

Ficha 8 (parte II) 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 neste grupo. 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.

Restrições de integridade

Garanta que a base de dados respeita cada um dos requisitos apresentados neste grupo. Assim, para cada requisito:

- 1º. Altere em conformidade o Modelo Conceptual fornecido (se aplicável);
- 2º. Altere em conformidade o Modelo Lógico associado ao Modelo Conceptual (se aplicável);
- 3º. Defina e caracterize as restrições de integridade associadas ao requisito (declarativa/não declarativa, necessidade de desnormalização dos modelos, etc.). **Nos requisitos que requerem desnormalização é necessária especial atenção potencial surgimento de incoerências nos dados caso as regras de integridade não sejam totalmente acauteladas.**
- 4º. Utilizando código SQL e PL/SQL, implemente as restrições de integridade associadas ao requisito.
- 5º. **Teste as restrições implementadas inserindo/atualizando/eliminando dados de forma a testar:**
 - a) **As situações que podem gerar quebra de integridade dos dados;**
 - b) **As situações que devem funcionar sem restrições.**

R8 (*)	Descrição
	O nome de cada instituição deverá ser único.
	Desnormalização
	Implementação
	Testes

R9 (*)	Descrição
	A atribuição do identificador da <u>pessoa</u> deverá ser feita a partir da base de dados, sempre de forma automática e sequencial. (*) O utilizador COORDENADOR deverá ser o responsável pela inserção de alunos e de docentes na base de dados.
	Desnormalização
	Implementação
	Sequência + Trigger before insert
	Testes
	Insert into pessoa ...

R10 (*)	Descrição
	O identificador da <u>instituição</u> deve ser gerado automaticamente pela base de dados de forma sequencial.
	Desnormalização
	Implementação
	Testes

R11	Descrição
	Para cada pessoa é necessário conhecer a data em que os seus dados foram <u>inicialmente inseridos</u> (<i>data de registo</i> da pessoa). A data de registo será gerida pela base de dados, pelo que qualquer data fornecida pelos utilizadores deverá ser ignorada.
	Desnormalização
	Adicionar coluna data à tabela pessoas
	Implementação
	Trigger before para colocar o sysdate
	Testes
	Adicionar pessoa com uma data diferente

R12 (*)	Descrição
	Para cada pessoa é necessário conhecer a data em que os seus dados foram <u>alterados</u> .
	Desnormalização
	Implementação
	Testes

R13 (*)	Descrição
	Apenas docentes podem mudar de instituição de ensino, alunos não podem fazê-lo.
	Desnormalização
	Implementação
	Sugestão: utilize a função criada no requisito R4.
	Testes

R14 (*)	Descrição
	Uma vez criada uma pessoa na base de dados, os seus dados não poderão mais ser removidos.
	Desnormalização
	Implementação
	Testes

R15	Descrição
	Por questões de desempenho, após avaliação cuidada, foi decidido facilitar ao máximo a obtenção do <u>tipo</u> de cada pessoa (docente ou aluno) e simultaneamente reduzir o tempo de resposta das pesquisas que precisem dessa informação. Comparar as vantagens e desvantagens desta nova abordagem face à forma usada anteriormente (R4).
	Desnormalização
	Implementação
	Testes

R16	Descrição
	Por questões de desempenho, após avaliação cuidada, foi decidido aplicar desnormalização à hierarquia da entidade Instituição. A remoção da tabela <code>instituição_ensino</code> , integrando-a na tabela <code>instituição</code> , foi o método escolhido. Será necessário garantir que as pessoas apenas são associadas a instituições de ensino, mantendo assim as regras presentes no modelo conceptual.
	Desnormalização
	Implementação
	Testes

R17	Descrição
	Garantir que a disjunção na hierarquia da entidade Pessoa está em todos os instantes garantida.
	Desnormalização
	Implementação
	Testes

R18 (*)	Descrição
	Dada a regular utilização do <u>n.º de estudante</u> (ex.: 2121234) de cada aluno como forma de identificação do mesmo e dada a complexidade do cálculo do mesmo, o n.º de estudante passará a ser armazenado na base de dados. A atribuição do n.º de estudante a cada aluno deve ser realizada no momento do registo do aluno na base de dados.
	Desnormalização
	Implementação
	Sugestão: utilize a função criada no requisito R7.
	Testes

R19 (*)	Descrição
	Cada pessoa poderá (ou não) ter um telefone considerado “preferencial”, de entre os vários telefones que para si estejam registados na base de dados. As pesquisas deverão ser otimizadas para <u>máximo desempenho</u> , aplicando desnormalização se necessário.
	Desnormalização
	Implementação
	Testes

R20	Descrição
	Deverá ser possível consultar o total de docentes de cada instituição com máximo desempenho , aplicando desnormalização se necessário. Nota: os docentes podem mudar de instituição.
	Desnormalização
	Implementação
	Testes

R21 (*)	Descrição
	Deverá ser possível consultar o volume salarial de cada instituição de ensino com máximo desempenho , aplicando desnormalização se necessário.
	Desnormalização
	Implementação
	Testes

R22 (*)	Descrição
	Pretende-se que seja possível consultar o nome da instituição onde cada docente trabalha. As pesquisas deverão ser otimizadas para máximo desempenho , aplicando desnormalização se necessário.
	Desnormalização
	Implementação
	Testes

R23 (*)	Descrição
	Deverá ser possível consultar o total de alunos de cada instituição com máximo desempenho , aplicando desnormalização se necessário. Nota: os alunos não podem mudar de instituição.
	Desnormalização
	Implementação
	Testes

R24	Descrição
	Garantir a integridade dos dados relativamente à participação obrigatória da hierarquia da entidade PESSOA. Assegurar ainda que: <ul style="list-style-type: none"> • a partir da conta PRESIDENTE podem inserir-se docentes; • (*) a partir da conta COORDENADOR podem inserir-se alunos.
	Desnormalização
	Implementação
	Testes
	Antes de realizar os testes execute o seguinte código na conta AULAS: <pre>GRANT EXECUTE ON proc_novo_docente TO presidente;</pre>

R25	Descrição
	<p>A partir da conta PRESIDENTE deverá ser possível apresentar um relatório com a informação de todas as pessoas de determinada instituição (dada pelo nome). Este relatório apresenta o nome da pessoa por tipo.</p> <p>No caso dos docentes, o relatório deve apresentar os seus telefones associados. No caso dos alunos, o relatório deve apresentar o ano escolar.</p> <p>A estrutura do relatório deve ser similar à apresentada abaixo (neste caso para a instituição ESTG). Deverão surgir primeiro os estudantes e depois os docentes, ambos por ordem alfabética.</p>
	<div> <p>RELATÓRIO DE PESSOAS da INSTITUIÇÃO ESTG</p> <p>*** ESTUDANTES ***</p> <p>Ana Silva, 2º ano</p> <p>(...)</p> <p>*** DOCENTES ***</p> <p>Ada "Lovelace" - 244101010, 960019847, 910036475</p> <p>Alan Turing - 244101010</p> <p>(...)</p> </div>
	Desnormalização
	Implementação
	Testes
	<p>Antes de realizar os testes execute o seguinte código na conta AULAS:</p> <pre>GRANT EXECUTE ON proc_report_pessoas TO presidente;</pre>

R26	Descrição
	Na base de dados existe a tabela T_DOCS_GRAU que armazena informação desnormalizada acerca do total de docentes por grau académico em cada instituição. Esta informação é atualizada <u>assincronamente</u> mediante a chamada a um procedimento PL/SQL. Apesar da tabela poder armazenar informação redundante acerca de várias instituições, o utilizador escolhe qual a instituição que pretende atualizar.
	Desnormalização
	Implementação
	Testes
	Teste na conta PRESIDENTE

R27 (*)	Descrição
	O requisito R26 deve ser alterado de forma que o utilizador possa, se assim o entender, atualizar a tabela t_docs_grau para todas as instituições.
	Desnormalização
	Implementação
	Sugestão: utilize o parâmetro já existente
	Testes

ORACLE syntax for PL/SQL objects and code

<p>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>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>Sequences</p> <pre> CREATE SEQUENCE [<i>schema.</i>]<i>sequence_name</i> [{INCREMENT BY START WITH} <i>integer</i> {MAXVALUE <i>integer</i> NOMAXVALUE} {MINVALUE <i>integer</i> NOMINVALUE} {CYCLE NOCYCLE} {CACHE <i>integer</i> NOCACHE} {ORDER NOORDER }] ... ; </pre>	<p>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>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>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



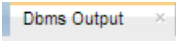

Prior to an effective use of the `DBMS_OUTPUT.PUT_LINE` method in PL/SQL blocks, the following steps are required.

- **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.