Existen dos maneras de realizar autenticación

- Seguridad de Windows

  Es el medio de seguridad que establece el S.O para ingresar

- Seguridad de la base de datos

  Especifica la seguridad del manejador de base de datos

Existen 3 consultas que nos permiten dar seguridad a la base de datos

- CREATE LOGIN
- ALTER LOGIN
- DROP LOGIN

Sentencia para CREATE LOGIN:

CREATE LOGIN login_name

{WITH option_list1 | FROM {WINDOWS[WITH option_list2[…]]

|CERTIFICATE certname

|ASYMMETRIC KEY key_name}}

Login name: Especifica el nombre del login(usuario) creado. La palabra WITH especifica una o más opciones de login.

Option_list1: contiene un conjunto de opciones. Una de las más importantes es : PASSWORD, para asignar la contraseña. Otras opciones son:

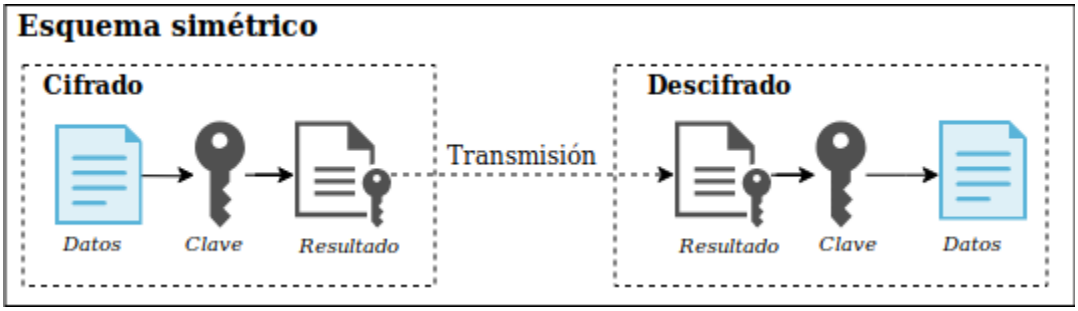
- DEFAULT_DATABASE
- DEFAULT_LANGUAGE
- CHECK_EXPIRATION

Como se ve en CREATE_LOGIN en el FROM contiene las siguientes opciones:

- WINDOWS: Especifica que el login se mapea con el usuario que está logeado
- CERTIFICATE: Especifica el nombre del certificado asociado con el login.
- ASYMMETRIC KEY: Especifica el certificado asimétrico.

Cifrado simétrico:

Son todos los algoritmos que usan la misma clave para realizar el cifrado y el descifrado de los datos.



Ejemplo de CREATE LOGIN:

USE BASE_UPIITA

GO

CREATE LOGIN NIELS_H WITH PASSWORD '123456'

Ejemplo de DROP LOGIN:

USE BASE_UPIITA

GO

DROP LOGIN NIELS_H

Nota: Para poder eliminar al usuario no debe estar activa esa sesión

ESQUEMAS

Los esquemas son modelos de seguridad que simplifican la relación entre los usuarios y los objetos. Estos esquemas tienen un impacto en como se va a interactuar en la base de datos. Básicamente son los recursos (tablas, catálogos, funciones, procedimientos, etc.) que el usuario de base de datos puede utilizar.

Pueden ser a nivel de usuario de grupo

Los esquemas que integran sql server son:

- Guest
- Dbo
- INFORMATION_SCHEMA
- Sys

El motor de base de datos permite a los usuarios sin cuentas acceder a la base de datos mediante el esquema **guest**. El esquema **guest** también se puede aplicar permisos como los otros.

El esquema que se crea por default y no debe definirse(crearse), se define explícitamente e implícitamente es el **dbo**. Cuando el propietario de la base de datos se logea va siempre a encontrase con el esquema **dbo.**

El esquema **INFORMATION_SCHEMA** contiene toda la información de los esquemas de las vistas.

El esquema **sys** contiene objetos del sistema como vistas, catalogos, etc.

Ejemplo de creación de esquema:

```
USE Base_upiita

GO

CREATE SCHEMA esquemita AUTHORIZATION Niels_H

CREATE TABLE CALIFICACIONES (

ID_REGISTRO INT NOT NULL UNIQUE,

DESC_CALIFICACION CHAR(20) NULL


);
```

Nota: antes de realizar esto se debe crear un usario, ver siguiente tema

ROLES

Cuando hay muchos usuarios y se necesita optimizar ciertas actividades que impactan en la base de datos. Es adecuado agregar un rol para agruparlos, no hay que confundir sobre

grupos. Para crear un rol se usa la sintaxis database role, el cual va a especificar la creación del mismo, enfocándose a un grupo de usuarios que pueden acceder a objetos en común dentro de la base de datos.

Los miembros de los roles pueden ser de la siguiente forma :

- Grupos o cuentas individuales de Windows
- Logins
- Otros roles

La arquitectura del motor de la base de datos incluye un conjunto de roles "system" que implícitamente ofrecen permisos especiales. Existen 2 tipos de roles predefinidos:

- Fixed server roles
- Fixed database roles

Junto a estos roles se pueden conocer también los otros tipos de roles:

- Roles por aplicación
- Roles definidos por el usuario
- Roles definidos a nivel de base de datos.

Fixed server roles

Fixed Server role	Descripción
sysadmin	Realizar cualquier actividad en el sistema de base de datos
serveradmin	Configura servicios
setupoadmin	Instala replicaciones y administra el procedimiento extendido
securityadmin	Administra permisos de inicio de sesión y creación de bases de datos y lee auditorias
processadmin	Administra el sistema de procesos
dbcreator	Crea y modifica la base de datos
diskadmin	Administra el control de los discos (mecánicos o solidos)

Fixed database roles

Fixed Database Role	Descripción
Db_owner	Usuarios quienes realizan todas las actividades en la baes de datos
Db_accesadmin	Usuarios quienes pueden agregar o quitar usuarios
Db_datareader	Usuarios quienes pueden ver los datos de todas las tablas en la base de datos
Db_datawriter	Usuarios quienes pueden agregar, modificar, o eliminar datos en todas las tablas de usuarios en la base de datos.
Db_ddladmin	Usuarios quienes pueden realizar operaciones DDL en la base de datos
Db_securityadmin	Usuarios quienes administran todas las actividades asociadas a la seguridad y permisos de la base de datos
Db_backupoperator	Usuarios quienes pueden realizar respaldos en la base de datos
Db_denydatareader	Usuarios quienes pueden ver los datos de cualquier base de datos
Db_denydatawriter	Usuarios quienes no pueden cambiar nada de los datos de la base de datos

USUARIOS

Los usuarios se autentican en una base de datos

CREATE USER

La sentencia CREATE USER, crea una base de datos con el nombre del usuario. El usuario creado va a tener el esquema dbo. Recordando que es el esquema por omisión para registrarse a la base de datos actual, la sintaxis es:

CREATE USER user_name

[FOR {LOGIN|CERTIFICATE certname |ASYMMETRIC KEY key_name}] WITH DEFAULT_SCHEMA = schema_name]

Ejemplo:

CREATE USER ALUMNO

FOR LOGIN Niels_H

WITH DEFAULT_SCHEMA = Mi_esquema

Nota: Para que deje crear el usuario se debe poner en “for login” un login ya creado

Esto se puede consultar en carpeta logins → clic derecho → propiedades de logins → user mapping

AUTORIZACION

Se enfoca en otorgar privilegios a un grupo de roles o usuarios para que realicen alguna actividad o actividades en especifico dentro de la base de datos. Si un usuario no cuenta con la autorización significa que se le van a negar todas las actividades asociadas a la base de datos. Poor lo que se van a usar instrucciones T-SQL con el fin de otorgar o denegar actividades.

Existen 3 consultas para realizar estas funciones:

- GRANT
- DENY
- REVOKE

GRANT

Permite otorgar permisos para realizar actividades dentro de la base de datos. La sintaxis es:

GRANT {ALL [Privilegios]}| lista_de_permisos

[ON [class::] securable] TO lista_principal [WITH GRANT OPTION]

[AS principal]

La palabra ALL es una función deprecada de acuerdo a la versión de SQL Server y solo se mantiene por cuestiones de compatibilidad (para mantener el funcionamiento de la base de datos).

ALL no implica que se le va a otorgar todos los permisos, solo serán unos cuantos de acuerdo a la siguiente lista:

Lista_de_permisos

Permite especificar que sentencias u objetos (separado por comas) se van a otorgar

class::

Especifica el tipo de securable (nivel de seguridad) el cual el permiso esta otorgado

Securable

Es el permiso que tengo otorgado a nivel de objeto o esquema.

lista_principal

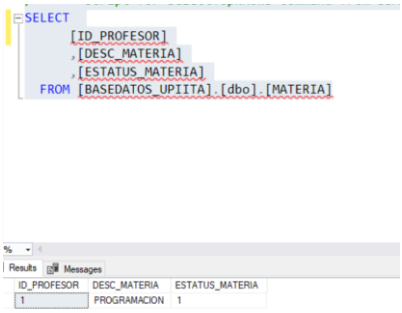
Lista todos los componentes en el que tiene permisos a nivel de usuario o rol.

```
GRANT INSERT ON PROFESOR
TO USUARIO_ALUMNO;


GO
GRANT UPDATE ON PROFESOR
TO USUARIO_ALUMNO;


USE BASEDATOS_UPIITA
GO
--PERMISOS PARA HACER CONSULTA A CIERTAS COLUMNAS DE LA TABLA MATERIA

GRANT SELECT ON MATERIA (ID_PROFESOR, DESC_MATERIA, ESTATUS_MATERIA)
TO USUARIO_ALUMNO;
```



DAR PERMISOS A UNA VISTA

```
USE BASEDATOS_UPIITA
GO
GRANT VIEW DEFINITION ON OBJECT:: dbo._MIVISTA TO USUARIO_ALUMNO


USE BASEDATOS_UPIITA
GO
--AGREGANDO EL OBJETO PARA QUE SE VISUALICE EN LA BASE DE DATOS
--GRANT VIEW DEFINITION ON OBJECT:: dbo._MIVISTA TO USUARIO_ALUMNO
--DANDO PERMISOS DE LECTURA PARA LA VISTA
GRANT SELECT ON dbo._MIVISTA TO USUARIO_ALUMNO
```

```
 USE BASEDATOS_UPIITA
GO
CREATE FUNCTION CONVERTIR_MAYUSCULAS( @VARIABLE_TEXTO  NVARCHAR(50) )

RETURNS NVARCHAR(50)  AS
BEGIN
   RETURN  UPPER(@VARIABLE_TEXTO);
END


USE BASEDATOS_UPIITA
GO
GRANT EXECUTE ON dbo.CONVERTIR_MAYUSCULAS TO USUARIO_ALUMNO
```

DENNY

Permite previene (bloquea) que el usuario realice actividades. Esto significa que quita los permisos que se le otorgaron con **GRANT** o cuando por default se asignaron los permisos al crear la base de datos.

La sintaxis es:

DENNY {**ALL** [Privilegios]}| lista_de_permisos

[**ON** [class::] securable] **TO** lista_principal [WITH GRANT OPTION]

[**CASCADE** AS principal]

La palabra **ALL** es una función deprecada de acuerdo a la versión de SQL Server y solo se mantiene por cuestiones de compatibilidad (para mantener el funcionamiento de la base de datos).

ALL no implica que se le van a quitar todos los permisos, solo serán unos cuantos de acuerdo a la siguiente lista:

Lista_de_permisos

Permite especificar que sentencias u objetos (separado por comas) se van a denegar

class::

Especifica el tipo de **securable** (nivel de seguridad) el cual el permiso esta denegado

Securable

Es el permiso que tengo otorgado a nivel de objeto o esquema.

lista_principal

Lista todos los componentes en el que tiene permisos a nivel de usuario o rol.

CASCADE

Esta opción se va a utilizar para asegurar que niegue todos los permiso aunque existan permisos asociados con otros.

```
--QUITAMOS EL PERMISO A NIELS PARA CONSULTAR LA TABLA PROFESOR
 USE BASEDATOS_UPIITA
GO
--DENY SELECT ON PROFESOR TO USUARIO_ALUMNO
--GRANT SELECT ON dbo._MIVISTA TO USUARIO_ALUMNO
DENY SELECT ON dbo._MIVISTA TO USUARIO_ALUMNO
```

REVOKE

Remueve permisos (GRANT) o bloqueos (DENY). Es decir, que deshabilita la instrucción que se le asigno.

Ej. Si yo asigno un permiso –grant- para realizar un select a una tabla. Después aplico un –Deny- para quitarle ese permiso de lectura, entonces con el –revoke- le remuevo la instrucción o ejecución del –Deny- por lo que se regresa a su estado original.

La sintaxis es:

REVOKE [GRANT OPTION FOR]

{ [ALL [PRIVILEGES] | lista_permisos] }

[ON [class::] securable] FROM lista_principal

[CASCADE AS principal]

```
USE BASEDATOS_UPIITA
 GO

 REVOKE SELECT ON OBJECT::[dbo].PROFESOR  FROM USUARIO_ALUMNO

 --GRANT SELECT ON PROFESOR TO USUARIO_ALUMNO;
```