IPS Instituto Politécnico de Setúbal
Escola Superior de Tecnologia de Setúbal

Complementos de Bases de Dados

2017/18

Avaliação Continua
2º Teste

- Versão A -

04 de Janeiro de 2018

Notas preliminares:

- Prova sem consulta.
- O teste é constituído por duas partes: I – Escolha múltipla; II – Resposta aberta.
- A prova deverá ser resolvida no próprio enunciado, em **particular relativamente à Parte I, preenchendo a tabela existente no final desta página.**
- Não serão consideradas respostas que não constem na tabela referida no ponto anterior.
- Poderá utilizar uma folha de rascunho desde que informe o docente e a entregue no final.
- Não é permitida a utilização de telefones ou de quaisquer outros equipamentos, com exceção de máquinas de calcular (não gráficas ou homologadas).
- Em caso de desistência só deverá abandonar a sala ao fim de 20 min. do inicio da prova, e deverá entregar o enunciado da prova identificada e inscrevendo na mesma "Desisto".
- A cotação das questões está entre [ ].
- Na parte I do teste, a cada resposta errada é atribuída 0 valores e adicionalmente descontado ¼ do valor da sua cotação.
- Leia atentamente cada questão antes de responder.

Identifique o rodapé das páginas ímpares do enunciado com o seu número e nome.

Enunciado

Parte I (10 valores)

Grelha de Respostas:

1	2	3	4	5	6	7	8	9	10
(1v)	(1v)	(1v)	(1v)	(1v)	(1v)	(1v)	(1v)	(1v)	(1v)

1. Considerando realizado um backup integral, a operação de backup diferencial, pode ser realizada após:

a. Um backup diferencial
b. Um backup de transacções
c. Qualquer das duas operações anteriores
d. Nenhuma das duas operações

2. Sejam os seguintes acontecimentos:
backup integral_1 + backup logs + backup diferencial + backup integral_2 + crash da BD.
A recuperação da BD, com vista a minimizar a perca de dados, deverá ter a seguinte sequência:

a. Repor Backup Integral_1 + repor Backup Logs + repor Backup Diferencial + Repor Backup Integral_2
b. Repor Backup Logs + repor Backup Diferencial + repor Backup Integral_2
c. Backup Log + repor Backup Integral_2 + repor Backup Log
d. Backup Log + repor Backup Diferencial + repor Backup Integral_2 + repor Backup Logs

3. Relativamente a índices *non-clustered*:

a. Podem ser criados vários por tabela para otimizar consultas
b. Apenas pode ser criado um por tabela
c. É o índice primário da tabela
d. Só pode ser criado para a chave primária quando esta é composta

4. Considere um índice Hash dinâmico e a ilustração do sufixo da chave de *hash* "H" na Figura 1.
A inserção do valor 21 (10101) da chave é colocado no bucket:

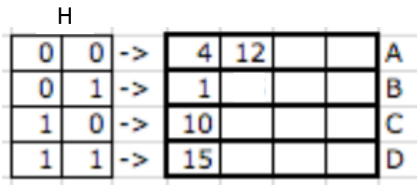

Figura 1

a. B
b. C
c. Necessita do aumento da profundidade da chave e coloca em C
d. Necessita do aumento da profundidade da chave e coloca num novo bucket

5. É característica de uma função *hash*:

a. Atomicidade
b. Aleatoriedade
c. Assimetria
d. Todas as anteriores

6. O Índice X da figura 2 é:

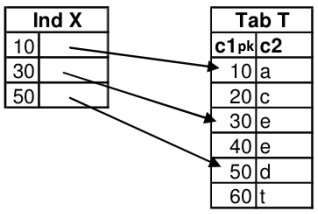

Figura 2

a. Secundário
b. Denso
c. Esparso
d. Nenhuma das anteriores

7. Relativamente aos *Fixed* e *Custom Server Roles* do MS SQL Server
a. Não podem ser redefinidas as permissões associadas aos *Fixed Server Roles*
b. Em caso de conflito de permissões entre *roles* é concedido o acesso ao(s) *securable(s)* em causa
c. Um *user* não pode ser membro simultaneamente de um *Fixed* e de um *Custom Server Role*
d. Todas as anteriores

8. Considere o código da Figura 2 (em baixo)
a. O utilizador TestUser pode realizar operações de INSERT na TestTable2
b. O utilizador TestUser não pode realizar operações de SELECT na TestTable2
c. Todos os membros de TestRole podem realizar operações de INSERT na TestTable1
d. Nenhuma das anteriores

```
Create database TestDB
GO
USE TestDB
Create Role TestRole
GO
Create User TestUser
Without Login
GO
EXEC sp_addrolemember
@rolename='TestRole' ,
@membername='TestUser'
GO
```

```
Create Schema TestSchema
GO
Create Table TestSchema.TestTable1 (id int);
Create Table TestSchema.TestTable2 (id int)
GO

GRANT SELECT on Schema::TestSchema To TestRole
GRANT INSERT on Schema::TestSchema To TestRole

DENY  INSERT on object::TestSchema.TestTable2 To
TestUser
GO
```

Figura 3

9. O nível de isolamento *Read Uncommitted*, evita as seguintes ocorrências:
a. Non-repeatable read e phantom read
b. Non-repeatable read, phantom read e Dirty read
c. Phantom read e dirty read
d. Nenhuma das anteriores

10. Se a opção principal for por um melhor desempenho das operações sobre a base de dados, o modelo de recuperação a selecionar deve ser:

a. Full Recovery
b. Simple Recovery
c. Bulk_Logged Recovery
d. Fast Recovery

Parte II [10 valores]

II.1 [2 val]

No contexto de transações concorrentes distinga os efeitos conhecidos por *phantom-read* e *dirty-read*.

II.2 [2 val]
Explique as implicações do nível de isolamento de transações "*Serializable"*?

II.3 [2 val]

No âmbito da encriptação de dados discuta as vantagens e desvantagens da utilização de chaves simétricas ou assimétricas e como podem ser utilizadas conjuntamente.

II.4 [2 val]

No contexto dos índices explique as diferenças entre *Hash Estático* e *Hash Dinâmico* e a sua realção com o *bucket overflow*.

II.5 [2 val]

Ordene indicando a sequência através das respetivas letras identificadoras os enxertos de código da tabela em baixo de modo a que se possa executar os comandos:

```
SELECT Encrypt(myColumn) FROM myTable

SELECT Decrypt(myColumn) FROM myTable
```

<table>
<tr><td rowspan="2">CREATE SYMMETRIC KEY MySymmetricKeyName
WITH
ALGORITHM = AES_256,
KEY_SOURCE = 'a very secure strong password or phrase'
ENCRYPTION BY CERTIFICATE ACertificate;</td><td>CREATE CERTIFICATE ACertificate
ENCRYPTION BY PASSWORD = ‘certfpass’
WITH SUBJECT = ’protect data’
EXPIRY_DATE = ‘20180131’</td></tr>
<tr><td>B</td></tr>
<tr><td>A</td><td>EXEC OpenKeys

C</td></tr>
<tr><td>CREATE FUNCTION Encrypt (@ValueToEncrypt varchar(max))
RETURNS varbinary(256)
AS
BEGIN
DECLARE @Result varbinary(256)

SET @Result = EncryptByKey(Key_GUID('MySymmetricKeyName'), @ValueToEncrypt)
RETURN @Result
END</td><td>CREATE PROCEDURE OpenKeys
AS
BEGIN
SET NOCCOUNT ON;

BEGIN TRY
OPEN SYMMETRIC KEY MySymmetricKeyName
DECRYPTION BY CERTIFICATE ACertificate
END TRY
BEGIN CATCH
-- Handle non-existent key here
END CATCH
END</td></tr>
<tr><td>D</td><td>E</td></tr>
<tr><td>CREATE FUNCTION Decrypt (@ValueToDecrypt varbinary(256))
RETURNS varchar(max)
AS
BEGIN
DECLARE @Result varchar(max)

SET @Result = DecryptByKey(@ValueToDecrypt)
RETURN @Result
END</td><td>Create a Database Master Key CREATE MASTER KEY ENCRYPTION BY PASSWORD = ‘myStrongPassword’</td></tr>
<tr><td>F</td><td>G</td></tr>
</table>

Sequência de execução proposta:

____ ____ ____ ____ ____ ____ ____

[Fim do Enunciado]