

b) Modificación de la estructura de la base de datos:

Paso 1: Para modificar la estructura solo se tiene que escribir el siguiente código:

```
ON PRIMARY (
    NAME = modulo_data,
    FILENAME = 'D:\modulo\modulo_tienda.mdf',
    SIZE = 10MB,
    MAXSIZE = 20MB,
    FILEGROWTH = 10%
)
LOG ON (
    NAME = modulo_log,
    FILENAME = 'D:\modulo\modulo_tienda.ldf',
    SIZE = 5MB,
    MAXSIZE = 8MB,
    FILEGROWTH = 1MB
)
```

Paso 2: Seleccionar el código y darle a "Execute":

```
USE modulo_tienda
go
ON PRIMARY (
    NAME = modulo_data,
    FILENAME = 'D:\modulo\modulo_tienda.mdf',
    SIZE = 10MB,
    MAXSIZE = 20MB,
    FILEGROWTH = 10%
)
LOG ON (
    NAME = modulo_log,
    FILENAME = 'D:\modulo\modulo_tienda.ldf',
    SIZE = 5MB,
    MAXSIZE = 8MB,
    FILEGROWTH = 1MB
)
```

96 %

Messages

Commands completed successfully.

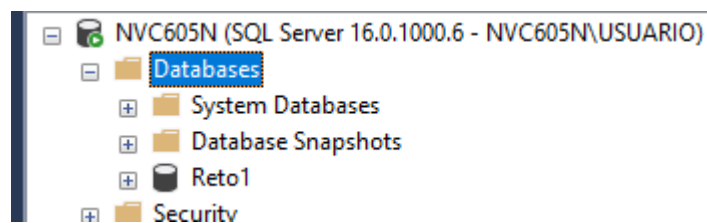
Completion time: 2025-06-19T17:15:05.0565915-05:00

c) Eliminación de una base de datos:

Paso 1: Para eliminar una base de datos solo se tiene que escribir el siguiente código:

```
USE master;
GO
ALTER DATABASE modulo_tienda SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
GO
DROP DATABASE modulo_tienda;
GO
```

Y ya se habrá eliminado:



2. Administración de Objetos de Base de Datos:

A. Creación y modificación de tablas:

Paso 1: Restaurar la base de datos, seleccionándola y ejecutándola:

```
SQLQuery1.sql - NVC605N\USUARIO (81)
SQLQuery1.sql - NVC605N\USUARIO (81)

USE master;
GO
CREATE DATABASE modulo_tienda
ON PRIMARY (
    NAME = modulo_data,
    FILENAME = 'D:\modulo\modulo_tienda.mdf',
    SIZE = 10MB,
    MAXSIZE = 20MB,
    FILEGROWTH = 10%
)
LOG ON (
    NAME = modulo_log,
    FILENAME = 'D:\modulo\modulo_tienda.ldf',
    SIZE = 5MB,
    MAXSIZE = 8MB,
    FILEGROWTH = 1MB
);
GO
```

Paso 2: Usar la base datos y crear la tabla como el siguiente ejemplo:

SQLQuery3.sql - N...605N\USUARIO (61)) * - SQLQuery1.sql

```
USE modulo_tienda;
GO
CREATE TABLE Productos (
    IdProducto INT PRIMARY KEY IDENTITY(1,1),
    Nombre NVARCHAR(100) NOT NULL,
    Precio DECIMAL(10, 2) NOT NULL,
    Stock INT NOT NULL,
    FechaRegistro DATETIME DEFAULT GETDATE()
);
GO
```

B. Creacion y modificación de Vistas:

Paso 1: Para crear una vista se usa el siguiente código:

```
CREATE VIEW Vista_ProductosDisponibles AS
SELECT IdProducto, Nombre, Precio, Stock
FROM Productos
WHERE Stock > 0;
GO
```

Paso 2: Para modificar esa vista se usa el siguiente código (ALTER):

```
ALTER VIEW Vista_ProductosDisponibles AS
SELECT IdProducto, Nombre, Precio, Stock, FechaRegistro
FROM Productos
WHERE Stock > 0;
GO
```

C. Creacion y modificación de INDICES:

Paso 1: Para crear un índice y mejorar el rendimiento de consultas se usa el siguiente código:

```
CREATE INDEX idx_Productos_Nombre
ON Productos (Nombre);
GO
```

Paso 2: Para modificar un índice, se elimina el creado y se vuelve a crear uno nuevo, se usa el siguiente código:

```
DROP INDEX idx_Productos_Nombre ON Productos;
GO
```

3. Mantenimiento de Base de Datos:

A. Copias de Seguridad y Restauracion de Base de Datos:

Paso 1: Para hacer una copia de seguridad, solo escribes el siguiente código:

```
BACKUP DATABASE modulo_tienda
TO DISK = 'D:\Backups\modulo_tienda.bak'
WITH FORMAT,
    MEDIANAME = 'BackupModuloTienda',
    NAME = 'Full Backup of modulo_tienda';
GO
```

Paso 2: Para restaurar una base de datos (si es que existe una copia de seguridad), solo se escribe este código.

```
-- Si existe, primero eliminarla (¡con cuidado!)
DROP DATABASE IF EXISTS modulo_tienda;
GO

-- Restaurar la base de datos desde el archivo .bak
RESTORE DATABASE modulo_tienda
FROM DISK = 'D:\Backups\modulo_tienda.bak'
WITH MOVE 'modulo_data' TO 'D:\modulo\modulo_tienda.mdf',
    MOVE 'modulo_log' TO 'D:\modulo\modulo_tienda.ldf',
    REPLACE;
GO
```

B. Optimizacion del rendimiento de las bases de datos.

Para esto, hay varias maneras de optimizar el rendimiento de la base de datos, aquí están las principales:

1. Crear Índices Adecuados:

Los índices aceleran las búsquedas, especialmente en columnas usadas en WHERE, JOIN, ORDER BY.

```
-- índice simple en la columna Nombre
CREATE INDEX idx_Productos_Nombre ON Productos(Nombre);
```

2. Eliminar Fragmentación de Índices:

Reorganizar o reconstruir índices mejora el rendimiento de consultas:

```
-- Reorganizar índice (menos costoso)
ALTER INDEX ALL ON Productos REORGANIZE;

-- Reconstruir índice (más efectivo, pero más pesado)
ALTER INDEX ALL ON Productos REBUILD;
```

3. Actualizar estadísticas:

SQL Server usa estadísticas para generar planes de ejecución eficientes.

```
-- Actualiza estadísticas de una tabla
UPDATE STATISTICS Productos;
```

4. Evitar Cursores y Usar Consultar Set-Based:

Los cursores son lentos. Usa consultas que operan sobre conjuntos de datos en lugar de filas individuales.

Evita esto:

```
sql

DECLARE cursorProductos CURSOR FOR
SELECT Nombre FROM Productos;
```

Mejor usa:

```
sql

SELECT Nombre FROM Productos WHERE Stock > 0;
```

5. Usar vistas indexadas (materializadas)

Son vistas que almacenan físicamente datos para acelerar consultas complejas.

```
-- Vista indexada (solo si se cumplen ciertas condiciones)
CREATE VIEW VistaStockTotal
WITH SCHEMABINDING
AS
SELECT COUNT_BIG(*) AS Total, SUM(Stock) AS StockTotal
FROM dbo.Productos;
GO

CREATE UNIQUE CLUSTERED INDEX idx_VistaStockTotal
ON VistaStockTotal (Total);
```

6. Reducir el tamaño de transacciones y uso de NOLOCK donde sea seguro:

```
-- Leer sin bloquear otras operaciones (puede leer datos no confirmados)
SELECT * FROM Productos WITH (NOLOCK);
```

7. Eliminar datos obsoletos

```
-- Ejemplo: eliminar productos sin stock y sin movimiento en 1 año
DELETE FROM Productos
WHERE Stock = 0 AND FechaRegistro < DATEADD(YEAR, -1, GETDATE());
```

C. Solución de Problemas de Bases de Datos:

Para esto, hay varias maneras de solucionar problemas que se presentan, aquí están las algunas:

1. Ver procesos que están bloqueando la base de datos

```
-- Ver procesos activos
EXEC sp_who2;
GO
```

2. Verificar y corregir corrupción de base de datos

```
-- Revisar corrupción (no cambia nada)
DBCC CHECKDB ('modulo_tienda');
GO
```

3. Verificar fragmentación de índices y corregirla

```
-- Ver fragmentación de índices
SELECT
    OBJECT_NAME(object_id) AS Tabla,
    name AS Indice,
    avg_fragmentation_in_percent
FROM sys.dm_db_index_physical_stats (DB_ID(), NULL, NULL, NULL, 'LIMITED')
WHERE avg_fragmentation_in_percent > 10;

-- Reparar
ALTER INDEX ALL ON Productos REBUILD;
```

4. Detectar Tablas Sin Indices

```
-- Tablas sin índices
SELECT t.name AS TablaSinIndice
FROM sys.tables t
LEFT JOIN sys.indexes i ON t.object_id = i.object_id AND i.type_desc <> 'HEAP'
WHERE i.object_id IS NULL;
```

5. Ver tamaño de archivos y uso de espacio

```
-- Ver tamaño actual de archivos MDF y LDF
SELECT
    name AS Archivo,
    physical_name,
    size * 8 / 1024 AS TamañoMB
FROM sys.database_files;
```

6. Verificar claves foráneas inválidas

```
-- Detectar registros que violan las claves foráneas
DBCC CHECKCONSTRAINTS;
GO
```