Stored Procedures

8. Atividades de Laboratório

Para os propósitos das atividades abaixo, você precisará do script “SQL Batch” da aula anterior e das seguintes tabelas já criadas no banco de dados com os seguintes esquemas:

```
tb_pessoas  = (Cod_Pes, Nome, Endereco)
tb_carros   = (Placa, Ano, Modelo)
tb_possui   = ((Cod_Pes, Placa),
                  (Cod_Pes) references tb_pessoas,
                  (Placa) references tb_carros)
tb_acidentes= ((Placa, Data_hora, Valor_Danos, Cod_Pes_Motorista),
                  (Placa) references tb_carros,
                  (Cod_Pes_Motorista) references tb_pessoas)
```

Os scripts produzidos nas atividades de hoje devem ser salvos no arquivo de respostas de nome:

```
ra99999999_proc1.txt,
```

substituindo a parte inicial do nome pelo seu registro acadêmico. Ao final, o arquivo deve ser enviado ao servidor.

8.1. Stored Procedure para obter dados dos amigos diretos e indiretos

Siga os passos abaixo para criar a primeira stored procedure.

a) Tabelas auxiliares

Você já possui todas as tabelas necessárias resultantes das atividades de aulas precedentes. Se ainda não as criou, execute os seguintes comandos:

```
create table tb_pessoas_ligadas(nivel int not null,
                                cod_pes char(7) not null,
                                cod_pes_anterior char(7) not null,
       constraint ct_pessoas_ligadas_pk primary key(cod_pes, cod_pes_anterior))

create table tb_fronteira(cod_pes char(7) not null,
       constraint ct_fronteira_pk primary key(cod_pes,cod_pes_anterior)
)

create table tb_adjacentes_temp(cod_pes char(7) not null,
                                cod_pes_anterior char(7) not null,
       constraint ct_adjacentes_temp_pk primary key(cod_pes,cod_pes_anterior)
)
```

b) Especificações da stored procedure

Para uma observação do funcionamento de “stored procedure”, vamos utilizar o script apresentado na aula sobre “SQL Batch”. Use todo o código do exemplo daquele roteiro com poucas alterações para compor o corpo de uma procedure com as seguintes características:

```
Nome: pr_obtem_amigos_diretos_indiretos.
Parâmetros de entrada (vide mais detalhes no próximo item):
      @cod_pes:
                A variável de mesmo nome declarada no SQL Batch passará a
                ser o parâmetro de entrada.
      @controle_impressao:
                Se o parâmetro vale "N", os dados coletados pela procedure
                não serão apresentados. Neste caso, presume-se que alguma
                atividade será realizada em passos posteriores utilizando
                esses dados.
                Se o parâmetro tem valor "S", serão apresentados
                os dados coletados na tabela tb_pessoas_ligadas.
Objetivo: É o mesmo do SQL Batch com a diferença de que o código foi
          transformado em “stored procedure”, permitindo a passagem de
          parâmetro. Ao final da execução, a tabela tb_pessoas_ligadas
          conterá os dados de todos os amigos diretos e indiretos da pessoa
          cujo código foi recebido como parâmetro de entrada.
          Ordene a apresentação dos resultados pelo nível e código da pessoa.
```

c) Alterações no código fonte apresentado como exemplo no roteiro sobre “SQL Batch”

i) A primeira variável declarada naquele script de nome

```
@cod_pes
```

deve ser excluída da declaração e incluído como o primeiro parâmetro de entrada com o mesmo tipo.

ii) Incluir o segundo parâmetro de entrada:

```
@controle_impressao
```

com o tipo “char(1)”.

Verificar se o valor recebido com parâmetro é válido, ou seja, somente pode ser “S” ou “N”. Se inválido, emite mensagem indicativa e encerra a execução da procedure com o código -99.

iii) Na parte final do código fonte, condicionar a apresentação do conteúdo dos dados coletados somente se o parâmetro de entrada:

```
@controle_impressão
```

for “S”.

d) Teste de execução

O resultado deverá ter um formato análogo ao seguinte aos mostrados abaixo:

Exemplo 1

```
execute pr_obtem_amigos_diretos_indiretos 'P302100','K'

O controle de impressco deve ter valor N ou S.
```

Exemplo 2

```
execute pr_obtem_amigos_diretos_indiretos 'P302100','S'

Encontrar os amigos diretos ou indiretos de
-------------------------------------------
P302100

nivel cod_pes  nome_pes     data_nasc    cod_pes_anterior  nome_pes_anterior
----- -------  -----------  -----------  ----------------  -----------------
    0 P302100  Odilon       Oct 11 2001  0000000           NULL
    1 P302000  Romulo       Oct  1 1999  P302100           Odilon
    2 P300000  Heraldo      Jun 26 2003  P302000           Romulo
    2 P302200  Alessandro   Aug 10 2001  P302000           Romulo
    2 P302300  Suzanne      Jul 28 2003  P302000           Romulo
    3 P301000  Terrence     Apr 29 2002  P300000           Heraida
```

Exemplo 3

```
execute pr_obtem_amigos_diretos_indiretos 'P100000','S'

Encontrar os amigos diretos ou indiretos de
-------------------------------------------
P100000

nivel cod_pes  nome_pes     data_nasc    cod_pes_anterior  nome_pes_anterior
----- -------  -----------  -----------  ----------------  -----------------
    0 P100000  Sandro       Jan  2 2000  0000000           NULL
    1 P101000  Ludmila      Apr 18 2002  P100000           Sandro
    1 P102000  Adriana      Jul 18 2001  P100000           Sandro
    1 P103000  Santoro      Dec  9 2004  P100000           Sandro
    1 P104000  Tobias       Oct 23 1999  P100000           Sandro
    1 P104200  Marco        Feb 18 2002  P100000           Sandro
    2 P101100  Jamil        Apr 20 2004  P101000           Ludmila
    2 P101100  Jamil        Apr 20 2004  P102000           Adriana
    2 P101100  Jamil        Apr 20 2004  P103000           Santoro
    2 P104100  Andressa     Nov  9 2001  P104000           Tobias
    3 P101110  Gian         Mar 25 2003  P101100           Jamil
    3 P101120  Amanda       Oct 15 2000  P101100           Jamil
    3 P101130  Fabiola      Nov 27 2000  P101100           Jamil
    4 P101111  Cassia       Apr 24 2002  P101110           Gian
```

Exemplo 4

```
execute pr_obtem_amigos_diretos_indiretos 'P701131','S'

Encontrar os amigos diretos ou indiretos de
-------------------------------------------
P701131

nivel cod_pes  nome_pes     data_nasc    cod_pes_anterior  nome_pes_anterior
----- -------  -----------  -----------  ----------------  -----------------
    0 P701131 Goulart       Jul 11 2002  0000000           NULL
    1 P701000 Renato        Oct 21 2003  P701131           Goulart
    1 P701130 Ruth          May 11 2004  P701131           Goulart
    2 P700000 Karina        Sep 30 2003  P701000           Renato
    2 P701100 Helena        Feb 21 2002  P701000           Renato
    2 P701100 Helena        Feb 21 2002  P701130           Ruth
    2 P701132 Tulio         Jul  7 1999  P701130           Ruth
    2 P701133 Jailson       Jun 23 2001  P701130           Ruth
    2 P701134 Jefferson     Mar 20 2002  P701130           Ruth
    2 P701200 Tania         Jul  2 2000  P701000           Renato
    2 P701300 Vanda         Aug  5 2005  P701000           Renato
    2 P701400 Lucilene      Jun 22 1999  P701000           Renato
    2 P701500 Liliane       Jun 19 2000  P701000           Renato
    2 P701600 Milena        Jun 11 2003  P701000           Renato
    3 P701110 Agnes         May 28 2000  P701100           Helena
    3 P701120 Francisca     Jun 15 2001  P701100           Helena
    3 P701140 Turmalino     May 24 2000  P701100           Helena
    3 P701150 Izabel        Dec 26 2000  P701100           Helena
    3 P701154 Reinaldo      Aug 21 2003  P700000           Karina
    3 P702000 Sueli         Aug 26 2004  P700000           Karina
    3 P703000 Geisly        Dec 21 2001  P700000           Karina
    3 P704000 Nataly        Jun  5 2003  P700000           Karina
    3 P705000 Lorenzo       Mar  2 2002  P700000           Karina
    4 P701111 Thalita       Oct 28 1998  P701110           Agnes
    4 P701112 Gustavo       Oct 22 2003  P701110           Agnes
    4 P701113 Fabiano       Oct 28 2004  P701110           Agnes
    4 P701114 Ciomara       May 14 2002  P701110           Agnes
    4 P701121 Luiz          Feb  8 2002  P701120           Francisca
    4 P701122 Cintia        Jul 21 2000  P701120           Francisca
    4 P701123 Milton        May 12 2000  P701120           Francisca
    4 P701124 Marilisa      Nov 20 2000  P701120           Francisca
    4 P701141 Samantha      Oct 30 2001  P701140           Turmalino
    4 P701142 Frederico     Dec 11 2002  P701140           Turmalino
    4 P701143 João          Aug  4 2004  P701140           Turmalino
    4 P701144 Raul          Apr 21 2000  P701140           Turmalino
    4 P701151 Claudio       Dec 18 1999  P701150           Izabel
    4 P701152 Fabrizio      May 26 2002  P701150           Izabel
    4 P701153 Leo           Oct  9 2002  P701150           Izabel
    4 P704010 Cleonice      Apr  7 2000  P704000           Nataly
    4 P705100 Amelia        Mar  4 2002  P705000           Lorenzo
```

e) Salve o script de criação da sua procedure no arquivo de resposta.

Observação: Como já percebemos na solução deste problema na aula sobre “SQL Batch”, em termos de grafos, estamos procurando os vértices de subgrafos conexos maximais. Neste sentido, a execução desta “procedure” passando como parâmetro qualquer vértice de um mesmo subgrafo conexo resulta no mesmo conjunto de vértices; as diferenças estão nos níveis, ou seja, nas distâncias relativas, e portanto nas distâncias médias na forma como as definimos.

8.2. Stored procedure para verificar pertinência ao conjunto de amigos diretos e indiretos

Crie uma procedure com as seguintes características:

```
Nome: pr_consulta_amigo_direto_indireto
Parâmetros de entrada:
     @cod_pes:      A pessoa de quem se deseja procurar amigo.
     @cod_pes_amg:  Cod_pes da pessoa que deve ser verificada se pertence
                    ao grupo de amigos diretos ou indiretos da pessoa
                    representada pelo primeiro parâmetro.
                    Os tipos dos dois parâmetros devem ser iguais ao
                    Utilizado na procedure anterior.
Objetivo: Verifica se a pessoa cujo código é recebido no primeiro
          parâmetro tem entre os seus amigos a segunda pessoa.
```

A “stored procedure” deve realizar os seguintes passos:

a) Executar a “stored procedure”:

   ```
   execute pr_obtem_amigos_diretos_indiretos
   ```

   com os parâmetros:

   ```
   Primeiro parâmetro: @cod_pes
   Segundo parâmetro: N.
   ```

b) A execução do passo anterior disponibilizará os dados dos amigos diretos e indiretos. Consultar o usuário

   ```
   @cod_pes_amg
   ```

   para verificar se está na lista obtida no item anterior de forma adequada para que o resultado seja apresentado no modelo proposto.

c) Tenha em mente que a procedure do item (a) pode incluir várias vezes a mesma pessoa no círculo direto e indireto de amigos porque ele pode chegar por caminhos diferentes e, sempre que isto acontece, será com a mesma distância. Se você utilizar um “sub-select” escalar, esteja atento à possibilidade de isto acontecer e, neste caso, terá de incluir a cláusula “distinct”.

Os resultados devem ser apresentados no formato análogo aos seguintes. Pelos dois primeiros exemplos, observamos que o indivíduo P705000 está entre os amigos diretos e indiretos de P701131 à distância de três arestas. Por outro lado, P101130 não faz parte do círculo de amigos diretos e indiretos de P701131.

Exemplo 1

```
execute pr_consulta_amigo_direto_indireto 'P701131', 'P705000'

cod_pes nome_pes                           data_nasc   cod_pes_amg nivel
------- ---------------------------------- ----------- ----------- -----
P701131 Goulart                            Jul 11 2002 P705000         3
```

Exemplo 2

```
execute pr_consulta_amigo_direto_indireto 'P701131', 'P101130'

cod_pes nome_pes                           data_nasc   cod_pes_amg nivel
------- ---------------------------------- ----------- ----------- -----
P701131 Goulart                            Jul 11 2002 P101130     NULL
```

Exemplo 3

```
execute pr_consulta_amigo_direto_indireto 'P101120', 'P102000'

cod_pes nome_pes                           data_nasc   cod_pes_amg nivel
------- ---------------------------------- ----------- ----------- -----
P101120 Amanda                             Oct 15 2000 P102000         2
```

Exemplo 4

```
execute pr_consulta_amigo_direto_indireto 'P101120', 'P302000'

cod_pes nome_pes                           data_nasc   cod_pes_amg nivel
------- ---------------------------------- ----------- ----------- -----
P101120 Amanda                             Oct 15 2000 P302000     NULL
```

Acrescente o código fonte da sua procedure ao arquivo de resposta.

8.3. Stored Procedure para cálculo da distância média

Crie uma stored procedure com as seguintes características:

```
Nome: pr_calcula_distancia_media
Parâmetros de entrada:
     @cod_pes:      A pessoa de quem se deseja obter a distância média dos
                    seus amigos diretos e indiretos. O tipo é o mesmo da
                    procedure anterior.
Objetivo: Obter a distância média dos amigos diretos e indiretos segundo
          a definição apresentada.
```

A “stored procedure” deve realizar os seguintes passos:

a) Verificar se o código da pessoa informada no parâmetro da “procedure” é válido, ou seja, se a pessoa está cadastrada no Banco de Dados;

   i. Utilize o teste de relação vazia (cláusula exists/not exists) sobre a tabela

      `tb_pessoas_teste`

para verificar se o código é válido.

ii. Se inválido:

Apresenta a mensagem: Codigo de pessoa não cadastrada
Retorna da “procedure” com o código -99.

iii. Se válido:

Prossegue com os passos seguintes.

b) Executar a procedure

```
execute pr_obtem_amigos_diretos_indiretos,
```

com os parâmetros:

```
Primeiro parâmetro: @cod_pes
Segundo parâmetro: N.
```

c) A execução do passo anterior disponibiliza os dados dos amigos diretos e indiretos que devem ser utilizados para o cálculo da distância média segundo a definição constante deste roteiro por um dos métodos de cálculo apresentado no texto.

Sugestão: Utilize o recurso de tabela derivada resultante de um dos seguintes comandos para obter a distância média de acordo com o método e sua escolha.

```
select distinct nivel,
                cod_pes
           from tb_pessoas_ligadas
```

```
select nivel,
       count(distinct cod_pes) as qtd
  from tb_pessoas_ligadas
group by nivel
```

d) Na fórmula de cálculo da distância média, os operandos da divisão são números inteiros o que implica num resultado da forma $\lfloor n/m \rfloor$, mas não é isto que desejamos. Portanto, utilize o “casting” na forma explicada neste roteiro.

e) Apresente o valor da distância média com três posições após a vírgula. Faça conversão adequada para o tipo numeric(6,3).

f) Fazer testes de execução para os seguintes parâmetros de entrada:

```
execute pr_calcula_distancia_media 'P000000' /* vértice inexistente */
execute pr_calcula_distancia_media 'P701160' /* vértice solitário */
execute pr_calcula_distancia_media 'P701170' /* vértice solitário */
execute pr_calcula_distancia_media 'P100000'
execute pr_calcula_distancia_media 'P701131'
```

```
execute pr_calcula_distancia_media 'P104100'
execute pr_calcula_distancia_media 'P700000'
```

Os resultados devem ser apresentados num formato análogo aos seguintes:

Exemplo 1

```
execute pr_calcula_distancia_media 'P000000'

Codigo de pessoa nco cadastrada.
```

Exemplo 2

```
execute pr_calcula_distancia_media 'P701160'

cod_pes nome_pes                           data_nasc        dist_media
------- ---------------------------------- ---------------- ----------
P701160 Judy                               Oct 18 2000      0.000
```

Exemplo 3

```
execute pr_calcula_distancia_media 'P701170'

cod_pes nome_pes                           data_nasc        dist_media
------- ---------------------------------- ---------------- ----------
P701170 Hillary                            Oct 12 1998      0.000
```

Exemplo 4

```
execute pr_calcula_distancia_media 'P100000'

cod_pes nome_pes                           data_nasc        dist_media
------- ---------------------------------- ---------------- ----------
P100000 Sandro                             Jan  2 2000      1.833
```

Exemplo 5

```
execute pr_calcula_distancia_media 'P701131'

cod_pes nome_pes                           data_nasc        dist_media
------- ---------------------------------- ---------------- ----------
P701131 Goulart                            Jul 11 2002      3.000
```

Exemplo 6

```
execute pr_calcula_distancia_media 'P104100'
```

```
cod_pes nome_pes                                 data_nasc        dist_media
------- ---------------------------------------- ---------------- ----------
P104100 Andressa                                 Nov  9 2001      3.250
```

Exemplo 7

```
execute pr_calcula_distancia_media 'P700000'

cod_pes nome_pes                                 data_nasc        dist_media
------- ---------------------------------------- ---------------- ----------
P700000 Karina                                   Sep 30 2003      2.743
```

Acrescente o código fonte da sua procedure ao arquivo de resposta.

8.4. Stored Procedure para obter dados dos acidentes de um motorista

Crie uma procedure com as seguintes características:

```
Nome: pr_lista_resumo_acidentes
Parâmetro de entrada: @cod_pes
          Valores possíveis: NULL – apresenta dados de todas as pessoas
                             Código válido – apresenta somente os dados da pessoa
                             Código inválido – apresenta mensagem de código inválido
                                               e retorna com o código de retorno -99.
                             A coluna cod_pes na tabela tb_pessoas está definida
                             com o tipo “char(15)”.
Objetivo: Apresenta a quantidade dos acidentes, valor máximo e mínimo dos danos
          nos acidentes, a data do acidente mais antigo e do mais recente, a soma
          dos danos e o valor médio considerando todos os acidentes em que era o
          condutor do veículo.
```

Teste a execução com os seguintes exemplos. Aproveite a oportunidade para refletir sobre a diferença entre os valores NULL e 'NULL'.

```
execute pr_lista_resumo_acidentes 'NULL'
execute pr_lista_resumo_acidentes NULL
execute pr_lista_resumo_acidentes 'a00003189'
execute pr_lista_resumo_acidentes 'a95003990'
execute pr_lista_resumo_acidentes 'a96003325'
execute pr_lista_resumo_acidentes 'a99004501'
```

O resultado deve ser apresentado no formato análogo ao seguinte. O que segue é uma ilustração para sua conferência.

```
cod_pes   Quant maximo  minimo último acidente     primeiro acidente   total   vlr médio
--------- ----- ------- ------ ------------------- ------------------- ------- ---------
a00003189     1  118.00 118.00 Aug 23 2001 12:00AM Aug 23 2001 12:00AM  118.00 118.00
a95003990     1  176.00 176.00 Nov  1 2005 12:00AM Nov  1 2005 12:00AM  176.00 176.00
a96003325     1  696.00 696.00 Apr 12 2000 12:00AM Apr 12 2000 12:00AM  696.00 696.00
a99004501     4  963.00 286.00 Apr 20 2005 12:00AM Jan  5 2002 12:00AM 2510.00 627.50
```

Acrescente o código fonte da sua procedure ao arquivo de resposta.

8.5. Stored Procedure para obtenção de dados sobre acidentes por motorista em relação à média de todos os acidentes

Crie uma procedure com as seguintes características:

```
Nome: pr_lista_estatísticas_acidentes
Parâmetro de entrada: @cod_pes
          Valores possíveis: NULL – apresenta dados de todos os motoristas
                             Código válido – apresenta somente os dados da pessoa
                             Código inválido – apresenta mensagem de código inválido.
                             Na tabela de acidentes o tipo do código da pessoa
                             é char(15).
Objetivo: Apresenta a soma do valor dos danos de cada motorista seguido da proporção
          que tal soma representa em relação à média dos danos de todos os
          acidentes. Este número deve ser apresentado com 5 algarismos após a
          vírgula e a ordenação dos resultados deve ser feita pelo cod_pes.
```

Observação: Use o “casting” para converter o resultado num dado do tipo “numeric(8,5)”.

O resultado deve ter o seguinte formato com mais de 1500 linhas quando o parâmetro for NULL. Os nomes estão truncados para que os dados de cada motorista caibam numa linha nesta apresentação.

```
execute pr_lista_estatisticas_acidentes 'NULL'
execute pr_lista_estatisticas_acidentes NULL
```

```
cod_pes    nome                         endereco               soma        Proporção
---------- ---------------------------- ---------------------- -------     ---------
a00003160  Adalberto                    Rua A,20, Santos        153.00     0.25677
a00003161  Adriano                      Rua EC,400, Atibaia     654.00     1.09758
a00003163  Alexandre Maciel Gomes       Rua AB,300, Tokyo       812.00     1.36274
a00003164  Aline Silva de Castro        Rua B,30, Sao Paulo    1563.00     2.62312
a00003172  Bruno Cunha Nagalli Ramia    Rua AB,300, Tokyo       197.00     0.33061
a00003174  Bruno Salcas Hilario         Rua E,60, Diadema       947.00     1.58931
a00003175  Caio Jorge Ruman             Rua B,30, Sao Paulo    1142.00     1.91657
.
.
.
```

```
execute pr_lista_estatisticas_acidentes 'a00003174' -- vide resultado acima
execute pr_lista_estatisticas_acidentes 'a00003163' -- vide resultado acima
```

Acrescente o código fonte da sua procedure ao arquivo de resposta e envie-o ao servidor.

Prof. Satoshi Nagayama