

INFORMACION RELATIVA A LAS CONEXIONES

Toda la información relativa a las conexiones se guarda en la base de datos, en las tablas de sistema de la base de datos Máster. Es posible conocer esta información consultando las vistas de sistema que se enumeran a continuación.

Se desaconsejan las acciones directas en las tablas para garantizar la ejecución correcta de los scripts en las diferentes versiones de SQL Server. Un script que se prueba y valida funcionará en las siguientes versiones de SQL Server.

- `sys.server_principals`

Lista de las conexiones definidas en el servidor.

- `sys.sql_logins`

Lista de las conexiones definidas con el modo de seguridad de SQL Server.

- `sys.server_permissions`

Lista de permisos asignados a las conexiones directamente en el servidor.

- `sys.server_role_members`

Lista de los roles de servidor y de las conexiones que los usan.

- `sys.system_components_surface_area_configuration`

Devuelve la lista de los objetos de sistema que se pueden ver o no, en función de la configuración efectuada. De esta manera, si la visibilidad está habilitada, un usuario puede ver los metadatos de un objeto, incluso si no tiene permisos para ello.

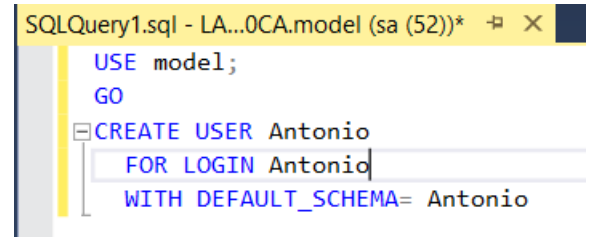
El `usuario guest` (invitado) es un usuario particular que se puede definir en las bases de datos de usuario. Este nombre de usuario se utilizará en las conexiones al servidor que no estén mapeadas con un usuario de base de datos. La gestión de este usuario es la misma que la de un usuario cualquiera de la base de datos.

1. Crear un usuario

La creación de un usuario de base de datos va a permitir unir una conexión a un usuario y, por tanto, autorizar la utilización de la base de datos.

SQL Server Management Studio

Para crear un usuario de base de datos es necesario, desde el explorador de objetos, situarse sobre la base de datos en la que se va a operar y realizar las acciones siguientes:



```
SQLQuery1.sql - LA...OCA.model (sa (52))*  
USE model;  
GO  
CREATE USER Antonio  
FOR LOGIN Antonio  
WITH DEFAULT_SCHEMA= Antonio
```

- Localizar el nodo Security y situarse sobre el nodo Logins.
- Desde el menú contextual asociado al nodo Logins, seleccionar New Login.
- Seleccionar la conexión asociada al usuario y posteriormente definir el nombre de este último.
- Finalmente, es posible indicar los roles de base de datos asignados a este usuario.

La instrucción CREATE USER permite definir cuentas de usuarios en la base de datos.

Sintaxis

```
CREATE USER nombreUsuario  
[ FOR {LOGIN nombreConexión |  
CERTIFICATE nombreCertificado|  
ASYMMETRIC KEY nombreClaveAsimétrica}}]  
[ WITH DEFAULT_SCHEMA = nombreEsquema ]
```

2. Información

Para obtener el conjunto de información relativa a un usuario de base de datos, es necesario proceder de la siguiente manera:

- Desde el explorador de objetos, situarse sobre la base de datos.
- Localizar los nodos **Security - Logins**.

- Situar sobre el usuario para el que se desea obtener información y solicitar que se muestren las propiedades a partir del menú contextual asociado.

Es posible obtener la información de los usuarios de base de datos consultando la vista `sys.database_principals`

El procedimiento `sp_who` permite conocer los usuarios actualmente conectados y los procesos en curso.

Para cada conexión, el procedimiento `sp_who` permite conocer, entre otra, la siguiente información:

- El nombre de conexión utilizado (`loginame`).
- El nombre del equipo desde el que se establece la conexión (`hostname`).
- El nombre de la base de datos actual (`dbname`).

3. Establecer la lista de conexiones y usuarios

Para realizar esta lista, lo más sencillo es usar las vistas de sistema con objeto de hacer el proceso automático. Esta consulta se basa en las vistas de sistema **`sys.database_principals`** de la base de datos, para obtener la lista de usuarios que se han definido en ella, y la vista **`sys.server_principals`**, que proporciona la lista de conexiones definidas.

El siguiente script permite construir esta lista completa. Para esto, se utiliza un cursor que recorre todas las bases de datos. Para limitar la lista que se devuelve y ganar en claridad, solo se tienen en cuenta las bases de datos de usuario, lo que explica la exclusión de la base de datos Master, model, tempdb y msdb.

Para cada valor que devuelve el cursor, se ejecuta la consulta anterior. Sin embargo, como una parte de la consulta es dinámica (el nombre de la base de datos), es necesario construir la consulta como una cadena de caracteres (lo que se hace dando valor a la variable `@consulta`).

Para ejecutar esta consulta dinámica, se utiliza el procedimiento almacenado `sp_executesql`.

```

DECLARE cLasBases cursor for select name from sys.databases where name
not in(
'master', 'model', 'tempdb', 'msdb');
DECLARE @nombreBase sysname,
DECLARE @consulta nvarchar(500)
BEGIN
    OPEN cLasBases;
    FETCH cLasBases INTO nombreBase;

While (@@fetch_status = 0) begin
    set @consulta = 'SELECT ''' + @nombreBase + ''' as Base,'
    set @consulta = @consulta + 's.name' as Conexion, p.name as Usuario '
    set @consulta = @consulta + 'FROM MASTER.SYS.DATABASE_PRINCIPALS p '
    set @consulta = @consulta + 'INNER JOIN ' + @nombreBase + '
.sys.server_principals s'
    set @consulta = @consulta + 'ON s.sid = p.sid'
    exec sp_executesql @consulta
    FETCHcLasBases INTO @nombreBase;
End
Close cLasBases;
DEALLOCATE cLasBases
End;

```

4. Modificación

Es posible modificar un usuario de base de datos para cambiar su nombre o el esquema asociado a él.

Es necesario situarse en el explorador de objetos, sobre la base de datos afectada por la modificación de la cuenta de usuario, y realizar las acciones siguientes:

- Localizar el nodo **Security** y posteriormente **Logins**.
- Situarse sobre la cuenta de usuario que hay que modificar.
- Visualizar la ventana de propiedades desde la opción **Properties**, situada en el menú contextual asociado al usuario.

La instrucción ALTER USER permite realizar esta operación.

```
ALTER USER nombreUsuario  
WITH NAME=nuevoNombreUsuario,  
DEFAULT_SCHEMA = nombreNuevoEsquema
```

5. Eliminación

Teniendo en cuenta que los usuarios no tienen objetos, su eliminación siempre es posible sin ningún riesgo de pérdida de datos. La eliminación de un usuario no implica la eliminación del esquema asociado al usuario.

Es necesario situarse, desde el explorador de objetos, sobre la base de datos afectada por la eliminación de la cuenta de usuario y realizar las acciones siguientes:

- Localizar el nodo Security y a continuación Logins.
- Situar sobre la cuenta de usuario que hay que eliminar.
- Seleccionar la opción Delete desde el menú contextual asociado a la cuenta de usuario.

La eliminación no es posible sobre las cuentas de usuario predefinidas, a saber: dbo, guest, INFORMATION_SCHEMA y sys.

Administración de los esquemas

El objetivo de los esquemas es separar los usuarios de base de datos de los objetos que pueden crear. Sin embargo, los objetos no se dejan tal cual, sino que se agrupan de manera lógica en el esquema. De esta manera, es posible definir un esquema como un conjunto lógico de objetos de una base de datos.

Permitiendo esta agrupación lógica de tablas, vistas, funciones y procedimientos en la base de datos, los esquemas ofrecen una mejor legibilidad sin complicar la estructura.

Los esquemas facilitan la compartición de información entre varios usuarios sin perder el nivel de seguridad.

Los esquemas facilitan la administración de los permisos de uso de los objetos que los componen, ya que es posible asignar los permisos de uso de los objetos directamente a nivel de estos.

Es responsabilidad del propietario del esquema gestionar los permisos de uso del esquema y de los objetos presentes en él. El propietario del esquema puede transferir la gestión de las tablas/vistas a otros usuarios.

Para acceder a los objetos situados fuera de su esquema predeterminado, un usuario de base de datos debe utilizar el nombre completo de un objeto, es decir, nombreEsquema.nombreObjeto.

Cuando se usa un nombre corto (simplemente el nombre del objeto, sin indicar el nombre del esquema), SQL Server busca ese objeto únicamente en el esquema actual del usuario.

1. Creación

Para crear un esquema de base de datos, es necesario situarse sobre la base de datos afectada y realizar las acciones siguientes desde el explorador de objetos:

- Localizar el modo Security y situarse sobre el nodo Schemas.
- Desde el menú contextual asociado al nodo Schemas, seleccionar New Schema.

b. Transact

```
SQL CREATE SCHEMA nombreEsquema
AUTHORIZATION nombrePropietario|
[definiciónDeTablas | definiciónDeVistas
| gestiónDePrivilegios]
```

La opción **AUTHORIZATION** permite definir el usuario de base de datos que es propietario del esquema. Un mismo usuario de base de datos puede ser el propietario de varios esquemas.

El propietario de un esquema no ha de tener necesariamente como esquema predeterminado uno del que es propietario. También es posible crear las tablas y las vistas de un esquema desde la construcción del propio esquema.

2. Modificación

La modificación de un esquema consiste en modificar los objetos contenidos en este esquema. Cuando un objeto se transfiere de un esquema a otro, los permisos relativos a estos objetos se pierden. La transferencia de un objeto entre dos esquemas es posible en el interior de una misma base de datos.

a. SQL Server Management Studio

Para modificar un esquema de base de datos, es necesario situarse sobre la base de datos en cuestión y realizar las acciones siguientes desde el explorador de objetos:

- Localizar el nodo **Security** y situarse sobre el nodo **Schemas**.
- Situarse sobre el esquema que hay que modificar.
- Visualizar la ventana de propiedades desde la opción **Properties**, disponible en el menú contextual asociado al esquema

3. Eliminación

La eliminación de un esquema es posible solo para los esquemas que no contienen objetos. Por ejemplo, si el esquema contiene tablas o vistas, es necesario eliminarlas o transferirlas a otro esquema antes de poder eliminar el esquema actual.

a. SQL Server Management Studio

Para eliminar un esquema de base de datos, es necesario situarse sobre la base de datos en cuestión y realizar las acciones siguientes desde el explorador de objetos:

- Localizar el nodo **Security** y situarse sobre el nodo **Schemas**.
- Situarse sobre el esquema que hay que eliminar.
- Seleccionar la opción **Delete** desde el menú contextual asociado al esquema.

4. La información relativa a los esquemas

Es posible encontrar la información de cada esquema consultando la vista sys.schemas. Esta vista tiene las siguientes columnas:

Name Representa el nombre del esquema. No puede haber dos esquemas con el mismo nombre en una base de datos.

Schema_id Identificador numérico que se asigna al esquema.

Principal_id Identificador numérico que corresponde al propietario del esquema.

Administración de los permisos

Todos los usuarios de base de datos, incluido guest (el invitado), pertenecen al grupo public. Los derechos que se detallan a continuación se pueden asignar directamente a public. Los derechos se organizan de manera jerárquica con relación a los elementos del servidor a los que se puede dar seguridad.

SQL Server gestiona los privilegios con tres tipos de palabras claves:

- GRANT
- REVOKE
- DENY

Es decir, un privilegio se puede asignar (**GRANT**) o retirar (**REVOKE**) si se ha asignado previamente. La instrucción **DENY** permite prohibir el uso de un privilegio particular, aunque el privilegio en cuestión haya sido asignado directamente o por medio de un rol.

1. Permisos de uso de las instrucciones

Los permisos principales de instrucciones disponibles son:

- CREATE DATABASE
- CREATE PROCEDURE
- CREATE FUNCTION
- CREATE TABLE
- BACKUP DATABASE
- CREATE VIEW
- BACKUP LOG

La asignación de privilegios

se efectúa utilizando la instrucción GRANT, cuya sintaxis se detalla a continuación. GRANT permiso [...], TO usuario[,...] [WITH GRANT OPTION]

permiso

Nombre del/de los permiso(s) relativo(s) a esta autorización. También es posible utilizar la palabra clave ALL en lugar de citar explícitamente el permiso o los permisos asignados. Sin embargo, este término ALL no permite asignar los privilegios de ejecución de todas las instrucciones, sino simplemente sobre las instrucciones para crear las bases de datos, tablas, procedimientos, funciones y para efectuar las copias de seguridad de la base de datos y del diario.

usuario Nombre del usuario o de los usuarios de base de datos que recibe(n) los permisos.

WITH GRANT OPTION Si el permiso se recibe con este privilegio, el usuario puede asignar el permiso a otros usuarios de base de datos.

REVOKE [GRANT OPTION FOR] permiso [...]
FROM usuario [...]
[CASCADE]

GRANT OPTION FOR Si el permiso se ha asignado con el privilegio de administración GRANT OPTION, es posible, por medio de esta opción, retirar simplemente el parámetro de administración.

permiso Lista de permisos que hay que retirar.

usuario Lista de usuarios afectados por esta eliminación.

CASCADE Si el permiso retirado se ha asignado con el privilegio de administración WITH GRANT OPTION, la opción CASCADE permite retirar el privilegio a los usuarios que lo han recibido a través del usuario afectado por la eliminación inicial del privilegio.

b. Prohibir

La instrucción DENY permite prohibir a un usuario la utilización de un privilegio, aunque haya recibido el permiso directamente o por su pertenencia a un grupo.

2. Derechos de utilización de los objetos

permiten fijar las operaciones (lectura, modificación, adición o eliminación) que el usuario puede realizar sobre los datos contenidos en una tabla, o bien dar el derecho de ejecución de un procedimiento almacenado. Estos derechos son, en general, gestionados por el propietario del objeto.

En lo que respecta al derecho de lectura de los datos contenidos en una tabla (utilización de la instrucción SELECT), es posible indicar aquellas columnas que el usuario puede visualizar. Por defecto, se trata de todas las columnas.

Los principales derechos de utilización de los objetos se refieren a las tablas, las vistas y los procedimientos almacenados, y corresponden a las sentencias:

- INSERT
- UPDATE
- DELETE
- SELECT
- EXECUTE (que solo se utiliza para los procedimientos almacenados).

Las instrucciones SELECT y UPDATE se pueden limitar a algunas columnas de la tabla o la vista. Sin embargo, es preferible no utilizar demasiado esta posibilidad, ya que da lugar a más trabajo administrativo de gestión de los derechos. Es preferible pasar por una vista o por un procedimiento almacenado para limitar el uso de la tabla.

a. Autorizar

Los privilegios de utilización de los objetos se pueden gestionar en dos niveles con SQL Server Management Studio:

A nivel usuario, lo que permite averiguar las posibilidades de trabajo que tiene cada usuario.

A nivel de los objetos, para saber cuáles son los usuarios que pueden utilizar el objeto en cuestión y cómo pueden utilizarlo.

En cualquiera de los dos casos, las autorizaciones se gestionan mediante la ventana de propiedades.

GRANT {ALL [PRIVILEGES]

```
|permiso[(columna,[...])] [...]}  
ON objeto[,...]  
TO usuario [...]  
[WITH GRANT OPTION ]
```

ALL Permite asignar todos los privilegios de utilización del objeto. Los privilegios asignados son siempre en función del tipo del objeto afectado por esta asignación del privilegio.

PRIVILEGES La palabra clave se ha añadido para respetar la norma ANSI-92. No aporta ninguna modificación en lo que respecta a la utilización de la palabra clave ALL.

permiso Permite especificar la operación o las operaciones que se asignarán a los usuarios.

objeto Corresponde al nombre completo del objeto o de los objetos sobre los que se realiza la asignación del privilegio de utilización.

usuario Corresponde al usuario o, más concretamente, a la entidad de seguridad que va a poder beneficiarse del privilegio o de los privilegios.

WITH GRANT OPTION El privilegio se asigna con una opción de administración que autoriza al beneficiario del privilegio a asignar este mismo privilegio a otros usuarios de la base de datos.

```
REVOKE [GRANT OPTION FOR ]  
{ ALL [PRIVILEGES]|permiso[(columna,[...])] [...]}  
ON objeto [(columna [...])]  
{FROM | TO} usuario [...]
```

GRANT OPTION FOR

Con esta opción, es posible retirar simplemente el privilegio de administración del privilegio.

permiso

Permite precisar el permiso o los permisos afectados por la acción de retirada de permisos. Igual que para la asignación, es posible indicar las columnas afectadas por la operación, aunque la gestión de este nivel de derecho es laboriosa.

objeto Es necesario precisar el nombre completo de los objetos en la base de datos; es decir: nombreEsquema.nombreObjeto.

FROM, TO Estos dos términos son sinónimos. Normalmente se utiliza TO al asignar un privilegio y FROM al eliminar un privilegio.

usuario Se trata de la entidad de seguridad a la que se retira el privilegio. Lo más habitual es que esta entidad de seguridad sea un usuario, pero se puede tratar de un rol, por ejemplo.

CASCADE Al retirar un permiso asignado con el privilegio de administración, esta opción permite retirarlo en cascada; es decir, efectuar la eliminación del permiso a todas las entidades de seguridad que lo han recibido a través de aquella a la que se le retira el permiso

b. Prohibir

La prohibición de utilizar un permiso para usar un objeto es una instrucción más fuerte que la eliminación, porque puede aplicarse a una entidad de seguridad, incluso aunque esta última no haya recibido todavía el privilegio de utilización del objeto de manera directa o no.

3. **Derechos a nivel de la base de datos** Los privilegios asignados a nivel de la base de datos ofrecen la posibilidad al usuario que los recibe de realizar las acciones autorizadas sobre el conjunto de la base de datos. Por ejemplo, el

privilegio ALTER ANY USER permite a un usuario crear, eliminar y modificar cualquier cuenta de la base de datos.

A nivel de la base de datos, es posible asignar privilegios a un usuario pero también a un esquema, un assembly o un objeto service bróker

Los privilegios que es posible asignar a este nivel son:

ALTER	CREATE DATABASE
ALTER ANY APPLICATION ROLE	CREATE DATABASE DDL EVENT NOTIFICATION
ALTER ANY ASSEMBLY	CREATE DEFAULT
ALTER ANY ASYMMETRIC KEY	CREATE FULLTEXT CATALOG
ALTER ANY CERTIFICATE	CREATE FUNCTION
ALTER ANY CONTRACT	CREATE MESSAGE TYPE
ALTER ANY DATABASE DDL TRIGGER	CREATE PROCEDURE
ALTER ANY DATABASE EVENT NOTIFICATION	CREATE QUEUE
ALTER ANY DATASPACE	CREATE REMOTE SERVICE BINDING
ALTER ANY FULLTEXT CATALOG	CREATE ROLE
ALTER ANY MESSAGE TYPE	CREATE ROUTE
ALTER ANY REMOTE SERVICE BINDING	CREATE RULE
ALTER ANY ROLE	CREATE SCHEMA
ALTER ANY ROUTE	CREATE SERVICE
ALTER ANY SCHEMA	CREATE SYMMETRIC KEY
ALTER ANY SERVICE	CREATE SYNONYM
ALTER ANY SYMMETRIC KEY	CREATE TABLE
ALTER ANY USER	CREATE TYPE
AUTHENTICATE	CREATE VIEW
BACKUP DATABASE	CREATE XML SCHEMA COLLECTION
BACKUP LOG	DELETE
CHECKPOINT	EXECUTE
CONNECT	INSERT
CONNECT REPLICATION	REFERENCES
CONTROL	SELECT
CREATE AGGREGATE	SHOWPLAN
CREATE ASSEMBLY	SUBSCRIBE QUERY NOTIFICATIONS
CREATE ASYMMETRIC KEY	TAKE OWNERSHIP
CREATE CERTIFICATE	UPDATE
CREATE CONTRACT	VIEW DATABASE STATE
ALTER ANY DATABASE EVENT SESSION	VIEW DEFINITION

4. Derechos a nivel del servidor

SQL Server permite asignar privilegios a nivel del servidor. Estos privilegios no se asignan a un usuario de base de datos, sino a una conexión.

Como para los derechos de utilización de los objetos y de las instrucciones, es posible asignar estos privilegios con una opción de administración. Así, la conexión que tiene este derecho puede asignar este mismo derecho a una o varias conexiones adicionales.

Los diferentes derechos que es posible asignar a nivel del servidor son:

ADMINISTER BULK OPERATIONS	CONNECT SQL
ALTER ANY CONNECTION	CONTROL SERVER
ALTER ANY CREDENTIAL	CREATE ANY DATABASE

ALTER ANY DATABASE	CREATE DDL EVENT NOTIFICATION
ALTER ANY ENDPOINT	CREATE ENDPOINT
ALTER ANY EVENT NOTIFICATION	CREATE TRACE EVENT NOTIFICATION
ALTER ANY LINKED SERVER	EXTERNAL ACCESS ASSEMBLY
ALTER ANY LOGIN	IMPERSONATE ANY LOGIN
ALTER RESOURCES	SELECT ALL USERS SECURABLES
ALTER SERVER STATE	SHUTDOWN
ALTER SETTINGS	UNSAFE ASSEMBLY
ALTER TRACE	VIEW ANY DATABASE
AUTHENTICATE SERVER	VIEW ANY DEFINITION
CONNECT ANY DATABASE	VIEW SERVER STATE

5. Consultar las vistas de sistema

Consultando las vistas de sistema del diccionario, y más concretamente las que se ocupan de la seguridad, es posible ver las diferentes autorizaciones y prohibiciones relativas a las diferentes entidades de seguridad.

Las principales vistas son:

sys.database_permissions, que permite listar las diferentes autorizaciones de utilización de los privilegios.

sys.database_principals, que permite listar las diferentes cuentas de seguridad.

sys.database_principals

Esta vista lista todas las entidades de seguridad definidas localmente en la base de datos. Las principales columnas de esta vista son:

name: nombre de la entidad de seguridad. Este nombre es único dentro de la base de datos.

principal_id: identificador numérico que permite identificar de manera única una entidad de seguridad en la base de datos.

type: código relativo al tipo del contexto de seguridad: usuario de SQL, usuario de Windows, rol, grupo de Windows...

type_desc: descripción para comprender el tipo mencionado en la columna anterior.

default_schema_name: nombre del esquema asociado por defecto a la entidad de seguridad.

create_date: fecha de creación de la entidad de seguridad

. modify_date: fecha de la última modificación sobre la entidad de seguridad.

owning_principal_id: identificador del contexto de seguridad que tiene la cuenta en cuestión. Todos, a excepción de los roles de base de datos, deben pertenecer a dbo.

sid: o Security Identifier. En este campo se informa de si la entidad de seguridad está asociada a un externo.

is_fixed_role: este atributo es 1 cuando la entidad de seguridad se asocia a un rol fijo predefinido sobre el servidor.

sys.database_permissions

Esta vista permite obtener la información de los diferentes permisos asignados a la base de datos. Existe un registro por cada permiso asignado a cada entidad de seguridad. En caso de administración de permisos a nivel de columna, existe un registro para cada columna afectada por el permiso.

Las columnas más importantes de esta vista son:

class: código de la clasificación del permiso, es decir, si se trata de un permiso a nivel de base de datos que se aplica al uso de un objeto o columna, etc.

class_desc: descripción de la clasificación del permiso.

mayor_id: identificador del objeto sobre el que se asigna el permiso.

minor_id: identificador de la columna sobre la que se asigna el permiso.

grantee_principal_id: identificador del contexto de seguridad al que concierne el permiso.

grantor_principal_id: identificador del contexto de seguridad en el origen del permiso. type: tipo del permiso.

permission_name: nombre del privilegio al que concierne el permiso.

state: código relativo al estado del permiso.

state_desc: descripción del estado actual del permiso: GRANT, REVOKE, DENY o bien GRANT_WITH_GRANT_OPTION.

El resto de las vistas de sistema que se pueden consultar para obtener los datos de los diferentes tipos de permisos asignados son:

sys.database_role_members: lista de beneficiarios (principal_id) de un rol de base de datos. **sys.master_key_passwords:** información de cada clave principal de la base de datos creada con el procedimiento sp_control_dbmasterkey_password.

Contexto de ejecución

El contexto de ejecución está directamente relacionado con la conexión y el usuario de bases de datos asociado. La conexión de ejecución permite establecer la lista de posibles acciones y aquellas que no se pueden realizar. Esta lista se

crea a partir de los permisos asignados a los usuarios directamente o a través de los roles.

En algunos casos puede ser necesario y deseable modificar el contexto de ejecución para aprovechar los permisos extendidos, pero solo en el ámbito de un script, procedimiento o función.

Asociada al contexto de ejecución, es necesario entender bien la noción de encadenamiento de propiedades en SQL Server.

En primer lugar, cuando un usuario accede a los objetos de los que es propietario, esto no representa ningún problema porque no hay ruptura de encadenamiento de propiedades. Este caso es relativamente raro, ya que los usuarios de las bases de datos que crean objetos raramente los utilizan a diario.

Ejemplo

el usuario Ángel recibe de María el permiso para usar (SELECT) una vista de María. La consulta SELECT de definición de la vista hace referencia a una tabla también de María. Ángel no tiene ningún permiso sobre esta tabla y podrá utilizar esta vista sin problemas, ya que los dos objetos (vista y tabla) tienen el mismo propietario.

EXECUTE AS

Con esta instrucción se puede solicitar la conexión a la base de datos usando una conexión diferente a la actual. Esta instrucción se puede ejecutar de manera autónoma en un script Transact SQL o como una cláusula durante la creación de un procedimiento, función o trigger.

Para los procedimientos, funciones o triggers, la cláusula EXECUTE AS ofrece mucha flexibilidad en términos de programación y permiten a un usuario realizar acciones para las que no tiene permisos. Para el desarrollador, el cambio de contexto de ejecución también permite garantizar que no se impedirá la correcta ejecución del código debido a un problema de permisos de acceso a los datos.

EJEMPLO: durante la ejecución de un script Transact SQL, la instrucción EXECUTE AS permite realizar de manera puntual operaciones que necesiten permisos extendidos, mientras que el resto del script no lo necesita.

```
EXECUTE AS {LOGIN|USER} = 'identificador' |CALLER;  
EXECUTE AS {LOGIN|USER} = 'identificador' |CALLER WITH NO  
REVERT;
```

Otra versión del comando EXECUTE_AS permite indicar el contexto de ejecución de un procedimiento, función o trigger de base de datos. El ámbito de este cambio de contexto de ejecución está limitado a la ejecución del módulo.

Procedimiento o función o trigger

```
EXECUTE AS {CALLER|SELF|OWNER|'nombreUsuario'}
```

CALLER Se tiene en cuenta el contexto del que ejecuta el módulo.

SELF Se tiene en cuenta el contexto del usuario que ha creado o modificado el módulo.

OWNER Se tiene en cuenta el contexto del propietario del módulo.

'nombreUsuario' Permite indicar el contexto de usuario que se debe utilizar.

SETUSER

Al contrario que la instrucción EXECUTE AS, que no modifica la conexión inicial del servidor, la instrucción SETUSER permite cambiar de conexión en un script Transact SQL. Es decir, se cierra el contexto de ejecución actual y se abre uno nuevo. No es posible volver al primer contexto de ejecución.

Original_Login

Esta función permite determinar el nombre exacto de la conexión que se utiliza inicialmente para conectarse al servidor. Conocer este nombre solo es interesante cuando el contexto de ejecución es diferente de la conexión inicial. El contexto de ejecución se puede modificar con la instrucción EXECUTE AS.

REVERT

Después de un cambio de contexto de ejecución con la instrucción EXECUTE AS, la instrucción REVERT permite volver al contexto de ejecución que había cuando se cambió con EXECUTE AS.

Los roles

Los roles son los conjuntos de permisos. Estos conjuntos existen a tres niveles distintos: servidor, base de datos y aplicación. Los roles permiten agrupar los derechos y gestionar más fácilmente los diferentes usuarios y las conexiones. Siempre es preferible asignar los derechos a los roles y posteriormente asignar los roles a los usuarios. Con una estructura como esta, la adición y la modificación de permisos o de usuarios son más sencillas.

Es posible definir un rol como un conjunto identificado de permisos. Para facilitar la gestión de los permisos, SQL Server ofrece los roles predefinidos, también llamados fijos, ya que no es posible añadir o eliminar privilegios en estos roles.

Estos roles fijos se definen en dos niveles:

- **Servidor.**
- **Base de datos.**

Además de estos roles fijos, es posible gestionar otros roles. Es conveniente establecer un nombre único para definir un rol y posteriormente asignar uno o varios permisos respetando un procedimiento en todo punto similar al utilizado para asignar los permisos a los usuarios. Estos roles se pueden

definir en tres niveles:

- **Servidor.**
- **Base de datos.**
- **Aplicación.**

Los roles permiten una gestión simplificada de los privilegios, ya que también es posible definir los perfiles tipo de privilegios y posteriormente asignar a cada usuario de base de datos uno o varios perfiles tipo con objeto de darle todas las autorizaciones que necesita para trabajar en la base de datos.

El usuario dispone al final del conjunto de permisos que se le asignan:

- Directamente a la conexión utilizada.
- Indirectamente por medio de un rol fijo de servidor asignado a la conexión.
- Indirectamente por medio de los roles asignados al usuario de base de datos.
- Directamente al usuario de base de datos.

Como complemento de estos diferentes roles, existe el rol public. Este rol es adicional, ya que todos los usuarios reciben el rol public y no pueden ignorarlo. Este rol también es particular, ya que se le pueden asignar, retirar o prohibir permisos. Todas las modificaciones hechas a nivel de los permisos sobre el rol public son válidas para todos los usuarios. Por lo tanto, no es recomendable trabajar con este rol, aunque, en algunos casos, añade flexibilidad a la gestión de los permisos.

1. Roles de servidor

Son los roles predefinidos que dan a las conexiones un cierto número de funcionalidades. Su uso facilita la gestión en caso de que haya muchos usuarios.

a. Los roles predeterminados

Hay 8 roles de servidor:

sysadmin Ejecuta cualquier operación sobre el servidor. Es el administrador del servidor.

serveradmin Permite configurar los parámetros a nivel del servidor.

setupadmin Permite añadir/eliminar los servidores asociados y ejecutar algunos procedimientos de sistema almacenados como sp_serveroptions.

securityadmin Permite gestionar las conexiones de acceso al servidor

processadmin Permite gestionar los tratamientos que se ejecutan sobre SQL Server.

dbcreator Permite crear y modificar las bases de datos.

diskadmin Permite gestionar los archivos sobre el disco.

bulkadmin Puede ejecutar la instrucción BULK INSERT para insertar los datos por bloques. El interés de este tipo de inserción reside en el hecho de que la operación no se realiza en modo conectado y los tiempos de inserción son mucho más rápidos.

c. Crear un rol de servidor

SQL Server ofrece la posibilidad de crear sus propios roles de servidor. Estos roles solo pueden contener los permisos disponibles a nivel de servidor y se van a asignar a las conexiones de la misma manera que los roles predeterminados.

Antes de crear un rol, es necesario identificar los permisos que deseamos asignarle. Para identificar estos permisos, lo más sencillo es consultar la función sys.fn_builtin_permissions, pasándole como argumento SERVER. Esta consulta se ilustra en el siguiente ejemplo. Tenga cuidado con subestimar los roles predeterminados, ya que permiten dar respuesta a la mayor parte de los casos de uso.

El rol se crea usando la instrucción CREATE SERVER ROLE. Como sucede con todos los objetos SQL Server, las instrucciones ALTER y DROP permiten modificar y eliminar los roles de servidor.

d. Asignar los roles

Los procedimientos sp_helpsrvrole y sp_helpsrvrolemember permiten obtener toda la información necesaria de los diferentes roles fijos del servidor y su asignación.

Para añadir una conexión a un rol de servidor, hay que utilizar la instrucción ALTER ROLE. El hecho de que una conexión se beneficie de un rol o de que se elimine este beneficio, se considera como una modificación.

```
ALTER SERVER ROLE nombreRolServidor ADD MEMBER  
nombreConexión;
```

```
ALTER SERVER ROLE nombreRolServidor DROP MEMBER  
nombreConexión;
```

2. Roles de base de datos

Los roles de bases de datos permiten siempre agrupar las diferentes autorizaciones o denegaciones de instrucciones o de objetos para, de esta manera, facilitar la gestión de los derechos. Existen roles predefinidos y es posible definir roles propios.

Las modificaciones sobre los roles son dinámicas y los usuarios no tienen necesidad de desconectarse/volverse a conectar de/a la base para disfrutar de los cambios.

Los roles predefinidos no pueden eliminarse.

Solo los miembros de los roles fijos de base de datos `db_owner` y `db_securityadmin` pueden gestionar la pertenencia de los usuarios a un rol fijo o no de la base de datos.

a. El rol public

Se trata de un rol predefinido particular, ya que todos los usuarios de la base tienen este rol. Así, cuando se asigna, se elimina o se prohíbe una autorización de instrucción a `public`, instantáneamente todos los usuarios de la base se benefician de las modificaciones.

El rol `public` contiene todas las autorizaciones por defecto que tienen los usuarios de la base de datos.

No es posible asignar miembros a este rol, ya que todo el mundo lo tiene implícitamente. Este rol está presente en todas las bases del servidor, tanto en las de sistema como en las de usuario.

Al crearse, un usuario no dispone de ninguna autorización, salvo aquellas asignadas a `public`.

b. Los roles predefinidos

Aparte del rol `public`, existen roles predefinidos en la base de datos

db_owner

Conjunto de derechos equivalentes a ser el propietario de la base. Este rol contiene todas las actividades posibles en los otros roles de la base de datos, así como la posibilidad de ejecutar algunas tareas de mantenimiento y de configuración de la base. Todos los miembros de este grupo van a crear objetos cuyo propietario es dbo. Se trata del propietario por defecto de todos los objetos y no es necesario especificarlo para manipular sus objetos.

db_accessadmin

Permite añadir o eliminar usuarios en la base de datos. Estos usuarios corresponden a conexiones SQL Server o a grupos de usuarios de Windows.

db_datareader

Permite consultar (SELECT) el contenido de todas las tablas de la base de datos.

db_datawriter

Permite añadir (INSERT), modificar (UPDATE) o eliminar (DELETE) datos en todas las tablas de usuario de la base de datos.

db_ddladmin

Permite añadir (CREATE), modificar (ALTER) o eliminar (DELETE) objetos de la base de datos.

db_securityadmin

Permite gestionar los roles, los miembros de los roles y las autorizaciones sobre las instrucciones y los objetos de la base de datos.

db_backupoperator

Permite efectuar una copia de seguridad de la base de datos.

db_denydatareader

Prohíbe la visualización de los datos de la base.

db_denydatawriter

Prohíbe la modificación de los datos contenidos en la base.

Todos los roles predefinidos están presentes en todas las bases de sistema.

c. Los roles de base de datos definidos por los usuarios

Es posible definir los propios roles con el objetivo de facilitar la administración de los derechos en el interior de la base.

El rol SQL Server se va a establecer cuando varios usuarios de Windows deseen efectuar las mismas operaciones en la base de datos y no exista el grupo Windows correspondiente, o cuando los usuarios autenticados por SQL Server y los autenticados por Windows deban compartir los mismos derechos.

Para los roles definidos a nivel de base de datos, es posible saber qué usuarios se benefician de ellos, viendo la ventana de propiedades del rol desde SQL Server Management Studio.

Los roles se pueden asignar directamente a un usuario o a otro rol. Sin embargo, la superposición repetida de roles puede afectar al rendimiento. Además, no es posible crear roles recursivos.

El uso de roles puede parecer pesado algunas veces en el momento de la creación, pero facilita enormemente las evoluciones que puedan tener lugar (modificación de las autorizaciones, de los usuarios).

Para definir un rol es necesario ser miembro del rol sysadmin o miembro de db_owner o db_securityadmin.

La consulta de las tablas de sistema sys.database_principals y sys.database_role_member permite conocer la lista de los usuarios que pertenecen a cada rol de base de datos.

e. Creación de un rol de base de datos

La creación de los roles de base de datos es posible realizando las operaciones siguientes:

- Desde el explorador de objetos, sitúese sobre la base de datos de usuario en la que se va a realizar la operación.
- Sitúese sobre el nodo Security - Roles - Application Roles.
- Desde el menú contextual asociado al nodo Application Roles, seleccione New Application Role.

```
CREATE ROLE nombreRol
[ AUTHORIZATION propietario ]
```

f. Administración de miembros de un rol

La administración de los miembros de un rol se hace en Transact SQL con la instrucción ALTER ROLE.

```
ALTER ROLE nombreRolServidor ADD MEMBER nombreUsuario;
ALTER ROLE nombreRolServidor DROP MEMBER nombreUsuario;
```

g. Eliminación de un rol

La opción de eliminación está disponible desde el menú contextual asociado al rol que se desea eliminar. Transact SQL

```
DROP ROLE nombreRol
```

3. Roles de aplicación

Los roles de aplicación son los roles definidos a nivel de la base de datos sobre la que existen. Como todos los roles, los roles de aplicación permiten agrupar autorizaciones de objetos y de instrucciones. Sin embargo, los roles de aplicación se distinguen porque no tienen ningún usuario y están protegidos por una contraseña.

La lógica de un rol de aplicación es permitir a todos los usuarios de una aplicación tener suficientes derechos para el correcto funcionamiento de dicha aplicación, donde las operaciones que pueden realizarse sobre la base de datos son controladas por el programa cliente. Sin embargo, los usuarios no disponen de suficientes privilegios para realizar el conjunto de las operaciones directamente en SQL.

Cuando un rol de aplicación se activa desde una aplicación cliente o desde un script, las autorizaciones contenidas en este rol tienen prioridad sobre todas las autorizaciones asignadas directamente o por medio de roles al usuario de la base de datos.

Los roles de aplicación permiten obtener un comportamiento estándar de la aplicación sea cual sea el usuario de Windows que ejecute dicha aplicación.

Como los roles de aplicación se definen a nivel de la base de datos, no es posible asignarles privilegios sobre otras bases. Es más, la conexión a otras bases de datos solo es posible por medio de la cuenta guest.

a. Creación de un rol de aplicación

Los roles de aplicación se gestionan de manera similar a la utilizada para los roles de base de datos.

- Desde el explorador de objetos, sitúese sobre la base de datos de usuario en la que se va realizar esta operación.
- Sitúese sobre el nodo Security - Roles - Application Roles.

```
CREATE APPLICATION ROLE nombreRolAplicación  
WITH PASSWORD = 'contraseña' [, DEFAULT_SCHEMA =  
esquema ]
```

b. Eliminar un rol de aplicación

La eliminación de un rol de servidor se realiza con el menú contextual asociado al rol, seleccionando la opción Delete. Transact SQL

```
DROP APPLICATION ROLE nombreRolAplicación
```