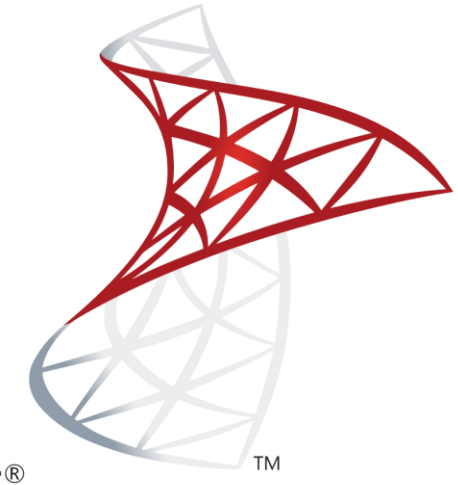
TP02 - Controlo de Acessos e Segurança

João Ramos (20200255)
Martim Bento (20200488)
Pedro Cunha (20200908)

25/04/2022



Microsoft® SQL Server®

Faculdade de Design, Tecnologia e Comunicação
Universidade Europeia

Equipa de pesquisa

Apresentação de competências

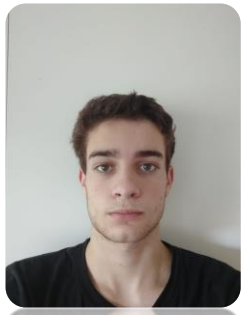
- João Ramos
- E-mail: 20200255@iade.pt

Hard Skills

- Curso técnico profissional de gestão de equipamentos informáticos
- Python, CSS, JavaScript, C#, Arduino, Java, HTML , SQL

Soft Skills

- Liderança, organização, flexibilidade , Aprendizagem Rápida



- Martim Bento
- E-mail: 20200488@iade.pt

Hard Skills

- Ensino Secundário - Ciências e Tecnologias.
- Java, Python, HTML, CSS, JavaScript, SQL

Soft Skills

- Espirito de colaboração, flexibilidade e empenho.

- Pedro Cunha
- E-mail: 20200908@iade.pt

Hard Skills

- Ensino Secundário - Ciências e Tecnologias
- Python, Java, HTML, CSS, Javascript, R, SQL
- Bases de Dados (16), Fundamentos de Programação (15), Desenvolvimento de Interfaces Web (16), Criatividade e Pensamento Crítico (17)

Soft Skills

- Organização, empenho, flexibilidade, colaboração

Segurança

Introdução

- **Segurança da base de dados**: Assegurar a Confidencialidade, Integridade e Disponibilidade das informações
- **Regulamento Geral de Proteção de Dados (RGPD):** intensificou esta necessidade, visto que estabelece as regras relativas ao tratamento de dados pessoais pertencentes a indivíduos da União Europeia, quer seja por um particular, uma empresa ou uma organização.
- Isto leva a um crescimento no setor da Cibersegurança, visto que, a segurança da informação tem um papel bastante importante para garantir a melhoria do desempenho de uma empresa.

Segurança

O que é a segurança nas bases de dados?

- Numa base de dados estão reunidas diversas informações sobre um negócio, que são acedidas pelos diferentes utilizadores da base de dados.
- Para que a segurança da base de dados não seja comprometida por acidentes ou ataques, é preciso assegurar três pilares:
  - **Confidencialidade**: garantir que a informação só é acedida por pessoas que possuam autorização para tal
  - **Integridade**: a informação é alterada apenas por indivíduos autorizados
  - **Disponibilidade**: garantir que as pessoas autorizadas obtenham a informação que pretendem quando necessário.

Segurança

A importância da segurança nas bases de dados

- A segurança numa base de dados compreende medidas que visam a proteção dos dados em duas vertentes:
  - Evitar que os dados caiam nas mãos de pessoas não autorizadas (hackers)
  - Quem tem de facto acesso autorizado aos dados, consoante o role de um utilizador.
- A importância deste controlo está na sobrevivência da empresa:
  - Na base de dados estão implementadas regras e lógicas de negócio. A recolha e proteção de dados recolhidos é essencial para orientar uma empresa no mercado:
    - descobrir o perfil dos clientes
    - definir estratégias de venda de modo a manter a competitividade no mercado.
  - Por este motivo, as organizações investem mais em Big Data, Machine Learning e Inteligência Artificial, para facilitar o estudo dos dados e permitem prever quais as decisões mais beneficiantes.

Segurança

Como garantir a segurança das bases de dados?

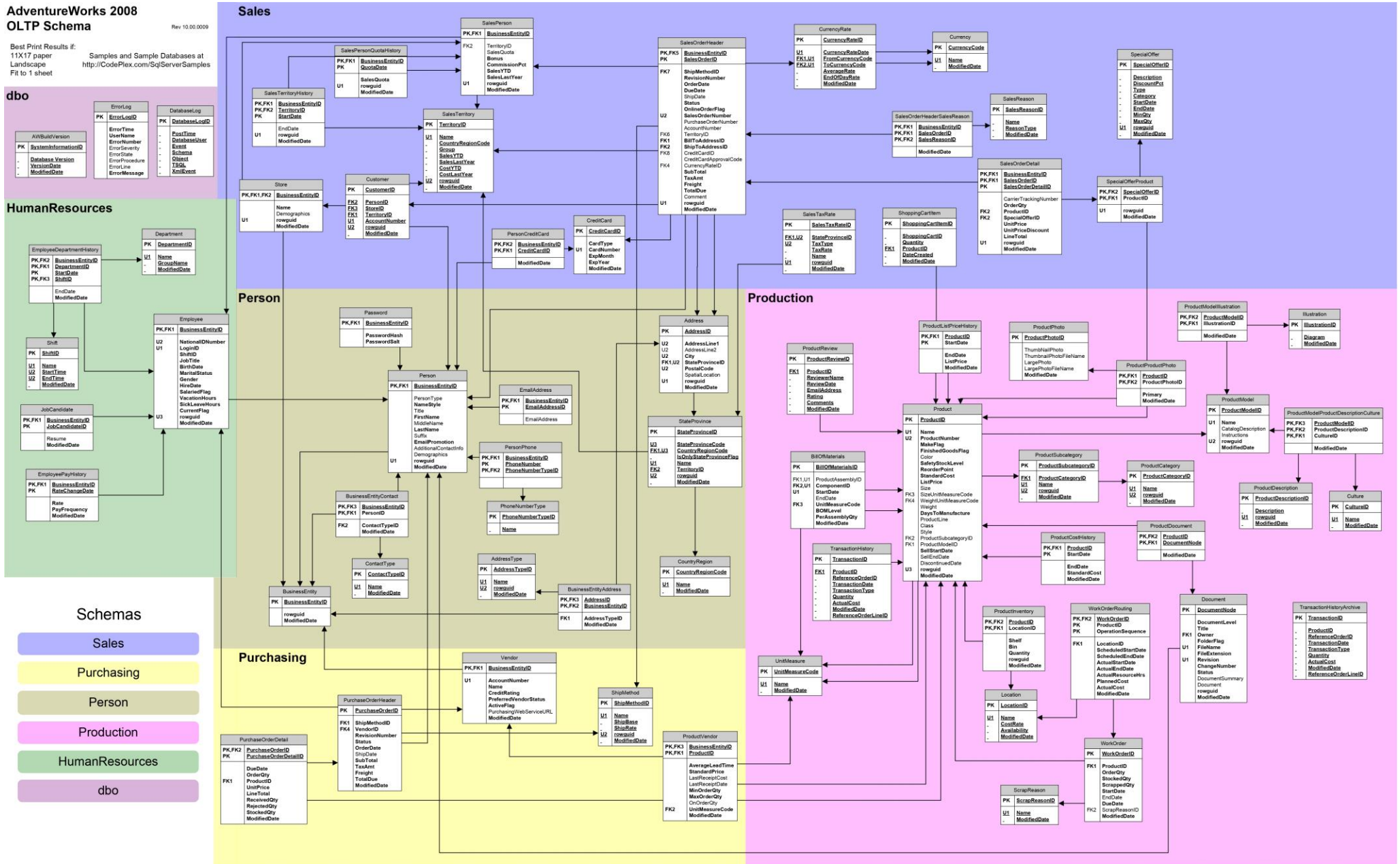
- Garantir a segurança de uma base de dados contra acessos indevidos e acidentes a 100% é uma tarefa quase impossível:
  - Estão sempre a surgir novas abordagens desenvolvidas por atores com intenções maliciosas e, consequentemente, a adaptação da base de dados para não ser suscetível a estas vulnerabilidades tem de ser constante.
  - Podemos recorrer a algumas medidas simples, que tornam as bases de dados mais protegidas e seguras, o que leva a que sejam menos vulneráveis e à mitigação do risco, como por exemplo:
    - Criação de Roles
    - Criação de Schemas
    - Encriptação de dados
    - Criação de Políticas de Backups

Schemas

O que são?

- Um Schema é uma lista de estruturas lógicas de dados. Um utilizador da base de dados possui o schema que possui o mesmo nome do administrador da base de dados. A partir do SQL Server 2005, um Schema é uma entidade individual (contentor de objetos) distinta do utilizador que constrói o objeto. Por outras palavras, schemas é como se fossem “contentores” usados para manipular tabelas da base de dados.
- Os schemas podem receber permissões de segurança, tornando-os num método eficaz para distinguir e defender objetos de base de dados com base nos privilégios de acesso que cada utilizador possui.

```
CREATE SCHEMA [DBA] AUTHORIZATION [dbo]
GO
```


AdventureWorks 2008
OLTP Schema
Rev 10.00.0009
Best Print Results if:
11X17 paper
Landscape
Fit to 1 sheet
Samples and Sample Databases at
http://CodePlex.com/SqlServerSamples
dbo
HumanResources
Sales
Person
Production
Purchasing
Schemas
Sales
Purchasing
Person
Production
HumanResources
dbo

Schemas

Vantagens

- Podem ser atribuidos facilmente a um utilizador.
- Podem ser partilhados entre vários utilizadores (Roles).
- Permitem transferir objetos da base de dados entre schemas.
- Permitem um beneficio no poder sobre os acessos e proteção dos objetos da base de dados.
- Um utilizador pode ser removido sem remover objetos conectados a este utilizador.
- Permite que varias entidades colaborarem dentro da mesma base de dados e ao mesmo tempo manter a credibilidade e integridade da base de dados

Roles

O que são?

- Quando se gere uma base de dados é importante manter a regra dos privilegios que permite aos seus utilizadores acederem apenas aos dados que necessitam para realizar as operações necessárias tendo em conta o seu cargo.
- Os roles permitem agrupar logins de utilizadores e gerir o nível de permissões que estes utilizadores devem ter , existem dois tipos de roles:
  - **Fixed server roles:** São roles default do SQL Server e nao são modificaveis
  - **User-defined roles:** São roles que podem ser criados e que são customizáveis consoante as necessidades dos requisitos de segurança e utilização.

```
CREATE ROLE [DBA] AUTHORIZATION [dbo]
GO
```

Roles

Fixed server roles (Built-in)

- **public:** Role default para utilizadores que não possuam permissões específicas. Deve-se atribuir apenas permissões public a objetos que possam ser disponibilizados para todos os users, pois não se pode retirar permissões public a nenhum role do servidor.
- **dbcreator:** Permite alterar, criar, eliminar e restaurar bases de dados.
- **diskadmin:** Permite a gestão de discos.
- **bulkadmin:** Permite executar bulk insert.
- **setupadmin:** Permite adicionar ou remover servidores e correr t-sql.
- **processadmin:** Permite encerrar processos na instância do sql server.
- **securityadmin:** Permite gerir os logins, podendo redifinir passwords e atribuir, negar e reverter permissões.
- **serveradmin:** Permite alterar a configuração do servidor e encerrar o mesmo.
- **sysadmin:** Permite fazer qualquer atividade.

Encriptação

Enquadramento

- Certas informações pessoais como passwords ou números de cartão de crédito necessitam de uma segurança adicional ao serem guardados na base de dados, de modo a garantir a sua integridade e confidencialidade.
- Para dar resposta a isto é utilizada a **Encriptação:** altera os dados originais convertendo-os para um formato codificado. Os dados encriptados só podem ser lidos ou processados depois de serem desencriptados.

Existem três tipos principais de encriptação:

- **Encriptação ao nível da BD (TDE - Transparent Data Encryption):** Serve para encriptar a base de dados como um todo incluindo os logs, por exemplo, no entanto a performance é comprometida.
- **Encriptação ao nível das colunas**: Somente as colunas que guardam dados sensíveis são encriptadas, possui uma melhor performance na maior parte dos casos e é o tipo de encriptação mais utilizada.
- **Always Encrypted**: Protege dados confidenciais, permite que sejam encriptados e que as aplicações nunca revelem as chaves para o mecanismo do SQL Server, ou seja, garante a separação entre quem detém os dados e quem os gere.

Encriptação

Conceitos

- **Master Key**: Topo da hierarquia da encriptação do SQL Server, cada base de dados possui uma master key, que é utilizada para proteger as chaves privadas dos certificados e chaves assimétricas e simétricas. Para criar uma Master Key é utilizado o seguinte comando (obrigatório definir senha):

```
CREATE MASTER KEY ENCRYPTION
BY PASSWORD = 'pass'
GO
```

- **Certificado**: Também responsável por proteger as chaves simétricas e assimétricas. Para criá-lo, é necessário que antes exista uma Master Key, devemos criar uma senha, que será utilizada para proteger os próximos objetos da hierarquia. Para criar um certificado usamos o seguinte comando:

```
CREATE CERTIFICATE CertificadoTest1
ENCRYPTION BY PASSWORD = 'password'
WITH SUBJECT = 'CertificadoTest'
GO
```

Encriptação

Conceitos

- **Chave Simétrica**: uma única chave é usada para encriptar e desencriptar os dados, o emissor e recetor partilham a mesma chave, é mais simples e mais rápida tendo uma melhor performance. Alguns dos algoritmos que se podem utilizar são: DES, AES e RC.

```
CREATE SYMMETRIC KEY ChavesimetricaTest
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE CertificadoTest1
```

- **Chave assimétrica**: composta por uma chave privada e uma chave pública. A chave pública é usada pelo emissor para encriptar os dados e a chave privada usada pelo recetor para desencriptar os dados. Fornece um nível mais alto de segurança em deterioramento da performance. Alguns dos algoritmos que se podem utilizar são: RSA, DAS.

```
CREATE ASYMMETRIC KEY chaveAssimetrica1
WITH ALGORITHM = RSA_2048
ENCRYPTION BY PASSWORD = 'Pass'
```

Encriptação

Conceitos

- **Database Encryption Key:** É utilizada no TDE e trata-se de uma chave simétrica que é protegida por outras chaves ou certificados que são protegidos pela Master Key da base de dados ou por uma chave assimétrica armazenada num módulo de Extensible Key Management (EKM)

```
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_256
ENCRYPTION BY SERVER CERTIFICATE CertificadoTest1
```

- Podemos consultar as chaves simétricas, assimétricas e certificados existentes na base de dados através dos através dos seguintes comandos:

```
SELECT * FROM SYS.symmetric_keys

SELECT * FROM  SYS.asymmetric_keys

SELECT * FROM SYS.certificates
```

Encriptação

Encriptar

- **Chave Simétrica**

```
OPEN SYMMETRIC KEY ChavesimetricaTest
DECRYPTION BY CERTIFICATE CertificadoTest1

Update MetPagamento
set metnumCartao_encrypt =ENCRYPTBYKEY(key_guid('ChavesimetricaTest'),Met_numCartao) from MetPagamento
CLOSE SYMMETRIC KEY ChavesimetricaTest
```

```
(2 rows affected)
A chave Simetrica a encriptar demorou (ms): 3
```

- **Chave assimétrica**

```
UPDATE Sadmin.Cliente
SET TelemovelEncriptado = ENCRYPTBYASYMKEY(ASYMKEY_ID('chaveAssimetrica1'), CAST(Cl_Telemovel AS varchar)) ,
    EmailEncriptado = ENCRYPTBYASYMKEY(ASYMKEY_ID('chaveAssimetrica1'),Cl_email)
FROM Sadmin.Cliente
```

```
A chave assimetrica a encriptar demorou (ms): 54
```

Encriptação

Desencriptar

- **Chave Simétrica**

```
SELECT met_ano, met_mes,met_numcartao,met_cvv,metnumCartao_encrypt AS 'Numero de Cartao Ecriptado',
CONVERT(varchar, DecryptByKey(metnumCartao_encrypt)) AS 'Numero de cartao desencriptado'
FROM MetPagamento;
```

```
A chave simetrica a desencriptar demorou (ms): 0
```

- **Chave assimétrica**

```
select *,TelemovelDesencriptado =  Convert(char(11),DECRYPTBYASYMKEY(ASYMKEY_ID('chaveAssimetrica1'), TelemovelEncriptado, N'Pass'))
FROM sadmin.cliente

select *,EmailDesencriptado =  Convert(char(11),DECRYPTBYASYMKEY(ASYMKEY_ID('chaveAssimetrica1'), EmailEncriptado, N'Pass'))
FROM sadmin.cliente;
```

```
A chave assimetrica a desencriptar demorou (ms): 73
```

Demonstração

Yes we Can!

Grupo 05

Faculdade de Design,
Tecnologia e Comunicação
Universidade Europeia