Sistemas de Bases de Dados

2.º teste (com consulta limitada: 2 folhas identificadas) - Duração: 2 horas

N.º:		Nome:	

Grupo 1 (7 valores)

1 a) Quais são as 4 propriedades ACID que um sistema de bases de dados deve idealmente garantir relativamente a uma transação?

1 b) Desenhe o diagrama de estados genérico para uma transação (com todos os possíveis estados e transições entre os mesmos).

1 c) Para cada um dos 6 escalonamentos abaixo, indique (com 'X') se é ou não serializável de conflito. Em caso afirmativo, indique também uma serialização que lhe seja equivalente (de conflito).

Nota: Nesta alínea cada opção certa vale 0,4 valores, e cada opção errada vale –0,4 valores (negativo, portanto). Uma resposta afirmativa correta mas com serialização incorreta será cotada a 50% (i.e. 0,2 valores positivos).

	Não serializável	**Serializável**	**Serialização**
S1:	☐	☐	_____
S2:	☐	☐	_____
S3:	☐	☐	_____
S4:	☐	☐	_____
S5:	☐	☐	_____
S6:	☐	☐	_____

S1

T1	T2
Read(Y)	
	Read(X)
	Read(Z)
Write(Y)	
	Write(Z)
	Write(X)
Read(X)	
Write(X)	

S2

T1	T2
Read(Y)	
	Read(X)
	Read(Z)
Write(Y)	
Read(X)	
Write(X)	
	Write(Z)
	Write(X)

S3

T1	T2
Write(X)	
	Write(Y)
Read(X)	
Read(Y)	
	Read(X)
	Read(Y)
Read(Z)	
	Read(Z)

S4

T1	T2	T3
Read(Y)		
Read(X)		
	Read(Y)	
Write(Y)		
	Write(Z)	
Write(X)		
		Write(X)

S5

T1	T2	T3	T4
		Write(W)	
	Read(Y)		
		Read(X)	
Write(Y)			
			Read(X)
			Read(Y)
		Write(X)	
	Write(X)		
			Write(Y)

S6

T1	T2
select * from s;	
	select * from t where a=0;
	insert into s(a,b) values (0,0);
update t set b=0 where a=1;	
select * from t;	
	select count(*) from s;

N.º:	

1 d) Considere a tabela ***alunos****(id, nota)* já com os tuplos (1,18) e (2,13), previamente criada com:

```
create table alunos(id int primary key,
                    nota int check(nota between 0 and 20));
```

Considerando execução das transações em modo *Read Uncommitted* (i.e., nível mais fraco de consistência, maior concorrência), apresente um escalonamento de transações com operações apenas sobre esta tabela, para cada caso:

I. Com *dirty read*

II. Com *unrepeatable read*

III. Com *phantom read*

IV. Escalonamento irrecuperável. Justifique também porque é que não é recuperável.

Grupo 2 (5 valores)

2 a) O protocolo de *locking* de 2 fases garante escalonamentos livres de *deadlock*? Se sim, justifique. Se não, dê um exemplo.

2 b) Considere agora o protocolo baseado em *timestamps* para o seguinte escalonamento das transações T1, T2, T3, e T4, com *timestamps* 1, 2, 3, e 4, respetivamente:

T1	T2	T3	T4
	Read(X)		
Read(Y)			
			Write(X)
		Read(Z)	
		Write(X)	
	Write(W)		
			Write(X)
	Write(Z)		
Read(X)			
			Read(W)

Com este protocolo, haverá transações a abortar? Se sim, quais, onde, e devido a quê?

N.º:	

2 c) Voltando à tabela ***alunos****(id, nota)* da questão 1.d (com tuplos <1,18> e <2,13>), considere as seguintes transações executando em *Snapshot Isolation* :

T1	T2
	Insert into alunos values (3,10);
	Update alunos set nota=nota+1 where mod(nota,2) = 1;
Select * from alunos;	
Update alunos set nota=nota-1 where mod(id, 2) = 1;	
Commit;	
	Commit;

I. Alguma transação abortará ou bloqueará? Se sim, qual e onde/quando?

II. O que verá o utilizador como resultado da instrução '*select * from alunos*' dentro de T1?

III. Qual o conteúdo da tabela *alunos* após a conclusão de todo o escalonamento?

2 d) Indique e descreva, por ordem, as 3 fases do protocolo baseado em validação para uma transação *T*.

Grupo 3 (8 valores)

3 a) Uma transação pode ficar *committed*, com escritas (*writes*) suas no buffer de memória ainda não refletidas em disco/armazenamento estável? Porquê?

3 b) Considere o modelo simplificado de *log* dado nas aulas, e o seguinte escalonamento.

T1	**T2**
X ← 5	
	Y ← 9
	Commit
Z ← 2*X	

I. Escreva a sequência de registos de *log* referentes a este escalonamento, sabendo que inicialmente X, Y, e Z tinham o valor 0.

II. Apresente agora a sequência adicional de registos fruto do abortar da transação T1.

N.º:	

3 c) Complete os (3 casos de) registos de *log* abaixo, após uma falha do sistema. I.e. acrescente os registos associados à sua recuperação.

I	**II.**	**III.**
<T0 start> <T0, A, 0,1> <T1 start> <T1, B, 2,4> <T0, A, 1,2> <T1 commit> <T0, C, 2,7>	<T5 start> <T5, B, 3,4> <checkpoint {T3,T5}> <T3, D, 2,1> <T6 start> <T6, F, 9,4> <T3, D,2> <T3 abort> <T5, D, 2,5>	<T2 start> <T2, O1, operation-begin> <T2, B, 2,5> <T2, O1, operation-end, (B,-3)> <T2, A, 3,4> <T2, O2, operation-begin> <T2, B, 5,6>

3 d) Para uma tabela ***clientes****(id, nome, cidade, saldo)*, dê um exemplo (baseando-se nos seus atributos) de fragmentação horizontal, e escreva uma consulta SQL que, nesse caso, poderia beneficiar de processamento paralelo.

3 e) O que pode correr mal na abordagem '*read one, write all available*' para controlo de concorrência em sistemas distribuídos?

2.º teste de SBD

- duração: 2 horas
- pode sair a partir das 18:00 (não antes)
- pode responder a lápis
- pode ter consigo 2 folhas agrafadas de consulta, identificadas com nome e número
- tenha o seu documento de identificação (e.g. cartão de estudante) visível ao seu lado
- não são permitidos equipamentos eletrónicos
- não pergunte pelas horas, por favor