Universidad de Manizales
Facultad de Ciencias e Ingeniería
Bases de Datos I
Tema: Lenguaje de Definición de Datos (Mysql)
Profesor: John Alejandro Cardona Valencia

acardona@umanizales.edu.co

Lenguaje de Definición de Datos (DDL)

El **Lenguaje de Definición de Datos (DDL, Data Definition Language)** es un subconjunto del lenguaje SQL que se encarga de definir, modificar y eliminar la estructura de los objetos dentro de una base de datos. Es decir, permite crear, modificar y eliminar bases de datos, tablas, índices, vistas y otros elementos de almacenamiento.

A diferencia de otros tipos de comandos SQL, como los de **Manipulación de Datos (DML)** (INSERT, UPDATE, DELETE), los comandos DDL trabajan sobre la estructura de la base de datos y no sobre los datos en sí. Además, las operaciones DDL **se ejecutan de manera inmediata y no pueden deshacerse con ROLLBACK**, ya que no son transaccionales en la mayoría de los sistemas de bases de datos.

Principales Comandos DDL

1. **CREATE** → Se usa para crear bases de datos y estructuras como tablas e índices.
2. **ALTER** → Permite modificar la estructura de tablas, índices o bases de datos.
3. **DROP** → Se usa para eliminar bases de datos, tablas e índices.
4. **TRUNCATE** → Elimina todos los registros de una tabla sin modificar su estructura. (NO INCLUIDO EN ESTA PRACTICA)
5. **RENAME** → Cambia el nombre de una tabla o base de datos.

Estos comandos son fundamentales en el diseño y administración de bases de datos, ya que establecen la estructura sobre la cual se almacenarán los datos y definirán cómo se interrelacionan los diferentes elementos del sistema.

COMANDO "CREATE"

Paso 1: Crear una base de datos

```
CREATE DATABASE tienda_online;
```

- Crea una base de datos llamada tienda_online.
- Si ya existe, MySQL mostrará un error.

Si queremos evitar el error si la base ya existe, podemos usar:

```
CREATE DATABASE IF NOT EXISTS tienda_online;
```

Paso 2: Usar la base de datos

```
USE tienda_online;
```

Paso 3: Crear una Tabla Clientes

```
CREATE TABLE clientes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    correo VARCHAR(100) UNIQUE,
    telefono VARCHAR(15),
    direccion TEXT,
    fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Paso 4: Crear una tabla productos

```
CREATE TABLE productos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(150) NOT NULL,
    descripcion TEXT,
    precio DECIMAL(10,2) NOT NULL,
    stock INT DEFAULT 0,
    fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Paso 5: Crear una Tabla Precios

```
CREATE TABLE pedidos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    cliente_id INT,
    fecha_pedido TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    total DECIMAL(10,2) NOT NULL,
    estado ENUM('Pendiente', 'Enviado', 'Entregado', 'Cancelado') DEFAULT
'Pendiente',
    FOREIGN KEY (cliente_id) REFERENCES clientes(id) ON DELETE CASCADE
);
```

Paso 6: Crear una tabla detalle_pedido

```
CREATE TABLE detalle_pedido (
    id INT AUTO_INCREMENT PRIMARY KEY,
    pedido_id INT,
```

```
    producto_id INT,
    cantidad INT NOT NULL,
    precio_unitario DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (pedido_id) REFERENCES pedidos(id) ON DELETE
CASCADE,
    FOREIGN KEY (producto_id) REFERENCES productos(id) ON DELETE
CASCADE
);
```

Paso 7: Ver la estructura Creada

SHOW TABLES;

Paso 8: Ver la estructura de una tabla:

DESCRIBE clientes;

En este ejercicio, hemos diseñado una base de datos denominada **tienda_online**, compuesta por **cuatro tablas interconectadas** que permiten gestionar la información esencial de una tienda en línea. A continuación, se describen las funciones principales de cada una de estas tablas:

1. **Clientes (clientes)**: Almacena los datos personales de los clientes, incluyendo su nombre, correo electrónico, teléfono, dirección y la fecha en que se registraron en la plataforma.
2. **Productos (productos)**: Registra la información de los productos disponibles en la tienda, como su nombre, descripción, precio, cantidad en stock y fecha de creación.
3. **Pedidos (pedidos)**: Administra los pedidos realizados por los clientes, registrando el identificador del cliente que realiza la compra, la fecha del pedido, el monto total y su estado (Pendiente, Enviado, Entregado o Cancelado).
4. **Detalle de Pedido (detalle_pedido)**: Relaciona los productos con los pedidos, indicando la cantidad adquirida y el precio unitario de cada artículo dentro de un pedido.

Esta estructura permite gestionar de manera eficiente los procesos de compra y venta en una tienda en línea, asegurando la correcta organización y relación de los datos. Además, las **claves foráneas** establecidas en las tablas garantizan la integridad referencial de la base de datos, permitiendo acciones como la eliminación en cascada para mantener la coherencia.

COMANDO "ALTER"

Con el crecimiento de la tienda en línea, se han identificado algunos requerimientos que exigen cambios en la estructura de la base de datos:

1. Agregar un campo para almacenar la fecha de nacimiento de los clientes.
2. Modificar la longitud del campo nombre en la tabla productos.
3. Cambiar el nombre de la columna telefono en la tabla clientes por telefono_contacto.
4. Agregar una nueva tabla llamada categorias y enlazarla con la tabla productos.
5. Eliminar la columna fecha_creacion en la tabla productos, ya que no se considera necesaria.

A continuación, aplicamos estos cambios con el comando ALTER.

Paso 1: Agregar una nueva columna (ADD COLUMN)

ALTER TABLE clientes ADD COLUMN fecha_nacimiento DATE;

Explicación: ADD COLUMN agrega una nueva columna fecha_nacimiento de tipo DATE en la tabla clientes.

Paso 2: Modificar la longitud de una columna (MODIFY COLUMN)

Se ha identificado que el campo nombre en la tabla productos es demasiado corto y debe aumentarse de 150 a 200 caracteres:

ALTER TABLE productos MODIFY COLUMN nombre VARCHAR(200) NOT NULL;

Explicación: MODIFY COLUMN cambia la definición de la columna nombre, permitiendo nombres de productos más largos.

Paso 3: Cambiar el nombre de una columna (CHANGE COLUMN)

Se ha decidido cambiar el nombre de la columna teléfono en clientes por telefono_contacto:

ALTER TABLE clientes CHANGE COLUMN telefono telefono_contacto VARCHAR(15);

Explicación: CHANGE COLUMN permite cambiar el nombre de la columna y su tipo de dato si es necesario.

Paso 4: Agregar una nueva tabla categorias y relacionarla con productos

Se ha decidido organizar los productos en categorías. Para ello, primero creamos la tabla:

```
CREATE TABLE categorias (
   id INT AUTO_INCREMENT PRIMARY KEY,
   nombre VARCHAR(100) NOT NULL
);
```

Luego, agregamos una clave foránea en productos para relacionarla con categorias:

```
ALTER TABLE productos ADD COLUMN categoria_id INT;
ALTER TABLE productos ADD CONSTRAINT fk_categoria FOREIGN KEY (categoria_id) REFERENCES categorias(id) ON DELETE SET NULL;
```

Explicación:

- ADD COLUMN agrega la columna categoria_id en productos.
- ADD CONSTRAINT define fk_categoria como clave foránea para enlazar productos con categorias.
- ON DELETE SET NULL significa que si una categoría es eliminada, los productos asociados tendrán NULL en categoria_id en lugar de ser eliminados.

Paso 5: Eliminar una columna innecesaria (DROP COLUMN)

La columna fecha_creacion en productos ya no es relevante, por lo que la eliminamos:

```
ALTER TABLE productos DROP COLUMN fecha_creacion;
```

Explicación:

- DROP COLUMN elimina fecha_creacion de productos.

Verificación de los cambios

Para comprobar la estructura actualizada de las tablas, usamos:

```
DESCRIBE clientes;
DESCRIBE productos;
DESCRIBE categorias;
```

Resumen

Hemos aplicado varias modificaciones a la base de datos tienda_online con ALTER, logrando lo siguiente:

- ✓ **Agregamos** una columna fecha_nacimiento en clientes.
- ✓ **Modificamos** la longitud del campo nombre en productos.
- ✓ **Renombramos** telefono a telefono_contacto en clientes.
- ✓ **Creamos** la tabla categorias y la enlazamos con productos.
- ✓ **Eliminamos** la columna fecha_creacion de productos.

Estos cambios permiten mejorar la organización de los datos y adaptarse a nuevas necesidades.

COMANDO "DROP"

El comando **DROP** en MySQL se usa para eliminar completamente **bases de datos, tablas, columnas e índices**. DROP **elimina toda la estructura** de un objeto de la base de datos. **⚠ Una vez ejecutado DROP, la eliminación es irreversible**.

Escenario: Eliminación de elementos innecesarios en la base de datos tienda_online

Después de una revisión de la base de datos, se han detectado los siguientes cambios necesarios:

- ✓ **Eliminar la tabla categorias** porque la tienda ha decidido gestionar los productos sin clasificaciones.
- ✓ **Eliminar la columna fecha_nacimiento de clientes**, ya que no se usa para ningún proceso.
- ✓ **Eliminar el índice idx_nombre en clientes**, porque ya no se realizan búsquedas por nombre.
- ✓ **Eliminar la base de datos tienda_online**, en caso de querer eliminar todo el sistema.

Paso 1: Eliminar una tabla (*DROP TABLE****)***
Para eliminar la tabla categorias de la base de datos:

DROP TABLE categorias;

Explicación:

- Elimina la tabla categorias y todos sus datos.
- Si la tabla no existe, generará un error.

- Si existen claves foráneas en otras tablas que dependen de categorias, la eliminación fallará a menos que se hayan definido con ON DELETE SET NULL o ON DELETE CASCADE.

Si queremos evitar errores si la tabla no existe, usamos:

DROP TABLE IF EXISTS categorias;

Paso 2: Eliminar una columna (*DROP COLUMN****)***

Para eliminar la columna fecha_nacimiento de clientes:

ALTER TABLE clientes DROP COLUMN fecha_nacimiento;

Explicación:

- ALTER TABLE permite modificar una tabla existente.
- DROP COLUMN elimina la columna fecha_nacimiento.
- **No se puede recuperar una columna eliminada a menos que se tenga un respaldo.**

Paso 3: Eliminar la base de datos completa (*DROP DATABASE****)***

DROP DATABASE tienda_online; ***(OJO NO EJECUAR ESTOS COMANDOS)***

Si queremos asegurarnos de que solo se elimine si existe:

DROP DATABASE IF EXISTS tienda_online; ***(OJO NO EJECUAR ESTOS COMANDOS)***

Resumen

En este ejemplo, hemos aprendido a usar **DROP** en diferentes niveles:

✅ **DROP TABLE** → Eliminamos la tabla categorias.
✅ **DROP COLUMN** → Eliminamos la columna fecha_nacimiento de clientes.
✅ **DROP DATABASE** → Eliminamos completamente la base de datos tienda_online.

El uso de DROP debe hacerse con **precaución**, ya que elimina datos de forma permanente.