DI-FCT-NOVA **18 de junho de 2016**

Bases de Dados (10640)

Exame de Recurso, 2015/16

Uma proposta de resolução

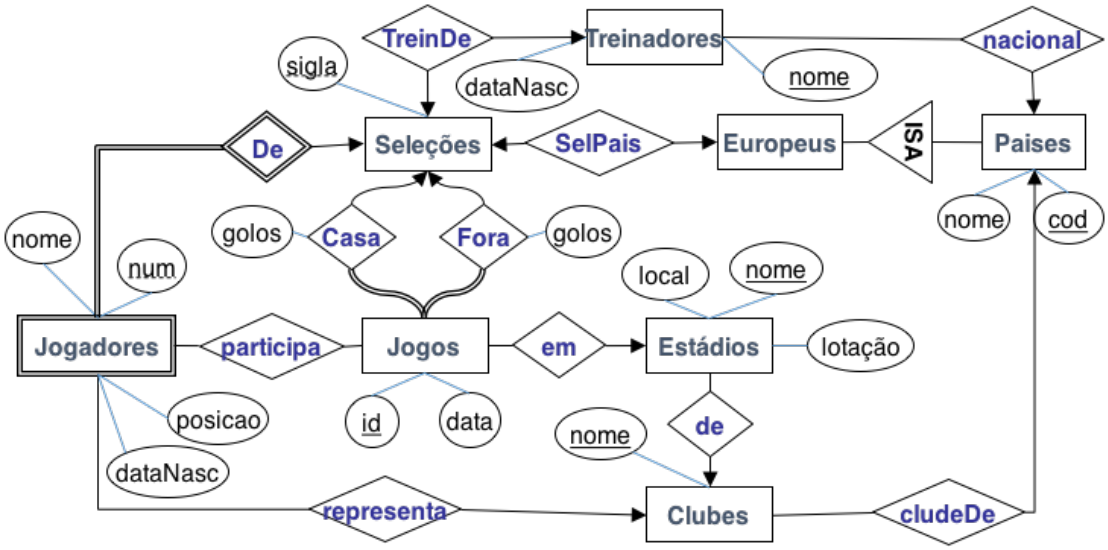
Parte I

(correspondente à matéria do 1º teste)

Grupo I.1

1.

a)



b)

```
create table paises(
    cod varchar(3) primary key,
    nome varchar(20),
);
```

```
create table europeus(
    codEu varchar(3) primary key,
    foreign key (codEU) references paises(cod)
);
```

```
create table selecoes(
    codS varchar(3) primary key,
    sigla varchar(10) not null,
    treinador varchar(40) not null,
    foreign key (codS) references paises(EU),
    foreign key (treinador) references treinadores
);
```

```
create table treinadores(
    nome varchar(40) primary key,
    dataNasc date,
    nacional varchar(3),
    foreign key (nacional) references paises(cod)
);
```

```
create table jogadores(
    num number,
    selecao varchar(3),
    nome varchar(40) not null,
    dataNasc date,
    posição varchar(12),
    clube varchar(30),
    primary key (num, selecao),
    foreign key (selecao) references selecoes,
    foreign key clube references clubes,
    check posicao in ('GR','D','M','A')
);
```

```
create table jogos(
    id number primary key,
    sCasa varchar(3) not null,
    sFora varchar(3) not null,
    golosCasa number,
    golosFora number,
    estadio varchar(20),
    data date,
    foreign key (sCasa) references selecoes,
    foreign key (sFora) references selecoes,
    foreign key (estadio) references estadios,
);
```

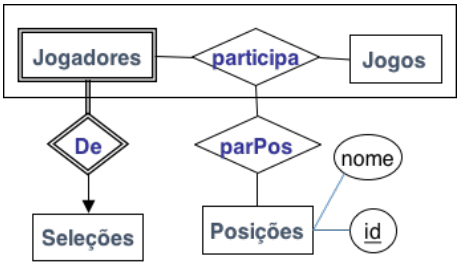
```
create table clubes(
    nome varchar(30) primary key,
    pais varchar(3),
    foreign key (pais) references paises(cod),
);
```

```
create table estadios(
    nome varchar(20) primary key,
    clube varchar(30),
    foreign key (clube) references clubes(nome),
);
```

```
create table participa(
    num number,
    selecao varchar(3),
    jogo number,
    primary key (num, selecao, jogo),
    foreign key (num,selecao) references jogadores(num,selecao),
    foreign key (jogo) references jogos(id),
);
```

2.

a) Apenas se apresentam os atributos da nova entidade "Posições". Os das outras entidades são os que se encontram em **1b)**:



b) As tabelas de jogadores, jogos, seleções e participa, são como em **1b)**:

```
create table parPos(
    num number,
    selecao varchar(3),
    jogo number,
    posição number,
    primary key (num, selecao, jogo, posição),
    foreign key (num, selecao, jogo) references participa,
    foreign key (posição) references posições(id),
);
```

```
create table posições(
    id number primary key,
    nome varchar(20)
);
```

Grupo I.2

a) A única chave candidata é {Jogador, Jogo, Posição}

b) Jogador → Número, Seleção; Jogo → Árbitro; Árbitro → Nacionalidade

c) R1 = (Jogador, Número, Seleção), R2 = (Jogo, Árbitro), R3 = (Árbitro, Nacionalidade), R4 = (Jogador, Jogo, Posição)

d) Sim, preserva as dependências. Para tal basta ver que a decomposição acima coincide com a que se obteria para a 3ª forma normal.

Parte II

(correspondente à matéria do 2º teste)

Descrição de uma base de dados de eventos de jogo, e de resultados de jogos.

Grupo II.1

1. Apresente uma expressão em **álgebra relacional e uma consulta SQL** para cada uma das perguntas:

a) $\Pi_{nomeJog}\left(jogadores \bowtie equipaInicial \bowtie \sigma_{data=14/06/2016}(jogos)\right)$

```
select nomeJog
from jogadores natural inner join equipaInicial natural inner join jogos
where data = 14/06/2016;
```

b) $\Pi_{idJogador}(jogadores) - \Pi_{idJogador}(equipaInicial)$

```
(select idJogador from jogadores) except (select idJogador from equipaInicial);
```

c)

$r \leftarrow \sigma_{nomePais='Portugal'}(equipaInicial \bowtie selecoes))$
$\Pi_{idJogador,idJogo}(r) \div \Pi_{idJogo}(r)$

```
with R as
( select idJogador, idJogo from equipainicial natural inner join selecoes
  where nomePais = 'Portugal')
select idJogador
from r A
where   (select distinct idJogo from r B  where B.idJogador = A.idJogador)
        contains
        (select distinct idJogo from r);
```

2. Apresente **uma consulta SQL** para cada uma das perguntas:

a)
```
select tipo, minuto , texto
from ocorrências, seleções P, seleções I, selecaoJogo JP, selecaoJogo JI
where   P.nomePais = 'Portugal' and I.nomePais = 'Islândia' and
        P.idSeleção = JP.idSeleção and I.idSeleção = JI.idSelecao and
        JP.idJogo = JI.idJogo and ocorrências.idJogo = JP.idJogo
order by minuto desc;
```

b)
```
with S as (select * from ocorrências natural inner join substituições)
select jogos.idJogo, count(S.idOccor)
from jogos J left outer join S on (J.idJogo = S.idJogo)
group by jogos.idJogo;
```

c)
```
with G as (  select * from ocorrências natural inner join golos),
     Q as (  select S.idJogo, S.idSeleção, count(G.idOcorr) as quantos
             from selecaoJogo S left outer join G
                  on (S.idJogo = G.idJogo and S. idSeleção = G. idSeleção)
             group by S.idJogo, S.idSeleção)
select idSeleção from Q group by idSelecao having avg(quantos) > 1;
```

Grupo II.2

1.

a)
```
create assertion duasEquipas check
not exists (select idJogo from selecaoJogo group by idJogo having count(idSelecao) != 2);
```

b)
```
alter table equipaInicial add constraint eqIni
        foreign key (idJogador,idSelecao) references jogadores(idJogador,idSelecao);
```

c) **create assertion** selecaoGolo **check**
((select * from selecaoJogo**)**
contains
(select idJogo, idSeleção **from** golos **natural inner join** ocorrências**))**

2.

a) **create view** ocorrenciasGolo **as (select * from** ocorrências **natural inner join** golos**);**

create view golosEquipaJogo **as**
(select S.idJogo, S.idSeleção**, count(**G.idOccor**) as** golos
from selecaoJogo S **left outer join ocorrenciasGolo** G
on (S.idJogo = G.idJogo **and** S. idSeleção = G. idSeleção**)**
group by S.idJogo, S.idSeleção**);**

b) **create trigger** atualizaMarcador **on insert of** ocorrência
referencing new row as nrow
for each row
when nrow.tipo = 'Fim do jogo'
begin
insert into golosFinal **select** * **from** golosEquipaJogo**;**

end;

Parte III

(correspondente à matéria do 3º teste)

Grupo III.1

1.

a) idJogo(ID,Nome1,Nome2) :-
seleções(S1,Nome1), seleções(S2,Nome2),
selecaoJogo(ID,S1), selecaoJogo(ID,S2).

resultado1(Nome) :-
idJogo(ID, 'Portugal','Islândia'), seleções(PT,'Portugal')
equipaInicial(ID,PT,Jog), jogadores(Jog,PT,Nome).

b) alinharam(Jog) :- equipaInicial(_,_,Jog).

resultado2(Nome) :- jogadores(Jog,_,Nome), not alinharam(Jog).

c) saiu(IdJogo, Jog) :- ocorrências(Id, IdJogo, _, _, _), substituições(Id,Jog,_).

entrou(IdJogo, Jog) :- ocorrências(Id, IdJogo, _, _, _), substituições(Id,_ ,Jog).

resultado3(Jog) :-
idJogo(ID, 'Portugal','Islândia'), equipaInicial(ID,_,Jog), not saiu(ID,Jog).

resultado3(Jog) :-
idJogo(ID, 'Portugal','Islândia'), entrou(ID,Jog), not saiu(ID,Jog).

Grupo III.2

1. Como para cada jogo é imposto que, ou não há nenhum tuplo na tabela selecaoJogo, ou então há dois tuplos, se se verificasse a consistência no fim de cada ação, acabava por ser impossível introduzir tuplos nessa tabela. Com efeito, mal se introduzisse o primeiro tuplo relativo a um jogo, a consistência seria imediatamente violado (pois passaria a haver um só tuplo para esse jogo). Assim, a única forma de introduzir e.g. que no jogo com id=12 participaram as seleções 'PT' e 'IS' seria através de uma transação como a abaixo, em que a consistência é só verificada no final.

begin transaction
insert into selecaoJogo **values (**12,'PT'**);**
insert into selecaoJogo **values (**12,'IS'**);**
commit;

2. Considere-se a seguinte identificação para cada operação:

Transação 1	Transação 2
A1. begin transaction	**B1. begin transaction**
A2. select * **from** golos**;**	**B2. select** * **from** golos**;**
A3. insert into golos **values** (1,...);	**B3. insert into** golos **values** (2,...);
A4. select * **from** golos**;**	**B4. select** * **from** golos**;**
A5. commit;	**B5 commit;**

a) Considere o escalonamento com a seguinte ordem das operações: A1; B1; A2; B2; A3; B3; A4; B4; A5; B5. Neste caso, em modo **read uncommitted** a operação A4 devolveria os dois tuplos (introduzidos em A3 e em B3), enquanto que em modo **read committed** devolveria apenas o tuplo introduzido em A3.

b) Considere agora o escalonamento com a seguinte ordem das operações: A1; B1; A2; B2; A3; B3; A4; A5; B4; B5. Neste caso, em modo **read committed** a operação B4 devolveria os dois tuplos (introduzidos em A3 e em B3), enquanto que em modo **serializable** devolveria apenas o tuplo introduzido em B3.

Grupo III.3

1.

```
<?xml version="1.0" encoding="UTF-8"?>
<jogo data = “14/06/2016”>
    <equipaCasa idEquipa = “PT”>
        <nomeEquipa>Portugal</nomeEquipa>
    </equipaCasa>
    <equipaFora idEquipa = “IS”>
        <nomeEquipa>Islândia</nomeEquipa>
    </equipaCasa>
    <ocorrencia>
        <golo equipa = “PT” minuto = 31>
            <marcador>Nani</marcador>
        </golo>
    </ocorrencia>
    <ocorrencia>
        <substituicao minuto = 75>
            <sai>João Mário</sai>
            <entra>Quaresma</entra>
        </substituicao>
    </ocorrencia>
</jogo>
```

2. Apresente expressões XPATH sobre um ficheiro de acordo com o DTD acima que devolvam os resultados das seguintes perguntas:

a) //marcador

b) //substituicao[@minuto =< 45]/entra

c) //golo[@minuto > 45]/id(@equipa)/nomeEquipa

3.

```
<!DOCTYPE jogo[
    <!Element jogo(equipaCasa, equipaFora, jogador*, ocorrencia*)>
        <!ATTLIST jogo data #REQUIRED >
    <!Element equipaCasa(nomeEquipa)>
        <!ATTLIST jogo idEquipa ID #REQUIRED >
    <!Element equipaFora(nomeEquipa)>
        <!ATTLIST jogo idEquipa ID #REQUIRED >
    <!Element jogador(nomeJog)>
        <!ATTLIST jogador idJog ID #REQUIRED equipa IDREF #REQUIRED>
    <!Element ocorrencia(golo | substituicao)>
    <!Element golo(marcador)>
        <!ATTLIST golo equipa IDREF #REQUIRED minuto>
    <!Element marcador EMPTY>
        <!ATTLIST marcador jogador IDREF #REQUIRED>
    <!Element substituicao(sai,entra)>
        <!ATTLIST substituicao minuto equipa IDREF #REQUIRED>
    <!Element sai EMPTY>
        <!ATTLIST sai jogador IDREF #REQUIRED>
    <!Element entra EMPTY>
        <!ATTLIST entra jogador IDREF #REQUIRED>
    <!Element nomeEquipa(#PCDATA)>
    <!Element nomeJog(#PCDATA)>
]>
```

Grupo III.4

1. **create type** jogador **as** (nome **varchar(**30**)**, numero **number**);
   create table equipaInicial(
   idJogo **number**,
   idSeleção **varchar**(2),
   inicio jogador **multiset);**

2. **select distinct** R.idJogo
   from equipaInicial R, **unnest**(R.inicio) **as** S
   where S.nome = 'Ronaldo' **and** S.numero = 7;