Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina Banco de Dados
AD2 2° semestre de 2007.

Nome –

Observações:

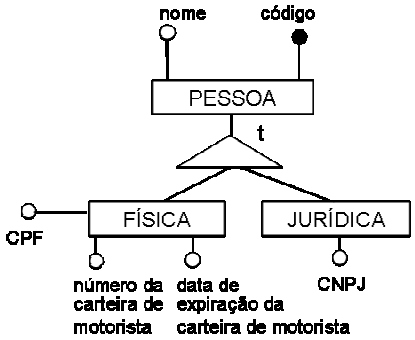
1. Prova com consulta.
2. Não aceitaremos provas feitas à mão.

Questão 1

Considere o diagrama ER mostrado abaixo. Construa dois esquemas relacionais equivalentes a este diagrama ER:

(a) usando tabela única para a hierarquia de generalização/especialização; (vale 1,25 pontos)

(b) usando uma tabela para cada entidade envolvida na generalização/especialização (vale 1,25 pontos).

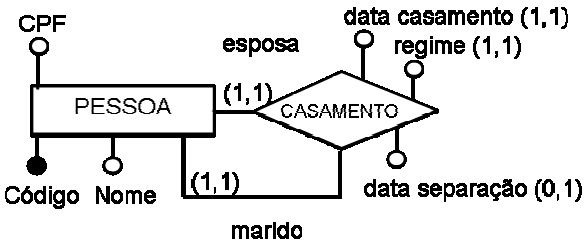


Resposta: Alternativa (a)
Pessoa(Cod, Nome, **Tipo**, CPF, NumCartMotorista, DataExpCartMotorista, CNPJ)

Resposta: Alternativa (b)
Pessoa(Cod, Nome, **Tipo**)
PessoaFísica(CodPes, CPF, NumCartMotorista, DataExpCartMotorista)
CodPes REFERENCIA Pessoa
PessoaJurídica(CodPes, CNPJ)
CodPes REFERENCIA Pessoa

Questão 2 (vale 2,5 pontos)

Considere o diagrama ER abaixo. Construa o esquema relacional equivalente a este diagrama ER, considerando a alternativa tabela própria para representar o relacionamento.



Resposta:

Pessoa(Cod, Nome, CPF)
Casamento(CodEsposa, CodMarido, DataCasamento, Regime, DataSeparação)
CodEsposa REFERENCIA Pessoa
CodMarido REFERENCIA Marido

Questão 3 (vale 2,5 pontos)

Responda cada uma das questões seguintes. As questões são baseadas no seguinte esquema relacional:

Estado (sigla: string, nome: string)
Time (codTime: integer, nome: string, siglaEstado: string)
SiglaEstado REFERENCIA Estado
Jogador (codJogador: integer, nome: string, idade: integer, salário: float, codTime: integer)
codTime REFERENCIA Time

No esquema acima, as chaves primárias estão sublinhadas.

(a) Escreva os comandos SQL necessários para criar as relações acima, incluindo as restrições de chave primária e chave estrangeira (vale 0,35 pontos).

```
CREATE TABLE Estado (
        Sigla VARCHAR(2) NOT NULL,
        Nome VARCHAR(30),
        PRIMARY KEY (Sigla)
        )
CREATE TABLE Time (
        CodTime INTEGER NOT NULL,
        Nome VARCHAR(30),
        SiglaEstado VARCHAR(2),
        PRIMARY KEY (CodTime),
```

FOREIGN KEY (SiglaEstado) REFERENCES Estado ON DELETE SET NULL
)

A chave estrangeira acima também poderia ter sido definida usando ON DELETE RESTRICT ou ON DELETE CASCADE.

CREATE TABLE Jogador (
CodJogador INTEGER NOT NULL,
Nome VARCHAR(30),
Idade INTEGER,
Salário FLOAT,
CodTime INTEGER,
PRIMARY KEY (CodJogador),
FOREIGN KEY (CodTime) REFERENCES Time ON DELETE SET NULL
)

A chave estrangeira acima também poderia ter sido definida usando ON DELETE RESTRICT ou ON DELETE CASCADE.

(b) Quais são as opções para garantir esta restrição de chave estrangeira na tabela Time, quando um usuário tenta excluir uma tupla de estado? (vale 0,35 pontos).

As opções para garantir a restrição quando o usuário tenta excluir uma tupla de Estado são as seguintes:

(a) Especificar o comando CREATE TABLE da tabela Time usando:

FOREIGN KEY (SiglaEstado) REFERENCES Estado

Neste caso, o SGBD não deixa que uma tupla de *Estado* seja excluída quando houver uma ou mais tuplas relacionadas em *Time*. O comando de criação acima utiliza a definição *default* de declaração de restrição de integridade, que é ON DELETE RESTRICT.

(b) Especificar o comando CREATE TABLE acima usando ON DELETE CASCADE na definição da chave estrangeira.

FOREIGN KEY (SiglaEstado) REFERENCES Estado ON DELETE CASCADE

Neste caso, ao excluir uma tupla de *Estado*, todas as tuplas relacionadas da tabela *Time* também serão excluídas.

(c) Especificar o comando CREATE TABLE usando ON DELETE SET NULL na definição da chave estrangeira.

FOREIGN KEY (SiglaEstado) REFERENCES Estado ON DELETE SET NULL

Neste caso, ao excluir uma tupla de *Estado*, todas as tuplas relacionadas da tabela *Time* terão o conteúdo da coluna *siglaEstado* alterados para NULL.

(c) Escreva uma instrução SQL para adicionar "Ronaldinho Gaúcho" como um jogador com cód = 101, idade = 32, salário = 100000, jogando no time cód 2 (vale 0,3 pontos).

INSERT INTO JOGADOR (CodJogador, Nome, Idade, Salário, CodTime)
VALUES (101, "Ronaldinho Gaúcho", 32, 100000, 2)

ou

INSERT INTO JOGADOR
VALUES (101, "Ronaldinho Gaúcho", 32, 100000, 2)

(d) Escreva uma instrução SQL para aumentar em um ano a idade de todos os jogadores (vale 0,3 pontos).

UPDATE JOGADOR
SET IDADE= IDADE + 1

(e) Escreva uma instrução SQL que exclua o time de nome "Tabajara". Dadas as restrições de integridade referencial que você definiu para o esquema, explique o que acontece quando esta instrução é executada (vale 0,3 pontos).

DELETE FROM TIME
WHERE CodTime IN (SELECT CodTime FROM Time WHERE NOME="Tabajara")

A resposta da segunda parte desta questão depende de como o aluno especificou a chave estrangeira para a tabela *Time* na tabela *Jogador*, na questão 2(a). As respostas para cada opção estão abaixo:

- FOREIGN KEY (CodTime) REFERENCES Time ON DELETE RESTRICT

Neste caso, ao excluir o *Time* Tabajara, se houver alguma tupla em *Jogador* que referencia este time, a exclusão será negada pelo SGBD.

- FOREIGN KEY (CodTime) REFERENCES Time ON DELETE SET NULL

Neste caso, ao excluir o *Time* Tabajara, se houver alguma tupla em *Jogador* que referencia este time, a coluna *CodTime* desta(s) tupla(s) da tabela *Jogador* será(ão) alteradas para NULL.

- FOREIGN KEY (CodTime) REFERENCES Time ON DELETE CASCADE

Neste caso, ao excluir *Time* Tabajara, se houver alguma tupla em *Jogador* que referencia este time, esta(s) tupla(s) será(ão) excluída(s) da tabela *Jogador*.

(f) Escreva uma instrução SQL que altere a o salário de todos os jogadores que jogam no time "ACME" para 50000.

UPDATE JOGADOR
SET SALARIO=50000
WHERE CodTime IN (SELECT CodTime FROM Time WHERE Nome="ACME")

(g) Escreva uma consulta SQL que retorne a média salarial dos jogadores, por time (vale 0,3 pontos).

```
SELECT CodTime, AVG(SALARIO)
FROM JOGADOR j, TIME t
WHERE j.CodTime=t.CodTime
GROUP BY CodTime
```

(h) Escreva uma consulta SQL que retorna o nome do time que paga o maior salário (vale 0,3 pontos).

```
SELECT Nome
FROM Time
WHERE CodTime=(SELECT CodTime FROM Jogador WHERE SALARIO=(SELECT MAX(SALARIO) FROM JOGADOR))
```

ou

```
SELECT Nome
FROM Time
WHERE CodTime IN (SELECT CodTime FROM Jogador WHERE SALARIO=(SELECT MAX(SALARIO) FROM JOGADOR))
```

Questão 4 (vale 2,5 pontos sendo 0,25 para cada item)

Considere o seguinte esquema relacional:

Empregados(*eid:* integer, *enome:* string, *end:* string)
Projetos(*pid:* integer, *pnome:* string, *loc:* string)
Alocação(*eid:* integer, *pid:* integer, *dur:* integer)

No esquema acima, as chaves primárias estão sublinhadas. A tabela de Projetos indica o nome do projeto e o local de realização do projeto. A tabela de Alocação lista em *dur* a duração em meses em que cada empregado está alocado a cada projeto em que faz parte.
Apresente, para cada consulta a seguir, as expressões em álgebra relacional e as instruções SQL correspondentes às consultas.

1. Obtenha o nome dos projetos localizados em Niterói.

$\pi_{pnome}(\sigma_{loc="Niteroi"} Projetos)$

SQL

```
SELECT pnome
FROM Projetos
WHERE loc='Niteroi'
```

2. Obtenha o nome dos projetos que possuem empregados alocados a eles.

π_{pnome} (*Projetos* $\bowtie$ *Alocacao*)

SQL

```
SELECT pnome
FROM Projetos P, Alocacao C
WHERE  C.pid=P.pid
```

3. Obtenha o nome dos empregados alocados a algum projeto localizado em Niterói.

π_{enome} (π_{eid} ((π_{pid} $\sigma_{loc="Niteroi"}$ *Projetos*) $\bowtie$ *Alocacao*) $\bowtie$ *Empregados*)

SQL

```
SELECT E.enome
FROM Empregados E, Projetos P, Alocacao C
WHERE P.loc='Niteroi' AND C.pid=P.pid AND C.eid=E.eid
```

4. Obtenha o eid dos empregados alocados a algum projeto em Niteroi ou Buzios.

π_{eid} (π_{pid}($\sigma_{loc="Niteroi" \vee loc = "Buzios"}$ *Projetos*) $\bowtie$ *Alocacao*)

SQL

```
SELECT C.eid
FROM Alocacao C, Projetos P
WHERE (loc="Niteroi" OR loc = "Buzios")
AND P.pid = C.pid
```

5. Obtenha o eid dos empregados que estão alocados a algum projeto de Niteroi ou que estão no endereço “Rua sobe e desce”.

$\rho(R1, \pi_{eid}((\pi_{pid}\sigma_{loc="Niteroi"} Projetos) \bowtie Alocacao))$
$\rho(R2, \pi_{eid}\sigma_{end="Rua\ Sobe\ e\ Desce"} Empregados)$
$R1 \cup R2$

```
SQL
SELECT E.eid
FROM Empregados E
WHERE E.end = 'Rua Sobe e Desce'
      OR E.eid IN ( SELECT C.eid
                    FROM Projetos P, Alocacao C
                    WHERE P.loc='Niteroi' AND P.pid = C.pid )
```

6. Obtenha o eid dos empregados alocados a algum projeto de Niteroi e algum projeto de Buzios.

$\rho(R1, \pi_{eid}((\pi_{pid}\sigma_{loc='Niteroi'} Projetos) \bowtie Alocacao))$
$\rho(R2, \pi_{eid}((\pi_{pid}\sigma_{loc='Buzios'} Projetos) \bowtie Alocacao))$
$R1 \cap R2$

```
SQL
SELECT C.eid
FROM Projetos P, Alocacao C
WHERE P.loc = 'Niteroi' AND P.pid = C.pid
AND EXISTS ( SELECT P2.pid
FROM Projetos P2, Alocacao C2
WHERE P2. loc = 'Buzios' AND C2.eid = C.eid
AND P2.pid = C2.pid )
```

7. Obtenha o nome dos empregados que estão alocados a algum projeto com uma duração abaixo de 12 meses.

$\pi_{enome}((\sigma_{dur<12} Alocacao) \bowtie Empregados)$

SQL
```
SELECT E.enome
FROM Empregados E, Alocacao C
WHERE dur < 12
   AND C.eid=E.eid
```

8. Obtenha o nome dos empregados que estão alocados a algum projeto de Niteroi com uma duração abaixo de 24 meses.

$\pi_{enome}(\pi_{eid}((\pi_{pid}\sigma_{loc="Niteroi"} Projetos) \bowtie \sigma_{dur<24} Alocacao) \bowtie Empregados)$

SQL

```
SELECT F.enome
FROM Empregados F, Projetos P, Alocacao C
WHERE P.loc='Niteroi'
    AND dur < 24
    AND C.pid=P.pid AND C.eid=F.eid
```

9. Obtenha o nome dos Projetos em que está alocado o empregado "Jorge de Lima".

$$\rho(R1,\ \pi_{pid,\ pnome}\ (Projetos \bowtie Alocacao \bowtie (\sigma_{enome\ =\ 'Jorge\ de\ Lima'}\ Empregados)))$$

```
SQL
SELECT P.pnome
FROM Empregados E, Projetos P, Alocacao C
WHERE E.enome = 'Jorge de Lima'
AND      P.pid = C.pid AND C.eid=E.eid
```

10. Obtenha o pid e o nome dos Projetos em que está alocado exclusivamente o empregado "Jorge de Lima" , isto é, aqueles projetos que não possuem nenhum outro empregado alocado.

$$\rho(R1,\ \pi_{pid,\ pnome}\ (Projetos \bowtie Alocacao \bowtie (\sigma_{enome\ =\ 'Jorge\ de\ Lima'}\ Empregados)))$$
$$\rho(R2,\ \pi_{pid,\ pnome}\ (Projetos \bowtie Alocacao \bowtie (\sigma_{enome\ <>\ 'Jorge\ de\ Lima'}\ Empregados)))$$
$$R1 - R2$$

```
SQL
SELECT P.pid, P.pnome
FROM Empregados E, Projetos P, Alocacao C
WHERE E.enome = 'Jorge de Lima'
AND      P.pid = C.pid AND C.eid=E.eid
AND      NOT EXISTS ( SELECT *
                      FROM Alocacao C2, Empregados E2
                       WHERE E2.enome <> 'Jorge de Lima'
                       AND      C2.eid = E2.eid
                       AND      P.pid = C2.pid )
```