Logotipo

El contenido generado por IA puede ser incorrecto.



Santiago Yosa González

**Ingeniero Sistemas de Información**

Logotipo

El contenido generado por IA puede ser incorrecto.}

**Creación y Administración de Bases de Datos**

* DDL: definición de un lenguaje,
* DML: manipulación de un lenguaje
* DCL: control de un lenguaje
* El Lenguaje SQL
* Creación de una base de datos en SQL Server
* Importación y exportación de bases de datos
* Creación de tablas
* Tipos de datos
* Tipos de Relaciones, clave primaria y clave foránea

**Unidad II**

**¿Qué es SQL y su división (El Lenguaje)?**

SQL(Structured Query Language) no es un lenguaje de programación tradicional (como Python o Java), es un lenguaje **declarativo**: le dices *qué* quieres, no *cómo* hacerlo.

Existen varios "dialectos" o variantes de SQL desarrollados por distintos proveedores (MySQL, PostgreSQL, Oracle, SQL Server), cada uno con sus propias sintaxis, funciones y optimizaciones. Estas variantes se usan según el sistema de gestión de bases de datos (SGBD) elegido.

Los principales dialectos y sistemas SQL son:

* **MySQL:** Popular en aplicaciones web, código abierto y conocido por su velocidad.
* **PostgreSQL:** Muy robusto, código abierto, enfocado en compatibilidad estándar y consultas complejas.
* **Microsoft SQL Server:** Solución propietaria de Microsoft, común en entornos corporativos.
* **Oracle Database:** Utilizado en grandes empresas por su alto rendimiento y escalabilidad.
* **SQLite:** Ligero y basado en archivos, ideal para aplicaciones móviles y pequeños proyectos.

**SQL se divide en tres grandes sublenguajes**

**DDL – Lenguaje de Definición de Datos**

Es el lenguaje del "Arquitecto". Sirve para crear, modificar o destruir las estructuras (tablas, bases de datos).

**Principales comandos DDL:**

| **Comando** | **Propósito** |
| --- | --- |
| CREATE | Crear una nueva base de datos o tabla |
| ALTER | Modificar una tabla existente |
| DROP | Eliminar una tabla o base de datos |
| TRUNCATE | Eliminar datos de una tabla sin borrar la tabla |

**DML – Lenguaje de Manipulación de Datos**

Es el lenguaje del "Usuario/Empleado". Sirve para insertar, leer, actualizar o borrar la información que va *dentro* de las estructuras.

**Principales comandos DML:**

|  |  |
| --- | --- |
| **Comando** | **Propósito** |
| INSERT | Agrega nuevos datos a una tabla |
| SELECT | Consulta datos de una o más tablas |
| UPDATE | Modifica datos existentes |
| DELETE | Elimina registros |

**DROP vs TRUNCATE vs DELETE:**

* **DROP TABLE:** Destruye el cajón de madera y todo lo que hay dentro. (Comando DDL).
* **TRUNCATE TABLE:** Voltea el cajón y vacía todo el contenido de golpe, reseteando el contador IDENTITY a 1, pero el cajón sigue existiendo. (Comando DDL).
* **DELETE FROM:** Saca los elementos del cajón uno por uno. El cajón sigue existiendo y el contador IDENTITY **no** se reinicia. (Comando DML).

**DCL – Lenguaje de Control de Datos**

Es el lenguaje del "Guardia de Seguridad". Da o quita permisos a los usuarios.

**Principales comandos DCL:**

|  |  |
| --- | --- |
| **Comando** | **Propósito** |
| GRANT | Concede permisos a un usuario |
| REVOKE | Revoca permisos anteriormente concedidos |

**DDL:** define la estructura de la base de datos.

**DML:** permite manipular los datos que contiene.

**DCL:** administra permisos y seguridad.

**Tipos de Datos**

SQL es fuertemente tipado. Si dices que una columna es un número, e intentas meter la letra "A", el sistema generará un error fatal. Esto garantiza la integridad.

|  |  |  |
| --- | --- | --- |
| **Tipo de Dato** | **Descripción** | **Ejemplo** |
| INT | Número entero | 1, 25, 100 |
| VARCHAR(n) | Cadena de texto de hasta n caracteres | 'Hola', 'Ana' |
| DECIMAL(p,s) | Números decimales | 199.99, 15.50 |
| DATE | Fecha | '2025-03-28' |
| BIT | Valor booleano (0 o 1) | 0, 1 |

[Más tipos de datos](https://learn.microsoft.com/es-es/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-ver16).

**CHULETA**

|  |  |  |
| --- | --- | --- |
| Comando | Caso de Uso (¿Para qué sirve?) | Comando SQL (Ejemplo Práctico) |
| CREATE DATABASE | Crear la base de datos para alojar todas las tablas de un proyecto. | CREATE DATABASE SistemaVentas; |
| CREATE TABLE (Básica) | Crear una tabla independiente definiendo su Llave Primaria (PK) autoincremental y sus columnas. | CREATE TABLE Categorias ( CategoriaID INT PRIMARY KEY IDENTITY(1,1), Nombre NVARCHAR(100) NOT NULL); |
| CREATE TABLE (Con FK) | Crear una tabla dependiente, que requiere conectarse obligatoriamente a otra tabla mediante una Llave Foránea. | CREATE TABLE Productos ( ProductoID INT PRIMARY KEY IDENTITY(1,1), Nombre NVARCHAR(150) NOT NULL, CategoriaID INT NOT NULL, CONSTRAINT FK\_Prod\_Cat FOREIGN KEY (CategoriaID) REFERENCES Categorias(CategoriaID)); |
| ALTER TABLE ... ADD | Agregar una nueva columna a una tabla que ya existe. (Nacerá permitiendo nulos por defecto). | ALTER TABLE Productos ADD FechaVencimiento DATE; |
| ALTER TABLE ... DROP COLUMN | Eliminar una columna entera de la base de datos. ¡Peligro! Destruye todos los datos almacenados en esa columna. | ALTER TABLE Productos DROP COLUMN FechaVencimiento; |
| ALTER TABLE ... ALTER COLUMN | Cambiar el tipo de dato de una columna (Ej: Cambiar de VARCHAR(50) a NVARCHAR(200) para hacerla más grande). | ALTER TABLE ProductosALTER COLUMN Nombre NVARCHAR(200) NOT NULL; |
| ALTER TABLE ... ADD CONSTRAINT | Inyectar una Llave Foránea (FK) a una tabla que ya fue creada previamente sin ella. | ALTER TABLE ProductosADD CONSTRAINT FK\_Prod\_Cat FOREIGN KEY (CategoriaID) REFERENCES Categorias(CategoriaID); |
| DROP TABLE | Eliminar una tabla completa y todos los registros que contenga adentro. | DROP TABLE Productos; |
| DROP TABLE IF EXISTS | Eliminar una tabla de forma segura solo si existe (evita que el script lance un error rojo si la tabla ya había sido borrada). | DROP TABLE IF EXISTS Productos; |
| DROP DATABASE | Destruir la base de datos completa con todas sus tablas, configuraciones y datos. | DROP DATABASE SistemaVentas; (Nota: No funciona si tienes esa BD en uso actualmente). |

**Tipos de Relaciones, Clave Primaria y Clave Foránea**

**Clave primaria (Primary Key)**

Identifica de forma única cada registro. Asegura que el dato sea único y no nulo. En SQL Server solemos usar la propiedad IDENTITY(1,1) para que el motor asigne los números automáticamente (1, 2, 3...).

id INT PRIMARY KEY

**Clave foránea (Foreign Key)**

Establece una relación con otra tabla. Es el puente. Una regla que dice: "El valor que intentes meter aquí, DEBE existir obligatoriamente en la PK de la otra tabla".

FOREIGN KEY (id\_cliente) REFERENCES Clientes(id)

**Orden lógico y técnico de los comandos SQL**

DDL (Definir estructura), primero se usa el DDL, porque no se puede manipular datos si no existe una base de datos ni tablas. En SQL Server, antes de crear tablas, debemos crear la "Caja Mayor" (La Base de Datos) y decirle al sistema que la vamos a usar.

CREATE DATABASE TiendaOnline;

USE TiendaOnline;

CREATE TABLE Clientes (

id INT PRIMARY KEY,

nombre VARCHAR(50),

correo VARCHAR(100)

);

ALTER TABLE Clientes

ADD direccion VARCHAR(100);

DROP TABLE Clientes;

Si quisiera puedo crear una tabla y luego agregarle la FK o crearla con la FK de una vez:

CREATE TABLE Pedidos (

id INT PRIMARY KEY,

fecha DATE,

cliente\_id INT

);

ALTER TABLE Pedidos

ADD CONSTRAINT fk\_cliente

FOREIGN KEY (cliente\_id) REFERENCES Clientes(id);

CREATE TABLE Pedidos (

id INT PRIMARY KEY,

fecha DATE,

cliente\_id INT,

CONSTRAINT fk\_cliente

FOREIGN KEY (cliente\_id)

REFERENCES Clientes(id)

);

**Preguntas Frecuentes**

**¿Por qué no le pongo VARCHAR(MAX) a todas las columnas de texto para no complicarme calculando tamaños?**

Por rendimiento y seguridad. Si le dices al motor de SQL Server que un campo puede almacenar millones de caracteres (MAX), este reservará recursos de memoria ineficientes al hacer consultas (SELECTs). Además, a nivel de negocio, limitar un "Teléfono" a VARCHAR(15) evita que un usuario malicioso o despistado guarde un párrafo entero en ese campo.

**¿Qué pasa si intento hacer un DROP TABLE de una tabla que tiene datos?**

Si es una tabla "Padre" (ej. Clientes) y hay registros en una tabla "Hija" (ej. Facturas) que le pertenecen, SQL Server te bloqueará y lanzará un error de *Infracción de restricción de clave externa (Foreign Key constraint)*. El motor protege la integridad estructural para que no queden "Facturas huérfanas". Debes borrar primero a los hijos, o la tabla hija completa, antes de poder destruir al padre.

Aunque, existe un eliminado en cascada, pero se debe usar con mucho cuidado.

**ALTER TABLE**

Su propósito principal es **modificar la estructura de una tabla existente** en una base de datos.

**Agregar columna**

ALTER TABLE Empleados ADD correo\_electronico VARCHAR(100);

**Eliminar columna**

ALTER TABLE Clientes DROP COLUMN telefono\_fax;

**Actualizar una columna**

ALTER TABLE Productos ALTER COLUMN nombre\_producto VARCHAR(200);

**Agregar un constraint columna**

ALTER TABLE Pedidos ADD CONSTRAINT FK\_Pedidos\_Clientes FOREIGN KEY (id\_cliente) REFERENCES Clientes(id\_cliente);

Eliminar **un constraint columna**

ALTER TABLE Pedidos DROP CONSTRAINT FK\_Pedidos\_Clientes

**Demo - Sistema de Biblioteca Básico (1:N). Un autor escribe muchos libros.**

Primero, creamos la tabla independiente (Fuerte), que es el Autor. Luego, creamos la tabla dependiente (Débil), que es el Libro, e inyectamos la llave foránea.

CREATE DATABASE BibliotecaDemo;

GO

USE BibliotecaDemo;

GO

El motor de base de datos (SQL Server) no tiene ni idea de qué significa **GO**. Actúa como un **Separador de Lotes.** Lo deberíamos usar después de cada bloque lógico importante (especialmente en DDL).

* Después de un CREATE DATABASE.
* Después de un USE MiBaseDeDatos.
* Después de la creación de un grupo de tablas.
* Antes de crear: CREATE VIEW, CREATE PROCEDURE, CREATE TRIGGER y CREATE FUNCTION

Luego vamos a crear la tabla padre (No depende de nadie)

CREATE TABLE Autores (

AutorID INT PRIMARY KEY IDENTITY(1,1), -- Llave primaria autoincremental

Nombre VARCHAR(100) NOT NULL,

Nacionalidad VARCHAR(50)

);

Seguimos creando la tabla hija (Depende del Autor)

CREATE TABLE Libros (

LibroID INT PRIMARY KEY IDENTITY(1,1),

Titulo VARCHAR(200) NOT NULL,

FechaPublicacion DATE,

-- Aquí definimos la columna que servirá de puente:

AutorID INT,

-- Aquí definimos la REGLA (Constraint) de relación:

CONSTRAINT FK\_Libro\_Autor FOREIGN KEY (AutorID) REFERENCES Autores(AutorID)

);

**Práctica A:- *Learning by Doing***

Vamos a modelar una Red Social pequeña. Sabemos que un **Usuario** puede publicar muchos **Posts**.

* Abre un *New Query* en SQL Server Management Studio.
* Escribe el código DDL para crear la base de datos RedSocialDB.
* Crea la tabla Usuarios (debe tener ID, Nickname y Email).
* Crea la tabla Posts (debe tener ID, Contenido, Fecha de publicación y debe estar conectada al Usuario).

**Pista:**

* El usuario debería tener su identificación, al menos, nombre y correo.
* Los posts deberían tener una identificación, fecha de publicación y ¿cómo decimos que un post pertenece a un usuario?

**Práctica B**

Eres el desarrollador de un sistema de Recursos Humanos.

Crea un script SQL que:

1. Cree la base de datos RecursosHumanosDB.
2. Cree una tabla **Departamentos** (Ej: Sistemas, Ventas). Debe tener DeptoID y NombreArea.
3. Cree una tabla **Empleados** (Ej: Santiago, Ana). Debe tener EmpleadoID, NombreCompleto, Salario (Usa tipo de dato DECIMAL) y debe pertenecer a un Departamento obligatorio.

**Troubleshooting**

**Error 1:** There is already an object named 'Usuarios' in the database.

* ***Causa:*** Ejecutaste el bloque de código CREATE TABLE dos veces. SQL Server no puede crear algo que ya existe.
* ***Solución:*** Borra la instrucción o utiliza un comentario (--) para inutilizarla.

**Error 2:** Foreign key 'FK\_Post\_Usuario' references invalid table 'Usuarios'.

* ***Causa:*** Intentaste crear la tabla hija ANTES que la tabla padre. SQL Server lee de arriba hacia abajo. No puede referenciar algo que aún no ha sido creado en memoria.
* ***Solución:*** Invierte el orden de tus CREATE TABLE. Los padres siempre van primero

**Error 3:** Column 'UsuarioID' in table 'Posts' is of a type that is invalid for use as a key column in an index. (O error de compatibilidad de tipos).

* ***Causa:*** Si en la tabla Usuarios declaraste UsuarioID INT, y en la tabla Posts pones UsuarioID VARCHAR(10), el puente se rompe.
* ***Solución:*** Una Foreign Key **debe tener exactamente el mismo tipo de dato** que la Primary Key a la que apunta (si es INT, debe ser INT).

**Ejercicio Independiente**

Basado en lo que aprendiste hoy sobre DDL, diseña y escribe el script (archivo.sql) completo para el núcleo de un sistema **E-Commerce**. Debes crear la base de datos TiendaOnlineDB y las siguientes tablas, respetando la lógica de llaves primarias y foráneas:

1. **Categorias:** CategoriaID, NombreCategoria.
2. **Productos:** ProductoID, NombreProducto, Precio (Decimal), Stock (Entero), y debe conectarse con Categorias (Un producto pertenece a una categoría).
3. **Clientes:** ClienteID, Nombre, Correo.
4. **Pedidos:** PedidoID, FechaPedido, y debe conectarse con el Cliente (Un cliente hace muchos pedidos).
5. **DetallePedidos:** Esta es la tabla intermedia que resuelve el Muchos a Muchos entre Pedidos y Productos. Debe conectar PedidoID y ProductoID, además de guardar la Cantidad comprada.

**Criterios de logro esperados:**

* Script ejecutable de principio a fin sin errores de sintaxis (Comas, paréntesis).
* Tipos de datos coherentes (No usar VARCHAR para precios, ni INT para nombres).
* Las tablas Padre (Categorias, Clientes) se crean antes que las tablas Hijas (Productos, Pedidos).
* La tabla DetallePedidos tiene correctamente asignadas sus DOS llaves foráneas.

**Palabras Clave**

|  |  |
| --- | --- |
| **Palabra clave** | **¿Qué hace?** |
| SELECT | Selecciona datos de una tabla. |
| FROM | Indica de qué tabla se toman los datos. |
| WHERE | Aplica condiciones para filtrar los datos. |
| INSERT INTO | Inserta nuevos datos en una tabla. |
| VALUES | Especifica los valores a insertar. |
| UPDATE | Modifica datos existentes. |
| SET | Indica qué columnas se van a actualizar. |
| DELETE | Elimina registros de una tabla. |
| DROP | Elimina objetos (tablas, bases de datos). |
| CREATE | Crea objetos como tablas o bases de datos. |
| ALTER | Modifica la estructura de una tabla. |
| JOIN | Une datos de dos o más tablas. |
| ON | Define la condición de unión (JOIN). |
| GROUP BY | Agrupa resultados según una o varias columnas. |
| ORDER BY | Ordena los resultados. |
| HAVING | Filtra los grupos (después de un GROUP BY). |
| AS | Crea un alias para una columna o tabla. |
| DISTINCT | Elimina duplicados en los resultados. |
| LIKE | Busca patrones dentro de cadenas de texto. |
| IN | Compara un valor con una lista de valores. |
| BETWEEN | Filtra valores dentro de un rango. |
| IS NULL | Verifica si un valor está vacío (nulo). |

Logotipo

El contenido generado por IA puede ser incorrecto.