INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia de Electrónica e Telecomunicações e de Computadores

LEIC

Sistemas de Informação 2

Semestre de Inverno 2020-2021

Trabalho prático
v1.02
Afonso Remédios e Nuno Datia

Outubro, 2020

Planeamento

As datas importantes a recordar são:

- Lançamento do enunciado: **26 de Outubro de 2020**
- Entrega intermédia (Fase 1): **04 de Dezembro de 2020** (6 semanas)
- Entrega intermédia (Fase 2): **15 de Janeiro de 2020** (5 semanas)

Cada entrega intermédia deve apresentar o relatório e código (se houver) referentes **exclusivamente** a essa fase. O relatório deve seguir um dos modelos fornecidos, obrigatoriamente, sob pena de penalização na nota. Este deve ser conciso e apresentar a justificação de todas as decisões tomadas. A capa do relatório deve indicar a composição do grupo, a unidade curricular e a fase do trabalho que relata. Caso tenha adendas e/ou correcções a fazer a modelos já entregues, deve indicá-las de forma explícita no relatório seguinte.

O pdf (ou, o zip) gerado deve seguir o padrão: 'TPSI2-2021SI-GrupoNNFaseN.ext' (N representa um dígito, e 'ext' a extensão do ficheiro), exemplo: TPSI2-2021SI-Grupo14Fase1.zip.

Objectivos de aprendizagem

No final da **primeira fase do trabalho**, os alunos devem ser capazes de:

- Desenvolver um modelo de dados adequado aos requisitos do sistema, normalizado até à 3NF;
- Conceber e implementar uma solução baseada em bases de dados dinâmicas, adequada aos requisitos;
- Utilizar corretamente controlo transacional;
- Utilizar corretamente níveis de isolamento;
- Utilizar corretamente vistas, justificando o seu uso na solução;
- Utilizar corretamente procedimentos armazenados, justificando o seu uso na solução;
- Utilizar corretamente gatilhos, justificando o seu uso na solução;
- Utilizar corretamente funções, justificando o seu uso na solução;
- Desenvolver código de teste, em T-SQL, para cada uma das funcionalidades requerida nos requisitos;
- Desenvolver código T-SQL para criar todos os objectos necessários à solução, a partir de uma base de dados vazia;
- Escrever um relatório técnico sobre as decisões tomadas e o trabalho desenvolvido.

Enunciado do trabalho (Documento de requisitos do sistema)

A empresa *FSolv* pretende desenvolver um sistema de informação para a gestão de facturas não simplificadas. O sistema tem de ser auditável. Uma factura é identificada por um código que segue o formato: **`FTyyyy-xxxxx`**, onde `yyyy` representa o ano e `xxxxx` representa o número da factura emitida num ano. Mesmo que uma factura seja anulada, o valor de `xxxxx` é monótono crescente. Cada factura é caracterizada por uma data de criação, uma data de emissão (eventualmente nula), um estado, um valor total (sem IVA) e o valor de IVA a pagar. Ambas as datas têm, pelo menos, resolução ao segundo. O valor total e o valor de iva são resultado dos valores associados dos itens existentes na factura. Cada item é identificado por um número dentro da factura e tem uma descrição. É possível definir para cada item da factura um desconto e o número de unidades presentes na factura. Considere que uma factura pode ter os seguintes estados:

- **Emitida**, impossibilitando posteriores alterações à factura;
- **Em actualização**, quando ainda não está finalizada (e.g. falta adicionar itens);
- **Proforma**, impossibilitando qualquer alteração à factura, excepto a passagem para os estado **Emitida** ou **Anulada**;
- **Anulada**, impossibilitando posteriores alterações à factura;

Quando é necessário devolver items presentes numa factura, deve ser criada uma nota de crédito. As notas de crédito são semelhantes a facturas, excepto nos itens que a constituem, que podem ser um subconjunto não vazio dos itens da factura anulada (e referenciada na nota de crédito). Além disso, a identificação da uma nota de crédito é dado por **`NCyyyy-xxxxx`**, seguindo a mesma lógica da identificação das facturas. Uma nota de crédito só pode ter os estados **Em actualização** e **Emitida**. Neste último caso são impossíveis alterações à nota de crédito.

Os itens de uma factura estão associados a produtos. Cada produto é identificado por um SKU[1], tem uma descrição, uma percentagem de iva (eventualmente nula), e o preço de venda unitário (sem iva). Deve ser possível registar o contribuinte associado a uma factura. Um contribuinte tem um número de identificação fiscal, um nome (eventualmente nulo) e uma morada (eventualmente nula). Sempre que é feita uma alteração a uma factura, por exemplo, uma mudança de estado, deve ficar registado no sistema a data em que a alteração foi feita, qual o estado antes da alteração (eventualmente o mesmo).

[1] `https://en.wikipedia.org/wiki/Stock_keeping_unit`

Resultados pretendidos

Tendo em conta os objectivos de aprendizagem, deverão ser produzidos os seguintes resultados:

1. Uma ilustração do modelo de dados (conceptual e relacional), incluindo todas as restrições de integridade;
2. O código T-SQL que permite:
   (a) Criar o modelo físico (1 *script* autónomo);
   (b) Remover o modelo físico (1 *script* autónomo);
   (c) Preenchimento inicial da base de dados (1 *script* autónomo);
   (d) Inserir, remover e atualizar informação de um produto;
   (e) Obter o próximo código de uma factura (ou nota de crédito);
   (f) Criar o procedimento `p_criaFactura` que permite criar uma factura, sem itens, mas com informação de um contribuinte (novo ou existente);
   (g) Criar uma nota de crédito;
   (h) Adicionar itens a uma factura;
   (i) Actualizar o valor total de uma factura;
   (j) Criar uma função para produzir a listagem das notas de crédito de um determinado ano;
   (k) Actualizar o estado de uma factura;
   (l) Criar uma vista que mostre o resumo das facturas (atributos de factura e de contribuinte), que possibilite a alteração do estado de uma factura;
   (m) Testar as funcionalidades de 2c a 2l (1 *script* autónomo).

Garanta que, para os utilizadores da base de dados, todas as funcionalidades de 2c a 2l produzem os resultados esperados, sendo sempre garantidas as restrições de integridade e regras de negócio. Deve reutilizar o codigo entre alíneas. Deve indicar no relatório a forma de testar as funcionalidades. Além disso, deve ser claro quando um teste falha.

Data limite para entrega: 04 de Dezembro de 2020 até às 23:59.

A entrega deve incluir um relatório, para além do código T-SQL, enviados de forma electrónica através do Moodle. O relatório é **entregue** em formato `PDF` (obrigatório).

Nota: Deve ser possível aferir cada um dos objectivos de aprendizagem no material que entregar.

Objectivos de aprendizagem

No final da **segunda fase do trabalho**, os alunos devem ser capazes de:

- Desenvolver uma aplicação em C#, que use diferentes implementações de acesso a dados;
- Utilizar corretamente processamento transaccional, concretizado através de mecanismos disponíveis na plataforma .NET;
- Utilizar corretamente ADO.NET em modo "conectado", para acessos a dados;
- Utilizar corretamente (ADO.NET) Entity Framework para acessos a dados;
- Garantir a correta libertação de ligações e recursos, quando estes não estejam a ser utilizados;
- Garantir a correta implementação das restrições de integridade e/ou lógica de negócio;
- Organizar o código de acesso a dados usando padrões de desenho, nomeadamente, `Data Mapper`, `Lazy load`, `Data Transfer Object` e `Virtual Proxy`;
- Organizar o código de acesso a dados para permitir usar processamento transaccional, iniciado nesse código ou existente no contexto de chamada;
- Escrever um relatório técnico sobre o trabalho desenvolvido.

Enunciado do trabalho (2ª fase)

Nesta fase do trabalho pretende-se que os alunos criem uma aplicação que use diferentes *frameworks* de acesso a dados. A aplicação deve ter como meta a reutilização de código, fácil manutenção e eficiência, e ser independente do modo de acesso a dados.

Resultados pretendidos

Tendo em conta os objetivos de aprendizagem, deverão ser produzidos os seguintes resultados:

1. Criação de uma aplicação .NET que, usando uma implementação de acesso a dados desenvolvida usando objetos "conectados" do ADO.NET, permita:
   (a) Aceder às funcionalidades 2f a 2j, descritas na fase 1 deste trabalho;
   (b) Implementar a funcionalidade 2e, descrita na fase 1 deste trabalho, sem usar qualquer procedimento armazenado;
   (c) Implementar a emissão de uma factura, usando conjuntamente as funcionalidades 2e, 2f, 2h, 2i e 2k;
2. Alteração (ou configuração) da aplicação desenvolvida na alínea 1, de forma a que esta use uma implementação de acesso a dados desenvolvida usando Entity Framework (EF). Esta alteração deve ter o mínimo impacto possível;
3. Apresentar testes comparativos de desempenho das tecnologias EF e ADO, na implementação da 1c.
4. Usando EF com optimistic locking, trocar os números de identificação fiscal em duas facturas cujos códigos são fornecidos. Apresente uma mensagem de erro adequada em caso de alteração concorrente conflituante que inviabilize a operação. No relatório deve estar descrita a forma como as situações de erro foram criadas para teste desta alínea.
5. Comparar as tecnologias EF e ADO.NET quanto à garantia da consistência dos dados.

Data limite para entrega: 15 de Janeiro de 2020 até às 23:59.

A entrega deve incluir um relatório (em formato PDF), o código SQL de criação, remoção e preenchimento da base de dados, os projetos Visual Studio e o código C# , submetidos de forma eletrónica via moodle.

Nota: Deve ser possível aferir cada um dos objetivos de aprendizagem no material que entregar.