

Ficha 8 (parte I) Restrições de integridade e programação em PL/SQL

Objetivos:

- Implementar restrições de integridade em bases de dados recorrendo às linguagens SQL e/ou PLSQL.
- Compreender as situações em que pode desnormalizar-se uma base de dados a par com as consequências dessa ação.
- Garantir a consistência de dados numa base de dados desnormalizada.

Notas:

- Na resolução desta ficha considere o caso de estudo fornecido em anexo;
- Os exercícios assinalados com (*) devem ser resolvidos em horas de estudo autónomo;
- **Garanta que os modelos e a implementação da base de dados respeitam cada um dos requisitos apresentados. Cada requisito é identificado de forma única pela letra R seguida de um número (R1, R2, etc.);**
- Sugere-se que a implementação de cada requisito e dos respetivos testes sejam adicionados no local reservado para o efeito.

Vistas e funções em PL/SQL

R1	Descrição
	Pretende-se que, a partir da conta PRESIDENTE, seja possível consultar a informação relativa aos campos calculados <code>total_doutorados</code> e (*) <code>total_alunos_benef.</code>
	Implementação
	Testes
	<p>Antes de realizar os testes execute o seguinte código na conta AULAS:</p> <pre>GRANT SELECT ON v_total_doutorados TO presidente;</pre>

R2	Descrição
	Pretende-se que, a partir da conta PRESIDENTE, seja possível consultar os dados de cada docente, nomeadamente o seu nome, o nome da instituição onde trabalha e todos os seus números de telefone (caso o docente não possua n.º de telefone, apresente, em substituição do n.º, a expressão "<sem contacto>").
	Implementação
	Testes
	<p>Antes de realizar os testes execute o seguinte código na conta AULAS:</p> <pre>GRANT SELECT ON v_docentes TO presidente;</pre>

R3	Descrição
	<p>Implementar uma função para obter o nome completo de cada pessoa, podendo ser essa funcionalidade integrada em pesquisas.</p> <p>Deverá ser possível, aquando da chamada da função, escolher o formato de apresentação: <code><primeiro_nome último_nome></code> (ex: "Charles Babbage") ou <code><último_nome, primeiro_nome></code> (ex: "Babbage, Charles").</p>
	Implementação
	Testes
	<i>Teste a função de duas formas diferentes: por comando SELECT e por bloco de PL/SQL anónimo (ver anexo).</i>

R4	Descrição
	Implementar uma função para fácil acesso ao tipo de determinada pessoa, podendo ser essa funcionalidade integrada em pesquisas.
	Implementação
	(teste a função de duas formas diferentes: por comando <i>SELECT</i> e por bloco <i>PL/SQL</i> anónimo.)

R5	Descrição															
	Pretende-se que, a partir da conta <i>PRESIDENTE</i> , seja possível consultar o nome completo e tipo de cada pessoa (utilize as funções disponíveis). O nome dos docentes deve ser sempre apresentado no formato <i><último_nome, primeiro_nome></i> e o dos estudantes no formato <i><primeiro_nome último_nome></i> . Por omissão, os estudantes deverão surgir em primeiro lugar.															
	Implementação															
	Testes															
	<p>Antes de realizar os testes execute o seguinte código na conta <i>AULAS</i>:</p> <pre>GRANT SELECT ON v_pessoas TO presidente;</pre> <p>Possível output:</p> <table> <thead> <tr> <th>NOME</th><th>Tipo de pessoa</th></tr> </thead> <tbody> <tr> <td>William "Bill" Henry Gates</td><td>Aluno</td></tr> <tr> <td>Grace Hopper</td><td>Aluno</td></tr> <tr> <td>...</td><td>...</td></tr> <tr> <td>Jobs, Steven Paul</td><td>Docente</td></tr> <tr> <td>"Lovelace", Ada</td><td>Docente</td></tr> <tr> <td>Turing, Alan Mathinson</td><td>Docente</td></tr> <tr> <td>...</td><td>...</td></tr> </tbody> </table>	NOME	Tipo de pessoa	William "Bill" Henry Gates	Aluno	Grace Hopper	Aluno	Jobs, Steven Paul	Docente	"Lovelace", Ada	Docente	Turing, Alan Mathinson	Docente	...
NOME	Tipo de pessoa															
William "Bill" Henry Gates	Aluno															
Grace Hopper	Aluno															
...	...															
Jobs, Steven Paul	Docente															
"Lovelace", Ada	Docente															
Turing, Alan Mathinson	Docente															
...	...															

R6 (*)	Descrição
	Pretende-se que, a partir da conta <i>COORDENADOR</i> , seja possível consultar o nome da instituição onde cada estudante estuda.
	Implementação
	Testes

<div>R7</div> <div>(*)</div>	<div>Descrição</div> <div>Criar uma função que permita o fácil cálculo do n.º de estudante de cada aluno, integrável em pesquisas. O n.º de estudante resulta da concatenação de três valores:<ul style="list-style-type: none">Campus onde estuda o aluno;Últimos 2 dígitos do ano letivo de entrada do aluno;Posição de registo do aluno na base de dados (no ano letivo de entrada) com, no mínimo, 4 algarismos.</div>
	<div>Implementação</div>
	<div>Testes</div>
	<div>Possível output:</div> <div><div>ESTUDANTE</div><div>2180006 (William "Bill" Henry Gates)</div><div>2180008 (Grace Hopper)</div></div>

ORACLE syntax for PL/SQL objects and code

<p style="text-align: center;">Anonymous blocks</p> <pre> [DECLARE <i>variables_declaration</i> BEGIN <i>SQL_and_PLSQL_statements</i>; [EXCEPTION -- <i>exception handling statements</i> WHEN <i>exception_name</i> THEN <i>SQL_and_PLSQL_statements</i> [WHEN <i>exception_name</i> THEN <i>SQL_and_PLSQL_statements</i> ...] END; </pre>	<p style="text-align: center;">Views</p> <pre> CREATE [OR REPLACE] VIEW [<i>schema.</i>]<i>view_name</i> [<i>column_list</i>] AS <i>SELECT_statement</i> ; </pre>
<p style="text-align: center;">Sequences</p> <pre> CREATE SEQUENCE [<i>schema.</i>]<i>sequence_name</i> [{INCREMENT BY START WITH} <i>integer</i> {MAXVALUE <i>integer</i> <u>NOMAXVALUE</u>} {MINVALUE <i>integer</i> <u>NOMINVALUE</u>} {CYCLE <u>NOCYCLE</u>} {CACHE <i>integer</i> NOCACHE} {<u>ORDER</u> NOORDER }]... ; </pre>	<p style="text-align: center;">Triggers</p> <pre> CREATE [OR REPLACE] TRIGGER <i>trigger_name</i> {BEFORE AFTER} {INSERT [OR] UPDATE [OF (<i>column1</i>, [<i>column2</i>, ...])] [OR] DELETE [OR] } ON <i>table_name</i> [FOR EACH ROW] [WHEN (<i>condition</i>)] [DECLARE <i>variables_declaration</i> BEGIN <i>SQL_and_PLSQL_statements</i> [EXCEPTION <i>exception_handling_statements</i> END [<i>trigger_name</i>]; / </pre>
<p style="text-align: center;">Procedures</p> <pre> CREATE [OR REPLACE] PROCEDURE [<i>schema.</i>]<i>procedure_name</i> [(<i>parameter_declaration_list</i>)] {IS AS} [<i>variables_declaration</i>] BEGIN <i>SQL_and_PLSQL_statements</i>; [EXCEPTION <i>exception_handling_statements</i> END [<i>procedure_name</i>]; / </pre>	<p style="text-align: center;">Functions</p> <pre> CREATE [OR REPLACE] FUNCTION <i>function_name</i> [(<i>parameter_declaration_list</i>)] RETURN <i>datatype</i> {IS AS} [<i>variables_declaration</i>] BEGIN <i>SQL_and_PLSQL_statements</i> [EXCEPTION <i>exception_handling_statements</i> END [<i>function_name</i>]; / </pre>

Syntax reading notes:

- Language reserved words have the bold style (ex: **CREATE**)
- Optional parts are enclosed with square brackets (ex: [*function_name*])
- Parts to be replaced have lowercase italic style (ex: *variable_declaration*)
- Sets of options are enclosed with braces (ex: {**BEFORE**|**AFTER**})

Configuring output display with DBMS_OUTPUT.PUT_LINE



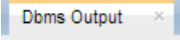

Prior to an effective use of the `DBMS_OUTPUT.PUT_LINE` method in PL/SQL blocks, the following steps are required.

- **In SQL*PLUS:**

Type and run

```
SET SERVEROUTPUT ON
```

- **In SQL Developer:**

- a) Go to the  menu and choose  `Dbms Output` ;
- b) The  tab will open: now press the  button;
- c) Choose the target connection from the dialog box.