Este texto describe al lector las características del sistema gestor de bases de datos relacionales SQL Server 7.0, relacionadas con su instalación, mantenimiento y administración.

Entre algunos de los temas abordados, se encuentran: La propia instalación del producto, configuración de usuarios, permisos, estrategias de copia de seguridad, transferencia de datos desde y hacia bases de datos distintas de SQL Server, etc., que permitirán al lector obtener los conocimientos para realizar todas las funciones propias de un administrador de bases de datos.

ADMINISTRACIÓN DE BASES DE DATOS CON SQL SERVER 7.0

LUIS MIGUEL BLANCO

GRUPO Eidos

ISBN 84-88457-14-6

Administración de Bases de Datos con SQL Server 7

Luis Miguel Blanco

Responsable editorial
Paco Marín (fmarin@eidos.es)

Coordinación de la edición
Antonio Quirós (aquiros@eidos.es)

Autoedición
Magdalena Marín (mmarin@eidos.es)
Luis Miguel Blanco (lmblanco@eidos.es)

Grupo EIDOS
C/ Téllez 30 Oficina 2
28007-Madrid (España)
Tel: 91 5013234 Fax: 91 (34) 5017824
www.grupoeidos.com/www.eidos.es
www.LaLibreriaDigital.com

Índice

1

Introducción a la administración de bases de datos con SQL Server 7.0

Recopilación de datos, obtención de información

La actual demanda de información crece a un ritmo cada vez más acelerado. Si bien es cierto que disponemos de multitud de medios que nos proporcionan una ingente cantidad de datos, corremos paradójicamente, el riesgo de estar más desinformados que nunca, ya que a partir de todos los datos disponibles, tenemos que filtrar y extraer lo que verdaderamente nos será útil, la información.

Esta cuestión nos lleva a la conclusión de que tan importante como obtener los datos y disponer de un medio para guardarlos, es el que dicho medio los organice y seleccione conforme a nuestras necesidades. La solución a este problema la proporcionan las aplicaciones denominadas gestores de bases de datos.

Gestores de bases de datos y administración

Un gestor de bases de datos se puede describir de un modo muy simple, como un contenedor de información, que organiza la misma en base a una serie de reglas. Dicha información puede ser manipulada mediante un conjunto de instrucciones que permitirán al usuario consultar y modificar los datos contenidos.

La administración de un gestor de datos, por otra parte, se puede definir como el conjunto de labores cuyo objetivo es conseguir un rendimiento óptimo del sistema de bases de datos, de forma que la

información esté en todo momento disponible y con el menor tiempo de espera posible para el usuario que la solicita. Alcanzar este objetivo depende de dos aspectos:

- **Elementos proporcionados por el gestor para la manipulación de datos.** Aquí podemos situar las diferentes herramientas, asistentes, etc., que nos proporciona el gestor para procesar los datos.

- **Estrategias de gestión de datos.** En este punto se enmarcan las diferentes políticas a aplicar para el uso de las herramientas proporcionadas por el gestor, de manera que no interfieran entre ellas al rendimiento conjunto del sistema y consigamos que las condiciones de funcionamiento del mismo sean óptimas el mayor tiempo posible.

Cuando nos referimos a administración, también debemos dedicar nuestros esfuerzos a optimizar la parte hardware del sistema, de forma que los equipos de la red en la que está implantada la base de datos estén en las mejores condiciones posibles, para que no provoquen fallos en el intercambio de información.

También utilizaremos el término servidor de bases de datos, para referirnos al programa gestor, debido a la arquitectura cliente-servidor que utilizan este tipo de sistemas y que veremos más en detalle posteriormente.

¿Cuándo es necesaria la administración?

Es obvio que una pequeña empresa no necesita una administración de datos dedicada. La aplicación o aplicaciones que utilicen en su sistema informático para la gestión del negocio, podrán cubrir sobradamente tales necesidades. Los usuarios de tales aplicaciones realizarán dichas labores sin una dedicación excesiva de tiempo.

Sin embargo, en una gran compañía, con un elevado intercambio de información entre clientes, sucursales, proveedores, etc., se hace necesaria la implantación de un medio que controle tal cantidad de datos.

La importante función del administrador

Al mismo tiempo que instalamos un servidor de datos, hemos de pensar en designar a una persona encargada de controlar todos los aspectos del servidor para su buen funcionamiento, o lo que es lo mismo, el administrador del servidor.

El administrador debe supervisar la actividad el servidor, para asegurarse de que no existan problemas que puedan bloquearlo y paralizar la actividad de la red. También se encarga del diseño y creación de las bases de datos, asignación de permisos a los usuarios de la red para el acceso a los diferentes elementos de las bases de datos, conversión de datos procedentes de fuentes externas, realización de copias de seguridad y restauración de las mismas, etc.

Técnicas de almacenamiento de datos

Las soluciones desarrolladas para la manipulación de datos han sido variadas, estando determinadas por factores como la tecnología disponible para crear herramientas de almacenamiento y la cantidad de información a guardar. A continuación se enumeran algunas de ellas.

Ficheros de texto plano

La aplicación que maneja estos ficheros es la encargada de su gestión, por lo que el programador debe desarrollar todo el código que realice las labores de grabación, organización y consulta de los datos. Cuanto mayor es la cantidad de datos, más difícil se vuelve su control.

Ficheros con formato proporcionado por el lenguaje

El lenguaje de programación proporciona un conjunto de instrucciones, que permiten disponer de un cierto nivel de organización en forma de registros y campos para los datos. Un ejemplo de este tipo lo encontraríamos en los ficheros de acceso aleatorio creados desde Visual Basic, en donde es el lenguaje el que organiza mediante una serie de comandos, la información en el fichero, sin que el programador tenga que preocuparse de crear los algoritmos de grabación y recuperación de los datos.

A pesar de ello, cuando comienza a aumentar su contenido, la dificultad de manejo es parecida a los ficheros de texto plano.

Ficheros con formato de registro

Son una variante o evolución del tipo anterior. En este caso, el fichero se organiza en forma de tabla, compuesta por registros; a su vez, cada registro está formado por una serie de campos.

El lenguaje de programación también proporciona instrucciones para la creación de este tipo de ficheros, pero a diferencia del caso anterior, en el que el formato del fichero es propiedad del lenguaje, aquí el lenguaje se adapta a una serie de normas para la creación de este tipo de ficheros, dado que ha alcanzado un gran nivel de adaptación, convirtiéndose en un estándar.

El máximo exponente de este tipo de ficheros es el formato DBF, que a pesar de haber sido superado por otras soluciones de almacenamiento más evolucionadas, aún conserva un gran número de aplicaciones que lo utilizan.

Permite un gran número de registros por fichero y como complemento, dispone de ficheros de índice, que guardan una referencia ordenada en base a uno o varios campos del fichero de datos, lo que posibilita un rápido acceso a los registros en un orden determinado.

Dispone de un buen control sobre una gran cantidad de datos, pero al tratarse de ficheros-tabla aislados, su manejo a la hora de relacionarlos debe ser resuelto por la aplicación o aplicaciones que los manipulen, o por productos de terceros fabricantes.

Las técnicas descritas hasta el momento han caído en un progresivo desuso, debido principalmente al auge de aplicaciones específicas de gestión de bases de datos, que incorporan toda la mecánica de almacenamiento y proceso de la información, facilitando la labor del programador. Los siguientes puntos, describen los dos modelos principales utilizados.

Bases de datos de tipo intermedio

Suponen una evolución importante y un punto de inflexión con respecto a las soluciones anteriores, ya que aquí es el propio medio de almacenamiento quien incorpora un conjunto de mecanismos para guardar y administrar los datos.

A este tipo de productos ya sí podemos referirnos como Sistemas Gestores de Bases de Datos Relacionales (SGBDR) o motores de datos. Una de sus ventajas es que incorporan mecanismos de integridad para los datos, con lo que liberan al programador de dicho trabajo, que hasta ese momento debía de realizarlo manualmente, codificando todas las reglas de integridad en la aplicación que manipulaba la información.

Aparte de la consabida organización en tablas de la información, otras cualidades de estos sistemas residen en la capacidad de manejar un elevado número de registros por tabla, establecimiento de relaciones entre las tablas, realizar consultas mediante sentencias SQL, etc. El exponente más destacado de este tipo lo constituye Access.

Bases de datos corporativas

Se trata de sistemas de gestión de datos, que partiendo de las características que tienen las bases de datos de tipo intermedio en cuanto a manipulación de información, están adaptadas para funcionar en entornos empresariales, proporcionando capacidades de procesamiento en red, control de usuarios, seguridad sobre los datos ante caídas del sistema mediante transacciones, etc.

SQL Server es uno de los ejemplos de este tipo de productos, que será analizado a lo largo de este texto en su faceta administrativa.

2

Características generales de SQL Server

SQL Server 7.0

SQL Server es un gestor de bases de datos relacionales compuesto por un conjunto de elementos, que se integran con el sistema operativo Windows NT y el resto de la familia de productos empresariales de Microsoft, BackOffice, para proporcionar un entorno avanzado de proceso de datos, dentro de una arquitectura cliente-servidor, (en próximos apartados trataremos el concepto cliente-servidor).

El rendimiento conseguido por SQL Server al ejecutarse en sistemas Windows NT, ediciones Server o Enterprise, es excelente, debido a la mencionada orientación cliente-servidor, y a los componentes específicamente desarrollados en estos sistemas operativos para la ejecución de SQL Server.

Algunas de las ventajas del trabajo conjunto entre SQL Server y Windows NT se enumeran a continuación:

- SQL Server aprovecha las características multiproceso de Windows NT, utilizando todos los procesadores instalados para optimizar el manejo de datos.

- El sistema de seguridad de SQL Server está integrado con el de Windows NT. De esta forma, el usuario sólo debe identificarse al comenzar su sesión de trabajo con NT, puesto que al conectar con SQL Server, se establece una relación de confianza en la que SQL Server asume que si el usuario ha iniciado su sesión en el sistema, sus claves de acceso son correctas también para el motor de datos, por lo que realiza la conexión.

- Para las labores de supervisión del funcionamiento, SQL Server aprovecha el Visor de sucesos del sistema operativo para insertar sus propios mensajes, unificando en un sólo lugar el

sistema de avisos. De igual modo, utiliza el Monitor del sistema de Windows NT para aspectos relacionados con el rendimiento de las bases de datos.

- Sobre la disponibilidad inmediata de los datos en casos de fallo del servidor, SQL Server aprovecha las capacidades de clustering de que dispone Windows NT Enterprise, de forma que si en un sistema se han instalado dos servidores en clúster, SQL Server realizará el cambio al servidor de respaldo en el caso de que se produzca una caída del principal.

En cuanto a la integración con las aplicaciones de BackOffice, SQL Server trabaja en equipo junto a este conjunto de herramientas para crear un sólido soporte a nivel empresarial. Algunas de estas herramientas son las siguientes:

- El propio sistema operativo Windows NT, del que acabamos de ver cómo permite que SQL Server comunique con los distintos elementos de la red para el intercambio de datos.

- El servidor de correo Exchange Server, que facilita a SQL Server el envío de mensajes cuando se producen errores en el motor de datos o al finalizar una tarea programada.

- El administrador de recursos Systems Management Server, que controla el software y hardware del sistema y utiliza SQL Server para almacenar su información.

SQL Server puede ejecutarse en un amplio abanico de sistemas operativos. Dependiendo del sistema, podrá actuar como cliente o servidor. Los sistemas a los que se proporciona capacidad de servidor son: Windows NT en cualquiera de sus ediciones (Server, Enterprise y Workstation) y Windows 9x. En cuanto a los sistemas para los que SQL Server dispone de elementos de cliente están los antes mencionados en el aspecto de servidor más Windows 3.x, MS-DOS, Macintosh y UNIX. Finalmente, fuera del ámbito de sistemas, también puede ejecutarse como cliente dentro de los navegadores de Internet.

En cuanto a la escalabilidad del motor de datos, puede manejar desde pequeñas bases de datos en modo local a grandes bases de datos con conexiones de miles de usuarios y más de un terabyte de capacidad de almacenamiento.

Bases de datos en SQL Server

El componente encargado de guardar la información y sobre el que giran el resto de componentes de SQL Server es la base de datos.

Una base de datos está formada por una serie de elementos, también denominados objetos de la base de datos, que permiten organizar la información, relacionarla con otros objetos de la base de datos, mantener su integridad, etc. Entre los objetos más importantes podemos destacar los siguientes:

- Tabla.

- Índice.

- Vista.

- Procedimiento almacenado.

- Función o rol.

- Desencadenador.

El lenguaje de consulta Transact-SQL

SQL Server incorpora Transact-SQL como medio de consulta de los datos, una versión del lenguaje SQL que cumple con la especificación ANSI SQL-92. De esta forma, tenemos la seguridad de que cualquier instrucción o expresión de consulta que cumpla con dicho estándar, podrá ser utilizada en la manipulación de los datos.

Adicionalmente, Transact-SQL incluye un conjunto de elementos propios, que extienden la funcionalidad del lenguaje, proporcionándole mayor flexibilidad a la hora de la manipulación de los datos.

Ejecución a través de servicios

Las partes integrantes de SQL Server se ejecutan como servicios del sistema operativo, pudiendo ser iniciados durante el proceso de arranque del sistema o manualmente por el administrador.

En el caso de que no queramos ejecutar estos componentes como servicios, también es posible su ejecución como aplicaciones normales.

Veamos a continuación, una breve descripción de cada uno de estos servicios:

- MSSQLSERVER. Es el servicio principal, corresponde al propio motor de datos. Realiza todas las labores de manipulación, mantenimiento e integridad de la información; control de los ficheros que componen las diferentes bases de datos, bloqueo de registros, etc.

- SQLServerAgent. Se encarga de las labores relacionadas con la programación de tareas y avisos.

- MSDTC. Es el coordinador de transacciones distribuidas, gestiona las transacciones en las que se ven envueltos diferentes orígenes de datos, vigilando que todas las modificaciones sobre la información resulten coherentes, o en caso contrario, deshacer la transacción.

- Microsoft Search. Se trata de un motor de datos que trabaja con texto y permite la creación de consultas e índices hacia esta información textual.

Herramientas administrativas

SQL Server dispone de un amplio conjunto de herramientas que puede utilizar el administrador del sistema como apoyo en sus tareas habituales de administración. A continuación se realiza una breve descripción de las más importantes.

Administrador corporativo

Se trata de la principal aplicación incluida en el producto. Nos permite de modo gráfico, controlar todos los aspectos de SQL Server, tales como la creación de bases de datos, tablas, índices, procedimientos almacenados, inicios de sesión para usuarios, visualización de registros, realizar copias de seguridad, etc. Figura 1.

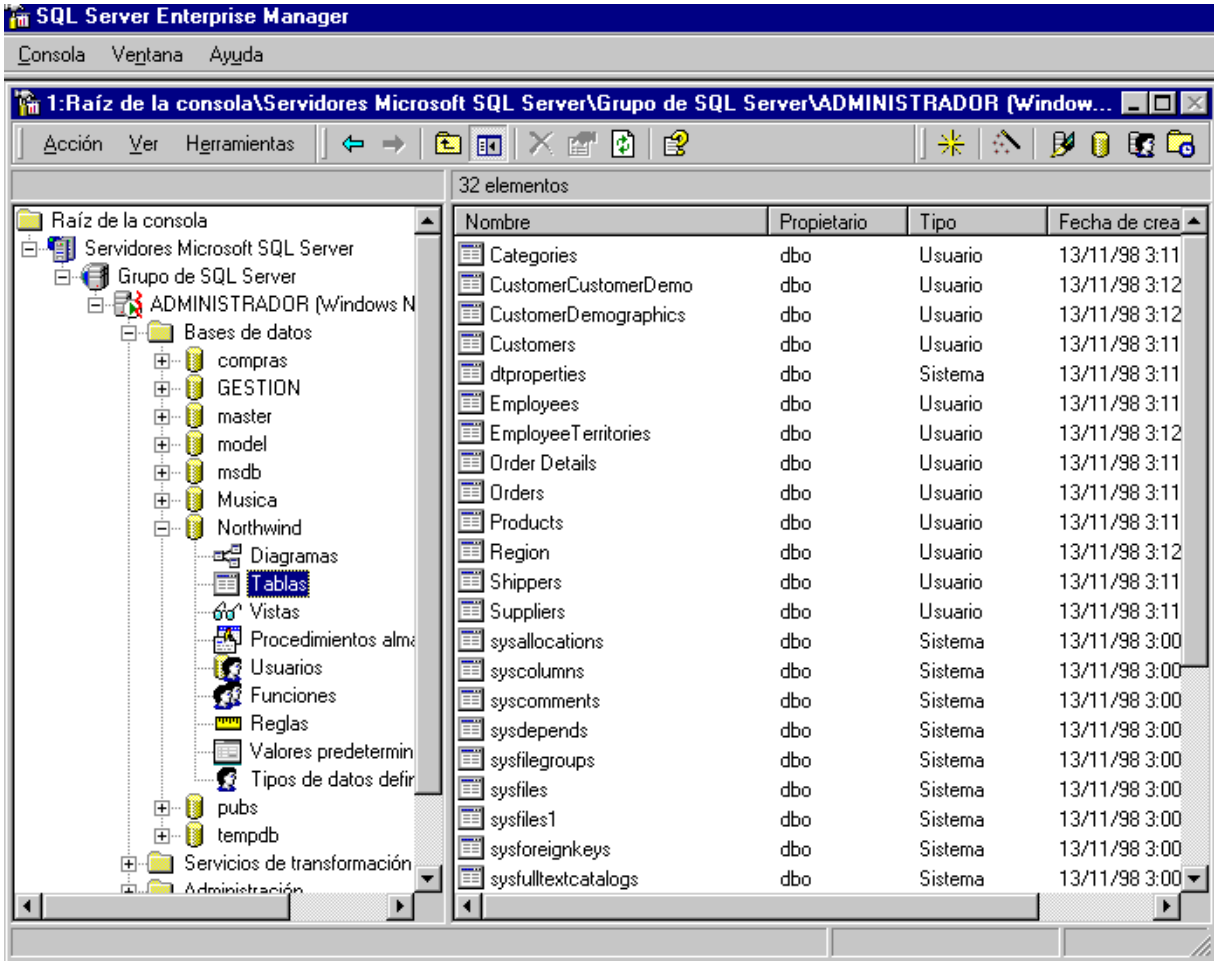


Figura 1. Administrador corporativo de SQL Server.

Administrador de servicios

Desde esta aplicación, Figura 2, podemos iniciar, finalizar o detener momentáneamente los diferentes servicios de SQL Server que se ejecutan en el sistema operativo, así como comprobar su estado de ejecución.

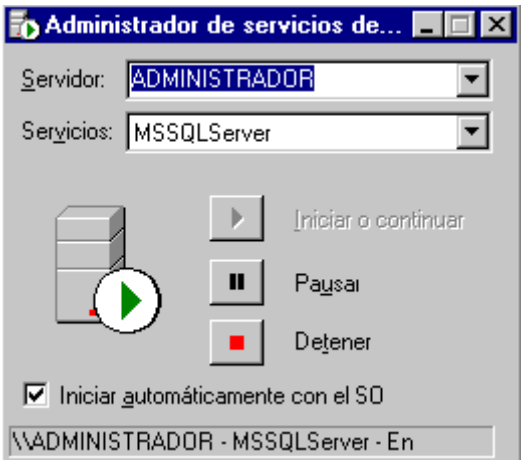


Figura 2. Administrador de servicios de SQL Server.

Para mayor comodidad, podemos configurar esta herramienta para que se inicie automáticamente cada vez que arranque el sistema operativo, ahorrándonos esa rutinaria tarea.

Analizador

Esta herramienta, ver Figura 3, permite controlar la actividad del servidor de datos para comprobar su rendimiento. Disponemos para ello de un elemento llamado traza, que consiste en un análisis de ejecución sobre uno o varios sucesos que tengan lugar en la base de datos. La traza devuelve una serie de estadísticas para averiguar los puntos débiles y optimizar el rendimiento.

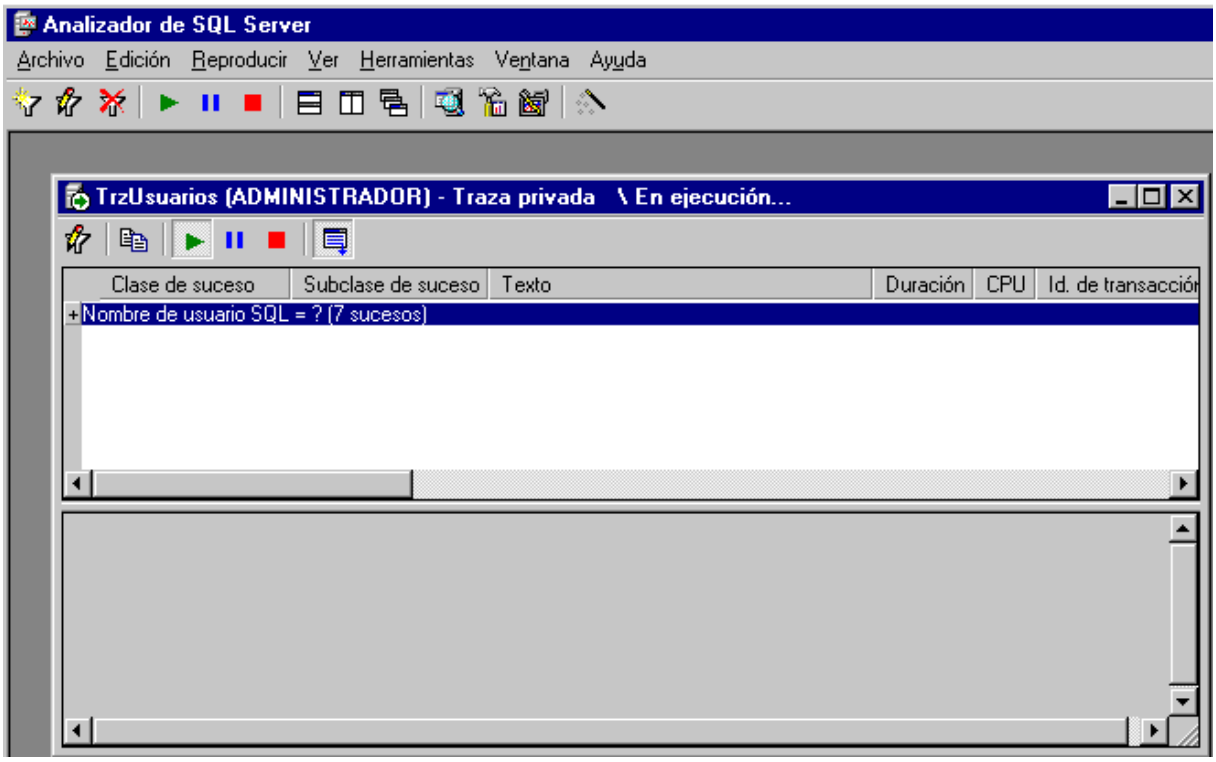


Figura 3. Analizador de SQL Server.

Desde esta herramienta, podemos acceder a otras de SQL Server como el Administrador corporativo, el Analizador de consultas, etc.

Analizador de consultas

Aplicación de gran utilidad que se utiliza para enviar instrucciones a las bases de datos empleando Transact-SQL. Podemos ejecutar más de una consulta al mismo tiempo y obtener las filas resultantes o el plan de ejecución, ver Figura 4, llevado a cabo por el motor de datos, que nos informa del consumo de recursos realizado por el servidor para completar dicha tarea.

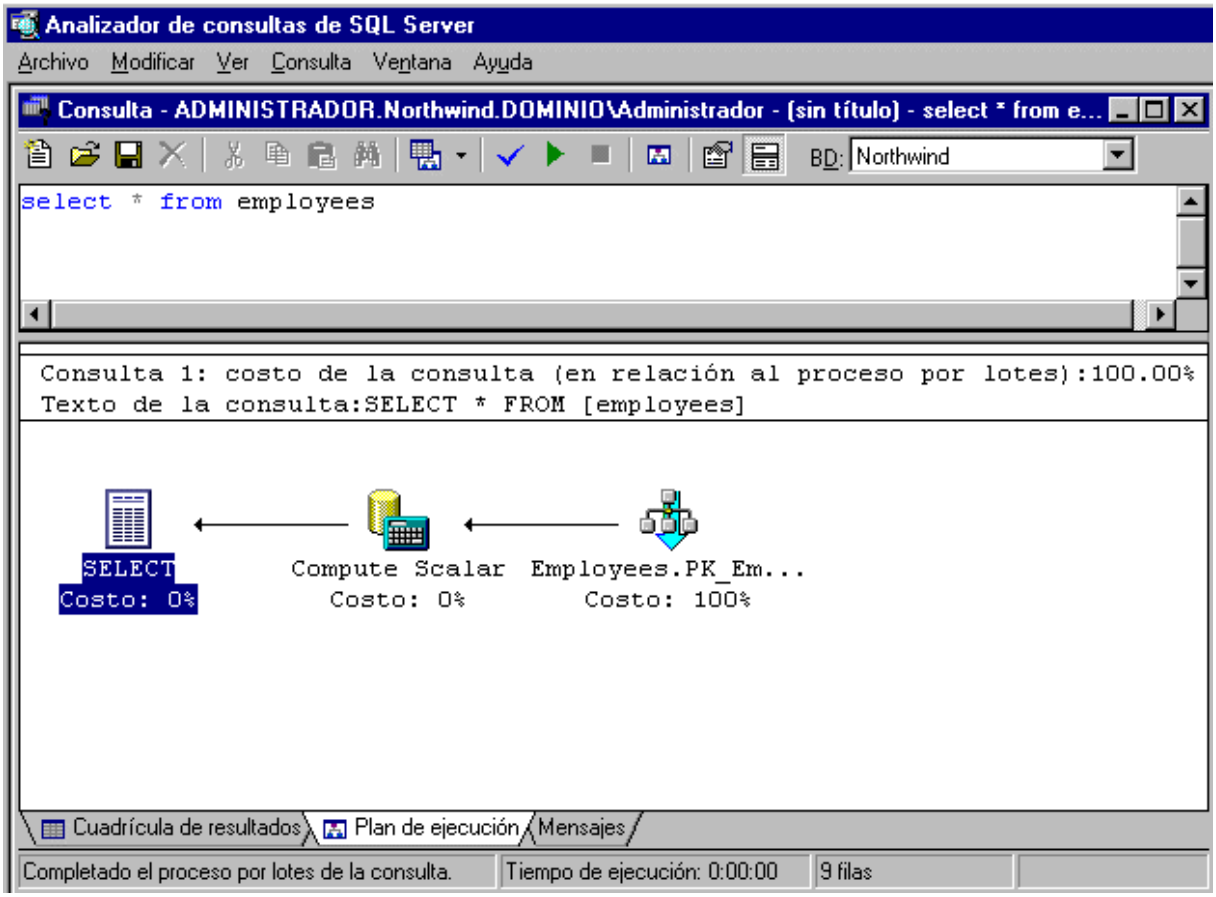


Figura 4. Analizador de consultas de SQL Server.

osql

Aplicación que se ejecuta desde el Interfaz de comandos de Windows NT (sesión MS-DOS), y permite la ejecución de consultas e instrucciones en modo carácter, desde el símbolo del sistema.

bcp

Al igual que en la anterior, esta utilidad también se ejecuta en modo MS-DOS. Su finalidad es la importación y exportación de datos entre SQL Server y ficheros de datos.

Además de estas aplicaciones, SQL Server incorpora asistentes para realizar las labores administrativas no contempladas en las anteriores herramientas, convirtiéndose en esta versión, en uno de los motores de datos más fáciles de administrar.

Características de seguridad

El control de usuarios a la información de una base de datos es un elemento de vital importancia. Cuanto más potente sea el gestor utilizado, mayores características de seguridad es necesario que incorpore. A este factor se une, de forma paradójica, el hecho de que progresivamente, parte de los datos en las empresas deben estar accesibles de manera cada vez global, debido al auge de Internet y la comunicación electrónica entre empresas.

Es por este motivo, que un buen sistema de seguridad debe buscar un difícil equilibrio: estar dotado de los mecanismos que le hagan robusto contra intentos no autorizados de acceso, pero ser al mismo tiempo flexible, para conseguir que todos los objetos de la base de datos sean susceptibles de ser protegidos o accesibles en función del usuario que intente acceder.

SQL Server es un gestor de bases de datos que consigue este objetivo: su sistema de autenticación integrada con Windows NT evita duplicar el trabajo de definición de cuentas de acceso, estableciendo una relación de confianza con el sistema operativo que facilita la conexión de usuarios ya registrados en el sistema. Mantiene además, su propio sistema de accesos, para las situaciones en que la conexión no haya sido validada por el sistema operativo.

Dispone también de un conjunto de objetos de seguridad: inicios de sesión, funciones y permisos, que adecuadamente configurados, permitirán a un mismo usuario acceder a información situada a gran profundidad en el sistema de niveles de seguridad del motor de datos, mientras que podrán denegar el acceso a datos del mismo tipo para los que dicho usuario no haya sido autorizado.

SQL Server como base de datos local

Una de las novedades que aporta SQL Server 7.0, es la posibilidad de instalación en un equipo local, sin necesidad de estar conectado a un servidor. En dicho equipo se instalaría el motor de datos y trabajaría de forma independiente con los mismos, al estilo de motores de datos de nivel intermedio como Access. De esta forma, si ya somos usuarios de SQL Server, pero debemos desempeñar parte o todo nuestro trabajo sin conexión a una red, podemos seguir disfrutando de las características de este motor de datos.

Esta nueva particularidad de SQL Server puede ser aprovechada por empresas en las que algunos de los usuarios del sistema, no puedan estar permanentemente conectados al mismo por motivos de movilidad, por ejemplo, agentes comerciales que realizan desplazamientos fuera de las instalaciones de la compañía.

En este caso, dichos usuarios pueden instalar SQL Server en equipos portátiles, junto a las aplicaciones que manipulen esos datos localmente el tiempo en que estén fuera de la oficina central. Cuando regresen a la empresa, se conectarán como clientes con el motor de datos central de la compañía y actualizarán la información desde sus equipos.

Otra de las ventajas de la instalación local de SQL Server, reside en que este se puede autoconfigurar en el equipo cliente, ahorrando tareas de ajuste al administrador del sistema.

Desarrollo de aplicaciones para SQL Server

SQL Server es un motor de datos para el que cada vez se desarrollan un mayor número de programas. Por este motivo, tanto el propio servidor de datos como las herramientas de desarrollo utilizadas en la creación de programas para acceder a SQL Server, proporcionan las últimas tecnologías disponibles para facilitar su labor a los programadores.

Al ser este un curso orientado a la administración, no vamos a profundizar en el aspecto de la programación, sin embargo, siempre es conveniente conocer, aunque sea mínimamente, el engranaje interno de una aplicación que trabaja contra una base de datos SQL Server, en la que debemos tener en cuenta los siguientes aspectos

Manipulación de datos

El acceso a los datos desde una aplicación se realiza a través de una serie de niveles que sirven de puente entre la aplicación y la base de datos. Cada uno de estos niveles está compuesto por un modelo de objetos basados en COM.

En el nivel situado a mayor profundidad, más próximo al motor de datos, se encuentra el modelo OLE DB, cuyos objetos se encargan de trabajar directamente con los datos.

En el nivel superior, el que corresponde a la aplicación, se encuentra el modelo ActiveX Data Objects (ADO), que se encarga de ocultar la complejidad de OLE DB.

La ventaja del trabajo con OLE DB-ADO, reside en que podemos trabajar tanto con orígenes de datos relacionales como no relacionales. Este diseño proporciona una mayor variedad de acceso a la información, estando preparado para ser usado en Internet, y es el sistema de acceso a datos que Microsoft recomienda para todos los proyectos nuevos que se inicien.

A pesar de que también están disponibles, las interfaces de acceso a datos ODBC-RDO están cayendo progresivamente en desuso, no van a ser mejoradas en sucesivas versiones, por lo que sólo se recomienda su empleo para las aplicaciones ya desarrolladas a las que se debe realizar alguna modificación.

Como herramienta de programación, podemos emplear cualquiera de las que cumplan con la especificación COM, por ejemplo: Visual Basic, VBA para cualquier producto que lo soporte (como Office), Visual C++; mientras que el entorno de ejecución puede ser Windows, páginas ASP, etc.

La Figura 5 muestra el esquema de trabajo de estos objetos.

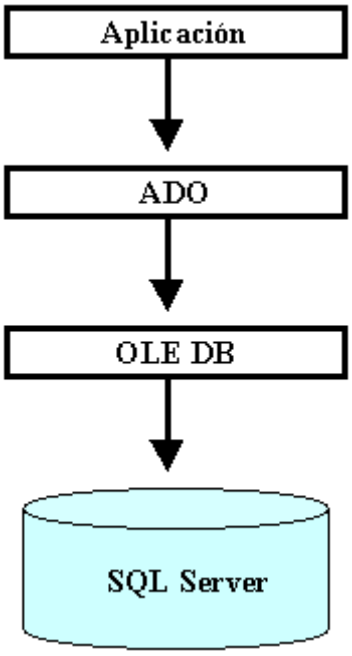


Figura 5. Mecanismo de trabajo de los modelos de objetos COM, para acceso a datos contra un origen de datos SQL Server.

Administración

Las tareas administrativas no son competencia exclusiva del Administrador corporativo de SQL Server y demás aplicaciones del motor de datos. Precisamente, todas estas herramientas de administración que se acompañan junto a SQL Server, están creadas utilizando un modelo de objetos denominado Objetos de Administración Distribuida de SQL (SQL-DMO), que como es lógico, están basados en COM y permiten a las herramientas de desarrollo (Visual Basic), desempeñar labores de administración desde las propias aplicaciones escritas con dichas herramientas.

DMO utiliza internamente instrucciones en Transact-SQL, pero gracias al conjunto de objetos de este modelo, el programador puede despreocuparse de tener que utilizar directamente Transact-SQL, lo que facilita enormemente el desarrollo de aplicaciones; ver Figura 6.

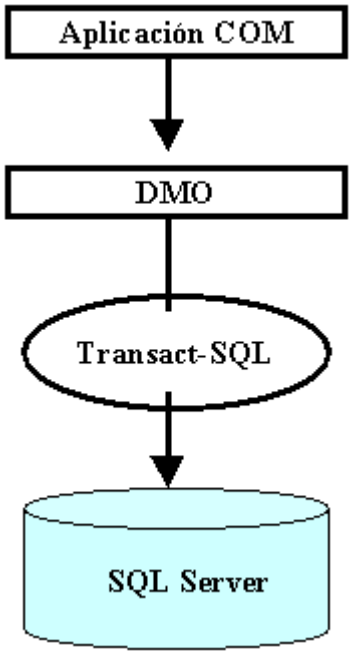


Figura 6. Administración de SQL Server desde una aplicación de usuario que utiliza el modelo DMO.

Transferencia de datos

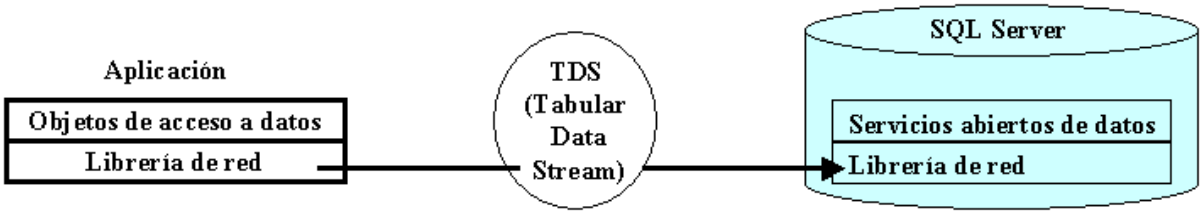
La estructura de comunicación de datos entre SQL Server y las aplicaciones que manipulan sus datos está diseñada mediante un conjunto de niveles, de tal manera que el programador no necesita preocuparse por los aspectos de intercambio de datos entre el servidor y su programa.

En la parte cliente donde reside la aplicación, el nivel con el que directamente trabaja el programador es el que contiene el modelo de objetos de acceso a datos. A partir de aquí, en el resto de niveles de comunicación de datos entre el cliente y el servidor de datos, existen un conjunto de librerías y protocolos que se encargan del envío y recepción de información entre la aplicación y la base de datos.

Tanto en el cliente como en el servidor, puede haber instalada una o varias librerías de red que comuniquen ambos puestos, siempre teniendo en cuenta, que deben de utilizar la misma librería para efectuar la comunicación.

Para poder manipular todo el proceso de información y solicitudes de los clientes realizadas con las diferentes librerías de red del servidor se emplea un componente de SQL Server denominado Servicios abiertos de datos.

Finalmente tenemos TDS (Tabular Data Stream), o secuencia de datos tabulares, que consiste en el protocolo a través del cual viajan los datos entre los clientes y el servidor de datos. Dependiendo de la librería de red empleada para la comunicación de datos, la información de TDS se introduce en paquetes adaptados a dicha librería.



3

Arquitectura Cliente-Servidor

Introducción a la arquitectura cliente-servidor

Al realizar la implantación de un sistema de gestión de datos, corremos el riesgo de que si aumenta el volumen de información a procesar o la cantidad de usuarios (puestos de trabajo), el rendimiento del sistema se vea afectado hasta el punto de llegar a colapsarse.

Como solución a este problema se desarrolló la arquitectura cliente-servidor, que permite una adecuada distribución de los componentes, de modo que el sistema pueda crecer, reduciendo los inconvenientes de la falta de rendimiento.

Antes de entrar a ver cómo es tratado este tema por SQL Server, repasemos unos conceptos previos sobre las partes que componen la lógica de una aplicación, que nos ayudarán a comprender la distribución del trabajo en un entorno cliente-servidor.

Lógica de una aplicación

Las aplicaciones en general y en nuestro caso particular, las aplicaciones que interactúan con bases de datos, están compuestas por una serie de lógicas de funcionamiento, que podemos ver en la Figura 7.

En primer lugar, encontramos la lógica de presentación, que se ocupa de tratar con el usuario, mostrándole o solicitando información.

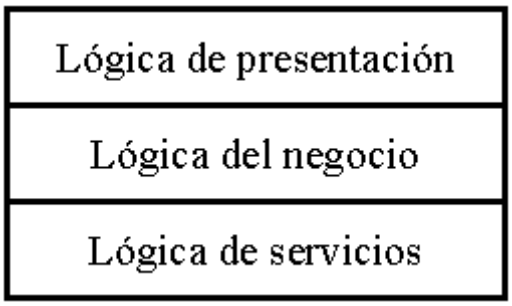


Figura 7. Lógicas de una aplicación.

A continuación se sitúa la lógica del negocio, cuya misión consiste en ejecutar las peticiones del usuario, como puedan ser consultas, agregados, modificaciones, etc., contra la base de datos, y validaciones de los datos introducidos por dicho usuario.

Finalmente tenemos la lógica de servicios, en la que reside la base de datos, encargada de devolver las peticiones o manipular los datos que le indica la lógica del negocio.

Una aplicación en la cual no se dividan las diferentes lógicas que la componen se denomina aplicación de una capa. Este tipo de diseño, en lo que respecta al enfoque por capas, es el más sencillo de desarrollar y funciona perfectamente en entornos que procesan poca cantidad de datos y tienen pocos usuarios.

Las aplicaciones de una capa no quiere decir que deban ejecutarse en un único equipo, podemos tener una red en la que el fichero o ficheros de la base de datos se instalen en cualquiera de los equipos o en un servidor de ficheros y compartirla con el resto. La característica más importante de este tipo de diseño, radica en que lo que compartimos son sólo los datos, ya que el gestor de base de datos debe estar instalado en cada puesto de la red que vaya a utilizar los datos. La única separación es a nivel físico, residiendo los datos en una máquina aparte.

El problema sobreviene cuando el número de datos a procesar y/o el número de usuarios del sistema aumenta. En este escenario, el riesgo de pérdidas de rendimiento crece exponencialmente, ya que al no estar el motor de datos centralizado en un servidor, se deben transferir los ficheros de datos a cada puesto que los solicite y procesarlos allí, con lo que la carga de tráfico en la red aumenta negativamente. La Figura 8 muestra un esquema del funcionamiento de aplicaciones en una capa.

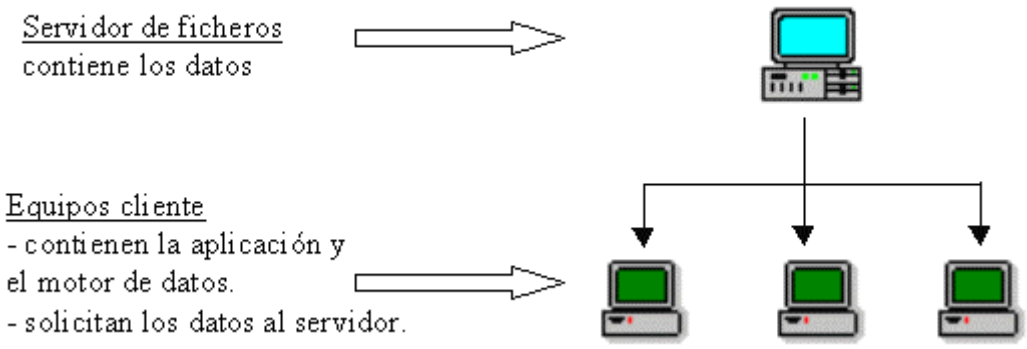


Figura 8. Estructura de funcionamiento en una aplicación de una capa.

Ante este tipo de situaciones, es donde se debe aplicar una solución cliente-servidor de varias capas.

Implementación de la arquitectura cliente-servidor en SQL Server

Un sistema que funcione bajo la filosofía cliente-servidor, ha de diseñarse de modo que el trabajo que lleven a cabo tanto su parte software como hardware esté repartido equilibradamente, para que el funcionamiento conjunto del sistema no provoque cuellos de botella que saturen ciertos componentes mientras que otros permanecen inactivos o con baja actividad.

El concepto cliente-servidor se aplica tanto a la parte lógica como a la física del sistema. En la parte física, el servidor está representado por un equipo de alta capacidad, que se encarga de proporcionar servicios (aplicaciones, datos) al resto de equipos del sistema o clientes. Esto no quiere decir, que sólo pueda haber un servidor para varios clientes, dependiendo de la cantidad de información a procesar, el número de clientes instalados y el diseño de las capas, es posible tener más de un servidor, cada uno proporcionando servicios específicos.

En cuanto a la lógica cliente-servidor, una aplicación servidora, es aquella encargada de procesar las peticiones o consultas que le demandan otras aplicaciones, denominadas cliente. En este caso, también puede existir más de una aplicación servidora, cada una especializada en procesar cierto tipo de información, ya que en función de las capas del sistema, podemos tener una aplicación servidora de bases de datos, otra servidora de aplicaciones, etc.

Las aplicaciones cliente no requieren ser ejecutadas sólo desde equipos cliente, es posible tener un equipo servidor con aplicaciones servidoras y también aplicaciones cliente que dentro de ese mismo equipo realicen peticiones a las aplicaciones servidoras, Figura 9; todo depende de los requerimientos de diseño del sistema.

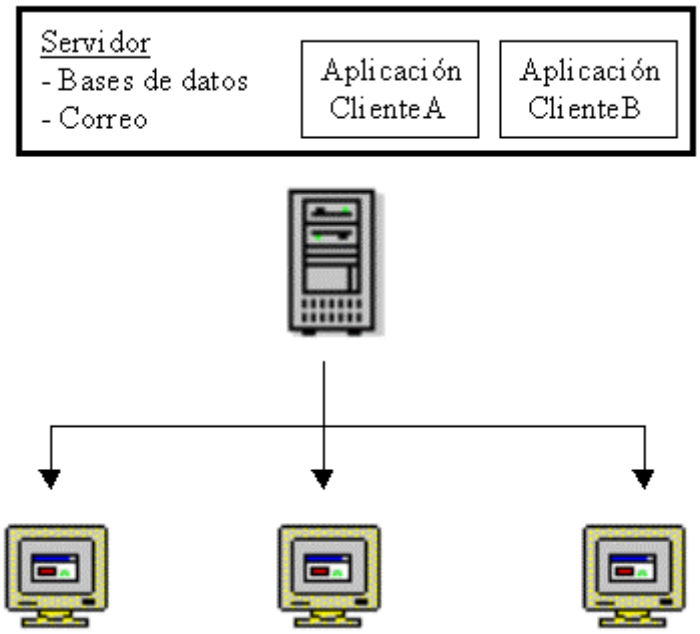


Figura 9. Servidor ejecutando sus propias aplicaciones cliente.

La Figura 10 muestra el modo de ejecución de aplicaciones servidoras distribuidas entre varios servidores físicos.

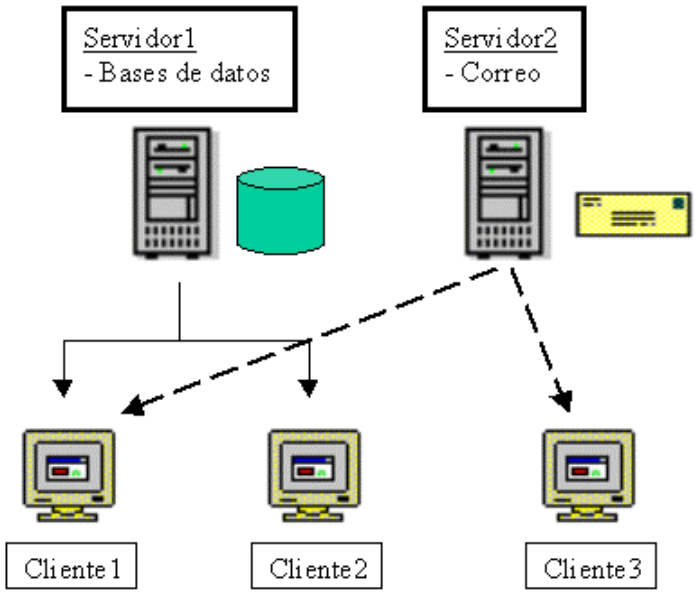


Figura 10. Aplicaciones servidoras repartidas entre varios servidores físicos.

SQL Server está preparado para funcionar en el entorno cliente-servidor tradicional de dos capas, y también en los cada vez más utilizados de n-capas. El modo de operación de uno y otro lo veremos seguidamente.

Sistemas cliente-servidor de dos capas

En un sistema cliente-servidor de dos capas, la lógica de presentación y la de negocio se encuentran situadas en la aplicación cliente, mientras que la lógica de servicios, en nuestro caso el servidor de bases de datos SQL Server sería la aplicación servidora, como podemos ver en la Figura 11.

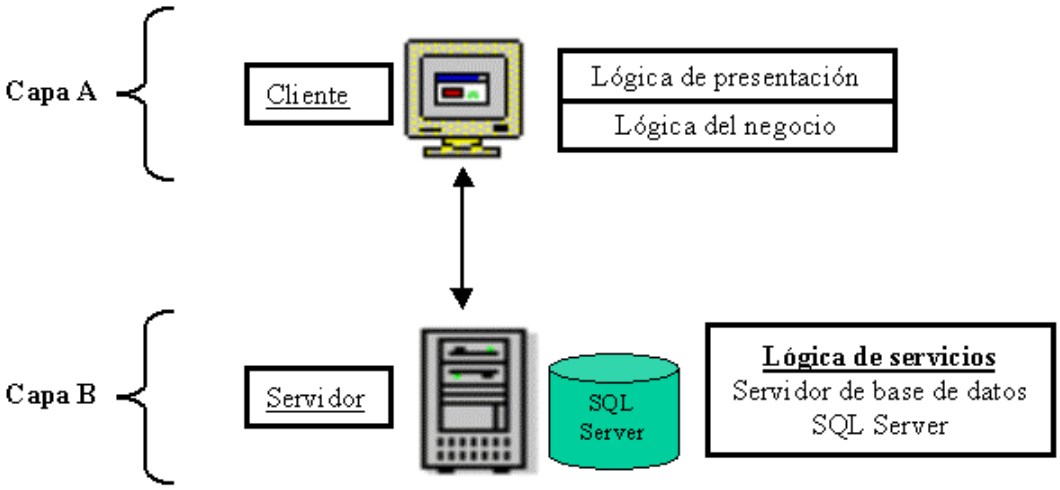


Figura 11. Distribución de la lógica de funcionamiento en un entorno cliente-servidor de dos capas.

A pesar de que en esta estructura se han separado físicamente los elementos, la aplicación cliente, como ya hemos comentado, podría residir perfectamente en el equipo servidor.

Al centralizar la gestión de los datos, obtenemos una serie de ventajas, como puedan ser una mayor velocidad en el tráfico de red, al transferirse sólo los datos que los clientes solicitan; no es necesario duplicar los datos, existe una sola copia de los mismos en el servidor y con ellos trabajan todos los clientes; las reglas de integridad, relaciones, índices, procedimientos almacenados, así como el control de usuarios a la base de datos, se establecen una única vez en el servidor, lo que ahorra trabajo de administración. Finalmente podemos apuntar que se produce una importante reducción en el gasto de equipos cliente, ya que estos necesitan menos recursos al trabajar con una copia de los datos que necesitan.

Sistemas cliente-servidor de n-capas o multicapa

En esta técnica de implementación de la arquitectura cliente-servidor, el trabajo se encuentra todavía más distribuido que en el anterior modelo, puesto que aquí la lógica de presentación y de negocio se encuentran separadas en dos aplicaciones distintas, lo que permite a la lógica de negocio una mayor independencia al no tener que preocuparse de cómo serán mostrados los datos al usuario.

La lógica de negocio se encuentra en este caso en una o varias aplicaciones, denominadas aplicaciones de servidor; su emplazamiento suele ser un servidor específico denominado servidor de aplicaciones. En el caso del trabajo con datos, la aplicación de servidor sería la encargada de conectar con la lógica de servicios (base de datos) y solicitar los datos pedidos por el usuario en la lógica de presentación, ver el esquema de la Figura 12.

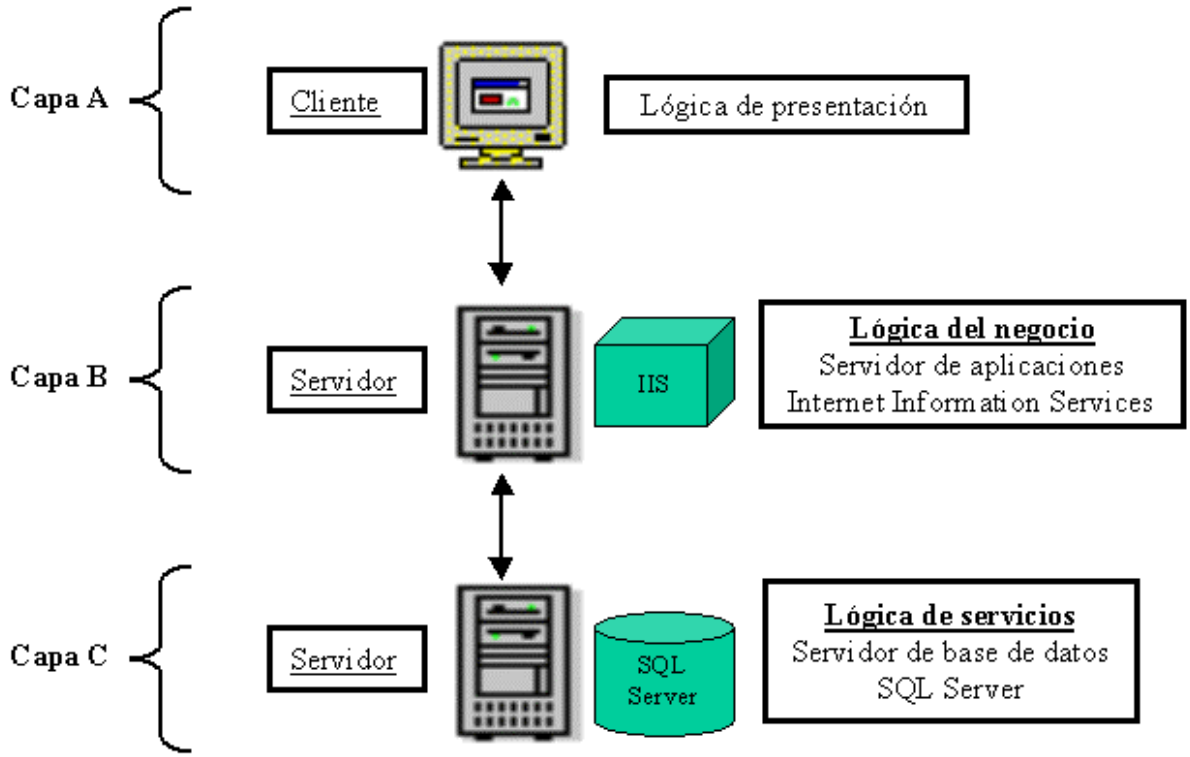


Figura 12. Distribución de la lógica de funcionamiento en un entorno cliente-servidor de n-capas

Un ejemplo de este modo de funcionamiento lo constituye la arquitectura de ejecución de Internet, en la que la lógica de presentación estaría representada por los programas navegadores web. Estos programas enviarían sus peticiones a la lógica de negocio, representada por aplicaciones que residieran en un equipo servidor con Microsoft Internet Information Services (IIS), que es el servidor de

aplicaciones de Microsoft para Internet, y finalmente, la aplicación situada en IIS encargada de realizar esta petición, conectaría con la lógica de servicios (SQL Server) para recuperar los datos solicitados. En este escenario, tanto IIS con SQL Server, aunque pertenezcan a lógicas de la ejecución distintas, pueden residir en el mismo o en distintos servidores físicos, no hay una regla fija a este respecto.

Aunque esta técnica nos indica que la lógica de presentación debe realizar el menor trabajo posible, puede darse el caso de que sea necesario incluir parte de lógica de negocio, como las validaciones de los valores introducidos por el usuario, en el navegador. De esta forma se evita realizar dichas validaciones en la aplicación de servidor, agilizando la ejecución global al eludir el paso de comunicación entre la lógica de presentación y la de negocio en este caso puntual.

Cuando se produce una situación de este tipo, a la aplicación que se ocupa de la lógica de presentación, pero que incluye parte de la lógica de negocio se le denomina cliente grueso (thick client). En el caso de que la aplicación de usuario sólo incluya la lógica de presentación se le denomina cliente delgado (thin client).

Preparación de la instalación de SQL Server 7.0

Consideraciones Hardware

Antes de proceder a instalar SQL Server en nuestro equipo, debemos comprobar que cumple con una serie de requisitos a nivel de hardware para asegurarnos que la instalación podrá llevarse a cabo.

SQL Server puede ser instalado en una máquina que disponga de un procesador Intel o compatible a partir de un Pentium 166MHz. También es posible utilizar SQL Server en plataformas con procesador Alpha.

En cuanto a memoria, podemos partir de 32 MB, aunque es una cantidad insuficiente en la mayoría de los casos, ya que en cuanto el tamaño de las bases de datos crece y se realiza un gran número de tareas transaccionales y de duplicación el sistema se resiente. Una cantidad algo más realista serían 64 MB y si de verdad queremos que el sistema vaya *a toda máquina*, deberemos emplear 128 MB.

Para el disco duro ocurre algo similar (nunca será demasiado grande), la instalación típica del gestor ocupa aproximadamente 175 MB, pero es necesario realizar un estudio sobre las necesidades en cuanto a ocupación de las bases de datos que vayamos a crear, índices, copias de seguridad, tareas de duplicación, etc., que serán las que realmente devoren nuestro disco. En este caso es difícil recomendar un tamaño de disco, lo mejor es que empleemos el mayor que podamos, con el tiempo a buen seguro que se quedará pequeño.

Consideraciones Software

De igual forma que para los requerimientos de máquina, es preciso analizar las características del sistema en el cual vamos a instalar SQL Server 7.

Con esta versión se hace incluso más necesario, ya que la versión 6.5 sólo permitía su instalación en sistemas servidores, mientras que la actual puede ser utilizada desde sistemas Windows 9x hasta servidores NT Enterprise.

Como sistema de ficheros debemos emplear siempre que sea posible NTFS, por sus mejores características de rendimiento y seguridad frente al tradicional sistema FAT. También es necesario disponer del navegador Internet Explorer 4.01 o posterior.

Todas las pruebas y ejemplos mostrados a lo largo de este curso han sido realizados en un equipo con procesador Pentium III a 450 MHz, 128 MB de memoria, 10 GB de disco duro y sistema operativo Windows NT Server 4.0.

Creación de una cuenta de inicio de sesión

Como ya hemos indicado anteriormente, bajo Windows NT, el motor de datos y el agente de SQL Server se ejecutan como servicios del sistema operativo. Para ello es necesario que sean asignados a una cuenta de usuario del sistema.

Del tipo de cuenta asignada dependerá que se ejecuten con mayor o menor funcionalidad. Expliquemos este punto más detalladamente.

Entre las diversas operaciones que estos servicios llevan a cabo se pueden destacar las copias de seguridad entre diferentes unidades de red; tareas de duplicación; llamadas a procedimientos remotos; tareas de correo del Agente SQL Server y SQL Mail, etc.

Por otra parte, los tipos de cuenta de usuario que podemos asignar a estos servicios son la cuenta de sistema local, cuenta de usuario local y cuenta de usuario de dominio.

Las dos primeras no tienen privilegios de acceso a la red, de forma que si asignamos una de ellas a los servicios de SQL Server estaremos limitando gran parte de su operatividad.

Este motivo hace que lo más recomendable sea crear una cuenta de usuario de dominio para asignarla a estos servicios del gestor de datos.

Podemos crear una cuenta para cada servicio, pero es posible y también recomendable para evitar complicaciones innecesarias, utilizar la misma cuenta para todos los servicios.

A continuación vamos a mostrar los pasos necesarios para crear una cuenta de este tipo en el sistema operativo. Esta labor debemos realizarla como paso previo a la instalación del gestor de datos, ya que durante dicho proceso se nos pedirá que especifiquemos la cuenta para los servicios de SQL Server.

En primer lugar, seleccionaremos la ruta del menú de Windows NT *Programas+Herramientas administrativas+Administrador de usuarios para dominios* (Figura 13), que abrirá esta utilidad para gestionar las cuentas del sistema. Ver Figura 14.

Figura 13. Ruta de acceso al administrador de usuarios de Windows NT.

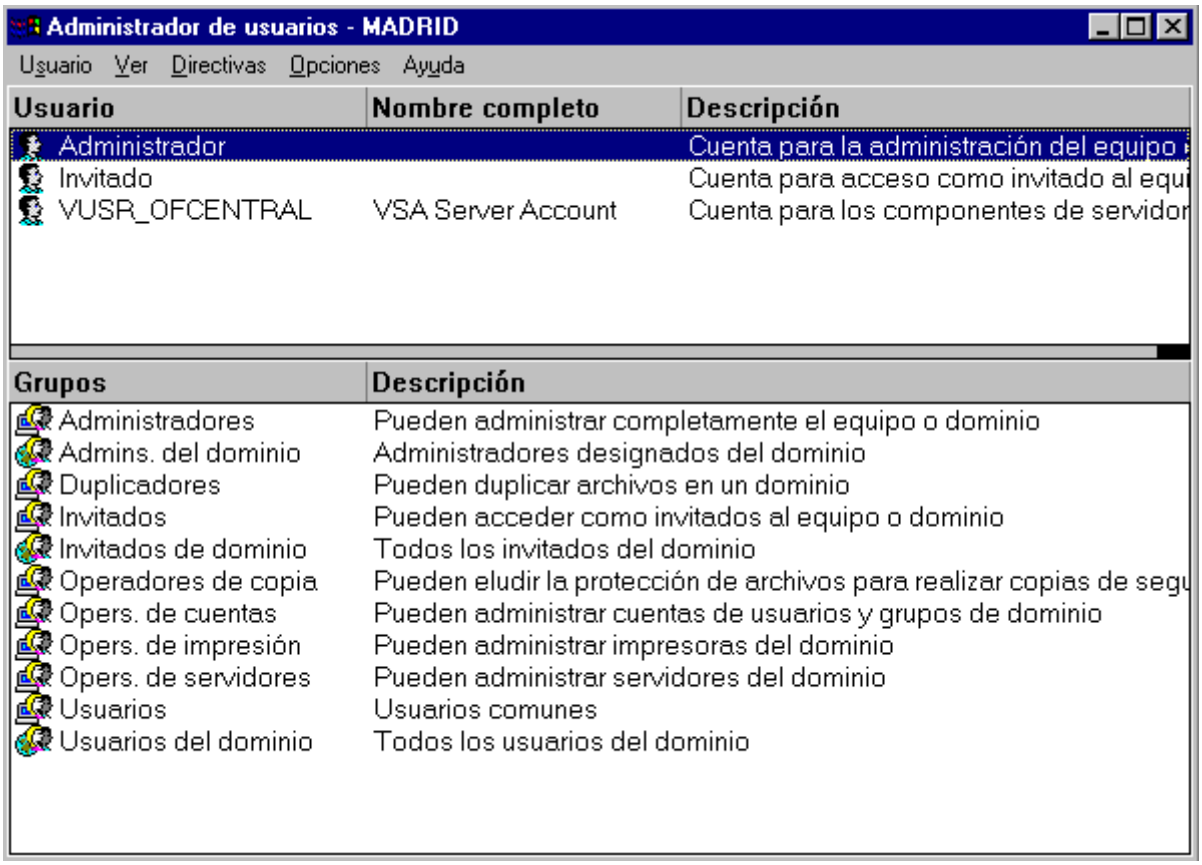


Figura 14. Administrador de usuarios de Windows NT.

- Seleccionaremos la opción *Usuario+Usuario nuevo,* para dar de alta la cuenta que utilizarán los servicios de SQL. Le asignaremos el nombre ServSQL, una contraseña y marcaremos la casilla *La contraseña nunca caduca*, para que la contraseña sea permanente. También podemos escribir una breve descripción del cometido de la cuenta, como se muestra en la Figura 15.

Figura 15. Creación de una nueva cuenta para Windows NT.

- Seguiremos con la asignación de la cuenta al grupo local Administradores, del equipo en donde vamos a instalar SQL Server. Para ello pulsaremos el botón Grupos, seleccionaremos el mencionado grupo y lo agregaremos como miembro de esta cuenta pulsando el botón Agregar.

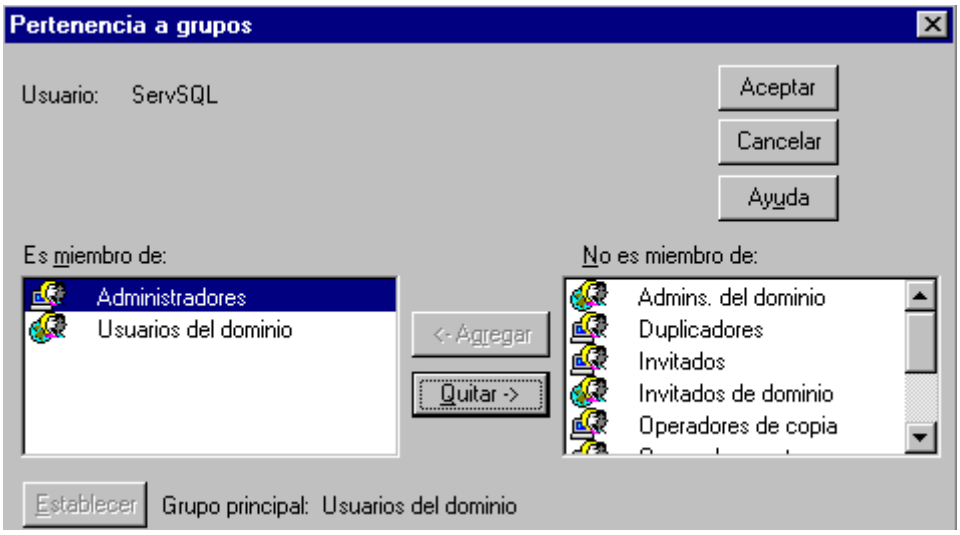


Figura 16. Asignación de cuentas a grupos del sistema.

- Hecho esto, aceptaremos todas las ventanas hasta volver a la de administración de usuarios.

- El siguiente paso consiste en asignar a la nueva cuenta el derecho de iniciarse como un servicio. Para ello seleccionaremos el menú *Directivas+Derechos de usuario*, que nos mostrará los derechos del sistema y los usuarios y grupos a los que se les han otorgado.

- Debemos marcar la casilla *Mostrar derechos de usuario avanzados,* para que la lista Derecho muestre todos los disponibles. Abrir la mencionada lista y seleccionar *Iniciar sesión como servicio*.

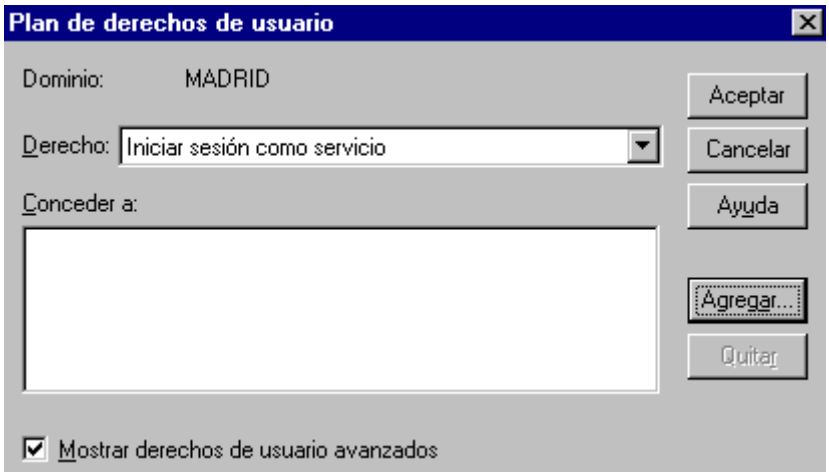


Figura 17. Seleccionar el derecho a asignar a la cuenta.

- Pulsando el botón Agregar, visualizaremos una ventana que nos permitirá seleccionar la cuenta de usuario o grupo a la que asignaremos el derecho seleccionado. Podemos pulsar el botón *Mostrar usuarios* para que en la lista de nombres aparezca la cuenta que hemos creado, pero si seleccionamos su grupo, Administradores, todas las cuentas que pertenezcan a este grupo adquirirán este derecho, y nuestra cuenta también.

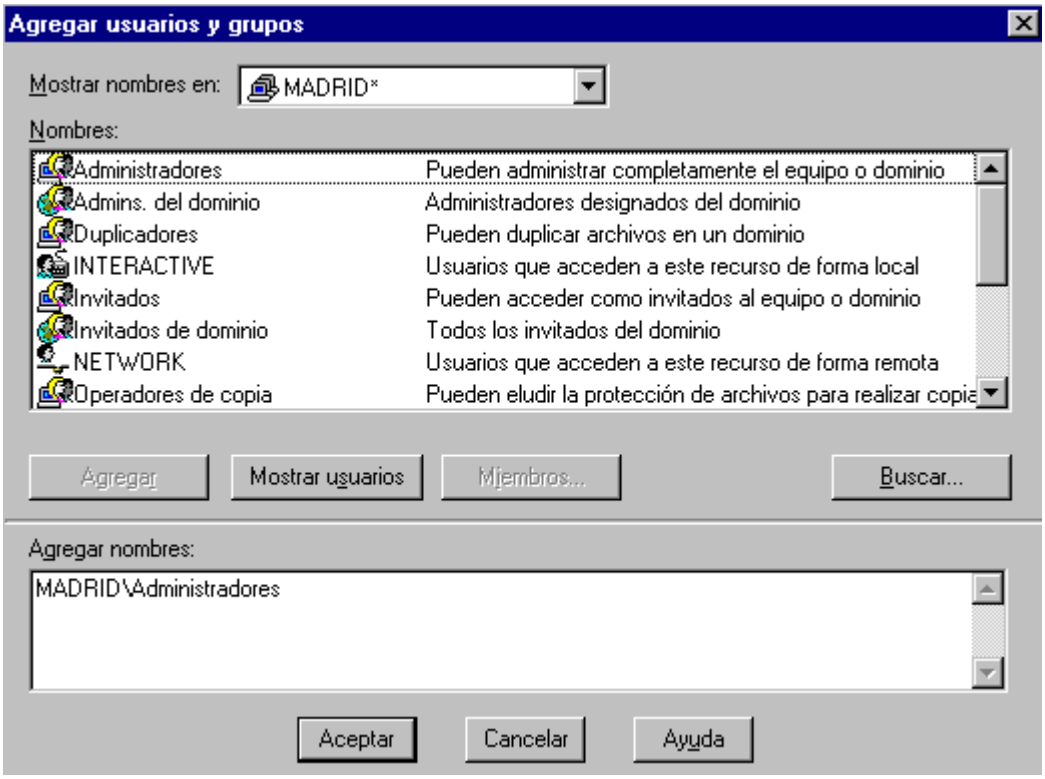


Figura 18. Agregar derechos a usuarios y grupos.

- Pulsando en esta ventana Agregar, se añadirá el grupo seleccionado. Después de agregar todos los grupos y usuarios, aceptaremos todas las ventanas incluida la correspondiente a la administración de usuarios, con lo que habremos concluido el proceso de creación de la cuenta de inicio para los servicios de SQL Server.

Versiones o ediciones de la instalación

SQL Server 7 dispone de las denominadas *versiones* o *ediciones* de instalación, que consisten en la posibilidad de realizar una instalación adecuada a las características del sistema operativo en el que deba de ejecutarse el servidor de datos. A continuación se muestra una breve descripción de cada una de estas ediciones.

Standard

Es la edición destinada a ejecutarse bajo Windows NT Server 4.0, dispone de características de multiproceso simétrico, aprovechando la instalación en el equipo de hasta cuatro procesadores; permite la ejecución de consultas en paralelo; búsquedas avanzadas mediante Microsoft Search, etc.

Debido a que el sistema operativo utilizado en este curso ha sido Windows NT Server 4.0, esta edición standard del motor de datos es sobre la que se han realizado todas las pruebas. Por este motivo, todas las observaciones que se realicen a nivel del sistema operativo a lo largo del curso se referirán a esta versión.

Enterprise

Dispone de todas las características de la anterior pero adaptada a entornos corporativos en los que se vaya a trabajar con un gran número de datos y soporten una elevada carga de transacciones. Debe instalarse en un sistema Windows NT Server Enterprise Edition 4.0. Permite el uso de hasta treinta y dos procesadores en multiproceso y soporta varios servidores funcionando en clustering, técnica que permite utilizar varios servidores físicos (clusters), simulando un único servidor lógico y permitiendo el balanceo de cargas, de forma que se optimiza al máximo el rendimiento de las aplicaciones en ejecución.

Desktop o SBS (Small Business Server)

Esta es la edición más limitada en funcionalidad, diseñada para ejecutarse bajo Windows 95-98 y para usuarios que habitualmente trabajan con aplicaciones que acceden a un servidor SQL Server Standard o Enterprise, pero que por diversos motivos deben manipular en ocasiones los datos desde el exterior, desde un portátil por ejemplo. De esta forma, pueden trabajar con sus aplicaciones habituales y realizar posteriormente la actualización de los datos en el servidor central de la empresa.

Observaciones a nivel del sistema operativo

Como paso previo a la instalación del gestor de datos, debemos tener en cuenta los siguientes puntos en lo que respecta al sistema operativo.

- No es conveniente instalar SQL Server en un servidor que sea controlador principal de dominio (PDC), ya que este tipo de servidor realiza un gran número de tareas administrativas a nivel del sistema que consumen una gran cantidad de recursos, mermando el rendimiento de SQL Server.

- Ya que habitualmente SQL Server se instala en un sistema formado por varios dominios de Windows NT, se recomienda la instalación en un dominio con acceso a todas las cuentas de todos los dominios.

- Para ejecutar el programa de instalación, debemos acceder al sistema como Administrador, de forma que no se produzcan errores por no contar con los permisos adecuados de acceso a los recursos.

5

Instalación de SQL Server 7.0

El programa de instalación

Una vez cumplidos los requerimientos previos a la instalación en cuanto a sistema operativo y máquina se refieren, procederemos a insertar en nuestro equipo el CD de instalación de SQL Server 7 y ejecutar el fichero AUTORUN.EXE, que nos situará en la primera pantalla del programa de instalación, Figura 19.

Figura 19. Pantalla inicial de instalación de SQL Server 7.0.

Antes de proceder a la instalación de los componentes de SQL Server 7 debemos asegurarnos que están instalados todos los prerrequisitos que este gestor de datos necesita, pues de lo contrario, no podremos continuar con el programa de instalación.

Instalación de los prerrequisitos

Al hacer clic sobre la opción Instalar los prerrequisitos de SQL Server 7.0 en la pantalla de instalación, podremos seleccionar el sistema operativo sobre el que vamos a instalar, del que se elegirán los prerrequisitos correspondientes, Figura 20.

Figura 20. Selección de los prerrequisitos de SQL Server 7.0.

Para Windows 95 (Figura 21), necesitamos instalar DCOM 95 o Internet Explorer 4.01 Service Pack 1 o superior.

Figura 21. Prerrequisitos necesarios para Windows 95.

Respecto a Windows NT 4.0, ver Figura 22, debemos instalar en primer lugar el Service Pack 4 de este sistema operativo y después Internet Explorer 4.01 Service Pack 1 o superior. En cuanto a este último, podemos realizar una instalación mínima o estándar, dependiendo de los requerimientos de conectividad necesarios para SQL Server.

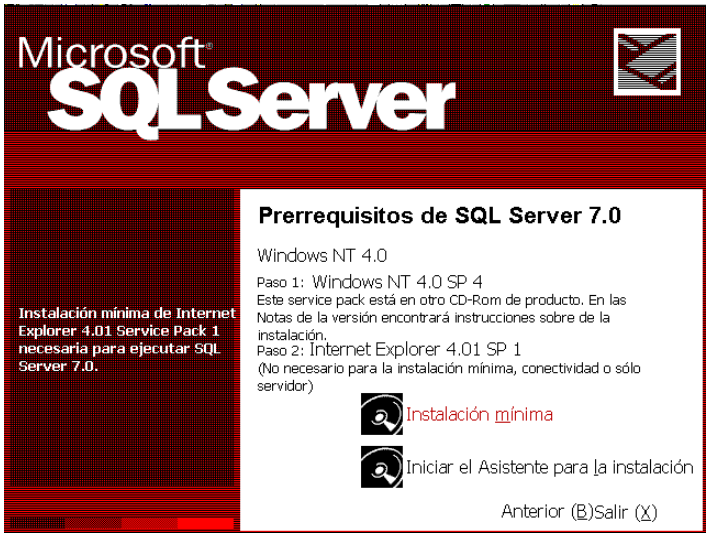


Figura 22. Prerrequisitos necesarios para Windows NT 4.0.

Seleccionar la edición a instalar

A continuación debemos elegir qué edición del gestor de datos queremos instalar, los servicios de análisis de información (OLAP), o la utilidad *English Query*, para generar consultas en lenguaje natural inglés. Seleccionaremos instalar el servidor de base de datos en su edición Standard, como muestra la Figura 23.

Figura 23. Selección de los componentes de SQL Server 7.0 para instalar.

El programa de instalación nos pregunta ahora si vamos a realizar este proceso en el equipo local o vamos a instalar en otro equipo de forma remota, ver Figura 24.

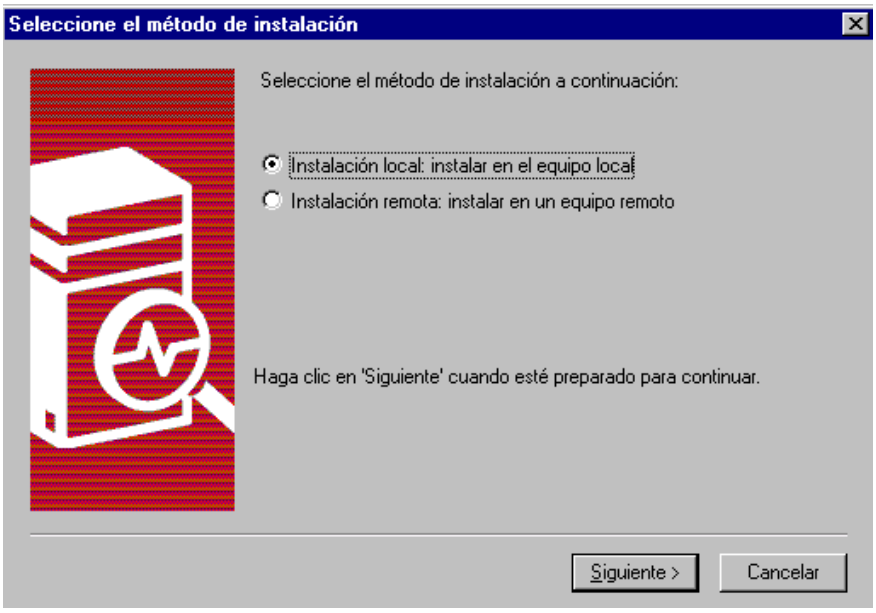


Figura 24. Método de instalación.

Después de aceptar los términos de la licencia del producto, rellenaremos una pantalla con nuestros datos e introduciremos la clave de instalación del CD.

Tipo de instalación

Seguiremos con el tipo de instalación a realizar: típica, mínima o personalizada, paso que muestra la Figura 25.

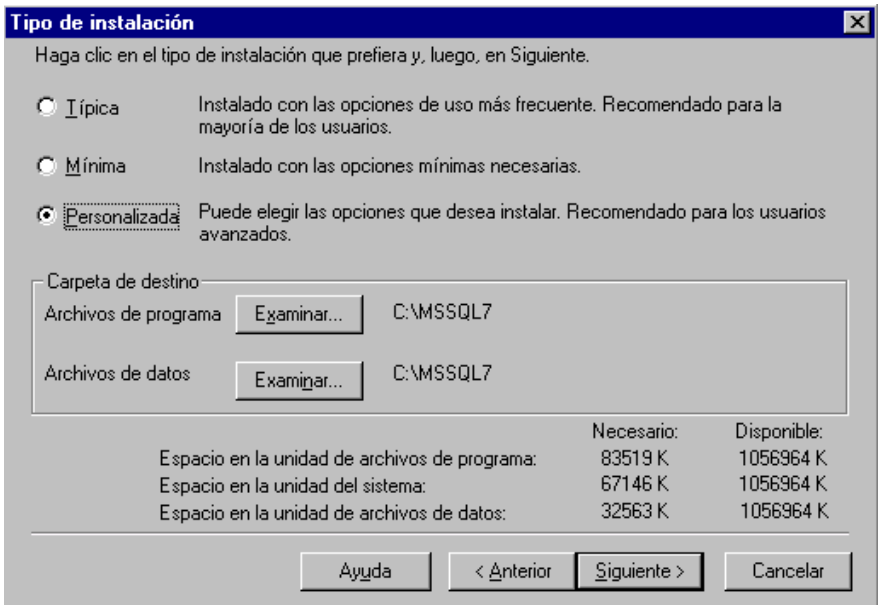


Figura 25. Tipo de instalación.

La instalación Típica es la más sencilla, ya que selecciona los valores más comunes de los componentes a instalar, y será la que elegiremos en la mayoría de las ocasiones. Debido a que los pasos en este tipo de instalación están muy automatizados y no entrañan dificultad, vamos a seleccionar la instalación Personalizada, que nos permitirá configurar manualmente los valores que la instalación típica establece por defecto, y de esta forma mostrar más detalladamente este proceso.

En este paso también podemos indicar el disco y ruta en el cual realizar la instalación, tanto para los archivos que componen el servidor como para los ficheros de bases de datos que vayamos creando. Completada esta pantalla, pulsaremos Siguiente para continuar con el próximo elemento a configurar.

Selección de componentes

A continuación se muestran los componentes de SQL Server que podemos instalar, ver Figura 26, de forma que podemos limitarnos a seleccionar únicamente el motor de datos si sólo queremos instalar los elementos para disponer de la funcionalidad más básica, o bien seleccionar otra serie de componentes como las herramientas administrativas, libros en pantalla, etc.

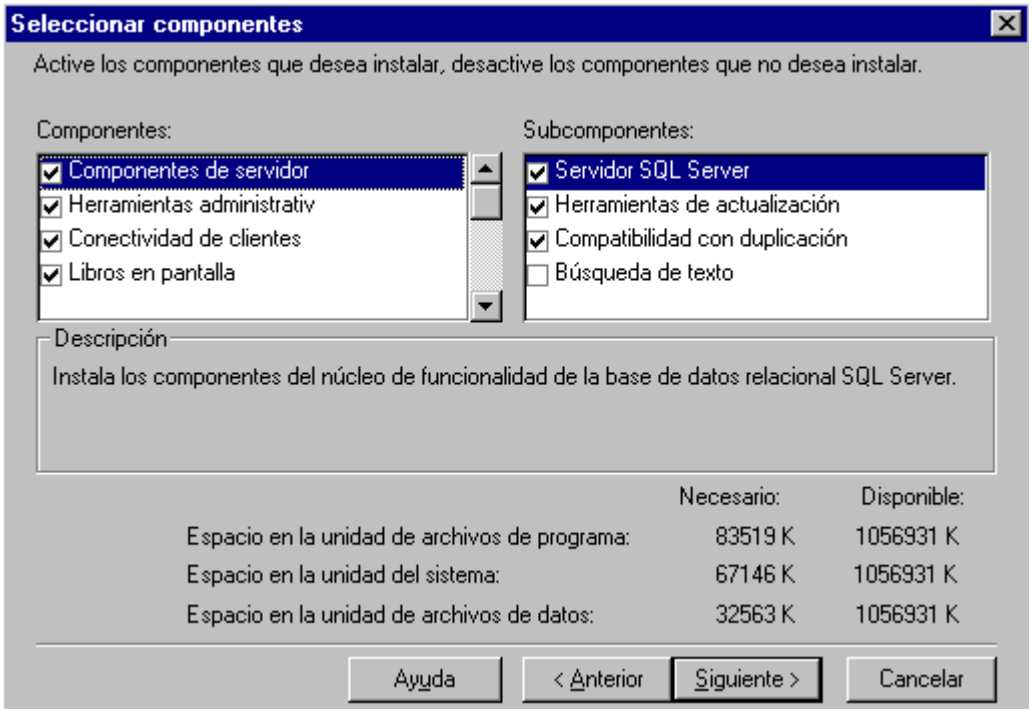


Figura 26. Selección de componentes a instalar.

Juego de caracteres y ordenación

En este punto debemos seleccionar el conjunto de caracteres reconocibles por el motor de datos, el modo en que se realizará su ordenación y la intercalación Unicode.

Debemos estar seguros de elegir los valores adecuados, ya que si posteriormente necesitamos cambiar alguno de los elementos de este paso, será necesario generar de nuevo las bases de datos, ver Figura 27.

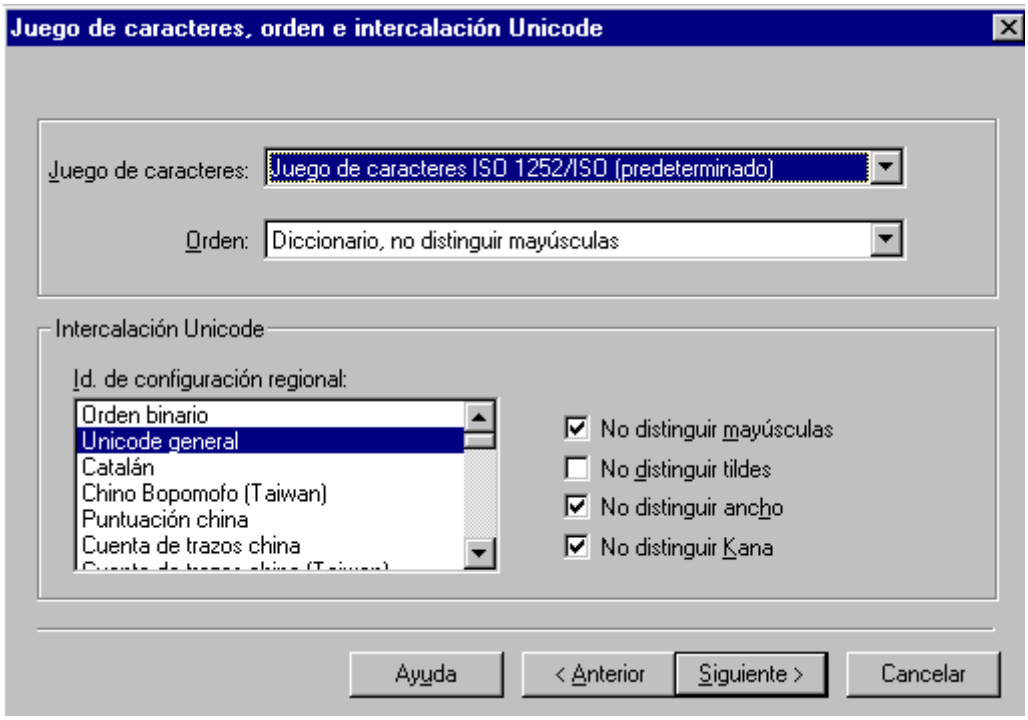


Figura 27. Elección del juego de caracteres y ordenación.

Veamos en detalle cómo podemos configurar cada uno de los elementos de este punto.

Juego de caracteres

Consiste en un grupo de 256 caracteres, de los cuales, los 128 primeros son comunes para todos los juegos, mientras que los 128 restantes representan a los caracteres particulares del idioma elegido.

La página seleccionada por defecto es *ISO 1252*, que es compatible con los caracteres empleados en los sistemas Windows, UNIX y VMS.

Es recomendable aunque no obligatorio, que el juego de caracteres seleccionado para SQL Server sea el mismo que el utilizado por Windows NT y también por los clientes que accedan al servidor de datos, de esta forma evitaremos incompatibilidades al utilizar los caracteres extendidos de cada juego.

Por otro lado, si es necesario trabajar con distintos idiomas en un servidor, deberemos seleccionar el juego de caracteres por defecto y usar tipos de datos Unicode al crear las bases de datos. Estos tipos de datos admiten un amplio rango de caracteres, dígitos y símbolos.

Orden

Se trata de las normas en las que se basa SQL Server para clasificar la información que es grabada en las bases de datos y devuelta en las consultas. Se permite una única ordenación para el servidor.

El tipo de orden predeterminado es de *diccionario sin distinguir mayúsculas y minúsculas*. Es importante establecer una ordenación adecuada, que no afecte negativamente (o lo menos negativamente posible) al rendimiento de la base de datos. Por ejemplo, siempre tardarán más tiempo

en ordenarse datos que deban distinguir entre mayúsculas y minúsculas, que datos que no necesiten realizar dicha distinción.

También es importante este aspecto a la hora de devolver información, ya que si utilizamos la mencionada distinción entre mayúsculas y minúsculas, al interrogar a la base de datos por el valor "Mesa", no se devolverán aquellos registros en los que pueda estar la palabra "mesa"; este hecho supone también una carga de trabajo extra para los clientes que atacan a la base de datos del servidor, puesto que se deben de encargar de realizar las conversiones oportunas antes de enviar sus consultas.

Finalmente, si necesitamos realizar copias de seguridad y restaurar dichas copias entre distintos servidores, debemos elegir el mismo tipo de ordenación para todos, lo que nos evitará problemas de incompatibilidad en este aspecto.

Intercalación Unicode

Los tipos de datos Unicode nos permiten mantener información en varios idiomas dentro de una base de datos. Debido a su diferencia con los tipos de datos habituales, necesitan una ordenación distinta de estos últimos, de ahí que sea necesario indicar lo que se denomina *intercalación* para datos Unicode.

Una intercalación Unicode incluye información local sobre la ordenación de los datos para una región específica dentro del idioma elegido. Salvo en casos en los que debamos realizar una ordenación en base a una de estas configuraciones, para evitar posibles problemas al realizar copias de seguridad de bases de datos entre distintos servidores y también a la hora de transferir datos no Unicode a Unicode, se recomienda utilizar la intercalación general, que está seleccionada por defecto.

Bibliotecas de red

Para permitir la comunicación entre los clientes de la red y SQL Server, debemos seleccionar a continuación las bibliotecas de red que realizarán este trabajo. Por defecto encontraremos seleccionadas *Canalizaciones con nombre*, *Sockets TCP/IP* y *Multiprotocolo*., véase la Figura 28.

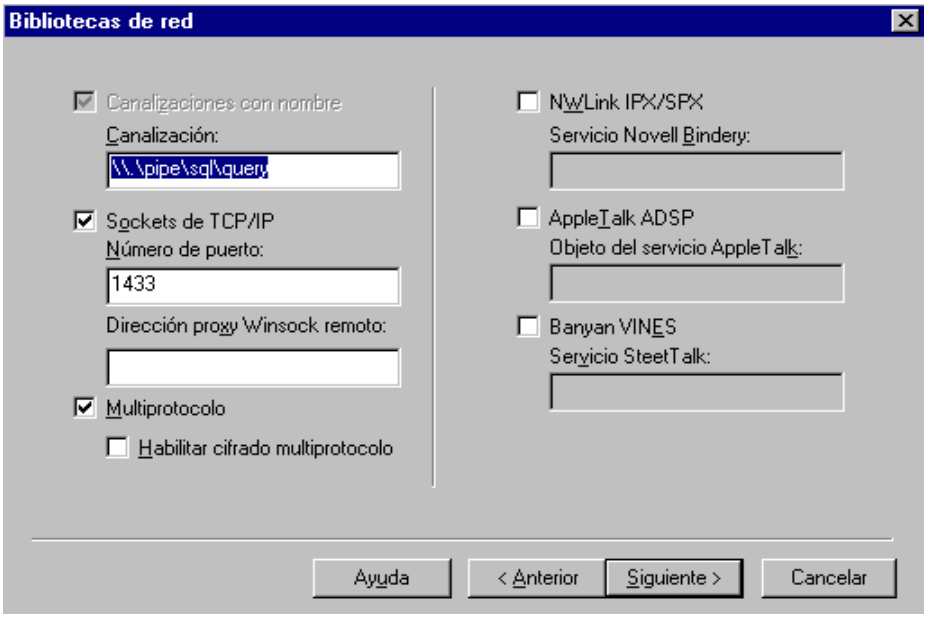


Figura 28. Bibliotecas de red disponibles.

Debemos tener en cuenta que es necesario que estén instalados los protocolos seleccionados en el cliente y el servidor para poder utilizar las bibliotecas aquí seleccionadas. De igual forma, si queremos aprovecharnos de las ventajas de la autenticación de Windows NT, es necesario instalar alguna de las bibliotecas seleccionadas por defecto.

Cuentas de servicios

Seguidamente debemos especificar la cuenta de usuario definida en Windows NT (operación descrita en el tema *Preparación de la instalación de SQL Server 7.0*), que servirá para iniciar los servicios de SQL Server.

Como vemos en la Figura 29, podemos especificar cuentas separadas para los diferentes servicios, emplear una cuenta de sistema local o de usuario de dominio, que como ya se explicó, es la opción recomendada para poder efectuar operaciones a través de la red.

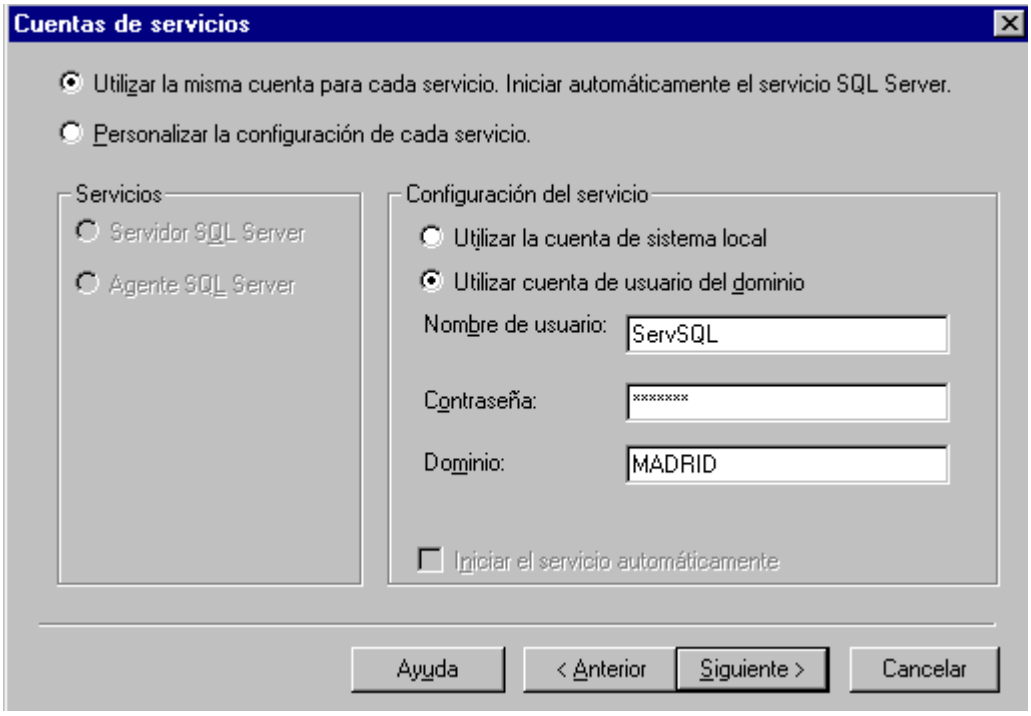


Figura 29. Cuenta del sistema para los servicios de SQL Server.

Durante esta etapa de instalación, se crea una cuenta de usuario local con el nombre SqlAgentCmdExec, sin derechos de administrador, que será utilizada por el servicio SQLServerAgent y el procedimiento almacenado extendido xp_cmdshell, para las operaciones que ejecuten usuarios que no sean administradores.

Modo de licencia

En este punto debemos indicar el tipo de licencia bajo el que se ejecutará SQL Server 7.0. Ver Figura 30.

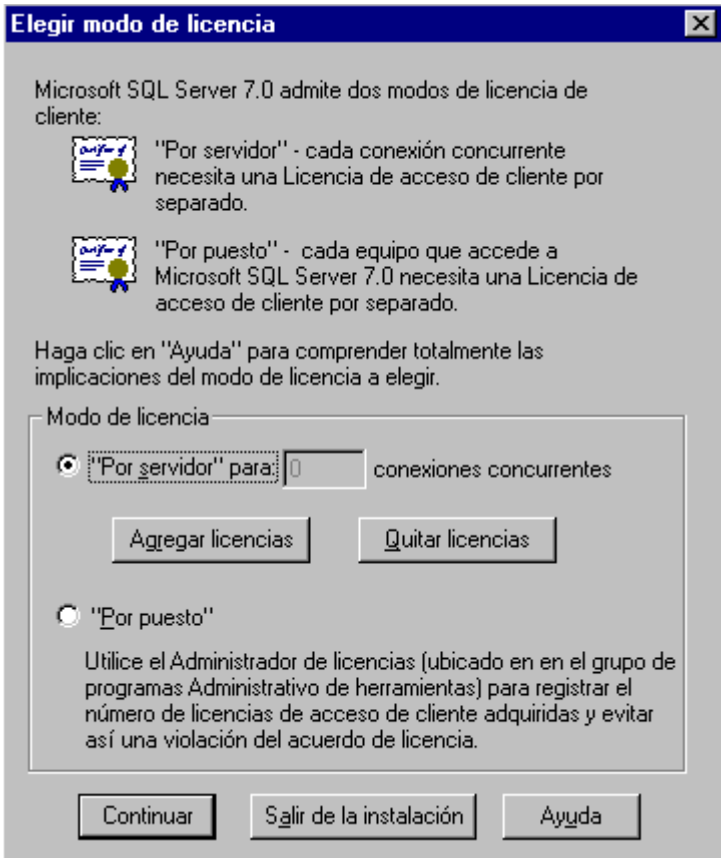


Figura 30. Elección del modo de licencia.

- Por servidor. En este tipo de licencia, las licencias de clientes se asignan a un servidor (Figura 31), de forma que debemos especificar cuantas licencias de cliente se disponen para el mencionado servidor, Figura 32.

- Por Puesto. Esta modalidad requiere una licencia por cada equipo que vaya a realizar accesos a un servidor Windows NT, con la ventaja de poder acceder a cualquier servidor que haya instalado en la red.

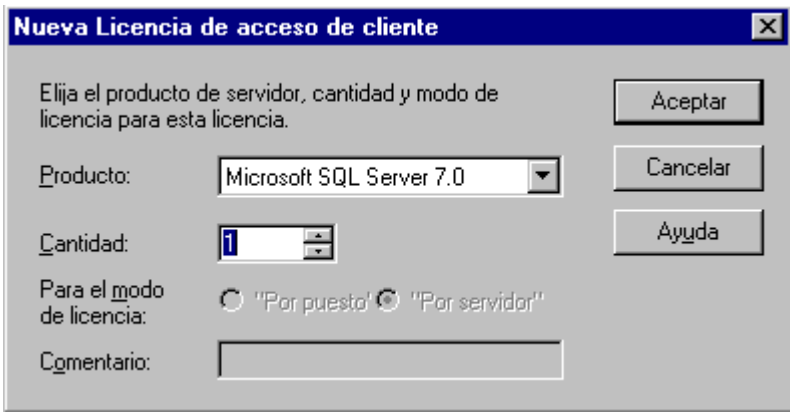


Figura 31. Establecimiento del número de licencias.

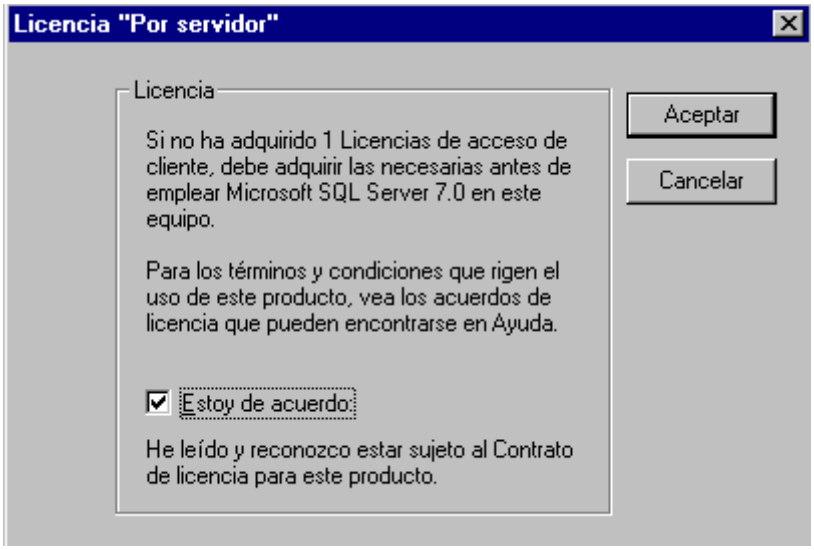


Figura 32. Confirmación del número de licencias.

Existe la posibilidad de realizar solamente un cambio del modo de licencia por servidor a otra por puesto en el caso de que consideremos más conveniente esta última, por lo que si no estamos seguros del tipo a elegir, podemos decantarnos a la modalidad por servidor y cambiar posteriormente.

Completado este paso, el programa nos avisa de que dispone de toda la información necesaria, tras lo que aceptamos el mensaje y comienza el proceso de instalación de SQL Server a nuestro equipo, que se realiza en los siguientes pasos:

- Copia de ficheros.
- Instalación de Microsoft Data Access Components (MDAC).
- Instalación de Microsoft Management Console (MMC).
- Instalación del coordinador de transacciones distribuidas (MDTC).
- Configuración del motor de datos.
- Comprobación de los servicios SQL Server.
- Registro de componentes ActiveX.

Los anteriores pasos son realizados automáticamente por el programa de instalación, sin que sea necesario que el usuario intervenga. Una vez terminados y si no ha sido mostrado ningún mensaje de error, la instalación de SQL Server 7.0 habrá finalizado satisfactoriamente.

Instalación desatendida

A pesar de la sencillez del proceso de instalación descrito anteriormente, puede convertirse en una labor tediosa en el caso de que tengamos que realizar múltiples instalaciones con la misma configuración. Para solventar este problema, SQL Server nos permite realizar una instalación desatendida mediante un fichero por lotes y otro de configuración que contienen las instrucciones y configuración para dicha instalación.

En la Tabla 1 se indican los ficheros disponibles y el tipo de instalación que realizan. Estos ficheros se encuentran en el CD de instalación de SQL Server.

Fichero de órdenes por lotes	Fichero de configuración	Tipo de instalación
SQL70CLI.BAT	SQL70CLI.ISS	Herramientas administrativas
SQL70INS.BAT	SQL70INS.ISS	Instalación típica con la cuenta del sistema local de Windows NT
SQL70CST.BAT	SQL70CST.ISS	Instalación personalizada con la cuenta del sistema local de Windows NT y todos los componentes habituales de SQL Server

Tabla 1. Ficheros de órdenes y configuración, para realizar una instalación desatendida.

Al ejecutar uno de los archivos de órdenes se detecta el sistema operativo, selecciona la información de configuración y ejecuta el archivo correspondiente de inicialización, que contiene las opciones que el usuario debería elegir en caso de realizar una instalación manual.

También es posible crear ficheros de inicialización personalizados de dos formas:

- Tomando uno de los ficheros existentes y modificándolo mediante un editor de texto allá en los puntos que necesitemos personalizar.

- Al realizar una instalación de SQL Server en la ruta Unidad:\MSSQL7\INSTALL, se crea un fichero con el nombre SETUP.ISS, conteniendo los pasos realizados en dicha instalación. Debemos editar este fichero para añadir las órdenes de inicio y fin de copia. En el caso de que no dispongamos de este archivo, ejecutaremos en una ventana MS-DOS la instrucción Setupsql.exe k=Rc, que creará una copia de dicho archivo.

Existe otra vía para crear instalaciones desatendidas bajo Windows NT consistente en utilizar Microsoft Systems Management Server, que nos permite realizar este tipo de operaciones desde el entorno de una aplicación.

6

Supervisión y configuración de la instalación

Comprobar los principales elementos del servidor de datos

Finalizado el proceso de instalación de SQL Server, es muy recomendable revisar los elementos básicos que se acaban de instalar aún en el caso de no haberse mostrado ningún error, ya que durante el proceso interno de registro de componentes pudiera haberse producido algún fallo no detectable a simple vista.

Ya que algunas de estas tareas de revisión implican el uso de las herramientas administrativas del producto, este tema servirá como introducción al uso de dichas herramientas, que utilizaremos con profusión a lo largo del curso debido a su clara orientación en las labores de administración del sistema.

Ejecutar cualquiera de estas aplicaciones, como el Administrador de servicios, Administrador corporativo, etc., para comprobar su correcto funcionamiento después de finalizar la instalación de SQL Server, es un medio muy conveniente de comprobar que las conexiones con los servicios más fundamentales del gestor se están llevando a cabo correctamente.

Administrador de servicios de SQL Server

Esta aplicación (Figura 33), permite gestionar de una forma cómoda y gráfica los diferentes servicios de SQL Server que se ejecutan en el sistema operativo, la siguiente imagen muestra el aspecto de esta utilidad.

Figura 33. Administrador de servicios de SQL Server 7.0.

El campo Servidor nos permite seleccionar el servidor sobre el que gestionar los servicios. En Servicios, podemos elegir cuál de los servicios del gestor de datos manipular. Una vez seleccionado un servicio, podemos iniciarlo, pararlo momentáneamente o detenerlo de forma permanente mediante los botones a tal efecto ubicados en esta ventana.

Para ahorrar al administrador la pesada tarea de iniciar los servicios cada vez que arranque el sistema operativo, podemos marcar la casilla *Iniciar automáticamente con el SO*, que ejecutará el servicio marcado al mismo tiempo que el sistema operativo se inicia.

Podemos acceder al administrador de servicios mediante la ruta de menú de Windows NT *Inicio+Programas+Microsoft SQL Server 7.0+Administración de servicios*, o bien haciendo clic con el botón derecho en el icono situado en el extremo derecho de la barra de menús del sistema operativo, ver Figura 34, lo que abrirá un menú contextual con diversas opciones de configuración.

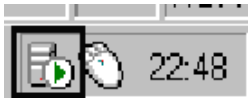


Figura 34. Icono en la barra de menús del Administrador de servicios.

El Administrador corporativo

Esta herramienta nos permite, mediante un sencillo interfaz gráfico, realizar prácticamente todas las tareas disponibles en SQL Server, o en su defecto, inicia el asistente adecuado que nos guiará para realizar la operación que necesitemos.

Dicho interfaz está basado en el sistema de consola Microsoft Management Console (MMC), que proporciona un medio unificado para administrar los diferentes componentes de BackOffice.

Cuando abrimos el Administrador corporativo, la ventana principal pertenece a la consola de Microsoft, mientras que la ventana que contiene la consola representa al Administrador corporativo, como muestra la Figura 35.

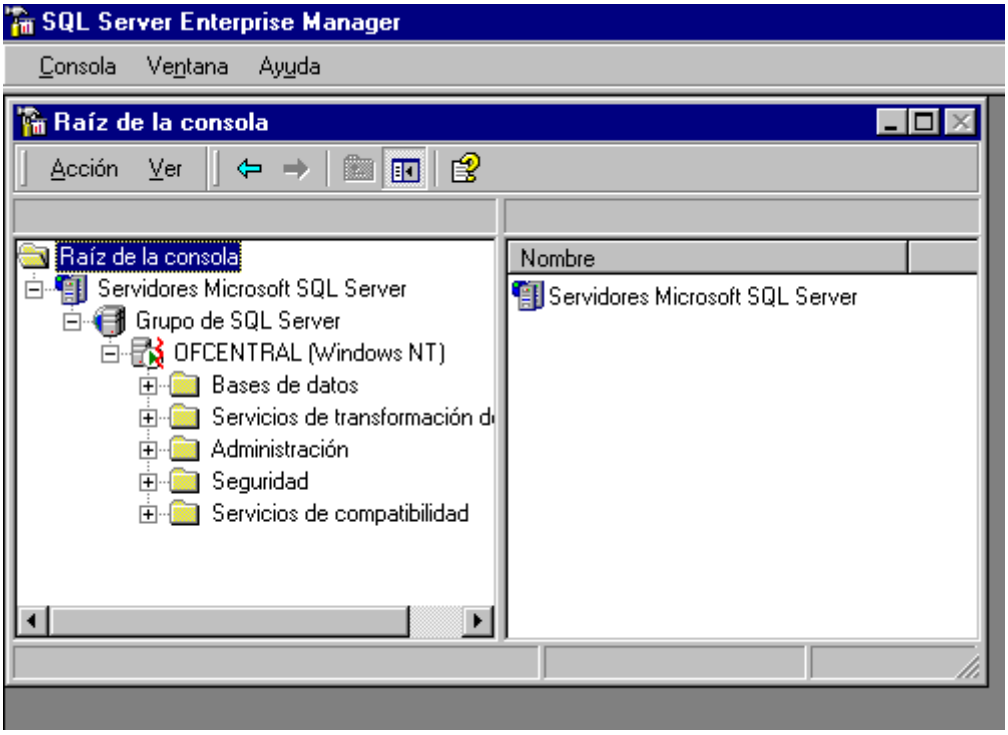


Figura 35. Administrador corporativo ejecutándose dentro de la consola MMC.

Todas las operaciones en esta herramienta están disponibles haciendo clic con el botón derecho sobre el elemento a utilizar y seleccionando la opción correspondiente en el menú contextual, o bien abriendo el menú Acción de la ventana, por lo que en adelante nos referiremos indistintamente a uno u otro medio de selección.

Grupos de servidores

Como el lector acaba de comprobar, los servidores dentro del administrador corporativo se organizan en una estructura jerárquica de grupos de servidores. Dentro de un grupo se pueden crear grupos anidados o subgrupos, dependiendo de la complejidad del conjunto de servidores a administrar.

Una vez creado un grupo, la forma de añadirle un servidor es registrarlo dentro de dicho grupo. En este apartado se describen los pasos para crear nuevos grupos dentro del administrador corporativo, mientras que el registro de servidores será tratado a continuación.

Al finalizar la instalación de SQL Server, se crea un grupo por defecto con el nombre *Servidores Microsoft SQL Server*, y dentro de este, un grupo llamado *Grupo de SQL Server*, en el que se registra automáticamente el servidor local.

Para crear un nuevo grupo al mismo nivel que *Grupo de SQL Server*, haremos clic con el botón derecho sobre este grupo y seleccionaremos la opción *Nuevo grupo de SQL Server* del menú, que abrirá la ventana *Grupos de servidores*, en la que podremos indicar el nombre del nuevo grupo y el nivel de grupo en el que se va a situar: superior o subgrupo, ver Figura 36.

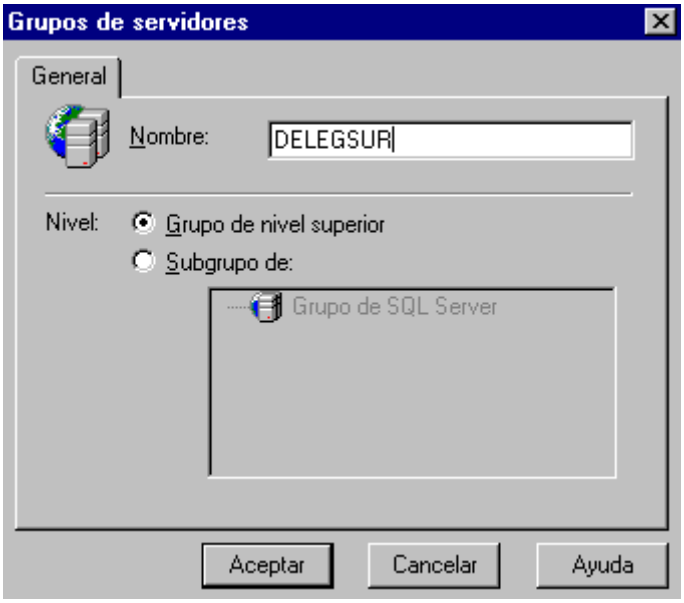


Figura 36. Creación de un nuevo grupo de servidores.

En este ejemplo vamos a crear un nuevo grupo denominado DELEGSUR, que va a estar al mismo nivel que el grupo por defecto de SQL Server. Al pulsar Aceptar, la organización de grupos queda como muestra la Figura 37.

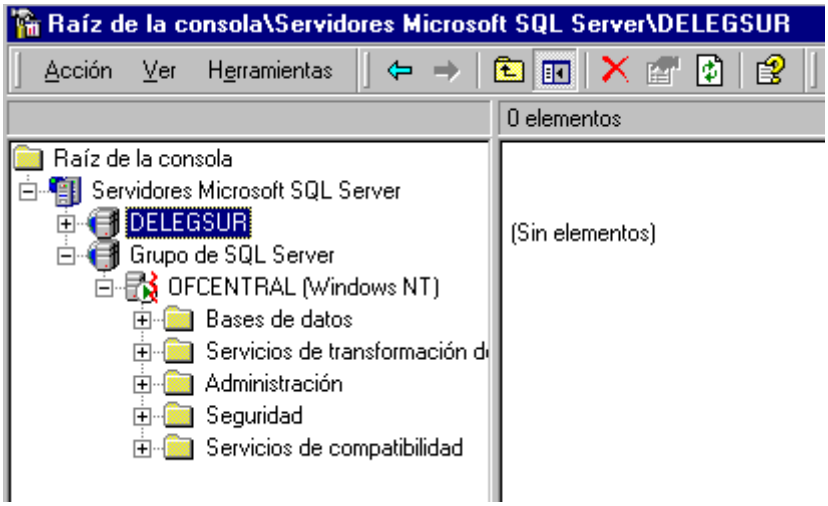


Figura 37. Jerarquía de grupos tras la creación de uno nuevo.

Registrar un servidor

Para esta operación, haremos clic con el botón derecho sobre el grupo dentro del cual queremos registrar el nuevo servidor, seleccionando la opción del menú contextual *Nuevo registro de servidor SQL Server*, que iniciará el *Asistente para registro de un servidor SQL Server*, Figura 38.

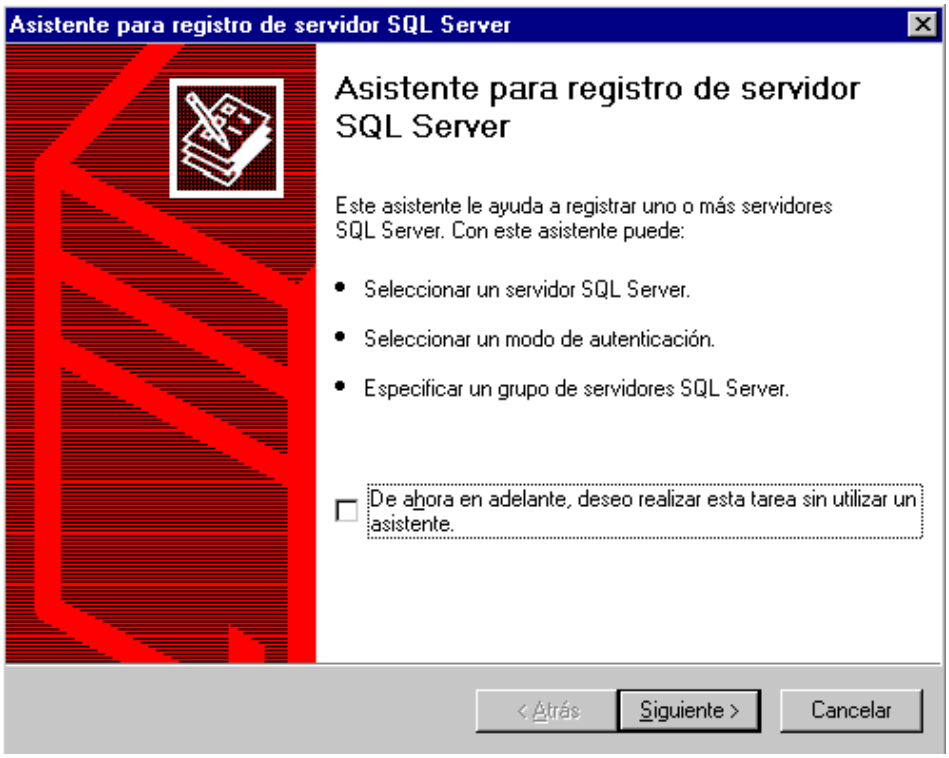


Figura 38. Asistente para registro de un servidor SQL Server.

En el primer paso de este asistente, seleccionaremos el servidor que vamos a registrar entre la lista de servidores disponibles y pulsaremos Agregar, ver Figura 39.

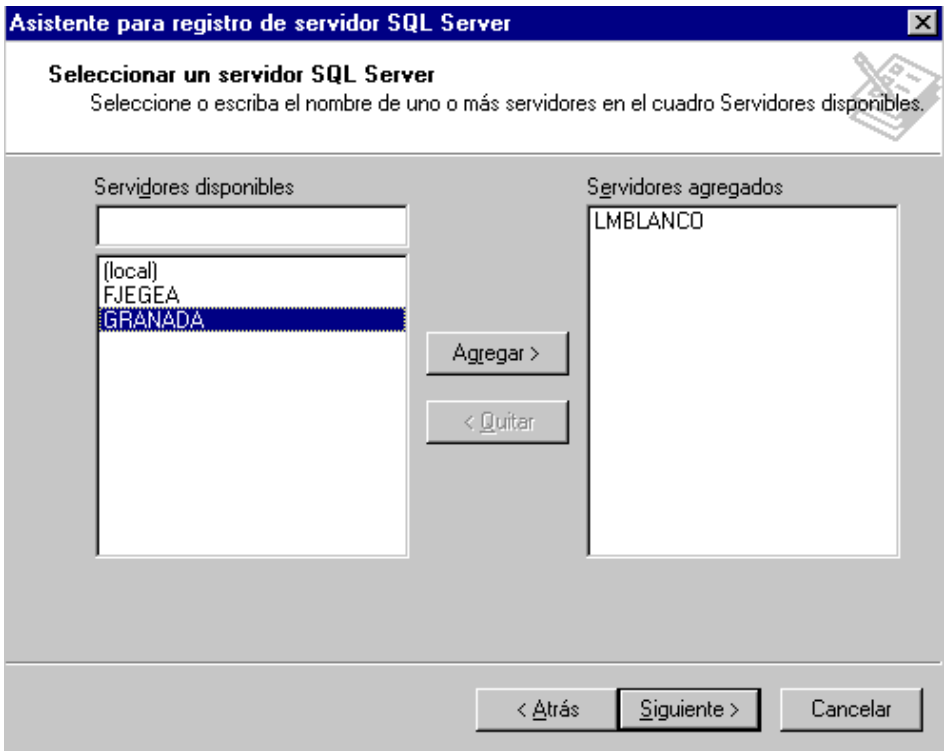


Figura 39. Seleccionar servidor SQL Server a registrar.

A continuación especificaremos el modo de autenticación para conectar con el servidor, como vemos en la Figura 40.

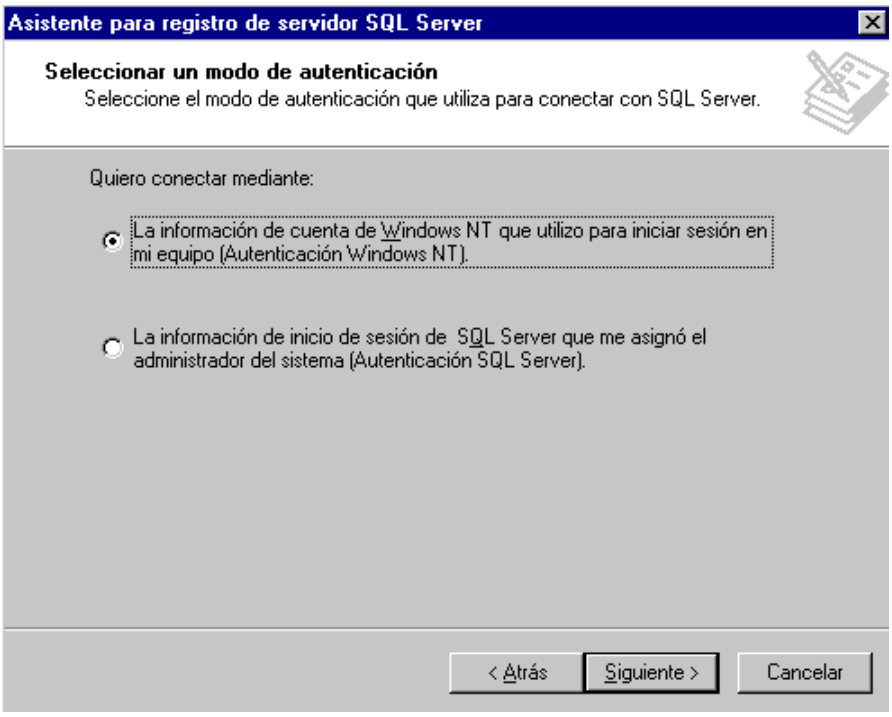


Figura 40. Indicar modo de autenticación para la conexión con el servidor.

En el siguiente paso indicaremos el grupo al que vamos a añadir el servidor, o bien, crearemos un nuevo grupo para dicho servidor, Figura 41.

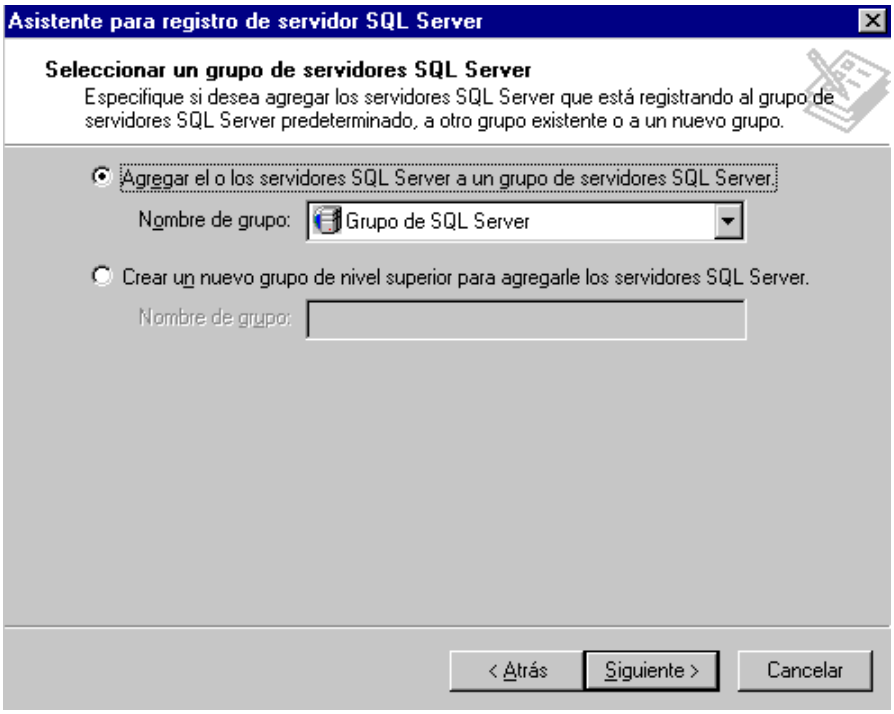


Figura 41. Selección del grupo de servidores al que se agregará el nuevo servidor.

Seguidamente se indica al usuario que se ha completado el asistente, con lo que al pulsar Finalizar, se intentará crear la conexión con el servidor especificado, ver Figura 42.

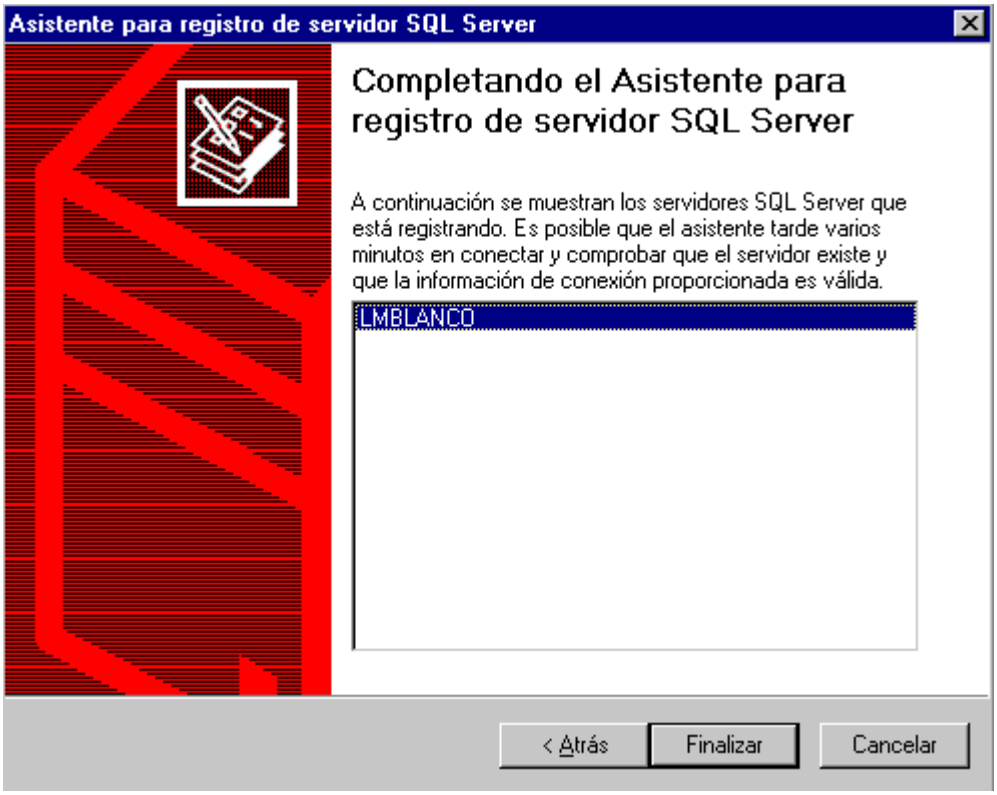


Figura 42. Final del asistente para registrar servidores SQL Server.

Tras el proceso de conexión, se avisará al usuario del resultado mediante un cuadro de diálogo como el mostrado en la Figura 43.

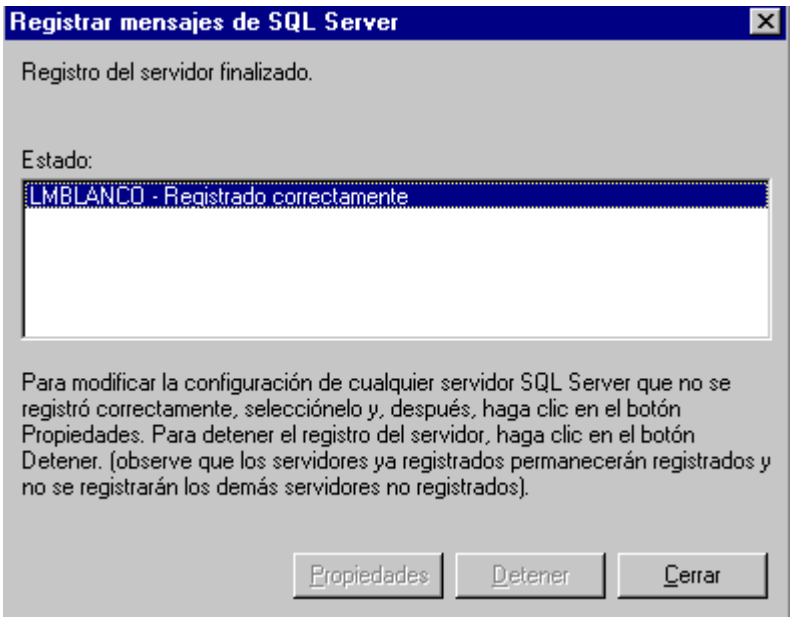


Figura 43. Resultado de la operación de registro de un servidor SQL Server.

Para conectar con un servidor y poder acceder a los objetos que contiene, sólo hemos de pulsar sobre el mismo en el Administrador corporativo, estableciéndose la conexión de forma automática.

En el caso de que al iniciarse este asistente, marquemos la casilla *De ahora en adelante, deseo realizar esta tarea sin utilizar un asistente*, realizaremos el registro del servidor de forma manual, lo que nos mostrará la ventana de propiedades de registro para el servidor SQL que vemos en la Figura 44.

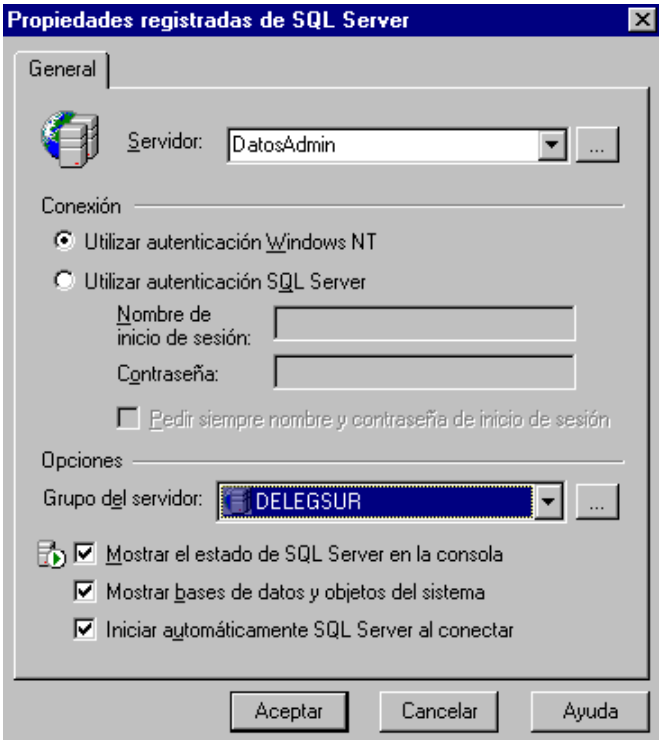


Figura 44. Propiedades de registro de un servidor SQL Server.

Debemos especificar el nombre del servidor que queremos registrar, el tipo de autenticación para la conexión, grupo del servidor dentro del que vamos a incluirlo y si queremos visualizar diversa información de estado del servidor una vez que hayamos conectado con él. Tras pulsar Aceptar se realizará la operación de registro, incluyéndose el nuevo servidor en la estructura de la consola del administrador, ver Figura 45.

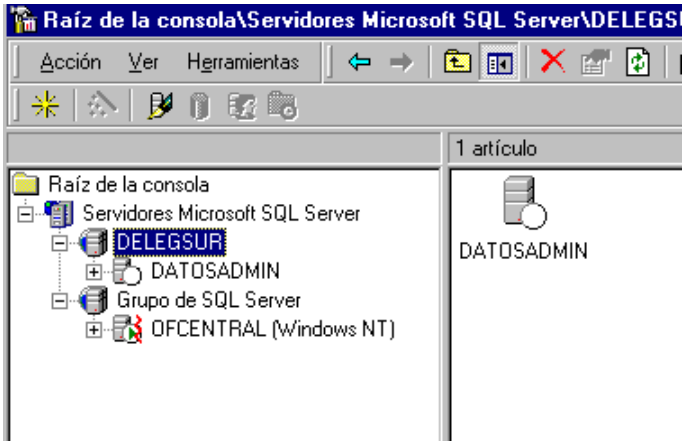


Figura 45. Grupos de servidores con sus servidores SQL Server correspondientes.

Propiedades del servidor

Un servidor se registra con una serie de valores por defecto que asigna SQL Server. Si necesitamos consultar o modificar cualquiera de estos valores, haremos clic en el servidor correspondiente y seleccionaremos el menú *Acción+Propiedades* del Administrador corporativo, que nos mostrará la ventana de propiedades del servidor, en la que disponemos de una serie de pestañas que agrupan las propiedades por categorías, ver Figura 46.

Figura 46. Propiedades de un servidor SQL Server, categoría General.

Dentro de la diversa información disponible, podemos entre otras propiedades, consultar el sistema operativo bajo el que se ejecuta el servidor de datos, memoria física instalada, idioma, iniciar el correo del servidor, aspectos de compatibilidad con el año 2000, asignar manualmente la memoria utilizada y la dedicación del trabajo de procesadores, etc.

La Figura 47 muestra los valores de configuración del servidor.

En la Figura 48 vemos la configuración de los parámetros de memoria para el servidor.

La configuración del procesador para SQL Server se muestra en la Figura 49.

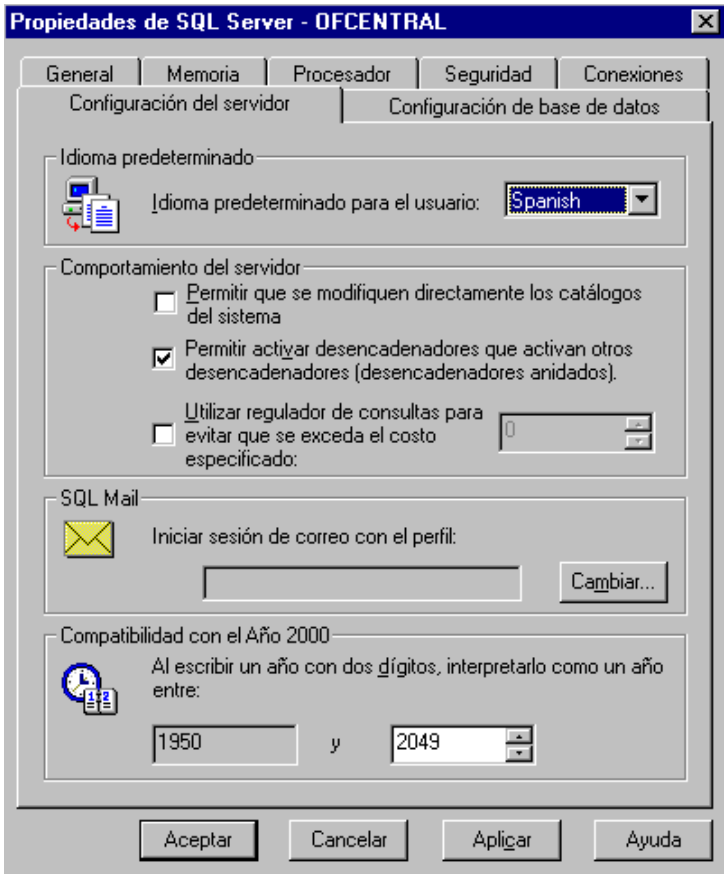


Figura 47. Propiedades de configuración de un servidor SQL Server.

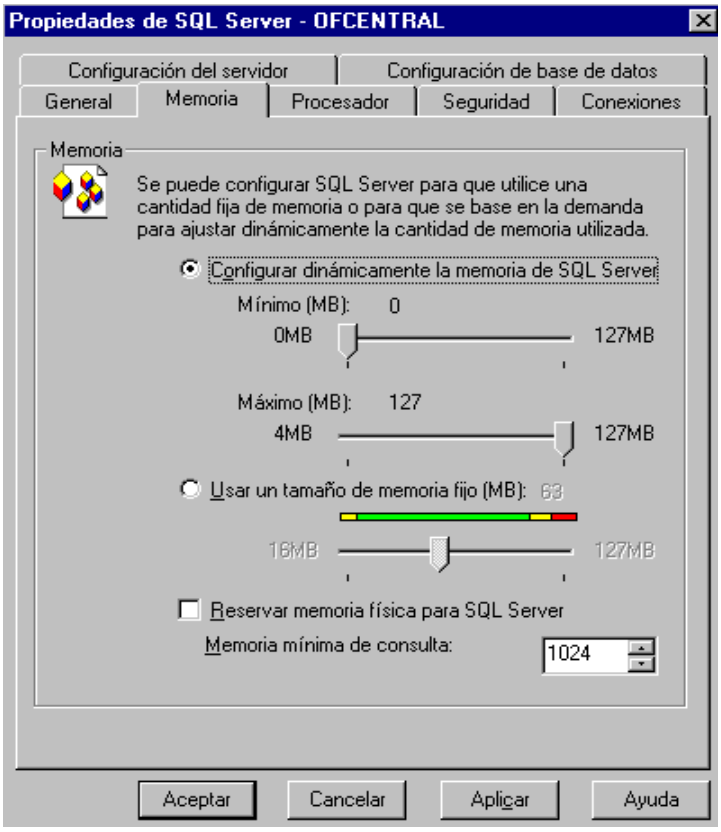


Figura 48. Propiedades de configuración de memoria para un servidor SQL Server.

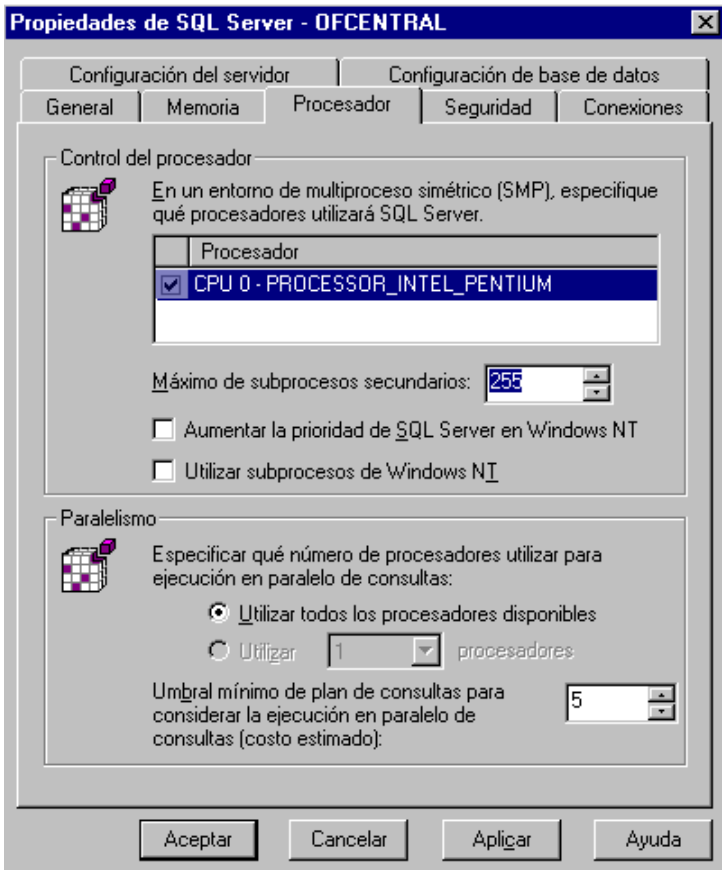


Figura 49. Configuración de los valores del procesador para un servidor SQL Server.

A pesar de disponer de todas estas opciones de configuración, es muy importante reseñar que en la actual versión de SQL Server se ha realizado un enorme esfuerzo para lograr que todas las tareas de administración de recursos sean gestionadas internamente por el servidor de datos y optimizadas a los valores más adecuados, de forma que en condiciones de trabajo normales, no será necesario que nos preocupemos por el ajuste preciso de los recursos.

La cuenta de inicio de sesión sa

Al instalar SQL Server, se crea una cuenta con el nombre sa que corresponde al administrador del sistema gestor de datos, con plenos permisos para acceder a todos los objetos del servidor, pero sin contraseña, lo cual entraña un considerable riesgo por los posibles accesos que pudieran realizar usuarios no autorizados que conozcan la existencia de esta cuenta.

Para evitar problemas de accesos no deseados, debemos desplegar en el Administrador corporativo la estructura de carpetas del servidor, abrir la carpeta Seguridad, y hacer clic en *Inicios de sesión*, lo que nos mostrará en el panel derecho las cuentas disponibles en el servidor, ver Figura 50.

Haremos clic sobre la cuenta sa y elegiremos la opción de menú *Acción+Propiedades* del Administrador corporativo, que nos mostrará su ventana de propiedades, véase la Figura 51. En el campo Contraseña de esta ventana de propiedades, escribiremos el nombre de la contraseña y aceptaremos la ventana. De esta forma el usuario que emplee la cuenta sa deberá conocer además la contraseña de acceso.

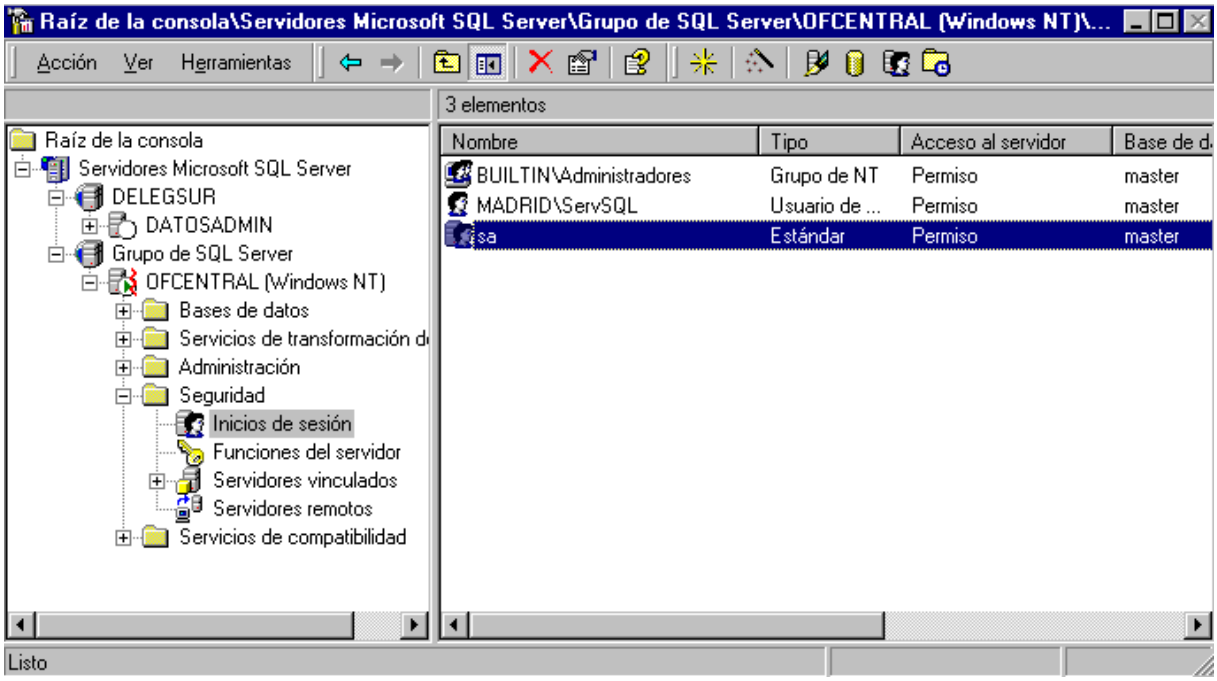


Figura 50. Inicios de sesión de un servidor SQL Server.

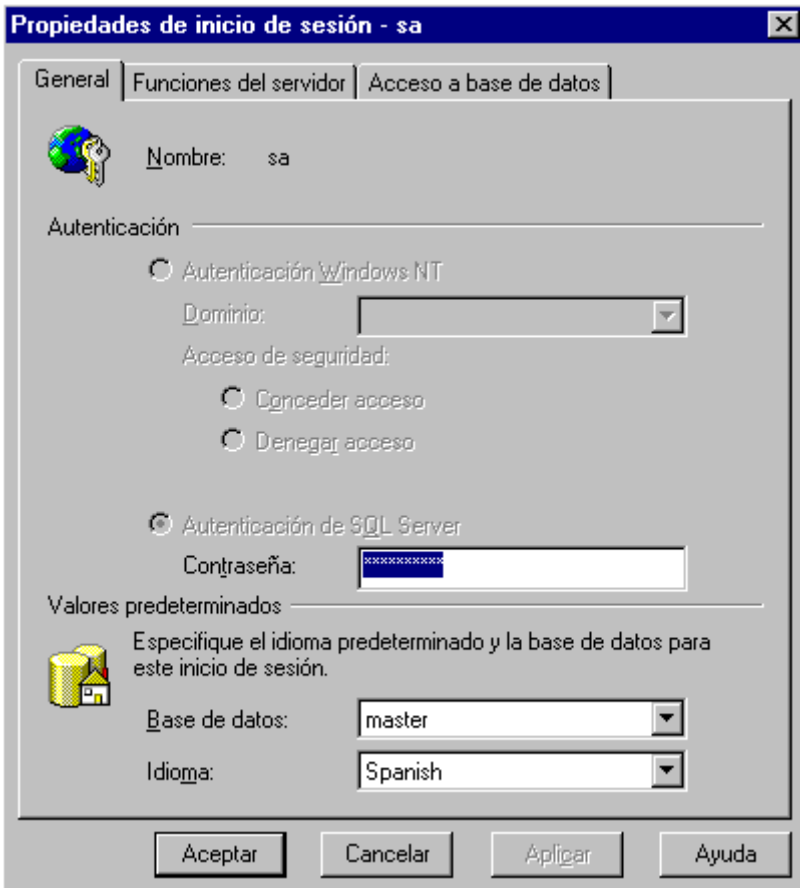


Figura 51

7

Aspectos conceptuales, estructuras de datos y almacenamiento

La base de datos, objetos y elementos físicos

Cuando trabajamos con SQL Server, los diferentes elementos que componen un servidor, desde la propia base de datos, pasando por las tablas, vistas, procedimientos almacenados, etc., son tratados como objetos, de forma que se facilite su manejo para el usuario del servidor, ya sea un administrador o un programador de aplicaciones, mediante el nivel de abstracción que permite el trabajo con objetos.

Pero por otra parte, debemos pensar que todos estos objetos deben de ubicarse físicamente en el disco del equipo en donde esté instalado el motor de datos. El almacenamiento tanto de la información como de los objetos que se utilizan para acceder a ella, se realiza dentro de archivos de bases de datos, que no son otra cosa que archivos del sistema operativo organizados mediante una serie de reglas que nos permiten acceder a la información en ellos grabada.

En este tema revisaremos los objetos más importantes de una base de datos, la estructura o sistema utilizado para su almacenamiento y la manipulación de los ficheros físicos que albergan toda esta información.

Objetos de base de datos

A continuación, se relacionan los objetos más importantes que forman parte de una base de datos, junto a una breve descripción.

- Base de datos. Es el objeto principal en la jerarquía, de él dependen el resto de objetos relacionados con el almacenamiento de datos.

- Tabla. Contiene la información sobre un tema concreto de la base de datos, por ejemplo, la tabla de Clientes. Se organiza en base a filas o registros, y cada fila está compuesta por columnas o campos.

  En el ejemplo comentado, cada fila proporcionaría información sobre un sólo cliente y cada columna información sobre una característica del cliente, por ejemplo, la columna Dirección nos indica el lugar de residencia del cliente.

- Índice. Permite el acceso a las filas de una tabla, a través de un orden determinado. Dependiendo de su tipo, no cambia el orden físico con el que se han agregado los registros a la tabla, se trata de un conjunto de indicadores (punteros) a las filas, que permiten mostrar la tabla en un orden lógico, distinto del original.

- Vista. Consiste en una selección o consulta de registros contra una o más tablas, que puede almacenarse en la base de datos, para poder ejecutarse más rápidamente.

- Procedimiento almacenado. Es un conjunto de instrucciones en Transact-SQL (la versión del lenguaje SQL que incorpora SQL Server), dispuesto en forma de rutina ejecutable, al estilo de los lenguajes de programación, que permiten realizar diversas operaciones en un sólo paso.

- Desencadenador. También conocido como disparador o trigger, se trata de una variante del procedimiento almacenado, que se ejecuta cada vez que se produce un suceso en la base de datos, como agregar un registro, modificarlo, etc.

Tipos de bases de datos en SQL Server

En SQL Server existen dos clases de bases de datos, que se describen a continuación.

Base de datos del sistema

Este tipo de base de datos se crea durante el proceso de instalación de SQL Server, conteniendo información variada sobre el gestor de datos. Veamos cuales son estas bases de datos.

- Master. Es la base de datos más importante, contiene información sobre todo el sistema, ubicaciones de los diferentes ficheros de las demás bases de datos, usuarios, configuración del sistema, etc.

- Model. Sirve como base o plantilla para las nuevas bases de datos creadas por el usuario.

- Msdb. Es utilizada por el servicio Agente SQL Server para los trabajos programados por el administrador del servidor.

- Distribution. Contiene la información realizada en las operaciones de duplicación.

- Tempdb. Almacena elementos temporales de la sesión de trabajo actual, creados por el propio servidor de datos y todos los usuarios que se conectan a él, como tablas, procedimientos almacenados, etc. Cada vez que se inicia SQL Server, se elimina todo su contenido, de forma que siempre partimos de cero en cada sesión.

Base de datos de usuario

Son las bases de datos creadas por el administrador del sistema o cualquier otro usuario con los permisos adecuados, y contienen la información referente al negocio de la empresa: clientes, cuentas contables, facturación, empleados, etc.

SQL Server incluye a modo de ejemplo, las bases de datos de usuario Pubs y Northwind, para que el usuario pueda realizar todo tipo de pruebas y familiarizarse con el sistema.

Metadatos

Los metadatos son datos sobre los propios datos. En el caso de SQL Server se trata de elementos del sistema como tablas, vistas, procedimientos almacenados, etc., que contienen información interna sobre el mismo, como por ejemplo: sysmessages, sysdatabases, syslogins, etc. Son generados para todo el servidor, y están contenidos en la base de datos Master.

Otros metadatos como las tablas sysusers y sysobjects, son particulares para cada base de datos de usuario, es decir, cada base de datos de usuario tiene su propia copia de sysusers y sysobjects, que son generadas cada vez que se crea una nueva base de datos de usuario.

En el caso de las tablas que contienen información interna del sistema, no es recomendable manipularlas directamente, ya que esto podría causar que el servidor de datos se volviera irrecuperable en caso de error. Para solventar este inconveniente, SQL Server dispone de elementos de consulta, algunos de los cuales, se relacionan a continuación.

Procedimientos almacenados

- sp_helpdb. Devuelve información sobre una o todas las bases de datos del servidor, en función de si pasamos como parámetro el nombre de una base de datos o no.

- sp_helpindex. Informa sobre los índices de la tabla pasada como parámetro.

Funciones del sistema

Proporcionan acceso a las tablas del sistema mediante funciones Transact-SQL.

- DB_ID(NombreBD). Proporciona el identificador de la base de datos pasada como parámetro.

- COL_LENGTH(NombreCol). Devuelve el ancho de la columna pasada como parámetro.

- USER_NAME(ID). Devuelve el nombre del usuario del cual hemos pasado su identificador.

Vistas

Contienen como ya se ha mencionado, consultas almacenadas en la base de datos.

- INFORMATION_SCHEMA.TABLES. Devuelve los nombres de las tablas de una base de datos.

- INFORMATION_SCHEMA.COLUMNS. Devuelve información sobre las columnas de una base de datos.

Un motor de datos renovado

Hasta la versión 6.5, SQL Server arrastraba una arquitectura de almacenamiento de información basada en dispositivos, cuya administración era bastante incómoda, por el gran número de tareas de configuración a realizar, quizás debido al legado del motor de datos de las primeras versiones de este servidor.

SQL Server 7.0 parte de un motor de datos completamente renovado, que le permite dejar a un lado los problemas ocasionados por las anteriores versiones y adaptarse más fácilmente a los requerimientos de las aplicaciones de gestión actuales.

Una de las novedades más interesantes en cuanto a la administración, consiste en que en la actual versión, el servidor asume un gran número de tareas que realiza automáticamente, eliminando en todo lo posible la intervención del administrador.

En cuanto al almacenamiento de información, se ha eliminado el sistema de dispositivos de la versión 6.5 por otro basado en archivos del sistema operativo, que proporciona una mayor escalabilidad para las bases de datos.

Unidades de almacenamiento

A pesar de no ser absolutamente necesario para su manejo, sí es muy recomendable tener unos ciertos conocimientos básicos acerca del mecanismo interno del motor de datos, de forma que a la hora de diseñar una nueva base de datos, esta información nos ayude a crearla de la manera más efectiva posible para que dicho motor no soporte una carga de trabajo extra debido a un mal diseño de la base de datos.

SQL Server 7.0 dispone de dos unidades básicas de almacenamiento: páginas y extensiones, que se describen a continuación.

Página

Es la unidad primordial de almacenamiento de datos, consiste en un elemento que tiene un tamaño de 8 KB, en cuya zona inicial o cabecera, que contiene 96 bytes, reside la información que necesita el servidor sobre el espacio libre de la página, el objeto al que pertenece, tipo de página, etc. Ver esquema de la Figura 52.

Se almacenan de forma contigua en disco, pudiendo disponer de 128 páginas por cada MB. Aunque en las páginas se guarda información de todo tipo, las asociaremos más frecuentemente a las filas de tablas de la base de datos. Debemos tener en cuenta que una fila puede tener como máximo el tamaño de una página, lo que supone una cantidad de 8060 bytes descontando el tamaño de la cabecera.

En la Tabla 2, se relacionan los tipos de páginas existentes en SQL Server.

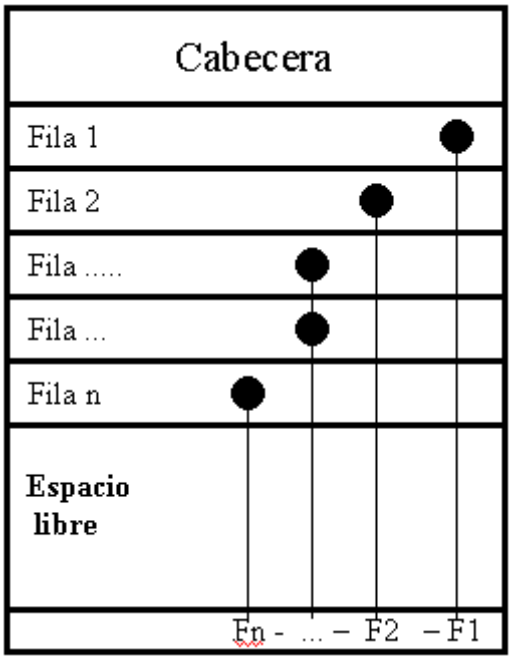


Figura 52. Esquema de una página de SQL Server 7.0

Tipo	Contenido
Datos	Filas de datos exceptuando los tipos text, ntext y image
Índice	Entradas de índices
Texto o imagen	Datos de tipo text, ntext y image
Mapa de asignación global	Información sobre extensiones
Espacio libre en página	Información sobre espacio libre en páginas
Mapa de asignación de índices	Información sobre extensiones usadas por tablas o índices

Tabla 2. Tipos de página en SQL Server.

Extensión

Es la unidad básica de representación para tablas e índices, consiste en una agrupación de 8 páginas, de forma que el tamaño resultante de una extensión es de 64 KB.

Para optimizar los recursos de espacio en las extensiones, SQL Server no asigna una extensión a una tabla con una pequeña cantidad de datos. Este hecho provoca que existan dos tipos de extensiones que detallamos a continuación.

- Mixta. Extensión compuesta por varias tablas de pequeño tamaño que ocupan el tamaño completo de la extensión, ver Figura 53.

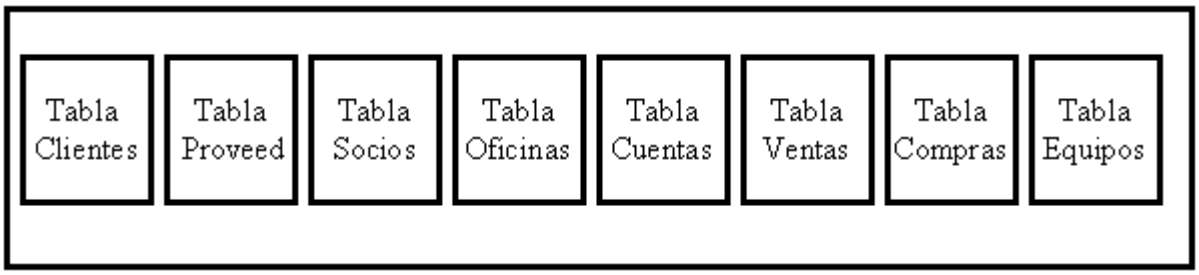


Figura 53. Esquema de una extensión mixta.

- Uniforme. Extensión compuesta por una única tabla. Cuando una tabla situada en una extensión mixta crece hasta llegar al tamaño de una extensión, SQL Server traslada automáticamente esta tabla a una extensión uniforme.

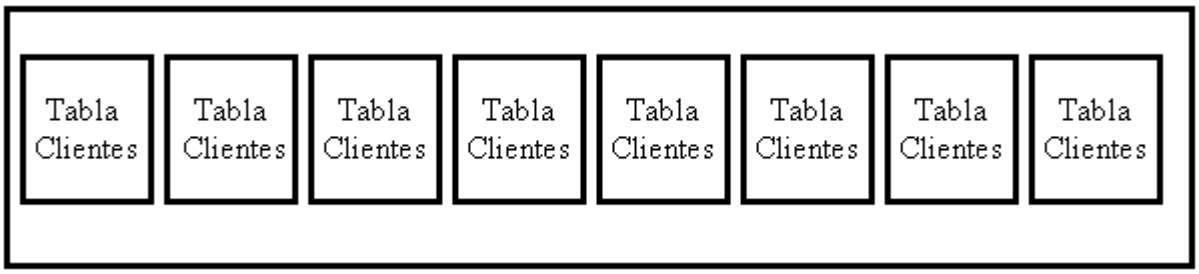


Figura 54. Esquema de una extensión uniforme.

Archivos de datos

En SQL Server 6.5, un archivo de datos podía contener más de una base de datos del servidor, sin embargo esta situación ha cambiado totalmente en la actual versión, ya que dentro de un archivo está la información referida a una base de datos, si bien es posible tener más de un archivo asociado a la misma base de datos, lo que permitirá su ampliación según vaya creciendo de tamaño.

A continuación se describen los tres tipos de archivo que puede tener una base de datos.

Principal

Contiene la información fundamental sobre la base de datos, tablas del sistema y sirve como referencia al resto de archivos de la base de datos. Sólo puede existir un archivo principal por base de datos. La extensión asociada por defecto a este tipo de archivos es .MDF, si bien es posible utilizar otra diferente.

Secundario

Son archivos que sirven de complemento al principal y que permiten situar tablas e índices en una ubicación distinta del archivo principal. Una base de datos puede tener uno, varios o ningún archivo secundario, ya que no son obligatorios. Su mayor utilidad reside en la posibilidad que otorga de crecimiento a la base de datos, si por ejemplo, tenemos el archivo principal en un disco que está a punto de llenarse, podemos utilizar un archivo secundario en otro disco para seguir grabando información de la base de datos. La extensión por defecto para estos archivos es .NDF.

Registro (log)

Contienen la información del registro de transacciones, necesaria para recuperar datos en el caso de problemas. La extensión de estos archivos es .LDF, y podemos tener uno o varios en la base de datos.

El Analizador de consultas

Aunque el Administrador corporativo será la herramienta principal del usuario encargado de las tares administrativas sobre el servidor, no podemos dejar de comentar brevemente las principales características del Analizador de consultas. Esta herramienta, ubicada en el grupo de programas de SQL Server (Figura 55), permite mediante una interfaz gráfica, ejecutar sentencias contra los diferentes servidores de datos a los que tengamos acceso.

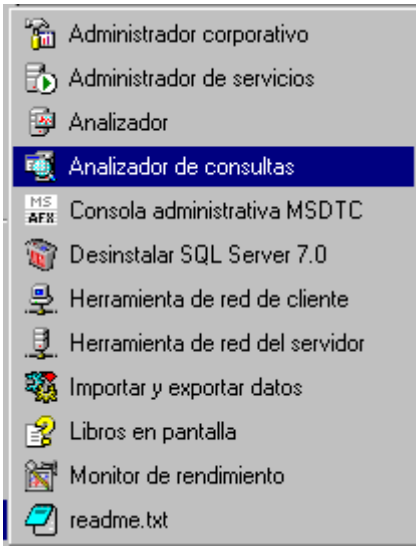


Figura 55. Opción del Analizador de consultas en el menú de SQL Server 7.0.

Las cualidades de esta aplicación, unidas al Administrador corporativo, las convierten en las principales utilidades de administración de SQL Server 7.0. Es una herramienta generalmente más útil para el desarrollador de aplicaciones, pero hemos de tener en cuenta, que a nivel administrativo, la mayoría de las operaciones pueden ser realizadas tanto por el Administrador corporativo, como mediante un procedimiento almacenado proporcionado por el motor de datos, o una sentencia Transact-SQL, y para estos dos últimos, el medio de ejecutarlos sin tener que recurrir al desarrollo de una aplicación que lo haga, pasa por el uso del Analizador de consultas. Por este motivo, sin hacer un recorrido exhaustivo, nos detendremos sólo en los puntos de esta aplicación que puedan resultar más interesantes para el administrador del servidor.

Podemos emplear sentencias pertenecientes tanto al lenguaje de definición de datos (DDL, Data Definition Language), empleado para la creación de objetos: base de datos, tablas, etc., como al de manipulación (DML, Data Manipulation Language), empleado para consultar datos, añadir, modificar, borrar, etc., así como procedimientos almacenados tanto del sistema como definidos por el usuario.

Al iniciar esta herramienta, se nos solicita la información de conexión, esto es, servidor al que queremos conectarnos y tipo de autenticación: del sistema operativo o de SQL Server, como se muestra en la Figura 56.

Figura 56. Ventana de conexión al Analizador de consultas.

Después de proporcionar los valores adecuados, se mostrará la ventana correspondiente al Analizador de consultas. Esta aplicación se basa en una ventana principal o MDI que puede contener una o varias ventanas hijas de consulta. Ver Figura 57.

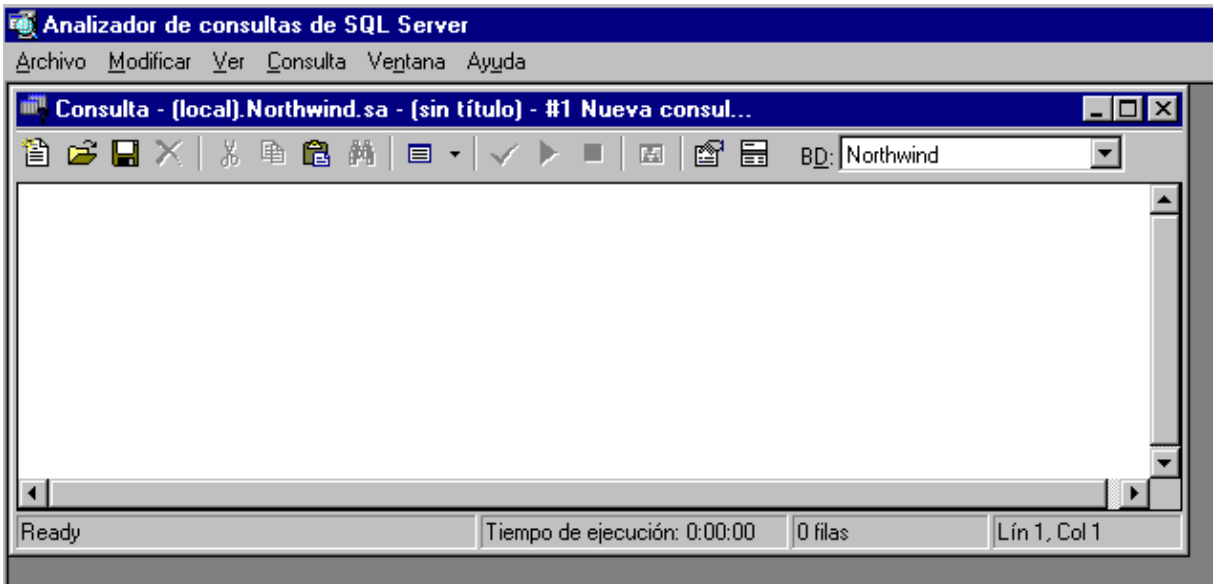


Figura 57. Analizador de consultas de SQL Server.

Cada ventana de consulta está dividida en un panel superior en el que se escriben las sentencias y en uno inferior donde se visualizan los resultados. Dispone de una barra de herramientas con las opciones de uso más frecuente. En primer lugar podemos destacar la lista desplegable BD, en la que seleccionaremos una de las bases de datos pertenecientes al servidor al que nos hemos conectado. La imagen anterior, muestra esta opción con la base de datos Northwind, lo que quiere decir que todas las sentencias emitidas se realizarán contra esta base de datos.

Después de escribir una consulta en el panel correspondiente, la ejecutaremos seleccionando la opción de menú *Consulta+Ejecutar*, que mostrará el resultado en el panel inferior. Esta misma operación podemos conseguirla también pulsando la tecla F5 o el botón correspondiente en la barra de herramientas. Para consultas complejas, podemos realizar una comprobación sintáctica mediante la

opción de menú *Consulta+Analizar*, que realiza una comprobación de las instrucciones que forman parte de la consulta sin ejecutarla. La Figura 58 muestra los botones de la barra de herramientas correspondientes a estas operaciones.

Figura 58. Botones de Analizar y Ejecutar consulta.

La Figura 59 muestra los resultados de la ejecución de una sentencia en el Analizador de consultas.

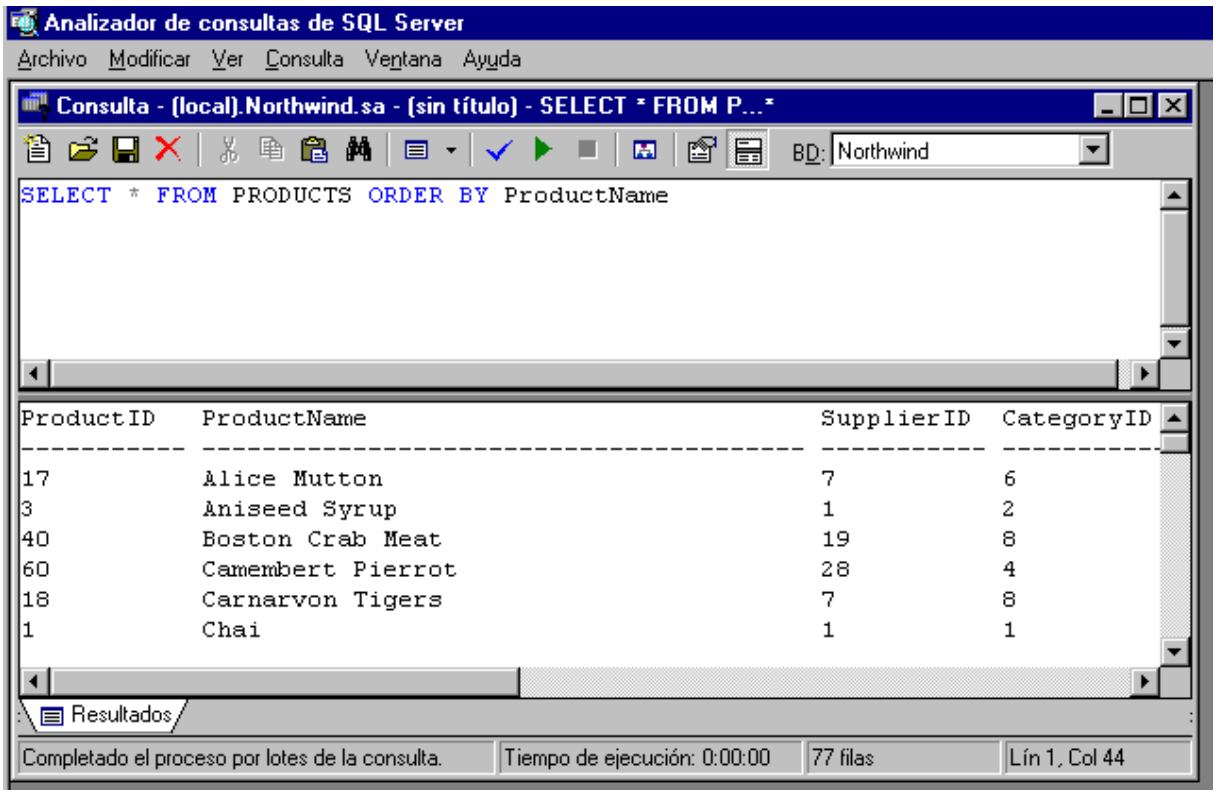


Figura 59. Resultados de la ejecución de una consulta.

Por defecto, el panel de resultados muestra la información en columnas de tipo texto. Si queremos visualizar los datos en forma de cuadrícula, seleccionaremos la opción de menú *Consulta+Resultados en la cuadrícula*, la combinación de teclado *Ctrl+D* o el icono de la barra de herramientas.

Podemos tener múltiples consultas en el panel y ejecutar sólo una seleccionando el texto que la compone.

Para guardar el contenido de cada panel, debemos situarnos sobre el mismo haciendo clic y seleccionando la opción de menú *Archivo+Guardar*, o bien la combinación de teclado *Ctrl+G* o el botón de la barra de herramientas. Las sentencias se grabarán en ficheros de texto con la extensión .SQL, y los resultados en ficheros delimitados por comas o por tabuladores con la extensión .CSV.

Otro detalle destacable de esta herramienta es la visualización de un plan de ejecución de consultas en el que se informa del gasto de recursos empleado en realizarla. La Figura 60 muestra una consulta en la que intervienen dos tablas de la base de datos Northwind junto a su plan de ejecución.

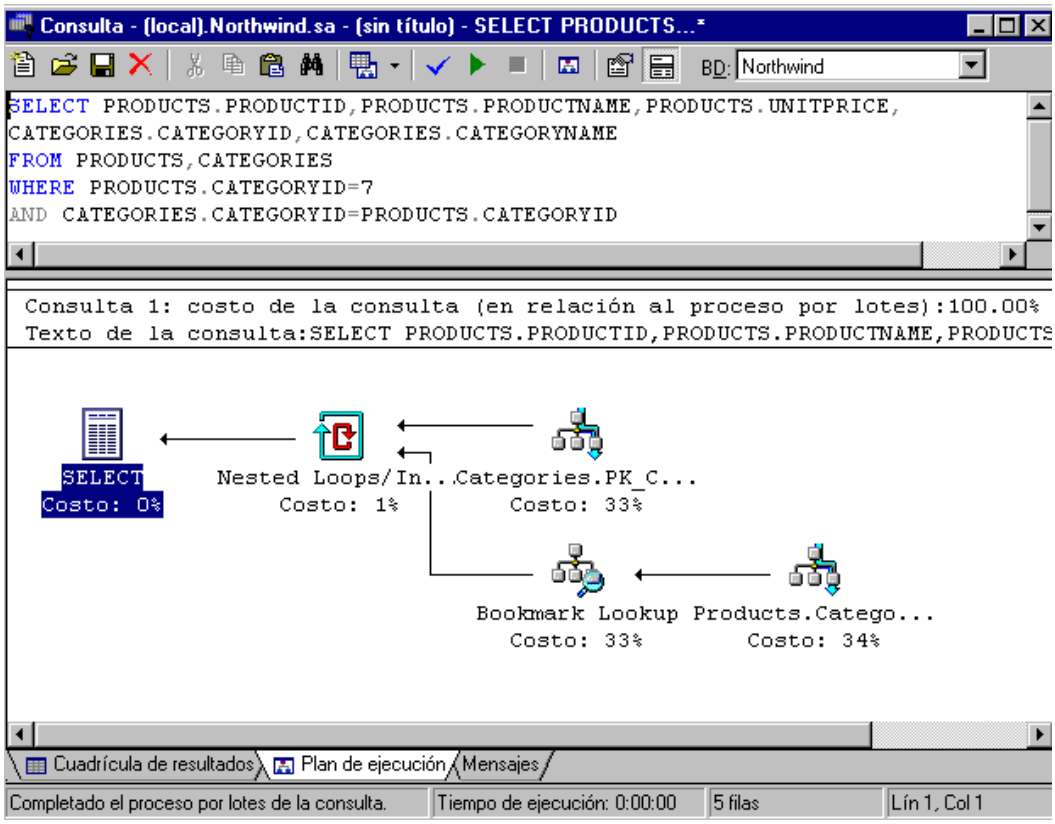


Figura 60. Plan de ejecución de una consulta.

Para ejecutar un procedimiento almacenado, debemos escribir la instrucción EXECUTE o EXEC, seguida del nombre del procedimiento almacenado y los valores para los correspondientes parámetros si tuviera. En la Figura 61, se muestra la ejecución de uno de los procedimientos almacenados de la base de datos Northwind junto a su resultado.

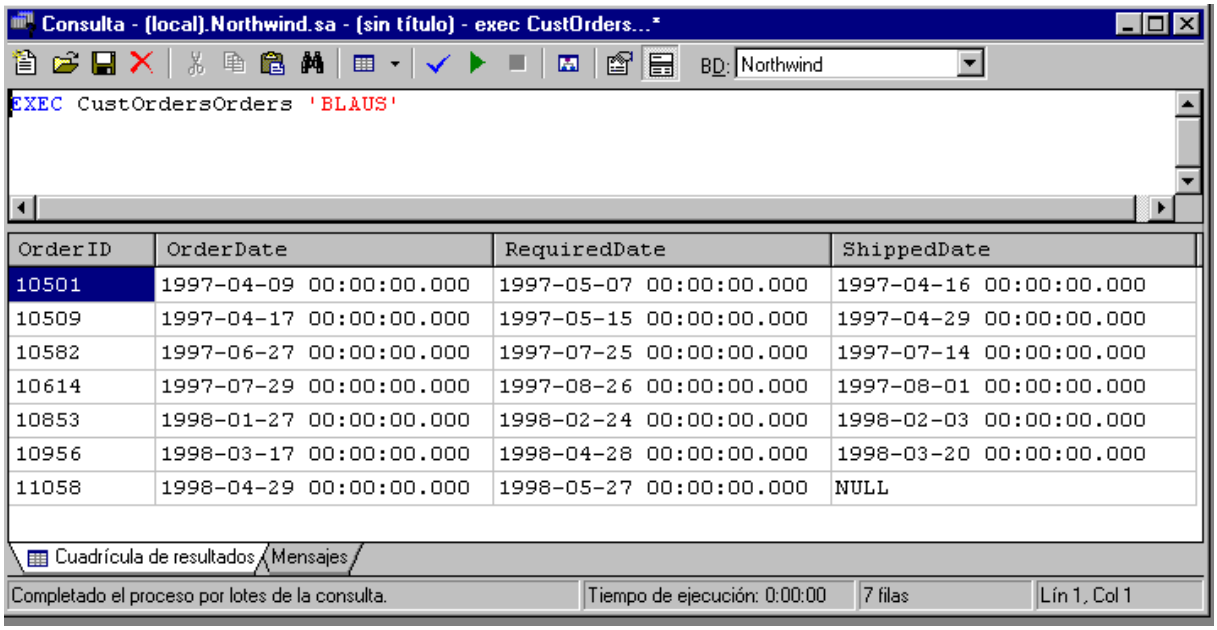


Figura 61. Ejecución de un procedimiento almacenado.

Generalmente, todas las operaciones en SQL Server pueden ejecutarse empleando tanto el Administrador corporativo, como una instrucción Transact-SQL o procedimiento almacenado propio de SQL Server (denominado procedimiento almacenado del sistema). Dependiendo de la situación en que nos encontremos, será más conveniente emplear uno u otro medio. En lo sucesivo, para las ocasiones en que tengamos que mostrar el modo de uso de una instrucción o procedimiento almacenado utilizaremos el Analizador de consultas.

8

Bases de datos. Creación y configuración

Creación de bases de datos

Esta operación crea una nueva base de datos vacía en el servidor sobre el que estemos situados. Podemos realizarla de las siguientes formas.

Administrador corporativo

Después de conectarnos a un servidor, haremos clic con el botón derecho en la carpeta *Bases de datos* y seleccionaremos la opción de menú *Nueva base de datos*, que abrirá una ventana en la que indicaremos las propiedades para la nueva base de datos, ver Figura 62.

En el campo Nombre de la pestaña General, deberemos indicar la denominación para la base de datos. Igualmente, aunque no obligatorio, podemos especificar el nombre del archivo que contendrá los datos, su ubicación en el disco y el tamaño inicial que tendrá dicho archivo. Por defecto, los ficheros de datos se almacenan en la ruta Unidad:\MSSQL7\DATA\, donde unidad es el disco en el que está instalado SQL Server.

El apartado *Propiedades del archivo* nos permite establecer si el archivo crecerá automáticamente, y si es así, el modo en que lo hará, en MB o porcentaje. También es posible limitar el tamaño del archivo o que aumente con la única restricción del espacio en disco disponible.

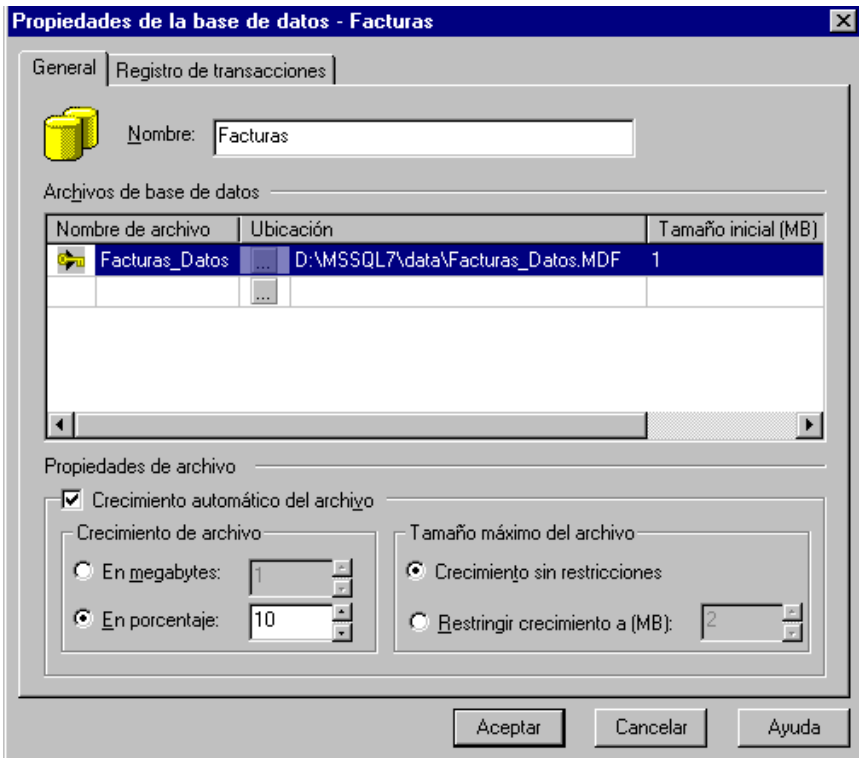


Figura 62. Ventana para la creación de una base de datos.

Daremos para este ejemplo el nombre Facturas a la base de datos, lo que nos creará un archivo con el nombre FACTURAS_DATOS.MDF.

En la pestaña *Registro de transacciones,* se encuentra la información sobre el archivo que se creará para guardar el registro de transacciones de la base de datos. Podemos establecer los mismos valores que para el archivo de datos. Si dejamos todo por defecto, se creará en la misma ruta un archivo con el nombre FACTURAS_REGISTRO.LDF.

Después de pulsar Aceptar, se creará la base de datos vacía. Si hacemos doble clic sobre su icono, el Administrador corporativo nos mostrará sus elementos, ver Figura 63.

Figura 63. Nueva base de datos en el Administrador corporativo.

Para comprobar que el tamaño de los ficheros físicos de datos creados, concuerda con el que hemos indicado en este proceso de creación, sólo debemos utilizar el *Explorador de Windows* y visualizar el contenido de los directorios en los que se han creado estos ficheros.

Instrucción CREATE DATABASE

Esta instrucción, empleada desde el Analizador de consultas, nos creará una base de datos de igual manera que la anterior, sólo que sin utilizar una interfaz gráfica. Su sintaxis es la siguiente.

```
CREATE DATABASE NombreBD
[ON
PRIMARY
[DetalleArch [,...DetalleArchN] ]
[GrupoArch [,...GrupoArchN] ] ]
[LOG ON DetalleArch [,...DetalleArchN] ]
```

Sintaxis para DetalleArch.

```
( [ NAME=NombreLógico, ]
FILENAME='NombreFísico'
[, SIZE=Tamaño ]
[, MAXSIZE={ TamañoMax|UNLIMITED } ]
[, FILEGROWTH=Incremento] )
```

- NombreBD. Nombre que utilizará SQL Server para identificar a la base de datos y mostrarla en el servidor.
- ON. Sirve para indicar a la instrucción que a continuación se especificarán los ficheros que deberán crearse. Si no utilizamos esta partícula ni el resto, SQL Server creará una base de datos basándose en el valor de NombreBD.
- PRIMARY. Especifica el grupo de archivos principal. En el caso de que no se use esta cláusula, el primer archivo de la lista se tomará como principal.
- DetalleArch. Contiene los siguientes valores empleados para especificar un archivo de la base de datos:
    - NAME. Nombre lógico del archivo.
    - FILENAME. Nombre físico del archivo.
    - SIZE. Tamaño del archivo, por defecto 1 MB.
    - MAXSIZE. Tamaño máximo de crecimiento para el archivo. Si utilizamos el valor UNLIMITED, el archivo crecerá hasta que el disco se llene.
    - FILEGROWTH. Valor de incremento para el archivo, que no podrá ser en ningún caso superior al de MAXSIZE.
- GrupoArch. Nombre del grupo de archivos al que se asociará la lista de archivos especificada. Los grupos de archivos serán tratados posteriormente.
- LOG ON. Permite especificar los archivos que almacenarán el registro de transacciones.

A continuación escribiremos la sentencia de la Figura 64, y la ejecutaremos en el Analizador de consultas, lo que nos creará una base de datos en una ruta distinta de la que asigna SQL Server por defecto.

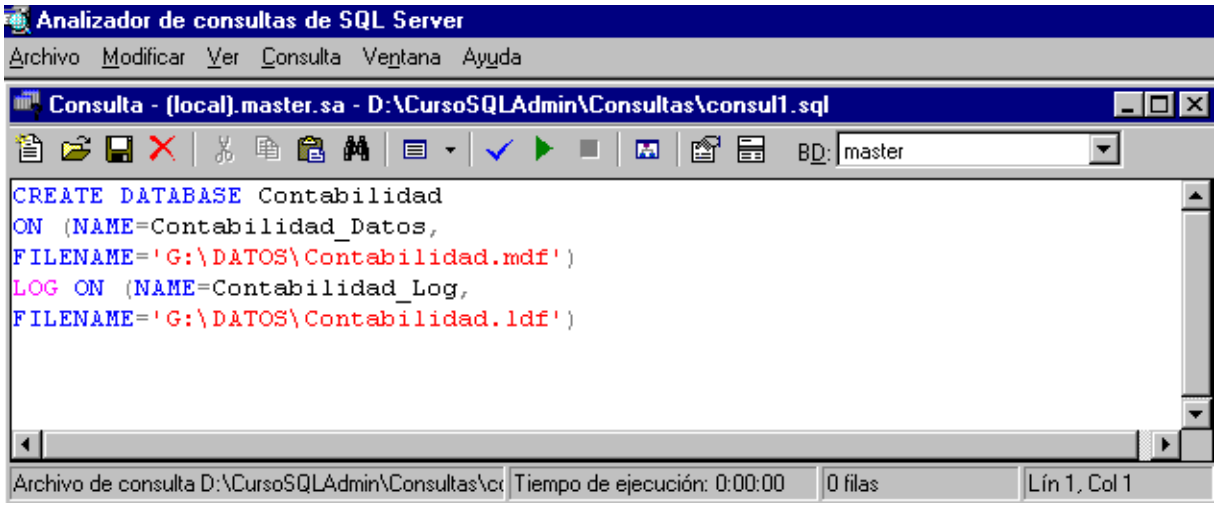


Figura 64. Instrucciones para crear una nueva base de datos.

Desde el Administrador corporativo, podemos acceder a las bases de datos existentes para comprobar que se ha creado correctamente la nueva base de datos mediante instrucciones.

En el caso de que tengamos abiertos ambos programas y no se haya refrescado la información en el Administrador corporativo, podemos seleccionar su opción de menú *Acción+Actualizar* para que se refresque su contenido y refleje los últimos cambios efectuados.

Propiedades de la base de datos

Una vez creada una base de datos, podemos acceder a sus propiedades para consultar o modificar valores de dos formas.

Administrador corporativo

Debemos hacer clic con el botón derecho sobre la nueva base de datos y seleccionar la opción del menú contextual Propiedades, que nos mostrará la misma ventana utilizada en la creación de la base de datos para establecer los ficheros, pero incorporando dos nuevas pestañas.

En la pestaña Opciones, entre otras, podemos establecer el modo de acceso y limitarlo sólo al propietario de la base de datos, que sólo pueda acceder un usuario al mismo tiempo, establecerla como sólo de lectura, cierre y reducción automática, creación de estadísticas, etc.

En la pestaña Permisos, indicaremos las operaciones que podrá realizar el usuario o función de la base de datos.

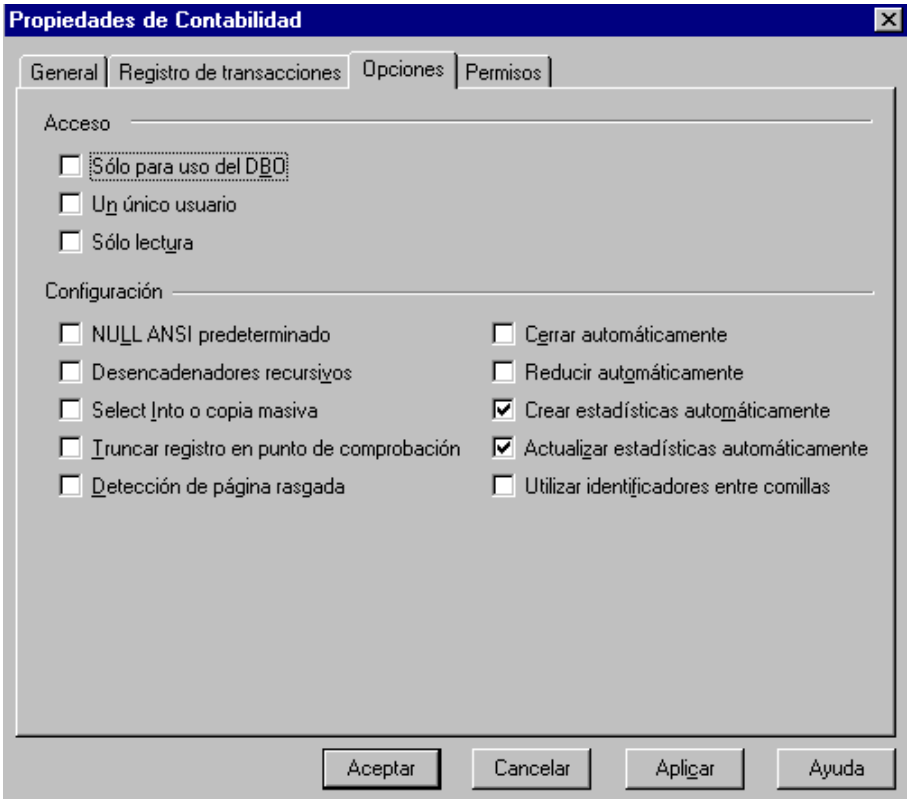


Figura 65. Propiedades de la base de datos.

Procedimientos almacenados

Otra forma de acceder a las propiedades de una base de datos pasa por utilizar alguno de los procedimientos almacenados del sistema de SQL Server, que nos devuelven o modifican la información de las propiedades de una base de datos, y que se relacionan a continuación.

sp_helpdb

Utilizado individualmente, proporciona información sobre las propiedades de todas las bases de datos del servidor actual. Podemos pasarle como parámetro el nombre de una base de datos, con lo que la información devuelta corresponderá sólo a dicha base de datos.

```
sp_helpdb [[@dbname=] 'NombreBD']
```

- NombreBD. Nombre de la base de datos de la que vamos a obtener información.

El Código fuente 1 muestra la forma de obtener información sobre la base de datos Contabilidad, creada anteriormente.

```
EXEC sp_helpdb 'Contabilidad'
```

Código fuente 1

El resultado se muestra en la Figura 66.

name	db_size	owner	dbid	created	status
Contabilidad	1.88 MB	sa	8	Jul 18...	no options set

Figura 66. Resultado de la ejecución de sp_helpdb.

sp_dboption

Muestra y permite cambiar las propiedades de la base de datos.

```
sp_dboption [[@dbname =] 'NombreBD'] [, [@optname =] 'NombreOpción']
[, [@optvalue =] 'ValorOpción']
```

- NombreBD. Cadena con el nombre de la base de datos. Si sólo pasamos este parámetro, se muestran las propiedades establecidas para la base de datos.
- Opcion. Cadena con la propiedad a modificar. Algunos de estos nombres son: autoshrink, dbo use only, recursive triggers, etc.
- Valor. Valor lógico. True establece la propiedad, False la deshabilita.

Sólo es posible establecer una propiedad a la vez. Por ejemplo, si queremos que la base de datos Contabilidad se reduzca automáticamente, ejecutaremos el Código fuente 2.

```
sp_dboption 'Contabilidad', 'autoshrink', 'true'
```

Código fuente 2

El Código fuente 3 usa este procedimiento almacenado para visualizar las propiedades de la base de datos, que se muestran en la Figura 67.

```
sp_dboption 'Contabilidad'
```

Código fuente 3

The following options are set:
autoshrink
auto create statistics
auto update statistics

Figura 67. Resultado de la ejecución de sp_dboption.

El registro de transacciones

Una transacción sobre una base de datos, consiste en un conjunto de operaciones, que realizan algún tipo de alteración sobre la información que dicha base de datos contiene (agregar, modificar, eliminar registros), pero que deben ser llevadas a cabo de forma unitaria, de forma que si alguno de los pasos

falla, se descarten todas las modificaciones llevadas a cabo hasta el momento, dejando la base de datos en el estado original en que se encontraba antes de comenzar la transacción.

El Registro de transacciones (Transaction log), es un elemento de suma importancia. Cada base de datos en SQL Server dispone de un registro de transacciones, que permite mantener la integridad de la base de datos si se produce un fallo del sistema durante un proceso que realice modificaciones en la información.

Cada vez que se envía una modificación a una base de datos, el registro de transacciones genera una nueva transacción. Dicha transacción es situada en la memoria caché para reducir el número de operaciones a realizar en el disco y agilizar el rendimiento. De igual forma, las páginas de datos implicadas en la modificación se sitúan también en la caché.

A continuación se apuntan en el Transaction log las operaciones realizadas, de forma que se pueda devolver la base de datos a su estado original en caso de fallo en la operación. Seguidamente se realizan las modificaciones sobre la base de datos, y finalmente se confirma la transacción, proceso que vuelca la información de dicha transacción en el registro de transacciones del disco, como muestra la Figura 68.

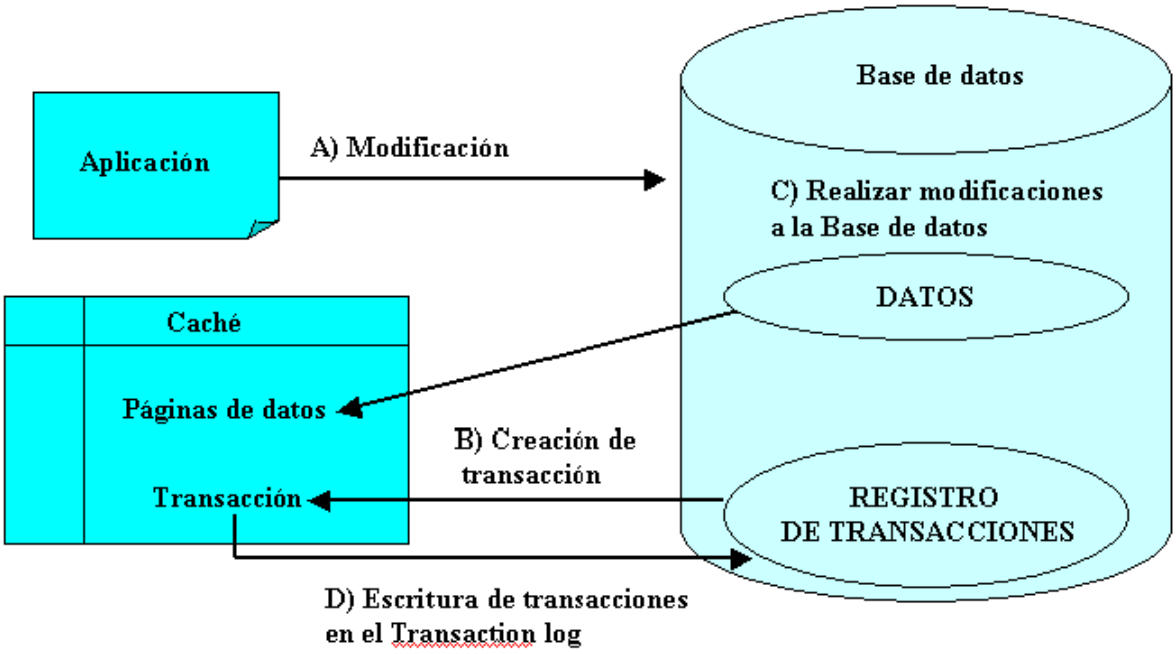


Figura 68. Esquema de trabajo en un proceso de transacción.

Se debe destacar el diferente manejo de la caché en esta situación, ya que mientras que al confirmarse la transacción, las páginas que alberga dicha transacción se pasan inmediatamente a disco, no ocurre lo mismo con las páginas de datos, que serán grabadas a disco cuando la caché se llene o cuando se ejecute un punto de comprobación (checkpoint).

Debido al intenso trabajo con la caché que SQL Server realiza en todas las operaciones que impliquen modificación de datos, es muy recomendable comprobar si el disco duro dispone de un controlador de caché, y en caso afirmativo deshabilitarlo, ya que podría influir en la integridad de la base de datos. No será necesario desconectar este controlador si está especialmente preparado para la base de datos.

En función de como sea ejecutada, podemos describir los siguientes tipos de transacciones:

- Transacciones implícitas. Tienen lugar sin que el usuario o aplicación que maneja la base de datos indique que sean ejecutadas, en cuyo caso, será el motor de datos el que se encargue internamente de crearlas y gestionarlas.

- Transacciones explícitas. La aplicación de datos o el usuario deben de crear previamente la transacción para que pueda llevarse a cabo. Para ello disponemos de las siguientes instrucciones:

    - BEGIN TRANSACTION. Indica el comienzo de una transacción.

    - COMMIT TRANSACTION. Finaliza la transacción, confirmando las operaciones realizadas para escribirlas en el registro de transacciones.

    - ROLLBACK TRANSACTION. Cancela una transacción en curso, deshaciendo los cambios efectuados hasta el momento.

El Código fuente 4 nos muestra un ejemplo de transacción explícita que realiza una modificación sobre una tabla.

```
BEGIN TRANSACTION
UPDATE Cuentas
SET Nombre='Entidades crédito'
WHERE IDCuenta=2
COMMIT TRANSACTION
```

Código fuente 4

Puntos de comprobación (Checkpoints)

Acabamos de ver en el apartado sobre transacciones, como se mantienen en caché durante un tiempo antes de escribirse en disco, para optimizar recursos.

Debido a esta situación, es necesario un mecanismo que proceda a realizar dicha escritura en algún momento. Dicho mecanismo se denomina punto de comprobación o checkpoint.

Cuando SQL Server encuentra un punto de comprobación, escribe en la base de datos las páginas modificadas desde que se realizó la última escritura en disco hasta el momento.

SQL Server establece automáticamente checkpoints de forma implícita, en intervalos de tiempo calculados por el motor de datos. Si queremos establecer un punto de comprobación explícito, debemos utilizar la instrucción CHECKPOINT.

Grupos de archivos

Una base de datos puede estar distribuida en varios archivos. Si creamos una base de datos que va a tener un gran tamaño y nuestro equipo dispone de varios discos duros, podemos crear archivos de bases de datos y agruparlos en varios discos, de forma que se soporte mejor la carga de trabajo entre todos los componentes físicos del sistema.

Otra situación frecuente, consiste en tener una base de datos en la que existen tablas que son muy consultadas y poco modificadas, y la situación inversa, tablas poco consultadas pero sobre las que se realiza un gran número de actualizaciones, inserciones, borrados, etc.

Para los casos que acabamos de comentar, los grupos de archivos es una solución que nos permite tener en un grupo las tablas con un gran número de consultas, y en otro grupo, las tablas que van a ser más modificadas.

Veamos las formas de crear grupos de archivos basándonos en el siguiente ejemplo: tenemos o necesitamos crear una base de datos de contabilidad, en la que debemos definir un grupo de archivos que tendrá las tablas de cuentas que son poco actualizadas y otro grupo para las tablas de apuntes que se modifican más a menudo.

Administrador corporativo

En la ventana de propiedades de la base de datos, ya sea durante su creación o posteriormente, en el apartado *Archivos de bases de datos*, podemos poner debajo del archivo principal, todos los demás archivos adicionales que necesitemos y especificar el nombre del grupo en la columna *Grupo de archivos*, tal y como muestra la Figura 69.

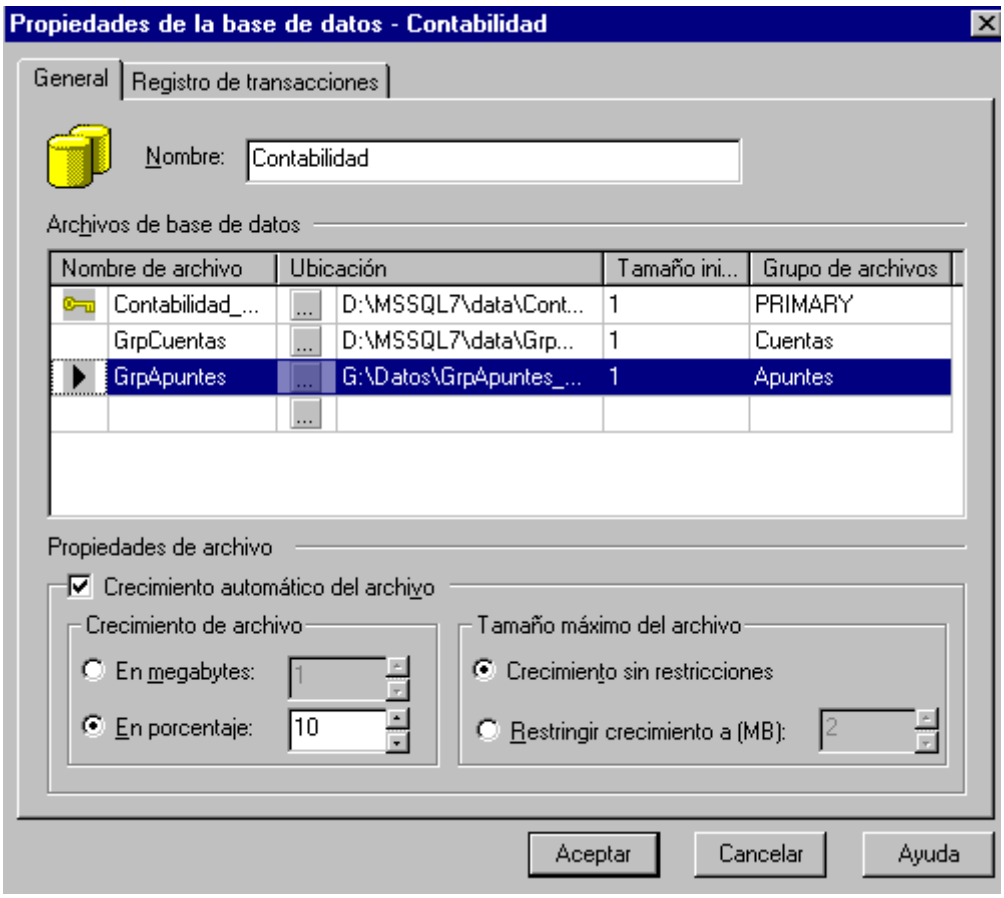


Figura 69. Creación de grupos de archivos desde el Administrador corporativo.

Instrucción CREATE DATABASE

La instrucción CREATE DATABASE, explicada anteriormente, la combinaremos junto a la palabra clave FILEGROUP, que será la que nos permita indicar el grupo de archivos a crear, como vemos en el Código fuente 5.

```
CREATE DATABASE Contabilidad

ON PRIMARY (NAME=Contabilidad_Datos,
FILENAME='D:\MSSQL7\DATA\Contabilidad.mdf'),

FILEGROUP Cuentas
(NAME=GrpCuentas,
FILENAME='D:\MSSQL7\DATA\GrpCta.ndf'),

FILEGROUP Apuntes
(NAME=GrpApuntes,
FILENAME='G:\DATOS\GrpApte.ndf')

LOG ON (NAME=Contabilidad_Log,
FILENAME='D:\MSSQL7\DATA\Contabilidad.ldf')
```

Código fuente 5

Observe el lector, cómo hemos creado el grupo de archivos correspondiente a los apuntes en una unidad diferente al grupo de archivos para cuentas, punto que podemos comprobar desde la ventana de propiedades de la base de datos en el Administrador corporativo.

Instrucción ALTER DATABASE

Esta instrucción nos permitirá crear nuevos grupos de archivos y añadirlos a una base de datos existente. Operación que realizaremos en dos fases: en primer lugar crearemos el grupo en la base de datos, y a continuación, crearemos un archivo que adjuntaremos al grupo recién creado.

```
ALTER DATABASE NombreBD
ADD FILE DetalleArch [,...DetalleArchN] [TO FILEGROUP GrupoArch]
| ADD LOG FILE DetalleArch [,...DetalleArchN]
| REMOVE FILE NombreLógico
| ADD FILEGROUP GrupoArch
| REMOVE FILEGROUP GrupoArch
| MODIFY FILE DetalleArch
| MODIFY FILEGROUP GrupoArch PropiedadGrupo
```

El Código fuente 6, muestra un ejemplo en el que se crean dos grupos de ficheros para la base de datos Contabilidad, y se añade un fichero a cada uno.

```
ALTER DATABASE Contabilidad
ADD FILEGROUP GrpCuentas
GO

ALTER DATABASE Contabilidad
ADD FILE
(NAME=Cuentas,
```

```
FILENAME='D:\MSSQL7\DATA\GrpCuentas.ndf')
TO FILEGROUP GrpCuentas
GO

ALTER DATABASE Contabilidad
ADD FILEGROUP GrpApuntes
GO

ALTER DATABASE Contabilidad
ADD FILE
(NAME=Apuntes,
FILENAME='G:\DATOS\GrpApuntes.ndf')
TO FILEGROUP GrpApuntes
GO
```

Código fuente 6

Aumentar el tamaño de una base de datos

Puede darse el caso de que una base de datos para la que en un principio se había estimado un tamaño medio, comience a crecer en mayor medida de lo esperado. En estos casos debemos modificar las propiedades relacionadas con el tamaño de forma que no se produzca un error por tener limitada su capacidad.

Esta ampliación de tamaño puede aplicarse a los ficheros de datos y a los del registro de transacciones.

Administrador corporativo

Desde esta herramienta, seleccionaremos la base de datos a modificar y accederemos a su ventana de propiedades. En el apartado dedicado a los archivos de la base de datos, seleccionaremos el archivo al que modificaremos su tamaño, y en la columna *Espacio asignado (MB),* introduciremos el nuevo tamaño. Aplicaremos los cambios, y desde el Explorador de Windows podremos comprobar como SQL Server ha modificado el tamaño de los archivos correspondientes. Ver Figura 70.

También es posible configurar los parámetros de crecimiento para cada uno de los archivos, simplemente seleccionando el archivo y estableciendo las propiedades del apartado *Propiedades del archivo* en esta ventana. Por ejemplo, si estimamos que el archivo que contiene la tabla de apuntes va a sufrir un gran incremento, marcamos la propiedad *Crecimiento sin restricciones* para indicar que ese archivo no tendrá límite en cuanto a su crecimiento.

Siempre que variemos el tamaño de la base de datos de esta forma, será para aumentarlo, ya que si intentamos establecer un tamaño menor que el existente hasta el momento se producirá un error. La reducción de una base de datos será tratada en un tema posterior.

Para el registro de transacciones daremos los mismos pasos, pero al abrir la ventana de propiedades de la base de datos, seleccionaremos la pestaña *Registro de transacciones*, modificando el tamaño del archivo existente de transacciones o añadiendo uno nuevo, tal y como muestra la Figura 71.

Como acaba el lector de comprobar, sobre la base de datos Contabilidad se ha ampliado el registro de transacciones existente Contabilidad_log a 4 MB y se ha añadido un nuevo archivo para dicho registro con el nombre Conta2_log, asignándole un tamaño de 2 MB. Al igual que en las operaciones de creación de archivos, podemos comprobar los nuevos tamaños de los ficheros desde el *Explorador de Windows*.

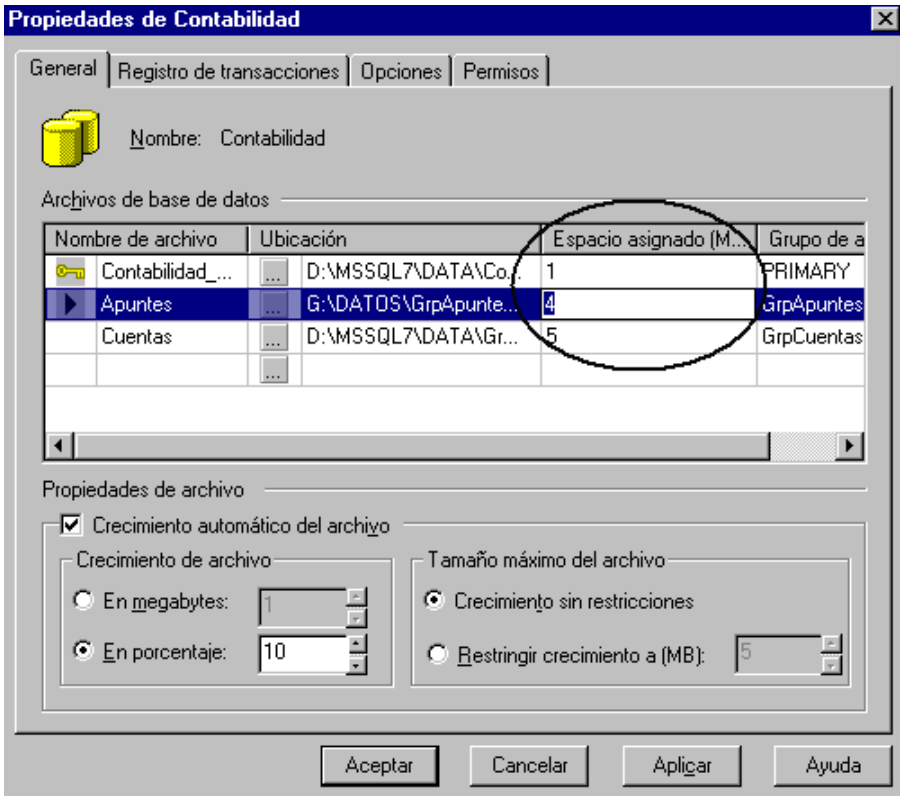


Figura 70. Modificación del tamaño de una base de datos desde el Administrador corporativo.

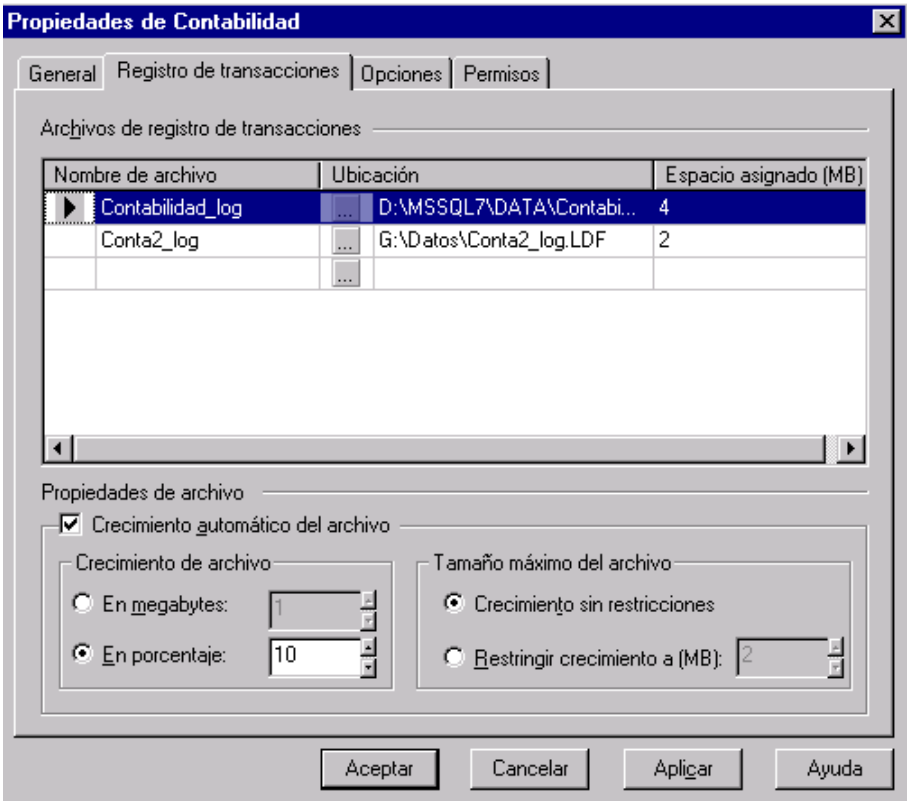


Figura 71. Modificación del tamaño de un registro de transacciones desde el Administrador corporativo.

Instrucción ALTER DATABASE

Esta instrucción, descrita en un apartado anterior, nos servirá en esta ocasión para aumentar el tamaño de los elementos de una base de datos existente, utilizando las palabras clave adicionales descritas en los siguientes puntos.

- MODIFY FILE. Emplearemos esta cláusula cuando modifiquemos un fichero integrante de una base de datos. Sólo podemos cambiar uno de sus aspectos a la vez (SIZE, FILEGROWTH, etc.)

- MODIFY FILEGROUP. Lo utilizaremos junto al nombre del grupo a modificar. Podemos emplear las siguientes palabras clave.

    - READONLY. Establece el grupo de archivos como sólo lectura.

    - READWRITE. Permite la lectura y escritura en el grupo de archivos.

    - DEFAULT. Establece el grupo de archivos como el predeterminado de la base de datos

Como ejemplo, en el Código fuente 7 aumentaremos el tamaño del archivo principal de la base de datos Contabilidad.

```
ALTER DATABASE Contabilidad
MODIFY FILE
(NAME='Contabilidad_Datos',
SIZE=4MB)
```

Código fuente 7

En cuanto al registro de transacciones, hemos de realizar una previsión en el momento de su creación, ya que si se queda sin espacio, al no poder registrar transacciones, no permitirá efectuar cambios sobre la base de datos.

Debemos tener en cuenta que todas las operaciones que modifiquen registros, suponen una carga importante de trabajo para la base de datos. Adicionalmente, si usamos tablas con índices, esta carga se incrementa, al tener que realizar también escrituras para los índices.

En el caso de que el tamaño del registro de transacciones se vea limitado, podemos utilizar también ALTER DATABASE para ampliarlo, como vemos en el Código fuente 8.

```
ALTER DATABASE Contabilidad
ADD LOG FILE
(NAME='Contab3_Reg',
FILENAME='G:\DATOS\Conta3.ldf',
SIZE=2MB)
```

Código fuente 8

Disminuir el tamaño de una base de datos

Es posible también, encontrarse en la situación inversa a la explicada en el tema anterior: disponer de una base de datos infrautilizada, diseñada para contener un gran número de datos, pero que finalmente no contiene todo el volumen de información esperado. En este caso, debemos reducir su tamaño para que los recursos de espacio puedan emplearse mejor.

Administrador corporativo

Una vez seleccionada la base de datos a reducir, elegiremos la ruta de menú *Acción+Todas las tareas+Reducir base de datos*, que nos mostrará un cuadro de diálogo en el que podremos indicar cómo realizar la reducción: moviendo páginas de datos y/o reduciendo archivos físicos. También es posible establecer que esta operación se realice de forma automática.

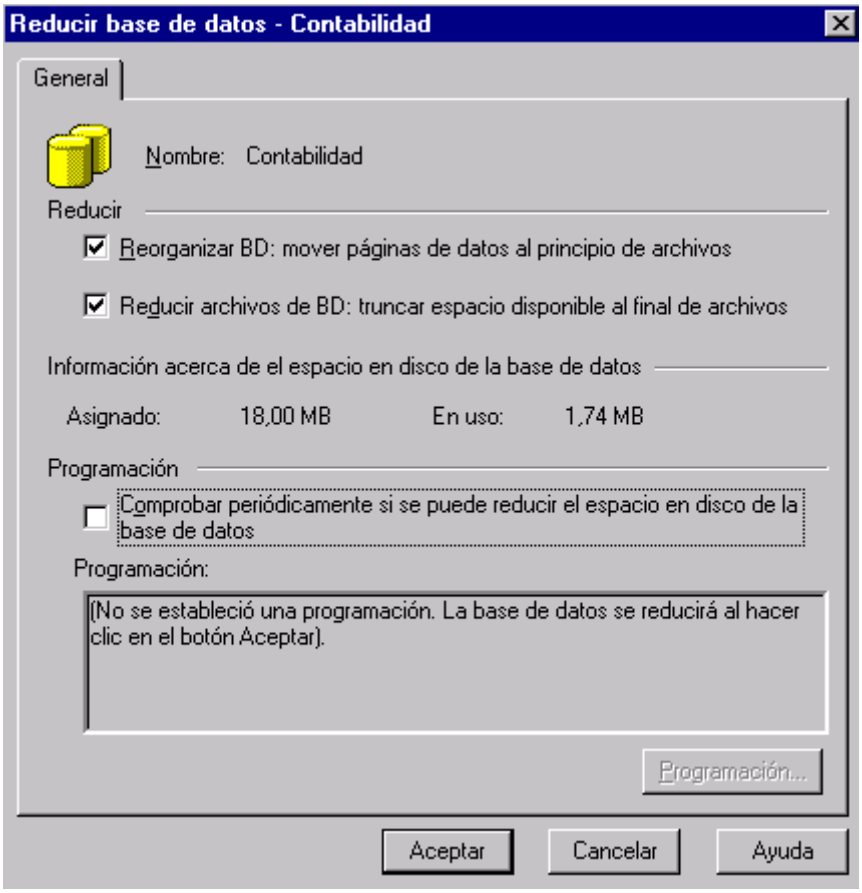


Figura 72. Reducción del tamaño de una base de datos mediante el Administrador corporativo.

DBCC (Database Consistency Checker)

Las instrucciones de comprobación de consistencia sobre la base de datos (DBCC), verifican la coherencia física y lógica de la información, solucionando incluso, los problemas localizados.

En este caso, utilizaremos la instrucción DBCC SHRINKDATABASE y DBCC SHRINKFILE, para reducir una base de datos al completo, o uno de los archivos que la componen.

```
DBCC SHRINKDATABASE
( NombreBD [, PorcentajeLibre]
[, NOTRUNCATE | TRUNCATEONLY])
```

- NombreBD. Nombre de la base de datos a reducir.
- PorcentajeLibre. Porcentaje libre que quedará al efectuar la reducción.
- NOTRUNCATE. El espacio liberado se mantiene en los archivos de la base de datos. Por defecto, el espacio libre se transfiere al sistema.
- TRUNCATEONLY. Se libera el espacio no usado, pasándolo al sistema operativo. No se mueven los datos, ni se reasignan las páginas.

Supongamos que la base de datos Contabilidad tiene en su archivo principal un tamaño de 4 MB, y en un archivo secundario, 5 MB. Al ejecutar el Código fuente 9 desde el Analizador de consultas, el archivo principal se reduce a 3 MB, y el secundario a 1 MB.

```
DBCC SHRINKDATABASE
(Contabilidad,70)
```

Código fuente 9

El panel de resultados del Analizador de consultas, nos proporcionará la información de la Tabla 3 tras ejecutar esta instrucción.

DbId	FileId	CurrentSize	MinimumSize	UsedPages	EstimatedPages
7	1	352	96	96	96
7	4	128	128	0	0

Tabla 3. Información resultante de la ejecución de DIC SHRINKDATABASE.

Si sólo queremos realizar una reducción específica sobre un archivo de la base de datos, utilizaremos SHRINKFILE.

```
DBCC SHRINKFILE
( NombreBD | IdArchivo
[, TamañoRes ]
[, EMPTYFILE | NOTRUNCATE | TRUNCATEONLY ] )
```

- NombreBD. Nombre de la base de datos a reducir.
- IdArchivo. Número correspondiente al identificador del archivo a reducir.
- TamañoRes. Valor expresado en MB, que indica el tamaño del archivo a obtener después de la reducción. Si no se expresa, se intenta reducir tanto como sea posible.

- EMPTYFILE. Vacía el archivo, distribuyendo su contenido entre los demás archivos de la base de datos.
- NOTRUNCATE. El espacio liberado se mantiene en los archivos de la base de datos. Por defecto, el espacio libre se transfiere al sistema.
- TRUNCATEONLY. Se libera el espacio no usado, pasándolo al sistema operativo. No se mueven los datos, ni se reasignan las páginas.

Supongamos que en la base de datos Contabilidad, tenemos un archivo Apuntes, con un tamaño de 5 MB. La instrucción del Código fuente 10, lo reduciría a 2 MB.

```
DBCC SHRINKFILE
(Apuntes,2)
```

Código fuente 10

Después de su ejecución, la instrucción nos devolverá un resultado de la operación similar al del ejemplo anterior.

Borrar una base de datos

Para eliminar una base de datos que no vamos a utilizar, y recuperar de esta manera espacio para el sistema, podemos utilizar los siguientes medios.

Administrador corporativo

Seleccionaremos la base de datos, borrándola con la opción de menú *Acción+Eliminar*.

Instrucción DROP DATABASE

```
DROP DATABASE NombreBD [ ,...NombreBDN ]
```

La ventaja de utilizar esta instrucción frente al Administrador corporativo, es que con la primera podemos eliminar varias bases de datos, mientras que este último sólo nos permite eliminar una base de datos a la vez.

El ejemplo del Código fuente 11, elimina la base de datos Contabilidad de SQL Server.

```
DROP DATABASE Contabilidad
```

Código fuente 11

Si el lector experimenta problemas al intentar eliminar la base de datos, puede ser porque esté abierto el Administrador corporativo, pruebe a cerrarlo y abrirlo de nuevo, de esta forma se actualiza toda la información mostrada en esta herramienta.

En una operación de borrado de una base de datos, debemos tener en cuenta ciertas limitaciones. No será posible borrarla en los siguientes casos:

- Cuando esté en uso por un usuario.
- Al realizar operaciones de duplicación que afecten a la base de datos.
- Al realizar operaciones de restauración.

La base de datos del sistema msdb es posible eliminarla, aunque no recomendable en el caso de que vayamos a realizar operaciones de duplicación, transformación de datos, utilizar el Agente de SQL Server, o el asistente para Internet de SQL Server.

Analizar el tamaño necesario para una base de datos

A efectos de realizar una buena planificación, que suponga un adecuado consumo de recursos para el sistema, es muy recomendable realizar una estimación del tamaño a asignar a una nueva base de datos, así como la cantidad de información que contendrán las tablas, de manera que podamos hacer una previsión de los recursos empleados tanto por el sistema operativo como por el gestor de datos.

Tamaño de la base de datos

En este apartado debemos considerar en primer lugar, que al crear una base de datos, la información básica de creación se toma de la base de datos del sistema model. Por lo tanto, es necesario tener en cuenta el tamaño de dicha base de datos.

A continuación debemos calcular las previsiones de datos que contendrán las tablas, así como el número de índices y el tamaño de sus claves.

Seguiremos con el registro de transacciones, que deberá ser mayor cuanto más elevado sea el número de operaciones de modificación sobre los datos. Un valor adecuado suele estar entre el 10 y 25 por ciento del tamaño de la base de datos.

Finalmente y en menor importancia, consideraremos las tablas del sistema, aunque representan una pequeña parte del tamaño total de la base de datos.

Establecer número de datos para las tablas.

Para realizar una correcta estimación de almacenamiento, debemos realizar unos sencillos cálculos que nos permitan saber de modo aproximado, cuál será el tamaño de las tablas de la base de datos.

En primer lugar, sumaremos los bytes de cada columna de la tabla para obtener el total de bytes por fila. En el caso de que tengamos columnas de longitud variable, sumaremos el valor medio de la columna.

A continuación calcularemos el número de filas que admite cada página de datos, dividiendo 8060 entre el número total de bytes por fila. Si el resultado no es exacto, lo redondearemos.

Finalmente calcularemos el número de filas que pueda tener la tabla, dividiendo este número entre el número de filas que caben en cada página, obteniendo el total de páginas necesario para contener la tabla.

Por ejemplo, supongamos que tenemos una tabla llamada Clientes, con la estructura mostrada en la Tabla 4.

IDCliente	**Nombre**	**Ciudad**
int 4 bytes	char 25 bytes	char 30 bytes

Tabla 4. Estructura de la tabla Clientes.

Los cálculos a efectuar serían los mostrados por la Tabla 5.

Número de bytes por fila	59 bytes
Número de filas por página	136 filas
Número aproximado de filas de la tabla	5000 filas
Número total de páginas que necesita la tabla	37 páginas

Tabla 5. Estimación de tamaño para una tabla.

Rendimiento y seguridad

Implementar estos dos aspectos en un sistema gestor de datos puede resultar en muchos casos complicado, ya que, a veces, un buen rendimiento se consigue a costa de un menor nivel de seguridad y viceversa.

Obtener un buen rendimiento y seguridad en SQL Server, depende tanto del propio gestor de datos como del sistema operativo. A continuación se describen de forma breve, algunos aspectos a tener en cuenta para ambos.

SQL Server

Utilizar grupos de archivos para facilitar las copias de seguridad, de forma que coloquemos en determinados grupos, los archivos que tienen mayor nivel de manipulación, siendo estos grupos sobre los que realizaremos copias más frecuentemente.

Situar los ficheros de datos y del registro de transacciones en discos físicos diferentes. Esto evitará conflictos en las operaciones de actualización sobre la base de datos y escritura de su registro.

Windows NT

Sistema de discos en array (RAID). Consiste en instalar varios discos duros y distribuir la información entre ellos mediante diferentes técnicas o niveles que describimos a continuación.

- Nivel 0. La información se divide entre los discos, de forma que las operaciones sobre los datos quedan más repartidas. En el caso de fallo, no existe un medio de recuperación de los datos.

- Nivel 1. Se realiza una copia exacta de un disco sobre otro del conjunto, al que se denomina *disco espejo*, de forma que si el disco principal falla, el espejo toma su lugar. Este nivel proporciona una gran seguridad, pero paga un alto precio en cuanto a rendimiento, al tener que realizar todas las operaciones por duplicado

- Nivel 2. Emplea una técnica de distribución de datos igual que la del nivel 0, añadiendo corrección de errores. Este último factor reduce el rendimiento debido a su alta ocupación en los discos.

- Nivel 3. Usa el mismo sistema que el nivel 2, con mejoras de recuperación sobre la coherencia de datos.

- Nivel 4. Es igual que los dos niveles anteriores, con la diferencia de que los bloques de datos son mayores.

- Nivel 5. Consiste en una mejora sobre la técnica de *disco espejo* que proporciona mejores resultados que este.

9

Bases de datos. Manipulación

Introducción

En la manipulación de una base de datos al nivel administrativo, podemos incluir aquellas operaciones que se ocupan de crear, configurar y eliminar objetos de la base de datos como tablas, índices, procedimientos almacenados, etc. No entraremos en las tareas que se ocupan de los datos (altas, bajas, modificaciones), ya que aunque en ciertas circunstancias puedan ser competencias del administrador, en normas generales, son cometido de las aplicaciones que trabajan contra la base de datos.

Creación de tablas

Como ya se ha comentado en anteriores apartados, una tabla es el elemento de la base de datos encargado de almacenar la información. A continuación se describen los modos de creación de una tabla.

Administrador corporativo.

Debemos situarnos sobre la base de datos y expandir todos sus elementos. Después haremos clic sobre Tablas y seleccionaremos la opción de menú *Acción+Nueva tabla*. En primer lugar, tendremos que asignar un nombre a la tabla, pasando acto seguido a una ventana en la que indicaremos la definición de los campos de la tabla, como podemos ver en la Figura 73.

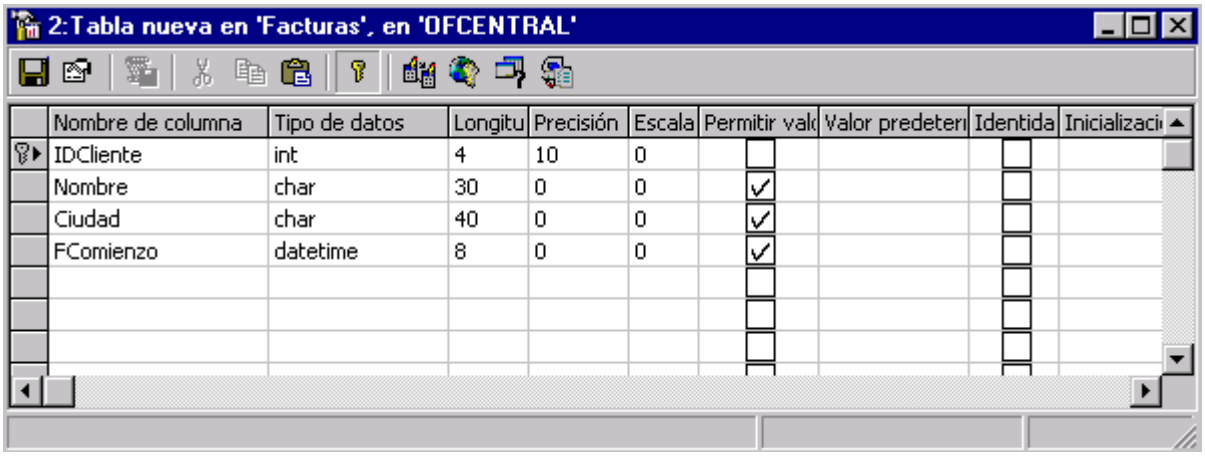


Figura 73. Ventana para la creación de una tabla.

En este caso, vamos a crear la tabla Clientes para una base de datos que llevará el control de las facturas de una empresa.

Al pulsar el botón con la imagen de una llave, en la barra de herramientas, el campo(s) en que estemos situados se convertirán en la clave primaria de la tabla (la columna *Permitir valores nulos* debe estar desmarcada).

Después de introducir todos los campos y sus configuraciones, haremos clic en el botón Guardar de la barra de herramientas, que tiene la imagen un disquete, finalizando el diseño de la tabla.

Instrucción CREATE TABLE

```
CREATE TABLE NombreTabla
( NombreCol Tipo [ CONSTRAINT Restricción ]
[ , ... NombreColN ] )
```

- NombreTabla. Nombre de la tabla.
- NombreCol. Nombre de la columna.
- Tipo. Tipo de datos para la columna
- CONSTRAINT. Indica el comienzo de una restricción
- Restricción. Nombre de una restricción. Se emplea para preservar la integridad de los datos de la tabla, y contiene una o varias palabras clave para definir esta integridad, como: PRIMARY KEY, NOT NULL, CHECK, etc.

Siguiendo con la creación de tablas para la base de datos Facturas, que estamos empleado como ejemplo, podemos generarlas desde código utilizando esta instrucción. En el Código fuente 12, añadiremos una tabla para las cabeceras de facturas en la base de datos.

```
CREATE TABLE Cabeceras
(IDFactura INT PRIMARY KEY,
IDCliente INT,
```

```
Fecha DATETIME,
Notas CHAR(50))
```

Código fuente 12

Creación de índices

Un índice es un objeto de la base de datos que se ocupa de ordenar los datos según un criterio basado en una o varias columnas. Internamente, consiste en un fichero que guarda ordenadas las posiciones de los registros de la tabla, de forma que al realizar una consulta ordenada basándose en dicho índice, se reduzca el tiempo de espera para obtener el resultado. Al igual que cuando se realiza una consulta a un libro, se busca el tema a consultar en el índice, para poder posicionarnos en la página adecuada, sin tener que comenzar la búsqueda desde el principio del libro, los índices de las tablas también nos posicionan directamente en el registro buscado, de forma que no tengamos que recorrernos la tabla desde el principio hasta localizar dicho registro.

Estructura interna

Según el diseño interno y su algoritmo de ordenación, el sistema de índices es más eficiente en unas bases de datos que en otras. La Figura 74 muestra un esquema genérico de la disposición de un índice.

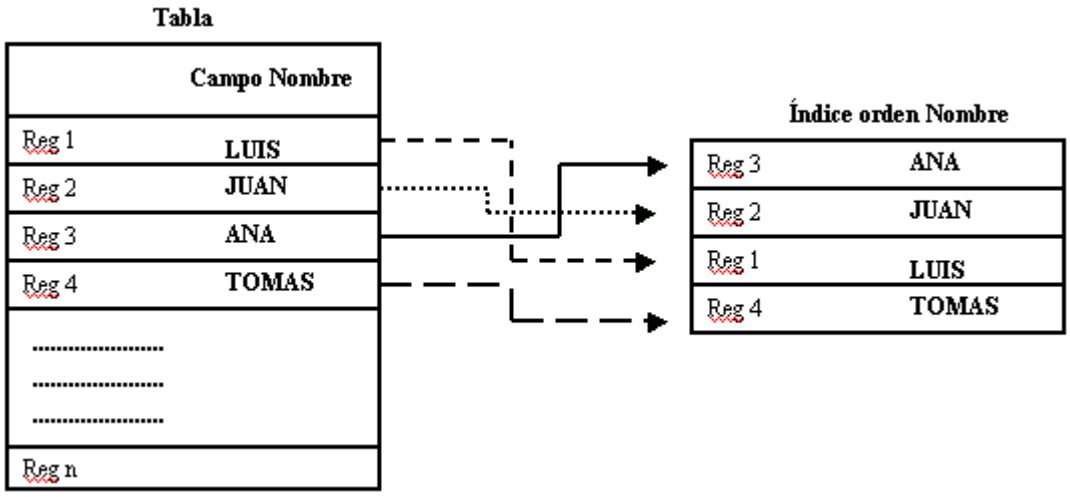


Figura 74. Esquema de ordenación interna de un índice.

SQL Server, debido a su sistema de almacenamiento basado en páginas, para lograr un mayor rendimiento, guarda los índices en páginas de índices separadas de las páginas normales, dentro de la base de datos.

Existen ciertos índices que SQL Server crea de forma implícita, sin que tengamos que indicarlo. Este es el caso de las columnas que se definen como clave primaria (PRIMARY KEY) o las que se establecen sin duplicados (UNIQUE).

Funcionamiento

Cuando ejecutamos una consulta sobre una tabla sobre la base del valor de un campo, si el motor de datos detecta que para ese campo por el que se realiza la búsqueda existe un índice definido, lo emplea para mejorar el rendimiento de dicha consulta. En el caso de que no haya un índice para el campo, comienza la búsqueda por el primer registro, devolviendo las coincidencias encontradas.

Inconvenientes y consideraciones

Los índices proporcionan enormes ventajas a la hora de buscar datos, pero también tienen un inconveniente, ya que necesitan espacio en la base de datos para almacenar las referencias ordenadas de los registros. Al realizar modificaciones sobre la base de datos, cuantos más índices existan, mayor número de escrituras se deberán hacer para mantener los índices actualizados, con el consiguiente retraso en la velocidad de tales operaciones.

Por este motivo, como administradores, a la hora de crear índices para una base de datos, debemos de analizar cuales van a ser los campos susceptibles de un mayor número de consultas, y crear los índices sólo por dichos campos, para que el rendimiento de la base de datos no se vea afectado.

Tipos de índice

Según el modo en que afecten a la ordenación física de la tabla, SQL Server distingue entre dos clases de índice.

- Unclustered. En este tipo de índice, la ordenación de las filas de la tabla se basa en el modo tradicional. Las filas se van añadiendo al final de la tabla, mientras que el índice es el encargado de mantener un conjunto de punteros ordenados a las filas.

- Clustered. Los índices clustered reordenan físicamente las filas de la tabla. Esto quiere decir, que cada vez que se añade un nuevo registro, no se sitúa forzosamente al final de la tabla, sino que se ubica según el orden definido por el índice, lo que proporciona una mayor optimización a la hora de las consultas. Debido a la naturaleza de estos índices, sólo es posible disponer de un índice clustered por tabla.

Factor de relleno

Esta característica de los índices de SQL Server se basa en que al crear un índice, se reserva un espacio en cada página que lo va a contener. Al realizar una previsión sobre la futura ocupación de datos del índice, se disminuyen las posibles divisiones que hubiera que realizar en la página.

Creación con el Administrador corporativo

Como todas las labores llevadas a cabo con esta herramienta, la creación de índices es una tarea sencilla. Abriremos la base de datos Facturas y haremos clic con el botón derecho del ratón sobre la tabla Cabeceras, seleccionando la opción *Administrar índices* del menú contextual, que nos presentará la ventana de administración de índices de la Figura 75.

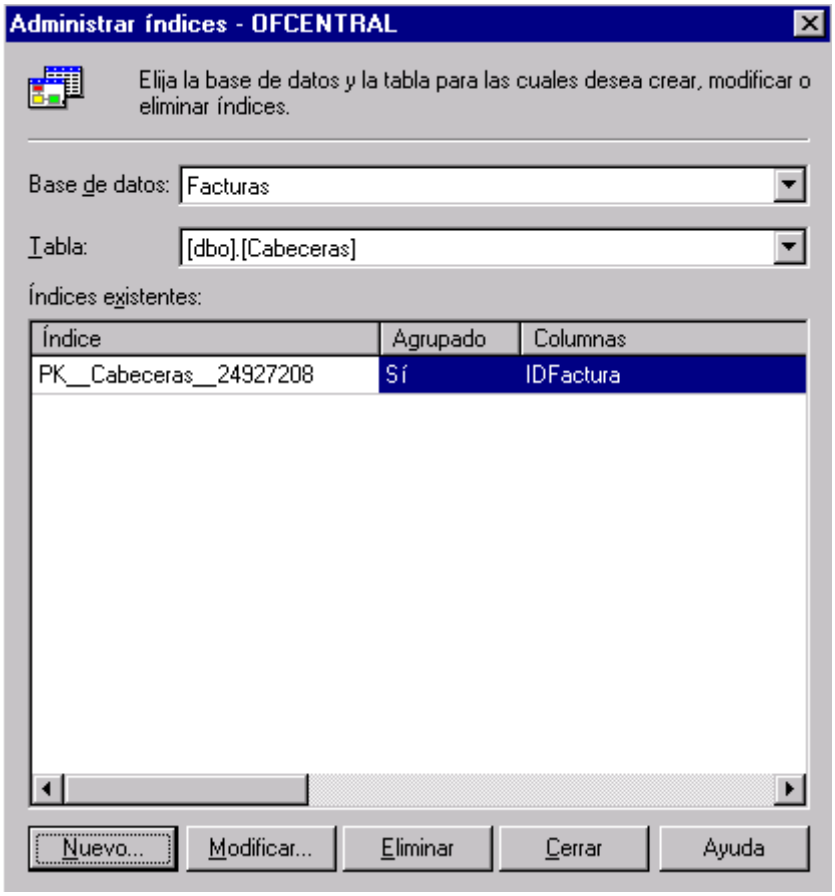


Figura 75. Ventana de administración de índices.

Dispone de una lista desplegable, llamada *Base de datos*, que permite seleccionar la base de datos sobre la que vamos a manipular los índices. De forma predeterminada, muestra la base de datos en la que estamos posicionados actualmente.

De igual forma tiene la lista Tabla, en la que elegiremos la tabla de la base de datos para la que crearemos o modificaremos el índice. Seleccionados ambos elementos, haremos clic en Nuevo, que abrirá la ventana de creación de índices mostrada en la Figura 76.

En este paso, escribiremos el nombre que vamos a dar al índice y elegiremos el campo en el que estará basado, además de otras propiedades adicionales, como si estará basado en valores únicos, el factor de relleno, grupo de archivos en el que se incluirá, etc.

En el caso de necesitar un índice compuesto por varias columnas, seleccionaremos aquellas que vayan a formar parte del índice y mediante los botones Subir y Bajar, la prioridad de las columnas en la creación del índice.

En nuestro caso, vamos a crear un índice por el campo Fecha, empleando el mismo nombre para denominar al índice.

Proporcionada toda la información, aceptaremos esta ventana, y cerraremos también la primera, de forma que habremos creado un nuevo índice para la base de datos.

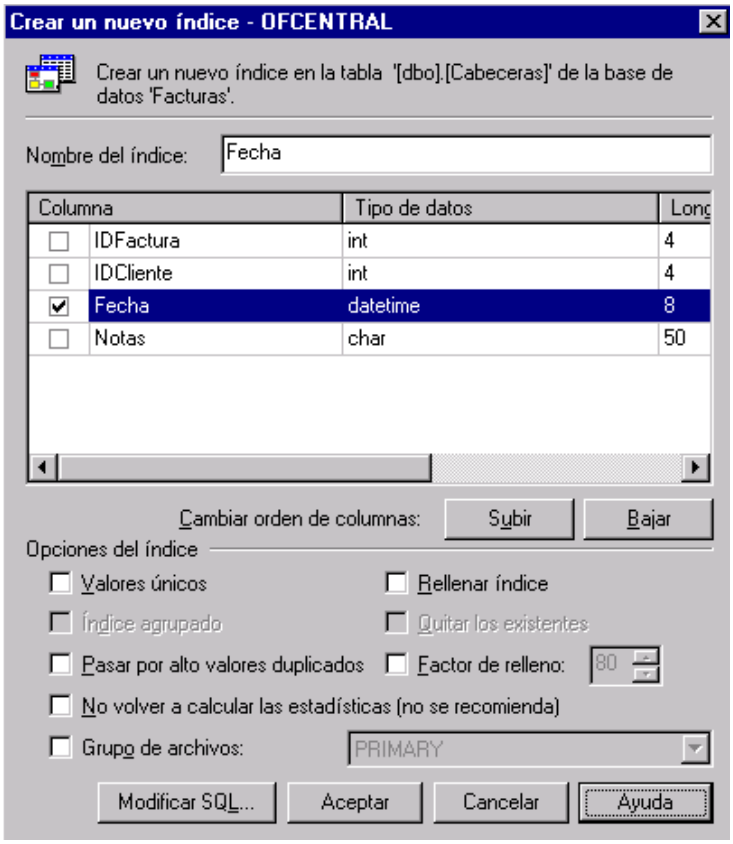


Figura 76. Ventana para crear un nuevo índice.

Creación con CREATE INDEX

Esta instrucción, ejecutada desde el *Analizador de consultas* o una aplicación, permitirá crear los índices para una tabla y configurarlos mediante sus parámetros.

```
CREATE [ UNIQUE] [CLUSTERED | NONCLUSTERED ]
INDEX NombreÍndice ON NombreTabla ( NombreColumna [ ,
...NombreColumnaN ] )
[ WITH
[ PAD_INDEX ]
[ [,] FILLFACTOR = FactorRelleno ]
[ [,] IGNORE_DUP_KEY ]
[ [,] DROP_EXISTING ]
[ [,] STATISTICS_NORECOMPUTE ] ]
[ ON GrupoArch ]
```

Proporcionada toda la información, aceptaremos esta ventana, y cerraremos también la primera, de forma que habremos creado un nuevo índice para la base de datos.

- UNIQUE. Crea un índice único, en el que no podrán existir dos claves iguales.
- CLUSTERED, NONCLUSTERED. Indica el tipo de índice.
- NombreÍndice. Nombre asignado al índice.
- NombreTabla. Tabla que contiene la columna a partir de la que se va a crear el índice.

- NombreColumna. Columna desde la que se creará el índice.
- PAD_INDEX. Espacio que debe dejarse abierto en cada página de los niveles intermedios del índice.
- FILLFACTOR. Porcentaje utilizado por SQL Server para conocer la cantidad de filas de índice que ocupará cada página.
- IGNORE_DUP_KEY. Permite controlar las situaciones en las que se intenta insertar una clave duplicada en un índice que no lo permite.
- DROP_EXISTING. Indica que el índice existente debe eliminarse antes de crear el actual.
- STATISTICS_NORECOMPUTE. Indica que las estadísticas no actualizadas del índice no se vuelvan a calcular.
- GrupoArch. Nombre del grupo de archivos dentro del cual se creará el índice.

La sentencia del Código fuente 13, creará un índice con el nombre IndNombre, basado en el campo Nombre de la tabla Clientes.

```
CREATE UNIQUE
INDEX IndNombre ON Clientes (Nombre)
```

Código fuente 13

Creación de procedimientos almacenados

Un procedimiento almacenado de SQL Server, consiste en un conjunto de instrucciones Transact SQL que se ejecutan como una rutina de un lenguaje de programación convencional. Con la posibilidad de recibir parámetros y devolver valores.

La instrucción CREATE PROCEDURE es la encargada de crear estos elementos en el gestor de datos.

```
CREATE PROCEDURE Procedimiento [ ;NúmeroProc ]
[ @Parámetro TipoDatos ] [ VARYING ] [=Defecto ] [ OUTPUT ]
[ ,... @ParamN ]
[ WITH RECOMPILE | ENCRYPTION |RECOMPILE, ENCRYPTION ]
[ FOR REPLICATION ]
AS
InstruccionesSQL
```

- Procedimiento. Nombre del procedimiento almacenado.
- NúmeroProc. Podemos crear varios procedimientos agrupados bajo un mismo nombre, asignando a cada uno un valor ordinal.
- @Parámetro. Nombre del parámetro que recibirá el procedimiento.
- TipoDatos. Tipo de dato del parámetro.

- VARYING. Esta cláusula se aplica a los parámetros de cursor e indica el conjunto de resultados admitido como parámetro de salida.
- Defecto. Valor por defecto para el parámetro.
- OUTPUT. Parámetro de retorno.
- RECOMPILE. SQL Server compila de nuevo el procedimiento cada vez que lo ejecuta.
- ENCRYPTION. El texto del procedimiento se codifica al ser creado.
- FOR REPLICATION. Indica que los procedimientos creados para duplicación no pueden ejecutarse en el servidor de suscripción. Se emplea al crear un procedimiento de filtro que se ejecuta mediante duplicación. Esta clausula y WITH RECOMPILE son mutuamente excluyentes.
- Instrucciones SQL. Conjunto de sentencias que componen el cuerpo del procedimiento almacenado.

A continuación, se ilustran las formas de creación de procedimientos almacenados para este gestor de datos.

Administrador corporativo

Nos situaremos en el elemento *Procedimientos almacenados* de una base de datos y seleccionaremos la opción de menú *Acción+Nuevo procedimiento almacenado*, que abrirá la ventana de código para los procedimientos almacenados mostrada en la Figura 77.

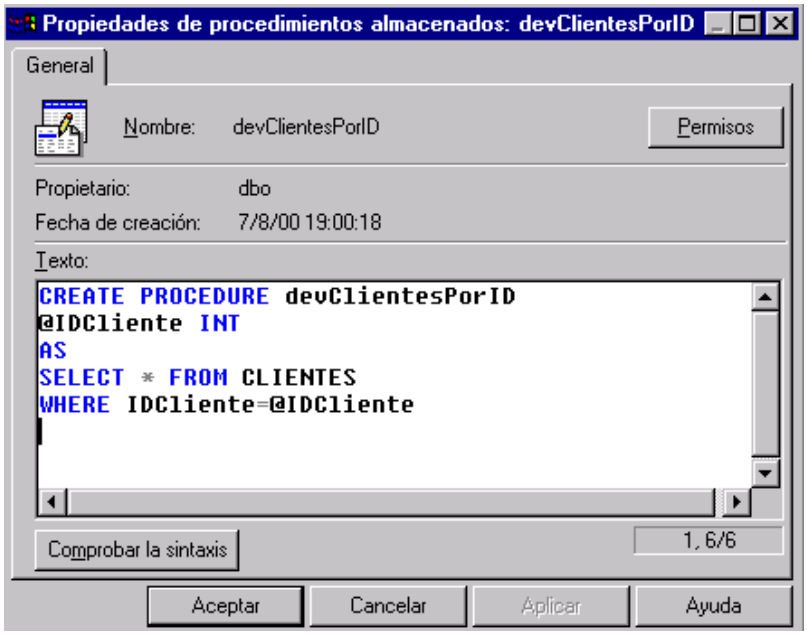


Figura 77. Ventana para creación de procedimientos almacenados.

Como podemos apreciar, en este ejemplo hemos escrito un procedimiento para la tabla Clientes de la base de datos Facturas, que nos devuelve la fila de la tabla que coincide con el parámetro pasado al procedimiento.

Antes de guardar el procedimiento, es conveniente pulsar el botón *Comprobar la sintaxis* para verificar posibles fallos, aunque si existe algún error, SQL no permitirá grabarlo. También podemos especificar los permisos de ejecución de este procedimiento pulsando el botón Permisos, y si hacemos clic con el botón derecho del ratón sobre la zona de edición y elegimos la opción Fuente, podremos configurar algunos aspectos de la edición, para adaptarla a nuestro tipo de fuente y colores preferidos. Ver Figura 78.

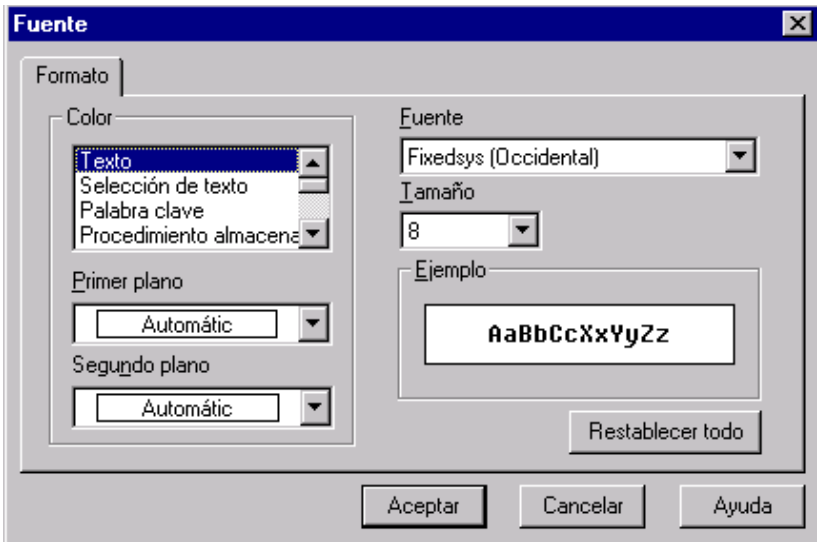


Figura 78. Configuración del editor de procedimientos almacenados.

Finalizada la escritura del procedimiento, pulsaremos Aceptar, con lo que se creará una nueva entrada en el apartado dedicado a procedimientos almacenados de la base de datos en la que estemos posicionados.

Analizador de consultas

Para crear un procedimiento almacenado mediante esta herramienta, debemos seleccionar en primer lugar la base de datos en la que crearemos el procedimiento, o emplear el comando *USE NombreBD,* para situarnos en dicha base de datos.

A continuación escribiremos el código del procedimiento de la misma manera que hemos hecho en el Administrador corporativo, y ejecutaremos dicho conjunto de líneas, creándose el nuevo procedimiento. En este caso crearemos un procedimiento almacenado que nos devuelva las filas de la tabla Clientes, ordenadas por la columna Nombre, como muestra el Código fuente 14.

```
CREATE PROCEDURE devClientesNombre
AS
SELECT * FROM CLIENTES
ORDER BY NOMBRE
```

Código fuente 14

Para ejecutar un procedimiento almacenado, emplearemos la instrucción EXECUTE (o EXEC, de forma abreviada), seguida del nombre del procedimiento almacenado y los parámetros a pasar si fuera necesario, como podemos ver en el Código fuente 15.

```
EXECUTE devClientesNombre
```

Código fuente 15

Parámetros denominados

La forma habitual de pasar parámetros a un procedimiento almacenado, consiste en situar los valores correspondientes a los parámetros, separados por comas, y en el mismo orden en que están dispuestos en la declaración del procedimiento almacenado. Por ejemplo, si disponemos del procedimiento almacenado mostrado en el Código fuente 16.

```
CREATE PROCEDURE VerClientes
@Ciudad VARCHAR(50),
@Sector VARCHAR(50)
AS
SELECT * FROM Clientes
WHERE Ciudad=@Ciudad
AND Sector=@Sector
ORDER BY Nombre
```

Código fuente 16

El modo más frecuenta de llamarlo sería como se indica en el Código fuente 17.

```
EXEC VerClientes 'Madrid','Oeste'
```

Código fuente 17

Sin embargo, existe otro modo de pasar parámetros a un procedimiento almacenado, consistente en el uso de parámetros denominados, es decir, indicando el nombre que recibe el parámetro en la declaración del procedimiento almacenado, seguido del valor que se asigna a dicho parámetro. Este sistema nos permite obviar los parámetros a los que no vamos a pasar valor, lo cual, supone una ventaja en procedimientos almacenados con un gran número de parámetros. Por otro lado, con esta técnica es posible cambiar el orden de los parámetros.

El Código fuente 18 muestra como llamar al procedimiento almacenado VerClientes, cambiando el orden de los parámetros pasados.

```
EXEC VerClientes @Sector='Oeste', @Ciudad='Madrid'
```

Código fuente 18

A lo largo del texto, en función de la situación, haremos uso de uno y otro modo de paso de parámetros.

Creación de vistas

Una vista es una instrucción de consulta con un nombre, sobre una o varias tablas. Estos objetos de la base de datos permiten al administrador poner a disposición de los usuarios los datos sólo a efectos de consulta, eliminando el potencial riesgo que supondría el acceso directo a las tablas.

Administrador corporativo

Después de abrir una base de datos, haremos clic con el botón derecho sobre Vistas y seleccionaremos la opción *Nueva vista* del menú contextual, mostrándose la ventana de creación de vistas de SQL Server. Ver Figura 79.

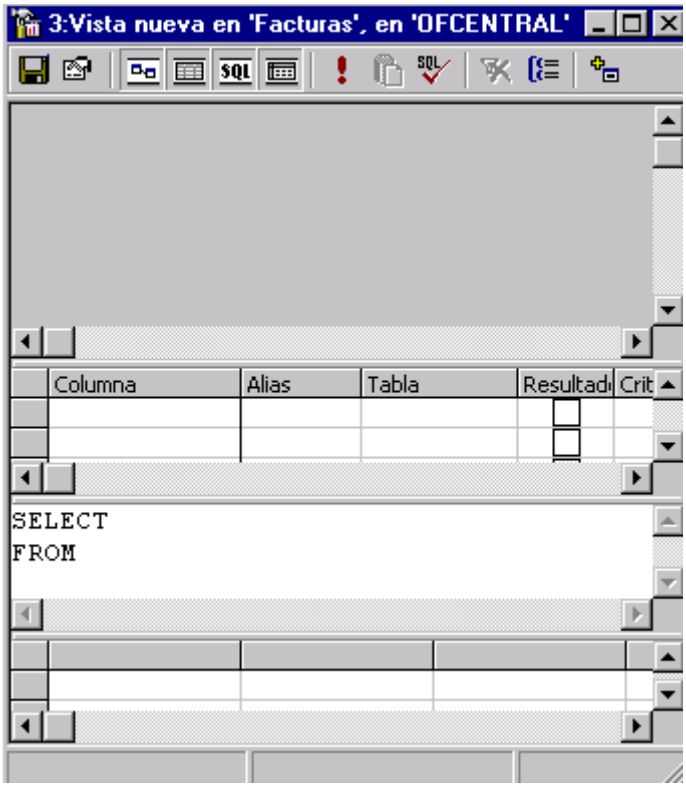


Figura 79. Ventana de vistas de SQL Server.

Esta ventana , Figura 80, está dividida en cuatro paneles, que se describen a continuación:

- Diagrama. Muestra las tablas que forman parte de la sentencia.
- Cuadrícula. Permite indicar las columnas que se mostrarán en la vista y las condiciones de filtrado de datos.
- SQL. Visualiza la consulta que vamos construyendo en lenguaje SQL.

- Resultados. Permite previsualizar el resultado de la sentencia construida en la vista.

Figura 80. Botones para acceder a los paneles de la ventana de vistas.

Para construir una vista, pulsaremos el botón de la barra de herramientas *Agregar tabla*, Figura 81, mediante el que añadiremos tablas de la base de datos a la vista.

Figura 81. Botón agregar tabla de la ventana de vistas.

Seguidamente, seleccionaremos las columnas a mostrar y el criterio de selección en el panel Cuadrícula. En todo momento, podremos pulsar el botón *Comprobar SQL* para verificar la sintaxis de la sentencia, y el botón Ejecutar para obtener el resultado correspondiente. Ver Figura 82.

Figura 82. Botones para comprobar instrucción y ejecutar la vista.

En el ejemplo mostrado a continuación, Figura 83, se crea una vista con el nombre VerNombreCli, que muestra una selección de registros de la tabla Clientes en la base de datos Facturas.

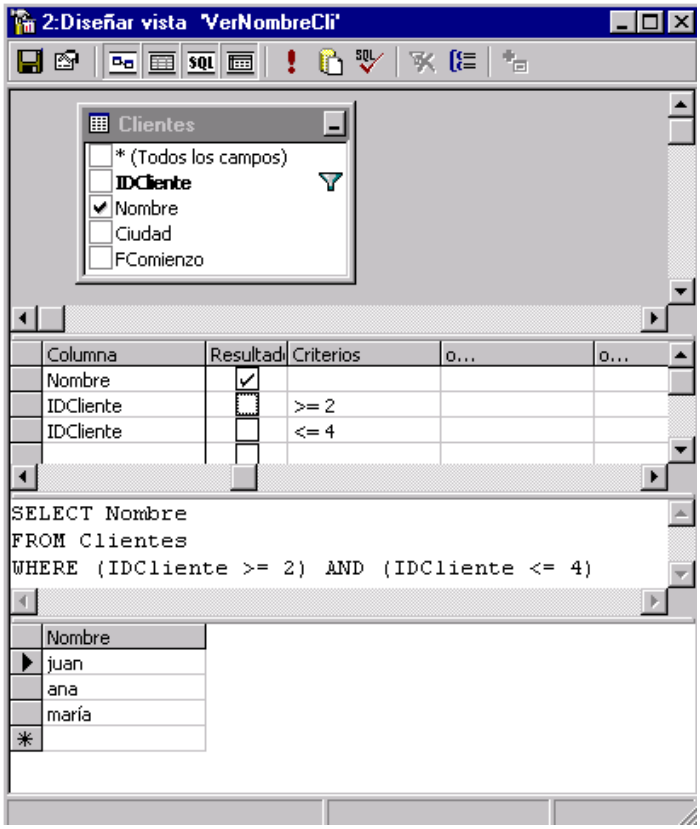


Figura 83. Vista creada desde el Administrador corporativo.

Una vez creada una vista, podemos modificarla haciendo doble clic sobre la misma, que nos abrirá la ventana de propiedades con el código que la compone, como vemos en la Figura 84.

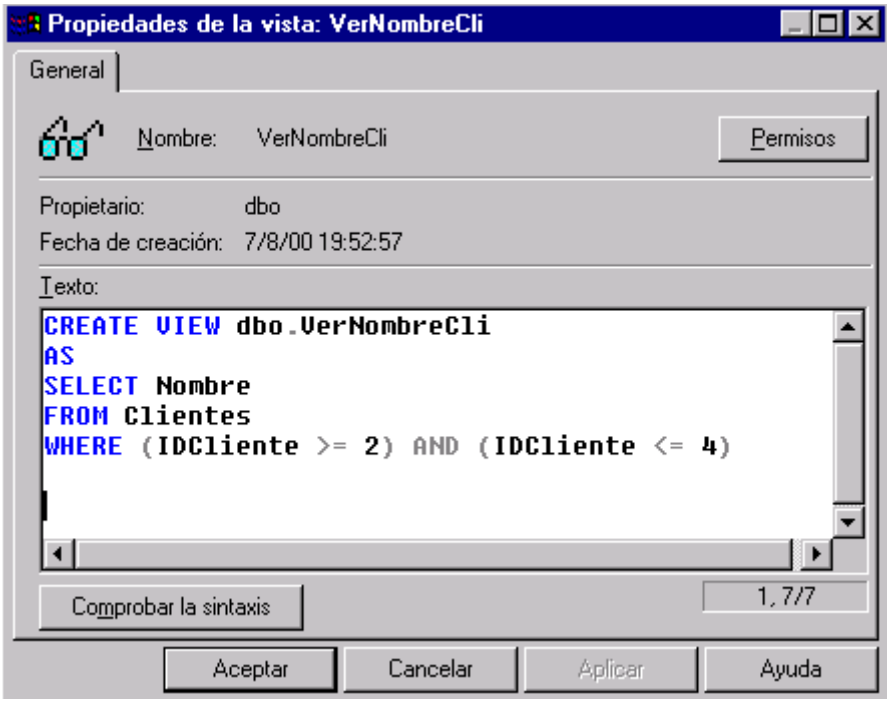


Figura 84. Ventana de propiedades de una vista.

Si preferimos realizar la modificación utilizando la ventana de diseño, haremos clic sobre la vista con el botón derecho del ratón, seleccionando la opción *Diseñar vista* del menú contextual.

Instrucción CREATE VIEW

Mediante esta sentencia ejecutada desde el Analizador de consultas, podremos crear vistas para una base de datos, con la única diferencia que no dispondremos del asistente en diseño del Administrador corporativo.

```
CREATE VIEW NombreVista [ ( ColCalculada [ ,…ColCalculadaN ] ) ]
[ WITH ENCRYPTION ]
AS
InstruccionesSelect
[ WITH CHECK OPTION ]
```

- NombreVista. Identificador de la vista.
- ColCalculada. Nombre que usaremos cuando utilicemos columnas calculadas o al recuperar columnas con nombres repetidos.
- ENCRYPTION. Codifica el contenido de la vista al crearse.
- InstruccionesSelect. Conjunto de sentencias que forman el cuerpo de la vista.

- CHECK OPTION. Fuerza a que las modificaciones ejecutadas contra la vista se atengan a los criterios establecidos InstruccionesSelect. Cuando al ejecutar una vista, se modifica una fila, esta cláusula garantiza que los datos se visualicen después de realizar la modificación.

El Código fuente 19, crea una vista llamada VerFacturas, que devuelve las filas de la tabla Cabeceras basándose en una condición. La forma de ejecución es igual que la explicada en el anterior punto.

```
CREATE VIEW VerFacturas
AS
SELECT *
FROM Cabeceras
WHERE IDCliente = 3
```

Código fuente 19

Modificación de tablas

Una vez creada una tabla, puede ser preciso modificar el tipo de dato de una columna, añadir, eliminar columnas, etc.

Administrador corporativo

Desde esta utilidad, haremos clic con el botón derecho del ratón sobre la tabla a modificar, seleccionando la opción de menú contextual *Diseñar tabla*, que abrirá la ventana de diseño de la tabla, sobre la que podemos realizar todo tipo de cambios, que deberemos guardar antes de cerrar.

Instrucción ALTER TABLE

Empleada desde el Analizador de consultas, esta instrucción nos permitirá manipular el contenido de una tabla, añadiendo, modificando o eliminando sus columnas.

```
ALTER TABLE NombreTabla
[ ALTER COLUMN NombreColumna TipoDatos ]
[ NULL | NOT NULL ]
| ADD [ NombreColumna TipoDatos ]
| [ NombreColumna AS ColumnaCalculada ][ ,... ColumnaN ]
| DROP NombreColumna [ ,... ColumnaN ]
```

- NombreTabla. Nombre de la tabla a modificar.
- NombreColumna. Nombre de la columna a modificar.
- TipoDatos. Tipo de datos de la columna a modificar.
- NULL, NOT NULL. Permiten indicar si la columna aceptará o no, valores nulos.
- ADD. Se emplea para añadir nuevas columnas.
- ColumnaCalculada. Nombre de una nueva columna, creada a partir de una operación.

- DROP. Se emplea para eliminar columnas existentes.

Las instrucciones del Código fuente 20, modifican la tabla Cabeceras de la base de datos Facturas, añadiendo una nueva columna, eliminando una columna existente, y modificando el tipo de datos de otra.

```
ALTER TABLE Cabeceras
ADD Saldo INT

ALTER TABLE Cabeceras
DROP COLUMN Fecha

ALTER TABLE Cabeceras
ALTER COLUMN Notas INT
```

Código fuente 20

Modificación de índices

Para modificar un índice, debemos acceder a la ventana de administración de índices explicada en el apartado de creación de índices y cambiar los valores del índice correspondiente, o bien utilizar la instrucción CREATE INDEX junto a la cláusula DROP EXISTING para regenerar el índice de nuevo, eliminando el existente.

Modificación de procedimientos almacenados

Administrador corporativo

Podemos cambiar la ejecución de un procedimiento almacenado, haciendo doble clic sobre el mismo, de forma que se abrirá su ventana de edición, permitiéndonos la modificación de las instrucciones que lo componen. Una vez finalizados los cambios y habiendo comprobado que la sintaxis es correcta, pulsaremos *Aceptar* para grabar los cambios.

Instrucción ALTER PROCEDURE

Otra opción para modificar un procedimiento almacenado, consiste en emplear esta instrucción desde código, que sustituirá el contenido actual del procedimiento por el nuevo pasado como parámetro. En el Código fuente 21, modificaremos el procedimiento devClientesNombre de la base de datos Facturas.

```
ALTER PROCEDURE devClientesNombre
AS
SELECT * FROM CLIENTES
WHERE IDCliente >3
ORDER BY NOMBRE
```

Código fuente 21

Modificación de vistas

Administrador corporativo

Podemos cambiar el contenido de una vista, haciendo doble clic sobre la misma y abriendo su ventana de edición de código, o mediante la ventana de diseño de vistas, comentada en un apartado anterior.

Una vez finalizados los cambios y habiendo comprobado que la sintaxis es correcta, grabaremos los cambios y cerraremos la ventana correspondiente.

Instrucción ALTER VIEW

Para modificar el contenido de una vista mediante código, usaremos esta instrucción, que sustituirá el contenido actual de la vista por el nuevo pasado como parámetro.

En el Código fuente 22, modificaremos el procedimiento devClientesNombre de la base de datos Facturas.

```
ALTER VIEW VerNombreCli
AS
SELECT *
FROM Clientes
WHERE IDCliente <>= 4
```

Código fuente 22

Eliminación de tablas

Administrador corporativo

Podemos borrar una tabla existente, haciendo clic con el botón derecho sobre la misma y seleccionando la opción de menú Eliminar, o bien, seleccionando la tabla de forma normal y pulsando la tecla [SUPR].

Instrucción DROP TABLE

Esta instrucción permite quitar una tabla de la base de datos, pasando su nombre como parámetro.

```
DROP TABLE NombreTabla
```

Eliminación de índices

Administrador corporativo

Haremos clic con el botón derecho del ratón sobre la tabla que contiene el índice y seleccionaremos la opción de menú *Todas las tareas+Administrar índices*. Una vez situados en esta ventana de manipulación de índices, haremos clic sobre el índice a borrar y pulsaremos el botón Eliminar.

Instrucción DROP INDEX

Mediante esta sentencia eliminaremos un índice de la base de datos, pasando su nombre como parámetro:

```
DROP INDEX NombreÍndice
```

Eliminación de procedimientos almacenados

Administrador corporativo

Para eliminar un procedimiento almacenado, haremos clic con el botón derecho sobre el mismo y seleccionaremos la opción de menú Eliminar, o bien, seleccionaremos el procedimiento y pulsaremos la tecla [SUPR].

Instrucción DROP PROCEDURE

Mediante esta instrucción borraremos un procedimiento almacenado de la base de datos, pasando su nombre como parámetro.

```
DROP PROCEDURE NombreProcedimiento
```

Eliminación de vistas

Administrador corporativo

Para eliminar una vista, haremos clic con el botón derecho sobre la misma y seleccionaremos la opción de menú Eliminar, o bien, seleccionaremos la vista y pulsaremos la tecla [SUPR].

Instrucción DROP VIEW

Mediante esta instrucción borraremos una vista de la base de datos, pasando su nombre como parámetro.

```
DROP VIEW NombreVista
```

19

Gestión de la seguridad. Autenticación

El entorno de seguridad de SQL Server 7.0

Como se apuntaba en un tema anterior, SQL Server ha optimizado el esquema de seguridad de versiones anteriores, integrándolo con la seguridad del sistema operativo, de modo que resulte más robusto ante posibles accesos no permitidos, pero al mismo tiempo, más fácil de gestionar de cara al administrador de la base de datos.

Cuando se menciona la integración con la seguridad del sistema operativo, estamos refiriéndonos a sistemas servidores: Windows NT Server, Windows 2000 Server, etc., ya que en las instalaciones de SQL Server bajo sistemas cliente: Windows 95/98, se seguirá usando el mecanismo de seguridad propio de SQL Server.

En la versión actual de SQL Server, el acceso al servidor y sus componentes está estructurado en una serie de niveles ante los que el usuario se deberá identificar, empleando para ello unos elementos del motor de datos creados para realizar tales labores, como son los inicios de sesión, funciones y permisos.

Niveles de seguridad

SQL Server 7.0 implementa una técnica de seguridad basada en niveles, que consiste en comprobar si las características del usuario que intenta acceder son las adecuadas en cada nivel. La Figura 85 muestra un esquema de este mecanismo de seguridad.

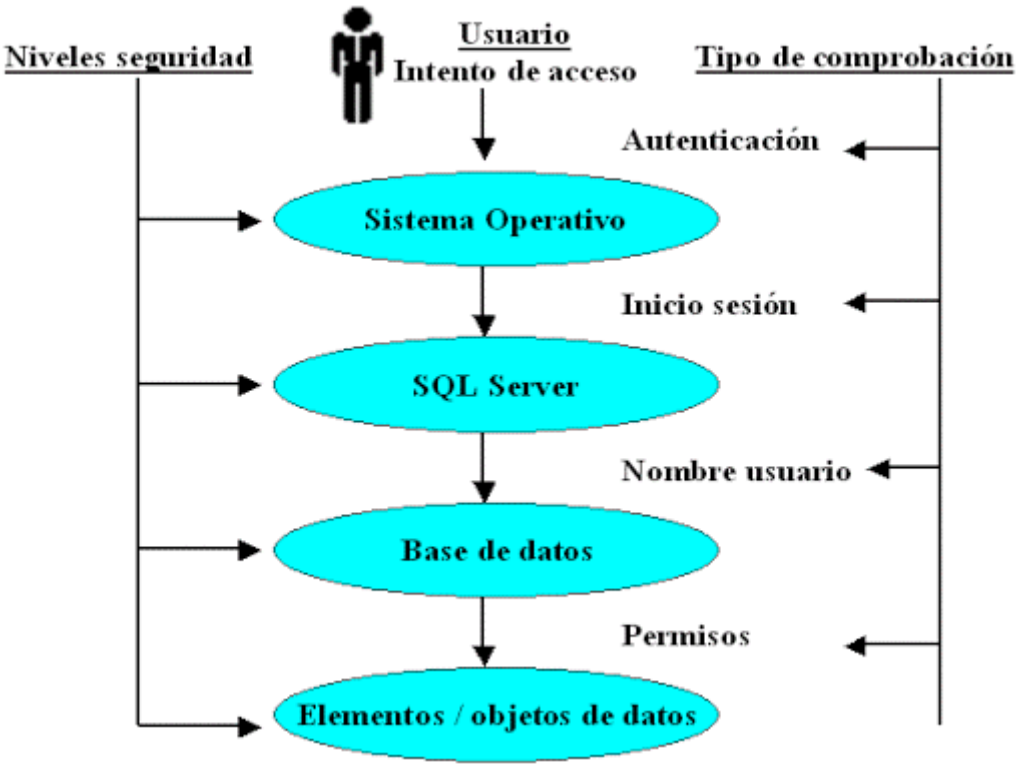


Figura 85. Niveles de seguridad en SQL Server 7.0.

Según el esquema anterior, cuando un usuario intenta acceder a los datos de un servidor SQL Server, debe traspasar los niveles de seguridad mostrados utilizando el mecanismo de comprobación adecuado para cada nivel. Repasemos este proceso paso a paso.

En primer lugar, tenemos el nivel del sistema operativo, al cual debemos acceder mediante el proceso denominado autenticación.

A continuación nos encontramos con el nivel de SQL Server, que pasaremos utilizando el denominado inicio de sesión.

Seguiremos con el nivel de base de datos, para el que necesitaremos estar dados de alta con un nombre de usuario de la base de datos. Debemos tener en cuenta que el inicio de sesión sólo nos conecta con el servidor SQL Server que indiquemos, pero eso no significa que podamos tener pleno acceso a cualquier base de datos del servidor. Supongamos que un servidor tiene una base de datos con las facturas de la empresa y otra con los resultados de los beneficios; en el caso de que no existiera este tipo de restricción, un usuario con perfil de administrativo podría acceder a la base de datos de resultados, que se supone sólo debe ser accesible por los usuarios con perfil de directivo.

Finalmente se sitúa el nivel de elementos y objetos de datos: tablas, vistas, procedimientos almacenados, tipos de manipulación de datos (altas, bajas y modificaciones), etc.; a los que podremos acceder en el caso de disponer de los permisos adecuados.

Con este último nivel de profundidad en la seguridad del servidor, conseguimos lo que se podría denominar un ajuste fino de la seguridad, estableciendo los objetos de datos sobre los que un usuario puede tener acceso y qué operaciones puede llevar a cabo con ellos.

Inicios de sesión

Un inicio de sesión es aquel componente en el sistema de seguridad, que le sirve a SQL Server para saber si el usuario que está intentando acceder tiene el perfil adecuado para establecer una conexión con el servidor. A este proceso de comprobación se le denomina autenticación.

Al igual que el sistema operativo dispone de las cuentas de usuario para reconocer quién está intentando acceder al sistema y permitirle o denegarle el paso, SQL Server utiliza este sistema similar, para controlar quién intenta entrar a sus servidores de datos.

Existen dos tipos de autenticación: autenticación de SQL Server y autenticación de Windows NT, que serán explicados en los siguientes apartados.

Autenticación de SQL Server

En este tipo de autenticación, es SQL Server el que se encarga directamente de comprobar la validez del usuario que intenta acceder al sistema. Para ello, debe existir un identificador de inicio de sesión válido que permita dicho acceso.

Creación de un inicio de sesión para autenticación SQL Server

Desde el Administrador corporativo, una vez conectados con un servidor SQL Server, y desplegadas todas las carpetas que lo componen, haremos doble clic sobre la carpeta Seguridad o clic sobre su icono de apertura, que nos mostrará todos los elementos de esta carpeta. A continuación, seleccionaremos el elemento *Inicios de sesión*, visualizando en el panel derecho la lista de inicios de sesión disponibles para el servidor, como muestra la Figura 86.

Figura 86. Inicios de sesión en SQL Server 7.0.

Situados de esta forma, seleccionaremos el menú *Acción+Nuevo inicio de sesión*, que nos abrirá la ventana de creación de un inicio de sesión con las propiedades para el mismo.

Esta ventana dispone de varias pestañas, de momento sólo utilizaremos la pestaña por defecto, General. En ella introduciremos el nombre del inicio: jroble, pulsaremos el botón de opción *Autenticación de SQL Server* y escribiremos la contraseña para este inicio. Veamos en la Figura 87, los valores introducidos.

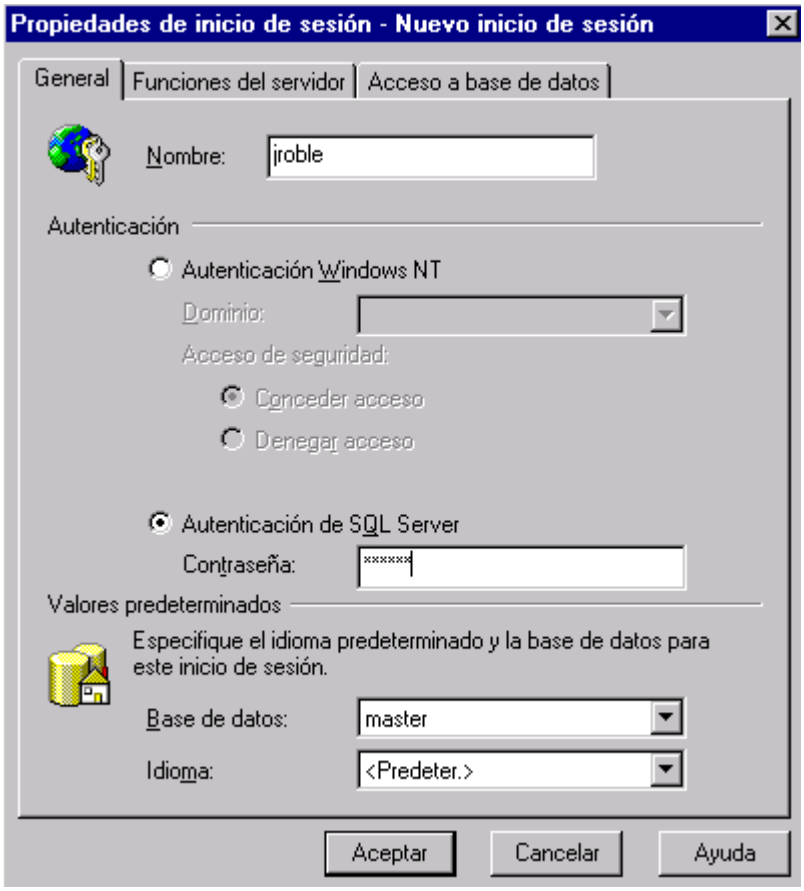


Figura 87. Propiedades de un nuevo inicio de sesión.

La contraseña es opcional, pero por motivos de seguridad siempre es recomendable introducirla. En el apartado *Valores predeterminados*, podemos establecer cuál será la base de datos e idioma por defecto a la que tendrá acceso el usuario cuando se conecte. Por el momento dejaremos los valores que nos ofrece SQL Server.

Finalmente pulsaremos Aceptar, y tras confirmar nuevamente la contraseña introducida, se creará el nuevo inicio de sesión, añadiéndose a la lista existente.

Conectar con un servidor mediante autenticación SQL Server

Para conectar con el nuevo inicio de sesión que hemos creado, en primer lugar tendremos que actualizar los cambios realizados en el servidor SQL Server. Esto lo conseguiremos, haciendo clic con el botón derecho sobre el nombre del servidor, y seleccionando la opción *Actualizar* del menú contextual mostrado.

Seguidamente y mediante el mismo menú contextual, elegiremos la opción *Desconectar*. Una vez desconectados del servidor como el usuario actual, mediante este mismo menú, seleccionaremos la opción *Modificar propiedades de registro del servidor SQL Server*, que nos mostrará el cuadro de

diálogo de propiedades del servidor. Aquí deberemos pulsar en el apartado Conexión, la opción *Utilizar autenticación SQL Server,* e introducir el nombre de inicio de sesión y contraseña que acabamos de crear, como vemos en la Figura 88.

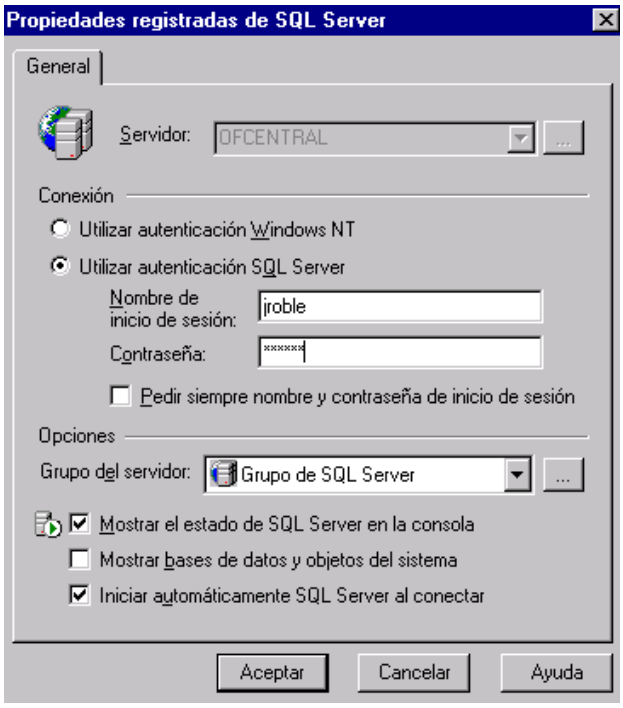


Figura 88. Propiedades de registro de un servidor SQL Server.

Pulsaremos Aceptar, quedando los nuevos valores de conexión actualizados para el servidor. Si se nos muestra un cuadro de confirmación en el que se nos pide desconectar, pulsaremos Aceptar para el caso de que todavía existiera alguna conexión con el servidor de datos.

Debemos recordar que a este nivel, sólo habremos conseguido conectar, no tener pleno acceso a los datos. Las únicas bases de datos a las que podremos acceder serán las del sistema o las de ejemplo: master, Northwind, etc. Este aspecto podemos comprobarlo si abrimos la carpeta *Bases de datos* e intentamos acceder a alguna de base de datos creada por el usuario, en nuestro caso Almacenes, y hacemos clic sobre Tablas, obteniendo el error de la Figura 89.

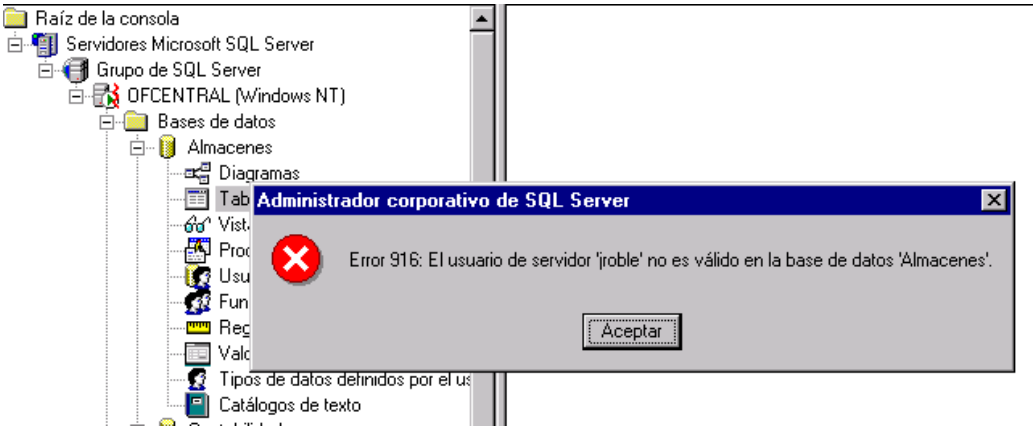


Figura 89. Error de acceso a una base de datos sin autorización.

La Figura 90 muestra un esquema de este tipo de autenticación.

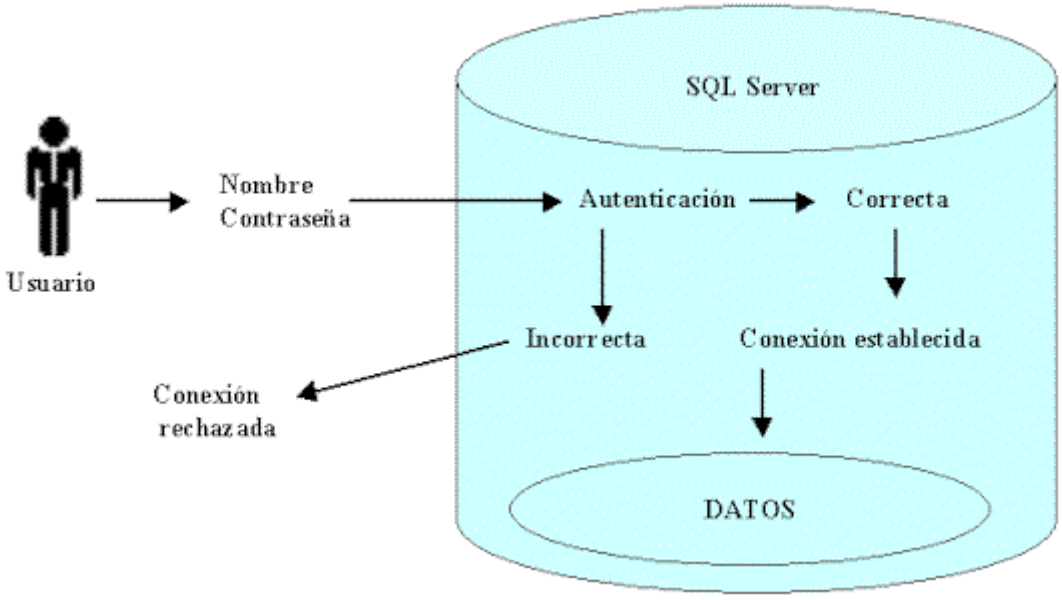


Figura 90. Esquema de conexión con autenticación SQL Server.

Autenticación de Windows NT

Este tipo de autenticación (sólo disponible para usuarios de Windows NT 4.0 o superior, en sus versiones servidor), permite un acceso directo del usuario al servidor de datos, siempre que dicho usuario haya sido validado por el sistema operativo.

SQL Server establece una relación de confianza con el sistema operativo, es decir, detecta que el usuario ya ha sido validado por Windows NT, de manera que evita realizar de nuevo el proceso de autenticación, confiando en que dicha validación ha sido efectuada correctamente, por lo que conecta al usuario con la base de datos.

Creación de un inicio de sesión para autenticación Windows NT

En primer lugar, situados como administradores en Windows NT, crearemos una nueva cuenta de usuario, con el nombre egaleo y el resto de opciones que se muestran en la Figura 91.

Abriremos seguidamente el Administrador corporativo de SQL Server, conectaremos con uno de los servidores disponibles y pulsaremos en la carpeta Seguridad. Haremos clic con el botón derecho en el elemento *Inicios de Sesión*, seleccionando la opción de menú *Nuevo inicio de sesión*, que mostrará la ventana dedicada a crear un nuevo inicio de sesión para el servidor. En esta ventana, deberemos introducir el nombre de la cuenta que acabamos de crear en el sistema operativo, indicar que el modo de autenticación será el de Windows NT, dominio, y demás opciones que se muestran en la Figura 92.

Es importante aclarar, que en el anterior paso, hemos agregado una cuenta de usuario del sistema operativo al servidor SQL Server, no hemos creado una nueva cuenta de inicio de sesión de SQL Server, a pesar de que externamente el proceso es igual.

Figura 91. Nueva cuenta de usuario en Windows NT.

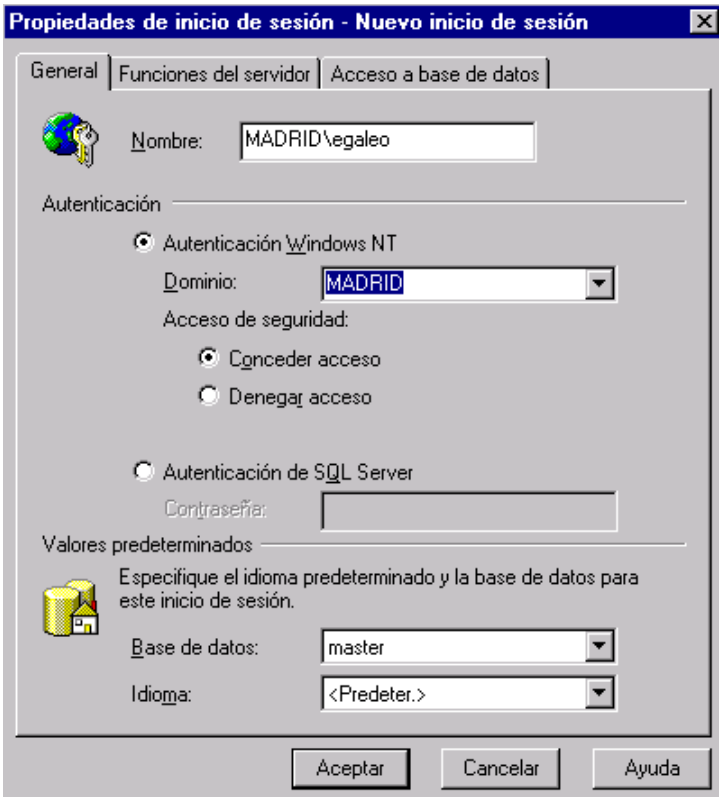


Figura 92. Agregando una cuenta de usuario de Windows NT a SQL Server

Desconectaremos del servidor haciendo clic sobre él y seleccionando con el botón derecho del ratón la opción del menú contextual *Desconectar*.

Abriremos la ventana de propiedades del registro del servidor, al igual que en el tipo de autenticación anterior y pulsaremos sobre la opción *Utilizar autenticación Windows NT*. Aceptando los cambios de esta ventana habremos completado esta fase de configuración del tipo de autenticación. Ver Figura 93.

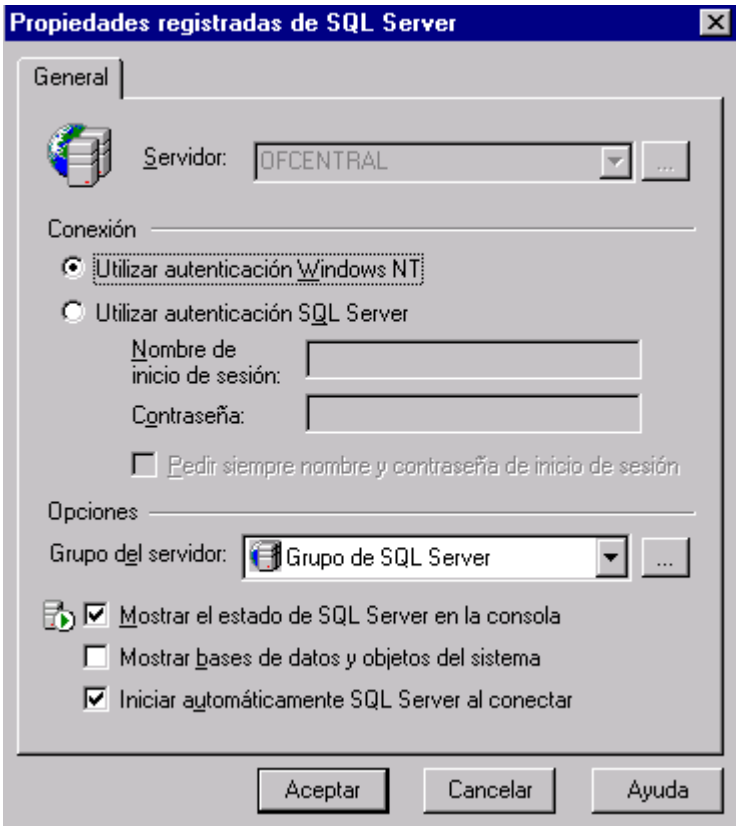


Figura 93. Propiedades de registro de un servidor SQL Server.

Conectar con un servidor mediante autenticación Windows NT

Realizaremos una nueva conexión a NT como el usuario recién creado, abriremos el Administrador corporativo de SQL, y conectaremos con el servidor antes mencionado desplegando sus elementos. En este punto SQL Server ha hecho uso de la autenticación de Windows NT; al comprobar que el usuario había sido validado por el sistema, ha confiado en dicha validación conectándole a su servidor. Ver Figura 94.

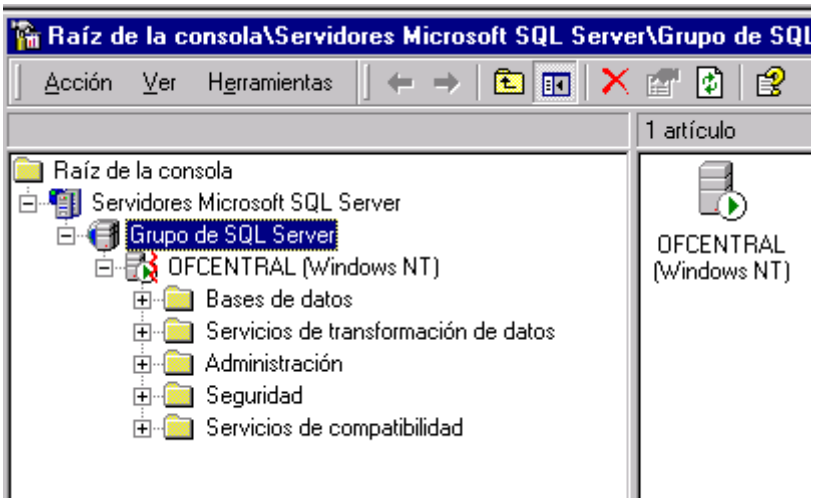


Figura 94. Conexión de un usuario a un servidor SQL Server mediante autenticación Windows NT.

Sin embargo, al igual que con la autenticación de SQL Server, nuestro usuario sólo ha logrado conectar con el servidor, no con sus objetos. Podrá acceder de esta forma a las bases de datos del servidor en las que esté incluido como usuario invitado (el usuario especial guest), como son master, Northwind, etc. Si intenta, por ejemplo, acceder a las tablas de la base de datos Almacenes, en las que no está definido este usuario, obtendrá el error mostrado en la Figura 95.

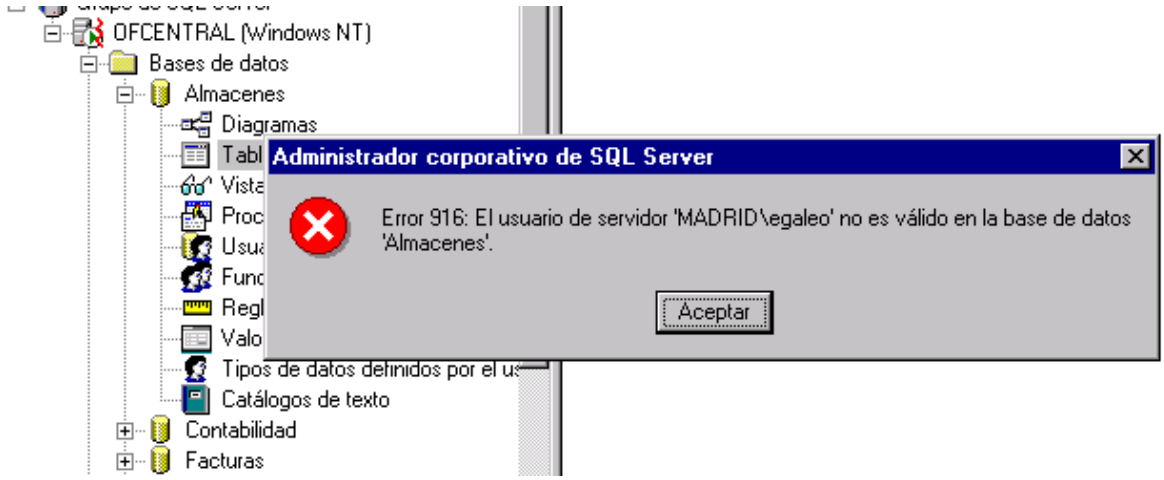


Figura 95. Acceso no autorizado a una base de datos en SQL Server.

Este tipo de autenticación se describe en el esquema de la Figura 96.

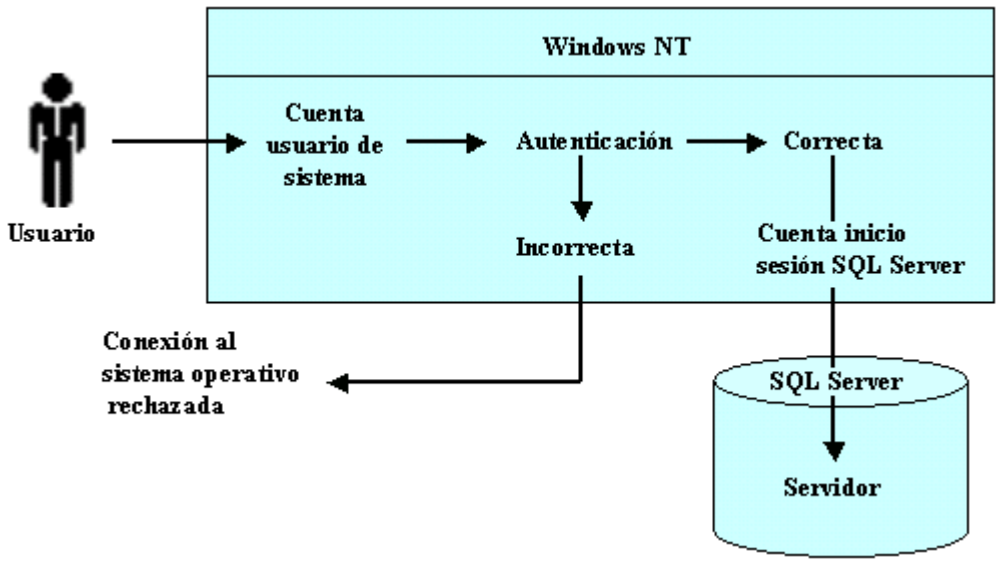


Figura 96. Esquema de conexión a SQL Server con autenticación de Windows NT.

Autenticación mediante grupos de Windows NT

Una de las ventajas de la integración de seguridad entre Windows NT y SQL Server, consiste en que podemos utilizar los grupos de usuarios de NT para crear un inicio de sesión único para varios usuarios, evitando la laboriosa tarea de crear inicios de sesión independientes para usuarios con idénticos perfiles a la hora de acceder a una base de datos. Veamos este aspecto mediante un ejemplo.

Creación de los usuarios en Windows NT

En primer lugar, entraremos en el *Administrador de usuarios para dominios* de Windows NT y crearemos tres nuevos usuarios con los nombres ameson, jcarrera y pmontes, tanto para el nombre identificativo como para la contraseña. El lector puede poner el que prefiera, teniendo en cuenta que también es importante establecer que la contraseña no debe caducar. La Figura 97 muestra uno de ellos.

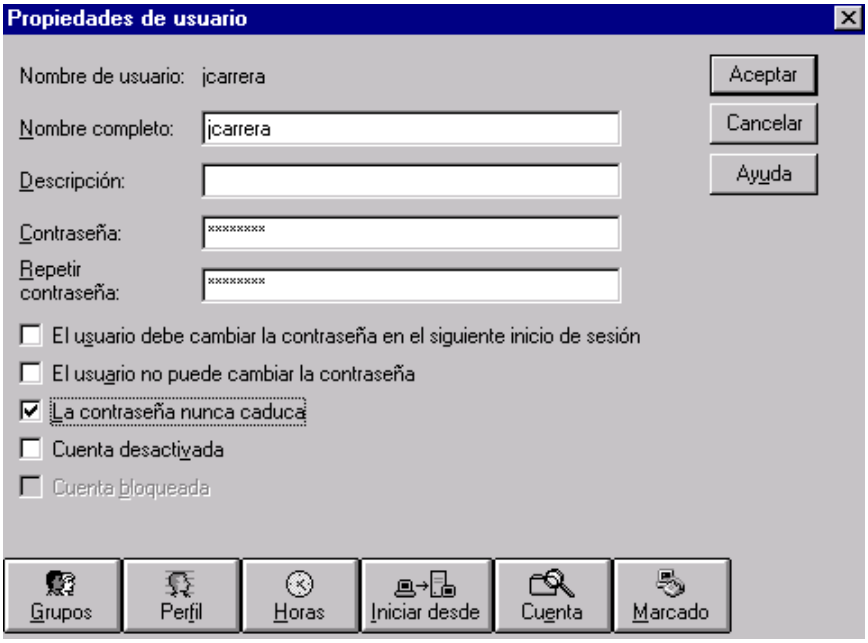


Figura 97. Nuevo usuario de Windows NT.

Creación de un grupo de usuarios en Windows NT

A continuación, y dentro de esta misma utilidad, crearemos un nuevo grupo de usuarios, seleccionando la opción de menú *Usuario+grupo local nuevo*, al que daremos el nombre *Revisores*, como se muestra en la Figura 98.

Figura 98. Nuevo grupo de usuarios de Windows NT.

El siguiente paso consiste en agregar los usuarios creados anteriormente al grupo. Para ello, pulsaremos el botón Agregar, que nos mostrará una ventana destinada a tal efecto, véase la Figura 99, en la que seleccionaremos los usuarios que formarán parte el grupo.

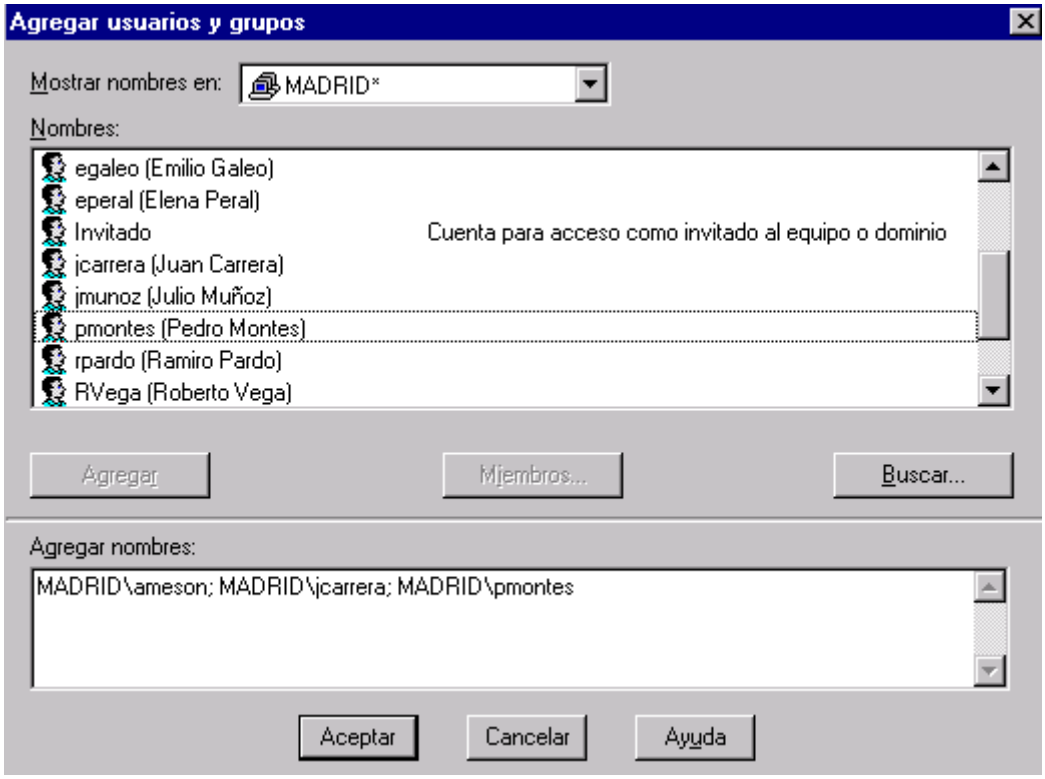


Figura 99. Agregando usuarios a un grupo de Windows NT.

Después de aceptar esta ventana, se mostrará el grupo con los usuarios que acabamos de añadir. Ver Figura 100.

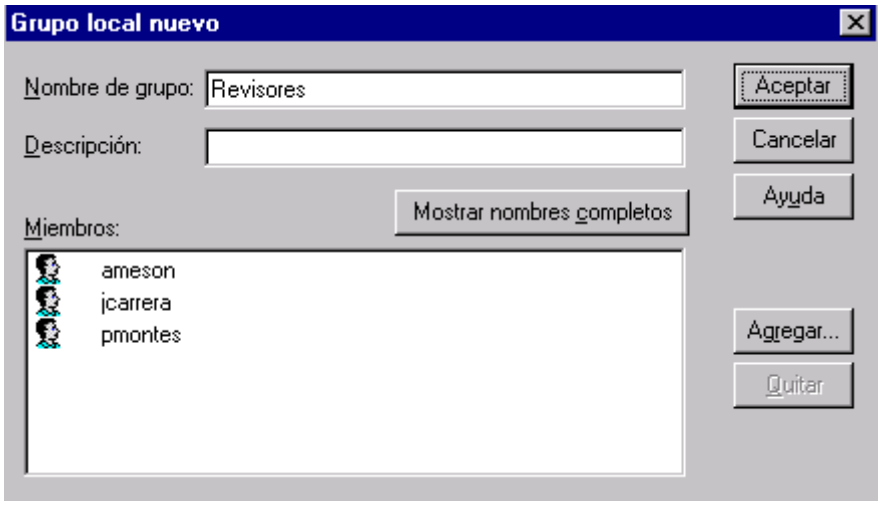


Figura 100. Nuevo grupo de Windows NT, con sus usuarios.

Creación de un inicio de sesión en SQL Server para el grupo de Windows NT

Ahora pasaremos a SQL Server y crearemos un inicio de sesión con el nombre del grupo que acabamos de crear en NT, según vemos en la Figura 101.

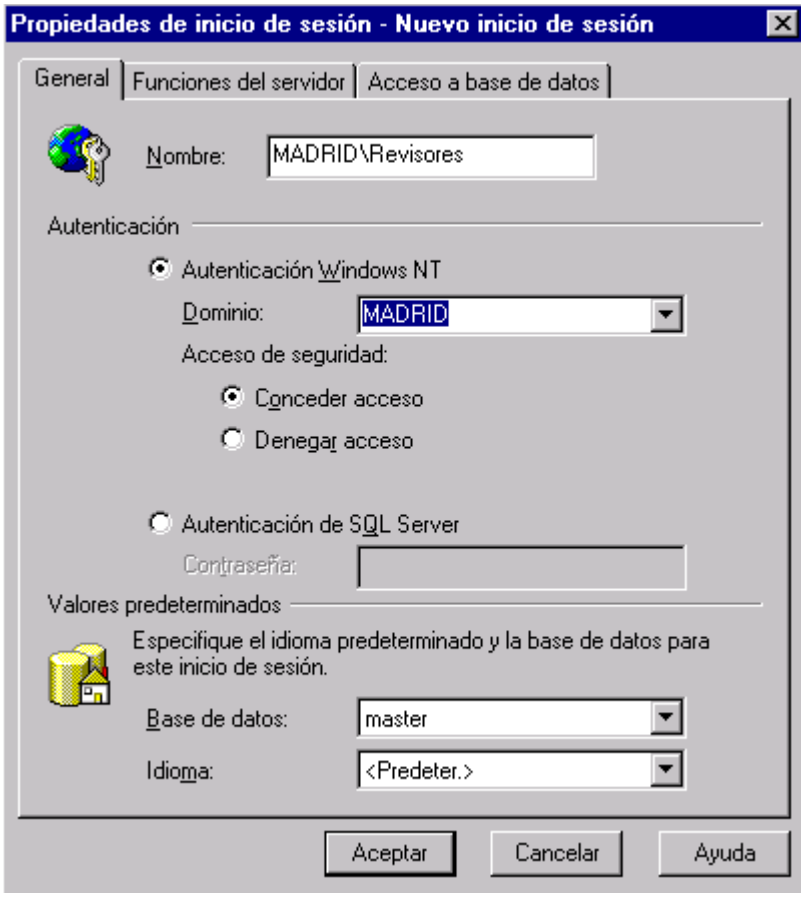


Figura 101. Nuevo inicio de sesión en SQL Server, para un grupo de Windows NT.

Finalmente, ya sólo queda que el lector realice conexiones al sistema con los nombres de los usuarios del grupo Revisores, y conecte con SQL Server para comprobar cómo se concede acceso al servidor. De esta forma y como indicábamos al comienzo de este apartado, nos hemos ahorrado una considerable cantidad de trabajo, creando un inicio de sesión en SQL Server que proporciona conexión a tres usuarios.

Modos de autenticación

Cuando SQL Server se ejecuta en Windows NT, la autenticación del usuario al intentar acceder a una base de datos puede especificarse en dos modos.

Modo de autenticación de SQL Server

El usuario sólo puede indicar las claves de acceso del sistema operativo, no puede utilizar las de SQL Server.

Modo de autenticación mixto

Cuando un usuario conecta con un servidor de datos mediante un inicio de sesión de SQL Server, se emplea la autenticación de SQL Server. Sin embargo, si no se utiliza el inicio de sesión, SQL Server toma los valores de autenticación que el usuario empleó para acceder al sistema operativo, es decir, la autenticación de Windows NT.

Para establecer uno de estos modos, debemos hacer clic con el botón derecho sobre un servidor SQL Server, y seleccionar la opción de menú contextual *Propiedades*. En dicha ventana de propiedades, pulsaremos sobre la pestaña *Seguridad*, y en el apartado del mismo nombre haremos clic sobre una de las opciones de autenticación que se muestran en la siguiente imagen: *SQL Server y Windows NT* o *Sólo Windows NT*. Ver Figura 102.

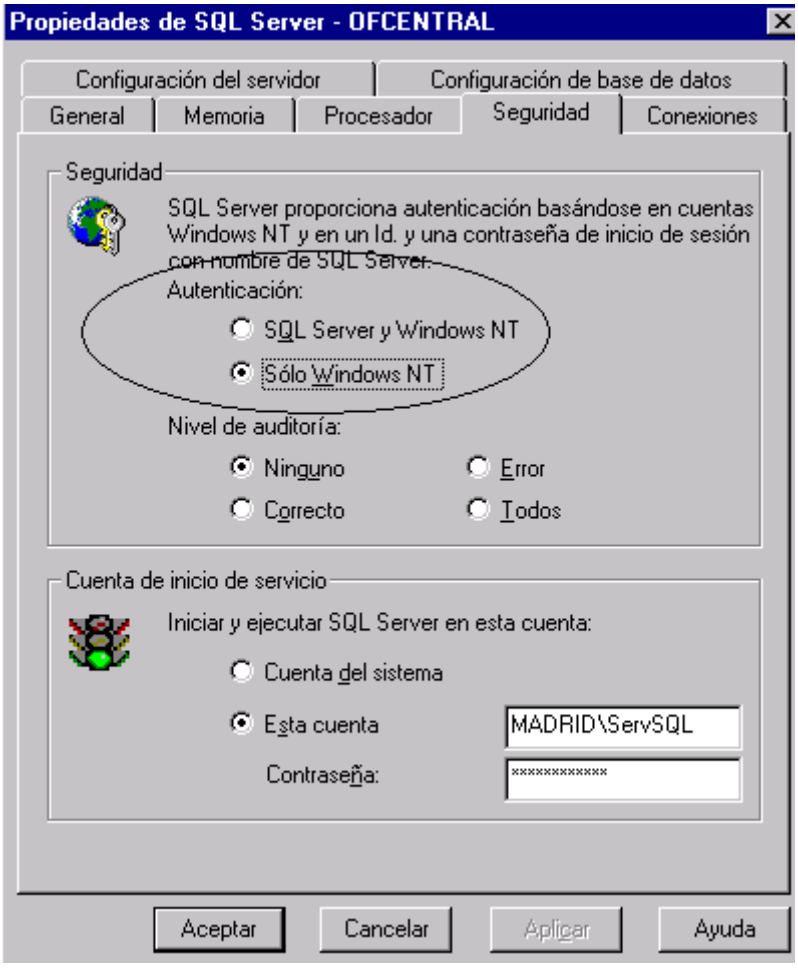


Figura 102

11

Gestión de la seguridad. Usuarios

Modificación de inicios de sesión

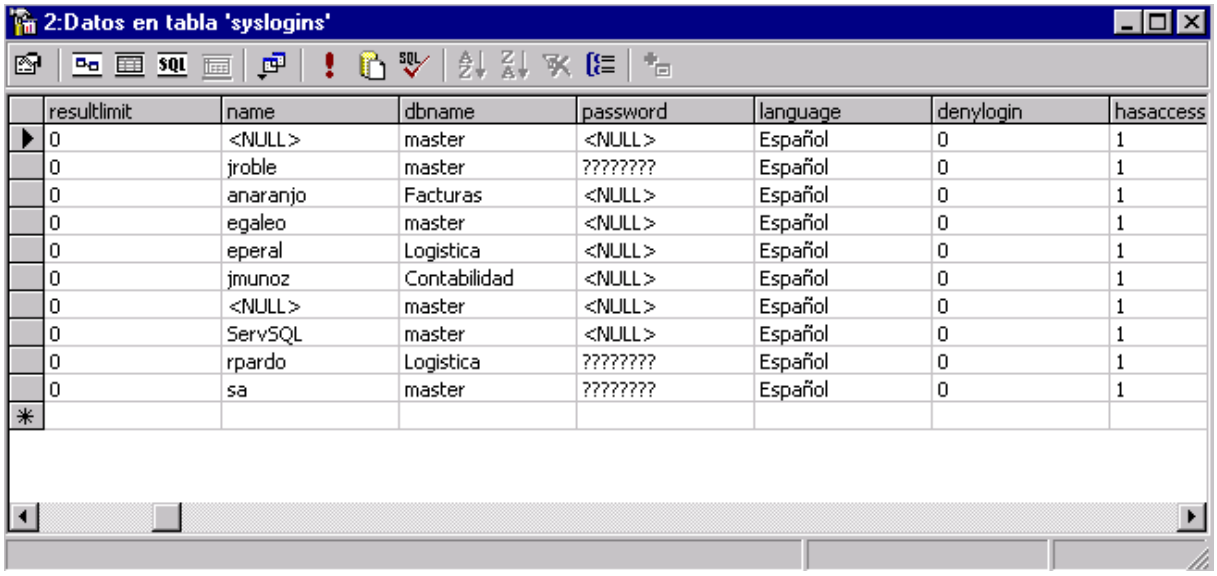
Cambiar los valores de un inicio de sesión es una tarea bien sencilla, una vez situados dentro del Administrador corporativo en el elemento *Inicios de sesión*, haremos doble clic en el identificador que necesitemos, abriéndose su ventana de propiedades, en la que cambiaremos los valores oportunos y pulsaremos Aceptar para grabarlos.

Para borrar un inicio de sesión, simplemente haremos clic en el mismo, pulsando la tecla [SUPR].

Syslogins. Información sobre inicios de sesión

Cada vez que se añade un nuevo inicio de sesión a un servidor SQL Server, queda registrado internamente en la base de datos master. Dicha base de datos, dispone de una vista llamada Syslogins, que contiene la información sobre todos los usuarios del servidor. A continuación, la Figura 103 muestra dicha vista, correspondiente al servidor en donde se han realizado estas pruebas.

Es importante hacer notar al lector, que en la mayoría de la documentación se hace referencia a este elemento como *la tabla Syslogins,* lo cual puede llevar a confusión, ya que realmente se trata de una vista. Esperamos que esta aclaración ahorre al lector tiempo de búsqueda de este elemento en la base de datos master.



resultlimit	name	dbname	password	language	denylogin	hasaccess
0	<NULL>	master	<NULL>	Español	0	1
0	jroble	master	????????	Español	0	1
0	anaranjo	Facturas	<NULL>	Español	0	1
0	egaleo	master	<NULL>	Español	0	1
0	eperal	Logistica	<NULL>	Español	0	1
0	jmunoz	Contabilidad	<NULL>	Español	0	1
0	<NULL>	master	<NULL>	Español	0	1
0	ServSQL	master	<NULL>	Español	0	1
0	rpardo	Logistica	????????	Español	0	1
0	sa	master	????????	Español	0	1

Figura 103. Vista del sistema syslogins.

Inicios de sesión predeterminados

SQL Server dispone de dos inicios de sesión orientados a los administradores del servidor, que tienen pleno acceso sobre todos los elementos de las bases de datos:

- BUILTIN\Administradores. Esta cuenta permite a los administradores de Windows NT todos los derechos sobre SQL Server.
- sa. Al igual que la anterior, esta cuenta tiene pleno derecho sobre SQL Server, permitiendo a un usuario conectado bajo la misma, manipular cualquier elemento del servidor.

Manipulación de inicios de sesión mediante código

SQL Server proporciona un conjunto de instrucciones que nos permitirán la creación, modificación y borrado de inicios de sesión, de forma que este tipo de operaciones pueda ser también implementada en aplicaciones que realicen operaciones de administración o ejecutadas desde otras herramientas como el Analizador de consultas.

Creación de una cuenta de inicio de sesión de SQL Server

Crearemos una nueva cuenta de inicio de sesión de SQL Server ejecutando el procedimiento almacenado del sistema sp_addlogin.

```
sp_addlogin 'InicioSesión' [,'Contraseña'] [,'BDPredet']
[,'IdiomaPredet'] [,'IDSeguridad'] [,'Cifrar']
```

- InicioSesión. Nombre del inicio de sesión.
- Contraseña. Contraseña para el inicio de sesión.
- BDPredet. Base de datos por defecto, asignada al inicio.

- IdiomaPredet. Idioma por defecto para el inicio.
- IDSeguridad. Número identificativo usado internamente para reconocer al inicio.
- Cifrar. Permite codificar la contraseña cuando se graba en las tablas del sistema. Los valores posibles se muestran en la Tabla 6.

Valor	Descripción
NULL	Valor predeterminado. La contraseña se cifra.
skip_ecryption	La contraseña no se cifra.
skip_ecryption_old	La contraseña no se cifra ya que fue cifrada en una versión anterior de SQL Server.

Tabla 6. Tipos de cifrado para una tabla, en el procedimiento almacenado sp_addlogin.

El Código fuente 23 nos muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_addlogin 'gamador','adminconta'
```

Código fuente 23

Eliminación de una cuenta de inicio de sesión de SQL Server

El procedimiento almacenado del sistema sp_droplogin permite borrar una cuenta de inicio de sesión de SQL Server.

```
sp_droplogin 'InicioSesión'
```

- InicioSesión. Nombre del inicio de sesión.

El Código fuente 24 muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_droplogin 'gamador'
```

Código fuente 24

Conceder permiso a un usuario de Windows NT para conectar con SQL Server

El procedimiento almacenado del sistema sp_grantlogin, permite que una cuenta de usuario o grupo de Windows NT, efectúe una conexión con SQL Server empleando autenticación de Windows NT.

```
sp_grantlogin 'InicioSesión'
```

- InicioSesión. Nombre del usuario o grupo de Windows NT. Debe tener el formato Dominio\Usuario.

El Código fuente 25 muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_grantlogin 'MADRID\RVega'
```

Código fuente 25

Impedir que un usuario de Windows NT conecte con SQL Server

Podemos denegar la conexión a SQL Server para un usuario de Windows NT utilizando el procedimiento almacenado del sistema sp_denylogin. Este procedimiento tiene la característica de que si el usuario no existe en SQL Server, se añade automáticamente, aunque sin el permiso de conexión.

```
sp_denylogin 'InicioSesión'
```

- InicioSesión. Nombre del usuario o grupo de Windows NT. Debe tener el formato Dominio\Usuario.

El Código fuente 26 muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_denylogin 'MADRID\RVega'
```

Código fuente 26

Borrar a un usuario de Windows NT como inicio de sesión de SQL Server

El procedimiento almacenado del sistema sp_revokelogin, elimina del registro de inicios de sesión de SQL Server al usuario de Windows NT pasado como parámetro. Sin embargo, si este usuario pertenece a un grupo que tiene acceso a SQL Server, todavía podrá conectar con el servidor de datos.

```
sp_revokelogin 'InicioSesión'
```

- InicioSesión. Nombre del usuario o grupo de Windows NT. Debe tener el formato Dominio\Usuario.

El Código fuente 27 muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_revokelogin 'MADRID\RVega'
```

Código fuente 27

Cuando se elimina una cuenta de Windows NT, no se elimina su inicio de sesión asociado (si es que existe) en SQL Server, evitando así la existencia de objetos huérfanos. Los pasos recomendados en este caso son: eliminar en primer lugar la cuenta de Windows NT, de esta forma se deshabilita el acceso desde la red. Posteriormente se eliminará el inicio de sesión de SQL Server.

Usuarios de la base de datos

Después de crear los inicios de sesión necesarios para conectar con un servidor SQL Server, debemos especificar los usuarios para cada base de datos que contiene el servidor. Un usuario es aquella figura dentro de SQL Server, que accede a una base de datos específica después de haber establecido conexión con un servidor de datos. Seguidamente se muestran las diferentes formas de crear un usuario para una base de datos.

Creación de un usuario junto al inicio de sesión.

Cuando se crea un inicio de sesión en SQL Server, podemos establecer que el usuario de dicho inicio de sesión pueda conectarse de forma predeterminada a una de las bases de datos del servidor, como veremos a continuación.

Vamos a crear un nuevo inicio de sesión con el nombre vabril y autenticación de SQL Server. En la ventana de creación del inicio de sesión, después de escribir el nombre y contraseña, seleccionaremos en la lista *Base de datos* una distinta a master (en este caso Pruebas), que se convertirá en la predeterminada. Ver Figura 104.

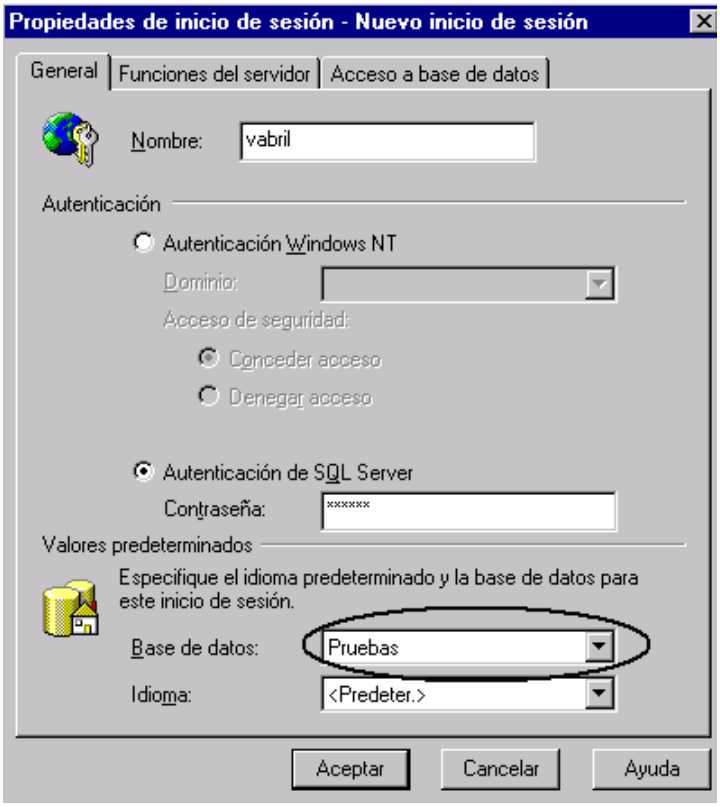


Figura 104. Base de datos predeterminada en un inicio de sesión.

A continuación, haremos clic en la pestaña *Acceso a base de datos* y marcaremos la base de datos Pruebas, de forma que se creará un usuario con el mismo nombre del inicio de sesión para dicha base de datos. Para el nombre de usuario se utiliza por defecto el mismo que el inicio de sesión, pero podemos cambiarlo por cualquier otro, simplemente haciendo clic en él y escribiendo el nuevo. Ver Figura 105.

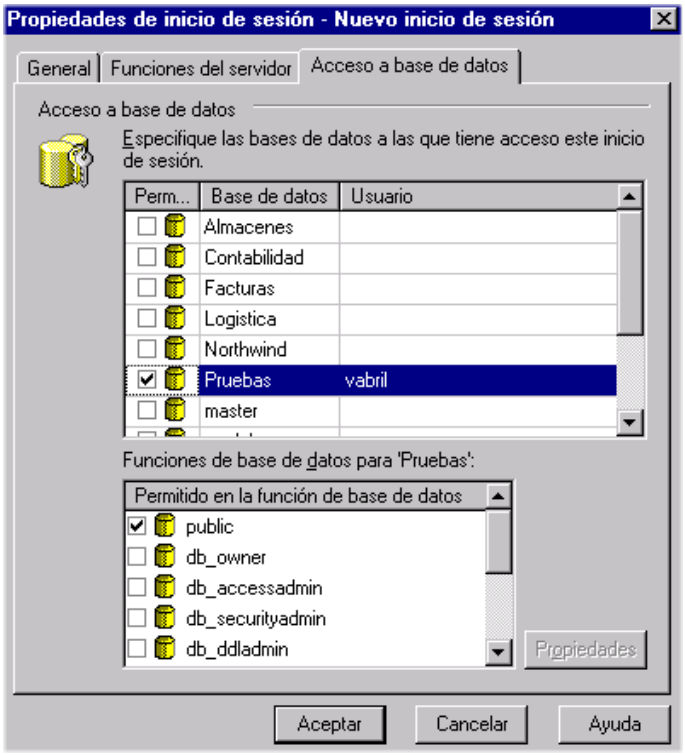


Figura 105. Establecimiento de las bases de datos a las que accederá un inicio de sesión.

Como podemos comprobar, es posible establecer el acceso a más de una base de datos para el inicio de sesión. Si conectamos con SQL Server utilizando el inicio de sesión que acabamos de crear y abrimos la base de datos Pruebas, podremos acceder por ejemplo al elemento Tablas sin producirse errores, puesto que SQL Server reconoce al usuario y le permite acceder a los componentes de la base de datos. Sin embargo, un usuario tiene por defecto un acceso restringido, si intenta ver el contenido de la tabla Clientes en la base de datos Pruebas, se producirá el error mostrado en la Figura 106.

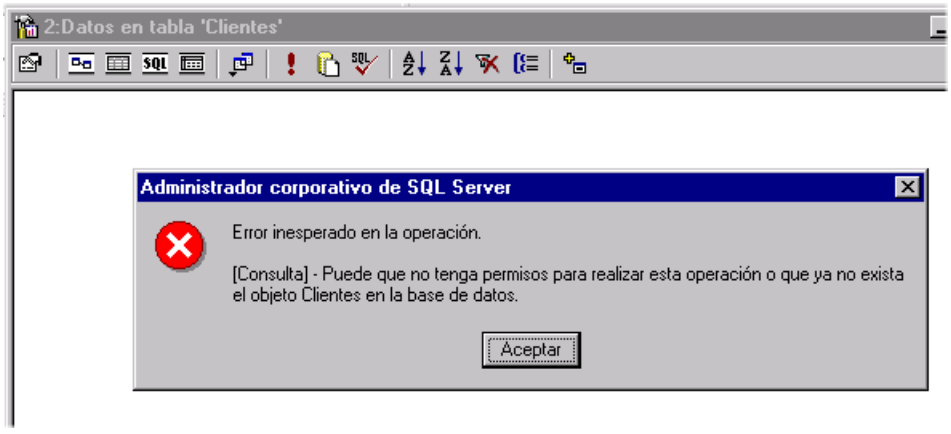


Figura 106. Error de acceso a una tabla, de un usuario.

Como indica el mensaje, el usuario no dispone de los permisos necesarios para acceder a este objeto (la tabla) de la base de datos. Los permisos serán tratados posteriormente.

Creación de un usuario desde la base de datos

Para crear un usuario desde una base de datos en el Administrador corporativo, abriremos la base de datos requerida y haremos clic con el botón derecho en el elemento Usuarios, seleccionando la opción del menú contextual *Nuevo usuario de base de datos*, que nos mostrará la ventana de creación de usuarios.

Abriremos la lista *Nombre de inicio de sesión* y elegiremos uno de los inicios disponibles. El nombre de usuario será por defecto el mismo que el inicio de sesión, aunque podremos cambiarlo por el que necesitemos.

La Figura 107, muestra la creación para una base de datos de un nuevo usuario, creado a partir de un inicio de sesión basado en autenticación de Windows NT.

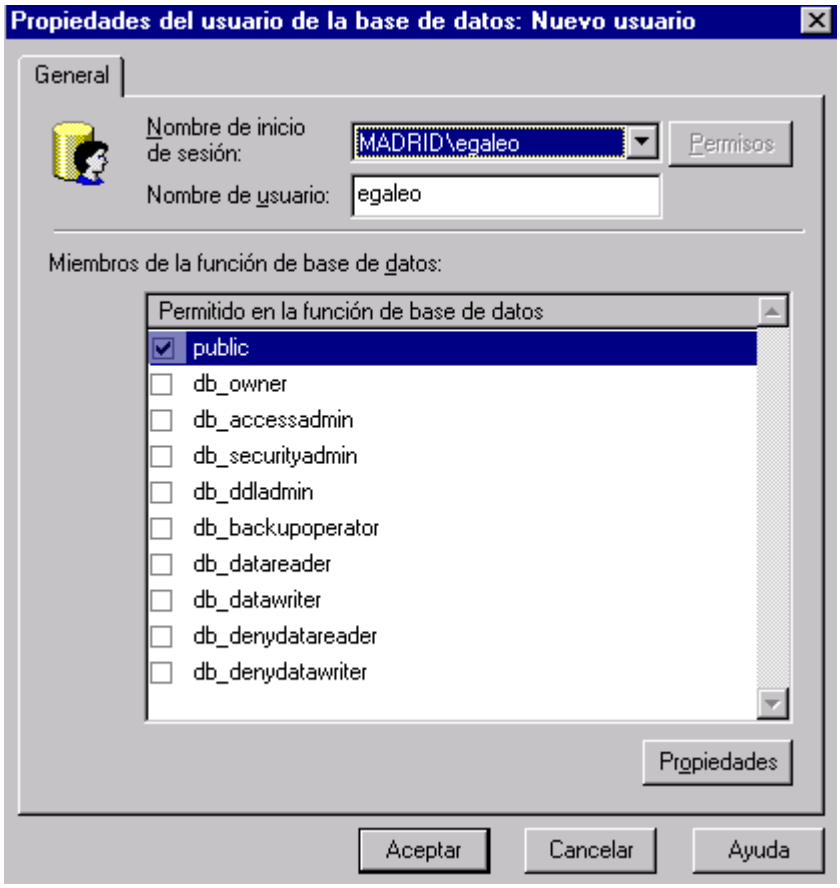


Figura 107. Creación de un usuario desde una base de datos.

Después de pulsar Aceptar para grabar el nuevo usuario, podremos ver en el panel derecho del Administrador corporativo todos los usuarios definidos para la base de datos. Ver Figura 108.

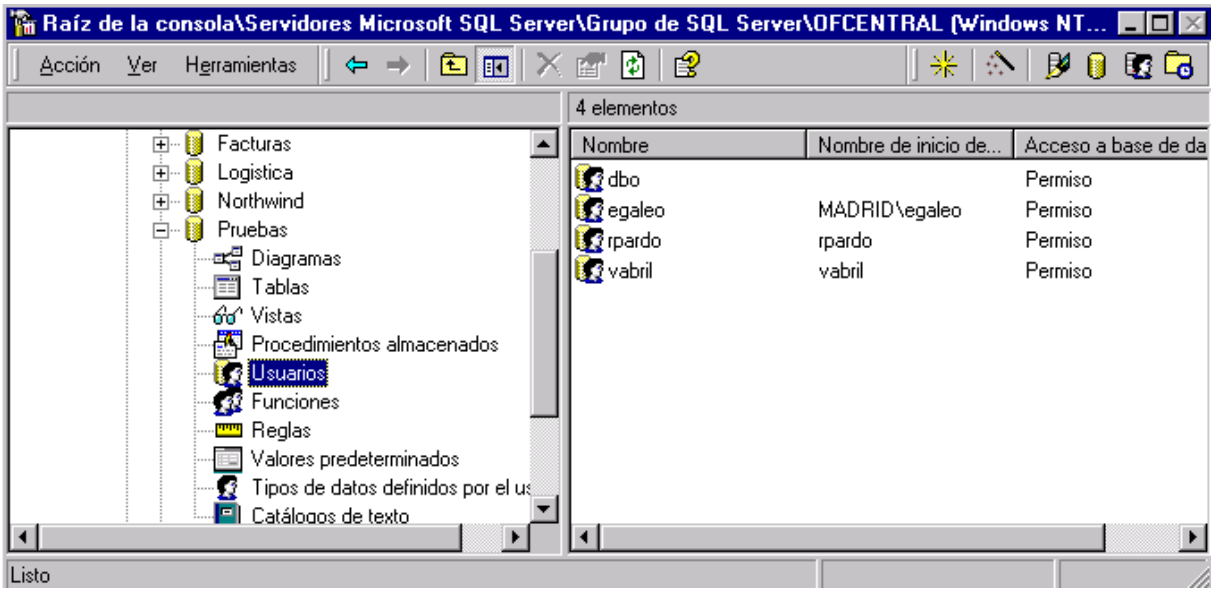


Figura 108. Lista de usuarios para una base de datos.

Para modificar las propiedades de un usuario, debemos hacer doble clic sobre el mismo en la lista de usuarios de la base de datos, y cambiar los valores en la ventana de propiedades del usuario.

Para eliminar un usuario, simplemente lo seleccionaremos y pulsaremos el botón [SUPR].

Manejo de un usuario desde código

Los siguientes procedimientos almacenados de SQL Server, nos permiten realizar operaciones sobre usuarios sin necesidad de emplear el Administrador corporativo.

Crear un usuario

Utilizaremos el procedimiento almacenado sp_grantdbaccess.

```
sp_grantdbaccess 'InicioSesión' [ , 'NombreBD' ]
```

- InicioSesión. Nombre del inicio de sesión del que se va a crear un usuario.
- NombreBD. Nombre de usuario que tendrá en la base de datos. Si no se especifica, se utiliza el valor de InicioSesión.

El Código fuente 28 muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_grantdbaccess 'MADRID\eperal', 'gestor'
```

Código fuente 28

Eliminar un usuario

Utilizaremos el procedimiento almacenado sp_revokedbaccess.

```
sp_revokedbaccess 'NombreUsuario'
```

- NombreUsuario. Nombre del usuario que se va a eliminar.

El Código fuente 29 muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_revokedbaccess 'gestor'
```

Código fuente 29

Cuentas de usuario especiales

dbo

Esta cuenta especial de SQL Server, está asociada a los administradores del sistema e inicio de sesión sa. Los objetos creados por estos usuarios, pertenecerán automáticamente a dbo.

guest

Esta cuenta permite conectar con SQL Server sólo mediante un inicio de sesión, sin disponer de una cuenta de usuario. Cuando ocurre esta situación, el inicio de sesión ocupa la identidad del usuario guest para acceder a las bases de datos que contengan definido este tipo de usuario.

12

Gestión de la seguridad. Funciones y permisos

Funciones y permisos

Una función o rol, permite agrupar usuarios de una base de datos con idéntico perfil (que deban realizar las mismas acciones), de forma similar al modo de operación de los usuarios y grupos en Windows NT.

Un permiso sirve para establecer qué acciones y sobre qué objetos de una base de datos, puede actuar una función o usuario de una base de datos.

Ambos elementos se combinan, constituyendo el nivel de mayor profundidad que debe franquear el usuario para obtener y manipular la información de una base de datos.

Funciones fijas de servidor

Las funciones fijas de servidor, son un tipo especial de funciones, con ámbito de acción sobre todo un servidor SQL Server y no sobre una sola base de datos. La Tabla 7 muestra una relación de estas funciones junto a una breve descripción de su cometido.

Para ver estas funciones, debemos abrir la carpeta Seguridad de SQL Server, y hacer clic en el elemento *Funciones del servidor*, mostrándose en el panel derecho del Administrador corporativo, como vemos en la Figura 109.

Función	Descripción
dbcreator	Permite crear y modificar bases de datos.
diskadmin	Permite administrar los archivos del disco.
processadmin	Permite administrar los procesos ejecutados en el servidor.
securityadmin	Permite administrar los inicios de sesión, permisos CREATE DATABASE y leer los registros de errores.
serveradmin	Permite configurar y apagar el servidor.
setupadmin	Permite administrar servidores vinculados y procedimientos de inicio.
sysadmin	Permite pleno acceso sobre SQL Server.

Tabla 7. Funciones fijas de servidor.

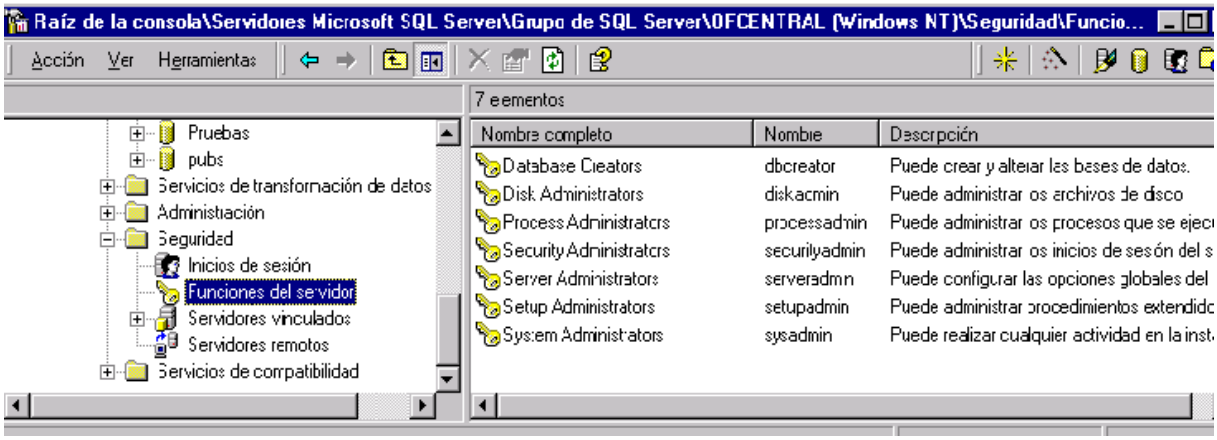


Figura 109. Funciones fijas del servidor.

Agregar un inicio de sesión a una función del servidor mediante el Administrador corporativo

Un nuevo inicio de sesión o usuario, no dispone por defecto de ninguna capacidad operativa sobre el servidor. No puede configurar el servidor, ni manipular los archivos de datos que lo componen o crear bases de datos. Veamos a continuación, un ejemplo de como asignar una función del servidor a un usuario, para que pueda crear bases de datos.

En primer lugar, debemos abrir la carpeta Seguridad del servidor, y hacer clic sobre el elemento *Funciones del servidor*. Después haremos doble clic sobre la función *Database Creators*, que visualizará una ventana en la que podremos añadir nuevos usuarios a esta función y ver los existentes. Al pulsar Agregar se mostrarán los inicios de sesión disponibles; seleccionaremos uno o varios y pulsaremos Aceptar. En este ejemplo, véase la Figura 110, hemos agregado el inicio de sesión jroble, de forma que cuando este usuario conecte con el servidor, tendrá habilitada la opción para crear nuevas bases de datos.

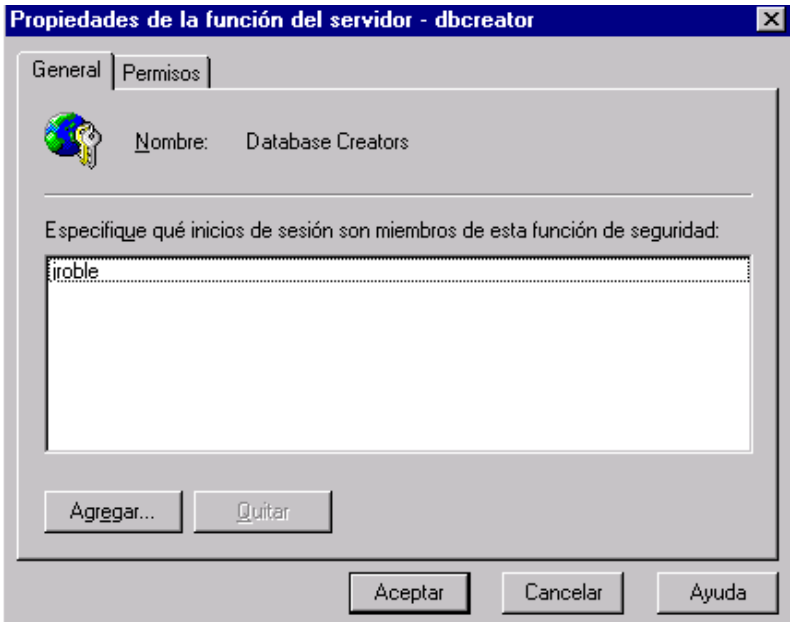


Figura 110. Función del servidor Database Creators, asignada a un inicio de sesión.

Otra forma de conseguir este mismo resultado consiste en abrir los inicios de sesión del servidor y hacer doble clic en el inicio de sesión que vayamos a asignar; en la ventana de propiedades de dicho inicio de sesión haremos clic sobre la pestaña *Funciones del servidor* y marcaremos las funciones que deseemos asignar al inicio, como se muestra en la Figura 111.

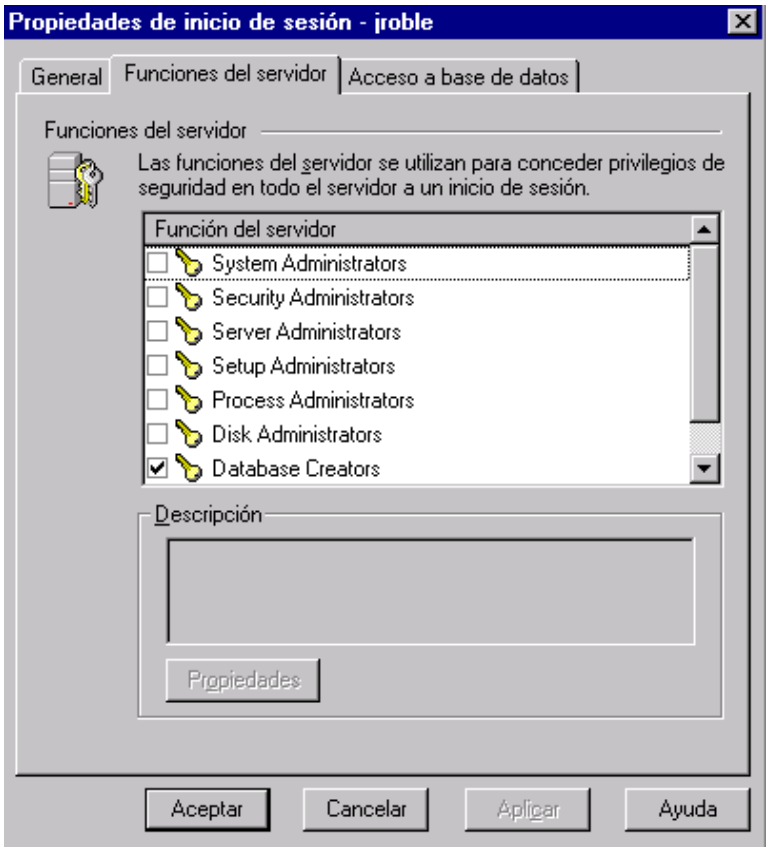


Figura 111. Asignación de funciones del servidor en la ventana de propiedades del inicio de sesión.

Agregar un inicio de sesión a una función del servidor mediante código

Si no podemos utilizar el Administrador corporativo, disponemos del siguiente procedimiento almacenado del sistema para asignar funciones fijas del servidor.

```
sp_addsrvrolemember 'InicioSesión' , 'FunciónServidor'
```

- InicioSesión. Nombre del inicio de sesión al que se asigna una función
- FunciónServidor. Nombre de la función fija del servidor. Consulte el lector la tabla de funciones del servidor al comienzo de este apartado.

El Código fuente 30, nos muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_addsrvrolemember 'MADRID\eperal', 'dbcreator'
```

Código fuente 30

Funciones fijas de base de datos

Las funciones fijas de base de datos, son también un tipo particular de funciones, que permiten establecer las operaciones autorizadas al nivel de una base de datos del servidor.

La Tabla 8, muestra una relación de estas funciones junto a una breve descripción de su cometido.

Función	**Descripción**
db_accessadmin	Permite agregar y quitar usuarios y grupos a la base de datos.
db_backupoperator	Permite realizar copias de seguridad.
db_datareader	Permite leer información de la base de datos.
db_datawriter	Permite escribir información en la base de datos.
db_ddladmin	Permite manejar objetos de la base de datos.
db_denydatareader	Impide leer información de la base de datos.
db_denydatawriter	Impide escribir información en la base de datos.
db_owner	Permite realizar todas las operaciones sobre la base de datos.
db_securityadmin	Permite administrar funciones y permisos de base de datos.

public	Es la función predeterminada de los usuarios de la base de datos; si no tienen ningún permiso asignado, podrán realizar las operaciones definidas en esta función.

Tabla 8. Funciones fijas de base de datos.

Agregar un usuario a una función de base de datos mediante el Administrador corporativo

Para realizar este proceso, debemos abrir una base de datos del servidor y seleccionar el componente Funciones. A continuación, haremos doble clic sobre la función a asignar, que nos mostrará una ventana para agregar usuarios de esta base de datos a dicha función. Los usuarios podremos elegirlos pulsando el botón Agregar, que nos mostrará una lista con los disponibles. Finalmente aceptaremos la ventana, agregándose los usuarios a la función de la base de datos.

La Figura 112, muestra un ejemplo de este proceso, en el que hemos agregado para la base de datos Pruebas, al usuario jroble en la función db_datareader. A partir de ahora, cada vez que este usuario se conecte, podrá visualizar el contenido de las tablas de esta base de datos, ya que hasta este momento sólo podía ver las tablas sin acceder a su contenido.

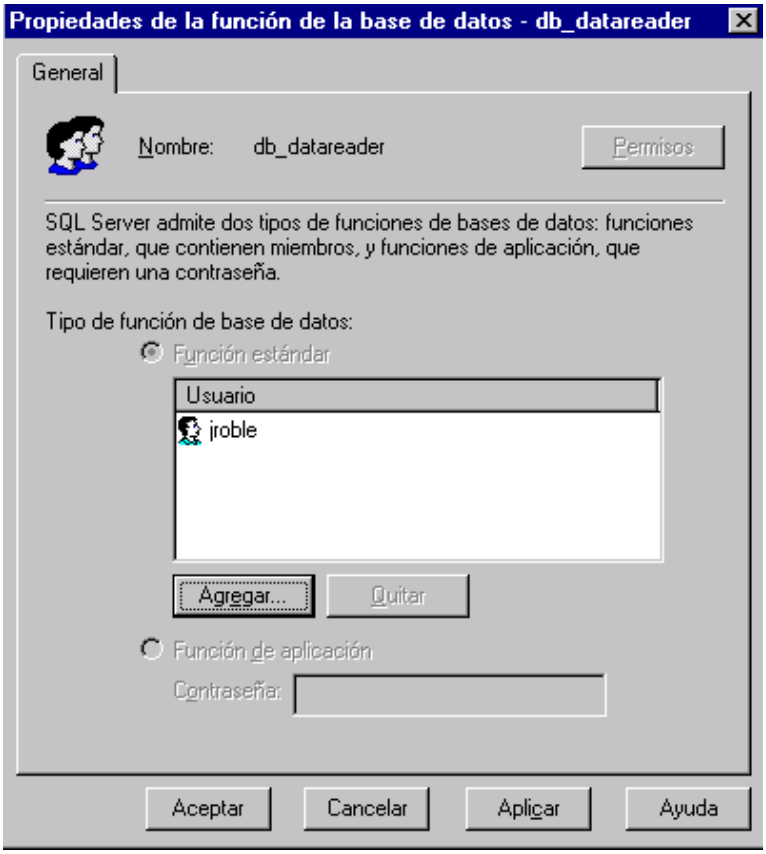


Figura 112. Asignación de funciones de base de datos a un usuario.

Existe otro medio de asignar usuarios a funciones de base de datos, consistente en seleccionar el componente Usuarios de la base de datos y hacer doble clic sobre un usuario, lo que nos mostrará las propiedades de dicho usuario en la base de datos. En este punto, marcaremos las funciones de la base

de datos a las que deseamos que pertenezca dicho usuario, como se muestra en la Figura 113. En este caso hemos conseguido el mismo resultado que en el anterior, asignar al usuario jroble a una función de una base de datos.

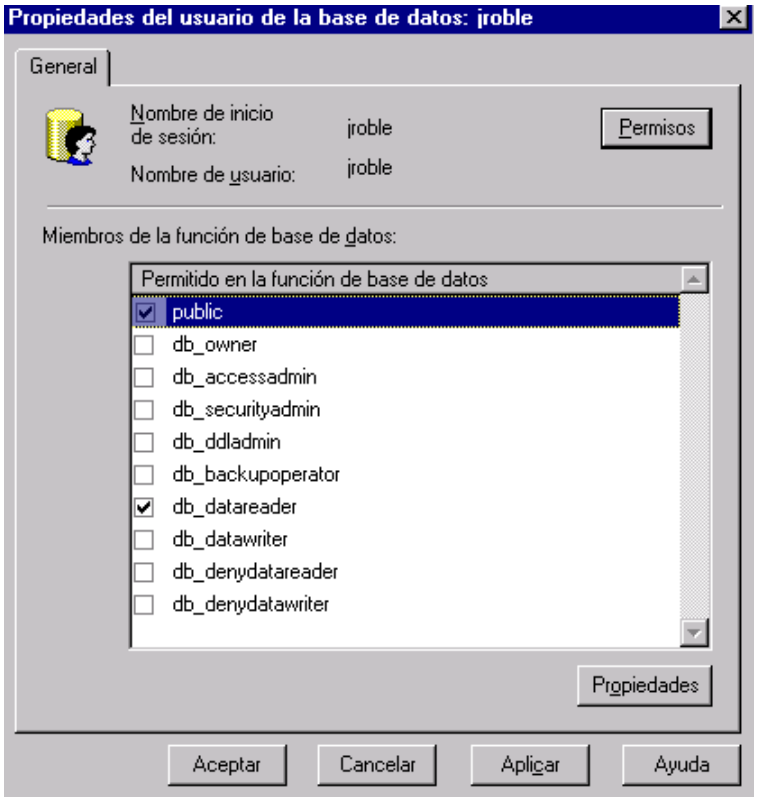


Figura 113. Asignación de funciones de base de datos a un usuario, desde las propiedades del usuario.

Agregar un usuario a una función de la base de datos mediante código

El procedimiento almacenado sp_addrolemember nos permite añadir un usuario a una función de la base de datos. Debemos tener en cuenta que este procedimiento almacenado sólo tiene efecto sobre la base de datos actual, por lo que si no estamos posicionados sobre ella deberemos emplear la instrucción USE.

```
sp_addrolemember 'FunciónBD' , 'Usuario'
```

- FunciónBD. Nombre de la función fija de base de datos. Consulte el lector la tabla de funciones de base de datos al comienzo de este apartado.

- Usuario. Nombre del usuario que se va a agregar a una base de datos.

El Código fuente 31 muestra un ejemplo del uso de este procedimiento almacenado.

```
USE Pruebas
EXEC sp_addrolemember 'db_datareader','egaleo'
```

Código fuente 31

Para comprobar si se ha añadido el usuario, podemos abrir la ventana de propiedades del usuario en la base de datos.

Eliminar un usuario de una función de la base de datos mediante código

De igual forma que hemos añadido al usuario, podemos quitarlo de la función de la base de datos usando el procedimiento almacenado sp_droprolemember.

```
sp_droprolemember 'FunciónBD' , 'Usuario'
```

- FunciónBD. Nombre de la función fija de base de datos. Consulte el lector la tabla de funciones de base de datos al comienzo de este apartado.
- Usuario. Nombre del usuario que se va a agregar a una base de datos.

El Código fuente 32 muestra un ejemplo del uso de este procedimiento almacenado.

```
USE Pruebas
EXEC sp_droprolemember 'db_datareader','egaleo'
```

Código fuente 32

Funciones de base de datos creadas por el usuario

Un usuario de SQL Server con los permisos adecuados, en nuestro caso el administrador, puede crear sus propias funciones o roles para una base de datos, configurando cada función con permisos sobre operaciones a efectuar con los objetos de la base de datos. Si tenemos un conjunto de usuarios de la base de datos, cuyo perfil se adapta a los permisos establecidos para la función, en lugar de asignar los mismos permisos de forma repetitiva a cada usuario por separado, incluiremos dichos usuarios como miembros de la función, evitando trabajo innecesario.

Creación de una función de usuario desde el Administrador corporativo

Después de abrir una base de datos del servidor, haremos clic con el botón derecho sobre su elemento Funciones y seleccionaremos del menú contextual la opción *Nueva función de base de datos*, que nos mostrará la ventana de propiedades de la función. Ver Figura 114.

Como vemos en la imagen anterior, debemos escribir el nombre de la función, e incluir los usuarios que formarán parte de ella pulsando el botón Agregar. Hemos dado a la función el nombre EscritoresPoblacion, y a continuación la configuraremos para que permita a sus miembros realizar inserciones de datos en la tabla Poblaciones situada en la base de datos Pruebas.

Como habrá observado el lector, durante el momento de creación de la función, no está habilitado el botón Permisos, para adjudicar los permisos sobre los objetos de la base de datos. Debemos pulsar

Figura 114. Ventana de creación de una función de base de datos de usuario.

Aceptar y volver a abrir esta ventana de propiedades de la función para poder acceder a los permisos, que como hemos indicado antes serán de escritura para la tabla Poblaciones, como muestra la Figura 115.

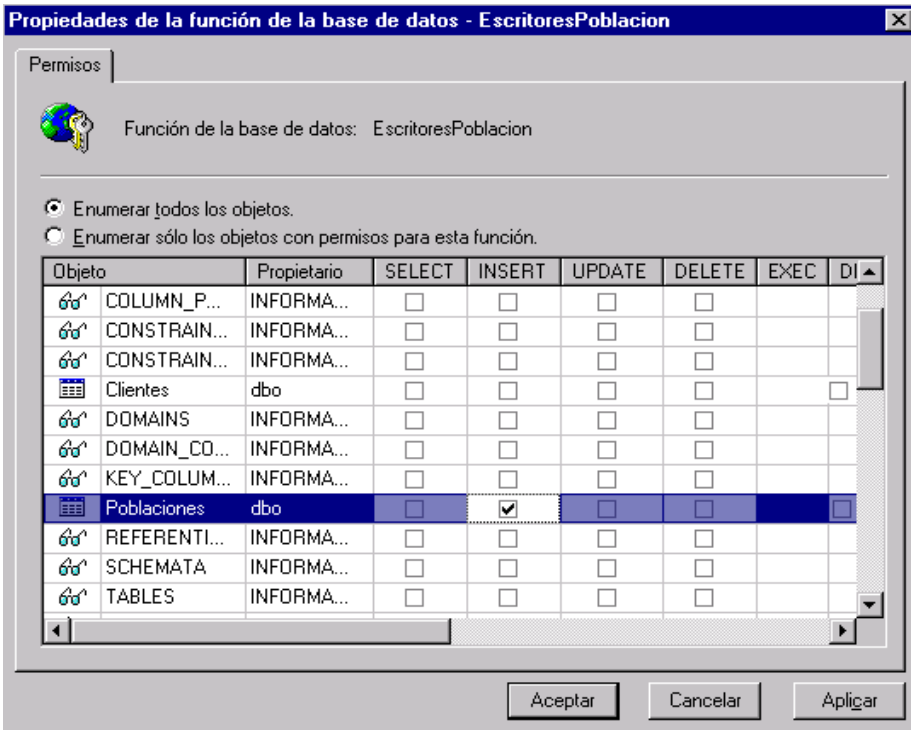


Figura 115. Asignación de permisos a una función de base de datos de usuario.

A partir de este momento, los usuarios egaleo y jroble, miembros de esta función, cuando hagan conexión con esta base de datos, podrán añadir registros a la tabla Poblaciones.

Si necesitamos eliminar una función de base de datos de usuario, debemos en primer lugar eliminar a los usuarios que forman parte de ella. Desde la ventana de propiedades de la función, seleccionaremos cada usuario y pulsaremos el botón Quitar. Por último, eliminaremos la función haciendo clic sobre ella y pulsando la tecla [SUPR], ya que no está permitido borrar una función que contenga usuarios.

Creación de una función de usuario mediante código

El procedimiento almacenado sp_addrole nos permite agregar una nueva función a la base de datos actual.

```
sp_addrole 'Función', 'Propietario'
```

- Función. Nombre de la función de usuario a crear.
- Propietario. Nombre del propietario de la función, por defecto es dbo.

El Código fuente 33 muestra un ejemplo de uso de este procedimiento almacenado.

```
USE Pruebas
EXEC sp_addrole 'LectoresPoblacion'
```

Código fuente 33

Después de crear la función, debemos agregarle usuarios mediante el procedimiento almacenado sp_addrolemember.

```
sp_addrolemember 'Función', 'Usuario'
```

- Función. Nombre de la función de usuario a la que se añade un usuario.
- Usuario. Nombre del usuario que pasa a formar parte de la función.

El Código fuente 34 muestra un ejemplo de uso de este procedimiento almacenado.

```
USE Pruebas
EXEC sp_addrolemember 'LectoresPoblacion','vabril'
```

Código fuente 34

Eliminación de una función de usuario mediante código

Como hemos indicado anteriormente, debemos eliminar en primer lugar los miembros de la función, antes de eliminar la función propiamente dicha. Para ello, aplicaremos el procedimiento almacenado sp_droprolemember sobre cada usuario de la función.

```
sp_droprolemember 'Función', 'Usuario'
```

- Función. Nombre de la función de usuario a la que se quita un usuario.
- Usuario. Nombre del usuario que se quita de la función.

El Código fuente 35 muestra un ejemplo de uso de este procedimiento almacenado.

```
USE Pruebas
EXEC sp_droprolemember 'LectoresPoblacion','vabril'
```

Código fuente 35

Una vez eliminados todos los miembros de la función, quitaremos la función de la base de datos con el procedimiento almacenado sp_droprole.

```
sp_droprole 'Función'
```

- Función. Nombre de la función de usuario que se elimina.

El Código fuente 36 muestra un ejemplo de uso de este procedimiento almacenado.

```
USE Pruebas
EXEC sp_droprole 'LectoresPoblacion'
```

Código fuente 36

Esta facilidad de manipulación de funciones de usuario, permite a un administrador de SQL Server crear funciones para diferentes departamentos de una empresa, de forma que si un usuario cambia de departamento, sólo tendrá que quitarlo de una función y añadirlo a otra, eliminando el molesto trabajo de dar y quitar permisos directamente al usuario cada vez que cambia de departamento; aparte del riesgo de asignar un permiso equivocado en alguna de estas modificaciones.

Adicionalmente, una función puede a su vez, ser miembro de otra función, facilitando enormemente su manejo y configuración.

Tipos de permisos

Un permiso se utiliza para asignar las acciones que puede realizar un usuario dentro de una base de datos. En el apartado anterior, el lector ha podido comprobar el modo más simple de uso de permisos, que es la concesión de un permiso de lectura a una función de usuario. Pero un permiso es un elemento que proporciona más funcionalidad como veremos a partir de ahora, según su estado y el objeto al que esté asignado: función o usuario.

Según la operación a realizar, existen diferentes tipos de permisos que se describen a continuación.

Instrucción

Están relacionados con la creación de bases de datos y sus distintos elementos, como puedan ser las instrucciones CREATE DATABASE, CREATE TABLE, etc.

Objeto

Estos permisos se relacionan con la manipulación de datos o los objetos que contienen dichos datos. Se dividen en los siguientes grupos.

- Permisos de tablas y vistas. Se aplican a las instrucciones SELECT, INSERT, UPDATE y DELETE.
- Permisos de columna. Se aplican a las instrucciones SELECT, UPDATE y REFERENCES.
- Permisos de procedimiento almacenado. Se aplican a la instrucción EXECUTE.

Predefinido

Son permisos que tienen de forma predeterminada las funciones fijas del servidor y los propietarios de objetos.

En el caso de un usuario que pertenezca a la función sysadmin, hereda todos los permisos de esta función, mientras que un usuario propietario de una tabla, tiene todos los permisos de manipulación la misma.

Estado de permisos

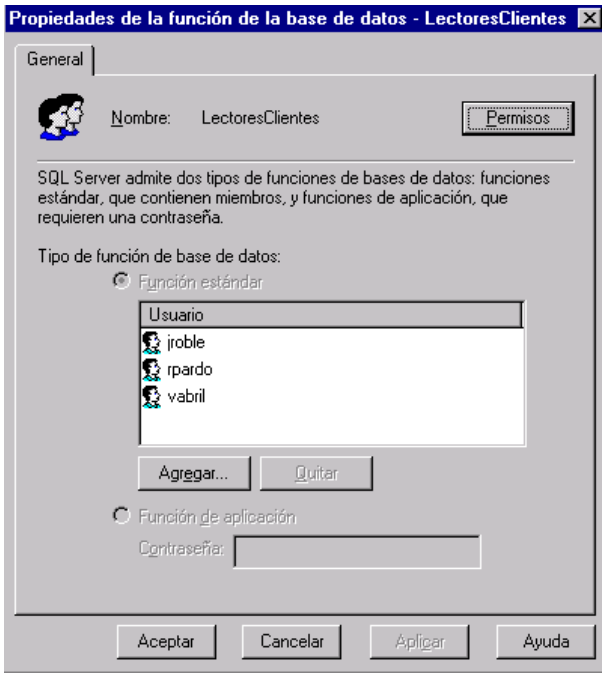


Figura 116. Función LectoresClientes para las pruebas con permisos.

Los permisos de un usuario o función pueden tener uno de los siguientes estados.

- GRANT. Permite ejecutar una operación.
- DENY. Impide la ejecución de una operación de forma directa e indirecta por pertenencia a una función.
- REVOKE. Impide la ejecución de una operación de forma directa, pero permite su ejecución indirectamente por pertenencia a una función.

Para establecer el estado de un permiso en una función o usuario, haremos doble clic en el mismo abriendo su ventana de propiedades y pulsando el botón Permisos que contiene.

Como ejemplo, crearemos en la base de datos Pruebas, una función con el nombre LectoresClientes a la que añadiremos tres usuarios: jroble, rpardo y vabril; adicionalmente utilizaremos el usuario vabril para establecer permisos aparte de la función y ver las diferencias entre estados. Ver Figura 116.

Concesión de permisos

Administrador corporativo

Como ya vimos en un apartado anterior, seleccionaremos la función LectoresClientes, y pulsando Permisos, estableceremos el permiso Select para la tabla Clientes. Ver Figura 117.

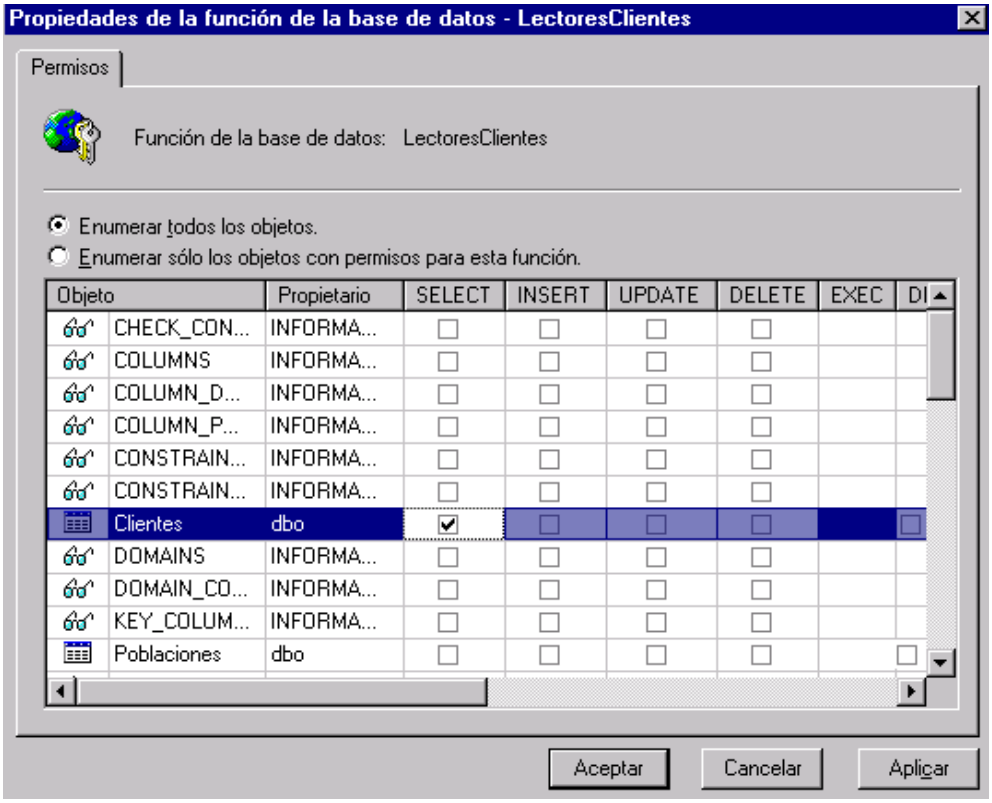


Figura 117. Concesión de permiso a la función LectoresClientes.

Instrucción GRANT

Mediante esta instrucción, concederemos el permiso de lectura sobre la tabla Clientes al usuario vabril de forma directa. Esto quiere decir que en el caso de que vabril fuese eliminado de la función LectoresClientes, podría seguir teniendo acceso a la tabla Clientes. El Código fuente 37 muestra un ejemplo de uso de esta instrucción.

```
GRANT ALL | Permiso [,...PermisoN ] ON Elemento TO Usuario [
,...UsuarioN ]
```

- ALL. Especifica que se conceden todos los permisos posibles.
- Permiso. Nombre del permiso a aplicar.
- Elemento. Indica un elemento de la base de datos sobre el que se concede el permiso: tabla, vista, procedimiento almacenado, etc.
- Usuario. Nombre de usuario o función al que se aplica el permiso.

```
GRANT SELECT ON Clientes TO vabril
```

Código fuente 37

Denegación de permisos

Administrador corporativo

En este caso, volveremos a abrir la ventana de permisos del usuario vabril y haremos clic sobre el permiso concedido anteriormente, de forma que cambiará su estado a denegado, mostrándose la marca correspondiente, que vemos en la Figura 118.

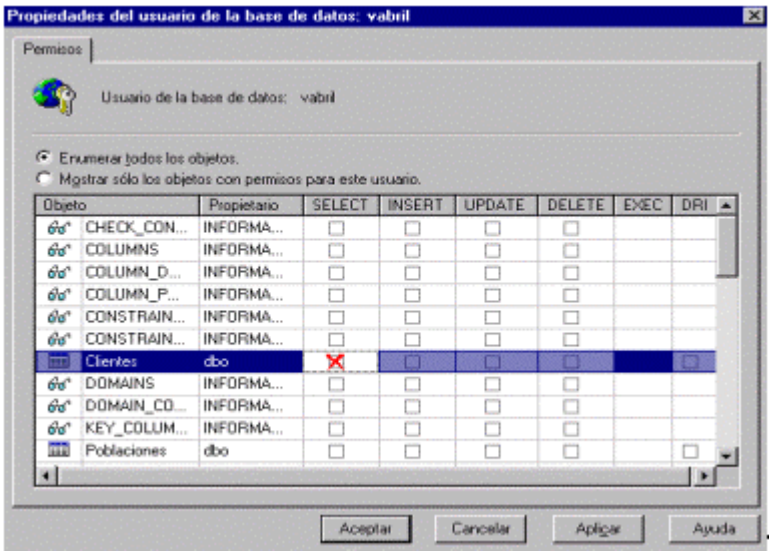


Figura 118. Denegación de permiso al usuario vabril.

A pesar de que este usuario hereda los permisos de la función LectoresClientes (por su pertenencia a la misma), que sí tiene permiso sobre la tabla Clientes, al haber denegado explícitamente este permiso al usuario, no podrá visualizar el contenido de dicha tabla.

Instrucción DENY

Si no utilizamos el Administrador corporativo, podemos emplear esta instrucción, con la que conseguiremos el mismo resultado.

```
DENY ALL | Permiso [ ,...PermisoN ] ON Elemento TO Usuario [
,...UsuarioN ]
```

- ALL. Especifica que se deniegan todos los permisos.
- Permiso. Nombre del permiso a denegar.
- Elemento. Indica un elemento de la base de datos sobre el que se deniega el permiso: tabla, vista, procedimiento almacenado, etc.
- Usuario. Nombre de usuario o función al que se deniega el permiso.

El Código fuente 38 muestra un ejemplo de uso de esta instrucción.

```
DENY SELECT ON Clientes TO vabril
```

Código fuente 38

Revocación de permisos

Administrador corporativo

Para revocar un permiso, regresaremos una vez más a los permisos del usuario vabril y volviendo a pulsar sobre el permiso SELECT de la tabla Clientes, este pasará al estado de revocado, como muestra la Figura 119.

Llegados a este punto es muy posible que el lector se pregunte lo siguiente: "Si tanto el estado de denegación como el de revocación sobre un permiso, impiden al usuario actuar sobre el objeto que se aplica, ¿qué diferencia hay entre ellos?". Expliquemos este aspecto sobre el ejemplo con el que estamos trabajando.

Antes de revocar el permiso sobre el usuario vabril, este no tenía ninguna manera de acceder a los datos de la tabla Clientes, ya que estaba explícitamente denegado. Sin embargo al revocar esta denegación, el permiso sobre el usuario vabril pasa a un estado neutro: no puede acceder por sí mismo a la tabla Clientes, pero tampoco se le prohíbe tal acceso en el caso de que esté incluido como miembro de una función de base de datos que sí tenga acceso a dicha tabla; situación que se cumple por la pertenencia de vabril a la función LectoresClientes. En conclusión, una vez que hemos revocado el permiso al usuario vabril, en adelante sí podrá consultar la tabla Clientes por el hecho de estar incluido en la función LectoresClientes.

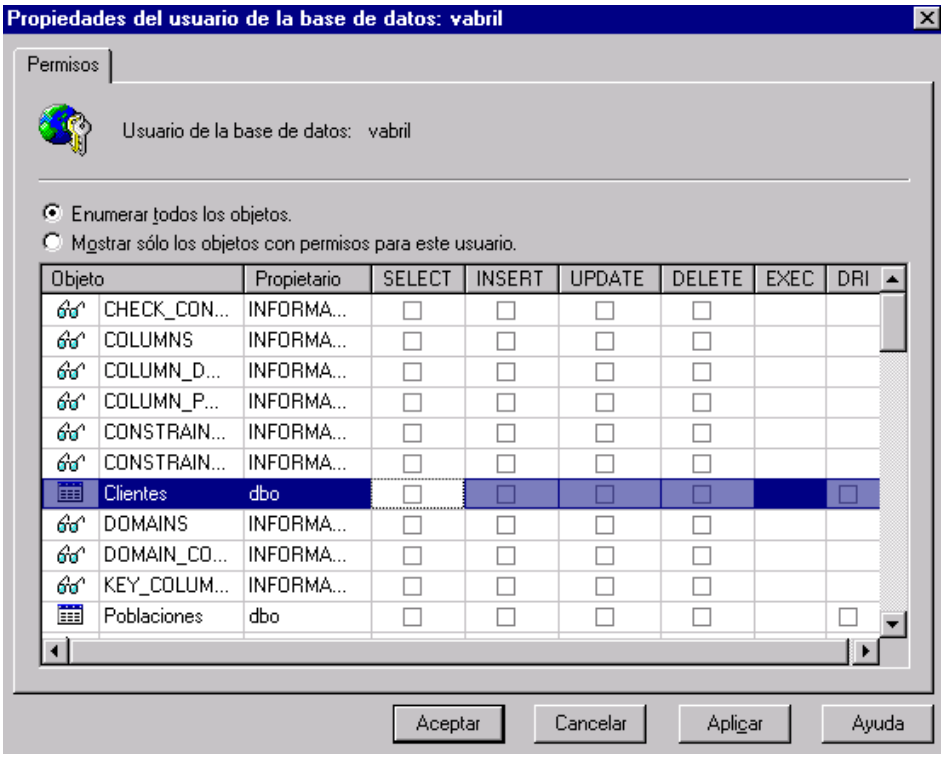


Figura 119. Revocación de permiso al usuario vabril.

Instrucción REVOKE

La instrucción REVOKE, consigue el mismo resultado del punto anterior, utilizada por ejemplo, desde el Analizador de consultas.

```
REVOKE ALL | Permiso [ ,...PermisoN ] ON Elemento TO Usuario [
,...UsuarioN ]
```

- ALL. Especifica que se revocan todos los permisos.
- Permiso. Nombre del permiso a revocar.
- Elemento. Indica un elemento de la base de datos sobre el que se revoca el permiso: tabla, vista, procedimiento almacenado, etc.
- Usuario. Nombre de usuario o función al que se revoca el permiso.

El Código fuente 39 muestra un ejemplo de uso de esta instrucción.

```
REVOKE SELECT ON Clientes TO vabril
```

Código fuente 39

Tablas del sistema con información de usuarios y permisos

Cada base de datos dispone de dos tablas: sysusers y sysprotects, que almacenan la información sobre los usuarios/funciones y permisos respectivamente. La Figura 120, muestra las filas de estas tablas correspondientes a las funciones LectoresPoblaciones y LectoresClientes para la base de datos Pruebas.

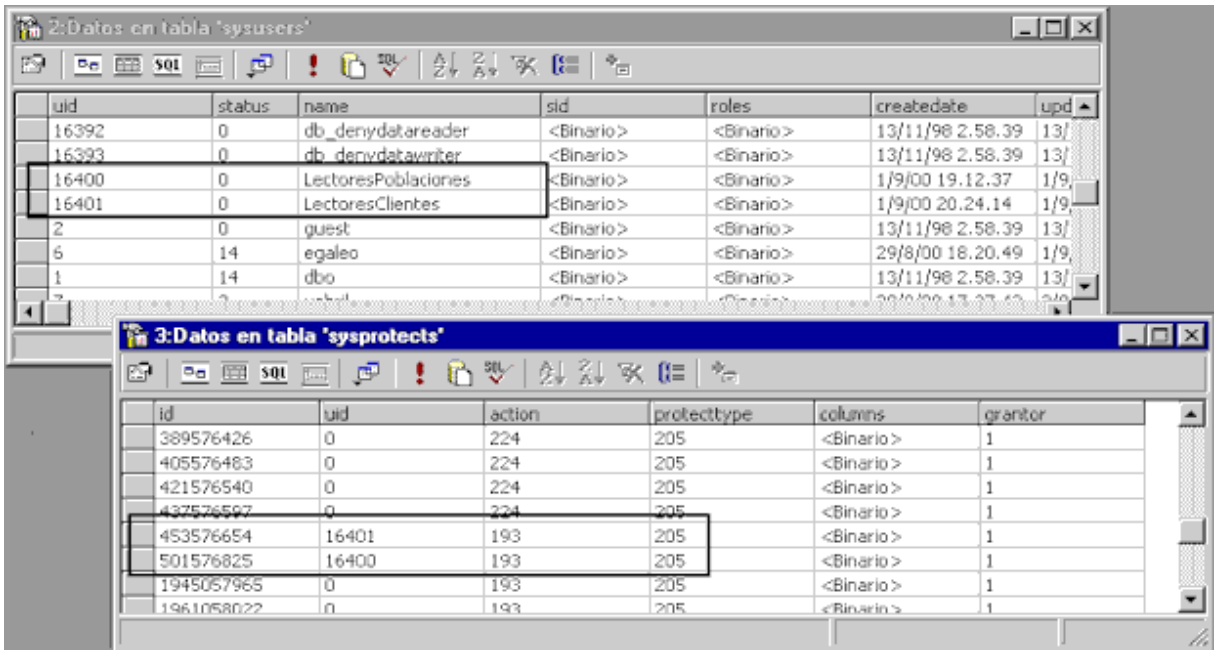


Figura 120. Tablas del sistema sysusers y sysprotects.

Funciones de aplicación

Las funciones de aplicación constituyen un potentísimo elemento en el esquema de seguridad de SQL Server 7.0. Se trata de funciones de base de datos que sólo pueden ser ejecutadas desde una aplicación de usuario. De esta forma, un programador puede desarrollar una aplicación para que los usuarios realicen operaciones sobre una tabla, activando desde dicha aplicación una función de usuario contra esa tabla, garantizando con ello, que ese va a ser el único medio que van a tener los usuarios de acceder a la tabla; no pudiendo usar una utilidad como el Analizador de consultas, Microsoft Excel, etc., para manipular la tabla.

Analicemos el siguiente escenario: disponemos en la base de datos Pruebas, de una tabla llamada Proveedores, y estamos interesados en que nuestros usuarios sólo puedan consultar su contenido a través de un programa específico destinado a ello.

La mejor opción en esta situación, es crear en la base de datos una función de aplicación, otorgándole el permiso de lectura sobre la tabla Proveedores, y desarrollar una aplicación que se encargue de activar la función, para permitir al usuario del programa consultar los datos.

Creación desde el Administrador corporativo

En primer lugar, desde el elemento Funciones de la base de datos, crearemos una nueva función, pulsando la opción *Función de aplicación* en la ventana de propiedades de la función. Un detalle

importante respecto a este tipo de funciones es que debemos asignarles una contraseña, como se muestra en la Figura 121.

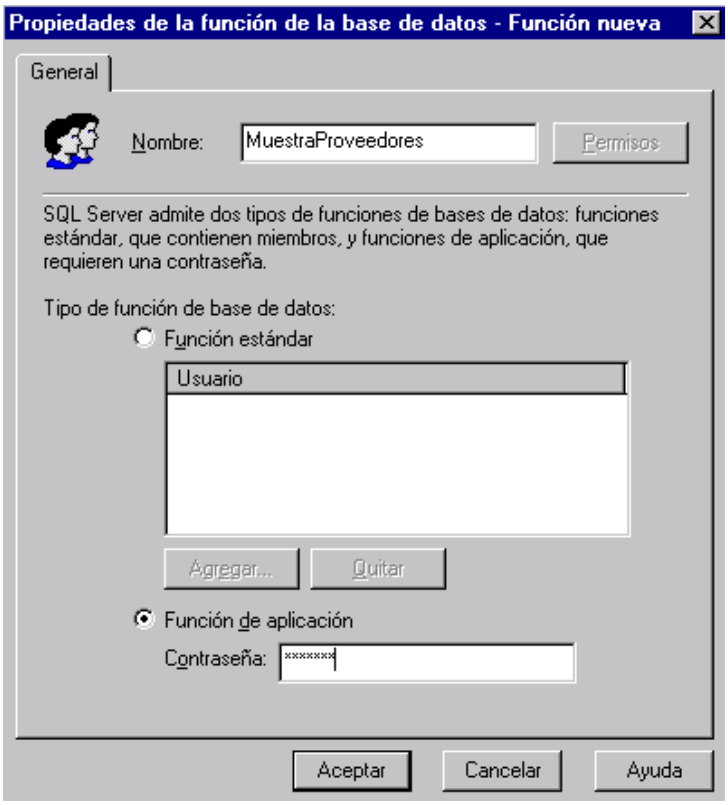


Figura 121. Creación de una función de aplicación.

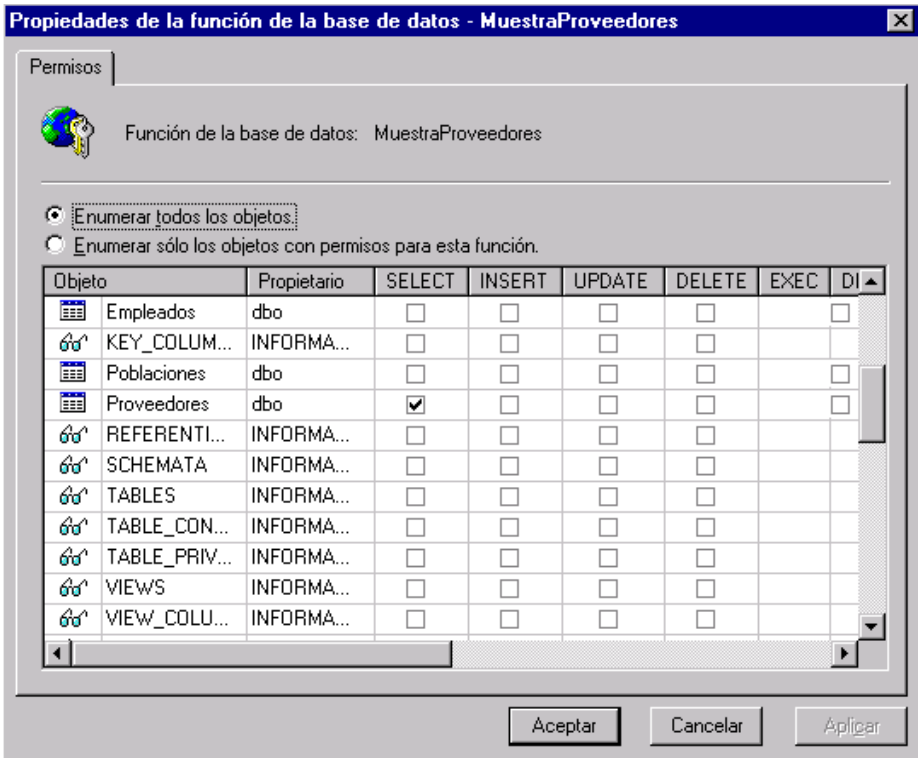


Figura 122. Concesión de permisos a la función de aplicación.

El nombre de la función es MuestraProveedores y la contraseña mercado. Es importante recordarlos, ya que deberemos utilizarlos al escribir la aplicación que active esta función. La contraseña es utilizada en la aplicación para activar la función, ya que este tipo de funciones se encuentran desactivadas.

A continuación, estableceremos permisos de lectura para esta función sobre la tabla Proveedores. Ver Figura 122.

Como habrá observado el lector, este tipo de funciones no contienen usuarios, es el usuario de la aplicación el que la activa cada vez que la ejecuta, de esta manera podemos conceder al usuario un gran número de permisos a través de este tipo de funciones, garantizando que sólo los podrá usar desde la aplicación, y evitando la asignación directa de permisos al usuario.

Creación desde código

Podemos también crear una función de aplicación mediante el procedimiento almacenado del sistema sp_addapprole.

```
sp_addapprole 'NombreFunción' , 'Contraseña'
```

- NombreFunción. Cadena con el nombre que tendrá la función.
- Contraseña. Cadena con la clave de activación de la función.

El Código fuente 40 muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_addapprole 'MuestraProveedores' , 'mercado'
```

Código fuente 40

Después tendríamos que asignarle el permiso mediante la instrucción GRANT, consiguiendo el mismo efecto que la creación desde el Administrador corporativo. Ver Código fuente 41.

```
GRANT SELECT ON Proveedores TO MuestraProveedores
```

Código fuente 41

Desarrollo de la aplicación

Como paso final, tendremos que utilizar una herramienta de desarrollo, para escribir un programa que permita al usuario acceder a los datos, activando la función de aplicación. La herramienta elegida en este caso ha sido Visual Basic, gracias a sus estupendas características de conexión con SQL Server, a través de la tecnología OLE DB, empleando objetos ADO.

El trabajo que realiza este programa es muy sencillo: crea los correspondientes objetos ADO, para realizar la conexión de un usuario de la base de datos con el servidor SQL Server, activa la función de aplicación y muestra dentro de un bucle, cada fila de la tabla. Todo ello en el procedimiento de inicio del programa, Main(). Ver el Código fuente 42.

```
Public Sub Main()

Dim lcn As ADODB.Connection
Dim lcm As ADODB.Command
Dim lparFuncionApp As ADODB.Parameter
Dim lparPassw As ADODB.Parameter
Dim lrs As ADODB.Recordset

On Error GoTo ControlErrores

' instanciar objeto Connection para realizar
' la conexión con el servidor SQL Server
Set lcn = New ADODB.Connection

' asignar cadena de conexión
lcn.ConnectionString = "Provider=SQLOLEDB.1;Password=contable;" & _
"Persist Security Info=True;User ID=jroble;" & _
"Initial Catalog=Pruebas;Data Source=OFCENTRAL"

' abrir la conexión
lcn.Open

' activar la función de aplicación
' mediante un objeto Command, que ejecute
' el procedimiento almacenado sp_setapprole,
' que es el encargado de activarla
Set lcm = New ADODB.Command
lcm.CommandType = adCmdStoredProc
lcm.CommandText = "sp_setapprole"

' el procedimiento almacenado sp_setapprole
' recibe dos parámetros: el nombre de la
' función de aplicación y su contraseña;
' ahora creamos los objetos Parameter
' que corresponden a estos parámetros y que
' pasaremos al procedimiento almacenado
Set lparFuncionApp = lcm.CreateParameter("FuncionApp", adVarChar, _
adParamInput, 18, "MuestraProveedores")

Set lparPassw = lcm.CreateParameter("Passw", adVarChar, adParamInput, 7, "mercado")

' añadir parámetros al comando
lcm.Parameters.Append lparFuncionApp
lcm.Parameters.Append lparPassw

' asignar la conexión al comando
' y ejecutar el procedimiento almacenado
Set lcm.ActiveConnection = lcn
lcm.Execute

' abrir la tabla Proveedores y mostrar
' sus filas una por una, mediante un
' objeto Recordset
Set lrs = New ADODB.Recordset
lrs.Open "Proveedores", lcn, adOpenDynamic, adLockOptimistic, adCmdTable

Do While Not lrs.EOF
MsgBox "IDProveedor: " & lrs("IDProveedor") & vbCrLf & _
"Nombre: " & lrs("Nombre") & vbCrLf & _
"Material: " & lrs("Material")

lrs.MoveNext
Loop
' eliminar los objetos de datos
' para liberar recursos
```

```
lrs.Close
Set lrs = Nothing

Set lcm = Nothing

lcn.Close
Set lcn = Nothing

Exit Sub

'*******************
ControlErrores:

MsgBox "Error: " & Err.Description, , Err.Number

End Sub
```

Código fuente 42

Para probar este ejemplo, el lector debe disponer de Visual Basic 6, crear un proyecto de tipo EXE Estándar y establecer la referencia a la librería de objetos Microsoft ActiveX Data Objects 2.1.

Cuando un usuario utiliza una aplicación en la que se activa una función de aplicación de SQL Server, dicho usuario pierde durante el tiempo que esté activa la función, todos los permisos asociados a su inicio de sesión, siendo reemplazados por los permisos de la función de aplicación. Esto es necesario, puesto que puede suceder, que el usuario tenga denegado un permiso que la función de aplicación deba tener concedido para realizar ciertas tareas.

13

Gestión de la seguridad. Estrategias

Estrategias de seguridad con vistas y procedimientos almacenados

Durante nuestro trabajo como administradores de SQL Server, puede ocurrir que al diseñar el sistema de seguridad de una base de datos, sea necesario que algunos usuarios puedan acceder u operar con la información de una tabla, pero al mismo tiempo, no deban visualizar parte de los datos contenidos en la misma. Para conseguir este objetivo, podemos emplear vistas y procedimientos almacenados, concediendo a nuestros usuarios, permisos sólo sobre dichas vistas y procedimientos, para que puedan ver únicamente la información que nos interesa.

Veámoslo con un ejemplo: tenemos una tabla llamada Empleados con las columnas IDEmpleado, Nombre, Dirección y Ciudad. La información de esta tabla ha de estar disponible para todos los usuarios, excepto las dos últimas columnas, cuya información es restringida.

Solucionaremos este inconveniente, creando una vista desde el Administrador corporativo, a la que daremos el nombre VerEmpleados, basada en una consulta sobre la tabla Empleados, que no incluirá las columnas mencionadas. Ver Figura 123.

A continuación, crearemos una función de base de datos con el nombre LectoresEmpleados, en la que otorgaremos permisos de lectura sobre la vista que acabamos de crear para los miembros de esta función. Ver Figura 124.

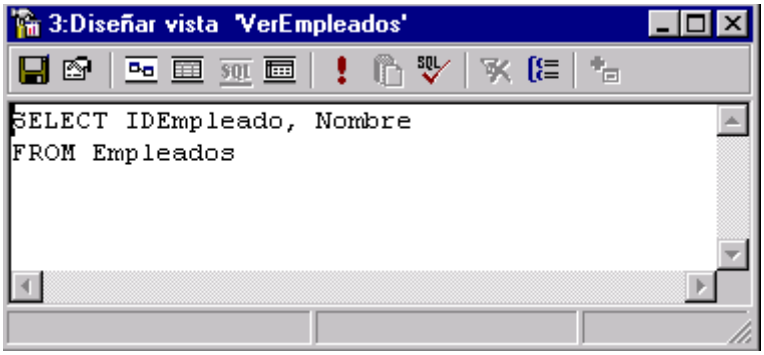


Figura 123. Creación de la vista VerEmpleados, para la tabla Empleados.

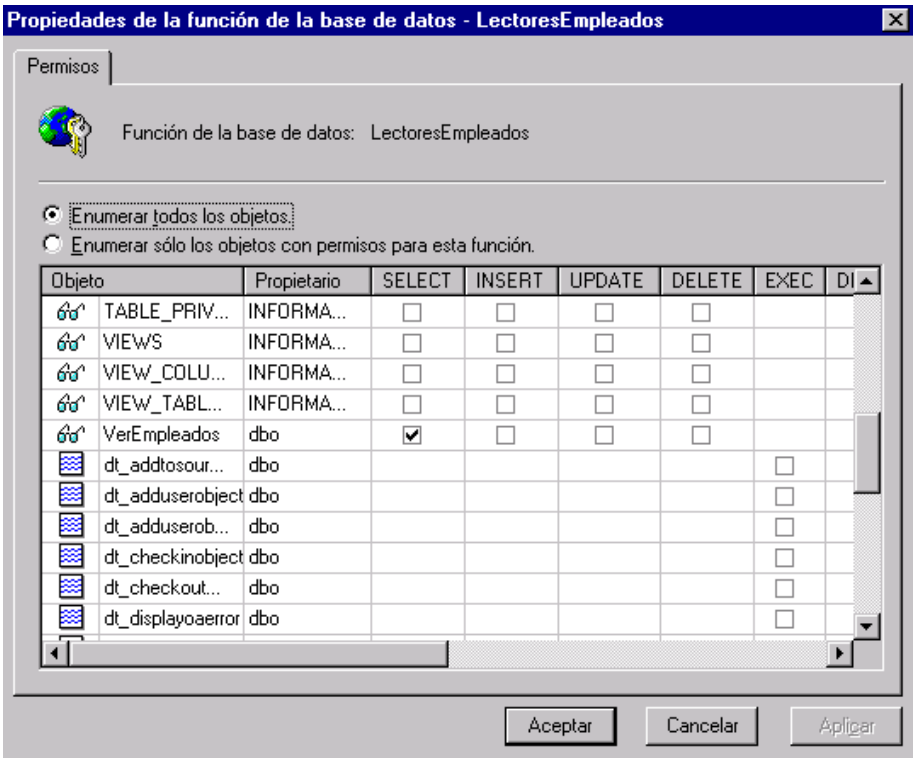


Figura 124. Asignación de permisos para la función LectoresEmpleados.

También podríamos haber asignado este permiso a cada usuario de la base de datos de forma individual, pero como se ha explicado en apartados anteriores, a través de una función, es una forma mucho más cómoda y recomendable de manejar los permisos.

A partir de este momento, los usuarios de esta función podrán acceder a la información de la tabla Empleados, pero obteniendo sólo los datos permitidos por el administrador.

Igual situación nos podemos encontrar a la hora de modificar filas de una tabla. Supongamos que esta tabla de Empleados tiene también una columna Departamento, y necesitamos que algunos usuarios puedan cambiar el departamento de un empleado pero no el resto de columnas. Solucionaremos este inconveniente, creando el procedimiento almacenado de la Figura 125, que sólo permitirá modificar la columna mencionada.

Al igual que hacíamos antes con la vista, podemos crear una función de base de datos para los usuarios que tengan que cambiar este dato en la tabla. Evitaremos repetir este proceso utilizando la función ya

existente LectoresEmpleados (aunque su nombre no sea identificativo de esta operación), asignando permiso de ejecución sobre este procedimiento almacenado. Ver Figura 126.

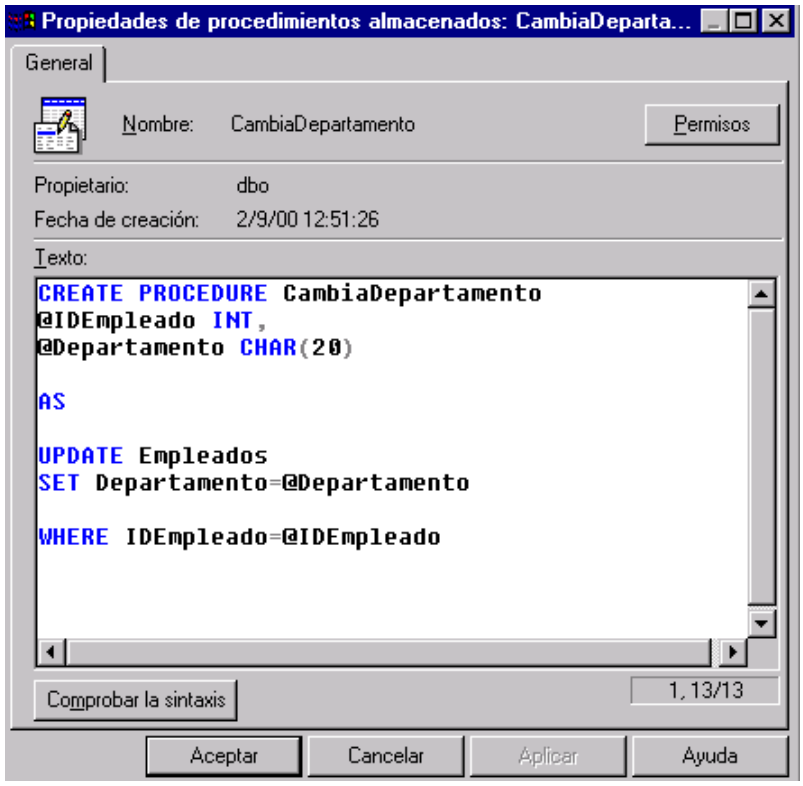


Figura 125. Procedimiento almacenado para realizar el cambio de una columna en la tabla Empleados.

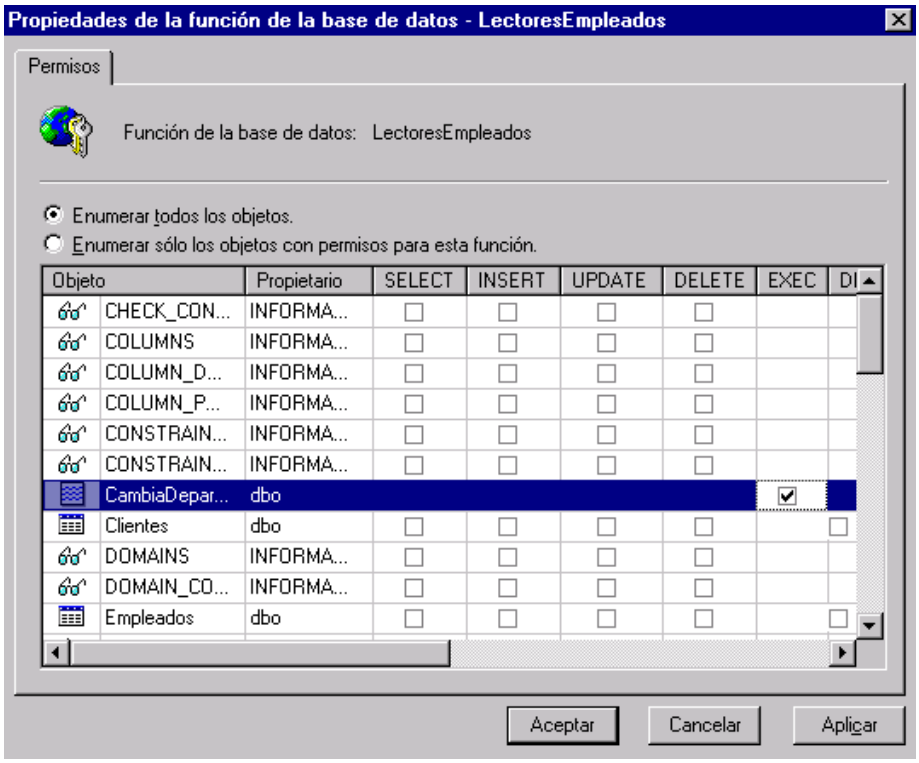


Figura 126. Asignando permiso de ejecución para un procedimiento almacenado en una función de base de datos.

De esta manera, los usuarios que pertenezcan a esta función podrán ejecutar este procedimiento almacenado y cambiar sólo los valores de la columna Departamento, de la forma mostrada en el Código fuente 43.

```
EXEC CambiaDepartamento 2,'Contabilidad'
```

Código fuente 43

14

Copia de seguridad

La importancia de mantener la información a salvo.

Un gestor de datos como SQL Server, es capaz de gestionar y mantener la información de una empresa de cualquier tamaño, desde una pequeña, con una mínima cantidad de datos, hasta una gran compañía con varias sucursales, que soporte un elevado número de operaciones diarias contra sus diferentes bases de datos.

No cabe duda, de que para cualquier empresa, la información contenida en sus bases de datos es de importancia vital; de ahí que sea necesario equipar al gestor de datos empleado, con una utilidad que permita crear copias de respaldo para los datos.

SQL Server integra como uno de sus componentes, una utilidad para realizar copias de seguridad de gran potencia, pero al mismo tiempo sencilla de manejar y flexible en cuanto a configuración, que se adapta por igual a los requerimientos de copia tanto para pequeñas como para grandes cantidades de datos.

¿Por qué hacer copias de seguridad?

A pesar de que esta pregunta pueda parecer un tanto absurda, siempre puede existir quien opine que no compensa el tiempo que se debe dedicar a establecer un sistema de copias seguridad, por diversos motivos: es muy complicado, se debe emplear mucho tiempo, tengo una cantidad de información muy pequeña, o simplemente "mi sistema lleva funcionando mucho tiempo y nunca ha fallado".

Pues ante estas posibles opiniones, debemos decir que nuestros datos están expuestos constantemente al riesgo de pérdida, mucho más de lo que los usuarios piensan, por ejemplo: borrados/modificaciones accidentales de los operadores de la base de datos por ejecución incorrecta de instrucciones DELETE/UPDATE; virus; errores en el hardware, más concretamente, en el disco duro; caídas y picos de tensión eléctrica, etc.

¿Cómo planificar una adecuada política de copias de seguridad?

Todas las empresas no son iguales, evidentemente; por dicho motivo, sus requerimientos en cuanto a la salvaguarda de sus datos tampoco serán los mismos.

Diseñar una estrategia adecuada de copias de seguridad para una base de datos, es un proceso que requiere tiempo; por tal causa, hemos de buscar un equilibrio, entre el nivel de integridad al que queremos devolver una base de datos dañada, y el tiempo que estamos dispuestos a invertir, en primer lugar, en diseñar la estrategia de copia, y después, en la aplicación de dicha estrategia, que deba ser realizada con una regularidad variable según la importancia de nuestros datos.

Generalmente, cuando ocurra un problema que requiera restaurar una base de datos dañada, siempre existirá una cantidad de datos que se perderá. Es en este punto, donde debemos sopesar cuanta información estamos dispuestos a perder, según sea nuestro sistema de copia de seguridad: una estrategia de copia intensiva, que se realice con mucha frecuencia, nos asegurará una cantidad de datos perdidos mínima, pero por contra, empleará mucho tiempo en realizarse; por otro lado, una copia de seguridad realizada de forma más espaciada, no requerirá que un usuario emplee mucho tiempo, pero correremos el riesgo de una mayor cantidad de información perdida en caso de fallos.

La frecuencia de las copias

Después de haber analizado cuanto riesgo estamos dispuestos a asumir en caso de pérdida de información, estableceremos la frecuencia con que realizaremos las copias.

Será conveniente una copia frecuente, si sobre la base de datos se producen constantes actualizaciones de información.

Si la base de datos no tiene un elevado número de modificaciones y se utiliza principalmente para consulta realizaremos una copia de seguridad con menor frecuencia

En cualquier caso, debemos buscar los intervalos de tiempo, si es que existen, en los que la base de datos tenga una menor actividad, para obtener el mejor rendimiento de la copia.

¿Quién puede hacer las copias?

Además del administrador del servidor SQL Server, los usuarios que conecten y estén incluidos en la función fija de servidor sysadmin y las funciones fijas de base de datos db_owner y db_backupoperator, podrán realizar copias de seguridad de las diferentes bases de datos existentes.

Adicionalmente, el administrador también puede crear nuevas funciones de base de datos, con permisos de copia de seguridad e incluir a los usuarios que considere conveniente.

¿Cuándo se deben hacer las copias?

Además de hacer copias de seguridad con la frecuencia establecida en la estrategia de copia habitual, existe un conjunto de casos especiales en los que tendremos que hacer una copia extra de nuestras bases de datos. A continuación se relacionan estas situaciones:

Creación de una base de datos

Es recomendable hacer una copia de seguridad de la base de datos, después de su creación o bien después de la primera grabación de datos. Sin al menos, una copia de seguridad completa de la base de datos, no se podrán restaurar las copias de seguridad del registro de transacciones.

Creación de índices

Aunque no es obligatorio, sí es muy conveniente hacer una copia de seguridad de la base de datos después de crear un índice, puesto que la copia de seguridad guarda los datos y estructura del índice ahorrando tiempo en la fase de restauración.

Tenga en cuenta el lector, que en determinados casos, puede ser mayor el tiempo que se tarda en crear de nuevo el índice que en restaurar una copia de seguridad.

En este caso, tampoco nos serviría hacer solamente una copia de seguridad del registro de transacciones, puesto que este sólo contiene la notificación de que se ha generado el índice y no su contenido.

Limpiar el registro de transacciones

Al ejecutar los comandos BACKUP LOG WITH TRUNCATE_ONLY o BACKUP LOG WITH NO_LOG, se limpia el registro de transacciones y no se puede usar para restaurar, por lo que se debe hacer una copia de seguridad completa de la base de datos.

Operaciones no registradas

Operaciones como la limpieza del registro, mencionada en el punto anterior; modificación de columnas de texto con las instrucciones WRITETEXT o UPDATETEXT; y la ejecución de la instrucción SELECT...INTO.

De este tipo de operaciones no queda constancia en el registro de transacciones, por lo que si ocurre un error, no se podrá restaurar la base de datos.

Existe también, un conjunto de operaciones que modifican la información almacenada en las bases de datos del sistema, y que requerirán realizar una copia de seguridad de las mismas ante posibles fallos del sistema de datos. Veamos a continuación, una relación de las bases de datos implicadas junto a las operaciones que las afectan:

Master

Deberemos realizar una copia de seguridad de esta base de datos en los siguientes casos:

- Cada vez que se crea, modifica o elimina una base de datos en el servidor, ya que dicha actividad se actualiza en master.

- Al ejecutar el procedimiento almacenado del sistema sp_logdevice, que cambia el registro de transacciones; y también al ejecutar los procedimientos almacenados sp_addserver, sp_dropserver y sp_addlinkedserver, que manipulan servidores de datos. Evidentemente, si realizamos estas operaciones desde el Administrador corporativo, también será necesaria una copia de seguridad de master.

Msdb

Es conveniente hacer una copia de la base de datos msdb después de realizar modificaciones sobre la misma, ya que contiene la información de trabajos, alertas y operadores utilizados por SQL Server.

Si se produce un fallo en el sistema y no disponemos de una copia de esta base de datos, tendremos que volver a crear todas las bases de datos del sistema y a continuación todos los trabajos, alertas y operadores.

Model

Si alteramos la base de datos model, para que las nuevas bases de datos que generemos tengan una configuración personalizada, deberemos crear una copia de seguridad de la misma, ya que en caso de error, si no disponemos de copia de esta base de datos, perderemos la configuración personalizada que establecimos.

La ubicación física de las copias

La herramienta de copia de SQL Server, permite almacenar las copias del servidor en uno de los siguientes elementos físicos:

- Archivos de disco. Constituyen la ubicación más frecuentemente utilizada.

- Cinta. El dispositivo lector/grabador debe estar conectado al servidor SQL Server de forma local.

- Canalización con nombre. Permite que aplicaciones software de terceros fabricantes proporcionen utilidades para restaurar sus datos en SQL Server.

Restricciones al realizar una copia de seguridad

Aunque es posible realizar una copia de seguridad mientras se trabaja con la base de datos, existen tareas que no podrán llevarse a cabo al mismo tiempo que se crea la copia. Estas operaciones son las siguientes:

- Creación y modificación de la estructura de una base de datos.
- Creación de índices.
- Las operaciones no registradas, indicadas en el apartado anterior: WRITETEXT, UPDATETEXT y SELECT...INTO.

Durante el transcurso de una copia de seguridad, no se permitirá al usuario la ejecución de ninguna de las anteriores operaciones.

De igual forma, si una de estas operaciones está ejecutándose y se intenta realizar una copia, el desarrollo de la copia se detendrá.

Copias de seguridad dinámicas

En SQL Server es posible realizar una copia de seguridad de una base de datos, al mismo tiempo que los usuarios modifican dicha base de datos, mediante un proceso denominado copia de seguridad dinámica, que ocurre de la siguiente forma:

En primer lugar, SQL Server establece un punto de comprobación en la base de datos y registra el número de secuencia de registro (Log Secuence Number, LSN), del registro de transacciones más antiguo.

A continuación, escribe los datos en la copia de seguridad.

Por último, escribe todos los cambios sufridos por la base de datos durante el proceso de copia y que están escritos en el registro de transacciones; esta información se encuentra desde el último número de secuencia de registro hasta el final del mismo.

Realización de una copia de seguridad con el Administrador corporativo

Después de conectar con un servidor SQL Server, abriremos la carpeta Administración y haremos clic con el botón derecho sobre el elemento *Copia de seguridad*, eligiendo la opción del menú contextual *Copia de seguridad de base de datos*, que nos mostrará la ventana de la Figura 127, para configurar la copia que vamos a realizar.

En esta ventana, debemos seleccionar de la lista Base de datos, cuál es la base de la que vamos a crear la copia; podemos asignarle también un nombre, aunque ya nos proporciona uno por defecto; escribir una breve descripción del contenido de la copia; el tipo de copia a realizar: completa, diferencial, etc.; la ubicación de la copia: disco, cinta, nombre del dispositivo de copia o fichero; tipo de escritura; y por último, configurar una copia programada para una fecha y hora determinadas.

Podemos acceder también a esta ventana de copia de seguridad, seleccionando la base de datos de la que vamos a crear la copia, y a continuación, en el panel derecho del Administrador corporativo, seleccionar el vínculo correspondiente del apartado *Copia de seguridad*, Figura 128.

Una tercera forma consiste en seleccionar la base de datos, elegir la opción de menú del Administrador corporativo *Acción+Todas las tareas+Copia de seguridad de la base de datos*.

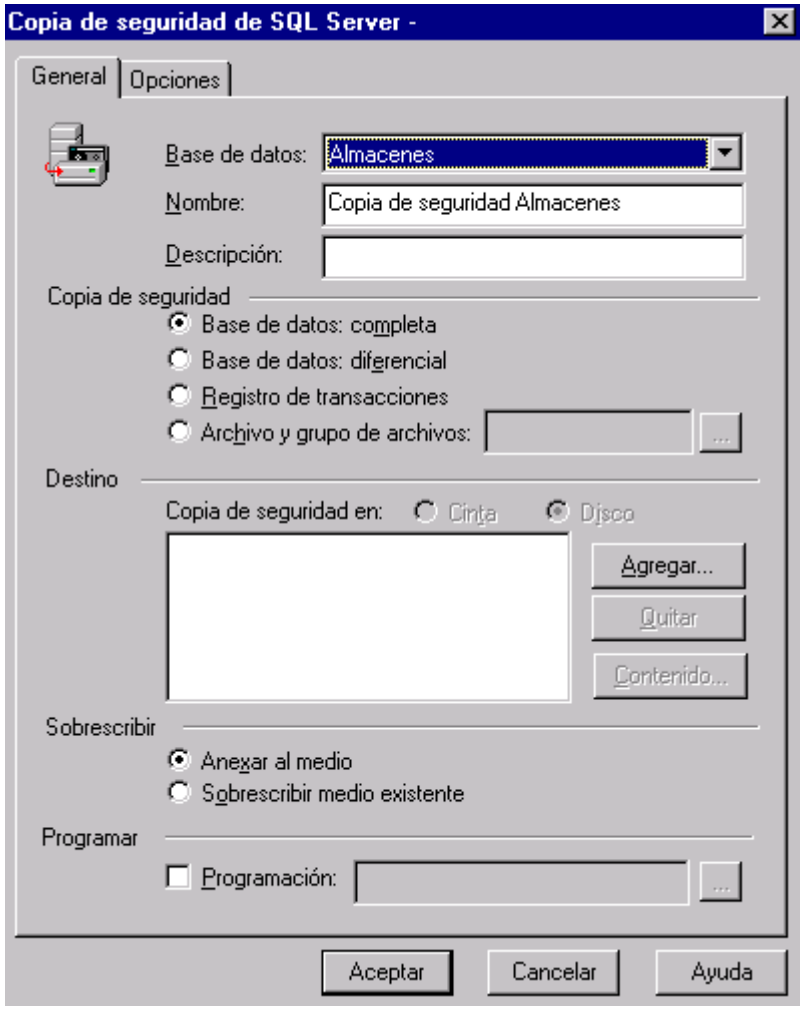


Figura 127. Ventana de configuración de copia de seguridad.

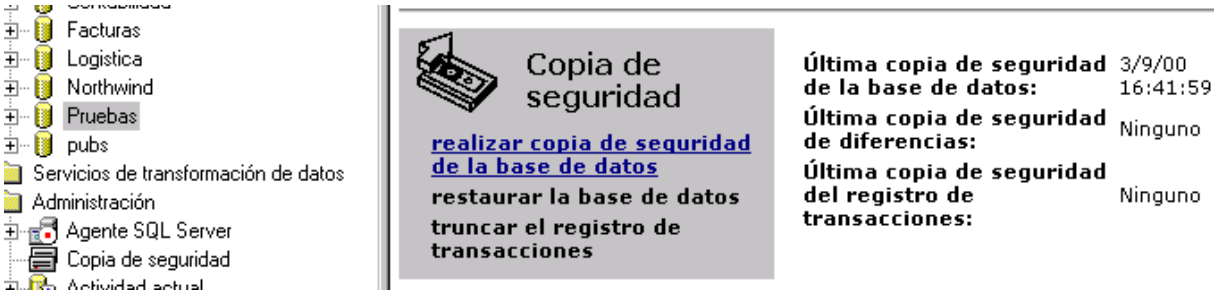


Figura 128. Acceso a la copia de seguridad desde una base de datos.

Una copia de seguridad puede tener un gran número de opciones de configuración, por este motivo y puesto que es nuestra primera copia, vamos a mostrar a través de un ejemplo, la realización de una copia de seguridad sencilla, con la mínima configuración precisa para poder ser realizada.

Seleccionaremos Pruebas como base de datos, en la lista que contiene las bases de datos del servidor; le daremos como nombre CopiaUnoPruebas y una descripción. En el tipo de copia marcaremos completa. Para indicar el destino de la copia, pulsaremos Agregar, en donde podremos escribir el nombre de un fichero en el que se depositará la copia, o un dispositivo en el que almacenar una o varias copias; en este caso, indicaremos la ruta y el nombre del fichero que se muestra en la Figura 129.

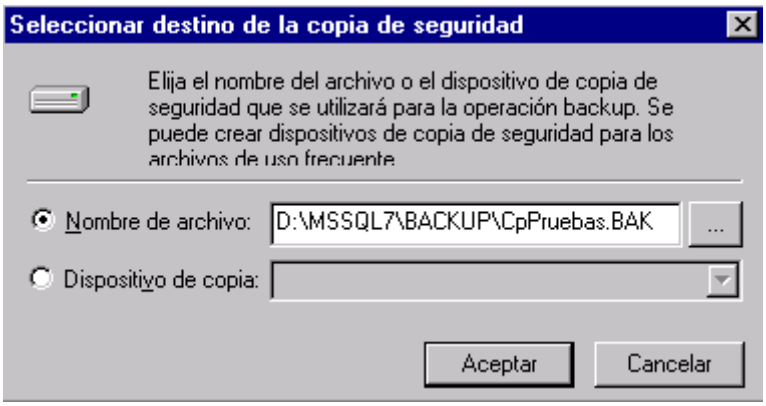


Figura 129. Nombre de archivo para la copia de seguridad.

Con toda esta información, la ventana para crear la copia de seguridad quedaría como se muestra en la Figura 130.

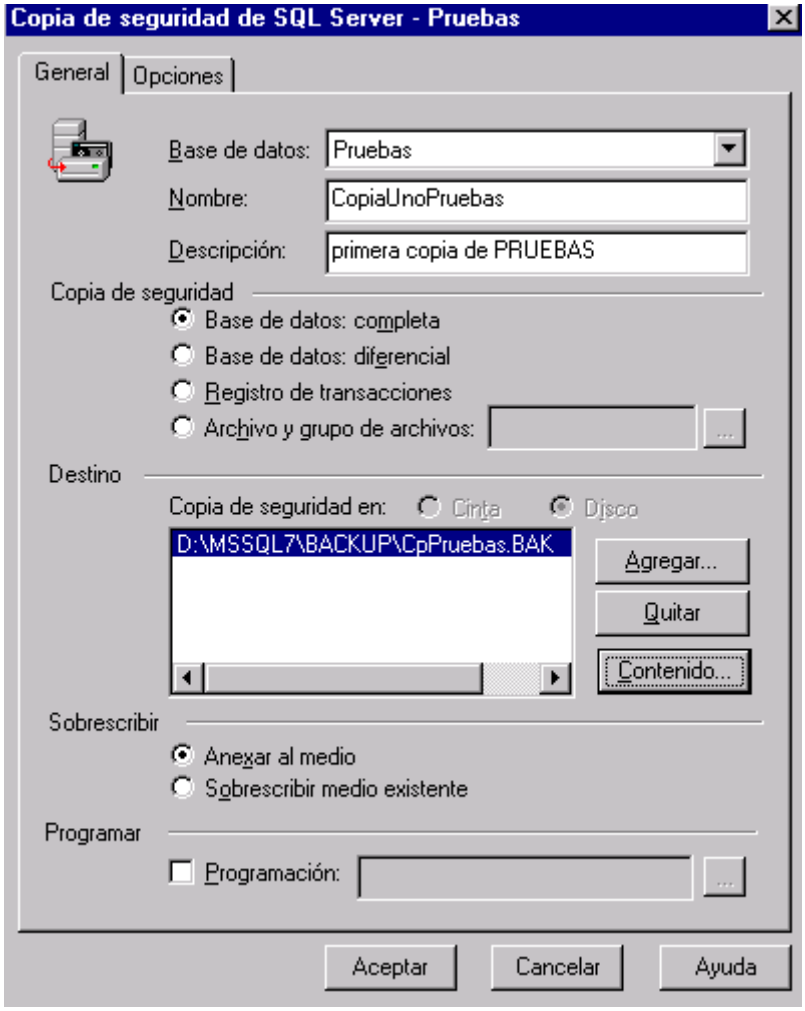


Figura 130. Valores establecidos para realizar la copia de seguridad.

Finalmente, pulsaremos Aceptar, comenzando el proceso de copia, que se creará en el destino indicado.

Realización de una copia de seguridad con la instrucción BACKUP

El comando BACKUP, nos permite realizar copias de seguridad de una base de datos, empleando una utilidad de línea de comandos como el Analizador de consultas de SQL Server. A continuación se describe su sintaxis básica.

```
BACKUP DATABASE NombreBD TO DispositivoCopia
[, ...DispositivoCopiaN]
```

- NombreBD. Nombre de la base de datos de la que va a crear la copia de seguridad.
- DispositivoCopia. Dispositivo de copia, temporal o permanente, sobre el que se va a crear la copia.

Este comando debe ser ejecutado desde la base de datos del sistema master, por lo que deberemos posicionarnos previamente sobre ella.

En el Código fuente 44, se realiza una copia de la base de datos Almacenes, en un dispositivo de copia temporal (archivo de disco), con el nombre BACK1ALMA.BAK.

```
USE master
BACKUP DATABASE Almacenes TO DISK='C:\DATOSBAK\BACK1ALMA.BAK'
```

Código fuente 44

Elementos opcionales de la instrucción BACKUP

Mediante la palabra clave WITH y el nombre de la opción, podemos añadir a este comando nuevos valores de configuración.

Por ejemplo, la opción NAME, nos permite asignar el nombre a la copia de seguridad, cumpliendo la misma función que el campo Nombre en la ventana de creación de la copia desde el Administrador corporativo.

```
BACKUP DATABASE NombreBD TO DispositivoCopia
[, ...DispositivoCopiaN] [WITH [NAME = 'NombreCopia'] ]
```

El Código fuente 44, crea una copia de seguridad de la base de datos Pruebas con el nombre CopiaUnoPruebas.

```
USE master
BACKUP DATABASE Pruebas TO DISK='C:\DATOSBAK\BAC1PRUEBAS.BAK' WITH
NAME='CopiaUnoPruebas'
```

Código fuente 45

Estructura interna de las copias de seguridad

Una copia de seguridad de SQL Server, se organiza internamente en base a los elementos descritos en los siguientes apartados:

Conjunto de copia de seguridad

Las copias de seguridad en SQL Server, se graban mediante el formato de cinta de Microsoft: MSTF (Microsoft Tape Format). Aunque se utilice el término cinta, no es de uso exclusivo para este tipo de soporte, pudiendo aplicarse también a discos y canalizaciones con nombre.

Cuando en Windows NT hacemos una copia de seguridad, se crea un conjunto de copia de seguridad que se guarda en una unidad MSTF denominada medio, Figura 131. Un medio puede contener conjuntos de copia de diferentes clases de software.

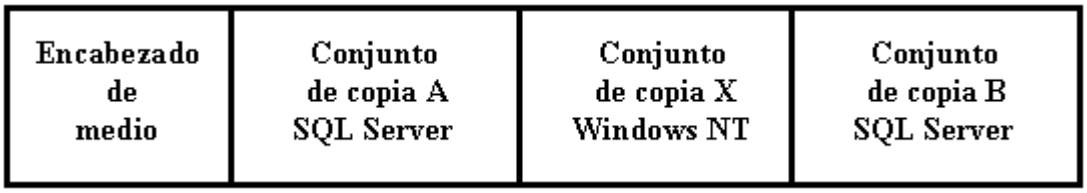


Figura 131. Estructura de un medio de copia de seguridad.

Como puede ver el lector, un medio contiene un encabezado, seguido de los diferentes conjuntos de copia que se han agregado al medio. El encabezado dispone de la información sobre los conjuntos de copia que contiene el medio, siendo como una guía que permite localizar y conocer qué hay dentro de cada conjunto de copia.

Tenga en cuenta el lector, que esta información no es exclusiva de las copias de seguridad en SQL Server, sino que se refiere a las copias de seguridad a nivel general realizadas desde el sistema operativo. La disposición interna de las copias de seguridad en SQL Server se comenta en los sucesivos puntos de este apartado.

Dispositivo

Un dispositivo es aquel componente de una copia de seguridad que almacena la información. Podemos crear dispositivos en disco (archivos de disco), en cinta y canalizaciones con nombre. Al mismo tiempo, el modo de creación de cada dispositivo puede ser temporal o permanente.

Los dispositivos de copia disponen de un nombre físico y lógico para poder ser reconocidos. Por ejemplo, en el caso de ficheros de disco, el nombre físico es el nombre que utiliza el sistema para identificar al fichero (C:\DATOS\COPIA1.BAK), mientras que el nombre lógico es un identificador genérico o alias (CopiaPrimera).

Medio

Un medio es la ubicación física en donde se graban los dispositivos lógicos de copia de seguridad.

En el caso de copias en disco, el dispositivo es el fichero del sistema que contiene los datos, mientras que el medio es el disco físico que contiene los ficheros del sistema. Si por el contrario, estamos utilizando cintas, el dispositivo en esta situación será la unidad lectora/grabadora de cintas, mientras que el medio será la cinta física en la que se graban los datos.

Al realizar copias de seguridad, podemos agregar la copia de seguridad en curso al final de la última copia existente en el medio o sobrescribir las copias existentes en el medio con la actual.

Conjuntos y familias de medios

Cuando en una copia de seguridad intervienen varios medios, estos se organizan en conjuntos y familias.

- Conjunto de medios. Representa a uno o varios conjuntos de copias de seguridad.
- Familia de medios. Representa a todos los medios contenidos en un dispositivo de copia de seguridad, que a su vez está dentro de un conjunto de copias.

La Figura 132 muestra el esquema de un conjunto de medios, con la disposición del propio conjunto, familias y medios que lo componen. El medio o soporte físico de copia utilizado en este esquema es la cinta.

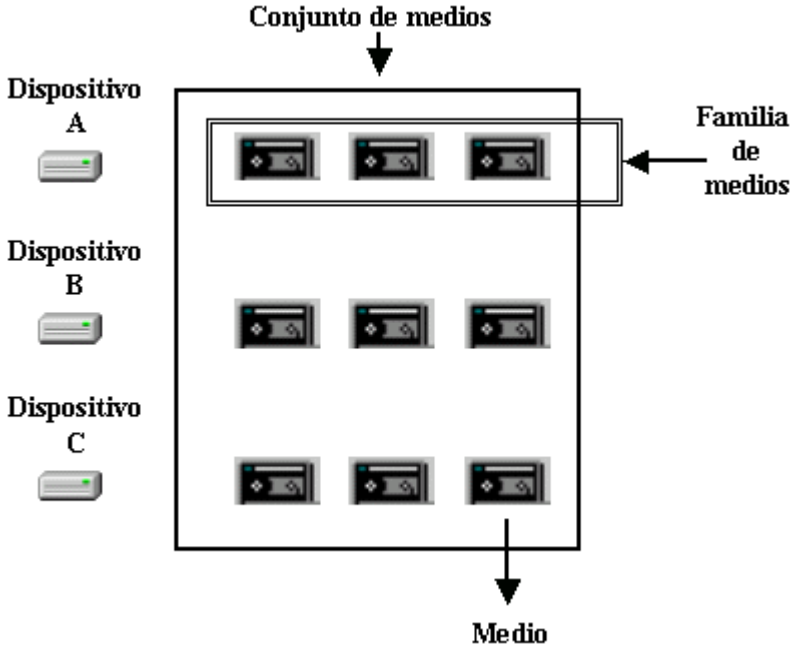


Figura 132. Esquema de un conjunto de medios.

Como acabamos de comprobar, la composición del conjunto de medios es de tres familias; cada familia contiene a su vez, tres medios o cintas.

Creación de dispositivos de copia de seguridad temporales

Cuando vayamos a hacer una prueba de copia de seguridad, que pensamos ejecutar de forma programada posteriormente, con un dispositivo de copia permanente, o si queremos hacer una única copia de seguridad, podemos emplear un dispositivo de copia temporal.

Un dispositivo de copia temporal, es la especificación directa de un fichero de disco o cinta, en el momento de realizar la copia, es decir, se crea en el mismo momento en que se genera la copia.

Administrador corporativo

Para crear un dispositivo de copia temporal con esta aplicación, en la ventana de creación de copia, apartado Destino, especificaremos el tipo de soporte pulsando los botones de opción Cinta o Disco, Figura 133.

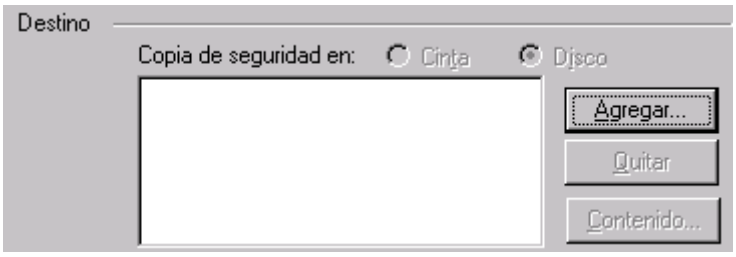


Figura 133. Opciones de dispositivo en la ventana de propiedades de la copia de seguridad

Si optamos por Disco, pulsaremos Agregar, apareciendo la ventana de selección del destino de la copia. Pulsaremos la opción Nombre de archivo, y escribiremos el nombre y ruta del fichero de la copia. Estas opciones se han explicado en el apartado Realización de una copia de seguridad con el Administrador corporativo.

Instrucción BACKUP

La opción DispositivoCopia de esta instrucción, nos permite especificar un dispositivo temporal de copia.

```
BACKUP DATABASE NombreBD TO 'DispositivoCopia'
```

El formato de esta opción es el siguiente:

```
{DISK|TAPE|PIPE} = 'DispositivoCopiaTemp'
```

- DISK. Indica que el dispositivo será un fichero.
- TAPE.. Indica que el dispositivo será una cinta.
- PIPE. Indica que el dispositivo será una canalización con nombre.
- DispositivoCopiaTemp. Especificación del dispositivo, por ejemplo, el nombre y ruta de un fichero.

El Código fuente 46 crea una copia de una base de datos en un fichero temporal del disco.

```
BACKUP DATABASE Pruebas TO DISK='C:\DATOSCOPIA\PRUCOPIA.BAK'
```

Código fuente 46

Creación de dispositivos de copia de seguridad permanentes

Cuando pensemos reutilizar los ficheros que componen un dispositivo, o al configurar una copia programada, usaremos un dispositivo de copia de seguridad permanente.

Un dispositivo de copia permanente, consiste en la especificación de los ficheros que componen el dispositivo antes de realizar la copia, creándose en un proceso aparte del propio proceso de copia.

Administrador corporativo

En primer lugar, abriremos la carpeta Administración, de un servidor SQL Server. A continuación, haremos clic con el botón derecho sobre el elemento Copia de seguridad, seleccionando la opción *Nuevo dispositivo de copia de seguridad*, de su menú contextual, que nos mostrará la ventana de creación de dispositivos, Figura 134, en donde daremos un nombre al mismo e indicaremos el fichero y ruta que servirán de soporte a este dispositivo. Por defecto, el nombre del fichero es el mismo que damos al dispositivo, y la ruta corresponde al directorio BACKUP de la instalación de SQL; todos estos valores son modificables.

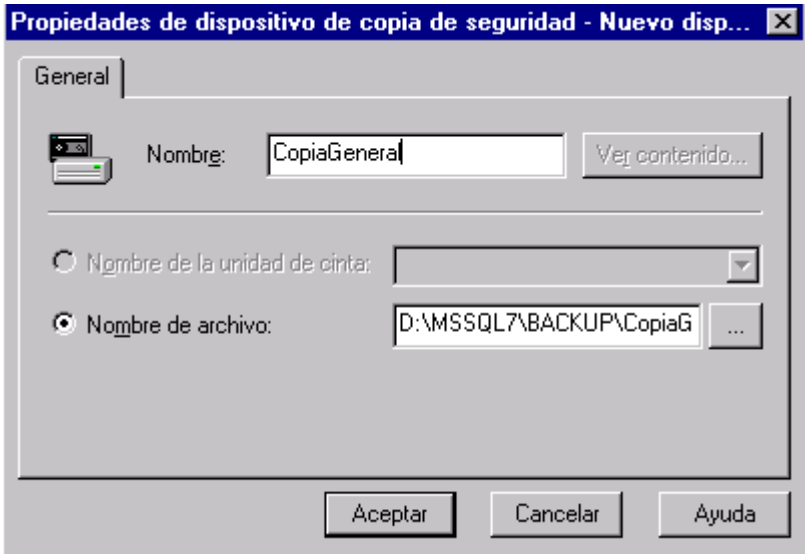


Figura 134. Ventana de propiedades de dispositivos de copia permanentes.

Finalizaremos la creación del dispositivo pulsando Aceptar. Hemos de tener en cuenta, que el fichero físico que sirve de almacén de datos al dispositivo, no se creará en este momento, sino en el momento de realizar la primera copia de seguridad que utilice este dispositivo.

sp_addumpdevice

Este procedimiento almacenado permite también la creación de dispositivos de copia permanentes, utilizando la siguiente sintaxis.

```
sp_addumpdevice 'TipoDispositivo' ,'NombreLógico' ,'NombreFísico'
```

- TipoDispositivo. Uno de los siguientes valores, que indican el tipo de dispositivo:
  - disk. Fichero en disco.
  - tape. Cinta compatible con las especificaciones Windows.
  - pipe. Canalización con nombre.
- NombreLógico. Nombre del dispositivo a efectos de SQL Server.
- NombreFísico. Ruta y nombre del fichero que contiene el dispositivo.

El Código fuente 47, muestra un ejemplo de creación de un dispositivo, utilizando este procedimiento almacenado. Al igual que sucedía con la instrucción BACKUP, debemos estar posicionados en la base de datos master para poder ejecutarlo.

```
EXEC sp_addumpdevice 'disk','DispCopiaTercero','D:\MSSQL7\BACKUP\DISPTERC.BAK'
```

Código fuente 47

Copias de seguridad en múltiples dispositivos

SQL Server puede realizar una copia de seguridad, escribiendo los datos simultáneamente en varios dispositivos, por ejemplo, ficheros o cintas. Esta opción es especialmente interesante en el caso de grandes bases de datos, de forma que si en condiciones normales, una copia de seguridad tarda, por ejemplo, dos horas, empleando la técnica de varios dispositivos con escritura en paralelo, podremos reducir ese tiempo a la mitad.

Tenga en cuenta el lector, que mediante este sistema, el medio físico de los dispositivos empleados debe ser del mismo tipo: todos ficheros, o bien, todos cintas; siendo posible utilizar una combinación de dispositivos temporales y permanentes.

Dispositivos, medios y familias

En este apartado se va a describir mediante un ejemplo, el proceso de creación de una copia de seguridad, empleando varios dispositivos o archivos; al mismo tiempo, servirá para realizar una descripción de la estructura interna de una copia de seguridad en función de sus elementos integrantes: dispositivos, medios y familias.

Creación de dispositivos

En primer lugar, crearemos tres dispositivos permanentes con los nombres Disp1Pruebas, Disp2Pruebas, Disp3Pruebas, en la forma descrita en el apartado dedicado a estos dispositivos. El nombre y ubicación, será el asignado por defecto por SQL Server, Figura 135.

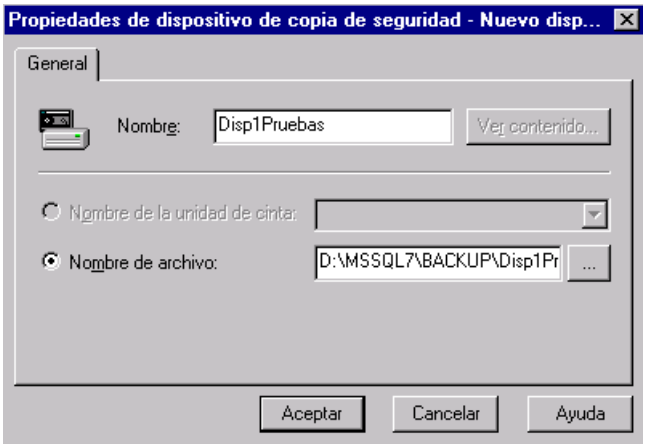


Figura 135. Creación del dispositivo de copia permanente Disp1Pruebas.

Configurar copia de seguridad

A continuación, abriremos la ventana de creación de copia de seguridad, seleccionaremos la base de datos Pruebas, escribiremos como nombre de la copia: Copia BaseDatos Prueba, y agregaremos los dispositivos creados, Figura 136.

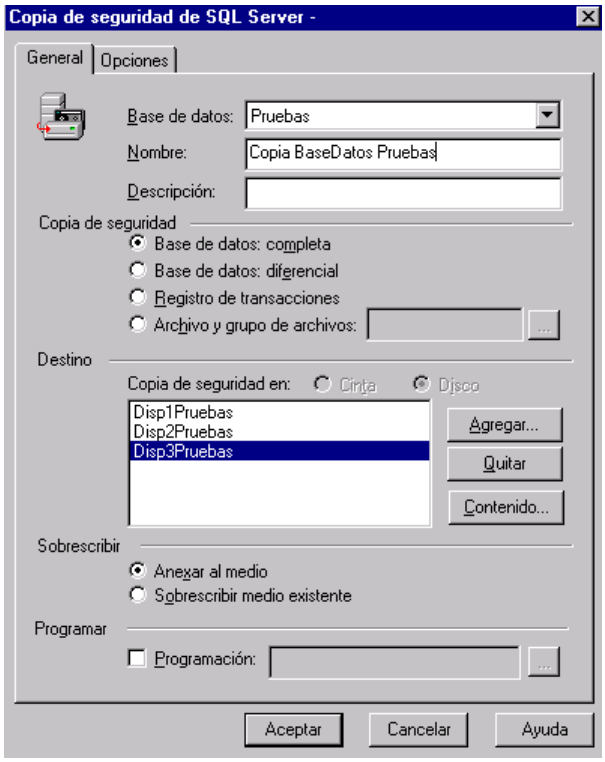


Figura 136. Creación de copia, pestaña General.

Haremos seguidamente clic en la pestaña Opciones de la ventana de copia de seguridad, e introduciremos como nombre para el conjunto de medios el valor ConjuntoCopiaPruebas , de forma que podamos tener identificado mediante un nombre lógico, todos los elementos que integrarán la copia que estamos configurando, Figura 137.

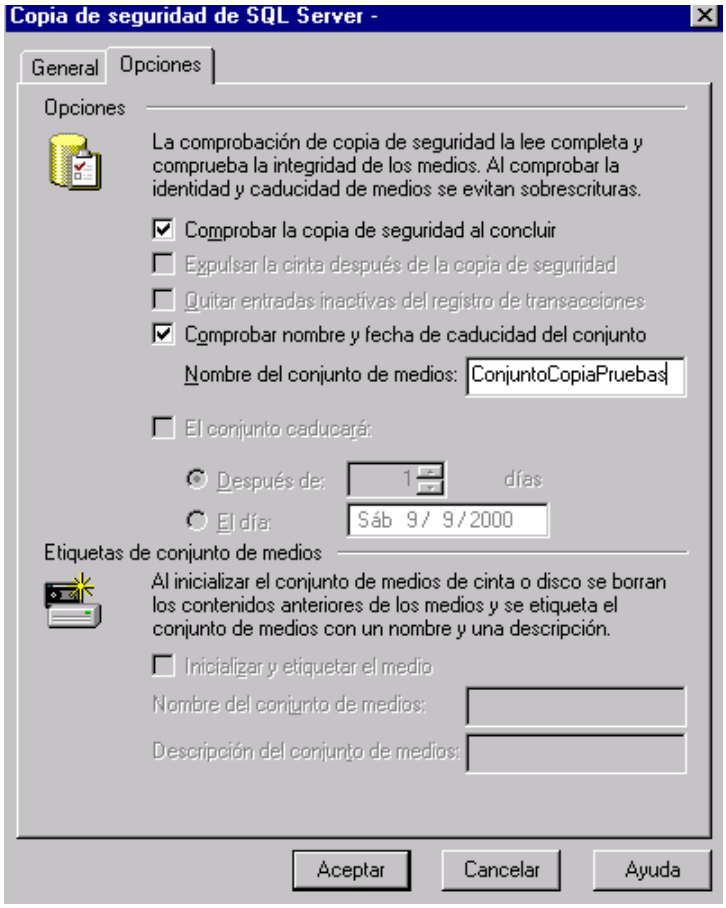


Figura 137. Creación de copia, pestaña Opciones.

Adicionalmente, es posible especificar en este punto, que se efectúe una comprobación tanto del contenido de la copia, como de su nombre y fecha de caducidad.

Pulsaremos Aceptar, con lo que creará la copia de seguridad y se efectuará una verificación de su contenido.

El interior de la copia de seguridad

Finalizada una copia de seguridad, para ver como ha quedado dispuesta la información internamente, abriremos los dispositivos de copia en los que se han depositado los datos, ubicados en la carpeta Administración, elemento Copia de seguridad, del servidor SQL Server.

Si en nuestro ejemplo, hacemos doble clic sobre el dispositivo Disp2Pruebas, se mostrará una primera ventana general de información del dispositivo, Figura 138.

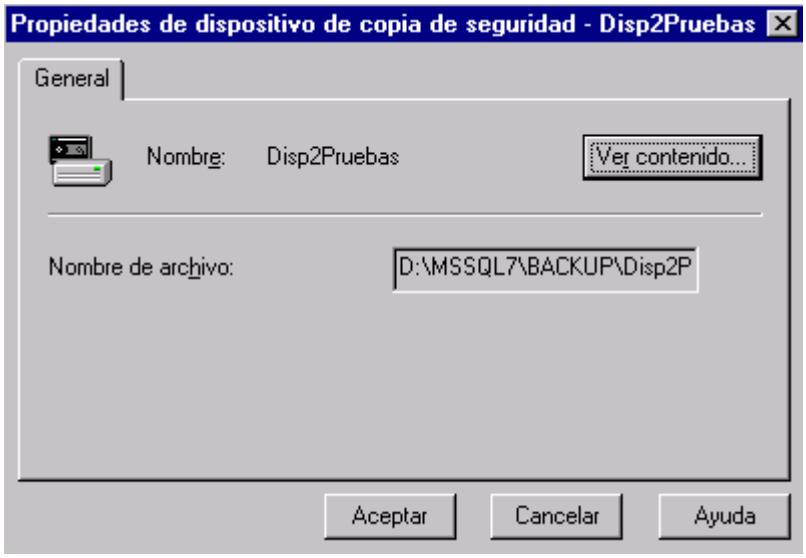


Figura 138. Propiedades de un dispositivo de copia.

Pulsando el botón Ver contenido, se mostrará la información que nos interesa: nombre del conjunto de medios, secuencia de la copia en donde está situado el dispositivo (número de familia y medio), e información que contiene el dispositivo, como puede ser el nombre de la copia de seguridad, base de datos de la que se ha hecho la copia, tipo, fecha, tamaño, etc.

En el caso de que este dispositivo se haya utilizado varias veces para hacer copias de seguridad que hayan sido anexadas, esta ventana mostrará una línea por cada copia de seguridad realizada.

La Figura 139 muestra esta información del dispositivo Disp2Pruebas.

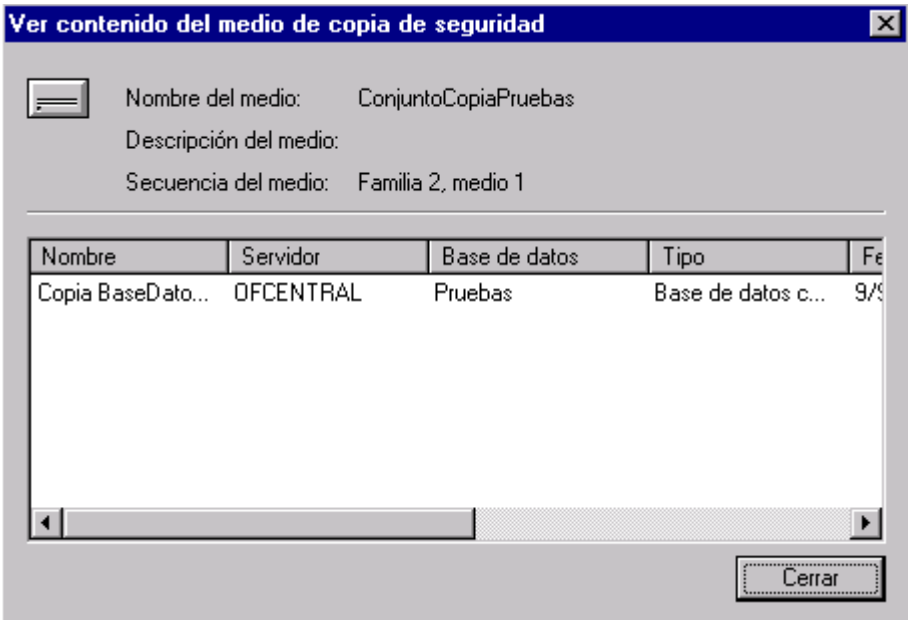


Figura 139. Información sobre el contenido de un dispositivo de copia de seguridad.

Si hemos realizado una copia de seguridad utilizando una combinación de dispositivos permanentes y temporales, estos últimos no serán visibles en la lista de dispositivos del elemento Copia de seguridad. Para poder acceder a un dispositivo temporal, tendremos que abrir la ventana de creación de copia de

seguridad, y seleccionar la base de datos sobre la que hemos hecho la copia, en nuestro ejemplo es Pruebas, y en la lista de dispositivos aparecerán todos los que componen esta copia.

Aquí sí podremos seleccionar los dispositivos temporales (y también los permanentes) y pulsar el botón Contenido, para ver la información que contienen.

La estructura de esta copia, quedaría esquemáticamente como se muestra en la Figura 140

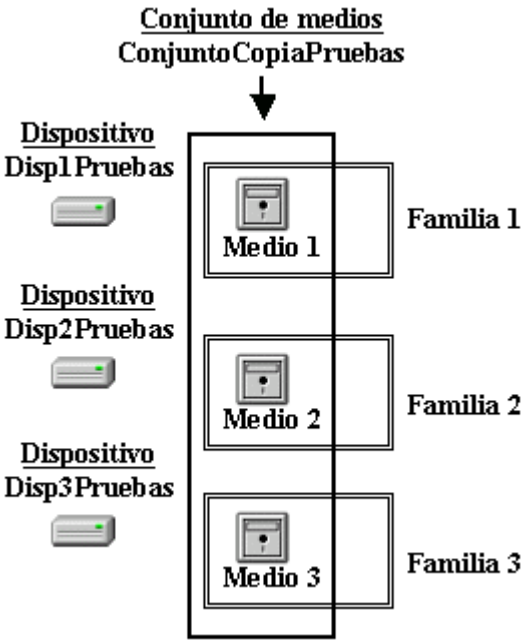


Figura 140. Esquema del conjunto de medios ConjuntoCopiaPruebas.

Debemos tener presente también, que en este tipo de copias, una vez que hemos añadido un dispositivo al conjunto de dispositivos que forman la copia, será obligatorio en lo sucesivo, utilizar ese dispositivo en las posteriores copias.

La única forma de volver a utilizar un dispositivo que forme parte de un conjunto de dispositivos en una copia, de forma independiente, o dentro con otro conjunto de dispositivos distinto, es quitar el dispositivo del conjunto, con lo que perderemos la copia de seguridad al completo.

Especificar el nombre del conjunto de medios mediante BACKUP

Al igual que hacíamos en el anterior apartado, desde la pestaña Opciones de la ventana de configuración de la copia de seguridad, la opción MEDIANAME del comando BACKUP, nos permitirá establecer el nombre del conjunto de medios para una copia de seguridad.

```
BACKUP DATABASE NombreBD TO DispositivoCopia
[, ... DispositivoCopiaN] [WITH MEDIANAME = 'NombreConjuntoMedios']
```

El Código fuente 48 muestra la creación de una copia de seguridad de la base de datos Contabilidad, repartida en dos dispositivos de copia permanentes, a los que se asignará el nombre de conjunto de medios ConjMediosContabilidad.

```
BACKUP DATABASE Contabilidad TO Disp1Conta, Disp2Conta WITH
MEDIANAME='ConjMediosContabilidad'
```

Código fuente 48

Modos de escritura de una copia de seguridad

Al realizar una copia de seguridad, es posible indicar si esta se va a anexar a una copia ya existente en el dispositivo/s de copia especificado/s, o bien, se va a sobrescribir, perdiendo la copia anterior.

Administrador corporativo

En la ventana de propiedades de la copia de seguridad, podemos establecer estos valores, pulsando en los botones de opción Anexar al medio o Sobrescribir medio existente, del apartado Sobrescribir, Figura 141, que respectivamente, anexarán la nueva copia al dispositivo o dispositivos especificados en Destino, o sobrescribirán la copia existente en el dispositivo.

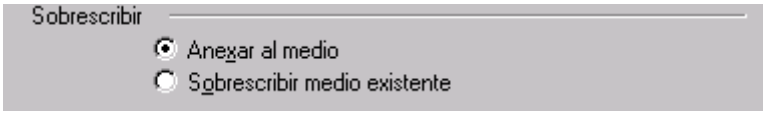


Figura 141. Modos de escritura en dispositivo de la copia de seguridad

Instrucción BACKUP

A continuación, se muestran las opciones de esta instrucción, relacionadas con el modo de escritura de una copia de seguridad.

```
BACKUP DATABASE NombreBD TO DispositivoCopia
[, ...DispositivoCopiaN] [WITH [EXPIREDATE = fecha] [[,]RETAINDAYS =
NumDías] [[,]FORMAT] [[,]INIT | NOINIT]]
```

- NOINIT. Opción predeterminada, la nueva copia se anexa a la ya existente.

- INIT. La copia sobrescribe a la que existía anteriormente, pero se conserva la información del encabezado en el conjunto de medios de la copia, naturalmente siempre y cuando esto sea posible, puesto que antes de sobrescribir la nueva copia, SQL Server analiza los valores del encabezado para determinar si la nueva copia se puede sobrescribir.

- FORMAT. Esta opción, como su nombre indica, formatea el dispositivo de copia, reiniciando toda su información, por lo que debe emplearse con cuidado. Escribe el encabezado del conjunto de medios de la copia, quedando la anterior copia invalidada, de forma que la nueva copia sobrescribe a la anterior.

- EXPIREDATE. Permite especificar una fecha a partir de la cual, la copia caduca y puede ser sobrescrita. Si la fecha actual no es superior, la copia fallará.

- RETAINDAYS. Permite especificar un número de días que deben transcurrir hasta que la copia pueda ser sobrescrita. Si no ha transcurrido el número de días indicado, la copia fallará.

El Código fuente 49 muestra algunos ejemplos de copia de seguridad con BACKUP y estas opciones de escritura.

```
/*Crea una copia que sobrescribe la anterior del dispositivo*/
BACKUP DATABASE Pruebas TO DispAPruebas WITH INIT

/*Crea una copia que se añade a las copias existentes en el dispositivo*/
BACKUP DATABASE Pruebas TO DispAPruebas WITH NOINIT, NAME='CopiaA3Pruebas'

/*Invalida toda la información de las anteriores copias y crea una nueva*/
BACKUP DATABASE Pruebas TO DispAPruebas WITH FORMAT
```

Código fuente 49

15

Métodos de copia de seguridad

Elegir la mejor técnica para realizar una copia de seguridad

SQL Server proporciona varios métodos de copia de seguridad, que nos permitirán seleccionar qué parte de la base de datos copiar. Este aspecto nos ayudará a que al diseñar una estrategia de copia, se adapte el máximo posible a nuestras necesidades.

El tipo de dispositivo empleado en los distintos métodos de copia es indiferente, pudiendo realizarse sobre dispositivos permanentes o temporales.

Copia de seguridad completa

Este tipo de copia es de realización obligada, al menos una vez, para la base de datos; creándose una copia de todo el contenido de la base de datos: estructuras, información y transacciones no confirmadas del registro de transacciones.

Para poder restaurar una base de datos desde copias de seguridad diferenciales o del registro de transacciones, es necesario que exista una copia de seguridad completa, que sirva como base de restauración.

Una copia de seguridad completa es recomendable para bases de datos de pequeño tamaño o en las que mayoritariamente se realicen consultas.

Administrador corporativo

Para crear una copia de seguridad mediante este método, en el apartado Copia de seguridad de la ventana de propiedades de la copia de seguridad, marcaremos la opción Base de datos: completa, Figura 142.

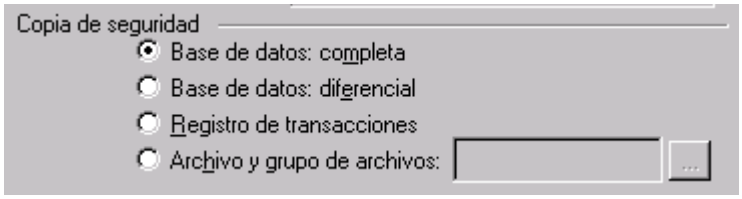


Figura 142. Especificación de copia completa de una base de datos.

Instrucción BACKUP

Empleando la sintaxis más básica de este comando, crearemos una copia de seguridad completa, como se muestra en el Código fuente 50.

```
BACKUP DATABASE Pruebas TO DISK='D:\MSSQL7\BACKUP\CP1PRUEBAS.BAK'
```

Código fuente 50

Copia de seguridad diferencial

Una copia de seguridad diferencial tiene como principal ventaja, reducir el tiempo que se tarda en crear la copia, al copiar sólo aquellas partes de la base de datos que han cambiado desde la última vez que se creó una copia de seguridad completa.

Este tipo de copia es recomendable para bases de datos de gran tamaño, puesto que se reduce el tiempo empleado en realizar la copia, al copiarse sólo las porciones de la base de datos que han cambiado. También se reduce el tiempo de restauración, al no tener que ejecutar varios registros de transacciones.

Debemos tener en cuenta, que si un dato ha sido modificado en varias ocasiones desde la última copia de seguridad completa, al restaurar la copia de seguridad diferencial, sólo recuperaremos la última modificación realizada al dato. Para disponer de un histórico de cambios sobre el dato, tendremos que emplear copias de seguridad del registro de transacciones.

Para saber qué información de la base de datos ha cambiado, SQL Server comprueba el valor del LSN del registro de transacciones de la página de datos, con el mismo valor grabado en la última copia de seguridad completa que debe existir de la base de datos.

Durante una copia de seguridad diferencial, SQL Server no copia páginas de datos, sino extensiones, copiándose una extensión cuando el valor LSN de una página contenida en una extensión es mayor que el LSN de la última copia completa de la base de datos.

Administrador corporativo

Para crear una copia de seguridad mediante este método, en el apartado Copia de seguridad de la ventana de propiedades de la copia de seguridad, marcaremos la opción Base de datos: diferencial, Figura 143.

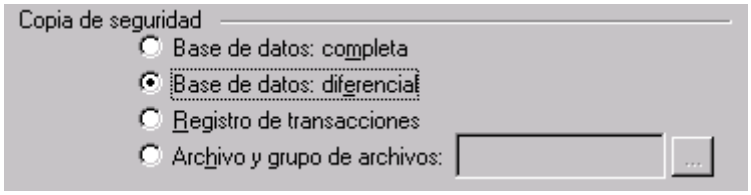


Figura 143. Especificación de copia diferencial de una base de datos.

Instrucción BACKUP

La opción DIFFERENTIAL de esta instrucción, nos permitirá crear una copia de seguridad de este tipo.

```
BACKUP DATABASE NombreBD TO DispositivoCopia
[, ...DispositivoCopiaN] [WITH [DIFFERENTIAL]]
```

El Código fuente 51 realiza una copia de seguridad diferencial de la base de datos Facturas a un dispositivo de copia temporal.

```
BACKUP DATABASE Facturas TO DISK='D:\MSSQL7\BACKUP\CPDIFFAC1.BAK' WITH DIFFERENTIAL
```

Código fuente 51

Copia de seguridad del registro de transacciones

Este tipo de copia, contiene la información del registro de transacciones, de modo que podemos disponer del historial de cambios realizados a los datos. El proceso consiste en copiar el registro desde su última copia, hasta el final; a continuación se trunca el registro de transacciones, para poder liberar espacio dentro del mismo.

Administrador corporativo

Podemos crear una copia del registro de transacciones en el apartado Copia de seguridad de la ventana de propiedades de la copia de seguridad, marcando la opción Registro de transacciones, Figura 144.

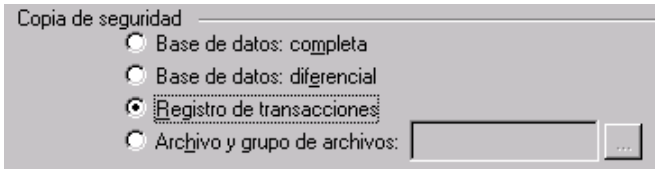


Figura 144. Especificación de copia del registro de transacciones de una base de datos.

Instrucción BACKUP

La palabra clave LOG de este comando, nos permite realizar una copia de seguridad del registro de transacciones.

```
BACKUP LOG NombreBD TO NombreBD TO DispositivoCopia
[,...DispositivoCopiaN]
```

El Código fuente 52 realiza una copia de seguridad del registro de transacciones de la base de datos Pruebas sobre un dispositivo creado previamente.

```
BACKUP LOG Pruebas TO PruebasLog
```

Código fuente 52

Al utilizar BACKUP LOG, se copia la información del registro de transacciones, desde la última copia correcta del mismo hasta el final; a continuación, trunca el registro, es decir, elimina su parte inactiva o transacciones confirmadas, y mantiene la información desde el comienzo de su parte activa, la que contiene transacciones sin confirmar.

Otras opciones de uso común junto a esta instrucción son las siguientes:

- NO_TRUNCATE. Realiza la copia de seguridad del registro de transacciones sin truncarlo al final. Esta opción es adecuada en el caso de que la base de datos esté dañada.

- NO_LOG – TRUNCATE_ONLY. Estas dos opciones realizan la misma labor: liberan espacio en la base de datos, truncando el registro de transacciones, sin realizar la copia del registro, por lo que no se debe especificar el dispositivo de copia

  El registro de transacciones, es un elemento de la base de datos que va creciendo progresivamente, por lo que se debe limpiar con cierta regularidad; para ello, podemos hacer una copia completa de la base de datos o truncar el registro con estas opciones.

El Código fuente 53 realiza una limpieza del registro de transacciones de la base de datos Pruebas

```
BACKUP LOG Pruebas WITH TRUNCATE_ONLY
```

Código fuente 53

Copia de seguridad de ficheros integrantes de una base de datos

Si nos encontramos ante una base de datos de gran tamaño, compuesta por varios ficheros, y no podemos realizar una copia de seguridad completa en una única sesión, es posible crear copias de seguridad separadas, de los ficheros que la componen, de forma que podemos distribuir una copia de seguridad completa en varias sesiones.

Supongamos para este caso, que disponemos de la base de datos Pruebas, creada según el Código fuente 54 mostrado a continuación.

```
CREATE DATABASE Pruebas
ON PRIMARY (NAME=PrincPruebas,
FILENAME='D:\MSSQL7\DATA\PRINCPRUEBAS.MDF'),

FILEGROUP Grp1Pruebas
(NAME=SegPruebas,
FILENAME='D:\MSSQL7\DATA\SEGPRUEBAS.NDF'),
(NAME=TercPruebas,
FILENAME='D:\MSSQL7\DATA\TERCPRUEBAS.NDF'),

FILEGROUP Grp2Pruebas
(NAME=CuarPruebas,
FILENAME='D:\MSSQL7\DATA\CUARPRUEBAS.NDF'),
(NAME=QuinPruebas,
FILENAME='D:\MSSQL7\DATA\QUINPRUEBAS.NDF')

LOG ON (NAME=PruebasLog,
FILENAME='D:\MSSQL7\DATA\PRUEBASLOG.LDF')
```

Código fuente 54

Como podemos comprobar, esta base de datos está compuesta por cinco ficheros de datos con el mismo nombre lógico que físico, y un fichero para el registro de transacciones. Los ficheros de datos se distribuyen en el fichero PrincPruebas, situado en el grupo primario de la base de datos (PRIMARY), y dos grupos de ficheros Gr1Pruebas y Grp2Pruebas, que contienen a los cuatro ficheros de datos restantes.

Durante el trabajo habitual con la base de datos, su ritmo crecimiento es elevado, por lo que realizar una copia de seguridad completa es complicado; sin embargo, al encontrarse repartida físicamente entre diversos ficheros y grupos, la tarea de copia de seguridad se facilita; veamos como realizar dicha copia.

Administrador corporativo

Desde la ventana de creación de la copia, en el apartado Copia de seguridad, marcaremos la opción Archivo y grupo de archivos, y pulsaremos el botón de selección de ficheros situado junto a esta opción, ver Figura 145 .

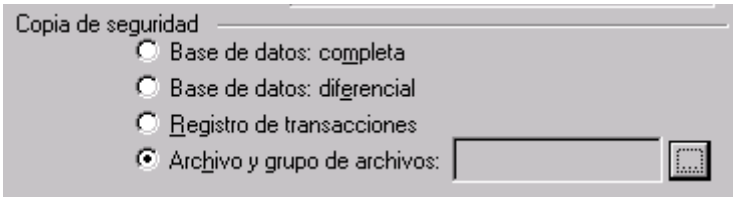


Figura 145. Seleccionar copia de seguridad de archivos y grupos.

A continuación, pulsaremos el botón de selección de ficheros situado junto a esta opción, que nos mostrará el árbol de distribución de ficheros y grupos de la base de datos. En este paso seleccionaremos el archivo o grupos que vamos a copiar en esta sesión, como se muestra en la Figura 146 , hemos seleccionado el grupo PRIMARY al completo, y el fichero CuarPruebas, perteneciente al grupo Grp2Pruebas.

Figura 146. Selección de ficheros y grupos para la copia de seguridad.

Después de aceptar esta ventana y realizar los pasos ya conocidos de selección de un dispositivo de destino, crearemos la copia de seguridad.

En sucesivas sesiones, podemos seguir copiando los ficheros restantes hasta completar la copia de toda la base de datos.

Instrucción BACKUP

Utilizaremos las opciones FILE y FILEGROUP de este comando, para realizar una copia de seguridad de una base de datos, basada en sus ficheros y grupos. La sintaxis de BACKUP para este aspecto de la copia es la siguiente:

```
BACKUP LOG NombreBD FicheroGrupoFich [,...FicheroGrupoFich]  TO
DispositivoCopia [,...DispositivoCopiaN]
```

En esta sintaxis, FicheroGrupoFich se sustituye por:

```
FILE = NombreFichLógico | FILEGROUP = NombreGrupoFichLógico
```

Siguiendo con el ejemplo de la base de datos Pruebas mostrado en este punto, utilizaremos esta instrucción para realizar una copia de su grupo de ficheros Grp1Pruebas, que vemos en el Código fuente 55.

```
BACKUP DATABASE Pruebas FILEGROUP='Grp1Pruebas'
TO DISK='D:\MSSQL7\BACKUP\CPSEGPRUEBAS10.BAK'
```

Código fuente 55

Empleando este método de copia, podemos indicar hasta un máximo de 16 ficheros o grupos de ficheros en cada sesión de copia. Es importante resaltar en este caso, la necesidad de establecer un sistema que de forma cíclica copie la base de datos en varias sesiones, ya que si sólo se copia una parte de los ficheros que componen la base de datos, corremos el riesgo de que al restaurar la base de datos, no mantenga su coherencia por no haber copiado ciertos ficheros.

Otro medio para mantener la coherencia de la base de datos, es hacer una copia de seguridad del registro de transacciones, por si ocurre un problema que impida tener un conjunto de copias de todos los ficheros que componen la base de datos.

Consideraciones sobre copias de seguridad de archivos e índices

Cuando realizamos copias de seguridad en archivos o grupos de archivos de una base de datos que contiene índices, debemos tener en cuenta las siguientes situaciones, en función de que la tabla y su índice asociado estén situados o no en el mismo grupo de archivos:

- Si la tabla y el índice se encuentran en el mismo grupo, la copia de seguridad se hará de todo el grupo de archivos al completo.
- Si la tabla y el índice están en distintos grupos de archivos, deberemos hacer la copia de seguridad de todos estos grupos de archivos unitariamente.

Programar una copia de seguridad

Al realizar una copia de seguridad desde el Administrador corporativo, podemos establecer que dicha copia no se realice inmediatamente, sino en un momento determinado, o a intervalos periódicos. En la ventana de configuración de la copia, marcaremos la casilla Programación, del apartado Programar, junto a la que se muestra la programación actual definida por defecto para la copia de seguridad. Para variar la programación definida, pulsaremos el botón que se encuentra en este apartado, junto a la descripción de la programación establecida, ver Figura 147.

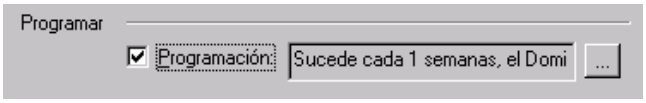


Figura 147. Valores de programación, en la ventana de propiedades de la copia de seguridad.

Realizado este paso, se abrirá la ventana de programación de la copia, en la que podremos establecer los diferentes tipos de programación existentes: en una fecha determinada, cada cierto periodo de tiempo, etc.

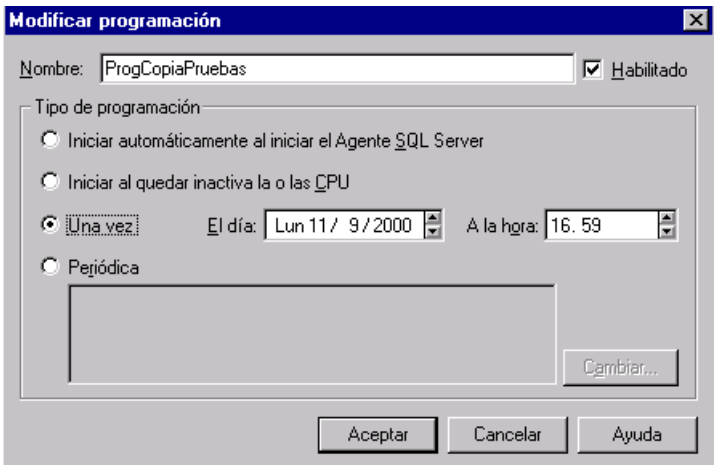


Figura 148. Programación de una copia de seguridad.

La Figura 148 muestra una programación para una base de datos que será realizada en una fecha y hora determinadas; este proceso será realizado internamente por SQL Server en segundo plano al cumplirse el plazo establecido.

Realizar copias de seguridad en cinta

La cinta streamer es un soporte muy difundido para la creación de copias de seguridad debido a su alta capacidad de almacenamiento, seguridad y bajo coste.

Si planeamos realizar nuestras copias de seguridad en este tipo de dispositivos, debemos tener en cuenta que la unidad lectora/grabadora de cinta debe estar conectada al servidor físico que almacena los datos de SQL Server.

Opciones de manipulación de cinta

La Tabla 9 muestra una relación de las opciones utilizadas en la instrucción de copia de seguridad, para las distintas operaciones que tengamos que realizar sobre una cinta.

Opción	**Descripción**
UNLOAD	Rebobina y descarga la cinta al terminar la copia de seguridad.
NOUNLOAD	Evita el rebobinado y descarga de la cinta que realiza automáticamente SQL Server al terminar la copia.
BLOCKSIZE	Permite definir el tamaño del bloque en bytes; por defecto SQL Server utiliza un tamaño por defecto.
FORMAT	Escribe un nuevo encabezado en la cinta, eliminando la información existente en la misma, quedando por lo tanto, eliminadas todas las copias que contuviese.
SKIP	Se utiliza para saltar los encabezados, ignorando las etiquetas ANSI de la cinta. Estas etiquetas pueden tener información como la fecha de caducidad de la cinta.
NOSKIP	Obliga a realizar la lectura de las etiquetas ANSI existentes en la cinta.
RESTART	Si una copia de seguridad implica el uso de varias cintas, mediante esta opción podemos continuar el proceso de copia al insertar una nueva cinta.

Tabla 9

16

Restaurar copias de seguridad

Situaciones en las que deberemos restaurar una base de datos.

Al producirse una circunstancia no habitual sobre una base de datos, que motive una posible pérdida de información, ya sea un fallo en el disco del servidor, caída del sistema por corte de fluido eléctrico, etc., tendremos que proceder a restaurar las copias de seguridad que previamente hayamos realizado de dicha base de datos, para devolverla a un estado coherente.

Ante fallos graves del sistema, SQL Server dispone de un proceso automático de recuperación, que intentará devolver el servidor de datos a un estado consistente; durante este proceso, SQL Server verifica el registro de transacciones partiendo del último punto de comprobación en el que sucedió el problema, escribe las transacciones confirmadas del registro y rechaza las transacciones no confirmadas, eliminándolas.

A pesar de tener este mecanismo de seguridad, debemos disponer de las correspondientes copias de seguridad, realizadas regularmente, para asegurarnos que al restaurar la base de datos, el grado de información perdida sea el menor posible.

Restaurar una base de datos

Podemos recobrar una base de datos dañada empleando el Administrador corporativo o el comando RESTORE. En este apartado, se mostrará la forma más básica de realizar un proceso de restauración,

de forma que el lector pueda introducirse progresivamente en las diferentes opciones existentes para recuperar una base de datos.

Administrador corporativo

Empleando esta utilidad de SQL Server, podemos acceder al proceso de restauración seleccionando la base de datos a restaurar y haciendo clic sobre el panel derecho, apartado Copia de seguridad, opción Restaurar la base de datos, como vemos en la Figura 149.

Figura 149. Selección de la opción de restauración de base de datos.

También podemos seleccionar la opción de menú *Herramientas+Restaurar base de datos*, existente en la ventana de esta aplicación. Ambas vías de acceso nos abrirán la ventana de opciones de restauración mostrada en la Figura 150.

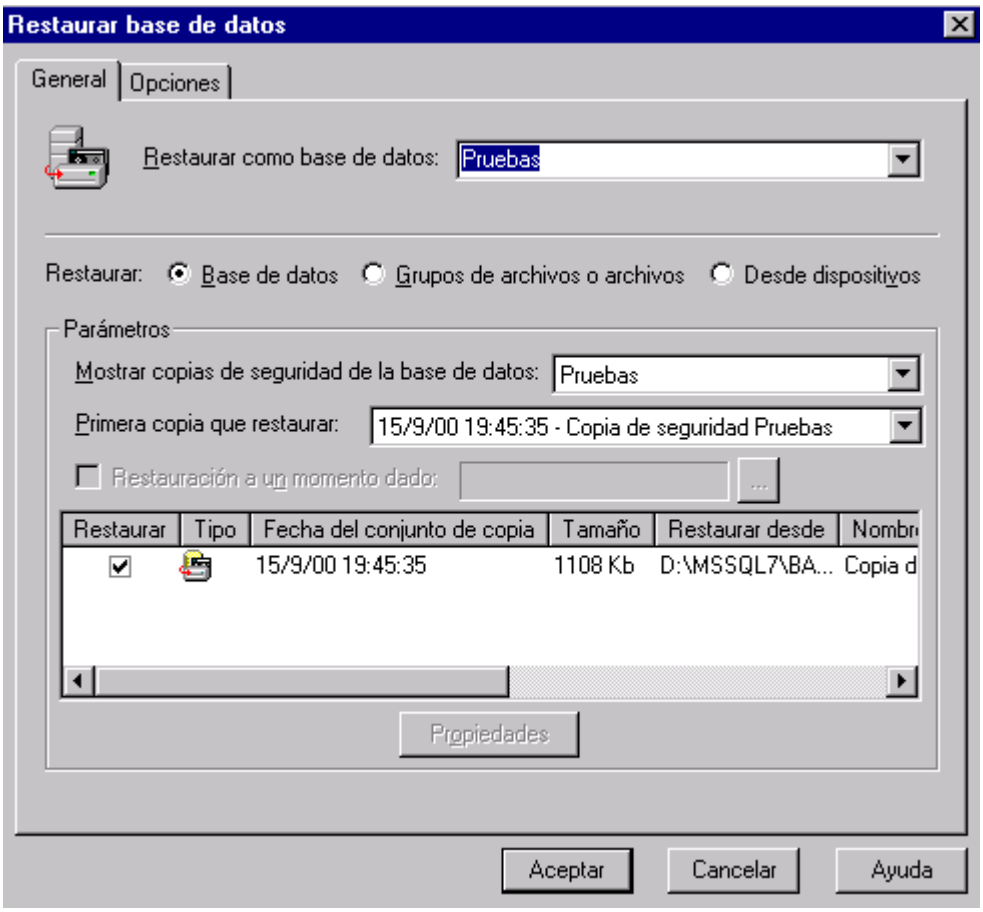


Figura 150. Ventana para restaurar bases de datos.

Debemos seleccionar en primer lugar, en la lista desplegable Restaurar como base de datos, la base de datos sobre la que vamos a realizar la restauración. En el apartado Restaurar, elegiremos qué elemento vamos a restaurar: una base de datos completamente en un único paso, los archivos o grupos en los que está distribuida la base de datos, o seleccionando los dispositivos que contienen las copias de seguridad de la base de datos.

En el apartado Parámetros, seleccionaremos en la lista Mostrar copias de seguridad de la base de datos, el nombre de la base de datos, mostrándose en la lista que hay debajo, Primera copia que restaurar, las copias realizadas de esta base de datos; cada vez que seleccionemos una copia, podremos ver un detalle de los elementos que la componen en el listado que hay debajo, y si necesitamos un detalle por línea, pulsaremos el botón Propiedades.

Seleccionaremos la copia que queramos restaurar y pulsaremos Aceptar para comenzar el proceso de restauración.

Instrucción RESTORE

Este comando, unido a su conjunto de opciones, nos permiten restaurar copias de seguridad de una forma cómoda y flexible, su sintaxis más básica se muestra a continuación.

```
RESTORE DATABASE NombreBD FROM DispositivoCopia
[,...DispositivoCopiaN] [WITH [FILE=NumArchivo]]
```

DispositivoCopia se sustituye por:

```
{DispositivoCopiaPermanente} | {DISK|TAPE|PIPE =
'DispositivoCopiaTemporal'}
```

- FILE. En dispositivos que contienen más de una copia, esta opción nos permite especificar cuál de las copias del dispositivo, ordenadas cronológicamente, queremos restaurar.

El Código fuente 56, restaura una base de datos desde un dispositivo de copia permanente; como el dispositivo contiene varias copias, se selecciona la segunda.

```
RESTORE DATABASE Pruebas FROM CopiaSegPruebas WITH FILE=2
```

Código fuente 56

Comprobaciones y tareas previas a la restauración de una base de datos

Antes de proceder a restaurar una base de datos, SQL Server realiza una serie de operaciones de comprobación que es recomendable que sean apoyadas por otras tareas complementarias por parte del administrador del sistema.

Realizadas por SQL Server

SQL Server comprueba de forma automática, que la copia de seguridad que vamos a restaurar, reúne todas las condiciones necesarias para recuperar la información de forma consistente.

Entre las situaciones que generan un error e impiden la restauración, encontramos las siguientes:

- Intentar restaurar una copia de seguridad, sobre una base de datos distinta del servidor de datos.
- Faltan dispositivos de copia o archivos, ya que la restauración debe realizarse empleando todos los elementos utilizados en la copia de seguridad.
- Los archivos contenidos en la copia de seguridad son diferentes de los que se encuentran en el servidor de datos.

En la Figura 151 se muestra el error emitido por el Administrador corporativo al intentar restaurar una copia de seguridad de la base de datos Pruebas sobre la base de datos Almacenes.

Figura 151. Error de restauración.

Realizadas por el administrador del sistema

Durante el espacio de tiempo transcurrido entre la última copia de seguridad, y el proceso de restauración, los usuarios de la base de datos han podido seguir actualizando datos en el sistema. Si bien es cierto que SQL Server dispone de mecanismos de recuperación automática, podría suceder que al restaurar la base de datos ese conjunto de información modificada se perdiera. La Figura 152 muestra un esquema de esta situación.

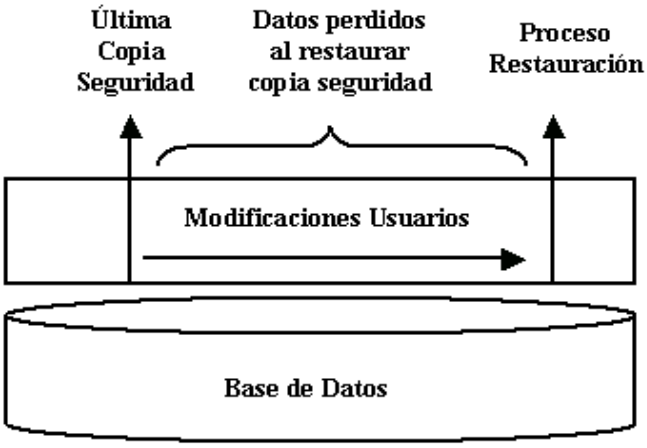


Figura 152. Información perdida entre última copia de seguridad y restauración de base de datos.

Para evitarlo, debemos limitar el acceso de los usuarios a la base de datos, de forma que nos aseguremos que no se van a producir operaciones de actualización sobre la misma durante el proceso de restauración.

El administrador de SQL Server o un usuario con los permisos adecuados (que pertenezca a la función del servidor sysadmin o db_owner), debe establecer que la base de datos sólo sea utilizable por el DBO.

Empleando el Administrador corporativo, debemos hacer clic con el botón derecho sobre la base de datos y seleccionar la opción Propiedades del menú contextual. En la ventana de propiedades de la base de datos accederemos a la pestaña Opciones, y en el apartado Acceso, marcaremos la propiedad Sólo para uso del DBO, ver Figura 153.

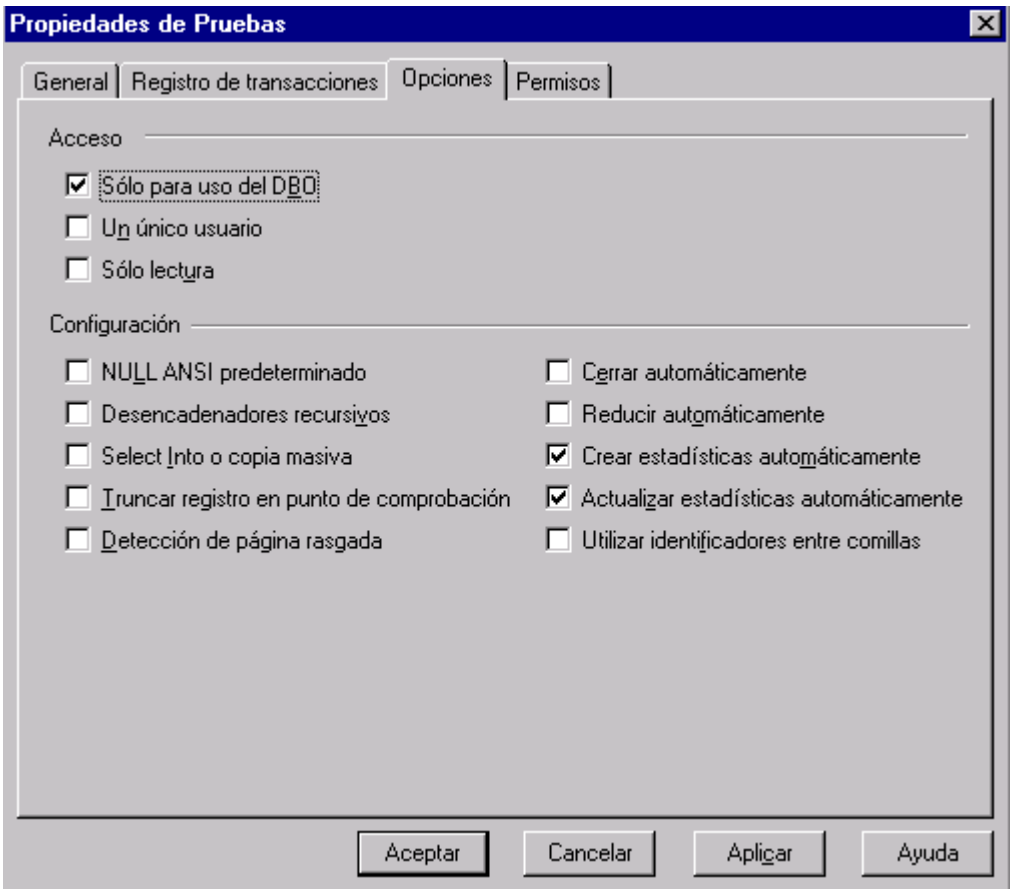


Figura 153. Establecer la base de datos sólo para uso del DBO.

Disponemos también, para realizar esta misma operación, del procedimiento almacenado del sistema sp_dboption, que nos permite configurar las opciones de la base de datos. Su sintaxis se muestra a continuación.

```
sp_dboption ['NombreBD'] [,'NombreOpción'] [,'Valor']
```

- NombreBD. Nombre de la base de datos a configurar.
- NombreOpción. Nombre de la opción que se va a cambiar

- Valor. Una cadena conteniendo TRUE para activar la opción, o FALSE para desactivarla.

Si no se utiliza ningún parámetro, este procedimiento almacenado, devuelve una lista de todas las opciones disponibles de la base de datos. En la documentación de SQL Server se describe con detalle la función de cada una de estas opciones.

Si sólo se envía el parámetro del nombre de la base de datos, se devuelven las opciones establecidas para esa base de datos.

Si se pasa además del nombre de la base de datos, el nombre de una opción, se devuelve su valor actual.

Especificando todas las opciones del procedimiento, se establece el valor correspondiente sin devolver ningún resultado.

El Código fuente 57 muestra el establecimiento de la opción de uso de la base de datos para el DBO.

```
EXEC sp_dboption 'Pruebas', 'dbo use only', 'TRUE'
```

Código fuente 57

Para volver a poner la base de datos a disposición de todos los usuarios utilizando este procedimiento almacenado, pasaríamos como último parámetro el valor FALSE.

La instrucción RESTORE también permite especificar este valor a través de su opción DBO_ONLY.

```
RESTORE DATABASE NombreBD FROM DispositivoCopia
[,...DispositivoCopiaN] [WITH [DBO_ONLY]]
```

Finalmente, debemos hacer una copia de seguridad del registro de transacciones, ya que en caso contrario, perderemos las modificaciones realizadas entre la última copia de seguridad del registro de transacciones y el momento de desconexión de la base de datos.

Acceso a la información de un dispositivo de copia

SQL Server permite realizar la lectura del contenido de una copia de seguridad, de forma que nos aseguremos que dicha copia sea la que realmente debemos restaurar.

Dentro del tema dedicado a la realización de copias de seguridad, en los apartados sobre dispositivos, medios y estructura interior de una copia de seguridad, se explicaba el modo de visualizar esta información desde el Administrador corporativo.

Adicionalmente, SQL Server proporciona el conjunto de instrucciones RESTORE, descritas a continuación, que nos permitirán en modo comando, disponer de estos interesantes datos, de cara a la correcta restauración de la base de datos.

Nótese que estas instrucciones no llevan a cabo ningún proceso de restauración, limitándose a devolver un conjunto de resultados con información sobre una copia de seguridad.

RESTORE HEADERONLY

Devuelve información sobre la cabecera del dispositivo de copia de seguridad pasado como parámetro.

```
RESTORE HEADERONLY FROM DispositivoCopia
```

- DispostivoCopia. Nombre de un dispositivo de copia temporal o permanente.

El resultado de la ejecución de este comando devuelve entre otros datos, el nombre de las copias de seguridad contenidas en el dispositivo, descripción si existe, método de copia de seguridad, fecha de caducidad, usuario que realizó la copia, base de datos copiada, etc.

El Código fuente 58 emplea esta instrucción sobre un dispositivo de copia de seguridad que contiene tres copias. El resultado lo vemos en la Figura 154.

```
RESTORE HEADERONLY FROM CopiaSegPruebas
```

Código fuente 58

BackupName	BackupDescription	BackupType	ExpirationDate	Comp...	Pos...
Copia1Pruebas	NULL	1	NULL	0	1
Copia2Pruebas	NULL	1	NULL	0	2
Copia3Pruebas	NULL	1	NULL	0	3

Figura 154. Resultado de la instrucción RESTORE HEADERONLY.

RESTORE FILELISTONLY

Devuelve información sobre los ficheros de la base de datos o registro que se encuentran en el dispositivo de copia de seguridad pasado como parámetro.

```
RESTORE FILELISTONLY FROM DispositivoCopia [WITH [FILE =
NumArchivo]]
```

- DispostivoCopia. Nombre de un dispositivo de copia temporal o permanente.
- FILE. Si el dispositivo contiene varias copias, con esta opción especificamos de cuál de ellas vamos a recuperar información.

El resultado de la ejecución de este comando devuelve entre otros datos, el nombre lógico y físico de los ficheros contenidos en el dispositivo, tipo de fichero, si pertenece a un grupo, tamaño de la copia y tamaño máximo permitido para el fichero. El Código fuente 59 muestra un ejemplo de uso de esta instrucción.

```
RESTORE FILELISTONLY FROM CopiaSegPruebas
```

Código fuente 59

El resultado obtenido se muestra en la Figura 155.

LogicalName	PhysicalName	Type	FileGroupName	Size	MaxSize
PrincPruebas	D:\MSSQL7\DATA\PRINC...	D	PRIMARY	786432	35184372080640
SegPruebas	D:\MSSQL7\DATA\SEGPR...	D	Grp1Pruebas	1048576	35184372080640
TercPruebas	D:\MSSQL7\DATA\TERCP...	D	Grp1Pruebas	1048576	35184372080640
CuarPruebas	D:\MSSQL7\DATA\CUARP...	D	Grp2Pruebas	1048576	35184372080640
QuinPruebas	D:\MSSQL7\DATA\QUINP...	D	Grp2Pruebas	1048576	35184372080640
PruebasLog	D:\MSSQL7\DATA\PRUEB...	L	NULL	1048576	35184372080640

Figura 155. Resultado de la instrucción RESTORE FILELISTONLY.

RESTORE LABELONLY

Devuelve información sobre el medio de copia de seguridad que se encuentra en el dispositivo de copia de seguridad pasado como parámetro.

```
RESTORE LABELONLY FROM DispositivoCopia
```

- DispostivoCopia. Nombre de un dispositivo de copia temporal o permanente.

El resultado de la ejecución de este comando devuelve entre otros datos, el nombre del medio de copia, identificador, número de familias que componen el medio, número de secuencia de la familia, descripción del medio, etc.

El Código fuente 60 muestra un ejemplo de uso de esta instrucción.

```
RESTORE LABELONLY FROM Disp2Pruebas
```

Código fuente 60

El resultado obtenido se muestra en la Figura 156.

MediaName	MediaSetId	Family...	FamilySequenceN...	MediaFamilyId	MediaSeq...
PruebasGeneral	NULL	1	1	{66D934A6-0000-00...	1

Figura 156. Resultado de la instrucción RESTORE LABELONLY.

RESTORE VERIFYONLY

Realiza una comprobación general de la copia de seguridad, exceptuando la estructura de los datos de la copia.

```
RESTORE VERIFYONLY FROM DispositivoCopia [,…DispositivoCopiaN] [WITH
[FILE = NumArchivo]]
```

- DispostivoCopia. Nombre de un dispositivo de copia temporal o permanente.
- FILE. Si el dispositivo contiene varias copias, con esta opción especificamos cuál de ellas se comprobará..

La ejecución de este comando devuelve un mensaje informativo sobre el estado de la copia de seguridad.

El Código fuente 61 muestra un ejemplo de uso de esta instrucción.

```
RESTORE VERIFYONLY FROM CopiaSegPruebas
```

Código fuente 61

Restaurar varias copias en un solo paso

Hay situaciones en que para poder devolver una base de datos a su estado normal después de un problema, es necesario restaurar el conjunto de copias realizadas a lo largo de un periodo de tiempo; por ejemplo, la copia de seguridad completa, las copias diferenciales y del registro de transacciones existentes. Empleando el Administrador corporativo, esta en principio, tediosa tarea, puede ser resuelta fácilmente desde la ventana de restauración de copias de seguridad.

Supongamos la siguiente situación: tenemos una base de datos llamada Contabilidad, de la que hemos generado una copia de seguridad completa, otra diferencial y una del registro de transacciones. En cierto momento, ocurre un fallo que provoca la pérdida de algunas tablas de la base de datos, que podemos recuperar restaurando las copias antes mencionadas.

Accederemos por tanto a la utilidad de restauración de copias de seguridad del Administrador corporativo, y una vez seleccionada la base de datos y la copia que necesitamos restaurar, se mostrará la ventana de la Figura 157.

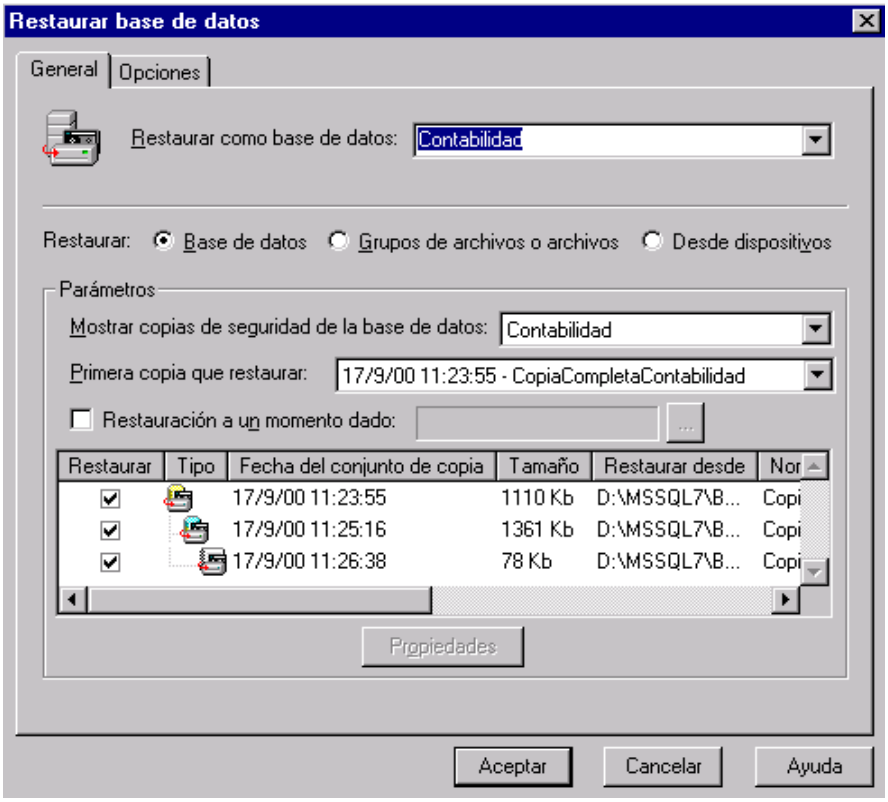


Figura 157. Restauración de varias copias en un solo paso.

Observe el lector cómo en la lista de detalle, las copias están dispuestas en orden jerárquico. Si marcamos en la columna Restaurar, la copia del registro de transacciones, automáticamente se marcarán las anteriores; esto evita que por error, intentemos restaurar sólo la copia del registro de transacciones sin restaurar las anteriores, lo que provocaría un problema de coherencia en los datos.

Como en este proceso de restauración, están implicadas varias copias de seguridad, podemos establecer que cada vez que finalice una copia y vaya a restaurarse la siguiente, SQL Server nos avise. Esto lo conseguimos, en la pestaña Opciones de la ventana de restauración, marcando la opción *Preguntar antes de restaurar cada copia de seguridad*, ver Figura 158.

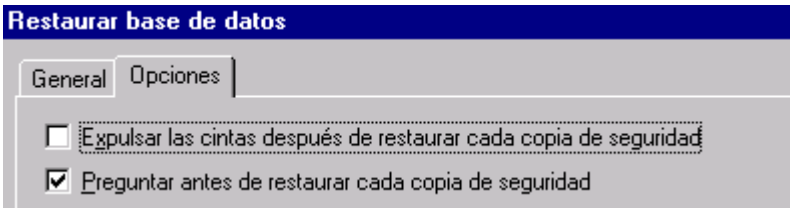


Figura 158. Selección de la opción de avisar antes de restaurar una copia de seguridad.

De esta forma, obtendríamos el mensaje de la Figura 159 entre cada restauración.

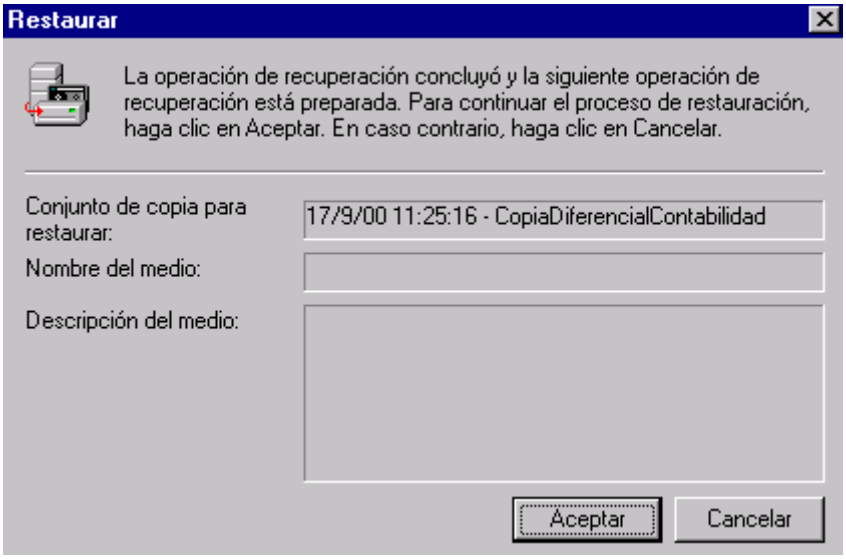


Figura 159. Mensaje de aviso al finalizar restauración de copia y comenzar nueva restauración.

Restaurar varias copias paso a paso y recuperación de una base de datos

El proceso de recuperación en una base de datos consiste en una operación por la cual SQL Server da por finalizadas las tareas de restauración de una base de datos, devolviendo su información a un estado coherente.

La lógica de esta operación la encontramos cuando tenemos que restaurar una base de datos a partir de varias copias de seguridad: completa, diferencial y registro de transacciones; en este tipo de casos, deberemos ir restaurando todas las copias sin recuperar, hasta que lleguemos a la última copia, en la

que sí especificaremos la opción de recuperar, con lo que SQL Server devolverá la base de datos a un estado consistente.

Administrador corporativo

Si utilizamos la técnica mostrada en el apartado anterior, basada en restaurar varias copias en un solo paso, el proceso de recuperación es automático, produciéndose finalizada la operación de restauración.

Sin embargo, puede que sea un proceso en el cuál, el administrador necesite restaurar manualmente cada una de las copias hasta dejar la base de datos en un estado de coherencia determinado. En este caso, una vez abierta la ventana de restauración de copias de seguridad, y seleccionada la base de datos a restaurar, pulsaremos la opción *Desde dispositivos* del apartado Restaurar, que cambiará la apariencia de esta ventana por la mostrada en la Figura 160.

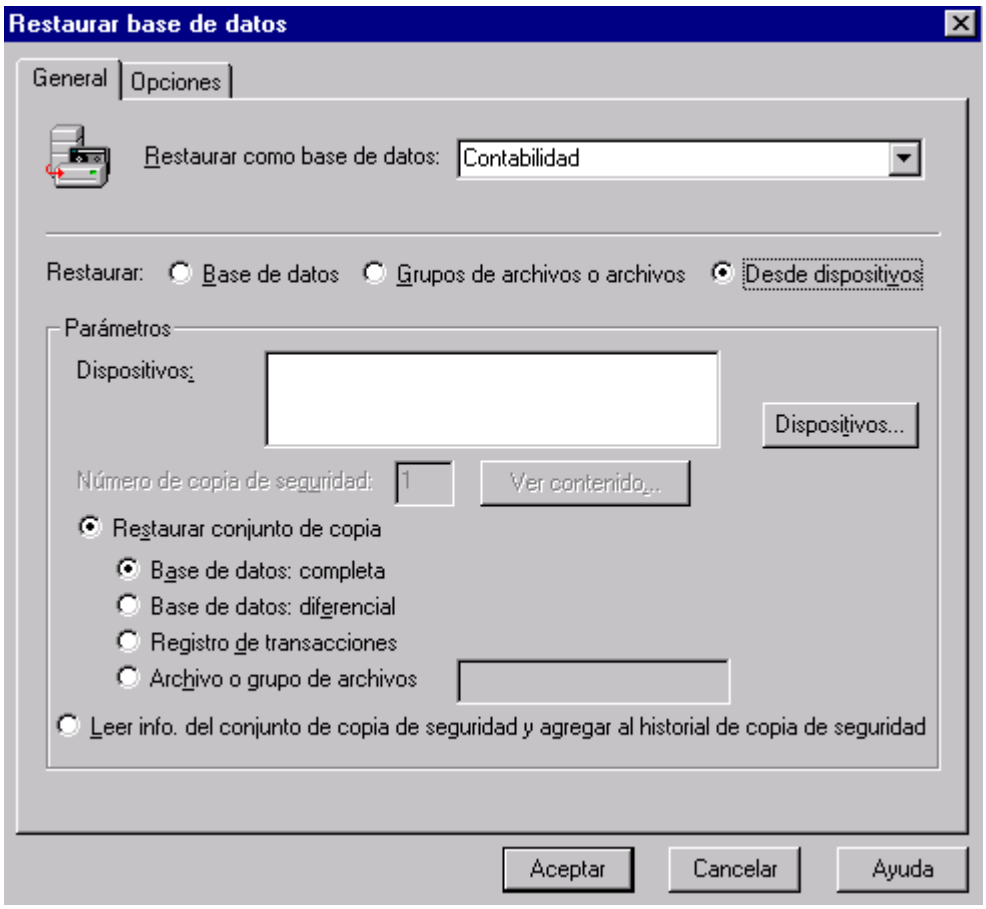


Figura 160. Restaurar una base de datos desde dispositivos.

Vamos a proceder a restaurar la misma base de datos del anterior apartado: Contabilidad, y las mismas copias: completa, diferencial y registro de transacciones; de forma que el lector pueda percibir las diferencias entre una y otra técnica de restaurado. Los siguientes apartados nos muestran el orden de realización de este proceso.

Restaurar la copia de seguridad completa

Haremos clic sobre las opciones *Restaurar conjunto de copia* y *Base de datos: completa*; a continuación pulsaremos el botón Dispositivos, que nos dará acceso a la ventana *Elegir dispositivos para restaurar*, véase la Figura 161.

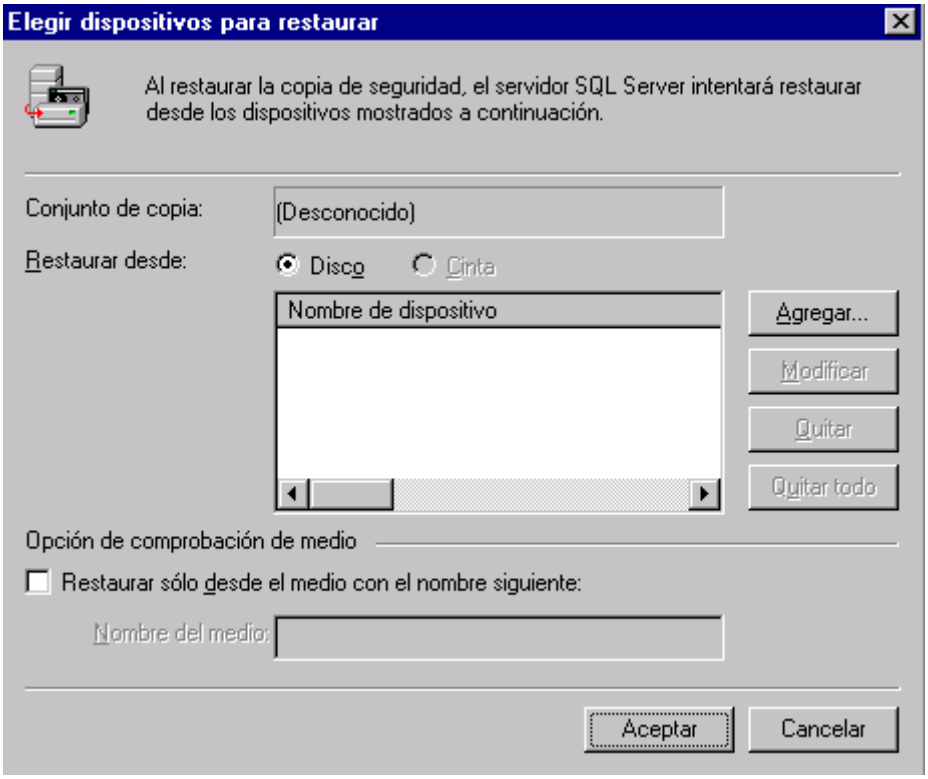


Figura 161. Elegir dispositivo que contiene copia a restaurar.

Pulsando el botón Agregar, elegiremos en una ventana de selección de dispositivos, el dispositivo permanente DispCompletaConta, que contiene la copia de seguridad completa, Figura 162.

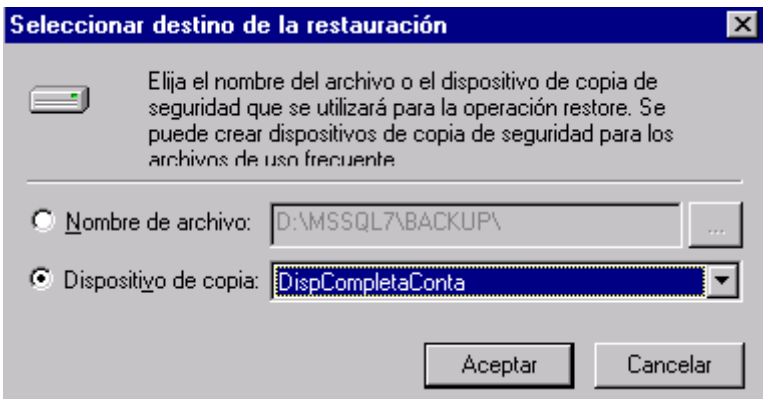


Figura 162. Selección de un dispositivo de copia para restaurar.

Esta acción, depositará este dispositivo en la lista de la ventana *Elegir dispositivos para restaurar;* podemos agregar más dispositivos si es necesario, quitarlos de la lista, etc. Finalizada la selección de dispositivos, pulsaremos Aceptar, regresando a la ventana de restauración.

Debido a que necesitamos restaurar varias copias para devolver la base de datos al estado necesario, es preciso indicar en este paso de restauración, que SQL Server no recupere de momento la base de datos, para lo que haremos clic en la pestaña Opciones de esta ventana, y en el apartado *Estado al concluir la recuperación*, pulsaremos la opción *Base de datos no operativa. Capaz de restaurar nuevos registros de transacciones*, ver Figura 163.

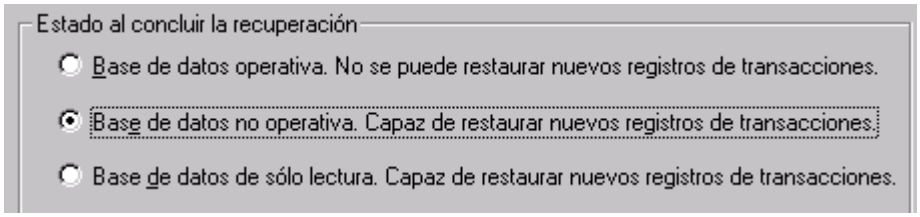


Figura 163. Opciones de recuperación al finalizar una restauración.

En este momento, procederemos a restaurar la copia completa pulsando Aceptar. Al haber impedido la recuperación después de restaurar la copia, la base de datos se hallará en un estado de carga, que nos impedirá cualquier operación hasta que no completemos la restauración de todas las copias necesarias, ver Figura 164.

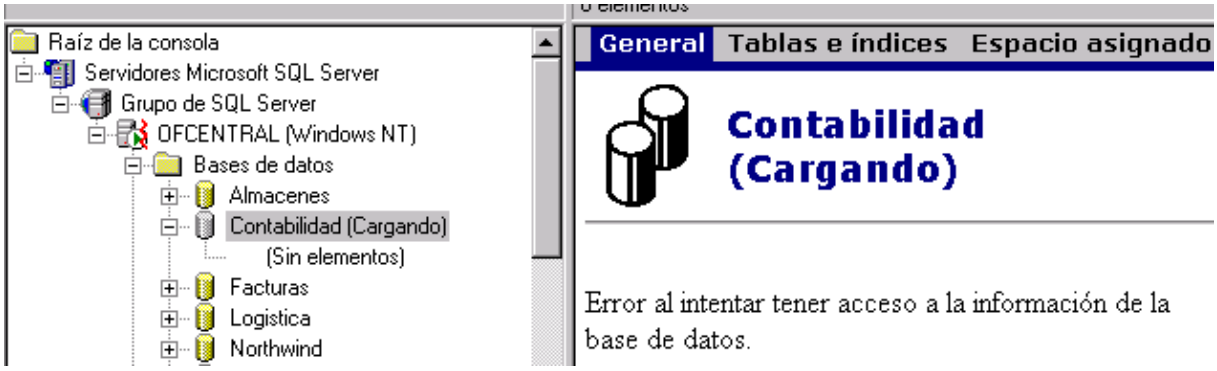


Figura 164. Base de datos en estado de carga.

Restaurar la copia de seguridad diferencial

Este tipo de copias contienen sólo la información de la base de datos que ha sufrido cambios desde la última copia de seguridad completa.

Siguiendo con nuestro ejemplo sobre la base de datos Contabilidad, volveremos a la ventana para restaurar bases de datos desde dispositivos, pulsando la opción *Base de datos: diferencial*, seleccionando el dispositivo que contiene la copia de seguridad diferencial de igual modo que se explicó en el apartado anterior. La única diferencia es que en este caso, la copia de seguridad diferencial está en un dispositivo temporal (archivo físico), con el nombre CPDIFCONTA.BAK, por lo que tendremos que seleccionar este archivo en lugar de un nombre lógico de dispositivo, como muestra la Figura 165.

A partir de aquí el proceso será igual que al restaurar la copia completa: aceptaremos la ventana de selección de dispositivos e indicaremos que la base de datos no esté operativa en la opción de recuperación. Pulsando Aceptar en la ventana de restauración, procederemos a restaurar esta copia de seguridad diferencial, permaneciendo la base de datos en el estado de carga.

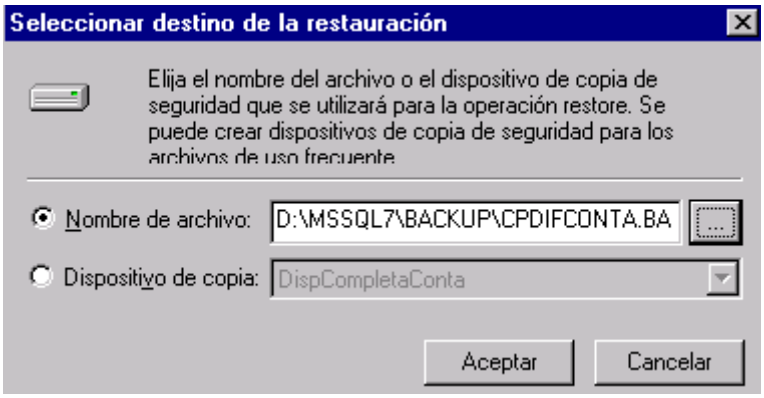


Figura 165. Selección de dispositivo temporal para restaurar copia de seguridad.

Restaurar la copia de seguridad del registro de transacciones

Este tipo de copia captura toda la actividad de la base de datos desde la última copia de seguridad realizada, ya sea completa o diferencial.

En el ejemplo con la base de datos Contabilidad, debemos pulsar en la ventana para restaurar, la opción *Registro de transacciones*, después añadiremos el dispositivo permanente DispLogConta de la forma explicada en los anteriores puntos, y en este caso, como se trata de la última copia de la base de datos que tenemos que restaurar , pulsaremos la opción *Base de datos operativa. No se puede restaurar nuevos registros de transacciones*, con lo SQL Server procederá a la recuperación de la base de datos al finalizar de restaurar esta copia, permitiendo ya su acceso a los usuarios. La Figura 166 muestra la ventana con toda las propiedades, justo antes de restaurar este registro de transacciones.

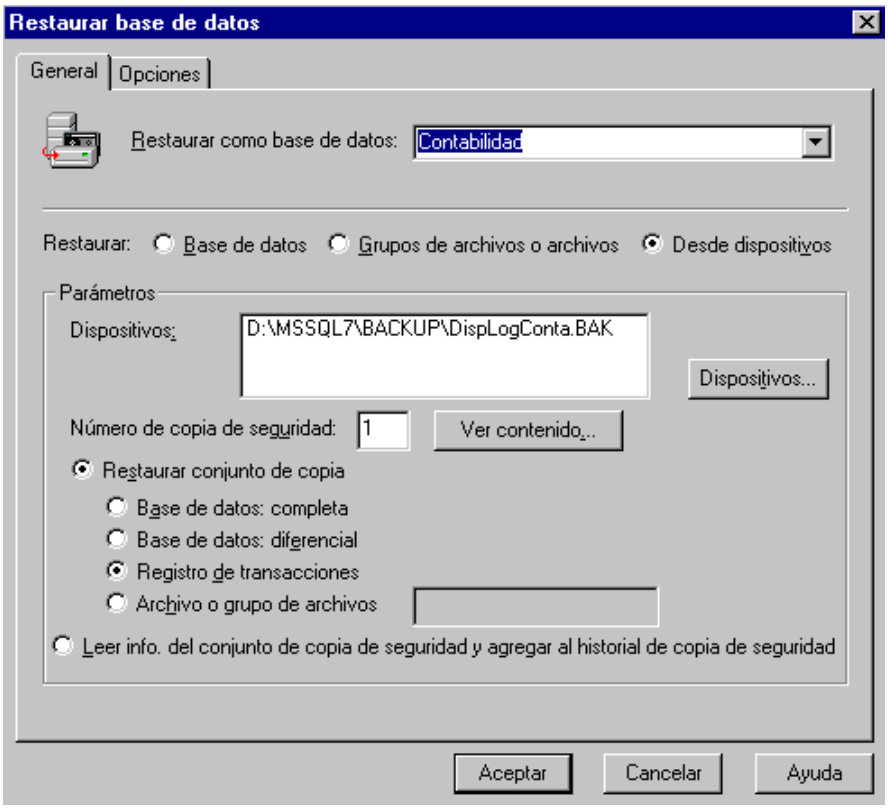


Figura 166. Ventana de restauración de una copia de seguridad del registro de transacciones.

Instrucción RESTORE

Al utilizar esta instrucción con cada una de las diferentes copias de seguridad, tendremos que especificar el modo de recuperación empleando las opciones RECOVERY y NORECOVERY.

```
RESTORE DATABASE|LOG NombreBD FROM DispositivoCopia
[,…DispositivoCopiaN] [WITH [RECOVERY | NORECOVERY]]
```

- RECOVERY. Valor predeterminado. Debe utilizarse al restaurar una copia de seguridad completa de una base de datos o con la última copia de seguridad del registro de transacciones. Aplica todas las transacciones del registro de transacciones, quedando la base de datos lista para utilizarse.

- NORECOVERY. Debe utilizarse al restaurar un conjunto de copias, salvo la última copia. Esta opción no aplica ni deshace las transacciones del registro, permaneciendo la base de datos en estado de carga hasta que se indique su recuperación.

Aplicando estas opciones al ejemplo anterior, en primer lugar tendríamos que restaurar la copia completa de la base de datos y la copia diferencial, ambas sin recuperar; finalizando con la restauración de la copia del registro de transacciones, en la que sí efectuaríamos el proceso de recuperación; observe el lector, que para restaurar el registro de transacciones, la instrucción utilizada es RESTORE LOG. El Código fuente 62 muestra las instrucciones implicadas en el proceso y su orden de ejecución.

```
RESTORE DATABASE Contabilidad FROM DispCompletaConta WITH NORECOVERY

RESTORE DATABASE Contabilidad FROM DISK='D:\MSSQL7\BACKUP\CPDIFCONTA.BAK' WITH
NORECOVERY

RESTORE LOG Contabilidad FROM DispLogConta WITH RECOVERY
```

Código fuente 62

Reemplazar una base de datos

La acción de reemplazar una base de datos, consiste en restaurar una copia de seguridad sobre una base de datos distinta de la que originalmente se obtuvo la copia.

Administrador corporativo

En la ventana de restauración, debemos seleccionar de la lista Restaurar como base de datos el nombre de la base de datos sobre la que vamos a realizar el proceso de restauración. Después elegiremos una base de datos distinta de la lista Mostrar copia de seguridad de la base de datos y la copia de seguridad correspondiente.

En la Figura 167 se va a proceder a reemplazar el contenido de la base de datos Contabilidad, por el de la base de datos Factura.

Para finalizar, haremos clic en la pestaña Opciones de esta ventana, y marcaremos la opción Forzar restauración sobre la base de datos existente, ver Figura 168, que anula la comprobación de seguridad que realiza SQL Server para evitar restauraciones sobre bases de datos distintas.

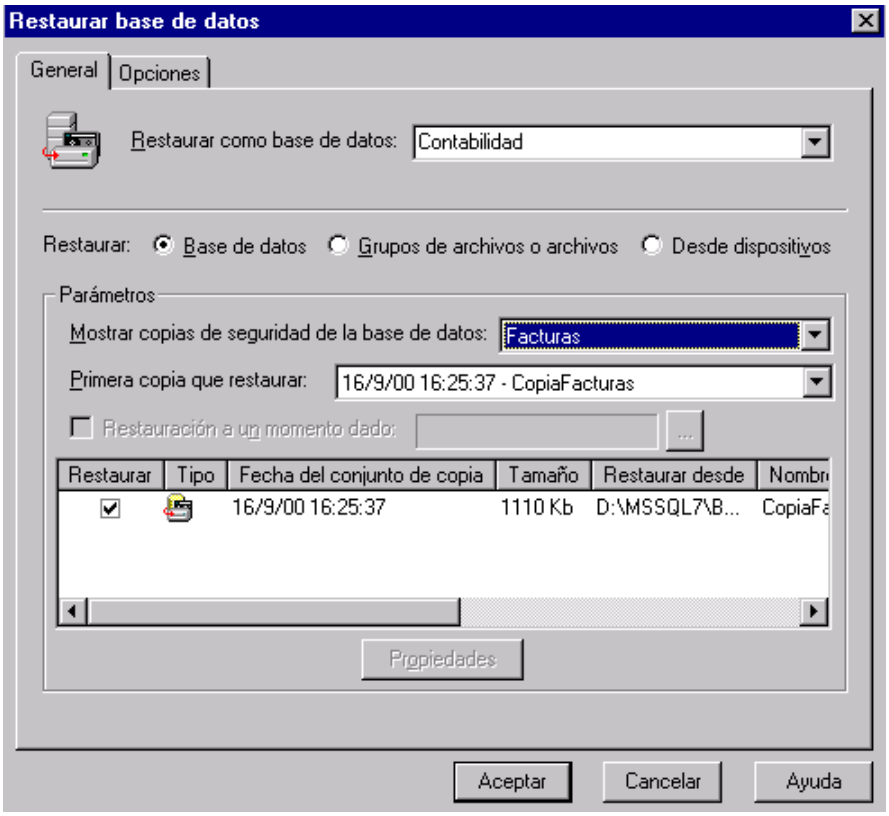


Figura 167. Restauración en una base de datos distinta de la original.

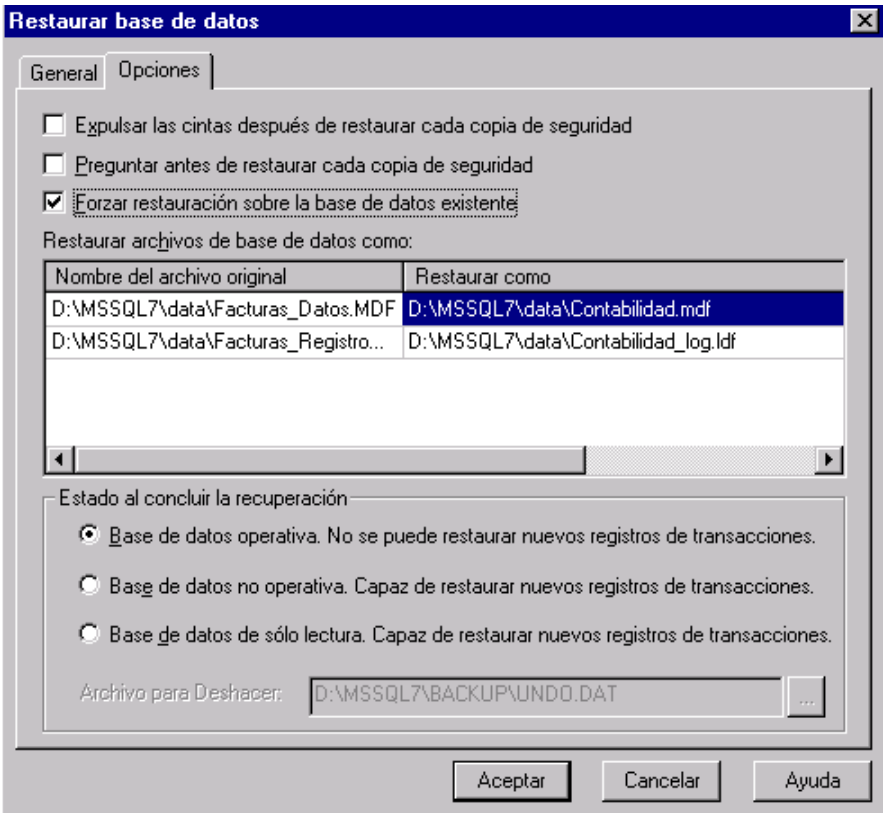


Figura 168. Forzar restauración entre distintas bases de datos.

Instrucción RESTORE

La opción REPLACE de este comando, nos permite evitar la comprobación de seguridad de SQL Server al restaurar, permitiéndonos sustituir el contenido de una base de datos por el que se encuentra en una copia de seguridad.

```
RESTORE NombreBD FROM DispositivoCopia [,…DispositivoCopiaN] [WITH
[REPLACE]]
```

El Código fuente 63 restaura la base de datos Contabilidad con la copia de seguridad de la base de datos Facturas.

```
RESTORE DATABASE Contabilidad FROM DispCopFacturas WITH REPLACE
```

Código fuente 63

Cambiar la ubicación de los datos al restaurar

Al realizar un proceso de restauración, podemos especificar una ruta y/o nombre distinto del original para los ficheros que contienen la base de datos.

Administrador corporativo

En la ventana de restauración de SQL Server, pulsaremos la pestaña Opciones, y en el apartado Restaurar archivos de base de datos como, especificaremos la nueva ruta en la columna Restaurar como. La Figura 169 muestra cómo se escribe una nueva ubicación para los archivos de la base de datos Facturas.

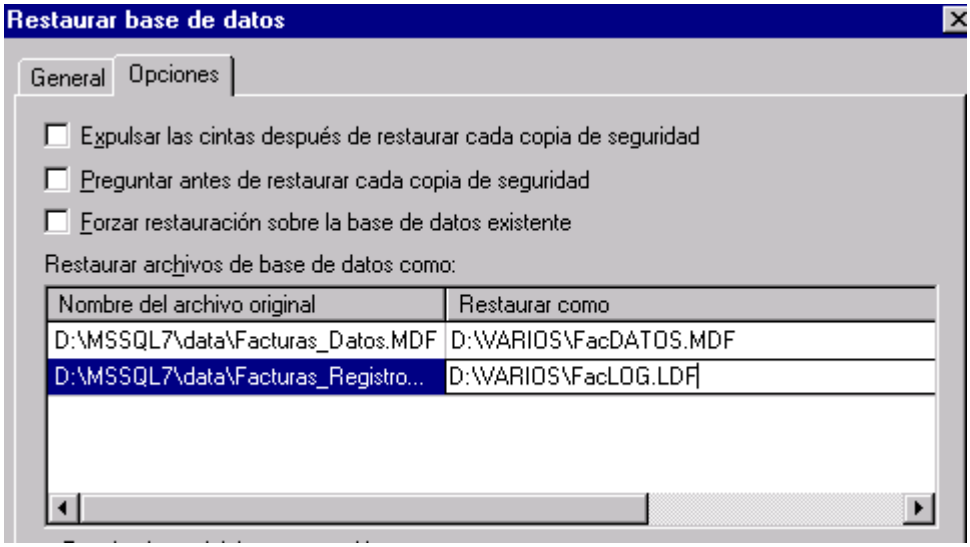


Figura 169. Restaurar una base de datos en una nueva ruta.

Instrucción RESTORE

La opción MOVE...TO de esta instrucción, restaura la base de datos en los archivos de una nueva ruta.

```
RESTORE NombreBD FROM DispositivoCopia [,…DispositivoCopiaN] [WITH
[MOVE ‘ArchivoLógico’ TO ‘ArchivoFísico’] [, MOVE …n ] ]
```

Es importante tener en cuenta que para realizar esta operación, debemos estar conectados al servidor como administradores de SQL Server y tener uso exclusivo sobre la base de datos.

El Código fuente 64 muestra como se restauran los archivos de la base de datos Facturas, contenidos en un dispositivo de copia permanente, en un nuevo directorio del disco.

```
RESTORE DATABASE Facturas FROM DispCopFacturas
WITH MOVE 'Facturas_Datos' TO 'D:\CORREO\FACFAT.MDF',
MOVE 'Facturas_Registro' TO 'D:\CORREO\FACREG.LDF'
```

Código fuente 64

17

Métodos para restaurar copias de seguridad

Restaurar una base de datos según su método de copia de seguridad

De igual manera que existen distintos métodos de copia de seguridad: completa, diferencial, registro de transacciones, etc., el modo de restaurar una copia de seguridad varía en función del método de copia empleado para crearla.

En este apartado, veremos como se realiza la restauración de una copia de seguridad creada mediante cada uno de los métodos de copia existentes.

Restaurar una copia de seguridad completa

Este es el tipo de restauración más simple, en el que se procede a recuperar la base de datos una vez restaurada. Para restaurar una copia de seguridad completa desde el Administrador corporativo, revise el lector el apartado *Restaurar una base de datos*, en este mismo tema, en el que se ilustra cómo restaurar una copia de seguridad completa de una base de datos.

En el caso de utilizar la instrucción RESTORE, deberemos especificar la opción RECOVERY si no vamos a restaurar más copias y queremos recuperar la base de datos; o bien NORECOVERY, si tenemos más copias de seguridad pendientes de restaurar sobre la misma base de datos.

En situaciones como la eliminación de la base de datos del disco, errores físicos del disco, o la necesidad de disponer de una copia de la base de datos en otro servidor, emplearemos este tipo de restauración.

Restaurar una copia de seguridad diferencial

Al realizar una copia de seguridad diferencial de una base de datos, se emplea menos tiempo debido a que sólo se copian los datos que han cambiado; de igual forma, al restaurar una copia de seguridad diferencial, tenemos la ventaja de emplear un menor tiempo, puesto que sólo restauramos la información que ha cambiado desde la última copia de seguridad completa

Para restaurar una copia de seguridad diferencial desde el Administrador corporativo, revise el lector en el apartado *Restaurar varias copias paso a paso y recuperación de una base de datos*, el punto *Restaurar la copia de seguridad diferencial*, en el que se explica este proceso.

En el caso de utilizar la instrucción RESTORE, las opciones RECOVERY y NORECOVERY se emplean igual que para las copias de seguridad completas.

Restaurar una copia de seguridad del registro de transacciones

Este tipo de copia de seguridad sirve para restaurar las modificaciones sufridas por la base de datos desde la última copia de seguridad completa o diferencial, asegurando la coherencia de la información.

Para restaurar una copia de seguridad del registro de transacciones desde el Administrador corporativo, revise el lector en el apartado *Restaurar varias copias paso a paso y recuperación de una base de datos*, el punto *Restaurar la copia de seguridad del registro de transacciones*, en el que se explica este proceso.

En el caso de utilizar la instrucción RESTORE, y disponer de varias copias de seguridad del registro de transacciones, deberemos utilizar la opción NORECOVERY al restaurar todas las copias excepto la última. La opción RECOVERY será empleada cuando tengamos una única copia del registro de transacciones o en la última de varias copias de este tipo.

Restaurar un registro de transacciones a un momento determinado

Al restaurar un registro de transacciones, es posible especificar un valor temporal hasta el cual se restaurará dicha copia de seguridad, quedando sin restaurar la información posterior.

Administrador corporativo

En la ventana de restauración de copias, marcaremos la casilla Restauración a un momento dado, Figura 170, que nos mostrará la ventana de especificación de la fecha y hora hasta la que vamos a recuperar la copia del registro de transacciones, ver Figura 171.

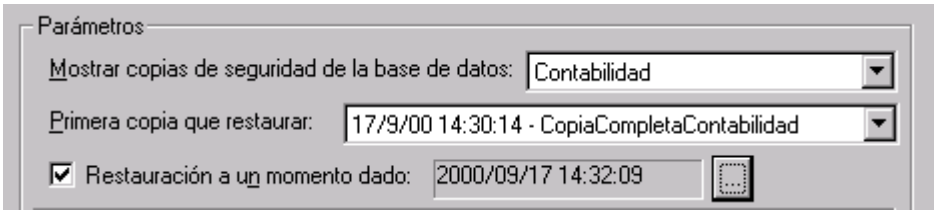


Figura 170. Restauración de una copia del registro de transacciones en un punto de tiempo.

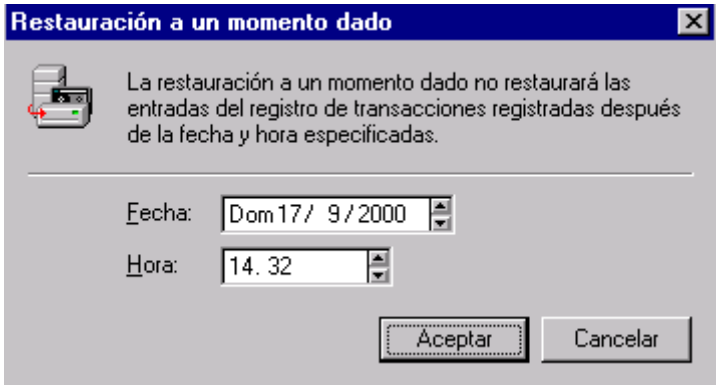


Figura 171. Valores de tiempo utilizados al restaurar una copia del registro de transacciones.

Instrucción RESTORE LOG

Esta instrucción dispone de la opción STOPAT, que nos servirá para indicar los valores de tiempo en los cuales la restauración de la copia de seguridad se detendrá.

```
RESTORE LOG NombreBD FROM DispositivoCopia [,…DispositivoCopiaN]
[WITH [STOPAT = FechaHora] ]
```

- STOPAT. Valor de fecha y hora en la que la restauración de la copia de seguridad de un registro de transacciones será detenida.

Siempre que utilicemos esta opción, deberemos recuperar la base de datos con RECOVERY.

El Código fuente 65 muestra un ejemplo en el que se restaura un registro de transacciones de la base de datos Contabilidad, hasta un punto determinado del tiempo.

```
RESTORE LOG Contabilidad FROM DispLogConta WITH RECOVERY STOPAT='Sep 15, 2000 18:00
PM'
```

Código fuente 65

Restaurar una copia de seguridad de ficheros o grupo de ficheros

Si disponemos de una base de datos de gran tamaño, que ha sido necesario situar en varios ficheros, y por requerimientos de tiempo, las copias de seguridad se realizan de tales ficheros, podemos restaurarlas a la base de datos con alguna variante respecto a la restauración habitual.

Administrador corporativo

En la ventana de restauración de copias, seleccionaremos en el apartado Restaurar, la opción *Grupos de archivos o archivos*, que nos mostrará en la lista de detalle, las copias realizadas de los archivos de la base de datos. En esta lista de detalle seleccionaremos los archivos a restaurar, marcando su casilla Restaurar, como muestra la Figura 172. Si necesitamos realizar una selección de archivos con mayor precisión, marcaremos la casilla *Seleccionar subconjunto de conjuntos de copia*, en la que podremos elegir los archivos por unidad, fecha o por los grupos en los que están organizados, Figura 173.

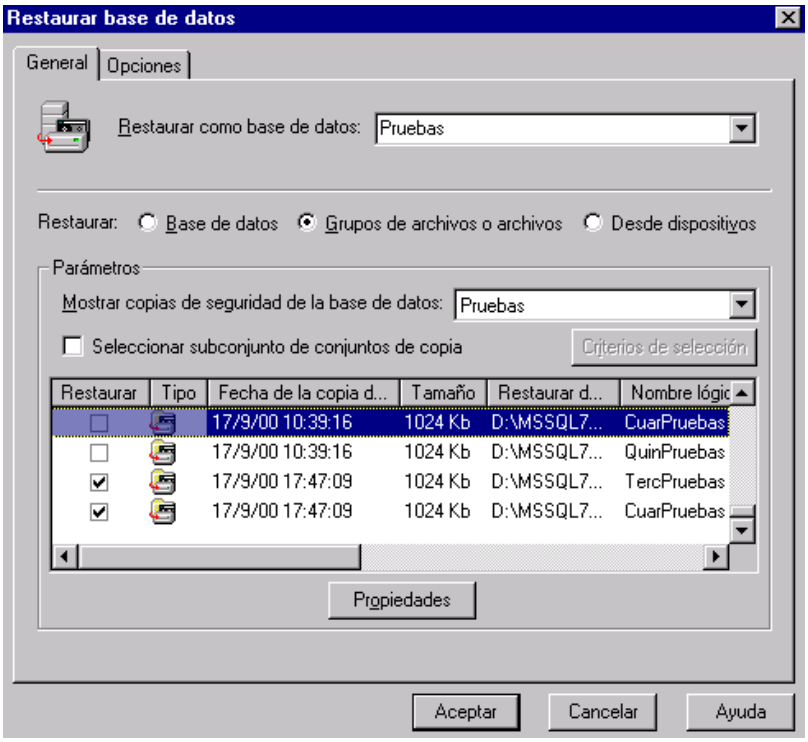


Figura 172. Restaurar archivos o grupos de archivos.

Figura 173. Selección de archivos para restaurar por subconjuntos.

Instrucción RESTORE

Las opciones FILE y FILEGROUP de esta instrucción, nos permiten restaurar los ficheros o grupos especificados

```
RESTORE DATABASE NombreBD FicheroGrupoFich [,...FicheroGrupoFich]
FROM DispositivoCopia [,...DispositivoCopiaN]
```

En esta sintaxis, FicheroGrupoFich se sustituye por:

```
FILE = NombreFichLógico | FILEGROUP = NombreGrupoFichLógico
```

El Código fuente 66 muestra un ejemplo de este tipo de restauración; en el que disponemos de una base de datos Contabilidad, compuesta por varios ficheros; las copias se han realizado sobre un dispositivo de copia permanente, y en este caso restauramos en primer lugar uno de los ficheros copiados sin recuperar la base de datos; a continuación restauramos la copia del registro de transacciones, recuperando la base de datos.

```
/************************/
RESTORE DATABASE Contabilidad
FILE=Conta2
FROM DispCopContabilidad
WITH NORECOVERY

/************************/
RESTORE LOG Contabilidad
FROM DispLogContabilidad
WITH RECOVERY
```

Código fuente 66

Empleo de un servidor SQL Server en espera

Un servidor de datos en espera puede utilizarse como respaldo al servidor de producción, en caso de fallo de este último, o para operaciones de lectura.

Creación de un servidor SQL Server en espera

En primer lugar realizaremos una copia de seguridad de las bases de datos y registros de transacciones del servidor de producción y las restauraremos en un nuevo servidor SQL Server, que utilizaremos como servidor en espera; podemos utilizar la opción MOVE...TO de RESTORE para indicar la nueva ubicación de la base de datos.

Tareas de mantenimiento

Para conseguir que el servidor en espera sea una copia lo más fiel posible del servidor de producción, es conveniente realizar con frecuencia copias de seguridad del registro de transacciones del servidor de producción y restaurarlas en el servidor en espera.

Uso del servidor en espera sólo para respaldo

En el caso de que el servidor en espera lo necesitemos para los casos en que exista un problema con el servidor de producción, al restaurar el registro de transacciones del servidor de producción, especificaremos la opción NORECOVERY, para que la base de datos no sea recuperada y su información permanezca inaccesible.

Recuperar un servidor en espera para operaciones de lectura

Si queremos que nuestros usuarios puedan consultar datos en el servidor en espera, se debe recuperar la base de datos después de restaurar el registro de transacciones del servidor de producción, pero existe el problema de que a partir de ese momento no se podrán restaurar más registros de transacciones del servidor de producción.

Para evitar este problema, debemos indicar a la base de datos que debe ser capaz de volver a restaurar más registros de transacciones desde el servidor de producción, empleando una de las siguientes técnicas.

Administrador corporativo

En la ventana de restauración, pestaña Opciones, apartado *Estado al concluir la recuperación*, pulsaremos la opción *Base de datos de sólo lectura. Capaz de restaurar nuevos registros de transacciones*, que se muestra en la Figura 174.

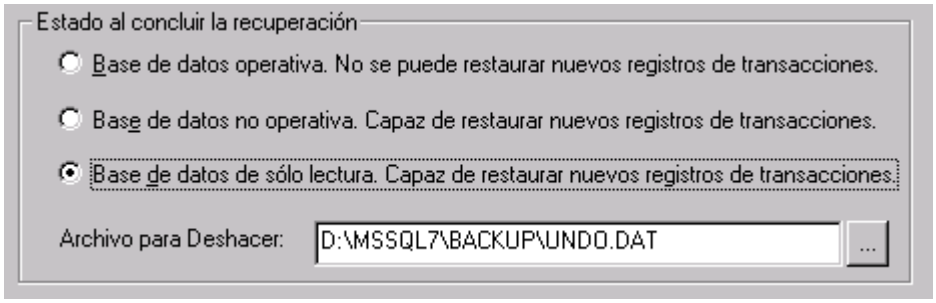


Figura 174. Opción para restaurar nuevos registros de transacciones en una base de datos de sólo lectura.

Debemos especificar en el campo *Archivo para Deshacer*, un nombre de archivo válido, que permitirá a SQL Server deshacer los cambios cuando se restauren nuevas copias del registro de transacciones. En el caso de que este archivo no exista, se creará automáticamente.

Instrucción RESTORE

La opción STANDBY de esta instrucción, permite restaurar registros de transacciones del servidor de producción sobre un servidor en espera, en el que los usuarios necesitan realizar operaciones de lectura.

```
RESTORE DATABASE NombreBD FROM DispositivoCopia
[,...DispositivoCopiaN] [WITH [STANDBY = 'ArchivoDeshacer']]
```

- ArchivoDeshacer. Nombre y ruta del archivo que SQL Server usará para deshacer los cambios de la base de datos.

El Código fuente 67 muestra una restauración de una base de datos y un registro de transacciones en un servidor en espera sobre el que se pueden realizar lecturas de datos.

```
/************************/
RESTORE DATABASE Contabilidad
FROM DispCompletaContab
WITH STANDBY='D:\DATOS\CONTABRECUP.LDF'

/************************/
RESTORE LOG Contabilidad
FROM DispLogContab
WITH STANDBY='D:\DATOS\CONTABRECUP.LDF'
```

Código fuente 67

Sustituir un servidor de producción por un servidor en espera

Si necesitamos realizar operaciones sobre un servidor SQL Server de producción, que requieran detener su actividad, podemos reemplazarlo por un servidor en espera, de forma que nuestros usuarios sigan teniendo acceso a los datos.

Conectar el servidor en espera como servidor de producción

En primer lugar, realizaremos una copia de seguridad del registro de transacciones del servidor de producción; especificaremos la opción NO_TRUNCATE, para obtener las transacciones confirmadas del registro.

A continuación, desconectaremos el servidor de producción y cambiaremos el nombre del servidor SQL Server en espera por el de producción.

Finalmente, restauraremos la última copia del registro de transacciones obtenida del servidor de producción sobre el servidor en espera, y recuperaremos el servidor con la opción RECOVERY.

Restaurar el servidor de producción original

Solucionados los problemas del servidor de producción que habíamos detenido, volveremos a ponerlo en funcionamiento; para ello, haremos en primer lugar una copia completa de la base de datos del servidor en espera.

A continuación, desconectaremos el servidor en espera y restauraremos la copia que acabamos de obtener en el servidor de producción.

Seguidamente conectaremos el servidor de producción, y crearemos una copia de seguridad completa de todas sus bases de datos, que por finalmente restauraremos en el servidor en espera, recuperando o no la base de datos, en función si es necesario que sea de lectura para los usuarios.

Restaurar bases de datos del sistema

Es muy importante mantener una copia regular de las bases de datos del sistema, ya que contienen información sensible, y en el caso de resultar perjudicadas, podrían impedir totalmente el acceso al servidor SQL Server.

Si ocurre un problema en alguna base de datos del sistema, pero podemos acceder al servidor, realizaremos la restauración de dicha base de datos mediante las técnicas explicadas a lo largo de este tema.

En el caso de daños que impidan el acceso al servidor SQL Server, deberemos ejecutar la aplicación REBUILDM.EXE, que se encuentra en la ruta UNIDAD:\...\MSSQL7\BINN, desde una sesión MS-DOS del sistema. Este programa regenera las bases de datos del sistema, permitiéndonos el acceso al servidor de datos; una vez conectados con el servidor, procederemos a restaurar nuestras copias de seguridad de las bases de datos del sistema.

18

Planes de copia de seguridad

Diseñar el plan de copias más adecuado

Como apuntábamos en el tema dedicado a la realización de copias de seguridad, no es posible aplicar una solución única para todas las empresas respecto a la estrategia de copias de seguridad que deben utilizar, debido a que cada compañía es diferente en cuanto al volumen de datos y manipulación de los mismos que maneja.

Sin embargo, si es posible trazar una serie de consejos, indicaciones, pautas, o como prefiera el lector denominarlas, que son de uso bastante generalizado y por otra parte, pueden darle una idea y servir como punto de partida para la construcción de su propia estrategia de copias de seguridad.

En los siguientes apartados, se describe un plan de copia de seguridad estándar para cada uno de los métodos de copia existentes; queda en manos del lector, usarlo tal y como se expone, o adaptarlo para sus necesidades particulares.

Plan de copia de seguridad completa

Situaciones

Cuando el tamaño de la base de datos es pequeño, sufre pocas actualizaciones o se usa fundamentalmente para consultas, es recomendable hacer una copia de seguridad completa.

Limpieza del registro de transacciones

Realizando únicamente copias de seguridad completa de la base de datos, corremos el riesgo de que el registro de transacciones se llene y SQL Server nos impida trabajar con la base de datos. Para evitarlo, debemos limpiarlo cada cierto tiempo o establecer en las opciones de la base de datos que el registro sea truncado al llegar a un punto de comprobación.

Para establecer esta opción desde el Administrador corporativo, abriremos la ventana de propiedades de la base de datos, pestaña Opciones, y marcaremos la opción *Truncar registro en punto de comprobación*, ver Figura 175.

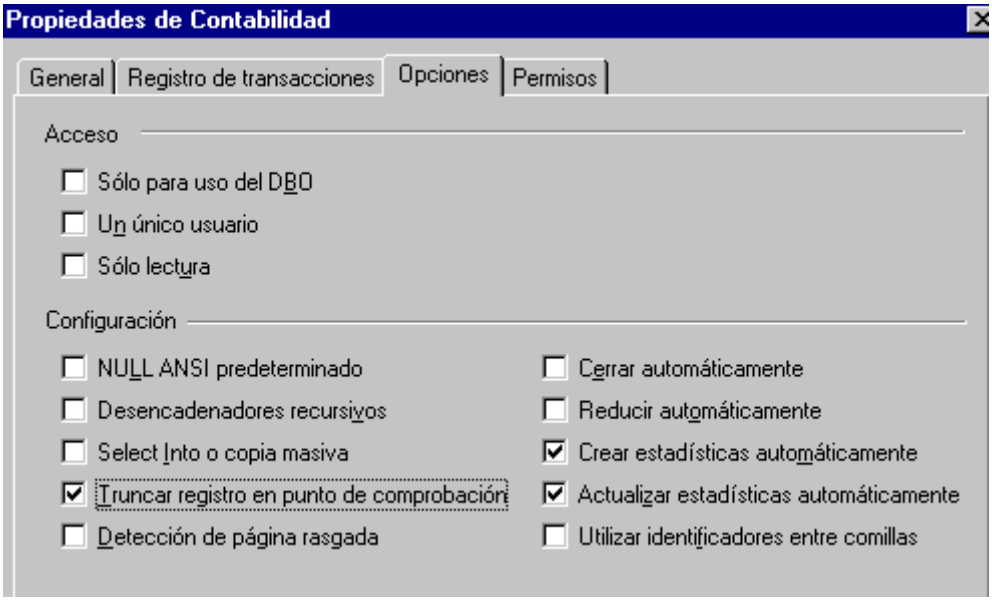


Figura 175. Establecimiento de la opción para trucar el registro de transacciones en la base de datos.

También podemos establecer esta opción utilizando el procedimiento almacenado del sistema sp_dboption, pasándole como parámetros la opción *trunc.log on chkpt* y el valor True.

Con esta acción, todas las transacciones confirmadas se escriben en la base de datos, y se eliminan del registro de transacciones, dejando espacio para nuevas transacciones sin confirmar.

Escenario de copia

- Disponemos de una base de datos de pequeño tamaño, aproximadamente 15 MB, por lo que el tiempo en realizar la copia de seguridad es corto.
- La cantidad de modificaciones en los datos no es alta, se usa fundamentalmente para consultar información.
- Como no queremos realizar mantenimientos sobre el registro de transacciones, estableceremos la opción en la base de datos para que sea truncado al llegar a un punto de comprobación.
- Se establece como horario de copia, todos los días a las 20:00 h.
- Ocurre un error en la base de datos a las 9:00 h., después de comenzar la jornada de trabajo.

Escenario de restauración

- Debemos restaurar la copia realizada el día anterior a las 20:00 h.
- Con esta estrategia, corremos el riesgo de perder datos, si se han producido modificaciones después de la última copia de seguridad.

La Figura 176 muestra un esquema de este plan de copia de seguridad.

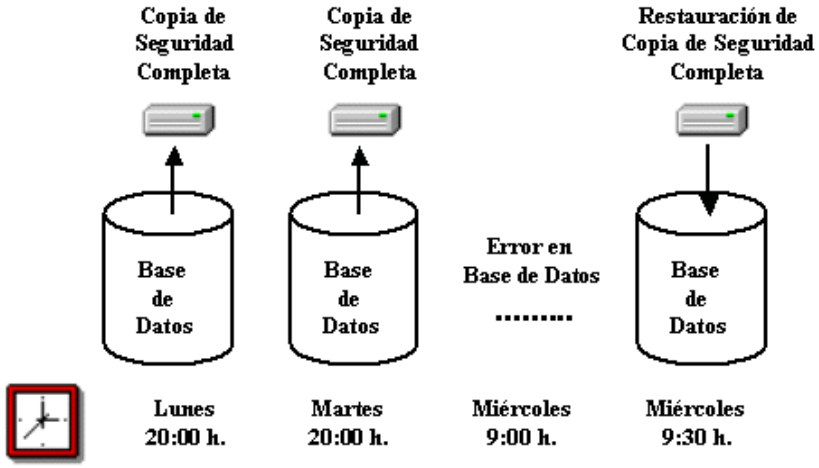


Figura 176. Plan de copia de seguridad completa de la base de datos.

Plan de copia de seguridad de base de datos y registro de transacciones

Situaciones

Si la base de datos tiene un considerable tamaño y además se realizan modificaciones con mucha frecuencia, es aconsejable diseñar un plan que incluya una copia completa de la base de datos con menor frecuencia, y copias del registro de transacciones más frecuentemente.

Escenario de copia

- Disponemos de una base de datos de mayor tamaño que la situación anterior, 170 MB aproximadamente, en la que el tiempo empleado en realizar la copia es algo más elevado.
- Se realizan numerosas actualizaciones diarias de información sobre la base de datos.
- Se crean dispositivos de copia separados para las copias de la base de datos y el registro de transacciones.
- Se realiza una copia de seguridad completa cada dos días, a las 8:00 h.

- Se realizan copias de seguridad del registro de transacciones, los días en que no se crea la copia de seguridad completa de la base de datos, a las 8:00 h.
- Ocurre un error en la base de datos a las 11:00 h., durante la jornada de trabajo.

Escenario de restauración

- Tras ocurrir el error, hacemos una copia de seguridad del registro de transacciones a las 11:05 h., sin truncarlo.
- A continuación, restauraremos la última copia de seguridad completa de la base de datos disponible.
- Restauraremos el registro de transacciones creados ese día a las 8.00 h.
- Finalmente, restauraremos la copia del registro de transacciones creada a las 11:05 h.

El lector puede observar el esquema del proceso de copia de esta estrategia en la Figura 177.

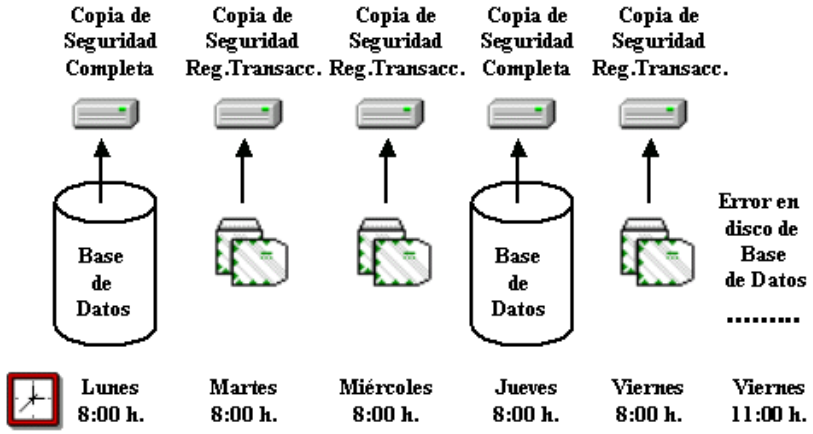


Figura 177. Proceso de copia de seguridad completa y del registro de transacciones.

La Figura 178 muestra el esquema del proceso de restauración copia de esta estrategia.

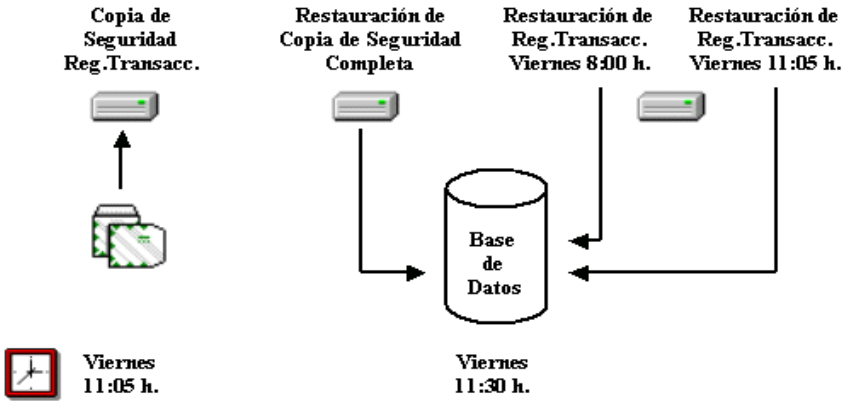


Figura 178. Proceso de restauración en plan de copia de seguridad completa y del registro de transacciones.

Plan de copia de seguridad diferencial

Situaciones

En bases de datos de gran tamaño, una copia de seguridad diferencial es de gran ayuda, ya que al contener sólo los cambios realizados a la base de datos, se emplea menos tiempo en la restauración.

Si disponemos de varias copias de seguridad diferencial, sólo es necesario restaurar la última. En cuanto a los registro de transacciones, es conveniente realizar copias frecuentes de los mismos, ya que una copia diferencial no captura su actividad.

Escenario de copia

- Realizamos una copia de seguridad completa de nuestra base de datos, una vez cada semana, todos los lunes, a las 19:00 h.
- Durante la jornada, se realizan copias de seguridad del registro de transacciones cada hora.
- Todos los días, al finalizar la jornada de trabajo, se realiza una copia de seguridad diferencial.
- Sucede un error sobre el servidor SQL Server el jueves, a las 17:30 h.

Escenario de restauración

El esquema de la Figura 179 muestra el proceso de copias de seguridad.

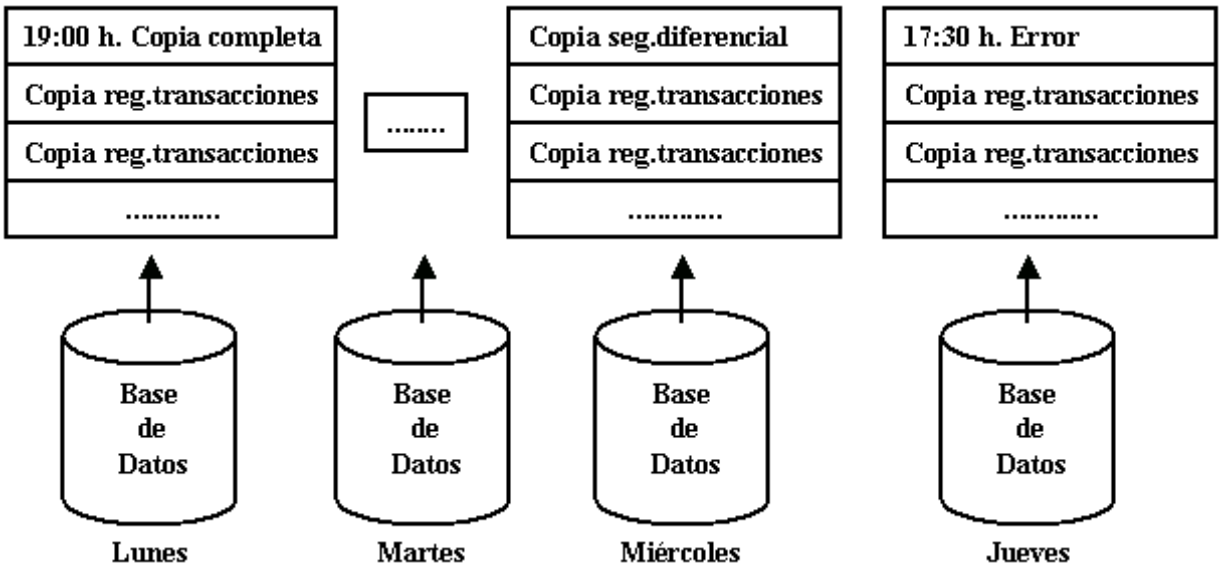


Figura 179. Proceso de copia de seguridad diferencial.

- Después del error, hacemos una copia de seguridad del registro de transacciones a las 17:45 h., sin truncarlo.

- Restauramos la copia de seguridad completa realizada el lunes.
- Restauramos la copia de seguridad diferencial creada el miércoles.
- Restauramos la copia de seguridad del registro de transacciones realizada el presente día a las 17:00 h.
- Por último, restauramos la copia de seguridad del registro de transacciones, creada al comienzo de este proceso de restauración.

El esquema de la Figura 180 muestra el proceso de restauración.

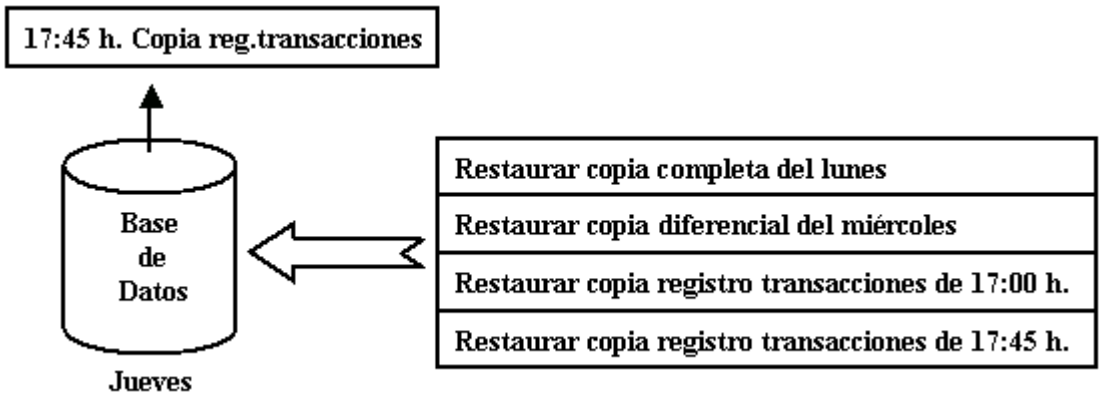


Figura 180. Proceso de restauración de copia de seguridad diferencial.

Plan de copia de seguridad de archivos o grupos

Situaciones

En grandes bases de datos, distribuidas en varios ficheros, y en las que realizar la copia de seguridad sea tarea que nos puedan ocupar varias horas, podemos diseñar una estrategia de copia de seguridad basada en la copia de los ficheros o grupos que componen la base de datos.

Escenario de copia

- Debemos hacer copias de una base de datos distribuida en los siguientes ficheros: Datos1, Datos2 y Datos3.
- Realizamos una copia de seguridad completa de nuestra base de datos, una vez cada semana, todos los lunes, a las 8:00 h.
- Cada día se realiza la copia de uno de los ficheros a las 20:00 h.; el lunes se copia Datos1, el martes Datos2, el miércoles Datos3, y así sucesivamente.
- Se realizan copias del registro de transacciones todos los días, a las 9:00 h. y 15:00 h.
- Ocurre un error en el fichero Datos2, el miércoles, a las 21:00 h.

Escenario de restauración

- Después del error, hacemos una copia de seguridad del registro de transacciones a las 21:15 h., sin truncarlo.
- Restauramos la copia de seguridad del fichero Datos2, creada el martes.
- Restauramos las copias del registro de transacciones, realizadas desde el miércoles a partir de las 20:00 h.
- Finalmente, restauramos la copia de seguridad del registro de transacciones, creada al comienzo de este proceso de restauración.

El esquema de la Figura 181 muestra el proceso de copias de seguridad.

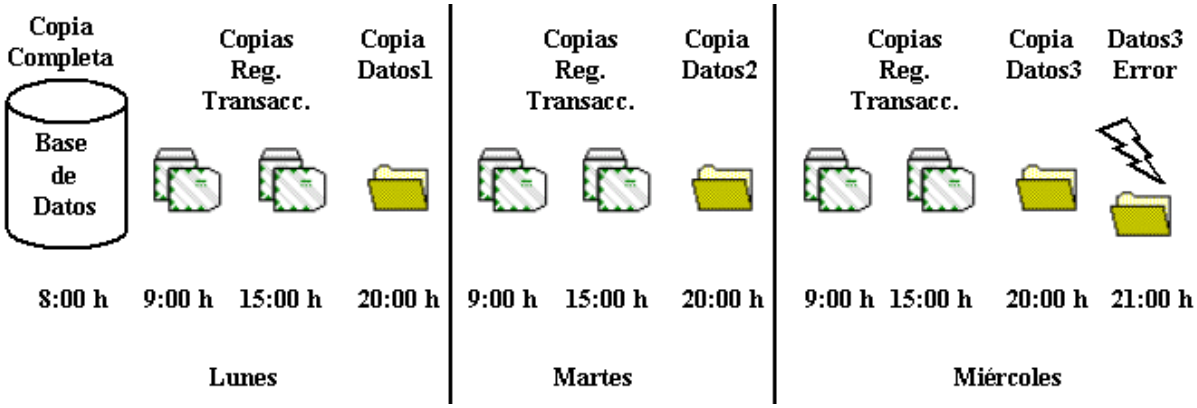


Figura 181. Plan de copia de seguridad de ficheros o grupos.

El esquema de la Figura 182 muestra el proceso de restauración.

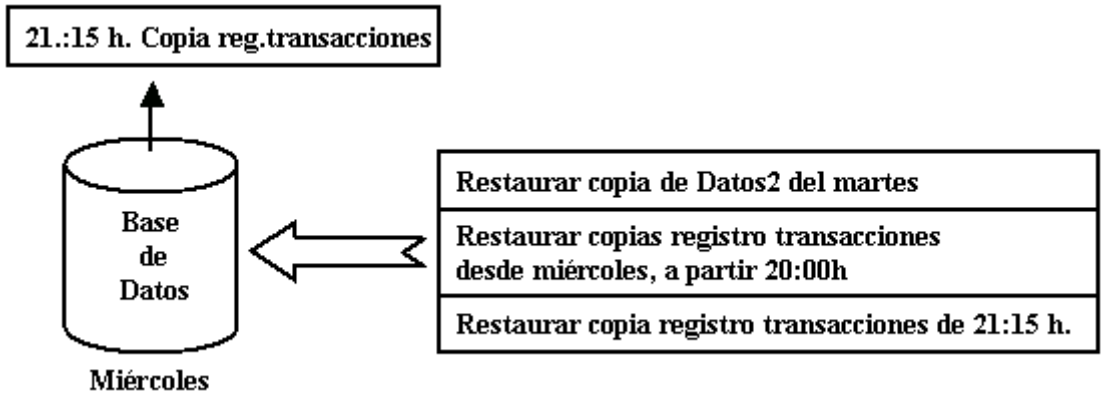


Figura 182

19

Programación de tareas

Automatización de tareas rutinarias

SQL Server permite programar el servidor de modo que pueda realizar ciertas operaciones prefijadas por el administrador en una fecha y hora determinadas o a intervalos regulares de tiempo.

De igual forma, es posible proporcionar soluciones a errores generados por el servidor de datos o problemas de rendimiento.

Agente SQL Server

El Agente SQL Server es un servicio del gestor de datos, que revisa las operaciones del servidor que son escritas en el registro de Windows NT. En el caso de que para una de estas operaciones haya definido un trabajo o alerta, el Agente SQL lo ejecuta.

Entre las situaciones en que SQL Server registra operaciones de Windows NT se encuentran los errores de SQL Server con nivel de 19 a 25, la ejecución del comando RAISERROR WITH LOG y los mensajes de error definidos expresamente con los procedimientos almacenados del sistema sp_addmessage o sp_altermessage.

Organizar el entorno de programación de tareas

Como paso previo a la creación de elementos de automatización del servidor SQL Server, debemos realizar las siguientes verificaciones.

- Comprobar que se está ejecutando el servicio Agente SQL Server desde el Administrador de servicios.
- Verificar que el inicio de sesión de Agente SQL Server está asignado a la función sysadmin de SQL Server.
- Verificar que el Agente SQL Server conecta con el servidor de datos de la máquina local.

Preparación del sistema de correo de SQL Server

Para poder enviar mensajes, el Agente SQL utiliza los componentes SQLAgentMail y SQL Main, que permiten el uso de un sistema de correo electrónico, utilizando un servidor de correo que cumpla con la especificación MAPI (Messaging Application Programming Interface). SQLAgentMail permite enviar un correo electrónico cuando una tarea programada se ejecuta y cuando se activa una alerta. SQL Mail por su parte, es empleado por SQL Server para procesar los mensajes de la bandeja de entrada, y enviar conjuntos de resultados como respuesta al mensaje recibido.

También es posible el envío de mensajes utilizando el procedimiento almacenado del sistema xp_sendmail, ejecutado desde una aplicación desarrollada con una herramienta de programación que permita el acceso a SQL Server como puede ser Visual Basic. Para poder ejecutar SQL Mail, es preciso que el servicio SQL Server use una cuenta de usuario de dominio. Para configurar el correo en el Agente SQL, una vez que hemos conectado con un servidor SQL Server, abriremos la carpeta Administración y haremos clic con el botón derecho en el elemento *Agente de SQL Server*, seleccionando del menú contextual la opción *Propiedades*, que nos mostrará la ventana de la Figura 183.

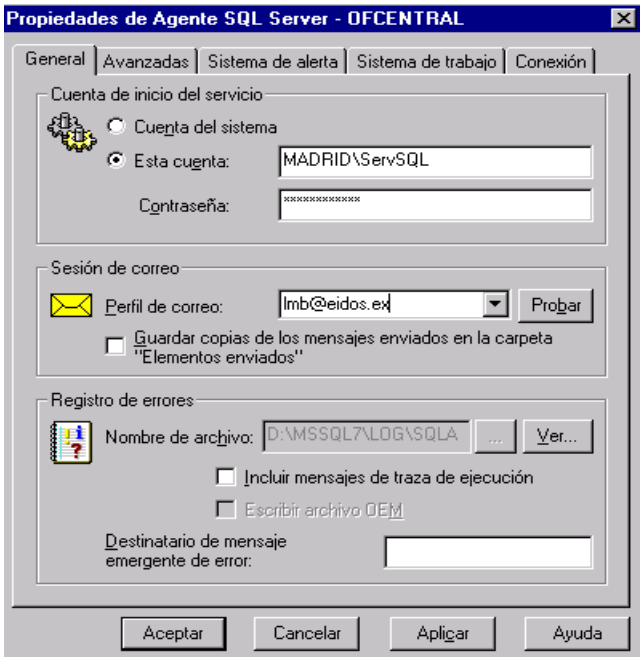


Figura 183. Propiedades del Agente SQL Server.

En el apartado Sesión de correo, estableceremos los valores para la comunicación mediante este sistema, y aceptaremos la ventana.

Trabajos

Un trabajo consiste en un proceso ejecutado en uno o varios pasos, se puede ejecutar directamente por parte de un usuario de SQL Server que tenga los permisos adecuados, o bien ser programado para su ejecución en un momento o intervalo fijado al definir el trabajo.

Adicionalmente podemos especificar un usuario u operador del sistema al que se notificará de las diversas situaciones que pueden acontecer al ejecutar este trabajo.

Administrador corporativo

Desde esta utilidad y conectados a un servidor de datos, abriremos la carpeta Administración y haremos clic con el botón derecho en el elemento Trabajos, seleccionando la opción del menú contextual *Nuevo trabajo*, que abrirá la ventana de creación del trabajo.

Esta ventana de propiedades del trabajo dispone de varias pestañas que describiremos seguidamente, al mismo tiempo que vamos creando un trabajo de ejemplo, ilustrativo del proceso.

El escenario de creación del trabajo es el siguiente: una empresa de ventas necesita actualizar el sector de trabajo de todos su comerciales llegada una determinada fecha y hora, de forma que los comerciales que trabajen en Cuenca, en el sector Norte, pasen a realizar sus ventas en el sector Oeste de la ciudad.

Al mismo tiempo, se marcarán pendientes de visita, los clientes de dicho sector.

Disponen para llevar el control de esta información en su servidor SQL Server, de la base de datos Logística, y la tabla Vendedores que contiene los datos de los comerciales; por otro lado tenemos la tabla Clientes en la que reflejaremos los Clientes pendientes de visita de la mencionada ciudad.

Comenzamos creando un nuevo trabajo, y en la primera pestaña, General, destacamos las siguientes propiedades:

- Nombre. Nombre identificativo del trabajo.
- Habilitado. Si marcamos este CheckBox, el trabajo se ejecutará según el modo en que se haya programado. Por defecto, los trabajos se encuentran habilitados.
- Categoría. Permite incluir el trabajo en un grupo para poder organizarlo según las funcionalidades que desempeñe.
- Propietario. Cuenta de inicio que se define como creadora del trabajo.

La Figura 184 muestra las propiedades generales del trabajo xxx que hemos comenzado a definir.

A continuación, hacemos clic en la pestaña Pasos, en la que definiremos las acciones que serán ejecutadas. La Figura 185 muestra esta ventana antes de definir los pasos.

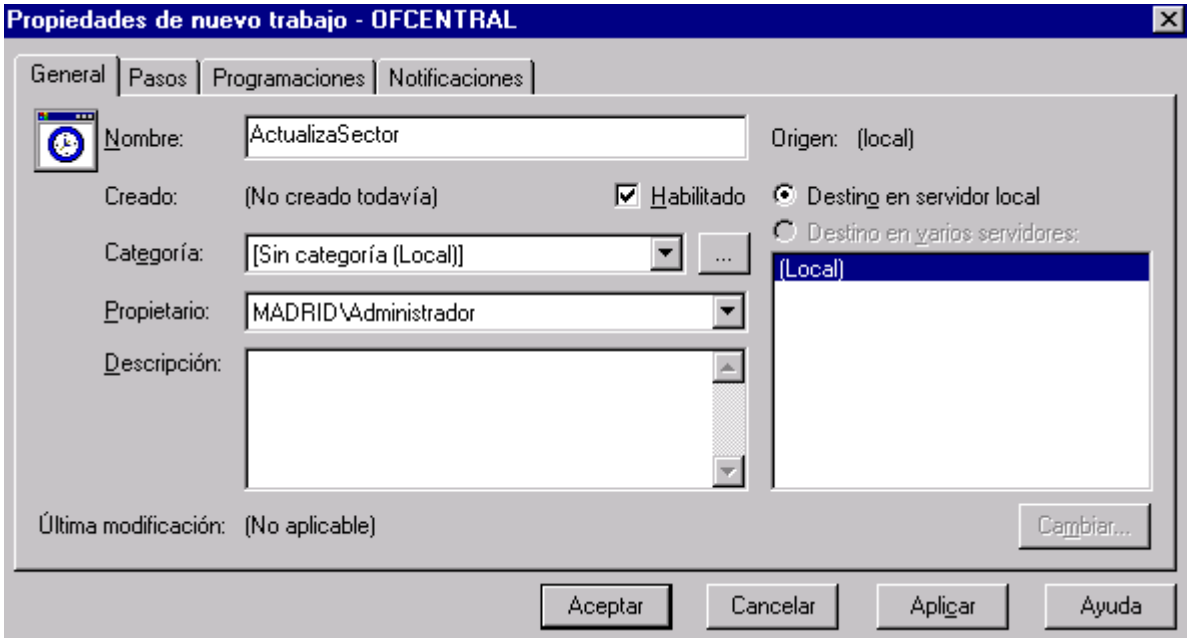


Figura 184. Propiedades generales de un trabajo.

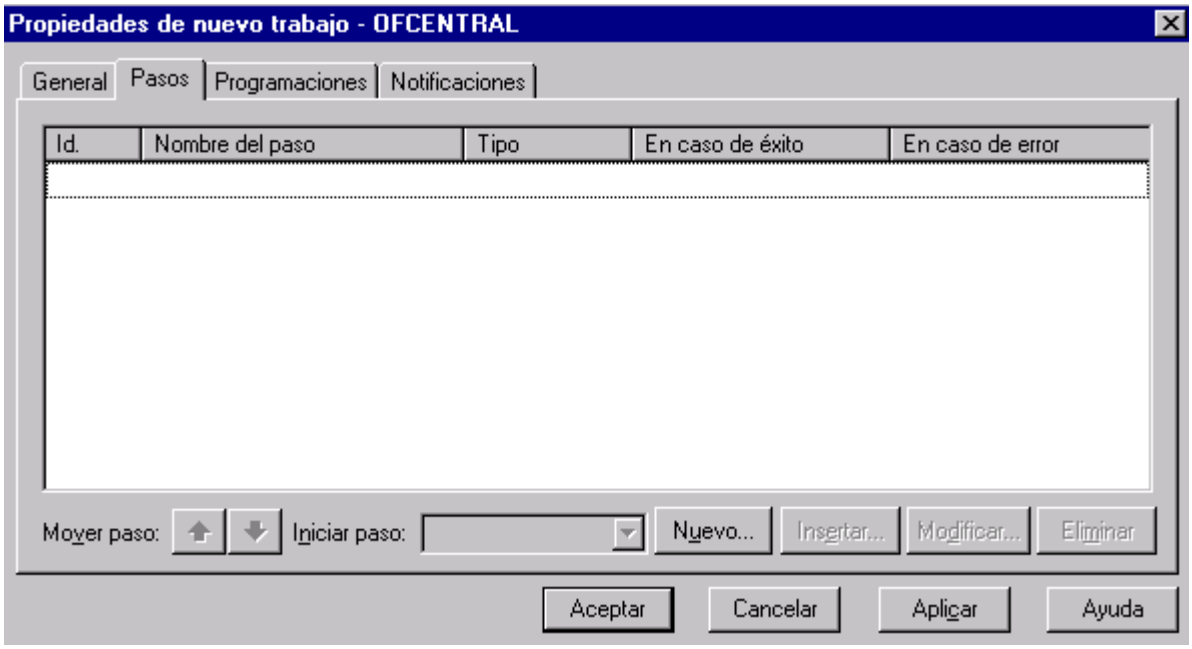


Figura 185. Definición de pasos del trabajo.

Pulsaremos el botón Nuevo para añadir un paso de trabajo. En la ventana de propiedades del paso, véase Figura 186, introduciremos las siguientes propiedades:

- Nombre del paso. Nombre identificativo para el paso.
- Tipo. La clase de acción que será ejecutada, tal como una instrucción SQL, comando del sistema operativo, etc.
- Base de datos. Nombre de la base de datos sobre la que se ejecutará el paso.

- Comando. Sentencia SQL que se ejecutará al invocar el paso.

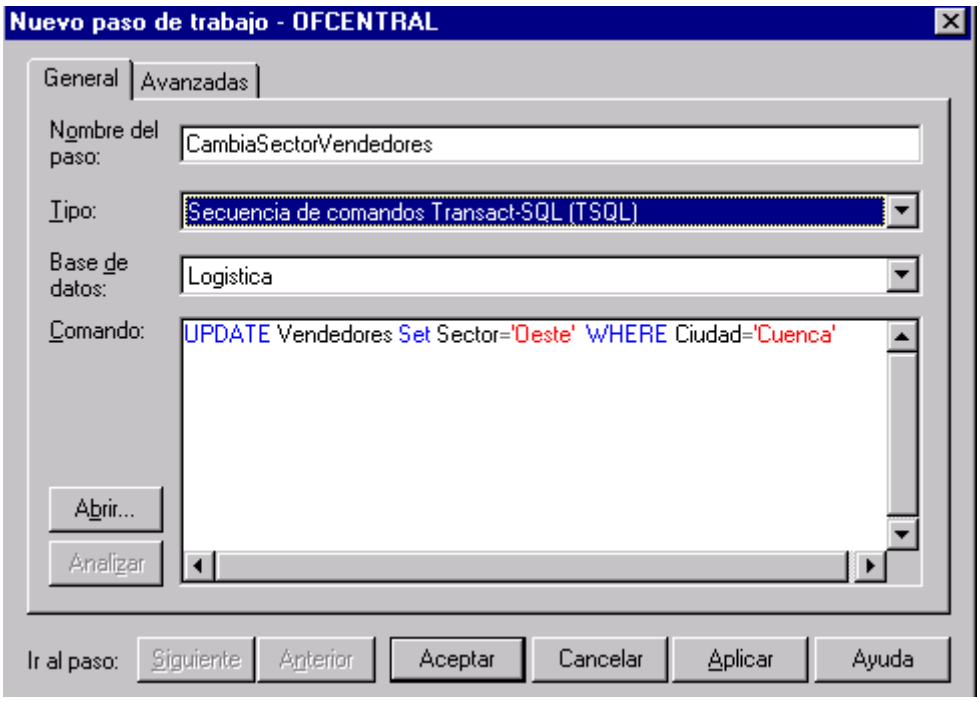


Figura 186. Propiedades de un paso para un trabajo.

Terminado de crear el paso, pulsaremos Aceptar, con lo que se añadirá a la lista de pasos de la ventana de creación del trabajo.

Realizaremos la misma acción para el siguiente paso que necesitamos crear, y que podemos ver en la Figura 187.

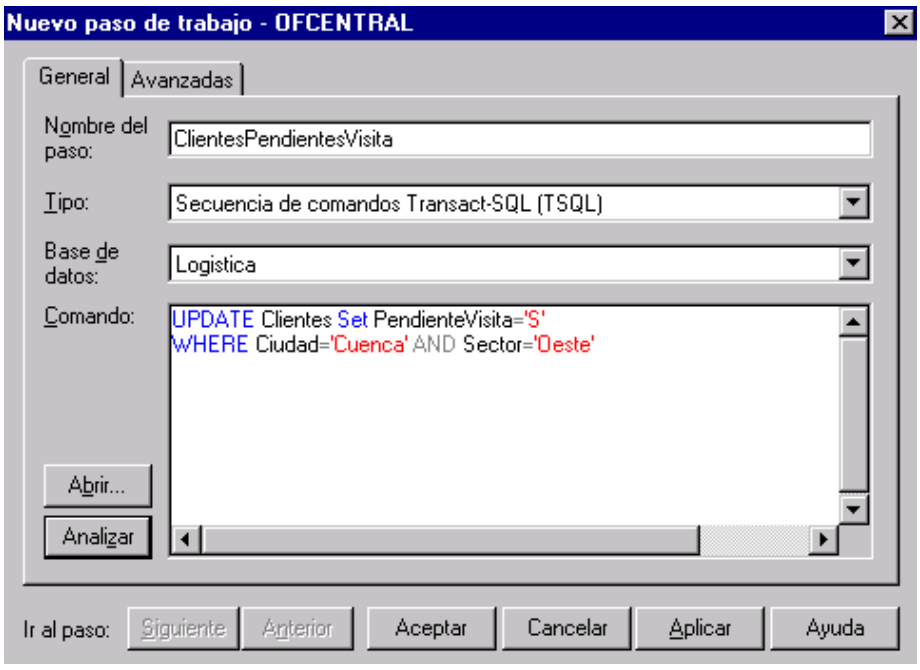


Figura 187. Segundo paso del ejemplo.

Finalizada la creación de todos los pasos, se mostrarán en la ventana de creación del trabajo, en la que podemos cambiar el orden de ejecución de los pasos, modificar, eliminar, etc., véase la Figura 188; el paso que tiene junto a su nombre un icono con una bandera será el primero que se ejecute.

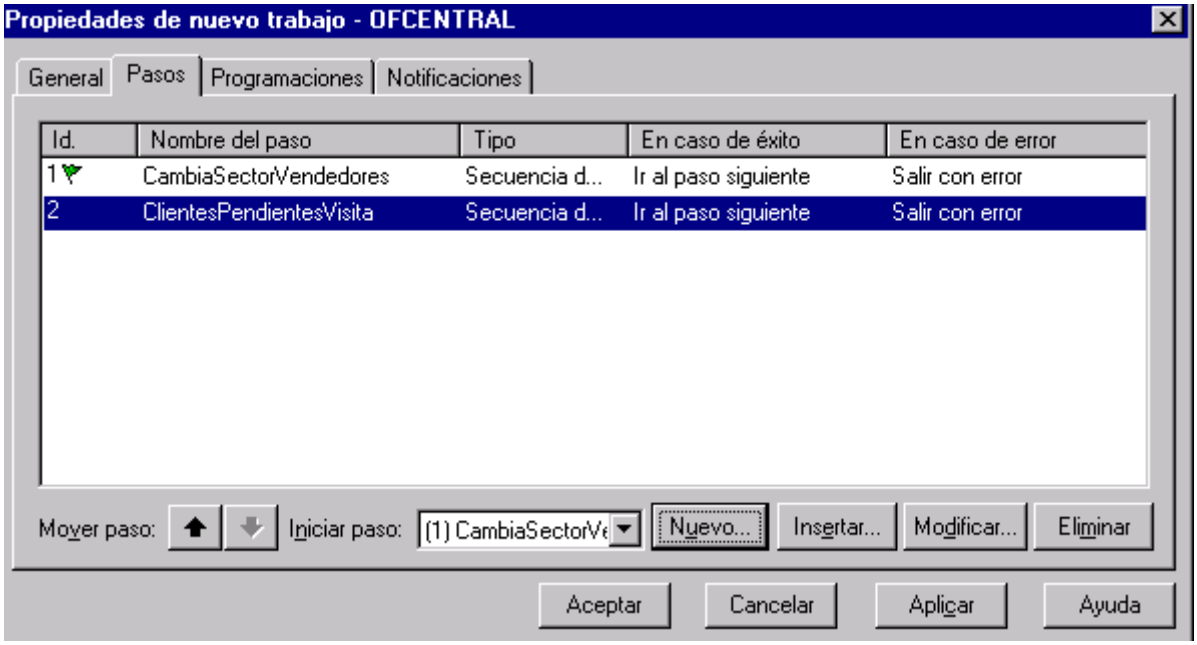


Figura 188. Ventana de creación de trabajo, con la lista de pasos.

En la pestaña Programaciones, configuraremos el trabajo para que pueda ser ejecutado en base a unos parámetros de tiempo.

Después de hacer clic en esta pestaña, pulsaremos el botón *Nueva programación*, que nos mostrará la ventana de la Figura 189, que dispone de las siguientes propiedades:

- Nombre. Nombre identificativo de la programación.
- Habilitado. Si marcamos esta opción, la programación se ejecutará según los valores configurados. Cuando esté deshabilitada una programación, no se ejecutará aunque se cumpla el plazo de su programación.
- Tipo de programación. Los diferentes modos en que podemos establecer un momento en el que se ejecutará el trabajo.

En este caso, hemos configurado la programación para que se ejecute una vez en una fecha y hora determinadas, quedando la ventana del trabajo en este punto como se muestra en la Figura 190.

Como paso final, debemos indicar a quién queremos que se comunique el resultado de la ejecución del trabajo, aspecto que establecemos en la pestaña Notificaciones.

Es posible enviar un correo electrónico a un operador del sistema, a un localizador (comúnmente denominados *busca*), o un mensaje de red. También podemos ajustar la situación en la que se generará la notificación: si el trabajo tiene o no éxito, o al finalizar el trabajo. La Figura 191 muestra las propiedades disponibles para enviar notificaciones.

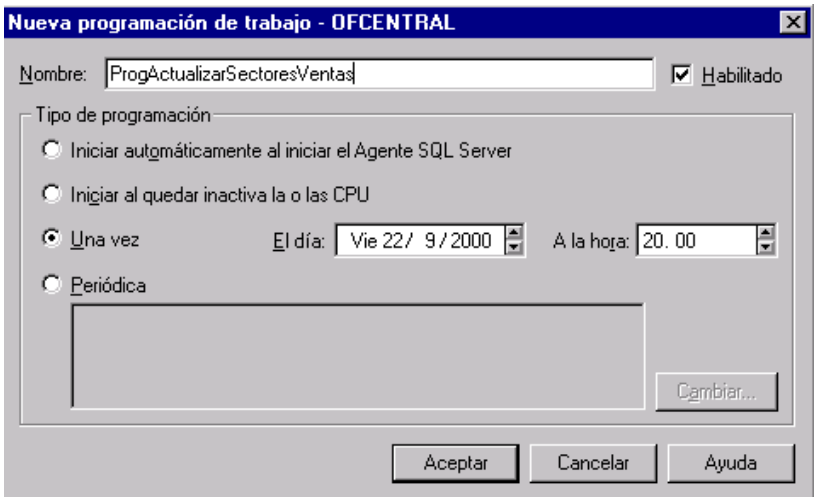


Figura 189. Asignación de una programación a un trabajo.

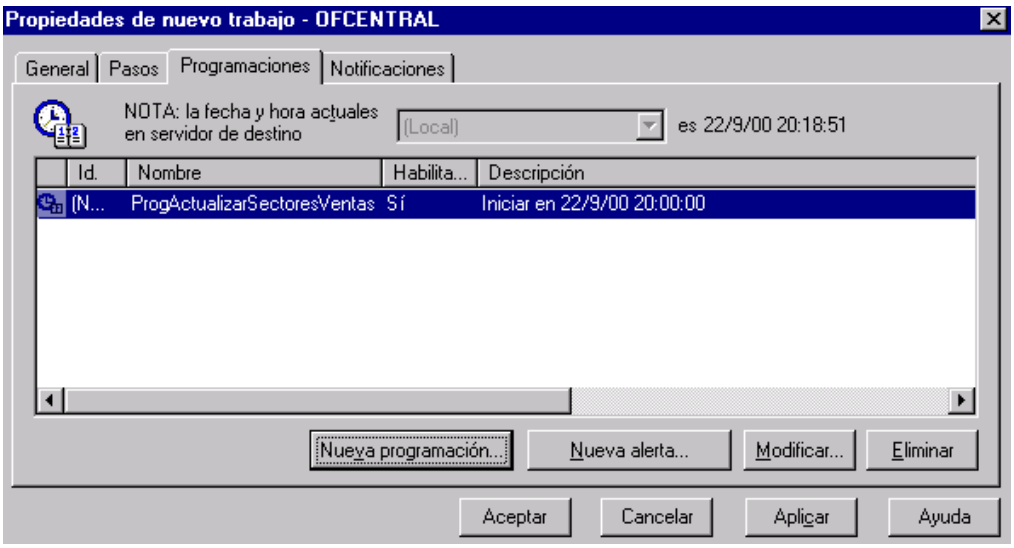


Figura 190. Ventana de creación de trabajo. Lista de programaciones.

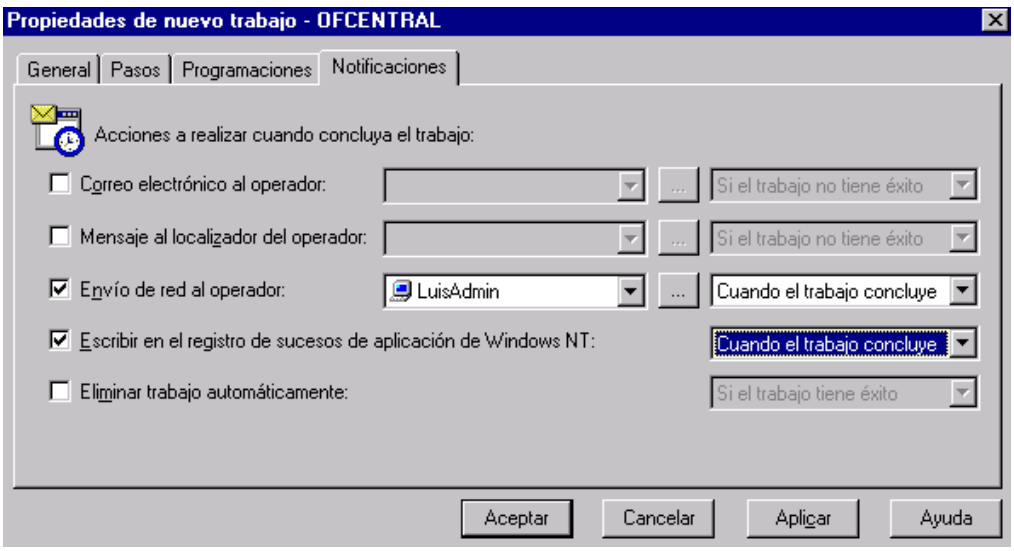


Figura 191. Definición de notificaciones para un trabajo.

Como puede observar el lector, en este caso, se notificará al operador LuisAdmin con un mensaje de red, a la finalización del trabajo.

Completada esta parte, pulsaremos Aceptar, grabándose el trabajo en el servidor.

Una vez creado un trabajo, podemos esperar su ejecución al cumplirse el plazo establecido en su programación, o bien ejecutarlo directamente, haciendo clic sobre el mismo con el botón derecho del ratón, y seleccionando la opción *Iniciar trabajo*, del menú contextual. En el trabajo de ejemplo que hemos definido, una vez concluido el trabajo, el sistema enviará al operador asignado para notificaciones el mensaje de la Figura 192.

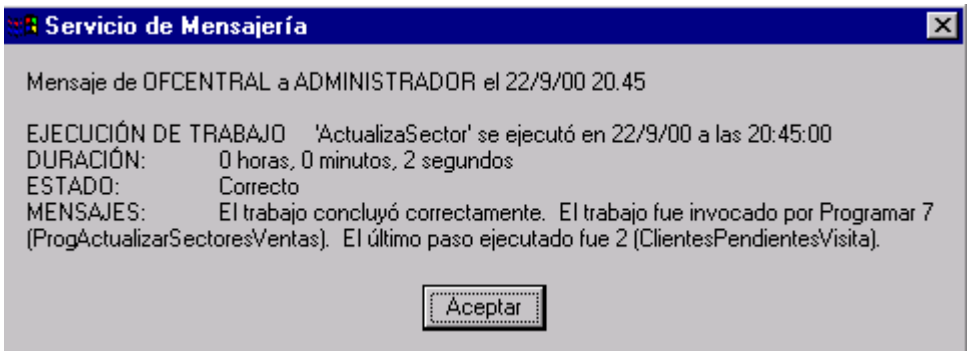


Figura 192. Notificación del sistema a la conclusión de un trabajo de SQL Server.

Procedimientos almacenados del sistema

Los procedimientos almacenados del sistema enumerados a continuación, nos permiten crear y configurar desde código trabajos de SQL Server. El lector debe tener presente que estos procedimientos deben ejecutarse posicionados en la base de datos del sistema msdb. Tenga en cuenta también que la sintaxis se muestra de forma parcial.

sp_add_job

Añade un nuevo trabajo al servidor SQL Server actual.

```
sp_add_job 'NombreTrabajo'
```

- NombreTrabajo. Cadena con el nombre que vamos a asignar al trabajo.

El Código fuente 68 crea un nuevo trabajo para el servidor, con todas sus configuraciones predeterminadas.

```
USE msdb
EXEC sp_add_job "CopiasSegExtra"
```

Código fuente 68

sp_add_jobstep

Agrega un nuevo paso para un trabajo ya creado.

```
sp_add_jobstep [@job_id =] IDTrabajo | [@job_name =]
'NombreTrabajo'] [,[@step_id =] IDPaso] ,[@step_name =] 'NombrePaso'
[,[@subsystem =] 'Subsistema'] [,[@command =] 'Comando']
```

- IDTrabajo. Número de identificación del trabajo.
- NombreTrabajo. Nombre del trabajo al que vamos a añadir un paso.
- IDPaso. Número que especifica el orden en que serán ejecutados los pasos del trabajo.
- NombrePaso. Cadena con el nombre que vamos a asignar al paso.
- Subsistema. Cadena con la descripción del subsistema que utilizará el Agente SQL Server para ejecutar el comando. Los posibles valores se muestra en la Tabla 10.

Valor	Descripción
'ACTIVESCRIPTING'	Secuencia de comandos activa.
'CMDEXEC'	Comando del sistema operativo o programa ejecutable.
'DISTRIBUTION'	Trabajo agente de distribución de duplicación.
'SNAPSHOT'	Trabajo agente de instantáneas de duplicación.
'LOGREADER'	Trabajo agente de lector del registro de duplicación.
'MERGE'	Trabajo agente de mezcla de duplicación.
'TSQL' (predeterminado)	Instrucción de Transact-SQL.

Tabla 10. Valores disponibles para el subsistema de un paso.

- Comando. Comando del paso que ejecutará el Agente SQL Server mediante el subsistema.

El Código fuente 69 crea un paso para el trabajo definido anteriormente, en el que se define una acción de copia de seguridad sobre una base de datos. Observe el lector cómo en este caso, incluimos los nombre de las variables parámetro al ejecutar el procedimiento almacenado mediante la sintaxis: @NombreVarParam='Valor'.

```
EXEC sp_add_jobstep @job_name='CopiasSegExtra', @step_name='CopiaBDLogistica',
@subsystem='TSQL', @command="BACKUP DATABASE Logistica TO
DISK='D:\MSSQL7\BACKUP\LOGISTIC1.BAK'"
```

Código fuente 69

sp_add_jobschedule

Define y añade una programación para un trabajo.

```
sp_add_jobschedule [@job_id =] IDTrabajo, | [@job_name =]
'NombreTrabajo',[@name =] 'NombreProg' [,[@enabled =] Habilitado]
[,[@freq_type =] TipoFrecuencia] [,[@freq_interval =]
IntervaloFrecuencia]
```

- IDTrabajo. Número de identificación del trabajo.
- NombreTrabajo. Nombre del trabajo al que vamos a añadir una programación.
- NombreProg. Nombre de la programación que vamos a crear.
- Habilitado. Estado de la programación. Si el valor es 1, la programación está habilitada y se ejecutará en el momento establecido; si el valor es 0, la programación estará deshabilitada.
- TipoFrecuencia. Permite establecer cuando se ejecutará el trabajo, según la Tabla 11.

Valor	Descripción
1	Una vez.
4	Diariamente.
8	Semanalmente.
16	Mensualmente.
32	Mensualmente, ver IntervaloFrecuencia.
64	Se ejecuta cuando se inicia el Agente SQL Server.
128	Se ejecuta cuando el equipo está inactivo.

Tabla 11. Valores para el parámetro TipoFrecuencia.

- IntervaloFrecuencia. Indica los días en que se ejecutará el trabajo, según los valores de la Tabla 12.

Valor de TipoFrecuencia	Efecto sobre IntervaloFrecuencia
1 (una vez)	IntervaloFrecuencia no se utiliza.
4 (diario)	Cada IntervaloFrecuencia días.

8 (semanal)	IntervaloFrecuencia es uno o varios (con OR) de los siguientes: 1=Domingo 2=Lunes 4=Martes 8=Miércoles 16=Jueves 32=Viernes 64 = Sábado
16 (mensual)	En el día IntervaloFrecuencia del mes.
32 (mensual relativo)	IntervaloFrecuencia es uno de los siguientes: 1=Domingo 2=Lunes 3=Martes 4=Miércoles 5=Jueves 6=Viernes 7=Sábado 8=Día 9=Día laborable 10 = Día de fin de semana
64 (Cuando se inicia el Agente SQL Server)	IntervaloFrecuencia no se utiliza.
128	IntervaloFrecuencia no se utiliza.

Tabla 12. Valores para el parámetro IntervaloFrecuencia.

El Código fuente 70 añade una programación al trabajo que hemos creado en los pasos anteriores.

```
EXEC sp_add_jobschedule @job_name='CopiasSegExtra', @name='ProgCopiaLogistica',
@freq_type=1
```

Código fuente 70

Trabajos multiservidor

En una red de equipos en la que existan varios servidores, es posible definir trabajos para que sean ejecutados en dichos servidores. Para ello, es necesario definir uno de los servidores como principal, y el resto como servidores de destino.

El servidor principal se encargará de la administración de trabajos y su distribución entre los servidores de destino, estos últimos, por su parte, enviarán los resultados al servidor principal.

Para definir los servidores desde el Administrador corporativo, en la carpeta Administración, haremos clic con el botón derecho en el elemento Agente SQL Server, y seleccionaremos del menú contextual la opción *Administración de multiservidor*, que a su vez, nos llevará a sendas opciones para definir el servidor principal o de destino mediante un asistente de SQL Server.

Por otra parte, disponemos del procedimiento almacenado sp_msx_enlist, que podremos ejecutar para hacer esta tarea sin el uso de asistentes.

```
sp_msx_enlist 'ServidorPrincipal' [,'Ubicación']
```

- ServidorPrincipal. Nombre del servidor principal que coordinará los trabajos de SQL Server.
- Ubicación. Servidor que se incorpora como destino al servidor principal.

Operadores

Un operador es aquel usuario que recibe las notificaciones de la ejecución de trabajos y alertas de SQL Server. A continuación se describe el procedimiento necesario para crear un operador para el servidor de datos.

Administrador corporativo

Para crear un operador de SQL Server, abriremos la carpeta Administración del servidor y el elemento Agente SQL Server; a continuación haremos clic con el botón derecho sobre Operadores y elegiremos la opción del menú contextual *Nuevo operador*, que nos mostrará la ventana de creación del operador de la Figura 193.

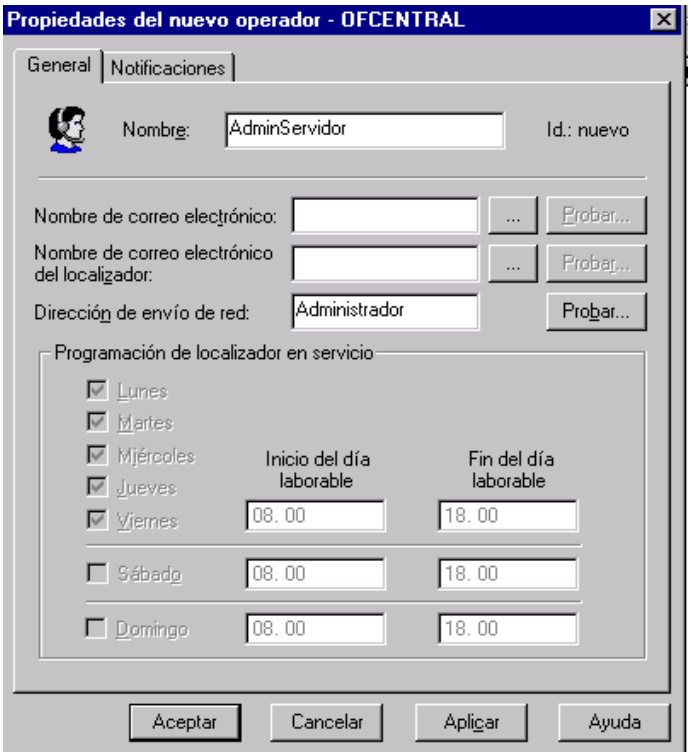


Figura 193. Ventana de creación de un operador de SQL Server.

Entre las propiedades que podemos especificar, se encuentran el nombre del operador, dirección de correo electrónico, localizador y red, a la que enviar los mensajes generados en los trabajos y alertas, etc.

Si hacemos clic en la pestaña Notificaciones, se muestran los trabajos y alertas definidos para los que podemos especificar si queremos que el operador reciba notificaciones, ver Figura 194.

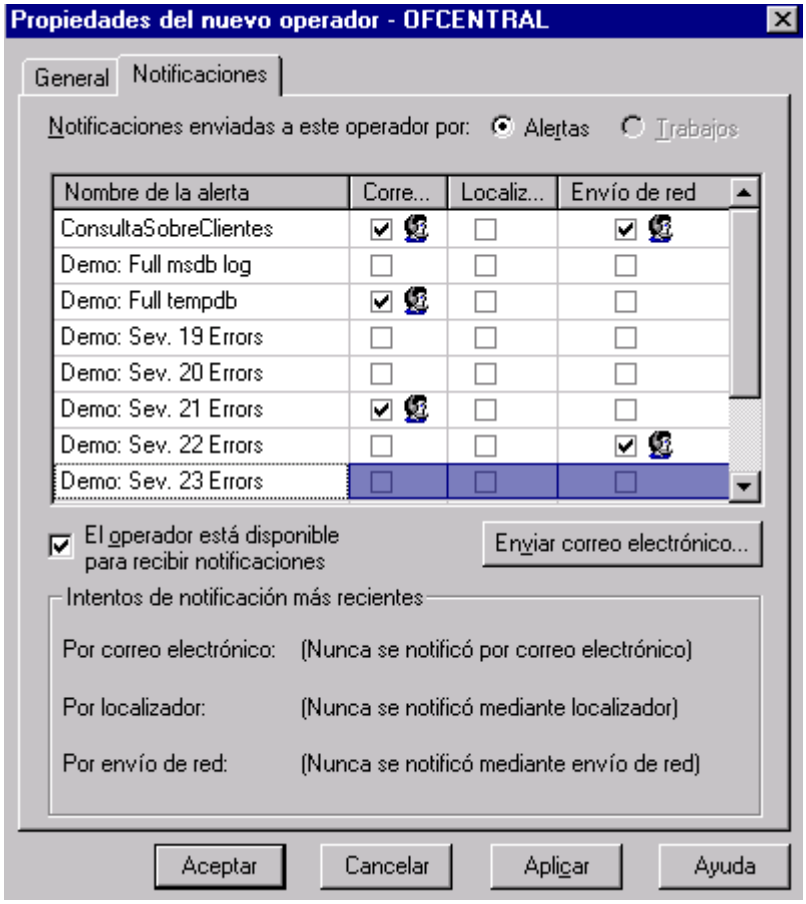


Figura 194. Selección de trabajos y alertas para notificar a un operador.

Establecidas todas las propiedades, pulsaremos Aceptar, con lo que se grabará el nuevo operador en SQL Server.

sp_add_operator

Procedimiento almacenado que permite agregar un nuevo operador al servidor SQL Server.

```
sp_add_operator 'Nombre'
```

- Nombre. Denominación para el operador.

El Código fuente 71 crea un nuevo operador de SQL Server, esta operación debe realizarse desde la base de datos msdb.

```
EXEC sp_add_operator 'Op1Ventas'
```

Código fuente 71

Alertas

A través de SQL Server podemos definir errores personalizados que serán invocados dentro de un procedimiento almacenado escrito por el usuario, que puede estar incluido en un programa que gestione la base de datos.

La alerta puede configurarse para que una vez que suceda, envíe un mensaje a un operador del sistema, por correo electrónico, mensaje de red, etc.

Supongamos una situación en la que existe una tabla de la base de datos con información sensible para la compañía, y es necesario conocer cuando se eliminan filas de dicha tabla. En este caso, podemos definir un operador que reciba dichas notificaciones, y una alerta que se producirá cada vez que se ejecute un procedimiento almacenado encargado de borrar los registros de la tabla.

En primer lugar, crearemos un operador al que llamaremos AuditorBorrados; el lector puede consultar el proceso de creación de auditores, en un apartado anterior de este mismo tema. Seguiremos con los pasos de creación de la alerta.

Administrador corporativo

Abriremos la carpeta Administración de SQL Server y el elemento Agente SQL Server; a continuación haremos clic con el botón derecho en el elemento Alertas, seleccionando la opción *Nueva alerta* del menú contextual, que nos mostrará la ventana de creación de alertas.

En esta ventana daremos un nombre a la alerta, especificaremos si estará o no habilitada, el número de error asignado a la alerta, base de datos sobre la cual actuará, etc.

Como vamos a crear una alerta personalizada, pulsaremos en el apartado *Definición de suceso de alerta*, la opción *Número de error*, y seguidamente pulsaremos el botón que hay junto al campo en el que se incluye el número de error, que nos mostrará la ventana de selección de errores, ver Figura 195.

En esta ventana, podemos buscar los mensajes de error ya definidos en SQL Server, para crear un nuevo número de error de usuario, haremos clic en la pestaña Mensajes, y aquí pulsaremos el botón Nuevo, que abrirá un cuado de diálogo en el que introduciremos el mensaje de nuestro error. Debemos tener en cuenta, que los número de error definidos por el usuario deben ser superiores al número 50000, y que SQL Server nos obliga al crear el mensaje, a definir su idioma como U.S.English, por lo que será recomendable crear en primer lugar, una versión del mensaje en este idioma y después un nuevo mensaje para el mismo error, esta vez con seleccionando en Idioma Spanish, ver Figura 196.

Los literales *%1!* y *%2!*, actuarán de parámetros sustituibles por valores al invocar el error.

Una vez creado el número de error, nuestra ventana de propiedades de la alerta tendrá un aspecto similar a la Figura 197.

Seguidamente pulsaremos en la pestaña Respuesta, para especificar las acciones a tomar cuando se genere la alerta. Podemos establecer un operador al que se enviará un mensaje, que se ejecute un trabajo, etc. Ver la Figura 198.

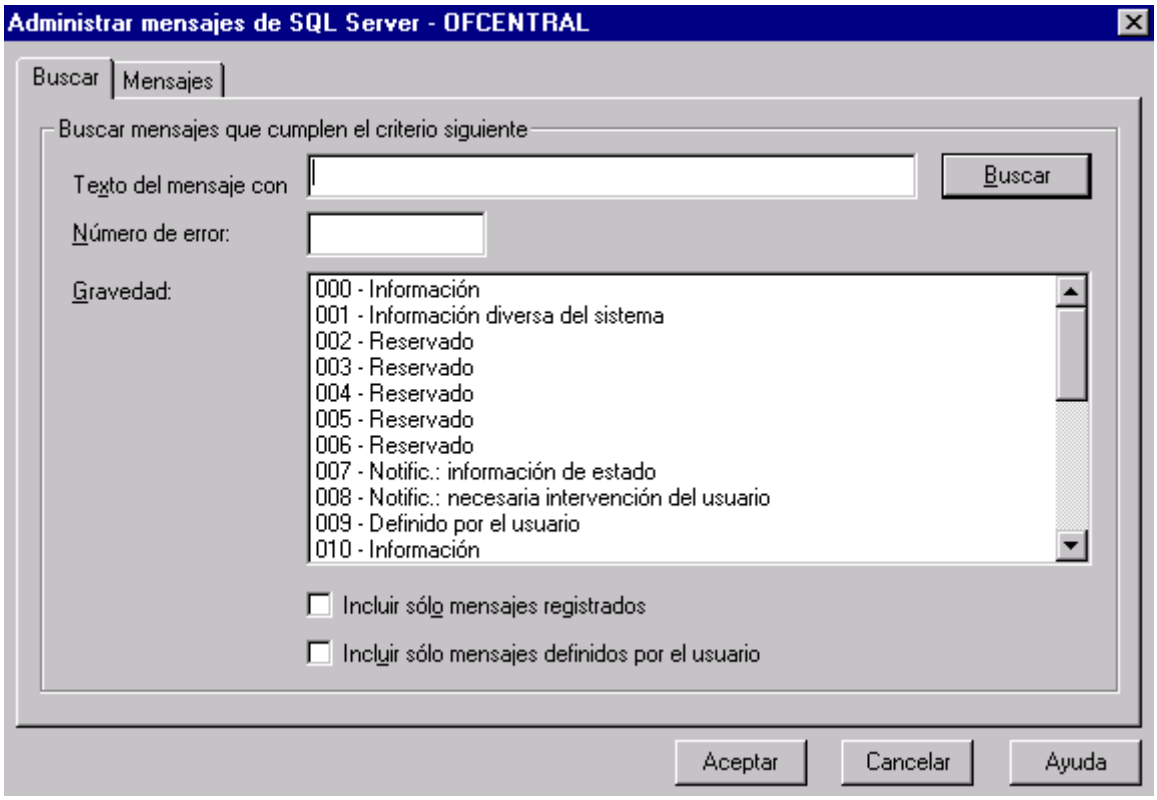


Figura 195. Ventana de selección del número de error asociado a la alerta.

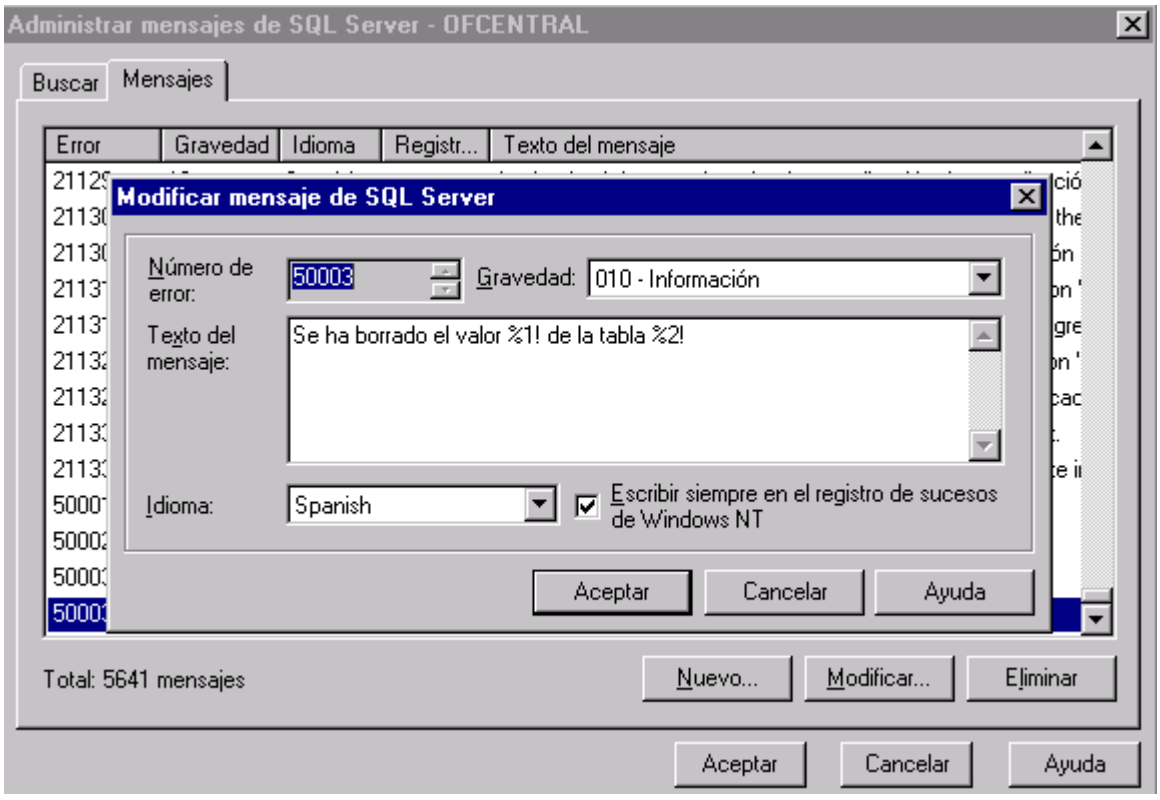


Figura 196. Definición del mensaje para un número de error.

Figura 197. Ventana de creación de una alerta.

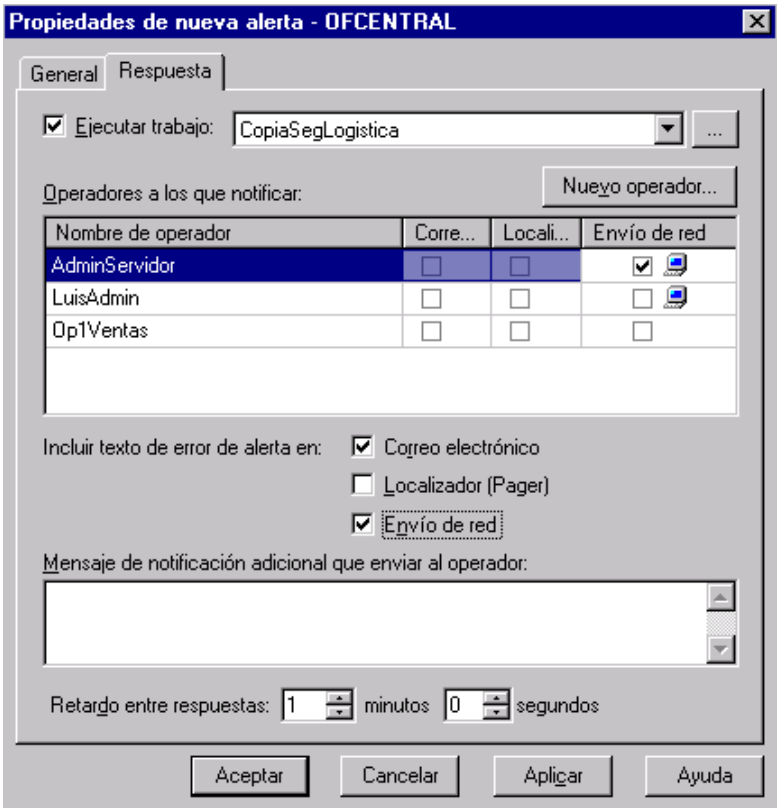


Figura 198. Opciones de respuesta a una alerta.

sp_add_alert

También podemos crear alertas ejecutando este procedimiento almacenado perteneciente a la base de datos del sistema msdb.

```
sp_add_alert 'Nombre' [, NumMensajeErr] [, Gravedad ] [, Habilitado]
```

- Nombre. Denominación para la alerta.
- NumMensajeErr. Número del mensaje de error que asociamos a la alerta. Si utilizamos un valor en el parámetro Gravedad, este parámetro debe ser 0 o NULL.
- Gravedad. Valor numérico de 1 a 25, que define el nivel de gravedad de la alerta. En el caso de usar el parámetro NumMensajeErr, este parámetro debe ser 0.
- Habilitado. Valor numérico que indica el estado de la alerta, 1 es habilitada, 0 es deshabilitada.

El ejemplo del Código fuente 72 muestra como crear una alerta mediante este procedimiento almacenado.

```
USE msdb
EXEC sp_add_alert 'NuevaAlerta', 50002,0,1
```

Código fuente 72

Procedimiento almacenado de llamada a la alerta

Una vez definida la alerta, podemos escribir un procedimiento almacenado que utilice la instrucción RAISERROR para invocarla.

```
RAISERROR (IDMensaje | CadenaMensaje , Gravedad, Estado
[, Parámetro [,...ParametroN] ] ) [WITH Opción[,...OpciónN]]
```

- IDMensaje. Número del mensaje de error.
- CadenaMensaje. Cadena con un mensaje para enviar de forma inmediata.
- Gravedad. Valor numérico de 1 a 25, que define el nivel de gravedad de la alerta.
- Estado. Valor numérico de 1 a 127, que proporciona información sobre el estado de llamada del error.
- Parámetro. Valores pasados al mensaje.
- Opción. Opciones asociadas al mensaje, que se muestran en la Tabla 13.

El Código fuente 73 muestra un procedimiento almacenado que realiza un borrado de fila sobre una tabla e invoca al error definido por el usuario en la alerta.

Valor	Descripción
LOG	Registra el error en el registro de errores del servidor y en el registro de la aplicación. Los errores guardados en el registro de errores del servidor están limitados actualmente a un máximo de 440 bytes.
NOWAIT	Envía inmediatamente los mensajes al cliente.
SETERROR	Establece el valor @@ERROR a IdMensaje o a 50000, independientemente del nivel de gravedad.

Tabla 13. Valores de opción que podemos utilizar en la instrucción RAISERROR.

```
CREATE PROCEDURE BorrarCliente
@IDCliente INT
AS
DELETE FROM Clientes
WHERE IDCliente=@IDCliente
RAISERROR (50003,10,1,@IDCliente,'Clientes')
```

Código fuente 73

Al haber definido una alerta para cuando se invoque a este error, el operador al que se ha establecido como destinatario de los mensajes resultantes de la alerta, recibirá el siguiente aviso del sistema operativo, véase Figura 199.

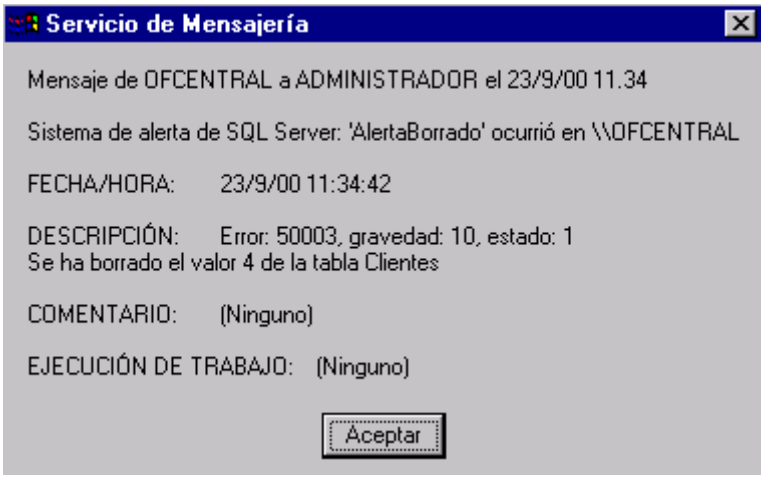


Figura 199. Notificación de alerta a un operador de SQL Server.

Definir alertas para problemas de rendimiento de SQL Server

Al crear una alerta, podemos especificar en su ventana de creación, que el tipo de alerta sea sobre el rendimiento del servidor SQL. En este caso, las propiedades a establecer para la alerta cambiarán, pudiendo provocar alertas referentes al rendimiento de las bases de datos, estadísticas, gestión de memoria, etc., como muestra la Figura 200.

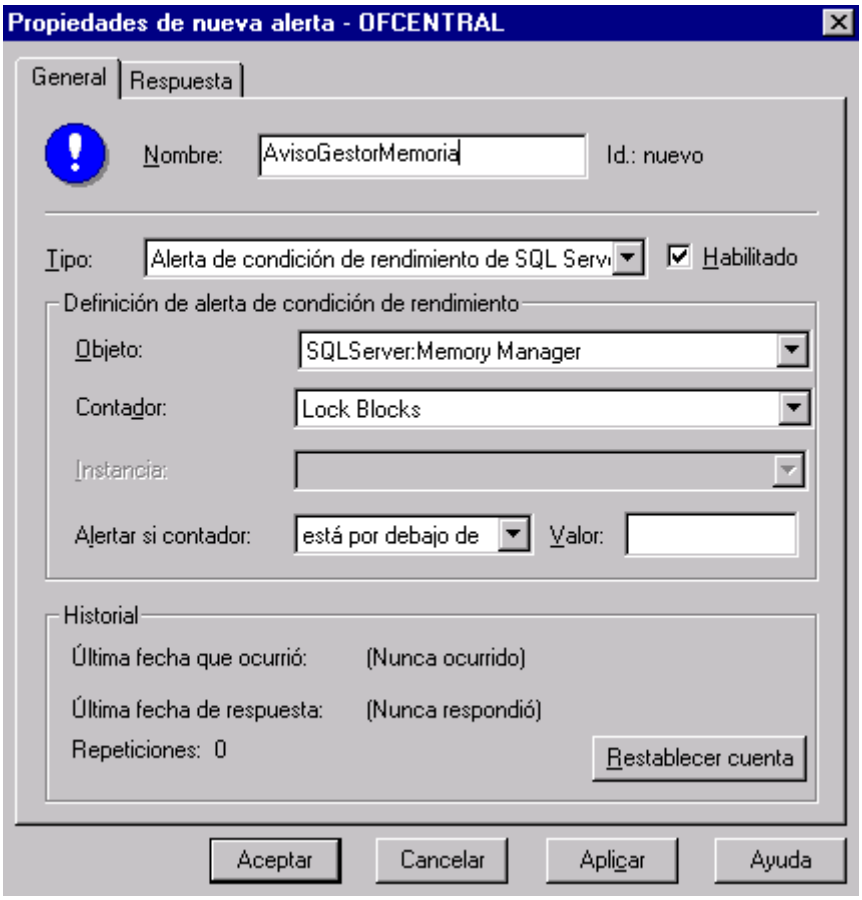

Propiedades de nueva alerta - OFCENTRAL
General
Respuesta
Nombre:
AvisoGestorMemoria
Id.: nuevo
Tipo:
Alerta de condición de rendimiento de SQL Serv
Habilitado
Definición de alerta de condición de rendimiento
Objeto:
SQLServer:Memory Manager
Contador:
Lock Blocks
Instancia:
Alertar si contador:
está por debajo de
Valor:
Historial
Última fecha que ocurrió:
(Nunca ocurrido)
Última fecha de respuesta:
(Nunca respondió)
Repeticiones: 0
Restablecer cuenta
Aceptar
Cancelar
Aplicar
Ayuda

Figura 200

20

Publicación en Internet

Distribución de datos SQL Server en Internet

SQL Server permite a través de un sencillo asistente, crear ficheros HTML para su publicación en Internet, con la información que seleccionemos de una base de datos.

Supongamos que una empresa quiere publicar en Internet la información sobre sus delegaciones, de forma que los clientes de dicha empresa, al conectar con su página web, puedan saber las sucursales que tiene, emplazamiento geográfico y horario.

También necesitaremos que cuando cambie o se incorpore alguna nueva sucursal a la empresa, dicha información sea actualizada de inmediato en la página web encargada de reflejar estos datos.

Administrador corporativo

Situados en la carpeta Administración de SQL Server, haremos clic con el botón derecho en el elemento *Publicación en Web*, eligiendo la opción del menú contextual *Nuevo trabajo del Ayudante de Web*, que pondrá en marcha este asistente.La Figura 201 muestra la pantalla inicial del asistente.

El siguiente paso consiste en elegir la base de datos de la cual vamos a publicar información en la Web, Figura 202.

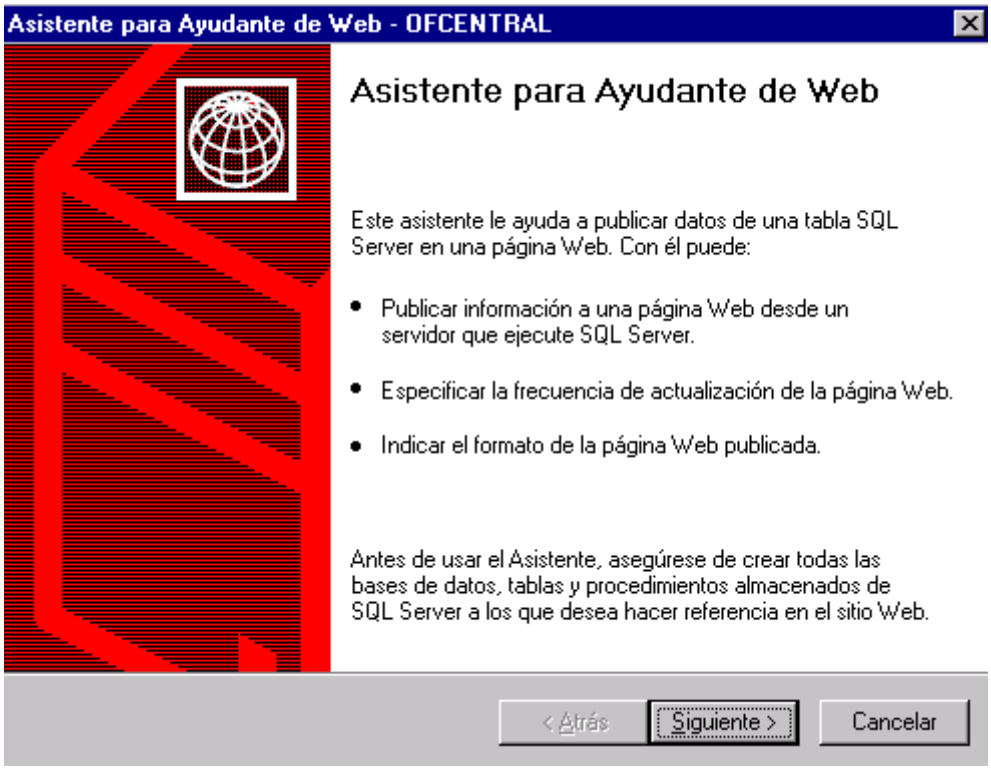


Figura 201. Inicio del asistente para Ayudante de Web.

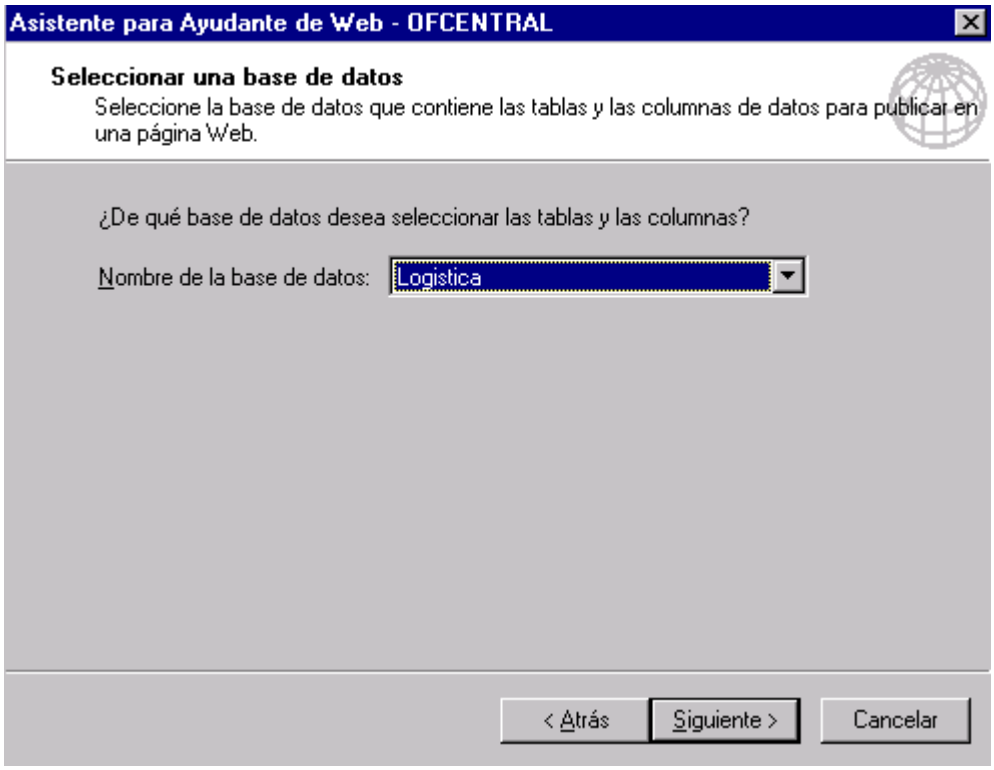


Figura 202. Selección de la base de datos a publicar.

Pulsando el botón Siguiente, deberemos escribir el nombre de esta tarea de publicación Web y el tipo de información que vamos a publicar, que será obtenida de las tablas y columnas de la base de datos, como muestra la Figura 203.

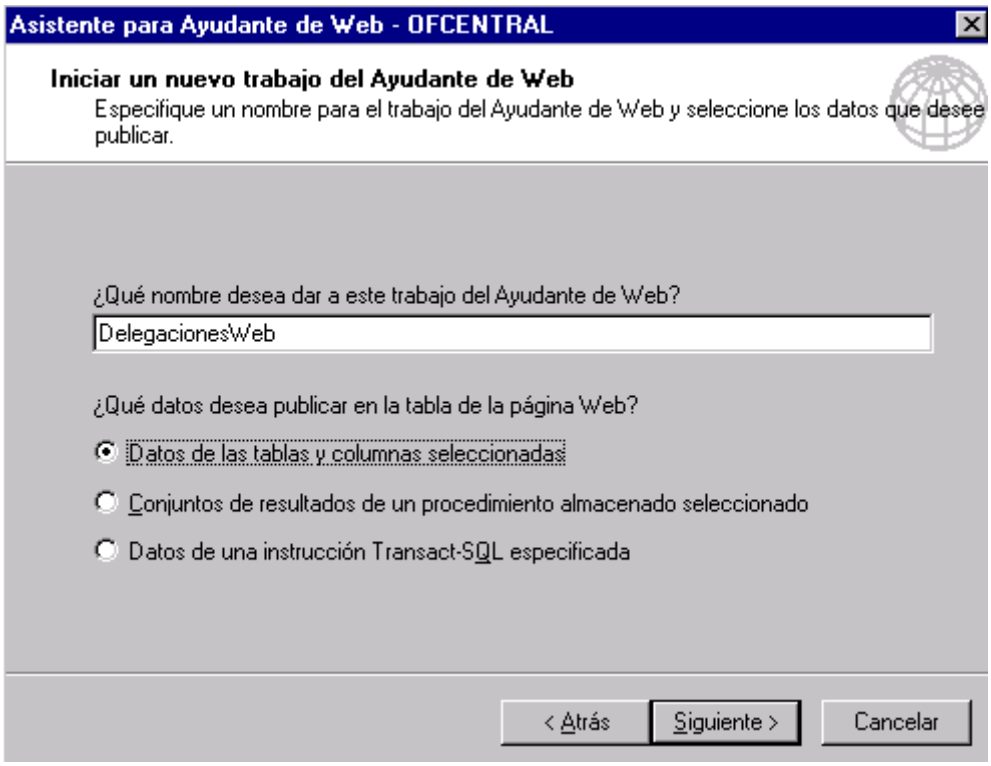


Figura 203. Nombre de la publicación y selección del tipo de información a publicar.

En el siguiente paso, seleccionaremos la tabla y columnas a mostrar en la página web, es decir, la tabla Delegaciones y todas sus columnas menos el número identificador de la delegación, Figura 204.

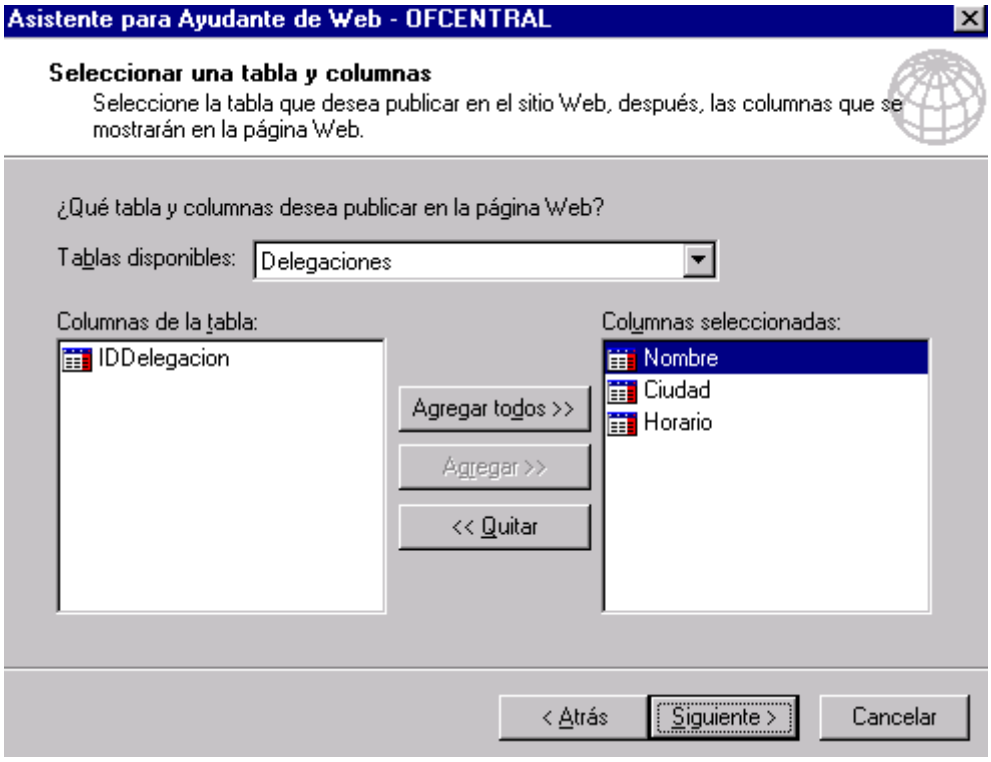


Figura 204. Selección de tabla y columnas a mostrar.

A continuación estableceremos un filtro de selección de datos si es necesario. La forma de crearlo es muy sencilla, podemos indicar que se devuelvan todas las filas de la tabla, o seleccionar en diversos controles de esta ventana, tanto la columna que hará de filtro como las condiciones y valores para el mismo. En el caso de que queramos nosotros escribir las instrucciones de la cláusula WHERE, también podremos hacerlo en la parte inferior de la ventana, véase la Figura 205.

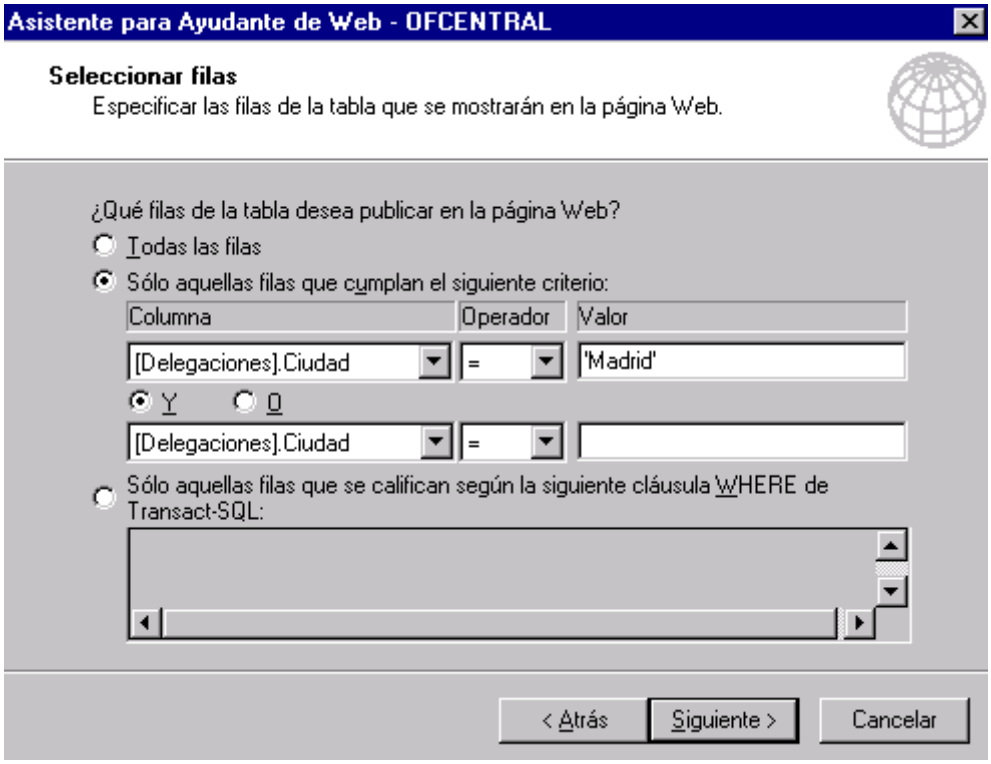


Figura 205. Opciones para establecer el filtro de datos a seleccionar.

En el próximo paso debemos indicar cuando se actualizarán los datos en la página web. Por defecto, los datos no se actualizan, se muestran de modo estático; este modo corresponde a la opción *Sólo una vez, al completar el asistente*. Si necesitamos refrescar la información podemos seleccionar alguna de las otras opciones, que nos permiten actualizar la información del fichero HTML cada vez que los datos de la tabla cambien, al cumplirse una fecha, cada cierto intervalo, etc., estas opciones las vemos en la Figura 206.

Si por ejemplo, decidimos que la página se refresque cuando cambie la información de la tabla asociada, el siguiente paso del asistente, nos pedirá que especifiquemos en base a que columna(s) de la tabla se deberán comprobar los cambios sobre la tabla, Figura 207.

En el siguiente paso indicaremos la ruta y nombre del fichero HTML en el que se generará la página con los datos a publicar en Internet, Figura 208.

Si disponemos de un archivo que sirva de plantilla para que el asistente deposite los datos, lo especificaremos en este paso; en caso contrario, indicaremos que necesitamos que el asistente nos ayude a crear el formato de la página web, Figura 209.

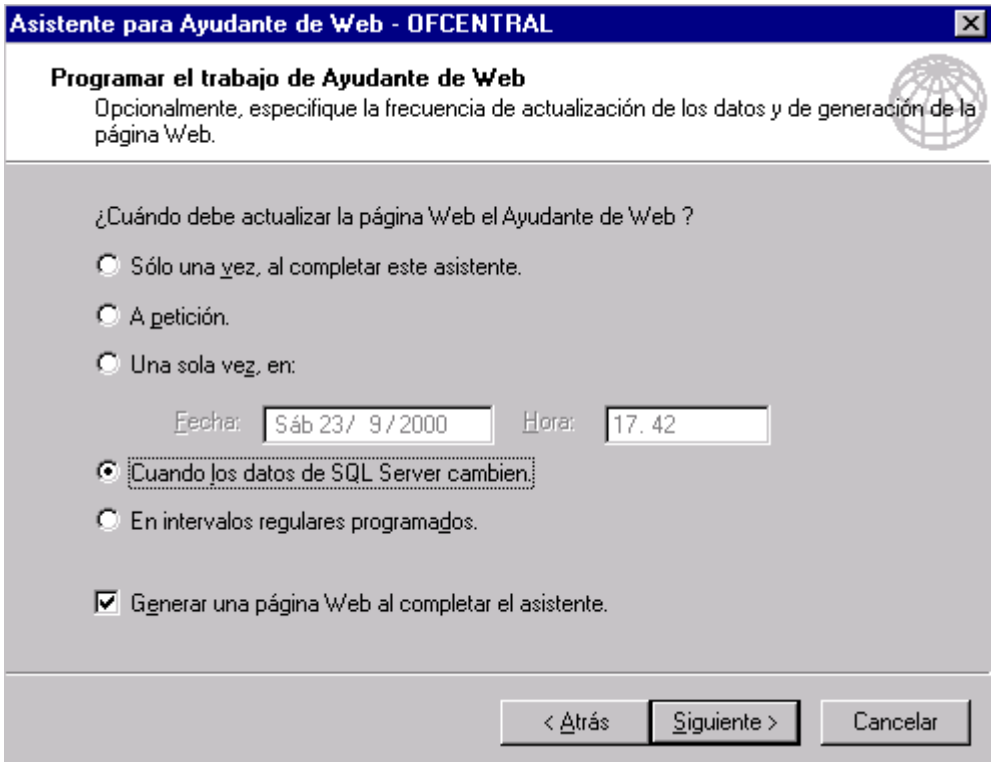


Figura 206. Modos de actualización de los datos de la tabla a la página web.

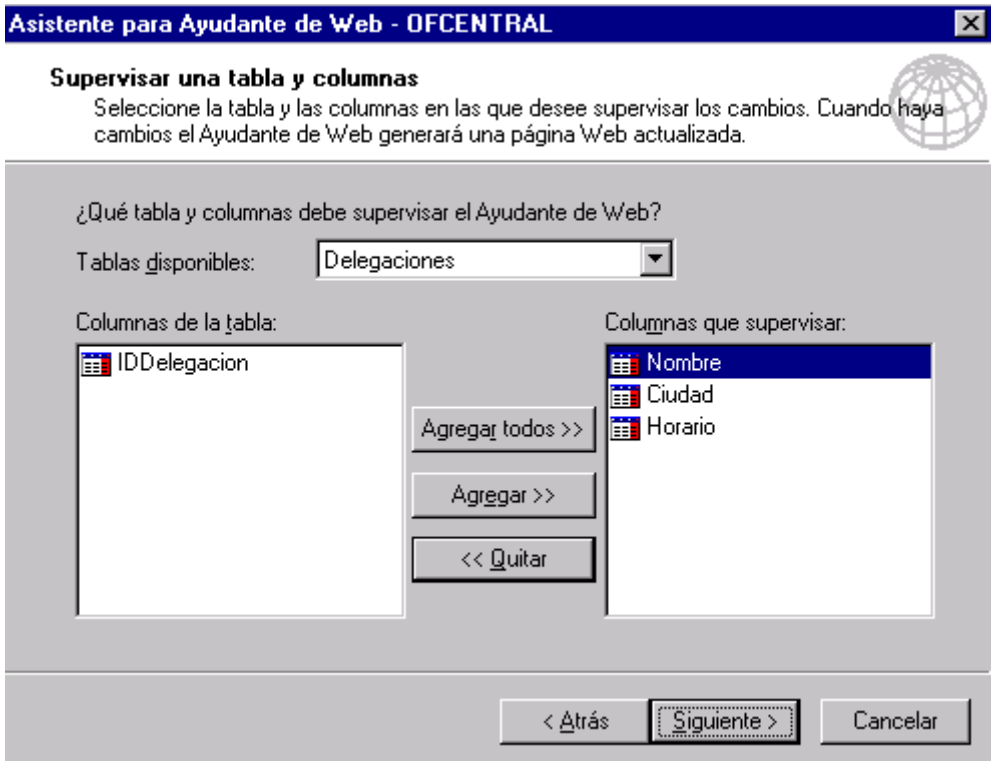


Figura 207. Indicar columnas para realizar la actualización de la página al cambiar el valor.

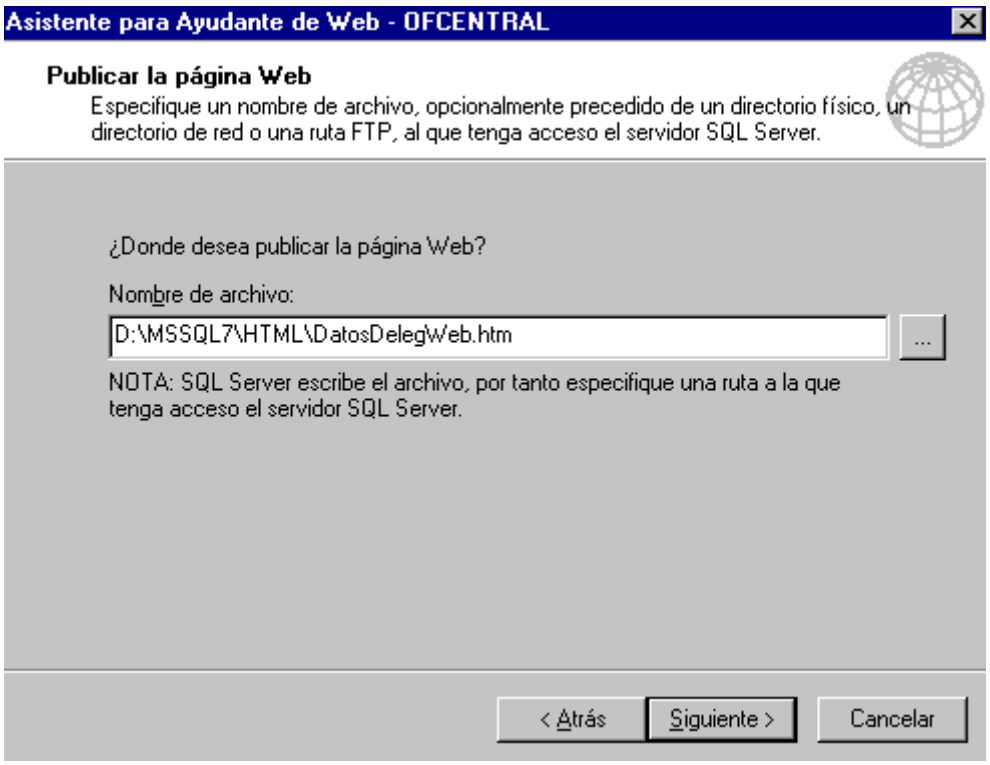


Figura 208. Especificar fichero HTML a generar.

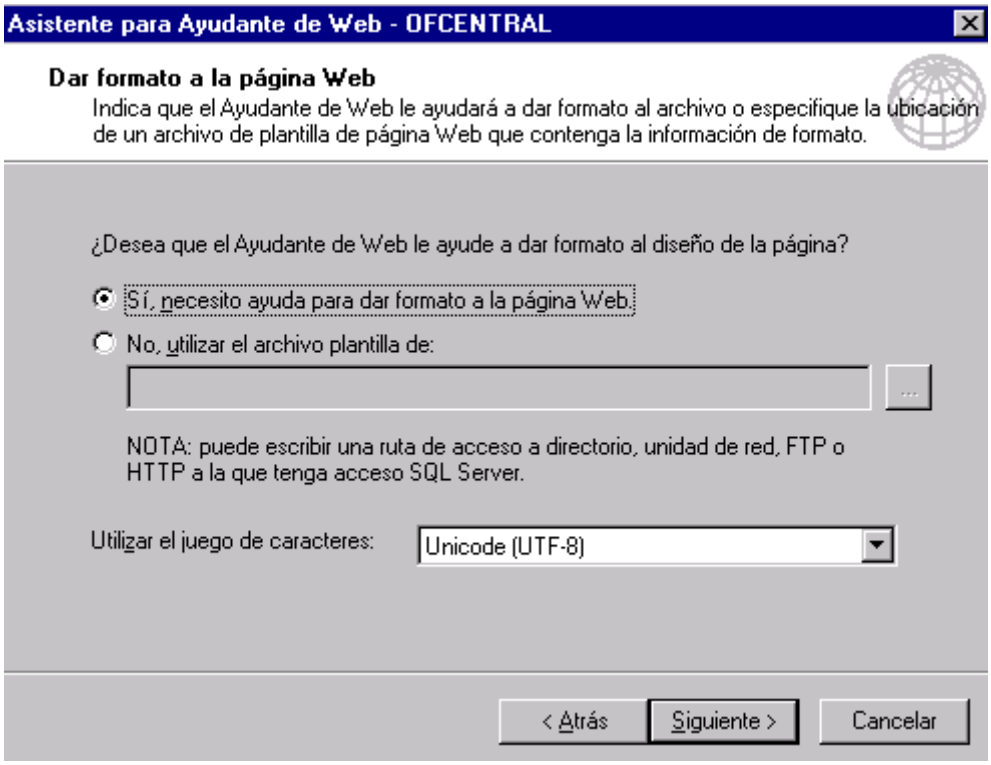


Figura 209. Opciones para crear el formato de la página web.

A continuación, escribiremos el título para la página y la tabla, Figura 210.

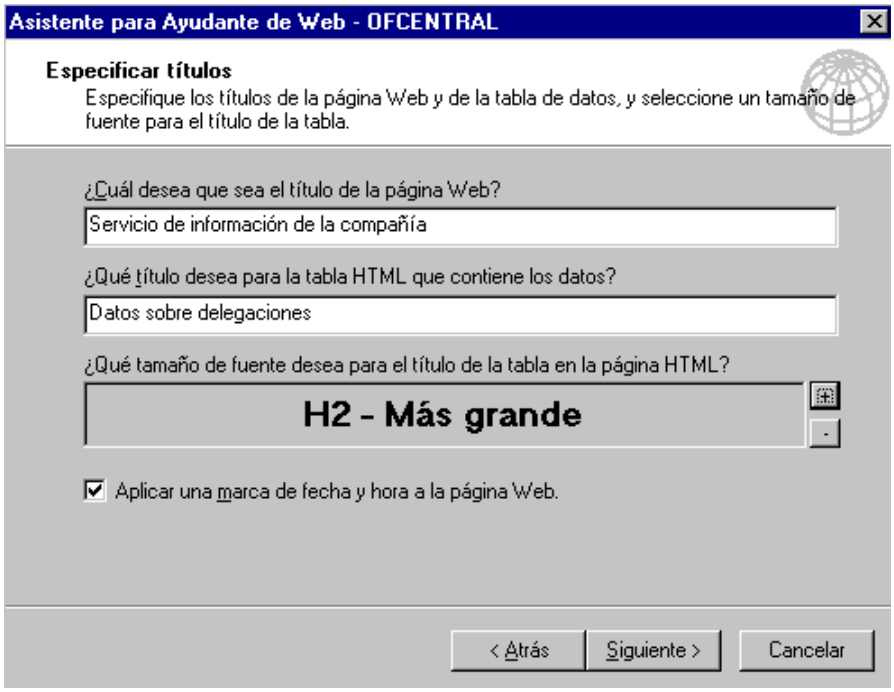


Figura 210. Títulos y tamaño de letra para la página.

Después indicaremos si queremos títulos para las columnas de la tabla y el formato de letra, Figura 211.

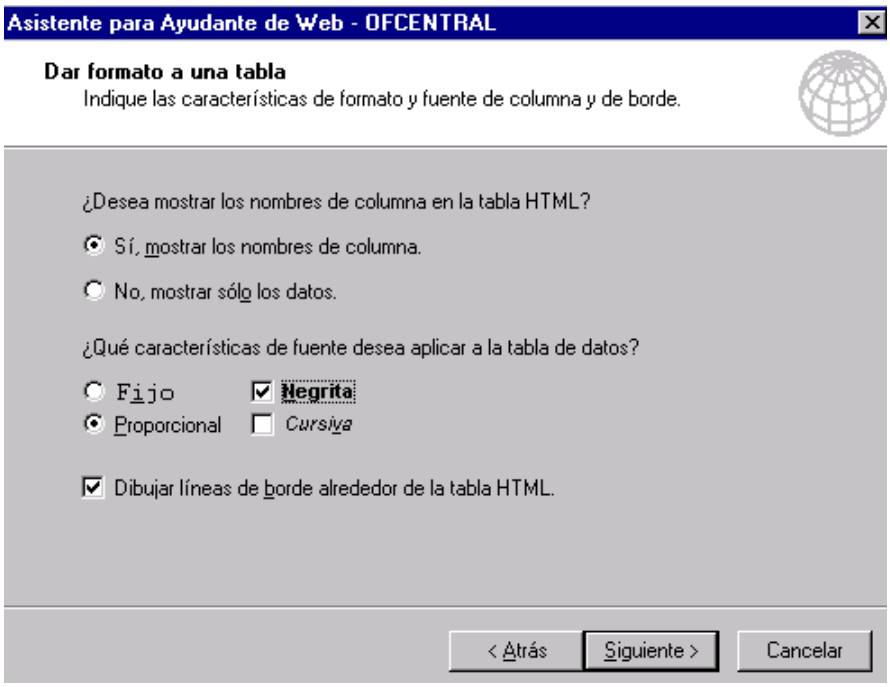


Figura 211. Títulos de columnas y formato de letra.

Esta página web necesitará probablemente retornar o direccionarse a otra página de la empresa, por lo que en el siguiente paso podemos indicar la información de un hipervínculo o una lista de hipervínculos a donde podremos pasar después de consultar la información, ver Figura 212.

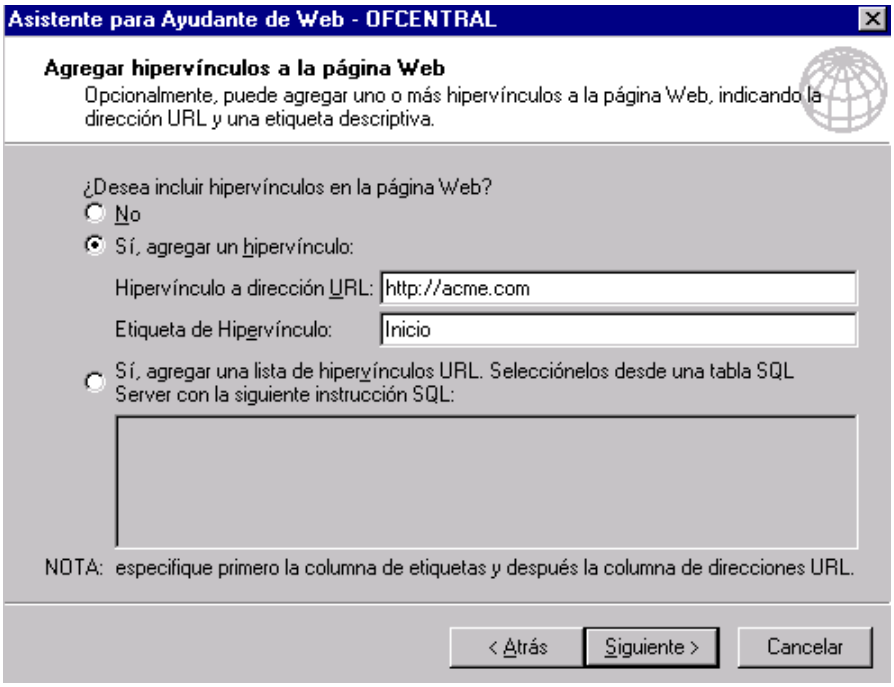


Figura 212. Hipervínculos de la página.

Para el caso de que la página muestre un gran número de datos, en este paso podemos limitar la cantidad de filas que serán mostradas.

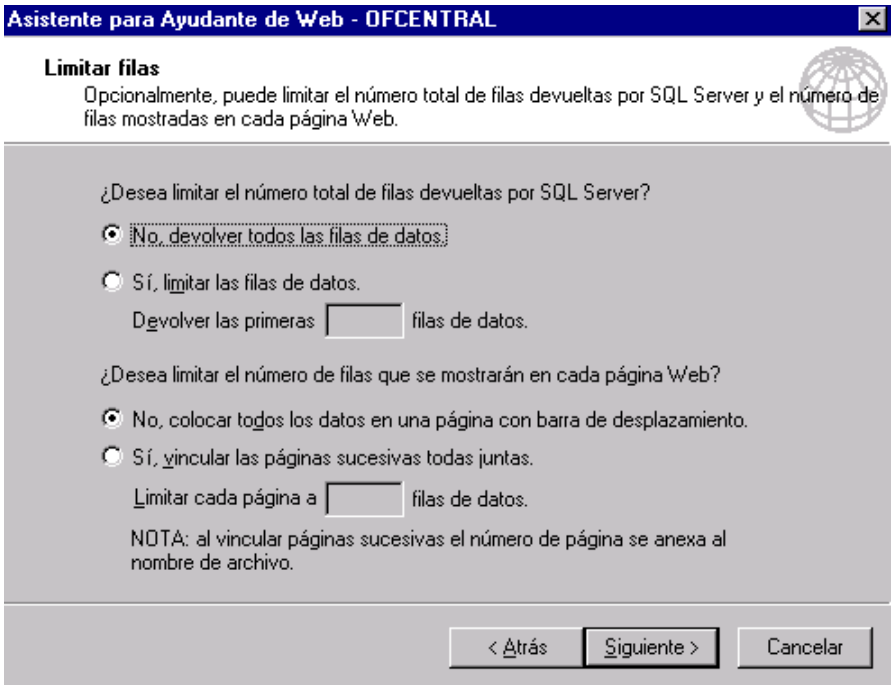


Figura 213. Especificación de la cantidad de filas a mostrar.

Y por fin llegamos al paso final del asistente. Si toda la información introducida es correcta, pulsaremos el botón Finalizar, que creará la página web, Figura 214.

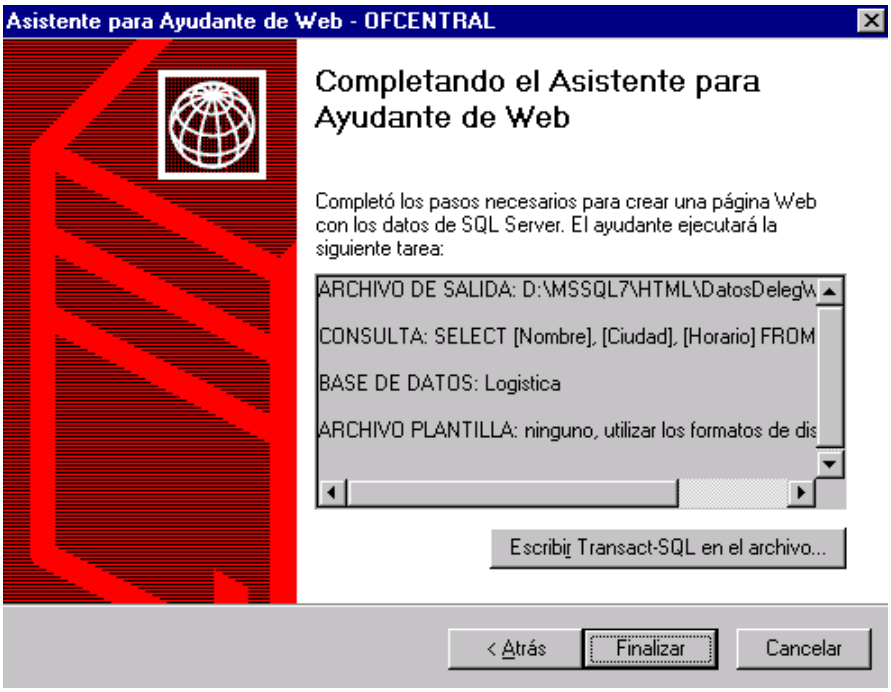


Figura 214. Fin del asistente para publicación en Web de SQL Server.

La Figura 215 muestra el resultado de la ejecución de la página HTML con los datos solicitados.

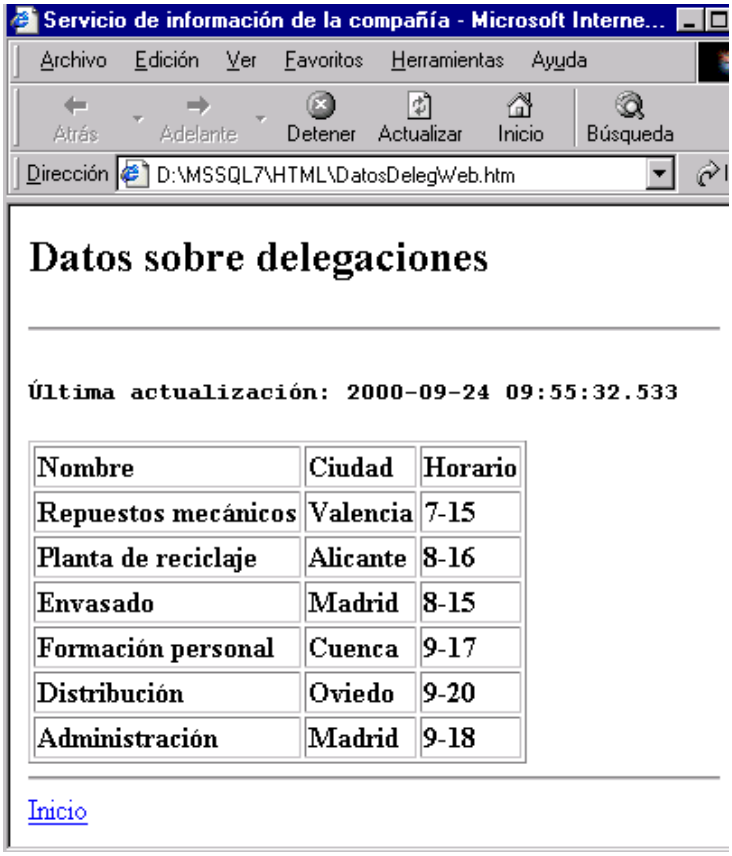


Figura 215. Página web con la información de una tabla de la base de datos.

Procedimientos almacenados para publicación en Web

Los siguiente procedimientos almacenados nos permiten desde código, publicar información de una base de datos en Internet, de igual forma que utilizando el asistente comentado anteriormente.

sp_makewebtask

Este procedimiento almacenado nos permite desde código, crear una página de Internet con la información obtenida de una consulta contra una base de datos.

```
sp_makewebtask [@outputfile =] 'ArchivoHTML', [@query =]
'Consulta' [, [@fixedfont =] FuenteFija] [, [@bold =] Negrita]
[, [@italic =] Cursiva] [, [@colheaders =] EncabezadosColumna]
[, [@lastupdated =] ÚltimaActualización] [, [@HTMLHeader =]
EncabezadoHTML] [, [@username =] Usuario] [, [@dbname =] 'NombreBD'
```

- ArchivoHTML. Nombre y ruta del fichero HTML resultante.
- Consulta. Sentencia SQL que devuelve los datos a mostrar en la página HTML.
- FuenteFija. Valor numérico, si contiene 1 (predeterminado) se utiliza una fuente fija, si el valor es 0 se utiliza una fuente proporcional.
- Negrita. Valor numérico, si contiene 1 la fuente usa negrita, si es 0 (predeterminado) no se usa.
- Cursiva. Valor numérico, si contiene 1 la fuente usa cursiva, si es 0 (predeterminado) no se usa.
- EncabezadosColumna. Valor numérico, si contiene 1 (predeterminado) se visualizarán los encabezados de las columnas, si es 0 no se mostrarán los encabezados.
- ÚltimaActualización. Valor numérico, si contiene 1 (predeterminado) la página muestra la fecha en que actualizaron los datos mostrados, si el valor es 0 no se muestra la fecha.
- EncabezadoHTML. Código HTML correspondiente al texto mostrado en el título. La Tabla 14 muestra los valores disponibles

Valor	Código de formato HTML
1	H1
2	H2
3	H3
4	H4
5	H5

6	H6

Tabla 14. Valores para el tipo de encabezado de la página web.

- Usuario. Nombre del usuario que efectúa la consulta.
- NombreBD. Nombre de la base de datos sobre la que se ejecuta la consulta.

El Código fuente 74 muestra la creación mediante este procedimiento almacenado, de un fichero HTML con una consulta sobre una base de datos.

```
EXEC sp_makewebtask @outputfile='D:\MSSQL7\HTML\INFOCLI.HTM',
@query='SELECT * FROM CLIENTES',
@dbname='Logistica'
```

Código fuente 74

sp_runwebtask

Ejecuta un trabajo de publicación en web ya creado.

```
sp_runwebtask [@procname=]'PublicaciónWeb' ,[@outputfile =]
'FicheroHTML'
```

- PublicaciónWeb. Nombre de la publicación web a ejecutar.
- FicheroHTML. Ruta y nombre del fichero HTML en el que se grabarán los datos de la publicación.

El Código fuente 75 muestra un ejemplo de uso de este procedimiento almacenado.

```
EXEC sp_runwebtask @procname='DelegacionesWeb', @outputfile='D:\MSSQL7\DELEG.HTM'
```

Código fuente 75

Plantilla HTML para personalizar la salida de los datos

En el caso de que el diseño estándar utilizado por el asistente de publicación en Web o el procedimiento almacenado sp_makewebtask no se adapte a nuestras necesidades, podemos crear un fichero HTML que el asistente utilice como plantilla.

Utilizando como marcadores las etiquetas <%insert_data_here%>, <%begindetail%> y <%enddetail%>, indicaremos los puntos de la página web en donde se depositarán los datos. El Código fuente 76 muestra en un fragmento de código de un fichero HTML cómo se emplean estos marcadores.

```
<TABLE>
<TR>
```

```
<TH>
Nombre
</TH>

<TH>
Dirección
</TH>
</TR>

<%begindetail%>

<TR>
<TD>
<%insert_data_here%>
</TD>

<TD>
<%insert_data_here%>
</TD>
</TR>

<%enddetail%>

</TABLE>
```

Código fuente 76

21

Servicios de transformación de datos

Algo más que una simple importación o exportación de datos

Los servicios de transformación de datos (Data Transformation Services), DTS a partir de ahora, ofrecen una amplia gama de operaciones de importación y exportación de información entre orígenes de datos de muy diverso tipo.

Gracias a su sólida cimentación, basada en tecnología OLE DB, estos servicios permiten efectuar transferencias entre cualquier origen de datos, sean o no relacionales, ya que lo importante es que el origen de datos disponga del correspondiente proveedor OLE DB.

Ciñéndonos al ámbito de SQL Server, contamos con la ventaja adicional, de que al efectuar transferencias entre bases de datos de este tipo, podremos además de datos, importar y exportar cualquier tipo de objeto de base de datos SQL Server: tablas, procedimientos almacenados, triggers, etc.

Otra característica muy destacable es la posibilidad de modificar la información durante el proceso de transferencia para que guarde el mayor nivel de coherencia en la base de datos destino.

Transferencia de datos entre bases de datos del mismo tipo

Vamos a comenzar realizando el tipo de traspaso de datos más simple: copiar una tabla entre dos bases de datos SQL Server.

En la base de datos Almacenes tenemos la tabla Zonas que vamos a exportar a la base de datos Logística, ubicada en el mismo servidor SQL Server.

En primer lugar seleccionaremos la opción de menú del Administrador corporativo *Herramientas+Servicios de transformación de datos+Exportar datos*, que iniciará el asistente de DTS correspondiente a la transferencia de datos solicitada, ver Figura 216.

Figura 216. Asistente de transformación de datos.

DTS dispone de diversas utilidades y herramientas para la transformación de datos, pero este asistente de importación/exportación, será el que con toda seguridad, empleemos con mayor frecuencia, debido a su gran sencillez de manejo y enorme potencia.

Al pulsar el botón Siguiente, seleccionaremos el origen de datos de la exportación que vamos a realizar. Como muestra la Figura 217, debemos elegir de una lista desplegable, el tipo de base de datos que actuará como origen.

En este caso, debido a que el origen es SQL Server, indicaremos el servidor en donde está alojada la base de datos, el tipo de autenticación necesario para conectar y la base de datos origen.

El siguiente paso es idéntico a este último, sólo que en este caso, debemos especificar la base de datos destino a la que se exportará la tabla, ver Figura 218.

A continuación debemos especificar qué deseamos copiar, en este caso marcaremos la opción *Copiar las tablas de la base de datos de origen*, como muestra la Figura 219.

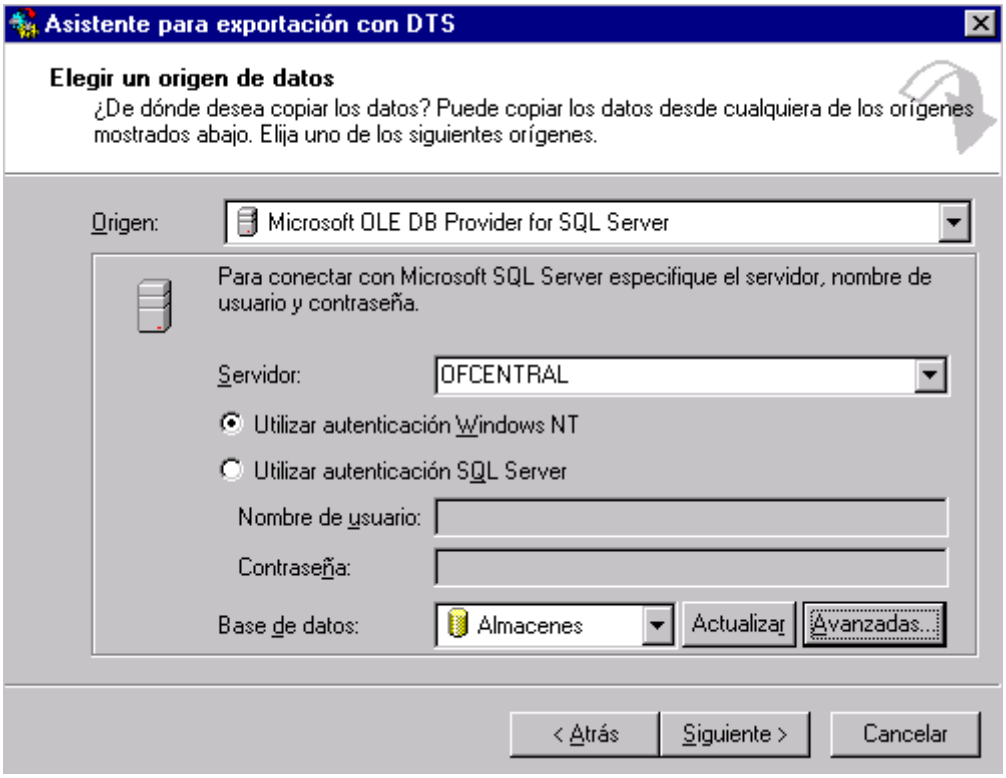


Figura 217. Elegir origen de datos de la exportación.

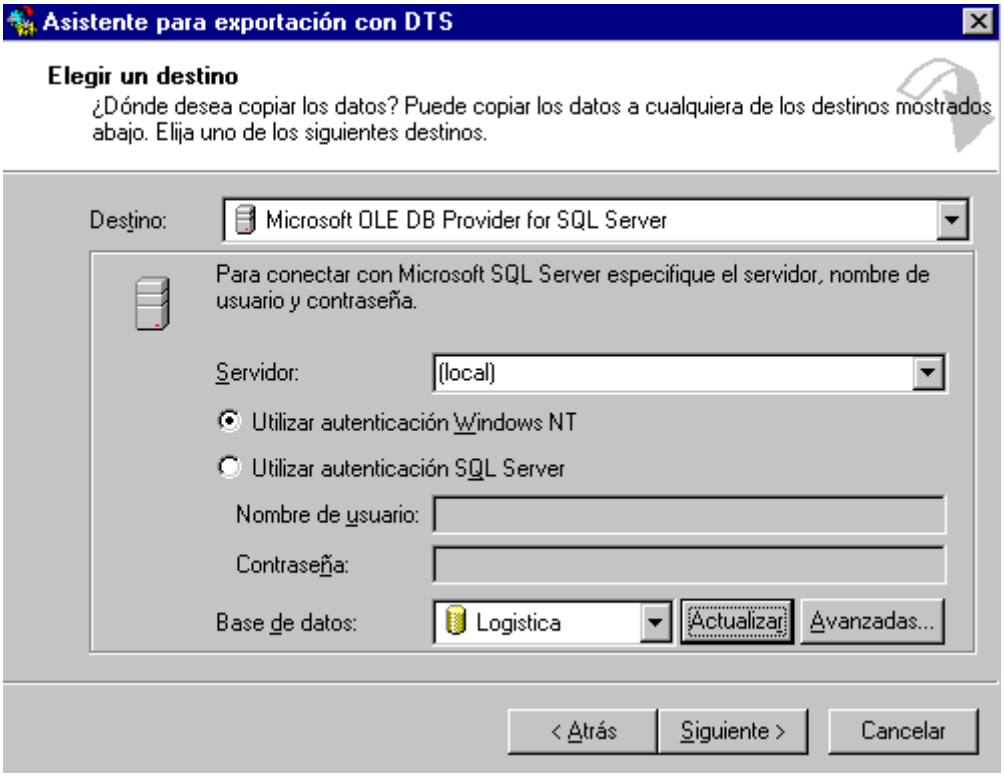


Figura 218. Elegir fuente de datos destino.

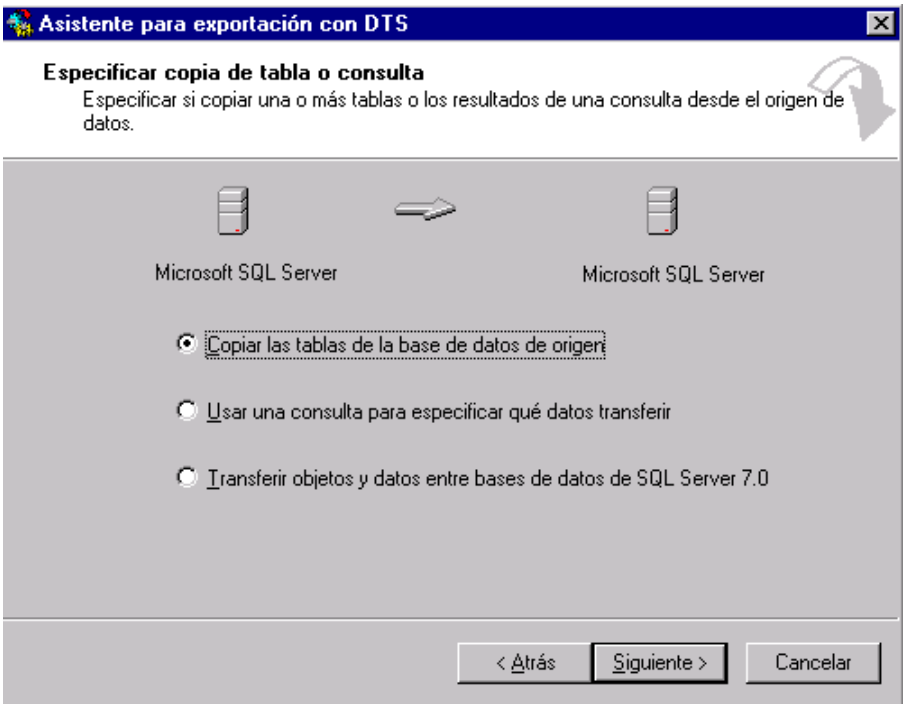


Figura 219. Selección de elementos a copiar desde la base de datos origen.

Las otras opciones de este paso nos permiten también escribir una consulta SQL, que seleccione un subconjunto de filas a copiar, o elegir objetos de la base de datos que no sean exclusivamente los datos de tablas; esta última opción sólo está disponible cuando se realizan transferencias entre bases de datos SQL Server. Ya que hemos elegido copiar tablas, en el siguiente paso deberemos indicar cuáles de las tablas de la base de datos de origen copiar a la base de datos destino. En este caso marcaremos la tabla Zonas, como muestra la Figura 220.

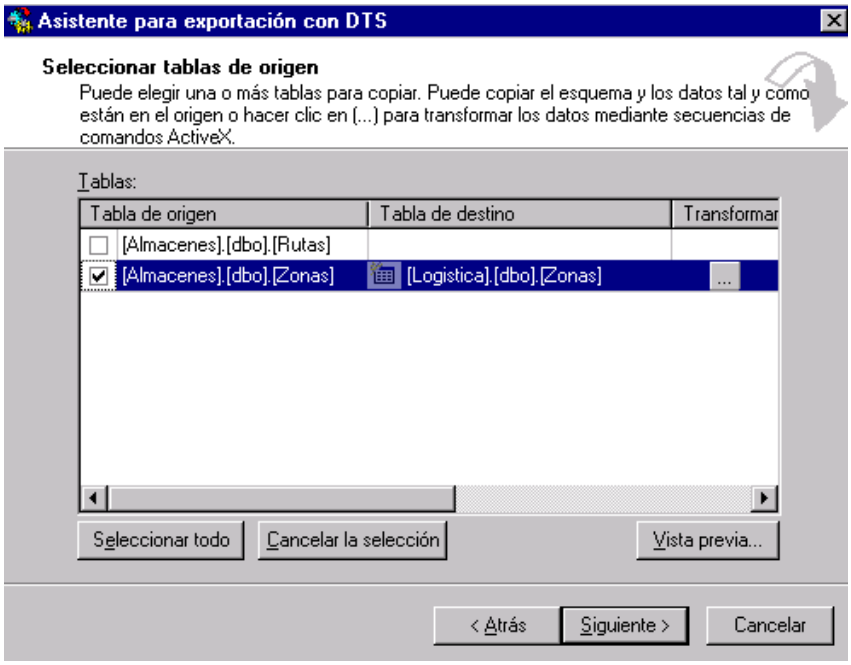


Figura 220. Selección de tablas a copiar.

Al llegar al siguiente paso debemos decidir si ejecutamos directamente la operación de transferencia que hemos diseñado, o si bien queremos guardarla antes en un paquete, que son los elementos utilizados por SQL Server para almacenar las operaciones del DTS, ver Figura 221.

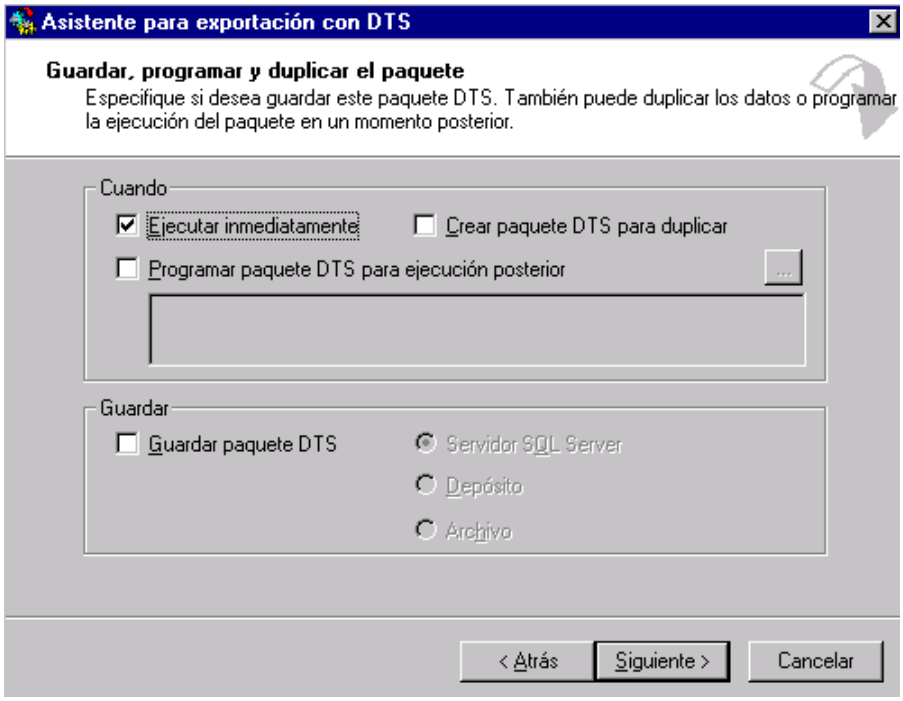


Figura 221. Opciones de grabación de la transferencia de datos a un paquete DTS.

En este caso no grabaremos las acciones definidas, por lo que pasaremos a la última pantalla de este asistente, que se muestra en la Figura 222.

Figura 222. Final del asistente DTS.

Al pulsar el botón Finalizar, se llevará a cabo el proceso de exportación de la tabla Zonas, desde la base de datos Almacenes a Logística. Durante esta operación seremos informados en todo momento de los pasos realizados mediante la ventana de la Figura 223.

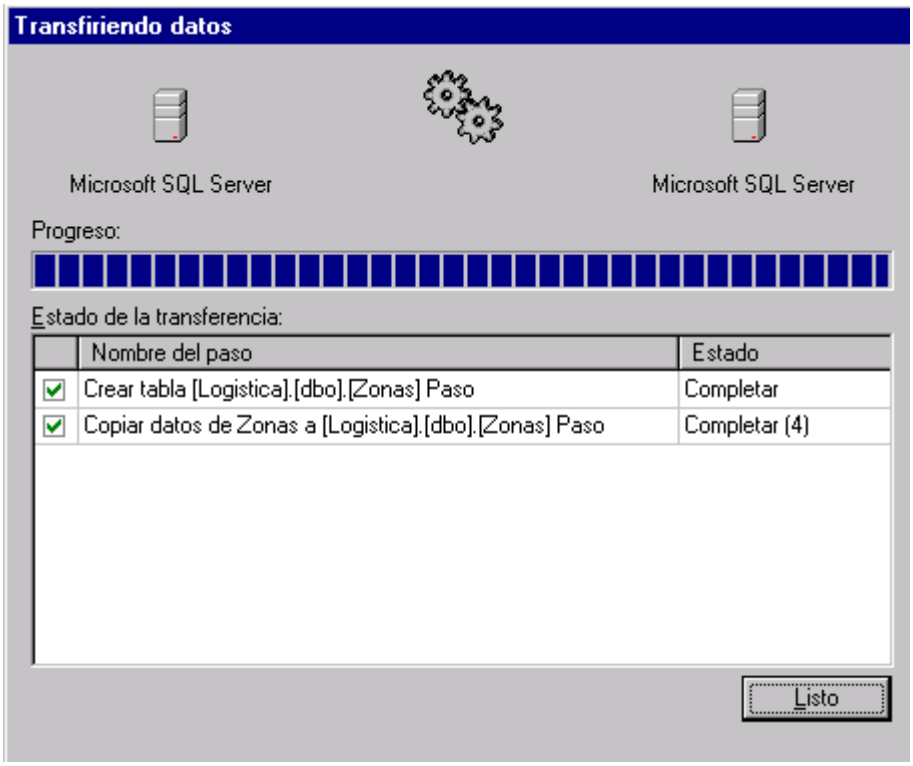


Figura 223. Ventana informativa del DTS sobre la transferencia en ejecución.

Completada la operación de transferencia y revisados todos los pasos, pulsaremos el botón Listo de esta ventana, con lo que tendremos la nueva tabla en la base de datos destino.

Pudiera darse el caso de que el lector al realizar una prueba parecida, no le aparezca la nueva tabla en la base de datos destino. Esto puede ser debido a que no se ha refrescado la información que ofrece el Administrador corporativo respecto a dicha base de datos, sin embargo la tabla sí que se ha copiado.

Para solucionar este inconveniente, haremos clic con el botón derecho sobre el nombre de la base de datos y elegiremos la opción del menú contextual *Actualizar*, que refrescará la información de la base de datos, mostrando todas las tablas disponibles.

Paquetes DTS

En el ejemplo del apartado anterior, hemos visto cómo al ejecutar una operación de transferencia de datos con el asistente DTS, se nos ofrecía la posibilidad de grabar el proceso en un elemento denominado paquete DTS.

Un paquete DTS, es el elemento utilizado por los servicios de transformación de datos para guardar los pasos de un proceso de transformación como una unidad.

Grabación de un paquete

En la ventana de operación de transferencia antes mencionada, al marcar la casilla *Guardar paquete DTS*, además de indicar que vamos a grabar los pasos realizados, designamos una ubicación, que puede ser el servidor (opción predeterminada), el depósito o un archivo; ver Figura 224.

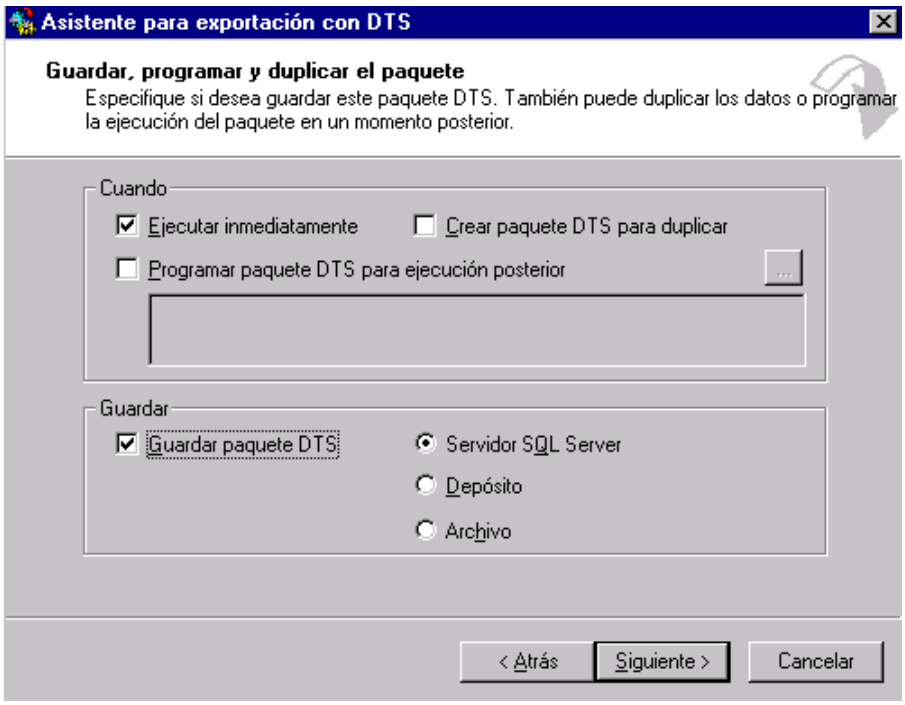


Figura 224. Grabación de la transferencia de datos a un paquete.

En esta situación, en el siguiente paso del asistente, deberemos proporcionar la información para el paquete DTS, como un nombre, descripción, contraseña, etc., véase la Figura 225.

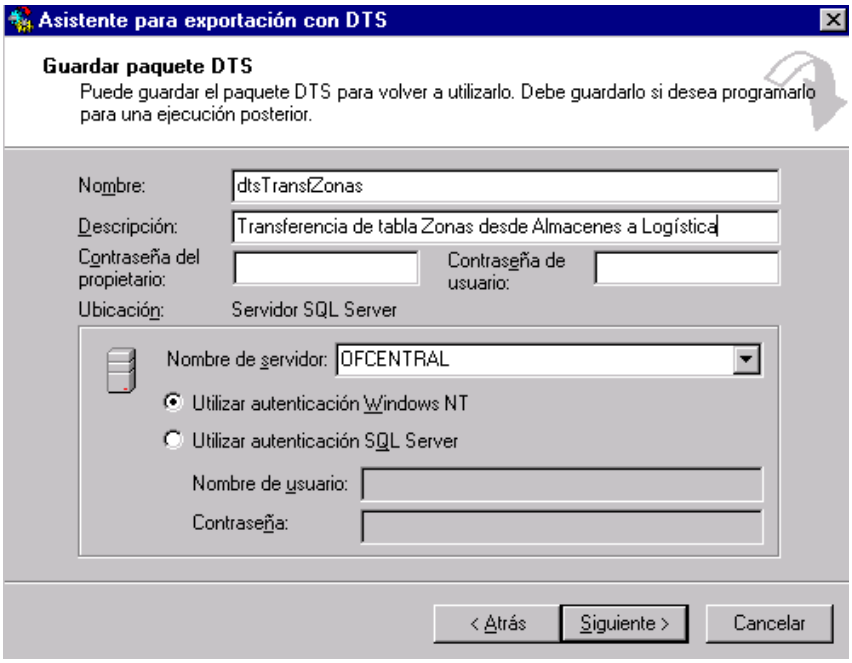


Figura 225. Introducción de datos para el paquete DTS.

Damos el nombre dtsTransfZonas y una descripción de lo que hace el paquete. El resto de pasos hasta el final del asistente ya han sido explicados.

Podemos ver las ubicaciones en las que se depositan los paquetes, abriendo la carpeta *Servicios de transformación de datos*, del servidor SQL Server. Dentro de esta carpeta tenemos los elementos *Paquetes locales* y *Paquetes de depósito*, ver Figura 226.

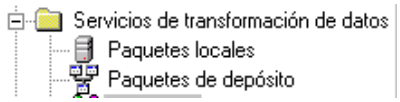


Figura 226. Elementos para grabar paquetes del DTS.

La Figura 227 muestra el elemento *Paquetes locales* con un conjunto de paquetes de este tipo en su interior.

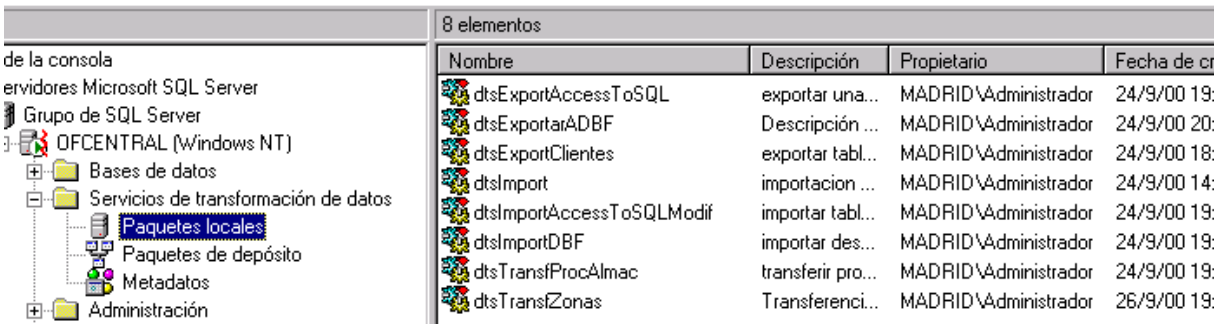


Figura 227. Lista de paquetes locales de DTS.

Cuando en la ventana de grabación del paquete del asistente DTS, elegimos grabar en el servidor SQL Server, el paquete se deposita en *Paquetes locales* (la información se graba en la base de datos del sistema msdb), mientras que si seleccionamos grabar al depósito, el paquete se grabará dentro de *Paquetes de depósito*, pasando la información al repositorio (Microsoft Repository).

Ejecución de un paquete

Una de las ventajas de grabar una operación de DTS en un paquete, es poder volverla a repetir sin tener que crear de nuevo los pasos para ejecutarla. En la situación sobre la que estamos trabajando, si por ejemplo, se elimina la tabla Zonas de la base de datos destino, podemos volver a ejecutar el paquete grabado haciendo clic con el botón derecho en el nombre del paquete, y seleccionando la opción de menú contextual *Ejecutar paquete*, lo que volverá a transferir la tabla entre las dos bases de datos.

Programación de un paquete

Puede darse el caso de que las operaciones grabadas en un paquete sea necesario repetirlas en un espacio de tiempo determinado. Para ello, es posible programar la ejecución del paquete, haciendo clic con el botón derecho sobre el mismo y seleccionando la opción del menú contextual *Programar paquete*, que nos mostrará la ventana de la Figura 228 en la que especificaremos el modo de programación que mejor se ajuste a nuestras necesidades.

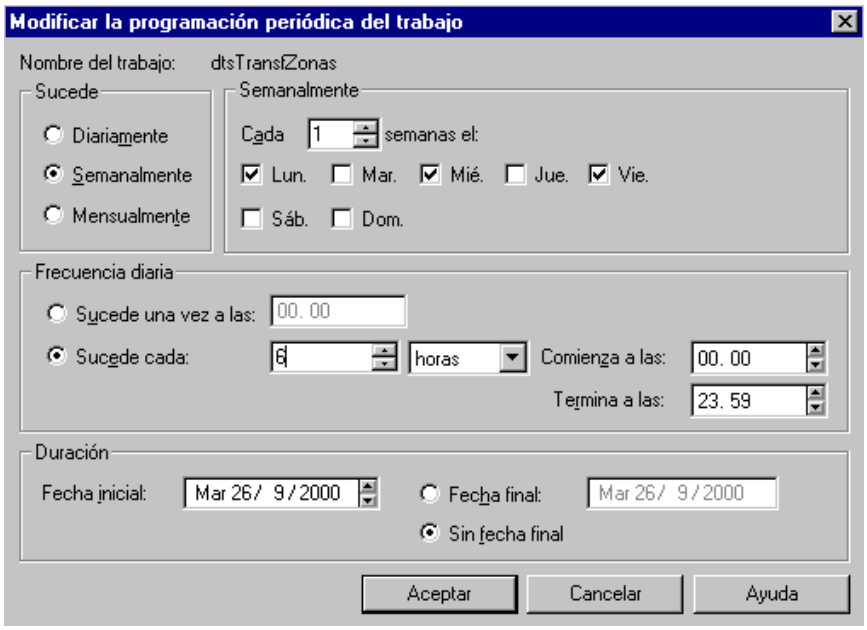


Figura 228. Ventana para la programación de un paquete DTS.

Establecidos los valores necesarios, pulsaremos Aceptar, ejecutándose el trabajo con el intervalo indicado.

Grabación de paquetes en el depósito

Durante el proceso de creación de un paquete con los servicios DTS, podemos indicar que dicho paquete se guarde en el depósito (Repository). Cuando grabamos un paquete en el depósito, tenemos la ventaja de que posteriormente podremos visualizar los metadatos que conforman ese paquete desde el elemento con el mismo nombre que se encuentra en la carpeta del DTS del servidor SQL Server. Estos metadatos serán comentados en el siguiente apartado.

Metadatos

Este elemento que encontramos en la carpeta *Servicios de transformación de datos*, del servidor SQL Server, nos permite obtener información acerca de la estructura interna de un origen de datos o de un paquete guardado en el depósito. La Figura 229 nos muestra el aspecto de este elemento.

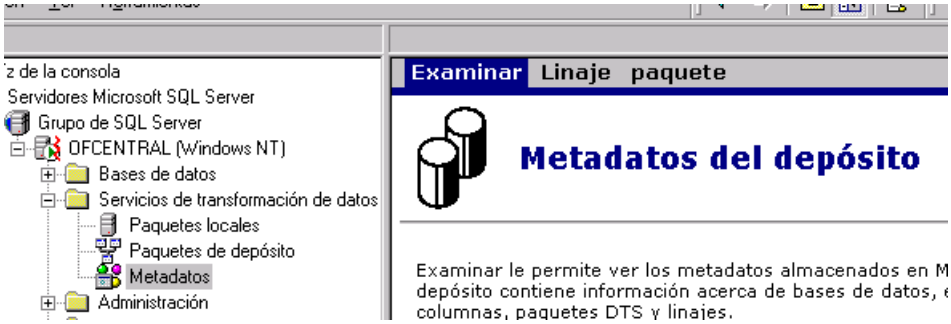


Figura 229. Contenido del elemento Medatados de la carpeta de DTS.

Metadatos de un origen de datos

Para obtener información sobre los metadatos de un origen de datos, haremos clic con el botón derecho en este elemento del DTS, seleccionando la opción *Importar metadatos* de su menú contextual, que nos mostrará una ventana en la que podremos seleccionar el origen de datos deseado, ver Figura 230. Dicho origen de datos no tiene que ser exclusivamente de SQL Server, sino que tenemos la ventaja de poder ver la estructura interna de cualquier fuente de datos de la lista disponible en esta ventana.

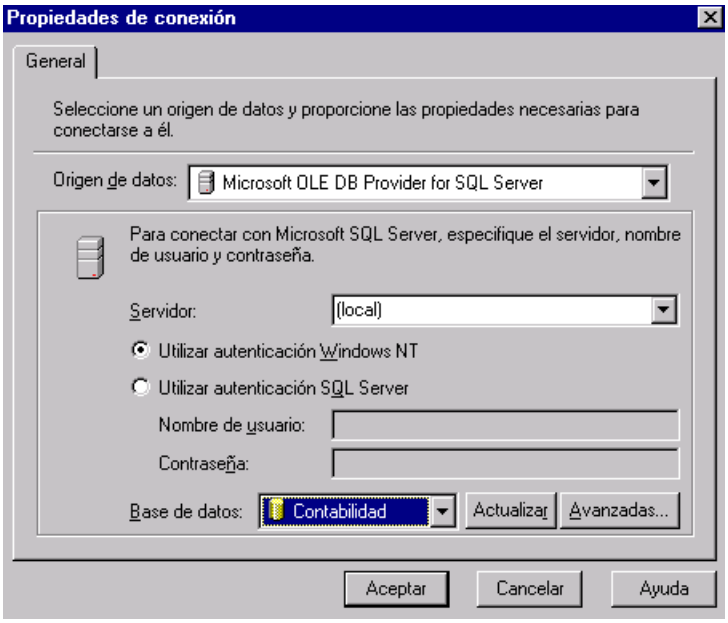


Figura 230. Selección de un origen de datos para obtener metadatos.

Una vez obtenidos los metadatos de un origen de datos, podremos visualizar su contenido haciendo clic en el elemento Metadatos del DTS, que nos mostrará en el panel derecho, los orígenes de datos importados, los cuales podremos recorrer desplegando su árbol de elementos, como muestra la Figura 231.

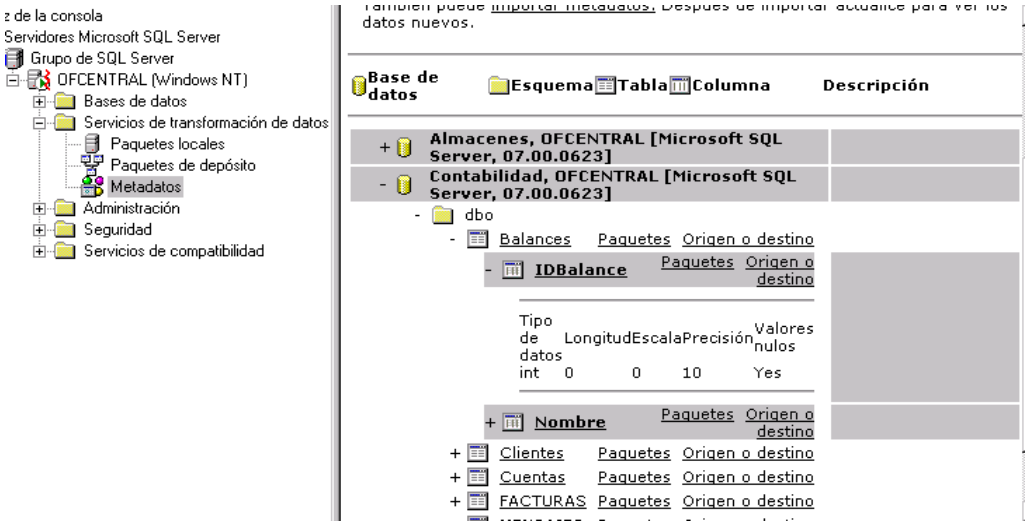


Figura 231. Árbol de metadatos para un origen de datos.

Metadatos de un paquete

Cuando situados en los metadatos del DTS, hacemos clic sobre la opción Paquete, en el panel derecho, se mostrarán todos los paquetes guardados en el depósito de los que podemos obtener sus metadatos (información interna), Figura 232. Para acceder desde este punto a los metadatos del paquete, haremos clic sobre el nombre de la versión, que nos mostrará la información de la Figura 233.

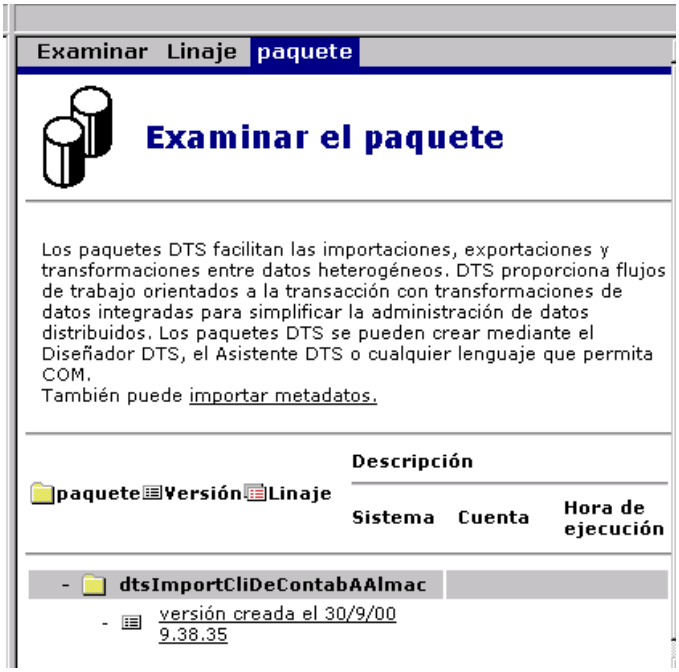


Figura 232. Metadatos de paquetes DTS grabados en el depósito.

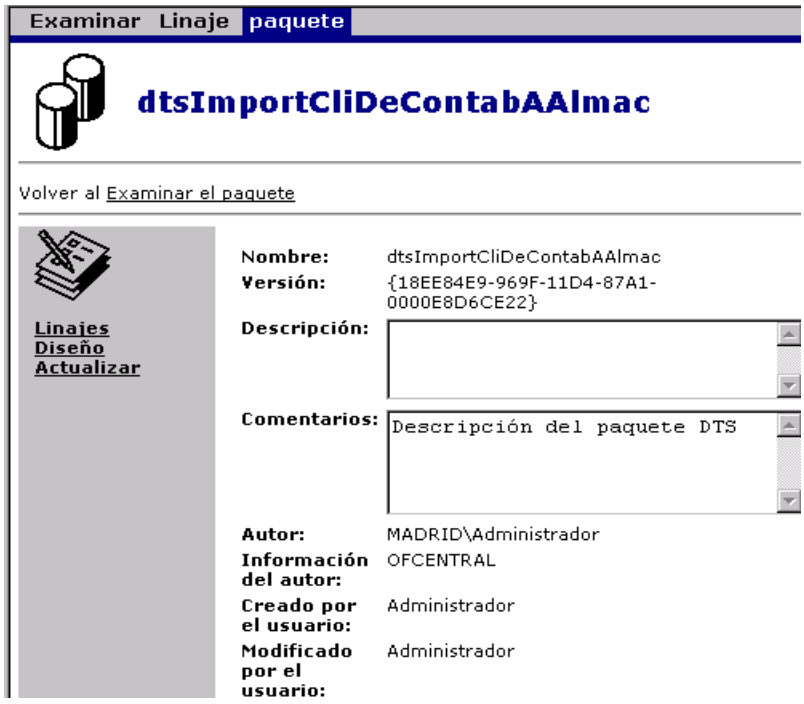


Figura 233

22

Diseño de paquetes DTS y tipos de transferencia

Diseñador DTS

Entre las diferentes utilidades y herramientas que los servicios DTS ponen a nuestra disposición para el trasvase de información entre fuentes de datos, se encuentra este diseñador, que nos permite de un modo gráfico, establecer los diferentes orígenes de datos entre los que realizar una transferencia y las operaciones o flujo de trabajo que se efectuarán entre dichos orígenes, grabando el resultado en un paquete DTS.

Podemos tomar un paquete ya existente y modificar su diseño, para ello haremos clic con el botón derecho sobre un paquete y seleccionaremos la opción de menú contextual *Diseñar paquete*, que abrirá el diseñador DTS cargando su contenido en la ventana de esta herramienta. La Figura 234 muestra el diseño del paquete dtsTransfZonas, creado en los apartados anteriores.

Como puede observar el lector, en esta ventana se muestran visualmente las operaciones de conexión y creación de tablas, representadas por las figuras con un nombre asociado al pie, y por otra parte, el flujo de trabajo entre dichos elementos, representado por las flechas que indican la dirección del flujo de trabajo.

Para agregar elementos al paquete, disponemos de las siguientes barras de herramientas en el lateral izquierdo de esta ventana.

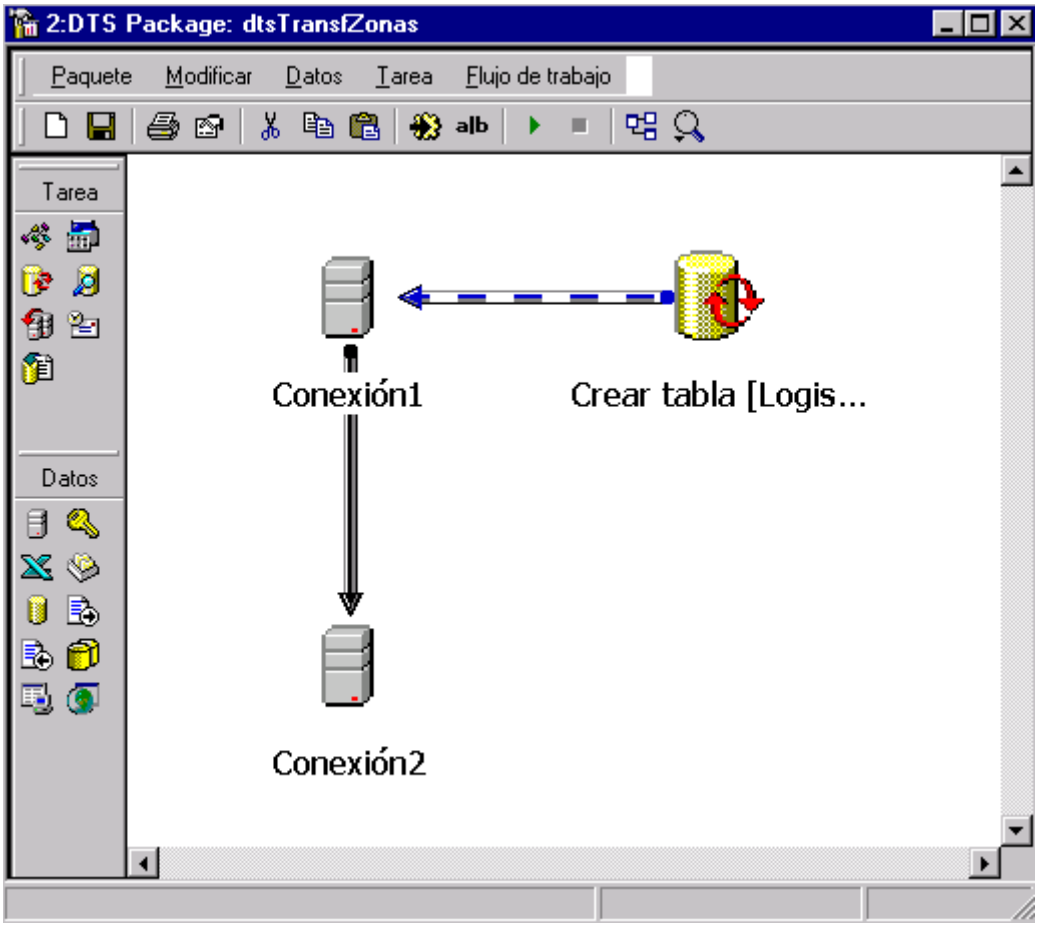


Figura 234. Diseñador de paquetes DTS.

- Datos. Cada uno de los tipos de fuente de datos con los que podemos establecer una conexión: SQL Server, Access, DBF, etc.

- Tarea. Cada una de las operaciones que se pueden ejecutar en un paquete: consulta SQL, inserción de datos, transferencia de objetos, etc.

Podemos también, utilizar las opciones de la barra de menús de esta ventana, tanto para añadir conexiones de datos y tareas al diseñador, como para establecer flujos de trabajo, grabar el paquete, etc., en definitiva, todas las operaciones disponibles en el diseñador.

Hemos de aclarar que durante las pruebas realizadas con esta ventana del diseñador, ha presentado cierto comportamiento inestable. Las barras de herramientas y menú en la parte superior no aparecían inicialmente, mostrándose cuando se pasaba el ratón sobre ellas; en otras ocasiones provocaba un error en la ejecución del Administrador corporativo, que obligaba a reiniciarlo, etc. Por este motivo, sugerimos al lector, que si se encuentra en una situación similar, acepte el mensaje de error del sistema y vuelva a ejecutar el Administrador corporativo, ya que este problema puede ser debido a un error interno del diseñador DTS.

Creación de un paquete con el diseñador DTS

En esta ocasión, en lugar de utilizar el asistente para transferencia de datos, vamos a crear manualmente esa operación de transferencia mediante el diseñador, grabándola al correspondiente paquete DTS.

Al igual que en el ejemplo anterior con el asistente, disponemos en la base de datos Almacenes de una tabla con el nombre Rutas, que deseamos exportar a la base de datos Logística.

En primer lugar hacemos clic con el botón derecho sobre la carpeta *Servicios de transformación de datos* de SQL Server, eligiendo la opción *Nuevo paquete* del menú contextual, que abrirá la ventana del diseñador de paquetes.

En la barra de herramientas Datos, pulsaremos sobre el botón correspondiente a la creación de una conexión con un origen de datos SQL Server, que nos mostrará el cuadro de diálogo de la Figura 235, en el que especificaremos el nombre de la conexión, la base de datos origen de la exportación, el servidor de datos al que pertenece, etc.

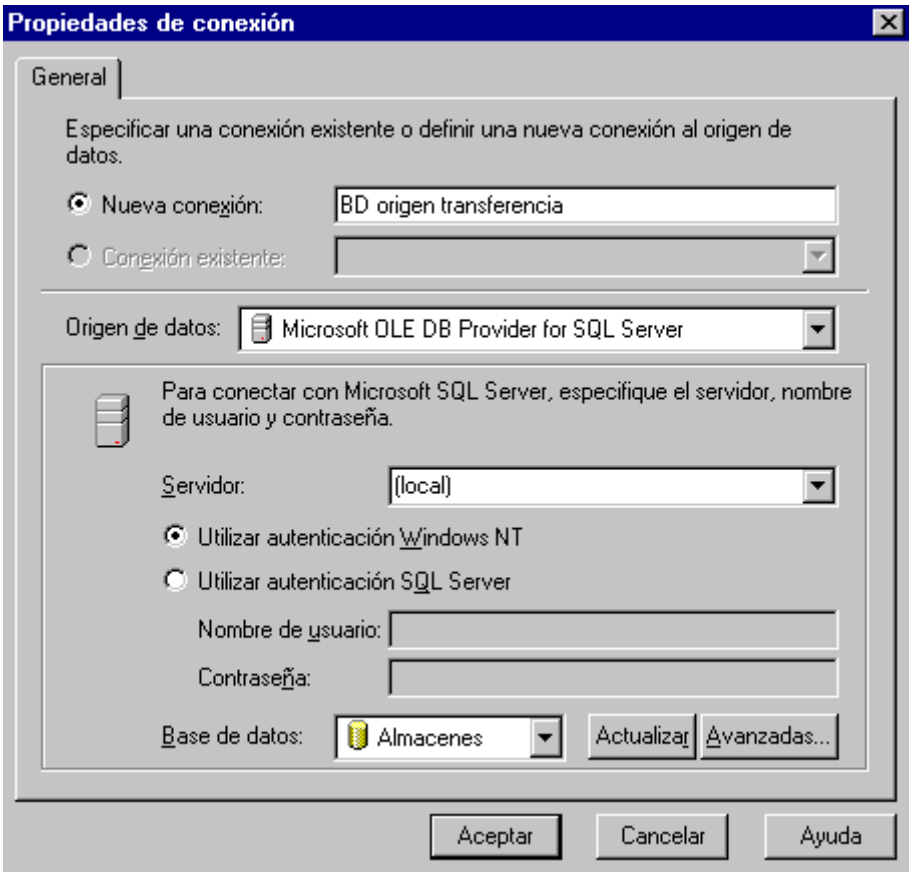


Figura 235. Creación del origen de datos en el diseñador DTS.

Seguidamente volvemos a pulsar el botón de creación de conexión con SQL Server, esta vez para indicar la base de datos destino del proceso de exportación: Logística, como muestra la Figura 236.

En la ventana del diseñador tendremos en este momento dos figuras representado a las conexiones establecidas. Mediante el teclado o ratón seleccionaremos las dos conexiones en el diseñador, de forma que la primera conexión seleccionada sea la correspondiente a la base de datos origen. A

continuación elegiremos la opción de menú *Flujo de trabajo+Agregar transformación*, que añadirá un flujo de trabajo al diseño en forma de flecha, indicando el sentido en el que se desarrollará el proceso, ver Figura 237.

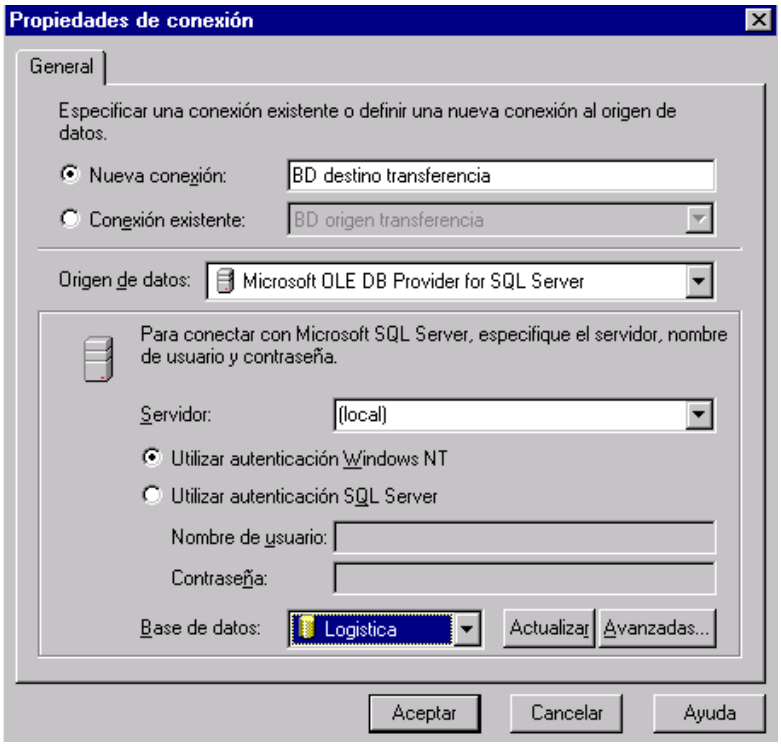


Figura 236. Creación del destino de datos en el diseñador DTS.

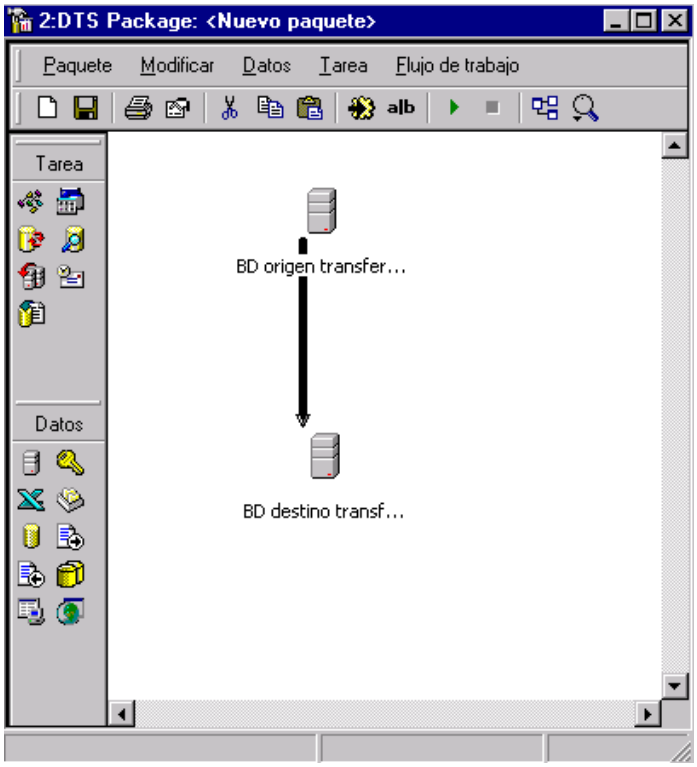


Figura 237. Creación del flujo de trabajo en el diseñador DTS.

En el siguiente paso debemos hacer clic con el botón derecho sobre la flecha correspondiente al flujo de trabajo, seleccionando de su menú contextual la opción *Propiedades*, que nos mostrará el cuadro de diálogo de la Figura 238, en el que tendremos que establecer los valores que debe ejecutar el proceso.

Para la pestaña Origen indicaremos la tabla de la base de datos de origen de la que vamos a tomar sus filas, o una sentencia SQL que tome los datos.

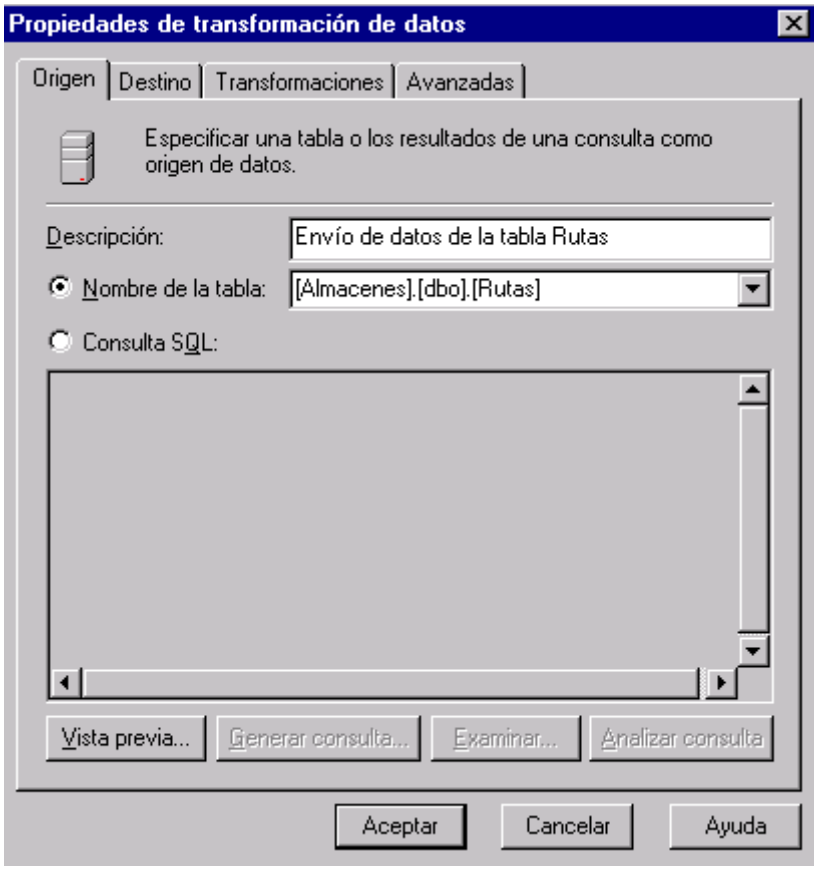


Figura 238. Flujo de trabajo del diseñador DTS, origen de los datos.

En la pestaña Destino indicaremos la base de datos que recibe la tabla. En este caso, como la tabla no existe en el destino, debemos especificar que tiene que crearse al mismo tiempo, pulsando el botón *Crear nueva*; al pulsar este botón, se mostrará en una ventana previa, la sentencia de creación, por si queremos modificarla.

Pulsando Aceptar se mostrará la estructura de la tabla que será creada en el flujo de trabajo, Figura 239.

La pestaña Transformaciones, muestra el proceso de trasvase de datos de origen a destino, permitiendo modificar la transformación o crear una nueva, ver Figura 240.

La pestaña Avanzadas muestra algunos valores que permiten controlar el número de errores producidos en el proceso de transformación, comprobaciones a efectuar por SQL Server, etc. Figura 241.

En este punto, podemos pulsar el botón Aceptar de la ventana de creación del flujo de trabajo para grabarlo en el diseñador.

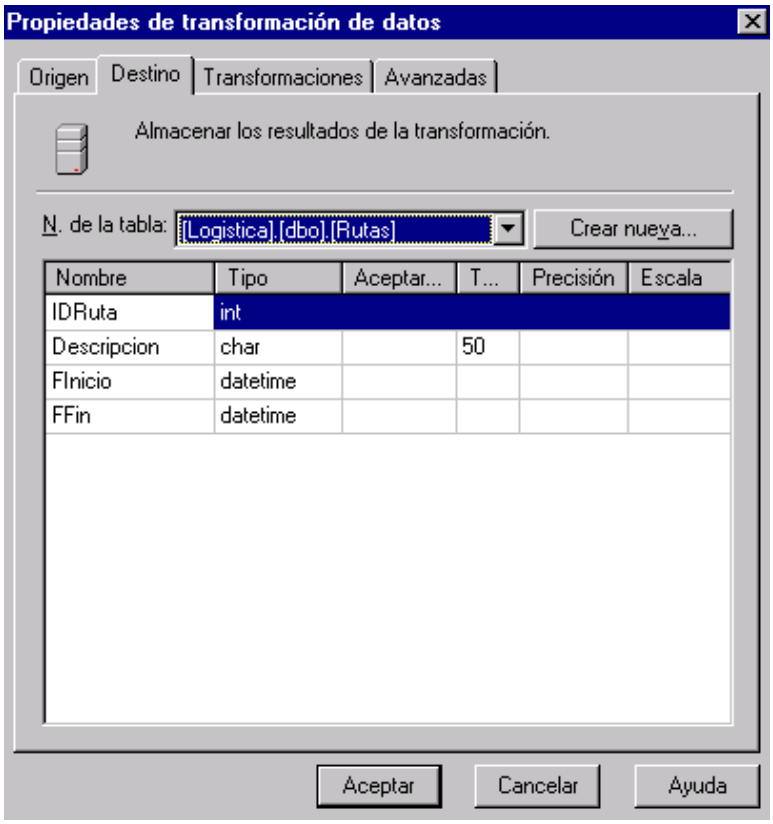


Figura 239. Flujo de trabajo del diseñador DTS, destino de los datos.

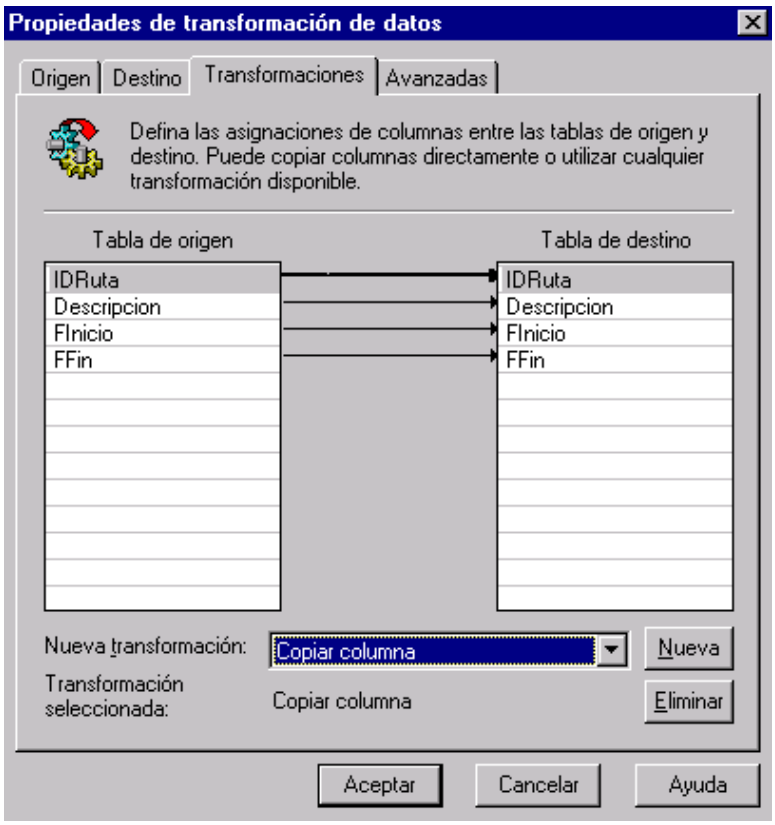


Figura 240. Flujo de trabajo del diseñador DTS, transformaciones de datos.

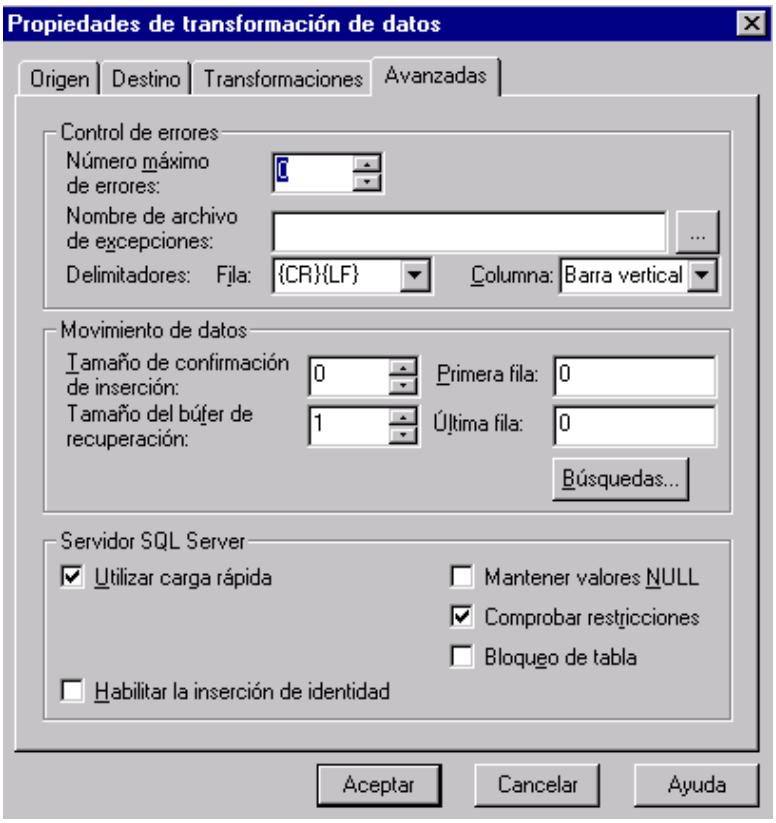


Figura 241. Flujo de trabajo del diseñador DTS, opciones avanzadas de transformación.

Finalizaremos la creación del paquete DTS seleccionando la opción de menú *Paquete+Guardar*, que nos mostrará una ventana en la que podremos asignar el nombre a este proceso de exportación de datos. Figura 242.

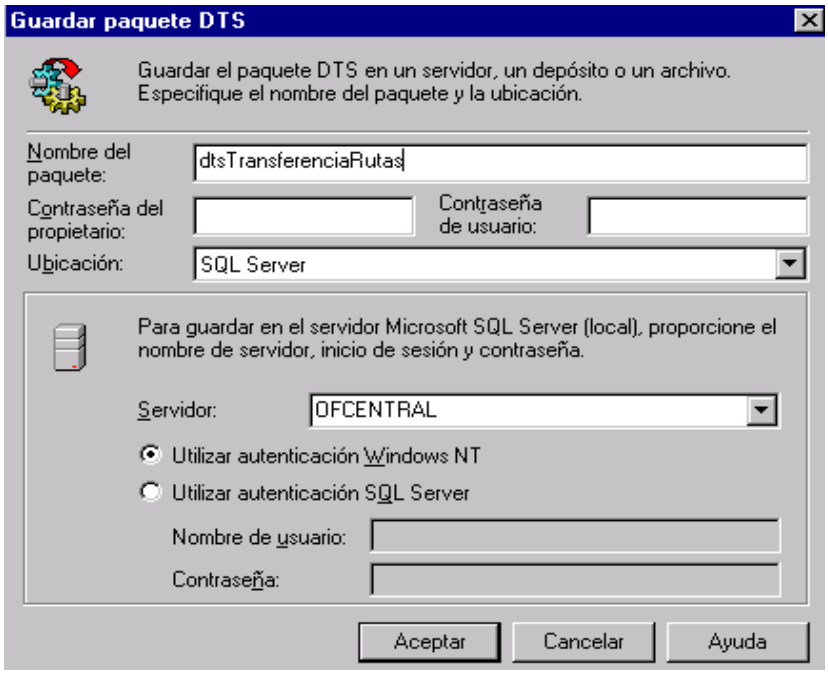


Figura 242. Grabación del paquete DTS.

Una vez creado el paquete DTS, podemos ejecutarlo directamente desde el diseñador, mediante la opción de menú *Paquete+Ejecutar*, o bien, si no tenemos abierto el diseñador, hacer clic con el botón derecho sobre el paquete y seleccionar la opción de menú contextual *Ejecutar paquete*. Ambos modos de ejecución pondrán en marcha el proceso de transferencia de datos indicado en el paquete, creando la tabla en la base de datos destino, y traspasando las filas a dicha tabla desde la base de datos origen.

Importar a una base de datos SQL Server un fichero DBF

En este caso vamos a realizar una importación desde un fichero DBF a una base de datos SQL Server, de forma que ilustremos como los servicios DTS se desenvuelven perfectamente trabajando entre fuentes de datos heterogéneas.

Realizaremos este proceso utilizando el asistente del DTS, obviando los pasos ya explicados en apartados anteriores.

Una vez que hemos iniciado este asistente, en el paso correspondiente a establecer el origen de datos, seleccionaremos dBaseIII (o la versión dBase correspondiente), y la carpeta del disco en donde reside el fichero DBF a importar. Debido a que cada fichero DBF representa una tabla, el nombre de la carpeta es interpretado por el servicio DTS como la base de datos. Ver Figura 243.

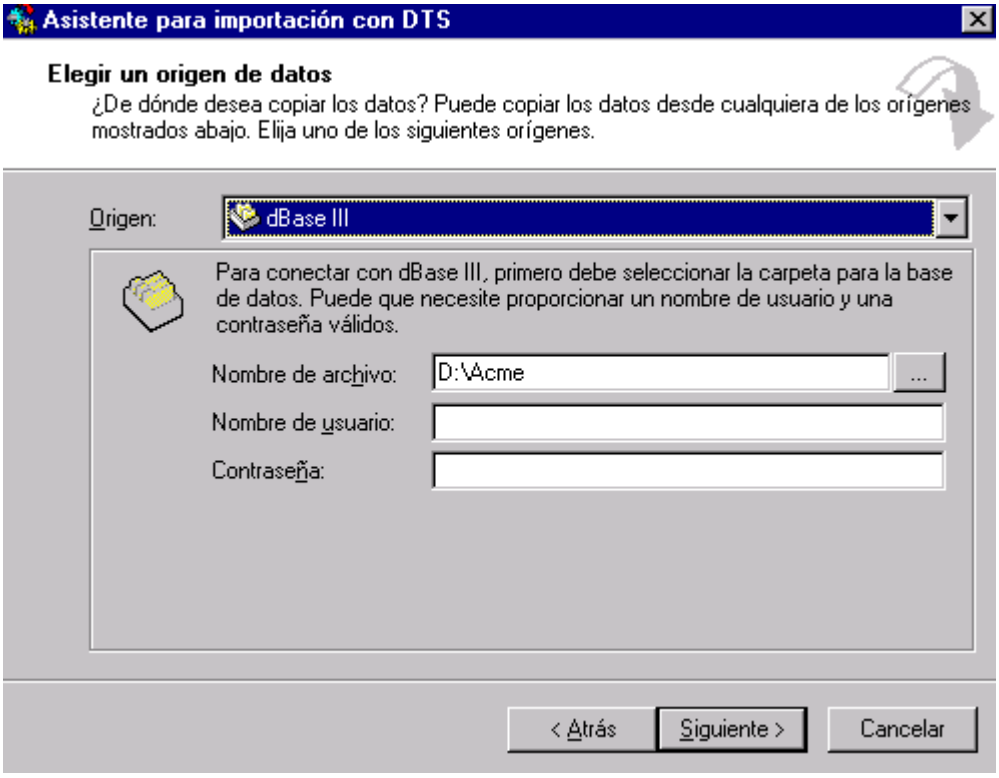


Figura 243. Especificar un origen de datos DBF en el asistente del DTS.

Si necesitamos transferir sólo un fichero DBF, el paso mostrado en la Figura 244 permite seleccionar qué ficheros dBaseIII exportar (mostrados como tablas, naturalmente), así como la tabla de destino en la base de datos SQL Server. El resto del proceso es igual al mostrado en el primer ejemplo del asistente DTS.

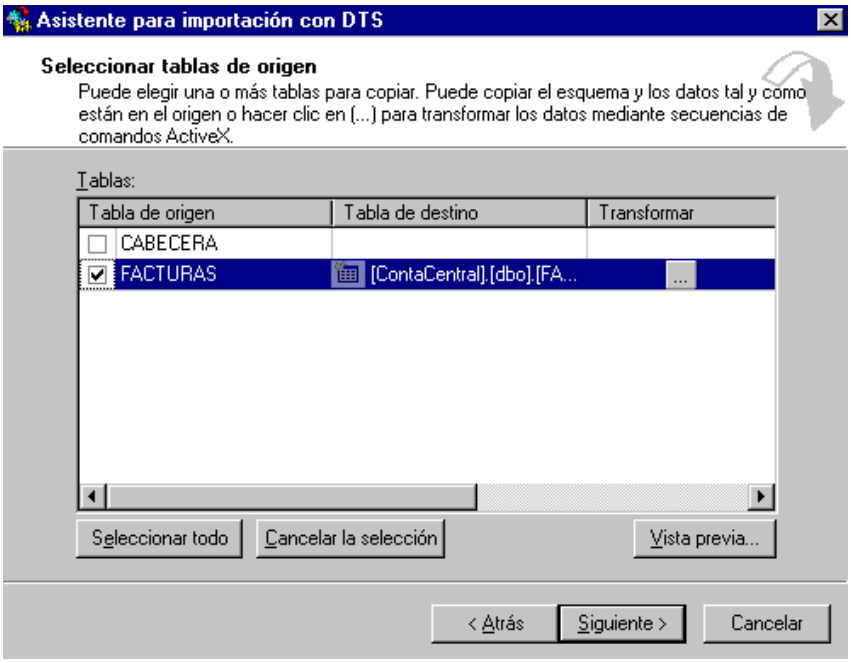


Figura 244. Especificación de ficheros DBF a exportar a SQL Server.

Importar una tabla desde Access con selección de filas

Además de transferir tablas completas, DTS nos permite mediante una consulta de selección de filas, copiar sólo los datos que necesitemos de una fuente de datos a otra. En el ejemplo de este apartado, vamos a realizar una selección de filas de una tabla de una base de datos Access, copiándolas a otra tabla que reside en una base de datos SQL Server. En el asistente del DTS daremos los pasos habituales de selección de origen y destino de datos. Al llegar al paso en que debemos indicar qué tipo de copia realizar, especificaremos que vamos a usar una consulta, como muestra la Figura 245.

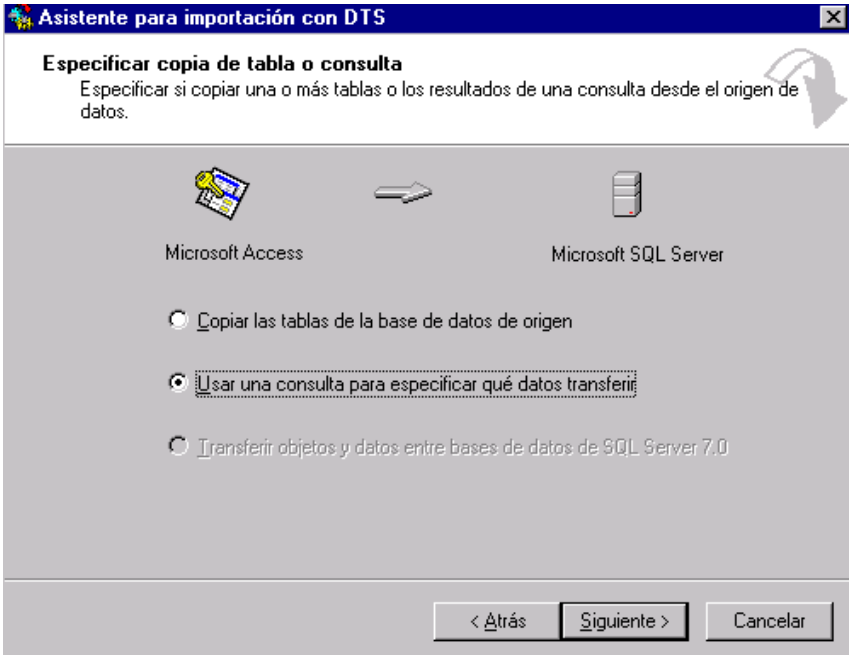


Figura 245. Asistente del DTS indicando que vamos a realizar una transferencia con una consulta.

En el siguiente paso, escribiremos una consulta SQL de selección de datos, o bien utilizaremos el generador de consultas accesible en el botón *Generador de consultas*, Figura 246.

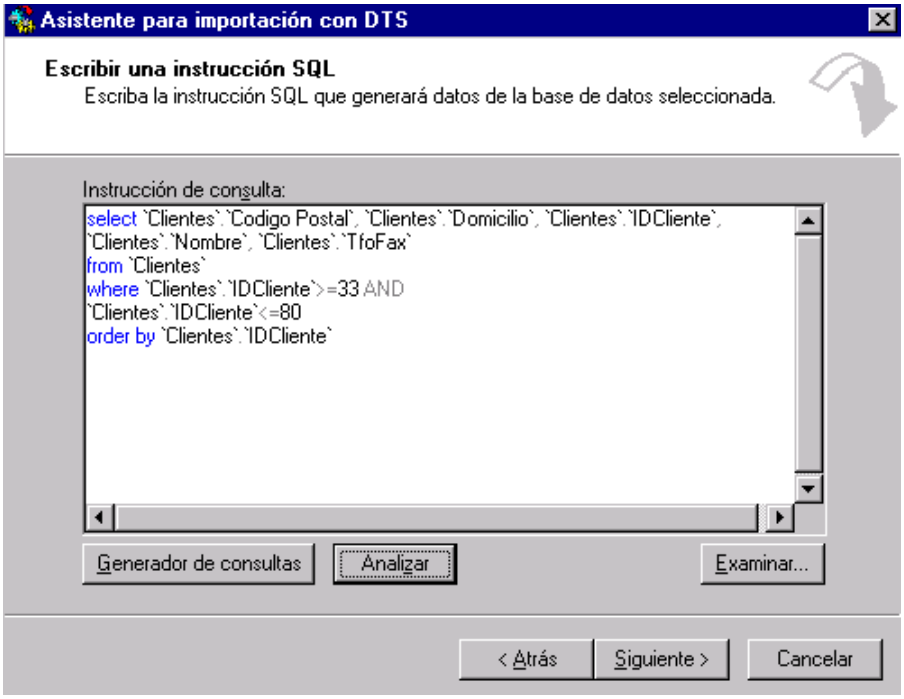


Figura 246. Consulta de selección de datos a transferir.

A continuación, si la tabla no existe, se indicará que se va a realizar la importación sobre una tabla a la que por defecto se le da el nombre Resultados. En este paso modificaremos si lo consideramos oportuno el nombre de la tabla que se va a crear en la base de datos SQL Server destino. Figura 247.

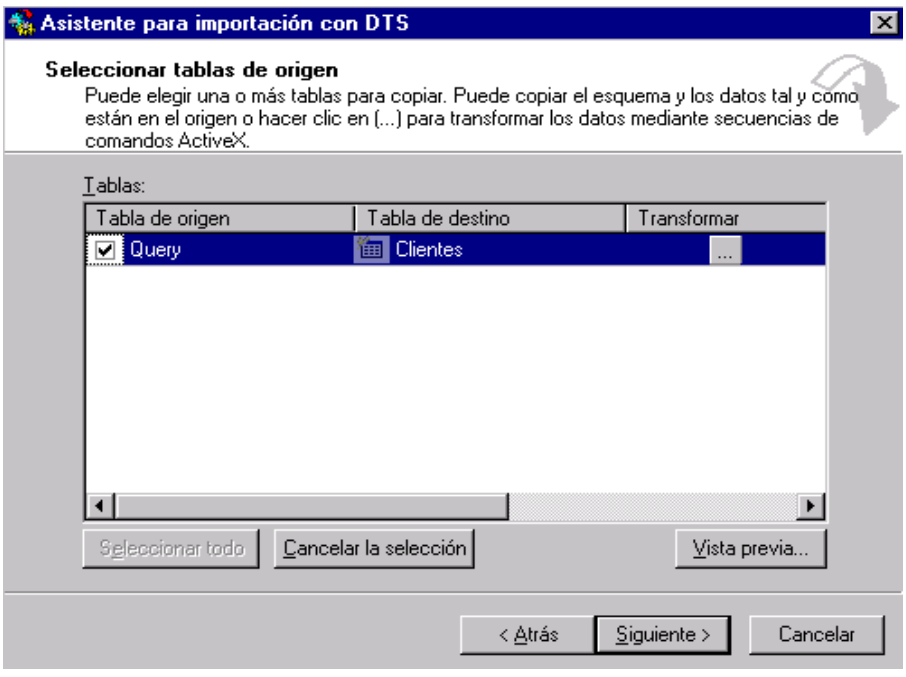


Figura 247. Especificar tabla a la que se copiarán los datos.

El resto de los pasos hasta completar el asistente serán los ya explicados al lector.

Proporcionar consistencia a los datos al mismo tiempo que se transfieren

Puede darse el caso de que necesitemos importar desde una base de datos, tablas o filas en las cuales la información no guarda la adecuada coherencia, por ejemplo, una tabla de mensajes a clientes en la que a un mismo cliente se le denomina de formas diversas: Repuestos Mecanizados S.A., Rep.Mecaniz.SA y R.Mecanizados.

Está claro que si importamos esta tabla, previamente deberíamos realizar un proceso de adecuación de los datos, de modo que en el caso comentado, todas las filas con el nombre del mencionado cliente se copiaran de igual forma.

Pues bien, los servicios de DTS no pueden realizar esta tarea directamente, pero sí nos permiten, conociendo algún lenguaje de script de los más habituales: Visual Basic Script, JScript,etc., escribir el código necesario para comprobar los datos en el momento previo del traspaso, realizando los cambios necesarios para que la información se transfiera con la mayor coherencia posible.

Partiendo de la situación antes planteada, disponemos en una base de datos Access, de una tabla de mensajes con nombres de un mismo cliente escritos de forma distinta. Iniciamos desde SQL Server el asistente de importación DTS para traer los datos de dicha tabla a una base de datos de SQL Server. En el paso correspondiente a la selección de las tablas de la base de datos origen, una vez marcada la tabla de mensajes, pulsamos el botón de la columna Transformar, como se muestra en la Figura 248.

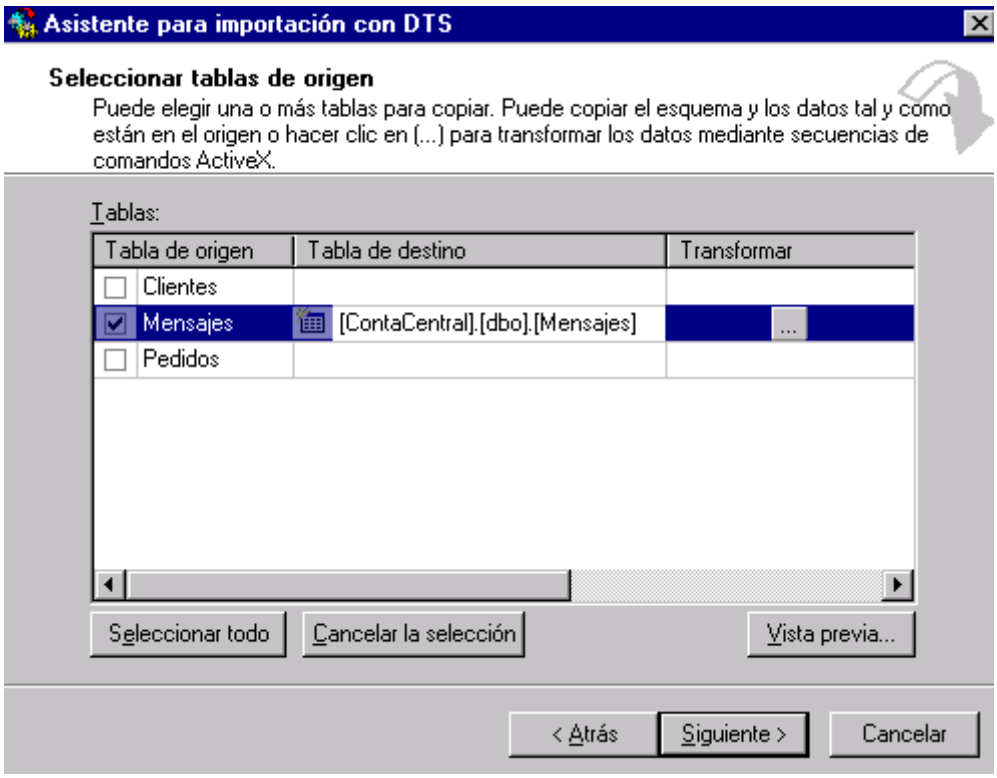


Figura 248. Selección de tablas de origen.

Una vez pulsado dicho botón, aparecerá la ventana *Asignaciones y transformaciones de columnas*, que nos permitirá modificar la información que va a ser transferida a SQL Server. En la pestaña *Asignaciones de columnas*, estableceremos si se debe crear la tabla para el caso de que no exista en la base de datos SQL Server de destino, pudiendo también modificar la estructura de la tabla a crear, Figura 249.

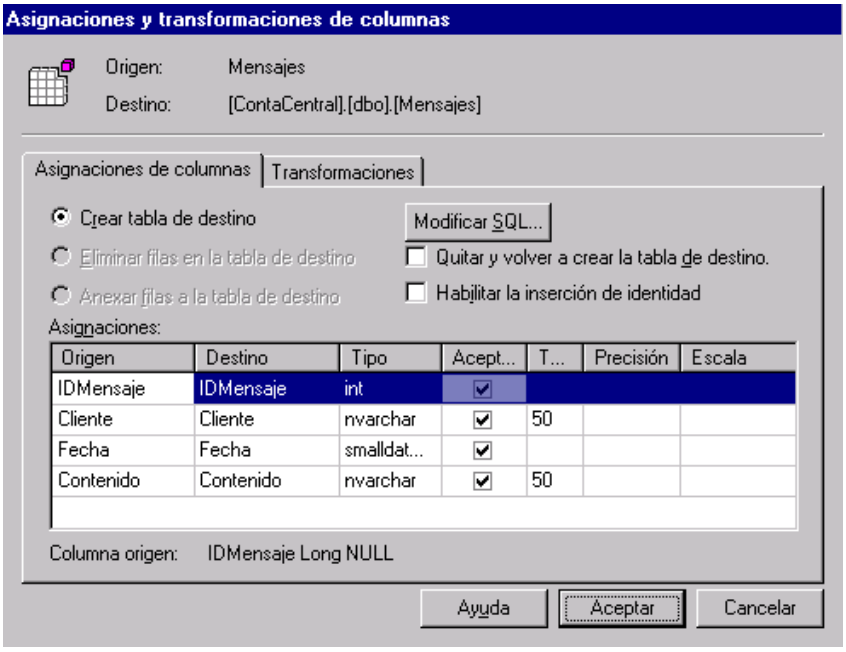


Figura 249. Estructura de la tabla a crear en el asistente DTS.

Pero es la pestaña Transformaciones, Figura 250, la que nos interesa en este caso, puesto que allí es donde se muestra la secuencia de código en Visual Basic Script, con las instrucciones de traspaso de datos por defecto.

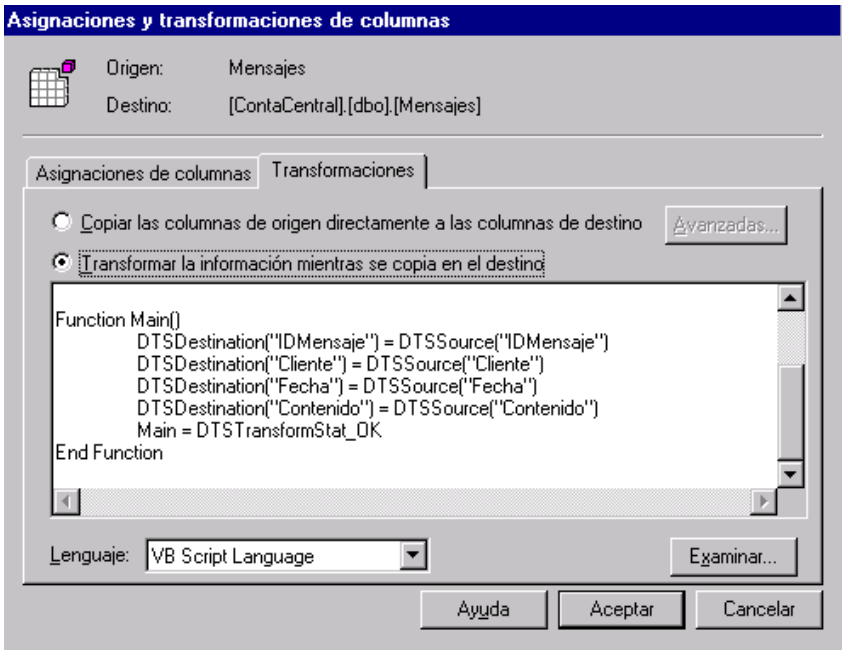


Figura 250. Código fuente VB Script de transformación de datos en el asistente DTS.

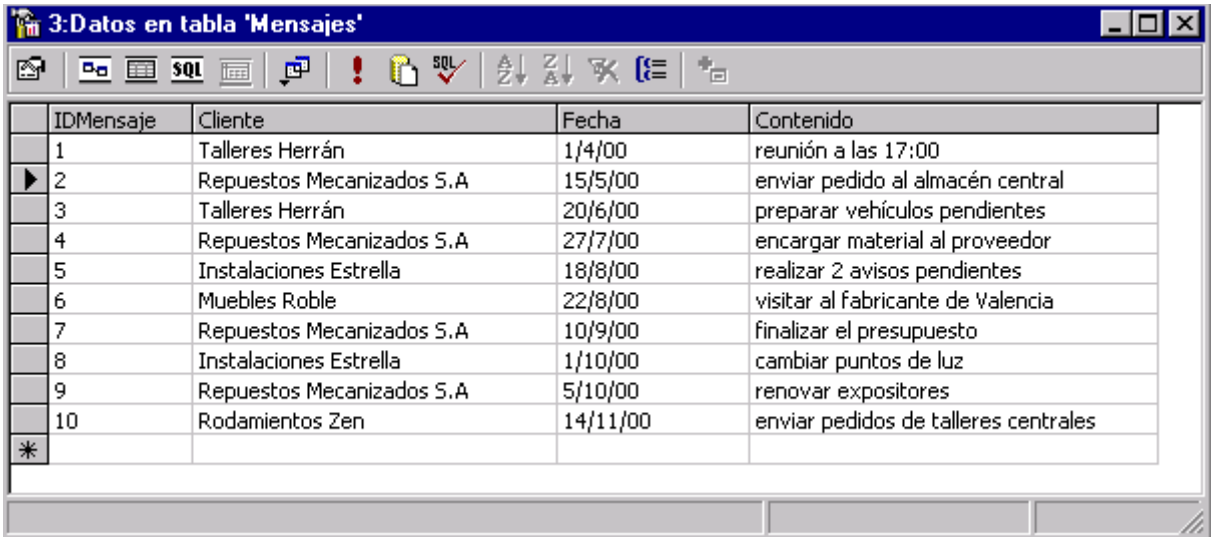
Para conseguir que todas las filas con el nombre de cliente mencionado anteriormente, al importarlas desde Access, se graben igual en la tabla de SQL Server, modificaremos estas instrucciones según se muestra en el Código fuente 77.

```
Function Main()
   DTSDestination("IDMensaje") = DTSSource("IDMensaje")

   If DTSSource("Cliente") = "R.Mecanizados" Or _
      DTSSource("Cliente") = "Rep.Mecaniz.SA" Then

      DTSDestination("Cliente") = "Repuestos Mecanizados S.A"

   Else
      DTSDestination("Cliente") = DTSSource("Cliente")

   End If

   DTSDestination("Fecha") = DTSSource("Fecha")
   DTSDestination("Contenido") = DTSSource("Contenido")

   Main = DTSTransformStat_OK
End Function
```

Código fuente 77

Aunque el código es muy simple, sirva como aclaración al lector que desconozca la sintaxis de Visual Basic, que las funciones DTSDestination("NombreCampo") y DTSSource("NombreCampo"), se utilizan para asignar y obtener respectivamente el valor de un campo en el proceso de transferencia de datos del DTS. En el caso del campo Cliente, si el valor del campo no corresponde al que va a ser el oficial para todas las filas de la tabla, asignamos el valor que queremos directamente a DTSDestination().

El resto de pasos en el asistente son los ya conocidos, por lo que una vez completado, se realizará la importación de datos, normalizando el nombre del cliente mencionado. La Figura 251 muestra una vista de la tabla resultante, una vez transferida a SQL Server.



IDMensaje	Cliente	Fecha	Contenido
1	Talleres Herrán	1/4/00	reunión a las 17:00
2	Repuestos Mecanizados S.A	15/5/00	enviar pedido al almacén central
3	Talleres Herrán	20/6/00	preparar vehículos pendientes
4	Repuestos Mecanizados S.A	27/7/00	encargar material al proveedor
5	Instalaciones Estrella	18/8/00	realizar 2 avisos pendientes
6	Muebles Roble	22/8/00	visitar al fabricante de Valencia
7	Repuestos Mecanizados S.A	10/9/00	finalizar el presupuesto
8	Instalaciones Estrella	1/10/00	cambiar puntos de luz
9	Repuestos Mecanizados S.A	5/10/00	renovar expositores
10	Rodamientos Zen	14/11/00	enviar pedidos de talleres centrales

Figura 251. Tabla resultante de la importación con modificación de datos mediante código.

Transferencia de objetos entre bases de datos SQL Server

El traspaso de objetos tales como procedimientos almacenados, sólo puede ser realizado entre bases de datos SQL Server, ya que debido a la naturaleza de otros motores de datos, es posible que los objetos disponibles en SQL Server no existan como tales en otros orígenes de datos diferentes.

Para exportar uno o varios procedimientos almacenados entre bases de datos, durante el proceso de transferencia del asistente DTS, al llegar al paso en el que debemos indicar qué elementos copiar, pulsaremos el botón de opción *Transferir objetos y datos entre bases de datos SQL Server 7.0*, véase la Figura 252.

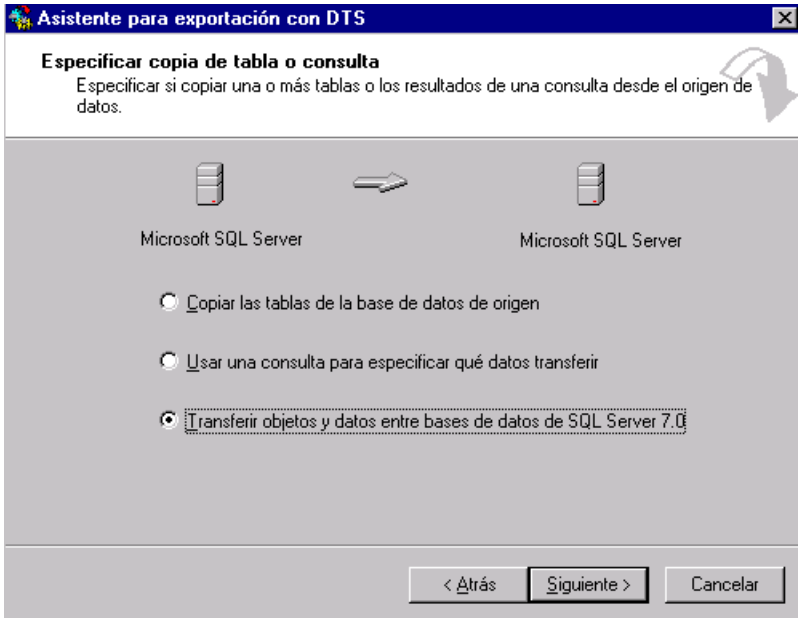


Figura 252. Especificar transferencia de objetos en el asistente DTS.

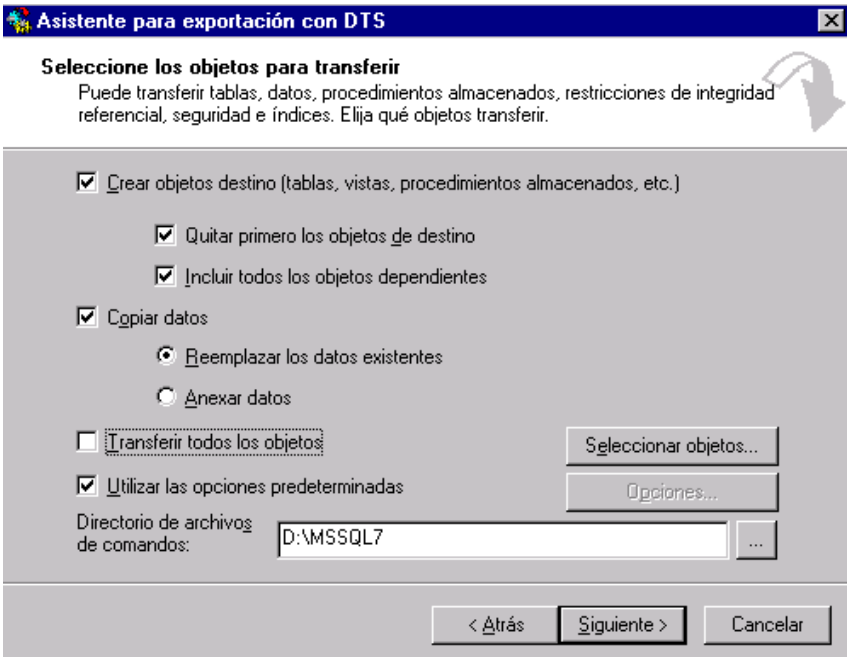


Figura 253. Modo de selección de objetos en el asistente DTS.

En el siguiente paso estableceremos el modo de selección de los objetos y datos, pudiendo especificar que vamos a seleccionar sólo unos objetos determinados o todos, como se muestra en la Figura 253.

En este mismo paso, pulsaremos el botón *Seleccionar objetos*, que nos mostrará el cuadro de diálogo de la Figura 254 en el que podremos realizar dicha selección. En este caso vamos a transferir un procedimiento almacenado entre bases de datos.

Después de aceptar esta ventana, continuaremos con los siguientes pasos del asistente DTS hasta completarlo. El resultado será la copia del procedimiento almacenado en la base de datos destino.

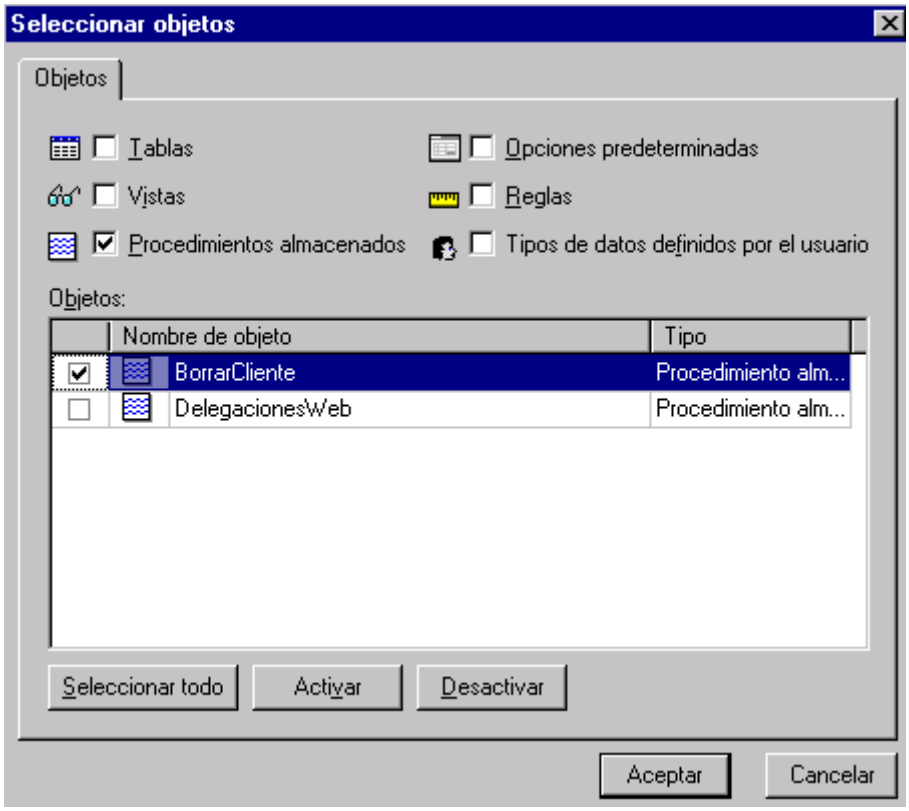


Figura 254.

23

Mover una base de datos

Cambiar la ubicación física de una base de datos

En nuestro trabajo cotidiano con SQL Server, podemos encontrarnos ante situaciones como tener que utilizar una base de datos en otro servidor distinto del actual, o que el disco físico que alberga la base de datos se quede sin espacio.

Ante estos problemas, una posible solución sería, en el primer caso, crear una nueva base de datos en el servidor SQL Server destino, con la misma estructura que tiene en el servidor original, y traspasar la información entre las bases de datos mediante una copia de seguridad. En el segundo caso, podemos añadir un fichero para la base de datos en el nuevo disco, donde se grabaría la nueva información.

Sin embargo, estos son recursos ya conocidos, y en este tema vamos a proponer una alternativa, que a pesar de no emplear ningún asistente ni utilidades gráficas sofisticadas, en ocasiones puede resultar más útil, debido precisamente a eso, su simplicidad. Nos referimos a la técnica de separar y adjuntar una base de datos a un servidor SQL Server.

Características generales de separar y adjuntar bases de datos

La información de una base de datos SQL Server se guarda en una serie de ficheros con extensión .MDF, .LDF y .NDF, generalmente en el directorio Data de la instalación de SQL Server. Pero el mero hecho de copiar o borrar estos ficheros de tal ubicación no presupone que el motor de datos “se percate” de tal situación.

Para quitar una base de datos del servidor sin eliminarla, debemos en primer lugar separarla, y si además, queremos adjuntar dicha base de datos a otra instalación de SQL Server, tendremos que copiar los ficheros de datos que la componen en el directorio Data de SQL Server o en el que corresponda, y anexarlos al servidor de datos para que sean reconocidos.

Puede que parezca esta una tarea muy primitiva, al obligar a manipular los ficheros de una base de datos, pero estamos seguros de que en ciertas situaciones, resultará más efectiva que el empleo de sofisticadas utilidades gráficas de gestión de datos.

Un escenario de aplicación

Como ejemplo, vamos a proponer la siguiente situación: una empresa dispone en una de sus oficinas de un servidor SQL Server en el que se encuentra la base de datos Facturas. La información de esta base de datos se encuentra en los ficheros FACTURAS_DATOS.MDF. y FACTURAS_REGISTRO .LDF. Para acceder a dicha base de datos, hay definido en el servidor de datos un inicio de sesión con autenticación SQL Server y con el nombre Facturador.

Por motivos de reestructuración en los departamentos, se debe cambiar la ubicación de dicha base de datos a un servidor SQL Server que reside en otra oficina.

Para realizar el traslado de la mencionada base de datos entre los servidores, emplearemos la técnica comentada de separar y adjuntar bases de datos, cuyo proceso será descrito en los sucesivos apartados.

Separar una base de datos de SQL Server

Para esta operación, emplearemos el procedimiento almacenado del sistema sp_detach_db, que tiene la siguiente sintaxis.

```
sp_detach_db [@dbname =] 'NombreBD' [, [@skipchecks =]
'ExcluirComprobaciones']
```

- NombreBD. Nombre de la base de datos a separar del servidor.
- ExcluirComprobaciones. Valor lógico que permite realizar las comprobaciones sobre las tablas con UPDATE STATISTICS. Si el valor es TRUE no se realizan las comprobaciones, cuando el valor es FALSE sí se realizan.

Para separar la base de datos Facturas, ejecutaremos desde el Analizador de consultas el Código fuente 78.

```
EXEC sp_detach_db @dbname='Facturas'
```

Código fuente 78

Podemos comprobar que la base de datos ya no pertenece al servidor, puesto que si intentamos acceder a ella desde el Administrador corporativo u otra utilidad, ya no estará accesible.

Adjuntar una base de datos a SQL Server

Una vez separada la base de datos del servidor, emplearemos el soporte más adecuado para transportar sus ficheros hasta el servidor destino: disquetes, cintas, discos removibles, etc. En dicho servidor, copiaremos los ficheros en el directorio que contenga las bases de datos de SQL Server, y desde el Analizador de consultas, situados en la base de datos master, ejecutaremos el procedimiento almacenado del sistema sp_attach_db, que se encarga de adjuntar una base de datos a SQL Server.

```
sp_attach_db [@dbname =] 'NombreBD', [@filename1 =] 'FicheroBD1'
[,…FicheroBD16]
```

- NombreBD. Nombre de la base de datos que se va a adjuntar al servidor.
- FicheroBDn. Nombre de fichero que corresponde a la base de datos. Podemos emplear hasta 16 ficheros, ya que una base de datos estará repartida al menos en dos ficheros: datos y registro.

El Código fuente 79 adjunta la base de datos Facturas al nuevo servidor.

```
sp_attach_db 'Facturas','d:\mssql7\data\Facturas_datos.mdf',
'd:\mssql7\data\Facturas_registro.ldf'
```

Código fuente 79

Crear el inicio de sesión de la base de datos

Puesto que la base de datos tenía un usuario específico con el nombre Facturador, deberemos crear también en el servidor SQL Server destino un usuario con el mismo nombre, ya que en este momento, la base de datos tendrá lo que se denomina un usuario huérfano, al no existir el inicio de sesión Facturador en el servidor SQL Server destino, y el único modo de acceder a los datos será a través de un usuario con privilegios de administrador.

Para solucionar esta situación, crearemos un nuevo inicio de sesión con el nombre Facturador, modo de autenticación SQL Server y contraseña en el caso de que sea necesaria. Como base de datos predeterminada, de momento dejaremos master. Ver Figura 255.

A continuación debemos vincular el nuevo inicio de sesión con la base de datos anexionada, esto lo conseguiremos ejecutando desde el Analizador de consultas, y situados sobre la base de datos Facturas, el procedimiento almacenado del sistema sp_change_users_login, que tiene la siguiente sintaxis.

```
sp_change_users_login [@Action =] 'TipoAcción' [,[@UserNamePattern
=] 'Usuario'][,[@LoginName =] 'InicioSesión']
```

- TipoAcción. Cadena que define la operación que va a realizar el procedimiento almacenado. Los valores disponibles son los siguientes:
  - Auto_fix. Vincula a los usuarios que hay en la tabla sysusers, pertenecientes a la base de datos actual, con los inicios de sesión del mismo nombre que están en syslogins. Al utilizar este valor, el parámetro Usuario debe ser un usuario válido para la base de

datos actual, y el parámetro InicioSesión deberá ser NULL, una cadena vacía o no utilizarse.

- Report. Devuelve los usuarios e identificadores de seguridad de la base de datos actual. En este caso, tanto el parámetro Usuario como InicioSesión deberán ser NULL, una cadena vacía o no utilizarse.

- Update_one. Vincula a Usuario con InicioSesión.

- Usuario. Nombre de usuario de la base de datos en la que estamos posicionados. Este procedimiento almacenado sólo es válido para usuarios de SQL Server, no pudiendo utilizarse con usuarios de Windows NT.

- InicioSesión. Nombre del inicio de sesión de SQL Server que vamos a vincular con el usuario de la base de datos.

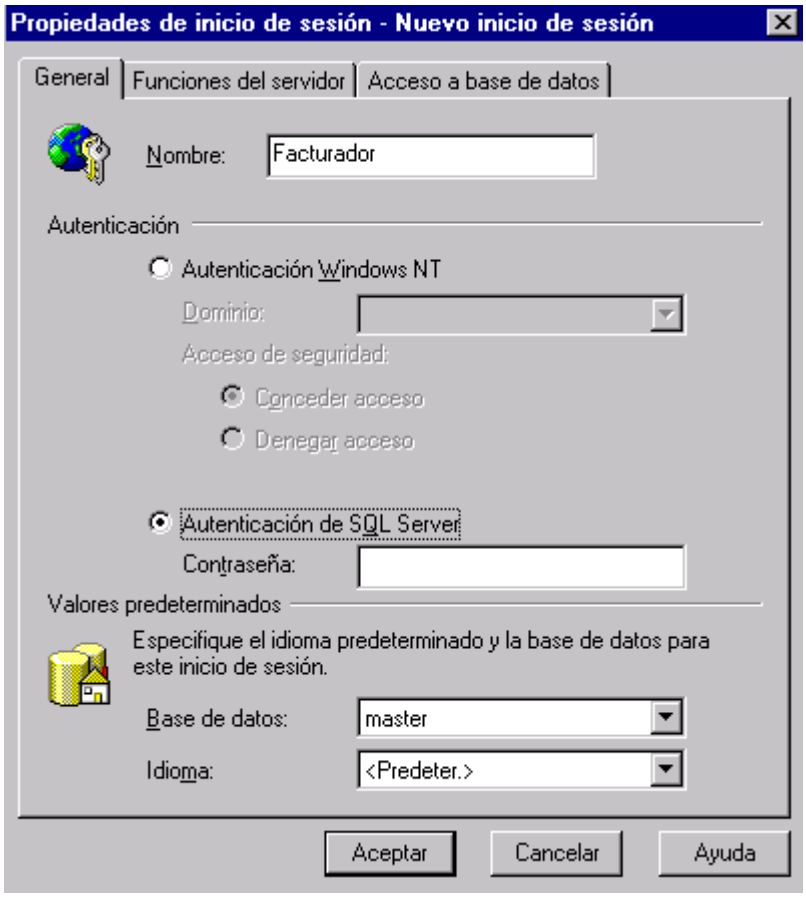


Figura 255. Creación del usuario para la base de datos a adjuntar.

El Código fuente 80 nos muestra como se lleva a cabo la vinculación del usuario Facturador, de la base de datos Facturas, con el inicio de sesión Facturador, de SQL Server.

```
sp_change_users_login
@action='update_one',@usernamepattern='Facturador',@loginname='Facturador'
```

Código fuente 80

Como paso final, debemos abrir la ventana de propiedades del inicio de sesión Facturador, y cambiar su base de datos por defecto a Facturas, como vemos en la Figura 256.

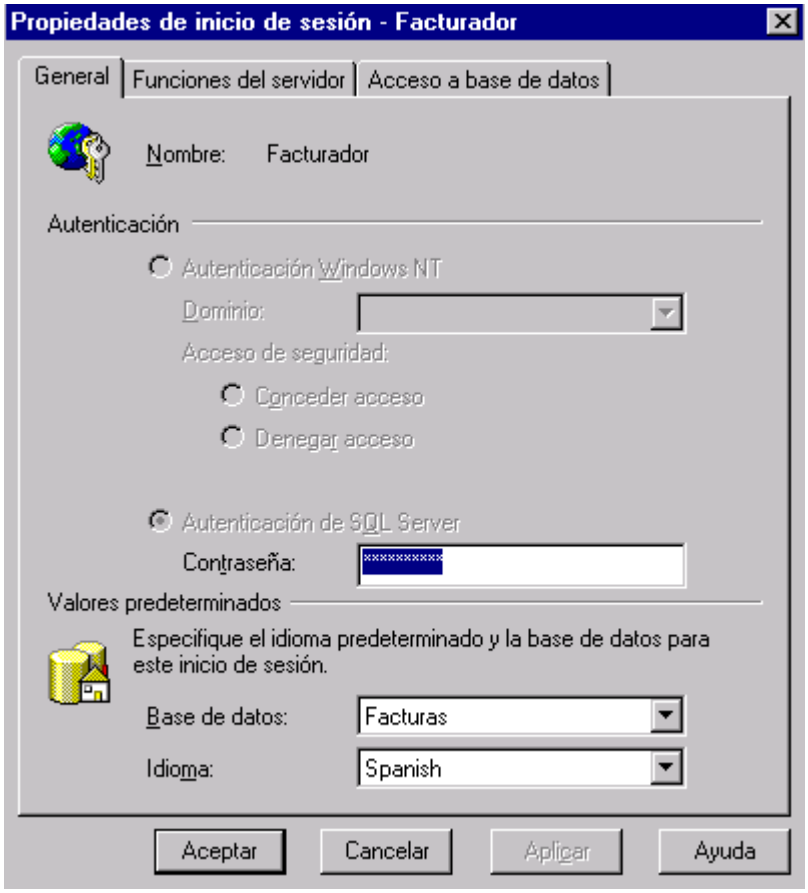


Figura 256. Cambiar base de datos por defecto para el inicio de sesión.

Con esta acción, el inicio de sesión Facturador podrá acceder a las tablas de esta base de datos.

24

Operaciones de mantenimiento y control

Motivos para establecer trabajos de revisión

Un servidor SQL Server no es un elemento del sistema que una vez instalado se automantenga. Si bien es cierto que posee un alto nivel de autoconfiguración y capacidad de resolver problemas, también es cierto que regularmente, el administrador de SQL Server debe realizar un conjunto de tareas de mantenimiento que aseguren un nivel de funcionamiento óptimo.

Entre las tareas de supervisión a realizar por el administrador, podríamos citar las siguientes:

- Servidores SQL Server. Evidentemente, por lo primero que debe preocuparse el administrador de datos es por la buena marcha del servidor o servidores SQL Server instalados en el sistema, vigilando el tamaño del registro de transacciones, bloqueos y toda actividad que pueda llegar a consumir excesivos recursos, perjudicando al resto de elementos del servidor.

- Programas de usuario. Un programa que haga uso de la información situada en las bases de datos de SQL Server, puede afectar negativamente al funcionamiento conjunto si efectúa un elevado nivel de operaciones contra una base de datos, bloqueos prolongados, etc. Se debe vigilar el diseño de las aplicaciones que trabajen contra la información del servidor de datos SQL Server en lo referente a todos estos aspectos: consultas, transacciones, cursores de navegación por las consultas, etc.

- Diseño de red. Todos los aspectos relacionados con el tráfico de red por la que circulan los datos de SQL Server: velocidad, ancho de banda, nivel de actividad, etc., afectan al rendimiento de los datos, por lo que deben ser tenidos en cuenta a la hora de implantar un sistema de información basado en este motor de datos. De igual forma, si el sistema operativo

de red es Windows NT, tanto el administrador del sistema como el de SQL Server, deben coordinarse de forma que la configuración del sistema operativo sea la más adecuada en cuanto a gestión de discos, servicios del sistema, archivos de paginación, etc.

- Hardware. También es importante comprobar la cantidad de memoria instalada en el servidor, número de procesadores y tipo, velocidad de los discos, etc.

Herramientas para mantenimiento y revisión

SQL Server pone a disposición del administrador un conjunto de herramientas, formado por aplicaciones, asistentes e instrucciones que le permiten realizar un seguimiento sobre determinados escenarios de ejecución y problemas de rendimiento, con el fin de localizar el origen de los fallos que se puedan producir en el sistema.

En los siguientes apartados se realiza una descripción de estas utilidades.

Visor de sucesos

Esta herramienta, ver Figura 257, a pesar de que forma parte de Windows NT, puede servirnos para detectar errores que afecten a SQL Server.

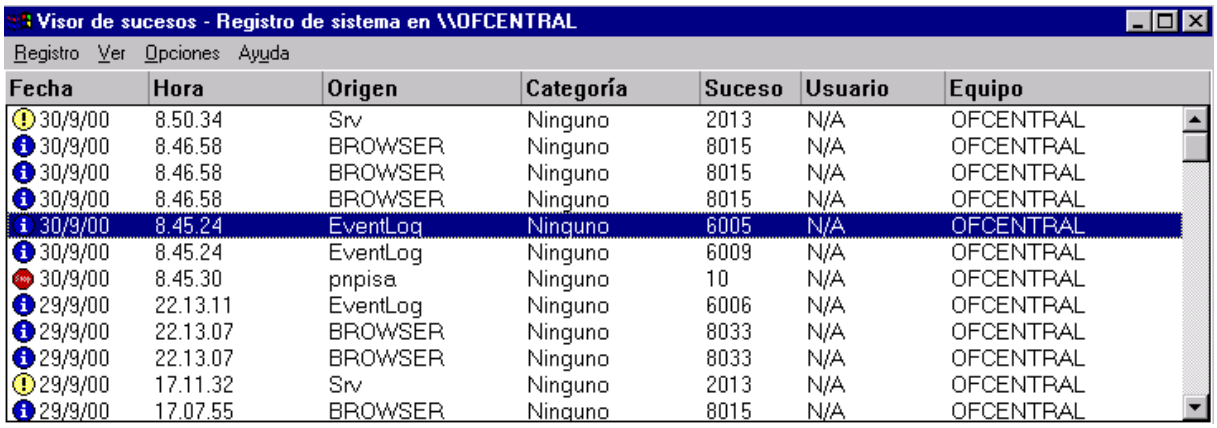


Figura 257. Visor de sucesos de Windows NT.

El Visor de sucesos registra los errores que le comunican las aplicaciones en ejecución, errores de carga de componentes, eventos del sistema, y en definitiva, cualquier evento que pueda detener o reducir el rendimiento del sistema.

Para obtener un detalle de cualquiera de los sucesos mostrados, basta con hacer doble clic sobre el mismo.

Monitor de rendimiento

A esta utilidad también se le denomina *Monitor de sistema*, está disponible tanto desde el menú de administración de Windows NT como desde el menú de SQL Server. La Figura 258 muestra una imagen de este monitor en ejecución.

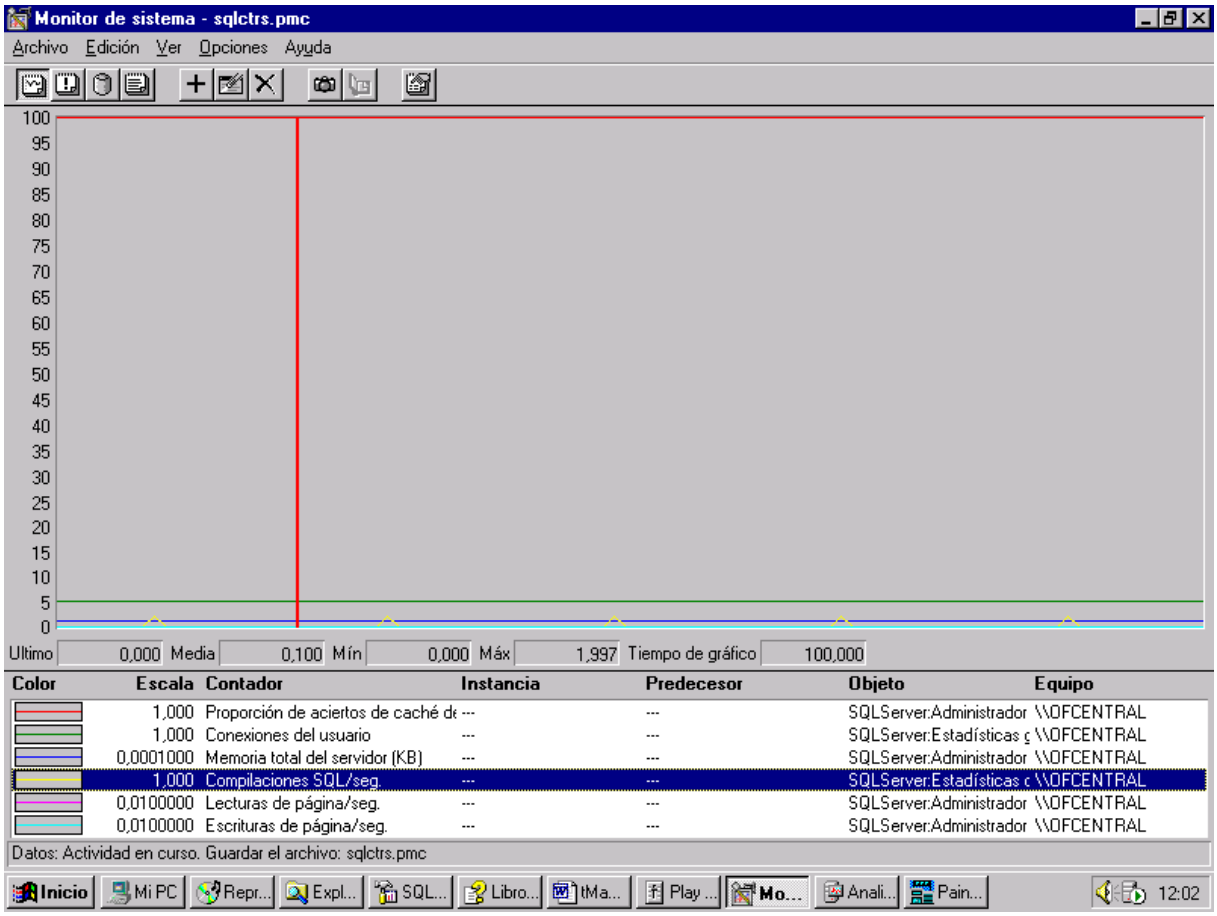


Figura 258. Monitor de sistema.

Cuando esta aplicación es iniciada desde el menú de SQL Server, muestra un conjunto de contadores de rendimiento específicos del servidor de datos. Cada contador está asociado a una gráfica que va mostrando durante el tiempo que esté en ejecución este monitor, su nivel de trabajo.

Entre los contadores para SQL Server disponibles, tenemos la cantidad de memoria usada por el servidor, número de compilaciones SQL realizadas, número de usuarios conectados, lecturas y escrituras en páginas de datos, etc.

Actividad actual del servidor

Dentro de la carpeta de SQL Server Administración, tenemos un elemento denominado *Actividad actual*, que contiene un conjunto de componentes orientados a proporcionar al administrador información diversa sobre el trabajo que se está produciendo en ese mismo instante sobre el servidor de datos. Figura 259.

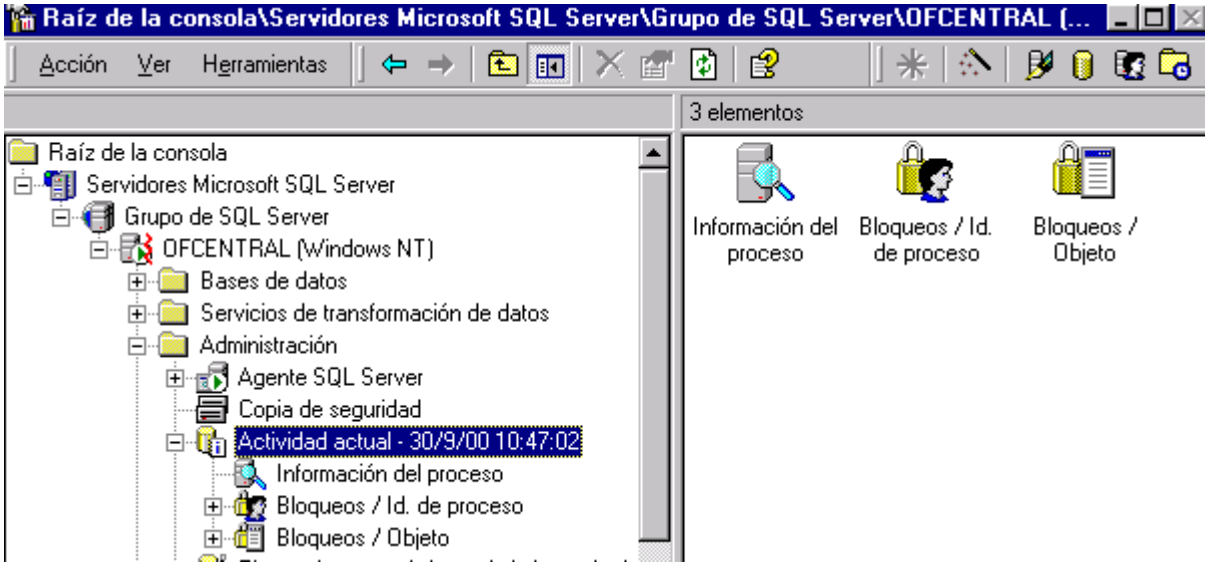


Figura 259. Actividad actual de SQL Server

Información del proceso

Muestra información sobre las conexiones abiertas hasta el momento. Haciendo doble clic sobre cualquiera de estas conexiones, obtendremos información adicional sobre la misma. Figura 260.

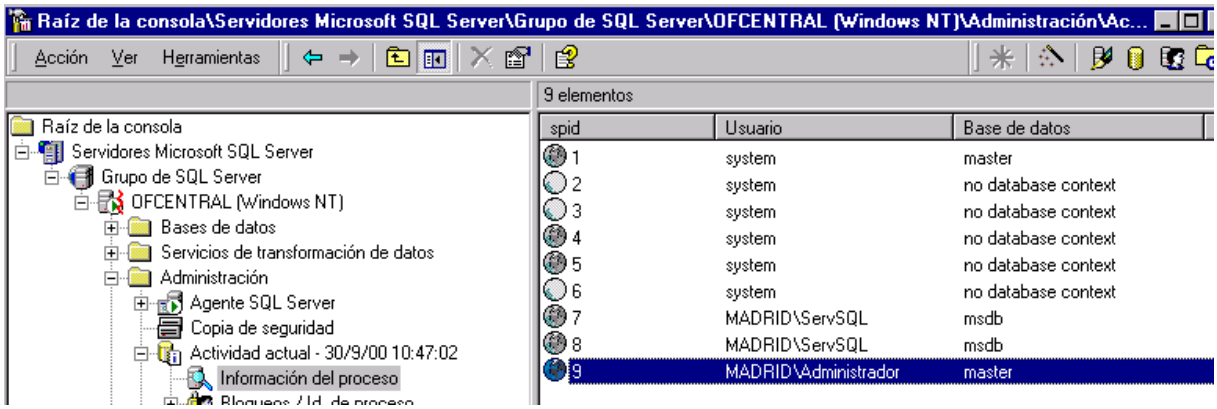


Figura 260. Procesos del servidor.

Para ver toda la información sobre las conexiones, deberemos mover la barra de desplazamiento del panel de la lista de conexiones.

Bloqueos / Id. de proceso

Este elemento muestra los bloqueos efectuados para impedir que las transacciones puedan interferirse mutuamente. Figura 261.

Podemos reconocer cada uno de los bloqueos por un identificador numérico que asigna SQL Server, y que tiene el siguiente formato: spid NumProceso.

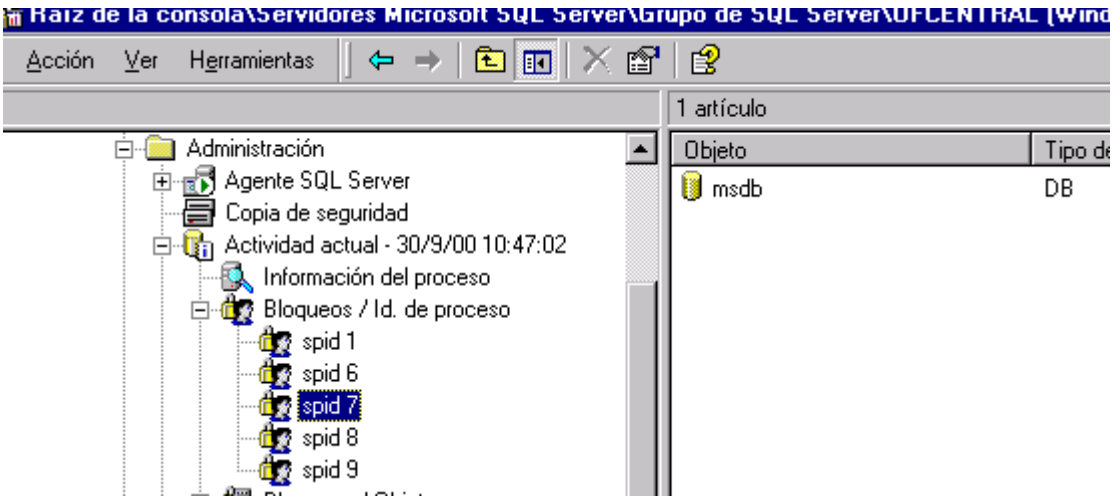


Figura 261. Bloqueos del sistema.

Bloqueos / Objeto

También muestra los bloqueos efectuados, pero en este caso, agrupados por objeto, como podemos ver en la Figura 262.

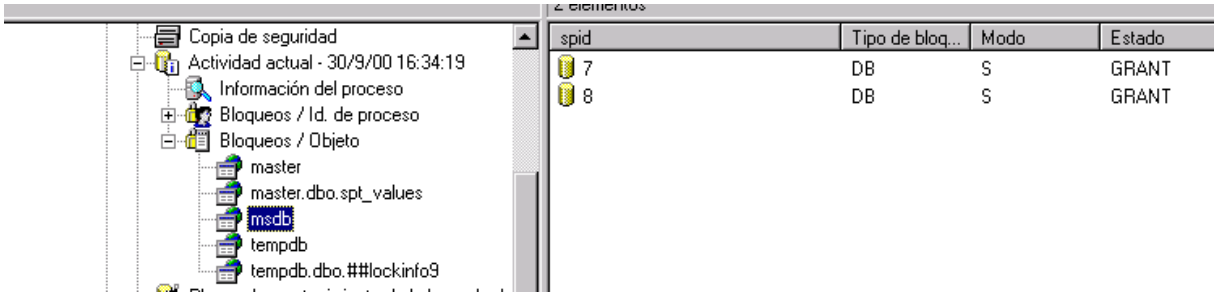


Figura 262. Bloqueos ordenados por objeto.

Registro de errores de SQL Server

El elemento *Registros de SQL Server*, que se encuentra en la carpeta Administración del Administrador corporativo, contiene el registro de errores del servidor de datos que se crea en cada sesión de trabajo de SQL Server.

Esta es una opción meramente informativa, pero que puede ser de gran ayuda al administrador de bases de datos, para averiguar la causa de problemas que hayan ocurrido en el servidor durante un periodo de actividad de la base de datos en el que no haya estado presente, y por lo tanto, no haya podido efectuar una supervisión personal.

Al hacer clic sobre cada elemento del registro, se muestra en el panel derecho del Administrador corporativo, la lista de mensajes creada por SQL Server, como muestra la Figura 263.

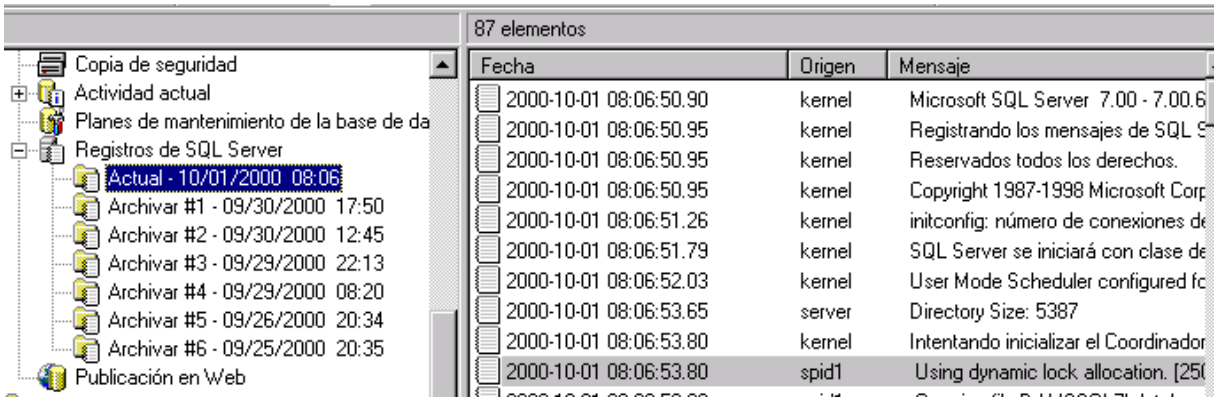


Figura 263. Registro de errores de SQL Server.

Instrucciones de comprobación del sistema

SQL Server proporciona un conjunto de instrucciones, que permiten realizar un seguimiento mediante código del rendimiento del servidor de datos. A continuación se enumeran algunos procedimientos almacenados del sistema que realizan operaciones de supervisión sobre SQL Server.

- sp_monitor. Devuelve el número de conexiones efectuadas, lecturas, escrituras de datos y tiempo empleado. La Tabla 15 muestra un ejemplo de la información devuelta.

last_run	**current_run**	**seconds**	
2000-10-01 13:49:01.870	2000-10-01 13:49:16.863	15	
cpu_busy	**io_busy**	**idle**	
115(0)-0%	3(0)-0%	19787(14)-93%	
packets_received	**packets_sent**	**packet_errors**	
6706(28)	3526(15)	0(0)	
total_read	**total_write**	**total_errors**	**connections**
918(5)	459(1)	0(0)	55(0)

Tabla 15. Datos retornados por sp_monitor.

- sp_lock. Proporciona información sobre bloqueos. En la Tabla 16 podemos ver la información que proporciona este procedimiento almacenado.

spid	**dbid**	**ObjId**	**IndId**	**Type**	**Resource**	**Mode**	**Status**
1	1	0	0	DB		S	GRANT

7	4	0	0	DB		S	GRANT
8	4	0	0	DB		S	GRANT
9	1	0	0	DB		S	GRANT
9	2	0	0	DB		S	GRANT
9	7	0	0	DB		S	GRANT
9	1	117575457	0	TAB		IS	GRANT
10	1	0	0	DB		S	GRANT

Tabla 16. Información de sp_lock.

- sp_who. Muestra los usuarios y procesos actuales del servidor de datos. Un ejemplo de su ejecución nos lo muestra la Tabla 17.

spid	status	Loginame	hostname	blk	dbname	cmd
1	sleeping	sa		0	master	SIGNAL HANDLER
2	background	sa		0	Contabilidad	LOCK MONITOR
3	background	sa		0	Contabilidad	LAZY WRITER
4	sleeping	sa		0	Contabilidad	LOG WRITER
5	sleeping	sa		0	Contabilidad	CHECKPOINT SLEEP
6	background	sa		0	Contabilidad	AWAITING COMMAND
7	sleeping	MADRID\ServSQL	OFCENTRAL	0	msdb	AWAITING COMMAND
8	sleeping	MADRID\ServSQL	OFCENTRAL	0	msdb	AWAITING COMMAND
9	runnable	MADRID\Administrador	OFCENTRAL	0	Contabilidad	SELECT
10	sleeping	MADRID\Administrador	OFCENTRAL	0	master	AWAITING COMMAND

Tabla 17. Resultado de la ejecución de sp_who.

- sp_spaceused. Devuelve información sobre el espacio utilizado por una tabla. La Tabla 18 muestra esta información sobre una tabla con el nombre Clientes.

name	rows	reserved	data	index_size	unused
Clientes	5	32 KB	8 KB	24 KB	0 KB

Tabla 18. Información del procedimiento almacenado sp_spaceused.

Seguidamente se muestran varias instrucciones del grupo *Comprobación de coherencia de base de datos*: DBCC (Database Consistency Checker), que comprueban la consistencia de los elementos de una base de datos.

- DBCC CHECKDB ('NombreBD'). Comprueba la integridad de los objetos pertenecientes a la base de datos pasada como parámetro.

- DBCC CHECKTABLE (NombreTabla'). Comprueba la integridad de páginas, columnas e índices de la tabla pasada como parámetro.

- DBCC MEMUSAGE. Devuelve información sobre el uso que están haciendo de la memoria, los objetos activos en el momento actual. La Tabla 19 muestra esta información relativa al servidor en el que se han realizado estas pruebas.

dbid	objectid	indexid	buffers
2	-516086717	0	44
2	-500086660	0	44
7	6	0	18
4	2	0	7
7	3	0	7
7	3	2	7
1	1	0	6
1	6	0	6
1	2	255	5
1	3	0	5
2	6	0	5
1	1	2	4
1	3	2	4

1	36	0	4
2	3	2	4
2	99	0	4
1	2	0	3
1	6	1	3
2	3	0	3
7	2	255	3

Tabla 19. Información sobre recursos de memoria utilizados por SQL Server.

Para una referencia completa de este tipo de instrucciones, consulte el lector los *Libros en pantalla de SQL Server*.

El Analizador de SQL Server

Nos permite obtener información sobre la gestión del servidor y todos los eventos que puedan suceder y que influyan negativamente en su funcionamiento, como problemas con los bloqueos, consultas que afecten al rendimiento, lecturas y escrituras en disco, etc. Todos estos datos serán recopilados a través de una traza.

Una traza es un elemento de SQL Server, que el administrador o cualquier usuario con los permisos necesarios define, indicando cuáles son los aspectos del servidor a supervisar. Después de configurar una traza, podemos ejecutarla de manera que vaya tomando la información que hemos indicado. La forma más sencilla de crear una traza en el Analizador es mediante el asistente de creación de trazas, que iniciaremos seleccionando la opción de menú Herramientas+Asistente para creación de trazas. La pantalla inicial de este asistente se muestra en la Figura 264.

Figura 264. Asistente para la creación de trazas.

El primer paso de este asistente, consiste en indicar el servidor SQL Server sobre el que vamos a crear la traza y el tipo de problema a seguir. En este ejemplo vamos a comprobar el rendimiento de un procedimiento almacenado, como muestra la Figura 265.

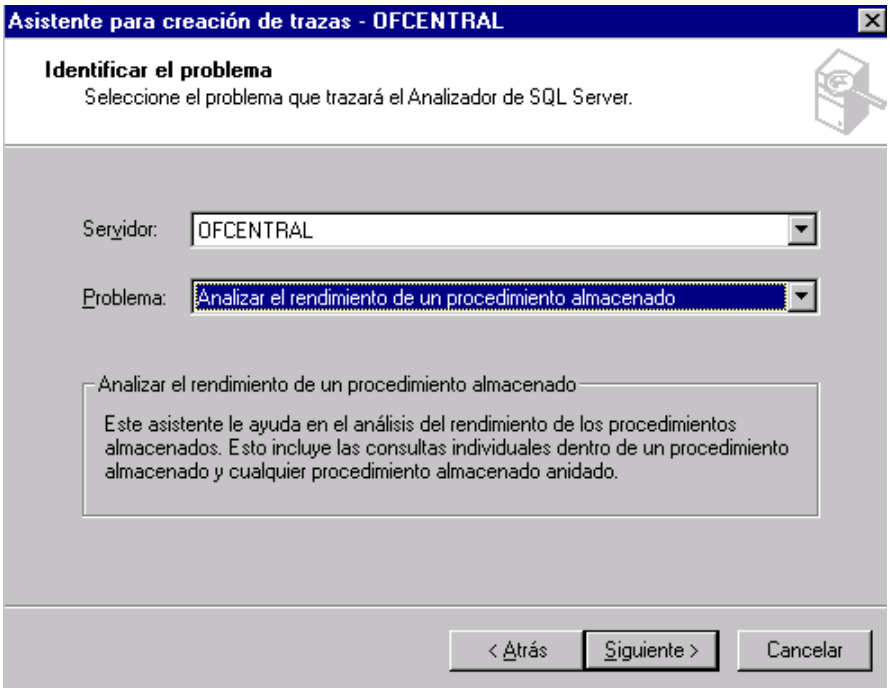


Figura 265. Indicación del problema a trazar en el Analizador.

En el siguiente paso, seleccionaremos la base de datos y el procedimiento almacenado que deseamos analizar, Figura 266.

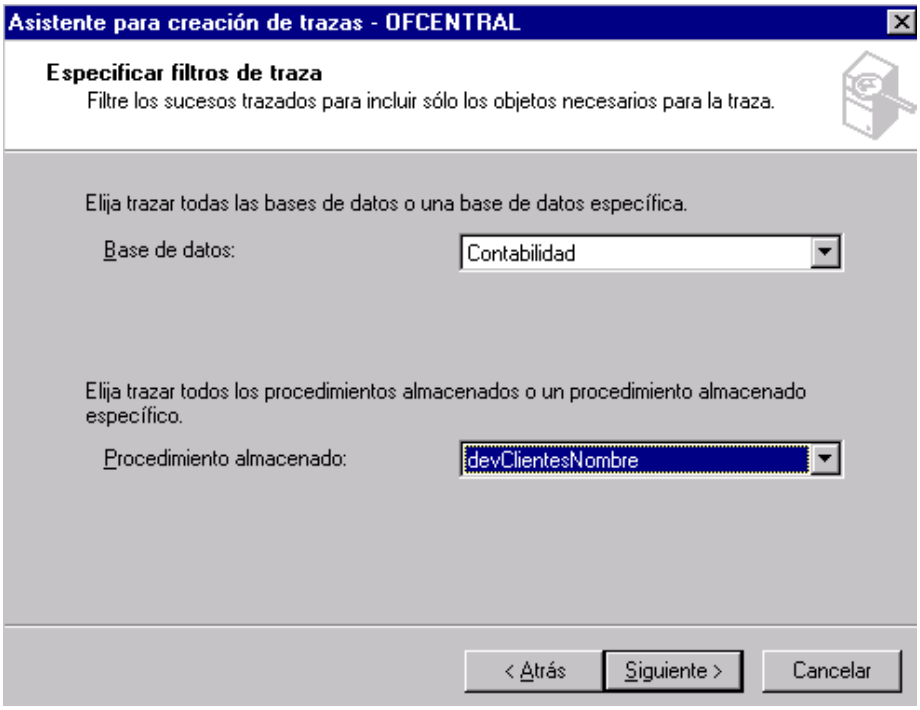


Figura 266. Selección de base de datos y procedimiento almacenado a analizar.

Como paso final, daremos un nombre a la traza que estamos creando, y pulsaremos Finalizar para completar la creación de la traza, ver Figura 267.

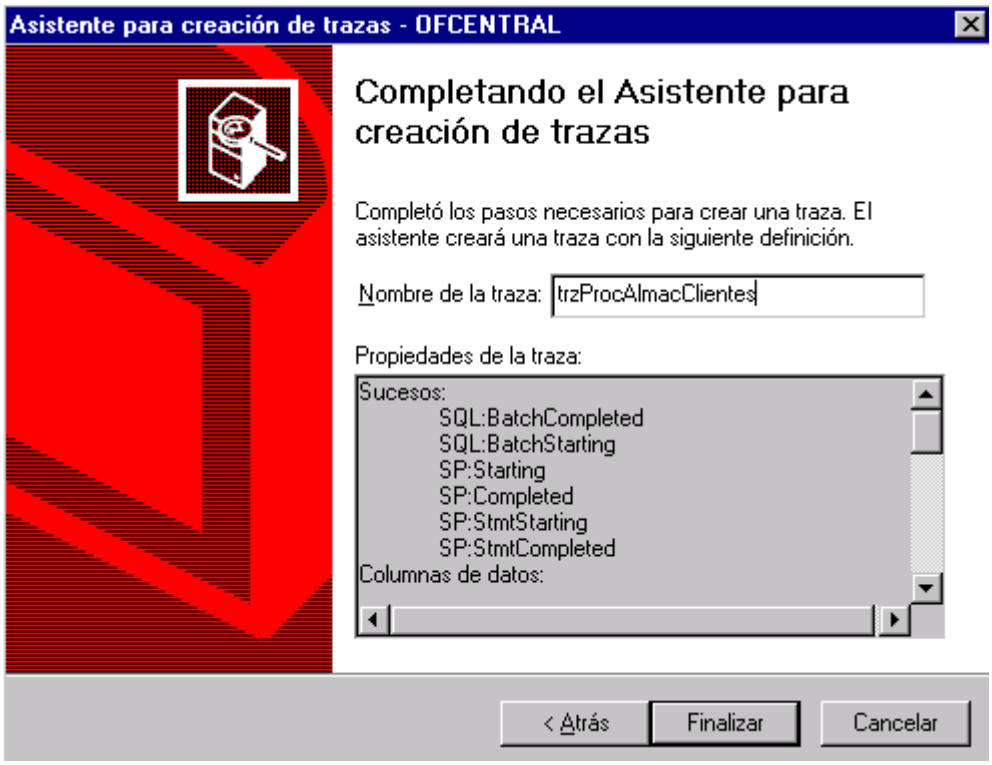


Figura 267. Finalizando la creación de la traza.

En el momento en que se complete la creación de la traza, esta se pondrá en ejecución, registrando todos los eventos para los que ha sido programada. La ejecución de trazas será tratada en un apartado posterior.

Creación manual de trazas

Es posible realizar la creación de una traza indicando manualmente los aspectos que debe supervisar, sin utilizar el asistente para trazas. Para ello, seleccionaremos en el Analizador de SQL Server la opción de menú *Archivo+Nuevo+Trazar*, que nos mostrará la ventana de propiedades de la traza.

En la pestaña General de esta ventana, escribiremos el nombre de la traza, servidor a supervisar y el destino de la información recopilada: archivo de traza o una tabla de base de datos. Véase la Figura 268.

En la pestaña Sucesos asignaremos a la traza los eventos que deberá controlar. Seleccionaremos el suceso en la lista *Sucesos disponibles*, que agrupa todos los sucesos por categorías, y pulsando el botón Agregar, se añadirá a la lista *Sucesos seleccionados*, como vemos en la Figura 269.

A continuación, en la pestaña *Columnas de datos*, debemos elegir la información que nos devolverá la traza al ejecutarse.

La lista *Datos seleccionados* muestra las columnas de datos que serán devueltas por defecto, como el nombre de usuario, tipo de suceso, etc., pero adicionalmente, podemos elegir en la lista *Datos no*

seleccionados otro tipo de información, como el nombre del servidor en el que se produce la traza, hora final del suceso, identificadores de base de datos, objeto, etc. Al hacer clic sobre el nombre de un dato en cualquiera de estas dos listas, se muestra en la parte inferior de la ventana, una breve descripción de la información devuelta por el dato, como muestra la Figura 270.

Figura 268. Propiedades generales de una traza.

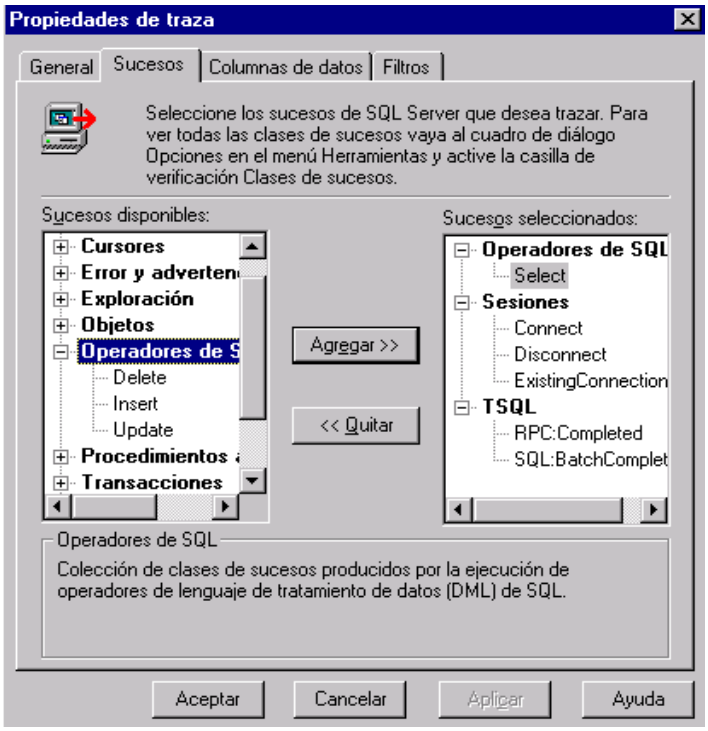


Figura 269. Selección de sucesos que la traza debe seguir.

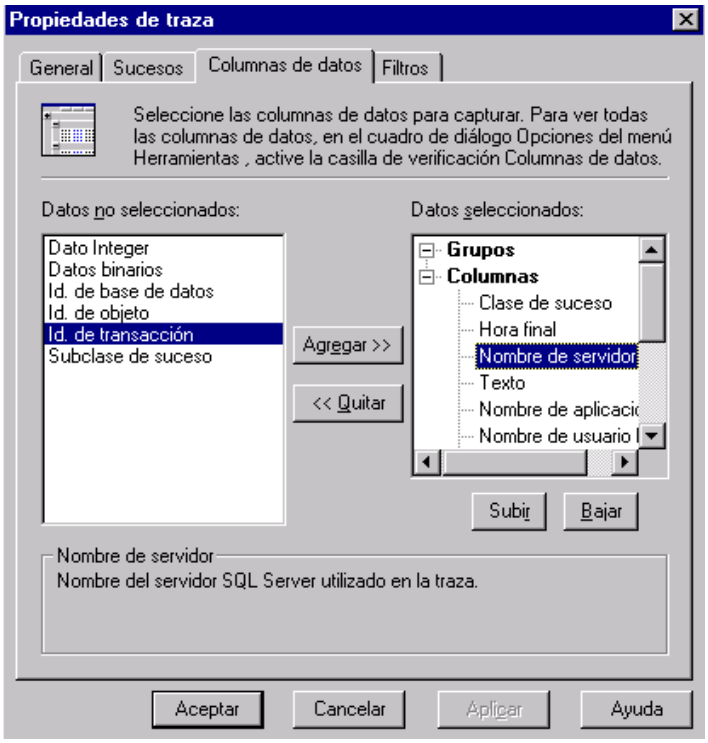


Figura 270. Selección de datos que devolverá la traza.

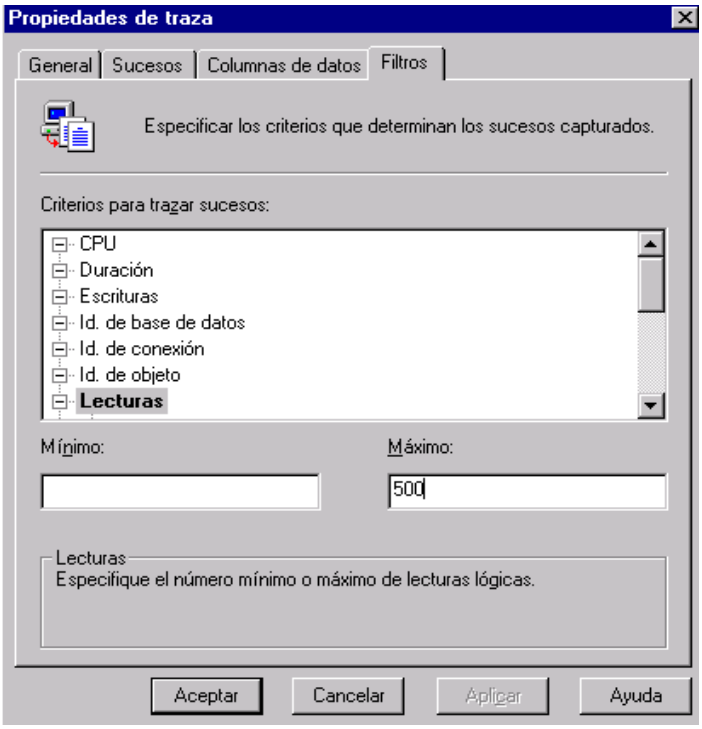


Figura 271. Filtros para la traza.

Finalmente, en la pestaña Filtros, indicaremos los rangos o valores que limitarán la información devuelta por la traza, es decir, podemos indicar un número mínimo y máximo de escrituras lógicas y físicas a realizar en la traza, un identificador de base de datos para supervisar la actividad sólo de dicha base de datos, etc. La Figura 271 muestra el aspecto de esta parte de las propiedades de la traza.

Al igual que sucedía en la creación de trazas con el asistente, al finalizar la creación de la traza, esta se pondrá en ejecución, comenzando a devolver toda la actividad para la que ha sido diseñada.

Ejecución de trazas

Una vez que hayamos finalizado la creación de una traza tanto con su asistente como manualmente, se iniciará su ejecución en la ventana correspondiente, aunque también es posible seleccionar una traza creada previamente mediante la opción de menú del Analizador *Archivo+Abrir*, que nos dará paso a un submenú en el que podremos abrir las trazas definidas en el Analizador, en un archivo o en una tabla de la base de datos.

Si elegimos abrir una definición de traza, se mostrará la ventana de propiedades de traza con la lista de trazas definidas. En la Figura 272 se muestra la selección de la traza que hemos generado anteriormente con el asistente de creación.

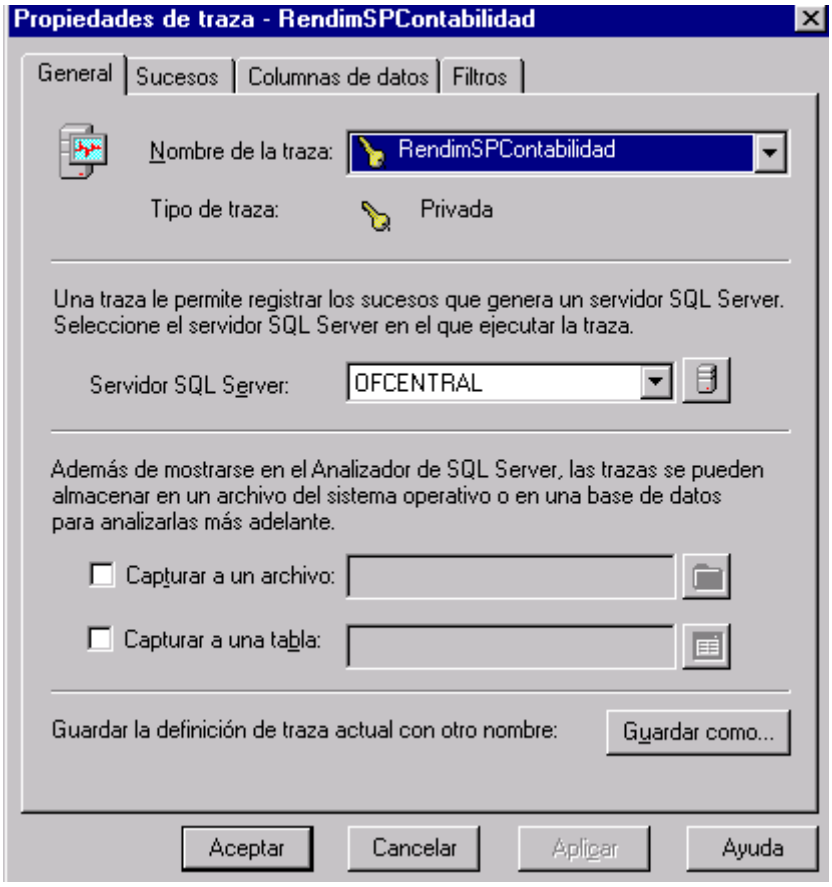


Figura 272. Abrir una definición de traza.

Al pulsar Aceptar, se abrirá la ventana de ejecución de traza en la que se registrará toda su actividad. En este caso concreto, cada vez que se ejecute el procedimiento almacenado definido en la traza, podremos ver la información de estas ejecuciones en los paneles informativos de la ventana, como se muestra en la Figura 273.

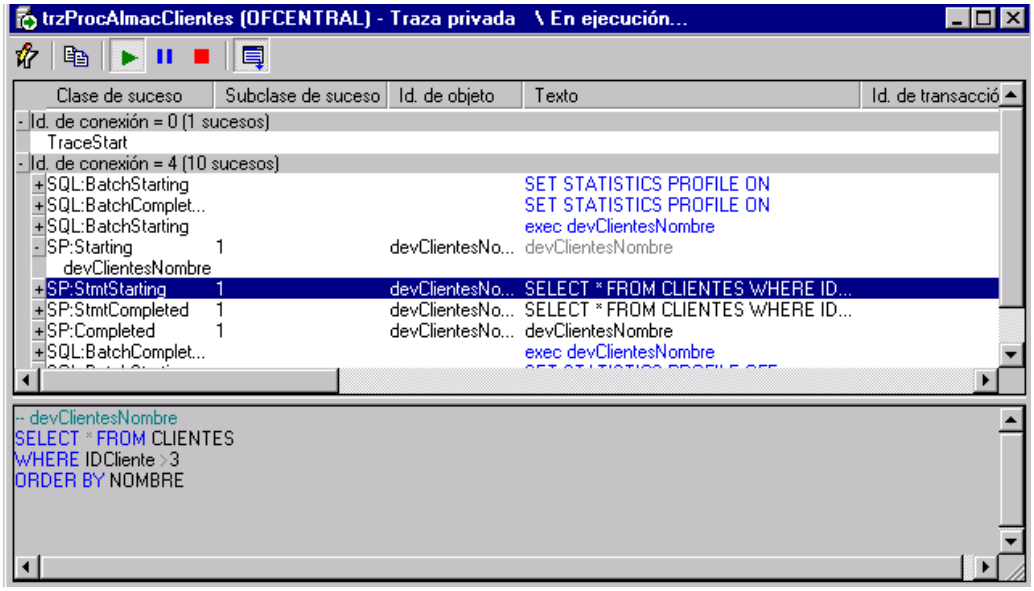


Figura 273. Ventana de ejecución de traza.

El panel superior muestra los sucesos que se van produciendo. Si un suceso contiene datos adicionales, podrá expandirse pulsando el signo más (+) que aparece en el lateral izquierdo. Para ver toda la información que proporciona una línea de suceso, deberemos desplazar la línea utilizando la barra de desplazamiento de dicho panel. El panel inferior contiene el detalle del suceso seleccionado en el panel superior.

Podemos detener la traza en cualquier momento, pulsando el botón de la barra de herramientas en la ventana de la traza o la opción de menú *Archivo+Detener trazas* del Analizador.

Ejecución de archivos de traza

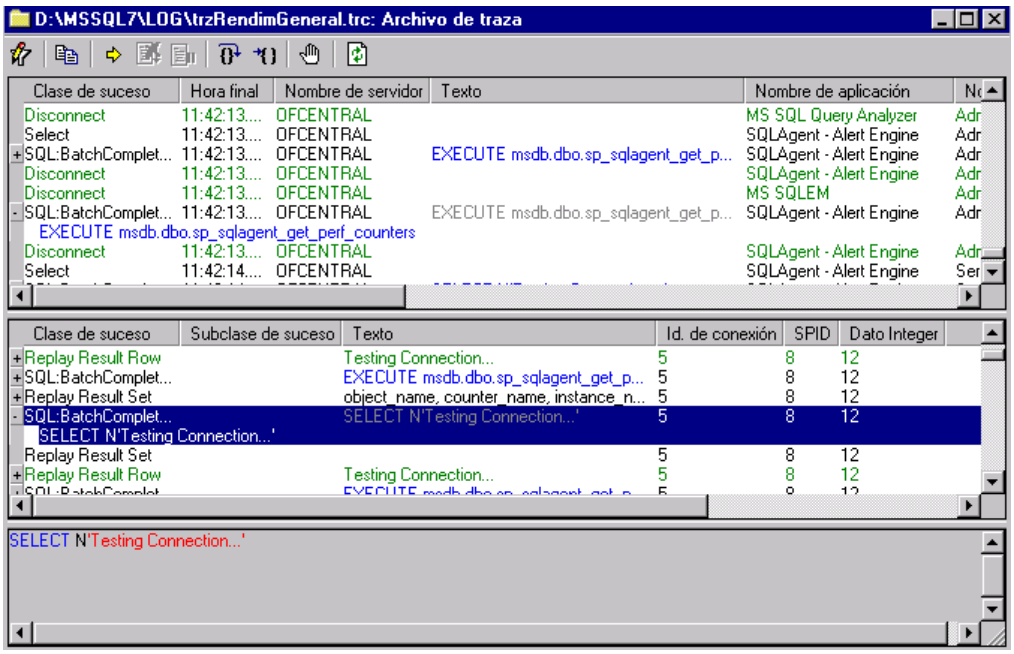


Figura 274. Reproducción de una traza grabada en un fichero.

Si hemos grabado la actividad de una traza a un fichero con extensión .TRC, al abrir dicho fichero desde el Analizador, se mostrará toda esa información registrada en el momento en que se realizó la traza.

Podemos volver a reproducir la situación grabada en la traza pulsando el botón *Comenzar la ejecución*, en la ventana de traza. En este caso, dicha ventana mostrará tres paneles: el primero contendrá el registro de actividad grabado en la traza; el siguiente mostrará los sucesos que ocurren en la ejecución actual de la traza; y el último visualizará el detalle de un suceso pulsado en cualquiera de los anteriores paneles. La Figura 274 muestra un ejemplo de esta situación.

El Analizador de consultas

A pesar de que es una herramienta fundamentalmente encaminada a realizar operaciones de manipulación de datos, el Analizador de consultas permite visualizar un estudio sobre el nivel de recursos consumidos al ejecutar cada consulta contra una base de datos.

Podemos ejecutar este analizador desde el menú de opciones de SQL Server 7.0 en el sistema operativo, o iniciarlo desde el propio Administrador corporativo, en su opción de menú *Herramientas+Analizador de consultas de SQL Server*.

Seleccionando su opción de menú *Consulta+Mostrar plan de ejecución*, se añadirá en la parte inferior del panel de resultados una pestaña, que mostrará una estimación sobre las operaciones que realiza el servidor para llevar a cabo la ejecución de la sentencia ejecutada en el panel de instrucciones.

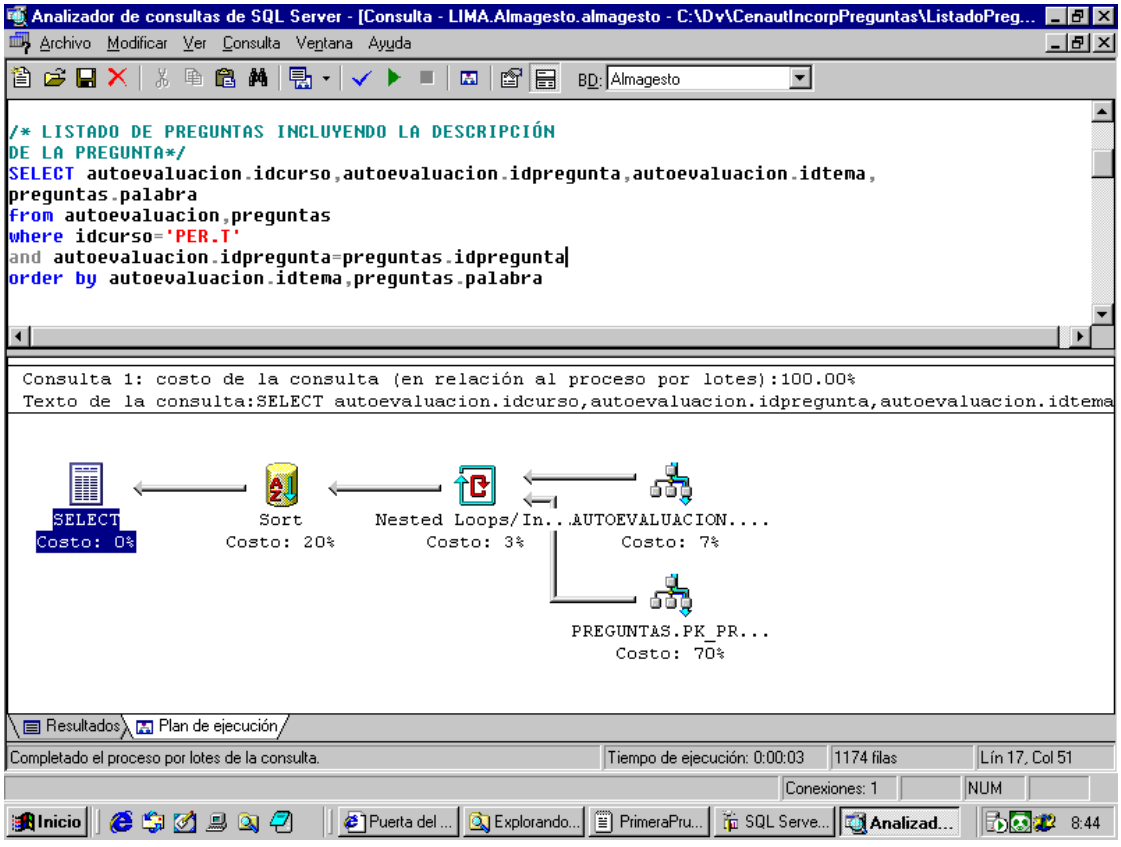


Figura 275

25

Planes de mantenimiento

Creación de un plan de mantenimiento para una base de datos

El diseño de un plan de mantenimiento para una o varias bases de datos de un servidor SQL Server, permite automatizar las tareas más habituales que el administrador del sistema debe hacer regularmente, liberándole de estos trabajos rutinarios y proporcionándole más tiempo para otras tareas administrativas que requieran su intervención personal.

El medio más sencillo de crear un plan de mantenimiento, consiste en abrir la carpeta Administración de SQL Server y hacer clic con el botón derecho sobre el elemento *Planes de mantenimiento de la base de datos*, seleccionando la opción del menú contextual *Nuevo plan de mantenimiento*, que pondrá en marcha el asistente para crear un nuevo plan, Figura 276.

El primer paso de este asistente consiste en especificar para qué bases de datos se va a crear el plan de mantenimiento: todas, sistema, servidor, etc. En nuestro ejemplo vamos a seleccionar sólo una base de datos de usuario, como vemos en la Figura 277.

A continuación se deben configurar los valores para organizar las páginas de datos e índices, liberación de espacio no usado por la base de datos y establecer si se desea, una programación para realizar este aspecto regularmente, como se muestra en la Figura 278.

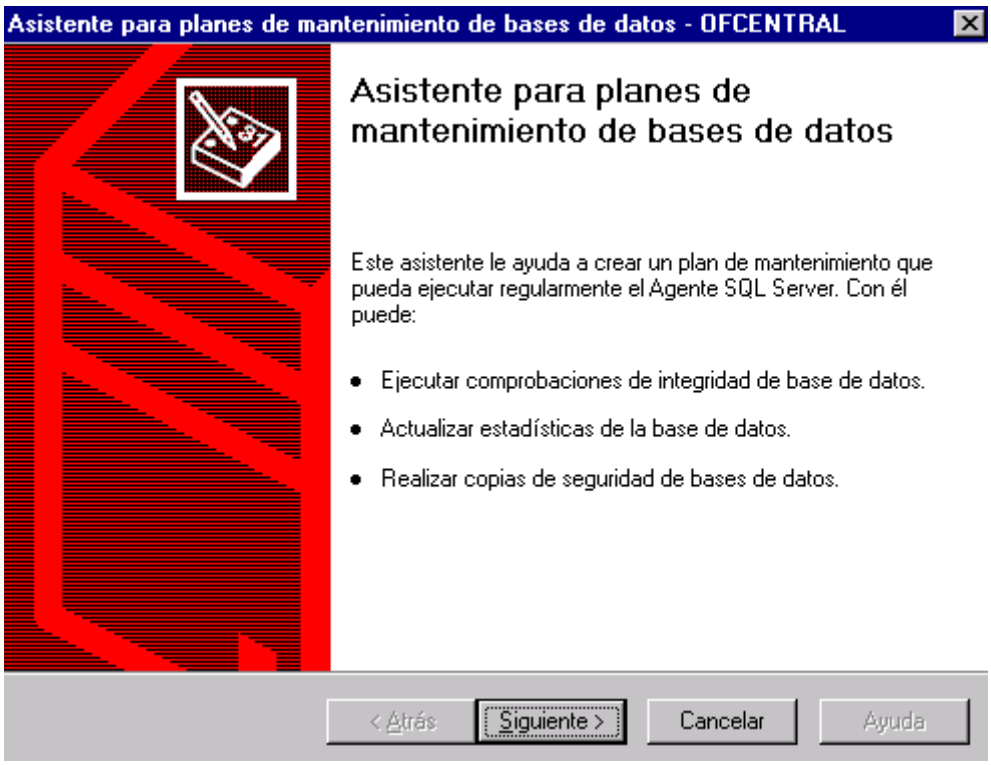


Figura 276. Inicio del asistente para planes de mantenimiento de bases de datos.

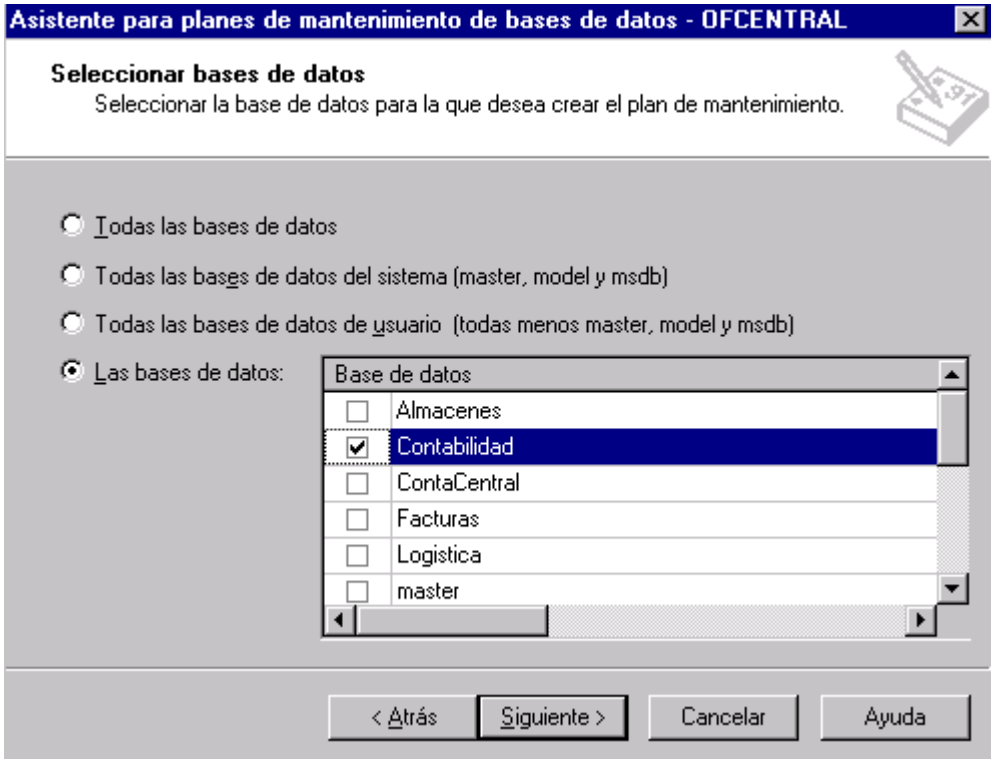


Figura 277. Selección de la base de datos para el plan de mantenimiento.

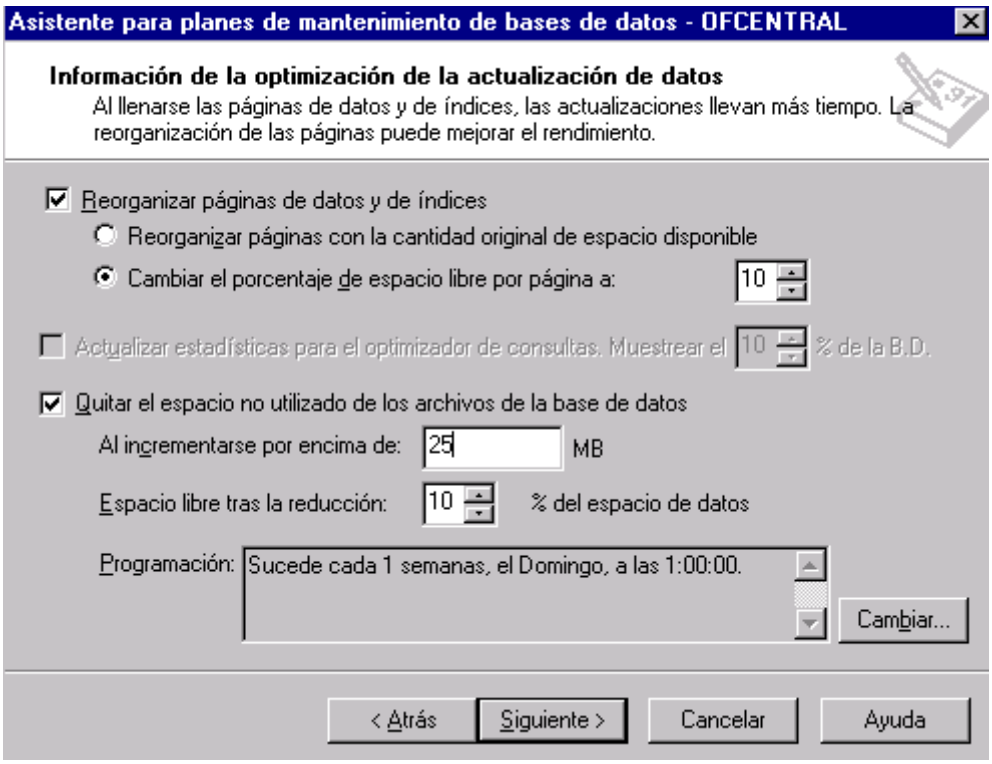


Figura 278. Configuración de la organización de páginas de la base de datos.

En el siguiente paso indicaremos si deseamos comprobar la integridad de datos e índices, ver Figura 279.

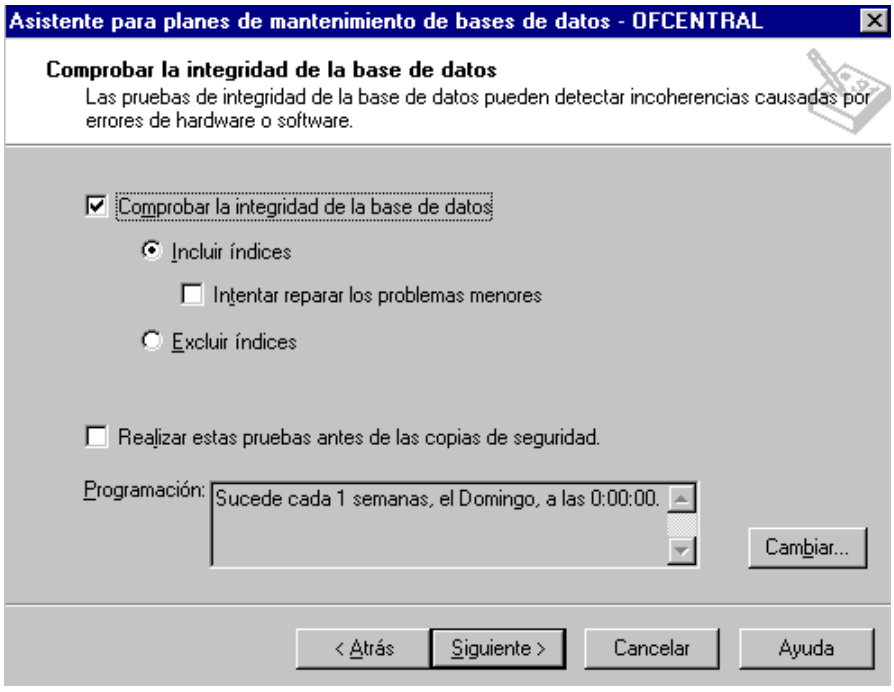


Figura 279. Valores para comprobar la integridad de los datos.

Si vamos a realizar una copia de seguridad, el soporte y programación de su intervalo, lo estableceremos en este paso que muestra la Figura 280.

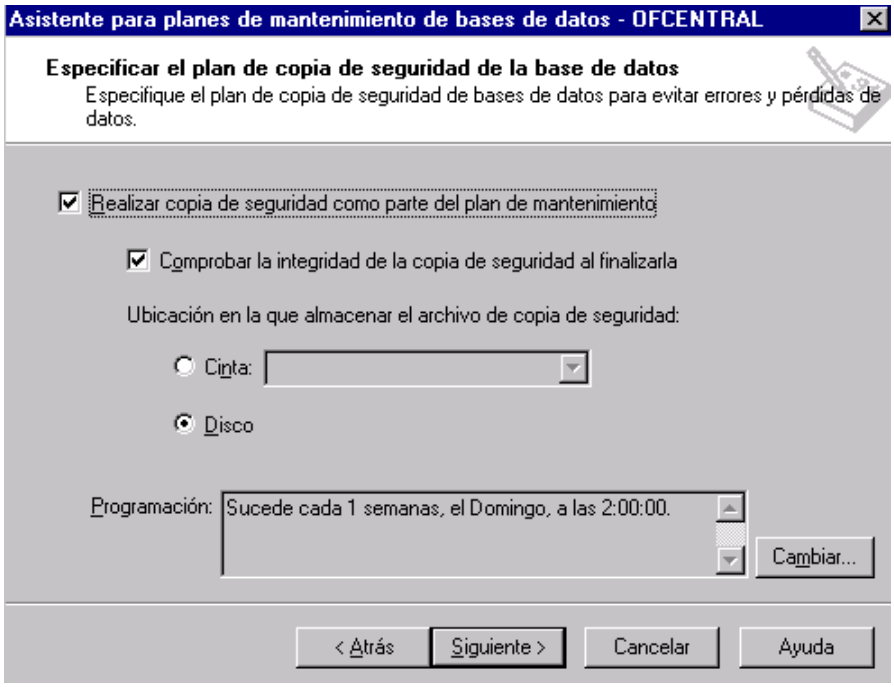


Figura 280. Valores para la copia de seguridad en el plan de mantenimiento.

Siguiendo con la copia de seguridad, en el siguiente paso estableceremos las propiedades para la ubicación de los ficheros generados por la copia, ver Figura 281.

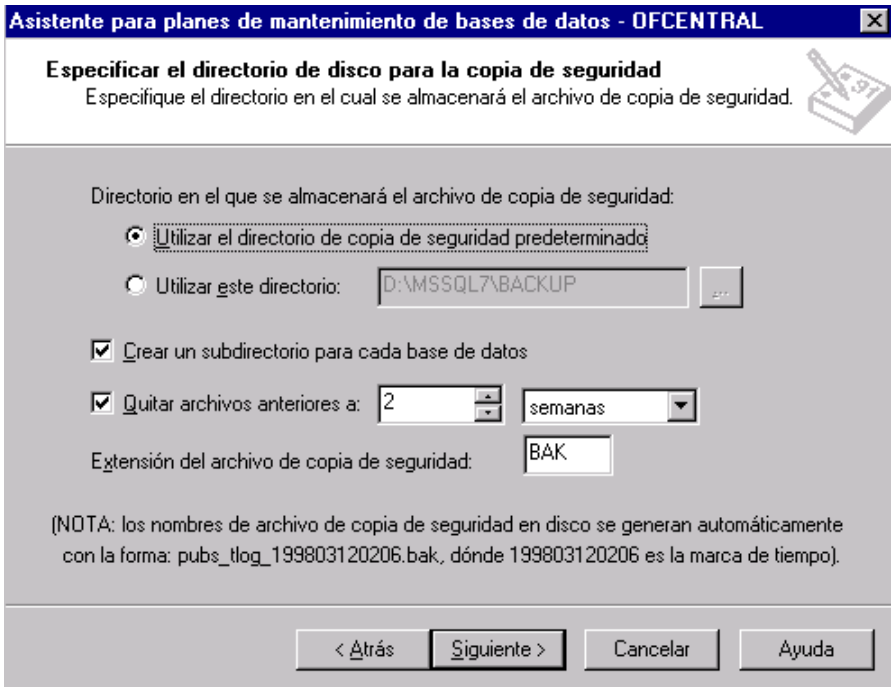


Figura 281. Ubicación de la copia de seguridad en el plan de mantenimiento.

En cuanto a la copia de seguridad del registro de transacciones, la Figura 282 nos muestra la pantalla de configuración correspondiente.

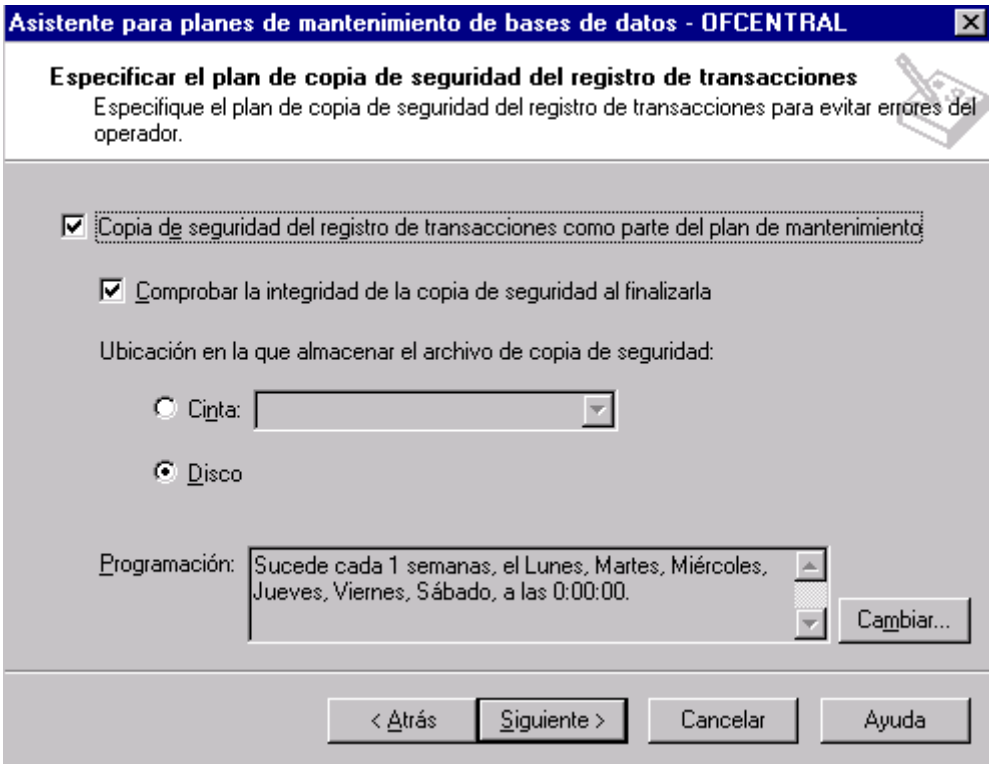


Figura 282. Configuración de la copia de seguridad del registro de transacciones.

Al igual que con la copia de seguridad de datos, en el siguiente paso, Figura 283, indicaremos la ubicación de la copia de seguridad del registro de transacciones.

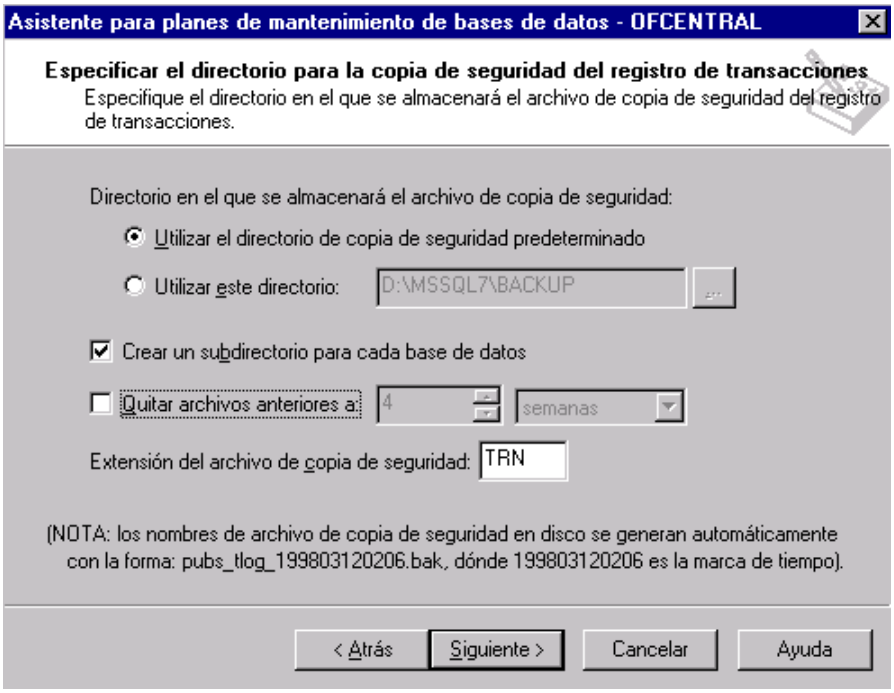


Figura 283. Ubicación de la copia de seguridad del registro de transacciones.

El plan de mantenimiento emite informes para que el administrador del sistema pueda conocer las incidencias acaecidas durante la ejecución de los distintos aspectos del plan. La Figura 284 muestra el siguiente paso del asistente, en el que indicaremos el lugar en el que dichos informes serán generados.

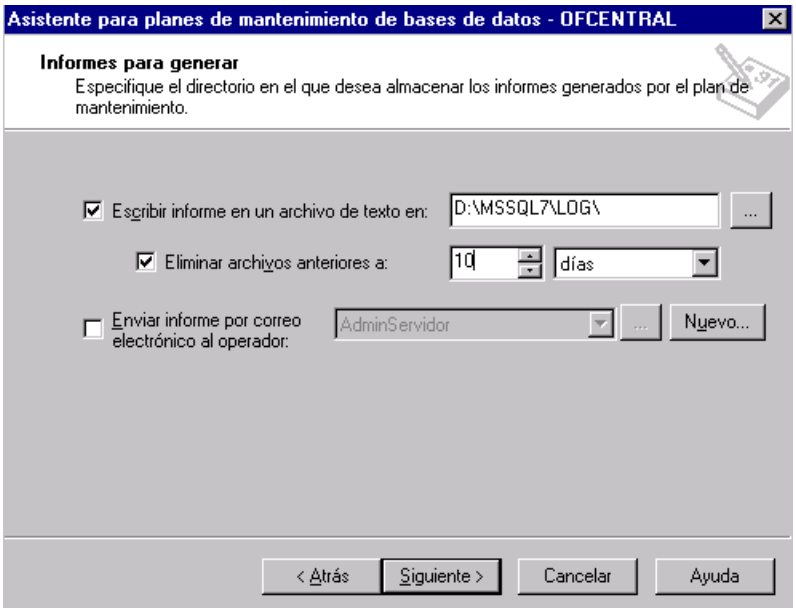


Figura 284. Ubicación de informes creados por el plan de mantenimiento.

El historial de las operaciones realizadas por el plan de mantenimiento puede ser grabado en el servidor local o en uno remoto. En este paso indicaremos dicho lugar, como vemos en la Figura 285.

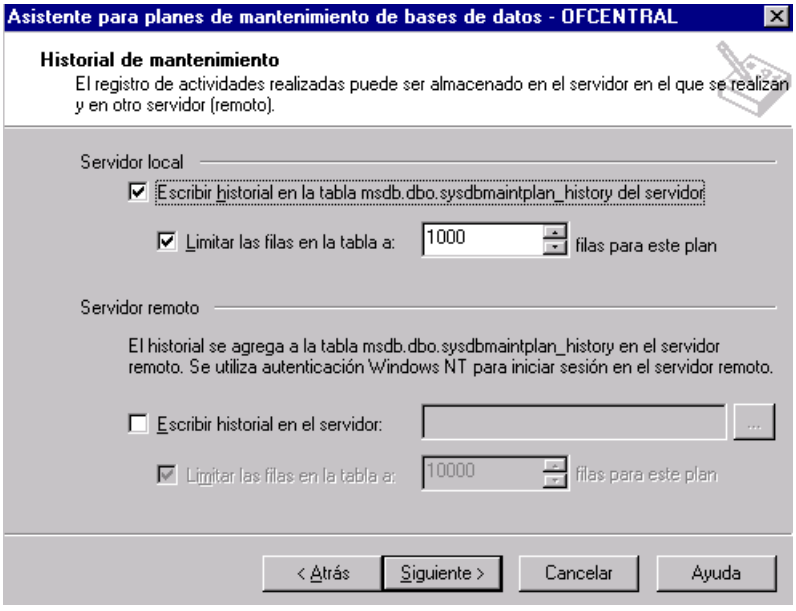


Figura 285. Valores de grabación del historial del plan de mantenimiento.

Por fin llegamos al paso final de este asistente, en el que daremos un nombre al plan de mantenimiento y lo grabaremos pulsando el botón Finalizar. Ver Figura 286.

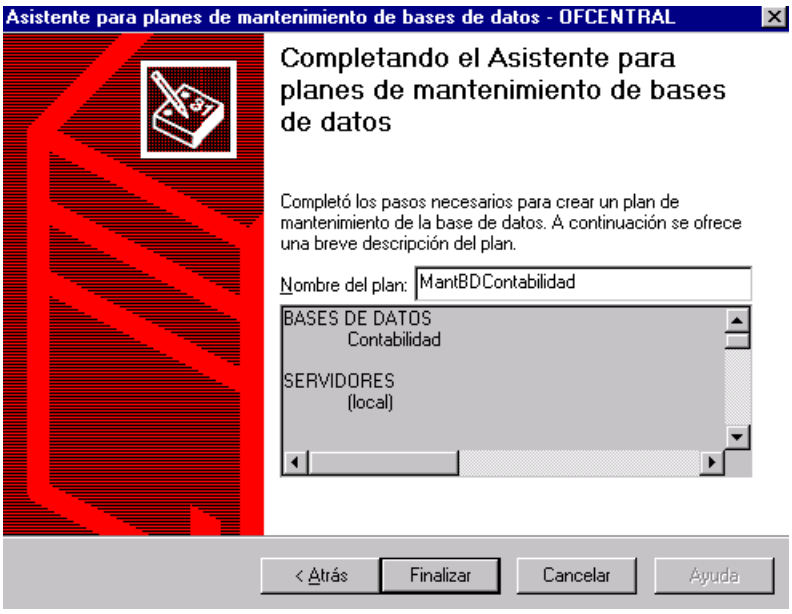


Figura 286. Fin del asistente para planes de mantenimiento de bases de datos.

Modificación de un plan de mantenimiento creado

Para visualizar o modificar la configuración de un plan de mantenimiento, sólo tenemos que pulsar sobre *Planes de mantenimiento de la base de datos* en el Administrador corporativo, que nos mostrará en el panel derecho la lista de planes creados. Haciendo doble clic sobre uno de los planes, se abrirá la ventana de la Figura 287, correspondiente a sus propiedades.

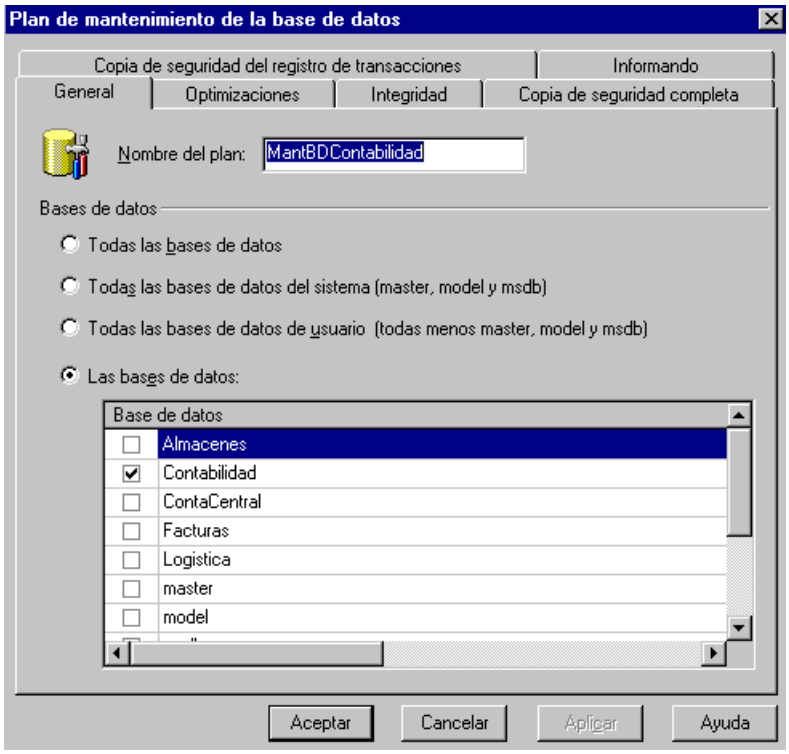


Figura 287. Propiedades de un plan de mantenimiento.

Cada pestaña de esta ventana corresponde a un aspecto específico del plan, sólo debemos hacer clic en la que necesitemos para alterar sus valores.

Historial de planes de mantenimiento

Para obtener información sobre los planes de mantenimiento del servidor, haremos clic con el botón derecho sobre *Planes de mantenimiento de la base de datos*, en el Administrador corporativo, seleccionando la opción del menú contextual *Historial del plan de mantenimiento*, que mostrará la ventana de la Figura 288, conteniendo la información de los planes de mantenimiento creados, si es que existe alguno.

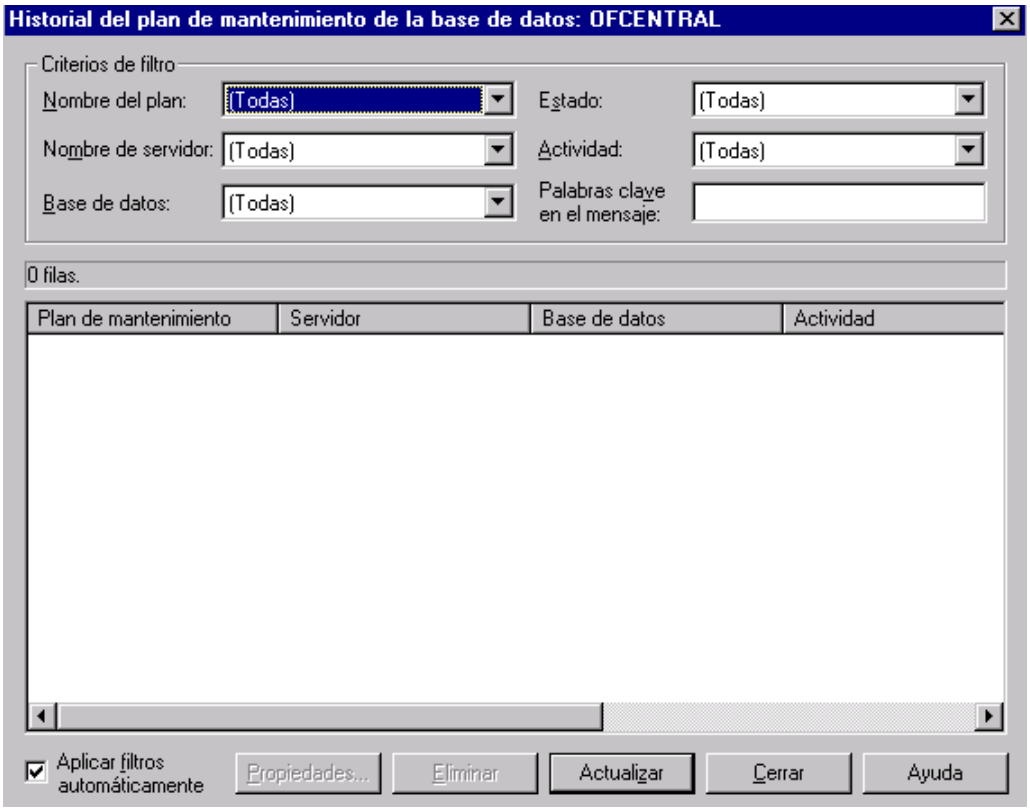


Figura 288

26

Duplicación de datos en SQL Server

¿En qué consiste la duplicación?

La duplicación o replicación de datos es una técnica que permite transmitir la información desde una base de datos origen a una o varias bases de datos destino, de forma que al ser implantada empresarialmente, se constituya en una solución que mantenga informados por igual a los diversos servidores departamentales de la compañía, distribuidos físicamente a lo largo de un territorio.

Causas para implantar un sistema de duplicación de datos

El lector puede estar preguntándose en este momento qué necesidad tiene de un nuevo medio de transferencia de datos, si ya dispone de los sistemas de copia de seguridad y los servicios de transformación de datos (DTS).

Supongamos que en nuestro servidor SQL Server tenemos una base de datos de un gran tamaño (superior a 80GB), que además debe soportar durante la jornada normal de trabajo consultas, actualizaciones y borrados constantes de datos realizadas por cientos de usuarios.

Si a esta situación unimos el hecho de que los datos deben actualizarse lo más rápidamente posible en todos los servidores de las diferentes sucursales de la empresa, para poder disponer en cualquier lugar de la misma información, la realización de copias de seguridad o transferencia de datos usando el DTS no es, evidentemente, la solución adecuada para este problema, ya que cuanto mayor es la cantidad de datos a copiar / restaurar o transferir, mayores son las posibilidades de bloqueo de las tablas que componen la base datos y consecuentemente, interrupción en el trabajo de los usuarios que utilizan dicha base de datos.

Al aplicar la duplicación para la distribución de nuestros datos, estos no se transmitirán más rápidamente, pero sí lo harán de manera coordinada con el resto de transacciones que estén en ese momento produciéndose sobre la base de datos, de forma que el impacto en el rendimiento sea mínimo, y la nueva información se actualice convenientemente. El resultado será un término medio entre velocidad de transferencia y disponibilidad de datos.

El esquema de duplicación de SQL Server

Para realizar la duplicación, SQL Server toma como ejemplo el modelo de distribución de documentación de una editorial, en el que según la información transmitida y el medio empleado para difundirla, podemos distinguir varios roles o personajes en los siguientes grupos lógicos.

Contenedores y transmisores de información

- Publicador. Servidor encargado del mantenimiento de las bases de datos de origen, proporciona la información que posteriormente se encargará de transmitir el distribuidor.

- Distribuidor. Es el servidor que efectúa la recepción de los datos modificados por el publicador y se encarga de propagarlos a los suscriptores con una frecuencia periódica.

- Suscriptor. Es un servidor que recibe los datos del publicador a través del distribuidor. Generalmente actuará como receptor de información, pero habrá ocasiones en que podrá modificar los datos recibidos y enviarlos de vuelta al publicador para actualizar este.

Cada uno de estos roles puede residir en servidores separados o estar todos juntos en un mismo servidor, ello dependerá de la cantidad de información a publicar y la estrategia de duplicación diseñada.

Información transmitida

- Artículo. Representa la unidad de información en el modelo de duplicación, pudiendo ser una tabla de una base de datos, una selección de datos de una tabla o un procedimiento almacenado.

- Publicación. Al igual que en el modelo editorial, una publicación es un conjunto de uno o varios artículos. La publicación es el elemento que se transmite entre publicadores y suscriptores.

La Figura 289 muestra un esquema del proceso de trabajo en una duplicación y las figuras que la integran.

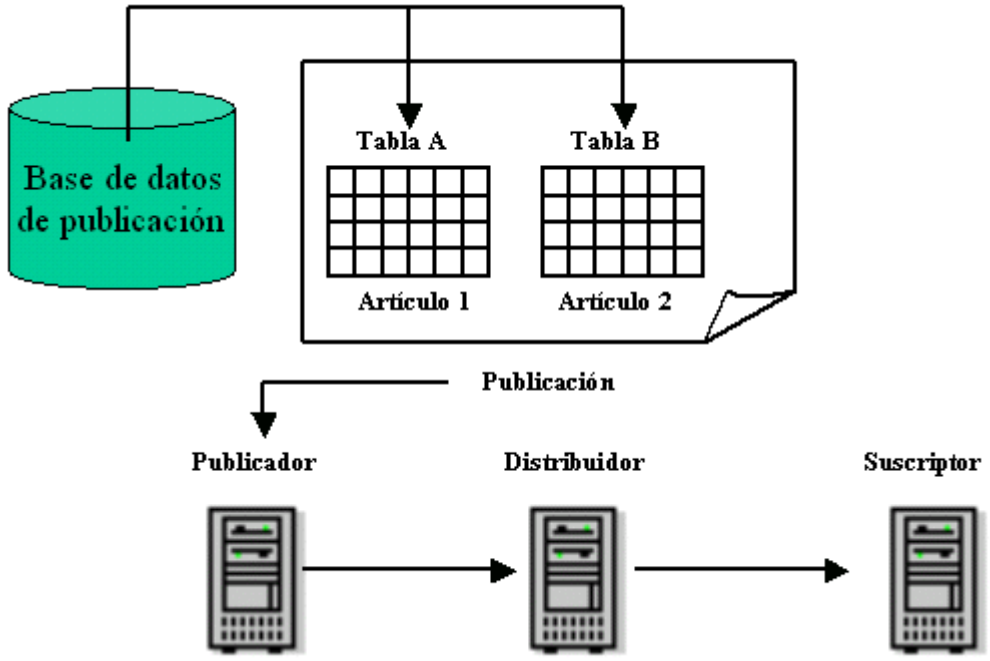


Figura 289. Esquema de funcionamiento de la duplicación.

Selección de información para un artículo

En el anterior apartado se ha explicado que un artículo puede estar formado por una tabla o una selección de datos de dicha tabla. Al conjunto de información extraída de una tabla para crear un artículo se le denomina filtro, y según como sea creado se dividen en los siguientes tipos:

- Filtro vertical. El artículo contiene un conjunto de columnas de una tabla, de forma que el suscriptor verá todas las filas de la tabla pero sólo determinadas columnas, Figura 290.

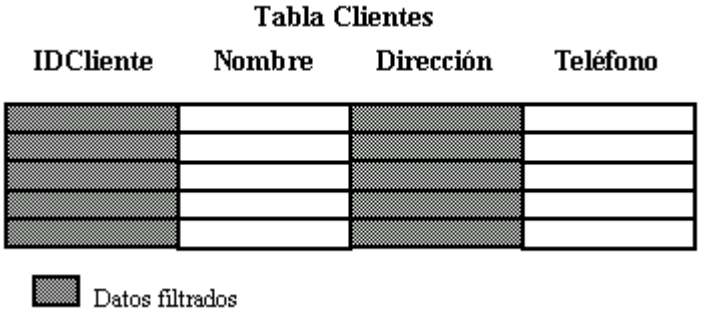


Figura 290. Filtro vertical para un artículo.

- Filtro horizontal. El artículo está formado por un conjunto de filas de una tabla. En este caso, el suscriptor verá todas las columnas de la tabla, pero no tendrá acceso a todos los registros que la componen, Figura 291.

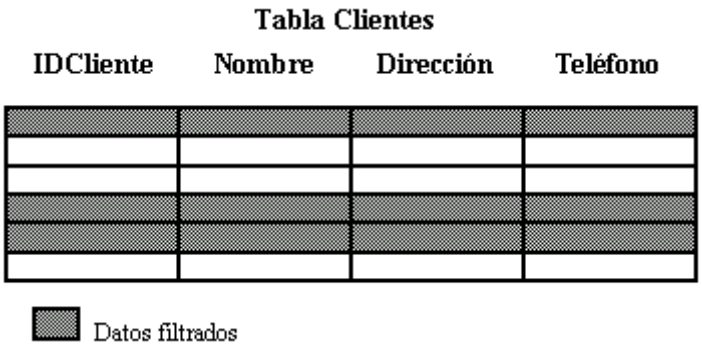


Figura 291. Filtro horizontal para un artículo.

Tipos de suscripción

Según la dirección en la que circule una publicación entre un suscriptor y un publicador existen los siguientes tipos de suscripción:

- Inserción. En este tipo de suscripción, la publicación es enviada desde el publicador hasta el suscriptor (el publicador inserta una publicación en el suscriptor). Es una suscripción recomendable cuando se necesitan enviar los cambios producidos en las publicaciones de la forma más rápida y fiable posible. Una suscripción de inserción puede ser configurada para que sea recibida por varios suscriptores al mismo tiempo. La Figura 292 muestra un esquema del funcionamiento de este tipo de suscripción.

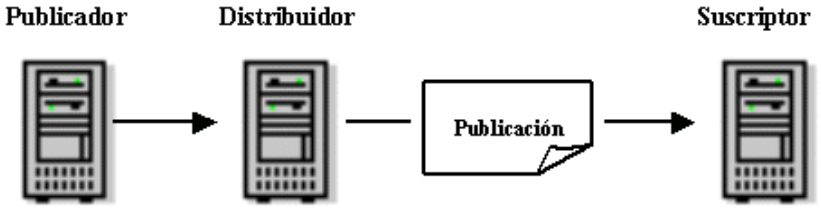


Figura 292. Suscripción de inserción.

- Extracción. Se trata de una suscripción creada por el suscriptor. En este caso, es el suscriptor el que realiza la petición de la publicación (extrae la información) al publicador. Debido a que un suscriptor puede realizar cambios en la publicación a la que está suscrito, puede enviar dicha publicación modificada al publicador para actualizar la información. El proceso por el cuál se actualiza la información en publicadores y suscriptores, de forma que en todos ellos existan los mismos datos se denomina sincronización. El esquema de funcionamiento se muestra en la Figura 293.

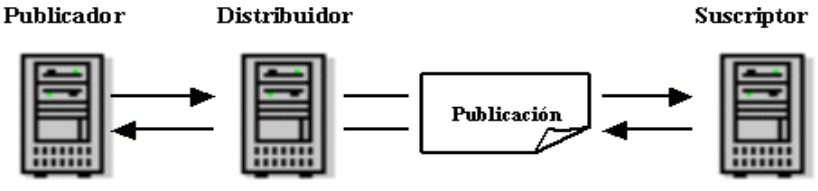


Figura 293. Suscripción de extracción.

Tipos de duplicación

Las diferentes formas en que una publicación puede ser transmitida entre publicador y suscriptor se conocen como tipos de duplicación. Seguidamente enumeramos los tipos disponibles.

- Instantáneas. En este tipo de duplicación, se toma una copia o instantánea de toda la información o publicación en un momento establecido y se envía al suscriptor. Como ventaja podemos destacar que liberamos al sistema de sobrecargas, al no tener que estar continuamente revisando la información para actualizarla en el suscriptor, sino sólo en determinados momentos.

  Esta misma ventaja que acabamos de comentar también supone un inconveniente, ya que el suscriptor no tendrá los datos actualizados en todo momento, por lo que este tipo de duplicación es recomendable aplicarla en situaciones que no requieran los datos más recientes en todo momento.

  Debido a que se envía toda la publicación al suscriptor, hemos de tener en cuenta que si el contenido de los datos de la publicación es muy grande, puede ocasionar problemas de rendimiento al sistema, por lo que es posible que debamos optar por otro tipo de duplicación.

- Transaccional. En este caso, el suscriptor se sincroniza sólo con los datos que han cambiado en el publicador. Si los componentes físicos del sistema lo permiten, la información modificada se envía muy rápidamente desde el publicador al suscriptor, por lo que este último dispone de información real casi en todo momento. Para conseguir tal nivel de optimización, este tipo de duplicación hace uso del registro de transacciones para supervisar los cambios producidos en la base de datos.

- Mezcla. La duplicación de mezcla se basa en la posibilidad de que tanto el publicador como los suscriptores puedan realizar cambios en los datos. Al efectuar el proceso de sincronización de información, todas las partes implicadas serán actualizadas con la información modificada de la que hasta el momento no disponían.

Agentes de duplicación

Para coordinar las diferentes operaciones, el sistema de duplicación de SQL Server dispone de los denominados *Agentes de duplicación*, que se encargan de la realización de los diversos tipos de duplicación, y que vemos a continuación:

- Instantáneas. El agente de instantáneas permite la sincronización de datos entre las tablas del publicador y suscriptor.

- Distribución. Este agente se encarga de enviar las instantáneas creadas en el distribuidor a todos los suscriptores.

- Mezcla. Se ocupa de mezclar los cambios de la información entre los distintos publicadores y suscriptores de la duplicación.

- Lector del registro. Tiene como misión revisar el registro de transacciones del publicador, y copiar las marcadas para duplicación en la base de datos distribution. Posteriormente, todas esas transacciones se aplicarán a las bases de datos de los suscriptores.

Implementación física de una estrategia de duplicación

Según nuestras necesidades de duplicación de datos, debemos diseñar una estrategia de instalación para los servidores físicos que van a compartir la información y el modelo de duplicación a utilizar. A continuación se comentan los modelos de implantación más comunes.

Publicador y distribuidor central con uno o varios suscriptores

Este modelo de duplicación (Figura 294), se basa en la existencia de un servidor central que actúe como publicador y distribuidor al mismo tiempo, y uno o varios servidores que realizarán las tareas de suscriptor.

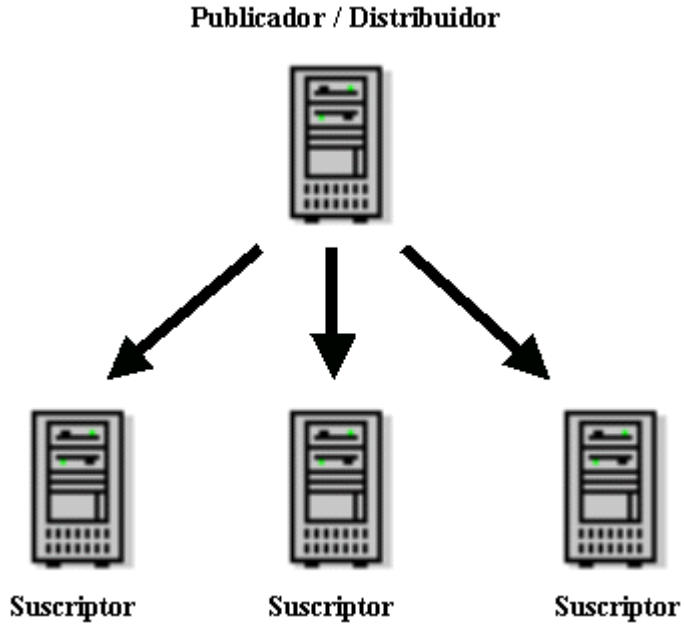


Figura 294. Duplicación con un publicador / distribuidor y varios suscriptores.

Escenario de implantación

Tomemos como ejemplo el caso de una empresa que dispone de una sede central y varias delegaciones. Trimestralmente, los directivos de cada delegación necesitan conocer la estadística de facturación total de un determinado producto que se vende en todas las delegaciones.

La respuesta a este problema sería crear en el servidor de la central un publicador y distribuidor, configurando los servidores de las delegaciones como suscriptores. De esta forma cada tres meses, mediante duplicación de instantáneas, se publicarían los datos necesarios a los suscriptores creados.

Suscriptor central con uno o varios publicadores y distribuidores

En esta situación se trata de tener una máquina que trabaja como suscriptor, mientras que existe un conjunto de servidores que realizan las tareas de publicadores y distribuidores de publicaciones para el suscriptor central. Ver la Figura 295.

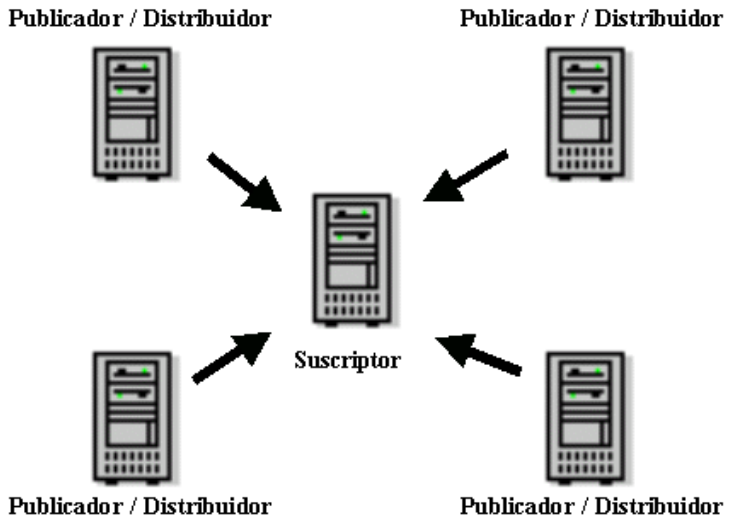


Figura 295. Duplicación con varios publicadores / distribuidores y varios suscriptores.

Escenario de implantación

Siguiendo con el ejemplo del anterior modelo, en el que una compañía vendía un mismo producto a través de sus delegaciones, supongamos que la sede central es la encargada de hacer los pedidos al proveedor de dicho producto, y para ello necesita saber cuando las existencias de las delegaciones llegan a una cantidad mínima en que sea necesario realizar un pedido.

La solución que podemos aplicar en este sentido es la siguiente: configurar como publicadores a las bases de datos de las delegaciones que contienen la tabla de existencias del producto, mientras que la base de datos de la sede central, que se ocupa de las existencias, se configura como suscriptor. Mediante duplicación transaccional, la central recibirá constantemente, puntual información sobre las existencias de las delegaciones.

Diversos publicadores / distribuidores y diversos suscriptores

El caso más variado, aquí existen múltiples servidores que realizan las funciones tanto de publicador como de suscriptor, dependiendo de si necesitan datos o los envían, véase la Figura 296.

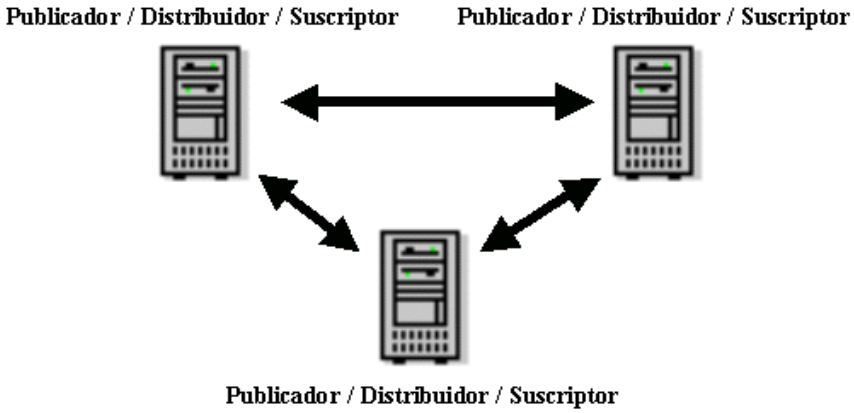


Figura 296. Duplicación con diversos publicadores / distribuidores / suscriptores.

Escenario de implantación

En este caso, el problema planteado consiste en una cadena de talleres, en la que cada taller dispone de un almacén de piezas. Cuando uno de los talleres se queda sin una determinada pieza, la solicita a otro que sí disponga de ella, consultando la correspondiente tabla de su base de datos.

Podría suceder que todos los talleres se quedaran sin una pieza en concreto, por lo que para solucionar este problema, se implanta una estrategia de duplicación con múltiples publicadores y suscriptores, en el que el tipo de duplicación sea transaccional o de mezcla.

Como resultado, todos los talleres conocerán las existencias totales de dicha pieza, evitando quedar desabastecidos.

Establecer un publicador y un distribuidor

Para poder utilizar las características de duplicación de SQL Server, en primer lugar debemos crear un publicador y un distribuidor que se ocupen de estas tareas, por lo que seleccionaremos desde el Administrador corporativo la opción de menú *Herramientas+Duplicación+Configurar publicar y suscriptores*, que iniciará el asistente para llevar a cabo esta labor, como vemos en la Figura 297.

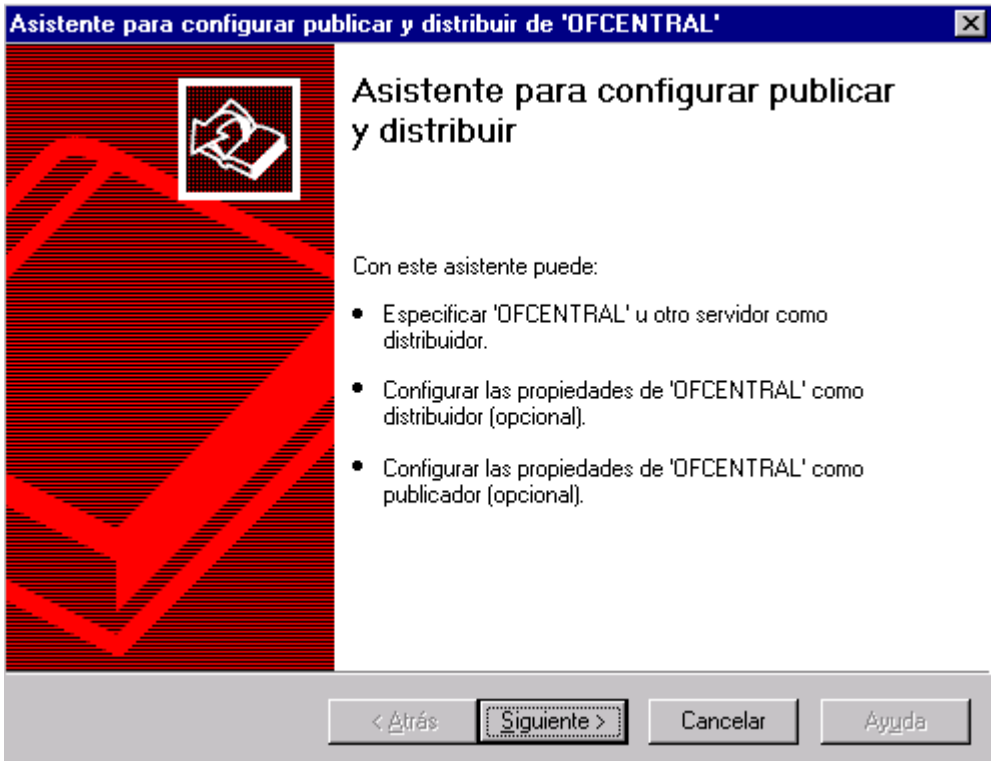


Figura 297. Asistente para la configuración de publicadores y distribuidores.

Como labor inicial, debemos especificar cuál de los servidores con los que podemos conectar estableceremos como distribuidor, ver Figura 298.

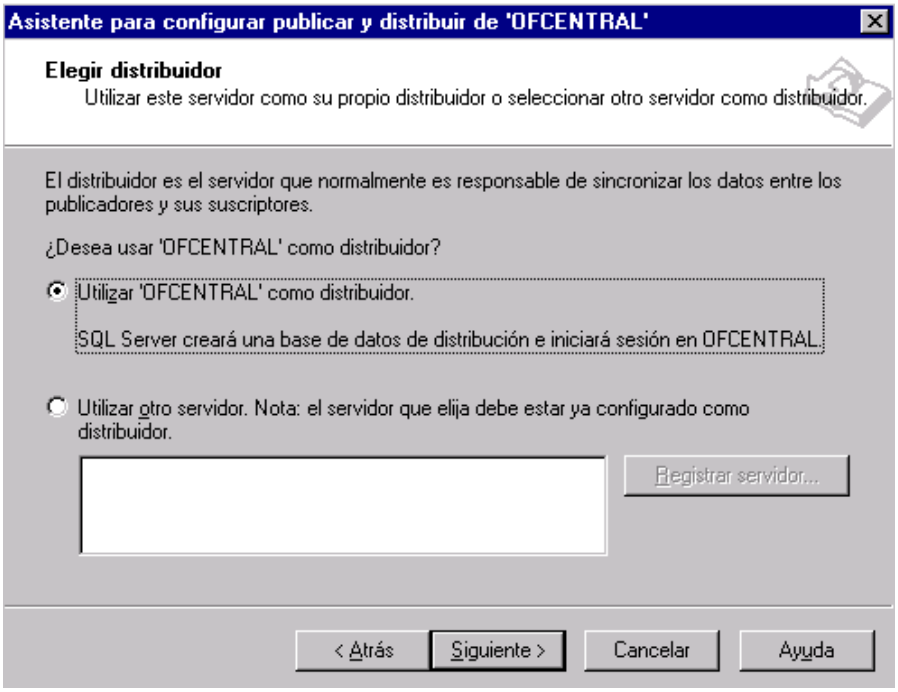


Figura 298. Elegir servidor de distribución.

A continuación, deberemos configurar el distribuidor con los valores predeterminados o estableciendo nosotros la configuración, ver Figura 299.

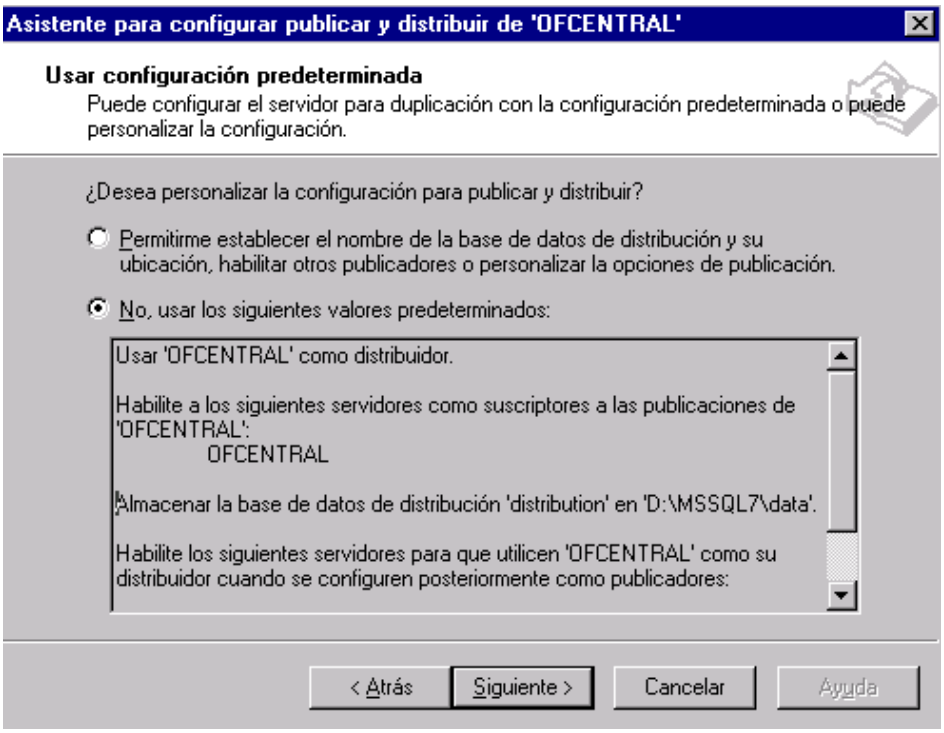


Figura 299. Selección del tipo de configuración para el distribuidor.

Completados estos pasos, habremos terminado la configuración del distribuidor, por lo que finalizaremos este asistente, ver Figura 300.

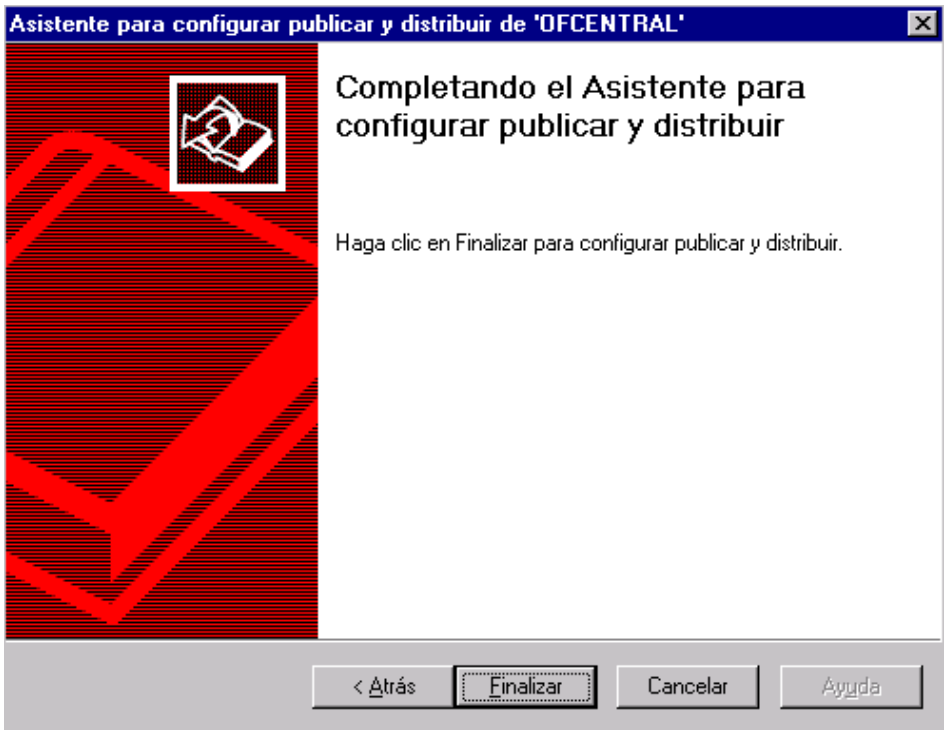


Figura 300. Finalización del asistente para configuración de un distribuidor.

A continuación, SQL Server iniciará el proceso de creación de la base de datos de distribución, mostrándonos un mensaje de confirmación al finalizar. Por último, se nos comunicará mediante la ventana de la Figura 301, que el servidor elegido ya ha sido establecido como distribuidor.

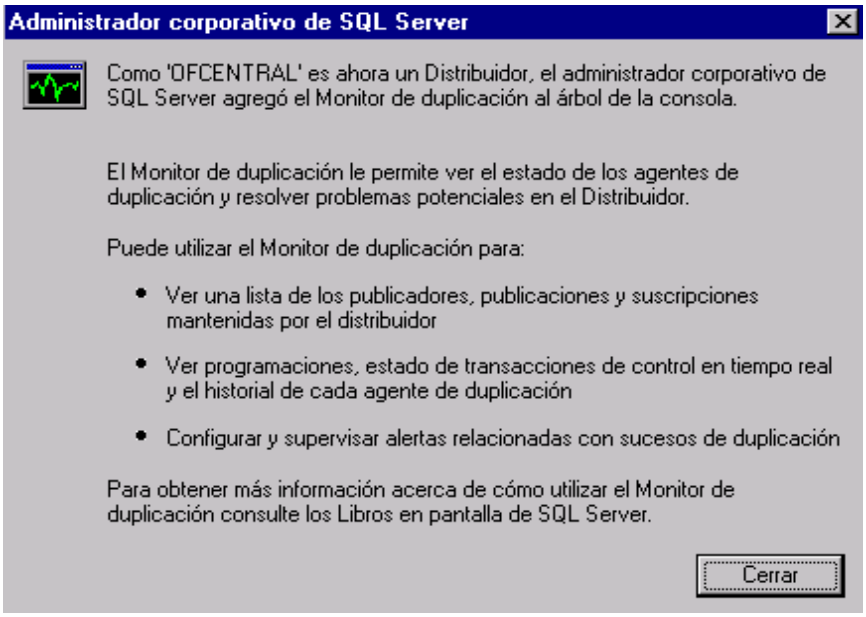


Figura 301. Aviso de creación del Supervisor de duplicación.

En este momento dispondremos de un nuevo elemento denominado *Supervisor de duplicación* en el árbol de la consola del servidor, Figura 302.

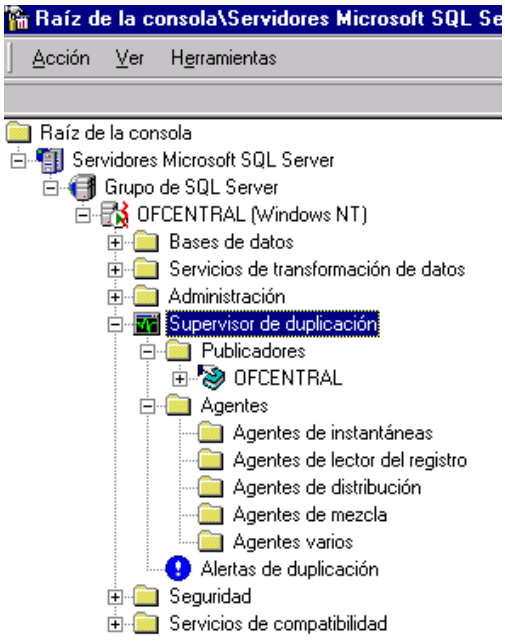


Figura 302. Supervisor de duplicación en el árbol de consola del Administrador corporativo.

Como el lector habrá observando, se ha creado al mismo tiempo un publicador situado en la carpeta Publicadores de este *Supervisor de duplicación*. La frecuencia con la que se actualizan los datos en este supervisor puede ser modificada si hacemos clic con el botón derecho sobre *Supervisor de duplicación* y seleccionamos del menú contextual la opción *Tasa de actualización y configuración*, que nos mostrará la ventana de la Figura 303.

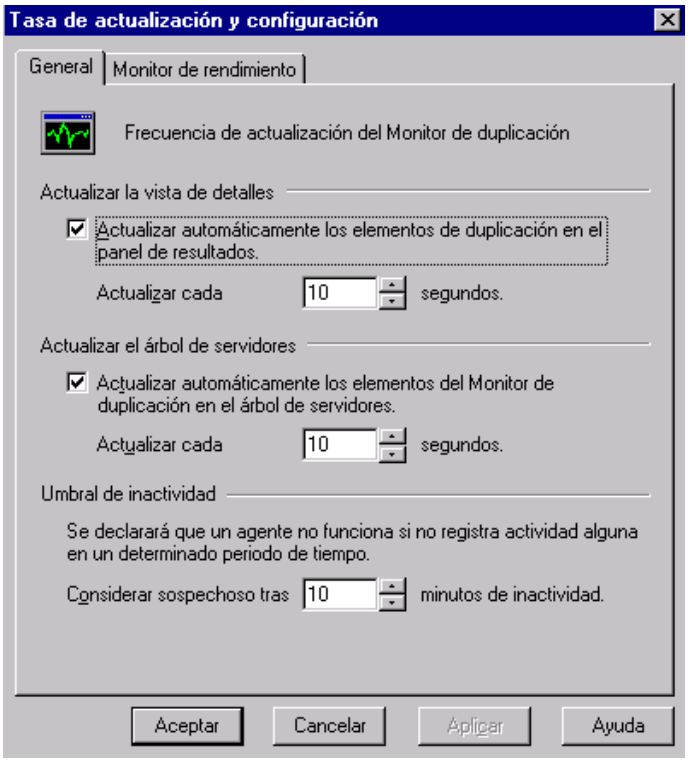


Figura 303

27

Publicaciones y suscripciones en duplicación de datos

Creación de publicaciones

Establecidos tanto el distribuidor como el publicador, el siguiente paso en nuestro proceso de implantación de un sistema de duplicación pasa por crear una o varias publicaciones, que usaremos para enviar información a los suscriptores.

Para ello, seleccionaremos en el Administrador corporativo la opción de menú *Herramientas + Duplicación + Crear y administrar publicaciones*, que nos mostrará una ventana con la lista de bases de datos y publicaciones disponibles hasta el momento, Figura 304.

Evidentemente, sólo se mostrarán bases de datos, ya que no hemos creado todavía ninguna publicación.

Para crear una publicación, seleccionaremos la base de datos que queremos publicar (para este ejemplo será Contabilidad) y pulsaremos el botón *Crear publicación*, que ejecutará el asistente para la creación de publicaciones, cuya pantalla inicial nos describe algunas de sus características, ver Figura 305.

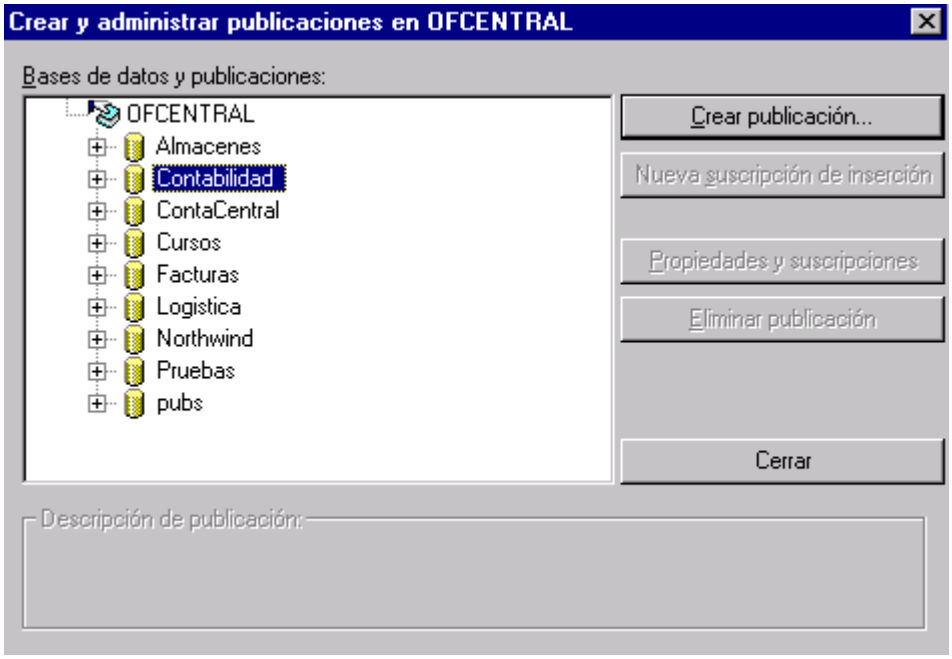


Figura 304. Ventana para la creación de publicaciones.

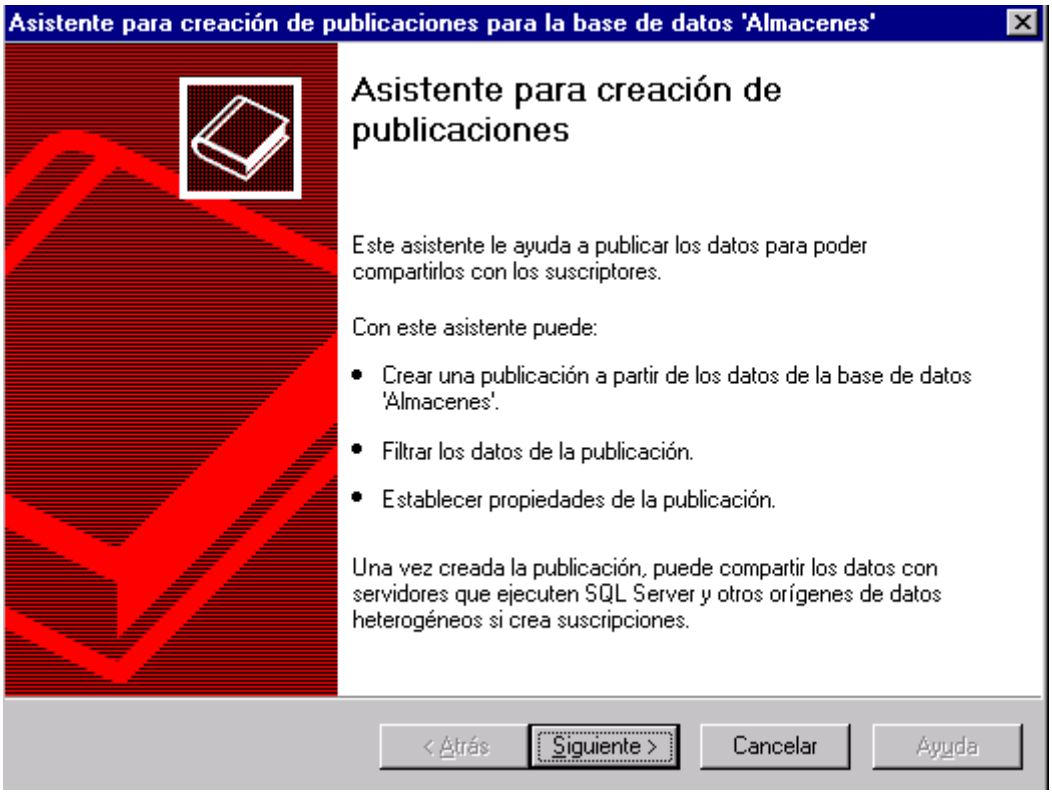


Figura 305. Asistente para la creación de publicaciones.

El primer paso de este asistente nos solicita que establezcamos el tipo de publicación a crear, que para este ejemplo será una publicación de instantáneas, como muestra la Figura 306.

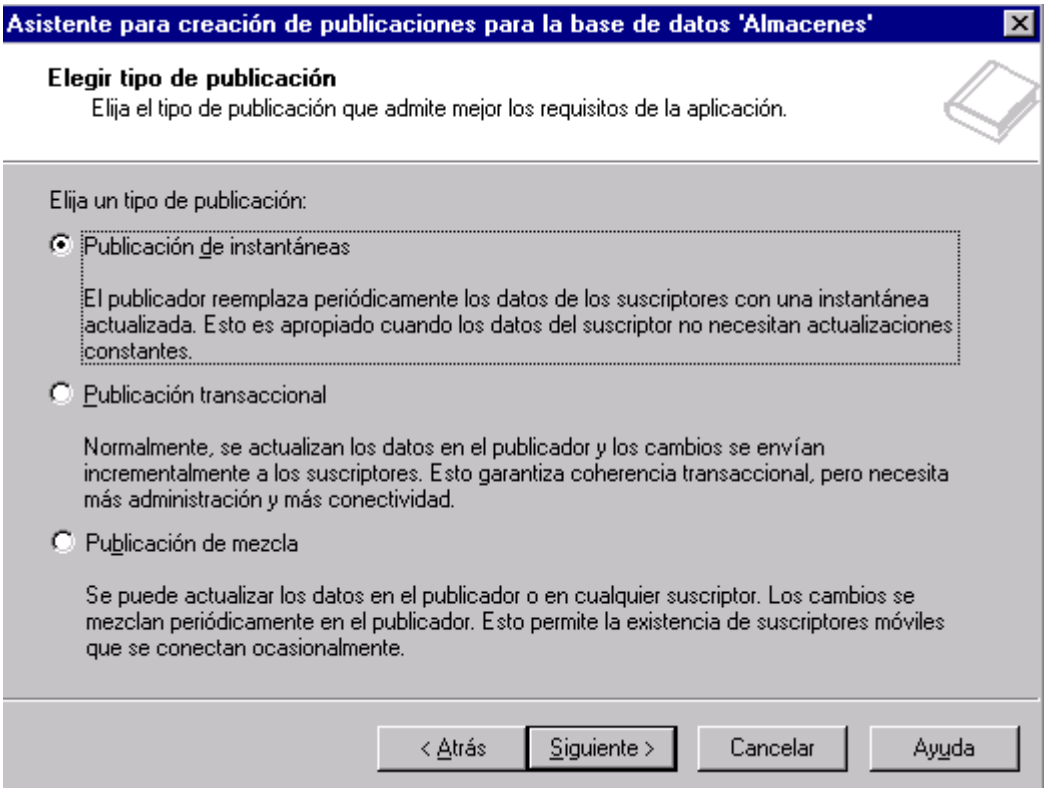


Figura 306. Elección del tipo de publicación a crear.

En el siguiente paso debemos indicar el tipo de suscripción que admitirá la publicación en cuanto al modo de actualización, Figura 307.

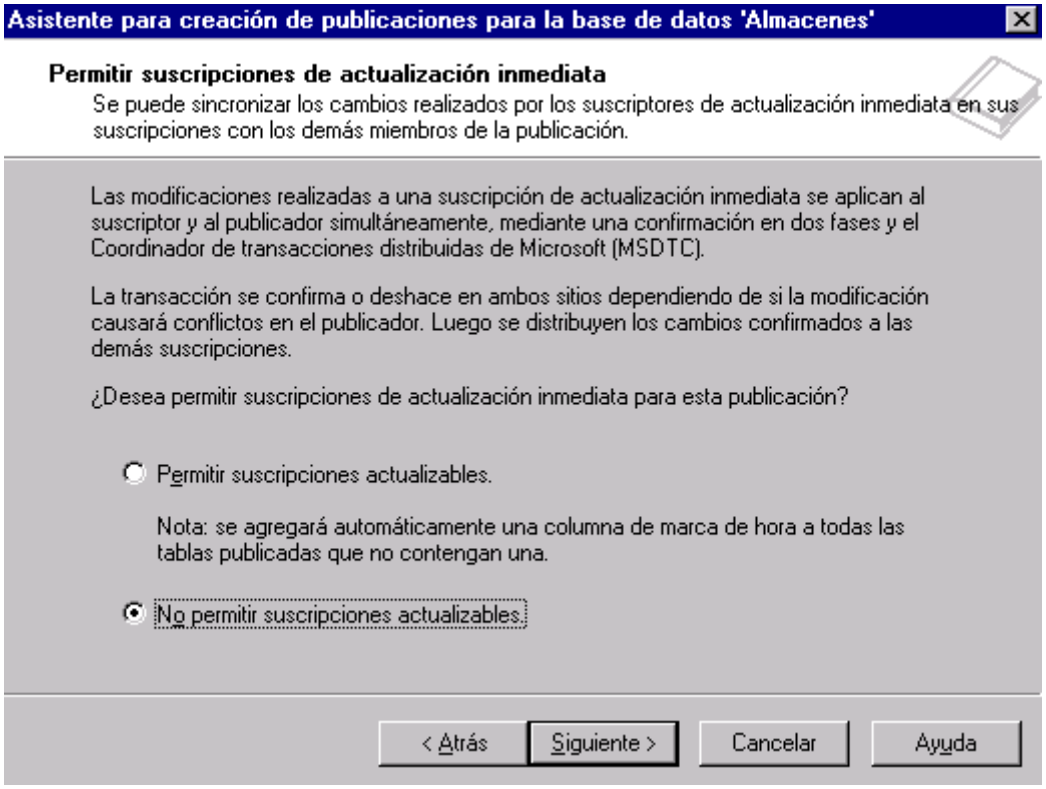


Figura 307. Selección del tipo de suscripción.

A continuación indicaremos si todos los suscriptores de la publicación serán servidores SQL Server o serán orígenes de datos heterogéneos, ver Figura 308. En el caso de que todas las fuentes de datos sean SQL Server, se empleará el formato de datos nativo de SQL Server que acelerará el proceso de duplicación.

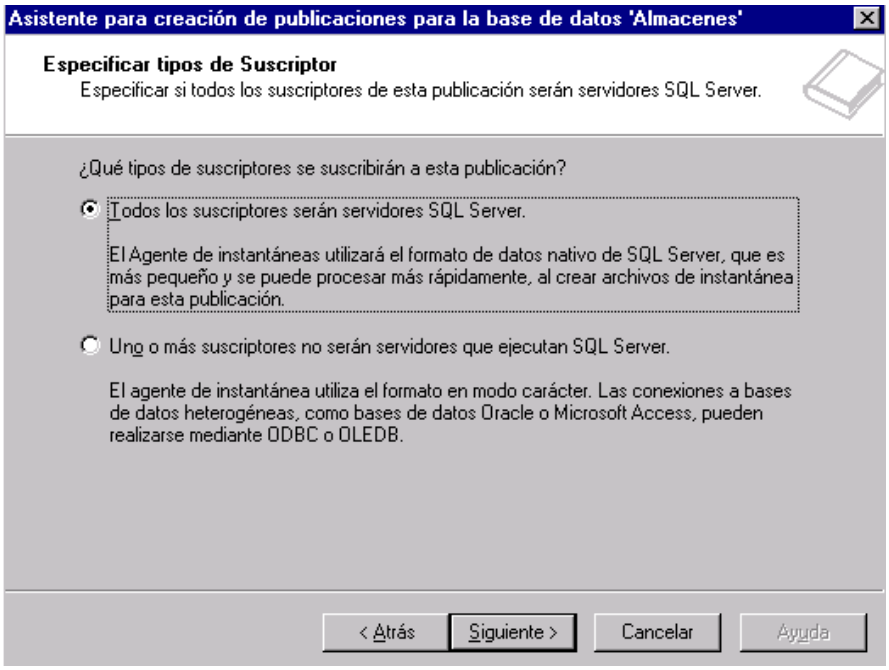


Figura 308. Especificación del tipo de datos del suscriptor.

En el siguiente paso, seleccionaremos las tablas y procedimientos almacenados de la base de datos que pasarán a ser los artículos de la publicación, Figura 309.

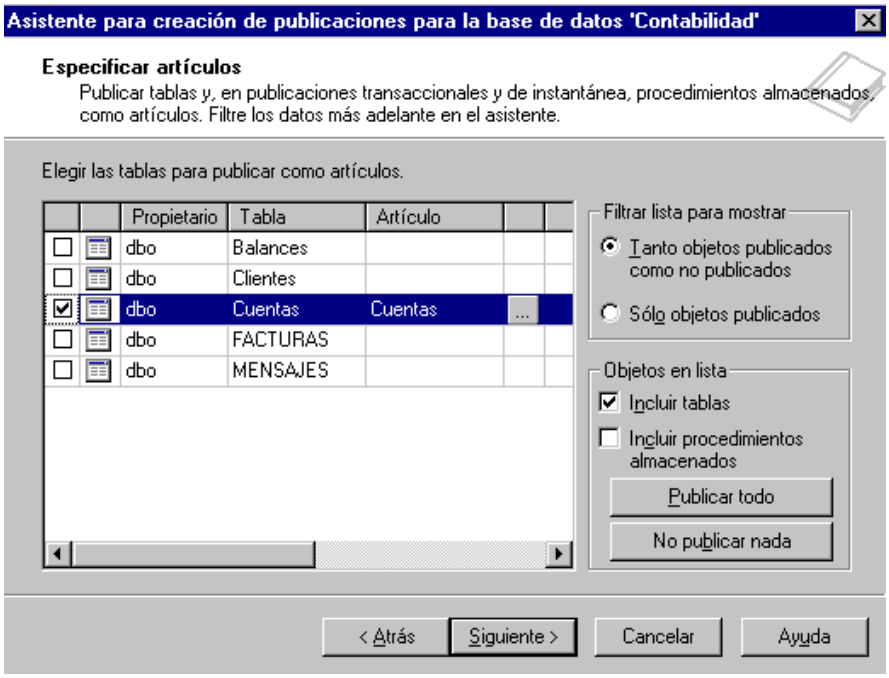


Figura 309. Selección de artículos para la publicación.

Pulsando el botón con puntos suspensivos que aparece al lado de la columna Artículo, accederemos a una ventana de propiedades del artículo (Figura 310), en la que podremos especificar entre otras, el nombre del artículo, el nombre con que se creará la tabla en la base de datos destino, el modo de copia de sus índices, etc.

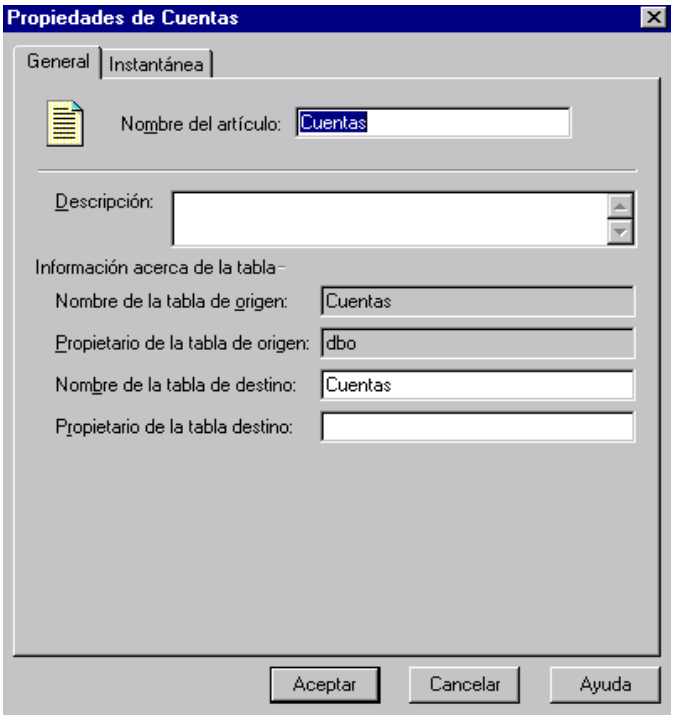


Figura 310. Propiedades del artículo.

Continuaremos asignando un nombre y descripción para la publicación, como vemos en la Figura 311.

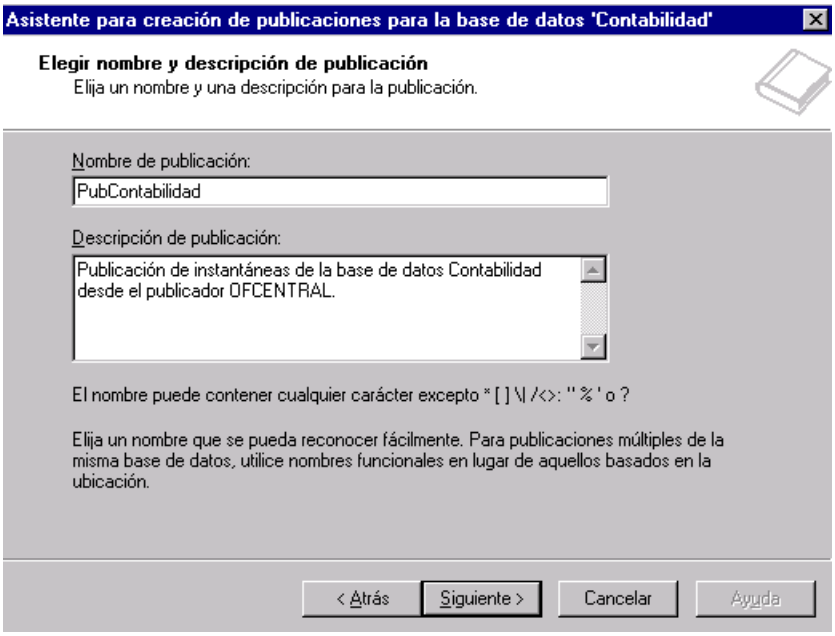


Figura 311. Establecer el nombre de la publicación.

Seguidamente indicaremos si queremos crear filtros de datos para los artículos de la publicación, o bien publicar toda la información de los artículos, ver Figura 312.

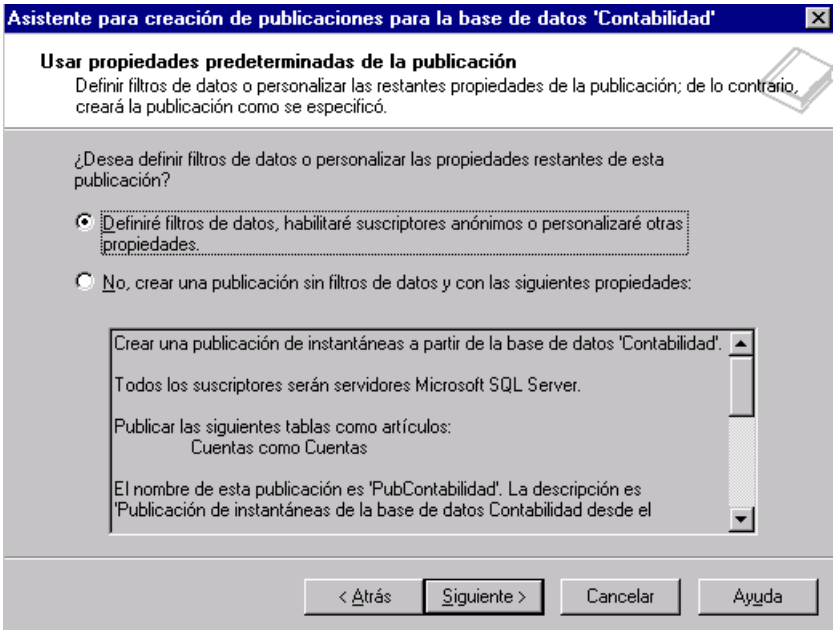


Figura 312. Creación de filtros de datos.

La opción más sencilla es seleccionar la creación de una publicación sin filtros, que incluya todos los datos. Para este ejemplo, sin embargo, pulsaremos el botón de opción que nos permite la definición de filtros, de forma que ilustremos como crear tanto un filtro vertical como uno horizontal.

Tras elegir esta opción, se pide al usuario la confirmación para crear los filtros en los artículos, ver Figura 313.

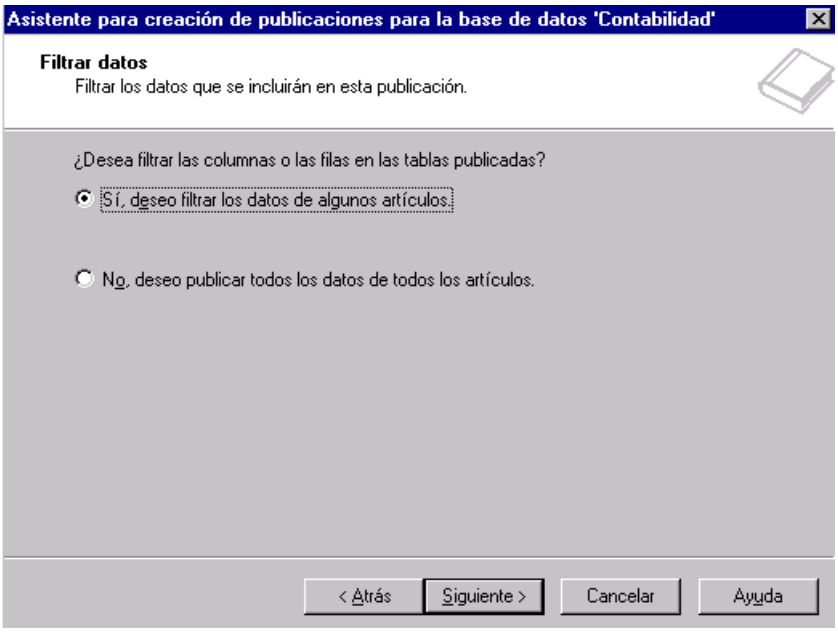


Figura 313. Confirmación para la creación de filtros.

Tras responder afirmativamente, en el siguiente paso procederemos a establecer el filtro vertical o dicho de otro modo, seleccionar las columnas que queremos publicar y las que no serán accesibles por el suscriptor. En la Figura 314 se muestra la selección de columnas realizadas para el artículo.

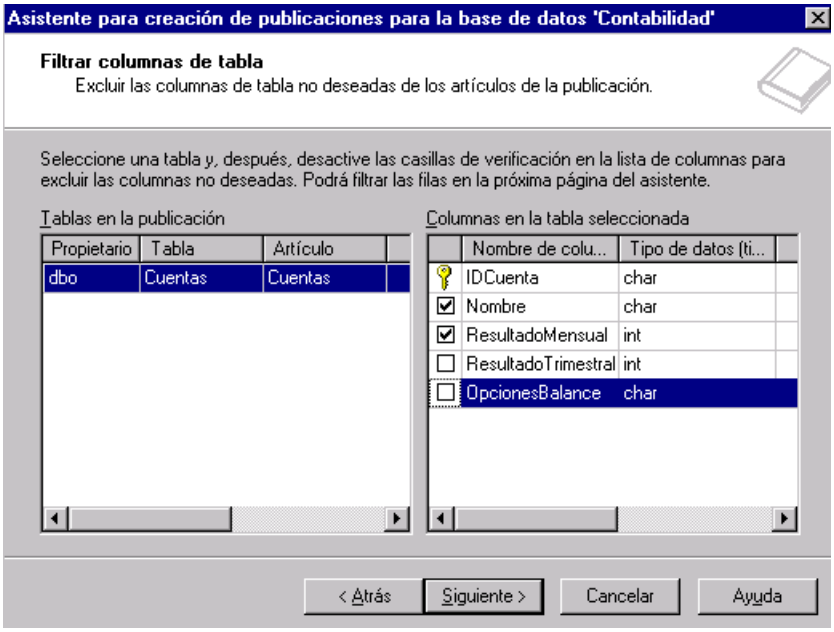


Figura 314. Creación de un filtro vertical o de columnas.

Las columnas publicadas serán las que estén marcadas en el panel derecho de este paso del asistente.

Seguidamente tenemos la posibilidad de establecer el filtro horizontal, o la selección de filas que podrá ver el suscriptor, Figura 315.

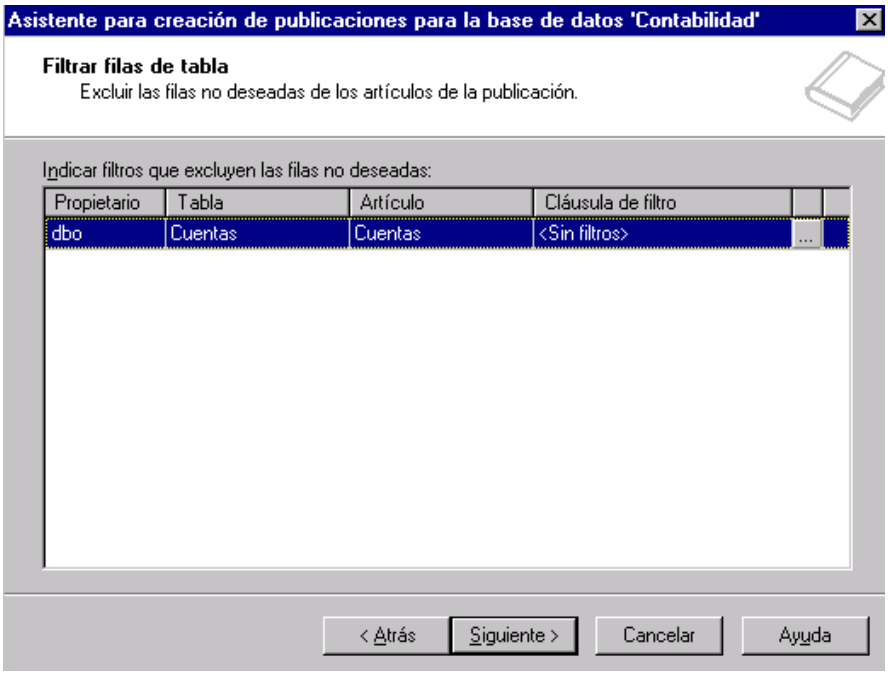


Figura 315. Filtro horizontal.

Pulsando el botón situado junto a la columna *Cláusula de filtro*, podremos establecer la sentencia SQL de selección de filas, como se muestra en la Figura 316.

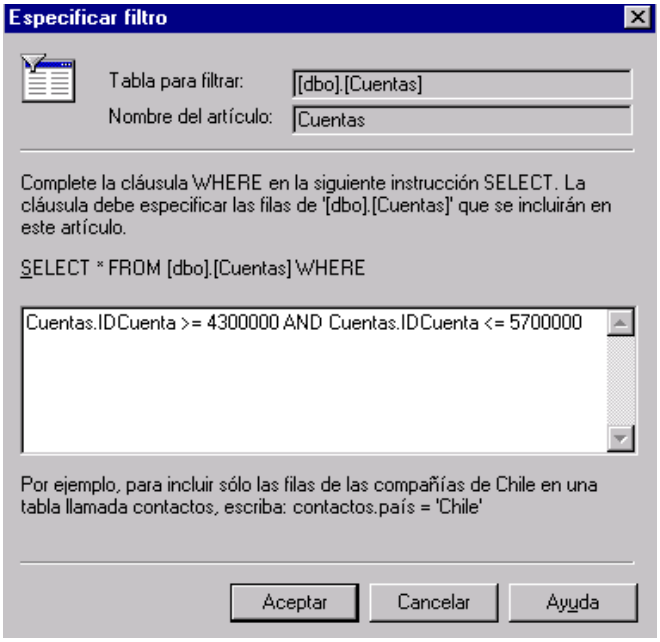


Figura 316. Sentencia de selección de filas para el filtro horizontal.

En el próximo paso especificaremos qué tipo de suscriptores podrán acceder a los datos de la publicación, ver Figura 317.

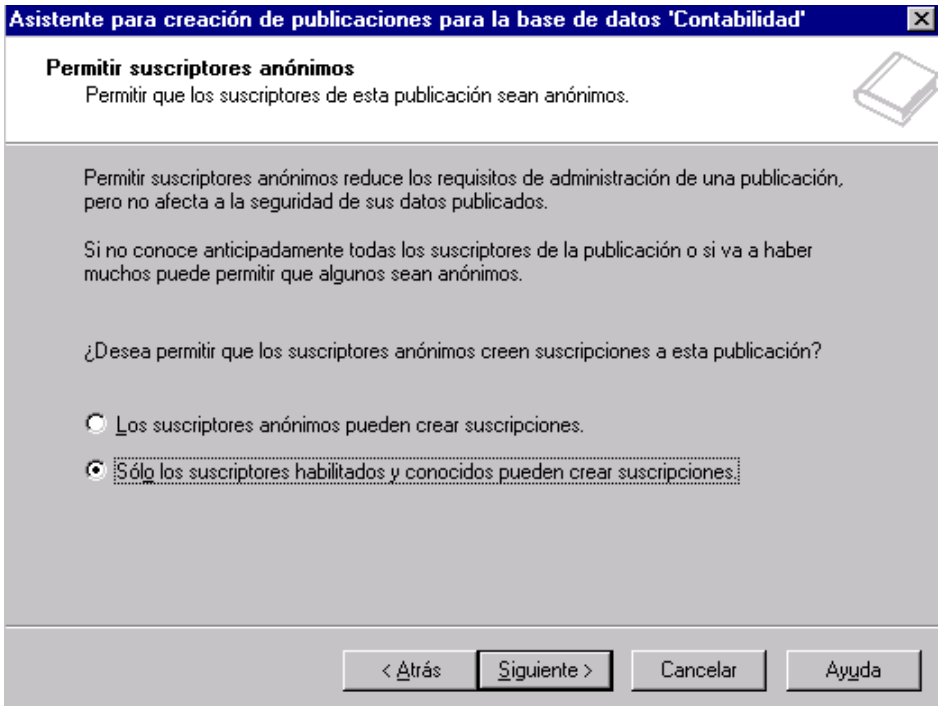


Figura 317. Selección del tipo de suscriptores para la publicación.

A continuación estableceremos la frecuencia de ejecución del *Agente de instantáneas*, y si se deberá crear una instantánea inicial, como vemos en la Figura 318.

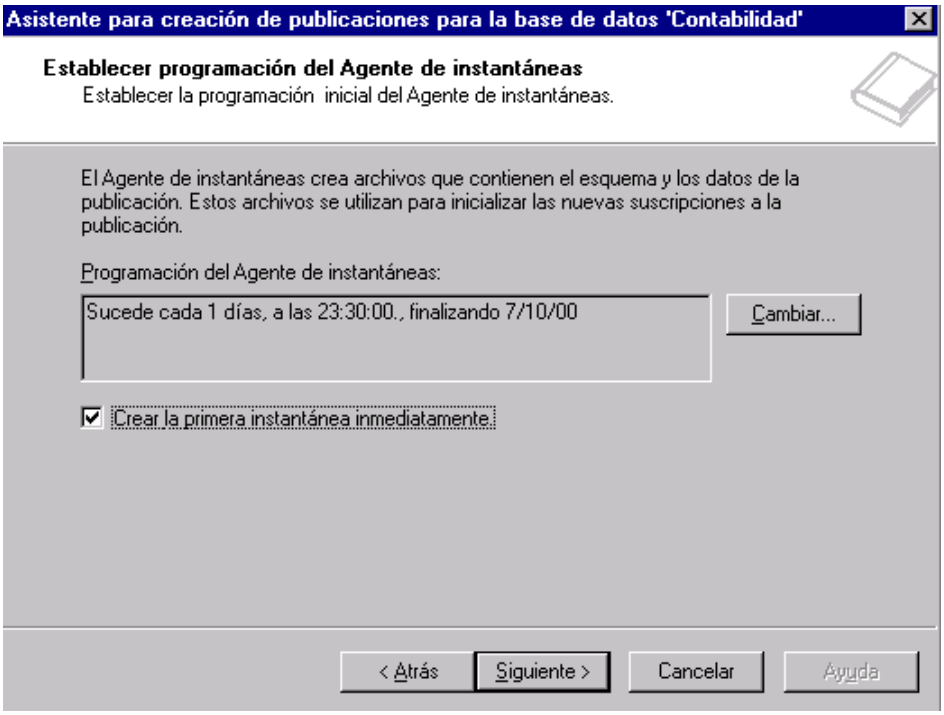


Figura 318. Valores de ejecución del Agente de instantáneas.

Y por fin llegamos al último paso de este asistente, tras el cual se creará la publicación con los valores que hemos asignado en todos los anteriores pasos, Figura 319.

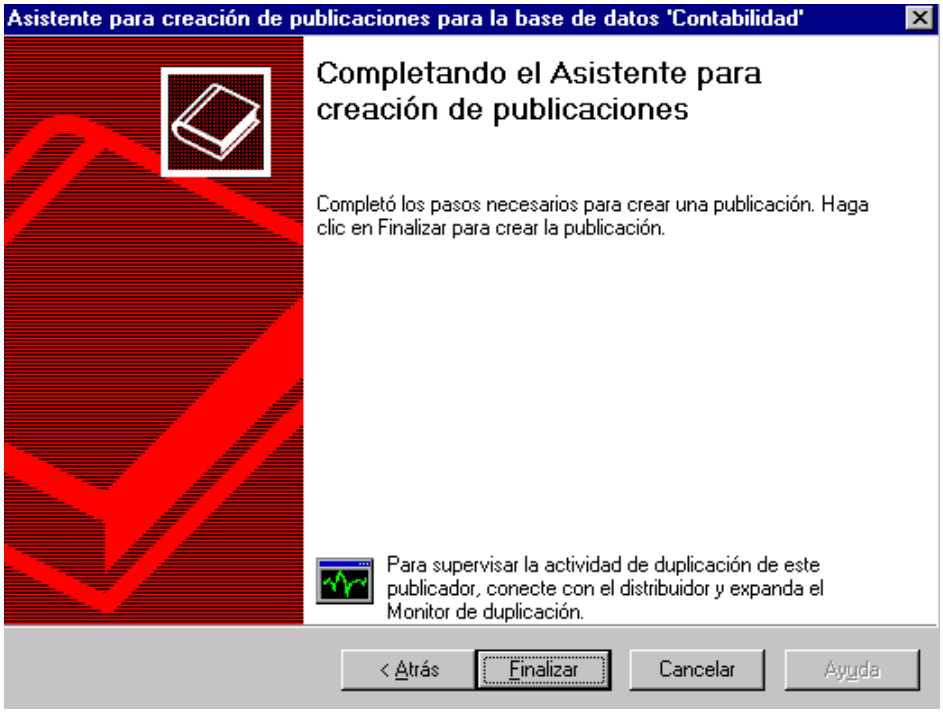


Figura 319. Final del asistente para creación de publicaciones.

La ventana de administración de publicaciones mostrará ahora la nueva publicación creada, como vemos en la Figura 320.

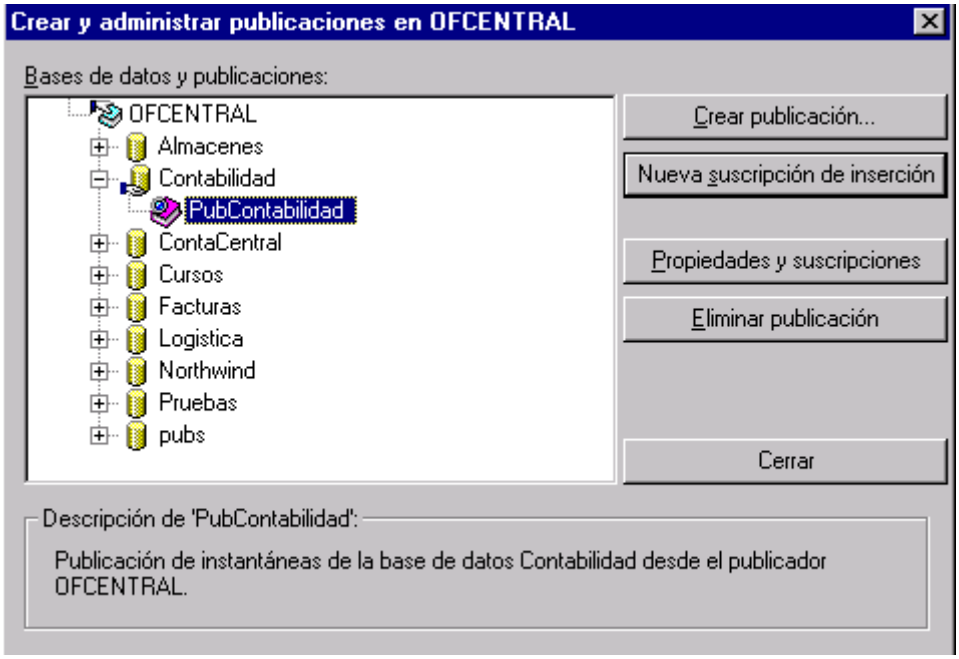


Figura 320. Nueva publicación creada en la ventana de publicaciones.

Igualmente, en la base de datos Contabilidad dispondremos de una nueva carpeta llamada Publicaciones, ver Figura 321.

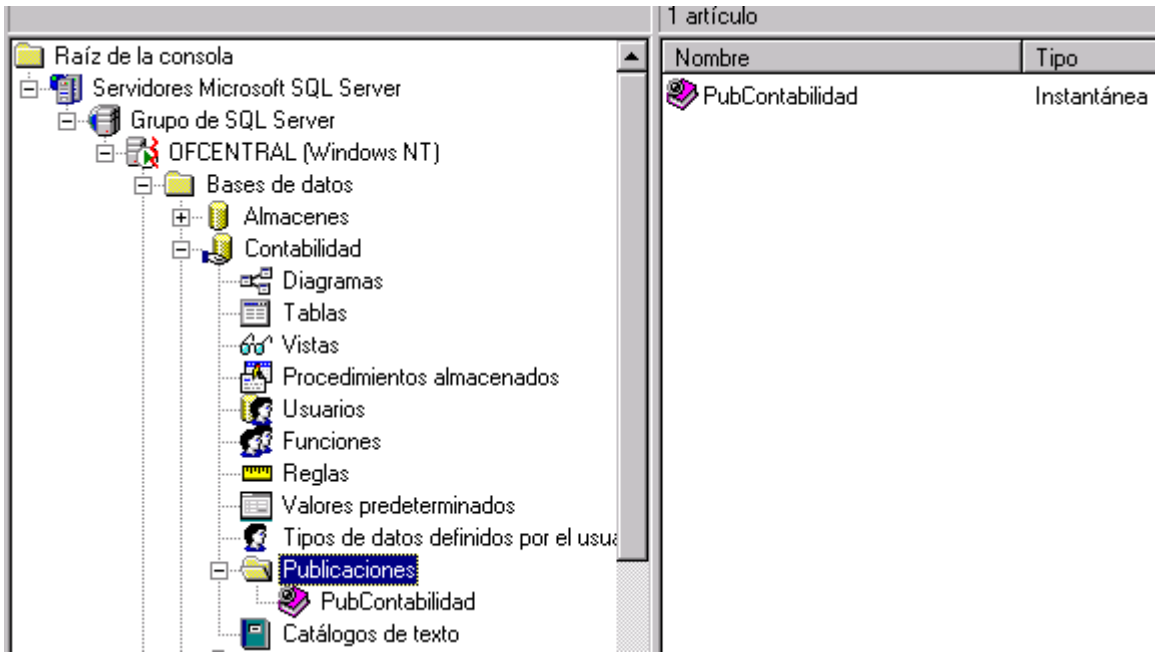


Figura 321. Carpeta Publicaciones de una base de datos.

Creación de suscripciones

Como eslabón final en la cadena de duplicación, sólo nos resta crear uno o varios suscriptores que reciban la información de la publicación creada.

En este ejemplo vamos a crear un suscriptor de inserción, que como ya explicamos, consiste en el envío de información desde el publicador al suscriptor.

Comenzaremos seleccionando en el menú del Administrador corporativo, la opción *Herramientas + Duplicación + Suscripciones de inserción a otros*, que abrirá la ventana de administración de publicaciones. Desplegando la base de datos Contabilidad, seleccionaremos la publicación que acabamos de crear y pulsaremos el botón *Nueva suscripción de inserción*, que iniciará el asistente para crear suscripciones de inserción, como muestra la Figura 322.

En el primer paso de este asistente, seleccionaremos un servidor configurado como suscriptor, en el que vamos a crear la suscripción, ver Figura 323.

En el siguiente paso, seleccionaremos la base de datos destino que actuará como suscriptora de la publicación. Pulsando el botón *Examinar bases de datos*, se abrirá una caja de diálogo con una lista de las bases de datos disponibles, Figura 324.

A continuación especificaremos la frecuencia de actualización de los datos desde el publicador al suscriptor, Figura 325.

El próximo paso (Figura 326) nos indica que se van a crear los esquemas de datos y publicación en el suscriptor, dándonos opción a la creación de una instantánea inicial al crear la suscripción.

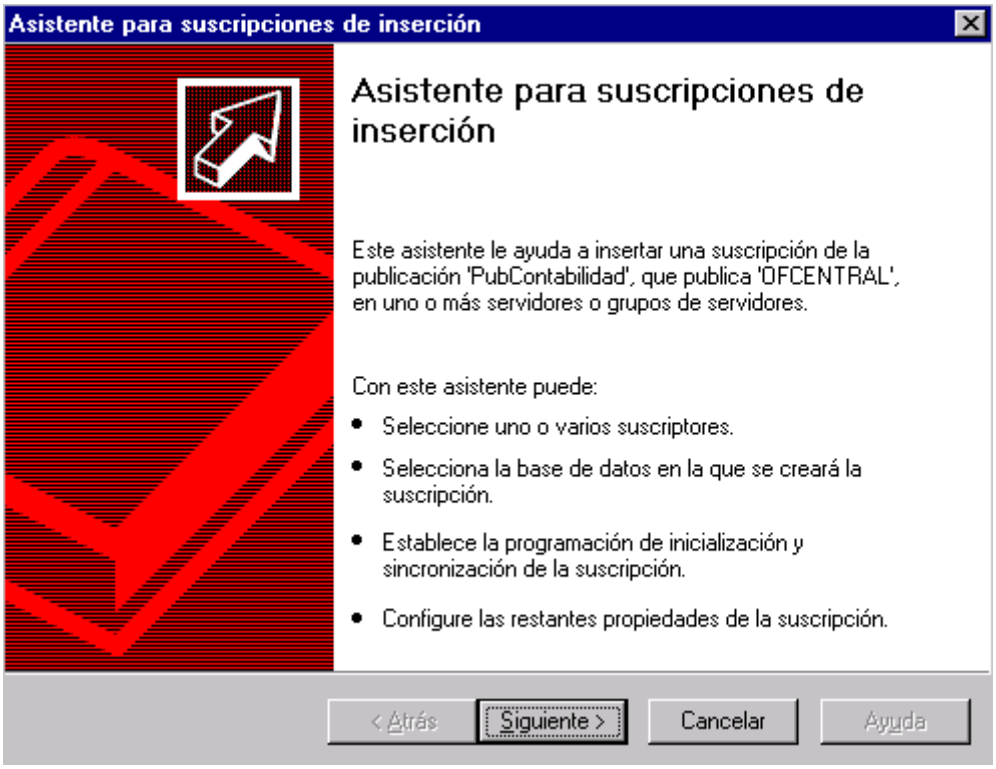


Figura 322. Asistente para suscripciones de inserción.

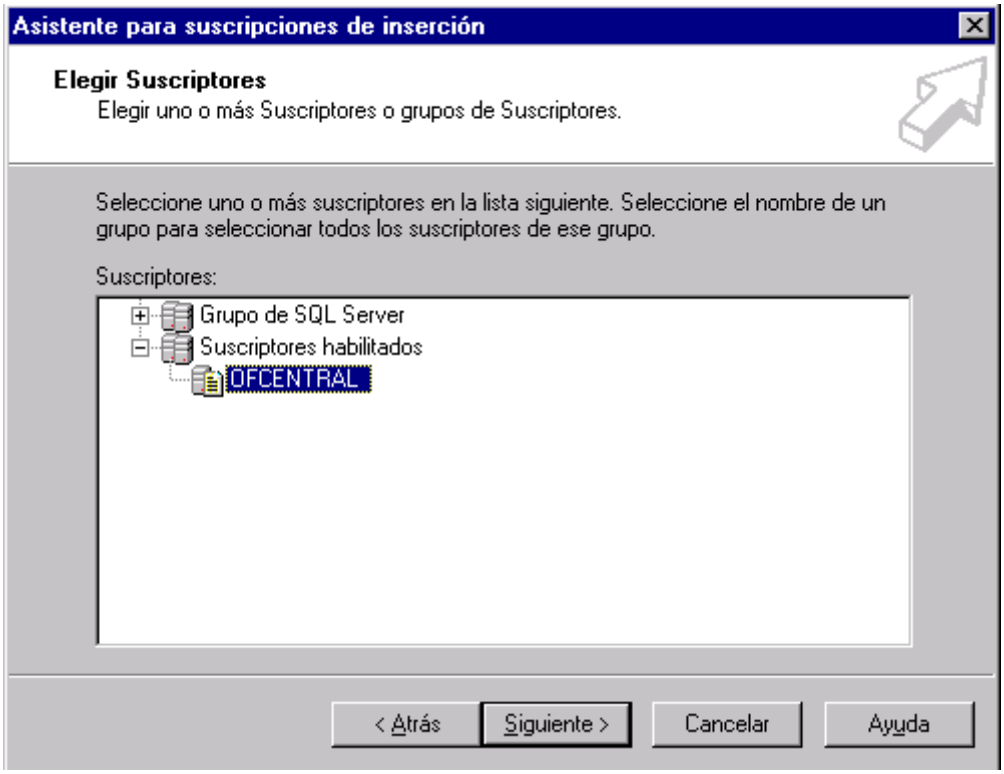


Figura 323. Selección del servidor suscriptor.

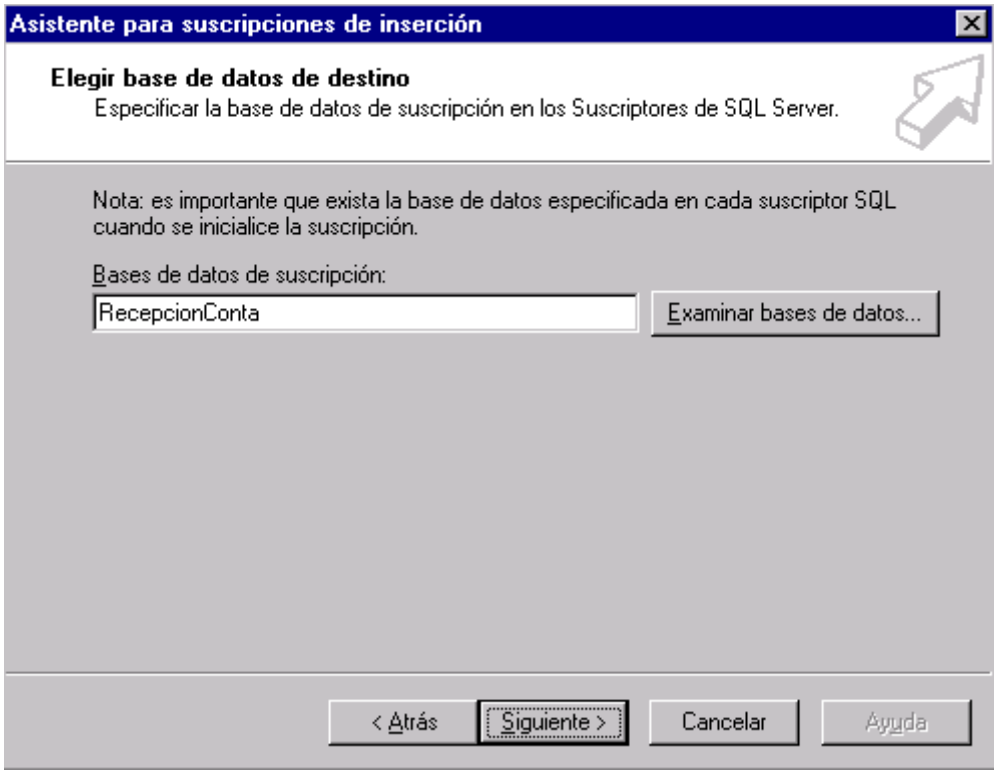


Figura 324. Selección de la base de datos destino de la suscripción.

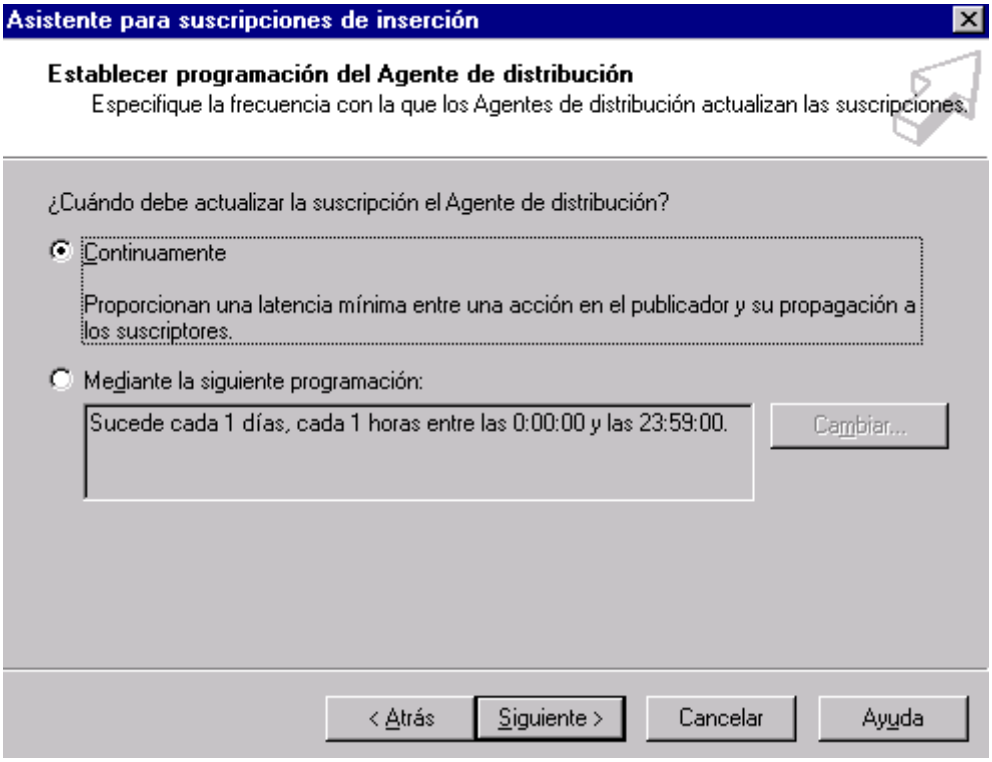


Figura 325. Frecuencia de actualización para la suscripción.

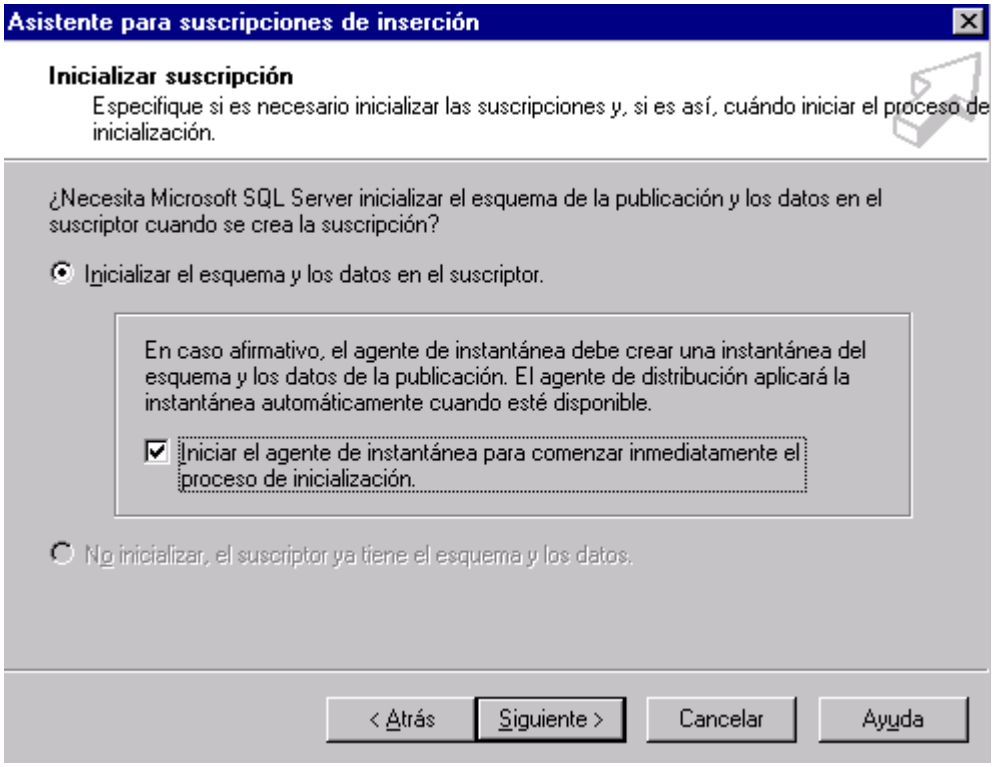


Figura 326. Opciones de inicialización de la suscripción.

Para la creación de la suscripción, es necesario que ciertos servicios de SQL Server estén en funcionamiento. En este paso, que vemos en la Figura 327, se indican los servicios necesarios y su estado.

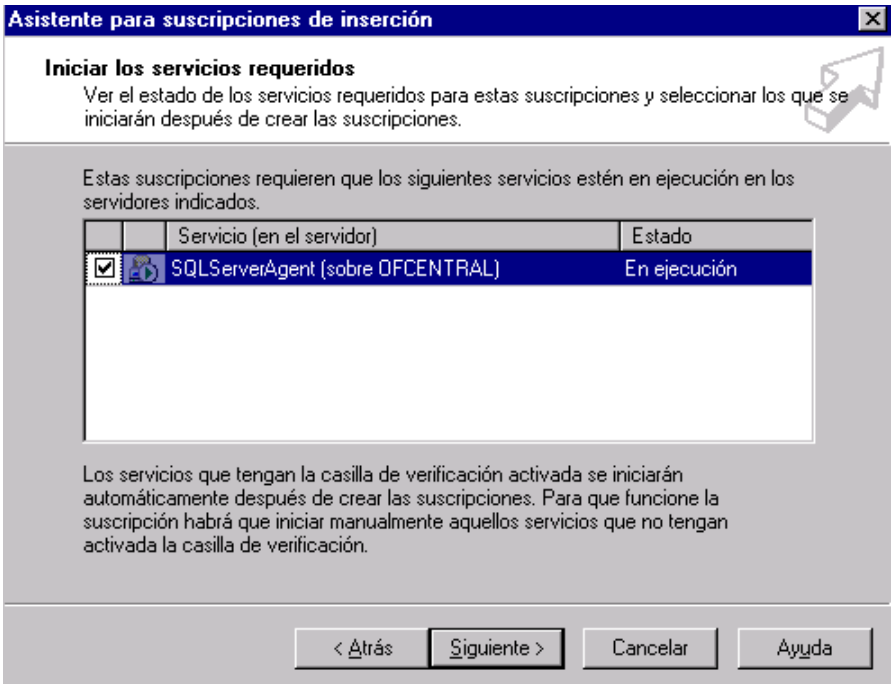


Figura 327. Servicios necesarios para la suscripción.

Completados todos los pasos, aparece la pantalla final del asistente, en la que al pulsar el botón Finalizar, se creará la suscripción según los valores que hemos indicado. Ver Figura 328.

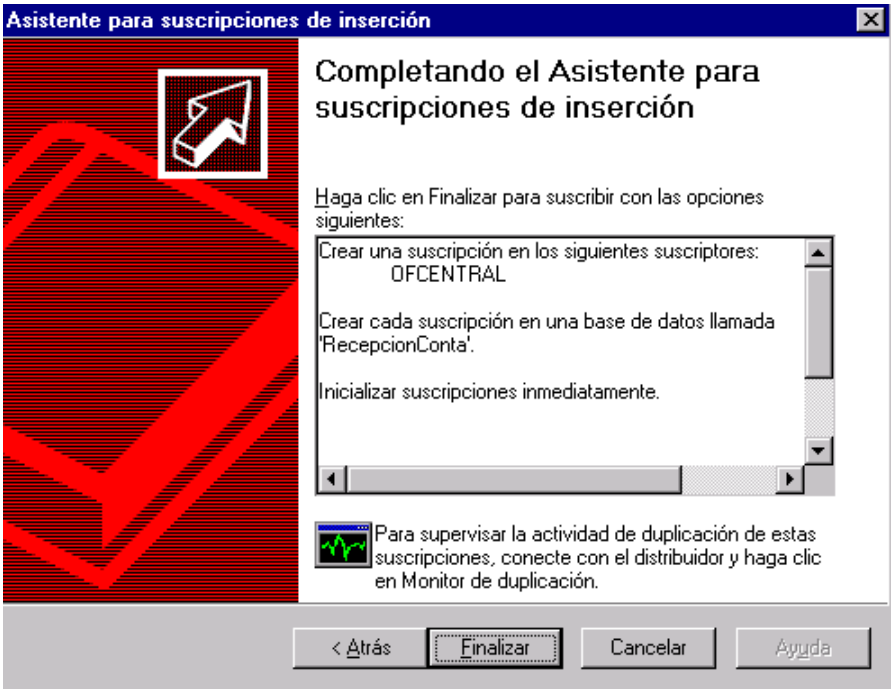


Figura 328. Final del asistente para creación de suscripciones.

El resultado en este ejemplo, será la creación de una replica de la tabla Cuentas en la base de datos RecepciónConta.

En el caso de que necesitemos crear una suscripción de extracción, seleccionaremos en el Administrador corporativo la opción de menú *Herramientas+Duplicación+Suscripción de extracción a <ServidorSQLServer>*, que nos mostrará la ventana de la Figura 329 con las suscripciones existentes y en la que pulsaremos el botón *Nueva suscripción de extracción*, que iniciará el asistente para suscripciones de extracción, y en el que a través de los mismos pasos que el asistente de inserción, nos creará una suscripción para extraer información de un publicador.

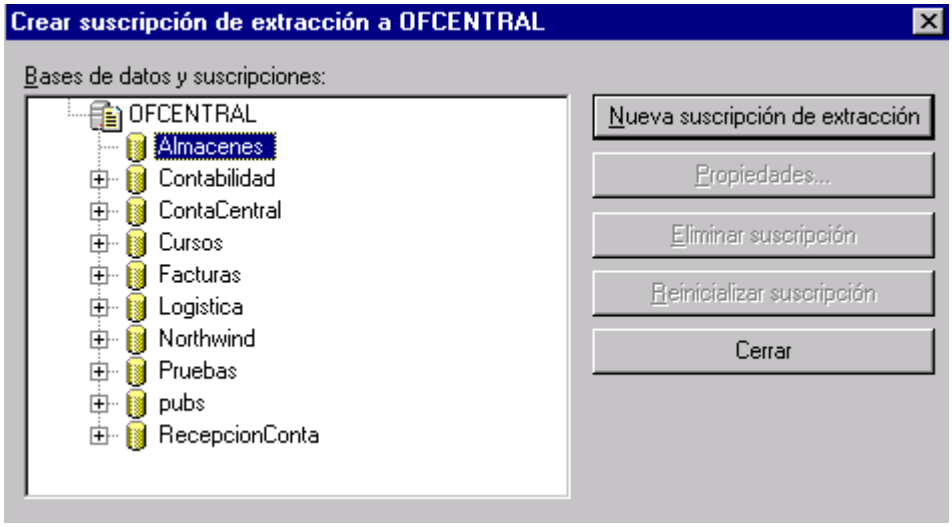


Figura 329