Administración de transacciones y bloqueos

Contenido

Notas para el instructor

Este módulo proporciona a los alumnos información acerca de cómo se utilizan las transacciones y los bloqueos para asegurar la integridad de las transacciones a la vez que se permite el uso simultáneo. El módulo prosigue con la descripción de cómo se ejecutan y se deshacen las transacciones. Una animación breve ayuda a describir cómo funciona el procesamiento de las transacciones.

El módulo describe a continuación cómo los bloqueos de Microsoft® SQL Server™ 2000 mantienen la coherencia y el uso simultáneo de los datos. Se presentan los recursos que se pueden bloquear, los diferentes tipos de bloqueos y la compatibilidad entre los bloqueos. La última sección describe algunas opciones de bloqueo, trata los interbloqueos y explica cómo presentar información acerca de los bloqueos activos.

En esta práctica los alumnos definen una transacción y observan el efecto de las instrucciones BEGIN TRAN, COMMIT TRAN y ROLLBACK TRAN. A continuación, observan lo que ocurre cuando se aplican diferentes opciones de bloqueo a una transacción.

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

- Describir el proceso de las transacciones.
- Ejecutar, cancelar o deshacer una transacción.
- Identificar los problemas de la simultaneidad de bloqueos.
- Identificar los recursos que se pueden bloquear y los tipos de bloqueos.
- Describir la compatibilidad de los bloqueos.
- Describir cómo SQL Server utiliza el bloqueo dinámico.
- Establecer opciones de bloqueo y presentar información acerca de los bloqueos.

Introducción

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

Explicación previa
En este módulo, trataremos...

- Introducción a las transacciones y los bloqueos
- Administración de las transacciones
- Bloqueos en SQL Server
- Administración de los bloqueos

Objetivos

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

- Describir el proceso de transacciones.
- Ejecutar, cancelar o deshacer una transacción.
- Identificar los problemas de la simultaneidad de bloqueos.
- Identificar los recursos que se pueden bloquear y los tipos de bloqueos.
- Describir la compatibilidad de los bloqueos.
- Describir cómo Microsoft® SQL Server™ 2000 utiliza el bloqueo dinámico.
- Establecer opciones de bloqueo y presentar información acerca de los bloqueos.

Introducción a las transacciones y los bloqueos

Objetivo del tema
Proporcionar una introducción a este tema.

Explicación previa
Las transacciones y los bloqueos aseguran la integridad de las transacciones.

- Las transacciones aseguran que varias modificaciones a los datos se procesan juntas
- Los bloqueos impiden los conflictos de actualización
  - Las transacciones están serializadas
  - El bloqueo es automático
  - Los bloqueos permiten usar los datos al mismo tiempo
- Control de simultaneidad

Sugerencia
Este módulo se centra en el proceso de transacciones en línea, no de aplicaciones de consulta, como almacenes de datos y ayuda a la toma de decisiones.

Las transacciones utilizan los bloqueos para impedir que otros usuarios cambien o lean los datos de una transacción que no se ha completado. El bloqueo es necesario en el Proceso de transacciones en línea (OLTP, *Online Transaction Processing*) en sistemas multiusuario. SQL Server utiliza el registro de transacciones para asegurar que las actualizaciones se han completado y son recuperables.

Transacciones

Las transacciones aseguran que varias modificaciones a los datos se procesan como una unidad; esto se conoce como *atomicidad*. Por ejemplo, una transacción bancaria podría abonar en una cuenta y cargar en otra. Los dos pasos se deben completar al mismo tiempo. SQL Server acepta que el proceso de transacciones administre varias transacciones.

Bloqueos

Los bloqueos impiden los conflictos de actualización. Los usuarios no pueden leer o modificar los datos que están en proceso de modificación por parte de otros usuarios. Por ejemplo, si desea calcular una función de agregado y asegurarse de que otra transacción no modifique el conjunto de datos que se utiliza para calcular la función de agregado, puede solicitar que el sistema establezca bloqueos en los datos. Tenga en cuenta los siguientes hechos acerca de los bloqueos:

- Los bloqueos hacen posible la serialización de transacciones de forma que sólo una persona a la vez pueda modificar un elemento de datos. Por ejemplo, en un sistema de reservas de una línea aérea los bloqueos aseguran que sólo se asigne un asiento concreto a una persona.
- SQL Server establece y ajusta dinámicamente el nivel de bloqueo apropiado durante una transacción. También se puede controlar manualmente cómo se utilizan algunos de los bloqueos.
- Los bloqueos son necesarios para que las transacciones simultáneas permitan que los usuarios tengan acceso y actualicen los datos al mismo tiempo. La alta simultaneidad significa que hay varios usuarios que consiguen un buen tiempo de respuesta con pocos conflictos. Desde la perspectiva del administrador del sistema, los problemas principales son el número de usuarios, el número de transacciones y el rendimiento. Desde la perspectiva del usuario, la preocupación principal es el tiempo de respuesta.

Control de simultaneidad

El control de simultaneidad garantiza que las modificaciones que realiza un usuario no afectan de forma negativa a las modificaciones que realice otro. Hay dos tipos.

- El control de simultaneidad pesimista bloquea los datos cuando se leen para preparar una actualización. Los demás usuarios no pueden realizar acciones que alteren los datos subyacentes hasta que el usuario que ha aplicado el bloqueo termine con los datos. Utilice la simultaneidad pesimista donde haya una alta contención de los datos y el costo de proteger los datos con bloqueos sea menor que el costo de deshacer transacciones si se producen conflictos de simultaneidad.
- El control de simultaneidad optimista no bloquea los datos cuando se leen inicialmente. En su lugar, cuando se realiza una actualización, SQL Server realiza comprobaciones para determinar si los datos subyacentes han cambiado desde que se leyeron inicialmente. De ser así, al usuario le aparece un error, la transacción se deshace y el usuario debe volver a empezar. Utilice la simultaneidad optimista cuando haya contención baja de los datos y el costo de deshacer ocasionalmente una transacción sea menor que el costo de bloquear los datos cuando se leen.

SQL Server admite una gran variedad de mecanismos de control de simultaneidad optimista y pesimista. Los usuarios indican el tipo de control de simultaneidad al especificar el nivel de aislamiento de transacciones para una conexión.

Administración de las transacciones

Objetivo del tema
Proporcionar un resumen de este tema.

Explicación previa
En esta sección, trataremos…

- Presentación multimedia: Transacciones de SQL Server
- Recuperación de transacciones y puntos de comprobación
- Consideraciones para el uso de transacciones
- Establecimiento de la opción de transacciones implícitas
- Restricciones en las transacciones definidas por el usuario

Esta sección describe cómo se definen las transacciones, qué hay que tener en cuenta al utilizarlas, cómo se establece una opción de transacción implícita y las restricciones en el uso de las transacciones. También describe el procesamiento y la recuperación de transacciones.

Transacciones de SQL Server

Advertencia
Para mostrar los subtítulos de la presentación, active la opción Títulos del menú Ver en el Reproductor de Windows Media™.

En SQL Server hay dos clases de transacciones:

- En una transacción implícita, cada instrucción Transact-SQL, como INSERT, UPDATE o DELETE, se ejecuta como una transacción.
- En una transacción explícita o definida por el usuario, las instrucciones de la transacción se agrupan entre las cláusulas BEGIN TRANSACTION y COMMIT TRANSACTION.

Punto clave
Una transacción confirmada no se puede deshacer.

El usuario puede establecer un punto de almacenamiento, o marcador, en una transacción. Un punto de almacenamiento define una ubicación a la que puede volver una transacción si parte de la misma se cancela condicionalmente. La transacción debe continuar hasta que se complete o se deshaga en su totalidad.

Las transacciones de SQL Server emplean la sintaxis siguiente.

Sintaxis

BEGIN TRAN[SACTION] [*transacción* | *@variableTransacción* [WITH MARK [‘*descripción’*]]]

La opción *transacción* especifica un nombre de transacción definido por el usuario. En *variableTransacción* se especifica el nombre de una variable definida por el usuario con un nombre de transacción válido. WITH MARK especifica que la transacción está marcada en el registro de transacciones. *Descripción* es una cadena que describe la marca que permite WITH MARK para restaurar un registro de transacciones a una marca con nombre.

Sintaxis

SAVE TRAN[SACTION] {*puntoAlmacenamiento* | *@variablePuntoAlmacenamiento*}

Sintaxis

BEGIN DISTRIBUTED TRAN[SACTION] [*transacción* | *@variableTransacción*]

Sintaxis

COMMIT [TRAN[SACTION] [transacción | @variableTransacción]]

Sintaxis

ROLLBACK [TRAN[SACTION] [*transacción* | *@variableTransacción* | *puntoAlmacenamiento* | *@variablePuntoAlmacenamiento*]]

Ejemplo

Este ejemplo define una transacción que transfiere fondos entre la cuenta corriente y la cuenta de ahorro de un cliente.

Este ejemplo no se puede ejecutar porque los procedimientos almacenados no existen.

```
BEGIN TRAN fund_transfer
   EXEC debit_checking 100, 'account1'
   EXEC credit_savings 100, 'account1'
COMMIT TRAN fund_transfer
```

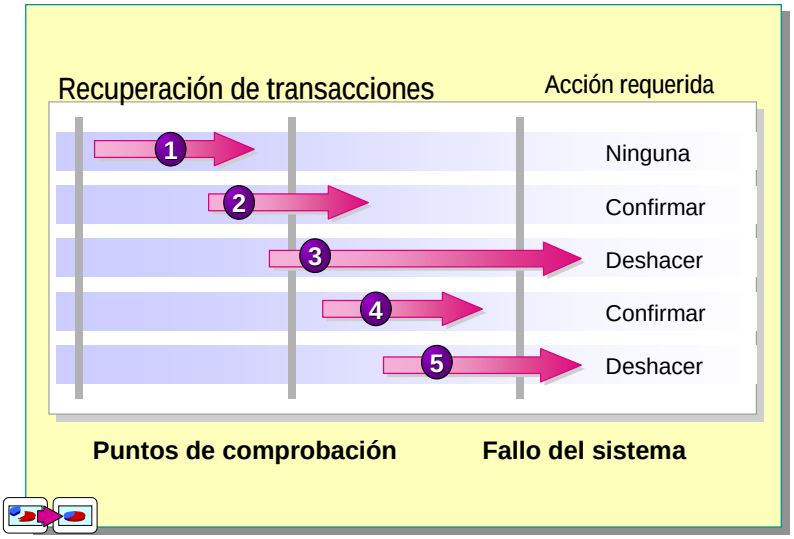
Descripción del registro de transacciones

Todas las transacciones se graban en un registro de transacciones para mantener la coherencia de la base de datos y facilitar la recuperación. El registro es un área de almacenamiento que efectúa automáticamente el seguimiento de todos los cambios realizados en la base de datos, a excepción de las operaciones no registradas. Las modificaciones se graban en el registro en disco cuando se ejecutan, antes de escribirse en la base de datos.

Recuperación de transacciones y puntos de comprobación

Objetivo del tema
Explicar el proceso de recuperación.

Explicación previa
Como el registro de transacciones graba todas las modificaciones, los datos se pueden recuperar fácilmente en caso de que se produzca un error del sistema.



Como el registro de transacciones graba todas las transacciones, SQL Server puede recuperar los datos automáticamente en el caso de un corte de energía, un error en el software del sistema, problemas en el cliente o una petición de cancelación de una transacción.

Sugerencia
Utilice la diapositiva como punto de partida para la explicación. Pregunte a los alumnos por qué es necesaria cada una de las acciones.

SQL Server garantiza automáticamente que todas las transacciones confirmadas quedan reflejadas en la base de datos en caso de que se produzca un error. Utiliza el registro de transacciones para rehacer todas las transacciones confirmadas y deshacer las no confirmadas. En el ejemplo de la diapositiva:

- La transacción 1 se ha confirmado antes del punto de comprobación, de modo que queda reflejada en la base de datos.
- Las transacciones 2 y 4 se han confirmado después del punto de comprobación, de modo que deben reconstruirse (rehacerse) a partir del registro.
- Las transacciones 3 y 5 no se han confirmado, por lo que SQL Server las deshace.

Inicialmente, las páginas de la caché de datos y las del disco son iguales. Después, tiene lugar el siguiente proceso:

- Los cambios que aparecen en la caché de datos como transacciones se confirman.
- Cuando la caché se llena, las páginas modificadas se escriben en disco.
- Cuando se produce un punto de comprobación, la caché se escribe en disco. El disco vuelve a tener los mismos datos que la caché.

Importante Utilice un controlador de disco con caché de escritura con SQL Server sólo si se ha diseñado para su uso con un servidor de bases de datos. Si no se hace así, se comprometerá la capacidad de SQL Server de administrar transacciones. Un controlador de disco con caché de escritura puede hacer que parezca que está terminado el registro de preescritura, incluso si no es así.

Consideraciones para el uso de transacciones

Objetivo del tema
Identificar los problemas del uso de transacciones.

Explicación previa
En general, mantenga las transacciones lo más cortas posible.

- Recomendaciones para las transacciones
  - Las transacciones deben ser lo más cortas posible
  - Preste atención a ciertas instrucciones Transact-SQL
  - Evite las transacciones que requieran la intervención del usuario
- Aspectos del anidamiento de transacciones
  - Se pueden anidar transacciones, pero no se recomienda
  - Utilice @@trancount para determinar el nivel de anidamiento

Suele ser conveniente mantener las transacciones en un tamaño reducido y evitar el anidamiento de transacciones.

Recomendaciones para las transacciones

Las transacciones deben ser lo más cortas posible. Las transacciones mayores aumentan la posibilidad de que los usuarios no puedan tener acceso a los datos bloqueados. He aquí algunos de los métodos para mantener las transacciones cortas:

- Para minimizar la duración de la transacción, preste atención cuando utilice ciertas instrucciones Transact-SQL, como WHILE o instrucciones del Lenguaje de definición de datos (DDL, *Data Definition Language*).
- No requiera la intervención del usuario durante una transacción. Resuelva los aspectos que requieran la intervención del usuario antes de iniciar la transacción. Por ejemplo, si va a actualizar el registro de un cliente, obtenga la información necesaria del usuario antes de comenzar la transacción.
- INSERT, UPDATE y DELETE deben ser las instrucciones principales de una transacción, y deben escribirse de forma que afecten al menor número de filas posible. Una transacción nunca debe ser menor que una unidad lógica de trabajo.
- Si es posible, no abra una transacción mientras examina los datos. Las transacciones no deben empezar hasta que no se hayan realizado todos los análisis de datos preliminares.
- Obtenga acceso a la mínima cantidad de datos posible mientras se encuentre en una transacción. De esta forma disminuye el número de filas bloqueadas y se reduce la contención.

Aspectos del anidamiento de transacciones

Tenga en cuenta lo siguiente en cuanto al anidamiento de transacciones:

- Se pueden anidar transacciones, pero el anidamiento no afecta a cómo SQL Server procesa la transacción. Debe utilizar el anidamiento cuidadosamente, si la hubiera, porque el no confirmar o deshacer una transacción deja activados los bloqueos indefinidamente.

  Sólo se aplica la pareja de instrucciones BEGIN…COMMIT más externa. Normalmente, el anidamiento de transacciones se produce cuando se invocan entre sí procedimientos almacenados con parejas de instrucciones BEGIN...COMMIT o desencadenadores.

- Puede utilizar la variable global **@@trancount** para determinar si hay alguna transacción abierta y su nivel de anidamiento:

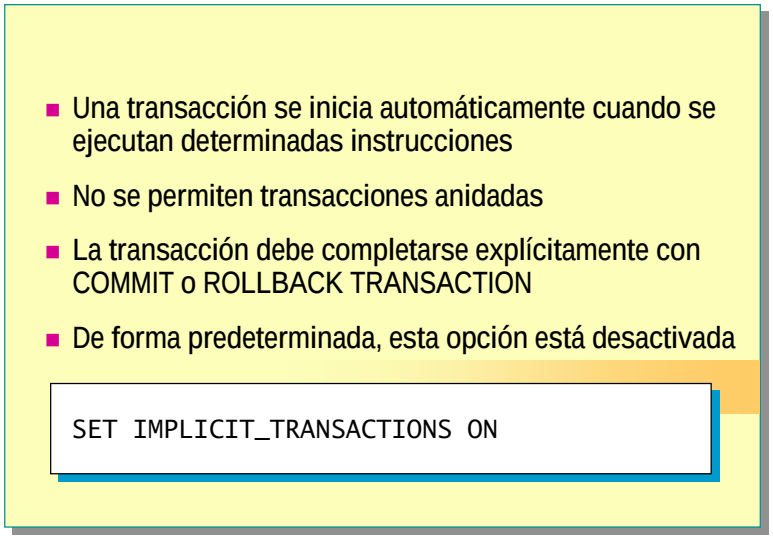
  - **@@trancount** es cero cuando no hay transacciones abiertas.
  - Una instrucción BEGIN TRAN incrementa **@@trancount** en uno y una instrucción ROLLBACK TRAN establece **@@trancount** en cero.

Nota También puede utilizar la instrucción DBCC OPENTRAN en la sesión actual para obtener información acerca de las transacciones activas.

Establecimiento de la opción de transacciones implícitas

Objetivo del tema
Describir cómo se establecen las transacciones implícitas.

Explicación previa
Las transacciones implícitas pueden ser útiles cuando se migran aplicaciones a SQL Server.



En la mayoría de los casos, es preferible definir las transacciones explícitamente con la instrucción BEGIN TRANSACTION. Sin embargo, en aplicaciones que se desarrollaron originalmente en sistemas diferentes de SQL Server, la opción SET IMPLICIT_TRANSACTIONS puede ser útil. Establece el modo de transacción implícita en una conexión.

Sintaxis

```
SET IMPLICIT_TRANSACTIONS {ON | OFF}
```

Al establecer transacciones implícitas, tenga en cuenta lo siguiente:

- Cuando el modo de transacción implícita de una conexión está activado, la ejecución de cualquiera de las instrucciones siguientes desencadena el inicio de una transacción:

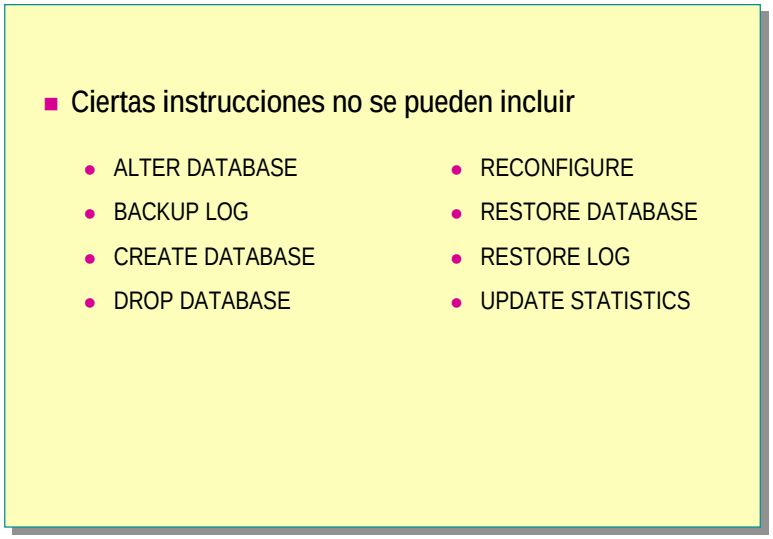
ALTER TABLE	INSERT
CREATE	OPEN
DELETE	REVOKE
DROP	SELECT
FETCH	TRUNCATE TABLE
GRANT	UPDATE

- No se permiten transacciones anidadas. Si la conexión ya se encuentra en una transacción abierta, las instrucciones no inician una nueva transacción.
- Cuando esta opción está activada, el usuario tiene que confirmar o deshacer la transacción explícitamente al final de la transacción. De lo contrario, cuando el usuario se desconecte se deshará la transacción y todos los cambios a los datos que contiene.
- De forma predeterminada, esta opción está desactivada.

Restricciones en las transacciones definidas por el usuario

Objetivo del tema
Indicar los elementos que no se pueden utilizar en las transacciones definidas por el usuario.

Explicación previa
Hay algunas restricciones a las transacciones definidas por el usuario.



Hay algunas restricciones a las transacciones definidas por el usuario:

- Ciertas instrucciones no se pueden incluir en una transacción explícita. Por ejemplo, algunas de ellas son operaciones de ejecución prolongada que no se suelen utilizar en el contexto de una transacción. Las instrucciones restringidas son las siguientes:
  - ALTER DATABASE
  - BACKUP LOG
  - CREATE DATABASE
  - DROP DATABASE
  - RECONFIGURE
  - RESTORE DATABASE
  - RESTORE LOG
  - UPDATE STATISTICS

Bloqueos en SQL Server

Objetivo del tema
Proporcionar un resumen de este tema.

Explicación previa
En esta sección, trataremos…

- Problemas de simultaneidad impedidos por los bloqueos
- Recursos que se pueden bloquear
- Tipos de bloqueos
- Compatibilidad de los bloqueos

En esta sección se describen los problemas de simultaneidad, los recursos que se pueden bloquear, los tipos de bloqueos que se pueden establecer sobre dichos recursos y cómo se pueden combinar los bloqueos.

Problemas de simultaneidad impedidos por los bloqueos

Objetivo del tema
Describir los problemas más comunes de la simultaneidad de los bloqueos.

Explicación previa
Los bloqueos son útiles para asegurar la integridad de las transacciones en estas situaciones…

- Actualización perdida
- Dependencia no confirmada (lectura no confirmada)
- Análisis incoherente (lectura no repetible)
- Lecturas fantasma

Los bloqueos pueden impedir las siguientes situaciones que comprometen la integridad de las transacciones:

Actualización perdida Una actualización se puede perder cuando una transacción sobrescribe los cambios de otra transacción. Por ejemplo, dos usuarios pueden actualizar la misma información, pero sólo la última modificación queda reflejada en la base de datos.

Dependencia no confirmada (lectura no confirmada) Una dependencia no confirmada ocurre cuando una transacción lee los datos sin confirmar de otra transacción. La transacción puede hacer cambios según datos que no son correctos o que no existen.

Análisis incoherente (lectura no repetible) Un análisis incoherente ocurre cuando una transacción lee la misma fila varias veces y cuando, entre las dos (o más) lecturas, otra transacción modifica esa fila. Como la fila se ha modificado entre lecturas de una misma transacción, cada lectura produce valores diferentes, lo que causa incoherencias.

Por ejemplo, un editor lee el mismo documento dos veces, pero de una lectura a otra, el escritor vuelve a escribir el documento. Cuando el editor lee el documento por segunda vez, ha cambiado por completo. La lectura original no se puede repetir, lo que produce confusión. Sería mejor que el editor sólo leyera el documento después de que el escritor hubiera terminado de escribirlo.

Lecturas fantasma Las lecturas fantasma pueden ocurrir cuando las transacciones no están aisladas unas de otras. Por ejemplo, se podría hacer una actualización en todos los registros de una región al mismo tiempo que otra transacción inserta un nuevo registro de esa región. La próxima vez que la transacción lea los datos, aparecerá un registro adicional.

Recursos que se pueden bloquear

Objetivo del tema
Enumerar los recursos que SQL Server puede bloquear.

Explicación previa
El número de bloqueos se tiene que adaptar a la cantidad de datos a los que afecta cada uno de los bloqueos.

Elemento	Descripción
RID	Identificador de fila
Clave	Bloqueo de fila dentro de un índice
Página	Página de datos o página de índice
Extensión	Grupo de páginas
Tabla	Tabla completa
Base de datos	Base de datos completa

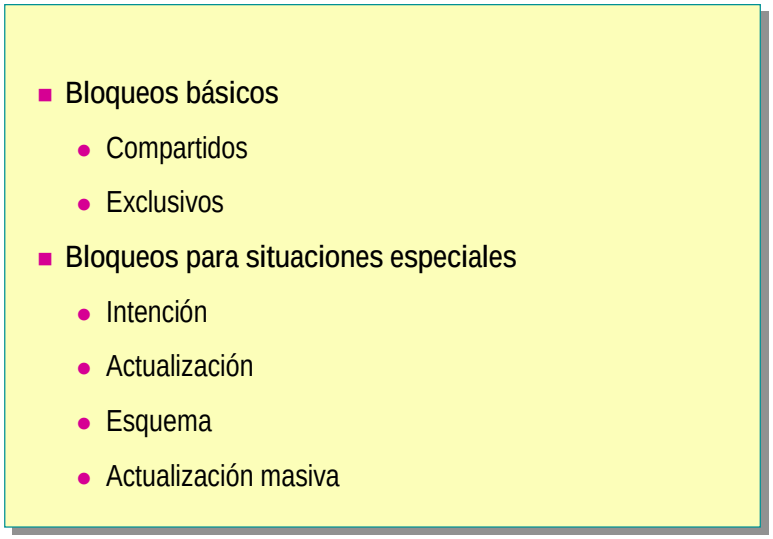
Para obtener el máximo rendimiento, el número de bloqueos mantenidos por SQL Server se tiene que adaptar a la cantidad de datos a los que afecta cada uno de los bloqueos. Para minimizar el costo de los bloqueos, SQL Server bloquea automáticamente los recursos en el nivel apropiado para la tarea. SQL Server puede bloquear los siguientes tipos de elementos.

Elemento	Descripción
RID	Identificador de fila: se utiliza para bloquear una sola fila de una tabla.
Clave	Bloqueo de fila dentro de un índice: se utiliza para proteger intervalos de claves en transacciones serializables.
Página	Página de datos o página de índice de 8 KB.
Extensión	Grupo contiguo de páginas de datos o páginas de índice: se utiliza durante la asignación de espacio.
Tabla	Tabla completa, incluidos todos los datos e índices.
Base de datos	Toda la base de datos: se utiliza durante la restauración de una base de datos.

Tipos de bloqueos

Objetivo del tema
Enumerar los tipos de bloqueos.

Explicación previa
SQL Server tiene dos tipos principales de bloqueos: bloqueos básicos y bloqueos para situaciones especiales.



SQL Server tiene dos tipos principales de bloqueos: bloqueos básicos y bloqueos para situaciones especiales.

Bloqueos básicos

En general, las operaciones de lectura adquieren bloqueos compartidos y las operaciones de escritura adquieren bloqueos exclusivos.

Bloqueos compartidos

SQL Server suele utilizar bloqueos compartidos (de lectura) en las operaciones que no modifican ni actualizan los datos. Si SQL Server ha aplicado un bloqueo compartido a un recurso, una segunda transacción también puede adquirir un bloqueo compartido, incluso si la primera transacción no ha terminado.

Tenga en cuenta los siguientes hechos acerca de los bloqueos compartidos:

- Sólo se utilizan en operaciones de lectura; los datos no se pueden modificar.
- SQL Server libera los bloqueos compartidos de un registro cuando se lee el registro siguiente.
- Un bloqueo compartido existe hasta que todas las filas que cumplen las condiciones de la consulta se han devuelto al cliente.

Bloqueos exclusivos

SQL Server utiliza bloqueos exclusivos (de escritura) en las instrucciones de modificación de datos INSERT, UPDATE y DELETE.

Tenga en cuenta los siguientes hechos acerca de los bloqueos exclusivos:

- Sólo una transacción puede conseguir un bloqueo exclusivo sobre un recurso.
- Una transacción no puede adquirir un bloqueo compartido sobre un recurso que tenga un bloqueo exclusivo.
- Una transacción no puede adquirir un bloqueo exclusivo sobre un recurso hasta que todos los bloqueos compartidos se hayan liberado.

Bloqueos para situaciones especiales

Dependiendo de la situación, SQL Server puede utilizar otros tipos de bloqueos:

Bloqueos de intención

SQL Server utiliza internamente los bloqueos de intención para minimizar los conflictos de bloqueo. Los bloqueos de intención establecen una jerarquía de bloqueo para que otras transacciones no puedan adquirir bloqueos en niveles más incluyentes que otros existentes. Por ejemplo, si una transacción tiene un bloqueo exclusivo de fila sobre un registro de cliente específico, el bloqueo de intención impide que otra transacción adquiera un bloqueo exclusivo en el nivel de tabla.

Los bloqueos de intención son: bloqueo compartido de intención (IS), bloqueo exclusivo de intención (IX) y compartido con bloqueo exclusivo de intención (SIX).

Bloqueos de actualización

SQL Server utiliza los bloqueos de actualización cuando va a modificar una página. Antes de modificar la página, SQL Server aumenta el nivel de bloqueo de actualización de página a bloqueo de página exclusivo para impedir conflictos de bloqueo.

Tenga en cuenta los siguientes hechos acerca de los bloqueos de actualización. Los bloqueos de actualización:

- Se adquieren durante la parte inicial de una operación de actualización al leer las páginas por primera vez.
- Son compatibles con los bloqueos compartidos.

Bloqueos de esquema

Los bloqueos de esquema aseguran que no se elimine una tabla o un índice, o que no se modifique su esquema, cuando se les hace referencia en otra sesión.

SQL Server proporciona dos tipos de bloqueos de esquema:

- Estabilidad del esquema (Sch-S), que asegura que no se eliminará un recurso.
- Modificación del esquema (Sch-M), que asegura que otras sesiones no hagan referencia a un recurso que está siendo modificado.

Bloqueos de actualización masiva

Los bloqueos de actualización masiva permiten procesos de copia masiva simultáneos en la misma tabla, a la vez que impiden que otros procesos que no hacen copias masivas tengan acceso a la tabla.

SQL Server utiliza bloqueos de actualización masiva cuando se especifica una de las opciones siguientes: la sugerencia TABLOCK o la opción **table lock on bulk load** (bloqueo de tabla en carga masiva), que se establece mediante el procedimiento almacenado de sistema **sp_tableoption**.

Compatibilidad de los bloqueos

Objetivo del tema
Explicar qué bloqueos son compatibles.

Explicación previa
Los bloqueos pueden ser compatibles o incompatibles con otros bloqueos.

- Los bloqueos pueden ser compatibles o incompatibles con otros bloqueos
- Ejemplos
  - Los bloqueos compartidos son compatibles con todos los bloqueos excepto con los exclusivos
  - Los bloqueos exclusivos no son compatibles con ningún otro bloqueo
  - Los bloqueos de actualización son compatibles sólo con los bloqueos compartidos

Los bloqueos pueden ser compatibles o incompatibles con otros bloqueos. Los bloqueos tienen una *matriz de compatibilidad* que muestra qué bloqueos son compatibles con otros bloqueos del mismo recurso. Los bloqueos mostrados en la siguiente tabla están enumerados en orden desde el menos restrictivo (compartido) al más restrictivo (exclusivo).

Sugerencia
Explique la matriz de bloqueos mediante los ejemplos de la diapositiva.

	Bloqueo concedido existente					
Bloqueo solicitado	**IS**	**S**	**U**	**IX**	**SIX**	**X**
Compartido de intención (IS)	Sí	Sí	Sí	Sí	Sí	No
Compartido (S)	Sí	Sí	Sí	No	No	No
Actualización (U)	Sí	Sí	No	No	No	No
Exclusivo de intención (IX)	Sí	No	No	Sí	No	No
Compartido con bloqueo exclusivo de intención (SIX)	Sí	No	No	No	No	No
Exclusivo (X)	No	No	No	No	No	No

Nota Un bloqueo IX es compatible con otros bloqueos IX porque IX implica la intención de actualizar sólo algunas de las filas, no todas.

Además, la compatibilidad de los bloqueos de esquema es la siguiente:

- El bloqueo de modificación de esquema (Sch-M) es incompatible con todos los bloqueos.
- El bloqueo de estabilidad de esquema (Sch-S) es compatible con todos los bloqueos excepto el bloqueo de modificación de esquema (Sch-M).

Administración de los bloqueos

Objetivo del tema
Proporcionar un resumen de este tema.

Explicación previa
En esta sección, trataremos…

- Opciones de bloqueo en el nivel de sesión
- Arquitectura de bloqueos dinámicos
- Opciones de bloqueo en el nivel de tabla
- Interbloqueos
- Presentación de información acerca de los bloqueos

Esta sección describe las opciones de bloqueo que se pueden especificar en los niveles de sesión y de tabla. También describe cómo SQL Server controla los interbloqueos y cómo se puede ver la información de los bloqueos.

Opciones de bloqueo en el nivel de sesión

Objetivo del tema
Presentar el nivel de aislamiento de las transacciones.

Explicación previa
SQL Server permite controlar las opciones de bloqueo en el nivel de sesión.

- Nivel de aislamiento de las transacciones
  - READ COMMITTED (DEFAULT)
  - READ UNCOMMITTED
  - REPEATABLE READ
  - SERIALIZABLE
- Tiempo de espera para los bloqueos
  - Limita el tiempo de espera para un recurso bloqueado
  - Use SET LOCK_TIMEOUT

SQL Server permite controlar las opciones de bloqueo en el nivel de sesión mediante el establecimiento del nivel de aislamiento de las transacciones.

Nivel de aislamiento de las transacciones

El *nivel de aislamiento* protege una transacción especificada de otras transacciones. Utilice el nivel de aislamiento de la transacción para establecer el nivel de aislamiento de todas las transacciones de una sesión. Al establecer el nivel de aislamiento, se especifica el comportamiento predeterminado de los bloqueos en todas las instrucciones de la sesión.

Establecer niveles de aislamiento de transacción permite a los programadores aceptar un riesgo mayor de problemas de integridad a cambio de un mayor acceso simultáneo a los datos. Cuanto mayor sea el nivel de aislamiento, durante más tiempo se mantienen los bloqueos y más restrictivos son éstos.

El nivel de aislamiento de la sesión se puede suplantar en instrucciones individuales mediante una especificación de bloqueo. También se puede utilizar la instrucción DBCC USEROPTIONS para especificar el aislamiento de la transacción en una instrucción.

Sintaxis

```
SET TRANSACTION ISOLATION LEVEL {READ COMMITTED | READ
  UNCOMMITTED | REPEATABLE READ | SERIALIZABLE}
```

La siguiente tabla describe las opciones de nivel de aislamiento de los bloqueos.

Opción	Descripción
READ COMMITTED	Indica a SQL Server que utilice bloqueos compartidos al leer. En este nivel, no pueden producirse lecturas no confirmadas.
	Indica a SQL Server que no establezca bloqueos compartidos y no atienda bloqueos exclusivos. Pueden producirse lecturas no confirmadas.
REPEATABLE READ	Indica que no pueden ocurrir lecturas no confirmadas y lecturas irrepetibles. Los bloqueos de lectura se mantienen hasta el final de la transacción.
SERIALIZABLE	Impide que otros usuarios actualicen o inserten nuevas filas que cumplan los criterios de la cláusula WHERE de la transacción. No se pueden producir datos fantasma.

Ejemplo

El siguiente ejemplo establece el nivel de aislamiento de la sesión actual como READ UNCOMMITTED y, después, comprueba DBCC USEROPTIONS para comprobar que SQL Server ha efectuado el cambio.

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED
DBCC USEROPTIONS
```

Resultado

```
set option                     value
------------------------------ ------------------------------
textsize                       64512
language                       us_english
dateformat                     mdy
datefirst                      7
.
.
.
isolation level                read uncommitted

(13 filas afectadas)
```

Nota DBCC siempre imprime el siguiente mensaje cuando se ejecuta:

```
DBCC execution completed. If DBCC printed error messages, see
your System Administrator.
```

Sugerencia
Demuestre los tiempos de espera de los bloqueos mediante varias consultas.

Tiempo de espera para los bloqueos

Con la opción SET LOCK_TIMEOUT, se puede establecer la cantidad máxima de tiempo que SQL Server permite que una transacción espere la liberación de un recurso bloqueado.

Sintaxis

SET LOCK_TIMEOUT *tiempoDeEspera*

tiempoDeEspera es el número de milisegundos que pasan hasta que SQL Server devuelve un error de bloqueo. Un valor de -1 (el valor predeterminado) indica que no hay tiempo de espera. Después de cambiarlo, el nuevo valor tiene efecto durante el resto de la sesión.

Ejemplo

En este ejemplo se establece el tiempo de espera del bloqueo en 180.000 milisegundos.

```
SET LOCK_TIMEOUT 180000
```

Para determinar el valor para la sesión actual, consulte la variable global **@@lock_timeout**.

Ejemplo

En este ejemplo se presenta el valor actual de **@@lock_timeout**.

```
SELECT @@lock_timeout
```

Resultado

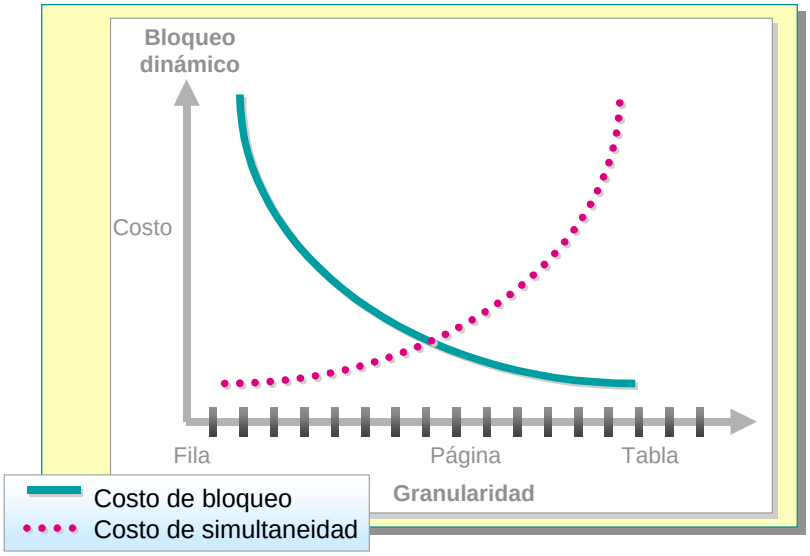
```
180000

(1 filas afectadas)
```

Arquitectura de bloqueos dinámicos

Objetivo del tema
Mostrar cómo se encuentran los bloqueos de costo más efectivo.

Explicación previa
La arquitectura de bloqueos dinámicos ayuda a determinar los bloqueos de costo más efectivo.



SQL Server utiliza una arquitectura de bloqueos dinámicos para determinar los bloqueos de costo más efectivo. Determina automáticamente qué bloqueos resultan más adecuados cuando se ejecuta una consulta, según las características del esquema y consulta.

SQL Server aumenta y reduce dinámicamente la granularidad y los tipos de bloqueos. Normalmente, el optimizador de consultas elige la granularidad de bloqueo correcta cuando se compila el plan de ejecución, con lo que se reduce la necesidad de concentrar bloqueos.

Por ejemplo, si una actualización adquiere una gran cantidad de bloqueos de nivel de fila y ha bloqueado un porcentaje significativo de una tabla, los bloqueos de nivel de fila se concentran en un bloqueo de tabla. La transacción contiene los bloqueos de nivel de fila, con lo que se reduce la carga de trabajo de bloqueo.

El bloqueo dinámico tiene las siguientes ventajas:

- Administración simplificada de bases de datos, ya que los administradores ya no se tienen que preocupar de ajustar los umbrales de concentración de bloqueos.
- Rendimiento aumentado, ya que SQL Server reduce la carga de trabajo del sistema mediante bloqueos adecuados a la tarea.

Opciones de bloqueo en el nivel de tabla

Objetivo del tema
Presentar las opciones de bloqueo de tablas.

Explicación previa
En la práctica diaria, no se debe especificar la opción de bloqueo de tabla.

- Úselas con precaución
- Puede especificar una o más opciones de bloqueo para una tabla
- Utilice la parte *sugerenciasDeOptimizador* de la cláusula FROM de las instrucciones SELECT o UPDATE
- Suplanta las opciones de bloqueo en el nivel de sesión

Aunque SQL Server utiliza una arquitectura de bloqueos dinámicos para seleccionar el mejor bloqueo para el cliente, se pueden especificar opciones de bloqueo en el nivel de tabla. Una sugerencia de tabla puede especificar un método para que lo utilice el optimizador de consultas con una tabla específica y para una instrucción.

Nota Utilice las opciones de bloqueo en el nivel de tabla con precaución, sólo después de comprender en detalle el funcionamiento de la aplicación y cuando haya determinado que el bloqueo que solicita seguirá siendo, con el tiempo, mejor que el bloqueo utilizado por SQL Server.

Las siguientes características se aplican a las opciones de bloqueo en el nivel de tabla:

- Puede especificar una o varias opciones de bloqueo para una tabla.
- Utilice la parte *sugerenciasDeOptimizador* de la cláusula FROM de las instrucciones SELECT o UPDATE.
- Estas opciones de bloqueo suplantan las opciones correspondientes del nivel de sesión (nivel de aislamiento de las transacciones) que se hayan especificado previamente con la instrucción SET.

La siguiente tabla describe las opciones de bloqueo de tabla.

Opción	Descripción
HOLDLOCK SERIALIZABLE REPEATABLEREAD READCOMMITTED READUNCOMMITTED NOLOCK	Controlan el comportamiento de bloqueo de una tabla y suplantan los bloqueos que se utilizarían para exigir el nivel de aislamiento de la transacción actual.
ROWLOCK PAGLOCK TABLOCK TABLOCKX	Especifican el tamaño y el tipo de los bloqueos que se utilizarán para una tabla.
READPAST	Salta las filas bloqueadas.
UPDLOCK	Utiliza bloqueos de actualización en lugar de bloqueos compartidos.

Interbloqueos

Objetivo del tema
Ilustrar por qué y cómo se produce un interbloqueo.

Explicación previa
Un interbloqueo se produce cuando dos transacciones tienen bloqueos sobre objetos diferentes y cada transacción solicita un bloqueo sobre el objeto bloqueado por la otra transacción.

- Cómo SQL Server termina los interbloqueos
- Cómo minimizar los interbloqueos
- Cómo personalizar la configuración de tiempo de espera de bloqueo

Sugerencia
Asegúrese de resaltar la diferencia entre los interbloqueos y los bloqueos.

Un interbloqueo se produce cuando dos transacciones tienen bloqueos sobre objetos diferentes y cada transacción solicita un bloqueo sobre el objeto bloqueado por la otra transacción. Las dos transacciones tienen que esperar a que la otra libere el bloqueo.

Un interbloqueo puede ocurrir cuando varias transacciones de duración prolongada se ejecutan simultáneamente en la misma base de datos. También pueden ocurrir interbloqueos como resultado del orden en el que el optimizador procesa una consulta compleja, como una combinación, en la que no se puede controlar el orden del proceso.

Cómo SQL Server termina los interbloqueos

Para terminar automáticamente los interbloqueos, SQL Server completa una de las transacciones. El proceso que utiliza SQL Server se encuentra en la lista siguiente.

1. Deshace la transacción del sujeto del interbloqueo.

   En un interbloqueo, SQL Server da prioridad a la transacción que ha estado en proceso durante más tiempo; dicha transacción prevalece. SQL Server deshace la transacción en la que ha invertido menos tiempo.
2. Notifica a la aplicación sujeto del interbloqueo (con el mensaje número 1205).
3. Cancela la petición actual del sujeto del interbloqueo.
4. Permite que continúe la otra transacción.

Importante En entornos multiusuario, todos los clientes deben comprobar con regularidad si reciben el mensaje número 1205, que indica que la transacción se ha deshecho. Si se encuentra el mensaje 1205, la aplicación tiene que volver a ejecutar la transacción.

Cómo minimizar los interbloqueos

Aunque los interbloqueos no se pueden eliminar siempre, puede reducir el riesgo de que aparezcan si tiene en cuenta las siguientes directrices:

- Utilice los recursos en la misma secuencia para todas transacciones. Por ejemplo, si es posible, haga referencia a las tablas en el mismo orden en todas las transacciones que hagan referencia a más de una tabla.
- Abrevie las transacciones minimizando el número de pasos.
- Abrevie la duración de las transacciones evitando las consultas que afecten a muchas filas.

Cómo personalizar la configuración de tiempo de espera de bloqueo

Si una transacción se bloquea mientras espera un recurso y se produce un interbloqueo, SQL Server terminará una de las transacciones participantes sin tiempo de espera.

Si no se produce ningún interbloqueo, SQL Server bloquea la transacción que solicita el bloqueo hasta que la otra transacción libere el bloqueo. De forma predeterminada, no hay ningún período de tiempo de espera obligatorio que tenga en cuenta SQL Server. La única forma de probar si el recurso que se desea bloquear ya está bloqueado es tener acceso a los datos, lo que podría dar lugar a que estuviera bloqueado indefinidamente.

LOCK_TIMEOUT permite que una aplicación establezca el tiempo máximo que una instrucción debe esperar en un recurso bloqueado antes de que la instrucción bloqueada se cancele automáticamente. La cancelación no deshace ni cancela la transacción. La aplicación debe detectar el error para tratar la situación de tiempo de espera y tomar una medida correctiva, como volver a enviar la transacción o deshacerla.

El comando KILL termina un proceso de usuario según el Id. de proceso de servidor (spid).

Presentación de información acerca de los bloqueos

Objetivo del tema
Mostrar la distinta información acerca de los bloqueos activos que se puede obtener.

Explicación previa
Para presentar un informe de los bloqueos activos, ejecute el procedimiento almacenado de sistema sp_lock.

- Ventana Actividad actual
- Procedimiento almacenado de sistema sp_lock
- Analizador de SQL
- Monitor de sistema de Windows 2000
- Información adicional

Sugerencia
Muestre la ventana Actividad actual del Administrador corporativo de SQL Server y del Analizador de SQL.

Normalmente, para presentar un informe de los bloqueos activos se utiliza el Administrador corporativo de SQL Server o el procedimiento almacenado de sistema **sp_lock**. Puede utilizar el Analizador de SQL para obtener información acerca de un conjunto específico de transacciones. También puede utilizar el Monitor de sistema de Microsoft Windows® 2000 para presentar el historial de bloqueos de SQL Server.

Ventana Actividad actual

Utilice la ventana Actividad actual del Administrador corporativo de SQL Server para presentar información acerca de la actividad actual de bloqueo. Puede ver la actividad del servidor por usuario, detallar la actividad por conexión y la información de bloqueo por objeto.

Procedimiento almacenado de sistema sp_lock

El procedimiento almacenado de sistema **sp_lock** devuelve información acerca de los bloqueos activos en SQL Server.

Sintaxis

EXECUTE **sp_lock**

Resultado

El resultado será similar al siguiente.

spid	dbid	ObjId	IndId	Type	Resource	Mode	Status
12	5	0	0	DB		S	GRANT
12	5	0	0	DB		S	GRANT
12	2	0	0	EXT	1:280	X	GRANT
12	5	0	0	PAG	1:528	IX	GRANT
12	5	981578535	0	RID	1:528:0	X	GRANT
12	1	5575058	0	TAB		IS	GRANT
12	5	981578535	0	TAB		IX	GRANT
13	1	0	0	DB		S	GRANT

Las cuatro primeras columnas hacen referencia a varios Id.: Id. de proceso del servidor (spid), Id. de base de datos (dbid), Id. de objeto (ObjId) e Id. de número de identificación de índice (IndId).

La columna **Type** muestra el tipo de recurso que está bloqueado actualmente. Los tipos de recursos pueden ser: DB (base de datos), EXT (extensión), TAB (tabla), KEY (clave), PAG (página) o RID (identificador de fila).

La columna **Resource** tiene información acerca del tipo de recurso que está bloqueado. Una descripción de recurso de 1:528:0 indica que la fila número 0 de la página número 528 del archivo 1 tiene aplicado un bloqueo.

La columna **Mode** describe el tipo de bloqueo que se está aplicando al recurso. Los tipos de bloqueo son: compartido (S), exclusivo (X), de intención (I), de actualización (U) o de esquema (Sch).

La columna **Status** muestra si el bloqueo se ha obtenido (GRANT), está bloqueado en espera de que termine otro proceso (WAIT) o está en proceso de conversión (CNVRT).

Analizador de SQL

El Analizador de SQL es una herramienta que supervisa las actividades del servidor. Para recopilar información acerca de diversos eventos, puede crear trazas, que proporcionan un perfil detallado de los eventos del servidor. Puede utilizar este perfil para analizar y resolver los problemas de recursos del servidor, supervisar los intentos de inicio de sesión y las conexiones, y corregir problemas de interbloqueo.

Monitor de sistema de Windows 2000

Puede ver la información de bloqueos de SQL Server con el Monitor de sistema. Utilice los objetos **SQL Server: administrador de bloqueos** y **SQL Server: bloqueos**.

Información adicional

Para buscar información acerca de los bloqueos y la actividad actual del servidor, puede consultar las tablas del sistema **syslockinfo**, **sysprocesses**, **sysobjects**, **systables** y **syslogins**, o puede ejecutar el procedimiento