



Universidad Tecnológica De Panamá
Facultad de Ingeniería en Sistemas Computacionales
2do Semestre

Profesor: Ronald Ponce

Integrantes: Jean Meléndez 8-985-955

Kevin Valdés 8-1021-301

Daniel Gonzales 8-1022-1099

Martin Liao 1-757-1706

Ricardo Rose

Asignatura: Base de datos

Laboratorio: #2

Año Lectivo: 2024

Introducción

En el presente laboratorio, desarrollado para la asignatura de Bases de Datos, se trabajó en la creación y gestión de un modelo relacional para la empresa ficticia GlobalComerce S.A., la cual se dedica a la venta y distribución de productos electrónicos. El objetivo principal fue implementar una base de datos que permita la gestión eficiente de los procesos internos de la empresa, incluyendo ventas, inventario, producción y recursos humanos.

Se diseñaron tablas clave como CLIENTES, PRODUCTOS, ORDENES, DETALLE_ORDEN, EMPLEADOS e INVENTARIO, aplicando principios de normalización y garantizando la integridad de los datos mediante el uso de llaves primarias y foráneas, restricciones de unicidad, y control de valores nulos. Adicionalmente, se implementaron consultas que cubren las necesidades de los diferentes departamentos de la empresa, permitiendo generar reportes de ventas, gestionar el inventario, y realizar análisis de recursos humanos, entre otros.

Este laboratorio no solo representa una práctica en la aplicación de SQL, sino que también refleja el uso de buenas prácticas en el modelado de bases de datos relacionales.

Código de creación de tablas e inserción realizado en la misma consulta en la base de datos Laboratorio 02:

```
USE Laboratorio_02;  
GO
```

```
-- Creación de la tabla DEPARTAMENTOS_GC (Previa a EMPLEADOS_GC)
```

```
CREATE TABLE DEPARTAMENTOS_GC  
(  
    ID_Departamento INT PRIMARY KEY NOT NULL,  
    Nombre_Departamento VARCHAR (50) NOT NULL  
);  
GO
```

```
-- Creación de la tabla CLIENTES_GC
```

```
CREATE TABLE CLIENTES_GC  
(  
    ID_Cliente INT PRIMARY KEY NOT NULL,  
    Nombre VARCHAR (50) NOT NULL,  
    Apellido VARCHAR (50) NOT NULL,  
    email VARCHAR (50) NOT NULL, -- Reducido el tamaño del campo  
    telefono VARCHAR (20) NOT NULL, -- Reducido el tamaño del campo  
    direccion VARCHAR (100) NOT NULL,  
    FechaRegistro DATE NOT NULL,  
  
    CONSTRAINT UQ_email UNIQUE (email)  
);  
GO
```

```
-- Creación de la tabla PRODUCTOS_GC
```

```
CREATE TABLE PRODUCTOS_GC  
(  
    ID_Producto INT PRIMARY KEY NOT NULL,  
    Nombre_Producto VARCHAR (100) NOT NULL,  
    Categoria VARCHAR (50) NOT NULL,  
    Precio DECIMAL (10,2) NOT NULL CHECK (Precio > 0),  
    stock INT NOT NULL CHECK (stock >= 0),  
    stock_minimo INT NOT NULL CHECK (stock_minimo >= 0),  
    fecha_ingreso DATE NOT NULL  
);  
GO
```

```
-- Creación de la tabla EMPLEADOS_GC (Relacionado con DEPARTAMENTOS_GC)
```

```
CREATE TABLE EMPLEADOS_GC  
(  
    ID_Empleado INT PRIMARY KEY NOT NULL,  
    Nombre_empleado VARCHAR (50) NOT NULL,  
    Apellido_empleado VARCHAR (50) NOT NULL,
```

```

ID_Departamento INT NOT NULL,
fecha_contratacion DATE NOT NULL,
salario DECIMAL (10,2) NOT NULL CHECK (salario >= 0),
fecha_creacion DATETIME DEFAULT GETDATE(), -- Campo para auditoría
ultima_modificacion DATETIME DEFAULT GETDATE(), -- Campo para auditoría

CONSTRAINT FK_Departamento FOREIGN KEY (ID_Departamento) REFERENCES
DEPARTAMENTOS_GC(ID_Departamento)
ON DELETE CASCADE
);
GO

-- Creación de la tabla ORDENES_GC (Relacionado con CLIENTES_GC)
CREATE TABLE ORDENES_GC
(
    ID_Orden INT PRIMARY KEY NOT NULL,
    ID_Cliente INT NOT NULL,
    Fecha_orden DATE NOT NULL,
    total DECIMAL (10,2) NOT NULL CHECK (total >= 0),
    estado VARCHAR (20) NOT NULL CHECK (estado IN ('pendiente', 'completada')),
    fecha_creacion DATETIME DEFAULT GETDATE(), -- Campo para auditoría
    ultima_modificacion DATETIME DEFAULT GETDATE(), -- Campo para auditoría

    CONSTRAINT FK_Cliente FOREIGN KEY (ID_Cliente) REFERENCES CLIENTES_GC(ID_Cliente)
    ON DELETE CASCADE
);
GO

-- Creación de la tabla DETALLES_ORDENES (Relacionado con ORDENES_GC y PRODUCTOS_GC)
CREATE TABLE DETALLES_ORDENES
(
    ID_Detalle INT PRIMARY KEY NOT NULL,
    ID_Orden INT NOT NULL,
    ID_Producto INT NOT NULL,
    cantidad INT NOT NULL CHECK (cantidad > 0),
    PrecioUnitario DECIMAL (10,2) NOT NULL CHECK (PrecioUnitario > 0),
    Subtotal AS (cantidad * PrecioUnitario) PERSISTED, -- Nuevo campo calculado

    CONSTRAINT FK_Orden FOREIGN KEY (ID_Orden) REFERENCES ORDENES_GC(ID_Orden) ON
DELETE CASCADE,
    CONSTRAINT FK_Producto FOREIGN KEY (ID_Producto) REFERENCES PRODUCTOS_GC(ID_Producto)
    ON DELETE CASCADE
);
GO

-- Creación de la tabla VENTAS_GC (Relacionado con CLIENTES_GC y ORDENES_GC)
CREATE TABLE VENTAS_GC
(
    ID_Venta INT PRIMARY KEY NOT NULL,
    ID_Cliente INT NOT NULL,
    ID_Orden INT NOT NULL,
    Fecha_Venta DATE NOT NULL,

```

```

Total_Venta DECIMAL (10,2) NOT NULL CHECK (Total_Venta >= 0),

CONSTRAINT FK_Cliente_Venta FOREIGN KEY (ID_Cliente) REFERENCES CLIENTES_GC(ID_Cliente),
CONSTRAINT FK_Orden_Venta FOREIGN KEY (ID_Orden) REFERENCES ORDENES_GC(ID_Orden)
);
GO

-- Creación de la tabla MOVIMIENTOS_INVENTARIO_GC (Relacionado con PRODUCTOS_GC y EMPLEADOS_GC)
CREATE TABLE MOVIMIENTOS_INVENTARIO_GC
(
    ID_Movimiento INT PRIMARY KEY NOT NULL,
    ID_Producto INT NOT NULL,
    Tipo_Movimiento VARCHAR(20) NOT NULL CHECK (Tipo_Movimiento IN ('entrada', 'salida')),
    Cantidad_Movida INT NOT NULL CHECK (Cantidad_Movida > 0),
    Fecha_Movimiento DATE NOT NULL,
    ID_Empleado INT NOT NULL, -- Nuevo campo para rastrear el empleado responsable

    CONSTRAINT FK_Producto_Movimiento FOREIGN KEY (ID_Producto) REFERENCES
PRODUCTOS_GC(ID_Producto) ON DELETE CASCADE,
    CONSTRAINT FK_Empleado_Movimiento FOREIGN KEY (ID_Empleado) REFERENCES
EMPLEADOS_GC(ID_Empleado)
);
GO

-- Índices adicionales para optimizar las consultas

-- Índice en la tabla ORDENES_GC para consultas por cliente
CREATE INDEX IDX_Cliente_Orden ON ORDENES_GC(ID_Cliente);
GO

-- Índice en la tabla ORDENES_GC para consultas por estado
CREATE INDEX IDX_Estado_Orden ON ORDENES_GC(estado);
GO

-- Índice en la tabla VENTAS_GC para optimizar búsquedas por fecha
CREATE INDEX IDX_Fecha_Venta ON VENTAS_GC(Fecha_Venta);
GO

-- Índice en la tabla MOVIMIENTOS_INVENTARIO_GC para optimizar búsquedas por fecha de movimiento
CREATE INDEX IDX_Fecha_Movimiento ON MOVIMIENTOS_INVENTARIO_GC(Fecha_Movimiento);
GO

```

--BLOQUE DE INSERCIÓN DE DATOS

--INSERTS DEL LABORATORIO 2

USE Laboratorio_02;

GO

-- Insertar datos en la tabla DEPARTAMENTOS_GC (Se agregarán algunos departamentos básicos)

INSERT INTO DEPARTAMENTOS_GC (ID_Departamento, Nombre_Departamento)

VALUES

(1, 'Ventas'),

(2, 'Recursos Humanos'),

(3, 'IT'),

(4, 'Logística'),

(5, 'Producción');

GO

-- Insertar datos en la tabla CLIENTES_GC (15 registros)

INSERT INTO CLIENTES_GC (ID_Cliente, Nombre, Apellido, email, telefono, direccion, FechaRegistro)

VALUES

(1, 'Juan', 'Perez', 'juan.perez@mail.com', '123456789', 'Av. Siempre Viva 123', '2023-01-15'),

(2, 'Maria', 'Garcia', 'maria.garcia@mail.com', '987654321', 'Calle Falsa 456', '2023-02-10'),

(3, 'Carlos', 'Lopez', 'carlos.lopez@mail.com', '555678902', 'Plaza Central 789', '2023-03-05'),

(4, 'Ana', 'Rodriguez', 'ana.rodriguez@mail.com', '444567890', 'Paseo de la Reforma 100', '2023-03-12'),

(5, 'Luis', 'Martinez', 'luis.martinez@mail.com', '333789123', 'Boulevard del Sol 99', '2023-04-20'),

(6, 'Jorge', 'Hernandez', 'jorge.hernandez@mail.com', '222345678', 'Calle Primavera 234', '2023-05-15'),

(7, 'Laura', 'Diaz', 'laura.diaz@mail.com', '999888777', 'Av. Los Pinos 567', '2023-06-01'),

(8, 'Gabriel', 'Sanchez', 'gabriel.sanchez@mail.com', '888777666', 'Calle Álamo 432', '2023-06-10'),

(9, 'Claudia', 'Vega', 'claudia.vega@mail.com', '777666555', 'Paseo del Valle 101', '2023-07-05'),

(10, 'Sofia', 'Ortiz', 'sofia.ortiz@mail.com', '666555444', 'Calle Las Rosas 456', '2023-07-18'),

(11, 'Ricardo', 'Mendoza', 'ricardo.mendoza@mail.com', '555444333', 'Av. Independencia 789', '2023-08-10'),
(12, 'Isabel', 'Ruiz', 'isabel.ruiz@mail.com', '444333222', 'Plaza Mayor 678', '2023-08-22'),
(13, 'Fernando', 'Silva', 'fernando.silva@mail.com', '333222111', 'Av. América 123', '2023-09-01'),
(14, 'Carolina', 'Guzman', 'carolina.guzman@mail.com', '222111000', 'Calle del Sol 200', '2023-09-10'),
(15, 'Pedro', 'Martinez', 'pedro.martinez@mail.com', '777123456', 'Av. Libertad 300', '2023-09-21');
GO

-- Insertar datos en la tabla PRODUCTOS_GC (20 registros)

INSERT INTO PRODUCTOS_GC (ID_Producto, Nombre_Producto, Categoria, Precio, stock, stock_minimo, fecha_ingreso)

VALUES

(1, 'Laptop Lenovo', 'Electrónica', 1200.00, 50, 10, '2023-01-10'),
(2, 'Mouse Inalámbrico', 'Electrónica', 25.50, 200, 30, '2023-01-12'),
(3, 'Teclado Mecánico', 'Electrónica', 75.00, 150, 20, '2023-02-01'),
(4, 'Monitor Samsung', 'Electrónica', 300.00, 75, 10, '2023-03-01'),
(5, 'Impresora HP', 'Electrónica', 150.00, 40, 5, '2023-03-15'),
(6, 'Disco Duro Externo', 'Electrónica', 100.00, 80, 10, '2023-04-10'),
(7, 'Memoria USB', 'Electrónica', 15.00, 500, 50, '2023-04-20'),
(8, 'Smartphone Samsung', 'Electrónica', 950.00, 120, 20, '2023-05-05'),
(9, 'Tablet Apple', 'Electrónica', 650.00, 60, 10, '2023-05-18'),
(10, 'Auriculares Sony', 'Electrónica', 45.00, 300, 30, '2023-06-02'),
(11, 'Cargador Portátil', 'Electrónica', 35.00, 150, 20, '2023-06-15'),
(12, 'Proyector Epson', 'Electrónica', 500.00, 25, 5, '2023-07-10'),
(13, 'Cámara Canon', 'Fotografía', 850.00, 40, 5, '2023-07-25'),
(14, 'Lente Fotográfico', 'Fotografía', 600.00, 30, 5, '2023-08-05'),
(15, 'Micrófono Blue Yeti', 'Audio', 120.00, 100, 10, '2023-08-20'),
(16, 'Tarjeta de Sonido', 'Audio', 220.00, 90, 10, '2023-09-01'),
(17, 'Cámara de Seguridad', 'Seguridad', 250.00, 30, 5, '2023-09-30'),
(18, 'Router TP-Link', 'Redes', 80.00, 100, 15, '2023-09-15'),
(19, 'Switch de Red', 'Redes', 120.00, 60, 10, '2023-10-01'),
(20, 'Modem Cisco', 'Redes', 200.00, 40, 10, '2023-10-05');

GO

-- Insertar datos en la tabla EMPLEADOS_GC (10 registros)

```
INSERT INTO EMPLEADOS_GC (ID_Empleado, Nombre_empleado, Apellido_empleado,  
ID_Departamento, fecha_contratacion, salario)
```

VALUES

```
(1, 'Laura', 'Fernandez', 1, '2021-05-10', 1800.00),  
(2, 'Miguel', 'Ramos', 2, '2020-06-15', 1500.00),  
(3, 'Ana', 'Gonzalez', 3, '2019-04-22', 2500.00),  
(4, 'Ricardo', 'Lopez', 1, '2021-02-14', 1700.00),  
(5, 'Julia', 'Martinez', 4, '2018-11-12', 2000.00),  
(6, 'Manuel', 'Castro', 2, '2020-08-01', 1550.00),  
(7, 'Carmen', 'Jimenez', 5, '2021-09-25', 1900.00),  
(8, 'Roberto', 'Vargas', 3, '2017-12-07', 2400.00),  
(9, 'Sofia', 'Perez', 4, '2019-05-22', 2100.00),  
(10, 'Jose', 'Santos', 5, '2022-09-11', 1700.00);
```

GO

-- Insertar datos en la tabla ORDENES_GC (25 registros)

```
INSERT INTO ORDENES_GC (ID_Orden, ID_Cliente, Fecha_orden, total, estado)
```

VALUES

```
(1, 1, '2023-04-15', 1350.00, 'completada'),  
(2, 2, '2023-05-22', 850.00, 'pendiente'),  
(3, 3, '2023-06-30', 2300.00, 'completada'),  
(4, 4, '2023-05-10', 300.00, 'completada'),  
(5, 5, '2023-06-01', 950.00, 'pendiente'),  
(6, 6, '2023-06-15', 700.00, 'completada'),  
(7, 7, '2023-07-05', 1500.00, 'completada'),  
(8, 8, '2023-07-22', 2500.00, 'pendiente'),  
(9, 9, '2023-08-12', 1800.00, 'completada'),  
(10, 10, '2023-08-28', 1200.00, 'pendiente'),  
(11, 11, '2023-09-15', 950.00, 'completada'),
```



```
(12, 12, '2023-09-25', 800.00, 'pendiente'),  
(13, 13, '2023-10-02', 1700.00, 'completada'),  
(14, 14, '2023-10-08', 2200.00, 'pendiente'),  
(15, 15, '2023-10-12', 1250.00, 'completada'),  
(16, 1, '2023-04-20', 500.00, 'pendiente'),  
(17, 2, '2023-05-30', 150.00, 'completada'),  
(18, 3, '2023-06-05', 3200.00, 'completada'),  
(19, 4, '2023-06-18', 800.00, 'pendiente'),  
(20, 5, '2023-07-10', 650.00, 'completada'),  
(21, 6, '2023-07-20', 2000.00, 'completada'),  
(22, 7, '2023-08-02', 1200.00, 'completada'),  
(23, 8, '2023-08-18', 2400.00, 'pendiente'),  
(24, 9, '2023-09-10', 3000.00, 'completada'),  
(25, 5, '2023-10-01', 950.00, 'pendiente');  
GO
```

-- Insertar datos en la tabla DETALLES_ORDENES (50 registros)

```
INSERT INTO DETALLES_ORDENES (ID_Detalle, ID_Orden, ID_Producto, cantidad, PrecioUnitario)  
VALUES  
(1, 1, 1, 1, 1200.00),  
(2, 2, 2, 2, 25.50),  
(3, 3, 3, 3, 75.00),  
(4, 4, 4, 1, 300.00),  
(5, 5, 5, 1, 150.00),  
(6, 6, 6, 1, 100.00),  
(7, 7, 7, 5, 15.00),  
(8, 8, 8, 2, 950.00),  
(9, 9, 9, 1, 650.00),  
(10, 10, 10, 5, 45.00),  
(11, 11, 11, 1, 35.00),  
(12, 12, 12, 2, 500.00),
```

(13, 13, 13, 1, 850.00),
(14, 14, 14, 1, 600.00),
(15, 15, 15, 3, 120.00),
(16, 16, 16, 2, 220.00),
(17, 17, 17, 2, 250.00),
(18, 18, 18, 3, 80.00),
(19, 19, 19, 4, 120.00),
(20, 20, 20, 2, 200.00),
(21, 21, 1, 1, 1200.00),
(22, 22, 2, 2, 25.50),
(23, 23, 3, 3, 75.00),
(24, 24, 4, 1, 300.00),
(25, 25, 5, 1, 150.00),
(26, 1, 6, 1, 100.00),
(27, 2, 7, 5, 15.00),
(28, 3, 8, 2, 950.00),
(29, 4, 9, 1, 650.00),
(30, 5, 10, 5, 45.00),
(31, 6, 11, 1, 35.00),
(32, 7, 12, 2, 500.00),
(33, 8, 13, 1, 850.00),
(34, 9, 14, 1, 600.00),
(35, 10, 15, 3, 120.00),
(36, 11, 16, 2, 220.00),
(37, 12, 17, 2, 250.00),
(38, 13, 18, 3, 80.00),
(39, 14, 19, 4, 120.00),
(40, 15, 20, 2, 200.00),
(41, 16, 1, 1, 1200.00),
(42, 17, 2, 2, 25.50),
(43, 18, 3, 3, 75.00),

(44, 19, 4, 1, 300.00),
(45, 20, 5, 1, 150.00),
(46, 21, 6, 1, 100.00),
(47, 22, 7, 5, 15.00),
(48, 23, 8, 2, 950.00),
(49, 24, 9, 1, 650.00),
(50, 25, 5, 2, 300.00);
GO

-- Insertar datos en la tabla MOVIMIENTOS_INVENTARIO_GC (30 registros)

INSERT INTO MOVIMIENTOS_INVENTARIO_GC (ID_Movimiento, ID_Producto, Tipo_Movimiento, Cantidad_Movida, Fecha_Movimiento, ID_Empleado)

VALUES

(1, 1, 'entrada', 50, '2023-01-10', 1),
(2, 2, 'salida', 20, '2023-01-15', 2),
(3, 3, 'entrada', 100, '2023-02-01', 3),
(4, 4, 'salida', 15, '2023-02-10', 4),
(5, 5, 'entrada', 40, '2023-03-01', 5),
(6, 6, 'salida', 25, '2023-03-12', 6),
(7, 7, 'entrada', 500, '2023-04-10', 7),
(8, 8, 'salida', 100, '2023-05-01', 8),
(9, 9, 'entrada', 60, '2023-06-01', 9),
(10, 10, 'salida', 75, '2023-07-01', 10),
(11, 1, 'entrada', 25, '2023-08-10', 1),
(12, 2, 'salida', 50, '2023-08-15', 2),
(13, 3, 'entrada', 30, '2023-09-01', 3),
(14, 4, 'salida', 10, '2023-09-10', 4),
(15, 5, 'entrada', 20, '2023-09-20', 5),
(16, 6, 'salida', 60, '2023-09-25', 6),
(17, 7, 'entrada', 150, '2023-09-30', 7),
(18, 8, 'salida', 25, '2023-10-01', 8),
(19, 9, 'entrada', 50, '2023-10-05', 9),

(20, 10, 'salida', 30, '2023-10-10', 10),
(21, 1, 'entrada', 45, '2023-10-15', 1),
(22, 2, 'salida', 40, '2023-10-20', 2),
(23, 3, 'entrada', 70, '2023-10-25', 3),
(24, 4, 'salida', 20, '2023-10-28', 4),
(25, 5, 'entrada', 25, '2023-10-30', 5),
(26, 6, 'salida', 35, '2023-10-31', 6),
(27, 7, 'entrada', 200, '2023-11-01', 7),
(28, 8, 'salida', 10, '2023-11-02', 8),
(29, 9, 'entrada', 40, '2023-11-03', 9),
(30, 10, 'salida', 10, '2023-11-04', 10);
GO

Códigos para ejecutar consultas requeridas del Laboratorio 02:

--Por parte de la Gerencia:

-- 1. Consultar el total de ventas realizadas en el mes

```
SELECT SUM(Total_Venta) AS TotalVentasMes
FROM VENTAS_GC
WHERE MONTH(Fecha_Venta) = MONTH(GETDATE()) AND YEAR(Fecha_Venta) = YEAR(GETDATE());
```

-- 2. Obtener un listado de los 5 productos más vendidos

```
SELECT TOP 5 P.Nombre_Producto, SUM(DO.cantidad) AS CantidadVendida
FROM DETALLES_ORDENES DO
JOIN PRODUCTOS_GC P ON DO.ID_Producto = P.ID_Producto
GROUP BY P.Nombre_Producto
ORDER BY CantidadVendida DESC;
```

-- 3. Calcular el ingreso mensual por ventas para el año actual

```
SELECT MONTH(Fecha_Venta) AS Mes, SUM(Total_Venta) AS IngresoMensual
FROM VENTAS_GC
WHERE YEAR(Fecha_Venta) = YEAR(GETDATE())
GROUP BY MONTH(Fecha_Venta)
ORDER BY Mes;
```

-- 4. Listar las órdenes de compra pendientes

```
SELECT ID_Orden, ID_Cliente, Fecha_orden, total
FROM ORDENES_GC
WHERE estado = 'pendiente';
```

-- Reunión con Ventas:

-- 1. Mostrar el detalle de las ventas realizadas a un cliente específico en el último trimestre

```
DECLARE @ClienteID INT = 1; -- Cambia el valor según el cliente específico
SELECT V.ID_Venta, V.Fecha_Venta, V.Total_Venta, P.Nombre_Producto, DO.cantidad, DO.PrecioUnitario
FROM VENTAS_GC V
JOIN ORDENES_GC O ON V.ID_Orden = O.ID_Orden
JOIN DETALLES_ORDENES DO ON O.ID_Orden = DO.ID_Orden
JOIN PRODUCTOS_GC P ON DO.ID_Producto = P.ID_Producto
WHERE V.ID_Cliente = @ClienteID
      AND V.Fecha_Venta >= DATEADD(QUARTER, -1, GETDATE());
```

-- 2. Consultar las órdenes de compra por cliente, indicando el total de cada orden

```
SELECT O.ID_Cliente, O.ID_Orden, O.Fecha_orden, O.total
FROM ORDENES_GC O
ORDER BY O.ID_Cliente;
```

-- 3. Listar todos los clientes que han realizado compras superiores a un monto determinado

```
DECLARE @MontoMinimo DECIMAL(10, 2) = 1000; -- Cambia el valor del monto mínimo
SELECT C.Nombre, C.Apellido, SUM(V.Total_Venta) AS TotalComprado
FROM CLIENTES_GC C
JOIN VENTAS_GC V ON C.ID_Cliente = V.ID_Cliente
GROUP BY C.Nombre, C.Apellido
HAVING SUM(V.Total_Venta) > @MontoMinimo;
```

-- 4. Obtener el número total de órdenes de compra en el sistema

```
SELECT COUNT(*) AS TotalOrdenes
FROM ORDENES_GC;
```

--Visita a Producción:

-- 1. Consultar el inventario actual de todos los productos

```
SELECT ID_Producto, Nombre_Producto, stock
FROM PRODUCTOS_GC;
```

-- 2. Listar los productos que están por debajo de un nivel de stock mínimo

```
SELECT Nombre_Producto, stock, stock_minimo
FROM PRODUCTOS_GC
WHERE stock < stock_minimo;
```

-- 3. Mostrar los movimientos de inventario por producto en el último mes

```
SELECT MI.ID_Movimiento, P.Nombre_Producto, MI.Tipo_Movimiento, MI.Cantidad_Movida,
MI.Fecha_Movimiento
FROM MOVIMIENTOS_INVENTARIO_GC MI
JOIN PRODUCTOS_GC P ON MI.ID_Producto = P.ID_Producto
WHERE MI.Fecha_Movimiento >= DATEADD(MONTH, -1, GETDATE());
```

-- 4. Calcular el valor total del inventario actual

```
SELECT SUM(P.Precio * P.stock) AS ValorTotalInventario
FROM PRODUCTOS_GC P;
```

--Recursos Humanos:

-- 1. Listar los empleados contratados en el último año

```
SELECT Nombre_empleado, Apellido_empleado, fecha_contratacion, salario
FROM EMPLEADOS_GC
WHERE fecha_contratacion >= DATEADD(YEAR, -1, GETDATE());
```

-- 2. Obtener el salario promedio de los empleados en cada departamento

```
SELECT D.Nombre_Departamento, AVG(E.salario) AS SalarioPromedio
FROM EMPLEADOS_GC E
JOIN DEPARTAMENTOS_GC D ON E.ID_Departamento = D.ID_Departamento
GROUP BY D.Nombre_Departamento;
```

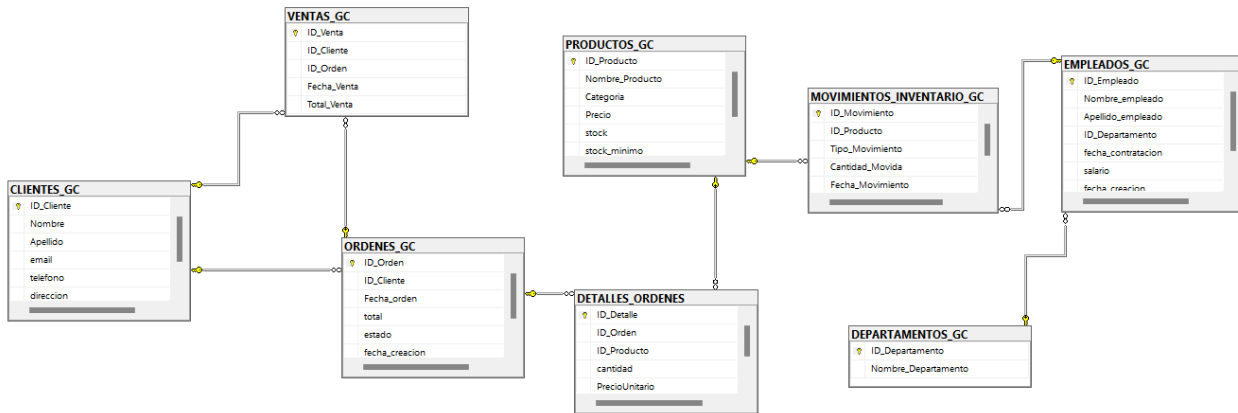
-- 3. Consultar el número total de empleados en cada departamento

```
SELECT D.Nombre_Departamento, COUNT(E.ID_Empleado) AS TotalEmpleados
FROM EMPLEADOS_GC E
JOIN DEPARTAMENTOS_GC D ON E.ID_Departamento = D.ID_Departamento
GROUP BY D.Nombre_Departamento;
```

-- 4. Mostrar los empleados que ganan por encima de un salario específico

```
DECLARE @SalarioMinimo DECIMAL(10, 2) = 2000; -- Cambia el valor según el salario específico
SELECT Nombre_empleado, Apellido_empleado, salario
FROM EMPLEADOS_GC
WHERE salario > @SalarioMinimo;
```


MODELO ER DE ESTE PROYECTO:



Sustentación del Proyecto:

DEPARTAMENTOS_GC

Esta tabla almacena información sobre los departamentos de la empresa. Es necesaria para las consultas de Recursos Humanos, específicamente para calcular el salario promedio de los empleados en cada departamento y para contar el número total de empleados por departamento.

CLIENTES_GC

Almacena los datos de los clientes, lo que es fundamental para las consultas de Ventas, como listar clientes que han realizado compras superiores a un monto determinado, mostrar detalles de ventas a un cliente específico y consultar las órdenes de compra por cliente. Además, la tabla permite realizar un seguimiento de los clientes y su información de contacto, lo que es esencial para las interacciones comerciales.

PRODUCTOS_GC

Contiene información sobre los productos disponibles, incluyendo su precio y stock. Esta tabla es crucial para calcular el ingreso mensual por ventas, listar los 5 productos más vendidos, y realizar consultas sobre el inventario actual y productos por debajo del nivel mínimo de stock. También se relaciona con los movimientos de inventario.

EMPLEADOS_GC

Esta tabla registra información sobre los empleados, incluyendo su salario y el departamento al que pertenecen. Es esencial para las consultas de Recursos Humanos que requieren información sobre empleados contratados recientemente, el salario promedio por departamento y el número total de empleados en cada departamento.

ORDENES_GC

Almacena información sobre las órdenes de compra, incluyendo el estado de la orden y el cliente asociado. Es fundamental para las consultas que listan las órdenes de compra pendientes, calculan el ingreso mensual por ventas y consultan las órdenes de compra por cliente, indicando el total de cada orden.

DETALLES_ORDENES

Esta tabla descompone las órdenes en detalles más específicos, registrando cada producto asociado a una orden y su cantidad. Es necesaria para las consultas que obtienen un listado de los 5 productos más vendidos y que muestran el detalle de ventas realizadas a un cliente en un periodo específico.

VENTAS_GC

Almacena información sobre las ventas realizadas, vinculando las ventas a clientes y órdenes. Es crucial para las consultas de Gerencia que requieren calcular el total de ventas en el mes y el ingreso mensual por ventas.

MOVIMIENTOS_INVENTARIO_GC

Esta tabla registra los movimientos de inventario, permitiendo hacer un seguimiento de las entradas y salidas de productos. Es necesaria para las consultas de Producción que requieren mostrar movimientos de inventario por producto en el último mes y calcular el valor total del inventario actual.

IMÁGENES DE LOS RESULTADOS DE LAS CONSULTAS REQUERIDAS:

Consultas realizadas por el departamento de gerencia:

1.Consultar el total de ventas realizadas en el mes.

```
-- 1. Consultar el total de ventas realizadas en el mes
SELECT SUM(Total_Venta) AS TotalVentasMes
FROM VENTAS_GC
WHERE MONTH(Fecha_Venta) = MONTH(GETDATE()) AND YEAR(Fecha_Venta) = YEAR(GETDATE());

-- 2. Obtener un listado de los 5 productos más vendidos
SELECT TOP 5 P.Nombre_Producto, SUM(DO.cantidad) AS CantidadVendida
FROM DETALLES_ORDENES DO
JOIN PRODUCTOS_GC P ON DO.ID_Producto = P.ID_Producto
GROUP BY P.Nombre_Producto
ORDER BY CantidadVendida DESC;
```

22 %

Results Messages

	TotalVentasMes
1	NULL

Query executed successfully.

2. Obtener un listado de los 5 productos más vendidos.

```
-- 2. Obtener un listado de los 5 productos más vendidos
SELECT TOP 5 P.Nombre_Producto, SUM(DO.cantidad) AS CantidadVendida
FROM DETALLES_ORDENES DO
JOIN PRODUCTOS_GC P ON DO.ID_Producto = P.ID_Producto
GROUP BY P.Nombre_Producto
ORDER BY CantidadVendida DESC;

-- 3. Calcular el ingreso mensual por ventas para el año actual
SELECT MONTH(Fecha_Venta) AS Mes, SUM(Total_Venta) AS IngresoMensual
FROM VENTAS_GC
WHERE YEAR(Fecha_Venta) = YEAR(GETDATE())
GROUP BY MONTH(Fecha_Venta)
```

122 %

Results Messages

	Nombre_Producto	CantidadVendida
1	Memoria USB	15
2	Auriculares Sony	10
3	Teclado Mecánico	9
4	Switch de Red	8
5	Microfono Blue Yeti	6

✓ Query executed successfully.

3. Calcular el ingreso mensual por ventas para el año actual

```
-- 3. Calcular el ingreso mensual por ventas para el año actual
SELECT MONTH(Fecha_Venta) AS Mes, SUM(Total_Venta) AS IngresoMensual
FROM VENTAS_GC
WHERE YEAR(Fecha_Venta) = YEAR(GETDATE())
GROUP BY MONTH(Fecha_Venta)
ORDER BY Mes;

-- 4. Listar las órdenes de compra pendientes
SELECT ID_Orden, ID_Cliente, Fecha_orden, total
FROM ORDENES_GC
WHERE estado = 'pendiente';
```

122 %

Results Messages

Mes	IngresoMensual
-----	----------------

✓ Query executed successfully.

4. Listar las órdenes de compra pendientes

```
-- 4. Listar las órdenes de compra pendientes
SELECT ID_Orden, ID_Cliente, Fecha_orden, total
FROM ORDENES_GC
WHERE estado = 'pendiente';
```

122 %

Results

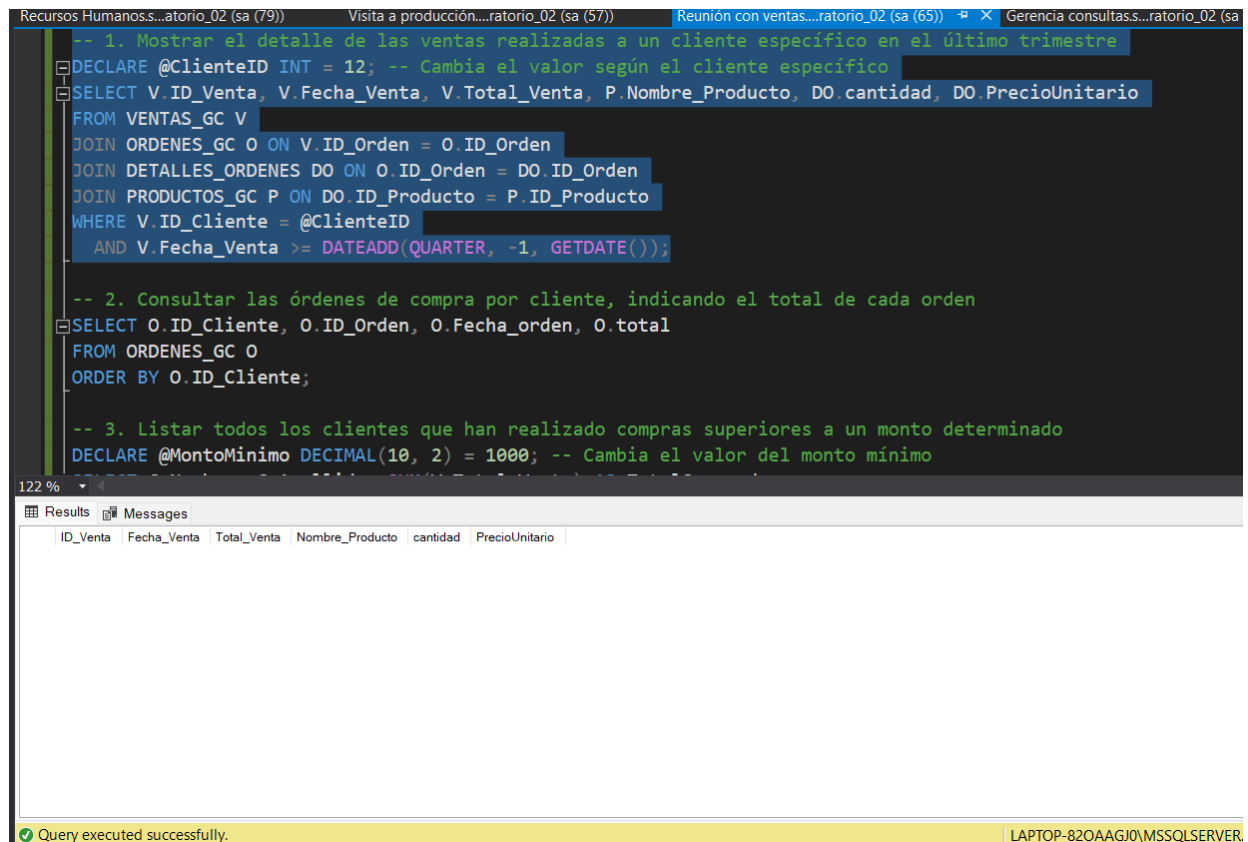
Messages

	ID_Orden	ID_Cliente	Fecha_orden	total
1	2	2	2023-05-22	850.00
2	5	5	2023-06-01	950.00
3	8	8	2023-07-22	2500.00
4	10	10	2023-08-28	1200.00
5	12	12	2023-09-25	800.00
6	14	14	2023-10-08	2200.00
7	16	1	2023-04-20	500.00
8	19	4	2023-06-18	800.00
9	23	8	2023-08-18	2400.00
10	25	5	2023-10-01	950.00

✓ Query executed successfully.

Consultas de la reunión de ventas:

1. Mostrar el detalle de las ventas realizadas a un cliente específico en el último trimestre.



The screenshot displays the SQL Server Enterprise Manager interface. The top pane shows a query with three comments and three SQL statements. The first query is highlighted in blue. The second query is also highlighted in blue. The third query is highlighted in blue. The bottom pane shows the results of the first query, which is a table with six columns: ID_Venta, Fecha_Venta, Total_Venta, Nombre_Producto, cantidad, and PrecioUnitario. The table is currently empty. The status bar at the bottom indicates that the query was executed successfully.

```
-- 1. Mostrar el detalle de las ventas realizadas a un cliente específico en el último trimestre
DECLARE @ClienteID INT = 12; -- Cambia el valor según el cliente específico
SELECT V.ID_Venta, V.Fecha_Venta, V.Total_Venta, P.Nombre_Producto, DO.cantidad, DO.PrecioUnitario
FROM VENTAS_GC V
JOIN ORDENES_GC O ON V.ID_Orden = O.ID_Orden
JOIN DETALLES_ORDENES DO ON O.ID_Orden = DO.ID_Orden
JOIN PRODUCTOS_GC P ON DO.ID_Producto = P.ID_Producto
WHERE V.ID_Cliente = @ClienteID
AND V.Fecha_Venta >= DATEADD(QUARTER, -1, GETDATE());

-- 2. Consultar las órdenes de compra por cliente, indicando el total de cada orden
SELECT O.ID_Cliente, O.ID_Orden, O.Fecha_orden, O.total
FROM ORDENES_GC O
ORDER BY O.ID_Cliente;

-- 3. Listar todos los clientes que han realizado compras superiores a un monto determinado
DECLARE @MontoMinimo DECIMAL(10, 2) = 1000; -- Cambia el valor del monto mínimo
```

ID_Venta	Fecha_Venta	Total_Venta	Nombre_Producto	cantidad	PrecioUnitario
----------	-------------	-------------	-----------------	----------	----------------

Query executed successfully. | LAPTOP-82OAGJ0\MSSQLSERVER

2. Consultar las órdenes de compra por cliente, indicando el total de cada orden

Recursos Humanos.s...atorio_02 (sa (79)) Visita a producción...atorio_02 (sa (57)) Reunión con ventas...atorio_02 (sa (65)) X Gerencia cor

```
-- 2. Consultar las órdenes de compra por cliente, indicando el total de cada orden
SELECT O.ID_Cliente, O.ID_Orden, O.Fecha_orden, O.total
FROM ORDENES_GC O
ORDER BY O.ID_Cliente;

-- 3. Listar todos los clientes que han realizado compras superiores a un monto determinado
DECLARE @MontoMinimo DECIMAL(10, 2) = 1000; -- Cambia el valor del monto mínimo
SELECT C.Nombre, C.Apellido, SUM(V.Total_Venta) AS TotalComprado
FROM CLIENTES_GC C
JOIN VENTAS_GC V ON C.ID_Cliente = V.ID_Cliente
GROUP BY C.Nombre, C.Apellido
HAVING SUM(V.Total_Venta) > @MontoMinimo;

-- 4. Obtener el número total de órdenes de compra en el sistema
SELECT COUNT(*) AS TotalOrdenes
FROM ORDENES_GC;
```

122 %

Results Messages

	ID_Cliente	ID_Orden	Fecha_orden	total
1	1	1	2023-04-15	1350.00
2	1	16	2023-04-20	500.00
3	2	2	2023-05-22	850.00
4	2	17	2023-05-30	150.00
5	3	3	2023-06-30	2300.00
6	3	18	2023-06-05	3200.00
7	4	4	2023-05-10	300.00
8	4	19	2023-06-18	800.00
9	5	5	2023-06-01	950.00
10	5	20	2023-07-10	650.00
11	5	25	2023-10-01	950.00
12	6	6	2023-06-15	700.00
13	6	21	2023-07-20	2000.00
14	7	7	2023-07-05	1500.00

Query executed successfully. LAPTOP-820

3. Listar todos los clientes que han realizado compras superiores a un monto determinado

```
-- 3. Listar todos los clientes que han realizado compras superiores a un monto determinado
DECLARE @MontoMinimo DECIMAL(10, 2) = 1000; -- Cambia el valor del monto mínimo
SELECT C.Nombre, C.Apellido, SUM(V.Total_Venta) AS TotalComprado
FROM CLIENTES_GC C
JOIN VENTAS_GC V ON C.ID_Cliente = V.ID_Cliente
GROUP BY C.Nombre, C.Apellido
HAVING SUM(V.Total_Venta) > @MontoMinimo;

-- 4. Obtener el número total de órdenes de compra en el sistema
SELECT COUNT(*) AS TotalOrdenes
FROM ORDENES_GC;
```

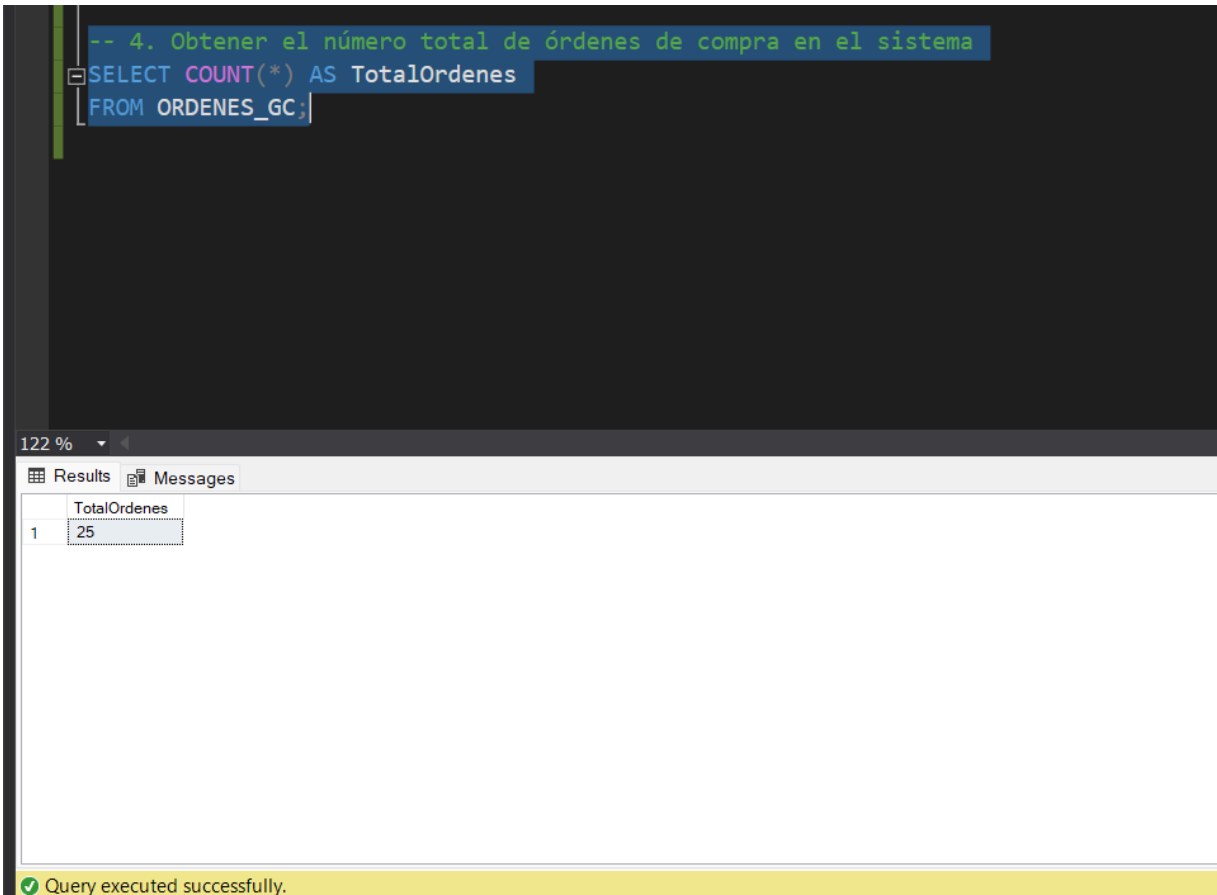
122 %

Results Messages

Nombre	Apellido	TotalComprado
--------	----------	---------------

✓ Query executed successfully. LAPTOP-820AAGJ0\M

4. Obtener el número total de órdenes de compra en el sistema



```
-- 4. Obtener el número total de órdenes de compra en el sistema
SELECT COUNT(*) AS TotalOrdenes
FROM ORDENES_GC;
```

122 %

Results Messages

	TotalOrdenes
1	25

✓ Query executed successfully.

Consultas de la visita al departamento de producción

1.Consultar el inventario actual de todos los productos

Recursos Humanos.s...atorio_02 (sa (79)) Visita a producción....ratorio_02 (sa (57)) Reunión con ven

```
-- 1. Consultar el inventario actual de todos los productos
SELECT ID_Producto, Nombre_Producto, stock
FROM PRODUCTOS_GC;

-- 2. Listar los productos que están por debajo de un nivel de stock
SELECT Nombre_Producto, stock, stock_minimo
FROM PRODUCTOS_GC
WHERE stock < stock_minimo;

-- 3. Mostrar los movimientos de inventario por producto en el último mes
SELECT MI.ID_Movimiento, P.Nombre_Producto, MI.Tipo_Movimiento, MI.Fecha_Movimiento
FROM MOVIMIENTOS_INVENTARIO_GC MI
JOIN PRODUCTOS_GC P ON MI.ID_Producto = P.ID_Producto
WHERE MI.Fecha_Movimiento >= DATEADD(MONTH, -1, GETDATE());
```

122 %

Results Messages

	ID_Producto	Nombre_Producto	stock
1	1	Laptop Lenovo	50
2	2	Mouse Inalámbrico	200
3	3	Teclado Mecánico	150
4	4	Monitor Samsung	75
5	5	Impresora HP	40
6	6	Disco Duro Externo	80
7	7	Memoria USB	500
8	8	Smartphone Samsung	120
9	9	Tablet Apple	60
10	10	Auriculares Sony	300
11	11	Cargador Portátil	150
12	12	Proyector Epson	25
13	13	Cámara Canon	40
14	14	Lente Fotográfico	30
15	15	Micrófono Blue Yeti	100
16	16	Tarjeta de Sonido	90
17	17	Cámara de Seguridad	20

Query executed successfully.

2. Listar los productos que están por debajo de un nivel stock mínimo

```
-- 2. Listar los productos que están por debajo de un nivel de stock mínimo
SELECT Nombre_Producto, stock, stock_minimo
FROM PRODUCTOS_GC
WHERE stock < stock_minimo;

-- 3. Mostrar los movimientos de inventario por producto en el último mes
SELECT MI.ID_Movimiento, P.Nombre_Producto, MI.Tipo_Movimiento, MI.Cantidad_Movida, MI.Fecha_Movimiento
FROM MOVIMIENTOS_INVENTARIO_GC MI
JOIN PRODUCTOS_GC P ON MI.ID_Producto = P.ID_Producto
WHERE MI.Fecha_Movimiento >= DATEADD(MONTH, -1, GETDATE());
```

22 %

Results Messages

Nombre_Producto	stock	stock_minimo
-----------------	-------	--------------

Query executed successfully. LAP

3. Mostrar los movimientos de inventario por producto en el último mes

```
-- 3. Mostrar los movimientos de inventario por producto en el último mes
SELECT MI.ID_Movimiento, P.Nombre_Producto, MI.Tipo_Movimiento, MI.Cantidad_Movida, MI.Fecha_Movimiento
FROM MOVIMIENTOS_INVENTARIO_GC MI
JOIN PRODUCTOS_GC P ON MI.ID_Producto = P.ID_Producto
WHERE MI.Fecha_Movimiento >= DATEADD(MONTH, -1, GETDATE());

-- 4. Calcular el valor total del inventario actual
SELECT SUM(P.Precio * P.stock) AS ValorTotalInventario
FROM PRODUCTOS_GC P;
```

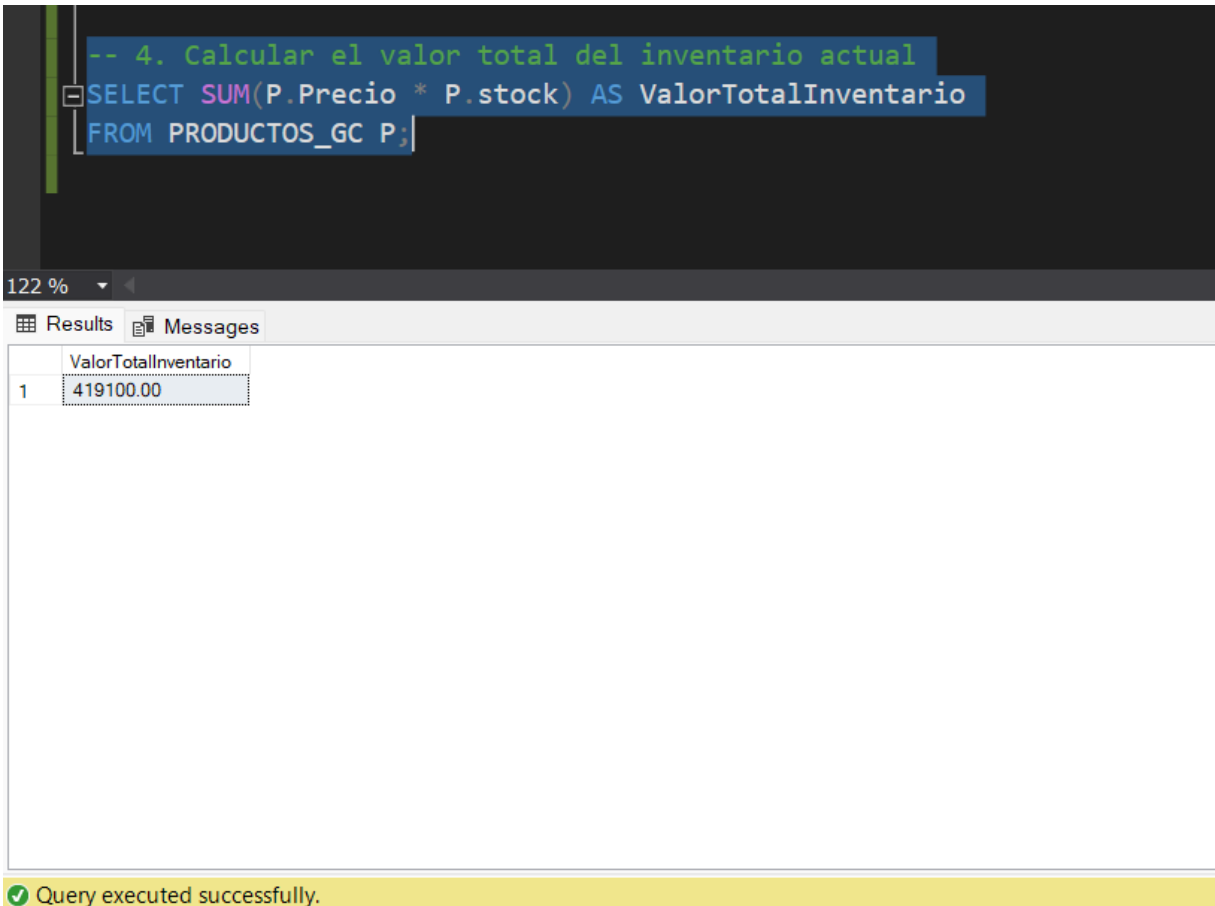
122 %

Results Messages

ID_Movimiento	Nombre_Producto	Tipo_Movimiento	Cantidad_Movida	Fecha_Movimiento
---------------	-----------------	-----------------	-----------------	------------------

Query executed successfully. LAPTOP-820AAGJ0\MSSQLSERVER... sa (57)

4. Calcular el valor total del inventario actual




The screenshot shows a SQL query being executed in a database client. The query is:

```
-- 4. Calcular el valor total del inventario actual
SELECT SUM(P.Precio * P.stock) AS ValorTotalInventario
FROM PRODUCTOS_GC P;
```

The results pane shows a single row with the value 419100.00 for the column ValorTotalInventario.

	ValorTotalInventario
1	419100.00

At the bottom, a status bar indicates:  Query executed successfully.

Consultas del departamento de recursos humanos

1. Listar los empleados contratados en el último año

```
-- 1. Listar los empleados contratados en el último año
SELECT Nombre_empleado, Apellido_empleado, fecha_contratacion, salario
FROM EMPLEADOS_GC
WHERE fecha_contratacion >= DATEADD(YEAR, -1, GETDATE());

-- 2. Obtener el salario promedio de los empleados en cada departamento
SELECT D.Nombre_Departamento, AVG(E.salario) AS SalarioPromedio
FROM EMPLEADOS_GC E
JOIN DEPARTAMENTOS_GC D ON E.ID_Departamento = D.ID_Departamento
GROUP BY D.Nombre_Departamento;

-- 3. Consultar el número total de empleados en cada departamento
SELECT D.Nombre_Departamento, COUNT(E.ID_Empleado) AS TotalEmpleados
FROM EMPLEADOS_GC E
JOIN DEPARTAMENTOS_GC D ON E.ID_Departamento = D.ID_Departamento
```

122 %

Results Messages

Nombre_empleado	Apellido_empleado	fecha_contratacion	salario
-----------------	-------------------	--------------------	---------

Query executed successfully.

2. Obtener el salario promedio de los empleados en cada departamento

```
-- 2. Obtener el salario promedio de los empleados en cada departamento
SELECT D.Nombre_Departamento, AVG(E.salario) AS SalarioPromedio
FROM EMPLEADOS_GC E
JOIN DEPARTAMENTOS_GC D ON E.ID_Departamento = D.ID_Departamento
GROUP BY D.Nombre_Departamento;

-- 3. Consultar el número total de empleados en cada departamento
SELECT D.Nombre_Departamento, COUNT(E.ID_Empleado) AS TotalEmpleados
FROM EMPLEADOS_GC E
JOIN DEPARTAMENTOS_GC D ON E.ID_Departamento = D.ID_Departamento
```

122 %

Results Messages

	Nombre_Departamento	SalarioPromedio
1	IT	2450.000000
2	Logística	2050.000000
3	Producción	1800.000000
4	Recursos Humanos	1525.000000
5	Ventas	1750.000000

Query executed successfully.

3. Consultar el número total de empleados en cada departamento

```
-- 3. Consultar el número total de empleados en cada departamento
SELECT D.Nombre_Departamento, COUNT(E.ID_Empleado) AS TotalEmpleados
FROM EMPLEADOS_GC E
JOIN DEPARTAMENTOS_GC D ON E.ID_Departamento = D.ID_Departamento
GROUP BY D.Nombre_Departamento;

-- 4. Mostrar los empleados que ganan por encima de un salario específico
DECLARE @SalarioMinimo DECIMAL(10, 2) = 2000; -- Cambia el valor según el salario
SELECT Nombre_empleado, Apellido_empleado, salario
FROM EMPLEADOS_GC
```

122 %

Results Messages

	Nombre_Departamento	TotalEmpleados
1	IT	2
2	Logística	2
3	Producción	2
4	Recursos Humanos	2
5	Ventas	2

✓ Query executed successfully.

4. Mostrar los empleados que ganan por encima de un salario específico

```
-- 4. Mostrar los empleados que ganan por encima de un salario específico  
DECLARE @SalarioMinimo DECIMAL(10, 2) = 2000; -- Cambia el valor según el salario específico  
SELECT Nombre_empleado, Apellido_empleado, salario  
FROM EMPLEADOS_GC  
WHERE salario > @SalarioMinimo;
```

122 %

Results Messages

	Nombre_empleado	Apellido_empleado	salario
1	Ana	Gonzalez	2500.00
2	Roberto	Vargas	2400.00
3	Sofia	Perez	2100.00

✓ Query executed successfully.

LAPTOP-820AAGJ0\M

Consideraciones finales y conclusión

A lo largo del desarrollo de este laboratorio, se logró estructurar un sistema de base de datos robusto que responde a las necesidades operativas de GlobalComerce S.A.

El diseño relacional permite mantener la consistencia de los datos y facilita la recuperación de información relevante para la toma de decisiones en distintos niveles, desde la gerencia hasta los departamentos de ventas y producción.

Las consultas implementadas permiten obtener datos clave como el total de ventas, productos más vendidos, movimientos de inventario, y gestión de empleados, lo que refleja la flexibilidad y capacidad del sistema para adaptarse a las necesidades de una empresa en crecimiento.

Como reflexión final, este laboratorio ha sido una excelente oportunidad para aplicar conocimientos adquiridos en el curso de Bases de Datos, reforzando el uso de SQL en un entorno práctico.

Asimismo, ha quedado evidenciado que una correcta planificación y diseño de la base de datos es fundamental para garantizar un rendimiento óptimo y una gestión eficiente de los recursos de una empresa.