Prova de Fundamentos de Bancos de Dados

2^a Prova

Prof. Carlos A. Heuser

Novembro de 2009

Prova *com* consulta – duas horas de duração

Considere a base de dados abaixo (base de dados idêntica a da 1^a prova). Esta base de dados é usada por uma oficina de manutenção de automóveis.

/* tabela de clientes cadastrados na oficina */

```
CLIENTE (cpf, nome_cli)
```

/* tabela com dados dos automóveis dos clientes da oficina */

```
AUTOMOVEL (placa, no_chassis, modelo, cpf);
   (cpf) references CLIENTE
```

/* tabela com as revisões periódicas programadas e feitas – para cada automóvel, a oficina cadastra todas revisões programadas –
`Km` e `data_programada` são a quilometragem e a data em que deve ser feita a revisão –
`data_ultim_telef` serve para informar quando o pessoal da oficina ligou para o cliente lembrando da provável necessidade de fazer a revisão – caso o cliente não tenha sido chamado, este campo contém NULL
`data_executada` e `Km_executada` informa a data e a quilometragem de uma revisão que já foi executada – caso a revisão não tenha sido chamado, estes campos contém NULL) */

```
REVISAO (placa, Km, data_programada, data_ultim_telef,
         data_executada, Km_executada)
   (placa) references AUTOMOVEL
```

/* tabela com as peças usadas em cada revisão */

```
PECA_REVISAO (placa, Km, cod_peca, quantidade)
   (placa, Km) references REVISAO
   (cod_peca) references PECA
```

/* tabela com as descrições das peças */

```
PECA (cod_peca, descricao_peca)
```

1. *(Peso 1,5)*
   Sobre a base de dados acima, resolver a seguinte consulta usando *cálculo relacional*:

   Para cada automóvel, que tenho sofrido uma revisão na qual tenha sido utilizada uma peça com descrição "Lanterna traseira esquerda PX-L", obter a placa e o modelo do automóvel

   Solução:

```
{r|
   ∃ a ∈ Automovel (
       a.placa=r.placa ∧
       a.modelo=r.model ∧
       ∃ pr ∈ Peca_Revisao (
          pr.placa=a.plca ∧
          ∃ p ∈ Peca (
             p.cod_peca=pr.cod_peca ∧
             p.descricao_peca="Lanterna traseira esquerda PX-L"
             )
          )
      )
}
```

2. Sobre a base de dados acima, resolver as consultas abaixo usando *SQL*:

   2.a *(Peso 1,33...)*
   Obter cpfs nomes dos clientes que não têm nenhum automóvel no banco de dados.

   Solução:

```
SELECT cpf, nome_cli
FROM cliente
EXCEPT
(SELECT cpf, nome_cli
 FROM cliente
      NATURAL JOIN
      automovel
)
```

ou

```
SELECT cpf, nome_cli
FROM cliente
WHERE
   cpf NOT IN
     (SELECT cpf
      FROM automovel
     )
```

2.b *(Peso 1,33...)*

Obter *uma* tabela contendo as seguintes colunas:

i. Cpf e nome de cada cliente;

ii. placa e modelo de cada automóvel que o cliente possui; (se o cliente não possuir automóveis, estas colunas devem aparecer vazias);

iii. data executada e quilometragem executada para cada revisão deste automóvel, que tenha sido executada com mais de 30.000Km (se não houver revisão nestas condições, estas colunas devem aparecer vazias).

Solução:

```
SELECT
   Cli.cpf,
   Cli.nome_cli,
   Aut.placa,
   Aut.modelo,
   Rev.data_executada,
   Rev.Km_executada
FROM (Cliente Cli
       NATURAL LEFT  JOIN
      Automovel Aut
     )
       LEFT  JOIN
     Revisao Rev
       ON (Aut.placa=Rev.placa AND
           Rev.Km_executada > 30000
          )
```

2.c *(Peso 1,33...)*

Foi detetado um problema na base de dados, gerado por um erro de software. Para alguma revisões, erroneamente, está informada que elas usaram todas peças cadastradas na base de dados. Obter os identificadores destas revisões (`placa, Km`)

Solução:

```
SELECT Revisao.placa,
       Revisao.Km
FROM   Revisao
WHERE
   NOT EXISTS (
      SELECT * FROM Peca
      WHERE
         NOT EXISTS (
            SELECT * FROM Peca_Revisao
               WHERE Peca.cod_peca
                       = Peca_Revisao.cod_peca AND
                     Revisao.placa
                       = Peca_revisao.placa AND
                     Revisao.Km
                       = Peca_revisa.Km
                     )
        )
```

2.d *(Peso 1)*

Deseja-se saber quantos automóveis já foram revisados.

Solução:

```
SELECT COUNT (DISTINCT placa)
FROM Revisao
WHERE data_executada IS NOT NULL
```

2.e *(Peso 2)*

Para cada cliente, obter seu cpf, seu nome e o número de automóveis que ele possui.

Resolver esta consulta considerando duas variantes:

i. Mesmo clientes que não têm automóveis registrados devem aparecer no resultado.
Solução:

```
SELECT Cli.cpf,
       Cli.nome_cli,
       COUNT (placa) AS NoDeAutomoveis
FROM Cliente Cli
      NATURAL LEFT JOIN
     Automovel
GROUP BY Cli.cpf,
         Cli.nome_cli
```

ii. Apenas os clientes com mais de três automóveis devem aparecer no resultado.
Solução:

```
SELECT Cli.cpf,
       Cli.nome_cli,
       COUNT (placa) AS NoDeAutomoveis
FROM Cliente Cli
      NATURAL  JOIN
     Automovel
GROUP BY Cli.cpf,
         Cli.nome_cli
HAVING COUNT(placa) > 3
```

3. *(Peso 1,5)*
   Considere a seguinte restrição de integridade: "Todas revisões já executadas devem ter os campos `data_executada` e `Km_executada` preenchidos (isto é, não `NULL`). Já as revisões programadas (ainda não executadas) devem ter estes campos vazios (`NULL`)."

   Esta restrição de integridade pode ser especificada através de um `CHECK` *constraint*. Escreva este *constraint*.

```
ALTER TABLE REVISAO
  ADD CONSTRAINT validaKM
     CHECK
        (data_executada IS NULL AND
         Km_executada IS NULL)
        OR
        (data_executada IS NOT NULL AND
         Km_executada IS NOT NULL)
```