

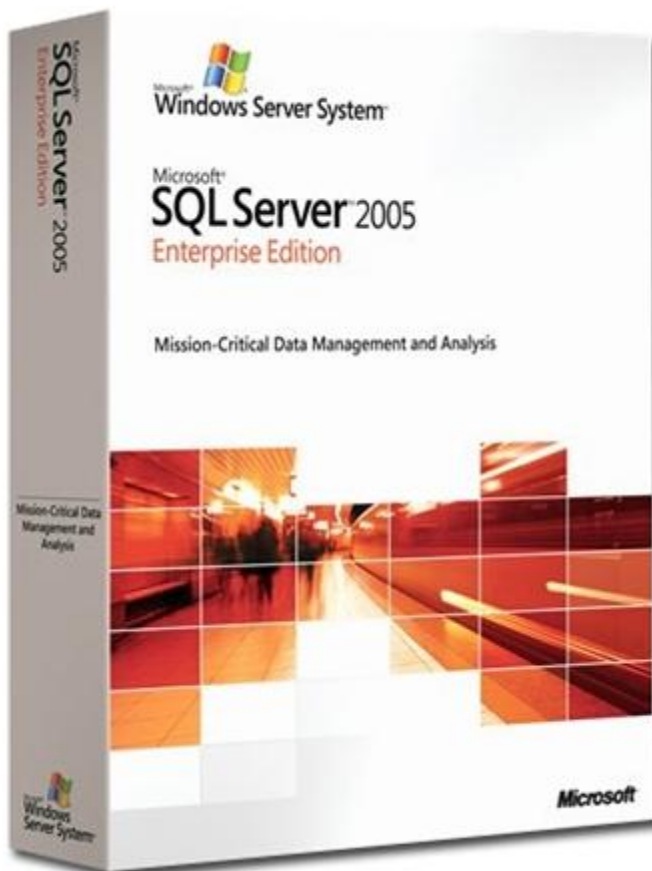
Curso SQL Server 2005

Nº- 1 Presentación del curso

La finalidad que buscamos en este curso podemos dividirla en dos objetivos:

1. Dominio del lenguaje SQL.
2. Administración de bases de datos.

Tanto para aprender y practicar con el lenguaje SQL, como para administrar y mantener una base de datos nos apoyaremos en el servidor de base de datos Microsoft SQL Server 2005.



Al finalizar el curso, el alumno será capaz de implementar sentencias SQL para realizar las más complejas consultas y sentencias de manipulación de datos. Por otro lado, tendrá los conocimientos necesarios para comenzar a administrar y mantener una base de datos empresarial mediante SQL Server 2005.

Como iremos viendo, SQL Server 2005 tiene una cantidad enorme de herramientas, tanto de cliente como de servidor, que nos permiten un control absoluto sobre nuestros datos. Para seguir este curso con eficacia no es necesario unos conocimientos previos en bases de datos, ni en programación SQL, ya que el curso comienza desde cero hasta alcanzar un nivel suficiente.

Tal y como habrás visto en la presentación previa del curso, veremos también el lenguaje T-SQL, lenguaje basado en SQL pero específico de Microsoft que nos permitirá diseñar código con mayores posibilidades de lo que ofrece SQL. De igual modo, tampoco es necesario conocimientos de programación ya que iremos viendo todo desde un principio.

Por lo tanto este curso pretende que los alumnos que se inician en el mundo de las bases de datos sean capaces de diseñar y administrar una base de datos y desenvolverse con soltura en estos entornos. Para aquellos alumnos que ya tengan nociones de bases de datos o lenguaje SQL, afiancen y amplíen esos conocimientos y puedan realizar las principales tareas de administración de uno de los servidores preferidos por muchas empresas, SQL Server 2005. Por otro lado, este curso esta orientado también a aquellas personas que se dedican al desarrollo de aplicaciones informáticas, tanto páginas web, intranets y programas de escritorio, con el aprendizaje de SQL y la administración de servidores de datos, comprenderán mejor el enlace de sus aplicaciones con las bases de datos y serán capaces de separar el desarrollo de sus aplicaciones de la capa de negocio que supone la parte de la base de datos, mediante el lenguaje T-SQL podrán incluir objetos que realicen tareas que solucionan cantidad de problemas que se plantean durante el desarrollo de aplicaciones, mejorando enormemente la eficacia y la seguridad de las aplicaciones.

Seguro que habrá programadores que desarrollan las conexiones a sus bases de datos mediante un único usuarios, otro que las realizan mediante el usuario 'sa'... Todos estos casos ponen en peligro la seguridad de los datos vitales de una empresa, y es motivo suficiente para animar a los alumnos a que realicen este curso.

Con la aparición de la informática, las empresas son capaces de gestionar los mismos datos en unas horas que lo que antes gestionaban durante meses. Según se han ido modernizando las características de hardware y software, cualquier empresa puede cubrir la necesidad del control de información de gran valor para su desarrollo y crecimiento de un modo sencillo y rentable para el resultado que obtienen.

El propio sistema de control de datos ha ido mejorando con el tiempo, desde las primeras aplicaciones que gestionaban su propia información alojándolas en unidades de almacenamiento externas (discos duros, disquetes, cintas...) con el problema de que sólo esa aplicación era capaz de interpretar y utilizar esa información. Más adelante comenzaron a imponerse unos sistemas de almacenamiento estándar que facilitaba la tarea de compartir esa información entre diferentes aplicaciones.

Por fin aparecieron los servidores de bases de datos, herramientas cuya única y principal función era la administración de información. Mediante diferentes protocolos de comunicación las diferentes aplicaciones pueden enlazarse con estos servidores, ordenar las tareas que necesiten para que el servidor se encargue de operar esas tareas y devolver los resultados deseados.

Podemos tener nuestra herramienta de administración de datos en un ordenador (Servidor) y que el resto de ordenadores (Clientes) se conecten a este servidor mediante sus aplicaciones para trabajar con estos datos, esta estructura Servidor/Cliente es la que se ha terminado de imponer pudiendo distinguir claramente tres niveles o capas de trabajo:

Capa	Descripción
Aplicaciones informáticas.	Estas aplicaciones pueden ser desde páginas web a aplicaciones de escritorio, encargadas de ofrecer interfaz de usuario para presentar la información y ofrecer la posibilidad de realizar operaciones al usuario.
Lógica de negocio.	Diferentes objetos diseñados (que más adelante aprenderemos a crear y utilizar) para operar con nuestros datos.
Administrador de bases de datos.	Servidor que se encargará de administrar y ejecutar las tareas que se le encarguen para gestionar esos datos. De este modo el programador puede dedicarse a su propia aplicación, encargando las tareas de administración al servidor de datos. El servidor recibirá esas ordenes o tareas en forma de instrucciones en lenguaje SQL generalmente, o de un modo más avanzado y con mayores posibilidades con lenguaje TSQL de Microsoft para servidores SQL Server. En este curso aprenderemos ambos lenguajes, abriendo un abanico de posibilidades que cumplan con cualquier necesidad que se plantee a administradores y desarrolladores.

Es común referirse a los servidores de datos como RDBMS, siglas de Relational DataBase Management System. Es el modo más común y también más correcto de nombrarlos.

Una persona encargada de administrar una base de datos tiene como misión no sólo controlar la base de datos empresarial, sino también de aconsejar, asesorar a los desarrolladores, usuarios y directiva de la empresa. Por normal general una empresa cuenta con una o varias personas encargadas de controlar el sistema de base de datos, programación, control de sistema operativos, hardware, comunicación, redes, etc...

Podemos decir que la persona que controla la administración de base de datos, tiene las siguientes funciones:

- Diseñar y controlar la estructura de la base de datos.
- Supervisar la actividad sobre los datos.

- Controlar la eficacia de la base de datos.
- Preocuparse de la seguridad de los datos.
- Supervisar el estado del sistema.
- Atender las quejas de usuarios sobre la información que obtienen y la velocidad, y poner remedio a esos problemas.
- Obtener y estudiar las estadísticas del funcionamiento y el rendimiento del sistema.
- Supervisar y conocer en todo momento la actividad que realizan los usuarios sobre los datos.
- Preocuparse de las nuevas actualizaciones que salen al mercado y en caso de ser necesario, estudiar el momento apropiado de actualizar el sistema, o de realizar migraciones de estructuras y datos hacia nuevos sistemas.

2 Microsoft SQL Server 2005

Actualmente podemos encontrarnos con varios servidores de base de datos (RDBMS):

- Oracle
- DB2
- MySQL
- SQL Server
- ...

Todos ellos desempeñan la misma función, pero tienen diferentes propiedades y herramientas que distinguen claramente unos de otros.

SQL Server 2005 es la siguiente edición de su predecesor en el mercado, SQL Server 2000, el cual tuvo muy buena aceptación en las empresas por su alta calidad.

Con esta última versión se ha conseguido mejorar aún mas SQL Server 2000, mejorando la fiabilidad, escalabilidad, rendimiento y manejo. Muchas de las empresas controlan sus redes locales con el conocido sistema operativo Windows Server 2003, siendo uno de los más extendidos y preferidos por la mayoría de empresas. Este sistema operativo de Microsoft es el entorno ideal para la instalación de SQL Server 2005, convirtiéndose en la mejor pareja posible para la administración.

Además la reciente aparición de SQL Server 2005, conlleva que este preparado para la expansión por la red de redes (Internet) ya que por ejemplo es capaz de generar automáticamente documentos XML, se trata del formato estándar de datos que facilita la transmisión de datos en Internet.

Como veremos en el siguiente punto, tenemos diferentes versiones de SQL Server 2005, cada una orientada a cubrir unas determinadas necesidades de diferentes tipos de empresas o clientes, pero podemos enumerar una serie de propiedades comunes para todas ellas, que demuestran que SQL Server es bastante más que un servidor de base de datos:

- Servidor de base de datos, de gran rendimiento.
- RDBMS que pueden ser instalados tanto en sistemas de usuarios como Windows XP, máquinas de multiprocesador de 64 bits, redes de ordenadores.
- La administración se facilita mediante interfaz gráfica de usuario.
- Capaz de tener varias instancias del servido en una única máquina.
- Acceso directo a datos desde página Web, gracias a la generación automática de documentos XML, consiguiendo una completa integración con Internet.
- Posibilidades de data warehousing y data mining, para almacenar y analizar datos, funcionando como Online Transaction Processing (OLTP) y con servicios Online Analytical Processing (OLAP).
- Comunicación perfecta con otras aplicaciones Microsoft, pudiendo presentar información en hojas de Excel, por citar un ejemplo.

- Integración perfecta con herramientas de desarrollo de software como Visual Studio 2005.
- Lenguaje T-SQL para ampliar las posibilidades de las tareas a realizar.
- Capacidad para interpretar funciones realizadas con CLR (Common Language Runtime) de plataformas .NET, esto nos permite realizar funciones en lenguajes muy conocidos como Visual Basic o C#.

2.1 Versiones de SQL Server 2005

Como acabamos de ver SQL Server 2005 tiene una serie de propiedades comunes a las cuales se le añaden una serie de herramientas para ir formando diferentes versiones orientadas a diferentes tipos de empresas y funciones.

Estas ediciones las presentamos a continuación comenzando con la que menos posibilidades ofrece hasta llegar a la más completa:

Versión	Descripción
SQL Server 2005 Express Edition	Versión básica del servidor, limitada en el número de usuarios y en cuanto al volumen de datos a gestionar. Es la versión que Microsoft nos ofrece gratuitamente y sin duda es la versión ideal para comenzar a trabajar y desde la cual podemos ir ampliando a versiones superiores si nos fuese necesario. Permite ser distribuida con programas de desarrollo propios con su correspondiente licencia.
SQL Server 2005 Workgroup Edition	Orientada a pequeñas empresas, no tiene límite de números de usuarios y ni de capacidad de almacenamiento en cuanto al tamaño de la base de dato. No incluye las herramientas avanzadas de las versiones superiores.
SQL Server 2005 Standard Edition	Pensada para empresas de mediano tamaño, cuenta con herramientas avanzadas para la administración y análisis de datos.
SQL Server 2005 Enterprise Edition	Esta edición está preparada para gestionar las empresas de mayor tamaño, ya que ofrece mayor potencia que las anteriores. Y además de añadir servicios avanzados y estar preparada para trabajar con multiprocesadores de 64 Bits, amplias

	<p>memorias RAM. Podemos hablar de bases de datos con tamaños dados en Terabytes, para que puedas hacer una idea de la cantidad de información que puede llegar a gestionar con fiabilidad. Tiene la capacidad de trabajar con Clústers de ordenadores, de modo que el fallo de uno de ellos active otro ordenador que se encontraba pasivo hasta el momento del error.</p>
SQL Server 2005 Developer Edition	<p>Se trata de una versión que cuenta con las mismas características que su versión anterior, con la diferencia de que va dirigida a desarrolladores. ¿Porque si tienen las mismas características, no se trata de la misma edición? Esta versión especial está limitada por su licencia que no permite que se ejecute en entornos de explotación y sólo se permite para desarrollo. Otra diferencia importante, es que permite ser instalada en Sistemas Operativos Windows XP Professional Edition.</p>

3 Requisitos

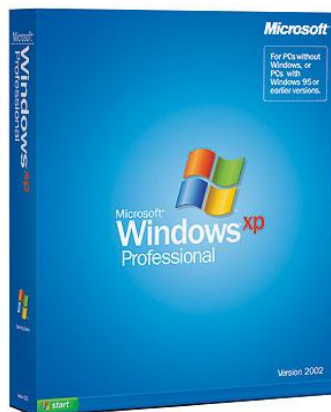
En función de la edición con la que vayamos a trabajar tendremos unos requisitos mínimos que nuestras máquinas deberán cumplir y del mismo modo nuestro software, y especialmente el sistema operativo sobre el que va a trabajar. Debes pensar que no estamos instalando un programa de escritorio, se trata de un potente RDBMS que cuenta con una serie de servicios avanzados, además de accesos desde Internet etc...

3.1 Requisitos de Software

El principal requisito y el más importante que debemos cumplir, es el sistema operativo sobre el que vamos a trabajar. Naturalmente este requisito dependerá de la versión de SQL Server 2005 que queramos instalar. Es lógico pensar que aquellas versiones que están orientadas a trabajar en un entorno empresarial no se podrán instalar sobre versiones de sistema operativo dirigidas a ordenadores personales, como Windows XP Home Edition.



Sin lugar a duda, para un entorno empresarial, con información de un tamaño muy importante, el mejor entorno de trabajo para SQL Server 2005, es Windows Server 2003/R2, última versión de este sistema operativo.



Ahora bien si vamos a trabajar con las versiones Express o Professional que están dirigidas a uso personal, o para desarrolladores, es posible instalarlas bajo Windows XP Professional.

Si tenemos como sistema operativo Windows 2000 será imprescindible instalar el SP4 o superior.

Es imposible que podamos instalar SQL Server 2005 sobre versiones anteriores a Windows XP (98, Me...) y en Windows NT.

Por lo tanto en cuanto a software los requisitos son bastante lógico, tanto que sería muy difícil encontrar una empresa que no los cumpla por pequeña que sea.

En cuanto a poder realizar y seguir el curso, no tendrás ningún problema, ya que es más que seguro que tendrás Windows XP (a ser posible la edición Professional, con SP2). En cuanto a la versión de SQL Server 2005 con la que vamos a seguir el curso, lógicamente será la versión Express, ya que podemos decir que es la versión académica que Microsoft ha decidido ofrecer al público de un modo gratuito.

Con esta versión tendremos todo lo necesario para el objetivo de este curso, nos permitirá realizar todas las prácticas que iremos viendo a lo largo del curso. No pienses que por ser la edición menos completa y ser gratuita, esta muy limitada, y nos encontramos antes una especie de demo de prueba del verdadero SQL Server 2005, la versión Express es posible instalarla para trabajar con pequeñas empresas con una licencia de explotación (lo mejor sería aconsejar a la empresa para que invirtiera dinero en una versión superior), lo que trato de decirte es que con esta versión contamos con una herramienta suficientemente potente, que supera con creces las bases de datos de escritorio.

3.2 Requisitos de Hardware

Podemos hablar de unos requisitos mínimos o recomendables de Hardware, hoy en día lo más seguro es que esto requisitos se cumplan con creces en cualquier empresa o incluso en ordenadores personales.



El procesador recomendado, es como mínimo un Pentium III, o de cualquier otro fabricante pero con al menos 1 GHz. En el mercado hoy en día, los PC vienen con Pentium IV como procesador mínimo y AMD Athlon supera las velocidades de estos.

En cuanto a la memoria RAM que debemos disponer, dependerá de la versión que vayamos a instalar. Con la que nosotros trabajaremos (SQL Server 2005 Express Edition) es suficiente con 128 Mb, en cambio para el resto de ediciones como mínimo tendremos 512 Mb, aunque para las versiones estándar y empresarial es más recomendable disponer de 1Gb.



Estos son los requisitos mínimos, pero cuanto mayor memoria tengamos, mejor se aprovecharán las características de SQL Server 2005, sobretodo para ediciones que trabajen con 64 bits y bases de datos de gran tamaño.

La memoria en disco mínima que tenemos que tener varía con las características que vayamos a instalar, ya que como veremos en este mismo capítulo, podremos elegir las herramientas que deseamos instalar.

Aunque no es un requisito obligatorio, el entorno de trabajo de SQL Server 2005 se trabaja de un modo más cómodo con una resolución de 1024x768.

4 Pasos Previos

Antes de comenzar con la instalación de SQL Server 2005 debemos tener en cuenta una serie de requisitos previos a cumplir. Alguno de estos requisitos debemos cumplirlos por seguridad y otros son obligatorios para poder finalizar la instalación correctamente.

Con versiones anteriores a SQL Server 2005, no teníamos la posibilidad de tener varias instancias del servidor de base de datos conviviendo en un mismo PC. Incluso podemos tener en el mismo equipo varias versiones de SQL Server.

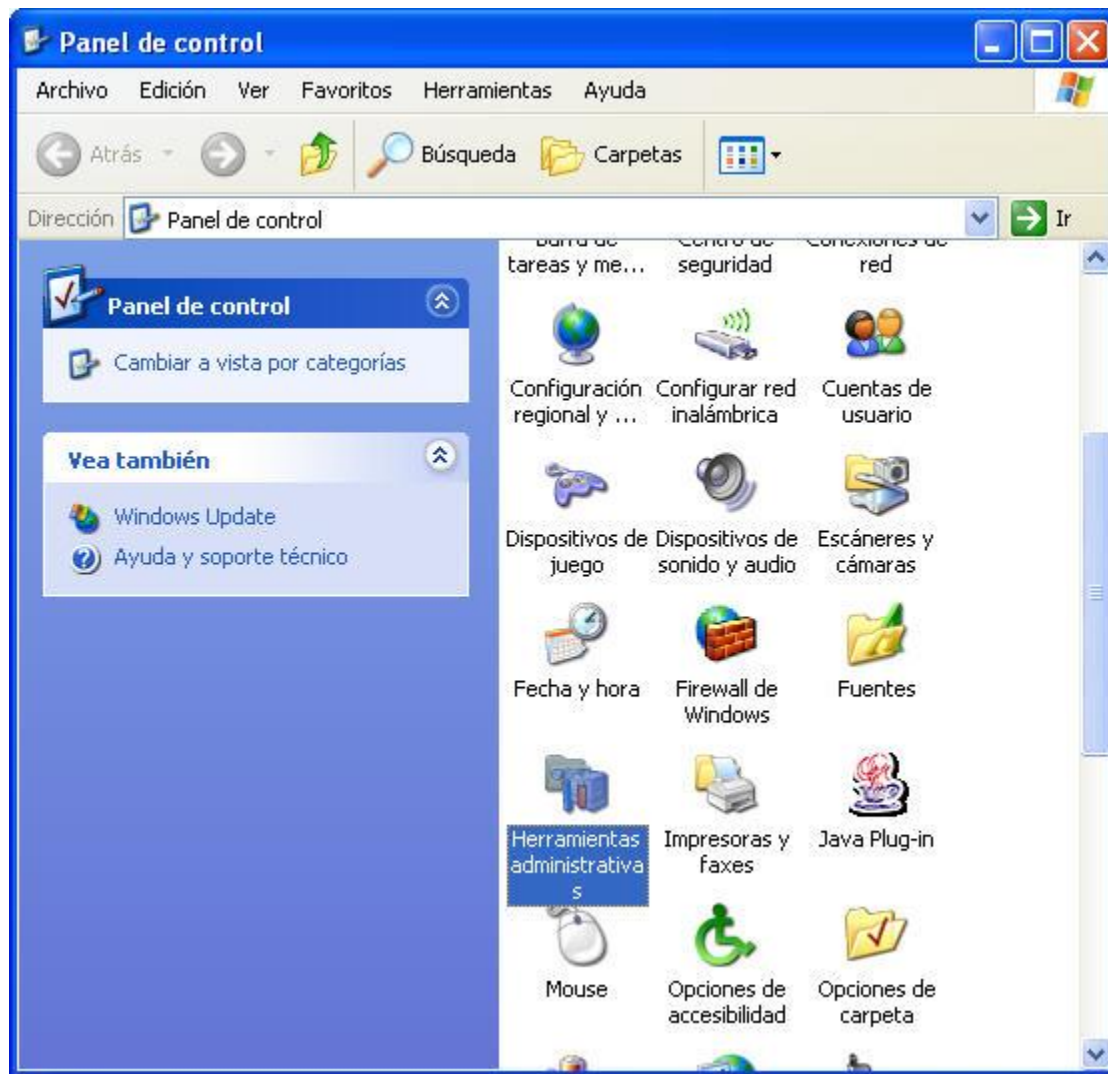
Al hacer una instalación tenemos la posibilidad de sobrescribir sobre una instalación anterior. También tenemos la posibilidad de instalar SQL Server 2005, actualizando una versión anterior. Si vamos a actualizar una versión de SQL Server 2000 tenemos que tener instalado el Service Pack 3 y si es aún más antigua, SQL Server 7, debemos de tener instalado previamente el Service Pack 7 de esta versión. Por lo tanto, revisaremos antes de instalar SQL Server 2005 sobre cualquiera de estas versiones si tenemos estos paquetes y sino es así adquirirlos desde la página de Microsoft.

En cualquier caso, si vamos a realizar una instalación sobre otra versión es más que recomendable realizar una copia de seguridad de toda la información, y sobre todo de nuestras bases de datos, para evitar problemas y sorpresas.

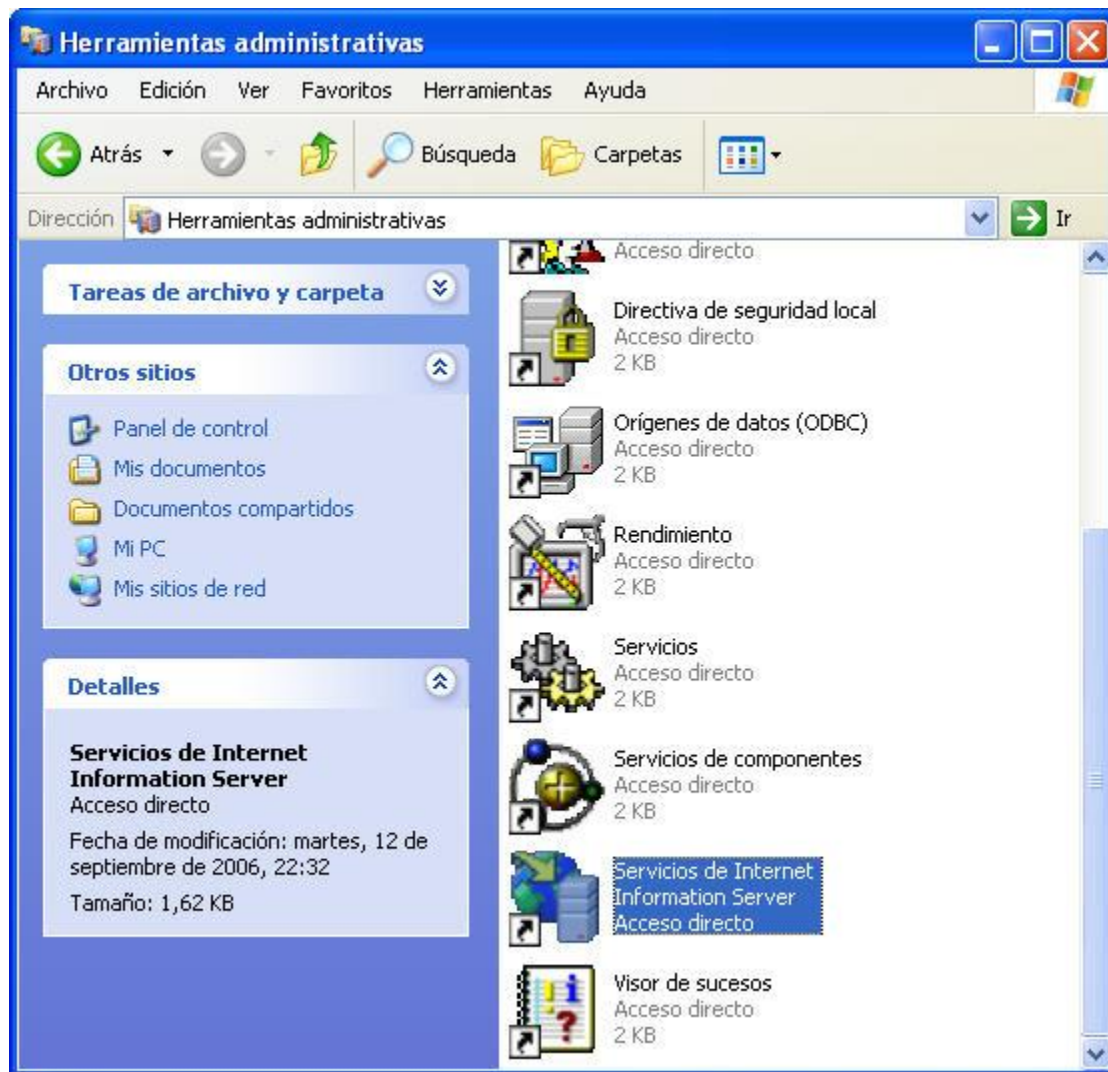
Si vamos a realizar la instalación en un servidor con Windows 2000 Server o Windows 2003 Server, es muy probable que tengamos instalado y en funcionamiento el Servidor de páginas web, IIS (Internet Information Server). En ese caso debemos parar este servicio antes de proseguir con la instalación. Para llevar a cabo esta tarea, abrimos el panel de control:



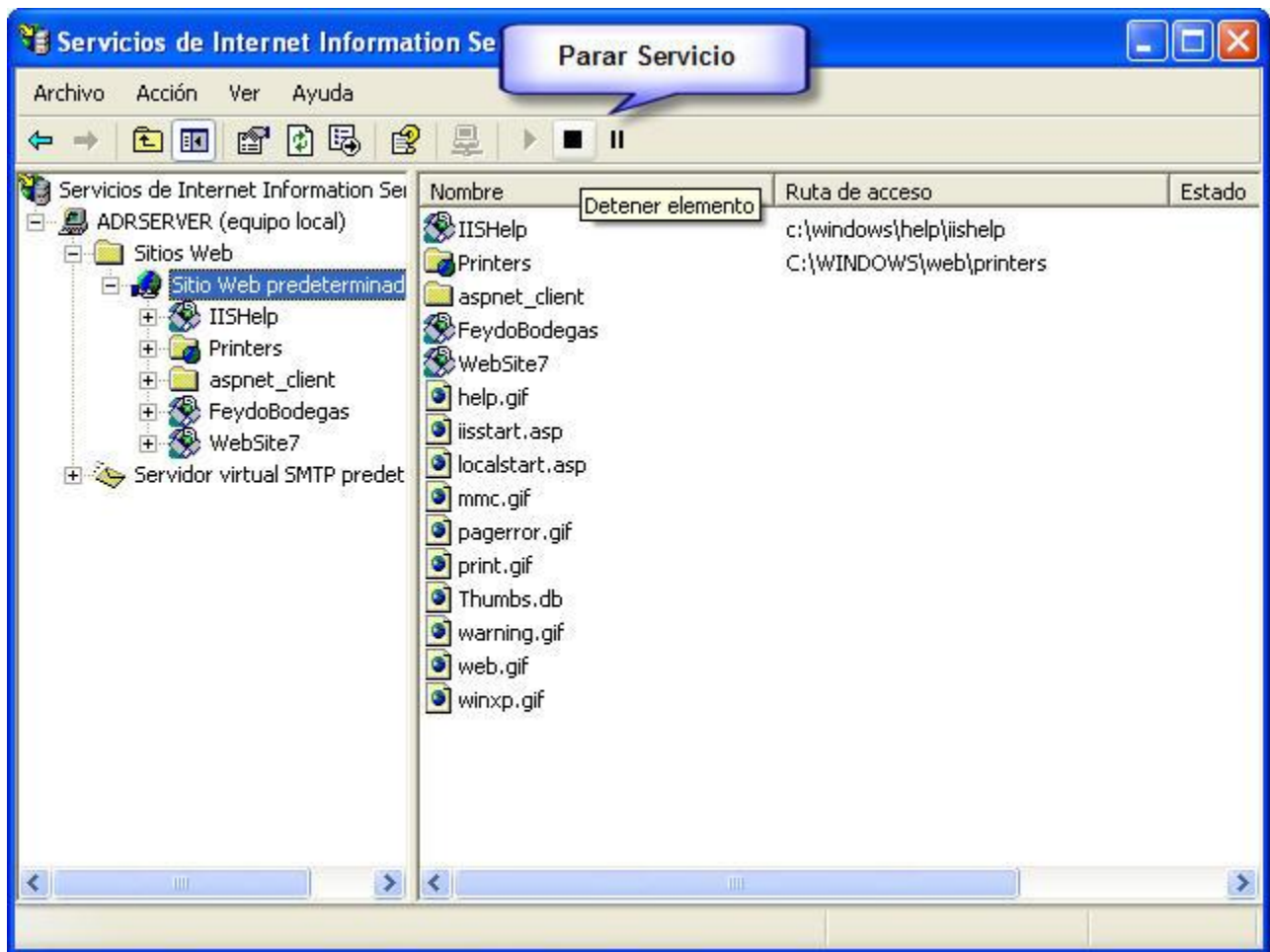
Entre los iconos del panel de herramientas seleccionamos "Herramientas Administrativas":



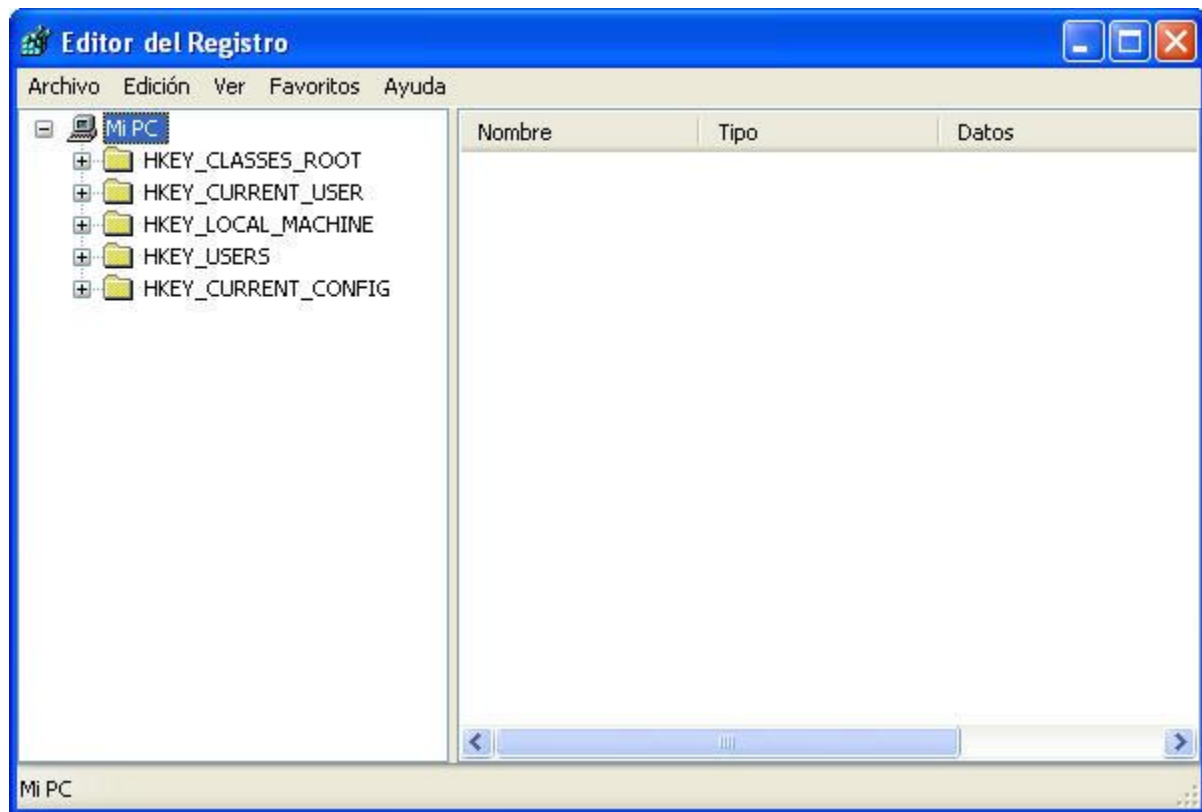
Esto nos abre una ventana con las herramientas administrativas que tenemos instaladas en nuestro equipo, siempre y cuando tengamos instalado los servicios de IIS encontraremos el icono:



Ejecutamos la herramienta y se nos presenta el panel de Internet Information Server, desplegamos el icono de nuestro servidor o pc, hasta encontrarnos con los sitios web que tenemos ejecutando en el servidor dentro de "Sitio Web predeterminado", con este seleccionado pulsamos en el botón destinado a parar el servicio como puedes ver en la siguiente figura:



Además de para el servicio de servidor de páginas web debemos parar también el visor de sucesos y el editor de registros de Windows:



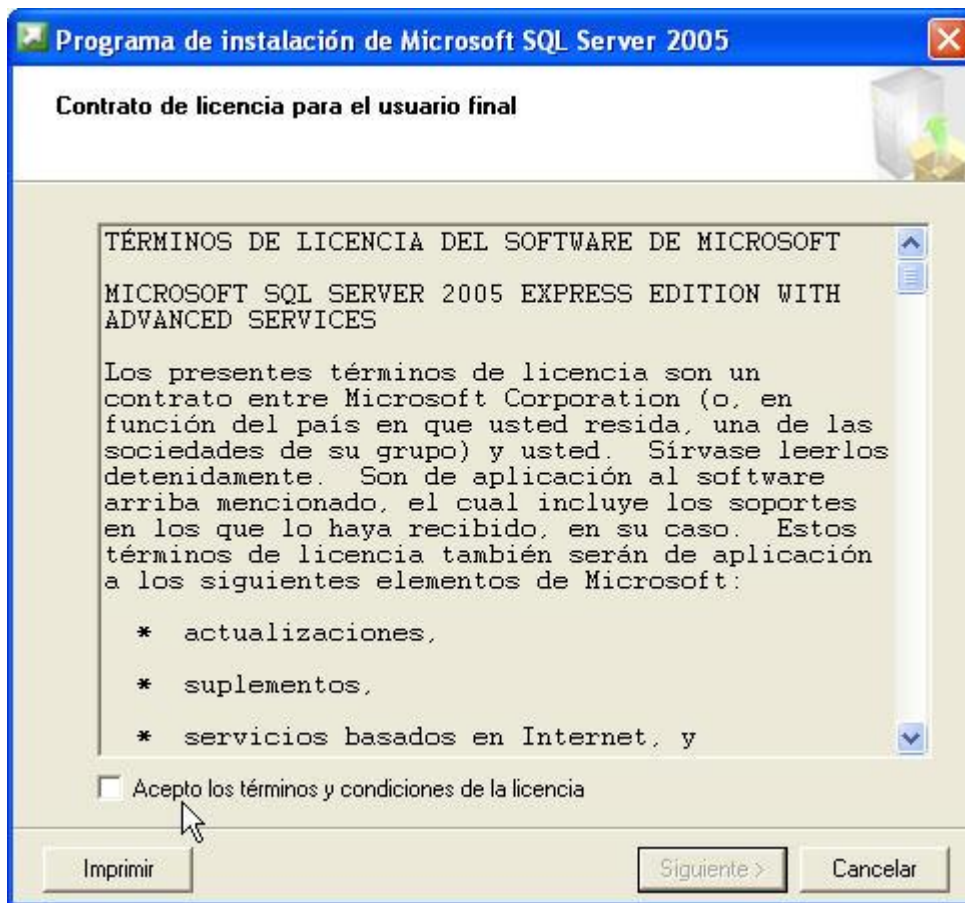
Para concluir, si no estamos instalando SQL Server 2005 como una copia personal para realizar pruebas o desarrollar, y lo estamos haciendo para explotar la base de datos en un servidor al que tendrán acceso clientes y usuarios, deberemos lo primero iniciar sesión en el servidor como Administrador para instalar SQL Server. El siguiente paso será crear las correspondientes cuentas de usuario para los clientes de SQL Server.

5 Instalación

Explicaremos la instalación de la versión gratuita que será con la que trabajaremos en este curso, desde que se lanzó al mercado SQL Server 2005, han añadido una versión más avanzada a SQL Server 2005 Express Edition, que por suerte incluye alguna herramienta más avanzada que la que en un principio publicaron. La versión se denomina Microsoft SQL Server 2005 Express Edition With Advanced Services. Para descargarte esta versión pincha en el siguiente enlace:

[Microsoft SQL Server 2005 Express Edition with Advanced Services](#)

Ejecutamos el programa de instalación y tras descomprimir en nuestro equipo los paquetes necesarios para llevar a cabo la instalación, lo primero que tenemos es el contrato de licencia.



Aceptamos las condiciones y proseguimos con el asistente. En la siguiente ventana nos indica los componentes que se van a instalar:



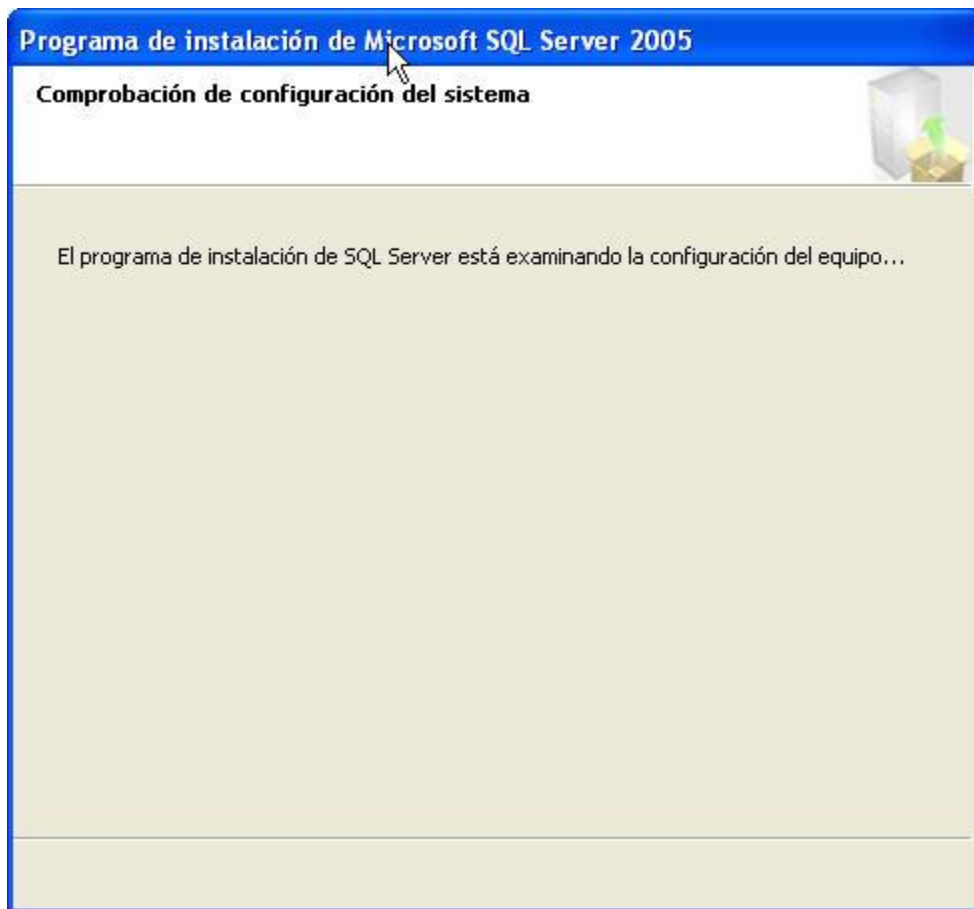
Pinchamos en instalar, y comienza el proceso de instalación de los componentes auxiliares que SQL Server 2005 necesita para instalar el producto final:



Esperamos unos minutos hasta que el proceso de instalación previo finaliza:



Una vez finalizado, pulsamos en siguiente, y el asistente comprobará la configuración de nuestro equipo para comprobar que cumplimos con unas características mínimas:



Una vez comprobada la configuración, comienza el asistente de instalación real de SQL Server 2005:




Pulsamos en siguiente, y vuelve a realizar una nueva configuración, esta vez de la configuración del sistema:



Una vez que el asistente comprueba que todo está correcto, pulsamos en siguiente, donde el asistente sigue recogiendo los paquetes necesarios para la instalación:



Cuando finaliza la extracción de ficheros, nos pide la información de registro de nuestro equipo, concretamente nuestro nombre y compañía.



Programa de instalación de Microsoft SQL Server 2005

Información de registro

La siguiente información se utilizará para personalizar la instalación.

Debe rellenar el campo Nombre para poder continuar. El campo Compañía es opcional.

Nombre:
Javier

Compañía:
ADR

☒ Ocultar opciones de configuración avanzadas

Ayuda < Atrás Siguiete > Cancelar

Introducimos estos valores personales y continuamos. En la siguiente pantalla debemos seleccionar los elementos que deseamos instalar, debemos marcar todas las opciones, incluidos los componentes de clientes que incluyen las herramientas que utilizaremos para administrar las bases de datos mediante SQL Server Management Studio Express:



Seguimos con el asistente, y nos pide el modo de autenticación para conectar con SQL Server. Tenemos dos modos de autenticación:

- Modo windows.
- Modo mixo (Windows y SQL Server)

De estas dos opciones, como veremos más adelante, la más segura de toda es utilizar autenticación windows, ya que de este modo en programas clientes que conecten con nuestro servidor evitamos el envío de información privada por la red, que puede ser interceptada y utilizada para dañar nuestro sistema. De todos modos veremos más adelante que podemos añadir diferentes modos de registro para la conexión con el servidor de base de datos. Por lo tanto seleccionamos "Modo de autenticación de Windows" y continuamos.

Programa de instalación de Microsoft SQL Server 2005

Modo de autenticación

El modo de autenticación especifica la seguridad utilizada para la conexión con SQL Server.

Seleccione el modo de autenticación que se utilizará para la instalación.

☒ Modo de autenticación de Windows

☐ Modo mixto (autenticación de Windows y autenticación de SQL Server)

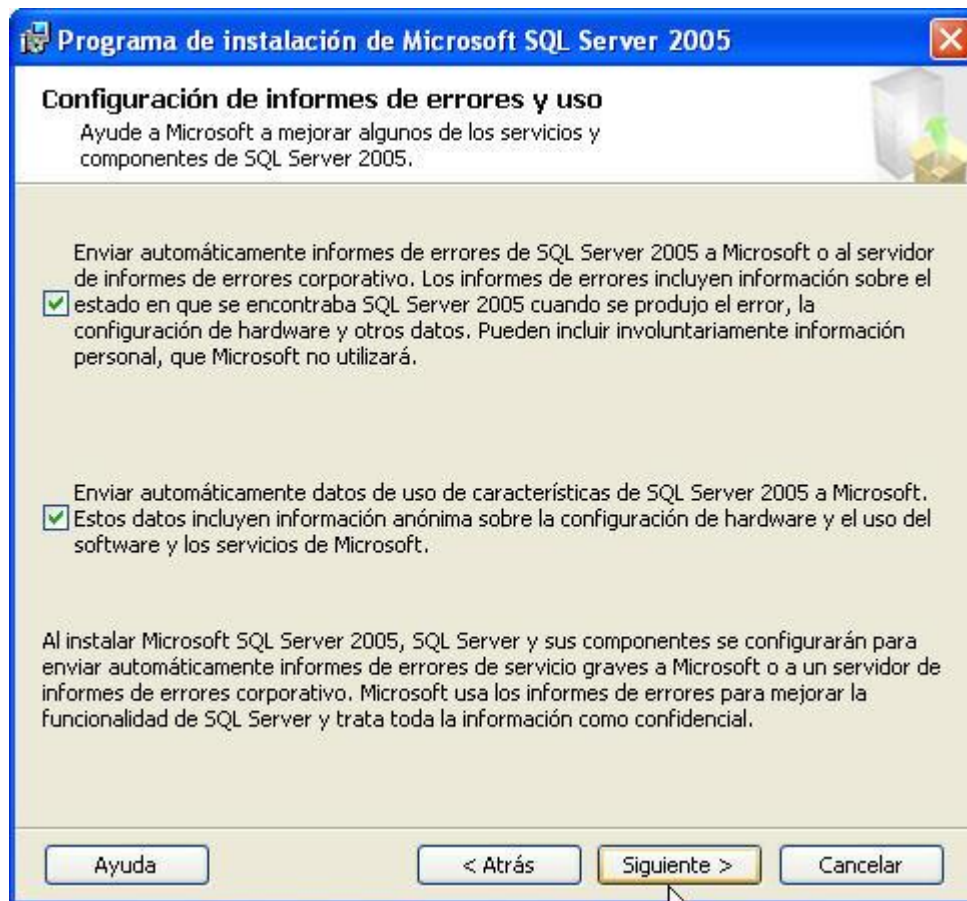
Especifique a continuación la contraseña de inicio de sesión de sa:

Escribir contraseña:

Confirmar contraseña:

Ayuda < Atrás Siguiete > Cancelar

Continuando con el asistente nos pide "permiso" para enviar a Microsoft los errores que puedan darse sobre el servidor de base de datos, e informes del uso que hacemos de las herramientas, todo esto para que Microsoft tenga datos para llevar a cabo actualizaciones para solucionar errores o mejoras para el rendimiento.

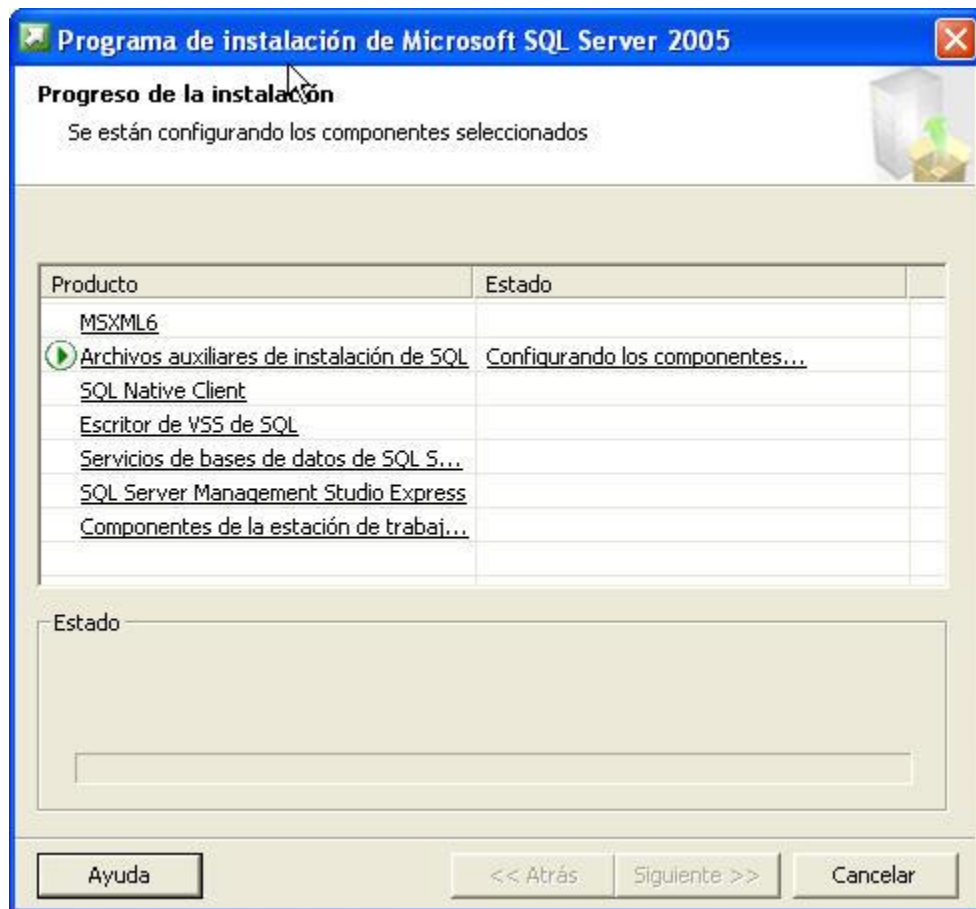


En nuestro caso, no queremos que nuestro servidor dedique recursos al envío de esta información. Además de que seguramente a nuestra empresa no le interese que exista la posibilidad de enviar información personal por error, por lo tanto desmarcamos estas casillas y continuamos.

En la siguiente venta, nos muestra los componentes y las herramientas que se van a instalar a continuación.



Pulsamos en instalar y comienza el proceso de instalación, mostrando información del estado en que se encuentra cada uno de los productos que se están instalando.



Cuando finaliza la instalación para cada uno de los productos, podemos continuar con el asistente:



Pulsamos en siguiente, y nos muestra la pantalla de finalización donde tenemos la posibilidad de ver informes sobre el proceso de instalación. Además nos informa de configuraciones e instalaciones que se han producido en el proceso:



Pulsamos en finalizar y el proceso habrá finalizado correctamente.

Para realizar la primera prueba, vamos a inicio programas y ejecutaremos "SQL Server Management Studio Express":



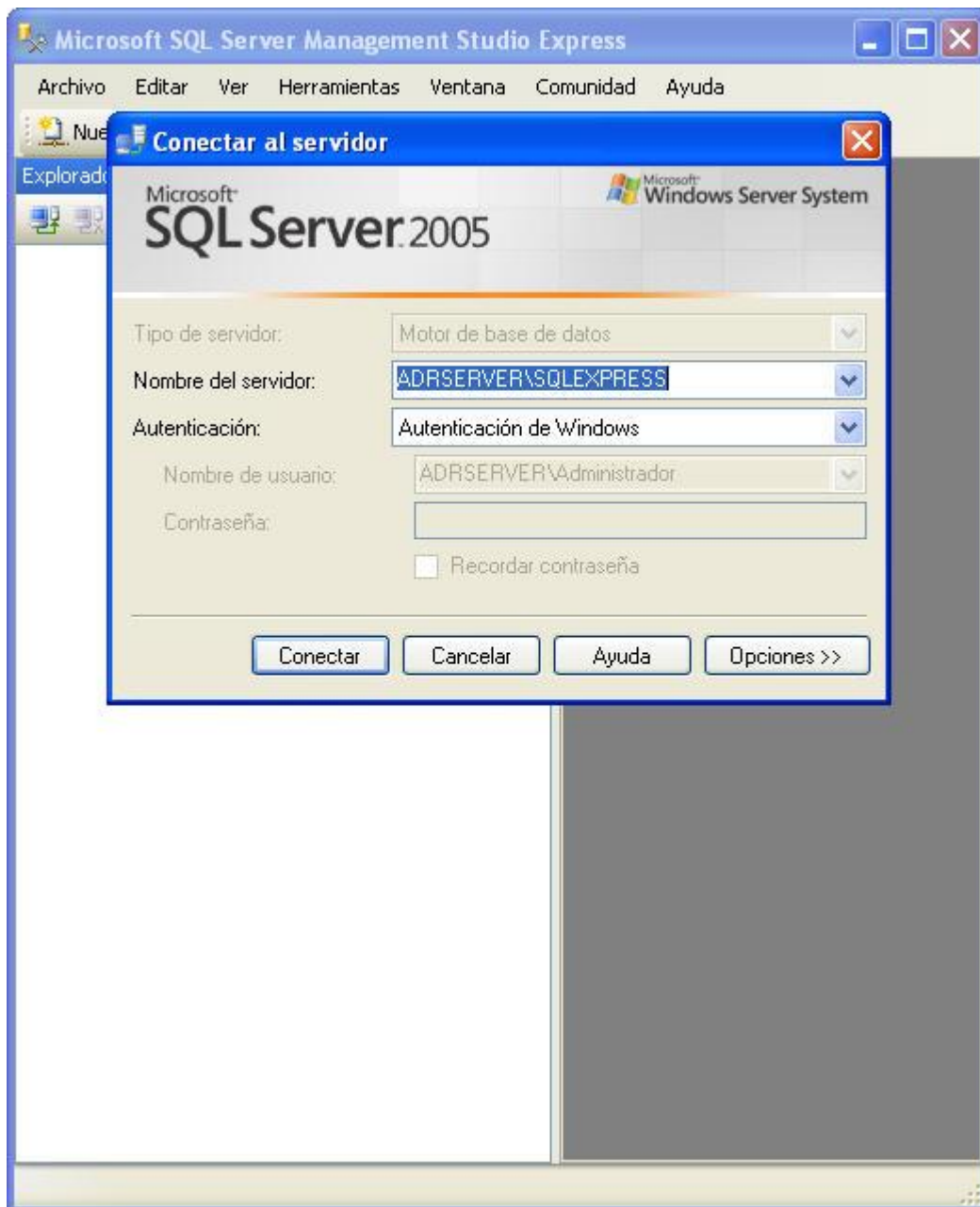
La primera vez que se ejecuta realiza una configuración previa del entorno de trabajo:



Una vez finalizada esta configuración, nos muestra la pantalla de presentación mientras que trabaja en segundo plano para lanzar la herramienta:

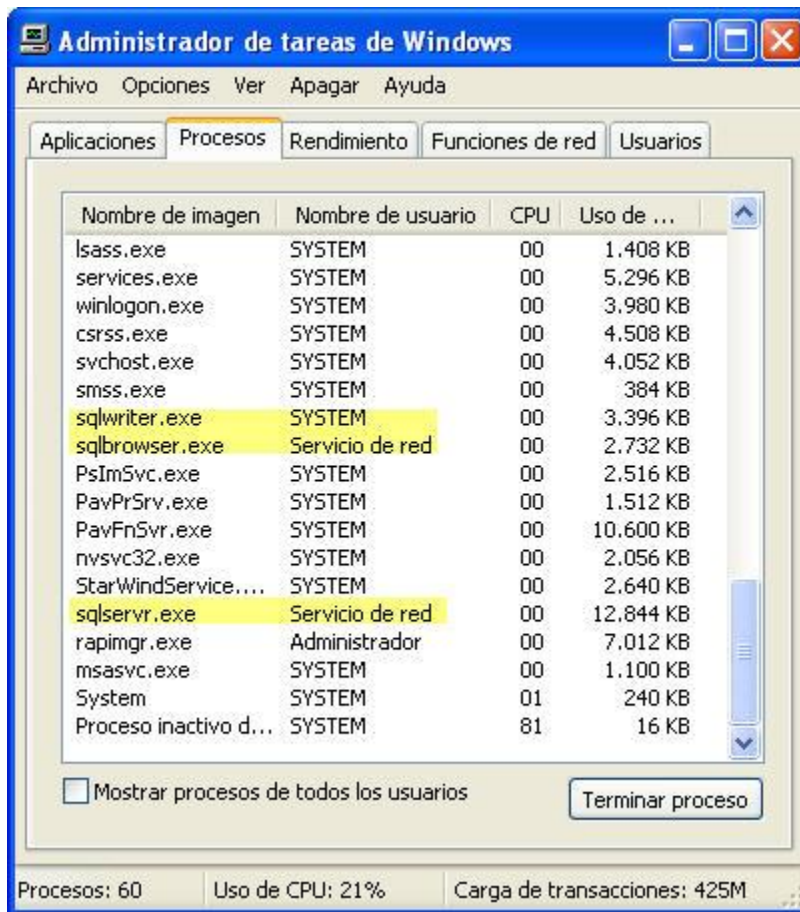


Lo primero que nos muestra cuando arrancamos SQL Server 2005, es una ventana para seleccionar el modo de autorización con el que vamos a conectar al servidor:



Para terminar de comprobar que SQL Server se ha instalado y configurado correctamente en el equipo, podemos mostrar el administrador de tareas de Windows y comprobar en la pestaña de procesos que tenemos los servicios de SQL Server en funcionamiento. Algunos de los procesos de SQL Server se configuran para ejecutarse automáticamente con el sistema operativo al iniciarse Windows, de este modo el servidor queda preparado para que los programas clientes,

páginas Web, etc...estén completamente funcionales para llevar cualquier actividad sobre nuestras bases de datos. Incluso las tareas que tengamos desarrolladas sobre el servidor estarán disponibles, como pueden ser los planes de mantenimiento, y copias de seguridad.



Si has llegado hasta este punto, enhorabuena, hemos conseguido instalar correctamente SQL Server 2005 en nuestro equipo y ya tenemos todo lo necesario para comenzar a practicar con las características de este servidor de base de datos.

6 SQL Server Configuration Manager

Con la instalación de SQL Server, hemos instalado dos herramientas:

- SQL Server Management Studio.
- SQL Server Configuration Manager.

SQL Server Management Studio es una potente herramienta utilizada por desarrolladores y administradores para trabajar y gestionar bases de datos. Esta herramienta la iremos estudiando a lo largo de todo el curso.

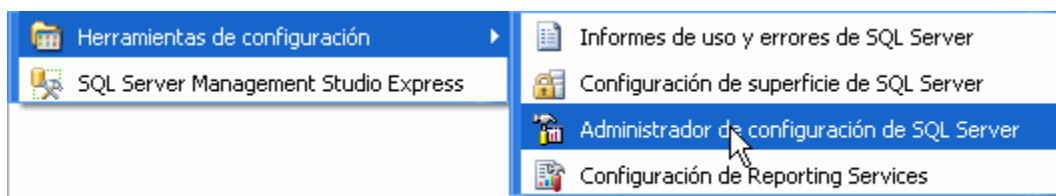
SQL Server Configuration Manager es una herramienta muy útil para trabajar con los servicios de SQL Server.

Como ya has visto, SQL Server se ejecuta como si fuese un servicio, que puede ponerse en marcha de modo automático junto con el sistema operativo en caso de que lo indiquemos así en el momento de la instalación.

El servicio recibe el nombre de MSSQLServer por defecto, este nombre puede variar si instalamos diversas instancias con diferente nombre.

No sólo tenemos el servicio MSSQLServer, tenemos otros como puede ser el MSDTC (Microsoft Distributed Transaction Coordinator) o coordinador de transacciones distribuidas, también tenemos el Agente de SQL Server o SQL - ServerAgent, entre otros...

Para acceder a la herramienta de configuración, lo tenemos disponible en el grupo de programas de Microsoft SQL Server 2005, dentro del grupo Herramientas de configuración, encontramos la opción SQL Server Configuration Manager:



La ventana que nos muestra esta herramienta podemos dividirla en diferentes zonas.

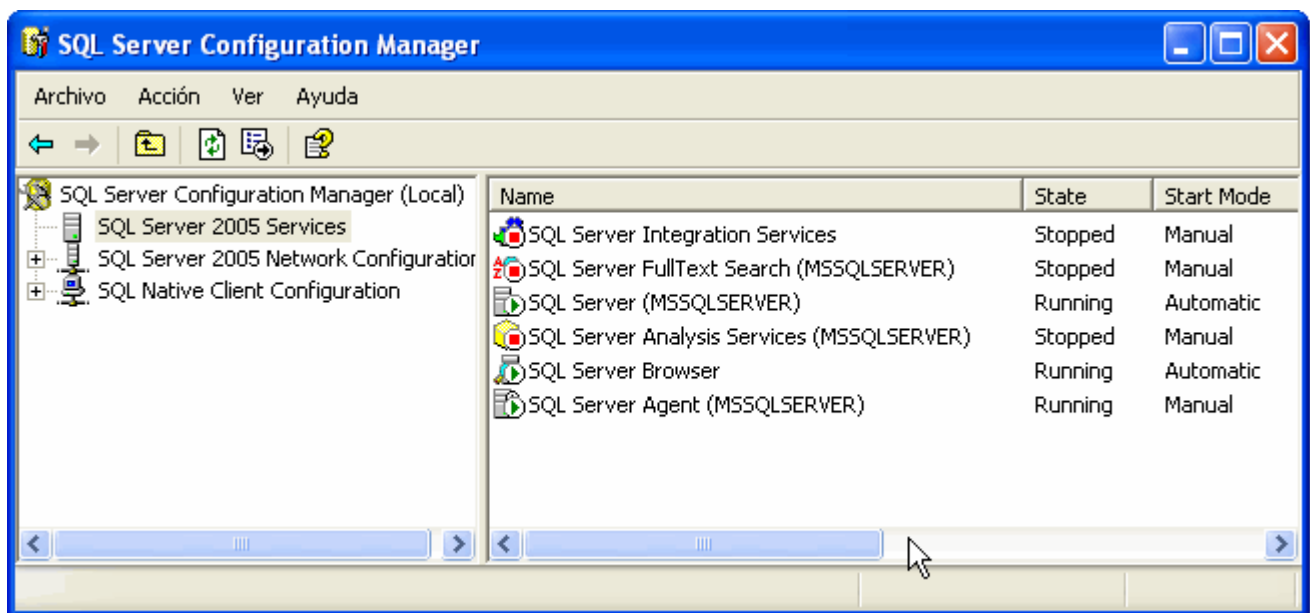
En el panel izquierdo, encontramos varios elementos, que a su vez contienen más subelementos. Por defecto aparece seleccionado "Servicios de SQL Server 2005".

Al tener seleccionado este elemento, en el panel de la derecha, aparecen como una lista todos los servicios que hay instalados, el estado y una serie de características.

El estado podemos comprobarlo también gracias al icono que aparece junto al nombre del servicio, un cuadradillo rojo indica que el servicio se encuentra detenido, mientras que un triángulillo verde indica que el servicio está activo.

Mediante la barra de tareas que aparece sobre este panel, podemos realizar cambios sobre el estado de los servicios, entre otras cosas.

En función de la edición de SQL Server instalada tendremos una lista de servicios mayor o menor, además estos servicios se pueden elegir en el momento de la instalación (servicios de informes, análisis, integración, etc...)

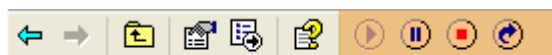


6.1 Modificar el estado de los servicios.

Tenemos tres diferentes estados para cada uno de los servicios:

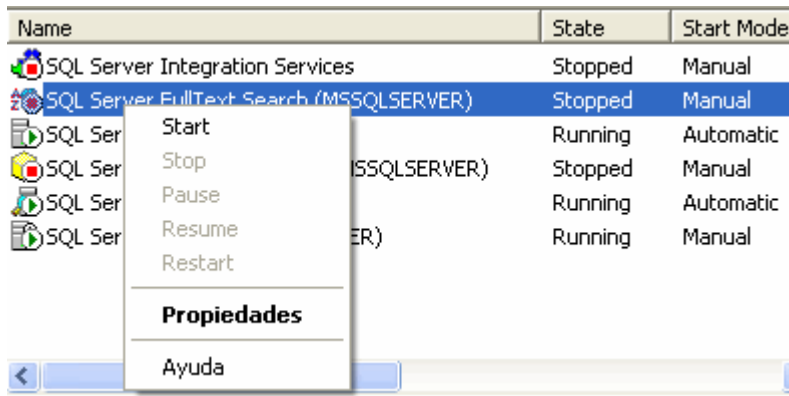
- iniciado: El servicio está en funcionamiento, los clientes podrán acceder a la información.
- pausado: El servicio parece estar parado, no responde a las peticiones, se encuentra congelado y no parado, ya que al reanudarlo, continuará en el mismo estado que se encontraba en el momento de pausarlo.
- detenido: El estado se para, y se desaloja deja de ocupar espacio de memoria. Cuando se reinicie, tomará los valores que se indiquen por defecto.

Para variar de un estado en otro, podemos realizarlo con la barra de herramientas que encontramos en la parte superior.



O bien, mediante el menú emergente que se muestra al pulsar con el botón derecho sobre el servicio del cual queremos variar su estado.

En este menú tendremos unas opciones u otras en función del estado en el que se encuentre actualmente.

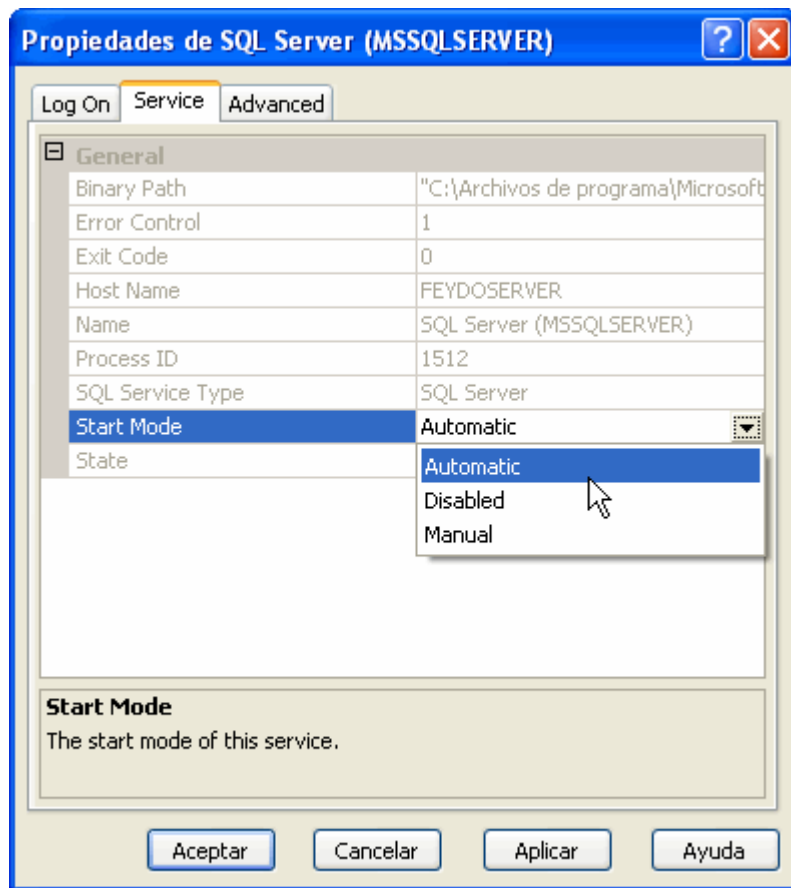


Es muy probable que nos interese que un servicio arranque de modo automático junto con el sistema operativo, de este modo no tendremos que ponerlo en funcionamiento cada vez que se reinicie el servidor, por citar alguna de la ventajas.

Para modificar el modo en el que se inician los servicios debemos pulsar con el botón derecho en uno de los servicios y seleccionar la opción "Propiedades".

En la pestaña servicio, tenemos la opción "Modo de inicio", donde podemos seleccionar los tres modos:

- Automático. Inicia de modo automático el servicio (no es necesario iniciar sesión en el sistema, lo que supone una gran ventaja).
- Deshabilitado. No puede entrar en funcionamiento, ni de modo manual ni automático.
- Manual. El servicio puede ponerse en marcha tal y como hemos explicado en este capítulo, mediante botones o menú emergente desde esta herramienta.



Ejercicios

Ejercicio 1

Una vez instalado SQL Server 2005 comprobar el estado de todos los servicios SQL Server y el modo de ejecución de cada uno de ellos, dejando todos los servicios en funcionamiento y el modo de inicio en automático.

Recuerda que para este tipo de configuraciones, utilizaremos SQL Server Configuration Manager.

Nº- 2 Introducción a las bases de datos

1 Definición

El concepto más general de una base de datos es el lugar donde se guardan los datos. Vamos a suponer que somos los dueños de una librería y deseamos tener almacenados todos los datos posibles de nuestro negocio. Para poder llevar cierta organización de estos datos, no queremos que estos datos sean un montón de información almacenada sin seguir ningún orden. Debemos seguir una estructura que permita que esa información esté enlazada, de modo que tengamos conectados unos datos con otros, para que por ejemplo no nos volvamos locos, buscando un libro que nos ha pedido un cliente.

Siguiendo con el ejemplo de la librería, podemos organizar nuestra necesidad en los siguientes puntos:

- Aspecto particular de un determinado libro, por ejemplo el precio.
- Aspectos importantes de un libro, por ejemplo Título, Autor, ISBN, Páginas, Género, etc...
- Aspectos importantes de todos los libros.
- Aspectos importantes de la librería al completo.

En este caso, hemos desglosado a primera vista, los cuatro puntos más importantes en cuanto a información se refiere de nuestra librería, podíamos haber entrado en niveles más complejos, como pueden ser clientes, proveedores, empleados, pero no es lo que deseamos para empezar a explicar los fundamentos de las bases de datos.

Como estamos diciendo, hemos separado en cuatro puntos, las necesidades que podemos tener en cuanto a información de un libro. Estos cuatro puntos, nos dan una idea muy clara de la jerarquía que sigue una base de datos, esta jerarquía la vamos a separar en cuatro niveles, y verás la lógica que tienen estos niveles respecto a los puntos de nuestra librería:

- Campo: Contiene un dato en particular, como puede ser el primer punto que hace referencia al precio de un libro.
- Registro: Almacena todos los datos de un determinado objeto de información, vemos que el segundo punto de nuestras necesidades reclama los aspectos más importantes de un libro. En este caso, el libro es el objeto de información, y sus aspectos (Título, Autor, ISBN, Páginas,...) de ese objeto de información serían un grupo de campos, al igual que sucede con el precio.
- Tabla: Almacena información de varios objetos de información que comparten aspectos similares. Estamos mencionando el tercer punto de nuestra librería, donde queremos almacenar la información de todos los libros, podemos pensar, pero cada libro es diferente al resto, y es cierto, pero todos los libros tienen en común que cada uno de ellos tiene un determinado Título, Autor, ISBN, Páginas, Género, etc... Por lo tanto, si hemos entendido bien, los conceptos de los dos anteriores niveles, podemos asegurar que una tabla almacena una serie de registros (libros).

- Base de datos: Cuarto y último nivel, de nuestro primer vistazo a la idea de base de datos, relacionada con el cuarto punto de nuestra librería el cual nos indica que queremos almacenar los aspectos de la empresa al completo, por lo tanto, este nivel guarda información de varios aspectos, no sólo de libros, sino de ventas, compras, clientes etc...Por lo tanto la base de datos, dicho de un modo muy simple y muy genérico, almacena las tablas.

Acabamos de mencionar los cuatro conceptos básicos de toda base de datos, si es la primera vez que te introduces en este mundillo deben quedarte muy claros estos cuatro pilares de información.

Tal y como hemos avisado, esta definición de base de datos es demasiado simple, decir que la base de datos se encarga de almacenar la información estructurada en esos cuatro niveles es decir demasiado poco. Muchos fabricantes ofrecen en sus servidores la posibilidad de almacenar muchas mas funcionalidades que estas cuatro.

Microsoft SQL Server 2005 ofrece una cantidad enorme de objetos, que al igual que los datos se almacenan en la base de datos, pero cuya función no es guardar información, sino trabajar con ella. Así a primera vista, puede parecer complicado, ¿Una base de datos almacena algo más que datos? Veremos que así es, y que son de una importancia grandísima, ya que tienen tareas tan importantes como asegurar que esos datos se almacenan correctamente, de la seguridad, del rendimiento que obtenemos de esos datos, etc...Pero como te digo, los iremos viendo a lo largo del curso.

2 Estructuración de una base de datos

2.1 Estructura física

Una base de datos se almacena en varios ficheros o archivos en disco. Como mínimo tendremos dos ficheros que explicaremos más adelante.

Tenemos la posibilidad de almacenar estos ficheros en discos que no estén ni tan siquiera formateados o que no tengan una partición hecha, pero este método no es el más aconsejable. Es más razonable almacenar estos archivos en un disco ya formateado, con formato NTFS.

En empresas cuyo volumen de datos es altísimo y el trabajo que se realiza sobre la base de datos soporta una actividad elevada, se almacenan los archivos en grupos de discos denominados RAID por hardware. Este método mejora considerablemente el rendimiento, y nos asegura que en caso de fallos inesperados no perdamos esa valiosa información.

Como es lógico, nosotros para realizar nuestros ejemplos, no vamos a basarnos en esta tipo de estructuras de hardware, lo almacenaremos en nuestro disco duro, aunque veremos como asegurar nuestros datos mediante planes de mantenimiento con copias de seguridad automáticas.

Como hemos mencionado, como mínimo tendremos dos archivos donde almacenar la base de datos:

- Archivo de datos.
- Archivo de registro de transacciones.

Pero debes saber que tenemos otras posibilidades y podemos utilizar archivos extras para mejorar el rendimiento de nuestra base de datos, podemos usar varios archivos, si pensamos que nuestra base de datos va a alcanzar un tamaño grande. O si deseamos que nuestros datos se almacenen en diferentes dispositivos de almacenamiento u ordenadores, y de este modo permitir un trabajo más rápido al poder acceder a la información en paralelo.

Centrándonos en lo principal:

- El archivo de datos, o aquellos que añadimos como extras, son los archivos que tendrán almacenada la información, los datos. Pero recuerda que hemos dicho que SQL Server 2005 nos permite también crear en nuestras bases de datos, no sólo información, sino también una serie de objetos que trabajan con la información. Pues bien, esta serie de objetos también se almacena en el archivo de datos.
- Por otro lado, tenemos el archivo de registro de transacciones. Este fichero es tan importante como el anterior. Su importante tarea es garantizar que esa base de datos permanece integra. Gracias a estos archivos de registros (puede haber más de uno), en caso de ser necesario,

podremos recuperar la base de datos, ya que almacena las modificaciones que se producen debido a la actividad o la explotación de la base de datos.

2.1.1 Nombres de archivos.

El modo de nombrar una base de datos, parte de una base fija, de un nombre principal que generalmente entrega el administrador de la base de datos. Una vez que tenemos este nombre principal, SQL Server 2005 se encarga de añadir terminaciones y unas determinadas extensiones, a ese nombre principal. El administrador además de seleccionar el nombre principal, puede elegir el destino donde se almacenarán los ficheros que forman la base de datos.

Vamos a suponer que estamos en una empresa como administradores, y estamos creando su base de datos. Nosotros como administradores le damos el nombre principal " miEmpresa ". Ese será el nombre de la base de datos, pero los ficheros donde se almacenará su información y el registro de transacciones, serán:

- Archivo de datos: miEmpresa_Data.MDF
- Archivo de registro de transacciones: miEmpresa_Log.LDF

En caso de tener archivos extras, nosotros como administradores también podremos darles su nombre principal, y la extensión que suele utilizarse es .NDF

Siguiendo con nuestra tarea de administrador, ahora sería el momento de seleccionar el lugar de almacenamiento, como ya sabes podemos seleccionar una determinada carpeta o directorio, incluso diferentes unidades físicas. Lo más aconsejable es guardar en diferentes unidades, por un lado el archivo de datos, y por otro el archivo de registro de transacciones. De modo que en caso de fallo, por lo menos tengamos uno de ellos.

A continuación puedes ver una figura que representa la estructura física de la base de datos, tomando como ejemplo el nombre principal "MiEmpresa".

No debes quedarte con la idea de que una base de datos, se compone sencillamente de dos archivos, es algo mucho más completo que todo eso lo que representa una base de datos como entidad.

2.1.2 Tamaño de la base de datos.

En el momento de crear la base de datos, es casi imposible conocer la cantidad de memoria que necesitará para almacenar toda la información. Es cierto que hay ciertas técnicas que nos permiten calcular el tamaño que podrá alcanzar la base de datos, pero estas estimaciones pueden venirse a bajo, por modificaciones imprevistas, como puede ser el crecimiento de la empresa y que se intensifique la actividad realizada sobre la información, por citar un ejemplo.

Tampoco es nada aconsejable pecar de precavidos y reservar una cantidad de memoria exagerada, y pensar que con esta cantidad casi infinita no tendremos problemas de espacio para

nuestros datos. De acuerdo, puede que no haya problemas de espacio (o quizá sí), pero lo que es seguro es que tendremos muchísimos problemas de rendimiento, de fragmentación etc...

SQL Server 2005 nos permite olvidarnos hasta cierto punto de este problema. Los archivos de datos y de registro, crecen automáticamente. No crecen con cada dato que se añade. Nosotros como administradores, le daremos un tamaño inicial sencillo de estimar (una cantidad muy pequeña, unos Megabytes), en ese momento SQL Server 2005 crea la estructura correcta para la base de datos, y una vez que nuestra base de datos está en explotación cuando alcanza el tamaño limite, lo incrementa una cantidad dada por un factor predeterminado.

Visto de modo teórico puede asustar un poco, sólo estamos comenzando a crear la base de datos, y estamos mencionando varias características a tener en cuenta. No tenemos porque asustarnos, veremos como estos parámetros se pueden dar de un modo altamente sencillo mediante el interfaz de SQL Server 2005, y como con pocos clicks, todos estos aspectos los realiza SQL Server por nosotros, así que no te preocupes y sigue leyendo.

2.2 Estructura lógica

Para entender que es la estructura lógica de una base de datos vamos a poner un sencillo ejemplo.

Cuando nosotros nos compramos un equipo de música, poco nos importa como funcionan los circuitos integrados, los elementos electrónicos que componen nuestro equipo. En este caso, esos circuitos, esos dispositivos electrónicos, sería la estructura física del equipo de música, al igual que hemos visto la estructura física de nuestra base de datos.

A lo que nosotros como usuarios vamos a dar importancia es al manejo del equipo de música: como subir el volumen, encenderlo, cambiar de emisoras, introducir un CD. De igual modo, como usuarios de la base datos, debemos conocer la estructura lógica de la base de datos para poder gestionar o trabajar con los datos.

Una estructura lógica mínima puede ser el ejemplo de la librería que hemos visto a modo de introducción en esta lección.

Lo que vamos a exponer a continuación a modo de introducción son los elementos principales que componen la estructura lógica de una base de datos, de modo que sepas de que estamos hablando en caso de que se mencionen en las diferentes lecciones. Sin embargo, los iremos viendo con más detenimiento más adelante, de momento es suficiente con que te suenen y las vayas conociendo.

2.2.1 Tablas

Las tablas son las unidades que almacenan los datos. Como norma general se suele imponer que cada tabla, almacena información común sobre una entidad en particular (recuerda los libros). Esta norma se conoce como **normalización**.

CustomerID	CompanyName	ContactName	ContactTitle	Address	City
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Represent...	Obere Str. 57	Berlin
ANATR	Ana Trujillo Emp...	Ana Trujillo	Owner	Avda. de la Con...	México D.F.
ANTON	Antonio Moreno ...	Antonio Moreno	Owner	Mataderos 2312	México D.F.
AROUT	Around the Horn	Thomas Hardy	Sales Represent...	120 Hanover Sq.	London
BERGS	Berglunds snabb...	Christina Berglund	Order Administr...	Berguvsvägen 8	Luleå
BLAUS	Blauer See Delik...	Hanna Moos	Sales Represent...	Forsterstr. 57	Mannheim
BLONP	Blondesddsl pèr...	Frédérique Citeaux	Marketing Manager	24, place Kléber	Strasbourg
BOLID	Bólido Comidas p...	Martín Sommer	Owner	C/ Araquil, 67	Madrid
BONAP	Bon app'	Laurence Lebihan	Owner	12, rue des Bou...	Marseille
BOTTM	Bottom-Dollar M...	Elizabeth Lincoln	Accounting Man...	23 Tsawassen Bl...	Tsawassen
BSBEV	B's Beverages	Victoria Ashworth	Sales Represent...	Fauntleroy Circus	London
CACTU	Cactus Comidas ...	Patricio Simpson	Sales Agent	Cerrito 333	Buenos Aires
CENTC	Centro comercial...	Francisco Chang	Marketing Manager	Sierras de Grana...	México D.F.

La mayor parte de la actividad producida en una base de datos se produce sobre las tablas, siendo las principales tareas las siguientes:

- Añadir información.
- Eliminar información.
- Modificar y actualizar información.
- Recoger información y mostrarla.

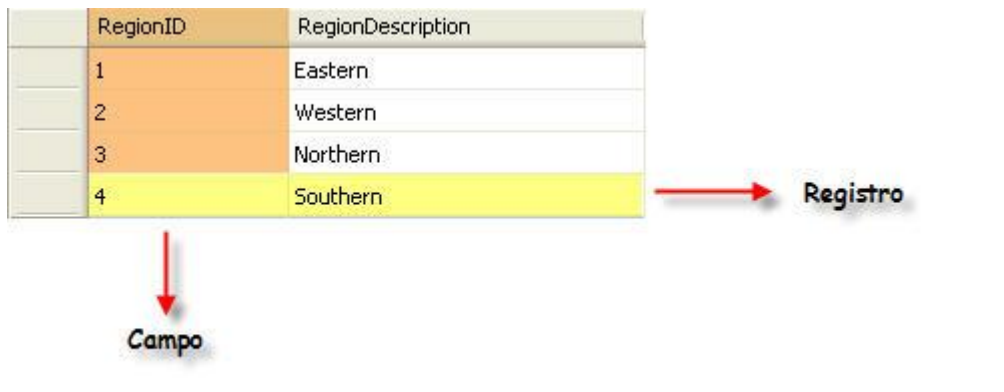
Comúnmente estas cuatro tareas se realizan mediante el lenguaje SQL (Structured Query Language) que significa Lenguaje de consultas estructurado, que como ya sabrás es el lenguaje estándar para gestionar bases de datos.

2.2.2 Campos y Registros.

Las tablas están compuestas de registros y campos. Si imaginamos el diseño de una tabla, como si de una cuadrícula se tratase, sabemos que está compuesta por varias filas y columnas. Las filas corresponden a los registros, mientras que las columnas serían los campos. Y cada una de las celdas que se forman de enlazar un registro (fila) con un campo (columna) formaría una celda, la cual almacena un valor.

Por lo tanto, un registro esta formado por varios campos, y cada campo almacena un determinado valor.

RegionID	RegionDescription
1	Eastern
2	Western
3	Northern
4	Southern



Acabamos de mencionar que en cada celda se almacena un valor. Pero ¿siempre debe almacenar un valor?, que sucede si estamos almacenando información de un determinado producto, y en el momento de almacenarlo desconocemos una de sus características (un valor). En ese caso ¿no podríamos almacenarlo?. Lógicamente la respuesta a esta pregunta es que sí, por supuesto que podríamos almacenarlo. Tenemos la posibilidad de almacenar valores nulos para este caso que hemos visto y muchos otros.

Los valores nulos se conocen como NULL, pero aunque se conoce como valor nulo, no debes pensar que se almacena un valor, el concepto de NULL podría ser un marcador que informa que hay que pasar por alto los datos de esa celda, son datos que se ignoran. Como veremos, hay que tener mucho cuidado con el uso de esta "ausencia" de información, ya que si tenemos algún despiste puede ser el causante de que no recibamos la información que realmente estamos reclamando en una consulta. Si realizamos operaciones matemáticas con varios valores de nuestra base de datos y uno de estos es NULL, el resultado siempre será NULL.

2.2.3 Índices

Seguimos hablando de la unidad o entidad principal de la base de datos, las tablas. Podemos tener tablas con millones de registros, si realizamos una consulta para recuperar información de un grupo de estos registros, podemos tener un rendimiento bajo debido a la gran cantidad de información que almacena esa tabla.

Sitúate en una tabla que almacena las ventas de productos que se han producido durante décadas en una gran empresa, y queremos recoger la información de una determinada venta. El proceso de búsqueda recorriendo cada uno de esos registros (ventas) de la tabla de principio a fin, puede necesitar un tiempo considerable hasta que encuentra la información deseada.

Para acelerar este tipo de consultas contamos con la ayuda de los **índices**. Un índice es una característica más de las tablas, el cual es una conjunto de valores clave. Este conjunto tiene una estructura estudiada para que el servidor pueda realizar las consultas con un rendimiento mucho mayor.

Estos valores claves pueden almacenar el contenido de una o varias columnas de la tabla sobre la que operan.

Además de mejorar el rendimiento, existen índices que pueden asegurar la integridad de los datos, indicando en que orden deben almacenarse los datos en una tabla. Más adelante veremos como trabajar con los índices.

Por lo tanto podemos decir que teóricamente nuestras tablas deberían todas incluir al menos un índice que asegure un mejor rendimiento. Y en la práctica, suele ser lo más común, pero debes tener en cuenta que cada vez que realizamos una tarea sobre una tabla que está relacionada con índices, el servidor, no sólo opera sobre la tabla para realizar las modificaciones que se le demanden, sino que también debe realizar operaciones sobre los índices para asegurar que la labor de estos sigue siendo la adecuada para las modificaciones realizadas sobre la información de la tabla. Por lo tanto la regla de tres es sencilla, a mayor número de índices, mas tiempo dedicará a las tareas pedidas.

2.2.4 Restricciones

Las restricciones son normas que imponemos a la información que puede ser almacenada, de modo que si no se cumple una de estas condiciones no permitamos que incluya ese valor en nuestra base de datos.

Si tenemos una tabla de clientes, podríamos poner como restricción que no se pueda almacenar un cliente cuya edad no supere los 18 años. El servidor se encargará de no permitir que se incluya ningún registro que incumpla nuestra condición.

Estas restricciones además de permitir controlar que valores pueden ser almacenados, con esta tarea aseguramos también la integridad de nuestros datos. Piensa que por descuido un usuario introduce por error un cero como el número de unidades que se han vendido a un minorista. Al calcular el precio de la venta y multiplicarlo por cero unidades, tendremos como precio de facturación cero.

Pero no sólo debemos pensar en que es el usuario quien comete el error, puede que el programador que ha desarrollado un software de facturación haya cometido un error al escribir el código. El programa por sí sólo, si no ha tenido en cuenta esta posibilidad, no lanzará ningún error, y todo parecerá ir correctamente y emitirá la factura al minorista. Pero gracias al servidor de base de datos, podemos tener controlado que uso hace el software que trabaja con nuestros datos.

2.2.5 Vistas

Las consultas que se realizan sobre algunas de las tablas de la base de datos pueden ser repetitivas, de modo que día tras día cientos de usuarios realizan las mismas consultas sobre la tabla. Todas esas consultas repetitivas reciben el mismo grupo de datos.

Para evitar la repetición de este tipo de consultas tenemos las vistas.

Podemos pensar que una vista es un conjunto de registros determinados de una o varias tablas. De hecho se trabaja sobre ella como una tabla, pero no es una tabla. Lo que almacena en realidad es una consulta. Pero debes tener claro que no almacena datos sino que los extrae.

Una vista puede crear los enlaces necesarios para obtener información de varias tablas como si fuese una única tabla. Esto puede facilitar mucho la tarea al desarrollador de software que no tiene que preocuparse de las tablas donde se almacena la información que quiere recoger, ya que lo tiene todo en una vista sobre la que puede operar como si fuese una tabla. Por lo tanto se olvida de construir complicadas sentencias de SQL que recoja esa información de múltiples tablas, con diferentes enlaces entre ellas.

2.2.6 SQL

Las consultas y las tareas de gestión que se realizan durante la explotación de una base de datos vienen escritas en lenguaje SQL, Structured Query Language, que como ya hemos mencionado significa Lenguaje de Consulta Estructurado.

El ANIS (Instituto Nacional de Normalización Estadounidense) ideó este lenguaje estándar, denominado ANSI SQL, o también SQL-92, por el último año en el que ANSI aceptó modificaciones sobre el estándar.

Como suele ocurrir en tantas ocasiones en el mundo del mercado informático. Este estándar fue recogido por los fabricantes para personalizarlos y crear sus propias extensiones para sus productos. Microsoft así lo hizo, para crear Transact-SQL, o más comúnmente conocido por su abreviatura T-SQL para sus servidores de base de datos SQL Server.

Como veremos, SQL Server 2005 y sus antecesores, pueden trabajar con SQL, pero gracias a T-SQL podemos realizar sentencias más completas que solventarán fácilmente problemas. Por lo tanto T-SQL no es un sucesor de SQL para nosotros, sino una ampliación, una herramienta extra que utilizaremos para fines más avanzados.

2.2.6 Procedimientos almacenados

SQL Server no sólo puede ejecutar las consultas de las tablas, o las vistas que ya hemos visto. También permite que desarrollemos procedimientos con código escrito íntegramente en SQL o con la ayuda extra de T-SQL. Tanto con el estándar como con la versión de Microsoft, podemos crear sentencias que vayan más allá de consultas, ya que como lenguajes de programación que son, pueden contener sentencias condicionales, bucles, etc... Si nunca te has introducido en ningún lenguaje de programación, no te preocupes porque veremos estos conocimientos con detenimiento, y si ya conoces otros lenguajes de programación, aprenderás la sintaxis específica de este lenguaje.

Los procedimientos almacenados, como cualquier función de otro lenguaje, pueden recibir parámetros de entrada y de salida, o no recibir ni devolver nada. Además de devolver parámetros, pueden devolver incluso tablas virtuales, vistas, etc...

Los procedimientos almacenados los almacena SQL Server 2005 del modo más óptimo para sacarles el mejor rendimiento posible. De este modo las instrucciones quedan almacenadas en la propia base de datos. Esto es una gran ventaja, en cuanto a seguridad y rendimiento, ya que los programas desarrollados por los programadores no necesitan tener estas sentencias SQL en el código de su software, y por lo tanto esta información que supone el propio código SQL, no tiene que "viajar" del programa a la base de datos. Y como es lógico pensar, cuanto menos información "viaje" del programa del cliente al servidor, ganaremos en seguridad y en rendimiento.

2.2.7 Varios

Además de los elementos que hemos ido numerando de la estructura lógica de una base de datos, tenemos otros muchos que como siempre se almacenan en nuestra base de datos y que sirven para realizar diversas tareas. Algunos de estos elementos los iremos estudiando, pero para poder continuar con la lección sin problemas ya tenemos los conocimientos básicos, que ampliaremos más adelante.

3 Planificación - Diseño

3.1 Introducción

Para aquellas personas que han trabajado con desarrollo de software, entenderán que la planificación de un programa es uno de los aspectos más importantes para un buen programa.

Cuando vamos a comenzar a crear una base de datos, el principio de todo no es arrancar nuestra herramienta de SQL Server 2005, Management Studio, y mediante sus asistentes ponernos directamente a crear nuestra base de datos, con sus tablas, etc...

Siempre debemos comenzar realizando un análisis de las necesidades que tenemos. Una vez que conocemos las necesidades y partiendo de estas, definimos la estructura lógica de nuestra base de datos, y por último pasamos a diseñarla, bien sea en papel, o mediante las herramientas de diseño que nos ofrece SQL Server 2005.

Por lo tanto tenemos diferenciados claramente tres pasos previos a la hora de planificar la creación de una base de datos:

- Estudio y análisis de las necesidades.
- Definición de los elementos que componen la estructura lógica de la base de datos.
- Diseñar o plasmar esa estructura con diagramas en papel, o con SQL Server 2005.

Estos pasos son importantísimos, en este tipo de desarrollos no podemos utilizar el método de ir probando estructuras hasta que funcionen, y una vez que funciona dar por sentado, que nuestra base de datos está bien diseñada y ya podemos instalarla en un servidor para que se comience a trabajar con ella.

La planificación debe prevenirnos de imprevistos que pueden surgir, y ya no sólo imprevistos del tipo que si tenemos mayor o menor actividad en la base de datos, hablamos de imprevistos que provocan que tengamos que rediseñar nuestras estructuras. Una vez que tenemos la base de datos en explotación, nos encontraremos con el problema de que será muy difícil realizar modificaciones sobre ella, ya que el método de trabajo que marcamos al definir nuestras estructuras, limita las posibles modificaciones futuras.

Con esto puedes hacerte una idea de lo que supone la planificación, invertir tiempo en estos primeros pasos, significa ahorrarnos tiempo en un futuro de posibles modificaciones.

3.2 Estudio de necesidades.

Como hemos dicho, se trata del primer paso que debemos analizar.

En este paso, tendremos que estudiar detenidamente que tipo de necesidad tiene nuestro cliente. Serán necesarias tantas reuniones como creamos oportunas para que el cliente nos explique cual es la información desea almacenar en nuestra base de datos. Una vez tenemos claro la información fundamental para el trabajo cotidiano de la empresa de nuestro cliente, tendremos que tener pleno conocimiento, de que trabajo quiere hacer con esa información, es decir ¿Cuáles son los resultados que va a requerir de esa información?.

Una vez, que tenemos todo lo necesario para conocer que necesidades tenemos que cubrir con nuestra base de datos, procedemos a identificar todos los objetos que necesitamos, con sus atributos y propiedades, y también como estarán relacionados unos con otros. Estos objetos se convertirán más adelante en tablas, vistas, índices, procedimientos almacenados, etc...

Es lógico penar que estos objetos que hemos planificados no serán los definitivos ni mucho menos, durante la etapa de diseño y normalización que explicaremos más adelante nos irán surgiendo nuestras propias necesidades para cumplir con los objetivos que nos hemos marcado con nuestro cliente, o los encargados de informarnos en la empresa que estamos contratados.

Lo que si debe ser definitivo (en la medida de lo posible) son los elementos que integran la base de datos, las entidades de información que debemos almacenar. Deberemos identificar estos elementos, de modo que podamos distinguir fácilmente unos de otros. Y más adelante, durante la planificación iremos encontrando las relaciones y dependencias que tienen unos con otros. Para entender mejor esta definición vamos a poner un ejemplo.

Vamos a suponer que tenemos como cliente a una empresa que se dedica a la venta de refrescos.

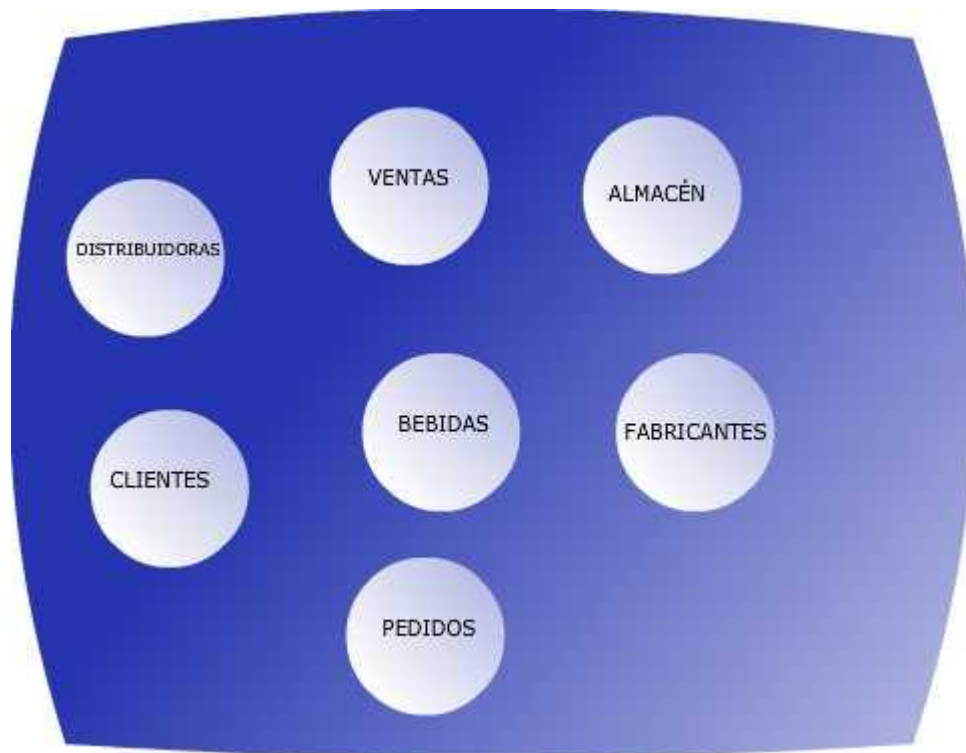
El primer paso es definir las necesidades que surgen de estudiar el funcionamiento de esta empresa. Por lo tanto comenzamos describiendo el funcionamiento de la empresa.

Nuestra empresa se dedica principalmente a la venta de **bebidas**, estas bebidas las producen las **fabricante** y las recibimos de estas a través de unas **distribuidoras** a las que previamente habremos realizado unos **pedidos**.

Una vez que hemos recibido los pedidos, estos permanecen en nuestro almacén, para que podamos realizar las ventas a nuestros clientes.

Está claro que una empresa de bebidas tiene un funcionamiento mucho más complejo que lo que acabamos de definir, pero para el objetivo que buscamos, nos es suficiente.

Hemos subrayado las entidades de información que han ido surgiendo en la descripción de las necesidades de la empresa que hemos tomado como ejemplo. Todas estas entidades forman un conjunto de entidades, como puedes ver en la siguiente figura:



3.1.2 Planificación

Una vez que tenemos descrita y definida la información de la etapa de análisis de necesidades y de las indicaciones del trabajo que se desea realizar sobre esas entidades marcadas, comenzamos con la etapa de planificación de la base de datos.

La etapa de planificación tiene el objetivo de definir las diferentes propiedades de cada entidad y analizar que relaciones serán necesarias entre ellas o con más elementos o entidades.

A primera vista, parece claro que cada una de las entidades de nuestro conjunto puede ser representada en una tabla en nuestra base de datos final.

Las propiedades de cada una de estas tablas, no son otra cosa que las columnas o campos que la van a formar. Es decir, los datos que nos interesa almacenar para las bebidas, pedidos, clientes, fabricantes etc...

Otra característica a definir en esta etapa son las dependencias que tendrán las tablas. ¿A qué cliente estamos realizando una venta? ¿Que fabricante a producido una determinada bebida?.

Una vez realizado este estudio, continuamos con un proceso de **normalización**, el cual modificará el número de entidades o tablas, de tal manera que puede aumentar el número de tablas pero disminuir la cantidad de información que almacena en cada una. Este proceso de normalización es de vital importancia, y lo estudiaremos con detenimiento en este mismo capítulo.

3.1.3 Propiedades

Partiendo de las entidades que hemos definido en nuestro conjunto como resultado del estudio de necesidades, vamos a analizar las propiedades de cada una de estas entidades.

Comenzamos con la más lógica, Bebidas. ¿Cuáles podrían ser las propiedades fundamentales? Piensa por ejemplo que dato te interesaría para realizar una búsqueda: el **nombre** de la bebida, y el **fabricante**.

También necesitaremos saber que **distribuidora** nos ha entregado esa bebida. Estas propiedades o campos y los anteriores, son los esenciales para cada bebida, por último podremos añadir otros campos que nos interesen, precio, el envase, cantidad, fecha de caducidad, si es alcohólica o no, en caso de ser alcohólica los grados y el número de lote.

Con estas propiedades ya podemos definir las columnas o campos de nuestra tabla Bebidas.

Bebidas
nombre
fabricante
distribuidora
precio
envase
cantidad

fecha_caducidad
alcohólica
grados
lote

Del mismo modo podemos ir definiendo las propiedades del resto de las tablas que representan nuestras entidades en la base de datos:

Cientes
nombre
dirección
teléfono

Fabricantes
nombre
distribuidora
dirección
teléfono
email

Distribuidoras

nombre
fabricantes
dirección
teléfono
email

Pedidos
fecha
distribuidora
bebidas
unidades

Almacén
bebida
unidades
situación

Ventas
fecha

bebida
unidades
cliente

Todas estas propiedades definidas para cada tabla, son de carácter temporal y no son definitivas, veremos como en el proceso de normalización, tendremos que realizar modificaciones como ya hemos comentado.

3.1.4 Dependencias

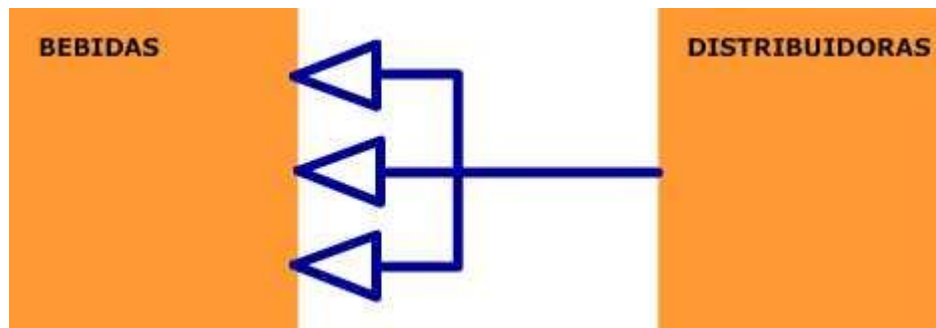
Hemos visto como las tablas formaban un conjunto de entidades. Nunca debes pensar en una tabla como un elemento aislado, cada una de las tablas forma parte de nuestro conjunto, y el conjunto vendría a ser la base de datos. Por lo tanto, cada tabla es la parte de un nivel superior y no puede ser tratada individualmente, ya que como parte de un todo, tendrá dependencias con el resto de tablas de la base de datos.

Ese es precisamente el trabajo que debemos llevar a cabo ahora, analizar las dependencias que tiene cada tabla con el resto para poder representarla la estructura lógica de la base de datos.

Estas dependencias reciben el nombre de relaciones, y estudiaremos que tipo de relaciones podemos tener, de momento, para nuestro ejemplo las veremos de un modo muy sencillo. Podemos encontrarnos tablas que dependen únicamente de una segunda tabla, mientras que habrá otras que dependerán de varias. Otro caso que nos podemos encontrar son tablas que tienen una relación única, como iremos viendo.

Cojamos el ejemplo de la tablas Bebidas y Distribuidoras, estas tablas tendrán una relación entre ellas de "uno a muchos". La razón de porque tendrán esta relación y porque recibe esa descripción de "uno a muchos" es la siguiente. Esta claro, que en la tabla distribuidoras almacenamos los datos de cada una de las distribuidoras que tenemos, y que lógicamente una distribuidora, aparecerá sólo una vez en esta tabla, es absurdo tener repetida la misma información. En cambio, si vamos a la tabla bebidas, donde almacenamos cada bebida, sabemos que una distribuidora no nos entregará una única bebida, sino varias. Por lo tanto, tendremos varias bebidas que han sido entregadas por la misma distribuidora.

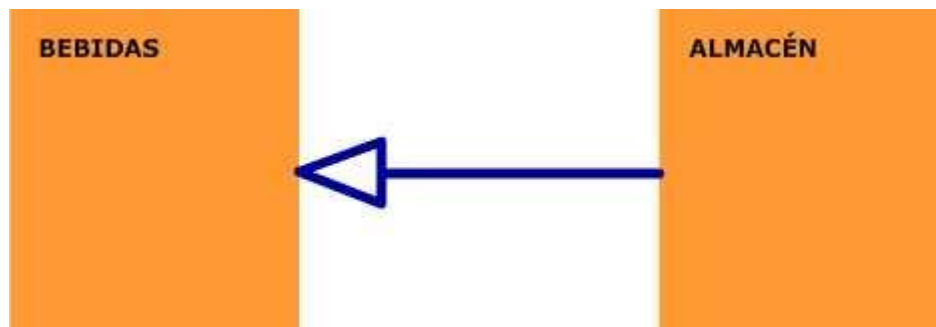
Resumiendo, una distribuidora sólo aparece una vez en la tabla distribuidoras, mientras que en la tabla bebidas aparecerá muchas veces (no sabemos cuantas). Por eso afirmamos que la relación entre estas tablas es de **una** (bebidas) a **muchas** (distribuidora). Si representamos de modo gráfico esta relación podría ser como ves en la siguiente figura:



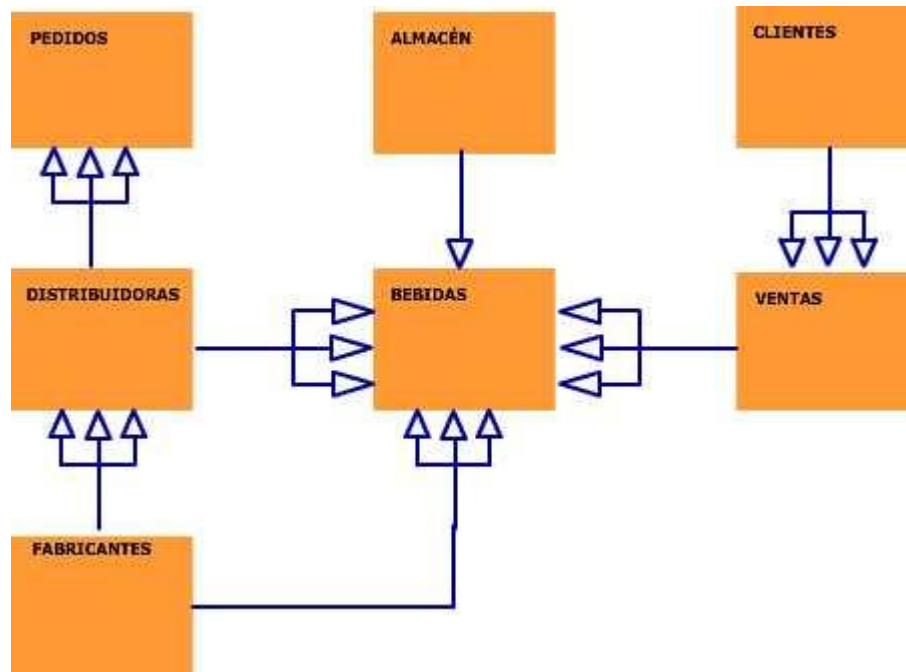
Tomemos ahora como ejemplo la relación entre las tablas Bebidas y Almacén.

La dependencia entre estas tablas es una relación "uno a uno", el motivo es el siguiente. Cada una de las bebidas que tenemos en nuestra tabla bebidas, tiene una situación determinada en el almacén, por lo tanto, existirán las mismas filas en la tabla almacén como tenemos en la tabla bebidas. Así pues a **un** registro de la tabla bebida le corresponde **un** registro en la tabla almacén. De ahí la definición de relación **uno a uno**.

Podemos representar esta relación gráficamente de la siguiente forma:



Si continuamos con el mismo estudio de dependencias con el resto de la tabla de nuestra base de datos, veremos que la tabla ventas depende de los clientes con relación uno a muchos y a su vez depende de la tabla bebidas, y la tabla almacén teniendo dependencias de varias tablas. Si representamos todas estas relaciones tendremos:



Estudia con detenimiento la figura anterior, y trata de entender las dependencias que hemos representado, y el motivo de estas relaciones a partir de los ejemplos de relaciones uno a uno y uno a muchos que hemos descrito anteriormente.

4 Normalización

4.1 Definición

La normalización es el mecanismo de toma de decisiones con el objetivo de recoger todos los datos de la información que se almacenará en una base de datos y distribuirlos en tablas.

Para tomar estas decisiones tenemos un número de **formas normales** que nos ayudará a diseñar la mejor estructura lógica con el mayor rendimiento posible.

Las formas normales, son los modelos o maneras en que se pueden representar la estructura de tablas. Gracias a estos modelos conseguiremos mayor eficacia. Pero no entiendas por eficacia como una reducción del tamaño, nos estamos refiriendo a que obtendremos una estructura muy bien organizada, de tal modo que será escalable fácilmente, permitiendo realizar modificaciones en un futuro sin muchos problemas. Aunque habrá veces donde gracias a la normalización también se reduzca el tamaño, este no es el objetivo que buscamos.

La función de la normalización es favorecer la integridad de los datos, sin importar la actividad que se desarrolle sobre la base de datos. Trata de evitar lo máximo posible la posibilidad de introducir datos que no sean razonables. Dentro del proceso de normalización podemos distinguir cuatro tipos de integridades:

- Integridad de entidad.
- Integridad de dominio.
- Integridad referencial.
- Integridad definida por el usuario.

Vamos a explicar cada una de estas integridades y al final de cada una nombraremos que herramientas nos ofrece SQL Server 2005 para cumplir con estas integridades, si desconoces estas herramientas, tranquilo porque las veremos con más detenimiento en las siguientes lecciones, tan sólo que te suene para cuando lleguemos a verlas con más detenimiento.

4.2 Integridad de entidad

Hasta ahora hemos utilizado en varias ocasiones la palabra entidad. Una entidad se define como un concepto del mundo real, de modo que nuestras bases de datos guardan información sobre entidades. Estas entidades puede ser de diferente carácter:

- Entidades físicas: un libro, una bebida, un empleado
- Entidades conceptuales: una empresa
- Entidades como eventos: una alerta de nuestra agenda que nos recuerda una tarea.

Uno de los pasos de nuestro proceso de planificación es detectar estas entidades que están relacionadas con la base de datos.

La integridad de entidad pretende que cada entidad que se guarda en la base de datos sea identificable de un modo único, es decir, que evitemos la información redundante.

Ahora bien, la identificación de entidades no es única, podemos tener varios modos de entidad para un mismo objeto real. Por ejemplo, seguimos con el ejemplo de nuestra empresa dedicada a la venta de bebidas, podríamos identificar las bebidas de un modo general, a un modo más individual:

- Todas las bebidas en un sólo grupo.
- Todas las bebidas de la misma marca en un grupo.
- Agrupar las bebidas en función de si son alcohólicas o no.
- Cada bebida de modo individual.
- Un hecho sobre una determinada bebida, como puede ser el sabor de un refresco.

¿Qué criterio debemos seguir entonces para identificar que es una entidad en nuestra base de datos?

La respuesta a esta pregunta dependerá de lo que deseemos hacer con estos datos. Lo más razonable es que se identifique como identidad aquellas cosas con las que vas a trabajar de modo unitario. Dicho de un modo más claro, la información que se almacena unida (de modo unitario) es más cómodo trabajar con ella, o recuperar esa información en una única operación.

Para tomar estas decisiones, simplemente debes realizarte las preguntas de la función que deseas realizar sobre los datos. Por ejemplo, ¿Cuántas bebidas tenemos de un determinado fabricante?,

se puede decir que nuestra entidad será todas las bebidas que provienen de un determinado fabricante.

En cambio, si nos realizamos preguntas como ¿En que fecha llegó al almacén una determinada bebida? Estamos requiriendo que la entidad sea cada bebida de modo individual.

Con las entidades ya identificadas, ahora debemos identificar los hechos que nos dan la descripción de cada entidad, en el ejemplo de las bebidas ya los hemos descrito, ya que estamos hablando de los campos:

- nombre
- fabricante
- distribuidora
- ...

Como siguiente paso, tenemos que identificar la entidad o grupo de entidades que en cierta medida, comparten este grupo de hechos que acabamos de describir. Para nuestro ejemplo está claro que sería las bebidas que vendemos en la empresa.

Una vez vistos estos pasos de la integridad de entidad, veamos como se relacionan o se representan en una base de datos:

- Una entidad se representa como el registro de una tabla.
- Un hecho (como ya hemos dicho) sería el campo o columna de la tabla.
- Un grupo de entidades que comparten unos hechos, representaría una tabla.
- Y como es lógico, la tabla está formada por la cuadrícula que se crea al unir las entidades con los hechos, por lo que cada entidad tiene un valor para cada hecho determinado.
- El conjunto de todos estos valores, representa todo lo que podemos saber de una entidad.

Presta atención a lo que vamos a explicar a continuación, es muy importante. Cada entidad almacenada debe tener una **clave principal**. Una clave principal es un hecho, o grupo de hechos, que distinguen esa entidad del resto de entidades que comparten unos mismos hechos. Bueno, visto así puede parecer un poco complicado, simplemente estamos explicando que cada registro de una tabla, debe tener un campo que identifique de modo exclusivo ese registro respecto al resto de registros de esa tabla.

Para el ejemplo de nuestra empresa de bebidas, los fabricantes que producen las bebidas que acaban en nuestro almacén, pueden ser muchísimas empresas, para identificar cada empresa del resto, tenemos el NIF de la empresa, sabemos que ese código es único y exclusivo para cada empresa y que no estará repetido en ninguna otra. Pues perfectamente el NIF sería nuestra clave principal. En caso de no encontrar un hecho o campo que pueda ser clave principal, lo que demos hacer nosotros es quedar un código que identifique exclusivamente cada registro. En caso de vernos obligados a añadir nosotros mismos un campo que haga las funciones de clave principal, estamos utilizando lo que se denomina una clave principal suplente. Mientras que si utilizamos un campo que ya existe como un hecho de la entidad, se denomina clave principal natural.

Otra posibilidad, es utilizar claves principales compuestas, estas claves principales son el resultado de unir dos columnas de nuestra tabla para formar una clave principal.

Para elegir una clave principal debemos valorar tanto los datos actuales como los datos futuros, ya que podemos seleccionar una columna como clave principal, porque actualmente nos sirve como tal ya que identifica cada registro, pero en un futuro pueden añadirse valores que se repiten para ese campo, siendo necesario utilizar otra columna como clave principal, o crear una compuesta.

Es de una gran importancia que definamos correctamente cada registro, ya que es la principal "herramienta" de la que se sirve el servidor de base de datos, para seleccionar la información que necesitamos. Mediante una clave principal el servidor conoce con que información queremos trabajar en cada momento. En caso de cometer errores y no tener claves principales que identifique de manera única cada entidad, tendríamos problemas con registros repetidos, ya que el servidor no sabría a que registro o entidad nos estamos refiriendo en nuestra actividad, y nos lanzaría continuas excepciones.

Con las claves principales identificadas, SQL Server nos ofrece unas características que nos ayudan a forzar la integridad de entidad:

- Introducir índices únicos en un campo para evitar la duplicación de datos por parte de los usuarios.
- Restricciones PRIMARY KEY o UNIQUE KEY
- Propiedad Identidad.

Todas estas características las iremos viendo más adelante.

4.3 Integridad de dominio

Ya hemos visto que la integridad de identidad permite obtener los datos almacenados en una base de datos. Con la integridad de dominio conseguimos controlar la información que guardamos en la base de datos. Como dominio, podemos entender como un conjunto de normas de negocio que gestionan la disponibilidad de datos en una determinada columna de una tabla. Por ejemplo que sólo podamos introducir nombres de fabricantes validados por un dominio de valores.

Tenemos una integridad de dominio básica, como no poder introducir letras en campos destinados para almacenar números. A mayor número de limitaciones, mejor aseguraremos el correcto funcionamiento de nuestra base de datos.

Estas normas o reglas de integridad de dominio pueden indicar que campos son necesarios tener obligatoriamente con valores (no se pueden dejar vacíos, NULL) para que la base de datos no tenga datos sin conectar en el caso de tener relaciones o dependencias entre tablas.

Las herramientas que nos ofrece SQL Server para asegurar la integridad de dominio y que iremos estudiando son:

- Tipos de datos
- Tipos de datos definidos por el usuario.
- Restricciones:
 - CHECK
 - DEFAULT
 - FOREIGN KEY
 - Reglas
 - NOT NULL

4.4 Integridad referencial.

Para ver este tipo de integridad tienes que pensar en las dependencias de tablas que hemos visto en la base de datos que hemos puesto como ejemplo de la empresa de venta de bebidas.

Hemos visto por ejemplo que para cada registro de la tabla Bebidas, teníamos un registro en la tabla Almacén. Y otro tipo de dependencias o relaciones que habíamos denominado relaciones "uno a muchos".

Estas relaciones se producen entre columnas comunes de las tablas que se relacionan. Como puede ser el nombre de una bebida, el de una distribuidora etc...

Con la integridad referencial tratamos de asegurar que las filas relacionadas entre tablas, no dejen de estarlo, o varíen esta relación cuando llevemos modificaciones a los datos. Con esta integridad limitaremos la actividad que puede realizar un usuario sobre la base de datos.

Vamos a ponernos en un ejemplo sencillo, nuestra tabla bebidas tiene una columna llamada "Distribuidoras", que se relaciona con nuestra tabla "Distribuidora" mediante esta misma columna. Ya hemos comentado este tipo de dependencia en este mismo tema. Llevando a cabo una integridad referencial, limitaremos las siguientes tareas a un usuario:

- El usuario no podrá cambiar el nombre de una distribuidora en una de las tablas, ya que si así lo hace, este valor no será el mismo en las dos tablas, y provoca que la relación quede rota. Un registro o varios (dependiendo de en que tabla realice esa modificación) se quedará sin su pareja y no podrá encontrar la relación.
- No podrá eliminar registros de la tabla distribuidora que se encuentren en la tabla Bebidas. Ya que todos aquellos registros de la tabla Bebidas que estuviesen vinculados a la Distribuidora eliminada se quedarán sin relación.
- No puede añadir registros nuevos en la tabla bebida cuyo campo Distribuidora no coincida con ninguna de las distribuidoras añadidos en la tabla Distribuidoras.

Por lo tanto, lo que debemos comprender de la integridad referencial es que existen relaciones entre tablas que deben permanecer invariables sea cual sea la actividad sobre ellas.

Para mantener esta integridad SQL Server nos ofrece:

- Restricciones FOREIGN KEY.
- Restricciones CHECK.

- Desencadenadores y procedimientos almacenados.

4.5 Integridad fijada por usuario.

Las tres integridades que acabamos de ver, están todas integradas en las bases de datos. Además no son exclusivas para SQL Server 2005, sino que las encontrarás en cualquier base de datos. Si bien puede que no estén completamente integradas y funcionales, son compatibles en cualquier ámbito.

La integridad que vamos a ver en este apartado, recoge todas las reglas que no están incluidas en ninguna de las integridades anteriores.

Un ejemplo de este tipo de integridad de usuario, sería obligar a que una determinada bebida siempre tenga dos tipos de envases. Este tipo de integridad no la cubre ni la de entidad, ni de dominio, ni referencial. Únicamente podemos controlarla mediante procedimientos almacenados, desencadenadores o reglas que se almacenen en la base de datos.

Esta integridad puede ser controlada también desde los programas clientes que conectan a la base de datos. Mediante el código de programación estos programas pueden comprobar antes de enviar los datos al servidor, si estos cumplen con un determinado juego de normas. De este modo el usuario estará limitado al utilizar el interface del programa, recibiendo los pertinentes avisos del modo de introducir los datos.

Ahora bien, aunque es completamente válido implementar esta integridad en el programa de cliente, lo más eficaz es colocarlo en el servidor, en la propia base de datos. Ya que no sabemos ni el número ni el tipo de programas que se conectará a la base de datos, y nosotros como desarrolladores tendríamos que incluir este tipo de restricciones en cada uno de los programas desarrollados, a parte del peligro que supondría aquellos programas clientes, que nuestra empresa a adquirido y a los que no tenemos acceso para modificar e incluir estas reglas.

4.6 Formas de normalización

Las formas normales definen una serie de normas o reglas que ayudan a organizar los datos en la estructura lógica de una base de datos.

Cada una de las formas que vamos a ir explicando heredan las reglas de su antecesora, así la forma normal C, incluye las reglas de las formas A y B. Para entender desde un sentido practico los diferentes modos de normalización, vamos a tomar como ejemplo la base de datos de la empresa vendedora de bebidas.

Recordamos la tabla Bebidas de esta base de datos:

Bebidas

nombre
fabricante
distribuidora
precio
envase
cantidad
fecha_caducidad
alcohólica
grados
lote

Vamos a suponer que tenemos esta tabla con los siguientes datos:

nombre	fabricante	distribuidora	precio	envase	cantidad	fecha_caducidad	alcohólica	grados	lote
Kas Naranja	Kas	Pardo	0.30	lata	0,33	22/07/2008	Falso	0	001
Coca-Cola	Coca Cola	Arros	0,45	vidrio	0,33	12/09/2009	Falso	0	002
La Navarra Etiqueta Verde	La Navarra	Enterlin	2,5	vidrio	1		Verdadero	25	075
Pacharan	Las Endrinas	Enterlin	3,00	vidrio	0,70		Verdadero	30	021

Endrinas									
----------	--	--	--	--	--	--	--	--	--

Debes tener claro que esta tabla contiene los datos sin ser normalizados, si bien son los datos que deseamos gestionar. A continuación, nos basaremos en esta tabla para ver como aplicar sobre ella el proceso de normalización. Por supuesto, sólo es una tabla de prueba, los campos que en un caso deberíamos controlar serían muchos más.

4.6.1 Forma Normal A

Las normas que debemos aplicar en la primera forma normal son muy básica y sencillas.

Regla: Cada uno de los campos de la tabla solo puede almacenar un tipo de datos, y además cada dato sólo se almacenar por separado, es decir individualmente.

Esta regla que hemos anunciado puedes encontrarla con la definición de la regla de datos atómicos o indivisibles.

Para entender mejor el significado de esta regla, vamos a explicar como podríamos quebrantarla. En la tabla que acabamos de presentar, vemos que estamos guardando los datos del fabricante y de la distribuidora en dos campos diferentes. Un modo de saltarnos la regla de la forma normal A, es guardar el nombre del fabricante y el nombre de la distribuidora en un único campo. De este modo no estamos cumpliendo la norma, ya que estamos guardando información de diferentes características en un único campo.

Otra manera de no cumplir la regla que estamos viendo es la repetición de un campo. Esta técnica es muy común en administradores que se están iniciando en el desarrollo de bases de datos.

Vamos a poner un ejemplo de este error tan común. Nos situamos de nuevo en nuestra tabla, ahora imaginamos que una bebida puede ser recibida en diferentes envases: lata, vidrio... El desarrollador puede pensar, como tenemos varias opciones de envase para una misma bebida, la forma más sencilla de cubrir esta información es tener los siguientes campos en la tabla: envase_A y envase_B. De este modo parece lógico que podemos guardar dos tipos de envases para cada bebida.

Este tipo de soluciones provoca bastantes problemas, el principal causante de estos problemas es la poca flexibilidad que nos ofrece esta estructura. Nuestra tabla dejará de cumplir con nuestras necesidades en el momento en que una bebida nos llegue con más de un tipo de envase, hemos nombrado como posibles envases lata y vidrio, pero también podrían llegar como plástico, por citar un ejemplo. Por lo tanto esta estructura lejos de ser sencilla es muy compleja, es imposible definir en un principio los diferentes tipos de envases que debemos controlar. Ahora podemos pensar, pues vamos a poner suficientes campos de modo que nuestras necesidades nunca superen

ese número de campos. Como ya imaginarás, tomar este tipo de soluciones es un error enorme. Esteremos reservando memoria sin ninguna necesidad, muchas bebidas vendrán en un sólo tipo de envase y tendremos campos vacíos que resultarán completamente inútiles.

Para solucionar el problema que hemos planteado como ejemplo, tenemos varias soluciones, como almacenar la misma bebida en diferentes registros de nuestra tabla, un registro por cada tipo de envase. Esta es una solución válida, pero muy poco o nada eficaz desde el punto de vista del rendimiento. La mejor solución es sacar "fuera" una tabla con los tipos de envase llamada Tipos_Envase por ejemplo y utilizar la integridad referencial para relacionarla con la tabla Bebidas.

Para aplicar la forma normal A, debemos pensar en que actividad vamos a realizar en la tabla, campos por los que vamos a ordenar, como vamos a recoger sus registros, si los vamos a agrupar, etc...

La tabla que hemos presentado al principio podemos decir que cumple la primera forma normal, ha excepción del caso que tengamos varios tipos de envases. Vamos a suponer que sólo nos llegan las bebidas en un único tipo de envase.

Si nos fijamos en los registros, vemos que tenemos información que se repite:

nombre	fabricante	distribuidor	precio	envase	cantidad	fecha_caducidad	alchólica	grados	lote
Kas Naranja	Kas	Pardo	0.30	lata	0,33	22/07/2008	Falso	0	001
Coca-Cola	Coca Cola	Arros	0,45	vidrio	0,33	12/09/2009	Falso	0	002
Pacharán La Navarra Etiqueta Verde	La Navarra	Enterlin	2,5	vidrio	1		Verdadero	25	075
Pacharán Endrinas	Las Endrinas	Enterlin	3,00	vidrio	0,70		Verdadero	30	021

Hemos resaltado en naranja algunos de los datos que se repiten. Vemos que la distribuidora Enterlin se repiten para las bebidas Pacharán La Navarra Etiqueta Verde y para Pacharán Endrinas. Si se diese el caso de que la empresa contrata otra distribuidora para la entrega de

pacharán, tendríamos que actualizar los dos registros. Esto puede provocar un fallo al realizar las actualizaciones si pasamos por alto uno de los registros, ya que tendríamos datos incoherentes. Por lo tanto tendremos que mejorar nuestra normalización.

4.6.1.1 Definición de claves principales.

Una clave principal contiene información de un registro de tal modo que gracias a esa información podamos distinguir ese registro de todos los demás, visto de otro forma, la información de la clave principal hace a un registro único e irrepetible en una tabla.

Una clave principal puede estar compuesta por una o varias columnas. En caso de estar formada por varias columnas es requisito indispensable que ninguna de esas columnas tenga información repetida en un mismo registro.

Como ya hemos visto, la clave principal garantiza la integridad de entidad en una tabla.

En una tabla podemos tener varias columnas que puedan formar parte de una clave principal, estas columnas reciben el nombre de **claves candidatas**. La función del administrador de la base de datos es definir estas claves candidatas como primer paso, para más adelante decidir de entre todas las claves candidatas cuales finalmente formarán parte de la clave principal. En nuestra tabla tenemos varios conjuntos de claves candidatas:

- Nombre
- nombre, fabricante
- nombre, fabricante, distribuidora

En cualquier tabla, podemos encontrar más de un grupo de claves candidatas. Veamos como seleccionar la clave principal más válida posible.

4.6.1.2 Selección de claves principales.

Para una correcta elección de la clave principal tenemos una serie de conceptos que nos ayudan:

- **Actividad:** Una buena clave principal puede ser aquella que tiene una información a la que los usuarios pueden acceder con facilidad, o que de la que tienen mayor conocimiento. Un buen ejemplo de este caso puede ser los números de factura.
- **Sencillez:** Hemos explicado que una clave principal puede tener varias columnas. La mejor opción es tratar de que el número de columnas que forman la clave principal sean las menos posibles. Si una clave principal es válida con dos columnas, añadir más columnas no incrementa la exclusividad de la clave principal (si es exclusiva con dos, será imposible aumentarla), lo único que provocamos si añadimos más columnas es bajar el rendimiento de las operaciones que se realicen con ellas.
- **Permanencia:** Una columna que pertenezca a una clave principal debe ser constante, es decir su valor no puede ser modificado o actualizado. Las claves principales juegan un papel importante en las relaciones entre tablas, y se supone que estas relaciones deben ser administradas con los valores de las claves principales, por lo tanto si permanecen constantes, ganaremos mucho en estabilidad.

4.6.1.3 Claves auxiliares

En una base de datos, nos podemos encontrar tablas de las cuales no podemos encontrar ninguna columna que sea clave candidata a formar una clave principal. Para cumplir con la forma normal A del proceso de normalización debemos incluir una clave principal como mínimo en nuestras tablas. La única solución que nos queda si se nos presenta una tabla de este tipo es incluir una columna extra en nuestra tabla, la cual no almacena información que defina la información que almacenamos, pero tiene la importante tarea de ser la clave principal que distinga un registro del resto, cumpliendo con la integridad de entidad, y nos ayude con las relaciones.

Por ejemplo podemos tener una agenda de teléfonos, con los campos nombre, teléfono y dirección. Esta persona puede cambiar de teléfono, dirección y hasta si nos ponemos drásticos, incluso de nombre. Solucionamos el problema añadiendo un campo con un número para cada uno de los contactos que tenemos en esta tabla, y el problema queda resuelto. Como vemos, este problema es muy frecuente en nuestras tablas y es una solución muy cómoda, incluso para tablas en las que dudamos que sus campos puedan cumplir con los tres conceptos que hemos definido para ayudarnos con la selección de claves principales.

4.6.2 Forma Normal B

La primera condición que debemos cumplir en la forma normal B, es que se cumplan las reglas que hemos fijado en la forma normal A.

La principal regla de esta segunda norma es:

Regla: Cada tabla sólo puede almacenar información de una única entidad.

Como hemos hecho antes, vamos a entender mejor estas condiciones, explicando como quebrantarlas. En nuestra base de datos de ejemplo, teníamos entre otras entidades los fabricantes y las distribuidoras. Para ambas entidades podríamos desear almacenar el nombre, la dirección y la ciudad. El primer error que podemos cometer como diseñadores de la base de datos es decidir que tanto los fabricantes como las distribuidoras pueden ser almacenadas en una misma tabla llamada Fabricantes_Distribuidoras. Esta decisión errónea la tomamos por el hecho de que ambas entidades comparten los mismos campos.

Esta decisión incumple la regla de la forma normal B que nos exige tener estas entidades en diferentes tablas, en caso de hacer caso omiso a esta regla nos encontraremos varios problemas. En el ejemplo que hemos tomado, tendríamos problemas para devolver la información de todos los fabricantes por un lado y por otro la de las distribuidoras.

Observa los datos de la siguiente tabla:

nombr	fabrica	localidad_fabric	distribuid	preci	enva	cantid	fecha_caduci	alchólic	grad	lot
-------	---------	------------------	------------	-------	------	--------	--------------	----------	------	-----

e	n	nte	ante	ora	o	se	ad	dad	a	os	e
Kas Naranj a	Kas	Zaragoza	Pardo	0.30	lata	0,33	22/07/2008	Falso	0	00 1	
Coca- Cola	Coca Cola	Zaragoza	Arros	0,45	vidri o	0,33	12/09/2009	Falso	0	00 2	
Pachar an La Navarr a Etiquet a Verde	La Navarra	Viana	Enterlin	2,5	vidri o	1		Verdad ero	25	07 5	
Pachar an Endrin as	Las Endrina s	Pamplona	Enterlin	3,00	vidri o	0,70		Verdad ero	30	02 1	

Esta tabla plantea el mismo problema que hemos identificado anteriormente si almacenamos los fabricantes y las distribuidoras en la misma tabla. Observa que entre otras cosas, estamos almacenando información relativa a las bebida y del fabricante, en una misma tabla.

Los campos fabricante y localidad_fabricante, almacena información relativa a la entidad Fabricante, que no depende en ningún caso con la entidad Bebida. La solución separarlo en dos tablas, quedando del siguiente modo:

nombre	fabricant e	distribuidor a	preci o	envas e	cantida d	fecha_caducida d	alchólica	grado s	lot e
Kas Naranja	Kas	Pardo	0.30	lata	0,33	22/07/2008	Falso	0	001
Coca- Cola	Coca Cola	Arros	0,45	vidrio	0,33	12/09/2009	Falso	0	002
Pachara	La	Enterlin	2,5	vidrio	1		Verdader	25	075

n La Navarra Etiqueta Verde	Navarra						o		
Pachara n Endrinas	Las Endrinas	Enterlin	3,00	vidrio	0,70		Verdader o	30	021

fabricante	localidad_fabricante
Kas	Zaragoza
Coca Cola	Zaragoza
La Navarra	Viana
Las Endrinas	Pamplona

Separamos las entidades en dos tablas y cumplimos con la segunda forma normal. Si observamos los registros de las dos tablas, vemos que la información de la tabla original permanece una vez desglosada en dos tablas. Incluso vemos que tenemos información que se repite. Recuerda que el proceso de normalización no consiste en minimizar el espacio de la base de datos, sino de mejorar el rendimiento y la funcionalidad.

4.6.2.1 Relaciones.

Vamos a definir las claves principales de estas dos tablas:

Tabla bebidas:

nombre	fabricante	distribuidor	precio	envase	cantidad	fecha_caducidad	alchólica	grados	lote
Kas	Kas	Pardo	0.30	lata	0,33	22/07/2008	Falso	0	001

Naranja									
Coca-Cola	Coca Cola	Arros	0,45	vidrio	0,33	12/09/2009	Falso	0	002
Pachara n La Navarra Etiqueta Verde	La Navarra	Enterlin	2,5	vidrio	1		Verdader o	25	075
Pachara n Endrinas	Las Endrinas	Enterlin	3,00	vidrio	0,70		Verdader o	30	021

Tabla Fabricantes:

fabricante	localidad_fabricante
Kas	Zaragoza
Coca Cola	Zaragoza
La Navarra	Viana
Las Endrinas	Pamplona

Hemos destacado en rojo las columnas que forman la clave principal de la primera tabla :nombre y fabricante, y en la segunda: fabricante.

Cuando dividimos una tabal en varias, como este caso, debemos ser capaces de combinar estas tablas para poder recoger los datos originales como si de una única tabla se tratara. Para ello, lo primero de todo buscamos un dato común que nos permite identificar uno (relación uno a uno) o varios (relación uno a varios) registros de una tabla a partir de un registro de la otra.

En nuestro caso el campo común es el nombre del fabricante.

El fabricante es la clave principal de nuestra tabla Fabricantes, y su correspondiente en la tabla Bebidas es la **clave externa o foránea**.

Cuando hemos identificado una clave externa y su correspondiente clave principal, nuestro servidor de base de datos es capaz de definir la integridad referencial que relacionan las dos tablas.

Ya conocemos las relaciones uno a varios y uno a uno, en función de los registros que se devuelven a partir de uno. Es posible crear relaciones varios a varios mediante una tercera tabla "auxiliar" que ayude a conectar dos tablas.

4.6.3 Forma Normal C

Para cumplir con la forma normal c debemos cumplir la forma normal B y la siguiente regla:

Regla: Todos los campos que no contengan una clave están obligados a depender de forma directa con la clave principal.

La definición de esta regla puede parecer complicada, pero es tan sencilla como cualquiera de las que hemos visto, o incluso más.

Una vez más, la explicamos comprobando como quebrantarla.

Observa la siguiente tabla Pedidos:

producto	unidades	precio	total
cuaderno	4	2	8
libreta	6	1,5	3

El camino más corto para cometer errores y no cumplir con esta tercera forma normal son los campos calculados. En la tabla, hemos incluido un campo Total que almacena el total de un pedido. Pero este campo es innecesario, ya que somos capaces de llegar a esa información a partir de los campos unidades y precio, en cualquier momento.

No debemos pensar, que este error que estamos cometiendo se limita a que almacenamos información extra y ocupamos espacio sin tener necesidad de ello. Además de esto, podemos tener problemas de coherencia, si por ejemplo modificamos las unidades del producto cuaderno y no recalculamos el valor del campo Total. Por lo tanto, debemos asegurarnos de que los campos de nuestras tablas son necesarios, y que no pueden ser obtenidos a partir de otros campos.

Otra característica importante de esta forma normal, es que es una gran ayuda para identificar que tablas deben ser divididas en otras a partir de la regla que nos exige que cada tabla sólo guarde información referente a una entidad.

4.6.4 Conclusión del proceso de normalización.

Podemos concluir el proceso de normalización cuando analizando nuestras tablas comprobamos que somos capaces de realizar una actualización sin tener que cambiar más de un dato para cada actualización.

Mencionar que el proceso de normalización ha ido evolucionando, los investigadores de bases de datos han incluido dos formas normales a las tres que hemos explicado, la forma normal D y E.

Estas dos últimas formas pertenecen a la normalización avanzada, y no son aplicables a la mayoría de las bases de datos, ya que es muy difícil alcanzar un nivel de complejidad tan alto como para tener que aplicarlas. Incluso se aconseja a los diseñadores que revisen sus estructuras cuando es necesario aplicar las formas normales D y E, ya que posiblemente si son necesarias es porque el desarrollador ha cometido errores en su diseño.

4.6.5 Desnormalización

Con el proceso de normalización hemos conseguido evitar al máximo la redundancia de datos, permitiendo realizar modificaciones de un modo cómodo. Para ello hemos indicado que desglosamos nuestra base de datos en tantas tablas como sea necesario.

De todas las reglas que hemos visto hay una que está por encima de todas, la lógica y la experiencia del administrador. Estas reglas no son obligatorias, son aconsejables en muchos casos y son de gran ayuda.

La lógica del programador puede indicarle que siguiendo la normalización de su base de datos, ha conseguido desglosar su estructura en tantas tablas con sus consiguientes relaciones. Esto puede provocar que la búsqueda de un registro tenga que llevarse a cabo a través de varias tablas y relaciones, con un rendimiento que deja mucho que desear.

Para solucionar esto, el desarrollador lleva a cabo el proceso de desnormalización, que tendrá consecuencias de redundancia de datos, pero que posiblemente sean necesario para la mejora del rendimiento.

Para conseguir alcanzar el término medio entre el proceso de normalización y desnormalización, el mejor medio es la experiencia. Se expone la base de datos a explotación como prueba piloto y se analiza la actividad que se realiza sobre ella, estudiando si los resultados se adaptan a las necesidades y cumplen con el rendimiento esperado, sino es así, gracias a estos estudios podremos ver que debemos modificar para mejorar nuestro diseño. SQL Server 2005 tiene la herramienta SQL Server Profiler que nos ayuda a realizar este tipo de análisis.

5 Herramientas para la normalización

El servidor SQL Server ofrece un grupo de herramientas que ayudan en el proceso de normalización. Gracias a estas herramientas podremos gestionar nuestras tablas de modo que los datos se añadan con la lógica deseada y que las modificaciones cumplan los requisitos deseados.

Con estas herramientas podremos indicar a nuestro servidor como debe administrar la normalización, y nos ahorraremos muchas líneas de código en aplicaciones para que se encarguen de ella. Esto supone una gran ventaja frente a otras bases de datos.

Vamos a explicar brevemente las herramientas que tendremos ocasión de ver como utilizarlas en próximos capítulos. Estas herramientas son:

- Identidad
- Restricciones
- Integridad en relaciones
- Disparadores

5.1 Identidad

Podemos tener una columna configurada como columna identidad, esta columna identidad es la manera más sencilla de garantizar la integridad de identidad.

La columna de identidad es una columna para la cual el propio servidor de base de datos se encarga de asignarle valores automáticamente. Por defecto, el primer valor es uno, y los siguientes registros van aumentando este valor de unidad en unidad. Aunque estos valores, son por defecto, veremos como se pueden modificar.

Una columna identidad es el mejor modo de añadir claves suplentes, como explicamos en el anterior capítulo, cuando una tabla no puede dar de modo natural una clave principal, será tarea nuestra añadir columnas que formen esa clave principal, pues el modo más eficaz es añadir columnas identidad.

Con este tipo de claves suplentes mejoramos considerablemente la relación entre tablas por columnas numéricas, bastante más eficaces que las claves principales formadas por textos.

5.2 Restricciones

Mediante las restricciones ponemos limitaciones a los datos que se van a introducir en la base de datos. Determinamos que datos son válidos para insertar en la columna de una tabla.

Tenemos las restricciones UNIQUE, DEFAULT y CHECK que fuerzan la integridad de identidad, dominio y la marcada por usuario. Y por otro lado contamos con las restricciones PRIMARY KEY y FOREIGN KEY para garantizar la integridad referencial en las relaciones.

5.2.1 UNIQUE

Esta restricción obliga a que todos los valores de una determinada columna no estén repetidos en otros registros. Si tenemos varias restricciones UNIQUE en una misma tabla, todas deben ser cumplidas a la vez para cada registro.

Con la restricción UNIQUE aseguramos la integridad de identidad de la tabla, ya que cumplimos con la norma de que cada registro es diferente al resto. Si aplicamos claves principales a una tabla, automáticamente se asigna esta restricción a esa columna.

No debes pensar que una columna identidad que se incrementa ella sólo automáticamente, es otro modo de tener una restricción UNIQUE, ya que se pueden dar casos en que tengamos valores duplicados, a no se que marquemos esa columna como clave suplente o principal. Puedes llegar a esta conclusión errónea si has trabajado con ACCESS, pero en SQL Server no es así.

5.2.2 DEFAULT

Como su propio nombre indica, esta restricción introduce un valor por defecto en una columna cuando no se indica ningún valor para insertar. Con esta restricción aseguramos la integridad de dominio, ya que aseguramos valores válidos para nuevos registros que se inserten.

5.2.3 CHECK

Esta restricción evalúa por medio de expresiones los valores que se insertan en una columna. Esta expresión, una vez que se evalúa devuelve un resultado, en función de si el dato es válido (Verdadero) o no (Falso), por lo tanto devuelve un valor booleano que indica si el dato tendrá permiso para ser ingresado o no.

Como puedes ver, nos ayuda a asegurar la integridad de dominio, y si vamos un poco más allá, también nos ayuda a asegurar la estabilidad de relaciones en configuraciones mucho más avanzadas.

5.3 Integridad en relaciones

Este tipo de integridad, denominada integridad referencial declarativa (DRI - Declarative Referential Integrity), es el proceso por el cual SQL Server fuerza de manera automática las relaciones entre tablas. Antes de aparecer este tipo de integridad para servidores SQL Server, era necesario desarrollar códigos para aplicaciones denominadas desencadenadores para cada tabla, y estos se encargaban de ejecutar una serie de acciones que asegurasen esta integridad, y siempre bajo la supervisión del administrador.

A este tipo de integridad llegamos ahora de manera automática, de un modo muy sencillo y con un rendimiento considerable, de modo que el administrador puede dedicarse a otras tareas. Para conseguir esta integridad tenemos dos tipos de restricciones: PRIMARY KEY y FOREIGN KEY.

5.3.1 PRIMARY KEY

La clave principal (PRIMARY KEY) nos permite asegurar la integridad de entidad (puesto que es única en cada registro) y por otro lado nos garantiza la estabilidad de las relaciones con otras tablas.

5.3.2 FOREIGN KEY

La restricción FOREIGN KEY, se conoce como la clave externa o foránea que ya hemos explicado. Y como ya sabes es la pareja de la restricción PRIMARY KEY, y juntas cumplen con la integridad referencial.

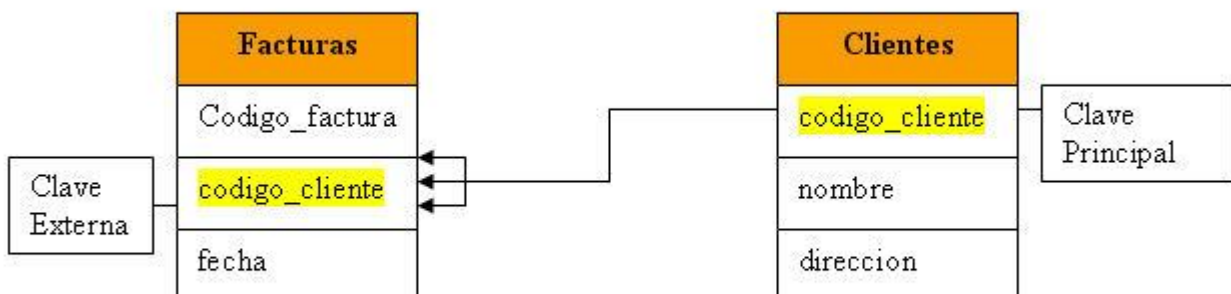
Una clave externa es una copia de la clave principal de la tabla principal, se inserta en la tabla que se pretende enlazar y con esto creamos la relación entre un par de tablas. Las claves externas pueden ser varias en una misma tabla, mientras que las principales deben ser únicas.

Para que esta relación que comentamos se cumpla, la clave principal que enlaza con la externa debe cumplir obligatoriamente que las dos columnas sean del mismo tipo.

5.3.2 Integridad referencial en cascada

Esta tipo de integridad que surgió con la versión 2000 de SQL Server, permite una serie de operaciones, que sólo pueden llevarse a cabo de este modo y no de otro.

Explicuemos porque a este tipo de integridad referencial se le añade el concepto de cascada. Imagina que tenemos una tabla que almacena las facturas emitidas para los clientes. Esta tabla entre sus campos contiene el campo `codigo_cliente`, que es la clave externa que se relaciona con la clave principal de la tabla clientes. Por lo tanto la tabla clientes tendrá un campo `codigo_cliente` que se relaciona con la tabla Facturas. Puedes verlo algo más claro en el siguiente gráfico:



Como ves, tenemos una relación uno a varios. Mientras en la tabla Clientes el campo `codigo_cliente` será único en cada registro, en la tabla Facturas, este mismo campo puede aparecer varias veces en diferentes registros, ya que emitimos varias facturas para el mismo cliente.

Esta relación representa una integridad referencial estricta, la cual no nos permite modificar el valor del código de cliente en la tabla clientes ya que de algún modo dejaría "huérfanos" a aquellos registros que anteriormente estaban relacionados al código de cliente que tratamos de modificar. Otra limitación que tenemos con esta integridad es que no nos permiten eliminar un cliente que tenga facturas emitidas en la tabla Facturas, por lo tanto no podemos eliminar un registro que tenga otros registros referenciados mediante estas relaciones.

Precisamente para solucionar estos problemas que hemos planteado tenemos la integridad referencial en cascada. Podemos añadir una serie de acciones que permita solventar estas complicaciones.

Podemos incluir actualizaciones en cascada, de modo que cuando modifiquemos el valor de una clave principal en la tabla principal, se modifiquen del mismo modo los valores de las claves externas en el resto de tablas enlazadas a la principal. En nuestro caso, al modificar el valor del campo `codigo_cliente` de un determinado cliente, este nuevo valor se cambiará para todos los registros referenciados en la tabla Facturas.

Igualmente podemos incluir eliminaciones en cascada, de tal forma que si eliminamos un registro de la tabla principal, se eliminen también los registros enlazados en la tabla subordinada. En nuestro caso, podríamos eliminar un cliente, y automáticamente se eliminarían todos sus registros de facturas.

Debes tener mucho cuidado a la hora de utilizar este tipo de relaciones en cascada, ya que si activamos por ejemplo la eliminación en cascada, corremos peligro de que un usuario no sea consciente de que perderá todos los registros vinculados a esa tabla.

Una clave principal, por norma general no cambiará su valor, por lo tanto no tiene sentido activar en esos casos tampoco la actualización en cascada.

5.4 Desencadenadores

Los desencadenadores representan aplicaciones que desarrollamos en lenguaje T-SQL y que se ejecutan, o mejor dicho, se "disparan" cuando sucede algún tipo de evento en una tabla. Los desencadenadores se llaman también disparadores o triggers.

En función del tipo de evento, tenemos los siguientes grupos de desencadenadores:

- Desencadenadores de inserción. Estos desencadenadores se ejecutan cuando se añade un registro o varios.
- Desencadenadores de actualización. Se ejecutan cuando se ha actualizado uno o varios registros.

- Desencadenadores de eliminación.

Con estos desencadenadores aseguramos la lógica de negocio y definimos la integridad de usuario. Antiguamente (versiones anteriores a SQL Server 2000), la integridad referencial en cascada tenía que implementarse mediante desencadenadores que permitiesen la actualización y eliminación en cascada.

Ejercicios

Ejercicio 1

En el siguiente ejercicio nos piden diseñar una base de datos para gestionar un videoclub. Después de varias reuniones con el dueño del negocio que nos ha contratado, podemos describir la información que el cliente necesita para poder gestionar los alquileres de su negocio:

- Título
- Productora
- País de la productora.
- Protagonista.
- País del protagonista.
- Director.
- País del director.
- Genero.
- Idioma.
- Duración.
- Nombre de cliente.
- Dirección de cliente.
- Teléfono de cliente.
- DNI de cliente.
- Fecha de alquiler.
- Precio.

A partir de estos datos, debemos indicar las diferentes identidades (tablas) que detectamos en el conjunto de ésta información.

Ejercicio 2

Una vez obtenidas las diferentes identidades, debemos indicar los diferentes campos de cada una de ellas y aplicar las relaciones que estimemos oportunas, con los cambios en las tablas que sean necesarios.

Nº- 3 Iniciación a la Administración.

1 Introducción

A lo largo de estos años Microsoft ha ido recogiendo y estudiando las diferentes solicitudes y sugerencias de administradores y desarrolladores de todo el mundo. Muchas de estas sugerencias solicitaban mejoras en el diseño y el modo de gestionar y administrar las tareas de SQL Server.

Microsoft se esforzó en tratar de cumplir con todas estas sugerencias y rediseño desde su servidor de base de datos. Como fruto de estas mejoras y otras muchas de rendimiento y estabilidad surgió SQL Server 2005.

Para usuarios veteranos y nuevos en administración de bases de datos con SQL Server, se quedarán sorprendidos en comprobar que con desde una única herramienta tenemos acceso a la gestión de casi todas las tareas de SQL Server 2005. Esta herramienta que engloba todas las funciones, es SQL Server 2005 Management Studio.

Para usuarios con experiencia ya, pueden ver que las antiguas herramientas como Analysis Manager, Administrador Corporativo y el Analizador de Consultas, están todas compactadas en SQL Server 2005 Management Studio.

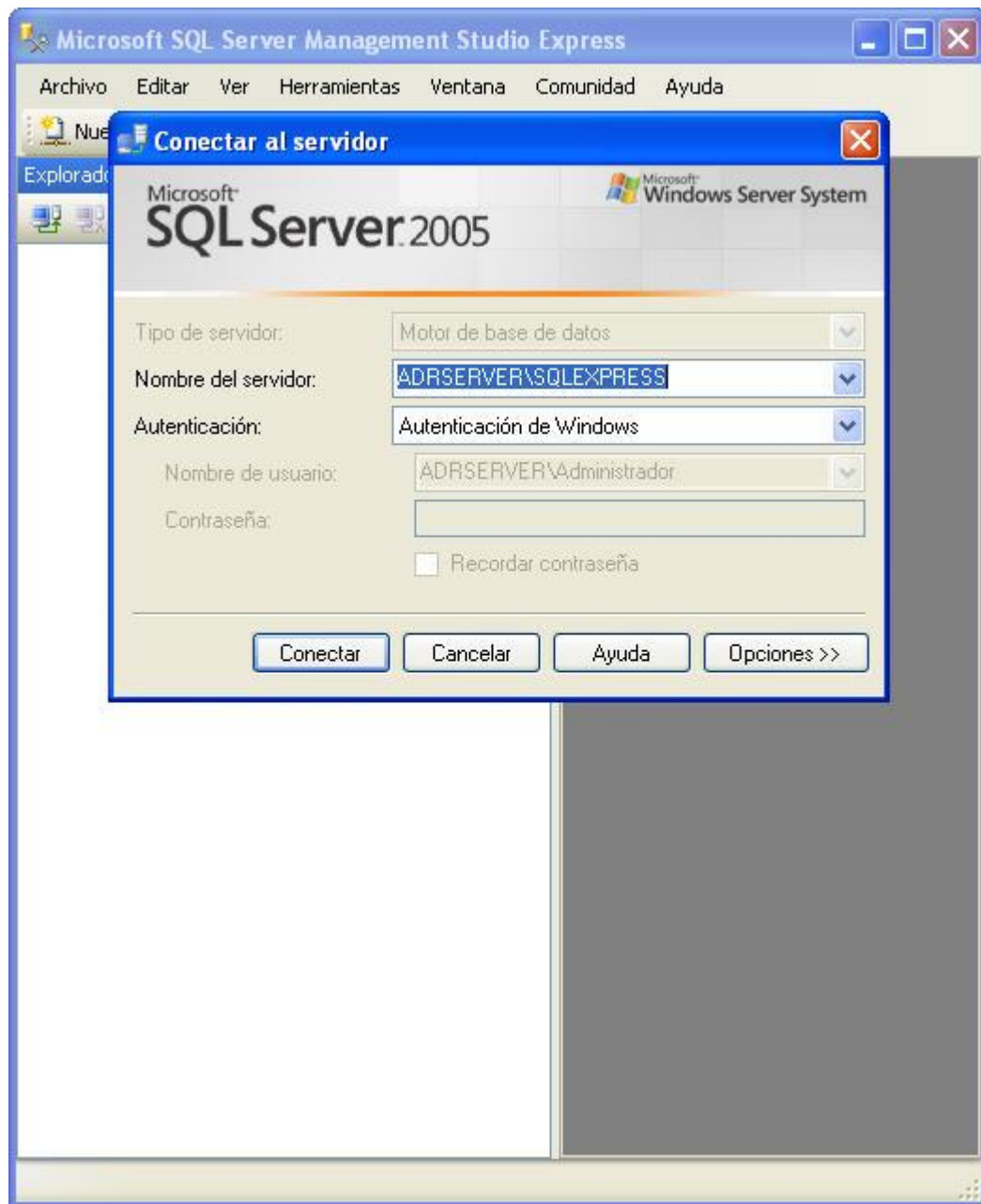
Si además eres desarrollador de software con .net, el entorno te resultará muy familiar al que se nos presenta en Visual Studio 2005.

Como es lógico la mayor parte del tiempo que dedicaremos como administradores o desarrolladores de bases de datos estará invertido en esta herramienta, por lo tanto debemos familiarizarnos al máximo a esta novedosa y potente herramienta.

Este capítulo pretende ser la primera toma de contacto del alumno con SQL Server 2005 Management Studio, presentaremos sus objetos, tareas etc...para al menos conocer donde se encuentran y para que sirven. En el resto de capítulos del curso aprenderemos a utilizarlos y sacarles partido, pero primero es importante tener una idea general de la herramienta que tenemos entre las manos y que podemos llegar a hacer con ella.

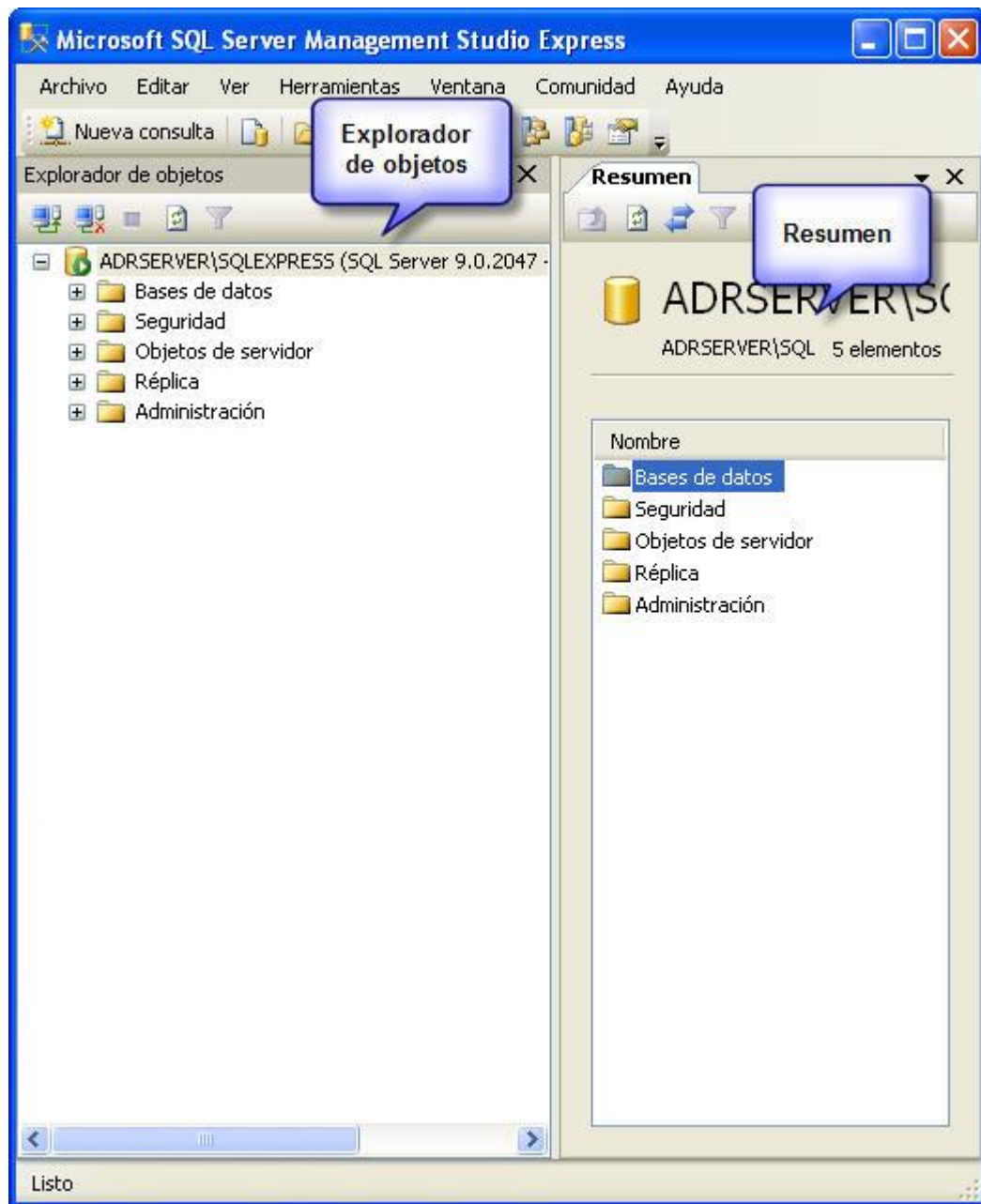
1.1 SQL Server Management Studio

Arrancamos la aplicación desde el menú de inicio tal y como vimos en el primer capítulo. Lo primero que haremos para acceder a la herramienta es conectarnos al servidor, para eso tendremos la conexión que hemos configurado durante la instalación del producto, así que seleccionamos esos parámetros: "Autenticación de Windows" y pulsamos en conectar:



Al conectarnos se nos presenta la primera pantalla de nuestra herramienta, y observamos que tenemos la pantalla dividida en dos ventanas:

- Explorador de objetos.
- Resumen.



Hay una tercera ventana que resulta muy útil y que por defecto es posible que SQL Server no muestre: Servidores registrados.



Para mostrar esta ventana vamos al menú "Ver" y seleccionamos "Servidores registrados", de este modo nuestra pantalla queda dividida en tres ventanas:

- Cada una de las ventanas puede cambiar a cualquier localización.
- Si tenemos la suerte de trabajar con monitores compartidos, tenemos la posibilidad de desacoplar la mayoría de ventanas y arrastrarla a cualquier posición.
- Las ventanas pueden ser ocultadas automáticamente, cuando una ventana permanece oculta se convierte en una pestaña en el borde de la ventana principal. Para mostrarlas de nuevo basta con colocar el puntero del ratón sobre esta ventana para que se muestre de nuevo. Para activar y desactivar esta posibilidad tenemos un botón representado por una chincheta. Otro modo de utilizar esta posibilidad es mediante el menú ventana->Ocultar automáticamente:

La ventana cuando se muestra:

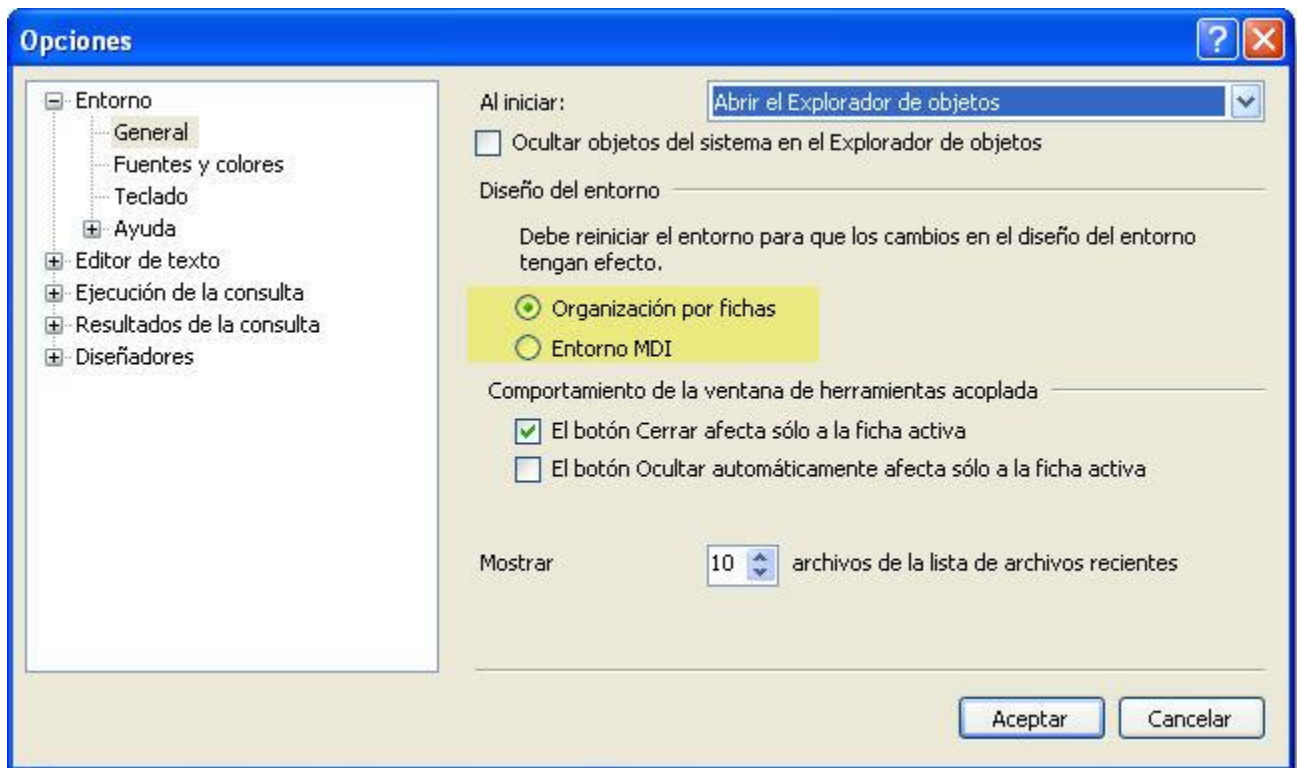
Si la ocultamos automáticamente, se convierte en pestaña al borde la pantalla principal:

Para activar y desactivar la posibilidad de ocultar automáticamente tenemos el botón chincheta:

Botón	Descripción
	Activada la opción "Ocultar Automáticamente".
	La ventana permanece fija, tenemos desactivada la opción "Ocultar automáticamente".

- Tenemos la posibilidad de configurar el entorno para que nos muestre la información como fichas. Cada elemento aparece como una ficha en la localización que se encuentre, o bien como una interfaz de múltiples documentos, conocida esta opción como MDI de modo que cada

documento está en su propia ventana. Para optar a esta configuración tenemos el menú Herramientas->Opciones->Entorno->General y marcar la opción Organización por fichas o Entorno MDI:

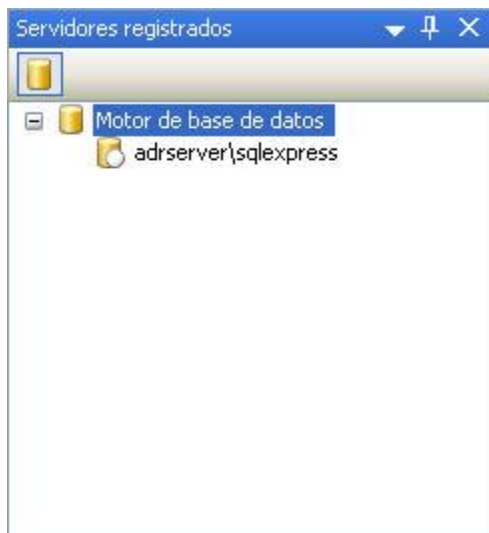


Los diferentes componentes de Management Studio están configurados para trabajar como un todo. Todas las herramientas están entrelazadas, lo que hace de Management Studio una herramienta muy eficaz.

Otra opción que debemos comentar es que Management Studio no tiene porque ser instalado en el mismo servidor donde estamos explotando nuestra base de datos. De hecho el modo más común de trabajar es tener SQL Server 2005 instalado en el servidor para explotar la base de datos, y en un equipo a parte y personal del administrador tener instalado Management Studio, de modo que el administrador pueda trabajar con la base de datos sin la necesidad de trabajar en el servidor. Además este método de trabajo es aún más eficaz, ya que desde Management Studio podemos estar conectados a varios servidores y administrar varias bases de datos instaladas en diferentes servidores y todo esto desde el ordenador personal del administrador.

2 Servidores registrados.

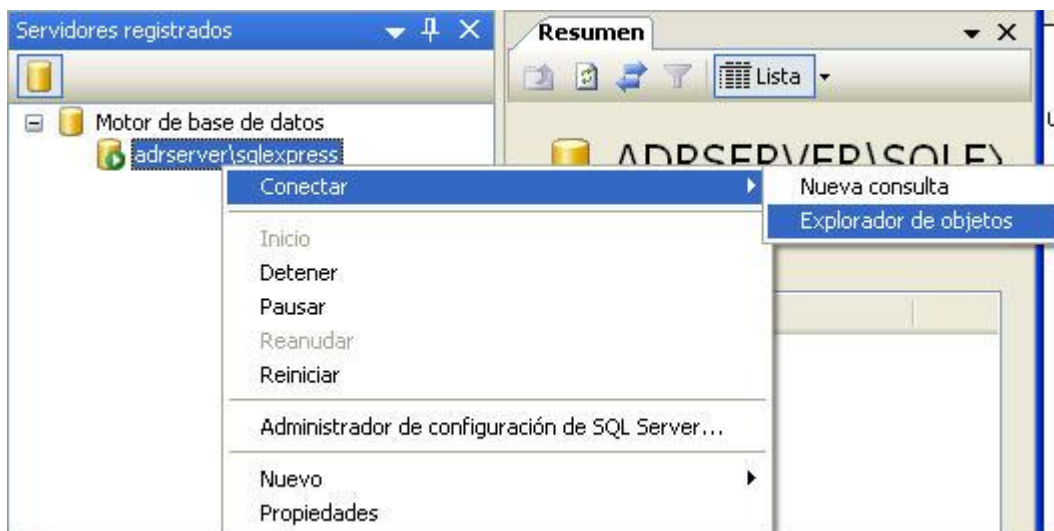
Hemos visto en el anterior capítulo que la ventana que nos presenta la información relativa a los Servidores registrados puede permanecer oculta por defecto.




Desde esta ventana podemos organizar los servidores "favoritos" a los que accedemos con mayor frecuencia. Entre otras opciones tenemos la posibilidad de crear grupos de servidores para facilitar la búsqueda de uno de ellos, en caso de que trabajemos con muchos servidores a la vez.

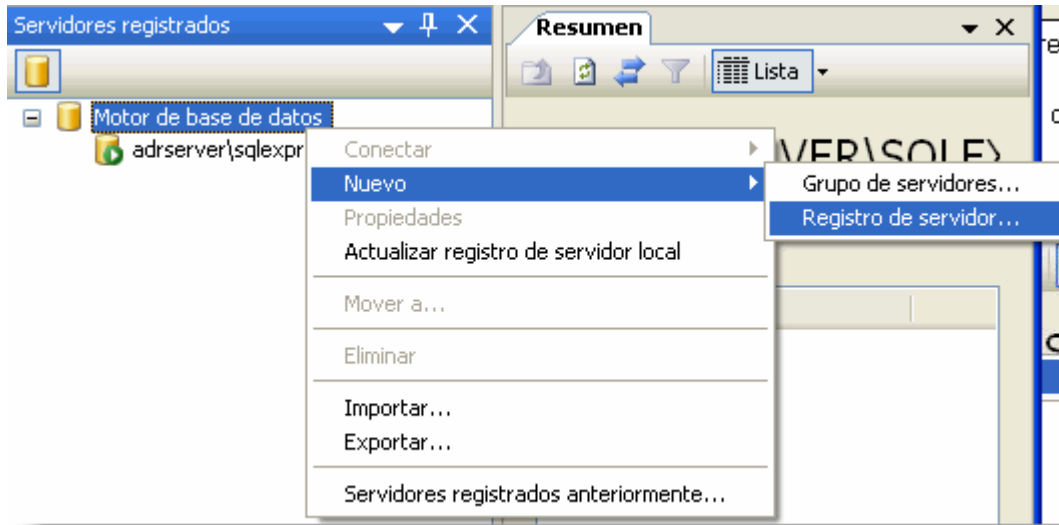
En el caso de SQL Server 2005 Express Edition tenemos limitados los tipos de servidores al motor de base de datos que es con el que nos centramos en este curso. En versiones más avanzadas, aparecerían el resto de tipos de servidores que nos ofrecen.

Desde esta ventana podemos pinchar con el botón derecho sobre el servidor y seleccionar Conectar->Explorador de objetos, y de este modo nos conectaremos con ese servidor en concreto:



Una vez que estamos conectados, vemos que el icono del servidor aparece con un triángulo con fondo verde que indica que estamos conectados actualmente a ese servidor: 

Tenemos la posibilidad de añadir un nuevo registro de servidor, para ello pulsamos con el botón derecho en el icono del tipo de servidor del cual queremos crear un nuevo registro. En nuestro caso sólo tenemos el motor de base de datos, pulsamos con el botón derecho y seleccionamos Nuevo->Registro de servidor...



Dentro de la ventana de Nuevo registro de servidor tenemos dos pestañas tal y como puedes ver en las siguientes figuras:

Pestaña General:

The image shows a Windows-style dialog box titled "Nuevo registro de servidor" (New server registration). It has two tabs: "General" and "Propiedades de conexión" (Connection properties), with the latter being the active tab. The dialog is divided into two main sections: "Inicio de sesión" (Login) and "Servidor registrado" (Registered server).

Inicio de sesión

Escriba el nombre del servidor o selecciónelo en la lista desplegable.

Tipo de servidor: Motor de base de datos

Nombre del servidor: [Empty text box]

Autenticación: Autenticación de Windows

Nombre de usuario: ADRSERVER\Administrador

Contraseña: [Empty password box]

☐ Recordar contraseña

Servidor registrado

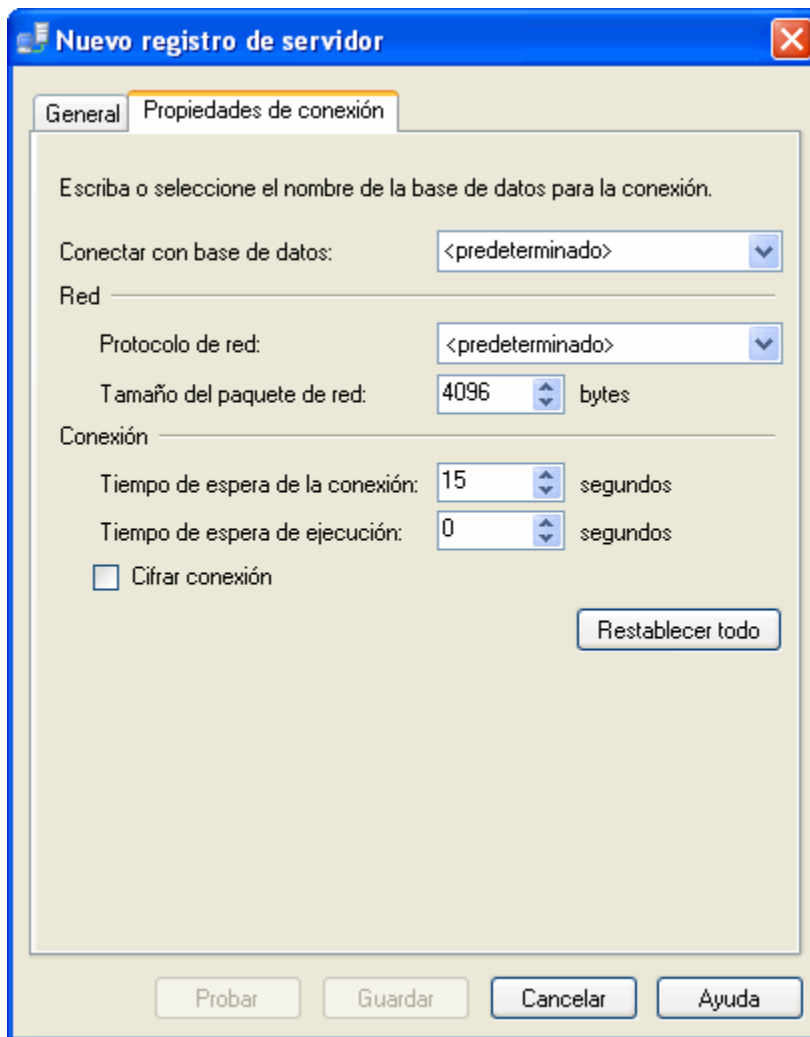
Puede reemplazar el nombre del servidor registrado por un nuevo nombre y una descripción de servidor opcional.

Nombre del servidor registrado: [Empty text box]

Descripción del servidor registrado: [Empty text box]

Buttons at the bottom: Probar, Guardar, Cancelar, Ayuda.

Pestaña propiedades de conexión:



Desde estas ventanas tenemos la posibilidad de configurar las siguientes tareas:

Pestaña General:

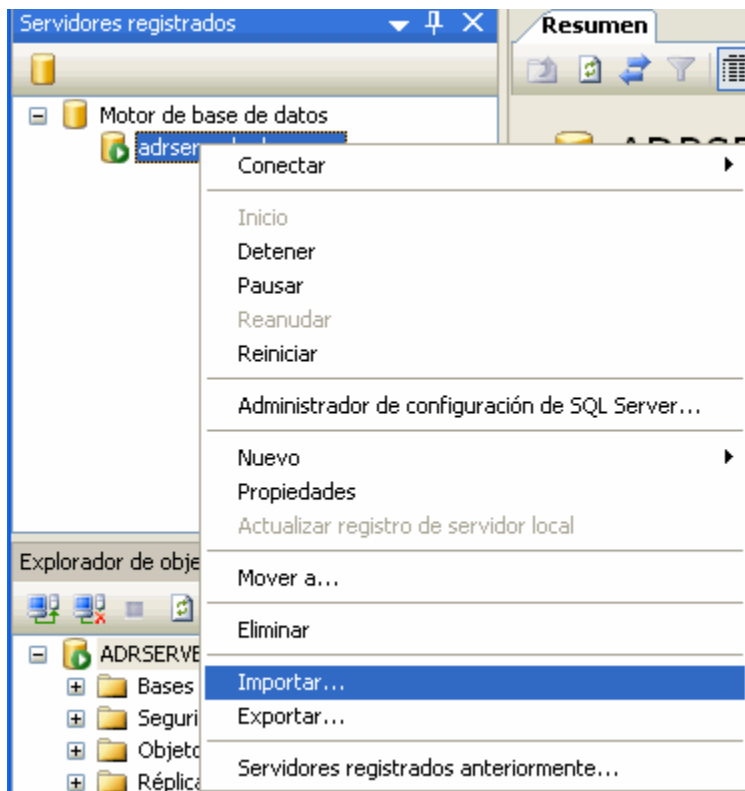
- Modificar el modo de autenticación: Windows o SQL Server
- En caso de seleccionar el modo SQL Server de autenticación podemos introducir el nombre de usuario y la contraseña.
- Incluir o modificar el nombre del servidor registrado y añadir una descripción si así lo deseamos. La descripción no es obligatoria.

Pestaña Propiedades de conexión:

- Indicar la base de datos con la que deseamos conectar directamente al conectar con el servidor.
- Seleccionar el protocolo de red con el cual conectamos.
- Definir el tamaño del paquete de red.

- Especificar el tiempo de espera de conexión y ejecución. Se trata del tiempo que transcurrirá para conectar, una vez pasado este tiempo el Management Studio entenderá que ocurre algún problema que no permite la conexión y lanzará un error.
- Activar o desactivar el cifrado de la conexión.

Para agilizar el proceso de registro de servidores, tenemos la opción de importar y exportar las características de conexión definidas de un servidor a otro, y viceversa. Como es razonable, podremos importar y exportar estos registros siempre y cuando se traten del mismo tipo de servidores. Esta información de registro se almacena en archivos. Para llevar a cabo este proceso, es tan sencillo como pinchar con el botón derecho en un servidor o en un grupo de servidores y seleccionar del menú la opción Importar o Exportar, dependiendo de la tarea que queramos realizar:







En la ventana que se nos muestra seleccionamos el archivo de registro y seleccionamos a que servidor o grupo de servidor queremos aplicarlo, mediante el árbol de servidores:



Para eliminar un servidor o un grupo, lo haremos desde el menú que emerge al pulsar con el botón derecho sobre el servidor a eliminar, nos mostrará un ventana de confirmación para evitar errores.

Para modificar el nombre o las propiedades de conexión de cualquier servidor podemos pulsar con el botón derecho y seleccionar la opción propiedades.

Para finalizar vamos a mostrar una tabla con los diferentes iconos que podemos encontrar en los servidores y lo que significan:

Icono	Descripción
	No es posible realizar una conexión con el servidor.
	El servidor está en uso actualmente.
	El servidor está pausado.
	El servidor está detenido.

2.1 Agrupar servidores.

Con la versión SQL Server 2000 apareció un nuevo concepto: Grupos de servidores.

El objetivo de los grupos de servidores es meramente organizativos. Es una utilidad administrativa que ayuda al desarrollador a ordenar varios servidores por grupos, y no tiene ninguna influencia sobre la actividad y la estructura de nuestros servidores. Físicamente no agrupa, ni conecta los servidores que pertenezcan a un mismo grupo.

Con los grupos de servidores se pueden mostrar los servidores con diferentes nombres a sus reales. La ventaja de esta característica es que podemos tener nombres reales que sean complejos y utilizar nombres más descriptivos para su administración.

Además los grupos de servidores pueden anidarse, es decir, podemos tener un grupo de servidores dentro de otro. Y podemos tener también un servidor en más de un grupo.

Imagina que nos encargan administrar una gran empresa dividida en diferentes departamentos: Administración, Recursos Humanos y Producción.

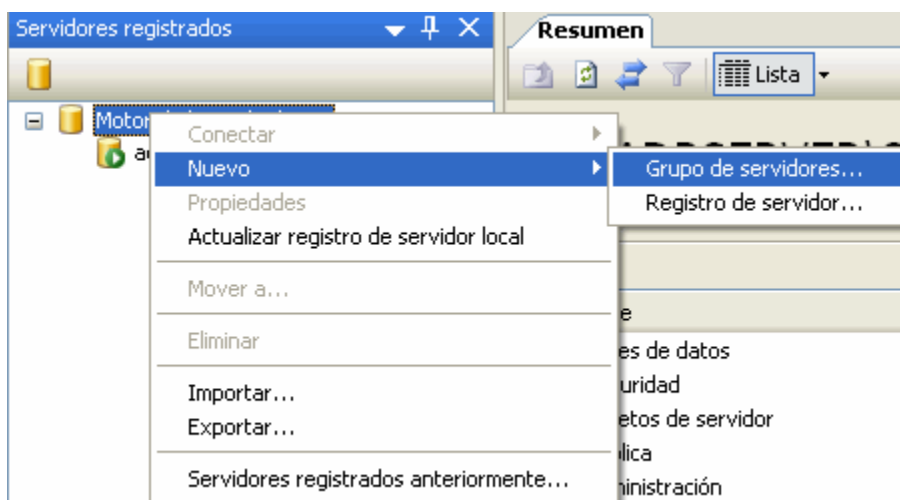
Como se trata de una gran empresa, tenemos además oficinas en Madrid y Barcelona. Cada departamento tiene su propio servidor con su correspondiente servidor de base de datos que guarda la información de la actividad realizada por su departamento.

En este caso podíamos organizar de varios modos nuestros servidores. El más lógico sería crear dos grupos principales a los que llamaremos Madrid y Barcelona.

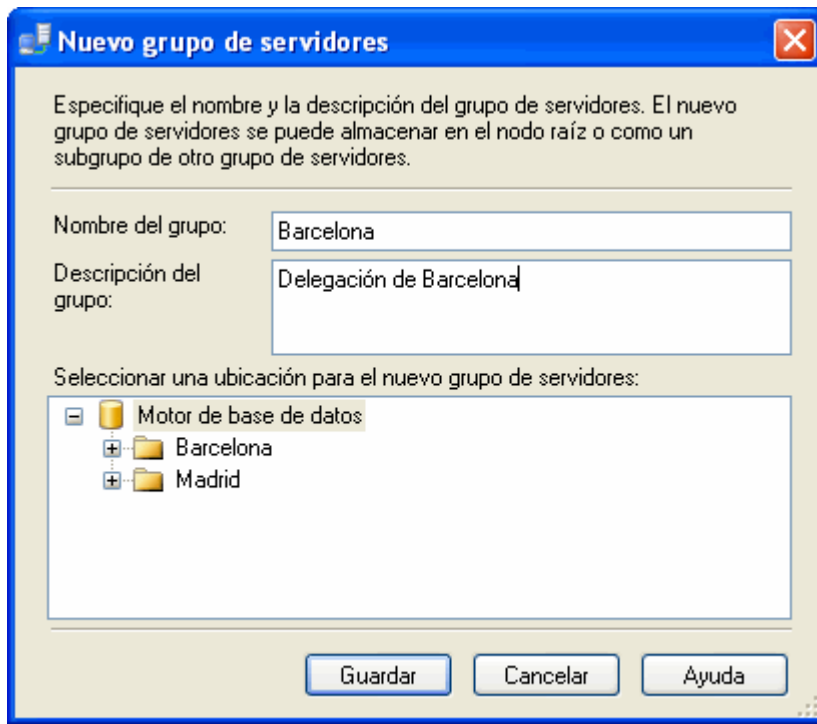
Y a su vez dentro de los grupos Madrid y Barcelona crear otros tres grupos que almacenen los servidores de cada departamento.

Vamos a crear este organigrama de servidores co SQL Server 2005:

Para crear un nuevo grupo pinchamos con el botón derecho en el lugar donde queremos crear un grupo y seleccionamos Nuevo->Grupo de servidores, podemos hacerlo en la raíz (sobre nuestro motor de base de datos) o sobre otro grupo de servidores si deseamos anidarlo.

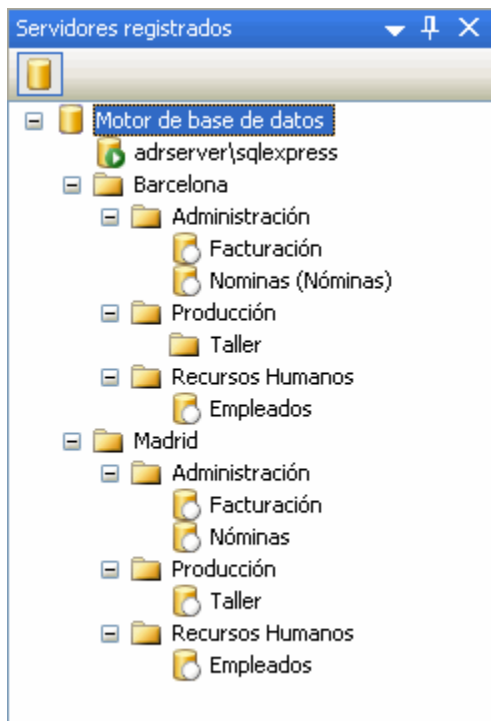


Esta operación nos mostrará una ventana donde nos podremos indicar un nombre para el grupo de servidores y una breve descripción para el grupo. Además nos muestra un panel con la estructura organizativa de nuestros servidores, como puedes ver en la siguiente figura:



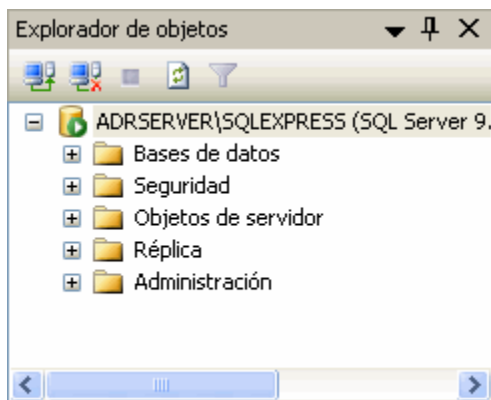
Una vez creados los grupos iremos añadiendo los registros de servidores tal y como hemos visto en capítulos anteriores.

Siguiendo estas operaciones, y anidando nuestros grupos finalmente conseguimos el siguiente esquema que nos muestra la ventana de servidores registrados:








3 Explorador de objetos

Otra de las ventanas principales que nos muestra Management Studio es el Explorador de Objetos.




El explorador de objetos se encuentra conectado a los diferentes tipos de servidores que tenemos. Los elementos que nos ofrece SQL Server 2005 varían en función del tipo de servidor, pero hay características de desarrollo y herramientas de administración comunes para todos.

En esta ventana podemos ver una barra de herramientas que nos permite realizar unas determinadas tareas. En la siguiente tabla mostramos estos botones con la tarea que tienen asignada:

Icono	Descripción
	Realiza una conexión con el servidor
	Realiza una desconexión del servidor.
	Detiene un proceso, sólo estará disponible durante la ejecución de uno
	Actualiza la lista de elementos de la carpeta que tengamos seleccionada.
	Realiza un filtro o selección de objetos, permanece en gris cuando no es posible filtrar.

3.1 Conectar a un servidor

Pulsando en el botón destinado a la conexión con un servidor (), nos muestra la pantalla de conexión que ya conocemos:



En esta ventana, tenemos que dar el nombre del servidor con el que deseamos conectar, y seleccionar el tipo de autenticación con el que realizaremos la conexión. Si seleccionamos Autenticación de Windows, podremos realizar la conexión, mientras que si elegimos la opción Autenticación SQL Server, deberemos introducir el nombre de usuario y la contraseña. Esta opción dependerá del tipo de registro que hayamos definido durante la instalación, o de la modificación que hayamos realizado sobre el registro del servidor en sus propiedades.

Si pulsamos sobre el botón opciones, se nos abre una nueva ventana que puedes ver en la siguiente figura:



Vemos que tenemos dos pestañas, la primera de ellas "Inicio de sesión" nos muestra la ventana de conexión de la que partimos. Mientras que en la pestaña "Propiedades de conexión" nos muestra la ventana que ves en la anterior figura. Esta ventana ya la conocemos, y la información que podemos introducir ya la hemos visto en la ventana de propiedades que hemos comentado en el anterior capítulo.

3.2 Carpetas del explorador de objetos.


La información y los elementos que tenemos en el explorador de objetos se organiza en carpetas con estructura de árbol.

Nota: Este capítulo muestra el "Explorador de objetos" a modo de presentación. En los siguientes capítulos aprenderemos a trabajar con estos objetos mediante el lenguaje SQL y con SQL Server 2005

para alcanzar un nivel avanzado.

Cuando expandamos una de estas carpetas, el explorador de objetos recibe información del servidor del contenido de la carpeta para poder mostrarlo. Esta petición sólo se realiza la primera vez que expandimos una carpeta, por lo tanto, si hemos realizado alguna modificación en algún componente de alguna lista desde la primera vez que expandimos la carpeta, esta modificación no se mostrará en ella. Para que se muestre esa información "nueva", debemos seleccionar esa carpeta y actualizarla, mediante la opción "actualizar" del menú que se muestra con el botón derecho, o mediante el botón actualizar de la barra de herramientas.

Si trabajamos con bases de datos empresariales de gran tamaño, es posible que tengamos en una carpeta un listado de objetos tan grande que no nos permita trabajar con comodidad. Lo más frecuente es que de ese listado, sólo nos interese trabajar con unos pocos en un determinado momento.

Para facilitar esta tarea, tenemos la opción de mostrar sólo aquellos objetos que nos interesa, mediante el botón de filtrado: 

Hay carpetas que no permiten realizar un filtro, nos situamos en una de las carpetas que lo permitan y pulsamos sobre el botón, para mostrar la ventana de "Configuración de filtro del explorador de objetos":

Configuración del filtro del Explorador de objetos

Servidor: ADRSERVER\SQLEXPRESS

Base de datos: Northwind

Criterios de filtro:

Propiedad	Operador	Valor
Name	Contiene	
Schema	Contiene	
Creation Date	Es igual a	

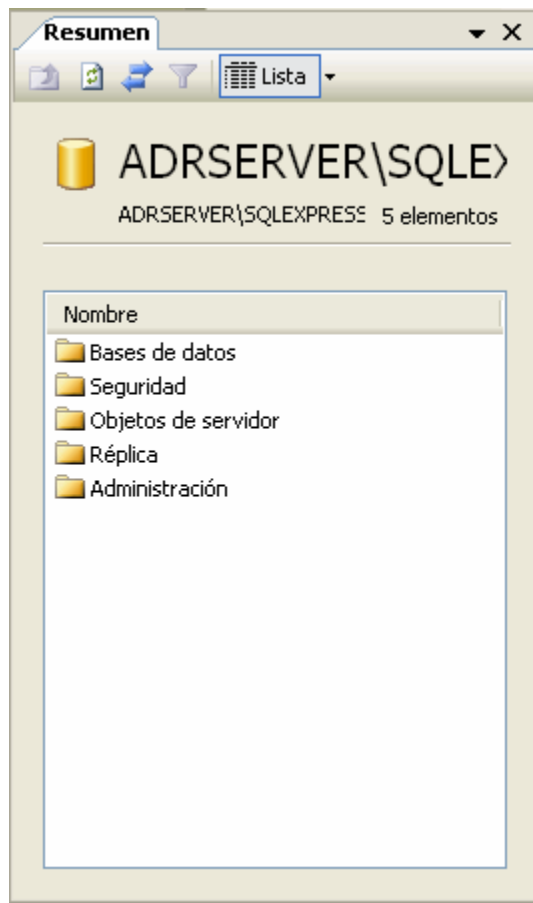
Incluir o excluir objetos basados en el nombre o en parte de un nombre.

Borrar filtro Aceptar Cancelar Ayuda





Desde esta ventana nos permite realizar filtros por el nombre, esquema (no siempre está disponible) y la fecha de creación, teniendo la opción de elegir el operador de comparación para el filtro (Es igual a, Contiene y Entre).


3.3 Pestaña Resumen

La tercera ventana que nos muestra Management Studio aparece por defecto con la pestaña "Resumen":



Esta pestaña es utilizada por el Explorador de Objetos para mostrar información del objeto que tengamos seleccionado. Como puedes ver, esta pestaña tiene su propia barra de herramientas con los siguientes botones:

Icono	Descripción
	Sube un nivel en el árbol de carpetas.
	Actualiza el elemento seleccionado.
	Sincroniza la información del servidor.
	Filtra el listado, en caso de ser posible.

	<p>Tipo de vista para mostrar la información:</p> <ul style="list-style-type: none"> • Lista: Listado de objetos de la carpeta seleccionada. • Detalles: Muestra información por diferentes categorías del objeto seleccionado.
---	---

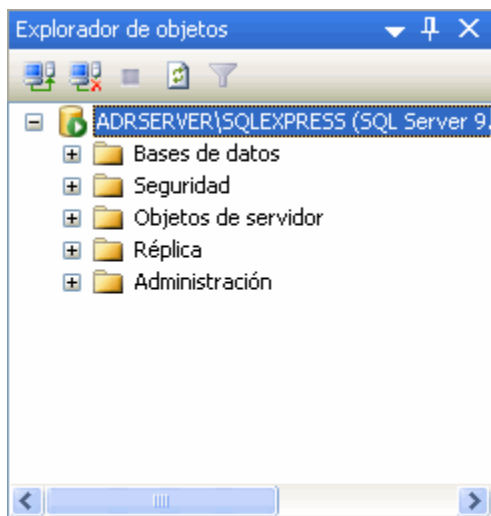
El modo de navegación por las carpetas de la pestaña "Resumen" es muy parecido al explorador de Windows.

3.4 Carpetas principales.

En el explorador de objetos de SQL Server 2005 Express Edition (Versión avanzada) tenemos las siguientes carpetas principales:

- Bases de datos.
- Seguridad.
- Objetos de servidor .
- Réplica.
- Administración.

En este curso iremos viendo la mayoría de objetos que encierran estas carpetas.



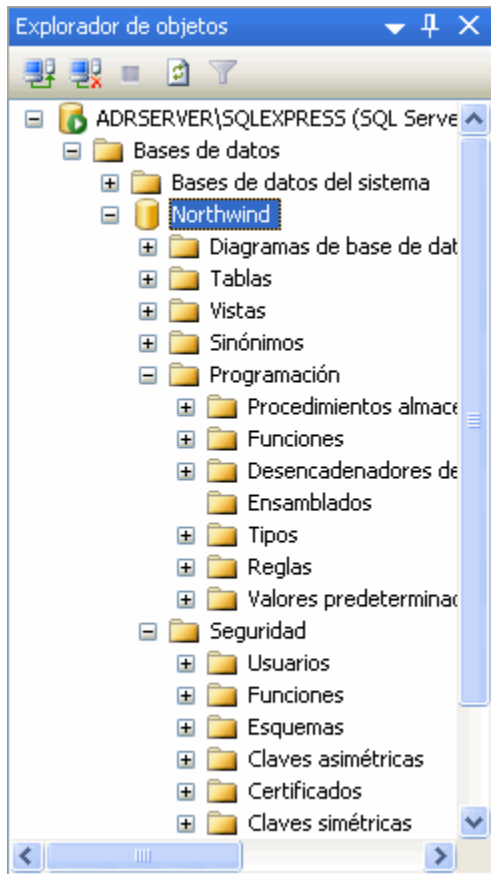
3.5 Carpeta de bases de datos.

En esta carpeta podemos encontrar las bases de datos del sistema y las de usuarios que vayamos creando nosotros.

Dentro de las bases de datos de usuarios podemos encontrar carpetas anidadas agrupadas por los objetos que contiene:

Árbol de Bases de Datos.		
Base de datos de usuario.	Diagramas de bases de datos.	
	Tablas	
	Vistas.	
	Sinónimos.	
	Programación.	<ul style="list-style-type: none">• Procedimientos Almacenados.• Funciones.• Desencadenadores de bases de datos.• Ensamblados.• Tipos.• Reglas.• Valores predeterminados.
	Seguridad.	<ul style="list-style-type: none">• Usuarios.• Funciones.• Esquemas.• Claves asimétricas.• Certificados.• Claves simétricas.

Todos estos objetos puedes verlos al extender una base de datos de usuario. En este caso mostramos en la siguiente figuras los nodos principales de la base de datos Northwind:



Hasta aquí llegamos con la primera toma de contacto con SQL Server Management Studio, a continuación tendremos dos temas (4, 5 y 6) para aprender a programar con SQL y como realizar este aprendizaje con SQL Server 2005.

A partir del tema 7, retomaremos de nuevo Management Studio, para profundizar con los objetos de las bases de datos. Explicaremos con un nivel mucho más avanzado como trabajar con las herramientas de SQL Server 2005 que a lo largo de estos tres primeros capítulos no hemos visto, o únicamente se han mencionado. Pero no abandonaremos el lenguaje SQL, ya que iremos viendo como realizar muchas de estas tareas mediante este lenguaje de programación, de modo que podamos realizar estas tareas en cualquier otro servidor de base de datos del mercado.

En el resto de capítulos seguiremos avanzando con SQL Server 2005, completando su aprendizaje. Una vez estudiadas todas las lecciones, tendremos los conocimientos necesarios para ser Administradores y desarrolladores de bases de datos, utilizando sin problemas herramientas profesionales de SQL Server 2005.

Ejercicios

Ejercicio 1

En este ejercicio vamos a organizar los servidores que tenemos en dos oficinas diferentes. En una de ellas tendríamos tres servidores y en la segunda dos.

Para organizar las conexiones, crea los grupos de servidores que creas conveniente y los registros necesarios.

