# e-Fólio A

**U.C. 21103**

**Sistemas de Gestão de Bases de Dados**

**2019-2020**

## Resolução e Critérios de Correção

### INSTRUÇÕES

- O e-fólio é constituído por 6 alíneas com cotação de 0,5 valores cada. A cotação global é de 3 valores.
- O e-fólio deve ser entregue num único ficheiro PDF, não zipado, com fundo branco, com perguntas numeradas e sem necessidade de rodar o texto para o ler. Penalização de 1 a 3 valores.
- Não são aceites e-fólios manuscritos, i.e. tem penalização de 100%.
- O nome do ficheiro deve seguir a normal "eFolioA" + <nº estudante> + <nome estudante com o máximo de 3 palavras>
- Durante a realização do e-fólio, os estudantes devem concentrar-se na resolução do seu trabalho individual, não sendo permitida a colocação de perguntas ao professor ou entre colegas.
- A interpretação das perguntas também faz parte da sua resolução, se encontrar alguma ambiguidade deve indicar claramente como foi resolvida.
- A legibilidade, a objetividade e a clareza nas respostas serão valorizadas, pelo que, a falta destas qualidades será penalizada.

Vetor Cotações
1 2 3, 4 5 6 pergunta
5 5 5, 5 5 5 décimas

Critérios de correção gerais: todas as respostas devem ser justificadas, incluir imagens e exemplos com vista a clarificar os argumentos expostos.

1) Relativamente ao Cap.10 – Storage and File Structure
Considere o efeito de atomicidade dos blocos num disco RAID nível 5 como o da figura seguinte.

| **Disk 1** | **Disk 2** | **Disk 3** | **Disk 4** |
|---|---|---|---|
| A1 | A2 | A3 | P4 |
| B1 | B2 | P3 | B4 |
| C1 | P2 | C3 | C4 |
| P1 | D2 | D3 | D4 |

Ocorreu um erro no disco 3. Qual o algoritmo para a recuperação da informação do disco 3? Qual a informação recuperada? Justifique a resposta e exemplifique.

| Disk 1 | Disk 2 | Disk 3 | Disk 4 |
|---|---|---|---|
| 1011 1011 | 1010 0011 | | 1000 1000 |
| 0110 0001 | 1100 0010 | | 1001 0011 |
| 1101 0111 | 1110 1110 | | 0111 0000 |
| 1000 0101 | 1110 0011 | | 0010 1010 |

**Resposta:**
A RAID5 utiliza um algoritmo para recuperação de dados que é baseado em funções XOR entre a informação dos restantes discos para obter a informação perdida.

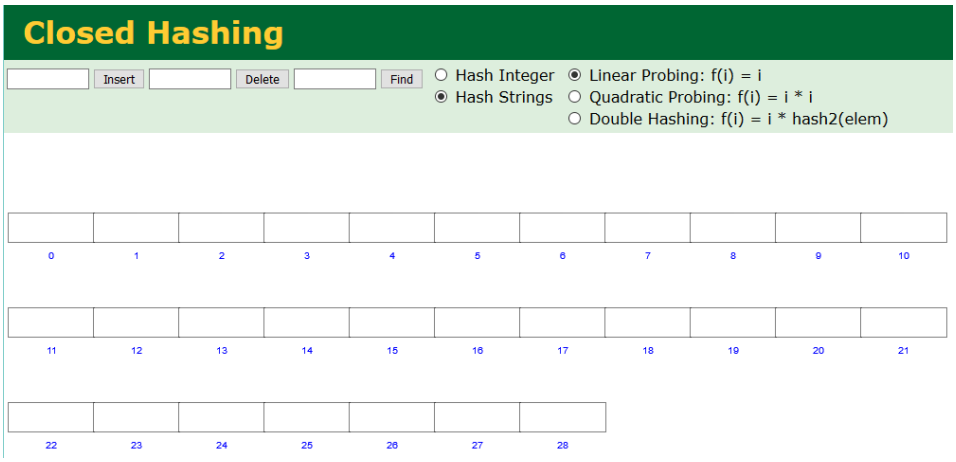
Para recuperar os dados do Disco 3 temos de fazer as seguintes operações:

A3 =
(A1 XOR A2) XOR P4 =
(1011 1011 XOR 1010 0011) XOR 1000 1000 =
0001 1000 XOR 1000 1000 =
1001 0000

| | Disk 1 | Disk 2 | Disk 3 | Disk 4 |
|---|---|---|---|---|
| A | 1011 1011 | 1010 0011 | 1001 0000 | 1000 1000 |
| B | 0110 0001 | 1100 0010 | 0011 0000 | 1001 0011 |
| C | 1101 0111 | 1110 1110 | 0100 1001 | 0111 0000 |
| D | 1000 0101 | 1110 0011 | 0100 1100 | 0010 1010 |

Critério de correção:
- 2 décimas para  o algoritmo recuperação
- 3 décimas para a informação recuperada
- erros, omissões, redundâncias ou indentação desadequada: -20% a -100%

2) Relativamente ao Cap. 11 - Indexing and Hashing




Considere a seguinte estrutura de "Closed Hashing" em https://www.cs.usfca.edu/~galles/visualization/ClosedHash.html. Escreva em pseudo-código o processo de inserção de 'strings'. Justifique a resposta e exemplifique.

**Resposta:**
Em primeiro lugar a string é convertida para um inteiro de 32 bits, utilizando somas, deslocamentos para a esquerda e o operador XOR. Por exemplo str2int ('aa')=25.

Inserting element: aaaa
Hashing: aa

```
00000000000000000110011100010000
XOR          0000
--------------------------------
00000000000000000110011100010000
```

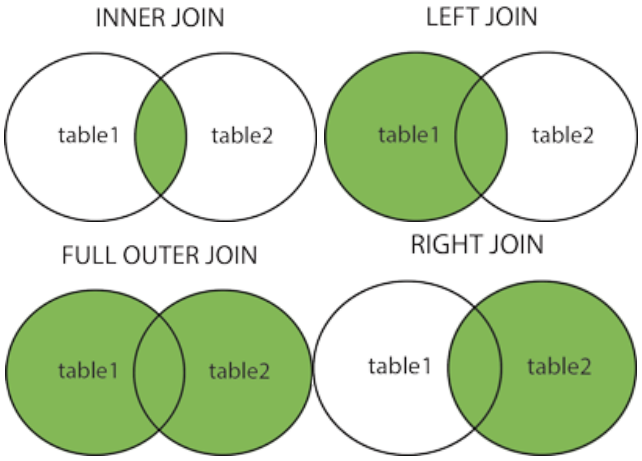
No exemplo Table_size=29 e o pseudo-código é o seguinte:

Pseudo-código: Int Insert_Closed_Hashing (str X)
int loc = str2int (X) % Table_Size;
if (Table[loc]==Empty) then Table[loc]=X, return 1
else old_loc=loc // colisão
repeat
loc=(loc+1)%Table_Size
until Table[loc]==Empty or loc==old_loc
if loc != old_loc then Table[loc]=X, return 1
else print ('Table with no empty cells'), return 0

Critério de correção:
- 1 função 'hash' para a 'string'
- 2 décimas, para inserção
- 2 décimas, para inserção com colisão e verificar a tabela cheia
- erros, omissões, redundâncias ou indentação desadequada: -20% a -100%

3) Relativamente ao Cap. 12 - Query Processing
a) O que entende por Outer Join? Justifique a resposta e exemplifique.
b) Quais as estratégias para implementação do Outer Join? Justifique a resposta e exemplifique.

**Resposta:**
3.a) Existem vários tipos de junção:
- inner join: retorna registros que possuem valores correspondentes nas duas tabelas
- left (outer) join: retorna todos os registros da tabela esquerda e os registros correspondentes da tabela direita
- right (outer) join: retorna todos os registros da tabela direita e os registros correspondentes da tabela esquerda
- full (outer) join: retorna todos os registros quando houver uma correspondência na tabela esquerda ou direita



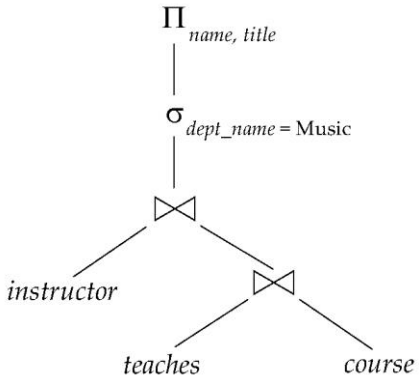

3.b) Podemos implementar o Outer-join usando uma de duas estratégias (pag. 563 manual):
- Para calcular o left-join (r ⟕ s) primeiro calculamos o inner-join (r ⋈ s) e guardamos o resultados em q1. Depois calculamos r - Π(q1) para obter os tuplos que não pertencem ao inner-join. Preenchemos os atributos de s com valores nulos e unimos com q1 para obter o oute-join, i.e., q1∪ r-Π(q1) com q1=(r ⋈ s).
- Modificar os algoritmos de junção. No algoritmo de junção com 'nested-loop' os tuplos à esquerda (ou direita conforme o caso), que não correspondem na junção, são escritos no output depois dos atributos de s serem preenchidos a valores nulos.

Critério de correção:
- 2 décimas, definição
- 3 décimas, estratégias de implementação
- erros, omissões, redundâncias ou indentação desadequada: -20% a -100%

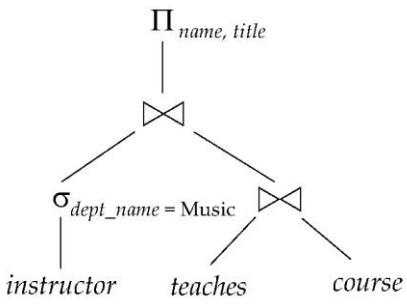
4) Relativamente ao Cap. 13 - Query Optimization
a) O que entende por otimização de consultas? Justifique a resposta e exemplifique.
b) Qual a relação entre a otimização de consultas com as linguagens de programação declarativas? Justifique a resposta e exemplifique.

**Resposta a)** Dada uma qualquer consulta em SQL existem várias expressões equivalentes em álgebra relacional que correspondem a diferentes implementações algorítmicas. Para a seguinte consulta em SQL existem pelo menos duas árvores ou planos de execução com diferentes custos.

```
SELECT name, title
FROM course, teaches, instructor
WHERE dept_name = 'Music'
AND course.attribute1 = teaches.attribute1
AND instructor.attribute2 = teaches.attribute2
```




Plano 1




Plano 2

O Plano 2 parece ter menos custo já que o filtro dept_name = 'Music' é realizado num ramo inferior da árvore.

**Resposta b)** Nas linguagens de programação existem essencialmente as linguagens imperativas e as declarativas.

Nas linguagens imperativas, como o C ou Java, o programador define a sequência das ações.

Já nas linguagens declarativas, como o SQL, o programador declara o que pretende e o SGDB escolhe o caminho para a execução da consulta. Portanto, a otimização das consultas em linguagens declarativas é especialmente importante nas linguagens de programação declarativas.

Critério de correção:
- 3 décimas, definição e exemplo
- 2 décimas, relação
- erros, omissões, redundâncias ou indentação desadequada: -20% a -100%

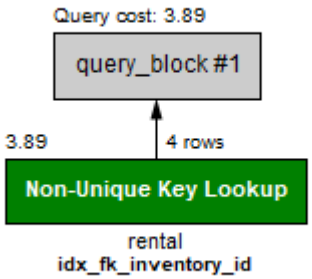
5) Relativamente ao Cap. 13 - Query Optimization
a) Instale no seu computador o SGBD MySQL.
b) Considere a base de dados de Aluguer de DVD com nome de Sakila dos exemplos do MySQL.
c) Escreva uma consulta que devolva os clientes que alugaram o filme 100.
d) Mostre o plano de execução disponível no MySQL e utilize o comando EXPLAIN. Analise e comente os resultados.

**Resposta:**

c) consulta SQL

```
-- EXPLAIN
SELECT *
FROM sakila.rental
WHERE inventory_id =100
```

d) plano em 'visual explain'




A consulta devolve 4 linhas com um custo estimado de 3,89

Critério de correção:
- 3 décimas, consulta no MySQL
- 2 décimas, explicação do EXPLAIN
- erros, omissões, redundâncias ou indentação desadequada: -20% a -100%

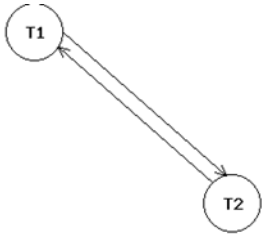
6) Relativo ao Cap. 14 – Transactions
Considere as seguintes transações T1 e T2 com os dados P e Q iniciados a zero:
T1: read (P); read (Q); if P == 0 then Q = Q + 1; write (Q);
T2: read (Q); read (P); if Q == 0 then P = P + 1; write (P);
Qualquer sequência que não seja em série de T1 e T2, leva a:
i) um escalonamento serializável
ii) um escalonamento serializável a conflitos
iii) um escalonamento não serializável a conflitos
Justifique a resposta e exemplifique.

**Resposta:**
Ponto iii) um escalonamento não serializável a conflitos;

Dada uma sequência de T1 e T2 que não seja em série como a seguinte, em primeiro lugar identificamos a utilização dos recursos P e Q, que denota que existe ciclicidade, pelo que o escalonamento não serializável a conflitos.

| **T1** | **T2** |
|---|---|
| r(P) | |
| | r(Q) |
| | r(P) |
| | w(P) |
| r(Q) | |
| w(Q) | |

| **P** | **Q** |
|---|---|
| r1 | |
| | r2 |
| r2 | |
| w2 | |
| | r1 |
| | w1 |




Critério de correção:
- 1 escolher iii)
- 4 justificação e exemplo
- erros, omissões, redundâncias ou indentação desadequada: -20% a -100%