

Mestrado em Engenharia Informática

Sistemas de Gestão de Bases de Dados (2020/21)

Trabalho Prático
Versão 1.0 (2020-10-05)

Controlo de Transações

1. Objetivos e competências

Objetivos: conhecer os meandros do controlo de transações, nomeadamente, nível de isolamento, tipos de trincos (*Locks*) e Impasse (*Deadlock*).

Competências: com a concretização deste trabalho os alunos compreendem e são capazes de explicar o que está a acontecer com os seus dados e com as transações que os manipulam.

2. Contexto

No desenvolvimento de sistemas de informação, é frequente lidar com cenários onde várias aplicações acedem a dados partilhados. O Sistema de Gestão de Bases de Dados (SGBD) é a entidade responsável pela gestão dos acessos concorrentes.

Suponha que a base de dados contém três tabelas: *Encomenda*, *EncLinha* e *LogOperations*. As tabelas *Encomenda* e *EncLinha* representam as encomendas e as suas linhas, respetivamente. A tabela *LogOperations* representa o histórico das alterações às tabelas *Encomenda* e *EncLinha*.

Para os objetivos do trabalho não é relevante o nível de normalização das relações.

Encomenda (EncId, *ClienteId*, *Nome*, *Morada*)

EncLinha(EncId, ProdutoId, *Designacao*, *Preco*, *Qtd*)

LogOperations(NumReg, *EventType*, *Objecto*, *Valor*, *Referencia*, *UserId*, *TerminalId*, *TerminalName*, *DCriacao*)

A base de dados e as tabelas são criadas a partir dos *scripts* (*CreateTables.sql*) fornecidos pelo docente. A estrutura das tabelas não deve ser alterada. A tabela *LogOperations* vai sendo alterada por ação de *triggers* (*Triggers.sql*).

Notas sobre a operação da base de dados:

1. Inserir uma encomenda com dois produtos.

```
Begin Tran
Insert Into Encomenda(EncId, ClienteId, Nome, Morada)
Values (1, 10, 'António', 'Aldeia do Monte')

Insert Into EncLinha(EncId, ProdutoId, Designacao, Preco, Qtd)
Values (1, 150, 'Batata', 200, 10)

Insert Into EncLinha(EncId, ProdutoId, Designacao, Preco, Qtd)
Values (1, 200, 'Feijão', 300, 2)
Commit
```

Esta transação insere na tabela *LogOperations*, pela ação dos *triggers*, as seguintes linhas:

NumReg	Event Type	Objecto	Valor	Referencia	UserID	TerminalID	TerminalName	DCriacao
15	I	Encomenda	1	NULL	dbo	14124	ASUS116	2019-10-01 10:27:31.437
16	I	EncLinha	1	150	dbo	14124	ASUS116	2019-10-01 10:27:31.443
17	I	EncLinha	1	200	dbo	14124	ASUS116	2019-10-01 10:27:31.443

Ou seja, a tabela *LogOperations*, indica que no dia 2019-10-01, o utilizador *dbo*, que estava no terminal 14124/Asus116, inseriu (atributo *EventType = I*) a encomenda (atributo *Objecto*) com o *EncId = 1* (atributo *Valor*).

2. Alterar uma linha de uma encomenda.

```
Update EncLinha
Set Qtd = 1
Where EncId = 1 and ProdutoId = 200
```

Efeitos sobre a tabela *LogOperations*:

NumReg	Event Type	Objecto	Valor	Referencia	UserID	TerminalID	TerminalName	DCriacao
18	U	EncLinha	1	200	dbo	14124	ASUS116	2019-10-01 10:35:41.320

Neste caso, o registo de *Log* indica que foi alterado um objecto do tipo *EncLinha*, sendo o *EncID* indicado pelo atributo *Valor* e o *ProdutoId* indicado pelo atributo *Referencia*.

3. Alterar o cabeçalho de uma encomenda.

```
Update Encomenda
Set Nome = 'António Miranda'
Where EncId = 1
```

Efeitos sobre a tabela *LogOperations*:

NumReg	Event Type	Objecto	Valor	Referencia	UserID	TerminalID	TerminalName	DCriacao
19	U	Encomenda	1	NULL	dbo	14124	ASUS116	2019-10-01 10:44:18.010

4. Apagar uma encomenda e suas linhas.

```
Begin Tran
Delete from EncLinha Where EncID = 1
Delete from Encomenda Where EncId = 1
Commit
```

Efeitos sobre a tabela *LogOperations*:

NumReg	Event Type	Objecto	Valor	Referencia	UserID	TerminalID	TerminalName	DCriacao
22	D	Encomenda	1 António Miranda Aldeia do Monte	NULL	dbo	14124	ASUS116	2019-10-01 10:51:49.763
21	D	EncLinha	1 150	Batata 200.00 10.00	dbo	14124	ASUS116	2019-10-01 10:51:49.763
20	D	EncLinha	1 200	Feijão 300.00 1.00	dbo	14124	ASUS116	2019-10-01 10:51:49.763

Primeiro são eliminadas as linhas da tabela *EncLinha*, dando origem a dois registos na tabela *LogOperation* (*NumReg* 20 e 21), e depois é eliminado o registo da tabela *Encomenda* (*NumReg* = 22).

3. Criação da Base de dados

Passos a realizar no SSMS (*SQL Server Management Studio*):

1. Criar a base de dados *MEI_TRAB* numa pasta à escolha.

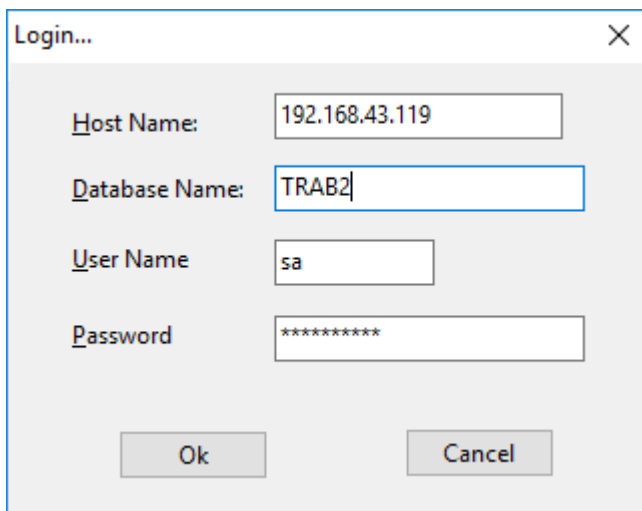
Antes de executar os *scripts* seguintes deve assegurar-se que a base de dados seleccionada é a correcta (*MEI_TRAB*).

2. Criar as tabelas (*CreateTables.sql*).
3. Criar os *triggers* (*Triggers.sql*).
4. Criar os procedimentos armazenados (*Procs.sql*)

4. Aplicações

O trabalho prático assenta sobre quatro aplicações sobre *SQL Server*.

As aplicações, ao iniciar, devem conectar-se ao servidor de base de dados. Para o efeito devem solicitar o endereço IP (ou o *Hostname*) e base de dados (*DatabaseName*), o *UserName* e a *Password*.



Após estabelecer ligação ao servidor, o utilizador deve escolher o nível de isolamento pretendido para as suas transações.

4.1 Aplicação *Edit*

Nesta aplicação, o utilizador indica o ID da encomenda a editar/alterar. O programa mostra os dados da encomenda, incluindo as linhas, e permite ao utilizador alterá-los.

Assuma que o utilizador não quer inserir novos produtos na encomenda nem eliminar produtos já encomendados. Ou seja, o utilizador “só” quer poder alterar a morada do cliente e/ou a quantidade de cada produto.

Modo de funcionamento:

Depois de obter o ID da encomenda, e antes de ler os dados da mesma, a aplicação deve escrever na tabela *LogOperations* um registo semelhante ao seguinte:

```
Insert Into LogOperations (EventType, Objecto, Valor, Referencia)
Values ('O', User_EncId, User_DateTime, User_Reference)
```

Onde *User_EncId* é o *Id* introduzido pelo utilizador, *User_DateTime* é o instante corrente (*datetime*) e *User_Reference* deve ser um identificador produzido pela aplicação e que só faça sentido no contexto desta, por exemplo “*G1-20191001101356321*”, para representar uma operação do grupo 1, no dia 2019-10-01 às 10:13:56:321.

Após o utilizador alterar os dados (morada e/ou qtds), a encomenda é atualizada na base de dados. No final a aplicação deve escrever um registo semelhante ao seguinte:

```
Insert Into LogOperations (EventType, Objecto, Valor, Referencia)
Values ('O', User_EncId, User_DateTime, User_Reference)
```

onde *EncId* é o *ID* da encomenda que esteve a ser editada, *User_Reference* é o valor usado no primeiro *Insert* (ex. *G1-20191001101356321*) e *User_DateTime* é o (novo) instante de tempo (*datetime*).

4.2 Aplicação *Browser*

Num sistema de informação os utilizadores tomam decisões com base nos dados que possuem. Dados desatualizados podem conduzir a decisões erradas.

Com o módulo *Browser* pretende-se visualizar as encomendas e as suas linhas. A janela desta aplicação deve conter duas grelhas, uma com as encomendas e a outra com os produtos (linhas) da encomenda selecionada.

A janela deve conter um botão para fazer o *refresh* dos dados. Deve também ser estudada a possibilidade de usar um *timer* para fazer o refrescamento automático. O estudo deve incidir sobre o intervalo (taxa) de refrescamento.

4.3 Aplicação “Log Tempo”

Esta aplicação mostra os registos do tipo *EventType* = ‘O’ (“Outro”) numa grelha. Cada linha da grelha deve conter as seguintes colunas *UserId*, *EncId*, *Tempo*, onde *Tempo* corresponde ao tempo de edição da encomenda *EncId* pelo utilizador *UserId*. Deve ser incluído um *timer* para o *refresh*.

```
SELECT L01.UserId, L01.Objecto as EncId, DATEDIFF(SS,L01.Valor, L02.Valor) as Tempo
FROM LogOperations L01, LogOperations L02
WHERE L01.Referencia = L02.Referencia and L01.DCriacao < L02.DCriacao and
      L01.Referencia = 'G1-20191001101356321'
```

4.4 Aplicação “Log”

A janela principal desta aplicação deve conter uma grelha onde sejam visualizados as “N” linhas mais recentes da tabela *LogOperations*. Considerar só eventos do tipo “I”, “U”, “D”. Deve ser incluído um *timer* para o *refresh*.

5. Testes

Depois de criada a base de dados e os seus elementos (tabelas, *triggers* e procedimentos armazenados), e construídas as aplicações, pretende-se, agora, trabalhar com transações em ambiente concorrente e testar os níveis de isolamento das transações e fenómenos associados à execução concorrente.

Os testes envolvem o uso do *SSMS*, para executar o *script DoWork.sql*, em simultâneo com as aplicações. De preferência a executar o *SSMS* e as aplicações em computadores diferentes.

Em modo concorrente, executar alguns testes do género:

1) *SMSS*:

- a) Estabelecer o nível de isolamento (*SET TRANSACTION ISOLATION LEVEL ...*)
- b) Executar o script *DoWork.sql*

2) Na aplicação *Edit*

- a) Estabelecer o nível de isolamento
- b) Editar uma encomenda
- c) Guardar as alterações ou descartar as alterações

3) Na aplicação *Browser*

- a) Estabelecer o nível de isolamento
- b) Visualizar os registos
- c) Fazer o *refresh* dos dados.

6. Relatório

Deve ser elaborado um relatório detalhado abordando, pelo menos, os seguintes tópicos:

- objetivos gerais;
- descrição do trabalho prático;
- objetivos para cada aplicação (*Edit*, *Browser*,...);
- transacções (gestão, controlo, níveis de isolamento e recuperação);
- Tipo de trincos (*locks*);
- testes e análise de resultados (sugestão ao nível de isolamento):
 - *Edit*:
 - editar encomenda não confirmada (*uncommitted*);
 - dois ou mais utilizadores a editar a mesma encomenda.
 - *Browser*: considerar diversas taxas de *refresh*.
- Incluir uma secção de conclusões. Indicar o que foi conseguido e o que não foi conseguido (e as suas razões);
- Sugestões de melhoria (para o próximo ano letivo).