



Introdução à Linguagem PL/SQL



que é PL/SQL?



PL/SQL é uma linguagem de programação, desenvolvida pela Oracle, como extensão da linguagem SQL. É uma linguagem procedural, estruturada, de alto desempenho, fácil leitura e totalmente integrada ao SGBDR Oracle. Diferentemente de outras linguagens de programação, como JAVA, Python e C#, PL/SQL não é uma linguagem de uso geral. Ela foi desenvolvida exclusivamente para automação de tarefas em bancos de dados Oracle. Não há compiladores ou interpretadores PL/SQL disponíveis fora do SGBDR Oracle.

Outros SGBDR oferecem suas próprias linguagens de programação integradas, com muitas características similares à PL/SQL. Por exemplo, a linguagem TRANSACT SQL (T-SQL) é a linguagem de programação desenvolvida pela Microsoft para seu SGBDR SQL Server. No entanto, dificilmente códigos em PL/SQL rodarão em outros SGBDR sem que sejam necessárias adaptações.

Como toda linguagem de programação, PL/SQL possui sua própria estrutura e comandos, desenvolvidos especificamente para a automação de tarefas e manipulação de objetos presentes nos esquemas relacionais suportados pelo SGBDR Oracle.

Programas PL/SQL podem ser submetidos ao SGBDR Oracle através de front end interativos/CLI¹ (*SQLPlus*), GUI/IDE² (*SQL Developer*) ou por programas aplicativos através das API (*Application Program Interface*) disponíveis. Tanto o *SQLPlus* quanto o *SQL Developer* estão disponíveis juntamente com o SGBDR Oracle.

Primeiros passos: Preparando o ambiente de desenvolvimento




O SQL Developer é uma interface gráfica que permite utilizar  as funcionalidades do SGBDR Oracle (Figura 1). Esta ferramenta será utilizada na apresentação dos exemplos ao longo do curso.

Figura 1 – SQL Developer

Os principais elementos do SQL Developer são: (1) Conexões, (2) Editor (Worksheet), (3) Resultado de Consultas (Script Output e Query Result) e (4) DBMS Output (esta janela provavelmente não estará visível). Antes de mais nada, é preciso estabelecer uma conexão com o SGBDR Oracle. Se não houver qualquer conexão, clique no sinal de '+' no canto superior esquerdo do painel de conexões e forneça os parâmetros da conexão (Figura 2).

É possível se conectar ou desconectar do servidor clicando com o botão direito sobre o nome da conexão. Uma vez estabelecida a conexão, se tem acesso a todos os objetos do esquema padrão (no SGBDR Oracle, o esquema padrão tem o nome igual ao login do usuário).

Submeta o comando `SELECT 'Hello, world!' FROM DUAL;` ao servidor. Digite o comando no editor e clique na seta verde no canto superior direito do editor. O texto Hello, world! deverá aparecer na aba Query Result. A aba Script Output é utilizada para exibir mensagens resultantes da execução de consultas.

Habilite agora a janela DBMS Output. No menu do programa, clique em View > DBMS Output, depois no '+' e selecione uma conexão ativa. Agora, submeta o comando `EXEC DBMS_OUTPUT.PUT_LINE('Hello, world!');` ao servidor (não se preocupe com o seu significado). A mensagem deverá aparecer na janela DBMS Output.

Figura 2 – Criação de uma nova conexão



No canto superior esquerdo do editor, há dois botões que possuem funções similares: o botão de execução de comando (▶) e o botão de execução de script, que fica à sua direita. O primeiro executa os comandos assinalados no editor (ou o comando sob o cursor, caso não haja nenhum texto assinalado) enquanto que o segundo executa todos os comandos do editor. Outra funcionalidade importante do editor é a exibição de número de linha. Para ativá-la, clique com o botão direito do mouse na faixa vertical à esquerda, bem debaixo do ▶, e selecione Toggle Line Numbers. Todas as mensagens de erro enviadas pelo SGBDR fazem referência ao número da linha no editor. Para tornar esta característica permanente, selecione Tools > Preferences > Code Editor > Line Gutter e marque a opção Show Line Numbers. Outro botão importante, presente em várias janelas, é o apagador (borracha vermelha na ponta do lápis). Ele apaga o conteúdo da respectiva janela. Para desfazer a ação, digite CTRL+Z (undo).

Explorando objetos

Após o estabelecimento de uma conexão, todos os objetos presentes no esquema padrão ficam disponíveis para o usuário. Para ter acesso a eles, clique no '+' à esquerda do nome da conexão ativa. Uma estrutura semelhante a um explorador de arquivos é exibida com um 'diretório' para cada tipo de objeto (Figura 3).

Figura 3 – Explorador de Objetos

Clicando no '+' ao lado de cada tipo de objeto, você consegue ver tudo o que existe no esquema padrão. Por exemplo, no meu esquema padrão foram

criadas as tabelas exibidas na Figura 4.



Figura 4 – Tabelas

Para explorar um determinado objeto, basta clicar sobre seu nome. Uma nova aba, com o mesmo nome da tabela, será criada no editor com diversas subabas exibindo todas as características do objeto selecionado. Por exemplo, ao clicar sobre a tabela EMPLOYEES, a aba mostrada na Figura 4 será criada. Veja que é possível saber tudo a respeito da tabela, inclusive navegar pelo seu conteúdo (subaba Data).

Figura 5 – Janela do explorador de objetos com as características e dados da tabela EMPLOYEE

Estrutura de um programa PL/SQL

Como ocorre em outras linguagens de programação, PL/SQL possui uma estrutura própria, além de comandos e regras sintáticas.

PL/SQL é uma linguagem estruturada em blocos. Um bloco define fronteiras para execução e escopo (visibilidade) para variáveis e tratamento de exceções. Um bloco pode ter 4 seções: cabeçalho (header), declaração (declaration), execução (execution) e tratamento de exceções (exception handling):

Blocos são utilizados para definição de blocos anônimos e programas armazenados (stored programs). Blocos anônimos não possuem cabeçalho e são voláteis, deixando de existir após a sua execução. Já programas armazenados são persistentes (ficam armazenados no banco de dados), possuem cabeçalho e podem ser de três tipos: procedimentos (procedures), funções (functions) e gatilhos (triggers).

Na seção de declaração são declaradas variáveis e outras estruturas de dados que serão utilizados na seção de execução. Toda e qualquer variável utilizada deve ser declarada.

A seção de execução é a única obrigatória. Nela, são colocados os comandos PL/SQL responsáveis por implementar a lógica do bloco. Deve haver pelo menos um comando na seção de execução.

Finalmente, a seção de tratamento de exceções implementa toda a lógica de tratamento de erros de execução e avisos (warnings). Embora esta última seção seja opcional, é importante que o tratamento de erros seja feito dentro do próprio bloco a fim de se evitar a exposição de situações anormais a usuários e programas aplicativos.

O menor bloco em PL/SQL possui apenas a seção de execução. O exemplo utilizado na seção em que foi apresentado o SQL Developer poderia ter sido executado como um bloco anônimo:

Sub-blocos e escopo de variáveis

Um conceito importante, presente associado a linguagens de programação, é o de escopo e visibilidade de variáveis. Blocos declarados dentro de outros blocos são chamados sub-blocos. Sub-blocos podem ser declarados dentro de outros sub-blocos e assim por diante.

Escopo de variáveis em PL/SQL é bastante simples: variáveis e outras estruturas de dados são visíveis dentro do bloco e sub-blocos onde são declaradas, deixando de existir quando o bloco chega ao final. Considere o

exemplo apresentado a seguir. Não se preocupe, por enquanto, com a sintaxe das seções de cabeçalho e declaração:



No exemplo acima, são declarados 3 blocos aninhados. A variável `var1`, declarada no bloco principal (linha 3) e inicializada com 0 (linha 5), é visível no bloco principal e em todos os sub-blocos nele declarados. No sub-bloco 1, são declaradas as variáveis `var2` e `var3` (linhas 8 e 9) e seus valores iniciais são atribuídos nas linhas 11 e 12. Por fim, o sub-bloco 3 é declarado a partir da linha 14. A variável `var2` nele declarado, se sobrepõe à variável de mesmo nome declarada no sub-bloco 1. Desta forma, o valor atribuído à `var2` na linha 17 altera a variável local ao sub-bloco 2. Já a soma envolvendo a variável `var1`, dentro da seção de execução do sub-bloco 2, altera a variável `var1` declarada no bloco principal (não há declaração de variável `var1` no sub-bloco 1). A saída do programa, portanto, é a mostrada à direita do programa. A partir da linha 28, apenas a variável `var1`, declarada no bloco principal, é visível.

Identificadores

Identificadores são nomes dados aos objetos utilizados nos programas PL/SQL: variáveis, constantes, blocos de programa (funções, procedimentos, pacotes, gatilhos etc.), exceções e rótulos. Identificadores podem ter até 30 caracteres, devem começar por uma letra e não podem ter espaços em branco. Os caracteres permitidos em identificadores são: letras (a-z, A-Z), números (0-9), dólar (\$), underscore () e *jogo da velha* (#)³. O compilador PL/SQL não diferencia letras maiúsculas de minúsculas para identificadores: os identificadores `ACT$22`, `Act_$22` e `act_$22` são considerados iguais.

Tipos simples

O tipo determina a faixa de valores que uma variável pode receber. Os tipos mais comuns disponíveis em PL/SQL são:



Constantes e variáveis

Constantes são identificadores cujos valores são definidos no momento de sua declaração e não podem ser alterados. Variáveis, por outro lado, podem ter seus valores alterados em tempo de execução. A declaração de constantes e variáveis possui a mesma forma geral:

O qualificador `CONSTANT` deve ser utilizado na declaração de constantes. Constantes devem ter o valor padrão, definido na cláusula `DEFAULT`, compatível com seu tipo. Qualquer tentativa de atribuição de valores a constantes causará um erro de execução. A cláusula `NOT NULL` impede que o valor `NULL` seja atribuído à variável (ocorre um erro de execução caso isto seja feito).

O exemplo a seguir ilustra a declaração de constantes e variáveis de diversos tipos.

O resultado, após a execução, é:

Observe o valor da variável c. Como o tamanho do tipo CHAR é fixo, valores atribuídos a variáveis deste tipo são completados com brancos à direita caso sejam menores que o tamanho declarado. Já variáveis do tipo VARCHAR2 tem tamanho dinâmico e não recebem brancos adicionais. Por último, observe os valores das variáveis j e k. Ambas representam o mesmo intervalo de tempo, porém em escalas diferentes.

Valores literais

Literais são valores fixos não identificados, ou seja, não associados a qualquer identificador. Literais de texto (strings) devem vir entre apóstrofes ('). Para incluir o apóstrofo em um literal de texto deve-se utilizar dois apóstrofes seguidos. Literais numéricos devem usar o ponto decimal para separar a parte inteira da fracionária. Veja o exemplo a seguir.

Após a execução, o resultado é:

Podem-se alterar, para fins de exibição apenas, os caracteres de separação de grupo e decimal. No SGBDR Oracle, estes caracteres são, por padrão, a vírgula e o ponto. Para trocar os dois, deve-se utilizar o comando ALTER SESSION SET NLS_NUMERIC_CHARACTERS = ','. Independentemente de qualquer coisa, deve utilizar o ponto como separador da parte inteira e decimal de um número. Veja o exemplo a seguir.

O resultado do trecho acima exibirá o valor 1234,56.



Rótulos (Labels)

Rótulos são identificadores utilizados para nomear uma parte específica de um programa. Eles são colocados imediatamente antes do trecho que nomeiam e têm o formato <<identificador>>. Rótulos são utilizados para qualificar variáveis em blocos aninhados e para alterar o fluxo de execução de um programa (comando GOTO). Veja os exemplos a seguir.

O resultado será:

Operadores

Operadores efetuam uma determinada operação com os seus operandos. Os operadores podem ser aritméticos, relacionais, lógicos e de comparação. A Tabela 1 apresenta os principais operadores da linguagem:

Tabela 1 – Principais operadores da linguagem PL/SQL

Os operadores aritméticos, relacionais e lógicos são similares aos de outras linguagens de programação. A precedência é maior para o menos unário, operadores lógicos multiplicativos (*, / e **), operadores lógicos aditivos (+ e

–), operadores relacionais e, por fim, operadores lógicos. Havendo mesma precedência, as expressões são avaliadas da esquerda para a direita. A precedência pode ser alterada com o uso de parênteses.

O operador BETWEEN funciona como um par de operadores relacionais: $x \text{ BETWEEN } a \text{ AND } b$ é equivalente a $(x \geq a) \text{ AND } (x \leq b)$.

O operador LIKE é muito útil em diversas situações. Com ele, é possível procurar por padrões em literais de texto. A busca pode ser explícita ou utilizando-se os caracteres coringa '_' (representa um único caractere na posição indicada) e '%' (representa 0, 1 ou mais caracteres na posição indicada). Veja os exemplos a seguir:

Exemplos de uso do operador LIKE

SQL Developer - Dicas e comandos úteis


O SQL Developer possui inúmeras funcionalidades que facilitam o dia-a-dia do desenvolvedor de aplicações. A seguir, são relacionadas algumas delas.

Dica 1 – Comentários

Durante o desenvolvimento de programas, é comum 'comentarizar' parte do código, ou seja, transformar parte do código em comentário. Quando são algumas poucas linhas, a tarefa é simples, porém transformar uma procedure inteira em comentário nem sempre é agradável. Existem dois tipos de comentário: comentário de linha e comentário de bloco.

Comentários de linha são iniciados por '--' e valem até o final da linha corrente. Comentários em bloco são iniciados por '/*' e *permanecem até que */* seja encontrado. O editor permite fazer isto automaticamente. Basta selecionar o trecho desejado do texto e selecionar Source > Toggle Line Comments. O mesmo deve ser feito para voltar o texto ao normal.

Dica 2 – Indentação de texto

Indentação (neologismo oriundo do inglês indentation) corresponde  aos espaços inseridos no código a fim de ressaltar a estrutura do programa. Como PL/SQL é uma linguagem estruturada em blocos, é comum que cada bloco de código tenha um recuo diferente.

Assim como ocorre com comentários, é possível adicionar/retirar o recuo de um trecho do código selecionado através dos comandos Source > Indent Text / Unindent Text.

Dica 3 – Formatação de texto


Assim como a indentação facilita a leitura de programas, diferenciar identificadores de palavras reservadas também ajuda. O editor utiliza cores diferentes para as palavras reservadas, porém nem sempre isto ajuda. As teclas CTRL+F7 aplicam um padrão de formatação ao texto selecionado, colocando todas as palavras chaves em letras maiúsculas, quebrando linhas no início de comandos e cláusulas SQL e indentando o texto. Faça uma experiência e verifique se o padrão lhe agrada. Os padrões do SQL Developer podem ser alterados em Tools > Preferences > Code Editor > Format e Advanced Format.

Dica 4 – Realce de nulls em tabelas

O valor null tem um significado especial em bancos de dados. É importante que se possa diferenciar campos com este valor e com o literal 'null', por exemplo. Para ver a diferença, execute o comando `SELECT * FROM ALL_TABLES`. Veja que há vários campos com o valor null. Para exibí-los em destaque, vá para Tools > Preferences > Database > Advanced e selecione a cor de fundo de campos com o valor null (Figura 5).

Figura 5 – Realce de campos com null

Dica 5 – Exibindo os resultados de múltiplas consultas

Por padrão, o SQL Developer exibe o resultado de uma consulta na aba  Query Result, normalmente posicionada abaixo do editor. A execução de uma segunda consulta é exibida na mesma aba, sobrescrevendo o resultado da consulta anterior. Para evitar que isto aconteça, fixe a aba utilizando o pequeno alfinete vermelho no canto superior esquerdo da aba. Para tornar este comportamento padrão, selecione Tools > Preferences > Database > Worksheet e marque a opção Show query results in new tabs. É possível fixar quantas abas forem necessárias. É possível também renomear as abas clicando com o botão da direita do mouse sobre a aba e selecionando Rename.

Algumas vezes deseja-se ver o código SQL que gerou o resultado. Pode-se deixar o mouse sobre a aba ou clicar no botão SQL na parte superior.

Dica 6 – Exportação de dados de consultas


A forma mais rápida e simples de exportar o resultado de uma consulta para outro programa é selecionando-se a região desejada e depois CTRL+SHIFT+C. Depois, basta colar o resultado em outro programa (Excel, por exemplo).

Pode-se também utilizar o assistente de exportação (opção Export, clicando com o botão direito do mouse sobre o resultado da consulta). A Figura 6 mostra o assistente de exportação. Com ele, é possível selecionar o formato do arquivo, separadores de linha, terminadores de linha etc. O mesmo pode ser feito para o conteúdo de tabelas.

Figura 6 – Assistente de Exportação

Dica 7 – Atalhos de código

O princípio de Pareto também se aplica à programação: em 80% das vezes, utilizam-se cerca de 20% dos comandos e estruturas disponíveis em uma

linguagem. O SQL Developer possui alguns atalhos de código pré-definidos, além de permitir que o usuário defina seus próprios atalhos. Selecione  > Preferences > Code Editor > Code Templates. São exibidos os atalhos já pré-definidos. No editor, digite, por exemplo, ssf e depois CTRL+Espaço. O atalho ssf foi substituído pelo trecho de código `SELECT * FROM`. Basta agora completar o comando. Você pode criar seus próprios atalhos clicando em Add Template.

Dica 8 – Trechos de código

SQL e PL/SQL são linguagens com muitos comandos com sintaxe específica. Lembrar dos detalhes de cada um, principalmente daqueles menos utilizados, é uma tarefa difícil. View > Snippets exibe a janela de snippets de código. Lá, podem ser encontrados muitos dos comandos e funções disponíveis, organizados por categoria. Para utilizá-los, basta arrastar e soltar o snippet desejado no editor. Alguns, são estruturas de programa completas (tente, por exemplo, CURSOR, na categoria PL/SQL Programming Techniques). É possível também acrescentar snippets à relação existente.

¹Command Line Interface (CLI) são interfaces similares a um terminal de texto, como o CMD do Windows.

²Graphical User Interface (GUI) e Integrated Development Environment (IDE).

³É possível quebrar as regras de formação de identificadores, exceto o limite de 30 caracteres, definindo-os entre aspas: "123 abc !" é um identificador válido em PL/SQL. Pode-se, inclusive, declarar o identificador " " (três espaços em branco). Por razões óbvias, não se recomenda o uso deste tipo de identificador.

Referência Bibliográfica

ELMASRI, R. e NAVATHE, S. B. Sistemas de Banco de Dados. 7ª Ed., São Paulo: Pearson, 2011.

PUGA, S., FRANÇA, E. e GOYA, M. Banco de Dados: Implementação em SQL, PL SQL e Oracle 11g. São Paulo: Pearson, 2014.

Gonçalves, E. PL/SQL: Domine a linguagem do banco de dados Oracle.

Versão Digital. Casa do Código, 2015.



FEUERSTEIN, S. Oracle PL/SQL Programming. 6a Ed., O'Reilly, 2014.

Ir para exercício