Implementación de Procedimientos Almacenados

Contenido

Notas para el instructor

Este módulo proporciona a los alumnos una explicación de cómo utilizar los procedimientos almacenados para mejorar el diseño y el rendimiento de las aplicaciones mediante la encapsulación de reglas de empresa. Se describen formas de procesar consultas comunes y modificaciones de datos.

El módulo comienza con la definición de los procedimientos almacenados y trata las ventajas de utilizarlos y la forma en que se procesan.

La siguiente sección describe cómo crear, ejecutar y modificar procedimientos almacenados. En la sección que sigue a continuación se describe cómo usar los procedimientos almacenados con parámetros de entrada y de salida. También se describen las opciones para volver a compilar.

Las secciones finales describen la ejecución de los procedimientos almacenados extendidos y la forma de controlar los mensajes de error, y tratan temas relativos al rendimiento que deben tenerse en cuenta a la hora de implementar procedimientos almacenados. A lo largo del módulo se proporcionan ejemplos y demostraciones de diversos procedimientos almacenados.

En la primera práctica los alumnos crean procedimientos almacenados basados en modelos proporcionados y utilizan el Administrador corporativo de Microsoft® SQL Server™ y el Analizador de consultas SQL para mostrar información acerca de los procedimientos almacenados. En la segunda práctica los alumnos utilizan un asistente para crear un procedimiento almacenado y generar una secuencia de comandos de procedimiento almacenado. Los alumnos crean procedimientos almacenados que aceptan información con parámetros de entrada y devuelven parámetros de salida.

Después de completar este módulo, los alumnos serán capaces de:

- Describir cómo se procesa un procedimiento almacenado.
- Crear, ejecutar, modificar y quitar un procedimiento almacenado.
- Crear procedimientos almacenados que acepten parámetros.
- Ejecutar procedimientos almacenados extendidos.
- Crear mensajes personalizados de error.

Introducción

Objetivo del tema
Proporcionar una introducción a los temas y objetivos del módulo.

Explicación previa
En este módulo aprenderá acerca de los procedimientos almacenados y por qué es conveniente utilizarlos.

- Introducción a los procedimientos almacenados
- Creación, ejecución, modificación y eliminación de procedimientos almacenados
- Utilización de parámetros en los procedimientos almacenados
- Ejecución de procedimientos almacenados extendidos
- Control de mensajes de error
- Consideraciones acerca del rendimiento

Objetivos

Después de completar este módulo, el alumno será capaz de:

- Describir cómo se procesa un procedimiento almacenado.
- Crear, ejecutar, modificar y eliminar un procedimiento almacenado.
- Crear procedimientos almacenados que acepten parámetros.
- Ejecutar procedimientos almacenados extendidos.
- Crear mensajes personalizados de error.

◆ Introducción a los procedimientos almacenados

Objetivo del tema
Presentar los temas de esta sección.

Explicación previa
En esta sección, trataremos…

- Definición de procedimientos almacenados
- Procesamiento inicial de los procedimientos almacenados
- Procesamientos posteriores de los procedimientos almacenados
- Ventajas de los procedimientos almacenados

Esta sección presenta los distintos tipos de procedimientos almacenados de Microsoft® SQL Server™ 2000, describe cómo se procesan, tanto inicialmente como en las ejecuciones siguientes, y enumera algunas de las ventajas de su utilización.

Definición de procedimientos almacenados

Objetivo del tema
Definir los procedimientos almacenados.

Explicación previa
Un procedimiento almacenado es una colección compilada previamente de instrucciones de Transac-SQL que se almacena en el servidor.

- Colecciones con nombre de instrucciones Transact-SQL
- Encapsulado de tareas repetitivas
- Admiten cinco tipos (del sistema, locales, temporales, remotos y extendidos)
- Aceptar parámetros de entrada y devolver valores
- Devolver valores de estado para indicar que se ha ejecutado satisfactoriamente o se ha producido algún error

Un procedimiento almacenado es una colección con nombre de instrucciones de Transact-SQL que se almacena en el servidor. Los procedimientos almacenados son un método para encapsular tareas repetitivas. Admiten variables declaradas por el usuario, ejecución condicional y otras características de programación muy eficaces.

SQL Server admite cinco tipos de procedimientos almacenados:

Procedimientos almacenados del sistema (sp_) Almacenados en la base de datos **master** e identificados mediante el prefijo sp_, los procedimientos almacenados del sistema proporcionan un método efectivo de recuperar información de las tablas del sistema. Permiten a los administradores del sistema realizar tareas de administración de la base de datos que actualizan las tablas del sistema aunque éstos no tengan permiso para actualizar las tablas subyacentes directamente. Los procedimientos almacenados del sistema se pueden ejecutar en cualquier base de datos.

Procedimientos almacenados locales Los procedimientos almacenados locales se crean en las bases de datos de los usuarios individuales.

Procedimientos almacenados temporales Los procedimientos almacenados temporales pueden ser locales, con nombres que comienzan por un signo de almohadilla (#), o globales, con nombres que comienzan por un signo de almohadilla doble (##). Los procedimientos almacenados temporales locales están disponibles en la sesión de un único usuario, mientras que los procedimientos almacenados temporales globales están disponibles para las sesiones de todos los usuarios.

Procedimientos almacenados remotos Los procedimientos almacenados remotos son una característica anterior de SQL Server. Las consultas distribuidas admiten ahora esta funcionalidad.

Para su información
Algunos procedimientos almacenados del sistema llaman a procedimientos almacenados extendidos.

Procedimientos almacenados extendidos (xp_) Los procedimientos almacenados extendidos se implementan como bibliotecas de vínculos dinámicos (DLL, *Dynamic-Link Libraries*) que se ejecutan fuera del entorno de SQL Server. Normalmente, se identifican mediante el prefijo xp_. Se ejecutan de forma similar a los procedimientos almacenados.

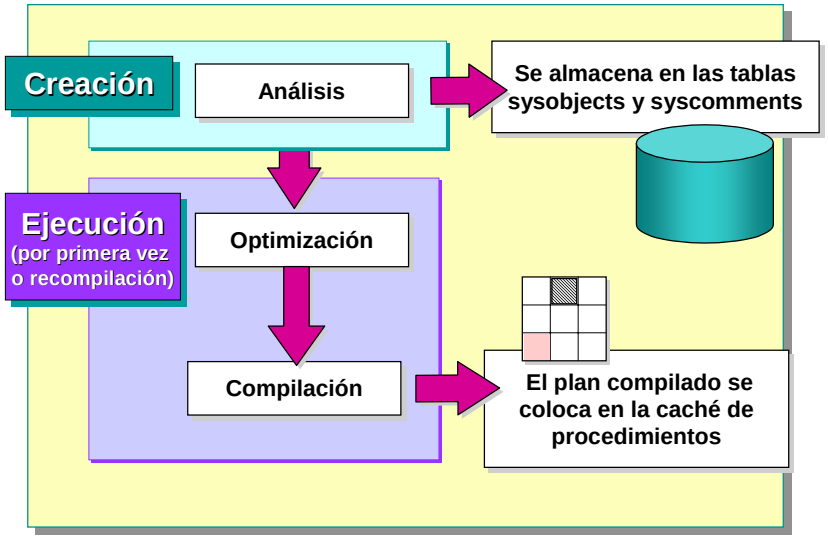
Los procedimientos almacenados en SQL Server son similares a los procedimientos de otros lenguajes de programación ya que pueden:

- Contener instrucciones que realizan operaciones en la base de datos; incluso tienen la capacidad de llamar a otros procedimientos almacenados.
- Aceptar parámetros de entrada.
- Devolver un valor de estado a un procedimiento almacenado o a un proceso por lotes que realiza la llamada para indicar que se ha ejecutado correctamente o que se ha producido algún error, y la razón del mismo.
- Devolver varios valores al procedimiento almacenado o al proceso por lotes que realiza la llamada en forma de parámetros de salida.

Procesamiento inicial de los procedimientos almacenados

Objetivo del tema
Mostrar los pasos para el procesamiento de un procedimiento almacenado.

Explicación previa
El procesamiento de un procedimiento almacenado incluye crearlo y ejecutarlo por primera vez, lo que coloca su plan de ejecución en la caché.



El procesamiento de un procedimiento almacenado conlleva crearlo y ejecutarlo la primera vez, lo que coloca su plan de consultas en la caché de procedimientos. La caché de procedimientos es un bloque de memoria que contiene los planes de ejecución de todas las instrucciones de Transact-SQL que se están ejecutando actualmente. El tamaño de la caché de procedimientos fluctúa dinámicamente de acuerdo con los grados de actividad. La caché de procedimientos se encuentra en el bloque de memoria que es la unidad principal de memoria de SQL Server. Contiene la mayor parte de las estructuras de datos que usan memoria en SQL Server.

Creación

Cuando se crea un procedimiento almacenado, las instrucciones que hay en él se analizan para ver si son correctas desde el punto de vista sintáctico. A continuación, SQL Server almacena el nombre del procedimiento almacenado en la tabla del sistema **sysobjects** y su texto en la tabla del sistema **syscomments** en la base de datos activa. Si se detecta un error de sintaxis, se devuelve un error y no se crea el procedimiento almacenado.

Sugerencia
Comente el proceso de resolución diferida de nombres.

Resolución diferida de nombres

Un proceso denominado resolución diferida de nombres permite a los procedimientos almacenados hacer referencia a objetos que no existen todavía cuando éste se crea. Este proceso ofrece flexibilidad porque los procedimientos almacenados y los objetos a los que hacen referencia no tienen que ser creados en ningún orden en particular. Los objetos deben existir en el momento en el que se ejecuta el procedimiento almacenado. La resolución diferida de nombres se lleva a cabo en el momento de ejecutar el procedimiento almacenado.

Ejecución (por primera vez o recompilación)

La primera vez que se ejecuta un procedimiento almacenado o si el procedimiento almacenado se debe volver a compilar, el procesador de consultas lo lee en un proceso llamado resolución.

Ciertos cambios en una base de datos pueden hacer que un plan de ejecución sea ineficaz o deje de ser válido. SQL Server detecta estos cambios y vuelve a compilarlo automáticamente cuando se produce alguna de las situaciones siguientes:

- Se realiza algún cambio estructural en una tabla o vista a la que hace referencia la consulta (ALTER TABLE y ALTER VIEW).
- Se generan nuevas estadísticas de distribución, bien de forma explícita a partir de una instrucción, como en UPDATE STATISTICS, o automáticamente.
- Se quita un índice usado por el plan de ejecución.
- Se realizan cambios importantes en las claves (la instrucción INSERT o DELETE) de una tabla a la que hace referencia una consulta.

Optimización

Cuando un procedimiento almacenado pasa correctamente la etapa de resolución, el optimizador de consultas de SQL Server analiza las instrucciones de Transact-SQL del procedimiento almacenado y crea un plan que contiene el método más rápido para obtener acceso a los datos. Para ello, el optimizador de consultas tiene en cuenta lo siguiente:

- La cantidad de datos de las tablas.
- La presencia y naturaleza de los índices de las tablas, y la distribución de los datos en las columnas indizadas.
- Los operadores de comparación y los valores de comparación que se usan en las condiciones de la cláusula WHERE.
- La presencia de combinaciones y las cláusulas UNION, GROUP BY u ORDER BY.

Compilación

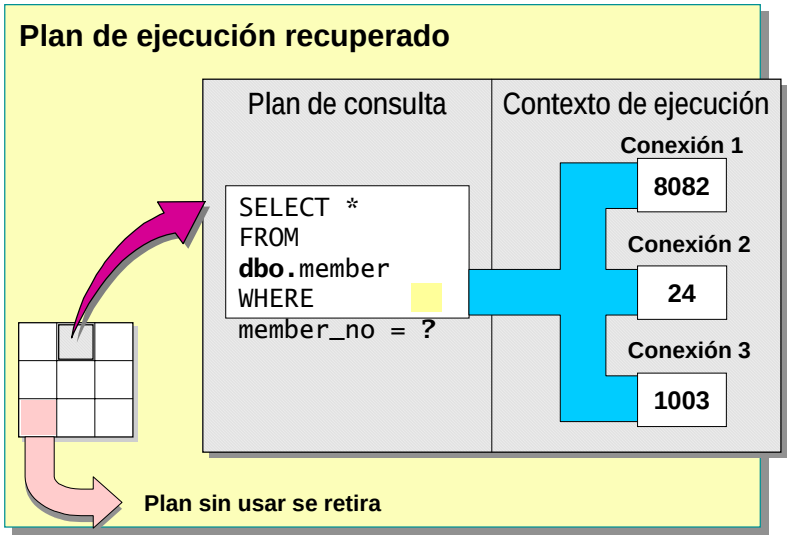
La compilación hace referencia al proceso consistente en analizar el procedimiento almacenado y crear un plan de ejecución que se encuentra en la caché de procedimientos. La caché de procedimientos contiene los planes de ejecución de los procedimientos almacenados más importantes. Entre los factores que aumentan el valor de un plan se incluyen los siguientes:

- Tiempo requerido para volver a compilar (costo de compilación alto)
- Frecuencia de uso

Procesamientos posteriores de los procedimientos almacenados

Objetivo del tema
Describir los procesos posteriores de los procedimientos almacenados.

Explicación previa
El proceso posterior de los procedimientos almacenados es más rápido que el proceso inicial porque SQL Server utiliza el plan de ejecución de la caché de procedimientos.



El proceso posterior de los procedimientos almacenados es más rápido que el inicial porque SQL Server utiliza el plan de ejecución optimizado de la caché de procedimientos.

Para la configuración del servidor, use sp_configure; para la configuración de la base de datos, use sp_dboption; para la configuración de la conexión, use las opciones de SET.

Si se dan las condiciones siguientes, SQL Server utiliza el plan que guarda en la memoria para ejecutar las consultas posteriores:

- El entorno actual es el mismo que el entorno en el que se compiló el plan.

  Las configuraciones del servidor, de la base de datos y de la conexión determinan el entorno.
- Los objetos a los que hace referencia el procedimiento almacenado no requieren que se lleve a cabo el proceso de resolución de nombres.

  Los objetos necesitan que se realice la resolución de nombres cuando hay objetos que pertenecen a distintos usuarios y tienen los mismos nombres. Por ejemplo, si la función **sales** es propietaria de una tabla **Product** y la función **development** es propietaria de otra tabla denominada **Product**, SQL Server debe determinar con qué tabla operar cada vez que se hace referencia a la tabla **Product**.

Los planes de ejecución de SQL Server tienen dos componentes principales:

- Plan de ejecución: la mayor parte del plan de ejecución se encuentra en esta estructura de datos reentrante y de sólo lectura que puede ser utilizada por un número cualquiera de usuarios.
- Contexto de ejecución: cada usuario que esté ejecutando actualmente la consulta tiene esta estructura de datos reutilizable que contiene los datos específicos de su ejecución, por ejemplo los valores de los parámetros. Si un usuario ejecuta una consulta y una de las estructuras no se está utilizando, ésta se reinicializa con el contexto del nuevo usuario.

Por tanto, en la caché siempre habrá, como máximo, un plan compilado para cada combinación exclusiva de procedimiento almacenado y entorno. Puede haber muchos planes para el mismo procedimiento almacenado si cada uno es para un entorno distinto.

Los factores siguientes dan como resultado distintos entornos que afectan a las opciones de compilación:

- Planes compilados en paralelo y planes compilados en serie
- Propiedad implícita de los objetos
- Distintas opciones de SET

Nota Para obtener más información acerca de los planes de ejecución paralelos, consulte el tema “Grado de paralelismo” en los Libros en pantalla de SQL Server.

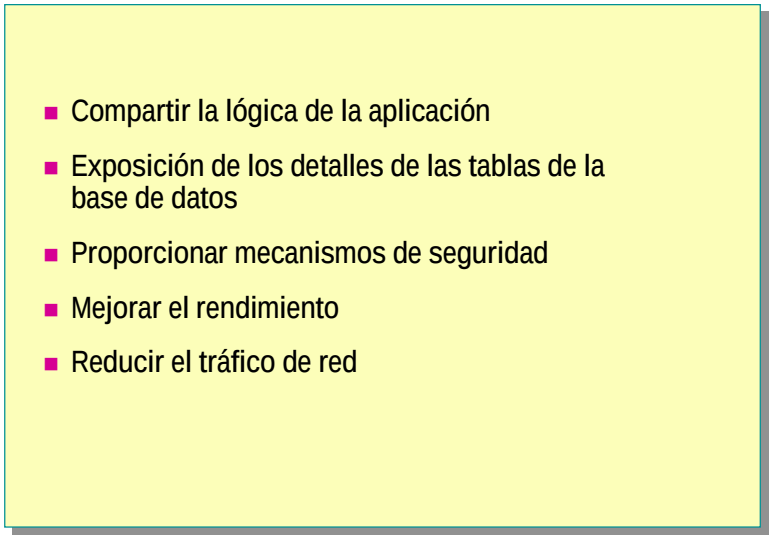
Los programadores deben elegir un entorno para sus aplicaciones y usarlo. Los objetos cuya resolución de propiedad implícita es ambigua deben usar la resolución explícita mediante la especificación del propietario del objeto. Las opciones de SET deben ser coherentes; deben establecerse al inicio de una conexión y no se deben cambiar.

Una vez generado un plan de ejecución, éste permanece en la caché de procedimientos. SQL Server sólo retira los planes antiguos y sin usar de la caché cuando necesita espacio.

Ventajas de los procedimientos almacenados

Objetivo del tema
Mostrar las ventajas de los procedimientos almacenados.

Explicación previa
Los procedimientos almacenados reducen significativamente los requisitos de ejecución en cuanto a recursos y tiempo.



Los procedimientos almacenados ofrecen varias ventajas. Pueden:

- Compartir la lógica de la aplicación con las restantes aplicaciones, lo que asegura que el acceso y la modificación de los datos se hace de una forma coherente.

  Los procedimientos almacenados pueden encapsular la funcionalidad del negocio. Las reglas o directivas empresariales encapsuladas en los procedimientos almacenados se pueden cambiar en una sola ubicación. Todos los clientes pueden usar los mismos procedimientos almacenados para asegurar que el acceso y modificación de los datos es coherente.
- Apartar a los usuarios de la exposición de los detalles de las tablas de la base de datos. Si un conjunto de procedimientos almacenados permite llevar a cabo todas las funciones de negocio que los usuarios necesitan, los usuarios no tienen que tener acceso a las tablas directamente.
- Proporcionar mecanismos de seguridad. Los usuarios pueden obtener permiso para ejecutar un procedimiento almacenado incluso si no tienen permiso de acceso a las tablas o vistas a las que hace referencia.
- Mejorar el rendimiento. Los procedimientos almacenados implementan muchas tareas como una serie de instrucciones de Transact-SQL. Se puede aplicar lógica condicional a los resultados de las primeras instrucciones de Transact-SQL para determinar cuáles son las siguientes que deben ejecutarse. Todas estas instrucciones de Transact-SQL y la lógica condicional pasa a ser parte de un único plan de ejecución del servidor.

El tráfico de red se reduce porque se necesitan menos paquetes para enviar solicitudes.

- Reducir el tráfico de red. En lugar de enviar cientos de instrucciones de Transact-SQL por la red, los usuarios pueden realizar una operación compleja mediante el envío de una única instrucción, lo que reduce el número de solicitudes que se pasan entre el cliente y el servidor.

Creación, ejecución, modificación y eliminación de procedimientos almacenados

Objetivo del tema
Presentar la creación, ejecución y modificación de procedimientos almacenados.

Explicación previa
Ahora que hemos definido los procedimientos almacenados y cómo se procesan, describiremos cómo crearlos, ejecutarlos y modificarlos.

- Creación de procedimientos almacenados
- Recomendaciones para la creación de procedimientos almacenados
- Ejecución de procedimientos almacenados
- Modificación y eliminación de procedimientos almacenados

Esta sección describe cómo crear, ejecutar, modificar y eliminar procedimientos almacenados.

Creación de procedimientos almacenados

Objetivo del tema
Presentar la sintaxis de CREATE PROCEDURE.

Explicación previa
Utilice la instrucción CREATE PROCEDURE para crear procedimientos almacenados en la base de datos activa.

- Utilice la instrucción CREATE PROCEDURE para crearlos en la base de datos activa

```
USE Northwind
GO
CREATE PROC dbo.OverdueOrders
AS
  SELECT *
   FROM dbo.Orders
   WHERE RequiredDate < GETDATE() AND ShippedDate IS Null
GO
```

- Puede anidar hasta 32 niveles
- Use sp_help para mostrar información

La sintaxis no permite especificar el nombre de la base de datos como prefijo del nombre del objeto.

Sólo se puede crear un procedimiento almacenado en la base de datos activa, excepto en el caso de los procedimientos almacenados temporales, que se crean siempre en la base de datos **tempdb**. La creación de un procedimiento almacenado es similar a la creación de una vista. Primero, escriba y pruebe las instrucciones de Transact-SQL que desea incluir en el procedimiento almacenado. A continuación, si recibe los resultados esperados, cree el procedimiento almacenado.

Uso de CREATE PROCEDURE

Los procedimientos almacenados se crean con la instrucción CREATE PROCEDURE. Considere los hechos siguientes cuando cree procedimientos almacenados:

- Los procedimientos almacenados pueden hacer referencia a tablas, vistas, funciones definidas por el usuario y otros procedimientos almacenados, así como a tablas temporales.
- Si un procedimiento almacenado crea una tabla local temporal, la tabla temporal sólo existe para atender al procedimiento almacenado y desaparece cuando finaliza la ejecución del mismo.
- Una instrucción CREATE PROCEDURE no se puede combinar con otras instrucciones de Transact-SQL en un solo proceso por lotes.
- La definición de CREATE PROCEDURE puede incluir cualquier número y tipo de instrucciones de Transact-SQL, con la excepción de las siguientes instrucciones de creación de objetos: CREATE DEFAULT, CREATE PROCEDURE, CREATE RULE, CREATE TRIGGER y CREATE VIEW. En un procedimiento almacenado se pueden crear otros objetos de la base de datos y deben calificarse con el nombre del propietario del objeto.

- Para ejecutar la instrucción CREATE PROCEDURE, debe ser miembro de la función de administradores del sistema (**sysadmin**), de la función de propietario de la base de datos (**db_owner**) o de la función de administrador del lenguaje de definición de datos (**db_ddladmin**), o debe haber recibido el permiso CREATE PROCEDURE.
- El tamaño máximo de un procedimiento almacenado es 128 megabytes (MB), según la memoria disponible.

Sintaxis parcial

```
CREATE PROC [ EDURE ] nombreProcedimiento [ ; número ]
    [ { @tipoDatos procedimiento }
        [ VARYING ] [ = predeterminado ] [ OUTPUT ]
    ] [ ,...n ]
  [ WITH
    { RECOMPILE | ENCRYPTION | RECOMPILE , ENCRYPTION } ]
  [ FOR REPLICATION ]
  AS instrucciónSql [ ...n ]
```

Ejemplo

Las instrucciones siguientes crean un procedimiento almacenado que enumera todos los pedidos atrasados de la base de datos **Northwind**.

```
USE Northwind
GO
CREATE PROC dbo.OverdueOrders
AS
  SELECT *
   FROM dbo.Orders
   WHERE RequiredDate < GETDATE() AND ShippedDate IS Null
GO
```

Anidamiento de procedimientos almacenados

Los procedimientos almacenados pueden anidarse, es decir, un procedimiento almacenado puede llamar a otro. Entre las características del anidamiento de procedimientos almacenados se incluyen las siguientes:

- Los procedimientos almacenados se pueden anidar hasta 32 niveles. Intentar superar 32 niveles de anidamiento hace que falle la llamada a la cadena completa de procedimientos almacenados.
- El nivel actual de anidamiento se almacena en la función del sistema **@@nestlevel**.
- Si un procedimiento almacenado llama a otro, éste puede obtener acceso a todos los objetos que cree el primero, incluidas las tablas temporales.
- Los procedimientos almacenados anidados pueden ser recursivos. Por ejemplo, el procedimiento almacenado X puede llamar al procedimiento almacenado Y. Al ejecutar el procedimiento almacenado Y, éste puede llamar al procedimiento almacenado X.

Ver información acerca de los procedimientos almacenados

Como con el resto de los objetos de base de datos, los procedimientos almacenados del sistema siguientes se pueden utilizar para buscar información adicional acerca de todos los tipos de procedimientos almacenados: **sp_help**, **sp_helptext** y **sp_depends**. Para imprimir una lista de procedimientos almacenados y nombres de propietarios de la base de datos, use el procedimiento almacenado del sistema **sp_stored_procedures**. También puede consultar las tablas del sistema **sysobjects**, **syscomments** y **sysdepends** para obtener información.

Recomendaciones para la creación de procedimientos almacenados

Objetivo del tema
Describir recomendaciones para la creación de procedimientos almacenados.

Explicación previa
Tenga en cuenta estas recomendaciones cuando cree procedimientos almacenados.

- El usuario dbo debe ser el propietario de todos los procedimientos almacenados
- Un procedimiento almacenado por tarea
- Crear, probar y solucionar problemas
- Evite sp_Prefix en los nombres de procedimientos almacenados
- Utilice la misma configuración de conexión para todos los procedimientos almacenados
- Reduzca al mínimo la utilización de procedimientos almacenados temporales
- No elimine nunca directamente las entradas de Syscomments

Considere las recomendaciones siguientes cuando cree procedimientos almacenados:

- Para evitar situaciones en las que el propietario de un procedimiento almacenado y el propietario de las tablas subyacentes sean distintos, se recomienda que el usuario **dbo** (propietario de base de datos) sea el propietario de todos los objetos de una base de datos. Como un usuario puede ser miembro de varias funciones, debe especificar siempre el usuario **dbo** como propietario al crear el objeto. En caso contrario, el objeto se creará con su nombre de usuario como propietario:
  - También debe tener los permisos adecuados en todas las tablas o vistas a las que se hace referencia en el procedimiento almacenado.
  - Evite situaciones en las que el propietario de un procedimiento almacenado y el propietario de las tablas subyacentes sean distintos.

Nota Si está creando un procedimiento almacenado del sistema definido por el usuario, debe haber iniciado una sesión como miembro de la función de administradores del sistema (**sysadmin**) y usar la base de datos **master**.

- Diseñe cada procedimiento almacenado para realizar una única tarea.
- Cree, pruebe y solucione los problemas del procedimiento almacenado en el servidor; a continuación, pruébelo desde el cliente.
- Para distinguir fácilmente los procedimientos almacenados del sistema, evite utilizar el prefijo **sp_** cuando nombre los procedimientos almacenados locales.

Sugerencia
Sugiera a los alumnos que primero utilicen osql para probar el funcionamiento del procedimiento almacenado debido a la reducida sobrecarga de trabajo de esta aplicación.

- Todos los procedimientos almacenados deben utilizar la misma configuración de conexiones.

  SQL Server guarda la configuración de SET QUOTED_IDENTIFIER y SET ANSI_NULLS cuando se crea o se modifica un procedimiento almacenado. Esta configuración original se usa cuando se ejecuta el procedimiento almacenado. Por tanto, cualquier configuración de la sesión del cliente para estas opciones de SET se pasa por alto durante la ejecución del procedimiento almacenado.

  Otras opciones de SET, como SET ARITHABORT, SET ANSI_WARNINGS y SET ANSI_PADDINGS, no se guardan cuando se crea o se modifica un procedimiento almacenado.

  Para determinar si las opciones de ANSI SET estaban habilitadas cuando se creó un procedimiento almacenado, consulte la función del sistema OBJECTPROPERTY. Las opciones de SET no deben cambiarse durante la ejecución de los procedimientos almacenados.
- Reduzca al mínimo la utilización de procedimientos almacenados temporales para evitar la competencia por las tablas del sistema en **tempdb**, que puede afectar al rendimiento desfavorablemente.
- Utilice **sp_executesql** en lugar de la instrucción EXECUTE para ejecutar dinámicamente una cadena en un procedimiento almacenado. El procedimiento **sp_executesql** es más eficaz porque genera planes de ejecución que SQL Server suele volver a utilizar. SQL Server compila las instrucciones de Transact-SQL de la cadena en un plan de ejecución independiente del plan del procedimiento almacenado. Puede utilizar **sp_executesql** cuando ejecute una instrucción de Transact-SQL en varias ocasiones si la única variación está en los valores de los parámetros suministrados a la instrucción de Transact-SQL.
- No elimine nunca directamente las entradas de la tabla del sistema **syscomments**. Si no desea que los usuarios puedan ver el texto de los procedimientos almacenados, debe crearlos usando la opción WITH ENCRYPTION. Si no utiliza WITH ENCRYPTION, los usuarios pueden usar el Administrador corporativo de SQL Server o ejecutar el procedimiento almacenado del sistema **sp_helptext** para ver el texto de los procedimientos almacenados que se encuentran en la tabla del sistema **syscomments**.

Ejecución de procedimientos almacenados

Objetivo del tema
Describir cómo se ejecuta un procedimiento almacenado.

Explicación previa
Puede ejecutar un procedimiento almacenado por sí mismo o como parte de una instrucción INSERT.

- Ejecución de un procedimiento almacenado por separado

```
EXEC OverdueOrders
```

- Ejecución de un procedimiento almacenado en una instrucción INSERT

```
INSERT INTO Customers
EXEC EmployeeCustomer
```

Sugerencia
Se recomienda que enseñe a los alumnos a usar la instrucción EXECUTE con el fin de ejecutar procedimientos almacenados uno por uno para ayudarles a evitar errores al ejecutarlos en procesos por lotes.

Puede ejecutar un procedimiento almacenado por sí mismo o como parte de una instrucción INSERT. Debe disponer del permiso EXECUTE en el procedimiento almacenado.

Ejecución de un procedimiento almacenado por separado

Para ejecutar un procedimiento almacenado puede emitir la instrucción EXECUTE junto con el nombre del procedimiento almacenado y de los parámetros.

Sintaxis

```
[ [ EXEC [ UTE ] ]
    {
        [@estadoDevuelto =]
            { nombreProcedimiento [;número] | @ nombreProcedimientoVar
    }
    [ [ @parámetro = ] { valor | @variable [ OUTPUT ] | [ DEFAULT ] ]
        [ ,...n ]
  [ WITH RECOMPILE ]
```

Ejemplo 1

La instrucción siguiente crea un procedimiento almacenado que enumera todos los pedidos atrasados de la base de datos **Northwind**.

```
EXEC OverdueOrders
```

Los procedimientos almacenados que se ejecutan en una instrucción INSERT deben devolver un conjunto de resultados relacional. Por ejemplo, no podría usar una instrucción COMPUTE BY.

Ejecución de un procedimiento almacenado en una instrucción INSERT

La instrucción INSERT puede rellenar una tabla local con un conjunto de resultados devuelto de un procedimiento almacenado local o remoto. SQL Server carga en el procedimiento almacenado la tabla con los datos que se devuelven de las instrucciones SELECT. La tabla debe existir previamente y los tipos de datos deben coincidir.

Ejemplo 2

La instrucción siguiente crea el procedimiento almacenado **EmployeeCustomer**, que inserta empleados en la tabla **Customers** de la base de datos **Northwind**.

```
USE Northwind
GO
CREATE PROC dbo.EmployeeCustomer
AS
SELECT
   UPPER(SUBSTRING(LastName, 1, 4)+SUBSTRING(FirstName, 1,1)),
   'Northwind Traders', RTRIM(FirstName)+' '+LastName,
   'Employee', Address, City, Region, PostalCode, Country,
   ('(206) 555-1234'+' x'+Extension), NULL
FROM Employees
WHERE HireDate < GETDATE ()
GO
```

Las instrucciones siguientes ejecutan el procedimiento almacenado **EmployeeCustomer**.

```
INSERT INTO Customers
EXEC EmployeeCustomer
```

El número de empleados contratados antes de la fecha de hoy se agrega a la tabla **Customers**.

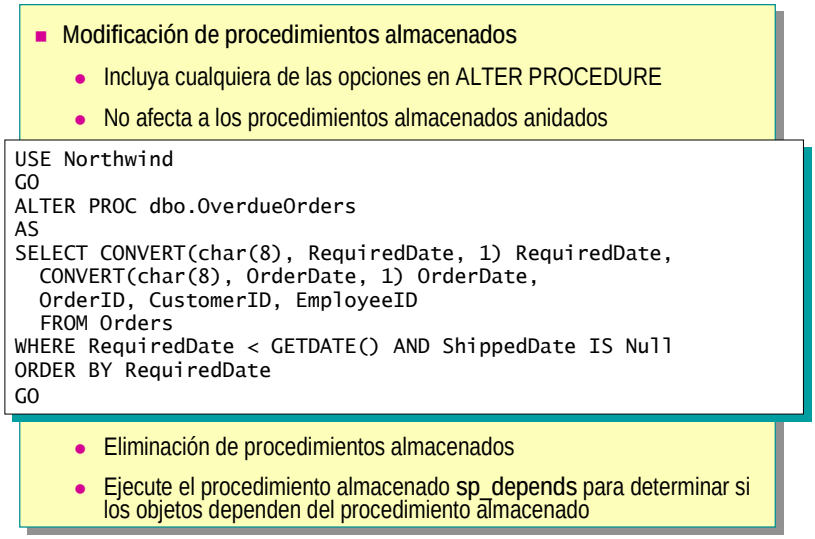
Resultado

```
(9 filas afectadas)
```

Modificación y eliminación de procedimientos almacenados

Objetivo del tema
Presentar la instrucción ALTER PROCEDURE.

Explicación previa
A menudo, los procedimientos almacenados se modifican en respuesta a solicitudes de los usuarios o a cambios en la definición de las tablas subyacentes.



A menudo, los procedimientos almacenados se modifican en respuesta a solicitudes de los usuarios o a cambios en la definición de las tablas subyacentes.

Modificación de procedimientos almacenados

Para modificar un procedimiento almacenado existente y conservar la asignación de los permisos, use la instrucción ALTER PROCEDURE. SQL Server sustituye la definición anterior del procedimiento almacenado cuando se modifica con ALTER PROCEDURE.

Precaución Se recomienda encarecidamente que no modifique de forma directa los procedimientos almacenados del sistema. En su lugar, copie las instrucciones desde un procedimiento almacenado del sistema existente para crear un procedimiento almacenado del sistema definido por el usuario y, a continuación, modifíquelo para adaptarlo a sus necesidades.

Cuando use la instrucción ALTER PROCEDURE, tenga en cuenta los hechos siguientes:

- Si desea modificar un procedimiento almacenado que se creó con opciones, como con la opción WITH ENCRYPTION, debe incluir la opción en la instrucción ALTER PROCEDURE para conservar la funcionalidad que proporciona la opción.
- ALTER PROCEDURE sólo altera un procedimiento. Si el procedimiento llama a otros procedimientos almacenados, los procedimientos almacenados anidados no se ven afectados.
- El permiso para ejecutar esta instrucción se concede de forma predeterminada a los creadores del procedimiento almacenado inicial, a los miembros de la función de servidor **sysadmin** y a los miembros de las funciones fijas de base de datos **db_owner** y **db_ddladmin**. No se pueden conceder permisos para ejecutar ALTER PROCEDURE.

Sintaxis

```
ALTER PROC [ EDURE ] nombreProcedimiento [ ; número ]
    [ { @tipoDatos parámetro }
        [ VARYING ] [ = valorPredeterminado ] [ OUTPUT ]
    ] [ ,...n ]
  [ WITH
    { RECOMPILE | ENCRYPTION
        | RECOMPILE , ENCRYPTION
    }
  ]
  [ FOR REPLICATION ]
  AS
    instrucciónSQL [...n]
```

Ejemplo

En el ejemplo siguiente se modifica el procedimiento almacenado **OverdueOrders** para seleccionar sólo los nombres de determinadas columnas en lugar de todas las columnas de la tabla **Orders** y para ordenar el conjunto de resultados.

```
USE Northwind
GO
ALTER PROC dbo.OverdueOrders
AS
SELECT CONVERT(char(8), RequiredDate, 1) RequiredDate,
  CONVERT(char(8), OrderDate, 1) OrderDate,
  OrderID, CustomerID, EmployeeID
  FROM Orders
WHERE RequiredDate < GETDATE() AND ShippedDate IS Null
ORDER BY RequiredDate
GO
```

La instrucción siguiente ejecuta el procedimiento almacenado **OverdueOrders**.

```
EXEC OverdueOrders
```

Resultado

Si el procedimiento almacenado **OverdueOrders** se ejecuta en función de la fecha de hoy, el conjunto de resultados se parecerá a lo siguiente.

RequiredDate	OrderDate	OrderID	CustomerID	EmployeeID
05/06/98	04/08/98	11008	ERNSH	7
05/11/98	04/13/98	11019	RANCH	6
05/19/98	04/21/98	11039	LINOD	1
05/21/98	04/22/98	11040	GREAL	4
05/25/98	04/23/98	11045	BOTTM	6

.
.
.
(21 filas afectadas)

Eliminación de procedimientos almacenados

Use la instrucción DROP PROCEDURE para quitar procedimientos almacenados definidos por el usuario de la base de datos actual.

Antes de quitar un procedimiento almacenado, ejecute el procedimiento almacenado **sp_depends** para determinar si los objetos dependen de él.

Sintaxis

DROP PROCEDURE { *procedimiento* } [,...n]

Ejemplo

En este ejemplo se elimina el procedimiento almacenado **OverdueOrders**.

```
USE Northwind
GO
DROP PROC dbo.OverdueOrders
GO
```

Utilización de parámetros en los procedimientos almacenados

Objetivo del tema
Presentar los temas de esta sección.

Explicación previa
En esta sección, trataremos…

- Utilización de parámetros de entrada
- Ejecución de procedimientos almacenados con parámetros de entrada
- Devolución de valores mediante parámetros de salida
- Volver a compilar explícitamente procedimientos almacenados

Los parámetros amplían la funcionalidad de los procedimientos almacenados. Mediante parámetros, puede pasar información hacia dentro y hacia fuera en los procedimientos almacenados. Permiten utilizar el mismo procedimiento almacenado para buscar en una base de datos muchas veces.

Por ejemplo, puede agregar un parámetro a un procedimiento almacenado para buscar en la tabla **Employee** empleados cuyas fechas de contratación coincidan con la fecha que especifique. A continuación, puede ejecutar el procedimiento almacenado cada vez que desee especificar una fecha de contratación distinta.

SQL Server admite dos tipos de parámetros: parámetros de entrada y parámetros de salida.

Utilización de parámetros de entrada

Objetivo del tema
Presentar los parámetros de entrada.

Explicación previa
Los parámetros de entrada permiten que se pase información a un procedimiento almacenado.

- Valide primero todos los valores de los parámetros de entrada
- Proporcione los valores predeterminados apropiados e incluya las comprobaciones de Null

```
CREATE PROCEDURE dbo.[Year to Year Sales]
  @BeginningDate DateTime, @EndingDate DateTime
AS
IF @BeginningDate IS NULL OR @EndingDate IS NULL
BEGIN
   RAISERROR('NULL values are not allowed', 14, 1)
   RETURN
END
SELECT O.ShippedDate,
       O.OrderID,
       OS.Subtotal,
       DATENAME(yy,ShippedDate) AS Year
FROM ORDERS O INNER JOIN [Order Subtotals] OS
   ON O.OrderID = OS.OrderID
WHERE O.ShippedDate BETWEEN @BeginningDate AND @EndingDate
GO
```

Los parámetros de entrada permiten que se pase información a un procedimiento almacenado. Para definir un procedimiento almacenado que acepte parámetros de entrada, declare una o varias variables como parámetros en la instrucción CREATE PROCEDURE.

Sintaxis parcial

@parámetro tipoDato [= *valorPredeterminado*]

Cuando especifique los parámetros, tenga en cuenta los hechos e instrucciones siguientes:

- Todos los valores de los parámetros de entrada deben ser comprobados al principio de un procedimiento almacenado para conocer rápidamente los valores que no sean válidos o que falten.
- Deben suministrarse valores predeterminados apropiados para un parámetro.

  Si se define un valor predeterminado, un usuario puede ejecutar el procedimiento almacenado sin especificar un valor para el parámetro.

  Nota Los parámetros predeterminados deben ser constantes o NULL. Cuando especifique NULL como valor predeterminado de un parámetro, debe usar =Null; IS NULL no funcionará porque la sintaxis no admite la designación de valores NULL ANSI.

- El número máximo de parámetros en un procedimiento almacenado es 1024.
- El número máximo de variables locales en un procedimiento almacenado sólo está limitado por la memoria disponible.
- Los parámetros son locales a un procedimiento almacenado. Los mismos nombres de parámetro se pueden usar en otros procedimientos almacenados.

La información de los parámetros se almacena en la tabla del sistema **syscolumns**.

Ejemplo

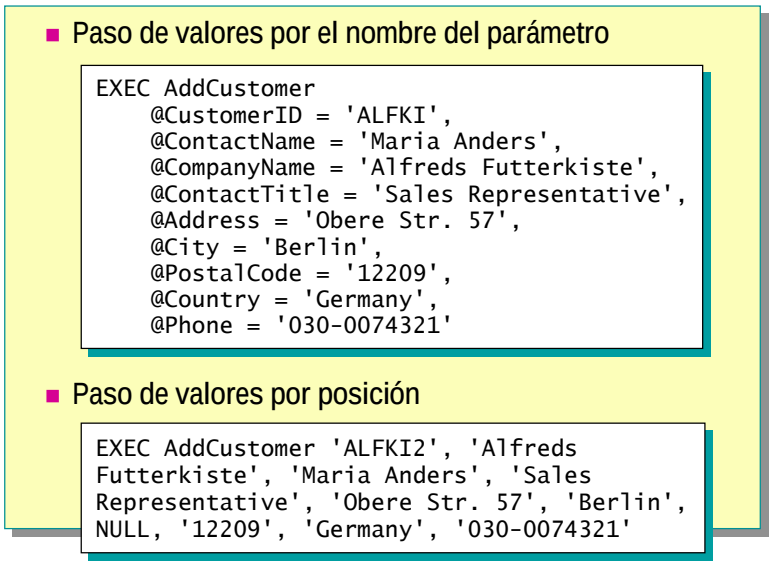
El siguiente ejemplo crea el procedimiento almacenado **Year to Year Sales**, que devuelve todas las ventas en un intervalo de fechas determinado.

```
CREATE PROCEDURE dbo.[Year to Year Sales]
  @BeginningDate DateTime, @EndingDate DateTime
AS
IF @BeginningDate IS NULL OR @EndingDate IS NULL
BEGIN
   RAISERROR('NULL values are not allowed', 14, 1)
   RETURN
END
SELECT O.ShippedDate,
       O.OrderID,
       OS.Subtotal,
       DATENAME(yy,ShippedDate) AS Year
FROM ORDERS O INNER JOIN [Order Subtotals] OS
   ON O.OrderID = OS.OrderID
WHERE O.ShippedDate BETWEEN @BeginningDate AND @EndingDate
GO
```

Ejecución de procedimientos almacenados con parámetros de entrada

Objetivo del tema
Mostrar cómo se ejecutan procedimientos almacenados mediante la instrucción EXECUTE con parámetros.

Explicación previa
Puede pasar valores de parámetros a un procedimiento almacenado por referencia o por posición.



Los valores de un parámetro se pueden establecer mediante el paso del valor al procedimiento almacenado mediante el nombre del parámetro o por su posición. No debe mezclar los distintos formatos cuando suministre valores.

Sugerencia
Para mejorar la legibilidad, se recomienda pasar los valores por nombre del parámetro.

Paso de valores por el nombre del parámetro

La especificación de un parámetro en una instrucción EXECUTE con el formato *@parámetro = valor* se conoce como *paso por nombre de parámetro*. Cuando se pasan valores por el nombre del parámetro, los valores de los parámetros se pueden especificar en cualquier orden y se puede omitir los que permitan valores nulos o tengan un valor predeterminado.

El valor predeterminado de un parámetro, si se ha definido para el parámetro en el procedimiento almacenado, se usa cuando:

- No se ha especificado ningún valor cuando se ejecuta el procedimiento almacenado.
- Se especifica la palabra clave DEFAULT como el valor del parámetro.

Sintaxis

```
[ [ EXEC [ UTE ] ]
    {
        [@estadoDevuelto =]
            { nombreProcedimiento [;número] | @ nombreProcedimientoVar
    }
    [ [ @parámetro = ] { valor | @variable [ OUTPUT ] | [ DEFAULT ] ]
        [ ,...n ]
  [ WITH RECOMPILE ]
```

Ejemplo parcial 1

El ejemplo parcial siguiente crea el procedimiento almacenado **AddCustomer**, que agrega un cliente nuevo a la base de datos **Northwind**. Observe que todas las variables excepto **CustomerID** y **CompanyName** se especifican para permitir un valor nulo.

```
USE Northwind
GO
CREATE PROCEDURE dbo.AddCustomer
    @CustomerID      nchar (5),
    @CompanyName     nvarchar (40),
    @ContactName     nvarchar (30) = NULL,
    @ContactTitle    nvarchar (30) = NULL,
    @Address         nvarchar (60) = NULL,
    @City            nvarchar (15) = NULL,
    @Region          nvarchar (15) = NULL,
    @PostalCode      nvarchar (10) = NULL,
    @Country         nvarchar (15) = NULL,
    @Phone           nvarchar (24) = NULL,
    @Fax             nvarchar (24) = NULL
    AS
.
.
.
```

Ejemplo parcial 2

El ejemplo siguiente pasa valores por el nombre del parámetro al procedimiento almacenado **AddCustomer**. Observe que el orden de los valores es distinto que en la instrucción CREATE PROCEDURE.

Observe también que no se especifican los valores de los parámetros **@Region** y **@Fax**. Si las columnas Region y Fax de la tabla permiten valores nulos, el procedimiento almacenado **AddCustomer** se ejecutará correctamente. Sin embargo, si no permiten valores nulos, deberá pasar un valor a un parámetro, con independencia de si ha definido el parámetro para permitir un valor nulo.

```
EXEC AddCustomer
    @CustomerID = 'ALFKI',
    @ContactName = 'Maria Anders',
    @CompanyName = 'Alfreds Futterkiste',
    @ContactTitle = 'Sales Representative',
    @Address = 'Obere Str. 57',
    @City = 'Berlin',
    @PostalCode = '12209',
    @Country = 'Germany',
    @Phone = '030-0074321'
.
.
.
```

Paso de valores por posición

El paso de valores únicamente, sin hacer referencia a los parámetros a los que se pasan, se conoce como *paso de valores por posición*. Cuando sólo se especifica un valor, los valores de los parámetros deben enumerarse en el orden en el que se han definido en la instrucción CREATE PROCEDURE.

Cuando se pasan valores por posición, puede omitir los parámetros donde haya valores predeterminados, aunque no puede interrumpir la secuencia. Por ejemplo, si un procedimiento almacenado tiene cinco parámetros, puede omitir los parámetros cuarto y quinto, pero no puede omitir el cuarto parámetro y especificar el quinto.

Ejemplo parcial 3

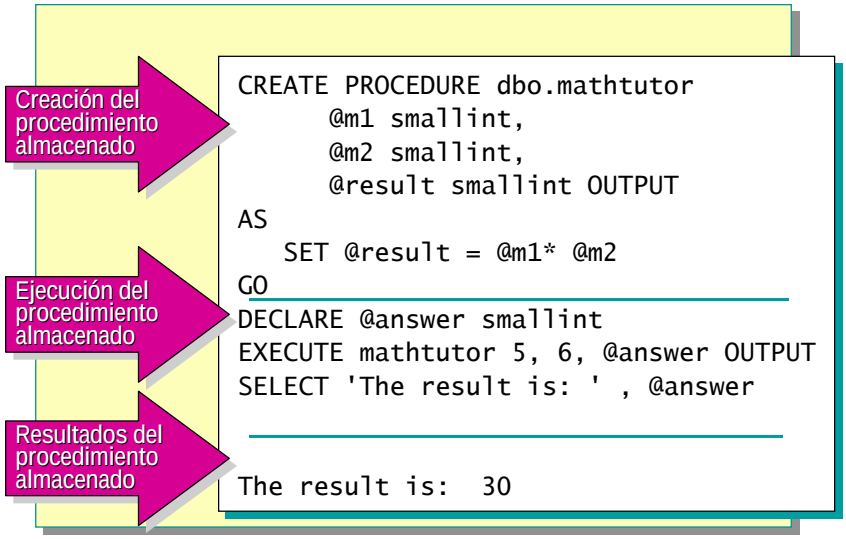
El ejemplo siguiente pasa valores por posición al procedimiento almacenado **AddCustomer**. Observe que los parámetros **@Region** y **@Fax** no tienen valores. Sin embargo, sólo el parámetro **@Region** se suministra con NULL. El parámetro **@Fax** se omite porque es el último.

```
EXEC AddCustomer 'ALFKI2', 'Alfreds Futterkiste', 'Maria
Anders', 'Sales Representative', 'Obere Str. 57', 'Berlin',
NULL, '12209', 'Germany', '030-0074321'
```

Devolución de valores mediante parámetros de salida

Objetivo del tema
Mostrar cómo la palabra clave OUTPUT puede producir un valor.

Explicación previa
Un procedimiento almacenado puede devolver varios valores. Cada uno debe ser definido como una variable con la palabra clave OUTPUT, tanto en el procedimiento almacenado como en las instrucciones que realizan la llamada.



Sugerencia
Describa en detalle la diapositiva de ejemplo y demuestre cómo crear y usar el procedimiento almacenado MathTutor.

Los procedimientos almacenados pueden devolver información al procedimiento almacenado o cliente que realiza la llamada con parámetros de salida (variables designadas con la palabra clave OUTPUT). Al usar parámetros de salida, cualquier cambio que se realice en el parámetro y que resulte de la ejecución del procedimiento almacenado se puede conservar, incluso después de que termine la ejecución.

Para usar un parámetro de salida, debe especificarse la palabra clave OUTPUT en las instrucciones CREATE PROCEDURE y EXECUTE. Si se omite la palabra clave OUTPUT cuando se ejecuta el procedimiento almacenado, éste se ejecuta igualmente, pero no devuelve ningún valor. Los parámetros de salida tienen las características siguientes:

- La instrucción que realiza la llamada debe contener un nombre de variable para recibir el valor devuelto. No se pueden pasar constantes.
- Puede usar la variable posteriormente en instrucciones de Transact-SQL adicionales del proceso por lotes o del procedimiento almacenado que realiza la llamada.
- El parámetro puede ser de cualquier tipo de datos, salvo **text** o **image**.
- Pueden ser marcadores de posición de cursores.

Ejemplo 1

En este ejemplo se crea un procedimiento almacenado **MathTutor** que calcula el producto de dos números. Este ejemplo utiliza la instrucción SET. No obstante, puede utilizar también la instrucción SELECT para concatenar dinámicamente una cadena. Una instrucción SET requiere que se declare una variable para imprimir la cadena “El resultado es:”.

```
CREATE PROCEDURE dbo.MathTutor
   @m1 smallint,
   @m2 smallint,
   @result smallint OUTPUT
AS
   SET @result = @m1* @m2
GO
```

Este proceso por lotes llama al procedimiento almacenado **MathTutor** y pasa los valores 5 y 6. Estos valores se convierten en variables que se introducen en la instrucción SET.

```
DECLARE @answer smallint
EXECUTE MathTutor 5,6, @answer OUTPUT
SELECT 'The result is: ', @answer
```

Resultado

El parámetro **@result** se designa con la palabra clave OUTPUT. SQL Server imprime el contenido de la variable **@result** cuando ejecuta el procedimiento almacenado **MathTutor**. La variable de resultado se define como el producto de los dos valores, 5 y 6.

```
The result is: 30
```

Volver a compilar explícitamente procedimientos almacenados

Objetivo del tema
Describir cuándo se debe volver a compilar procedimientos almacenados y qué opciones se deben utilizar.

Explicación previa
Los procedimientos almacenados se pueden volver a compilar explícitamente, aunque es aconsejable no hacerlo a menudo.

- Volver a compilar cuando
  - El procedimiento almacenado devuelve conjuntos de resultados que varían considerablemente
  - Se agrega un nuevo índice a una tabla subyacente
  - El valor del parámetro es atípico
- Volver a compilar mediante
  - CREATE PROCEDURE [WITH RECOMPILE]
  - EXECUTE [WITH RECOMPILE]
  - sp_recompile

Los procedimientos almacenados se pueden volver a compilar explícitamente, aunque debe tratar de evitarlo y hacerlo sólo cuando:

- Los valores de los parámetros se pasan a un procedimiento almacenado que devuelve conjuntos de resultados que varían considerablemente.
- Se agrega un índice nuevo a una tabla subyacente del que puede beneficiarse un procedimiento almacenado.
- El valor del parámetro que está suministrando es atípico.

SQL Server proporciona tres métodos para volver a compilar explícitamente un procedimiento almacenado.

CREATE PROCEDURE...[WITH RECOMPILE]

La instrucción CREATE PROCEDURE...[WITH RECOMPILE] indica que SQL Server no almacene en la caché un plan para este procedimiento almacenado. En su lugar, la opción indica que se vuelva a compilar el procedimiento almacenado cada vez que se ejecute.

Ejemplo 1

En el ejemplo siguiente se crea un procedimiento almacenado llamado **OrderCount** que se vuelve a compilar cada vez que se ejecuta.

```
USE Northwind
GO
CREATE PROC dbo.OrderCount
@CustomerID nchar (10)
WITH RECOMPILE
AS
   SELECT count(*) FROM [Orders Qry]
   WHERE CustomerID = @CustomerID
GO
```

EXECUTE...[WITH RECOMPILE]

La instrucción EXECUTE...[WITH RECOMPILE] crea un plan de ejecución nuevo cada vez que se ejecuta el procedimiento, si se especifica WITH RECOMPILE. El nuevo plan de ejecución no se almacena en la caché. Utilice esta opción si el parámetro que está pasando varía mucho de los que normalmente se pasan a este procedimiento almacenado. Puesto que este plan optimizado es la excepción de la regla, cuando se termina la ejecución, debe volver a ejecutar el procedimiento almacenado con los parámetros que se le pasan normalmente. Esta opción es útil también si los datos han cambiado significativamente desde que se compiló por última vez el procedimiento almacenado.

Ejemplo 2

Este ejemplo vuelve a compilar el procedimiento almacenado **sp_help** en el momento en el que se ejecuta.

```
EXEC  sp_help WITH RECOMPILE
```

sp_recompile

El procedimiento almacenado del sistema **sp_recompile** vuelve a compilar el procedimiento almacenado o desencadenador especificado la próxima vez que se ejecute. Si el parámetro **@objname** especifica una tabla o vista, todos los procedimientos almacenados que usan el objeto nombrado se volverán a compilar la siguiente vez que se ejecuten.

Use el procedimiento almacenado del sistema **sp_recompile** con la opción *nombreTabla* si ha agregado un índice nuevo a una tabla subyacente a la que hace referencia el procedimiento almacenado y cree que el rendimiento del procedimiento almacenado se beneficiará del nuevo índice.

Ejemplo 3

En este ejemplo se vuelven a compilar todos los procedimientos almacenados o desencadenadores que hacen referencia a la tabla **Customers** en la base de datos **Northwind**.

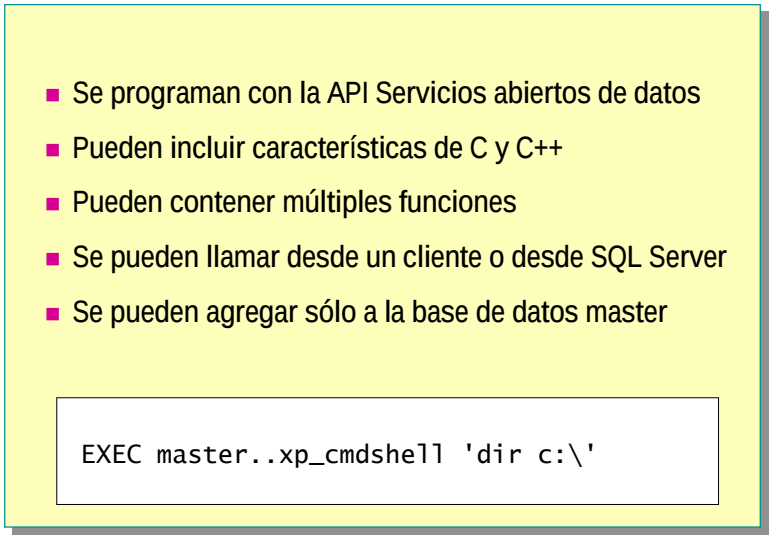
```
EXEC sp_recompile Customers
```

Nota Puede utilizar DBCC FREEPROCCACHE para borrar de la caché todos los planes de procedimientos almacenados.

Ejecución de procedimientos almacenados extendidos

Objetivo del tema
Describir las características de los procedimientos almacenados extendidos.

Explicación previa
Los procedimientos almacenados extendidos...



Los procedimientos almacenados extendidos son funciones de una biblioteca DLL que aumentan las funcionalidades de SQL Server. Se ejecutan de la misma forma que los procedimientos almacenados y admiten parámetros de entrada, códigos de estado de retorno y parámetros de salida.

Ejemplo 1

En este ejemplo se ejecuta el procedimiento almacenado extendido **xp_cmdshell** que muestra una lista de archivos y subdirectorios al ejecutar el comando del sistema operativo **dir**.

```
EXEC master..xp_cmdshell 'dir c:\ '
```

Los procedimientos almacenados extendidos:

- Se programan con la interfaz de programación de aplicaciones (API) Servicios abiertos de datos (ODS, *Open Data Services*).
- Le permiten crear sus propias rutinas externas en lenguajes de programación como Microsoft Visual C++® y Visual C.
- Pueden contener múltiples funciones.
- Se pueden llamar desde un cliente o desde SQL Server.
- Se pueden agregar sólo a la base de datos **master**.

Nota Un procedimiento almacenado extendido sólo se puede ejecutar desde la base de datos **master** o al especificar de forma explícita la ubicación de **master**. También puede crear un procedimiento almacenado del sistema definido por el usuario que llame al procedimiento almacenado extendido. Esto le permite ejecutar el procedimiento almacenado extendido desde cualquier base de datos.

Sugerencia
Demuestre cómo ejecutar el procedimiento almacenado extendido xp_cmdshell para ver los archivos que hay en alguno de los directorios de SQL Server.

En la tabla siguiente se incluyen algunos procedimientos almacenados extendidos utilizados comúnmente.

Procedimiento almacenado extendido	Descripción
xp_cmdshell	Ejecuta una cadena de comandos determinada como un comando del núcleo del sistema operativo y devuelve el resultado como filas de texto.
xp_logevent	Registra un mensaje definido por el usuario en un archivo de registro de SQL Server o en el Visor de sucesos de Windows 2000.

Ejemplo 2

En este ejemplo se ejecuta el procedimiento almacenado del sistema **sp_helptext** para mostrar el nombre de la biblioteca DLL que contiene el procedimiento almacenado extendido **xp_cmdshell**.

```
EXEC master..sp_helptext xp_cmdshell
```

Resultado

El resultado muestra la biblioteca DLL que contiene el procedimiento almacenado extendido **xp_cmdshell**.

```
xplog70.dll
```

Puede crear sus propios procedimientos almacenados extendidos. Generalmente, puede llamar a procedimientos almacenados extendidos para comunicarse con otras aplicaciones o con el sistema operativo. Por ejemplo, la biblioteca Sqlmap70.dll le permite enviar mensajes de correo electrónico desde SQL Server mediante el procedimiento almacenado extendido **xp_sendmail**.

Cuando selecciona **Herramientas de desarrollo** durante la instalación de SQL Server, SQL Server instala procedimientos almacenados extendidos de ejemplo en la carpeta C:\Archivos de programa\Microsoft SQL Server\80\Tools\Devtools\Samples\ODS como archivo ejecutable comprimido autoextraíble.

Control de mensajes de error

Objetivo del tema
Describir las distintas opciones para crear mensajes de error en procedimientos almacenados.

Explicación previa
Para mejorar la efectividad de los procedimientos almacenados, debe incluir mensajes de error que comuniquen el estado de las transacciones al usuario.

- La instrucción RETURN sale incondicionalmente de una consulta o procedimiento
- sp_addmessage crea mensajes de error personalizados
- @@error contiene el número de error de la instrucción ejecutada más recientemente
- Instrucción RAISERROR
  - Devuelve un mensaje de error del sistema definido por el usuario
  - Establece un indicador del sistema para registrar un error

En las páginas siguientes puede encontrar una demostración del funcionamiento de los mensajes de error.

Para mejorar la efectividad de los procedimientos almacenados, debe incluir mensajes de error que comuniquen el estado de las transacciones (éxito o error) al usuario. Es conveniente realizar la comprobación de la lógica y de los errores de las tareas y de las funciones *antes* de comenzar las transacciones; asimismo, éstas deben ser cortas.

Se pueden utilizar estrategias de codificación, como la realización de comprobaciones de existencia, para reconocer los errores. Cuando se produzca un error, proporcione al cliente tanta información como sea posible. En la lógica del control de errores, puede comprobar los códigos de retorno, los errores de SQL Server y los mensajes de error personalizados.

Instrucción RETURN

La instrucción RETURN sale incondicionalmente de una consulta o procedimiento almacenado. También puede devolver el estado como un valor entero (código de retorno).

El valor 0 indica éxito. En la actualidad se utilizan los valores de retorno de 0 a -14, mientras que los valores de retorno de -15 a -99 están reservados para usarse en el futuro. Si no se proporciona un valor de retorno definido por el usuario, se usa el valor de SQL Server. Los valores de retorno definidos por el usuario tienen precedencia sobre los que suministra SQL Server.

Ejemplo 1

En este ejemplo se crea el procedimiento almacenado **GetOrders** que recupera información de las tablas **Orders** y **Customers** mediante la consulta de la vista **Orders Qry**. La instrucción RETURN del procedimiento almacenado **GetOrders** devuelve el número total de filas de la instrucción SELECT a otro procedimiento almacenado. También puede anidar el procedimiento almacenado **GetOrders** dentro de otro.

```
USE Northwind
GO
CREATE PROCEDURE dbo.GetOrders
   @CustomerID nchar (10)
AS
   SELECT OrderID, CustomerID, EmployeeID
   FROM [Orders Qry]
   WHERE CustomerID = @CustomerID
   RETURN (@@ROWCOUNT)
GO
```

sp_addmessage

Este procedimiento almacenado permite a los programadores crear mensajes de error personalizados. SQL Server trata los mensajes de error personalizados y del sistema de la misma forma. Todos los mensajes se almacenan en la tabla **sysmessages** de la base de datos **master**. Estos mensajes de error se pueden escribir automáticamente en el registro de aplicación de Windows 2000.

Ejemplo 2

En este ejemplo se crea un mensaje de error definido por el usuario que requiere que el mensaje se escriba en el registro de aplicación de Windows 2000 cuando ocurra.

```
EXEC sp_addmessage
@msgnum = 50010,
@severity = 10,
@lang= 'us_english',
@msgtext = 'Customer cannot be deleted.',
@with_log = 'true'
```

@@error

Esta función del sistema contiene el número de error de la instrucción de Transact-SQL ejecutada más recientemente. Se borra y se reinicializa con cada instrucción que se ejecuta. Si la instrucción se ejecuta correctamente, se devuelve el valor 0. Puede usar la función del sistema **@@error** para detectar un número específico de error o para salir condicionalmente de un procedimiento almacenado.

Ejemplo 3

En este ejemplo se crea el procedimiento almacenado **AddSupplierProduct** en la base de datos **Northwind**. Este procedimiento almacenado utiliza la función del sistema **@@error** para determinar si se produce un error cuando se ejecuta cada instrucción INSERT. Si se produce el error, la transacción se deshace.

```
USE Northwind
GO
CREATE PROCEDURE dbo.AddSupplierProduct
        @CompanyName nvarchar (40) = NULL,
        @ContactName nvarchar (40) = NULL,
        @ContactTitle nvarchar (40)= NULL,
        @Address nvarchar (60) = NULL,
        @City nvarchar (15) = NULL,
        @Region nvarchar (40) = NULL,
        @PostalCode nvarchar (10) = NULL,
        @Country nvarchar (15) = NULL,
        @Phone nvarchar (24) = NULL,
        @Fax nvarchar (24) = NULL,
        @HomePage ntext = NULL,
        @ProductName nvarchar (40) = NULL,
        @CategoryID int = NULL,
        @QuantityPerUnit nvarchar (20) = NULL,
        @UnitPrice money = NULL,
        @UnitsInStock smallint = NULL,
        @UnitsOnOrder smallint = NULL,
        @ReorderLevel smallint = NULL,
        @Discontinued bit  = NULL
AS
BEGIN TRANSACTION
   INSERT Suppliers (
     CompanyName,
     ContactName,
     Address,
     City,
     Region,
     PostalCode,
     Country,
     Phone)
   VALUES (
     @CompanyName,
     @ContactName,
     @Address,
     @City,
     @Region,
     @PostalCode,
     @Country,
     @Phone)
  IF @@error <> 0
     BEGIN
       ROLLBACK TRAN
          RETURN
      END
  DECLARE @InsertSupplierID int
  SELECT @InsertSupplierID=@@identity
  INSERT Products (
     ProductName,
     SupplierID,
     CategoryID,      QuantityPerUnit,
     Discontinued)
  VALUES (
```

```
    @ProductName,
    @InsertSupplierID,
    @CategoryID,
    @QuantityPerUnit,
    @Discontinued)
  IF @@error <> 0
    BEGIN
        ROLLBACK TRAN
           RETURN
    END
COMMIT TRANSACTION
```

Instrucción RAISERROR

La instrucción RAISERROR devuelve un mensaje de error definido por el usuario y establece un indicador del sistema para advertir de que se ha producido un error. Cuando la utilice, debe especificar un nivel de gravedad del error y un estado del mensaje.

La instrucción RAISERROR permite a la aplicación recuperar una entrada de la tabla del sistema **master.sysmessages** o crear un mensaje dinámicamente con la gravedad y la información de estado que especifique el usuario. La instrucción RAISERROR puede escribir mensajes de error en el registro de errores de SQL Server y en el registro de aplicación de Windows 2000.

Ejemplo 4

En este ejemplo se genera un mensaje de error definido por el usuario y se escribe en el registro de aplicación de Windows 2000.

```
RAISERROR(50010, 16, 1) WITH LOG
```

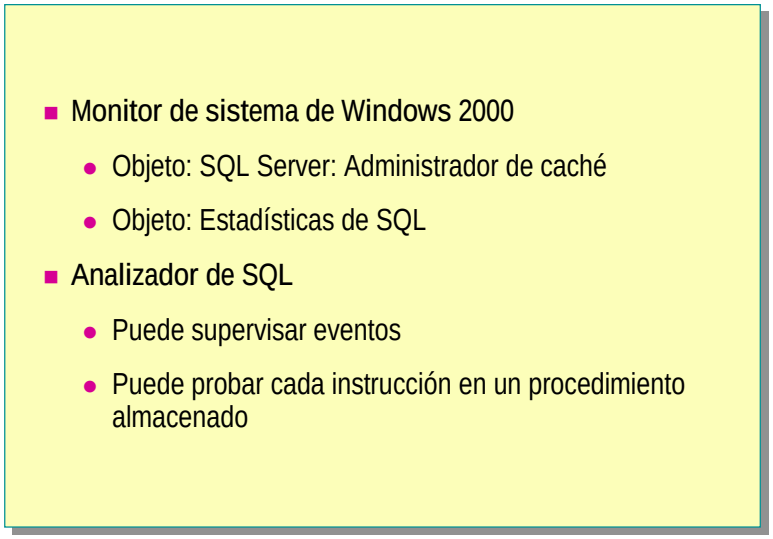
Sugerencia
La instrucción RAISERROR requiere que se especifique el nivel de gravedad del error y el estado del mensaje.

Notas La instrucción PRINT devuelve un mensaje definido por el usuario al controlador de mensajes del cliente; sin embargo, al contrario que la instrucción RAISERROR, la instrucción PRINT no almacena el número de error en la función del sistema **@@error**.

Consideraciones acerca del rendimiento

Objetivo del tema
Explicar algunas de las consideraciones de rendimiento que hay que tener en cuenta al implementar procedimientos almacenados.

Explicación previa
Cuando se implementan procedimientos almacenados, es necesario tener en cuenta estas consideraciones de rendimiento.



Puede usar las herramientas siguientes para ayudarle a detectar el origen de problemas de rendimiento que pueden estar relacionados con la ejecución de procedimientos almacenados.

Monitor de sistema de Windows 2000

El Monitor de sistema de Windows 2000 supervisa la utilización de la caché de procedimientos, además de otras muchas actividades relacionadas.

Los objetos y contadores siguientes proporcionan información general acerca de los planes compilados de la caché de procedimientos y del número de recompilaciones. También puede monitorizar una instancia específica, como el **plan de procedimientos**.

Objeto	Contadores
SQL Server: Administrador de caché	Proporción de aciertos de caché
	Contador de objetos de caché
	Páginas de la caché
	Contador de uso de caché/seg.
Estadísticas de SQL	Recompilaciones de SQL/seg.

Analizador de SQL

El Analizador de SQL es una herramienta gráfica que le permite supervisar eventos, como cuándo se ha iniciado o se ha completado el procedimiento almacenado o cuándo se han iniciado o completado determinadas instrucciones de Transact-SQL individuales de un procedimiento almacenado. Además, puede supervisar si un procedimiento almacenado se encuentra en la caché de procedimientos.

En la fase de desarrollo de un proyecto, también puede probar las instrucciones del procedimiento almacenado, de línea en línea, para confirmar que las instrucciones funcionan como se esperaba.

Nota Tenga cuidado cuando cree procedimientos almacenados anidados. La anidación agrega un nivel de complejidad que dificulta la resolución de problemas de rendimiento.