|  |  |
| --- | --- |
|  |  |
| **SUPPORT DE COURS SLAM1** | |

**SQL**

|  |  |  |
| --- | --- | --- |
| **date** | **révision** | |
| 2019 | Timothée Robert : création du document | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  | |  |

|  |  |
| --- | --- |
| Sécurité des données |  |

## Sources

SQL Server 2014 : F. Brouard & alii, Eyrolles

Discovering SQL: A. Kriegel, Wrox (Wiley)

Beginning SQL : P. Wilson & alii, Wrox (Wiley)

## Introduction

La sécurité des bases de données est gérée le plus souvent à plusieurs niveaux :

* **Système**:
  + Authentification : pas d’accès généralisé (login) au serveur où se situe la Base de données
  + Droits – autorisations : pas d’accès généralisé en lecture et écriture aux partitions ou fichiers où se situent les données du SGBD

Rmq : au moment de la mise en place de la base de données, l'administrateur système va autoriser à la base de données d'écrire sur différents disques, de s'étendre jusqu'à un certain niveau, d'utiliser un certain nombre de ressources telles que RAM, processeurs etc

* **Base de données:** 
  + Authentification : accès aux utilisateurs de confiance seulement
  + Droits – autorisations pour la base de données : instructions GRANT et REVOKE pour définir et limiter les opérations possibles pour chaque utilisateur ou groupe.
* **Applicatif** : le concepteur de l'application, logiciel ou progiciel qui effectue les accès à la base de la données met en place une authentification dédiée pour l'application, ou qui s'appuie sur un annuaire LDAP
* **Modélisation**: le concepteur de la base de données va utiliser des Vues SQL pour sécuriser et synthétiser l’accès aux données

Le périmètre de ce cours aujourd’hui concerne les parties Base de données et Applicatif.

La partie Système est vue en SISR et la partie Modélisation a été abordée la semaine dernière avec les Vues

## Généralités

Seules les tables contiennent des données. Mais pour l’utilisateur, une vue apparaît comme une table et permet, en plus du contrôle d’accès aux données :

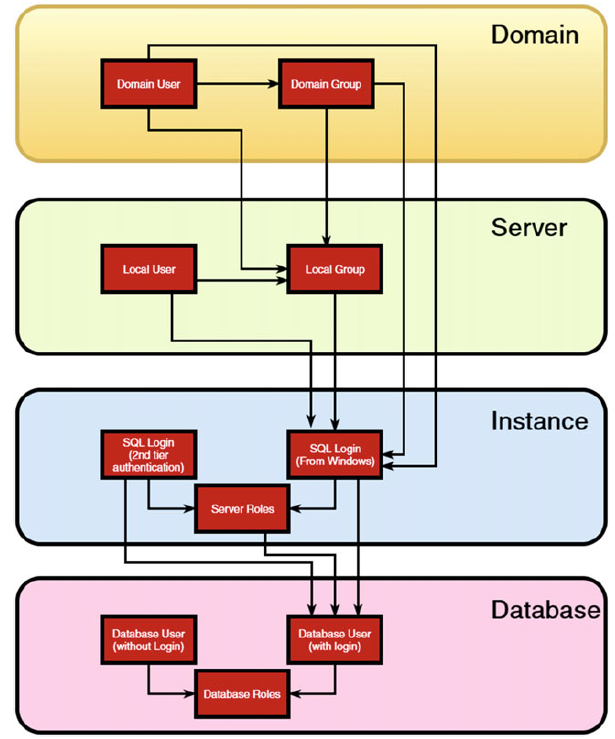
* Masquer la complexité un modèle relationnel
* Simplifier les requêtes complexes

# La sécurité sous la Base de données SQL Server

## Authentification dans SQL Server 2016

### Principes d’authentification

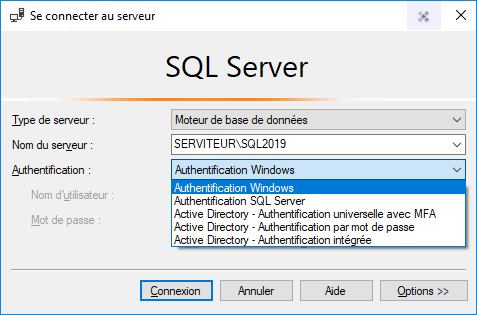
Le schéma ci dessous décrit comment fonctionne l'authentification dans Microsoft SQL Server 2016:



### Principaux modes d’authentification

L’authentification dans Microsoft SQL Server peut être d’une forme suivante :

* **Authentification SQL Server** : c’est celle que vous utilisez, qui est indépendante du système et du domaine (INFOGEST)
* **Authentification Windows** : elle se base sur la même authentification que l’utilisateur effectue pour se logger sur son poste de travail. Le mode d’authentification Windows permet à l’utilisateur de se connecter au moyen d’un compte d’utilisateur Windows
* **Les autres modes sont liés à AZURE** avec Active Directory, qui se base sur LDAP (cf plus bas définition en annexe).



## Les droits dans SQL Server

### Etapes pour sécuriser des objets SQL Server

Les étapes sont les suivantes :

* Création d’un Login via Compte Windows ou SQL pur
* Création d’un utilisateur lié à un Login
* Création de rôle
* Attribution de rôles à des utilisateurs
* Octroi ou suppression de droits sur des objets à des rôles

### Création d’un Login ou Connection

2 possibilités :

1. A partir d’un compte Windows :

-- Create a login for SQL Server by specifying a server name and a Windows domain account name.

CREATE LOGIN [<domainName>\<loginName>] FROM WINDOWS;

GO

1. Création d’un compte SQL Server uniquement

- Creates the user "shcooper" for SQL Server using the security credential "RestrictedFaculty"

-- The user login starts with the password "Baz1nga," but that password must be changed after the first login.

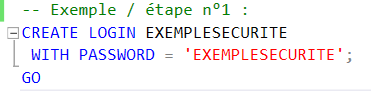
CREATE LOGIN shcooper

WITH PASSWORD = 'Baz1nga' MUST\_CHANGE,

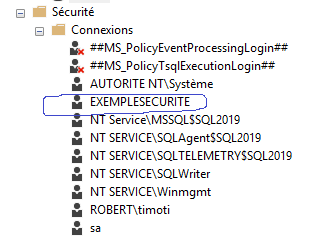
CREDENTIAL = RestrictedFaculty;

GO

Exemple n°1: creation d’un compte SQL Server uniquement nommé EXEMPLESECURITE



Le compte créé sera visible dans l’onglet Sécurité\Connexions de l’INSTANCE du server SQL Server considéré.



### Création d’un utilisateur

Chaque SGBD a sa syntaxe et ses fonctions particulières pour définir des utilisateurs en mode « intégré » (Built in).

Sous Microsoft SQL Server, on utilise l’instruction suivante :

**Syntaxe**

-- Syntax for SQL Server Database

-- Syntax Users based on logins in master

CREATE USER user\_name

[

{ FOR | FROM } LOGIN login\_name

]

[ WITH <limited\_options\_list> [ ,... ] ]

[ ; ]

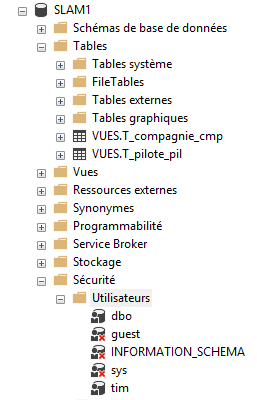
**Exemple / étape n°2** :



***Explications***:

On crée un utilisateur nommé tim associé au compte SQL Server créé nommé EXEMPLESECURITE

L’utilisateur créé sera visible dans l’onglet Sécurité\Utilisateurs de la BASE DE DONNES associée.



### Création d’un rôle

# L’instruction CREATE ROLE permet de créer un rôle SQL sous la base de données Microsoft SQL Server.

Seul le propriétaire de la base de données peut créer un rôle.

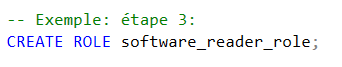
Un rôle permet de regrouper des utilisateurs. On affecte des rôles à des utilisateurs au moyen de l’instruction GRANT.

**Syntaxe**

**CREATE ROLE** [***roleName***](https://db.apache.org/derby/docs/10.8/ref/rrefrolename.html#rrefrolename)

Remarque: certaines bases de données ont des fonctions plus avancées et définissent également des groupes, permettant de regrouper des utilisateurs.

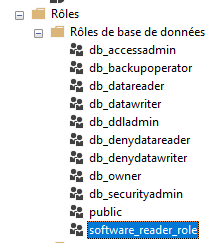
**Exemple: étape n° 3**:



***Explications***:

Un rôle nommé software\_reader\_role est crée. L’instruction ALTER ROLE permettra d’attribuer des rôles à des utilisateurs Ensuite l’instruction GRANT permettra d’affecter des utilisateurs à ce rôle.

Le rôle créé sera visible à l’emplacement Rôles\Rôles de base de données de la BASE DE DONNEES considérée.



### Attribution de rôle à des utilisateurs

**Syntaxe**

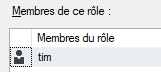
ALTER ROLE <role\_name> ADD MEMBER <user\_name>

Assigne le role [***role\_Name***](https://db.apache.org/derby/docs/10.8/ref/rrefrolename.html#rrefrolename)aux utilisateurs mentionnés dans [***user***](https://db.apache.org/derby/docs/10.8/ref/rrefsqljgrant.html#rrefsqljgrant__grantgrantees)***\_name.***

**Etape 4 / exemple 4**:



Les membres d’un rôle sont visibles lorsqu’on examine les propriétés du rôle (click droit 🡪 Propriétés).

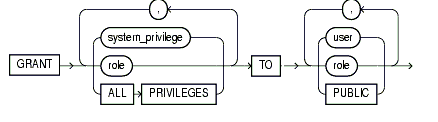


### Attribution de droits sur des objets à un rôle : instruction GRANT

L’instruction GRANT permet d’accorder des droits, nommés privilèges, à des utilisateurs ou à des rôles.

Dans le cadre de ce cours, on distingue 2 formes particulières de l’instruction GRANT :

* Assigner des rôles à des utilisateurs
* Assigner des droits (privilèges) à des utilisateurs ou des rôles



GRANT <permission> [ ,...n ] ON

[ OBJECT :: ][ schema\_name ]. object\_name [ ( column [ ,...n ] ) ]

TO <database\_principal> [ ,...n ]

[ WITH GRANT OPTION ]

[ AS <database\_principal> ]

<database\_principal> ::=

Database\_user

| Database\_role

| Application\_role

| Database\_user\_mapped\_to\_Windows\_User

| Database\_user\_mapped\_to\_Windows\_Group

| Database\_user\_mapped\_to\_certificate

| Database\_user\_mapped\_to\_asymmetric\_key

| Database\_user\_with\_no\_login

**Exemple / étape 5 :**

Nous voulons autoriser l’utilisateur tim , via son rôle software\_reader\_role, à effectuer des instructions SELECT sur toutes les colonnes de la table SECURITE.REF\_LIB\_REGION.



***Explications*** :

1. les rôles permettent d’éviter des erreurs, et de gagner en productivité, puisqu’on peut assigner plusieurs

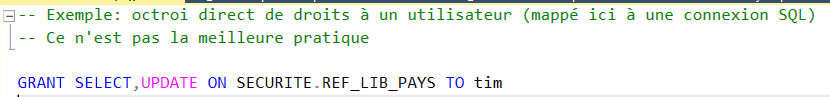
utilisateurs à un ou plusieurs rôles, et ensuite préciser quel rôle est actif.

Ces précisions peuvent être fournies par l’application, ce qui permet de combiner une sécurité Base de données avec une sécurité applicative.

1. Cela évite de donner des droits inconditionnels à des utilisateurs sur des tables, pour des UPDATE ou DELETE par

exemple. Via les rôles, c’est seulement dans un ***contexte*** donné, qu’ils peuvent avoir ces droits.

**Remarque** : on peut accorder donc des droits directement à un utilisateur (database\_user) ou un utilisateur mappé à un user windows, mais ce n’est pas la meilleure pratique : on préfère passer par la notion de rôle.



### Suppression de rôle pour des utilisateurs et suppression de droits

**Suppression de rôle pour un utilisateur**

**REVOKE** [***roleName***](https://db.apache.org/derby/docs/10.8/ref/rrefrolename.html#rrefrolename) **FROM** [***user***](https://db.apache.org/derby/docs/10.8/ref/rrefsqljgrant.html#rrefsqljgrant__grantgrantees)

Révoque, c'est-à-dire enlève le role [***roleName***](https://db.apache.org/derby/docs/10.8/ref/rrefrolename.html#rrefrolename)aux utilisateurs mentionnés dans [***user***](https://db.apache.org/derby/docs/10.8/ref/rrefsqljgrant.html#rrefsqljgrant__grantgrantees)

**Suppression de droits pour un rôle**

**REVOKE** [***privilege*-type**](https://db.apache.org/derby/docs/10.8/ref/rrefsqljgrant.html#rrefsqljgrant__grantprivtype) **ON OBJECT:: {** [**table-Name**](https://db.apache.org/derby/docs/10.8/ref/rreftablename.html#rreftablename)**} FROM** [**grantees**](https://db.apache.org/derby/docs/10.8/ref/rrefsqljgrant.html#rrefsqljgrant__grantgrantees)

Exemple 6 :

Nous n’avons plus besoin d’autoriser les utilisateurs ayant le rôle software\_reader\_role à accéder à la table SECURITE.REF\_LIB\_REGION.



***Explications*** :

Lors d’un contexte SOFTWARE\_READER\_ROLE, les utilisateurs assigné au rôle SOFTWARE\_READER\_ROLE n’auront plus d’autorisation de SELECT pour la table SECURITE.REF\_LIB\_REGION.

# Prolongements

## En savoir plus

Assurer la sécurité d'une base de données, ça consiste en maintenir:

1. la confidentialité

2. l’intégrité et

3. la disponibilité des données

## Maintien de la confidentialité

Il s’agit de détecter ou d’empêcher des accès non autorisés.

C’est important notamment dans les contextes comme:

- les environnements critiques ou stratégiques : militaires ou commerciaux, par exemple.

- en accord avec des contraintes CNIL: pour respecter le droit des individus à décider comment et dans quel but les informations les concernant peuvent être extraites, mémorisées ou transmises à d’autres individus.

## Maintien de l’intégrité et de la disponibilité

### Maintien de l’intégrité

Il s’agit de détecter ou d’empêcher des modifications illicites des données qu’elles soient dues à des pannes de système, des manipulations erronées, hacking ou des sabotages.

### Maintien de la disponibilité

Il s’agit de détecter ou d’empêcher des dénis de service.

Il y a déni de service lorsqu’un utilisateur ne parvient pas à accéder dans un délai raisonnable, à une information ou à une ressource pour laquelle il a une autorisation d’accès.

Par exemple, une attaque consistant à saturer un serveur de fausses requêtes empêchant les requêtes valides d’être

exécutées.

## Conclusion :

L’aspect sécurité, intégrité et performance d’une base de données est un véritable ***métier***, celui d’Administrateur de Bases de données.

Nous n’avons examiné qu’un succinct aspect des questions d’intégrité et sécurité, et ignoré pour le moment les questions de performance.

La sécurité implique différents niveaux et couches qui se superposent et interagissent. ***La connaissance des systèmes d’exploitation, de protocoles tels que LDAP et des bases de données sont un pré requis indépassable dorénavant***.