
# LABORATÓRIO 7

## SEGURANÇA

## COMPLEMENTOS DE BASES DE DADOS

### Licenciatura em Engenharia Informática

Grupo Nº 6

Alexandre Coelho, Nº 190221093

Sérgio Veríssimo, Nº 190221128

# Índice

# I – DEFINIÇÃO DE UTILIZADORES (USERS)

1. **Crie o schema [Developer_Schema].**

```
create schema [Developer_Schema];
```

2. **Crie 2 tabelas [table1] e [table2] com 2 colunas cada, sendo que a [table2] pertence ao schema criado na alínea anterior.**

```
create table [table1](
[id] [int] IDENTITY(1,1) NOT NULL
,[name] varchar(50) NULL);

create table [Developer_Schema].[table2](
[id] [int] IDENTITY(1,1) NOT NULL
,[name] varchar(50) NULL);
```

3. **Crie 3 logins.**

```
CREATE LOGIN [UserS1] WITH PASSWORD = '123';
CREATE LOGIN [UserS2] WITH PASSWORD = '321';
CREATE LOGIN [UserS3] WITH PASSWORD = '132';
```

4. **Crie 3 utilizadores, um para cada login, com o schema Developer_Schema pré-definido.**

```
CREATE USER [User1] FOR LOGIN [UserS1]
WITH DEFAULT_SCHEMA = [Developer_Schema];

CREATE USER [User2] FOR LOGIN [UserS2]
WITH DEFAULT_SCHEMA = [Developer_Schema];

CREATE USER [User3] FOR LOGIN [UserS3]
WITH DEFAULT_SCHEMA = [Developer_Schema];
```

# II – DEFINIÇÃO DE PERMISSÕES (ROLES)

5. **Defina 3 tipos de permissão (através de script) para acesso à BD (schema *dbo*), da seguinte forma:**

a. ***Developer***: Tem todo o tipo de acesso aos dados (ver, inserir, modificar, etc.), e pode criar/apagar tabelas no schema *Developer_Role*.

b. ***Client***: Tem acesso para consulta/inserção/modificação de dados sobre tabelas existentes.

c. ***Reader***: Só pode consultar os dados existentes.

```
--Developer—
```

```
CREATE SERVER ROLE Developer;

GRANT CONTROL ON SCHEMA::[Developer_Schema] TO
Developer;

--Client—

CREATE SERVER ROLE Client;

GRANT SELECT, INSERT, ALTER ON
SCHEMA::[Developer_Schema] TO Client;

--Reader—

CREATE SERVER ROLE Reader;

GRANT SELECT, INSERT, ALTER ON
SCHEMA::[Developer_Schema] TO Client;
```

6. **Atribua cada uma das roles ao respetivo utilizador definido no ponto 3.**

```
--Developer—

ALTER SERVER ROLE Developer ADD MEMBER [UserS1];

--Client--

ALTER SERVER ROLE Client ADD MEMBER [UserS2];

--Reader--

ALTER SERVER ROLE Reader ADD MEMBER [UserS3]
```

# III – ACESSOS E RESTRIÇÕES

7. **Verifique graficamente a configuração das Roles, bem com a associação das mesmas a cada utilizador**
8. **Faça login com o utilizador UserS1 e:**

```
EXECUTE AS LOGIN = 'UserS1';
```

a. insira uma linha na tabela [table1] e outra na [table2]

```
INSERT INTO [table1] VALUES ('TesteUserS1T1');

INSERT INTO [Developer_Schema].[table2] VALUES
('TesteUserS1T2');
```

O utilizador não consegue inserir valores na table1, devido a pertencer ao schema [dbo], enquanto que ao inserir na table2, já lhe é permitido devido ás permissões fornecidas anteriormente.

b. crie a tabela [table3] com este utilizador com o schema [dbo]

```
create table [dbo].[table3](

[id] [int] IDENTITY(1,1) NOT NULL

,[name] varchar(50) NULL);
```

O UserS1, não consegue criar a tabela no schema [dbo], devido a não ter permissões para alterar algo no schema [dbo].

c. crie a tabela [table3] com este utilizador com o schema [Developer_Schema]

```
create table [Developer_Schema].[table3](
[id] [int] IDENTITY(1,1) NOT NULL
,[name] varchar(50) NULL);
```

Agora é permitido ao utilizador criar a tabela, devido a ter permissões para criar tabelas no schema definido.

**9. Faça login com o utilizador UserS2 e:**

```
REVERT
EXECUTE AS LOGIN = 'UserS2';
```

a. insira uma linha na tabela [table1] e outra na [table2]

```
INSERT INTO [table1] VALUES ('TesteTable1UserS2');
INSERT INTO [Developer_Schema].[table2] VALUES
('TesteTable2UserS2');
```

De forma igual ao UserS1, o cliente também pode inserir dados na table2, devido às permissões fornecidas anteriormente (Client) enquanto que quanto à inserção na table1, por este não conter permissões de inserção no schema [dbo], não lhe será permitida a inserção.

b. crie a tabela [table4] com este utilizador com o schema [dbo]

```
create table [dbo].[table4](
[id] [int] IDENTITY(1,1) NOT NULL
,[name] varchar(50) NULL);
```

Não é permitido a este utilizador (UserS2) criar a tabela dentro do schema [dbo], devido a este não possuir qualquer permissão sobre o schema [dbo].

c. crie a tabela [table4] com este utilizador com o schema [Developer_Schema]

```
create table [Developer_Schema].[table4](
[id] [int] IDENTITY(1,1) NOT NULL
,[name] varchar(50) NULL);
```

Não é permitido a este utilizador (UserS2) criar a tabela dentro do schema Developer_Schema, pois este só pode efetuar consulta/inserção/modificação de dados sobre o schema.

**10. Faça login com o utilizador UserS3 e:**

```
REVERT
EXECUTE AS LOGIN = 'UserS3';
```

a. insira uma linha na tabela [table1] e outra na [table2]

```
INSERT INTO [table1] VALUES ('TesteTable1UserS2');
INSERT INTO [Developer_Schema].[table2] VALUES
('TesteTable2UserS2');
```

Este utilizador não possui permissão para inserir em ambas as tabelas (dos dois schemas). Este apenas tem permissão para listar os dados do schema [Developer_Schema].

b. liste os valores inseridos em cada tabela

```
Select * from [table1];
Select * from [Developer_Schema].[table2];
```

O utilizador UserS3 não consegue listar os dados da table1, por não ter qualquer permissão para o schema [dbo]. Mas consegue listar os dados da table2, devido a pertencer ao schema [Developer_Schema], onde este possui como única permissão fornecida anteriormente a listagem.

**11. Comente os resultados obtidos nas 3 alíneas anteriores.**

Os resultados encontram-se comentados nas alíneas a cima.

# IV– ENCRIPTAÇÃO

**Utilizando com o seu user default…**

**12. Insira algumas linhas na tabela [table1]**

```
INSERT INTO [dbo].[table1] (name)
VALUES ('Teste1'),
       ('Teste2'),
       ('Teste3');
```

**13. Crie uma master key com uma chave à sua escolha**

```
Create MASTER KEY ENCRYPTION
BY PASSWORD = 'PasswordSuperSegura';
```

**14. Crie um certificado de encriptação**

```
Create CERTIFICATE EncryptCertificate
With subject='Secure data';
```

**15. Crie uma chave simétrica com o algoritmo AES256 utilizando o certificado definido na alínea anterior**

```
Create SYMMETRIC KEY TableSynKey
With Algorithm = AES_256
ENCRYPTION BY CERTIFICATE EncryptCertificate;
```

**16. Altere a tabela [table1] e adicione a coluna [EncryptName] do tipo VARBINARY(256)**

```
ALTER TABLE [dbo].[table1]
ADD encryptName VARBINARY(256);
```

**17. Atualize a nova coluna com os dados encriptados da coluna [name] criados com o certificado e chave definidos anteriormente**

```
OPEN SYMMETRIC KEY TableSynKey DECRYPTION BY
CERTIFICATE EncryptCertificate;
UPDATE [dbo].[table1]
SET [encryptName] =
  ENCRYPTBYKEY(KEY_GUID('TableSynKey'),[name]);
Close SYMMETRIC KEY TableSynKey;
```

**18. Liste a tabela [table1] e comente o resultado**

```
SELECT * FROM [dbo].[table1];
```

Ao executar o select a siga obtenho o id, o name e o name encriptado em AES256.

| | id | name | encryptName |
|---|---|---|---|
| 1 | 1 | Teste1 | 0x007A78EFF4689F459C74384102A67A25020000009F5BB6DFAF53C40A37ABADB3476117E44E8B4A235D715799D0CF3CCB45AC6A86 |
| 2 | 2 | Teste2 | 0x007A78EFF4689F459C74384102A67A2502000000557D887A7DB845FF39D9E2519148EFB6C8E87D813559799312C4B3663216593B |
| 3 | 3 | Teste3 | 0x007A78EFF4689F459C74384102A67A250200000032E65A946939BCD44F4D383752922E206509464A34F69D465A30A80E5D2F13EE |

**19. Liste a tabela [table1] desencriptando a coluna [EncryptName] de forma a devolver o valor original**

```
OPEN SYMMETRIC KEY TableSynKey DECRYPTION BY
CERTIFICATE EncryptCertificate;
Select *,
convert(Nvarchar(50),DECRYPTBYKEY(encryptName)) as
'Decrypted Name'
From [dbo].[table1]
Close SYMMETRIC KEY TableSynKey;
```