21053 - Fundamentos de Bases de Dados
2015-2016
e-fólio B
Resolução e Critérios de Correção

PARA A RESOLUÇÃO DO E-FÓLIO, ACONSELHA-SE QUE LEIA ATENTAMENTE O SEGUINTE:

1) O e-fólio é constituído por 3 perguntas. A cotação global é de 3 valores.

2) 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.

3) Não são aceites e-fólios manuscritos, i.e. tem penalização de 100%.

4) O nome do ficheiro deve seguir a normal "eFolioB" + <nº estudante> + <nome estudante com o máximo de 3 palavras>

5) 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.

6) A interpretação das perguntas também faz parte da sua resolução, se encontrar alguma ambiguidade deve indicar claramente como foi resolvida.

7) A legibilidade, a objectividade e a clareza nas respostas serão valorizadas, pelo que, a falta destas qualidades serão penalizadas.

A informação da avaliação do estudante está contida no **vetor das cotações**:

Questão:	1	2	3	
Cotação:	10	10	10	décimas

1) (cap.3) Considere a seguinte base de dados relativas a familiares

Person (name -> gender)
Parent (parent, child -> birthdate)
Parent.parent $\subseteq$ Person.name
Parent.child $\subseteq$ Person.name

a) crie uma consulta que devolva os irmãos de uma dada pessoa
b) crie uma consulta que devolva os primos de uma dada pessoa

Resposta:
a) irmãos de uma dada pessoa, são filhos do mesmo pai (ou mãe)

```
SELECT DISTINCT A.child AS irmaoA, B.child AS irmaoB
FROM Parent A, Parent B
WHERE A.parent = B.parent
And A.child <> B.child
And A.child like "luis*"
```

b) primos de uma dada pessoa
A resposta envolve 5 consultas/vistas que chamam umas às outras:
primoX -> primos -> tios -> irmãos -> pais -> parents

Consulta/vista pais:

```
SELECT DISTINCT parent as pai, child as filho
FROM parents;
```

Consulta/vista irmaos:

```
SELECT DISTINCT A.filho AS irmaoA, B.filho AS irmaoB
FROM pais AS A, pais AS B
WHERE A.filho <> B.filho
AND A.pai = B.pai;
```

Consulta/vista tios:

```
SELECT DISTINCT irmaos.irmaoA AS tio, pais.filho AS sobrinho
FROM irmaos, pais
WHERE irmaos.irmaoB = pais.pai;
```

Consulta/vista primos:

```
SELECT DISTINCT pais.filho AS primoA, tios.sobrinho AS primoB
FROM tios, pais
WHERE tios.tio= pais.pai;
```

Consulta primoX:

```
SELECT DISTINCT primos.primoB
FROM primos
WHERE primoA like (X+"*");
```

ou ainda:

```
select distinct p4.child
from parent as p1, parent as p2, parent as p3, parent as p4
where p1.child like "rita*"
and p2.child=p1.parent
and p3.parent=p2.parent
and p3.child<>p2.child
and p4.parent=p3.child
```

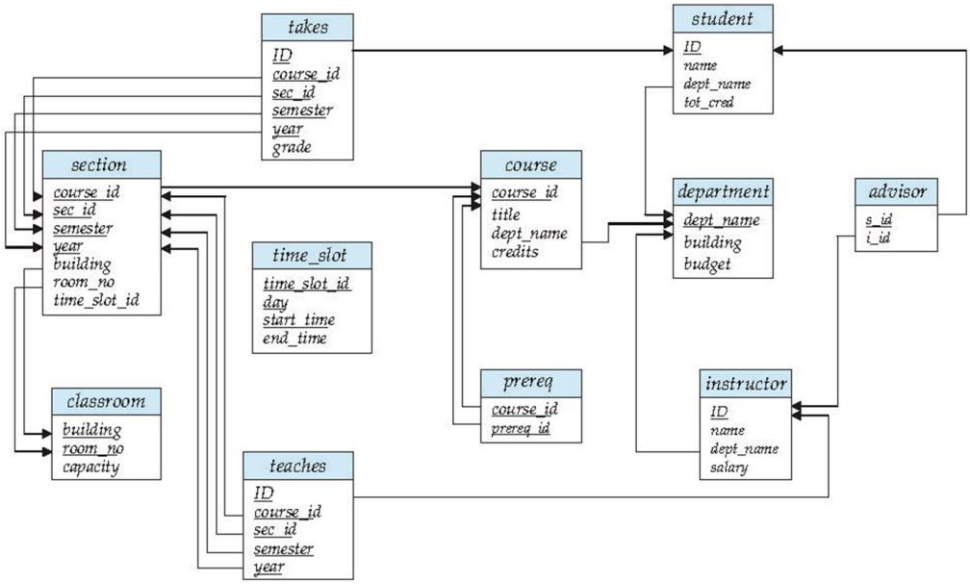
Em Access a consulta “primoX” apresenta-se da seguinte forma:

Solução alternativa apresentada por muitos estudantes:

```
select distinct p1.child from parent as p1 where p1.parent in
        (select p2.child from  parent p2  where p2.parent in
                (select p3.parent from parent p3 where p3.child in
                        (select p4.parent from parent p4 where p4.child like "rita*")))
and p1.parent not in
       (select p5.parent from parent p5 where child like "rita*");
```

Critérios de correção:
alínea a) 5 décimas
alínea b) 5 décimas
- erros, omissões, redundâncias ou indentação desadequada: -20% a -100%

2&3) Utilize o esquema de base de dados da universidade do manual, nas perguntas 2 e 3:



2) (cap.4) É esperado que um instrutor não possa ensinar em duas ou mais aulas (sections) no mesmo semestre em simultâneo (time_slot_id).

a) Escreve uma consulta SQL (instructor, section, timeslot_id) que devolva todos os casos que violam a restrição.

```
SELECT ID, I.name, S.section_id, S.semester, S.year, S.time_slot_id,
        COUNT (distinct S.building, S.room_no)
FROM instructor I, teaches T, section S
WHERE I.ID =T.ID
AND T.course_id = S.course_id
AND T.sec_id    = S.sec_id
AND T.semester  = S.semester
AND T.year      = S.year
GROUP BY (ID, I.name, S.section_id, S.semester, S.year, S.time_slot_id)
HAVING COUNT (S.building, S.room_no) > 1
```

b) Escreve uma asserção (predicado que deve ser sempre verdadeiro) para reforçar a restrição da alínea anterior.

```
CREATE ASSERTION instructor_ubiquo
CHECK NOT EXISTS
        (SELECT ID, I.name, S.section_id, S.semester, S.year, S.time_slot_id,
        COUNT (distinct S.building, S.room_no)
        FROM instructor I, teaches T, section S
        WHERE I.ID =T.ID
        AND T.course_id = S.course_id
        AND T.sec_id      = S.sec_id
        AND T.semester  = S.semester
        AND T.year        = S.year
        GROUP BY (ID, I.name, S.section_id, S.semester, S.year, S.time_slot_id)
        HAVING COUNT (S.building, S.room_no) > 1  )
```

Critérios de correção:
alínea a) 5 décimas
alínea b) 5 décimas
- erros, omissões, redundâncias ou indentação desadequada: -20% a -100%

3) (cap.6 e bibliografa adicional) Escreva em Álgebra Relacional, consideração os operadores de σ, Π, |><| e G para as funções agregadoras de G_{sum}, G_{count}, $G_{average}$, etc.

a) Qual o identificador dos estudantes que foram ensinados pelo instrutor com o nome de Einstein? Verifique se existem linhas duplicadas.

b) Qual o salário mais alto do conjunto dos instrutores?

Resposta:

a) estudantes de Einstein

$\Pi_{takes.ID}$ ($\sigma_{instructor.name='Einstein'}$ (takes |><| section |><| teaches |><| instructor))
Em álgebra relacional não existe a necessidade de remover linhas duplicadas.

b) Qual o salário mais alto do conjunto dos instrutores?

$G_{max(salary)}$ (instructor)

Critérios de correção:
alínea a) 5 décimas
alínea b) 5 décimas
- erros, omissões, redundâncias ou indentação desadequada: -20% a -100%
- nota: devem considerar "|><|" para junção de tabelas e "x" para produto cartesiano