Implementación de vistas

Contenido

Notas para el instructor

Este módulo proporciona a los alumnos la información que necesitan para crear y utilizar vistas. Las vistas ofrecen la posibilidad de almacenar una consulta predefinida como un objeto en la base de datos para usarse posteriormente. Proporcionan un método adecuado para ocultar los datos confidenciales o la complejidad del diseño de las bases de datos, y para facilitar información sin que el usuario tenga que escribir o ejecutar instrucciones de Transact-SQL.

El módulo define las vistas y sus ventajas. A continuación, describe cómo crear vistas y proporciona ejemplos de proyecciones y combinaciones. Estos ejemplos ilustran el modo de incluir columnas calculadas y funciones integradas en las definiciones de vistas. A continuación, el módulo cubre las restricciones en la modificación de datos mediante vistas. La última sección explica cómo las vistas pueden mejorar el rendimiento.

En la práctica los alumnos crearán y probarán vistas, incluidas las vistas con definiciones cifradas. Los alumnos también realizarán cambios en las tablas de origen mediante una vista y examinarán definiciones de vistas.

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

- Describir el concepto de vista.
- Enumerar las ventajas de las vistas.
- Definir una vista con la instrucción CREATE VIEW.
- Modificar datos mediante vistas.
- Optimizar el rendimiento mediante vistas.

Introducción

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

Explicación previa
En este módulo aprenderá a crear, utilizar y mantener vistas de datos.

- Introducción a las vistas
- Ventajas de las vistas
- Definición de vistas
- Modificación de datos mediante vistas
- Optimización del rendimiento mediante vistas

Este módulo define las vistas y sus ventajas. A continuación, describe cómo crear vistas y proporciona ejemplos de proyecciones y combinaciones. Estos ejemplos ilustran el modo de incluir columnas calculadas y funciones integradas en las definiciones de vistas. A continuación, el módulo cubre las restricciones en la modificación de datos mediante vistas. La última sección explica cómo las vistas pueden mejorar el rendimiento.

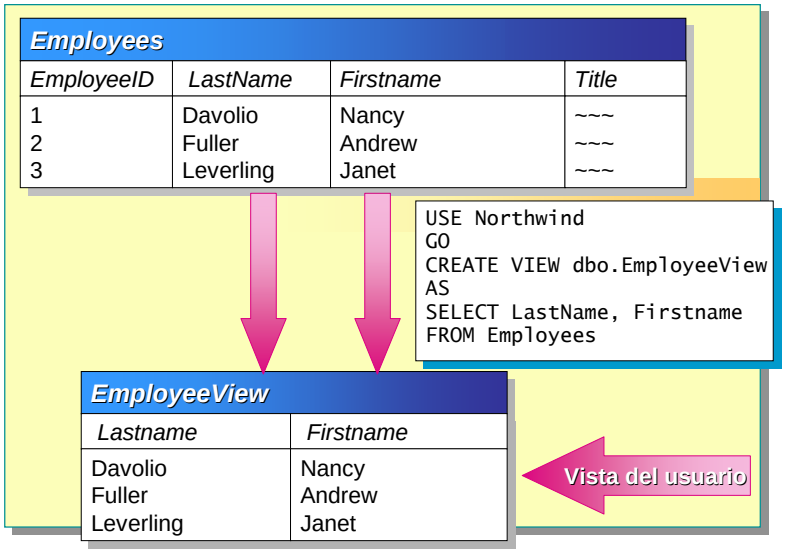
Después de completar este módulo, el alumno será capaz de:

- Describir el concepto de vista.
- Enumerar las ventajas de las vistas.
- Definir una vista con la instrucción CREATE VIEW.
- Modificar datos mediante vistas.
- Optimizar el rendimiento mediante vistas.

Introducción a las vistas

Objetivo del tema
Presentar el concepto de vista y exponer un ejemplo.

Explicación previa
Una vista es un modo alternativo de ver los datos de una o varias tablas.



Sugerencia
Hasta este punto del curso, hemos escrito consultas "ad hoc". Con las vistas comenzaremos la explicación de la forma en que las consultas se almacenan como objetos (vistas, procedimientos almacenados y desencadenadores) en la base de datos.

Una vista ofrece la posibilidad de almacenar una consulta predefinida como un objeto en una base de datos para usarse posteriormente. Las tablas consultadas en una vista se denominan *tablas base*. Salvo algunas excepciones, es posible dar nombre a cualquier instrucción SELECT y almacenarla como una vista. Algunos ejemplos habituales de vistas son los siguientes:

- Un subconjunto de las filas o columnas de una tabla base.
- Una unión de dos o más tablas base.
- Una combinación de dos o más tablas base.
- Un resumen estadístico de una tabla base.
- Un subconjunto de otra vista o alguna combinación de vistas y tablas base.

Ejemplo

En este ejemplo se crea la vista **dbo.EmployeeView** en la base de datos **Northwind**. La vista muestra dos columnas de la tabla **Employees**.

```
USE Northwind
GO
CREATE VIEW dbo.EmployeeView
AS
SELECT LastName, Firstname
FROM employees
```

Consulta

```
SELECT * from EmployeeView
```

Resultado

```
Lastname                                           FirstName
Davolio                                            Nancy
Fuller                                             Andrew
Leverling                                          Janet
.
.
.
(9 filas afectadas)
```

Ventajas de las vistas

Objetivo del tema
Explicar por qué puede ser conveniente crear o utilizar vistas.

Explicación previa
Las vistas ofrecen diversas ventajas.

- Centrar el interés en los datos de los usuarios
  - Centrarse sólo en los datos importantes o adecuados
  - Limitar el acceso a los datos confidenciales
- Enmascarar la complejidad de la base de datos
  - Ocultar el diseño de la base de datos compleja
  - Simplificar las consultas complejas, incluyendo las consultas distribuidas a datos heterogéneos
- Simplificar la administración de los permisos de usuario
- Mejorar el rendimiento
- Organizar los datos para exportarse a otras aplicaciones

Las vistas ofrecen diversas ventajas, como la capacidad de centrar el interés en los datos de los usuarios, enmascarar su complejidad, simplificar la administración de los permisos y organizar los datos para exportarlos a otras aplicaciones.

Centrar el interés en los datos de los usuarios

Las vistas crean un entorno controlado que permite el acceso a datos específicos mientras se oculta el resto. Los datos innecesarios, confidenciales o inadecuados pueden quedar fuera de la vista. Los usuarios pueden tratar la presentación de los datos en una vista de forma similar a como lo hacen en una tabla. Además, con los permisos adecuados y algunas restricciones, también pueden modificar los datos que muestra la vista.

Sugerencia
Señale que las vistas de esquema de información permiten a SQL Server presentar los datos del sistema de modo coherente, incluso cuando las tablas del sistema han sufrido cambios significativos.

Enmascarar la complejidad de la base de datos

Las vistas ocultan al usuario la complejidad del diseño de la base de datos. De este modo, los programadores pueden cambiar el diseño sin afectar a la interacción entre el usuario y la base de datos. Además, el usuario verá una versión más descriptiva de los datos, con nombres más fáciles de comprender que los términos crípticos que a menudo se utilizan en las bases de datos.

Las consultas complejas, como las consultas distribuidas de datos heterogéneos, también pueden enmascararse con vistas. De este modo el usuario consulta la vista, en lugar de escribir la consulta o ejecutar una secuencia de comandos.

Simplificar la administración de los permisos de usuario

En lugar de conceder a los usuarios permisos para consultar columnas específicas de las tablas base, los propietarios de las bases de datos pueden conceder permisos para que el usuario sólo pueda consultar los datos a través de vistas. Además, de esta forma se protegen los cambios en el diseño de las tablas base subyacentes. Los usuarios pueden continuar consultando la vista sin interrupciones.

Mejorar el rendimiento

Las vistas le permiten almacenar los resultados de consultas complejas. Otras consultas pueden utilizar estos resultados resumidos. Las vistas también permiten dividir datos. Es posible colocar divisiones individuales en equipos distintos.

Organizar los datos para exportarse a otras aplicaciones

Puede crear una vista basada en una consulta compleja que combine dos o más tablas y, a continuación, exportar los datos a otra aplicación para analizarlos.

Definición de vistas

Objetivo del tema
Presentar una sección acerca del trabajo con las vistas.

Explicación previa
Esta sección describe cómo crear, modificar y quitar vistas.

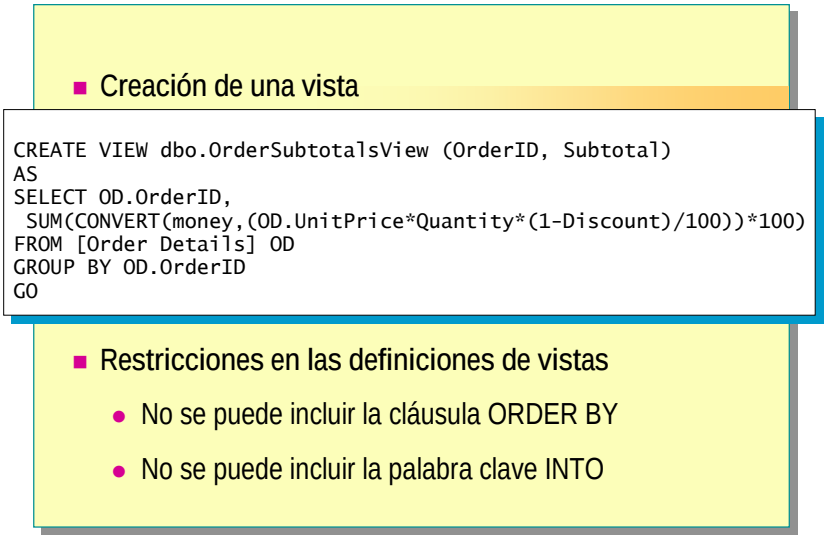
- Creación de vistas
- Ejemplo: Vista de tablas combinadas
- Modificación y eliminación de vistas
- Evitar la interrupción de las cadenas de pertenencia
- Ubicación de la información de definición de vistas
- Ocultación de la definición de las vistas

Esta sección describe cómo crear, modificar y quitar vistas. También explica cómo evitar la interrupción de las cadenas de pertenencia, ocultar las definiciones de las vistas y obtener información acerca de las vistas de una base de datos.

Creación de vistas

Objetivo del tema
Explicar cómo crear y quitar vistas.

Explicación previa
Ahora que hemos definido las vistas, explicaremos cómo crear una.



Puede crear vistas con el Asistente para creación de vistas, con el Administrador corporativo de SQL Server o con Transact-SQL. Las vistas sólo se pueden crear en la base de datos actual.

Creación de una vista

Sugerencia
Recomiende a los alumnos que establezcan una convención de denominación coherente para distinguir las vistas de las tablas y que especifiquen **dbo** como nombre del propietario.

Al crear una vista, Microsoft® SQL Server™ versión 2000 comprueba la existencia de los objetos a los que se hace referencia en su definición. El nombre de la vista debe ajustarse a las normas para los identificadores. Opcionalmente, es posible especificar un nombre de propietario para la vista. Debe establecer una convención de denominación coherente para distinguir las vistas de las tablas. Por ejemplo, puede agregar la palabra “vista” como sufijo de cada objeto vista que cree. De este modo podrá distinguir fácilmente entre objetos similares (tablas y vistas) al consultar la vista **INFORMATION_SCHEMA.TABLES**.

Sintaxis

CREATE VIEW *propietario.nombreVista* [(*columna* [,*n*])]
[WITH {ENCRYPTION | SCHEMABINDING | VIEW_METADATA} [,n]]
AS
instrucciónSelect

[WITH CHECK OPTION]

Sugerencia
Es posible tener permiso para crear una vista y no tener permiso de acceso a las tablas subyacentes. Sin embargo, una vista creada en estas condiciones no devolverá ningún conjunto de resultados.

Para poder ejecutar la instrucción CREATE VIEW es necesario ser miembro de la función de administradores del sistema (**sysadmin**), de la función propietario de la base de datos (**db_owner**) o de la función administrador de lenguaje de definición de datos (**db_ddladmin**), o bien tener el permiso CREATE VIEW. También es necesario tener el permiso SELECT en todas las tablas o vistas a las que la vista haga referencia.

Para evitar situaciones en las que el propietario de una vista 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 la base de datos. Especifique siempre el usuario **dbo** como propietario al crear el objeto pues, de lo contrario, usted, es decir, su nombre de usuario, será el propietario.

El contenido de una vista se especifica con una instrucción SELECT. Con algunas excepciones, las vistas pueden ser tan complejas como se requiera. Debe especificar los nombres de columna en las situaciones siguientes:

Sugerencia
Los nombres de columna pueden especificarse de dos maneras: en la instrucción SELECT con alias de columnas o en la instrucción CREATE VIEW.

- Alguna de las columnas de la vista se deriva de una expresión aritmética, de una función integrada o de una constante.
- Hay columnas con el mismo nombre en las tablas que se van a combinar.

Importante Al crear vistas, es importante probar la instrucción SELECT que las define para asegurarse de que SQL Server devuelve el conjunto de resultados esperado. Una vez escrita y probada la instrucción SELECT, y habiendo comprobado sus resultados, puede crear la vista.

Restricciones en las definiciones de vistas

Al crear vistas, tenga presentes las restricciones siguientes:

- La instrucción CREATE VIEW no puede incluir las cláusulas COMPUTE o COMPUTE BY. La instrucción CREATE VIEW no puede incluir la palabra clave INTO.
- La instrucción CREATE VIEW puede incluir la cláusula ORDER BY, sólo si se utiliza la palabra clave TOP.
- Las vistas no pueden hacer referencia a tablas temporales.
- Las vistas no pueden hacer referencia a más de 1.024 columnas.
- La instrucción CREATE VIEW no puede combinarse con otras instrucciones de Transact-SQL en un mismo lote.

Ejemplo 1

Éste es un ejemplo de una vista que crea una columna (**Subtotal**) que calcula los subtotales de un pedido a partir de las columnas **UnitPrice**, **Quantity** y **Discount**.

```
CREATE VIEW dbo.OrderSubtotalsView (OrderID, Subtotal)
AS
SELECT OD.OrderID,
   SUM(CONVERT
    (money,(OD.UnitPrice*Quantity*(1-Discount)/100))*100)
FROM [Order Details] OD
GROUP BY OD.OrderID
GO
```

Ejemplo 2

En este ejemplo se consulta la vista para ver los resultados.

```
SELECT * FROM OrderSubtotalsView
```

Resultado

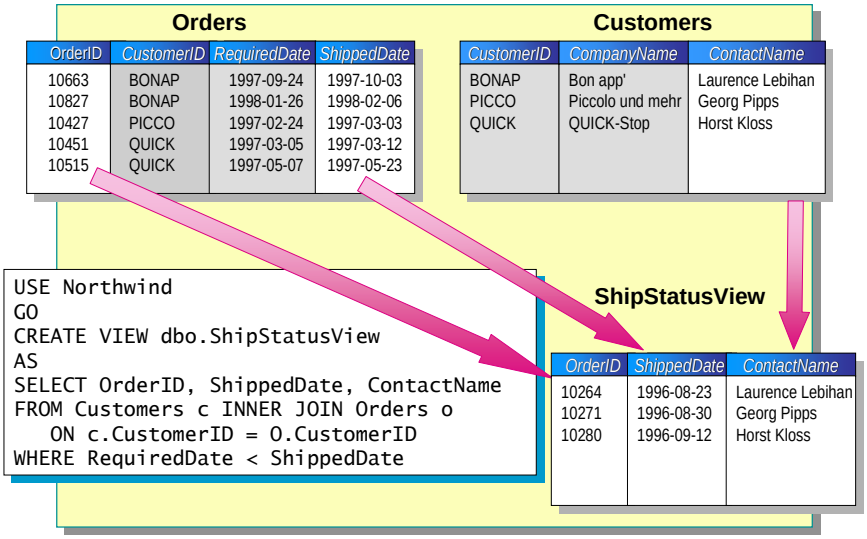
OrderID	Subtotal
10271	48.0000
10977	2233.0000
10440	4924.1400
.	
.	
.	

```
(830 filas afectadas)
```

Ejemplo: Vista de tablas combinadas

Objetivo del tema
Ofrecer un ejemplo de una vista de dos o más tablas combinadas.

Explicación previa
Es posible crear vistas de varios tipos. Uno de ellos es un subconjunto de columnas, como vimos en una diapositiva anterior. Otro tipo, más habitual, es la vista de dos o más tablas combinadas.



A menudo se crean vistas para conseguir una forma más práctica de ver información centralizada de dos o más tablas combinadas.

Ejemplo 1

En este ejemplo, **ShipStatusView** combina las tablas **Customers** y **Orders**.

```
USE Northwind
GO
CREATE VIEW dbo.ShipStatusView
AS
SELECT OrderID, ShippedDate, ContactName
FROM Customers c INNER JOIN Orders o
   ON c.CustomerID = O.CustomerID
WHERE RequiredDate < ShippedDate

SELECT * FROM ShipStatusView
```

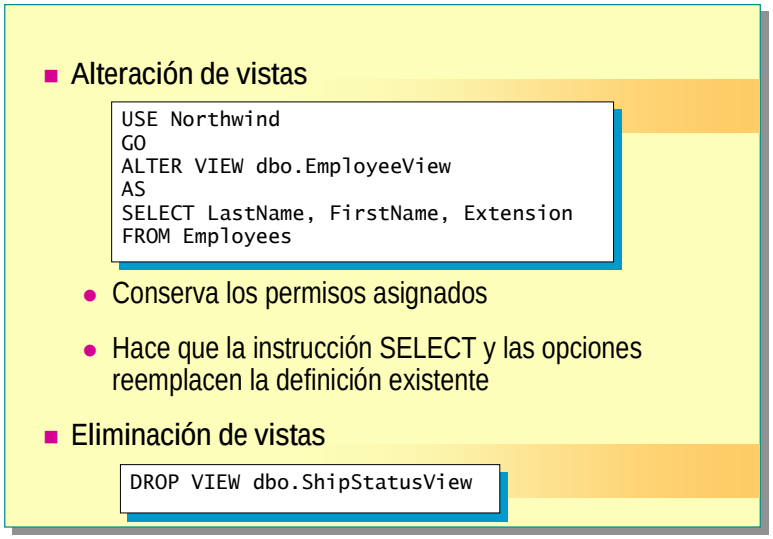
Resultado

```
OrderID     ShippedDate             ContactName
10264       1996-08-23              Maria Larsson
10271       1996-08-30              Art Braunschweiger
10280       1996-09-12              Christina Berglund
.
.
.
(37 filas afectadas)
```

Modificación y eliminación de vistas

Objetivo del tema
Indicar cómo alterar una vista.

Explicación previa
Es posible alterar una vista existente.



A menudo, las vistas se alteran como respuesta a las peticiones de información adicional por parte de los usuarios o a causa de cambios en la definición de las tablas subyacentes. Para alterar una vista puede quitarla y volverla a crear, o bien puede ejecutar la instrucción ALTER VIEW.

Alteración de vistas

La instrucción ALTER VIEW cambia la definición de una vista, incluidas las vistas de índices, sin afectar a los procedimientos almacenados o desencadenadores dependientes. Esto le permite conservar los permisos de la vista. Esta instrucción está sujeta a las mismas restricciones que CREATE VIEW. Si quita una vista y la vuelve a crear, deberá volver a asignar los permisos asociados a ella.

Sintaxis

```
ALTER VIEW propietario.nombreVista
  [(columna [,...n ])]
  [WITH  {ENCRYPTION | SCHEMABINDING | VIEW_METADATA} [,...n]]
  AS
  instrucciónSelect
  [WITH CHECK OPTION]
```

Nota Si utiliza las opciones WITH CHECK OPTION, WITH ENCRYPTION, WITH SCHEMABINDING o WITH VIEW_METADATA al crear la vista, deberá incluirlas también en la instrucción ALTER VIEW para que la vista conserve la funcionalidad que ofrecen.

Ejemplo

En el ejemplo siguiente se altera **EmployeeView** para agregar la columna **Extension**.

```
USE Northwind
GO
ALTER VIEW dbo.EmployeeView
AS
SELECT LastName, FirstName, Extension
FROM employees
```

Consulta

```
SELECT * from  dbo.EmployeeView
```

Resultado

Lastname	FirstName	Extension
Davolio	Nancy	5467
Fuller	Andrew	3457
Leverling	Janet	3355

```
.
.
.
(9 filas afectadas)
```

Sugerencia
Al crear la vista, la lista de columnas se almacena en la tabla syscolumns.

Nota Si define una vista con una instrucción SELECT * y, a continuación, agrega columnas a la estructura de las tablas subyacentes y, por tanto, altera dicha estructura, las nuevas columnas no aparecerán en la vista. Cuando en la instrucción CREATE VIEW se seleccionan todas las columnas, la lista de columnas sólo se interpreta en el momento de crear la vista. Para incluir las columnas nuevas en la vista, es necesario alterar ésta.

Eliminación de vistas

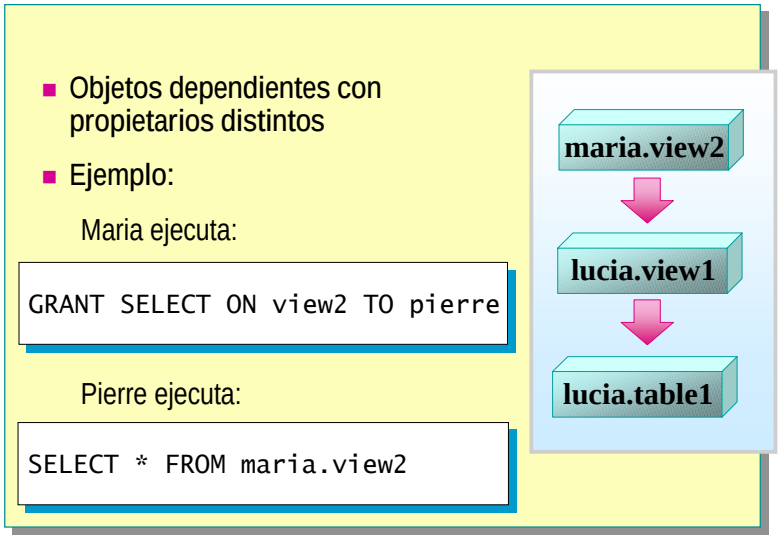
Si ya no necesita una vista, puede quitar su definición de la base de datos con la instrucción DROP VIEW. Al quitar una vista se quita su definición y todos los permisos que tenga asignados. Además, si los usuarios consultan vistas que hagan referencia a la vista quitada, obtendrán un mensaje de error. Sin embargo, al quitar una tabla que hace referencia a una vista, ésta no se quita automáticamente. Es necesario quitarla de forma explícita.

Nota El permiso para quitar una vista corresponde a su propietario y no es transferible. Es la configuración predeterminada. Sin embargo, el administrador del sistema o el propietario de la base de datos pueden quitar cualquier objeto si especifican el nombre de su propietario en la instrucción DROP VIEW.

Evitar la interrupción de las cadenas de pertenencia

Objetivo del tema
Presentar el concepto de cadenas de pertenencia.

Explicación previa
Para evitar la interrupción de las cadenas de pertenencia, el usuario **dbo** debe ser el propietario de todas las vistas.



SQL Server permite que el propietario del objeto original conserve el control de qué usuarios tienen autorización de acceso al mismo.

Objetos dependientes con propietarios distintos

Las definiciones de las vistas dependen de objetos subyacentes (vistas o tablas). Estas dependencias pueden considerarse como una cadena de pertenencia. Si el propietario de una vista también posee los objetos subyacentes, sólo tendrá que conceder permisos en la vista. Cuando se utiliza el objeto, sólo se comprueban los permisos de la vista.

Para evitar interrumpir las cadenas de pertenencia, el usuario **dbo** debe ser el propietario de todas las vistas. Al utilizar el objeto, se comprobarán los permisos de cada objeto dependiente con un propietario distinto.

Ejemplo

Maria crea la vista **view2**. Con la instrucción siguiente, concede a Pierre permiso para consultarla.

Sintaxis

```
GRANT select ON view2 TO pierre
```

Sin embargo, **maria.view2** depende de un objeto (**view1**) propiedad de otra usuaria (Lucia). Los permisos de cada objeto dependiente se comprueban con un propietario distinto.

Pierre consulta la vista con la instrucción siguiente:

Sintaxis

```
SELECT * FROM maria.view2
```

Como **maria.view2** depende de **lucia.view1**, SQL Server comprueba los permisos de **maria.view2** y **lucia.view1**. Si Lucia ha concedido previamente a Pierre permisos en **view1**, Pierre tendrá acceso a los datos. En caso contrario, se le denegará el acceso, porque Lucia mantiene el control sobre qué usuarios tienen acceso a los objetos que ella crea.

Ubicación de la información de definición de vistas

Objetivo del tema
Describir el modo de ver la información de las vistas.

Explicación previa
Al crear, alterar o quitar tablas, es conveniente tener información acerca de las vistas de la base de datos.

- Ubicación de las definiciones de vistas
  - No disponible si la vista fue creada con la opción WITH ENCRYPTION
- Ubicación de las dependencias de una vista
  - Muestra los objetos de los que depende una vista
  - Muestra los objetos que dependen de una vista

Para alterar la definición de una vista o comprender cómo se obtienen sus datos a partir de las tablas base, puede ser necesario ver su definición.

Ubicación de las definiciones de las vistas

Para ubicar la información de definición de las vistas puede utilizar el Administrador corporativo de SQL Server o consultar las vistas y tablas del sistema siguientes.

Vista de esquema de información o tabla del sistema	Muestra información acerca de
INFORMATION_SCHEMA.TABLES o **sysobjects**	Nombres de vistas.
INFORMATION_SCHEMA.VIEW_TABLE_USAGE o **sysdepends**	Nombres de objetos base.
INFORMATION_SCHEMA.VIEWS o **syscomments**	Definición de vistas.
INFORMATION_SCHEMA.VIEW_COLUMN_USAGE o **syscolumns**	Columnas definidas en una vista.

Nota **INFORMATION_SCHEMA.VIEW_TABLE_USAGE** e **INFORMATION_SCHEMA.VIEW_COLUMN_USAGE** sólo muestran información referida a su nombre de usuario.

Para mostrar el texto utilizado para crear una vista puede usar el Administrador corporativo de SQL Server, consultar **INFORMATION_SCHEMA.VIEWS** o ejecutar el procedimiento almacenado de sistema **sp_helptext** con el nombre de la vista como parámetro.

Sintaxis

sp_helptext *nombreObjeto*

Sugerencia
Muestre cómo obtener la información de dependencia con el Administrador corporativo de SQL Server.

Ubicación de las dependencias de una vista

Para obtener un informe de las tablas o vistas de las que depende una vista, y de los objetos que dependen de una vista determinada puede usar el Administrador corporativo de SQL Server o ejecutar el procedimiento almacenado de sistema **sp_depends**.

Antes de eliminar cualquier objeto, debe ver las dependencias. Antes de alterar o quitar una tabla, ejecute el procedimiento almacenado de sistema **sp_depends** para determinar si hay objetos que hagan referencia a ella.

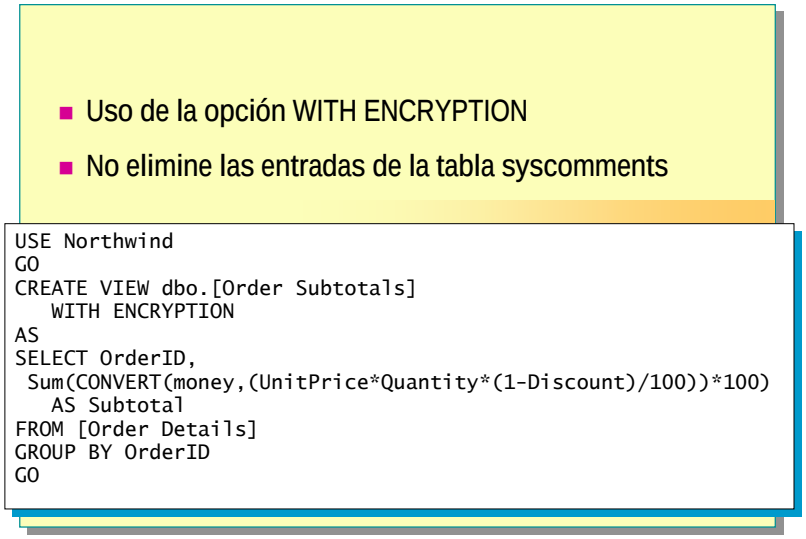
Sintaxis

sp_depends *nombreObjeto*

Ocultación de la definición de las vistas

Objetivo del tema
Explicar el modo de cifrar las definiciones de las vistas.

Explicación previa
Puede cifrar la definición de una vista para ocultar los detalles de las tablas base consultadas por ella.



Debido a que los usuarios pueden mostrar la definición de una vista mediante el Administrador corporativo de SQL Server, por medio de consultas a **INFORMATION_SCHEMA.VIEWS** o consultas a la tabla del sistema **syscomments**, puede que desee ocultar ciertas definiciones de vistas.

Uso de la opción WITH ENCRYPTION

Puede cifrar las entradas de la tabla **syscomments** que contengan el texto de la instrucción CREATE VIEW. Para ello, especifique la opción WITH ENCRYPTION en la definición de la vista.

Antes de cifrar una vista, asegúrese de guardar su definición (secuencia de comandos) en un archivo. Para descifrar el texto de una vista debe quitarla y volverla a crear, o alterarla y utilizar la sintaxis original.

Ejemplo

En este ejemplo se crea **dbo.[Order Subtotals**] con la opción WITH ENCRYPTION para que la definición de la vista quede oculta.

```
USE Northwind
GO
CREATE VIEW dbo.[Order Subtotals]
   WITH ENCRYPTION
AS
SELECT OrderID,
 Sum(CONVERT(money,(UnitPrice*Quantity*(1-Discount)/100))*100)
   AS Subtotal
FROM  [Order Details]
GROUP BY OrderID
```

No elimine las entradas de la tabla syscomments

Cuando, por motivos de seguridad, la definición de una vista no deba estar a disposición de los usuarios, cifre la vista. No elimine nunca las entradas de la tabla del sistema **syscomments**. Si lo hiciera, no podría utilizar la vista e impediría que SQL Server volviera a crear la vista cuando se actualizara la base de datos con una versión más reciente de SQL Server.

Modificación de datos mediante vistas

Objetivo del tema
Exponer consideraciones que los alumnos deben tener presentes al modificar datos mediante vistas.

Explicación previa
Los cambios en los datos realizados a través de vistas modifican las tablas subyacentes.

- No pueden afectar a más de una tabla subyacente
- No pueden afectar a ciertas columnas
- Pueden provocar errores si afectan a columnas a las que la vista no hace referencia
- Se comprueba si se ha especificado WITH CHECK OPTION

Las vistas no mantienen una copia aparte de los datos sino que muestran el conjunto de resultados de una consulta de una o varias tablas base. Por ello, al modificar los datos de una vista, en realidad se modifica la tabla base.

Con algunas restricciones, es posible insertar, actualizar o eliminar libremente datos de una tabla a través de una vista. En general, la vista debe estar basada en una sola tabla y no debe incluir funciones de agregado o cláusulas GROUP BY en la instrucción SELECT.

Específicamente, las modificaciones realizadas a través de vistas:

- No pueden afectar a más de una tabla subyacente.

  Puede modificar vistas derivadas de dos o más tablas, pero cada actualización o modificación sólo puede afectar a una de ellas.

- No pueden afectar a ciertas columnas.

  SQL Server no permite cambiar las columnas que son resultado de un cálculo, como las que contienen valores calculados, funciones integradas o funciones de agregado de filas.

- Pueden provocar errores si afectan a columnas a las que la vista no hace referencia.

  Por ejemplo, al insertar una fila en una vista definida en una tabla que contiene columnas no incluidas en la vista y que no permiten valores NULL o contienen valores predeterminados, se producirá un error.

- Se comprueba si se ha especificado WITH CHECK OPTION en la definición de la vista.

  La opción WITH CHECK OPTION obliga a todas las instrucciones de modificación de datos que se ejecutan contra la vista a cumplir ciertos criterios. Estos criterios se especifican en la instrucción SELECT que define la vista. Si los valores cambiados están fuera del intervalo válido de la definición de la vista, SQL Server rechazará la modificación.

Optimización del rendimiento mediante vistas

Objetivo del tema
Describir cómo optimizar el rendimiento mediante vistas.

Explicación previa
Esta sección describe cómo optimizar el rendimiento mediante vistas.

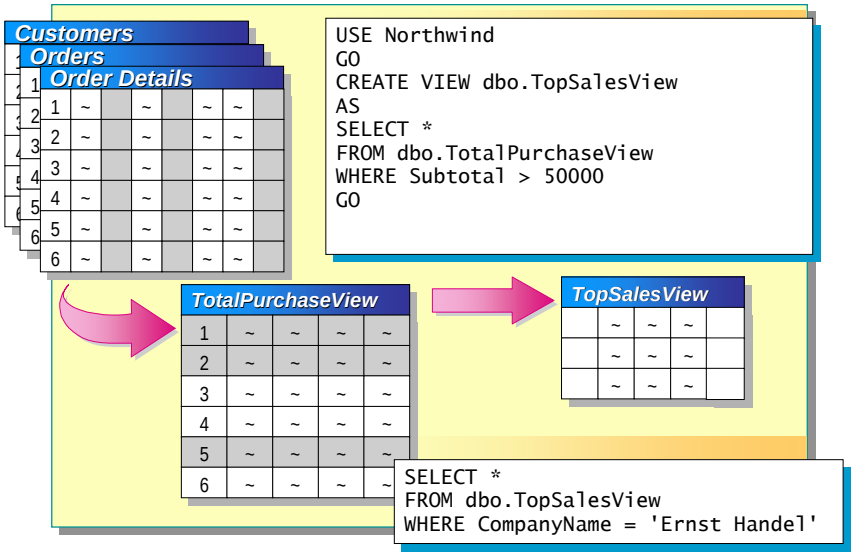
- Consideraciones acerca del rendimiento
- Uso de vistas indizadas
- Uso de vistas para dividir datos

Esta sección describe las consideraciones acerca del rendimiento en el uso de vistas y cómo las vistas le permiten optimizar el rendimiento al almacenar los resultados de consultas complejas y dividir los datos.

Consideraciones acerca del rendimiento

Objetivo del tema
Mostrar el modo de crear una vista que contiene otra vista.

Explicación previa
Si desea aprovechar el trabajo que ya ha hecho, puede crear una vista de otra vista. Sin embargo, debe tener en cuenta los posibles problemas de rendimiento que ello implica.



Cuando se anidan en una vista otras vistas que combinan varias tablas y evalúan expresiones complejas, el origen inmediato de los posibles problemas de rendimiento puede ser difícil de determinar. En consecuencia, puede ser conveniente crear definiciones de vistas separadas, en lugar de anidarlas.

Ejemplo

En el ejemplo siguiente, **TopSalesView** consulta un subconjunto de filas de **TotalPurchaseView**.

Sugerencia
Pregunta: ¿hasta dónde puede anidarse una vista de otra vista?

Respuesta: el único límite son los recursos disponibles. En general, no se deben anidar más de tres niveles si no se desea ocultar posibles problemas de rendimiento.

```
USE Northwind
GO
CREATE VIEW dbo.TopSalesView
AS
SELECT *
FROM dbo.TotalPurchaseView
WHERE Subtotal > 50000
GO
```

La definición de la vista **dbo.TopSalesView** oculta la complejidad de la consulta subyacente utilizada para crear TotalPurchaseView, que combina tres tablas base.

```
USE Northwind
GO
CREATE VIEW dbo.TotalPurchaseView
AS
SELECT CompanyName, Sum(CONVERT(money,
   (UnitPrice*Quantity*(1-Discount)/100))*100) AS Subtotal
FROM Customers c INNER JOIN Orders o
     ON c.CustomerID=o.CustomerID
   INNER JOIN [Order Details] od
     ON o.OrderID = od.OrderID
GROUP BY CompanyName
GO
```

Consulta

Si los usuarios encuentran problemas de rendimiento al ejecutar la consulta siguiente para obtener los libros disponibles en francés, el origen del problema no será evidente.

```
SELECT *
FROM dbo.TopSalesView
WHERE CompanyName = 'Ernst Handel'
```

Resultado

CompanyName	Subtotal
Ernst Handel	104874.98

(1 fila(s) afectada(s))

Uso de vistas indizadas

Objetivo del tema
Describir las vistas indizadas.

Explicación previa
Es posible crear índices en las vistas.

- Las vistas indizadas almacenan el conjunto de resultados en la base de datos
- Creación de una vista indizada
- Recomendaciones para la creación de vistas indizadas

  Utilizar si:

  - El rendimiento mejora el costo del incremento en el mantenimiento
  - Los datos subyacentes no se actualizan con frecuencia
  - Las consultas realizan una gran cantidad de combinaciones y operaciones de agregado
- Restricciones en la creación de vistas indizadas

Es posible crear índices en las vistas. Una *vista indizada* almacena el conjunto de resultados de una vista en la base de datos. Debido a que la velocidad de recuperación es rápida, puede utilizar vistas indizadas para mejorar el rendimiento de las consultas.

Creación de una vista indizada

Para crear una vista indizada implemente un índice único y agrupado (UNIQUE CLUSTERED) en una vista. Los resultados de la vista se almacenan en las páginas de nivel de hoja del índice agrupado. Una vez creado el índice UNIQUE CLUSTERED, puede crear otros índices en esa vista.

Una vista indizada refleja automáticamente las modificaciones realizadas en los datos de la tablas base. A medida que los datos cambian, el índice UNIQUE CLUSTERED se actualiza.

Recomendaciones para la creación de vistas indizadas

El optimizador de consultas determina automáticamente si una consulta dada se beneficiará del uso de una vista indizada. Puede determinar este suceso si la consulta no hace referencia a la vista indizada. Como recomendación general, permita al optimizador de consultas determinar cuándo utilizar vistas indizadas.

Con el Asistente para optimización de índices, puede mejorar extraordinariamente su capacidad para determinar la mejor mezcla de índices y vistas indizadas con el fin de optimizar el rendimiento de las consultas.

Cree vistas indizadas cuando:

- La mejora en el rendimiento por el aumento de velocidad al recuperar los resultados supere el costo del incremento en el mantenimiento.
- Los datos subyacentes no se actualicen con frecuencia.
- Las consultas realicen una cantidad significativa de combinaciones y operaciones de agregado que procesen muchas filas o que sean realizadas frecuentemente por muchos usuarios.

Restricciones en la creación de vistas indizadas

Considere las siguientes instrucciones al crear vistas indizadas:

- El primer índice que crea en una vista debe ser un índice agrupado único.
- Debe crear la vista con la opción SCHEMABINDING.
- La vista puede hacer referencia a tablas base, pero no a otras vistas.
- Para hacer referencia a tablas y funciones definidas por el usuario debe utilizar nombres compuestos por dos partes.
- Las conexiones posteriores deben tener la misma configuración de opciones para utilizar la vista indizada.

Nota Para estar seguro de que puede indizar una vista debe utilizar la propiedad **IsIndexable** de la función OBJECTPROPERTY.

Uso de vistas para dividir datos

Objetivo del tema
Presentar las vistas divididas.

Explicación previa
Puede utilizar vistas para dividir datos entre varias bases de datos o copias de SQL Server.

- Puede utilizar las vistas para dividir los datos en varios servidores o instancias de SQL Server
- Cómo utiliza SQL Server las vistas para dividir datos
- Cómo las vistas divididas mejoran el rendimiento

Puede utilizar vistas para dividir datos entre varias bases de datos o copias de SQL Server y así mejorar el rendimiento.

Cómo utiliza SQL Server las vistas para dividir datos

Puede utilizar el operador UNION dentro de una vista para combinar los resultados de dos o más consultas de tablas distintas en un solo conjunto de resultados. Esta operación aparece ante el usuario como una sola tabla llamada *vista dividida*. Las vistas divididas se pueden actualizar aunque hagan referencia a varias tablas.

Pueden estar basadas en datos procedentes de muchos orígenes heterogéneos, como servidores remotos, y no sólo de tablas de la misma base de datos. Esto permite distribuir el procesamiento de la base de datos entre un grupo de servidores. El grupo de servidores puede hacer frente a las necesidades de procesamiento de grandes aplicaciones de comercio electrónico o centros de datos de empresa.

Cómo las vistas divididas mejoran el rendimiento

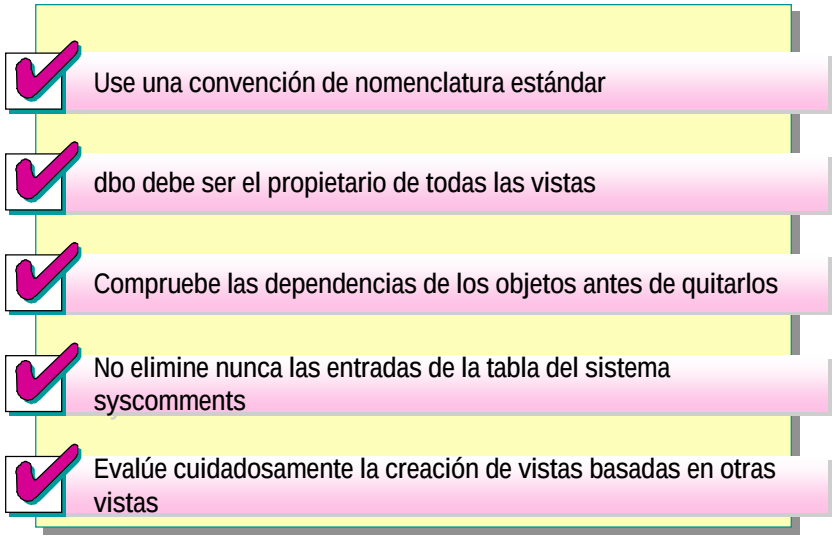
Si las tablas de una vista dividida se encuentran en distintos servidores o en un equipo con varios procesadores, cada tabla implicada en la consulta puede recorrerse en paralelo, con lo que se mejora el rendimiento de la consulta. Además, las tareas de mantenimiento, como la regeneración de índices o la copia de seguridad de una tabla, pueden ejecutarse a mayor velocidad.

Nota No se puede crear un índice en una vista dividida. La definición de vista requerida para crear la vista indizada sólo permite nombres de dos partes; una vista dividida requiere el uso de nombres de tres o cuatro partes, como, *NombreServidor.nombreBaseDeDatos.nombrePropietario.nombreObjeto*.

Procedimientos recomendados

Objetivo del tema
Presentar algunas recomendaciones para el uso de las vistas.

Explicación previa
Las siguientes son algunas recomendaciones para utilizar las vistas.



Las recomendaciones siguientes le ayudarán a utilizar y administrar las vistas de bases de datos:

- Debe establecer una convención de denominación coherente para distinguir las vistas de las tablas.
- Especifique **dbo** como propietario al crear las vistas. El **dbo** debe poseer todos los objetos a los que se hace referencia en la definición de vista. Con ello no tendrá que especificar el nombre del propietario al consultar la vista, ya que el propietario de la base de datos es el propietario predeterminado. Además, el propietario de la base de datos tiene permisos en todos los objetos subyacentes de la base de datos, lo que evita la posibilidad de que se interrumpan las cadenas de propiedad.
- Compruebe las dependencias de los objetos antes de quitarlos de la base de datos. Ejecute el procedimiento almacenado de sistema **sp_depends** o muestre las dependencias en el Administrador corporativo de SQL Server para asegurarse de que ningún objeto depende del que va a quitar.
- Nunca elimine entradas de la tabla del sistema **syscomments**. Si la aplicación requiere que la definición sea invisible para otros usuarios, utilice la opción WITH ENCRYPTION en la instrucción CREATE VIEW o ALTER VIEW. Asegúrese de guardar la secuencia de comandos de la definición antes de cifrar la secuencia.
- Evalúe cuidadosamente si debe crear vistas basadas en otras vistas. Puede ocultar su complejidad y provocar problemas de rendimiento.

En los Libros en pantalla de SQL Server encontrará información adicional acerca de estos temas.

Tema	Buscar en
CREATE VIEW	“crear vista”
ALTER VIEW	“alterar vista”
DROP VIEW	“quitar vista”
Cadenas de pertenencia interrumpidas	“cadena de pertenencia”
Generación de secuencias de comandos SQL	“documentación y secuencias de comandos de bases de datos”