

Licenciatura em Engenharia Informática e de Computadores

Área Departamental de Engenharia de Electrónica e Telecomunicações e de Computadores

1º Teste, 30 de Janeiro de 2020

Sistemas de Informação II

Duração: 2h30m

Justifique devidamente cada resposta.

1. Considere o modelo físico criado pelo Código 1.

```
create table t_Conta(
    keycol int not null primary key,
    tipo char not null
        check (tipo in ('S','C'))
)
create table t_titular(
    num_bi int not null primary key,
    nome varchar(30) not null
)
create table t_contaSolidaria(
    keycol int not null primary key
        references t_Conta,
    num_bil int not null references t_titular,
    num_biz int not null references t_titular
)
```

```
    num_bi int not null references t_titular
)

create table t_contaSolidaria(
    keycol int not null primary key
        references t_Conta,
    num_bil int not null references t_titular,
    num_biz int not null references t_titular
)
```

Código 1: Ilustração do modelo de dados de suporte ao grupo 1

- (a) [2] Crie em T-SQL o procedimento armazenado `InsereConta(keycol, num_bil, num_biz)` que insere uma nova conta no sistema. Se `num_biz = NULL`, a conta é singular (`tipo = 'S'`); caso contrário é solidária. Admita que o modelo de dados implementa uma generalização total e exclusiva. Deve garantir que se verificam as condições `num_bil ≠ num_biz` e `num_bil ≠ NULL`, necessárias a uma inserção com sucesso. Se não forem garantidas as condições, deve gerar um erro específico para cada situação. Garanta sempre a consistência da base de dados, o nível de isolamento adequado, e a propagação de eventuais erros para quem evoca o procedimento.
- (b) [1] Apresente o código em T-SQL para criar a vista `v_contas`, com o esquema `v_contas(keycol, tipo, titular1, titular2)`, onde `titular1` e `titular2` são as chaves de Titular. Note que `titular2` pode ser `NULL`.
- (c) [2] Apresente o código em T-SQL para permitir inserções sobre a vista `v_contas`. Garanta que sempre que as colunas `tipo` e `titular2` forem inconsistentes (ver Código 1 e Questão 1a) é gerado um erro. Explique sucintamente a solução implementada.
- (d) [1] Considera que se pretende desenvolver uma *Data Access Layer* (DAL) para o modelo criado pelo Código 1. Apresente a definição da interface `IContaSingular` que contém propriedades para aceder a toda a informação de uma conta, e disponibiliza a propriedade de navegação `TitularPrincipal` do tipo `Titular`.
- (e) [2] Defina um tipo C# que implemente `IContaSingular`, garantindo um carregamento lazy da propriedade de navegação `TitularPrincipal`. Assuma a existência do *Data Mapper* `TitularMapper` com o método `Read(int bi)`. Considere que o tipo a implementar, quer os *Data Mappers* recebem como parâmetro na criação a *connection string* a usar em todos os acessos à base de dados, guardada na propriedade `ConnectionString`.

Não assuma a existência de um contexto.

- (f) [2] Implemente o método `Read(int id)` do Data Mapper `ContaSingularMapper` que usa o tipo definido na alínea 1e. Implemente, em ADO.NET, todo o código de acesso e processamento dos dados de uma conta obtida da base de dados.
- (g) [1] Comente a afirmação: "Na *Entity Framework*, a implementação do padrão *Unit of Work*, para funcionar correctamente, necessita de determinar a identidade de cada objecto.".

2. Considere os seguintes Códigos em T-SQL:

```
create table ZT
(
    zt_id int primary key,
    data nvarchar(10) null
);

insert into ZT(st_id,data)
values(-1,'a'),(-2,'b'),(3,'c');
```

Código 2: Tabela ZT

```
delete from ZT where zt_id <= 0 -- I1
update ZT set data = 'c' where data = 'b' -- I2
select zt_id from ZT where zt_id > 0 -- I3
rollback -- I4
commit -- I5

-- I6
set tran isolation level ?
begin tran
```

Código 3: Instruções SQL

De acordo com as listagens ilustradas nos Códigos 2 e 3:

- a) [1.5] Considere dois processamentos transacionais T1 e T2 e um hipotético escalonamento $\langle T1.I6, T2.I6, T1.I1, T2.I2, T1.I5, T2.I4 \rangle$. Indique o resultado da execução do escalonamento considerando duas situações: (i) com READ UNCOMMITTED para ambos os processamentos; (ii) com READ COMMITTED em ambos os processamentos. Justifique.
- b) [1.5] Considere dois processamentos transacionais T2 e T3 e um hipotético escalonamento $\langle T2.I6, T3.I6, T3.I3, T2.I2, T2.I4, T3.I5 \rangle$. Indique o resultado da execução do escalonamento considerando duas situações: (i) com READ COMMITTED para ambos os processamentos; (ii) com REPEATABLE READ em ambos os processamentos. Justifique.
- c) [1] Indique, justificando, se é necessário modificar o escalonamento da Alínea 2b para não exibir a propriedade ~~•~~cascadeless. Em caso afirmativo, apresente o novo escalonamento.
- d) [1.5] Considerando os processamentos transacionais T1, T2 e T3, usados anteriormente, apresente, justificando, um escalonamento que entra em *deadlock* quando colocado em execução. Se não for possível encontrar um escalonamento com essa característica, diga porquê.
- e) [1.5] Crie um processamento transaccional T4, que colocado em execução concorrente com T3, tenha uma anomalia de NON-REPEATABLE READ. Apresente: (i) o escalonamento, (ii) os níveis de isolamento de T3 e T4, (iii) a transacção que padece da anomalia.
- f) [1] Qual o papel do sistema de log de um SGDB?
- g) [1] Compare os objetivos da técnica de processamento transaccional *Saga* com a de *mini batch*.