Nombre de la práctica	Operaciones CRUD 2			No.	1
Asignatura:	Taller de Base de datos	Carrera:	Ingeniería en Sistemas Computacionales	Duración de la práctica (Hrs)	

NOMBRE DEL ALUMNO: Vanesa Hernández Martínez
GRUPO: 3501

II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):
Actividades en aula de clases y en equipo personal

III. Material empleado:
- Laptop
- Navicat

Creación de las tablas

```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
);
```

```sql
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.

```sql
-- 1. Inserta 5 productos diferentes en la tabla productos.

INSERT INTO productos (nombre, categoria, precio, stock) VALUES
('Laptop Dell', 'Electrónica', 1200.00, 15),
('Smartphone Samsung', 'Electrónica', 800.00, 30),
('Teclado Mecánico', 'Accesorios', 100.00, 50),
('Monitor LG', 'Electrónica', 250.00, 20),
('Mouse Inalámbrico', 'Accesorios', 35.00, 100);
```

producto_id # int	nombre varchar(100)	categoria varchar(50)	precio # decimal(10,2)	stock # int	fecha_creacion datetime
1	Laptop Dell	Electrónica	1200.00	15	2024-10-21 12:17:30
2	Smartphone Samsung	Electrónica	800.00	30	2024-10-21 12:17:30
3	Teclado Mecánico	Accesorios	100.00	50	2024-10-21 12:17:30
4	Monitor LG	Electrónica	250.00	20	2024-10-21 12:17:30
5	Mouse Inalámbrico	Accesorios	35.00	100	2024-10-21 12:17:30

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.

```sql
-- 2. Registra 3 clientes en la tabla clientes.
INSERT INTO clientes (nombre, correo) VALUES
('Carlos García', 'carlos.garcia@example.com'),
('Ana López', 'ana.lopez@example.com'),
('Miguel Torres', 'miguel.torres@example.com');
```

cliente_id # int	nombre ABC varchar(100)	correo ABC varchar(100)	fecha_registro date
1	Carlos García	carlos.garcia@example.co	2024-10-21
2	Ana López	ana.lopez@example.com	2024-10-21
3	Miguel Torres	miguel.torres@example.cc	2024-10-21

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.

```sql
-- 3. Inserta 2 pedidos hechos por diferentes clientes.

-- Insertar el pedido 1
INSERT INTO pedidos (cliente_id, total) VALUES
(1, 1550.00);

-- Insertar los detalles del pedido 1
INSERT INTO detalle_pedidos (pedido_id, producto_id, cantidad, precio_unitario) VALUES
(1, 1, 1, 1200.00),  -- Laptop Dell
(1, 3, 2, 100.00);    -- Teclado Mecánico

-- Insertar el pedido 2
INSERT INTO pedidos (cliente_id, total) VALUES
(2, 1085.00);

-- Insertar los detalles del pedido 2
INSERT INTO detalle_pedidos (pedido_id, producto_id, cantidad, precio_unitario) VALUES
(2, 2, 1, 800.00),   -- Smartphone Samsung
(2, 5, 3, 35.00);    -- Mouse Inalámbrico
```

pedido_id # int	cliente_id # int	fecha_pedido datetime	total # decimal(10,2)
1	1	2024-10-21 22:53:55	1550.00
2	2	2024-10-21 22:55:02	1085.00

detalle_id # int	pedido_id # int	producto_id # int	cantidad # int	precio_unitario # decimal(10,2)
1	1	1	1	1200.00
2	1	3	2	100.00
3	2	2	1	800.00
4	2	5	3	35.00

Ejercicios READ

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

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

```
-- 1. Obtén una lista de todos los productos que tienen un stock mayor a 10 unidades.
SELECT producto_id, nombre, precio, stock FROM productos WHERE stock > 10;
```

producto_id	nombre	precio	stock
1	Laptop Dell	1200.00	15
2	Smartphone Samsung	800.00	30
3	Teclado Mecánico	100.00	50
4	Monitor LG	250.00	20
5	Mouse Inalámbrico	35.00	100

2. Encuentra los pedidos realizados por un cliente en particular.

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

```
-- 2. Encuentra los pedidos realizados por un cliente en particular.
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 = 'Carlos García';
```

nombre	pedido_id	fecha_pedido	total
Carlos García	1	2024-10-21 22:53:55	1550.00

3. 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.

```
-- 3. Muestra el total de ventas por cada producto
SELECT p.producto_id, p.nombre, SUM(dp.cantidad) AS total_vendido
FROM detalle_pedidos dp
JOIN productos p ON dp.producto_id = p.producto_id
GROUP BY p.producto_id, p.nombre;
```

producto_id	nombre	total_vendido
1	Laptop Dell	1
3	Teclado Mecánico	2
2	Smartphone Samsung	1
5	Mouse Inalámbrico	3

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.

```
-- 1. Actualiza el precio de todos los productos de una categoria aumentando un 15%.
UPDATE productos SET precio = precio * 1.15 WHERE categoria = 'Electrónica';
```

producto_id # int	nombre ABC varchar(100)	categoria ABC varchar(50)	precio # decimal(10,2)	stock # int	fecha_creacion datetime
1	Laptop Dell	Electrónica	1380.00	15	2024-10-21 12:17:30
2	Smartphone Samsung	Electrónica	920.00	30	2024-10-21 12:17:30
3	Teclado Mecánico	Accesorios	100.00	50	2024-10-21 12:17:30
4	Monitor LG	Electrónica	287.50	20	2024-10-21 12:17:30
5	Mouse Inalámbrico	Accesorios	35.00	100	2024-10-21 12:17:30

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

Instrucción: Asegúrate de que el nuevo correo sea único.

```
-- 2. Modifica el correo de uno de los clientes por un nuevo correo electrónico.
UPDATE clientes SET correo = 'carlos.nuevo@example.com' WHERE nombre = 'Carlos García';
```

cliente_id # int	nombre ABC varchar(100)	correo ABC varchar(100)	fecha_registro date
1	Carlos García	carlos.nuevo@example.com	2024-10-21
2	Ana López	ana.lopez@example.com	2024-10-21
3	Miguel Torres	miguel.torres@example.com	2024-10-21

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.

```
-- 3. Corrige el stock de un producto cuyo stock actual es incorrecto.
UPDATE productos SET stock = 60 WHERE producto_id = 3;
```

producto_id # int	nombre ABC varchar(100)	categoria ABC varchar(50)	precio # decimal(10,2)	stock # int	fecha_creacion datetime
1	Laptop Dell	Electrónica	1380.00	15	2024-10-21 12:17:30
2	Smartphone Samsung	Electrónica	920.00	30	2024-10-21 12:17:30
3	Teclado Mecánico	Accesorios	100.00	60	2024-10-21 12:17:30
4	Monitor LG	Electrónica	287.50	20	2024-10-21 12:17:30
5	Mouse Inalámbrico	Accesorios	35.00	100	2024-10-21 12:17:30

Ejercicios 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.

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

```
-- 2. Borra un pedido que fue cancelado por el cliente.
-- Eliminar los detalles del pedido
DELETE FROM detalle_pedidos
WHERE pedido_id = 1;

-- Eliminar el pedido
DELETE FROM pedidos
WHERE pedido_id = 1;
```

detalle_id # int	pedido_id # int	producto_id # int	cantidad # int	precio_unitario # decimal(10,2)
3	2	2	1	800.00
4	2	5	3	35.00

pedido_id # int	cliente_id # int	fecha_pedido datetime	total # decimal(10,2)
2	2	2024-10-21 22:55:02	1085.00

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.

```
-- 3. Elimina un cliente que ha solicitado la eliminación de su cuenta.
-- Eliminar los detalles de los pedidos relacionados con el cliente
DELETE FROM detalle_pedidos
WHERE pedido_id IN (SELECT pedido_id FROM pedidos WHERE cliente_id = 2);

-- Eliminar los pedidos del cliente
DELETE FROM pedidos
WHERE cliente_id = 2;

-- Eliminar al cliente
DELETE FROM clientes
WHERE cliente_id = 2;
```

detalle_id # int	pedido_id # int	producto_id # int	cantidad # int	precio_unitario # decimal(10,2)
(N/A)	(N/A)	(N/A)	(N/A)	(N/A)

pedido_id # int	cliente_id # int	fecha_pedido datetime	total # decimal(10,2)
(N/A)	(N/A)	(N/A)	(N/A)

cliente_id # int	nombre ABC varchar(100)	correo ABC varchar(100)	fecha_registro date
1	Carlos García	carlos.nuevo@example.com	2024-10-21
3	Miguel Torres	miguel.torres@example.com	2024-10-21

Conclusión

Las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) son fundamentales para la gestión eficiente de datos en una base de datos.

- **Crear** (INSERT) permite agregar nuevos registros a las tablas, como productos, clientes y pedidos.
- **Leer** (SELECT) extrae información relevante, facilitando la consulta de datos como stock o historial de pedidos.
- **Actualizar** (UPDATE) modifica registros existentes, como precios o correos, manteniendo la información actualizada.
- **Eliminar** (DELETE) borra datos específicos, como productos sin stock o pedidos cancelados, manteniendo la integridad referencial.

Estas operaciones permiten una gestión precisa y dinámica de los datos, asegurando que la base de datos refleje de forma precisa el estado actual del sistema.