The page features a decorative graphic on the right side consisting of three blue circles of varying sizes, each with a lighter blue ring around its center. Two thin blue lines intersect at the top left, forming a large 'V' shape that frames the circles. The largest circle is at the top, a smaller one is in the middle, and the largest of all is at the bottom right, partially cut off by the edge of the page.

EL ENTORNO GRÁFICO SSMS

SQL SERVER 2005

Manual de Referencia para usuarios

Salomón Ccance
CCANCE WEBSITE

MICROSOFT SQL SERVER 2005

SQL Server 2005 es un sistema gestor de bases de datos relacionales de Microsoft Corporation orientado a sistemas medianos y grandes aunque también puede rodar en ordenadores personales. SQL Server Management Studio (SSMS) es la herramienta de SQL Server que permite definir y gestionar todas las bases de datos almacenadas en el servidor SQL Server 2005.

En este tema veremos cómo utilizar el SQL Server Management Studio para manejar las bases de datos del servidor y organizaremos el texto en los siguientes puntos:

- Instalar SQL Server 2005.
- Entrada al SQL Server Management Studio
- Las bases de datos: Estructura interna, crear, adjuntar, conectar y desconectar.
- Las tablas: crear tablas, definir las, modificar su contenido, etc.
- Relacionar tablas
- Las Consultas
- Las Vistas

Un sistema gestor de bases de datos por definición debe cumplir los siguientes objetivos:

- **Independencia de los datos.**

Recordando uno de los problemas que plantean los ficheros convencionales. La independencia de los datos consiste en hacer que los programas no sean tan dependientes de la estructura de los datos.

Se han definido dos tipos de independencia:

1. La independencia **física**: consiste en poder modificar la definición interna de los datos (el esquema interno) sin que ello suponga una modificación de los programas existentes. Por ejemplo, se puede cambiar la ubicación de la base de datos, o se puede añadir un índice sobre una tabla para que las consultas se ejecuten más rápidamente, sin que eso suponga una variación en los esquemas externos y conceptual, por lo que los programas (que utilizan el esquema externo) no se verán afectados.
2. La independencia **lógica**: consiste en poder cambiar el esquema conceptual sin que ello suponga una modificación de los programas existentes.

Por ejemplo podemos añadir un nuevo dato en la tabla de clientes como la dirección de email sin que los esquemas externos se vean afectados.

- **Seguridad e integridad.**

La **seguridad** consiste en que los usuarios no puedan acceder a datos sin autorización.

Si juntamos toda la información de la empresa en un mismo sitio, el SGBD debe tener mecanismos para que cualquier usuario pueda tener acceso a únicamente la información que necesita para las tareas que tiene encomendadas.

Esta seguridad se consigue por medio de los esquemas externos, ya que el usuario sólo tiene acceso a su esquema externo que le proporciona los datos que el administrador ha considerado incluir en ese esquema. Para el usuario no habrá más datos que estos. Además, los SGBD tienen mecanismos para definir autorizaciones que pueden ser de distinto tipo: autorización de lectura, de inserción, de actualización, autorizaciones especiales para poder variar el esquema conceptual etc.

La **integridad** se refiere a que la información almacenada en la base de datos esté libre de

errores. Esto no siempre es posible ya que existen distintos tipos de errores que tienen diferentes soluciones:

* Fallos de hardware. Estos errores no los puede evitar el SGBD pero se pueden subsanar facilitando copias de seguridad y procesos de recuperación.

* Fallos del programador. Puede que aparezcan datos erróneos en la base de datos como consecuencia de errores en el programa que genera estos datos. Para evitar al máximo este tipo de errores el sistema debe ser fácil de programar, cuantos más controles realice el sistema de forma automática, menos controles habrá que incluir a nivel de programación por lo que limitaremos la probabilidad de fallo y los programas deben ser probados con juegos de ensayos bien definidos.

* Fallos del usuario final. El usuario que introduce datos en la base de datos también puede cometer errores, el sistema debe permitir controlar al máximo la información que se introduce para limitar el número de estos errores, para ello los SGBD incluyen cláusulas de validación de los datos, validaciones de diferentes tipos que veremos con más detalle más adelante.

* Fallos derivados de la concurrencia. Ya que toda la información está centralizada y los distintos usuarios acceden a ella de forma simultánea, pueden ocurrir problemas cuando dos usuarios quieren acceder al mismo dato a la vez. Por ello el SGBD debe tener establecidos mecanismos para evitar este tipo de problema, bloquear registros, abortar automáticamente transacciones etc.

- **Redundancia mínima**

La redundancia consiste en que exista algún dato repetido en varios lugares. Por ejemplo si tenemos la dirección del cliente en la factura, en la cuenta contable, en los datos generales del cliente; esto como ya vimos anteriormente nos producirá varios problemas:

- ✓ La información repetida ocupa espacio innecesario.
- ✓ La variación de un domicilio supone el variar ese domicilio en todos los lugares donde esté almacenado.
 - mayor tiempo de proceso
 - posibilidad de inconsistencia

Por todo ello hay que evitar al máximo esa redundancia, esto se consigue utilizando herramientas de diseño y obteniendo un diseño óptimo de la base de datos.

- **Facilidad de recuperación de la información**

Otro objetivo muy importante de un SGBD es el proporcionar al usuario (o programador) unas herramientas potentes de manejo de datos para que pueda de manera sencilla y rápida, obtener toda la información que desea sin que, por ello se tenga que hacer un programa complejo.

Veremos que el SQL, lenguaje empleado para recuperar información de la base de datos, es un lenguaje muy potente y cercano al lenguaje hablado, y además los SGBD incluyen entornos gráficos sencillos de utilizar.

ADMINISTRADOR DE BASE DE DATOS

El administrador es el encargado de gestionar y controlar todo el sistema con la ayuda que le proporciona el SGBD. Tiene una gran responsabilidad ya que de él depende que el sistema funcione





correctamente y como tiene el máximo nivel de privilegios, sus errores pueden ser desastrosos. Entre sus responsabilidades se incluye:

- Instalar el SGBD en el sistema informático (a veces).
- Realizar el diseño de la base de datos.
- Crear las bases de datos que se vayan a gestionar.
- Crear y mantener los esquemas de las bases de datos.
- Crear y mantener las cuentas de los usuarios de las bases de datos.
- Colaborar con el administrador del sistema en las tareas de ubicación, dimensionado y control de los archivos y espacios de disco ocupados por el SGBD.
- Establecer estándares de uso, políticas de acceso y protocolos de trabajo diario para los usuarios de las bases de datos.
- Efectuar tareas de explotación como:
 - ✓ Vigilar el trabajo diario colaborando en la resolución de las dudas de los usuarios.
 - ✓ Controlar los tiempos de acceso, tasas de uso, cargas en los servidores, anomalías, etc.
 - ✓ Llegado el caso, reorganizar las bases de datos.
 - ✓ Diseñar y efectuar el planning de copias de seguridad periódicas.
 - ✓ Restaurar la base de datos después de un incidente.
 - ✓ Estudiar las auditorías mediante el ajuste de parámetros y con ayuda de las herramientas de monitorización del sistema y de las estadísticas.

ELEMENTO DE UNA BASE DE DATOS RELACIONAL

- Los datos se organizan en **relaciones** compuestas por **tuplas** de **atributos**. Si convertimos esta definición a tablas tenemos que los datos se organizan en **tablas** compuestas por **filas** (registros) y **columnas** (campos).
- A cada tabla se le asigna un **nombre único**.
- Una tabla tiene 0 o más **filas**, y cada fila contiene la información de un determinado ‘sujeto’ de la relación.
- Las filas en un principio están desordenadas.
- La lista de los atributos dispuestos en un orden específico de izquierda a derecha y que forman la definición de una tabla se denomina **esquema de la tabla**, mientras que los valores concretos de los datos que están almacenados en la tabla se llaman **ocurrencias**.

Por ejemplo, tenemos estas dos tablas:

Piezas

Codigo	Denominación	Precio	Fabricante	Codigo segun fab
1	Taburete 3 patas	25	Fab1	T123-34
2	Mesa ovalada	1200	Fab1	M234-87
3	Taburete 3 patas	78	Fab2	T2S-0P





Fabricantes

IdFab	Nombre	Direccion
Fab1	Muebles La Madera	Null
Fab2	Maderas Asociados	Avda. del Sol, nave3
Fab5	Industrias Madereras	Pol. Ind 3 fuentes, nave 23

El esquema de la tabla *Piezas* está compuesto por las columnas (*Codigo*, *Denominación*, *Precio*, *Fabricante*, *Codigo_según_fab*).

Codigo es el código de la pieza, *Denominacion* el nombre de la pieza, *Fabricante* el código del fabricante que nos suministra la pieza y *Código_según_fab* el código que utiliza ese fabricante para identificar la pieza en su sistema de gestión.

Una ocurrencia de fila de la tabla *Piezas* sería: 1, *Taburete 3 patas*, 25, *Fab1*, T123-34.

- Todos los valores de una columna determinada tienen el **mismo tipo de datos**, y éstos están extraídos de un conjunto de valores legales llamado **dominio** de la columna. Muchas veces el dominio se corresponderá con un tipo de datos estándar del sistema. Por ejemplo en la tabla *Piezas* la columna *Codigo* está definida sobre el dominio de los enteros.
- A parte de los valores del dominio, la columna puede contener un valor especial, el **valor nulo**. El valor nulo (NULL) es importante porque representa la ausencia de valor en el campo y no es lo mismo que el valor cero "0" o la cadena vacía o espacios en blanco. De hecho es un valor tan especial que no funciona como los demás valores, por ejemplo no podemos comparar (con el operador de comparación =) un campo con el valor nulo, tenemos que utilizar un operador especial (IS NULL). Incluso se han tenido que redefinir los operadores lógicos para tener en cuenta el valor nulo.

Ej. En la tabla *Fabricantes* el campo *dirección* de la primera fila contiene el valor nulo (*null*) esto significa que este fabricante no tiene dirección (al menos conocida).

- En una tabla cada columna tiene un único nombre y éste no se puede utilizar para nombrar otra columna de la misma tabla pero sí de otra tabla.

Por ejemplo en la tabla *Piezas* no se pueden definir dos columnas llamadas *Codigo*, por eso el segundo código lo hemos llamado *Codigo_según_fab*. Pero en la tabla *Fabricantes* la columna *IdFab* se podía haber llamado *Codigo* sin problema.

- En una tabla no se admiten dos filas con los valores coincidentes en todos sus campos. Esta restricción no se suele cumplir.

Esta regla nos dice que por ejemplo en la tabla *Fabricantes* no pueden haber dos filas con los valores *Fab1*, *Muebles la Madera*, *null*.

Realmente sería información redundante, por eso la existencia de esta regla, no obstante en algunos casos muy concretos sí es necesario poder almacenar dos ocurrencias de fila idénticas, por esta razón muchos SGBD no cumplen esta regla.

- Toda tabla debe tener una **clave principal (clave primaria)**.

Una clave primaria es cualquier una columna (o combinación de columnas) que permite identificar de forma unívoca cada una de las filas de la tabla. Para que pueda cumplir su cometido, **la clave primaria no puede contener valores nulos ni valores duplicados** (no podrá haber dos filas con el mismo valor en este campo). Hay SGBD que incluyen el concepto de clave primaria pero no la hacen obligatoria, por lo que en estos sistemas se pueden definir tablas sin



clave primaria.

- En una tabla pueden existir más de una columna que permita identificar las filas de la tabla, si queremos utilizar tales columnas como identificadores las definiremos como claves **secundarias (alternativas)**. Una clave secundaria tiene las mismas restricciones que una clave primaria, pero como no podemos definir dos claves primarias, definimos la que se vaya a utilizar más frecuentemente como clave primaria y la otra (u otras) como secundarias. Por ejemplo en la tabla *Piezas* la clave primaria es el campo *Codigo* ya que no hay ni puede haber dos piezas con el mismo código. Este campo realmente sirve para identificar las filas de la tabla, sabiendo un valor de código (por ejemplo el 2) sabremos que nos referimos a la fila de la mesa ovalada.

En esta misma tabla tenemos una posible clave alternativa, la formada por los campos *Fabricante* y *Codigo_segun_fab* ya que en la tabla *Piezas* es imposible tener dos filas con la misma combinación de valores en estos campos (el *codigo_segun_fab* es el código que utiliza el fabricante para identificar sus piezas).

- Otro concepto muy importante, fundamental en las bases de datos relacionales, es la **clave ajena (externa o foránea)**.

Una clave ajena es un campo (o combinación de campos) que contiene la referencia a una fila de otra tabla, también puede referirse a la misma tabla. En otras palabras, es un campo que señala a un registro de otra tabla, contiene un valor que identifica un registro de la otra tabla. Son los campos que se utilizan para relacionar las tablas entre sí. Una tabla puede tener 0, una o varias **claves ajenas (externas, foráneas)**.

- Una **clave ajena puede contener valores duplicados y valores nulos**.

Siguiendo el ejemplo anterior, en la tabla *Piezas* tenemos la clave ajena *Fabricante* ya que en este campo nos guardamos un valor que señala a una fila de la tabla *Fabricantes*, en este campo tenemos el código del fabricante que nos suministra la pieza y este código nos lleva al fabricante correspondiente en la tabla *Fabricantes*.

Por ejemplo la pieza 3 es servida por el fabricante *Fab2*, valor que señala al fabricante *Maderas Asociados*.

- El SGBD deberá velar por la integridad de los datos, para ello incluye varias reglas de integridad que se comprobarán de forma automática sin necesidad de la intervención externa de los usuarios o de los programas de aplicación.

Existen distintos tipos de reglas de integridad:

- **La integridad de entidades (integridad de claves):** “Toda tabla debe tener una clave primaria que permite identificar unívocamente los registros que contiene, por lo tanto no puede contener el valor nulo ni valores duplicados”. En el ejemplo anterior si intentamos insertar una nueva pieza con el código 2, el sistema no nos dejará porque ya hay una pieza con este mismo código en la tabla.
- **La integridad referencial:** “En una clave ajena no puede haber un valor no nulo que no exista en la tabla de referencia”. Para que no existan errores de integridad referencial en la base de datos, el sistema comprueba automáticamente que los valores introducidos en las claves ajenas existan en el campo de referencia en la otra tabla, si no existe, no nos dejará insertar el registro.

Volviendo al ejemplo anterior, si intentamos insertar una pieza con un código de fabricante que no existe en la tabla de fabricantes, el sistema no nos dejará.

- A nivel de control sobre los datos, el SGBD debe de proporcionar herramientas para poder definir restricciones de dominio que se comprobarán de forma automática (se comprueba que el valor introducido en una columna pertenece al dominio de la columna, al tipo de datos), y reglas de negocio, que son reglas específicas sobre los datos, en este tipo de reglas entran las reglas de validación y reglas definidas a nivel superior que veremos más adelante. Una regla de validación sería por ejemplo que el precio no pueda ser inferior a 10 euros, y una regla de negocio, que no pueda haber más de 20 fabricantes.
- Un SGBD relacional sigue la arquitectura de tres niveles en la que tenemos en el nivel externo las **vistas**, en nivel conceptual el esquema conceptual con la definición de todas las tablas, columnas que las componen y relaciones entre ellas, en el nivel interno tenemos la definición física de la base de datos.
- Finalmente tenemos para poder manejar la información almacenada en la base de datos un lenguaje que cumple las reglas de Codd, el lenguaje SQL que veremos en próximos temas.

INSTALAR SQL SERVER 2005

Existen diferentes versiones (ediciones) del producto, por lo que es un producto muy versátil, que puede cumplir con las exigencias de cualquier empresa, puede ser utilizado para gestionar bases de datos en un PC en modo local a gestionar todo el sistema de información de grandes empresas pasando por sistemas que requieran menos potencia y por sistemas móviles.

Actualmente se utiliza más en entornos Cliente/servidor con equipos medianos y grandes.

Si la instalación se realiza a partir del archivo descargado de Internet, la descarga se empaqueta como un único ejecutable mediante una tecnología de instalación de Microsoft llamada SFXCab. Al hacer doble clic en el .exe se inicia automáticamente el proceso de instalación.

Tan sólo deberemos seguir el asistente. Los puntos más importantes a tener en cuenta son:

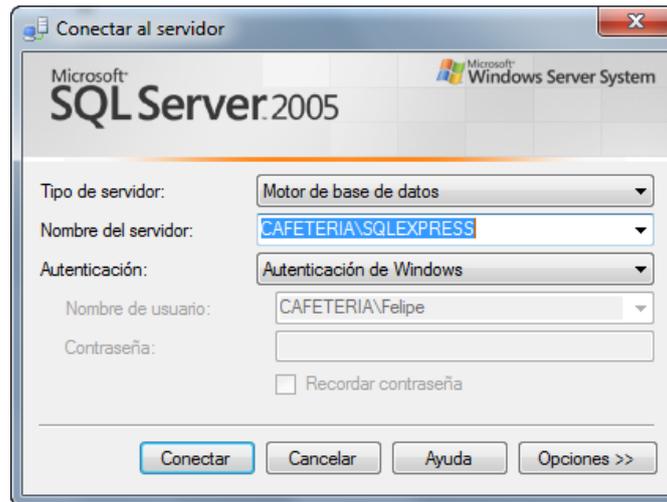
- Habilitar el SQL Server Management Studio en la instalación (si no lo está por defecto) cuando nos pregunte qué componentes deseamos instalar.
- Indicar que se trata de una Instancia predeterminada.

ENTRADA AL SQL SERVER MANAGEMENT STUDIO

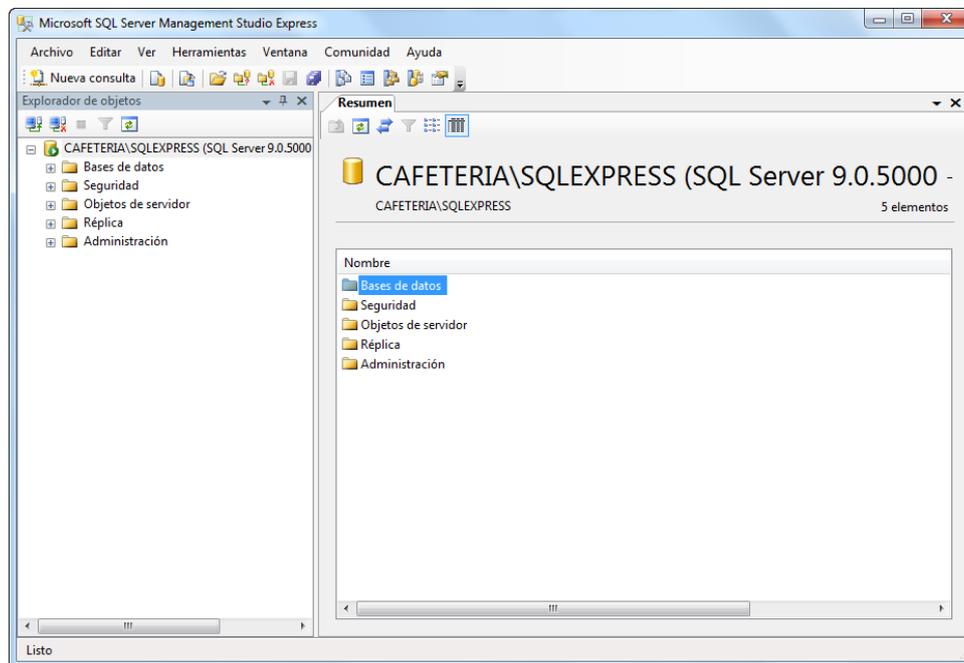
Aunque trabajemos en modo local, la entrada a la herramienta es la misma. Para empezar entramos a través del acceso directo  o a través de Inicio, Programas, Microsoft SQL Server 2005, SQL Server Management Studio.

Lo primero que deberemos hacer es establecer la conexión con el servidor:





Seleccionamos el nombre del servidor y pulsamos el botón *Conectar*. Se abrirá la ventana inicial del SQL Server Management Studio (en adelante SSMS):



En la parte izquierda tenemos abierto el panel Explorador de Objetos en el que aparece debajo del nombre del servidor con el que estamos conectados una serie de carpetas y objetos que forman parte del servidor.

En el panel de la derecha se muestra la zona de trabajo, que varía según lo que tengamos seleccionado en el Explorador de objetos, en este caso vemos el contenido de la carpeta que representa el servidor CAFETERIA.

En la parte superior tenemos el menú de opciones y la barra de herramientas Estándar.

Con las siguientes opciones:



1. Nueva consulta	6. Consulta de SQL Server Mobile	11. Resumen
2. Consulta de motor de Base de datos	7. Abrir archivo	12. Explorador de Objetos
3. Consulta MDX de Analysis Services	8. Guardar	13. Explorador de Plantillas
4. Consulta DMX de Analysis Services	9. Guardar todo	14. Ventana de Propiedades
5. Consulta MXLA de Analysis Services	10. Servidores registrados	

En caso de que utilices la versión Express, es posible que no dispongas de algunos de éstos botones.

ESTRUCTURA INTERNA DE UNA BASE DE DATOS

Antes de empezar tenemos que tener claro cómo se organiza la información en una base de datos SQL Server 2005.

Las bases de datos de SQL Server 2005 utilizan tres tipos de archivos:

- **Archivos de datos principales**

En una base de datos SQLServer los datos se pueden repartir en varios archivos para mejorar el rendimiento de la base de datos.

El archivo de datos principal es el punto de partida de la base de datos y apunta a los otros archivos de datos de la base de datos. Cada base de datos tiene obligatoriamente un archivo de datos principal. La extensión recomendada para los nombres de archivos de datos principales es .mdf.

- **Archivos de datos secundarios**

Los archivos de datos secundarios son todos los archivos de datos menos el archivo de datos principal. Puede que algunas bases de datos no tengan archivos de datos secundarios, mientras que otras pueden tener varios archivos de datos secundarios. La extensión de nombre de archivo recomendada para los archivos de datos secundarios es .ndf. Además los archivos de datos se pueden agrupar en grupos de archivos. Para cada base de datos pueden especificarse hasta 32.767 archivos y 32.767 grupos de archivos.

- **Archivos de registro**

Los archivos de registro (archivos de log) almacenan toda la información de registro que se utiliza para recuperar la base de datos, el también denominado registro de transacciones. Como mínimo, tiene que haber un archivo de registro por cada base de datos, aunque puede haber varios. La extensión recomendada para los nombres de archivos de registro es .ldf.

SQL Server 2005 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.

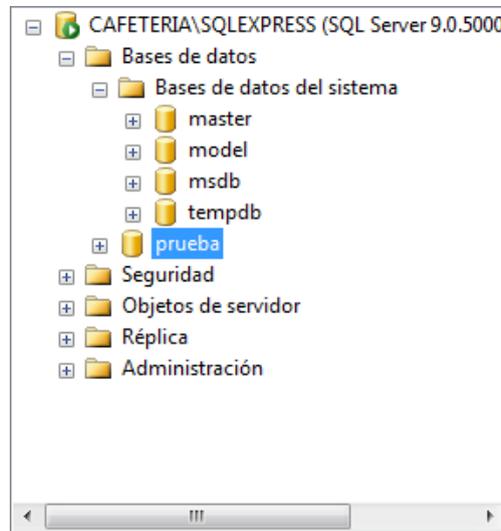
Cada base de datos tiene al menos 2 archivos (un archivo de datos principal y un archivo de registro) y opcionalmente un grupo de archivos.

Los archivos de datos y de registro de SQL Server se pueden colocar en sistemas de archivos FAT o NTFS. Se recomienda utilizar NTFS por los aspectos de seguridad que ofrece. No se pueden colocar grupos de archivos de datos de lectura y escritura, y archivos de registro, en un sistema de archivos NTFS comprimido. Sólo las bases de datos de sólo lectura y los grupos de archivos secundarios de sólo lectura se pueden colocar en un sistema de archivos NTFS comprimido.



CREAR UNA BASE DE DATOS EN SSMS

En el Explorador de objetos, si desplegamos la carpeta Bases de datos nos aparecen Bases de datos del sistema y las bases de datos de usuario después de la carpeta Instantáneas...



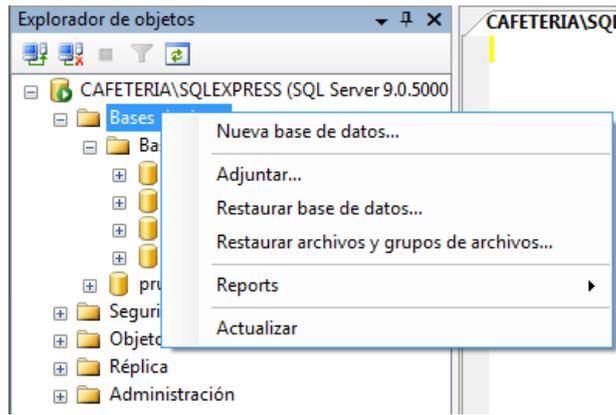
Después de la instalación, en la carpeta Bases de datos del sistema se habrá creado una especial denominada master se utiliza como base de datos de usuario por defecto. Las demás bases de datos forman también parte del diccionario de datos y las utiliza el sistema para llevar a cabo su gestión.

Base de datos	Descripción
master	Controla las bases de datos de usuario y el funcionamiento de SQL Server de forma global al realizar un seguimiento de información como las cuentas de usuario, las variables de entorno configurables y los mensajes de error del sistema.
model	Proporciona una plantilla o un prototipo para las nuevas bases de datos de usuario.
tempdb	Proporciona un área de almacenamiento para tablas temporales y otras necesidades de almacenamiento temporal.
msdb	Ofrece un área de almacenamiento para información de programación e historial de trabajos.

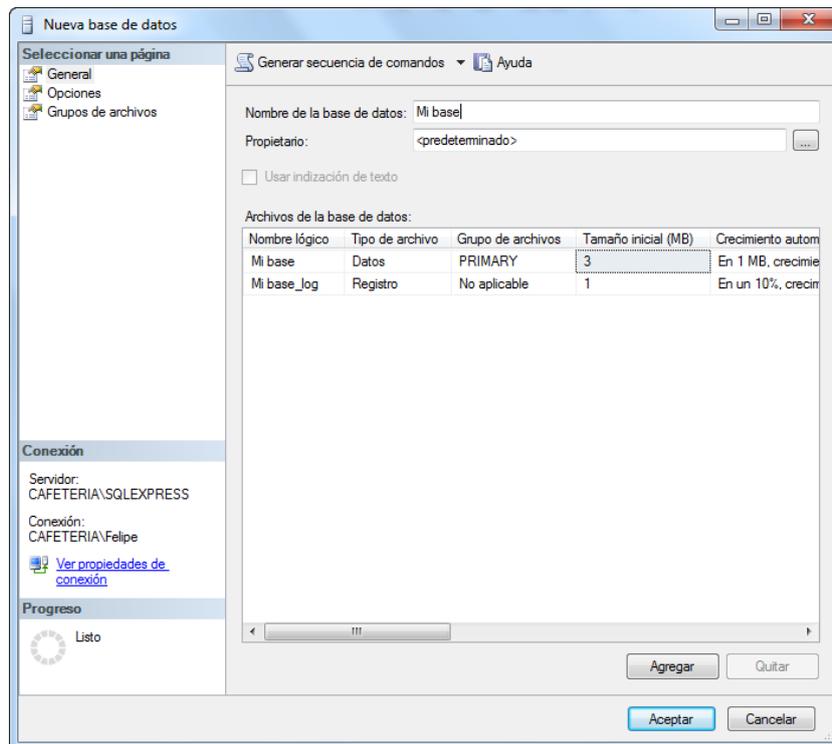
Las bases de datos de los usuarios se deben crear preferentemente fuera de la carpeta Bases de datos del sistema.

Para crear una nueva base de datos de usuario nos posicionamos sobre la carpeta Bases de datos y con el botón derecho del ratón desplegamos el menú contextual del que elegimos la opción Nueva base de datos...





Se abre a continuación el cuadro de diálogo donde definiremos la base de datos que queremos crear:



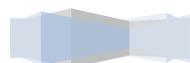
Lo mínimo a introducir será el campo Nombre de la base de datos, éste es el nombre de la base de datos lógica, la base de datos a la que nos referiremos dentro del SSMS, a nivel conceptual (en la imagen Mibase).

Esta base de datos está asociada a dos archivos físicos, en la parte inferior aparecen esos archivos. Para facilitarnos la tarea, al teclear el nombre de la bd lógica, se rellenan automáticamente los nombres de los archivos físicos, el de datos con el mismo nombre y el del archivo de registro con el mismo nombre seguido de _log. Estos nombres son los nombres que se asumen por defecto pero los podemos cambiar, posicionando el cursor en el nombre y cambiándolo.

Para cada archivo físico podemos definir una serie de parámetros como el tipo de archivo (si es de datos o de transacciones Registro) y su ocupación inicial (Tamaño inicial).

Si no indicamos ninguna ubicación podemos ver que los guarda en la carpeta del SQL Server/MSSQL.n/MSSQL/DATA.

n representa un número que puede variar de una instalación a otra.





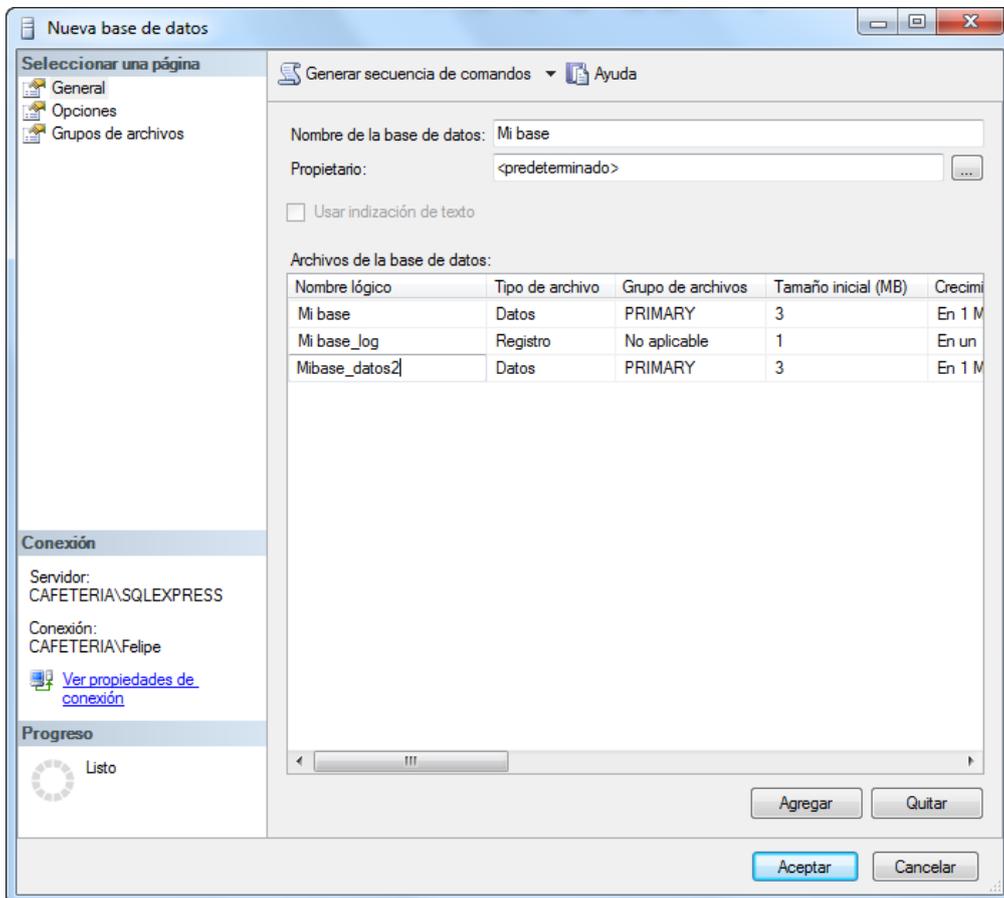
Archivos de la base de datos:

Nombre lógico	Ruta de acceso
Mi base	C:\Program Files (x86)\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
Mi base_log	C:\Program Files (x86)\Microsoft SQL Server\MSSQL.1\MSSQL\DATA

Estos son los archivos mínimos en los que se almacenará la base de datos, pero como ya vimos anteriormente se puede almacenar en más archivos, los tenemos que definir todos en esta ventana a continuación de los dos obligatorios.

Para añadir más archivos físicos disponemos del botón Agregar.

Al pulsar el botón Agregar se crea una nueva fila en la tabla de archivos físicos donde deberemos escribir el nombre del archivo, su tipo (desplegando la lista podemos elegir entre de datos o de registro) y demás parámetros.

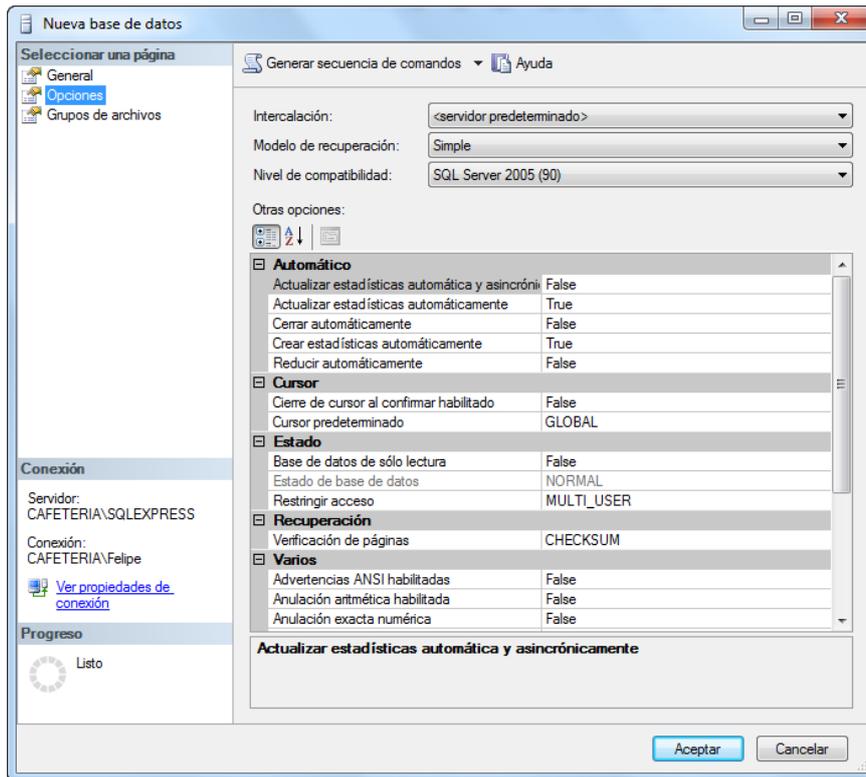


Al agregar un nuevo archivo se activa el botón Quitar, siempre que estemos posicionados encima de un archivo secundario para poder así eliminarlo si lo queremos.

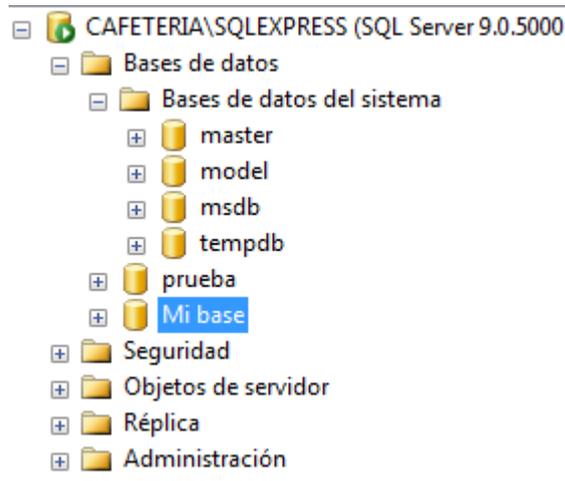
No podemos eliminar ni el de datos primario, ni el de registro inicial.

Si nos fijamos en la zona de la izquierda, vemos que nos encontramos en la pestaña General, podemos cambiar otros parámetros de la base de datos pulsando en Grupos de archivos o en Opciones:





Al final pulsamos en Aceptar y se creará la base de datos.



Aparecerá dentro de la carpeta Bases de datos. Si no se ve pulsa en el icono Actualizar

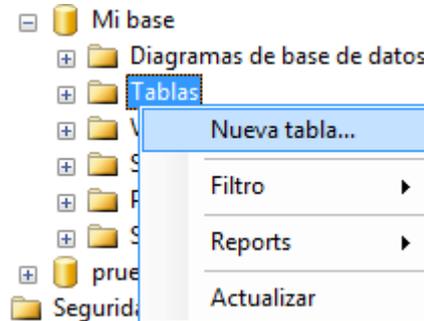
Desde el Explorador de Windows podemos ver que en la carpeta indicada se han creado los archivos físicos con los nombres que le hemos indicado.



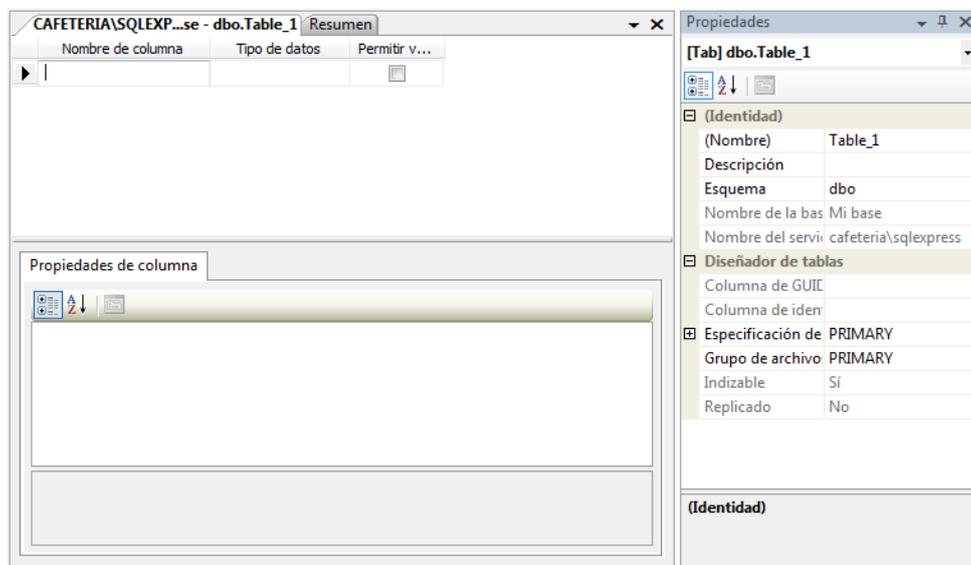


CREAR UNA NUEVA TABLA

Para crear una nueva tabla primero nos tenemos que posicionar en la base de datos donde queremos que se almacene la tabla, desplegar el menú contextual y seleccionar la opción Nueva tabla.



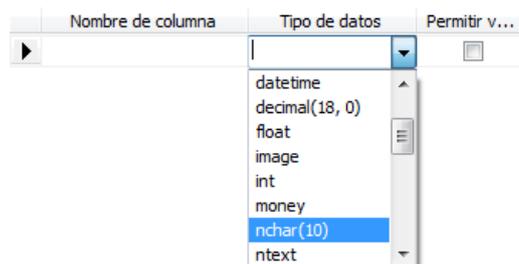
En la ventana que se abre debemos definir las columnas de la tabla:



A cada columna se le asigna un nombre, un tipo de datos, y opcionalmente una serie de propiedades, en este tema veremos las básicas y las demás las veremos con más detalle cuando veamos la instrucción SQL CREATE TABLE.

De momento no tenemos definida ninguna columna, al teclear un nombre se crea una primera entrada en esta tabla con la definición de la primera columna. En la columna Tipo de datos elegimos qué tipo de valores se podrán almacenar en la columna.

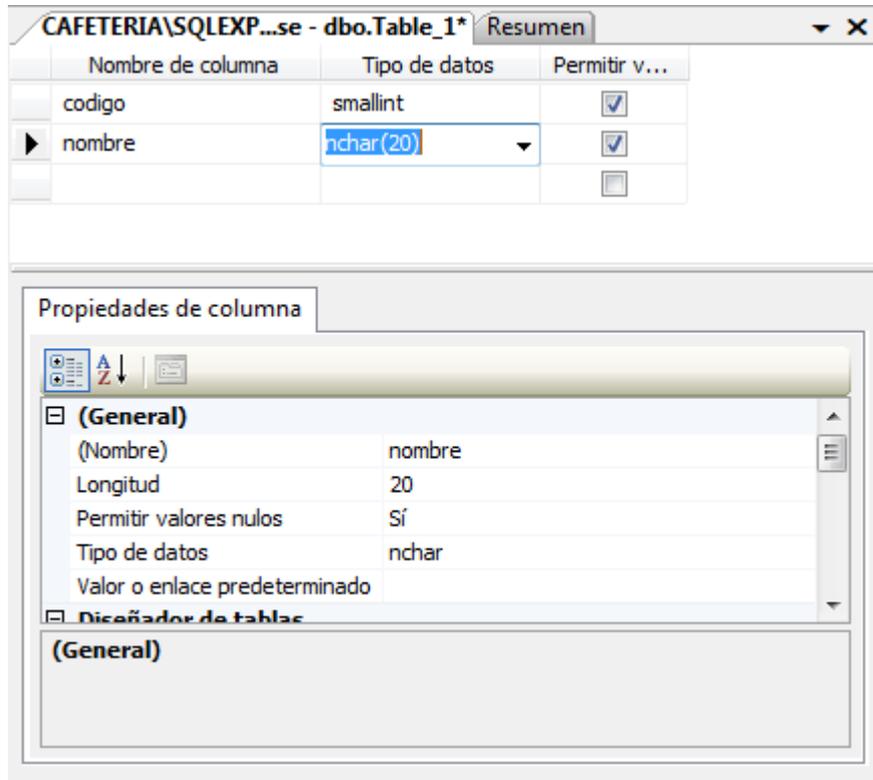
Tipos de datos





Podemos elegir entre todos los tipos que aparecen arriba.

Algunos tipos no necesitan más, como por ejemplo el tipo entero (int), y otros se pueden completar con una longitud, como los tipos alfanuméricos:



En este ejemplo hemos definido una columna (Codigo) de tipo Entero corto (Smallint), y una columna (Nombre) que almacenará hasta 20 caracteres alfanuméricos (nchar(20)), en este caso la longitud la indicamos en la pestaña Propiedades de columna en la propiedad Longitud.

Las propiedades de la columna pueden variar dependiendo del tipo de datos de la columna seleccionada, por ejemplo los campos enteros no tienen la propiedad longitud, ya que el propio tipo define la longitud del campo, en cambio los campos de tipo numérico o decimal no tienen la propiedad longitud pero sí las propiedades escala y precisión, los valores que permiten definir el tamaño del campo.

Valores nulos

También podemos indicar si la columna permitirá valores nulos o no, o bien cambiando la propiedad Permitir valores nulos que aparece debajo de la propiedad Longitud, o bien simplemente marcando o desmarcando la casilla de la columna Permitir valores nulos que se encuentra al lado de la columna Tipo de datos. Si la casilla está marcada, el usuario podrá no rellenar el campo cuando inserte una fila de datos en la tabla.

Columna con contador

En la mayoría de los sistemas gestores de bases de datos tenemos un tipo de datos de tipo contador, autonumérico, autoincremental, etc. Este tipo hace que el propio sistema es el encargado de rellenar el campo con un valor que va incrementando conforme se crean más filas de datos en la tabla.

Las columnas de este tipo se utilizan normalmente para numerar las filas de la tabla, como no habrán dos filas con el mismo valor (el sistema se encarga de incrementar el valor cada vez que se crea una

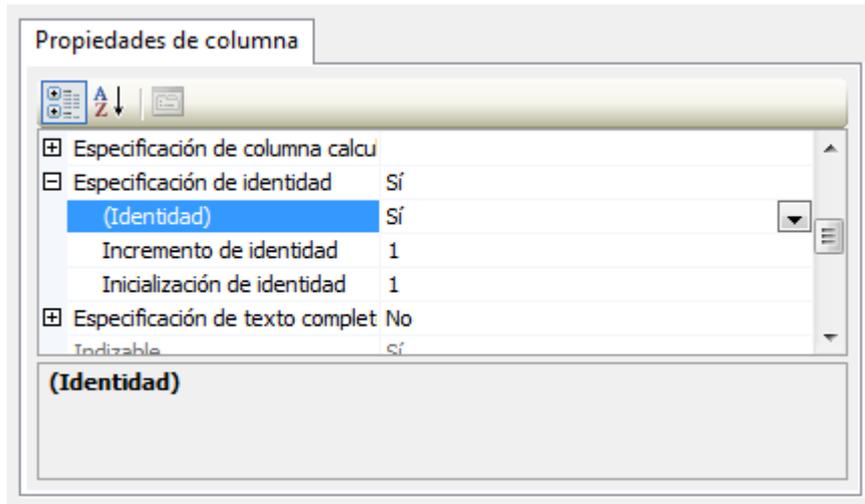




nueva fila), estos campos se suelen utilizar como claves primarias.

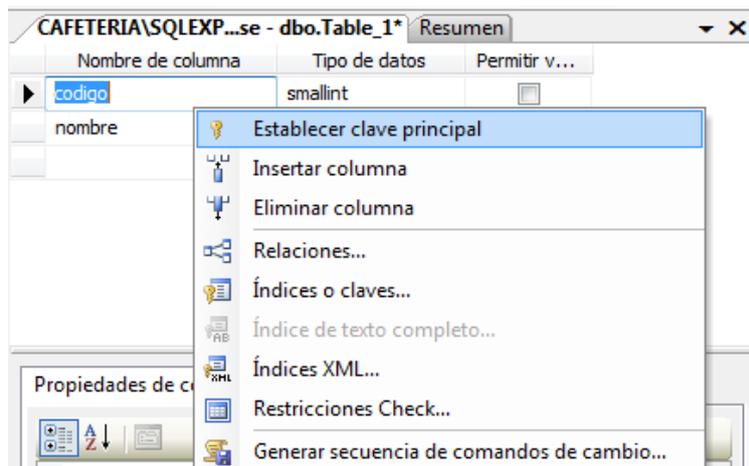
En SQL Server 2005 no existe el tipo de datos Contador pero se consigue el mismo funcionamiento asignando a la columna un tipo de datos numérico y definiendo la columna como columna de identidad. En las propiedades de la columna marcamos Sí en la propiedad (Identidad) y a continuación podemos indicar en qué valor queremos que empiece el contador (Inicialización de identidad) y en cuánto incrementará cada vez que se cree un nuevo registro (Incremento de identidad).

Aunque este tipo de columnas se utiliza frecuentemente como clave primaria, SQL Server no le asigna automáticamente esta función, la tenemos que definir nosotros mismos, pero sí fuerza a que sea una columna sin valores nulos. No se puede definir más de una columna de identidad por tabla.



Clave primaria

Para definir una columna como clave primaria, posicionamos el puntero del ratón sobre la columna, desplegamos el menú contextual y seleccionamos la opción Establecer Clave principal:



Aparecerá una llave a la izquierda del nombre, símbolo de las claves principales:

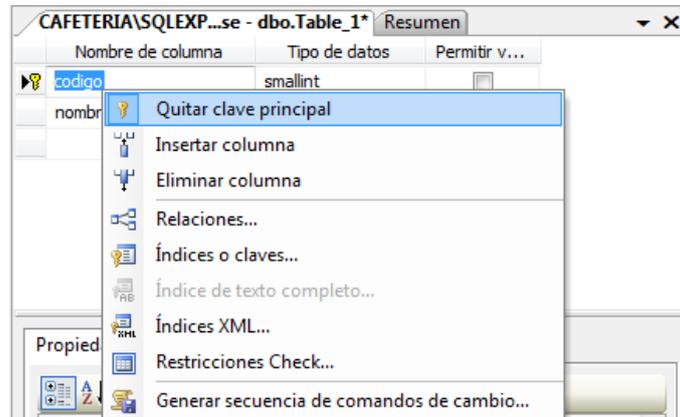
Nombre de columna	Tipo de datos	Permitir v...
 codigo	smallint	<input type="checkbox"/>
nombre	nchar(20)	<input type="checkbox"/>

Para definir una clave primaria compuesta por varias columnas, seleccionamos las columnas manteniendo pulsada la tecla Ctrl y luego seleccionamos la opción.





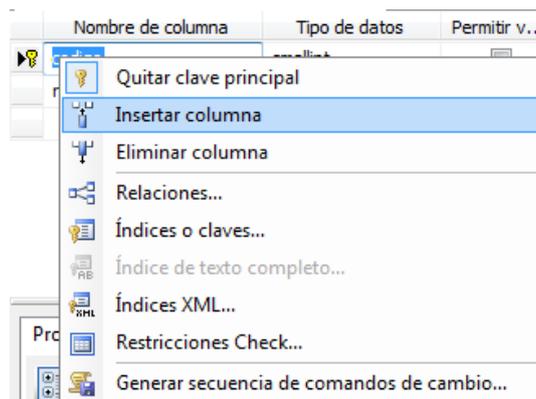
Para quitar una clave principal, hacemos lo mismo pero en esta ocasión seleccionamos la opción Quitar clave principal.



También podemos utilizar el icono de la barra de herramientas.

Añadir o eliminar columnas

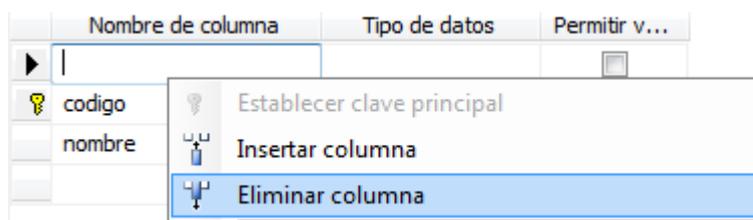
Una vez definidas algunas columnas, si queremos añadir una nueva columna entre dos, nos posicionamos en la segunda y seleccionamos la opción Insertar columna del menú contextual.



La nueva columna se colocará delante:

	Nombre de columna	Tipo de datos	Permitir v...
			<input type="checkbox"/>
	codigo	smallint	<input type="checkbox"/>
	nombre	nchar(20)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

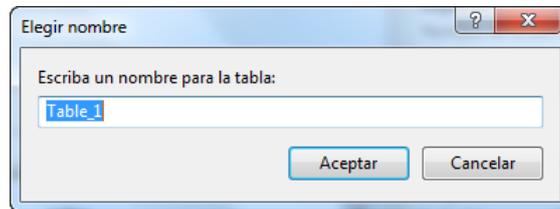
Del mismo modo si queremos eliminar la definición de una columna, nos posicionamos en la columna a eliminar y seleccionamos la opción Eliminar columna:



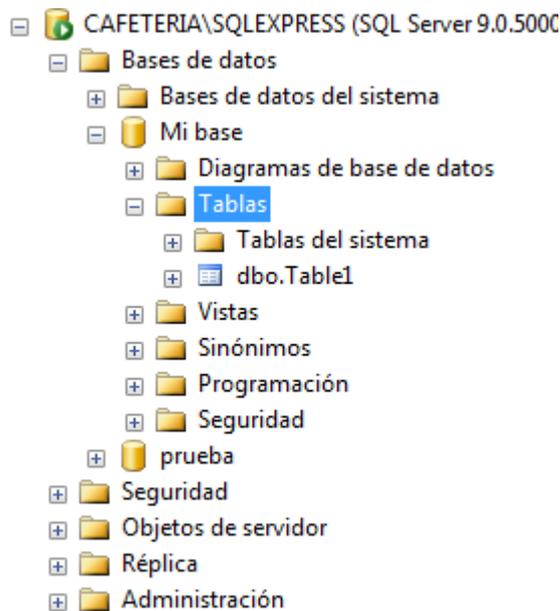


O simplemente hacemos clic en la zona a la izquierda del nombre y pulsamos la tecla Supr.

Finalmente guardamos la tabla, nos pedirá el nombre de la tabla:

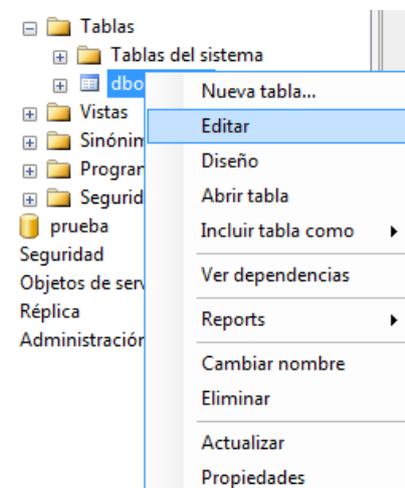


La nueva tabla aparecerá en la lista de tablas de la base de datos:



MODIFICAR LA DEFINICIÓN DE UNA TABLA

Para entrar a la ventana de definición de la tabla utilizamos la opción Modificar de su menú contextual (También es posible que se llame Diseño):



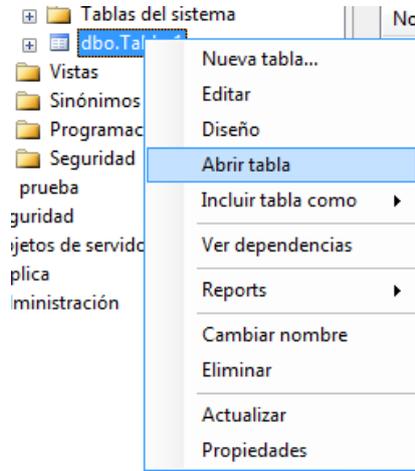
Se abrirá la ventana que ya conocemos para definir las columnas de la tabla.





INSERTAR DATOS EN LA TABLA

Ahora que tenemos la tabla creada podemos rellenarla con datos. Para eso debemos abrir la tabla:



Se abrirá una ventana parecida a esta:

CAFETERIA\SQLEXP...se - dbo.Table_1 Re		
	codigo	nombre
*	NULL	NULL

La primera columna sirve para indicarnos el estado de una fila, por ejemplo el * nos indica que es una nueva fila, esta fila realmente no está en la tabla, nos sirve de contenedor para los nuevos datos que queremos insertar.

Para insertar una nueva fila de datos sólo tenemos que rellenar los campos que aparecen en esa fila (la del *), al cambiar de fila los datos se guardarán automáticamente en la tabla a no ser que alguno infrinja alguna regla de integridad, en ese caso SQL Server nos devuelve un mensaje de error para que corrijamos el dato erróneo, si no lo podemos corregir entonces sólo podemos deshacer los cambios.

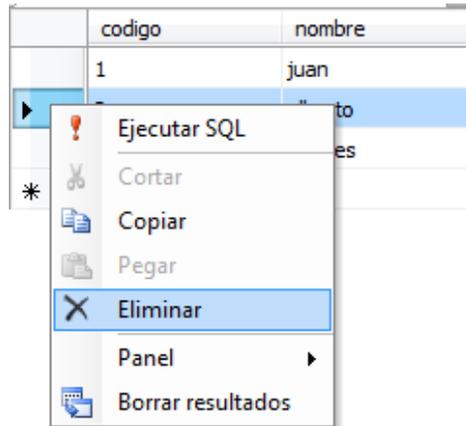
MODIFICAR DATOS

Para modificar un valor que ya está en una fila de la tabla sólo tenemos que posicionarnos en el campo y rectificar el valor. En cuanto modificamos un valor, la fila aparece con un lápiz escribiendo (ver imagen), este lápiz nos indica que la fila se ha modificado y tiene nuevos datos por guardar. Al salir de la fila ésta se guardará automáticamente a no ser que el nuevo valor infrinja alguna regla de integridad. Si queremos salir de la fila sin guardar los cambios, tenemos que cancelar la actualización pulsando la tecla ESC.

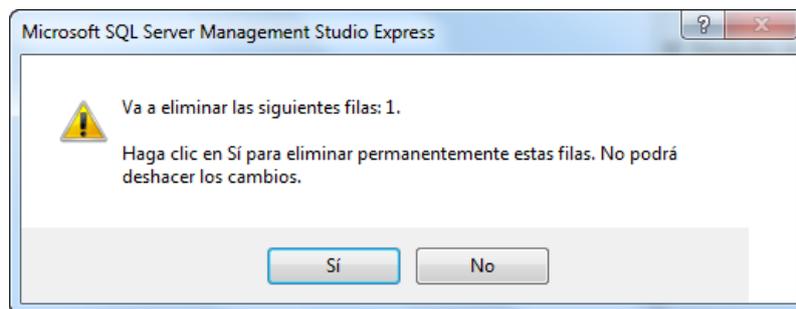
ELIMINAR FILAS

Para eliminar una fila completa, la seleccionamos y pulsamos la tecla Supr o bien desplegamos su menú contextual y seleccionamos la opción Eliminar.





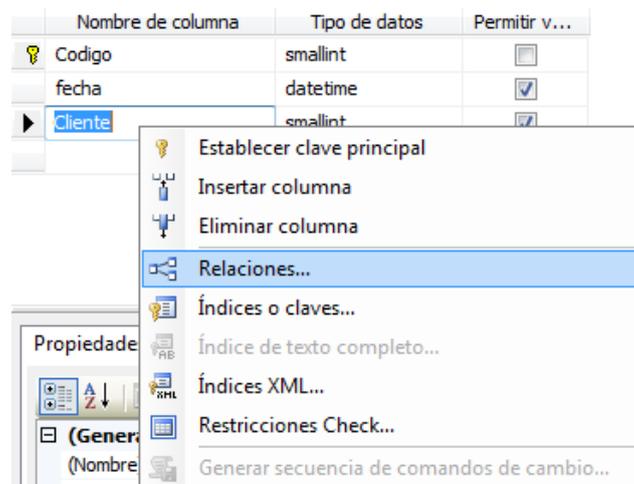
En cualquiera de los dos casos nos aparece un mensaje de confirmación.



RELACIONAR TABLAS

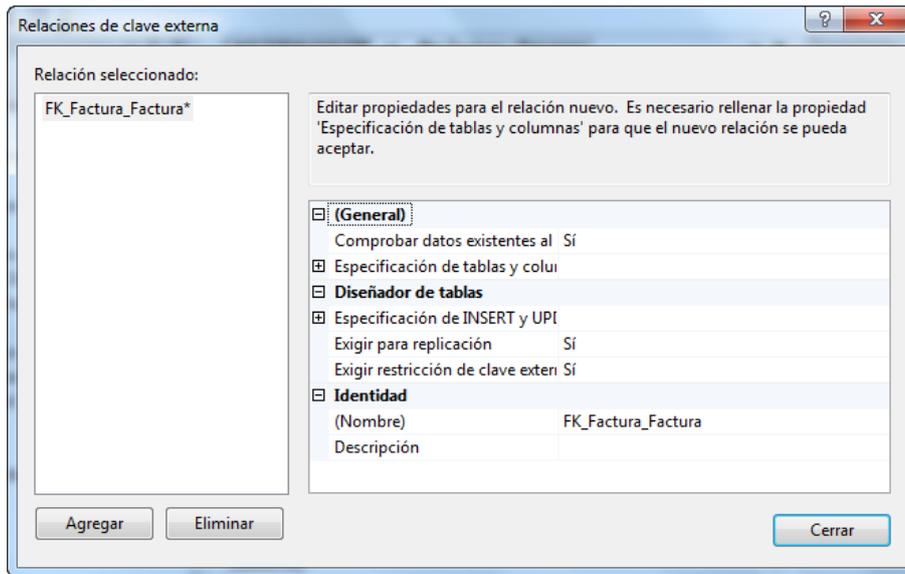
Como ya hemos visto, en una base de datos relacional, las relaciones entre las tablas se implementan mediante la definición de claves ajenas, que son campos que contienen valores que señalan a un registro en otra tabla, en esta relación así creada, la tabla referenciada se considera principal y la que contiene la clave ajena es la subordinada.

Desde el entorno gráfico del SSMS podemos definir claves ajenas entrando en el diseño de la tabla y desplegando el menú contextual del campo que va a ser clave ajena:

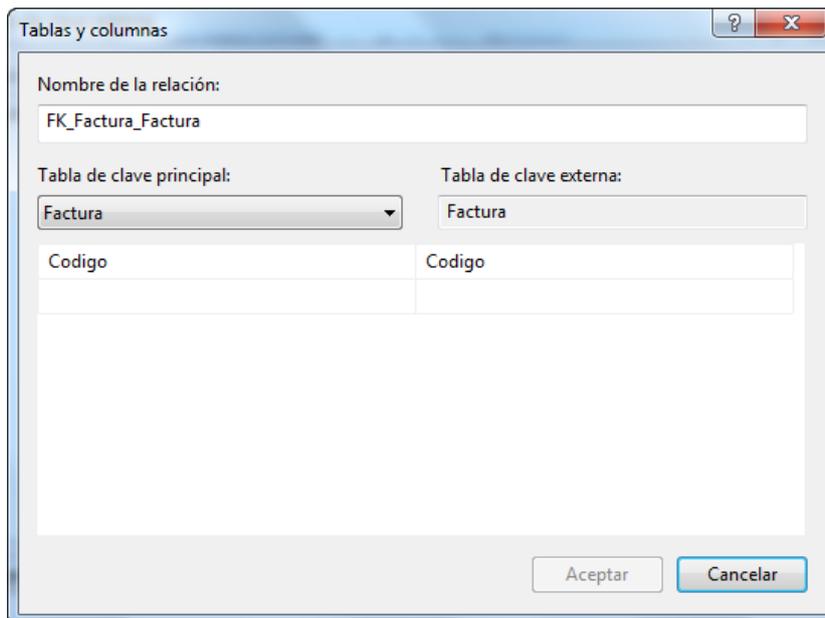


Seleccionamos la opción Relaciones y se abre la ventana:



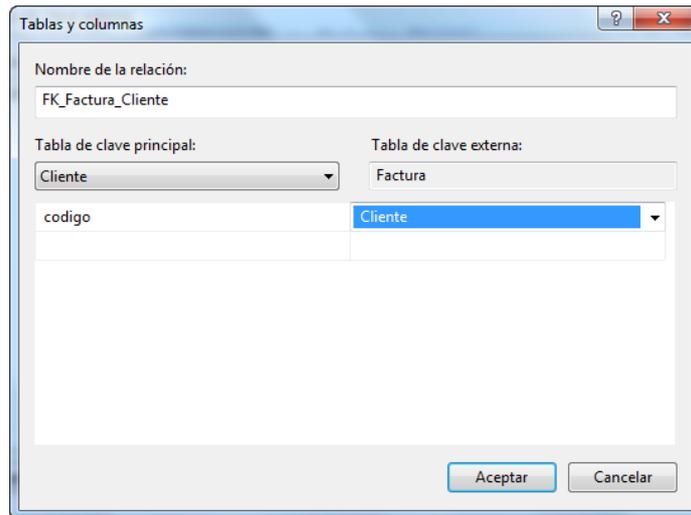


Al pulsar el botón que se encuentra en la fila Especificación de tablas y columnas se abre el diálogo donde definiremos la relación:



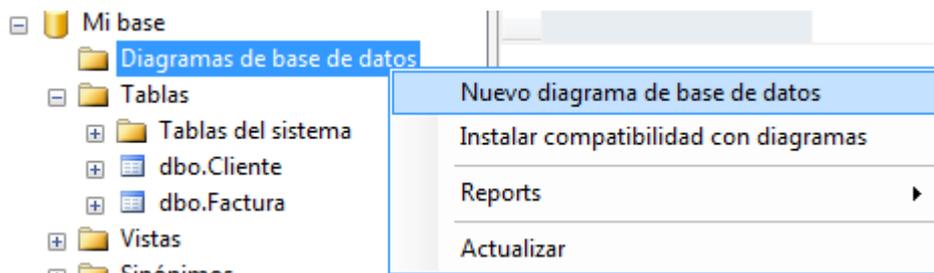
En la parte derecha tenemos la tabla en la que estamos y el campo que va a actuar como clave ajena, sólo nos queda elegir en el desplegable de la izquierda la tabla a la que hace referencia la clave y al seleccionar una tabla, a la izquierda del campo clave ajena podremos elegir el campo de la otra tabla por el que se relacionarán las tablas. En nuestro caso será:





De esta forma hemos definido una relación entre las tablas Facturas y Clientes. Para ver las relaciones existentes entre las diferentes tablas tenemos los diagramas.

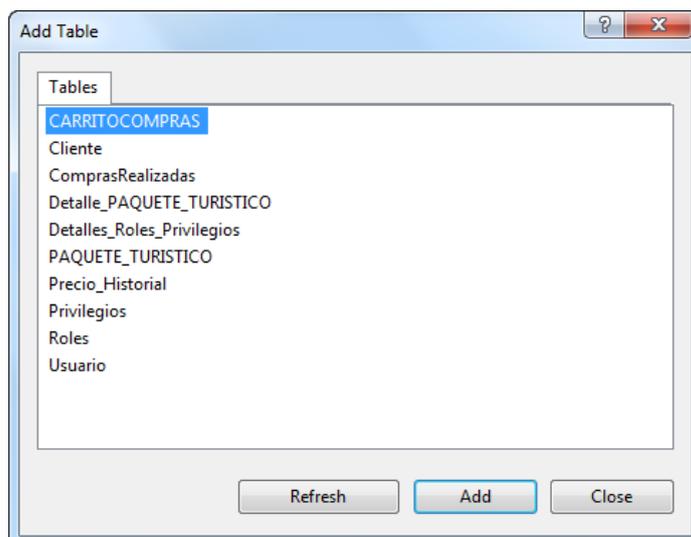
Primero debemos definir el diagrama, para ello seleccionamos la opción correspondiente:



Si no tenemos todavía ningún diagrama creado, nos aparece un mensaje.

Elegimos Sí y se crea digamos el soporte donde se pintará el diagrama.

A continuación nos aparece el nuevo diagrama ahora si elegimos crear un nuevo diagrama nos preguntará las tablas a incluir en el diagrama:

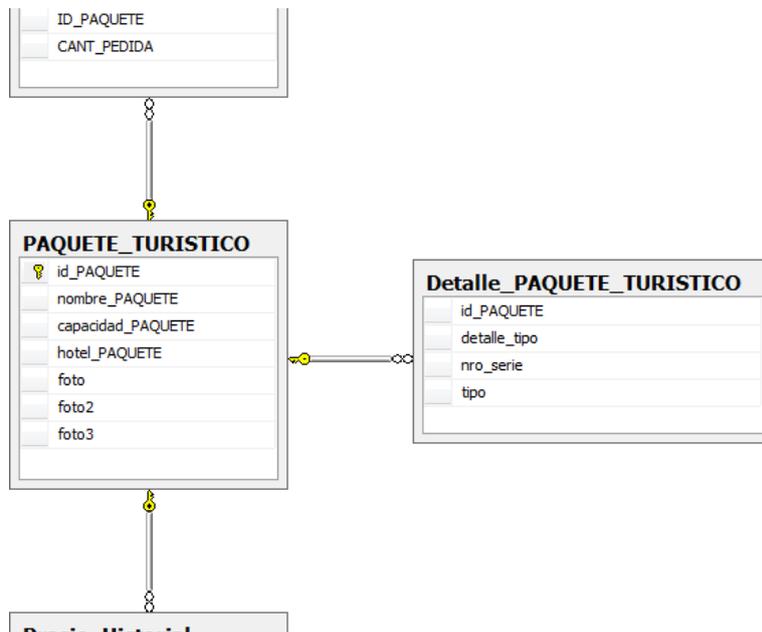


Seleccionamos cada una y pulsamos Agregar, cuando hayamos agregado al diagrama todas las que

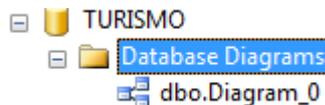




queremos pulsamos en Cerrar y aparecerán en el diagrama las tablas con las relaciones que tengan definidas en ese momento:



La llave indica la tabla principal (padre) y el símbolo infinito señala la tabla que contiene la clave ajena. En el examinador de objetos en la carpeta Diagramas de base de datos aparecen todos los diagramas definidos hasta el momento:

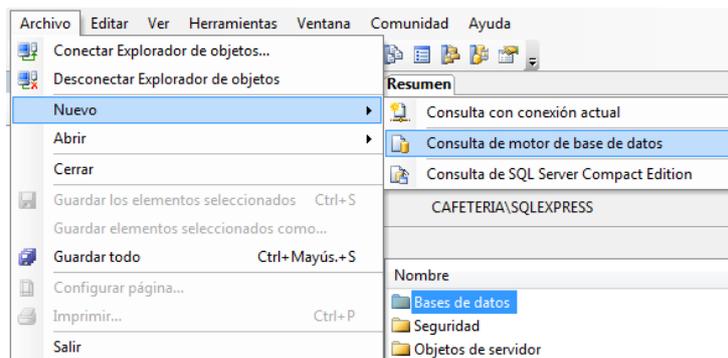


Hemos aprendido hasta ahora lo básico para poder crear una base de datos y rellenarla con tablas relacionadas entre sí y con datos, ahora veamos cómo recuperar esos datos.

ABRIR UNA NUEVA CONSULTA

Vamos a ver ahora cómo crear consultas SQL y ejecutarlas desde el entorno del SSMS.

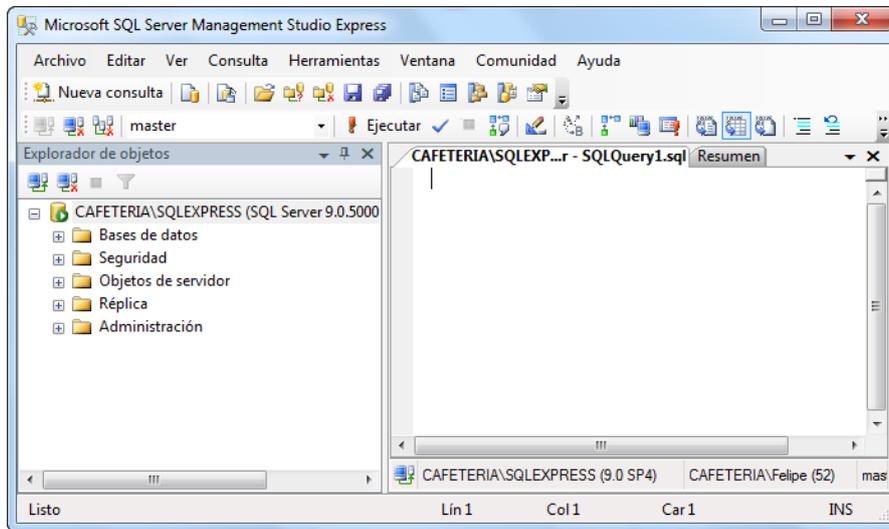
Para ello debemos abrir la zona de trabajo de tipo Query, abriendo una nueva consulta, seleccionando previamente el servidor y pulsando el botón **Nueva consulta** de la barra de botones o si queremos realizar la consulta sobre un servidor con el cual todavía no hemos establecido conexión, seleccionando de la barra de menús la opción Nuevo > Consulta de motor de base de datos:





En este último caso nos aparecerá el cuadro de diálogo para establecer la conexión (el mismo que vimos al principio del tema).

A continuación se abrirá una nueva pestaña donde podremos teclear las sentencias SQL:



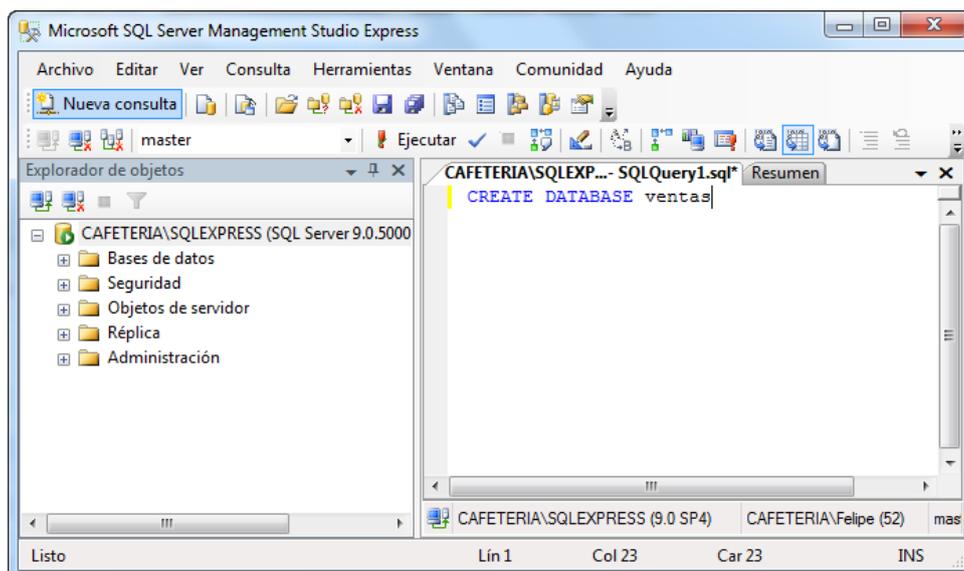
Además aparece una nueva barra de botones que nos permitirá ejecutar los comandos más útiles del modo query.

ESCRIBIR Y EJECUTAR CÓDIGO TRANSACT-SQL

Sólo tenemos que teclear la sentencia a ejecutar, por ejemplo empezaremos por crear la base de datos. Utilizaremos la sentencia CREATE DATABASE mínima:

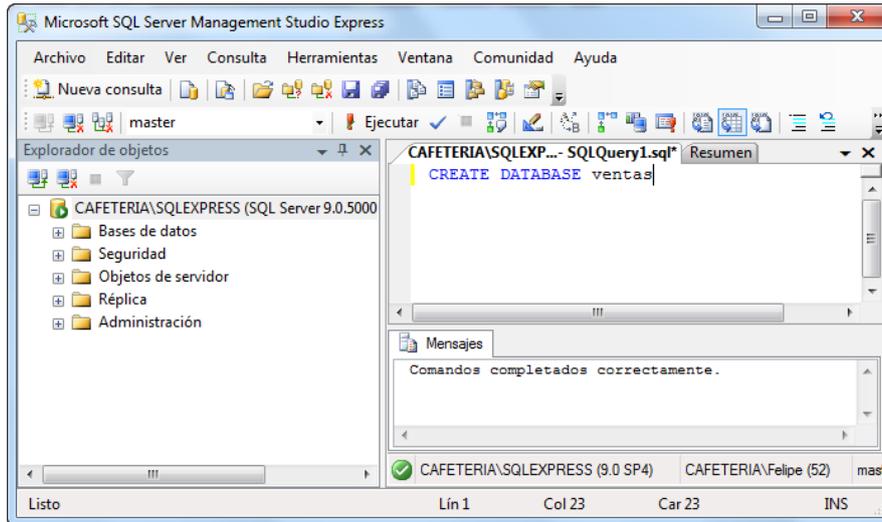
CREATE DATABASE ventas;

Al pulsar el botón Ejecutar se ejecuta la sentencia y aparece en la parte inferior el resultado de la ejecución, en la pestaña Mensajes:



Si ahora desplegamos la carpeta Bases de Datos del Explorador de Objetos, observaremos la base de datos que hemos creado:





Si la ejecución de la sentencia produce un error, el sistema nos devolverá el mensaje de error escrito en rojo en la pestaña Mensajes.

Podemos incluir en una misma consulta varias sentencias SQL, cuando pulsamos Ejecutar se ejecutarán todas una detrás de otra. Si tenemos varias consultas y sólo queremos ejecutar una, la seleccionaremos antes de ejecutarla.

LA BASE DE DATOS PREDETERMINADA

Cuando ejecutamos consultas desde el editor, nos tenemos que fijar sobre qué base de datos se va a actuar.

Fijándonos en la pestaña de la consulta, en el nombre aparece el nombre del servidor seguido de un punto y el nombre de la base de datos sobre la que se va a actuar y luego un guión y el nombre de la consulta.

En la imagen anterior tenemos CAFETERIA\SQLEXPRESS.master – SQLQuery1.sql, lo que nos indica que la consulta se llama SQLQuery1.sql, y que se va a ejecutar sobre la base de datos master que se encuentra en el servidor CAFETERIA\SQLEXPRESS.

Cuando creamos una nueva consulta, ésta actuará sobre la base de datos activa en ese momento. Por defecto la base de datos activa es la predeterminada (master). Si queremos que la base de datos activa sea por ejemplo la base de datos ventas, hacemos clic sobre su nombre en el Explorador de objetos, y ésta pasará a ser la base de datos activa. Si ahora creamos una nueva consulta, ésta actuará sobre la base de datos ventas.

Si queremos crear una consulta que siempre actúe sobre una determinada base de datos y no nos queremos preocupar de qué base de datos tenemos activa podemos añadir al principio de la consulta la instrucción USE nombreBaseDatos; esto hará que todas las instrucciones que aparezcan después, se ejecuten sobre la base de datos indicada.

Por ejemplo:

```
USE ventas;
SELECT * FROM pedidos;
```

Obtiene todos los datos de la tabla *pedidos* que se encuentra en la base de datos *ventas*. Si no utilizamos USE y almacenamos la consulta, al abrirla otra vez, cogerá como base de datos la





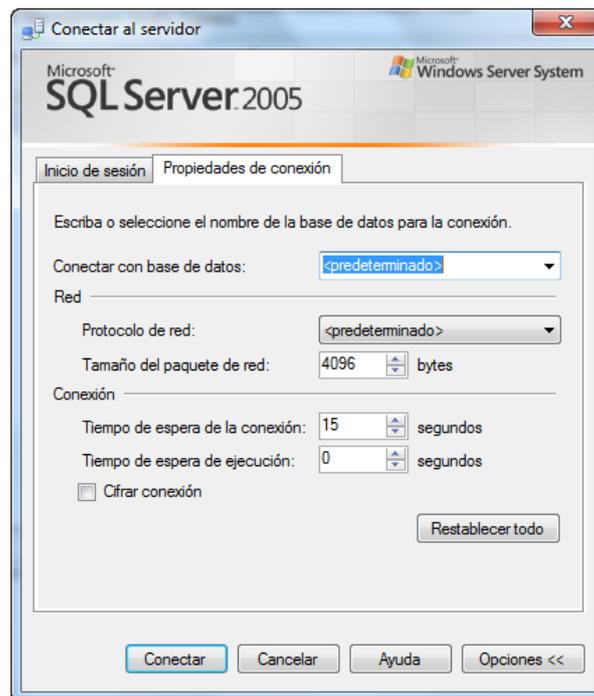
predeterminada (no la activa) y se volverá a ejecutar sobre la base de datos master.

Normalmente utilizaremos como base de datos la nuestra y no la base de datos *master*, por lo que nos será útil cambiar el nombre de la base de datos por defecto, esto lo podemos hacer cambiando la base de datos por defecto en el id de sesión.

Para ello, cuando vamos a conectar con el servidor:

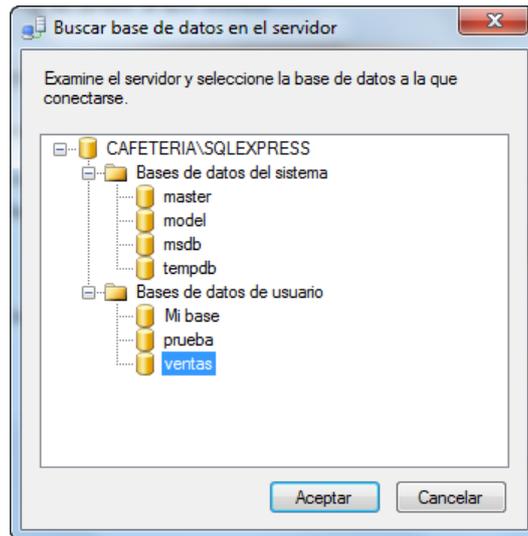


Pulsamos en el botón Opciones >>



En la pestaña Propiedades de conexión, en el cuadro Conectar con base de datos: Seleccionamos <Examinar servidor > para elegir la base de datos.





La elegimos y aceptamos. A partir de ese momento la base de datos elegida será la que SQL Server coja por defecto en todas las sesiones de ese usuario.

EL EDITOR DE TEXTO

Para facilitarnos la redacción y corrección de las sentencias, el editor de SQL presenta las palabras de distintos colores según su categoría y podemos utilizar el panel Explorador de Objetos para arrastrar desde él los objetos sobre la zona de trabajo y así asegurarnos de que los nombres de los objetos (por ejemplo nombre de tabla, de columna, etc.) sean los correctos.

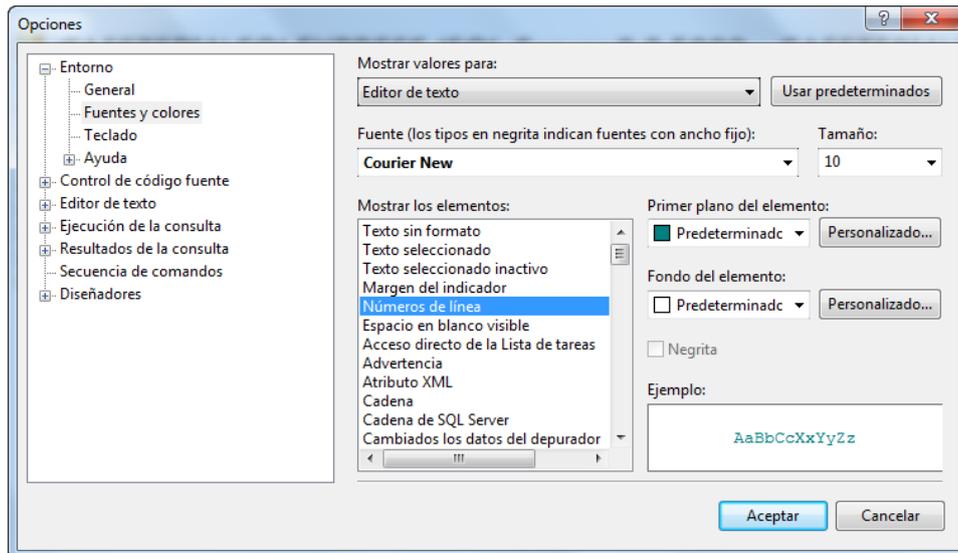
Como hemos dicho el texto que se escribe en este editor de código se colorea por categoría. Los colores son los mismos que se utilizan en todo el entorno SQL Server. En esta tabla aparecen los colores más comunes.

Color	Categoría
Rojo	Cadena de caracteres
Verde oscuro	Comentario
Negro sobre fondo plateado	Comando SQLCMD
Fucsia	Función del sistema
Verde	Tabla del sistema
Azul	Palabra clave
Verde azulado	Números de línea o parámetro de plantilla
Rojo oscuro	Procedimiento almacenado de SQL Server
Gris oscuro	Operadores

CONFIGURAR UN ESQUEMA DE COLORES PERSONALIZADOS

En el menú Herramientas > Opciones, desplegando la opción Entorno, Fuentes y colores, se puede ver la lista completa de colores y sus categorías, así como configurar un esquema de colores personalizado:





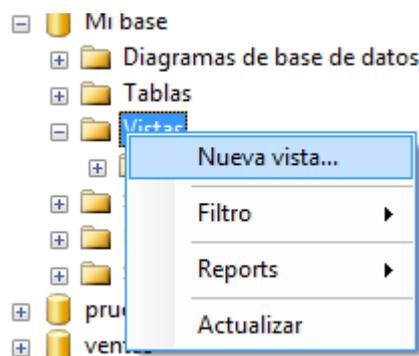
En la lista **Mostrar valores para**, seleccionamos el entorno que se verá afectado.

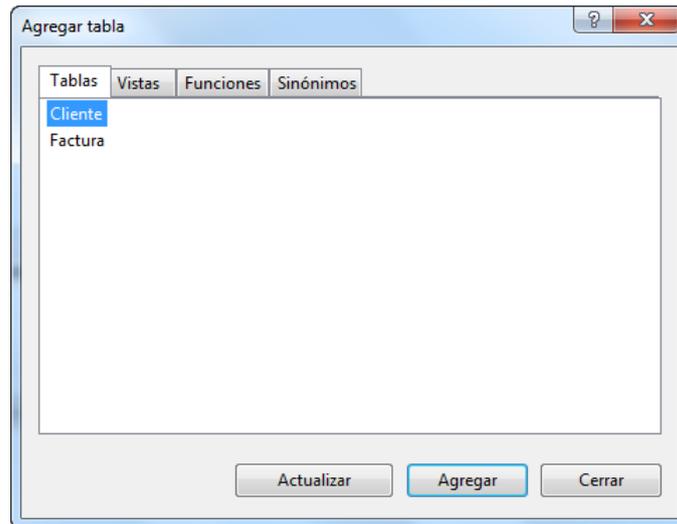
El botón **Usar predeterminados** nos permite volver a la configuración predeterminada.

Ahora sólo nos queda aprender a redactar sentencias SQL, cosa que se verá en otro momento, mientras tanto podemos utilizar el **Generador de Consulta** que incluye **SSMS** y que veremos a continuación en el apartado sobre vistas.

LAS VISTAS

Las consultas que hemos visto hasta ahora son trozos de código SQL que podemos guardar en un archivo de texto y abrir y ejecutar cuando queramos, pero si queremos que nuestra consulta de recuperación de datos se guarde en la propia base de datos y se comporte como una tabla (algo parecido a una consulta almacenada de Access), la tenemos que definir como una vista. Esta vista tiene la ventaja entre otras de poder ser utilizada como si fuese una tabla en otras consultas. Realmente al ejecutarla obtenemos una tabla lógica almacenada en memoria y lo que se guarda en la base de datos es su definición, la instrucción SQL que permite recuperar los datos.





Para definir una vista en el Explorador de Objetos desplegamos la base de datos donde la guardaremos y elegimos la opción Nueva vista del menú contextual de la carpeta Vistas, se pondrá en funcionamiento el generador de consultas pidiéndonos las tablas en las que se basará la vista. Pulsamos sobre la tabla a añadir al diseño de la vista y pulsamos el botón Agregar, podemos añadir así cuántas tablas queramos.

Después de Cerrar, vemos a la derecha del Explorador de Objetos la pestaña con la definición de la vista que puede incluir varios paneles:

Columna	Alias	Tabla	Resultados	Tipo de orden	Criterio de ord...	Filtro	O...	O...
*		Factura	<input checked="" type="checkbox"/>					

```

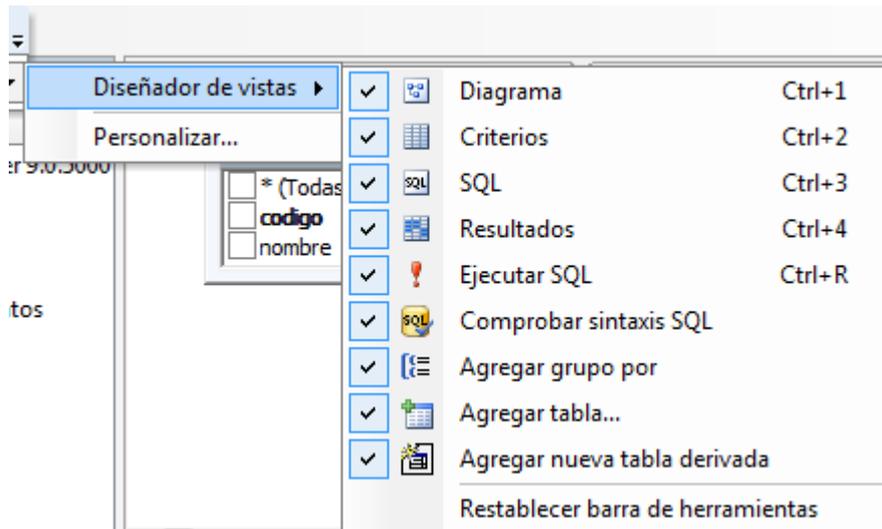
SELECT  dbo.Factura.*
FROM    dbo.Cliente INNER JOIN
        dbo.Factura ON dbo.Cliente.codigo = dbo.Factura.Cliente
    
```

Codigo	fecha	Cliente
20	15/06/1991 12:...	1
21	14/06/2008 12:...	2

1 de 2 Celda de sólo lectura.

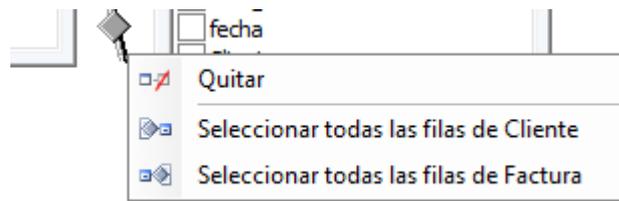
La aparición de estos paneles es configurable, en la barra de herramientas Diseñador de vistas los iconos remarcados en azul son los correspondientes a cada panel:





EL PANEL DE DIAGRAMA

Es el primero que aparece, incluye una representación gráfica de las tablas con sus campos y de la forma en que se juntan en la vista. En este caso, como las tablas tienen relaciones definidas (claves ajenas), esta relación ha aparecido automáticamente al añadir la segunda tabla. Pero se puede cambiar el tipo de relación eligiendo la opción correspondiente en el menú contextual que aparece con el clic derecho sobre la relación:



Desde el panel diagrama podemos añadir cómodamente campos de las tablas a la consulta marcando la casilla correspondiente. En la imagen anterior la única casilla seleccionada es la del * en la tabla *Libros* por lo que se visualizarán todas las columnas de la tabla *Libros* y ninguna de la tabla *Préstamos*. Conforme vamos marcando casillas de las tablas del panel diagrama, los cambios se ven reflejados en los demás paneles excepto en el panel de resultados que se actualiza ejecutando la consulta.

EL PANEL DE CRITERIOS

Es una rejilla en la que podemos definir las columnas del resultado de la consulta (las columnas de la vista).

Columna	Alias	Tabla	Resultados	Tipo de orden	Criterio de ord...	Filtro	O...	O...
codigo	codCie...	Cliente	<input checked="" type="checkbox"/>					
nombre		Cliente	<input checked="" type="checkbox"/>					
Codigo	Expr1	Factura	<input checked="" type="checkbox"/>					
fecha		Factura	<input checked="" type="checkbox"/>					

- En cada fila de la rejilla se define una columna del resultado o una columna que se utiliza para obtener el resultado.
- En *Columna* tenemos el nombre de la columna de la se obtienen los datos o la expresión cuando se trata de una columna calculada.





- En *Alias* escribimos el nombre que tendrá la columna en la vista, también corresponde con el encabezado de la columna en la rejilla de resultado. Si se deja el campo en blanco, por defecto se asume el mismo nombre que hay en *Columna*.
- En *Tabla* tenemos el nombre de la tabla del origen de la consulta a la que pertenece la *Columna*, por ejemplo la primera columna del resultado se saca de la columna *Codigo* de la tabla *CLIENTE* y se llamará *CodCliente*. La cuarta columna de la vista cogerá sus datos de la columna *Usuario* de la tabla *Prestamos* y se llamará *Usuario* (*Alias* se ha dejado en blanco por lo que asume el nombre que hay en *Columna*).
- En la columna *Resultados* indicamos si queremos que la columna se visualice o no, las columnas con la casilla marcada se visualizan.
- Las columnas *Criterio de ordenación* y *Tipo de orden* permiten ordenar las filas del resultado según una o más columnas. Se ordena por las columnas que tienen algo en *Tipo de orden* y cuando se ordena por varias columnas *Criterio de ordenación* indica que primero se ordena por la columna que lleva el nº 1 y después por la columna que lleva el nº 2 y así sucesivamente.

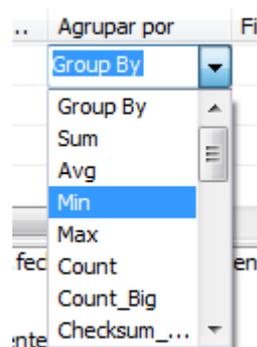
Si queremos añadir unos criterios de selección tenemos las columnas *Filtro* y *O...*

En cada celda indicamos una condición que debe cumplir la columna correspondiente y se puede combinar varias condiciones mediante O (OR) e Y (AND) según coloquemos las condiciones en la misma columna o en columnas diferentes. En el ejemplo anterior tenemos la condición compuesta: ((usuario=1) AND (Dias>5)) OR (Usuario=2).

Podemos variar el orden de aparición de las columnas arrastrando la fila correspondiente de la rejilla hasta el lugar deseado.

También podemos Elimnar filas de la rejilla para eliminar columnas del resultado, lo conseguimos seleccionando la fila haciendo clic sobre su extremo izquierda y cuando aparece toda la fila remarcada pulsamos Supr o desde el menú contextual de la fila.

Podemos definir consultas más complejas como por ejemplo consultas de resumen, pulsando sobre el botón Agrupar por de la barra de herramientas, se añade a la rejilla una nueva columna Agrupar por con las siguientes opciones:



EL PANEL SQL

En él vemos la instrucción SQL generada, también podemos redactar directamente la sentencia SQL en el panel y ver los cambios equivalentes en los distintos paneles. Para ver estos cambios debemos de ejecutar o Comprobar la sintaxis para que se actualicen los demás paneles.

Por defecto el generador añade a la consulta una cláusula TOP (100) PERCENT que indica que se

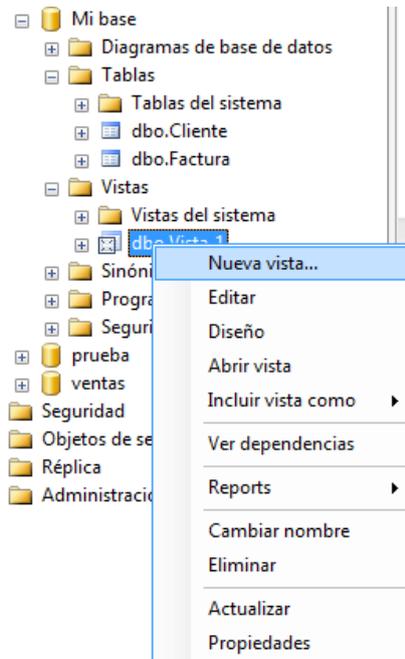




visualizarán el 100% de las filas. Esta cláusula no la hemos definido nosotros sino que la añade automáticamente el generador.

```
SELECT dbo.Cliente.codigo AS codCliente, dbo.Cliente.nombre, dbo.Factura.Codigo AS Expr1, dbo.Factura.fecha, dbo.Factura.Cliente
FROM   dbo.Cliente INNER JOIN
       dbo.Factura ON dbo.Cliente.codigo = dbo.Factura.Cliente
```

Una vez tenemos la vista definida la guardamos y podremos hacer con ella casi todo lo que podemos hacer con una tabla. De hecho si nos fijamos en el Explorador de objetos, en la carpeta Vistas:



Vemos que la estructura es muy similar a la estructura de una tabla. Y que podemos modificar su definición y ejecutarla, igual que con las tablas:

- Modificar para modificar la definición de la vista.
- Abrir vista para ejecutarla y ver los datos como si fuese una tabla real.

