BASES DE DATOS CON SQL

Y algunas consultas para comenzar...

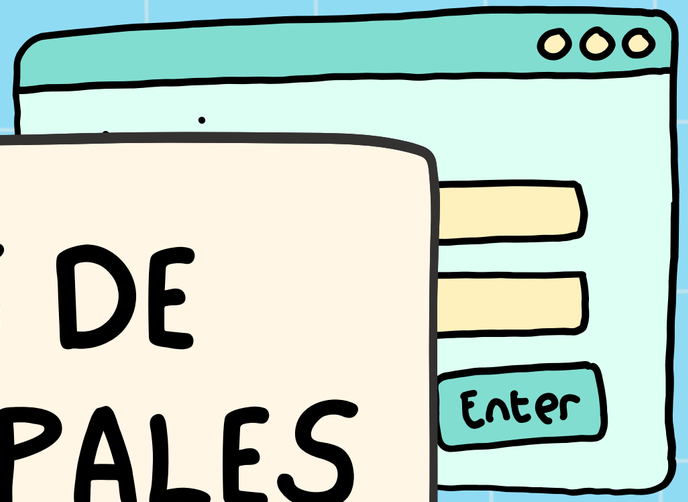
¿QUÉ ES UNA BASE DE DATOS?

Una base de datos es una colección organizada de información estructurada, o datos, que normalmente se almacena electrónicamente en las tablas de un sistema.

¿QUÉ ES SQL?

Es un lenguaje de programación para almacenar y procesar información en una base de datos.

TRES TIPOS DE DATOS PRINCIPALES



INT

Almacena números enteros sin parte decimal.

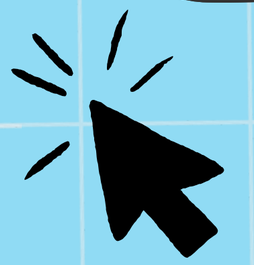
VARCHAR

Almacena cadenas de caracteres (texto) de longitud variable.

DATE

Almacena fechas con formato Año-Mes-Día

¿PARA QUÉ SIRVE SQL?



- Consultar datos de la base de datos
- Insertar datos en la base de datos
- Actualizar datos de la base de datos
- Eliminar datos de la base de datos

¿QUÉ ES UNA PRIMARY KEY?

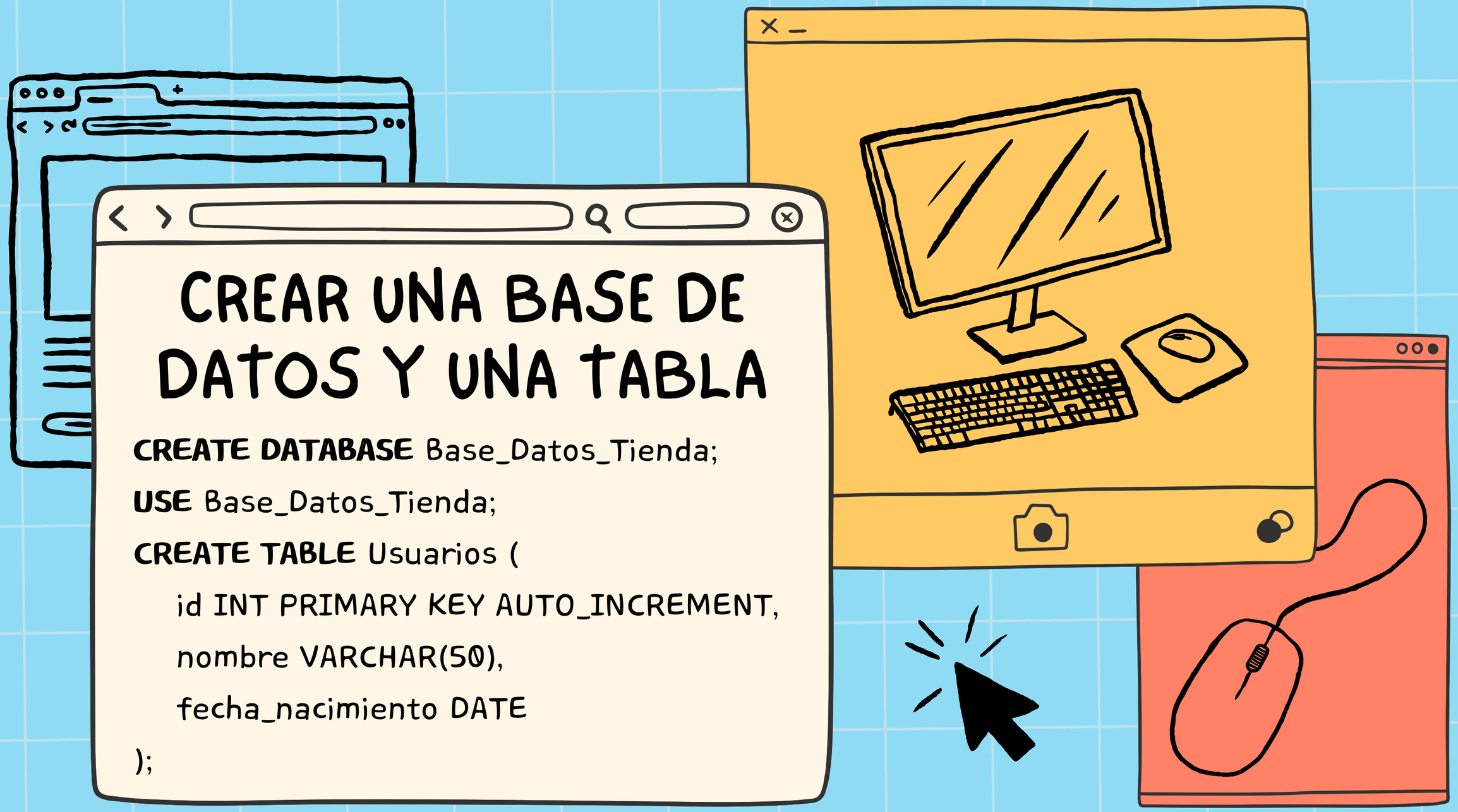
Una clave primaria (PRIMARY KEY) es la columna (o conjunto de columnas) que identifica de forma única cada fila en una tabla. Los valores de la clave primaria deben ser únicos y no pueden ser nulos NULL.

¿QUÉ ES UNA FOREIGN KEY?

Es una columna o conjunto de columnas en una tabla de base de datos que establece un vínculo con la clave primaria de otra tabla.

CREAR UNA BASE DE DATOS Y UNA TABLA

```
CREATE DATABASE Base_Datos_Tienda;
USE Base_Datos_Tienda;
CREATE TABLE Usuarios (
    id INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(50),
    fecha_nacimiento DATE
);
```



ESTRUCTURA DE UNA CONSULTA

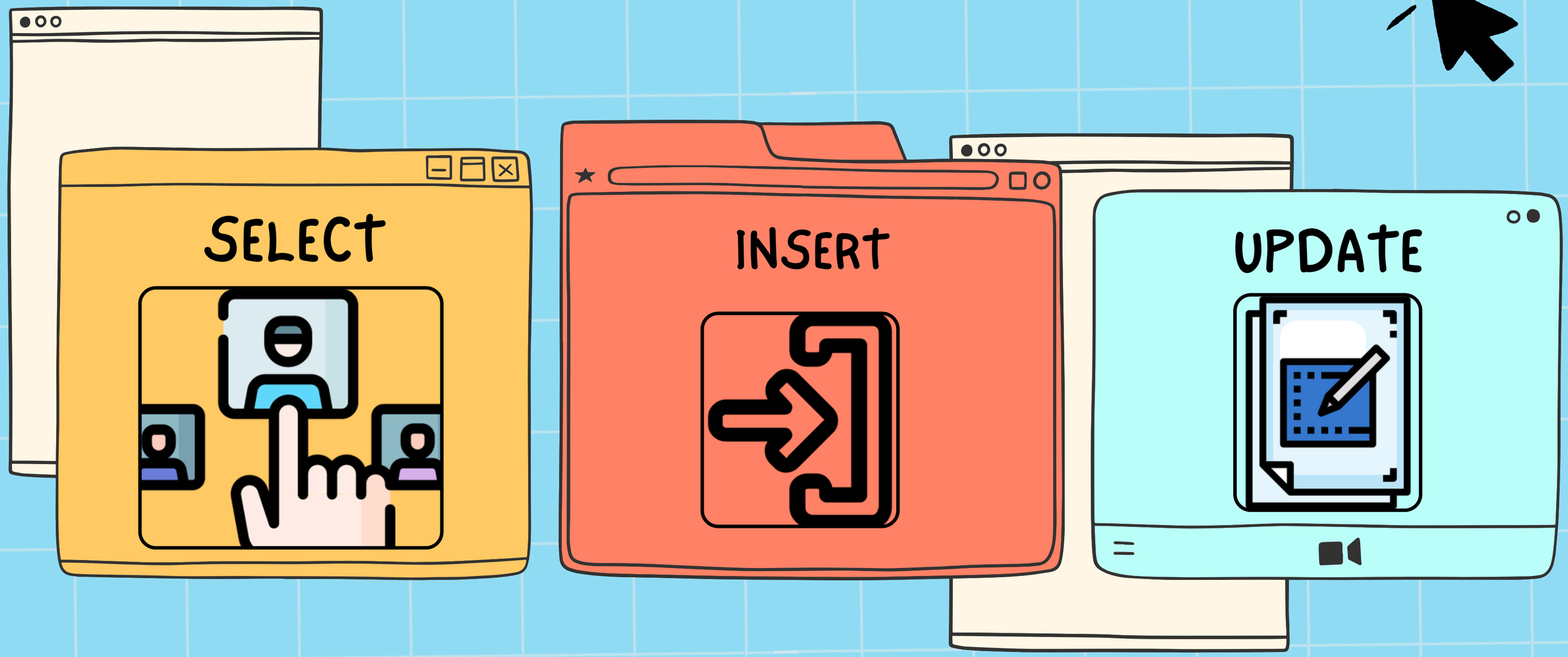
SELECT
columna1

FROM
nombre_tabla

WHERE
condicion

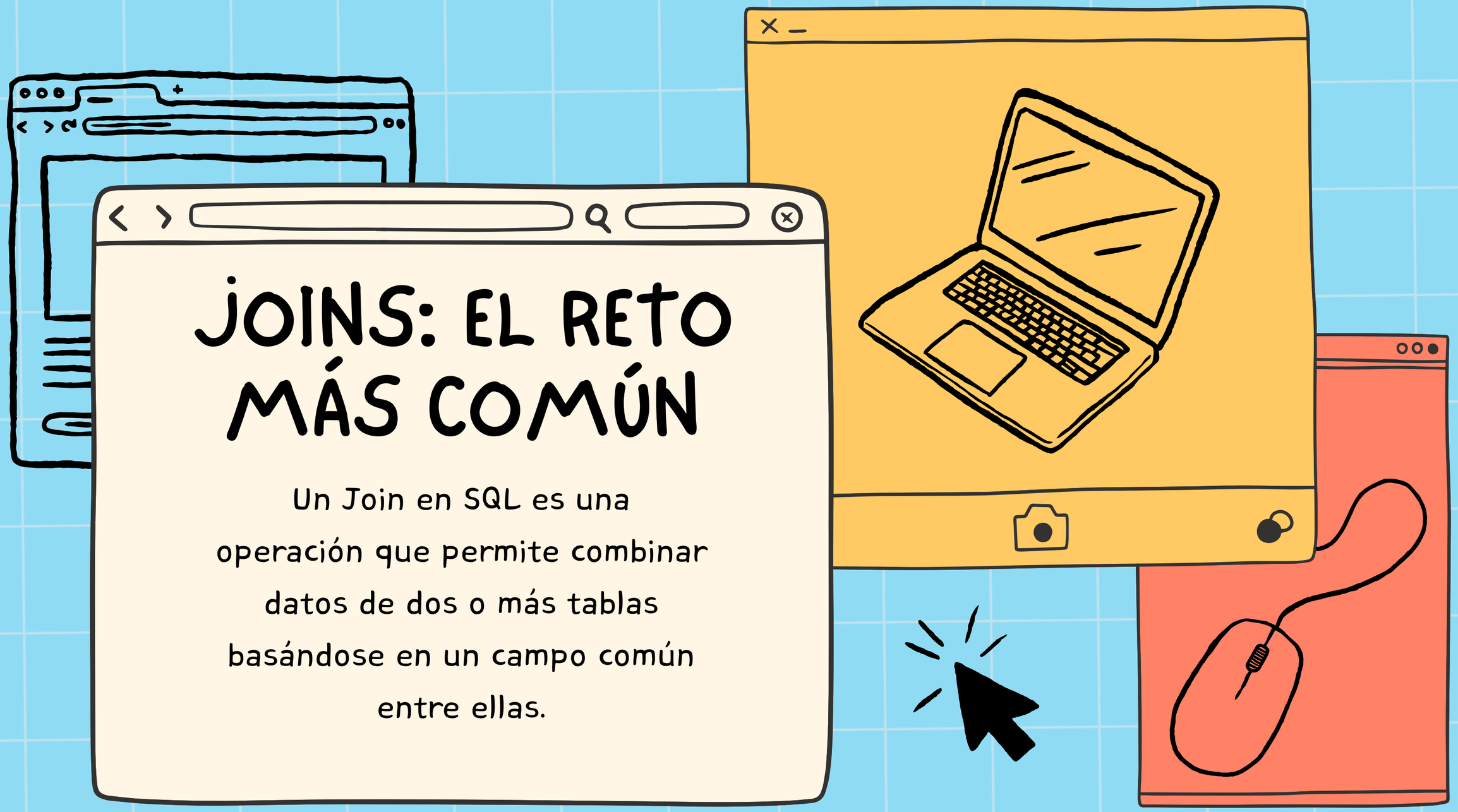
ORDER BY
[ASC | DESC];

COMANDOS BÁSICOS



JOINS: EL RETO MÁS COMÚN

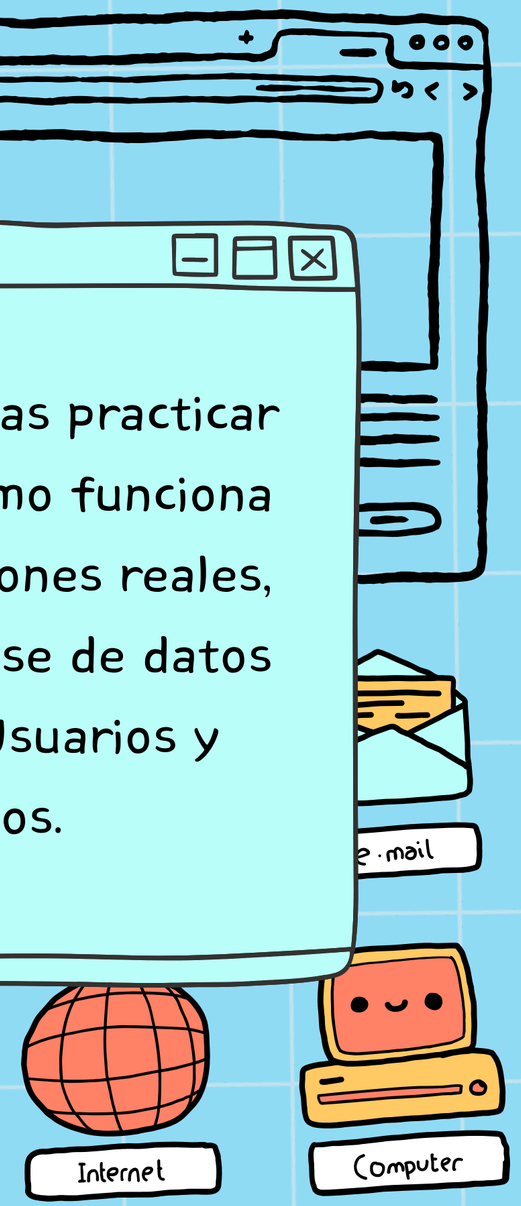
Un Join en SQL es una operación que permite combinar datos de dos o más tablas basándose en un campo común entre ellas.



CONSULTAS EJEMPLO

Para que puedas practicar y entender cómo funciona SQL en situaciones reales, usando una base de datos sencilla de Usuarios y Pedidos.

```
--TABLAS QUE USAREMOS
Usuarios (id, nombre, fecha_nacimiento)
Pedidos (id, usuario_id, producto)
```



EJEMPLO #1

```
USE Base_Datos_Tienda;

SELECT id, nombre, fecha_nacimiento
FROM Usuarios
WHERE fecha_nacimiento >= '2000-01-01'
ORDER BY fecha_nacimiento ASC;
```

EL RESULTADO ESPERADO SERÍA ASÍ

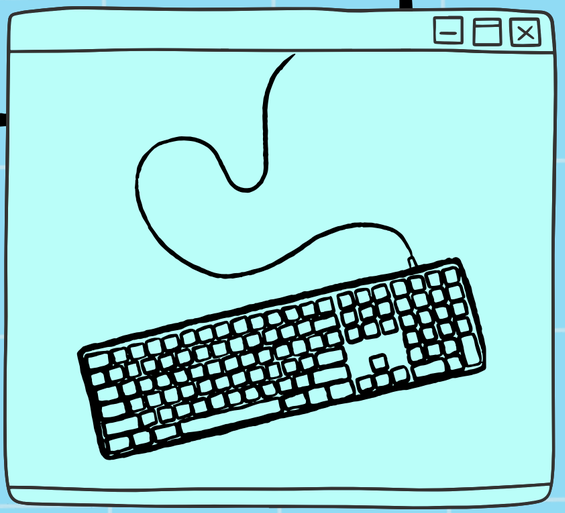
nombre	fecha_nacimiento
Ana López	2000-03-15
Carlos Ruiz	2003-08-22
Sofía Méndez	2005-01-10

EJEMPLO #2

Con **INSERT** agregas un registro nuevo. Con **UPDATE** modificas uno existente, con **SET** indicas que columna vas a cambiar y cual será su nuevo valor, el **WHERE** es clave para no editar toda la tabla por accidente.

```
USE Base_Datos_Tienda;
-- Insertar un nuevo usuario
INSERT INTO Usuarios (nombre,
fecha_nacimiento)
VALUES ('Luis Torres', '2001-07-14');

-- Actualizar el nombre de un usuario
UPDATE Usuarios SET nombre = 'Luis A.
Torres' WHERE id = 3;
```



Recuerda que la tabla usuarios la creamos antes.

El **INNER JOIN** combina ambas tablas y solo muestra los usuarios que tienen al menos un pedido registrado.

IMPORTANTE: Después de el JOIN siempre va un ON.

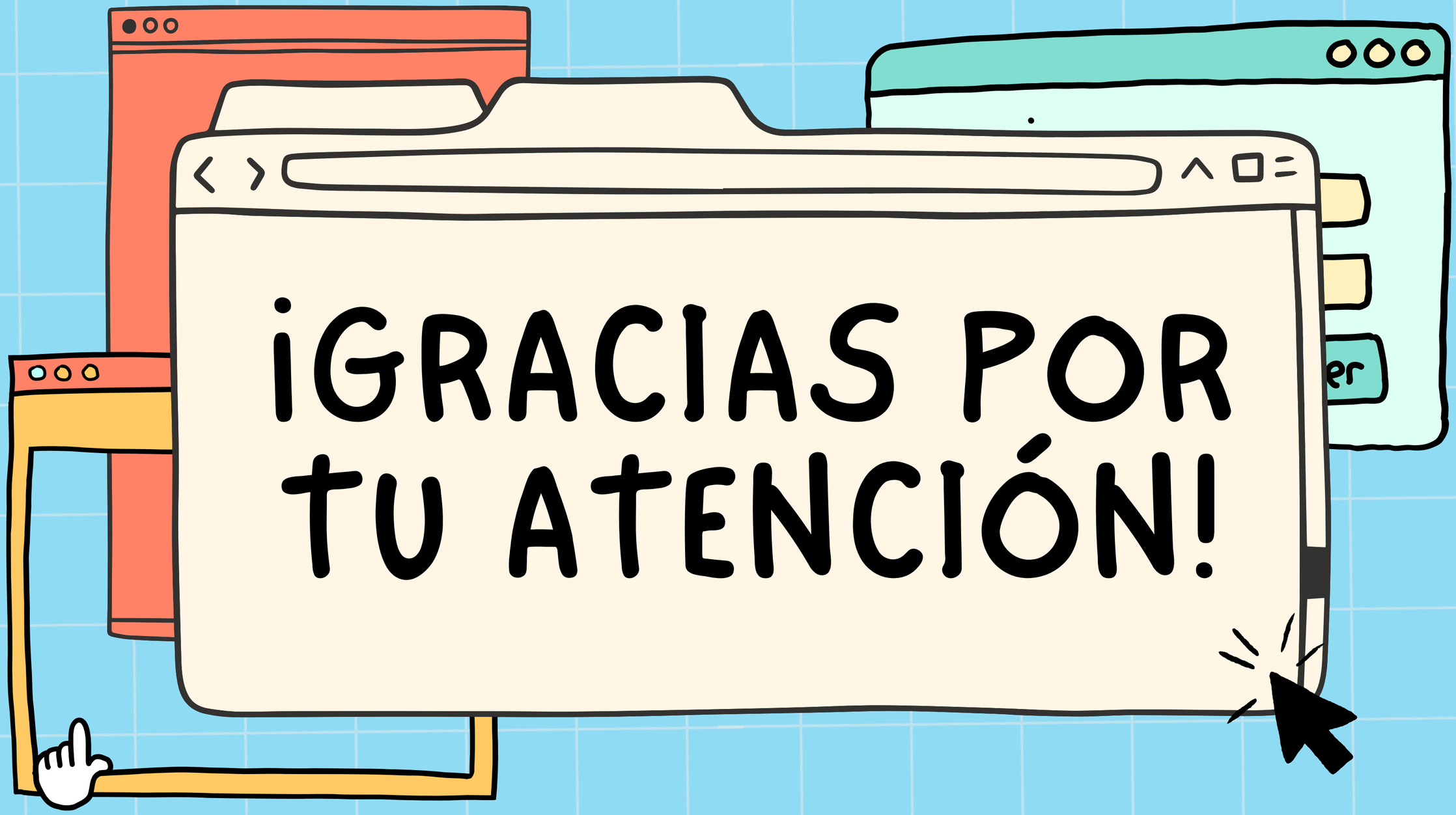
```
USE Base_Datos_Tienda;
--CREAMOS OTRA BASE DE DATOS
CREATE TABLE Pedidos (
    id INT PRIMARY KEY AUTO_INCREMENT,
    usuario_id INT, producto VARCHAR(50),
    FOREIGN KEY (usuario_id) REFERENCES Usuarios(id)
);
```

```
USE Base_Datos_Tienda;
--Consulta con JOIN
SELECT Usuarios.nombre, Pedidos.producto
FROM Usuarios
INNER JOIN Pedidos
ON Usuarios.id = Pedidos.usuario_id;
```

--Así deberia de verse

nombre	producto
Ana López	Laptop
Carlos Ruiz	Mouse
Luis Torres	Teclado

EJEMPLO #3


¡GRACIAS POR
TU ATENCIÓN!