Bases de Dados

1º semestre

Lab 10 : Transacções

1ª Parte

1. Abra uma linha de comando para o sistema MySQL e use o script ***bank.sql*** para recriar a base de dados "bank".

2. Escreva uma consulta para obter os dados das contas do cliente *Smith*.

3. O cliente *Smith* pretende transferir 500€ a conta A-444 para a conta A-215. Inicie uma transacção com o comando `START TRANSACTION`;

4. Coloque 500€ na conta A-215.

5. Consulte os saldos das contas do cliente *Smith*. Neste momento quanto dinheiro tem o cliente *Smith* no banco?

6. Mantendo a transacção em aberto, abra uma segunda linha de comando do sistema e ligue-se à mesma base de dados.

7. Consulte os saldos das contas do cliente *Smith*. Afinal quanto dinheiro tem o cliente *Smith* no banco?

8. Na primeira linha de comando, onde está a correr a transacção, retire 500€ da conta A-444.

9. Na segunda linha de comando consulte os saldos das contas do cliente *Smith*.

10. Na primeira linha de comando confirme a transacção.

11. Na segunda linha de comando consulte novamente os saldos das contas do cliente *Smith*. Confirme os resultados da transacção.

2ª Parte

12. O cliente *Smith* acabou de abastecer 25€ de gasolina numa estação de serviço e vai pagar com multibanco (conta A-444). Ao mesmo tempo a sua esposa, que tem um cartão multibanco para a mesma conta, vai levantar 50€ numa máquina multibanco do outro lado da cidade.

13. Na primeira linha de comando, inicie uma transacção.

14. Na segunda linha de comando, inicie uma transacção.

15. Na primeira transacção retire 25€ da conta (abastecimento).

16. Na segunda transacção retire 50€ da conta (levantamento). O que se passa quando tenta fazer isto? Porquê?

17. A linha telefónica da estação de serviço está intermitente e a ligação cai. O pagamento que estava a ser feito tem de ser cancelado. Cancele essa transacção.

18. Ao mesmo tempo que faz a alínea 17, repare no que acontece na segunda linha de comando. Como explica este fenómeno?

19. Confirme a transacção de levantamento.

20. Consulte os saldos das contas do cliente *Smith*.

3ª Parte

21. No caso que se segue, utilize apenas uma linha de comando.

22. Inicie uma transacção.

23. Escreva um comando para apagar da tabela *depositor* todos os registos que mencionam o cliente *Smith*.

24. Agora que o cliente *Smith* deixou de ser titular das suas contas, é possível que essas contas não tenham titular. Escreva uma consulta para determinar quais são as contas na tabela *account* que não estão associadas a clientes na tabela *depositor*.

25. Escreva um comando para apagar da tabela *account* as contas que se encontram nessa situação (i.e. sem titular na tabela *depositor*).

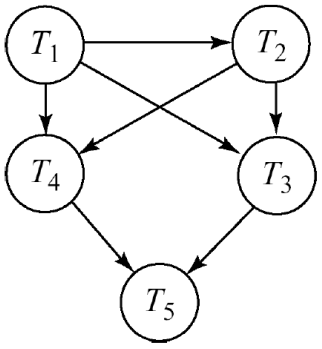
26. Confirme que as contas A-215 e A-444, que pertenciam ao cliente *Smith*, foram apagadas.

27. Cancele a transacção.

28. Verifique que as contas do cliente *Smith* foram repostas, tanto na tabela *account* como na tabela *depositor*.

4ª Parte

1. Considere o seguinte grafo de precedências, que representa os conflitos existentes num escalonamento que envolve cinco transacções concorrentes. O escalonamento é serializável? Justifique.



2. Considere as seguintes transacções:

T1	T2
read(A)	read(B)
read(B)	read(A)
if A = 0 then B := B + 1	if B = 0 then A := A + 1
write(B)	write(A)

Inicialmente a base de dados tem *A = 0* e *B = 0*. Para que a base de dados se mantenha consistente é necessário que *A = 0* ou *B = 0*.

(a) Mostre que os escalonamentos em série destas duas transacções mantêm a consistência da base de dados.
(b) Apresente um exemplo de um escalonamento concorrente de T1 e T2 que não seja serializável. Explique porquê.
(c) Existe algum escalonamento concorrente de T1 e T2 que seja serializável? Justifique.
(d) Adicione instruções de **lock-S()**, **lock-X()** e **unlock()** às duas transacções de modo a que obedeçam ao protocolo em duas fases.
(e) Com o resultado da alínea (d) dê um exemplo de um escalonamento concorrente que resulte em *deadlock*.