Complementos de Bases de Dados

2020/2021

Licenciatura em Engenharia Informática

IPS Instituto Politécnico de Setúbal
Escola Superior de Tecnologia de Setúbal

Laboratório 2 – Funções, Stored Procedures, Triggers e Metadados

Objetivos:

- Criação de funções, stored procedures e triggers;
- Utilização dos metadados

ETAPA 1

a) Criar a função *fnTotalVendasProduto* que calcule o valor total das vendas para um determinado produto. O identificador do produto deverá ser passado como argumento. Sugestão para determinar o valor total de vendas (`SUM(OrderQty*UnitPrice))`

b) Utilizando a função *fnTotalVendasProduto*, elabore uma pesquisa (i.e., *query* SQL) que apresente o nome do produto e o respetivo total de vendas;

c) Utilizando a função *fnTotalVendasProduto*, faça uma função *fnTotalVendas* que calcule o total de vendas. Neste caso pretende-se obter o total global de vendas dos produtos registados na BD;

d) Criar o procedimento *spClientesCidade* que recebe uma cidade (e.g., Las Vegas) e lista os clientes residentes na respetiva cidade.

e) Criar o procedimento *spListaCompra*, que apresente para uma determinada compra (*SalesOrderId*) a lista de produtos que a compõe. Utilize cursores e a instrução print para gerar o output (ver exemplo).

```
Exec spListaCompra 71863;
```

```
------------------------------------------------------------
Customer: walterl@adventure-works.com
Order: SO71863
Date: Jun  1 2008 12:00AM
Total: 3673.32
------------------------------------------------------------
Product: Racing Socks, L / OrderQty: 11 / UnitPrice: 5.21 / UnitPriceDiscount: 0.02 / LineTotal: 56.21
Product: Racing Socks, M / OrderQty: 2 / UnitPrice: 5.39 / UnitPriceDiscount: 0.00 / LineTotal: 10.79
Product: Road-750 Black, 48 / OrderQty: 2 / UnitPrice: 323.99 / UnitPriceDiscount: 0.00 / LineTotal: 647.99
Product: Road-350-W Yellow, 40 / OrderQty: 1 / UnitPrice: 1020.59 / UnitPriceDiscount: 0.00 / LineTotal: 1020.59
Product: Sport-100 Helmet, Blue / OrderQty: 1 / UnitPrice: 20.99 / UnitPriceDiscount: 0.00 / LineTotal: 20.99
Product: Road-750 Black, 52 / OrderQty: 3 / UnitPrice: 323.99 / UnitPriceDiscount: 0.00 / LineTotal: 971.98
Product: LL Road Pedal / OrderQty: 2 / UnitPrice: 24.29 / UnitPriceDiscount: 0.00 / LineTotal: 48.59
```

ETAPA 2

a) Criar uma função *fnCodificaPassword* que codifica uma password em SHA1 (utilize a função HASHBYTES). A função recebe a password e retorna a sua codificação.

b) Crie a tabela "CustomerPW", para guardar o ID e a password de novos clientes.

c) Criar o procedimento *spNovoCliente*, que recebe os dados obrigatórios para inserir um novo cliente mais um parâmetro com a nova password (ver na tabela *Customer* qual a lista de atributos obrigatórios), e insere um novo cliente na tabela Customer e o registo correspondente na tabela CustumerPW. A password é guardada em SHA1.

d) Verifique a autenticação (EmailAddress/Password) de um utilizador através de uma função *fnAutenticar*, que devolve o ID do utilizador se a autenticação é válida ou 0 se for inválida.

ETAPA 3

a) Crie a tabela CustomerLog no schema "Logs" (deve ser criado previamente) e os *Triggers* necessários, de modo a implementar um mecanismo de auditoria sobre a tabela Customer. A tabela CustomerLog para além dos atributos da tabela Customer, devem ser adicionados os seguintes atributos:

- Log_Data: *timestamp* da alteração;
- Log_Operacao: 'U' – update, 'D' – delete
- Quando se altera ou se apaga um registo da tabela Customer, deve ser executada uma cópia do registo que sofreu as alterações para a tabela de log, adicionando o *rowversion* e o tipo de operação.

b) Crie os *Triggers* necessários para implementar as seguintes funcionalidades:

- Quando se atualiza a password de um cliente, esta deve ser guardada na tabela CustomerPW com codificação em SHA1;
- Verifica a restrição de não poderem existir utilizadores com o mesmo login (EmailAddress).

ETAPA 4

Utilizando os metadados do servidor SQL Server:

a) Crie um conjunto de *queries* que para uma determinada tabela (e.g., *Customer*) permita:

- Visualizar para todas as colunas, o nome e se essa coluna tem a restrição NOT NULL;
- Visualizar o atributo IDENTITY;
- Visualizar a(s) coluna(s) que constituem a chave primária;
- Visualizar para as chaves estrangeiras, qual o nome da coluna e a tabela/coluna que é referenciada;

b) Crie o stored procedure *sp_disable_FK* que recebe como argumento o nome de uma tabela (@p_table_name), e gera como saída um script (i.e., lista de comandos SQL) que permite fazer *disable* a todas as chaves estrangeiras que fazem referência à tabela.

(fim de enunciado)