# Taller Mysql

Este taller está diseñado para  profundizar en el manejo y optimización de bases de datos MySQL. A través de ejercicios prácticos, se explorarán temas avanzados para reforzar el conocimiento en normalización, joins, consultas complejas, subconsultas, procedimientos almacenados, funciones definidas por el usuario y triggers.

### Requisitos previos:

- Conocimiento básico de SQL y MySQL
- MySQL instalado y configurado en tu máquina

### Objetivos:

Al finalizar este taller, el participante será capaz de:

1. Diseñar bases de datos optimizadas mediante técnicas de normalización.
2. Realizar consultas avanzadas en múltiples tablas.
3. Utilizar subconsultas para consultas complejas.
4. Crear y ejecutar procedimientos almacenados y funciones definidas por el usuario.
5. Implementar triggers para automatizar operaciones en la base de datos.

# Base de datos

```sql
-- Creación de la base de datos
CREATE DATABASE vtaszfs;
USE vtaszfs;

-- Tabla Clientes
CREATE TABLE Clientes (
    id INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(100),
    email VARCHAR(100) UNIQUE
);

-- Tabla UbicacionCliente
CREATE TABLE UbicacionCliente (
    id INT PRIMARY KEY AUTO_INCREMENT,
    cliente_id INT,
    direccion VARCHAR(255),
    ciudad VARCHAR(100),
    estado VARCHAR(50),
    codigo_postal VARCHAR(10),
    pais VARCHAR(50),
    FOREIGN KEY (cliente_id) REFERENCES Clientes(id)
);

-- Tabla Empleados
CREATE TABLE Empleados (
    id INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(100),
```

```
    puesto VARCHAR(50),
    salario DECIMAL(10, 2),
    fecha_contratacion DATE
);

-- Tabla Proveedores
CREATE TABLE Proveedores (
    id INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(100),
    contacto VARCHAR(100),
    telefono VARCHAR(20),
    direccion VARCHAR(255)
);

-- Tabla TiposProductos
CREATE TABLE TiposProductos (
    id INT PRIMARY KEY AUTO_INCREMENT,
    tipo_nombre VARCHAR(100),
    descripcion TEXT
);

-- Tabla Productos
CREATE TABLE Productos (
    id INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(100),
    precio DECIMAL(10, 2),
    proveedor_id INT,
    tipo_id INT,
    FOREIGN KEY (proveedor_id) REFERENCES Proveedores(id),
    FOREIGN KEY (tipo_id) REFERENCES TiposProductos(id)
);

-- Tabla Pedidos
CREATE TABLE Pedidos (
    id INT PRIMARY KEY AUTO_INCREMENT,
    cliente_id INT,
    fecha DATE,
    total DECIMAL(10, 2),
    FOREIGN KEY (cliente_id) REFERENCES Clientes(id)
);

-- Tabla DetallesPedido
CREATE TABLE DetallesPedido (
    id INT PRIMARY KEY AUTO_INCREMENT,
    pedido_id INT,
    producto_id INT,
    cantidad INT,
    precio DECIMAL(10, 2),
    FOREIGN KEY (pedido_id) REFERENCES Pedidos(id),
    FOREIGN KEY (producto_id) REFERENCES Productos(id)
);
```

> REVISAR LA ESTRUCTURA DE LA BASE DE DATOS PARA VALIDAR QUE SE ENCUENTRA DEBIDAMENTE NORMALIZADA.

## Ejercicios por Tema

Cada tema incluye 10 ejercicios que van de dificultad básica a avanzada.

---

## 1. Normalización

1. Crear una tabla `HistorialPedidos` que almacene cambios en los pedidos.
2. Evaluar la tabla `Clientes` para eliminar datos redundantes y normalizar hasta 3NF.
3. Separar la tabla `Empleados` en una tabla de `DatosEmpleados` y otra para `Puestos`.
4. Revisar la relación `Clientes` y `UbicacionCliente` para evitar duplicación de datos.
5. Normalizar `Proveedores` para tener `ContactoProveedores` en otra tabla.
6. Crear una tabla de `Telefonos` para almacenar múltiples números por cliente.
7. Transformar `TiposProductos` en una relación categórica jerárquica.
8. Normalizar `Pedidos` y `DetallesPedido` para evitar inconsistencias de precios.
9. Usar una relación de muchos a muchos para `Empleados` y `Proveedores`.
10. Convertir la tabla `UbicacionCliente` en una relación genérica de `Ubicaciones`.

---

## 2. Joins

1. Obtener la lista de todos los pedidos con los nombres de clientes usando `INNER JOIN`.
2. Listar los productos y proveedores que los suministran con `INNER JOIN`.
3. Mostrar los pedidos y las ubicaciones de los clientes con `LEFT JOIN`.
4. Consultar los empleados que han registrado pedidos, incluyendo empleados sin pedidos (`LEFT JOIN`).
5. Obtener el tipo de producto y los productos asociados con `INNER JOIN`.
6. Listar todos los clientes y el número de pedidos realizados con `COUNT` y `GROUP BY`.
7. Combinar `Pedidos` y `Empleados` para mostrar qué empleados gestionaron pedidos específicos.
8. Mostrar productos que no han sido pedidos (`RIGHT JOIN`).
9. Mostrar el total de pedidos y ubicación de clientes usando múltiples `JOIN`.
10. Unir `Proveedores`, `Productos`, y `TiposProductos` para un listado completo de inventario.

---

## 3. Consultas Simples

1. Seleccionar todos los productos con precio mayor a $50.
2. Consultar clientes registrados en una ciudad específica.
3. Mostrar empleados contratados en los últimos 2 años.
4. Seleccionar proveedores que suministran más de 5 productos.
5. Listar clientes que no tienen dirección registrada en `UbicacionCliente`.
6. Calcular el total de ventas por cada cliente.

7. Mostrar el salario promedio de los empleados.
8. Consultar el tipo de productos disponibles en `TiposProductos`.
9. Seleccionar los 3 productos más caros.
10. Consultar el cliente con el mayor número de pedidos.

---

## 4. Consultas Multitabla

1. Listar todos los pedidos y el cliente asociado.
2. Mostrar la ubicación de cada cliente en sus pedidos.
3. Listar productos junto con el proveedor y tipo de producto.
4. Consultar todos los empleados que gestionan pedidos de clientes en una ciudad específica.
5. Consultar los 5 productos más vendidos.
6. Obtener la cantidad total de pedidos por cliente y ciudad.
7. Listar clientes y proveedores en la misma ciudad.
8. Mostrar el total de ventas agrupado por tipo de producto.
9. Listar empleados que gestionan pedidos de productos de un proveedor específico.
10. Obtener el ingreso total de cada proveedor a partir de los productos vendidos.

---

## 5. Subconsultas

1. Consultar el producto más caro en cada categoría.
2. Encontrar el cliente con mayor total en pedidos.
3. Listar empleados que ganan más que el salario promedio.
4. Consultar productos que han sido pedidos más de 5 veces.
5. Listar pedidos cuyo total es mayor al promedio de todos los pedidos.
6. Seleccionar los 3 proveedores con más productos.
7. Consultar productos con precio superior al promedio en su tipo.
8. Mostrar clientes que han realizado más pedidos que la media.
9. Encontrar productos cuyo precio es mayor que el promedio de todos los productos.
10. Mostrar empleados cuyo salario es menor al promedio del departamento.

---

## 6. Procedimientos Almacenados

1. Crear un procedimiento para actualizar el precio de todos los productos de un proveedor.
2. Un procedimiento que devuelva la dirección de un cliente por ID.
3. Crear un procedimiento que registre un pedido nuevo y sus detalles.
4. Un procedimiento para calcular el total de ventas de un cliente.
5. Crear un procedimiento para obtener los empleados por puesto.
6. Un procedimiento que actualice el salario de empleados por puesto.
7. Crear un procedimiento que liste los pedidos entre dos fechas.

8. Un procedimiento para aplicar un descuento a productos de una categoría.
9. Crear un procedimiento que liste todos los proveedores de un tipo de producto.
10. Un procedimiento que devuelva el pedido de mayor valor.

---

## 7. Funciones Definidas por el Usuario

1. Crear una función que reciba una fecha y devuelva los días transcurridos.
2. Crear una función para calcular el total con impuesto de un monto.
3. Una función que devuelva el total de pedidos de un cliente específico.
4. Crear una función para aplicar un descuento a un producto.
5. Una función que indique si un cliente tiene dirección registrada.
6. Crear una función que devuelva el salario anual de un empleado.
7. Una función para calcular el total de ventas de un tipo de producto.
8. Crear una función para devolver el nombre de un cliente por ID.
9. Una función que reciba el ID de un pedido y devuelva su total.
10. Crear una función que indique si un producto está en inventario.

---

## 8. Triggers

1. Crear un trigger que registre en `HistorialSalarios` cada cambio de salario de empleados.
2. Crear un trigger que evite borrar productos con pedidos activos.
3. Un trigger que registre en `HistorialPedidos` cada actualización en `Pedidos`.
4. Crear un trigger que actualice el inventario al registrar un pedido.
5. Un trigger que evite actualizaciones de precio a menos de $1.
6. Crear un trigger que registre la fecha de creación de un pedido en `HistorialPedidos`.
7. Un trigger que mantenga el precio total de cada pedido en `Pedidos`.
8. Crear un trigger para validar que `UbicacionCliente` no esté vacío al crear un cliente.
9. Un trigger que registre en `LogActividades` cada modificación en `Proveedores`.
10. Crear un trigger que registre en `HistorialContratos` cada cambio en `Empleados`.

## Ejercicios Combinados de Funciones y Consultas

### Función de Descuento por Categoría de Producto

- **Objetivo:** Crear una función que aplique un descuento sobre el precio de un producto si pertenece a una categoría específica.
- Pasos:
    1. Crear una función `CalcularDescuento` que reciba el `tipo_id` del producto y el `precio` original, y aplique un descuento del 10% si el tipo es "Electrónica".
    2. Realizar una consulta para mostrar el nombre del producto, el precio original y el precio con descuento.

## Función para Obtener la Edad de un Cliente y Filtrar Clientes Mayores de Edad

- **Objetivo:** Crear una función que calcule la edad de un cliente en función de su fecha de nacimiento y luego usarla para listar solo los clientes mayores de 18 años.
- Pasos:
    1. Crear la función `CalcularEdad` que reciba la fecha de nacimiento y calcule la edad.
    2. Consultar todos los clientes y mostrar solo aquellos que sean mayores de 18 años.

## Función de Cálculo de Impuesto y Consulta de Productos con Precio Final

- **Objetivo:** Crear una función que calcule el precio final de un producto aplicando un impuesto del 15% y luego mostrar una lista de productos con el precio final incluido.
- Pasos:
    1. Crear la función `CalcularImpuesto` que reciba el precio del producto y aplique el impuesto.
    2. Mostrar el nombre del producto, el precio original y el precio final con impuesto.

## Función para Calcular el Total de Pedidos de un Cliente

- **Objetivo:** Crear una función que calcule el total de los pedidos de un cliente y usarla para mostrar los clientes con total de pedidos mayor a $1000.
- Pasos:
    1. Crear la función `TotalPedidosCliente` que reciba el ID de un cliente y calcule el total de todos sus pedidos.
    2. Realizar una consulta que muestre el nombre del cliente y su total de pedidos, y filtrar clientes con un total mayor a $1000.

## Función para Calcular el Salario Anual de un Empleado

- **Objetivo:** Crear una función que calcule el salario anual de un empleado y usarla para listar todos los empleados con un salario anual mayor a $50,000.
- Pasos:
    1. Crear la función `SalarioAnual` que reciba el salario mensual y lo multiplique por 12.
    2. Realizar una consulta que muestre el nombre del empleado y su salario anual, filtrando empleados con salario mayor a $50,000.

## Función de Bonificación y Consulta de Salarios Ajustados

- **Objetivo:** Crear una función que calcule la bonificación de un empleado (10% de su salario) y mostrar el salario ajustado de cada empleado.
- Pasos:
    1. Crear una función `Bonificacion` que reciba el salario y calcule el 10%.
    2. Realizar una consulta que muestre el salario ajustado (salario + bonificación).

## Función para Calcular Días Transcurridos Desde el Último Pedido

- **Objetivo:** Crear una función que calcule los días desde el último pedido de un cliente y mostrar clientes que hayan hecho un pedido en los últimos 30 días.
- Pasos:
    1. Crear la función `DiasDesdeUltimoPedido` que reciba el ID de un cliente y calcule los días desde su último pedido.
    2. Realizar una consulta que muestre solo a los clientes con pedidos en los últimos 30 días.

## Función para Calcular el Total en Inventario de un Producto

- **Objetivo:** Crear una función que calcule el total en inventario (cantidad x precio) de cada producto y listar productos con inventario superior a $500.
- Pasos:
    1. Crear la función `TotalInventarioProducto` que multiplique cantidad y precio de un producto.
    2. Realizar una consulta que muestre el nombre del producto y su total en inventario, filtrando los productos con inventario superior a $500.

## Creación de un Historial de Precios de Productos

- **Descripción:** Crear un trigger y una tabla para mantener un historial de precios de productos. Cada vez que el precio de un producto cambia, el trigger debe guardar el ID del producto, el precio antiguo, el nuevo precio y la fecha de cambio.
- Pasos:
    1. Crear la tabla `HistorialPrecios`.
    2. Crear el trigger `RegistroCambioPrecio` en la tabla `Productos` para registrar los cambios de precio.

## Procedimiento para Generar Reporte de Ventas Mensuales por Empleado

- **Descripción:** Crear un procedimiento almacenado que genere un reporte de ventas mensual para cada empleado. El procedimiento debe recibir como parámetros el mes y el año, y devolver una lista de empleados con el total de ventas que gestionaron en ese periodo.
- Pasos:
    1. Crear el procedimiento `ReporteVentasMensuales`.
    2. Usar una subconsulta que agrupe las ventas por empleado y que filtre por el mes y el año.

## Subconsulta para Obtener el Producto Más Vendido por Cada Proveedor

- **Descripción:** Realizar una consulta compleja que devuelva el producto más vendido para cada proveedor, mostrando el nombre del proveedor, el nombre del producto y la cantidad vendida.
- Pasos:
    1. Utilizar una subconsulta para calcular la cantidad total de ventas por producto.
    2. Filtrar en la consulta principal para obtener el producto más vendido de cada proveedor.

### Función para Calcular el Estado de Stock de un Producto

- **Descripción:** Crear una función que calcule el estado de stock de un producto y lo clasifique en “Alto”, “Medio” o “Bajo” en función de su cantidad. Usar esta función en una consulta para listar todos los productos y su estado de stock.
- Pasos:
    1. Crear la función `EstadoStock` que reciba la cantidad de un producto.
    2. En la consulta principal, utilizar la función para clasificar el estado de stock de cada producto.

### Trigger para Control de Inventario en Pedidos

- **Descripción:** Crear un trigger que, al insertar un nuevo pedido, disminuya automáticamente la cantidad en stock del producto. El trigger debe también prevenir que se inserte el pedido si el stock es insuficiente.
- Pasos:
    1. Crear el trigger `ActualizarInventario` en la tabla `DetallesPedido`.
    2. Controlar que no se permita la inserción si la cantidad es mayor que el stock disponible.

### Procedimiento para Generar Informe de Clientes Inactivos

- **Descripción:** Crear un procedimiento almacenado que genere un informe de clientes inactivos (aquellos que no han realizado pedidos en los últimos 6 meses).
- Pasos:
    1. Crear el procedimiento `ClientesInactivos`.
    2. Filtrar clientes que no tengan pedidos recientes usando una subconsulta.

# Entregable

1. Enlace repositorio con la solución de todas las consultas: El readme debe tener la descripción de cada una de la consulta con su solución y prueba.
2. El repositorio debe contener los comandos DML(Inserción de datos) Y DDL (Creación de la base de datos y tablas).

# Causales de nulidad

En caso de encontrarse igualdad en las soluciones de dos o mas proyectos se procederá a la **ANULACION** de todos los proyectos implicados.