
FIAP
FIAP GRADUAÇÃO

FIAP

Tecnologia em Análise e Desenvolvimento de Sistemas

Profº Ms. Alexandre Barcelos
profalexandre.barcelos@fiap.com.br

2019

FIAP

Database Application Development

Profº Ms. Alexandre Barcelos
profalexandre.barcelos@fiap.com.br

2019

Declarando Variávies PL/SQL

Objetivos

FIAP

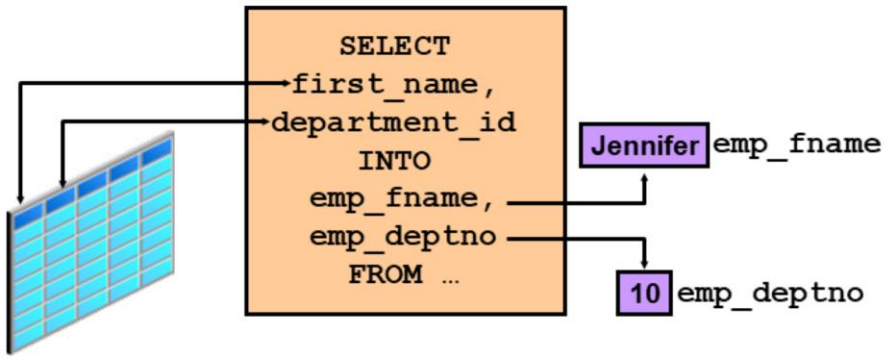
- Identificar os identificadores válidos e inválidos
- Entender a utilização das variáveis
- Declarar e inicializar variáveis
- Listar e descrever vários tipos de dados
- Identificar os benefícios da utilização do atributo%TYPE
- Declarar e utilizar as Binds Variables

Utilização das Variáveis

FIAP

As variáveis podem ser usadas para:

- Armazenamento temporário de dados
- Manipulação de valores armazenados
- Reutilização



1-6

Uso de Variáveis

Com o código PL/SQL, você poderá declarar variáveis e depois usá-las em instruções procedurais e SQL onde uma expressão possa ser usada.

- Armazenamento temporário de dados

Os dados podem ser armazenados temporariamente em uma ou mais variáveis para uso quando na validação da entrada de dados para processamento posterior no processo de fluxo de dados.

- Manipulação de valores armazenados

As variáveis podem ser usadas para cálculo e manipulação de outros dados sem acessar o banco de dados.

- Reutilização

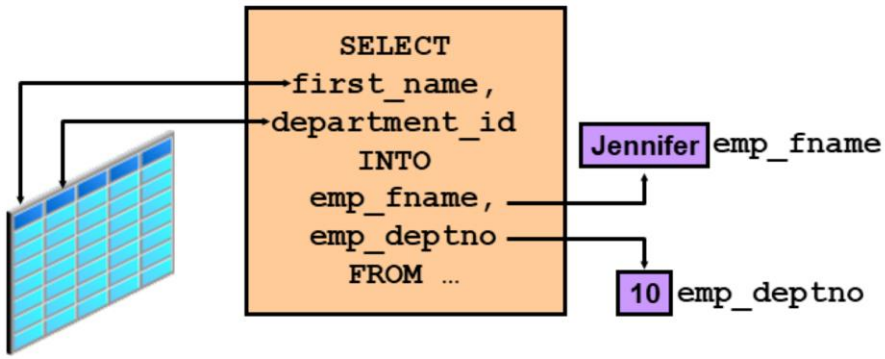
Quando declaradas, as variáveis podem ser usadas repetidamente em uma aplicação simplesmente referenciando-as em outras instruções, incluindo outras instruções declarativas.

Utilização das Variáveis

FIAP

As variáveis podem ser usadas para:

- Armazenamento temporário de dados
- Manipulação de valores armazenados
- Reutilização



1-7

Uso de Variáveis

- De fácil manutenção

Ao usar %TYPE e %ROWTYPE (mais informações sobre %ROWTYPE são abordadas em uma lição subseqüente), você declara variáveis, baseando as declarações nas definições das colunas de banco de dados. As variáveis PL/SQL ou variáveis de cursor anteriormente declaradas no escopo atual poderão usar também os atributos %TYPE e %ROWTYPE como especificadores de tipos de dados. Se uma definição subjacente for alterada, a declaração de variável se altera de acordo durante a execução. Isso permite a independência dos dados, reduz custos de manutenção e permite que os programas se adaptem, conforme o banco de dados for alterado, para atender às novas necessidades comerciais.

Idetificadores

FIAP

Identificadores são utiizados para:

- Nomear uma variável
- Seguir a convenção para momeação de variáveis:
  - Deve começar com uma letra
  - Pode incluir letras ou números
  - Pode incluir caracteres especiais, como sinal de dólar, sublinhado e cerquilha
  - Deve ter no máximo 30 caracteres
  - Não devem ser utilizadas palavras reservadas

1-8

Regras para Nomeação

Dois objetos podem ter o mesmo nome, contanto que eles sejam definidos em diferentes blocos. Onde eles coexistirem, apenas o objeto declarado no bloco atual pode ser usado. Você não deve escolher o nome (identificador) para uma variável igual ao nome das colunas de tabela usado no bloco. Se as variáveis PL/SQL ocorrerem nas instruções SQL e tiverem o mesmo nome que uma coluna, o Oracle Server supõe que seja a coluna que esteja sendo referenciada.

Embora o código de exemplo no slide funcione, o código criado usando o mesmo nome para uma tabela de banco de dados e para uma variável não é fácil de ler ou manter.

Considere a adoção de uma convenção de nomeação de vários objetos como o seguinte exemplo. O uso de v_ as como um prefixo que representa uma *variável* e g_ representando uma variável *global* impede conflitos de nomeação com os objetos do banco de dados.

Observação: Os identificadores não devem ter mais de 30 caracteres. O primeiro caracter deve ser uma letra; os restantes podem ser letras, números ou símbolos especiais..

Manipulação de variáveis PL/SQL

FIAP

As variáveis são:

- Declaradas e iniciadas na seção declare
- Novos valores usados e atribuídos na seção executável
- Podem ser passadas como parâmetros para subprogramas PL/SQL
- Utilizadas para armazenar a saída de um subprograma PL/SQL

Declarando e Inicializando Variáveis PL/SQL

FIAP

Sintaxe:

```
identifier [CONSTANT] datatype [NOT NULL]
     [:= | DEFAULT expr];
```

Exemplos:

```
DECLARE
  emp_hiredate    DATE;
  emp_deptno      NUMBER(2) NOT NULL := 10;
  location        VARCHAR2(13) := 'Atlanta';
  c_comm          CONSTANT NUMBER := 1400;
```

1-10

Declarando Variáveis PL/SQL

Você precisa declarar todos os identificadores PL/SQL na seção de declaração antes de referenciá-los no bloco PL/SQL. Você tem a opção de atribuir um valor inicial. Você não precisa atribuir um valor a uma variável para declará-la. Se você referenciar outras variáveis em uma declaração, você deverá estar certo de declará-las separadamente em uma instrução anterior.

Na sintaxe:

identificador é o nome da variável

CONSTANT restringe as variáveis para que o seu valor não possa ser alterado; as constantes devem ser inicializadas

tipo de dados são tipos de dados escalares, compostos, referenciais ou LOB

NOT NULL restringe a variável para que ela contenha um valor (variáveis NOT NULL devem ser inicializadas.)

expr é uma expressão PL/SQL que pode ser uma literal, uma outra variável ou uma expressão que envolve operadores e funções

Declarando e Inicializando Variáveis PL/SQL

FIAP

1

```
SET SERVEROUTPUT ON
DECLARE
  Myname VARCHAR2(20);
BEGIN
  DBMS_OUTPUT.PUT_LINE('My name is: '||Myname);
  Myname := 'John';
  DBMS_OUTPUT.PUT_LINE('My name is: '||Myname);
END;
/
```

2

```
SET SERVEROUTPUT ON
DECLARE
  Myname VARCHAR2(20):= 'John';
BEGIN
  Myname := 'Steven';
  DBMS_OUTPUT.PUT_LINE('My name is: '||Myname);
END;
/
```

1-11

Atribuindo Valores às Variáveis

Para atribuir ou reatribuir um valor a uma variável, crie uma instrução de atribuição PL/SQL. Você deve nomear explicitamente a variável para receber o novo valor à esquerda do operador de atribuição (:=).

Na sintaxe:

Uma outra maneira de atribuir valores às variáveis é selecionar ou trazer valores de banco de dados para dentro delas. O exemplo a seguir calcula um bônus de 10% ao seleciona o salário do funcionário:

```
SQL> SELECT     sal * 0.10
  2 INTO        v_bonus
  3 FROM        emp
  4 WHERE       empno = 7369;
```

Observação: Para atribuir um valor em uma variável do banco de dados, use uma instrução SELECT ou FETCH.

Diretrizes para declaração e inicialização variáveis PL/SQL

- Siga as convenções de nomenclatura.
- Use nomes significativos para variáveis.
- Inicie as variáveis definidas como NOT NULL e CONSTANT.
- Inicie as variáveis com o operador de atribuição (:=) ou com a palavra-chave DEFAULT:

```
Myname VARCHAR2(20):='John';
```

```
Myname VARCHAR2(20) DEFAULT 'John';
```

- Declare um identificador por linha para melhora a legibilidade e a manutenção de código.

1-12

Diretrizes

A expressão atribuída pode ser uma literal, uma outra variável ou uma expressão que envolve operadores e funções.

- Nomeie o identificador de acordo com as mesmas regras usadas por objetos SQL.
- Você poderá usar convenções de nomeação — por exemplo, *v_name* para representar uma variável e *c_name* e representar uma variável constante.
- Inicialize a variável em uma expressão com o operador de atribuição (:=) ou, equivalentemente, com a palavra reservada DEFAULT. Se você não atribuir um valor inicial, a nova variável conterá NULL por default até que você o atribua mais tarde.
- Se você usar a restrição NOT NULL, você deve atribuir um valor.
- A declaração de apenas um identificador por linha torna o código mais fácil de ler e de manter.
- Em declarações constantes, a palavra-chave CONSTANT deve preceder o especificador de tipo. A declaração a seguir nomeia a constante NUMBER, subtipo REAL e atribui o valor de 50000 à constante. Uma constante deve ser inicializada em sua declaração; senão, você obterá uma mensagem de erro de compilação quando a declaração for elaborada (compilada).

```
v_sal CONSTANT REAL := 50000.00;
```

Diretrizes para declaração e inicialização variáveis PL/SQL

FIAP

- Evite usar nomes de coluna como identificadores.

```
DECLARE
  employee_id  NUMBER(6);
BEGIN
  SELECT    employee_id
  INTO      employee_id
  FROM      employees
  WHERE     last_name = 'Kochhar';
END;
/
```

- Use a restrição NOT NULL quando a variável tiver que conter um valor.

Declarando Variáveis Escalares

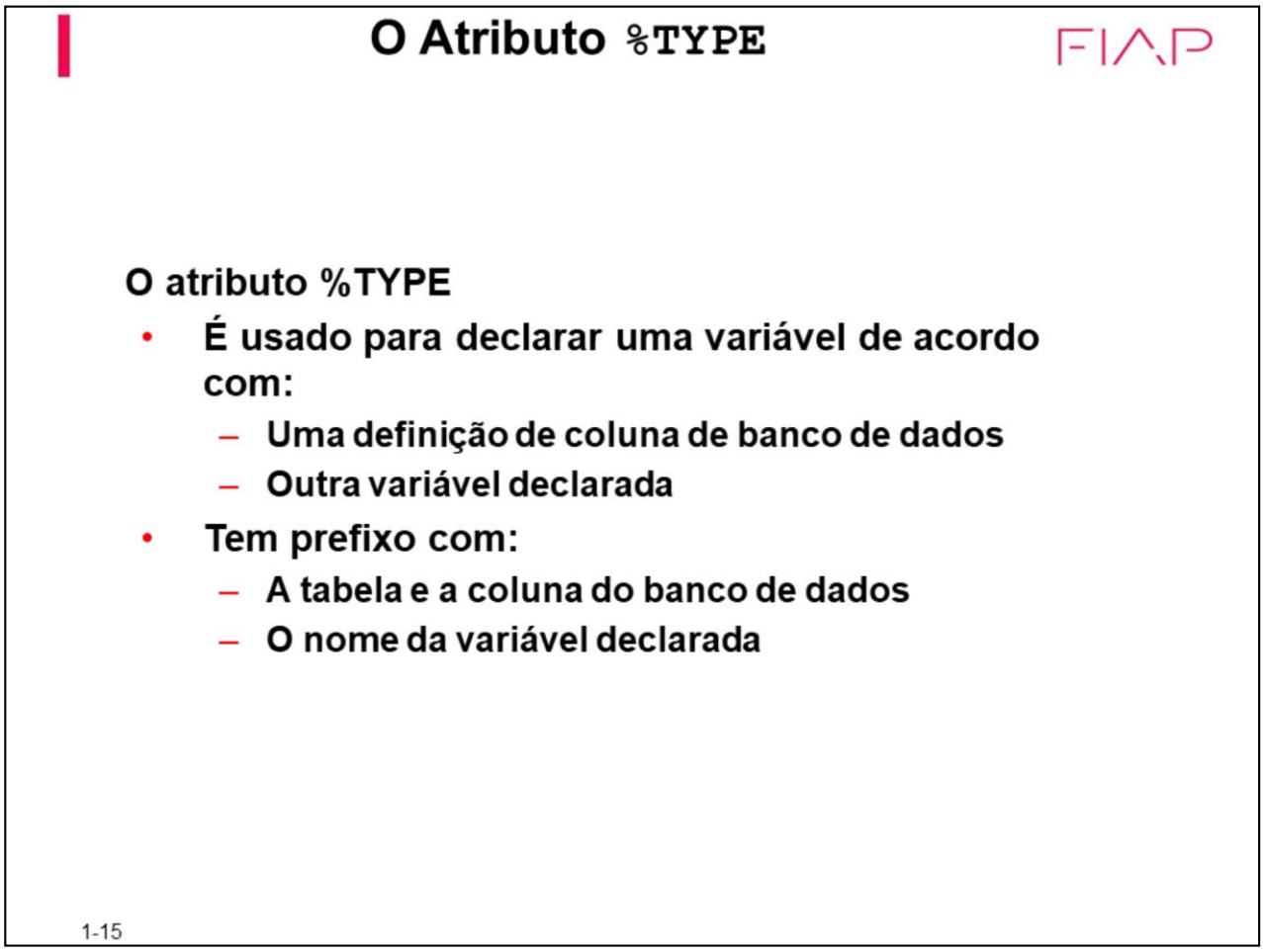
FIAP

Examples:

```
DECLARE
  emp_job          VARCHAR2(9);
  count_loop       BINARY_INTEGER := 0;
  dept_total_sal   NUMBER(9,2) := 0;
  orderdate        DATE := SYSDATE + 7;
  c_tax_rate       CONSTANT NUMBER(3,2) := 8.25;
  valid            BOOLEAN NOT NULL := TRUE;
  ...
```


Tipos de Dados Escalares Básicos

Tipo de Dados	Descrição
VARCHAR2 (*maximum_length*)	Tipo básico para dados de caractere de tamanho variável com até 32.767 bytes. Não há tamanho default para as constantes e variáveis VARCHAR2.
NUMBER [(*precisão, escala*)]	Tipo básico para números fixos e de ponto flutuante.
DATE	Tipo básico para datas e horas. Valores DATE incluem a hora do dia em segundos desde a meia-noite. A faixa para datas é entre 4712 A.C. e 9999 D.C.
CHAR [(*maximum_length*)]	Tipo básico para dados de caractere de tamanho fixo até 32.767 bytes. Se você não especificar um *comprimento_máx*, o tamanho default será definido como 1.
LONG	Tipo básico para dados de caracter de tamanho variável até 32.760 bytes. A largura máxima de uma coluna de banco de dados LONG é de 2.147.483.647 bytes.



O Atributo %TYPE

Ao declarar variáveis PL/SQL para armazenar os valores de coluna, você deverá garantir que a variável seja de precisão e tipo de dados corretos. Caso contrário, uma mensagem de erro PL/SQL ocorrerá durante a execução.

Em vez de embutir no código o tipo de dados e a precisão de uma variável, você poderá usar o atributo %TYPE para declarar uma variável de acordo com uma outra coluna de banco de dados ou variável declarada anteriormente. O atributo %TYPE é usado com mais freqüência quando o valor armazenado na variável for derivado de uma tabela no banco de dados ou se ocorre alguma gravação na variável. Para usar o atributo no lugar do tipo de dados necessário na declaração da variável, crie um prefixo para ele com o nome da coluna e da tabela de banco de dados. Se estiver referenciando uma variável declarada anteriormente, crie um prefixo para o nome da variável relativa ao atributo.

O código PL/SQL determina o tipo de dados e o tamanho da variável quando o bloco for compilado, de modo que ele seja sempre compatível com a coluna usada para preenche-lo. Isso definitivamente é uma vantagem para criar e manter o código, porque não há necessidade de se preocupar com alterações no tipo de dados da

coluna feitos no nível de banco de dados. Você poderá também declarar uma variável de acordo com uma outra declarada anteriormente pela prefixação do nome da variável relativa ao atributo

Declarando variáveis com o atributo %TYPE

FIAP

Sintaxe:

```
identifier     table.column_name%TYPE;
```

Examplos:

```
...
  emp_lname       employees.last_name%TYPE;
  balance         NUMBER(7,2);
  min_balance     balance%TYPE := 1000;
...
```

1-16

Declarando Variáveis com o Atributo %TYPE

Declare as variáveis para armazenar o nome de um funcionário.

```
...
v_ename                 emp.ename%TYPE;
...
```

Declare as variáveis para armazenar o saldo de uma conta corrente bancária, assim como um saldo mínimo, que inicie em 10.

```
...
v_balance               NUMBER(7,2);
v_min_balance           v_balance%TYPE := 10;
...
```

Uma restrição da coluna NOT NULL não se aplica a variáveis declaradas usando %TYPE. Por isso, se você declarar uma variável que estiver usando o atributo %TYPE usando uma coluna de banco de dados definida como NOT NULL, você poderá atribuir um valor NULL à variável.

Declarando Variáveis Booleanas

FIAP

- Somente os valores TRUE, FALSE, e NULL podem ser atribuídos a uma variável booleana.
- As expressões condicionais usam os operadores lógicos AND, OR e operador unário NOT para verificar os valores das variáveis.
- As variáveis sempre produzem TRUE, FALSE, ou NULL.
- As expressões aritméticas, de caractere e de data podem ser usadas para retornar um valor booleano.

1-17

Declarando Variáveis Booleanas

Com o código PL/SQL você poderá comparar variáveis em instruções SQL e procedurais. Essas comparações, chamadas *expressões booleanas*, consistem em expressões simples ou complexas separadas por operadores relacionais. Em uma instrução SQL, você poderá usar expressões booleanas para especificar as linhas em uma tabela que são afetadas por uma instrução. Em instruções procedurais, as expressões booleanas são a base para o controle condicional. NULL significa um valor ausente, inaplicável ou desconhecido.

Exemplos

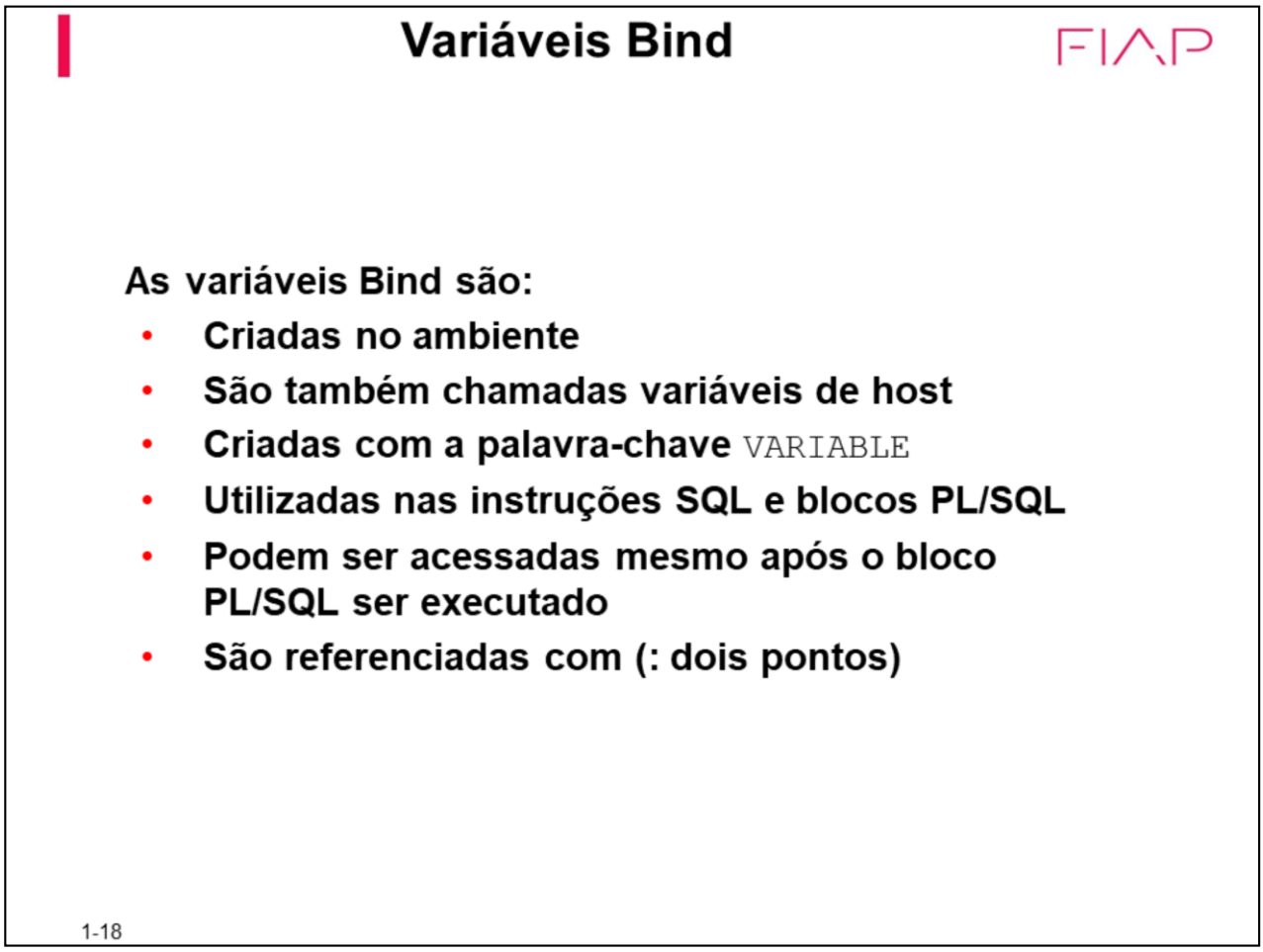
```
v_sal1 := 50000;
v_sal2 := 60000;
```

A expressão a seguir produz TRUE:

```
v_sal1 < v_sal2
```

Declare e inicialize uma variável booleana:

```
v_comm_sal BOOLEAN := (v_sal1 < v_sal2);
```



Variáveis de Ligação

Uma variável de ligação é uma variável que você declara em um ambiente de host e usa para passar valores de tempo de execução, número ou caractere, para ou de um ou mais programas PL/SQL, os quais podem usá-la como usariam qualquer outra variável. Você poderá referenciar variáveis declaradas em ambientes de host ou chamada em instruções PL/SQL, a não ser que a instrução esteja em um procedimento, função ou pacote. Isso inclui as variáveis de linguagem declaradas em programas do pré-compilador, campos de tela em aplicações de Form do Oracle Developer e as variáveis de ligação SQL*Plus.

Criando Variáveis de Ligação

Para declarar uma variável de ligação no ambiente SQL*Plus, você deve usar o comando VARIABLE. Por exemplo, você poderá declarar uma variável de tipo NUMBER e VARCHAR2 como se segue:

```
VARIABLE return_code NUMBER
VARIABLE return_msg VARCHAR2(30)
```

Imprimindo Bind Variables

FIAP

Exemplo:

```
VARIABLE emp_salary NUMBER
BEGIN
   SELECT salary  INTO :emp_salary
   FROM  employees WHERE employee_id = 178;
END;
/
PRINT emp_salary
```

1-19

Exibindo Variáveis de Ligação

Para exibir o valor atual das variáveis de ligação no ambiente SQL*Plus, você deverá usar o comando PRINT. Entretanto, PRINT não poderá ser usado em um bloco PL/SQL como um comando. O exemplo a seguir ilustra um comando PRINT:

EMP_SALARY
7000

```
...
SQL> PRINT g_n
```

FIRST_NAME	LAST_NAME
Oliver	Tuvault
Sarath	Sewall
Kimberely	Grant

Imprimindo Bind Variables

FIAP

Example:

```
VARIABLE emp_salary NUMBER
SET AUTOPRINT ON
BEGIN
   SELECT salary  INTO :emp_salary
   FROM  employees WHERE employee_id = 178;
END;
/
```

1-20

Variáveis de ligação (continuação)

Use o comando SET AUTOPRINT ON para exibir automaticamente as variáveis de ligação usadas em um bloco PL/SQL bem-sucedido.

Variáveis de Substituição

FIAP

- São usadas para solicitar ao usuário uma entrada em tempo de execução
- São referenciadas dentro de um bloco PL/ QL com um (& e comercial)
- São usados para evitar hard coding em tempo de execução

```
VARIABLE emp_salary NUMBER
SET AUTOPRINT ON
DECLARE
   empno NUMBER(6):=&empno;
BEGIN
  SELECT salary
  INTO :emp_salary
  FROM  employees
  WHERE employee_id = empno;
END;
/
```

Solicitar Variáveis de Substituição

FIAP

```
SET VERIFY OFF
VARIABLE emp_salary NUMBER
ACCEPT empno PROMPT 'Please enter a valid employee
number: '
SET AUTOPRINT ON
DECLARE
  empno NUMBER(6):= &empno;
BEGIN
  SELECT salary  INTO :emp_salary  FROM  employees
  WHERE employee_id = empno;
END;
/
```

Uilizando o DEFINE

FIAP

Example:

```
SET VERIFY OFF
DEFINE lname= Urman
DECLARE
  fname VARCHAR2(25);
BEGIN
  SELECT first_name INTO fname FROM employees
  WHERE last_name='&lname';
END;
/
```

1-23

Usando o DEFINE para a variável de usuário

O comando DEFINE especifica uma variável de usuário e atribui-lhe um valor CHAR. Você pode definir somente variáveis de tipo de dados CHAR. Mesmo que você digite o número 50000, será atribuído um valor CHAR a uma variável constituída pelos caracteres, 5,0,0,0 e 0. Você pode fazer referência a essas variáveis com um “&" .

Exercício: Visão Geral

FIAP

Esta prática abrange os seguintes tópicos:

- Determinando identificadores válidos
- Determinando declarações de variáveis válidas
- Declarando variáveis dentro de um bloco anônimo
- Utilizando o atributo `%TYPE` para declarar variáveis
- Declarando e imprimindo uma bind variable
- Executando um bloco PL/SQL

Exercícios

1. Identifique se os nomes dos identificadores são válidos ou inválidos:
   a. `today`
   b. `last_name`
   c. `today's_date`
   d. `Number_of_days_in_February_this_year`
   e. `Isleap$year`
   f. `#number`
   g. `NUMBER#`
   h. `number1to7`

2. Identifique se a declaração de variável e a inicialização é válida ou inválida:
   a. printer_name constant `VARCHAR2(10);`
   b. `deliver_to` `VARCHAR2(10)`:=Johnson;
   c. `by_when` `DATE`:= `SYSDATE`+1;

3. Examine o seguinte bloco anônimo e escolha a resposta correta apropriada.

```
SET SERVEROUTPUT ON
DECLARE
  fname VARCHAR2(20);
  lname VARCHAR2(15) DEFAULT 'fernandez';
BEGIN
  DBMS_OUTPUT.PUT_LINE( FNAME ||' ' ||lname);
END;
/
```

   a. O bloco será executado com sucesso e irá imprimir 'fernandez'
   b. O bloco retornará um erro porque a variável fname é declarada sem ser iniciada.
   c. O bloco será executado com sucesso e irá imprimir 'null fernandez'. (null significa ausência de valor)
   d. O bloco retornará um erro porque você não pode usar a palavra-chave DEFAULT para inicializar uma variável do tipo VARCHAR2.
   e. O bloco retornará um erro porque a variável FNAME não foi declarada.

Exercícios (Continuação)

4. Crie um bloco anônimo. Carregue o script da aula 01 lab_01_02_soln.sql, que você criou na pergunta 2 do exercício 1.
   a. Adicione a seção DECLARE nesse bloco PL/SQL. Na seção DECLARE declare as seguintes variáveis:
      1. `Variável today do tipo DATE. Inicializar today com SYSDATE`
      2. `Variável tomorrow do tipo today. Use o atributo% TYPE para declarar essa variável.`
   b. Na seção executável inicialize a variável tomorrow com uma expressão que calcula a data de tomorrow (adicione um ao valor de today). Exiba o valor de today e tomorrow após de imprimir 'Hello World‘
   c. Execute e salve este script como lab_02_04_soln.sql. Um exemplo da saída é mostrada a seguir.

```
Hello World
TODAY IS : 12-JAN-04
TOMORROW IS : 13-JAN-04
PL/SQL procedure successfully completed.
```

Exercícios (Continuação)

5. Edite o script lab_02_04_soln.sql.
    a. Crie as bind variables basic_percent e pf_percent do tipo NUMBER.
    b. Na seção executável do bloco PL/SQL atribua os valores 45 e 12 a basic_percent e pf_percent, respectivamente.
    c. Encerre o bloco PL/SQL com "/" e exiba o valor das bind variables utilizando o comando PRINT.
    d. Execute e salve seu arquivo de script como lab_02_05_soln.sql.

    Na seção executável do bloco PL/SQL atribua os valores 45 e 12 a basic_percent e pf_percent, respectivamente.
    Encerre o bloco PL/SQL com "/" e exibir o valor das variáveis de ligação usando o comando PRINT.
    Execute e salve seu arquivo de script como lab_02_05_soln.sql.