DI- FCT-NOVA **5 de junho de 2018**

Bases de Dados

3º teste, 2017/18

Uma resolução

Grupo I

1 a)

```
nome_pessoas_1996(Nome) :- pessoas(_,Nome,D,_), D >= '1996-01-01'.
```

1 b)

```
clientes_AC(Nome) :- pessoas(P,Nome,_,M),
                     moradas(M,_,_,'Almada'),
                     contratos(_,_,_,P,_).
clientes_AC(Nome) :- pessoas(P,Nome,_,M),
                     moradas(M,_,_,'Caparica'),
                     contratos(_,_,_,P,_).
```

1 c)

```
nome_func(Nome) :- pessoas(P,Nome,_,M),
                   funcionário(P,_,_)),
                   contratos(C,_,_,_,P),
                   assinaturas(C,S),
                   serviços(S,'Fibra100',_).
```

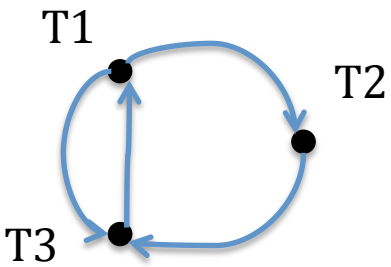
1 d)

```
preços_servicos(S,P) :- serviços(S,_,P), not assinado(S).
assinado(S) :- assinaturas(_,S).
```

Grupo II

1a)

O escalonamento não é serializável de conflito pois o grafo de precedências é cíclico.



Por exemplo, os conflitos que originam o problema são o read(x) de T1 com o write(x) de T3, e o read(z) de T3 com o write(z) de T1.

2a)

As possibilidades de execução em série são T1 seguido de T2 e T2 seguido de T1. Na primeira hipótese os valores de A3 e B2 são respectivamente 2 e 3, enquanto que na 2ª execução os valores de B2 e A3 são respetivamente 1 e 3.

2 b)

Um escalonamento possível seria A1, A2, B1, B2, B3, A3, A4. Em modo read committed os valores retornados por B2 e A3 seriam 1 e 3, enquanto que em read uncommited os valores seriam 3 e 3, respetivamente.

2 c)

Sim, o escalonamento é serializável pois a sua execução em modo read committed corresponde à execução de T2 seguida de T1. A transação T2 quando executa a operação B2 só vê a sua inserção enquanto que a operação A3 já vê o tuplo inserido pela transação T1.

Grupo III

1a)

```
<FCTel>
        <cliente idP="P1">
              <nome>Alfredo Almeida</nome>
              <dataNasc>1976-04-03</dataNasc>
              <morada idM="M1" rua="Rua do Mar" numandar="3-4°Esq" localidade="Caparica"></morada>
        </cliente>
        <funcionário idP="P2" salário="1000" dept="vendas">
              <nome>Maria Manuela</nome>
              <dataNasc>1980-05-23</dataNasc>
              <morada idM="M2" rua="Rua da Praia" numandar="3" localidade="Lisboa"></morada>
              <morada idM="M3" rua="Rua do Sol" numandar="27A" localidade="Caparica"></morada>
        </funcionário>
</FCTel>
```

2a)

```
/FCTel/funcionário/dataNasc
```

2b)

```
//morada[@localidade='Caparica']/..
```

2c)

```
//funcionárionome = //cliente/nome]
```

3)

Devolve tantos elementos pessoa quantos clientes ou funcionário, com o seu nome, data de nascimento e a primeira morada por extenso.

```
<pessoa idP="P1">
   <nome>Alfredo Almeida</nome>
   <dataNasc>1976-04-03</dataNasc>
   <morada>Rua do Mar, 3-4°Esq, Caparica</morada>
</pessoa>
<pessoa idP="P2">
   <nome>Maria Manuela</nome>
   <dataNasc>1980-05-23</dataNasc>
   <morada>Rua da Praia, 3, Lisboa</morada>
</pessoa>
```

4)

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE FCTel[
<!ELEMENT FCTel (pessoa*,morada*)>
<!ELEMENT pessoa (funcionário?, nome, dataNasc)>
<!ATTLIST pessoa idP ID #REQUIRED morada IDREF #REQUIRED>
<!ELEMENT funcionário EMPTY>
<!ATTLIST funcionário salário CDATA #REQUIRED dept CDATA #REQUIRED>
<!ELEMENT nome (#PCDATA)>
<!ELEMENT dataNasc (#PCDATA)>
<!ELEMENT morada EMPTY>
<!ATTLIST morada idM ID #REQUIRED
          rua CDATA #REQUIRED
          numandar CDATA #REQUIRED
          cpostal CDATA #REQUIRED
          localidade CDATA #REQUIRED>
]>
```