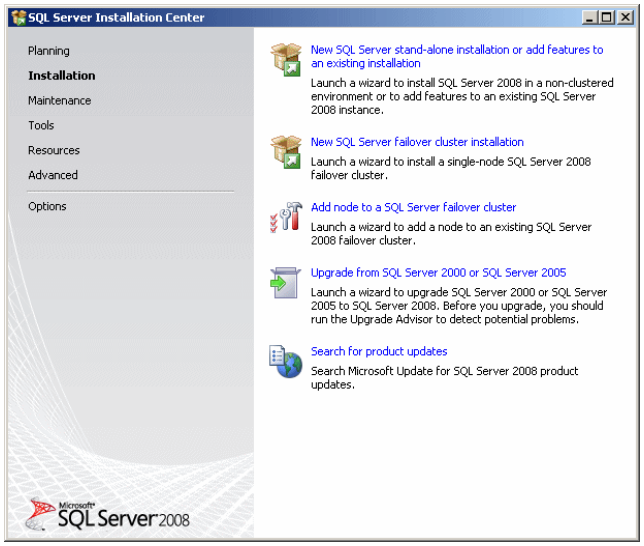
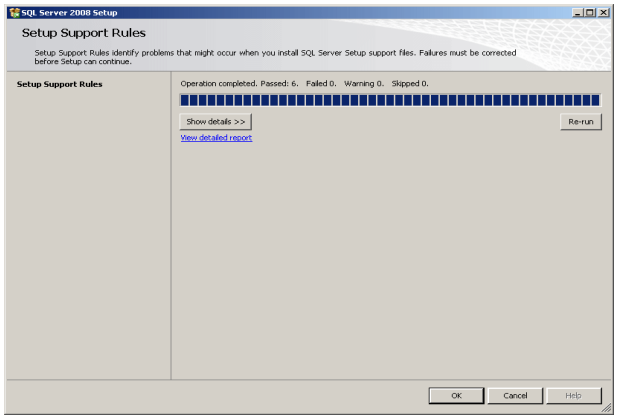
1. Para iniciar la instalación, poner el Microsoft SQL Server 2008 en la unidad de DVD. Una ventana aparecerá:

2. En la izquierda, haga clic en Instalación

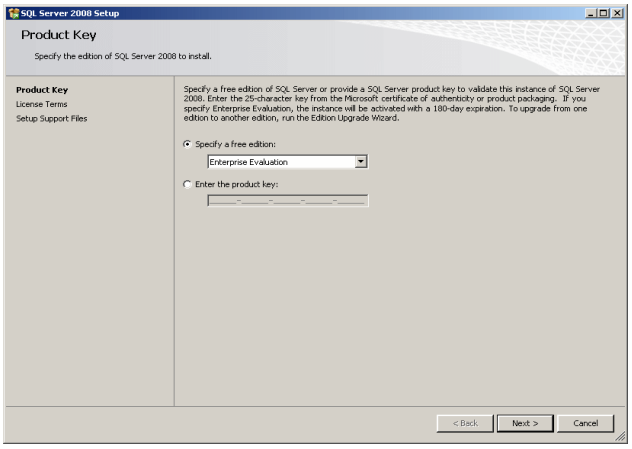


3. Si sólo se insterested en el estudio de diseño y desarrollo de bases de datos en Microsoft SQL Server, en el lado derecho, haga clic en Nuevo SQL Server autónomo ... La instalación se iniciará:

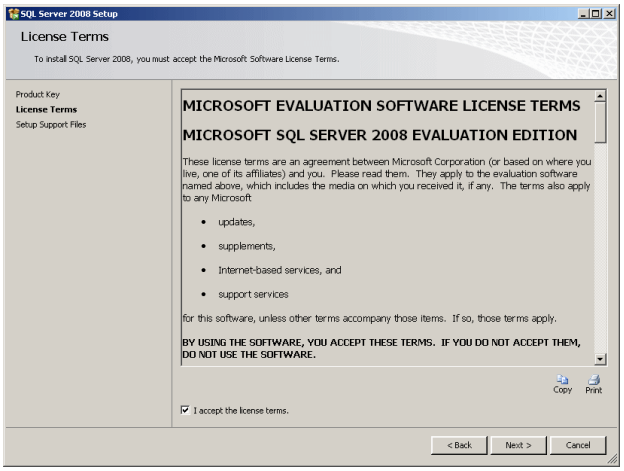
4. Después de que el cuadro de mensaje, haga clic en Aceptar



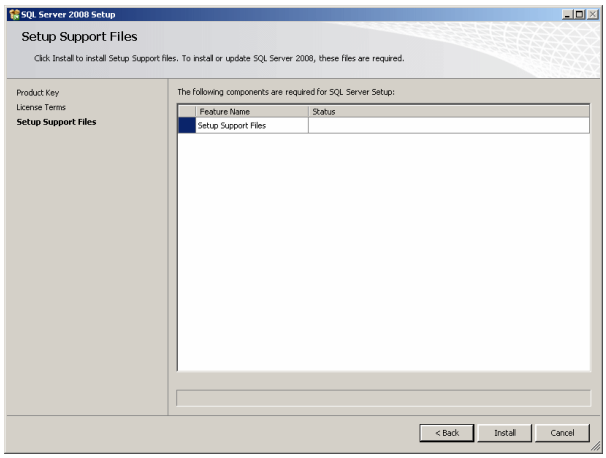
5. Otro cuadro de mensaje aparecerá. Cuando termine, la siguiente página del asistente aparecerá. Si está utilizando la versión de evaluación, leer el mensaje y haga clic en Siguiente



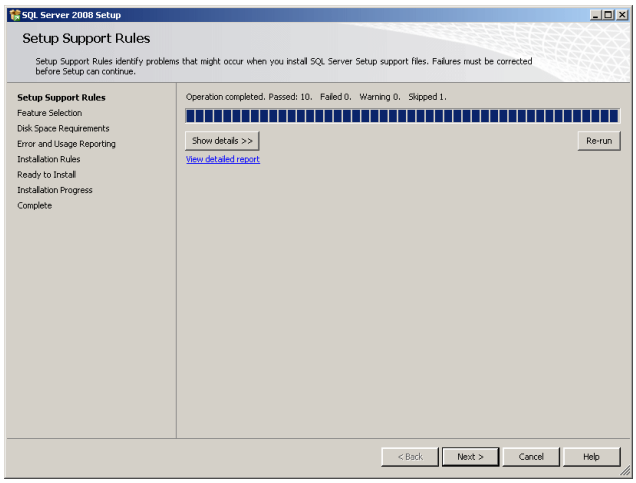
6. La página Términos de licencia se presenta a usted. Leerlo. Si no está de acuerdo con la licencia, haga clic en Cancelar. En caso contrario, haga clic en la casilla de verificación Acepto y haga clic en Siguiente:



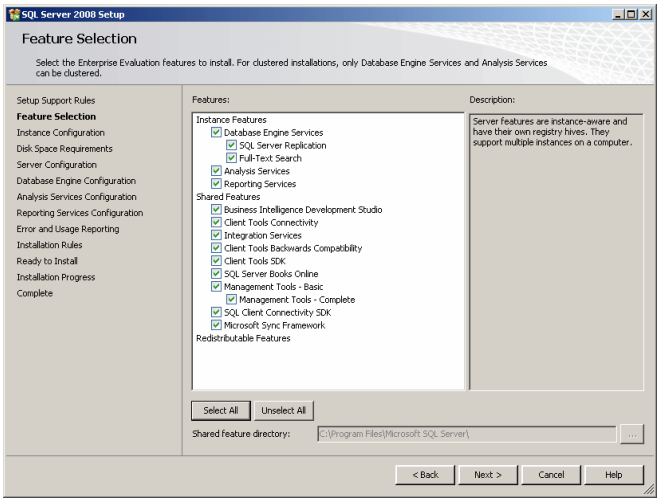
7. En la siguiente página, lea los mensajes y haga clic en Instalar:



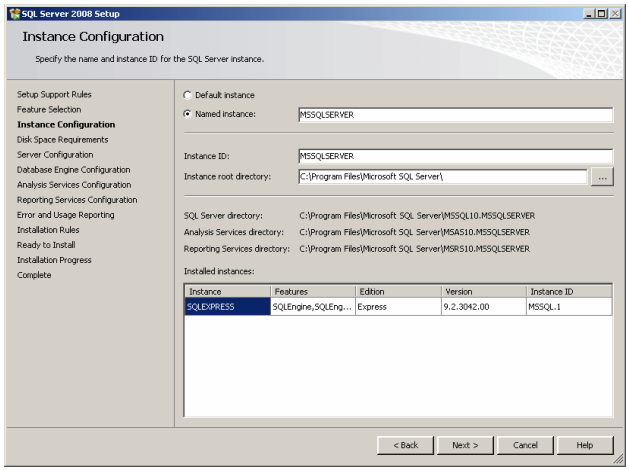
8. En la siguiente página, lea los mensajes y haga clic en Siguiente:



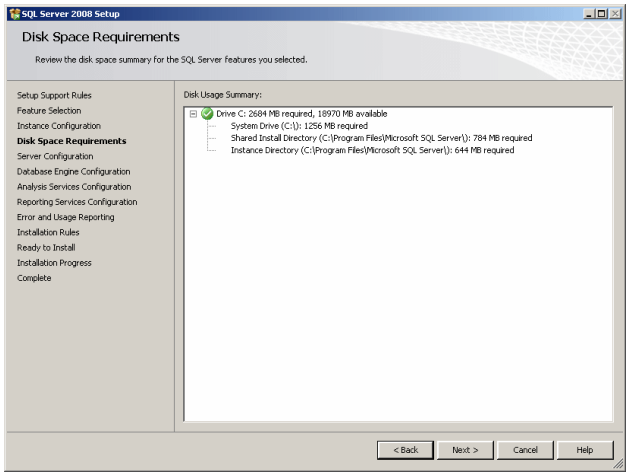
9. La siguiente página le permite especificar lo que quieres instalar y cuál sería excluido. Después de hacer sus selecciones, haga clic en Siguiente



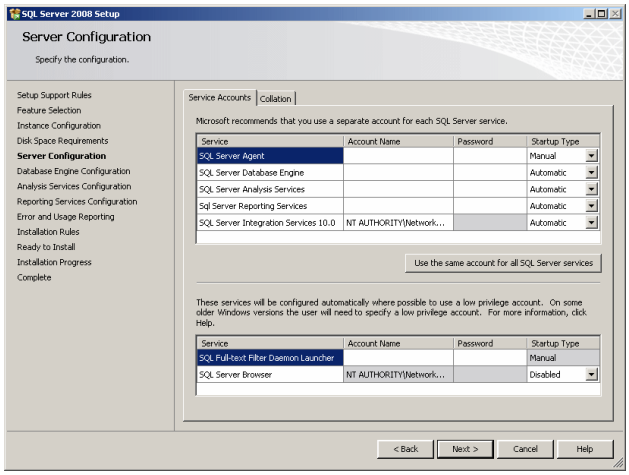
10. La siguiente página le permite nombrar la instancia y para especificar el servidor donde se instala. Después de hacer sus selecciones, haga clic en Siguiente:



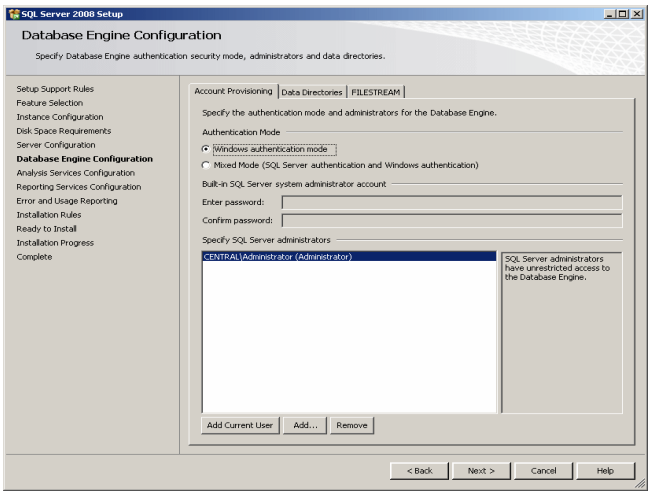
11. La siguiente página mayoría le permite verificar y confirmar el espacio en disco. Después de comprobar que, haga clic en Siguiente



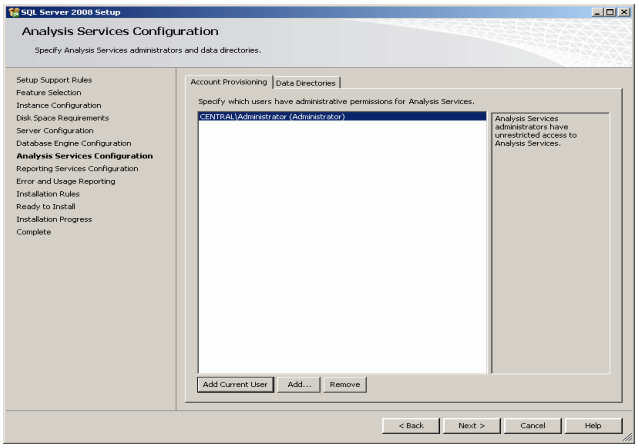
12. La página siguiente le permite configurar el servidor. Después de hacer sus selecciones, haga clic en Siguiente



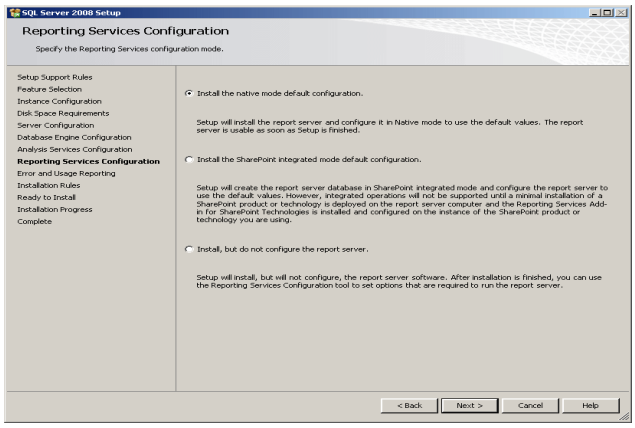
13. En la siguiente página, especificar cómo se haría la autenticación. También debe especificar la cuenta utilizada como administrador. Para ello, puede hacer clic en Agregar usuario actual. Haga clic en Siguiente



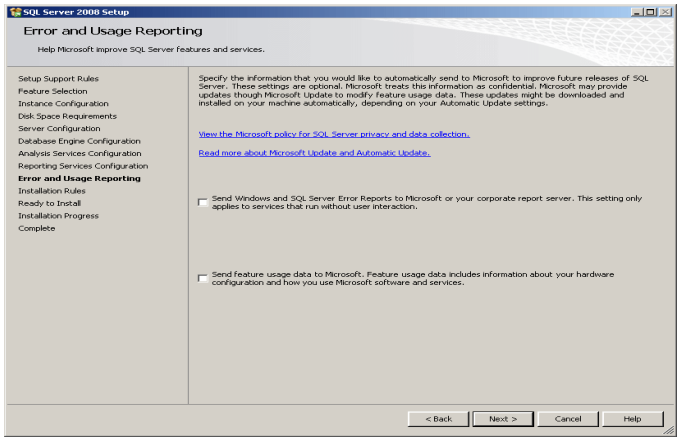
14. El siguiente paso le permite especificar una cuenta para el análisis de los servicios. Usted puede simplemente haga clic en Agregar usuario actual y haga clic en Siguiente



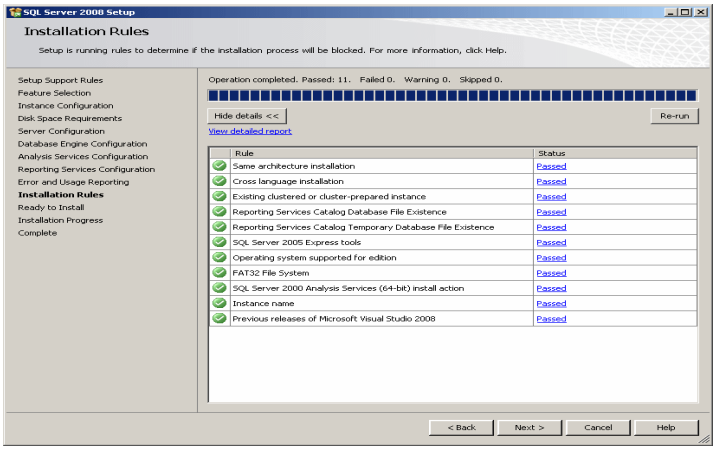
15. La siguiente página le permite especificar la información que se instale el servicio. Aceptar el valor predeterminado o hacer que el cambio necessairy. Haga clic en Siguiente:



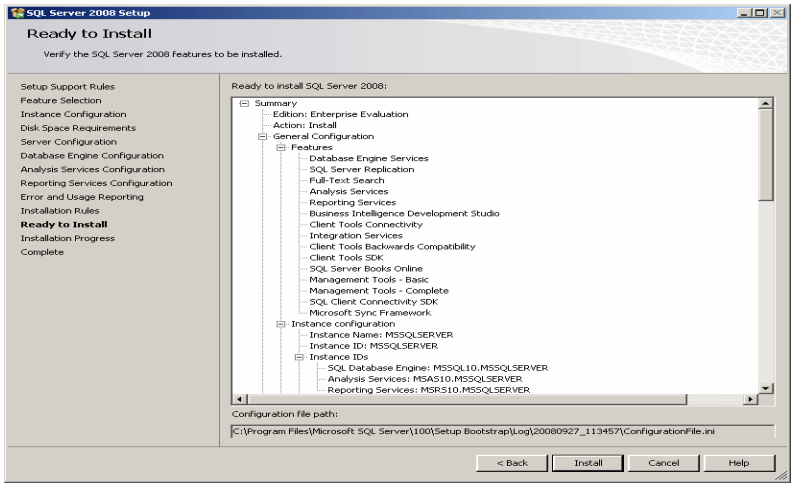
16. La siguiente página le permite especificar si desea información sobre la instalación que se recojan y se envía a Microsoft. Después de hacer su elección (s), haga clic en Siguiente



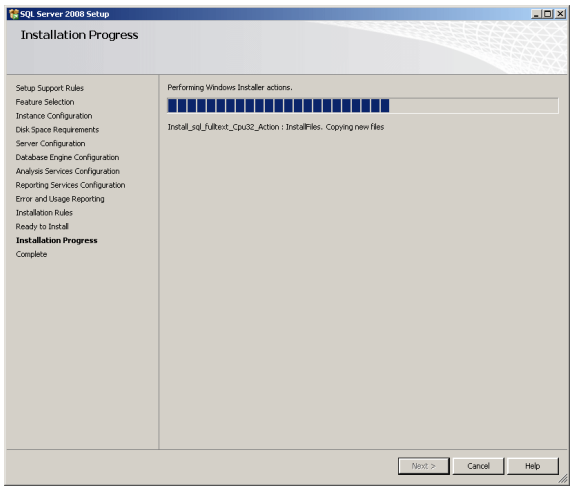
17. El pre-instalación proceder y le permiten saber cuándo se termina con la pantalla de instalación de las Reglas:

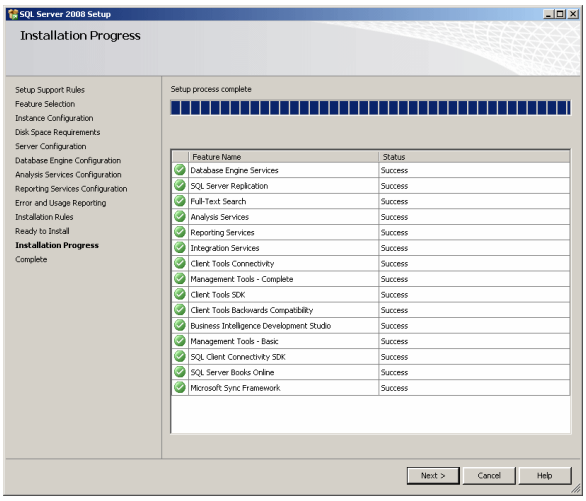


18. Después de leer, haga clic en Siguiente.
Esto indicaría que la instalación está lista para proceder:

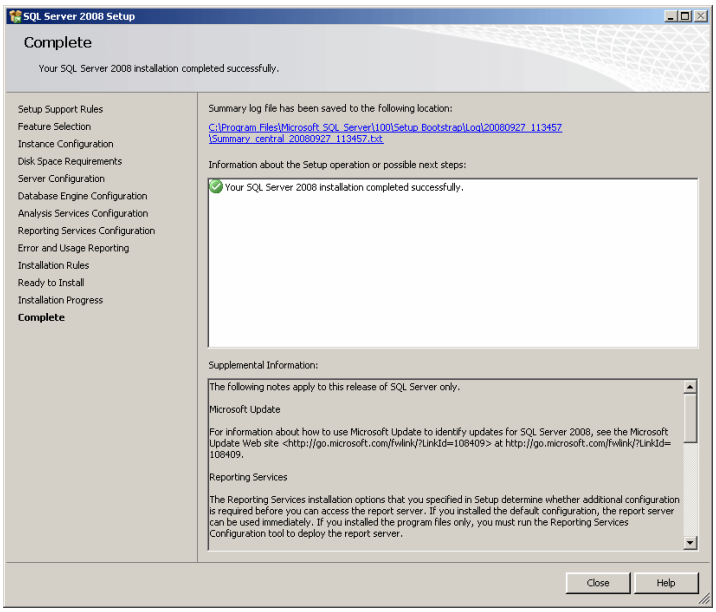


19. Para iniciar la instalación, puede hacer clic en Instalar. La instalación comenzará y la barra de progreso que indique la evolución. Esto puede tomar un tiempo





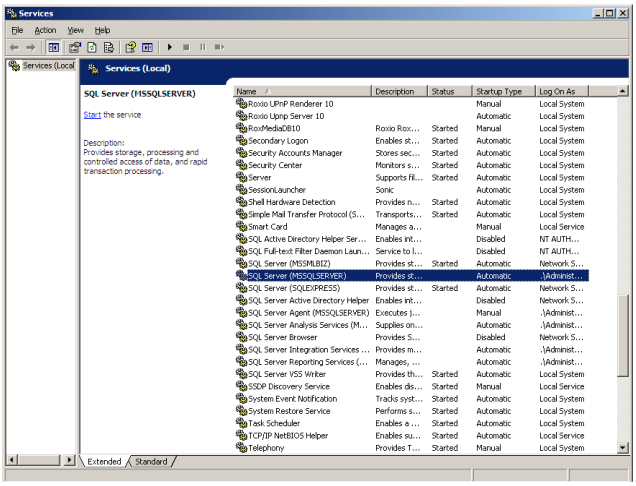
20. Cuando esta fase ha terminado, un cuadro de mensaje le dejará saber. Después de leerlo, usted puede hacer clic en Siguiente. La última página del asistente debe anunciar que la instalación se completa



21. Después de leer el mensaje, puede hacer clic en Cerrar

Después de instalar Microsoft SQL Server, puede comenzar a usarla.

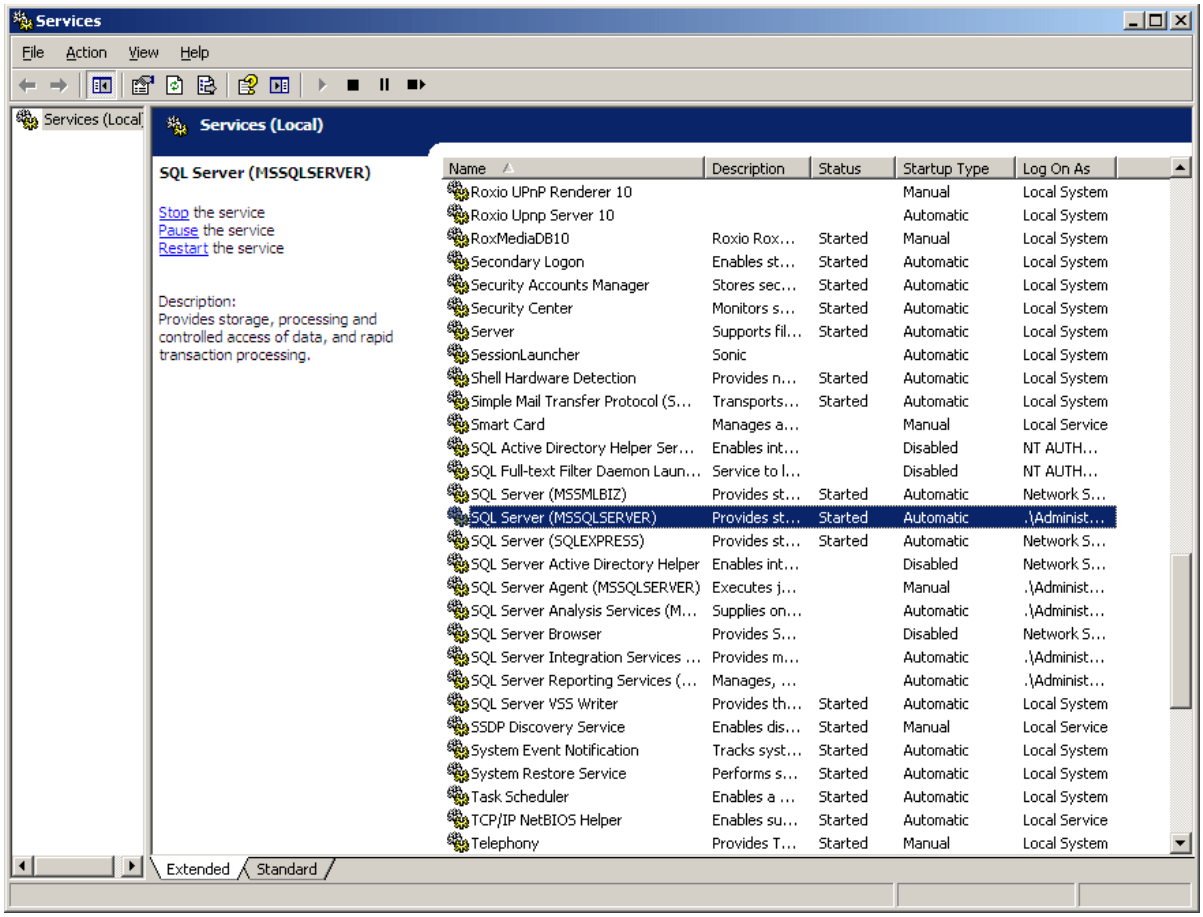
Dado que Microsoft SQL Server funciona como un servicio para el sistema operativo, con el fin de usarlo, usted debe asegurarse de que su servicio se ha iniciado. Para comprobar que (en Microsoft Windows XP Professional, Windows Vista, Windows Server 2003 o Windows Server 3008) puede abrir el Panel de control y Herramientas administrativas. En la ventana Herramientas administrativas, puede abrir los Servicios. En la ventana Servicios, comprobar el estado de SQL Server (MSSQLSERVER) artículo:



Si el servicio MSSQLSERVER está detenido, debe iniciarlo. Para ello, puede hacer clic derecho y haga clic en Iniciar. Si no se inicia, consulte la cuenta con la que iniciar sesión en:

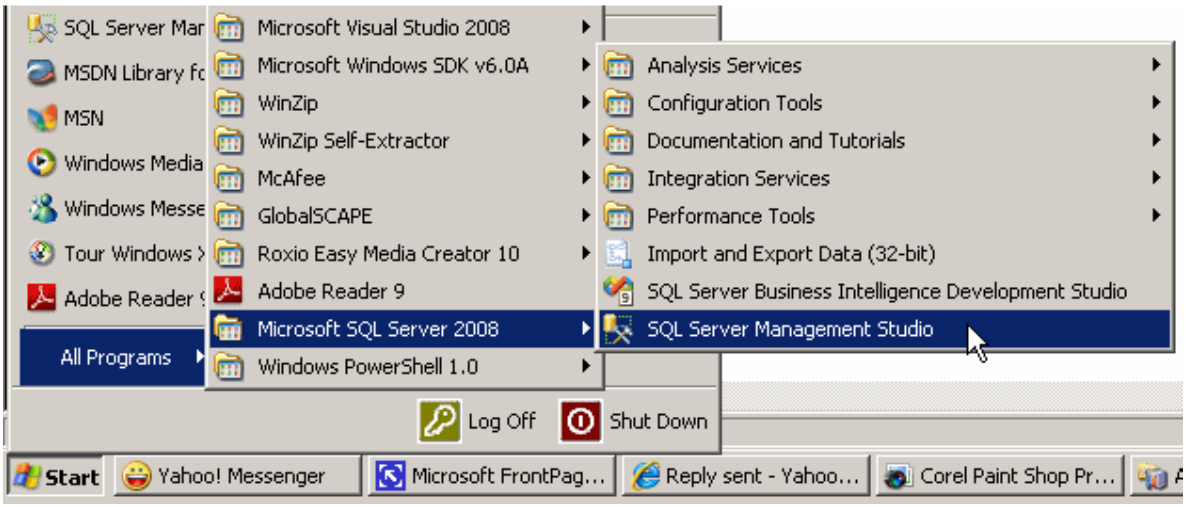
- Si utiliza Microsoft Windows XP Professional y está conectado como administrador, pero no proporcionar una contraseña, usted debe abrir el Panel de control, el acceso en Cuentas de usuario, abra la cuenta de administrador, y crear una contraseña para que
- Si utiliza un servidor (Microsoft Windows Server 2003 o Microsoft Windows Server 2008), asegúrese de iniciar sesión con una cuenta que puede iniciar un servicio

Una vez que el servicio ha iniciado, debe empezar la etiqueta:



Apertura de Microsoft SQL Server

Para iniciar Microsoft SQL Server, puede hacer clic en Inicio -> (Todos los) Programas -> Microsoft SQL Server 2008 -> SQL Server Management Studio



Cuando se inicia, se presentará un cuadro de diálogo que espera que usted registre

Aprendizaje práctico: El lanzamiento de Microsoft SQL Server

1. Para iniciar Microsoft SQL Server, haga clic en Inicio -> (Todos los) Programas -> Microsoft SQL Server 2008 -> SQL Server Management Studio. Una pantalla aparecerá:

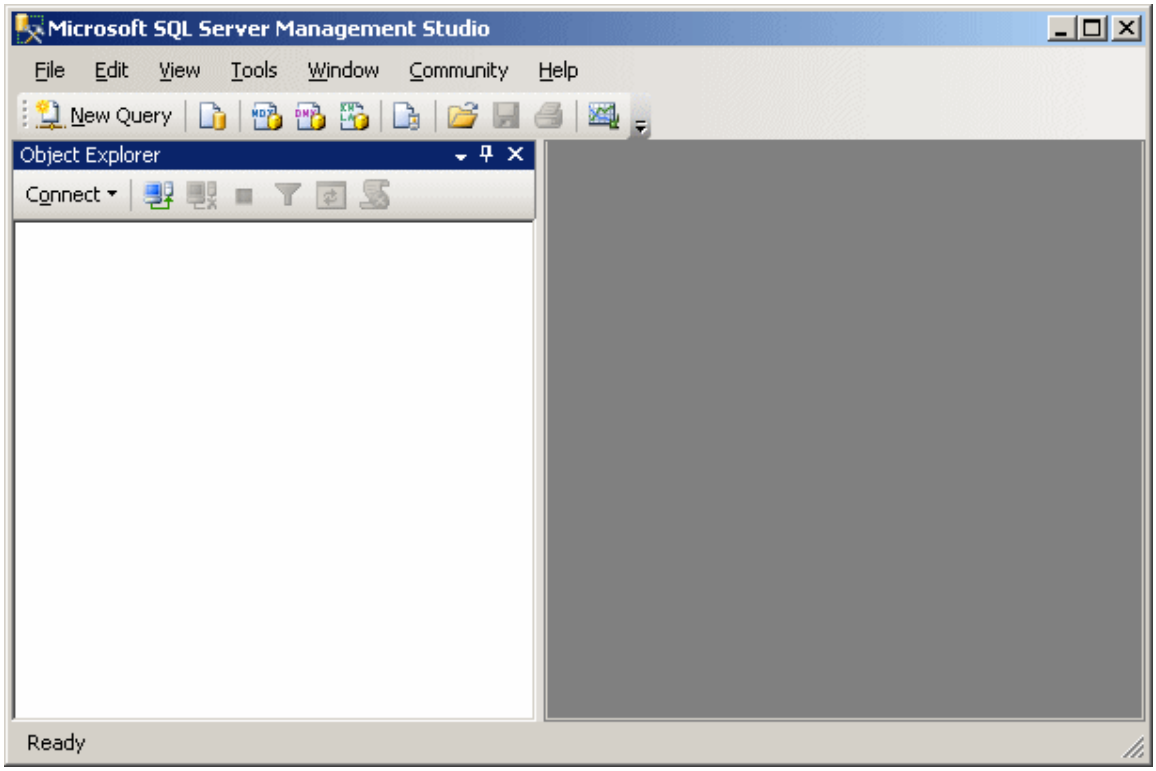
2. En el cuadro de diálogo Conectar al servidor, haga clic en Cancelar

Microsoft SQL Server Management Studio

Introducción

Existen muchas herramientas que va a utilizar en el servidor Microsoft SQL. Uno de ellos se llama Microsoft SQL Server Management Studio. Para acceder a ella,

puede hacer clic en Inicio -> (Todos los) Programas -> Microsoft SQL Server 2008 -> SQL Server Management Studio. Un cuadro de diálogo que vienen, pero usted puede hacer clic en Cancelar en ella:



La sección superior de la de SQL Server Management Studio muestra la clásica barra de título de una ventana, con un icono de la izquierda, seguido por el título de la solicitud, y el sistema de botones en el lado derecho.

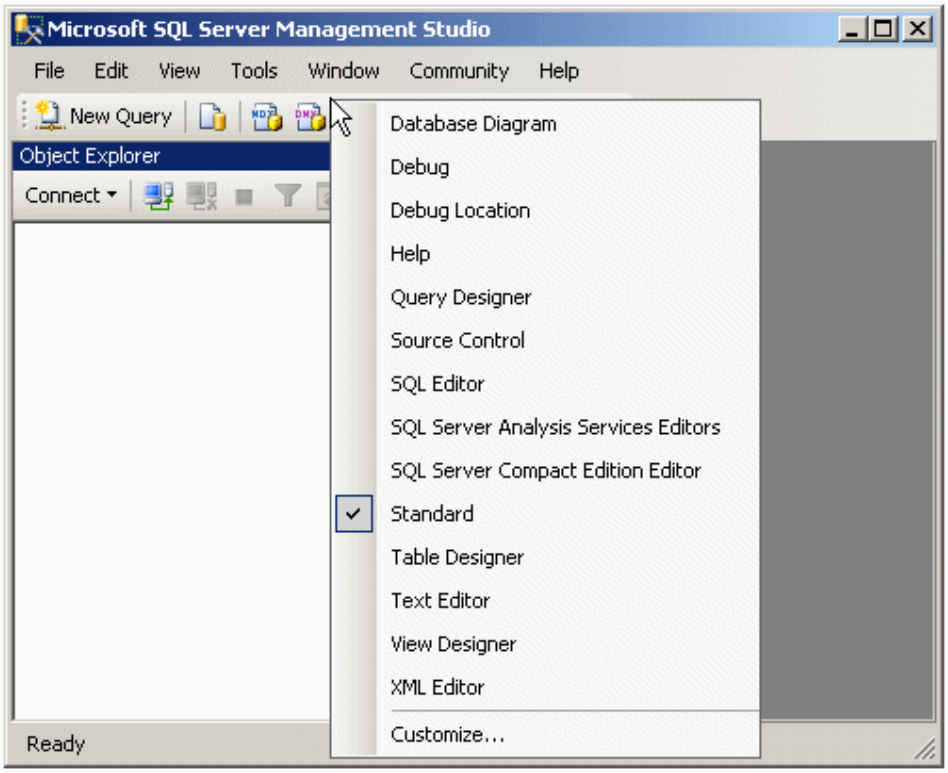
Menú Principal

Bajo la barra de título, la barra de menú muestra las categorías de menús que utilizará para realizar las diferentes operaciones necesarias.

La barra de herramientas estándar

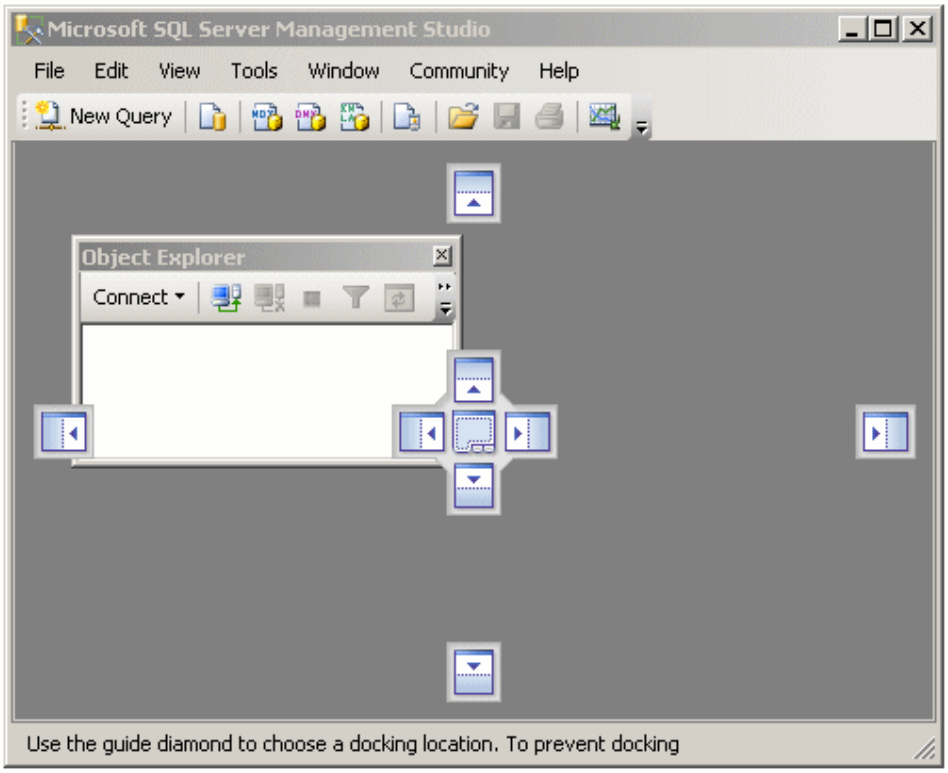
La barra de herramientas Estándar aparece bajo el menú principal:

La barra de herramientas es sólo una de las disponibles. Finalmente, cuando se realiza una acción que se beneficiaría de otra barra de herramientas, Microsoft SQL Server Management Studio que mostrar que la barra de herramientas. Sin embargo, si desea mostrar cualquier barra de herramientas, puede hacer clic derecho en cualquier elemento de menú en el menú principal o cualquier botón de una barra de herramientas. Un menú aparecerá:



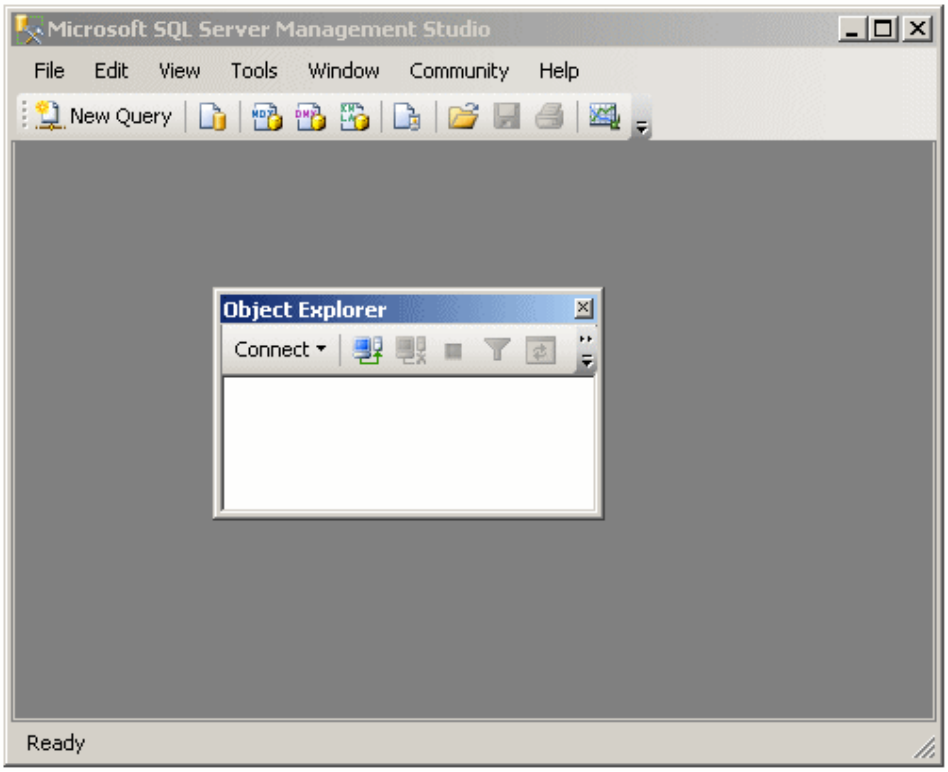
El Explorador de objetos

La parte izquierda de la interfaz de la muestra de objetos del Explorador de Windows, con su barra de título de la etiqueta de objeto Explorer. Esta ventana se dockable, lo que significa que puede moverse desde el lado izquierdo a otro lado de la interfaz. Para ello, puede hacer clic y arrastrar la barra de título a un lugar de su elección. Al empezar a arrastrar, pequeñas cajas que representan los posibles marcadores de posición encontrarían:



Puede arrastrar y soltar a uno de los marcadores de posición.

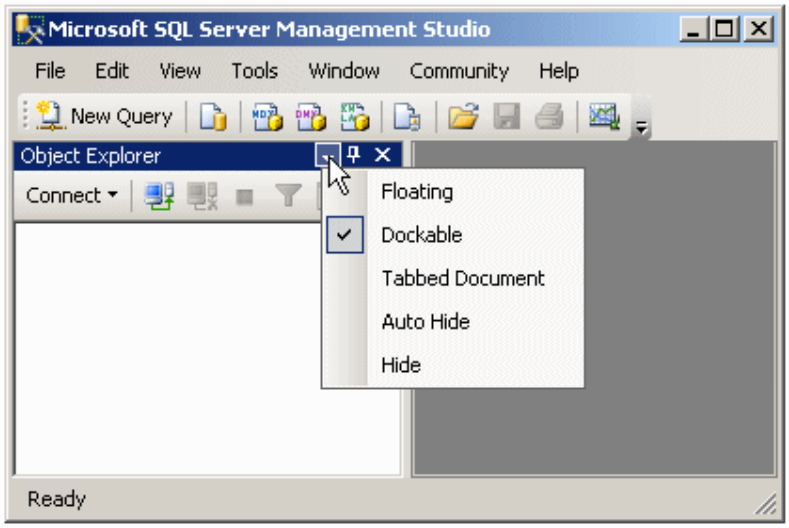
El Explorador de objetos también es floatable, lo que significa que puede que en algún lugar en medio de la interfaz:



Para colocar la ventana de nuevo a su posición anterior, puede hacer doble clic en su barra de título. La ventana también se puede pestañas. Esto significa que la ventana se puede colocar tanto vertical como horizontal.

En cualquier momento, si usted no quiere que el Explorador de objetos, puede cerrar u ocultarla. Para cerrar el Explorador de objetos, haga clic en su botón de cierre.

En la parte derecha del Explorador de objetos título, hay tres botones. Si hace clic en el primer botón que señala hacia abajo, un menú que aparece:



El menú le permite especificar si desea que la ventana que hay que flotaba, acoplado o pestañas.

La parte derecha de la ventana es de una ventana vacía. Esta área será utilizada para mostrar el contenido de lo que está seleccionado en el Explorador de objetos, o para mostrar un resultado de alguna operación. Como se verá más adelante, muchas otras ventanas ocupará la sección correcta, pero que comparten la misma zona . Para que cada uno sabe que estará representada con una ficha y la ficha muestra el nombre (o título) de una ventana.

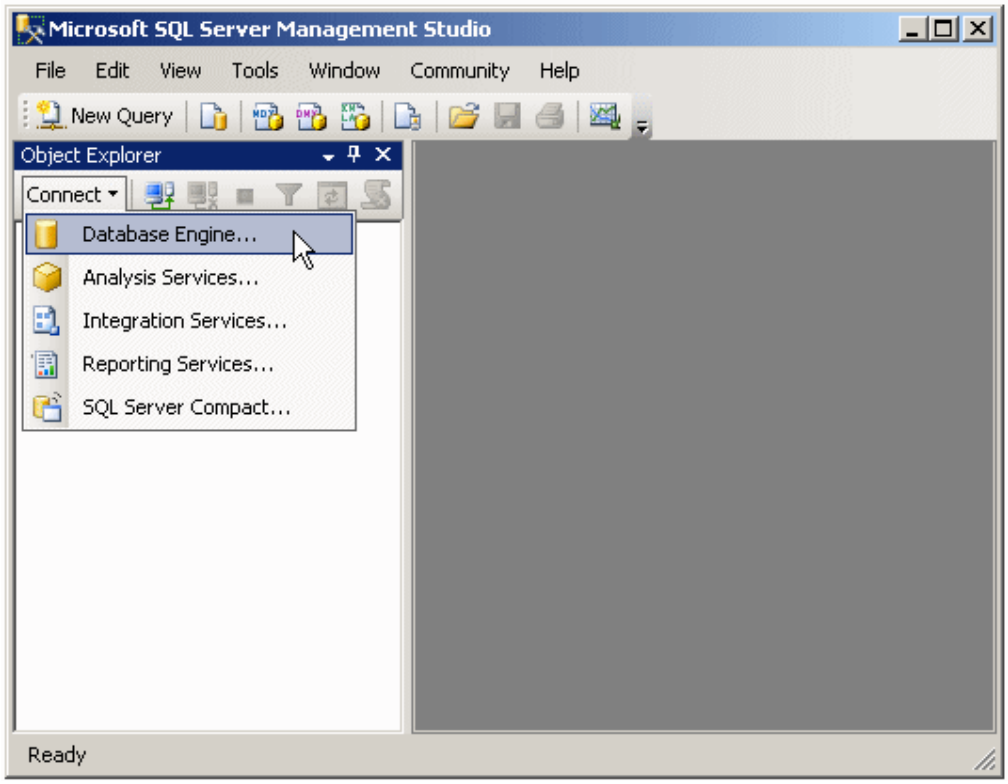
Conexión a un servidor

Conectar al servidor usando

Con el fin de hacer algo importante en Microsoft SQL Server, tendrá que acceder a un servidor. Si inicia Microsoft SQL Server Management Studio desde el botón Inicio, Conectar al servidor cuadro de diálogo que surgen. Si se había iniciado desde el botón Inicio hace clic en Cancelar, pero, para conectarse a un servidor:

- En el menú principal, haga clic en Archivo -> Conectar Explorador de objetos
- En la barra de herramientas Estándar, haga clic en el botón Nueva consulta

- En el Explorador de objetos, puede hacer clic en la flecha del botón Conectar y haga clic en una de las opciones, como motor de base de datos ...

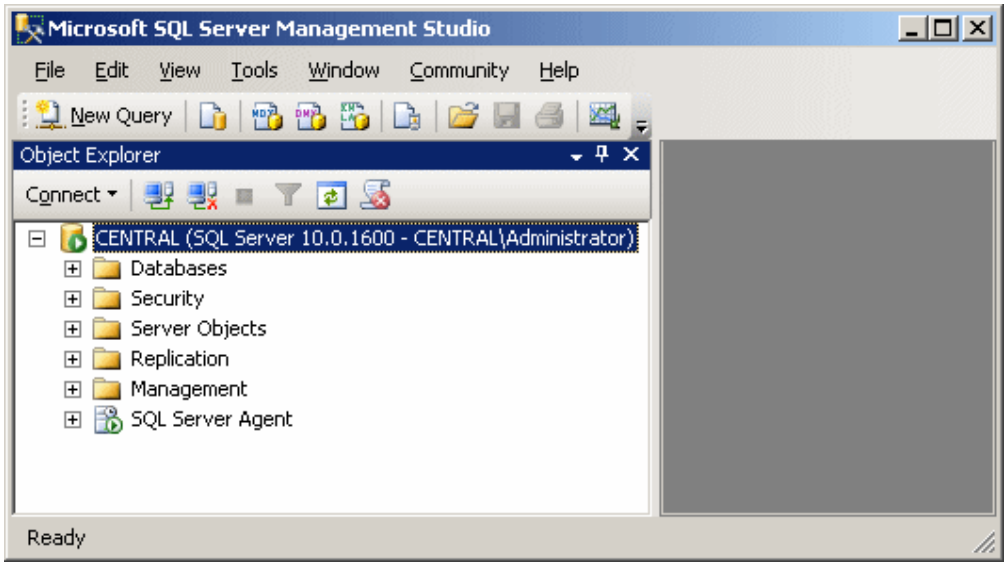


Cualquiera de estas acciones se mostrará el cuadro de diálogo Conectar al servidor de correos:

Desde allí, seleccione el nombre del servidor y el tipo de autenticación para su uso. Si elige la autenticación de Windows, puede simplemente haga clic en Conectar. En caso contrario, la autenticación en el cuadro combinado, puede

seleccionar la autenticación de SQL Server. En este caso, debe proporcionar un nombre de usuario y una contraseña:

Una vez esté listo, haga clic en Conectar. Si te conectas con éxito, el Explorador de objetos que aparecen con unos pocos nodos:

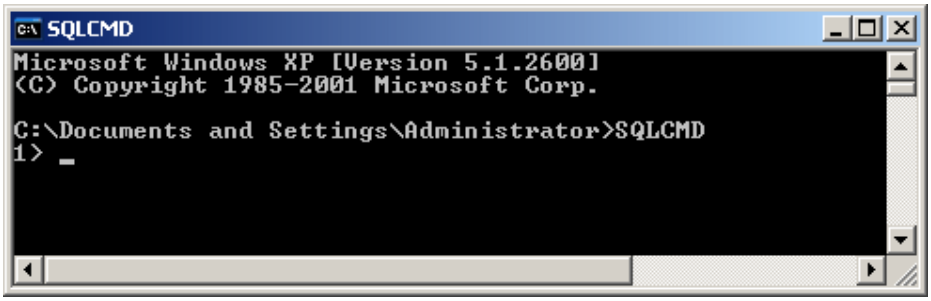


Después de usar Microsoft SQL Server Management Studio, puede cerrarla. Para ello:

- Haga clic en el icono de la izquierda de Microsoft SQL Server Management Studio y haga clic en Cerrar
- En la parte derecha de la barra de título, haga clic en el botón Cerrar sistema
- En el menú principal, haga clic en Archivo -> Salir
- Pulse Alt, M, X

Utilizando el símbolo

Además de la de SQL Server Management Studio, también puede trabajar en Microsoft SQL Server desde el símbolo de DOS. Esto se hace con una aplicación o un comando llamado SQLCMD.EXE. Para usarlo, abra el símbolo del sistema, escriba **SQLCMD** (mayúsculas y minúsculas) y pulse Enter.

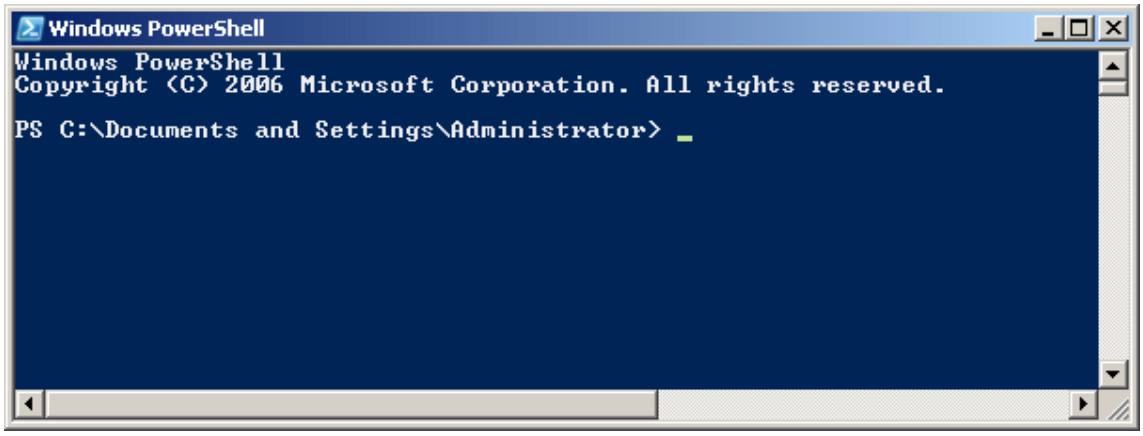


Después de usar Microsoft SQL Server desde el símbolo del sistema, para cerrarla, escriba Salir (mayúsculas y minúsculas) y pulse Enter. Para cerrar la ventana de DOS:

- **Escriba** Exit (mayúsculas y minúsculas) y pulse Enter
- Haga clic en el botón Cerrar sistema

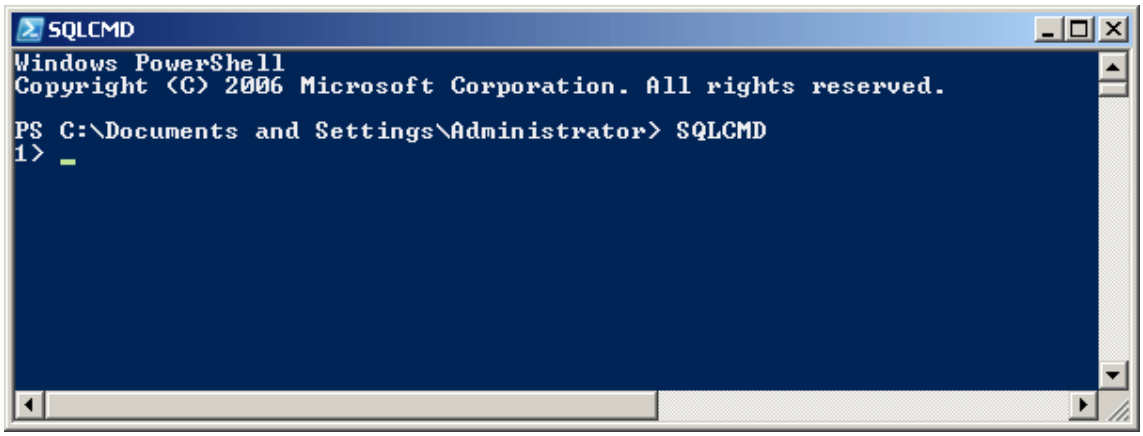
Utilizando el Windows PowerShell

Cuando Microsoft SQL Server 2008 está instalado, también se instala el Windows PowerShell 1.0, que es un nuevo comando de la aplicación basada en Microsoft. Además de la de SQL Server Management Studio y el símbolo del sistema, puede usar PowerShell para crear y gestionar bases de datos. Para acceder a ella, puede hacer clic en Inicio -> (Todos los) Programas -> Windows PowerShell 1.0 -> Windows PowerShell. Una ventana de DOS a pantalla:



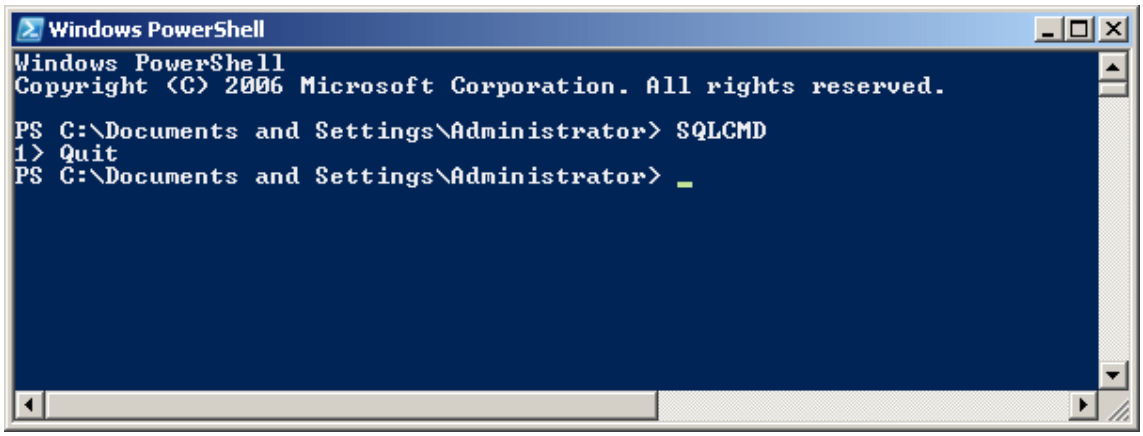
Observe que la barra de título muestra de Windows PowerShell.

Para tener acceso a SQL Server de Microsoft PowerShell, escriba SQLCMD y pulse Intro:



Observe que, en esta ocasión, la barra de título muestra SQLCMD, lo que indica que la aplicación está lista para recibir comandos que se relacionan con Microsoft SQL Server.

Después de usar el PowerShell, para salir de Microsoft SQL Server, escriba Salir (mayúsculas y minúsculas) y pulse Intro:



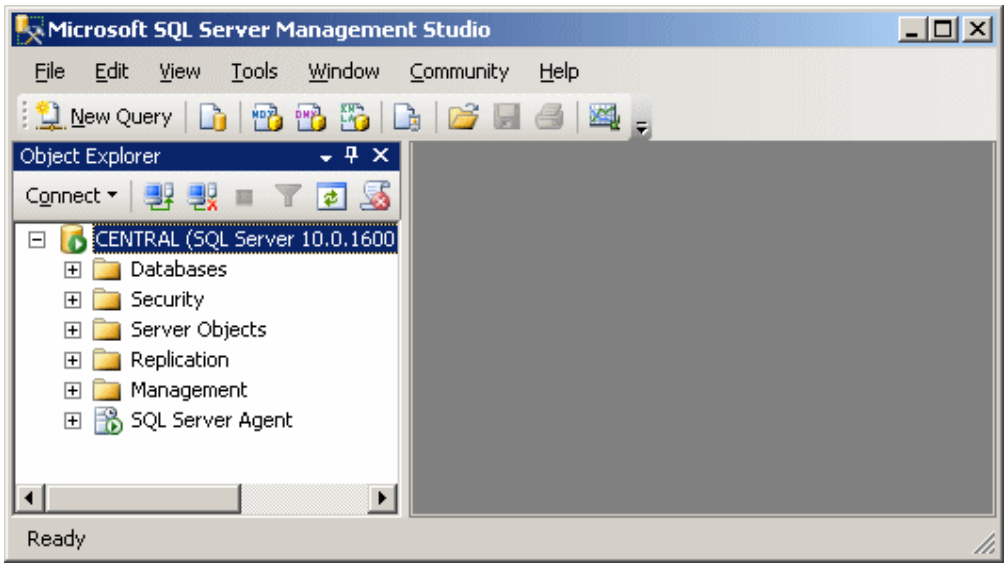
PowerShell y para cerrar la ventana de DOS, usted puede:

- **Escriba** Exit (mayúsculas y minúsculas) y pulse Enter
- Haga clic en el botón Cerrar sistema

Uso de Microsoft SQL Server Management Studio

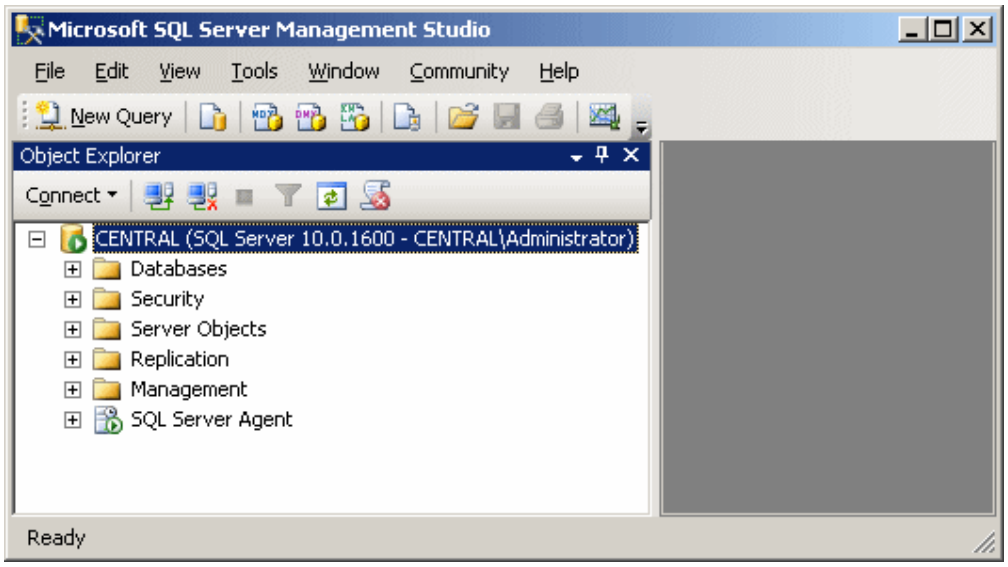
Introducción

El Objeto Explorer muestra una lista de artículos como un árbol de estilo. Uno de los más utilizados regularmente artículos será el nombre del servidor que está utilizando. Si usted está recién empezando a aprender el desarrollo de bases de datos o usted es un desarrollador de base de datos junior, usted puede usar o ver sólo un servidor. En algunos casos, se le puede hacer frente a muchos. A pesar de todo, usted debe saber siempre lo que el servidor que se está conectando. Esto es fácil de comprobar con el primer nodo de la norma Object Explorer. En el siguiente ejemplo, el servidor central se denomina:



El nombre del servidor es seguido de paréntesis.

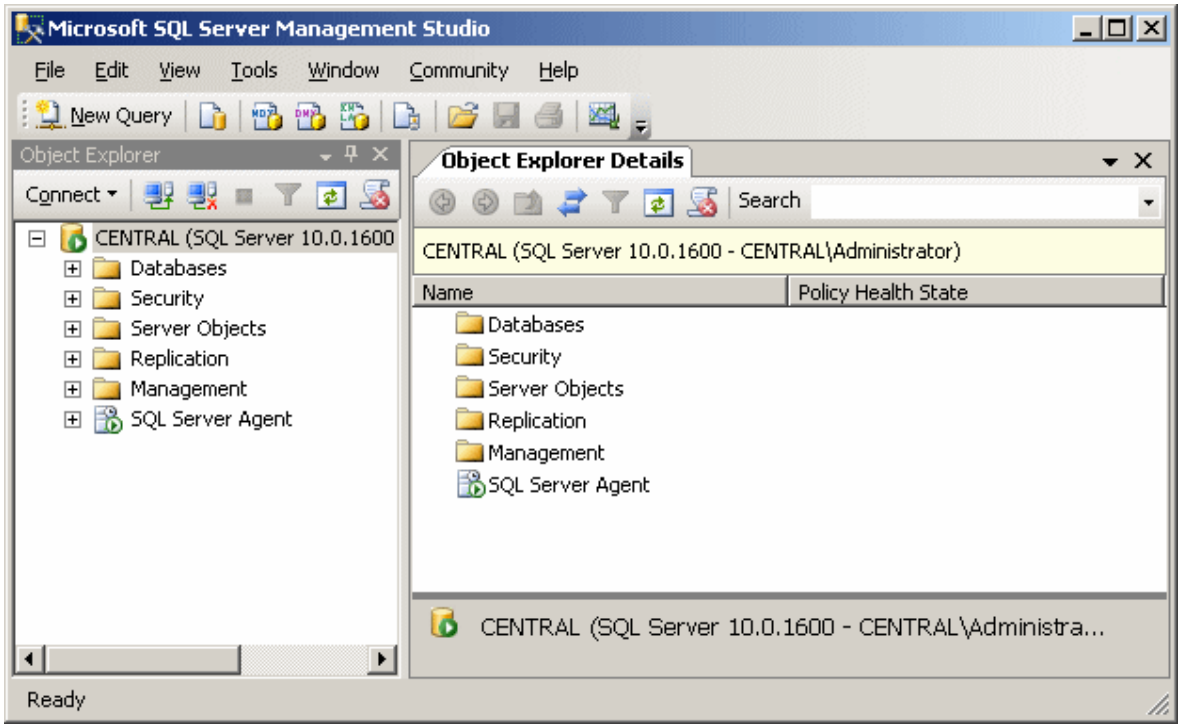
En la sección anterior, vimos que, para establecer una conexión con un servidor, debe autenticarse. En algunos casos, puede utilizar la misma cuenta una y otra vez. En algunos otros casos, usted puede tener diferentes cuentas que se utiliza para los diferentes escenarios, tales como una cuenta para el desarrollo de bases de datos, una cuenta de gestión de bases de datos, y / o una cuenta para la base de datos de prueba. Algunas operaciones no pueden ser realizadas por algunas cuentas. Al llevar a cabo algunas operaciones, usted debe saber siempre lo que cuenta que está utilizando. Puede comprobar esto en los paréntesis del nombre del servidor. En la siguiente conexión, una cuenta llamada Administrador está conectado a un servidor llamado Central:



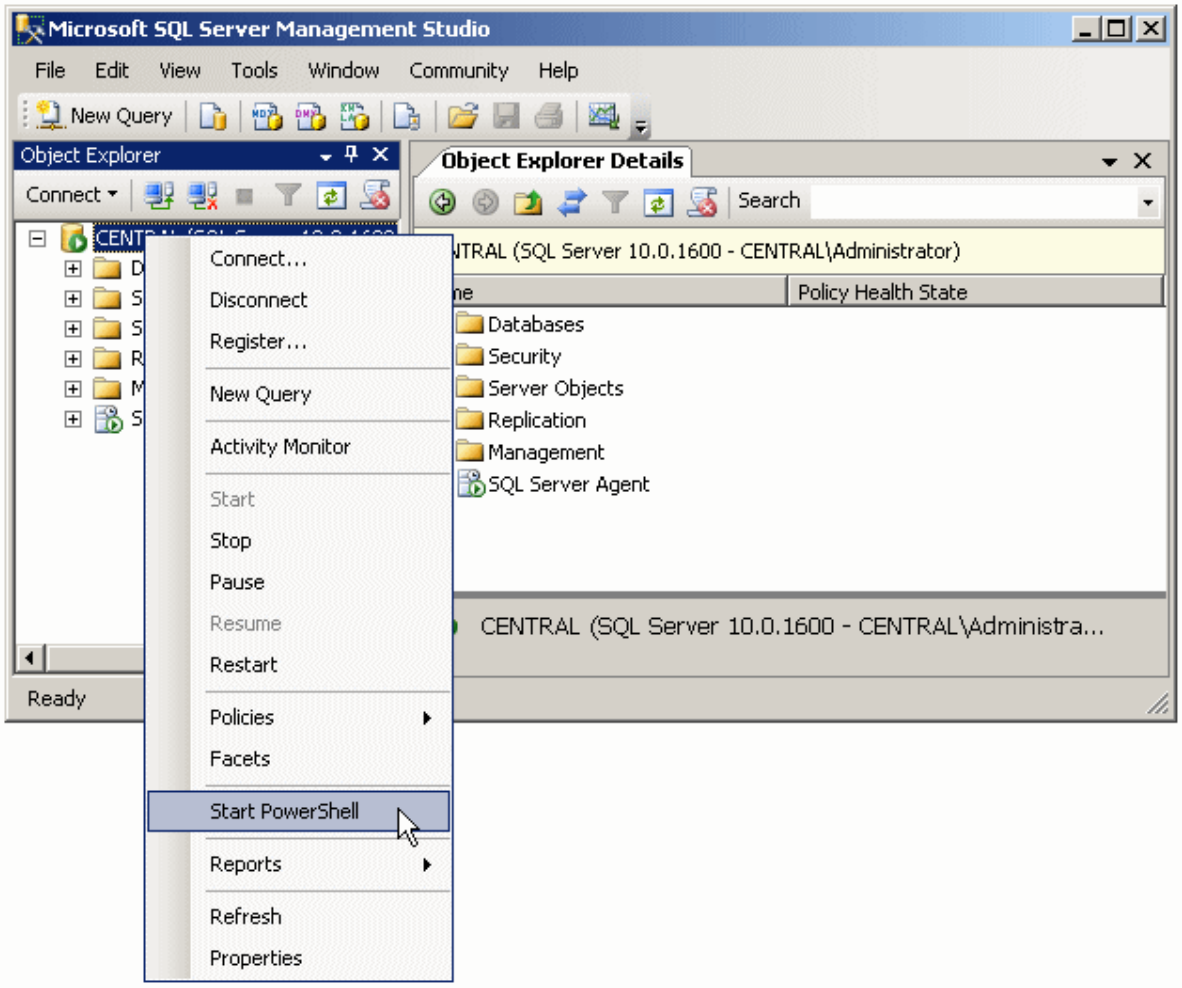
Detalles del Explorador de objetos

Hemos visto que, por defecto, el área de derecho de Microsoft SQL Server Management Studio muestra una ventana vacía de color gris. Cuando seleccione

algo en el Explorador de objetos, puede utilizar ese derecho para mostrar la zona más detallada información acerca de seleccionar el tema. Para ello, en el menú principal, puede hacer clic en Ver -> Detalles del Explorador de objetos. Los principales están en el lado derecho sería entonces lleno de información:



Probablemente el más ordinario de nodo que se interesa, tiene la etiqueta Bases de datos. Este nodo contiene los nombres de las bases de datos en el servidor que está conectado. También, a partir de ese nodo, que puede realizar casi cualquier operación necesaria de una base de datos. Para ver la mayoría de las acciones disponibles con regularidad, puede ampliar el nodo Bases de datos y algunos de sus hijos. A continuación, puede hacer clic derecho en cualquiera de las bases de datos o uno de sus nodos secundarios. Por ejemplo, para iniciar PowerShell, puede hacer clic derecho en el nodo Bases de datos o el nombre del servidor y haga clic en PowerShell:

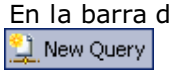


Cuando el PowerShell viene, lo que muestra depende de lo que había hecho clic-derecho.

Introducción al Código

A pesar de que usted llevará a cabo muchas de sus operaciones de base de datos de forma visual, algunas otras operaciones, será necesario que escribir código. Para ayudar en este sentido, Microsoft SQL Server proporciona un editor de código y diferentes plantillas de código.

Para abrir el editor:

- En el menú principal, puede hacer clic en Archivo -> Nuevo -> Consulta de conexión con los actuales
- En la barra de herramientas Estándar, haga clic en el botón Nueva consulta 

- En el Explorador de objetos, haga clic derecho en el nombre del servidor y haga clic en Nueva consulta

Esto crearía una nueva ventana y colóquelo en la parte derecha de la interfaz. Si ya has escrito el código o no, puede guardar el documento de el editor de código en cualquier momento. Para guardarlo:

- Puede pulsar Ctrl + S
- En el menú principal, puede hacer clic en Archivo -> Guardar SQLQueryX.sql ...
- En la barra de herramientas Estándar, haga clic en el botón Guardar

Usted estará obligado a facilitar un nombre para el archivo. Después de guardar el archivo, su nombre aparecerá en la pestaña del documento.

El Structured Query Language

Introducción

Después de establecer una conexión, puede tomar medidas, como la creación de una base de datos y datos.

El Structured Query Language, conocido como el SQL, es un lenguaje universal utilizado en c informáticos para crear y gestionar bases de datos.

SQL puede ser pronunciada Sequel o SQL En nuestra experiencia, vamos a considerar la pronu Por esta razón, la abreviatura será siempre considerada como una palabra, lo que supondría "u SQL" en lugar de "una sentencia SQL". Además, vamos a escribir con regularidad, "SQL" en lu lenguaje SQL, como la L, ya representa Idioma.

Al igual que otros no-específica de la plataforma lenguas como el C / C + +, Pascal o Java, SQL se a aplicado a diferentes sistemas de bases de datos. Para adaptar el SQL de Microsoft SQL Server, desarrollado de Transact-SQL como Microsoft SQL de la aplicación. Transact-SQL es el lenguaje utiliz por Microsoft SQL Server y MSDE. Aunque SQL Server altamente adhiere a los estándares de SQL, tien detalles que no puede aplicarse a otros sistemas de bases de datos como MySQL, Oracle, o incluso de etc, aunque también bastante conformes a la norma.

La sentencia de SQL que aprender y usar en este caso es de Transact-SQL. En otras palabras, nosotros usted está utilizando Microsoft SQL Server como su plataforma de aprendizaje sobre bases de datos. Es menos que se especifique lo contrario, la mayor parte del tiempo, en este sitio, la palabra se refiere a S o la forma en que el idioma se lleva a cabo en Microsoft SQL Server.

El intérprete de SQL

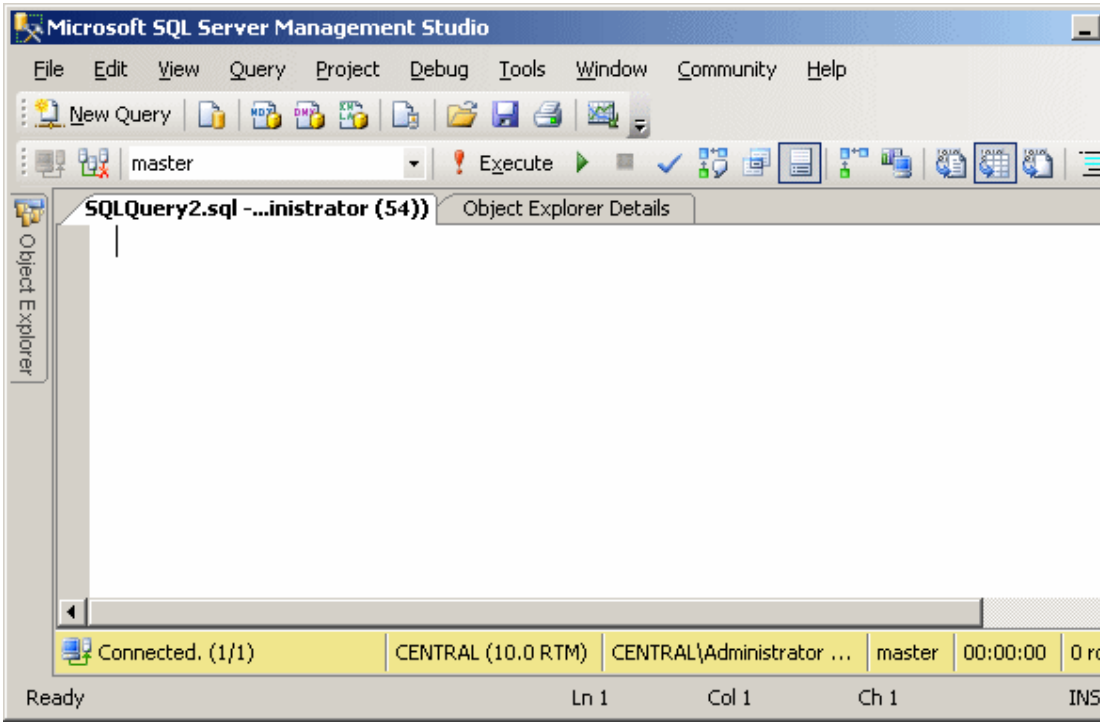
Como un lenguaje de ordenador, el SQL se utiliza para dar instrucciones a un programa llamad intérprete. Como vamos a ver en varias secciones, debe asegurarse de que usted da instrucciones pre sensible a mayúsculas y minúsculas. Esto significa que crear, crear, y crear significa la misma cosa. E escribir las propias palabras de SQL en mayúsculas. Esto ayuda a distinguir las instrucciones SQL con utiliza para su base de datos.

Como vamos a ver en este y los demás lecciones de este sitio, que utiliza SQL declaraciones por escrit con esto, Microsoft SQL Server proporciona una ventana, también conocida como la ventana de con utilizar para escribir su código SQL. Para acceder a ella, en el lado izquierdo de la ventana, puede hac el nombre del servidor y haga clic en Nueva consulta. De la misma manera, puede abrir el mayor núme la Nueva consulta que quieras.

Cuando la ventana de consulta, se mostrará una ventana en blanco en el cual usted puede escribir su se escribe es un documento y se pueden guardar como un archivo. El archivo tendrá la extensión. S abre una nueva consulta, que se representa con una ficha. Para pasar de un código a otro, puede hace Para despedir a una instancia de la consulta, en primer lugar acceder a él (haciendo clic en su ficha), y derecho, haga clic en el botón de cierre . Si usted ha escrito en el código de ventana de consulta, usted se pregunta para guardar el código. Si desea preservar su código, y luego guardarlo. Si ya ha ej en la ventana (que aprender a escribir y ejecutar código SQL), no tiene que guardar el contenido de la v

Ejecución de una Declaración

En las siguientes secciones y la experiencia, vamos a aprender diferentes técnicas de creación de SQL manera predeterminada, cuando una nueva ventana de consulta aparece, se hace blanco de una ampl usted escribe sus declaraciones:

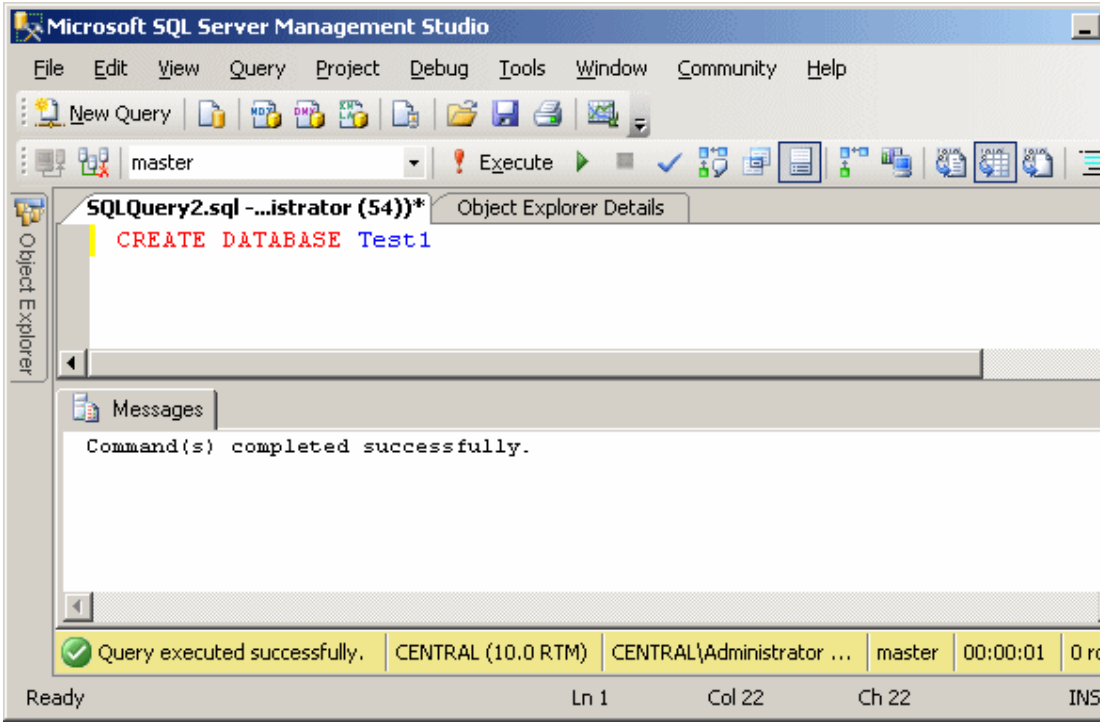


Después de escribir una declaración, puede ejecutar, ya sea para que sea activa o simplemente par ejecutar una declaración:

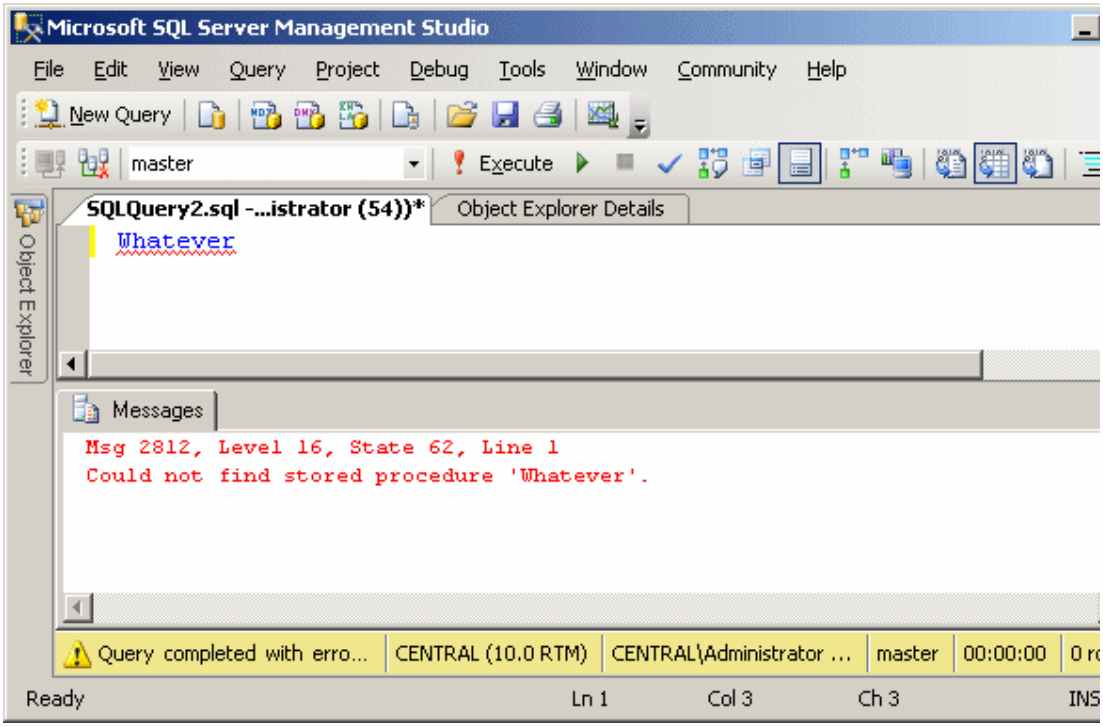
- Puede presionar la tecla F5
- En el menú principal, puede hacer clic en Consulta -> Ejecutar
- En la barra de herramientas del Editor de SQL, puede hacer clic en el botón Ejecutar

- Puede hacer clic derecho en algún lugar en el editor de código y haga clic en Ejecutar

Al ejecutar código, se convierte en editor de código dividido en dos secciones horizontales:



Además, cuando ejecute el código, el intérprete que analizar en primer lugar. Si hay un error, es una
rojo o más líneas de texto en su sección inferior. Aquí está un ejemplo:



Si no hay ningún error en el código, lo que pasa cuando usted ejecuta una declaración depende del c
declaración.

Accesorios para el Código de escritura de SQL

Comentarios

Un comentario de texto es que el intérprete de SQL no considerar como código. Como tal, es un c como desee. Sea cual sea el hecho de que no se puede leer. Transact-SQL admite dos tipos de coment comentario que comienza con / * y termina con * / se pueden utilizar. Para aplicarla, inicie una líne cualquier tipo de texto que te gusta, en tantas líneas como desee. Para cerrar la sección de observac Aquí hay un ejemplo de una línea de comentario:

/ * En primer lugar saber si la base de datos que queremos crear ya existe * /

Un comentario también se puede propagar en más de una línea, como un párrafo. Aquí está un ejempl

```
/ * En primer lugar saber si la base de datos que MotorVehicleDivision
    desea crear ya existe.
    Si la base de datos que existe, no queremos más.  Tan,
    eliminarlo del sistema.  * /
```

Transact-SQL también apoya la doble guión comentario. Este comentario se aplica a una sola línea de te con el inicio de la línea --. Nada en el lado derecho de - forma parte de un comentario y no se consider Aquí está un ejemplo:

```
- ==============================================
- Base de datos: MotorVehicleDivision
- ==============================================

/ * En primer lugar saber si la base de datos que MotorVehicleDivision
    desea crear ya existe.
    Si la base de datos que existe, no queremos más.  Tan,
    eliminarlo del sistema.  * /


- Ahora que la base de datos no está en el sistema, cree que
```

El fin de una declaración

En SQL, después de una declaración por escrito, usted puede terminar con un punto y coma. De hech muchas declaraciones en un bloque, que debería poner fin a cada uno de ellos con un punto y coma. C muchas declaraciones, algunas de ellas debe venir en pos de otros.

Hora de irse

De declaraciones separadas, es decir, para indicar cuando termina una declaración, puede utilizar **el** (en la realidad y las normas basadas en SQL, es el punto y coma que se necesitaría, pero la de Micr como intérprete acepta la **IR** final de una declaración).

Resumen de la lección

Concepto aprendidas

- Definición de Microsoft SQL Server
- ¿Cuáles son las distintas maneras en que puede obtener de Microsoft SQL Server?
- ¿Cuál es la versión actual de Microsoft SQL Server?
- ¿Cómo conectar con el servidor?

Ejercicio: Utilizando el símbolo

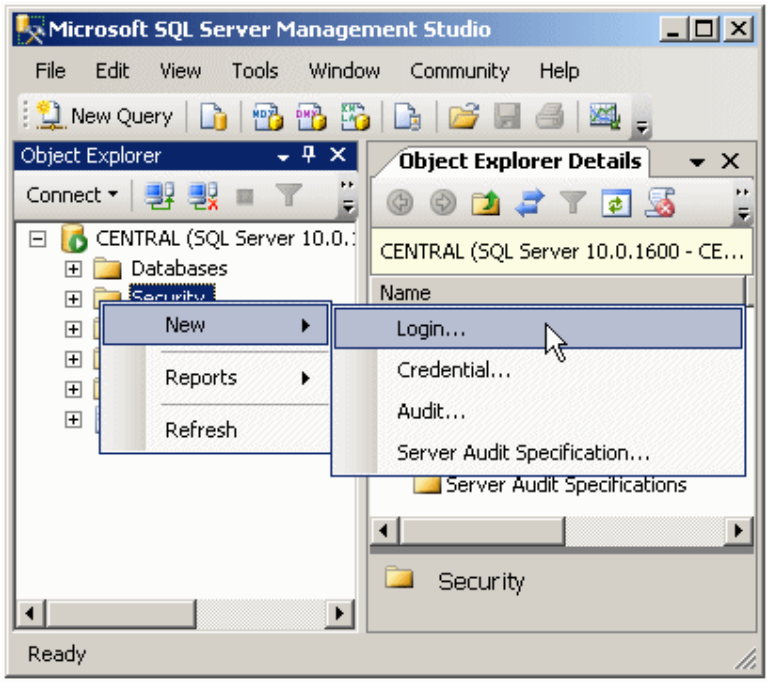
1. Compruebe la ayuda de Microsoft SQL Server (Libros en pantalla) y buscar el comando que se utili conectarse al servidor desde la línea de comandos
2. La práctica de conectar con el servidor desde la línea de comandos y de la de Windows PowerShell
3. Salga del símbolo del sistema y el de Windows PowerShell

Una base de datos es principalmente un grupo de archivos de computadora que cada uno tiene un nombre y una ubicación. Al igual que hay diferentes maneras de conectarse a un servidor, de la misma manera, hay diferentes formas de crear una base de datos.

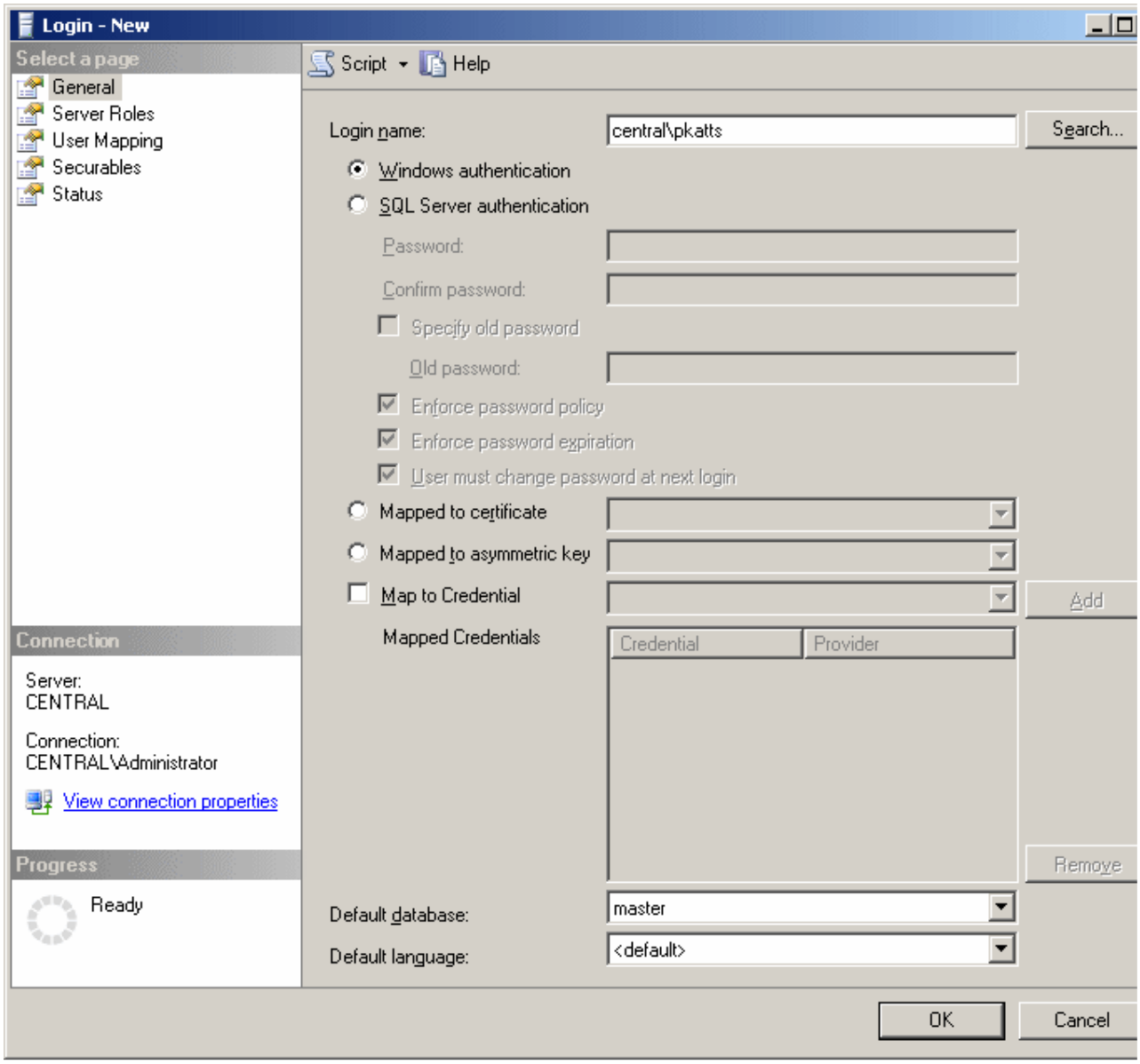
Para crear una nueva base de datos en Microsoft SQL Server Management Studio, en el Explorador de objetos, puede hacer clic derecho en el nodo Bases de datos y haga clic en Nueva base de datos ... Esto abriría el cuadro de diálogo Nueva base de datos.

Aprendizaje práctico: Creación de un inicio de sesión

1. Asegúrese de que ha creado una cuenta de usuario llamada **pkatts** presentó en la Lección 1.
   Para iniciar Microsoft SQL Server, haga clic en Inicio -> (Todos los) Programas -> Microsoft SQL Server 2008 -> SQL Server Management Studio
2. En el Conectar al servidor cuadro de diálogo, hacer la correcta selección y proporcionar la información adecuada y, a continuación, haga clic en Conectar
3. En el Explorador de objetos, expanda el nombre del servidor si es necesario.
   Haga clic con el botón de seguridad, coloca el ratón sobre la nueva, y haga clic en Iniciar sesión ...



4. Establecer el propietario para *nombreDeEquipo* \ pkatts (Windows XP o Vista) o *nombreDeDominio* \ pkatts (Windows Server) (también puede hacer clic en Buscar para localizar el nombre de usuario)
5. Haga clic en Aceptar



El nombre de una base de datos

Probablemente, el requisito más importante de la creación de una base de datos es para darle un nombre. El SQL es muy flexible en lo que se refiere a los nombres. De hecho, es menos restrictiva que la mayoría de los otros lenguajes. Sin embargo, existen reglas que debes seguir al nombrar los objetos en sus bases de datos:

- Un nombre puede empezar por una letra (a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u , v, w, x, y, z, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T , U, V, W, X, Y o Z), un dígito (0, 1, 2, 3, 4, 5, 6, 7, 8 o 9), un guión bajo (_) o no legible carácter. Ejemplos de ello son **_n, actuar,% 783, Segunda**
- Después del primer carácter (letra, dígito, subrayar, o un símbolo), el nombre puede tener combinaciones de relieve, las letras, números o símbolos. Ejemplos de ello son **_n24** o **act_52_t**
- Un nombre puede incluir espacios. Ejemplo son **c0untries ª @ ts, govmnt (registros),** o **gl0b # $ ENCUESTA | |**

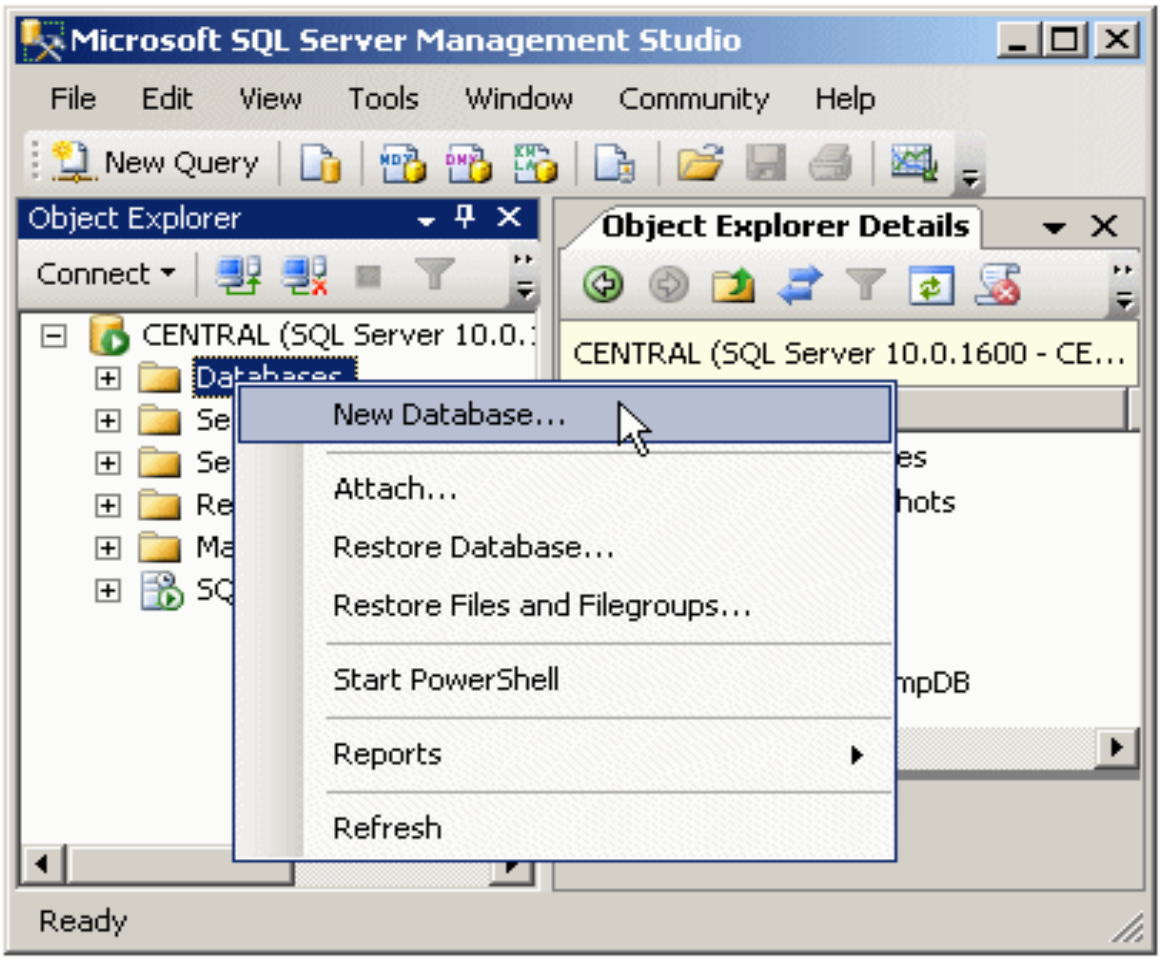
Gracias a la flexibilidad de SQL, puede ser difícil de mantener los nombres en una base de datos. Sobre esta base, hay convenios vamos a utilizar para nuestros objetos. De hecho, vamos a adoptar las normas utilizadas en C / C + +, C #, Pascal, Java, y Visual Basic, etc en nuestras bases de datos:

- A menos que se indique de otro modo (se mencionan las excepciones, por ejemplo, con variables, tablas, etc), un nombre o bien se iniciará con una carta (a, b, c, d, e, f, g, h, i, j, k , l, m, n, o, p, q, r, s, t, u, v, w, x, y, z, A, B, C, D, E, F, G, H, I, J , K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y o Z) o un guión bajo
- Después del primer carácter, vamos a utilizar cualquier combinación de letras, dígitos, o de relieve
- Un nombre no se pondrá en marcha con dos de relieve
- Si el nombre es una combinación de palabras, por lo menos la segunda palabra, se iniciará en mayúsculas. Ejemplos de ello son **los países de Estadística, Encuesta Mundial, _RealSport, FullName,** o **DriversLicenseNumber**

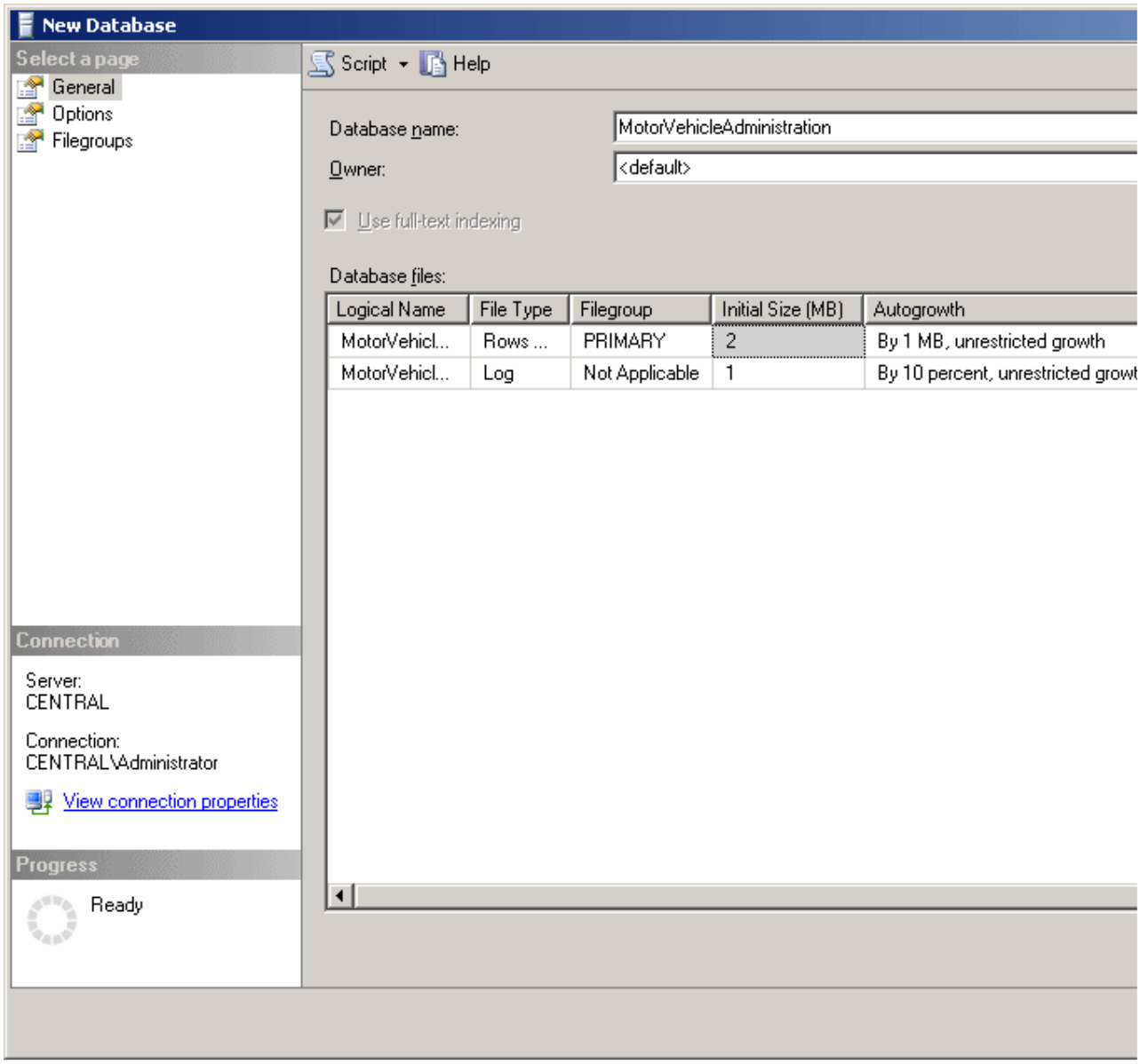
Después de crear un objeto cuyo nombre incluye el espacio, siempre que utilice ese objeto, incluya su nombre entre [y]. Ejemplos de ello son **[los países de Estadística], [Encuesta Mundial],** o **[Fecha de nacimiento].** Incluso si usted ha creado un objeto con un nombre que no incluye el espacio, al utilizar ese nombre, aún puede incluirlo entre corchetes. Ejemplos de ello son **[UnitedStations], [FullName], [DriversLicenseNumber],** y **[el país].**

Aprendizaje práctico: Puesta en marcha de la Gestión de Estudio

1. En el Explorador de objetos, haga clic derecho en las bases de datos y haga clic en Nueva base de datos ...



2. En el cuadro de texto Nombre, escriba **MotorVehicleAdministration**

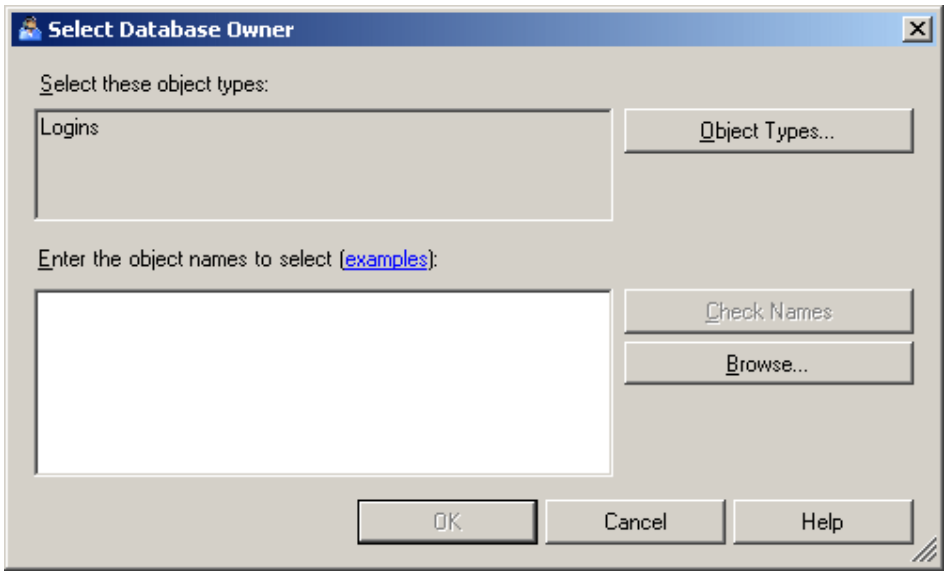


El propietario de una base de datos

Cuando una nueva base de datos es creada, el servidor quiere seguir la pista de la base de datos que creó. Esto se conoce como la base de datos propietario. De forma predeterminada, Microsoft SQL Server crea una cuenta especial denominada **dbo** (propietario de la base de datos). Al crear una base de datos, pero no especifica el propietario, esta cuenta se utiliza. El **dbo** También se da cuenta de derechos de todos los tipos de operaciones que se pueden realizar sobre la base de datos. Esto es conveniente en la mayoría de los casos. Sin embargo, si lo desea, puede especificar otro usuario como propietario de la base de datos. Por supuesto, la cuenta debe existir, lo que significa que debe haber creado previamente o puede usar una ya existente.

Para especificar visualmente el propietario de una base de datos que está creando, puede hacer clic en el propietario <default> cuadro de texto, escriba el nombre del dominio, seguido de la barra trasera, y seguido por el nombre de usuario que será el propietario de la base de datos. Alternativamente, puede hacer clic en el botón de puntos suspensivos en el lado derecho del propietario de cuadro de texto. Esto abre el cuadro de diálogo Seleccionar base de datos

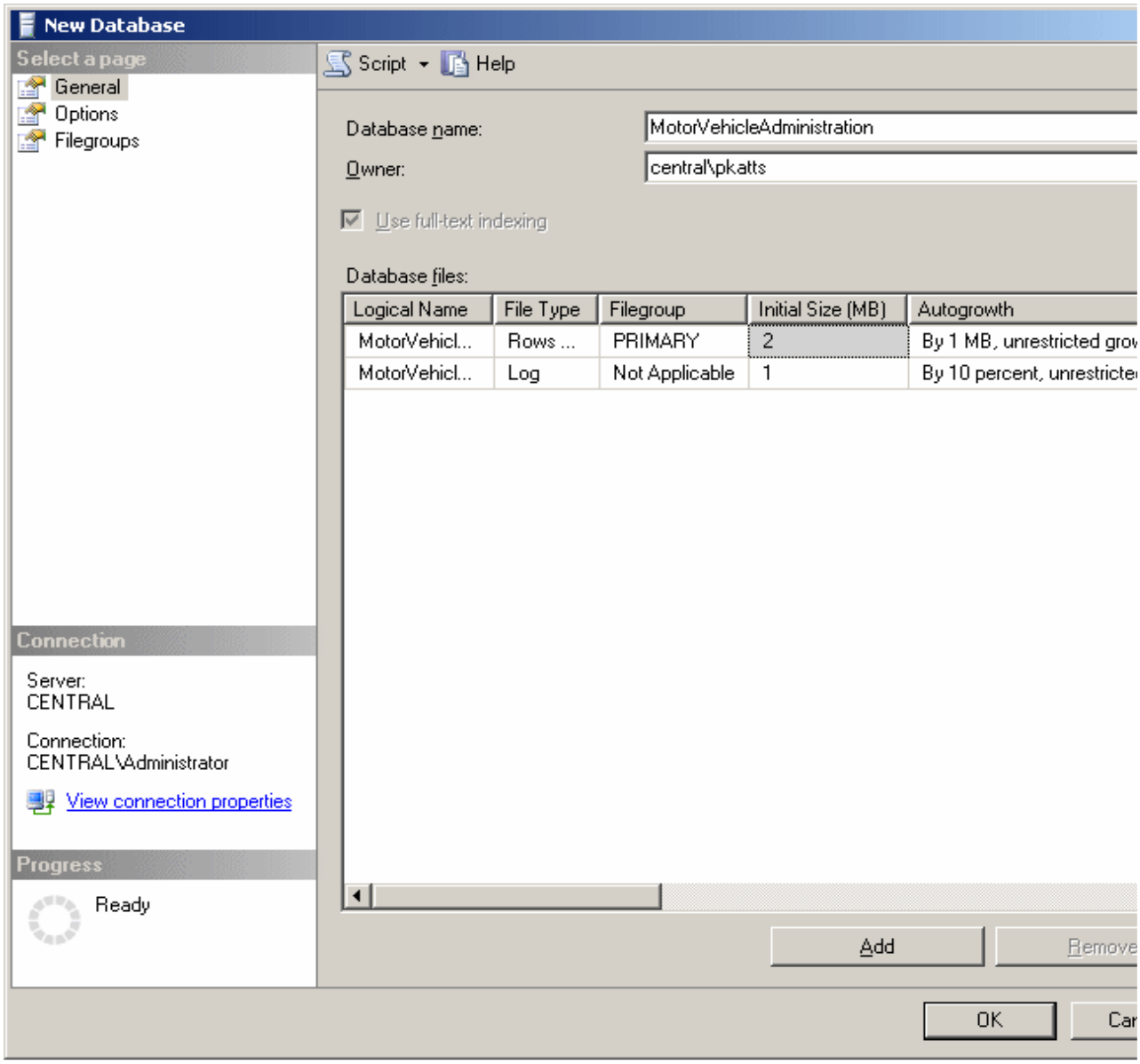
propietario de la casilla:



En el Escriba los nombres de objeto para seleccionar el cuadro de diálogo, introduzca el nombre completo o el nombre del usuario al que desea asignar la base de datos. Después de hacerlo, haga clic en Comprobar nombres. Si el nombre es correcto, el cuadro de diálogo que aceptarlo. Si el nombre no está bien, usted recibirá un mensaje de error. Puede hacer clic en el botón Examinar. Esto abriría las Personas para los objetos cuadro de diálogo. Si aparece el objeto de usuario que desea utilizar, haga clic en su casilla de verificación y haga clic en Aceptar.

Aprendizaje práctico: Especificación de la Base de Datos Propietario

- En el cuadro de diálogo propietario, haga clic en <default> y el tipo *nombreDeDominio* **\ pkatts** (nombreDeDominio reemplazar con el nombre de su dominio, de lo contrario, omita este paso) (también puede hacer clic en el botón en el lado derecho del propietario para localizar y seleccionar el nombre de usuario deseado)



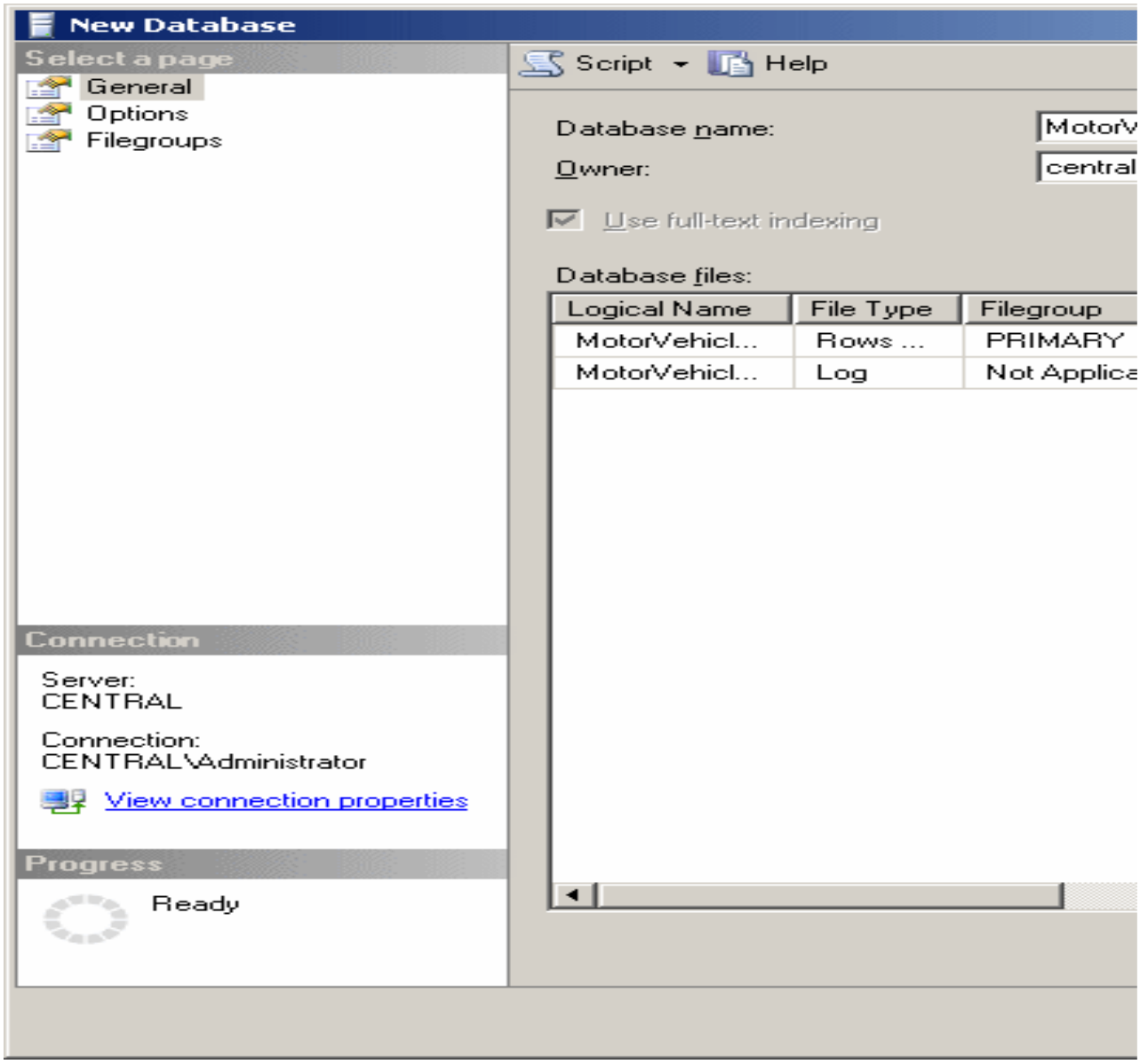
Tamaño de la primaria de una base de datos

Cuando originalmente la creación de una base de datos, usted puede o no saber cuántas listas, archivos, objetos o el proyecto tendría. Sin embargo, como usuario de la memoria de la computadora, la base de datos debe utilizar una cierta parte, al menos al principio. La cantidad de espacio que está utilizando una base de datos se refiere a su tamaño. Si utiliza el cuadro de diálogo Nueva base de datos, después de especificar el nombre de la base de datos y haga clic en Aceptar, el intérprete automáticamente especifica que la base de datos se utilizan principalmente 2MB. Esto es suficiente para iniciar una base de datos. Por supuesto, puede cambiar este valor predeterminado más adelante o puede aumentar cuando sea necesario.

Si desea especificar un tamaño diferente de la predeterminada, si está utilizando la nueva base de datos para crear su base de datos, archivos de base de datos en la sección inicial y bajo la columna Tamaño, cambie el tamaño que desee.

Aprendizaje práctico: Configuración de la base de datos Tamaño del archivo

- En la sección archivos de base de datos, haga clic en el cuadro Tamaño inicial bajo la cabecera de la columna, haga clic en la flecha hacia arriba del botón de girar y aumentar su valor a 5



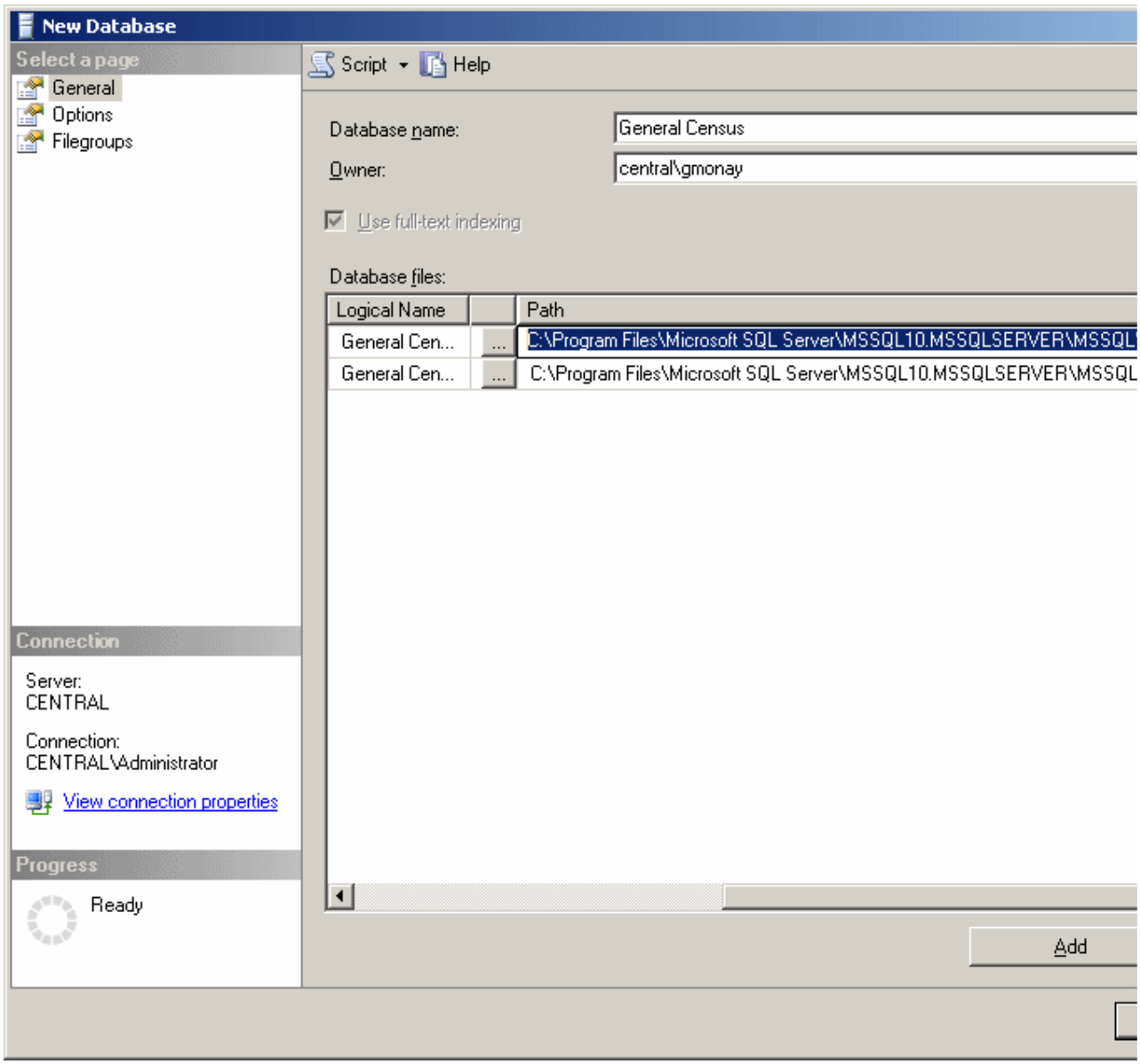
La ubicación de una base de datos

Como usted debe ser consciente de que ya a partir de su experiencia sobre el uso de ordenadores, cada ordenador debe tener un archivo de ruta. El camino es donde el archivo se encuentra en una de las unidades del ordenador. Esto permite al sistema operativo para saber donde está el archivo, de modo que cuando usted u otra aplicación que pide, el sistema operativo que no debe confundirse.

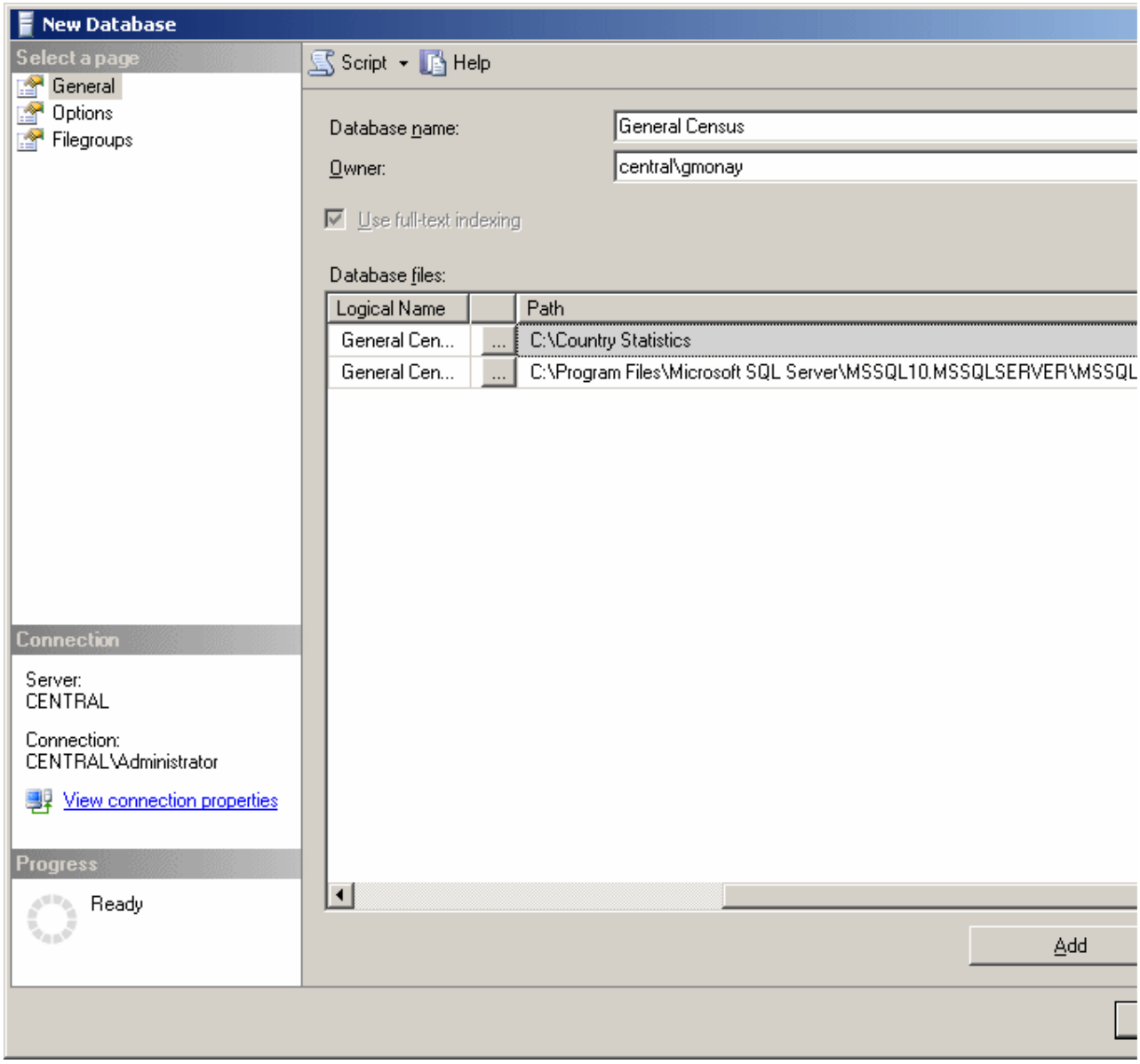
De forma predeterminada, al crear una nueva base de datos, Microsoft SQL Server supone que sería ubicado en *unidad:* \ Archivos de programa \ Microsoft SQL Server \ MSSQL10.MSSQLSERVER \ MSSQL \ DATA carpeta. Si utiliza el cuadro de diálogo Nueva base de datos de SQL Server Management Studio, si especifica el nombre de la base de datos y haga clic

en Aceptar, el intérprete crea automáticamente un nuevo archivo, y agrega el. MDF a la extensión de archivo: es el (principal), archivo de datos primario de su base de datos.

Si no desea utilizar la ruta por defecto, puedes cambiarlo. Si está utilizando el cuadro de diálogo Nueva base de datos, cambiar la ruta, bajo la cabecera de ruta, seleccione la actual serie de caracteres:



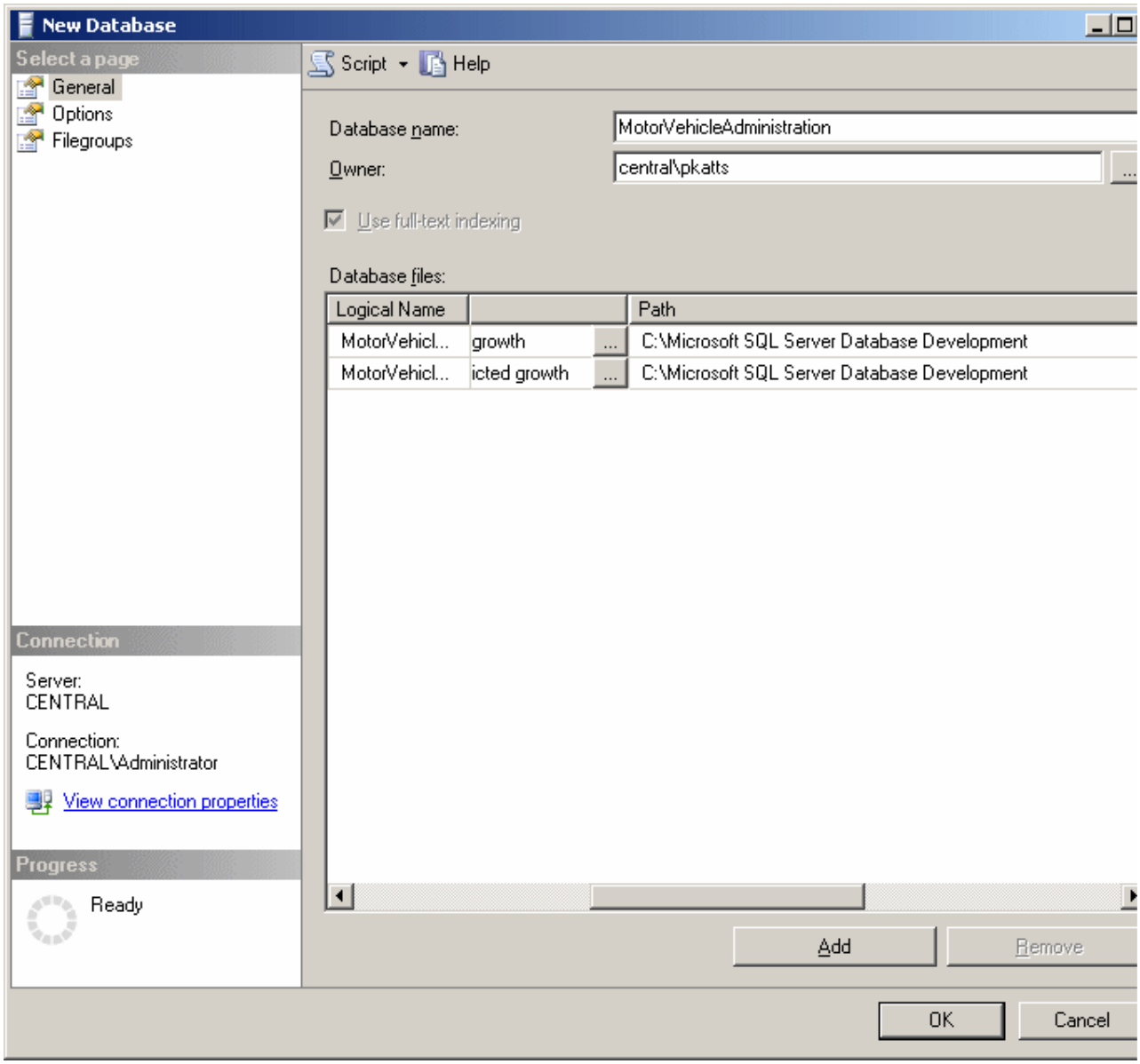
Sustituirla por una ruta de acceso de su elección



Aprendizaje práctico: Control de la Situación de los archivos de datos

1. Vaya a la derecha y, bajo la cabecera de ruta, aviso la ubicación del archivo
2. Inicie el Explorador de Windows
3. En el marco izquierdo, haga clic en la unidad C:
4. Haga clic derecho en un área en blanco en el marco derecho -> Nuevo -> Carpeta
5. Tipo **de base de datos de Microsoft SQL Server Desarrollo** como el nombre de la nueva carpeta
6. Volver a la nueva base de datos cuadro de diálogo.
   En virtud de Ruta, haga clic en el botón Examinar
7. Busque la base de datos de Microsoft SQL Server Desarrollo carpeta que ha creado y selecciónela
8. Haga lo mismo para el otro camino

9. Haga clic en Aceptar



Bases de datos por defecto

Introducción

Al instalar Microsoft SQL Server, también se instala 4 bases de datos llamado **maestro, modelo, msdb** y **tempdb.** Estas bases de datos serán de uso interno. Esto significa que se debe evitar el uso de ellos directamente, a menos que sepa exactamente lo que está haciendo.

El Sistema de Bases de Datos

Una de las bases de datos se instala con Microsoft SQL Server es nombrado **capitán.** Esta base de datos contiene toda la información sobre el servidor en el que el MS SQL Server está instalado. Por ejemplo, hemos visto anteriormente que, para llevar a cabo cualquier operación en el servidor, debe iniciar sesión. La base de datos master se identifica a cualquier persona,

llamada a un usuario, que accede a la base de datos, acerca de cuándo y cómo.

Además de la identificación de que accede el sistema, la base de datos **master** también realiza un seguimiento de todo lo que haces en el servidor, incluida la creación y gestión de bases de datos.

Usted no debe jugar con la base de datos master, de lo contrario puede dañar el sistema. Por ejemplo, si la base de datos master no está funcionando bien, el sistema no funcionaría.

Creación de bases de datos con el código

Introducción

El comando utilizado para crear una base de datos en SQL utiliza la siguiente fórmula:

CREATE DATABASE *DatabaseName*

Para ayudarle a escribir código, en la experiencia anterior, vimos que se puede usar la ventana de consulta.

Los comandos CREATE DATABASE (recuerde que SQL no es sensible a mayúsculas y minúsculas) de expresión es necesaria. El factor de *DatabaseName* es el nombre que la nueva base de datos tendrá. Aunque SQL no es sensible a mayúsculas y minúsculas, debe hacer un hábito que debe conocer de los casos el nombre que utiliza para sus objetos. Cada declaración de SQL puede ser terminado con un punto y coma. Aunque se trata de un requisito en muchas implementaciones de SQL en Microsoft SQL Server, puede omitir el punto y coma. En caso contrario, la fórmula anterior sería

CREATE DATABASE *DatabaseName;*

Aquí está un ejemplo:

CREATE DATABASE NationalCensus;

Esta fórmula se utiliza si no quiere dar ninguna opción. Vimos anteriormente que una base de datos tiene uno o más archivos y vimos que se encuentran por defauft. También vieron que se puede especificar la ubicación de los archivos si lo desea. Para especificar que el principal archivo de la base de datos se encuentra, puede usar la siguiente fórmula:

CREATE DATABASE *DatabaseName*
DE PRIMARIA
(NAME = *LogicalName,* FILENAME = *Camino)*

Los tres únicos factores cuyos valores tienen que ser cambiado a partir de esta fórmula son el nombre de base de datos que ya hemos visto, el nombre lógico, y el nombre de ruta. El nombre lógico puede ser cualquier palabra de un nombre, sino que debe ser diferente del nombre de base de datos. La ruta es la ubicación del directorio del archivo. Este sendero termina con un nombre para el archivo con la extensión. Mdf. El camino debe ser completa y se incluyen en una sola cita. Aquí está un ejemplo:

CREATE DATABASE NationalCensus
DE PRIMARIA
(NAME = DataRepository, FILENAME = 'C: \ Ejercicios \ NationalCensus.mdf')
IR

Además del archivo principal, es posible que desee crear y almacenar un archivo de registro. Para especificar que el archivo de registro de la base de datos se encuentra, puede usar la siguiente fórmula:

CREATE DATABASE *DatabaseName*
DE PRIMARIA

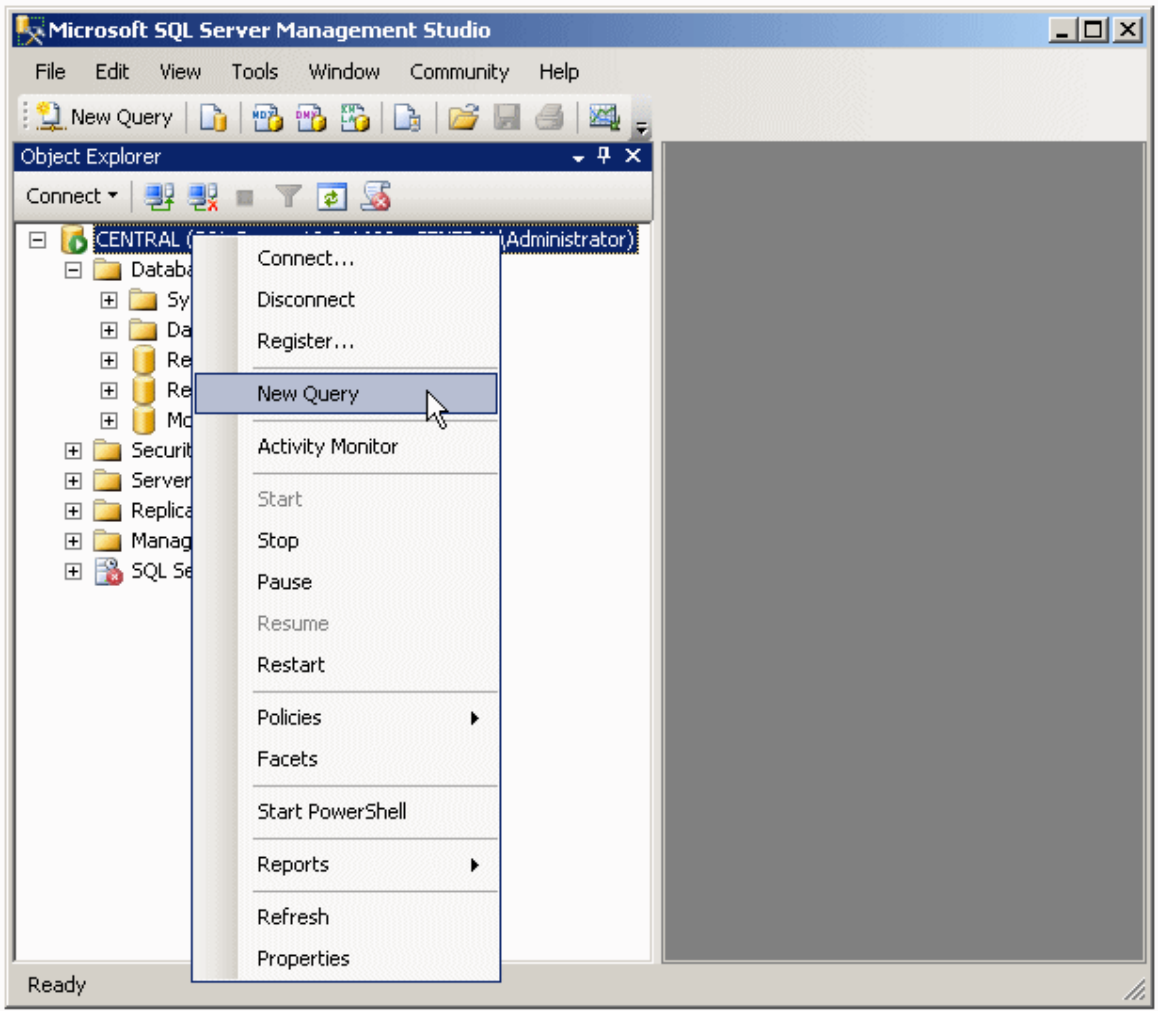
(NAME = *LogicalName,* FILENAME = *Camino.* Mdf)
REGISTRO DE
(NAME = *LogicalName,* FILENAME = *Camino.* Ldf)

El nuevo factor en esta fórmula es la ruta de acceso del archivo de registro. Al igual que el archivo principal, el archivo de registro debe ser llamado (con un nombre lógico). El camino termina con un nombre de archivo cuya extensión es. Ldf. Aquí está un ejemplo:

CREATE DATABASE NationalCensus
DE PRIMARIA
(NAME = DataRepository, FILENAME = 'C: \ Ejercicios \ NationalCensus.mdf')
REGISTRO DE
(NAME = DATALOG, FILENAME = 'C: \ Ejercicios \ NationalCensus.ldf')
IR

Aprendizaje práctico: Creación de una base de datos usando SQL

1. Para abrir el editor de código, en el Explorador de objetos, haga clic derecho en el nombre del servidor y haga clic en Nueva consulta

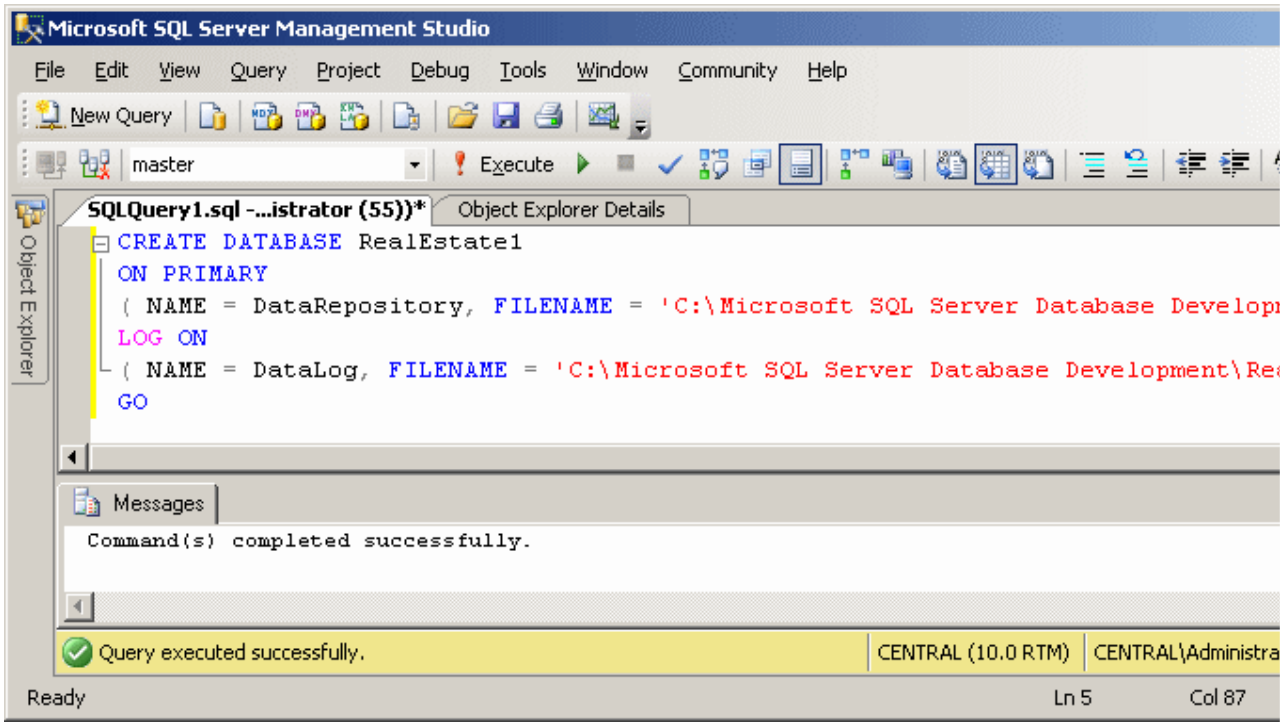


2. En la ventana vacía, escriba:

CREATE DATABASE RealEstate1
DE PRIMARIA
(NAME = DataRepository, FILENAME = 'C: \ Microsoft SQL Server Base de datos

Desarrollo \ RealEstate1.mdf')
REGISTRO DE
(NAME = DATALOG, FILENAME = 'C: \ Microsoft SQL Server Base de datos
Desarrollo \ RealEstate1.ldf')
IR

3. Para ejecutar la declaración, presione F5



Uso de Código de plantilla

Para especificar más opciones con el código, Microsoft SQL Server se suministra con varios códigos de ejemplo que puede utilizar para diferentes tareas. Por ejemplo, puede usar código de ejemplo para crear una base de datos. La muestra que los códigos de Microsoft SQL Server son accesibles desde el Explorador de plantillas.

Para acceder a la plantilla de Explorer, en el menú principal, puede hacer clic en Ver -> Explorador de plantillas. Antes de crear una base de datos, abrir una nueva ventana de consulta. A continuación:

- Para crear una nueva base de datos utilizando el código de muestra, en la plantilla Explorer, expanda el nodo Bases de datos, a continuación, arrastre el nodo de base de datos Crear y suéltelo en la ventana de consulta. La nueva base de datos se creó en el servidor que mantiene la conexión actual
- Si tiene acceso a más de un servidor, crear una base de datos en otro servidor o con otro sentido, en la plantilla Explorer, expanda el nodo Bases de datos, haga clic en Crear base de datos y haga clic en Abrir. En el motor de base de datos para conectar el cuadro de diálogo, seleccione las opciones apropiadas, y haga clic en Aceptar

Con cualquiera de estas acciones, Microsoft SQL Server, se generará el código de muestra para usted:

- ==========
- Crear base de datos de plantilla

```
- ==========================================
USO maestro
IR

- Excluir de la base de datos si ya existe
IF EXISTS (
        SELECT name
                DE sys.databases
                WHERE name = N '<Database_Name, sysname, Database_Name>'
),
DROP DATABASE <Database_Name, sysname, Database_Name>
IR

CREATE DATABASE <Database_Name, sysname, Database_Name>
IR
```

Usted tendría que editar y ejecutar el código para crear la base de datos. A partir de la experiencia anterior y de los artículos, hemos revisado algunos caracteres ran como las observaciones - y algunas palabras o expresiones tales como **GO, CREATE DATABASE, y** SELECT. Vamos a estudiar las otras palabras o expresiones en el futuro las lecciones y secciones.

Base de datos de mantenimiento

Introducción

Si ha creado una base de datos, pero no necesita más, puede eliminarla. Es importante saber, independientemente de cómo crear una base de datos, ya sea utilizando SQL Server Management Studio, el código en la ventana de consulta, o el símbolo del sistema, cada base de datos se puede acceder por cualquiera de estas herramientas y se puede eliminar cualquiera de las bases de datos utilizando cualquiera de estas herramientas.

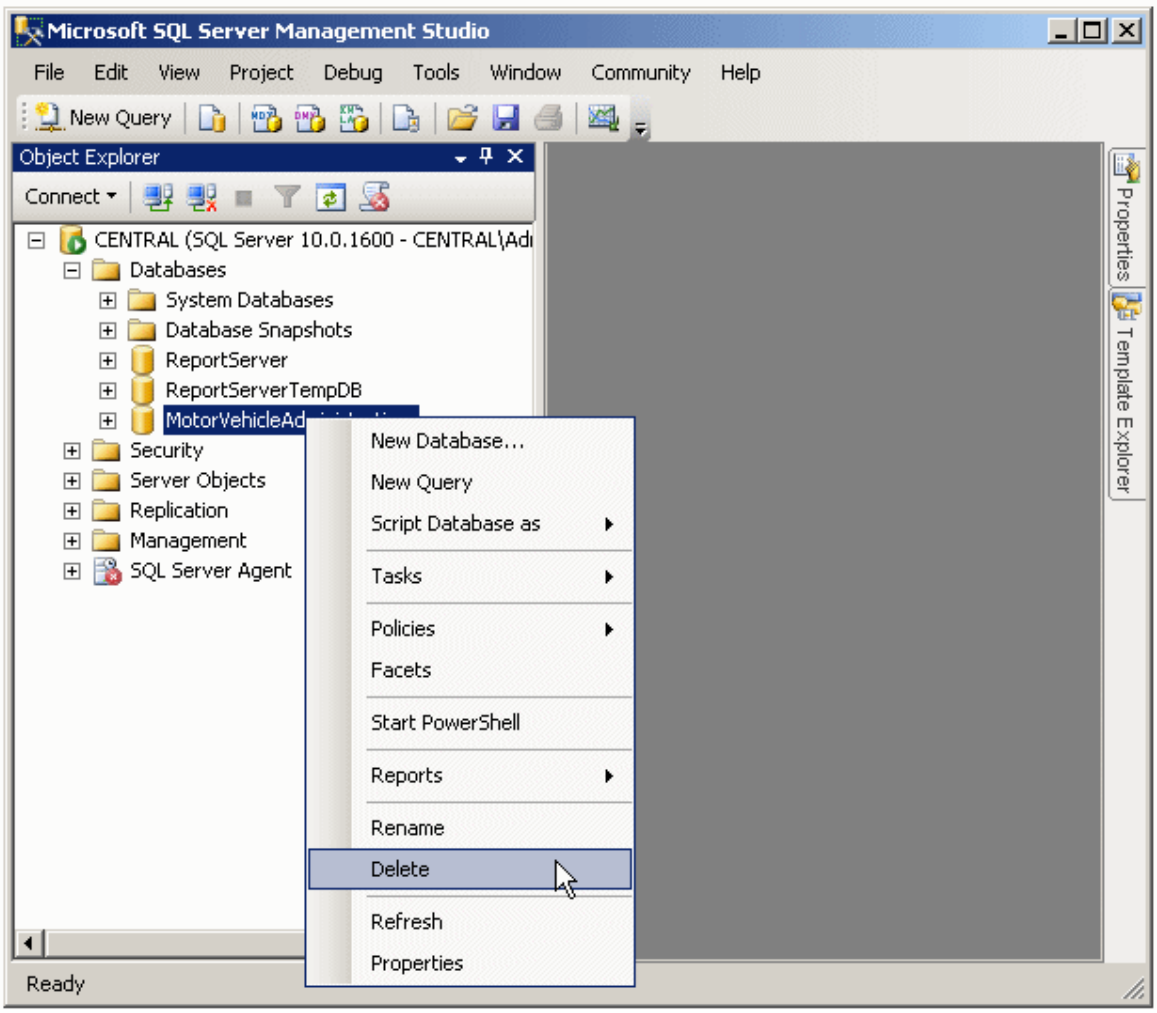
Como se hizo con la creación de una base de datos, todas las herramientas a disposición de sus propios medios.

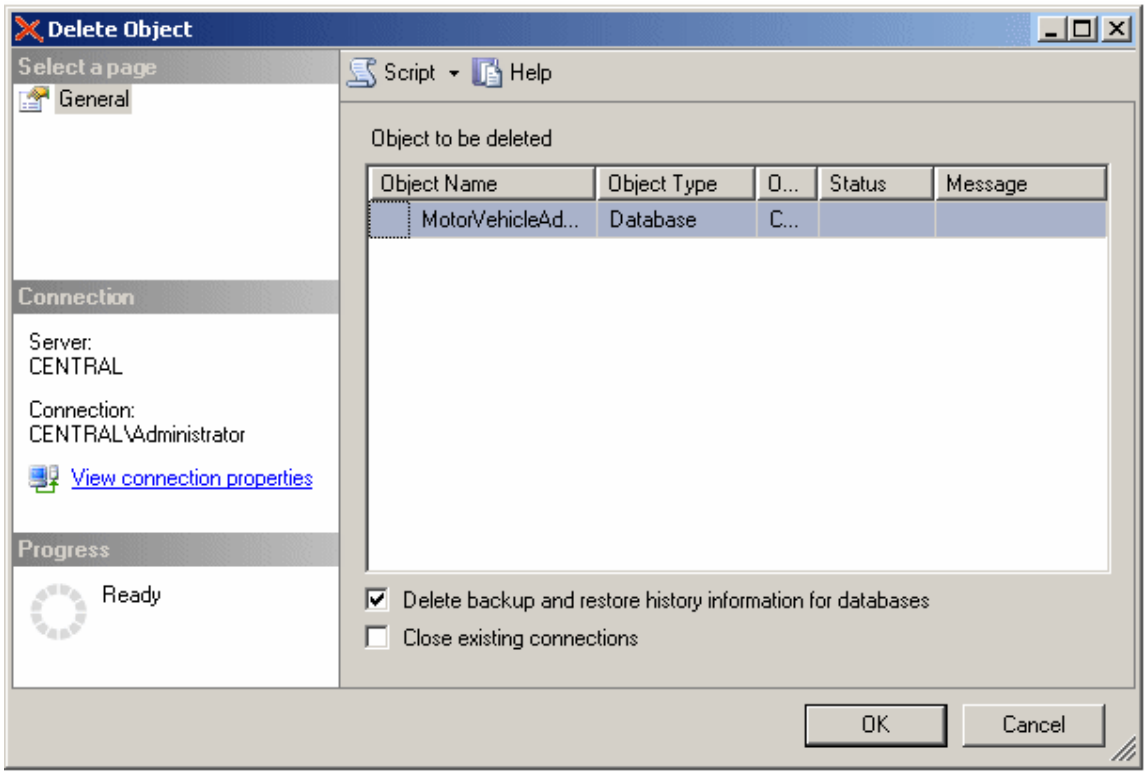
SQL Server Management Studio

Para eliminar una base de datos en SQL Server Management Studio, en el Explorador de objetos, expanda el nodo Bases de datos, haga clic derecho en la base de datos mal, y haga clic en Eliminar. Un cuadro de diálogo que le pedirá que confirme su intención. Si aún desea eliminar la base de datos, puede hacer clic en Aceptar. Si cambia de opinión, puede hacer clic en Cancelar.

Aprendizaje práctico: Eliminar una Base de Datos

1. En el Explorador de objetos, haga clic derecho y haga clic en Eliminar **MotorVehicleAdministration**


Microsoft SQL Server Management Studio
File
Edit
View
Project
Debug
Tools
Window
Community
Help
New Query
Object Explorer
Connect
CENTRAL (SQL Server 10.0.1600 - CENTRAL\Adi
Databases
System Databases
Database Snapshots
ReportServer
ReportServerTempDB
MotorVehicleAd
Security
Server Objects
Replication
Management
SQL Server Agent
New Database...
New Query
Script Database as
Tasks
Policies
Facets
Start PowerShell
Reports
Rename
Delete
Refresh
Properties
Ready
Properties
Template Explorer



2. En el cuadro de diálogo Eliminar objetos, haga clic en Aceptar

Eliminación de una base de datos usando SQL

Para eliminar una base de datos en el Analizador de consultas SQL, se utiliza la expresión **de DROP DATABASE** seguida del nombre de la base de datos. La fórmula utilizada es la siguiente:

DROP DATABASE *DatabaseName;*

Antes de borrar una base de datos en SQL, debe asegurarse de la base de datos no está siendo utilizado o visitado por alguien más, o por otro objeto.

Aprendizaje práctico: Eliminar una Base de Datos con el código

1. En la barra de herramientas Estándar, haga clic en el botón Nueva consulta

2. Para eliminar una base de datos, tipo:

   DROP DATABASE RealEstate1;
   IR

3. Presione la tecla F5 para ejecutar la declaración

Rutinas de base de datos

La actual base de datos

Si bien la escritura de código en una ventana de consulta, siempre debe saber qué base de datos que está trabajando, de lo contrario puede añadir código a la base de datos incorrecto. Programación especificar la base de datos actual, escriba la palabra clave **USO** seguida del nombre de la base de datos. La fórmula a utilizar es:

USO *DatabaseName;*

Aquí está un ejemplo:

USO GovernmentStatistics;

Actualizar la lista de las bases de datos

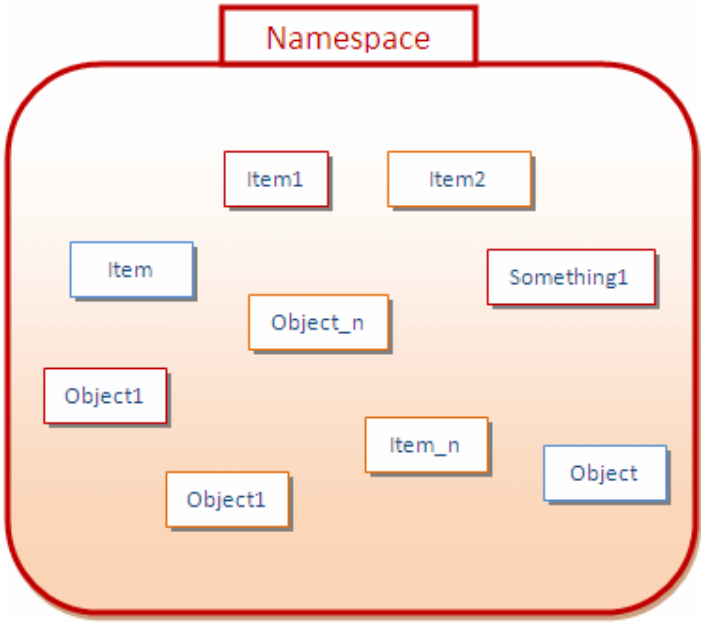
Algunas de las ventanas que muestran las bases de datos, como SQL Server Management Studio, no actualiza su lista de inmediato si una operación se produjo fuera de su confinamiento. Por ejemplo, si crea una base de datos en la consulta ventanas, su nombre no se actualiza en el Explorador de objetos. Para ver esos cambios externos, puede refrescar la ventana que contiene la lista.

En SQL Server Management Studio, para actualizar una lista, puede hacer clic derecho en su categoría en el Explorador de objetos y haga clic en Actualizar. Sólo de esa categoría puede ser renovado. Por ejemplo, para actualizar la lista de bases de datos, en el Explorador de objetos, puede hacer clic derecho en el nodo Bases de datos y haga clic en Actualizar.

Schemas

Introducción a los Namespaces

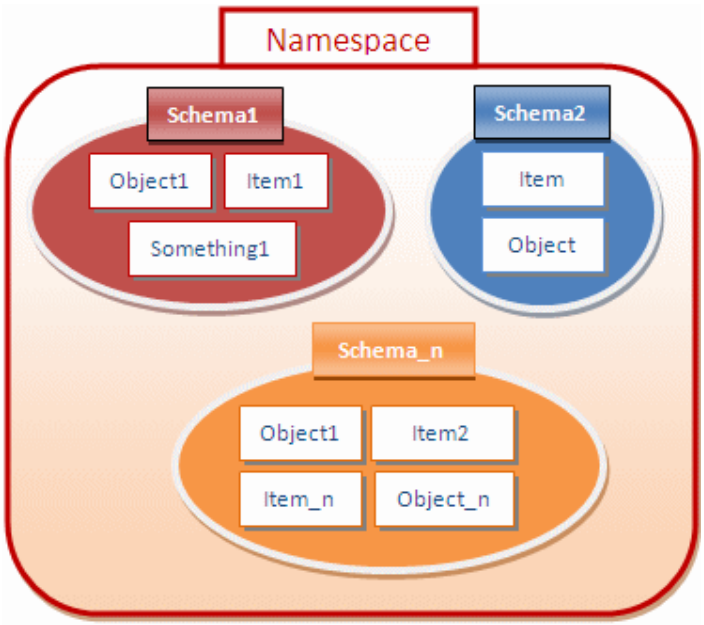
Un nombre es una técnica de creación de una serie de elementos que cada uno tiene un nombre único. Por ejemplo, si se inicia la creación de numerosas bases de datos, existe la posibilidad de que puede haber riesgo de diversas bases de datos con el mismo nombre. Si se utiliza un espacio de nombres, puede aislar las bases de datos en varios nombres. En realidad, para gestionar muchos otros aspectos de su servidor de base de datos, que utiliza nombres y colocar objetos, distintas de las bases de datos, dentro de estos espacios de nombres. Por lo tanto, un espacio de nombres y su contenido puede ilustrarse de la siguiente manera:



Observe que hay varios tipos de objetos dentro de un espacio de nombres.

Introducción a los esquemas

Dentro de un espacio de nombres, puede crear los objetos que desee. A fin de controlar y administrar los objetos dentro de un espacio de nombres, puede poner en subgrupos llamados esquemas. Por lo tanto, un esquema es un grupo de objetos dentro de un espacio de nombres. Esto también significa que, dentro de un espacio de nombres, puede tener tantos esquemas como desee:



Observe que, al igual que un espacio de nombres puede contener objetos (esquemas), un esquema puede contener también los objetos (los objetos que se creará a lo largo de nuestra experiencia).

Para gestionar los esquemas en un espacio de nombres, se necesita una manera de identificar a cada esquema. Sobre esta base, cada esquema debe tener un nombre. En nuestro ejemplo, un esquema se denomina Schema1. Otro esquema se denomina Schema2. Sin embargo, otro esquema se denomina Schema_n.

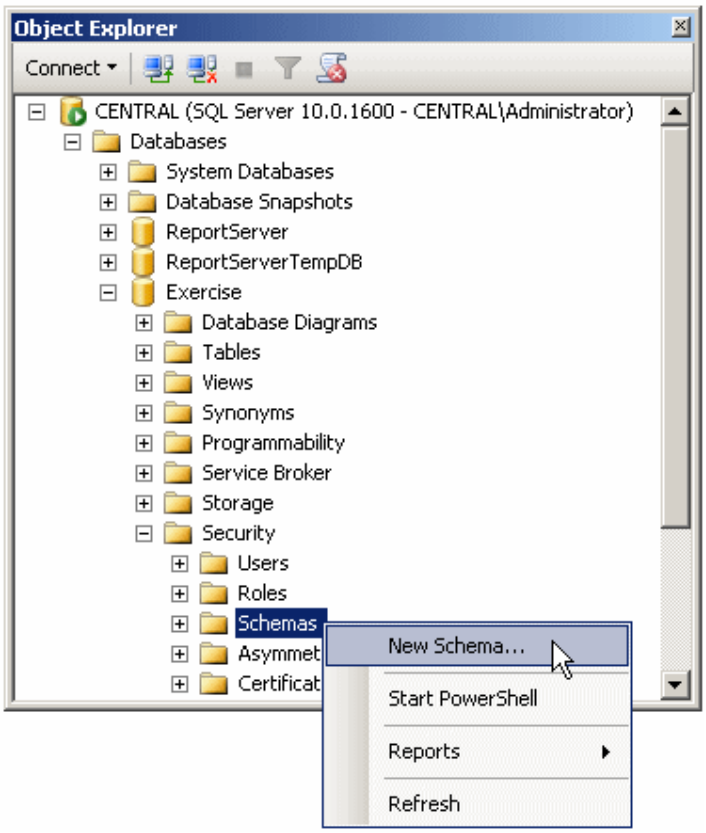
Un esquema es un objeto que contiene otros objetos. Antes de usarlo, debe crear o puede utilizar un esquema existente. Hay dos tipos de esquemas se pueden utilizar, los incorporados y los que cree. Cuando Microsoft SQL Server está instalado, también crea unos esquemas. Uno de los esquemas que se llama **sistema.**

El **sistema** de esquema contiene una lista de algunos de los objetos que existen en su sistema. Uno de estos objetos se conoce como **bases de datos** (en realidad, es una opinión). Al crear una base de datos, su nombre se consignan en las **bases** de **datos,** con el mismo nombre que le dieron.

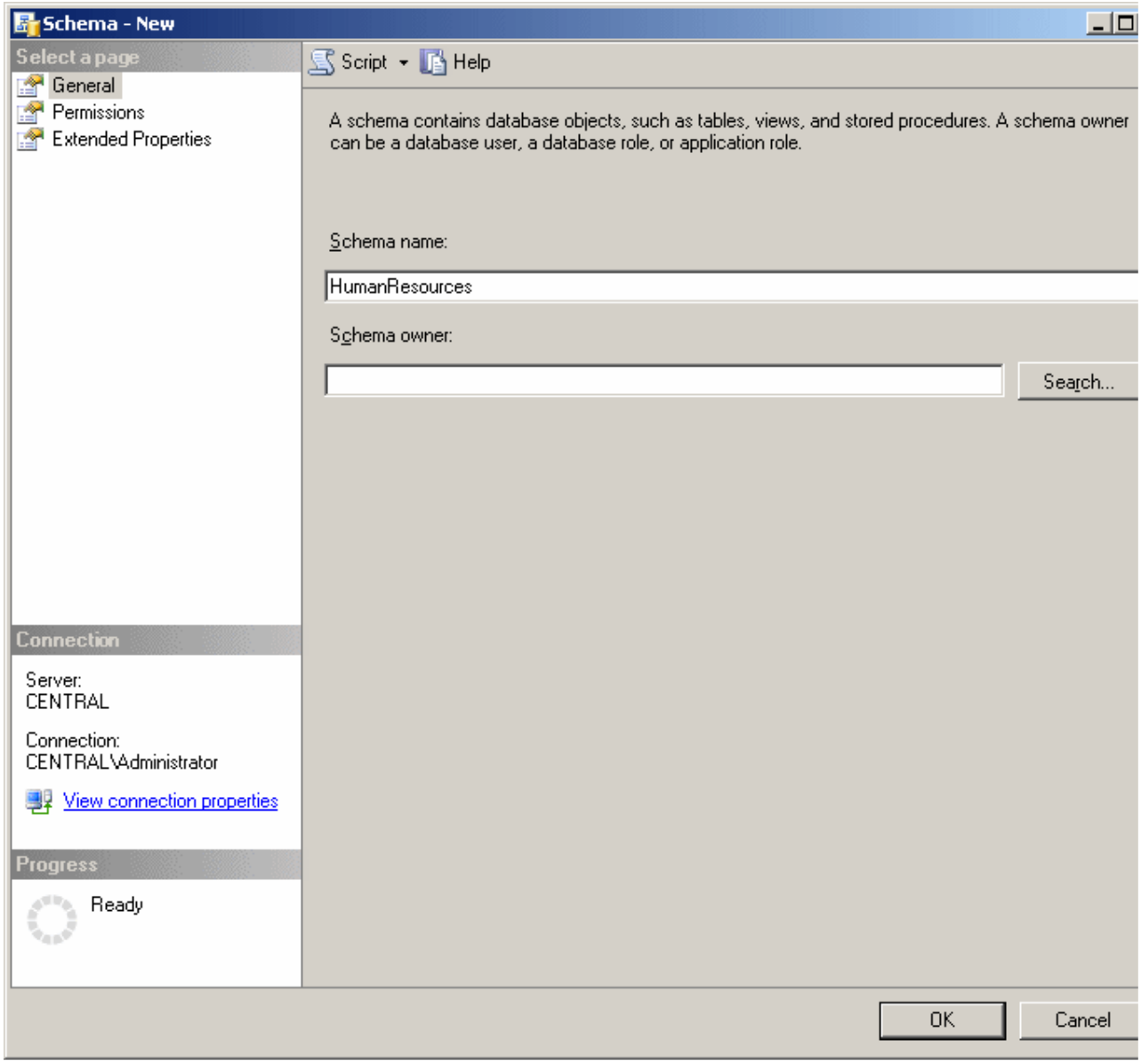
Para acceder a los esquemas de una base de datos, en el Explorador de objetos, expanda el nodo Bases de datos, ampliar la base de datos que contendrá el esquema o propia, y ampliar la Seguridad nodo.

Creación de un esquema

Para crear un esquema, haga clic derecho en esquemas de esquema y haga clic en Nuevo ...



Esto abriría el esquema - Nuevo cuadro de diálogo. En el esquema de cuadro de texto Nombre, escriba un nombre de una palabra. Aquí está un ejemplo:



Después de proporcionar un nombre, usted puede hacer clic en Aceptar.

Vamos a ver un ejemplo práctico de la creación de un esquema en la Lección 7.

Acceso a un objeto de un esquema

Dentro de un esquema, dos objetos no pueden tener el mismo nombre, sino un objeto en un esquema puede tener el mismo nombre que un objeto en otro esquema. Basado en esto, si usted tiene acceso a un objeto dentro de su esquema, puede simplemente usar su nombre, ya que este nombre sea único. Por otra parte, debido a la posibilidad implícita de hacer frente a los objetos con nombres similares en el servidor, al acceder a un objeto fuera de su esquema, debe calificar la misma. Para ello, escriba el nombre del esquema que contiene el objeto que desea utilizar, seguido por el operador período, seguido por el nombre del objeto que desea utilizar. De nuestro ejemplo, para acceder a la Something1 objeto que pertenece a Schema1, debería escribir:

Schema1.Something1

Resumen de la lección

Ejercicio: Utilidad de la empresa

- Utilizando Microsoft SQL Server Management Studio, crear una base de datos llamada **UtilityCompany1** (Acepte la configuración predeterminada)

Ejercicio: EE.UU. Regiones

1. Realizar algunas investigaciones en la Internet para averiguar acerca de las diferentes regiones de los Estados Unidos. En una hoja de papel, crear una lista de las regiones de los EE.UU.
2. Conectarse al servidor desde la línea de comandos
3. Desde el símbolo del sistema, crear una base de datos llamada **UnitedStatesRegions1**
4. Salga del símbolo del

Introducción

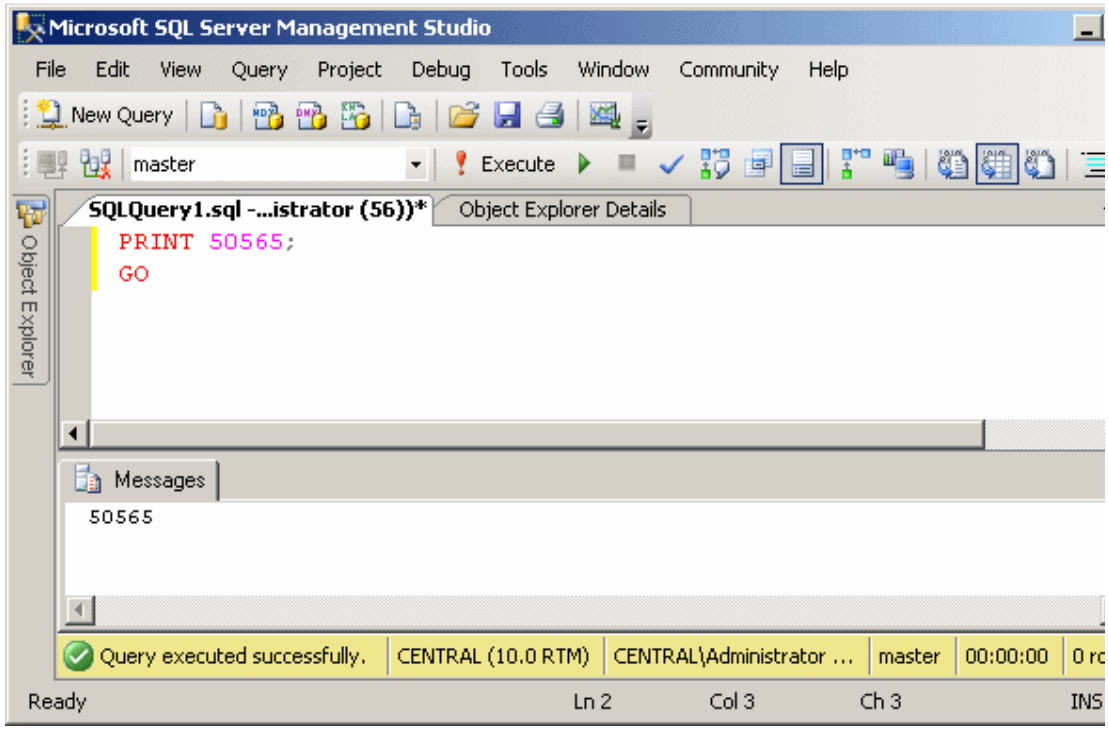
Una operación es una acción realizada en uno o más valores, ya sea para modificar el valor en poder de una o ambas de las variables o para producir un nuevo valor mediante la combinación de valores. Por lo tanto, una operación se realiza mediante el uso de al menos un símbolo y un valor. El símbolo utilizado en una operación que se llama un operador. Un valor que participan en una operación que se llama un operando.

Algo IMPRIMIR

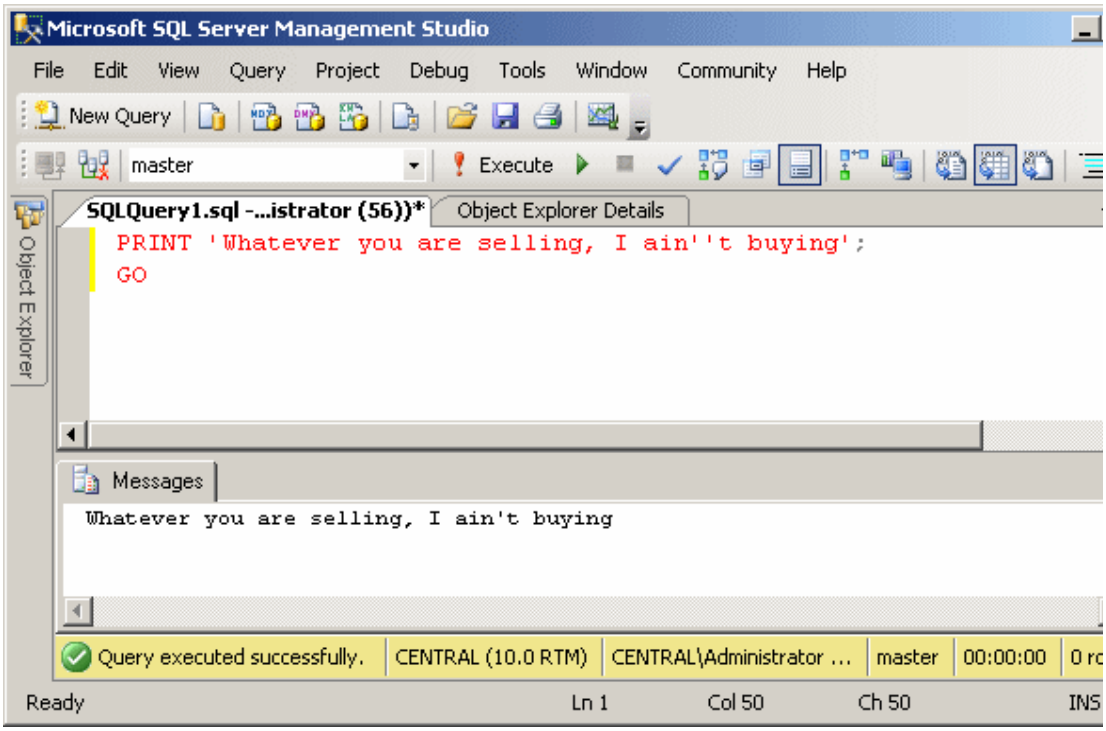
Al igual que todos los idiomas, SQL algunas palabras con los buques utilizados para el transporte de sus diversas operaciones. Una de estas palabras es IMPRIMIR. Para ver algo en texto plano como resultado de una declaración, seguida de **IMPRIMIR** tipo que desea mostrar. Por lo tanto, **IMPRIMIR** utiliza la siguiente fórmula:

IMPRIMIR *WhatToPrint*

Para mostrar el tema puede ser cualquier cosa que se permite y se dispone en el lado derecho de **IMPRIMIR.** Si es un número constante, simplemente escriba en el lado derecho de **IMPRIMIR.** Aquí está un ejemplo:



El tema para mostrar también puede ser una operación o el resultado de una operación. Si desea mo
una palabra o una frase, solo incluirlo entre comillas. Si desea incluir una sola cita en su declaración,
decir, escribir dos veces. Aquí está un ejemplo:



También puede mostrar una expresión como una combinación de número (s) y frases como veremos m

SELECCIONE

Seleccionar nada

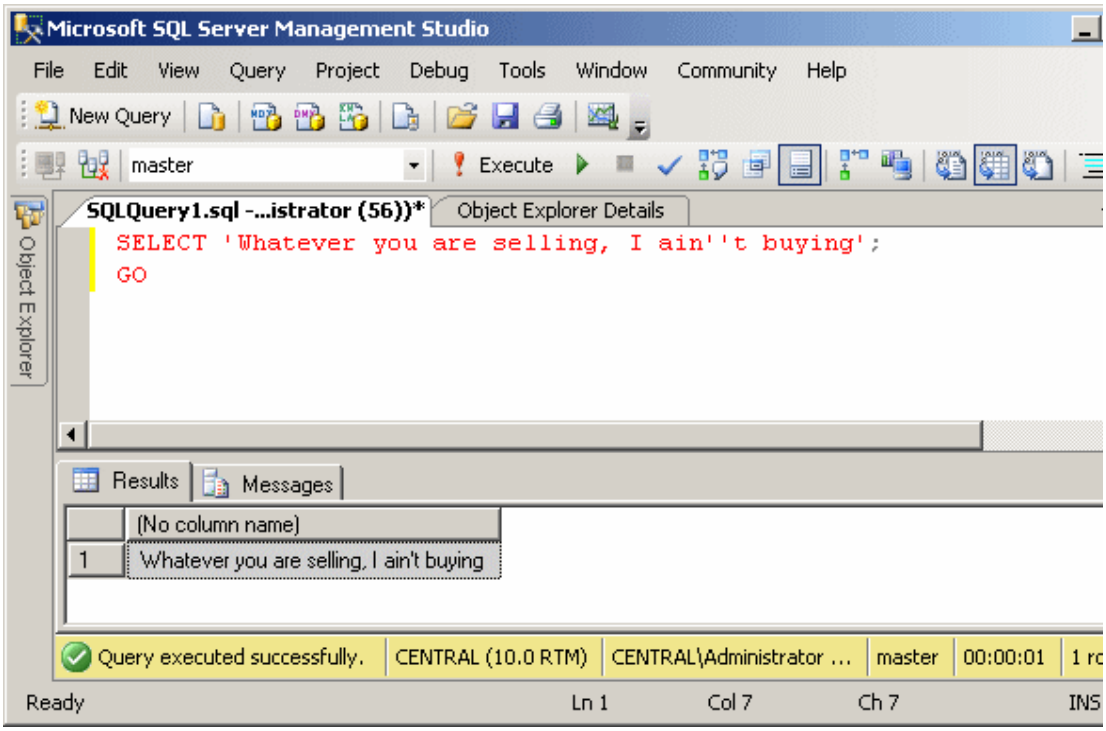
El operador de **selección** se pueden utilizar, entre otras cosas, para mostrar un valor. La **selección** utiliza la sintaxis siguiente:

Seleccione *lo*

Sobre esta base, su uso, cuando sea necesario, el tipo **SELECCIONE** seguido de un número, una palab una expresión. El siguiente tema para mostrar algunas de las mismas normas que **IMPRESIÓN.** Una entre **IMPRIMIR y** SELECT es que:

- **IMPRIMIR** se utiliza para la prueba un simple valor, una cadena, o una expresión. Por lo tanto, n resultados en una ventana en blanco en virtud de una pestaña Mensajes. IMPRIMIR se puede utiliz valor
- **SELECT** es el más utilizado SQL operador. Vamos a ver que se utiliza para recuperar los registros Por esta razón, **SELECCIONE** muestra sus resultados en una ventana organizada de categorías der columnas, en una pestaña Resultados. **SELECCIONE** se puede utilizar con más de un valor

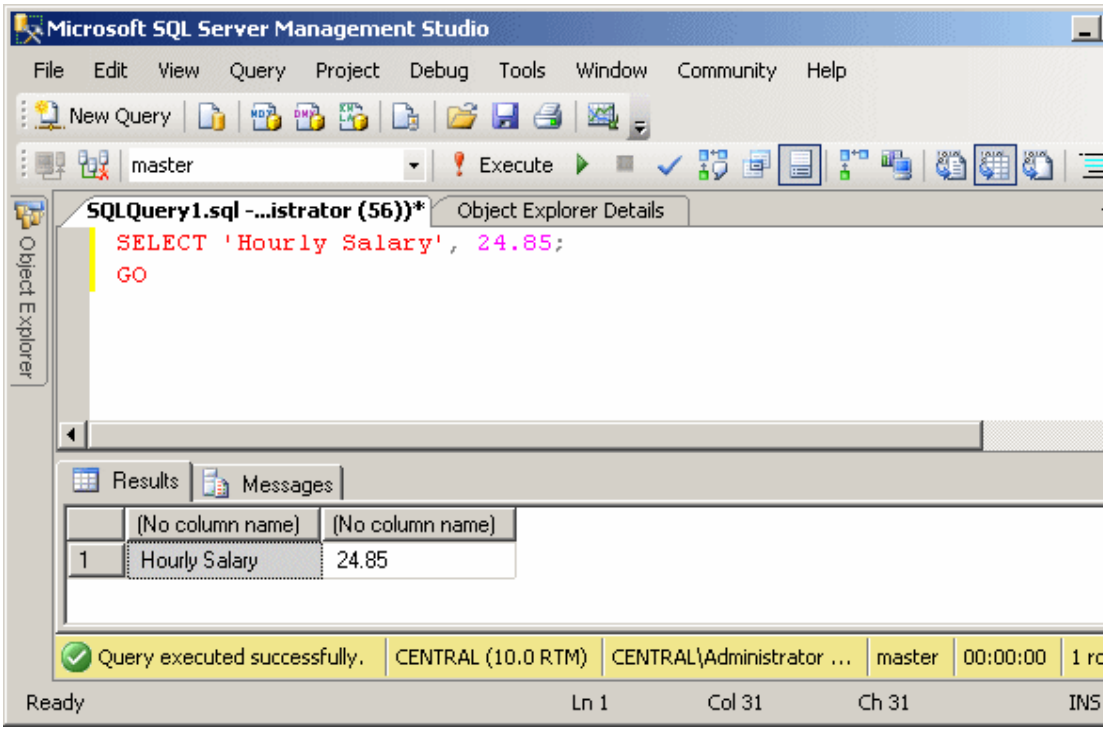
Como hacer **para** la impresión, para mostrar una frase utilizando **SELECT,** escriba en una sola cita en de este operador. Aquí está un ejemplo de ejecución:



Como ya se ha mencionado, a diferencia de imprimir, seleccione puede ser usado para mostrar más valores deben estar separados por comas. Aquí está un ejemplo:

SELECT 'Salario por hora', 24,85;

Esto produciría:



De anidación de un SELECT

Cuando se crea una **sentencia SELECT,** lo que hay en el lado derecho de la **selección** debe ser un va ejemplo:

SELECCIONE 226,75;

Basándose en esta definición, en lugar de ser sólo un valor, la cosa en el lado derecho de la **selecció** de producir un valor. Como veremos en las siguientes secciones, se puede crear operación algeb derecha de SELECT. Debido a que mencionó que la cosa en el lado derecho debe producir un resultad utilizar otro SELECT que se evalúa a un resultado. Para distinguir el SELECT secciones, la segunda debe paréntesis. Aquí está un ejemplo:

SELECT (SELECT 448,25);
IR

Cuando una **sentencia SELECT** es creado después de otro, el segundo se conoce como anidados.

Así como usted puede anidar un comando SELECT dentro de otro, también puede anidar una de declaración que en sí es anidados. Aquí está un ejemplo:

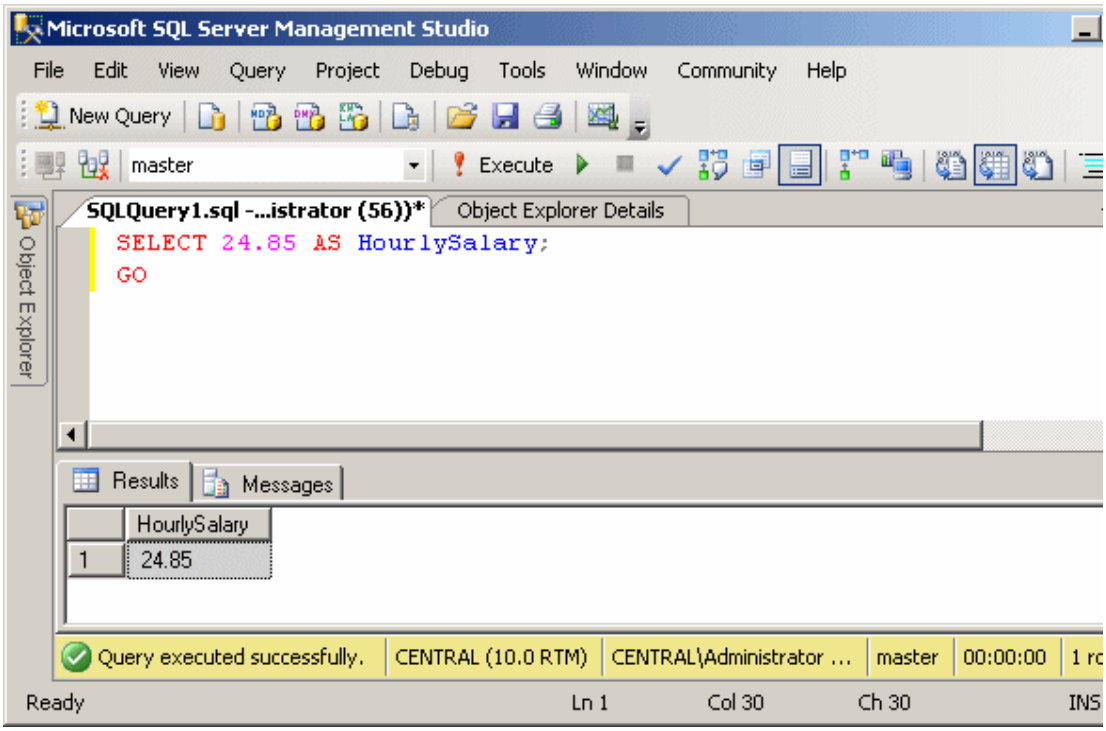
SELECT (SELECT (SELECT 1350,75));
IR

Esta selección que el

En la presentación, hemos utilizado bien **IMPRIMIR o** SELECT para mostrar algo en la ventana de con características de **selección** es que puede resultar en su segmento de diferentes secciones. **SELECC** cada uno de los valores en una sección llamada una columna. Cada columna se representa con un non se llama leyenda. De forma predeterminada, el título se muestra como "(no nombre de columna)". Si propia leyenda, en el lado derecho de una expresión, **como** el tipo de palabras clave seguidas por el títu parte derecha de **la** AS palabra clave debe ser considerado como una palabra. Aquí está un ejemplo:

24,85 SELECCIONE COMO HourlySalary;

Esto produciría:



También puede incluir el tema en el lado derecho de **AS** en una sola cita. Aquí está un ejemplo:

24,85 SELECCIONE COMO 'HourlySalary';

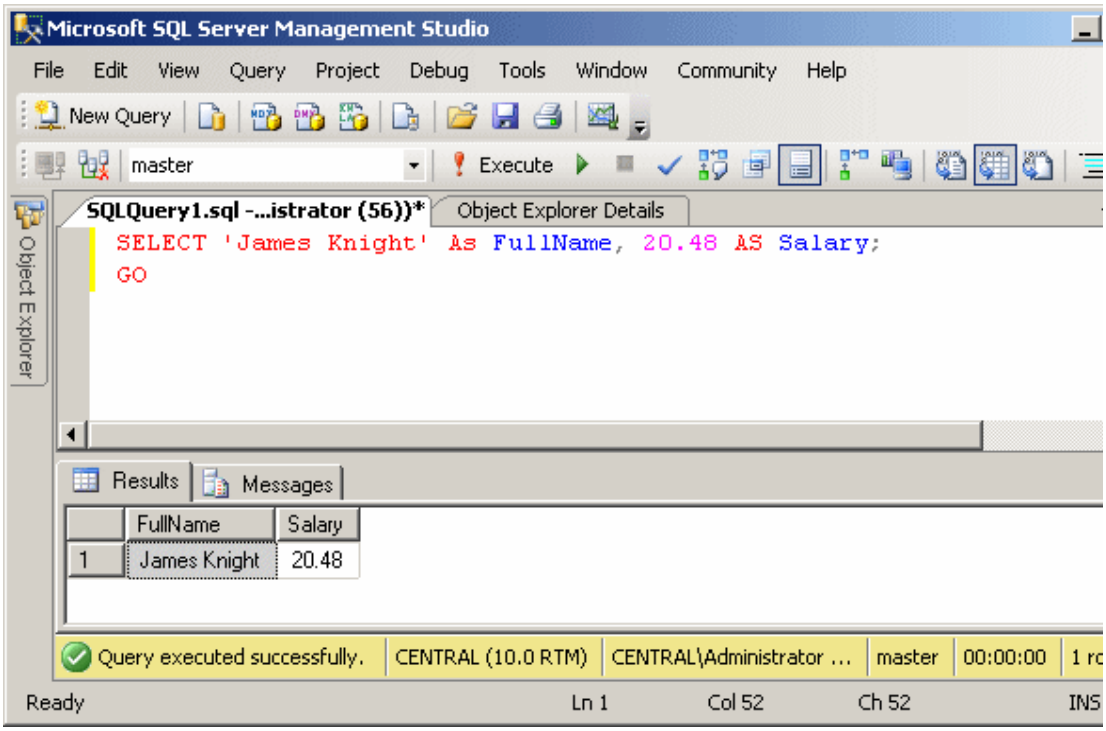
Si el elemento en el lado derecho de **AS** es en otras palabras, usted debe incluir en una sola cita o pone de una apertura de corchete "[" y un cierre de corchete "]". Aquí está un ejemplo:

24,85 SELECCIONE COMO "Salario por hora ';

Si crea diferentes secciones, separadas por una coma, puede seguir cada uno con **AS** y un subtítul ejemplo:

SELECT 'James Knight' Como FullName, 20,48 en concepto de salarios;

Esto produciría:



La declaración anterior podría también ser escrito como sigue:

SELECT 'James Knight' Como [Nombre Completo], 20,48 COMO [Salario por hora];

Los operadores unario

El operador positivos +

Un operador unario es un operador que realiza sus operaciones en un solo operando.

Álgebra utiliza un tipo de regla para clasificar los números. Esta regla tiene una posición de cero. L parte izquierda del 0 se refiere a los números negativos, mientras que en el lado derecho de los consideran positivos:

- ∞		-6	-5	-4	-3	-2	-1		1	2	3	4	5	6		+ ∞
0																
- ∞		-6	-5	-4	-3	-2	-1		1	2	3	4	5	6		+ ∞

Un valor en el lado derecho del 0 se considera positivo. Para expresar que un número es positivo, signo + en la parte izquierda. Ejemplos de ello son 4, 228, 90335. En este caso, el símbolo + se ll unario porque actúa en un solo operando.

El operador unario positivo, cuando se utilicen, deberán estar colocados en el lado izquierdo de su opera lado derecho.

Como matemático convención, cuando un valor es positivo, no es necesario expresar con el operador escribiendo el número sin ningún tipo de símbolo significa que el número es positivo. Por lo tanto, los 90335 y puede ser, y son mejores, expresado en 4, 228, 90335. Porque el valor no muestra un signo **sin firmar** como hemos aprendido en la lección anterior.

Expresar como una variable positiva o **sin firma,** puede escribirla. aquí está un ejemplo:

IMPRESIÓN 1250

La negativa del operador --

Como se puede ver en la regla anterior, con el fin de expresar cualquier número en la parte izquierd adjunta con un signo, a saber, la - símbolo. Ejemplos de ello son -12, -448, -32706. Acompañada conoce como negativo.

El signo - debe ser escrito en la parte izquierda del número que se utiliza para negar.

Recuerde que si un número no tiene un signo, se considera positiva. Por lo tanto, cuando un número e tener un signo -. De la misma manera, si desea cambiar un valor de positivo a negativo, se puede añad izquierda.

Aquí hay un ejemplo que utiliza dos variables. Uno tiene un valor positivo, mientras que el otro tiene u

SELECCIONE -1250

Operadores binarios

La adición

El operador se denomina binario si se opera sobre dos operandos.

La adición, también llamada la suma, es una operación utilizada para agregar un elemento a otro. La utilizando el signo +. Para obtener la suma de dos valores, se escribe + entre ellos, como en Valor1 a de la Además se ha realizado, se obtiene un nuevo valor que se puede poner a disposición o pantalla Usted puede realizar la adición de dos números. Aquí está un ejemplo:

IMPRIMIR 125 + 4088

En Transact-SQL, también se puede realizar la adición en el texto. Aquí está un ejemplo:

IMPRIMIR 'Henry' + 'Kono »

También puede añadir más de dos valores, como a + b + c. El fin que se utiliza para añadir dos c importa. Esto significa Valor1 + Valor2 es el mismo que Valor1 + Valor2. De la misma forma a + b + c a + c + b el mismo que a + b + c y la misma que c + b + a.

La Substracción

La operación de resta, a veces llamado la diferencia, se utiliza para sacar o restar un valor de básicamente lo opuesto de la adición. La sustracción se realiza con el signo -. Aquí está un ejemplo:

IMPRIMIR 1240 - 608

A diferencia de la otra parte, la operación de resta no es asociativa. Esto significa que A - B - C no es igual a c - b - a. Esto se ilustra en las siguientes declaraciones:

IMPRIMIR 128 - 42 - 5
IMPRESIÓN 5 - 42 - 128

Esto produciría:

81
-165

Observe que tanto la adición de las operaciones de transmitir el mismo resultado. En la sección de la re siguen el mismo orden, sino una operación diferente, y las dos últimas operaciones hacen diferentes resu

La Multiplicación

La multiplicación permite añadir un valor a sí mismo un cierto número de veces que, establecidos por u
Como un ejemplo, en lugar de añadir un valor a sí mismo de esta manera: a + a + a + a, ya que la
una y otra vez, usted podría simplemente averiguar cuántas veces se añade una a sí misma, entonces s
por un número que, es este caso, es de 4. Esto significaría añadir un 4 veces a sí mismo, y que
resultado.

La multiplicación se realiza con el signo *. Al igual que la adición, la multiplicación es asociativa: a * b
Aquí está un ejemplo:

IMPRIMIR 128 * 42

Esto produciría 5376

La División

La operación de división es similar a un punto de corte en trozos o en fracciones de un conjunto de valo
división se utiliza para obtener la fracción de un número en términos de otro. La división se realiz
inclinada /. Aquí está un ejemplo:

IMPRIMIR 128 / 42

Esto producir 3

Al realizar la división, sea consciente de sus muchas reglas. Nunca dividir por cero (0). Asegúrese d
relación (s) entre el número de empresas implicadas en la operación.

El Modulo

En la mencionada división, 128/42, el resultado es 3. Al multiplicar 42 por 3, como en 42 * 3, se
algunos casos, puede que esté interesado en conocer la cantidad que se dejó afuera, después de l
operación de módulo se utiliza para obtener el resto de una división como un número natural. El resto
operación con el signo de porcentaje (%). Aquí está un ejemplo:

IMPRIMIR 128% 42

Esto produciría 2.

Paréntesis

Como la mayoría de lenguajes de computación, de Transact-SQL utiliza paréntesis para aislar a un gru
que deben ser considerados como pertenecientes a una entidad. Por ejemplo, como vamos a ver en bre
función paréntesis para delimitar la lista de sus argumentos. Paréntesis también se puede utilizar p
operación o una expresión con respecto a otra operación o expresión. Por ejemplo, al estudiar
algebraicas, vimos que la sustracción no es asociativa y puede dar lugar a resultados impredecible
manera, si su operación incluye diversos operadores, como una mezcla de otra (s) y resta (s), puede u
para especificar la forma de proceder con las operaciones, es decir, lo que la operación se debe (deb
primero. Aquí está un ejemplo:

IMPRIMIR (154 - 12) + 8
IMPRIMIR 154 - (12 + 8)

Esto produciría:

150
134

Como puede ver, utilizando los paréntesis, la forma en que el control de toda la operación se lleva
diferencia puede ser aún más acentuada si la operación incluye 3 o más operadores y 4 o más operando
ejemplo de un comando SELECT anidado que utiliza paréntesis:

```
SELECCIONE
      (SELECT 448,25 * 3) +
      (SELECT 82,28 - 36,04);
IR
```

Thi

Manipulaciones poco

Introducción

Cuando se utiliza un valor en su base de datos o aplicación, el valor debe ser almacenado en algún lug de la computadora utilizando una cierta cantidad de espacio. Un valor ocupa el espacio que se aseme pequeñas cajas. En nuestro entendimiento humano, no siempre es fácil de averiguar cómo una ca almacena en pequeñas cajas de 7 siete cuando sabemos que B es sólo una letra.

Manipulación de bits o un poco relacionadas con la operación le permite controlar cómo se almacenan es una operación que tendrá que realizar muy a menudo, sobre todo en las primeras etapas de su ba obstante, las operaciones de poco (y sobrecargado operadores) están presentes en todos o la mayorí programación, por lo mucho que debería estar al tanto de lo que hacen o lo que ofrecen.

Operadores de bits: El operador no bitwise ~

Una de las operaciones que puede realizar consiste en un poco de invertir su valor. Es decir, si algo tie es posible que desee cambiar a 0 y viceversa. Esta operación puede ser atendido por el operador NO representa con el símbolo de tilde ~

El bitwise **NO** es un operador unario que debe ser colocado en el lado izquierdo de su operando como e

~ Valor

Aquí está un ejemplo:

IMPRIMIR ~ 158

Para realizar esta operación, el intérprete de Transact-SQL considera que cada bit es parte de los invierte el valor de cada bit de 1 a 0 o de 0 a 1 en función del valor de la explotación es poco. Esta op reanudar en el cuadro siguiente:

Trozo	~ Bit
1	0
0	1

Considere la posibilidad de un número con un valor de byte como el 248. En nuestro estudio **numéricos,** definimos cómo convertir números de un sistema a otro. Sobre esta base, el valor binario es 1111 1000 (y su valor hexadecimal es 0xF8). Si se aplica el operador NOT bitwise para invertir en los bits, se obtiene el siguiente resultado:

Valor	1	1	1	1	1	0	0	0
~ Valor	0	0	0	0	0	1	1	1

Comparación bits: El operador AND y bitwise

El bitwise y es un operador binario que usa la sintaxis siguiente

Operand1 y Operand2

Este operador considera dos valores y se comparan los bits de cada uno de ellos con el bit correspon forma de valoración. Si ambos bits son 1 correspondiente, la comparación produce 1. En caso contrario

poco es 0, la comparación produce 0. Esta comparación se reanuda como sigue:

Bit1	Bit2	Bit1 y Bit2
0	0	0
1	0	0
0	1	0
1	1	1

Imagínese que usted tiene dos valores de bytes representados como 187 y 242. Sobre la base de n
los sistemas numéricos, el valor binario de decimales 187 es 1011 1011 (y su valor hexadecimal es
binario decimal de 242 es 1111 0010 (y su valor hexadecimal es 0xF2). Vamos a comparar estos dos va
poco, bitwise utilizando el operador AND:

	Binario								Decimal
N1	1	0	1	1	1	0	1	1	187
N2	1	1	1	1	0	0	1	0	242
N1 y N2	1	0	1	1	0	0	1	0	178

La mayoría de las veces, se desea que el intérprete para realizar esta operación y usar el resultado
Esto significa que usted puede obtener el resultado de esta operación y, posiblemente, mostrar al usua
anterior se puede realizar el siguiente programa:

IMPRIMIR 187 y 242

Esto produciría 178

Comparación bits: El operador OR |

Usted puede realizar otro tipo de comparación de bits utilizando el operador OR de que está represe
sintaxis es:

Valor1 | Valor2

Una vez más, el intérprete se comparan los bits correspondientes de cada operando. Si al menos
equivalente es 1, la comparación produce 1. La comparación sólo produce 0 si ambos bits son 0. E
reanuda de la siguiente manera:

Bit1	Bit2	Bit1 | Bit2
0	0	0
1	0	1
0	1	1
1	1	1

Una vez más, vamos a considerar decimales 187 y 242. Su OR comparación haría el siguiente resultado

	Binario								Decimal
N1	1	0	1	1	1	0	1	1	187
N2	1	1	1	1	0	0	1	0	242
N1 | N2	1	1	1	1	1	0	1	1	251

También puede dejar que el compilador de realizar la operación y producir un resultado. Aquí está un e

IMPRIMIR 187 | 242

Esto produciría 251

Comparación bits: La bitwise XOR-Exclusivo Operador ^

Al igual que los dos operadores, el bitwise exclusivo operador OR realiza un poco la comparación de d
sintaxis es la siguiente:

Valor1 ^ Valor2

El compilador compara los bits de un valor correspondiente a la de los demás poco valor. Si uno de
otro es 1, la comparación produce 1. En los otros dos casos, es decir, si ambos bits tienen el
comparación produce 0. Esta operación se reanuda de la siguiente manera:

Bit1	Bit2	Bit1 ^ Bit2
0	0	0
1	0	1
0	1	1
1	1	0

Vamos a examinar de nuevo decimales 187 y 242. Su exclusiva bitwise XOR comparación haría el sigui

	Binario								Decimal
N1	1	0	1	1	1	0	1	1	187
N2	1	1	1	1	0	0	1	0	242
N1 ^ N2	0	1	0	0	1	0	0	1	73

Si el intérprete lleva a cabo esta operación, que puede producir un resultado como en el ejemplo siguie

IMPRIMIR 187 ^ 242;

Esto produciría 73.

Resumen de la lección

Ejercicio: Utilidad de la empresa

1. Abrir una ventana de consulta de la base de datos UtilityCompany1
2. Crear y ejecutar una operación que puede multiplicar 224 kWh por $ 0.0900/kWh
3. Crear y ejecutar una operación que puede añadir 8,50 dólares al resultado de multiplicar por 148 k

Ejercicio: Estados de EE.UU.

1. Hacer una investigación y averiguar cuáles son los estados que pertenecen a la región de Nueva In
2. En una hoja de papel, escriba el nombre de cada estado que es parte de Nueva Inglaterra, seguido
   población (la población de ese estado)
3. Conectarse al servidor desde la línea de comandos y acceder a la base de datos UnitedStatesRegio
4. Escribir y ejecutar una expresión que calcula el total de la población de Nueva Inglaterra
5. Salga del símbolo del

En la lección anterior, hemos utilizado algunos valores como el 242 o "James Knight '. Estos tipos de valores se hace referencia como constante, ya que sin duda conocen antes de su uso y no cambiamos en nuestras declaraciones. Si va a utilizar una determinada categoría de valor de una y otra vez, usted puede reservar una sección de la memoria de ese valor. Esto le permite poner el valor en un espacio de la memoria de la computadora, cambiar fácilmente el valor de otro, una y otra vez.

Para utilizar el mismo espacio de memoria para almacenar y eliminar los valores según sea necesario, el intérprete de SQL necesita dos principales elementos de información: un nombre y la cantidad de espacio en la memoria capaz de almacenar el valor.

Aprendizaje práctico: Puesta en marcha de la Gestión de Estudio

1. Para iniciar Microsoft SQL Server, haga clic en Inicio -> (Todos los) Programas -> Microsoft SQL Server 2008 -> SQL Server Management Studio
2. En el Conectar al servidor cuadro de diálogo, hacer la correcta selección y proporcionar la información adecuada y, a continuación, haga clic en Conectar
3. Para abrir el editor de código, en el menú principal, haga clic en Archivo -> Nuevo -> Consulta de conexión con los actuales

Declarando variables

Una variable es un área de memoria utilizada para almacenar los valores que pueden ser utilizados Antes de utilizar una variable, debe informar a la intérprete. Esto también se conoce como declaración Para declarar una variable, utilice la palabra clave **DECLARE** utilizando la fórmula siguiente:

DECLARAN *Opciones*

DECLARAN la palabra clave permite al intérprete saber que usted está haciendo una declaración. palabra clave es seguido por un nombre para la variable. En Transact-SQL, el nombre de una variabl signo @. El nombre de una variable le permite identificar el área de la memoria cuando el valor almacena. Mientras que otros lenguajes como C / C + +, Pascal, Java, C #, etc, imponen reglas estricta de Transact-SQL es extremadamente flexible. Un nombre puede ser de sólo dígitos. He aquí un ejemplo

DECLARE @ 264

Ese nombre de dígitos puede crear confusión con un número normal. Un nombre también puede s palabras.

Nombres de objetos

Para evitar confusiones, he aquí las reglas vamos a utilizar en nuestras clases:

- Un nombre se iniciará con un guión bajo, o bien una carta. Ejemplos de ello son **_n @, @ acto,** o
- Después del primer carácter de subrayado o una carta, el nombre tendrá combinaciones de relieve Ejemplos de ello son **_n24 @** o **@ act_52_t**
- Un nombre no incluir caracteres especiales como!, @, #, $,%, ^, &, O *
- Si el nombre es una combinación de palabras, cada palabra comenzará en mayúsculas. Ejemplos **DateHired @, @ _RealSport,** o **@ DriversLicenseNumber**

Para declarar una variable, como veremos en las siguientes secciones, después de haber dado u variable, también debe especificar la cantidad de memoria que la variable se necesitan. La cantidad de se llama un tipo de datos. Por lo tanto, la declaración de una variable se utiliza la siguiente fórmula:

DECLARE @ *VariableName DataType;*

También puede declarar más de una variable. Para ello, sepárelos con una coma. La fórmula sería:

DECLARE @ *Variable1 DataType1, Variable2 DataType2* @, @ *Variable_n DataType_n;*

A diferencia de muchos otros lenguajes como C / C + +, C #, Java o Pascal, si declarar las variabl mismo tipo de datos, el nombre de cada variable debe ser seguida por su propio tipo de datos.

Inicialización de una variable

Después de declarar una variable, el intérprete se reserva un espacio en la memoria de la computa espacio, pero no necesariamente tienen un valor reconocible. Esto significa que, en este momento, la Una forma puede cambiar esto es para dar un valor a la variable. Esto se conoce como inicializar la vari

Recuerde que el nombre de una variable comienza con @ y siempre que lo necesite para hacer referer debe asegurarse de que incluyen el signo @. Para inicializar una variable, en la sección necesaria, el tip palabra clave seguida por el nombre de la variable, seguido por el operador de asignación "=", segu apropiado. La fórmula utilizada es la siguiente:

SELECT @ *VariableName = DesiredValue*

o

SET @ *VariableName = DesiredValue*

Una vez que una variable ha sido inicializada, puede realizar su valor disponible o mostrarlo. Esta vez, nombre de la variable en el lado derecho de **IMPRIMIR o** SELECT.

Tipos de datos

Introducción

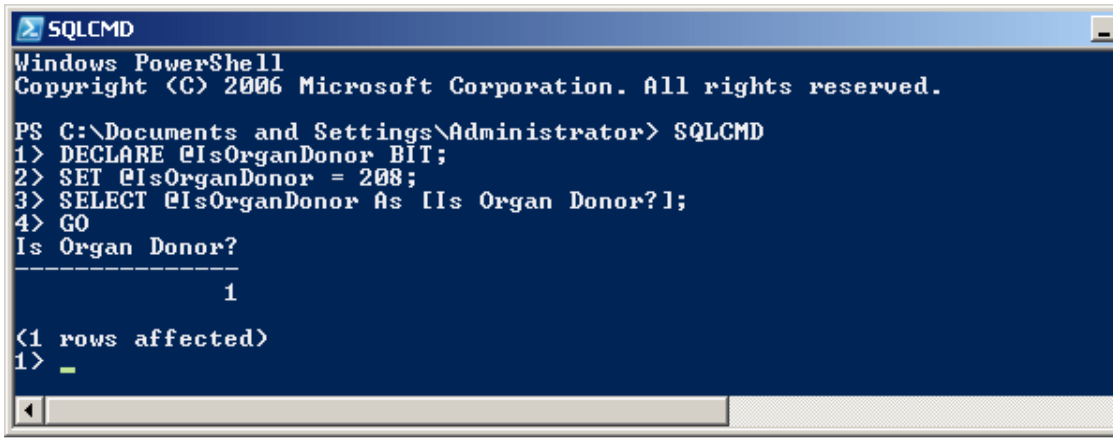
Después de configurar el nombre de una variable, debe especificar la cantidad de memoria que la va guardar su valor. Puesto que hay varios tipos de información de una base de datos puede abordar, SQ conjunto de tipos de datos.

Las variables booleanas

Un valor booleano es una pieza de información que sean verdaderas o falsas, o desactivar, Sí o No, 1 o una variable que tiene un valor booleano, puede utilizar la **BIT poco** o palabra clave. Aquí está un ejem

DECLARE @ IsOrganDonor poco;

Después de declarar una variable booleana, puede inicializar con 0 o en otro valor. Si la variable se ini recibe el valor booleano de falso. Si se inicia con cualquier otro número, que recibe un verdadero va ejemplo de la utilización de una variable booleana:



Aprendizaje práctico: Uso de variables booleanas

1. En la ventana Consulta, escriba lo siguiente:

   DECLARE @ IsMarried poco
   SET @ IsMarried = 1
   SELECT @ IsMarried COMO [está casado?];
   IR

2. Ejecución de la declaración

Integer Variables

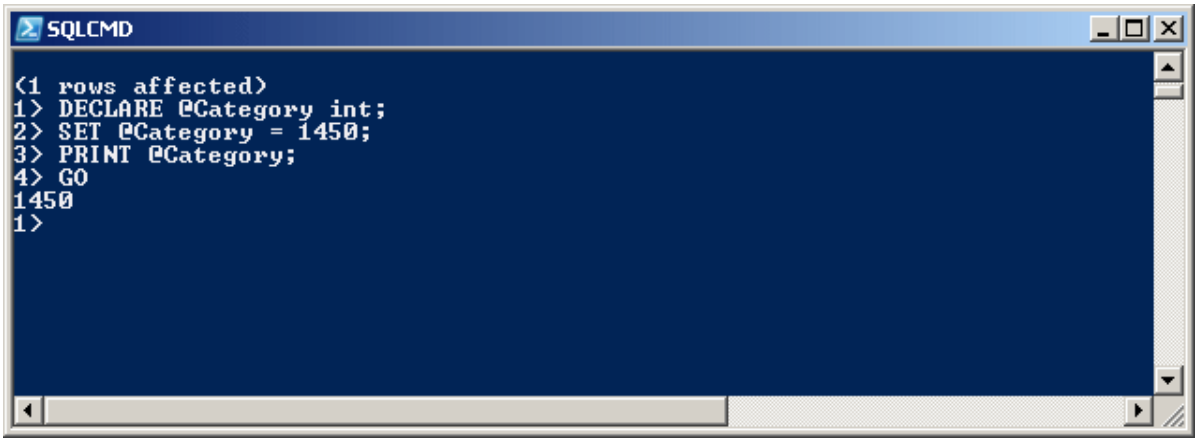
Un número entero, también llamado un número natural, o un número entero, es un número que puede + o un signo - y está hecho de cifras. Entre los dígitos, ningún carácter que no sea un dígito está mundo real, cuando un número es (muy) larga y difícil de leer, como el 79435794, se le permite a u símbolo separador de miles de miles en cada incremento. Un ejemplo es 79.435.794. En las expresion incluyen el separador de miles: usted recibe un mensaje de error.

Cuando el número empieza con +, como el 44 o 8025, este número se conoce como positiva y que deb signo +. Esto significa que el número debe ser escrito como 44 o 8025. Cualquier número que co simplemente un dígito se considera superior a 0 o positivo. Un positivo también se conoce como *sin* lado, un número que comienza con un - símbolo se denomina negativo.

Si una variable celebrará números naturales en el rango de -2147483648 a 2147483647, puede declar clave **int** como tipo de datos. Aquí está un ejemplo:

Categoría int DECLARE @;
SET @ Categoría = 1450;
PRINT @ Categoría;
IR

Esto produciría 1450:



La longitud de un entero es el número de bytes de su campo puede contener. Para un tipo **int,** que ser

Si desea utilizar un número muy reducido de estudiantes, tales como la edad o el número de página periódico, utilizar el tipo de datos **tinyint.** Una variable con el tipo de datos **tinyint** puede almacenar n que van de 0 a 255. Aquí está un ejemplo:

```
SQLCMD
Windows PowerShell
Copyright (C) 2006 Microsoft Corporation. All rights reserved.

PS C:\Documents and Settings\Administrator> SQLCMD
1> DECLARE @StudentAge tinyint;
2> SET @StudentAge = 14;
3> SELECT @StudentAge AS [Student's Age];
4> GO
Student's Age
-------------
           14

(1 rows affected)
1> _
```

El tipo de datos **smallint** sigue las mismas reglas y principios que el tipo de datos **int,** excepto qu almacenar números más pequeños que oscilan entre -32.768 y 32.767. He aquí un ejemplo:

```
1> DECLARE @ NumberOfPages SMALLINT;
2> SET @ NumberOfPages = 16;
3> SELECT @ NumberOfPages COMO [Número de páginas];
4> IR
Número de páginas
---------------
                16

(1 filas afectadas)
```

El tipo de datos **bigint** sigue las mismas reglas y principios que el tipo de datos **int,** excepto que pu gran número de -9.223.372.036.854.775.808 para 9.223.372.036.854.775.807. Aquí está un ejemplo:

```
1> DECLARE @ CountryPopulation bigint;
2> SET @ CountryPopulation = 16500000;
3> SELECT @ CountryPopulation COMO 'País Población';
4> IR
País Población
--------------------
               16500000

(1 filas afectadas)
```

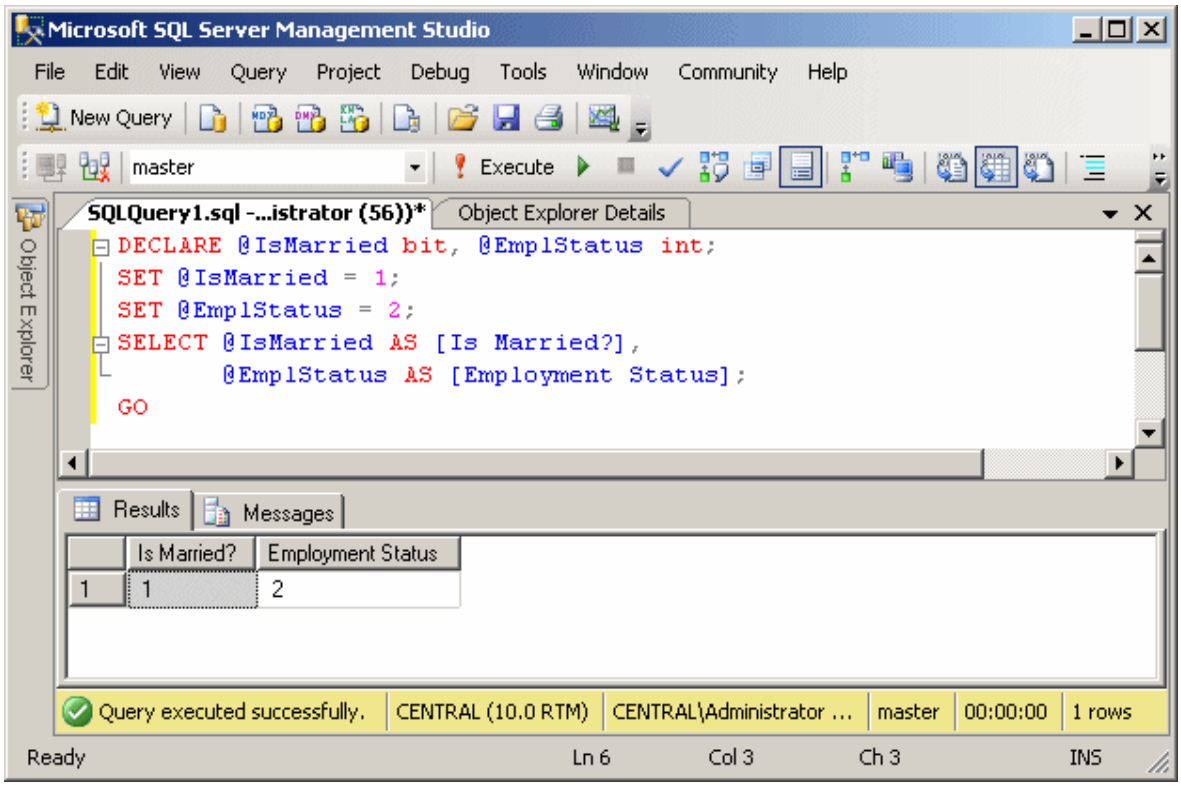
El tipo de datos binarios se utiliza una variable que se celebrará números hexadecimales. Ejem hexadecimales se 0x7238, 0xFA36, o 0xAA48D. Utilice el tipo de datos binarios, si todos los valor tendría la misma longitud (o cantidad). Si cree que algunas entradas sería diferente que otros y, a con el tipo de datos **varbinary** alternativa. El tipo **varbinary** también se utiliza para números hexadecima diferentes entradas, siempre y cuando todas las entradas son hexadecimales.

Aprendizaje práctico: Uso de variables de tipo entero

1. Cambiar la declaración de la siguiente manera:

```
DECLARE @ IsMarried poco, @ EmplStatus int;
SET @ IsMarried = 1;
SET @ EmplStatus = 2;
SELECT @ IsMarried COMO [está casado?],
        @ EmplStatus COMO [Empleo Estado];
IR
```

2. Ejecución de la declaración:



Variables decimal

Un número decimal es un número que puede tener un período (o el carácter utilizado como separador se establece en el Panel de control) entre los dígitos. Un ejemplo sería 12.625 o 44.80. Como un er decimal puede comenzar con un + o un dígito, lo que haría un número positivo. Un número decima empezar con un - símbolo, que lo convierten en un número negativo. Si el número representa una fra de entre los dígitos que especifica qué parte se redujo de 1. Si cree que dicho número para un camp tipo de datos **numéricos** o como **decimales** (ya sea **decimal** o **numérico** produciría el mismo efecto Aquí está un ejemplo:

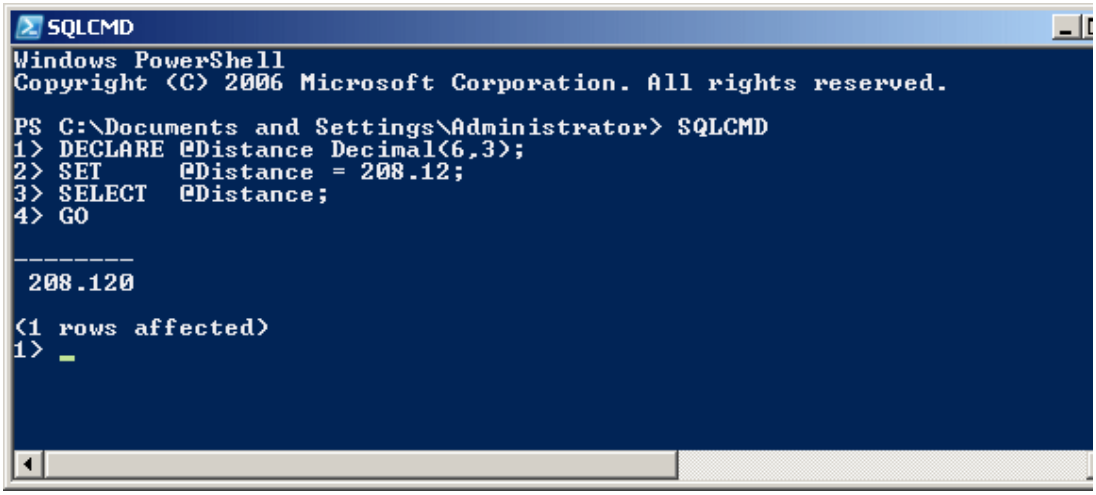
```
1> DECIMAL DECLARE @ Distancia;
2> SET @ Distancia = 648,16;
3> PRINT @ Distancia;
4> IR
648
```

Un número de coma flotante es un número fraccionario, como el tipo decimal. Números de punto f utilizar si usted permitiría que el motor de base de datos para aplicar una aproximación al número re una variable, utilice el **flotador** o la **verdadera** palabra clave. Aquí está un ejemplo:

```
1> DECLARE @ Radio FLOTADOR;
2> SET @ Radio = 48,16;
3> SELECT @ Radio como Radio;
4> IR
Radio
------------------------
        48,159999999999997

(1 filas afectadas)
```

Una precisión es el número de dígitos usado para mostrar un valor numérico. Por ejemplo, el número precisión de 5, mientras que 226 tiene una precisión de 3. Si el tipo de datos se especifica como un en variantes) o un número de coma flotante (float y **real),** la precisión es determinada por la base de aceptar el valor fijado por el intérprete de Microsoft SQL Server. Para un número decimal **(decimal numéricos),** Microsoft SQL Server le permite especificar la cantidad de precisión que desee. El v número entero entre 1 y 38 (28 si está utilizando SQL Server 7).

Un número decimal es un número que tiene una sección fraccionada. Ejemplos de ello son 12,05 escala de un número si el número de dígitos a la derecha del período (o el conjunto de caracteres decimal para los números de su idioma, tal como se especifica en el Panel de control). La escala se ut números que tienen un decimal parte, que incluye la moneda **(dinero** y **smallmoney)** y decimal **decimal).** Si se declara una variable con el **decimal** o el tipo de datos **numéricos,** puede especific escala que desee. El valor debe ser un número entero entre 0 y 18. Aquí está un ejemplo:



Aprendizaje práctico: Uso de variables decimales

1. Cambiar la declaración de la siguiente manera:

```
DECLARE @ IsMarried poco,
        @ EmplStatus int,
        @ WeeklyHours decimal (6,2);
SET @ IsMarried = 1;
SET @ EmplStatus = 2;
SET @ WeeklyHours = 36,50;
SELECT @ IsMarried COMO [está casado?],
       @ EmplStatus COMO [Empleo Estado],
       @ WeeklyHours como las horas;
IR
```

2. Ejecución de la declaración

Variables de moneda

Si una variable se mantenga el valor monetario, se puede declarar que con el **dinero** de palabras cl con un tipo de datos de **dinero** puede tener valores positivos o negativos de -922,337,203 +922,337,203,685,477.5807. Aquí está un ejemplo:

```
1> DECLARE @ YearlyIncome dinero;
2> SET @ YearlyIncome = 48500.15;
3> SELECT @ YearlyIncome COMO [Ingresos anuales];
4> IR
```

```
Ingreso Anual
---------------------
              48500.1500

(1 filas afectadas)
```

Mientras que el tipo de datos de **dinero** puede ser utilizado para una variable que mantenga una moneda, el tipo de datos **smallmoney** se puede aplicar para una variable cuyo valor no puede 214,748.3648 ni superior a 214,748.3647.

La precisión y la escala de un **dinero** o **smallmoney** variables son fijados por Microsoft SQL Server. La 4.

Aprendizaje práctico: Uso de variables de divisas

1. Cambiar la declaración de la siguiente manera:

```
DECLARE @ EmplStatus int,
          @ IsMarried poco,
          @ WeeklyHours decimal (6,2),
          @ HourlySalary SmallMoney,
          @ WeeklySalary SmallMoney;
SET @ IsMarried = 1;
SET @ EmplStatus = 2;
SET @ WeeklyHours = 36,50;
SET @ HourlySalary = 15,72;
SET @ WeeklySalary = WeeklyHours * @ @ HourlySalary;
SELECT @ EmplStatus COMO [Empleo Estado],
         @ IsMarried COMO [Casado?],
         @ WeeklyHours como las horas,
         @ HourlySalary COMO hora,
         @ WeeklySalary COMO semanal;
IR
```

2. Ejecución de la declaración

Microsoft SQL Server Management Studio

File Edit View Query Project Debug Tools Window Community Help

New Query | master | Execute

SQLQuery1.sql -...istrator (56))* | Object Explorer Details

```
DECLARE @EmplStatus int,
        @IsMarried bit,
        @WeeklyHours Decimal(6,2),
        @HourlySalary SmallMoney,
        @WeeklySalary SmallMoney;
SET @IsMarried = 1;
SET @EmplStatus = 2;
SET @WeeklyHours = 36.50;
SET @HourlySalary = 15.72;
SET @WeeklySalary = @WeeklyHours * @HourlySalary;
SELECT @EmplStatus AS [Empl Status],
       @IsMarried AS [Married?],
       @WeeklyHours AS Hours,
       @HourlySalary AS Hourly,
       @WeeklySalary AS Weekly;
GO
```

Results | Messages

	Empl Status	Married?	Hours	Hourly	Weekly
1	2	1	36.50	15.72	573.78

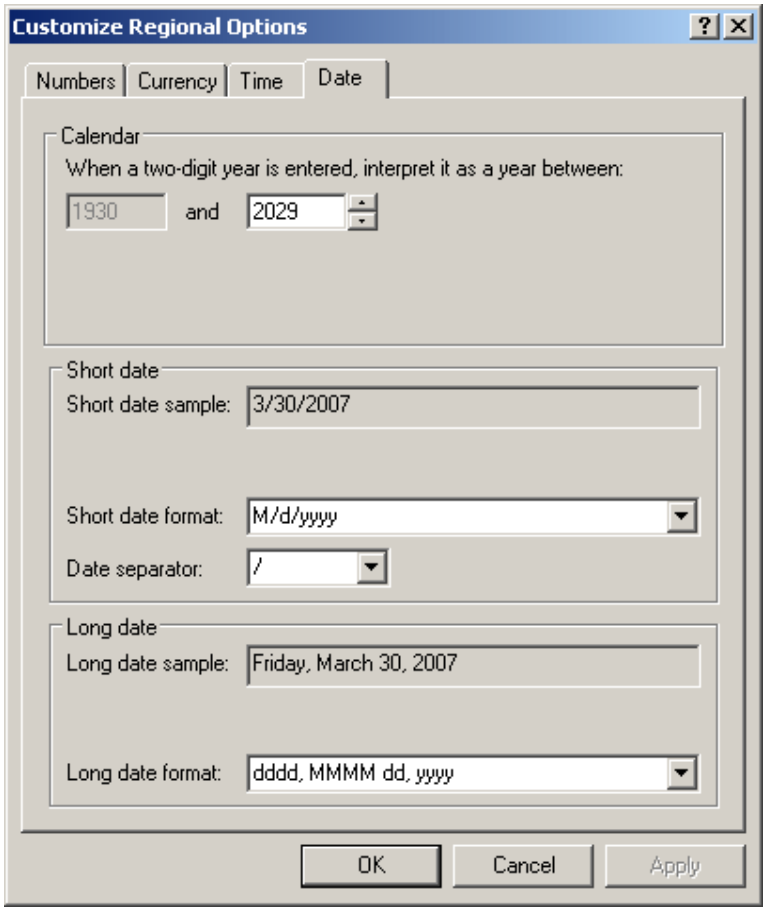
Query executed successfully. | CENTRAL (10.0 RTM) | CENTRAL\Administrator ... | master | 00:00:00 | 1 rows

Ready | Ln 16 | Col 3 | Ch 3 | INS

Fecha y hora Variables

Un tipo de datos **DATETIME** se utiliza para una columna cuyos datos se componen de fecha y / o va
Las entradas deben ser válida la fecha o la hora, pero los valores de Microsoft SQL Server permite una
incluso para mostrar una fecha en un formato no tradicional. La fecha valor de un campo **date**
comprendido entre el 1 de enero de 1753 y el 31 de diciembre de 9999.

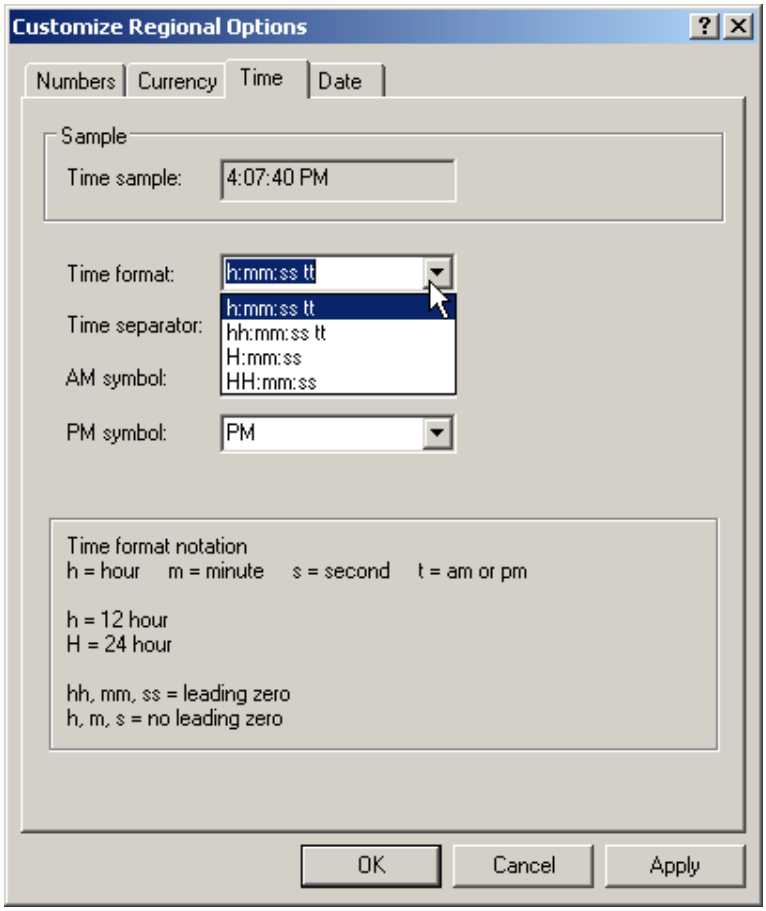
Para inicializar una variable **de datos DATETIME,** incluya su valor entre un solo presupuesto. Si el va
separar los componentes del valor con el símbolo reconocido en el Panel de control como la fecha de sep



Aquí está un ejemplo:

```
1> DECLARE @ IndependenceDay DATETIME;
2> SET @ IndependenceDay ='01 / 01/1960 ';
3> SELECT @ IndependenceDay COMO [Día de la Independencia];
4> IR
Día de la Independencia
-----------------------
1960-01-01 00:00:00.000

(1 filas afectadas)
```

Si el valor es un período de tiempo, todavía se incluyen en una sola cita. Dentro de las comillas, s formatos especificados en el Panel de control:



Aquí está un ejemplo:

```
1> DECLARE @ datetime ArrivalTime;
2> SET @ ArrivalTime = «18: 22 ';
3> SELECT @ ArrivalTime COMO [Hora de llegada];
4> IR
Hora de llegada
-----------------------
1900-01-01 18:22:00.000

(1 filas afectadas)
```

El tipo de datos **smalldatetime** es una alternativa a **datetime.** Se sigue las mismas reglas y princip datos **datetime** a no ser que un valor debe ser la fecha comprendida entre el 1 de enero de 1900 y 6 de

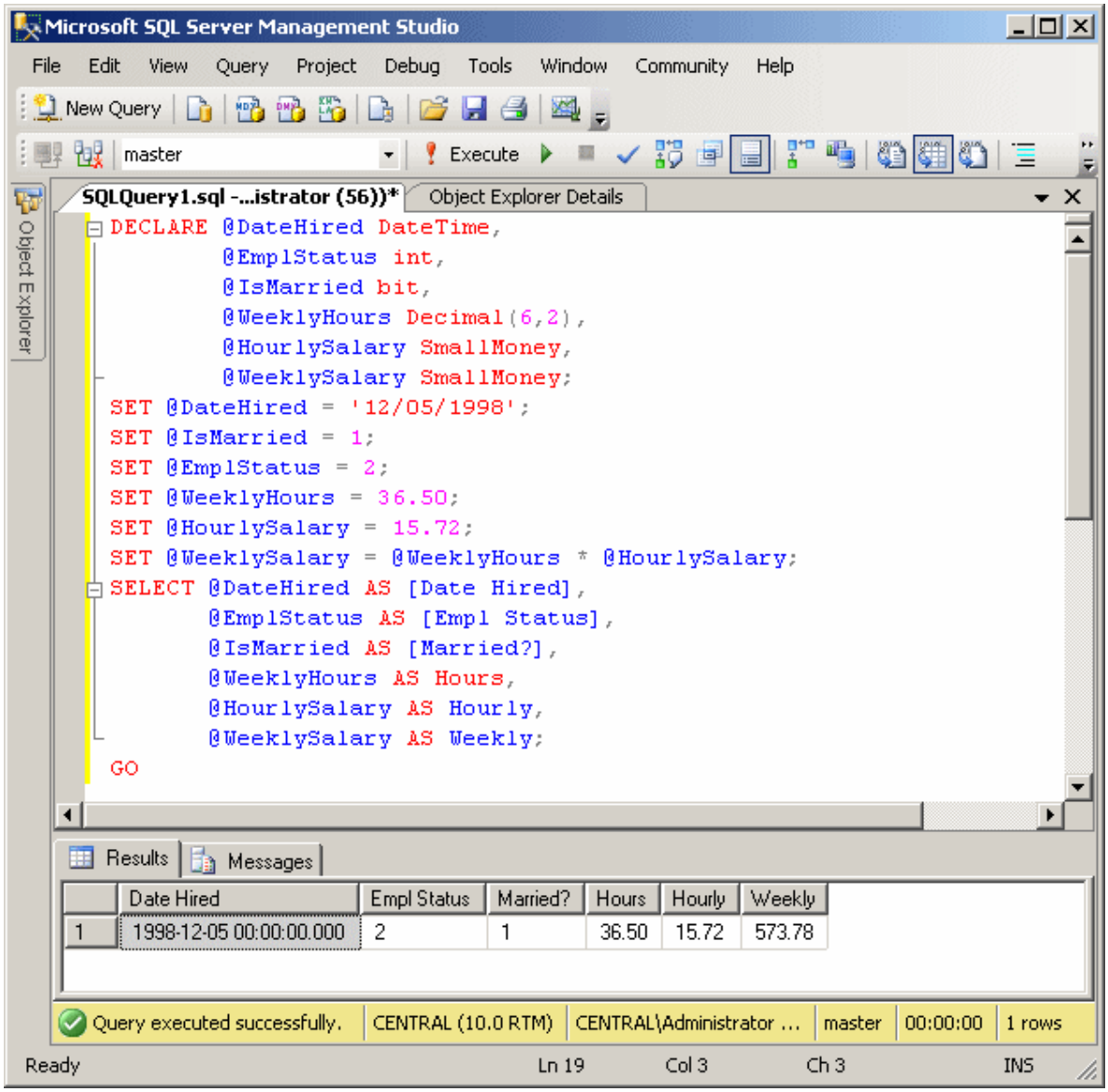
Aprendizaje práctico: Uso de Fecha / Hora Variables

1. Cambiar la declaración de la siguiente manera:

```
DECLARE @ DateHired DATETIME,
        @ EmplStatus int,
        @ IsMarried poco,
        @ WeeklyHours decimal (6,2),
        @ HourlySalary SmallMoney,
        @ WeeklySalary SmallMoney;
SET @ DateHired = «12 / 05/1998 ';
SET @ IsMarried = 1;
```

```
SET @ EmplStatus = 2;
SET @ WeeklyHours = 36,50;
SET @ HourlySalary = 15,72;
SET @ WeeklySalary = WeeklyHours * @ @ HourlySalary;
SELECT @ DateHired COMO [Fecha Ocupado],
        @ EmplStatus COMO [Empleo Estado],
        @ IsMarried COMO [Casado?],
        @ WeeklyHours como las horas,
        @ HourlySalary COMO hora,
        @ WeeklySalary COMO semanal;
IR
```

2. Ejecución de la declaración



Variables de carácter

Un campo de caracteres que puede consistir en cualquier tipo de símbolos alfabético en cualquier comb no. Si desea una variable para almacenar un número fijo de caracteres, como el libro de los números de una biblioteca, declarar que con el tipo de datos **char.** Aquí está un ejemplo:

DECLARE @ caracteres de género;

De forma predeterminada, el tipo de datos char se puede aplicar a una variable que mantenga un c
Después de declarar la variable, cuando la inicialización es, incluir su valor en una sola cita. Aquí está u

```
1> DECLARE @ caracteres de género;
2> SET @ SEXO = 'M';
3> SELECT @ género como la igualdad de género;
4> IR
Género
------
H

(1 filas afectadas)
```

Si incluye más de un carácter único en la cita-, sólo la primera (la mayoría de la izquierda) y el carácte
la variable. Aquí está un ejemplo:

```
1> DECLARE @ caracteres de género;
2> SET @ Género = 'Hombre';
3> SELECT @ género como la igualdad de género;
4> IR
Género
------
M

(1 filas afectadas)
```

Una cadena es un carácter o una combinación de caracteres. Si una variable celebrará cuerdas de difer
se declarará con el tipo de datos **varchar.** La longitud máxima de texto que un campo de tipo **varcha**
es equivalente a 8 kilobytes.

En algunas circunstancias, usted tendrá que cambiar o especificar el número de caracteres utilizados e
cadena. Aunque un nombre y un título del libro debe utilizar la variables de tipo **varchar,** ambas varia
misma longitud de entradas. Si llega el caso, la gente apenas tiene un nombre que está más allá de
muchos títulos de libros van más allá de 32 caracteres. En este caso, ambas variables se utiliza el mis
pero diferentes longitudes.

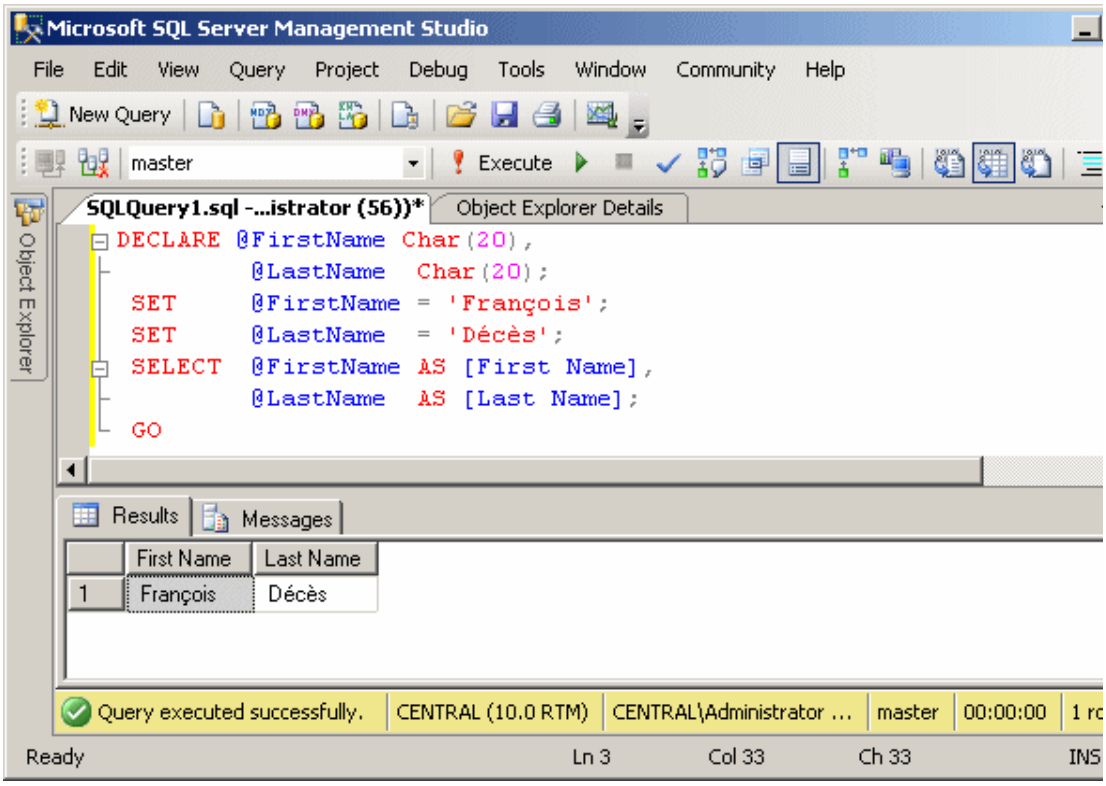
Para especificar el número máximo de caracteres que se pueden almacenar en una variable de ca
derecho de **char** o **varchar,** un tipo de apertura y cierre de un paréntesis. Dentro de los paréntesis, e
deseado. Para inicializar la variable, si está utilizando el símbolo (SQLCMD.EXE), incluya su valor entre
Aquí está un ejemplo:

```
SQLCMD
Windows PowerShell
Copyright (C) 2006 Microsoft Corporation. All rights reserved.

PS C:\Documents and Settings\Administrator> SQLCMD
1> DECLARE @FirstName Char(20),
2>         @LastName  Char(20);
3> SET     @FirstName = "François";
4> SET     @LastName  = "Décès";
5> SELECT  @FirstName AS [First Name],
6>         @LastName  AS [Last Name];
7> GO
First Name           Last Name
-------------------- --------------------
François             Décès

(1 rows affected)
1>
```

Si está utilizando una ventana de consulta, no incluyen el valor de la cadena entre comillas dobles, de lo recibirá un mensaje de error:

```
Microsoft SQL Server Management Studio
File Edit View Query Project Debug Tools Window Community Help
New Query
master  Execute
SQLQuery1.sql -...istrator (56))*  Object Explorer Details
Object Explorer
DECLARE @FirstName Char(20),
        @LastName  Char(20);
SET     @FirstName = "François";
SET     @LastName  = "Décès";
SELECT  @FirstName AS [First Name],
        @LastName  AS [Last Name];
GO

Messages
Msg 207, Level 16, State 1, Line 3
Invalid column name 'François'.
Msg 207, Level 16, State 1, Line 4
Invalid column name 'Décès'.

Query completed with errors.  CENTRAL (10.0 RTM)  CENTRAL\Administrator ...  master  00:00:00  0 rc
Ready  Ln 1  Col 1  Ch 1  INS
```

Por lo tanto, si utiliza la ventana de consulta, incluye la cadena en una sola cita:



El **texto** tipo de datos se puede utilizar en una variable cuyos datos se componen de caracteres ASCII un tipo **varchar** de campo, un campo de tipo texto puede contener texto que tiene más de 8 kilobytes.

El **NCHAR, nvarchar, ntext** tipos y siga las mismas normas que el **CHAR, VARCHAR,** y el **texto,** salvo que puedan ser aplicados a variables que tienen caracteres internacionales, es decir, los car idiomas aparte del Inglés EE.UU.. Esto se hace siguiendo las normas de Unicode formatos.

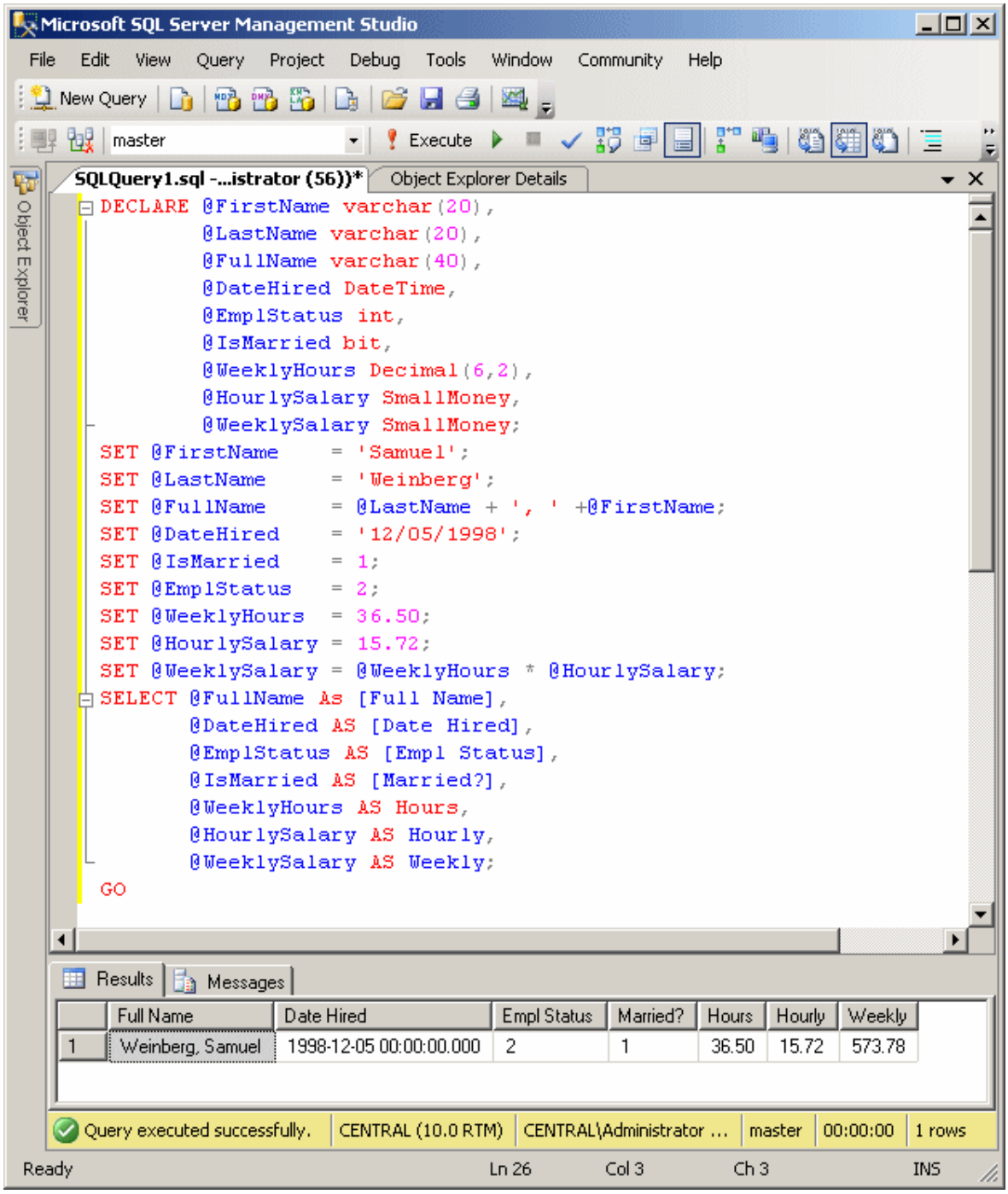
Aprendizaje práctico: Uso de variables de caracteres

1. Cambiar la declaración de la siguiente manera:

```
DECLARE @ Nombre varchar (20),
        @ Apellido varchar (20),
        @ FullName varchar (40),
        @ DateHired DATETIME,
        @ EmplStatus int,
        @ IsMarried poco,
        @ WeeklyHours decimal (6,2),
        @ HourlySalary SmallMoney,
        @ WeeklySalary SmallMoney;
SET @ Nombre = 'Samuel';
SET @ Apellido = 'Weinberg;
SET @ FullName = @ Apellido + ',' + @ Nombre;
SET @ DateHired = «12 / 05/1998 ';
SET @ IsMarried = 1;
SET @ EmplStatus = 2;
SET @ WeeklyHours = 36,50;
SET @ HourlySalary = 15,72;
SET @ WeeklySalary = WeeklyHours * @ @ HourlySalary;
SELECT @ FullName Como [Nombre Completo],
```

@ DateHired COMO [Fecha Ocupado],
@ EmplStatus COMO [Empleo Estado],
@ IsMarried COMO [Casado?],
@ WeeklyHours como las horas,
@ HourlySalary COMO hora,
@ WeeklySalary COMO semanal;
IR

2. Ejecución de la declaración



3. Guarde el archivo como **variables** en la carpeta Mis documentos

Resumen de la lección

Varios Ejercicios

1. Escribir una declaración de que, cuando se administra el salario anual de una persona, puede evalu hora (consideran que el salario anual se basa en 40 horas a la semana)
2. Escribir una declaración de manera que, dada la base y la altura de un triángulo, calcula e indica la un triángulo es b * h / 2)

Ejercicio: Utilidad de la empresa

1. Abrir una ventana de consulta de la base de datos UtilityCompany1
2. Declare una variable llamada moneda BaseCharge y inicializarlo a las 8.50
3. Declarar una variable llamada First700 y inicializarlo en 6,50
4. Crear y ejecutar una declaración de que se multiplique un número como 224 a la First700 variable, First700 debe ser considerada en un porcentaje
5. Crear y ejecutar una declaración que añadir 8,50 al resultado anterior para obtener el total de la fa mes

Ejercicio: Estados de EE.UU.

1. Consigue tu documentos de investigación en EE.UU. y las regiones de Nueva Inglaterra
2. Conectarse al servidor desde la línea de comandos y acceder a la base de datos UnitedStatesRegio
3. Declarar el nombre de una variable Region1 e inicializarlo con cero
4. Escribir una declaración que se suma la población de los diferentes estados de Nueva Inglaterra, y media de la población de Nueva Inglaterra, y asignar este valor a la variable declarada anteriormen
5. Mostrar el resultado en la columna media de Población de Nueva Inglaterra
6. Salga del símbolo del

Para sus bases de datos, puede crear expresiones que representan una combinación de valores, las variables, y los operadores. Para apoyar las expresiones de Transact-SQL proporciona a los operadores que no sean, o además de, los que vimos en la Lección 2.

La comparación es una operación booleana que produce un cierto o un resultado falso, en función de los valores en los que la comparación se realiza. La comparación se realiza entre dos valores del mismo tipo, por ejemplo, puede comparar dos números, dos personajes, o los nombres de dos ciudades. De apoyo a las comparaciones, de Transact-SQL proporciona todo lo necesario operadores.

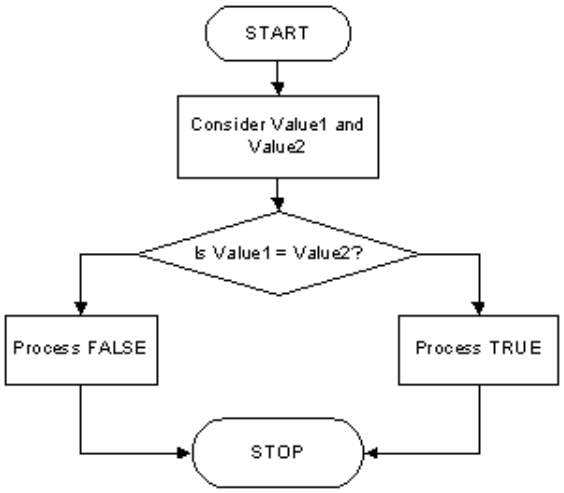
Operador de igualdad =

Para comparar dos valores de igualdad, utiliza el operador =. Su fórmula es:

Valor1 = Valor2

La igualdad de funcionamiento se utiliza para averiguar si dos valores son los mismos. De la fórmula a de SQL comparar *Valor1* y *Valor2.* Si *Valor1* y *Valor2* son los mismos, la comparación produce un resu diferentes, la comparación hace **FALSE.**

La igualdad se puede ilustrar el funcionamiento de la siguiente manera:



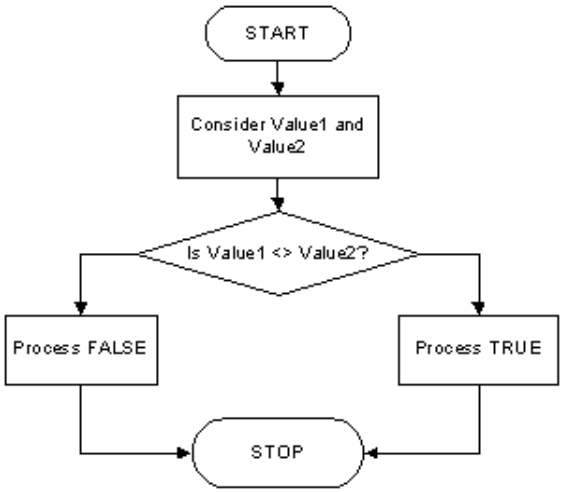
No la igualdad de <>

En contraposición a la igualdad, para averiguar si dos valores no son iguales, utilice el <> operador. Su

Valor1 <> Valor2

La <> es un operador binario (al igual que todos los operadores lógicos) que se utiliza para compar valores pueden provenir de dos variables como en Variable1 <> Variable2. Al comparar los valor diferentes valores, la comparación produce un **TRUE.** En caso contrario, la comparación hace **FALSE** o u

Se puede ilustrar como sigue:



Observe que el operador no la igualdad de <> es lo contrario a la igualdad operador =.

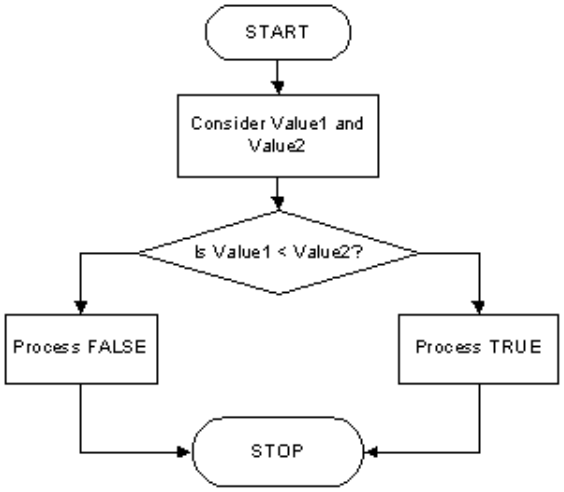
Menor que <

Para averiguar si un valor es inferior a otro, utiliza el operador <. Su fórmula es:

Valor1 <Valor2

El valor en poder de Valor1 se compara con la de Valor2. Como se haría con otras operaciones, la hacerse entre dos variables, como en Variable1 <Variable2. Si el valor en poder de Valor1 es inferio comparación produce un resultado positivo o verdadero.

Menor que el operador "<" puede ilustrarse de la siguiente manera:



Inferior o igual a <=

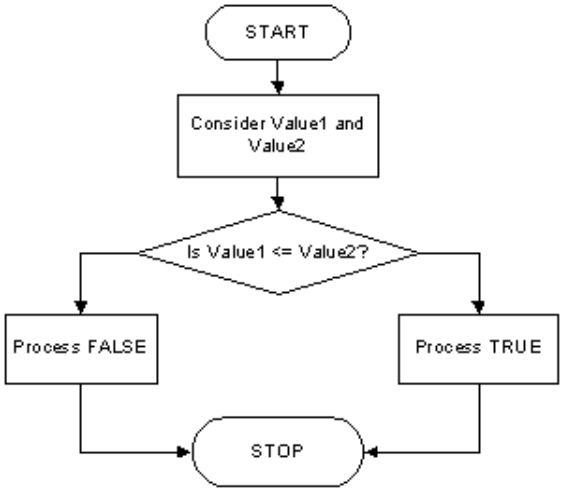
La Igualdad y el Menor que se pueden combinar los operadores para comparar dos valores. Esto le p valores son iguales o si la primera es menor que el segundo. El operador utilizado es <= y su fórmula e

Valor1 <= Valor2

Si ambos *Valor1* y *Valor2* son los mismos, el resultado es verdadero o positivo. Si la izquierda ope *Valor1,* tiene un valor menor que el segundo operando, en este caso *Valor2,* el resultado sigue siendo c

Valor1 es estrictamente mayor que el de valor, la comparación produce **un** resultado falso.

A <= operación puede ilustrarse de la siguiente manera:



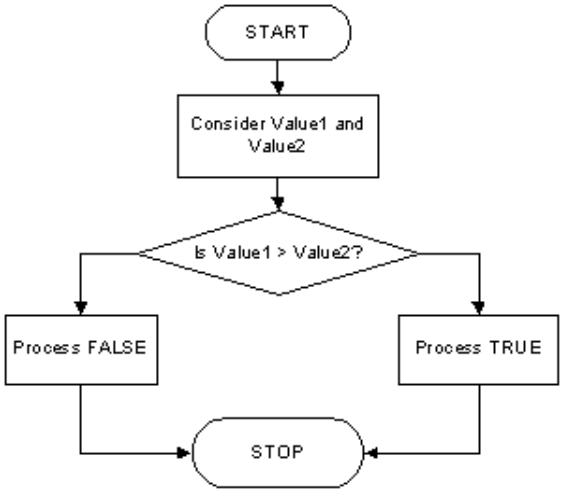
Mayor que>

Para averiguar si un valor es estrictamente mayor que otro, puede usar el operador>. Su fórmula es:

Valor1> Valor2

Ambos operandos, en este caso *Valor1* y *Valor2,* pueden ser variables o la izquierda de operandos pue mientras que el derecho operando es una constante. Si el valor de la izquierda del operador> es mayo lado derecho o una constante, la comparación produce un valor verdadero o positivo. En caso contra hace falsa o nula.

El operador> puede ilustrarse de la siguiente manera:



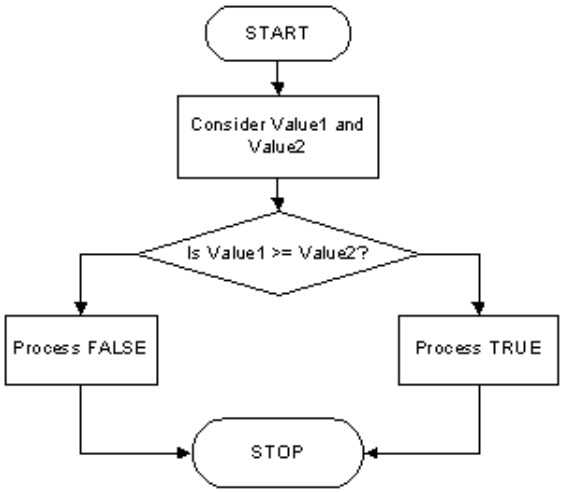
Observe que el operador> es el opuesto al de <=.

Mayor o igual que> =

El mayor de la igualdad y los operadores pueden combinarse para producir un operador de la siguiente es la "mayor o igual a" operador. Su fórmula es:

Valor1> = Valor2

La comparación se realiza en ambos operandos: *Valor1* y *Valor2.* Si el valor de *Valor1* y *Valor2* de qu comparación produce un valor verdadero o positivo. Si el valor de la izquierda operando es mayor qu operandos, la comparación produce también cierto o positivo. Si el valor de la izquierda operando es e que el valor del derecho de operandos, la comparación produce un resultado falso o nulo. Esto pu siguiente manera:



Observe que el> = operador es lo contrario a <.

Las declaraciones condicionales

Introducción

Una condición declaración es una expresión que formular para evaluarlo. La mayoría de las veces, escrita de manera que, cuando se evalúa, puede producir un resultado de verdadero o falso, y luego, resultados, usted puede tomar acción. Una condición es generalmente por escrito lo más simple pos claro a usted y el intérprete de SQL. Aunque la intérprete no se confunde, si crea una difícil declaraci resultado impredecible.

En las próximas secciones, vamos a revisar las palabras clave y las fórmulas de Transact-SQL qu ayudarle a formular expresiones claras. Expresiones por lo general comienzan con una palabra c expresión en sí. Después de la expresión, puede decirle al intérprete qué hacer . La declaración pu sigue:

Expresión de palabra clave
Declaración

BEGIN ... END

Con la fórmula anterior, siempre le permiten saber qué palabras clave se puede utilizar, por qué y cuá expresión, puede escribir la declaración en una sola línea. Esta es la declaración de que sería ejecu *expresión* de nuestra fórmula está satisfecho. En la mayoría de los casos, usted necesitará más de una especificar la *Declaración.*

En este caso, el intérprete considera que lo que venga después de la *Declaración* como una unidad inmediatamente después de la *expresión.* Para indicar que su *declaración* se refiere a más de una palabra clave **COMENZAR.** Entonces usted debe usar la palabra clave **FINAL** para indicar dónde termin este caso, la fórmula de una sentencia condicional que aparece de la siguiente manera:

Expresión de palabra clave
COMENZAR
Declaración de la Línea 1

Declaración de la Línea 2

Declaración de la línea n
FIN

Aún puede utilizar la combinación **BEGIN ...** END, incluso si su *Estado* cubre sólo una línea:

Expresión de palabra clave
COMENZAR
Declaración
FIN

Usando el **BEGIN ...** END combinación hace que tu código sea más fácil de leer, ya que indica claramen
de la *declaración.*

Si una condición es True

Probablemente, la principal que puede realizar la comparación en una declaración es para averigua
operación se realiza mediante una declaración en **caso** de Transact-SQL. Su fórmula básica es:

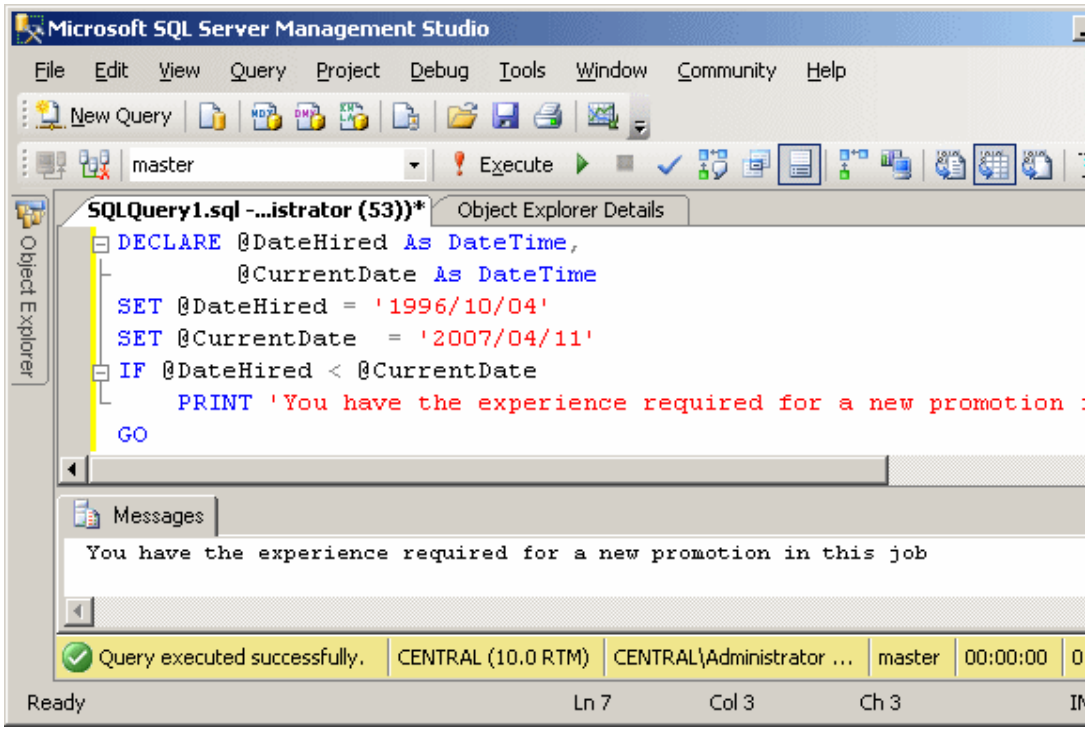
Si la *condición*
Declaración

Cuando la creación de un **SI,** primero asegúrese de que usted proporcione una *condición* que pued
producir verdaderas o falsas. Para crear esta *condición,* puede utilizar las variables y el operador de
examinó anteriormente.

Cuando el intérprete ejecuta esta declaración, primero se examina *la* situación para evaluar a un cie
Estado produce cierto, entonces el intérprete ejecuta el *Estado.* Aquí está un ejemplo:

```
DECLARE @ DateHired Como DATETIME,
                    Como @ CurrentDate de fecha
SET @ DateHired = «1996 / 10/04 '
SET @ CurrentDate ='2007 / 04/11 '
SI @ DateHired <@ CurrentDate
        IMPRIMIR 'Ustedes tienen la experiencia necesaria para una nueva promoción en este trabajo'
IR
```

Esto produciría:

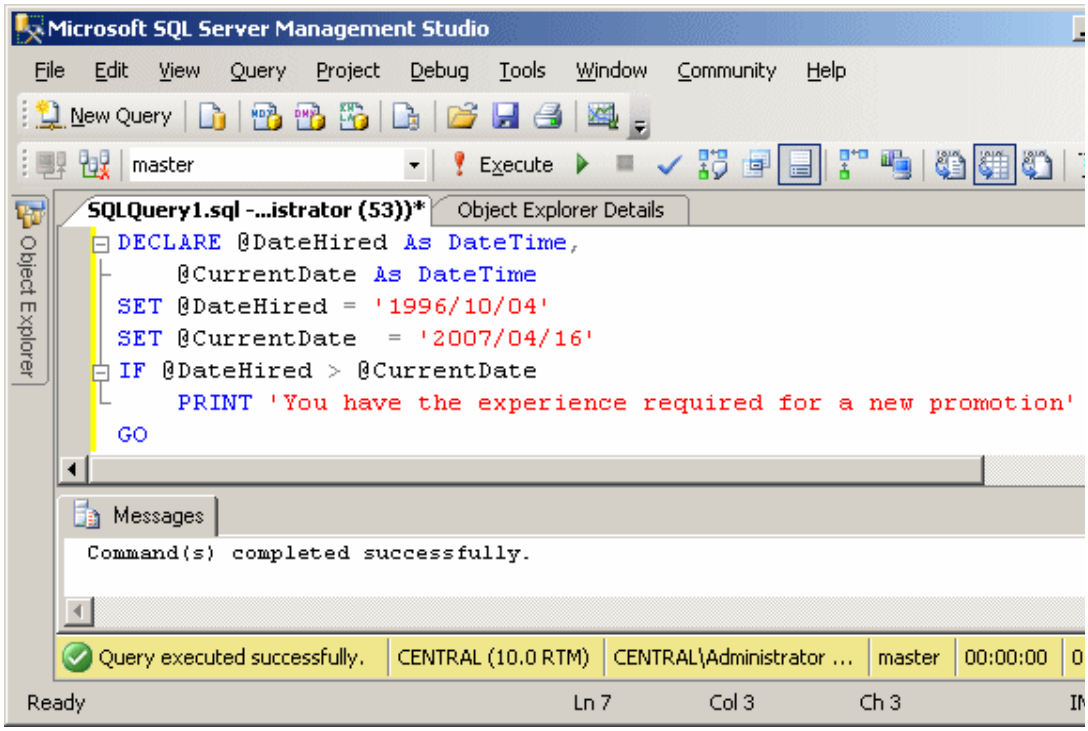


SI ... ELSE

SI la condición de que usamos es apropiado cuando usted sólo necesita saber si una expresión es verda
que hacer en otras alternativas. Considere el siguiente código:

```
DECLARE @ DateHired Como DATETIME,
        Como @ CurrentDate de fecha
SET @ DateHired = «1996 / 10/04 '
SET @ CurrentDate ='2007 / 04/16 '
SI @ DateHired> @ CurrentDate
        IMPRIMIR 'Ustedes tienen la experiencia necesaria para una nueva promoción'
IR
```

Esto produciría:



Observe que, en caso de que la expresión de examinar produce un resultado falso, no hay nada que l puede suceder.

CASO CUANDO ... ENTONCES ...

El **caso** de palabras clave se utiliza como un operador condicional, que considera un valor, lo exam opción en función del valor. La fórmula de la **CASE** es el siguiente:

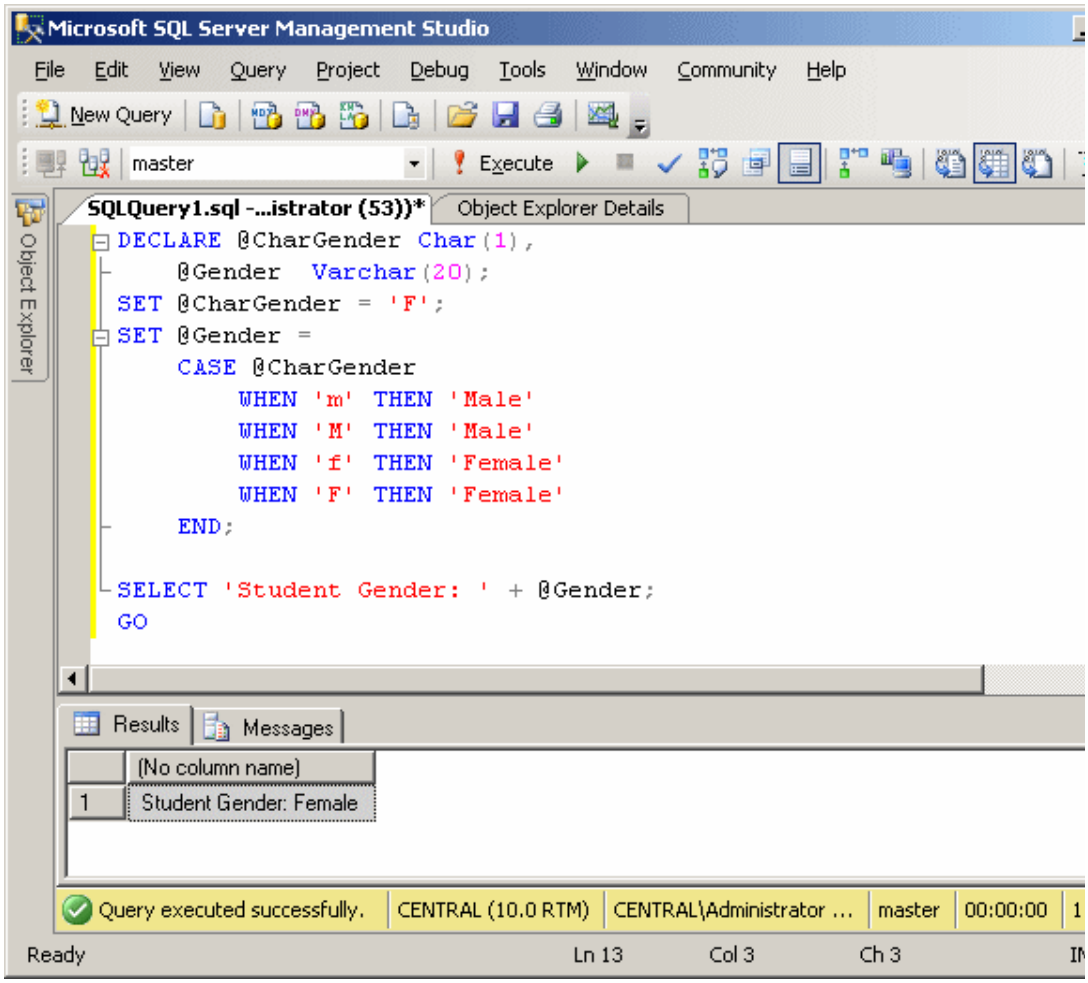
CASO *Expresión*
CUANDO ENTONCES *Valor1 Resultados*
CUANDO ENTONCES *Valor2 Resultados*

CUANDO ENTONCES *Value_n Resultados*
FIN

En el siguiente ejemplo, una carta que representa a un estudiante se presenta. Si la carta es m o M, es como Masculino. Si el valor es siempre como f o F, es una cadena creada como Mujeres:

```
DECLARE @ CharGender CHAR (1),
        @ Género Varchar (20);
SET @ CharGender = 'F';
SET @ Género =
        CASO CharGender @
                CUANDO 'm' y luego 'Hombres'
                CUANDO 'M' y 'Hombres'
                CUANDO 'f' y 'Mujeres'
                CUANDO 'F' y 'Mujeres'
        END;

SELECT 'Estudiante de Género:' + @ Género;
IR
```

Aquí está el resultado de la ejecución es el siguiente:



CASO ... CUANDO ... then ... else

En la mayoría de los casos, usted puede saber el único tipo de valor que se sometería a una **CASE.** En un valor puede ser impredecible presentado. Si usted anticipa un valor distinto de los que son cons proporciona un "apto para todos alternativa mediante el uso de la última declaración como **ELSE.** En e de la declaración se **CASO:**

CASO *Expresión*
CUANDO ENTONCES *Valor1 Resultados*
CUANDO ENTONCES *Valor2 Resultados*
CUANDO ENTONCES *Value_n Resultados*

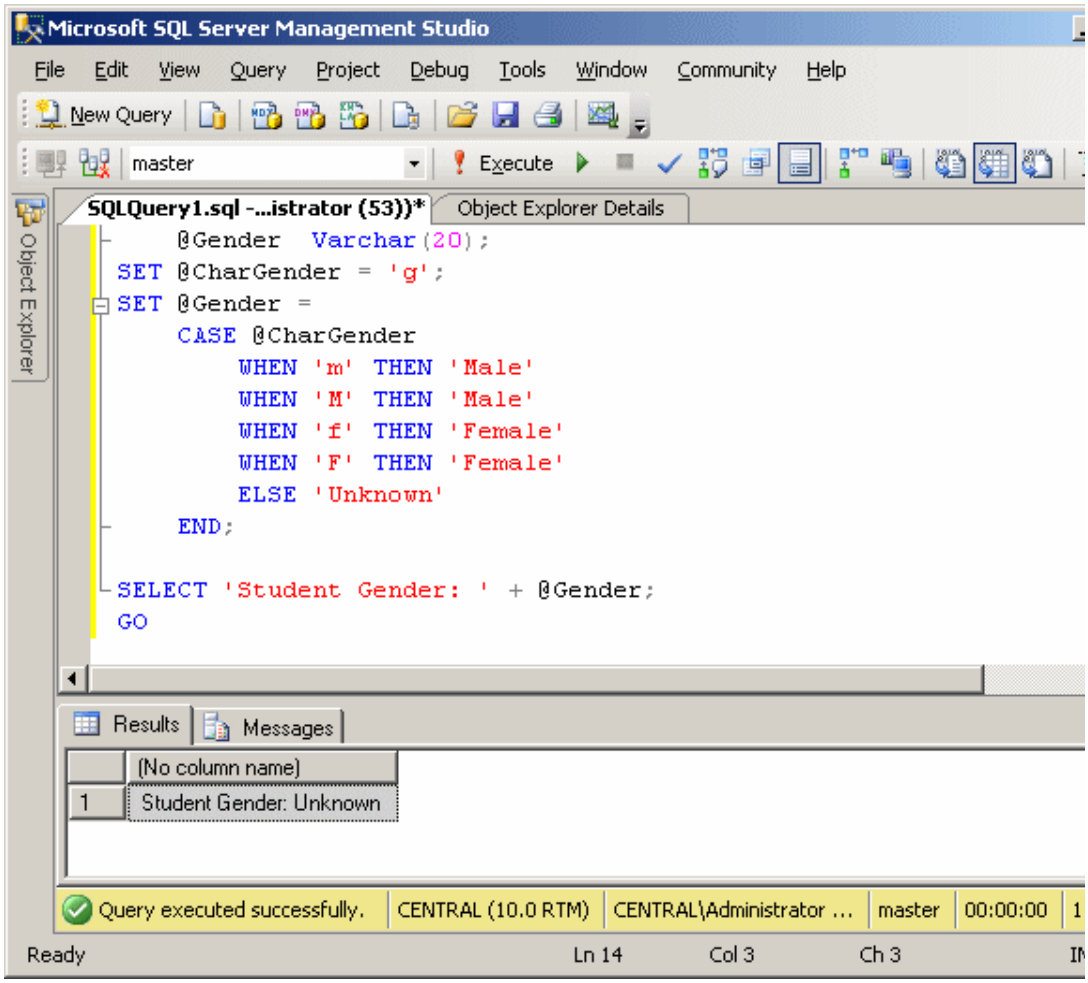
ELSE *Alternativa*
FIN

ELSE la declaración, como la última, se utiliza cuando ninguno de los valores de las declaraciones cuar está un ejemplo:

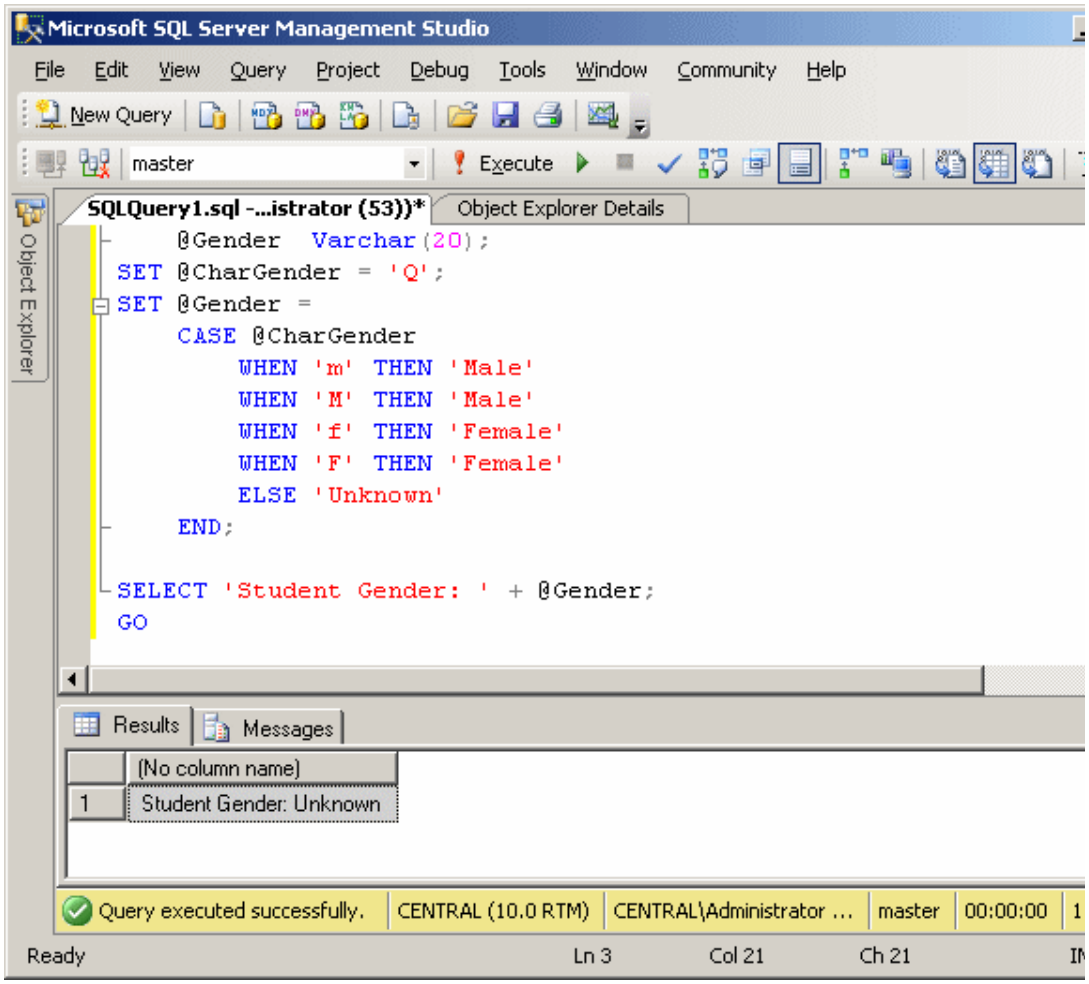
DECLARE @ CharGender CHAR (1),
@ Género Varchar (20);
SET @ CharGender = 'g';
SET @ Género =

```
CASO CharGender @
        CUANDO 'm' y luego 'Hombres'
        CUANDO 'M' y 'Hombres'
        CUANDO 'f' y 'Mujeres'
        CUANDO 'F' y 'Mujeres'
        ELSE 'Desconocido'
END;

SELECT 'Estudiante de Género:' + @ Género;
IR
```

Esto produciría:



Si no producir una declaración **ELSE** un valor, pero no se hace referencia en ninguna de las declar produce, el resultado sería NULL. Aquí está un ejemplo:



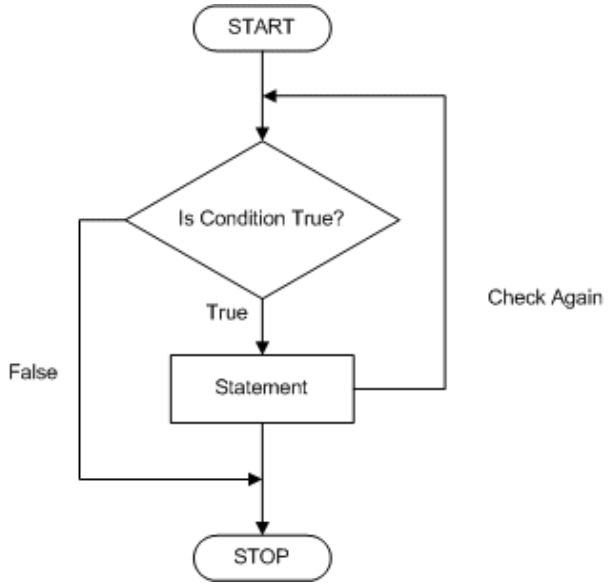
Esto significa que es una valiosa salvaguardia para incluir siempre una declaración **ELSE** sub-declaració

MIENTRAS

Para examinar y evaluar una condición que antes de tomar acción, puede usar el operador **MIENTRAS** de esta declaración es el siguiente:

MIENTRAS *Expresión*
Declaración

Al aplicar la presente declaración, en primer lugar dar una *expresión* a partir de la **MIENTRAS** palabra debe producir un cierto o un resultado falso. Si la *expresión* es cierto, entonces el intérprete ejecuta el ejecutar el *Estado,* la *expresión* se comprueba de nuevo. Mientras la *expresión* es cierto, que se mante la *Declaración.* O cuando una vez que la *expresión* se convierte en falso, se detiene la ejecución de l situación puede ilustrarse de la siguiente manera:

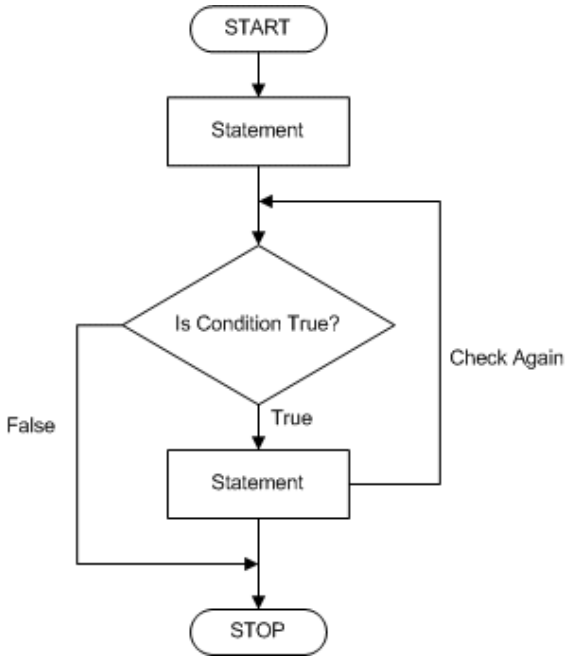


Aquí está un ejemplo:

DECLARE @ Número Como int

MIENTRAS @ Número <5
SELECT @ COMO Número Número
IR

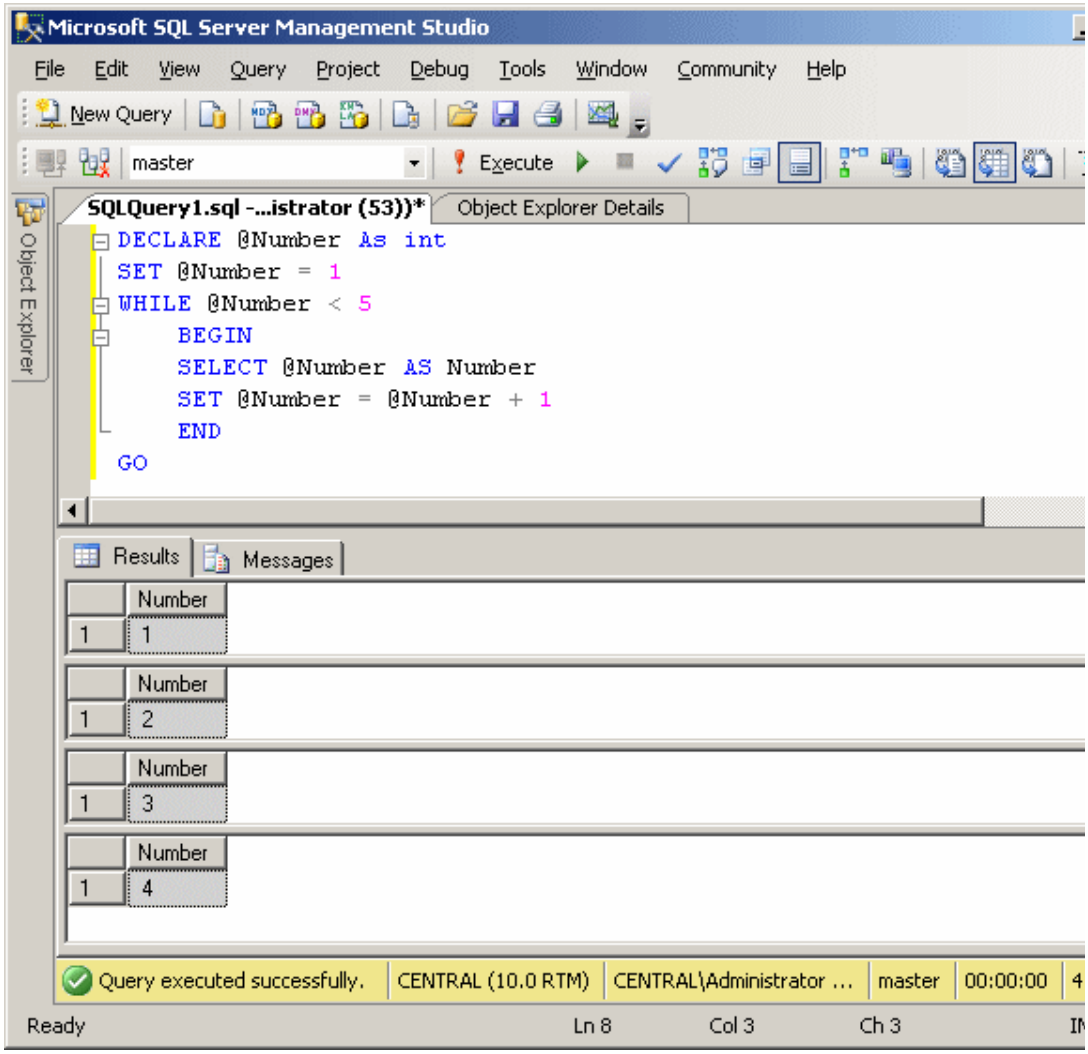
Para ejecutar de manera efectiva, **mientras que** un estado, usted debe asegurarse de que proporci para que el intérprete para obtener un valor de referencia para la enfermedad, variable o expresión que Esto es a veces en forma de una variable está inicializada, aunque podrían ser algunas de otra expresi condición puede ser ilustrado como sigue:



Esta vez, la declaración se aplicaría de la siguiente manera:

```
DECLARE @ Número Como int
SET @ Numero = 1
MIENTRAS @ Número <5
      COMENZAR
            SELECT @ COMO Número Número
            SET @ @ = Número Número + 1
      FIN
IR
```

Esto produciría:



Constantes booleanas

Introducción

Bases de datos y otros entornos de programación proporcionan a los operadores que puede utilizar par datos. Los operadores utilizados son llamados operadores lógicos porque son utilizados para realizar las producen un resultado de verdadero o falso (no hay medio resultado, en otras palabras, algo que no e medio falso o "No sé": o bien es cierto o es falso).

TRUE y FALSE la Constantes

En el álgebra de Boole, lo **cierto** cuando se considera que tiene un valor. El valor también se considera contrario, si algo no tiene un valor, se considera inexistente y no dignos de consideración. Tal cos **FALSE,** 0, o No. Para recuperar este valor, se puede averiguar si el valor de un campo es inexistente o r

La comparación de un valor Verdadero o Falso se realiza principalmente en campos booleanos, tal es e especifica si el estudiante vive en un hogar monoparental) sobre el terreno de los estudiantes de la tab Highschool. Si un registro tiene un valor de 1, la mesa considera que este campo es True. Si el cam entonces tiene un valor **FALSE.**

La constante NULL

Después de haber declarado una variable, el intérprete de SQL se reserva un espacio en la memoria para ello, pero no pone nada en ese espacio de la memoria. En ese momento, esa área de memor significativo. También en ese momento, la variable se considera nulo.

Aquí se nota que tener cuidado con: cuando una variable se dice que tienen un valor nulo, no signific Ni siquiera quiere decir que la variable del espacio de memoria está vacía. En realidad significa que no p claramente el valor actual que la variable es explotación.

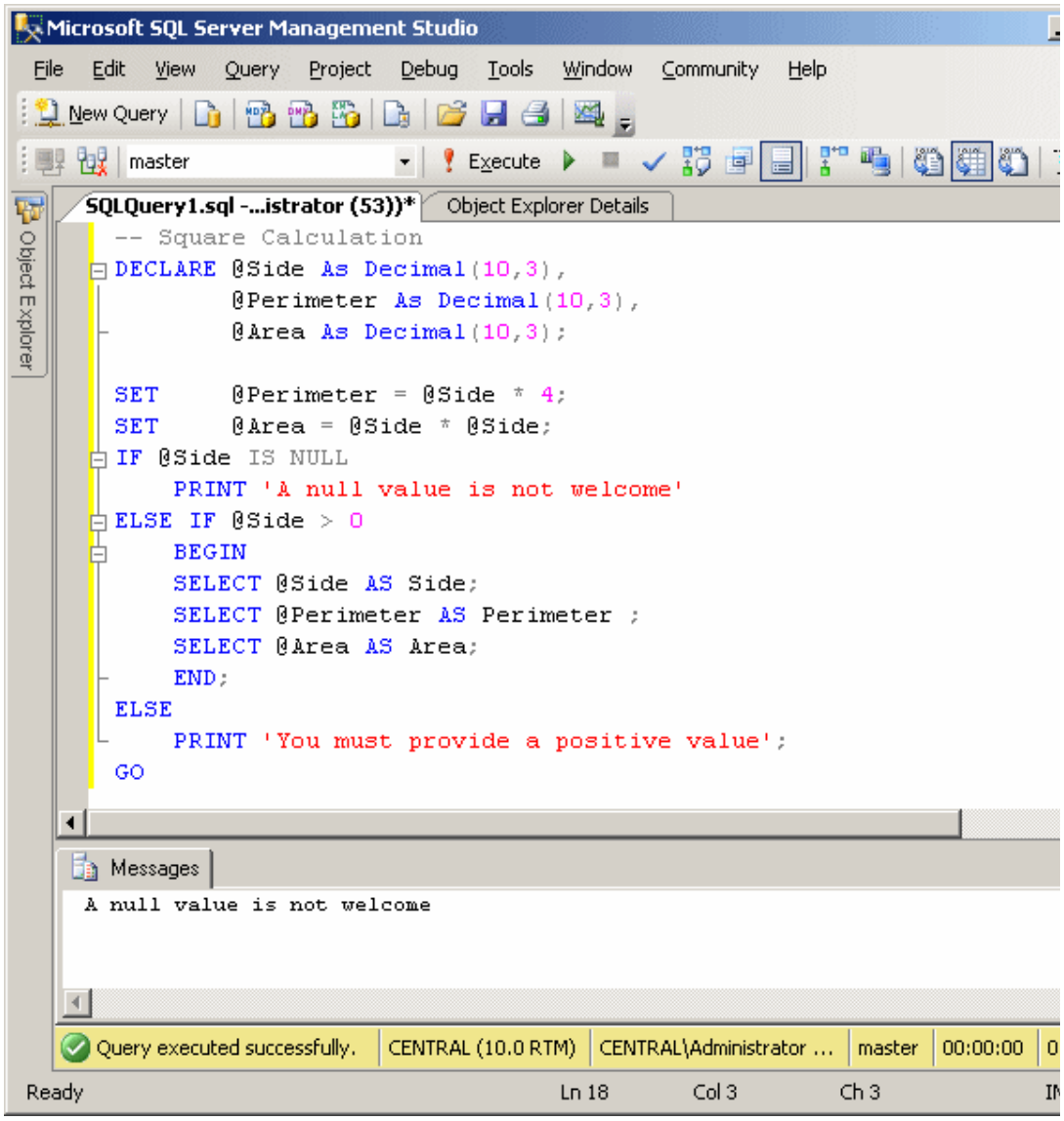
Para apoyar el valor nulo, Transact-SQL proporciona una constante llamada **NULL. NULL** constante se comparación. Por ejemplo, puede utilizar una declaración para comprobar si la nulidad de una variable.

El operador se

Para validar algo como sea posible, puede usar el operador **es.** Por ejemplo, para reconocer que al utilizar la expresión **IS NULL.** He aquí un ejemplo:

```
- Plaza de Cálculo
DECLARE @ lateral como DECIMAL (10,3),
          @ Perímetro decimal (10,3),
          @ Espacio Como decimal (10,3);

SET @ @ Perímetro = lado * 4;
SET @ = Área * @ @ Lado Lado;
SI Lado @ IS NULL
          IMPRIMIR 'Un valor nulo no es bienvenida "
ELSE IF @ lateral> 0
     COMENZAR
          SELECT @ lado como lateral;
          SELECT @ perímetro del perímetro;
          SELECT @ Zona COMO Zona;
     END;
ELSE
          IMPRIMIR 'Usted debe proporcionar un valor positivo »;
IR
```

Esto produciría:

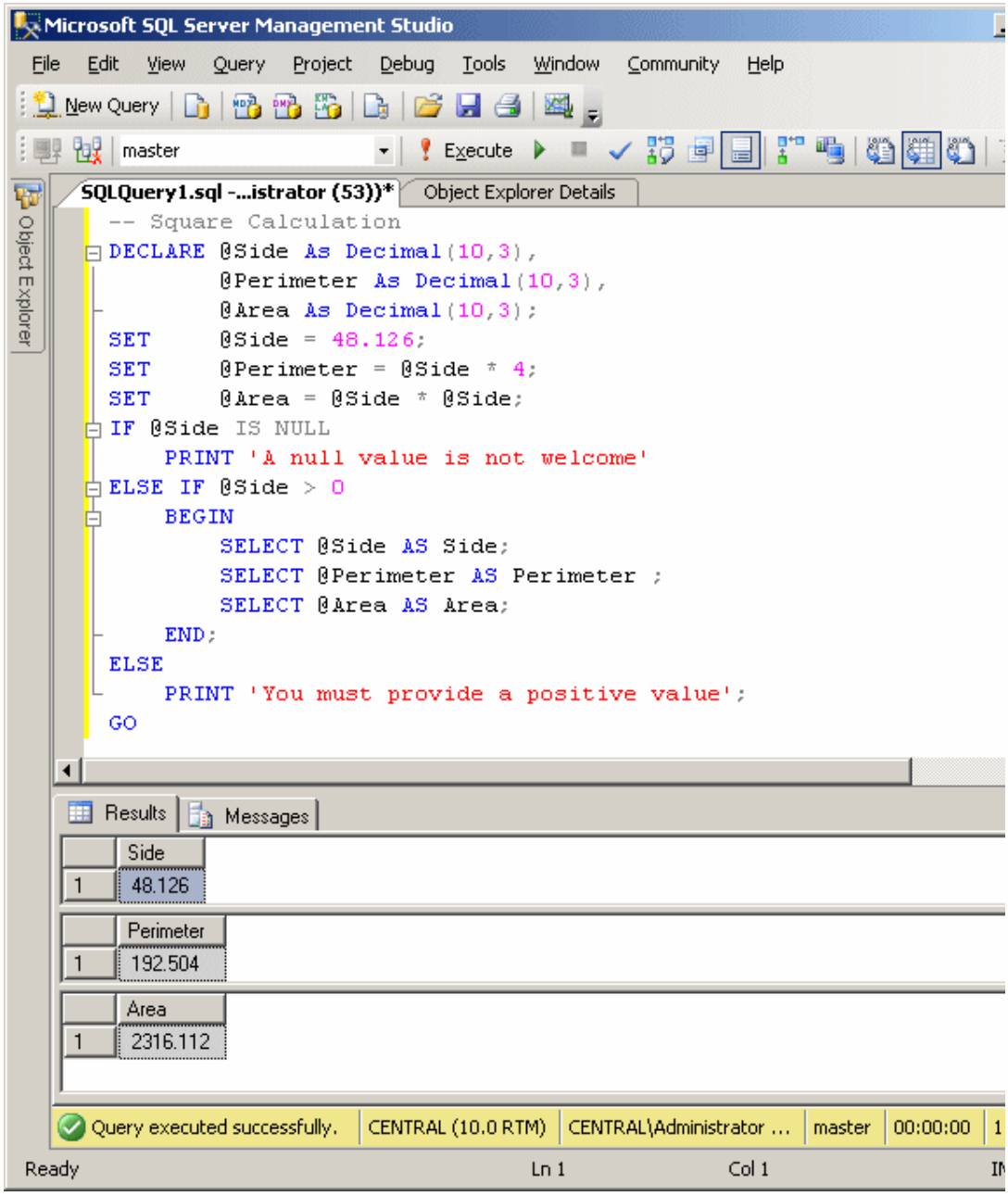


Para evitar tener un valor NULL, puede inicializar la variable o puede asignarle un valor. Aquí está un e

```
- Plaza de Cálculo
DECLARE @ lateral como DECIMAL (10,3),
          @ Perímetro decimal (10,3),
          @ Espacio Como decimal (10,3);
SET @ lateral = 48,126;
SET @ @ Perímetro = lado * 4;
SET @ = Área * @ @ Lado Lado;
SI Lado @ IS NULL
          IMPRIMIR 'Un valor nulo no es bienvenida "
ELSE IF @ lateral> 0
     COMENZAR
                    SELECT @ lado como lateral;
                    SELECT @ perímetro del perímetro;
                    SELECT @ Zona COMO Zona;
     END;
ELSE
```

IMPRIMIR 'Usted debe proporcionar un valor positivo »;
IR

Esto produciría:



El operador NOT

Para negar la presencia, la disponibilidad, o la existencia de un valor, puede utilizar el **operador** NO
utiliza principalmente para invertir un valor booleano. Por ejemplo, hemos aprendido que es lo contrari
la misma manera, **cierto** es lo contrario de **FALSO.** Si desea comparar un valor que no se **TRUE,** el
producen el mismo resultado que el valor **FALSE.** Por la misma razón, la expresión **no sea falsa** es el

Resumen de la lección

Varios Ejercicios

1. Escribir una declaración en la que puede encontrar y mostrar que cada número es divisible por 3, e
2. Escribir una declaración de que, cuando se administra el salario anual de una persona, puede evalu hora (consideran que el salario anual se basa en 40 horas a la semana). Despedir a cualquier salar
3. Escribir una declaración de manera que, si se le da la radio de un círculo, calcula y muestra: el diá circunferencia y el área
4. Escribir una declaración de manera que, habida cuenta de la radio y la altura de un cilindro, calcula diámetro de la base, la circunferencia de la base, la superficie de base, el área lateral, el área total,

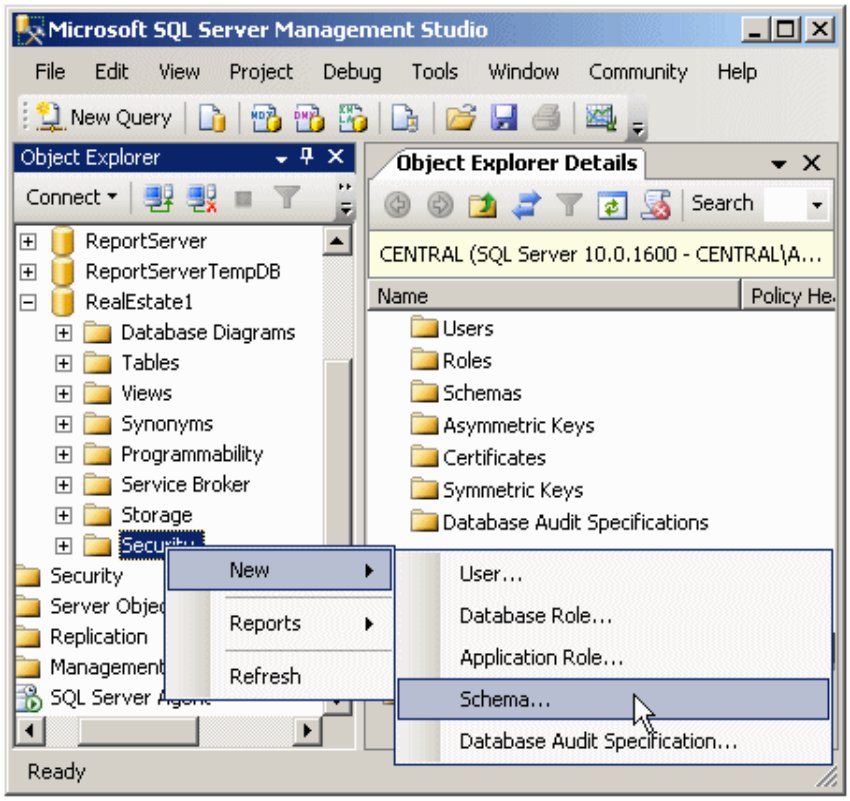
Funciones Fundamentos

Introducción

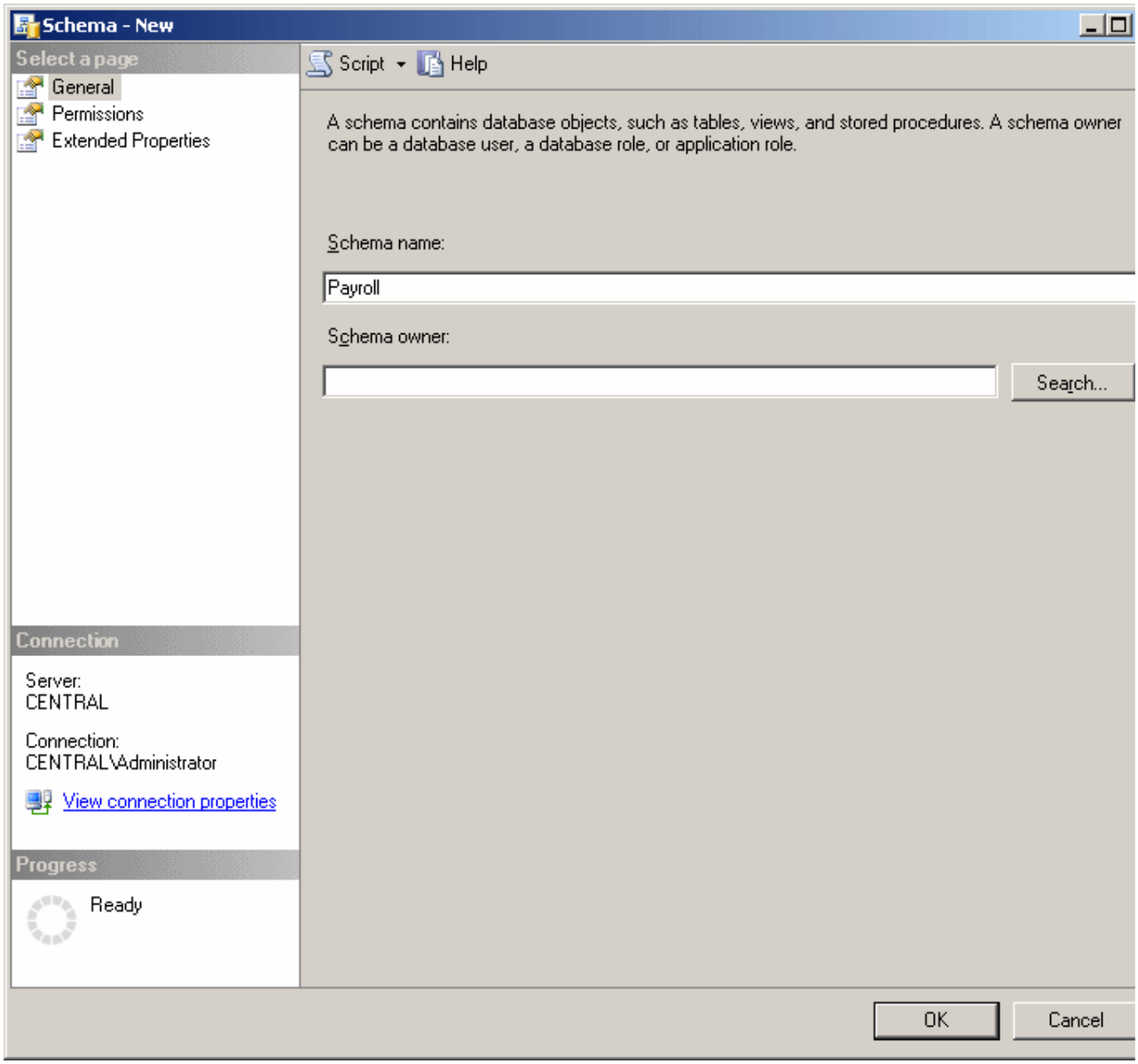
Una función es una parte relativamente pequeña tarea que debe realizarse de lado, pero se puede acceder en cualquier momento para dar un resultado. En Transact-SQL, una función se considera un objeto. Sobre esta base, debe crear y ejecutar una función que antes de usarlo. La función se convierte en parte de una base de datos y se puede acceder a ella.

Aprendizaje práctico: Introducción Funciones

1. Asegúrese de tener una cuenta de usuario llamada pkatts (creado en la Lección 1).
   Asegúrese de que ha creado un nombre de usuario para la cuenta pkatts (creado en la Lección 3).
   Inicie SQL Server Management Studio
2. Para crear una nueva base de datos, haga clic derecho en las bases de datos y haga clic en Nueva base de datos ...
3. En el cuadro de texto Nombre, escriba **RealEstate1**
4. Establecer el propietario para *nombreDeEquipo* \ pkatts (Windows XP o Vista) o *nombreDeDominio* \ pkatts (Windows Server)
5. Establecer rutas de acceso a la C: \ Microsoft SQL Server Base de datos de desarrollo
6. Haga clic en Aceptar
7. En el Explorador de objetos, expanda el nodo Bases de datos seguido por el nodo RealEstate1
8. En virtud de RealEstate1, haga clic derecho en la seguridad, la posición del ratón sobre la nueva, y haga clic en el esquema de ...



9. Escriba el nombre de la nómina y presione Enter



Fundamentos de la creación de la función

Hay varias maneras en que usted puede iniciar la creación de una función:

- En el Explorador de objetos, expanda la base de datos. Expanda el nodo de programación. Expanda el nodo Funciones. Haga clic con el botón de función escalar-Valor y haga clic en Nuevo Valor escalar-Función ... Código de ejemplo se generó para usted. A continuación, puede modificar para personalizar su
- Abra una ventana de consulta vacía. Mostrar las plantillas de la ventana del Explorador y ampliar la función de nodo. Arrastre-Crear Valor Función escalar y suéltelo en la ventana de consulta
- Usted puede abrir una nueva ventana de consulta vacía y comience a escribir su código en él

En Transact-SQL, la primera fórmula de la creación de una función es:

CREATE FUNCTION *FunctionName* ()

El nombre de una función

Hemos mencionado ya que, en SQL, una función se crea como un objeto. Como tal, debe tener un nor experiencia, aquí están las reglas que se utilizan para nombrar nuestras funciones:

- Un nombre se iniciará con un guión bajo, o bien una carta. Ejemplos de ello son _n, actuar, o segu
- Después del primer carácter de subrayado o una carta, el nombre tendrá combinaciones de relieve Ejemplos de ello son _n24, act_52_t
- Un nombre no incluir caracteres especiales como!, @, #, $,%, ^, &, O *
- Vamos a evitar el uso de espacios en un nombre, con pocas excepciones
- Si el nombre es una combinación de palabras, cada palabra comenzará en mayúsculas. Ejemplos DateHired, _RealSport, o DriversLicenseNumber

Volviendo un valor de una función

Para una función de ser útil, debe producir un resultado. Esto también se dice que la función devuelve u valor. Al crear una función, debe especificar el tipo de valor que la función de ida y vuelta. Para p información, después de que el nombre de la función, escriba la palabra clave **VUELVE** seguida por una tipo de datos. Aquí está un ejemplo sencillo:

```
CREATE FUNCTION adjunta ()
VUELVE decimal (6,3)
```

Después de especificar el tipo de valor que la función de retorno, puede crear un organismo para la fu de una función comienza con el **BEGIN** y termina con las palabras clave **FIN.** Aquí está un ejemplo:

```
CREATE FUNCTION adjunta ()
VUELVE decimal (6,3)
COMENZAR

FIN
```

Opcionalmente, puede escribir la palabra clave antes de la **COMO INICIAR** palabra clave:

```
CREATE FUNCTION adjunta ()
VUELVE decimal (6,3)
COMO
COMENZAR

FIN
```

Entre el **inicio** y el **fin** las palabras clave, que es la sección que representa el cuerpo de la función, asignación de la función debe llevar a cabo. Después de realizar esta tarea, justo antes del **final** de pal especificar el valor que la función devuelve. Esto se hace escribiendo la palabra clave **RETORNO** expresión. Un ejemplo de fórmula es la siguiente:

```
CREATE FUNCTION adjunta ()
VUELVE decimal (6,3)
COMO
COMENZAR
    VOLVER Expresión
FIN
```

Aquí hay un ejemplo

```
CREATE FUNCTION GetFullName ()
VUELVE varchar (100)
```

```
COMO
COMENZAR
          RETORNO "Doe, John
FIN
```

Aprendizaje práctico: Creación de Funciones

1. En el Explorador de objetos, haga clic derecho en RealEstate1 y haga clic en Nueva consulta ...
2. Para crear una función, escriba la siguiente declaración:

```
CREATE FUNCTION CalculateWeeklySalary ()
VUELVE decimal (8, 2)
COMO
COMENZAR
      VOLVER 880,44
END;
IR
```

3. Para ejecutar la declaración, en la barra de herramientas del Editor SQL, haga clic en el botón Ejec
4. Para guardar el archivo que contiene el código de la función, en la barra de herramientas Estándar botón Guardar
5. **Calcule** el tipo como el nombre del archivo
6. Haga clic en Guardar
7. En el Explorador de objetos, expanda el nodo RealEstate1, ampliar programabilidad. Expanda Fun ampliar las funciones escalares-Valor. Notificación de la presencia del nodo CalculateWeeklySalary

Función de llamada

Después de una función se ha creado, puede utilizar el valor que devuelve. Uso de una función tambié llamarlo. Para llamar a una función, debe acompañar su nombre. Para ello, escriba el nombre de la ba que se creó, seguido por el operador período, seguido por **dbo,** seguido por el operador período, segui de la función, y sus paréntesis. La fórmula a utilizar es:

DatabaseName. Dbo. *FunctionName* ()

Debido a que una función devuelve un valor, puede utilizar ese valor de lo que precises. Por ejemp cualquiera de **IMPRIMIR o** SELECT para visualizar la función de su valor en una ventana de consul ejemplo que llama la adjunta () Función:

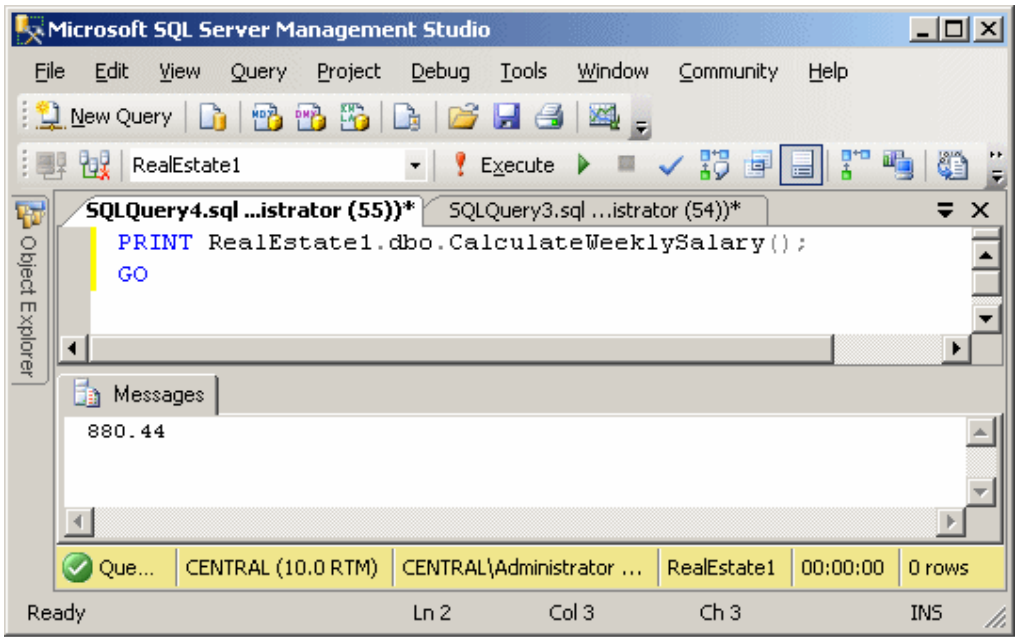
```
IMPRIMIR Exercise.dbo.GetFullName ();
```

Como alternativa, para llamar a una función, en el Explorador de objetos, haga clic derecho en su no del ratón sobre la función de comandos de Como, seleccione y haga clic en Nueva ventana del editor de

Aprendizaje práctico: Realización de una llamada a la función

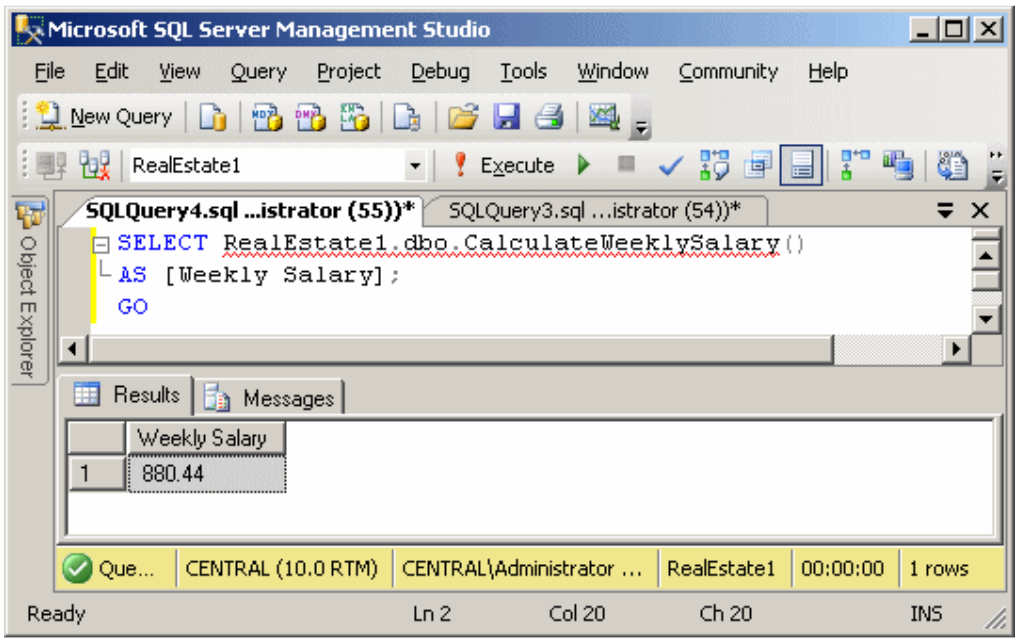
1. En el Explorador de objetos, haga clic derecho en RealEstate1 y haga clic en Nueva consulta
2. Para ejecutar la función que acaba de crear, ejecutar la siguiente declaración:

```
IMPRIMIR RealEstate1.dbo.CalculateWeeklySalary ();
IR
```



3. Para especificar un nombre de columna para el valor devuelto de una función, cambie la función de manera y lo ejecutamos:

SELECCIONE RealEstate1.dbo.CalculateWeeklySalary () COMO [salario semanal];
IR



4. Para guardar la ventana actual, en la barra de herramientas, haga clic en el botón Guardar

Función de mantenimiento

Introducción

Debido a una función de Transact-SQL se trata como un objeto, que pueda necesitar para el mantenimi las acciones que se incluyen cambiar el nombre, modificar o borrar una función.

Cambiar el nombre de una función

Si crea una función y ejecutarlo, sino que se almacena en el valor escalar Funciones nodo con el nomb
Si lo desea, puede cambiar ese nombre, sino mantener la funcionalidad de la función.

Para cambiar el nombre de una función, en el Explorador de objetos, haga clic derecho y haga clic en
Escriba el nuevo nombre que desee y pulse Intro.

Eliminación de una Función

Si crea una función y decidir que no es necesario que más, puede eliminarla.

Para eliminar una función en el Explorador de objetos, busque la función en la sección Funciones, ha
haga clic en Eliminar. El cuadro de diálogo Eliminar objetos que surgen. Si todavía desea suprimir la
en Aceptar, de lo contrario, haga clic en Cancelar.

Para eliminar una función de programación:

- En una ventana de consulta, escriba **DROP FUNCTION** seguida por el nombre de la función y la e
declaración
- En el Explorador de objetos, haga clic derecho en el nombre de la función, coloca el ratón sobre la
comandos de Como, DROP A, y haga clic en Nueva ventana del editor de Consulta
- Abra una nueva ventana de consulta asociada con la base de datos que contiene la función. Mostr
de plantillas y ampliar la función de nodo. Arrastre el nodo DROP FUNCTION y suéltelo en la venta
vacía

Aprendizaje práctico: Eliminación de una Función

1. En el Explorador de objetos, en las funciones escalares-Valor de nodo, haga clic derecho y haga cli
dbo.CalculateWeeklySalary
2. En el cuadro de diálogo Eliminar objetos, haga clic en Aceptar

Modificación de una Función

Como ya se ha mencionado, en el cuerpo de la función, la función de definir lo que se supone que de
mínimo, una función puede devolver un simple número, escrito en la parte derecha de la palabra clav
Aquí está un ejemplo:

```
CREATE FUNCTION adjunta ()
VUELVE int
COMENZAR
      Return 1
FIN
```

También puede declarar nuevas variables en el cuerpo de la función de ayudar en la ejecución de
variable declarada en el cuerpo de una función se conoce como una variable local. Una vez que una
declarada, puede ser utilizado como cualquier otra variable. Aquí está un ejemplo:

```
CREATE FUNCTION adjunta ()
VUELVE int
COMENZAR
      Int DECLARE @ Number1
      SET @ Number1 = 588
      VOLVER Number1 @ + 1450
FIN
```

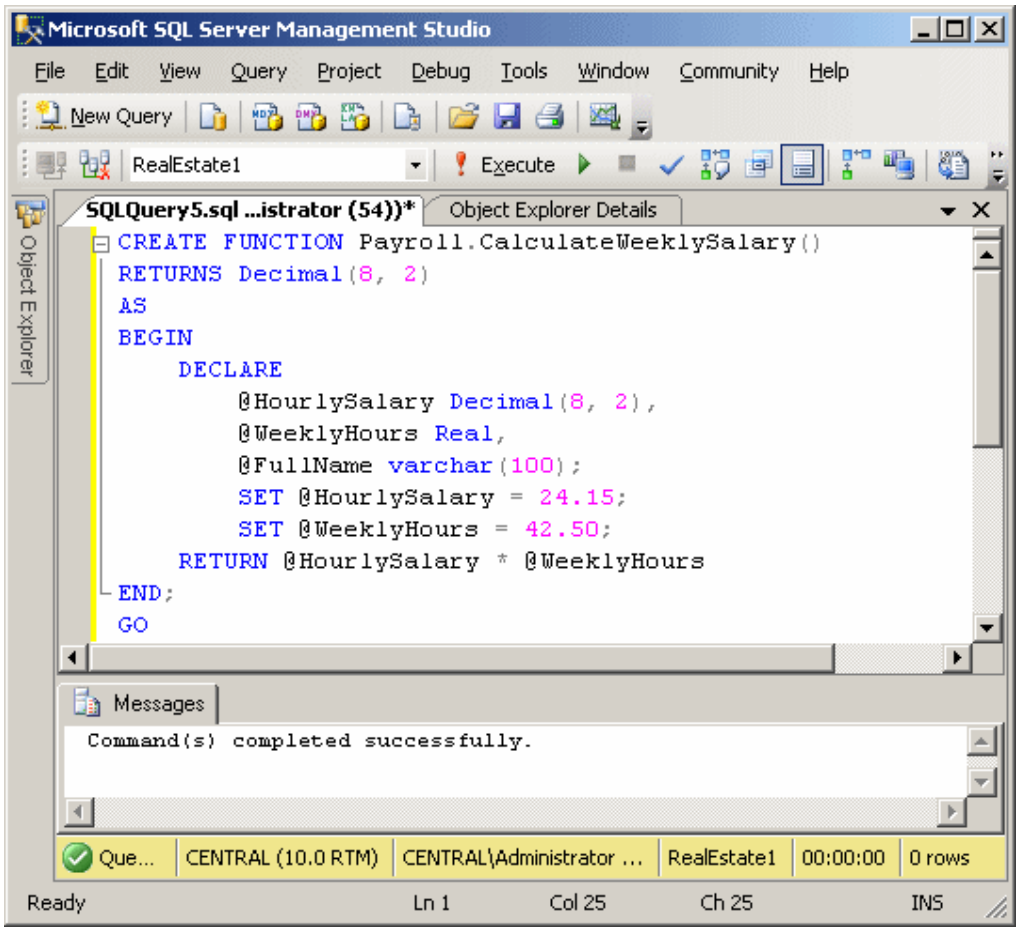
Aprendizaje práctico: Declarar variables locales

1. Calcular en la ventana de consulta, cambie el código de la siguiente manera (nótese la adición del

```
CREATE FUNCTION Payroll.CalculateWeeklySalary ()
```

```
VUELVE decimal (8, 2)
COMO
COMENZAR
        DECLARAN
                @ HourlySalary decimal (8, 2),
                @ WeeklyHours Real,
                @ FullName varchar (100);
                SET @ HourlySalary = 24,15;
                SET @ WeeklyHours = 42,50;
        VOLVER HourlySalary * @ @ WeeklyHours
END;
IR
```

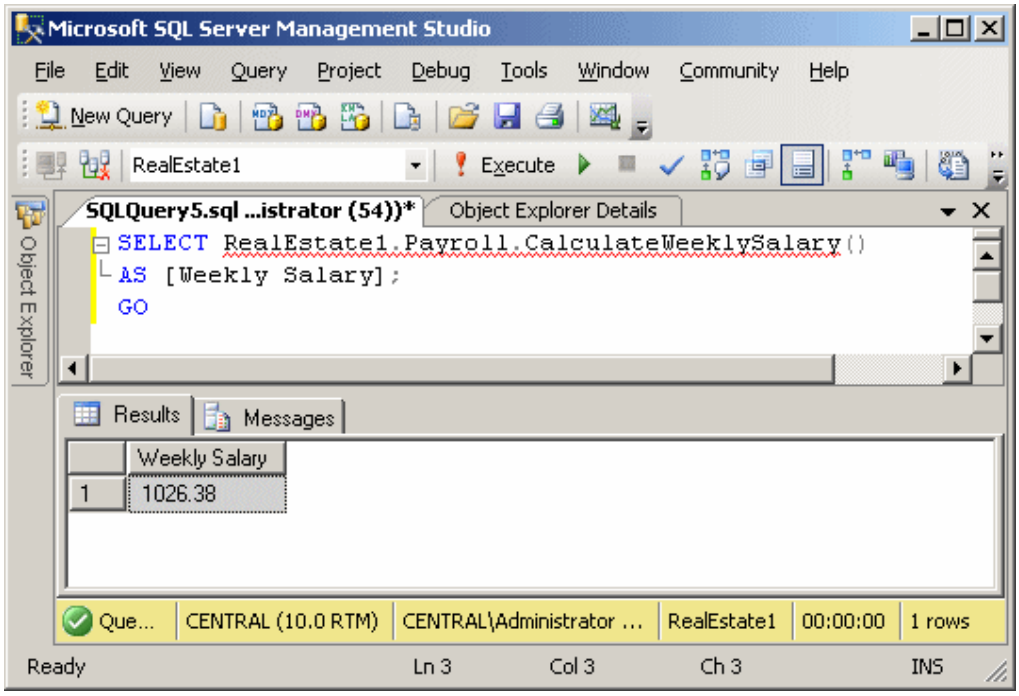
2. Presione la tecla F5 para ejecutar la declaración



3. Para llamar a la función, seleccionar y eliminar el código. Sustituirlo por el texto siguiente:

```
SELECCIONE RealEstate1.Payroll.CalculateWeeklySalary ()
COMO [salario semanal];
IR
```

4. Ejecute el código pulsando la tecla F5



Argumentos de Función

Introducción

Con el fin de cumplir su misión, una función se puede contar con algunos valores. Dicho de otro m función, en lugar de, o además de, las variables locales, es posible que desee el código que llamar proporcionar los valores necesarios para llevar a cabo la cesión. Por ejemplo, imagine que desea crea podría generar empleados direcciones de correo electrónico cuando un usuario ha introducido un nomb el momento que está creando la función, no se puede saber o predecir los nombres de los empleados, i siquiera han sido contratados todavía. En este caso, usted puede escribir toda la función, pero uno o v de posición para los valores que se suministran cuando la función es llamada.

Un valor externo que se presta a una función se denomina parámetro. Una función también puede parámetro. Por lo tanto, al crear una función, que también decidirá si su función de tomar uno o va cuáles son esos parámetros, si las hubiere, se.

Una función parametrizados

Ya hemos visto que una función es también el nombre seguido de paréntesis. Si la función no utiliza su paréntesis se pueden dejar vacíos. Si una función se utiliza un valor externo, cuando se crea especificar un nombre y el tipo de valor de los parámetros. El nombre del parámetro se crea con el si variable como vimos en la lección anterior. Aquí está un ejemplo:

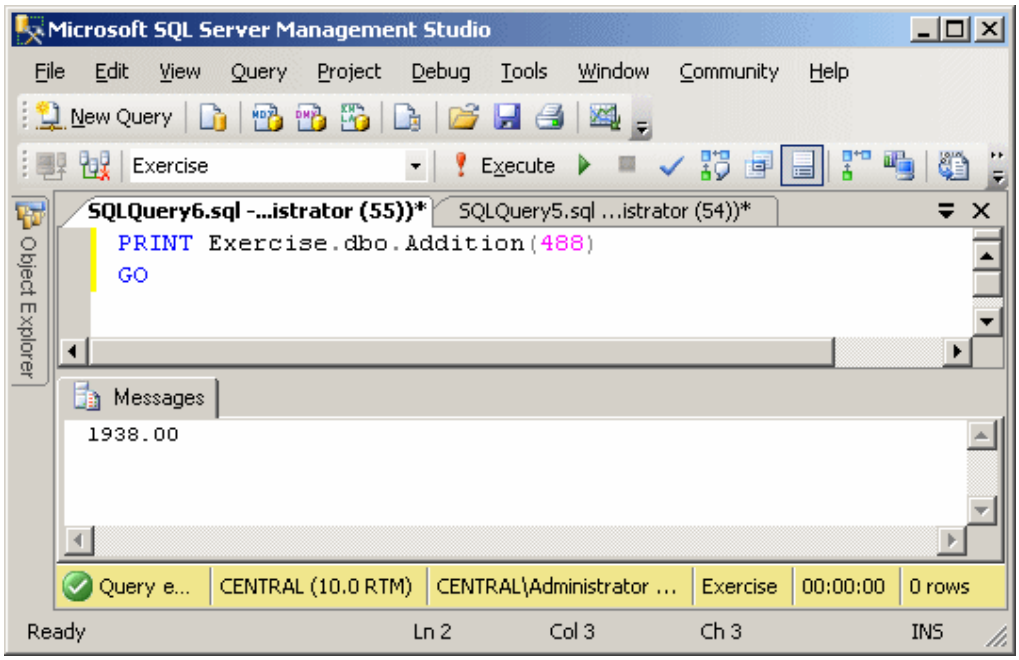
```
CREATE FUNCTION adjunta (@ Number1 decimal (6,2))
```

Cuando una función tiene un parámetro, en el cuerpo de la función, puede utilizar el parámetro co valor, siempre y cuando el respeto que el tipo de valor. Aquí está un ejemplo:

```
CREATE FUNCTION adjunta (@ Number1 decimal (6,2))
VUELVE decimal (6,2)
COMENZAR
    VOLVER Number1 @ + 1450
FIN
```

Realización de una llamada a la función parametrizados

Cuando se llama a una función que toma un parámetro, debe suministrar un valor para este argum escriba el valor del parámetro en el paréntesis de la función. Aquí está un ejemplo:



Una función con diversos argumentos

En lugar de un solo parámetro, también puede crear una función que tiene más de un parámetro. En de los argumentos en el paréntesis de la función con una coma. Aquí está un ejemplo:

```
CREATE FUNCTION adjunta (@ Number1 decimal (6,2), @ Number2 decimal (6,2))
```

Una vez más, en el cuerpo de la función, puede utilizar los parámetros como si ya conocía su valor. declarar variables locales y hacer que participen con los parámetros de lo que precises. Aquí está un eje

```
CREATE FUNCTION adjunta (@ Number1 decimal (6,2),
                         @ Number2 decimal (6,2))
VUELVE decimal (6,2)
COMENZAR
    DECLARE @ resultado decimal (6,2)
    SET @ Resultado = @ @ Number1 + Number2
    RETORNO @ Resultados
END;
IR
```

Cuando se llama a una función que tiene más de un parámetro, en el paréntesis de la función, prop para cada parámetro, en el orden exacto en que aparecen en los paréntesis de la función. Aquí está un

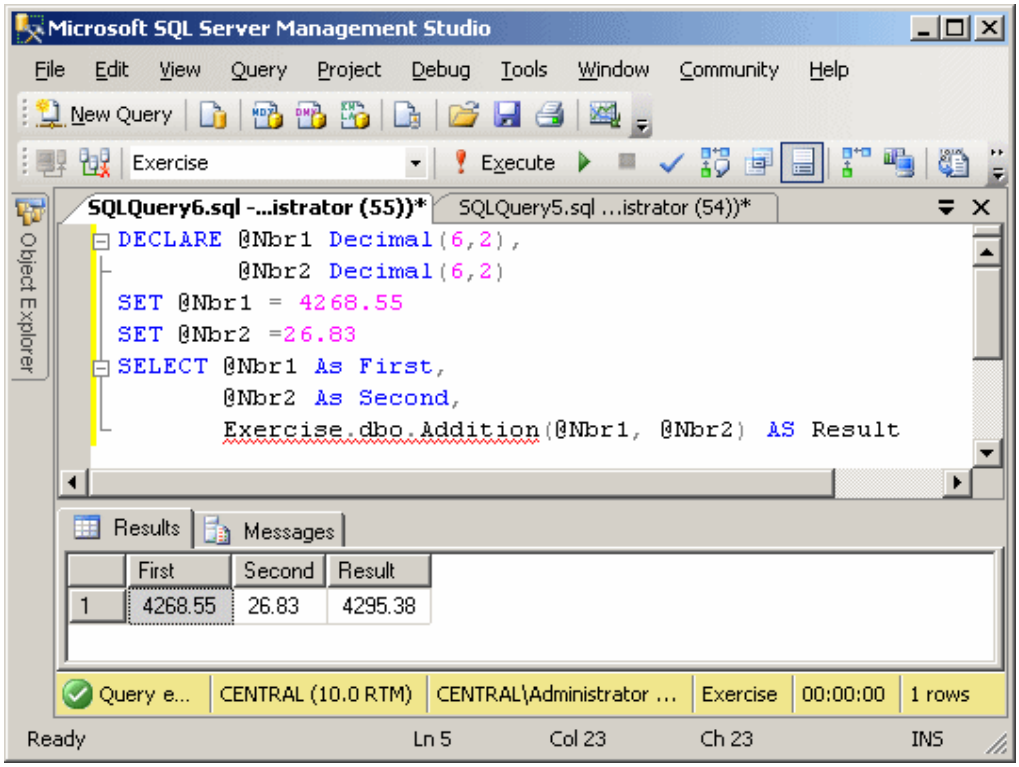
```
IMPRIMIR Variables1.dbo.Addition (1450, 228);
```

Usted también puede pasar los nombres de los ya declarados y las variables inicializadas. Aquí hay llama a la función:

```
DECLARE @ nbr1 decimal (6,2),
        @ Nbr2 decimal (6,2)
SET @ nbr1 = 4.268,55
SET @ Nbr2 = 26,83
```

```
SELECT @ nbr1 En primer lugar,
        @ Nbr2 En segundo lugar,
        Variables1.dbo.Addition (nbr1 @, @ Nbr2) como resultado
```

Esto produciría:



Aprendizaje práctico: Creación de funciones con argumentos

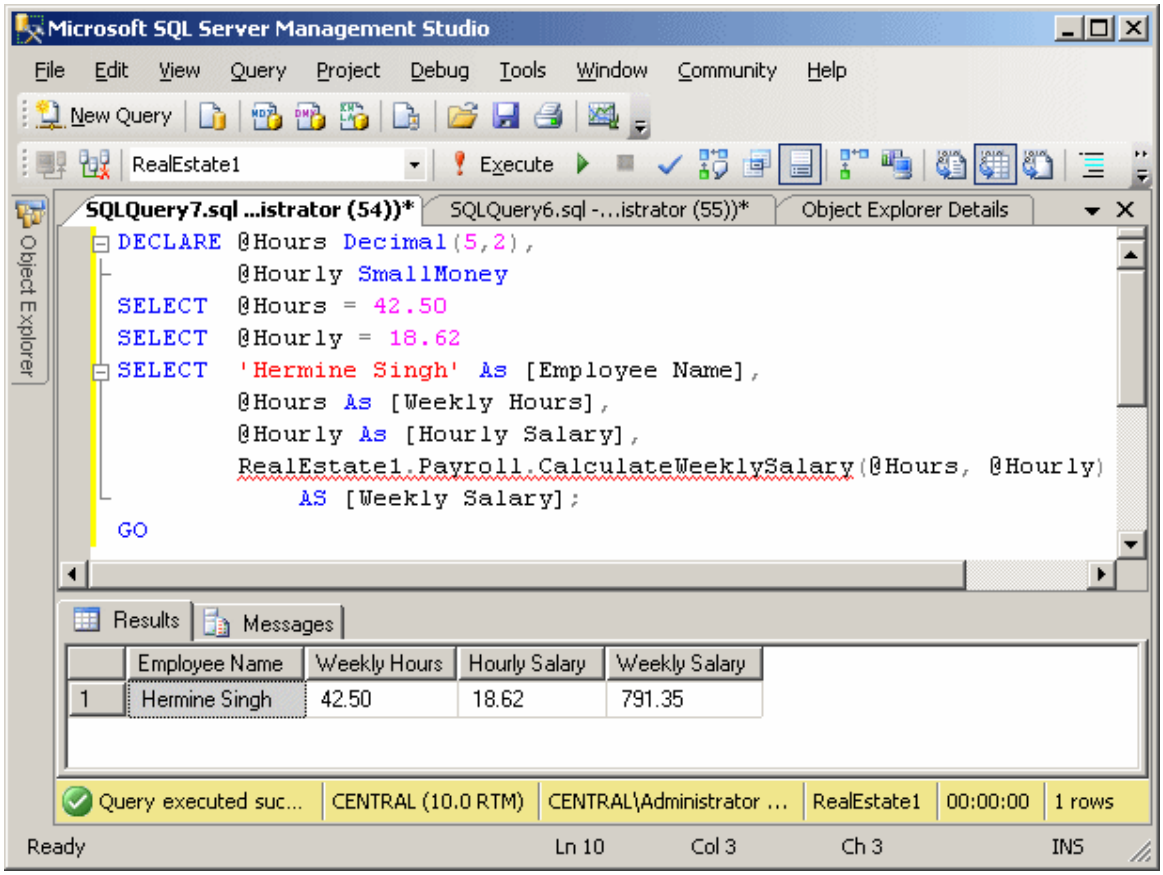
1. En el Explorador de objetos, en las funciones escalares-Valor de nodo, haga clic derecho y haga cli
   Payroll.CalculateWeeklySalary
2. En el cuadro de diálogo Eliminar objetos, haga clic en Aceptar
3. Para añadir argumentos, cambiar el código de la Calcular () la función de la siguiente manera:

```
  CREATE FUNCTION Payroll.CalculateWeeklySalary (@ WeeklyHours
decimal (6,2),
                                                @ HourlySalary
SmallMoney)
  VUELVE decimal (8, 2)
  COMO
  COMENZAR
        DECLARE @ semanal SmallMoney
        SELECT @ @ semanal = WeeklyHours * @ HourlySalary

        RETORNO @ Semanal
  END;
  IR
```

4. Presione la tecla F5 para crear la función
5. Eliminar el código en la ventana y se sustituya por el texto siguiente:

```
DECLARE @ horas decimal (5,2),
        @ Hora SmallMoney
SELECT @ horas = 42,50
SELECT @ hora = 18,62
SELECT 'Hermine Singh' Como [Nombre del empleado],
        @ Horas [horas semanales]
        Como @ hora [Salario por hora],
        RealEstate1.Payroll.CalculateWeeklySalary (@ horas, hora @)
                COMO [salario semanal];
IR
```

6. Presione la tecla F5 para ejecutar la declaración



7. Cierre la ventana de consulta sin guardar el archivo

8. En el Explorador de objetos, de conformidad con el nodo Bases de datos, haga clic derecho y haga RealEstate1

9. En el cuadro de diálogo, haga clic en Aceptar

Resumen de la lección

Varios Ejercicios

1. Escribir una función que puede calcular el salario anual de un puesto de trabajo si un solicitante se por hora. Considere la posibilidad de que el salario anual se basa en 40 horas a la semana. En su salario por hora se da como un valor negativo, no realizar el cálculo

2. Escribir una función que puede calcular el área de un círculo si se le da la radio

3. Escribir un procedimiento (función) que los procesos de un cilindro. Calcula e indica el diámetro de circunferencia de la base, la superficie de base, el área lateral, el área total, y el volumen, habida c

y la altura de un cilindro

Mientras que su principal trabajo como desarrollador de una base de datos consiste en la creación de listas, probablemente su segundo trabajo más importante es ayudar a sus usuarios con las diferentes tareas que deben desempeñar en su aplicación. Una forma puede ayudar es utilizar las funciones que llevan a cabo tareas complejas de otra. Nos presenta y las funciones descritas en la lección anterior. Para ayudar a su desarrollo con las diferentes tareas de una base de datos Transact-SQL de los buques con diversas ya creadas y probadas funciones. Tan sólo hay que ser conscientes de estas funciones, su sintaxis, y los resultados que producen.

Para ayudarle a identificar las funciones que pueden utilizar, son clasificados por su tipo y, probablemente, su utilidad.

Debido a su complejidad, algunos valores pueden ser fácilmente reconocidos o fijo. Por ejemplo, una fecha como el 6 de enero de 1995 es constante y no puede cambiar. Este tipo de valor que se denomina determinista, ya que es siempre la misma. De la misma manera, un valor de tiempo de 5 p.m., como es constante y no puede cambiar. Hay otros valores que no pueden ser conocidos de antemano, ya que el cambio sobre la base de algunas circunstancias. Por ejemplo, la fecha de inicio del año escolar, los cambios de un año a otro, pero siempre se produce. Esto significa que, usted sabe que va a pasar, pero usted no sabe la fecha exacta. Ese valor se denomina no-determinista.

Para apoyar el determinismo y no determinismo, Transact-SQL proporciona dos grandes categorías función que siempre devuelve el mismo valor conocido o que se denomina determinista. Una función puede depender de una condición que se denomina no-determinista.

Valor de un casting

En la mayoría de los casos, un valor que el usuario se somete a su base de datos es principalmen cadena. Esto es conveniente si eso es lo que esperan. Si el valor que el usuario proporciona deben ser más que una cadena, por ejemplo, si el usuario proporciona un número, antes de usar uno de estos va convertir al tipo adecuado, es decir, de una cadena a la tipo esperado.

Para ayudar con la conversión, puede usar tanto el **CAST ()** o el **CONVERT ()** función. La sintaxi función de:

CAST *(expresión* como *DataType)*

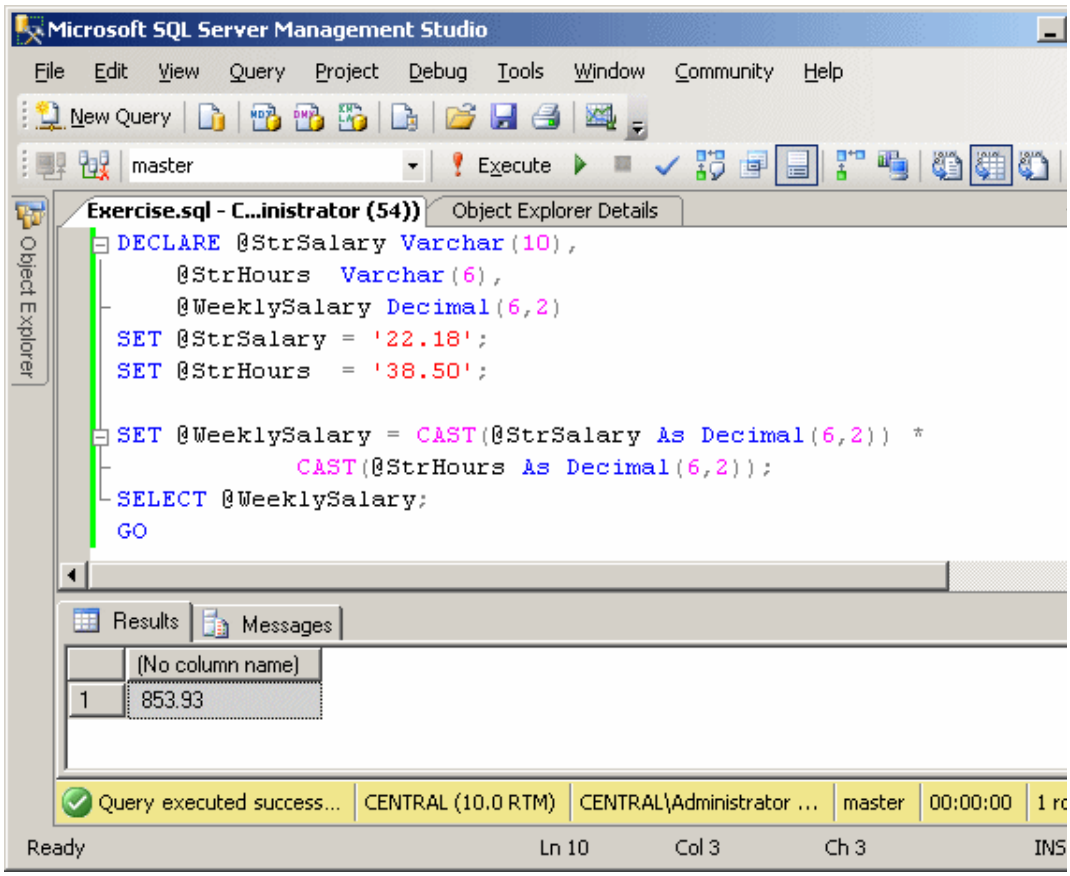
La *expresión* es el valor que tiene que ser emitidos. *DataType* El factor es el tipo de valor que c *expresión.* El *DataType* puede ser uno de los que examinó en **Lección 4.**

En el siguiente ejemplo, dos variables se declaran como cadenas y inicializa. Debido a que debe multiplicación, cada uno se convierte en un tipo de decimales:

```
DECLARE @ StrSalary Varchar (10),
        @ StrHours Varchar (6),
```

```
        @ WeeklySalary decimal (6,2)
SET @ StrSalary = «22 .18 ';
SET @ StrHours ='38 .50 ';

SET @ WeeklySalary = CAST (@ StrSalary Como decimal (6,2)) *
                     CAST (@ StrHours Como decimal (6,2));
SELECT @ WeeklySalary;
IR
```

Aquí hay un ejemplo de ejecución de la declaración anterior:



Convertir un valor

Como **CAST (),** el **CONVERT ()** la función se usa para convertir un valor. A diferencia de **CAST ()** puede utilizar para convertir un valor original de su tipo en un mismo tipo no. Por ejemplo, puede utiliza emitir un número en una cadena y viceversa.

La sintaxis de la **CONVERT ()** es función de:

CONVERT *(DataType* [(longitud)], *expresión* [, estilo])

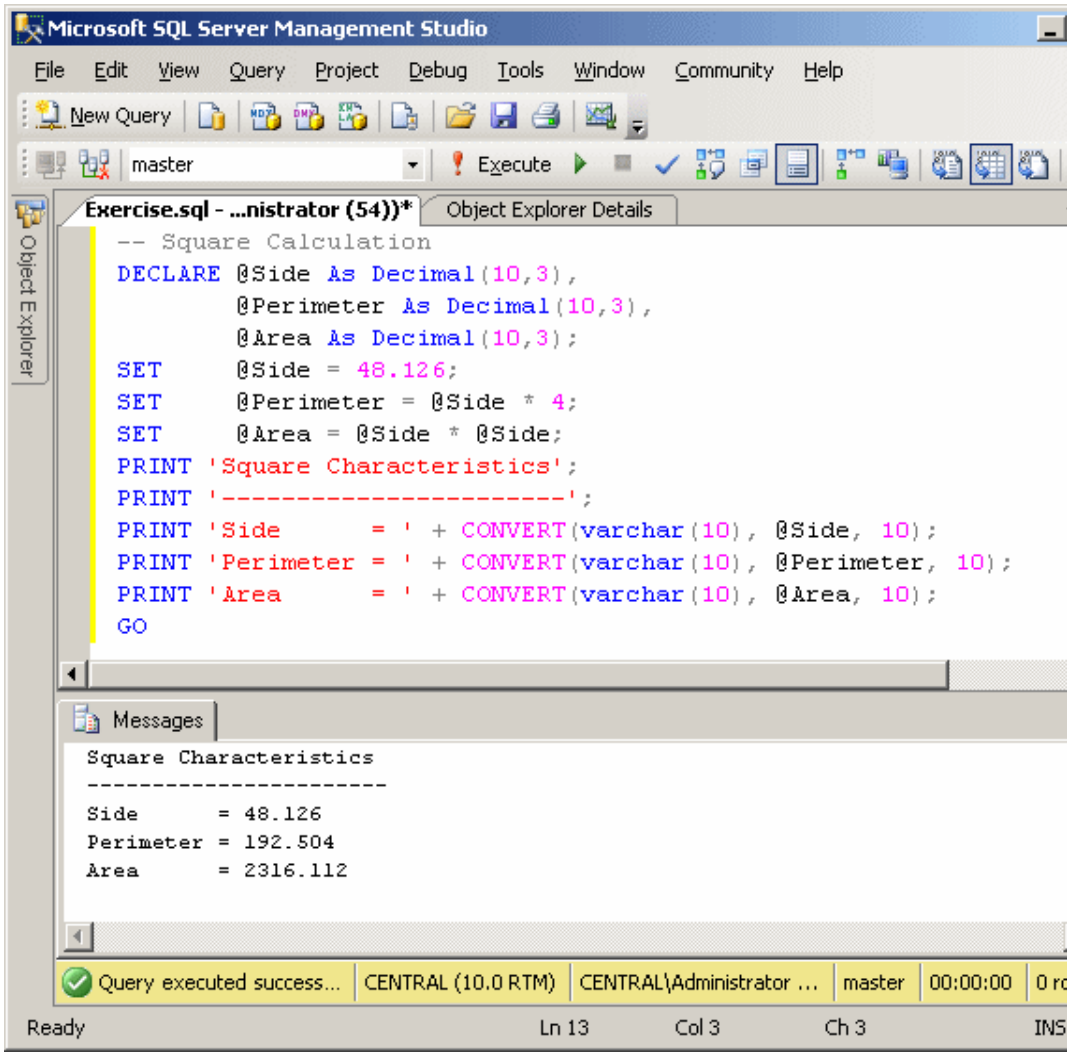
El primer argumento debe ser un conocido tipo de datos, como los que hemos revisado en **Lecció** convierte en una cadena (varchar, nvarchar, char, NCHAR) o un tipo binario, debe especificar el nú permitido el tipo de datos de la propia paréntesis. Como se revisará por **CAST ()** función, la *expresión* que convertir.

Aquí está un ejemplo:

- Plaza de Cálculo

```
DECLARE @ lateral como DECIMAL (10,3),
          @ Perímetro decimal (10,3),
          @ Espacio Como decimal (10,3);
SET @ lateral = 48,126;
SET @ @ Perímetro = lado * 4;
SET @ = Área * @ @ Lado Lado;
IMPRIMIR 'Características de la Plaza ";
IMPRIMIR'-----------------------';
IMPRIMIR 'Cara =' + CONVERT (varchar (10), @ lateral, 10);
IMPRIMIR 'Perímetro =' + CONVERT (varchar (10), @ perimetral, 10);
IMPRIMIR Area = '+ CONVERT (varchar (10), @ Zona, 10);
IR
```

Esto produciría:



Basado en las funciones de cadena

Introducción

La cadena es la más básica, el valor principal que se presenta a una base de datos. Esto se debe a que de ser tratada en particular, en primer lugar se considera una cadena. En una aplicación, existen divers o de obtener una cadena. Usted puede obtener o proporcionar a una función como una constante de c cadena cuyo valor y sin duda usted sabe que le pasa a una función. También puede obtener una ca

proporciona. Otras funciones también pueden producir o devolver una cadena.

Para ayudarle en la gestión de cadenas o la realización de operaciones sobre los mismos, de Transa diversas funciones. Las funciones se pueden dividir en categorías que incluyen basadas en caract además, sub-cadenas, etc

Aprendizaje práctico: Introducción Funciones de Transact-SQL

1. Inicie Microsoft SQL Server y conectar con el servidor
2. Haga clic derecho en el nodo Bases de datos y haga clic en Nueva base de datos ...
3. Establecer el nombre de **Exercise1**
4. Establecer rutas de acceso a la **C: \ Microsoft SQL Server Base de datos de desarrollo**
5. Haga clic en Aceptar
6. Expanda el nodo Bases de datos
7. Expandir Exercise1
8. Expandir programabilidad
9. Expandir Funciones

La longitud de una cadena

Algunas operaciones realizadas en cadenas requieren que usted sabe el número de caracteres de una ca a que algunas operaciones requieren un número mínimo de caracteres y algunas otras funciones que con al menos un carácter. El número de caracteres de una cadena se llama también la longitud de la ca

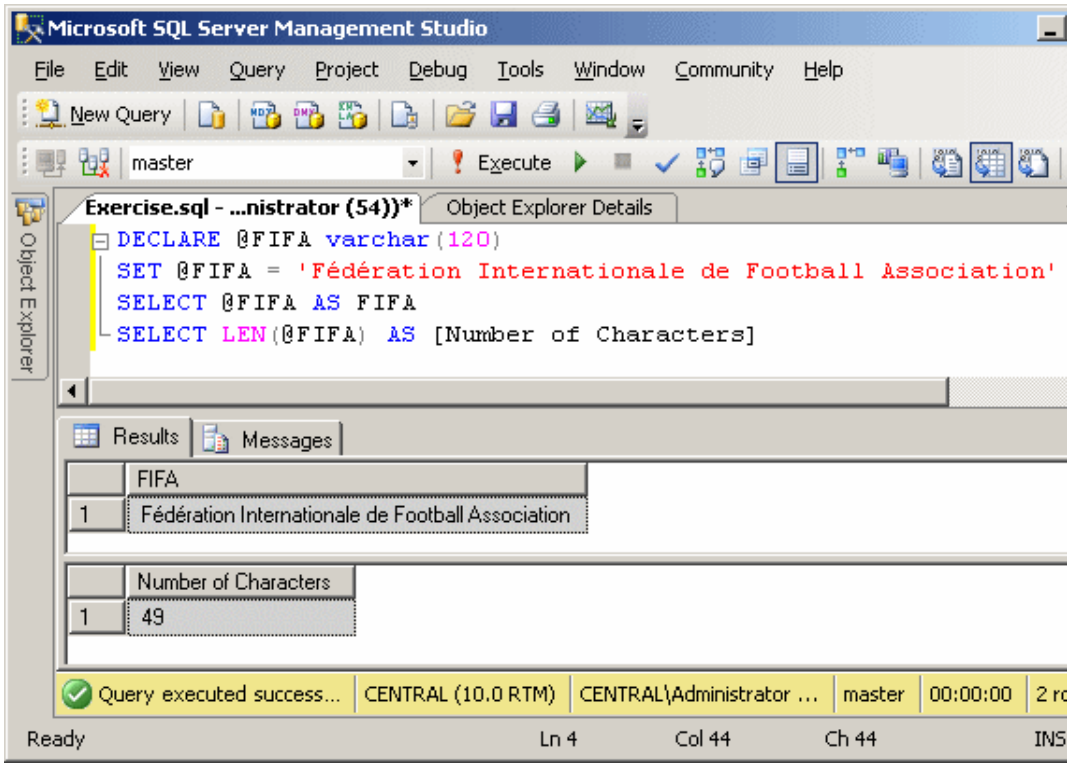
Para obtener la longitud de una cadena, puede usar la **LEN ()** función. Su sintaxis es:

int LEN (String)

Esta función toma un argumento como la cadena a ser considerado. Devuelve el número de caracteres está un ejemplo:

```
DECLARE @ FIFA varchar (120)
SET @ FIFA = 'Fédération Internationale de Football Association "
SELECT @ FIFA como la FIFA
SELECCIONE LEN (@ FIFA) como el [número de caracteres]
```

Esto produciría:



Cadena de conversiones: la conversión de entero a ASCII

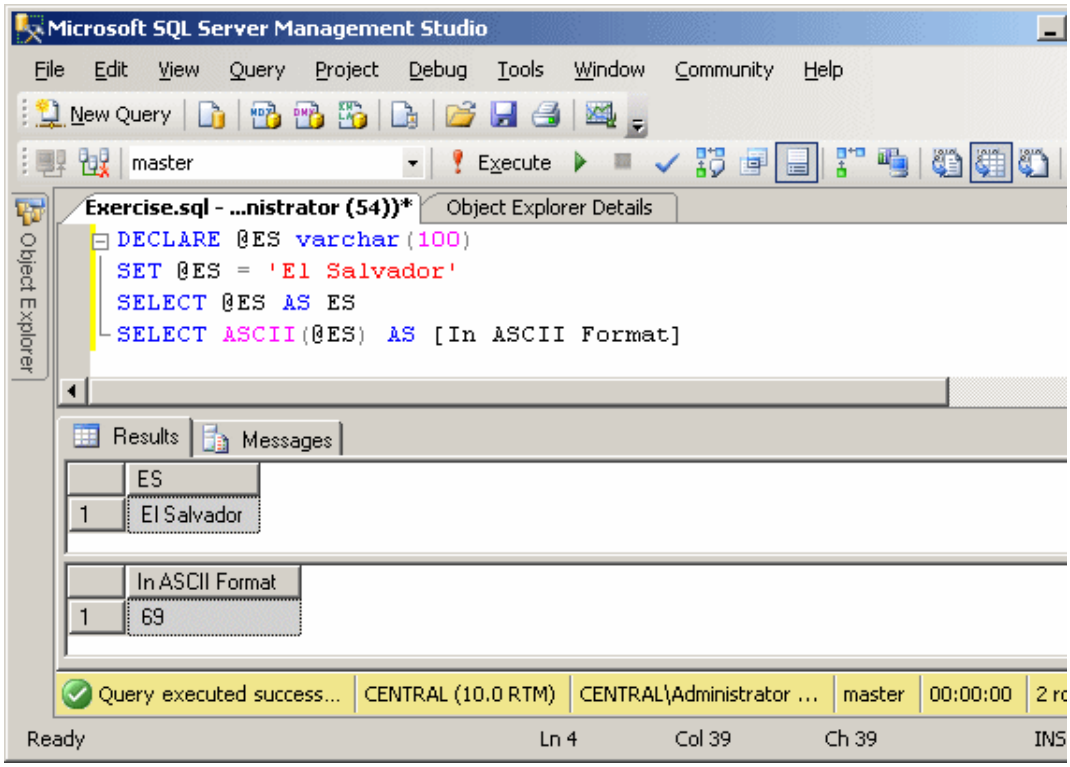
Como ustedes saben ya, una cadena es principalmente una o un grupo de caracteres. Estos person ASCII. Si usted tiene una cadena, para obtener el código ASCII de su carácter extremo izquierdo, pue **()** función. Su sintaxis es:

int ASCII *(String)*

Esta función toma como argumento de cadena y devuelve el código ASCII de la primera (la izquierc cadena. Aquí está un ejemplo:

DECLARE @ ES varchar (100)
SET @ ES = 'El Salvador'
SELECT @ ES COMO ES
SELECCIONE ASCII (@ ES) COMO [en formato ASCII]

Esto produciría:



Cadena de conversiones: la conversión de ASCII a Integer

Si tiene el código ASCII de un carácter y quiere encontrar su carácter real, puede usar el **CHAR ()** funci

char CHAR (int *valor)*

Esta función toma como parámetro un valor numérico como un entero. Tras la conversión, la equivalente en ASCII de ese número.

Cadena de conversiones: Minúsculas

Como ustedes saben ya, una cadena puede ser de mayúsculas, minúsculas y símbolos que no tiener Cuando reciba una cadena, si desea convertir todos sus caracteres en minúsculas, puede utilizar el **LO** sintaxis es:

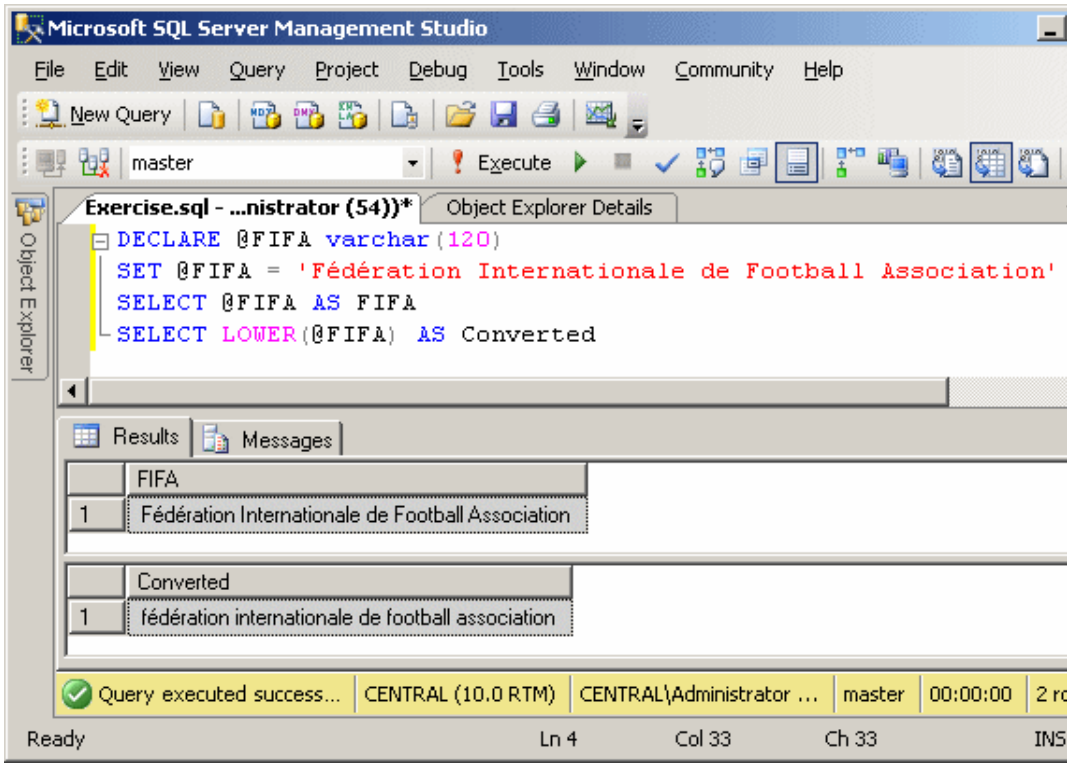
varchar LOWER (String)

Esta función toma como argumento una cadena. Cualquier letra minúscula, que es parte de la cadena la correspondencia que es parte de la cadena se convierte a minúsculas. Cualquier otro carácte mantenga "como está". Después de la conversión, el **LOWER ()** devuelve una nueva cadena.

Aquí está un ejemplo:

DECLARE @ FIFA varchar (120)
SET @ FIFA = 'Fédération Internationale de Football Association "
SELECT @ FIFA como la FIFA
SELECCIONE LOWER (@ FIFA) y que se convirtió

Esto produciría:



Aprendizaje práctico: Conversión de una cadena a minúsculas

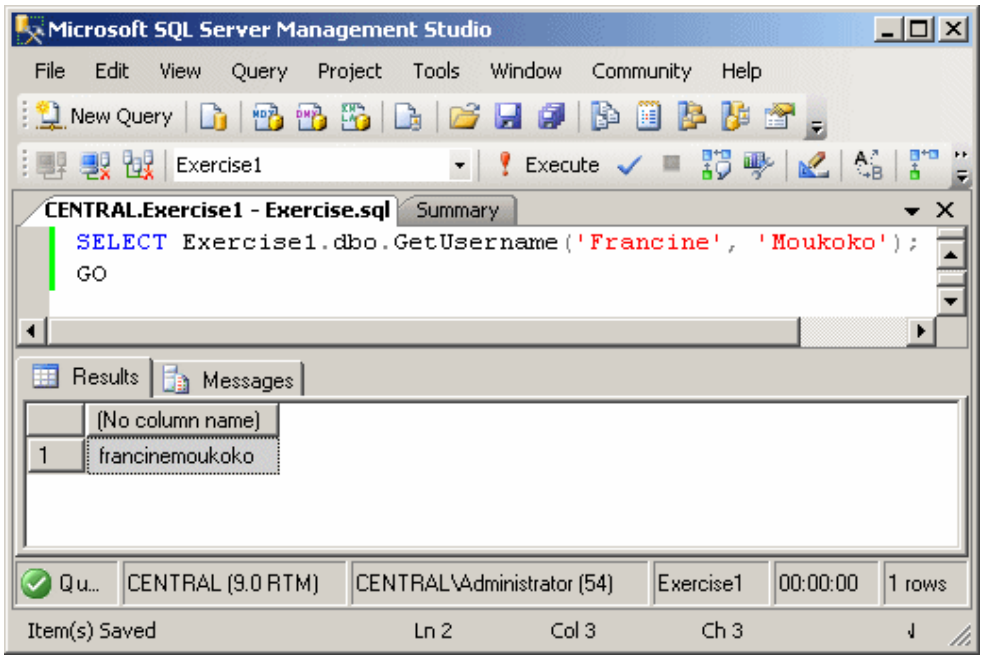
1. En el Explorador de objetos, haga clic derecho en Exercise1 y haga clic en Nueva consulta ...
2. Para crear una función, escriba lo siguiente:

```
- ==============================================
- Función: GetUserName
- ==============================================

CREATE FUNCTION GetUserName
          (@ Nombre varchar (40),
           @ Apellido varchar (40))
VUELVE varchar (50)
COMO
COMENZAR
          DECLARE @ Usuario varchar COMO (50);
          SELECT @ Nombre de usuario = LOWER (@ Nombre) + LOWER (@
Apellido);
          RETORNO @ Nombre de usuario;
FIN
IR
```

3. Presione la tecla F5 para ejecutar la declaración
4. Guardar el código como **ejercicio**
5. Presione Ctrl + A para seleccionar el código y pulse Borrar para eliminar el código
6. Para probar la función, escriba lo siguiente:

```
SELECCIONE Exercise1.dbo.GetUsername ( 'Francine', 'Moukoko');
IR
```

7. Presione la tecla F5 para ejecutar la declaración



8. En el Explorador de objetos, expanda el nodo Bases de datos en caso necesario, y ampliar Exercise
9. Expandir programabilidad
10. Expandir Funciones
11. Ampliar las funciones escalares-Valor
12. Haga clic con el botón y haga clic en Eliminar dbo.GetUserName
13. En el cuadro de diálogo Eliminar objetos, haga clic en Aceptar

Sub-Cuerdas: El Inicio de una cadena de caracteres

Un sub-cadena es una sección obtenido a partir de una cadena. La idea es aislar a uno o un grupo de c para cualquier razón.

Un sub-izquierda es una cadena o un grupo de personajes extraídos del lado izquierdo de una con obtener la sub-cadena de izquierda de una cadena, puede usar la **izquierda ()** función. Su sintaxis es:

IZQUIERDA varchar (String, *NumberOfCharacters)*

Esta función toma dos argumentos. El primer argumento especifica la cadena original. El segundo arg número de caracteres de la mayoría de izquierda, que constituyen la sub-cadena. Después de la operac devuelve una nueva cadena de la izquierda + el carácter *NumberOfCharacters* en su derecho de la *cuerd*

Aprendizaje práctico: Creación de una sub-cadena con caracteres restante

1. Eliminar el código anterior de la ventana de consulta
2. Para utilizar la **izquierda ()** función, escriba lo siguiente:

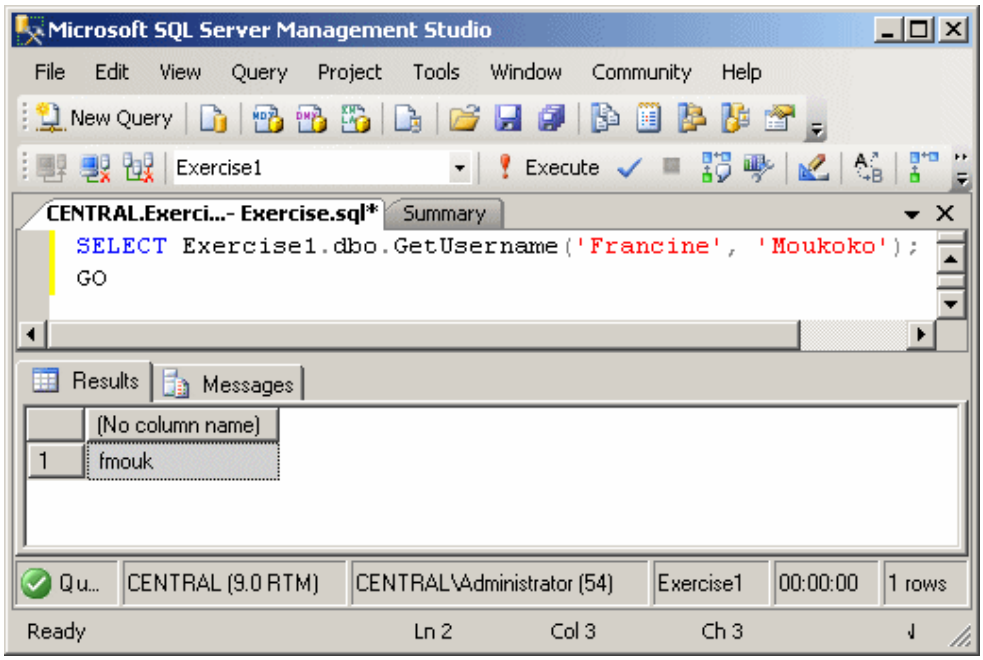
```
- ============================================
- Función: GetUserName
- ============================================

CREATE FUNCTION GetUserName
```

```
            (@ Nombre varchar (40),
             @ Apellido varchar (40))
VUELVE varchar (50)
COMO
COMENZAR
            DECLARE @ Usuario varchar COMO (50);
            SELECT @ Nombre de usuario = LOWER (IZQUIERDA (@ FirstName, 1)) +
                                         IZQUIERDA (LOWER (@ Apellido), 4)
            RETORNO @ Nombre de usuario;
FIN
IR
```

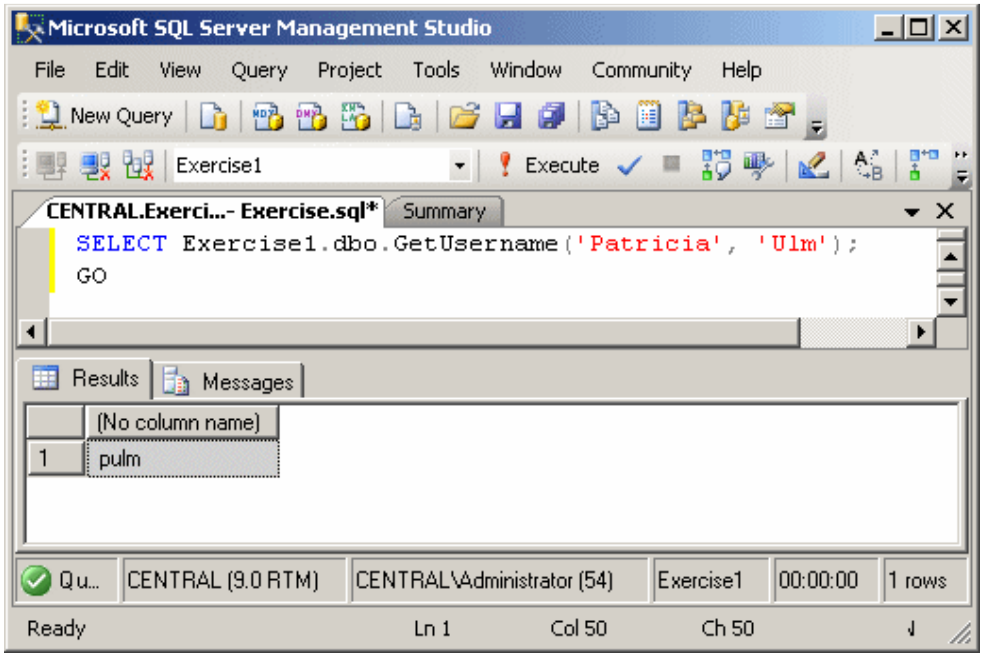
3. Presione la tecla F5 para ejecutar la declaración
4. Eliminar el código en la ventana de consulta
5. Para probar la función, escriba lo siguiente:

```
SELECCIONE Exercise1.dbo.GetUsername ( 'Francine', 'Moukoko');
IR
```

6. Ejecución de la declaración en la ventana



7. Cambiar la llamada con un nombre más corto que 5 caracteres como "Um" y la ejecución de la dec
un ejemplo:



Sub-cadenas: el final de una cadena de caracteres

En lugar de los caracteres a partir de una cadena, es posible que desee crear una cadena usando caracteres de la derecha una cadena existente. En apoyo de esta operación, de Transact-SQL estab función. Su sintaxis es:

DERECHO varchar (String, *NumberOfCharacters)*

Esta función toma dos argumentos. El primer argumento especifica la cadena original. El segundo arg número de caracteres de la mayoría de derecha, que constituyen la sub-cadena.

Aprendizaje práctico: Obtener el derecho Personajes

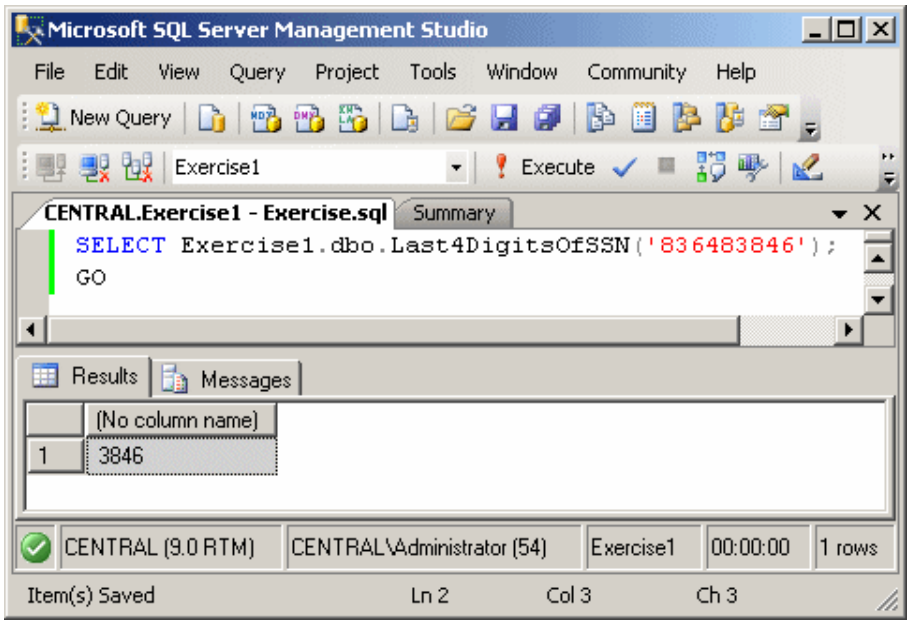
1. Eliminar el código anterior de la ventana de consulta
2. Para crear una función, escriba lo siguiente:

```
- ==============================================
- Función: Last4DigitsOfSSN
- ==============================================

CREATE FUNCTION Last4DigitsOfSSN (SSN @ varchar (12))
VUELVE char (4)
COMO
COMENZAR
        DERECHO DE DEVOLUCIÓN (SSN @, 4);
FIN
IR
```

3. Presione la tecla F5 para ejecutar la declaración
4. Eliminar el código anterior de la ventana de consulta
5. Para probar la función, escriba lo siguiente:

```
SELECCIONE Exercise1.dbo.Last4DigitsOfSSN ('836483846 ');
IR
```

6. Ejecución de la declaración



Sub-Cuerdas: Sustitución de sucesos en un String

Una de las más molestas situaciones que puede encontrar con una cadena es para hacer frente caracteres inesperados. Esto podría ser debido a su formato o cualquier otra razón. Por ejemplo, número de teléfono de un usuario, hay varias formas en que la cadena se podría presentar a usted. E 000-000-0000, o 0000000000, o (000) 000-0000. Cada uno de estos formatos es aceptable EE.UU. y C teléfono pero si implica que la cadena en una operación, usted puede obtener un resultado impredecible resolver este tipo de problemas es eliminar cualquier mal los caracteres de la cadena. Esta opera consistir en la sustitución de algún personaje (s) con otro (s).

Para sustituir un carácter o una sub-cadena a partir de una cadena, puede usar el **SUST ()** función. Su

SUST varchar (String, *FindString, ReplaceWith)*

o

SUST binario (String, *FindString, ReplaceWith)*

Esta función tiene tres argumentos. El primero es la cadena que se utilizará como referencia. El s *FindString,* es un personaje o un sub-cadena a buscar en la *cadena* argumento. *FindString* Si el carác encuentra en la *cuerda,* entonces se sustituye por el valor del último argumento, *ReplaceWith.*

Aprendizaje práctico: La sustitución de personajes o Sub-Cuerdas

1. En el Explorador de objetos, eliminar dbo.Last4DigitsOfSSN
2. Eliminar el código en la ventana de consulta
3. Para utilizar el **SUST ()** función, cambie el código de la siguiente manera:

```
- ==========================================
- Función: Last4DigitsOfSSN
- ==========================================

CREATE FUNCTION Last4DigitsOfSSN (SSN @ varchar (12))
VUELVE char (4)
COMO
COMENZAR
```

```
        DECLARE @ StringWithoutSymbol Como varchar (12);
        - En primer lugar eliminar los espacios vacíos
        SET @ StringWithoutSymbol = SUST (SSN @ ',','');
        - Ahora eliminar los guiones "-" si existen
        SET @ StringWithoutSymbol = SUST (@ StringWithoutSymbol, '-','');
        DERECHO DE DEVOLUCIÓN (@ StringWithoutSymbol, 4);
FIN
IR
```

4. Ejecución de la declaración
5. Eliminar el código en la ventana de consulta
6. Para probar la función, escriba lo siguiente

```
SELECCIONE Exercise1.dbo.Last4DigitsOfSSN ('244-04-8502 ');
IR
```

7. Ejecución de la declaración en la ventana

Funciones aritméticas

El signo de un número

En aritmética, un número, ya que se considera negativo (menos de 0), nulo (igual a 0) o positiva (su un número es negativo, debe tener una - símbolo a su izquierda. Si es positivo, puede mostrar un símb o puede omitirlo. Un número sin el - o + símbolo a la izquierda se considera positiva, también conocido símbolo que determina si un número es positivo o negativo se refiere a su signo. El signo es fácilme sabe ya el número. En algunos casos, cuando un número se presenta a su solicitud, antes de tomar cua que sea necesario para obtener esta pieza de información.

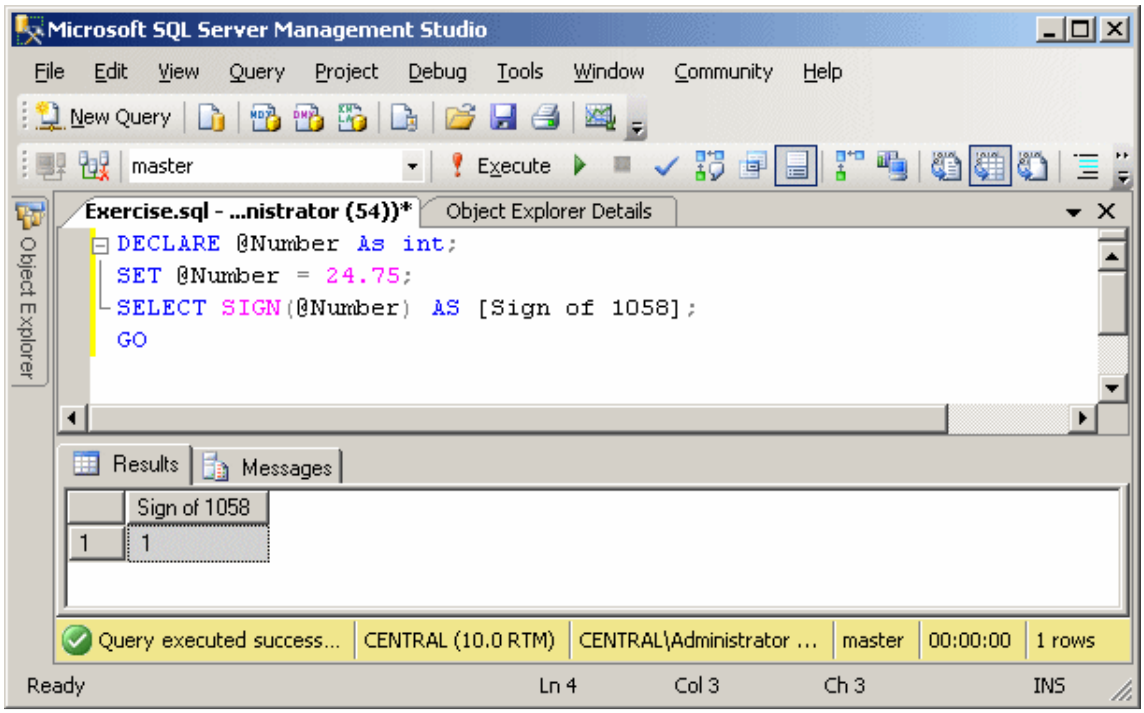
Para averiguar si un valor es positivo, nulo o negativo, de Transact-SQL proporciona la **SIGNO ()** funció

SIGN *(Expresión)*

Esta función toma como argumento un número o una expresión que puede ser evaluado a un núme examinará a continuación el número:

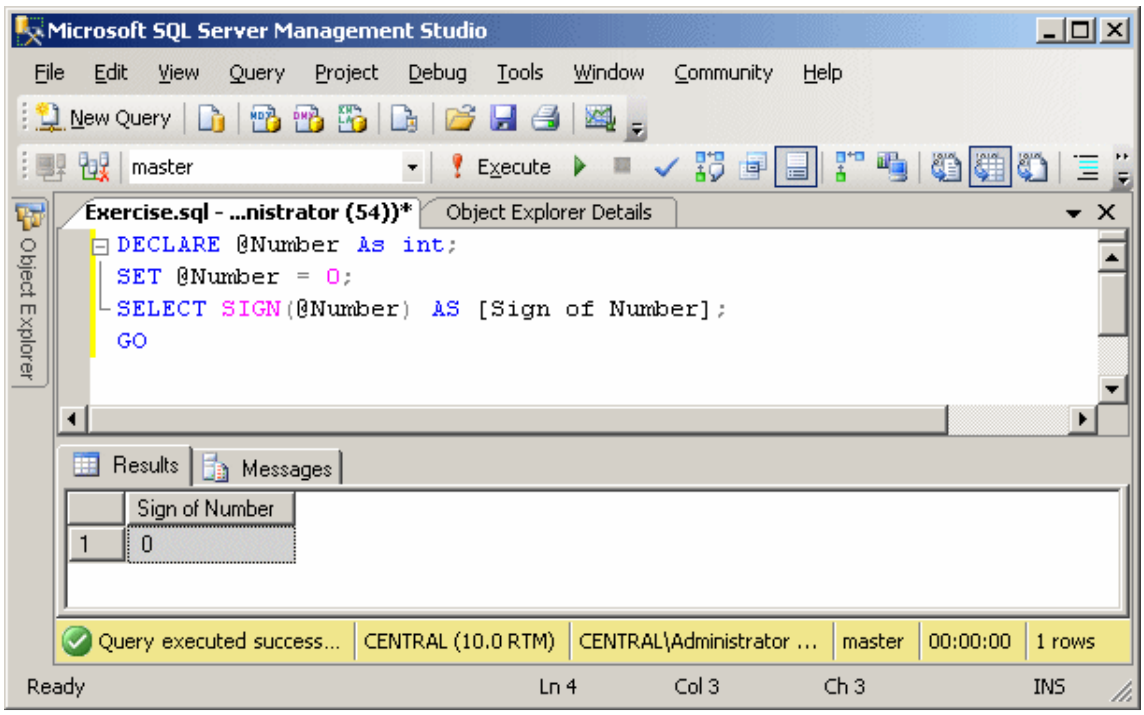
- Si la *expresión* es positiva, la función devuelve 1. Aquí está un ejemplo:

```
DECLARE @ Número Como int;
SET @ Numero = 24,75;
Seleccionar SIGN (@ Numero) COMO [Entrar de 1058];
IR
```



- Si la *expresión* es nula, la función devuelve 0

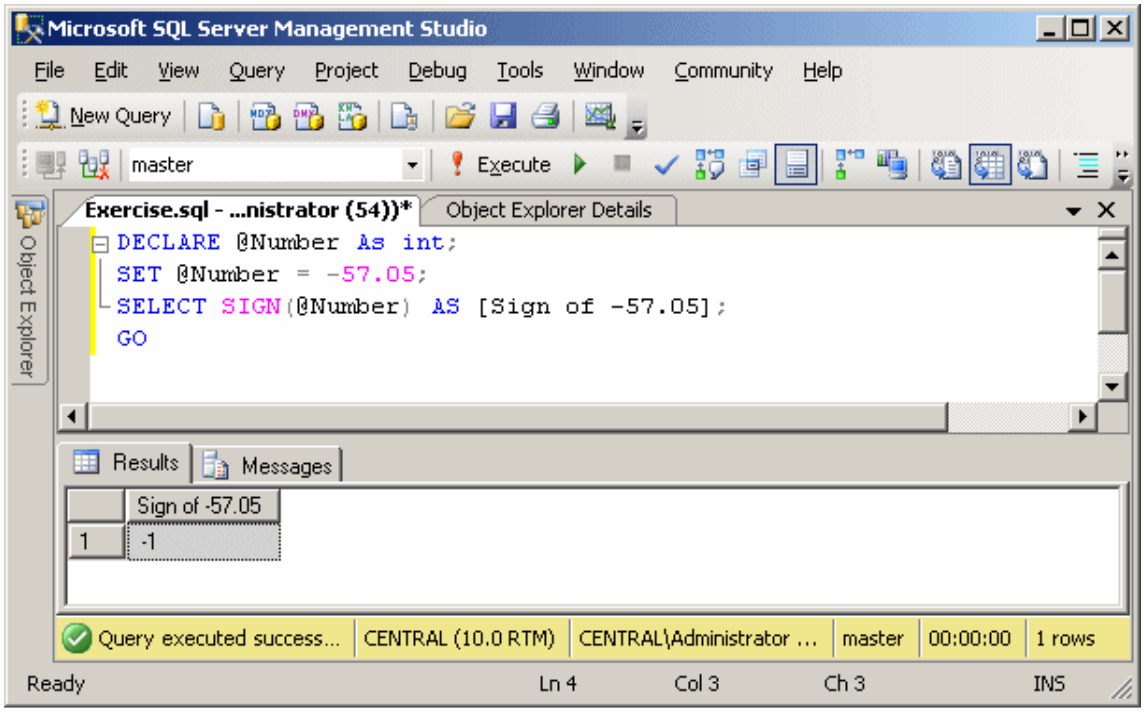
  DECLARE @ Número Como int;
  SET @ Numero = 0;
  Seleccionar SIGN (@ Numero) COMO [Entrar Número de];
  IR



- Si la *expresión* es negativa, la función devuelve -1

  DECLARE @ Número Como int;

SET @ Numero = -57,05;
Seleccionar SIGN (@ Numero) COMO [Entrar de -57,05];
IR

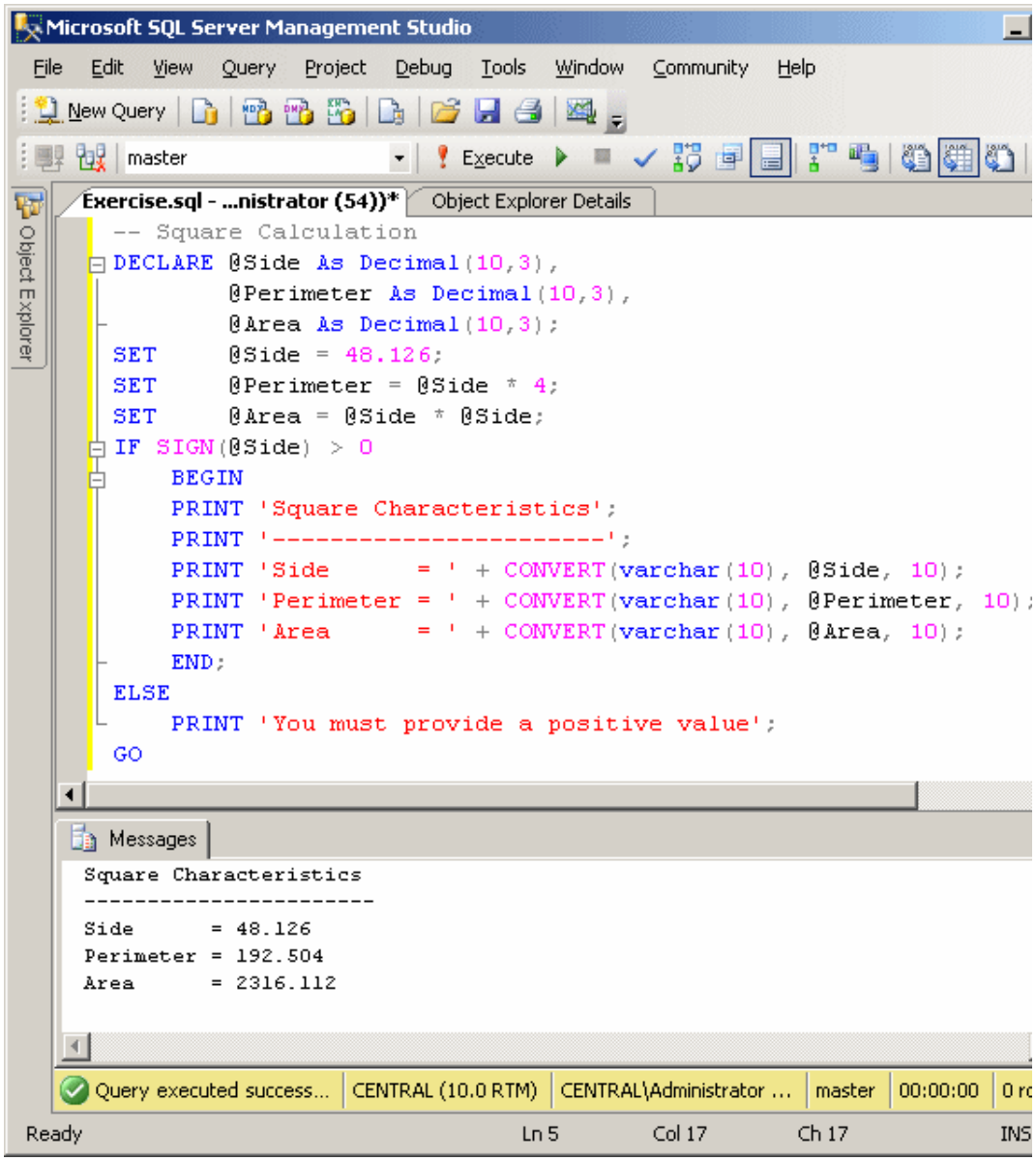


Basado en esto, puede utilizar **el** signo **()** para averiguar si un valor es negativo, nulo, o positivo: si valor (o una variable) **para** firmar **()** y usar un operador lógico, para comprobar su signo. Aquí es un eje

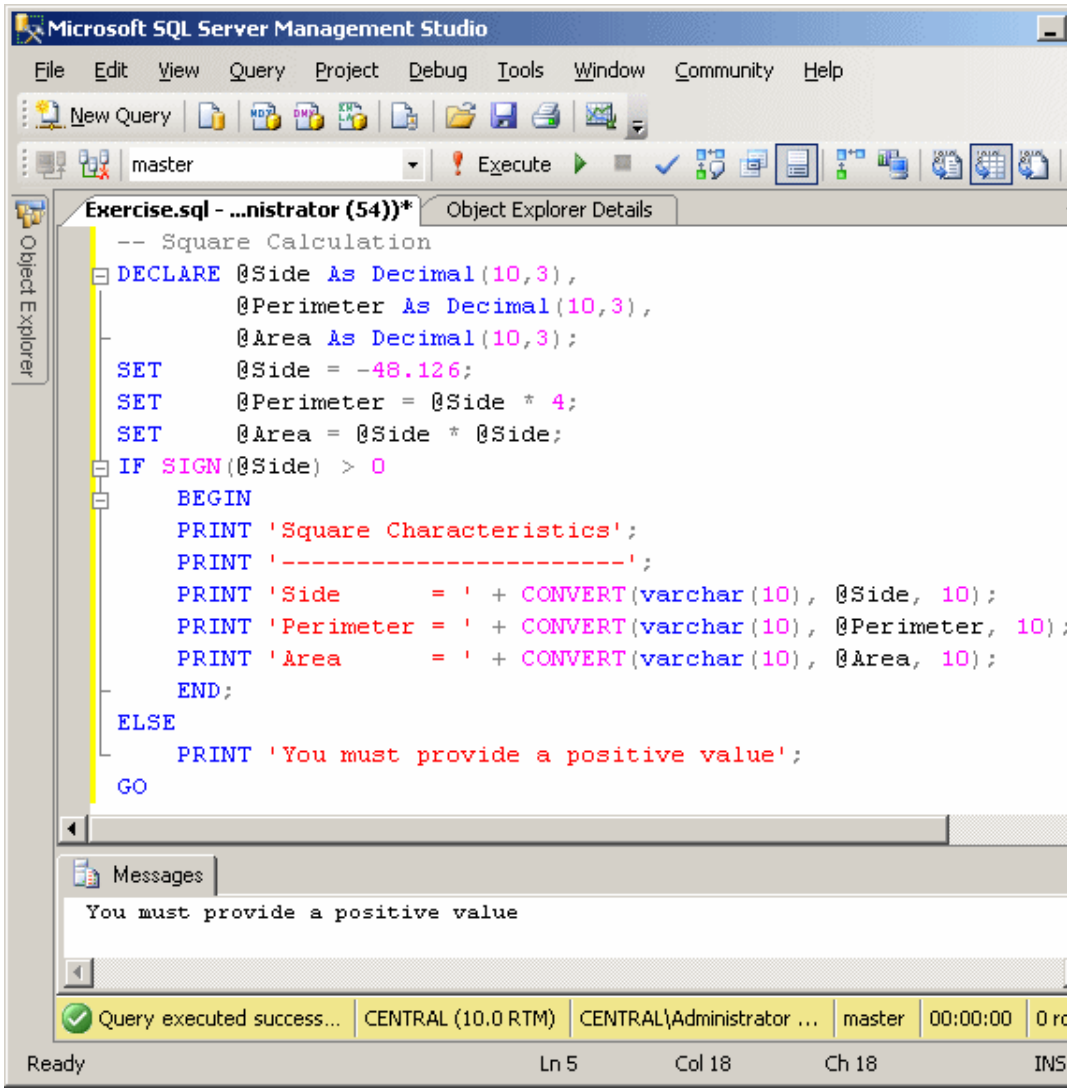
- Plaza de Cálculo

```
DECLARE @ lateral como DECIMAL (10,3),
          @ Perímetro decimal (10,3),
          @ Espacio Como decimal (10,3);
SET @ lateral = 48,126;
SET @ @ Perímetro = lado * 4;
SET @ = Área * @ @ Lado Lado;
SI FIRME (@ lateral)> 0
     COMENZAR
          IMPRIMIR 'Características de la Plaza ";
          IMPRIMIR'-----------------------';
          IMPRIMIR 'Cara =' + CONVERT (varchar (10), @ lateral, 10);
          IMPRIMIR 'Perímetro =' + CONVERT (varchar (10), @ perimetral, 10);
          IMPRIMIR Area = '+ CONVERT (varchar (10), @ Zona, 10);
     END;
ELSE
          IMPRIMIR 'Usted debe proporcionar un valor positivo »;
IR
```

Aquí hay un ejemplo de ejecución de la declaración:



Aquí hay otro ejemplo de la ejecución de la declaración:



Observe que el intérprete actúa de forma diferente en respuesta al resultado **del** signo **()** función.

El valor absoluto de un número

El sistema numérico decimal cuenta desde menos infinito hasta el infinito. Esto significa que los negativos o positivos, en función de su posición de 0, que es considerado como neutro. En a considerado el número tendrá que ser positiva, incluso si sólo se presenta en un formato de negativo. un número x es x si el número es (ya) positivo. Si el número es negativo, su valor absoluto es su equiv ejemplo, el valor absoluto de 12 es 12, mientras que el valor absoluto de -12 es 12.

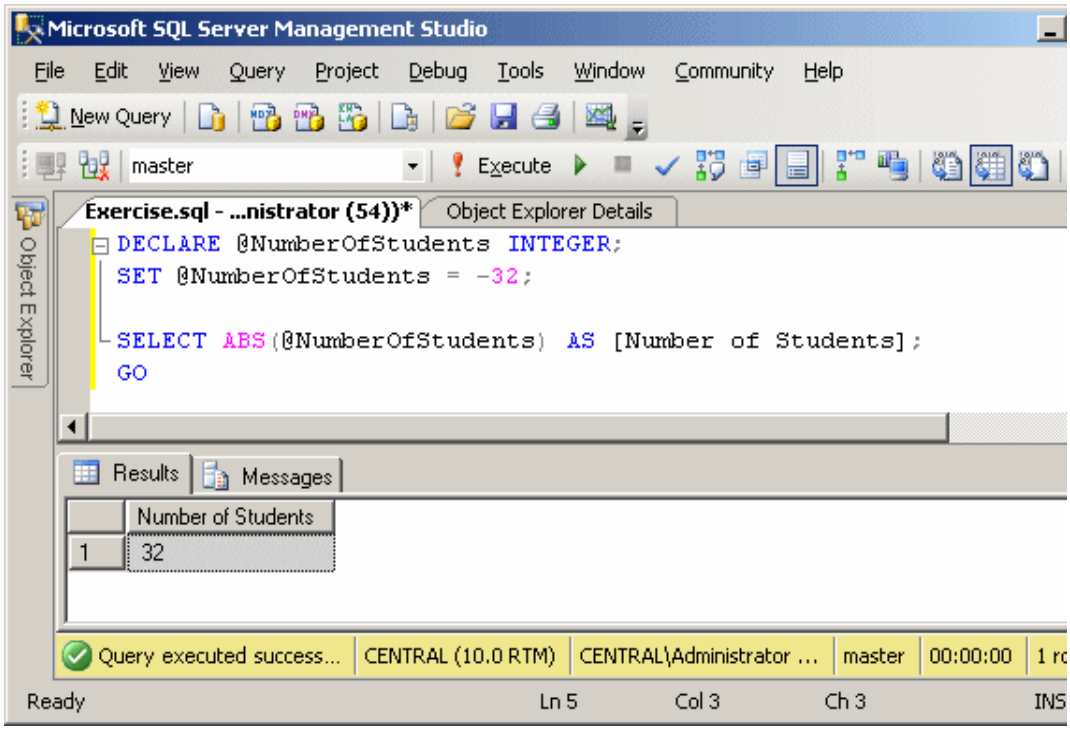
Para obtener el valor absoluto de un número, puede utilizar el **ABS ()** función. Su sintaxis es:

ABS *(expresión)*

Esta función tiene una expresión o un número como argumento y devuelve su valor absoluto. Aquí está

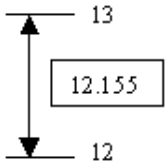
```
DECLARE @ NumberOfStudents INTEGER;
SET @ NumberOfStudents = -32;

SELECCIONE ABS (@ NumberOfStudents) COMO [Número de estudiantes];
IR
```

Esto produciría:



El techo de un número

Considere la posibilidad de un valor decimal, como 12,155. Este número está entre 12 y entero entero



De la misma manera, considerar un número como -24,06. Como este número es negativo, es de entre ser mayor.

En álgebra, el límite máximo de un número entero más cercano es el que es superior o mayor que el r En el primer caso, el límite máximo de 12.155 es de 13 porque 13 es el más cercano número entero 12,155. El límite máximo de -24,06 es -24.

Para obtener el límite máximo de un número, de Transact-SQL establece el **límite máximo ()** función.

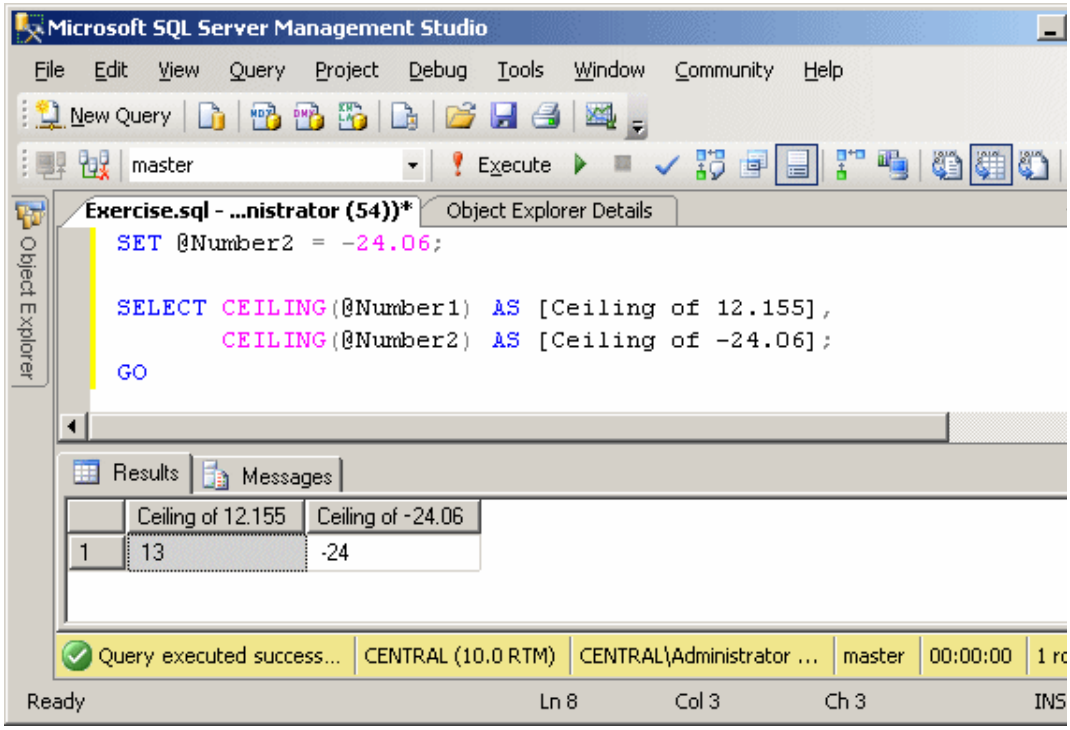
TECHO *(Expresión)*

Esta función toma como argumento un número o una expresión que puede evaluar a un núme conversión, si la función tiene éxito, devuelve un número de doble precisión que es mayor o igual a la e un ejemplo:

```
DECLARE @ Number1 Como Numérico (6, 2),
                    Como @ Number2 Numérico (6, 2)
SET @ Number1 = 12,155;
SET @ Number2 = -24,06;

SELECCIONE TECHO (@ Number1) COMO [techo de 12,155],
```

TECHO (@ Number2) COMO [techo de -24,06];
IR

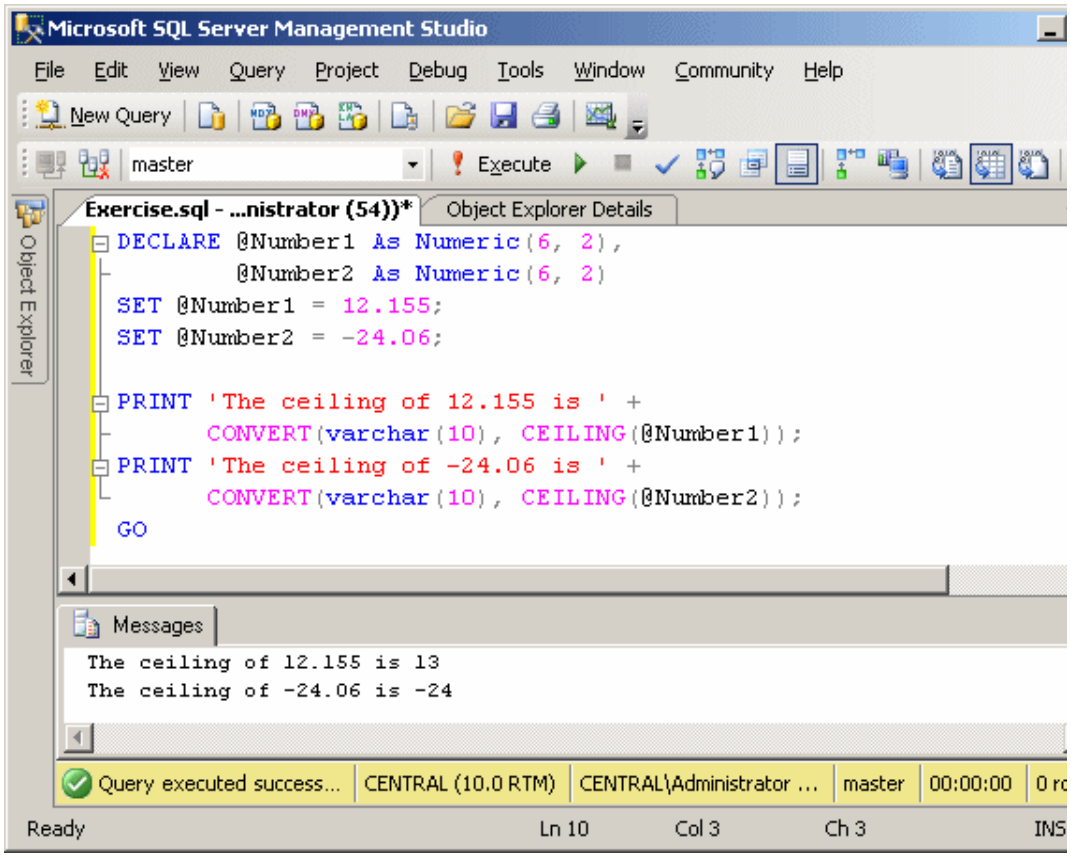
Esto produciría:



Es otra forma de mostrar los resultados anteriores:

```
DECLARE @ Number1 Como Numérico (6, 2),
                    Como @ Number2 Numérico (6, 2)
SET @ Number1 = 12,155;
SET @ Number2 = -24,06;

IMPRIMIR 'El límite máximo es de 12,155 +
            CONVERT (varchar (10), TECHO (@ Number1));
IMPRIMIR 'El límite máximo es de -24,06' +
        CONVERT (varchar (10), TECHO (@ Number2));
IR
```

Esto produciría:



El piso de un número

Considere la posibilidad de dos números decimales, tales como 128.44 y -36.72. 128,44 es el número
128 es la más baja. El número es -36,72 entre -37 y -36, con -37 es la más baja. El menor número
pero el valor de una serie que se conoce como su palabra. Sobre esta base, el piso de 128,44 es de
36,72 es -37.

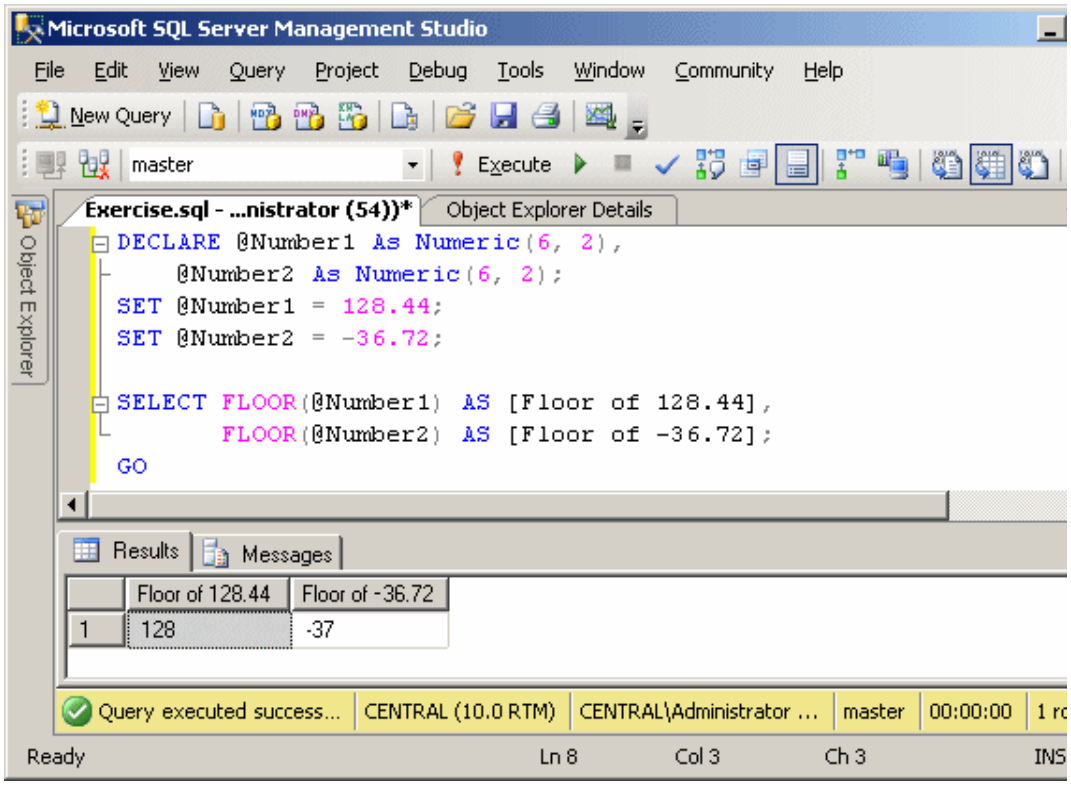
Para apoyar la búsqueda de la palabra de un número, de Transact-SQL proporciona la **PISO ()** función.

PISO *(Expresión)*

La **PISO ()** la función toma como argumento un valor numérico o una expresión que puede ser evalua
la función tiene éxito en su conversión, que produce el entero es que el piso de la argumentación. Aquí

```
DECLARE @ Number1 Como Numérico (6, 2),
          Como @ Number2 Numérico (6, 2);
SET @ Number1 = 128,44;
SET @ Number2 = -36,72;

SELECCIONE PISO (@ Number1) COMO [Piso de 128,44],
          PISO (@ Number2) COMO [Planta de -36,72];
IR
```

Esto produciría:



El exponente de un número

Para calcular el valor exponencial de un número, de Transact-SQL proporciona el **CAD ()** función. Su si

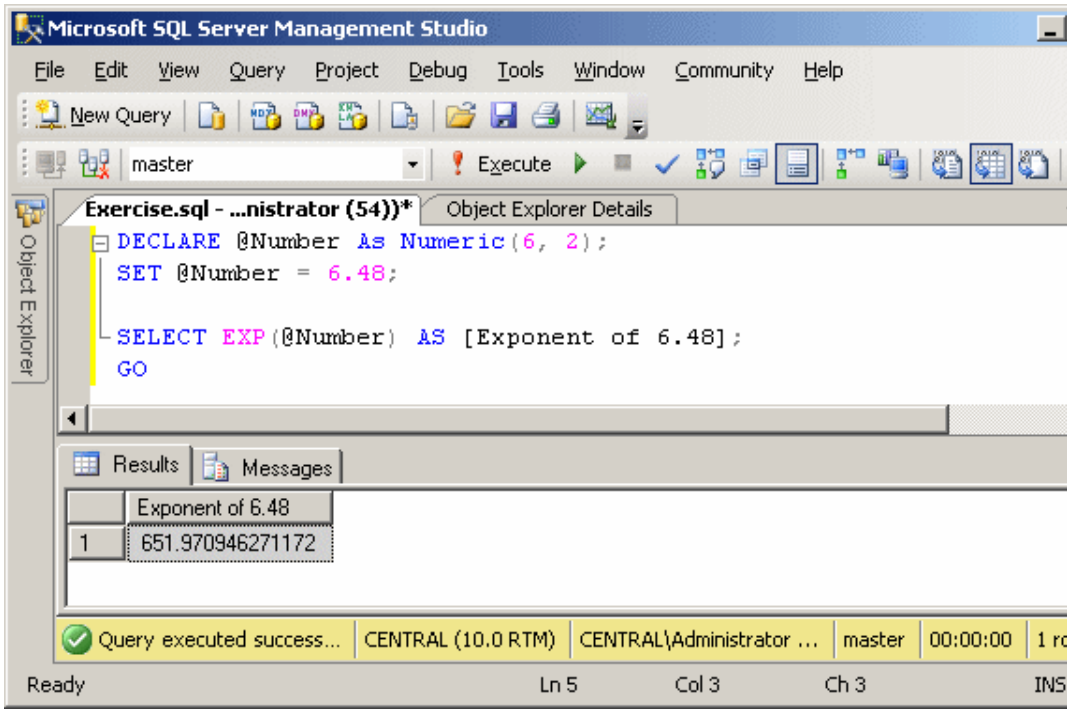
CAD *(Expresión)*

Esta función toma un argumento como un número o una expresión que puede ser evaluado a un nú
ejemplo:

DECLARE @ Número Como numérico (6, 2);
SET @ Numero = 6,48;

SELECCIONE CAD (@ Numero) COMO [Exponente de 6,48];
IR

Esto produciría:



El poder de un número

El poder de un número es el valor de ese número cuando se planteó a otro número. Esto se hace u siguiente:

ReturnValue = x^y

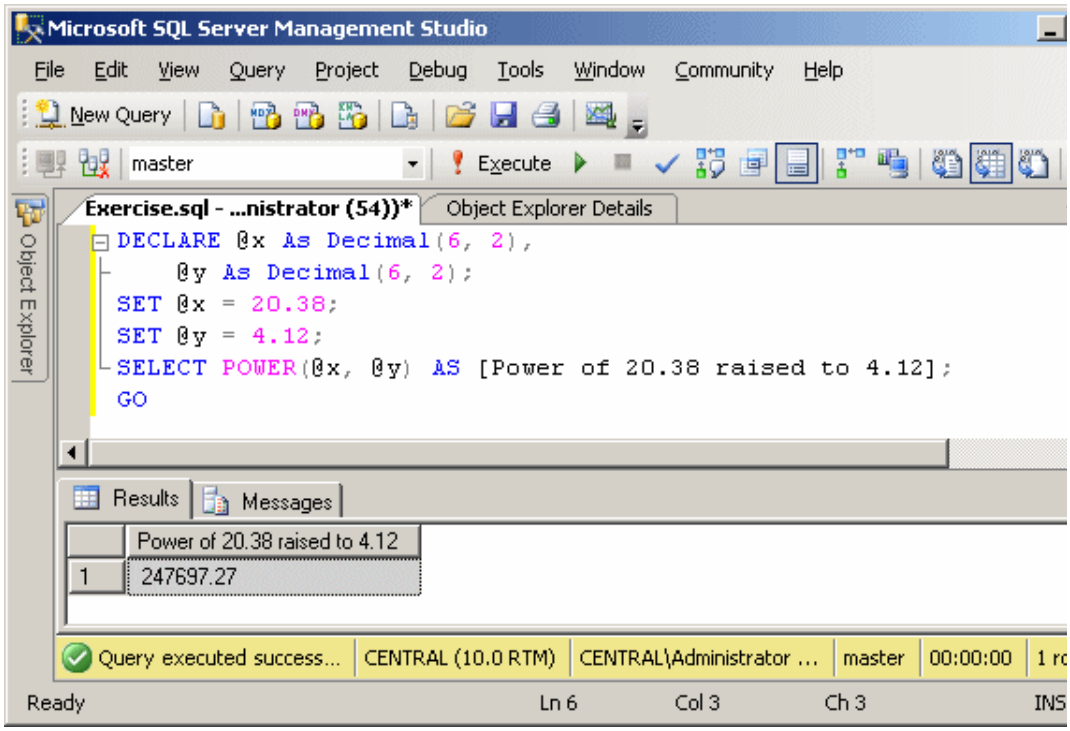
Para apoyar la búsqueda de la potencia de un número, de Transact-SQL proporciona la **potencia ()** f es:

POWER (x, y)

Esta función toma dos argumentos necesarios. El primer argumento, x, se utiliza como base el nú segundo argumento, y, también llamado el exponente, se x para aumentar este valor. He aquí un ejemp

```
DECLARE @ x Como decimal (6, 2),
          Como @ y decimales (6, 2);
SET @ x = 20,38;
SET @ y = 4,12;
SELECCIONE POWER (@ x, @ y) como el [Poder de 4,12 a 20,38 planteadas];
IR
```

Esto produciría:



El logaritmo natural de un número

Para ayudar a encontrar el logaritmo natural de un número, de Transact-SQL proporciona el **LOG ()** f
es:

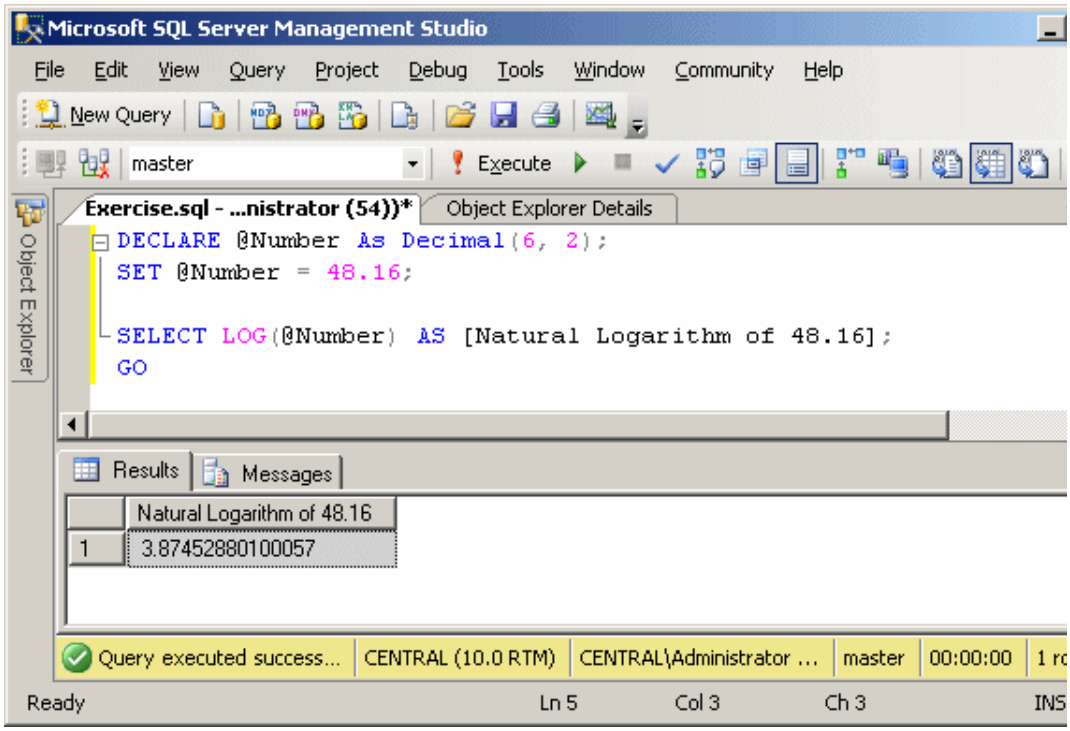
LOG *(Expresión)*

Esta función toma un argumento como un número o una expresión que puede evaluar a un número.
devuelve el logaritmo natural del argumento. Aquí está un ejemplo:

DECLARE @ Número Como decimales (6, 2);
SET @ Numero = 48,16;

Seleccione Registro (Número @) como el [logaritmo natural de 48,16];
IR

Esto produciría:



El logaritmo en base 10 de un número

Para calcular el logaritmo en base 10 de un número, de Transact-SQL proporciona la **LOG10 ()** función.

LOG10 *(Expresión)*

El número a ser evaluados se pasa como argumento X. La función devuelve el logaritmo en base 10 siguiente:

y = log10x

lo que equivale a

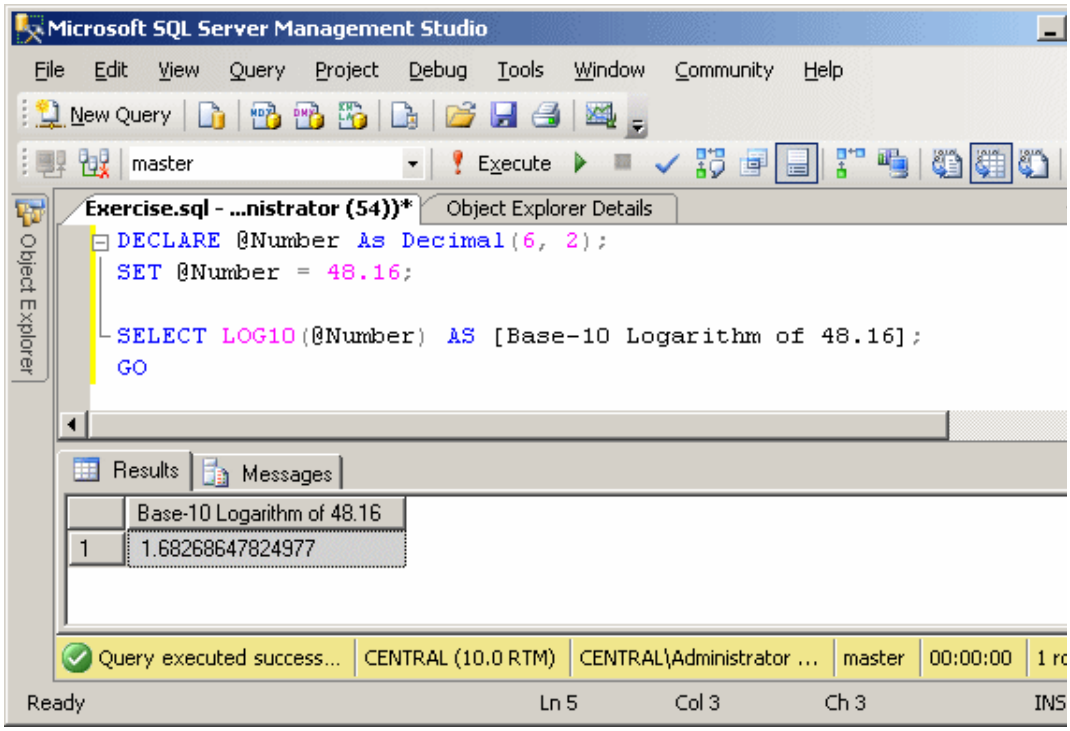
$x = 10^{y}$

Aquí está un ejemplo:

DECLARE @ Número Como decimales (6, 2);
SET @ Numero = 48,16;

SELECCIONE LOG10 (@ Numero) COMO [logaritmo en base 10 de 48,16];
IR

Esto produciría:



La raíz cuadrada

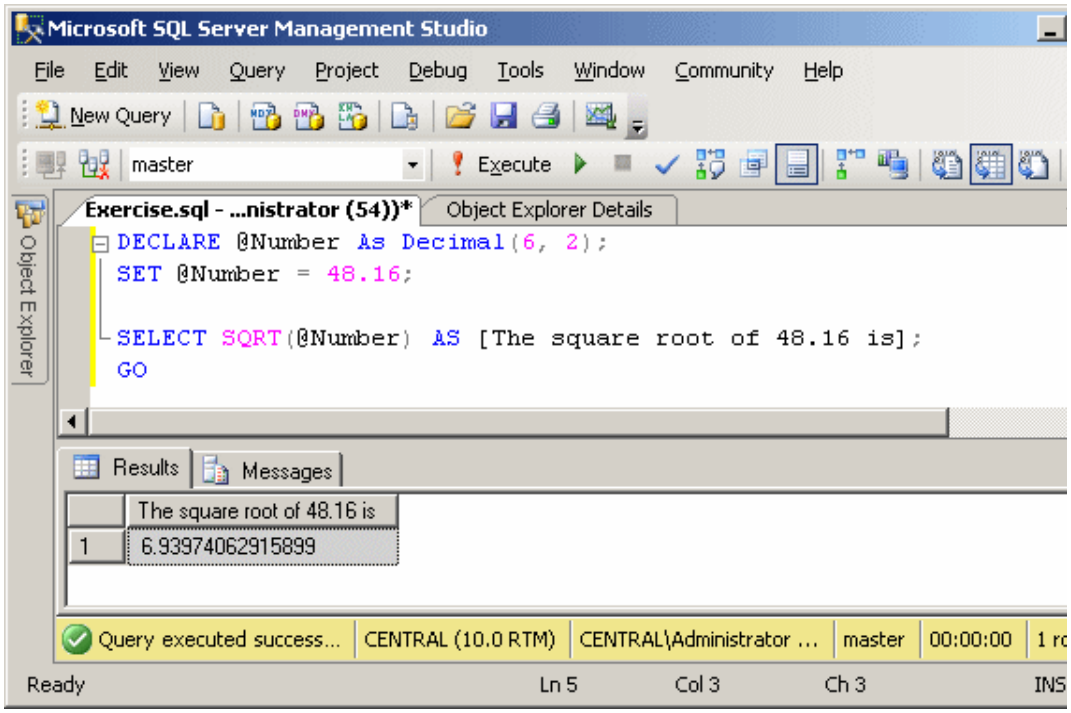
Para apoyar el cálculo de una raíz cuadrada, de Transact-SQL proporciona la **SQRT ()** función. Su sinta

SQRT *(expresión)*

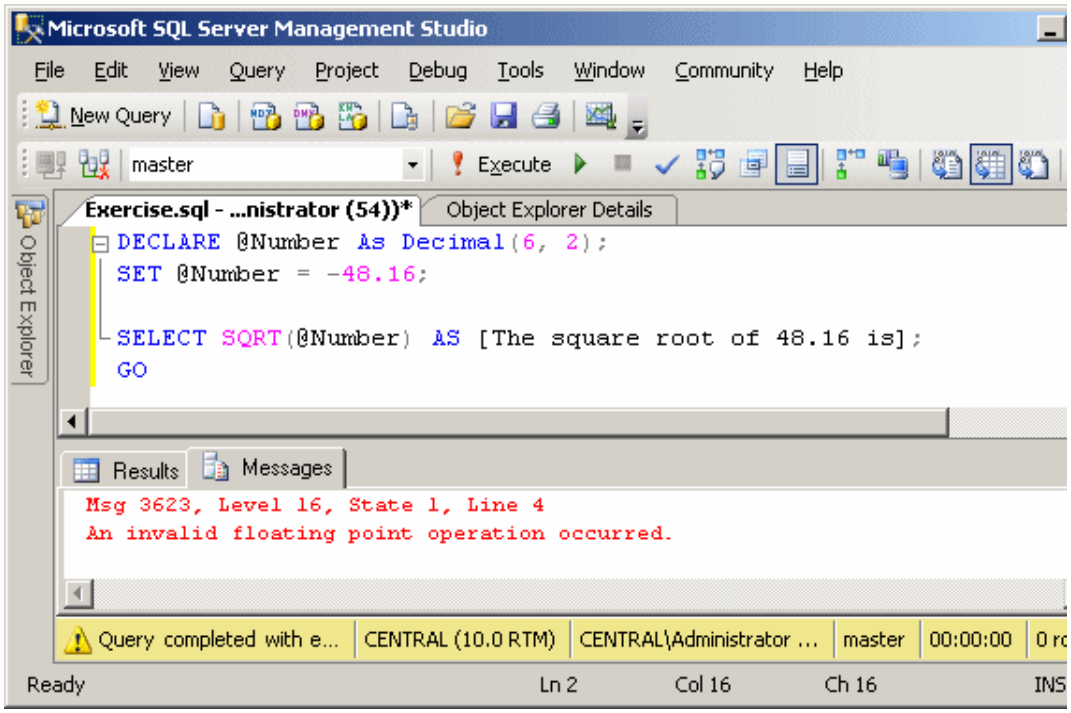
Esta función toma un argumento como un número decimal. Si el número es positivo, después de que devuelve la raíz cuadrada de x. Aquí está un ejemplo:

```
DECLARE @ Número Como decimales (6, 2);
SET @ Numero = 48,16;

SELECT SQRT (@ Numero) COMO [La raíz cuadrada es de 48,16];
IR
```

Esto produciría:



Si el número es negativo, usted recibirá un mensaje de error. Aquí está un ejemplo:



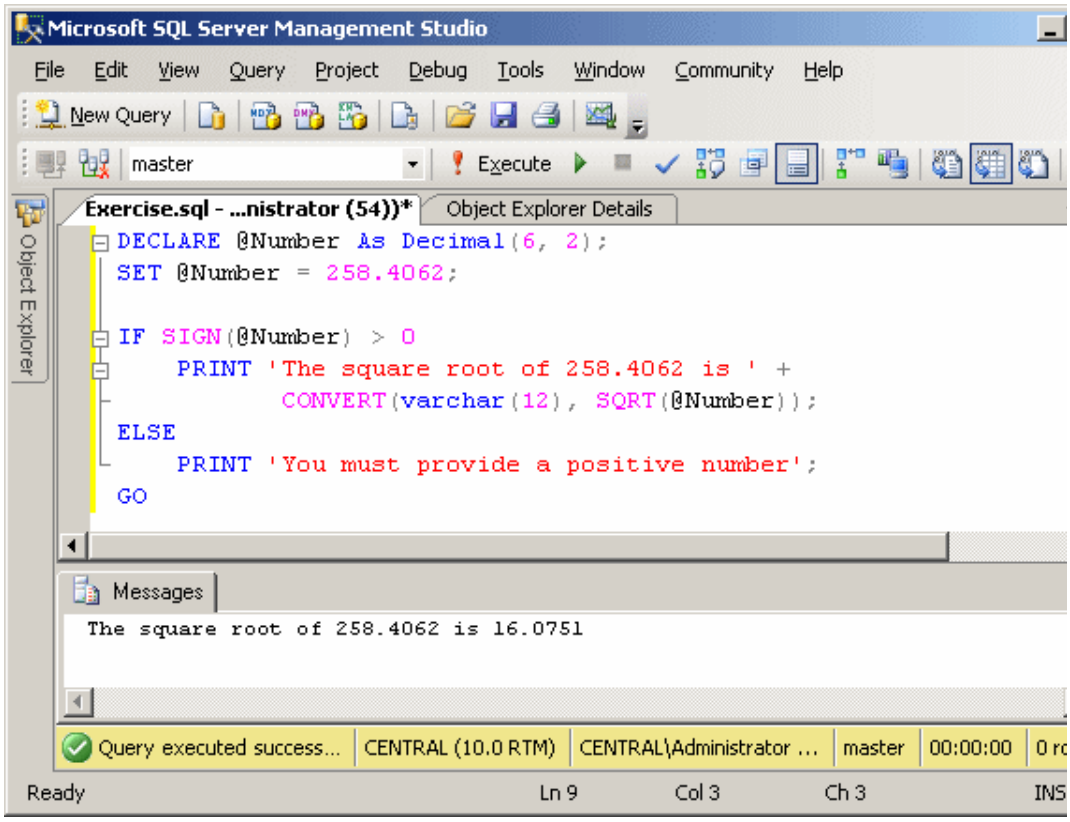
En este caso, puede utilizar una declaración de control para averiguar si la *expresión* es positiva. Aquí e

DECLARE @ Número Como decimales (6, 2);
SET @ Numero = 258.4062;

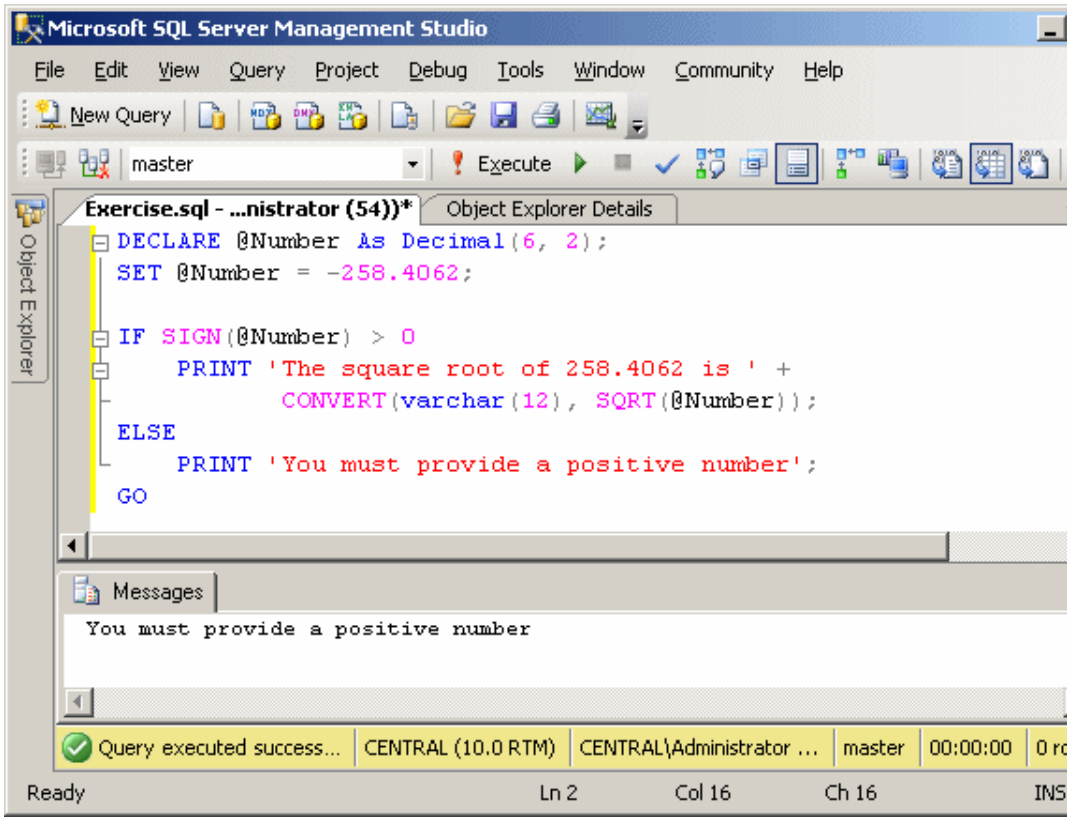
SI FIRME (@ Numero)> 0

```
        IMPRIMIR 'La raíz cuadrada de 258.4062 es' +
                CONVERT (varchar (12), SQRT (@ Numero));
ELSE
            IMPRIMIR 'Usted debe proporcionar un número positivo »;
IR
```

Aquí está un ejemplo de ejecución de la declaración:

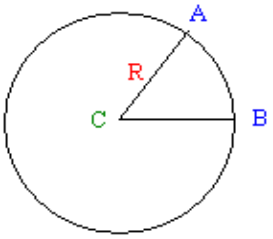


Aquí hay otro ejemplo de la ejecución de la declaración:



Medida basada en funciones

Introducción



Un círculo es una serie de distintos puntos opuestos, cada uno en una posición exactamente la misr punto a que se refiere como el centro. La distancia desde el centro C a uno de estos puntos equidistant R. La línea que conecta todos los puntos que son equidistantes del centro se llama la circunferencia del es la distancia entre dos puntos de la circunferencia al centro. En otras palabras, un diámetro es el dobl

Para la gestión de las mediciones y otras operaciones, la circunferencia se divide en 360 partes. porciones se llama grado. La unidad utilizada para representar el grado es el grado, escrito como °. P un círculo de 360 grados, que es de 360 °. La medición de dos puntos A y D de la circunferencia pued la circunferencia. En este caso, esta medida se representa el 15 °.

La distancia equidistante entre dos puntos A y B es una forma redonda geométricamente definida como es la relación entre la distancia entre dos puntos A y B de la circunferencia dividida por el radio R. Es como:

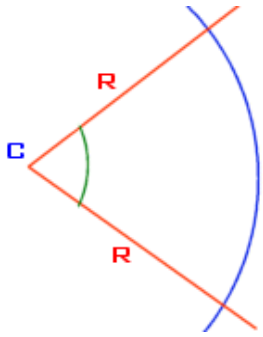
$$\theta = \frac{AB}{R}$$

PI

La carta π, también escrito como PI, es un número usado en diversos cálculos matemáticos. Su v
3.1415926535897932. La calculadora de Microsoft Windows le representa como 3,141592653589793
Para obtener el valor de la PI, de Transact-SQL ofrece la **PI ()** función. Su sintaxis es simple:

PI ()

Radianes



Un ángulo es la relación de un arco sobre el radio. Debido a un ángulo es una relación y no un "físico"
significa un punto de vista no es una dimensión, que es independiente del tamaño de un círculo. Obvi
vista representa el número de porciones cubiertas por tres puntos. Una mejor unidad utilizada para m
radián rad o.

Si usted conoce el valor de un ángulo en grados y que desea obtener el radianes, Transact-SQL propo
() función. Su sintaxis es:

RADIANS *(Expresión)*

Esta función toma como argumento un valor en grados. Si tiene éxito en su cálculo, se devuelve el valo

Un ciclo es una medición de la rotación alrededor del círculo. Dado que la rotación no es necesa
dependiendo de la situación, una medida que se haga sobre la base del ángulo que se cubrió durante l
podría cubrir una parte del círculo, en cuyo caso la rotación no se han completado. Un ciclo podría
totalidad del 360 ° del círculo y continuar allí después. Un ciclo es equivalente a la radián dividido por 2

Grados

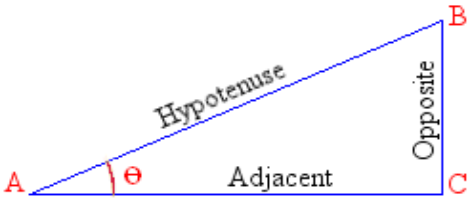
Si conoce el radianes, pero desea obtener los grados de un ángulo, puede utilizar el **GRADOS ()** función

GRADOS *(Expresión)*

Esta función toma como argumento un valor en radianes. Si tiene éxito, devuelve el valor equivalente e

Funciones trigonométricas

El coseno de un valor



AB considerar la longitud de la A a la B, también conocida como la hipotenusa. También se tendrá en c
la AC A a C, que es el lado adyacente al punto A. El coseno del ángulo en el punto A es la relación A
relación entre la longitud adyacentes, AC, a lo largo de la longitud de la hipotenusa, AB:

$$\cos\theta = \frac{AC}{AB} = \frac{\text{Adjacent}}{\text{Hypotenuse}}$$

El valor devuelto, la relación, es un número de doble precisión entre -1 y 1.

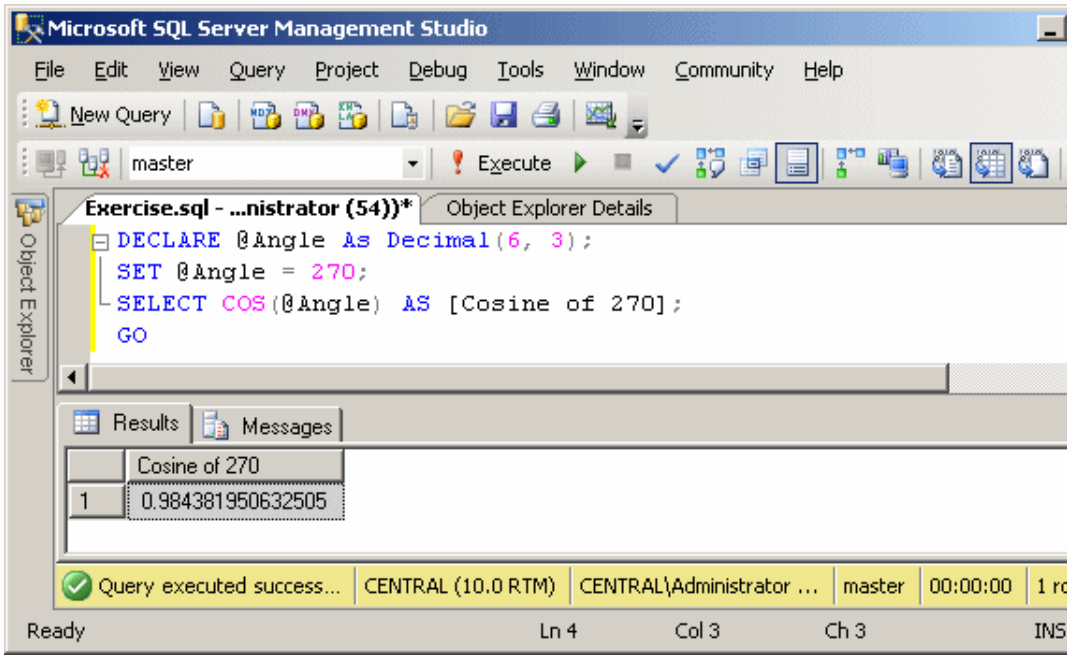
Para obtener el coseno de un ángulo, puede llamar al **COS ()** función. Su sintaxis es:

COS *(Expresión)*

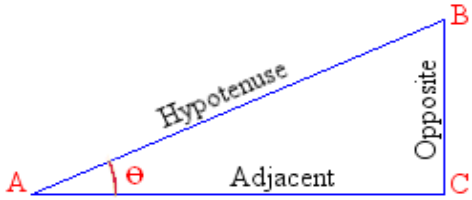
El ángulo que se considera es pasado como argumento para esta función. La función calcula y devue está un ejemplo:

```
DECLARE @ Ángulo Como decimal (6, 3);
SET @ Ángulo = 270;
SELECCIONE COS (@ Angulo) COMO [Coseno de 270];
IR
```

Esto produciría:



El Seno de un Valor



AB considerar la longitud de la A a la B, también llamado la hipotenusa al punto A. También se tendrá e de CB C a B, que es el lado opuesto al punto A. La condición representa la proporción de BC / AB, es de la cara opuesta, CB sobre la hipotenusa AB.

Para obtener el seno de un ángulo, puede utilizar el **SIN (),** cuya función es la sintaxis:

SIN *(Expresión)*

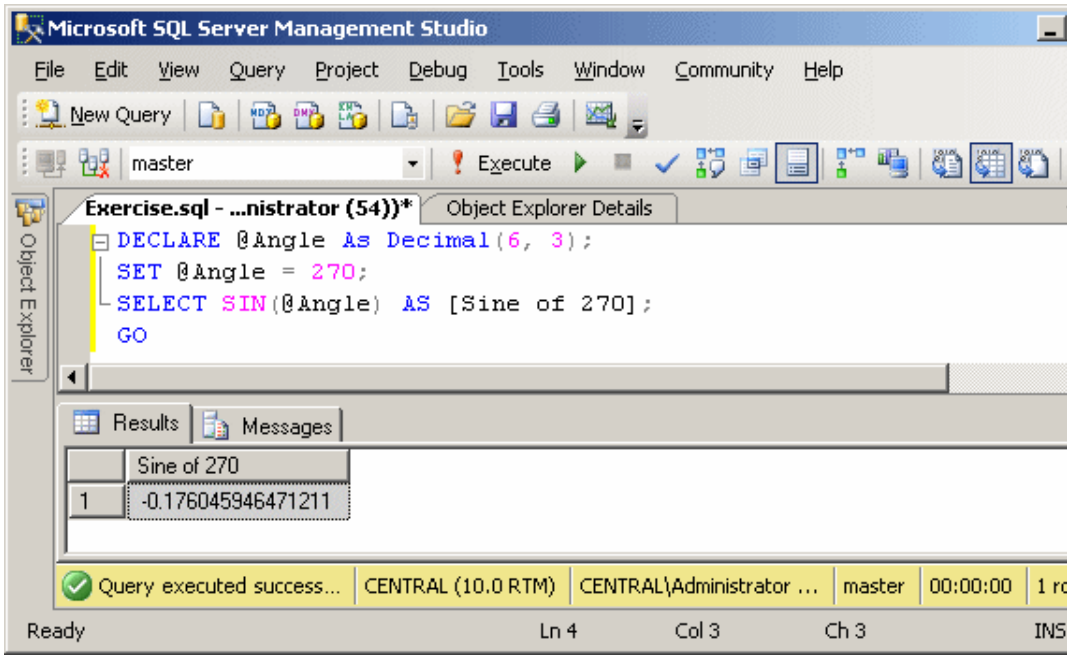
El ángulo que se considera es pasado como argumento. Después de su cálculo, la función devuelve

entre -1 y 1.

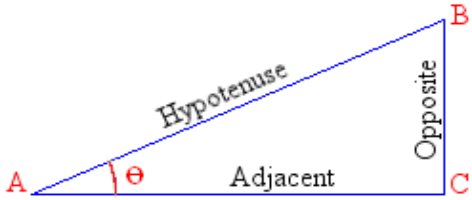
Aquí está un ejemplo:

```
DECLARE @ Ángulo Como decimal (6, 3);
SET @ Ángulo = 270;
SELECT SIN (@ Angulo) COMO [Seno de 270];
IR
```

Esto produciría:



La tangente de un valor



En geometría, AC considerar la longitud de la A a la C. aC También se tendrá en cuenta la longitud de B el resultado de AC / AC, es decir, la proporción de más de BC AC.

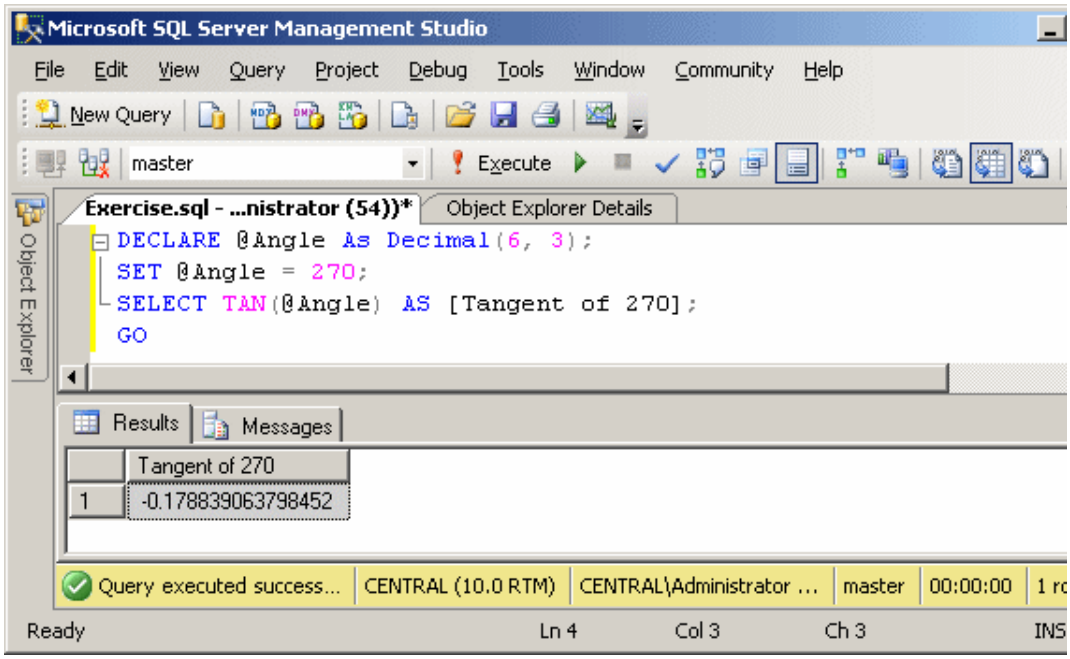
Para obtener la tangente de un ángulo, puede utilizar el **TAN ()** la función de Transact-SQL. Su sintaxis

TAN (Expresión)

Aquí está un ejemplo:

```
DECLARE @ Ángulo Como decimal (6, 3);
SET @ Ángulo = 270;
SELECCIONE TAN (@ Angulo) COMO [Tangente de 270];
IR
```

Esto produciría:



Fecha y Hora Funciones de la base

Introducción

Fecha y hora los valores son muy utilizados en aplicaciones de bases de datos. Se trata de las ventas, impuestos, las horas extraordinarias, etc Sobre la base de esta utilidad, sus operaciones cuentan cor bibliotecas que vas a utilizar en el desarrollo de su aplicación. Sin ser la más elaborada sobre esta cu SQL proporciona su propio nivel de apoyo a los valores de fecha y hora.

Antes de utilizar una fecha o un valor de tiempo en un cálculo, recuerde que primero debe obtener Puede definir una fecha o una constante de tiempo en su aplicación. Un ejemplo sería'1992 / 10/28 '. **DATETIME** o una variable y **SmallDateTime** inicializarla según sus necesidades. Usted puede obter hora de otra función. Como la última alternativa, usted puede obtener una fecha o la hora a partir de un usuario. Una vez que haya una fecha apropiada, puede utilizarlo.

El sistema actual de fecha y / o Tiempo

Una de las maneras en que puede ayudar al usuario con la fecha y la hora para obtener la fecha act Por ejemplo, si crea una hoja de tiempo, cuando el usuario comienza a utilizar, sería conveniente compl de tiempo con estos valores previsibles.

Para obtener la fecha y la hora actual de la computadora que el usuario está usando, puede usar el **g** de Transact-SQL. Su sintaxis es:

GETDATE ()

Esta función simplemente devuelve la fecha y hora actuales del sistema operativo.

Fecha / Hora adjunta

Una de las principales operaciones que se pretenda llevar a cabo en una fecha o una hora valor con valor a la misma. En apoyo de esta operación, de Transact-SQL proporciona la **DateAdd ()** función. Su

DateAdd *(TypeOfValue, ValueToAdd, DateOrTimeReferenced)*

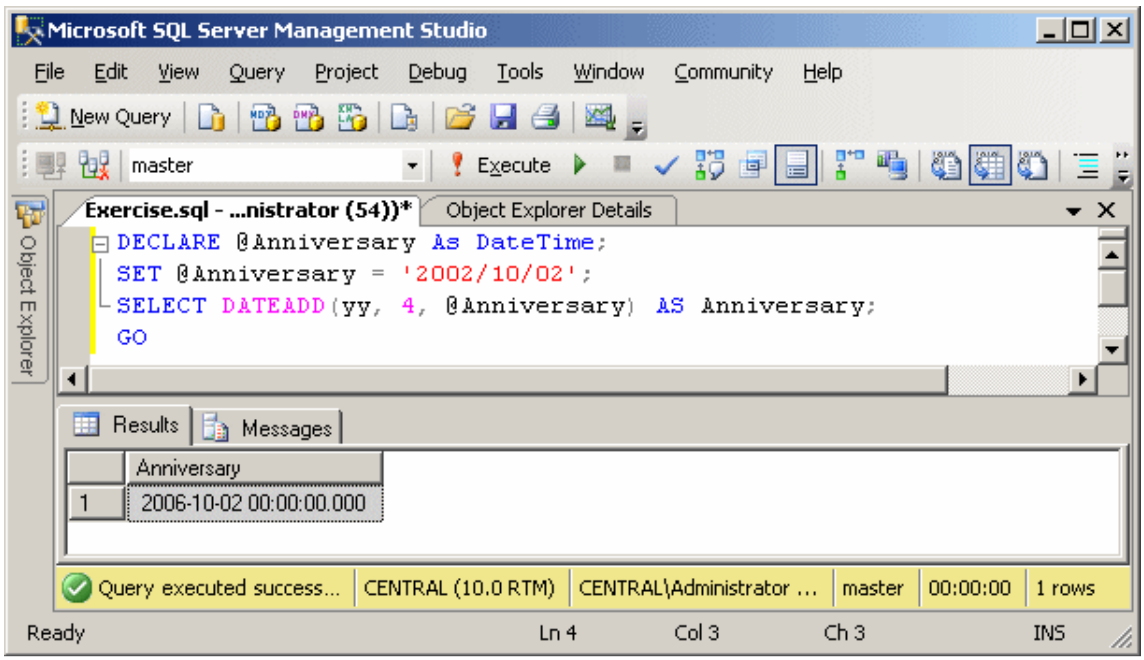
El tercer argumento de esta función es el valor de una fecha o un momento en que la operación se lle ser un valor constante en la forma de "año / mes / día" para una fecha o la hora: minutos am / pm 'dura

El segundo argumento es el valor que se añadirá. Debe ser una constante entero, como el 8, o un valo como la 4.06.

Al llamar esta función, primero debe especificar el tipo de valor que desee agregar. Este tipo se argumento. Se utiliza de la siguiente manera:

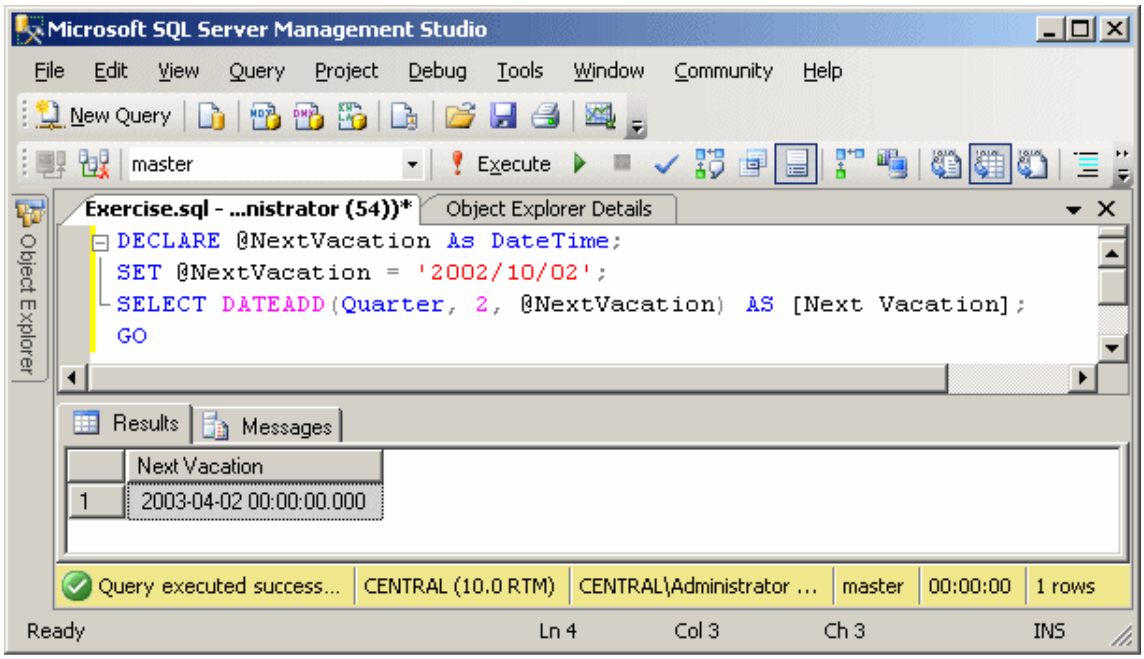
- Si desea añadir un número de años a una fecha, especifique el **año** o como *TypeOfValue* **aa** o **aaa** SQL es sensible a las mayúsculas). Aquí está un ejemplo:

```
DECLARE @ Aniversario Como datetime;
SET @ Aniversario ='2002 / 10/02 ';
SELECCIONE DateAdd (AA, 4, @ Aniversario) COMO Aniversario;
IR
```



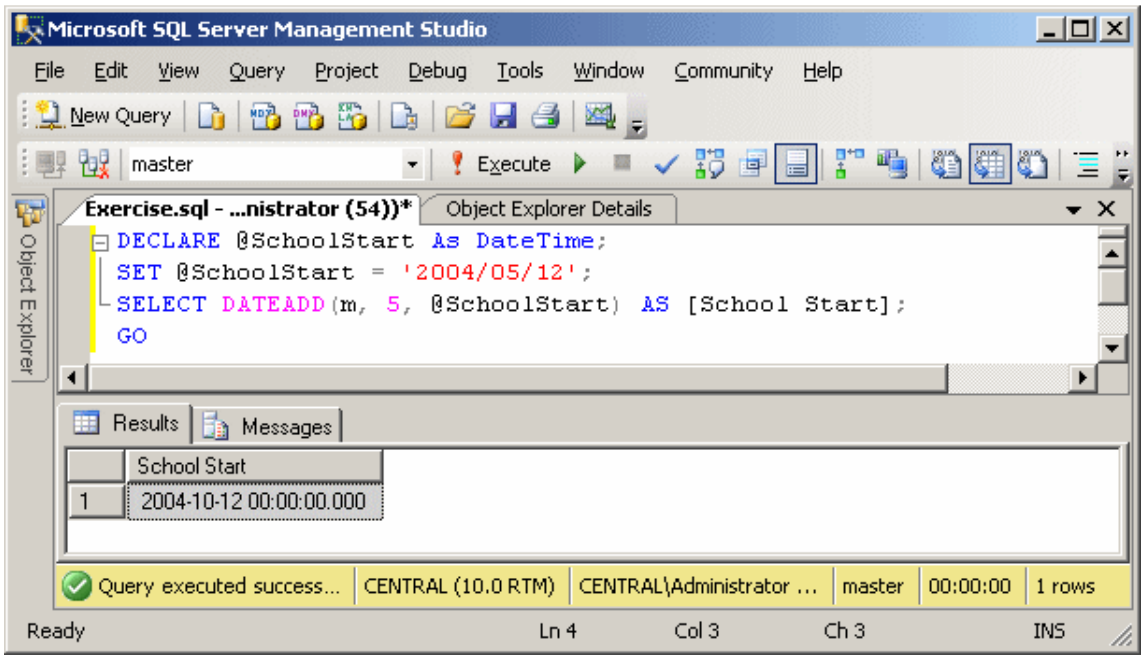
- Si desea añadir un número de cuartos de un año a una fecha, especifique el *TypeOfValue* como **Ba** aquí un ejemplo:

```
DECLARE @ NextVacation Como datetime;
SET @ NextVacation ='2002 / 10/02 ';
SELECCIONE DateAdd (Barrio, 2, @ NextVacation) COMO [Siguiente vacaciones];
IR
```



- Si desea añadir un número de meses a una fecha, como especificar el **mes** o *TypeOfValue* **m,** o **m** ejemplo añade 5 meses a la fecha:

```
DECLARE @ SchoolStart Como datetime;
SET @ SchoolStart ='2004 / 05/12 ';
SELECCIONE DateAdd (m, 5, @ SchoolStart) COMO [Escuela Inicio];
IR
```



De la misma manera, usted puede agregar los valores de la siguiente manera:

Tipo de Valor	Abreviatura	Como resultado
Año	aa	Un número de años que se añadirá a la fecha valor

	aaaa	
trimestre	q	Una serie de cuartos de un año se añadirá a la fecha valor
	qq	
Mes	m	Un número de meses se añadirán a la fecha valor
	mm	
dayofyear	y	Un número de días de un año se añadirá a la fecha valor
	dy	
Día	d	Un número de días que se añadirán a la fecha valor
	dd	
Semana	Semana	Un número de semanas se añadirá a la fecha valor
	peso húmedo	
Hora	hh	Un número de horas se añadirá a la vez el valor
minutos	n	Un número de minutos se añadirá al valor del tiempo
	mi	
segundo	s	Un número de segundos que se añadirán a la vez el valor
	ss	
milisegundo	ms	Un número de milisegundos se añadirá al valor del tiempo

Fecha / Hora Substracción

Otra operación regular realizado en una fecha o un valor de tiempo consiste en obtener el número transcurrido en el rango de dos fechas o dos valores de tiempo. En apoyo de esta operación, de Trans la **DATEDIFF ()** función. Su sintaxis es:

DATEDIFF *(TypeOfValue, StartDate, EndDate)*

Esta función tiene tres argumentos. El segundo argumento es la fecha o la hora de inicio del rango a c argumento es el final o la última fecha o la hora de la gama considerada. Puede utilizar el prim especificar el tipo de valor que desea que la función de producir. Este argumento utiliza el mismo **DateAdd ()** Función:

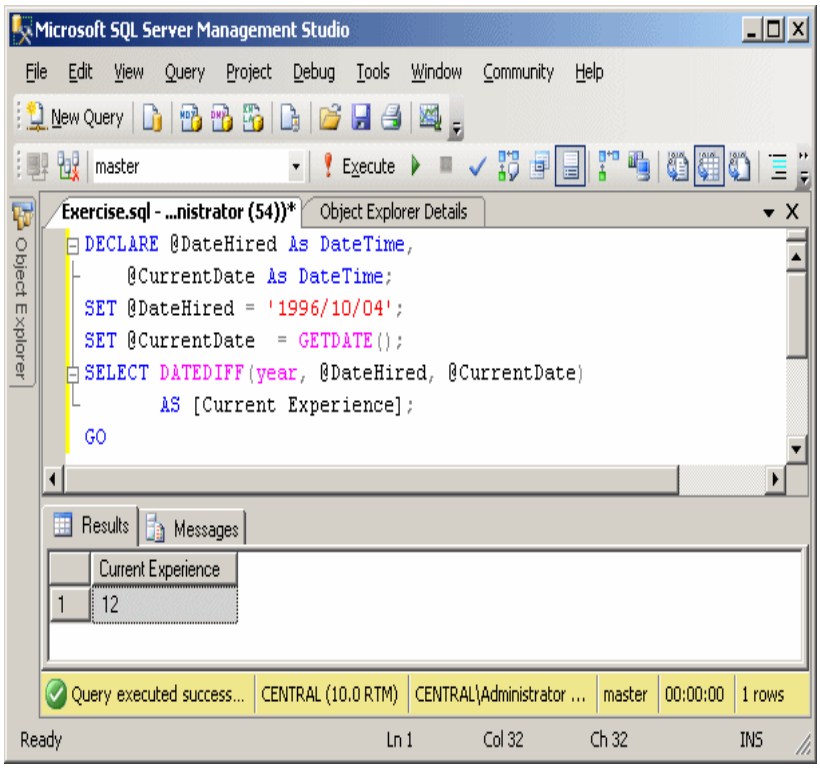
Tipo de Valor	Abreviatura	Como resultado
Año	aa	La función devolverá el número de años que han transcurrido entre el inicio y la fecha de finalización
	aaaa	
trimestre	q	La función devolverá el número de trimestres de un año q han transcurrido entre el inicio y la fecha de finalización
	qq	
Mes	m	La función devolverá el número de meses que han transcurrido entre el inicio y la fecha de finalización
	mm	
dayofyear	y	La función devolverá el número de días de un año que han transcurrido entre el inicio y la fecha de finalización
	dy	
Día	d	La función devolverá el número de días que han transcurri entre el inicio y la fecha de finalización
	dd	
Semana	Semana	La función devolverá el número de semanas que han transcurrido entre el inicio y la fecha de finalización
	peso húmedo	
Hora	hh	La función devolverá el número de horas que han

		transcurrido entre el inicio y el fin de los tiempos o fechas
minutos	n	La función devolverá el número de minutos que han transcurrido entre el inicio y el fin de los tiempos o fechas
	mi	
segundo	s	La función devolverá el número de segundos que han transcurrido entre el inicio y el fin de los tiempos o fechas
	ss	
milisegundo	ms	La función devolverá el número de milisegundos que han transcurrido entre el inicio y el fin de los tiempos o fechas

Aquí hay un ejemplo que calcula el número de años que uno ha estado con los trabajadores de la empre

```
DECLARE @ DateHired Como DATETIME,
          @ CurrentDate Como datetime;
SET @ DateHired = «1996 / 10/04 ';
SET @ CurrentDate = GETDATE ();
SELECCIONE DATEDIFF (año, DateHired @, @ CurrentDate)
        COMO [actual Experiencia];
IR
```

Esto produciría:



Aprendizaje práctico: Acabar con la Lección

1. Cierre la ventana de consulta sin guardar el archivo
2. En el Explorador de objetos, de conformidad con el nodo Bases de datos, haga clic derecho y haga Exercise1
3. En el cuadro de diálogo, haga clic en Aceptar

Resumen de la lección

Varios Ejercicios

1. Escribir una función llamada ProcessPayroll1 que tenga el número de horas trabajadas en una sem función devuelve un valor que representa las horas extraordinarias. Si el empleado trabajó menos horas extraordinarias. Si el empleado trabajó durante más de 40 horas, el número se considera má extraordinarias
2. Escribir una función llamada GetWeekdayName que, cuando se les da una fecha, puede encontrar de la semana para esa fecha
3. Escribir una función llamada GetNumberDays que lleva dos fechas y devuelve el número de días er
4. Escribir una función llamada AddNumberDays que tenga una fecha y un número entero y, a contin fecha agregó que el número

Introducción

Una tabla es principalmente una lista de temas o de un grupo de listas. Para la gestión de dicha lista, debe ser meticulosamente organizado. Para organizar esta información, se divide en secciones. Aquí está un ejemplo:

Nombre	Edad	Género	Relación
Judie	18	Hembra	Hermana
Ernest	24	Macho	Primo
Cuenta	52	Desconocido	Tío
David	36	Macho	Hermano
Hermine	12	Desconocido	Sobrina

Sobre esta base, una lista es simplemente una disposición de la información y esta información, también llamada de datos, se almacena en tablas.

Aprendizaje práctico: A partir de una base de datos

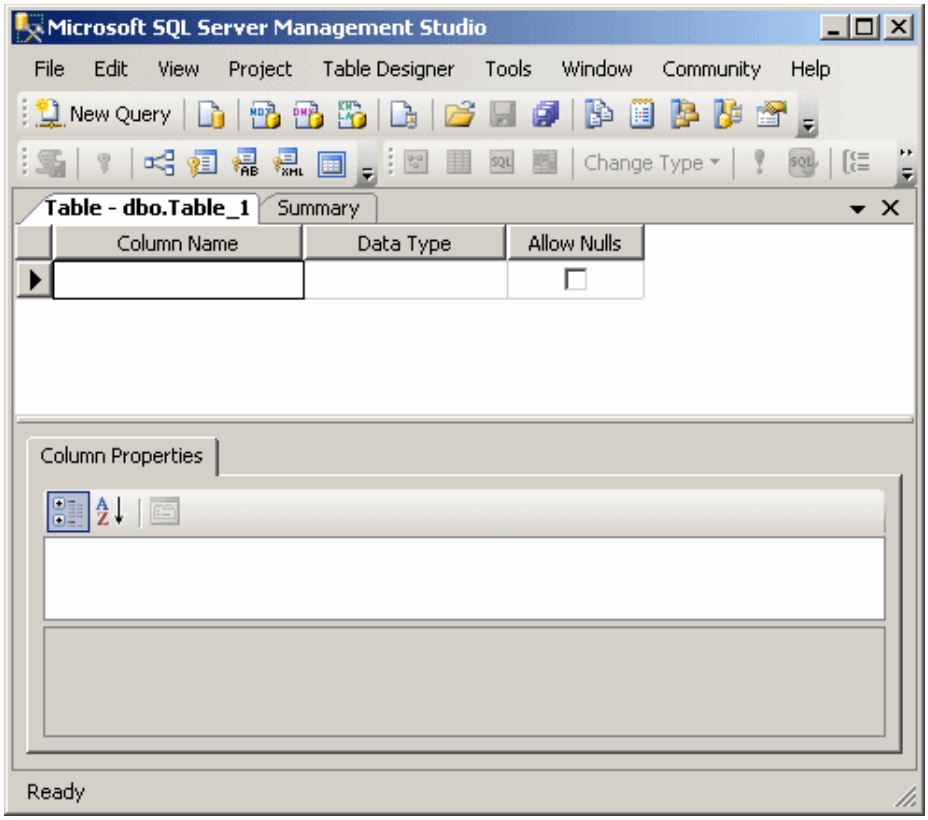
1. Inicie Microsoft SQL Server, conectar y, en el Explorador de objetos, expanda el nodo Bases de dat
2. En el Explorador de objetos, haga clic derecho en las bases de datos y haga clic en Nueva base de
3. En el cuadro de diálogo Nueva base de datos, establezca el nombre de base de datos de **BCR**
4. En virtud de Tamaño inicial (MB) de la columna, establecer el tamaño de archivos de los principale
5. Haga clic en Aceptar

Creación visual de un cuadro

La información de una tabla se organiza en categorías denominadas columnas y acuerdos horiz registros o filas. Una columna tiene una categoría de datos que es común a todos los registros. Una ta menos una columna. Esto significa que no se puede crear una tabla sin definir al menos una columna.

Aprendizaje práctico: Crear una tabla

1. En el Explorador de objetos, expanda el nodo BCR (haga clic en el botón +)
2. En virtud del BCR, haga clic derecho en los cuadros y haga clic en Nuevo Cuadro ...



3. Como el cursor parpadea en el primer campo vacío en la columna Nombre de columna, escriba **el n**
Intro

Nombres de los cuadros

Para completar la creación de una mesa, usted debe guardarlo. Si usted es recién la creación de una salvarlo, a usted se le pedirá que se nombre. El nombre de una tabla:

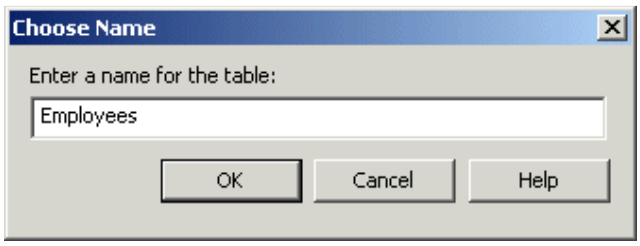
- Se puede hacer sólo de cifras. Por ejemplo, puede tener una tabla llamada 148
- Puede empezar con un dígito, una letra o un guión bajo
- Puede hacerse de las letras, números y espacios

Además de estas reglas, puede realizar hasta el suyo. Para evitar confusiones, he aquí las reglas qu nombrar nuestras mesas:

- Un nombre se iniciará con una carta. Ejemplos de ello son **actos** o **Segunda**
- Después del primer carácter de subrayado o una carta, el nombre tendrá combinaciones de relieve Ejemplos de ello son **_n24, act_52_t**
- A menos que se indique otra cosa, un nombre no incluir caracteres especiales como!, @, #, $,%, ^
- Si el nombre es una combinación de palabras, cada palabra comenzará en mayúsculas. Ejemplos **miembros del personal** o **los títulos de vídeo**

Aprendizaje práctico: una tabla de nombres

1. Para guardar la tabla, en la barra de herramientas Estándar, haga clic en el botón Guardar



2. Seleccione en el cuadro de diálogo Nombre, escriba **los empleados** y pulse Enter
3. Después de guardar la tabla, cerrarla haciendo clic en su botón Cerrar sistema

Con la creación de una tabla de SQL

Introducción

En SQL, para crear una tabla, se inicia con la siguiente declaración:

CREATE TABLE *TableName;*

La expresión **de CREATE TABLE** es obligatorio. El factor de *TableName* especifica el nombre de la puede utilizar el *TableName* **normas y sugerencias** que hemos examinado para las tablas.

Después de especificar el nombre de la tabla, debe crear por lo menos una categoría, llamada la columr

Uso de Código de muestra

Para ayudarle con la creación de una tabla, Microsoft SQL Server puede generar código de ejempl continuación, puede simplemente modificar o personalizar la misma. En primer lugar mostrar o abrir consulta vacía. Para mostrar el Explorador de plantillas, en el menú principal, puede hacer clic en Ver plantillas. En el Explorador de plantillas, expanda el nodo de la tabla. En virtud de la tabla, arrastre suéltelo en la ventana de consulta. Código de ejemplo se generó para usted.

Mantenimiento de tablas

Introducción

Cuadro de mantenimiento consiste en revisar o cambiar sus aspectos. Esto incluye la revisión de la una base de datos, cambiar el nombre de una tabla, o borrarlo.

Visualización de las propiedades de una tabla

Como cualquier otro objeto de una base de datos o de la computadora, una mesa posee algunas caract propias de ella. Para ver estas características, en el Explorador de objetos, haga clic derecho en la tab Propiedades.

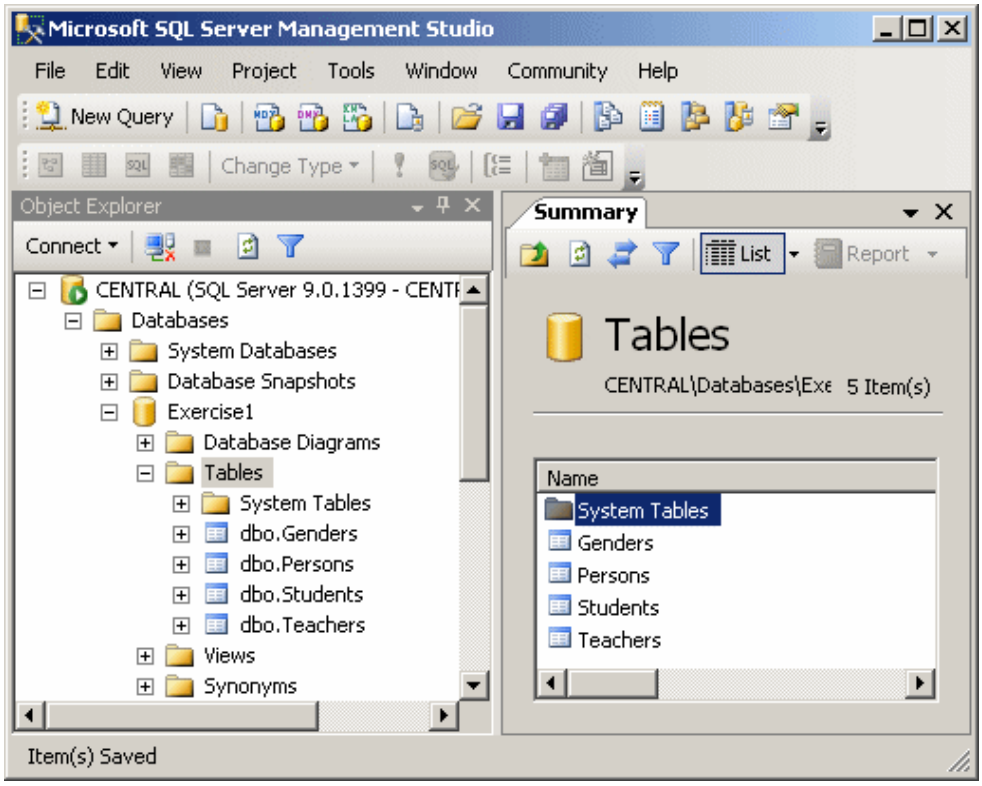
La apertura de una tabla

La mayoría de las operaciones requieren que se abra una mesa antes de usarlo. Hay varias maneras m dependiendo de la forma en que desea estudiar es el siguiente:

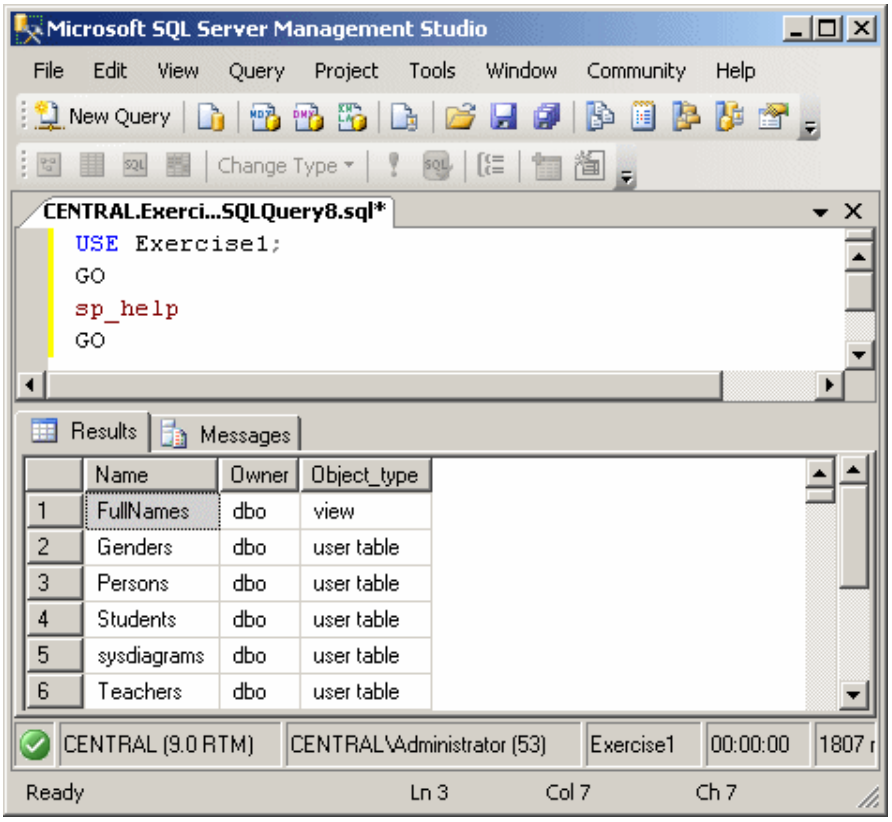
- Para ver la estructura de una tabla, tal vez a cambio de sus columnas, en el Explorador de objetos de datos y su nodo Tablas. Haga clic derecho en el cuadro y haga clic en Modificar. El cuadro se ab Diseño, el mismo punto de vista visual que se utiliza para crear una tabla.
- Si desea ver el código SQL de una tabla, en el Explorador de objetos, haga clic derecho en el cuadr ratón sobre el cuadro de comandos de AS, A CREAR, y haga clic en Nueva ventana del editor de Co
- Para abrir una tabla para ver sus datos, tal vez para realizar la entrada de datos, en el Explorador clic derecho en la tabla y haga clic en Abrir tabla

Mesas de examen

Para ver la lista de tablas de una base de datos en el Explorador de objetos, puede hacer clic en el nodo



Para ver la lista de tablas de una base de datos con SQL, en una ventana de consulta, especifique (usando una declaración **USO),** y ejecutar **sp_help** (es un procedimiento almacenado). Aquí está un ej



Cambiar el nombre de una tabla

Si encuentra que el nombre de una tabla no es apropiada, se puede cambiar. Para cambiar el nombre SQL Server Management Studio, en el Explorador de objetos, haga clic derecho en la tabla y haga nombre. Escriba el nombre deseado y pulse Enter.

Para cambiar el nombre de una tabla con el código, ejecute **sp_rename,** seguido por el nombre actua coma, y el nuevo nombre de la tabla. La fórmula a utilizar es:

sp_rename *ExistingTableName, TableNewName;*

Los nombres de las tablas deben incluirse en una sola cita. Aquí está un ejemplo:

sp_rename 'StaffMembers "," Empleados ";
IR

En este caso, el intérprete busca una tabla denominada StaffMembers o en el actual base de datos se encuentra, que le cambie el nombre de los empleados. Si la tabla no existe, usted recibirá un mensaje c

Eliminación de un cuadro

Si usted tiene un mal cuadro en una base de datos, puede eliminarla. Para eliminar una tabla Management Studio, en el Explorador de objetos, haga clic derecho en la tabla de la base de datos de su en Eliminar. Usted recibirá un aviso dándole la oportunidad de confirmar sus intenciones. Si todavía c tabla, haga clic en Aceptar.

Para eliminar una tabla usando SQL, utilice la siguiente fórmula:

DROP TABLE *TableName*

DROP TABLE La expresión que se requiere y es seguido por el nombre de la tabla indeseable.

declaración, no recibirá una advertencia antes de que el cuadro se suprime.

También puede usar código de ejemplo de Microsoft SQL Server que puede generar para usted. En prir una ventana de consulta vacía. También mostrar el Explorador de plantillas y ampliar la tabla nodo. cuadro de caída arrastre y suéltelo en la ventana de consulta vacía. Código de ejemplo se genere continuación, puede simplemente modificar y ejecutar la sentencia.

Refiriéndose a una tabla

En el futuro la experiencia, vamos a escribir diversas expresiones que incluyen los nombres de expresiones, es necesario especificar una tabla que desea utilizar. Existen tres maneras de hacer referencia a, o para indicar, una tabla:

- Usted puede simplemente escriba su nombre. Un ejemplo podría ser **estudiantes**
- Puede escribir **dbo,** seguido por el operador período, seguido por el nombre de la tabla. Un ejemp **dbo.Students**
- Puede escribir el nombre de la base de datos a la que pertenece la tabla, seguido por el operador p por **dbo,** seguido por el período comprendido entre el operador, y seguido por el nombre de la tabla sería **RedOakHighSchool.dbo.Students**

Las columnas de una tabla

Introducción

En nuestra introducción a los cuadros, vimos que la lista podría ser organizado en categorías denom Aquí está el ejemplo que vimos:

Nombre	Edad	Género	Relación
Judie	18	Hembra	Hermana
Ernest	24	Macho	Primo
Cuenta	52	Desconocido	Tío
David	36	Macho	Hermano
Hermine	12	Desconocido	Sobrina

Como puedes ver en este acuerdo, una columna se utiliza para clasificar un tipo particular de datos. columna se puede utilizar a la lista algunos nombres. Otra columna se puede utilizar para los núme embargo, otra columna se puede utilizar para seleccionar una lista de los temas que mantienen a la r temas.

Para organizar la información que tiene una columna, un cuadro que necesita una serie de detalles columna. Dos aspectos son particularmente importantes: el nombre y el tipo de datos que una co pueden sostener.

El nombre de una columna

Para ser capaz de reconocer las categorías de información que mantiene una columna, la column nombre. En Microsoft SQL Server, el nombre de una columna aparece en la parte superior, la cabecer El nombre de una columna permite que la base de datos como un archivo para identificar la columna. columna también ayudará a usted, desarrollador de la base de datos, para identificar a esa columna. sugerencias que usted debe o deben seguir al nombrar las columnas de una tabla.

El nombre de una columna:

- Puede empezar con una letra, un dígito o un guión bajo

- Puede incluir letras, dígitos, y los espacios en cualquier combinación

Después de respetar estas normas, puede agregar sus propias normas. En nuestra experiencia, aqu vamos a utilizar nuestro nombre a las columnas:

- Un nombre se iniciará con una carta. Ejemplos de ello son **n, actuar,** o **segundo**
- Después del primer carácter de subrayado o una carta, el nombre tendrá combinaciones de relieve Ejemplos de ello son **n24** o **col_52_t**
- A menos que se especifique otra cosa, un nombre no incluir caracteres especiales como!, @, #, $,
- Si el nombre es una combinación de palabras, cada palabra comenzará en mayúsculas. Ejemplos **Fecha Ocupado, Apellido, número de licencia de conducir,** o **dirección de E-mail**

Aprendizaje práctico: Configuración de las Columnas de Nombres

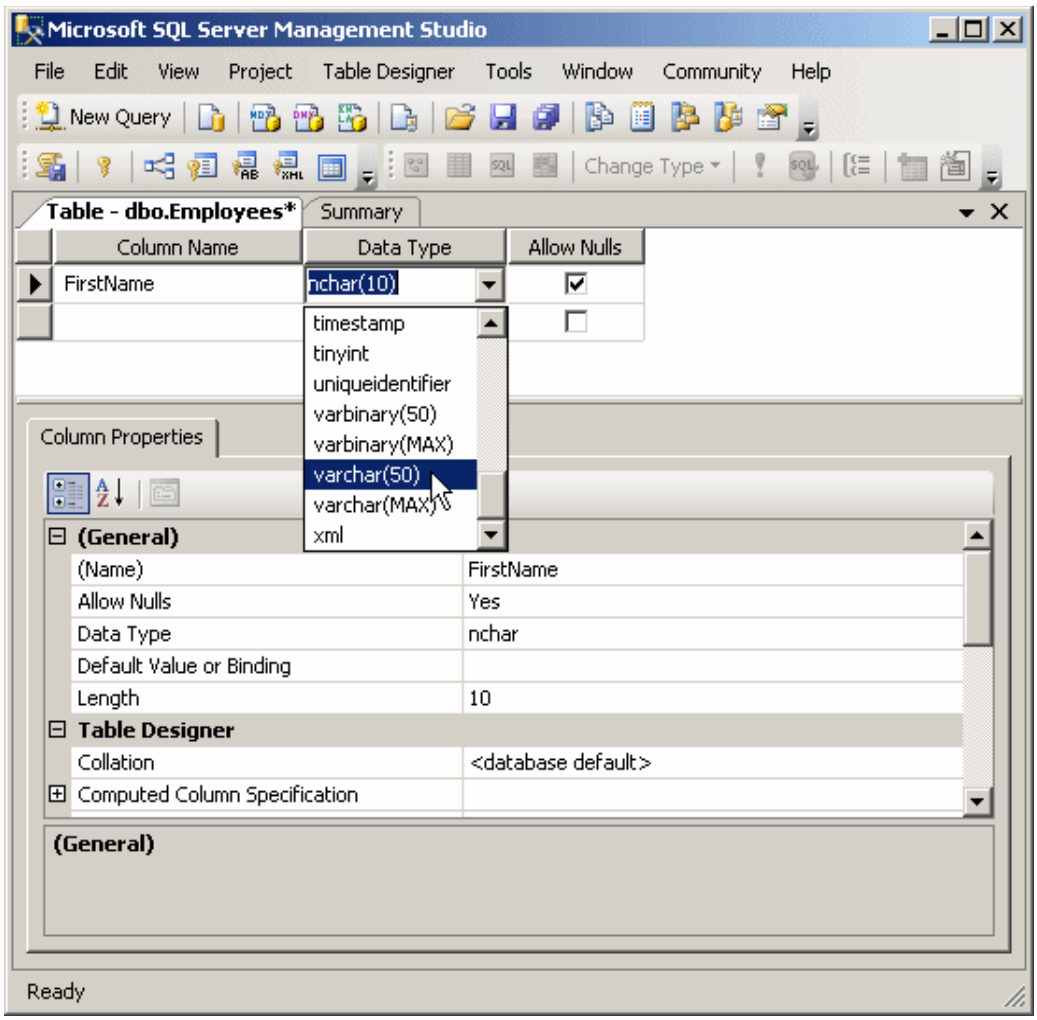
1. Bajo la columna Nombre de columna, haga doble clic en el nombre para resaltarlo
2. **Nombre** Tipo de sustituir y pulse Intro

Los tipos de datos

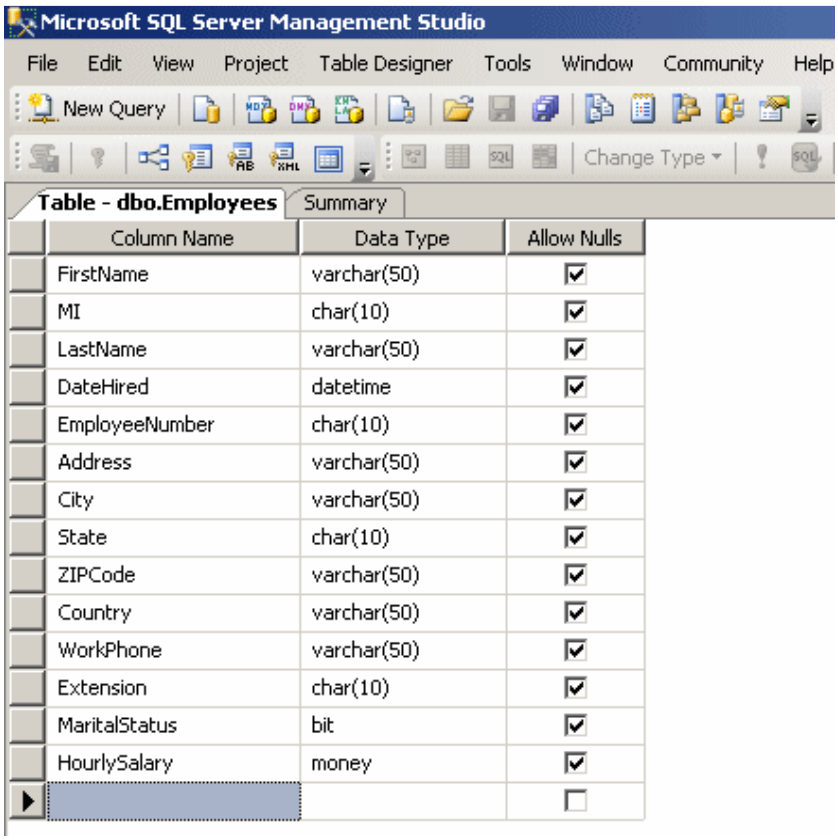
Después de decidir sobre el nombre de una columna, la base de datos necesita saber qué tipo de i columna se mantenga. Puesto que hay varios tipos de información de una base de datos puede trata Lección 3 tipos de datos que Microsoft SQL Server apoyo. Por lo tanto, debe especificar el tipo de datos para una columna.

Aprendizaje práctico: Configuración de Tipos de datos

1. Haga clic en la flecha del cuadro combinado bajo la columna Tipo de datos
2. Desplácese hacia abajo y seleccione de la lista **varchar**



3. Haga clic en el primer campo vacío bajo Nombre y tipo **MI**
4. Pulse la tecla de flecha abajo para posicionar el cursor bajo **MI**
5. **Apellido** Tipo y pulse la tecla de flecha abajo
6. Tipo **DateHired**
7. Ficha de prensa y el tipo **d**
8. Observe que el tipo de datos **datetime** está seleccionado.
9. Pulse Aceptar tres veces para colocar el cursor del ratón bajo **DateHired**
10. Tipo **EmployeeNumber** y pulse la tecla de flecha abajo
11. Completa el cuadro de la siguiente manera:



Column Name	Data Type	Allow Nulls
FirstName	varchar(50)	☑
MI	char(10)	☑
LastName	varchar(50)	☑
DateHired	datetime	☑
EmployeeNumber	char(10)	☑
Address	varchar(50)	☑
City	varchar(50)	☑
State	char(10)	☑
ZIPCode	varchar(50)	☑
Country	varchar(50)	☑
WorkPhone	varchar(50)	☑
Extension	char(10)	☑
MaritalStatus	bit	☑
HourlySalary	money	☑
		☐

12. Guardar la tabla

La Longitud de datos

Una base de datos se refiere a diversos tipos de datos adecuados o no para determinados ámbitos. debe tener cuidado de los puestos de trabajo detrás de las escenas lo más que pueda. Una forma que u esto es mediante el control de la cantidad de información que puede almacenarse en un campo determ columnas puede contener diferentes tipos de datos, por lo que puede el mismo tipo de datos de con mecanismo interno de entrada de datos. La duración de los datos significa diferentes cosas para di Columnas que llevan el mismo tipo de datos pueden tener diferentes longitudes.

Campos bits: Vimos un poco ya que la columna tipo se entiende por una de estas dos respues simplemente se supone que la base de datos permiten saber que la respuesta es sí o no, verdadero o f apagado, 1 o 0. Por lo tanto, sólo la longitud de este campo es 1.

Enteros: La longitud de un entero es el número de bytes de su campo puede contener. Para un tipo **in** bytes.

Decimales y números de punto flotante: La longitud especifica cuántos bytes puede almacenar el ca

Cuerdas: la longitud de una columna o una cadena de caracteres especifica el número máximo de campo puede contener.

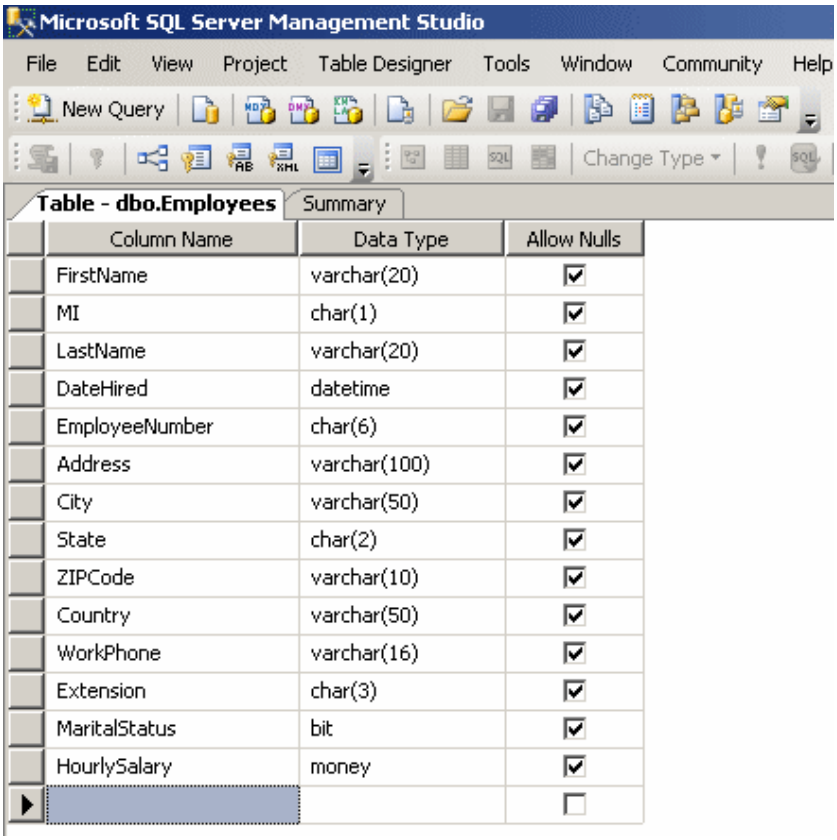
En algunas circunstancias, usted tendrá que cambiar o la duración que se aplica a un campo particular. que debe usar el tipo de datos **varchar** para una cadena de campo cuyo contenido va a cambiar de un r todas las columnas **varchar** deben tener la misma longitud. Aunque el primer nombre y el título de u columnas **varchar** tipo, las dos columnas que no tienen la misma longitud de entradas. Si llega el caso tiene un nombre que está más allá de 20 caracteres, y muchos títulos de libros van más allá de 32 car caso, ambos campos se utiliza el mismo tipo de datos, pero diferentes longitudes. Por otra parte, p **datetime** de **dinero** y tipos de datos, usted debe aceptar la propuesta por defecto de longitud de la bas

Hay dos maneras que usted puede cambiar la longitud de una cadena a base de la columna:

- En la sección superior de la ventana, para cambiar la longitud del campo, en el paréntesis del tipo introduzca el valor deseado
- En la sección superior de la ventana, haga clic en el nombre de la columna. En la sección inferior, campo de longitud y el tipo el valor deseado

Aprendizaje práctico: Configuración de Tipos de datos

1. En la sección superior, haga clic para seleccionarlo EmployeeNumber
2. En la sección inferior, haga clic en Longitud y tipo **6**
3. En la sección superior de la tabla, haga clic en Dirección y pulse Tabulador
4. Por el tipo de datos, tipo **VARCHAR (100)**
5. De la misma manera, completar el cuadro de la siguiente manera:



Column Name	Data Type	Allow Nulls
FirstName	varchar(20)	☑
MI	char(1)	☑
LastName	varchar(20)	☑
DateHired	datetime	☑
EmployeeNumber	char(6)	☑
Address	varchar(100)	☑
City	varchar(50)	☑
State	char(2)	☑
ZIPCode	varchar(10)	☑
Country	varchar(50)	☑
WorkPhone	varchar(16)	☑
Extension	char(3)	☑
MaritalStatus	bit	☑
HourlySalary	money	☑
		☐

6. Guardar la tabla

Creación programática de las Columnas

Hemos visto que la principal fórmula para crear una tabla fue la siguiente:

CREATE TABLE *TableName*

Después de especificar el nombre de la tabla, debe enumerar las columnas de la tabla. La lista de c con una apertura de paréntesis "(". La lista termina con un cierre de paréntesis ")". Cada columna deb del siguiente por una coma, excepto la última columna. Puede incluir todas las columnas en la misma lín de la siguiente manera:

CREATE TABLE País *(column1, column2, column3)*

Alternativamente, para hacer su declaración sea más fácil de leer, debe crear cada columna en una líne
manera:

CREATE TABLE País (
Column1,
Column2,
Column3);

Hay dos piezas de información que debe especificar para cada columna: su nombre y su tipo. Por lo tan
la creación de una columna es la siguiente:

Opciones ColumnName DataType

El nombre de una columna, debería seguir las mismas **reglas y sugerencias** que hemos examinado pa

Después de escribir el nombre de la columna, escriba el tipo de datos o apropiado para la columna. P
utilice uno de los (el caso) los tipos de datos que hemos examinado.

Recuerde que algunos de los tipos de datos deben tener una longitud. Esto es cierto para todas las ca
basado en las columnas **(char,** el **texto, varchar,** etc.) En el caso de los basados en texto columnas,
SQL para crear sus columnas, porque es menos visuales que el diseño de la tabla de SQL Server Manag
se puede confiar en la longitud de las cadenas por defecto propuestos por SQL (de hecho, en MySQL,
una longitud de **varchar).** En este caso, el SQL Server Management Studion especifica diferentes va
para el texto a base de columnas. Por lo tanto, cuando se utiliza SQL para crear sus columnas, u
especificar su duración predeterminada para el texto a base de columnas.

También vieron que se puede usar código de ejemplo para crear una tabla. Esto le permite tener más
distintas columnas de la tabla que desea tener. Para ello, abra una ventana de consulta vacía y mostrar
plantillas. Expanda el nodo de la tabla. En el cuadro, puede arrastrar el cuadro Crear, Añadir columna
gota, y colóquelo en la ventana de consulta. Si utiliza caído Añadir columna o la columna de caída, pued
de las secciones del código y aislar solamente la parte que se ocupa de la creación de la tabla. Aquí está

```
--==================================================
==========================
- Añadir columna plantilla
--
- Esta plantilla crea una tabla y, a continuación, se añade una nueva columna a la tabla.
--==================================================
==========================
USO <database, sysname, AdventureWorks>
IR

CREATE TABLE <schema_name, sysname, dbo>. <table_name, Sysname, sample_table>
(
        column1 int,
        column2 char (10)
),
IR
```

Aprendizaje práctico: Crear una tabla

1. En el Explorador de objetos, haga clic con el botón BCR y haga clic en Nueva consulta
2. En el editor de código, escriba lo siguiente:

```
CREATE TABLE Clientes (
DrvLicNbr VARCHAR (50),
DateIssued DATETIME,
DateExpired DATETIME,
FullName varchar (120),
```

Dirección VARCHAR (120),
Ciudad varchar (50),
Estado varchar (100),
PostalCode varchar (20),
HomePhone varchar (20),
OrganDonor bits)
IR

3. Para ejecutar la declaración, presione F5
4. Cierre la ventana Analizador de consultas SQL
5. Cuando se le preguntó si desea guardar el texto, haga clic en Sí.
6. Tipo de **clientes** como el nombre del archivo y pulse Intro

En referencia a una columna

Introducción

Vamos a escribir muchas expresiones que incluyen los nombres de columnas. En dichas expresiones, las columnas que usted se refiere. Hay varias maneras de hacerlo. Para referirse a, o para indicar, una

- Usted debe escribir el nombre de la tabla a la que pertenece la columna, seguido por el operador p por el nombre de la columna. Un ejemplo sería **Employee.LastName**
- Puede escribir **dbo,** seguido por el operador período, seguido por el nombre de la tabla a la que pe columna, seguido por el operador período, seguido por el nombre de la columna. Un ejemplo sería **dbo.Employee.LastName**
- Puede escribir el nombre de la base de datos que posee la mesa de la columna, seguido por el ope seguido por **dbo,** seguido por el operador período, seguido por el nombre de la tabla a la que perte seguido por el período comprendido entre el operador, seguido por el nombre de la columna. Un ej **RedOakHighSchool.dbo.Employee.LastName**

Uso del nombre de alias de una tabla

Puede crear un nombre de alias de una tabla para utilizar en una expresión que implica una columna. una letra o una palabra que representa la tabla a la que pertenece la columna. La letra o la palabra e período de operador, y seguido por el nombre de la columna. Un ejemplo sería **empl.LastNam** declaración, usted debe escribir el nombre de la tabla, seguido por un espacio, y seguido de la letra ejemplo podría ser **empleado empl**

.

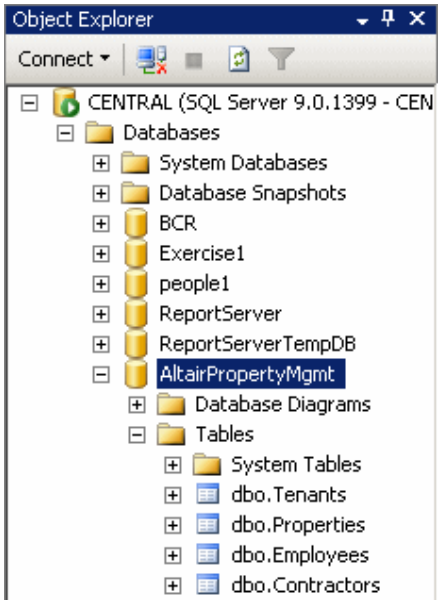
Columnas de mantenimiento

Introducción

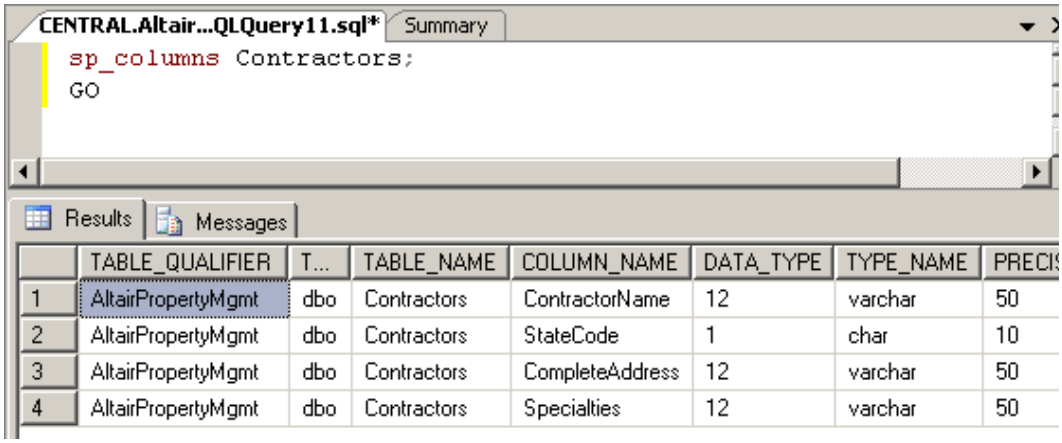
Columna de mantenimiento consiste en revisar o cambiar cualquiera de sus aspectos. Esto incluye estructura de columnas de una tabla, cambiar el nombre de una columna, la supresión de una colum tipo de datos o la nulidad de una columna, etc

Examen de la columna

Para ver la estructura de una tabla en SQL Server Management Studio, en el Explorador de objetos, pue



Para ver las columnas de una tabla usando código SQL, en una ventana de consulta, ejecute **sp_colu** el nombre de la tabla corresponden a las columnas. Aquí está un ejemplo:



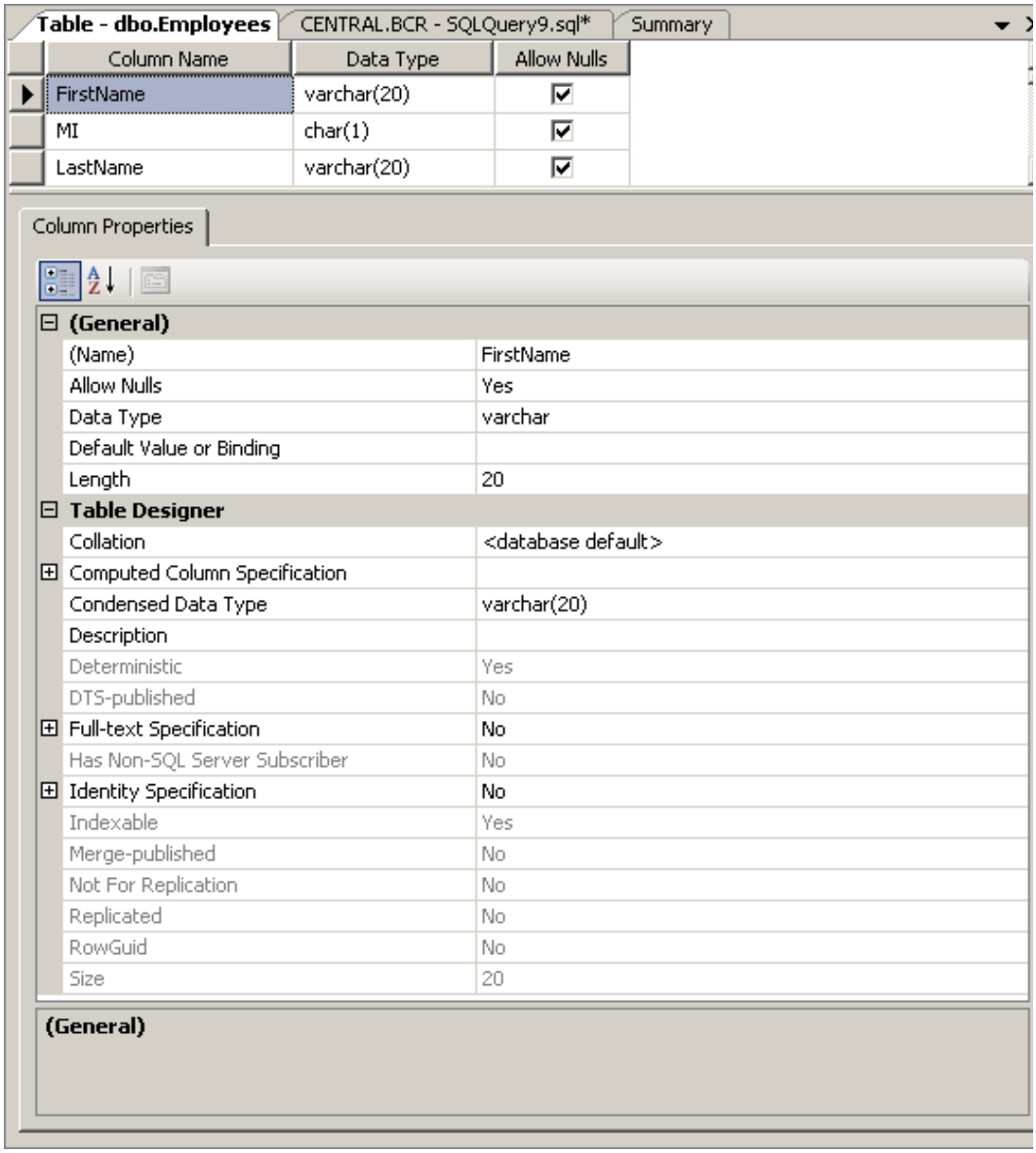
	TABLE_QUALIFIER	T...	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECIS
1	AltairPropertyMgmt	dbo	Contractors	ContractorName	12	varchar	50
2	AltairPropertyMgmt	dbo	Contractors	StateCode	1	char	10
3	AltairPropertyMgmt	dbo	Contractors	CompleteAddress	12	varchar	50
4	AltairPropertyMgmt	dbo	Contractors	Specialties	12	varchar	50

Esta acción muestra la lista de las columnas de la columna COLUMN_NAME y otras característica columnas.

Las propiedades de una columna

Una columna de una tabla de control de qué tipo de datos es apropiada para esa columna. Las ca identificar o describir un cuadro se definen como sus propiedades. Como hemos visto anterior propiedades son particularmente importantes y necesarias para cada columna: el nombre, el tipo de dat Además de estas, otras propiedades pueden ser usados para seguir controlando el comportamiento de campo.

Además del nombre, tipo de datos y la longitud de una columna, usted puede controlar las colum usando la hoja de propiedades de **las columnas** en la parte inferior de la tabla en la vista Diseño. Est veces depende de el tipo de datos de la columna. Por lo tanto, para especificar las propiedades de una debe seleccionarlo en la parte superior de la tabla. Esta selección se puede hacer con sólo hacer clic tipo de datos, o la longitud de la columna. Luego, puedes presionar la tecla F6 o haga clic en el primer inferior, seleccione el tipo de propiedad y el valor necesario:



Descripción

Descripción: común y habilitada para todos los campos, la descripción se utiliza una frase que des
Puede escribir cualquier cosa en ese campo.

Cotejo

Puesto que los diferentes lenguajes que utilizan diferentes mecanismos en sus caracteres alfabéticos, e
la manera en algún tipo de algoritmos o consultas se realizan en los datos, usted puede pedir la ba
aplicar un determinado lenguaje mecanismo sobre el terreno mediante el cambio de la **colación de**
contrario, deberá aceptar el valor por defecto especificado en la tabla.

Para especificar la colación de una columna en la hora de crear, tipo **COLLATE,** seguido por el códig
Aquí está un ejemplo:

```
CREATE TABLE Clientes (
	FullName varchar (50) COLLATE SQL_Latin1_General_CP1_CI_AS
);
```

Modificación de una columna

Al realizar un cambio en una columna, se le dijo también a modificar la tabla. En apoyo de esta comienza con la siguiente fórmula:

ALTER TABLE *TableName*

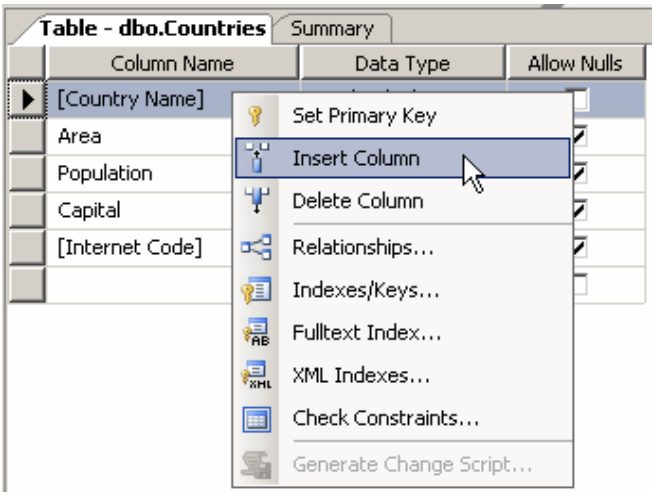
Al utilizar esta declaración, la expresión **de ALTER TABLE** se requiere y es seguido por el nombre de la

Añadir una nueva columna

Después de una tabla ya se ha creado, puede añadir una nueva columna a la misma.

Para añadir una nueva columna en SQL Server Management Studio, la primera a la derecha, haga cl haga clic en el cuadro de diseño. Para añadir una nueva columna a la final de la tabla, haga clic en vacío en la columna Nombre, escriba un nombre y especificar las otras opciones.

Para insertar una nueva columna entre dos existente, haga clic derecho en la columna que tendrá éxi Insertar columna:



Esto crearía un nuevo campo vacío. Escriba el nombre deseado y especificar las otras opciones.

En SQL, la fórmula básica para añadir una nueva columna a una tabla existente es:

ALTER TABLE *TableName*
Añadir *ColumnName Propiedades*

El *ColumnName* factor es necesario. De hecho, en el lado derecho de la palabra clave AGREGUE, defin su nombre y usando todas las opciones que hemos revisado para las columnas.

Aquí está un ejemplo:

ALTER TABLE StaffMembers
Añadir Dirección varchar (100) NULL
IR

Cuando este código se ejecuta, un nuevo nombre de columna de direcciones, de tipo **varchar,** con caracteres, y que permiten la entrada vacía, se añadirá a una tabla denominada StaffMembers en actual.

También puede utilizar el código de muestra para añadir una nueva columna a una tabla. En primer l vacía ventana de consulta y mostrar el Explorador de plantillas. Expanda el nodo de la tabla. En el cu soltar "Añadir columna en la ventana de consulta. Elimine el mal secciones de código y guardar sólo

refiere a la adición de una columna. Aquí está un ejemplo:

```
--==================================================
==========================
- Añadir columna plantilla
--
- Esta plantilla crea una tabla y, a continuación, se añade una nueva columna a la tabla.
--==================================================
==========================
USO <database, sysname, AdventureWorks>
IR

- Añadir una nueva columna a la tabla
ALTER TABLE <schema_name, sysname, dbo>. <table_name, Sysname, sample_table>
        Añadir <new_column_name, sysname, column3>
            <new_column_datatype,, datetime>
            <new_column_nullability,, NULL>
IR
```

Cambiar el nombre de una columna

Si encuentra que el nombre de una columna no es adecuada, se puede cambiar. Para cambiar el columna en el Explorador de objetos, haga clic derecho en la tabla que pertenece a la columna y haga En la vista diseño, resalte el nombre de la columna de ponerlo en modo de edición y editar.

En SQL, para cambiar el nombre de una columna, en primer lugar abrir una ventana de consulta vacía. de consulta, ejecute **sp_rename** utilizando la fórmula siguiente:

sp_rename *"TableName. ColumnName* ',' *NewColumnName* ',' COLUMN '

Sp_rename y el factor de la **"columna"** cadena son obligatorios. El factor de *TableName* es el nombr pertenece a la columna. El *ColumnName* es el nombre actual de la columna. El *NewColumnName* e desea darle a la columna.

Aquí está un ejemplo:

```
sp_rename 'StaffMembers.FullName', 'EmployeeName', 'COLUMN'
IR
```

Cuando este código se ejecuta, el intérprete busca una columna llamada FullName StaffMembers en actual base de datos seleccionada. Si se considera que la columna de la tabla, entonces se camb EmployeeName.

Eliminación de una columna

Si tienes un mal de columna que no quiere más en una tabla, puede eliminarla. Para eliminar una colu en el Explorador de objetos, expanda la base de datos, los cuadros, las columnas y nodos. Haga clic de haga clic en Eliminar columna. El cuadro de diálogo Eliminar objetos de pantalla. Si aún desea elin haga clic en Aceptar. Para cambiar tu mente, haga clic en Cancelar.

Para eliminar una columna utilizando el código, en primer lugar abrir o acceder a una ventana de utilizar la siguiente fórmula:

ALTER TABLE *TableName*
DROP *ColumnName* COLUMNA

En la parte derecha de la expresión **de ALTER TABLE,** escriba el nombre de la tabla. En la par expresión **de DROP columna,** escriba el nombre de la columna indeseable. Aquí está un ejemplo:

```
ALTER TABLE StaffMembers
DROP COLUMNA CurrentResidence;
```

IR

Cuando este código se ejecuta, el intérprete busca una columna llamada CurrentResidence e StaffMembers de la actual base de datos o seleccionado. Si se considera que la columna, se retire de la

Microsoft SQL Server también puede generar código de ejemplo que puede utilizar para eliminar una tabla. Antes de hacer esto, en primer lugar una pantalla vacía ventana de consulta y mostrar el Explora Expanda el nodo de la tabla. En el cuadro sección, arrastrar y soltar la columna caída en la ventana de el mal secciones de código y guardar sólo la parte que se refiere a la adición de una columna. Aquí está

```
--==========================================
- Caída de columna de plantilla
--
- Esta plantilla crea una tabla, entonces
- Cae una de las columnas de la tabla.
--==========================================
USO <database, sysname, AdventureWorks>
IR

- Caída de una columna de la tabla
ALTER TABLE <schema_name, sysname, dbo>. <table_name, Sysname, sample_table>
        DROP COLUMNA <new_column_name, sysname, column3>
IR
```

Aprendizaje práctico: Acabar con la Lección

1. Cierre la ventana de consulta sin guardar el archivo
2. En el Explorador de objetos, de conformidad con el nodo Bases de datos, haga clic derecho y haga BCR
3. En el cuadro de diálogo, haga clic en Aceptar

Resumen de la lección

Temas de Revisión

- Cuadros
- Columnas

Ejercicio: Utilidad de la empresa

1. UtilityCompany1 acceder a la base de datos
2. Visualmente crear una tabla llamada Empleados y que tiene las siguientes columnas: **EmployeeN Nombre, Apellido,** y el **título**
3. Crear una tabla llamada Clientes **AccountNumber, DateAccountCreated, CustomerName, Dir Estado,** y la **dirección de E-mail**

Ejercicio: Estados de EE.UU.

1. Consigue tu documentos de investigación en EE.UU. y las regiones de Nueva Inglaterra
2. En la hoja de papel, completa la lista con las regiones
3. Completar la lista con todos los Estados
4. Conectarse al servidor desde la línea de comandos y acceder a la base de datos UnitedStatesRegio
5. Crear una tabla con el nombre de las columnas Regiones **Región** y **descripción**
6. Crear una tabla denominada Estados con las columnas **Nombre, Código, Zona, Población,** y **de**
7. Salga del símbolo del

Fundamentos de los registros

Introducción

Un cuadro es un objeto que contiene la información de una base de datos. Debido a que una tabla es la parte central de una base de datos, la información que contiene debe ser meticulosamente organizado. Para gestionar mejor su información, los datos de una tabla se organiza en una serie de campos llamados células. Una vez que el cuadro contiene la información, puede revisar, ya sea utilizando SQL Server Management Studio o una aplicación externa.

Las tablas de una base de datos se muestran en el Explorador de objetos en su base de datos de nodo. Para abrir una tabla, puede hacer clic derecho y haga clic en haga clic en Abrir tabla.

Cuadro de datos de navegación en el SQL Server Management Studio

Consiste en datos de navegación y visualización de visualización de datos. Ya que la información de una base de datos se almacena en las tablas, su principal medio de visualización de datos consiste en la apertura de una mesa en una vista que muestra su información.

Cuando se muestra una tabla sus registros, a navegar a través de sus campos con el ratón o el teclado. Con el ratón, para llegar a cualquier célula, sólo haga clic en él. Para navegar a través de los registros mediante el teclado, puede pulsar:

- La tecla de flecha derecha para desplazarse a la celda derecha, si el cursor ya está en la celda más derecho, sería trasladado a la primera celda de la siguiente registro, hasta la última celda vacía de la primera acta vacía
- La tecla de flecha izquierda para moverse a la celda anterior, si el cursor se encuentra en, o alcance, la mayoría de las células de la izquierda el primer registro, no hay nada que sucedería cuando se pulsa la tecla de la flecha hacia la izquierda
- La tecla de flecha abajo para desplazarse a la celda en el marco del actual, si el cursor ya está en la última celda de la columna actual, que pasaría nada
- La tecla de flecha arriba para desplazarse a la celda justo por encima de la actual, si el cursor está ya en la primera celda de la columna actual, que pasaría nada
- La Página Abajo para pasar al siguiente grupo de células que corresponden a la página siguiente, si el número de registros es inferior a una página, el cursor se mueva a la última celda de la columna actual
- Hasta la página para pasar al siguiente grupo de células que corresponden a la página siguiente, si el número de registros es inferior a una página, el cursor se mueva a la primera celda de la columna actual

Visual de introducción de datos

Introducción

Como usted probablemente ya sabe, las columnas se utilizan para organizar los datos por categorías. Cada columna tiene una serie de campos en el encabezado de la columna. Uno de los efectos reales de una tabla se muestran los datos que está disponible para cada campo en una columna especial. Entrada de datos consiste en proporcionar los valores necesarios de los campos de una tabla. Los datos se introducen en un campo y cada vez hecho esto, la base de datos crea una fila de datos. En esta fila se denomina un registro. Esto significa que la introducción de datos también se auto-crea filas.

Hay cuatro formas principales que puede realizar el ingreso de datos para una tabla de Microsoft SQL Server:

- Puede utilizar una tabla a partir de la Objeto Explorer
- Puede introducir datos escribiendo código en una ventana de consulta
- Puede importar datos de otro objeto u otra base de datos
- Usted puede usar una aplicación externa como Microsoft Access, Microsoft Visual Basic, Borland C + + Builder, Microsoft Visual C + +, Borland Delphi, Microsoft Visual Basic, C #, Visual C #, J #, etc

Utilizando el Explorador de objetos

Probablemente la manera más fácil y rápida para introducir datos en una tabla es mediante el uso de SQL Server Management Studio. Por supuesto, primero debe abrir el cuadro de una base de datos disponible. En el Explorador de objetos, después de la ampliación de la bases de datos y nodos de los cuadros, para abrir un cuadro de entrada de datos. Si la tabla no contiene datos, al parecer con una fila vacía. Si algunos expedientes ya fueron consignados, se muestran sus filas y la tabla sería una fila vacía al final, esperando un nuevo registro.

Para realizar el ingreso de datos en una tabla, puede hacer clic en un campo. Cada columna tiene un título, un subtítulo llamado, en la parte superior. Esta sección gris en la parte superior se llama un encabezado de columna. En SQL Server, se muestra el nombre real de la columna. Se refiere al encabezado de columna para saber qué tipo de datos deben / debemos ir en un campo particular, en virtud de una columna. Esta es la razón por la que debe diseñar sus columnas meticulosamente. Después de la identificación de una columna, puede escribir un valor. A excepción del texto basada en las columnas, un campo puede aceptar o rechazar un valor si el valor no se ajusta al tipo de datos que se fija para la columna. Esto significa que en algunas circunstancias, puede que tenga que proporcionar cierta información explícita o más para el usuario.

Aprendizaje práctico: Presentación de introducción de datos

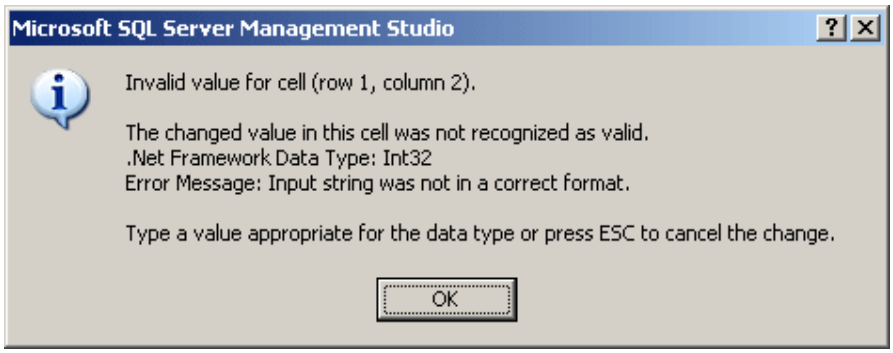
1. Inicie Microsoft SQL Server, seleccione las opciones apropiadas en el cuadro de diálogo Conectar al servidor "caja y conectar con el servidor
2. Haga clic derecho en el nombre del servidor y haga clic en Nueva consulta
3. Para crear una nueva base de datos, en la ventana vacía, escriba lo siguiente:

```
CREATE DATABASE WorldStatistics;
IR
USO WorldStatistics;
IR
CREATE TABLE países
(
    [País] VARCHAR (80),
    INT zona,
    Población BIGINT,
    Capital VARCHAR (50),
    [Código de Internet] char (2)
);
IR
```

4. Para ejecutar la sentencia SQL, presione F5
5. Cierre la ventana de consulta
6. Cuando se le preguntó si desea guardar, haga clic en Sí.
7. Tipo de **países** y haga clic en Guardar
8. En el Explorador de objetos, haga clic derecho en el nodo Bases de datos y haga clic en Actualizar. Expanda el nodo Bases de datos.

En virtud de las bases de datos, expanda WorldStatistics y ampliar los cuadros

9. Si no ve una tabla denominada países, haga clic derecho en el nodo Tablas y haga clic en Actualizar.
   Haga clic derecho en los países y haga clic en Abrir el cuadro, coloque el ratón en el cuadro Abrir y haga clic en Volver Todas las filas
10. Como el cursor se coloca en el primer campo vacío bajo **CountryName,** tipo **Costa de** Marfil y presione Enter
11. Tipo de 322.460 para el área y pulse Tabulador
12. Tipo de **16.393.221** y pulse Enter
13. Observe que recibe un error, porque las comas no se permiten:



14. Haga clic en Aceptar en el cuadro de mensaje de error.
15. Cambia el valor a **16393221** y pulse Tabulador **Personas**
16. Observe que usted recibe un error porque la columna está configurado para un número natural y no una cadena
17. Haga clic en Aceptar en el cuadro de mensaje de error y borrar Personas
18. En virtud del **código de Internet,** el tipo y pulse Intro **ci**
19. Haga clic en el campo **de capital,** el tipo **de Yamusukro** y presione Enter dos veces
20. Completa el cuadro de la siguiente manera:

Nombre País	Espacio	Población	Capital	Código de Internet
Côte d'Ivoire	322460	16393221	Yamusukro	ci
Panamá	78200	3191319	Panamá	pa
Australia	7686850	20264082	Camberra	au
Canadá	9984670	33098932	Otawa	CA
Irán	1648000	68688433	Teherán	ir

21. Cierre la tabla

Con la introducción de datos de SQL

Introducción

Para realizar la entrada de datos con SQL:

- En el Explorador de objetos, puede derecha de la mesa, coloca el ratón sobre el cuadro de comandos de Como -> INSERT A -> Nueva ventana del editor de Consulta
- Abra una ventana de consulta vacía y escriba su código

En el SQL, la entrada de datos se realiza utilizando **el** comando INSERT combinado con los

valores palabras. La principal declaración utiliza la sintaxis siguiente:

INSERTE *TableName* VALUES *(column1, column2, Column_n);*

Como alternativa, o para ser más exactos, puede utilizar la palabra clave **en el** comando INSERT entre palabra clave y el factor de *TableName* para especificar que está introduciendo datos en la tabla. Esto se hace con la siguiente sintaxis:

INSERT INTO *TableName* VALUES *(column1, column2, Column_n)*

TableName El factor debe ser un nombre válido de una tabla existente en la base de datos que está utilizando. Si el nombre está mal, el intérprete de SQL simplemente consideran que la tabla que se refiere a que no existe. Por consiguiente, usted recibirá un mensaje de error.

VALORES La palabra clave indica que está listo para la lista de los valores de las columnas. Los valores de las columnas deben ser incluidos entre paréntesis.

Si la columna es **un** poco el tipo de datos, debe especificar uno de sus valores 0 o 1.

Si la columna es un tipo numérico, se debe prestar atención al número que escriba. Si la columna ha sido configurado para recibir un número entero **(int, bigint, smallint),** debe proporcionar un número válido naturales sin el separador decimal.

Si la columna es un número decimal (float, **real, decimal, numérico),** puede escribir el valor de su carácter separador (el período de EE.UU. Inglés).

Si la columna se creó para una fecha tipo de datos, asegúrese de proporcionar una fecha válida.

Si el tipo de datos de una columna es una cadena tipo, deberá incluir su entrada entre comillas simples. Por ejemplo, un estante número puede especificarse como "HHR-604 y una inicial puede darse como" D ".

En los párrafos anteriores, se nos afirma que "usted", como si va a la realización de una entrada de datos. En realidad, el usuario deberá realizar la entrada de datos en sus productos. Por lo tanto, es su responsabilidad para reducir, tanto como sea posible, la probabilidad de errores. Por supuesto, hay diversas maneras, a través de una "visual", como el de Borland C + + Builder, Microsoft Visual Basic, C #, o MS Visual C + +, etc, que usted puede cuidar de ello.

Adyacentes de introducción de datos

La técnica más común de realizar el ingreso de datos que requiere conocer la secuencia de los campos de la tabla en la que desee introducir datos. Con esta lista en cuenta posteriores, introduzca el valor de cada campo en su posición correcta.

Durante la entrada de datos en los campos adyacentes, si no tiene un valor para un campo numérico, debe escriba 0 como su valor. Para una cadena de campo cuyos datos no tiene y no puede proporcionar, escriba dos comillas''solo para especificar un campo vacío.

Aprendizaje práctico: Realización de lado de entrada de datos

1. Para abrir una nueva ventana de consulta, pulse Ctrl + N
2. En la ventana de consulta, a fin de crear un registro, escriba:

```
USO WorldStatistics;
IR
INSERT INTO países
VALUES ( 'Angola', 1246700, 12127071, 'Luanda', 'ao');
IR
```

3. Presione la tecla F5 para ejecutar
4. Eliminar la sección superior de la ventana
5. Para introducir varios registros al mismo tiempo, entrar en la siguiente declaración:

```
USO WorldStatistics;
IR
INSERT INTO países
VALUES ( 'Mexico', 1972550, 107449525, 'Ciudad de México', 'x');
IR
INSERT INTO países
VALUES ( 'Sudáfrica ", 1219912, 44187637,' Pretoria ',' za ');
IR
INSERT INTO países
VALUES ( 'Irak', 0, 0, 'Bagdad', 'iq');
IR
INSERT INTO países
VALUES ( 'Estados Unidos', 9826630, 0,'','');
IR
INSERT INTO países
VALUES ( 'Arabia Saudita', 2149690, 0, 'Riad','');
IR
```

6. Presione la tecla F5 para ejecutar la declaración

De introducción de datos al azar

La entrada de datos adyacentes, hemos realizado es necesario que usted sepa la posición de cada columna. SQL ofrece una alternativa que le permite realizar la entrada de datos usando el nombre de una columna en vez de su posición. Esto le permite proporcionar los valores de las columnas de una orden de su elección. Hemos visto algunos ejemplos en los que los valores de algunos de los campos no estaban disponibles durante la entrada de datos. En lugar de recordar a tipo 0 o NULL para esos campos o dejar vacío cita para un campo, puede utilizar los campos "para especificar los nombres de los campos cuyos datos desee proporcionar.

Para realizar el ingreso de datos en un orden de su elección, usted debe proporcionar la lista de los campos de la tabla. Puede utilizar todas las columnas o proporcionar una lista de las mismas columnas, pero en su propio orden. De la misma manera, usted no tiene que proporcionar los datos de todos los campos, sólo los que desee, en el orden que desee.

Aprendizaje práctico: Realización de introducción de datos al azar

1. Para realizar el ingreso de datos, el tipo y la ejecución de la siguiente declaración:

```
USO WorldStatistics;
IR
Países de INSERT ([País], Capital, [código de Internet], Población, Espacio)
VALUES ( 'China', 'Pekín', 'cn', 1313973713, 9596960)
IR
```

2. Presione la tecla F5 para ejecutar la declaración
3. Para realizar otras entradas, escriba la siguiente declaración:

```
USO WorldStatistics;
IR
Países de INSERT (Capital, [código de Internet], [País])
VALUES ( 'Nouakchott', 'mr', 'Mauritania')
```

```
IR
Países de INSERT ([código de Internet], Población, [País])
VALUES ( 'ro', 22303552, «Rumanía»)
IR
INSERTE Países (Zona, [País], Población)
VALORES (21040, 'El Salvador', 6822378)
IR
Países de INSERT (Capital, [País])
VALUES ( 'Phnom Penh', 'Camboya')
IR
```

4. Para ejecutar la declaración, presione F5
5. Cierre la ventana de consulta
6. Cuando se le preguntó si desea guardar, haga clic en No

La nulidad de un Campo

Introducción

Durante la entrada de datos, los usuarios de su base de datos se enfrentan a los campos de datos que esperar. A veces, por una razón u otra, los datos no estarán disponibles para un campo determinado. Un ejemplo sería un MI (inicial) de campo: algunas personas tienen una inicial, o bien algunos otros no lo tiene o no (o no) le proporcione. Este aspecto puede ocurrir por cualquier campo de su tabla. Por lo tanto, usted debe pensar en una manera de tratar con ella.

Un campo que se conoce como nulo cuando no la entrada de datos se ha realizado a la misma:

- Decir que el campo es nulo, no significa que contiene 0 porque 0 es un valor
- Decir que el campo es nulo, no significa que está vacío. Un campo que se vacía puede significar que el usuario ha suprimido su contenido o que el propio campo no aceptar lo que el usuario está tratando de entrar en ese campo, un campo vacío, pero puede tener un valor

Un campo que se conoce como nulo si no hay manera de determinar el valor de su contenido (en realidad, el ordenador, es decir, el sistema operativo, tiene su propio mecanismo interno de verificar el valor de un campo) o su valor es simplemente desconocidas. Como puedes imaginar, no es una buena idea tener un campo nulo en su mesa. Como un desarrollador de base de datos, es su responsabilidad siempre se sabe con certeza el valor de cada campo de su tabla.

Un campo se refiere a si el usuario lo requiere debe proporcionar un valor para que, antes de trasladarse a otro récord. En otras palabras, el campo no se pueden dejar vacíos durante el ingreso de datos.

Para resolver el problema de la nula y campos obligatorios, Microsoft SQL Server propone una de las dos opciones: permitir o no permitir valores nulos en un campo. Por un típico cuadro, son piezas de información que el usuario debe asegurarse de entrar, de lo contrario, la entrada de datos no sería validado. Para asegurarse de que el usuario siempre se llena un campo determinado antes de pasar al siguiente campo, es decir, para exigir que el valor, si son visualmente la creación de la mesa, limpiar la casilla de verificación Permitir nulos para el campo. Por otro lado, si el valor de un campo no es especialmente importante, por ejemplo, si usted no tiene intención de incluir este valor en una operación algebraica, comprobar su casilla de verificación Permitir nulos.

NULL o NOT NULL?

Si la creación de una tabla usando SQL, para especificar que puede permitir valores nulos, tipo

NULL en la parte derecha de la columna. Para especificar que los valores de la columna están obligados, en el lado derecho, de tipo **NOT NULL.** Si no se especifica NULL o NOT NULL, la columna será creado como NULL. Aquí están algunos ejemplos:

```
CREATE TABLE personas
(
      Nombre varchar (20) NULL,
      Apellido varchar (20) NOT NULL,
      Género smallint
);
IR
```

Si la tabla ya está creada y ya tiene algunos valores, no se puede establecer la opción Permitir valores NULL en las columnas que no tienen valores.

Aprendizaje práctico: Aplicación de Campos Nulidad

1. En el Explorador de objetos, haga clic derecho en los Países del WorldStatistics nodo y haga clic en Modificar
2. Aplicar la nulidad de los campos de la siguiente manera:

Table - dbo.Countries | Summary

Column Name	Data Type	Allow Nulls
[Country Name]	varchar(80)	☐
Area	int	☑
Population	bigint	☑
Capital	varchar(50)	☑
[Internet Code]	char(2)	☑
		☐

3. Guardar la tabla

El valor por defecto de una columna

Introducción

A veces, la mayoría de los registros por debajo de cierta columna puede celebrar el mismo valor, aunque sólo unos pocos serían diferentes. Por ejemplo, si una escuela está utilizando una base de datos para registrar a sus estudiantes, todos ellos tienen más probabilidades de ser del mismo estado. En tal caso, puede ayudar al usuario de forma automática proporciona un valor para esa columna. El usuario simplemente aceptar el valor y el cambio sólo en los raros casos en que el valor pasaría a ser diferente. Para ayudar al usuario con este valor común, se crea lo que se denomina un valor por defecto.

Creación de una forma visual Valor predeterminado

Puede crear un valor por defecto de una columna al crear una tabla. Para especificar el valor por defecto de una columna, en la sección superior, haga clic en la columna. En la sección inferior, haga clic en Valor predeterminado o encuadernación, escriba el valor deseado con arreglo a las normas de la columna del tipo de datos:

Es el tipo de datos es	Instrucciones
Basados en texto (char, varchar, texto, y sus variantes)	Introduzca el valor en una sola cita
Numéricos basados en	Introduzca el valor como un número, pero siguiendo las reglas del tipo de datos.

	Por ejemplo, si introduce un valor superior a 255 para un **tinyint,** usted recibirá un mensaje de error
Fecha o la hora	Introduzca la fecha, bien como DD / MM / AAAA o AAAA / MM / DD. Opcionalmente se puede incluir la fecha en una sola cita. Introduzca la hora siguiente a la normas establecidas en el Panel de control (Configuración regional).
Trozo	Introduzca el valor 0 para FALSE largo o cualquier otro valor de TRUE

Creación de una programación Valor predeterminado

Para especificar el valor por defecto en una sentencia SQL, al crear la columna, antes de la coma o el paréntesis de cierre de la última columna, asignar el valor deseado para la **DEFAULT.** Aquí están algunos ejemplos:

```
CREATE TABLE empleados
(
      FullName VARCHAR (50),
      Dirección VARCHAR (80),
      Ciudad VARCHAR (40),
      Estado VARCHAR (40) DEFAULT = "Nueva Gales del Sur»,
      PostalCode VARCHAR (4) DEFAULT ='2000 ',
      País VARCHAR (20) DEFAULT = 'Australia'
);
IR
```

Después de crear la tabla, el usuario no tiene que proporcionar un valor para una columna que tiene un defecto. Si el usuario no proporciona el valor, el valor por defecto se utiliza cuando se guarda el registro.

Si el usuario proporciona un valor para una columna que tiene un valor predeterminado y, a continuación, elimina el valor, el valor por defecto no se aplicará la norma más: El campo simplemente se vacían

Columnas de identidad

Introducción

Uno de los objetivos de una buena mesa es ser capaz de identidad única cada registro. En la mayoría de los casos, el motor de base de datos no debe confundir dos registros. Considere la siguiente tabla:

Categoría	**Nombre del artículo**	**Tamaño**	**Precio**
Mujer	Jersey de manga larga de vestir	Grandes	39,95
Chicos	Libre de hierro plisado Caqui Pantalones	S	39,95
Hombres	Rayas camisa de manga larga	Grandes	59,60
Mujer	Jersey de manga larga de vestir	Grandes	45,95
Chicas	Bolso del hombro		45,00

Mujer	Continental falda	Petite	39,95

Imagínese que usted desea cambiar el valor de un elemento llamado Jersey manga larga de vestir. Porque usted debe encontrar el elemento mediante programación, puede empezar a buscar un tema con ese nombre. Este cuadro pasa a tener dos temas con ese nombre. Usted puede decidir entonces a buscar un artículo con su categoría. En la columna Categoría, hay demasiados elementos cuyos nombres Mujer. De la misma manera, hay demasiados registros que tienen un gran valor en la columna Tamaño, lo mismo problema en la columna Precio por unidad. Esto significa que usted no tiene un buen criterio que puede utilizar para aislar el registro cuyo nombre es Punto de manga larga camisa.

Para resolver el problema de la identificación de forma única un registro, puede crear una columna cuyo principal objetivo es distinguir un registro de otra. Para ayudarle con esto, la sentencia SQL le permite crear una columna cuyo tipo de datos es un tipo, pero el usuario no tendrá que introducir los datos de esa columna. Un valor que automáticamente se introducirán en el campo cuando un nuevo registro es creado. Este tipo de columna que se llama una columna de identidad.

Usted no puede crear una identidad de una columna de una tabla existente, sólo en una nueva tabla.

Creación de una identidad visual Columna

Para crear una columna de identidad, si está visualmente que trabajan en el diseño habida cuenta de la mesa, en la sección superior, especifique el nombre de la columna. Por tradición, el nombre de esta columna se asemeja a la de la tabla, pero en singular. Además, por costumbre, el nombre de la columna termina con _id, ID, o ID.

Después de especificar el nombre de la columna, fijar su tipo de datos a un entero basado en el tipo. Por lo general, el tipo de datos utilizado es **int.** En la sección inferior, haga clic en y ampliar la propiedad **Especificación de identidad.** La primera acción que debe tomar es establecer su **(¿Es la identidad)** la propiedad de No a Sí.

Una vez establecido el valor de la persona **(identidad)** de propiedad a Sí, la primera vez que el usuario realiza la entrada de datos, el valor del primer registro que se establece en 1. Esta característica está controlada por la **propiedad de Semillas de Identidad.** Si desea que la cuenta para empezar a un valor distinto de 1, especificar en esta propiedad.

Después de que el (¿Es la identidad) de propiedad se ha establecido en Sí, el intérprete de SQL se incremento el valor de cada nuevo registro por 1, que es el valor por defecto. Esto significa que el primer registro tendrá un valor de 1, el segundo tendría un valor de 2, y así sucesivamente. Este aspecto es controlado por el **Incremento de identidad de** la propiedad. Si desea incremento por más que eso, puede cambiar el valor de la propiedad **de identidad de Incremento.**

Aprendizaje práctico: Creación de una columna de identidad

1. En el Explorador de objetos, en virtud de WorldStatistics, haga clic derecho en los cuadros y haga clic en Nuevo Cuadro ...
2. Defina el nombre de la columna y pulse Tabulador **ContinentID**
3. Establezca su tipo de datos a **int** y pulse F6.
   En la parte inferior de la tabla, ampliar la especificación de la identidad y haga doble clic **(identidad)** para ajustar su valor a Sí
4. Completa el cuadro de la siguiente manera:

Nombre de columna	**Tipo de datos**	**Permitir valores**

		NULL
ContinentID		
Continente	varchar (80)	*Sin marcar*
Espacio	bigint	
Población	bigint	

5. Guarde la tabla como **Continentes**

Creación de una columna de identidad de Uso de SQL

Si está programada la creación de una columna, para indicar que sería utilizado como una columna de identidad después de su nombre y tipo de datos, el tipo **de identidad** seguido de paréntesis. Entre los paréntesis, introduzca el valor de las semillas, seguido de una coma, seguido por el valor de incremento. Aquí está un ejemplo:

```
CREATE TABLE StoreItems (
ItemID int IDENTIDAD (1, 1) NOT NULL,
Categoría varchar (50),
[Nombre del artículo] varchar (100) NOT NULL,
Tamaño varchar (20),
[Precio] dinero);
IR
```

Funciones y de introducción de datos

Introducción

Usted puede participar en una función de entrada de datos. Como ejemplo, usted puede llamar a una función que devuelve un valor que para asignar valor a una columna. Usted puede crear su propia función y uso, o puede utilizar una de las funciones incorporadas.

Uso de Funciones

Con el fin de participar con su función de entrada de datos, debe tener una identidad y. Puede utilizar una de las funciones de Transact-SQL. Usted puede comprobar una de las funciones que hemos examinado en **Lección 7.** Normalmente, la mejor manera es comprobar la documentación en línea para averiguar si la tarea que desea llevar a cabo ya está creado. Usando una función de espacio que el problema de tener una función. Por ejemplo, imagine que tiene una base de datos llamada AutoRepairShop e imaginar que tiene un cuadro que se utilizará para crear las órdenes de reparación para los clientes:

```
CREATE TABLE RepairOrders
(
   Int RepairID Identidad (1,1) NOT NULL,
   CustomerName varchar (50),
   CustomerPhone varchar (20),
   RepairDate de fecha
);
IR
```

Al realizar el ingreso de datos para este cuadro, puede dejar que el usuario introduzca el nombre del cliente y número de teléfono. Por otro lado, puede ayudar al usuario entrar en la programación por fecha actual. Para ello, se llame a la **GETDATE ()** función. Aquí están algunos ejemplos:

```
INSERT INTO RepairOrders (CustomerName, CustomerPhone, RepairDate)
                VALUES ( 'Annette Berceau »,« 301-988-4615', GETDATE ());
IR
```

```
INSERT INTO RepairOrders (CustomerPhone, CustomerName, RepairDate)
                VALUES ( '(240) 601-3795 "," Paulino Santiago', GETDATE ());
IR
INSERT INTO RepairOrders (CustomerName, RepairDate, CustomerPhone)
                VALUES ( 'Alicia Katts', GETDATE (), "(301) 527-3095');
IR
INSERT INTO RepairOrders (RepairDate, CustomerPhone, CustomerName)
                VALUES (getdate (),'703-927-4002 ',' Bertrand Nguyen ');
IR
```

También puede implicar la función en una operación y, a continuación, utilizar el resultado como el valor a asignar a un campo. También puede llamar a una función que toma uno o más argumentos, asegúrese de respetar las normas de pasar un argumento a una función cuando se llama a él.

Si ninguno de los de Transact-SQL funciones incorporadas satifies sus necesidades, puede crear su propia iniciativa y utilizando las técnicas que estudiamos en **la Lección 6.**

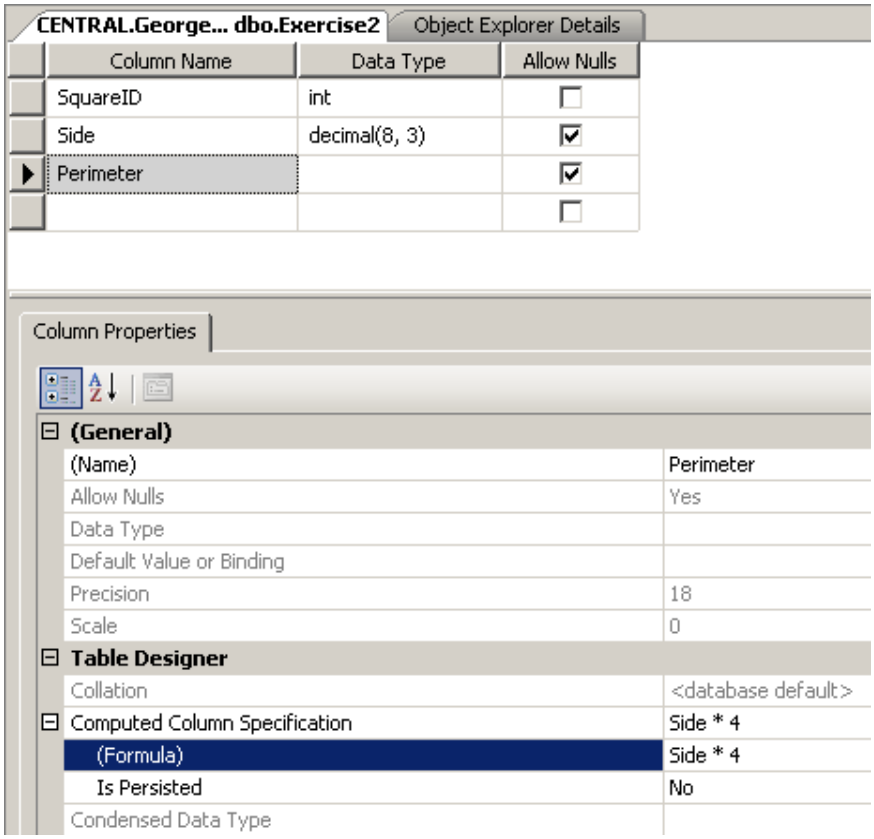
Uso de expresiones para la entrada de datos

Introducción

Hay varias maneras en que puede ayudar al usuario con la entrada de datos. Además de utilizar una función, puede crear una expresión usando los operadores como los que hemos revisado en las clases 3 y 5. Puede crear una expresión cuando se crea una tabla, ya sea en la ventana o utilizar la tabla de SQL en una ventana de consulta.

Creación de una expresión visual

Para crear una expresión visual cuando la creación de una mesa, en la sección superior, especifique el nombre de la columna (sólo el nombre de la columna es importante). En la sección inferior, ampliar el campo de la especificación de columnas computarizada y, en su (Fórmula), escriba la expresión deseada. Aquí está un ejemplo:



Creación de una expresión de SQL

También puede crear una expresión en la expresión SQL que está utilizando para crear una tabla. Para ello, en el marcador de posición de la columna, escriba el nombre de la columna, seguido por AS, y seguida de la expresión deseada. Aquí está un ejemplo:

```
CREATE TABLE Círculo
(
    Int CircleID identidad (1,1) NOT NULL,
    Decimales radio (8, 3) NOT NULL,
    Zona como Radio Radio * * PI ()
);
IR
```

Durante utilizando una expresión de entrada de datos

Al realizar el ingreso de datos, no debe proporcionar un valor para una columna que tiene una expresión, el intérprete de SQL proporcionaría el valor automáticamente. Aquí hay un ejemplo de introducción de datos para el círculo por encima de la tabla:

```
INSERT INTO Circle (Radio) VALUES (46,82);
IR
INSERT INTO Circle (Radio) VALUES (8,15);
IR
INSERT INTO Circle (Radio) VALUES (122,57);
IR
```

Compruebe Limitaciones

Introducción

Al realizar el ingreso de datos, en algunas columnas, incluso después de que indica los tipos

de valores que espera el usuario para establecer una cierta columna, es posible que desee restringir una serie de valores que se permiten. Para ayudar a comprobar si con una nueva entrada de valor se inscribe en el rango deseado, Transact-SQL proporciona lo que se denomina un cheque limitación.

Un cheque es un obstáculo booleano operación realizada por el intérprete de SQL. El intérprete se examina un valor que ha sido proporcionada por una columna. Si el valor es el caso:

1. La limitación produce **TRUE**
2. El valor se acepta
3. El valor es asignado a la columna

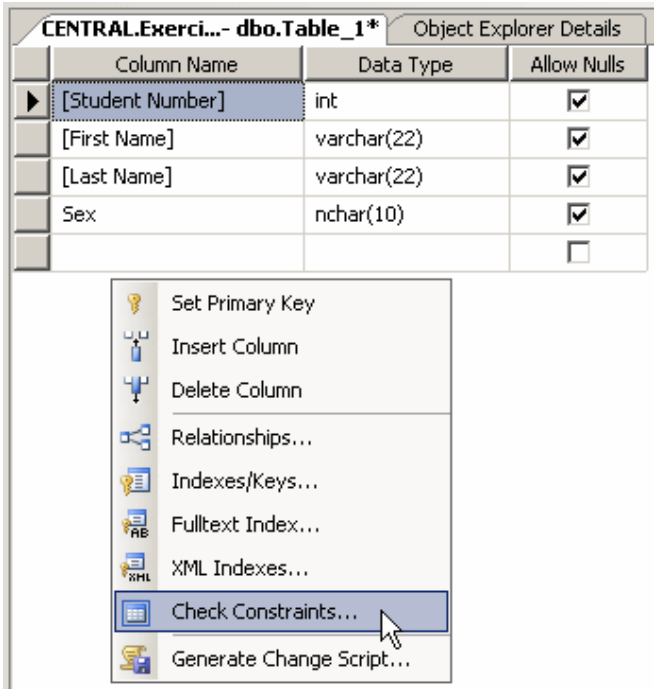
Si el valor no es el caso:

1. La limitación produce **FALSE**
2. El valor se rechazó
3. El valor no es asignado a la columna

Puede crear un cheque limitación en el momento que está creando una tabla.

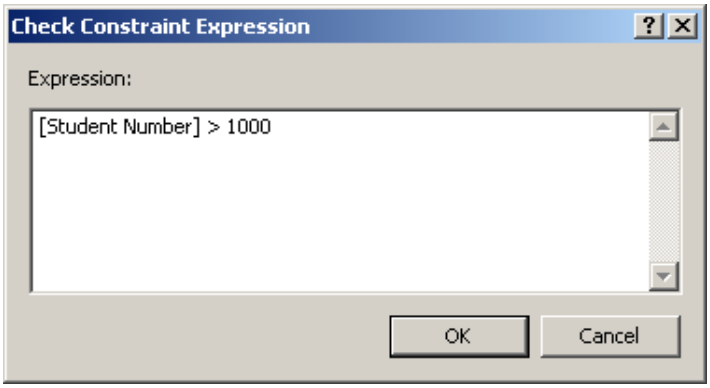
Creación de un Compruebe visualmente Restricción

Para crear un cheque limitación, cuando se crea una tabla, haga clic derecho en cualquier parte (incluso fuera de) la mesa y haga clic en Comprobar las limitaciones ...



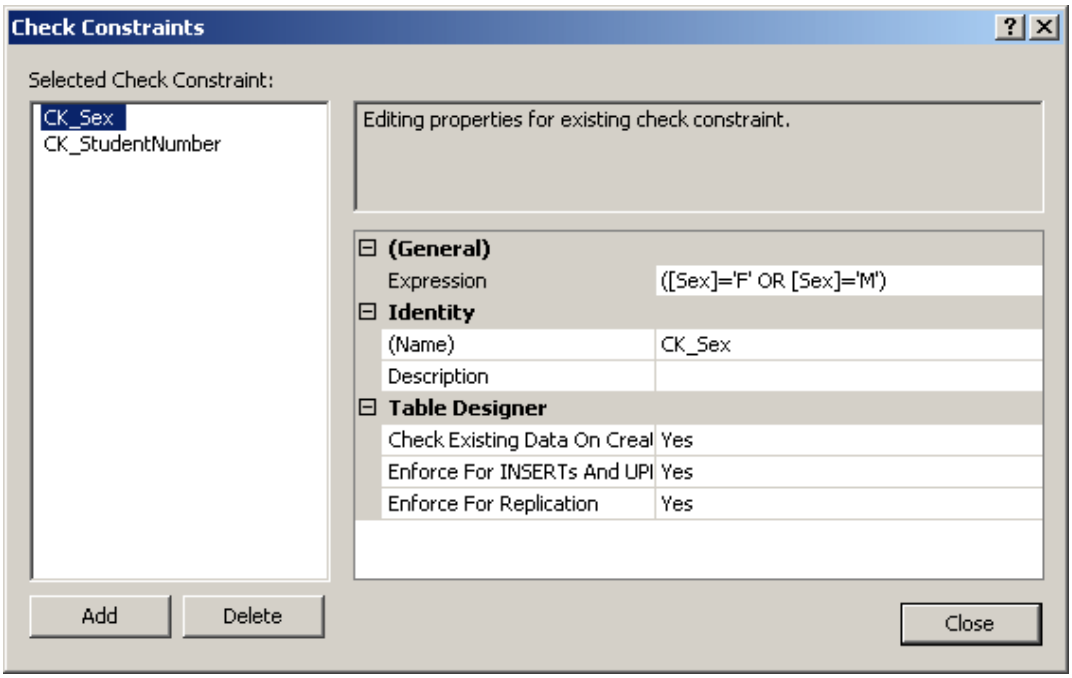
Esto abriría el cuadro de diálogo Comprobar Limitaciones. Desde esa ventana, puede hacer clic en Añadir. Debido a una limitación es un objeto, debe proporcionar un nombre para él. La pieza más importante de información que un cheque limitación debe celebrar es el mecanismo que se utiliza para comprobar sus valores. Esto está previsto como una expresión. Por lo tanto, para crear una limitación, puede hacer clic en Expresión y haga clic en su botón de puntos suspensivos. Esto abriría el CHECK constraint cuadro de diálogo Expresión.

Para crear la expresión, en primer lugar escriba el nombre de la columna en la que la limitación se aplicará, seguido de paréntesis. En el paréntesis, el uso de la aritmética y / o SQL operadores ya hemos estudiado. Aquí hay un ejemplo de que se compruebe que un nuevo valor especificado para el número de estudiante es mayor de 1000:



Después de crear la expresión, puede hacer clic en Aceptar. Si la expresión no es válida, usted recibirá un mensaje de error y les da la oportunidad de corregirlo.

Puede crear tantos como comprobar las limitaciones que consideren necesarios para su mesa:



Después de crear el cheque limitaciones, puede hacer clic en Aceptar.

Creación de una programación CHECK constraint

Para crear una limitación en SQL comprobar, en primer lugar crear la columna en la que la limitación se aplicará. Antes de que el paréntesis de cierre de la definición de la tabla, utilice la siguiente fórmula:

CONSTRAINT *nombre* CHECK *(expresión)*

La limitación y **el** CHECK palabras son obligatorios. Como un objeto, asegúrese de que usted

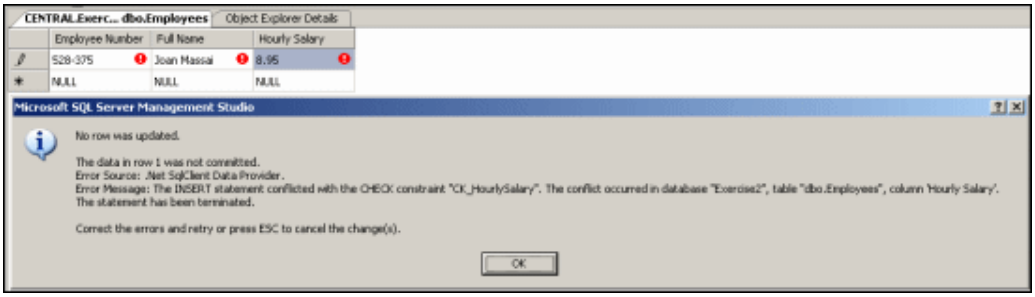
proporciona un *nombre* para él. Dentro de los paréntesis que siguen a la **VERIFICA** operador, introduzca la expresión que se aplicarán. Aquí hay un ejemplo que se asegurará de que el salario por hora especificada para un empleado es superior a 12,50:

```
CREATE TABLE empleados
(
        [Número de empleados] NCHAR (7),
        [Nombre completo] varchar (80),
        [Salario por hora] smallmoney,
        CK_HourlySalary CONSTRAINT CHECK ([Salario por hora]> 12.50)
);
```

Es importante comprender que una limitación comprobar que no es ni una expresión ni una función. Un cheque contiene una limitación de expresión y pueden contener una función como parte de su definición.

Después de crear la limitación (s) para una mesa, en el Explorador de objetos de Microsoft SQL Server Management Studio, en el interior del nodo de la tabla, hay un nodo llamado limitaciones y, si expandirla, aparecerá el nombre de la limitación.

Con la limitación (s) en su lugar, durante la entrada de datos, si el usuario (o su código), establece un valor no válido, un error de pantalla. Aquí está un ejemplo:



En lugar de una expresión que sólo utiliza a los operadores, puede utilizar una función para ayudar en el proceso de chequeo. Puede crear y utilizar su propia función o puede usar uno de los incorporados en las funciones de Transact-SQL.

Otras funciones de introducción de datos

Es RowGuid

Esta propiedad le permite especificar que una columna con la propiedad **de identidad** ajustado a **Sí** se utiliza como una columna **ROWGUID.**

Cotejo

Puesto que los diferentes lenguajes que utilizan diferentes mecanismos en sus caracteres alfabéticos, esto puede afectar la manera en algún tipo de algoritmos o consultas se realizan en los datos, usted puede pedir la base de datos para aplicar un determinado lenguaje mecanismo sobre el terreno mediante el cambio de la **colación de** bienes. En caso contrario, deberá aceptar el valor por defecto especificado en la tabla.

Importación de datos

Otra técnica utilizada para obtener datos en una o más tablas consiste en la importación de los datos ya existentes a partir de otra base de datos o de cualquier otro archivo de datos reconocibles. Microsoft SQL Server proporciona diversas técnicas y medios de importación de datos.

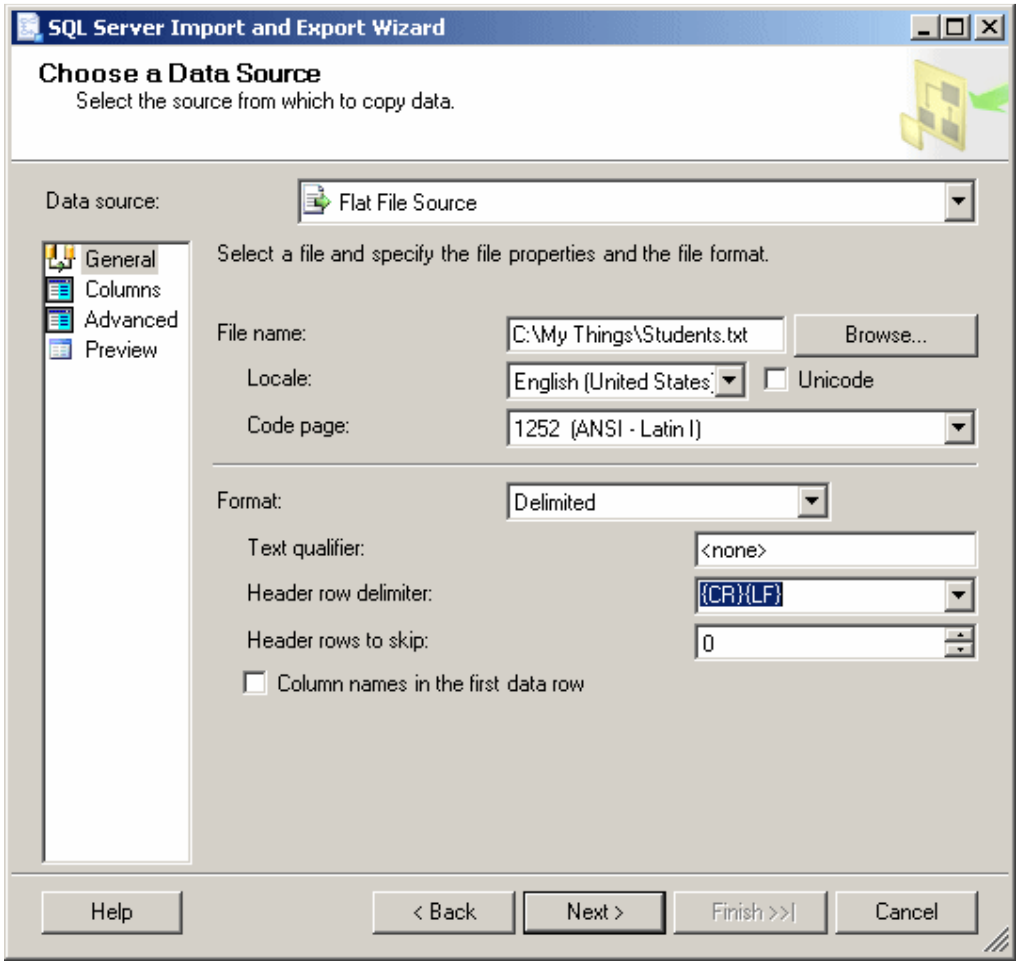
La forma más fácil el tipo de datos que puede ser importada en SQL Server, y que está disponible en casi todos los entornos de bases de datos, es el archivo de texto. Casi todos los entorno de base de datos le permite importar un archivo de texto pero los datos de ese archivo debe estar formateado correctamente. Por ejemplo, la información almacenada en el archivo debe definir las columnas que distingue por un personaje que sirve como un separador. Este separador puede ser la única cita, la doble cotización, o de cualquier carácter válido. Datos entre las comillas se considera como perteneciente a un ámbito distinto. Además de esta información, la base de datos tendría que separar la información de dos columnas. Una vez más, un carácter válido debe ser utilizada. La mayoría de bases de datos, incluyendo Microsoft SQL Server, reconoce la coma como tal un carácter. La última pieza de información debe proporcionar el archivo es distinguir cada registro de otra. Esto es fácilmente adoptadas por el coche de fin de línea de un registro. Esto también es reconocido como el retorno de carro.

Estas directrices pueden ayudarle a crear manualmente un archivo de texto que puede ser importada en Microsoft SQL Server. En la práctica, si desea importar los datos que reside en otra base de datos, usted puede pedir que la aplicación para crear la fuente de datos. La mayoría de las aplicaciones que pueden hacer y el formato de los datos. Tal es el caso de los datos que usaremos en el siguiente ejercicio: se trata de datos que residen en una base de datos Microsoft Access y está dispuesta a ser importadas en Microsoft SQL Server.

Después de importar los datos, deberá verificar y, posiblemente, el formato para personalizar sus campos.

Aprendizaje práctico: Importación de datos desde una fuente externa

1. Descargue el **Estudiantes** archivo de texto y guardarlo en tu disco duro
2. En el SQL Server Management Studio, haga clic derecho en el nodo Bases de datos y haga clic en Nueva base de datos ...
3. **Rosh** Tipo y pulse Enter
4. En el Explorador de objetos, haga clic derecho en Rosh, coloca el ratón sobre **las tareas** y haga clic en **Importar datos**
5. En la primera página del asistente, haga clic en Siguiente
6. En la segunda página, haga clic en la flecha del cuadro combinado de origen de datos y seleccione Archivo Plana Fuente
7. En la parte derecha de Nombre de archivo, haga clic en el botón Examinar
8. Busque y seleccione el archivo que Students.txt había ahorrado

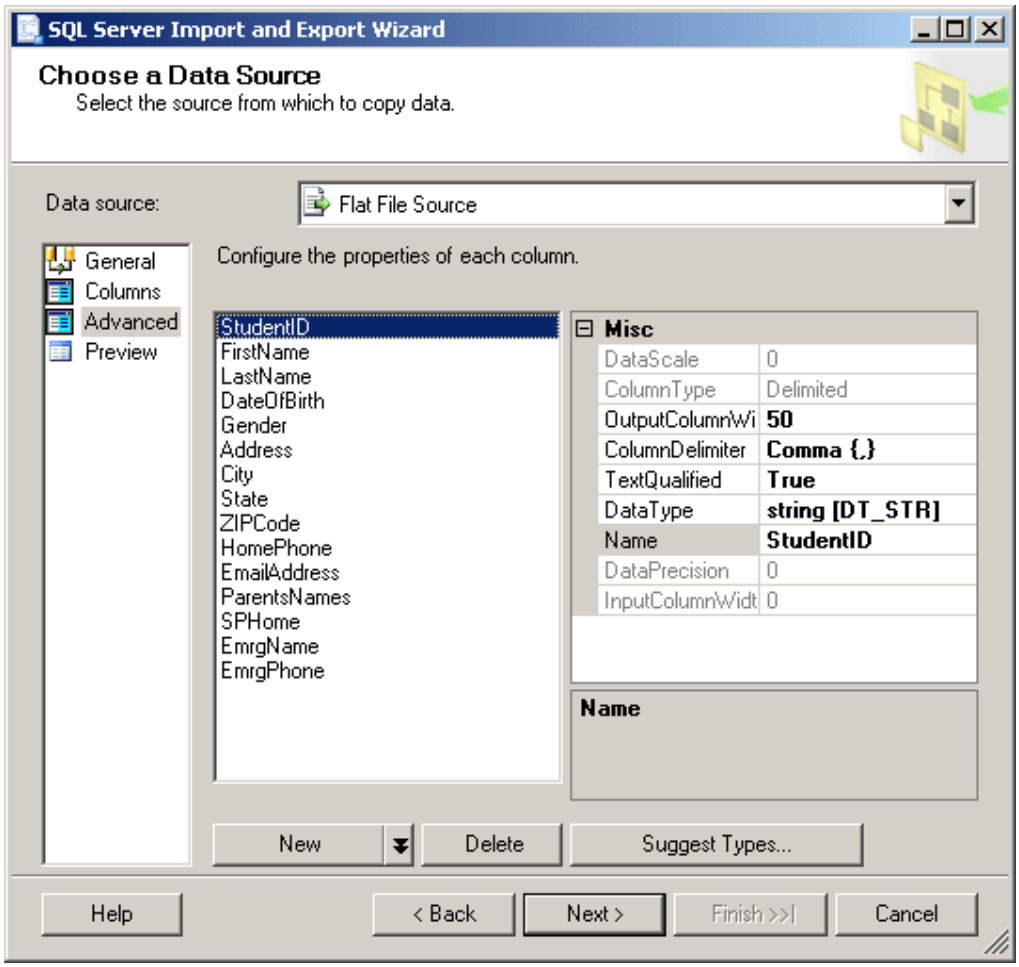


9. En virtud del origen de datos, haga clic en Opciones avanzadas
10. En la columna se ha seleccionado, en la lista de la derecha, haga clic en Nombre y tipo StudentID
11. En el centro la lista, haga clic en cada columna y cambiar su nombre en la columna de la derecha de la siguiente manera:

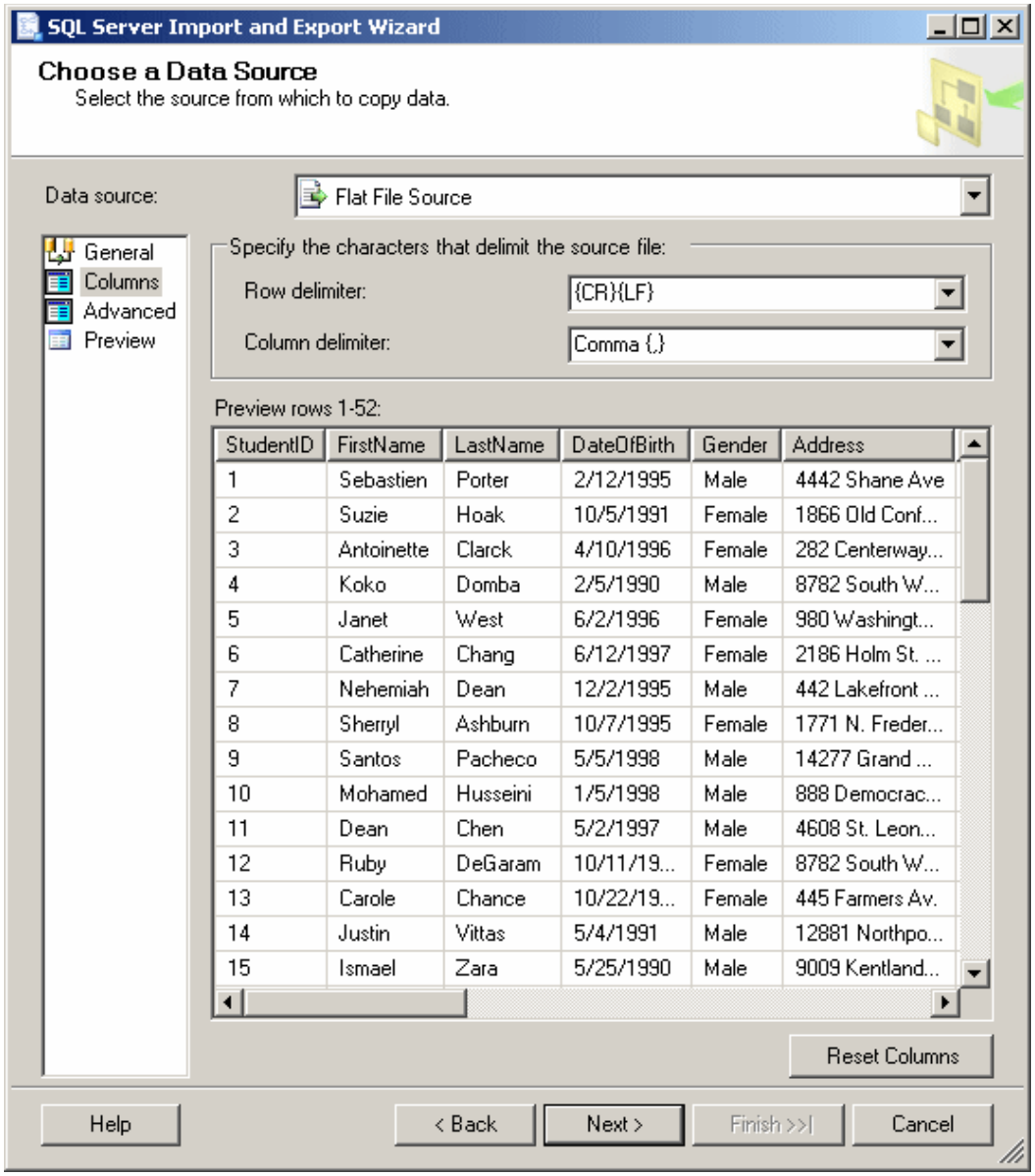
Columna	**Nombre**
Column0	StudentID
Column1	Nombre
Column2	Apellido
Column3	DateOfBirth
Column4	Género
Column5	Dirección
Column6	Ciudad
Column7	Estado
Column8	Código postal
Column9	HomePhone
Column10	Dirección de E-mail

Column11	ParentsNames
Column12	SPHome
Column13	EmrgName
Column14	EmrgPhone

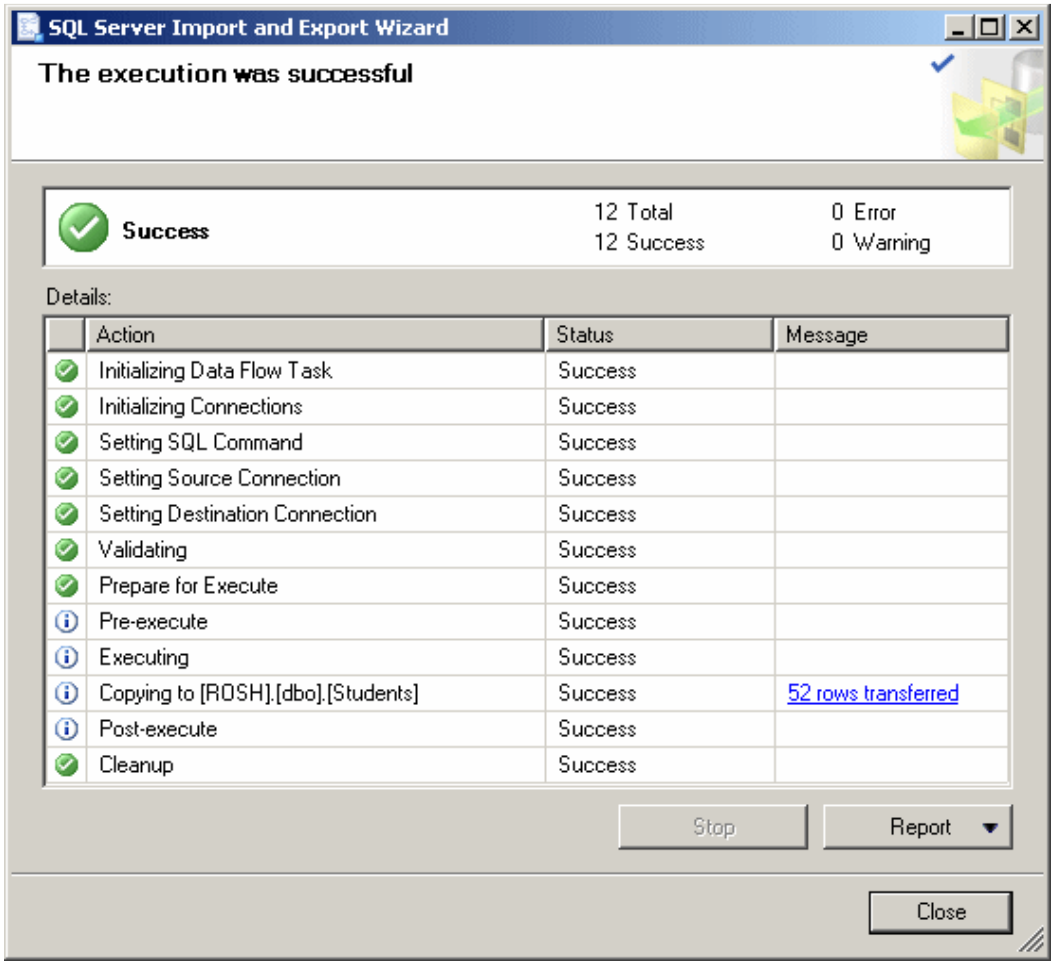
12.



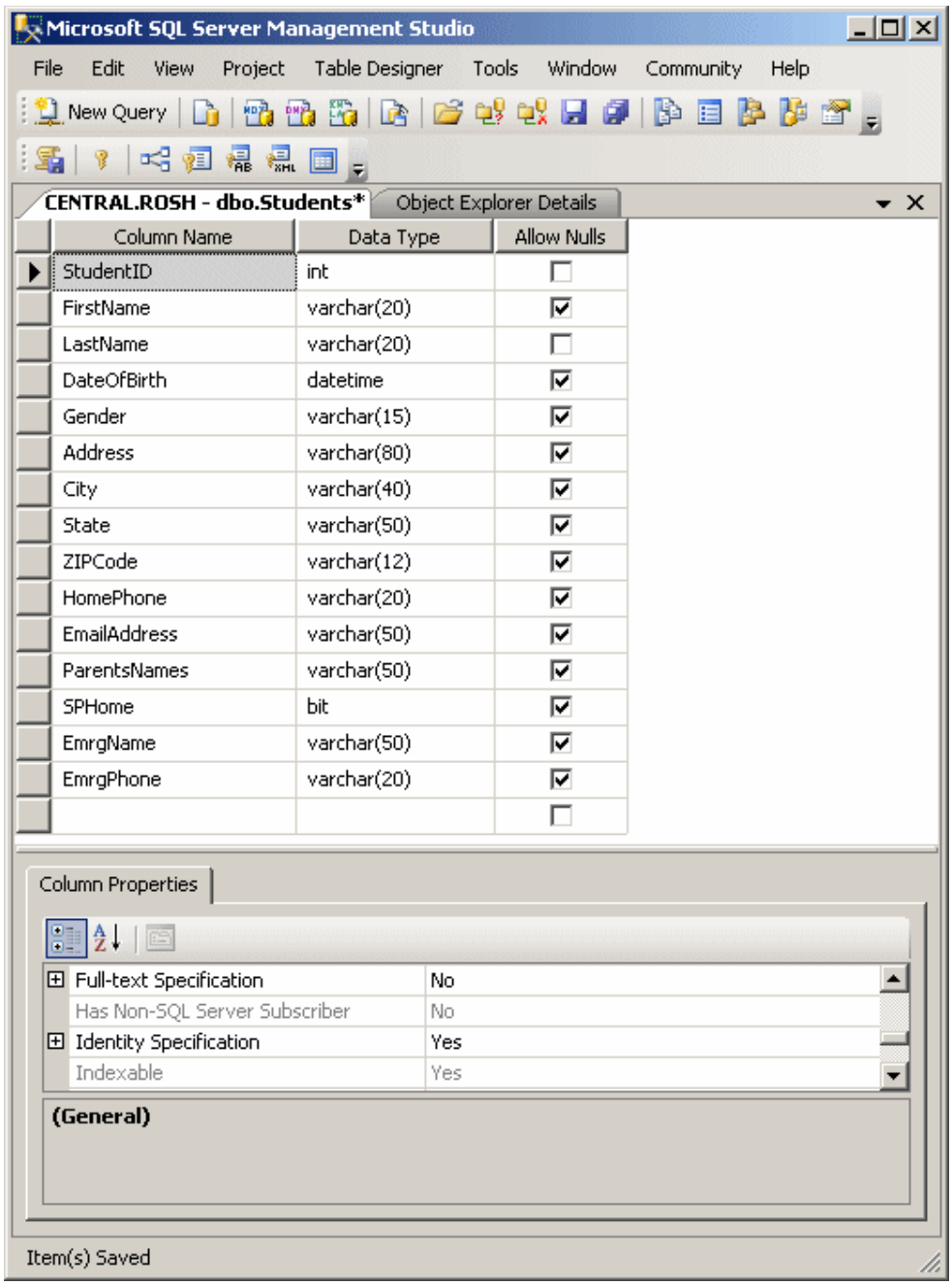
13. Para ver la lista de columnas, bajo Fuente, haga clic en Columnas



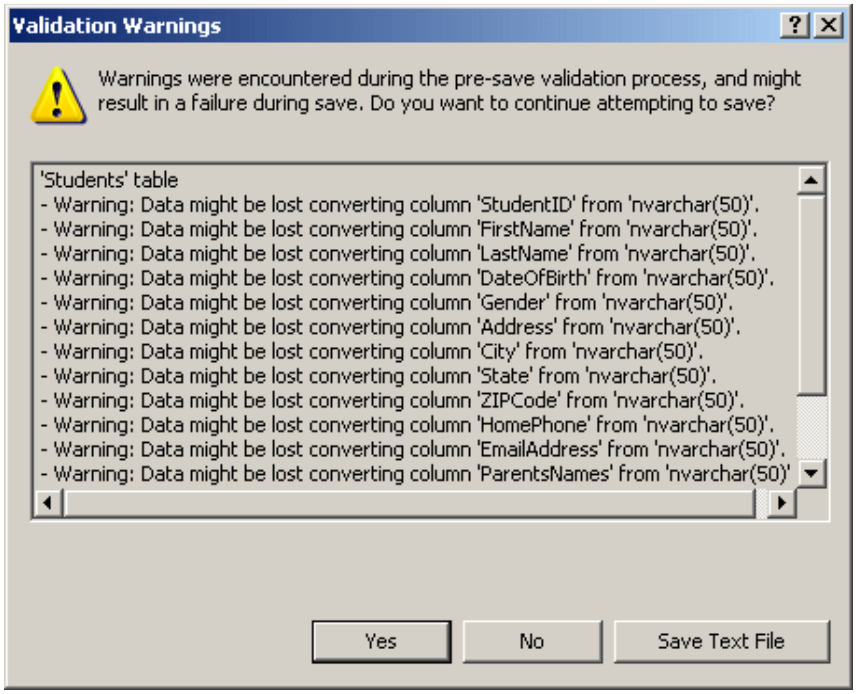
14. Haga clic en Siguiente 4 veces
15. Haga clic en Finalizar



16. Haga clic en Cerrar
17. Ya en el Explorador de objetos, expanda el Rosh cuadros y sus nodos.
    Haga clic con el botón y haga clic en los estudiantes de diseño
18. Como el **StudentID** campo seleccionado, pulse Tab y cambiar su tipo de datos de **int**
19. Presione F6 y ampliar la especificación de la identidad. Doble clic **(identidad)** para ajustar su valor a Sí
20. Cambiar las otras columnas de la siguiente manera:



21. Para guardar la tabla, haga clic en el botón Guardar en la barra de herramientas estándar:



22. Cuando un cuadro de diálogo Validación Advertencias presenta algunas advertencias, haga clic en Sí.
23. Cierre la tabla
24. Para ver los datos almacenados en la tabla, en el Explorador de objetos, haga clic derecho en dbo.Students y haga clic en Abrir tabla

Comprobación de registros

Comprobación de la existencia de un registro

Una de las operaciones más simples que un usuario puede realizar consiste en una mesa de buscar un registro. Para ello, el usuario abra la tabla que contiene los registros y comprobar visualmente en busca de un pedazo de información, como un estudiante de apellido.

Como desarrollador de la base de datos, usted también puede buscar un registro y hay varias técnicas que puede utilizar. Para ayudarle con esto, de Transact-SQL proporciona una función llamada **EXISTE.** Su sintaxis es:

BIT EXISTS (SELECT *Algo)*

Esta función tiene un argumento. El argumento debe ser un **SELECT** que se utiliza para obtener el valor, cuya existencia se verificó. Por ejemplo, en la **Lección 2,** que menciona un sistema de base de datos los nombres de **bases de datos** que contiene un registro de todas las bases de datos almacenadas en su servidor. Puede utilizar la **EXISTE ()** para comprobar la existencia de una base de datos. La fórmula que se utiliza es la siguiente:

```
IF EXISTS (
    SELECT name
    DE sys.databases
    WHERE name = N 'DatabaseName'
)
```

DatabaseName en el marcador de posición, puede introducir el nombre de la base de datos.

Selección de Documentos

Visualmente antes de la realización de algunas operaciones sobre una tabla, primero debe seleccionar uno o más registros. En el cuadro ventana, para seleccionar un disco, la posición del ratón sobre el botón izquierdo del registro y haga clic en:

Table - dbo.Videos | CENTRAL.Video...QLQuery1.sql* | Summary

VideoID	VideoTitle	Director	YearReleased	VideoLength	Rating
1	A Few Good Men	Rob Reiner	1992	138 Minutes	R
2	The Silence of the Lambs	Jonathan Demme	1991	118 Minutes	R
3	The Distinguished Gentleman	James Groeling	NULL	112 Minutes	R
4	The Lady Killers	Joel Coen & Eth...	NULL	104 Minutes	R
5	Ghosts of Mississippi	Rob Reiner	NULL	130 Minutes	R
NULL	NULL	NULL	NULL	NULL	NULL

Para seleccionar un rango de registros, haga clic en el botón gris de uno de los registros, mantenga pulsada la tecla Mayús y, a continuación, haga clic en el botón gris del registro en el otro extremo.

Para seleccionar los registros de forma aleatoria, seleccione un registro, mantenga pulsada la tecla Ctrl y, a continuación, haga clic en el botón gris de cada registro que desee:

Table - dbo.Videos | CENTRAL.Video...QLQuery1.sql* | Summary

VideoID	VideoTitle	Director	YearReleased	VideoLength	Rating
1	A Few Good Men	Rob Reiner	1992	138 Minutes	R
2	The Silence of the Lambs	Jonathan Demme	1991	118 Minutes	R
3	The Distinguished Gentleman	James Groeling	NULL	112 Minutes	R
4	The Lady Killers	Joel Coen & Eth...	NULL	104 Minutes	R
5	Ghosts of Mississippi	Rob Reiner	NULL	130 Minutes	R
NULL	NULL	NULL	NULL	NULL	NULL

Para seleccionar todos los registros de una tabla, haga clic en el botón gris a la izquierda de la primera columna:

Table - dbo.Videos | CENTRAL.Video...QLQuery1.sql* | Summary

VideoID	VideoTitle	Director	YearReleased	VideoLength	Rating
1	A Few Good Men	Rob Reiner	1992	138 Minutes	R
2	The Silence of the Lambs	Jonathan Demme	1991	118 Minutes	R
3	The Distinguished Gentleman	James Groeling	NULL	112 Minutes	R
4	The Lady Killers	Joel Coen & Eth...	NULL	104 Minutes	R
5	Ghosts of Mississippi	Rob Reiner	NULL	130 Minutes	R
NULL	NULL	NULL	NULL	NULL	NULL

Para modificar visualmente una o más registros en una tabla, en primer lugar abrirlo (usted haga clic derecho en la tabla de objetos en el Explorer y haga clic en Abrir el cuadro) para ver sus registros. Localice el registro y el campo que desea trabajar y llevar a cabo la operación deseada.

Mantenimiento de registros

Introducción

Mantenimiento de registros incluye la visualización de los registros en busca de uno o más registros, la modificación de uno o más registros, o la supresión de uno o más registros.

La actualización de un registro

La actualización de un registro consiste en el cambio de su valor para una columna. Para actualizar un registro utilizando SQL:

- En el Explorador de objetos, puede derecha de la mesa, coloca el ratón sobre el cuadro de comandos de Como -> ACTUALIZACIÓN A -> Nueva ventana del editor de Consulta
- Abra una ventana de consulta vacía y escriba su código

Para apoyar las operaciones de mantenimiento de registros, la proporciona **la** instrucción UPDATE de SQL palabra clave que se utiliza para especificar la tabla sobre la que desea mantener el registro (s). La fórmula básica para usar es la siguiente:

ACTUALIZACIÓN *TableName*
AJUSTE *ColumnName* = *Expresión*

Con esta fórmula, debe especificar el nombre de la tabla como la que participan *TableName* factor de nuestra fórmula. **El** comando SET permite especificar un nuevo valor, *expresión,* para el campo en el marco del *ColumnName* columna.

Considere el siguiente código para crear una nueva base de datos llamada VideoCollection y añadir una tabla con el nombre que se le Vídeos:

```
CREATE DATABASE VideoCollection;
IR
USO VideoCollection;
IR
CREATE TABLE Vídeos (
   VideoID INT NOT NULL IDENTIDAD (1,1),
   VideoTitle varchar (120) NOT NULL,
   Director varchar (100) NULL,
   YearReleased SMALLINT,
   VideoLength varchar (30) NULL,
   Calificación varchar (6)
);
IR
INSERT INTO Vídeos (VideoTitle, Director de YearReleased, VideoLength)
VALUES ( 'Algunos hombres buenos',' Rob Reiner, 1992,'138 Minutos');

INSERT INTO Vídeos (VideoTitle, Director de YearReleased, VideoLength)
VALUES ( 'El silencio de los corderos',' Jonathan Demme, 1991,'118 Minutos');

INSERT INTO Vídeos (VideoTitle, Director de VideoLength)
VALUES ( 'El Caballero Distinguido', 'James Groeling', 112 Minutos);

INSERT INTO Vídeos (VideoTitle, Director de VideoLength)
VALUES ( 'La Dama Asesinos',' Joel Coen y Ethan Coen ','104 Minutos');

INSERT INTO Vídeos (VideoTitle, Director de VideoLength)
VALUES ( 'Fantasmas de Mississippi "," Rob Reiner »,« 130 Minutos');
IR
```

La actualización de todos los registros

Imagina que, al mismo tiempo, en una tabla, todos los registros deben recibir un nuevo valor

en virtud de un particular, la columna o de ciertas columnas. No hay manera visual a actualizar todos los registros de una tabla. Usted puede abrir la tabla para ver sus registros y, a continuación, uno a uno.

En SQL, la primera fórmula de la **UPDATE** introducido en nuestra fórmula se puede utilizar para actualizar todos los registros. Aquí está un ejemplo:

```
USO VideoCollection;
IR
ACTUALIZACIÓN Vídeos
AJUSTE Puntuación = "R";
IR
```

Con este código, todos los registros de la tabla de Vídeos tendrá sus campos de Calificación establecido en un valor de R:

Table - dbo.Videos | CENTRAL.Video...QLQuery1.sql* | Summary

VideoID	VideoTitle	Director	YearReleased	VideoLength	Rating
1	A Few Good Men	Rob Reiner	1992	138 Minutes	R
2	The Silence of the Lambs	Jonathan Demme	1991	118 Minutes	R
3	The Distinguished Gentleman	James Groeling	NULL	112 Minutes	R
4	The Lady Killers	Joel Coen & Eth...	NULL	104 Minutes	R
5	Ghosts of Mississippi	Rob Reiner	NULL	130 Minutes	R
* NULL	NULL	NULL	NULL	NULL	NULL

Cómo editar un registro

Edición de un registro consiste en cambiar un valor en un campo. Podría ser que el campo está vacío, como el Año de la © los "asesinos de La Dama de vídeo de la siguiente tabla. Podría ser que el valor es incorrecto, como el Director de la la «Distinguido Señor de vídeo de esta tabla:

Título del vídeo	**Director**	**© Año**	**Longitud**	**Calificación**
Algunos hombres buenos	Rob Reiner	1992	138 Minutos	I
El silencio de los corderos	Jonathan Demme	1991	118 minutos	
El Distinguido Señor	James Groeling		112 Minutos	I
La Dama asesinos	Joel Coen y Ethan Coen		104 Minutos	I
Fantasmas de Mississippi	Rob Reiner		130 minutos	

Para editar un registro, en primer lugar abrir la tabla para ver sus registros. Localice el registro, la columna sobre la que desea trabajar, y localice el valor que desea cambiar, y después cambiarlo.

En SQL, debe proporcionar un camino para el intérprete de localizar el registro. Para ello, se asocian el operador en **caso de una** sentencia UPDATE mediante la siguiente fórmula:

ACTUALIZACIÓN *TableName*

AJUSTE *ColumnName = Expresión*
DÓNDE *Estado (s)*

Cuando el operador le permite especificar la forma en que el particular registro de participantes a efectos de identificación. Es muy importante, en la mayoría de los casos, que el criterio utilizado sea capaz de identificar de forma exclusiva el registro. En el cuadro anterior, imagine que usted le pide al intérprete en libertad para cambiar el año de 1996 cuando el director del vídeo es Rob Reiner. La sentencia UPDATE se escribiría como sigue:

ACTUALIZACIÓN Vídeos
AJUSTE YearReleased = 1996
DÓNDE Director = 'Rob Reiner;

En el cuadro anterior, hay al menos dos vídeos dirigido por Rob Reiner. Cuando se ejecuta esta declaración, todos los registros de vídeo cuyo director es Rob Reiner sería cambiado, lo que pondría en peligro los registros existentes que no era necesario este cambio. Aquí es donde la identidad se convierte en la columna valor. Vimos anteriormente que, cuando se la usa con la **identidad** característica, el intérprete agrega un valor único a cada registro. A continuación, puede utilizar este valor para identificar a un registro, ya que estás seguro de que el valor es único.

Aquí hay un ejemplo usado para especificar la falta de derechos de autor un año de registro:

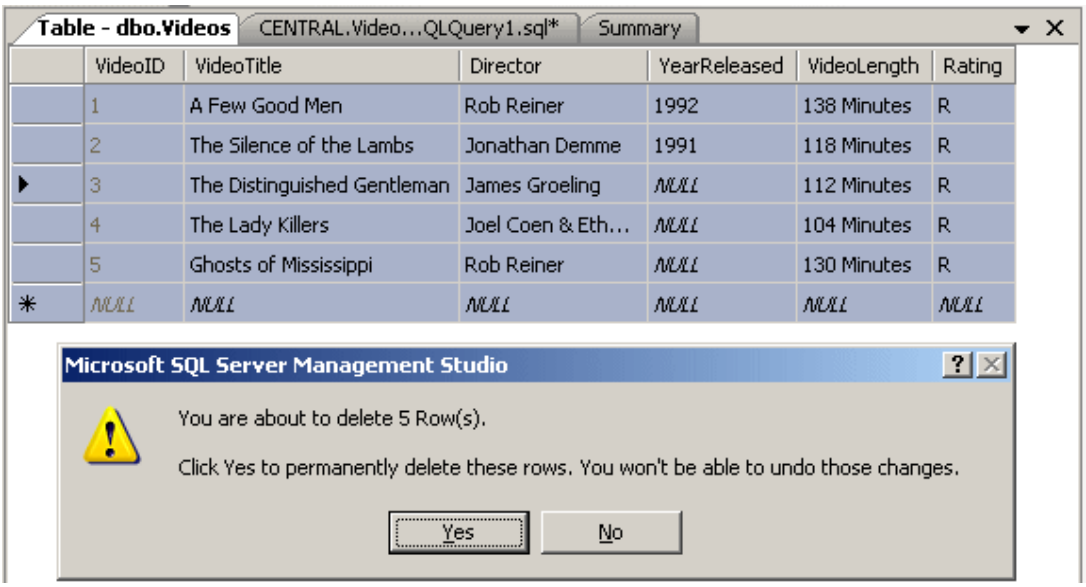
ACTUALIZACIÓN Vídeos
AJUSTE YearReleased = 1996
DÓNDE VideoID = 5;
IR

Aquí está un ejemplo utilizado para cambiar el nombre del director de un video:

ACTUALIZACIÓN Vídeos
AJUSTE Director = 'Jonathan Lynn'
DÓNDE VideoTitle = 'El Distinguido Señor »;

Eliminar todos los registros

Si usted piensa que todos los registros de una tabla, o se han convertido en inútil, puede borrar toda la tabla, que mantienen su estructura. Para borrar todos los registros de una tabla, seleccione primero la totalidad de ellas, y pulse Suprimir. Usted recibirá una advertencia:



Si aún desea borrar los registros, haga clic en Sí. Si cambia de opinión, haga clic en No.

Uso de SQL, para borrar una tabla de todos los registros, utilice el **operador** delete con la siguiente fórmula:

ELIMINAR *TableName;*

Cuando se ejecuta esta declaración, todos los registros de la *TableName* factor sería removido de la mesa. Tenga cuidado al hacer esto porque una vez que los registros se han suprimido, no se puede obtener de vuelta.

La eliminación de un registro

Si usted encuentra que un registro no es necesario, no más, o está fuera de lugar, puede eliminarlo de una tabla. Para eliminar un registro de una tabla, puede hacer clic derecho en el cuadro gris y haga clic en Eliminar. Puede también seleccionar primero el disco y pulse Suprimir. Usted recibirá una advertencia para confirmar su intención.

Para borrar un registro utilizando SQL:

- En el Explorador de objetos, puede derecha de la mesa, coloca el ratón sobre el cuadro de comandos de Como -> ELIMINAR A -> Nueva ventana del editor de Consulta
- Abra una ventana de consulta vacía y escriba su código

En SQL, para eliminar un registro, utilice la declaración **DELETE FROM** asociar el **caso de** operador. La fórmula a seguir es:

DELETE FROM *TableName*
DÓNDE *Estado (s)*

TableName es el factor utilizado para identificar una tabla cuyo registro (s) sería removido.

El *Estado (s)* de los factores permite identificar un registro o un grupo de registros que lleva a un criterio. Una vez más, asegúrese de que son precisos en sus criterios de modo que no se elimine el mal registro (s).

Aquí está un ejemplo utilizado para eliminar un registro de la tabla:

DELETE FROM Vídeos
DÓNDE VideoTitle = 'La Dama Asesinos';

Aquí hay un ejemplo usado para limpiar la mesa de todos los vídeos:

DELETE FROM Vídeos;

Aprendizaje práctico: Acabar con la Lección

1. Cierre la ventana de consulta sin guardar el archivo
2. En el Explorador de objetos, de conformidad con el nodo Bases de datos, haga clic derecho y haga clic en Eliminar WorldStatistics
3. En el cuadro de diálogo, haga clic en Aceptar

Resumen de la lección

Temas de Revisión

Temas Comentarios

- Registro
- Fila
- Cuadro de navegación

- Visual de introducción de datos
- De introducción de datos de SQL
- Adyacentes de introducción de datos
- De introducción de datos al azar
- Valores por defecto
- Columnas de identidad
- Expresiones
- Compruebe Limitaciones
- Cotejo
- Importación de datos
- Selección de Documentos
- Edición de registros
- Actualización de Registros
- Borrado de registros

Palabras clave, operadores, y Propiedades

- **NULL**
- **NOT NULL**
- **PREDETERMINADO**
- **IDENTIDAD**
- Especificación de la identidad
- (Identidad)
- Semillas de identidad de la propiedad
- Incremento de identidad
- **CONSTRAINT**
- **CHEQUE**
- Cotejo
- **bases de datos**
- **EXISTE**
- **ACTUALIZACIÓN**
- **BORRAR**

Ejercicios

Utilidad de la empresa

1. En Microsoft SQL Server Management Studio, acceder a la base de datos UtilityCompany1
2. Abrir la tabla Employees y entrar en unos registros

EmployeeNumber	Nombre	Apellido	Título
	Robert	Anson	
	Justine	Teclas	

	Edward	Kirkland	
	Kimberly	Eisner	
	Jonathan	Adamson	
	Steve	Fox	
	Andrew	Distritos	

3. Abra la tabla Clientes y la creación de unos registros

AccountNumber	**DateAccountCreated**	**CustomerName**	**Dirección**	**Ciudad**	**Estado**	**Dirección de E-mail**

Estados de EE.UU.

1. Consigue tu documentos de investigación en EE.UU. regiones y sus estados
2. Conectarse al servidor desde la línea de comandos y acceder a la base de datos UnitedStatesRegions1
3. Introduzca los nombres de las regiones en las Regiones de mesa
4. Introduzca los nombres de los estados en los Estados de mesa
5. Salga del símbolo del

Fundamentos de Análisis de Datos

Introducción

Después de crear y rellenar una tabla con registros, puede que desee ver lo que puedes hacer con los datos almacenados en ella. Una de las operaciones más comunes realizadas por los usuarios de una base de datos es la búsqueda de datos o para aislar a los datos que responde a un criterio particular. En busca de datos que se ajusten a un criterio que se conoce como consultas. El resultado de la recuperación de los datos basados en un criterio que se llama una consulta.

Como un desarrollador de base de datos, realizar consultas mediante la aprobación de instrucciones para el motor de base de datos. Esto se hace utilizando algunas palabras reservadas.

En Microsoft SQL Server, el análisis de los datos se pueden realizar utilizando SQL Server Management Studio o en una ventana de consulta.

Los datos en la ventana de la tabla

Para analizar los datos visualmente, en el Explorador de objetos, puede hacer clic derecho en una me
Abrir tabla. Aquí está un ejemplo:

Table - dbo.Videos | Summary

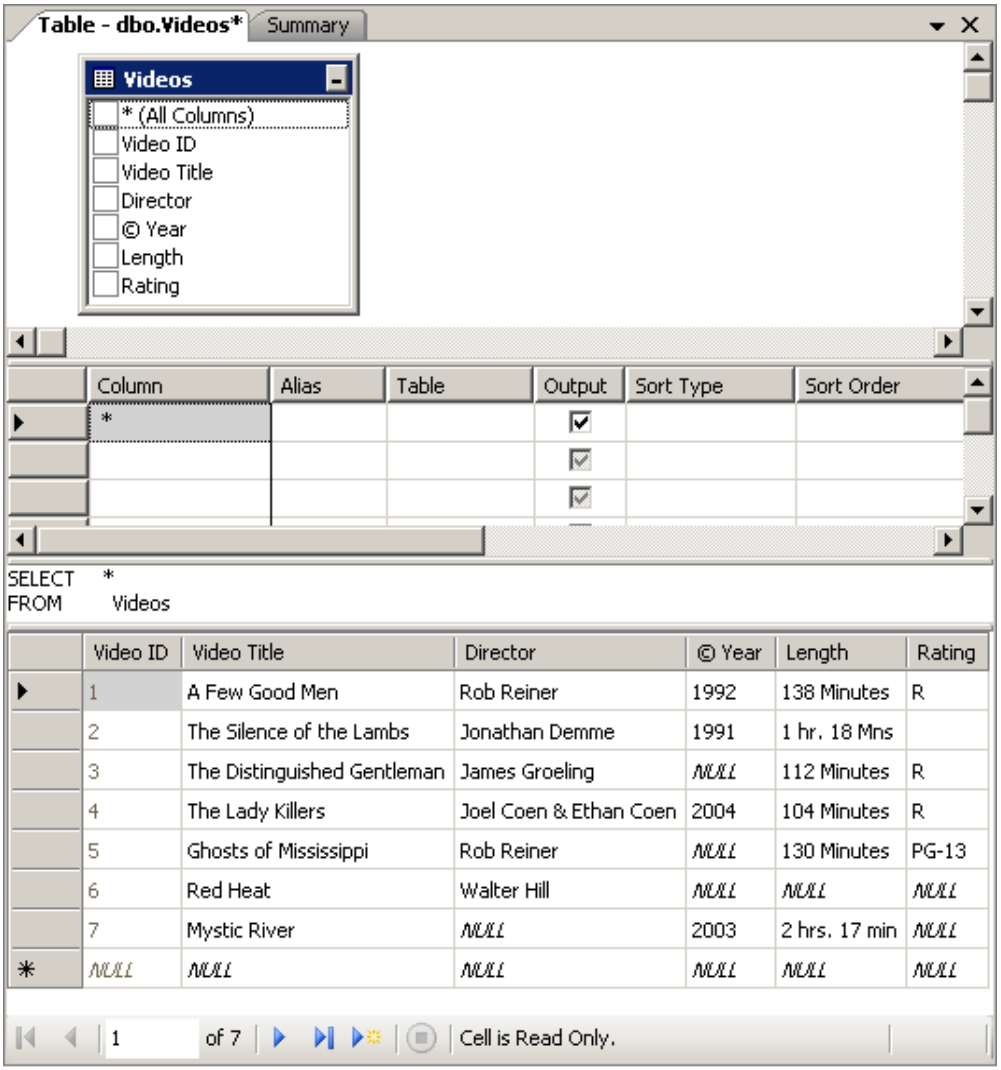
	Video ID	Video Title	Director	© Year	Length	Rating
▶	1	A Few Good Men	Rob Reiner	1992	138 Minutes	R
	2	The Silence of the Lambs	Jonathan Demme	1991	1 hr. 18 Mns	
	3	The Distinguished Gentleman	James Groeling	NULL	112 Minutes	R
	4	The Lady Killers	Joel Coen & Ethan Coen	2004	104 Minutes	R
	5	Ghosts of Mississippi	Rob Reiner	NULL	130 Minutes	PG-13
	6	Red Heat	Walter Hill	NULL	NULL	NULL
	7	Mystic River	NULL	2003	2 hrs. 17 min	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL

1 of 7 | Cell is Read Only.

De forma predeterminada, al abrir una tabla, la barra de herramientas Diseñador de consultas aparece t

Una vez que el cuadro se abre, en el menú principal, puede hacer clic en Diseñador de consultas. Si l
hacer clic derecho en cualquier lugar de la tabla. En ambos casos

1. En el menú que aparece, la posición del ratón y haga clic en Panel de Diagrama
2. Una vez más, abra el menú y haga clic en Panel de Criterios
3. De nuevo, abra el Panel de menú y haga clic en SQL:



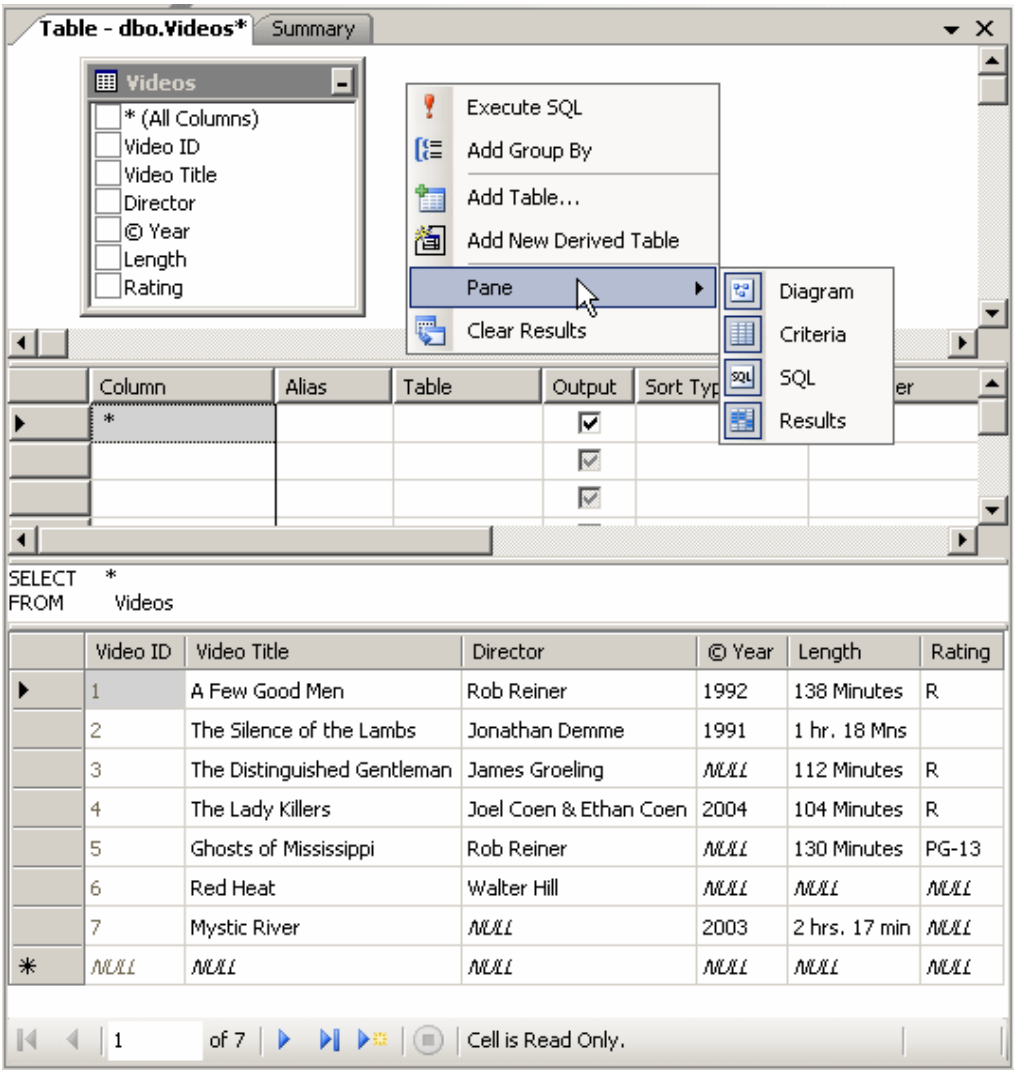
Como alternativa, en la barra de herramientas Diseñador de consultas, puede hacer clic en el bot
Diagrama , El botón Mostrar panel de criterios , Y el botón Mostrar panel SQL .

El cuadro ventana se divide en cuatro secciones:

1. Diagrama: La sección superior se refiere a la ventana del diagrama. Se muestra la tabla (s) que c
las columnas que desea consultar. Cada columna muestra una casilla de verificación sobre su izqui
de la columna de la derecha. El primer punto de la lista tiene un nombre de un asterisco y el (Toda
de expresión.
Si la lista de temas es demasiado larga para el rectángulo asignados para mostrar, el cuadro sería
una barra de desplazamiento vertical.
2. Criterios: Bajo la mesa, la segunda sección se denomina Criterios. Se muestra una lista de las colu
para construir visualmente la sentencia SQL.
3. SQL: En la tercera sección de arriba, llamado SQL, se muestra la instrucción SQL que los resultado
selecciones en el diagrama o los criterios secciones
4. Resultados: La sección inferior, denominado Resultados, muestra el resultado producido por la sen
cuando se ejecuta

Si no desea que una sección o desea ocultar algunas secciones, puede hacer clic derecho en cualquier

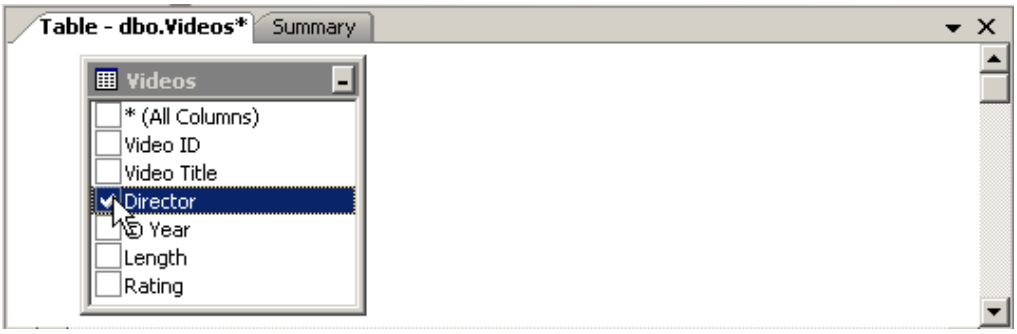
coloca el ratón sobre panel y haga clic en el nombre de la sección. Para ocultar cualquier sección, derecho en cualquier lugar de la ventana, coloque el mouse sobre Mostrar paneles y haga clic en una de



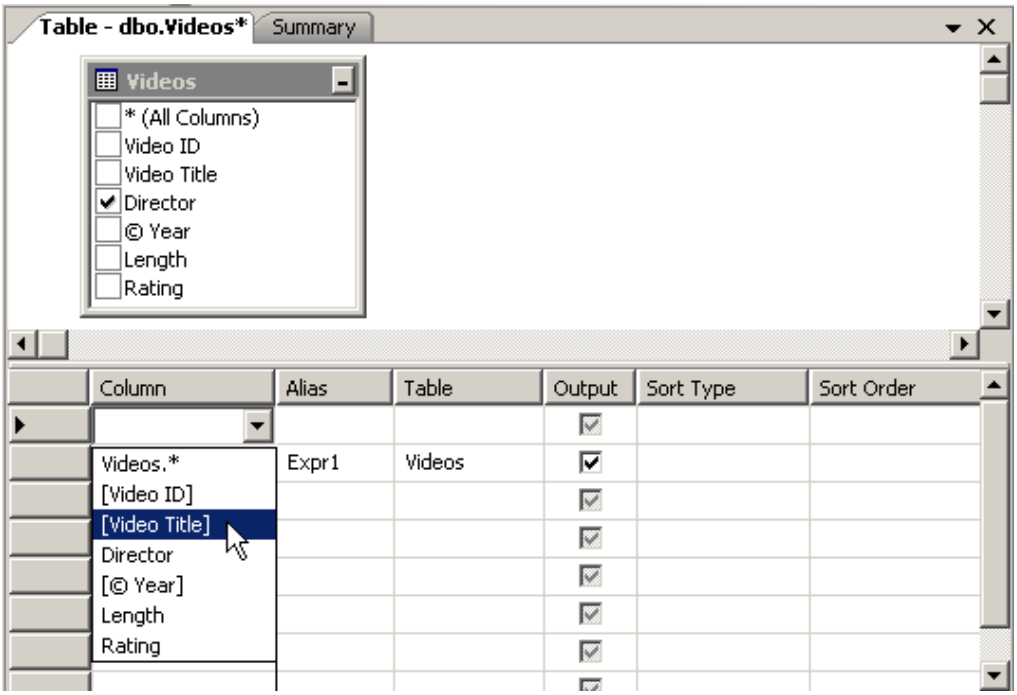
Cuando una sección se muestra, su opción de menú está rodeado.

Selección de la columna

Una sentencia SQL está construido principalmente por la selección de una o más columnas cuyos datos seleccionar una columna, en el diagrama de la sección de la tabla ventana, puede hacer clic en la casi en el lado izquierdo del nombre:



Después de hacer clic en la casilla de verificación de una columna, se convierte en los criterios sel sección y su nombre también aparece en la sección de SQL. Otra técnica utilizada para seleccionar una en hacer clic en un recuadro que hay bajo el encabezado de la columna de la sección de Criterios. manifiesto que se trata de un cuadro combinado. Puede a continuación, haga clic en la flecha del cuadro mostrar la lista y seleccionar una columna de la lista:



Criterios en la sección, si hace clic en un cuadro combinado que contiene ya una columna, pero se anterior sería sustituido por uno nuevo. Además, después de seleccionar una columna en la sección convierte en su casilla de verificación seleccionado en el diagrama de la sección y su nombre se añadi SQL.

Si sabe el nombre de una columna que desea agregar, que se puede ver en el diagrama de la secc entrar directamente en la sentencia SQL.

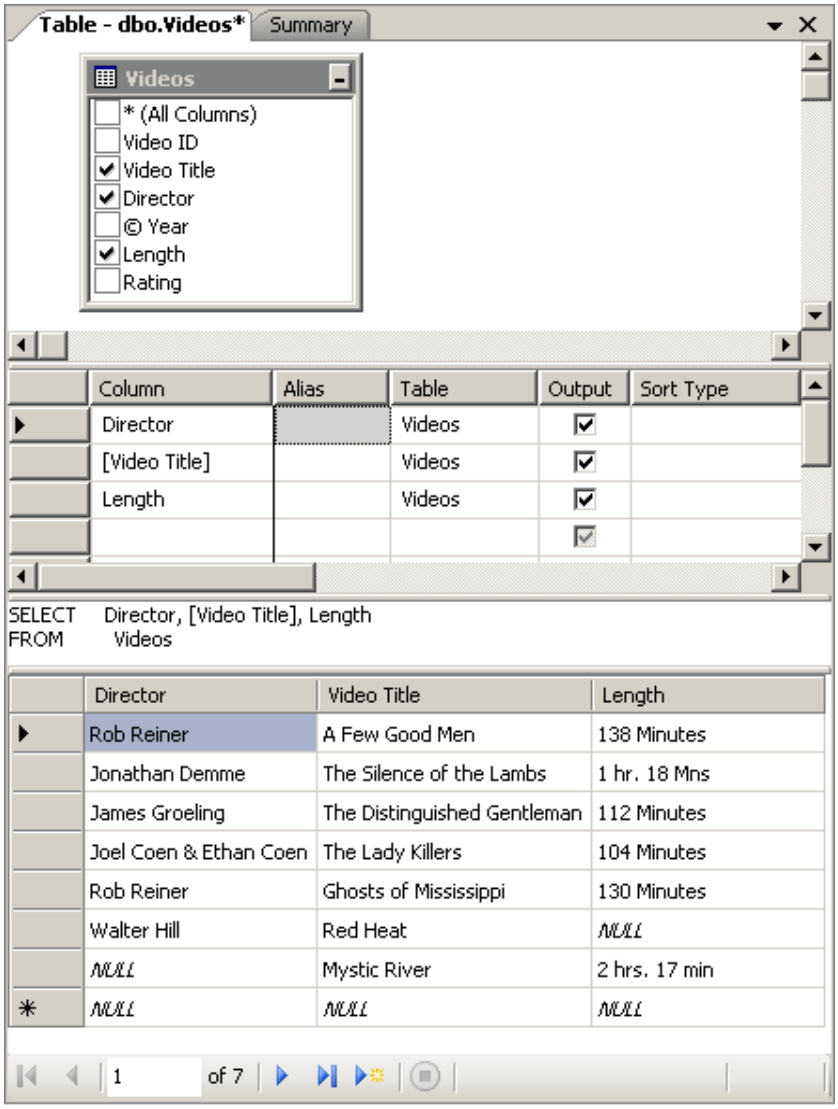
Cualquiera de estas tres técnicas permite seleccionar una o más columnas para construir la deseada SQ

Ejecución de sentencias SQL

Después de crear una instrucción SQL, puede ver su resultado, que puede obtener mediante la declaración. Para ello, puede hacer clic derecho en cualquier lugar de la tabla y haga clic en la ventana

otra parte, en la barra de herramientas, puede hacer clic en el botón Ejecutar SQL .

Después de ejecutar la sentencia, la sección inferior se llena sólo con los datos de la columna selec tabla. Aquí está un ejemplo:



Transact-SQL y Análisis de Datos

Introducción

Análisis de los datos es efectuado mediante código SQL que contiene uno o más criterios. Para pr análisis de datos, usted tiene varias opciones:

- Si aún no ha decidido qué tabla contiene los datos que desea analizar, en el Explorador de objetos, derecho en una base de datos y haga clic en Nueva consulta. Luego, en la ventana de consulta, es en el SQL
- Si conoce el cuadro que contiene los datos que desea analizar, en el Explorador de objetos, haga c cuadro, coloque el ratón sobre el cuadro de comandos de Como, seguido por **SELECCIONAR** A, y h Nueva ventana del editor de consultas. Una ventana de consulta se abrió con una muestra de códi

Aprendizaje práctico: La introducción de análisis de datos

1. Inicie Microsoft SQL Server con el SQL Server Management Studio y conectarse al servidor
2. En el menú principal, haga clic en Archivo -> Nuevo -> Consulta de conexión con los actuales
3. Para preparar una base de datos de nuestros ejercicios, escriba lo siguiente:

```
- ==============================================
- Autor: FunctionX
- Base de datos: RealEstate1
- ==============================================
IF EXISTS (
   SELECT *
      DE sys.databases
    WHERE name = N'RealEstate1 '
),
   DROP DATABASE RealEstate1
IR
CREATE DATABASE RealEstate1;
IR

- ==============================================
- Autor: FunctionX
- Base de datos: RealEstate1
- Tabla: Propiedades
- ==============================================
USO RealEstate1;
IR
CREATE TABLE Propiedades
(
   Int PropertyID identidad (1,1) NOT NULL,
   PropertyNumber char (6),
   Dirección varchar (100),
   Ciudad varchar (50),
   Estado char (2),
   Código postal varchar (12),
   PropertyType varchar (40),
   Condición varchar (32),
   Dormitorios smallint,
   Baños de flotación,
   FinishedBasement poco,
   IndoorGarage poco,
   Historias smallint,
   YearBuilt smallint,
   MarketValue dinero
);
IR

INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('524880 ','1640 Ave. Lombardo', 'Silver Spring', 'MD',
   '20904 ',' Única Familia ',' Buenas', 4, 2,5, 3, 1, 3, 1995, 495,880.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('688364 ','10315 Norte Hacht Rd', 'College Park', 'MD',
          '20747 ',' Única Familia ',' Excelente ', 4, 3.5, 3,
```

```
        1, 2, 2000, 620,724.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, FinishedBasement,
   Historias, MarketValue)
VALUES ('611464 ','6366 Lolita Drive', 'Laurel', 'MD',
        '20707 ',' Único Familiar "," buena ", 1, 2, 422,625.00);
IR
INSERT INTO Propiedades (Dirección, Ciudad, PropertyType,
   Dormitorios, MarketValue)
VALUES ('9002 Palasko Hwy », Tysons Corner,
        "Condominio", 2, 422,895.00);
IR
INSERT INTO Propiedades (PropertyNumber, Estado,
   Código postal, Dormitorios, YearBuilt, MarketValue)
VALUES ('420115 ',' CC ',
        '20011 ', 2, 1982, 312555);
IR
INSERT INTO Propiedades (PropertyNumber, ciudad, código postal,
  PropertyType, Dormitorios, YearBuilt, MarketValue)
VALUES ('917203 ',' Alejandría ','22024',
        «Unifamiliar ', 3, 1965, 345,660.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   PropertyType, en la condición, Dormitorios, Baños, MarketValue)
VALUES ('200417 ','4140 Holisto Crt', 'Germantown', 'MD',
        "Condominio", "Excelente", 2, 1, 215,495.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('927474 ','9522 Lockwood Road', 'Chevy Chase "," MD ",
        '20852 ',' Casa ',' mal ', 3, 2.5, 3, 0, 3,
        1992, 415.665,00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('207850 ','14250 Parkdoll Road', 'Rockville "," MD ",
        '20854 ',' Casa ',' Bueno ', 3, 2.5, 2, 1, 2,
        1988, 325.995,00);
IR
INSERT INTO Propiedades (Ciudad, PropertyType, Dormitorios,
   YearBuilt, MarketValue)
VALUES ( 'Washington', 'Casa', 4, 1975, 366,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   YearBuilt, MarketValue)
VALUES ('288540 ','10340 Helmes Street # 408', 'Silver Spring', 'MD',
        '20906 ',' Condominio ',' Bueno ', 1, 1, 2000, 242,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('247472 ','1008 Coppen Street', 'Silver Spring', 'MD',
```

```
        '20906 ',' Única Familia ',' Excelente ',
        3, 3, 3, 1, 3, 1996, 625,450.00);
IR
INSERT INTO Propiedades (ciudad, código postal, PropertyType,
    Historias, YearBuilt, MarketValue)
VALUES ( 'Chevy Chase','20956 ',' Único Familiar ",
        3, 2001, 525,450.00);
IR
INSERT INTO Propiedades (Dirección, Ciudad, Estado,
  PropertyType, en la condición, Dormitorios, MarketValue)
VALUES ('686 Herodes Ave. # D04 ',' Takoma Park ',' MD ',
        "Condominio", "Excelente", 2, 360,885.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
  Código postal, PropertyType, en la condición, Dormitorios, Baños,
  FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('297446 ','14005 Sniders Blvd.', 'Laurel', 'MD',
        '20707 ',' Casa ',' Necesidades de reparación ",
        4, 1.5, 3, 1, 2, 2002, 412,885.00);
IR
INSERT INTO Propiedades (Ciudad, Código postal, en la condición, Dormitorios,
  Historias, YearBuilt)
VALUES ( 'Silver Spring','20905 ',' Bueno ',
        4, 2, 1965);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
  Código postal, PropertyType, en la condición, Dormitorios, Baños,
  FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('924792 ','680 Prushia Road', 'Washington', 'CC',
        '20008 ',' Único Familiar "," buena ",
        5, 3.5, 3, 0, 3, 2000, 555,885.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
  Código postal, PropertyType, en la condición, Dormitorios, Baños,
  FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('294796 ','14688 Parrison Street', 'College Park', 'MD',
        '20742 ',' Única Familia ',' Excelente ',
        5, 2.5, 2, 1, 2, 1995, 485,995.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
  Código postal, PropertyType, en la condición, Dormitorios, Baños,
  YearBuilt, MarketValue)
VALUES ('811155 ','10340 Helmes Street # 1012', 'Silver Spring »,
        'MD','20906 ',' Condominio ',' Bueno ',
        1, 1, 2000, 252,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
  Código postal, PropertyType, en la condición, Dormitorios, Baños,
  FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('447597 ','4201 Vilamar Ave', 'Hyattsville', 'MD',
        '20782 ',' Casa ',' Excelente ',
        3, 2, 2, 1, 3, 1992, 365,880.00);
IR
INSERT INTO Propiedades (Dirección, código postal, Baños)
VALUES ('1622 Rombard Str., 20904, 2.5);
IR
```

```
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('297415 ','980 Phorwick Street', 'Washington', 'CC',
         '20004 ',' Único Familiar "," buena ",
         4, 3.5, 3, 3, 1, 2004, 735,475.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('475974 ','9015 Marvin Ave. Crow', 'Gaithersburg', 'MD',
         '20872 ',' Único Familiar "," Necesidades de reparación ",
         4, 2.5, 3, 1, 1, 1965, 615,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('836642 ','3016 Feldman Corte', 'Rockville "," MD ",
         '20954 ',' Única Familia ',' mal ',
         5, 3, 3, 1, 3, 1960, 528,555.00);
IR
INSERT INTO Propiedades (Ciudad, Estado, PropertyType, Historias)
VALUES ( 'Rockville "," MD ",
         'Casa', 1);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('208304 ','7307 Everett Hwy', 'Washington', 'CC',
         '20012 ',' Casa ',' Excelente ',
         2, 2.5, 2, 0, 4, 2006, 420,550.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios,
   Baños, YearBuilt, MarketValue)
VALUES ('644114 ','10340 Helmes Street # 1006', 'Silver Spring »,
         'MD','20906 ',' Condominio ',' Bueno ',
         2, 2, 2000, 258,445.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('937966 ','7303 Warfield Corte "," Tysons Corner', 'VA',
         '22131 ',' Único Familiar "," buena ",
         3, 2.5, 3, 1, 4, 2006, 825,775.00);
IR
INSERT INTO Propiedades (Ciudad, Código postal, en la condición, Dormitorios,
   Historias, YearBuilt)
VALUES ( 'Fairfax','22232 ',' Bueno ', 3, 3, 1985);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('297497 ','12401 Conniard Ave', 'Takoma Park', 'MD',
         '20910 ',' Casa ',' Bueno ',
         3, 2.5, 3, 1, 3, 2004, 280,775.00);
```

```
IR
INSERT INTO Propiedades (ciudad, código postal, PropertyType,
                Dormitorios, Baños, MarketValue)
VALUES ( 'Alejandría','22035 ',' Condominio ',
         2, 2, 425,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, ciudad, código postal,
   PropertyType, en la condición, Dormitorios, Baños,
   YearBuilt, Historias, MarketValue)
VALUES ('855255 ',' Laurel ','20707', 'Único Familiar ",
         "Necesidades de reparación", 3, 2, 1962, 2, 342,805.00);
IR
INSERT INTO Propiedades (PropertyNumber, ciudad, código postal,
PropertyType,
   Condición, Dormitorios, Baños, MarketValue)
VALUES ('225227 ',' Rockville ','20857', 'Condominio', 'Bueno',
         1, 1, 525,885.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('469750 ','6124 Falk Road', 'Arlington', 'VA',
         '22031 ',' Único Familiar "," Necesidades de reparación ",
         4, 3.5, 3, 1, 1, 1982, 635,995.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('826927 ','5121 Riehl As',' Fairfax ',' VA ',
         '22232 ',' Casa ',' Excelente ',
         3, 1.5, 2, 0, 1, 2002, 325,620.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('287064 ','9533 Pensulian Road', 'Silver Spring', 'MD',
         '20904 ',' Única Familia ',' mal ',
         3, 1.5, 3, 1, 2, 1992, 485,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, ciudad, código postal,
   PropertyType, en la condición, Dormitorios, YearBuilt, Historias)
VALUES ('724001 ','705 Helios Ave','20004 ',
         'Casa', 'mal', 3, 1974, 4);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('209275 ','944 Freidora Ave', 'Chevy Chase "," MD ",
         '20852 ',' Única Familia ',' Excelente ',
         5, 2.5, 3, 0, 2, 2002, 625,665.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('204759 ','1950 Galego Street', 'Germantown', 'MD',
         '20874 ',' Única Familia ',' Excelente ',
```

```
        4, 3.5, 2, 1, 4, 2007, 428,665.00);
IR
INSERT INTO Propiedades (Ciudad, Estado, PropertyType,
   Dormitorios, Baños, YearBuilt, MarketValue)
VALUES ( 'Takoma Park', 'MD',
        'Conbominium', 2, 2, 2000, 225,885.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyType, en la condición, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('937259 ','12366 Fowler Ave', 'Alejandría', 'VA',
        '22031 ',' Casa ',' Bueno ',
        3, 1.5, 3, 1, 3, 2007, 402,815.00);
IR
```

4. Presione la tecla F5 para ejecutar
5. Guarde el archivo como RealEstate1 y cerrar la ventana de consulta

Listado de los resultados de análisis de datos de SQL

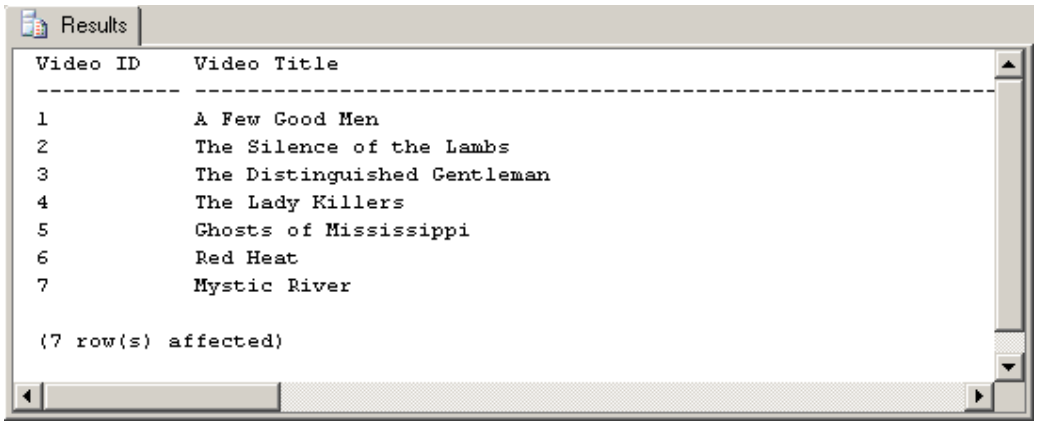
Después de escribir la sentencia SQL, se puede ejecutar para ver el resultado. Esto mostrará la vent resultado se muestra en la sección inferior. Hay dos maneras en que puede mostrar el resultado. Pa estas opciones, se puede mostrar la barra de herramientas del Editor SQL. Para mostrar la barra de SQL Editor:

- En el menú principal, puede hacer clic en Ver -> Barras de herramientas -> Editor de SQL
- Puede hacer clic derecho en cualquier barra de herramientas y haga clic en SQL Editor

Para especificar cómo desea mostrar los resultados de su sentencia SQL, tiene dos opciones:

- Para mostrar el resultado en forma de texto, en la barra de herramientas del Editor de SQL, puede botón de resultados al texto . Alternativamente, puede hacer clic derecho en algún lugar de la t ratón sobre Resultados y haga clic en Resultados al texto.

  Los resultados aparecen en dos columnas de texto. Aquí está un ejemplo:



- Para mostrar el resultado en una hoja de cálculo, sobre la barra de herramientas del Editor de SQL clic en el botón de resultados a la rejilla . Alternativamente, puede hacer clic derecho en algún l coloca el ratón sobre Resultados y haga clic en los resultados a la rejilla.

  Los resultados aparecen como una hoja de cálculo de una o varias columnas. Aquí está un ejempl

Results | Messages

	Video ID	Video Title	Director	© Y...	Length	Rating
1	1	A Few Good Men	Rob Reiner	1992	138 Minutes	R
2	2	The Silence of the Lambs	Jonathan Demme	1991	1 hr. 18 Mns	
3	3	The Distinguished Gentleman	James Groeling	NULL	112 Minutes	R
4	4	The Lady Killers	Joel Coen & Ethan Coen	2004	104 Minutes	R
5	5	Ghosts of Mississippi	Rob Reiner	NULL	130 Minutes	PG-13
6	6	Red Heat	Walter Hill	NULL	NULL	NULL
7	7	Mystic River	NULL	2003	2 hrs. 17 min	NULL

Ni en el cuadro de la ventana o ventana de consulta, se espera que escribir el código adecuado qu sentencia SQL.

Aprendizaje práctico: Creación de una consulta

1. En el Explorador de objetos, haga clic derecho en el nodo Bases de datos y haga clic en Actualizar
2. Expanda el nodo Bases de datos
3. Expandir RealEstate1
4. RealEstate1 clic con el botón derecho y haga clic en Nueva consulta

Campo de selección

El más fundamental de palabras clave utilizadas por SQL es **SELECT.** Con el fin de procesar un pedido qué seleccionar. Para realizar el análisis de los datos, la **selección** de palabras clave utiliza la sintaxis s

¿Qué WhatObject DE **SELECCIONAR;**

Como se dijo ya, SQL no es sensible a mayúsculas y minúsculas. Esto significa **SELECT, seleccione** y **seleccione** representan la misma palabra.

Para seleccionar todo, desde una tabla, puede utilizar el asterisco como el rango de valores. Por ejem todos los registros de una tabla llamada alumnos, puede escribir:

SELECT * FROM estudiantes;

Después de escribir la expresión, debe ejecutar la sentencia SQL para ver su resultado. Aquí está un ej

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT * FROM   Students;
GO
```

Results | Messages

	StudentID	FirstName	LastName	DateOfBirth	Gender	Address
1	1	Sebastien	Porter	1995-02-12 00:00:00.000	Male	4442 Shane A
2	2	Suzie	Hoak	1991-10-05 00:00:00.000	Female	1866 Old Conf
3	3	Antoinette	Clarck	1996-04-10 00:00:00.000	Female	282 Centerway
4	4	Koko	Domba	1990-02-05 00:00:00.000	Male	8782 South W
5	5	Janet	West	1996-06-02 00:00:00.000	Female	980 Washingto
6	6	Catherine	Chang	1997-06-12 00:00:00.000	Female	2186 Holm St.
7	7	Nehemiah	Dean	1995-12-02 00:00:00.000	Male	442 Lakefront
8	8	Sherryl	Ashburn	1995-10-07 00:00:00.000	Female	1771 N. Freder
9	9	Santos	Pacheco	1998-05-05 00:00:00.000	Male	14277 Grand M
10	10	Mohamed	Husseini	1998-01-05 00:00:00.000	Male	888 Democrac
11	11	Dean	Chen	1997-05-02 00:00:00.000	Male	4608 St. Leona
12	12	Ruby	DeGaram	1998-10-11 00:00:00.000	Female	8782 South W
13	13	Carole	Chance	1990-10-22 00:00:00.000	Female	445 Farmers A
14	14	Justin	Vittas	1991-05-04 00:00:00.000	Male	12881 Northpo
15	15	Ismael	Zara	1990-05-25 00:00:00.000	Male	9009 Kentland
16	16	Anselme	Waters	1990-07-23 00:00:00.000	Male	1872 Hunters D
17	17	Brenda	Lobo	1989-10-05 00:00:00.000	Female	2490 Chanting
18	18	Suzanna	Verde	1988-03-12 00:00:00.000	Female	3844 John Mcl
19	19	Danilo	Chico	1997-06-20 00:00:00.000	Male	291 Nixon Av.
20	20	Mincy	Franse	1991-10-08 00:00:00.000	Female	1788 Old Mont

Query... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 rows

También puede optar por el anterior * selector con el nombre de la tabla seguido por el período com operador. La declaración anterior es equivalente a:

SELECCIONE .* Los estudiantes de los estudiantes;

En la **Lección 8,** vimos que puede crear un alias para una tabla anterior por una columna con una letra un período de operador y, a continuación, introducir el nombre de la tabla seguido de la carta o de p función, la declaración anterior puede escribirse como:

SELECCIONE estd.. Estd. * de los estudiantes;

A diferencia de la visualización de todos los datos, también puede seleccionar una columna particular, c desea ver. Para ello, se puede sustituir el *¿Qué* en nuestra sintaxis con el nombre de la columna. P obtener una lista de apellidos de los estudiantes, que ejecutar la siguiente declaración:

SELECCIONAR Apellido de los estudiantes;
IR

También puede optar por una columna anterior con el nombre de la tabla seguido por el período com operador. La declaración anterior es equivalente a:

SELECCIONE Students.LastName de los estudiantes;

Al ejecutar la declaración, se mostrará solamente la columna que contiene los últimos nombres. A c
una columna en una declaración, puede en la lista *¿Qué* factor de nuestra sintaxis, separándolas con u
para la última columna. La sintaxis que se utiliza es la siguiente:

SELECT *column1, column2, Column_n* DE ***WhatObject;***

Por ejemplo, para mostrar una lista que incluye únicamente los nombres, sexo, dirección de correo elec
de los registros de una tabla llamada alumnos, debería escribir:

Seleccionar nombre, apellido, sexo, dirección de E-mail, HomePhone
De los estudiantes;

Una vez más, usted puede optar por cada columna anterior con el nombre de la tabla seguid
comprendido entre el operador. La declaración anterior es equivalente a:

SELECCIONE estudiantes. Nombre,
Estudiantes. Apellido,
Estudiantes. Género,
Estudiantes. Direcciónde correo electrónico,
Estudiantes. HomePhone
De los estudiantes;

Usted no tiene que calificar todas las columnas, puede calificar a unos y otros no califican. La declar
equivalente a:

SELECCIONE estudiantes. Nombre,
Apellido,
Estudiantes. Género,
Dirección de E-mail,
HomePhone
De los estudiantes;

Cuando se ejecuta, esta expresión se producen:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT FirstName, LastName, Gender,
       EmailAddress, HomePhone
FROM Students;
GO
```

Results | Messages

	StudentID	FirstName	LastName	DateOfBirth	Gender	Address
1	1	Sebastien	Porter	1995-02-12 00:00:00.000	Male	4442 Shane A
2	2	Suzie	Hoak	1991-10-05 00:00:00.000	Female	1866 Old Conf
3	3	Antoinette	Clarck	1996-04-10 00:00:00.000	Female	282 Centerway
4	4	Koko	Domba	1990-02-05 00:00:00.000	Male	8782 South W
5	5	Janet	West	1996-06-02 00:00:00.000	Female	980 Washingto
6	6	Catherine	Chang	1997-06-12 00:00:00.000	Female	2186 Holm St.
7	7	Nehemiah	Dean	1995-12-02 00:00:00.000	Male	442 Lakefront
8	8	Sherryl	Ashburn	1995-10-07 00:00:00.000	Female	1771 N. Freder
9	9	Santos	Pacheco	1998-05-05 00:00:00.000	Male	14277 Grand M
10	10	Mohamed	Husseini	1998-01-05 00:00:00.000	Male	888 Democrac
11	11	Dean	Chen	1997-05-02 00:00:00.000	Male	4608 St. Leona
12	12	Ruby	DeGaram	1998-10-11 00:00:00.000	Female	8782 South W
13	13	Carole	Chance	1990-10-22 00:00:00.000	Female	445 Farmers A
14	14	Justin	Vittas	1991-05-04 00:00:00.000	Male	12881 Northpo
15	15	Ismael	Zara	1990-05-25 00:00:00.000	Male	9009 Kentland
16	16	Anselme	Waters	1990-07-23 00:00:00.000	Male	1872 Hunters D
17	17	Brenda	Lobo	1989-10-05 00:00:00.000	Female	2490 Chanting
18	18	Suzanna	Verde	1988-03-12 00:00:00.000	Female	3844 John McI

Query... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 rows

Una vez más, recuerde que usted puede utilizar un nombre de alias para una tabla anterior por cada letra o una palabra y un período de operador y, a continuación, introducir el nombre de la tabla seguido palabra. Aquí está un ejemplo:

SELECCIONE estd.. FirstName, estd.. Apellido, estd.. Género, estd.. Direcciónde correo electrónico, estd.. Estd. de los estudiantes;

Aprendizaje práctico: Selección de datos

1. En la ventana de consulta, escriba lo siguiente:

   USO RealEstate1;
   IR
   SELECCIONE p. * FROM propiedades p;
   IR

2. Presione la tecla F5 para ejecutar la declaración

3. Para obtener una lista de las propiedades y los números de su tipo, el cambio de la declaración de manera:

   SELECCIONE dbo.Properties.PropertyNumber,
   dbo.Properties.PropertyType
   De Propiedades;

IR

4. Presione la tecla F5 para ejecutar
5. Para obtener una lista de los tipos de propiedades, cada año se construyó, y su valor de mercado, declaración de la siguiente manera:

```
SELECCIONE house.PropertyNumber,
      house.PropertyType,
      house.YearBuilt,
      house.MarketValue
Propiedades de la casa;
IR
```

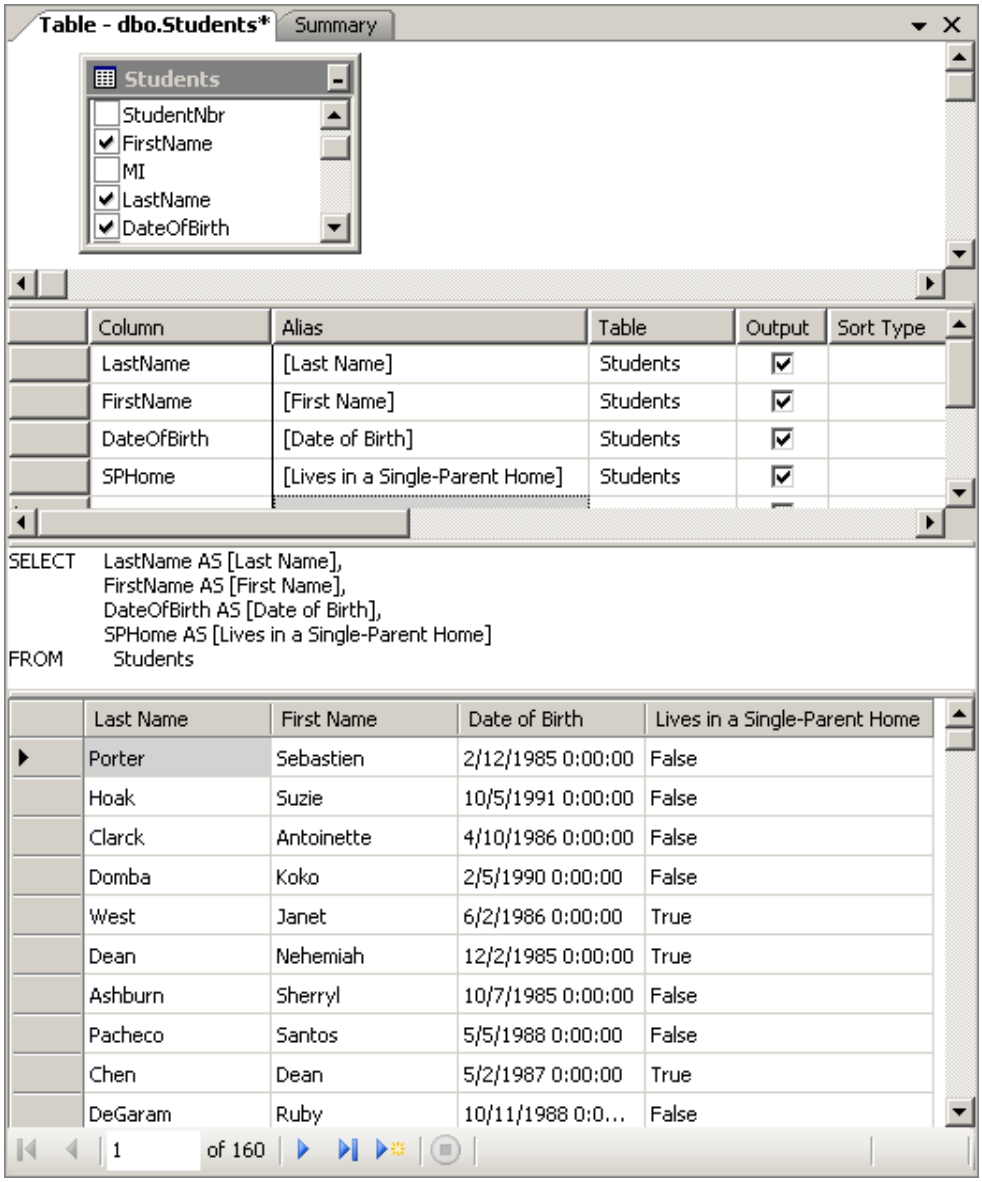
6. Clic con el botón derecho en algún lugar de la tabla, coloca el ratón sobre Resultados y haga clic en texto
7. Presione la tecla F5 para ejecutar

El uso de un nombre de alias de una columna

Introducción

En su **sentencia SELECT,** después de especificar la columna (s) como lo hemos hecho hasta ahora, c la sentencia SQL, el nombre de cada columna que aparecerá como el encabezado de la columna. A usted puede mostrar cualquier cadena de su elección para un encabezado de columna.

Para especificar un encabezado de columna que no sea el nombre de la columna, si está utilizando tabla, escriba el alias de la cadena en la columna correspondiente a la columna. Aquí está un ejemplo:



Si está utilizando una ventana de consulta o si está escribiendo el **comando SELECT,** en la parte derec
nombre, el **tipo** seguido por el nombre de la cabecera de la columna. Si el encabezado de columna es
sólo tiene que escribirlo. Aquí está un ejemplo:

```
SELECCIONE FirstName,
        Apellido,
        COMO HomePhone número de teléfono,
        ParentsNames COMO NamesOfParents
De los estudiantes;
IR
```

Si desea que el encabezado de columna para aparecer con más de una palabra, puede proporcionar la
una cadena en una sola cita o entre los corchetes: [y]. Aquí está un ejemplo:

```
SELECCIONE COMO Nombre [Nombre],
        Apellido COMO [Apellido],
        COMO HomePhone [Teléfono],
```

COMO ParentsNames [Nombres de los padres]
De los estudiantes;
IR

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT FirstName AS [First Name],
       LastName AS [Last Name],
       HomePhone AS [Phone Number],
       ParentsNames AS [Names of Parents]
FROM   Students;
GO
```

Results | Messages

	StudentID	FirstName	LastName	DateOfBirth	Gender	Address
1	1	Sebastien	Porter	1995-02-12 00:00:00.000	Male	4442 Shane A
2	2	Suzie	Hoak	1991-10-05 00:00:00.000	Female	1866 Old Conf
3	3	Antoinette	Clarck	1996-04-10 00:00:00.000	Female	282 Centerway
4	4	Koko	Domba	1990-02-05 00:00:00.000	Male	8782 South W
5	5	Janet	West	1996-06-02 00:00:00.000	Female	980 Washingto
6	6	Catherine	Chang	1997-06-12 00:00:00.000	Female	2186 Holm St.
7	7	Nehemiah	Dean	1995-12-02 00:00:00.000	Male	442 Lakefront
8	8	Sherryl	Ashburn	1995-10-07 00:00:00.000	Female	1771 N. Freder
9	9	Santos	Pacheco	1998-05-05 00:00:00.000	Male	14277 Grand M
10	10	Mohamed	Husseini	1998-01-05 00:00:00.000	Male	888 Democrac
11	11	Dean	Chen	1997-05-02 00:00:00.000	Male	4608 St. Leona
12	12	Ruby	DeGaram	1998-10-11 00:00:00.000	Female	8782 South W
13	13	Carole	Chance	1990-10-22 00:00:00.000	Female	445 Farmers A
14	14	Justin	Vittas	1991-05-04 00:00:00.000	Male	12881 Northpo
15	15	Ismael	Zara	1990-05-25 00:00:00.000	Male	9009 Kentland
16	16	Anselme	Waters	1990-07-23 00:00:00.000	Male	1872 Hunters D
17	17	Brenda	Lobo	1989-10-05 00:00:00.000	Female	2490 Chanting
18	18	Suzanna	Verde	1988-03-12 00:00:00.000	Female	3844 John Mcl

Query... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 rows

Calificación por cada columna, el anterior también se puede escribir como sigue:

SELECCIONE estudiantes. Nombre COMO [Nombre],
Estudiantes. Apellido COMO [Apellido],
Estudiantes. HomePhone COMO [Teléfono],
Estudiantes. ParentsNames COMO [Nombres de los padres]
De los estudiantes;
IR

También puede ser escrito como sigue:

SELECCIONE dbo. Students.FirstName COMO [Nombre],
dbo. Students.LastName COMO [Apellido],
dbo. Students.HomePhone COMO [Teléfono],
dbo. Students.ParentsNames COMO [Nombres de los padres]

De los estudiantes;
IR

También puede ser escrito como sigue:

SELECCIONE estd.. Nombre COMO [Nombre],
estd.. Apellido COMO [Apellido],
estd.. HomePhone COMO [Teléfono],
estd.. ParentsNames COMO [Nombres de los padres]
Estd. de los estudiantes;
IR

Aprendizaje práctico: El uso de alias

1. Para utilizar el alias de nombres de columnas, cambiar el estado de la siguiente manera:

   SELECCIONE COMO house.PropertyNumber [Prop. #],
   COMO house.ZIPCode [Ubicación]
   COMO house.YearBuilt [Año de construcción],
   COMO house.PropertyType [Tipo],
   COMO house.Bedrooms [Camas],
   COMO house.Bathrooms [Baños],
   COMO house.MarketValue [valor]
   Propiedades de la casa;
   IR

2. Clic con el botón derecho en algún lugar de la tabla, coloca el ratón sobre Resultados y haga clic en la rejilla

3. Presione la tecla F5 para ejecutar

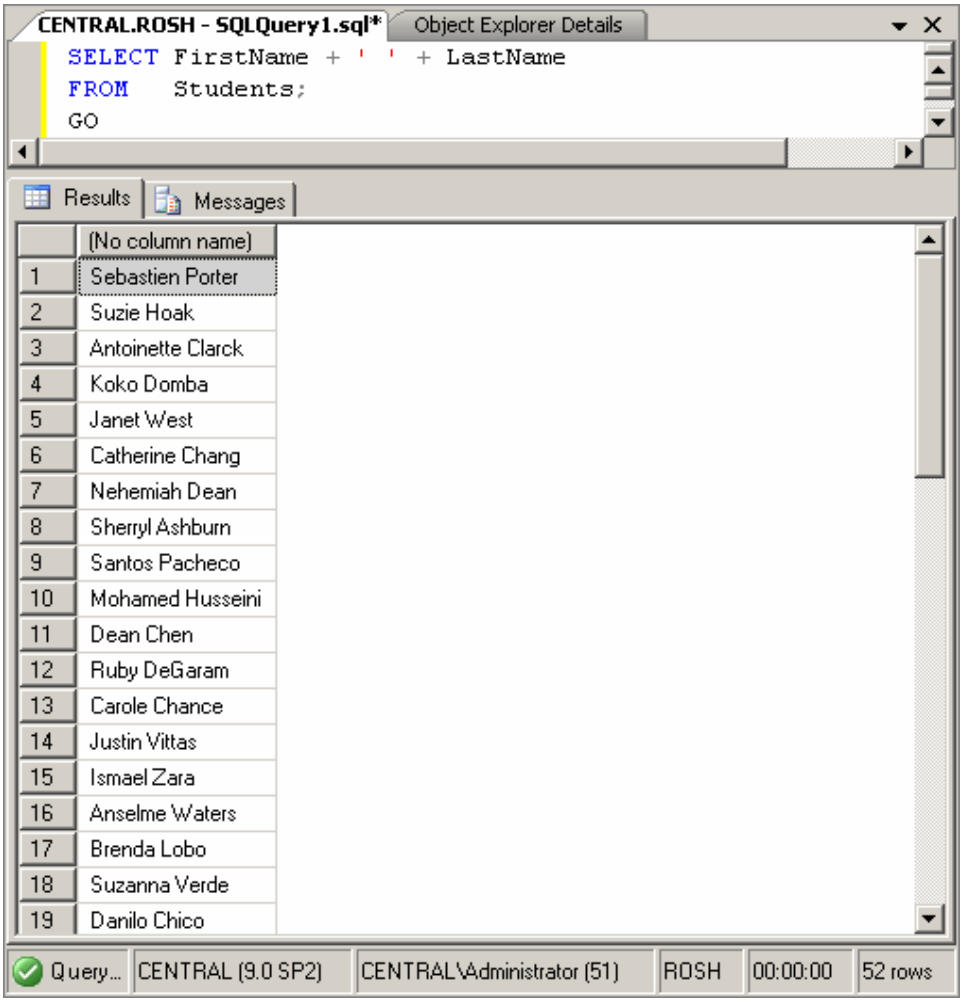
Una combinación de expresión o de las Columnas

El uso de **la** palabra reservada SELECT, hemos aprendido a crear una lista de columnas aisladas. En e quedaron sin separar el uno del otro. En lugar de tener columnas separadas, puede combinarlos para o un valor que es, de hecho, una expresión. Por ejemplo, puede combinar un nombre y un apell completo como una expresión. Otra expresión puede usar una fecha en el cuadro, añadir un número a una fecha en otro día. Una expresión también puede ser usado para realizar un cálculo en dos o más co empleados las horas semanales multiplicado por su sueldo por hora para obtener su salario semanal.

El más común utilizado es el operador de adición. Se puede utilizar para combinar dos o más hilos nueva. Aquí está un ejemplo:

SELECCIONE Nombre + '' + Apellido
De los estudiantes;
IR

Esto produciría:



La adición también se pueden usar en los valores numéricos. Todos los demás operadores aritmé utilizados. Por ejemplo, se puede multiplicar un valor de horas semanales a una hora para obtener un s sueldos. La declaración de este tipo de expresión puede ser escrita como sigue:

SELECCIONE WeeklyHours * HourlySalary
DE Nómina

También puede crear un alias para una expresión para darle el nombre deseado. Para ello, en el la expresión, como el tipo seguido por el nombre. Como hemos aprendido anteriormente, si el alias es palabra, o bien incluirla en comillas simples o corchetes. Aquí está un ejemplo:

SELECCIONE Nombre + "" + Apellidos como 'Nombre completo',
EmrgName + "" + EmrgPhone COMO [Contacto de Emergencia]
De los estudiantes;
IR

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT FirstName + ' ' + LastName AS 'Full Name',
       EmrgName + ' ' + EmrgPhone AS [Emergency Contact]
FROM   Students;
GO
```

Results | Messages

	Full Name	Emergency Contact
1	Sebastien Porter	John Porter (202) 662-4825
2	Suzie Hoak	Christianne Hoak (301) 364-0221
3	Antoinette Clarck	Henriette Clarck (301) 598-9292
4	Koko Domba	Jeremiah Domba
5	Janet West	Kirsten Farmer
6	Catherine Chang	Ernie Change (202) 215-6663
7	Nehemiah Dean	Marie Rodnat (703) 434-8756
8	Sherryl Ashburn	Shelia Ashburn (703) 292-8340
9	Santos Pacheco	(202) 938-7772
10	Mohamed Husseini	Dr. Phyllis Husseini (202) 631-8855
11	Dean Chen	Alexandra Chen
12	Ruby DeGaram	Anthony DeGaram (301) 938-1220
13	Carole Chance	Andy Chance (301) 422-1001
14	Justin Vittas	(301) 549-0020
15	Ismael Zara	Robert Zara (202) 978-6642
16	Anselme Waters	Celestin Waters (703) 894-6624
17	Brenda Lobo	(703) 894-2888
18	Suzanna Verde	Thommy Verde

Query... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 rows

El operador de asignación

Si sólo crear una expresión regular utilizando operadores aritméticos, la nueva columna no tiene permite especificar un nombre diferente para cualquier columna durante el análisis de los datos o un expresión. Esto se realiza utilizando el operador de asignación "=".

Para cambiar el nombre de una columna durante el análisis de datos, en el lado derecho de **sele** nombre deseado, seguido por el operador de asignación, seguido por el nombre real de la column ejemplo:

SELECCIONE EmergencyName = EmrgName
De los estudiantes;
IR

Si desea utilizar más de una columna, escriba cada uno y asignarle el nombre deseado, sepárelos con un ejemplo:

SELECCIONAR Apellido,
EmergencyName = EmrgName,
EmergencyPhone = EmrgPhone
De los estudiantes;
IR

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT LastName,
       EmergencyName = EmrgName,
       EmergencyPhone = EmrgPhone
FROM   Students;
GO
```

Results | Messages

	Full Name	Emergency Contact
1	Sebastien Porter	John Porter (202) 662-4825
2	Suzie Hoak	Christianne Hoak (301) 364-0221
3	Antoinette Clarck	Henriette Clarck (301) 598-9292
4	Koko Domba	Jeremiah Domba
5	Janet West	Kirsten Farmer
6	Catherine Chang	Ernie Change (202) 215-6663
7	Nehemiah Dean	Marie Rodnat (703) 434-8756
8	Sherryl Ashburn	Shelia Ashburn (703) 292-8340
9	Santos Pacheco	(202) 938-7772
10	Mohamed Husseini	Dr. Phyllis Husseini (202) 631-8855
11	Dean Chen	Alexandra Chen
12	Ruby DeGaram	Anthony DeGaram (301) 938-1220
13	Carole Chance	Andy Chance (301) 422-1001
14	Justin Vittas	(301) 549-0020
15	Ismael Zara	Robert Zara (202) 978-6642
16	Anselme Waters	Celestin Waters (703) 894-6624
17	Brenda Lobo	(703) 894-2888
18	Suzanna Verde	Thommy Verde

Query... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 rows

También puede incluir el nombre entre comillas o solo los corchetes. Aquí están algunos ejemplos:

SELECCIONAR Apellido + ',' + Nombre COMO [Nombre Completo],
[Nombre de Emergencia] = EmrgName,
"Teléfono de emergencia '= EmrgPhone
De los estudiantes;
IR

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```sql
SELECT LastName + ', ' + FirstName AS [Full Name],
       [Emergency Name] = EmrgName,
       'Emergency Phone' = EmrgPhone
FROM   Students;
GO
```

Results | Messages

	Full Name	Emergency Name	Emergency Phone
1	Porter, Sebastien	John Porter	(202) 662-4825
2	Hoak, Suzie	Christianne Hoak	(301) 364-0221
3	Clarck, Antoinette	Henriette Clarck	(301) 598-9292
4	Domba, Koko	Jeremiah Domba	
5	West, Janet	Kirsten Farmer	
6	Chang, Catherine	Ernie Change	(202) 215-6663
7	Dean, Nehemiah	Marie Rodnat	(703) 434-8756
8	Ashburn, Sherryl	Shelia Ashburn	(703) 292-8340
9	Pacheco, Santos		(202) 938-7772
10	Husseini, Mohamed	Dr. Phyllis Husseini	(202) 631-8855
11	Chen, Dean	Alexandra Chen	
12	DeGaram, Ruby	Anthony DeGaram	(301) 938-1220
13	Chance, Carole	Andy Chance	(301) 422-1001
14	Vittas, Justin		(301) 549-0020
15	Zara, Ismael	Robert Zara	(202) 978-6642
16	Waters, Anselme	Celestin Waters	(703) 894-6624
17	Lobo, Brenda		(703) 894-2888
18	Verde, Suzanna	Thommy Verde	

Query... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 rows

Los detalles de Análisis de Datos

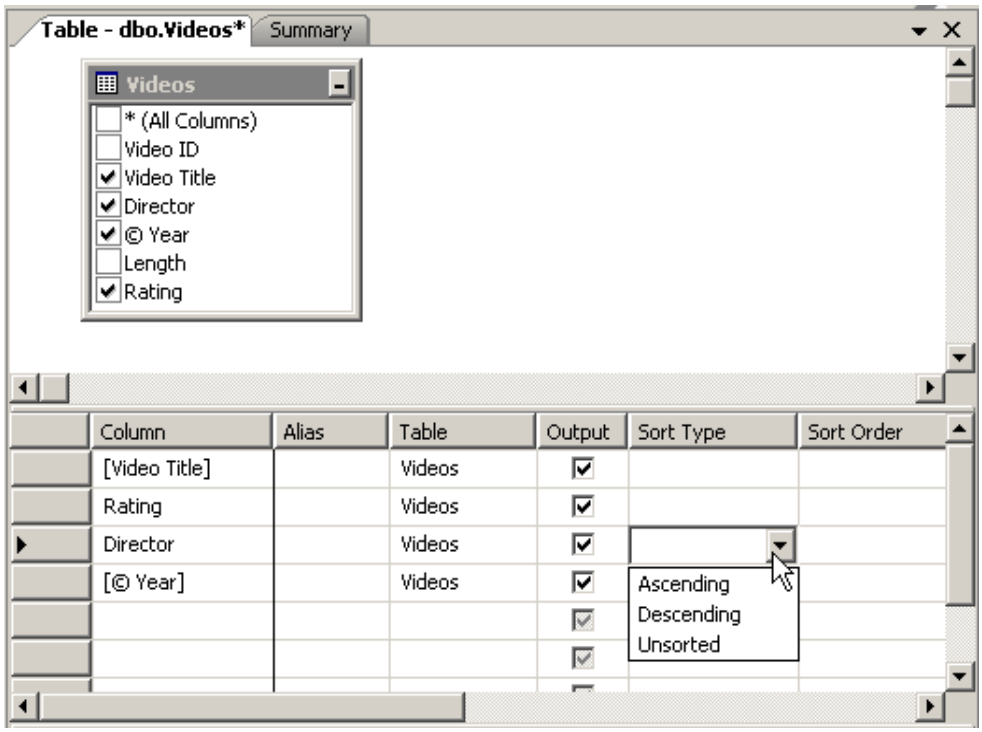
Clasificación de las Actas

Clasificación de los registros en la tabla Ventana

Las listas de los documentos que recibimos con un **SELECT** se presentan en el orden que tienen en permite organizar los registros en orden alfabético, por orden cronológico o en orden numérico increme seleccionar una serie de columnas, es posible que desee a la lista de los registros a raíz de una orden a de un campo específico. Para obtener una orden alfabético o de una orden de incremento de los regis saber que la base de datos de campo se utilizará como referencia.

Para especificar el orden, si está utilizando una tabla ventana:

- En el diagrama de la sección, usted puede hacer clic derecho en un campo y seleccione Ordenar Or Ascendente o Descendente
- Criterios en la sección de la ventana, bajo la columna Tipo Ordenar, haga clic en la casilla correspo columna deseada. Esto pondría de manifiesto que se trata de un cuadro combinado. A continuació flecha del cuadro combinado y que haga su selección entre ascendente y descendente:



Si selecciona Ascendente Ordenar o Ascendente, la lista de registros se volverá a organizarse en func columna seleccionada:

- Si la columna está basada en texto **(CHAR, VARCHAR,** y sus variantes), los registros se organizar alfabético
- Si la columna es la fecha o la hora a base de **(datetime** o **smalldatetime),** los registros se organ cronológico
- Si la columna es el número de base, los registros se organizan en orden incremental
- Si la columna está basada en booleano (bits), las actas FALSE aparecerá en primer lugar

Si selecciona Ordenar Descendente o Descendente, la lista de los registros se volverá a organizarse e

de la columna seleccionada:

- Si la columna está basada en texto **(CHAR, VARCHAR,** y sus variantes), los registros se organizar alfabético inverso
- Si la columna es la fecha o la hora a base de **(datetime** o **smalldatetime),** los registros se organ cronológico inverso
- Si la columna es el número de base, los registros se organizan en orden de decrecimiento
- Si la columna está basada en booleano (bits), el verdadero actas aparecen primero

Después de seleccionar el tipo Ordenar, puede ejecutar la sentencia SQL.

Aprendizaje práctico: Condiciones de Uso de Análisis de Datos

1. Inicie Microsoft SQL Server y SQL Server Management Studio.
   Si aún no, crear la **base de datos RealEstate1**
2. En el Explorador de objetos, haga clic derecho en las bases de datos y haga clic en Nueva consulta

Ordenar los registros en el SQL

En SQL, para especificar el orden, la utilización de la expresión **ORDER BY.** La sintaxis utilizada sería la

Seleccione *lo WhatObject* DE ORDEN POR *WhatField;*

La columna utilizada como la base debe ser reconocido como parte de las columnas seleccionadas. P obtener una lista de los estudiantes en orden alfabético sobre la base de la columna Apellidos, puede ut declaración:

```
SELECCIONE FirstName,
          Apellido,
          Género,
          ParentsNames,
          SPHome
De los estudiantes
ORDER BY Apellidos;
IR
```

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT FirstName,
       LastName,
       Gender,
       ParentsNames,
       SPHome
FROM Students
ORDER BY LastName;
GO
```

Results | Messages

	FirstName	LastName	Gender	ParentsNames	SPHome
1	Tim	Amorros	Male	Eduardo Amoros	0
2	Arlene	Andriamirano	Female	Christianne Burton	1
3	Sherryl	Ashburn	Female	Shelia and Patrick Ashburn	0
4	Martha	Bastens	Female		0
5	Lester	Bell	Male	Bernadette and Paulin Bell	0
6	Millicent	Broadskey	Female		0
7	Carole	Chance	Female	Carole and Andy Chance	0
8	Catherine	Chang	Female	Julie Best and Dr. Peter Chang	0
9	Dean	Chen	Male	Alexandra Chen	1
10	Danilo	Chico	Male	Michelle and Stephen Chico	0
11	Antoinette	Clarck	Female	Daniel and Henriette Clarck	0
12	Martin	Davis	Male		0
13	Nehemiah	Dean	Male	Marie Rodnat	1
14	Ruby	DeGaram	Female	Lynn and Anthony DeGaram	0
15	Koko	Domba	Male	Dr. Julie and Mr. Jeremiah Domba	0
16	Arlette	Duma	Female	Ursula Duma	1
17	Charles	Edelman	Male	Jasmine Beaulieu	1
18	Paul	Farms	Male	Rebecca and Peter Farms	0

Query... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 rows

De la misma manera, puede obtener la lista de las niñas seguido por la lista de los niños, ordenando alfabético basado en el género. La declaración de producir este puede ser escrito como sigue:

Seleccionar nombre, apellido, sexo, dirección de E-mail
De los estudiantes
ORDER BY de Género;
IR

Como otro ejemplo, para una lista de todos los estudiantes en orden alfabético por el apellido, p declaración de la siguiente manera:

SELECT * FROM estudiantes
ORDER BY Apellidos;
IR

De forma predeterminada, los archivos están ordenados en orden ascendente. Sin embargo, el orde controla mediante la palabra clave **ASC** especificado tras la orden sobre el terreno. Por ejemplo, últimos nombres en orden ascendente incluyendo el nombre y el apellido, debería utilizar una declaraci

manera:

```
SELECT * FROM estudiantes
ORDER BY Apellido      ASC;
IR
```

Por otro lado, si desea ordenar los registros en orden inverso, puede usar la palabra clave **DESC** lugar resultado opuesto a la **ASC** efecto. He aquí un ejemplo:

```
SELECCIONE FirstName,
          Apellido,
          Género,
          ParentsNames,
          SPHome
De los estudiantes
ORDER BY Apellidos DESC;
IR
```

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT FirstName,
       LastName,
       Gender,
       ParentsNames,
       SPHome
FROM Students
ORDER BY LastName DESC;
GO
```

Results | Messages

	FirstName	LastName	Gender	ParentsNames	SPHome
1	Ismael	Zara	Male	Melinda and Robert Zara	0
2	Janet	West	Female	Kirsten Farmer	1
3	Anselme	Waters	Male	Claudette and Dr. Celestin Waters	1
4	Maurice	Walken	Male		0
5	Justin	Vittas	Male	Clarice Vittas	1
6	Suzanna	Verde	Female	Suzanne and Thommy Verder	0
7	Judith	Steinberg	Female	Jim	0
8	Harriette	Sans	Female	Thomas Sans	0
9	Martine	Quarles	Female		0
10	Sebastien	Porter	Male	John and Christine Porter	0
11	Santos	Pacheco	Male	Dr. Julia Santana	0
12	George	Orion	Male	Jennifer Freemont	1
13	Ella	Napolis	Female	Sandra and Pedro Napolis	0
14	Thomas	Moore	Male	Deborah Moore	0
15	Arthur	Milley	Male	Jeffrey Milley	0
16	Ann	Miller	Female		0
17	Victoria	Milchen	Female	Paul Milchen	1
18	Donnie	Mart	Female	Alan Marly	1

Q | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 rows

Aprendizaje práctico: la clasificación Resords

1. Para mostrar la lista de las propiedades, basada en valores, empezando por lo desconocido seguido caro de ellos, escriba la siguiente declaración:

   SELECCIONE COMO house.YearBuilt [Año de construcción],
   COMO house.PropertyType [Tipo],
   COMO house.Bedrooms [Camas],
   COMO house.Bathrooms [Baños],
   COMO house.MarketValue [valor]
   Propiedades de casa
   ORDER BY house.MarketValue
   IR

2. Presione la tecla F5 para ejecutar

3. Para mostrar la lista de propiedades por orden cronológico empezando por el más reciente, el cam declaración de la siguiente manera:

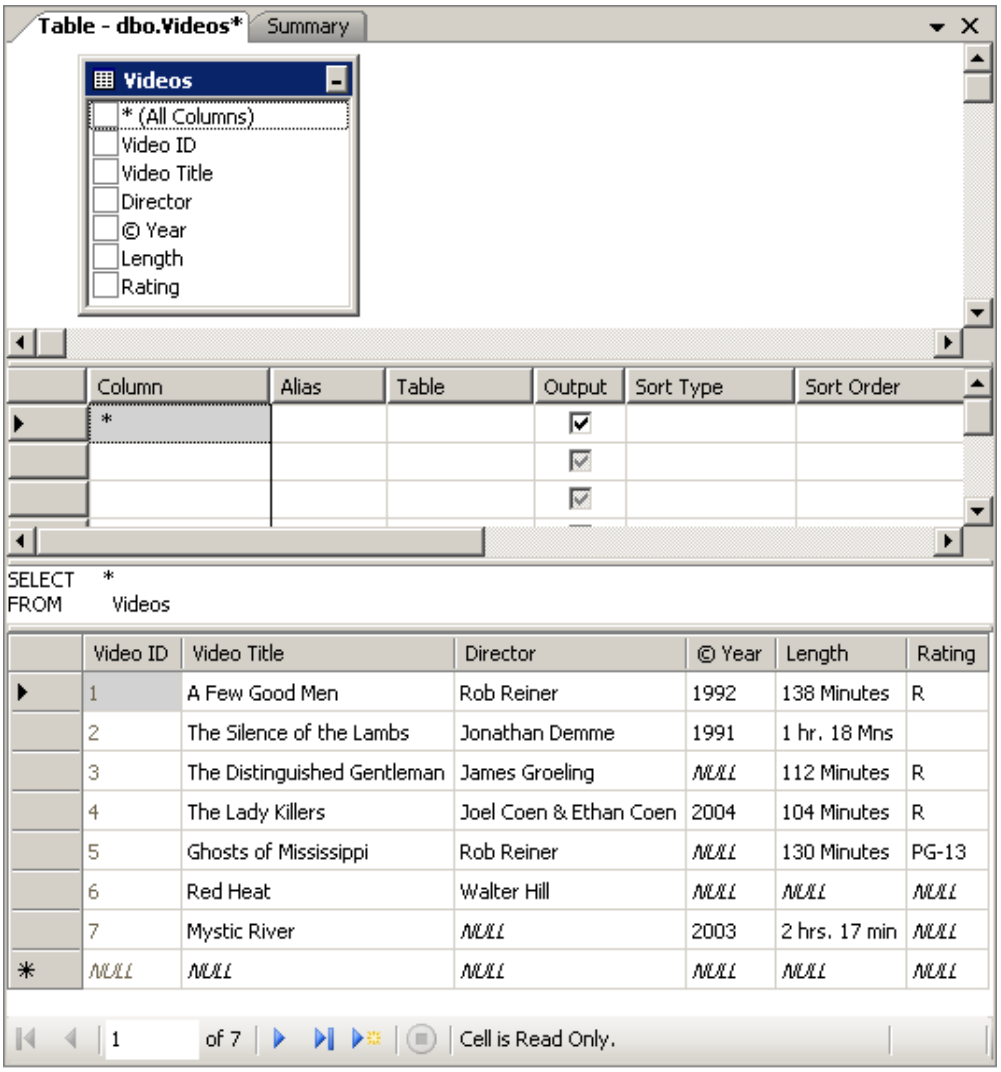
SELECCIONE COMO house.YearBuilt [Año de construcción],
COMO house.PropertyType [Tipo],
COMO house.Bedrooms [Camas],
COMO house.Bathrooms [Baños],
COMO house.MarketValue [valor]
Propiedades de casa
ORDER BY DESC YearBuilt
IR

4. Presione la tecla F5 para ejecutar

Operadores y Análisis de Datos

Introducción a las condiciones

En la lección anterior, hemos aprendido que podemos analizar los datos mediante la ventana de Tab
ejemplo:



También aprendieron a analizar los datos mediante la creación y ejecución de una sentencia SQL en
consulta. En lugar de seleccionar todos los datos como lo hemos hecho hasta ahora utilizando **la** p
SELECT, usted puede presentar una condición que la base de datos que seguir para aislar registros espe

Al analizar los datos o si está creando una consulta utilizando el cuadro ventana, puede escribir una exp

uno o más operadores lógicos revisamos en **Lección 5.** Aquí hay un ejemplo de una expresión

> «12 / 31/1993 '

Esto significa que las fechas que se producen después de 1993 sería seleccionado.

Cuando es el Estado

Si usted está escribiendo el **comando SELECT,** en la formulación de una condición, **cuando** se utiliza con una fórmula básica de la siguiente manera:

Seleccione *lo WhatObject* DE DONDE ***expresión;***

Las expresiones utilizadas en las condiciones se construyen utilizando algebraica, lógica, y los operado La *expresión* se denomina factor de criterio. Aunque un grupo de expresiones, lo que se llama crit palabra se utiliza a veces los criterios de una singular expresión también. Es la expresión escrita util siguiente:

ColumnName = Valor

El factor de *ColumnName* existentes debe ser una columna de una tabla. Es seguido por el operador factor de *valor* es el valor que establezca la condición. Si el valor es una palabra o un grupo de p llamado una cadena), debe incluir en una sola cita. Si se trata de un número, puede escribir su valor nu

Aquí hay un ejemplo de una base de datos de los estudiantes, una tabla con el nombre de los estudian una lista de estudiantes de sexo femenino:

```
SELECCIONE DateOfBirth, LastName, FirstName,
        Género, Estado, ParentsNames
De los estudiantes
DÓNDE Género = 'Mujeres';
IR
```

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT DateOfBirth, LastName, FirstName,
       Gender, State, ParentsNames
FROM Students
WHERE Gender='Female';
GO
```

Results | Messages

	DateOfBirth	LastName	FirstName	Gender	State	ParentsNames
1	1991-10-05 00:00:00.000	Hoak	Suzie	Female	MD	Christianne Hoak
2	1996-04-10 00:00:00.000	Clarck	Antoinette	Female	MD	Daniel and Henriette Clarck
3	1996-06-02 00:00:00.000	West	Janet	Female	MD	Kirsten Farmer
4	1997-06-12 00:00:00.000	Chang	Catherine	Female	DC	Julie Best and Dr. Peter Chang
5	1995-10-07 00:00:00.000	Ashburn	Sherryl	Female	MD	Shelia and Patrick Ashburn
6	1998-10-11 00:00:00.000	DeGaram	Ruby	Female	MD	Lynn and Anthony DeGaram
7	1990-10-22 00:00:00.000	Chance	Carole	Female	MD	Carole and Andy Chance
8	1989-10-05 00:00:00.000	Lobo	Brenda	Female	VA	Priscilla and Michel Lobo
9	1988-03-12 00:00:00.000	Verde	Suzanna	Female	MD	Suzanne and Thommy Verder
10	1991-10-08 00:00:00.000	Franse	Mincy	Female	MD	Dr. Melanie and Cleve Franse
11	1989-02-15 00:00:00.000	Andriamirano	Arlene	Female	VA	Christianne Burton
12	1989-06-23 00:00:00.000	Ledoux	Gabrielle	Female	MD	Brenda and Patrick Fasten
13	1990-08-20 00:00:00.000	Lobila	Koko	Female	DC	Aquilla and Dr. Diallo Lobila
14	1995-09-28 00:00:00.000	Duma	Arlette	Female	MD	Ursula Duma
15	1996-08-02 00:00:00.000	Sans	Harriette	Female	VA	Thomas Sans
16	1996-06-11 00:00:00.000	Howerson	Bernadette	Female	MD	Caron and Rick Howerson
17	1989-02-22 00:00:00.000	Steinberg	Judith	Female	MD	Jim
18	1991-04-24 00:00:00.000	Napolis	Ella	Female	MD	Sandra and Pedro Napolis

Query executed succ... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 27 row

En **caso de** una declaración, también puede utilizar la expresión **ORDER BY** para ordenar una lista de en una columna de su elección. Aquí está un ejemplo:

SELECCIONE DateOfBirth, LastName, FirstName,
Género, Estado, ParentsNames
De los estudiantes
DÓNDE Estado = 'MD'
ORDER BY Apellidos;
IR

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT DateOfBirth, LastName, FirstName,
       Gender, State, ParentsNames
FROM Students
WHERE State='MD'
ORDER BY LastName;
GO
```

Results | Messages

	DateOfBirth	LastName	FirstName	Gender	State	ParentsNames
1	1989-02-14 00:00:00.000	Amorros	Tim	Male	MD	Eduardo Amoros
2	1995-10-07 00:00:00.000	Ashburn	Sherryl	Female	MD	Shelia and Patrick Ashburn
3	1995-02-12 00:00:00.000	Bastens	Martha	Female	MD	
4	1995-12-06 00:00:00.000	Bell	Lester	Male	MD	Bernadette and Paulin Bell
5	1989-09-10 00:00:00.000	Broadskey	Millicent	Female	MD	
6	1990-10-22 00:00:00.000	Chance	Carole	Female	MD	Carole and Andy Chance
7	1997-06-20 00:00:00.000	Chico	Danilo	Male	MD	Michelle and Stephen Chico
8	1996-04-10 00:00:00.000	Clarck	Antoinette	Female	MD	Daniel and Henriette Clarck
9	1996-10-14 00:00:00.000	Davis	Martin	Male	MD	
10	1995-12-02 00:00:00.000	Dean	Nehemiah	Male	MD	Marie Rodnat
11	1998-10-11 00:00:00.000	DeGaram	Ruby	Female	MD	Lynn and Anthony DeGaram
12	1995-09-28 00:00:00.000	Duma	Arlette	Female	MD	Ursula Duma
13	1996-12-26 00:00:00.000	Edelman	Charles	Male	MD	Jasmine Beaulieu
14	1996-04-30 00:00:00.000	Farms	Paul	Male	MD	Rebecca and Peter Farms
15	1991-10-08 00:00:00.000	Franse	Mincy	Female	MD	Dr. Melanie and Cleve Franse
16	1996-04-19 00:00:00.000	Fuller	Clint	Male	MD	Shelley and Dr. Oregon Fuller
17	1990-02-21 00:00:00.000	Hagers	Ralph	Male	MD	
18	1991-10-05 00:00:00.000	Hoak	Suzie	Female	MD	Christianne Hoak

Query executed s... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 39 rows

Aprendizaje práctico: Uso DONDE

1. Para ver una lista de propiedades sólo en DC, escriba lo siguiente:

   SELECCIONE COMO house.PropertyNumber [Prop. #],
   house.Address,
   house.City,
   house.State,
   COMO house.ZIPCode [Ubicación]
   COMO house.YearBuilt [Año de construcción],
   COMO house.PropertyType [Tipo],
   COMO house.MarketValue [valor]
   Propiedades de casa
   DÓNDE house.ZIPCode <20500
   IR

2. Presione la tecla F5 para ejecutar la declaración

3. Para obtener una lista de propiedades sólo en Virginia, el cambio de la declaración de la siguiente r

   SELECCIONE COMO house.PropertyNumber [Prop. #],
   house.Address,

```
            house.City,
            house.State,
            COMO house.ZIPCode [Ubicación]
            COMO house.YearBuilt [Año de construcción],
            COMO house.PropertyType [Tipo],
            COMO house.MarketValue [valor]
    Propiedades de casa
    DÓNDE house.State = 'va'
    IR
```

4. Presione la tecla F5 para ejecutar

5. Para obtener una lista de sólo los más nuevos o en las propiedades construidas después de 2000, e
declaración de la siguiente manera:

```
    SELECCIONE house.Address,
        house.City,
        house.State,
        COMO house.YearBuilt [Año de construcción],
        COMO house.PropertyType [Tipo],
        COMO house.MarketValue [valor]
    Propiedades de casa
    DÓNDE house.YearBuilt> = 2000
    IR
```

6. Presione la tecla F5 para ejecutar

CENTRAL.RealEs...1 - Results.sql | Object Explorer Details

```
SELECT house.Address,
       house.City,
       house.State,
       house.YearBuilt AS [Year Built],
       house.PropertyType AS [Type],
       house.MarketValue AS [Value]
FROM Properties house
WHERE house.YearBuilt >= 2000
GO
```

Results | Messages

	Address	City	State	Year Built	Type	Value
1	10315 North Hacht Rd	College Park	MD	2000	Single Family	620724.00
2	10340 Helmes Street #408	Silver Spring	MD	2000	Condominium	242775.00
3	NULL	Chevy Chase	NULL	2001	Single Family	525450.00
4	14005 Sniders Blvd	Laurel	MD	2002	Townhouse	412885.00
5	680 Prushia Rd	Washington	DC	2000	Single Family	555885.00
6	10340 Helmes Street #1012	Silver Spring	MD	2000	Condominium	252775.00
7	980 Phorwick Street	Washington	DC	2004	Single Family	735475.00
8	7307 Everett Hwy	Washington	DC	2006	Townhouse	420550.00
9	10340 Helmes Street#1006	Silver Spring	MD	2000	Condominium	258445.00
10	7303 Warfield Court	Tysons Corner	VA	2006	Single Family	825775.00
11	12401 Conniard Ave	Takoma Park	MD	2004	Townhouse	280775.00
12	5121 Riehl Ace	Fairfax	VA	2002	Townhouse	325620.00
13	944 Fryer Ave	Chevy Chase	MD	2002	Single Family	625665.00
14	1950 Galego Street	Germantown	MD	2007	Single Family	428665.00
15	NULL	Takoma PArk	MD	2000	Conbominium	225885.00
16	12366 Fowler Ave	Alexandria	VA	2007	Townhouse	402815.00

Quer... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (52) | RealEstate1 | 00:00:00 | 16 rows

7. Para obtener una lista de propiedades en Maryland organizado por códigos de área, el cambio de la la siguiente manera:

```
SELECCIONE house.PropertyType como tipo,
      COMO house.YearBuilt [Año de construcción],
      house.City,
      house.ZIPCode,
      house.Bedrooms como camas,
      house.Bathrooms como los baños,
      house.MarketValue como valor
Propiedades de casa
DÓNDE house.State = 'md'
ORDER BY house.ZIPCode
IR
```

8. Presione la tecla F5 para ejecutar la declaración

Cómo ocultar una columna

En nuestra **SELECT** hasta ahora, se nos seleccionar las columnas que necesitamos para mostrar. declaración, puede solicitar una de las condiciones para una columna sin incluir la columna en el resulta

considere la posibilidad de la consulta. Se utiliza para mostrar una lista de estudiantes de sexo feme sabemos que el resultado podría mostrar sólo las niñas, se convierte en redundante incluir la columna declaración. En este caso, puede ocultar esa columna en el resultado.

Para ocultar una columna de una consulta, omita esa columna en **la** instrucción SELECT, pero la partici **de** enfermedad. Aquí está un ejemplo:

```
SELECCIONE DateOfBirth, LastName,
        Nombre, Estado, ParentsNames
De los estudiantes
DÓNDE Género = 'Mujeres';
IR
```

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT DateOfBirth, LastName,
       FirstName, State, ParentsNames
FROM Students
WHERE Gender='Female';
GO
```

Results | Messages

	DateOfBirth	LastName	FirstName	State	ParentsNames
1	1991-10-05 00:00:00.000	Hoak	Suzie	MD	Christianne Hoak
2	1996-04-10 00:00:00.000	Clarck	Antoinette	MD	Daniel and Henriette Clarck
3	1996-06-02 00:00:00.000	West	Janet	MD	Kirsten Farmer
4	1997-06-12 00:00:00.000	Chang	Catherine	DC	Julie Best and Dr. Peter Chang
5	1995-10-07 00:00:00.000	Ashburn	Sherryl	MD	Shelia and Patrick Ashburn
6	1998-10-11 00:00:00.000	DeGaram	Ruby	MD	Lynn and Anthony DeGaram
7	1990-10-22 00:00:00.000	Chance	Carole	MD	Carole and Andy Chance
8	1989-10-05 00:00:00.000	Lobo	Brenda	VA	Priscilla and Michel Lobo
9	1988-03-12 00:00:00.000	Verde	Suzanna	MD	Suzanne and Thommy Verder
10	1991-10-08 00:00:00.000	Franse	Mincy	MD	Dr. Melanie and Cleve Franse
11	1989-02-15 00:00:00.000	Andriamirano	Arlene	VA	Christianne Burton
12	1989-06-23 00:00:00.000	Ledoux	Gabrielle	MD	Brenda and Patrick Fasten
13	1990-08-20 00:00:00.000	Lobila	Koko	DC	Aquilla and Dr. Diallo Lobila
14	1995-09-28 00:00:00.000	Duma	Arlette	MD	Ursula Duma
15	1996-08-02 00:00:00.000	Sans	Harriette	VA	Thomas Sans
16	1996-06-11 00:00:00.000	Howerson	Bernadette	MD	Caron and Rick Howerson
17	1989-02-22 00:00:00.000	Steinberg	Judith	MD	Jim
18	1991-04-24 00:00:00.000	Napolis	Ella	MD	Sandra and Pedro Napolis

Query execu... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 27 rows

Observe que el **comando SELECT** no tiene la columna de Género y de la consiguiente consulta no mu de Género.

Aprendizaje práctico: No Listado una columna

1. Para ver una lista de propiedades sólo en Maryland, el cambio de la declaración de la siguiente ma

   SELECCIONE house.PropertyType como tipo,

```
            COMO house.YearBuilt [Año de construcción],
            house.City,
            house.Bedrooms como camas,
            house.Bathrooms como los baños,
            house.MarketValue como valor
      Propiedades de casa
      DÓNDE house.State = 'md'
      IR
```

2. Presione la tecla F5 para ejecutar la declaración

Negando algunos registros

En la lección 5, vimos que se puede usar el operador NOT para negar la validez de una expresión boo
la siguiente declaración:

```
SELECCIONE DateOfBirth, LastName, FirstName,
        Estado, ParentsNames
De los estudiantes
DÓNDE Género = 'Mujeres';
IR
```

Cuando este se ejecuta, aparece una lista de estudiantes de sexo femenino que mostrar. En lugar d
obtener una lista de estudiantes de sexo masculino, puede negar esta condición. Para ello, escriba
condición. Esto se haría de la siguiente manera:

```
SELECCIONE DateOfBirth, LastName, FirstName,
        Género, Estado, ParentsNames
De los estudiantes
DONDE NO Género = 'Mujeres';
IR
```

Para que esta condición sea más fácil de leer, debe incluir la expresión positiva entre paréntesis. Es
siguiente manera:

```
SELECCIONE DateOfBirth, LastName, FirstName,
        Género, Estado, ParentsNames
De los estudiantes
DONDE NO (Sexo = 'Femenino');
IR
```

Esto indica claramente que es la expresión en el paréntesis que se niega. De la misma manera, pued
NULL para encontrar los registros que no son nulos. Por ejemplo, puede crear una lista de sólo los
tienen un valor nulo en una determinada columna. Aquí está un ejemplo:

```
SELECCIONE DateOfBirth, LastName, FirstName,
        Estado, ParentsNames
De los estudiantes
DÓNDE Estado IS NOT NULL;
IR
```

Cuando este se ejecuta, el cuadro que muestre sólo los registros que incluyen un estado para cada estu

Aprendizaje práctico: negar un valor

1. Para ver una lista de propiedades que sólo se les ha asignado un número de propiedad, cambio de
la siguiente manera:

```
SELECCIONE COMO dbo.Properties.PropertyNumber [Prop. #],
            dbo.Properties.PropertyType como tipo,
```

```
        COMO dbo.Properties.YearBuilt [Año de construcción],
        dbo.Properties.City,
        dbo.Properties.State,
        COMO dbo.Properties.ZIPCode [CP],
        dbo.Properties.Bedrooms como camas,
        dbo.Properties.Bathrooms como los baños,
        dbo.Properties.MarketValue como valor
DE dbo.Properties
DÓNDE dbo.Properties.PropertyNumber IS NOT NULL
IR
```

2. Presione la tecla F5 para ejecutar la declaración

Operaciones de Análisis de Datos

Lógica Conjunción y disyunción

Introducción

Hasta el momento, hemos señalado las condiciones de uno en uno. Esto hizo más fácil su interpre usted tendrá que probar una condición que depende de otro. Álgebra booleana le permite combinar d uso y el resultado, o para poner a prueba dos condiciones, sino considerar si bien es cierto.

Aprendizaje práctico: Condiciones de Uso de Análisis de Datos

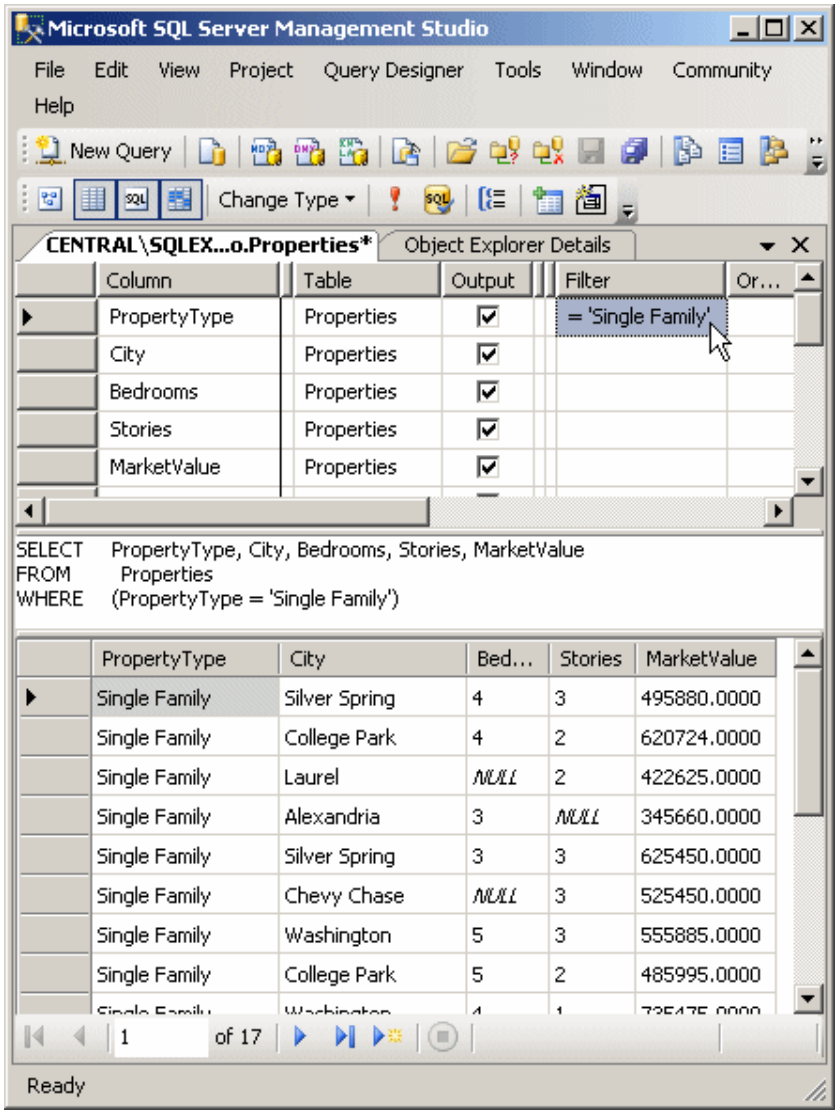
1. Inicie Microsoft SQL Server y SQL Server Management Studio.
   Si aún no, crear la **base de datos RealEstate1**
2. En el Explorador de objetos, haga clic derecho en las bases de datos y haga clic en Nueva consulta

Lógico Conjunciones

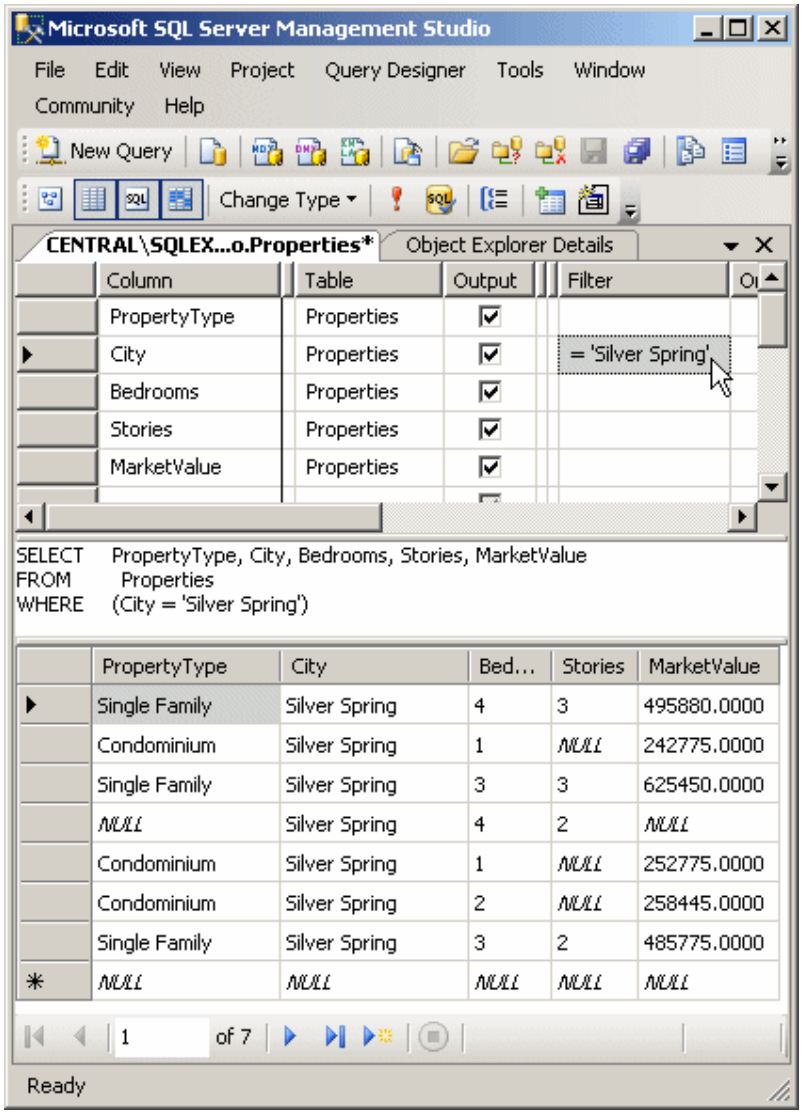
Para una compañía de bienes raíces, suponga que tiene un cliente que está considerando la compra de una sola casa de familia en torno a Silver Spring, Maryland, que comprobar la lista de las propiedades y saber lo que tienes. Para responder a esta solicitud, usted debe examinar dos condiciones para cada propiedad:

- La propiedad debe ser una sola familia
- La propiedad debe estar ubicada en Silver Spring

Al preparar su lista antes de ver el cliente, puede empezar por construir una consulta que sólo las li unifamiliares. Aquí está un ejemplo:



La otra condición requiere que la casa se encuentra en Silver Spring. Usted podría establecer como sigu



De estos dos resultados, nota que no hay ninguna relación entre el hecho de que una propiedad es u que se encuentra en Silver Spring. Sin embargo, nuestro cliente se compra la propiedad sólo si ambas verdaderas: La propiedad es una sola familia Y la propiedad está ubicada en Silver Spring. Este tipo conoce como relación lógica.

Para crear una relación lógica en SQL, utilice el **operador** AND. T o escribir la sentencia, se utiliza la sig

SELECCIONE *WhatColumn (s)*
DE *WhatObject*
DÓNDE *Condition1* Y *Condition2*

El *WhatColumn (s)* y la *WhatObject* factores son los mismos que hemos utilizado hasta ahora. **Y** la pa operador que se une a las condiciones. Cada condición se escribe como una operación de SQL utili siguiente:

Operador de la columna Valor

En este caso, **cuando** el operador condicional **Si** se asemeja a la declaración. El *Condition1* es la examinó. Si la primera condición es falsa, toda la declaración es falsa y no hay ninguna razón para exa

condición. Si la primera condición es verdadera, entonces la segunda condición sería examinada.

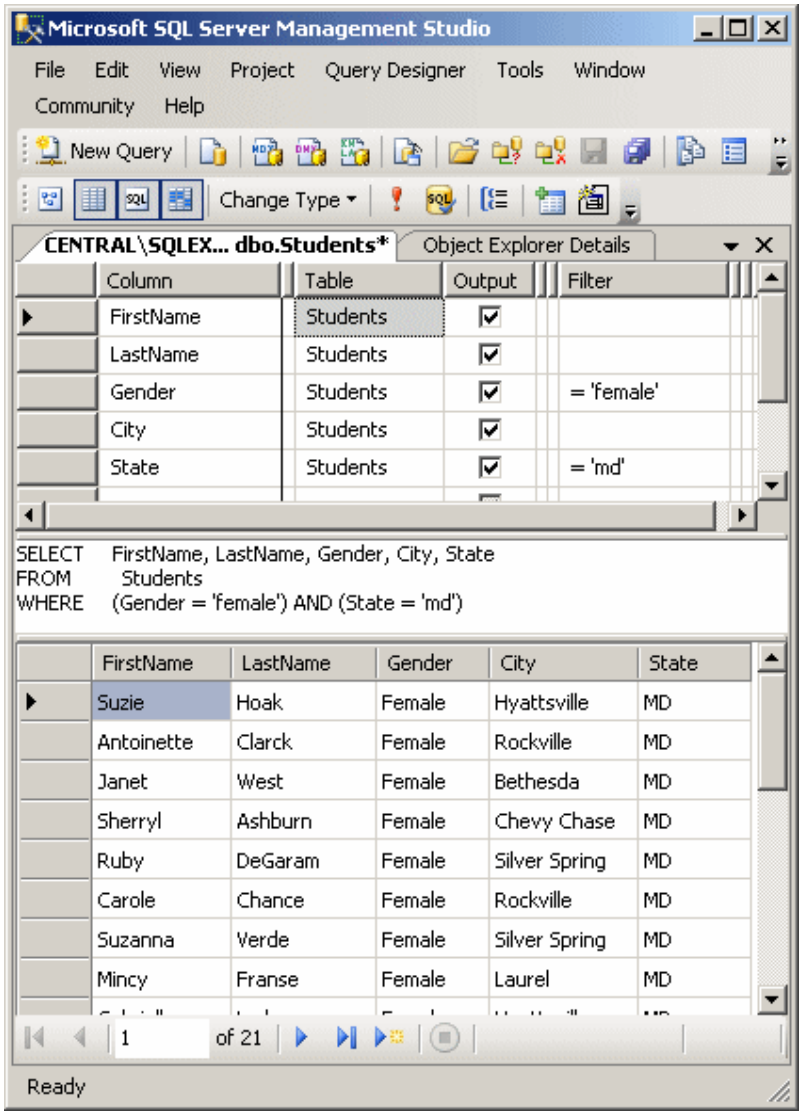
Sobre esta base, supongamos que queremos obtener una lista de las estudiantes que viven en Maryla de SQL utiliza para obtener esta lista se puede escribir como sigue:

SELECCIONE Nombre, Apellidos, Sexo, Ciudad, Estado
De los estudiantes
DÓNDE Género = "femeninos" y el Estado = 'md';

Nos dijo que cada una de las condiciones se evaluarán por separado. Por esta razón, para hacer relación más fácil de leer, cada estado debe incluirse entre paréntesis. Por lo tanto, la sentencia SQL como sigue:

SELECCIONE Nombre, Apellidos, Sexo, Ciudad, Estado
De los estudiantes
WHERE (Sexo = "femeninos") Y (Estado = 'md')

Esto produciría:



También puede negar la condición de que lo precede con el **operador** NOT.

Aprendizaje práctico: El uso de una lógica de conjunto

1. Para ver una lista de casas unifamiliares ubicada en Silver Spring, escriba la siguiente declaración:

```
SELECCIONE COMO h.PropertyNumber [Prop. #],
        h.PropertyType como tipo,
        COMO h.YearBuilt [Año de construcción],
        h.City,
        h.State,
        COMO h.ZIPCode [CP],
        h.Bedrooms como camas,
        h.Bathrooms como los baños,
        h.MarketValue como valor
Propiedades de h
WHERE (h.PropertyType = 'Unifamiliar') Y (Ciudad = 'Silver Spring');
IR
```

2. Presione la tecla F5 para ejecutar la declaración

3. Supongamos que un posible comprador está considerando comprar una casa pero no pueden gasta 400.000.
   Para obtener una lista de la ciudad, casas que cuestan menos de 400.000 dólares, el cambio de la siguiente manera:

```
SELECCIONE COMO h.PropertyNumber [Prop. #],
        h.PropertyType como tipo,
        COMO h.YearBuilt [Año de construcción],
        h.City,
        h.State,
        COMO h.ZIPCode [CP],
        h.Bedrooms como camas,
        h.Bathrooms como los baños,
        h.MarketValue como valor
Propiedades de h
WHERE (h.PropertyType = 'Casa') Y (h.MarketValue <400000);
IR
```

4. Presione la tecla F5 para ejecutar la declaración

CENTRAL.RealEs...ts\Results.sql | Object Explorer Details

```
SELECT h.PropertyNumber AS [Prop #],
       h.PropertyType AS Type,
       h.YearBuilt AS [Year Built],
       h.City,
       h.State,
       h.ZIPCode AS [ZIP Code],
       h.Bedrooms AS Beds,
       h.Bathrooms AS Baths,
       h.MarketValue AS Value
FROM Properties h
WHERE (h.PropertyType = 'Townhouse') AND
      (h.MarketValue < 400000);
GO
```

Results | Messages

	Prop #	Type	Year Built	City	State	ZIP Code	Beds	Baths	Value
1	207850	Townhouse	1988	Rockville	MD	20854	3	2.5	325995.00
2	NULL	Townhouse	1975	Washington	NULL	NULL	4	NULL	366775.00
3	447597	Townhouse	1992	Hyattsville	MD	20782	3	2	365880.00
4	297497	Townhouse	2004	Takoma Park	MD	20910	3	2.5	280775.00
5	826927	Townhouse	2002	Fairfax	VA	22232	3	1.5	325620.00

Query exe... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (52) | RealEstate1 | 00:00:00 | 5 rows

5. Supongamos que el cliente por encima de la que preferiría estar en casa de Rockville.
   Para obtener una lista de hogares en la ciudad de Rockville primavera que cuestan menos de 400.0
   cambio de la declaración de la siguiente manera:

```
SELECCIONE COMO h.PropertyNumber [Prop. #],
        h.PropertyType como tipo,
        COMO h.YearBuilt [Año de construcción],
        h.City,
        COMO h.ZIPCode [CP],
        h.Bedrooms como camas,
        h.Bathrooms como los baños,
        h.MarketValue como valor
Propiedades de h
WHERE (h.PropertyType = 'Casa') Y
      (h.MarketValue <400000) Y
      (h.City = 'Rockville');
IR
```

6. Presione la tecla F5 para ejecutar la declaración

Disyunción lógica

Supongamos que un cliente está considerando la compra de una sola casa familiar o una casa. Para p propiedades, debe crear una consulta que considera que sólo estas dos opciones. Antes de constru puede señalar lo siguiente:

- La propiedad es una sola familia
- La propiedad es una casa

Al crear la lista, usted desea incluir una propiedad sólo si es una sola familia o una casa:

1. Si la propiedad es una sola familia, nuestra declaración es verdadera y no es necesario para compr condición
2. Si la propiedad no es una casa, entonces se considera la segunda condición. Si la propiedad es un ciudad, nuestro estado se convierte en verdad

Este tipo de declaración se denomina lógica disyuntiva. La lógica disyunción se expresa en SQL con **el** c

Aprendizaje práctico: El uso de una lógica disyunción

1. Para ver una lista única de la familia y adosadas, cambiar la declaración de la siguiente manera:

```
SELECCIONE COMO h.PropertyNumber [Prop. #],
        h.PropertyType como tipo,
        COMO h.YearBuilt [Año de construcción],
        h.City,
        COMO h.ZIPCode [CP],
        h.Bedrooms como camas,
        h.Bathrooms como los baños,
        h.MarketValue como valor
Propiedades de h
WHERE (h.PropertyType = 'familia') O
      (h.PropertyType = 'Casa');
IR
```

2. Presione la tecla F5 para ejecutar la declaración

Otros Operadores lógicos en las consultas

Rango de prueba: ENTRE

Si usted tiene un rango de valores lógico y lo que desea saber si un determinado valor se encuent puede agregar **entre** un operador de un **caso de** declaración. **ENTRE** el operador se combina con **Y** lista de registros entre dos valores. La fórmula básica de este operador es la siguiente:

DÓNDE *expresión entre* el comienzo y *de finalización*

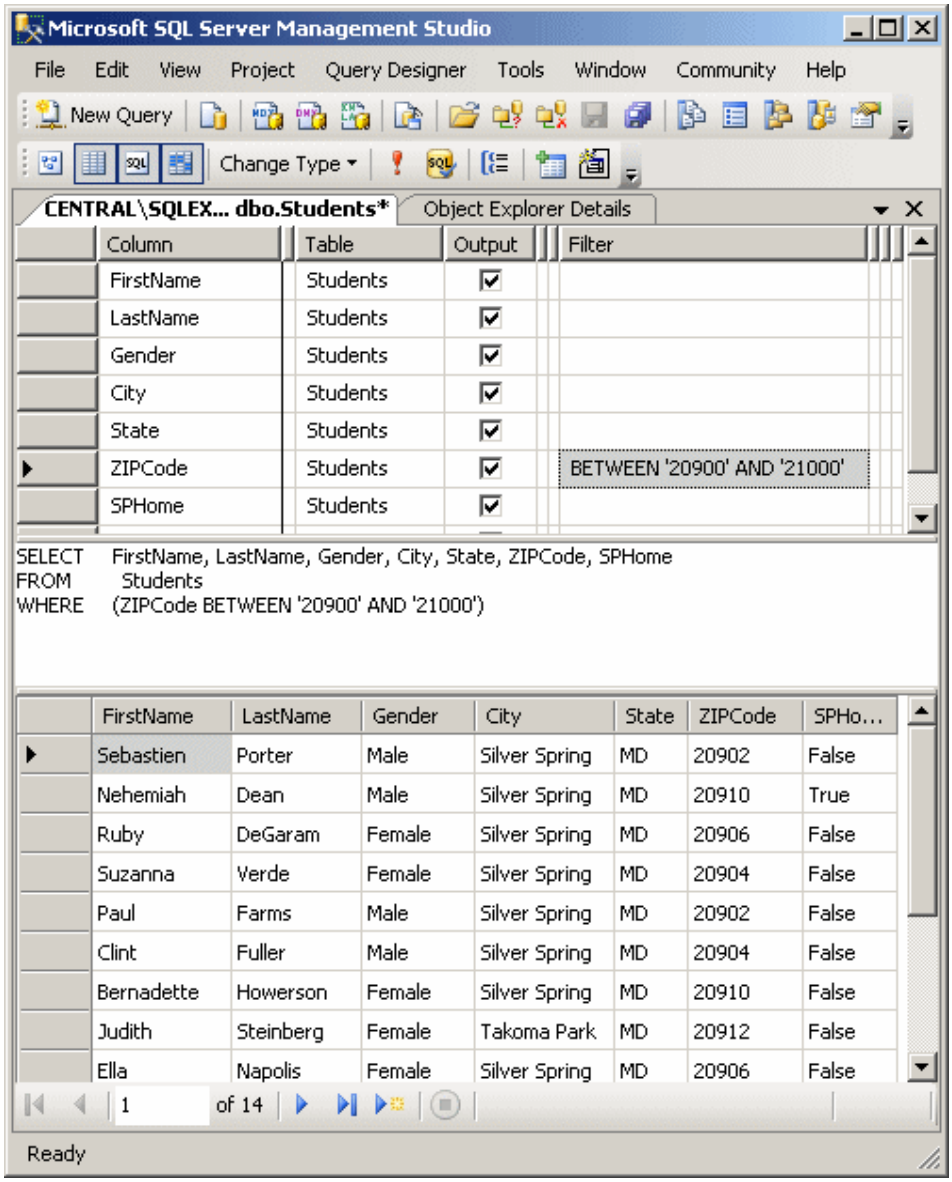
La *expresión* de asignación es generalmente el nombre de la columna cuyos valores desea examina factor de valor inicial de la gama a considerar. El *fin* es el factor de mayor valor a considerar en el ra esta condición se ejecuta, produce la lista de valores entre el *inicio* y de *finalización.*

Para crear una expresión **entre** en el cuadro ventana, seleccione el que desee columnas. En el mar columna que corresponde al campo en el que desea establecer como condición, **entre** el tipo de expres ejemplo que produce una lista de estudiantes que viven en el sur de Maryland, donde el código postal e (excluido):

```
SELECCIONE Nombre, Apellidos, Sexo, ciudad, estado, código postal, SPHome
De los estudiantes
DONDE ENTRE Código postal'20500 'Y'21000'
```

Por lo general, es un hábito de incluir la declaración en su conjunto **entre** paréntesis.

La declaración anterior se producen:



Aprendizaje práctico: Búsqueda de registros entre los valores

1. Para obtener una lista de los bienes cuyos precios están en el rango de $ 350.000 y $ 450.000, el declaración de la siguiente manera:

```
SELECCIONE COMO h.PropertyNumber [Prop. #],
        h.PropertyType como tipo,
        COMO h.YearBuilt [Año de construcción],
        h.City,
        COMO h.ZIPCode [CP],
        h.Bedrooms como camas,
        h.Bathrooms como los baños,
        h.MarketValue como valor
Propiedades de h
WHERE (h.MarketValue ENTRE 350000 Y 450000);
IR
```

2. Presione la tecla F5 para ejecutar la declaración

3. Después de ejecutar la declaración, usted sabe que usted no tiene nada de eso en Silver Spring. P sugieren que el cliente considere otras casas alrededor de Silver Spring. Para crear la declaración, cuyos códigos postales de entre 20800 y 20999 (Condado de Montgomery, Maryland).
   Para obtener una lista de hogares de la ciudad para este cliente, el cambio de la declaración de la s manera:

```
SELECCIONE COMO h.PropertyNumber [Prop. #],
          h.PropertyType como tipo,
          COMO h.YearBuilt [Año de construcción],
          h.City,
          COMO h.ZIPCode [CP],
          h.Bedrooms como camas,
          h.Bathrooms como los baños,
          h.MarketValue como valor
Propiedades de h
WHERE (h.PropertyType = 'Casa') Y
        (h.MarketValue <400000) Y
        (h.ZIPCode = (SELECT h.ZIPCode
                        DÓNDE h.ZIPCode ENTRE'20500 'Y'21000'));
IR
```

4. Presione la tecla F5 para ejecutar la declaración

En una selección de la serie

Si usted tiene una serie de registros y que desea encontrar un registro o un grupo de registros entre ell el operador **EN** añadiendo a un **caso de** declaración. **EN** el operador es un tipo **o** de varios operadores. esta fórmula:

EN *(expresion1, Expression2, Expression_n)*

Cada *expresión* puede ser un factor de los valores de una columna. Esto es equivalente a *expresion1 Expression_n,* etc

Para crear una expresión **en** el cuadro en la ventana, seleccione el que desee columnas. En el mar columna que corresponde al campo en el que desea establecer como condición, el tipo **de** expresión.

De nuestra lista de los estudiantes, imagine que usted quiere obtener una lista de estudiantes que viver Spring, en Rockville, o en Chevy Chase. Puede escribir en una expresión de la siguiente manera:

```
SELECCIONE Nombre, Apellidos, Sexo, ciudad, estado, código postal, SPHome
De los estudiantes
EN DONDE Ciudad ( «la primavera de plata ',' Rockville", "Chevy Chase ');
```

Por lo general, es un hábito de incluir **en** toda la declaración en su propio paréntesis.

La declaración anterior se producen:

Microsoft SQL Server Management Studio

File Edit View Project Query Designer Tools Window Community Help

New Query | Change Type

CENTRAL\SQLEX... dbo.Students* | Object Explorer Details

Column	Table	Output	Filter
FirstName	Students	☑	
LastName	Students	☑	
Gender	Students	☑	
City	Students	☑	IN ('silver spring', 'rockville', 'chevy chase')
State	Students	☑	
ZIPCode	Students	☑	
SPHome	Students	☑	
		☑	

```
SELECT   FirstName, LastName, Gender, City, State, ZIPCode, SPHome
FROM      Students
WHERE    (City IN ('silver spring', 'rockville', 'chevy chase'))
```

FirstName	LastName	Gender	City	State	ZIPCode	SPHo...
Sebastien	Porter	Male	Silver Spring	MD	20902	False
Antoinette	Clarck	Female	Rockville	MD	20875	False
Nehemiah	Dean	Male	Silver Spring	MD	20910	True
Sherryl	Ashburn	Female	Chevy Chase	MD	20870	False
Ruby	DeGaram	Female	Silver Spring	MD	20906	False
Carole	Chance	Female	Rockville	MD	20875	False
Suzanna	Verde	Female	Silver Spring	MD	20904	False
Danilo	Chico	Male	Chevy Chase	MD	20872	False
Paul	Farms	Male	Silver Spring	MD	20902	False
Lester	Bell	Male	Silver Spring	MD	20815	False
Clint	Fuller	Male	Silver Spring	MD	20904	False
Bernadette	Howerson	Female	Silver Spring	MD	20910	False

1 of 25

Ready

Aprendizaje práctico: Intensificación EN

1. Para obtener una lista única de la familia y adosadas, cambiar la declaración de la siguiente maner

```
SELECCIONE COMO house.PropertyNumber [Prop. #],
        house.PropertyType como tipo,
        COMO house.YearBuilt [Año de construcción],
        house.City,
        house.State,
        COMO house.ZIPCode [CP],
        house.Bedrooms como camas,
        house.Bathrooms como los baños,
        house.MarketValue como valor
```

```
Propiedades de casa
EN DONDE house.PropertyType ( 'Unifamiliar', 'Casa');
IR
```

2. Presione la tecla F5 para ejecutar la declaración
3. Para obtener una lista de casas unifamiliares y ubicado en Rockville, cambiar la declaración de la s

```
SELECCIONE COMO house.PropertyNumber [Prop. #],
        house.PropertyType como tipo,
        COMO house.YearBuilt [Año de construcción],
        house.City,
        house.State,
        COMO house.ZIPCode [CP],
        house.Bedrooms como camas,
        house.Bathrooms como los baños,
        house.MarketValue como valor
Propiedades de casa
DONDE (EN house.PropertyType ( 'Unifamiliar', 'Casa')) Y
      (house.City = 'Rockville');
IR
```

4. Presione la tecla F5 para ejecutar la declaración

Plan Operador: SIMILAR

Introducción

La mayoría o la totalidad de los criterios que hemos especificado con la palabra clave **cuando** tie exactamente con el criterio especificado. En algunos casos, usted no puede recordar el valor exacto pero que desea especificar algún tipo de aproximación. Para ello, se utiliza el **operador** LIKE.

Si usted es visualmente la creación de la declaración, en el cuadro ventana, haga clic en el cu correspondientes a la columna en la que la condición se aplica y el tipo.

En una sentencia SQL, **el** operador LIKE se utiliza en una fórmula como sigue:

Patrón de expresión similar

La *expresión* es la expresión que se evaluará. Esta debe ser una expresión clara y válida.

El *patrón* puede ser un factor de valor que se encuentran en *Expresión.* Por ejemplo, puede ser el mi utilizado en **caso de** una declaración. En este caso, el operador de igualdad sería el mismo que **desee.**

```
SELECCIONE DateOfBirth, LastName, FirstName,
        Género, Estado, ParentsNames
De los estudiantes
DÓNDE Estado = 'VA';
IR
```

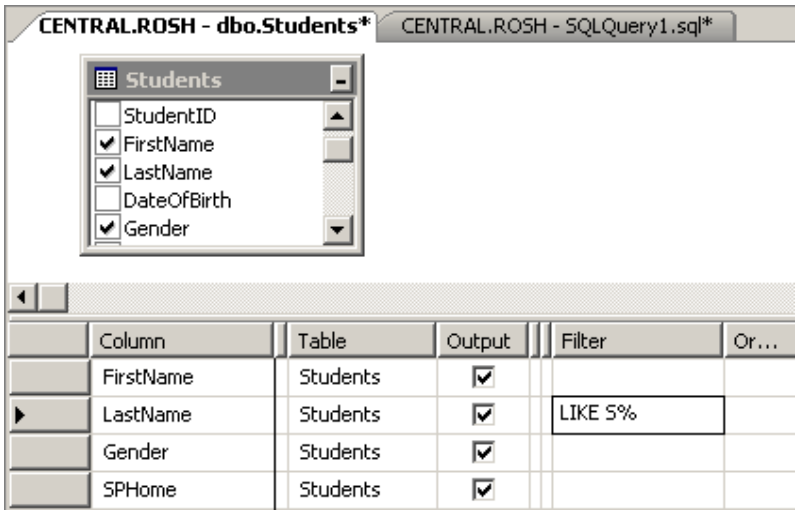
es equivalente a

```
SELECCIONE DateOfBirth, LastName, FirstName,
        Género, Estado, ParentsNames
De los estudiantes
DÓNDE Estado LIKE 'VA';
IR
```

La idea de utilizar un **operador** LIKE es dar una aproximación del tipo de resultado que desea. Ha usar con **el** operador LIKE.

Cualquier SIMILAR caracteres%

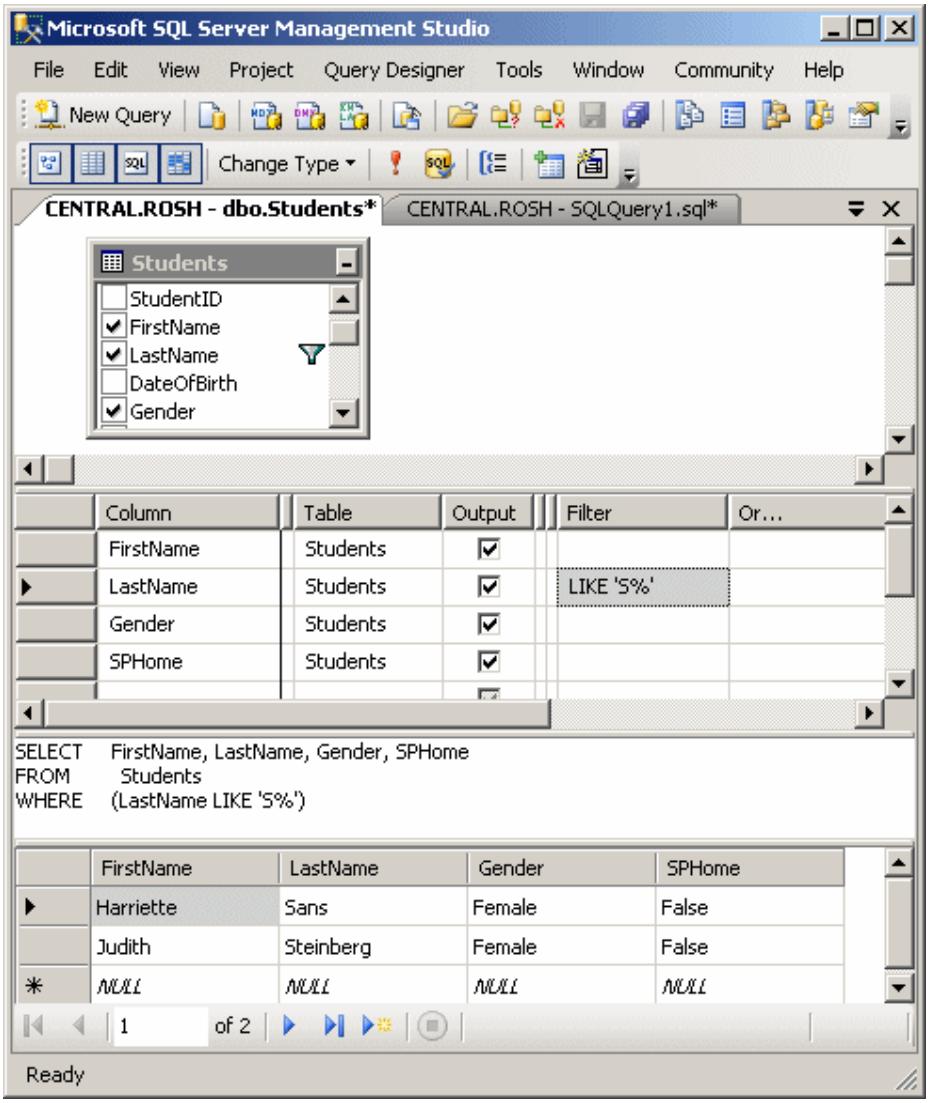
Si desea para adaptarse a cualquier carácter, en cualquier combinación, de cualquier longitud, utiliza e
se coloca una carta, como en el **S%,** la condición consistiría en la búsqueda de cualquier cadena que
Imagínese que usted desea obtener una lista de los estudiantes cuyos apellidos comienzan con S. Ento
condición **SIMILAR '% S'.** Para hacer esto visualmente, en la sección de Criterios, en el marco del filt
escriba la condición. Aquí está un ejemplo:



La sentencia SQL es esta consulta es:

SELECCIONE Nombre, Apellido, Género, SPHome
De los estudiantes
WHERE (Apellido LIKE '% S')

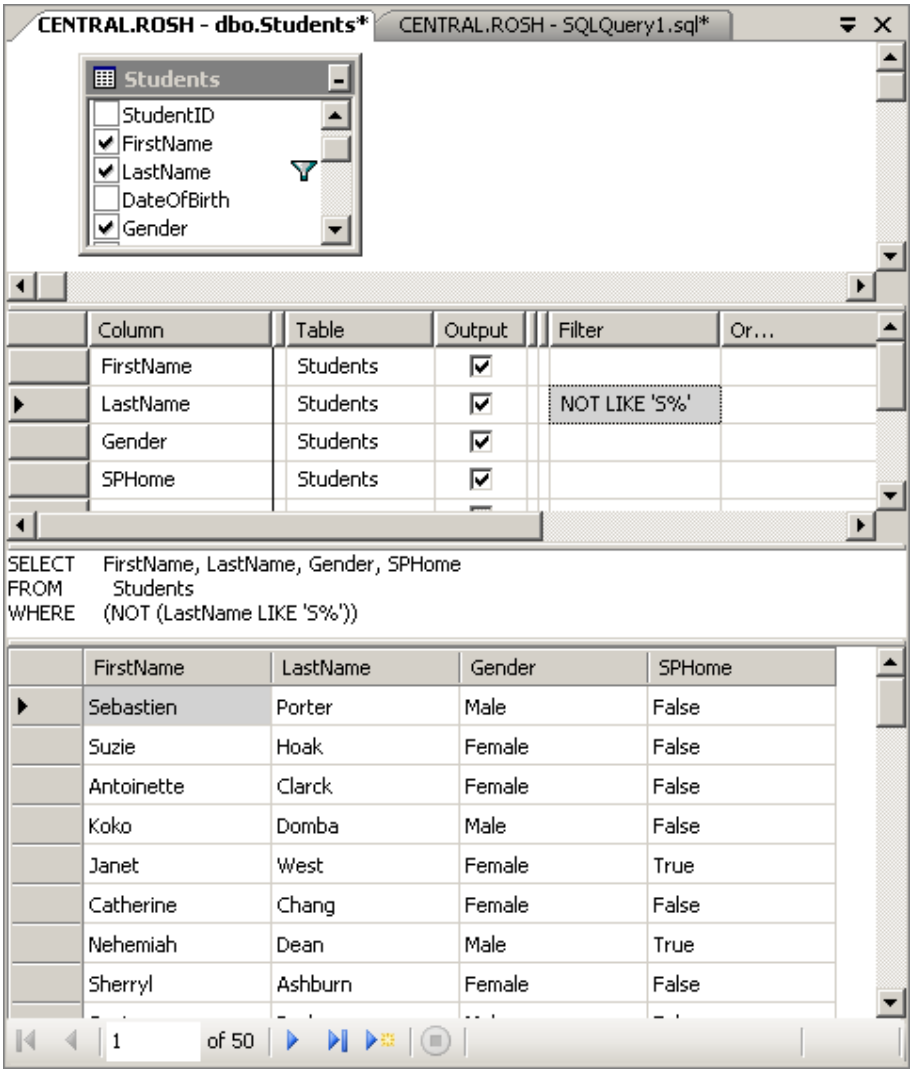
Esto produciría:



Usted puede negar esta condición por lo anterior con **NO.** Aquí está un ejemplo:

SELECCIONE Nombre, Apellido, Género, SPHome
De los estudiantes
DONDE (NO (Apellido LIKE '% S'))

Esto produciría:

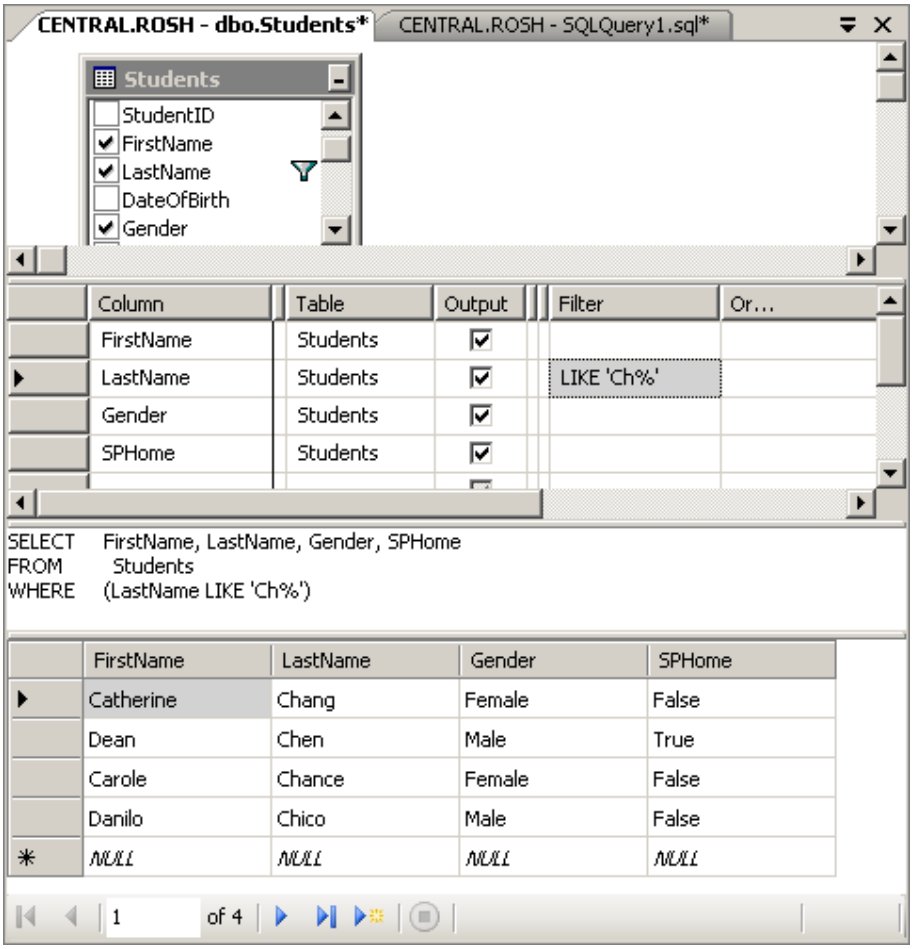


Esta vez, el resultado es la lista de alumnos cuyos apellidos no se inician con S.

Cuando preceder el **carácter%** con una carta, esa carta sólo se consideraría. Si lo prefiere, puede esp de caracteres que preceden al **símbolo%.** Por ejemplo, si usted tiene algunos nombres que comienza en la lista pero no recuerdo el final del nombre que está buscando, para crear la lista, puede especifi nombre que comience con canal y al final con lo que sea. En este caso, usted utilizaría **Ch.%** de la sigui

SELECCIONE Nombre, Apellido, Género, SPHome
De los estudiantes
WHERE (Apellido LIKE 'Ch.%')

Esto produciría:

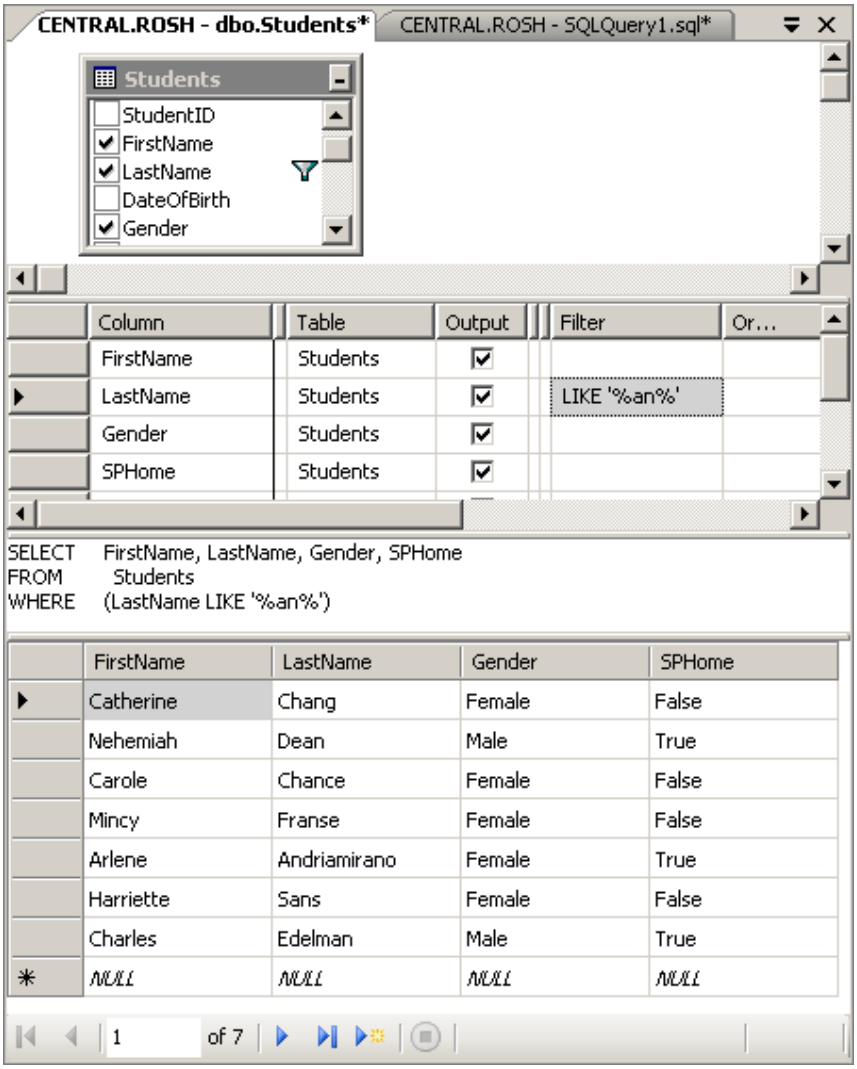


En lugar de poner fin a una carta o un grupo de letras **con%,** puede comenzar con la **instrucción LIK sería** como **"hijo%".** En este caso, todas las cadenas que terminan con hijo, como Johnson considerados.

Si recuerda, ni el principio ni el final de una cadena que desea buscar, pero ¿sabes una sub probablemente, incluido en el tipo de cadena que está buscando, puede **preceder%** y con efecto, **con sería** como **"un%%".** En este caso, todas las cadenas que incluyen "una" en cualquier lugar dentro s He aquí un ejemplo:

SELECCIONE Nombre, Apellido, Género, SPHome
De los estudiantes
WHERE (Apellido LIKE '% uno%')

Esto produciría:



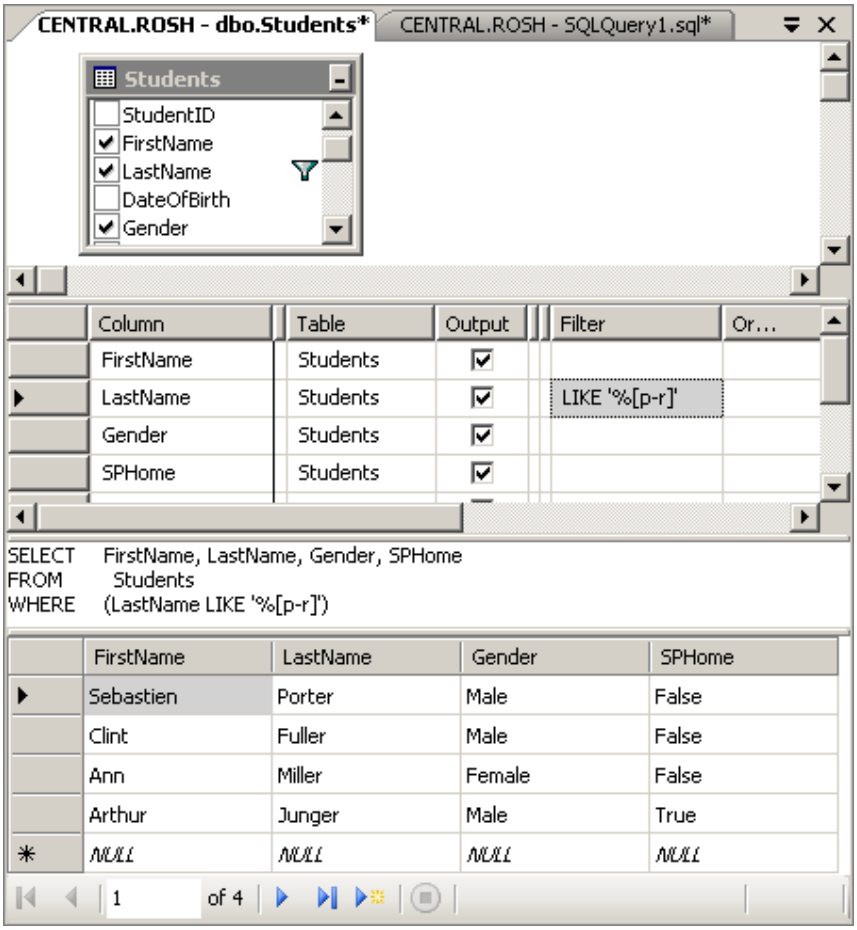
Al igual que los demás comandos SQL, también se puede negar esta utilizando el **operador** NOT.

COMO un rango de caracteres []

El **comodín%** se utiliza para preceder el éxito o un carácter específico o de un grupo de caracteres, e carácter. Si desea examinar únicamente una serie de caracteres del alfabeto, puede incluir el rango Para ello, escriba **[,** seguido por el carácter más bajo de la gama, seguido de -, seguido por el carácte gama, seguido **por].** Por ejemplo, para considerar el rango de notas entre s y p, que se utiliza **'[PS].** a la izquierda, a la derecha, o bien a ambos lados de esta expresión, **escriba%** para especificar s cualquier carácter o combinación de caracteres antes o después de la expresión. Aquí está un ejemplo:

SELECCIONE Nombre, Apellido, Género, SPHome
De los estudiantes
WHERE (Apellido LIKE '% [PS])

En el caso, el resultado sería una lista de los estudiantes cuyos apellidos terminan con p, q, r, o s. Esto



Observe que la lista sólo incluye a los estudiantes cuyos apellidos terminan con una letra de la p y s.

No finalizan con una serie de caracteres

En oposición a la consideración de los personajes que están en un rango específico, para especificar u serie de caracteres que no deben ser considerados, utiliza el carácter **^** dentro de los corchetes, perc rango deseado. Aquí está un ejemplo:

SELECCIONE Nombre, Apellido, Género, SPHome
De los estudiantes
WHERE (Apellido LIKE '% [^ pr]')

El resultado sería una lista de estudiantes cuya última final con una carta que no sea p, q, r, o s.

Una vez más, recuerde que usted puede negar esta expresión por la anterior con **NO.** Tenga en cue una expresión que incluyen ^, usted obtendría el mismo resultado que no utilicen ^.

Funciones y análisis de datos

Uso de una función

Para refinar su análisis de los datos, puede utilizar funciones, si las funciones que cree usted o la funciones incorporadas. Como siempre se menciona, los primeros candidatos de las funciones debe tr las funciones incorporadas, algunas de las cuales hemos examinado en **Lección 7.**

Para utilizar una función, en el marcador de posición de la columna, escriba el nombre de la funció paréntesis. Si la función tiene algunos parámetros, no olvide seguir las reglas para llamar a una funció Aquí hay un ejemplo que utiliza algunas basado en la fecha funciones incorporadas para mostrar la

alumnos:

SELECCIONE Nombre, Apellido, Género,
DATEDIFF (año, DateOfBirth, GETDATE ()) como la edad
De los estudiantes;
IR

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT FirstName, LastName, Gender,
       DATEDIFF(year, DateOfBirth, GETDATE()) AS Age
FROM Students;
GO
```

Results | Messages

	FirstName	LastName	Gender	Age
1	Sebastien	Porter	Male	12
2	Suzie	Hoak	Female	16
3	Antoinette	Clarck	Female	11
4	Koko	Domba	Male	17
5	Janet	West	Female	11
6	Catherine	Chang	Female	10
7	Nehemiah	Dean	Male	12
8	Sherryl	Ashburn	Female	12
9	Santos	Pacheco	Male	9
10	Mohamed	Husseini	Male	9
11	Dean	Chen	Male	10
12	Ruby	DeGaram	Female	9
13	Carole	Chance	Female	17
14	Justin	Vittas	Male	16
15	Ismael	Zara	Male	17
16	Anselme	Waters	Male	17

Q | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 rows

También puede incluir una función en cualquiera de los operadores se han revisado hasta el momen
ejemplo:

SELECCIONE Nombre, Apellido, Género, DateOfBirth, SPHome
De los estudiantes
WHERE (DateOfBirth ENTRE CONVERT (DATETIME,'1995-01-01 ', 102) Y
CONVERT (DATETIME,'1999-12-31 ', 102))

También puede implicar una función en una expresión.

Usando una función definida por el usuario

Si ninguna de las funciones incorporadas satisfaga sus necesidades, puede crear su propio y utilizarlo
los datos. Obviamente, primero debe crear la función. Aquí hay un ejemplo de dos funciones creadas
de datos:

/ * ==========
Autor: FunctionX

```
    Fecha de creación: viernes 6 de abril, 2007
    Descripción: Esta función se usa
                    para obtener el nombre completo de un estudiante
    ==============================================*/
CREATE FUNCTION GetFullName
(
        @ FName varchar (20),
        @ Lname varchar (20)
)
VUELVE varchar (41)
COMO
COMENZAR
        VOLVER lname @ + ',' + @ FName;
END;
IR
/ * ==============================================
    Autor: FunctionX
    Crear fecha: Sábado 7 de abril, 2007
    Descripción: Esta función se usa
                    para mostrar Sí o No
    ============================================== * /
CREATE FUNCTION ShowYesOrNo
(
     @ SPHomeStatus poco
)
VUELVE varchar (3)
COMO
COMENZAR
     DECLARE @ Resultado varchar (3);

     SI @ SPHomeStatus = 0
         SET @ Resultado = "No";
     ELSE
         SET @ Resultado = 'Sí';

     RETORNO @ resultados;
END;
IR
```

Una vez que una función está dispuesta, en el marcador de posición de su sentencia SQL, escriba **dbo** nombre de la función, sus paréntesis, y su paremeter (s), en su caso, dentro de los paréntesis. Aquí est

```
SELECCIONE StudentID,
        dbo.GetFullName (Nombre, Apellido) COMO [Nombre del estudiante],
        Género,
        dbo.ShowYesOrNo (SPHome) COMO [Live en un único Padre Inicio?],
        COMO ParentsNames [el nombre de los padres]
De los estudiantes;
IR
```

Esto produciría:

CENTRAL.ROSH - SQLQuery1.sql* | Object Explorer Details

```
SELECT StudentID,
       dbo.GetFullName(FirstName, LastName) AS [Student's Name],
       Gender,
       dbo.ShowYesOrNo(SPHome) AS [Live's in a Single Parent Home?],
       ParentsNames AS [Parents' Names]
FROM Students;
GO
```

Results | Messages

	StudentID	Student's Name	Gender	Live's in a Single Parent Home?	Parents' Names
1	1	Porter, Sebastien	Male	No	John and Christine Porter
2	2	Hoak, Suzie	Female	No	Christianne Hoak
3	3	Clarck, Antoinette	Female	No	Daniel and Henriette Clarck
4	4	Domba, Koko	Male	No	Dr. Julie and Mr. Jeremiah Domba
5	5	West, Janet	Female	Yes	Kirsten Farmer
6	6	Chang, Catherine	Female	No	Julie Best and Dr. Peter Chang
7	7	Dean, Nehemiah	Male	Yes	Marie Rodnat
8	8	Ashburn, Sherryl	Female	No	Shelia and Patrick Ashburn
9	9	Pacheco, Santos	Male	No	Dr. Julia Santana
10	10	Husseini, Moha...	Male	No	Drs. Phyllis and Ezra Husseini
11	11	Chen, Dean	Male	Yes	Alexandra Chen
12	12	DeGaram, Ruby	Female	No	Lynn and Anthony DeGaram
13	13	Chance, Carole	Female	No	Carole and Andy Chance
14	14	Vittas, Justin	Male	Yes	Clarice Vittas
15	15	Zara, Ismael	Male	No	Melinda and Robert Zara
16	16	Waters, Anselme	Male	Yes	Claudette and Dr. Celestin Waters
17	17	Lobo, Brenda	Female	No	Priscilla and Michel Lobo
18	18	Verde, Suzanna	Female	No	Suzanne and Thommy Verder

Query executed successful... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (51) | ROSH | 00:00:00 | 52 r

Otras Técnicas de Análisis de Datos

Agrupación

Contar los registros

SELECT COUNT (GenderID) COMO sexos
DE LAS PERSONAS

Cuente el número de registros sobre la base de un criterio, por ejemplo, contar el número de niñas en la e

SELECT COUNT (*) que a las niñas
De los estudiantes
WHERE (Sexo = 'Femenino')

La aplicación de un descuento a todos los artículos de la tienda:

SELECCIONE ItemNumber, ItemName, ItemSize, UnitPrice,

```
        UnitPrice - UnitPrice 0,20 * COMO [Descontado el precio]
 DE StoreItems
```

Usando CASO

```
 SELECCIONE ItemNumber, ItemName, ItemSize, ItemCategoryID, UnitPrice,
     CASO ItemCategoryID
         CUANDO 1 THEN UnitPrice - UnitPrice * 0.10
         CUANDO 2 THEN UnitPrice - UnitPrice * 0.50
         CUANDO ENTONCES UnitPrice 3 - UnitPrice * 0.35
         ELSE UnitPrice - UnitPrice * 0.45
     COMO FIN [Descontado el precio]
 DE StoreItems
```

Relaciones e integridad de los datos

La clave principal

Bases de datos relacionales

Una base de datos relacional es un sistema en el que los flujos de información de base de datos de un objeto a otro. Por ejemplo, en un banco de datos, puede utilizar un objeto para crear cuentas de clientes y el uso de otro objeto para procesar las transacciones que los propietarios de cuentas bancarias necesitan. La razón es que el mismo cliente pueda necesitar para llevar a cabo diversas operaciones, con regularidad. En lugar de crear una nueva cuenta cada vez que el cliente desea realizar una nueva transacción, puede utilizar una cuenta como referencia y criar a esta cuenta cuando el cliente quiere depositar o retirar dinero.

Para aplicar las normas de las bases de datos relacionales, se crean algunos tipos de relaciones entre los objetos de la base de datos.

Las transacciones entre los diversos objetos de una base de datos debe asegurarse de que la informa
es accesible a otro objeto. Los objetos que contienen información, como hemos mencionado ya, son las

Para gestionar el flujo de información de una tabla (A) a otra tabla (B), el cuadro que contiene la inforr
ponerlos a disposición de otros cuadros, como el B. Hay varias cuestiones que deben tratarse:

1. Usted debe ser capaz de identificar de forma exclusiva cada registro de una tabla (A) sin ningún ti
   Por ejemplo, si crea una lista de los coches en una mesa, debe asegurarse de que existe un único (
   número de etiqueta para cada vehículo, ya que cada auto debe tener y debe tener una etiqueta de
   Esto asegura que no existen registros duplicados en la mesa.
2. Un cuadro (A), que contiene información debe hacer que la información a disposición de los demás
   B)
3. Dos cuadros no deben servir para el mismo propósito. Una vez que haya información única en cad
   tabla puede hacer que sus datos a disposición de otros cuadros en los que la necesitan para que la
   información no debe ser inscrita en más de una tabla

Estos problemas se resuelven por una columna que especifique que la "clave" de la tabla. Dicha colum
clave principal.

En una base de datos relacional, que es el caso de la mayoría de las bases de datos que vas a cr
debería tener al menos una clave principal. Como ejemplo, una clave principal en un cuadro de cuenta
datos se puede establecer en un campo de cuenta bancaria, ya que cada cliente debe tener un único n
bancaria. Una tabla puede también utilizar más de una columna para representar a la clave principal
necesario.

Una vez que haya decidido que una mesa tendrá una clave principal, usted debe decidir qué tipo de da
sobre el terreno. Si usted está construyendo una tabla que puede utilizar un terreno conocido y evident
ejemplo podría ser la plataforma número uno de la biblioteca, puede definir su tipo de datos como **c**
hacer de él una clave principal. En muchos otros casos, por ejemplo, si usted no puede decidir sobre un
que tienen información única, un ejemplo sería clientes Nombre de Contacto, debe crear su propio camp
en la clave principal. Este campo debe tener un tipo de datos **int.**

Aprendizaje práctico: La introducción de Relaciones

1. Abrir SQL Server Management Studio y conectarse al servidor
2. Haga clic derecho en el nombre del servidor y haga clic en Nueva consulta
3. Para iniciar una nueva base de datos, escriba el siguiente código:

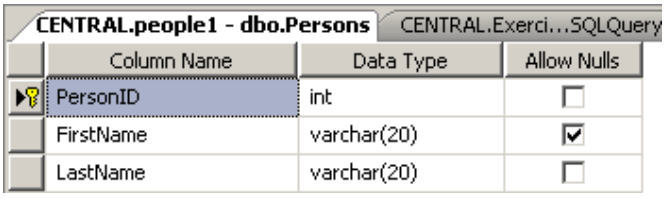
```
-
==========================================
- Base de datos: YugoNationalBank
- Autor: FunctionX
- Fecha de creación: lunes 09 abril 2007
-
==========================================
USO maestro
IR

- Excluir de la base de datos si ya existe
IF EXISTS (
        SELECT name
                DE sys.databases
                WHERE name = N'YugoNationalBank '
),
DROP DATABASE YugoNationalBank
IR

CREATE DATABASE YugoNationalBank
IR
```

Creación de una forma visual de clave principal

Para crear una clave principal en SQL Server Management Studio, crear una columna de datos y especif

- Luego, en la barra de herramientas, haga clic en el botón Establecer clave principal
- También puede hacer clic derecho en una columna y haga clic en Establecer clave principal

Aquí está un ejemplo:



Creación de una clave principal con SQL

Para crear una columna usando SQL primaria, la principal que hay que hacer es, en la parte derecha definición, de tipo **PRIMARY KEY.** Aquí está un ejemplo:

```
CREATE TABLE personas
(
    Int PersonID identidad (1,1) PRIMARY KEY NOT NULL,
    Nombre varchar (20),
    Apellido varchar (20) NOT NULL
);
```

La clave principal Restricción

En SQL, se puede dar un nombre específico a una clave principal. Para ello, se puede crear la colum algún lugar antes de que el paréntesis de cierre de la tabla, especifique la columna clave primaria med fórmula:

PrimaryKeyName CONSTRAINT PRIMARY KEY *(ColumnName)*

En esta fórmula, la palabra clave y el CONSTRAINT PRIMARY KEY (mayúsculas y minúsculas) c

necesarias. *PrimaryKeyName* en el marcador de posición, escriba el nombre que desea darle a la clave paréntesis de la clave primaria de expresión, escriba el nombre de la columna que se utilizará como l Aquí está un ejemplo:

```
CREATE TABLE personas
(
    Int PersonID identidad (1,1) NOT NULL,
    Nombre varchar (20),
    Apellido varchar (20) NOT NULL,
    PrimKeyPeople CONSTRAINT PRIMARY KEY (PersonID)
);
```

Por convención o tradición, el nombre de la primaria se inicia con PK_ seguida del nombre de la tab ejemplo:

```
USO Exercise2;
IR

CREATE TABLE personas
(
    Int PersonID identidad (1,1) NOT NULL,
    Nombre varchar (20),
    Apellido varchar (20) NOT NULL,
    PK_Persons CONSTRAINT PRIMARY KEY (PersonID)
);
IR
```

El Ministerio de Relaciones Exteriores clave

Introducción

Continuando con nuestro banco de datos, imaginar que un cliente llega al banco a depositar dir establecido que sería redundante para crear una nueva cuenta cada vez que el cliente llegue a realizar Por el contrario, usted tendría la información del cliente de su cuenta, facilitar dicha información a la ta procesar las transacciones. Como hemos descrito anteriormente, la consideración debe ser capaz de datos a los demás cuadros que se necesitan esos datos. Para que este flujo de información posible de debe crear una relación entre ellos.

Creación de una clave foránea en la tabla en Vista Diseño

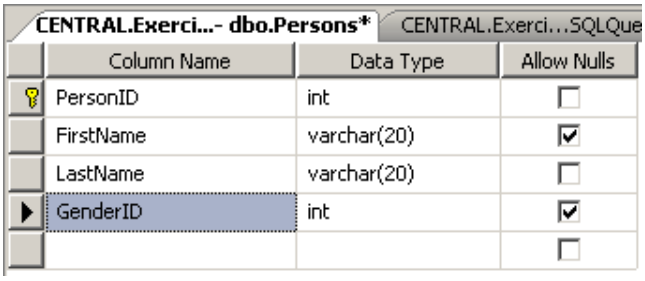
Para hacer posible que un cuadro B, para recibir datos de una tabla A, el cuadro B debe tener u representa la tabla A. Esta columna actúa como un "embajador" o un enlace. Como un pseudo-embaj en el cuadro B casi no pertenece a dicho cuadro: en primer lugar permite que ambas tablas a com motivo, la columna de la tabla B se llama una clave foránea.

Una clave externa es una columna en una tabla cuyos datos procedentes de otra tabla.

Para crear una clave foránea en la tabla de diseño de ventanas, en el cuadro que recibirá la clave, bas columna con las siguientes reglas:

- La columna debe tener el mismo nombre que la primera columna del cuadro que representa (pero requisito)
- La columna debe (es necesario) tener el mismo tipo de datos como la principal columna del cuadro

Aquí hay un ejemplo de una columna llamada GenderID que es una clave externa:



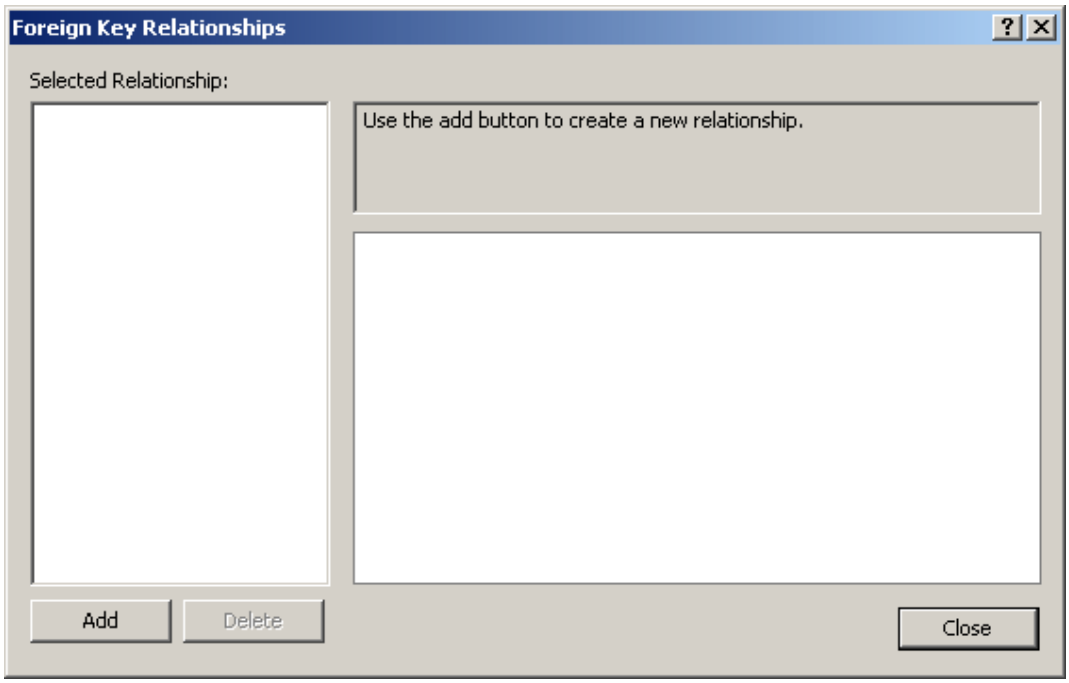
Obviamente, a fin de disponer de información que fluye de un cuadro a otro, el cuadro que contier primaria debe ser creado. Puede crear antes o después de crear la mesa de otros, siempre y cuando n ninguna relación entre ambas tablas, no importa lo que la secuencia que se utiliza para crearlas.

La tabla que contiene una clave principal y que contiene la información que se utiliza otra tabla que se tabla o la tabla padre. El cuadro que recibirá la información de la otra mesa se llama el cuadro exterior

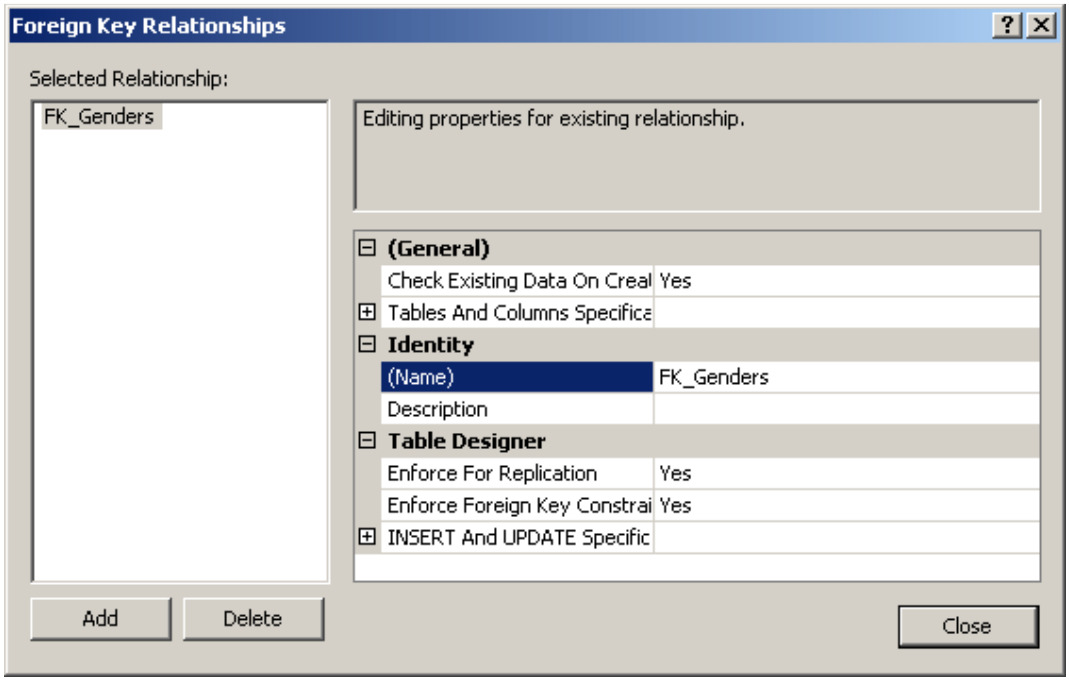
Creación de una clave externa en el cuadro de diálogo Relaciones de

Para crear una clave foránea en una tabla:

1. Desde el Explorador de objetos, abra la tabla hija en la vista Diseño
2. Clic con el botón derecho en cualquier lugar de la tabla y haga clic en Relaciones ...



3. En clave de Relaciones Exteriores de la caja de diálogo, haga clic en Agregar
4. Un nombre por defecto se sugiere a usted. Usted puede aceptar o cambiar. Para cambiar el nomb externa, en el lado derecho, ampliar y modificar la identidad de la cadena (nombre) sobre el terren



5. Si es necesario, de la misma manera, puede crear otras claves foráneas, haga clic en Agregar. Par clave foránea, primero seleccione en virtud de Relaciones seleccionados y haga clic en Eliminar. Una vez esté listo, haga clic en Cerrar

Creación de una clave externa en SQL

También puede crear una clave externa en SQL. La fórmula básica para usar es la siguiente:

REFERENCIAS *ParentTableName* FOREIGN KEY *(ForeignKeyCcolumn)*

La expresión **FOREIGN KEY** y **las** referencias de palabras clave son obligatorios. *ParentTableName* e posición, escriba el nombre de la tabla principal que contiene la información que se accede a ellos en la el paréntesis de *ParentTableName,* escriba el nombre de la principal columna de la tabla padre. Aquí est

```
CREATE TABLE personas
(
    Int PersonID identidad (1,1) PRIMARY KEY NOT NULL,
    Nombre varchar (20),
    Apellido varchar (20) NOT NULL,
    GenderID int NULL FOREIGN KEY REFERENCIAS géneros (GenderID)
);
```

La clave foránea

Observe que la clave foránea no tiene un nombre de objeto, como hemos visto para la clave principal. un nombre para la clave foránea, el intérprete de SQL se creará automáticamente un nombre prec usted. En caso contrario, para crear un nombre, después de crear la columna, escriba la palabra clav seguido por el nombre deseado y continuar el resto como hemos visto anteriormente. Ella es un ejempl

```
CREATE TABLE personas
(
    Int PersonID identidad (1,1) PRIMARY KEY NOT NULL,
    Nombre varchar (20),
    Apellido varchar (20) NOT NULL,
    GenderID int NULL CONSTRAINT FKGenders
                        FOREIGN KEY REFERENCIAS géneros (GenderID)
);
```

Establecer una relación

Introducción

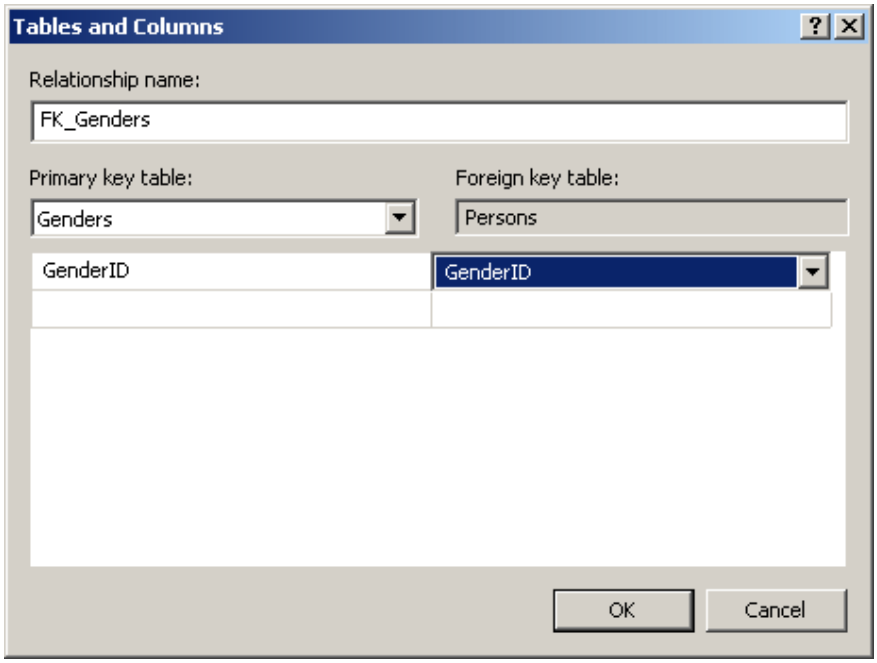
Como ya se ha mencionado, una base de datos relacional es aquella en la que los flujos de informació
otro. Para preparar las tablas para esto, crear la enseñanza primaria y claves foráneas, lo que hemos he
Una vez que los cuadros están listos, se puede vincular, lo que se refiere a la creación de una relación en

Si no crea una clave externa con código SQL, puede crear cuando se establece una relación entre dos ta

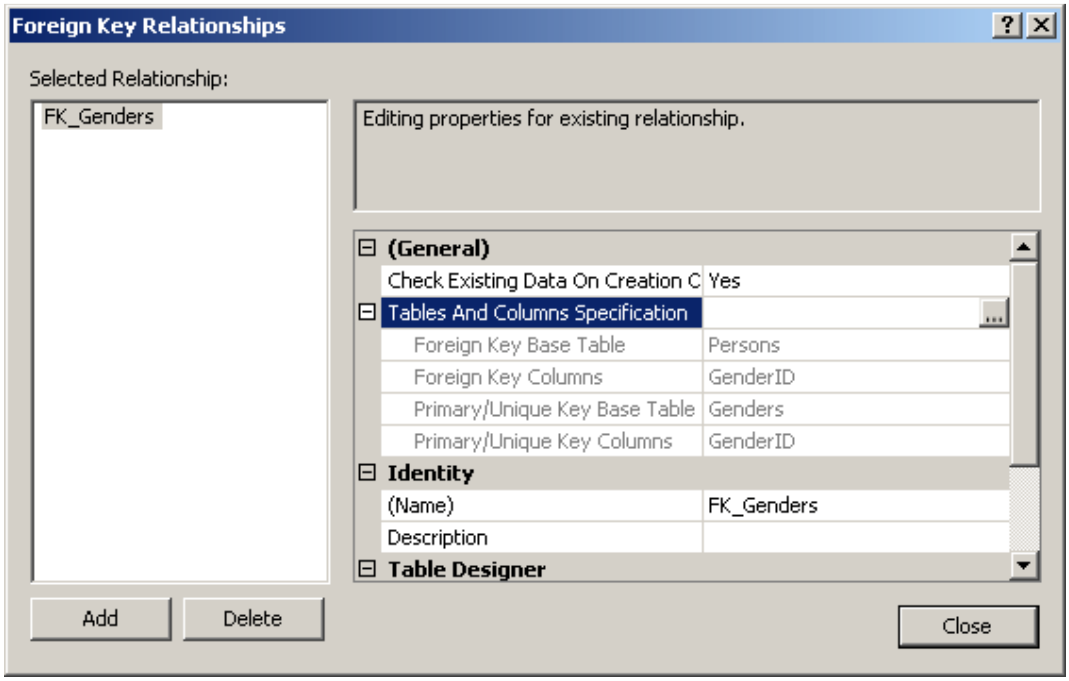
Creación de una Relación

Para crear una relación entre dos tablas

1. Abra el cuadro del niño vista del diseño
2. Haga clic con el botón (en cualquier parte de) la tabla y haga clic en Relaciones ...
   Si el (necesario) clave foránea no existe, haga clic en Agregar y especifique su nombre en virtud d
   el lado derecho.
3. Seleccionados en virtud de Relaciones, haga clic en la clave foránea que se mantenga la relación
4. En el lado derecho, expanda Especificación de tablas y columnas
5. Haga clic en su botón de puntos suspensivos ...
6. La clave principal en el cuadro desplegable, seleccione la tabla que contiene los datos primarios
7. En virtud de la tabla, haga clic y seleccione su clave principal columna
8. En virtud de la Tabla de Relaciones Exteriores clave, asegúrese de que el nombre actual de la tabla
   Bajo el nombre de la tabla hija, haga clic en y seleccione el nombre de la columna clave foránea. A
   ejemplo:



9. Haga clic en Aceptar.
   Cuando una relación se ha creado, se muestran en los cuadros y la especificación de columnas sec



10. De la misma manera, puede crear otro tipo de relaciones haciendo clic en Agregar y configurar el e
Una vez que haya terminado, haga clic en Cerrar

Aprendizaje práctico: Creación de Relaciones

1. Para completar nuestra base de datos con sus mesas, sus claves primarias, sus claves foráneas, y las tablas, cambiar el código en la ventana de consulta de la siguiente manera:

```
- ==============================================
- Base de datos: YugoNationalBank
- Autor: FunctionX
- Fecha de creación: lunes 09 abril 2007
- ==============================================
USO maestro
IR

- Excluir de la base de datos si ya existe
IF EXISTS (
        SELECT name
                DE sys.databases
                WHERE name = N'YugoNationalBank '
),
DROP DATABASE YugoNationalBank
IR

CREATE DATABASE YugoNationalBank
IR
- ==========================================
- Base de datos: YugoNationalBank
- Tabla: Localizaciones
- ==========================================
USO YugoNationalBank
IR

SI object_id ( 'dbo.Locations',' U ') IS NOT NULL
```

```
   DROP TABLE dbo.Locations
IR

CREATE TABLE Ubicaciones
(
      Int LocationID Identidad (1,1) NOT NULL,
      LocationCode varchar (10) NOT NULL,
      Dirección varchar (120),
      Ciudad varchar (50),
      Estado varchar (50),
      Notas de texto NULL,
      PK_Locations CONSTRAINT PRIMARY KEY (LocationID)
);
IR
- ==========================================
- Base de datos: YugoNationalBank
- Tabla: AccountTypes
- ==========================================
USO YugoNationalBank
IR

SI object_id ( 'dbo.AccountTypes',' U ') IS NOT NULL
   DROP TABLE dbo.AccountTypes
IR

CREATE TABLE AccountTypes
(
      Int AccountTypeID Identidad (1,1) NOT NULL,
      AccountType varchar (40) NOT NULL,
      Notas de texto NULL,
      PK_AccountTypes CONSTRAINT PRIMARY KEY (AccountTypeID)
);
IR

- ==========================================
- Base de datos: YugoNationalBank
- Tabla: Empleados
- ==========================================
SI object_id ( 'dbo.Employees',' U ') IS NOT NULL
   DROP TABLE dbo.Employees
IR
CREATE TABLE empleados
(
      IdEmpleado int identidad (1,1) NOT NULL,
      EmployeeNumber char (6),
      Nombre varchar (32),
      Apellido varchar (32) NOT NULL,
      Título varchar (50),
      CanCreateNewAccount poco,
      HourlySalary smallmoney,
      Direcciónde correo electrónico varchar (100),
      Nombre de usuario varchar (20),
      Contraseña varchar (20),
      Toma nota de texto,
      PK_Employees CONSTRAINT PRIMARY KEY (IdEmpleado)
);
IR
```

```
- ==========================================
- Base de datos: YugoNationalBank
- Tabla: Clientes
- ==========================================
SI object_id ( 'dbo.Customers',' U ') IS NOT NULL
   DROP TABLE dbo.Customers
IR
CREATE TABLE clientes
(
      IdCliente int Identidad (1,1) NOT NULL,
      DateCreated datetime,
      AccountNumber varchar (20),
      Restricción AccountTypeID int FK_TypeOfAccount
                        Referencias AccountTypes (AccountTypeID),
      CustomerName varchar (50) NOT NULL,
      DateUpdated smallDateTime,
      Toma nota de texto,
      PK_Customers CONSTRAINT PRIMARY KEY (IdCliente)
);
IR

- ==========================================
- Base de datos: YugoNationalBank
- Tabla: ChargeReasons
- ==========================================
SI object_id ( 'dbo.ChargeReasons',' U ') IS NOT NULL
   DROP TABLE dbo.ChargeReasons
IR
- ==========================================
- Base de datos: YugoNationalBank
- Tabla: Depósitos
- ==========================================
SI object_id ( 'dbo.Deposits',' U ') IS NOT NULL
   DROP TABLE dbo.Deposits
IR

CREATE TABLE Depósitos
(
      Int DepositID identidad (1, 1) NOT NULL,
      Restricción LocationID int FK_DepositLocation
                        Ubicaciones Referencias (LocationID) NOT NULL,
      Restricción IdEmpleado int FK_Clerk
                        Referencias Empleados (IdEmpleado),
      IdCliente int Restricción FK_Depositor
                        Referencias Clientes (IdCliente) NOT NULL,
      DepositDate smalldatetime NOT NULL,
      DepositAmount smallmoney NOT NULL,
      Toma nota de texto,
      PK_Deposits CONSTRAINT PRIMARY KEY (DepositID)
);
IR

- ==========================================
- Base de datos: YugoNationalBank
- Tabla: Retiros
- ==========================================
```

```
SI object_id ( 'dbo.Withdrawals',' U ') IS NOT NULL
   DROP TABLE dbo.Withdrawals
IR

CREATE TABLE Retiradas
(
      Int WithdrawalID identidad (1, 1) NOT NULL,
      Restricción LocationID int FK_WithdrawlLocation
                     Ubicaciones Referencias (LocationID) NOT NULL,
      Restricción IdEmpleado int FK_ProcessedBy
                     Referencias Empleados (IdEmpleado),
      IdCliente int Restricción FK_CustomerAccount
                     Referencias Clientes (IdCliente) NOT NULL,
      WithdrawalDate smalldatetime NOT NULL,
      WithdrawalAmount smallmoney NOT NULL,
      WithdrawalSuccessful poco NOT NULL,
      Toma nota de texto,
      PK_Withdrawas CONSTRAINT PRIMARY KEY (WithdrawalID)
);

- ==========================================
- Base de datos: YugoNationalBank
- Tabla: CheckCashing
- ==========================================
SI object_id ( 'dbo.CheckCashing', 'U') IS NOT NULL
   DROP TABLE dbo.CheckCashing
IR

CREATE TABLE CheckCashing
(
      Int CheckCashingID identidad (1, 1) NOT NULL,
      Restricción LocationID int FK_BranchLocations
                     Ubicaciones Referencias (LocationID) NOT NULL,
      Restricción IdEmpleado int FK_Employees
                     Referencias Empleados (IdEmpleado),
      IdCliente int Restricción FK_Customers
                     Referencias Clientes (IdCliente) NOT NULL,
      CheckCashingDate smalldatetime NOT NULL,
      CheckCashingAmount smallmoney NOT NULL,
      CheckCashingSuccessful poco NOT NULL,
      Toma nota de texto,
      PK_CheckCashing CONSTRAINT PRIMARY KEY (CheckCashingID)
);
IR
```

2. Presione la tecla F5 para ejecutar la declaración

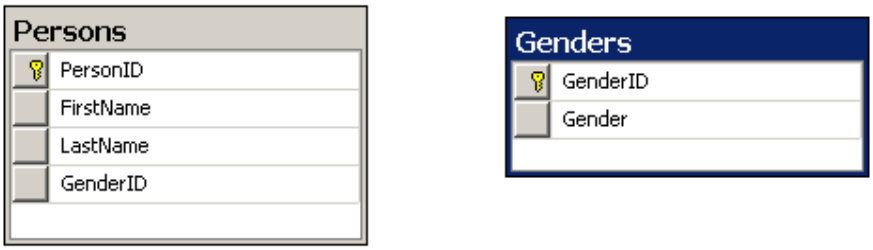
Diagramas

Un diagrama es una ventana que muestra visualmente las relaciones entre las tablas de una base de d
un diagrama:

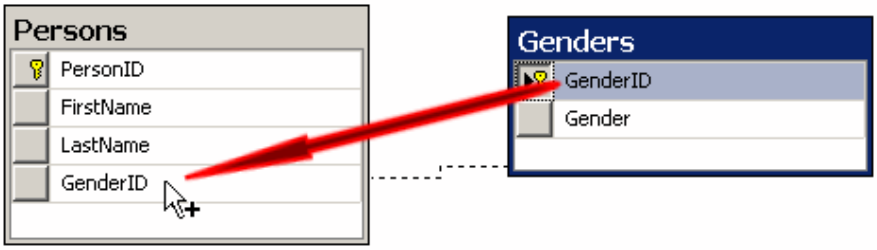
1. En el Explorador de objetos, en la base de datos de nodo, puede hacer clic en Base de datos de los
2. Un cuadro de diálogo le informará de que esta base de datos no dispone de un diagrama. Lea el m
   clic en Sí.
3. Haga clic con el botón base de datos de los diagramas y haga clic en Nuevo diagrama de base de d

4. En el cuadro de diálogo Agregar tabla, haga clic en cada cuadro y haga clic en Agregar. Alternativamente, se puede d ouble clic para añadir una tabla que

5. En el cuadro de diálogo Agregar tabla, puede hacer clic en Cerrar.
En la barra de herramientas, puede hacer clic en el botón Zoom y seleccione un mayor o menor va
Para mover una tabla, puede arrastrar la barra de título. Aquí está un ejemplo:



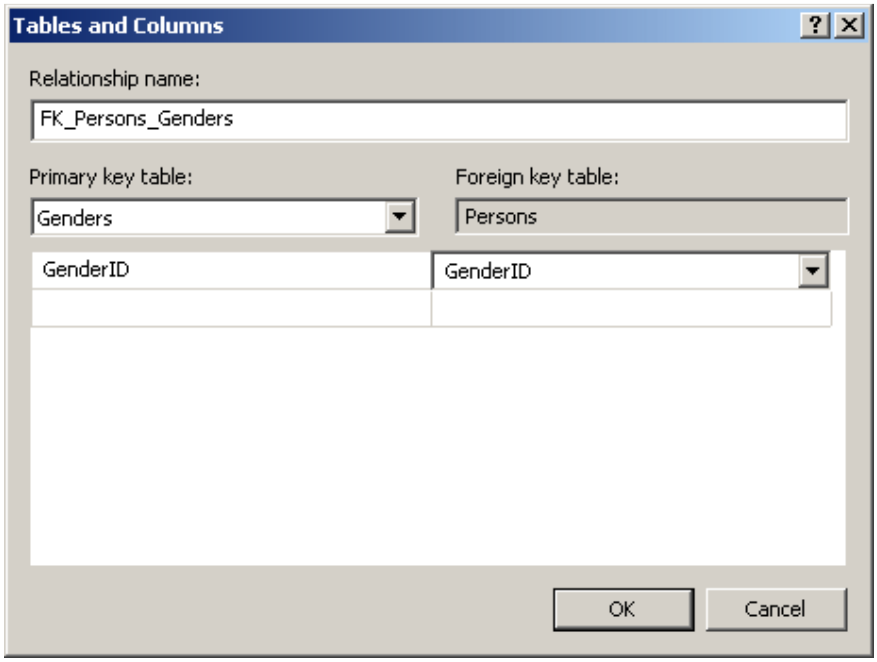
6. Para establecer una relación, puede hacer clic en el recuadro gris a la izquierda de cualquier colum padre y suéltelo en cualquier columna en la tabla hija. Una mejor manera es hacer clic en el recua clave principal columna de la tabla, arrastre el recuadro que luego suéltelo en la columna de clave e tabla hija. Aquí está un ejemplo:



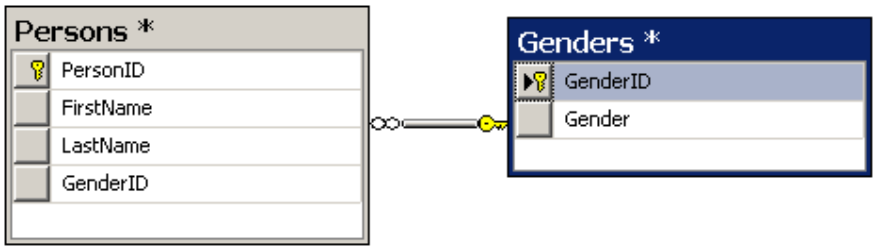
7. Cuadros y un cuadro de diálogo Columnas encontrarían. Que muestre la columna que fue arrastra sobre la que cayó.
Si había seleccionado a cualquier columna, se muestran pero no puede ser el que usted quería arra puede que no sea la columna que se supone que la gestión de la relación.
Independientemente, en virtud de la tabla de clave principal, usted debe seleccionar la tabla padre

8. En virtud de la tabla padre, seleccione su principal columna

9. En virtud de la Tabla de Relaciones Exteriores, seleccione la columna clave foránea. Aquí está un e



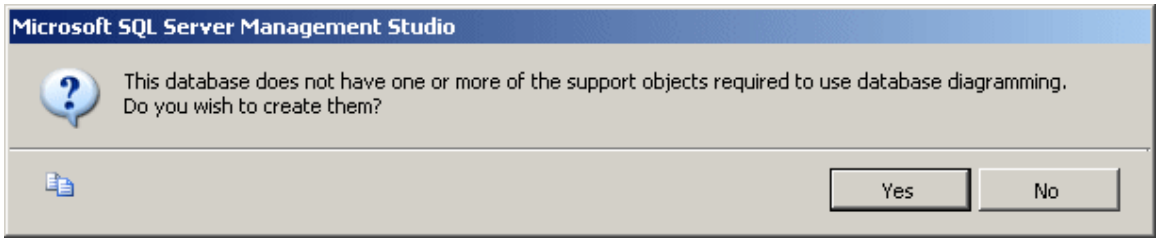
10. Una vez esté listo, haga clic en Aceptar. Un vínculo que se crea entre las tablas



11. De la misma manera, puede crear otro tipo de relaciones.
    Cuando haya terminado, puede guardar y cerrar la base de datos

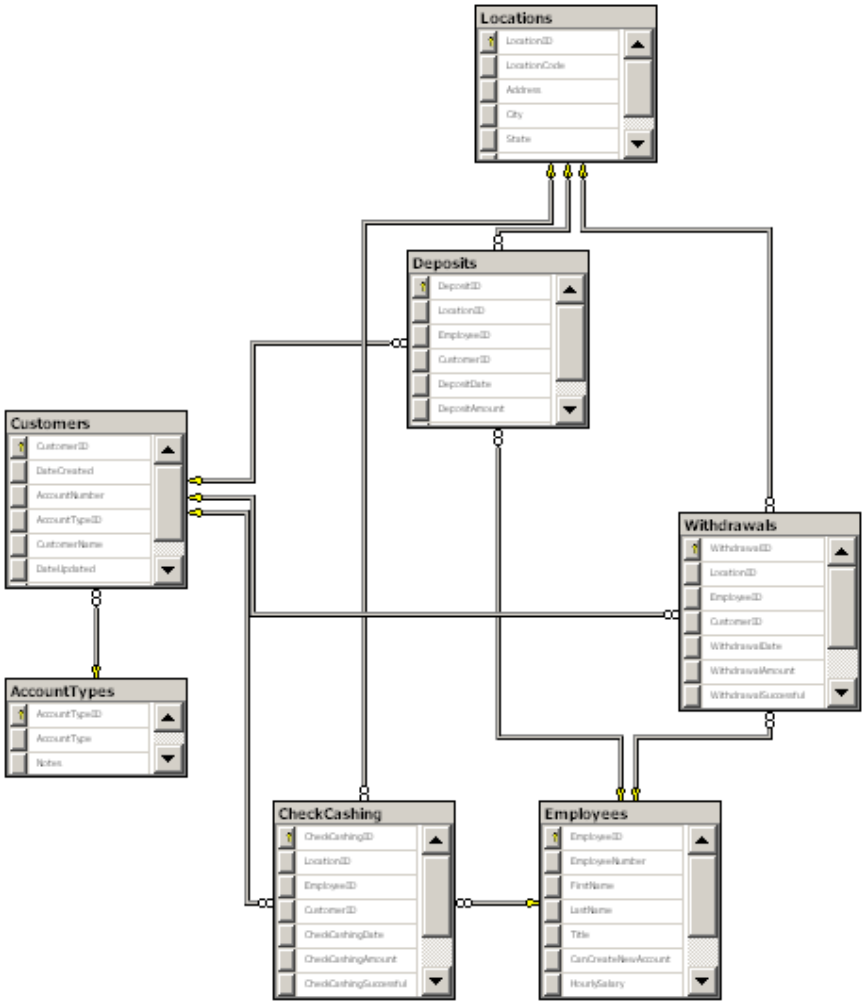
Aprendizaje práctico: Creación de un diagrama

1. En el Explorador de objetos, en el YugoNationalBank nodo, haga clic en Base de datos de los diagra
2. Un cuadro de diálogo le informará de que esta base de datos no tiene un diagrama:



   Lea el mensaje y haga clic en Sí.
3. Haga clic con el botón base de datos de los diagramas y haga clic en Nuevo diagrama de base de d
4. En el cuadro de diálogo Agregar tabla, haga clic en Cliente y haga clic en Agregar> botón
5. Haga doble clic para agregarlo CheckCashing
6. De la misma manera, agregar el AccountTypes, Depósitos, empleados, locales, mesas y Retiros

7. En el cuadro de diálogo Agregar tabla, haga clic en Cerrar.
   Observe que, sobre la base de cómo hemos creado la base de datos y sus objetos, las relaciones s



8. Para guardar el diagrama, en la barra de herramientas Estándar, haga clic en Guardar
9. Establezca su nombre a **dgmYugoNationalBank** y haga clic en Aceptar
10. Cerrar la ventana

Integridad referencial

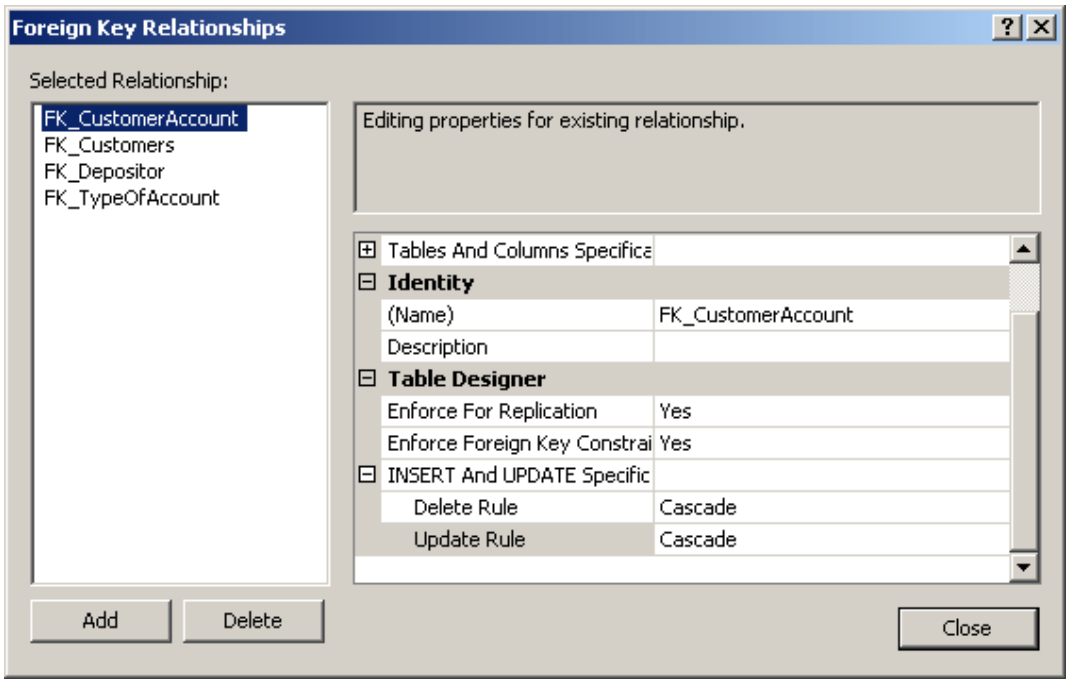
En una típica base de datos, la información va y viene. Para un banco, los clientes de cuentas se crea una base regular. Cuando se suprime una cuenta, existe la preocupación sobre las transacciones rela cuenta. Integridad referencial le permite gestionar estos aspectos de una base de datos. Usted debe a cuando los datos se elimina de una tabla padre, el niño se notifican las tablas y sus registros se borran t se modifique la información en una tabla padre, es la información relacionada con el niño cambiado en ta

Para la gestión de integridad referencial, se utiliza la clave de Relaciones Exteriores de cuadro de diálo acceder a él desde la vista diseño de una tabla o diagrama de la ventana.

Aprendizaje práctico: Gestión de la Integridad Referencial

1. En el Explorador de objetos, expanda el nodo Tablas en virtud YugoNationalBank.
   Haga clic con el botón y haga clic en Diseño dbo.Customers
2. Clic con el botón derecho en la tabla y haga clic en Relaciones

3. Seleccionados en virtud de Relaciones, haga clic en FK_CustomerAccount. En la sección derecha, e
   Especificación de INSERT y UPDATE
4. Haga clic en Eliminar regla. En su cuadro combinado, seleccione Cascada
5. Haga clic en Actualizar la Regla. En su cuadro combinado, seleccione en cascada:



6. De la misma manera, la especificación de los siguientes

FOREIGN KEY	**Eliminar regla**	**Actualización de la Regla**
FK_ChargeReasons	Cascada	Cascada
FK_Customers	Cascada	Cascada
FK_Depositor	Cascada	Cascada
FK_TypeOfAccount	Cascada	Cascada

7. Haga clic en Cerrar
8. Guarde y cierre el cuadro
9. De la misma manera, abra el cuadro Depósitos en vista diseño
10. Acceder a su cuadro de diálogo Relaciones de
11. El acceso de las propiedades de su FK_Clerk campo.
    Especifique su Eliminar regla y su norma de actualización tanto para cascada
12. Realizar las mismas acciones de las otras relaciones
13. Cierre el cuadro de diálogo Relaciones de
14. Guarde y cierre el cuadro

Se une a los datos

Une

Introducción

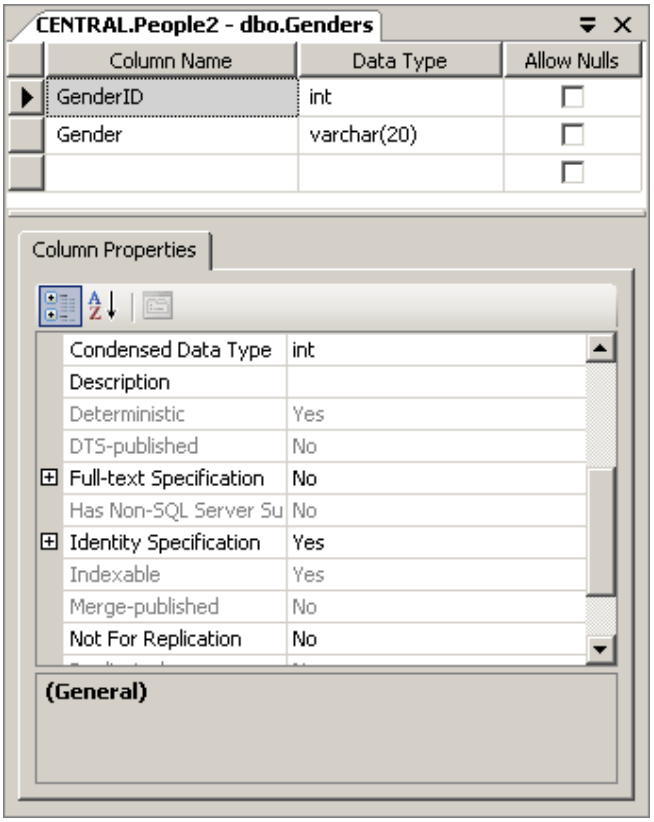
Al estudiar las relaciones, hemos revisado las técnicas de toma de datos de una tabla a disposición de los registros de otra tabla. Esta demostrado para reducir la duplicación de datos y errores. Otra cuestión que implica la combinación de tablas consiste en la creación de registros de más de una mesa y haciendo que el resultado en una sola lista. Esta es la base de datos une.

A unirse a los datos es una técnica de creación de una lista de registros de más de una mesa, utilizando todas las columnas de todas las tablas en cuestión, o seleccionar sólo las columnas que desee a partir de uno o todos de los cuadros en cuestión. Esto significa que una base de datos es esencialmente creado unirse en tres pasos:

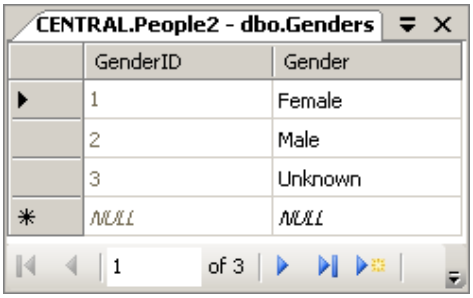
1. Los cuadros que participarán en la unión
2. Una columna que creará el enlace en cada cuadro
3. Una sentencia SQL que crearán los registros

Los cuadros de una parte de

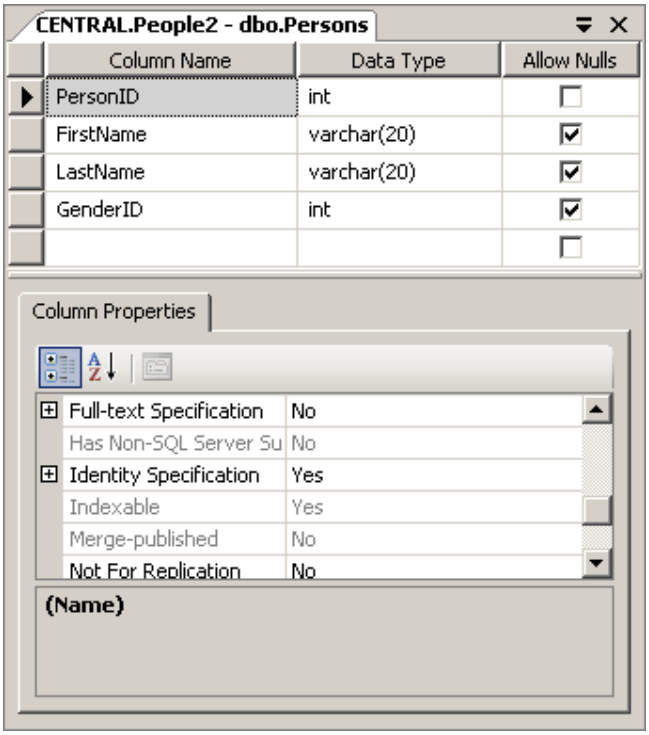
Antes de crear una unión, debe tener los cuadros en los que podría participar en ellos. Las tablas se cr técnicas que hemos visto en las lecciones anteriores. También es importante para crear una clave pr tabla. El padre de mesa por lo general sólo necesitan esta clave principal que se utiliza para "vincul mesa. Aquí está un ejemplo de este tipo de cuadro:



Si es necesario, puede crear los registros necesarios para la tabla. Aquí está un ejemplo:



Al crear la tabla hijo, recuerde que crear una columna que serviría de enlace con la tabla padre. costumbre como vimos al estudiar las relaciones, el nombre y el tipo de datos de esta columna son lo clave primaria de la tabla padre. Aquí hay un ejemplo de un niño de mesa que se sumaron a la tabla pa



Una vez más, en caso necesario, puede agregar la necesidad de registros a la tabla. Aquí está un ejem

CENTRAL.People2 - dbo.Persons

PersonID	FirstName	LastName	GenderID
1	Gertrude	Larson	1
2	Raymond	Kouma	NULL
3	Peter	Mukoko	2
4	Wally	Baston	2
5	Sylvia	Nguyen	NULL
6	Donald	Wallace	2
7	Hermine	Kana	1
8	Charlotte	Thomas	NULL
9	Paula	Barbers	NULL
10	Chrissie	Dentd	1
11	Ernestine	Essien	1
12	Barbara	Randt	NULL
13	Helene	Cranston	1
14	Robert	Palau	3
15	Paulette	Krazucky	1
16	Frank	Cranston	NULL
NULL	NULL	NULL	NULL

1 of 16

Aprendizaje práctico: Presentación de une

1. Inicie Microsoft SQL Server con el SQL Server Management Studio y conectarse al servidor
2. En el menú principal, haga clic en Archivo -> Nuevo -> Consulta de conexión con los actuales
3. Para preparar una base de datos de nuestros ejercicios, escriba lo siguiente:

```
- ==============================================
- Autor: FunctionX
- Base de datos: RealEstate2
- ==============================================
IF EXISTS (
   SELECT *
      DE sys.databases
     WHERE name = N'RealEstate2 '
),
   DROP DATABASE RealEstate2
IR
CREATE DATABASE RealEstate2;
IR

- ==============================================
- Autor: FunctionX
- Base de datos: RealEstate2
- Tabla: PropertyTypes
- ==============================================
USO RealEstate2;
IR
CREATE TABLE PropertyTypes
```

```
(
   Int PropertyTypeID identidad (1,1) NOT NULL,
   PropertyType varchar (20)
);
IR
INSERT INTO PropertyTypes (PropertyType)
VALUES ( 'Condominio');
IR
INSERT INTO PropertyTypes (PropertyType)
VALUES ( 'Individual Familia');
IR
INSERT INTO PropertyTypes (PropertyType)
VALUES ( 'Casa');
IR
INSERT INTO PropertyTypes (PropertyType)
VALUES ( 'Unknown');
IR

- ==============================================
- Autor: FunctionX
- Base de datos: RealEstate2
- Tabla: Condiciones
- ==============================================
USO RealEstate2;
IR
CREATE TABLE Condiciones
(
   Int ConditionID identidad (1,1) NOT NULL,
   Condición varchar (20)
);
IR
INSERT INTO Condiciones (Estado)
VALUES ( 'Excelente');
IR
INSERT INTO Condiciones (Estado)
VALUES ( 'Bueno');
IR
INSERT INTO Condiciones (Estado)
VALUES ( 'mal');
IR
INSERT INTO Condiciones (Estado)
VALUES ( 'Mayormente dañados');
IR

- ==============================================
- Autor: FunctionX
- Base de datos: RealEstate2
- Tabla: Propiedades
- ==============================================
CREATE TABLE Propiedades
(
   Int PropertyID identidad (1,1) NOT NULL,
   PropertyNumber char (6),
   Dirección varchar (100),
   Ciudad varchar (50),
   Estado char (2),
   Código postal varchar (12),
```

```
   PropertyTypeID int,
   ConditionID int,
   Dormitorios smallint,
   Baños de flotación,
   FinishedBasement poco,
   IndoorGarage poco,
   Historias smallint,
   YearBuilt smallint,
   MarketValue dinero
);
IR

INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('524880 ','1640 Ave. Lombardo', 'Silver Spring', 'MD',
   '20904 ', 2, 2, 4, 2,5, 3, 1, 3, 1995, 495,880.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('688364 ','10315 Norte Hacht Rd', 'College Park', 'MD',
         '20747 ', 2, 1, 4, 3.5, 3,
         1, 2, 2000, 620,724.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, FinishedBasement,
   Historias, MarketValue)
VALUES ('611464 ','6366 Lolita Drive', 'Laurel', 'MD',
         '20707 ', 2, 2, 1, 2, 422,625.00);
IR
INSERT INTO Propiedades (Dirección, Ciudad, PropertyTypeID,
   Dormitorios, MarketValue)
VALUES ('9002 Palasko Hwy », Tysons Corner,
         1, 2, 422,895.00);
IR
INSERT INTO Propiedades (PropertyNumber, Estado,
   Código postal, Dormitorios, YearBuilt, MarketValue)
VALUES ('420115 ',' CC ',
         '20011 ', 2, 1982, 312555);
IR
INSERT INTO Propiedades (PropertyNumber, ciudad, código postal,
  PropertyTypeID, Dormitorios, YearBuilt, MarketValue)
VALUES ('917203 ',' Alejandría ','22024',
         2, 3, 1965, 345,660.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   PropertyTypeID, ConditionID, Dormitorios, Baños, MarketValue)
VALUES ('200417 ','4140 Holisto Crt', 'Germantown', 'MD',
         1, 1, 2, 1, 215,495.00);
IR
INSERT INTO Propiedades (Ciudad, Estado, PropertyTypeID, ConditionID,
   Dormitorios, Baños, YearBuilt, MarketValue)
VALUES ( 'Rockville', 'MD', 1, 2, 2, 2, 1996, 436,885.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
```

```
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('927474 ','9522 Lockwood Road', 'Chevy Chase "," MD ",
         '20852 ', 3, 3, 3, 2.5, 3, 0, 3,
         1992, 415.665,00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('207850 ','14250 Parkdoll Road', 'Rockville "," MD ",
         '20854 ', 3, 2, 3, 2.5, 2, 1, 2,
         1988, 325.995,00);
IR
INSERT INTO Propiedades (Ciudad, PropertyTypeID, Dormitorios,
   YearBuilt, MarketValue)
VALUES ( 'Washington', 3, 4, 1975, 366,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   YearBuilt, MarketValue)
VALUES ('288540 ','10340 Helmes Street # 408', 'Silver Spring', 'MD',
         '20906 ', 1, 2, 1, 1, 2000, 242,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('247472 ','1008 Coppen Street', 'Silver Spring', 'MD',
         '20906 ', 2, 1,
         3, 3, 3, 1, 3, 1996, 625,450.00);
IR
INSERT INTO Propiedades (ciudad, código postal, PropertyTypeID,
    Historias, YearBuilt, MarketValue)
VALUES ( 'Chevy Chase','20956 ', 2,
         3, 2001, 525,450.00);
IR
INSERT INTO Propiedades (Dirección, Ciudad, Estado,
   PropertyTypeID, ConditionID, Dormitorios, MarketValue)
VALUES ('686 Herodes Ave. # D04 ',' Takoma Park ',' MD ',
         1, 1, 2, 360,885.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('297446 ','14005 Sniders Blvd.', 'Laurel', 'MD',
         '20707 ', 3, 4,
         4, 1.5, 3, 1, 2, 2002, 412,885.00);
IR
INSERT INTO Propiedades (ciudad, código postal, ConditionID, Dormitorios,
   Historias, YearBuilt)
VALUES ( 'Silver Spring','20905 ', 2,
         4, 2, 1965);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('924792 ','680 Prushia Road', 'Washington', 'CC',
```

```
        '20008 ', 2, 2,
        5, 3.5, 3, 0, 3, 2000, 555,885.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('294796 ','14688 Parrison Street', 'College Park', 'MD',
        '20742 ', 2, 1,
        5, 2.5, 2, 1, 2, 1995, 485,995.00);
IR
INSERT INTO Propiedades (Ciudad, Estado, PropertyTypeID, ConditionID,
   Dormitorios, Baños, YearBuilt, MarketValue)
VALUES ( 'Rockville', 'MD', 1, 2, 1, 1, 1996, 418,885.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   YearBuilt, MarketValue)
VALUES ('811155 ','10340 Helmes Street # 1012', 'Silver Spring »,
        'MD','20906 ', 1, 2,
        1, 1, 2000, 252,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('447597 ','4201 Vilamar Ave', 'Hyattsville', 'MD',
        '20782 ', 3, 1,
        3, 2, 2, 1, 3, 1992, 365,880.00);
IR
INSERT INTO Propiedades (Dirección, código postal, Baños)
VALUES ('1622 Rombard Str., 20904, 2.5);
IR
INSERT INTO Propiedades (Ciudad, Estado, PropertyTypeID, ConditionID,
   Dormitorios, Baños, YearBuilt, MarketValue)
VALUES ( 'Rockville', 'MD', 1, 2, 1, 1, 1996, 420,555.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('297415 ','980 Phorwick Street', 'Washington', 'CC',
        '20004 ', 2, 2,
        4, 3.5, 3, 3, 1, 2004, 735,475.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('475974 ','9015 Marvin Ave. Crow', 'Gaithersburg', 'MD',
        '20872 ', 2, 4,
        4, 2.5, 3, 1, 1, 1965, 615,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('836642 ','3016 Feldman Corte', 'Rockville "," MD ",
        '20954 ', 2, 3,
        5, 3, 3, 1, 3, 1960, 528,555.00);
IR
```

```
INSERT INTO Propiedades (Dirección, ciudad, código postal, PropertyTypeID,
   Dormitorios, Baños, MarketValue)
VALUES ('2444 Arielson Road ',' Rockville ','20854', 1, 2, 1, 1996, 475,555.00);
IR

INSERT INTO Propiedades (Ciudad, Estado, PropertyTypeID, Historias)
VALUES ( 'Rockville "," MD ",
         3, 1);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('208304 ','7307 Everett Hwy', 'Washington', 'CC',
         '20012 ', 3, 1,
         2, 2.5, 2, 0, 4, 2006, 420,550.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios,
   Baños, YearBuilt, MarketValue)
VALUES ('644114 ','10340 Helmes Street # 1006', 'Silver Spring »,
         'MD','20906 ', 1, 2,
         2, 2, 2000, 258,445.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('937966 ','7303 Warfield Corte "," Tysons Corner', 'VA',
         '22131 ', 2, 2,
         3, 2.5, 3, 1, 4, 2006, 825,775.00);
IR
INSERT INTO Propiedades (ciudad, código postal, ConditionID, Dormitorios,
   Historias, YearBuilt)
VALUES ( 'Fairfax','22232 ', 2, 3, 3, 1985);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('297497 ','12401 Conniard Ave', 'Takoma Park', 'MD',
         '20910 ', 3, 2,
         3, 2.5, 3, 1, 3, 2004, 280,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, ciudad, código postal,
   PropertyTypeID, ConditionID, Dormitorios, Baños,
   YearBuilt, Historias, MarketValue)
VALUES ('855255 ',' Laurel ','20707', 2,
         4, 3, 2, 1962, 2, 342,805.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('469750 ','6124 Falk Road', 'Arlington', 'VA',
         '22031 ', 2, 4,
         4, 3.5, 3, 1, 1, 1982, 635,995.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
```

```
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('826927 ','5121 Riehl As',' Fairfax ',' VA ',
        '22232 ', 3, 1,
        3, 1.5, 2, 0, 1, 2002, 325,620.00);
IR
INSERT INTO Propiedades (ciudad, código postal, PropertyTypeID, Dormitorios,
   Baños, MarketValue)
VALUES ( 'Silver Spring','20906 ', 1, 2, 2, 335,655.00);
IR

INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('287064 ','9533 Pensulian Road', 'Silver Spring', 'MD',
        '20904 ', 2, 3,
        3, 1.5, 3, 1, 2, 1992, 485,775.00);
IR
INSERT INTO Propiedades (PropertyNumber, ciudad, código postal,
   PropertyTypeID, ConditionID, Dormitorios, YearBuilt, Historias)
VALUES ('724001 ','705 Helios Ave','20004 ',
        3, 3, 3, 1974, 4);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('209275 ','944 Freidora Ave', 'Chevy Chase "," MD ",
        '20852 ', 2, 1,
        5, 2.5, 3, 0, 2, 2002, 625,665.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('204759 ','1950 Galego Street', 'Germantown', 'MD',
        '20874 ', 2, 1,
        4, 3.5, 2, 1, 4, 2007, 428,665.00);
IR
INSERT INTO Propiedades (PropertyNumber, Dirección, Ciudad, Estado,
   Código postal, PropertyTypeID, ConditionID, Dormitorios, Baños,
   FinishedBasement, IndoorGarage, Historias, YearBuilt, MarketValue)
VALUES ('937259 ','12366 Fowler Ave', 'Alejandría', 'VA',
        '22031 ', 3, 2,
        3, 1.5, 3, 1, 3, 2007, 402,815.00);
IR
```

4. Presione la tecla F5 para ejecutar
5. Guarde el archivo como RealEstate2 y cerrar la ventana de consulta

Únete a la Creación

Equipado con las tablas necesarias y sus columnas, puede crear la unión. Para hacer esto en Management Studio, usted puede hacer clic derecho en uno de los cuadros implicados y haga clic en mostrará la ventana de Tabla. Usted debe mostrar el diagrama y las secciones de SQL. Debido a que una unión se encuentra en al menos dos tablas, debe agregar una. Para ello

- En el menú principal, puede hacer clic en Diseñador de consultas -> Añadir Cuadro
- Diseñador de consultas sobre la barra de herramientas, puede hacer clic en el botón Agregar Tabla

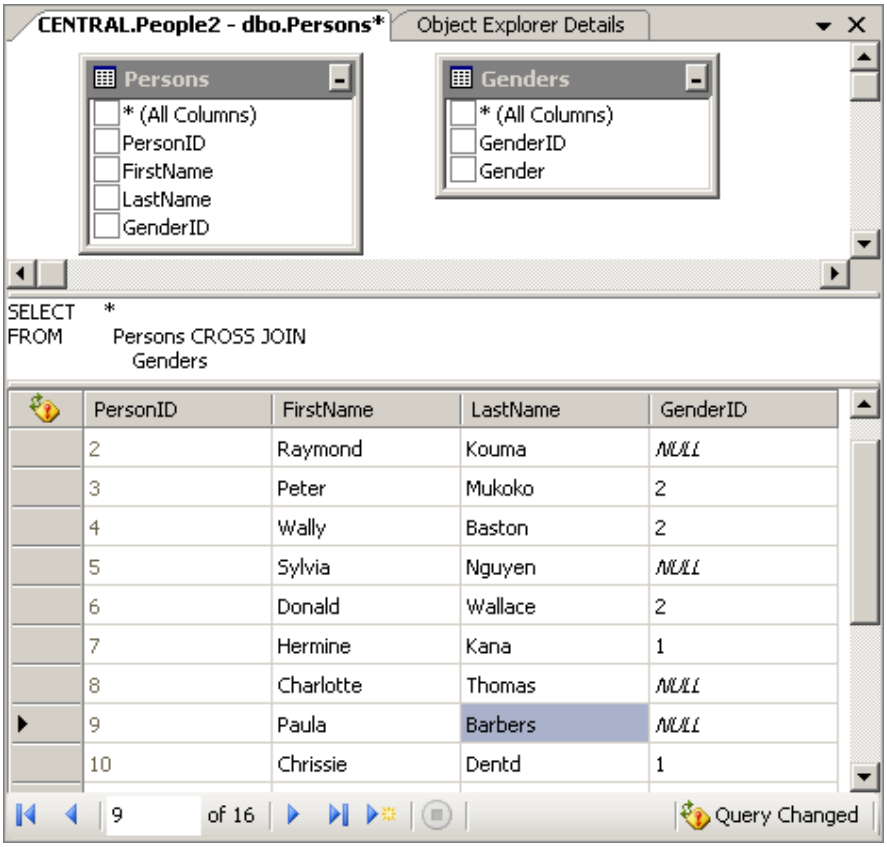
- Puede hacer clic derecho en el diagrama de la sección de la ventana y haga clic en Agregar tabla ..

Cualquiera de estas acciones se mostrará el cuadro de diálogo Agregar tabla. Para seleccionar una tabl

- Puede hacer clic en el nombre de la tabla y haga clic en Añadir
- Puede hacer doble clic en un cuadro

Alternativamente, en lugar de utilizar Añadir cuadro, puede arrastrar el cuadro del niño Explorer y caída diagrama de la sección.

Aquí hay un ejemplo de dos cuadros que se han añadido:



Recuerde que usted puede arrastrar las barras de título de los cuadros para moverlos y la posición a su de seleccionar la tabla (s), en el cuadro de diálogo Agregar tabla, puede hacer clic en Cerrar.

Si una relación ya establecida entre las tablas, una línea que une mostrarla.

En código SQL, la fórmula básica para crear una unión es:

SELECCIONE *WhatColumn (s)*
DE *ChildTable*
TypeOfJoin ParentTable
A *condición de*

El factor de *ChildTable* especifica la tabla que contiene los registros que se recuperarán. Puede ser rep siguiente manera:

SELECCIONE *WhatColumn (s)*
DE LAS PERSONAS
TypeOfJoin ParentTable

A *condición de*

El factor de *ParentTable* especifica la tabla que sostiene la columna con la clave principal que cor registros relacionados con el niño el cuadro, que se mostrará. Este factor estaría representado de la sigu

SELECCIONE *WhatColumn (s)*
DE LAS PERSONAS
TypeOfJoin los géneros
SOBRE Persons.GenderID = Genders.GenderID

La *condición* es un factor de lógica expresión utilizada para validar los registros que serán aislado condición, debe asignar la clave principal columna de la tabla padre a la columna de clave externa Probablemente porque las dos columnas tienen el mismo nombre, para distinguirlos, sus nombres debe Esto se haría de la siguiente manera:

SELECCIONE *WhatColumn (s)*
DE LAS PERSONAS
TypeOfJoin los géneros
SOBRE Persons.GenderID = Genders.GenderID

El *WhatColumn (s)* de factor de nuestra fórmula le permite hacer una lista de las columnas que de declaración. Como debe ser consciente, puede incluir todas las columnas utilizando el operador *. Aquí

SELECT *
DE LAS PERSONAS
TypeOfJoin los géneros
SOBRE Persons.GenderID = Genders.GenderID

En este caso, todas las columnas de todas las tablas se incluirán en el resultado. En lugar de todas las que quiera una lista restringida. En este caso, crear la lista después de la **selección** de palabras clave comas. Puede utilizar el nombre de una columna que normalmente si el nombre no es más que columna. Aquí está un ejemplo:

SELECCIONAR Apellido, Nombre, Sexo
DE LAS PERSONAS
TypeOfJoin los géneros
SOBRE Persons.GenderID = Genders.GenderID

Si el mismo nombre de una columna se encuentra en más de una tabla, como es el caso de una pr combinación de teclas, usted debe calificar el nombre. Aquí está un ejemplo:

SELECCIONAR Apellido, Nombre, Persons.GenderID,
Genders.GenderID, Género
DE LAS PERSONAS
TypeOfJoin los géneros
SOBRE Persons.GenderID = Genders.GenderID

De hecho, para hacer su código más fácil de leer, debe calificar el nombre de cada columna de su **SELE** ejemplo:

SELECCIONE Persons.LastName, Persons.FirstName, Persons.GenderID,
Genders.GenderID, Genders.Gender
DE LAS PERSONAS
TypeOfJoin los géneros
SOBRE Persons.GenderID = Genders.GenderID

Aprendizaje práctico: Preparación de una parte de

1. En el Explorador de objetos, haga clic derecho en las bases de datos y haga clic en Actualizar
2. Ampliar las bases de datos y ampliar RealEstate2

3. Ampliar sus cuadros de nodo
4. Haga clic con el botón Propiedades de la tabla y haga clic en Abrir tabla
5. Diseñador de consultas sobre la barra de herramientas, haga clic en el botón Mostrar panel Diagra

   botón Mostrar panel SQL
6. Consulta sobre la barra de herramientas de diseño, haga clic en el botón Agregar Tabla
7. En el cuadro de diálogo Agregar tabla, haga doble clic en PropertyTypes
8. Haga clic en Cerrar

Interior y se une a la Cruz

Introducción

Al estudiar las relaciones de datos, hemos visto el papel de la enseñanza primaria y claves foráneas intercambio de información entre dos tablas. Esta técnica de vincular cuadros desempeña un papel im una unión. Le permite decidir si desea incluir todos los registros o sólo aislar algunos de ellos. Respe de una relación entre dos tablas, ya que se aplica a una consulta Transact-SQL admite tres tipos de junt

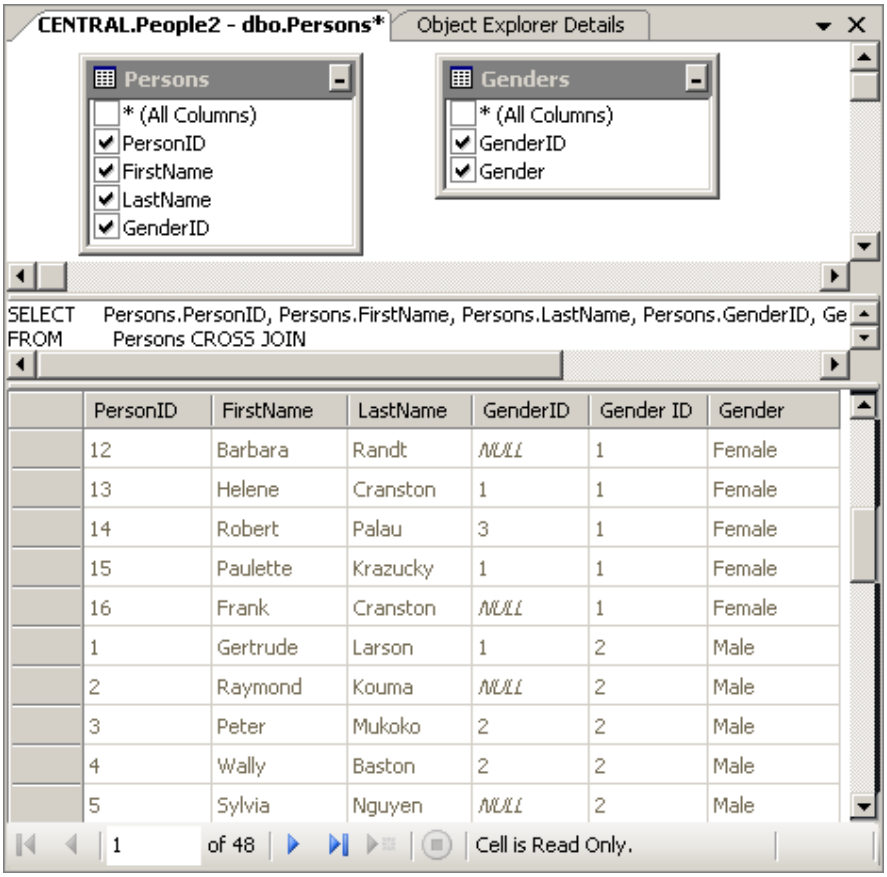
Cruz se une a

Una cruz unirse crea una lista de todos los registros de ambas tablas de la siguiente manera: el prin tabla padre se asocia a cada registro de la tabla hija, entonces el segundo disco de la tabla padre registro de la tabla hija, y así sucesivamente. En este caso también, no hay necesidad de una colur ambas tablas. En otras palabras, usted no va a usar **en** la cláusula.

Para crear una cruz unirse, puede reemplazar el factor de *TypeOfJoin* nuestra fórmula con **ÚNASE OUTER JOIN.** He aquí un ejemplo:

```
SELECCIONE Persons.PersonID, Persons.FirstName, Persons.LastName,
                Genders.GenderID, Genders.Gender
DE LAS PERSONAS
CRUZ ÚNASE géneros
IR
```

Por defecto, desde el SQL Server Management Studio, después de que acaba de añadir una tabla a ot relación ya establecida entre ambas tablas), la consulta sería automáticamente a unirse a una cruz. T que hacer es seleccionar las columnas necesarias. Después de seleccionar las columnas, puede ejecuta ver el resultado:



Interior se une a

Imagina que tienes dos tablas que pueden ser enlazados a través de una clave principal y otra exterior

CENTRAL.People2 - dbo.Genders

GenderID	Gender
1	Female
2	Male
3	Unknown
NULL	NULL

1 of 3

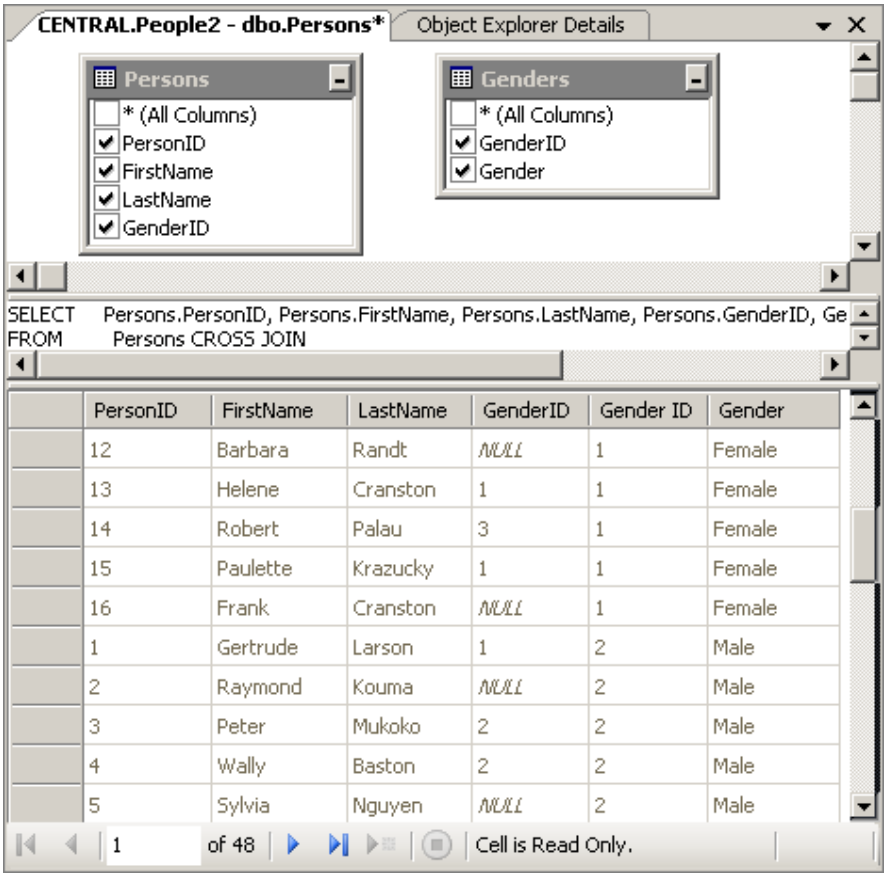
CENTRAL.People2 - dbo.Persons

PersonID	FirstName	LastName	GenderID
1	Gertrude	Larson	1
2	Raymond	Kouma	NULL
3	Peter	Mukoko	2
4	Wally	Baston	2
5	Sylvia	Nguyen	NULL
6	Donald	Wallace	2
7	Hermine	Kana	1
8	Charlotte	Thomas	NULL
9	Paula	Barbers	NULL
10	Chrissie	Dentd	1
11	Ernestine	Essien	1
12	Barbara	Randt	NULL
13	Helene	Cranston	1
14	Robert	Palau	3
15	Paulette	Krazucky	1
16	Frank	Cranston	NULL
NULL	NULL	NULL	NULL

1 of 16

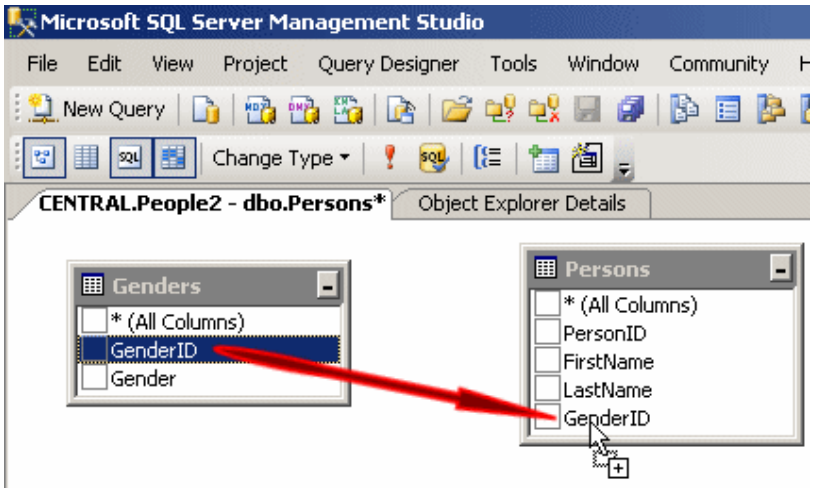
Observe que algunos registros en la tabla de personas no tienen una entrada para la columna y marcados con *NULL* por el motor de base de datos. Al crear una consulta de los registros de la tabla usted quiere que su lista para incluir sólo los registros que han una entrada, puede crear como inner join

Por defecto, desde el SQL Server Management Studio, al crear una nueva consulta, si una relación ya ambas tablas, la consulta se hace un inner join. Si no hay relación explícitamente establecido ent tendría que crearlo editar la sentencia SQL. Considere lo siguiente:



Observe que, debido a que no se había establecido la relación entre ambas tablas, la unión se cruzaron.

Para crear un inner join, tiene dos opciones. Puede arrastrar la clave principal de la tabla padre y su foránea en la tabla hija. Aquí está un ejemplo:

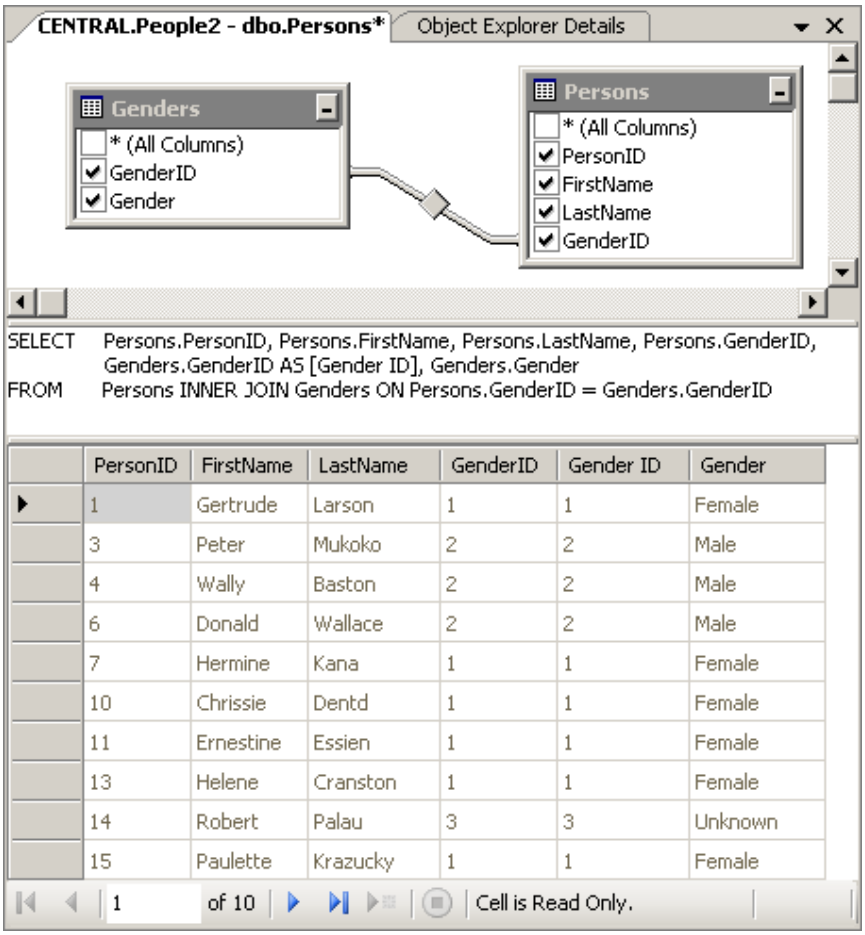


Si lo prefiere, puede editar la sentencia SQL manualmente para que sea un inner join. Para ello, s *TypeOfJoin* factor de nuestra fórmula con la expresión **INNER JOIN.** He aquí un ejemplo:

SELECCIONE Persons.PersonID, Persons.FirstName, Persons.LastName, Persons.GenderID,
COMO Genders.GenderID [Género ID], Genders.Gender

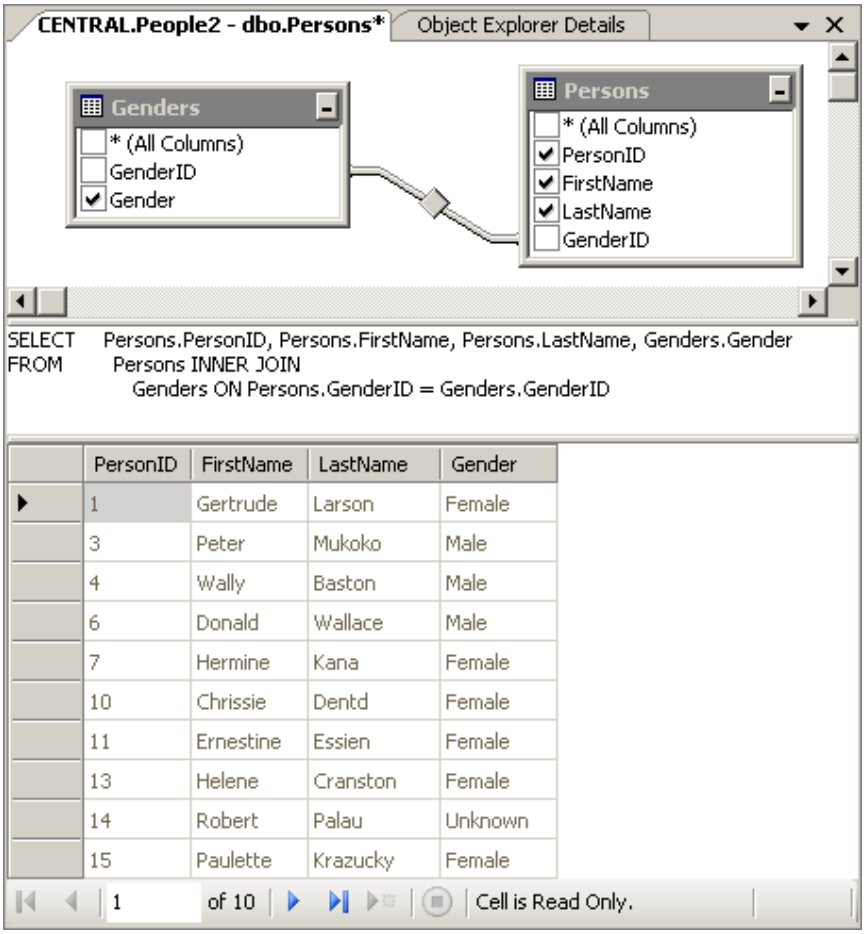
DE LAS PERSONAS DE INNER JOIN géneros Persons.GenderID = Genders.GenderID

Después de crear la unión, en el diagrama de la sección, una línea que se creó para unir las tablas. puede ejecutar la consulta para ver el resultado. Esto produciría:



PersonID	FirstName	LastName	GenderID	Gender ID	Gender
1	Gertrude	Larson	1	1	Female
3	Peter	Mukoko	2	2	Male
4	Wally	Baston	2	2	Male
6	Donald	Wallace	2	2	Male
7	Hermine	Kana	1	1	Female
10	Chrissie	Dentd	1	1	Female
11	Ernestine	Essien	1	1	Female
13	Helene	Cranston	1	1	Female
14	Robert	Palau	3	3	Unknown
15	Paulette	Krazucky	1	1	Female

Hemos mencionado anteriormente que puede incluir todas las columnas en la consulta. En nuestro r están más interesados en el género de cada registro de las personas, no necesitaríamos GenderID la co de los géneros. Aquí está un ejemplo:



Como se mencionó anteriormente, la notificación de que el resultado sólo incluye los registros que tie (una entrada no NULL) en la columna clave foránea GenderID de la mesa de las personas.

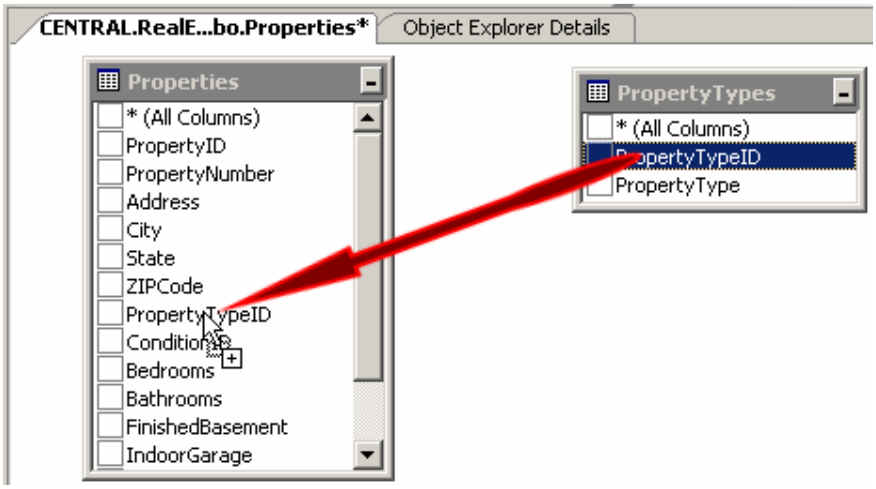
Una alternativa a la expresión **INNER JOIN** es simplemente el tipo **de JOIN.** Aquí está un ejemplo:

```
SELECCIONE Persons.PersonID, Persons.FirstName, Persons.LastName,
          Genders.Gender
DE LAS PERSONAS
ÚNASE géneros
SOBRE Persons.GenderID = Genders.GenderID
IR
```

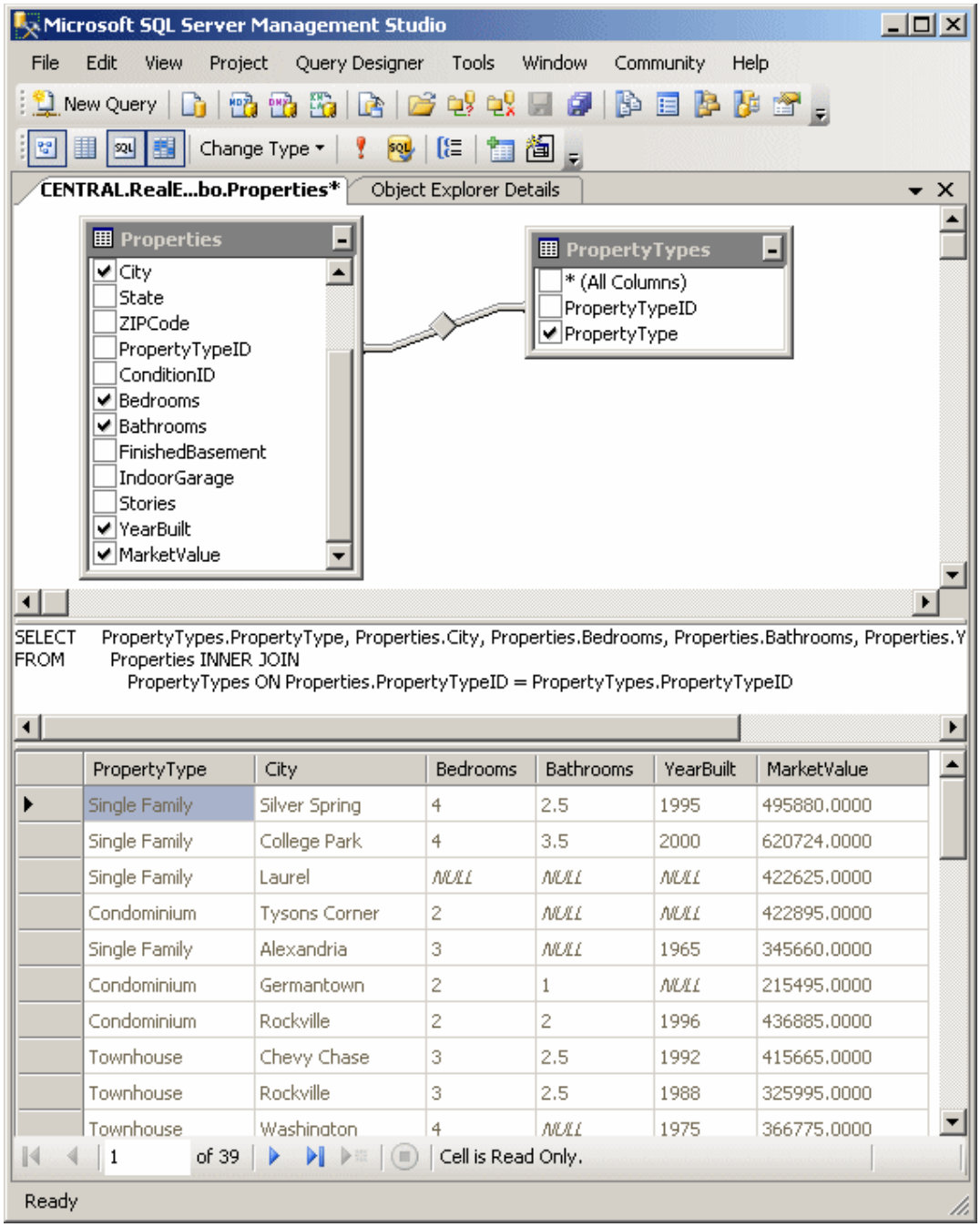
Para destruir una unión entre dos tablas, si se está trabajando en el cuadro ventana, puede hacer c línea que une las tablas y haga clic en Eliminar. En SQL, debe modificar las expresiones que conforman y ON expresiones).

Aprendizaje práctico: Creación de un INNER JOIN

1. Para crear un inner join, PropertyTypes de la mesa, PropertyTypeID arrastrar y soltar en el campo PropertyTypeID Propiedades de tabla:



2. Suelte el ratón
3. En las tablas, seleccione los campos siguientes: PropertyType, Ciudad, Dormitorio, Baño, YearBuilt
4. Diseñador de consultas sobre la barra de herramientas, haga clic en el botón Ejecutar para ver el r



Exterior se une a

Introducción

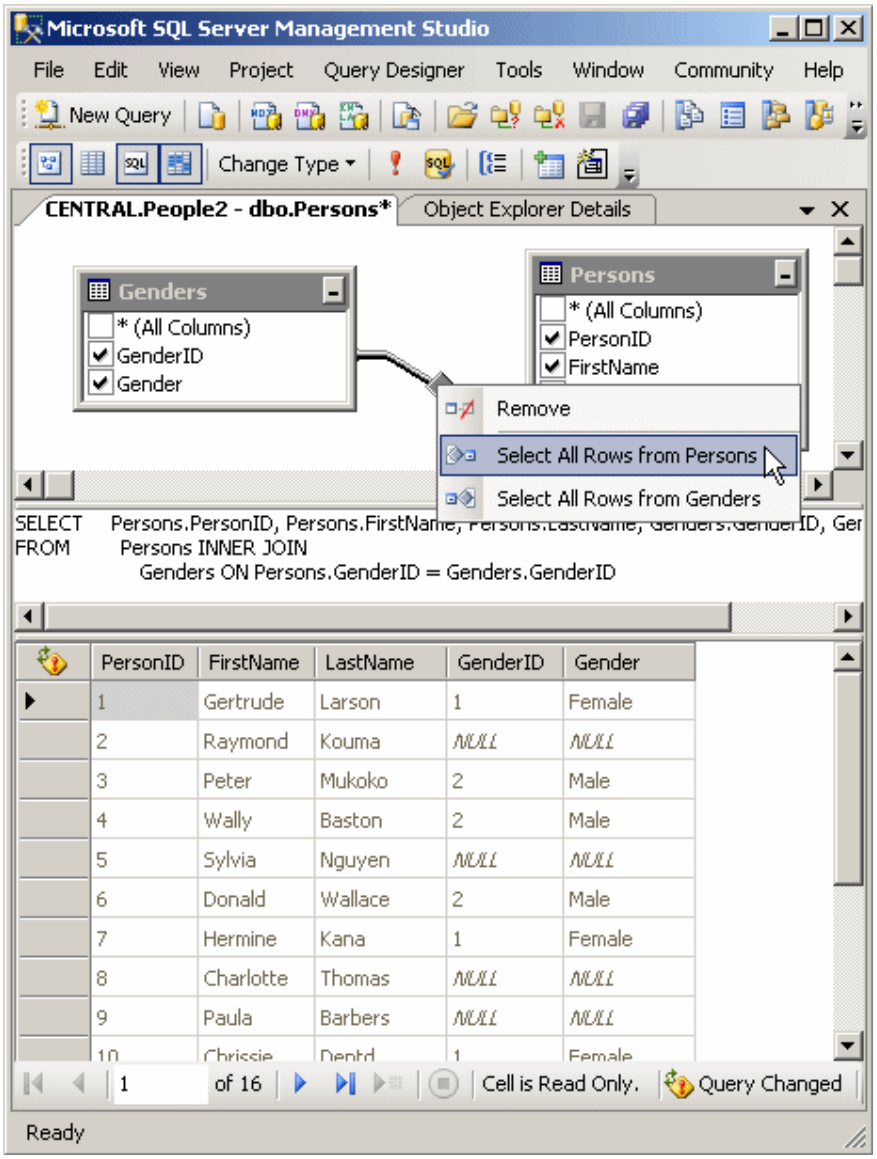
En lugar de mostrar sólo los registros que tienen las entradas en la tabla hija, es posible que desee su incluir todos los registros, incluidas las que son nulas. Para obtener este resultado, tendría que crear Tiene tres opciones.

Exterior se une a la izquierda

Un LEFT OUTER JOIN produce todos los registros de la tabla hija, también llamada el derecho de mesa. la tabla hija que no tienen una entrada en la columna clave foránea están marcados como *NULL.*

Para crear un LEFT OUTER JOIN, si está trabajando en la ventana de Tabla, en el diagrama de la s

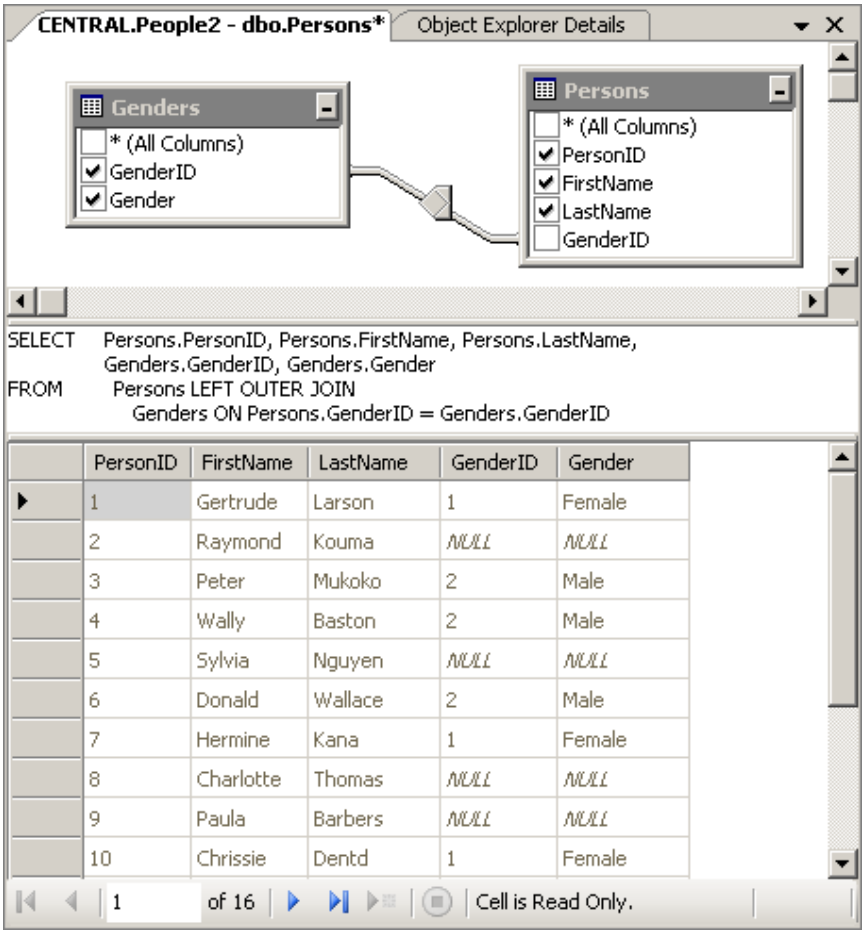
derecho en la línea que une las tablas y haga clic en la opción que seleccione todos los registros de la t
caso, que se Seleccionar todas las filas de las personas):



Si lo prefiere, puede sustituir el factor de *TypeOfJoin* nuestra fórmula con cualquiera de **los LEFT JOI**
JOIN. He aquí un ejemplo:

SELECCIONE Persons.PersonID, Persons.FirstName, Persons.LastName,
Genders.GenderID, Genders.Gender
DE LAS PERSONAS
LEFT OUTER JOIN géneros
SOBRE Persons.GenderID = Genders.GenderID
IR

En ambos casos, el botón en el centro de la línea, se añade una flecha que apunta a la tabla padre.
puede ejecutar la consulta para ver el resultado. Aquí está un ejemplo:



Observe que el resultado incluye todos los registros de las personas (también llamado de la derecha) registros que no tienen una entrada en la columna de la GenderID personas (el derecho) están marcad *NULL.*

Exterior se une a la derecha

El derecho considera exterior unirse todos los registros de la tabla padre y encuentra un registro coinci hija. Para ello, se comienza con el primer registro de la tabla padre (en este caso, los géneros de la cada uno de los registros de la tabla hija (en este caso las personas tabla), que tiene una entrada corre significa que, en nuestro ejemplo, un derecho exterior unirse en primer lugar crear una lista de los personas que tienen un 1 (Mujeres) GenderID valor de la columna. Tras el primer registro, el derecho s sumarse a la segunda acta, y así sucesivamente, cada vez que la lista de las actas de la mesa de los una entrada correspondiente de la clave primaria de la tabla padre.

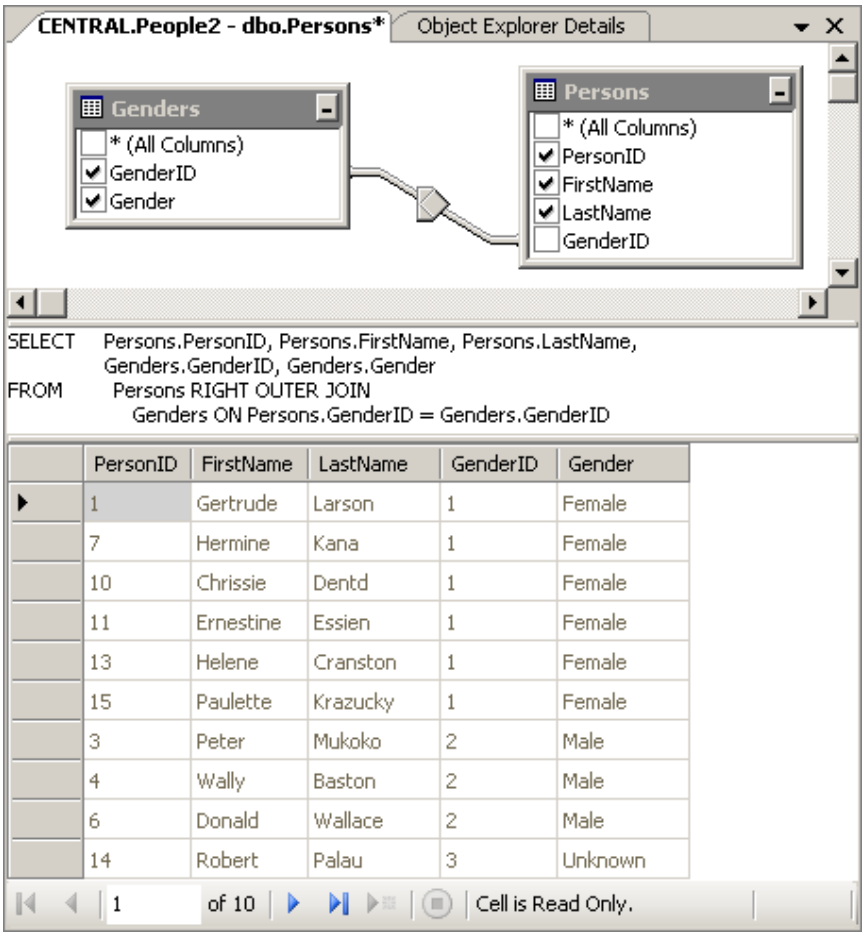
Para crear visualmente un derecho exterior se unan a la ventana de la tabla, tras el establecimiento de ambas tablas, si había creado un LEFT OUTER JOIN, debe eliminarlo haciendo clic derecho en la línea en seleccionar la segunda opción en virtud de quitar . A continuación, puede hacer clic derecho en la línea haga clic en la opción que seleccione todos los registros de la tabla padre. En nuestro ejemplo, Seleccionar todas las filas de los géneros.

Para crear un derecho OUTER JOIN en SQL, se puede sustituir el factor de *TypeOfJoin* nuestra fórmula **OUTER JOIN** o **DERECHA.** He aquí un ejemplo:

```
SELECCIONE Persons.PersonID, Persons.FirstName, Persons.LastName,
                Genders.GenderID, Genders.Gender
DE LAS PERSONAS
```

DERECHO OUTER JOIN géneros
SOBRE Persons.GenderID = Genders.GenderID
IR

En ambos casos, el botón de la unión entre las tablas tendría una flecha que apunta a la tabla hija. puede ejecutar la consulta. Aquí está un ejemplo:

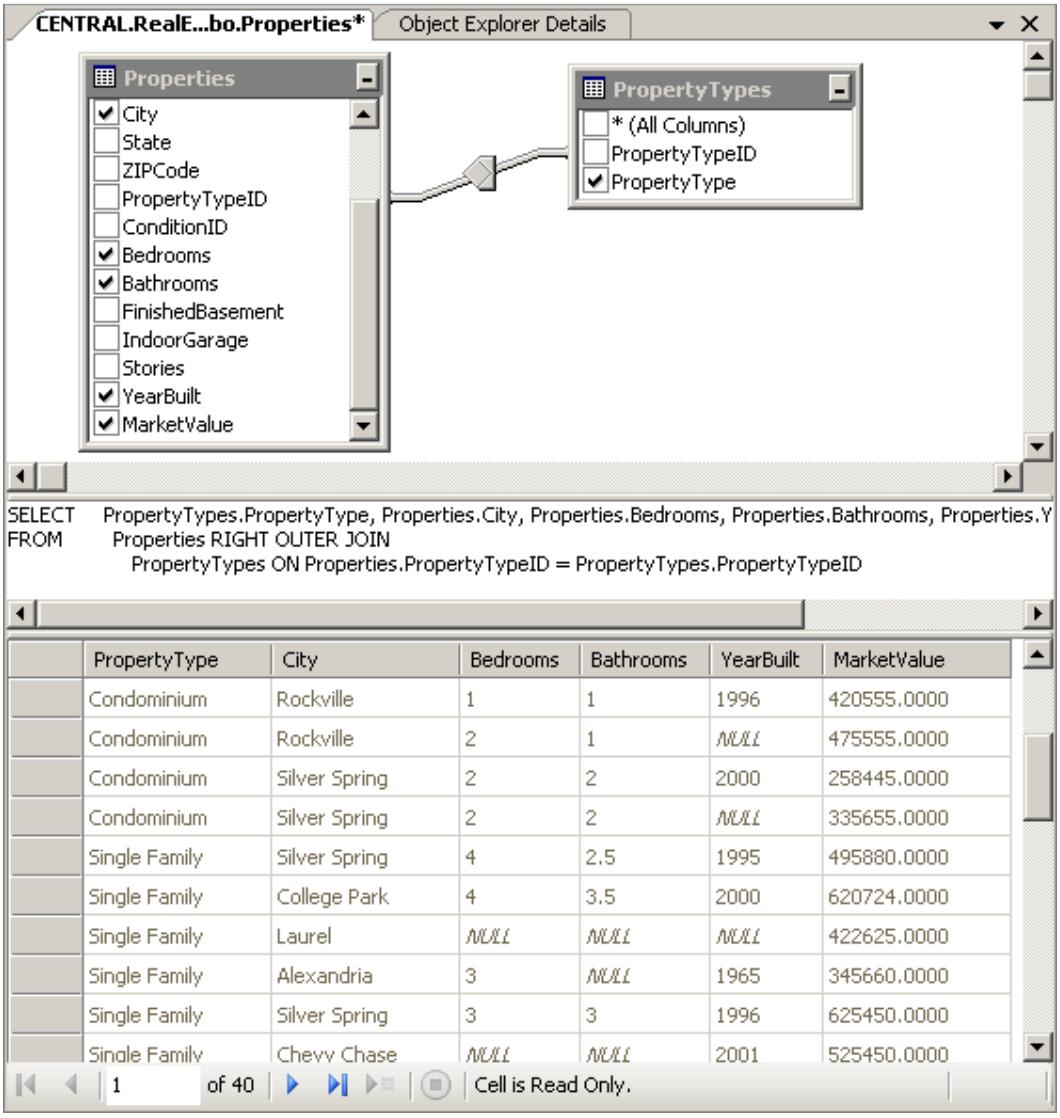


Observe que el resultado de consulta se inicia con el primer registro de la tabla padre, también lla izquierda (en este caso, los géneros de la tabla), y listas de los registros de la tabla hija, también lla derecha (en este caso las personas tabla), que tienen la entrada correspondiente a ese primer disco. A mueve a la siguiente GenderID valor. También, observe que no hay registros *NULL* en el género.

Aprendizaje práctico: Cómo no NULL Documentos

1. Para obtener una lista de propiedades que sólo se conocen los tipos, haga clic derecho en la línea e haga clic en Seleccionar todas las filas de PropertyTypes

2. Diseñador de consultas sobre la barra de herramientas, haga clic en el botón Ejecutar



3. Observe que el resultado es la lista de cuadros con el fin de los tipos (condominios, solo las familia la ciudad)

Exterior se une a la plena

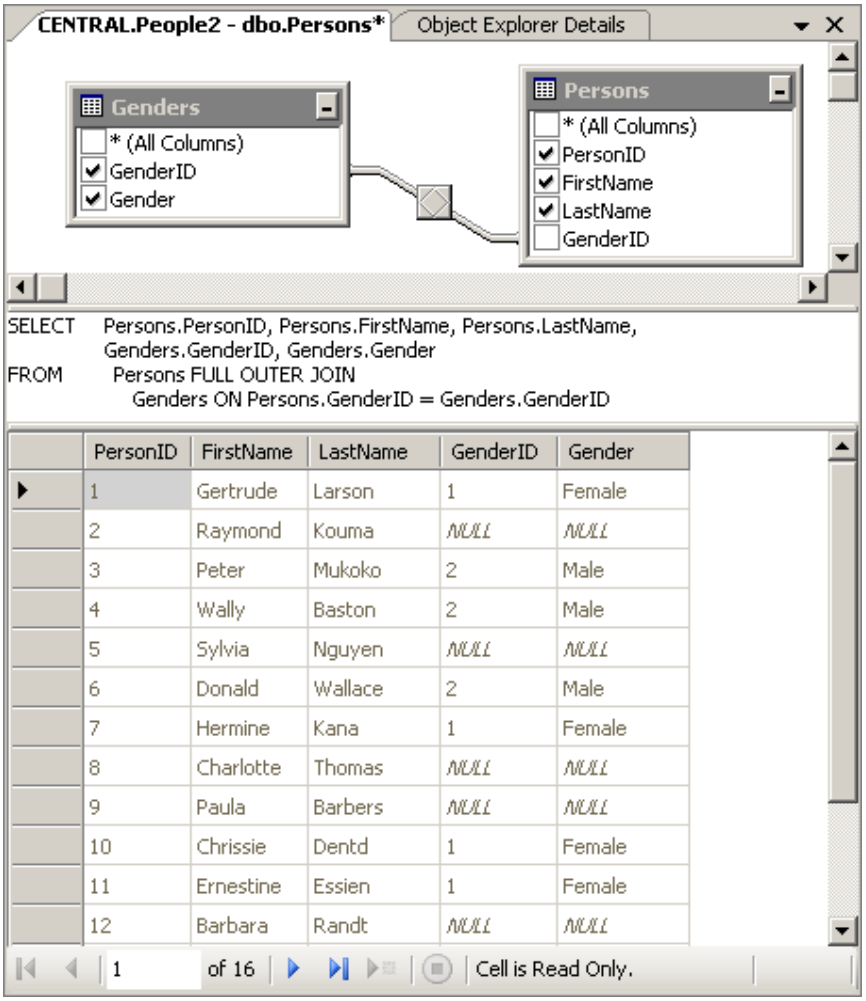
Un outer join produce todos los registros de los padres y el niño tablas. Si un registro de una tabla no el otro valor, el valor de registro que está marcada como *NULL.*

Para crear visualmente un completo unirse exterior, en el cuadro ventana, haga clic derecho en l cuadros y seleccionar cada opción en virtud de quitar de manera que ambos se verificó. Para crear un JOIN en SQL, *TypeOfJoin* reemplazar el factor de nuestra fórmula con **COMPLETA INSCRIBETE** o **CO JOIN.** He aquí un ejemplo:

```
SELECCIONE Persons.PersonID, Persons.FirstName, Persons.LastName,
                Genders.GenderID, Genders.Gender
DE LAS PERSONAS
COMPLETA OUTER JOIN géneros
SOBRE Persons.GenderID = Genders.GenderID
IR
```

El botón en la línea entre los cuadros que ahora aparecen como un cuadrado. A continuación, p

consulta. Aquí está un ejemplo:



Así como hemos participado sólo en dos cuadros se une a nuestros hasta ahora, puede crear una u varias tablas.

Las uniones y Análisis de Datos

Introducción

Como se ha demostrado hasta el momento y en las lecciones anteriores, la razón principal para la preguntas es aislar registros. Esto se hace con las condiciones y criterios. Se une a mejorar esta cap permiten examinar los registros de diferentes tablas e incluirlos en una declaración común de SQL.

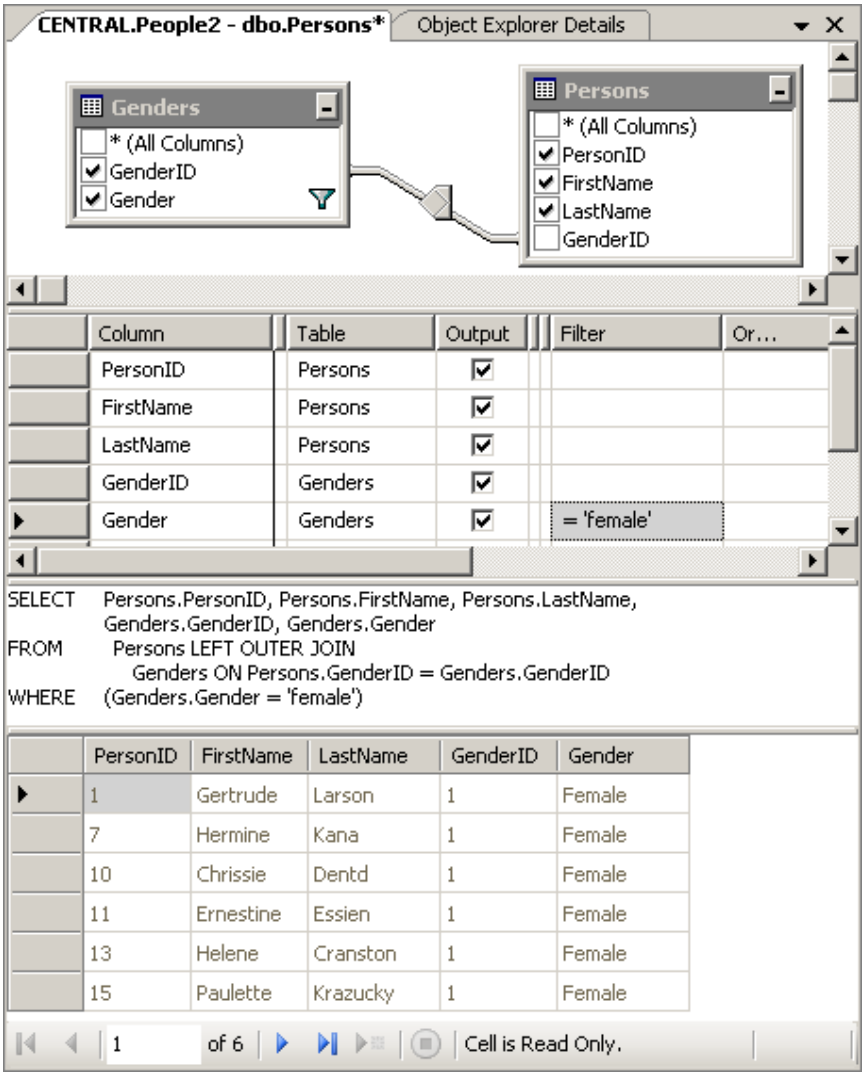
En la suma a que hemos creado hasta la fecha, hemos considerado todos los registros y permitir que de datos lista de ellos utilizando únicamente las reglas de la suma incorporada en el SQL. Para hacer es restrictivas, puede plantear sus propias condiciones que deben respetarse para aislar a los registros c Como se hizo en la experiencia anterior, para incluir un criterio en una **sentencia SELECT,** puede cre WHERE.

El uso de criterios

Para crear un criterio en una consulta se crea a partir de la de SQL Server Management Studio, prime columna para mostrar en la cuadrícula sección. Al igual que revisarse a la experiencia anterior cua consulta, para especificar un criterio, en el cuadro de criterios correspondientes a la columna, escriba la utilicen cualquiera de los operadores que hemos examinado en las lecciones anteriores. Aquí está un eje

SELECCIONE Persons.PersonID, Persons.FirstName, Persons.LastName,
Genders.GenderID, Genders.Gender
De las personas LEFT OUTER JOIN
SOBRE géneros Persons.GenderID = Genders.GenderID
DÓNDE Genders.Gender = "femeninos")

Esto produciría:



Aprendizaje práctico: Análisis de datos se une la participación de

1. Para ver una lista de sólo casas, cambiar la declaración en la sección de SQL como sigue:

   SELECCIONE PropertyTypes.PropertyType, Properties.City,
   Properties.State, Properties.ZIPCode, Properties.Bedrooms,
   Properties.Bathrooms, Properties.Stories, Properties.MarketValue
   Propiedades DE DERECHO OUTER JOIN PropertyTypes
   SOBRE Properties.PropertyTypeID = PropertyTypes.PropertyTypeID
   WHERE (PropertyTypes.PropertyTypeID = 3)

2. Diseñador de consultas sobre la barra de herramientas, haga clic en el botón Ejecutar SQL
3. Para obtener una lista única de casas y familias, cambiar la sentencia SQL de la siguiente manera:

```
SELECCIONE PropertyTypes.PropertyType, Properties.MarketValue,
      Properties.City, Properties.State, Properties.Bedrooms,
      Properties.YearBuilt, Properties.Bathrooms,
      Properties.FinishedBasement, Properties.Stories
Propiedades DE DERECHO OUTER JOIN PropertyTypes
SOBRE Properties.PropertyTypeID = PropertyTypes.PropertyTypeID
WHERE (PropertyTypes.PropertyTypeID EN (2, 3))
```

4. Diseñador de consultas sobre la barra de herramientas, haga clic en el botón Ejecutar SQL
5. Para obtener una lista única de las familias organizadas en orden cronológico empezando por el ma
cambiar la sentencia SQL de la siguiente manera:

```
SELECCIONE PropertyTypes.PropertyType,
      Properties.City, Properties.State, Properties.ZIPCode,
      Properties.Bedrooms, Properties.Bathrooms, Properties.Stories,
      Properties.YearBuilt, Properties.MarketValue
Propiedades DE DERECHO OUTER JOIN PropertyTypes
SOBRE Properties.PropertyTypeID = PropertyTypes.PropertyTypeID
WHERE (PropertyTypes.PropertyTypeID = 2)
ORDER BY DESC Properties.YearBuilt
```

6. Haga clic derecho en el cuadro y haga clic en Ejecutar SQL
7. Para obtener una lista de propiedades que cuestan entre $ 350.000 y $ 425.000, cambiar la senten
siguiente manera:

```
SELECCIONE Properties.PropertyNumber, PropertyTypes.PropertyType,
      Properties.MarketValue, Properties.City, Properties.State,
      Properties.Bedrooms, Properties.YearBuilt
Propiedades DE DERECHO OUTER JOIN PropertyTypes
SOBRE Properties.PropertyTypeID = PropertyTypes.PropertyTypeID
WHERE (Properties.MarketValue ENTRE 350000 Y 475000)
```

8. Diseñador de consultas sobre la barra de herramientas, haga clic en el botón Ejecutar SQL
9. Para obtener una lista de casas unifamiliares en Virginia dispuestas cronológicamente y sólo si el n
propiedad que se conoce, cambiar la sentencia SQL de la siguiente manera:

```
SELECCIONE Properties.PropertyNumber,
      PropertyTypes.PropertyType, Properties.MarketValue, Properties.City,
      Properties.State, Properties.Bedrooms,
      Properties.FinishedBasement, Properties.YearBuilt
Propiedades DE DERECHO OUTER JOIN PropertyTypes
SOBRE Properties.PropertyTypeID = PropertyTypes.PropertyTypeID
WHERE (PropertyTypes.PropertyTypeID = 2) Y
      (Properties.PropertyNumber IS NOT NULL) Y
      (Properties.State = 'VA')
ORDER BY DESC Properties.YearBuilt
```

10. Diseñador de consultas sobre la barra de herramientas, haga clic en el botón Ejecutar SQL
11. Para obtener una lista de propiedades en el sur de Maryland, pero que cuestan menos de 400.000
la sentencia SQL de la siguiente manera:

```
SELECCIONE Properties.PropertyNumber, PropertyTypes.PropertyType,
      Properties.MarketValue, Properties.City, Properties.State,
      Properties.Bedrooms, Properties.YearBuilt
Propiedades DE DERECHO OUTER JOIN PropertyTypes
SOBRE Properties.PropertyTypeID = PropertyTypes.PropertyTypeID
WHERE (Properties.MarketValue <400000) Y
      (Properties.ZIPCode ENTRE'20500 'Y'21000')
```

12. Diseñador de consultas sobre la barra de herramientas, haga clic en el botón Ejecutar SQL

Reproducciones

Listado de Vista

Introducción

Al estudiar el análisis de datos, una consulta es una técnica de aislamiento de una serie de columnas y / o registros de una tabla. Esto suele hacerse para los fines de análisis de datos. Esto también se puede hacer para crear una nueva lista de temas para cualquier razón. La mayoría de las veces, una consulta se crea con carácter temporal, como por ejemplo durante el análisis de los datos durante el uso de una tabla, un formulario, o una página web. Después de usar una lista temporal, es despedido. Muchas aplicaciones de bases de datos, incluyendo Microsoft SQL Server, le permiten crear una consulta y poder guardarlo para su uso posterior, o incluso para utilizarlo como si se tratara de su propia tabla. Esta es la idea detrás de un punto de vista.

Definición

Una vista es una lista de columnas o una serie de registros recuperados de una o más tablas existen combinación de uno o más puntos de vista y una o más tablas. Sobre esta base, antes de crear u primero debe decidir si sus columnas y los registros que vienen. Obviamente, el punto de vista es columnas uno y vienen de una tabla.

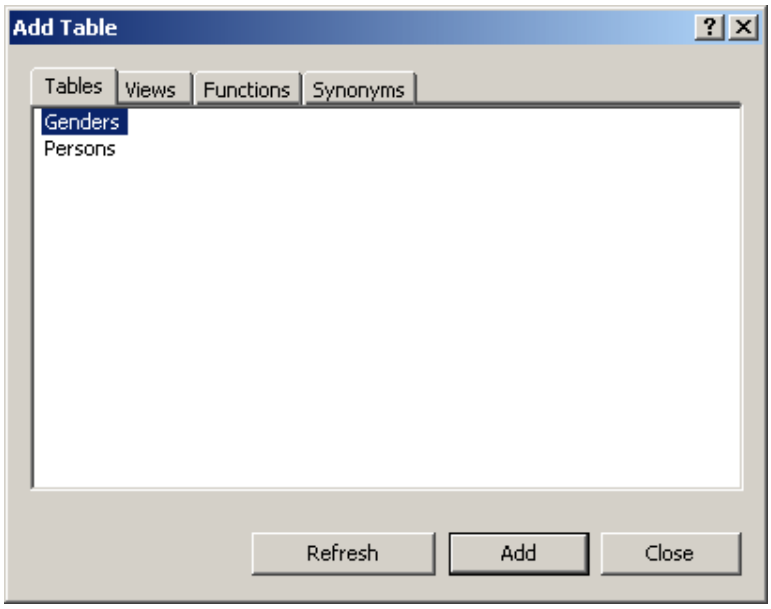
Aprendizaje práctico: La introducción de Vista

- Inicie Microsoft SQL Server y conectar con el servidor.
  Asegúrese de que tiene la **RealEstate2** base de datos creada en la lección anterior. Si no se crea

Fundamentos de la Creación de Vista

Creación de una vista visual

Para crear un punto de vista, puede usar el objeto Explorer o una ventana de consulta. Antes de c usted tendrá que especificar la tabla (s) de que podría participar en ellos. Para crear una vista desde objetos, y usted puede ampliar el base de datos, haga clic con el botón Vista y haga clic en Nueva Vist cuadro de diálogo Agregar tabla:



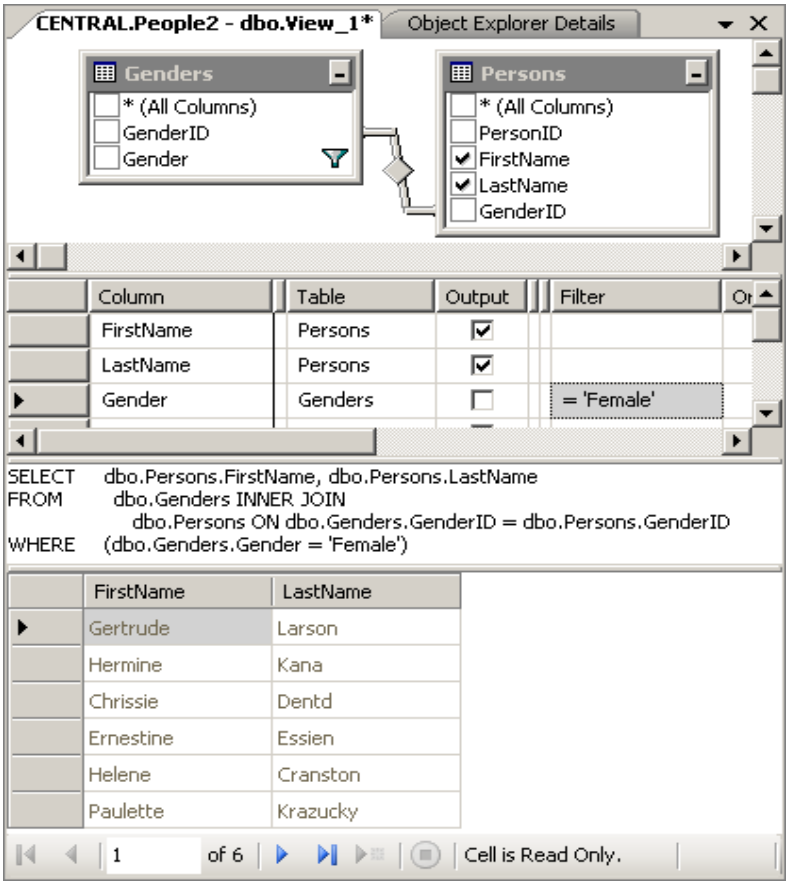
La funcionalidad básica es exactamente la misma que hemos examinado en la lección anterior:

- Para especificar la tabla que se utilizará como la fuente, puede hacer clic en el cuadro de la lista de

propiedad de la página

- Si se utiliza otro existente vista, a partir de la página Vista de propiedad, puede hacer clic en el no
- Si una función se utiliza para generar los registros, se puede localizar en la página de propiedades Después de seleccionar la fuente de objetos, puede haga doble clic en él o usted puede hacer clic u clic en Agregar. En la lección anterior, vimos que se puede añadir más de una tabla existente. Del puede añadir más de una opinión o funciones
- Después de seleccionar la fuente (s), usted puede hacer clic en Cerrar en el cuadro de diálogo Agre
- Después de seleccionar los objetos, como vimos en la lección anterior, que se muestran en la vent
- Como se ha visto en la lección anterior, si usted está usando más de una mesa y que no son (toda puede arrastrar una columna de una tabla y soltarlo en otra tabla para crear una unión entre ellas
- Como vimos en lecciones anteriores, para seleccionar una columna, puede hacer clic en su casilla d la lista superior. Esto lo mostramos en la primera caja vacía en la columna la columna y añadir su sentencia SELECT. Alternativamente, puede hacer clic en una caja vacía en la columna la columna cuadro combinado y, a continuación, haga clic en la flecha del cuadro combinado y seleccione la col
- Después de seleccionar la columna, su casilla de verificación se comprobará en la sección superior su nombre aparecerá en la columna de la columna, por lo que se añade a la **sentencia SELECT.** S de la columna que desea agregar, puede escribir manualmente en el **SELECT.**

Para la estructura de una vista puede ser considerado completo cuando **la** sentencia SELECT es lo más En cualquier momento, para poner a prueba los resultados de una vista, puede ejecutar la misma. Para clic en el botón Ejecutar SQL . Esto podría causar la sección inferior de la vista para mostrar los consulta. Aquí está un ejemplo:



Como se dijo ya, una de las razones para crear un punto de vista es que se puede utilizar una y otra v vista debe ser guardado. Como la mayoría de los objetos en SQL Server, un punto de vista debe tene guarda en su propio archivo. Para guardar una vista desde la ventana de la vista, puede hacer clic en en la barra de herramientas. También puede tratar de cerrar la ventana. A continuación se le pedira guardar un punto de vista, usted debe darle un nombre que sigue las reglas y sugerencias de SQL.

Aprendizaje práctico: Creación de una visualmente Ver

1. En el Explorador de objetos, ampliar las bases de datos y los nodos RealEstate2
2. Haga clic con el botón Vista y haga clic en Nuevo Ver
3. En el cuadro de diálogo Agregar tabla, haga doble clic en PropertyTypes, Propiedades y Condicione
4. Haga clic en Cerrar
5. PropertyTypes de la mesa, PropertyTypeID arrastrar y soltar en el campo de la PropertyTypeID Pro tabla
6. Condiciones de la tabla, arrastre y suelte ConditionID en el ámbito de la ConditionID Propiedades c
7. En las tablas, seleccione los campos siguientes: PropertyNumber, PropertyType, Estado, Ciudad, co Dormitorios, Baños, FinishedBasement, Historias, YearBuilt, y MarketValue
8. En la sección de Criterios, haga clic en PropertyType y pulse Tabulador 6 veces. En el campo del fil **de la familia**
9. Ficha de prensa y, en su campo O, el tipo **Townhouse**

El nombre de un Vea

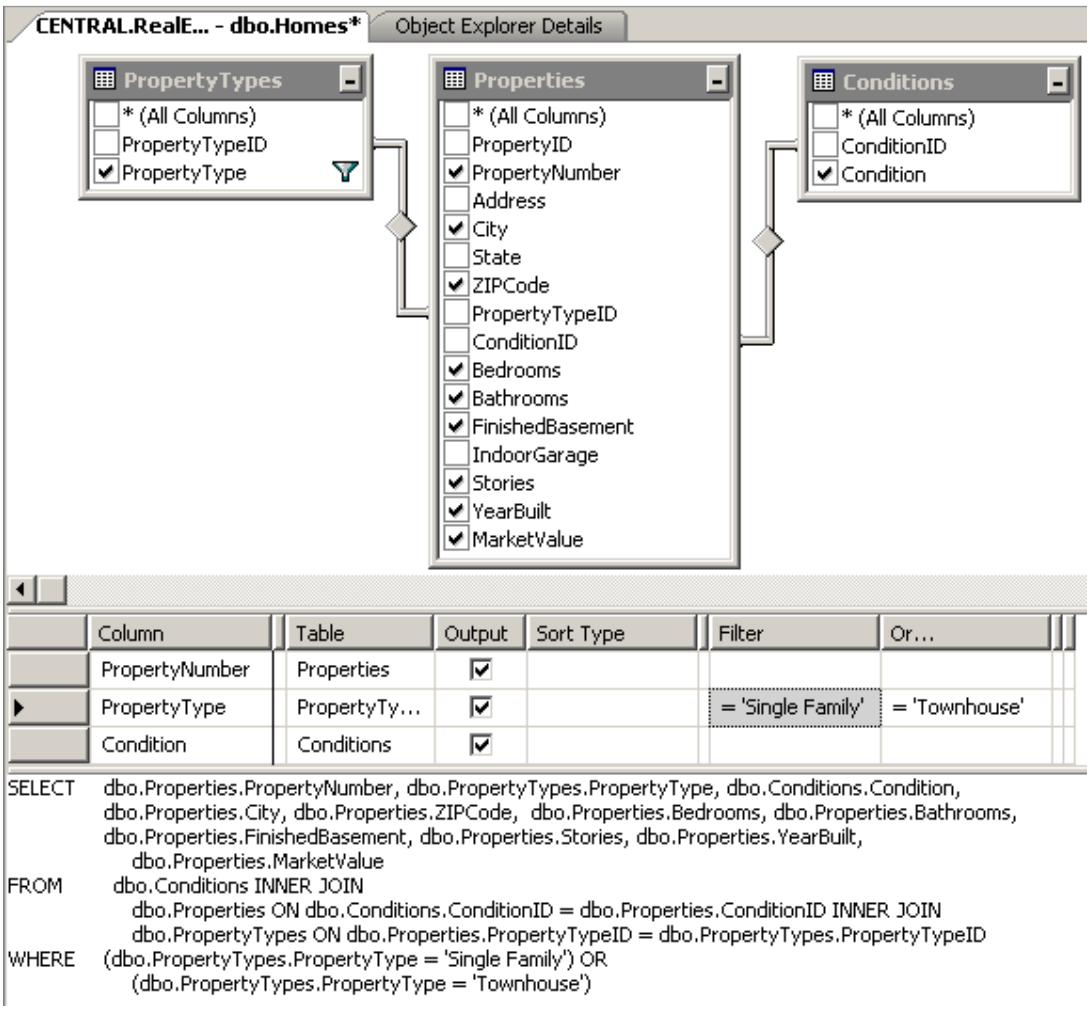
En nuestra experiencia, aquí están las reglas que se utilizan para nombrar nuestros puntos de vista:

- Un nombre se iniciará con una carta. Ejemplos de ello son n, actuar, o segundo
- Después de la primera carta, el nombre tendrá combinaciones de relieve, letras y dígitos. Ejemplo n24, act_52_t
- Un nombre no incluir caracteres especiales como!, @, #, $,%, ^, &, O *
- Un nombre no tiene espacios
- Si el nombre es una combinación de palabras, cada palabra comenzará en mayúsculas. Ejemplos d DateHired, RealSport, o DriversLicenseNumber

Después de guardar un punto de vista, se convierte en parte del dictamen de su nodo de base de datos para él y su nombre aparece en el dictamen de su nodo de base de datos.

Aprendizaje práctico: Cómo guardar un Vea

1. Para guardar el punto de vista, en la barra de herramientas Estándar, haga clic en el botón Guarda
2. Establecer el nombre de **Casas** y haga clic en Aceptar



3. Cierre la ventana de la vista
4. En el Explorador de objetos, en virtud de RealEstate2, haga clic con el botón Vista y haga clic en N
5. En el cuadro de diálogo Agregar tabla, haga doble clic en PropertyTypes, Propiedades y Condicione
6. Haga clic en Cerrar
7. PropertyTypes de la mesa, PropertyTypeID arrastrar y soltar en el campo de la PropertyTypeID Pro tabla
8. Condiciones de la tabla, arrastre y suelte ConditionID en el ámbito de la ConditionID Propiedades c
9. En las tablas, seleccione los campos siguientes: PropertyNumber, PropertyType, Estado, Ciudad, co dormitorios, baños, y MarketValue
10. En la sección de Criterios, haga clic en PropertyType. Pulse Tabulador 3 veces y presione la barra quitar la marca de su salida de campo
11. Pulse Tabulador 3 veces. En su campo Filtro, escriba **Condominios** y pulse Enter
12. En el menú principal, haga clic en Archivo -> Guardar
13. Establecer el nombre de **Condominios** y haga clic en Aceptar
14. Cierre la ventana de la vista

Con Transact-SQL

Para crear una vista mediante programación, utilice la siguiente sintaxis SQL:

CREATE VIEW *ViewName*
COMO
SELECT

Microsoft SQL Server puede generar código del esqueleto de una vista para usted. Para usarlo, primer ventana de consulta vacía. Mostrar el Explorador de plantillas. En el Explorador de plantillas, expa Desde el nodo Ver, arrastrar y soltar Crear Ver en la ventana de consulta.

La creación de una vista comienza con la expresión **de CREATE VIEW** seguido de un nombre. E opinión de la siguiente manera **las normas y sugerencias** que hemos examinado anteriormente. D nombre de la vista, el uso de palabras clave **como** para indicar que está listo para definir el punto de vis

Porque el fin es principalmente una sentencia SQL, que se define utilizando una **sentencia SELEC** mismas reglas que estudiamos para el análisis de datos. Aquí hay un ejemplo de un punto de vista:

```
CREATE VIEW dbo.ListOfMen
COMO
SELECCIONE dbo.Genders.Gender,
          dbo.Persons.FirstName, dbo.Persons.LastName
DE dbo.Genders INNER JOIN dbo.Persons
SOBRE dbo.Genders.GenderID = dbo.Persons.GenderID
WHERE (dbo.Genders.Gender = 'Masculino');
IR
```

Después de crear la instrucción SQL que define el punto de vista, debe ejecutar la sentencia. Si se usa consulta, puede hacerlo pulsando F5. Una vez que se ejecuta, su nombre se añade automáticamente a de su base de datos incluso si no guardar su código.

Aprendizaje práctico: Creación de una vista

1. Asegúrese de que tiene el **Banco Nacional de Yugoslavia** en la base de datos creada Lesson13. crea, lo hacen ahora.
   En el Explorador de objetos, haga clic derecho en YugoNationalBank y haga clic en Nueva consulta
2. Para crear una nueva vista, escriba el código siguiente

```
YugoNationalBank uso;
IR
CREATE VIEW PayrollPreparation
COMO
SELECCIONE dbo.Employees.EmployeeNumber,
          dbo.Employees.LastName + ',' +
          COMO dbo.Employees.FirstName [Nombre Completo],
          dbo.Employees.HourlySalary,
           dbo.TimeSheets.TimeSheetCode, dbo.TimeSheets.Week1Monday,
           dbo.TimeSheets.Week1Tuesday, dbo.TimeSheets.Week1Wednesday,
           dbo.TimeSheets.Week1Thursday, dbo.TimeSheets.Week1Friday,
           dbo.TimeSheets.Week1Saturday, dbo.TimeSheets.Week1Sunday,
           dbo.TimeSheets.Week2Monday, dbo.TimeSheets.Week2Tuesday,
           dbo.TimeSheets.Week2Wednesday, dbo.TimeSheets.Week2Thursday,
           dbo.TimeSheets.Week2Friday, dbo.TimeSheets.Week2Saturday,
dbo.TimeSheets.Week2Sunday

DE dbo.Employees INNER JOIN dbo.TimeSheets
SOBRE dbo.Employees.EmployeeNumber = dbo.TimeSheets.EmplNumber;
IR
```

3. Para ejecutar el código, presione F5
4. Borrar el contenido de toda la ventana de la vista

La apertura o ejecución de un Vea

La apertura de un Vea

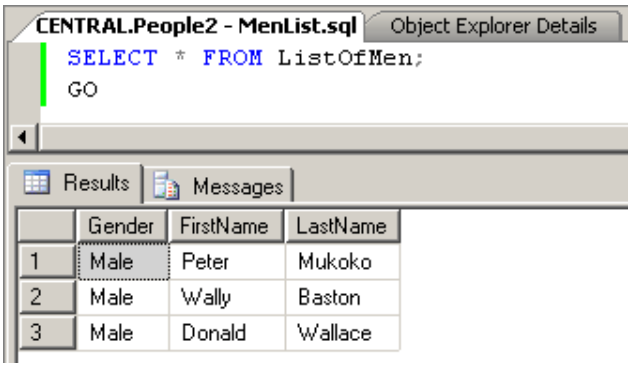
Como se dijo ya, un punto de vista es una técnica de selección de los registros para ver o usar r
Después de un fin se ha creado, puede abrirlo. Usted tiene dos opciones principales.

- Para ver la estructura de un punto de vista, como la tabla (s) en que se basa y las relaciones que, lo componen, en el Explorador de objetos, haga clic derecho en la vista Diseño y haga clic en
- Para ver el código SQL que constituye un punto de vista, en el Explorador de objetos, haga clic der y haga clic en Modificar

Ver la ejecución de una

Ejecución de una vista consta de ver sus resultados. Para ello, usted tiene varias opciones. Para ver un punto de vista:

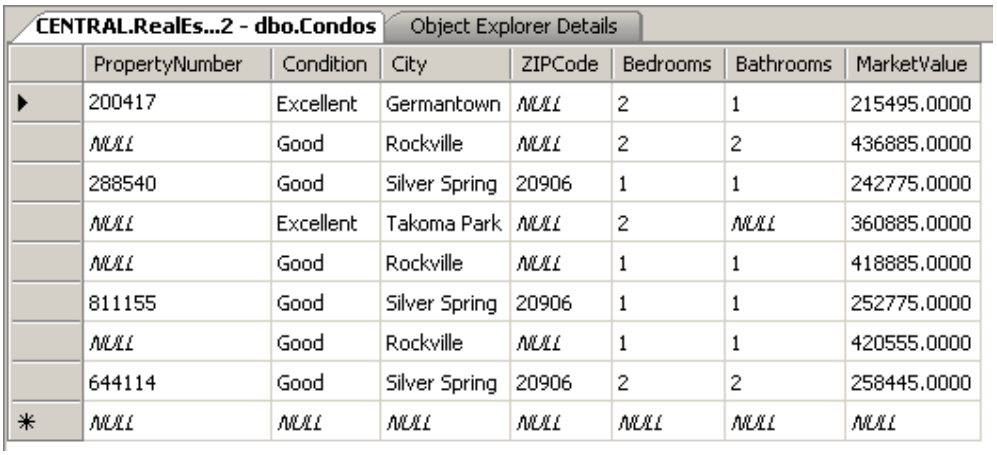
- Abra una ventana de consulta vacía asociados con la base de datos que contiene el punto de vista. de consulta, escriba **un** comando SELECT usando las mismas fórmulas y las reglas que vimos a las está un ejemplo:



- Desde el Explorador de objetos, expanda la base de datos y sus Reproducciones nodo. Haga clic d nombre de la vista y haga clic en Abrir Ver

Aprendizaje práctico: La ejecución de un Vea

1. En el Explorador de objetos, en virtud de RealEste2, si es necesario ampliar Vista. Condominios Ha botón Ver y haga clic en Abrir



PropertyNumber	Condition	City	ZIPCode	Bedrooms	Bathrooms	MarketValue
200417	Excellent	Germantown	NULL	2	1	215495.0000
NULL	Good	Rockville	NULL	2	2	436885.0000
288540	Good	Silver Spring	20906	1	1	242775.0000
NULL	Excellent	Takoma Park	NULL	2	NULL	360885.0000
NULL	Good	Rockville	NULL	1	1	418885.0000
811155	Good	Silver Spring	20906	1	1	252775.0000
NULL	Good	Rockville	NULL	1	1	420555.0000
644114	Good	Silver Spring	20906	2	2	258445.0000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

2. Cierre la ventana de la vista
3. En el Explorador de objetos, haga clic derecho en YugoNationalBank y haga clic en Nueva consulta
4. Para ejecutar el punto de vista, en la ventana de la vista vacía, escriba lo siguiente

```
USO YugoNationalBank;
IR
SELECCIONE PayrollPreparation .* FROM PayrollPreparation;
IR
```

5. Para ejecutar el punto de vista, presione F5
6. Borrar el contenido de la ventana

Ver Mantenimiento

Las propiedades de un Vea

En Transact-SQL, un punto de vista se considera un objeto. Como tal, se puede ver, modificado o su que cualquier otro objeto, un punto de vista tiene sus propias características. Para verlos, usted derecho en la vista y haga clic en Propiedades. Ver un cuadro de diálogo Propiedades que surgen. información como el nombre de la base de datos pertenece a la vista, la fecha de la vista se ha creado, e

Aprendizaje práctico: Acceso a las propiedades de un Vea

1. Para crear otro punto de vista, escriba lo siguiente:

```
- ==============================================
- Base de datos: YugoNationalBank
- Vista: de tiempo
- ==============================================
USO YugoNationalBank
IR

SI object_id (N'TimeSheet ',' V ') IS NOT NULL
          DROP VIEW dbo.TimeSheet
IR

CREATE VIEW dbo.TimeSheet
COMO
SELECCIONE EmplNumber, StartDate,
        Week1Monday, Week1Tuesday, Week1Wednesday,
        Week1Thursday, Week1Friday,
        Week1Saturday, Week1Sunday,
        Week2Monday, Week2Tuesday, Week2Wednesday,
        Week2Thursday, Week2Friday, Week2Saturday,
        Week2Sunday, Notas
De tiempo;
```

2. Presione la tecla F5 para ejecutar la declaración
3. Borrar el contenido de la ventana
4. En el Explorador de objetos, en virtud de YugoNationalBank, haga clic con el botón Vista y haga cli
5. Reproducciones ampliar si es necesario. Haga clic con el botón de tiempo y haga clic en Propiedad
6. Presione Esc para cerrar el cuadro de diálogo Propiedades

Modificación de una Vista

Después de un fin se ha creado, ya sea por usted o alguien más, usted puede descubrir que tie innecesario, es necesario una columna desaparecidos, que incluye registros innecesarios, o algun desaparecido. Afortunadamente, usted puede cambiar la estructura o el código de una opinión. Esto

alteración de una vista. Usted tiene dos opciones principales:

- Para cambiar una vista visual, en el Explorador de objetos, haga clic derecho en la vista y haga clic Desde la ventana de la vista, usted puede agregar o quitar las columnas. También puede cambiar la una de las secciones de la ventana. Después de modificar el punto de vista, guardarlo y cerrarlo
- Para cambiar el código de una vista, en el Explorador de objetos, haga clic con el botón Editar y ve editar la vista del código, puede guardarlo
- Desde el Explorador de objetos, haga clic derecho en la vista, coloca el ratón sobre comandos de A ALTER -> Nueva ventana del editor de Consulta

La fórmula básica de programación modificar un punto de vista es:

ALTER VIEW *ViewName*
COMO
SELECT

Usted comienza con la alteración de la expresión **de ALTER VIEW** seguido por el nombre de la vista. D nombre de la vista, **como** el uso de palabras clave para especificar que está listo para mostrar el cambio AS palabra clave, puede definir la opinión según sus necesidades. Por ejemplo, puede crear una **inst** que incluye una modificación del código existente o una nueva declaración.

En la opinión de que hemos creado para mostrar una lista de los hombres de una tabla, que incluye una género. Esta columna es inútil o redundante porque ya sabemos que la lista incluye sólo los hombr ejemplo de la alteración de la vista para eliminar (o más bien de omitir) la columna de género de las per

```
ALTER VIEW dbo.ListOfMen
COMO
SELECCIONE dbo.Persons.FirstName, dbo.Persons.LastName
DE dbo.Genders INNER JOIN dbo.Persons
SOBRE dbo.Genders.GenderID = dbo.Persons.GenderID
WHERE (dbo.Genders.Gender = 'Masculino');
```

Eliminación de una vista

En lugar de modificar un punto de vista, si la encuentra del todo inútil, puede eliminarlo de su base tiene varias opciones. Para eliminar un punto de vista:

- En el Explorador de objetos, en una base de datos, haga clic derecho en el nombre de la vista y ha Eliminar. Usted se daría la oportunidad de confirmar su intención o de cambiar de opinión
- En el Explorador de objetos, haga clic derecho en la vista, coloca el ratón sobre comandos de Al Ve ventana del editor de Nueva consulta
- Abra una ventana de consulta vacía asociada con la base de datos que tiene el mal vista. Desde el plantillas, en el nodo Ver, Ver soltar arrastrar y soltar en la ventana de consulta

La fórmula de programación eliminar un punto de vista es:

DROP VIEW *ViewName*

En la parte derecha de la expresión **de DROP VIEW,** escriba el nombre de la mal vista y la ejecución No se le advirtió antes de que el intérprete elimina el punto de vista.

El uso de un Vea

Con una introducción de datos Ver

Como se ha visto hasta ahora, un punto de vista es una lista seleccionada de registros de una tabla. C sospechar que, la mejor vista es probablemente uno creado a partir de una tabla. Imagina que tie empleados y que desea crear una vista que muestra sólo sus nombres. Puede crear una vista de la sigu

CREATE VIEW dbo.EmployeesNames

```
COMO
SELECCIONE FirstName,
          Apellido,
          Apellidos + ',' + Nombre FullName como a personas;
IR
```

En ese punto de vista que se basa en una tabla, puede realizar la entrada de datos, utilizando el punto de la tabla. Para ello, siga las mismas normas que hemos examinado en la Lección 9. Aquí está un ejer

```
INSERT INTO dbo.EmployeesNames (Nombre, Apellido)
VALUES ( 'Pedro', 'justicia');
```

Si realiza la entrada de datos mediante un punto de vista, los datos que nos proporcione será incor base, lo que significa que la tabla se actualiza automáticamente. Sobre la base de esta función, puede propósito la intención de actualizar una tabla a fin de que, en el punto de vista, que incluyen sólo l deben ser actualizados.

Aprendizaje práctico: Realización de una entrada de datos Uso de Ver

1. Para crear una función vamos a utilizar, introduzca el siguiente código:

```
  --==================================================
==
  - Base de datos: YugoNationalBank
  - Función: CreateTimeSheetCode
  - Objetivo: Esta función toma un número de empleado
  - Y la fecha de inicio de una hoja de tiempo.
  - A continuación, se crea un número único
  - En el formato 0000000000000
  - Los 5 primeros dígitos representan el
  - Número de empleados,
  - El segundo 4 dígitos representan el año,
  - Los 2 dígitos representan el mes,
  - 2 últimas cifras que representan el día
  --==================================================
==
  USO YugoNationalBank
  IR

  SI object_id (N'dbo. CreateTimeSheetCode ') IS NOT NULL
      DROP FUNCTION dbo.CreateTimeSheetCode
  IR

  CREATE FUNCTION dbo.CreateTimeSheetCode (@ EmplNbr varchar (6),
                                           @ dteStart datetime)
  VUELVE varchar (15)
  COMO
  COMENZAR
        DECLARE @ strMonth COMO varchar (20);
        DECLARE @ strDay COMO varchar (20);
        DECLARE @ iMonth COMO int;
        DECLARE @ iDay COMO int;
        DECLARE @ strTimeSheetCode varchar (20);

        SET @ iMonth = CONVERT (varchar (20), MES (@ dteStart));
        SET @ iDay = CONVERT (varchar (20), DIA (@ dteStart));

        SI @ iMonth <10
```

```
                SET @ strMonth = CONVERT (varchar (20), AÑO (@ dteStart)) +
                                '0 '+ CONVERT (varchar (20), @ iMonth);
        ELSE
            SET @ strMonth = CONVERT (varchar (20), AÑO (@ dteStart)) +
                            CONVERT (varchar (20), @ iMonth);

        SI @ iDay <10
                SET @ strDay @ strMonth + ='0 '+
                                CONVERT (varchar (20), @ iDay);
        ELSE
            SET @ strDay = @ strMonth + CONVERT (varchar (2), @ iDay);

        SET @ strTimeSheetCode = @ @ EmplNbr + strDay;
        VOLVER strTimeSheetCode @;
    FIN
    IR
```

2. Presione la tecla F5 para ejecutar
3. Para realizar la entrada de datos mediante un punto de vista, entrar en el siguiente código:

```
USO YugoNationalBank
IR
- El siguiente código realiza la entrada de datos usando una vista
INSERT INTO dbo.TimeSheet
VALUES ('46288 ','1 / 1 / 2007 »,
        0,00, 8,50, 9,50, 8,50, 9,00, 0,00, 0,00,
        10.00, 9.50, 8.50, 10.50, 9.00, 0.00, 0.00,'');
IR
INSERT INTO dbo.TimeSheet
VALUES ('66286 ','1 / 1 / 2007 »,
        0,00, 8,50, 6,50, 5,50, 6,50, 0,00, 0,00,
        4,00, 6,00, 6,50, 6,00, 5,50, 0,00, 0,00,'');
IR
INSERT INTO dbo.TimeSheet
VALUES ('92493 ','1 / 1 / 2007 »,
        0,00, 8,00, 9,00, 8,50, 9,50, 0,00, 0,00,
        5.50, 6.50, 4.50, 6.00, 4.00, 0.00, 0.00,'');
IR
INSERT INTO dbo.TimeSheet
VALUES ('27199 ', 1 / 15/2007',
        6.00, 8.50, 0.00, 4.00, 6.50, 0.00, 0.00,
        4,00, 0,00, 6,00, 4,00, 0,00, 0,00, 0,00,'');
IR
INSERT INTO dbo.TimeSheet
VALUES ('39538 ', 1 / 15/2007',
        8,00, 8,00, 6,00, 8,50, 6,00, 0,00, 0,00,
        9.50, 10.50, 8.00, 8.00, 8.50, 0.00, 0.00,
        "Hubo algunos que faltan veces en el tiempo hoja.   '+
        'Ellos se han registrado.');
IR
INSERT INTO dbo.TimeSheet
VALUES ('40550 ', 1 / 15/2007',
        8,50, 8,00, 0,00, 8,50, 0,00, 0,00, 0,00,
        6.00, 6.50, 6.50, 0.00, 4.00, 0.00, 0.00,'');
IR
INSERT INTO dbo.TimeSheet
VALUES ('66286 ', 1 / 29/2007',
```

```
            8.00, 6.50, 9.50, 8.00, 7.50, 0.00, 0.00,
           10.50, 9.50, 8.50, 8.00, 10.00, 0.00, 0.00,'');
IR
INSERT INTO dbo.TimeSheet
VALUES ('90026 ','2 / 12/2007',
           8.50, 6.50, 8.00, 8.00, 9.50, 0.00, 0.00,
           9.50, 8.00, 8.50, 8.00, 8.00, 0.00, 0.00,'');
IR
INSERT INTO dbo.TimeSheet
VALUES ('92493 ','2 / 12/2007',
           4.00, 6.50, 5.50, 8.00, 6.50, 0.00, 0.00,
           8,00, 8,00, 8,00, 6,00, 8,00, 0,00, 0,00,'');
IR
- El siguiente código actualiza una tabla usando una función
ACTUALIZACIÓN dbo.TimeSheets
AJUSTE TimeSheetCode = dbo.CreateTimeSheetCode ('46288 ','1 / 1 / 2007 »)
WHERE (EmplNumber ='46288 ') Y (StartDate ='1 / 1 / 2007');
IR
ACTUALIZACIÓN dbo.TimeSheets
AJUSTE TimeSheetCode = dbo.CreateTimeSheetCode ('66286 ','1 / 1 / 2007 »)
WHERE (EmplNumber ='66286 ') Y (StartDate ='1 / 1 / 2007');
IR
ACTUALIZACIÓN dbo.TimeSheets
AJUSTE TimeSheetCode = dbo.CreateTimeSheetCode ('92493 ','1 / 1 / 2007 »)
WHERE (EmplNumber ='92493 ') Y (StartDate ='1 / 1 / 2007');
IR
ACTUALIZACIÓN dbo.TimeSheets
AJUSTE TimeSheetCode = dbo.CreateTimeSheetCode ('27199 ', 1 / 15/2007')
WHERE (EmplNumber ='27199 ') Y (StartDate ='1 / 15/2007');
IR
ACTUALIZACIÓN dbo.TimeSheets
AJUSTE TimeSheetCode = dbo.CreateTimeSheetCode ('39538 ', 1 / 15/2007')
WHERE (EmplNumber ='39538 ') Y (StartDate ='1 / 15/2007');
IR
ACTUALIZACIÓN dbo.TimeSheets
AJUSTE TimeSheetCode = dbo.CreateTimeSheetCode ('40550 ', 1 / 15/2007')
WHERE (EmplNumber ='40550 ') Y (StartDate ='1 / 15/2007');
IR
ACTUALIZACIÓN dbo.TimeSheets
AJUSTE TimeSheetCode = dbo.CreateTimeSheetCode ('66286 ', 1 / 29/2007')
WHERE (EmplNumber ='66286 ') Y (StartDate ='1 / 29/2007');
IR
ACTUALIZACIÓN dbo.TimeSheets
AJUSTE TimeSheetCode = dbo.CreateTimeSheetCode ('90026 ','2 / 12/2007')
WHERE (EmplNumber ='90026 ') Y (StartDate ='2 / 12/2007');
IR
ACTUALIZACIÓN dbo.TimeSheets
AJUSTE TimeSheetCode = dbo.CreateTimeSheetCode ('92493 ','2 / 12/2007')
WHERE (EmplNumber ='92493 ') Y (StartDate ='2 / 12/2007');
IR
```

4. Presione la tecla F5 para ejecutar
5. Borrar el contenido de la ventana

Ver Con un alias

Es importante saber que un punto de vista es más de una tabla tipo que cualquier otro objeto. Esto

punto de vista no es una función, pero puede utilizar una función. La palabra argumento aquí sólo sigr valores pueden ser transmitidos a un punto de vista, pero estos valores sólo se puede especificar la vista. No son argumentos reales.

Al estructurar un punto de vista, puede crear marcadores de posición para las columnas y pasar en e vista. Esto se haría de la siguiente manera:

CREATE VIEW CarIdentifier ([Etiqueta #], Fabricante, [Tipo de coche], Disponible)
. . .

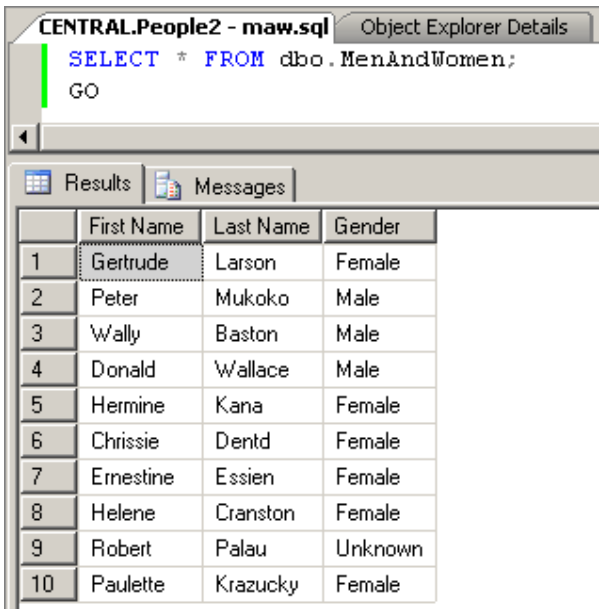
Si se utiliza esta técnica, los nombres en el pasado entre paréntesis son de la opinión de los títulos qu lugar de las columnas de la vista. Esta técnica le permite especificar las cuerdas de su elección para desea un encabezado de columna para mostrar el nombre real de la columna, escriba la misma. E puede utilizar cualquier cadena que desea para la columna. Si el nombre está en una palabra, puedes nombre incluye varios términos, entre ellos incluyen una apertura de corchete "[" y un cierre de corchete

Después de la lista de las cadenas necesarias en los títulos de las columnas, en su **sentencia SEL** deberá utilizar el mismo número de columnas como el número de argumentos de la vista. De hecho, ca **sentencia SELECT** debe corresponder a un argumento de la misma orden.

Aquí está un ejemplo:

```
CREATE VIEW dbo.MenAndWomen ([Nombre], [Apellido], de género)
COMO
SELECCIONE dbo.Persons.FirstName,
          dbo.Persons.LastName,
          dbo.Genders.Gender
DE dbo.Genders INNER JOIN dbo.Persons
SOBRE dbo.Genders.GenderID = dbo.Persons.GenderID;
IR
```

Porque, como ya hemos señalado, un punto de vista no es una función y los valores pasados a l argumentos no son reales, al ejecutar el punto de vista, no especifica los nombres de los argumentos. un **SELECT** y especificar el nombre de la vista como la fuente. Aquí está un ejemplo:



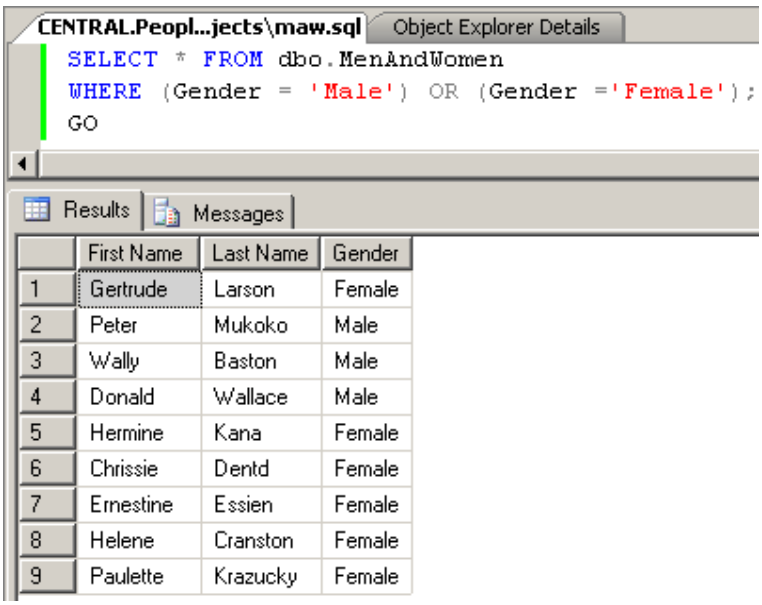
	First Name	Last Name	Gender
1	Gertrude	Larson	Female
2	Peter	Mukoko	Male
3	Wally	Baston	Male
4	Donald	Wallace	Male
5	Hermine	Kana	Female
6	Chrissie	Dentd	Female
7	Ernestine	Essien	Female
8	Helene	Cranston	Female
9	Robert	Palau	Unknown
10	Paulette	Krazucky	Female

Opiniones y declaraciones condicionales

Además de su consulta de las características que le permiten realizar análisis de datos, probablemente

más importante de una consulta es su capacidad de ser tan complejas como sea posible por condicionales. Esto hace que sea posible utilizar una vista en lugar de una tabla en las operaciones y las complicaría el código o la estructura de una tabla. Al crear un punto de vista, en su **sentencia SELEC** selecciones de la columna, para ellos, y establecer criterios para excluir algunos registros.

Aquí está un ejemplo:



	First Name	Last Name	Gender
1	Gertrude	Larson	Female
2	Peter	Mukoko	Male
3	Wally	Baston	Male
4	Donald	Wallace	Male
5	Hermine	Kana	Female
6	Chrissie	Dentd	Female
7	Ernestine	Essien	Female
8	Helene	Cranston	Female
9	Paulette	Krazucky	Female

Opiniones y Funciones

Para crear más complejas o avanzadas de puntos de vista, puede implicar funciones. Como siempre, ta utilizar las funciones son los incorporados.

Si no hay incorporado en la función que realiza la operación que desea, puede crear su propio archiv ejemplo:

```
USO personas;
IR
CREATE FUNCTION dbo.GetFullName
(
        @ FName varchar (20),
        @ Lname varchar (20)
),
VUELVE varchar (41)
COMO
COMENZAR
        VOLVER lname @ + ',' + @ FName;
FIN
IR
```

Una vez que haya una función que desea utilizar, usted puede llamar en el cuerpo de su punto consideran necesario. Aquí está un ejemplo:

```
CREATE VIEW dbo.MyPeople
COMO
SELECCIONE dbo.GetFullName (Nombre, Apellido) COMO [Nombre Completo],
        dbo.Genders.Gender
INNER JOIN DE géneros dbo.Persons
```

SOBRE dbo.Genders.GenderID = dbo.Persons.GenderID;

Esto produciría:

CENTRAL.Peopl... dbo.MyPeople | Object Explorer Details

Full Name	Gender
Larson, Gertrude	Female
Mukoko, Peter	Male
Baston, Wally	Male
Wallace, Donald	Male
Kana, Hermine	Female
Dentd, Chrissie	Female
Essien, Ernestine	Female
Cranston, Helene	Female
Palau, Robert	Unknown
Krazucky, Paulette	Female
NULL	NULL

Aprendizaje práctico: Uso de una función en Vista

1. En el Explorador de objetos, en virtud de YugoNationalBank, si es necesario ampliar Vista. Haga clic con el botón PayrollPreparation, coloca el ratón sobre comandos de Al Ver -> Para ALTER ventana del editor de Consulta
2. Cambiar el código de la siguiente manera:

```
USO [YugoNationalBank]
IR
AJUSTE DE ANSI_NULLS
IR
AJUSTE DE QUOTED_IDENTIFIER
IR
ALTER VIEW [dbo]. [PayrollPreparation]
COMO
SELECCIONE dbo.Employees.EmployeeNumber,
          dbo.Employees.LastName + ',' +
          COMO dbo.Employees.FirstName [Nombre Completo],
          dbo.Employees.HourlySalary,
          dbo.TimeSheets.TimeSheetCode, dbo.TimeSheets.Week1Monday,
          dbo.TimeSheets.Week1Tuesday, dbo.TimeSheets.Week1Wednesday,
          dbo.TimeSheets.Week1Thursday, dbo.TimeSheets.Week1Friday,
          dbo.TimeSheets.Week1Saturday, dbo.TimeSheets.Week1Sunday,
          dbo.TimeSheets.Week2Monday, dbo.TimeSheets.Week2Tuesday,
          dbo.TimeSheets.Week2Wednesday, dbo.TimeSheets.Week2Thursday,
          dbo.TimeSheets.Week2Friday, dbo.TimeSheets.Week2Saturday,
          dbo.TimeSheets.Week2Sunday,

          CAST (dbo.TimeSheets.Week1Monday COMO SmallMoney) +
          CAST (dbo.TimeSheets.Week1Tuesday COMO SmallMoney) +
          CAST (dbo.TimeSheets.Week1Wednesday COMO SmallMoney) +
          CAST (dbo.TimeSheets.Week1Thursday COMO SmallMoney) +
          CAST (dbo.TimeSheets.Week1Friday COMO SmallMoney) +
          CAST (dbo.TimeSheets.Week1Saturday COMO SmallMoney) +
          CAST (dbo.TimeSheets.Week1Sunday COMO SmallMoney)
```

```
            COMO [Total Week1],
            CAST (dbo.TimeSheets.Week2Monday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Tuesday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Wednesday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Thursday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Friday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Saturday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Sunday COMO SmallMoney)
            COMO [Total Week2],
            CAST (dbo.TimeSheets.Week1Monday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week1Tuesday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week1Wednesday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week1Thursday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week1Friday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week1Saturday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week1Sunday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Monday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Tuesday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Wednesday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Thursday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Friday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Saturday COMO SmallMoney) +
            CAST (dbo.TimeSheets.Week2Sunday COMO SmallMoney)
            COMO [Tiempo total de hojas]
    DE dbo.Employees INNER JOIN dbo.TimeSheets
    SOBRE dbo.Employees.EmployeeNumber = dbo.TimeSheets.EmplNumber;
```

3. Para actualizar la consulta, en el menú principal, haga clic en Consulta -> Ejecutar
4. Borrar el contenido de la ventana
5. Para ver el resultado de la consulta, escriba lo siguiente:

```
YugoNationalBank uso;
IR
USO YugoNationalBank;
IR
SELECCIONE PayrollPreparation .* FROM PayrollPreparation;
IR
```

6. Borrar el contenido de la ventana

CENTRAL.YugoN...LQuery15.sql* | Object Explorer Details

```sql
Use YugoNationalBank;
GO
USE YugoNationalBank;
GO
SELECT PayrollPreparation.* FROM PayrollPreparation;
GO
```

Results | Messages

	Empl...	Full Name	HourlySalary	TimeSheetCode	Week1Mond...	Week1Tuesd...
1	46288	Marconi, Catherine	22.55	4628820070101	0.00	8.50
2	27199	Ankoma, Leonie	14.88	2719920070115	6.00	8.50
3	66286	Zbrnitz, Lienev	15.75	6628620070101	0.00	8.50
4	66286	Zbrnitz, Lienev	15.75	6628620070129	8.00	6.50
5	39538	Waste, Plant	16.75	3953820070115	8.00	8.00
6	92493	Roberts, Kirsten	18.05	9249320070101	0.00	8.00
7	92493	Roberts, Kirsten	18.05	9249320070212	4.00	6.50
8	90026	Lamy, Roger	10.24	9002620070212	8.50	6.50
9	40550	Kirkland, Lorraine	12.86	4055020070115	8.50	8.00

7. Borrar el contenido de la ventana

Procedimientos almacenados

Fundamentos de procedimientos almacenados

Introducción

En **Lección 6,** hemos tenido una introducción a algunos tipos de acciones que podrían realizarse en una base de datos. Estas acciones fueron llamados funciones. SQL ofrece otro tipo de acción un procedimiento almacenado. Si se han desarrollado aplicaciones en otros lenguajes como Visual Basic o Pascal, probablemente familiarizados con la idea de un

procedimiento. Al igual que una función, un procedimiento almacenado se utiliza para realizar una acción sobre una base de datos.

Aprendizaje práctico: La introducción de procedimientos almacenados

1. Inicie Microsoft SQL Server Management Studio y accede a tu servidor
2. En el menú principal, haga clic en Archivo -> Nuevo -> Consulta de conexión con los actuales
3. Para crear una nueva base de datos, introduzca el siguiente código en la ventana

```
- ==============================================
- Base de datos: WattsALoan
- ==============================================
USO maestro
IR
- Excluir de la base de datos si ya existe
IF EXISTS (
        SELECT name
                DE sys.databases
                WHERE name = N'WattsALoan '
),
DROP DATABASE WattsALoan
IR

CREATE DATABASE WattsALoan
IR
- ==========================================
- Tabla: Empleados
- ==========================================
USO WattsALoan
IR

SI object_id ( 'dbo.Employees',' U ') IS NOT NULL
   DROP TABLE dbo.Employees
IR

CREATE TABLE dbo.Employees
(
      IdEmpleado int identidad (1,1) NOT NULL,
      EmployeeNumber NCHAR (10) NULL,
      Nombre varchar (20) NULL,
      Apellido varchar (10),
      FullName COMO ((Apellido + ',') + Nombre),
      Título varchar (100),
      HourlySalary dinero,
      Nombre de usuario varchar (20),
      Contraseña varchar (20),
      PK_Employees CONSTRAINT PRIMARY KEY (IdEmpleado)
),
IR
INSERT INTO dbo.Employees (EmployeeNumber, Nombre, Apellido, Título,
HourlySalary)
VALUES ('293747 ',' Jeanne ',' Tryler ',' Administrador de cuentas ", 22,24);
IR
INSERT INTO dbo.Employees (EmployeeNumber, Nombre, Apellido, Título,
HourlySalary)
```

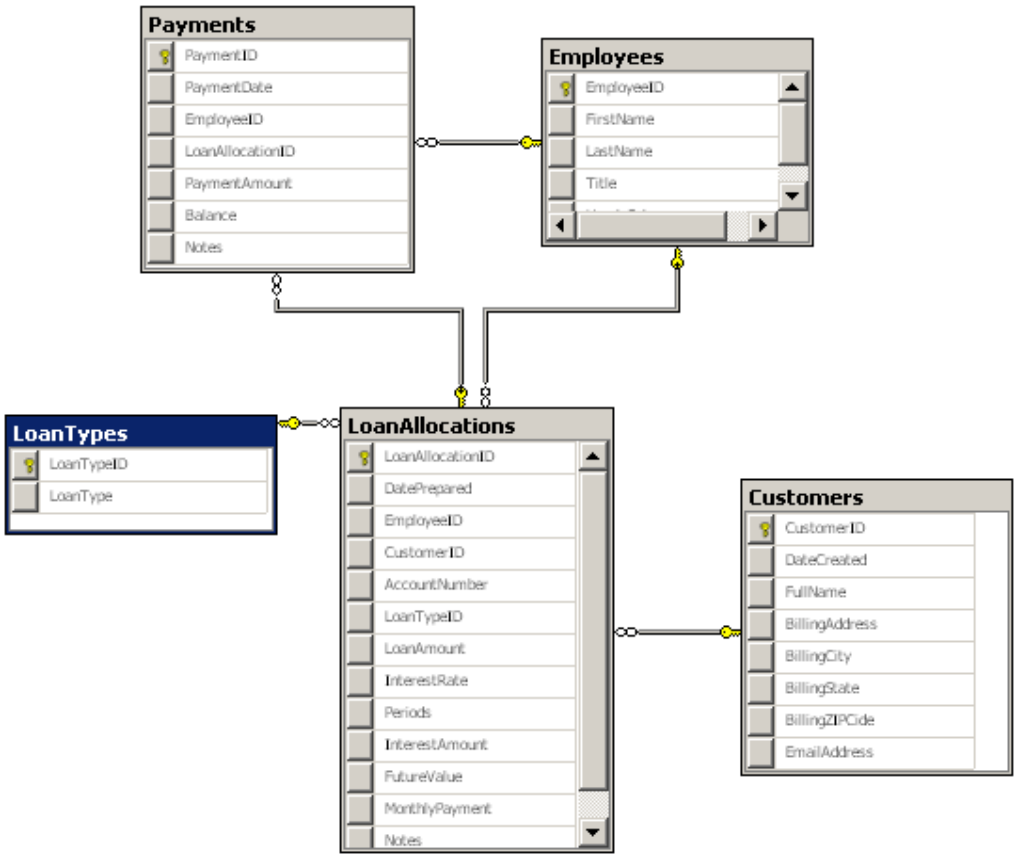
```
VALUES ('492947 ',' Helene ',' Gustman ',' Cuentas de Representante ", 14,55);
IR
INSERT INTO dbo.Employees (EmployeeNumber, Nombre, Apellido, Título,
HourlySalary)
VALUES ('804685 ',' Ernesto ',' Thomas', 'Cuentas de Representante ", 12,75);
IR
- ==========================================
- Tabla: LoanTypes
- ==========================================
USO WattsALoan
IR

SI object_id ( 'dbo.LoanTypes',' U ') IS NOT NULL
   DROP TABLE dbo.LoanTypes
IR

CREATE TABLE dbo.LoanTypes
(
      Int LoanTypeID identidad (1,1) NOT NULL,
      LoanType varchar (50) NOT NULL,
      PK_LoanTypes CONSTRAINT PRIMARY KEY (LoanTypeID)
);
IR
INSERT INTO LoanTypes (LoanType) VALUES ( 'Préstamo Personal');
IR
INSERT INTO LoanTypes (LoanType) VALUES ( 'coche Financiación');
IR
INSERT INTO LoanTypes (LoanType) VALUES ( 'Tarjeta de Crédito');
IR
INSERT INTO LoanTypes (LoanType) VALUES ( 'Mobiliario de Préstamo');
IR
- ==========================================
- Tabla: Clientes
- ==========================================
USO WattsALoan
IR

SI object_id ( 'dbo.Customers',' U ') IS NOT NULL
   DROP TABLE dbo.Customers
IR

CREATE TABLE dbo.Customers
(
      IdCliente int identidad (1,1) NOT NULL,
      DateCreated datetime NULL,
      FullName varchar (50) NOT NULL,
      BillingAddress varchar (100),
      BillingCity varchar (50),
      BillingState varchar (50),
      BillingZIPCide varchar (10),
      Direcciónde correo electrónico varchar (100),
      PK_Customers CONSTRAINT PRIMARY KEY (IdCliente)
),
IR
INSERT INTO Clientes (DateCreated, FullName,
      BillingAddress, BillingCity, BillingState,
      BillingZIPCide, DIRECCIÓN DE CORREO ELECTRÓNICO)
```

```
VALUES (2 / 26/2004 ',' Julius Ramse »,
          '927 Feuler Ave ',' Silver Spring »,
          'MD','20904 ',' ramses1990@netscape.net ');
IR
INSERT INTO Clientes (DateCreated, FullName,
      BillingAddress, BillingCity, BillingState,
           BillingZIPCide)
VALUES ('06 / 22/2006 ',' Gertrude Vaillant ',
          '10055 Larsenic Road ',' Takoma Park,
          'MD','20910 ');
IR
INSERT INTO Clientes (DateCreated, FullName,
      BillingAddress, BillingCity, BillingState,
           BillingZIPCide, DIRECCIÓN DE CORREO ELECTRÓNICO)
VALUES ( «12 / 3 / 2004 ',' James Barrouch»,
          '4204 Fallon Drive ',' Silver Spring »,
          'MD','20906 ',' barrouchj@hotmail.com ');
IR
INSERT INTO Clientes (DateCreated, FullName,
      BillingAddress, BillingCity, BillingState,
           BillingZIPCide)
VALUES ('08 / 02/2006 ',' Christine ásperas »,
          Dotación'825 Street ',' Alejandría ',
          'VA','22231 ');
IR
INSERT INTO Clientes (DateCreated, FullName,
      BillingAddress, BillingCity, BillingState,
           BillingZIPCide, DIRECCIÓN DE CORREO ELECTRÓNICO)
VALUES ('10 / 08/2006 ',' Patrick Heller »,
          Drive NW Clarington'2480 ',' Washington ',
          'DC','20006 ',' hellerp@yahooo.com ');
IR
- ==========================================
- Tabla: LoanAllocation
- ==========================================
USO WattsALoan
IR

SI object_id ( 'dbo.LoanAllocations',' U ') IS NOT NULL
   DROP TABLE dbo.LoanAllocations
IR

CREATE TABLE dbo.LoanAllocations
(
      Int LoanAllocationID identidad (1,1) NOT NULL,
      DatePrepared datetime NOT NULL,
      IdEmpleado int NULL
                    CONSTRAINT FK_LoanPreparer
                    FOREIGN KEY REFERENCIAS Empleados (IdEmpleado),
      IdCliente int NOT NULL
                    CONSTRAINT FK_LoanReceiver
                    FOREIGN KEY Referencias Clientes (IdCliente),
      AccountNumber char (10),
      LoanTypeID int NOT NULL
                    CONSTRAINT FK_LoanTypes
                    REFERENCIAS LoanTypes FOREIGN KEY (LoanTypeID),
      LoanAmount dinero NOT NULL,
```

```
        InterestRate decimal (6,2) NOT NULL,
        Períodos decimal (6,2) NOT NULL,
        InterestAmount COMO ((LoanAmount * (InterestRate / (100 )))*( períodos
/ (12))),
        FutureValue AS (LoanAmount + (LoanAmount * (InterestRate / (100 )))*(
períodos / (12))),
        MonthlyPayment COMO ((LoanAmount + (LoanAmount * (InterestRate /
(100 )))*( períodos / (12))) / períodos),
        Toma nota de texto,
        PK_LoanAllocations CONSTRAINT PRIMARY KEY (LoanAllocationID)
  ),
  IR
  INSERT INTO LoanAllocations (DatePrepared, IdEmpleado,
            IdCliente, AccountNumber, LoanTypeID, LoanAmount,
        InterestRate, períodos, Notas)
  VALUES (2 / 26/2004 ', 2, 1,'9171394', 4, 6.500,00, 12,65, 36,
           "El préstamo será entregado por nuestro socio de negocios de muebles
Helios Furnian ');
  IR
  INSERT INTO LoanAllocations (DatePrepared, IdEmpleado,
            IdCliente, AccountNumber, LoanTypeID, LoanAmount,
        InterestRate, períodos, Notas)
  VALUES ('06 / 22/2007 ', 2, 2,'8628064', 2, 16.500,00, 10,20, 60,
           'Por este préstamo, nuestro socio de Arlington Honda proceso y
entregar el automóvil.');
  IR
  INSERT INTO LoanAllocations (DatePrepared, IdEmpleado,
            IdCliente, AccountNumber, LoanTypeID, LoanAmount,
        InterestRate, períodos, Notas)
  VALUES ( «12 / 3 / 2006 ', 1, 3,'8468364', 3, 500,00, 18,65, 48,
           'Esta es una tarjeta de crédito.');
  IR
  INSERT INTO LoanAllocations (DatePrepared, IdEmpleado,
            IdCliente, AccountNumber, LoanTypeID, LoanAmount,
        InterestRate, períodos, Notas)
  VALUES ('08 / 02/2006 ', 3, 4,'2483047', 1, 3.500,00, 12,74, 36,
           'Esta es personal / préstamo en efectivo asignado a un cliente que
caminó en la tienda y le pidió que.');
  IR
  INSERT INTO LoanAllocations (DatePrepared, IdEmpleado,
            IdCliente, AccountNumber, LoanTypeID, LoanAmount,
        InterestRate, períodos, Notas)
  VALUES ('10 / 08/2006 ', 2, 5,'1311804', 4, 22.748,36, 12,28, 60,
           'Esto es un coche de financiación de préstamos');
  IR
  - ==========================================
  - Cuadro: Pagos
  - ==========================================
  USO WattsALoan
  IR

  SI object_id ( 'dbo.Payments',' U ') IS NOT NULL
     DROP TABLE dbo.Payments
  IR

  CREATE TABLE dbo.Payments
  (
```

```
        Int PaymentID identidad (1, 1) NOT NULL,
        PaymentDate datetime NOT NULL,
        IdEmpleado int NULL
                        CONSTRAINT FK_Employees
                        FOREIGN KEY REFERENCIAS Empleados (IdEmpleado),
        LoanAllocationID int NOT NULL
                        CONSTRAINT FK_LoanAllocations
                        REFERENCIAS LoanAllocations FOREIGN KEY
(LoanAllocationID),
        PaymentAmount dinero NOT NULL,
        Balance de dinero,
        Toma nota de texto,
        PK_Payments CONSTRAINT PRIMARY KEY (PaymentID)
  ),
  IR
```

4. Para ejecutar el código, presione F5
5. En el Explorador de objetos, expanda el nodo Bases de datos y si es necesario ampliar WattsALoan
6. Haga clic en Diagrama de base de datos
7. Cuando el cuadro de mensaje aparece, lea y haga clic en Sí.
8. Haga clic con el botón Diagrama de base de datos y haga clic en Nuevo diagrama de base de datos ...
9. En el cuadro de diálogo, haga doble clic en cada cuadro y, cuando todas las tablas se han añadido, haga clic en Cerrar



10. Guardar el diagrama como **dgmWattsALoan** y ciérrela

Crear un procedimiento almacenado

Para crear un procedimiento:

- En el Explorador de objetos, expanda la base de datos para la que desea crear el procedimiento, a programabilidad nodo, haga clic derecho en procedimientos almacenados, y haga clic en Nuevo pro almacenado ... Una ventana de consulta con un esqueleto de sintaxis se muestran. A continuación modificar el código que utilizan las técnicas que aprenderá en esta lección.
- Abra una ventana de consulta vacía asociada con la base de datos para la que desea crear el proce almacenado y mostrar el Explorador de plantillas. En el Explorador de plantillas, expanda el nodo d procedimiento. Crear un procedimiento almacenado de arrastrar y soltar en la ventana de consulta
- Abra una ventana de consulta vacía asociada con la base de datos para la que desea crear el proce almacenado e introduzca el código necesario

En SQL, para crear un procedimiento, de comenzar con el **CREATE PROCEDURE** expresión. También **CREATE PROC.** Ambas expresiones producir el mismo resultado. Como todo en su base de datos, de procedimiento:

- El nombre de un procedimiento puede ser cualquier cadena que sigue las reglas que hemos examin **nombrar las funciones**
- Abstenerse de iniciar el nombre de un procedimiento con **sp_** porque podrían entrar en conflicto co procedimientos almacenados que ya vienen con Microsoft SQL Server.

Después de que el nombre del procedimiento, escriba la palabra clave **AS.** La sección, grupo de pal líneas después de la palabra clave **como** se llama el cuerpo del procedimiento. Afirma lo que ust procedimiento para hacer o lo que usted quisiera que producen.

Sobre esta base, la sintaxis más simple de crear un procedimiento es:

CREATE PROCEDURE *ProcedureName*
COMO
Cuerpo del procedimiento

Es importante tener en cuenta que hay muchas otras cuestiones relacionadas con la creación de un pro por ahora, vamos a considerar que la sintaxis.

Después de crear el procedimiento, debe guardarlo como un objeto en su base de datos. Para ello herramientas del Editor de SQL, puede hacer clic en el botón Ejecutar . Si el código del procedimi que sería creado y un nuevo nodo de su nombre se agregará a la sección de procedimientos almacenad datos.

Procedimientos de gestión

Modificación de un procedimiento

Como un objeto de base de datos de SQL Server, puede modificar un procedimiento almacenado sin q ello:

- En el Explorador de objetos, puede hacer clic derecho en el procedimiento y haga clic en Modificar
- En el Explorador de objetos, puede hacer clic derecho en el procedimiento, coloque el ratón sobre procedimiento almacenado Como -> Para ALTER -> Nueva ventana del editor de Consulta
- Abra una ventana de consulta vacía asociada con la base de datos que contiene el procedimiento a Desde el Explorador de plantillas, ampliar el procedimiento almacenado. Arrastre el nodo de un pro almacenado de la gota y soltar en la ventana de consulta vacía

En cada caso, un mínimo de código se genera para usted. A continuación, puede editar para crear una

su procedimiento almacenado. Después de editar el código, puede ejecutar la sentencia SQL p procedimiento almacenado.

En SQL, la fórmula básica para modificar un procedimiento almacenado es:

ALTER PROCEDURE *ProcedureName*
COMO
Órgano de Procedimiento

Eliminación de un procedimiento

Una de las mayores características de un procedimiento almacenado es que es tratado como un obj derecho. Por lo tanto, después de crearlo, si no necesita más, usted puede deshacerse de ella.

Existen varios tipos de procedimientos almacenados, algunos de los cuales se consideran de carácte tipos de procedimientos de borrarse a sí mismo cuando no se necesita más, como cuando la pers procedimiento almacenado se desconecta de la base de datos o se apaga el equipo. En caso contrario, procedimiento, puede usar el Explorador de objetos o SQL. Como se mencionó con mesas, aun cu procedimiento mediante el Explorador de objetos, puede eliminarla utilizando SQL y viceversa.

Para eliminar un procedimiento en el Explorador de objetos, después de la ampliación de su ba programación, y su procedimiento almacenado nodos, puede hacer clic derecho en el procedimiento al clic en Eliminar. También puede hacer clic en el Explorador de objetos para seleccionar y luego pulse E el cuadro de diálogo Objeto vendría a dejar de tomar una decisión.

Para eliminar un procedimiento en SQL, la sintaxis a utilizar es:

DROP PROCEDURE *ProcedureName*

Por supuesto, usted debe asegurarse que está en la base de datos y también que el *ProcedureName* exi

Exploración de Procedimientos

Introducción

Probablemente el procedimiento más simple se puede escribir consistiría en la selección de columnas de se hace con la **selección** del operador y la aplicación de las técnicas que hemos revisado para el análi ejemplo, para crear un procedimiento almacenado que se celebre una lista de estudiantes de una t Estudiantes, que daría lugar al procedimiento de la siguiente manera:

```
CREATE PROCEDURE GetStudentIdentification
COMO
COMENZAR
    SELECCIONE Nombre, Apellido, DateOfBirth, Género
    De los estudiantes
FIN
IR
```

Además de las operaciones de **selección,** en un procedimiento almacenado, puede realizar c operaciones de la base de datos que hemos aplicado hasta ahora. Estos incluyen la creación y m registros, etc

Aprendizaje práctico: Crear un procedimiento almacenado

1. Asegúrese de que tiene el **Banco Nacional de Yugoslavia** en la base de datos creada Lesson13. crea, lo hacen ahora.
   En el Explorador de objetos, haga clic derecho en YugoNationalBank y haga clic en Nueva consulta
2. Para iniciar un procedimiento almacenado, escriba lo siguiente:

   ```
   USO YugoNationalBank;
   IR
   ```

```
- ==============================================
- Autor: FunctionX
- Crear la fecha: Viernes, 25 de mayo de 2007
- Descripción: Este procedimiento almacenado se asigna un
- Contraseña por defecto para cada empleado.
- ==============================================

CREATE PROCEDURE AssignDefaultPassword
COMO
COMENZAR
        ACTUALIZACIÓN dbo.Employees
        SET PASSWORD = 'Password1' DE dbo.Employees;
FIN
IR
```

3. Para crear el procedimiento almacenado, presione F5

La ejecución de un procedimiento

Para obtener los resultados de la creación de un procedimiento almacenado, debe ejecutarlo (en otras un procedimiento almacenado, debe llamar a la misma). Para ejecutar un procedimiento, se utiliza **EXECUTE** seguido por el nombre del procedimiento. Aunque hay algunas otras cuestiones relacionadas de un procedimiento, por ahora, consideran que la más sencilla sintaxis para llamar a un procedimiento

EXECUTE *ProcedureName*

Alternativamente, en lugar de **EXECUTE,** puede utilizar la palabra clave **EXEC:**

EXEC *ProcedureName*

Por ejemplo, si tiene un procedimiento llamado GetStudentIdentification, para ejecutarlo, debería escrib

EXECUTE GetStudentIdentification

También puede preceder el nombre del procedimiento con su esquema, como **dbo.** Aquí está un ejemp

EXECUTE dbo. GetStudentIdentification;

También puede preceder el nombre del esquema con el nombre de la base de datos. Aquí está un ejem

EXECUTE Rosh. Dbo.GetStudentIdentification;

Aprendizaje práctico: La ejecución de un procedimiento almacenado

1. Elimine el contenido de la ventana de consulta y reemplazarlo por el texto siguiente:

```
EXECUTE AssignDefaultPassword;
IR
```

2. Para ejecutar el procedimiento almacenado, presione F5

Uso de expresiones y funciones

Una de las ventajas de utilizar un procedimiento almacenado es que no sólo puede producir la misma hemos visto durante el análisis, pero también puede almacenar este tipo de expresión que recor momento sin tener que volver a escribir (la expresión). Sobre esta base, puede crear una expresión primer y un apellido para producir y almacenar un nombre completo. Aquí está un ejemplo:

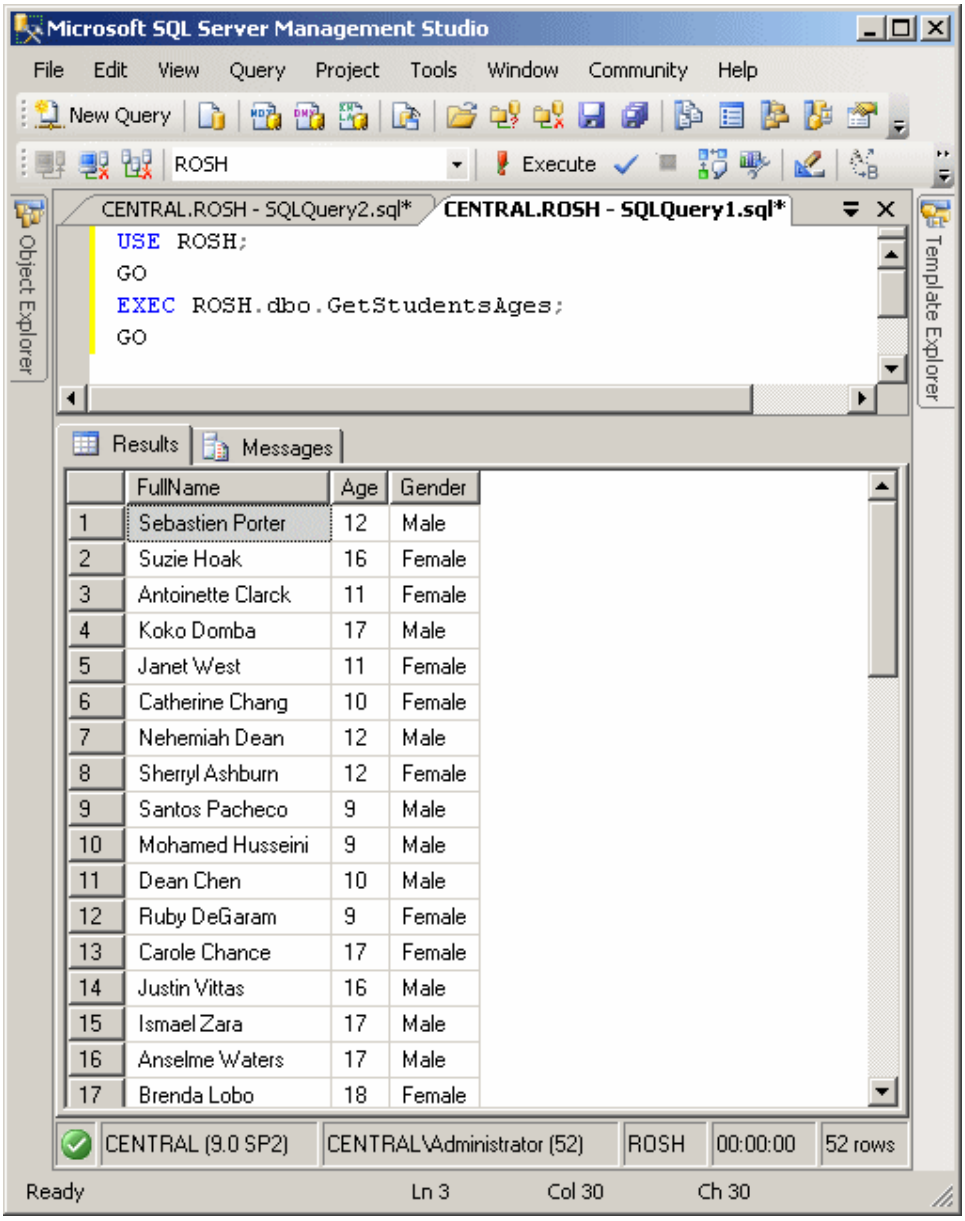
```
CREATE PROCEDURE GetStudentIdentification
COMO
COMENZAR
    SELECCIONE FullName = Nombre + "" + Apellido,
            DateOfBirth, Género
```

```
    De los estudiantes
FIN
IR
```

Un procedimiento almacenado también puede llamar a una función en su cuerpo. Para ello, siga las mis
hemos examinado para **llamar funciones durante el análisis de los datos.** Aquí hay un ejemplo de
almacenado que llama a una función:

```
Rosh USO;
IR
CREATE PROCEDURE GetStudentsAges
COMO
COMENZAR
    SELECCIONE FullName = Nombre + "" + Apellido,
            DATEDIFF (año, DateOfBirth, GETDATE ()) como la edad,
            Género
    De los estudiantes
FIN
IR
```

Aquí hay un ejemplo de la ejecución del procedimiento:



Aprendizaje práctico: Realización de una llamada a la función en un procedimiento almacenado

1. Elimine el contenido de la ventana de consulta
2. Para iniciar un nuevo procedimiento almacenado, escriba lo siguiente:

```
USO YugoNationalBank;
IR

- ==========================================
- Autor: FunctionX
- Crear la fecha: Viernes, 25 de mayo de 2007
- Descripción: crea un procedimiento almacenado
- Nombre de usuario para cada empleado.
- También se asigna a un mensaje de correo electrónico del empleado.
- ==========================================
```

```
CREATE PROCEDURE CreateUsername
COMO
COMENZAR
        ACTUALIZACIÓN dbo.Employees
        AJUSTE Usuario = LOWER (IZQUIERDA (Nombre, 1) + IZQUIERDA
(Apellido, 5))
                DE dbo.Employees;
        ACTUALIZACIÓN dbo.Employees
        AJUSTE direcciónde correo electrónico = LOWER (IZQUIERDA (Nombre,
1) + IZQUIERDA (Apellido, 5)) +
                                        '@ yugonationalbank.com'
                                        DE dbo.Employees;
FIN
IR
```

3. Para crear el procedimiento almacenado, presione F5
4. Elimine el contenido de la ventana de consulta y reemplazarlo por el texto siguiente:

```
EXECUTE CreateUsername;
IR
```

5. Para ejecutar el procedimiento almacenado, presione F5

Argumentos y Parámetros

Introducción

Todos los procedimientos almacenados que hemos creado y utilizado hasta la fecha del supuesto de qu ya eran necesarias en una tabla de la base de datos. En algunos casos, puede que necesite crear almacenado que incluye los valores que no son parte de la base de datos. En este escenario, por el pr su asignación, se le facilitará uno o más valores.

Un valor externo que se proporciona a un procedimiento almacenado es un parámetro. Al crear almacenado, también debe crear el parámetro, si lo consideran necesario. Cuando un procedimiento equipado con un parámetro, se dice que el procedimiento almacenado tiene un argumento. U almacenado también puede tomar más de un argumento.

Al ejecutar un procedimiento almacenado que toma uno o más argumentos, debe proporcionar un argumento. En este caso, se dice que pasar un valor para el argumento. Hay casos en que no tiene que argumento.

Pasando Argumentos

Para crear un procedimiento almacenado que tiene un argumento, escriba la fórmula **de CREATE** CREATE **PROC** seguida del nombre del procedimiento, a continuación, escriba el nombre del argumento parámetro se crea como una columna de una tabla. Es decir, un parámetro debe tener un nombre, u una longitud opcional. Aquí está la sintaxis que se utiliza:

CREATE PROCEDURE *ProcedureName*
@ *ParameterName DataType*
COMO
Cuerpo del procedimiento

Al aplicar el procedimiento almacenado, puede definir lo que quiere hacer con el parámetro (s), procedimiento. Una de las formas que puede utilizar un parámetro es ejecutar una consulta cuyo fact proporcione. Por ejemplo, imagine que desea crear un procedimiento que, siempre que sea ejecutado, un género y, a continuación, se mostrará la lista de los estudiantes de que las cuestiones de género. desea que el usuario especifique el género de los estudiantes a visualizar, puede crear un procedimiento recibe el género. Aquí está un ejemplo:

```
CREAR PROC GetListOfStudentsByGender
        @ GdR VARCHAR (12)
COMO
SELECCIONE Nombre, Apellido,
        DateOfBirth, HomePhone, Género
De los estudiantes
DÓNDE Género = @ GdR
```

Aprendizaje práctico: Crear un procedimiento almacenado

1. Elimine el contenido de la ventana de consulta y, para pasar argumentos a un procedimiento alma lo siguiente en la ventana:

```
USO WattsALoan;
IR

CREATE PROCEDURE SpecifyCurrentBalance
      @ PmtDate datetime,
      @ EmplID int,
      AL @ int,
      @ PmtAmt dinero
COMO
COMENZAR
      - Obtener la cantidad que se presta al cliente
      DECLARE @ AmountOfLoan dinero;
      SET @ AmountOfLoan = (SELECT las.FutureValue
                            DE LoanAllocations las
                            WHERE (las.LoanAllocationID = @ AL));

      - Si el cliente ya ha hecho al menos un pago,
      - Obtener el saldo actual de la cuenta del cliente
      DECLARE @ CurrentBalance dinero;
      SET @ CurrentBalance = (SELECT MIN (pay.Balance)
                                  Pago de los pagos
                                  WHERE (pay.LoanAllocationID = @ AL));

      - Si el cliente nunca ha hecho un pago (todavía),
      - Especificar el saldo, resta el actual pago
      - A partir de la cantidad original del préstamo
      SI CurrentBalance @ IS NULL
          COMENZAR
              INSERT INTO Pagos (PaymentDate, IdEmpleado,
                          LoanAllocationID, PaymentAmount, Balance)
              VALUES (@ PmtDate, @ EmplID, AL @, @ PmtAmt,
                      AmountOfLoan @ - @ PmtAmt);
          FIN
      - Si el cliente ya había al menos un pago,
      - Restar el pago corriente de los últimos equilibrio
      ELSE
          COMENZAR
              INSERT INTO Pagos (PaymentDate, IdEmpleado,
                                  LoanAllocationID, PaymentAmount, Balance)
              VALUES (@ PmtDate, EmplID @, @ definido,
                      @ PmtAmt, CurrentBalance @ - @ PmtAmt);
          FIN
FIN
IR
```

2. Para crear el procedimiento almacenado, presione F5

La ejecución de un procedimiento almacenado argumentativo

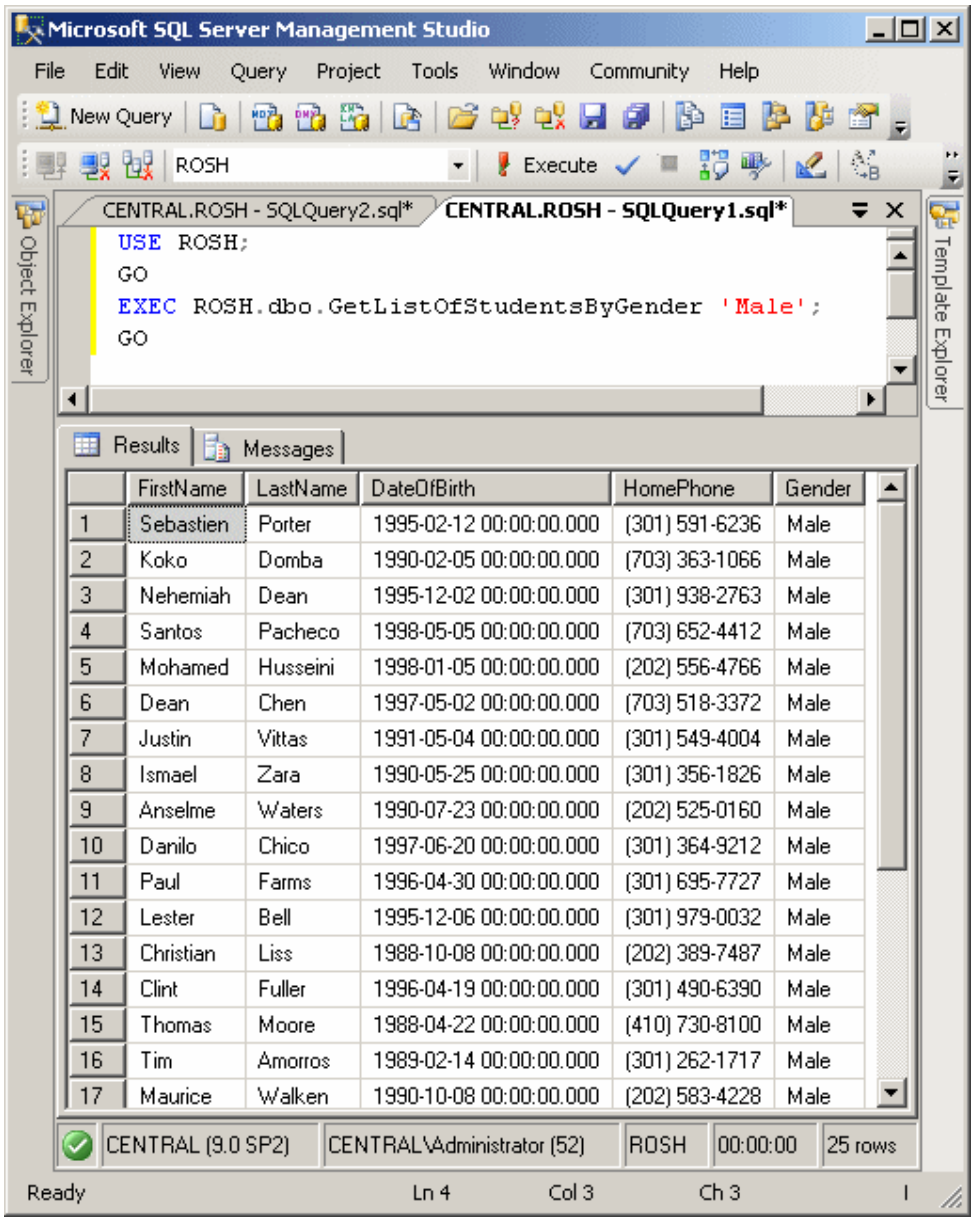
Como ya se ha mencionado, al ejecutar un procedimiento almacenado que toma un parámetr proporcionar un valor para el parámetro. La sintaxis utilizada es la siguiente:

EXEC *ProcedureName ParameterValue*

Si el parámetro es booleano o numérico, asegúrese de que el valor 0 o por un valor booleano o de otr tipo numérico. Si el parámetro es un carácter o una cadena, escriba su valor en una sola cita. Aquí esta

EXEC ROSH.dbo.GetListOfStudentsByGender 'Hombre';

Aquí hay un ejemplo de ejecución es el siguiente:



Nótese que podría / debería haber omitido incluir la columna de Género en el estado, ya que de lo que usuario.

Otro tipo de procedimiento almacenado se puede hacer para tener más de un parámetro. En est parámetros de la sección antes de la palabra clave **como,** separados por una coma. La sintaxis qu siguiente:

CREATE PROCEDURE *ProcedureName*
@ *ParameterName1 DataType,* **@** *ParameterName2 DataType,* **@** *ParameterName_n DataType*
COMO
Cuerpo del procedimiento

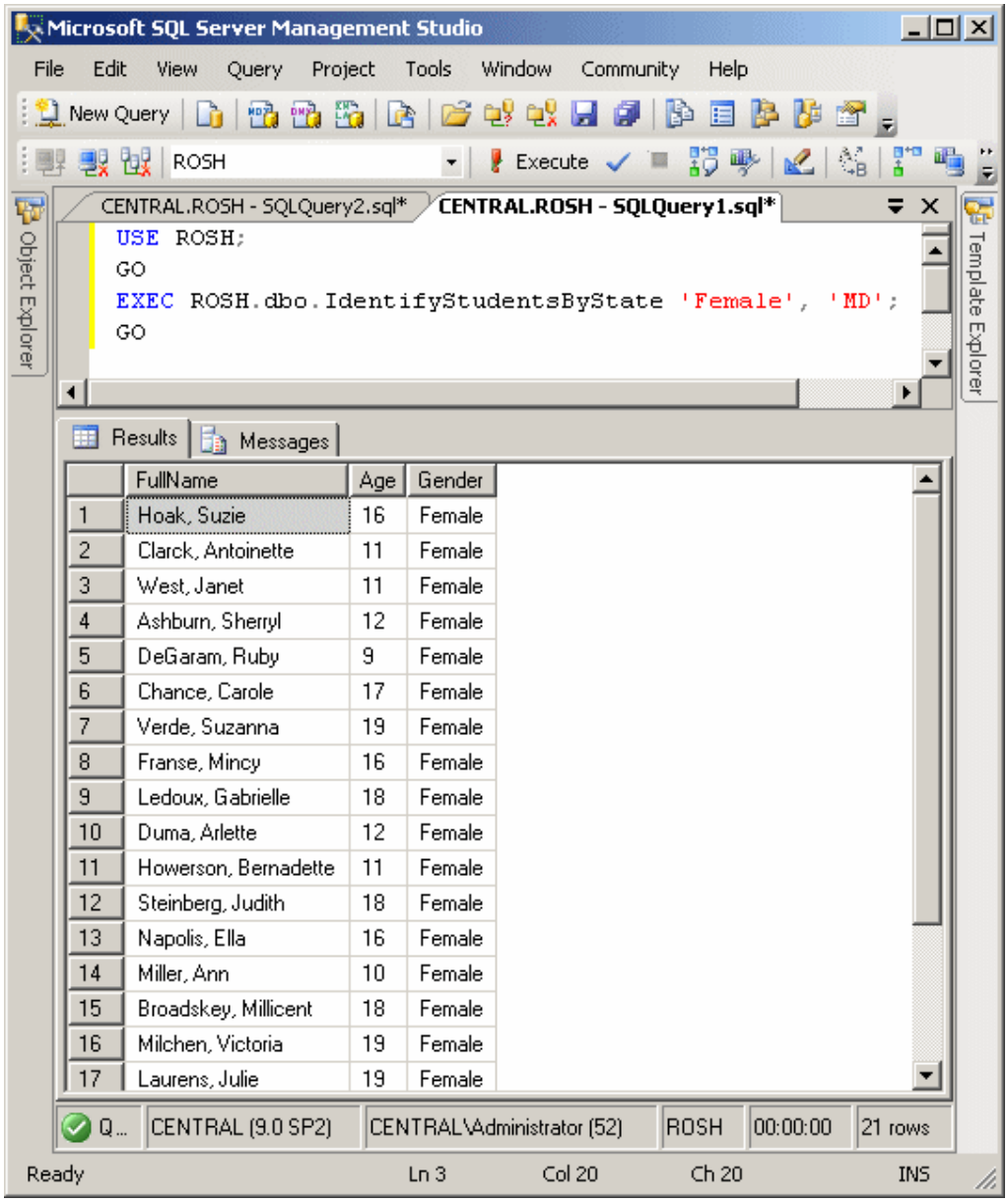
Aquí está un ejemplo:

```
Rosh USO;
IR
CREATE PROCEDURE IdentifyStudentsByState
        @ GdR varchar (20),
        @ StateOrProvince char (2)
COMO
COMENZAR
    SELECCIONAR Apellido FullName = + ',' + FirstName,
            DATEDIFF (año, DateOfBirth, GETDATE ()) como la edad,
            Género
    De los estudiantes
    WHERE (Sexo = @ RDA) Y (Estado = @ StateOrProvince)
FIN
IR
```

Cuando llamas a un procedimiento almacenado que tiene más de un parámetro, debe proporcionar ur parámetro, pero usted tiene dos alternativas. La técnica más simple consiste en proporcionar un parámetro en el orden exacto en que aparecen en el procedimiento almacenado. Aquí está un ejemplo:

```
Rosh USO;
IR
EXEC ROSH.dbo.IdentifyStudentsByState 'Mujeres',' MD ';
IR
```

Esto produciría:



Alternativamente, usted puede proporcionar el valor de cada parámetro en el orden de su elecció
siguiente procedimiento que toma 3 argumentos:

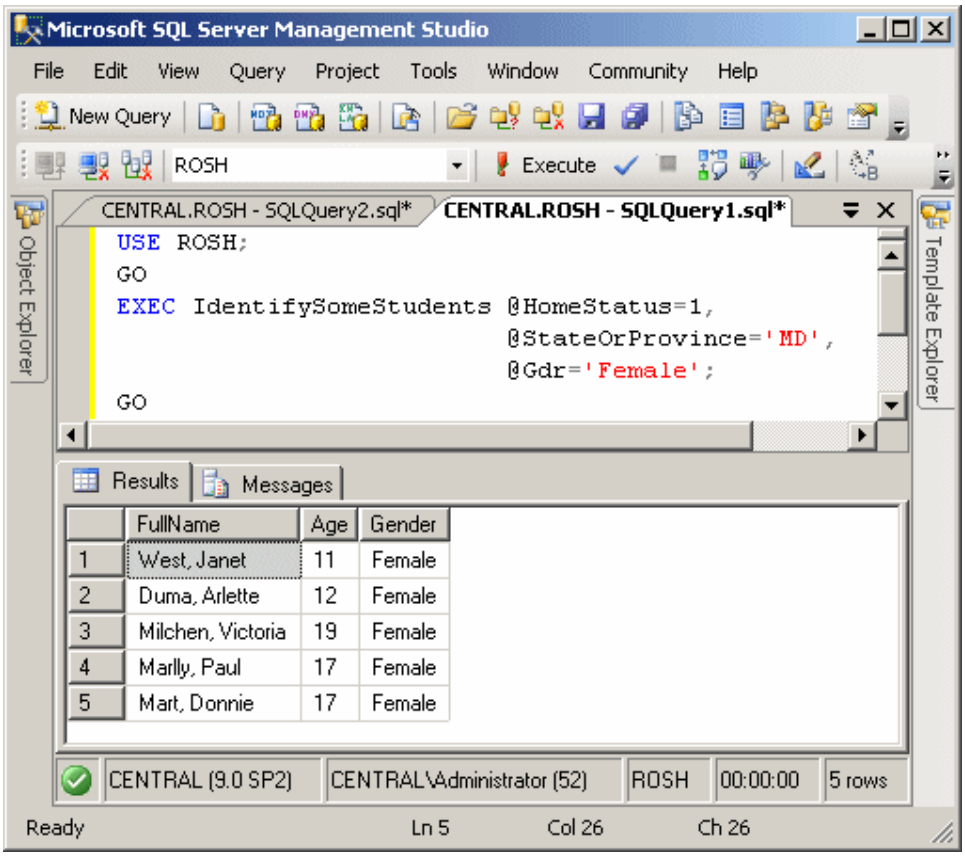
```
Rosh USO;
IR
CREATE PROCEDURE IdentifySomeStudents
        @ GdR varchar (20),
        @ StateOrProvince char (2),
        @ HomeStatus poco
COMO
COMENZAR
    SELECCIONAR Apellido FullName = + ',' + FirstName,
            DATEDIFF (año, DateOfBirth, GETDATE ()) como la edad,
            Género
    De los estudiantes
    WHERE (Sexo = @ RDA) Y
```

(Estado = @ StateOrProvince) Y
(SPHome = @ HomeStatus)
FIN
IR

Al llamar a este tipo de procedimiento, puede escribir el nombre de cada parámetro y asignarle el valor
Aquí está un ejemplo:

IdentifySomeStudents EXEC @ HomeStatus = 1, @ StateOrProvince = 'MD', @ GdR = 'Mujeres';

Aquí hay un ejemplo de la ejecución del procedimiento:



Aprendizaje práctico: Procedimiento de ejecución una argumentativo

1. Borrar el contenido de la ventana de código y, a fin de crear un procedimiento almacenado, escriba la ventana:

   USO WattsALoan;
   IR
   EXECUTE SpecifyCurrentBalance'03 / 25/2004 ', 2, 1, 249,08;
   IR
   EXECUTE SpecifyCurrentBalance'01 / 30/2006 ', 2, 5, 611,93;
   IR
   EXECUTE SpecifyCurrentBalance'04 / 20/2004 ', 1, 1, 249,08;
   IR
   EXECUTE SpecifyCurrentBalance'10 / 28/2006 ', 2, 4, 134,38;
   IR

2. A ejecutar, pulse la tecla F5

Argumentos por defecto

Imagínese que usted crear una base de datos para una tienda y una tabla que contiene la lista de artí
la tienda:

CENTRAL.Depar...emsCategories | CENTRAL.Depa...bo.StoreItems

Column Name	Data Type	Allow Nulls
ItemCategoryID	int	☐
ItemCategory	varchar(50)	☐
		☐

CENTRAL.Depar...emsCategories | CENTRAL.Depa...bo.StoreItems

Column Name	Data Type	Allow Nulls
StoreItemID	int	☐
ItemNumber	char(8)	☐
ItemCategoryID	int	☑
ItemName	varchar(50)	☐
ItemSide	varchar(30)	☑
UnitPrice	smallmoney	☑
		☐

Supone que han llenado la mesa con unos cuantos objetos de la siguiente manera:

CENTRAL.Depar...emsCategories | CENTR

ItemCategoryID	ItemCategory
1	Men
2	Women
3	Boys
4	Girls
5	Miscellaneous
NULL	NULL

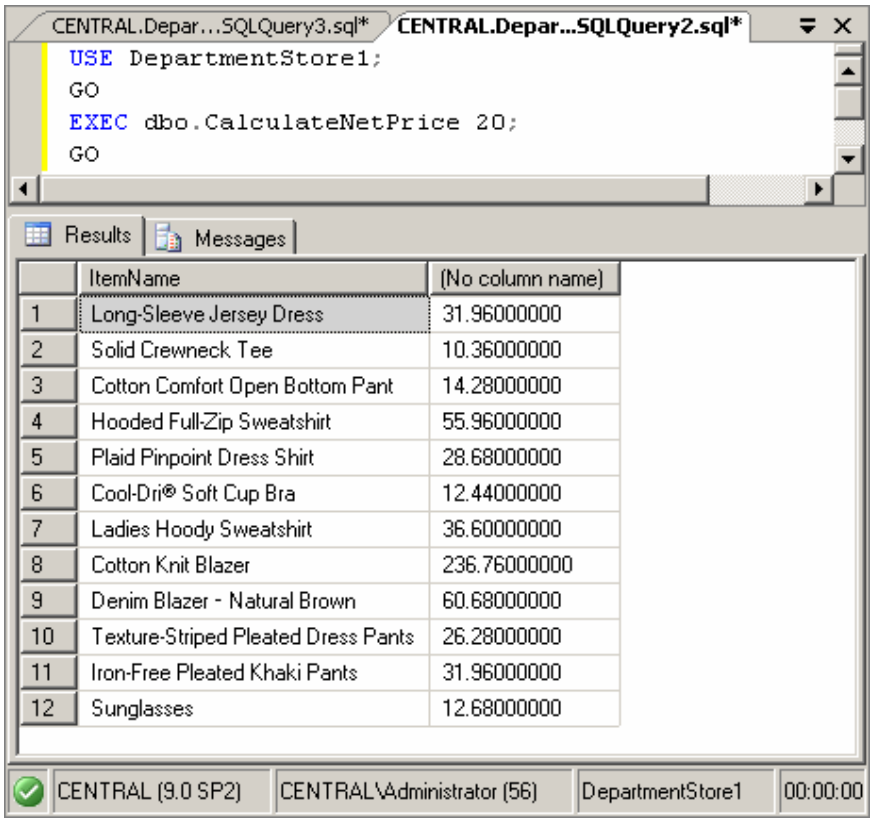
ItemNumber	ItemCategoryID	ItemName	ItemSize	UnitPric
264850	2	Jersey de manga larga-Vestido	Petite	39,95
930405	4	Sólidos Crewneck Tee	Medio	12,95
293004	1	Abrir el fondo de algodón Confort Pant	XLarge	17,85
924515	1	Completo con capucha Sudadera-Zip	S	69,95
405945	3	Plaid Definir Viste camiseta	22 35-36	35,85
294936	2	De enfriamiento en seco suave Cup Bra	36D	15,55
294545	2	Señoras Sudadera con capucha	Medio	45,75
820465	2	Blazer de algodón Knit	H	295,95
294694	2	Denim Blazer - Natural Brown	Grandes	75,85
924094	3	Textura-Striping plisado DRESS	44x30	32,85

		PANTS		
359405	3	Libre de hierro plisado Caqui Pantalones	32x32	39,95
192004	3	Gafas de sol		15,85

Imagínese que usted desea crear un mecanismo de cálculo del precio de un artículo después de un aplicado a la misma. Este procedimiento se puede crear de la siguiente manera:

CREAR PROC CalculateNetPrice
@ descuento decimales
COMO
SELECCIONE ItemName, UnitPrice - (* @ UnitPrice descuento / 100)
DE StoreItems

Esto puede ser ejecutado de la siguiente manera:



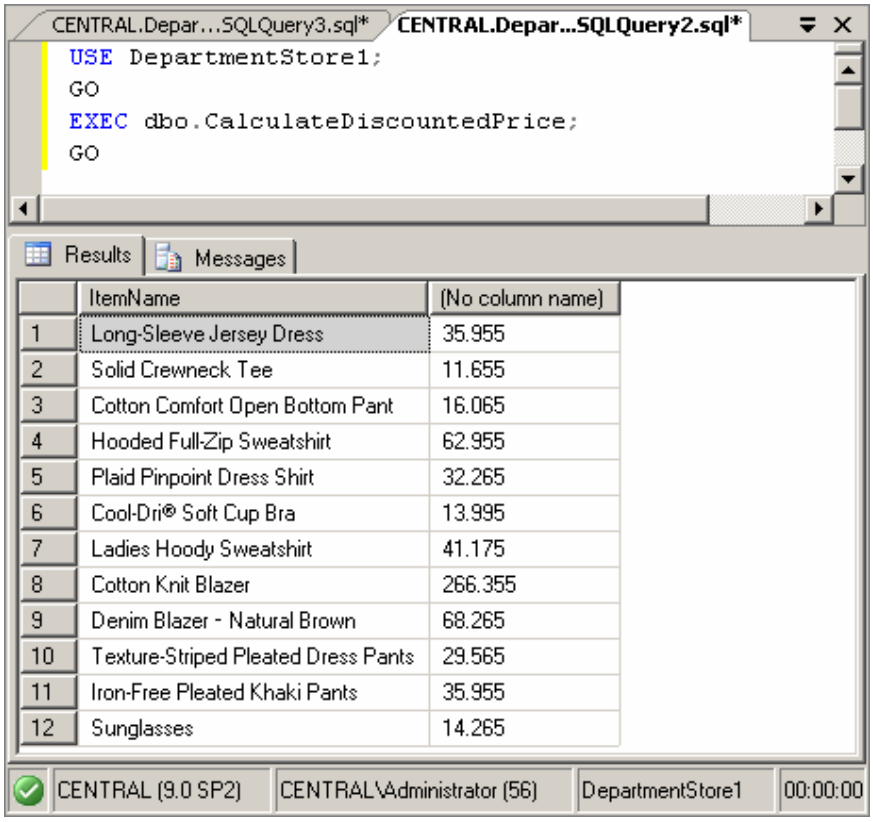
Si está pensando en crear un procedimiento almacenado que tiene un argumento y el argumento de qu tendrá el mismo valor la mayor parte del tiempo, puede proporcionar ese valor como parámetro, pero para otros valores de este argumento. Un valor a un argumento que se conoce como defecto. Lo qu que, cuando el usuario las llamadas de procedimiento almacenado que, si el usuario no tiene que prop para el argumento, el valor por defecto se utilizará.

Para crear un procedimiento almacenado que tiene un argumento que lleva a un valor por defecto, des el valor, sobre su lado derecho, escriba = seguido por el valor deseado. Aquí hay un ejemplo aplic datos:

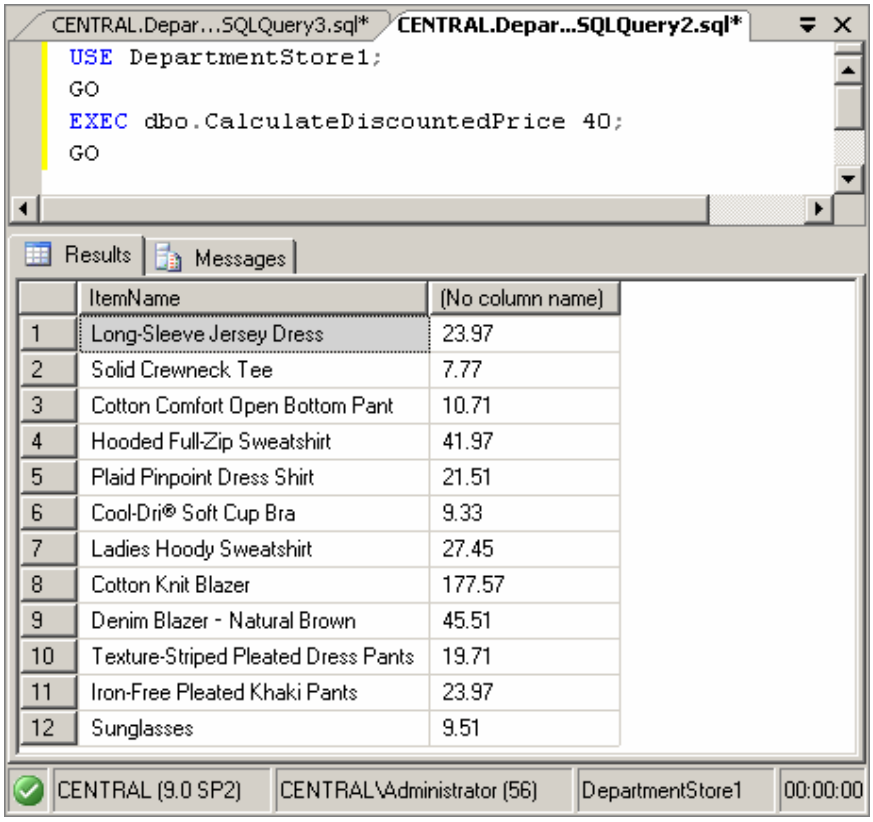
CREAR PROC CalculateDiscountedPrice
@ descuento decimales = 10.00
COMO

SELECCIONE ItemName, UnitPrice - (* @ UnitPrice descuento / 100)
DE StoreItems;
IR

Al ejecutar un procedimiento almacenado que toma un argumento por defecto, no tiene que proporcio el argumento de si el valor por defecto más le convenga. Sobre esta base, el procedimiento almacena puede ser llamada de la siguiente manera:



Si el valor por defecto no se aplica a su cálculo actual, puede proporcionar un valor para el argumen ejemplo:



El uso de este mismo enfoque, se puede crear un procedimiento almacenado que tiene más de u valores por defecto. Para proporcionar un valor por defecto para cada argumento, después de declara valor deseado a su lado derecho. Aquí hay un ejemplo de un procedimiento almacenado que tiene dos a uno con un valor por defecto:

```
CREAR PROC CalculateSalePrice
Descuento @ decimal = 20.00,
@ TaxRate decimal = 7,75
COMO
SELECCIONE ItemName Como [Descripción del artículo],
        Como UnitPrice [Marcado Precio],
        * @ UnitPrice Descuento / 100 Como [Descuento Amt],
        UnitPrice - (* @ UnitPrice Descuento / 100) En [Después de Descuento],
        UnitPrice * @ TaxRate / 100 Como [Impuesto Monto],
        (* @ UnitPrice TaxRate / 100) + UnitPrice --
        (* @ UnitPrice Descuento / 100) + (@ TaxRate / 100) En [precio neto]
DE StoreItems;
IR
```

Aquí hay un ejemplo de la ejecución del procedimiento:

CENTRAL.Depar...SQLQuery3.sql* | CENTRAL.Depar...SQLQuery2.sql*

```
USE DepartmentStore1;
GO
EXEC dbo.CalculateSalePrice;
GO
```

Results | Messages

	Item Description	Marked...	Discount ...	After Discount	Tax Amount	Net Price
1	Long-Sleeve Jersey ...	39.95	7.99000000	31.96000000	3.19600000	35.23600000
2	Solid Crewneck Tee	12.95	2.59000000	10.36000000	1.03600000	11.47600000
3	Cotton Comfort Open ...	17.85	3.57000000	14.28000000	1.42800000	15.78800000
4	Hooded Full-Zip Swe...	69.95	13.99000...	55.96000000	5.59600000	61.63600000
5	Plaid Pinpoint Dress ...	35.85	7.17000000	28.68000000	2.86800000	31.62800000
6	Cool-Dri® Soft Cup Bra	15.55	3.11000000	12.44000000	1.24400000	13.76400000
7	Ladies Hoody Sweats...	45.75	9.15000000	36.60000000	3.66000000	40.34000000
8	Cotton Knit Blazer	295.95	59.19000...	236.76000000	23.67600000	260.51600000
9	Denim Blazer - Natur...	75.85	15.17000...	60.68000000	6.06800000	66.82800000
10	Texture-Striped Pleat...	32.85	6.57000000	26.28000000	2.62800000	28.98800000
11	Iron-Free Pleated Kha...	39.95	7.99000000	31.96000000	3.19600000	35.23600000
12	Sunglasses	15.85	3.17000000	12.68000000	1.26800000	14.02800000

Q... | CENTRAL (9.0 SP2) | CENTRAL\Administrator (56) | DepartmentStore1 | 00:00:00 | 12 rows

Cuando llamas a un procedimiento almacenado que tiene más de un argumento y todos los argume valores por defecto, no es necesario para proporcionar un valor para cada argumento, puede proporc sólo uno o algunos de los argumentos. El procedimiento anterior puede ser llamada con un argumento c

EXEC CalculateSalePrice2 55,00

En este caso, el otro argumento (s) se utiliza su valor por defecto.

Hemos visto que, al llamar a un procedimiento almacenado que tiene más de un argumento, qu proporcionar los valores de los argumentos en el orden exacto en que apareció en el procedimient escribir el nombre de cada argumento y asignar que el valor deseado. De la misma manera, si u almacenado toma más de un argumento y algunos de los argumentos que tienen valores por defecto, puede proporcionar los valores en el orden de su elección, con el nombre de cada argumento y a deseado. Sobre esta base, el procedimiento almacenado anteriormente se puede llamar con sólo el v argumento de la siguiente manera:

CalculateSalePrice2 EXEC @ TaxRate = 8,55

En este caso, el primer argumento se utiliza su valor por defecto.

Aprendizaje práctico: El uso de argumentos por defecto

1. Elimine el contenido de la ventana de consulta
2. Para crear una nueva versión de un procedimiento almacenado se utilizó anteriormente, escriba lo ventana:

```
USO WattsALoan;
IR
DROP PROCEDURE SpecifyCurrentBalance;
IR
```

```
CREATE PROCEDURE SpecifyCurrentBalance
        @ PmtDate datetime,
        @ EmplID int,
        AL @ int,
        @ PmtAmt dinero,
        Comentarios @ Texto =''
COMO
COMENZAR
        - Obtener la cantidad que se presta al cliente
        DECLARE @ AmountOfLoan dinero;
        SET @ AmountOfLoan = (SELECT las.FutureValue
                                DE LoanAllocations las
                                WHERE (las.LoanAllocationID = @ AL));

        - Si el cliente ya ha hecho al menos un pago,
        - Obtener el saldo actual de la cuenta del cliente
        DECLARE @ CurrentBalance dinero;
        SET @ CurrentBalance = (SELECT MIN (pay.Balance)
                                  Pago de los pagos
                                  WHERE (pay.LoanAllocationID = @
AL));

        - Si el cliente nunca ha hecho un pago (todavía),
        - Especificar el saldo, resta el actual pago
        - A partir de la cantidad original del préstamo
        SI CurrentBalance @ IS NULL
        COMENZAR
                INSERT INTO Pagos (PaymentDate, IdEmpleado,
                                LoanAllocationID, PaymentAmount,
                                Equilibrio, Notas)
                VALUES (@ PmtDate, @ EmplID, AL @, @ PmtAmt,
                        AmountOfLoan @ - @ PmtAmt, @ Comentarios);
        FIN
        - Si el cliente ya había al menos un pago,
        - Restar el pago corriente de los últimos equilibrio
        ELSE
        COMENZAR
                INSERT INTO Pagos (PaymentDate, IdEmpleado,
                                LoanAllocationID, PaymentAmount,
                                Equilibrio, Notas)
                VALUES (@ PmtDate, EmplID @, @ definido,
                        @ PmtAmt, CurrentBalance @ - @ PmtAmt, @
Comentarios);
        FIN
FIN
IR
```

3. Para crear el procedimiento almacenado, presione F5
4. Elimine el contenido de la ventana de consulta
5. Para utilizar el procedimiento almacenado, escriba lo siguiente:

```
USO WattsALoan;
IR

EXECUTE SpecifyCurrentBalance'07 / 15/2004 ', 3, 1, 498.16,
   'El cliente envía un doble pago para cubrir el presente y el mes pasado »;
```

```
IR
EXECUTE SpecifyCurrentBalance'01 / 26/2007 ', 1, 3, 50;
IR
EXECUTE SpecifyCurrentBalance'08 / 26/2004 ', 2, 1, 249,08;
IR
EXECUTE SpecifyCurrentBalance'02 / 28/2006 ', 3, 5, 611,93;
IR
EXECUTE SpecifyCurrentBalance'10 / 24/2007 ', 2, 2, 415,25;
IR
EXECUTE SpecifyCurrentBalance'05 / 30/2004 ', 1, 1, 249,08;
IR
EXECUTE SpecifyCurrentBalance'02 / 22/2007 ', 2, 3, 20;
IR
EXECUTE SpecifyCurrentBalance'03 / 23/2006 ', 2, 5, 611,93;
IR
EXECUTE SpecifyCurrentBalance'07 / 22/2007 ', 2, 2, 415,25,
        "En primer lugar el pago regular";
IR
EXECUTE SpecifyCurrentBalance «12 / 24/2006 ', 1, 5, 611,93;
IR
EXECUTE SpecifyCurrentBalance'04 / 25/2006 ', 3, 5, 611,93;
IR
EXECUTE SpecifyCurrentBalance'09 / 26/2007 ', 2, 2, 415,25;
IR
EXECUTE SpecifyCurrentBalance'09 / 24/2006 ', 3, 4, 134,38;
IR
EXECUTE SpecifyCurrentBalance'03 / 25/2007 ', 2, 3, 25;
IR
EXECUTE SpecifyCurrentBalance «11 / 28/2006 ', 2, 5, 611,93,
 "En primer lugar de coches de pago ';
IR
EXECUTE SpecifyCurrentBalance'08 / 28/2007 ', 1, 2, 415,25,
 «Segundo pago»;
IR
```

6. Elimine el contenido de la ventana de consulta

Parámetros de salida

Transact-SQL utiliza el concepto de pasar un argumento por referencia. Este tipo de argument procedimiento, sino que pretende devolver un valor. En otras palabras, usted puede crear un procedim que toma un parámetro, pero el propósito del parámetro es para llevar un nuevo valor cuand procedimiento para que pueda utilizar ese valor de lo que precises.

Para crear un parámetro de que un valor de retorno del procedimiento almacenado, después procedimiento, si desea que el procedimiento almacenado para tener argumentos, escriba ellos. E omiten. Por otra parte, deberá pasar al menos un argumento, empezando por el nombre del símbolo tipo de datos, e introduzca la palabra clave **salida** a su derecha. Sobre esta base, la sintaxis básica se p

CREATE PROCEDURE *ProcedureName*
@ *ParameterName DataType* SALIDA
COMO
Cuerpo del procedimiento

En el cuerpo del procedimiento, puede realizar la cesión de lo que precises. La principal regla que de antes de que finalice el procedimiento, debe haber especificado un valor para el argumento **SALIDA.** E el argumento celebrará cuando el procedimiento almacenado salidas. Aquí está un ejemplo:

CREATE PROCEDURE dbo.CreateFullName

```
    @ FName varchar (20),
    @ Lname varchar (20),
    @ FullName varchar (42) SALIDA
COMO
    SELECT @ @ lname FullName = + ',' + @ FName
IR
```

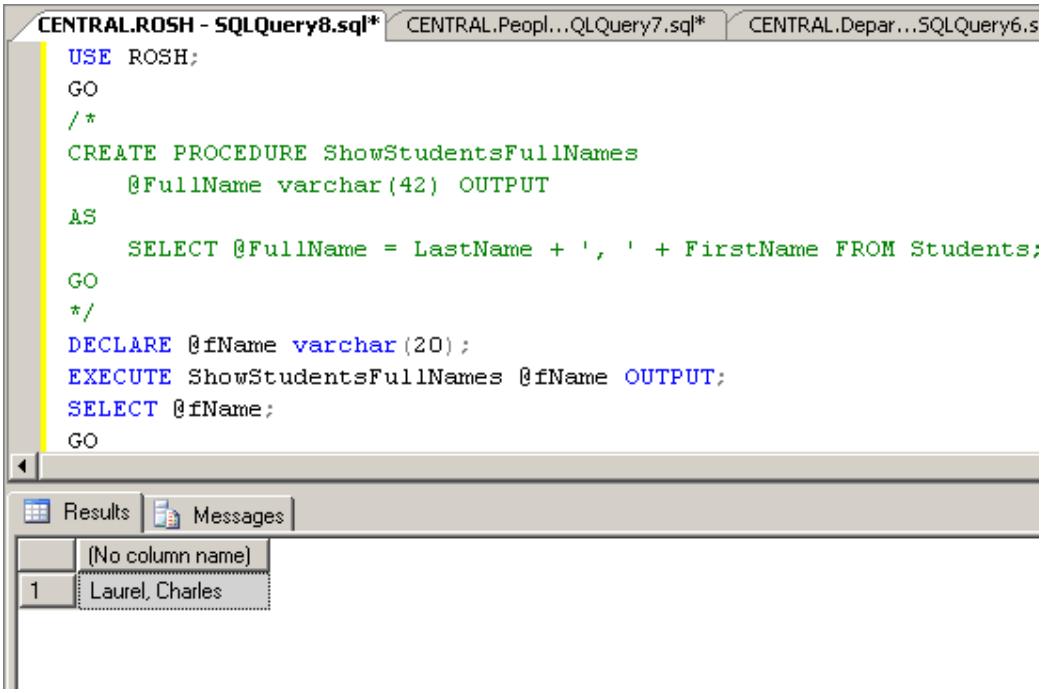
Al llamar al procedimiento almacenado, debe pasar un argumento para el parámetro de **salida** y, una que escribir **salida** en el lado derecho de la alegación. Recuerde que el procedimiento almacenado argumento. Esto significa que, después de llamar al procedimiento, puede volver el argumento de **s** según sus necesidades. Aquí está un ejemplo:

```
DECLARE @ Nombre varchar (20),
        @ Apellido varchar (20),
        @ Completo varchar (42)
SET @ Nombre = 'Melanie';
SET @ Apellido = 'Johanssen ";

EXECUTE dbo.CreateFullName @ Nombre, @ Apellidos, @ completo SALIDA

SELECT @ completo;
IR
```

Una de las ventajas de la utilización de una función o un procedimiento almacenado es que tiene acce registros de su base de datos. Esto significa que usted puede acceder a las columnas y los registros, s especifique la tabla o la vista, que se hace con una **cláusula** FROM asociado con un **SELECT.** Consi procedimiento almacenado creado en una base de datos que contiene una tabla denominada Estudiantes

```
Rosh USO;
IR

CREATE PROCEDURE ShowStudentsFullNames
    @ FullName varchar (42) SALIDA
COMO
    SELECT @ FullName = Apellidos + ',' + Nombre de los estudiantes;
IR
```

Al ejecutar este procedimiento almacenado, se trabajo sobre los registros de la tabla. Una de las parti procedimiento almacenado que tiene una **salida** argumento es que sólo puede devolver un valor. Cons ejemplo de la ejecución del procedimiento anterior:



Al llamar a este procedimiento, si no especifica una condición para producir un resultado determinado SQL, en este caso, seleccionar el último registro. Esto significa que siempre debe asegurarse de que almacenado que toma un parámetro de **salida** que tienen una forma de aislar a un resultado. Si almacenado procesa una **sentencia SELECT,** puede utilizar un **caso de** enfermedad. Aquí hay u procedimiento:

```
Rosh USO;
IR

CREATE PROCEDURE ShowStudentsFullNames
      @ FullName varchar (42) SALIDA
COMO
      SELECT @ FullName = Apellidos + ',' + Nombre de los estudiantes
      DÓNDE StudentID = 8;
IR
```

Cuando se ejecuta este procedimiento, se producen sólo el registro almacenado en la 8 ª posición de la

Resumen de la lección

Ejercicios

1. Crear un procedimiento almacenado llamado ProcessPayroll que tiene 11 argumentos:
    a. El número de horas trabajadas para la primera semana (pasado por valor)
    b. El número de horas trabajadas para la segunda semana (pasado por valor)
    c. Un número que representa el número de horas trabajadas para las dos semanas (pasado p
    d. Un número por el sueldo pagado por las horas regulares de las dos semanas (pasado por re
    e. Un número que representa el número de horas extraordinarias trabajadas para las dos sem referencia)
    f. Un número por el sueldo pagado por las horas extraordinarias de las dos semanas (pasado
    g. Un número que representa el sueldo neto que paga por nómina (pasado por referencia)

Fundamentos de los índices

Introducción

Si usted echa un vistazo a las últimas páginas de un libro de no ficción (como un libro sobre historia, economía, matemáticas, sociología, o estadísticas, etc), puede encontrar una serie de páginas que se inician en la sección Índice etiqueta. Las palabras en serie que le permite localizar una sección del libro que menciona, explica, o describe la palabra y temas relacionados. Un índice de un libro hace que sea fácil y rápido para llegar a una sección de un libro que se ocupa de un tema en particular.

Como un libro, un cuadro o una opinión puede utilizar el mecanismo previsto por un índice. En una tabla o un punto de vista, un índice es una columna (o varias columnas) que puede ser usado para localizar los registros y tomar una acción específica sobre la base de alguna regla reforzada en que (las) columna (s).

Aprendizaje práctico: La introducción de índices

1. Inicie Microsoft SQL Server y para abrir el acceso de Microsoft SQL Server Management Studio
2. Para crear una nueva base de datos, en el Explorador de objetos, haga clic derecho en las bases de datos y haga clic en Nueva base de datos ...
3. Defina el nombre de CollinsUniversity y haga clic en Aceptar
4. En el Explorador de objetos, expanda CollinsUniversity
5. Derecho-> haga clic en los cuadros y haga clic en Nuevo Cuadro ...
6. Haga clic en los campos de la siguiente manera:

Nombre de columna	Tipo de datos	Permitir valores NULL
StudentNumber	int	*Sin marcar*
Nombre	nvarchar (32)	*Comprobado*
MI	char (1)	*Comprobado*
Apellido	nvarchar (32)	*Sin marcar*
Nombre de usuario	nvarchar (20)	*Sin marcar*
Contraseña	nvarchar (20)	*C hecked*

7. Cierre la tabla
8. Cuando se le preguntó si desea guardar, haga clic en Sí.
9. Defina el nombre de StudentRegistration y haga clic en Aceptar
10. En el Explorador de objetos, haga clic derecho en las bases de datos y haga clic en Iniciar PowerShell
11. Para crear una nueva base de datos y una tabla, escriba lo siguiente (se presiona Enter al final de cada línea:

```
CREATE DATABASE CeilInn1;
IR
USO CeilInn1;
CREATE TABLE Habitaciones
(
    RoomNumber nvarchar (10),
    LocationCode NCHAR (10) no nulo,
    LocationName nvarchar (50) NULL,
```

```
        RoomType nvarchar (50)
    );
    IR
```

12. Volver a Microsoft SQL Server

Creación de un índice visual

El motor de base de datos ya está equipada con un mecanismo por defecto para crear automáticamente un índice sobre una mesa en función de cómo se crea la tabla. Por ejemplo, si crea una clave principal en una mesa, el motor de base de datos crea automáticamente un índice de la columna (s) utilizado como la clave principal. En caso contrario, puede crear un índice explícitamente. Puede crear un índice visual o utilizando código SQL.

Visualmente para crear un índice, puede utilizar la Índices / Claves cuadro de diálogo. Para mostrar que, en el Explorador de objetos, expanda la base de datos que contiene la tabla o vista y expandir el nodo Tablas. Haga clic derecho en la tabla o vista para la que desea crear el índice y haga clic en Diseño. En la ventana, haga clic derecho en Índices / Claves ... Esto abriría la Índices / Claves cuadro de diálogo:

Para crear un índice, haga clic en el botón Añadir. La primera pieza de información que debe proporcionar es el nombre. Normalmente, el motor de base de datos proporciona un nombre predeterminado. Si desea cambiarlo, haga clic en (nombre) y escriba el nombre deseado. La otra pieza muy importante de información que deben proporcionar al menos una columna. Por defecto, el motor de base de datos selecciona la primera columna de la tabla. Si esto no es la columna que desea utilizar, haga clic en Columnas y haga clic en el botón de puntos suspensivos . Esto abre el cuadro de diálogo Índice:

De las casillas en la columna Nombre, puede seleccionar cada columna. Una vez esté listo, haga clic en Aceptar.

Aprendizaje práctico: Creación de un índice Visualmente

1. En el Explorador de objetos, en las Tablas de CollinsUniversity nodo, haga clic derecho y haga clic en Diseño StudentRegistration
2. Clic con el botón derecho en cualquier lugar de la zona blanca y haga clic en Índices / Claves ...
3. Haga clic en Agregar
4. Haga clic en Columnas y haga clic en el botón de puntos suspensivos
5. Haga clic en el cuadro combinado de la StudentNumber por defecto y seleccione Apellido
6. Haga clic en el cuadro bajo Apellido
7. Haga clic en la flecha del cuadro combinado y seleccione Nombre de usuario
8. Haga clic en Aceptar
9. Haga clic en (nombre) y el tipo IX_Credentials
10. Haga clic en Cerrar

Con la creación de un Índice de SQL

Para crear un índice de SQL, la fórmula básica a seguir es:

CREATE INDEX SOBRE *IndexName Cuadro / Ver (Columna (s))*

Como alternativa, abra la ventana de consulta. A continuación, en el Explorador de plantillas, expanda el nodo Índice. Arrastre Crear Índice Base (u otra sub-nodo) y suéltelo en la ventana. Esqueleto de código se genera para usted:

```
- ============================================
- Crear plantilla básica
- ============================================
USO <database_name, sysname, AdventureWorks>
IR
```

```
CREATE INDEX <index_name, sysname, ind_test>
SOBRE <schema_name, sysname, Person>. <table_name, Sysname, Address>
(
        <column_name1, sysname, PostalCode>
),
IR
```

La creación de un índice comienza con la expresión **de CREATE INDEX.** Esta expresión es seguida por un nombre para el índice. El nombre es seguido por la palabra clave **SOBRE.** En el *Cuadro / Ver* marcador de posición, escriba el nombre de la tabla o vista para la que desea crear el índice. Esto es seguido por paréntesis. En el paréntesis, introduzca al menos una columna. Aquí está un ejemplo:

```
USO Exercise1;
IR
CREATE INDEX IX_Employees
A los empleados (EmployeeNumber);
IR
```

Si el índice se incluyen más de una columna, lístelas separadas por comas. Aquí está un ejemplo:

```
USO Exercise1;
IR
CREATE INDEX IX_Employees
A los empleados (Apellido, Nombre de usuario);
IR
```

Aprendizaje práctico: Creación de un índice con SQL

1. PowerShell en la ventana, para crear un índice de uso de Transact-SQL, escriba lo siguiente:

   ```
   USO CeilInn1;
   IR
   CREATE INDEX IX_Rooms
   SOBRE Habitaciones (RoomNumber, LocationCode);
   IR
   ```

2. Volver a Microsoft SQL Server Management Studio

Índice de mantenimiento

Introducción

En Microsoft SQL Server (y la mayoría de sistemas de bases de datos), un índice que se trata como un objeto. Es decir, un índice se puede comprobar o suprimido a voluntad.

Eliminar un índice

Si no necesita un índice de más, puede eliminarla. Puede hacer esto visualmente o manualmente.

Para borrar un índice visual, abra su tabla en la vista Diseño. Haga clic derecho en algún lugar de la tabla y haga clic en la ventana Índices / Vista. En el marco izquierdo, haga clic en el nombre del índice para seleccionarlo, y haga clic en el botón Eliminar. Usted no recibirá una advertencia. Por lo tanto, asegúrese de que desea eliminar el índice antes de comenzar esta operación.

La sintaxis básica para eliminar un índice de Transact-SQL es la siguiente:

DROP INDEX *IndexName* SOBRE *TableName;*

En esta fórmula, reemplace el *TableName* con el nombre de la tabla que contiene el índice. Sustitúyase el *IndexName* con el nombre del índice que desea eliminar.

Aquí está un ejemplo:

USO Exercise1;
IR
DROP INDEX IX_Students
Sobre los estudiantes;
IR

Los Tipos de índices

Introducción

Microsoft SQL Server es compatible con varios tipos de índices. Las dos categorías se agrupan más amplio y NONCLUSTERED.

Índices agrupados

En nuestra presentación, vimos que es ante todo un índice creado con una o más columnas de una tabla designado. Esto significa que, cuando se trata de utilizar el índice, que se utilizan los valores almacenados en la columna (s) que fue (fueron) seleccionado para el índice. Este índice se conoce como cluster. Las columnas que se han hecho parte de un índice se denominan llaves.

Para crear visualmente un índice agrupado, los índices de pantalla / Llaves cuadro de diálogo. En el cuadro de diálogo, al crear una nueva indexados o después de hacer clic en el nombre de un índice, en la lista de la derecha, haga clic en Crear Como clúster y seleccione Sí:

Una vez esté listo, haga clic en Cerrar.

Para crear un índice agrupado en SQL, utilice la siguiente fórmula:

CREAR *IndexName* índice agrupado en la *tabla / vista (columna (s))*

De la descripción que dio anteriormente, sólo la nueva palabra clave aquí es clúster. Basado en esto, aquí está un ejemplo:

```
USO Exercise1;
IR
CREATE TABLE dbo.Students
(
        Int StudentID identidad (1,1) NOT NULL,
        Nombre nvarchar (40) NULL,
        Apellido nvarchar (40) NOT NULL,
),
IR

CREAR índice agrupado IX_Student
SOBRE Estudiantes (Apellido);
IR
```

Una tabla que contiene un índice agrupado se llama un cluster de mesa.

Hay varios aspectos a un índice agrupado:

- Para facilitar la búsqueda de los registros, ellos (los registros) se ordenan. Esto hace posible que el motor de base de datos para proceder en un enfoque de arriba hacia abajo y rápidamente llegar a los registros
- Sin que esto sea un requisito, cada registro debe ser único (ya hemos visto cómo cuidar de ello, las limitaciones de uso de comprobar, más adelante, veremos de nuevo cómo crear registros únicos)
- Sólo debe haber un índice agrupado por tabla. Esto significa que, si (a decidir), crear un índice agrupado en una mesa, la mesa se convierte en equipado con uno. Si crea un índice agrupado, el anterior (un índice agrupado), se suprime

NONCLUSTERED Índice

Mientras que un índice agrupado utiliza una lista ordenada de los registros de una tabla o vista, otro tipo de índice, puede utilizar un mecanismo no basado en el los registros ordenados, pero en un marcador. Esto se llama una NONCLUSTERED índice. A diferencia de un cuadro en el que se agrupan sólo puede contener un índice agrupado, puede crearse no sólo uno, sino tantos como 249 NONCLUSTERED índices.

Para crear visualmente un NONCLUSTERED índice, los índices de pantalla / Llaves cuadro de diálogo. Para crear un nuevo índice, haga clic en el botón Añadir. Si un índice siempre fue creado o establecido como agrupado y desea cambiar, puede cambiar su Crear clúster Como propiedad de Sí a No.

Para crear un índice en SQL NONCLUSTERED, utilice la siguiente fórmula:

CREAR NONCLUSTERED ÍNDICE DE *IndexName Cuadro / Ver (Columna (s))*

La nueva palabra clave de esta fórmula es **NONCLUSTERED.** Todo es lo mismo que se ha descrito anteriormente. Basado en esto, aquí está un ejemplo:

```
CREAR NONCLUSTERED ÍNDICE IX_Student
SOBRE Estudiantes (Apellido, StudentNumber);
IR
```

Si crea un índice sin especificar **clúster** o **NONCLUSTERED,** el motor de base de datos se realiza automáticamente NONCLUSTERED.

De introducción de datos y análisis con los índices

Introducción

En nuestra presentación, vimos que un índice puede hacer que sea posible tomar algunas medidas en la entrada de datos, tales como la restricción de una columna que tiene valores únicos para cada registro o asegurarse de que la combinación de valores de un grupo de columnas en el mismo registro produce un valor único. Además de esta característica de los índices, que son realmente muy valioso cuando se trata de análisis de datos.

Como se ha mencionado de un libro, el principal objetivo de un índice para que sea fácil de localizar los registros de una tabla o vista.

Índices únicos

Un índice se hace de dos maneras valiosas. Por un lado, los registros deben ser ordenados. Un índice agrupado en sí se ocupa de este aspecto, ya que automáticamente su tipo y los registros. ¿Qué sucede si los registros no son exclusivos. Por ejemplo, en una mala entrada de datos en una lista de los estudiantes, es posible que tenga dos o más alumnos con el mismo los registros de los estudiantes. Si crea un índice para una tabla, el motor de base de datos podría crear registros duplicados en el índice. Normalmente, esto no es bueno porque a la hora de seleccionar los registros, es posible que tenga demasiados archivos y tomar una acción equivocada.

Al crear una tabla, puede que para el índice y que el índice de aplicación de una norma que establece que cada registro sería único. Para hacerse cargo de esto, puede aplicar una norma de carácter único en el índice.

Si usted es visualmente la creación de un índice, en los Índices / Llaves cuadro de diálogo, seleccione el índice en el lado izquierdo. En la lista de la derecha, el conjunto es único campo en Sí. Por otro lado, si quieres eliminar esta regla, configure el campo es único en No.

Para crear un índice en SQL singularidad, a la **UNICA** palabra clave en la fórmula:

CREATE [UNIQUE] [clúster | NONCLUSTERED] ÍNDICE DE INDEX_NAME *Cuadro / Ver (Columna (s))*

Comience con la palabra clave de CREATE. Siga por ÚNICO. A continuación, especifique si sería o no agrupados. El resto sigue las descripciones que vimos anteriormente. Aquí está un ejemplo:

```
- ==========================================
- Tabla: Empleados
- ==========================================
USO Exercise1
IR

CREATE TABLE dbo.Employees
(
      EmployeeNumber nvarchar (10),
      DateHired datetime NULL,
      Nombre nvarchar (40) NULL,
      Apellido nvarchar (40) NOT NULL,
      HourlySalary decimal (6, 2)
),
IR
```

```
Create unique IDX_EmployeesIDs ÍNDICE
SOBRE dbo.Employees (EmployeeNumber, LastName);
IR
```

Una vez que haya especificado la singularidad, en la entrada de datos, si el usuario introduce un valor que existe en la mesa ya, él o ella recibe un mensaje de error. Aquí está un ejemplo:

```
- ==============================================
- Crear plantilla básica
- ==============================================
USO Exercise1;
IR

CREATE TABLE dbo.Employees
(
	EmployeeNumber integer NOT NULL,
	Nombre varchar (50) NULL,
	Apellido varchar (50) NOT NULL,
	HourlySalary decimal (6, 2)
),
IR

Create unique IDX_EmployeeNumber ÍNDICE
SOBRE dbo.Employees (EmployeeNumber);
IR

INSERT INTO Empleados (EmployeeNumber, FirstName,
                       Apellido, HourlySalary)
VALORES (92935, 'Joan', 'Hamilton', 22,50)
IR

INSERT INTO Empleados (EmployeeNumber, FirstName,
                       Apellido, HourlySalary)
VALORES (22940, 'Pedro', 'Malley', 14,25)
IR

INSERT INTO Empleados (EmployeeNumber, FirstName,
                       Apellido, HourlySalary)
VALORES (27495, 'Christine', 'Fink', 32,05)
IR
INSERT INTO Empleados (EmployeeNumber, FirstName,
                       Apellido, HourlySalary)
VALORES (22940, 'Gertrude', 'Monay', 15,55)
IR
INSERT INTO Empleados (EmployeeNumber, FirstName,
                       Apellido, HourlySalary)
VALORES (20285, "Helene", "Mukoko ', 26,65)
IR
```

Esto produciría:

```
(1 row (s) affected)

(1 row (s) affected)

(1 row (s) affected)
Mensaje 2601, Nivel 14, Estado 1, Línea 1
```

No se puede insertar fila clave duplicada en objeto 'dbo.Employees'
con índice único 'IDX_EmployeeNumber ».
La declaración se haya dado por terminada.

(1 row (s) affected)