

Bases de datos



Menú del día

- Motor **SQLite**
- Introducción a **SQL**:
 - Definición de Datos
 - Manipulación de Datos

SQLite

Es un **motor de bases de datos liviano** diseñado para **administrar BD relacionales**.

SQLite es ampliamente utilizado en la industria, incluyendo software como navegadores de internet, sistemas operativos y teléfonos.



DB Browser

- <http://sqlitebrowser.org/>

The image shows the logo for 'DB Browser for SQLite'. It features the text 'DB Browser for SQLite' in a white, handwritten-style font against a blue background with a light grid pattern. Below the main title, the text 'The Official home of the DB Browser for SQLite' is written in a smaller, light blue, handwritten-style font.

DB Browser
for SQLite

The Official home of the DB Browser for
SQLite

SQL

Structured Query Language



SQL

Es un **lenguaje de programación** diseñado para **administrar DB relacionales**.

Se lo considera un lenguaje **declarativo**.

→ **Lenguaje de Definición de Datos**
Creación de tablas y registros.

→ **Lenguaje de Manipulación de Datos**
Consulta, actualización y borrado de datos.

Creación de Tablas

Se utiliza la sentencia **CREATE TABLE**:

```
CREATE TABLE NombreDeLaTabla(  
    NombreColumna1 tipoDeDato PRIMARY KEY,  
    NombreColumna2 tipoDeDato,  
    NombreColumna3 tipoDeDato NOT NULL  
);
```

Las instrucciones detalladas en azul son optativas. **PRIMARY KEY**: indica que la columna es la primary key de la tabla e indica que la columna se auto-incrementará automáticamente en caso de no proporcionar dicho valor y que el tipo de dato sea numérico.

Tipos de datos comunes: **INTEGER, BOOL, DOUBLE, FLOAT, TEXT.**

Creación de Tablas

Ejemplo:

```
CREATE TABLE Clientes(  
    DNI TEXT PRIMARY KEY,  
    Nombre TEXT NOT NULL,  
    Apellido TEXT NOT NULL  
);
```

Clientes
DNI (PK)
Nombre
Apellido

Creación de Tablas

Ejemplo:

```
CREATE TABLE Productos(  
    SKU TEXT PRIMARY KEY,  
    Nombre TEXT NOT NULL  
);
```

Productos
SKU (PK) Nombre

Creación de Relaciones

```
CREATE TABLE NombreDeLaTabla(  
    Columna1 tipoDeDato,  
    Columna2 tipoDeDato,  
    Columna3 tipoDeDato,  
    PRIMARY KEY(Columna1, Columna2),  
    FOREIGN KEY(Columna3) REFERENCES NombreDeOtraTabla(Columna)  
);
```

Creación de Relaciones

Ejemplo:

```
CREATE TABLE Compras(  
    SKU TEXT,  
    DNI TEXT,  
    Fecha DATE,  
    PRIMARY KEY(DNI, SKU, Fecha),  
    FOREIGN KEY(DNI) REFERENCES Clientes(DNI),  
    FOREIGN KEY(SKU) REFERENCES Productos(SKU)  
);
```

Inserción de registros

Se utiliza la sentencia **INSERT INTO**

Estructura:

```
INSERT INTO NombreDeLaTabla VALUES(  
    valor1, valor2, valor3  
);
```

Los valores deben ir **en el mismo orden** que se encuentran las columnas.

Inserción de registros

Se utiliza la sentencia **INSERT INTO**

Estructura:

```
INSERT INTO NombreDeLaTabla (Columna2, Columna3) VALUES(  
    valorColumna2, valorColumna3  
);
```

Los valores deben ir **en el mismo orden** en que proporcionamos las columnas.

Inserción de registros

Ejemplos:

```
INSERT INTO Clientes(DNI,Nombre,Apellido) VALUES (  
    '3552222', 'Mariano', 'Perez'  
);
```

```
INSERT INTO Productos(SKU,Nombre) VALUES (  
    '12', 'Lentes Reef'  
);
```

Los valores de tipo **TEXT** se expresan entre comillas simples.

Selección de registros

Para hacer consultas sobre los registros de la tabla y seleccionar datos, se utiliza la sentencia **SELECT**. El resultado de la consulta es una nueva tabla que contiene la información solicitada.

Estructura:

```
SELECT column1, column2,..., columnN  
FROM table_name  
WHERE condition  
ORDER BY column1 ASC/DESC;
```

Las instrucciones detalladas en azul son optativas.

WHERE: permite especificar alguna condición para filtrar datos.

ORDER BY: permite ordenar los resultados obtenidos en forma ascendente (ASC) o descendente (DESC)

Se usa el * para informar a la consulta que quiero todas las columnas.

Selección de registros

Ejemplos

```
SELECT *  
FROM Clientes;
```

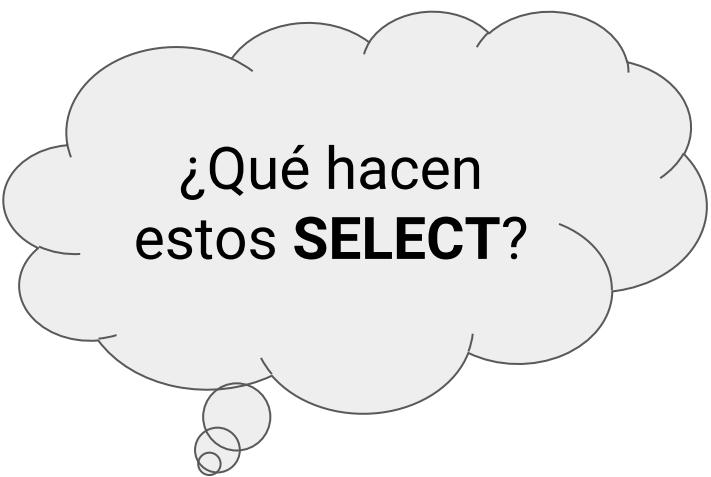
```
SELECT DNI  
FROM Clientes;
```


Selección de registros

Ejemplos

```
SELECT *  
FROM Productos  
WHERE SKU = '12';
```

```
SELECT *  
FROM Compras  
WHERE Fecha > '2016-03-03';
```



¿Qué hacen
estos **SELECT**?

Modificación de registros

Para modificar registros de la tabla se utiliza la sentencia **UPDATE**

Estructura:

```
UPDATE NombreDeLaTabla  
SET Columna1 = nuevoValor1, Columna2 = nuevoValor2  
WHERE condition;
```


Las instrucciones detalladas en azul son optativas.

WHERE: permite especificar alguna condición para filtrar datos.

Modificación de registros

Ejemplo:

```
UPDATE Clientes  
SET nombre = 'Alberto'  
WHERE DNI = '34522122';
```



¿Qué hace este
UPDATE?

Borrado de registros

Para eliminar registros de la tabla se utiliza la sentencia **DELETE**

Estructura:

```
DELETE  
FROM NombreDeLaTabla  
WHERE condition;
```

Las instrucciones detalladas en azul son optativas.

WHERE: permite especificar alguna condición para filtrar datos.

Borrado de registros

Ejemplo

```
DELETE  
FROM Cliente  
WHERE DNI = '34522122';
```

```
DELETE  
FROM Producto  
WHERE SKU = '12';
```



Borrado de registros

Ejemplo

```
DELETE  
FROM Cliente
```

```
DELETE  
FROM Cliente  
WHERE Apellido = 'Lee';
```



¿Qué hacen
estos **DELETE**?