

IES CHAN DO MONTE  
C.S. de Desarrollo de Aplicaciones Multiplataforma  
Módulo Base de datos

## UNIDAD 4 ACTIVIDAD2: Lenguaje de Definición de Datos (DDL) CREACIÓN DE BASE DE DATOS

### Índice

<b>1.</b>	<b>Introducción al lenguaje de definición de datos (DDL)</b>	<b>2</b>
<b>2.</b>	<b>Archivos de bases de datos de SQL Server</b>	<b>3</b>
2.1	Tipos de archivos	3
2.2	Nombres de archivo lógico y físico	4
2.3	Como funciona el registro de transacciones	4
2.3.1	Forma de trabajar del registro de transacciones.	5
<b>3.</b>	<b>Grupos de archivos.</b>	<b>6</b>
3.1	Estrategia para el relleno de archivos y grupos de archivos	7
3.2	Tipos de grupos de archivos	7
3.3	Consideraciones acerca del rendimiento	8
<b>4.</b>	<b>Cómo crear, modificar y eliminar bases de datos.</b>	<b>8</b>
4.1	Como crear bases de datos	8
4.1.1	Instrucción CREATE DATABASE	9
4.2	Visualizar información de las bases de datos y los grupos de archivos	12
4.3	Eliminar una base de datos	13
4.3.1	Consideraciones para eliminar una base de datos	13
4.4	Estados de base de datos	13
4.5	Modificación de un base de datos	14
4.6	Establecer las opciones de base de datos:	17
4.6.1	Establecimiento de las opciones utilizando ALTER DATABASE.	18

# 1. Introducción al lenguaje de definición de datos (DDL)

Un sistema de bases de datos debe proporcionar una serie de lenguajes diferentes: para especificar el esquema relacional de base de datos, para expresar las consultas y actualizaciones de la base de datos y para la administración de usuarios y seguridad en la base de datos.

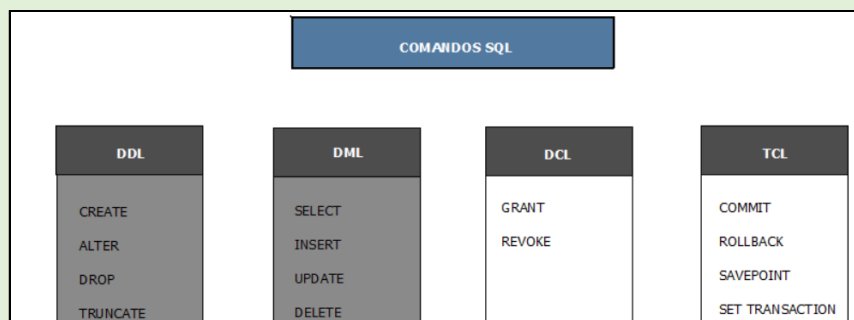
Los **lenguajes de base de datos se clasifican:**

- **Lenguaje de definición de datos (DDL).** Para **definir los esquemas relacionales** de la base de datos
- **Lenguaje de manipulación de datos (DML).** Para **manipular los datos** de la base de datos
- **Lenguaje de control de datos (DCL).** Para la **administración de usuarios y seguridad** en la base de datos.
- **Lenguaje de control de Transacciones (TCL).** Permiten **controlar y gestionar transacciones** para mantener la integridad de los datos dentro de las sentencias SQL.

El **lenguaje de gestión de bases de datos relacionales** más conocido en la actualidad es el **SQL** (Structured Query Language), que es un **lenguaje estandar internacional**, comúnmente aceptado por los fabricantes de Sistemas de bases de datos y representa una combinación de **LDD + LMD+LCD+TCL**.

**El SQL es un lenguaje declarativo:** sólo hay que indicar qué se quiere hacer. En cambio, en los lenguajes procedimentales es necesario especificar cómo hay que hacer cualquier acción sobre la base de datos. Las sentencias de SQL son:

- Las **sentencias DDL** son las que se encargan de la creación y modificación de la estructura de los objetos de la base de datos. Existen cuatro operaciones básicas: **CREATE, ALTER, DROP y TRUNCATE**.
- Las **sentencias LMD** son las que proporciona acceso y actualización de los datos contenidos en las tablas. Las instrucciones LMD son: **SELECT, INSERT, UPDATE Y DELETE**. También se incluyen las sentencias de control de concurrencia para la transacción como son **COMMIT Y ROLLBACK**.
- Las **sentencias DCL** permiten conceder y revocar permiso de accesos a los datos de la base de datos y son: **GRANT y REVOKE**
- Las **sentencias TCL** permiten contralar y gestionar las transacciones: **COMMIT, ROLLBACK, SAVEPOINT**



El lenguaje SQL en SQL SERVER se llama **TRANSACT-SQL**.

Las **sentencias SQL comienzan con un verbo**, una palabra clave que describe lo que la sentencia hace: INSERT, DELETE, CREATE, etc... y la sentencia continua **con una o más cláusulas**. Una cláusula puede especificar los datos sobre los que la sentencia debe actuar o proporciona más detalles acerca de lo que la sentencia debe hacer. Todas las cláusulas comienzan también con una palabra reservada, por ejemplo: WHERE, FROM, INTO etc...

**El estándar SQL ha pasado por muchos cambios durante los años, en los cuales se han añadido una gran cantidad de nuevas funcionalidades** al estándar, como el soporte para XML, triggers, expresiones regulares, consultas recursivas, secuencias estandarizadas y mucho más.

En muchos casos, el comportamiento de la base de datos para el almacenamiento de archivos o índices no está bien definido y depende de los proveedores de las distintas implementaciones SQL para decidir cómo se comportará la base de datos.

En el siguiente esquema se ve **la evolución del lenguaje SQL**:

Año	Nombre	Alias	Comentarios
1986	SQL-86	SQL-87	Primera publicación hecha por ANSI. Confirmada por la <a href="#">Organización Internacional de Normalización</a> en 1987.
1989	SQL-89		Revisión menor.
1992	SQL-92	SQL2	Revisión mayor.
1999	SQL:1999	SQL2000	Se agregaron <b>expresiones regulares</b> , consultas recursivas (para relaciones jerárquicas), triggers y algunas características orientadas a objetos.
2003	SQL:2003		Introduce algunas características de <b>XML</b> , cambios en las funciones, estandarización del objeto sequence y de las columnas autonuméricas. <sup>3</sup>
2005	SQL:2005		ISO/IEC 9075-14:2005 Define las maneras en las cuales SQL se puede utilizar conjuntamente con XML. Define maneras de importar y guardar datos XML en una base de datos SQL, manipulándolos dentro de la base de datos y publicando el XML y los datos SQL convencionales en forma XML. Además, proporciona facilidades que permiten a las aplicaciones integrar dentro de su código SQL el uso de XQuery, lenguaje de consulta XML publicado por el W3C (World Wide Web Consortium) para acceso concurrente a datos ordinarios SQL y documentos XML.
2008	SQL:2008		Permite el uso de la cláusula ORDER BY fuera de las definiciones de los cursores. Incluye los disparadores del tipo INSTEAD OF. Añade la sentencia TRUNCATE. <sup>4</sup>
2011	SQL:2011		Datos temporales (PERIOD FOR). Mejoras en las funciones de ventana y de la cláusula FETCH.
2016	SQL:2016		Permite búsqueda de patrones, funciones de tabla polimórficas y compatibilidad con los ficheros JSON.

Los diferentes proveedores de SGBDR, sean comerciales o no comerciales, incluyen en sus implementaciones del lenguaje SQL añadidos particulares, con lo cual ocasiona variaciones de diversa importancia entre las diferentes implementaciones. El almacenamiento de archivos o índices en las bases de datos no está bien definido en los estándares SQL y depende de las implementaciones de los diferentes proveedores de SGBDR. El soporte al estándar SQL-92 es uno de lo más amplios y extendidos.

En este capítulo vamos a ver las sentencias de definición de datos. Antes de ponernos a crear una base de datos es importante entender dos conceptos: cómo almacena SQL Server los datos y la función del archivo de registro de transacciones.

## 2. Archivos de bases de datos de SQL Server

### 2.1 Tipos de archivos

Las bases de datos se almacenan en sus propios archivos. Cuando se [crea una base de datos](#) se establece una **estructura de almacenamiento de datos**. Esta estructura incluye, **al menos**, un [archivo de datos](#) y un [archivo de registro de transacciones](#). Tipos de archivos de bases de datos.

Hay [tres tipos de archivos de bases de datos](#):

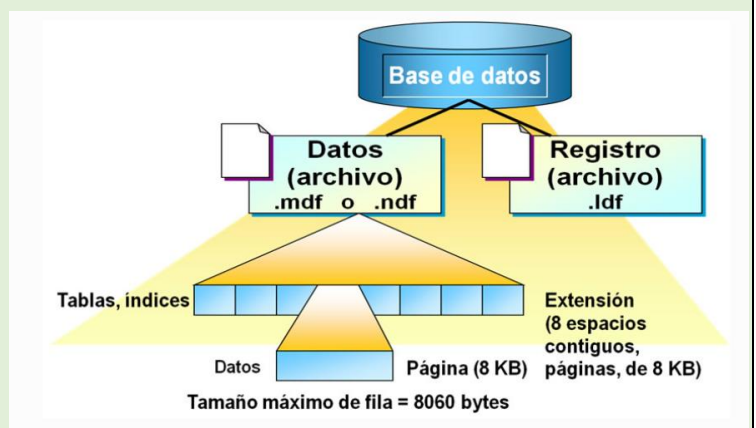
- **Archivo de datos principales.** Cada base de datos debe **incluir única y exclusivamente un archivo de datos principal**. La extensión recomendada para los nombres de archivos de datos principales es **.mdf**. Contiene la información de **inicio para la base de datos y se utiliza para almacenar datos**.

- **Archivos de datos secundarios.** Estos archivos pueden **contener todos los datos y objetos que no estén incluidos en el archivo de datos principal**.

Las bases de datos no necesitan archivos de datos secundarios si el archivo principal puede contener todos los datos de la base de datos, por lo tanto, este **tipo de archivos es opcional**.

Algunas bases de datos pueden ser muy grandes y necesitar varios archivos de datos secundarios, o bien utilizar archivos secundarios en unidades de disco distintas para distribuir los datos en varios discos. La extensión de nombre de archivo recomendada para los archivos de datos secundarios es **.ndf**.

- **Archivos de registros.** Estos archivos contienen toda la **información de registro de transacciones requerida para recuperar la base de datos**. Cada base de datos **debe tener al menos un archivo de registro**, aunque puede haber varios. La extensión de nombre de archivo recomendada para los archivos de registro es **.ldf**.



Nota: SQL Server no exige las extensiones de nombre de archivo .mdf, .ndf y .ldf, pero estas extensiones ayudan a identificar las distintas clases de archivos y su uso.

El término archivo de base de datos generalmente significa cualquiera de los tres tipos de archivos. El término **archivo de datos** se refiere bien al archivo de **datos primario o al secundario**. El término **archivo de registro** se refiere a una parte del **registro de transacción** de la base de datos.

## 2.2 Nombres de archivo lógico y físico






Los **archivos de SQL Server tienen dos nombres**:

- **nombre lógico** (logical **file name**): es el nombre que se utiliza para **hacer referencia al archivo en todas las instrucciones Transact-SQL**. El nombre de archivo lógico tiene que cumplir las reglas de los identificadores de SQL Server y tiene que **ser único** entre los nombres de archivos lógicos de la base de datos.

- **nombre físico** (os **file name** ()): es el **nombre del archivo físico** que incluye la **ruta de acceso al directorio**. Debe seguir las reglas para nombres de archivos del sistema operativo.

La siguiente ilustración muestra ejemplos de los nombres de archivo lógico y físico de una base de datos creada en una instancia predeterminada de SQL Server.

En SQL Server, **las ubicaciones de todos los archivos** de una base de datos se guardan tanto en el **archivo principal de la base de datos** como en la base de datos **maestra**.

<b>MyDB_primary</b> c:\Archivos de programa\Microsoft SQL Server\MSSQL.1\MSSQL\Data\MyDB_Data1.mdf	
Archivo de datos principal	
<b>MyDB_secondary1</b> c:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Data\MyDB_Data2.ndf	
Archivo de datos secundario	
<b>MyDB_secondary2</b> c:\Archivos de programa\Microsoft SQL Server\MSSQL.1\MSSQL\Data\MyDB_Data3.ndf	
Archivo de datos secundario	
<b>MyDB_log1</b> c:\Archivos de programa\Microsoft SQL Server\MSSQL.1\MSSQL\Data\MyDB_Log1.ldf	
Log file	
<b>MyDB_log2</b> c:\Archivos de programa\Microsoft SQL Server\MSSQL.1\MSSQL\Data\MyDB_Log2.ldf	
Archivo de registro	

## 2.3 Como funciona el registro de transacciones

Los archivos de registro de transacciones contienen toda la **información necesaria para recuperar la base de datos en caso de fallo del sistema**.

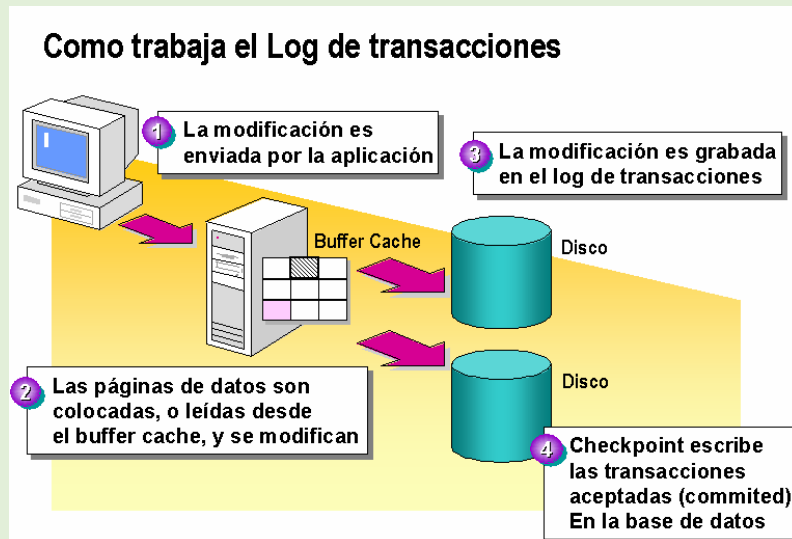
La **recuperación implica la capacidad del SQL Server para superar un fallo del sistema y volver a ejecutar (recuperar) las transacciones**.

- El **registro de transacciones**:
  - ✓ **En él se graban todas las transacciones y las actualizaciones de datos** -instrucciones **INSERT, UPDATE y DELETE**- **cuando son ejecutadas**.  
SQL Server permite, sin embargo, dos métodos de carga masiva de datos pasando por alto el registro de transacción: el programa de copia masiva (BCP) y la instrucción **SELECT INTO**. Estas operaciones no se graban en el registro de transacciones.
  - ✓ **Es el único medio** para la recuperación de las transacciones si hay un fallo.
  - ✓ **No es opcional**.
  - ✓ **No se puede desactivar**
  - ✓ Es importante que el **registro de transacciones nunca se llene**, ya que esto puede impedir modificaciones en la base de datos.
- Cada vez que hay una **actualización de datos en la base de datos**:
  - ✓ **No se graban en el disco de forma inmediata**,  
*Si lo hiciera, en un sistema grande (como el sistema de un banco) se trabajaría muy lentamente, porque la transacción tendría que esperar a que se terminara de escribir antes de proseguir, lo que provocaría un retraso en todas las transacciones.*
  - ✓ **No se escriben los cambios en la base de datos hasta que no se confirmen** (es decir, se ejecute la instrucción **COMMIT TRANSACTION**).  
*SQL Server, para mantener la integridad de la base de datos, anota en el registro de transacciones toda la información necesaria para que la transacción confirmada sea ejecutada de nuevo si los datos se perdieron.*
  - ✓ **Con el retraso de pasar los cambios al disco, un fallo en el sistema puede dejar la base de datos en un estado inconsistente** debido a que los cambios hechos en la base de datos pudieran no haberse pasado a disco o los cambios pasados a disco pudieran no haber sido confirmados.

### 2.3.1 Forma de trabajar del registro de transacciones.

El **proceso de registro** sucede de la manera siguiente:

1. La aplicación envía una actualización de datos.
2. Las páginas de datos afectadas se cargan desde los archivos de datos a la memoria (llamada **caché de datos**) si las páginas no están ya en la caché de datos de una consulta anterior.
3. Cada instrucción de modificación de datos se registra en el registro a medida que se ejecuta. El cambio siempre se registra en el registro y se escribe en el archivo de registro antes de que se realice el cambio en la base de datos. Este tipo de registro se llama un registro de escrituras adelantadas.
4. Una vez que las páginas de datos están en la caché de datos y que las páginas de registro están registradas en el disco, en el archivo de registro de transacciones, el proceso de punto de comprobación escribe todas las transacciones confirmadas a la base de datos en el disco.



#### ■ Una transacción:

- ✓ Puede incluir varias instrucciones de actualización de datos.
- ✓ Cada transacción comienza con el marcador **BEGIN TRANSACTION**.
- ✓ Si la aplicación completa correctamente todas las actualizaciones de datos, la transacción termina con la instrucción **COMMIT TRANSACTION**. Esta transacción es enviada como una transacción confirmada.

#### ■ Durante una operación normal, el **proceso de punto de comprobación (checkpoint)**

- *Un punto de comprobación (checkpoint) es una operación que se usa para sincronizar los ficheros físicos de datos con la caché de datos con el fin de reducir el tiempo necesario para la recuperación en caso de fallo del sistema. En un punto de comprobación, las páginas de datos desfasadas (que están en la caché de datos y no se han escrito en el disco) se graban en el disco.*
- ✓ Realiza, de manera rutinaria, una búsqueda de transacciones confirmadas que no se han escrito en el archivo de datos y escribe estas modificaciones en el archivo de datos.
- ✓ Registra que ha sido escrita en el archivo de datos.
- ✓ **En el registro de transacciones**, cada cierto volumen de información, **se graba un punto de comprobación** (checkpoint). En **un punto de comprobación**, SQL Server se asegura de que todos los registros del registro de transacciones y las páginas de la base de datos modificadas se escriban en el disco. Por lo tanto, representa el punto en el que la recuperación de inicio debe empezar a rehacer transacciones.

#### ■ Cuando SQL Server se reinicia después de una caída del sistema,

- ✓ El mecanismo de recuperación se inicia automáticamente.
- ✓ Se utiliza el registro de transacciones para determinar que transacciones necesitan recuperarse y cuáles no.
- ✓ SQL Server comienza leyendo el registro de transacciones en el último punto de comprobación.
- ✓ Realiza todas las transacciones confirmadas (begin y commit) que no han sido comprobadas y devuelve (elimina) cualquier transacción incompleta.



### 3. Grupos de archivos.

Para **simplificar la administración de los archivos** de base de datos múltiples, SQL Server proporciona **grupos de archivos**.

- **Grupos de archivos:** son *colecciones con nombre* que se utilizan para *facilitar la colocación de datos y tareas administrativas* como las operaciones de copia de seguridad y restauración.

- ✓ Cada base de datos tiene **un grupo de archivos predeterminado** :  
Los objetos que se crean en la base de datos se crean automáticamente en este grupo de archivos si no se especifica ningún grupo de archivos).
- ✓ Se pueden **crear tantos grupos de archivos adicionales como se necesiten**.
- ✓ SQL Server utiliza **grupos de archivos** para **distribuir los datos de tablas o índices entre los discos**.  
Los objetos de la base de datos, como las tablas o los índices, pertenecen a un grupo de archivos.
- ✓ Un **grupo de archivos** contiene, **como mínimo, un archivo físico**.
- ✓ Para **mejorar el rendimiento** se puede **controlar la ubicación de los datos mediante la creación de tablas e índices en grupos de archivos diferentes**.

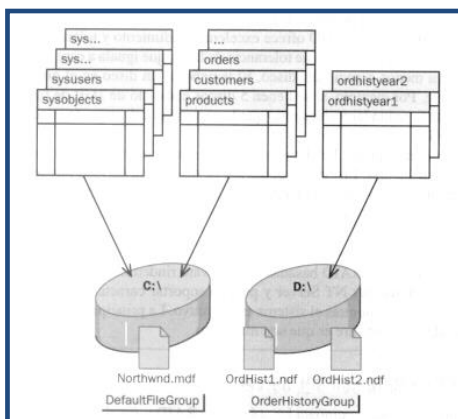
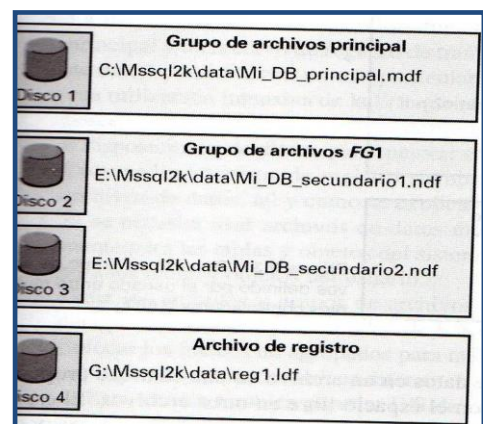
Por ejemplo:

- Se podría poner una tabla que es utilizada intensivamente en un grupo de archivos en un grupo grande de discos (compuesto por ejemplo de 10 discos) y poner otra tabla que se utiliza de manera menos intensiva en otro grupo de archivos creado en otro grupo de disco distinto más pequeño (compuesto de 4 disco, por ejemplo). Así la tabla a la que se accede más frecuentemente estará repartida en el número mayor de discos, permitiendo una E/S con mayor paralelismo. Se podría usar también **un grupo de archivos para repartir los datos entre varios disco**. En la siguiente figura, se muestra un grupo de archivos (FG1) que incluye dos archivos secundarios, uno en la unidad E y otro en la unidad de disco E y otro en la F, junto al archivo principal en la unidad C y el de registro en la unidad G.

Las consultas de datos de la tabla se distribuirán por los tres discos, con lo que mejorará el rendimiento.

Los archivos y grupos de archivos permiten agregar fácilmente nuevos archivos a discos nuevos.

Se pueden asignar tablas específicas, índices o los datos de texto, ntext e imagen desde una tabla a un determinado grupo de archivos..



En la figura siguiente los archivos Ordhist1.ndf y Ordhist2.ndf son situados en discos separados para mantener los archivos que son altamente consultados de los que son altamente modificados y reducir los conflictos de controladores de discos y están en el grupo de archivos OrderHistoryGroup.

Los administradores de sistemas pueden **hacer copias de seguridad y restaurar archivos individuales o grupos de archivos** en vez de hacer copias de seguridad o restaurar una base de datos completa.

Se pueden **crear grupos de archivos cuando se crea la base de datos** (Con la instrucción **CREATE DATABASE**) o bien posteriormente, cuando se agregan más archivos a la base de datos (Con **ALTER DATABASE**).

Hay que tener presente que:

- Un archivo o un grupo de archivos **no puede ser utilizado** por más de una base de datos.
- Una vez que se han agregado archivos a la base de datos, no es posible moverlos a otro grupo de archivos. Si se quiere mover un archivo, hay que borrarlo y volver a crearlo.
- Además, un archivo **puede pertenecer únicamente** a un grupo de archivos.
- Los archivos de registro no forman parte de un grupo de archivos. El espacio de registro es administrado separadamente del espacio de datos.
- Los **grupos de archivos** son utilizados solamente para **administrar archivos de datos**.

## 3.1 Estrategia para el relleno de archivos y grupos de archivos

- ✓ Los **grupos de archivos** utilizan una estrategia de relleno proporcional entre **todos los archivos de cada grupo de archivos**.
- ✓ A medida que se escriben los datos en el grupo de archivos, Microsoft SQL Server escribe una cantidad proporcional al espacio libre del archivo en cada archivo del grupo, en vez de escribir todos los datos en el primer archivo hasta que se llene y, a continuación, sigue escribiendo en el siguiente archivo. Por lo tanto, los **grupos de archivos utilizan una estrategia de relleno proporcional** entre todos los archivos de cada grupo de archivos.  
 Por ejemplo, si archivo1 tiene 600 MB de espacio libre y archivo2 tiene 100 MB de espacio libre, SQL Server utiliza más espacio del archivo1 y mucho menos espacio del archivo2.
- ✓ Si la base de datos está configurada para crecer automáticamente (crecimiento automático), **cuando se llenan todos los archivos de un grupo de archivos**, SQL Server **expande automáticamente los archivos por turnos** para que quepan más datos.

*Por ejemplo, un grupo de archivos consta de tres archivos configurados para crecer automáticamente. Cuando se agota el espacio de todos los archivos del grupo de archivos, sólo se expande el primer archivo. Cuando se llena el primer archivo y no se pueden escribir más datos en el grupo de archivos, se expande el segundo archivo. Cuando se llena el segundo archivo y no pueden escribirse más datos en el grupo de archivos, se expande el tercer archivo. Si se llena el tercer archivo y no se pueden escribir más datos en el grupo de archivos, se vuelve a expandir el primer archivo, y así sucesivamente.*

## 3.2 Tipos de grupos de archivos

SQL Server ofrece los siguientes tres tipos de grupos de archivos:

1. **El grupo de archivos principal:** contiene el archivo de datos principal y cualquier archivo que no forme parte de ningún grupo de archivos.
  - ✓ Todas los **objetos y tablas del sistema** son colocadas siempre en el **grupo de archivos principal**, independientemente si este no es el predeterminado.
  - ✓ Por esta razón es muy importante que el **grupo de archivos principal disponga de espacio suficiente**.
  - ✓ Inicialmente el grupo de archivos principal es el grupo de archivos predeterminado. Se puede cambiar el grupo predeterminado en cualquier momento.
2. **Grupos de archivos definidos por el usuario:** que son especificados con la palabra clave FILEGROUP en las instrucciones CREATE DATABASE o ALTER DATABASE.
3. **Grupos de archivos predeterminados:** Cuando se crean objetos en la base de datos **sin especificar a qué grupo de archivos pertenecen**, se asignan al **grupo de archivos predeterminado**. **Siempre existe un grupo de archivos designado como predeterminado**.
  - ✓ Los archivos del grupo de archivos predeterminado deben ser lo suficientemente grandes como para dar **cabida a todos los objetos nuevos no asignados a otros grupos de archivos**.

- **Cualquier grupo de archivos existente, excepto el grupo de archivos principal, puede marcarse como de sólo lectura.**  
 Para evitar actualizaciones accidentales es conveniente colocar las tablas que no deban modificarse (como las de datos históricos) en uno o varios grupos de archivos y, a continuación, marcarlos como de sólo lectura.
- Es muy importante ajustar el tamaño del grupo de archivos principal correctamente.
  - ✓ **Debe ser lo suficientemente grande como para contener todas las tablas del sistema**, y si se mantiene como el grupo de archivos predeterminado, para contener todas las tablas no asignadas a un grupo de archivos definido por el usuario.
  - ✓ Si el grupo de archivos principal se queda sin espacio, **no puede ser añadida nueva información a las tablas del sistema**.  
 El grupo de archivos principal sólo podrá llenarse si la opción de crecimiento está desactivada o el disco que contiene al grupo de archivos principal está lleno. Para permitir crecer al grupo de archivos principal, se reactiva la opción de crecimiento automático o se libera más espacio en el disco.
- **Si un grupo de archivos definido por el usuario se queda sin espacio**, sólo se verán afectados los archivos de usuario que estén situados en el grupo de archivos afectado.

### 3.3 Consideraciones acerca del rendimiento

Si se quiere **obtener el mejor rendimiento de las base de datos**, hay que considerar las siguientes directrices:

- La mayoría de las bases de datos funcionarán **correctamente** con sólo **un archivo de datos principal** y **un archivo de registro de transacciones**. Este es el diseño recomendado para bases de datos que no hacen un uso particularmente intenso de la E/S.
- Si se **tiene un sistema con una utilización intensiva de la E/S** que requiera **muchas unidades de disco**, probablemente se desee utilizar **grupos de archivos** definidos por el usuario **que repartan los datos entre discos** para **mejorar el rendimiento de la E/S paralela**. Esta implementación permite construir bases de datos muy grandes que pueden mejorar el rendimiento, ya que los discos funcionan simultáneamente.
- Si se **utiliza varios archivos**, mejor es **crear un segundo grupo de archivos** para el archivo adicional y conviértalo en el **grupo de archivos predeterminado**. De ese modo, el archivo principal sólo contendrá objetos y tablas del sistema.
- Colocar **los archivos de registro** siempre **en discos distintos** de los discos que contengan **archivos de datos** para que las escrituras en el registro de transacciones simultáneas no compitan con las acciones de INSERT, UPDATE o DELETE a las bases de datos.
- Utilice **grupos de archivos para colocar objetos de bases de datos en discos separados**. Esto le permite planear estrategias individuales de copias de seguridad basadas en la frecuencia en que los datos son modificados. Si tiene un grupo de archivos que cambian a menudo, puede hacer copias de seguridad solamente de aquellas tablas u objetos frecuentemente.
- Poner las **tablas diferentes que se usen en las mismas consultas de combinación en discos físicos diferentes**. De ese modo, el rendimiento mejorará debido a la búsqueda de datos combinados que realizan las operaciones de E/S en paralelo en los discos.
- Otro beneficio de la utilización de grupos de archivos es que SQL permite **realizar copias de seguridad de la base de datos basadas en un archivo o grupo de archivos**. Si la base de datos es demasiado grande como para hacer copia de seguridad de toda ella de una sola vez, se puede hacer copia de seguridad de sólo ciertas partes.

## 4. Cómo crear, modificar y eliminar bases de datos.

### 4.1 Como crear bases de datos

Antes de **crear una base de datos**, hay que tener en cuenta lo siguiente:

- Se debe, como mínimo, **disponer de permiso CREATE DATABASE, CREATE ANY DATABASE o ALTER ANY DATABASE**.
  - Cada base de datos **tiene un propietario** que puede realizar actividades especiales en ella. El **usuario que crea la base de datos se convierte en su propietario**. El propietario de la base de datos se puede cambiar mediante *sp\_changedbowner* (Transact-SQL).
- El **nombre de la base de datos** debe **ajustarse a las reglas establecidas para los identificadores**.
- **Todos los objetos** definidos por el usuario **de la base de datos model** se **copiarán en todas las bases de datos recién creadas**.  
Se puede agregar a la base de datos model todos los objetos (tablas, vistas, procedimientos almacenados, tipos de datos, etc.) que desee incluir en todas las bases de datos recién creadas.
  - Es muy **importante hacer una copia de seguridad de la base de datos master antes de crear, modificar o eliminar una base de datos**, pues la información acerca de cada base de datos de SQL Server está almacenada en la tabla del sistema **sysdatabases** en la base de datos **master**.
  - Las **instrucciones DDL generan acciones que no se pueden deshacer** (salvo que dispongamos de alguna copia de seguridad). Por lo que hay que tener mucha precaución a la hora de utilizarlas

En el **proceso de creación** de una base de datos :

- Se **define su nombre**
- Se establecen sus **propiedades** y **se fija la ubicación de sus archivos**.



Al crear una base de datos, SQL Server actúa de la siguiente manera:

1. **SQL Server utiliza una copia de la base de datos model** para **inicializar la base de datos y sus metadatos**.
2. Se asigna **un GUID a la base de datos**.  
*GUID (Globally Unique Identifier) es un tipo de datos binarios de 16 bytes de SQL Server que es globalmente único en tablas, base de datos y servidores*
3. A continuación, el SQL Server rellena el resto de la base de datos **con páginas vacías** hasta completar el espacio inicial reservado

### 4.1.1 Instrucción CREATE DATABASE

El comando SQL de **creación de una base de datos** es **CREATE DATABASE**. Este comando crea una base de datos con el nombre que se indique. Ejemplo:

```
CREATE DATABASE prueba;
```

Si no se especifica más parámetros crea la base de datos con las opciones por defecto. Pero normalmente se indican más parámetros:

```
CREATE DATABASE nombreBaseDatos
[ ON [ PRIMARY ]
  [ <filespec> [ ,...n ]
    [ , <grupoArchivos> [ ,...n ] ]
  [ LOG ON { <filespec> [ ,...n ] } ]
  ]
[ COLLATE nombre_intercalación ]
]
<filespec> ::=
(
  NAME = nombreArchivoLógico,
  FILENAME = { nombreArchivoFísico | 'rutaFilestream' }
  [ , SIZE = tamaño [ KB | MB | GB | TB ] ]
  [ , MAXSIZE = { tamañoMáximo [ KB | MB | GB | TB ] | UNLIMITED } ]
  [ , FILEGROWTH = incrementoCrecimiento [ KB | MB | GB | TB | % ] ]
) [ ,...n ]
<grupoArchivos> ::=
FILEGROUP nombreGrupoArchivo [ CONTAINS FILESTREAM ] [ DEFAULT ] <filespec>
```

- **nombreBaseDatos:** Es el nombre de la nueva base de datos. Deben ser **únicos** en una instancia de SQL Server y cumplir las reglas de los **identificadores**. La longitud máxima es de 128 caracteres.
- **ON:** Sirve para especificar en **qué grupo de archivos** se va a crear la base de datos.
- **PRIMARY:** Especifica los **archivos en el grupo de archivos principal**. El grupo de archivos principal contiene todas las tablas del sistema de bases de datos.
  - ✓ **Una base de datos sólo puede tener un archivo principal.** La extensión recomendada para un archivo principal es **.mdf**.
  - ✓ Si **no se especifica PRIMARY**, el **primer archivo enumerado** en la instrucción CREATE DATABASE se convierte en el **archivo principal**.
- **<filespec>.** Indica los **archivos** de los que consta el grupo de archivos
- **<grupoArchivos>.** Permite definir **nuevos grupos de archivos**.
- **LOG ON:** Especifica que **los archivos utilizados para almacenar el registro** de la base de datos (archivos de registro) se han **definido explícitamente**. La palabra clave va seguida de una lista delimitada por comas de elementos <filespec> que definen los archivos de registro.
  - ✓ **Si no se especifica LOG ON**, se **crea automáticamente un único archivo de registro** con un nombre generado por el sistema y un tamaño que es el **25 por ciento de la suma de los tamaños de todos los archivos de datos** de la base de datos.
- **COLLATE nombre\_intercalación:** Especifica **la intercalación predeterminada de la base de datos**. El nombre de la intercalación puede ser un nombre de intercalación de Windows o un nombre de intercalación de SQL. Si no se especifica, se asigna a la base de datos la intercalación predeterminada de la instancia de SQL Server.

*Nota: Las intercalaciones **especifican las reglas que determinan la forma en que las cadenas de caracteres se ordenan y comparan** en función de las normas de cada idioma y configuración regional. Por ejemplo, en una cláusula ORDER BY, un angloparlante esperaría que la cadena de caracteres 'Chiapas' se mostrara antes que 'Colima' en orden ascendente. Pero un hispanoparlante de México esperaría que las palabras que empiezan con 'Ch' se mostraran al final de una lista de palabras que empiezan por 'C'. Las intercalaciones establecen estos tipos de reglas para la ordenación y*

comparación. La intercalación Latin\_1 General ordenará 'Chiapas' delante de 'Colima' en una cláusula ORDER BY ASC, mientras que la intercalación Traditional\_Spanish ordenará 'Chiapas' después de 'Colima'.

**<filespec>**: Controla las propiedades de archivo.

- **NAME**: Especifica el **nombre lógico** del archivo
  - ✓ **nombreArchivoLógico** es el **nombre utilizado para hacer referencia al archivo en las instrucciones Transact-SQL** que se ejecuten después de que se haya creado la base de datos. Debe ser **único en la base de datos** y debe seguir las reglas de los identificadores.
- **FILENAME**: Especifica el **nombre de archivo (físico) del sistema operativo**.
  - ✓ **nombreArchivoFísico**: Es la **ruta de acceso y el nombre de archivo** que el sistema operativo utiliza cuando se crea el archivo. La ruta tiene que estar creada antes.
  - ✓ **rutaFilestream**: Para un grupo de archivos FILESTREAM, FILENAME hace referencia a la ruta de acceso donde se almacenarán los datos **FILESTREAM**.  
*Nota: Especifica que el grupo de archivos almacena objetos binarios grandes (BLOB) de FILESTREAM en archivos separados, en lugar de almacenarlos en los archivos de datos junto con los otros archivos*
- **SIZE**: especifica **el tamaño inicial (mínimo) de cada archivo**. El archivo **puede crecer**, pero **no reducirá menos del tamaño mínimo especificado**. Es un **valor entero**.
  - ✓ El **tamaño especificado para el archivo principal** debe tener al **menos el tamaño del archivo principal de la base de datos model**.
  - ✓ Se pueden utilizar los sufijos kilobyte (KB), megabyte (MB), gigabyte (GB) o terabyte (TB). El valor predeterminado es MB. El **valor mínimo que se puede asignar a un archivo** es de **512 KB**.
  - ✓ **Cuando no se proporciona size**:
    - ✓ Para el **archivo principal**: Se utiliza el **tamaño del archivo principal** de la base de datos **model**.
    - ✓ Para un **archivo de datos secundario o un archivo de registro** = tamaño es de **1 MB** (versiones anteriores al 2016) y **8 MB** (a partir de la versión 2016).
  - ✓ Si **no se especifica ningún archivo de registro**:  
 Por defecto, se crea un archivo de registro que tiene un tamaño del **25 por 100 del tamaño total de los archivos de datos** o que tiene un tamaño de 512 KB, escogiendo el que sea más grande.
- **MAXSIZE**: Especifica el **tamaño máximo que un archivo puede alcanzar**. Se pueden utilizar los sufijos KB, MB, GB y TB. El valor predeterminado es MB. Es un **número entero**.
  - ✓ **Si no se especifica ningún tamaño máximo**, el tamaño puede aumentar hasta llenar el disco (**UNLIMITED**)
  - ✓ **si se especifica un crecimiento ilimitado** para un archivo de **registro**, su tamaño máximo será de **2 TB**, y para un archivo de datos será de **16 TB**.
- **FILEGROWTH**: Especifica el **incremento de crecimiento de un archivo**.
  - ✓ Cuando se necesita, SQL Server **aumenta el tamaño del archivo en la proporción especificada** en la opción FILEGROWTH.
  - ✓ Un valor de **0 indica que no aumentará**, es decir, el **crecimiento automático está desactivado** y **no se permite más espacio**.
  - ✓ El valor se puede especificar en MB, KB, GB, TB o como porcentaje (%). Valor predeterminado es en MB. Cuando se especifica %, el incremento de crecimiento es el porcentaje especificado **del tamaño del archivo en el momento en que tiene lugar el incremento**. El tamaño especificado se redondea al múltiplo de 64 KB más cercano y el valor mínimo es 64 KB.
  - ✓ **Si no se especifica FILEGROWTH**, el valor predeterminado es:
    - (A partir de la versión 2005) **1 MB** para **los archivos de datos** y el **10%** para **los archivos de registro**, y el **valor mínimo** es 64 KB.
    - (A partir de la versión 2016) **64 MB** para **los archivos de datos** y **64 MB** para **los archivos de registro**, y el **valor mínimo** es 64 KB.
  - ✓ El **valor mínimo** que se puede asignar es de **64 KB** y **el tamaño especificado es redondeado automáticamente en múltiplos de 64 KB**.

Ejemplo 1:

El siguiente ejemplo crea una base de datos llamada Ejemplo, que tiene un **archivo principal de datos de 10 MB** y un **archivo de registro de 3 MB**. El archivo de datos principal puede crecer hasta un máximo de 15 MB en incrementos del 20 por 100. En otras palabras, la primera que vez necesite más espacio, su tamaño se incrementará en 2 MB. El archivo de registro tendrá un tamaño máximo de 5 MB en incrementos de 1 MB.

```

CREATE DATABASE Ejemplo
ON
PRIMARY
( NAME=Ejemplo_data,
FILENAME= 'C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQL\DATA\Ejemplo.mdf',
SIZE=10MB,
MAXSIZE=15MB,
FILEGROWTH=20%
)
LOG ON
(
NAME=Ejemplo_log,
FILENAME= 'C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\
MSSQL\DATA\Ejemplo.ldf',
SIZE=3MB,
MAXSIZE=5MB,
FILEGROWTH=1MB )

```

*Ejemplo2:* En este ejemplo se crea la base de datos *Ejemplo* y el archivo principal y de registro de transacciones correspondientes con las opciones por defecto. Debido a que la instrucción no tiene elementos <filespec>, el archivo principal de la base de datos tiene el tamaño del archivo principal de la base de datos model. El registro de transacciones se establece en el mayor de estos valores: 512 KB o el 25% del tamaño del archivo de datos principal. Como no se ha especificado MAXSIZE, los archivos pueden crecer hasta llenar todo el espacio disponible en el disco. El resultado es: *El tamaño se expresa en páginas de 8 KB.*

```

--Se comprueba que no existe la base de datos que vamos a crear. Si existe se borra
IF DB_ID('Ejemplo') IS NOT NULL
DROP DATABASE Ejemplo;
GO
--Se crea la base de datos ejemplo con las opciones predeterminadas
CREATE DATABASE Ejemplo;
GO
-- Se comprueba el nombre lógico del archivo de la base de datos, el nombre físico y el tamaño en páginas de 8K creado anteriormente y el de la base de datos model
SELECT name,physical_name,size FROM sys.master_files
WHERE name LIKE 'Ejemplo%'or name='modeldev';
GO

```

*Ejemplo 3:*

```

IF DB_ID('VENTAS') IS NOT NULL
DROP DATABASE VENTAS;
GO
CREATE DATABASE VENTAS
ON PRIMARY
( NAME='ventas_dat',
FILENAME='C:\ventas\ventasdat.mdf',
SIZE=10,
MAXSIZE=50,
FILEGROWTH=15% ),
( NAME='ventas1_dat',
FILENAME='C:\ventas\ventasdat1.ndf',
SIZE=10,
MAXSIZE=50,
FILEGROWTH=15%),
FILEGROUP Grupo1Ventas
( NAME='Ventas2_dat',
FILENAME='C:\ventas\ventasdat2.ndf',
SIZE=10,
MAXSIZE=50,
FILEGROWTH=5 ),
( NAME='ventas3_dat',
FILENAME='C:\ventas\ventasdat3.ndf',
SIZE=10,
MAXSIZE=50,
FILEGROWTH=5 ),
FILEGROUP Grupo2Ventas
( NAME='ventas4_dat',
FILENAME='C:\ventas\ventasdat4.ndf',
SIZE=10,
MAXSIZE=50,
FILEGROWTH=5 ),

```

	name	db_size	owner	dbid	created	status
1	VENTAS	65.00 MB	WSERVER2008\Administrador	17	Nov 23 2011	Status=ONLINE, Updateability=READ_V

	name	fileid	filename	filegroup	size	maxsize	growth	usage
1	ventas_dat	1	C:\ventas\ventasdat.mdf	PRIMARY	10240 KB	51200 KB	15%	data only
2	Ventas_log	2	C:\ventas\ventaslog.ldf	NULL	5120 KB	25600 KB	5120 KB	log only
3	ventas1_dat	3	C:\ventas\ventasdat1.ndf	PRIMARY	10240 KB	51200 KB	15%	data only
4	Ventas2_dat	4	C:\ventas\ventasdat2.ndf	Grupo1Ventas	10240 KB	51200 KB	5120 KB	data only
5	ventas3_dat	5	C:\ventas\ventasdat3.ndf	Grupo1Ventas	10240 KB	51200 KB	5120 KB	data only
6	ventas4_dat	6	C:\ventas\ventasdat4.ndf	Grupo2Ventas	10240 KB	51200 KB	5120 KB	data only
7	ventas5_dat	7	C:\ventas\ventasdat5.ndf	Grupo2Ventas	10240 KB	51200 KB	5120 KB	data only

```

FILEGROWTH = 5 ),
( NAME = 'ventas5_dat',
  FILENAME = 'C:\ventas\ventasdat5.ndf',
  SIZE = 10,
  MAXSIZE = 50,
  FILEGROWTH = 5 )
LOG ON
( NAME = 'Ventas_log',
  FILENAME = 'C:\ventas\ventaslog.ldf',
  SIZE = 5MB,
  MAXSIZE = 25MB,
  FILEGROWTH = 5MB )

```

**Ejemplo 4:** Creación de un grupo que contiene un archivo **FILESTREAM**

```

IF DB_ID ('Ejemplo4') IS NOT NULL
DROP DATABASE Ejemplo4;
GO
CREATE DATABASE EJEMPLO4
ON PRIMARY
( NAME = 'EJEMPLO4',
  FILENAME='C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQL\DATA\EJEMPLO4.mdf' ,
  SIZE = 3072KB ,
  MAXSIZE = UNLIMITED,
  FILEGROWTH = 1024KB ),
FILEGROUP fileSTREAM1 CONTAINS FILESTREAM DEFAULT
( NAME = 'EJ4FILESTREAM',
  FILENAME='C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQL\DATA\EJ4FILESTREAM' )
LOG ON
( NAME = 'EJEMPLO3_log',
  FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQL\DATA\EJEMPLO3_log.ldf' ,
  SIZE = 1024KB ,
  MAXSIZE = 2048GB ,
  FILEGROWTH = 10%)

```

## 4.2 Visualizar información de las bases de datos y los grupos de archivos

La siguiente tabla muestra los **procedimientos almacenados y vistas de catálogo** de sistema que muestran información acerca de las bases de datos y grupos de archivos.

Procedimiento almacenado de sistema	Descripción
<b>sp_helpfile</b> ['archivo_lógico']	Devuelve los nombres y atributos físicos de todos los archivos o un archivo especificado asociado con la base de datos actual.
<b>sp_helpfilegroup</b> ['grupodearchivo']	Devuelve los nombres y atributos de grupos de archivos asociados con la base de datos actual. Si se especifica un nombre de grupo de archivos sp_helpfilegroup devuelve una lista de los archivos del grupo.
<b>sp_helpdb</b> ['nombre_base_datos']	Informa sobre todas las bases de datos del servidor (muestra el nombre de la base de datos, tamaño, dueño, ID, fecha de creación y opciones) o sobre una base de datos. La misma información que el anterior pero además muestra detalles sobre cada archivo de datos y registro.
<b>sp_spaceused</b> [nombre_objeto]	Resume el espacio de almacenamiento usado por la base de datos actual o por una tabla de la base de datos actual.
<b>sp_dboption</b> ['nombre_base_datos']	Muestra o cambia las opciones de las bases de datos o una base de datos especificadas.  Nota: Esta característica se quitará en la versión siguiente de Microsoft SQL Server. Se utilizará ALTER DATABASE en su lugar
<b>Vistas de catálogo</b>	<b>Descripción</b>
<b>sys.master_files</b>	Contiene una fila por archivo de una base de datos almacenada en la base de datos master.

<code>sys.database_files</code>	Contiene una fila por cada archivo de una base de datos como se almacena en la propia base de datos
<code>sys.databases</code>	Muestra información de todas las bases de datos y podemos ver las diferentes opciones que podemos configurar
<code>sys.filegroups</code>	Contiene una fila por cada espacio de datos que es un grupo de archivos.

El tamaño informado por `sp_helpdb` y `sp_spaceused` es el espacio total actual de la base de datos, incluyendo el tamaño de los archivos de registro. Para determinar el tamaño de los datos en la base de datos, reste el espacio de los archivos de registro del tamaño total de la base de datos.

## 4.3 Eliminar una base de datos

Se puede eliminar una base de datos definida por el usuario cuando ya no sea necesaria o si se mueve a otra base de datos o servidor.

Cuando se elimina una base de datos, se eliminan del disco del servidor los archivos y sus datos. Una vez eliminada una base de datos, se quita de forma permanente y sólo podrá recuperarse mediante una copia de seguridad previa.

Las bases de datos del sistema no se pueden eliminar.

### 4.3.1 Consideraciones para eliminar una base de datos

Antes de eliminar una base de datos, hay que considerar:

- Una base de datos se puede quitar sea cual sea su estado (sin conexión, sólo lectura, sospechosa, etc.). Para ver el estado actual de una base de datos, se utiliza la vista de catálogo `sys.databases`.
- Para quitar una base de datos del servidor actual sin eliminar los archivos del sistema de archivos, se usa `sp_detach_db`.
- Una **base de datos que se ha quitado sólo puede volver a crearse si se restaura una copia de seguridad**. Para eliminar una base de datos, el usuario debe tener, como mínimo, el permiso **CONTROL** en la base de datos.
- **NO se puede eliminar:**
  - Una base de datos que esté en proceso de ser restaurada ni una base de datos que este abierta en modo lectura o escritura por algún usuario.
  - SQL Server no permite eliminar la base de datos master, model ni tempdb.

Sintaxis del comando **DROP DATABASE**:

```
DROP DATABASE nombre_database [... n]
```

Ejemplo 1: En el ejemplo siguiente se quita la base de datos ventas.

```
DROP DATABASE ventas;
```

Ejemplo 2: Quitar varias bases de datos

```
DROP DATABASE ventas,compras;
```

## 4.4 Estados de base de datos

En SQL Server, una base de datos tiene dos estados básicos: disponibilidad completa (**ONLINE**) o no disponibilidad completa (**OFFLINE**).

- ✓ Para poner una base de datos **fuera de línea**:

```
ALTER DATABASE < nombreBasedeDatos >
SET OFFLINE
GO
```

- ✓ Para poner una base de datos **en línea**:

```
ALTER DATABASE < nombreBasedeDatos >
SET ONLINE GO
```

- ✓ Para **comprobar el estado actual de una base de datos**, seleccione la columna `state_desc` de la vista de catálogo `sys.databases`.



En la siguiente tabla se define los estados de la base de datos.

Estado	Definición
<b>ONLINE (NORMAL)</b>	La base de datos <b>está disponible</b> para su acceso.
<b>OFFLINE</b>	La base de datos <b>no está disponible</b> . Una base de datos pasa a estar sin conexión por la acción explícita del usuario y permanece sin conexión hasta que el usuario toma otra acción. Por ejemplo, la base de datos puede desconectarse para mover un archivo a un nuevo disco. La base de datos se vuelve a poner en línea una vez completado el traslado.
<b>RESTORING</b>	Uno o varios archivos del grupo de archivos principal <b>se están restaurando</b> , o uno o varios archivos secundarios se están restaurando sin conexión. <b>La base de datos no está disponible</b> .
<b>RECOVERING</b>	<b>Se está recuperando la base de datos</b> . El proceso de recuperación es un estado transitorio, la base de datos se pone automáticamente en línea si la recuperación tiene éxito. Si la recuperación no tiene éxito, la base de datos pasa a ser sospechosa. <b>La base de datos no está disponible</b> .
<b>RECOVERY PENDING</b>	SQL Server <b>ha encontrado un error relacionado con un recurso durante la recuperación</b> . La base de datos no está dañada pero pueden faltar archivos o bien limitaciones de recursos del sistema pueden estar impidiendo que se inicie. <b>La base de datos no está disponible</b> . Se necesita una acción adicional por parte del usuario para resolver el error y permitir que se complete el proceso de recuperación.
<b>SUSPECT</b>	Como mínimo un <b>grupo de archivos principal es sospechoso</b> y puede estar <b>dañado</b> . La base de datos no se puede recuperar durante el inicio de SQL Server. <b>La base de datos no está disponible</b> . Se requiere una acción adicional por parte del usuario para resolver el problema.
<b>EMERGENCY</b>	El usuario ha cambiado la base de datos y ha establecido el estado en EMERGENCY. La base de datos <b>está en modo de usuario único</b> y se puede <b>reparar o restaurar</b> . La base de datos está marcada como READ_ONLY, el registro está deshabilitado y <b>el acceso está limitado a miembros de la función fija de servidor sysadmin</b> . EMERGENCY se utiliza principalmente para solucionar problemas. Por ejemplo, una base de datos marcada como sospechosa se puede establecer en el estado EMERGENCY. Esto puede permitir al administrador del sistema acceso de sólo lectura a la base de datos.

## 4.5 Modificación de un base de datos

Se puede [cambiar la configuración de una base de datos](#), empleando **SQL Server Management Studio** o con la instrucción **ALTER DATABASE**.

Podemos realizar las tareas de configuración siguiente:

- [Agregar o quita archivos y grupos de archivos](#) en una base de datos
- [Modificar propiedades de archivos de datos y de registro](#), como aumentar su tamaño del archivo, cambiar el tamaño máximo o establecer reglas de crecimiento una base de datos o los archivos y grupos de archivos asociados a la base de datos.
- [Cambiar las propiedades de una grupo de archivo existente](#), como designar si el grupo de archivos es de sólo lectura, lectura-escritura y cuál grupo es el predeterminado.

*Sintaxis:*

```
ALTER DATABASE <nombreBaseDeDatos>
{
    <añadir_o_modificar_archivos>
    | <añadir_o_modificar_gruposArchivos>
    | MODIFY NAME = nuevo_nombre
    | COLLATE nombreintercalacion
};
<añadir_o_modificar_archivos>
{ ADD FILE <filespec> [ ,...n ]
  [ TO FILEGROUP { nombreGrupoArchivo } ]
  | ADD LOG FILE <filespec> [ ,...n ]
  | REMOVE FILE nombreLogico
  | MODIFY FILE <filespec>
}
<filespec>::=
( NAME = nombreLogico
  [ , NEWNAME = nuevo_nombreLogico ]
  [ , FILENAME = nombreFisico ]
  [ , SIZE = tamaño [ KB | MB | GB | TB ] ]
```

```
[ , MAXSIZE = { max_size [ KB | MB | GB | TB ] | UNLIMITED } ]
[ , FILEGROWTH = incremento [ KB | MB | GB | TB | % ] ]
[ , OFFLINE ]
)
| <añadir_o_modificar_gruposArchivos>{
| ADD FILEGROUP nombre_GrupoArchivo
| [ CONTAINS FILESTREAM ]
| REMOVE FILEGROUP nombre_GrupoArchivo
| MODIFY FILEGROUP nombre_GrupoArchivo
| { <cambiar_propiedadesdeActualizacion>
| DEFAULT
| NAME = nuevo_nombreGrupoArchivo
| }
| }
| <cambiar_propiedadesdeActualizacion> {
| READONLY | READ_ONLY | READ_WRITE
| }
}
```

- **nombreBasedeDatos** : Es el nombre de la base de datos que se va a modificar.
- **MODIFY NAME = nuevo\_nombre** : Cambia el nombre de la base de datos por el nombre especificado como nuevo\_nombre.
- **COLLATE collation\_name**: Especifica la **intercalación** para la base de datos.

**<añadir\_o\_modificar\_archivos>**:= Especifica el archivo que se va a agregar, quitar o modificar.

- **ADD FILE <filespec>**: Agrega un archivo a la base de datos.
- **TO FILEGROUP { nombreGrupoArchivo }**: Especifica el grupo de archivos al que se agrega el archivo especificado. Para mostrar los grupos de archivos actuales y qué grupo de archivos es el predeterminado, se utiliza la vista de catálogo **sys.filegroups**.
- **ADD LOG FILE <filespec>**: Agrega un archivo de registro a la base de datos especificada.
- **REMOVE FILE nombreLogico**. Quita la descripción del archivo lógico de una instancia de SQL Server y elimina el archivo físico. El archivo no se puede quitar a menos que esté vacío.  
     **logical\_file\_name** :Es el nombre lógico utilizado en SQL Server cuando se hace referencia al archivo.
- **MODIFY FILE <filespec>** : Especifica el archivo que se debe modificar.
  - ✓ Sólo se puede cambiar una propiedad cada vez.
  - ✓ **NAME** se debe especificar siempre en para identificar el archivo que se va a modificar.
  - ✓ Si se especifica **SIZE**, el nuevo tamaño debe ser mayor que el tamaño actual del archivo.

Para modificar el nombre lógico de un archivo de datos o de un archivo de registro, especifique el nombre del archivo lógico que se va a cambiar en la cláusula **NAME** y especifique el nombre lógico nuevo para el archivo en la cláusula **NEWNAME**. Por ejemplo:

```
MODIFY FILE ( NAME = nombreLogico, NEWNAME = nuevo_nombreLogico )
```

Para mover un archivo de datos o un archivo de registro a otra ubicación, especifique el nombre del archivo lógico actual en la cláusula **NAME** y especifique la ruta y el nombre del archivo del sistema operativo nuevos en la cláusula **FILENAME**. Por ejemplo:

```
MODIFY FILE ( NAME = nombreLogico, FILENAME = ' NuevoPath/ nuevo_nombreFisico ' )
```

**<filespec>::=**Controla las propiedades del archivo.

- **NAME nombreLogico**: Especifica el nombre lógico del archivo.
- **NEWNAME Nuevo\_nombreLogico**: Especifica un nombre lógico nuevo para el archivo.
- **FILENAME ' nombreFisico '**:Especifica un nombre de archivo (físico) del sistema operativo.
- **SIZE tamaño**: Especifica el tamaño del archivo.
- **MAXSIZE { maximotamaño| UNLIMITED }**:Especifica el tamaño máximo que puede alcanzar el archivo.
- **FILEGROWTH incremento**. Especifica el aumento automático del archivo. El valor **FILEGROWTH** de un archivo no puede superar el valor **MAXSIZE**.
- **OFFLINE**: Establece el archivo sin conexión e impide el acceso a todos los objetos del grupo de archivos.

*Advertencia: Esta opción sólo se debe utilizar si el archivo está dañado y se puede restaurar. Un archivo establecido en OFFLINE sólo se puede establecer con conexión mediante la restauración del archivo a partir de una copia de seguridad.*

**<add\_or\_modify\_filegroups>::=** Agrega, modifica o quita un grupo de archivos de la base de datos.

- **ADD FILEGROUP** nombreGrupoArchivo : Agrega un grupo de archivos a la base de datos.
- **CONTAINS FILESTREAM**: Especifica que el grupo de archivos almacena objetos binarios grandes (BLOB) de FILESTREAM en el sistema de archivos.
- **REMOVE FILEGROUP** nombreGrupoArchivo: Quita un grupo de archivos de la base de datos. El grupo de archivos no se puede quitar a menos que esté vacío.
- **MODIFY FILEGROUP** nombreGrupoArchivo {<Cambiar\_propiedadesdeActualizacion> | **DEFAULT** | **NAME** = Nuevo\_nombreGrupoArchivo}:  
Modifica el grupo de archivos al establecer el estado en READ\_ONLY o READ\_WRITE, o hace que el grupo de archivos se convierta en predeterminado para la base de datos o que se cambie el nombre del grupo de archivos.
  - ✓ **DEFAULT**: Cambia el grupo de archivos predeterminado de la base de datos a filegroup\_name. Sólo un grupo de archivos de la base de datos puede ser el grupo de archivos predeterminado.
  - ✓ **NAME** = new\_filegroup\_name: Cambia el nombre del grupo de archivos a new\_filegroup\_name.

**<filegroup\_updatability\_option>**: Establece la propiedad de sólo lectura o sólo lectura y escritura para el grupo de archivos.

- **READ\_ONLY** : Especifica que el grupo de archivos es de sólo lectura. No se permite la actualización de los objetos del mismo. El grupo de archivos principal no puede ser de sólo lectura. Para **cambiar este estado**, debe tener acceso exclusivo a la base de datos (SINGLE\_USER).
- **READ\_WRITE** : Pueden realizarse actualizaciones en los objetos del grupo de archivos. Para cambiar este estado, debe tener acceso exclusivo a la base de datos (SINGLE\_USER).

Ejemplos:

### Cambiar el nombre de una base de datos

Antes de cambiar el nombre de una base de datos, debe asegurarse de que ningún usuario la utiliza y de que está configurada con el modo de usuario único.

```
ALTER DATABASE database_name
SET Single_user
--Para cambiar el nombre
ALTER DATABASE Ejemplo1
MODIFY NAME = NuevaBasededatos
```

### Aumentar el tamaño de un archivo existente:

La opción SIZE configura el tamaño mínimo de un archivo. El archivo **puede crecer, pero no puede reducir por debajo del tamaño designado**.

No se puede reducir el tamaño del archivo usando la instrucción ALTER DATABASE. Para reducir el tamaño mínimo de un archivo, se utiliza la instrucción **DBCC SHRINKFILE**.

Se aumenta el valor de la columna Tamaño inicial (SIZE) correspondiente al archivo. Debe aumentar el tamaño de la base de datos en 1 megabyte como mínimo.

Si se aumenta la configuración del **tamaño (SIZE)** por encima del tamaño máximo actual sin incrementar la instrucción MAXSIZE, el tamaño máximo será igualado al nuevo tamaño.

Ejemplo: Se crea una base de datos con tamaño inicial 3 y tamaño de crecimiento máximo 6

```
CREATE DATABASE prueba
ON PRIMARY
(NAME = BD_prueba,
FILENAME = 'C:\Archivos de programa\Microsoft SQL Server\
MSSQL.1\MSSQL\Data\ BD_prueba_Data.mdf',
SIZE = 3,
MAXSIZE = 6)
```

Ahora modificamos el tamaño a 8 por encima del tamaño máximo, en este caso, el tamaño MAXSIZE será igual a 8.

```
ALTER DATABASE prueba
MODIFY FILE
(NAME= BD_prueba,
SIZE= 8)
```

name	fileid	filename	filegroup	size	maxsize	growth	usage
BD_prueba	1	C:\Archivos de programa\Microsoft SQL Server\MSS...	PRIMARY	8192 KB	8192 KB	1024 KB	data
prueba_log	2	C:\Archivos de programa\Microsoft SQL Server\MSS...	NULL	768 KB	2147483648 KB	10%	log

## Cómo añadir archivos secundarios o archivos de registro

Otra manera de aumentar el tamaño de la base de datos es crear archivos de datos secundarios.

Si se necesita agregar archivos debido a espacio insuficiente, una manera es utilizar archivos secundarios de datos o archivos de registro para utilizar discos físicos separados cuando no se emplee la utilidad de discos con bandas de los sistemas RAID.

Ejemplo: En el siguiente ejemplo se agrega un archivo de datos de 5 MB a la base de datos Ventas, el tamaño máximo que puede alcanzar el archivo es 100 MB y crecimiento automático de 5 MB

```
ALTER DATABASE Ventas
ADD FILE
( NAME = EjADDarchivo,
  FILENAME = 'C:\Archivos de programa\Microsoft SQL Server\MSSQL.1\MSSQL\EjADDarchivo.ndf',
  SIZE = 5MB,
  MAXSIZE = 100MB,
  FILEGROWTH = 5MB
)
```

## 4.6 Establecer las opciones de base de datos:

Después de crear una base de datos, se puede ver información de la base de datos y cambiar varias de sus opciones. Las opciones de bases de datos determinan las características de una base de datos. Para cada base de datos, es posible configurar varias opciones de base de datos que determinen sus características.

Si hacemos: **SELECT \* FROM sys.databases**

Nos muestra información de todas las bases de datos y podemos ver las diferentes opciones que podemos configurar:

**OPCIONES**

	name	database...	so...	owner_sid	create_date	compatibility_l...	collation_name	user_acce...	user_access_d...	is_read_o...	is_auto_close
1	master	1	N...	0x01	2003-04-08 09:13:36.390	100	Modern_Spanish_CI_AS	0	MULTI_USER	0	0
2	tempdb	2	N...	0x01	2020-01-22 21:17:54.077	100	Modern_Spanish_CI_AS	0	MULTI_USER	0	0
3	model	3	N...	0x01	2003-04-08 09:13:36.390	100	Modern_Spanish_CI_AS	0	MULTI_USER	0	0
4	msdb	4	N...	0x01	2010-04-02 17:35:08.970	100	Modern_Spanish_CI_AS	0	MULTI_USER	0	0
5	Empresa_Case	5	N...	0x0105000000000000...	2013-12-17 23:29:51.980	80	Modern_Spanish_CI_AS	0	MULTI_USER	0	0
6	BDBiblioteca	6	N...	0x0105000000000000...	2015-01-20 23:00:07.410	80	SQL_Latin1_General_C...	0	MULTI_USER	0	0
7	BDEMPRESA	7	N...	0x0105000000000000...	2018-12-11 11:35:36.483	100	Modern_Spanish_CI_AS	0	MULTI_USER	0	0
8	EMPRESAHR	8	N...	0x0105000000000000...	2019-02-07 15:51:05.140	80	Modern_Spanish_CI_AS	0	MULTI_USER	0	0

Las opciones son únicas para cada base de datos y no afectan a otras bases de datos.

Cuando crea una base de datos, estas opciones se establecen en sus valores predeterminados heredados de la base de datos model. Estos valores se pueden modificar mediante el uso de la cláusula **SET** de la instrucción **ALTER DATABASE**, o empleando **SQL Server Management Studio**

*Nota: Sólo el **administrador del sistema** o los **propietarios de la base de datos**, miembros de las funciones fijas de los servidores **sysadmin** y **dbcreator** pueden modificar estas opciones.*

### 4.6.1 Establecimiento de las opciones utilizando ALTER DATABASE.

Solo vamos a ver las siguientes opciones

```
ALTER DATABASE <nombreBasedeDatos>
{ SET
  <opciones-estado-bd>
  | <opciones-acceso-usuario>
  | <opciones-actualizacion-bd>
}
<opciones-estado-bd>
{ ONLINE | OFFLINE | EMERGENCY }
<opciones-acceso-usuario>
{ SINGLE_USER | RESTRICTED_USER | MULTI_USER }
<opciones-actualizacion-bd>
{ READ_ONLY | READ_WRITE }
```

- **Estado de base de datos:** Permite cambiar el estado actual de la base de datos. Puede estar:
  - **OFFLINE:** La base de datos no está disponible. Una base de datos pasa a estar sin conexión. Por ejemplo, la base de datos puede dejarse sin conexión para mover un archivo a un nuevo disco.
  - **ONLINE:** La base de datos está abierta y disponible para su uso.
  - **EMERGENCY:** La base de datos está en modo de usuario único y READ\_ONLY, el registro está deshabilitado y el acceso está limitado a los miembros de la **función fija del servidor sysadmin**. Se utiliza principalmente para la solución de problemas.
- **Base de datos de sólo lectura (READ\_ONLY):** Especifica si la base de datos es de **sólo lectura**. Los valores posibles son True o False.
  - **Con el valor True, los usuarios sólo pueden leer los datos de la base de datos.** Los usuarios no pueden modificar los datos ni objetos de la base de datos; sin embargo, la base de datos se puede eliminar con la instrucción DROP DATABASE.
- **Restringir acceso ({ SINGLE\_USER | RESTRICTED\_USER | MULTI\_USER }):** Especifica los usuarios que pueden tener acceso a la base de datos. Los valores posibles son:
  - **MULTIPLE:** El estado normal de una base de datos de producción, permite **que varios usuarios tengan acceso a la base de datos a la vez**.
  - **SINGLE:** Se utiliza en acciones de mantenimiento, **sólo un usuario puede tener acceso a la base de datos**. De debería establecer esta opción antes de restaurar o cambiar el nombre de una base de datos, puesto que así nos aseguramos de que nadie intente usar la base de datos durante estas actividades.
  - **RESTRICTED:** Sólo los miembros de las funciones **db\_owner, dbcreator o sysadmin pueden utilizar la base de datos**. Las personas que están usando ya la base de datos no se pueden desconectar, pero en cuanto salen, no pueden volver a entrar.

Esta opción se suele utilizar durante el desarrollo inicial de la base de datos o cuando se necesita cambiar la estructura de algún objeto de la base de datos como añadir una nueva columna a una tabla.

Ejemplo:

Cambiar la base de datos empresa\_clase para que un único usuario tenga acceso.

```
ALTER DATABASE EMPRESA_CLASE
SET SINGLE_USER
```

Comprobamos:

```
SELECT * FROM sys.databases
```

name	database...	source_database...	owner_sid	create_date	compatibility_le...	collation_name	user_acce...	user_access_d...	i
master	1	NULL	0x01	2003-04-08 09:13:36.390	100	Modern_Spanish_CI_AS	0	MULTI_USER	i
tempdb	2	NULL	0x01	2020-01-22 21:17:54.077	100	Modern_Spanish_CI_AS	0	MULTI_USER	i
model	3	NULL	0x01	2003-04-08 09:13:36.390	100	Modern_Spanish_CI_AS	0	MULTI_USER	i
msdb	4	NULL	0x01	2010-04-02 17:35:08.970	100	Modern_Spanish_CI_AS	0	MULTI_USER	i
Empresa_Clase	5	NULL	0x010500000...	2013-12-17 23:29:51.980	80	Modern_Spanish_CI_AS	1	SINGLE_USER	i
BDBiblioteca	6	NULL	0x010500000...	2015-01-20 23:00:07.410	80	SQL_Latin1_General_CP1_CI_AS	0	MULTI_USER	i
BDEMPRESA	7	NULL	0x010500000...	2018-12-11 11:35:36.483	100	Modern_Spanish_CI_AS	0	MULTI_USER	i
EMPRESAHR	8	NULL	0x010500000...	2019-02-07 15:51:05.140	80	Modern_Spanish_CI_AS	0	MULTI_USER	i
...	...	...	...	...	...	...	...	...	...

O también



EXEC sp\_helpdb 'Empresa\_clase'

Empresa_Clase	5.69 MB	win7-clases\usuario	5	Dic 17 2013	Status=ONLINE, Updateability=READ_WRITE, UserAccess=SINGLE_USER, Recovery=FU...	80
---------------	---------	---------------------	---	-------------	---	----

name	fileid	filename	filegroup	size	maxsize	growth	usage
EMPRESA_Data	1	C:\Program Files\Microsoft SQL Server\MSSQL10_50...	PRIMARY	1984 KB	Unlimited	10%	data only
EMPRESA_Log	2	C:\Program Files\Microsoft SQL Server\MSSQL10_50...	NULL	3840 KB	Unlimited	10%	log only

Y en el explorador de objetos aparecerá marcada como usuario único:

BDPINACOTECAS

BDTURISMO

Empresa\_Clase (Usuario único)

EMPRESAHB

EMPRESAHBCONSULTAS