

```
CREATE TABLE Libros (  
    ID_Libro INTEGER PRIMARY KEY AUTOINCREMENT,  
    Titulo VARCHAR(255),  
    ISBN VARCHAR(13),  
    Editorial VARCHAR(100),  
    Año_Publicación INT,  
    Ejemplares_Disponibles INT  
);  
  
CREATE TABLE Autores (  
    ID_Autor INTEGER PRIMARY KEY AUTOINCREMENT,  
    Nombre VARCHAR(100),  
    Apellidos VARCHAR(100),  
    Fecha_Nacimiento DATE,  
    Nacionalidad VARCHAR(100)  
);  
  
CREATE TABLE Libro_Autor (  
    ID_Libro INTEGER,  
    ID_Autor INTEGER,  
    PRIMARY KEY (ID_Libro, ID_Autor),  
    FOREIGN KEY (ID_Libro) REFERENCES Libros(ID_Libro),  
    FOREIGN KEY (ID_Autor) REFERENCES Autores(ID_Autor)  
);
```

```
CREATE TABLE Socios (  
    ID_Socio INTEGER PRIMARY KEY AUTOINCREMENT,  
    Nombre VARCHAR(100),  
    Apellidos VARCHAR(100),  
    Dirección VARCHAR(255),  
    Teléfono VARCHAR(15),  
    Email VARCHAR(100),  
    Fecha_Alta DATE  
);  
  
CREATE TABLE Prestamos (  
    ID_Prestamo INTEGER PRIMARY KEY AUTOINCREMENT,  
    ID_Socio INT,  
    Fecha_Prestamo DATE,  
    Fecha_Devolución DATE,  
    Estado VARCHAR(20),  
    FOREIGN KEY (ID_Socio) REFERENCES Socios(ID_Socio)  
);  
  
CREATE TABLE Prestamo_Libro (  
    ID_Prestamo INT,  
    ID_Libro INT,  
    Cantidad INT,
```

```
PRIMARY KEY (ID_Prestamo, ID_Libro),  
FOREIGN KEY (ID_Prestamo) REFERENCES Prestamos(ID_Prestamo),  
FOREIGN KEY (ID_Libro) REFERENCES Libros(ID_Libro)  
);
```

Insert

- Se pueden realizar múltiples inserts en 1 query
 - Es posible obviar los campos si son todos.
 - Campos autoincrementales no se especifican
-

```
INSERT INTO Libros (ID_Libro, Titulo, ISBN, Editorial, Año_Publicación,  
Ejemplares_Disponibles) VALUES  
(1, 'Dune', '978-0441013593', 'Chilton Books', 1965, 5),  
(2, 'El Hobbit', '978-0618968633', 'Allen & Unwin', 1937, 3),  
  
INSERT INTO Autores (Nombre, Apellidos, Fecha_Nacimiento, Fecha_Fallecimiento,  
Nacionalidad) VALUES  
( 'Frank', 'Herbert', '1920-10-08', '1986-02-11', 'Estadounidense'),  
( 'J.R.R.', 'Tolkien', '1892-01-03', '1973-09-02', 'Británico')  
  
INSERT INTO Libro_Autor (ID_Libro, ID_Autor) VALUES  
(1,1),  
(2,2)
```

ALTER

Agregar un columna

```
ALTER TABLE empleados ADD columna_edad INT;
```

Modificar el tipo de dato de una columna existente:

```
ALTER TABLE empleados MODIFY columna_edad DECIMAL(5, 2);
```

Cambiar el nombre de una columna:

```
ALTER TABLE empleados CHANGE columna_edad edad_empleado INT;
```

Eliminar una columna de una tabla:

```
ALTER TABLE empleados DROP COLUMN edad_empleado;
```

Agregar una restricción de clave primaria:

```
ALTER TABLE empleados ADD CONSTRAINT pk_empleados PRIMARY KEY (id_empleado);
```

Delete

El DELETE en SQL és una instrucció que s'utilitza per **eliminar files** d'una taula en una base de dades. Aquesta operació és irreversible, és a dir, un cop eliminada la fila, no es pot recuperar, excepte si es té una còpia de seguretat de la base de dades.

```
DELETE FROM Libros  
DELETE FROM Libros WHERE ID_Libro = 3  
DELETE FROM Libros WHERE Titulo = "Dune"
```

UPDATE

UPDATE en SQL és una instrucció que s'utilitza per modificar els valors de les files existents en una taula. Aquesta operació permet actualitzar dades específiques basant-se en una condició

Actualizar un campo de todos los registros.

```
UPDATE Libros SET Ejemplares_Disponibles = 5
```

Permite actualizar datos de ciertos registros

```
UPDATE Libros SET Ejemplares_Disponibles = 5 WHERE ID_Libro = 1
```

Aumentar salario un 10% a ciertos trabajadores

```
UPDATE empleados SET salario = salario * 1.10 WHERE fecha_ingreso < '2015-01-01';
```

SELECTS

Permite recuperar registros de 1 o más tablas.

1. Select simple: (Devuelve todos los campos de todos los registros)

```
SELECT * FROM Libros
SELECT * FROM Autores
```

2. Select campos: (Devuelve ciertos campos de todos los registros)

```
SELECT Titulo FROM Libros
SELECT Nombre, Apellidos FROM AUTORES
```

-
3. Select con cláusula **WHERE**:

```
SELECT Nombre, Apellidos, Fecha_Nacimiento
FROM Autores
WHERE Fecha_Nacimiento > '1900-01-01';
```

4. Select con operador **LIKE**:

```
SELECT Titulo
FROM Libros
WHERE Titulo LIKE '%Dune%';
```

Select con operadores lógicos (**AND**, **OR**)

6 o más ejemplares y publicados a partir de 1951 (incluido)

```
SELECT Titulo, Ejemplares_Disponibles, Año_Publicacion
FROM Libros
WHERE Ejemplares_Disponibles > 5 AND Año_Publicacion > 1950;
```

Título que comienza por D o con 7 o más ejemplares disponibles.

```
SELECT Titulo, Ejemplares_Disponibles
FROM Libros
WHERE Titulo LIKE 'D%' OR Ejemplares_Disponibles > 6;
```

7. Select con ORDER BY (ASC por defecto, DESC opcional)

```
SELECT Titulo, Ejemplares_Disponibles
FROM Libros
ORDER BY Titulo
```

8. Select con LIMIT

```
SELECT Titulo, Ejemplares_Disponibles
FROM Libros
LIMIT 3;
```

9. Select con COUNT

```
SELECT COUNT(*) AS Total_titulos
FROM Libros
```

Select con SUM

```
SELECT SUM(Ejemplares_Disponibles) AS Total_libros FROM Libros
```

La consulta SQL suma tots els valors de la columna Ejemplares_Disponibles de la taula Libros i mostra el resultat total amb el nom d'Total_libros.

Per exemple, si la taula Libros té tres llibres amb 3, 5 i 2 exemplars disponibles, el resultat de la consulta serà:

```
Total_libros
-----
10
```

Select con COUNT

Així, la consulta retorna el nombre total de títols disponibles (llibres diferents)

```
SELECT COUNT(*) AS Total_titulos FROM Libros
```

Select con COUNT y SUM

Així, la consulta retorna el nombre total d'exemplars disponibles de tots els llibres combinats.

```
SELECT COUNT(*) AS Total_titulos, SUM(Ejemplares_Disponibles) AS Total_libros FROM Libros
```

Resum:

Aquesta consulta retorna dues coses:

- **Total_titulos**: El nombre total de títols (llibres) que hi ha a la taula **Libros**. S'obté mitjançant la funció **COUNT(*)**.
- **Total_libros**: El nombre total d'exemplars disponibles de tots els llibres combinats. Es calcula amb la funció **SUM(Ejemplares_Disponibles)**.

Exemple:

Imagina que tens la següent taula **Libros**:

Título	Ejemplares_Disponibles
Llibre A	3
Llibre B	5
Llibre C	2

El resultat de la consulta serà:

Total_titulos	Total_libros
3	10

JOIN

Un JOIN en SQL és una operació que permet combinar dades de **dues o més taules** en una consulta, basant-se en una columna comuna entre elles. Això és útil quan les dades estan distribuïdes en diverses taules relacionades, i vols obtenir informació combinada.

Selección de tablas

Al querer información de libros con sus autores, por fuerza debremos hacer referencia a las tres tablas, incluída Libro_autor, pues es la que relaciona a las otras dos.

```
SELECT *  
FROM Libros  
JOIN Libro_Autor  
JOIN Autores
```

Select todos con todos

Seleccionamos únicamente los campos que nos interesan.

```
SELECT L.ID_Libro, L.Titulo, LA.ID_Libro, LA.ID_Autor, A.ID_Autor, A.Nombre  
FROM Libros L  
JOIN Libro_Autor  
JOIN Autores A
```

Select filtrado

Ahora impondremos con las clausulas ON las condiciones para filtrar únicamente las filas que cumplan las condiciones.

```
SELECT L.ID_Libro, L.Titulo, LA.ID_Libro, LA.ID_Autor, A.ID_Autor, A.Nombre  
FROM Libros L  
JOIN Libro_Autor LA ON L.ID_Libro = LA.ID_libro  
JOIN Autores A ON LA.ID_Autor = A.ID_Autor
```

Select con campos y Orden

Finalmente seleccionamos solo los campos que queremos mostrar, y ordenamos por título en orden alfabético ascendente.

```
SELECT L.Titulo, A.Nombre, A.Apellidos  
FROM Libros L
```

```
JOIN Libro_Autor LA ON L.ID_Libro = LA.ID_Libro  
JOIN Autores A ON LA.ID_Autor = A.ID_Autor  
ORDER BY L.Titulo;
```