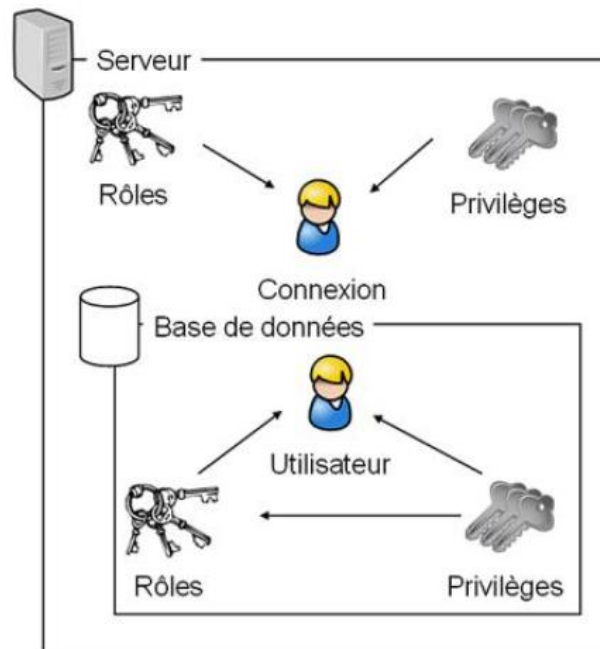


Chapitre 3:

Gestion des Utilisateurs, Droits d'accès et Rôles sous SQL Server



Pr. Jaber EL BOUHDIDI

I. Introduction

- ❖ Avant de pouvoir travailler avec les données gérées par les bases, il faut dans un premier temps se connecter au serveur SQL. Cette étape permet de se faire identifier par le serveur SQL et d'utiliser par la suite tous les droits qui sont accordés à notre connexion.
- ❖ Il existe dans SQL Server deux modes de gestion des accès au serveur de base de données.
 - ✓ Le mode de sécurité Windows
 - ✓ Le mode de sécurité Mixte

I I. Gestion des connexions à SQL SERVER

- ❖ La gestion des connexions peut être réalisée de façon graphique par l'intermédiaire de l'explorateur d'objets depuis SQL Server Management Studio, ou bien sous forme de script T-SQL.
- ❖ Toutes les opérations de gestion des connexions sont réalisables en Transact SQL avec l'aide des instructions:
 - ✓ **CREATE LOGIN**
 - ✓ **ALTER LOGIN**
 - ✓ **DROP LOGIN**

II. Gestion des connexions à SQL SERVER

I. Création d'une connexion:

- ❖ On peut créer une nouvelle connexion tout en se connectant en mode de sécurité Windows ou bien en mode de sécurité mixte
- ❖ **Dans les deux situations**, Il faut posséder une permission d'administrateur (**sysadmin**) ou une permission de gestionnaire de sécurité (**securityadmin**) pour pouvoir réaliser les différentes opérations relatives à la gestion des connexions.

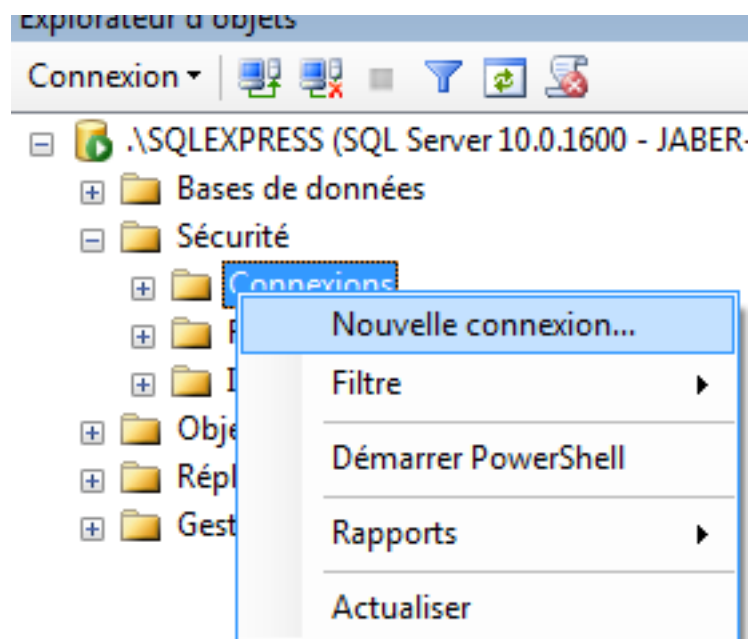
II. Gestion des connexions à SQL SERVER

I. Création d'une connexion:

➤ En mode de sécurité Windows

- ❖ Les noms des utilisateurs devront correspondre à ceux définis dans Windows.

✓ EN utilisant SQL SERVER Management Studio



II. Gestion des connexions à SQL SERVER

Nouvelle connexion

Sélectionner une page

- Général
- Rôles du serveur
- Mappage de l'utilisateur
- Éléments sécurisables
- État

Script Aide

Nom d'accès : Jaber-PC\ELBOUHDIDI Rechercher...

☒ Authentification Windows
☐ Authentification SQL Server

Mot de passe :

Confirmer le mot de passe :

☐ Spécifier l'ancien mot de passe

Ancien mot de passe :

☒ Conserver la stratégie de mot de passe
☒ Conserver l'expiration du mot de passe
☒ L'utilisateur doit changer de mot de passe à la prochaine connexion

☐ Mappé au certificat
☐ Mappé à la clé asymétrique
☐ Mapper aux informations d'identification Ajouter

Informations d'identification mappées

Informations...	Fournisseur

Supprimer

Serveur : .\SQLEXPRESS
 Connexion : JABER-PC\Jaber
[Afficher les propriétés de connexion](#)

Progression

Une erreur s'est produite

Base de données par défaut : master
 Langue par défaut : <par défaut>

II. Gestion des connexions à SQL SERVER

I. Création d'une connexion:

➤ En mode de sécurité Windows

✓ EN utilisant Transact-SQL

```
CREATE LOGIN nom_connexion
FROM WINDOWS
[WITH DEFAULT_DATABASE=baseDeDonnées|
    DEFAULT_LANGUAGE=langue]
```

Nom	Signification
nom_connexion	Nom de l'utilisateur ou du groupe Windows à ajouter. Il doit être donné sous la forme domaine\utilisateur.
baseDeDonnées	Précise la base de données qui sera utilisée par défaut. Si aucune base de données n'est précisée, alors la base de données par défaut est la base Master.
langue	Langue de travail par défaut pour cette connexion. Cette option n'est à préciser que si la langue relative à cette connexion est distincte de celle définie par défaut sur le serveur SQL.

II. Gestion des connexions à SQL SERVER

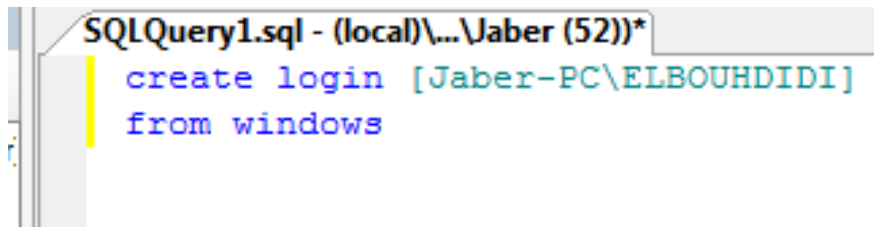
I. Création d'une connexion:

➤ En mode de sécurité Windows

✓ EN utilisant Transact-SQL

```
CREATE LOGIN nom_connexion
FROM WINDOWS
[WITH DEFAULT_DATABASE=baseDeDonnées |
    DEFAULT_LANGUAGE=langue]
```

Exemple



SQLQuery1.sql - (local)\... \Jaber (52))*

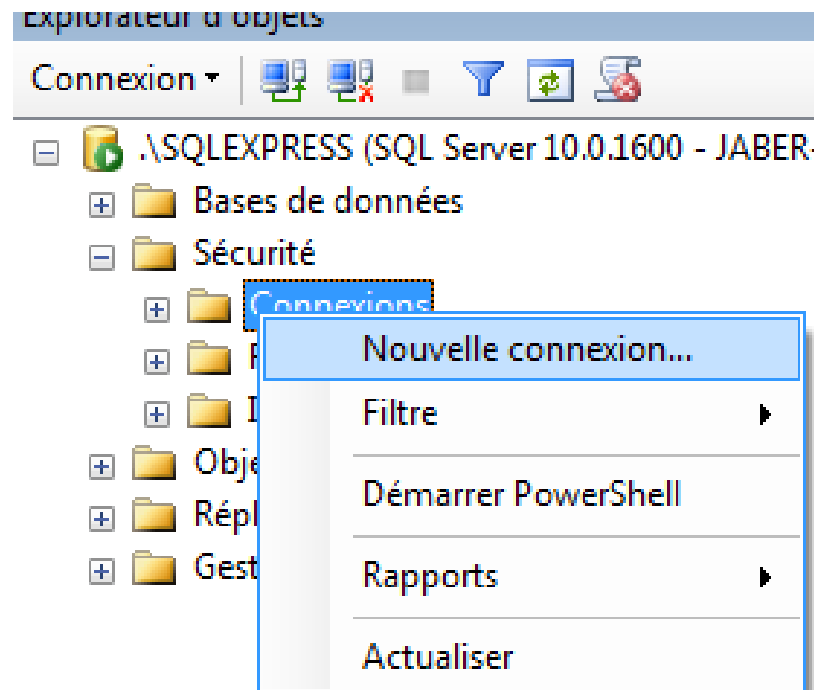
```
create login [Jaber-PC\ELBOUHDIDI]
from windows
```


II. Gestion des connexions à SQL SERVER

I. Création d'une connexion:

➤ En mode de sécurité Mixte

✓ EN utilisant SQL SERVER Management Studio



II. Gestion des connexions à SQL SERVER

Login - New

Select a page

- General
- Server Roles
- User Mapping
- Securables
- Status

Connection

Server:
JABER-PC\SQLEXPRESS

Connection:
JABER-PC\Asus

[View connection properties](#)

Progress

Ready

Script Help

Login name: Ali Search...

☐ Windows authentication
☒ SQL Server authentication

Password:

Confirm password:

☐ Specify old password
 Old password:

☐ Enforce password policy
☐ Enforce password expiration
☐ User must change password at next login

☐ Mapped to certificate
☐ Mapped to asymmetric key
☐ Map to Credential

Mapped Credentials

Credential	Provider

Add Remove

Default database: master

Default language: <default>

OK Cancel

II. Gestion des connexions à SQL SERVER

I. Création d'une connexion:

➤ En mode de sécurité Mixte

✓ EN utilisant Transact-SQL

```
CREATE LOGIN nom_connexion
WITH {PASSWORD='motDePasse' |
      motDePasseHachéHASHED}

[MUST_CHANGED]
[,SID=sid |
,DEFAULT_DATABASE=baseDeDonnées |
,DEFAULT_LANGUAGE=langue |
,CHECK_EXPIRATION={ON | OFF} |
,CHECK_POLICY={ON | OFF}
,[CREDENTIAL=nom_credit]]
```

II. Gestion des connexions à SQL SERVER

I. Création d'une connexion:

➤ En mode de sécurité Mixte

✓ EN utilisant Transact-SQL

```
CREATE LOGIN nom_connexion  
WITH PASSWORD='motDePasse'
```

Exemple

```
SQLQuery1.sql - JABER-PC\...VA...(53))*  
1 create login Test_login with password = '123456'  
2
```

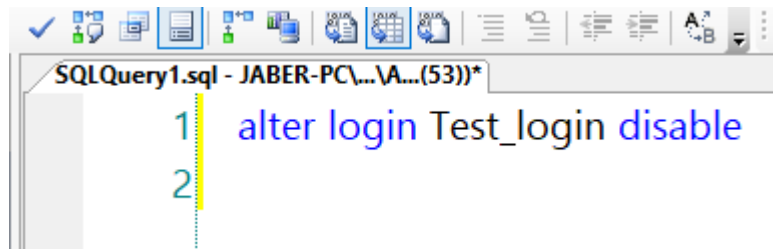
11. Gestion des connexions à SQL SERVER

2. Activer et désactiver une connexion:

✓ **EN utilisant Transact-SQL**

ALTER LOGIN nomConnexion {ENABLE|DISABLE}

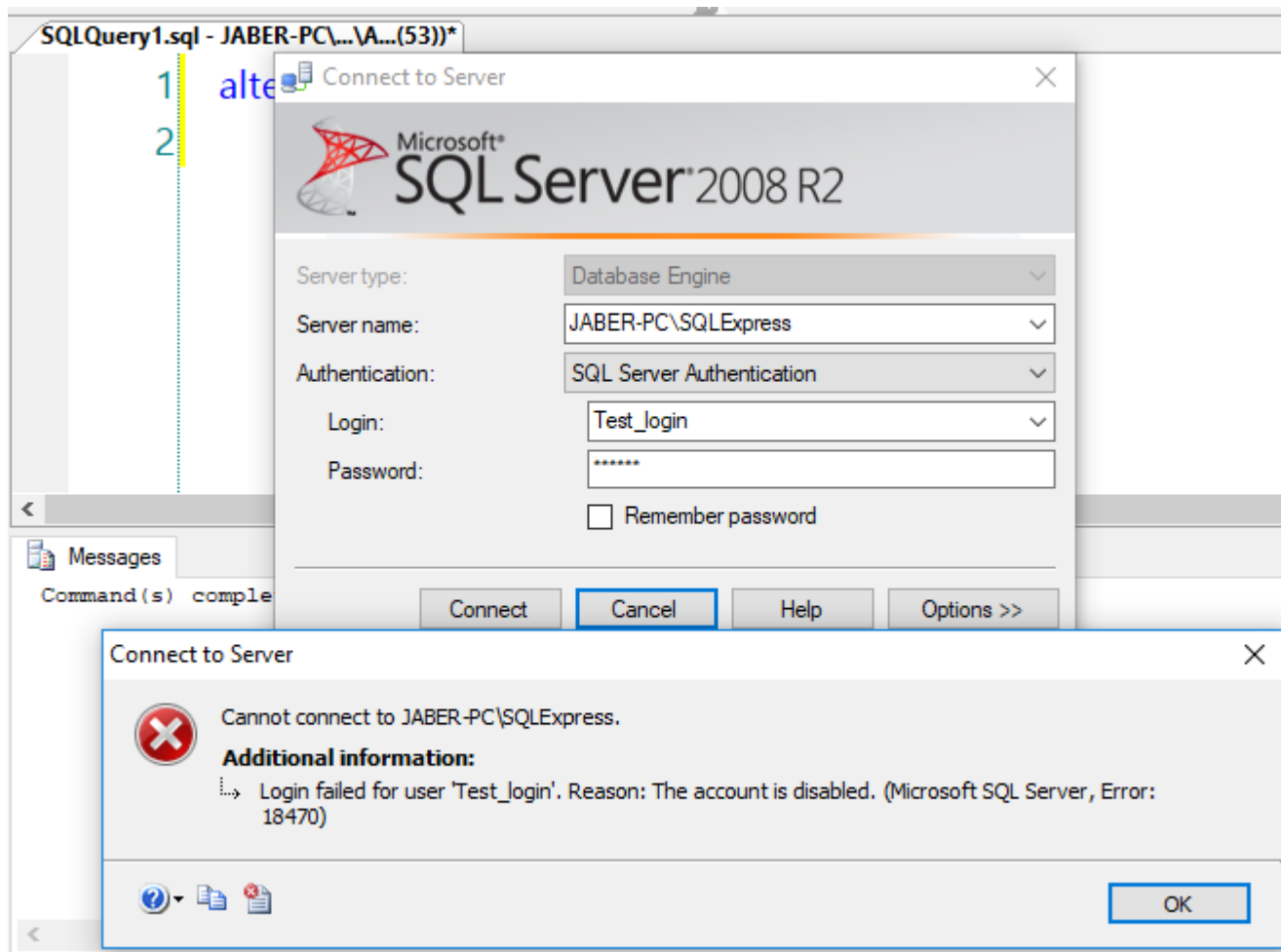
Exemple: Désactiver « Test_login »



1. Gestion des connexions à SQL SERVER

2. Activer et désactiver une connexion:

✓ **EN utilisant Transact-SQL**



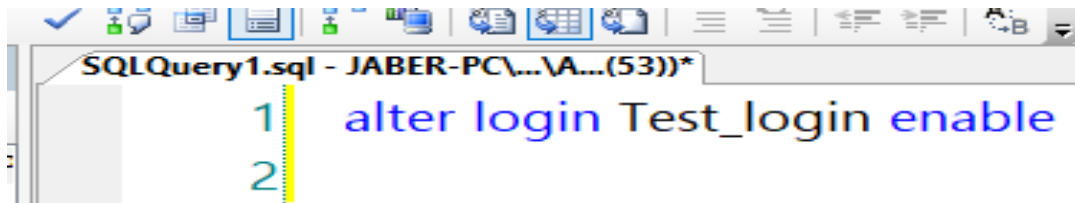
II. Gestion des connexions à SQL SERVER

2. Activer et désactiver une connexion:

✓ EN utilisant Transact-SQL

ALTER LOGIN nomConnexion {ENABLE|DISABLE}

Exemple: Activer à nouveau le compte « Test_login »



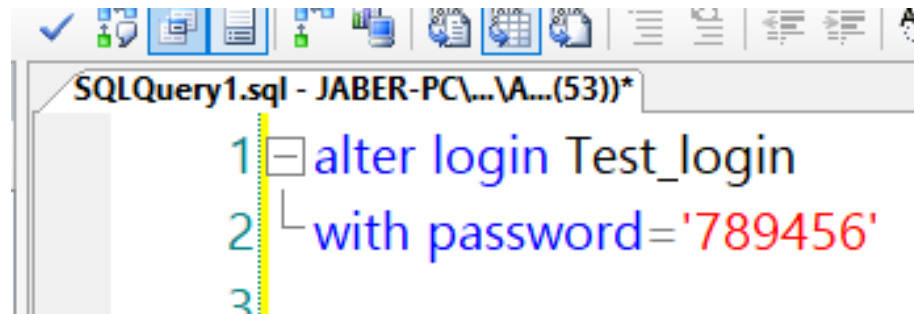
I I. Gestion des connexions à SQL SERVER

3. Modifier le mot de passe d'une connexion

✓ EN utilisant Transact-SQL

```
ALTER LOGIN nom_connexion  
WITH PASSWORD='NouveauMotDePasse'
```

Exemple



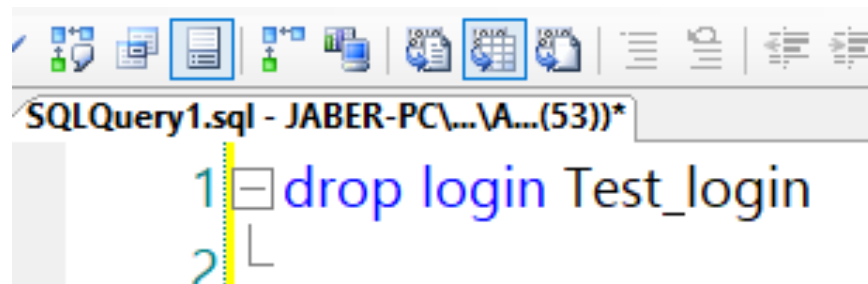
II. Gestion des connexions à SQL SERVER

4. Supprimer une connexion

✓ EN utilisant Transact-SQL

DROP LOGIN nom_connexion

Exemple



II. Gestion des connexions à SQL SERVER

Obtenir les comptes de connexion purement Windows :

```
SELECT *  
FROM sys.server_principals  
WHERE type_desc = 'WINDOWS_LOGIN'
```

III. Gestion des utilisateurs de base de données

I. Introduction

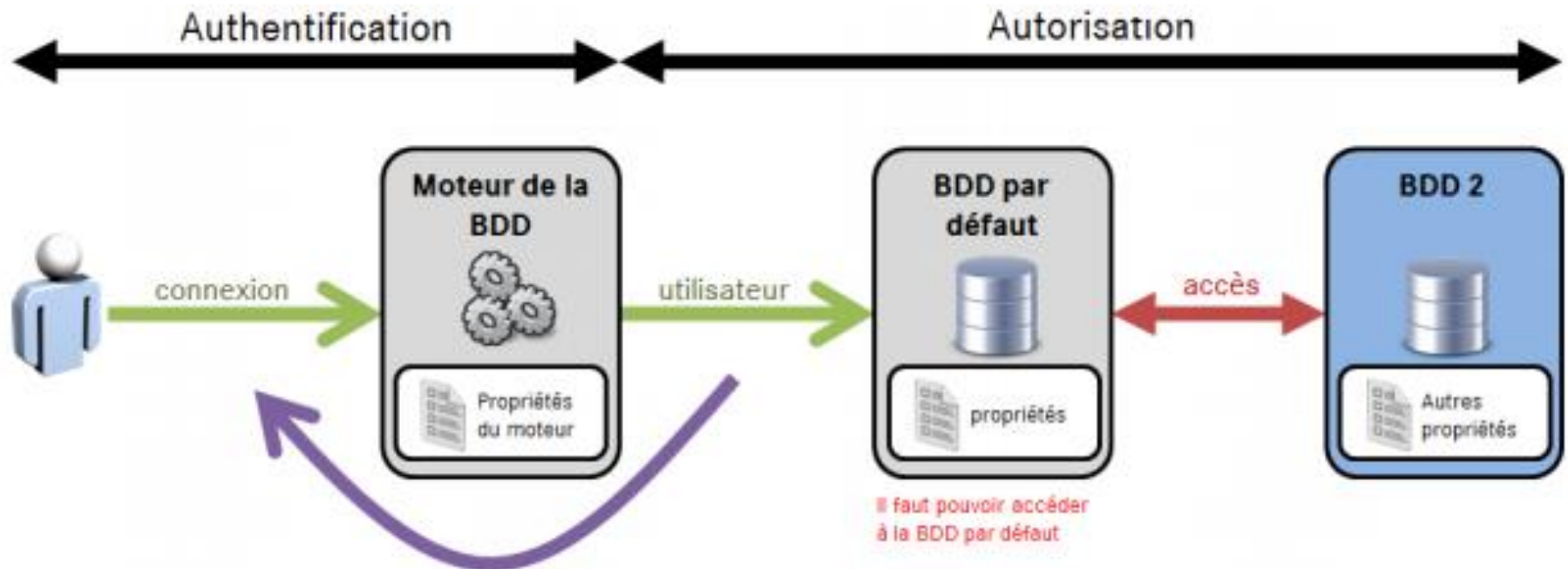
- ❖ Après la définition des connexions (login) au niveau du serveur, il est nécessaire de définir des utilisateurs dans les différentes bases de données..
- ❖ Lors de la définition d'une connexion, la base de données par défaut permet de positionner le compte de connexion sur une base de données pour commencer à travailler. Cependant, la connexion ne pourra réellement travailler sur la base que s'il existe un compte d'utilisateur défini au niveau base et associé à la connexion.

III. Gestion des utilisateurs de base de données

I. Introduction

- ❖ C'est un point de passage obligatoire, **sauf si la connexion s'est vue attribuée des privilèges de haut niveau.**
- ❖ C'est au niveau des utilisateurs de base de données que seront attribués **les droits d'utilisation des objets** définis dans la base de données.
- ❖ À sa création, un utilisateur d'une BDD, **ne dispose d'aucun privilège.** Il est nécessaire **de lui accorder tous les droits** dont il a besoin.

III. Gestion des utilisateurs de base de données



Deux niveaux de sécurité : au niveau de l'instance (du serveur) + au niveau de chaque base

III. Gestion des utilisateurs de base de données

2. Créer un utilisateur

- ❖ La création d'un utilisateur de base de données va permettre de lier une connexion avec un utilisateur, et donc autoriser l'utilisation de cette base.

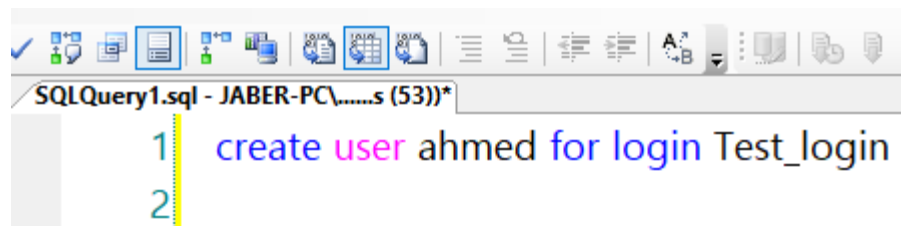
Syntaxe

CREATE USER nomUtilisateur **FOR LOGIN** nomConnexion

Remarque:

Avant d'écrire cette requête, il faut tout d'abord se positionner sur la base de données concernée ou bien d'écrire (use Nom_BD;) au début.

➤ Exemple



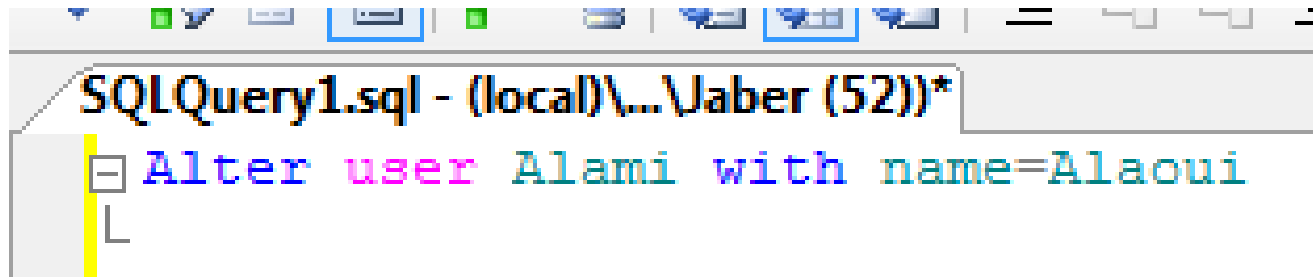
III. Gestion des utilisateurs de base de données

3. Modifier un utilisateur

➤ Syntaxe

ALTER USER nomUtilisateur **WITH NAME =** nouveauNomUtilisateur

➤ Exemple



```
SQLQuery1.sql - (local)\... \Jaber (52))*
Alter user Alami with name=Alaoui
```

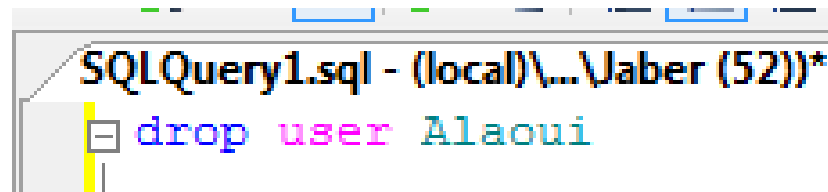
III. Gestion des utilisateurs de base de données

4. Supprimer un utilisateur

➤ Syntaxe

DROP USER nomUtilisateur

➤ Exemple



The screenshot shows a window titled "SQLQuery1.sql - (local)\... \Jaber (52))*". Inside the editor, the SQL command "drop user Alaoui" is written in a monospaced font. The word "drop" is in blue, "user" is in magenta, and "Alaoui" is in green. A yellow vertical cursor is positioned at the end of the command.

```
SQLQuery1.sql - (local)\... \Jaber (52))*
drop user Alaoui
```


Gestion des Droits

IV. Gestion des droits

- ❖ Dans SQL, le sous ensemble DCL dit "data control language" s'occupe de la gestion des privilèges.
- ❖ Cette partie de SQL s'occupe de contrôler quel utilisateur peut ou ne peut pas utiliser tel ou tel ordre SQL sur tel ou tel objet.

IV. Gestion des droits

- ❖ SQL Server gère les privilèges avec trois types de mots clés :
 - ✓ GRANT
 - ✓ REVOKE
 - ✓ DENY
- ❖ C'est –à-dire qu'un privilège peut être accordé(**GRANT**), ou bien retiré (**REVOKE**) s'il a été accordé. L'instruction **DENY** permet d'interdire l'utilisation d'un privilège particulier même si le privilège en question a été accordé soit directement soit par l'intermédiaire d'un rôle.

IV. Gestion des droits

SQL Server gère quatre types de droits:

- 1) **Droits d'utilisation :** INSERT, UPDATE, SELECT, DELETE, EXECUTE.
- 2) **Droits d'instruction:** CREATE TABLE, CREATE FUNCTION, CREATE PROCEDURE, CREATE VIEW, BACKUP DATABASE, BACKUP LOG, ...
- 3) **Droits au niveau base de données:** Au niveau base de données, il est possible de donner des droits à un utilisateur, à un schéma, etc.
- 4) **Droit au niveau serveur:** ces privilèges sont attribués à une connexion et non pas à un utilisateur, Exemples: CREATE ANY DATABASE, ALTER ANY DATABASE, SHUTDOWN, etc.

IV. Gestion des droits(Droits d'utilisation)

I. Attribuer des privilèges

- ❖ Les privilèges sont, pour un ou plusieurs utilisateurs la possibilité d'utiliser certains ordres SQL sur certains objets
- ❖ L'ordre SQL **GRANT** permet d'attribuer un privilège à différents utilisateurs sur différents objets.
- ❖ Cet ordre est utilisé avec la syntaxe suivante:

```
GRANT { <privilège> [ { , <privilège> }... ] ON [TABLE] <objet table> |
      ALL PRIVILEGES ON [TABLE] <objet table> |
      REFERENCES [ ( <liste de colonne> ) ] ON <nom de table> }
TO <user1> [ { , <user2> }... ] | public
[ WITH GRANT OPTION ]
```

IV. Gestion des droits(Droits d'utilisation)

- ❖ AVEC privilège : soit **SELECT**, soit **DELETE**, soit **INSERT**, soit **UPDATE**.
- ❖ La clause **WITH GRANT OPTION**, est utilisée pour autoriser la transmission des droits.
- ❖ La clause **ALL PRIVILEGES** n'a d'intérêt que dans le cadre de la transmission des droits.

IV. Gestion des droits (Droits d'utilisation)

I. Attribuer des privilèges

➤ Exemples

Exemple	Signification
GRANT SELECT ON EMP TO Ahmed	Autorise l'utilisateur Ahmed à lancer des ordres SQL SELECT sur la table EMP
GRANT SELECT, UPDATE ON DEPT TO PUBLIC	Autorise tous les utilisateurs présents et à venir à lancer des ordres SQL SELECT et UPDATE sur la table DEPT .

IV. Gestion des droits(Droits d'utilisation)

I. Attribuer des privilèges

➤ Exemples

Exemple	Signification
GRANT SELECT, INSERT, DELETE ON CLIENT TO Kamal WITH GRANT OPTION	Autorise Kamal à lancer des ordres SQL SELECT, INSERT, DELETE sur la table CLIENT mais aussi à transmettre à tout autre utilisateur les droits qu'il a acquis dans cet ordre.
Kamal lance l'ordre suivant: GRANT ALL PRIVILEGES ON CLIENT TO Ahmed	Autorise Ahmed à lancer sur la table CLIENT, les mêmes ordres SQL, que ceux autorisé à Kamal (SELECT, INSERT, DELETE).
Kamal lance l'ordre suivant: GRANT UPDATE ON CLIENT TO Ahmed	Cet ordre va provoquer une erreur, car kamal n'est pas autorisé à lancer des ordres UPDATE sur la table CLIENT et ne peut donc transmettre un droit qu'il n'a pas !

IV. Gestion des droits (Droits d'utilisation)

2. Révoquer des privilèges

- ❖ L'ordre SQL **REVOKE** permet de révoquer, c'est à dire "retirer" un privilège.
- ❖ Sa syntaxe est la suivante :

```
REVOKE [ GRANT OPTION FOR ]
    { <privilège> [ { , <privilège> }... ] ON [TABLE] <objet table> |
    ALL PRIVILEGES ON [TABLE] <objet table> |
    REFERENCES [ ( <liste de colonne> ) ] ON <nom de table> }
FROM <user1> [ { , <user2> }... ] | public
[RESTRICT | CASCADE]
```

- ❖ AVEC privilège : soit **SELECT**, soit **DELETE**, soit **INSERT**, soit **UPDATE**

IV. Gestion des droits(Droits d'utilisation)

2. Révoquer des privilèges

➤ Exemples

Exemple	Signification
REVOKE SELECT ON EMP FROM Ahmed	Supprime le privilège de sélection de la table EMP attribué à Ahmed
REVOKE SELECT, UPDATE ON DEPT FROM Ahmed, kamal	Supprime les privilèges de sélection et de la modification de la table DEPT attribués à Ahmed et Kamal
REVOKE GRANT OPTION FOR SELECT ON CLIENT FROM Kamal	Supprime la possibilité pour Kamal de transmettre le privilège de sélection sur la table CLIENT.

IV. Gestion des droits (Droits d'utilisation)

2. Révoquer des privilèges



- ❖ l'utilisation de l'expression **GRANT OPTION FOR** ne révoque pas les droits mais supprime la possibilité de cession des droits lorsque ces droits ont été définis en utilisant la clause **WITH GRANT OPTION**.
- ❖ **CASCADE** : si la permission retirée a été accordée **WITH GRANT OPTION, CASCADE** retirera également le privilège aux utilisateurs qui l'ont reçue ainsi de l'utilisateur concerné par la suppression initiale du privilège. SQL Server oblige d'ailleurs à mentionner **CASCADE** quand un droit a été accordé **WITH GRANT OPTION**

IV. Gestion des droits (Droits d'utilisation)

3. Interdire des privilèges

- ❖ L'ordre SQL **DENY** permet d'interdire l'utilisation d'un privilège particulier.
- ❖ Sa syntaxe est la suivante :

```
DENY { <privilège> [ { , <privilège> }... ] ON [TABLE] <objet table> |
      ALL PRIVILEGES ON [TABLE] <objet table> |
      REFERENCES [ ( <liste de colonne> ) ] ON <nom de table> }
To <user1> [ { , <user2> }... ] | public
```

- ❖ AVEC privilège : soit **SELECT**, soit **DELETE**, soit **INSERT**, soit **UPDATE**

IV. Gestion des droits (Droits d'utilisation)

3. Interdire des privilèges

➤ Exemples

Exemple	Signification
DENY DELETE ON EMP TO Ahmed	Interdire le privilège de suppression des données de la table EMP pour l'utilisateur Ahmed

- avec le compte sa:
- 1) créer une nouvelle connexion login_admin avec un mot de passe 123456;
- 2) donner à cette connexion la possibilité d'accéder à une base de données de votre choix?
- 3) donner la possibilité à cette connexion de selectionner , modifier les données d'une table de votre choix avec possibilité de transmission ?
- 4) donner la possibilité à cette connexion d'insérer et de supprimer les données d'une table de votre choix sans possibilité de transmission ?
- 5) créer une nouvelle connexion login_invite avec un mot de passe 123456;
- 6) donner à cette connexion la possibilité d'accéder à la même base de données que le login_admin.

- avec le compte login_admin:
- 7) donner à la connexion login_invite le droit de selectionner les données(même table de login_admin) avec possibilité de transmission? vérifier le droit reçu.
- 8) donner à la connexion login_invite le droit d'insérer les données(même table de login_admin) avec possibilité de transmission? vérifier le droit reçu.
- avec le compte sa:
- 9) retirer les droits d'insérer et de supprimer les données de la connexion login_admin? vérifier les droits à l'aide du compte admin_login et login_invite?
- 10) retirer le droit de modifier les données de la connexion login_admin? vérifier les droits à l'aide du compte admin_login et login_invite?
- 11) retirer la possibilité de transmission de droits de selection les données de la connexion login_admin? vérifier les droits à l'aide du compte admin_login?

Gestion des rôles

V. Gestion des rôles

- ❖ les rôles sont des sortes de groupements de droits. Il existe 2 niveaux d'actions pour les rôles : **Server, Base de données.**
- ❖ L'utilité des rôles réside dans le fait qu'il **est plus simple** d'attribuer des **droits à des rôles** puis d'attribuer des rôles à des utilisateurs ou des connexions plutôt que **d'attribuer directement des droits** à ces derniers.
- ❖ les rôles sont **des ensembles de droits** qui portent un nom et qu'on peut les attribuer aux utilisateurs.

rôles server

SERVER LEVEL ROLES AND PERMISSIONS: 9 fixed server roles, 34 server permissions

sysadmin fixed server role

CONTROL SERVER: Has all permissions in the server

bulkadmin fixed server role

ADMINISTER BULK
OPERATIONS

dbcreator fixed server role

ALTER ANY DATABASE
CREATE ANY DATABASE

setupadmin fixed server role

ALTER ANY LINKED SERVER

securityadmin fixed server role

ALTER ANY LOGIN

Important: The securityadmin role should be treated as equivalent to the sysadmin role.

diskadmin fixed server role

ALTER RESOURCES

serveradmin fixed server role

ALTER SETTINGS

SHUTDOWN

ALTER ANY ENDPOINT

CREATE ENDPOINT

ALTER SERVER STATE

VIEW SERVER STATE

processadmin fixed server role

ALTER ANY CONNECTION

VIEW ANY DEFINITION

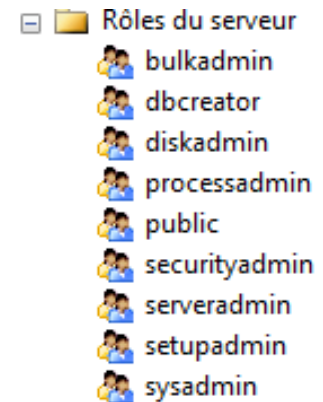
public fixed server role

There are no server-level permissions inherent in the public server role, however some server permissions are present by default. Specifically, VIEW ANY DATABASE, and CONNECT permission to the endpoints. These permissions can be revoked.

The other server level permissions, are not granted to any fixed server role except sysadmin.

V. Les rôles prédéfinis

V.1. Rôles serveur :



1. **Sysadmin** : Administrateur du serveur.
2. **Serveradmin** : Permet de configurer les paramètres niveau serveur.
3. **Setupadmin** : Permet d'exécuter certaines procédures stockées et d'ajouter des serveurs liés.
4. **Securityadmin** : Permet de gérer les connexions serveur.
5. **Processadmin** : Permet de gérer les traitements au sein de SQL Server.
6. **Dbcreator** : Permet de créer ou modifier des bases de données.
7. **Diskadmin** : Permet de gérer les fichiers sur le disque.
8. **Bulkadmin** : Permet d'exécuter l'instruction BULK INSERT.

V. Affecter et supprimer des rôles server prédéfinis

- Pour affecter un rôle prédéfini, on utilise la procédure stockée:

sp_addsrvrolemember

Syntaxe: Exec sp_addsrvrolemember 'login' , 'role'

- Pour supprimer un rôle prédéfini, on utilise la procédure

stockée: **sp_dropsrvrolemember**

Syntaxe: Exec sp_dropsrvrolemember 'login' , 'role'

Avec rôle:

- **sysadmin**
- **securityadmin**
- **serveradmin**
- **setupadmin**
- **processadmin**
- **diskadmin**
- **dbcreator**
- **bulkadmin**

V. Affecter des rôles server prédéfinis

Exemple :

```
EXECUTE sp_addsrvrolemember 'Test_login','dbcreator';
```

- ✓ Notez que par défaut, tous les membres du groupe Windows sont membres du rôle serveur **fixe sysadmin**.
- ✓ Il **n'est pas possible** de créer vos propres rôles au niveau serveur. Il est en revanche possible d'affecter des privilèges au niveau du compte de connexion, ce qui se fait par la commande GRANT dans le contexte de la base master.

Exemple :

```
USE master;
```

```
GRANT SHUTDOWN TO Test_login;
```

Rôles bases de données

DATABASE LEVEL ROLES AND PERMISSIONS: 11 fixed database roles, 77 database permissions

db_owner fixed database role

CONTROL DATABASE: Has all permissions in the database

db_datareader

GRANT SELECT ON DATABASE::

db_denydatareader

DENY SELECT ON DATABASE::

db_datawriter

GRANT INSERT ON DATABASE::

GRANT UPDATE ON DATABASE::

GRANT DELETE ON DATABASE::

db_denydatawriter

DENY INSERT ON DATABASE::

DENY UPDATE ON DATABASE::

DENY DELETE ON DATABASE::

db_accessadmin

CREATE SCHEMA

ALTER ANY USER

CONNECT

db_securityadmin

ALTER ANY ROLE, CREATE ROLE

ALTER ANY APPLICATION ROLE

VIEW DEFINITION

public

There are no database-level permissions inherent in the public database role, however some database permissions are present by default. Specifically, VIEW ANY COLUMN MASTER KEY DEFINITION, VIEW ANY COLUMN ENCRYPTION KEY DEFINITION, and SELECT permission on many individual system tables. These permissions can be revoked.

db_backupoperator

BACKUP DATABASE

BACKUP LOG

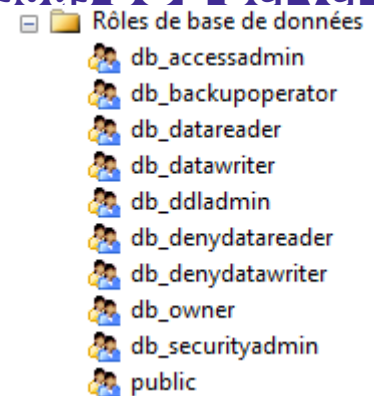
CHECKPOINT

db_ddladmin

ALTER ANY ASSEMBLY
ALTER ANY ASYMMETRIC KEY
ALTER ANY CERTIFICATE
ALTER ANY CONTRACT
ALTER ANY DATABASE DDL TRIGGER
ALTER ANY DATABASE EVENT NOTIFICATION
ALTER ANY DATASPACE
ALTER ANY FULLTEXT CATALOG
ALTER ANY MESSAGE TYPE
ALTER ANY REMOTE SERVICE BINDING
ALTER ANY ROUTE
ALTER ANY SCHEMA
ALTER ANY SERVICE
ALTER ANY SYMMETRIC KEY
CHECKPOINT
CREATE AGGREGATE
CREATE DEFAULT
CREATE FUNCTION
CREATE PROCEDURE
CREATE QUEUE
CREATE RULE
CREATE SYNONYM
CREATE TABLE
CREATE TYPE
CREATE VIEW
CREATE XML SCHEMA COLLECTION
REFERENCES

There are various special purpose roles in the msdb database

The other database level permissions, are not granted to any fixed database role except db_owner.



V. Les rôles prédéfinis BDD

V.2. Rôles base de données:

1. **Db_owner** : Equivalent au propriétaire de base de données.
2. **Db_accessadmin** : Permet d'ajouter ou supprimer des utilisateurs dans la BD.
3. **Db_datareader** : Permet d'utiliser l'instruction SELECT.
4. **Db_datawriter** : Permet les instructions INSERT, UPDATE et DELETE.
5. **Db_ddladmin** : Permet d'ajouter (CREATE), modifier (ALTER), supprimer (DROP) des objets dans la BD.
6. **Db_securityadmin** : Permet de gérer les rôles, les membres des rôles et les droits directs sur tous les objets de la BD.
7. **Db_backupoperator** : Permet l'utilisation des backups.
8. **Db_denydatareader** : Interdit l'instruction SELECT.
9. **Db_denydatawriter** : Interdit l'écriture sur la base de données.

V. Les rôles prédéfinis

Pour affecter un rôle prédéfini des bases de données, on utilise la procédure stockée: **sp_addrolemember**

Exemple :

```
USE mabase;
```

```
EXEC sp_addrolemember 'db_datareader','Test_login';
```

V. Rôles définis par l'utilisateur

L'utilisateur d'une BDD peut créer ses propres rôles à l'aide de la commande CREATE ROLE.

Exemple :

Use BDD;

❖ CREATE ROLE R_CreerTable_Vue;

→ Le rôle **R_CreerTable_Vue** permet de créer des tables et des vues :

❖ GRANT CREATE TABLE ,CREATE VIEW TO **R_CreerTable_Vue**

❖ sp_addrolemember **R_CreerTable_Vue**,Test_login



- Pour autoriser une connexion (Par ex. Test_login) à créer des bases de données, il faudra lui donner la permission à l'aide de la connexion **sa** ou compte **administrateur windows**.

sp_addsrvrolemember 'Test_login','dbcreator'

- Pour associer à une connexion une BDD, exécuter la procédure stockée :

Use nom_bdd;

exec sp_grantdbaccess 'Test_login'

- Pour donner à l'utilisateur *Test_user* le droit de création de tables :

use nom_bdd;

grant create table to Test_user

- Pour donner à un utilisateur le droit de créer des triggers sur une BDD

GRANT ALTER ANY DATABASE DDL TRIGGER TO user_name

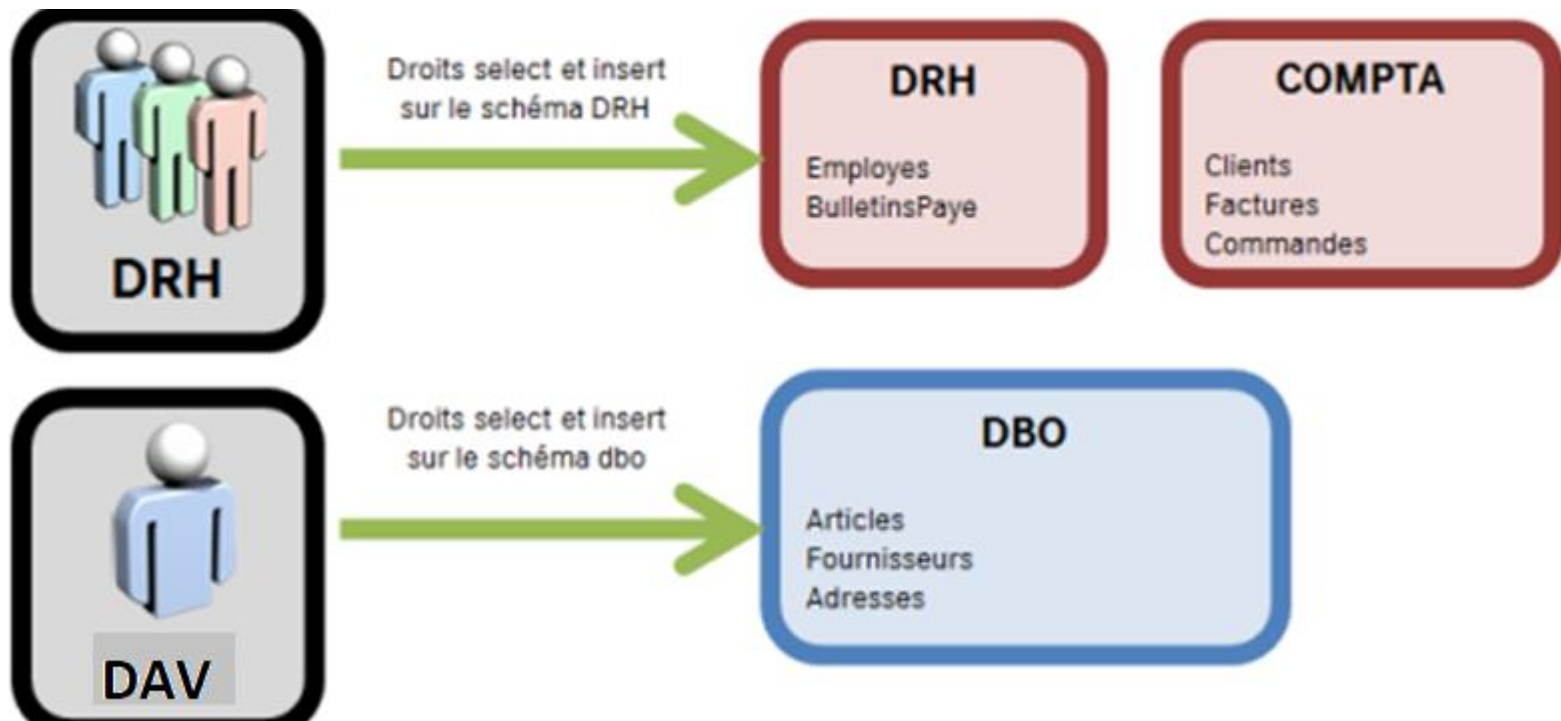
Gestion des Schémas

Gestion des Schémas

Un schéma est un ensemble logique d'objets (tables, vues, procédures, fonctions) à l'intérieur d'une BD (très utile dans les grandes BD). Ils facilitent la gestion des privilèges d'utilisation des objets. Si aucun nom de schéma n'est précisé, l'utilisateur travaille par défaut sur le schéma **dbo**.

On peut associer plusieurs utilisateurs à un même schéma.

Gestion des Schémas



Un schéma est un rassemblement logique d'objets. Le but est de permettre une administration plus facile des objets contenus dans ce schéma.

Création d'un schéma à l'aide de l'Interface graphique

Pour créer un schéma:

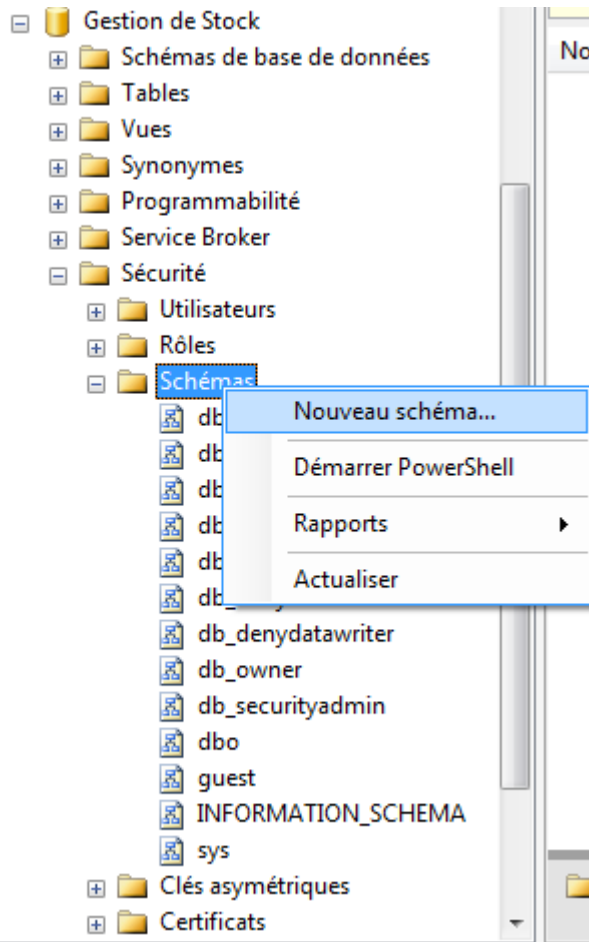
1) Dans l'Explorateur d'objets, développez le dossier **Bases de données**.

2) Développez la base de données où créer le schéma de la base de données.

3) Cliquez avec le bouton droit sur le dossier **Sécurité**, pointez sur **Nouveau**, puis sélectionnez **Schéma**.

4) Dans la boîte de dialogue **Schéma - Nouveau**, sur la page **Général**, entrez un nom pour le nouveau schéma dans la zone **Nom du schéma**.

5) Dans la zone **Propriétaire du schéma**, entrez le nom d'un utilisateur de base de données ou d'un rôle propriétaire du schéma. Vous pouvez également cliquer sur **Rechercher** pour ouvrir la boîte de dialogue **Rechercher les rôles et les utilisateurs**. Cliquez sur **OK**.



Création d'un schéma à l'aide du T-SQL

Syntaxe:

```
CREATE SCHEMA schema_name  
AUTHORIZATION propriétaire
```

Création d'un schéma à l'aide du T-SQL

Exemple:

USE BDI;

CREATE SCHEMA Vente
authorization user3

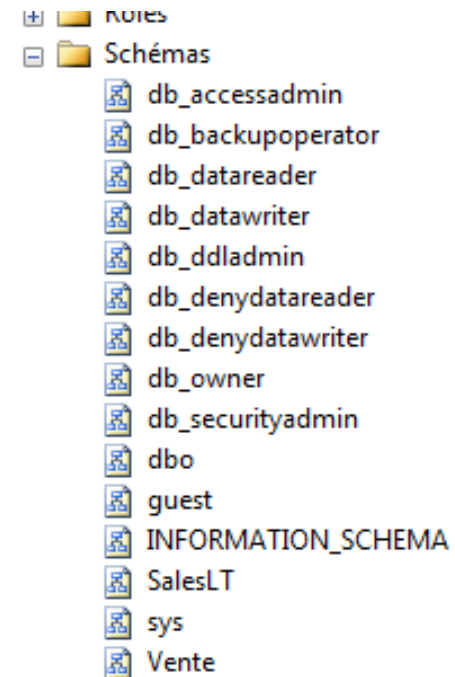
GO

Droits à user1 d'utiliser select sur tous les objets du schéma vente

GRANT SELECT ON SCHEMA::Vente TO user1

DENY SELECT ON SCHEMA::Vente TO user2;

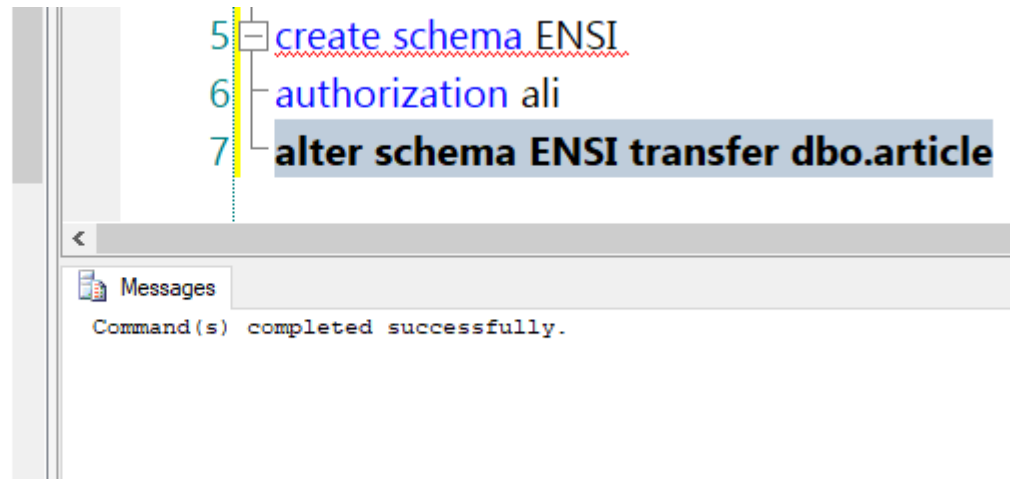
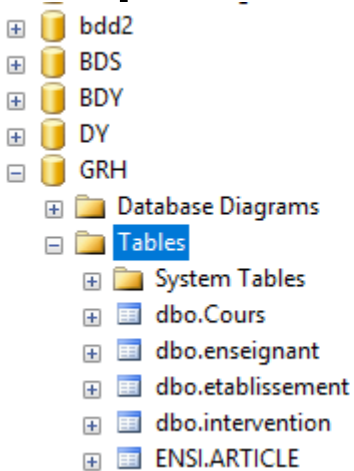
go



Transférer un objet vers un schéma

```
alter schema Vente transfer dbo.Customer;  
Go
```

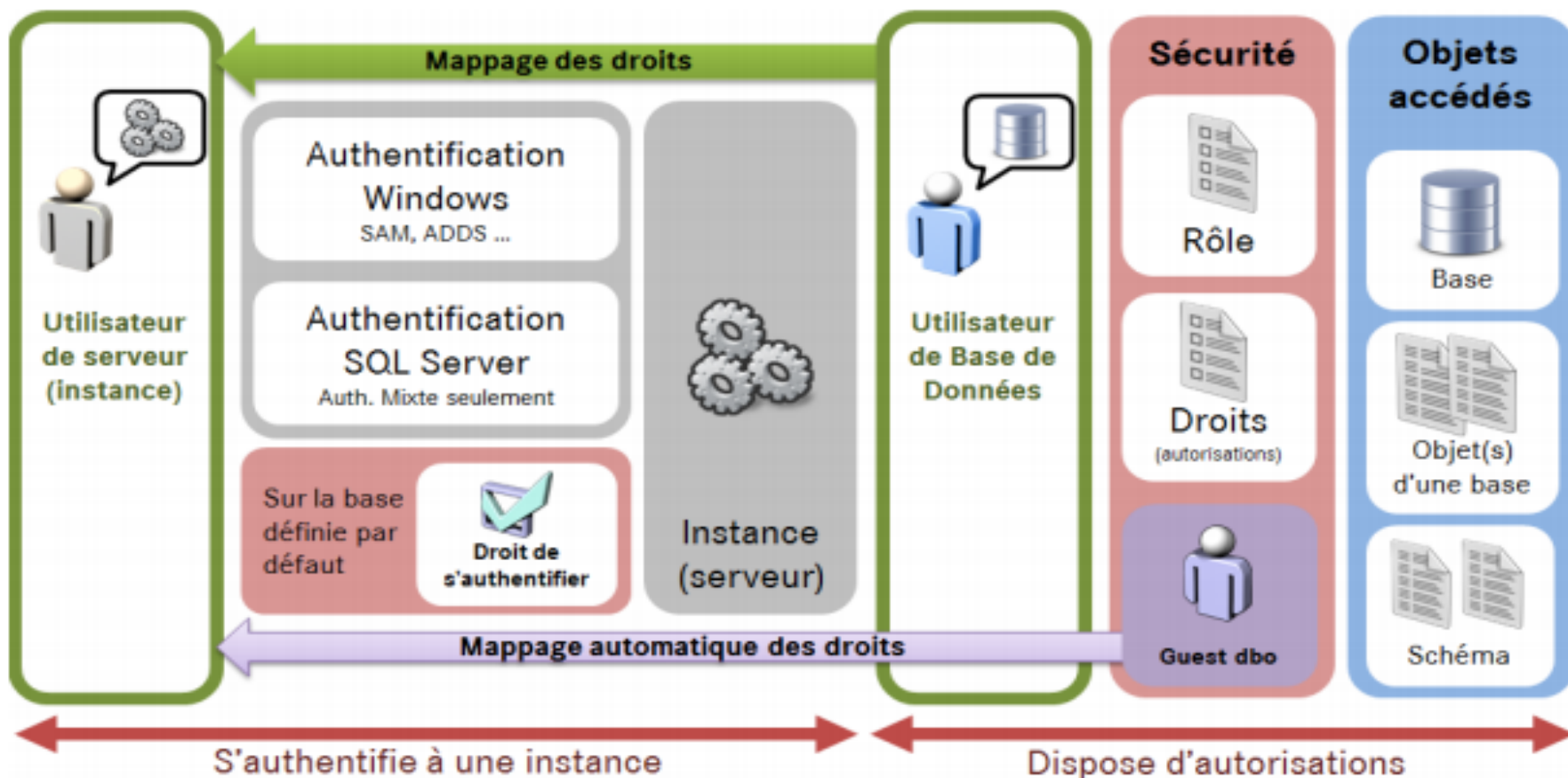
Exemple



Conclusion

Un utilisateur dispose d'un ensemble de privilèges accordés :

- directement à la connexion utilisée
- indirectement par l'intermédiaire d'un rôle fixe de serveur accordé à la connexion
- indirectement par l'intermédiaire des rôles accordés à l'utilisateur de la base de données
- directement à l'utilisateur de la base de données



Exercice:

- 1) Créer une connexion Test_log avec le mot de passe 123456,
- 2) Créer un utilisateur UserI et l'associer à la connexion Test_log
- 3) Donner a cette connexion le rôle serveur de créer des BDD.
- 4) Donner le droit à UserI de consulter et de mettre à jour la table Client de la BDD GSTOCK.
- 5) Retirer le droit de mise à jour de la table Client pour UserI
- 6) Créer un schéma vide au niveau de la BDD GSTOCK et lui transférer la table dbo.client