POLITÉCNICO DE LEIRIA
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

BASES DE DADOS

Engenharia Informática regimes Diurno e Pós-Laboral

2º Ano 1.º Semestre **2018/19**

Ficha 8

Restrições de integridade e programação em PL/SQL

Objetivos:

- Implementar restrições de integridade de dados em bases de dados.
- Criar vistas

Nota: os exercícios assinalados com (*) devem ser resolvidos em horas de estudo autónomo.

GRUPO I – Vistas

SINTAXE de criação de vistas

```
CREATE VIEW [ schema. ] view_name
  [column_list]
AS
  SELECT command
;
```

1. Crie a vista `V_DOUTORADOS` que calcule o total de docentes doutorados em cada instituição de ensino.

- **Para testar** utilize a vista `V_DOUTORADOS` para apresentar o número total de doutorados do IPLeiria.

2.

2.1. **(*)** Crie a vista `V_DADOS_DOCENTES` que permita obter para todos os docentes, o seu nome, o nome da sua instituição e os seus números de telefone.

2.2. **(*)** Altere a vista anterior de forma a assinalar a linha correspondente ao telefone mais usado para cada docente com um asterisco (*).

2.3. **(*)** Altere a vista anterior de forma a apresentar também os professores que não tenham telefone associado.

3. **(*)** Crie a vista `V_ALUNOS_BENEF` que determine a quantidade de alunos de cada instituição de ensino do IPLeiria que usufruem de algum tipo de benefício.

GRUPO II – Restrições de Integridade

4. Determine todas as restrições declarativas, ou seja, restrições implementáveis por comandos DDL necessárias à manutenção de integridade dos dados na base de dados.

5. Implemente as restrições encontradas na pergunta anterior para a tabela `T_ALUNO`.

6. Determine todas as restrições **não** implementáveis por comandos DDL (não declarativas) necessárias à manutenção de integridade dos dados na base de dados.

GRUPO III – Blocos PL/SQL

SINTAXE de um bloco de código PL/SQL

```
[DECLARE
      -- parte reservada à declaração de variáveis
]
BEGIN
      -- parte reservada ao código SQL e PL/SQL
      -- Comentário de uma só linha
      /* Comentário que pode utilizar
           várias linhas */
[EXCEPTION
      /* parte reservada ao tratamento de excepções */
]
END;
/
```

IMPORTANTE: No final de cada bloco PL/SQL coloque sempre uma linha com o caracter / (barra) para indicar o fim desse bloco.

7. **(*)** Crie um bloco anónimo PL/SQL que apresente no ecrã "Olá Mundo PL/SQL".

8. **(*)** Crie um bloco anónimo PL/SQL que, para uma determinada instituição, escreva no ecrã o n.º de pessoas afectas a essa instituição.

9. **(*)** Altere o código anterior de forma a que indique se o nº de pessoas afetas à instituição é inferior, igual ou superior ao n.º médio de pessoas. Apresente também este valor médio arredondado a duas casas decimais.

10.

10.1. **(*)** Utilizando consultas SQL obtenha a soma dos salários de todos os docentes do IPLeiria e também o salário e grau de cada docente. Guarde estes resultados para posterior comparação.

10.2. **(*)** Crie um bloco PL/SQL que aumente os salários dos docentes tendo em conta os seguintes critérios:

- Se a soma total dos salários pagos pelo instituto for inferior a um determinado valor de referência, o aumento salarial será de 5%, apenas para docentes com o grau de *Licenciado*;
- Nos restantes casos, o aumento será de 10% para os docentes com os graus de *Licenciado* ou *Mestre*.

10.3. **(*)** Teste o bloco criado. Após a execução, repita a alínea **4.1** para comparar os valores obtidos com os antes da execução do bloco.

GRUPO IV – Sequências e Triggers

SINTAXE de criação de sequências

```
CREATE SEQUENCE [ schema. ] sequence_name
   [ { INCREMENT BY | START WITH } integer
   | { MAXVALUE integer | NOMAXVALUE }
   | { MINVALUE integer | NOMINVALUE }
   | { CYCLE | NOCYCLE }
   | { CACHE integer | NOCACHE }
   | { ORDER | NOORDER }
   ]...
;
```

SINTAXE de criação de *triggers*

```
CREATE [OR REPLACE] TRIGGER trigger_name
  {BEFORE | AFTER}
  {INSERT [OR]|
   UPDATE [OF (coluna1, [coluna2,...])] [OR] |
   DELETE [OR]}
  ON nome_tabela
  [FOR EACH ROW]
  [WHEN (condição)]
[DECLARE
..-- parte reservada à declaração de variáveis]
BEGIN
  -- parte reservada ao código SQL e PL/SQL
[EXCEPTION
..-- parte reservada ao tratamento de excepções]
END [trigger_name];
/
```

11. Crie o código PL/SQL necessário para impor as restrições não declarativas:

- de chave primária da tabela `T_INSTITUICAO`, tendo em conta que existem sempre 3 instituições inseridas na primeira utilização da base de dados (ver script `EIBD1819_ficha8_dados.sql`);
- da entidade Instituição, nomeadamente dos atributos deEnsino, numDocs e numAlunos.

Para testar insira as seguintes instituições na base de dados:

- Escola Superior de Saúde (instituição de ensino)
- Escola Superior de Turismo e Tecnologia do Mar (instituição de ensino)
- Unidade de Ensino a Distância (não é instituição de ensino)

12. Altere a tabela `T_PESSOA`, de forma a armazenar a data em que é inserida uma pessoa na base de dados (`dataRegisto`). a adicionar o atributo. Crie também o código PL/SQL necessário para garantir que a data de registo é inferior ou igual à data do sistema, senão deve ser emitida a mensagem de erro *"Data inválida: a data de registo deve ser inferior ou igual à data actual"*.

13. Crie o código PL/SQL necessário para garantir a integridade do atributo calculado `numDocs` da tabela `T_INSTITUICAO`.

Para testar realize as seguintes operações na base de dados:

- Selecionar o nome e o número de docentes das instituições e observe o número de docentes que existem na ESTG e na ESTM
- Inserir o docente Carlos Matos com NIF 555555555 que pertence à instituição ESTG
- Inserir o docente Ana Martins com NIF 444444444 que pertence à instituição ESTG
- Inserir o docente Joaquim Pereira com NIF 333333333 que pertence à instituição ESTM
- Selecionar o nome e o número de docentes das instituições e observe o número de docentes que existem na ESTG e na ESTM
- Mudar o docente Joaquim Pereira para a instituição ESTG
- Eliminar a docente Ana Martins
- Selecionar o nome e o número de docentes das instituições e observe o número de docentes que existem na ESTG e na ESTM

14. **(*)** Crie o código PL/SQL necessário para garantir a integridade do atributo calculado `numAlunos` da tabela `T_INSTITUICAO`.

Para testar realize as seguintes operações na base de dados:

- Observar o número de alunos nas instituições
- Inserir o aluno Filipe Raia com NIF 123456789 à instituição ESTG
- Inserir a aluna Joana Marques com NIF 98764531 à instituição ESTM
- Mudar a Joana Marques para a ESTG
- Observar o número de alunos nas instituições
- Eliminar a Joana Marques
- Voltar a observar o número de alunos nas instituições e verificar que a atualização está correta

15. Observe o código da função `FUNC_TIPO_INST` fornecido no ficheiro EIBD1819_Ficha8_função.txt. Execute-o na base de dados e altere os scripts correspondentes.

```
CREATE OR REPLACE FUNCTION func_tipo_inst
      (pCodI t_instituicao.id%TYPE)
RETURN INTEGER
IS
   resultado t_instituicao.deEnsino%TYPE;
BEGIN
-- verifica se instituição recebida por parâmetro é ou não
-- uma instituição de Ensino
      SELECT deEnsino
        INTO resultado
      FROM t_instituicao
      WHERE id=pCodI;

      IF UPPER(resultado)='S' THEN
         RETURN 1;
      ELSE
         RETURN 0;
      END IF;
EXCEPTION
     WHEN NO_DATA_FOUND THEN
         RETURN -1;
END;
/
```

A função `FUNC_TIPO_INST` permite determinar o tipo de uma determinada instituição, devolvendo:

- o valor -1, se for chamada para uma instituição inexistente;
- o valor 1, se a instituição for de ensino;
- o valor 0, em caso contrário.

16. Crie o código PL/SQL necessário para que só seja possível associar pessoas a instituições de ensino, tal como definido no Diagrama de Entidade-Relacionamento. Nota: utilize a função `FUNC_TIPO_INST`.

Para testar realize as seguintes operações na base de dados:

- Tente inserir os dados da nova aluna Ana Valente, do 1º ano do curso de EI, na Unidade de Ensino a Distância (que não é instituição de ensino) .
- Tente inserir novamente os dados da aluna Ana Valente, do 1º ano do curso de EI, na Escola Superior de Saúde (é uma instituição de ensino).

17. **(*)** Crie o código PL/SQL necessário para garantir que o tipo de pessoa (coluna `T_PESSOA.tipo`) não é alterável.

Para testar realize as seguintes operações na base de dados:

- Tente atualizar o tipo de *Ana Valente* mudando-o de 'A' para 'D' na coluna `tipo` na tabela `T_PESSOA`.

18. Crie o código PL/SQL necessário para garantir que as inserções de alunos e de **(*)** docentes respeitam a disjunção.

Para testar realize as seguintes operações na base de dados:

- Escolha um docente presente na tabela `T_DOCENTE` e depois insira uma linha na tabela `T_ALUNO` utilizando o identificador desse docente.
  - **(*)** Escolha um aluno presente na tabela `T_ALUNO` e depois insira uma linha na tabela `T_DOCENTE` utilizando o identificador desse aluno

19.

19.1. **(*)** Crie o código PL/SQL necessário para garantir a integridade do atributo `telefoneMaisUsado` da tabela `T_PESSOA` aquando da inserção de uma pessoa.

19.2. **(*)** Crie o código PL/SQL necessário para garantir a integridade do atributo `telefoneMaisUsado` da tabela `T_PESSOA` nas restantes situações que esta seja posta em causa.

Para testar realize as seguintes operações na base de dados:

- Inserir o docente Carlos Andrade com NIF 183456789 da instituição ESTG com telefone principal 244877447
- Alterar o telefone do docente Carlos Andrade para 244888447
- Inserir o docente Nuno Lopes com NIF 654897456, telefone principal 91 1234567 e telefones secundários 244123456 e 961234567
- Inserir o aluno Rui Lopes com NIF 554897456, telefone principal 911234568 e telefones secundários 244523456 e 96 8234567
- Atualizar o número de telefone principal do Nuno Lopes para 915234568
- Atualizar o número de telefone principal do Rui Lopes para 244523456

GRUPO V – Procedimentos e Funções

SINTAXE de criação de procedimentos

```
CREATE [ OR REPLACE ]
PROCEDURE [ schema. ] procedure_name
   [ ( parameter_declaration_list ) ]
{ IS | AS }
   [ declare_section ]
BEGIN
  -- parte reservada ao código SQL e PL/SQL
[EXCEPTION
..-- parte reservada ao tratamento de excepções]
END [procedure_name];
/
```

SINTAXE de criação de funções

```
CREATE [OR REPLACE] FUNCTION function_name
   [ ( parameter_declaration_list ) ]
RETURN datatype
{ IS | AS }
   [ declare_section ]
BEGIN
  -- parte reservada ao código SQL e PL/SQL
[EXCEPTION
..-- parte reservada ao tratamento de excepções]
END [function_name];
/
```

IMPORTANTE: Para chamar um procedimento fora de um bloco PL/SQL use uma das funções `CALL` ou `EXECUTE` seguida do nome do procedimento e os parâmetros entre parêntesis separados por vírgulas.

20. Crie o código PL/SQL necessário para garantir que as inserções de docentes e de **(*)** alunos respeitam a participação obrigatória da disjunção.

Para testar realize as seguintes operações na base de dados:

- Inserir o docente Carlos Andrade com NIF 183456789 da instituição ESTG com telefone principal 244877447;
- Inserir a docente Rute Francisco na ESTG, com NIF 888888888, e com telefone principal 244555447 da área de Matemática com grau doutoramento e salário de 1500 euros;
- Inserir o docente Márcio Lopes na ESTG, com NIF 7777777 e telefone principal 244877447 da área de Informática com Mestrado e salário de 1500 euros;
- **(*)** Inserir o aluno Jorge Marques na ESTG com NIF 877777777 e telefone principal 91555887 do 1º ano do curso de EI.

21. Altere a função `FUNC_TIPO_INST` de forma a que devolva o valor **-2** se a instituição for de ensino mas não tiver docentes.

Para testar realize as seguintes operações na base de dados:

- Chame a função para todas as instituições.

22. Na base de dados existe a tabela `T_DOCS_GRAU` que armazena informação desnormalizada. Mais concretamente, a tabela `T_DOCS_GRAU` guarda o total de docentes por grau académico em cada instituição. Esta informação é atualizada assincronamente a pedido do presidente do Instituto.

22.1 Crie o procedimento `PROC_CALCULA_GRAUS` que, ao ser executado, atualize a tabela referida, tendo em conta que o procedimento deverá poder ser usado para recalcular os dados apenas de uma das instituições, recebendo o código da instituição por parâmetro.

22.2 Teste o procedimento `PROC_CALCULA_GRAUS` para a instituição ESTG.

22.3 Altere o procedimento `PROC_CALCULA_GRAUS` de modo a recalcular os valores de todas as instituições, caso código da instituição passado com o valor `NULL`.

22.4 Teste o procedimento `PROC_CALCULA_GRAUS` para todas as instituições.

22.5 Actualize o procedimento `PROC_CALCULA_GRAUS` de modo a apresentar uma mensagem de erro adequada sempre que seja chamado para uma instituição não existente ou que não seja de ensino. O procedimento deverá ter tratamento adequado a outras situações anómalas que possam ocorrer.

NOTA: use a função `FUNC_TIPO_INST` para verificar se a instituição não existe ou não é de ensino.

22.6 Realize os seguintes testes ao procedimento `PROC_CALCULA_GRAUS`.

- Invocar o procedimento para a instituição com código 30.
- Observar os dados em `T_DOCS_GRAU`.
- Alterar o grau de um docente da instituição ESTG.
- Invocar o procedimento para a instituição ESTG.
- Observar os dados em `T_DOCS_GRAU`.

23. **(*)** Crie o procedimento `PROC_MUDAR_DOCENTE` que possa ser usado para trocar um docente entre instituições de ensino. Os critérios de construção do procedimento devem ser os seguintes:

- caso a nova instituição do docente não exista ou não seja uma instituição de ensino, não deverá ser feita a transferência, alertando o utilizador da BD para o problema (pode usar a função `FUNC_TIPOINST` já criada);
- no caso de a transferência ter sucesso, o procedimento deve atualizar a tabela `T_DOCS_GRAU`;
- em caso de ocorrência de situações com erros desconhecidos, o procedimento não deve provocar alterações no estado da BD.

Para testar realize as seguintes operações na base de dados:

- Inserir o docente Ana Silva com NIF 342545679 e telefone 244 844223 na instituição ESTG (área Informática, grau licenciado, salário 1500);
- Inserir o docente Pedro Lopes com NIF 342598584 e telefone 91 333444 na Instituição ESTM (área matemática, grau licenciado, salário 1500);
- Invocar o procedimento para trocar Ana Silva da instituição ESTG para a instituição SAS- IPLeiria;
- Invocar o procedimento para trocar Ana Silva da instituição ESTG para a instituição ESTM.

24. **(*)** Crie o procedimento `PROC_LISTAGEM` que faça uma listagem com a informação de todas as pessoas de determinada instituição recebida por parâmetro. Esta listagem apresenta o nome da pessoa e o tipo. No caso dos docentes deve apresentar a quantidade de telefones que tem associados. No caso dos alunos deve apresentar o ano do curso. A listagem deve ser similar à apresentada abaixo,

neste caso para a ESTG. A listagem deve apresentar primeiro os estudantes e depois os docentes, ambos por ordem alfabética.

```
LISTAGEM DE PESSSOAS da INSTITUIÇÃO ESTG
Ana Silva – Estudante – Ano 2
(...)
Manuel Morais – Docente – 3 telefones
(...)
```

GRUPO VI – Scripts

25. Teste a estrutura de *scripts* criada, atualizando e executando o *script* `EIBD1819_ficha8.sql`.