Nombre de la práctica	OPERACIONES CRUD			No.	01
Asignatura:	Taller de bases de datos	Carrera:	Ingeniería en sistemas	Duración de la práctica (Hrs)	

Nombre del alumno: Jesús Navarrete Martínez
Grupo: 3501

I. Competencia(s) específica(s):

II. Lugar de realización de la práctica (laboratorio, taller, aula u otro): Aula

III. Material empleado:

- **Equipo de computo**

IV. Desarrollo de la práctica:

Problemario de operaciones CRUD #2

Creacion de la base de datos

```sql
CREATE DATABASE tienda_virtual;

USE tienda_virtual;

CREATE TABLE productos (
    producto_id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    categoria VARCHAR(50),
    precio DECIMAL(10, 2),
    stock INT,
    fecha_creacion DATETIME DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE clientes (
    cliente_id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    correo VARCHAR(100) UNIQUE,
    fecha_registro DATE DEFAULT CURDATE()
);

CREATE TABLE pedidos (
    pedido_id INT AUTO_INCREMENT PRIMARY KEY,
    cliente_id INT,
    fecha_pedido DATETIME DEFAULT CURRENT_TIMESTAMP,
    total DECIMAL(10, 2),
```

```
    FOREIGN KEY (cliente_id) REFERENCES clientes(cliente_id)
    ON UPDATE CASCADE ON DELETE RESTRICT
);

CREATE TABLE detalle_pedidos (
    detalle_id INT AUTO_INCREMENT PRIMARY KEY,
    pedido_id INT,
    producto_id INT,
    cantidad INT,
    precio_unitario DECIMAL(10, 2),
    FOREIGN KEY (pedido_id) REFERENCES pedidos(pedido_id)
    ON UPDATE CASCADE ON DELETE RESTRICT,
    FOREIGN KEY (producto_id) REFERENCES productos(producto_id)
    ON UPDATE CASCADE ON DELETE RESTRICT
);
```

Ejercicios CREATE

1. **Inserta 5 productos diferentes en la tabla `productos`.**

   Instrucción: Los productos deben incluir un nombre, categoría, precio y stock inicial.

   Solución:

```sql
INSERT INTO productos (nombre, categoria, precio, stock)
VALUES
('Laptop HP', 'Electrónica', 15000.00, 10),
('Smartphone Samsung', 'Electrónica', 8000.00, 20),
('Audífonos Sony', 'Accesorios', 1500.00, 50),
('Cámara Canon', 'Fotografía', 12000.00, 5),
('Teclado Mecánico', 'Accesorios', 2000.00, 30);
```

2. **Registra 3 clientes en la tabla `clientes`.**

   Instrucción: Ingresa datos de nombre y correo para cada cliente. Asegúrate de que los correos sean únicos.

   Solución:

```sql
INSERT INTO clientes (nombre, correo)
VALUES
('Juan Pérez', 'juan.perez@example.com'),
('María López', 'maria.lopez@example.com'),
('Carlos Ramírez', 'carlos.ramirez@example.com');
```

3. **Inserta 2 pedidos hechos por diferentes clientes.**

   Instrucción: Cada pedido debe tener al menos 2 productos, especifica la cantidad y el precio unitario de cada uno.

   Solución:

```
-- Pedido 1 (hecho por el cliente con cliente_id = 1):
-- Primero insertamos el pedido en la tabla pedidos:

INSERT INTO pedidos (cliente_id, total) VALUES (1, 23000.00);

-- Luego agregamos los detalles de los productos para este pedido:
INSERT INTO detalle_pedidos (pedido_id, producto_id, cantidad, precio_unitario)
VALUES
(1, 1, 1, 15000.00), -- Laptop HP
(1, 3, 2, 1500.00); -- Audífonos sony

-- Pedido 2 (hecho por el cliente con cliente_id = 2):
-- Insertamos el pedido en la tabla pedidos:

INSERT INTO pedidos (cliente_id, total)
VALUES (2, 17000.00);

-- Luego agregamos los detalles de los productos para este pedido:

INSERT INTO detalle_pedidos (pedido_id, producto_id, cantidad, precio_unitario)
VALUES
(2, 2, 1, 8000.00), -- Smartphone Samsung
(2, 4, 1, 12000.00); -- Cámara Canon
```

Ejercicios READ

2. **Obtén una lista de todos los productos que tienen un stock mayor a 10 unidades.**

   Instrucción: Muestra el `producto_id`, `nombre`, `precio` y `stock`.

   Solución:

```
SELECT producto_id, nombre, precio, stock FROM productos WHERE stock > 10;
```

3. **Encuentra los pedidos realizados por un cliente en particular.**

   Instrucción: Muestra el `nombre` del cliente, `pedido_id`, `fecha_pedido` y el `total`.

   Solución:

```
SELECT c.nombre, p.pedido_id, p.fecha_pedido, p.total
FROM pedidos p
JOIN clientes c ON p.cliente_id = c.cliente_id
WHERE c.nombre = 'Juan Pérez';
```

4. **Muestra el total de ventas por cada producto.**

   Instrucción: Agrupa por `producto_id` y muestra el `nombre` del producto y la cantidad total vendida en todos los pedidos.

   Solución:

```
SELECT p.nombre, SUM(dp.cantidad) AS total_vendido
FROM detalle_pedidos dp
JOIN productos p ON dp.producto_id = p.producto_id
GROUP BY dp.producto_id;
```

Ejercicios UPDATE

1. **Actualiza el precio de todos los productos de una categoria aumentando un 15%.**

   Instrucción: Usa la columna `categoria` para filtrar los productos.

   Solución:

```
UPDATE productos
SET precio = precio * 1.15
WHERE categoria = 'Electrónica';
```

2. **Modifica el correo de uno de los clientes por un nuevo correo electrónico.**

   Instrucción: Asegúrate de que el nuevo correo sea único.

   Solución:

```
UPDATE clientes
SET correo = 'nuevo.correo@example.com'  -- Cambia por el nuevo correo
WHERE cliente_id = 1;  -- Cambia el ID del cliente según sea necesario
```

3. **Corrige el stock de un producto cuyo stock actual es incorrecto.** *Instrucción:* Busca el producto por su `producto_id` y actualiza el campo `stock`.

   Solución:

```sql
UPDATE productos
SET stock = 25
WHERE producto_id = 1;
```

Ejercicos DELETE

1. **Elimina todos los productos de la tabla `productos` que no tienen stock disponible.**

   Instrucción: Debes usar la columna `stock` para identificar productos con stock igual a 0.

   Solución:

   ```sql
   DELETE FROM productos WHERE stock = 0;
   ```

2. **Borra un pedido que fue cancelado por el cliente.**

   Instrucción: Elimina el pedido junto con todos los registros relacionados en la tabla `detalle_pedidos`.

   Solución:

   ```sql
   DELETE FROM detalle_pedidos
   WHERE pedido_id = 1;

   DELETE FROM pedidos
   WHERE pedido_id = 1;
   ```

3. **Elimina un cliente que ha solicitado la eliminación de su cuenta.**

   Instrucción: Asegúrate de borrar primero los registros relacionados en la tabla `pedidos` y luego el cliente de la tabla `clientes`.

   Solución:

   ```sql
   DELETE FROM detalle_pedidos
   WHERE pedido_id    (SELECT pedido_id FROM pedidos WHERE cliente_id = 1);

   DELETE FROM pedidos
   WHERE cliente_id = 1;

   DELETE FROM clientes
   WHERE cliente_id = 1;
   ```

V. Conclusiones:

La implementación de las operaciones CRUD es fundamental para la gestión eficiente de bases de datos en aplicaciones modernas. A lo largo de la práctica, se demostró cómo cada una de estas operaciones permite la interacción con los datos de manera estructurada y organizada. Crear registros, leer información, actualizar datos existentes y eliminar entradas son acciones clave para el ciclo de vida de la información. Además, se evidenció la importancia de validar los datos y asegurar que las operaciones sean realizadas de manera segura y precisa, minimizando el riesgo de errores y asegurando la integridad de la base de datos. En resumen, el dominio de CRUD es esencial para el desarrollo de aplicaciones sólidas y escalables.