.ID_Cliente WHERE v.Abonado = 1;
Todas las tablas	Muestra todos los registros de cada tabla.	SELECT * FROM Clientes; SELECT * FROM Articulos; SELECT * FROM Ventas;
Clientes que compran tomates o pimientos	Lista de clientes que han comprado estos productos.	SELECT c.Nombre, c.Apellidos, a.Nombre FROM Clientes c JOIN Ventas v ON c.ID_Cliente = v.ID_Cliente JOIN Articulos a ON v.ID_Articulo = a.ID_Articulo WHERE a.Nombre IN ('tomates', 'pimientos');
Clientes cuyo nombre empieza por "S" y compran tomates o pimientos	Filtra los clientes cuyo nombre inicia con "S" y han comprado estos productos.	SELECT c.Nombre, c.Apellidos, a.Nombre FROM Clientes c JOIN Ventas v ON c.ID_Cliente = v.ID_Cliente JOIN Articulos a ON v.ID_Articulo = a.ID_Articulo WHERE a.Nombre IN ('tomates', 'pimientos') AND c.Nombre LIKE 'S%';
Clientes que compran más de 10 tomates	Lista de clientes que han comprado más de 10 tomates.	SELECT c.*, a.Nombre, v.Cantidad FROM Clientes c JOIN Ventas v ON c.ID_Cliente = v.ID_Cliente JOIN Articulos a ON v.ID_Articulo = a.ID_Articulo WHERE a.Nombre = 'tomates' AND v.Cantidad > 10;